Tiva™ TM4C1231H6PM 微控制器

数据手册

DS-TM4C1231H6PM-15553.2700
ZHCSC78

版权 © 2007-2013
Texas Instruments Incorporated

版权

版权 © 2007-2013 Texas Instruments Incorporated。Tiva 和 TivaWare 均为 Texas Instruments Incorporated 的注册商标。ARM 和 Thumb 是 ARM 公司的注册商标，Cortex 是 ARM 公司的商标。所有其他商标均属于其所有者的财产。

产品数据信息为发布时的信息。根据 Texas Instruments 的标准条款，产品符合规格。生产过程中没有必要对产品的所有参数进行测试。

请留意，此数据手册最后部分包含 Texas Instruments 半导体产品的可用性、标准保修期和关键任务应用的重要信息通告以及相关的免责声明。

Texas Instruments Incorporated
108 Wild Basin, Suite 350
Austin, TX 78746
http://www.ti.com/tm4c
http://www-k.ext.ti.com/sc/technical-support/product-information-centers.htm

目录

关于本文档 31
受众 31
关于本手册 31
相关文档 31
文档约定 32
1 结构概述 34
1.1 Tiva™ C 系列 概述 34
1.2 TM4C1231H6PM 微控制器概述 34
1.3 TM4C1231H6PM 微控制器特性 37
1.3.1 ARM Cortex-M4F 处理器核心 37
1.3.2 片上存储器 39
1.3.3 串行通讯外设 40
1.3.4 系统集成 43
1.3.5 模拟 49
1.3.6 JTAG 和 ARM 串行线调试 50
1.3.7 封装和温度 51
1.4 TM4C1231H6PM 微控制器硬件细节 51
1.5 开发套件 51
1.6 支持信息 52
2 Cortex-M4F 处理器 53
2.1 结构框图 54
2.2 概述 55
2.2.1 系统级接口 55
2.2.2 集成的可配置调试 55
2.2.3 跟踪端口的接口单元（TPIU） 55
2.2.4 Cortex-M4F 系统组件细节 56
2.3 编程模型 56
2.3.1 处理器模式和软件执行的权限级别 57
2.3.2 堆栈 57
2.3.3 寄存器映射 57
2.3.4 寄存器描述 59
2.3.5 异常和中断 75
2.3.6 数据类型 75
2.4 存储模型 75
2.4.1 内存区，类型和属性 77
2.4.2 内存访问存储系统顺序 77
2.4.3 存储器访问行为 78
2.4.4 存储器访问的软件顺序 78
2.4.5 位带区 79
2.4.6 数据保存 81
2.4.7 同步原语 82
2.5 异常模式 83
2.5.1 异常状态 83
2.5.2 异常类型 84
2.5.3 异常处理程序 87

2.5.4 向量表 87
2.5.5 异常优先级 88
2.5.6 中断优先级分组 89
2.5.7 异常进入和返回 89
2.6 故障处理 91
2.6.1 故障类型 92
2.6.2 故障扩大和硬件故障 92
2.6.3 故障状态寄存器和故障地址寄存器 93
2.6.4 死锁 93
2.7 电源管理 93
2.7.1 进入睡眠模式 93
2.7.2 从睡眠模式唤醒 94
2.8 指令集总结 94
3 Cortex-M4 外设 101
3.1 功能说明 101
3.1.1 系统定时器（SysTick） 101
3.1.2 嵌套式向量化中断控制器（NVIC） 102
3.1.3 系统控制模块 (SCB) 103
3.1.4 存储器保护单元 (MPU) 104
3.1.5 浮点单元 (FPU) 108
3.2 寄存器映射 111
3.3 系统定时器（SysTick）寄存器描述 114
3.4 NVIC寄存器描述 118
3.5 系统控制模块（SCB）寄存器描述 133
3.6 存储器保护单元（MPU）寄存器描述 161
3.7 浮点单元 (FPU) 寄存器描述 169
4 JTAG 接口 175
4.1 结构框图 176
4.2 信号描述 176
4.3 功能说明 177
4.3.1 JTAG 接口管脚 177
4.3.2 JTAG TAP 控制器 178
4.3.3 移位寄存器 179
4.3.4 操作注意事项 179
4.4 初始化和配置 182
4.5 寄存器描述 182
4.5.1 指令寄存器 (IR) 182
4.5.2 数据寄存器 183
5 系统控制 186
5.1 信号描述 186
5.2 功能说明 186
5.2.1 器件标识 186
5.2.2 复位控制 186
5.2.3 不可屏蔽的中断 191
5.2.4 功率控制 192
5.2.5 时钟控制 193
5.2.6 系统控制 199
5.3 初始化和配置 202

5.4 寄存器映射 202
5.5 系统控制寄存器描述 207
5.6 系统控制传统寄存器描述 367
6 系统异常模块 421
6.1 功能说明 421
6.2 寄存器映射 421
6.3 寄存器描述 421
7 休眠模块 429
7.1 结构框图 430
7.2 信号描述 430
7.3 功能说明 431
7.3.1 寄存器访问间隙 431
7.3.2 休眠时钟源 431
7.3.3 系统实现 433
7.3.4 电池管理 433
7.3.5 实时时钟 434
7.3.6 带备用电池的存储器 435
7.3.7 电源控制：使用 HIB 436
7.3.8 以 VDD3ON 模式管理电源 436
7.3.9 启动休眠 436
7.3.10 从休眠模式唤醒 436
7.3.11 仲裁性电源移除 437
7.3.12 中断和状态 437
7.4 初始化和配置 437
7.4.1 初始化 437
7.4.2 RTC 匹配功能（无休眠） 438
7.4.3 RTC 匹配/唤醒 438
7.4.4 外部唤醒 439
7.4.5 RTC 或外部唤醒 439
7.5 寄存器映射 439
7.6 寄存器描述 440
8 内部存储器 458
8.1 结构框图 458
8.2 功能说明 459
8.2.1 SRAM 459
8.2.2 ROM 460
8.2.3 Flash 存储器 461
8.2.4 EEPROM 466
8.3 寄存器映射 470
8.4 Flash 存储器寄存器描述 (Flash 控制偏移量) 472
8.5 EEPROM 寄存器描述（EEPROM 偏移量） 488
8.6 存储器寄存器描述 (系统控制偏移量) 504
9 微型直接存储器访问（μDMA） 512
9.1 结构框图 513
9.2 功能说明 513
9.2.1 通道分配 513
9.2.2 优先级 515

9.2.3 仲裁数目 515
9.2.4 请求类型 515
9.2.5 通道配置 516
9.2.6 传输模式 517
9.2.7 待传输数目及增量 526
9.2.8 外设接口 526
9.2.9 软件请求 526
9.2.10 中断及错误 526
9.3 初始化和配置 527
9.3.1 模块初始化 527
9.3.2 存储器到存储器传输的配置 527
9.3.3 外设简单发送的配置 529
9.3.4 外设乒乓接收的配置 530
9.3.5 通道分配的配置 532
9.4 寄存器映射 532
9.5 µDMA 通道控制结构体 534
9.6 µDMA 寄存器描述 541
10 通用输入/输入端口（GPIOs） 575
10.1 信号描述 575
10.2 功能说明 577
10.2.1 数据控制 579
10.2.2 中断控制 580
10.2.3 模式控制 581
10.2.4 确认控制 581
10.2.5 引脚(Pad)控制 582
10.2.6 标识 582
10.3 初始化和配置 582
10.4 寄存器映射 583
10.5 寄存器描述 585
11 通用定时器 628
11.1 结构框图 629
11.2 信号描述 630
11.3 功能说明 631
11.3.1 GPTM复位条件 632
11.3.2 定时器模式 632
11.3.3 等待触发模式 642
11.3.4 同步通用定时器模块 642
11.3.5 DMA 操作 643
11.3.6 访问连接的 16/32 位 GPTM 寄存器值 643
11.3.7 访问连接的 32/64 位宽 GPTM 寄存器值 644
11.4 初始化和配置 645
11.4.1 单次触发/周期定时器模式 645
11.4.2 实时时钟 (RTC) 模式 646
11.4.3 输入边沿计数模式下： 646
11.4.4 输入边沿定时模式 647
11.4.5 PWM 模式 648
11.5 寄存器映射 648
11.6 寄存器描述 649

12 看门狗定时器 692
12.1 结构框图 693
12.2 功能说明 693
12.2.1 寄存器访问间隙 694
12.3 初始化和配置 694
12.4 寄存器映射 694
12.5 寄存器描述 695
13 模-数转换器（ADC） 717
13.1 结构框图 718
13.2 信号描述 718
13.3 功能说明 719
13.3.1 采样序列发生器 719
13.3.2 模块控制 720
13.3.3 硬件采样平均电路 723
13.3.4 模-数转换器 724
13.3.5 差分采样 726
13.3.6 内部温度传感器 728
13.3.7 数字比较器 729
13.4 初始化和配置 733
13.4.1 模块初始化 733
13.4.2 采样序列发生器的配置 734
13.5 寄存器映射 734
13.6 寄存器描述 736
14 通用异步收发器（UART） 799
14.1 结构框图 800
14.2 信号描述 800
14.3 功能说明 801
14.3.1 发送/接收逻辑 801
14.3.2 波特率的产生 802
14.3.3 数据传输 802
14.3.4 串行红外（SIR） 803
14.3.5 对ISO 7816的支持 804
14.3.6 对调制解调器握手信号的支持 804
14.3.7 9 位 UART 模式 805
14.3.8 FIFO操作 806
14.3.9 中断信号 806
14.3.10 回送操作 807
14.3.11 DMA 操作 807
14.4 初始化和配置 807
14.5 寄存器映射 809
14.6 寄存器描述 810
15 同步串行接口（SSI） 856
15.1 结构框图 857
15.2 信号描述 857
15.3 功能说明 858
15.3.1 位速率的产生 858
15.3.2 FIFO操作 859
15.3.3 中断信号 859

15.3.4 帧格式 860
15.3.5 DMA 操作 867
15.4 初始化和配置 867
15.5 寄存器映射 869
15.6 寄存器描述 870
16 内部集成电路（I²C）接口 899
16.1 结构框图 900
16.2 信号描述 900
16.3 功能说明 900
16.3.1 I²C 总线功能概览 901
16.3.2 可用的速度模式 904
16.3.3 中断信号 906
16.3.4 回送操作 907
16.3.5 命令序列流程图 907
16.4 初始化和配置 915
16.4.1 将 I²C 模块配置为以主机身份传输单字节数据 915
16.4.2 将 I²C 主机配置为高速模式 916
16.5 寄存器映射 917
16.6 寄存器描述（I²C 主机） 918
16.7 寄存器描述（I²C 从机） 934
16.8 寄存器描述（I²C 状态和控制寄存器） 944
17 控制器局域网（CAN）模块 947
17.1 结构框图 948
17.2 信号描述 948
17.3 功能说明 949
17.3.1 初始化 950
17.3.2 基本操作 950
17.3.3 报文对象的发送 951
17.3.4 待发送报文对象的配置 951
17.3.5 待发送报文对象的刷新 952
17.3.6 已接收报文对象的接受 953
17.3.7 接收数据帧 953
17.3.8 接收远程帧 953
17.3.9 接收/发送优先级 954
17.3.10 接收报文对象的配置 954
17.3.11 已接收报文对象的处理 955
17.3.12 中断的处理 956
17.3.13 测试模式 957
17.3.14 位定时配置错误的注意事项 958
17.3.15 位时间与位速率 958
17.3.16 位定时参数的计算 960
17.4 寄存器映射 963
17.5 寄存器描述 965
18 模拟比较器 993
18.1 结构框图 994
18.2 信号描述 994
18.3 功能说明 995

18.3.1 内部参考电压编程 995
18.4 初始化和配置 997
18.5 寄存器映射 998
18.6 寄存器描述 998
19 管脚图 1007
20 信号表 1008
20.1 按管脚编号分类的信号 1008
20.2 按信号名称分类的信号 1014
20.3 按功能分类的信号 (GPIO 除外) 1019
20.4 GPIO 管脚和复用功能 1024
20.5 复用功能的可能的管脚赋值 1026
20.6 未用管脚的处理 1028
21 Electrical Characteristics 1029
21.1 Maximum Ratings 1029
21.2 Operating Characteristics 1030
21.3 Recommended Operating Conditions 1031
21.4 Load Conditions 1033
21.5 JTAG and Boundary Scan 1034
21.6 Power and Brown-Out 1036
21.6.1 VDDA Levels 1036
21.6.2 VDD Levels 1037
21.6.3 VDDC Levels 1038
21.6.4 VDD Glitches 1039
21.6.5 VDD Droop Response 1039
21.7 Reset 1041
21.8 On-Chip Low Drop-Out (LDO) Regulator 1043
21.9 Clocks 1044
21.9.1 PLL Specifications 1044
21.9.2 PIOSC Specifications 1045
21.9.3 Low-Frequency Internal Oscillator (LFIOSC) Specifications 1045
21.9.4 Hibernation Clock Source Specifications 1045
21.9.5 Main Oscillator Specifications 1046
21.9.6 System Clock Specification with ADC Operation 1049
21.10 Sleep Modes 1050
21.11 Hibernation Module 1052
21.12 Flash Memory and EEPROM 1053
21.13 Input/Output Pin Characteristics 1054
21.13.1 GPIO Module Characteristics 1054
21.13.2 Types of I/O Pins and ESD Protection 1054
21.14 Analog-to-Digital Converter (ADC) 1058
21.15 Synchronous Serial Interface (SSI) 1061
21.16 Inter-Integrated Circuit (I^2C) Interface 1064
21.17 Analog Comparator 1065
21.18 Current Consumption 1067
A 封装信息 1070
A.1 可订购器件 1070
A.2 型号标识 1071

A.3 封装图 1072
A.4 封装材料 1073

插图清单

图 1-1. Tiva™ TM4C1231H6PM 微控制器高级框图 36
图 2-1. CPU 结构图 54
图 2-2. TPIU方框图 56
图 2-3. Cortex-M4F 寄存器组 58
图 2-4. 位带映射 81
图 2-5. 数据保存 82
图 2-6. 向量表 88
图 2-7. 异常堆栈框 90
图 3-1. SRD 使用示例 106
图 3-2. FPU 寄存器块 109
图 4-1. JTAG 模块方框图 176
图 4-2. 测试访问端口状态机 179
图 4-3. IDCODE 寄存器格式 184
图 4-4. BYPASS 寄存器格式 184
图 4-5. 边界扫描寄存器格式 184
图 5-1. 基本 $\overline{RST}$ 配置 189
图 5-2. 延长上电复位时间的外部电路 189
图 5-3. 复位电路由开关控制 190
图 5-4. 功率结构 193
图 5-5. 主时钟树 195
图 5-6. 模块时钟选择 200
图 7-1. 休眠模块结构图 430
图 7-2. 使用晶振作为休眠模块的时钟源（单一电池源） 432
图 7-3. 在 VDD3ON 模式中使用专用振荡器作为休眠模块的时钟源 432
图 7-4. V_{DD} 和 V_{BAT} 使用稳压器 433
图 7-5. TRIM 值为 0x8002 时的计数器行为 435
图 7-6. TRIM 值为 0x7FFC 时的计数器行为 435
图 8-1. 内部存储器结构图 458
图 8-2. EEPROM 结构图 459
图 9-1. µDMA 结构图 513
图 9-2. 乒乓式 µDMA 数据会话的示例 519
图 9-3. 存储器散聚模式：创建及配置 521
图 9-4. 存储器散聚模式：µDMA 复制序列 522
图 9-5. 外设散聚模式：创建及配置 524
图 9-6. 外设散聚模式：µDMA 复制序列 525
图 10-1. 数字 I/O 口 578
图 10-2. 模拟/数字 I/O 口 579
图 10-3. GPIODATA 写实例 580
图 10-4. GPIODATA 读实例 580
图 11-1. GPTM 模块的结构图 629
图 11-2. 读取 RTC 值 636
图 11-3. 输入边沿计数模式实例，递减计数 637
图 11-4. 16 位输入边沿计时模式实例 639
图 11-5. 16-位PWM模式实例 640
图 11-6. CCP 输出，GPTMTnMATCHR > GPTMTnILR 641
图 11-7. CCP 输出，GPTMTnMATCHR = GPTMTnILR 641

图 11-8. CCP 输出，GPTMTnILR > GPTMTnMATCHR 642
图 11-9. 定时器菊花链 642
图 12-1. WDT模块的结构图 693
图 13-1. 两个 ADC 模块的连接结构图 718
图 13-2. ADC模块框图 718
图 13-3. ADC 采样相位 721
图 13-4. ADC 采样率倍增 722
图 13-5. 交错采样 722
图 13-6. 采样平均的实例 724
图 13-7. ADC 输入端等效框图 725
图 13-8. ADC 参考电压 725
图 13-9. ADC 转换结果 726
图 13-10. 差分电压表达式 728
图 13-11. 内部温度传感器特性 729
图 13-12. 低值带工作（CIC = 0x0） 731
图 13-13. 中值带工作（CIC = 0x1 ） 732
图 13-14. 高值带工作（CIC = 0x3） 733
图 14-1. UART模块的结构图 800
图 14-2. UART字符帧 802
图 14-3. IrDA 数据调制 804
图 15-1. SSI模块的结构图 857
图 15-2. TI 同步串行帧格式（单次传输） 860
图 15-3. TI 同步串行的帧格式（连续传输） 861
图 15-4. SPO = 0 和 SPH = 0 时的飞思卡尔 SPI 格式（单次传输） 862
图 15-5. SPO = 0 和 SPH = 0 时的飞思卡尔 SPI 格式 (连续传输) 862
图 15-6. SPO = 0、SPH = 1 时的飞思卡尔 SPI 帧格式 863
图 15-7. SPO = 1 和 SPH = 0 时的飞思卡尔 SPI 帧格式（单次传输） 864
图 15-8. SPO = 1 和 SPH = 0 时的飞思卡尔 SPI 帧格式（连续传输） 864
图 15-9. SPO = 1、SPH = 1 时的飞思卡尔 SPI 帧格式 865
图 15-10. MICROWIRE的帧格式 (单帧) 865
图 15-11. MICROWIRE的帧格式 (连续传输) 866
图 15-12. MICROWIRE 帧格式，SSInFss 输入建立和保持时间要求 867
图 16-1. I^2C 结构图 900
图 16-2. I^2C 总线配置 901
图 16-3. START 和 STOP 条件 901
图 16-4. 带 7 位地址的完整数据传输 902
图 16-5. 首字节的R/S位 902
图 16-6. I^2C 总线位传输过程中的数据有效性 902
图 16-7. 高速数据格式 906
图 16-8. 主机单次传输 908
图 16-9. 主机单次接收 909
图 16-10. 多数据字节的主机传输 910
图 16-11. 多数据字节的主机接收 911
图 16-12. 主机传输后以重复开始序列进行的主机接收 912
图 16-13. 主机接收后以重复开始序列进行的主机传输 913
图 16-14. 标准高速模式主机传输 914
图 16-15. 从机命令序列 915
图 17-1. CAN 控制器结构图 948

图 17-2. CAN 数据帧/远程帧 949
图 17-3. FIFO 缓冲区中的报文对象 956
图 17-4. CAN 的位时间 959
图 18-1. 模拟比较器模块的结构图 994
图 18-2. 比较单元的结构 995
图 18-3. 比较器内部参考结构 996
图 19-1. 64 管脚 LQFP 封装管脚图 1007
图 21-1. Load Conditions 1033
图 21-2. JTAG Test Clock Input Timing 1034
图 21-3. JTAG Test Access Port (TAP) Timing 1035
图 21-4. Power Assertions versus VDDA Levels 1037
图 21-5. Power and Brown-Out Assertions versus VDD Levels 1038
图 21-6. POK assertion vs VDDC 1039
图 21-7. POR-BOR0-BOR1 VDD Glitch Response 1039
图 21-8. POR-BOR0-BOR1 VDD Droop Response 1040
图 21-9. Digital Power-On Reset Timing 1041
图 21-10. Brown-Out Reset Timing 1041
图 21-11. External Reset Timing ($\overline{RST}$) 1042
图 21-12. Software Reset Timing 1042
图 21-13. Watchdog Reset Timing 1042
图 21-14. MOSC Failure Reset Timing 1042
图 21-15. Hibernation Module Timing 1052
图 21-16. ESD Protection on Fail-Safe Pins 1055
图 21-17. ESD Protection on Non-Fail-Safe Pins 1056
图 21-18. ADC Input Equivalency Diagram 1060
图 21-19. SSI Timing for TI Frame Format (FRF=01), Single Transfer Timing Measurement 1062
图 21-20. SSI Timing for MICROWIRE Frame Format (FRF=10), Single Transfer 1062
图 21-21. Master Mode SSI Timing for SPI Frame Format (FRF=00), with SPH=1 1063
图 21-22. Slave Mode SSI Timing for SPI Frame Format (FRF=00), with SPH=1 1063
图 21-23. I^2C Timing 1064
图 A-1. 器件型号说明 1070
图 A-2. TM4C1231H6PM 64 管脚 LQFP 封装图 1072
图 A-3. 64 管脚 LQFP PM 封装载带 1073
图 A-4. 64 管脚 LQFP PM 封装塑料圆盘 1074
图 A-5. 64 管脚 LQFP PM 封装用载带圆盘箱 1074

表格清单

表 1. 文档约定 32
表 1-1. TM4C1231H6PM 微控制器特性 35
表 2-1. 处理器模式、特权等级和堆栈使用摘要 57
表 2-2. 处理器 寄存器映射 58
表 2-3. PSR 寄存器组合 64
表 2-4. 存储器映射 75
表 2-5. 存储器访问行为 78
表 2-6. SRAM 存储器位带区 79
表 2-7. 外设存储器位带区 80
表 2-8. 异常类型 85
表 2-9. 中断信号 85
表 2-10. 异常返回行为 91
表 2-11. 故障 92
表 2-12. 故障状态寄存器和故障地址寄存器 93
表 2-13. Cortex-M4F 指令摘要 95
表 3-1. 内核外设寄存器区域 101
表 3-2. 存储器属性摘要 104
表 3-3. TEX、S、C 和 B 位域编码 107
表 3-4. 存储器属性编码对应的高速缓存策略 107
表 3-5. AP 位域编码 107
表 3-6. Stellaris Tiva™ C 系列 微控制器的存储器区属性 108
表 3-7. QNaN 和 SNaN 处理 110
表 3-8. 外设 寄存器映射 111
表 3-9. 中断优先级分组 141
表 3-10. SIZE 域数值示例 168
表 4-1. JTAG_SWD_SWO 信号 (64LQFP) 176
表 4-2. 上电复位或 RST 生效后的 JTAG 端口管脚状态 177
表 4-3. JTAG 指令寄存器命令 182
表 5-1. 系统控制; 时钟 信号 (64LQFP) 186
表 5-2. 复位源 187
表 5-3. 时钟源选项 194
表 5-4. 使用 SYSDIV 域可能实现的系统时钟频率 195
表 5-5. 使用 SYSDIV2 域可能实现的系统时钟频率示例 196
表 5-6. 当 DIV400=1 时可能实现的系统时钟频率示例 196
表 5-7. 系统控制 寄存器映射 203
表 5-8. 替代 RCC 域的 RCC2 域 227
表 6-1. 系统异常 寄存器映射 421
表 7-1. 休眠 信号 (64LQFP) 430
表 7-2. 休眠模块的时钟运行 438
表 7-3. 休眠模块 寄存器映射 439
表 8-1. Flash 存储器保护策略组合 462
表 8-2. 用户可编程的 Flash 存储器驻留寄存器 466
表 8-3. Flash 寄存器映射 470
表 9-1. µDMA 通道分配 514
表 9-2. 所支持的请求类型 515
表 9-3. 控制结构体的存储器映射 516

表 9-4. 通道控制结构体 516
表 9-5. µDMA 读操作实例：8 位外设 526
表 9-6. µDMA 中断分配 527
表 9-7. 第 30 号通道的通道控制结构体偏移量 528
表 9-8. 存储器传输示例的通道控制字配置 528
表 9-9. 第 7 号通道的通道控制结构体偏移量 529
表 9-10. 外设传输示例的通道控制字配置 529
表 9-11. 第 8 号通道的主控制结构体及副控制结构体偏移量 531
表 9-12. 外设乒乓接收示例的通道控制字配置 531
表 9-13. µDMA 寄存器映射 533
表 10-1. 具有非 0 复位值的 GPIO 管脚 576
表 10-2. GPIO 管脚和复用功能 (64LQFP) 576
表 10-3. GPIO 端口配置示例 583
表 10-4. GPIO 中断配置示例 583
表 10-5. 具有非 0 复位值的 GPIO 管脚 584
表 10-6. GPIO 触发 寄存器映射 584
表 10-7. 具有非 0 复位值的 GPIO 管脚 595
表 10-8. 具有非 0 复位值的 GPIO 管脚 601
表 10-9. 具有非 0 复位值的 GPIO 管脚 603
表 10-10. 具有非 0 复位值的 GPIO 管脚 606
表 10-11. 具有非 0 复位值的 GPIO 管脚 612
表 11-1. 可用的 CCP 管脚 629
表 11-2. 通用定时器 信号 (64LQFP) 630
表 11-3. 通用定时器功能 631
表 11-4. 单次触发或周期模式下启用定时器时的计数器值 633
表 11-5. 带预分频器的 16 位定时器配置 634
表 11-6. 带预分频器配置的 32 位定时器（配置为 32/64 位模式） 634
表 11-7. RTC 模式下启用定时器时的计数器值 634
表 11-8. 输入边沿计数模式下启用定时器时的计数器值 637
表 11-9. 输入事件计数模式下启用定时器时的计数器值 638
表 11-10. PWM 模式下启用定时器时的计数器值 639
表 11-11. GPTM 模式的超时动作 643
表 11-12. 定时器触发 寄存器映射 649
表 12-1. 看门狗定时器 寄存器映射 694
表 13-1. ADC 信号 (64LQFP) 719
表 13-2. 采样序列发生器的采样数和 FIFO 深度 719
表 13-3. 差分采样对 727
表 13-4. ADC 寄存器映射 734
表 14-1. UART 信号 (64LQFP) 800
表 14-2. 流控模式 805
表 14-3. UART 寄存器映射 809
表 15-1. SSI 信号 (64LQFP) 858
表 15-2. SSI 寄存器映射 869
表 16-1. I2C 信号 (64LQFP) 900
表 16-2. I^2C 主机定时器周期与速度模式示例 905
表 16-3. 高速模式下 I^2C 主机定时器周期的示例 906
表 16-4. 内部集成电路（I^2C）接口 寄存器映射 917
表 16-5. 为 I2CMCS[3:0] 域写入域解码 923

表 17-1. 控制器局域网 信号 (64LQFP) 948
表 17-2. 报文对象的配置 953
表 17-3. CAN 协议范围 959
表 17-4. CANBIT 寄存器值 960
表 17-5. CAN 寄存器映射 963
表 18-1. 模拟比较器触发 信号 (64LQFP) 994
表 18-2. 内部参考电压和 ACREFCTL 域值 996
表 18-3. 模拟比较器参考电压特性，V_{DDA} = 3.3V，EN= 1 且 RNG = 0 997
表 18-4. 模拟比较器参考电压特性，V_{DDA} = 3.3V，EN= 1 且 RNG = 1 997
表 18-5. 模拟比较器触发 寄存器映射 998
表 20-1. 默认为复用功能的 GPIO 管脚 1008
表 20-2. 按管脚编号分类的信号 1008
表 20-3. 按信号名称分类的信号 1014
表 20-4. 按功能分类的信号（GPIO 除外） 1019
表 20-5. GPIO 管脚和复用功能 1024
表 20-6. 复用功能的可能的管脚赋值 1026
表 20-7. 未用信号的连接（64 管脚 LQFP） 1028
表 21-1. Maximum Ratings 1029
表 21-2. ESD Absolute Maximum Ratings 1029
表 21-3. Temperature Characteristics 1030
表 21-4. Thermal Characteristics 1030
表 21-5. Recommended DC Operating Conditions 1031
表 21-6. Recommended GPIO Pad Operating Conditions 1031
表 21-7. GPIO Current Restrictions 1031
表 21-8. GPIO Package Side Assignments 1032
表 21-9. JTAG Characteristics 1034
表 21-10. Power-On and Brown-Out Levels 1036
表 21-11. Reset Characteristics 1041
表 21-12. LDO Regulator Characteristics 1043
表 21-13. Phase Locked Loop (PLL) Characteristics 1044
表 21-14. Actual PLL Frequency 1044
表 21-15. PIOSC Clock Characteristics 1045
表 21-16. Low-Frequency internal Oscillator Characteristics 1045
表 21-17. Hibernation Oscillator Input Characteristics 1045
表 21-18. Main Oscillator Input Characteristics 1046
表 21-19. Crystal Parameters 1047
表 21-20. Supported MOSC Crystal Frequencies 1048
表 21-21. System Clock Characteristics with ADC Operation 1049
表 21-22. Sleep Modes AC Characteristics 1050
表 21-23. Time to Wake with Respect to Low-Power Modes 1050
表 21-24. Hibernation Module Battery Characteristics 1052
表 21-25. Hibernation Module AC Characteristics 1052
表 21-26. Flash Memory Characteristics 1053
表 21-27. EEPROM Characteristics 1053
表 21-28. GPIO Module Characteristics 1054
表 21-29. Pad Voltage/Current Characteristics for Fail-Safe Pins 1055
表 21-30. Fail-Safe GPIOs that Require an External Pull-up 1056
表 21-31. Non-Fail-Safe I/O Pad Voltage/Current Characteristics 1056

表 21-32. ADC Electrical Characteristics 1058
表 21-33. SSI Characteristics 1061
表 21-34. I^2C Characteristics 1064
表 21-35. Analog Comparator Characteristics 1065
表 21-36. Analog Comparator Voltage Reference Characteristics 1065
表 21-37. Analog Comparator Voltage Reference Characteristics, V_{DDA} = 3.3V, EN= 1, and RNG = 0 1065
表 21-38. Analog Comparator Voltage Reference Characteristics, V_{DDA} = 3.3V, EN= 1, and RNG = 1 1066
表 21-39. Current Consumption 1067
表 A-1. 可订购的器件型号 1070

寄存器列表

Cortex-M4F 处理器 53
寄存器 1: Cortex 通用寄存器 0（R0） 60
寄存器 2: Cortex 通用寄存器 1（R1） 60
寄存器 3: Cortex 通用寄存器 2（R2） 60
寄存器 4: Cortex 通用寄存器 3（R3） 60
寄存器 5: Cortex 通用寄存器 4（R4） 60
寄存器 6: Cortex 通用寄存器 5（R5） 60
寄存器 7: Cortex 通用寄存器 6（R6） 60
寄存器 8: Cortex 通用寄存器 7（R7） 60
寄存器 9: Cortex 通用寄存器 8（R8） 60
寄存器 10: Cortex 通用寄存器 9（R9） 60
寄存器 11: Cortex 通用寄存器 10（R10） 60
寄存器 12: Cortex 通用寄存器 11（R11） 60
寄存器 13: Cortex 通用寄存器 12（R12） 60
寄存器 14: 堆栈指针（SP） 61
寄存器 15: 链接寄存器（LR） 62
寄存器 16: 程序计数器（PC） 63
寄存器 17: 程序状态寄存器（PSR） 64
寄存器 18: 优先级屏蔽寄存器（PRIMASK） 68
寄存器 19: 故障屏蔽寄存器（FAULTMASK） 69
寄存器 20: 基本优先级屏蔽寄存器（BASEPRI） 70
寄存器 21: 控制寄存器（CONTROL） 71
寄存器 22: 浮点状态控制（FPSC）寄存器 73
Cortex-M4 外设 101
寄存器 1: SysTick 控制及状态寄存器（STCTRL），偏移量 0x010 115
寄存器 2: SysTick 重载值寄存器（STRELOAD），偏移量 0x014 117
寄存器 3: SysTick 当前值寄存器（STCURRENT），偏移量 0x018 118
寄存器 4: 中断 0-31 置位启用寄存器（EN0），偏移量 0x100 119
寄存器 5: 中断 32-63 置位启用寄存器（EN1），偏移量 0x104 119
寄存器 6: 中断 64-95 置位启用寄存器（EN2），偏移量 0x108 119
寄存器 7: 中断 96-127 置位启用寄存器（EN3），偏移量 0x10C 119
寄存器 8: 中断 128-138 设置启用寄存器（EN4），偏移量 0x110 120
寄存器 9: 中断 0-31 清除启用寄存器（DIS0），偏移量 0x180 121
寄存器 10: 中断 32-63 清除启用寄存器（DIS1），偏移量 0x184 121
寄存器 11: 中断 64-95 清除启用寄存器（DIS2），偏移量 0x188 121
寄存器 12: 中断 96-127 清除启用寄存器（DIS3），偏移量 0x18C 121
寄存器 13: 中断 128-138 清除启用寄存器（DIS4），偏移量 0x190 122
寄存器 14: 中断 0-31 置位挂起寄存器（PEND0），偏移量 0x200 123
寄存器 15: 中断 32-63 置位挂起寄存器（PEND1），偏移量 0x204 123
寄存器 16: 中断 64-95 置位挂起寄存器（PEND2），偏移量 0x208 123
寄存器 17: 中断 96-127 置位挂起寄存器（PEND3），偏移量 0x20C 123
寄存器 18: 中断 128-138 置位挂起寄存器（PEND4），偏移量 0x210 124
寄存器 19: 中断 0-31 清除挂起寄存器（UNPEND0），偏移量 0x280 125
寄存器 20: 中断 32-63 清除挂起寄存器（UNPEND1），偏移量 0x284 125
寄存器 21: 中断 64-95 清除挂起寄存器（UNPEND2），偏移量 0x288 125

寄存器 22: 中断 96-127 清除挂起寄存器（UNPEND3），偏移量 0x28C 125
寄存器 23: 中断 128-138 清除挂起寄存器（UNPEND4），偏移量 0x290 126
寄存器 24: 中断 0-31 活动位寄存器（ACTIVE0），偏移量 0x300 127
寄存器 25: 中断 32-63 活动位寄存器（ACTIVE1），偏移量 0x304 127
寄存器 26: 中断 64-95 活动位寄存器（ACTIVE2），偏移量 0x308 127
寄存器 27: 中断 96-127 活动位寄存器（ACTIVE3），偏移量 0x30C 127
寄存器 28: 中断 128-138 活动位寄存器（ACTIVE4），偏移量 0x310 128
寄存器 29: 中断 0-3 优先级寄存器（PRI0），偏移量 0x400 129
寄存器 30: 中断 4-7 优先级寄存器（PRI1），偏移量 0x404 129
寄存器 31: 中断 8-11 优先级寄存器（PRI2），偏移量 0x408 129
寄存器 32: 中断 12-15 优先级寄存器（PRI3），偏移量 0x40C 129
寄存器 33: 中断 16-19 优先级寄存器（PRI4），偏移量 0x410 129
寄存器 34: 中断 20-23 优先级寄存器（PRI5），偏移量 0x414 129
寄存器 35: 中断 24-27 优先级寄存器（PRI6），偏移量 0x418 129
寄存器 36: 中断 28-31 优先级寄存器（PRI7），偏移量 0x41C 129
寄存器 37: 中断 32-35 优先级寄存器（PRI8），偏移量 0x420 129
寄存器 38: 中断 36-39 优先级寄存器（PRI9），偏移量 0x424 129
寄存器 39: 中断 40-43 优先级寄存器（PRI10），偏移量 0x428 129
寄存器 40: 中断 44-47 优先级寄存器（PRI11），偏移量 0x42C 129
寄存器 41: 中断 48-51 优先级寄存器（PRI12），偏移量 0x430 129
寄存器 42: 中断 52-55 优先级寄存器（PRI13），偏移量 0x434 129
寄存器 43: 中断 56-59 优先级寄存器（PRI14），偏移量 0x438 129
寄存器 44: 中断 60-63 优先级寄存器（PRI15），偏移量 0x43C 129
寄存器 45: 中断 64-67 优先级寄存器（PRI16），偏移量 0x440 131
寄存器 46: 中断 68-71 优先级寄存器（PRI17），偏移量 0x444 131
寄存器 47: 中断 72-75 优先级寄存器（PRI18），偏移量 0x448 131
寄存器 48: 中断 76-79 优先级寄存器（PRI19），偏移量 0x44C 131
寄存器 49: 中断 80-83 优先级寄存器（PRI20），偏移量 0x450 131
寄存器 50: 中断 84-87 优先级寄存器（PRI21），偏移量 0x454 131
寄存器 51: 中断 88-91 优先级寄存器（PRI22），偏移量 0x458 131
寄存器 52: 中断 92-95 优先级寄存器（PRI23），偏移量 0x45C 131
寄存器 53: 中断 96-99 优先级寄存器（PRI24），偏移量 0x460 131
寄存器 54: 中断 100-103 优先级寄存器（PRI25），偏移量 0x464 131
寄存器 55: 中断 104-107 优先级寄存器（PRI26），偏移量 0x468 131
寄存器 56: 中断 108-111 优先级寄存器（PRI27），偏移量 0x46C 131
寄存器 57: 中断 112-115 优先级寄存器（PRI28），偏移量 0x470 131
寄存器 58: 中断 116-119 优先级寄存器（PRI29），偏移量 0x474 131
寄存器 59: 中断 120-123 优先级寄存器（PRI30），偏移量 0x478 131
寄存器 60: 中断 124-127 优先级寄存器（PRI31），偏移量 0x47C 131
寄存器 61: 中断 128-131 优先级寄存器（PRI32），偏移量 0x480 131
寄存器 62: 中断 132-135 优先级寄存器（PRI33），偏移量 0x484 131
寄存器 63: 中断 136-138 优先级寄存器（PRI34），偏移量 0x488 131
寄存器 64: 软件触发中断寄存器（SWTRIG），偏移量 0xF00 133
寄存器 65: 辅助控制寄存器（ACTLR），偏移量 0x008 134
寄存器 66: CPU ID 基础寄存器（CPUID），偏移量 0xD00 136
寄存器 67: 中断控制及状态寄存器（INTCTRL），偏移量 0xD04 137
寄存器 68: 向量表寄存器（VTABLE），偏移量 0xD08 140
寄存器 69: 应用程序中断及复位控制寄存器（APINT），偏移量 0xD0C 141

寄存器 70: 系统控制寄存器（SYSCTRL），偏移量 0xD10 143
寄存器 71: 配置及控制寄存器（CFGCTRL），偏移量 0xD14 145
寄存器 72: 系统处理程序优先级寄存器 1（SYSPRI1），偏移量 0xD18 147
寄存器 73: 系统处理程序优先级寄存器 2（SYSPRI2），偏移量 0xD1C 148
寄存器 74: 系统处理程序优先级寄存器 3（SYSPRI3），偏移量 0xD20 149
寄存器 75: 系统处理程序控制及状态寄存器（SYSHNDCTRL），偏移量 0xD24 150
寄存器 76: 可配置故障状态寄存器（FAULTSTAT），偏移量 0xD28 153
寄存器 77: 硬故障状态寄存器（HFAULTSTAT），偏移量 0xD2C 159
寄存器 78: 存储器管理故障地址寄存器（MMADDR），偏移量 0xD34 160
寄存器 79: 总线故障地址寄存器（FAULTADDR），偏移量 0xD38 161
寄存器 80: MPU 类型寄存器（MPUTYPE），偏移量 0xD90 162
寄存器 81: MPU 控制寄存器（MPUCTRL），偏移量 0xD94 163
寄存器 82: MPU 区编号寄存器（MPUNUMBER），偏移量 0xD98 165
寄存器 83: MPU 区基地址寄存器（MPUBASE），偏移量 0xD9C 166
寄存器 84: MPU 区基地址别名寄存器 1（MPUBASE1），偏移量 0xDA4 166
寄存器 85: MPU 区基地址别名寄存器 2（MPUBASE2），偏移量 0xDAC 166
寄存器 86: MPU 区基地址别名寄存器 3（MPUBASE3），偏移量 0xDB4 166
寄存器 87: MPU 区属性和大小寄存器（MPUATTR），偏移量 0xDA0 168
寄存器 88: MPU 区属性和大小别名寄存器 1（MPUATTR1），偏移量 0xDA8 168
寄存器 89: MPU 区属性和大小别名寄存器 2（MPUATTR2），偏移量 0xDB0 168
寄存器 90: MPU 区属性和大小别名寄存器 3（MPUATTR3），偏移量 0xDB8 168
寄存器 91: 协处理器访问控制（CPAC），偏移量 0xD88 170
寄存器 92: 浮点上下文控制（FPCC），偏移量 0xF34 171
寄存器 93: 浮点上下文访问（FPCA），偏移量 0xF38 173
寄存器 94: 浮点默认状态控制（FPDSC），偏移量 0xF3C 174
系统控制 186
寄存器 1: 器件标识 0（DID0），偏移量 0x000 208
寄存器 2: 器件标识寄存器 1（DID1），偏移量 0x004 210
寄存器 3: 掉电复位控制（PBORCTL），偏移量 0x030 212
寄存器 4: 原始中断状态（RIS），偏移量 0x050 213
寄存器 5: 中断屏蔽控制（IMC），偏移量 0x054 215
寄存器 6: 屏蔽的中断状态和清除（MISC），偏移量 0x058 217
寄存器 7: 复位原因（RESC），偏移量 0x05C 219
寄存器 8: 运行模式时钟配置（RCC），偏移量 0x060 221
寄存器 9: GPIO 高性能总线控制（GPIOHBCTL），偏移量 0x06C 225
寄存器 10: 运行模式时钟配置 2（RCC2），偏移量 0x070 227
寄存器 11: 主振荡器控制（MOSCCTL），偏移量 0x07C 230
寄存器 12: 深度睡眠时钟配置（DSLPCLKCFG），偏移量 0x144 231
寄存器 13: 系统属性寄存器（SYSPROP），偏移量 0x14C 233
寄存器 14: 精确内部振荡器校准（PIOSCCAL），偏移量 0x150 235
寄存器 15: 精确内部振荡器统计寄存器（PIOSCSTAT），偏移量 0x154 236
寄存器 16: PLL 频率寄存器 0（PLLFREQ0），偏移量 0x160 237
寄存器 17: PLL 频率寄存器 1（PLLFREQ1），偏移量 0x164 238
寄存器 18: PLL 状态寄存器（PLLSTAT），偏移量 0x168 239
寄存器 19: 睡眠功率配置寄存器（SLPPWRCFG），偏移量 0x188 240
寄存器 20: 深度睡眠功率配置寄存器（DSLPPWRCFG），偏移量 0x18C 241
寄存器 21: LDO 睡眠功率控制寄存器（LDOSPCTL），偏移量 0x1B4 242
寄存器 22: LDO 睡眠功率校准寄存器（LDOSPCAL），偏移量 0x1B8 244

寄存器 23: LDO 深度睡眠功率控制寄存器（LDODPCTL），偏移量 0x1BC 245
寄存器 24: LDO 深度睡眠功率校准寄存器（LDODPCAL），偏移量 0x1C0 247
寄存器 25: 睡眠/深度睡眠功率模式状态寄存器（SDPMST），偏移量 0x1CC 248
寄存器 26: 看门狗定时器外设存在寄存器（PPWD），偏移量 0x300 251
寄存器 27: 16/32 位通用定时器外设存在寄存器（PPTIMER），偏移量 0x304 252
寄存器 28: 通用输入/输出外设存在寄存器（PPGPIO），偏移量 0x308 254
寄存器 29: 微型直接存储器访问外设存在寄存器（PPDMA），偏移量 0x30C 257
寄存器 30: 休眠外设存在寄存器（PPHIB），偏移量 0x314 258
寄存器 31: 通用异步收发器外设存在寄存器（PPUART），偏移量 0x318 259
寄存器 32: 同步串行接口外设存在寄存器（PPSSI），偏移量 0x31C 261
寄存器 33: 内部集成电路外设存在寄存器（PPI2C），偏移量 0x320 262
寄存器 34: 通用串行总线外设存在寄存器（PPUSB），偏移量 0x328 264
寄存器 35: 控制器局域网外设存在寄存器（PPCAN），偏移量 0x334 265
寄存器 36: 模数转换器外设存在寄存器（PPADC），偏移量 0x338 266
寄存器 37: 模拟比较器外设存在寄存器（PPACMP），偏移量 0x33C 267
寄存器 38: 脉宽调解器外设存在寄存器（PPPWM），偏移量 0x340 268
寄存器 39: 正交编码器接口外设存在寄存器（PPQEI），偏移量 0x344 269
寄存器 40: EEPROM 外设存在寄存器（PPEEPROM），偏移量 0x358 270
寄存器 41: 32/64 位宽通用定时器外设存在寄存器（PPWTIMER），偏移量 0x35C 271
寄存器 42: 看门狗定时器软件复位寄存器（SRWD），偏移量 0x500 273
寄存器 43: 16/32 位通用定时器软件复位寄存器（SRTIMER），偏移量 0x504 274
寄存器 44: 通用输入/输出软件复位寄存器（SRGPIO），偏移量 0x508 276
寄存器 45: 微型直接存储器访问软件复位寄存器（SRDMA），偏移量 0x50C 278
寄存器 46: 休眠软件复位寄存器（SRHIB），偏移量 0x514 279
寄存器 47: 通用异步收发器软件复位寄存器（SRUART），偏移量 0x518 280
寄存器 48: 同步串行接口软件复位寄存器（SRSSI），偏移量 0x51C 282
寄存器 49: 内部集成电路软件复位寄存器（SRI2C），偏移量 0x520 284
寄存器 50: 控制器局域网软件复位寄存器（SRCAN），偏移量 0x534 286
寄存器 51: 模数转换器软件复位寄存器（SRADC），偏移量 0x538 287
寄存器 52: 模数比较器软件复位寄存器（SRACMP），偏移量 0x53C 288
寄存器 53: EEPROM 软件复位寄存器（SREEPROM），偏移量 0x558 289
寄存器 54: 32/64 位宽通用定时器软件复位寄存器（SRWTIMER），偏移量 0x55C 290
寄存器 55: 看门狗定时器运行模式时钟门控控制寄存器（RCGCWD），偏移量 0x600 292
寄存器 56: 16/32 位通用定时器运行模式时钟门控控制寄存器（RCGCTIMER），偏移量 0x604 293
寄存器 57: 通用输入/输出运行模式时钟门控控制寄存器（RCGCGPIO），偏移量 0x608 295
寄存器 58: 微型直接存储器访问运行模式时钟门控控制寄存器（RCGCDMA），偏移量 0x60C 297
寄存器 59: 休眠运行模式时钟门控控制寄存器（RCGCHIB），偏移量 0x614 298
寄存器 60: 通用异步收发器运行模式时钟门控控制寄存器（RCGCUART），偏移量 0x618 299
寄存器 61: 同步串行接口运行模式时钟门控控制寄存器（RCGCSSI），偏移量 0x61C 301
寄存器 62: 内部集成电路运行模式时钟门控控制寄存器（RCGCI2C），偏移量 0x620 303
寄存器 63: 控制器局域网运行模式时钟门控控制寄存器（RCGCCAN），偏移量 0x634 305
寄存器 64: 模数转换器运行模式时钟门控控制寄存器（RCGCADC），偏移量 0x638 306
寄存器 65: 模拟比较器运行模式时钟门控控制寄存器（RCGCACMP），偏移量 0x63C 307
寄存器 66: EEPROM 运行模式时钟门控控制寄存器（RCGCEEPROM），偏移量 0x658 308
寄存器 67: 32/64 位宽通用定时器运行模式时钟门控控制寄存器（RCGCWTIMER），偏移量 0x65C 309
寄存器 68: 看门狗定时器睡眠模式时钟门控控制寄存器（SCGCWD），偏移量 0x700 311
寄存器 69: 16/32 位通用定时器睡眠模式时钟门控控制寄存器（SCGCTIMER），偏移量 0x704 312

寄存器 70: 通用输入/输出睡眠模式时钟门控控制寄存器（SCGCGPIO），偏移量 0x708 314
寄存器 71: 微型直接存储器访问睡眠模式时钟门控控制寄存器（SCGCDMA），偏移量 0x70C 316
寄存器 72: 休眠睡眠模式时钟门控控制寄存器（SCGCHIB），偏移量 0x714 317
寄存器 73: 通用异步收发器睡眠模式时钟门控控制寄存器（SCGCUART），偏移量 0x718 318
寄存器 74: 同步串行接口睡眠模式时钟门控控制寄存器（SCGCSSI），偏移量 0x71C 320
寄存器 75: 内部集成电路睡眠模式时钟门控控制寄存器（SCGCI2C），偏移量 0x720 322
寄存器 76: 控制器局域网睡眠模式时钟门控控制寄存器（SCGCCAN），偏移量 0x734 324
寄存器 77: 模数转换器睡眠模式时钟门控控制寄存器（SCGCADC），偏移量 0x738 325
寄存器 78: 模拟比较器睡眠模式时钟门控控制寄存器（SCGCACMP），偏移量 0x73C 326
寄存器 79: EEPROM 睡眠模式时钟门控控制寄存器（SCGCEEPROM），偏移量 0x758 327
寄存器 80: 32/64 位宽通用定时器睡眠模式时钟门控控制寄存器（SCGCWTIMER），偏移量 0x75C 328
寄存器 81: 看门狗定时器深度睡眠模式时钟门控控制寄存器（DCGCWD），偏移量 0x800 330
寄存器 82: 16/32 位通用定时器深度睡眠模式时钟门控控制寄存器（DCGCTIMER），偏移量 0x804 331
寄存器 83: 通用输入/输出深度睡眠模式时钟门控控制寄存器（DCGCGPIO），偏移量 0x808 333
寄存器 84: 微型直接存储器访问深度睡眠模式时钟门控控制寄存器（DCGCDMA），偏移量 0x80C 335
寄存器 85: 休眠深度睡眠模式时钟门控控制寄存器（DCGCHIB），偏移量 0x814 336
寄存器 86: 通用异步收发器深度睡眠模式时钟门控控制寄存器（DCGCUART），偏移量 0x818 337
寄存器 87: 同步串行接口深度睡眠模式时钟门控控制寄存器（DCGCSSI），偏移量 0x81C 339
寄存器 88: 内部集成电路深度睡眠模式时钟门控控制寄存器（DCGCI2C），偏移量 0x820 341
寄存器 89: 控制器局域网深度睡眠模式时钟门控控制寄存器（DCGCCAN），偏移量 0x834 343
寄存器 90: 模数转换器深度睡眠模式时钟门控控制寄存器（DCGCADC），偏移量 0x838 344
寄存器 91: 模拟比较器深度睡眠模式时钟门控控制寄存器（DCGCACMP），偏移量 0x83C 345
寄存器 92: EEPROM 深度睡眠模式时钟门控控制寄存器（DCGCEEPROM），偏移量 0x858 346
寄存器 93: 32/64 位宽通用定时器深度睡眠模式时钟门控控制寄存器（DCGCWTIMER），偏移量 0x85C 347
寄存器 94: 看门狗定时器外设就绪寄存器（PRWD），偏移量 0xA00 349
寄存器 95: 16/32 位通用定时器外设就绪寄存器（PRTIMER），偏移量 0xA04 350
寄存器 96: 通用输入/输出外设就绪寄存器（PRGPIO），偏移量 0xA08 352
寄存器 97: 微型直接存储器访问外设就绪寄存器（PRDMA），偏移量 0xA0C 354
寄存器 98: 休眠外设就绪寄存器（PRHIB），偏移量 0xA14 355
寄存器 99: 通用异步收发器外设就绪寄存器（PRUART），偏移量 0xA18 356
寄存器 100: 同步串行接口外设就绪寄存器（PRSSI），偏移量 0xA1C 358
寄存器 101: 内部集成电路外设就绪寄存器（PRI2C），偏移量 0xA20 360
寄存器 102: 控制器局域网外设就绪寄存器（PRCAN），偏移量 0xA34 362
寄存器 103: 模数转换器外设就绪寄存器（PRADC），偏移量 0xA38 363
寄存器 104: 模拟比较器外设就绪寄存器（PRACMP），偏移量 0xA3C 364
寄存器 105: EEPROM 外设就绪寄存器（PREEPROM），偏移量 0xA58 365
寄存器 106: 32/64 位宽通用定时器外设就绪寄存器（PRWTIMER），偏移量 0xA5C 366
寄存器 107: 器件功能寄存器 0（DC0），偏移量 0x008 368
寄存器 108: 器件功能寄存器 1（DC1），偏移量 0x010 370
寄存器 109: 器件功能寄存器 2（DC2），偏移量 0x014 373
寄存器 110: 器件功能寄存器 3（DC3），偏移量 0x018 375
寄存器 111: 器件功能寄存器 4（DC4），偏移量 0x01C 379
寄存器 112: 器件功能寄存器 5（DC5），偏移量 0x020 381
寄存器 113: 器件功能寄存器 6（DC6），偏移量 0x024 383
寄存器 114: 器件功能寄存器 7（DC7），偏移量 0x028 384
寄存器 115: 器件功能寄存器 8（DC8），偏移量 0x02C 387

寄存器 116: 软件复位控制寄存器 0（SRCR0），偏移量 0x040 390
寄存器 117: 软件复位控制寄存器 1（SRCR1），偏移量 0x044 392
寄存器 118: 软件复位控制寄存器 2（SRCR2），偏移量 0x048 394
寄存器 119: 运行模式时钟门控控制寄存器 0（RCGC0），偏移量 0x100 396
寄存器 120: 运行模式时钟门控控制寄存器 1（RCGC1），偏移量 0x104 399
寄存器 121: 运行模式时钟门控控制寄存器 2（RCGC2），偏移量 0x108 402
寄存器 122: 睡眠模式时钟门控控制寄存器 0（SCGC0），偏移量 0x110 404
寄存器 123: 睡眠模式时钟门控控制寄存器 1（SCGC1），偏移量 0x114 406
寄存器 124: 睡眠模式时钟门控控制寄存器 2（SCGC2），偏移量 0x118 409
寄存器 125: 深度睡眠模式时钟门控控制寄存器 0（DCGC0），偏移量 0x120 411
寄存器 126: 深度睡眠模式时钟门控控制寄存器 1（DCGC1），偏移量 0x124 413
寄存器 127: 深度睡眠模式时钟门控控制寄存器 2（DCGC2），偏移量 0x128 416
寄存器 128: 器件功能寄存器 9（DC9），偏移量 0x190 418
寄存器 129: 非易失性存储器信息寄存器（NVMSTAT），偏移量 0x1A0 420

系统异常模块 421
寄存器 1: 系统异常原始中断状态（SYSEXCRIS），偏移量 0x000 422
寄存器 2: 系统异常中断屏蔽（SYSEXCIM），偏移量 0x004 424
寄存器 3: 系统异常屏蔽的中断状态（SYSEXCMIS），偏移量 0x008 426
寄存器 4: 系统异常中断清零（SYSEXCIC），偏移量 0x00C 428

休眠模块 429
寄存器 1: 休眠 RTC 计数器寄存器（HIBRTCC），偏移量 0x000 441
寄存器 2: 休眠 RTC 匹配寄存器 0（HIBRTCM0），偏移量 0x004 442
寄存器 3: 休眠 RTC 加载寄存器（HIBRTCLD），偏移量 0x00C 443
寄存器 4: 休眠控制寄存器（HIBCTL），偏移量 0x010 444
寄存器 5: 休眠中断屏蔽寄存器（HIBIM），偏移量 0x014 448
寄存器 6: 休眠原始中断状态寄存器（HIBRIS），偏移量 0x018 450
寄存器 7: 休眠屏蔽中断状态寄存器（HIBMIS），偏移量 0x01C 452
寄存器 8: 休眠中断清除寄存器（HIBIC），偏移量 0x020 454
寄存器 9: 休眠 RTC 修正寄存器（HIBRTCT），偏移量 0x024 455
寄存器 10: 休眠 RTC 亚秒寄存器（HIBRTCSS），偏移量 0x028 456
寄存器 11: 休眠数据寄存器（HIBDATA），偏移量 0x030-0x06F 457

内部存储器 458
寄存器 1: Flash 存储器地址（FMA），偏移量 0x000 473
寄存器 2: Flash 存储器数据寄存器（FMD），偏移量 0x004 474
寄存器 3: Flash 存储器控制（FMC），偏移量 0x008 475
寄存器 4: Flash 控制器原始中断状态（FCRIS），偏移量 0x00C 477
寄存器 5: Flash 控制器中断屏蔽（FCIM），偏移量 0x010 479
寄存器 6: Flash控制器可屏蔽中断的状态和清除（FCMISC），偏移量 0x014 481
寄存器 7: Flash存储器控制2（FMC2），偏移量 0x020 483
寄存器 8: Flash 写缓冲器有效（FWBVAL），偏移量 0x030 484
寄存器 9: Flash 写缓冲器n（FWBn），偏移量 0x100 - 0x17C 485
寄存器 10: Flash 容量寄存器（FSIZE），偏移量 0xFC0 486
寄存器 11: SRAM 大小寄存器（SSIZE），偏移量 0xFC4 487
寄存器 12: ROM 软件映射寄存器（ROMSWMAP），偏移量 0xFCC 488
寄存器 13: EEPROM 大小信息寄存器（EESIZE），偏移量 0x000 489
寄存器 14: EEPROM 当前块寄存器（EEBLOCK），偏移量 0x004 490
寄存器 15: EEPROM 当前寄存器（EEOFFSET），偏移量 0x008 491
寄存器 16: EEPROM 读写寄存器（EERDWR），偏移量 0x010 492

寄存器 17: EEPROM 读写加 1 寄存器（EERDWRINC），偏移量 0x014 493
寄存器 18: EEPROM 完成状态寄存器（EEDONE），偏移量 0x018 494
寄存器 19: EEPROM 支持控制和状态寄存器（EESUPP），偏移量 0x01C 496
寄存器 20: EEPROM 解锁寄存器（EEUNLOCK），偏移量 0x020 498
寄存器 21: EEPROM 保护寄存器（EEPROT），偏移量 0x030 499
寄存器 22: EEPROM 密码寄存器（EEPASS0），偏移量 0x034 500
寄存器 23: EEPROM 密码寄存器（EEPASS1），偏移量 0x038 500
寄存器 24: EEPROM 密码寄存器（EEPASS2），偏移量 0x03C 500
寄存器 25: EEPROM 中断寄存器（PWM0CTL），偏移量 0x040 501
寄存器 26: EEPROM 块隐藏寄存器（EEHIDE），偏移量 0x050 502
寄存器 27: EEPROM 调试整体擦除寄存器（EEDBGME），偏移量 0x080 503
寄存器 28: EEPROM 外设属性寄存器（EEPROMPP），偏移量 0xFC0 504
寄存器 29: ROM 控制（RMCTL），偏移量 0x0F0 505
寄存器 30: Flash 存储器保护读取启用寄存器 0（FMPRE0），偏移量 0x130 和 0x200 506
寄存器 31: Flash 存储器保护读取启用寄存器 1（FMPRE1），偏移量 0x204 506
寄存器 32: Flash 存储器保护读取启用寄存器 2（FMPRE2），偏移量 0x208 506
寄存器 33: Flash 存储器保护读取启用寄存器 3（FMPRE3），偏移量 0x20C 506
寄存器 34: Flash 存储器保护编程启用寄存器 0（FMPPE0），偏移量 0x134 和 0x400 507
寄存器 35: Flash 存储器保护编程启用寄存器 1（FMPPE1），偏移量 0x404 507
寄存器 36: Flash 存储器保护编程启用寄存器 2（FMPPE2），偏移量 0x408 507
寄存器 37: Flash 存储器保护编程启用寄存器 3（FMPPE3），偏移量 0x40C 507
寄存器 38: 启动配置（BOOTCFG），偏移量 0x1D0 508
寄存器 39: 用户寄存器 0（USER_REG0），偏移量 0x1E0 511
寄存器 40: 用户寄存器 1（USER_REG1），偏移量 0x1E4 511
寄存器 41: 用户寄存器 2（USER_REG2），偏移量 0x1E8 511
寄存器 42: 用户寄存器 3（USER_REG3），偏移量 0x1EC 511
微型直接存储器访问（µDMA） 512
寄存器 1: DMA 通道源地址末指针寄存器（DMASRCENDP），偏移量 0x000 535
寄存器 2: DMA 通道目的地址末指针寄存器（DMADSTENDP），偏移量 0x004 536
寄存器 3: DMA 通道控制字寄存器（DMACHCTL），偏移量 0x008 537
寄存器 4: DMA 状态寄存器（DMASTAT），偏移量 0x000 542
寄存器 5: DMA 配置寄存器（DMACFG），偏移量 0x004 544
寄存器 6: DMA 通道控制基指针寄存器（DMACTLBASE），偏移量 0x008 545
寄存器 7: DMA 副通道控制基指针寄存器（DMAALTBASE），偏移量 0x00C 546
寄存器 8: DMA 通道等待请求状态寄存器（DMAWAITSTAT），偏移量 0x010 547
寄存器 9: DMA 通道软件请求寄存器（DMASWREQ），偏移量 0x014 548
寄存器 10: DMA 通道采用猝发置位寄存器（DMAUSEBURSTSET），偏移量 0x018 549
寄存器 11: DMA 通道采用猝发清除寄存器（DMAUSEBURSTCLR），偏移量 0x01C 550
寄存器 12: DMA 通道请求屏蔽置位寄存器（DMAREQMASKSET），偏移量 0x020 551
寄存器 13: DMA 通道请求屏蔽清零寄存器（DMAREQMASKCLR），偏移量 0x024 552
寄存器 14: DMA 通道启用置位寄存器（DMAENASET），偏移量 0x028 553
寄存器 15: DMA 通道启用清除寄存器（DMAENACLR），偏移量 0x02C 554
寄存器 16: DMA 通道主副置位寄存器（DMAALTSET），偏移量 0x030 555
寄存器 17: DMA 通道主副清除寄存器（DMAALTCLR），偏移量 0x030 556
寄存器 18: DMA 通道优先置位寄存器（DMAPRIOSET），偏移量 0x038 557
寄存器 19: DMA 通道优先清零寄存器（DMAPRIOCLR），偏移量 0x03C 558
寄存器 20: DMA 总线错误清除寄存器（DMAERRCLR），偏移量 0x04C 559
寄存器 21: DMA 通道分配寄存器（DMACHASGN），偏移量 0x500 560

寄存器 22: DMA 通道中断状态寄存器（DMACHIS），偏移量 0x504 561
寄存器 23: DMA 通道映射选择寄存器 0（DMACHMAP0），偏移量 0x510 562
寄存器 24: DMA 通道映射选择寄存器 1（DMACHMAP1），偏移量 0x514 563
寄存器 25: DMA 通道映射选择寄存器 2（DMACHMAP2），偏移量 0x518 564
寄存器 26: DMA 通道映射选择寄存器 3（DMACHMAP3），偏移量 0x51C 565
寄存器 27: DMA 外设标识寄存器 0（DMAPeriphID0），偏移量 0xFE0 566
寄存器 28: DMA 外设标识寄存器 1（DMAPeriphID1），偏移量 0xFE4 567
寄存器 29: DMA 外设标识寄存器 2（DMAPeriphID2），偏移量 0xFE8 568
寄存器 30: DMA 外设标识寄存器 3（DMAPeriphID3），偏移量 0xFEC 569
寄存器 31: DMA 外设标识寄存器 4（DMAPeriphID4），偏移量 0xFD0 570
寄存器 32: DMA PrimeCell 标识寄存器 0（DMAPCellID0），偏移量 0xFF0 571
寄存器 33: DMA PrimeCell 标识寄存器 1（DMAPCellID1），偏移量 0xFF4 572
寄存器 34: DMA PrimeCell 标识寄存器 2（DMAPCellID2），偏移量 0xFF8 573
寄存器 35: DMA PrimeCell 标识寄存器 3（DMAPCellID3），偏移量 0xFFC 574
通用输入/输入端口（GPIOs） 575
寄存器 1: GPIO 数据寄存器（GPIODATA），偏移量 0x000 586
寄存器 2: GPIO 方向寄存器（GPIODIR），偏移量 0x400 587
寄存器 3: GPIO 中断检测寄存器（GPIOIS），偏移量 0x404 588
寄存器 4: GPIO 中断双边沿（GPIOIBE），偏移量 0x408 589
寄存器 5: GPIO 中断事件寄存器（GPIOIEV），偏移量 0x40C 590
寄存器 6: GPIO 中断屏蔽寄存器（GPIOIM），偏移量 0x410 591
寄存器 7: GPIO 原始中断状态寄存器（GPIORIS），偏移量 0x414 592
寄存器 8: GPIO 屏蔽中断状态寄存器（GPIOMIS），偏移量 0x418 593
寄存器 9: GPIO 中断清除寄存器（GPIOICR），偏移量 0x41C 594
寄存器 10: GPIO 备用功能选择寄存器（GPIOAFSEL），偏移量 0x420 595
寄存器 11: GPIO 2-mA 驱动选择寄存器（GPIODR2R），偏移量 0x500 597
寄存器 12: GPIO 4-mA 驱动选择寄存器（GPIODR4R），偏移量 0x504 598
寄存器 13: GPIO 8-mA 驱动选择寄存器（GPIODR8R），偏移量 0x508 599
寄存器 14: GPIO 开漏选择寄存器（GPIOODR），偏移量 0x50C 600
寄存器 15: GPIO 上拉电阻选择寄存器（GPIOPUR），偏移量 0x510 601
寄存器 16: GPIO 下拉电阻选择寄存器（GPIOPDR），偏移量 0x514 603
寄存器 17: GPIO 斜率控制选择寄存器（GPIOSLR），偏移量 0x518 605
寄存器 18: GPIO 数字使能寄存器（GPIODEN），偏移量 0x51C 606
寄存器 19: GPIO 锁定寄存器（GPIOLOCK），偏移量 0x520 608
寄存器 20: GPIO 确认寄存器（GPIOCR），偏移量 0x524 609
寄存器 21: GPIO 模拟选择寄存器（GPIOAMSEL），偏移量 0x528 611
寄存器 22: GPIO端口控制寄存器（GPIOPCTL），偏移量0x52C 612
寄存器 23: GPIO ADC 控制寄存器（GPIOADCCTL），偏移量 0x530 614
寄存器 24: GPIO DMA 控制寄存器（GPIODMACTL），偏移量 0x534 615
寄存器 25: GPIO 外设标识寄存器 4（GPIOPeriphID4），偏移量 0xFD0 616
寄存器 26: GPIO 外设标识寄存器 5（GPIOPeriphID5），偏移量 0xFD4 617
寄存器 27: GPIO 外设标识寄存器 6（GPIOPeriphID6），偏移量0xFD8 618
寄存器 28: GPIO 外设标识寄存器 7（GPIOPeriphID7），偏移量 0xFDC 619
寄存器 29: GPIO 外设标识寄存器 0（GPIOPeriphID0），偏移量 0xFE0 620
寄存器 30: GPIO 外设标识寄存器 1（GPIOPeriphID1），偏移量 0xFE4 621
寄存器 31: GPIO 外设标识寄存器 2（GPIOPeriphID2），偏移量 0xFE8 622
寄存器 32: GPIO 外设标识寄存器 3（GPIOPeriphID3），偏移量 0xFEC 623
寄存器 33: GPIO PrimeCell 标识寄存器 0（GPIOPCellID0），偏移量 0xFF0 624

寄存器 34: GPIO PrimeCell 标识寄存器 1（GPIOPCellID1），偏移量 0xFF4 625
寄存器 35: GPIO PrimeCell 标识寄存器 2（GPIOPCellID2），偏移量 0xFF8 626
寄存器 36: GPIO PrimeCell 标识寄存器 3（GPIOPCellID3），偏移量 0xFFC 627
通用定时器 628
寄存器 1: GPTM 配置寄存器（GPTMCFG），偏移量 0x000 650
寄存器 2: GPTM Timer A 模式寄存器（GPTMTAMR），偏移量 0x004 651
寄存器 3: GPTM Timer B 模式寄存器（GPTMTBMR），偏移量 0x008 654
寄存器 4: GPTM 控制寄存器（GPTMCTL），偏移量 0x00C 657
寄存器 5: GPTM 同步寄存器（GPTMSYNC），偏移量 0x010 660
寄存器 6: GPTM 中断屏蔽寄存器（GPTMIMR），偏移量 0x018 663
寄存器 7: GPTM 原始中断状态寄存器（GPTMRIS），偏移量 0x01C 666
寄存器 8: GPTM 屏蔽的中断状态寄存器（GPTMMIS），偏移量 0x020 669
寄存器 9: GPTM 中断清除寄存器（GPTMICR），偏移量 0x024 672
寄存器 10: GPTM Timer A 间隔加载寄存器（GPTMTAILR），偏移量 0x028 674
寄存器 11: GPTM Timer B 间隔加载寄存器（GPTMTBILR），偏移量 0x02C 675
寄存器 12: GPTM Timer A 匹配寄存器（GPTMTAMATCHR），偏移量 0x030 676
寄存器 13: GPTM Timer B 匹配寄存器（GPTMTBMATCHR），偏移量 0x034 677
寄存器 14: GPTM Timer A 预分频寄存器（GPTMTAPR），偏移量 0x038 678
寄存器 15: GPTM Timer B 预分频寄存器（GPTMTBPR），偏移量 0x03C 679
寄存器 16: GPTM TimerA 预分频匹配寄存器（GPTMTAPMR），偏移量 0x040 680
寄存器 17: GPTM TimerB 预分频匹配寄存器（GPTMTBPMR），偏移量 0x044 681
寄存器 18: GPTM Timer A 寄存器（GPTMTAPR），偏移量 0x048 682
寄存器 19: GPTM Timer B 寄存器（GPTMTBPR），偏移量 0x04C 683
寄存器 20: GPTM Timer A 值寄存器（GPTMTAV），偏移量 0x050 684
寄存器 21: GPTM Timer B 值寄存器（GPTMTBV），偏移量 0x054 685
寄存器 22: GPTM RTC 预分频寄存器（GPTMRTCPD），偏移量 0x058 686
寄存器 23: GPTM Timer A 预分频快照寄存器（GPTMTAPS），偏移量 0x05C 687
寄存器 24: GPTM Timer B 预分频快照寄存器（GPTMTBPS），偏移量 0x060 688
寄存器 25: GPTM Timer A 预分频值寄存器（GPTMTAPV），偏移量 0x064 689
寄存器 26: GPTM Timer B 预分频值寄存器（GPTMTBPV），偏移量 0x068 690
寄存器 27: GPTM 外设属性寄存器（GPTMPP），偏移量 0xFC0 691
看门狗定时器 692
寄存器 1: 看门狗加载寄存器（WDTLOAD），偏移量 0x000 696
寄存器 2: 看门狗当前值寄存器（WDTVALUE），偏移量 0x004 697
寄存器 3: 看门狗控制寄存器（WDTCTL），偏移量 0x008 698
寄存器 4: 看门狗中断清除寄存器（WDTICR），偏移量 0x00C 700
寄存器 5: 看门狗原始中断状态寄存器（WDTRIS），偏移量 0x010 701
寄存器 6: 看门狗可屏蔽中断状态寄存器（WDTMIS），偏移量 0x014 702
寄存器 7: 看门狗测试寄存器（WDTTEST），偏移量 0x418 703
寄存器 8: 看门狗锁定寄存器（WDTLOCK），偏移量 0xC00 704
寄存器 9: 看门狗外设标识寄存器 4（WDTPeriphID4），偏移量 0xFD0 705
寄存器 10: 看门狗外设标识寄存器 5（WDTPeriphID5），偏移量 0xFD4 706
寄存器 11: 看门狗外设标识寄存器 6（WDTPeriphID6），偏移量 0xFD8 707
寄存器 12: 看门狗外设标识寄存器 7（WDTPeriphID7），偏移量 0xFDC 708
寄存器 13: 看门狗外设标识寄存器 0（WDTPeriphID0），偏移量 0xFE0 709
寄存器 14: 看门狗外设标识寄存器 1（WDTPeriphID1），偏移量 0xFE4 710
寄存器 15: 看门狗外设标识寄存器 2（WDTPeriphID2），偏移量 0xFE8 711
寄存器 16: 看门狗外设标识寄存器 3（WDTPeriphID3），偏移量 0xFEC 712

寄存器 17: 看门狗 PrimeCell 标识寄存器 0（WDTPCellID0），偏移量 0xFF0 713
寄存器 18: 看门狗 PrimeCell 标识寄存器 1（WDTPCellID1），偏移量 0xFF4 714
寄存器 19: 看门狗 PrimeCell 标识寄存器 2（WDTPCellID2），偏移量 0xFF8 715
寄存器 20: 看门狗 PrimeCell 标识寄存器 3（WDTPCellID3），偏移量 0xFFC 716
模-数转换器（ADC） 717
寄存器 1: ADC 有效采样序列发生器寄存器（ADCACTSS），偏移量 0x000 737
寄存器 2: ADC原始中断状态寄存器（ADCRIS），偏移量 0x004 739
寄存器 3: ADC 中断掩码寄存器（ADCIM），偏移量 0x008 741
寄存器 4: ADC 中断状态及清除寄存器（ADCISC），偏移量 0x00C 743
寄存器 5: ADC 上溢状态寄存器（ADCOSTAT），偏移量 0x010 746
寄存器 6: ADC 事件复用选择寄存器（ADCEMUX），偏移量 0x014 748
寄存器 7: ADC 下溢状态寄存器（ADCUSTAT），偏移量 0x018 753
寄存器 8: ADC 采样序列发生器优先级寄存器（ADCSSPRI），偏移量 0x020 754
寄存器 9: ADC 采样相位控制寄存器（ADCSPC），偏移量 0x024 755
寄存器 10: ADC 处理器采样序列启动寄存器（ADCPSSI），偏移量 0x028 756
寄存器 11: ADC 采样平均控制寄存器（ADCSAC），偏移量 0x030 758
寄存器 12: ADC 数字比较器中断状态及清除寄存器（ADCDCISC），偏移量 0x034 759
寄存器 13: ADC 控制寄存器（ADCCTL），偏移量 0x038 761
寄存器 14: ADC 采样序列输入复用选择寄存器 0（ADCSSMUX0），偏移量 0x040 762
寄存器 15: ADC 采样序列控制寄存器 0（ADCSSCTL0），偏移量 0x044 763
寄存器 16: ADC 采样序列结果 FIFO 寄存器 0（ADCSSFIFO0），偏移量 0x048 769
寄存器 17: ADC 采样序列结果 FIFO 寄存器 1（ADCSSFIFO1），偏移量 0x068 769
寄存器 18: ADC 采样序列结果 FIFO 寄存器 2（ADCSSFIFO2），偏移量 0x088 769
寄存器 19: ADC 采样序列结果 FIFO 寄存器 3（ADCSSFIFO3），偏移量 0x0A8 769
寄存器 20: ADC 采样序列 FIFO 0 状态寄存器（ADCSSFSTAT0），偏移量 0x04C 770
寄存器 21: ADC 采样序列 FIFO 1 状态寄存器（ADCSSFSTAT1），偏移量 0x06C 770
寄存器 22: ADC 采样序列 FIFO 2 状态寄存器（ADCSSFSTAT2），偏移量 0x08C 770
寄存器 23: ADC 采样序列 FIFO 3 状态寄存器（ADCSSFSTAT3），偏移量 0x0AC 770
寄存器 24: ADC 采样序列寄存器 0（ADCSSOP0），偏移量 0x050 772
寄存器 25: ADC 采样序列数字比较器选择寄存器 0（ADCSSDC0），偏移量 0x054 774
寄存器 26: ADC 采样序列输入复用选择寄存器 1（ADCSSMUX1），偏移量 0x060 776
寄存器 27: ADC 采样序列输入复用选择寄存器 2（ADCSSMUX2），偏移量 0x080 776
寄存器 28: ADC 采样序列控制寄存器 1（ADCSSCTL1），偏移量 0x064 777
寄存器 29: ADC 采样序列控制寄存器 2（ADCSSCTL2），偏移量 0x084 777
寄存器 30: ADC 采样序列 1 工作寄存器（ADCSSOP1），偏移量 0x070 781
寄存器 31: ADC 采样序列 2 工作寄存器（ADCSSOP2），偏移量 0x090 781
寄存器 32: ADC 采样序列数字比较器选择寄存器 1（ADCSSDC1），偏移量 0x074 782
寄存器 33: ADC 采样序列数字比较器选择寄存器 2（ADCSSDC2），偏移量 0x094 782
寄存器 34: ADC 采样序列输入复用选择寄存器 3（ADCSSMUX3），偏移量 0x0A0 783
寄存器 35: ADC 采样序列控制寄存器 3（ADCSSCTL3），偏移量 0x0A4 784
寄存器 36: ADC 采样序列器 3 工作寄存器（ADCSSOP3），偏移量 0x0B0 786
寄存器 37: ADC 采样序列 3 数字比较器选择寄存器（ADCSSDC3），偏移量 0x0B4 787
寄存器 38: ADC 数字比较器复位启动条件寄存器（ADCDCRIC），偏移量 0xD00 788
寄存器 39: ADC 数字比较器控制寄存器 0（ADCDCCTL0），偏移量 0xE00 792
寄存器 40: ADC 数字比较器控制寄存器 1（ADCDCCTL1），偏移量 0xE04 792
寄存器 41: ADC 数字比较器控制寄存器 2（ADCDCCTL2），偏移量 0xE08 792
寄存器 42: ADC 数字比较器控制寄存器 3（ADCDCCTL3），偏移量 0xE0C 792
寄存器 43: ADC 数字比较器控制寄存器 4（ADCDCCTL4），偏移量 0xE10 792

寄存器 44: ADC 数字比较器控制寄存器 5（ADCDCCTL5），偏移量 0xE14 792
寄存器 45: ADC 数字比较器控制寄存器 6（ADCDCCTL6），偏移量 0xE18 792
寄存器 46: ADC 数字比较器控制寄存器 7（ADCDCCTL7），偏移量 0xE1C 792
寄存器 47: ADC 数字比较器范围寄存器 0（ADCDCCMP0），偏移量 0xE40 794
寄存器 48: ADC 数字比较器范围寄存器 1（ADCDCCMP1），偏移量 0xE44 794
寄存器 49: ADC 数字比较器范围寄存器 2（ADCDCCMP2），偏移量 0xE48 794
寄存器 50: ADC 数字比较器范围寄存器 3（ADCDCCMP3），偏移量 0xE4C 794
寄存器 51: ADC 数字比较器范围寄存器 4（ADCDCCMP4），偏移量 0xE50 794
寄存器 52: ADC 数字比较器范围寄存器 5（ADCDCCMP5），偏移量 0xE54 794
寄存器 53: ADC 数字比较器范围寄存器 6（ADCDCCMP6），偏移量 0xE58 794
寄存器 54: ADC 数字比较器范围寄存器 7（ADCDCCMP7），偏移量 0xE5C 794
寄存器 55: ADC 外设属性寄存器（ADCPP），偏移量 0xFC0 795
寄存器 56: ADC 外设配置寄存器（ADCPC），偏移量 0xFC4 797
寄存器 57: ADC 时钟配置寄存器（ADCCC），偏移量 0xFC8 798
通用异步收发器（UART） 799
寄存器 1: UART 数据寄存器（UARTDR），偏移量 0x000 811
寄存器 2: UART 接收状态/错误清除寄存器（UARTRSR/UARTECR），偏移量 0x004 813
寄存器 3: UART 标志寄存器（UARTFR），偏移量 0x018 816
寄存器 4: UART IrDA 低功耗寄存器（UARTILPR），偏移量 0x020 818
寄存器 5: UART 波特率分频值整数寄存器（UARTIBRD），偏移量 0x024 819
寄存器 6: UART 波特率分频值小数寄存器（UARTFBRD），偏移量 0x028 820
寄存器 7: UART 线控寄存器（UARTLCRH），偏移量 0x02C 821
寄存器 8: UART 控制寄存器（UARTCTL），偏移量 0x030 823
寄存器 9: UART 中断 FIFO 深度选择寄存器（UARTIFLS），偏移量 0x034 827
寄存器 10: UART 中断屏蔽寄存器（UARTIM），偏移量 0x038 829
寄存器 11: UART 原始中断状态寄存器（UARTRIS），偏移量 0x03C 831
寄存器 12: UART 屏蔽中断状态寄存器（UARTMIS），偏移量 0x040 834
寄存器 13: UART 中断清除寄存器（UARTICR），偏移量 0x044 837
寄存器 14: UART DMA 控制寄存器（UARTDMACTL），偏移量 0x048 839
寄存器 15: UART 9 位模式自身地址寄存器（UART9BITADDR），偏移量 0x0A4 840
寄存器 16: UART 9 位模式自身地址屏蔽寄存器（UART9BITAMASK），偏移量 0x0A8 841
寄存器 17: UART 外设属性寄存器（UARTPP），偏移量 0xFC0 842
寄存器 18: UART 时钟配置寄存器（UARTCC），偏移量 0xFC8 843
寄存器 19: UART 外设标识寄存器 4（UARTPeriphID4），偏移量 0xFD0 844
寄存器 20: UART 外设标识寄存器 5（UARTPeriphID5），偏移量 0xFD4 845
寄存器 21: UART 外设标识寄存器 6（UARTPeriphID6），偏移量 0xFD8 846
寄存器 22: UART 外设标识寄存器 7（UARTPeriphID7），偏移量 0xFDC 847
寄存器 23: UART 外设标识寄存器 0（UARTPeriphID0），偏移量 0xFE0 848
寄存器 24: UART 外设标识寄存器 1（UARTPeriphID1），偏移量 0xFE4 849
寄存器 25: UART 外设标识寄存器 2（UARTPeriphID2），偏移量 0xFE8 850
寄存器 26: UART 外设标识寄存器 3（UARTPeriphID3），偏移量 0xFEC 851
寄存器 27: UART PrimeCell 标识寄存器 0（UARTPCellID0），偏移量 0xFF0 852
寄存器 28: UART PrimeCell 标识寄存器 1（UARTPCellID1），偏移量 0xFF4 853
寄存器 29: UART PrimeCell 标识寄存器 2（UARTPCellID2），偏移量 0xFF8 854
寄存器 30: UART PrimeCell 标识寄存器 3（UARTPCellID3），偏移量 0xFFC 855
同步串行接口（SSI） 856
寄存器 1: SSI 控制寄存器0（SSICR0），偏移量 0x000 871
寄存器 2: SSI 控制寄存器1（SSICR1），偏移量 0x004 873

寄存器 3: SSI 数据寄存器（SSIDR），偏移量 0x008 875
寄存器 4: SSI 状态寄存器（SSISR），偏移量 0x00C 876
寄存器 5: SSI 时钟预分频寄存器（SSICPSR），偏移量 0x010 878
寄存器 6: SSI 中断屏蔽寄存器（SSIIM），偏移量 0x014 879
寄存器 7: SSI 原始中断状态寄存器（SSIRIS），偏移量 0x018 880
寄存器 8: SSI 屏蔽中断状态寄存器（SSIMIS），偏移量 0x01C 882
寄存器 9: SSI 中断清除寄存器（SSIICR），偏移量 0x020 884
寄存器 10: SSI DMA 控制寄存器（SSIDMACTL），偏移量 0x024 885
寄存器 11: SSI 时钟配置寄存器（SSICC），偏移量 0xFC8 886
寄存器 12: SSI 外设标识寄存器 4（SSIPeriphID4），偏移量 0xFD0 887
寄存器 13: SSI 外设标识寄存器 5（SSIPeriphID5），偏移量 0xFD4 888
寄存器 14: SSI 外设标识寄存器 6（SSIPeriphID6），偏移量 0xFD8 889
寄存器 15: SSI 外设标识寄存器 7（SSIPeriphID7），偏移量 0xFDC 890
寄存器 16: SSI 外设标识寄存器 0（SSIPeriphID0），偏移量 0xFE0 891
寄存器 17: SSI 外设标识寄存器 1（SSIPeriphID1），偏移量 0xFE4 892
寄存器 18: SSI 外设标识寄存器 2（SSIPeriphID2），偏移量 0xFE8 893
寄存器 19: SSI 外设标识寄存器 3（SSIPeriphID3），偏移量 0xFEC 894
寄存器 20: SSI PrimeCell 标识寄存器 0（SSIPCellID0），偏移量 0xFF0 895
寄存器 21: SSI PrimeCell 标识寄存器 1（SSIPCellID1），偏移量 0xFF4 896
寄存器 22: SSI PrimeCell 标识寄存器 2（SSIPCellID2），偏移量 0xFF8 897
寄存器 23: SSI PrimeCell 标识寄存器 3（SSIPCellID3），偏移量 0xFFC 898
内部集成电路（I²C）接口 899
寄存器 1: I²C 主机从机地址寄存器（I2CMSA），偏移量 0x000 919
寄存器 2: I²C 主机控制/状态寄存器（I2CMCS），偏移量 0x004 920
寄存器 3: I²C 主机数据寄存器（I2CMDR），偏移量 0x008 925
寄存器 4: I²C 主机定时器周期寄存器（I2CMTPR），偏移量 0x00C 926
寄存器 5: I²C 主机中断屏蔽寄存器（I2CMIMR），偏移量 0x010 927
寄存器 6: I²C 主机原始中断状态寄存器（I2CMRIS），偏移量 0x014 928
寄存器 7: I²C 主机屏蔽中断状态寄存器（I2CMMIS），偏移量 0x018 929
寄存器 8: I²C 主机中断清除寄存器（I2CMICR），偏移量 0x01C 930
寄存器 9: I²C 主机配置寄存器（I2CMCR），偏移量 0x020 931
寄存器 10: I²C 主机时钟低电平超时计数寄存器（I2CMCLKOCNT），偏移量 0x024 932
寄存器 11: I²C 主机总线监视寄存器（I2CMBMON），偏移量 0x02C 933
寄存器 12: I²C 主机配置 2 寄存器（I2CMCR2），偏移量 0x038 934
寄存器 13: I²C 从机本身地址寄存器（I2CSOAR），偏移量 0x800 935
寄存器 14: I²C 从机控制/状态寄存器（I2CSCSR），偏移量 0x804 936
寄存器 15: I²C 从机数据寄存器（I2CSDR），偏移量 0x808 938
寄存器 16: I²C 从机中断屏蔽（I2CSIMR），偏移量 0x80C 939
寄存器 17: I²C 从机原始中断状态寄存器（I2CSRIS），偏移量 0x810 940
寄存器 18: I²C 从机屏蔽中断状态寄存器（I2CSMIS），偏移量 0x814 941
寄存器 19: I²C 从机中断清除寄存器（I2CSICR），偏移量 0x818 942
寄存器 20: I²C 从机自身地址寄存器 2（I2CSOAR2），偏移量 0x81C 943
寄存器 21: I²C 从机应答控制寄存器（I2CSACKCTL），偏移量 0x820 944
寄存器 22: I²C 外设属性寄存器（I2CPP），偏移量 0xFC0 945
寄存器 23: I²C 外设配置寄存器（I2CPC），偏移量 0xFC4 946
控制器局域网（CAN）模块 947
寄存器 1: CAN 控制寄存器（CANCTL），偏移量 0x000 966

寄存器 2: CAN 状态寄存器（CANSTS），偏移量0x004 968
寄存器 3: CAN 错误计数寄存器（CANERR），偏移量 0x008 970
寄存器 4: CAN 位时序寄存器（CANBIT），偏移量 0x00C 971
寄存器 5: CAN 中断寄存器（CANINT），偏移量 0x010 972
寄存器 6: CAN 测试寄存器（CANTST），偏移量 0x014 973
寄存器 7: CAN 波特率预分频器扩展寄存器（CANBRPE），偏移量 0x018 975
寄存器 8: CAN IF1 指令请求寄存器（CANIF1CRQ），偏移量 0x020 976
寄存器 9: CAN IF2 指令请求寄存器（CANIF2CRQ），偏移量 0x080 976
寄存器 10: CAN IF1 指令屏蔽寄存器（CANIF1CMSK），偏移量 0x024 977
寄存器 11: CAN IF2 指令屏蔽寄存器（CANIF2CMSK），偏移量 0x084 977
寄存器 12: CAN IF1 屏蔽寄存器 1（CANIF1MSK1），偏移量 0x028 980
寄存器 13: CAN IF2 屏蔽寄存器 1（CANIF2MSK1），偏移量 0x088 980
寄存器 14: CAN IF1 屏蔽寄存器 2（CANIF1MSK2），偏移量 0x02C 981
寄存器 15: CAN IF2 屏蔽寄存器 2（CANIF2MSK2），偏移量 0x08C 981
寄存器 16: CAN IF1 仲裁寄存器 1（CANIF1ARB1），偏移量 0x030 982
寄存器 17: CAN IF2 仲裁寄存器 1（CANIF2ARB1），偏移量 0x090 982
寄存器 18: CAN IF1 仲裁寄存器 2（CANIF1ARB2），偏移量 0x034 983
寄存器 19: CAN IF2 仲裁寄存器 2（CANIF2ARB2），偏移量 0x094 983
寄存器 20: CAN IF1 报文控制寄存器（CANIF1MCTL），偏移量 0x038 985
寄存器 21: CAN IF2 报文控制寄存器（CANIF2MCTL），偏移量 0x098 985
寄存器 22: CAN IF1 数据寄存器 A1（CANIF1DA1），偏移量 0x03C 988
寄存器 23: CAN IF1 数据寄存器 A2（CANIF1DA2），偏移量 0x040 988
寄存器 24: CAN IF1 数据寄存器 B1（CANIF1DB1），偏移量 0x044 988
寄存器 25: CAN IF1 数据寄存器 B2（CANIF1DB2），偏移量 0x048 988
寄存器 26: CAN IF2 数据寄存器 A1（CANIF2DA1），偏移量 0x09C 988
寄存器 27: CAN IF2 数据寄存器 A2（CANIF2DA2），偏移量 0x0A0 988
寄存器 28: CAN IF2 数据寄存器 B1（CANIF2DB1），偏移量 0x0A4 988
寄存器 29: CAN IF2 数据寄存器 B2（CANIF2DB2），偏移量 0x0A8 988
寄存器 30: CAN 传输请求寄存器 1（CANTXRQ1），偏移量 0x100 989
寄存器 31: CAN 传输请求寄存器 2（CANTXRQ2），偏移量 0x104 989
寄存器 32: CAN 新数据寄存器 1（CANNWDA1），偏移量 0x120 990
寄存器 33: CAN 新数据寄存器 2（CANNWDA2），偏移量 0x124 990
寄存器 34: CAN 报文 1 中断挂起寄存器（CANMSG1INT），偏移量 0x140 991
寄存器 35: CAN 报文 2 中断挂起寄存器（CANMSG2INT），偏移量 0x144 991
寄存器 36: CAN 报文 1 有效寄存器（CANMSG1VAL），偏移量 0x160 992
寄存器 37: CAN 报文 2 有效寄存器（CANMSG2VAL），偏移量 0x164 992
模拟比较器 993
寄存器 1: 模拟比较器屏蔽中断状态寄存器（ACMIS），偏移量 0x00 999
寄存器 2: 模拟比较器原始中断状态寄存器（ACRIS），偏移量 0x04 1000
寄存器 3: 模拟比较器中断启用寄存器（ACINTEN），偏移量 0x08 1001
寄存器 4: 模拟比较器参考电压控制寄存器（ACREFCTL），偏移量 0x010 1002
寄存器 5: 模拟比较器状态寄存器 0（ACSTAT0），偏移量 0x020 1003
寄存器 6: 模拟比较器状态寄存器 1（ACSTAT1），偏移量 0x040 1003
寄存器 7: 模拟比较器控制寄存器 0（ACCTL0），偏移量 0x024 1004
寄存器 8: 模拟比较器控制寄存器 1（ACCTL1），偏移量 0x044 1004
寄存器 9: 模拟比较器外设属性寄存器（ACMPPP），偏移量 0xFC0 1006

关于本文档

该数据手册提供 TM4C1231H6PM 微控制器的参考信息，描述了围绕 ARM® Cortex™-M4F 内核设计的片上系统 (SoC) 器件的各功能模块。

受众

本手册的受众是系统软件开发人员、硬件设计人员和应用设计人员。

关于本手册

本文档分成若干章节，每个章节与主要的特性相对应。

相关文档

相关文档可从以下网站获取： Tiva™ C 系列 网址 http://www.ti.com/tiva-c ：

- “Tiva™ C Series TM4C123x Silicon Errata”（文献编号 SPMZ849）
- “ARM® Cortex™-M4 Errata（文献编号 SPMZ637）”
- “TivaWare™ Boot Loader for C Series User's Guide（文献编号 SPMU301）”
- “TivaWare™ Graphics Library for C Series User's Guide（文献编号 SPMU300）”
- “TivaWare™ Peripheral Driver Library for C Series User's Guide（文献编号 SPMU298）”
- “TivaWare™ USB Library for C Series User's Guide（文献编号 SPMU297）”
- “TM4C1231H6PM ROM User's Guide”

下列相关文档也可能包含有用信息：

- “ARM® Cortex™-M4 Devices Generic User Guide（文献编号 ARM DUI 0553A）中的 Cortex™-M4 指令集章节”
- “ARM® Debug Interface V5 Architecture Specification”
- “ARM® Embedded Trace Macrocell Architecture Specification”
- “IEEE Standard 1149.1-Test Access Port and Boundary-Scan Architecture”

这个文献列表的实时性根据发布的日期来判断。其他文献（包括应用说明和白皮书），请访问网站获取。

文档约定

本文档使用的约定如表1（32页）所示。

表 **1.** 文档约定

表示法	含义
通用寄存器的表示法	
寄存器	APB 寄存器用大写的粗体表示。例如，PBORCTL 是上电和掉电复位控制寄存器。如果一个寄存器名称包含一个小写的 n，它就代表了多个寄存器。例如，SRCRn 代表了下面三个软件复位控制寄存器的任何一个或全部：SRCR0、SRCR1 和 SRCR2。
位	寄存器的一个位。
位域	2 个或更多连续和相关联的位。
偏移量 0x*nnn*	寄存器地址的一个十六进制增量，增量是相对表2-4（75页）中指定的模块基址而言的。
寄存器 *N*	为了方便引用，在整篇文档中，寄存器被顺序编号。寄存器编号对软件没有意义。
保留	标注为保留的寄存器位保留下来供将来使用在大多数情况下，保留位被设置成 0；但是，用户软件不应当依赖保留位的值。为了使软件兼容未来的器件，保留位的值应当在读-修改-写过程中保留下来。
yy:xx	寄存器位的范围从 xx 到 yy（xx 和 yy 包括在内）。例如，31:15 表示相应寄存器的位 15 到 31。
寄存器 位/域 类型	寄存器位框图中的这个值指明了在控制器上运行的软件是否能改变位域的值。
RC	软件可以读取这个域。位/域在被读取之后由硬件清零。
RO	软件可以读取这个域。始终写入芯片复位值。
R/W	软件可以读或写这个域。
R/WC	软件可以读或写这个域。向该寄存器写入任意值都会将寄存器清零。
R/W1C	软件可以读或写这个域。向 W1C 位写入 0 不影响寄存器中的位值。写 1 清零寄存器中位的值；剩余的位保持变。 这个寄存器类型主要用来清零中断状态位，执行读操作来提供中断状态，写入读取的值只清除在读寄存器时被报告的中断。
R/W1S	软件可以读或向此域写 1。向 R/W1S 位写 0 不影响寄存器中的位值。
W1C	软件可以写这个域。向 W1C 位写入 0 不影响寄存器中的位值。写 1 清零寄存器中位的值；剩余的位保持变。读寄存器返回的数据没有意义。 这个寄存器通常用来清零中断寄存器中相应的位。
WO	只有软件的写操作才有效；读寄存器返回的数据没有意义。
寄存器 位/域 复位值	寄存器位框图中的这个值是任何复位后的位/域值，但特别注明的除外。
0	芯片复位时位被清零。
1	芯片复位时位被置1。
-	不确定。
管脚/信号表示法	
[ ]	管脚备用功能；管脚默认用作不带方括号的信号。
管脚	参考封装上的物理连接
信号	参考管脚的电气信号编码。
使一个信号有效	将信号的值从逻辑假状态变为逻辑真状态对于高电平有效的信号，有效的信号值为 1（高）；对于低电平有效的信号，有效的信号值为0（低）。有效极性（高或低）由信号名称来定义（见下面的 SIGNAL 和 SIGNAL）。
使一个信号无效	将信号的值从逻辑真状态变为逻辑假状态。
SIGNAL	信号名称采用大写字母，使用 Courier 字体。信号名称带有上横线表示该信号低有效。要使 SIGNAL 有效就将其驱动为低电平；要使 SIGNAL 无效就将其驱动为高电平。
SIGNAL	信号名称采用大写字母，使用 Courier 字体。高有效的信号没有上横线。要使 SIGNAL 有效就将其驱动为高电平；要使 SIGNAL 无效就将其驱动为低电平。

表 1. 文档约定（续）

表示法	含义
数值	
X	大写的X表示几个值都是可取的，此处的X可以是任何合法的样式。例如，二进制值0X00可以是0100或0000，十六进制值0xX可以是0x0或0x1，等等。
0x	十六进制值带有前缀 0x。例如，0x00FF 是十六进制值 FF。 寄存器表中的所有其它数假设为二进制。在概念信息中，二进制数用一个 b 后缀表示，例如，1011b，写十进制数无需前缀或后缀。

1 结构概述

德州仪器的 Tiva™ C 系列 微控制器采用基于 ARM® Cortex™-M 的卓越架构，具备强大的集成能力，并提供成熟的软件和开发工具生态系统，是设计师的理想选择。为了提供最佳的性能和灵活性，我们为 Tiva™ C 系列 架构推出了带 FPU、各种集成存储器以及可编程 GPIO 的 80 MHz Cortex-M。Tiva™ C 系列 能够集成适合特定应用的外设，并提供广泛的软件工具选项，可大大降低电路板成本、减少设计周期时间，是用户最理想的成本效益型解决方案。我们的 Tiva™ C 系列 微控制器能够帮助您缩短产品面市时间、节省成本，是高性能 32 位应用的领先之选。

本章给出了 Tiva™ C 系列 微控制器的概述以及有关 TM4C1231H6PM 微控制器的细节：

- “Tiva™ C 系列 概述”（34页）
- “TM4C1231H6PM 微控制器概述”（34页）
- “TM4C1231H6PM 微控制器特性”（37页）
- “TM4C1231H6PM 微控制器硬件细节”（51页）
- “开发套件”（51页）
- “支持信息”（52页）

1.1 Tiva™ C 系列 概述

德州仪器的 Tiva™ C 系列 ARM Cortex-M4 微控制器具有顶级性能和高级集成功能。本产品系列适用于需要控制成本，同时又需要大量控制和连接能力的应用，例如：

- 低功耗手持智能设备
- 游戏设备
- 家用和商用监控
- 运动控制
- 医疗器械
- 测试和测量仪器
- 工厂自动化
- 火警和安防用具
- 智能能源/智能电网解决方案
- 智能照明控制
- 运输业

至于那些对功耗有特别要求的应用方案，TM4C1231H6PM 微控制器还具有一个带备用电池的休眠模块，从而有效地使 TM4C1231H6PM 在长时间未被激活时进入低功耗状态。休眠模块拥有一个上电/掉电序列发生器、连续时间计数器 (RTC)、多种休眠唤醒选项、以及专用的带备用电池的存储器，使 TM4C1231H6PM 微控制器极其适合电池应用。

另外，TM4C1231H6PM 微控制器的优势还在于能够方便的运用多种 ARM 的开发工具和片上系统 (SoC) 的底层 IP 应用方案，以及广大的用户群体。另外，该微控制器使用了兼容 ARM的Thumb® 指令集的 Thumb2 指令集来减少存储容量的需求，并以此达到降低成本的目的。最后，TM4C1231H6PM 微控制器代码大多与 Tiva™ C 系列 产品线兼容，为设计带来了灵活性。

Texas Instruments 为了能够帮助用户产品快速的上市，提供了一整套的解决方案，包括评估和开发用的板卡、白皮书和应用笔记 、方便使用的外设驱动程序库、以及强劲的支持、销售和分销网络。

1.2 TM4C1231H6PM 微控制器概述

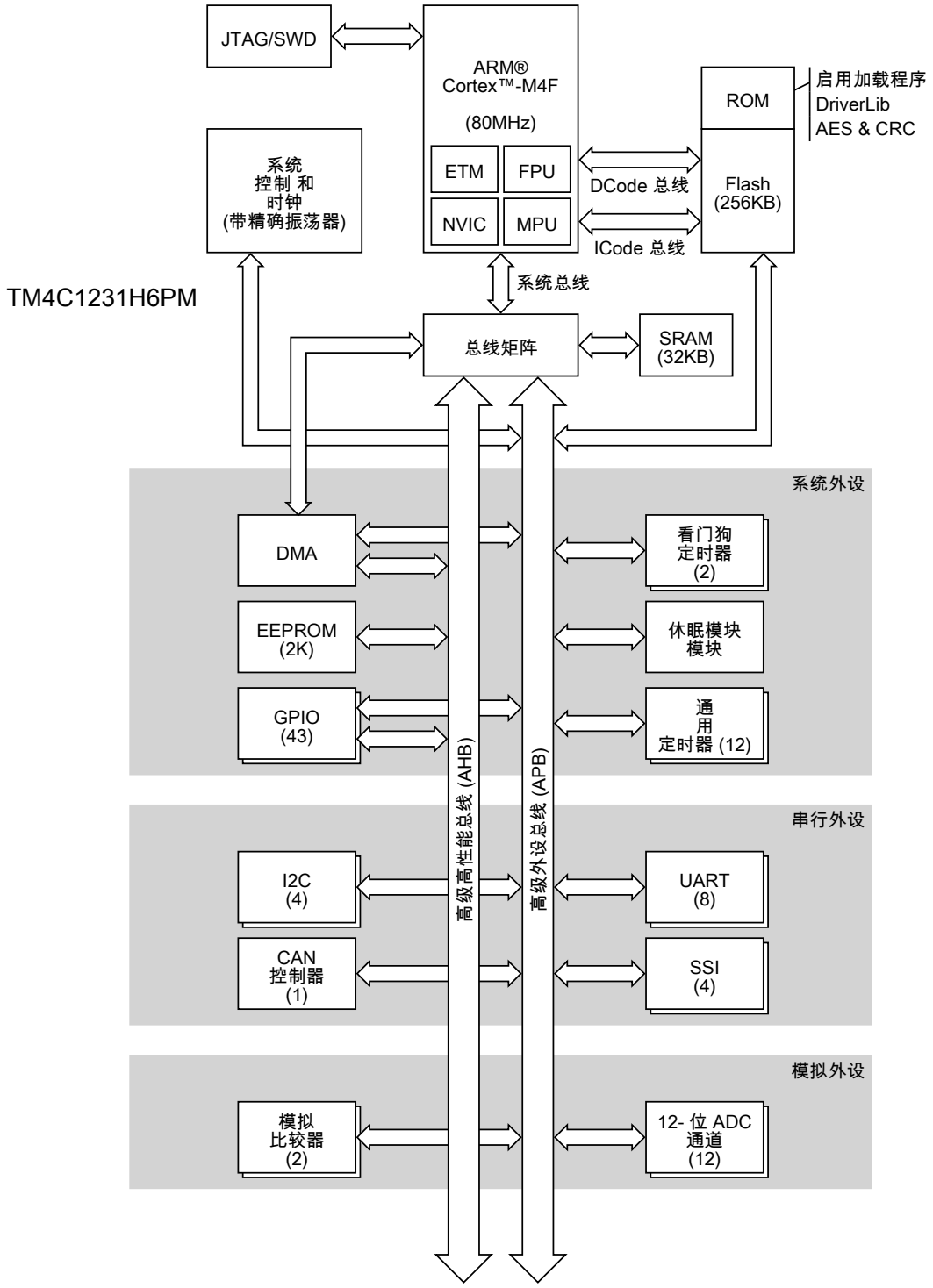
TM4C1231H6PM 微控制器结合了复杂集成功能和高性能，并具备如表1-1 中所示的各种特性。

表 **1-1. TM4C1231H6PM** 微控制器特性

特性	描述
性能	
内核	ARM Cortex-M4F 处理器核心
性能	80-MHz 运行速度；100 DMIPS 性能
Flash	256 KB 单周期 Flash 存储器
系统 SRAM	32 KB 单周期访问的 SRAM
EEPROM	2KB EEPROM
内置 ROM	搭载 TivaWare™（适用于 C 系列）软件的内置 ROM
安全性	
通信接口	
通用异步收发器 (UART)	八 个 UART
同步串行接口（SSI）	四 个 SSI 模块
内部集成电路 (I^2C)	4 个 I^2C 模块，具有四种传输速率（包括高速模式）
控制器局域网 (CAN)	个 CAN 2.0 A/B 控制器
系统集成	
微型直接存储器访问 (µDMA)	ARM® PrimeCell® 32 通道的可配置 µDMA 控制器
通用定时器 (GPTM)	6 个 16/32 位 GPTM 模块和 6 个32/64 位宽 GPTM 模块
看门狗定时器 (WDT)	2 个看门狗定时器
休眠模块 (HIB)	带备用电池的低功耗休眠模块
通用输入/输出端口 (GPIO)	6 个物理 GPIO 模块
模拟支持	
模-数转换器（ADC）	2 个 12 位 ADC 模块，每个的最大采样速率达 1M 次采样每秒
模拟比较器控制器	两个独立集成的模拟比较器
数字比较器	16 个数字比较器
JTAG 和串行线调试 (SWD)	一个 JTAG 模块，带集成的 ARM SWD
封装信息	
封装	64 管脚 LQFP
工作温度范围（环境）	工业（-40°C 到 85°C）温度范围

图1-1（36页）显示了 TM4C1231H6PM 微控制器的特性。请注意有两条片内总线连接内核和外设。高级外设总线 (APB) 是旧系统的总线，高级高性能总线 (AHB) 提供比 APB 总线更好的背靠背的访问性能。

图 1-1. Tiva™ TM4C1231H6PM 微控制器高级框图



1.3 TM4C1231H6PM 微控制器特性

有关 TM4C1231H6PM 微控制器元件特性和通用功能的更多详细信息请参考下面的章节。

1.3.1 ARM Cortex-M4F 处理器核心

德州仪器 Tiva™ C 系列的所有产品（包括 TM4C1231H6PM 微控制器）都是围绕 ARM Cortex-M 处理器内核设计的。这款 ARM Cortex-M 处理器为高性能、低成本的平台提供了一个满足最小存储需求、简化管脚数以及低功耗三方面要求的内核，与此同时，它还提供出色的计算性能和卓越的系统中断响应能力。

1.3.1.1 处理器内核 （见第**53**页）

- 32 位 ARM Cortex-M4F 架构针对小封装嵌入式应用进行了优化
- 80-MHz 运行速度；100 DMIPS 性能
- 优越的处理性能和更快的中断处理。
- 混合 16 位/32 位的 Thumb-2 指令集提供与 32 位 ARM 内核所期望的高性能而采用了更紧凑的内存大小，而这通常在 8 位和 16 位设备相关的存储容量中，特别是在微控制器级应用的几千字节存储中。
  - 单周期乘法指令和硬件除法器
  - 精确的位操作（bit-banding），不仅最大限度的利用了存储器空间而且还改良了对外设的控制
  - 非对齐式数据访问，使数据能够更为有效的安置到存储器中
- 符合 IEEE754 的单精度浮点单元 (FPU)
- 16 位 SIMD 矢量处理单元
- 快速代码执行允许更低的处理器时钟和增加休眠模式时间
- Harvard 结构 - 将数据和指令所使用的总线进行了分离
- 高效的处理器内核，系统和存储器
- 硬件除法和以快速数字信号处理为导向的乘加
- 采用饱和算法处理信号
- 对时间苛刻的应用提供可确定的, 高性能的中断处理
- 存储器保护单元为操作系统机能提供特权操作模式
- 增强的系统调试提供全方位的断点和跟踪能力
- 串行线调试和串行线跟踪减少调试和跟踪过程中需求的管脚数
- 从 ARM7™ 处理器系列中移植过来，以获得更好的性能和电源效率
- 针对高达指定频率的单周期 Flash 存储器使用情况而设计。详见 “内部存储器”（458页）。

- 集成多种休眠模式，更低功耗

1.3.1.2 系统定时器 (SysTick)（见第101页）

ARM Cortex-M4F 包含一个集成的系统定时器 SysTick。SysTick 提供了一种简单的、24 位写 1 清 0 (clear-on-write)、递减的、自加载 (wrap-on-zero) 的计数器，同时具有灵活的控制机制。该计数器可被用在不同的方面，比如：

- 用作 RTOS 的节拍定时器，按照可编程的频率（例如，100Hz）定时触发，调用系统定时器服务子程序
- 用作高速报警定时器，采用系统时钟作为时钟源
- 用作频率可变的报警或信号定时器——其周期取决于所采用的参考时钟源以及计数器的动态范围
- 用作简单计数器，测量任务的完成时刻、总体耗时等等
- 基于未到达/到达时间的内部时钟源控制

1.3.1.3 嵌套式向量化中断控制器 (NVIC)（见第102页）

TM4C1231H6PM 控制器包含 ARM 嵌套向量中断控制器 (NVIC)。NVIC 和 Cortex-M4F 可以在“处理模式”中对所有的异常进行优先级划分并进行处理。异常发生时处理器状态被自动存储到堆栈，中断服务程序 (ISR) 结束时又自动被恢复。向量的读取与状态保存并行，高效率进入中断。处理器支持尾链 (tail-chaining)，这样使得执行背靠背中断不需要重叠的状态保存和恢复。可通过软件设置 7 个异常（系统处理）和 64 个中断的 8 级优先级。

- 明确、快速地处理中断：总是 12 周期或 6 周期（后者带尾部连接）（这些值不会体现 FPU 堆栈）
- 外部不可屏蔽中断信号 (NMI) 可用于安全关键应用
- 动态优先级中断
- 通过硬件实现所需的寄存器可灵活的处理特殊的中断

1.3.1.4 系统控制模块（SCB）（见第103页）

系统控制块（SCB）提供系统实现信息和系统控制，包括系统异常的配置，控制和报告。

1.3.1.5 存储器保护单元（MPU）（见第104页）

MPU 支持标准的 ARM7 受保护的内存系统架构（PMSA）模型。MPU提供保护区支持，重叠保护区，访问权限和导出存储属性到系统。

1.3.1.6 浮点单元 (FPU)（见第108页）

该 FPU 完全支持单精度的加法、减法、乘法、除法、乘累加和平方根操作。它还可用于转换定点和浮点数据格式，并提供浮点常数指令。

- 适用于单精度（C 浮点）数据处理操作的 32 位指令
- 适用于更高精度（熔合 MAC）的组合乘加指令
- 适用于换算、加法、减法、可选累加的乘法、除法和平方根计算的硬件支持
- 适用于反常和所有 IEEE 舍入模式的硬件支持

- 32 个专用的 32 位单精度寄存器，也可作为 16 位双字寄存器进行寻址
- 去耦三级流水线

1.3.2 片上存储器

TM4C1231H6PM 微控制器整合了以下片上存储器和特性：

- 32 KB 单周期访问的 SRAM
- 256 KB Flash 存储器
- 2KB EEPROM
- 搭载 TivaWare™（适用于 C 系列）软件包的内部 ROM：
  – TivaWare™ 外设驱动库
  – TivaWare 引导加载程序
  – 高级加密标准 (AES) 密码表
  – 循环冗余检验 (CRC) 错误检测功能

1.3.2.1 SRAM（见第459页）

TM4C1231H6PM 微控制器提供 32 KB 单周期片上 SRAM。器件的内部 SRAM 位于器件存储器映射的偏移量 0x2000.0000 处。

由于读-修改-写 (RMW) 操作是很耗时的，因此 ARM 将位带 (*bit-banding*) 技术引入到 Cortex-M4F 处理器中。在位带启用的处理器中，存储器映射的特定区域 (SRAM 和外设空间) 能够使用地址别名，在单个原子操作中访问各个位。

数据可由以下主机实现与 SRAM 的相互传输：

- µDMA

1.3.2.2 Flash 存储器（见第461页）

TM4C1231H6PM 微控制器提供 256 KB 单周期片上 Flash 存储器。Flash 存储器由一系列 1 KB 的块组织在一起，这些块可以被单独擦除。擦除一个块会将块中的所有位都复位为 1。这些区块配对后便组成了一组可分别进行保护的 2 KB 区块。该保护允许块被标记为只读或只执行，以提供不同等级的代码保护。只读块不能被擦除或编程，块的内容受保护不能修改。只执行块不能被擦除或编程，只能通过控制器取指机制来读取它的内容，块的内容受保护不能被控制器或调试器读取。

1.3.2.3 ROM（见第460页）

TM4C1231H6PM ROM 出厂时经过预编程，包含了下面的软件和程序：

- TivaWare 外设驱动库
- TivaWare 引导加载程序
- 高级加密标准 (AES) 密码表
- 循环冗余检验 (CRC) 错误检测功能

所述的 TivaWare 外设驱动库是具有引导加载能力的用于控制片上外设的免版税软件库。该库可用于外设初始化和控制功能，可选择轮询式和中断驱动式外设支持。另外，经过设计，这个库可以充分利用 ARM Cortex-M4F 核心的优异中端性能。不需要特殊的程序或者用户汇编代码开端/收尾功

能。在需要现场编程能力的应用中，免版税的 TivaWare 引导加载程序可用于应用程序加载，支持现场固件升级。

高级加密标准 (AES) 是美国政府使用的公开定义的加密标准。AES 是一种强大的加密方法，拥有不错的性能和大小。AES 在硬件和软件方面都很快，它非常容易使用，并且只需要很少的存储空间。其 Texas Instruments 加密数据包开放了所有源代码，并且建立在宽通用许可证 (LGPL) 源代码基础之上。LGPL 是指可在应用中使用，但不受任何 copyleft 的影响的代码（代码并不会自动变成开源代码）。然而对于该数据包源的修改必须是开源的。

循环冗余检查 (CRC) 是一项用来确认一段数据的内容与先前检验的数据是否相同的技术。CRC 技术可用来验证信息是否正确接收（在传送中没有丢失或改变）、解压后的数据、Flash 存储器的内容是否更改以及其它需要验证数据的情况。CRC 优于简单的校验和 (例如异或所有的位)，因为它更容易捕捉到变化。

1.3.2.4 EEPROM（见第466页）

TM4C1231H6PM 微控制器包含 1 个 EEPROM 单元，其特性如下：

- 可用 2K 字节的存储器，即 512 32 位字
- 32 个块区，每区 16 字（64 字节）
- 内置的换位写入技术
- 每个模块的访问保护
- 整个外设的锁定保护选项和每个块的锁定保护一样，都使用 32 位到 96 位的解锁代码（根据应用的需要选择）
- 支持写完成中断，避免轮询
- 每个 2 页面块可进行 500K 次写操作（按周期使用固定偏移量对隔页进行写操作时）到 15M 次操作（在两个页面之间循环时）。

1.3.3 串行通讯外设

TM4C1231H6PM 控制器支持同步和异步串行通信：

- CAN 2.0 A/B 控制器
- 8 个 UART，支持 IrDA、9 位以及 ISO 7816。
- 4 个 I^2C 模块，具有四种传输速率（包括高速模式）
- 4 个同步串行接口模块 (SSI)

下面的章节各个通讯功能的提供更多细节。

1.3.3.1 控制器局域网 (CAN)（见第947页）

控制器局域网 (CAN) 是一种用于连接电子控制设备（Electronic Control Unit，简写为ECU）的多播共享型串行总线标准。CAN 总线针对抗电磁干扰进行了专门设计，适用于具有较强电磁干扰的环境，不但可以使用与 RS-485 类似的差分平衡传输线，也可以使用更加可靠的双绞线。CAN 总线最初是针对汽车应用而研发的，不过时至今日已经广泛应用于各种嵌入式控制领域（例如工业方面和医疗方面）。CAN 总线在总线长度小于 40 米时最高可达 1 Mbps 位速率。位速率越低则有效通讯距离越远（例如125 kbps 时通讯距离可达 500 米）。

发送器向所有CAN结点发送一条报文（广播）。每个节点根据接收到的标识符来判断是否处理该报文。标识符还决定了总线访问竞争中报文享有的优先级。每个CAN报文都能够传输0到8个字节的用户信息。

TM4C1231H6PM 微控制器包括 1 个 CAN 单元，有如下特性：

- 支持 CAN 总线协议 2.0 A/B
- 位速率最高可达 1 Mbps
- 32 个报文对象，每个报文对象都具有独立的标识符掩码
- 可屏蔽中断；
- 支持禁用自动重新发送（Disable Automatic Retransmission，简写为 DAR）模式，因此可用于时间触发 CAN（Time Triggered CAN，简写为 TTCAN）应用
- 可编程的回送模式，用于实现自检
- 可编程的 FIFO 模式，可使能多个报文对象存储
- 通过 CANnTX 和 CANnRX 管脚无缝连接片外 CAN 收发器

1.3.3.2 UART（见第799页）

通用异步收发器 (UART) 用于 RS-232C 串行通讯的集成电路。包含可异步工作的发送器（并转串）和接收器（串转并）。

TM4C1231H6PM 微控制器包含八个完全可编程的 16C550 型 UART。虽然其功能与 16C550 UART 相似，但并不是寄存器兼容的。UART 能够根据 RX、TX、调制解调器流量控制、和错误条件产生独立可屏蔽的中断。如果任何中断发生并且未被屏蔽，那么模块将生成一个组合的中断。

这八个 UART 如下特性：

- 可编程的波特率发生器，在常规模式（16 分频）下最高可达 5 Mbps，在高速模式（8 分频）下最高可达 10 Mbps
- 相互独立的 16×8 发送 (TX) FIFO和接收 (RX) FIFO，可降低中断服务对 CPU 的占用
- FIFO 长度可编程，包括提供传统双缓冲接口的 1 字节深的操作
- FIFO触发深度有如下级别可选：1/8、1/4、1/2、3/4或7/8；
- 标准的异步通讯位：起始位、停止位、奇偶校验位；
- 线中止的产生与检测；
- 完全可编程的串行接口特性
  - 可包含5、6、7或8个数据位
  - 可产生/检测奇偶校验位，支持偶校验位、奇校验位、粘着校验位或无校验位
  - 可产生1或2个停止位
- IrDA串行红外（SIR）编解码器
  - 可选择采用 IrDA 串行红外 (SIR) 输入输出或普通 UART 输入输出

- 支持 IrDA SAR 编解码功能，半双工时数据传输率最高 115.2 Kbps
- 支持标准的 3/16 位时间以及低功耗位时间（1.41~2.23 µs）
- 可编程的内部时钟发生器，可对参考时钟源进行1~256分频以提供低功耗位时间

■ 支持与 ISO 7816 智能卡的通讯

■ 调制解调器流量控制和状态（在 UART1 模块上）

■ 支持 EIA-485（9 位）

■ 提供标准的基于 FIFO 深度的中断以及发送结束中断

■ 用微型直接内存访问 (µDMA) 有效的传输数据

- 相互独立的发送通道和接收通道
- 当接收 FIFO 中有数据时产生单次请求；当接收 FIFO 到达预设的触发深度时产生猝发请求
- 当发送 FIFO 中有空闲单元时产生单次请求；当发送 FIFO 到达预设的触发深度时产生猝发请求

1.3.3.3 I^2C （见第**899**页）

内部集成电路 (I^2C) 总线通过一个两线设计（串行数据线 SDA 和串行时钟线 SCL）来提供双向数据传输。I^2C 总线可与诸如串行存储器（RAM 和 ROM）、网络设备、LCD、音频发生器等外部 I^2C 器件相连。I^2C 总线还可在产品开发和制造过程中用于系统测试和诊断。

I^2C 总线上的设备可被指定为主机或从机。I^2C 模块支持作为主机或从机来发送和接收数据，也支持既用作主机又用作从机的同步操作。I^2C 主机和从机都能够产生中断。

TM4C1231H6PM 微控制器包含四个具有以下特性的 I^2C 模块：

■ I^2C 总线上的设备可被指定为主机或从机

- 在主机或从机模式下都支持发送和接受数据
- 支持它们作为主机和从机的同步操作

■ 四种 I^2C 模式：

- 主机发送
- 主机接收
- 从机发送
- 从机接收

■ 四种传输速度：

- 标准 (100 Kbps)
- 快速 (400 Kbps)
- 超快速 (1 Kbps)

  - 高速 (3.33 Mbps)
- 时钟低电平超时中断
- 双从机地址功能
- 故障抑制
- 主机和从机产生中断
  - 主机因为传送或接收数据结束(或者是因为错误而取消)产生中断
  - 从机在主机向其发送数据或发出请求时，或检测到START或STOP信号时产生中断。
- 主机带有仲裁和时钟同步功能，支持多主机以及 7 位寻址模式

1.3.3.4 SSI （见第856页）

同步串行接口 (SSI) 是一个用于将数据并转串的 4 线双向型通信接口。此 SSI 模块对从外围器件接收到的数据执行串并转换，对发送到外围设备的数据执行并串转换。这个 SSI 模块可以配置用作主设备或从设备。作为从机设备的时候，还可以通过配置将 SSI 模块的输出禁能，从而使一个主设备可以与多个从设备相连。TX 和 RX 的通路都有内部 FIFO 负责缓冲。

此 SSI 模块还包含一个可编程的位速率时钟分频器和预分频器，SSI 模块输入的时钟信号将通过它们来生成 SSI 输出的串行时钟信号。位速率根据输入时钟产生，最大位速率取决于连接的外设。

TM4C1231H6PM 微控制器包含四个具有以下特性的 SSI 模块：

- 提供可编程控制的接口，可与 Freescale SPI、MICROWIRE 或者 Texas Instruments 同步串行接口相连
- 主机或从机工作方式；
- 可编程的时钟位速率以及预分频器；
- 相互独立的发送 FIFO 和接收 FIFO，二者均为 16 位宽、8 个单元深；
- 可编程的数据帧长度，4位到16位可选；
- 内部环回测试模式，能够很方便地实现诊断/调试；
- 标准 FIFO 中断以及发送结束中断；
- 用微型直接内存访问 (µDMA) 有效的传输数据
  - 相互独立的发送通道和接收通道
  - 当接收 FIFO 中有数据时产生单次请求；当接收 FIFO 中包含 4 个数据单元时产生猝发请求
  - 发送单次请求在 FIFO 有空闲单元时有效；发送猝发请求在有 4 个及以上条目可以写入 FIFO 中时有效

1.3.4 系统集成

TM4C1231H6PM 微控制器将各种标准系统功能集成到设备中，包括：

- 直接存储器访问控制器 (DMA)

- 系统控制和时钟，包括片上 16-M 高精度振荡器
- 6 个 32 位定时器（最多可达 12 个 16 位定时器）
- 六个 64 位宽定时器（最多可达 12 个 32 位宽定时器）
- 十二个 32/64 位捕获比较 PWM (CCP) 管脚
- 带备用电池的低功耗休眠模块
- 在休眠模块中的实时时钟
- 两个看门狗定时器
  - 一个定时器使用主时钟振荡器
  - 一个定时器使用内部时钟振荡器
- 高达 43 个 GPIO，具体取决于配置
  - 高度灵活的管脚复用，可配置为GPIO或任一外设功能
  - 可独立配置为 2、4 或 8 mA 端口驱动能力
  - 高达 4 个 GPIO，具有 18 mA 驱动能力

下面的章节提供这些功能的更多信息。

1.3.4.1 直接内存访问 （见第**512**页）

TM4C1231H6PM 微控制器内置一个直接存储器访问（Direct Memory Access，简写为DMA）控制器，我们称之为微型 DMA (µDMA) 控制器。µDMA 控制器所提供的工作方式能够分载 Cortex-M4F 处理器参与的数据传输任务，从而使处理器得到更加高效的利用和腾出更多的总线带宽。µDMA 控制器能够自动执行存储器与外设之间的数据传输。片上每个支持 µDMA 功能的外设都有专用的 µDMA 通道，通过合理的编程配置，当外设需要时能够自动在外设和存储器之间传输数据。µDMA 控制器具有以下特性：

- ARM PrimeCell® 32 通道的可配置 µDMA 控制器；
- 支持存储器到存储器、存储器到外设、外设到存储器的 µDMA 传输，包括：
  - 基本模式，用于简单的传输需求
  - 乒乓模式，用于实现持续数据流
  - 散聚模式，借助一个可编程的任务列表，由单个请求触发多达 256 个指定传输
- 高度灵活的可配置的通道配置；
  - 各通道均可独立配置、独立操作
  - 每个支持 µDMA 功能的片上模块都有其专用通道
  - 灵活的通道分配
  - 对于双向模块，为其接收和发送各提供一个通道
  - 专用的软件通道，可由软件启动 µDMA 传输
  - 每通道都可分别配置优先级

  - 可选配置：任一通道均可用作软件启动传输

- 优先级分为两级；
- 通过优化设计，改进了 µDMA 控制器与处理器内核之间的总线访问性能：
  - 当内核不访问总线时，µDMA 控制器即可占用总线
  - RAM条带处理
  - 外设总线分段
- 支持 8 位、16 位或 32 位数据宽度
- 待传输数目可编程为 2 的整数幂，有效范围 1 到 1024
- 源地址及目的地址可自动递增，递增单位可以是字节、半字、字、不递增
- 可屏蔽的外设请求
- 传输结束中断，且每个通道有独立的中断

1.3.4.2 系统控制及时钟 （见第**186**页）

系统控制决定了器件的所有操作它不但提供了有关器件的信息，控制节能特性、对器件和单个外设的时钟进行控制，还处理复位的检测和报告。

- 器件识别信息：版本、部件号、SRAM 大小、Flash 存储器大小等
- 功率控制
  - 片上低压差线性稳压器 (LDO)
  - 3.3 V 通电/断电序列，并控制内核的数字逻辑和模拟电路
  - 微控制器的低功耗选项：带有时钟门控的休眠和深度休眠模式
  - 片上模块的低功耗选择：软件控制各独立外设和存储器的掉电
  - 3.3 V 电源掉电检测，可通过中断或复位来报告
- 微控制器的系统时钟有多个时钟源。TM4C1231H6PM 微控制器配有以下时钟源：
  - 精确内部振荡器 (PIOSC)，提供 16-MHz 频率
    - 整个温度和电压范围 16 MHz ±3%
    - 可用 7 位的调整分辨率进行校准，实现更高的准确度 (16 MHz ±1%)
    - 低功耗模式可软件控制掉电
  - 主振荡器 (MOSC)：主振荡器可通过两种方式提供一个频率精确的时钟源：外部单端时钟源连接到 OSC0 输入管脚，或者外部晶振串接在 OSC0 输入管脚和 OSC1 输出管脚间。
  - 低频内部振荡器 (LFIOSC)：在节电模式期间使用的片上时钟源
  - 可配置为休眠 (HIB) 模块 32.768-kHz 外部振荡器时钟源的休眠 RTC 振荡器 (RTCOSC) 时钟，或者位于休眠模块中的 HIB 低频时钟源 (HIB LFIOSC)。
- 灵活的复位源

- 上电复位 (POR)
- 复位管脚有效
- 欠压 (BOR) 警告系统电源掉电
- 软件复位
- 看门狗定时器复位
- 主振荡器 (MOSC) 故障

1.3.4.3 可编程定时器 （见第628页）

可编程定时器可对驱动定时器输入管脚的外部事件进行计数或定时。每个 16/32 位 GPTM 模块包含两个 16 位的定时器/计数器，用户可以将它们配置成独立运行的定时器或事件计数器，或将它们配置成一个 32 位定时器或一个 32 位实时时钟 (RTC)。每个 32/64 位宽 GPTM 模块包含两个 32 位的定时器/计数器（称作 TimerA 和 TimerB），用户可以将它们配置成独立运行的定时器或事件计数器或将它们配置成一个 64 位定时器或一个 64 位实时时钟 (RTC)。定时器也可以用来触发模数转换 (ADC) 和 DMA 传输。

通用定时器模块 (GPTM) 包含六个 16/32 位 GPTM 时钟以及六个 32/64 位宽 GPTM 时钟，有以下功能选项：

- 16/32 位运行模式：
  - 16 位或 32 位可编程的单次定时器
  - 16 位或 32 位可编程的周期定时器
  - 具有 8 位预分频的 16 位通用定时器
  - 当有 32.768 KHz 的外部时钟源时可作为 32 位的实时时钟
  - 16 位输入沿计数或定时捕获模式，并带 8 位的预分频器
  - 带 8 位预分频器的 16 位 PWM 模式以及软件编程实现的 PWM 信号反相输出
- 32/64 位运行模式：
  - 32 位或 64 位可编程的单次定时器
  - 32 位或 64 位可编程的周期定时器
  - 具有 16 位预分频的 32 位通用定时器
  - 当有 32.768 KHz 的外部时钟源时可作为 64 位的实时时钟
  - 带有 16 位预分频器的 32 位输入沿计数或定时捕获模块
  - 带有 16 位预分频器 的 32 位 PWM 模式以及软件编程实现的 PWM 信号反相输出
- 可以向上或向下计数
- 十二个 16/32 位捕获比较 PWM 管脚 (CCP)
- 十二个 32/64 位捕捉比较 PWM 管脚 (CCP)

- 菊花链式的定时器模块允许一个定时器开始计时多路时钟事件
- 定时器同步功能允许所选的定时器在同一时钟周期开始计数
- 模数转换(ADC)触发器
- 当调试时，CPU 出现暂停标识时，用户可以停止定时器事件（包括 RTC 模式）
- 可以确定从产生定时器中断到进入中断服务程序所经过的时间
- 用微型直接内存访问 (µDMA) 有效的传输数据
  - 每个定时器具有专用通道
  - 定时器中断响应突发请求

1.3.4.4 CCP 管脚 （见第636页）

捕获/比较/PWM 管脚 (CCP) 可以被通用定时器模块使用，通过将 CCP 管脚作为输入，可定时/计数外部事件。此外，GPTM 模块可以在 CCP 管脚上产生一个简单的 PWM 输出。

TM4C1231H6PM 微控制器包括十二个 16/32 位 CCP 管脚，通过编程，这些管脚能够按以下模式运行：

- 捕捉：通用定时器根据 CCP 输入的编程事件递增/递减。当编程事件发生时，它捕获和存储当前定时器的值。
- 比较：通用定时器根据 CCP 输入的编程事件递增/递减。它将当前定时器的值和存储的值相比较，当匹配时，产生中断信号。
- PWM：通用定时器根据系统时钟递增/递减。PWM信号的产生基于计数值和存储值的匹配，并输出到 CCP 脚。

1.3.4.5 休眠模块 (HIB)（见第429页）

休眠模块提供了一种逻辑，将主处理器和外设电源暂时关闭，在外部事件或基于时间的事件发生时唤醒休眠模块包括上电顺序逻辑，有如下特性：

- 32 位实时秒计数器 (RTC)，其时钟分辨率是 1/32,768 秒；以及一个 15 位亚秒计数器
  - 32 位的 RTC 秒匹配寄存器，以及一个 15 位的亚秒匹配寄存器（其时钟分辨率是 1/32,768 秒），用于定时唤醒和产生中断
  - RTC 预分频器调整，对时钟速率进行良好地调节
- 电源控制的两种机制：
  - 使用离散的外部稳压器控制系统电源
  - 使用在寄存器控制下的内部开关控制片上电源
- 使用外部信号用作唤醒的专用管脚
- 只要 V_{DD} 或 V_{BAT} 有效，RTC 运行的内存和休眠所占内存就一直有效
- 电池电量低检测、发出信号和中断发生；在电量低时提供可选的唤醒操作

- GPIO 管脚的状态在休眠过程中可保持不变
- 时钟源可以是 32.768 KHz 的外部晶振或振荡器
- 16 个 32 位字的带备用电池存储器，用于在休眠过程中保存状态
- 可为以下事件设置中断信号：
  - RTC匹配
  - 外部唤醒
  - 电池电量过低

1.3.4.6 看门狗定时器 （见第692页）

当系统由于软件错误或是由于因外部设备故障而无法按预期的方式响应的时候，使用看门狗定时器可以重新获得控制权。TM4C1231H6PM 看门狗定时器在到达超时值时，可能会产生一个中断、一个不可屏蔽的中断或者复位。此外，看门狗定时器兼容 ARM FIRM，可配置成在第一次超时时产生中断请求并发送到微控制器，并在第二次超时时产生复位信号。配置好看门狗定时器后，即可写入锁定寄存器，从而防止定时器配置被意外更改。

TM4C1231H6PM 微控制器含有两个看门狗定时器模块：看门狗定时器 0 使用系统时钟计时；看门狗定时器 1 由 PIOSC 定时器时钟驱动。看门狗定时器模块具有如下特性：

- 32位递减并且可编程装载的寄存器
- 独立的看门狗时钟使能
- 带中断屏蔽功能和可选 NMI 功能的可编程中断产生逻辑
- 软件跑飞时保护锁定寄存器
- 复位使能/禁止产生逻辑
- 调试期间，微控制器的 CPU 暂停时，用户可启用的停滞

1.3.4.7 可编程的 GPIO （见第575页）

通用输入/输出 (GPIO) 管脚为各种连接方式带来了灵活性。TM4C1231H6PM GPIO 模块包含 6 个物理 GPIO 块，每个块对应一个独立的 GPIO 端口。GPIO 模块遵循 FiRM 规范（遵循 ARM 实时微控制器底层 IP 规范）并支持 TM4C1231H6PM 的可编程输入/输出管脚。可用的 GPIO 数目取决于正在使用的外设（有关每个 GPIO 管脚可用的信号见 “信号表”（1008页））

- 高达 43 个 GPIO，具体取决于配置
- 高度灵活的管脚复用，可配置为GPIO或任一外设功能
- 配置为输入模式可承受 5 V 电压
- 端口 A-G 可通过高级外设总线 (APB) 访问
- 快速切换能力，在 AHB 端口每个时钟周期实现一次变化；在 APB 端口每两个时钟周期实现一次变化
- 可编程控制的GPIO中断

  - 产生中断屏蔽
  - 上升沿、下降沿或是双边沿(上升沿和下降沿)触发
  - 高电平或低电平触发
- 读写操作时刻可过地址线进行位屏蔽的操作
- 可用于启动一个 ADC 采样序列或 µDMA 传输
- 在休眠模式中，可以保持管脚的状态
- 配置为数字输入的管脚均为施密特触发
- 可编程控制的GPIO引脚配置
  - 弱上拉或下拉电阻
  - 数字通信时可配置为 2 mA、4 mA 或 8 mA 驱动电流；对于需要大电流的应用，可通过多达四个管脚承载 18 mA
  - 8 mA 驱动电流的斜率控制
  - 开漏启用
  - 数字输入启用

1.3.5 模拟

TM4C1231H6PM 微控制器将模拟功能集成到芯片中，包括：

- 2 个 12 位模数转换器 (ADC)，总共带有 12 个模拟输入通道；每个的采样速率为 1M 采样/秒
- 两个模拟比较器
- 片上电压稳压器

下面的章节提供模拟功能的更多信息。

1.3.5.1 ADC （见第717页）

模-数转换器（ADC）是一种能够将连续的模拟电压信号转换为离散的数字量的外设。TM4C1231H6PM ADC 模块具有 12 位转换精度并支持 12 个输入通道；同时还内置一个温度传感器。4 个带缓冲的采样序列无需使用控制器，就可以对最多 12 个模拟输入源进行快速采样。每个采样序列发生器都可灵活配置其输入源、触发事件、中断的产生、序列发生器的优先级等内容。每个 ADC 块内置数字比较器功能，采样转换结果可移交给数字比较器模块，该数字比较器模块内置 8 路数字比较器。

TM4C1231H6PM 微控制器内置 2 个 ADC 模块，每个模块均具有以下特性：

- 12 个共用模拟输入通道
- 12 位精度的 ADC
- 可配置为单端输入或差分输入；
- 片上内置温度传感器

- 1M 次/秒的采样率
- 可选的移相器，采样点以采样周期计可延后 22.5° 到 337.5°
- 4 个可编程的采样转换序列发生器，序列长度 1 到 8 个单元不等，且各自带有相应长度的转换结果 FIFO
- 灵活的转换触发控制：
  - 控制器（软件）触发
  - 定时器触发
  - 模拟比较器触发
  - GPIO
- 硬件可对多达 64 个采样值进行平均计算
- 八个数字比较器
- 模拟部分的电源/地与数字部分的电源/地相互独立
- 用微型直接内存访问 (µDMA) 有效的传输数据
  - 每个采样序列发生器各自有专用的通道
  - ADC 模块的 DMA 操作均采用猝发请求

1.3.5.2 模拟比较器 （见第**993**页）

模拟比较器是一个外设，它能比较两个模拟电压的大小，并通过自身提供的逻辑输出端将比较结果以信号的形式输出。该 TM4C1231H6PM 微控制器提供两个独立集成的模拟比较器，该模拟比较器可配置为驱动输出或产生中断或 ADC 事件。

比较器可以向器件管脚提供输出，以替换板上的模拟比较器。比较器也可以通过中断或触发 ADC 通知应用让它开始捕获采样序列。中断产生逻辑和 ADC 触发是各自独立的。这就意味着，中断可以在上升沿产生，而 ADC 在下降沿触发。

TM4C1231H6PM 微控制器提供两个独立集成的模拟比较器，具有如下功能：

- 可以比较外部输入管脚和外部输入管脚或内部可编程的参考电压
- 比较器可将测试电压与下面的其中一种电压相比较
  - 独立的外部参考电压
  - 共用的外部参考电压
  - 共用的内部参考电压

1.3.6 JTAG 和 ARM 串行线调试 （见第**175**页）

联合测试行动组 (JTAG) 是一项 IEEE 标准，它定义了数字集成电路的测试访问端口和边界扫描结构，并且提供了一个标准化的串行接口来控制相关的测试逻辑。TAP、指令寄存器 (IR) 和数据寄存器 (DR) 可用来测试组装好的印刷电路板的互连性，并获取元件的制造信息。JTAG 端口还可用于访问和控制测试用设计的特性，如 I/O 管脚的观察和控制、扫描测试以及调试。 Texas Instruments 实

现了通过 ARM 串行线 JTAG 调试端口 (SWJ-DP) 接口取代 ARM SW-DP 和 JTAG-DP 的功能。SWJ-DP 接口将 SWD 和 JTAG 调试端口集成到一个模块中，提供所有的正常 JTAG 调试和测试功能，可实时访问系统存储器而不停止内核，且不需要任何目标寄居代码。SWJ-DP 接口具有如下特性：

- IEEE 1149.1-1990 兼容的测试访问端口 (TAP) 控制器
- 4 位指令寄存器 (IR) 链，用于存储 JTAG 指令
- IEEE 标准指令：BYPASS、IDCODE、SAMPLE/PRELOAD 和 EXTEST
- ARM 附加指令：APACC、DPACC 和 ABORT
- 集成的 ARM 串行线调试 (SWD)
  - 串行线 JTAG 调试端口 (SWJ-DP)
  - Flash 修补和断点 (FPB) 单元，用于实现断点操作
  - 数据观察点和触发 (DWT) 单元，用于执行观察点、触发源和系统性能分析
  - 仪表跟踪宏单元 (ITM)，用于支持 printf 型调试
  - 用于指令追踪捕捉的嵌入式追踪宏单元 (ETM)
  - 跟踪端口接口单元 (TPIU) 用作跟踪端口分析仪的桥接

1.3.7 封装和温度

- 符合 RoHS 标准的 64 管脚 LQFP 封装
- 工业（-40°C 到 85°C）环境温度范围

1.4 TM4C1231H6PM 微控制器硬件细节

有关管脚和封装的详细信息可在下一节中找到：

- “管脚图”（1007页）
- “信号表”（1008页）
- “Electrical Characteristics”（1029页）
- “封装信息”（1070页）

1.5 开发套件

其 Tiva™ C 系列 为工程师提供了各种硬件和软件工具，以便快速开始进行产品研发。

- 参考设计套件通过提供可以运行的硬件和完整的文档来加速产品开发，这些完整的文档包含了硬件设计文件
- 评估套件方便您在购买之前以低廉的成本有效评估 TM4C1231H6PM 微控制器
- 开发套件为你提供所有需要开发的工具和原型嵌入式应用

请访问 Tiva 系列的网站 http://www.ti.com/tiva-c 了解最新的可用工具，或者请直接联系分销商。

1.6 支持信息

关于 Tiva™ C 系列 产品的支持服务，请就近联系 TI Worldwide Product Information Center。

2 Cortex-M4F 处理器

ARM® Cortex™-M4F 处理器提供了一个高性能、低成本的平台，可满足系统对降低存储需求、简化管脚数以及降低功耗三方面的要求，与此同时，它还提供出色的计算性能和优越的系统中断响应能力。特性包括：

- 32 位 ARM® Cortex™-M4F 架构针对小封装嵌入式应用进行了优化
- 80-MHz 运行速度；100 DMIPS 性能
- 优越的处理性能和更快的中断处理。
- 混合 16 位/32 位的 Thumb-2 指令集提供与 32 位 ARM 内核所期望的高性能而采用了更紧凑的内存大小，而这通常在 8 位和 16 位设备相关的存储容量中，特别是在微控制器级应用的几千字节存储中。
  - 单周期乘法指令和硬件除法器
  - 精确的位操作（bit-banding），不仅最大限度的利用了存储器空间而且还改良了对外设的控制
  - 非对齐式数据访问，使数据能够更为有效的安置到存储器中
- 符合 IEEE754 的单精度浮点单元 (FPU)
- 16 位 SIMD 矢量处理单元
- 快速代码执行允许更低的处理器时钟和增加休眠模式时间
- Harvard 结构 - 将数据和指令所使用的总线进行了分离
- 高效的处理器内核，系统和存储器
- 硬件除法和以快速数字信号处理为导向的乘加
- 采用饱和算法处理信号
- 对时间苛刻的应用提供可确定的, 高性能的中断处理
- 存储器保护单元为操作系统机能提供特权操作模式
- 增强的系统调试提供全方位的断点和跟踪能力
- 串行线调试和串行线跟踪减少调试和跟踪过程中需求的管脚数
- 从 ARM7™ 处理器系列中移植过来，以获得更好的性能和电源效率
- 针对高达指定频率的单周期 Flash 存储器使用情况而设计。详见 “内部存储器”（458页）。
- 集成多种休眠模式，更低功耗

然后 Tiva™ C 系列 微控制器基于此内核之上，提供了高性能的 32 位运算能力

本章提供关于 Tiva™ C 系列 Cortex-M4F 处理器的执行信息，包括编程模块、存储器模块、异常模块、故障处理和电源管理。

关于指令集的技术细节，请参考“ ARM® Cortex™-M4 Devices Generic User Guide（文献编号 ARM DUI 0553A ）”中的 Cortex™-M4 指令集章节。

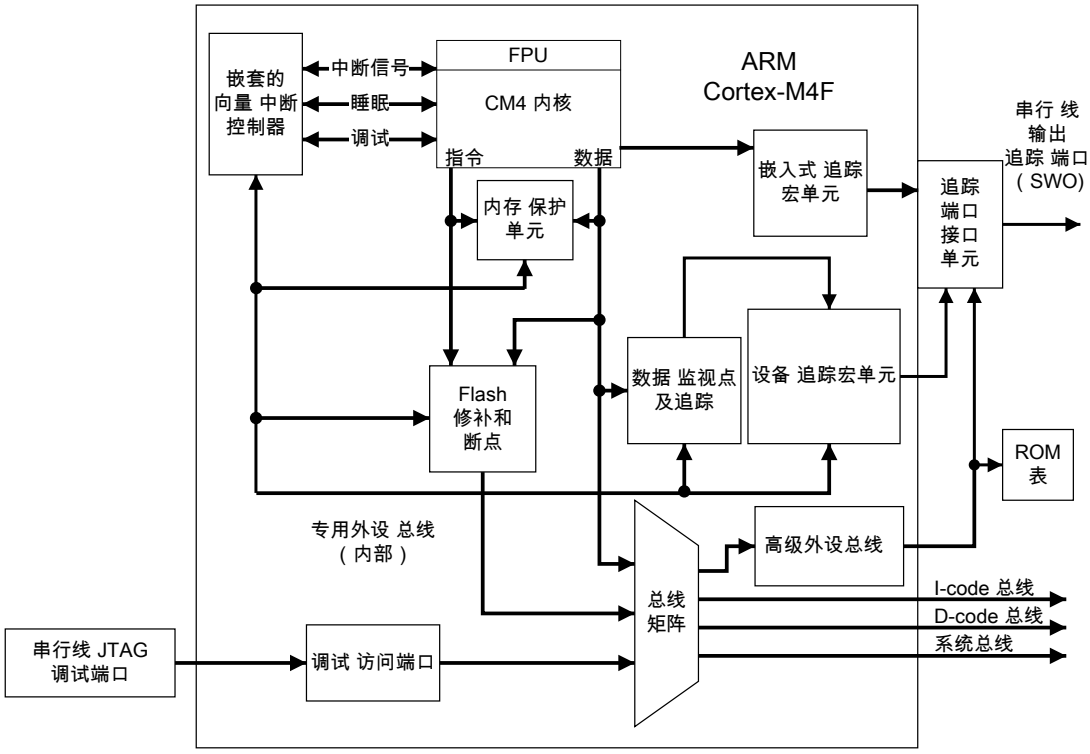
2.1 结构框图

Cortex-M4F 处理器基于高性能的处理器内核，采用三级流水线的哈佛架构，是满足嵌入式应用的理想的处理器。该处理器带有高效的指令集和特别优化的设计，具有优异的能耗效率，并提供符合 IEEE754 的单精度浮点型计算单元、一系列单周期和 SIMD 乘法器和乘加功能，以及专用的硬件除法器等高端处理硬件。

为促进成本敏感型设备的设计，Cortex-M4F 处理器采用了紧耦合的系统部件以降低处理器尺寸，同时显著提高了中断处理能力和系统调试能力。Cortex-M4F 处理器采用了基于 Thumb-2 技术的 Thumb® 指令集，确保高代码密度和降低程序存储需求。Cortex-M4F 采用现代 32 位架构和 8 位、16 位微处理器的高密度指令集，因而性能优异。

Cortex-M4F 处理器高度集成了嵌入式中断控制器 (NVIC)，可达到业界领先的中断性能。TM4C1231H6PM NVIC 包括一个不可屏蔽中断 (NMI)，并提供 8 个中断优先级。紧密集成的处理器内核和NVIC提供快速的中断服务程序和显著的降低了中断延迟。硬件入栈和停止多步装载和存储操作进一步降低了中断延迟。中断处理不需要任何的汇编从何减少了ISR的代码开销。尾链优化同样显著地降低了ISR切换时的开销。为优化低功耗设计，NVIC集成了睡眠模式，包括深度睡眠模式，该模式可使整个芯片迅速地降低功耗。

图 **2-1. CPU** 结构图



2.2 概述

2.2.1 系统级接口

Cortex-M4F 处理器采用 AMBA® 技术来提供多接口，以实现高速、低延迟的存储器访问。内核支持非对齐的数据访问和原子位操作，使得外设的控制，系统自旋锁和线程安全布尔数据处理更快。

Cortex-M4F 处理器内有一个存储器保护单元 (MPU)，可提供细粒度的存储器控制，使应用可以实现安全特权级别和基于各个任务的独立代码、数据和堆栈。

2.2.2 集成的可配置调试

Cortex-M4F处理器提供一个完整的硬件调试方案，通过一个传统的 JTAG 端口或者适合于微控制器和其他小封装设备的 2 脚串行线调试 (SWD) 端口来实现处理器和存储器的系统高度可观测性。所述的 Tiva™ C 系列 以兼容 ARM CoreSight™ 的串行线 JTAG 调试端口 (SWJ-DP) 接口取代了 ARM SW-DP 和 JTAG—DP。SWJ-DP 接口将 SWD 和 JTAG 调试端口组合到一个模块中。有关 SWJ-DP 的详细信息，请参考 “ARM® Debug Interface V5 架构规格”。

对于系统跟踪，处理器集成了一个仪表跟踪宏单元（ITM），具有数据断点和分析单元。能够简单地低成本的系统跟踪事件，串行线观测器（SWV）通过一个单引脚能导出软件产生的信息、数据跟踪和分析信息的数据流。

嵌入式跟踪宏单元 (ETM) 提供了优异的指令追踪能力，相比传统的追踪单元，其追踪范围更加精确，同时还提供了全指令追踪功能。更多有关 ARM ETM 的详细信息，请参考 “ARM® Embedded Trace Macrocell 架构规格”。

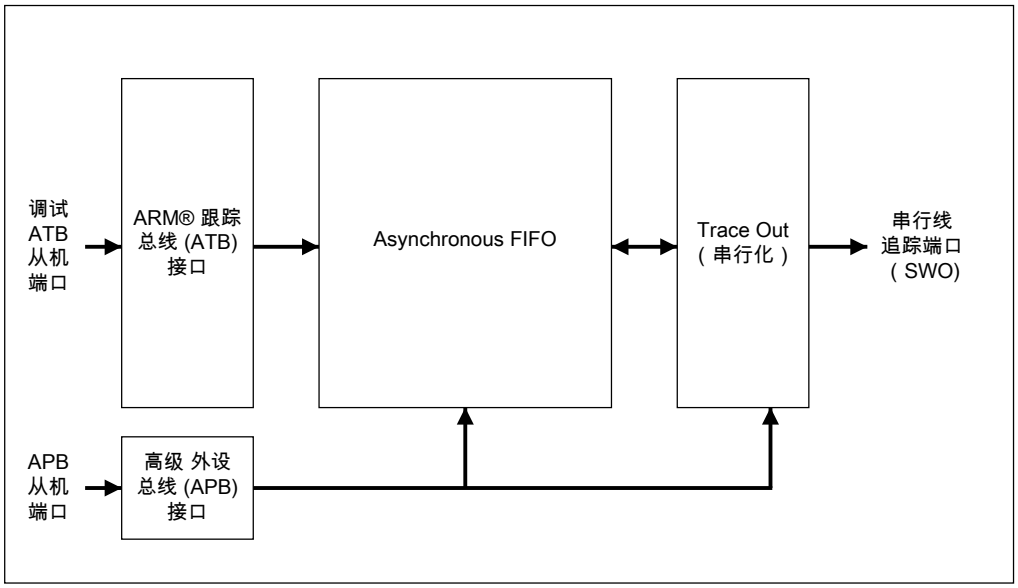
Flash补丁和断点单元（FPB）提供高达8个硬件断点比较仪，这些可被调试器使用。FPB 的这些比较仪还提供在程序代码中的代码内存区多达 8 个字的重映射功能。这个 FPB 允许存储在 Flash 存储器只读区的应用可以拼接到片上 SRAM 或 Flash 存储器的另一个区。当需要拼接时，应用编程FPB来重映射一组地址。当这些地址被访问时，访问被重定位到FPB配置中指定的重映射表中。

更多有关 Cortex-M4F 调试功能的信息，请参考 “ARM® Debug Interface V5 Architecture Specification”。

2.2.3 跟踪端口的接口单元（TPIU）

TPIU 充当来自 ITM 的 Cortex-M4F 跟踪数据以及片外跟踪端口分析仪之间的桥接器，如 图 2-2（56页） 中所示。

图 2-2. TPIU方框图



2.2.4 Cortex-M4F 系统组件细节

Cortex-M4F 包含以下系统组件：

- SysTick

  24 位的递减定时器，可被用作实时操作系统 (RTOS) 的节拍定时器，或者作为一个简单的计数器，参见 “系统定时器（SysTick）”（101页）。

- 嵌套式向量化中断控制器（NVIC）

  一个嵌入的中断控制器，支持低延迟中断处理，参见 “嵌套式向量化中断控制器（NVIC）”（102页）。

- 系统控制模块 (SCB)

  处理器的编程模型接口。系统控制块 (SCB) 提供系统实现信息和系统控制，包括系统异常的配置、控制和报告（请参考“系统控制模块 (SCB)”（103页））。

- 存储器保护单元 (MPU)

  通过为不同的内存区定义内存属性来提高系统的稳定性。MPU 提供多达 8 个不同区和一个可选的预定义的背景区，参见 “存储器保护单元 (MPU)”（104页）。

- 浮点单元 (FPU)

  完全支持单精度的加、减、乘、除、乘加以及平方根操作。它还可用于转换定点和浮点数据格式，并提供浮点常数指令（请参考 “浮点单元 (FPU)”（108页））。

2.3 编程模型

这部分描述了 Cortex-M4F 的编程模型。另外还有单个的内核寄存器描述，处理器模式的信息和软件执行、堆栈的权限级别。

2.3.1 处理器模式和软件执行的权限级别

Cortex-M4F 模块具有两种工作模式：

- 线程模式

  用于执行应用程序软件。处理器复位后，进入线程模式。

- 处理器模式

  用于处理异常。当处理器完成异常的处理之后返回到线程模式。

另外，Cortex-M4F 有两个权限级别：

- 无特权级

  在此模式下，软件有如下限制：

  – 限制访问 MSR 和 MRS 指令，且不使用 CPS 指令

  – 不能访问系统定时器、NVIC 或者系统控制块

  – 限制对某些存储器和外设的访问

- 特权级

  在此模式下，软件可以使用所有的指令和访问所有的资源。

在线程模式下，CONTROL 寄存器（参见 71页）控制软件执行是在特权级还是非特权级。在处理模式下，软件执行总是在特权级下。

在线程模式下，只有特权级软件可以写CONTROL 寄存器来改变软件的特权级。非特权级软件可使用 SVC 指令来产生一个系统调用，把控制权转移到特权级软件。

2.3.2 堆栈

该处理器使用向下的满栈，意味着在存储器中堆栈指针指向的是最后入栈项目。当处理器推入一个新的项目入栈时，先递减堆栈指针，再把新项目写入到内存中。处理器实现了两个堆栈：主堆栈和处理堆栈，每个堆栈的指针都包含于独立的寄存器中（请参考61页上的 SP 寄存器）。

在线程模式，CONTROL 寄存器（见 71页）控制处理器是使用主堆栈还是处理堆栈。在处理模式下，处理器总是使用主堆栈。处理器操作选项如 表2-1（57页） 所示。

表 2-1. 处理器模式、特权等级和堆栈使用摘要

处理器模式	用途	特权等级	堆栈使用
Thread	应用程序	特权级或非特权级 [a]	主堆栈或进程堆栈 [a]
Handler	异常处理程序	特权级	主堆栈

a. 参见 CONTROL（71页）。

2.3.3 寄存器映射

图2-3（58页） 描述了 Cortex-M4F 的寄存器组。表 2-2（58页） 列出了内核寄存器。核心寄存器并没有映射存储器且可以通过寄存器名称访问，所以基址是 n/a（不适用）且没有偏移。

图 2-3. Cortex-M4F 寄存器组

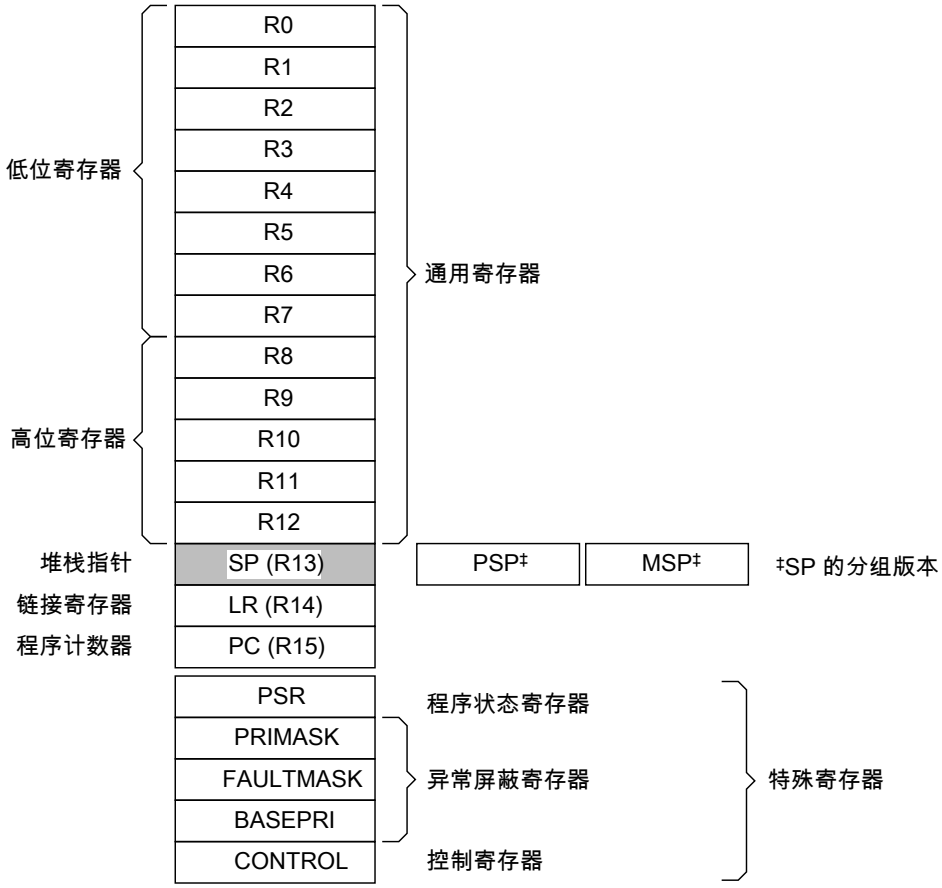


表 2-2. 处理器 寄存器映射

偏移量	名称	类型	复位	描述	见页面
-	R0	R/W	-	Cortex 通用寄存器 0	60
-	R1	R/W	-	Cortex 通用寄存器 1	60
-	R2	R/W	-	Cortex 通用寄存器 2	60
-	R3	R/W	-	Cortex 通用寄存器 3	60
-	R4	R/W	-	Cortex 通用寄存器 4	60
-	R5	R/W	-	Cortex 通用寄存器 5	60
-	R6	R/W	-	Cortex 通用寄存器 6	60
-	R7	R/W	-	Cortex 通用寄存器 7	60
-	R8	R/W	-	Cortex 通用寄存器 8	60
-	R9	R/W	-	Cortex 通用寄存器 9	60
-	R10	R/W	-	Cortex 通用寄存器 10	60
-	R11	R/W	-	Cortex 通用寄存器 11	60

表 2-2. 处理器 寄存器映射（续）

偏移量	名称	类型	复位	描述	见页面
-	R12	R/W	-	Cortex 通用寄存器 12	60
-	SP	R/W	-	堆栈指针	61
-	LR	R/W	0xFFFF.FFFF	链接寄存器	62
-	PC	R/W	-	程序计数器	63
-	PSR	R/W	0x0100.0000	程序状态寄存器	64
-	PRIMASK	R/W	0x0000.0000	优先级屏蔽寄存器	68
-	FAULTMASK	R/W	0x0000.0000	故障屏蔽寄存器	69
-	BASEPRI	R/W	0x0000.0000	基本优先级屏蔽寄存器	70
-	CONTROL	R/W	0x0000.0000	控制寄存器	71
-	FPSC	R/W	-	浮点状态控制	73

2.3.4 寄存器描述

本章列出（以 图2-3（58页） 中所示的顺序）并描述了 Cortex-M4F 寄存器。内核寄存器地地址不是存储器映射的，因此只能通过寄存器名称访问，而不能通过使用偏移量的方式访问。

注意: 在寄存器描述栏中，描述了程序在线程模式和处理器模式执行时的类型。调试访问可能不同。

寄存器 **1: Cortex** 通用寄存器 **0**（**R0**）

寄存器 **2: Cortex** 通用寄存器 **1**（**R1**）

寄存器 **3: Cortex** 通用寄存器 **2**（**R2**）

寄存器 **4: Cortex** 通用寄存器 **3**（**R3**）

寄存器 **5: Cortex** 通用寄存器 **4**（**R4**）

寄存器 **6: Cortex** 通用寄存器 **5**（**R5**）

寄存器 **7: Cortex** 通用寄存器 **6**（**R6**）

寄存器 **8: Cortex** 通用寄存器 **7**（**R7**）

寄存器 **9: Cortex** 通用寄存器 **8**（**R8**）

寄存器 **10: Cortex** 通用寄存器 **9**（**R9**）

寄存器 **11: Cortex** 通用寄存器 **10**（**R10**）

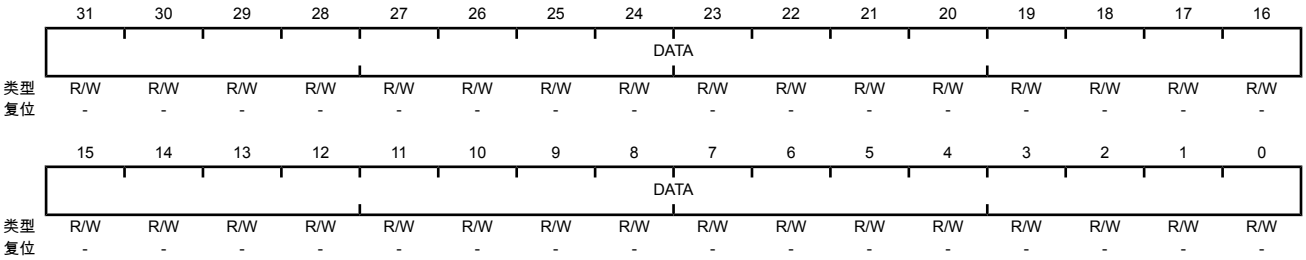
寄存器 **12: Cortex** 通用寄存器 **11**（**R11**）

寄存器 **13: Cortex** 通用寄存器 **12**（**R12**）

Rn 寄存器是供数据操作的 32 位通用寄存器，既可在特权模式下访问，也可在在非特权模式下访问。

Cortex 通用寄存器 0 (R0)

类型 R/W, 复位 -



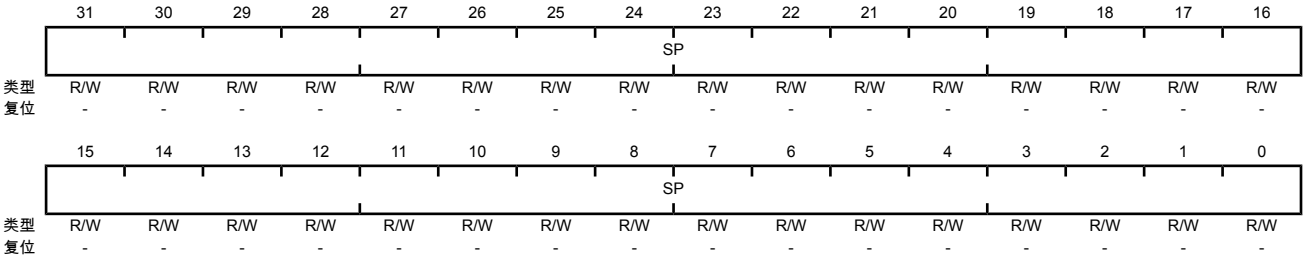
位/域	名称	类型	复位	描述
31:0	DATA	R/W	-	寄存器数据。

寄存器 14: 堆栈指针（SP）

堆栈指针 (SP) 即寄存器 R13。在线程模式下，该寄存器功能取决于控制寄存器 (CONTROL) 寄存器中的 ASP 位。当 ASP 位为 0 时，此寄存器是主堆栈指针 (MSP)。当 ASP 位置位时，此寄存器是进程堆栈指针 (PSP)。复位时，ASP 清零，同时处理器将地址 0x0000.0000 处的值载入 MSP。仅在特权级模式下才可访问 MSP 寄存器；在特权级和非特权级模式都可访问 PSP 寄存器。

堆栈指针 (SP)

类型 R/W, 复位 -



位/域	名称	类型	复位	描述
31:0	SP	R/W	-	该字段是堆栈指针的地址

寄存器 15: 链接寄存器（LR）

R14 寄存器是链接寄存器 (LR)。它存储关于子程序、函数调用和异常事件的返回信息。在特权和非特权模式下均可访问链接寄存器。

遇到异常条目时，EXC_RETURN 被载入到 LR。其值及相关描述参见 表2-10（91页）。

链接寄存器 (LR)

类型 R/W, 复位 0xFFFF.FFFF

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	LINK															
类型	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	LINK															
类型	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

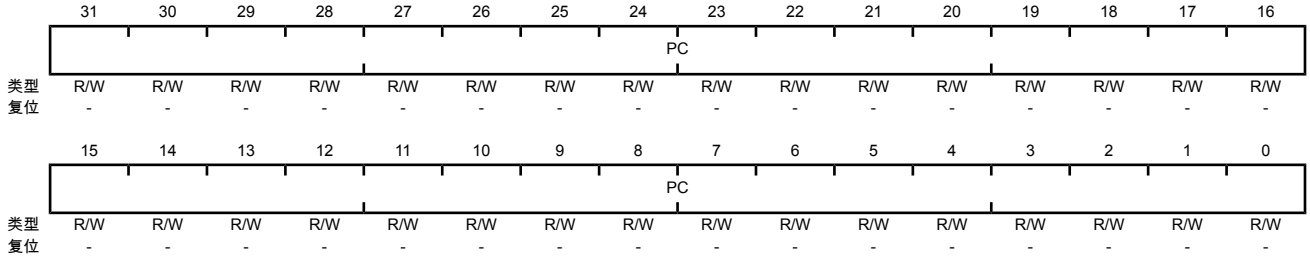
位/域	名称	类型	复位	描述
31:0	LINK	R/W	0xFFFF.FFFF	此字段是返回地址。

寄存器 16: 程序计数器（PC）

R15 寄存器是程序计数器 (PC)，保存的是当前程序的地址。复位时，处理器将地址 0x0000.0004 处的复位向量载入到 PC 寄存器。复位时，该复位向量的位 0 载入到 EPSR 寄存器中的 THUMB 位，此时该位必须是 1。在特权和非特权模式下均可访问 PC 寄存器。

程序计数器 (PC)

类型 R/W, 复位 -



位/域	名称	类型	复位	描述
31:0	PC	R/W	-	此字段是当前程序的地址

寄存器 17: 程序状态寄存器（PSR）

注意: 此寄存器也被称为 xPSR。

程序状态寄存器 (PSR) 有三个功能，且寄存器中的位被分配为不同的功能：

- 应用程序状态寄存器 (APSR)，位域 31:27，位域 19:16
- 运行程序状态寄存器 (EPSR)，位域 26:24、15:10
- 中断程序状态寄存器 (IPSR)，位域 7:0

PSR、IPSR 和 EPSR 寄存器只能在特权模式下访问；APSR 寄存器在特权和非特权模式下均可访问。

APSR 寄存器保存前面指令执行状态标记的当前状态。

EPSR 包含 Thumb 状态位和 If-Then (IT) 指令或中断的多周期载入和存储指令的“可中断-可继续指令”(ICI) 域的执行状态位。尝试通过应用软件使用 MSR 指令直接读取 EPSR 的操作，其返回值总是为零。尝试在应用程序软件中使用 MSR 指令对 EPSR 进行写操作，将始终被忽略。故障处理器可通过检查堆栈式的 PSR 中的 EPSR 值来确定出现故障的操作（见 “异常进入和返回”（89页））。

IPSR 寄存器包含的是当前中断服务程序 (ISR) 的异常类型号。

这些寄存器可以单独访问或者是任何两三个一起访问，访问时使用寄存器的名称作为 MSR 或 MRS 指令的参数。例如，可以使用 PSR + MRS 指令组合对所有寄存器进行读操作；APSR + MSR 指令组合只能对 APSR 寄存器进行写操作。64页 列出了 PSR 所有可能的寄存器组合。请参见 MRS 和 MSR 指令的描述（在 “ARM® Cortex™-M4 Devices Generic User Guide [文献编号 ARM DUI 0553A]” 的 Cortex™-M4 指令集章节中），了解关于如何访问程序状态寄存器的更多信息。

表 **2-3. PSR** 寄存器组合

寄存器	类型	组合
PSR	R/W[a, b]	APSR、EPSR 和 IPSR
IEPSR	RO	EPSR 和 IPSR
IAPSR	R/W[a]	APSR 和 IPSR
EAPSR	R/W[b]	APSR 和 EPSR

a. 处理器忽略对 IPSR 位的写操作。

b. 读 EPSR 位返回零，处理器忽略这些位的写操作。

程序状态寄存器 (PSR)

类型 R/W, 复位 0x0100.0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
N	Z	C	V	Q	ICI / IT		THUMB	保留				GE			
类型 R/W	R/W	R/W	R/W	R/W	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W
复位 0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ICI / IT						保留		ISRNUM							
类型 RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位 0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位/域	名称	类型	复位	描述
31	N	R/W	0	APSR 负数或小于标志 值 描述 1 先前操作的结果是负数或小于。 0 先前操作的结果是整数、零、大于或相等。 当访问 PSR 或 APSR 时该位才有意义。
30	Z	R/W	0	APSR 零标志 值 描述 1 先前操作结果为0。 0 先前操作结果非0。 当访问 PSR 或 APSR 时该位才有意义。
29	C	R/W	0	APSR 进位或借位标志 值 描述 1 先前加法操作导致进位或者先前的减法操作没有产生借位。 0 先前的加法操作没有导致进位或者先前的减法操作导致了借位。 当访问 PSR 或 APSR 时该位才有意义。
28	V	R/W	0	APSR 上溢标志 值 描述 1 先前的操作导致了上溢。 0 先前的操作没导致上溢。 当访问 PSR 或 APSR 时该位才有意义。
27	Q	R/W	0	APSR DSP 上溢和饱和标志 值 描述 1 使用 SIMD 指令时，发生 DSP 上溢或饱和。 0 自从复位或自从该位上次清零以来没有发生过 DSP 上溢或饱和。 当访问 PSR 或 APSR 时该位才有意义。 该位可由软件使用 MRS 指令清零。
26:25	ICI / IT	RO	0x0	EPSR ICI/IT 状态 这些位以及位 15:10 包含中断的多指令载入操作或多指令存储操作的“可中断-可继续指令”(ICI) 域或者 IT 指令的执行状态位。 当 EPSR 保持 ICI 执行状态时，位 26:25 都是 0。 If-Then 模块在 IT 指令之后最多包含四条指令。该模块中的每条指令都是带有条件的。这些指令的条件有可能都一样，其中一些也可能和其它相反。更多信息请参考 “ARM® Cortex™-M4 Devices Generic User Guide（文献编号 ARM DUI 0553A）”中的 Cortex™-M4 指令集章节。 当访问 PSR 或 EPSR 时该位才有意义。注意这些 EPSR 位不能使用 MRS 和 MSR 指令访问，但提供了定义，以便在异常处理程序中解译堆栈式 (E)PSR 值。

位/域	名称	类型	复位	描述
24	THUMB	RO	1	EPSR Thumb 状态 该位指示Thumb的状态且应该一直为1。 下面的方式可清除 THUMB 位： ■ BLX、BX 和 POP{PC} 指令 ■ 异常返回时的堆栈式 xPSR 值恢复 ■ 异常进入或复位时的向量值的位 0 尝试在该位为0时执行指令或导致产生fault后者锁住。更多信息参见 “死锁”（93页）。 当访问 PSR 或 EPSR 时该位才有意义。
23:20	保留	RO	0x00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
19:16	GE	R/W	0x0	大于或等于标志 更多信息请参考 “ARM® Cortex™-M4 Devices Generic User Guide（文献编号 ARM DUI 0553A）”中 Cortex™-M4 指令集章节的 ARM DUI 0553A 指令描述。 当访问 PSR 或 APSR 时该位域才有意义。
15:10	ICI / IT	RO	0x0	EPSR ICI/IT 状态 这些位以及位 26:25 都包含可用于中断的多指令载入操作或多指令存储操作的“可中断-可继续指令”(ICI) 域以及 IT 指令的执行状态位。 在执行 LDM、STM、PUSHPOP、VLDM、VSTM、VPUSH 或 VPOP 指令过程中，如果出现中断，处理器将暂时停止多指令载入或多指令存储操作，并将多指令操作中的下一个寄存器操作数存储到位 15:12 中。处理完中断后，处理器返回到位 15:12 指向的寄存器，然后恢复多指令载入和存储操作。当 EPSR 保持 ICI 执行状态时，位 11:10 都是 0。 If-Then 模块在 16 位的 IT 指令之后最多包含四条指令。该模块中的每条指令都是带有条件的。这些指令的条件有可能都一样，其中一些也可能和其它相反。更多信息请参考 “ARM® Cortex™-M4 Devices Generic User Guide（文献编号 ARM DUI 0553A）”中的 Cortex™-M4 指令集章节。 当访问 PSR 或 EPSR 时该位才有意义。
9:8	保留	RO	0x0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。

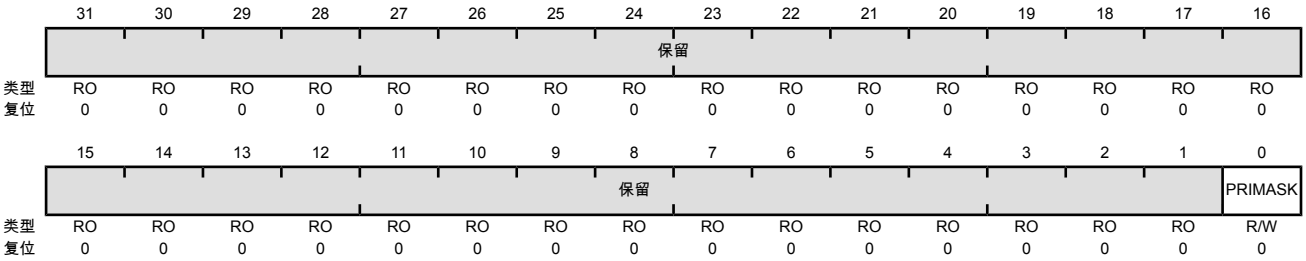
位/域	名称	类型	复位	描述
7:0	ISRNUM	RO	0x00	IPSR ISR 号 该域包含的是当前中断服务程序 (ISR) 的异常类型号。 值 描述 0x00 线程模式 0x01 保留 0x02 NMI 0x03 硬故障 0x04 存储器管理故障 0x05 总线故障 0x06 用法故障 0x07-0x0A 保留 0x0B SVCall 0x0C 保留用于调试 0x0D 保留 0x0E PendSV 0x0F SysTick 0x10 中断向量0 0x11 中断向量1 ... ... 0x9A 中断向量138 更多信息参见 “异常类型”（84页）。 当访问 PSR 或 IPSR 时该字段才有意义。

寄存器 18: 优先级屏蔽寄存器（PRIMASK）

PRIMASK 寄存器可屏蔽所有优先级可编程的异常。只有固定优先级的复位、NMI 和硬故障是例外的。当异常可能影响到关键任务执行时间时应该被禁止。该寄存器只能在特权模式下访问。MSR 和 MRS 指令用于访问 PRIMASK 寄存器，CPS 指令可用于更改 PRIMASK 寄存器的值。关于这些指令的更多信息请参考 “ARM® Cortex™-M4 Devices Generic User Guide（文献编号 ARM DUI 0553A）”中的 Cortex™-M4 指令集章节。有关异常优先级的更多信息，请参考“异常类型”（84页）。

优先级屏蔽寄存器 (PRIMASK)

类型 R/W, 复位 0x0000.0000



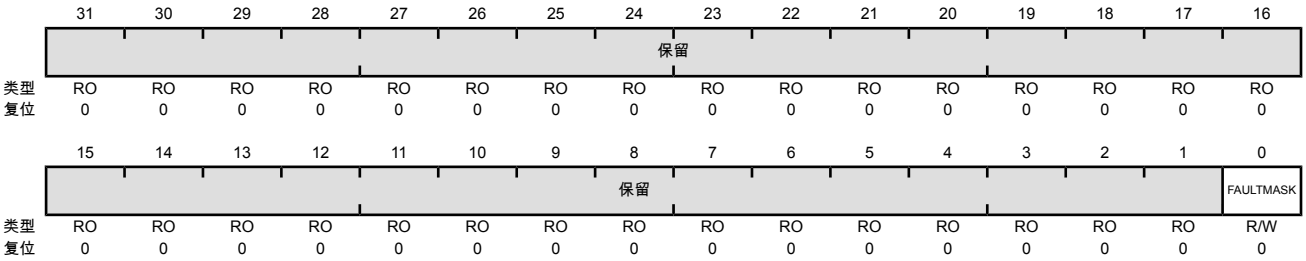
位/域	名称	类型	复位	描述
31:1	保留	RO	0x0000.000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
0	PRIMASK	R/W	0	优先级屏蔽 值 描述 1 禁止所有可配置优先级的异常。 0 没有影响

寄存器 19: 故障屏蔽寄存器（FAULTMASK）

FAULTMASK 寄存器可屏蔽除 NMI（不可屏蔽中断）外的所有异常。当异常可能影响到关键任务执行时间时应该被禁止。该寄存器只能在特权模式下访问。MSR 和 MRS 指令用于访问 FAULTMASK 寄存器，CPS 指令可用于更改 FAULTMASK 寄存器的值。关于这些指令的更多信息请参考“ARM® Cortex™-M4 Devices Generic User Guide（文献编号 ARM DUI 0553A）”中的 Cortex™-M4 指令集章节。有关异常优先级的更多信息，请参考“异常类型”（84页）。

故障屏蔽寄存器 (FAULTMASK)

类型 R/W, 复位 0x0000.0000



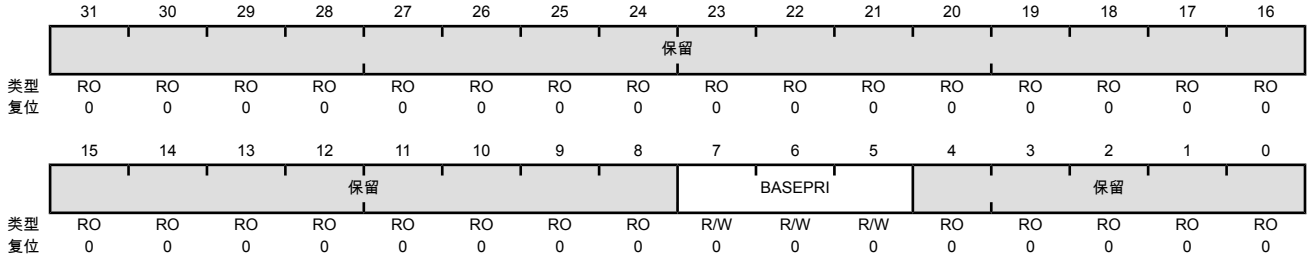
位/域	名称	类型	复位	描述
31:1	保留	RO	0x0000.000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
0	FAULTMASK	R/W	0	故障屏蔽 值 描述 1 禁止除了NMI的所有异常。 0 没有影响 处理器从除了 NMI 外的任何异常处理程序返回时清零 FAULTMASK 位。

寄存器 20: 基本优先级屏蔽寄存器（BASEPRI）

BASEPRI 寄存器定义了异常处理的最小优先级。当 BASEPRI 寄存器是非零值时，它将会禁止所有异常优先级比 BASEPRI 寄存器低或相等的异常。当异常可能影响到关键任务执行时间时应该被禁止。该寄存器只能在特权模式下访问。有关异常优先级的更多信息，请参考 “异常类型”（84页）。

基本优先级屏蔽寄存器 (BASEPRI)

类型 R/W, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:8	保留	RO	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
7:5	BASEPRI	R/W	0x0	基本优先级 任何可编程优先级的异常的优先级低于或等于该字段的值时将被屏蔽。PRIMASK 寄存器可用来屏蔽所有优先级可编程的异常。优先级越高，优先级别越低。 值 描述 0x0 所有异常都不屏蔽 0x1 所有优先级别在1-7的异常都将屏蔽。 0x2 所有优先级别在2-7的异常都将屏蔽。 0x3 所有优先级别在3-7的异常都将屏蔽。 0x4 所有优先级别在4-7的异常都将屏蔽。 0x5 所有优先级别在5-7的异常都将屏蔽。 0x6 所有优先级别在6-7的异常都将屏蔽。 0x7 所有优先级别为7的异常都将屏蔽。
4:0	保留	RO	0x0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。

寄存器 21: 控制寄存器（CONTROL）

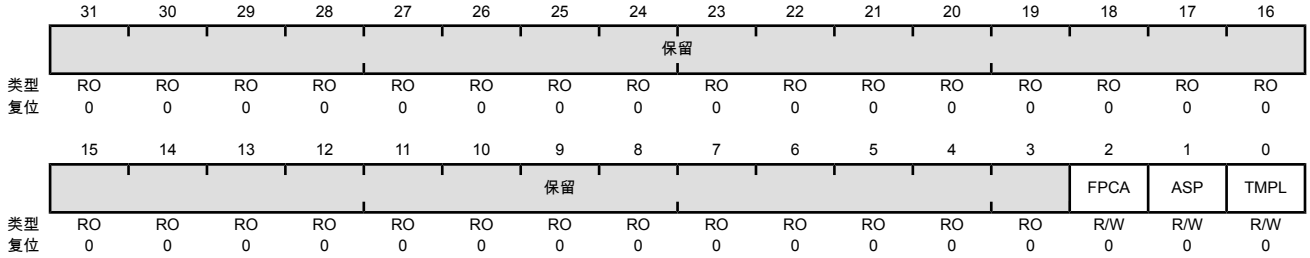
当处理器处于线程模式时，CONTROL 寄存器控制使用的堆栈和软件执行的特权等级，并指示 FPU 是否处于活动状态。该寄存器只能在特权模式下访问。

处理程序模式总是使用 MSP，处理器将忽略在处理模式下向 CONTROL 寄存器的 ASP 位写入具体的值。异常进入和返回机制自动采用 EXC_RETURN（参见 表2-10（91页））的值来更新 CONTROL 寄存器。在带操作系统环境下，线程在线程模式下运行应该使用process堆栈，而内核和异常处理应该使用main中断。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到 PSP，可按照 “ARM® Cortex™-M4 Devices Generic User Guide（文献编号 ARM DUI 0553A）”中 Cortex™-M4 指令集章节的详细介绍，使用 MSR 指令将 ASP 位置位，或使用相应的 EXC_RETURN 值在发生异常时返回线程模式，如表2-10（91页）所示。

注意: 当改变堆栈指针时，软件必须在执行 ISB 指令后立即使用一条 MSR 指令，确保 ISB 后的指令使用新的堆栈指针。请参见 “ARM® Cortex™-M4 Devices Generic User Guide（文献编号 ARM DUI 0553A）”中的 Cortex™-M4 指令集章节。

控制寄存器 (CONTROL)

类型 R/W, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:3	保留	RO	0x0000.000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
2	FPCA	R/W	0	激活浮点上下文 值 描述 1 激活浮点上下文 0 没有激活浮点上下文 Cortex-M4F 使用该位来确定在处理异常时是否保留浮点状态。 **重要:** 两个位控制何时启用 FPCA：浮点上下文控制 (FPCC) 寄存器中的 ASPEN 位和辅助控制 (ACTLR) 寄存器中的 DISFPCA 位。
1	ASP	R/W	0	活动堆栈指针 值 描述 1 PSP 是当前堆栈指针。 0 MSP 是当前堆栈指针。 在处理模式下，该位读出零写会忽略。当异常返回时，Cortex-M4F 自动更新该位。

位/域	名称	类型	复位	描述
0	TMPL	R/W	0	线程模式特权级 值 描述 1 非特权级软件可以在线程模式被执行。 0 只有特权级的软件才能在线程模式下被执行。

寄存器 22: 浮点状态控制（FPSC）寄存器

FPSC 寄存器提供浮点系统所有必要的用户级控制。

浮点状态控制 (FPSC)

类型 R/W, 复位 -

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
域	N	Z	C	V	保留	AHP	DN	FZ	RMODE		保留					
类型	R/W	R/W	R/W	R/W	RO	R/W	R/W	R/W	R/W	R/W	RO	RO	RO	RO	RO	RO
复位	-	-	-	-	0	-	-	-	-	-	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
域	保留								IDC	保留		IXC	UFC	OFC	DZC	IOC
类型	RO	RO	RO	RO	RO	RO	RO	RO	R/W	RO	RO	R/W	R/W	R/W	R/W	R/W
复位	0	0	0	0	0	0	0	0	-	0	0	-	-	-	-	-

位/域	名称	类型	复位	描述
31	N	R/W	-	负极状态码标志 浮点比较操作会更新此状态码标志。
30	Z	R/W	-	零状态码标志 浮点比较操作会更新此状态码标志。
29	C	R/W	-	进位状态码标志 浮点比较操作会更新此状态码标志。
28	V	R/W	-	上溢状态码标志 浮点比较操作会更新此状态码标志。
27	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
26	AHP	R/W	-	Alternative 半精度 置位时，选择 Alternative 半精度格式。清零时，选择 IEEE 半精度格式。 FPDSC 寄存器中的 AHP 位包含该位的默认值。
25	DN	R/W	-	默认 NaN 模式 置位时，任何涉及一个或多个 NaN 的操作都会返回默认 NaN。清零时，NaN 操作数传递到浮点操作的输出。 FPDSC 寄存器中的 DN 位包含该位的默认值。
24	FZ	R/W	-	清零模式 置位时，清零模式启用。清零时，清零模式被禁用，浮点系统的行为完全符合 IEEE 754 标准。 FPDSC 寄存器中的 FZ 位包含该位的默认值。

位/域	名称	类型	复位	描述
23:22	RMODE	R/W	-	舍入模式 几乎所有的浮点指令都使用指定的舍入模式。 FPDSC 寄存器中的 RMODE 位包含该位的默认值。 值 描述 0x0 向最接近值舍入 (RN) 模式 0x1 向正无穷大舍入 (RP) 模式 0x2 向负无穷大舍入 (RM) 模式 0x3 向 0 舍入 (RZ) 模式
21:8	保留	RO	0x0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
7	IDC	R/W	-	输入反常累积异常 置位时，表示该异常自上一次向该位写 0 已经发生。
6:5	保留	RO	0x0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
4	IXC	R/W	-	不精确的累积异常 置位时，表示该异常自上一次向该位写 0 已经发生。
3	UFC	R/W	-	下溢累积异常 置位时，表示该异常自上一次向该位写 0 已经发生。
2	OFC	R/W	-	上溢累积异常 置位时，表示该异常自上一次向该位写 0 已经发生。
1	DZC	R/W	-	除零累积异常 置位时，表示该异常自上一次向该位写 0 已经发生。
0	IOC	R/W	-	无效操作累积异常 置位时，表示该异常自上一次向该位写 0 已经发生。

2.3.5 异常和中断

Cortex-M4F处理器支持中断和系统异常。处理器和嵌入向量中断控制器分级和处理所有异常。异常改变了软件控制流。处理器使用处理模式处理除了复位之外的所有异常。更多信息参见“异常进入和返回”（89页）。

NVIC寄存器控制中断处理。更多信息参见 “嵌套式向量化中断控制器（NVIC）”（102页）。

2.3.6 数据类型

Cortex-M4F 支持 32 位字、16 位半字和 8 位字节。处理器也支持 64 位数据传输指令。所有指令和数据访问都是小端模式。更多信息参见 “内存区，类型和属性”（77页）。

2.4 存储模型

本节描述处理器存储映射、存储器访问和位带特征。处理器提供4GB的寻址空间和混合的存储映射。

TM4C1231H6PM 控制器的存储器映射请参考表2-4（75页）。在本手册中，寄存器地址将采用相对于存储器映射表中各模块基址的十六进制递增的形式给出。

在SRAM和外设区包含位带区。位带区对位数据提供原子操作，参见 “位带区”（79页）。

处理器为核心外设寄存器保留了相应范围的专用外设总线 (PPB) 地址区（参见 “Cortex-M4 外设”（101页））。

注意: 在存储器映射中，尝试在保留空间读、写地址会导致总线故障。此外，尝试在 Flash 范围写入地址也会导致总线故障。

表 **2-4.** 存储器映射

开始	结束	描述	详见页
存储器			
0x0000.0000	0x0003.FFFF	片上Flash	472
0x0004.0000	0x1FFF.FFFF	保留	-
0x2000.0000	0x2000.7FFF	片上 SRAM 位带	459
0x2000.8000	0x21FF.FFFF	保留	-
0x2200.0000	0x220F.FFFF	片上 SRAM 的位带别名区，起始地址为 0x2000.0000	459
0x2210.0000	0x3FFF.FFFF	保留	-
外设			
0x4000.0000	0x4000.0FFF	看门狗定时器0	695
0x4000.1000	0x4000.1FFF	看门狗定时器1	695
0x4000.2000	0x4000.3FFF	保留	-
0x4000.4000	0x4000.4FFF	GPIO 端口 A	585
0x4000.5000	0x4000.5FFF	GPIO 端口 B	585
0x4000.6000	0x4000.6FFF	GPIO 端口 C	585
0x4000.7000	0x4000.7FFF	GPIO 端口 D	585
0x4000.8000	0x4000.8FFF	SSI0	870
0x4000.9000	0x4000.9FFF	SSI1	870
0x4000.A000	0x4000.AFFF	SSI2	870
0x4000.B000	0x4000.BFFF	SSI3	870
0x4000.C000	0x4000.CFFF	UART0	810
0x4000.D000	0x4000.DFFF	UART1	810

表 2-4. 存储器映射（续）

开始	结束	描述	详见页
0x4000.E000	0x4000.EFFF	UART2	810
0x4000.F000	0x4000.FFFF	UART3	810
0x4001.0000	0x4001.0FFF	UART4	810
0x4001.1000	0x4001.1FFF	UART5	810
0x4001.2000	0x4001.2FFF	UART6	810
0x4001.3000	0x4001.3FFF	UART7	810
0x4001.4000	0x4001.FFFF	保留	-
外设			
0x4002.0000	0x4002.0FFF	I^2C 0	918
0x4002.1000	0x4002.1FFF	I^2C 1	918
0x4002.2000	0x4002.2FFF	I^2C 2	918
0x4002.3000	0x4002.3FFF	I^2C 3	918
0x4002.4000	0x4002.4FFF	GPIO 端口 E	585
0x4002.5000	0x4002.5FFF	GPIO 端口 F	585
0x4002.6000	0x4002.FFFF	保留	-
0x4003.0000	0x4003.0FFF	16/32 位定时器 0	649
0x4003.1000	0x4003.1FFF	16/32 位定时器 1	649
0x4003.2000	0x4003.2FFF	16/32 位定时器 2	649
0x4003.3000	0x4003.3FFF	16/32 位定时器 3	649
0x4003.4000	0x4003.4FFF	16/32 位定时器 4	649
0x4003.5000	0x4003.5FFF	16/32 位定时器 5	649
0x4003.6000	0x4003.6FFF	32/64 位定时器 0	649
0x4003.7000	0x4003.7FFF	32/64 位定时器 1	649
0x4003.8000	0x4003.8FFF	ADC0	736
0x4003.9000	0x4003.9FFF	ADC1	736
0x4003.A000	0x4003.BFFF	保留	-
0x4003.C000	0x4003.CFFF	模拟比较器触发	993
0x4003.D000	0x4003.FFFF	保留	-
0x4004.0000	0x4004.0FFF	CAN0 控制器	965
0x4004.1000	0x4004.BFFF	保留	-
0x4004.C000	0x4004.CFFF	32/64 位定时器 2	649
0x4004.D000	0x4004.DFFF	32/64 位定时器 3	649
0x4004.E000	0x4004.EFFF	32/64 位定时器 4	649
0x4004.F000	0x4004.FFFF	32/64 位定时器 5	649
0x4005.0000	0x4005.7FFF	保留	-
0x4005.8000	0x4005.8FFF	GPIO 端口 A（AHB 槽）	585
0x4005.9000	0x4005.9FFF	GPIO 端口 B（AHB 槽）	585
0x4005.A000	0x4005.AFFF	GPIO 端口 C（AHB 槽）	585
0x4005.B000	0x4005.BFFF	GPIO 端口 D（AHB 槽）	585
0x4005.C000	0x4005.CFFF	GPIO 端口 E（AHB 槽）	585
0x4005.D000	0x4005.DFFF	GPIO 端口 F（AHB 槽）	585
0x4005.E000	0x400A.EFFF	保留	-

表 2-4. 存储器映射（续）

开始	结束	描述	详见页
0x400A.F000	0x400A.FFFF	EEPROM 和密钥锁	488
0x400B.0000	0x400F.8FFF	保留	-
0x400F.9000	0x400F.9FFF	系统异常模块	421
0x400F.A000	0x400F.BFFF	保留	-
0x400F.C000	0x400F.CFFF	睡眠模块	440
0x400F.D000	0x400F.DFFF	Flash 存储器控制	472
0x400F.E000	0x400F.EFFF	系统控制	207
0x400F.F000	0x400F.FFFF	µDMA	532
0x4010.0000	0x41FF.FFFF	保留	-
0x4200.0000	0x43FF.FFFF	0x4000.0000 到 0x400F.FFFF 的位带别名	-
0x4400.0000	0xDFFF.FFFF	保留	-
专用的外设总线			
0xE000.0000	0xE000.0FFF	仪表跟踪宏单元（ITM）	55
0xE000.1000	0xE000.1FFF	数据观察点和跟踪（DWT）	55
0xE000.2000	0xE000.2FFF	Flash 修补和断点（FPB）	55
0xE000.3000	0xE000.DFFF	保留	-
0xE000.E000	0xE000.EFFF	Cortex-M4F 外设（SysTick、NVIC、MPU、FPU 和 SCB）	111
0xE000.F000	0xE003.FFFF	保留	-
0xE004.0000	0xE004.0FFF	跟踪端口的接口单元（TPIU）	55
0xE004.1000	0xE004.1FFF	嵌入式跟踪宏单元（ETM）	55
0xE004.2000	0xFFFF.FFFF	保留	-

2.4.1 内存区，类型和属性

存储器映射和 MPU 的编程将存储器映射分割成几个区域。每个区被定义了存储类型且有些区有附件的内存属性。存储类型和属性决定访问该区的行为。

存储类型：

- 普通:处理器为了效率可重新排序和不确定的读操作。
- 设备：处理器保存传送顺序并严格依照顺序和其它设备或严格排序存储器交换信息。
- 严格排序处理器保存传送顺序并严格依照顺序和其它设备交换信息。

设备和严格排序存储器的顺序要求不同，这就意味着，存储系统可以将写操作缓冲到设备存储器，而不可缓冲到严格排序存储器。

附件的存储属性是永不执行区 (XN)。意味着处理器阻止指令访问。只要在 XN 区执行指令就会产生故障异常。

2.4.2 内存访问存储系统顺序

大多数的内存访问时通过具体的内存访问指令，存储系统并不能保证访问的顺序和指令的编程顺序一致，提供的顺序不影响指令序列的行为。通常，如果程序的正常执行依赖于两次存储器访问依照编程顺序，则软件要在两次存储器访问指令之间设置存储器阻碍指令。（参见“存储器访问的软件顺序”（78页））。

然而，内存系统保证设备和严格顺序存储之间的访问顺序。两条内存访问指A1和A 2，如果A1和A2都访问设备或者严格顺序存储器，并且在编程顺序上A1在A2前边，则A1将一直在A2前边被获取。

2.4.3 存储器访问行为

表2-5（78页） 示出存储器映射中每个区的访问行为。关于存储器类型和 XN 属性的具体信息，参见 “内存区，类型和属性”（77页）。 Tiva™ C 系列 设备保留如下表所示的地址范围。更多信息，参见 表2-4（75页）。

表 **2-5.** 存储器访问行为

地址范围	存储器区域	存储器类型	从不执行 **(XN)**	描述
0x0000.0000 - 0x1FFF.FFFF	代码	正常模式	-	这个可执行区域用于存放程序代码。数据也可以保存到这里。
0x2000.0000 - 0x3FFF.FFFF	SRAM	正常模式	-	这个可执行区域用于存放数据。代码也可以保存在这里。这个区域包括了位带和位带别名区（参见 表2-6（79页））。
0x4000.0000 - 0x5FFF.FFFF	外设	设备	XN	这个区域包括了位带和位带别名区（参见 表2-7（80页））。
0x6000.0000 - 0x9FFF.FFFF	外部 RAM	正常模式	-	这个可执行区域用于存放数据。
0xA000.0000 - 0xDFFF.FFFF	外部设备	设备	XN	这个区域用作外部器件存储器。
0xE000.0000- 0xE00F.FFFF	专用外设总线	严格排序	XN	这个区域包括 NVIC、系统定时器和系统控制模块。
0xE010.0000- 0xFFFF.FFFF	保留	-	-	-

CODE、SRAM 和外部 RAM 可以保存程序。然而，推荐在 CODE 区保存程序，因为 Cortex-M4F 分离总线可以同时读取和访问数据。

MPU 可以不理会默认存储器访问。详见 “存储器保护单元 (MPU)”（104页）。

Cortex-M4F 会在执行前预取指令，并且它会以推测的方式从分支目标地址进行预取。

2.4.4 存储器访问的软件顺序

程序流中的指令顺序并不总能保证相应的内存传送顺序，有以下几个原因：

- 为提高效率，处理器能够记录一些存储器访问，提供这些并不影响指令序列的行为。
- 处理器有多个总线接口。
- 存储器映射中的存储器或设备有不同的等待状态。
- 有些存储器访问被缓冲或者是不确定的。

“内存访问存储系统顺序”（77页） 描述了存储系统保证存储器访问顺序的几种情况。然而，如果内存访问顺序是关键的，软件必须包含内存边界指令强制顺序。Cortex-M4F 有如下的内存边界指令：

- 数据存储器边界 (DMB) 指令确保先执行尚未完成的存储器传输，再执行后续存储器传输。
- 数据同步边界 (DSB) 指令确保先执行尚未完成的存储器传输，再执行后续指令。
- 指令同步边界 (ISB) 指令确保所有已完成的存储器传输的影响能被后续指令识别。

内存边界指令可被用在下面的情况：

- MPU 编程
  - 如果 MPU 设置改变，且改变必须在下条指令时有效，应使用 DSB 指令以确保 MPU 在指令切换结束后立即生效。
  - 如果使用分支或者调用进入MPU配置代码，应使用ISB指令确保新的MPU设置在编程MPU区后立即生效。如果 MPU 配置代码使用异常机制进入，则不需要 ISB 指令。
- 向量表

  如果程序改变了向量表的入口并启用了相应的异常，应在两次操作之间使用 DMB 指令。当异常在启用后立即发生时，DMB 指令可确保处理器使用新的异常向量。
- 自修改代码

  如果程序包含自修改代码，应在程序代码修改后立即使用 ISB 指令。ISB 指令确保在执行后续指令时使用更新的程序。
- 存储器映射切换

  如果系统包含存储器映射切换机制，应在程序中切换存储器映射之后使用 DSB 指令。DSB 指令确保在执行后续指令时使用更新的存储器映射。
- 动态异常优先级改变

  当异常处于挂起或者活动状态时，如果其优先级需要更改，应在更改之后使用 DSB 指令。在 DSB 指令执行之后，更改将生效。

访问严格顺序存储器时（例如系统控制块），不需要使用 DMB 指令。

关于存储器边界指令的更多细节，请参考“ ARM® Cortex™-M4 Devices Generic User Guide（文献编号 ARM DUI 0553A）”中的 Cortex™-M4 指令集章节。

2.4.5 位带区

位带区映在位带别名区中的每个字到位带区中的单个位。位带区占用SRAM和外设内存区中最少1MB空间。对 32-MBSRAM 别名区的访问映射到 SRAM 中 1-MB 的位带区，如 表2-6（79页）所示。对 32-MB 外设别名区的访问映射到 1-MB 的外设位带区，如 表2-7（80页）所示。关于位带区的特定地址范围，参见 表2-4（75页）。

注意: 对在SRAM或外设位带别名区中一个字的访问映射到SRAM或外设位带区中的一个位。

对位带区的一个字访问结果是对相应内存的一个字访问，类似半字或字节访问。这样就与相应外设的访问要求相符合。

表 **2-6. SRAM** 存储器位带区

地址范围		存储器区域	指令和数据访问
开始	结束		
0x2000.0000	0x2000.7FFF	SRAM 位带区	对这个存储器范围的直接访问行为如同对SRAM存储器的访问。但是该区域也可通过位带别名进行位寻址。
0x2200.0000	0x220F.FFFF	SRAM 位带别名	对这个区域的数据访问被重新映射到位带区。一个写操作被执行为读-修改-写。指令访问没有重新映射。

表 2-7. 外设存储器位带区

地址范围		存储器区域	指令和数据访问
开始	结束		
0x4000.0000	0x400F.FFFF	外设位带区	对这个存储器范围的直接访问行为如同对外设存储器的访问。但是该区域也可通过位带别名进行位寻址。
0x4200.0000	0x43FF.FFFF	外设位带别名	对这个区域的数据访问被重新映射到位带区。一个写操作被执行为读-修改-写。指令访问没有重新映射。

下面的公式演示了别名区和位带区的映射关系：

```
bit_word_offset = (byte_offset x 32) + (bit_number x 4)
```

```
bit_word_addr = bit_band_base + bit_word_offset
```

此处，

bit_word_offset
在位带区中的目标位的位置。

bit_word_addr
在别名区中雨目标位映射的字的地址。

bit_band_base
别名区的起始地址。

byte_offset
字节在包含目标位的位带区中的编号。

bit_number
目标位的位置，0-7。

图2-4（81页）示出了 SRAM 位带别名区和 SRAM 位带区映射的例子：

- 在 0x23FF.FFE0 处的别名字映射到了位带区中的 0x200F.FFFF 的第 0 位：

  ```
  0x23FF.FFE0 = 0x2200.0000 + (0x000F.FFFF*32) + (0*4)
  ```

- 在 0x23FF.FFFC 处的别名字映射到了位带区中的 0x200F.FFFF 的第 7 位：

  ```
  0x23FF.FFFC = 0x2200.0000 + (0x000F.FFFF*32) + (7*4)
  ```

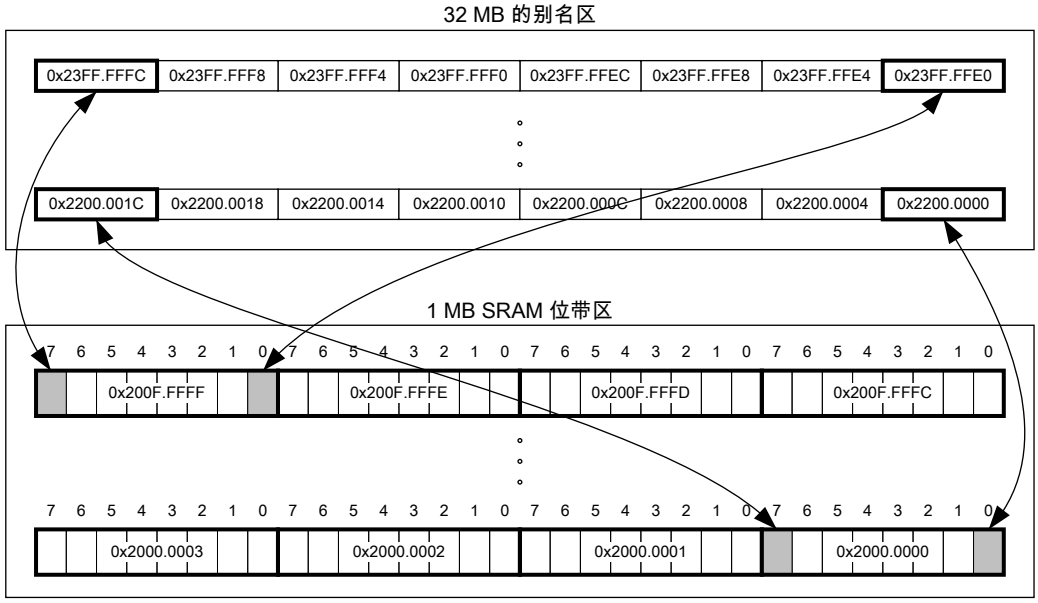
- 在 0x2200.0000 处的别名字映射到了位带区中的 0x2000.0000 的第 0 位：

  ```
  0x2200.0000 = 0x2200.0000 + (0*32) + (0*4)
  ```

- 在 0x2200.001C 处的别名字映射到了位带区中的 0x2000.0000 的第 7 位：

  ```
  0x2200.001C = 0x2200.0000+ (0*32) + (7*4)
  ```

图 **2-4.** 位带映射



2.4.5.1 直接访问别名区

在别名区写一个字会更新位带区的一个位。

在别名区写入的字，其值的第0位决定了位带区对应位的值。写入值的第0位置位使位带位为1，写入值的第0位清零使位带位为0。

别名区字的位 31:1 对位带区对应位没有影响。写入0x01和写入0xFF的影响是一样的。写入0x00和写入0x0E的影响是一样的。

当在别名区写入一个字时，0x0000.0000 表示位带区中的对应位清零，0x0000.0001 表示位带区中的对应位置位。

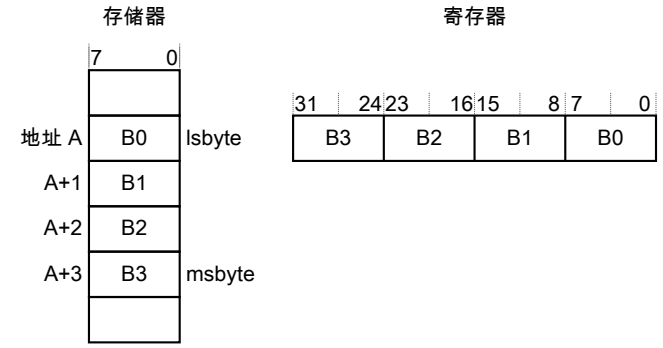
2.4.5.2 直接访问位带区

“存储器访问行为”（78页） 描述了对位带区的字节、半字或字进行直接访问的行为。

2.4.6 数据保存

处理器将存储器看做一个从零开始按增加顺序编号的线性字节集合。例如，字节0-3占据第一个存储字，字节4-7占据第二个存储字。数据以小端法 (little-endian) 格式存储，字的最不重要字节 (lsbyte) 存储在最低编号字节，最重要字节 (msbyte) 存储在最高编号字节。图2-5（82页）示出了数据是如何存储的。

图 **2-5.** 数据保存



2.4.7 同步原语

Cortex-M4F 指令集包含多对同步原语，可以提供一个没有阻碍的机制，使一个线程或进程能用于获取对存储位置的专有访问。软件可以使用这些原语执行有保证的读-修改-写存储器更新序列，或者用作信号量机制。

一对同步原语包括：

- 一个下载专用指令，用于读取一个存储器位置的值并请求对该位置的专有访问。
- 一个保存专用指令，用于尝试写入相同的存储器位置并想一个寄存器返回一个状态位。如果这个状态位为 0，表示该线程或进程获得了对存储器的专有访问，写入成功；如果这个状态位为 1，表示该线程或进程没有获得对存储器的专有访问，写入没有执行。

这对下载专用和保存专用指令是：

- 字指令 LDREX 和 STREX
- 半字指令 LDREXH 和 STREXH
- 字节指令 LDREXB 和 STREXB

软件必须使用对应的下载专用指令和保存专用指令。

要对存储器位置执行专有的读-修改-写操作，软件必须：

1. 使用一个下载专用指令读取该位置的值。
2. 按照需要修改该值。
3. 使用保存专用指令来尝试将新值写入存储器位置。
4. 测试返回的状态位。

   如果状态位清零，读-修改-写操作成功完成。如果状态位为 0，不执行写操作，表示第 1 步返回的值可能已过时。软件必须重试整个读-修改-写序列。

软件可以使用同步原语来实现信号量，如下：

1. 使用一个下载专用指令读取信号量地址，检测信号量是否自由。
2. 如果信号量是自由的，使用保存专用指令向信号量地址写入获取值。

3. 如果第2步返回的状态位表示保存成功，那么软件已经获取了信号量。但是，如果保存失败，那么可能在软件执行第1步后另外的进程已经获取了信号量。

Cortex-M4F 包含一个专用的访问监视器，处理器执行一个下载专用指令后，监视器会对这个事件做一个标签。如果发生下列情况，处理器会移除这个专用访问标签。

- 执行一条 CLREX 指令。
- 执行一条 Store-Exclusive 指令，不管是否写成功。
- 发生异常时，表示处理器可以在不同线程间解决信号量冲突。

关于同步原语指令的更多细节，请参考“ ARM® Cortex™-M4 Devices Generic User Guide（文献编号 ARM DUI 0553A）”中的 Cortex™-M4 指令集章节。

2.5 异常模式

ARM Cortex-M4F 处理器和嵌套矢量中断控制器 (NVIC) 在处理模式对所有的异常进行优先级划分和处理。异常发生时处理器状态被自动存储到堆栈，中断服务程序 (ISR) 结束时又自动被恢复。向量的读取与状态保存并行，高效率进入中断。处理器支持尾链 (tail-chaining)，这样使得执行背靠背中断不需要重叠的状态保存和恢复。

表2-8（85页） 列出了所有的异常类型。软件可在七个异常（系统处理程序）以及 64 中断（在表 2-9（85页）中列出）上设置 8 个优先级。

系统处理程序的优先级由 NVIC 的系统处理程序优先级 n 寄存器 (SYSPRIn) 设定。中断是通过 NVIC 中断设置启用 n (ENn) 寄存器来启用的，并且由 NVIC 中断优先级 n (PRIn) 寄存器来区分其优先等级。优先级可以被分组为先发优先级和子优先级。在“嵌套式向量化中断控制器（NVIC）”（102页）中描述了所有中断寄存器。

在内部用户可编程的最高优先级 (0) 是第4优先级，按顺序在复位、非屏蔽中断 (NMI) 和硬件故障之后。注意 0 是所有可编程优先级的默认优先级。

重要: 在一次写操作清除中断源后，对于 NVIC 来说，可能需要花费几个处理器周期才能看到中断源被禁止。所以如果在中断处理程序中最后清除中断，有可能中断处理程序结束了但是NVIC看到中断仍然有效，导致错误的重新进入中断处理程序。这种情况可以避免，或者通过在中断处理程序开始时清除中断源，或者在写操作清除中断源后执行一个读或写操作 (刷新写缓冲器)。

关于异常和中断的具体信息，参见 “嵌套式向量化中断控制器（NVIC）”（102页）。

2.5.1 异常状态

每种异常都处于下列状态之一：

- 不活动的. 异常是不活动的，也不是挂起的。
- 待定. 异常正在等待处理器处理。来自外设或软件的中断请求可以将相应的中断变为挂起状态。
- 活动的. 处理器正在处理的异常，并且异常没有结束。

  注意: 一个异常处理程序可以中断另一个异常处理程序的执行。就此来说，两个异常都处于活动状态。

- 活动的和挂起的. 异常正被处理器处理，并且有一个挂起的异常来自相同的源。

2.5.2 异常类型

异常类型有：

- 复位值. 上电或热复位会引起复位。异常模式将复位看做一种特殊形式的异常。当复位有效时，在指令的任何时刻，处理器的操作都会停止。当复位不再有效时，从向量表中复位入口的地址重新开始执行。在线程模式中该执行是特权执行。
- **NMI.** 一个非屏蔽中断 (NMI) 可以使用 NMI 信号来发出通知或者使用中断控制及状态 (INTCTRL) 寄存器由软件触发。除了复位，该异常有最高的优先级。NMI 永久启用并拥有一个固定的优先级 -2。NMI 的激活不能被其他任何异常屏蔽或阻止，也不能被除复位外的其他任何异常抢占。
- 硬件故障. 硬故障是一个异常，它的发生是由于异常处理期间有错误，或者异常不能被任何异常机制管理。硬件故障拥有一个固定的优先级-1，表明它的优先级高于任何可配置的优先级。
- 存储器管理故障. 存储器管理故障是一个异常，它的发生是由于存在与存储器保护相关的故障（包括访问冲突和不匹配）。在处理指令存储器和数据存储器时，MPU 或固定存储器保护限制决定了该故障。该故障用来取消指令对不执行 (XN) 存储区域的访问，即使 MPU 是禁止的也不例外。
- 总线故障. 总线故障是一个异常，它的发生是由于在处理指令或数据存储器时发生了与存储器相关的故障，如预取错误或存储器访问故障。该故障可以被使能或禁止。
- 使用故障. 使用故障是一个异常，它的发生是由于指令执行的相关故障，如：
  - 一个未定义的指令
  - 一次非法的未对齐访问
  - 指令执行时的无效状态
  - 异常返回错误

  当内核被正确配置后，在字或半字存储器访问未对齐的地址或除以0都会引起一个使用故障。
- **SVCall.** 系统调用 (SVC) 是一个异常，它由 SVC 指令触发。在OS环境，应用程序可以使用SVC指令来访问OS内核函数和器件驱动。
- 调试监控器. 这个异常是由于调试监视器（没有停止时）引起的。该异常只有在使能时才激活。如果该异常的优先级低于当前的动作，那么它不会激活。
- **PendSV.** PendSV 是一个对系统级服务发出的请求，它可以挂起，并由中断驱动。在OS环境，当没有其它异常活动时，可使用PendSV作为背景切换。PendSV 由中断控制和状态寄存器 (INTCTRL) 触发。
- **SysTick.** SysTick 异常是在系统定时器启用中断时，系统定时器达到 0 时产生的。软件也可以通过中断控制和状态寄存器 (INTCTRL) 来产生一个 SysTick 异常。在OS环境，处理器可以使用该异常作为系统时标。
- 中断 **(IRQ).** 中断（即 IRO）是一个异常，它由外设标记，或者通过软件请求并由 NVIC（划分优先顺序的）反馈产生。所有的中断与指令执行都是异步的。在系统中，外设使用中断与处理器通信。表2-9（85页）列出了 TM4C1231H6PM 控制器上的中断。

对于异步异常，除复位外，处理器可以在异常触发和处理器进入异常处理程序之间执行其他指令。

特权软件可以禁止 表2-8（85页） 上显示的可配置优先级的异常（参见 150页 上的 SYSHNDCTRL 寄存器和 121页 上的 DIS0 寄存器）。

关于硬故障、存储器管理故障、总线故障和使用故障的更多信息，参见 “故障处理”（91页）。

表 2-8. 异常类型

异常类型	向量号	优先级[a]	向量地址或偏移量[b]	激活
-	0	-	0x0000.0000	复位时向量表的首入口
复位	1	-3 (最高)	0x0000.0004	异步
不可屏蔽的中断(NMI)	2	-2	0x0000.0008	异步
硬件故障	3	-1	0x0000.000C	-
存储器管理	4	可编程[c]	0x0000.0010	同步
总线故障	5	可编程[c]	0x0000.0014	精确时同步，不精确时异步
使用故障	6	可编程[c]	0x0000.0018	同步
-	7-10	-	-	保留
SVCall	11	可编程[c]	0x0000.002C	同步
调试监控器	12	可编程[c]	0x0000.0030	同步
-	13	-	-	保留
PendSV	14	可编程[c]	0x0000.0038	异步
SysTick	15	可编程[c]	0x0000.003C	异步
中断信号	16及其以上	可编程[d]	0x0000.0040 及其以上	异步

a. 0 是所有可编程优先级的默认优先级。

b. 请参阅“向量表”（87页）。

c. 参见 147页 上的 SYSPRI1 寄存器。

d. 参见 129页 上的 PRIn 寄存器。

表 2-9. 中断信号

向量号	中断编号（在中断寄存器中的位）	向量地址或偏移量	描述
0-15	-	0x0000.0000 - 0x0000.003C	处理器异常
16	0	0x0000.0040	GPIO 端口 A
17	1	0x0000.0044	GPIO 端口 B
18	2	0x0000.0048	GPIO 端口 C
19	3	0x0000.004C	GPIO 端口 D
20	4	0x0000.0050	GPIO 端口 E
21	5	0x0000.0054	UART0
22	6	0x0000.0058	UART1
23	7	0x0000.005C	SSI0
24	8	0x0000.0060	I^2C0
25-29	9-13	-	保留
30	14	0x0000.0078	ADC0 序列 0
31	15	0x0000.007C	ADC0 序列 1
32	16	0x0000.0080	ADC0 序列 2
33	17	0x0000.0084	模拟比较器 0
34	18	0x0000.0088	看门狗定时器 0 和 1
35	19	0x0000.008C	16/32 位定时器 0A

表 2-9. 中断信号（续）

向量号	中断编号（在中断寄存器中的位）	向量地址或偏移量	描述
36	20	0x0000.0090	16/32 位定时器 0B
37	21	0x0000.0094	16/32 位定时器 1A
38	22	0x0000.0098	16/32 位定时器 1B
39	23	0x0000.009C	16/32 位定时器 2A
40	24	0x0000.00A0	16/32 位定时器 2B
41	25	0x0000.00A4	模拟比较器0
42	26	0x0000.00A8	模拟比较器1
43	27	-	保留
44	28	0x0000.00B0	系统控制
45	29	0x0000.00B4	Flash 存储器控制和 EEPROM 控制
46	30	0x0000.00B8	GPIO 端口 F
47-48	31-32	-	保留
49	33	0x0000.00C4	UART2
50	34	0x0000.00C8	SSI1
51	35	0x0000.00CC	16/32 位定时器 3A
52	36	0x0000.00D0	16/32 位定时器 3B
53	37	0x0000.00D4	I^2C1
54	38	-	保留
55	39	0x0000.00DC	CAN0
56-58	40-42	-	保留
59	43	0x0000.00EC	睡眠模块
60-61	44-45	-	保留
62	46	0x0000.00F8	µDMA 软件
63	47	0x0000.00FC	µDMA 出错
64	48	0x0000.0100	ADC1 序列 0
65	49	0x0000.0104	ADC1 序列 1
66	50	0x0000.0108	ADC1 序列 2
67	51	0x0000.010C	ADC1 序列 3
68-72	52-56	-	保留
73	57	0x0000.0124	SSI2
74	58	0x0000.0128	SSI3
75	59	0x0000.012C	UART3
76	60	0x0000.0130	UART4
77	61	0x0000.0134	UART5
78	62	0x0000.0138	UART6
79	63	0x0000.013C	UART7
80-83	64-67	0x0000.0140 - 0x0000.014C	保留
84	68	0x0000.0150	I^2C2
85	69	0x0000.0154	I^2C3
86	70	0x0000.0158	16/32 位定时器 4A

表 2-9. 中断信号（续）

向量号	中断编号（在中断寄存器中的位）	向量地址或偏移量	描述
87	71	0x0000.015C	16/32 位定时器 4B
88-107	72-91	0x0000.0160 - 0x0000.01AC	保留
108	92	0x0000.01B0	16/32 位定时器 5A
109	93	0x0000.01B4	16/32 位定时器 5B
110	94	0x0000.01B8	32/64 位定时器 0A
111	95	0x0000.01BC	32/64 位定时器 0B
112	96	0x0000.01C0	32/64 位定时器 1A
113	97	0x0000.01C4	32/64 位定时器 1B
114	98	0x0000.01C8	32/64 位定时器 2A
115	99	0x0000.01CC	32/64 位定时器 2B
116	100	0x0000.01D0	32/64 位定时器 3A
117	101	0x0000.01D4	32/64 位定时器 3B
118	102	0x0000.01D8	32/64 位定时器 4A
119	103	0x0000.01DC	32/64 位定时器 4B
120	104	0x0000.01E0	32/64 位定时器 5A
121	105	0x0000.01E4	32/64 位定时器 5B
122	106	0x0000.01E8	系统异常（不精确）
63	47	-	保留

2.5.3 异常处理程序

处理器处理异常是通过使用：

- 中断服务程序 **(ISR).** 中断 (IRQx) 是由 ISR 处理的异常。
- 故障处理程序**.** 硬件障、存储器管理故障、使用故障以及总线故障都是故障异常，由故障处理程序处理。
- 系统处理程序**.** NMI、PendSV、SVCall、SysTick 和故障异常都是系统异常，由系统处理程序处理。

2.5.4 向量表

向量表包含了堆栈指针的复位值和开始地址，对于所有的异常处理程序也可称作异常向量。向量表由 表2-8（85页） 中示出的向量地址或偏移量组成。图2-6（88页） 示出向量表中异常向量的次序。每个向量的最低位必须为1，表示异常处理程序是Thumb码。

图 **2-6.** 向量表



系统复位时，向量表固定在地址 0x0000.0000。特权软件可以通过向向量表偏移量 (VTABLE) 寄存器进行写操作，来将向量表的开始地址重新定位为不同的存储器地址，范围是 0x0000.0400 至 0x3FFF.FC00（请参考“向量表”（87页））。注意在配置 VTABLE 寄存器时，偏移量必须在 1024 字节的边界对齐。

2.5.5 异常优先级

如 表2-8（85页）所示，所有的异常都有其相关的优先级，优先级的值低表示其优先级高，除复位、硬件复位和NMI外的所有异常都可以配置优先级。如果软件没有配置任何优先级，那么可配置优先级的所有异常的优先级值为 0。关于配置异常优先级的更多信息，参见 147页 和 129页。

注意: 对于 Tiva™ C 系列 的可配置优先级值的范围是 0-7。这就意味着带有负的优先级值的复位、硬故障和 NMI 等异常，总是比其他异常有更高的优先级。

例如，分配一个较高的优先级值给 IRQ[0] 同时分配一个较低的优先级值给 IRQ[1] ，表示 IRQ[1] 比 IRQ[0] 有更高的优先级。如果 IRQ[1] 和 IRQ[0] 都有效，那么 IRQ[1] 先于 IRQ[0] 被处理。

如果多个挂起的异常有相同的优先级，那么异常号最低的优先。例如，如果 IRQ[1] 和 IRQ[0] 都挂起并且它们有相同的优先级，那么 IRQ[0] 优先于 IRQ[1] 。

当处理器正在执行一个异常处理程序时，如果有个更高优先级的异常发生，那么它将取代当前的异常处理程序。如果有一个同样优先级的异常发生，那么不管其异常号如何都不会取代当前的异常处理程序。但是，新中断的状态变为挂起。

2.5.6 中断优先级分组

为提高系统中对中断优先级的控制，NVIC支持优先级分组。这个分组将中断优先级寄存器入口分为两个区域。

- 高区域定义组的优先级
- 低区域定义在同一组内的子优先级

只有组的优先级可以决定中断异常的取代。当处理器正在执行一个中断异常处理程序时，同优先级组的另一个中断不能取代当前的处理程序。

如果多个挂起的中断处于相同的优先级组，那么子优先级决定处理的顺序。如果多个挂起的中断有相同的组优先级和子优先级，那么最低IRQ编号的中断先被处理。

关于将中断优先级域分为组优先级和子优先级的更多信息，参见 141页。

2.5.7 异常进入和返回

异常处理的描述使用以下术语：

- 抢占式. 当处理器正在执行一个异常处理程序时，如果另一个异常的优先级更高，那么它可以抢占当前正在执行的异常处理程序。关于中断抢占的更多信息，参见“中断优先级分组”（89页）。当一个异常取代另一个时，它们称为嵌套异常。更多信息参见 “异常进入”（89页）。
- 返回. 当异常处理程序完成，并且没有挂起的有足够优先级的异常被服务，并且完成的异常处理程序不是一个后到的异常时，发生返回。处理器弹出堆栈并恢复到中断发生前的状态。更多信息参见 “异常返回”（90页）。
- 尾链. 使用尾链机制可加速异常处理。当一个异常处理程序完成时，如果有一个挂起的异常满足进入的要求，堆栈弹出将被跳过并且控制权直接转移到新的异常处理程序。
- 后到. 使用后到机制可加速抢占。如果在先前的异常正在保存状态期间有一个更高优先级的异常发生，那么处理器会切换到处理更高优先级的异常并开始为该异常取出向量。后到不会影响状态保存，因为对于前后两个异常来说，要保存的状态是一样的。所以，状态保存会没有中断的持续进行。处理器可以在直到先前异常处理程序的第一条指令进入到执行阶段时再接受后到的异常。从后到的异常处理程序返回时，正常的尾链规则有效。

2.5.7.1 异常进入

异常进入发生的条件是：有一个挂起的有足够优先级的异常，同时处理器处于线程模式，或者新的异常优先级高于正在处理的异常，这种情况新的异常将取代原先的异常。

当一个异常取代另一个异常时，它们是嵌套的。

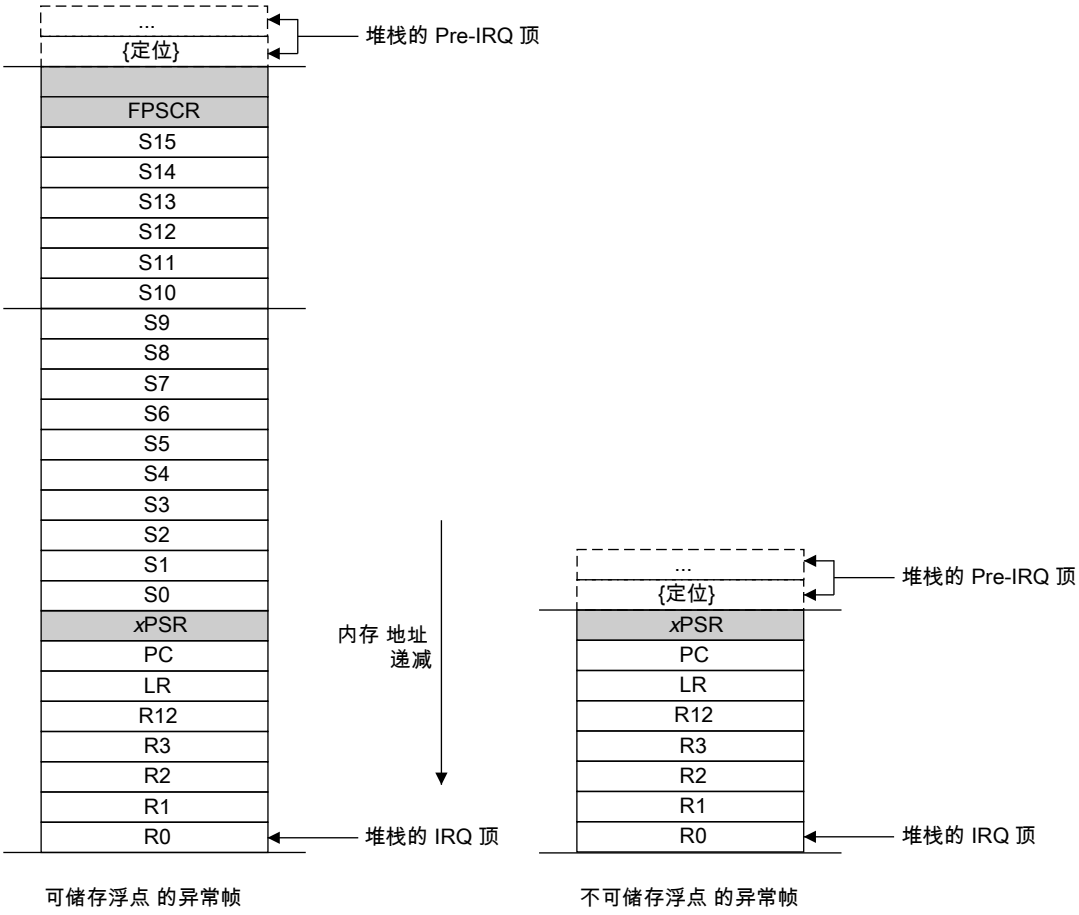
足够的优先级表示异常比屏蔽寄存器设置的任何限制都要更优先 (参见 68页 上的 PRIMASK；69页上的 FAULTMASK；以及 70页 上 的 BASEPRI)。比它的优先级低的异常会挂起而不是由处理器处理。

当处理器取得一个异常时，除非该异常是尾链的或后到的异常，否则处理器会将信息压入当前堆栈中。这个操作被称作*堆栈*，8 个数据字的结构被称作*堆栈框*。

当使用浮点程序时，Cortex-M4F 处理器自动将已建立的浮点状态压入异常入口堆栈。在 图 2-7（90页） 显示了当浮点状态由于中断或异常而保存在堆栈中时，Cortex-M4F 堆栈框的布局。

注意: 如果没有分配浮点状态的堆栈空间，该堆栈框就与不带有 FPU 的 ARMv7-M 具有相同结构。图2-7（90页） 也示出了堆栈框。

图 **2-7.** 异常堆栈框



入栈操作完成后，堆栈指针立即指示堆栈框的最低地址。

堆栈框包含返回地址，它是被中断程序的下一条指令的地址。当异常返回时这个地址重新载入PC，以便被中断的程序重新开始。

当入栈操作进行时，处理器并行从向量表读取异常处理程序的开始地址。当入栈操作完成时，处理器开始执行异常处理程序。同时，处理器将 EXC_RETURN 值写入 LR，指明哪个堆栈指针与堆栈框相对应，以及处理器在进入异常前处于何种操作模式。

如果在异常进入期间没有更高优先级的异常发生，处理器开始执行异常处理程序并自动将相应挂起的中断状态更改为活动的。

如果在异常进入期间有另一个更优先的异常发生，即后到，处理器开始执行后到的异常处理程序，并且不改变先前异常的挂起状态。

2.5.7.2 异常返回

异常返回发生在处理器的处理模式，处理器执行下列指令之一，将 EXC_RETURN 值装载到 PC：

- 装载 PC 的 POP 或 POP 指令
- 使用任意寄存器的 BX 指令
- 将 PC 作为目标的 LDR 指令

EXC_RETURN 是异常进入时载入 LR 的值。异常机制依赖这个值来检测处理器什么时候完成异常处理程序。这个值的低 5 位提供了有关返回堆栈和处理器模式的信息。表2-10（91页） 示出了 EXC_RETURN 值，包括其异常返回行为的描述。

EXC_RETURN 的位 31:5 都将置位。当这个值载入 PC 时，它表示对于处理器而言异常已经完成，处理器开始执行适合的异常返回序列。

表 **2-10.** 异常返回行为

EXC_RETURN[31:0]	描述
0xFFFF.FFE0	保留
0xFFFF.FFE1	返回到处理模式。 异常返回使用来自 MSP 的浮点状态。 返回后，执行程序将使用 MSP。
0xFFFF.FFE2 - 0xFFFF.FFE8	保留
0xFFFF.FFE9	返回到线程模式。 异常返回使用来自 MSP 的浮点状态。 返回后，执行程序将使用 MSP。
0xFFFF.FFEA - 0xFFFF.FFEC	保留
0xFFFF.FFED	返回到线程模式。 异常返回使用来自 MSP 的浮点状态。 返回后执行操作使用 PSP。
0xFFFF.FFEE - 0xFFFF.FFF0	保留
0xFFFF.FFF1	返回到处理模式。 异常返回使用来自 MSP 的非浮点状态。 返回后，执行程序将使用 MSP。
0xFFFF.FFF2 - 0xFFFF.FFF8	保留
0xFFFF.FFF9	返回到线程模式。 异常返回使用来自 MSP 的非浮点状态。 返回后，执行程序将使用 MSP。
0xFFFF.FFFA - 0xFFFF.FFFC	保留
0xFFFF.FFFD	返回到线程模式。 异常返回使用来自 PSP 的非浮点状态。 返回后执行操作使用 PSP。
0xFFFF.FFFE - 0xFFFF.FFFF	保留

2.6 故障处理

故障是异常的一个子集（见 “异常模式”（83页））。下面的条件产生一个故障：

- 在取指或载入向量表或访问数据时的总线错误。
- 内部检测出的错误，如没有定义的指令或尝试用一个 BX 指令更改状态。
- 尝试从一个标记为不可执行 (XN) 的存储区域执行指令。
- 由于权限冲突或尝试访问未管理区域而产生的 MPU 错误。

2.6.1 故障类型

表2-11（92页）示出了故障类型、用于处理故障的处理程序、相应的故障状态寄存器和指示故障发生的寄存器位。关于故障状态寄存器的更多信息，参见 153页。

表 **2-11.** 故障

故障	**Handler**	故障状态寄存器	位名字
读向量时的总线错误	硬故障	硬件故障状态 (HFAULTSTAT)	VECT
故障扩大到硬件故障	硬故障	硬件故障状态 (HFAULTSTAT)	FORCED
存取指令时，MPU 或默认的存储器不匹配	存储器管理故障	存储器管理故障状态 (MFAULTSTAT)	IERR [a]
存取数据时，MPU或默认的存储器不匹配	存储器管理故障	存储器管理故障状态 (MFAULTSTAT)	DERR
异常堆栈时，MPU或默认的存储器不匹配	存储器管理故障	存储器管理故障状态 (MFAULTSTAT)	MSTKE
异常退出堆栈时，MPU或默认的存储器不匹配	存储器管理故障	存储器管理故障状态 (MFAULTSTAT)	MUSTKE
在浮点怠惰状态保存过程中，MPU 或默认的存储器不匹配	存储器管理故障	存储器管理故障状态 (MFAULTSTAT)	MLSPERR
异常堆栈时的总线错误	总线故障	总线故障状态 (BFAULTSTAT)	BSTKE
异常退出堆栈时的总线错误	总线故障	总线故障状态 (BFAULTSTAT)	BUSTKE
预取指令时的总线错误	总线故障	总线故障状态 (BFAULTSTAT)	IBUS
在浮点怠惰状态保存过程中，总线错误	总线故障	总线故障状态 (BFAULTSTAT)	BLSPE
精确数据总线错误	总线故障	总线故障状态 (BFAULTSTAT)	PRECISE
非精确数据总线错误	总线故障	总线故障状态 (BFAULTSTAT)	IMPRE
尝试访问协处理器	用法故障	使用故障状态 (UFAULTSTAT)	NOCP
没有定义的指令	用法故障	使用故障状态 (UFAULTSTAT)	UNDEF
尝试进入无效指令集状态 [b]	用法故障	使用故障状态 (UFAULTSTAT)	INVSTAT
无效的 EXC_RETURN 值	用法故障	使用故障状态 (UFAULTSTAT)	INVPC
非法的未对齐下载或存储	用法故障	使用故障状态 (UFAULTSTAT)	UNALIGN
除以0	用法故障	使用故障状态 (UFAULTSTAT)	DIV0

a. 此故障出现在对 XN 区访问时（即使禁用了 MPU）。

b. 尝试使用一个非 Thumb 指令集的指令集，或者在 ICI 持续期间返回一个非下载存储复合指令。

2.6.2 故障扩大和硬件故障

除硬件故障外的所有故障异常都有可配置的异常优先级（参见 147页 上的 SYSPRI1）。软件可以禁止执行这些故障的处理程序（参见 150页 上的 SYSHNDCTRL）。

通常，异常优先级和异常屏蔽寄存器的值决定了处理器是否可以进入故障处理程序，一个故障处理程序是否可以取代另外一个故障处理程序，参见 “异常模式”（83页）。

在某些状况下，一个带有可配置优先级的故障可被看做一个硬件故障。该处理称为优先级扩大，该故障被描述为*扩大到硬件故障*。扩大到硬件故障发生在：

- 故障处理程序引起了与它所服务的故障类型相同的故障。这种扩大到硬件故障的发生是因为故障处理程序不能取代自身，因为它的优先级与当前优先级一样。
- 故障处理程序引起了与它所服务的故障优先级相同或更低的故障。这种情况的发生是因为新的故障处理程序不能取代当前正在执行的故障程序。

- 一个异常引起了故障，该故障的的优先级等于或低于当前正在执行的异常。
- 一个故障发生了，但是该故障的处理程序没有启用。

当进入一个总线故障处理程序时，如果在堆栈入栈期间发生了一个总线故障，该总线故障不会扩大到硬件故障。因此如果一个损坏的堆栈引起了一个故障，即使该处理程序的入栈失败，故障处理程序依然会执行。故障处理程序运行，但是堆栈内容是损坏的。

注意: 只有复位和 NMI 可以取代固定优先级的硬件故障。硬件故障可以取代除复位、NMI或另外硬件故障之外的任何异常。

2.6.3 故障状态寄存器和故障地址寄存器

故障状态寄存器显示了故障原因。对于总线故障和存储器管理故障，故障地址寄存器显示了造成故障的操作要访问的地址，参见 表2-12（93页）。

表 **2-12.** 故障状态寄存器和故障地址寄存器

Handler	状态寄存器名称	地址寄存器名称	寄存器描述
硬故障	硬件故障状态 (HFAULTSTAT)	-	159页
存储器管理故障	存储器管理故障状态 (MFAULTSTAT)	存储器管理故障地址寄存器（MMADDR）	153页 160页
总线故障	总线故障状态 (BFAULTSTAT)	总线故障地址寄存器（FAULTADDR）	153页 161页
用法故障	使用故障状态 (UFAULTSTAT)	-	153页

2.6.4 死锁

当处理器执行NMI或硬件故障处理程序时，如果一个硬件故障发生，那么处理器进入死锁状态。当处理器处于死锁状态时，它不执行任何指令。处理器将一直处于死锁状态，直至其置位、产生 NMI 或被调试器停止。

注意: 如果死锁状态从NMI处理程序引发，那么随后的NMI不会使处理器离开死锁状态。

2.7 电源管理

Cortex-M4F 处理器的睡眠模式减少了功耗：

- 睡眠模式停止处理器时钟。
- 深度睡眠模式停止系统时钟并关闭 PLL 和 Flash 存储器。

系统控制 (SYSCTRL) 寄存器中的 SLEEPDEEP 位用于选择睡眠模式（参见 143页）。关于睡眠模式行为的更多信息，参见 “系统控制”（199页）。

本节描述进入睡眠模式的机制和从睡眠模式唤醒的条件，它们对与睡眠模式和深度睡眠模式都适用。

2.7.1 进入睡眠模式

本节描述使用软件将处理器进入一种睡眠模式的机制。

系统可以产生假造的唤醒事件，如调试操作可唤醒处理器。所以软件必须能够在该事件后将处理器返回到睡眠模式。一段程序可能有一个空循环来将处理器返回到睡眠模式。

2.7.1.1 等待中断

等待中断指令 WFI 会导致立即进入睡眠模式，除非唤醒条件为真（请参考“从WFI或睡眠中退出(Sleep-on-Exit)唤醒”（94页））。当处理器执行一条 WFI 指令时，它将停止执行其他指令并进入睡眠模式。更多信息请参考 “ARM® Cortex™-M4 Devices Generic User Guide（文献编号 ARM DUI 0553A）”中的 Cortex™-M4 指令集章节。

2.7.1.2 等待事件

等待事件指令 WFE 会导致以 1 位事件寄存器的值为条件进入睡眠模式。当处理器执行一条 WFE 指令时，它会检测事件寄存器。如果寄存器为0，处理器停止执行指令并进入睡眠模式。如果寄存器为 1，处理器清零寄存器，接着继续执行指令而不进入睡眠模式。

如果事件寄存器为 1，处理器执行 WFE 指令时一定不会进入睡眠模式。通常，这种情况在执行 SEV 指令后发生。软件不能直接访问这个寄存器。

更多信息请参考 “ARM® Cortex™-M4 Devices Generic User Guide（文献编号 ARM DUI 0553A）”中的 Cortex™-M4 指令集章节。

2.7.1.3 睡眠中退出 (Sleep-on-Exit)

如果 SYSCTRL 寄存器中的 SLEEPEXIT 位置位，当处理器完成所有异常处理器程序的执行后，它将返回线程模式并立即进入睡眠模式。这种机制可用于异常发生时需要处理器运行的情况。

2.7.2 从睡眠模式唤醒

处理器唤醒的条件依赖于促使其进入睡眠模式的机制。

2.7.2.1 从WFI或睡眠中退出(Sleep-on-Exit)唤醒

通常，只有在 NVIC 检测到一个异常，并且该异常的优先级足够导致异常进入时，处理器才被唤醒。有些嵌入式系统可能会在处理器唤醒后执行中断处理程序前必须执行系统恢复任务。置位 PRIMASK 位和清零 FAULTMASK 位可以延迟进入中断处理程序。如果一个中断使能并且比当前异常的优先级高，该中断到来时处理器唤醒，但直到处理器清零 PRIMASK 后才执行中断处理程序。关于 PRIMASK 和 FAULTMASK 的更多信息，参见 68页 和 69页。

2.7.2.2 从WFE 唤醒

处理器如果检测到一个足够优先执行的异常，处理器会唤醒。

另外，如果 SYSCTRL 寄存器的 SEVONPEND 位置位，任何新挂起的中断，即使该中断被禁止或没有足够的优先级执行，都会触发一个事件并唤醒处理器。关于 SYSCTRL 的更多信息，参见 143 页。

2.8 指令集总结

该处理器执行一个Thumb指令集版本。表2-13（95页） 列出了所支持的指令。

注意: 在 表2-13（95页） 中：

- 尖括号 <> 包含了操作数的复用形式
- 大括号 {} 包含了可选的操作数
- 操作数列是不全面的
- 0p2 是第二操作数，它可以是一个寄存器，也可以是一个常数。
- 大部分指令可以使用一个可选的状态码后缀

有关指令和操作数的更多信息，请参考 “ARM® Cortex™-M4 技术参考手册”中的指令描述。

表 2-13. Cortex-M4F 指令摘要

助记符	操作数	简要描述	标志
ADC, ADCS	{Rd,} Rn, Op2	带进位加法	N,Z,C,V
ADD, ADDS	{Rd,} Rn, Op2	加法	N,Z,C,V
ADD, ADDW	{Rd,} Rn , #imm12	加法	-
ADR	Rd, label	载入PC相对地址	-
AND, ANDS	{Rd,} Rn, Op2	逻辑与	N,Z,C
ASR, ASRS	Rd, Rm, <Rs|#n>	算术右移	N,Z,C
B	label	转移	-
BFC	Rd, #lsb, #width	位域清零	-
BFI	Rd, Rn, #lsb, #width	位域插入	-
BIC, BICS	{Rd,} Rn, Op2	位清零	N,Z,C
BKPT	#imm	断点	-
BL	label	带连接转移	-
BLX	Rm	带连接的间接转移	-
BX	Rm	间接转移	-
CBNZ	Rn, label	比较非零转移	-
CBZ	Rn, label	比较为零转移	-
CLREX	-	清除互斥	-
CLZ	Rd, Rm	计算前导0的数目	-
CMN	Rn, Op2	负向比较	N,Z,C,V
CMP	Rn, Op2	比较	N,Z,C,V
CPSID	i	改变处理器状态，禁止中断	-
CPSIE	i	改变处理器状态，使能中断	-
DMB	-	数据存储隔离	-
DSB	-	数据同步隔离	-
EOR, EORS	{Rd,} Rn, Op2	近位异或	N,Z,C
ISB	-	指令同步隔离	-
IT	-	If-Then 条件块	-
LDM	Rn{!}, reglist	加载多个寄存器，加载后自增加	-
LDMDB, LDMEA	Rn{!}, reglist	加载多个寄存器，加载前自减	-
LDMFD, LDMIA	Rn{!}, reglist	加载多个寄存器，加载后自增加	-
LDR	Rt, [Rn, #offset]	从寄存器加载字	-
LDRB, LDRBT	Rt, [Rn, #offset]	从寄存器中加载字节	-
LDRD	Rt, Rt2, [Rn, #offset]	从寄存器中加载双字节	-
LDREX	Rt, [Rn, #offset]	加载寄存器，标记互斥	-
LDREXB	Rt, [Rn]	从寄存器加载字节，标记互斥	-
LDREXH	Rt, [Rn]	从寄存器加载半字，标记互斥	-
LDRH, LDRHT	Rt, [Rn, #offset]	从寄存器加载半字	-
LDRSB, LDRSBT	Rt, [Rn, #offset]	从寄存器加载带符号的字节	-
LDRSH, LDRSHT	Rt, [Rn, #offset]	从寄存器加载带符号的半字	-
LDRT	Rt, [Rn, #offset]	从寄存器加载字	-
LSL, LSLS	Rd, Rm, <Rs|#n>	逻辑左移	N,Z,C
LSR, LSRS	Rd, Rm, <Rs|#n>	逻辑右移	N,Z,C

表 2-13. Cortex-M4F 指令摘要（续）

助记符	操作数	简要描述	标志
MLA	Rd, Rn, Rm, Ra	乘加，32位结果	-
MLS	Rd, Rn, Rm, Ra	乘减，32位结果	-
MOV, MOVS	Rd, Op2	加载	N,Z,C
MOV, MOVW	Rd, #imm16	加载16位常数	N,Z,C
MOVT	Rd, #imm16	加载高位	-
MRS	Rd, spec_reg	从特殊寄存器加载到通用寄存器	-
MSR	spec_reg, Rm	从通用寄存器加载到特殊寄存器	N,Z,C,V
MUL, MULS	{Rd,} Rn, Rm	乘法，32位结果	N,Z
MVN, MVNS	Rd, Op2	取反加载	N,Z,C
NOP	-	无操作	-
ORN, ORNS	{Rd,} Rn, Op2	逻辑或取反	N,Z,C
ORR, ORRS	{Rd,} Rn, Op2	逻辑或	N,Z,C
PKHTB, PKHBT	{Rd,} Rn, Rm, Op2	包半字	-
POP	reglist	从堆栈中弹出到寄存器	-
PUSH	reglist	将寄存器值压入堆栈	-
QADD	{Rd,} Rn, Rm	饱和加法	Q
QADD16	{Rd,} Rn, Rm	饱和加法 (16)	-
QADD8	{Rd,} Rn, Rm	饱和加法 (8)	-
QASX	{Rd,} Rn, Rm	可互调的饱和加法和饱和减法	-
QDADD	{Rd,} Rn, Rm	饱和倍乘加法	Q
QDSUB	{Rd,} Rn, Rm	饱和倍乘减法	Q
QSAX	{Rd,} Rn, Rm	可互调的饱和减法和饱和加法	-
QSUB	{Rd,} Rn, Rm	饱和减法	Q
QSUB16	{Rd,} Rn, Rm	饱和减法 (16)	-
QSUB8	{Rd,} Rn, Rm	饱和减法 (8)	-
RBIT	Rd, Rn	位反转	-
REV	Rd, Rn	在一个字中反转字节顺序	-
REV16	Rd, Rn	在每个半字中反转字节顺序	-
REVSH	Rd, Rn	反转低半字的字节顺序，带符号扩展	-
ROR, RORS	Rd, Rm, <Rs|#n>	循环右移	N,Z,C
RRX, RRXS	Rd, Rm	带进位的循环右移	N,Z,C
RSB, RSBS	{Rd,} Rn, Op2	反向减法	N,Z,C,V
SADD16	{Rd,} Rn, Rm	带符号的加法 (16)	GE
SADD8	{Rd,} Rn, Rm	带符号的加法 (8)	GE
SASX	{Rd,} Rn, Rm	可互调的带符号加法和减法	GE
SBC, SBCS	{Rd,} Rn, Op2	带借位的减法	N,Z,C,V
SBFX	Rd, Rn, #lsb, #width	带符号位域扩展	-
SDIV	{Rd,} Rn, Rm	带符号除法	-
SEL	{Rd,} Rn, Rm	选择字节	-
SEV	-	发送事件	-
SHADD16	{Rd,} Rn, Rm	带符号的半字加法 (16)	-
SHADD8	{Rd,} Rn, Rm	带符号的半字加法 (8)	-

表 2-13. Cortex-M4F 指令摘要（续）

助记符	操作数	简要描述	标志
SHASX	{Rd,} Rn, Rm	可互调的带符号半字加法和半字减法	-
SHSAX	{Rd,} Rn, Rm	可互调的带符号半字加法和半字减法	-
SHSUB16	{Rd,} Rn, Rm	带符号的半字减法 (16)	-
SHSUB8	{Rd,} Rn, Rm	带符号的半字减法 (8)	-
SMLABB, SMLABT, SMLATB, SMLATT	Rd, Rn, Rm, Ra	带符号的长整型（半字）乘加	Q
SMLAD, SMLADX	Rd, Rn, Rm, Ra	带符号的倍乘加	Q
SMLAL	RdLo, RdHi, Rn, Rm	带符号乘加 (32x32+64), 64位结果	-
SMLALBB, SMLALBT, SMLALTB, SMLALTT	RdLo, RdHi, Rn, Rm	带符号的长整型（半字）乘加	-
SMLALD, SMLALDX	RdLo, RdHi, Rn, Rm	带符号的长整型整数（半字）倍乘加	-
SMLAWB,SMLAWT	Rd, Rn, Rm, Ra	带符号的乘加，字乘半字	Q
SMLSD SMLSDX	Rd, Rn, Rm, Ra	带符号的倍乘减	Q
SMLSLD SMLSLDX	RdLo, RdHi, Rn, Rm	带符号的长整型（半字）倍乘减	
SMMLA	Rd, Rn, Rm, Ra	带符号的最高有效字乘加	-
SMMLS, SMMLR	Rd, Rn, Rm, Ra	带符号的最高有效字乘减	-
SMMUL, SMMULR	{Rd,} Rn, Rm	带符号的最高有效字乘法	-
SMUAD SMUADX	{Rd,} Rn, Rm	带符号的倍乘加	Q
SMULBB, SMULBT, SMULTB, SMULTT	{Rd,} Rn, Rm	带符号的半字乘法	-
SMULL	RdLo, RdHi, Rn, Rm	带符号乘法 (32x32), 64位结果	-
SMULWB, SMULWT	{Rd,} Rn, Rm	带符号乘法（乘半字）	-
SMUSD, SMUSDX	{Rd,} Rn, Rm	带符号的倍乘减	-
SSAT	Rd, #n, Rm {,shift #s}	带符号的饱和运算	Q
SSAT16	Rd, #n, Rm	带符号的饱和运算 (16)	Q
SSAX	{Rd,} Rn, Rm	可互调的饱和减法和饱和加法	GE
SSUB16	{Rd,} Rn, Rm	带符号的减法 (16)	-
SSUB8	{Rd,} Rn, Rm	带符号的减法 (8)	-
STM	Rn{!}, reglist	保存多个寄存器，保存后自增加	-

表 2-13. Cortex-M4F 指令摘要（续）

助记符	操作数	简要描述	标志
STMDB, STMEA	Rn{!}, reglist	保存多个寄存器，保存前自减	-
STMFD, STMIA	Rn{!}, reglist	保存多个寄存器，保存后自增加	-
STR	Rt, [Rn {, #offset}]	保存寄存器字	-
STRB, STRBT	Rt, [Rn {, #offset}]	保存寄存器字节	-
STRD	Rt, Rt2, [Rn {, #offset}]	保存寄存器双字	-
STREX	Rt, Rt, [Rn {, #offset}]	在互斥状态保存寄存器	-
STREXB	Rd, Rt, [Rn]	在互斥状态保存寄存器字节	-
STREXH	Rd, Rt, [Rn]	在互斥状态保存寄存器半字	-
STRH, STRHT	Rt, [Rn {, #offset}]	保存寄存器半字	-
STRSB, STRSBT	Rt, [Rn {, #offset}]	保存寄存器带符号字节	-
STRSH, STRSHT	Rt, [Rn {, #offset}]	保存寄存器带符号半字	-
STRT	Rt, [Rn {, #offset}]	保存寄存器字	-
SUB, SUBS	{Rd,} Rn, Op2	减法	N,Z,C,V
SUB, SUBW	{Rd,} Rn, #imm12	减12位常数	N,Z,C,V
SVC	#imm	系统服务调用	-
SXTAB	{Rd,} Rn, Rm, {,ROR #}	将 8 位扩展到 32 位后做加法运算	-
SXTAB16	{Rd,} Rn, Rm,{,ROR #}	将 8 位扩展到 16 位后做加法运算	-
SXTAH	{Rd,} Rn, Rm,{,ROR #}	将 16 位扩展到 32 位后做加法运算	-
SXTB16	{Rd,} Rm {,ROR #n}	带符号扩展字节 (16)	-
SXTB	{Rd,} Rm {,ROR #n}	带符号扩展一个字节	-
SXTH	{Rd,} Rm {,ROR #n}	带符号扩展一个半字	-
TBB	[Rn, Rm]	查表转移字节	-
TBH	[Rn, Rm, LSL #1]	查表转移半字	-
TEQ	Rn, Op2	测试是否相等	N,Z,C
TST	Rn, Op2	测试	N,Z,C
UADD16	{Rd,} Rn, Rm	无符号的加法 (16)	GE
UADD8	{Rd,} Rn, Rm	无符号的加法 (8)	GE
UASX	{Rd,} Rn, Rm	可互调的无符号加法和减法	GE
UHADD16	{Rd,} Rn, Rm	无符号的半字加法 (16)	-
UHADD8	{Rd,} Rn, Rm	无符号的半字加法 (8)	-
UHASX	{Rd,} Rn, Rm	可互调的无符号半字加法和半字减法	-
UHSAX	{Rd,} Rn, Rm	可互调的无符号半字减法和半字加法	-
UHSUB16	{Rd,} Rn, Rm	无符号的半字减法 (16)	-
UHSUB8	{Rd,} Rn, Rm	无符号的半字减法 (8)	-
UBFX	Rd, Rn, #lsb, #width	无符号位域扩展	-
UDIV	{Rd,} Rn, Rm	无符号除法	-
UMAAL	RdLo, RdHi, Rn, Rm	无符号的长整型乘加 (32x32+64), 64 位结果	-
UMLAL	RdLo, RdHi, Rn, Rm	无符号乘加 (32 x 32 + 32 + 32), 64 位结果	-
UMULL	RdLo, RdHi, Rn, Rm	无符号乘法 (32x32), 64位结果	-
UQADD16	{Rd,} Rn, Rm	无符号的饱和加法 (16)	-

表 2-13. Cortex-M4F 指令摘要（续）

助记符	操作数	简要描述	标志
UQADD8	{Rd,} Rn, Rm	无符号的饱和加法 (8)	-
UQASX	{Rd,} Rn, Rm	可互调的无符号饱和加法和饱和减法	-
UQSAX	{Rd,} Rn, Rm	可互调的无符号饱和减法和饱和加法	-
UQSUB16	{Rd,} Rn, Rm	无符号的饱和减法 (16)	-
UQSUB8	{Rd,} Rn, Rm	无符号的饱和减法 (8)	-
USAD8	{Rd,} Rn, Rm	无符号的绝对差和	-
USADA8	{Rd,} Rn, Rm, Ra	无符号的绝对差及累加和	-
USAT	Rd, #n, Rm {,shift #s}	无符号饱和操作	Q
USAT16	Rd, #n, Rm	无符号的饱和操作 (16)	Q
USAX	{Rd,} Rn, Rm	可互调的无符号减法和加法	GE
USUB16	{Rd,} Rn, Rm	无符号的减法 (16)	GE
USUB8	{Rd,} Rn, Rm	无符号的减法 (8)	GE
UXTAB	{Rd,} Rn, Rm, {,ROR #}	轮换，将 8 位扩展到 32 位后做加法运算	-
UXTAB16	{Rd,} Rn, Rm, {,ROR #}	轮换，将 8 位扩展到 16 位后做加法运算	-
UXTAH	{Rd,} Rn, Rm, {,ROR #}	轮换，无符号扩展并加半字	-
UXTB	{Rd,} Rm, {,ROR #n}	使用零扩展一个字节	-
UXTB16	{Rd,} Rm, {,ROR #n}	无符号扩展字节 (16)	-
UXTH	{Rd,} Rm, {,ROR #n}	使用零扩展一个半字	-
VABS.F32	Sd, Sm	浮点绝对	-
VADD.F32	{Sd,} Sn, Sm	浮点加法	-
VCMP.F32	Sd, <Sm | #0.0>	比较两个浮点型寄存器，或者是浮点型寄存器和零	FPSCR
VCMPE.F32	Sd, <Sm | #0.0>	比较两个浮点型寄存器，或者是带有无效操作校验的浮点型寄存器和零	FPSCR
VCVT.S32.F32	Sd, Sm	在浮点型数据和整数之间的转换	-
VCVT.S16.F32	Sd, Sd, #fbits	在浮点型数据和定点数据之间的转换	-
VCVTR.S32.F32	Sd, Sm	在浮点型数据和舍位整数之间的转换	-
VCVT<B|H>.F32.F16	Sd, Sm	将半精度的值转换为单精度值	-
VCVTT<B|T>.F32.F16	Sd, Sm	将单精度的值转换为半精度值	-
VDIV.F32	{Sd,} Sn, Sm	浮点除法	-
VFMA.F32	{Sd,} Sn, Sm	浮点熔合乘加	-
VFNMA.F32	{Sd,} Sn, Sm	浮点熔合乘加取反	-
VFMS.F32	{Sd,} Sn, Sm	浮点熔合乘减	-
VFNMS.F32	{Sd,} Sn, Sm	浮点熔合乘减取反	-
VLDM.F<32|64>	Rn{!}, list	加载多个扩展寄存器	-
VLDR.F<32|64>	<Dd|Sd>, [Rn]	从存储器加载扩展寄存器	-
VLMA.F32	{Sd,} Sn, Sm	浮点乘加	-
VLMS.F32	{Sd,} Sn, Sm	浮点乘减	-
VMOV.F32	Sd, #imm	浮点立即移动	-
VMOV	Sd, Sm	浮点移动寄存器	-
VMOV	Sn, Rt	将 ARM 内核寄存器复制到单精度	-
VMOV	Sm, Sm1, Rt, Rt2	将 2 个 ARM 内核寄存器复制到 2 个单精度	-

表 2-13. Cortex-M4F 指令摘要（续）

助记符	操作数	简要描述	标志
VMOV	Dd[x], Rt	将 ARM 内核寄存器复制到标量	-
VMOV	Rt, Dn[x]	将标量复制到 ARM 内核	-
VMRS	Rt, FPSCR	将 FPSCR 移动到 ARM 内核寄存器或 APSR	N,Z,C,V
VMSR	FPSCR, Rt	从 ATM 内核寄存器移动到 FPSCR	FPSCR
VMUL.F32	{Sd,} Sn, Sm	浮点乘法	-
VNEG.F32	Sd, Sm	浮点取反	-
VNMLA.F32	{Sd,} Sn, Sm	浮点乘加	-
VNMLS.F32	{Sd,} Sn, Sm	浮点乘减	-
VNMUL	{Sd,} Sn, Sm	浮点乘法	-
VPOP	list	从扩展寄存器弹出	-
VPUSH	list	压入扩展寄存器	-
VSQRT.F32	Sd, Sm	计算浮点数据平方根	-
VSTM	Rn{!}, list	浮点寄存器多存储	-
VSTR.F3<32|64>	Sd, [Rn]	将扩展寄存器存到存储器	-
VSUB.F<32|64>	{Sd,} Sn, Sm	浮点除法	-
WFE	-	等待事件	-
WFI	-	等待中断	-

3 Cortex-M4 外设

本章提供关于 Tiva™ C 系列 Cortex-M4 处理器外设的信息，包括：

- SysTick（见第101页）

  提供一个简单易用、配置灵活的24位单调递减计数器。该计数器具有写入即清零、过零自动重载等特性。

- 嵌套式向量化中断控制器 (NVIC)（见第102页）
  - 提供低-等待延时异常和中断处理
  - 可控制电源管理
  - 执行系统控制寄存器

- 系统控制模块（SCB）（见第103页）

  可提供系统构成信息，并进行系统控制，包括系统异常的配置、控制以及上报。

- 存储器保护单元（MPU）（见第104页）

  支持标准的ARMv7受保护存储器系统架构（PMSA）模型。MPU提供保护区支持，重叠保护区，访问权限和导出存储属性到系统。

- 浮点单元 (FPU)（见第108页）

  完全支持单精度的加、减、乘、除、乘加以及平方根操作。它还可用于转换定点和浮点数据格式，并提供浮点常数指令。

表3-1（101页）显示专用外设总线（PPB）的地址映射。某些外设寄存器空间还会进一步划分为两个区，在表中分别列为两组地址。

表 **3-1.** 内核外设寄存器区域

地址	内核级外设	描述（见页面）
0xE000.E010-0xE000.E01F	系统定时器	101
0xE000.E100-0xE000.E4EF 0xE000.EF00-0xE000.EF03	嵌套式向量化中断控制器	102
0xE000.E008-0xE000.E00F 0xE000.ED00-0xE000.ED3F	系统控制模块	103
0xE000.ED90-0xE000.EDB8	存储器保护单元	104
0xE000.EF30-0xE000.EF44	浮点单元	108

3.1 功能说明

本章提供关于 Tiva™ C 系列 Cortex-M4 处理器外设的信息：SysTick、NVIC、SCB、MPU、FPU。

3.1.1 系统定时器（SysTick）

Cortex-M4 内核集成有一个系统定时器 SysTick，提供简单易用、配置灵活的 24 位单调递减计数器，还具有写入即清零、过零自动重载等特性。该计数器的用途广泛，举例来说，可以：

- 用作RTOS的节拍定时器，按照可编程的频率（例如100Hz）定时触发，调用系统定时器服务子程序。

- 用作一个使用系统时钟的高速报警定时器。

- 速率可变的报警或信号定时器—时间延迟的范围由使用的参考时钟和计数器的动态范围决定。
- 用作简单计数器，测量任务的完成时刻、总体耗时等等；
- 用于实现基于失配/匹配周期的内部时钟源控制。此时通过查询 STCTRL 控制及状态寄存器的 COUNT 标志位，可以判定某个动作是否在指定的时间内完成；以此作为动态时钟管理控制环的一部分。

系统定时器包含以下3个寄存器：

- SysTick 控制和状态 (STCTRL)：控制和状态计数器用来配置其时钟、使能计数器、使能SysTick 中断以及确定计数器状态。
- SysTick 重载值 (STRELOAD)：计数器的重装值，用来提供计数器的重装值（wrap value）。
- SysTick 当前值 (STCURRENT)：计数器的当前值。

使能系统定时器后，计数器将在每个时钟递减一次，从重载值逐个递减到 0，之后在下一个时钟沿翻转（重载 STRELOAD 寄存器的值），之后继续每个时钟递减一次，如此周而复始。如果将 STRELOAD 寄存器清零，则会在下次重载时禁用该计数器。当计数器递减到 0 时，COUNT 标志位将置位。读取 COUNT 标志位后其自动清零。

对 STCURRENT 寄存器进行写操作，即可将此寄存器清零，同时还将清零 COUNT 标志位。这个写操作并不会触发SysTick异常逻辑。读取该寄存器时，返回值是该寄存器被访问时刻的内容。

SysTick 计数器根据系统时钟或精确内部振荡器 (PIOSC) 4 分频运行。假如在某些低功耗模式下停止提供该时钟信号，则系统定时器的计数器也将停止运行。在 SysTick 控制和状态寄存器 (STCTRL) 中设置 CLK_SRC 位并确保深度睡眠时钟配置 (DSLPCLKCFG) 寄存器中的 PIOSCPD 位已清零，从而使 SysTick 在深度休眠模式中保持运行状态。软件在访问系统定时器的寄存器时，应确保始终采用字对齐操作予以访问。

在复位时，SysTick 计数器的重载值和当前值并未定义；SysTick 计数器的正确初始化序列为：

1. 将 STRELOAD 寄存器中的值编程。
2. 向 STCURRENT 寄存器中写入任意值来将其清零。
3. 配置 STCTRL 寄存器以执行所需操作。

注意: 在调试过程中处理器暂停时，此计数器不会递减。

3.1.2 嵌套式向量化中断控制器（NVIC）

本节介绍嵌套式向量化中断控制器（NVIC）及其寄存器。NVIC支持：

- 64 个中断。
- 每个中断的优先级均可编程，取值范围0~7。优先级数字越大则其优先级越低，也就是说0代表最高优先级；
- 可实现异常及中断的快速响应处理；
- 中断信号可以是电平检测或脉冲检测；
- 动态重设中断优先级；
- 优先级可分组，划分为分组优先级域以及子优先级域；

- 支持咬尾中断；
- 提供一个外部的不可屏蔽中断（NMI）。

处理器在异常入口处能够自动将状态入栈，在退出异常时能够自动将状态出栈。这个过程是处理器自行完成的、无需多余的指令开销，因此可快速响应异常并进行处理。

3.1.2.1 电平式中断及脉冲式中断

处理器支持电平式中断及脉冲式中断。脉冲式中断通常又称为边沿触发中断。

对于电平式中断而言，只要外设产生中断信号就会始终保持处于触发状态，直到外设中断信号复原后才不再触发。一般来说这需要ISR（中断服务子程序）对外设进行操作，使得外设不再产生中断请求信号。脉冲中断是在处理器时钟的上升沿同步采样的中断信号。为了确保 NVIC 能够检测到中断，外设所产生的中断信号必须保持至少一个时钟周期，在此期间 NVIC 可检测到脉冲并锁存中断。

处理器进入 ISR 后，将自动清除该中断的挂起状态（参见“中断的硬件控制及软件控制”（103页）以了解更多信息）。对于电平式中断，如果处理器从 ISR 返回后，中断信号仍未复原，则中断将再次挂起，于是处理器必须再次运行其 ISR。外设可以像这样保持中断信号持续享用服务，直到其不再需要服务为止。

3.1.2.2 中断的硬件控制及软件控制

Cortex-M4 内核能够锁存所有中断。当满足以下条件之一，外设中断即变为挂起状态：

- NVIC检测到某个中断信号为高电平，并且该中断为未激活状态；
- NVIC检测到某个中断信号的上升沿；
- 软件对中断设置挂起寄存器的相应位写操作，或向软件触发中断寄存器 (SWTRIG) 的相应位写操作，形成软件产生中断的挂起。请参阅 123页 上 PEND0 寄存器中的 INT 位或 133页 上的 SWTRIG。

中断挂起后将保持挂起状态，直到满足以下条件之一：

- 处理器进入该中断的 ISR，将中断状态由挂起改为已激活。然后：
  - 对于电平式中断，当处理器从ISR返回后，NVIC将再次采样中断信号。假如仍然检测到中断信号，那么中断状态将再次变为挂起，使得处理器立即重新进入该ISR。否则，中断状态将变为未激活。
  - 对于脉冲式中断，处理器时刻监视着中断信号的状态。只要检测到中断脉冲信号就会将中断状态改为挂起并激活。在此情况下，当处理器从ISR返回后，中断状态将再次变为挂起，使得处理器立即重新进入该ISR。

    如果当处理器在ISR内期间并未产生中断脉冲信号，那么当处理器从ISR返回后，中断状态将变为未激活。
- 软件对中断清除挂起寄存器中的相应标志位执行写操作
  - 对于电平式中断，若仍旧产生中断信号，那么中断状态将保持不变。否则，中断状态将变为未激活。
  - 对于脉冲式中断，假如中断状态是挂起并激活，则将变为未激活。

3.1.3 系统控制模块 (SCB)

系统控制模块提供系统构成信息，并能实现系统控制功能，包括系统异常的配置、控制以及上报。

3.1.4 存储器保护单元 (MPU)

本节介绍存储器保护单元（MPU）。MPU将存储器映射空间划分为若干个存储器区，并分别规定每一个存储器区的起始地址、大小、访问权限以及存储属性。MPU支持为每一个存储器区分别定义其属性设置，支持重叠区的设置，此外还能将存储属性导出到系统。

存储属性影响对该存储器区进行访问时的表现。Cortex-M4 MPU 定义了八个单独存储器区 (0-7) 以及背景区。

当存储区出现重叠时，重叠部分的访问将由编号最大的存储区属性决定。例如，第7存储器区始终优先于其它存储器区，因此一旦与其它区发生重叠时，都将以第7区的存储属性为准。

背景区的存储器访问属性与默认的存储器映射相同，但只允许特权级软件依此进行访问。

Cortex-M4 的 MPU 存储器映射是统一的，也就是说指令访问和数据访问都是共用相同的存储区设置。

假如程序试图访问某个地址，而该地址被MPU设置为禁止访问，那么处理器将产生一个存储管理故障，并由此触发故障异常。这个异常可能会导致操作系统环境下某个进程终止运行。在操作系统环境下，操作系统内核能够按照进程的实际需求来动态更新MPU区的设置。一般来说，嵌入式操作系统都需要通过MPU实现存储器保护。

MPU 区的配置是基于存储器类型的（参见“内存区，类型和属性”（77页）以了解更多信息）。

表3-2（104页） 显示 MPU 区的可能属性。请参阅 “一个 Tiva™ C 系列 微控制器的 MPU 配置”一节（108页） 以了解关于将微控制器实施编程的准则。

表 **3-2.** 存储器属性摘要

存储器类型	描述
严格排序	对严格顺序存储器的所有访问必须按照程序顺序进行。
设备	存储器映射外部设备
正常模式	普通存储器

为避免出现无法预料的执行结果，如果某些中断的处理函数可能访问某存储器区，那么在更新该存储器区的属性之前，应当先关闭这些中断。

在访问MPU寄存器时，应确保软件按照正确的宽度进行对齐访问：

- 除 MPU 区属性及大小 (MPUATTR) 寄存器外，所有 MPU 寄存器必须按字对齐进行访问；
- MPUATTR 寄存器可按字节对齐、半字对齐或字对齐进行访问。

处理器不支持对MPU寄存器进行未对齐访问。

在配置MPU时，由于MPU可能曾经被编程过，因此所有当前不用的存储器区都应当禁用，防止存储器区的旧设置对当前更新后的MPU设置产生不良影响。

3.1.4.1 更新单个**MPU**区

要更新某个 MPU 区的属性，必须更新 MPU 区编号寄存器 (MPUNUMBER)、MPU 区基地址寄存器 (MPUBASE) 以及 MPUATTR 寄存器。用户可以分别编程各个寄存器，也可以利用多字写操作来同时编程所有寄存器。您可以使用 MPUBASEx 和 MPUATTRx 别名，通过使用 STM 指令同时对最多四个区进行编程。

采用单字写操作，更新单个**MPU**区

下面的例程可以完成单个MPU区的配置：

```
; R1 = region number
; R2 = size/enable
; R3 = attributes
; R4 = address
LDR R0,=MPUNUMBER        ; 0xE000ED98, MPU region number register
STR R1, [R0, #0x0]       ; Region Number
STR R4, [R0, #0x4]       ; Region Base Address
STRH R2, [R0, #0x8]      ; Region Size and Enable
STRH R3, [R0, #0xA]      ; Region Attribute
```

假如之前启用了某个存储器区，又需要更改其设置，那么在向MPU写入新的设置之前应当先将此存储器区禁用。例如：

```
; R1 = region number
; R2 = size/enable
; R3 = attributes
; R4 = address
LDR R0,=MPUNUMBER         ; 0xE000ED98, MPU region number register
STR R1, [R0, #0x0]        ; Region Number
BIC R2, R2, #1            ; Disable
STRH R2, [R0, #0x8]       ; Region Size and Enable
STR R4, [R0, #0x4]        ; Region Base Address
STRH R3, [R0, #0xA]       ; Region Attribute
ORR R2, #1                ; Enable
STRH R2, [R0, #0x8]       ; Region Size and Enable
```

在以下情况下，软件中必须采用存储器隔离指令：

- 假如在修改MPU配置时，可能存在某些仍在进行中的存储器传输过程（例如缓冲式写操作），则该操作将可能受到MPU配置改变的影响。
- 此时在配置MPU之前必须采用存储器隔离指令。

假如进入异常处理程序后立即开始配置MPU，或配置完MPU后立即从异常中返回，那么可以不采用存储器隔离指令，因为异常入口和异常出口已经能够实现相当于存储器隔离指令的保护机制。

在配置MPU期间，并不需要软件再增加存储器隔离指令，这是因为访问MPU时必须通过私有外设总线（Private Peripheral Bus，简写为PPB），而PPB是严格顺序存储区。

举例来说，如果要让所有的存储器访问行为在编程序列后立即生效，则应使用 DSB 指令和 ISB 指令。更改 MPU 配置后应当立即发出一条 DSB 指令，例如上下文切换的结尾。如果对 MPU 存储区编程的代码是通过分支语句或调用语句进入的，那么应当发出一条 ISB 指令。如果编程序列使用异常返回或通过进入异常来输入，那么无需发出 ISB 指令。

采用多字写操作，更新单个**MPU**区

取决于写入信息如何分割，MPU也可以直接采用多字写操作进行编程。例如：

```
; R1 = region number
; R2 = address
; R3 = size, attributes in one
LDR R0, =MPUNUMBER  ; 0xE000ED98, MPU region number register
STR R1, [R0, #0x0]  ; Region Number
STR R2, [R0, #0x4]  ; Region Base Address
STR R3, [R0, #0x8]  ; Region Attribute, Size and Enable
```

STM 指令可以用于优化：

```
; R1 = region number
; R2 = address
; R3 = size, attributes in one
LDR R0, =MPUNUMBER  ; 0xE000ED98, MPU region number register
STM R0, {R1-R3}     ; Region number, address, attribute, size and enable
```

此操作可以按预整合信息的两个字来完成，表示 MPU 区基地址 (MPUBASE) 寄存器（请参阅 166 页）包含所需区域号且具有 VALID 位集。这种方式比较适合静态配置数据的应用场合，例如对于 Bootloader，就能够进一步精简其大小。

```
; R1 = address and region number in one
; R2 = size and attributes in one
LDR R0, =MPUBASE    ; 0xE000ED9C, MPU Region Base register
STR R1, [R0, #0x0]  ; Region base address and region number combined
                    ; with VALID (bit 4) set
STR R2, [R0, #0x4]  ; Region Attribute, Size and Enable
```

存储子区

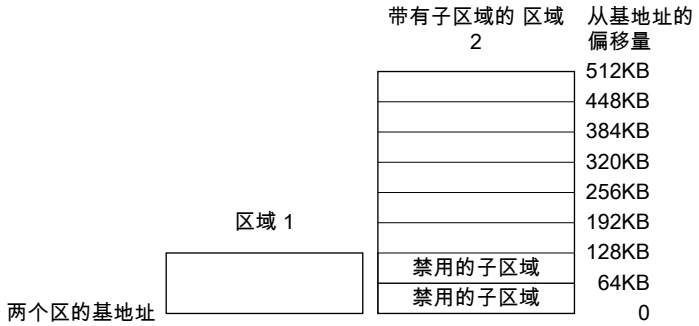
任何不小于256字节的存储器区还能进一步均分为8个存储子区。在 MPU 区属性及大小 (MPUATTR) 寄存器的 SRD 域中设置相应位（请参阅 168页）以禁用存储子区。SRD 位域的最低位控制着第一个子区，最高位控制着最后一个子区。禁用某个子区表示另一区域重叠禁用的范围匹配。如果禁用某个子区后，也没有其它已启用的存储器区重叠在该区域上，那么MPU会产生一个故障。

如果存储器区的大小为32、64或128字节，是无法再划分为子区的。对于这些大小的区域，SRD 域必须配置为 0x00，否则 MPU 行为将无法预测。

SRD的用法示例

在下面的例子中，两个存储器区具有相同的基地址，因此它们相互重叠。第1个区的大小为128kB，第2个区的大小为512kB。为了确保来自第一个区域的属性应用于第一个 128 KB 区域，请将第二个区域的 SRD 域配置为 0x03 以禁用前两个子区域，如 图3-1（106页） 所示。

图 **3-1. SRD** 使用示例



3.1.4.2 MPU访问权限属性

MPUATTR 寄存器中的访问权限位（TEX、S、C、B、AP、XN）负责控制相应存储区的访问权限。假如试图访问某个区域，而该区域并未开放相关权限，那么MPU将会产生一个访问权限故障。

表3-3（107页） 显示 TEX、C、B, 和 S 访问权限位的编码。出于完整方面的考虑，表中列出了所有可能的编码组合，不过目前的 Cortex-M4 内核尚不支持可高速缓冲性以及可共享性。请参阅 “一个

Tiva™ C 系列 微控制器的 MPU 配置”一节（108页） 以了解对 MPU 编程以实施 TM4C1231H6PM 的相关信息。

表 **3-3. TEX**、**S**、**C** 和 **B** 位域编码

TEX	S	C	B	存储器类型	可共享性	其他属性
000b	X[a]	0	0	严格排序	可共享	-
000	X[a]	0	1	设备	可共享	-
000	0	1	0	正常模式	不可共享	外部及内部均为完全写入式。无写分配。
000	1	1	0	正常模式	可共享	
000	0	1	1	正常模式	不可共享	
000	1	1	1	正常模式	可共享	
001	0	0	0	正常模式	不可共享	外部及内部均不可缓冲。
001	1	0	0	正常模式	可共享	
001	X[a]	0	1	保留的编码	-	-
001	X[a]	1	0	保留的编码	-	-
001	0	1	1	正常模式	不可共享	外部及内部均为回写式。写及读均分配
001	1	1	1	正常模式	可共享	
010	X[a]	0	0	设备	不可共享	不可共享的设备。
010	X[a]	0	1	保留的编码	-	-
010	X[a]	1	X[a]	保留的编码	-	-
1BB	0	A	A	正常模式	不可共享	可高速缓冲的存储器（BB代表外部策略，A代表内部策略）。 关于 AA 和 BB 位的编码，参见表3-4。
1BB	1	A	A	正常模式	可共享	

a. MPU 将忽略此位的值。

表3-4（107页） 显示存储器属性编码（TEX 值在范围 0x4-0x7 中）的高速缓存策略。

表 **3-4.** 存储器属性编码对应的高速缓存策略

编码（**AA** 或 **BB**）	相应的高速缓存策略
00	不可高速缓冲
01	回写式，写及读均分配
10	完全写入式，无写分配
11	回写式，无写分配

表3-5（107页） 显示 MPUATTR 寄存器中的 AP 编码，这些编码定义特权软件和非特权软件的访问权限。

表 **3-5. AP** 位域编码

AP 位域	特权权限	非特权权限	描述
000	无访问权	无访问权	任何访问操作都会导致产生一个访问权限故障。
001	R/W	无访问权	仅特权级软件才拥有访问权限。
010	R/W	RO	非特权级软件的写操作会导致产生一个访问权限故障。
011	R/W	R/W	任何软件均可任意访问。
100	无法预测	无法预测	保留

表 **3-5. AP** 位域编码（续）

AP 位域	特权权限	非特权权限	描述
101	RO	无访问权	仅特权级软件才拥有只读权限。
110	RO	RO	特权级软件或非特权级软件均拥有只读权限。
111	RO	RO	特权级软件或非特权级软件均拥有只读权限。

一个 **Tiva™ C** 系列 微控制器的 **MPU** 配置

Tiva™ C 系列 微控制器只有一个处理器，并且不带高速缓存。因此，应按 表3-6（108页）中所示对 MPU 编程。

表 **3-6. Stellaris Tiva™ C** 系列 微控制器的存储器区属性

存储器区域	**TEX**	**S**	**C**	**B**	存储器类型和属性
Flash 存储器	000b	0	1	0	普通存储器，不可共享，完全写入式
内部 SRAM	000b	1	1	0	普通存储器，可共享，完全写入式
外部 SRAM	000b	1	1	1	普通存储器，可共享，回写式，写分配
外设	000b	1	0	1	设备存储器，可共享

在当前的 Tiva™ C 系列 微控制器的实施中，可共享性和高速缓冲策略属性对系统行为并无影响。不过，如果正确配置这些MPU区属性，有益于充分保障代码的可移植性。上表给出的都是常见配置的参考值。

3.1.4.3 MPU失配

若某个访问操作违反了 MPU 权限，处理器将会产生存储器管理故障（请参阅“异常和中断”（75页）以了解更多信息）。MFAULTSTAT 寄存器指示故障原因。更多信息参见 153页。

3.1.5 浮点单元 (FPU)

本节描述浮点单元 (FPU) 及其使用的寄存器。FPU 提供：

- 适用于单精度（C 浮点）数据处理操作的 32 位指令
- 适用于更高精度（熔合 MAC）的组合乘加指令
- 适用于换算、加法、减法、可选累加的乘法、除法和平方根计算的硬件支持
- 适用于反常和所有 IEEE 舍入模式的硬件支持
- 32 个专用的 32 位单精度寄存器，也可作为 16 位双字寄存器进行寻址
- 去耦三级流水线

Cortex-M4F FPU 完全支持单精度的加法、减法、乘法、除法、乘加和平方根计算。它还可用于转换定点和浮点数据格式，并提供浮点常数指令。FPU 提供浮点计算功能，该功能符合 ANSI/IEEE 754-2008 标准（适用于二进制浮点运算的 IEEE 标准，也称为 IEEE 754 标准）。FPU 的单精度扩展寄存器也可以作为 16 位双字寄存器，并支持加载、存储和移动操作。

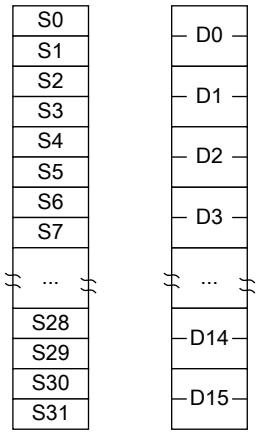
3.1.5.1 寄存器块的 **FPU** 视图

FPU 提供包含 32 个单精度寄存器的扩展寄存器文件。这些寄存器可视为：

- 十六个 64 位双字寄存器，D0-D15

- 三十二个 32 位单字寄存器，S0-S31
- 以上寄存器的组合

图 **3-2. FPU** 寄存器块



寄存器之间的映射如下：

- S<2n> 映射到 D<n> 低位的那一半
- S<2n+1> 映射到 D<n> 高位的那一半

例如，您可以通过访问 S12 来访问 D6 中的低位值，通过访问 S13 来访问该元件的高位值。

3.1.5.2 工作模式

FPU 提供三种工作模式以适应多种应用程序的需要。

完全合规性模式. 在完全合规性模式下，FPU 按照硬件中的 IEEE 754 标准来处理所有工作。

清零模式. 将浮点状态和控制 (FPSC) 寄存器的 FZ 位置位可启用清零模式。在该模式下，FPU 将算术 CDP 操作的所有非规格化输入操作数作为 0 来处理。由 0 操作数造成的异常会发出相应的信号。VABS、VNEG 和 VMOV 不被视为算术 CDP 操作，因此不受清零模式的影响。根据 IEEE 754 标准，如果某个结果极小，其目标精度的大小小于舍入前的最小正常值，则该结果将被 0 取代。FPSC 寄存器中的 IDC 位指示清空输入的时间。FPSC 寄存器中的 UFC 位指示清空结果的时间。

默认 **NaN** 模式. 将 FPSC 寄存器中的 DN 位置位可启用默认 NaN 模式。在该模式下，包含输入 NaN 或产生 NaN 结果的任何算术数据处理操作的结果都会返回默认 NaN。小数位的传播仅由 VABS、VNEG 和 VMOV 操作维持。所有其他 CDP 操作会忽略输入 NaN 的小数位中的任何信息。

3.1.5.3 符合 IEEE 754 标准

当禁用默认 NaN (DN) 和清零 (FZ) 模式时，FPv4 功能在硬件上符合 IEEE 754 标准。实现此合规性不需要支持代码。

3.1.5.4 完全实现 IEEE 754 标准

Cortex-M4F 浮点指令集并不完全支持 IEEE 754-2008 标准中定义的所有操作。不支持的操作包括但不限于：

- 余数
- 将浮点数舍入为整数值的浮点数

- 二进制转十进制
- 十进制转二进制
- 单精度和双精度值的直接比较

根据 IEEE 标准，Cortex-M4 FPU 支持熔合 MAC 操作。要完全实现 IEEE 754-2008 标准，浮点功能必须随库函数增加。

3.1.5.5 IEEE 754 标准实现选项

NaN 处理

所有具有最大指数域值和非零小数域的单精度值都是有效的 NaN。最高小数位为 0 指示信号非数 (SNaN)。最高小数位为 1 指示静默非数 (QNaN)。如果两个 NaN 的任何一位都不同，则将其视为不同的 NaN 值。下表显示了默认的 NaN 值。

符号	小数	小数
0	0xFF	位 [22] = 1，位 [21:0] 全为 0

对 ARM 浮点功能和库的输入 NaN 的处理方式定义如下：

- 在完全合规性模式下，NaN 按《ARM 架构参考手册》中的描述处理。硬件直接将 NaN 用于算术 CDP 指令。对于数据传输操作，NaN 被传输而不引起无效操作异常。对于非算术 CDP 指令、VABS、VNEG 和 VMOV，如果指令中有说明，则 NaN 被复制且符号改变，而不引起无效操作异常。
- 在默认 NaN 模式下，涉及 NaN 操作数的算术 CDP 指令将返回默认 NaN，而不考虑 NaN 操作数的小数。算术 CDP 操作中的 SNaN 将 IOC 标志 FPSCR[0] 置位。由数据传输和非算术 CDP 指令进行的 NaN 处理与在完全合规性模式下一样。

表 **3-7. QNaN** 和 **SNaN** 处理

指令类型	默认 **NaN** 模式	带有 **QNaN** 操作数	带有 **SNaN** 操作数
算术 CDP	关闭	如果有多个 QNaN 操作数，则将按《ARM 架构参考手册》中的规则返回一个 QNaN 或 QNaN 操作数。	IOC[a] 置位。SNaN 最高有效位被清除，得出的 NaN 取决于《ARM 架构参考手册》中的规则。
	开启	默认 NaN 返回。	IOC[a] 置位。默认 NaN 返回。
非算术 CDP	关闭/开启	NaN 传输到目的地，符号会发生相应改变。	
FCMP(Z)	-	无序比较。	IOC 置位。无序比较。
FCMPE(Z)	-	IOC 置位。无序比较。	IOC 置位。无序比较。
加载/存储	关闭/开启	传输所有 NaN。	

a. IOC 是无效操作异常标志，FPSCR[0]。

比较

比较结果会修改 FPSCR 中的标志。您可以使用 MVRS APSR_nzcv 指令（即之前的 FMSTAT）将当前标志从 FPSCR 传输到 APSR。有关将 IEEE 754-2008 标准谓词映射到 ARM 条件的详细信息，请参考《ARM 架构参考手册》。应仔细选择所使用的标志，使后续的 ARM 指令在指定执行条件下能测试 IEEE 标准中定义的谓词。

下溢

Cortex-M4F FPU 使用 IEEE 754-2008 标准中描述的舍入前的极小值和丢失精度的不准确结果来产生下溢异常。

在清零模式下，根据 IEEE 标准，舍入前极小的结果将清空为 0，且 UFC 标志 FPSCR[3] 置位。有关清零模式的信息，请参考《ARM 架构参考手册》。

当 FPU 未处于清零模式时，将对非规格化操作数进行操作。如果操作没有产生极小的结果，它将返回计算结果，且 UFC 标志 FPSCR[3] 不置位。如果操作不精确，IXC 标志 FPSCR[4] 置位。如果操作产生了极小的结果，该结果为非规格化值或 0，如果结果同样不精确，UFC 标志 FPSCR[3] 置位。

3.1.5.6 异常

根据 FPv4 架构，FPU 将如每条指令所要求地将 FPSCR 寄存器中的累积异常状态标志置位。FPU 不支持用户模式处理。FPSCR 中的异常启用位读数为 0 并且忽略写操作。处理器也有六个输出管脚：FPIXC、FPUFC、FPOFC、FPDZC、FPIDC 和 FPIOC，每个管脚反映一个累积异常标志的状态。有关这些输出的描述，请参考 “ARM Cortex-M4 集成和实现手册”（ARM DII 0239，可通过 ARM 获取）。

该处理器可通过使用怠惰堆栈来降低异常延迟。请参考第 4-5 页上的辅助控制寄存器，ACTLR。这意味着处理器在堆栈上保留了 FP 状态的空间，但并未在堆栈上保存该状态的信息。有关更多信息，请参考《ARMv7-M 架构参考手册》（可通过 ARM 获取）。

3.1.5.7 启用 FPU

FPU 通过复位被禁用。您必须先启用 FPU，才能使用浮点指令。处理器必须处于特权模式，才能读取/写入协处理器访问控制 (CPAC) 寄存器。以下示例代码序列可同时在特权模式和用户模式中启用 FPU。

```
; CPACR is located at address 0xE000ED88
LDR.W R0, =0xE000ED88
; Read CPACR
LDR R1, [R0]
; Set bits 20-23 to enable CP10 and CP11 coprocessors
ORR R1, R1, #(0xF << 20)
; Write back the modified value to the CPACR
STR R1, [R0]; wait for store to complete
DSB
;reset pipeline now the FPU is enabled
ISB
```

3.2 寄存器映射

表 3-8（111页）列出了 Cortex-M4 外设 SysTick、NVIC、MPU、FPU 和 SCB 寄存器。列出的偏移量是相对于 0xE000.E000。

注意: 未使用的寄存器空间都是为将来或内部使用保留的。软件不得修改任何保留的存储器地址。

表 **3-8.** 外设 寄存器映射

偏移量	名称	类型	复位	描述	见页面
系统定时器（**SysTick**）寄存器					
0x010	STCTRL	R/W	0x0000.0004	SysTick 控制及状态寄存器	115
0x014	STRELOAD	R/W	-	SysTick 重载值寄存器	117
0x018	STCURRENT	R/WC	-	SysTick 当前值寄存器	118

表 **3-8.** 外设 寄存器映射（续）

偏移量	名称	类型	复位	描述	见页面
嵌套式向量化中断控制器（**NVIC**）寄存器					
0x100	EN0	R/W	0x0000.0000	中断 0-31 置位启用寄存器	119
0x104	EN1	R/W	0x0000.0000	中断 32-63 置位启用寄存器	119
0x108	EN2	R/W	0x0000.0000	中断 64-95 置位启用寄存器	119
0x10C	EN3	R/W	0x0000.0000	中断 96-127 置位启用寄存器	119
0x110	EN4	R/W	0x0000.0000	中断 128-138 设置启用寄存器	120
0x180	DIS0	R/W	0x0000.0000	中断 0-31 清除启用寄存器	121
0x184	DIS1	R/W	0x0000.0000	中断 32-63 清除启用寄存器	121
0x188	DIS2	R/W	0x0000.0000	中断 64-95 清除启用寄存器	121
0x18C	DIS3	R/W	0x0000.0000	中断 96-127 清除启用寄存器	121
0x190	DIS4	R/W	0x0000.0000	中断 128-138 清除启用寄存器	122
0x200	PEND0	R/W	0x0000.0000	中断 0-31 置位挂起寄存器	123
0x204	PEND1	R/W	0x0000.0000	中断 32-63 置位挂起寄存器	123
0x208	PEND2	R/W	0x0000.0000	中断 64-95 置位挂起寄存器	123
0x20C	PEND3	R/W	0x0000.0000	中断 96-127 置位挂起寄存器	123
0x210	PEND4	R/W	0x0000.0000	中断 128-138 置位挂起寄存器	124
0x280	UNPEND0	R/W	0x0000.0000	中断 0-31 清除挂起寄存器	125
0x284	UNPEND1	R/W	0x0000.0000	中断 32-63 清除挂起寄存器	125
0x288	UNPEND2	R/W	0x0000.0000	中断 64-95 清除挂起寄存器	125
0x28C	UNPEND3	R/W	0x0000.0000	中断 96-127 清除挂起寄存器	125
0x290	UNPEND4	R/W	0x0000.0000	中断 128-138 清除挂起寄存器	126
0x300	ACTIVE0	RO	0x0000.0000	中断 0-31 活动位寄存器	127
0x304	ACTIVE1	RO	0x0000.0000	中断 32-63 活动位寄存器	127
0x308	ACTIVE2	RO	0x0000.0000	中断 64-95 活动位寄存器	127
0x30C	ACTIVE3	RO	0x0000.0000	中断 96-127 活动位寄存器	127
0x310	ACTIVE4	RO	0x0000.0000	中断 128-138 活动位寄存器	128
0x400	PRI0	R/W	0x0000.0000	中断 0-3 优先级寄存器	129
0x404	PRI1	R/W	0x0000.0000	中断 4-7 优先级寄存器	129
0x408	PRI2	R/W	0x0000.0000	中断 8-11 优先级寄存器	129
0x40C	PRI3	R/W	0x0000.0000	中断 12-15 优先级寄存器	129
0x410	PRI4	R/W	0x0000.0000	中断 16-19 优先级寄存器	129
0x414	PRI5	R/W	0x0000.0000	中断 20-23 优先级寄存器	129
0x418	PRI6	R/W	0x0000.0000	中断 24-27 优先级寄存器	129

表 3-8. 外设 寄存器映射（续）

偏移量	名称	类型	复位	描述	见页面
0x41C	PRI7	R/W	0x0000.0000	中断 28-31 优先级寄存器	129
0x420	PRI8	R/W	0x0000.0000	中断 32-35 优先级寄存器	129
0x424	PRI9	R/W	0x0000.0000	中断 36-39 优先级寄存器	129
0x428	PRI10	R/W	0x0000.0000	中断 40-43 优先级寄存器	129
0x42C	PRI11	R/W	0x0000.0000	中断 44-47 优先级寄存器	129
0x430	PRI12	R/W	0x0000.0000	中断 48-51 优先级寄存器	129
0x434	PRI13	R/W	0x0000.0000	中断 52-55 优先级寄存器	129
0x438	PRI14	R/W	0x0000.0000	中断 56-59 优先级寄存器	129
0x43C	PRI15	R/W	0x0000.0000	中断 60-63 优先级寄存器	129
0x440	PRI16	R/W	0x0000.0000	中断 64-67 优先级寄存器	131
0x444	PRI17	R/W	0x0000.0000	中断 68-71 优先级寄存器	131
0x448	PRI18	R/W	0x0000.0000	中断 72-75 优先级寄存器	131
0x44C	PRI19	R/W	0x0000.0000	中断 76-79 优先级寄存器	131
0x450	PRI20	R/W	0x0000.0000	中断 80-83 优先级寄存器	131
0x454	PRI21	R/W	0x0000.0000	中断 84-87 优先级寄存器	131
0x458	PRI22	R/W	0x0000.0000	中断 88-91 优先级寄存器	131
0x45C	PRI23	R/W	0x0000.0000	中断 92-95 优先级寄存器	131
0x460	PRI24	R/W	0x0000.0000	中断 96-99 优先级寄存器	131
0x464	PRI25	R/W	0x0000.0000	中断 100-103 优先级寄存器	131
0x468	PRI26	R/W	0x0000.0000	中断 104-107 优先级寄存器	131
0x46C	PRI27	R/W	0x0000.0000	中断 108-111 优先级寄存器	131
0x470	PRI28	R/W	0x0000.0000	中断 112-115 优先级寄存器	131
0x474	PRI29	R/W	0x0000.0000	中断 116-119 优先级寄存器	131
0x478	PRI30	R/W	0x0000.0000	中断 120-123 优先级寄存器	131
0x47C	PRI31	R/W	0x0000.0000	中断 124-127 优先级寄存器	131
0x480	PRI32	R/W	0x0000.0000	中断 128-131 优先级寄存器	131
0x484	PRI33	R/W	0x0000.0000	中断 132-135 优先级寄存器	131
0x488	PRI34	R/W	0x0000.0000	中断 136-138 优先级寄存器	131
0xF00	SWTRIG	WO	0x0000.0000	软件触发中断寄存器	133
系统控制模块（**SCB**）寄存器					
0x008	ACTLR	R/W	0x0000.0000	辅助控制寄存器	134
0xD00	CPUID	RO	0x410F.C241	CPU ID 基础寄存器	136
0xD04	INTCTRL	R/W	0x0000.0000	中断控制及状态寄存器	137

表 **3-8.** 外设 寄存器映射（续）

偏移量	名称	类型	复位	描述	见页面
0xD08	VTABLE	R/W	0x0000.0000	向量表寄存器	140
0xD0C	APINT	R/W	0xFA05.0000	应用程序中断及复位控制寄存器	141
0xD10	SYSCTRL	R/W	0x0000.0000	系统控制寄存器	143
0xD14	CFGCTRL	R/W	0x0000.0200	配置及控制寄存器	145
0xD18	SYSPRI1	R/W	0x0000.0000	系统处理程序优先级寄存器 1	147
0xD1C	SYSPRI2	R/W	0x0000.0000	系统处理程序优先级寄存器 2	148
0xD20	SYSPRI3	R/W	0x0000.0000	系统处理程序优先级寄存器 3	149
0xD24	SYSHNDCTRL	R/W	0x0000.0000	系统处理程序控制及状态寄存器	150
0xD28	FAULTSTAT	R/W1C	0x0000.0000	可配置故障状态寄存器	153
0xD2C	HFAULTSTAT	R/W1C	0x0000.0000	硬故障状态寄存器	159
0xD34	MMADDR	R/W	-	存储器管理故障地址寄存器	160
0xD38	FAULTADDR	R/W	-	总线故障地址寄存器	161
存储器保护单元（**MPU**）寄存器					
0xD90	MPUTYPE	RO	0x0000.0800	MPU 类型寄存器	162
0xD94	MPUCTRL	R/W	0x0000.0000	MPU 控制寄存器	163
0xD98	MPUNUMBER	R/W	0x0000.0000	MPU 区编号寄存器	165
0xD9C	MPUBASE	R/W	0x0000.0000	MPU 区基地址寄存器	166
0xDA0	MPUATTR	R/W	0x0000.0000	MPU 区属性和大小寄存器	168
0xDA4	MPUBASE1	R/W	0x0000.0000	MPU 区基地址别名寄存器 1	166
0xDA8	MPUATTR1	R/W	0x0000.0000	MPU 区属性和大小别名寄存器 1	168
0xDAC	MPUBASE2	R/W	0x0000.0000	MPU 区基地址别名寄存器 2	166
0xDB0	MPUATTR2	R/W	0x0000.0000	MPU 区属性和大小别名寄存器 2	168
0xDB4	MPUBASE3	R/W	0x0000.0000	MPU 区基地址别名寄存器 3	166
0xDB8	MPUATTR3	R/W	0x0000.0000	MPU 区属性和大小别名寄存器 3	168
浮点单元 **(FPU)** 寄存器					
0xD88	CPAC	R/W	0x0000.0000	协处理器访问控制	170
0xF34	FPCC	R/W	0xC000.0000	浮点上下文控制	171
0xF38	FPCA	R/W	-	浮点上下文访问	173
0xF3C	FPDSC	R/W	0x0000.0000	浮点默认状态控制	174

3.3 系统定时器（SysTick）寄存器描述

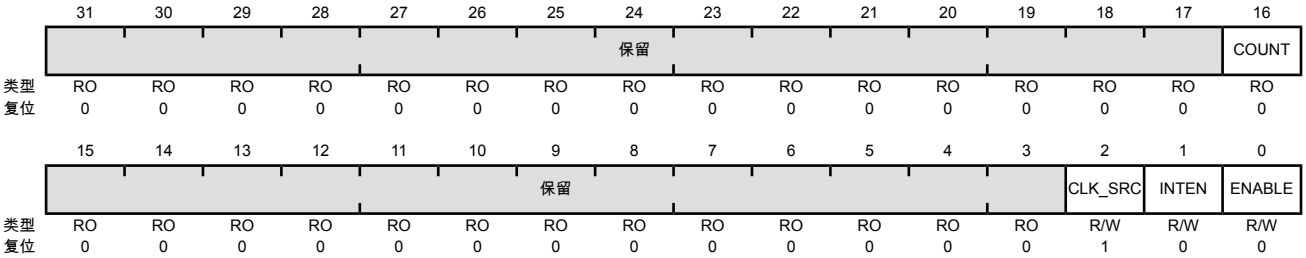
本节按照地址偏移量由小到大的顺序依次详细介绍系统定时器的各寄存器。

寄存器 1: SysTick 控制及状态寄存器（STCTRL），偏移量 0x010

注意: 本寄存器只能在特权模式下访问。

SysTick STCTRL 寄存器启用 SysTick 功能。

SysTick 控制及状态寄存器 (STCTRL)
基址 0xE000.E000
偏移量 0x010
类型 R/W, 复位 0x0000.0004



位/域	名称	类型	复位	描述
31:17	保留	RO	0x000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
16	COUNT	RO	0	计数标志 值 描述 0 自上一次读取此位之后，SysTick 定时器尚未计数到 0。 1 自上一次读取本标志位之后，系统定时器曾经计数到0。 读取本寄存器后，本标志位将自动清零；此外，若对 STCURRENT 寄存器写入任意值，本标志位也将自动清零。 如果调试器使用 DAP 读取，仅在 AHB-AP 控制寄存器中的 MasterType 位清零时，该位才清零。否则，COUNT 位不会因为调试器的读操作而改变。关于 MasterType 的更多信息，请参考“《ARM® 调试接口 V5 架构规范》”。
15:3	保留	RO	0x000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
2	CLK_SRC	R/W	1	时钟源 值 描述 0 精确内部振荡器 (PIOSC) 4 分频 1 采用系统时钟。
1	INTEN	R/W	0	中断启用 值 描述 0 禁用中断产生软件可以使用 COUNT 位来确定计数器是否曾达到 0。 1 当系统定时器计数到0时，向NVIC产生一个中断。

位/域	名称	类型	复位	描述
0	ENABLE	R/W	0	启用

值	描述
0	禁用计数器。
1	使 SysTick 以多次触发方式进行操作。即，计数器装入 RELOAD 值并开始递减计数。达到 0 时，COUNT 位将置位，如果由 INTEN 启用，将生成一个中断。然后，计数器重新装入 RELOAD 值并开始计数。

寄存器 2: SysTick 重载值寄存器（STRELOAD），偏移量 0x014

注意: 本寄存器只能在特权模式下访问。

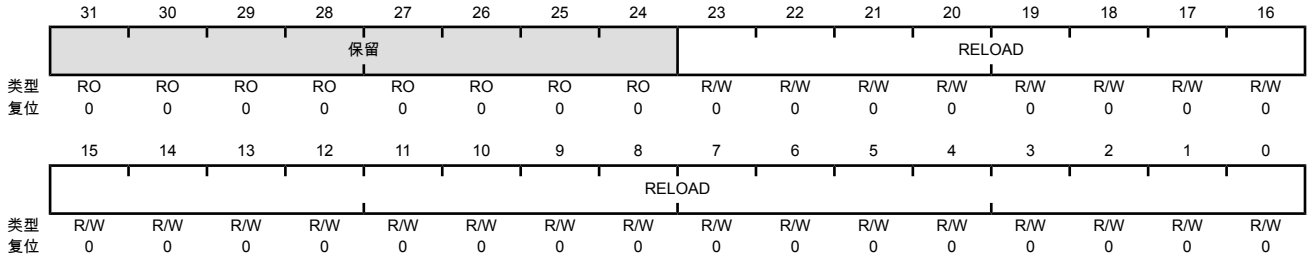
STRELOAD 寄存器指定在计数器达到 0 的时候要装入 SysTick 当前值 (STCURRENT) 寄存器的开始值。开始值可以在 0x1 和 0x00FF.FFFF 之间。因为从 1 计数到 0 时激活了 SysTick 中断和 COUNT 位，所以开始值 0 是可能的，但是没有效果。

系统定时器按照自动重载方式工作，每N+1个时钟动作一次（N的取值范围为0x0000 0001~0x00FF FFFF），如是周而复始、永不停歇。举例来说，如果需要每 100 个时钟周期产生一次系统定时中断，则应向 RELOAD 位域写入 99。

请注意，为了正确访问此寄存器，系统时钟速度必须超过 8 MHz。

SysTick 重载值寄存器 (STRELOAD)

基址 0xE000.E000
偏移量 0x014
类型 R/W, 复位 -



位/域	名称	类型	复位	描述
31:24	保留	RO	0x00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
23:0	RELOAD	R/W	0x00.0000	重载值 当计数器计数到 0 时，本位域的内容将自动载入到系统定时器当前值 (STCURRENT) 寄存器中。

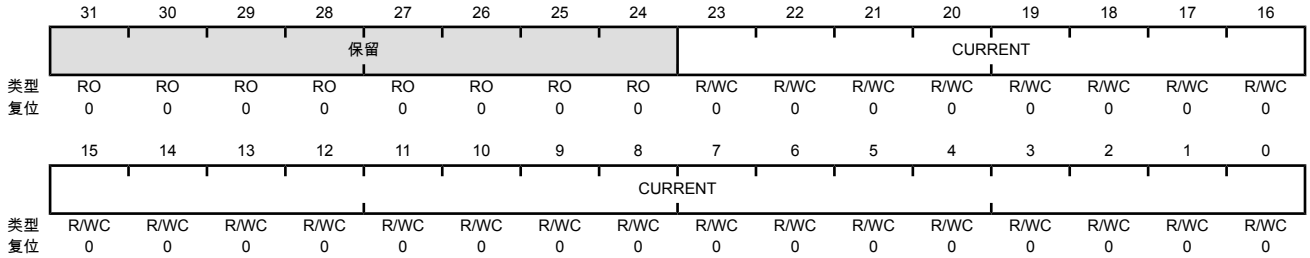
寄存器 3: SysTick 当前值寄存器（STCURRENT），偏移量 0x018

注意: 本寄存器只能在特权模式下访问。

STCURRENT 寄存器中包含 SysTick 计数器的当前值。

SysTick 当前值寄存器 (STCURRENT)

基址 0xE000.E000
偏移量 0x018
类型 R/WC, 复位 -



位/域	名称	类型	复位	描述
31:24	保留	RO	0x00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
23:0	CURRENT	R/WC	0x00.0000	当前值 该域包含访问寄存器时的当前值。没有提供读-修改-写保护，所以在更改时要特别注意。 该寄存器是写清零。向该寄存器写入任意值都会将寄存器清零。清空此寄存器还会清零 STCTRL 寄存器的 COUNT 位。

3.4 NVIC寄存器描述

本节按照地址偏移量由小到大的顺序依次详细介绍NVIC的各寄存器。

在特权模式下才可以完全访问 NVIC 寄存器。在非特权模式下，允许通过配置及控制寄存器（CFGCTRL）屏蔽中断；对其它NVIC寄存器的访问都会导致产生总线故障。

软件访问寄存器时应确保其对齐正确。处理器不支持对NVIC寄存器执行非对齐访问。

即使某个中断已被禁用，该中断也能够进入挂起状态。

在对 VTABLE 寄存器编程（重定位向量表）之前，应确保新向量表中已经包含正确的故障处理程序入口、NMI 处理程序入口以及所有已经使能的异常（例如中断）的入口。详见 140页。

寄存器 4: 中断 0-31 置位启用寄存器（EN0），偏移量 0x100

寄存器 5: 中断 32-63 置位启用寄存器（EN1），偏移量 0x104

寄存器 6: 中断 64-95 置位启用寄存器（EN2），偏移量 0x108

寄存器 7: 中断 96-127 置位启用寄存器（EN3），偏移量 0x10C

注意: 本寄存器只能在特权模式下访问。

ENn 寄存器启用中断并显示启用了哪些中断。EN0 的第 0 位对应于中断 0；第 31 位对应于中断 31。EN1 的第 0 位对应于中断 32；第 31 位对应于中断 63。EN2 的第 0 位对应于中断 64；第 31 位对应于中断 95。EN3 的第 0 位对应于中断 96；第 31 位对应于中断 127。EN4 的第 0 位（参见 120页）对应于中断 128；第 10 位对应于中断 138。

请参阅表2-9（85页）以了解中断分配。

假如某个已挂起的中断被启用，NVIC将按照其优先级激活中断。假如某个中断未启用，那么当中断源产生中断信号后，虽然该中断状态变为挂起，但NVIC永远不会按照其优先级将其激活。

中断 0-31 置位启用寄存器 (EN0)

基址 0xE000.E000
偏移量 0x100
类型 R/W, 复位 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	INT															
类型	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	INT															
类型	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位/域	名称	类型	复位	描述
31:0	INT	R/W	0x0000.0000	中断启用

值	描述
0	读出为0，代表该中断已被禁用。 写入0时，无意义。
1	读出为1，代表该中断已经启用。 写入1时，将启用对应中断。

只能通过在 DISn 寄存器中设置相应的 INT[n] 位来清除某个位。

寄存器 8: 中断 128-138 设置启用寄存器（EN4），偏移量 0x110

注意: 本寄存器只能在特权模式下访问。

EN4 寄存器启用中断并显示启用了哪些中断。0 位对应中断 128；10 位对应中断 138。请参阅表 2-9（85页）以了解中断分配。

假如某个已挂起的中断被启用，NVIC将按照其优先级激活中断。假如某个中断未启用，那么当中断源产生中断信号后，虽然该中断状态变为挂起，但NVIC永远不会按照其优先级将其激活。

中断 128-138 设置启用寄存器 (EN4)

基址 0xE000.E000
偏移量 0x110
类型 R/W, 复位 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	保留															
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	保留					INT										
类型	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位/域	名称	类型	复位	描述
31:11	保留	RO	0x0000.000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
10:0	INT	R/W	0x0	中断启用

值	描述
0	读出为0，代表该中断已被禁用。 写入0时，无意义。
1	读出为1，代表该中断已经启用。 写入1时，将启用对应中断。

只能通过在 DIS4 寄存器中设置相应的 INT[n] 位来清除某个位。

寄存器 9: 中断 0-31 清除启用寄存器（DIS0），偏移量 0x180

寄存器 10: 中断 32-63 清除启用寄存器（DIS1），偏移量 0x184

寄存器 11: 中断 64-95 清除启用寄存器（DIS2），偏移量 0x188

寄存器 12: 中断 96-127 清除启用寄存器（DIS3），偏移量 0x18C

注意: 本寄存器只能在特权模式下访问。

DISn 寄存器禁用中断。DIS0 的第 0 位对应于中断 0；第 31 位对应于中断 31。DIS1 的第 0 位对应于中断 32；第 31 位对应于中断 63。DIS2 的第 0 位对应于中断 64；第 31 位对应于中断 95。DIS3 的第 0 位对应于中断 96；第 31 位对应于中断 127。DIS4 的第 0 位（参见122页）对应于中断 128；第 10 位对应于中断 138。

请参阅表2-9（85页）以了解中断分配。

中断 0-31 清除启用寄存器 (DIS0)

基址 0xE000.E000
偏移量 0x180
类型 R/W, 复位 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	INT															
类型	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	INT															
类型	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位/域	名称	类型	复位	描述
31:0	INT	R/W	0x0000.0000	中断禁用

值 描述

0 读出为0，代表该中断已被禁用。
写入0时，无意义。

1 读出为1，代表该中断已经启用。
写入时，清除 EN0 寄存器中相应的 INT[n] 位即可禁用中断 [n]。

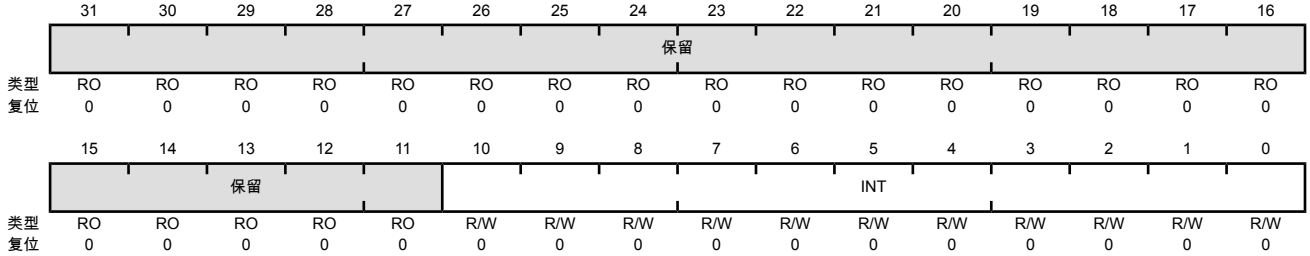
寄存器 13: 中断 128-138 清除启用寄存器（DIS4），偏移量 0x190

注意: 本寄存器只能在特权模式下访问。

DIS4 寄存器禁用中断。0 位对应中断 128；10 位对应中断 138。请参阅表2-9（85页）以了解中断分配。

中断 128-138 清除启用寄存器 (DIS4)

基址 0xE000.E000
偏移量 0x190
类型 R/W, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:11	保留	RO	0x0000.000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
10:0	INT	R/W	0x0	中断禁用

值 描述

0 读出为0，代表该中断已被禁用。
写入0时，无意义。

1 读出为1，代表该中断已经启用。
写入时，清除 EN4 寄存器中相应的 INT[n] 位即可禁用中断 [n]。

寄存器 14: 中断 0-31 置位挂起寄存器（PEND0），偏移量 0x200

寄存器 15: 中断 32-63 置位挂起寄存器（PEND1），偏移量 0x204

寄存器 16: 中断 64-95 置位挂起寄存器（PEND2），偏移量 0x208

寄存器 17: 中断 96-127 置位挂起寄存器（PEND3），偏移量 0x20C

注意: 本寄存器只能在特权模式下访问。

PENDn 寄存器强制将中断置为挂起状态，并显示哪些中断处于挂起状态。PEND0 的第 0 位对应于中断 0；第 31 位对应于中断 31。PEND1 的第 0 位对应于中断 32；第 31 位对应于中断 63。PEND2 的第 0 位对应于中断 64；第 31 位对应于中断 95。PEND3 的第 0 位对应于中断 96；第 31 位对应于中断 127。PEND4 的第 0 位（参见124页）对应于中断 128；第 10 位对应于中断 138。

请参阅表2-9（85页）以了解中断分配。

中断 0-31 置位挂起寄存器 (PEND0)

基址 0xE000.E000
偏移量 0x200
类型 R/W, 复位 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	INT															
类型	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	INT															
类型	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位/域	名称	类型	复位	描述
31:0	INT	R/W	0x0000.0000	中断置位暂挂

值	描述
0	读出为0，代表该中断未挂起。 写入0时，无意义。
1	读出为1，代表该中断已挂起。 写入1时，对应中断的状态将变为挂起，即使该中断已被禁用也是如此。

如果对应的中断已经暂挂，那么该位置位将没有效果。

要想将本寄存器中的某个位清零，需将 UNPEND0 寄存器的相应 INT[n] 位置位。

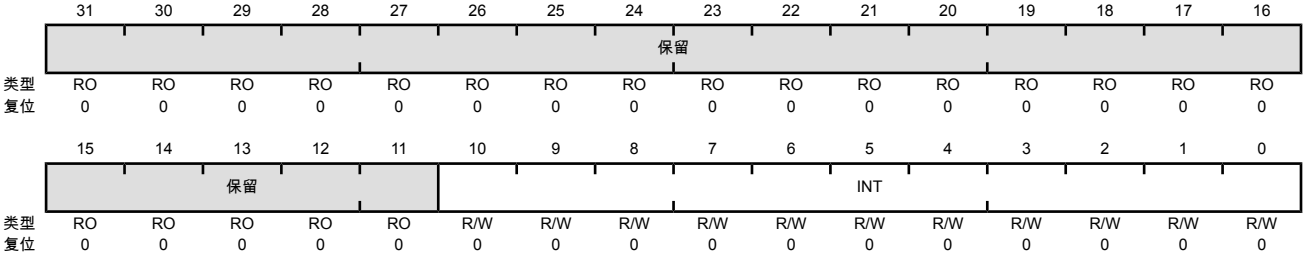
寄存器 18: 中断 128-138 置位挂起寄存器（PEND4），偏移量 0x210

注意: 本寄存器只能在特权模式下访问。

PEND4 寄存器强制将中断置为暂挂状态并显示哪些中断处于暂挂状态。0 位对应中断 128；10 位对应中断 138。请参阅表2-9（85页）以了解中断分配。

中断 128-138 置位挂起寄存器 (PEND4)

基址 0xE000.E000
偏移量 0x210
类型 R/W, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:11	保留	RO	0x0000.000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
10:0	INT	R/W	0x0	中断置位暂挂

值	描述
0	读出为0，代表该中断未挂起。 写入0时，无意义。
1	读出为1，代表该中断已挂起。 写入1时，对应中断的状态将变为挂起，即使该中断已被禁用也是如此。

如果对应的中断已经暂挂，那么该位置位将没有效果。

只能将 UNPEND4 寄存器中的相应 INT[n] 位置位来清除该位。

寄存器 19: 中断 0-31 清除挂起寄存器（UNPEND0），偏移量 0x280

寄存器 20: 中断 32-63 清除挂起寄存器（UNPEND1），偏移量 0x284

寄存器 21: 中断 64-95 清除挂起寄存器（UNPEND2），偏移量 0x288

寄存器 22: 中断 96-127 清除挂起寄存器（UNPEND3），偏移量 0x28C

注意: 本寄存器只能在特权模式下访问。

UNPENDn 寄存器显示哪些中断处于暂挂状态并从中断除去暂挂状态。UNPEND0 的第 0 位对应于中断 0；第 31 位对应于中断 31。UNPEND1 的第 0 位对应于中断 32；第 31 位对应于中断 63。UNPEND2 的第 0 位对应于中断 64；第 31 位对应于中断 95。UNPEND3 的第 0 位对应于中断 96；第 31 位对应于中断 127。UNPEND4 的第 0 位（参见126页）对应于中断 128；第 10 位对应于中断 138。

请参阅表2-9（85页）以了解中断分配。

中断 0-31 清除挂起寄存器 (UNPEND0)

基址 0xE000.E000
偏移量 0x280
类型 R/W, 复位 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	INT															
类型	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	INT															
类型	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位/域	名称	类型	复位	描述
31:0	INT	R/W	0x0000.0000	中断清除暂挂

值 描述

0 读出为0，代表该中断未挂起。
写入0时，无意义。

1 读出为1，代表该中断已挂起。
写入时，清除 PEND0 寄存器中的相应 INT[n] 位，以便使中断 [n] 不再处于暂挂状态
如果某个中断已激活，那么写1将失去实际意义。

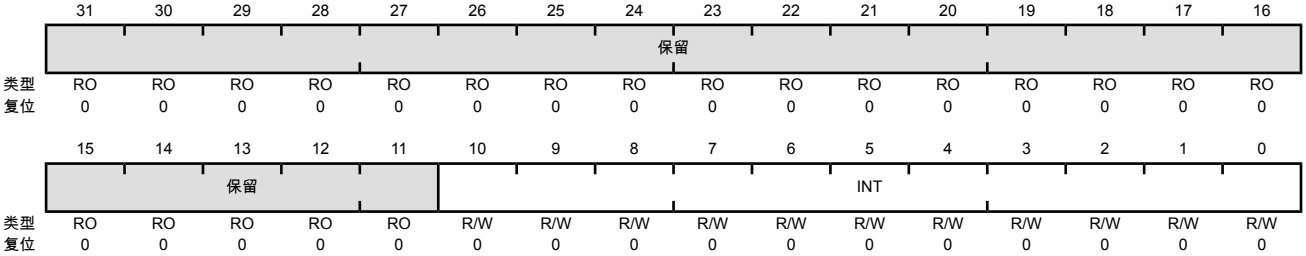
寄存器 23: 中断 128-138 清除挂起寄存器（UNPEND4），偏移量 0x290

注意: 本寄存器只能在特权模式下访问。

UNPEND4 寄存器显示哪些中断处于暂挂状态并清除中断的暂挂状态。0 位对应中断 128；10 位对应中断 138。请参阅表2-9（85页）以了解中断分配。

中断 128-138 清除挂起寄存器 (UNPEND4)

基址 0xE000.E000
偏移量 0x290
类型 R/W, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:11	保留	RO	0x0000.000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
10:0	INT	R/W	0x0	中断清除暂挂 值 描述 0 读出为0，代表该中断未挂起。 写入0时，无意义。 1 读出为1，代表该中断已挂起。 写入时，清除 PEND4 寄存器中的相应 INT[n] 位，以便使中断 [n] 不再处于暂挂状态 如果某个中断已激活，那么写1将失去实际意义。

寄存器 24: 中断 0-31 活动位寄存器（ACTIVE0），偏移量 0x300

寄存器 25: 中断 32-63 活动位寄存器（ACTIVE1），偏移量 0x304

寄存器 26: 中断 64-95 活动位寄存器（ACTIVE2），偏移量 0x308

寄存器 27: 中断 96-127 活动位寄存器（ACTIVE3），偏移量 0x30C

注意: 本寄存器只能在特权模式下访问。

UNPENDn 寄存器指示哪些中断处于激活状态。ACTIVE0 的第 0 位对应于中断 0；第 31 位对应于中断 31。ACTIVE1 的第 0 位对应于中断 32；第 31 位对应于中断 63。ACTIVE2 的第 0 位对应于中断 64；第 31 位对应于中断 95。ACTIVE3 的第 0 位对应于中断 96；第 31 位对应于中断 127。ACTIVE4 的第 0 位（参见128页）对应于中断 128；第 10 位对应于中断 138。

请参阅表2-9（85页）以了解中断分配。

小心 – 切勿手动将本寄存器中的位置位或清零。

中断 0-31 活动位寄存器 (ACTIVE0)

基址 0xE000.E000
偏移量 0x300
类型 RO, 复位 0x0000.0000

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	INT															
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	INT															
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位/域	名称	类型	复位	描述
31:0	INT	RO	0x0000.0000	中断激活

值 描述

0 对应中断未激活。

1 对应中断已激活，或处于激活并等待状态。

寄存器 28: 中断 128-138 活动位寄存器（ACTIVE4），偏移量 0x310

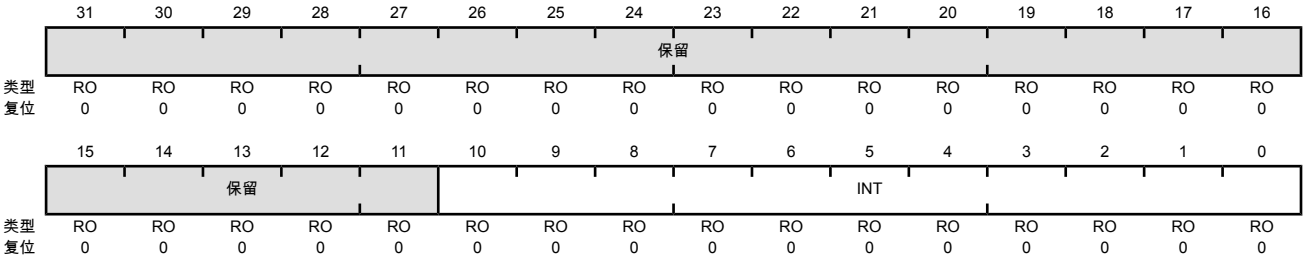
注意: 本寄存器只能在特权模式下访问。

ACTIVE4 寄存器指示哪些中断处于激活状态。0 位对应中断 128；10 位对应中断 131。请参阅表 2-9（85页）以了解中断分配。

小心 – 切勿手动将本寄存器中的位置位或清零。

中断 128-138 活动位寄存器 (ACTIVE4)

基址 0xE000.E000
偏移量 0x310
类型 RO, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:11	保留	RO	0x0000.000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
10:0	INT	RO	0x0	中断激活 值 描述 0 对应中断未激活。 1 对应中断已激活，或处于激活并等待状态。

寄存器 29: 中断 0-3 优先级寄存器（PRI0），偏移量 0x400

寄存器 30: 中断 4-7 优先级寄存器（PRI1），偏移量 0x404

寄存器 31: 中断 8-11 优先级寄存器（PRI2），偏移量 0x408

寄存器 32: 中断 12-15 优先级寄存器（PRI3），偏移量 0x40C

寄存器 33: 中断 16-19 优先级寄存器（PRI4），偏移量 0x410

寄存器 34: 中断 20-23 优先级寄存器（PRI5），偏移量 0x414

寄存器 35: 中断 24-27 优先级寄存器（PRI6），偏移量 0x418

寄存器 36: 中断 28-31 优先级寄存器（PRI7），偏移量 0x41C

寄存器 37: 中断 32-35 优先级寄存器（PRI8），偏移量 0x420

寄存器 38: 中断 36-39 优先级寄存器（PRI9），偏移量 0x424

寄存器 39: 中断 40-43 优先级寄存器（PRI10），偏移量 0x428

寄存器 40: 中断 44-47 优先级寄存器（PRI11），偏移量 0x42C

寄存器 41: 中断 48-51 优先级寄存器（PRI12），偏移量 0x430

寄存器 42: 中断 52-55 优先级寄存器（PRI13），偏移量 0x434

寄存器 43: 中断 56-59 优先级寄存器（PRI14），偏移量 0x438

寄存器 44: 中断 60-63 优先级寄存器（PRI15），偏移量 0x43C

注意: 本寄存器只能在特权模式下访问。

PRIn 寄存器（另请参阅 131页）为每个中断提供 3 位的优先级域。所有PRIn寄存器都是可字节访问的。每个寄存器中包含4个中断的优先级位域，如下所示：

PRIn 寄存器位域	中断
位 31:29	中断 [4n+3]
位 23:21	中断 [4n+2]
位 15:13	中断 [4n+1]
位 7:5	中断 [4n]

请参阅表2-9（85页）以了解中断分配。

优先级位域还可以进一步划分为组优先级位域以及子优先级位域。应用程序中断及复位控制 (APINT) 寄存器中的 PRIGROUP 域（请参阅 141页）指示划分优先级和亚优先级域的二进制点的位置。

这些寄存器只能在特权模式下访问。

中断 0-3 优先级寄存器 (PRI0)

基址 0xE000.E000
偏移量 0x400
类型 R/W, 复位 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	INTD			保留					INTC			保留				
类型	R/W	R/W	R/W	RO	RO	RO	RO	RO	R/W	R/W	R/W	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	INTB			保留					INTA			保留				
类型	R/W	R/W	R/W	RO	RO	RO	RO	RO	R/W	R/W	R/W	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位/域	名称	类型	复位	描述
31:29	INTD	R/W	0x0	中断[4n+3]的中断优先级 此域为编号 [4n+3] 的中断保留优先级值 0-7，其中 n 是中断优先级寄存器的编号（n=0 表示 PRI0，依此类推）。该值越低，相应中断的优先级越高。
28:24	保留	RO	0x0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
23:21	INTC	R/W	0x0	中断[4n+2]的中断优先级 此域为编号 [4n+2] 的中断保留优先级值 0-7，其中 n 是中断优先级寄存器的编号（n=0 表示 PRI0，依此类推）。该值越低，相应中断的优先级越高。
20:16	保留	RO	0x0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
15:13	INTB	R/W	0x0	中断[4n+1]的中断优先级 此域为编号 [4n+1] 的中断保留优先级值 0-7，其中 n 是中断优先级寄存器的编号（n=0 表示 PRI0，依此类推）。该值越低，相应中断的优先级越高。
12:8	保留	RO	0x0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
7:5	INTA	R/W	0x0	中断[4n] 的中断优先级 此域为编号 [4n] 的中断保留优先级值 0-7，其中 n 是中断优先级寄存器的编号（n=0 表示 PRI0，依此类推）。该值越低，相应中断的优先级越高。
4:0	保留	RO	0x0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。

寄存器 **45:** 中断 **64-67** 优先级寄存器（**PRI16**），偏移量 **0x440**

寄存器 **46:** 中断 **68-71** 优先级寄存器（**PRI17**），偏移量 **0x444**

寄存器 **47:** 中断 **72-75** 优先级寄存器（**PRI18**），偏移量 **0x448**

寄存器 **48:** 中断 **76-79** 优先级寄存器（**PRI19**），偏移量 **0x44C**

寄存器 **49:** 中断 **80-83** 优先级寄存器（**PRI20**），偏移量 **0x450**

寄存器 **50:** 中断 **84-87** 优先级寄存器（**PRI21**），偏移量 **0x454**

寄存器 **51:** 中断 **88-91** 优先级寄存器（**PRI22**），偏移量 **0x458**

寄存器 **52:** 中断 **92-95** 优先级寄存器（**PRI23**），偏移量 **0x45C**

寄存器 **53:** 中断 **96-99** 优先级寄存器（**PRI24**），偏移量 **0x460**

寄存器 **54:** 中断 **100-103** 优先级寄存器（**PRI25**），偏移量 **0x464**

寄存器 **55:** 中断 **104-107** 优先级寄存器（**PRI26**），偏移量 **0x468**

寄存器 **56:** 中断 **108-111** 优先级寄存器（**PRI27**），偏移量 **0x46C**

寄存器 **57:** 中断 **112-115** 优先级寄存器（**PRI28**），偏移量 **0x470**

寄存器 **58:** 中断 **116-119** 优先级寄存器（**PRI29**），偏移量 **0x474**

寄存器 **59:** 中断 **120-123** 优先级寄存器（**PRI30**），偏移量 **0x478**

寄存器 **60:** 中断 **124-127** 优先级寄存器（**PRI31**），偏移量 **0x47C**

寄存器 **61:** 中断 **128-131** 优先级寄存器（**PRI32**），偏移量 **0x480**

寄存器 **62:** 中断 **132-135** 优先级寄存器（**PRI33**），偏移量 **0x484**

寄存器 **63:** 中断 **136-138** 优先级寄存器（**PRI34**），偏移量 **0x488**

注意: 本寄存器只能在特权模式下访问。

PRIn 寄存器（另请参阅 129页）为每个中断提供 3 位的优先级域。所有PRIn寄存器都是可字节访问的。每个寄存器中包含4个中断的优先级位域，如下所示：

PRIn 寄存器位域	中断
位 31:29	中断 [4n+3]
位 23:21	中断 [4n+2]
位 15:13	中断 [4n+1]
位 7:5	中断 [4n]

请参阅表2-9（85页）以了解中断分配。

优先级位域还可以进一步划分为组优先级位域以及子优先级位域。应用程序中断及复位控制(APINT)寄存器中的 PRIGROUP 域（请参阅 141页）指示划分优先级和亚优先级域的二进制点的位置。

这些寄存器只能在特权模式下访问。

中断 64-67 优先级寄存器 (PRI16)

基址 0xE000.E000
偏移量 0x440
类型 R/W, 复位 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	INTD			保留					INTC			保留				
类型	R/W	R/W	R/W	RO	RO	RO	RO	RO	R/W	R/W	R/W	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	INTB			保留					INTA			保留				
类型	R/W	R/W	R/W	RO	RO	RO	RO	RO	R/W	R/W	R/W	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位/域	名称	类型	复位	描述
31:29	INTD	R/W	0x0	中断[4n+3]的中断优先级 此域为编号 [4n+3] 的中断保留优先级值 0-7，其中 n 是中断优先级寄存器的编号（n=0 表示 PRI0，依此类推）。该值越低，相应中断的优先级越高。
28:24	保留	RO	0x0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
23:21	INTC	R/W	0x0	中断[4n+2]的中断优先级 此域为编号 [4n+2] 的中断保留优先级值 0-7，其中 n 是中断优先级寄存器的编号（n=0 表示 PRI0，依此类推）。该值越低，相应中断的优先级越高。
20:16	保留	RO	0x0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
15:13	INTB	R/W	0x0	中断[4n+1]的中断优先级 此域为编号 [4n+1] 的中断保留优先级值 0-7，其中 n 是中断优先级寄存器的编号（n=0 表示 PRI0，依此类推）。该值越低，相应中断的优先级越高。
12:8	保留	RO	0x0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
7:5	INTA	R/W	0x0	中断[4n] 的中断优先级 此域为编号 [4n] 的中断保留优先级值 0-7，其中 n 是中断优先级寄存器的编号（n=0 表示 PRI0，依此类推）。该值越低，相应中断的优先级越高。
4:0	保留	RO	0x0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。

寄存器 64: 软件触发中断寄存器（SWTRIG），偏移量 0xF00

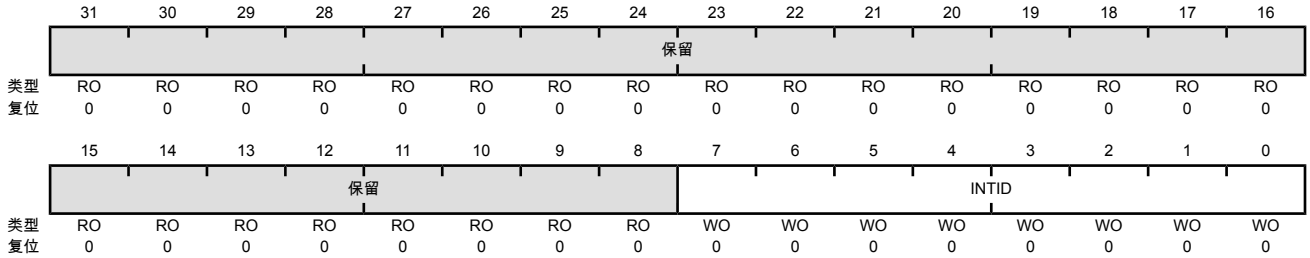
注意: 只有特权软件可以启用对 SWTRIG 寄存器的非特权访问。

向 SWTRIG 寄存器写入中断编号会产生“软件生成的中断” (SGI)。请参阅表2-9（85页）以了解中断分配。

将配置和控制 (CFGCTRL) 中的 MAINPEND 位置位时（请参阅 145页），非特权软件可以访问 SWTRIG 寄存器。

软件触发中断寄存器 (SWTRIG)

基址 0xE000.E000
偏移量 0xF00
类型 WO, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:8	保留	RO	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
7:0	INTID	WO	0x00	中断标识 此域保留所需 SGI 的中断标识。例如，写入值 0x3 会在 IRQ3 上生成中断。

3.5 系统控制模块（SCB）寄存器描述

本节按照地址偏移量由小到大的顺序依次详细介绍系统控制模块（SCB）的各寄存器。SCB寄存器只能在特权模式下访问。

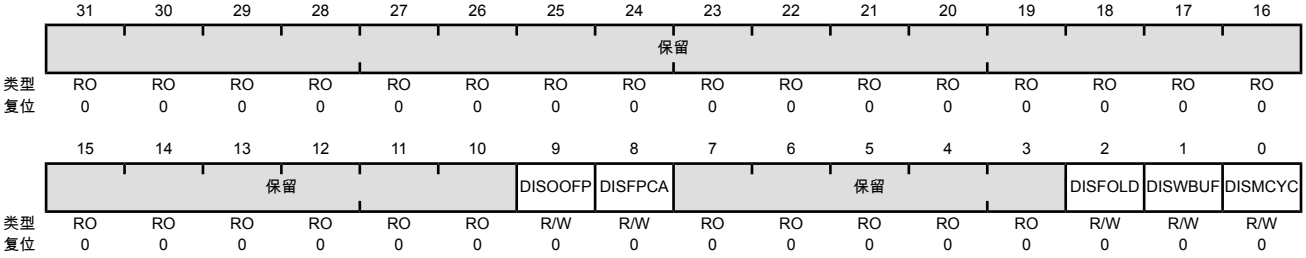
所有寄存器必须以对齐字访问进行访问，除了 FAULTSTAT 和 SYSPRI1-SYSPRI3 寄存器，这两个寄存器可以使用字节或对齐半字或字访问进行访问。处理器不支持对SCB寄存器执行非对齐访问。

寄存器 65: 辅助控制寄存器（ACTLR），偏移量 0x008

注意: 本寄存器只能在特权模式下访问。

ACTLR 寄存器为禁用以下功能提供控制位：IT 堆叠；访问默认存储器映射时的写缓冲；对多周期指令的中断。本寄存器的默认配置已经为 Cortex-M4 处理器内核实现了优化的性能，一般来说无需修改。

辅助控制寄存器 (ACTLR)
基址 0xE000.E000
偏移量 0x008
类型 R/W, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:10	保留	RO	0x00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
9	DISOOFP	R/W	0	禁用无序浮点 禁用相对于整数指令的无序完成的浮点指令。
8	DISFPCA	R/W	0	禁用 CONTROL.FPCA 禁用 CONTROL 寄存器中 FPCA 位的自动更新。 **重要:** 两个位控制何时启用 FPCA：浮点上下文控制 (FPCC) 寄存器中的 ASPEN 位和辅助控制 (ACTLR) 寄存器中的 DISFPCA 位。
7:3	保留	RO	0x00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
2	DISFOLD	R/W	0	禁用IT堆叠。 值 描述 0 没有影响 1 禁用 IT 堆叠。 执行 IT 指令时，在某些情况下，处理器可以继续执行 IT 模块中的第一条指令。这种行为即 *IT 堆叠*，可以提高性能，但是，IT 堆叠会引起循环抖动。如果某项任务禁止产生抖动，那么必须在执行任务之前将 DISFOLD 位置位以禁用 IT 堆叠。

位/域	名称	类型	复位	描述
1	DISWBUF	R/W	0	禁用写入缓冲 值 描述 0 没有影响 1 在访问默认的存储空间时禁用写入缓冲。在此情况下，一旦产生总线故障，则必然是精确总线故障；但处理器性能将会有所下降，因为处理器要等待对内存的存贮操作完全结束后才会执行下一条指令。 注意: 此标志位只影响 Cortex-M4 处理器中执行的写入缓冲。
0	DISMCYC	R/W	0	禁用多周期指令的中断 值 描述 0 没有影响 1 禁止在多周期装载指令以及多周期存贮指令执行期间产生中断。若本标志位置位，由于处理器必须等待 LDM 或 STM 指令执行完成后才将当前状态入栈并进入中断处理程序，因此处理器的中断延时将会增加。

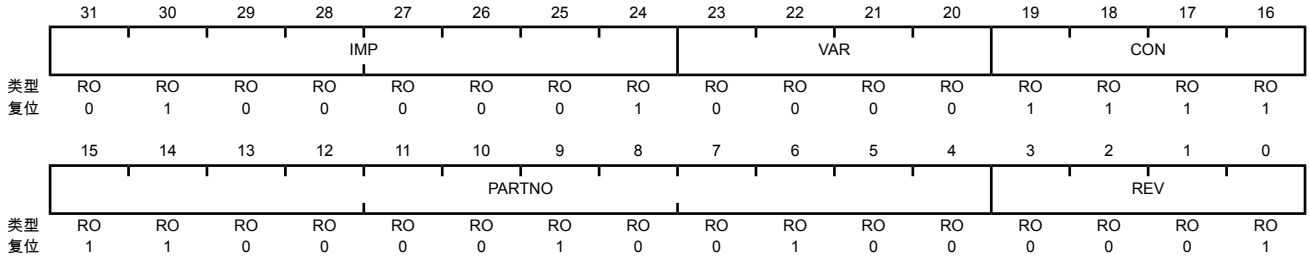
寄存器 66: CPU ID 基础寄存器（CPUID），偏移量 0xD00

注意: 本寄存器只能在特权模式下访问。

CPUID 寄存器中包含 ARM® Cortex™-M4 处理器的芯片编号、版本以及其它构成信息。

CPU ID 基础寄存器 (CPUID)

基址 0xE000.E000
偏移量 0xD00
类型 RO, 复位 0x410F.C241



位/域	名称	类型	复位	描述
31:24	IMP	RO	0x41	标准制订商代码 值 描述 0x41 ARM
23:20	VAR	RO	0x0	修订版本代码 值 描述 0x0 此位域的值即为 rnpn 产品版本识别号中的 rn 值。例如，对于 r0p0 版，此位域的值为 0。
19:16	CON	RO	0xF	常量 值 描述 0xF 此位域始终为0x0F。
15:4	PARTNO	RO	0xC24	器件型号 值 描述 0xC24 Cortex-M4 处理器。
3:0	REV	RO	0x1	版本号 值 描述 0x1 此位域的值即为 rnpn 产品版本识别号中的 pn 值。例如，对于 r0p1 版，此位域的值为 1。

寄存器 67: 中断控制及状态寄存器（INTCTRL），偏移量 0xD04

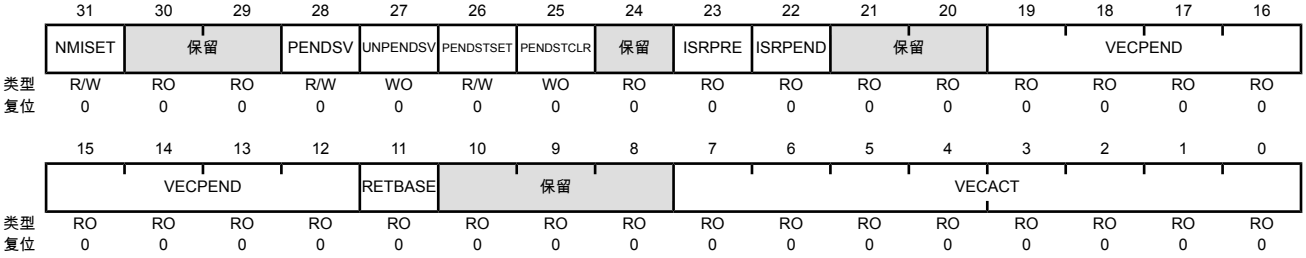
注意: 本寄存器只能在特权模式下访问。

INCTRL 寄存器为 NMI 异常提供了设置挂起位，为 PendSV 以及 SysTick 异常提供了设置挂起位以及清除挂起位。此外，本寄存器还能指示出正在处理的异常序号、是否存在抢占式激活的异常、挂起异常中优先级最高的异常序号，以及是否有中断正在挂起。

若对 INCTRL 寄存器执行写操作时，对 PENDSV 和 UNPENDSV 同时写 1 或对 PENDSTSET 和 PENDSTCLR 同时写 1，可能产生不确定的结果。

中断控制及状态寄存器 (INTCTRL)

基址 0xE000.E000
偏移量 0xD04
类型 R/W, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31	NMISET	R/W	0	NMI设置挂起 值 描述 0 读出为0，代表NMI异常并未挂起。 写入0时，无意义。 1 读出为1，代表NMI异常正在挂起。 写入1时，NMI异常的状态将变为挂起。 由于NMI是最高优先级的异常，因此一般来说，处理器只要检测到此标志置位就会立即进入NMI异常处理程序，并在进入中断处理程序后即刻清除此标志位。只有当处理器执行 NMI 异常处理程序期间又再度产生了 NMI 信号，才可能在异常处理程序读取此标志位时返回 1。
30:29	保留	RO	0x0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
28	PENDSV	R/W	0	PendSV 置位挂起 值 描述 0 读出为0，代表PendSV异常并未挂起。 写入0时，无意义。 1 读出为 1，代表 PendSV 异常挂起。 写入1时，PendSV异常的状态将变为挂起。 将该位置位是把 PendSV 的异常状态设置成挂起的唯一方法。向 UNPENDSV 位写 1 可以将该位清零。

位/域	名称	类型	复位	描述
27	UNPENDSV	WO	0	PendSV清除挂起 值 描述 0 写入0时，无意义。 1 写入 1 时，清除 PendSV 异常的挂起状态。 此标志位为只写。读本寄存器时本标志位的状态不定。
26	PENDSTSET	R/W	0	SysTick设置挂起 值 描述 0 读出为0，代表SysTick异常并未挂起。 写入0时，无意义。 1 读出为 1，代表 SysTick 异常挂起。 写入1时，SysTick异常的状态将变为挂起。 对 PENDSTCLR 位写 1 即可将本标志位清零。
25	PENDSTCLR	WO	0	PendSV 清除挂起 值 描述 0 写入0时，无意义。 1 写入 1 时，清除 SysTick 异常的挂起状态。 此标志位为只写。读本寄存器时本标志位的状态不定。
24	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
23	ISRPRE	RO	0	调试中断处理 值 描述 0 处理器从暂停状态恢复运行并非基于中断。 1 处理器从暂停状态恢复运行是基于中断。 该位只有在调试模式下才有意义，当处理器不是调试模式时，该位读取为零。
22	ISRPEND	RO	0	中断挂起 值 描述 0 无中断正在挂起。 1 有中断正在挂起。 此标志位反映除NMI及故障之外的所有中断状态。
21:20	保留	RO	0x0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。

位/域	名称	类型	复位	描述
19:12	VECPEND	RO	0x00	挂起中断的向量序号 该域包含了最高优先级挂起启用异常的序号。该域的数值与 BASEPRI 和 FAULTMASK 寄存器有关，但是与 PRIMASK 寄存器无关。 值 描述 0x00 无正在挂起的异常 0x01 保留 0x02 NMI 0x03 硬故障 0x04 存储器管理故障 0x05 总线故障 0x06 用法故障 0x07-0x0A 保留 0x0B SVCall 0x0C 保留用于调试 0x0D 保留 0x0E PendSV 0x0F SysTick 0x10 中断向量0 0x11 中断向量1 ... ... 0x9A 中断向量138
11	RETBASE	RO	0	返回基地址 值 描述 0 有其它抢占式激活异常等待处理。 1 当前无激活的异常，或当前正在处理的异常是唯一激活的异常。 此标志位反映除NMI及故障之外的所有中断状态。只有当处理器正在执行某个ISR时（即中断程序状态(IPSR)寄存器非0），该位才有意义。
10:8	保留	RO	0x0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
7:0	VECACT	RO	0x00	挂起中断的向量序号 此位域包含当前激活的异常的序号。各异常的序号参见 VECPEND 域的介绍。假如此位域为0，则处理器将进入线程模式。此位域的值与 IPSR 寄存器的 ISRNUM 位域的值相同。 从该值减去 16 以获取编制索引到中断置位启用 (ENn)、中断清除启用 (DISn)、中断置位挂起 (PENDn)、中断清除挂起 (UNPENDn) 和中断优先级 (PRIn) 寄存器中所需的 IRQ 编号（请参阅 64页）。

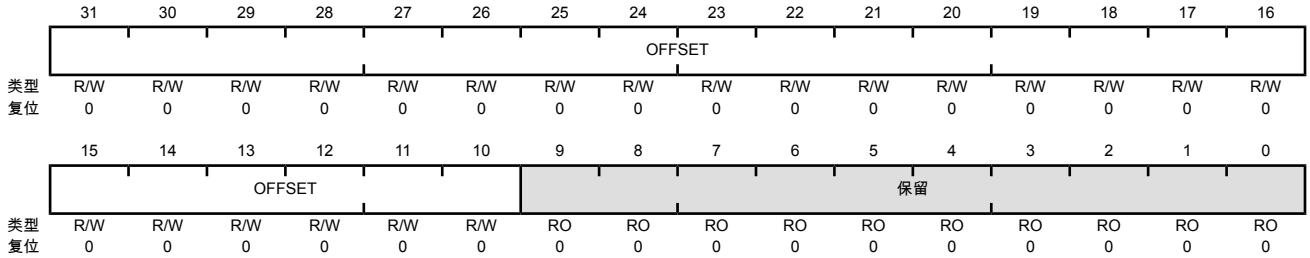
寄存器 68: 向量表寄存器（VTABLE），偏移量 0xD08

注意: 本寄存器只能在特权模式下访问。

VTABLE 寄存器中包含向量表基地址相对于存储器地址 0x0000.0000 的偏移量。

向量表寄存器 (VTABLE)

基址 0xE000.E000
偏移量 0xD08
类型 R/W, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:10	OFFSET	R/W	0x000.00	向量表偏移量 配置 OFFSET 域时，该偏移量必须与向量表中的异常编号对齐。由于有 138 个中断，所以偏移量必须按 1024 字节边界对齐。
9:0	保留	RO	0x00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。

寄存器 69: 应用程序中断及复位控制寄存器（APINT），偏移量 0xD0C

注意: 本寄存器只能在特权模式下访问。

APINT 寄存器为异常模型、数据访问的端状态、系统复位控制提供分组优先级控制。要想对本寄存器执行写操作，必须对 VECTKEY 位域写入 0x05FA，否则写操作将被忽略。

PRIGROUP 域用于设置二进制小数点的位置，该小数点将中断优先级 (PRIx) 寄存器中的的 INTx 域划分为组优先级域以及子优先级域。表3-9（141页）列出了 PRIGROUP 的值与该划分的关系。表中“组优先级位域”以及“子优先级位域”两列的位编号都指针对于 INTA 位域中的位偏移。对于 INTB 域来说，相应的位是 [15:13]；对于 INTC 域来说，相应的位是 [23:21]；对于 INTD 域来说，相应的位是 [31:29]。

注意: 在判定异常的抢占优先级时，只考虑组优先级域。

表 **3-9.** 中断优先级分组

PRIGROUP 位域	二进制小数点[a]	组优先级位域	子优先级位域	组优先级数目	子优先级数目
0x0 - 0x4	bxxx.	[7:5]	无	8	1
0x5	bxx.y	[7:6]	[5]	4	2
0x6	bx.yy	[7]	[6:5]	2	4
0x7	b.yyy	无	[7:5]	1	8

a. 从 INTx 位域可查询二进制小数点的位置。表中x代表分组优先级位，y代表子优先级位。

应用程序中断及复位控制寄存器 (APINT)

基址 0xE000.E000
偏移量 0xD0C
类型 R/W, 复位 0xFA05.0000

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
域	VECTKEY															
类型	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位	1	1	1	1	1	0	1	0	0	0	0	0	0	1	0	1

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
域	ENDIANESS	保留				PRIGROUP			保留					SYSRESREQ	VECTCLRACT	VECTRESET
类型	RO	RO	RO	RO	RO	R/W	R/W	R/W	RO	RO	RO	RO	RO	WO	WO	WO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位/域	名称	类型	复位	描述
31:16	VECTKEY	R/W	0xFA05	注册密钥 该域用来防止对该寄存器的意外写入操作。改变该寄存器的位之前必须将 0x05FA 写入该域。读取返回 0x05FA。
15	ENDIANESS	RO	0	数据端模式 德州仪器 Tiva™ C 系列 仅支持小端模式，因此该位被清零。
14:11	保留	RO	0x0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
10:8	PRIGROUP	R/W	0x0	中断优先级分组 该域确定分组优先级与子优先级的分隔界限（更多信息请参考 表 3-9（141页））。
7:3	保留	RO	0x0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。

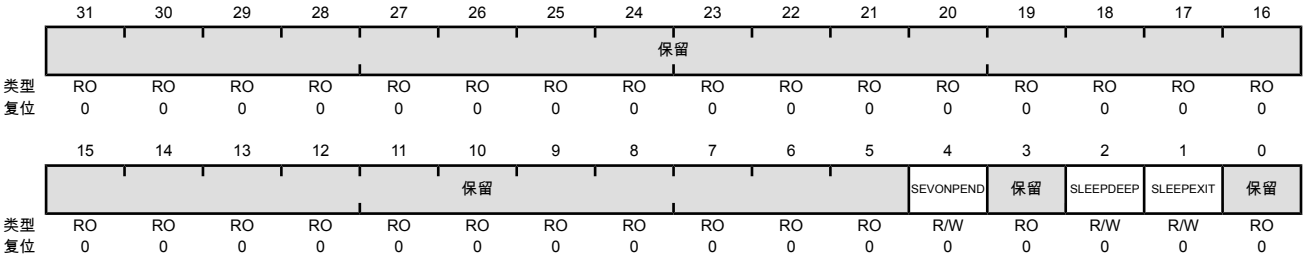
位/域	名称	类型	复位	描述
2	SYSRESREQ	WO	0	系统复位请求 值 描述 0 没有影响 1 复位处理器内核以及所有片上外设（调试接口除外）。 此标志位在处理器内核复位期间自动清零。此标志位的回读值始终为0。
1	VECTCLRACT	WO	0	清除已激活的NMI/故障 该位保留用于调试功能，读取为 0。平时此标志位必须写为0，否则可能产生无法预料的后果。
0	VECTRESET	WO	0	系统复位 该位保留用于调试功能，读取为 0。平时此标志位必须写为0，否则可能产生无法预料的后果。

寄存器 70: 系统控制寄存器（SYSCTRL），偏移量 0xD10

注意: 本寄存器只能在特权模式下访问。

SYSCTRL 寄存器负责控制进入/退出低功耗模式。

系统控制寄存器 (SYSCTRL)
基址 0xE000.E000
偏移量 0xD10
类型 R/W, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:5	保留	RO	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
4	SEVONPEND	R/W	0	挂起时唤醒 值 描述 0 只有已使能的中断或事件才能唤醒处理器；如果某中断已禁用，则不能唤醒处理器。 1 已使能的事件以及所有中断（包括已禁用的中断）都能唤醒处理器。 当一个事件或中断进入挂起状态时，该事件信号会将处理器从 WFE 中唤醒。如果处理器没有等待事件，那么该事件被登记，并会影响下个 WFE。 处理器也可以在执行 SEV 指令时或通过外部事件唤醒。
3	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
2	SLEEPDEEP	R/W	0	深度睡眠使能 值 描述 0 以睡眠模式作为低功耗模式。 1 以深度睡眠模式作为低功耗模式。
1	SLEEPEXIT	R/W	0	在退出ISR后睡眠 值 描述 0 当从处理模式返回到线程模式后，并不进入睡眠状态。 1 当从处理模式返回到线程模式后，一退出ISR则立即进入睡眠或深度睡眠状态。 此功能适用于中断驱动型的应用程序。若此标志位置位，可避免处理器返回到无内容的主函数。

位/域	名称	类型	复位	描述
0	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。

寄存器 71: 配置及控制寄存器（CFGCTRL），偏移量 0xD14

注意: 本寄存器只能在特权模式下访问。

CFGCTRL 寄存器负责控制线程模式的入口，此外还能够启用：NMI 处理程序、硬故障处理程序以及 FAULTMASK 寄存器升级后的故障处理程序忽略总线故障；被零除以及未对齐访问的额外处理；非特权级软件对 SWTRIG 寄存器的访问（见 133页）。

配置及控制寄存器 (CFGCTRL)

基址 0xE000.E000
偏移量 0xD14
类型 R/W, 复位 0x0000.0200

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
域	保留															
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
域	保留						STKALIGN	BFHFNMIGN	保留			DIV0	UNALIGNED	保留	MAINPEND	BASETHR
类型	RO	RO	RO	RO	RO	RO	R/W	R/W	RO	RO	RO	R/W	R/W	RO	R/W	R/W
复位	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0

位/域	名称	类型	复位	描述
31:10	保留	RO	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
9	STKALIGN	R/W	1	异常入口的栈对齐 值 描述 0 栈按4字节对齐。 1 栈按8字节对齐。 在异常进入时，处理器利用压栈 PSR 寄存器的第 9 位保存栈对齐状态。当从异常中返回时，处理器根据出栈后的此标志位恢复栈对齐状态。
8	BFHFNMIGN	R/W	0	在NMI及故障中忽略总线故障 通过此标志位可以使得优先级为-1和-2的异常处理程序忽略由装载指令和存贮指令产生的数据总线故障。此标志位的设置适用于 NMI、硬故障以及经 FAULTMASK 寄存器升级后的处理程序。 值 描述 0 由装载指令及存贮指令产生的数据总线故障都会导致错误锁定。 1 优先级为-1和-2的处理程序均忽略装载指令及存贮指令产生的数据总线故障。 只有当处理程序及其数据均位于绝对安全的存储器中时，才允许将此标志位置位。此标志位通常用于检测系统设备及总线桥有无控制通道隐患，并予以修复。
7:5	保留	RO	0x0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。

位/域	名称	类型	复位	描述
4	DIV0	R/W	0	被零除的额外处理 若处理器执行 SDIV 或 UDIV 指令时遭遇除数为零的状况，本标志位将决定是否产生故障或暂停运行。 值 描述 0 当遇到被零除的状况时，不进行额外处理。被零除返回的商为 0。 1 当遇到被零除的状况时，进行额外处理。
3	UNALIGNED	R/W	0	未对齐访问的额外处理 值 描述 0 当遇到未对齐的半字访问或字访问时，不进行额外处理。 1 当遇到未对齐的半字访问或字访问时，进行额外处理。未对齐的访问会产生一个用法错误。 未对齐的 LDM、STM、LDRD 和 STRD 指令始终产生故障，不论 UNALIGNED 是否置位。
2	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
1	MAINPEND	R/W	0	允许主中断触发 值 描述 0 禁止非特权级软件访问 SWTRIG 寄存器。 1 允许非特权级软件访问 SWTRIG 寄存器（请参考 133页）。
0	BASETHR	R/W	0	线程状态控制 值 描述 0 仅当异常激活后才允许处理器进入线程模式。 1 取决于对 EXC_RETURN 值的控制，处理器随时都可进入线程模式（详情请参考 “异常返回”（90页））。

寄存器 72: 系统处理程序优先级寄存器 1（SYSPRI1），偏移量 0xD18

注意: 本寄存器只能在特权模式下访问。

SYSPRI1 寄存器用于配置用法故障、总线故障、存储器管理故障的异常处理程序的优先级，取值范围为 0~7。本寄存器可以按字节访问。

系统处理程序优先级寄存器 1 (SYSPRI1)

基址 0xE000.E000
偏移量 0xD18
类型 R/W, 复位 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	保留								USAGE			保留				
类型	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	BUS			保留					MEM			保留				
类型	R/W	R/W	R/W	RO	RO	RO	RO	RO	R/W	R/W	R/W	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位/域	名称	类型	复位	描述
31:24	保留	RO	0x00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
23:21	USAGE	R/W	0x0	用法故障优先级 本位域用于配置用法故障的优先级。优先级的取值范围为 0~7，其中数值越小、所代表的优先级越高。
20:16	保留	RO	0x0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
15:13	BUS	R/W	0x0	总线故障优先级 本位域用于配置总线故障的优先级。优先级的取值范围为 0~7，其中数值越小、所代表的优先级越高。
12:8	保留	RO	0x0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
7:5	MEM	R/W	0x0	存储器管理故障优先级 本位域用于配置存储器管理故障的优先级。优先级的取值范围为 0~7，其中数值越小、所代表的优先级越高。
4:0	保留	RO	0x0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。

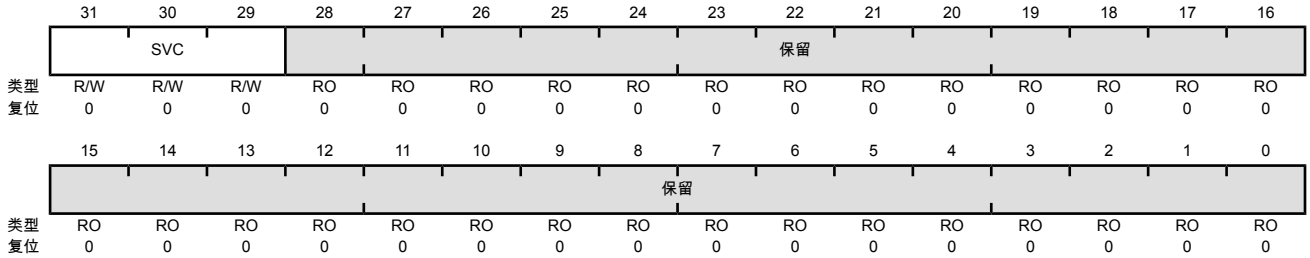
寄存器 73: 系统处理程序优先级寄存器 2（SYSPRI2），偏移量 0xD1C

注意: 本寄存器只能在特权模式下访问。

SYSPRI2 寄存器用于配置 SVCall 处理程序的优先级，取值范围为 0~7。本寄存器可以按字节访问。

系统处理程序优先级寄存器 2 (SYSPRI2)

基址 0xE000.E000
偏移量 0xD1C
类型 R/W, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:29	SVC	R/W	0x0	SVCall优先级 本位域用于配置SVCall的优先级。优先级的取值范围为 0~7，其中数值越小、所代表的优先级越高。
28:0	保留	RO	0x000.0000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。

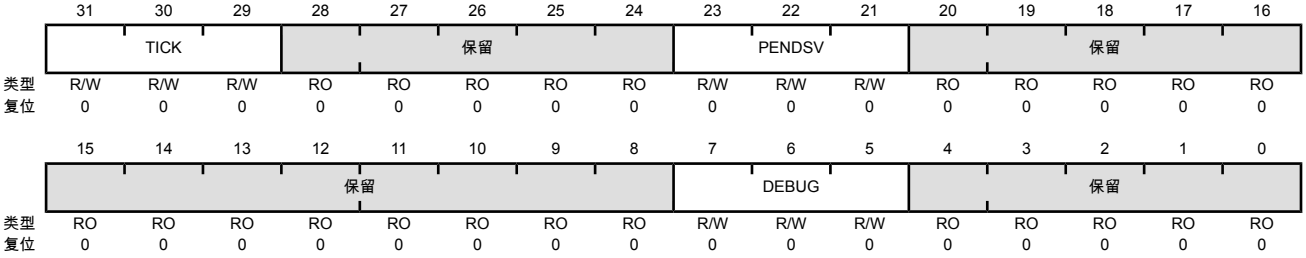
寄存器 74: 系统处理程序优先级寄存器 3（SYSPRI3），偏移量 0xD20

注意: 本寄存器只能在特权模式下访问。

SYSPRI3 寄存器用于配置 SysTick 异常处理程序、PendSV 处理程序的优先级，取值范围为 0~7。本寄存器可以按字节访问。

系统处理程序优先级寄存器 3 (SYSPRI3)

基址 0xE000.E000
偏移量 0xD20
类型 R/W, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:29	TICK	R/W	0x0	SysTick异常优先级 本位域用于配置SysTick异常的优先级。优先级的取值范围为 0~7，其中数值越小、所代表的优先级越高。
28:24	保留	RO	0x0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
23:21	PENDSV	R/W	0x0	PendSV优先级 本位域用于配置PendSV的优先级。优先级的取值范围为 0~7，其中数值越小、所代表的优先级越高。
20:8	保留	RO	0x000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
7:5	DEBUG	R/W	0x0	调试优先级 该域用于配置调试的优先级。优先级的取值范围为 0~7，其中数值越小、所代表的优先级越高。
4:0	保留	RO	0x0.0000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。

寄存器 75: 系统处理程序控制及状态寄存器（SYSHNDCTRL），偏移量 0xD24

注意: 本寄存器只能在特权模式下访问。

SYSHNDCTRL 寄存器用于使能系统处理程序，并且能够指示用法故障、总线故障、存储器管理故障、SVC 异常的挂起状态以及系统处理程序的激活状态。

假如禁用了某个系统处理程序，又产生了相应故障，那么处理器会按照硬故障对其进行处理。

通过修改本寄存器可以改变系统异常的挂起状态或激活状态。操作系统内核可以通过写激活标志位来进行上下文切换，以这种方式更改当前的异常类型。

小心 – 软件在更改本寄存器中某个激活标志位时，如果并未对已入栈内容进行正确的修正，则可能导致处理器产生故障异常。应确保写本寄存器的软件保存当前激活状态，并在之后复原。

使能系统处理程序后，假如必须修改本寄存器中的某个标志位，则必须通过读-修改-写流程予以操作，确保只修改所需的位。

系统处理程序控制及状态寄存器 (SYSHNDCTRL)

基址 0xE000.E000
偏移量 0xD24
类型 R/W, 复位 0x0000.0000

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	保留													USAGE	BUS	MEM
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SVC	BUSP	MEMP	USAGEP	TICK	PNDSV	保留	MON	SVCA	保留			USGA	保留	BUSA	MEMA
类型	R/W	R/W	R/W	R/W	R/W	R/W	RO	R/W	R/W	RO	RO	RO	R/W	RO	R/W	R/W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位/域	名称	类型	复位	描述
31:19	保留	RO	0x000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
18	USAGE	R/W	0	用法故障使能 值 描述 0 禁用用法故障异常。 1 使能用法故障异常。
17	BUS	R/W	0	总线故障使能 值 描述 0 禁用总线故障异常。 1 使能总线故障异常。
16	MEM	R/W	0	存储器管理故障使能 值 描述 0 禁用存储器管理故障异常。 1 使能存储器管理故障异常。

位/域	名称	类型	复位	描述
15	SVC	R/W	0	SVC调用挂起 值 描述 0 当前无正在挂起的SVC调用异常。 1 当前有正在挂起的SVC调用异常。 通过本标志位可修改SVC调用异常的挂起状态。
14	BUSP	R/W	0	总线故障挂起 值 描述 0 当前无正在挂起的总线故障异常。 1 当前有正在挂起的总线故障异常。 通过本标志位可修改总线故障异常的挂起状态。
13	MEMP	R/W	0	存储器管理故障挂起 值 描述 0 当前无正在挂起的存储器管理故障异常。 1 当前有正在挂起的存储器管理故障异常。 通过本标志位可修改存储器管理故障异常的挂起状态。
12	USAGEP	R/W	0	用法故障挂起 值 描述 0 当前无正在挂起的用法故障异常。 1 当前有正在挂起的用法故障异常。 通过本标志位可修改用法故障异常的挂起状态。
11	TICK	R/W	0	SysTick异常激活 值 描述 0 SysTick异常未激活。 1 SysTick异常已激活。 通过本标志位可修改SysTick异常的激活状态，不过在修改本标志位之前，务必仔细阅读本寄存器的“重要提示”部分内容。
10	PNDSV	R/W	0	PendSV异常激活 值 描述 0 PendSV异常未激活。 1 PendSV异常已激活。 通过本标志位可修改 PendSV 异常的激活状态，不过在修改本标志位之前，务必仔细阅读本寄存器的“重要提示”部分内容。
9	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。

位/域	名称	类型	复位	描述
8	MON	R/W	0	调试监视器激活 值 描述 0 调试监视器未激活。 1 调试监视器已激活。
7	SVCA	R/W	0	SVC调用激活 值 描述 0 SVC调用未激活。 1 SVC调用已激活。 通过本标志位可修改SVC调用异常的激活状态，不过在修改本标志位之前，务必仔细阅读本寄存器的“重要提示”部分内容。
6:4	保留	RO	0x0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
3	USGA	R/W	0	用法故障激活 值 描述 0 用法故障未激活。 1 用法故障已激活。 通过本标志位可修改用法故障异常的激活状态，不过在修改本标志位之前，务必仔细阅读本寄存器的“重要提示”部分内容。
2	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
1	BUSA	R/W	0	总线故障激活 值 描述 0 总线故障未激活。 1 总线故障已激活。 通过本标志位可修改总线故障异常的激活状态，不过在修改本标志位之前，务必仔细阅读本寄存器的“重要提示”部分内容。
0	MEMA	R/W	0	存储器管理故障激活 值 描述 0 存储器管理故障未激活。 1 存储器管理故障已激活。 通过本标志位可修改存储器管理故障异常的激活状态，不过在修改本标志位之前，务必仔细阅读本寄存器的“重要提示”部分内容。

寄存器 76: 可配置故障状态寄存器（FAULTSTAT），偏移量 0xD28

注意: 本寄存器只能在特权模式下访问。

FAULTSTAT 寄存器能够指示产生存储器管理故障、总线故障、用法故障的原因。这些功能可以进一步划分为子寄存器，如下所示：

- 用法故障状态子寄存器（UFAULTSTAT），即本寄存器的 [31:16] 位
- 总线故障状态子寄存器（BFAULTSTAT），即本寄存器的 [15:8] 位
- 存储器管理故障状态子寄存器（MFAULTSTAT），即本寄存器的 [7:0] 位

FAULTSTAT 寄存器可按字节访问。FAULTSTAT 寄存器及其子寄存器可按如下方式访问：

- 对偏移量 0xD28 按字访问，即可访问完整的 FAULTSTAT 寄存器
- 对偏移量 0xD28 按字节访问，即可访问 MFAULTSTAT 子寄存器
- 对偏移量 0xD28 按半字访问，即可访问 MFAULTSTAT 与 BFAULTSTAT 子寄存器
- 对偏移量 0xD29 按字节访问，即可访问 BFAULTSTAT 子寄存器
- 对偏移量 0xD2A 按半字访问，即可访问 UFAULTSTAT 子寄存器

本寄存器中的标志位写1即可清零。

在故障处理程序中，发生故障的实际地址可按如下方法确定：

1. 读取并保存存储器管理故障地址 (MMADDR) 或总线故障地址 (FAULTADDR) 的值；
2. 读取 MFAULTSTAT 子寄存器的 MMARV 位，或 BFAULTSTAT 子寄存器的 BFARV 标志位，以确定 MMADDR 或 FAULTADDR 中的内容是否有效。

软件必须按照此序列操作，因为另一个优先级更高的异常有可能会更改 MMADDR 或 FAULTADDR 寄存器的内容。例如，如果某个优先级更高的处理程序抢占了当前的处理程序，那么另一个故障可能会改写 MMADDR 或 FAULTADDR 寄存器的内容。

可配置故障状态寄存器 (FAULTSTAT)

基址 0xE000.E000
偏移量 0xD28
类型 R/W1C, 复位 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	保留						DIV0	UNALIGN	保留				NOCP	INVPC	INVSTAT	UNDEF
类型	RO	RO	RO	RO	RO	RO	R/W1C	R/W1C	RO	RO	RO	RO	R/W1C	R/W1C	R/W1C	R/W1C
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	BFARV	保留	BLSPERR	BSTKE	BUSTKE	IMPRE	PRECISE	IBUS	MMARV	保留	MLSPERR	MSTKE	MUSTKE	保留	DERR	IERR
类型	R/W1C	RO	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C	RO	R/W1C	R/W1C	R/W1C	RO	R/W1C	R/W1C
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位/域	名称	类型	复位	描述
31:26	保留	RO	0x00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。

位/域	名称	类型	复位	描述
25	DIV0	R/W1C	0	被零除用法故障 值 描述 0 未发生被零除故障，或并未使能被零除的额外处理。 1 处理器执行 SDIV 或 UDIV 指令时，除数为 0。 若此标志位置位，原本用于异常返回的已入栈 PC 值将指向执行零除的那条指令。 如果需要对被零除进行额外处理，应将配置和控制 (CFGCTRL) 寄存器（请参考 145页）中的 DIV0 位置位。 此位写 1 清 0。
24	UNALIGN	R/W1C	0	未对齐访问用法故障 值 描述 0 未发生未对齐访问故障，或并未启用未对齐访问的额外处理。 1 处理器企图执行未对齐访问操作。 未对齐的 LDM、STM、LDRD 和 STRD 指令始终产生故障，不论本标志位配置如何。 如果需要对未对齐访问进行额外处理，应将CFGCTRL 寄存器（请参考 145页）中的 UNALIGNED 位置位。 此位写 1 清 0。
23:20	保留	RO	0x00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
19	NOCP	R/W1C	0	无协处理器用法故障 值 描述 0 未发生企图访问协处理器的用法故障。 1 处理器企图访问协处理器。 此位写 1 清 0。
18	INVPC	R/W1C	0	装载无效PC值用法故障 值 描述 0 未发生企图装载无效 PC 值的用法故障。 1 处理器企图执行非法地将 EXC_RETURN 装载到 PC 的操作。这可能是由于上下文无效，或由于 EXC_RETURN 值无效。 若此标志位置位，原本用于异常返回的已入栈 PC 值将指向企图非法装载 PC 值的那条指令。 此位写 1 清 0。

位/域	名称	类型	复位	描述
17	INVSTAT	R/W1C	0	无效状态用法故障 值 描述 0 未发生无效状态的用法故障。 1 处理器执行了某条指令，该指令企图非法使用 EPSR 寄存器。 若此标志位置位，原本用于异常返回的已入栈 PC 值将指向企图非法使用执行程序状态寄存器 (EPSR) 的那条指令。 如果某条未定义的指令使用了 EPSR 寄存器，本标志位不会置位。 此位写 1 清 0。
16	UNDEF	R/W1C	0	未定义指令用法故障 值 描述 0 未发生未定义指令的用法故障。 1 处理器企图执行一条未定义指令。 若此标志位置位，原本用于异常返回的已入栈 PC 值将指向那条未定义指令。 所谓未定义指令，就是指处理器内核无法对其正确解码的指令。 此位写 1 清 0。
15	BFARV	R/W1C	0	总线故障地址寄存器有效 值 描述 0 总线故障地址寄存器 (FAULTADDR) 中未包含有效的故障地址。 1 FAULTADDR 寄存器中包含有效的故障地址。 当出现总线故障后，若故障地址已知，则本标志位将置位。其它故障有可能清零本标志位，例如之后若发生存储器管理故障，即会清零本标志位。 若产生总线故障，并且由于优先级而升级为硬故障，那么硬故障处理程序必须清零本标志位。否则，当程序返回到已入栈的总线故障处理程序时，处理程序以为 FAULTADDR 寄存器仍然有效，但实际上其内容已经被覆盖。 此位写 1 清 0。
14	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
13	BLSPERR	R/W1C	0	浮点怠惰状态保持的总线故障 值 描述 0 在浮点怠惰状态保持时没有发生总线故障。 1 在浮点怠惰状态保持时发生了总线故障。 此位写 1 清 0。

位/域	名称	类型	复位	描述
12	BSTKE	R/W1C	0	入栈总线故障 值 描述 0 未发生入栈操作时的用法故障。 1 当在异常入口处进行入栈操作时产生一个或多个总线故障。 若此标志位置位，SP 寄存器的值仍然会正常变更，但是栈内上下文的内容可能包含错误内容。此类总线故障并不向 FAULTADDR 寄存器写入故障地址。 此位写 1 清 0。
11	BUSTKE	R/W1C	0	出栈总线故障 值 描述 0 未发生出栈操作时的用法故障。 1 当在异常出口处进行出栈操作时产生一个或多个总线故障。 此故障将使得处理器再次返回到异常处理程序。因此，若该位置位，原始的返回栈并不会被破坏。由于前一次返回失败，因此 SP 寄存器的值不变，也不会进行入栈操作。此类总线故障并不向 FAULTADDR 寄存器写入故障地址。 此位写 1 清 0。
10	IMPRE	R/W1C	0	不精确数据总线错误 值 描述 0 描述未产生不精确数据总线错误。 1 产生了数据总线错误，但栈中的返回地址未必指向导致错误产生的那条指令。 若该位置位，不会向 FAULTADDR 寄存器写入故障地址。 此故障是异步故障。因此，当检测到故障发生时，当前进程的优先级高于总线故障优先级，总线故障将保持挂起状态，只有等到处理器完成所有高优先级的进程后，才会转入激活状态。假如在处理器进入处理程序处理不精确总线故障之前，某个精确故障发生，那么处理程序可以查询到 IMPRE 位与某个精确故障状态位同时置位。 此位写 1 清 0。
9	PRECISE	R/W1C	0	精确数据总线错误 值 描述 0 未产生精确数据总线错误。 1 产生了数据总线错误，并且原本用于异常返回的已入栈 PC 值指向导致错误发生的那条指令。 若该位置位，将会向 FAULTADDR 寄存器写入故障地址。 此位写 1 清 0。

位/域	名称	类型	复位	描述
8	IBUS	R/W1C	0	指令总线错误 值 描述 0 未发生指令总线错误。 1 发生了指令总线错误。 处理器在预取指时就已经检测指令总线是否出错，不过只有在试图执行错误指令时才会置位本标志位。 若该位置位，不会向 FAULTADDR 寄存器写入故障地址。 此位写 1 清 0。
7	MMARV	R/W1C	0	存储器管理故障地址寄存器有效 值 描述 0 存储器管理故障地址 (MMADDR) 寄存器中未包含有效的故障地址。 1 MMADDR 寄存器中包含有效的故障地址。 若产生存储器管理故障，并且由于优先级而升级为硬故障，那么硬故障处理程序必须清零本标志位。否则，当程序返回到已入栈的存储器管理故障处理程序时，处理程序以为 MMADDR 寄存器仍然有效，但实际上其内容已经被覆盖。 此位写 1 清 0。
6	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
5	MLSPERR	R/W1C	0	浮点怠惰状态保持的存储器管理故障 值 描述 0 浮点怠惰状态保持时产生了寄存器管理故障。 1 浮点怠惰状态保持时产生了寄存器管理故障。 此位写 1 清 0。
4	MSTKE	R/W1C	0	入栈访问违规 值 描述 0 未发生入栈操作时的访问违规。 1 当在异常入口处进行入栈操作时产生一个或多个访问违规。 若此标志位置位，SP 寄存器的值仍然会正常变更，但是栈内上下文的内容可能包含错误内容。此类故障并不向 MMADDR 寄存器写入故障地址。 此位写 1 清 0。

位/域	名称	类型	复位	描述
3	MUSTKE	R/W1C	0	未发生出栈操作时的访问违规。 值 描述 0 异常返回出栈操作时没有发生存储器管理故障。 1 当在异常出口处进行出栈操作时产生一个或多个访问违规。 此故障将使得处理器再次返回到异常处理程序。因此，若该位置位，原始的返回栈并不会被破坏。由于前一次返回失败，因此 SP 寄存器的值不变，也不会进行入栈操作。此类故障并不向 MMADDR 寄存器写入故障地址。 此位写 1 清 0。
2	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
1	DERR	R/W1C	0	数据访问违规 值 描述 0 未发生数据访问违规。 1 处理器试图对某个地址进行装载或存贮操作，但该地址并不允许执行该操作。 若此标志位置位，原本用于异常返回的已入栈 PC 值将指向出错的那条指令，并且企图访问的地址将写入 MMADDR 寄存器中。 此位写 1 清 0。
0	IERR	R/W1C	0	指令访问违规 值 描述 0 未发生指令访问违规。 1 处理器试图从某个地址取指，但该地址并不允许执行指令。 当对任何 XN 存储器区访问时，均会产生此故障，即使 MPU 被禁用、甚至未集成 MPU 也是如此。 若此标志位置位，原本用于异常返回的已入栈 PC 值将指向出错的那条指令，并且企图访问的地址将写入 MMADDR 寄存器中。 此位写 1 清 0。

寄存器 77: 硬故障状态寄存器（HFAULTSTAT），偏移量 0xD2C

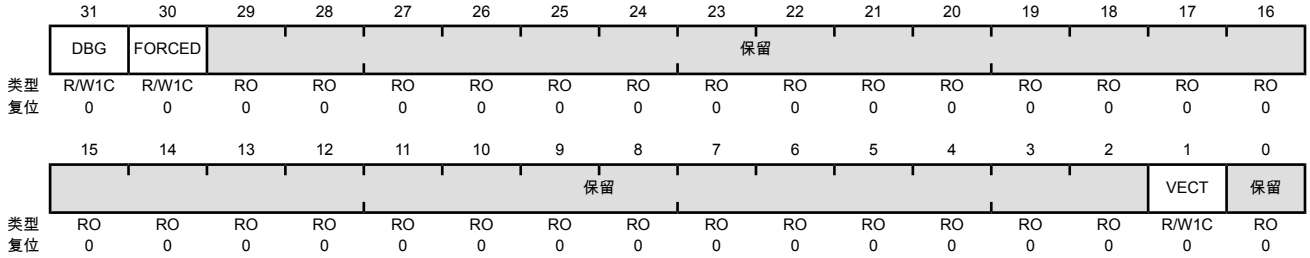
注意: 本寄存器只能在特权模式下访问。

HFAULTSTAT 寄存器提供导致触发硬故障处理程序的事件的相关信息。

本寄存器中的标志位写1即可清零。

硬故障状态寄存器 (HFAULTSTAT)

基址 0xE000.E000
偏移量 0xD2C
类型 R/W1C, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31	DBG	R/W1C	0	调试事件 此标志位保留用于调试。平时此标志位必须写为0，否则可能产生无法预料的后果。
30	FORCED	R/W1C	0	强制硬故障 值 描述 0 未发生强制硬故障。 1 发生了强制硬故障。其原因可能是某个优先级可配置的故障无法得到及时处理（可能是因为优先级较低或被禁用）导致其升级。 若此标志位置位，则硬故障处理程序必须读取其它故障状态寄存器，找出故障原因。 此位写 1 清 0。
29:2	保留	RO	0x00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
1	VECT	R/W1C	0	向量表读取故障 值 描述 0 在读取向量表时未发生总线故障。 1 在读取向量表时发生总线故障。 此错误始终由硬故障处理程序处理。 若此标志位置位，原本用于异常返回的已入栈 PC 值将指向被异常抢占使用权的那条指令。 此位写 1 清 0。
0	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。

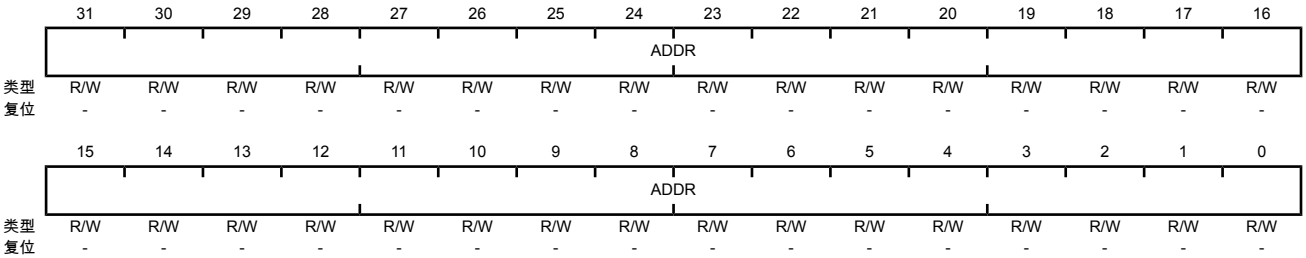
寄存器 78: 存储器管理故障地址寄存器（MMADDR），偏移量 0xD34

注意: 本寄存器只能在特权模式下访问。

MMADDR 寄存器中包含造成存储器管理故障的地址。当发生未对齐访问故障时，MMADDR 寄存器中包含的是产生故障的实际地址。对于其它类型故障，由于单条读写指令还可能进一步分割为多个对齐操作，因此故障地址可能是被请求访问范围内的任一地址。存储器管理故障状态(MFAULTSTAT)中的位能够指示出故障原因以及 MMADDR 寄存器中的值是否有效（请参考 153页）。

存储器管理故障地址寄存器 (MMADDR)

基址 0xE000.E000
偏移量 0xD34
类型 R/W, 复位 -



位/域	名称	类型	复位	描述
31:0	ADDR	R/W	-	故障地址 若 MFAULTSTAT 寄存器的 MMARV 置位，则本位域包含产生存储器管理故障的地址。

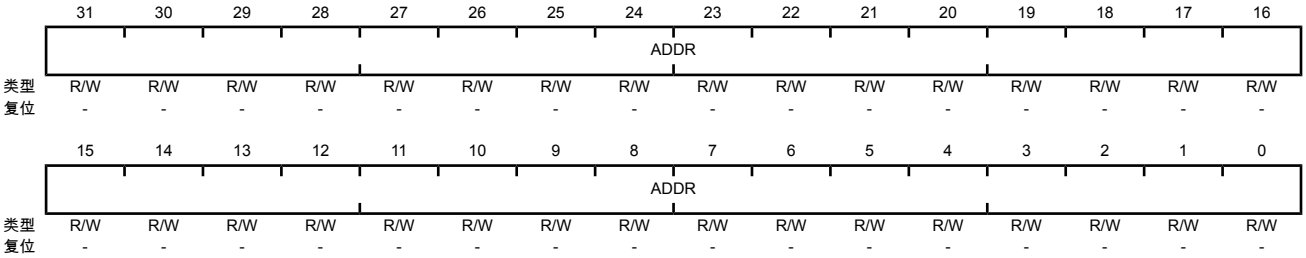
寄存器 79: 总线故障地址寄存器（FAULTADDR），偏移量 0xD38

注意: 本寄存器只能在特权模式下访问。

FAULTADDR 寄存器包含生成总线故障的地址。当发生未对齐访问故障时，FAULTADDR 寄存器中包含的是错误指令所请求的地址（即使其并非产生故障的实际地址）。总线故障状态 (BFAULTSTAT) 寄存器中的状态位能够指示出故障原因以及 FAULTADDR 寄存器中的值是否有效（请参考 153页）。

总线故障地址寄存器 (FAULTADDR)

基址 0xE000.E000
偏移量 0xD38
类型 R/W, 复位 -



位/域	名称	类型	复位	描述
31:0	ADDR	R/W	-	故障地址 若 BFAULTSTAT 寄存器的 FAULTADDRV 置位，则本位域包含产生总线故障的地址。

3.6 存储器保护单元（MPU）寄存器描述

本节按照地址偏移量由小到大的顺序依次详细介绍存储器保护单元（MPU）的各寄存器。

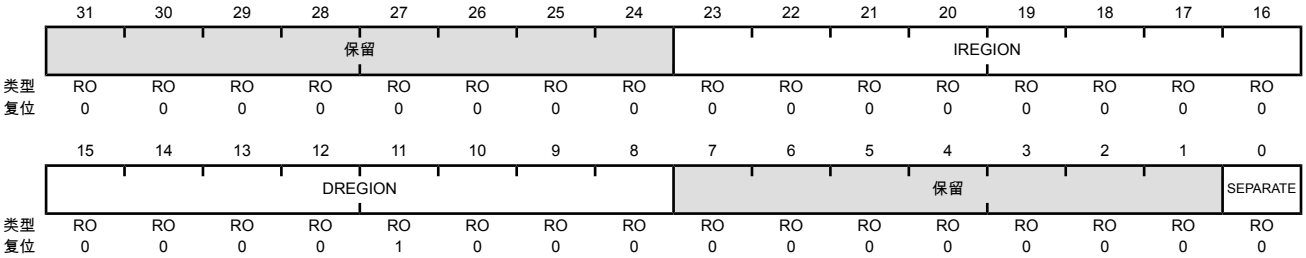
MPU寄存器只能在特权模式下访问。

寄存器 80: MPU 类型寄存器（MPUTYPE），偏移量 0xD90

注意: 本寄存器只能在特权模式下访问。

MPUTYPE 寄存器用于指示本芯片是否包含 MPU，若包含 MPU 还能指示出其支持多少个存储区。

MPU 类型寄存器 (MPUTYPE)
基址 0xE000.E000
偏移量 0xD90
类型 RO, 复位 0x0000.0800



位/域	名称	类型	复位	描述
31:24	保留	RO	0x00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
23:16	IREGION	RO	0x00	I区数目 此位域可显示出芯片所支持的MPU指令区数目。此位域始终为0。MPU存储器映射是统一的，由 DREGION 位域进行描述。
15:8	DREGION	RO	0x08	D区数目 值 描述 0x08 表示总共支持8个MPU数据区。
7:1	保留	RO	0x00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
0	SEPARATE	RO	0	MPU分立还是统一 值 描述 0 表示MPU是统一管理的。

寄存器 81: MPU 控制寄存器（MPUCTRL），偏移量 0xD94

注意: 本寄存器只能在特权模式下访问。

MPUCTRL 寄存器能够使能 MPU、使能默认存储器映射背景区，此外还能选择在硬故障、不可屏蔽中断 (NMI)，以及故障掩码寄存器 (FAULTMASK) 升级后的处理程序中是否允许使用 MPU。

当 ENABLE 与 PRIVDEFEN 同时置位后：

- 对于特权级访问，默认的存储器映射参见“存储模型”（75页）。若某存储器区并未启用，则特权级软件的任何访问特性将与默认的存储器映射相同。
- 若某存储器区并未启用，则非特权级软件的任何访问都将产生存储器管理故障。

不论 ENABLE 位的值如何，从未执行 (XN) 以及严格顺序规则总适用于系统控制空间。

若 ENABLE 位置位，存储器映射中必须至少使能一个存储区，这样才能保障系统工作正常，除非 PRIVDEFEN 位置位。若没有使能的存储区，且 PRIVDEFEN 置位，那么只有特权级软件能正常操作。

若 ENABLE 清零，系统将采用默认的存储器映射，其存储器属性与未采用 MPU 之前的属性相同（请参考 表2-5（78页））。此时无论特权级软件还是非特权级软件均按照默认的存储器映射进行访问。

当使能MPU后，将始终允许对系统控制空间以及向量表的访问。对其它区域的访问权限则取决于区属性设置以及 PRIVDEFEN 是否置位。

除非 HFNMIENA 置位，否则当处理器执行优先级为 -1 或 -2 的异常处理程序时不能启用 MPU。当处理硬故障或者 NMI 异常或者当 FAULTMASK 启用时，这些优先级才可用。若 HFNMIENA 置位，则在这两个优先级下工作时，也将使能 MPU。

MPU 控制寄存器 (MPUCTRL)

基址 0xE000.E000
偏移量 0xD94
类型 R/W, 复位 0x0000.0000

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	保留															
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	保留													PRIVDEFEN	HFNMIENA	ENABLE
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位/域	名称	类型	复位	描述
31:3	保留	RO	0x0000.000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。

位/域	名称	类型	复位	描述
2	PRIVDEFEN	R/W	0	MPU默认区 此标志位用于选择是否允许特权级软件按照默认的存储器映射进行访问。 值 描述 0 若MPU已经使能，此标志位为0代表禁用默认存储器映射。对某个地址进行访问时，若该地址不处于任一启用的区中，则会产生故障。 1 若MPU已经使能，此标志位为1代表允许特权级软件按照默认存储器映射作为背景区进行访问。 若此标志位置位，背景区将作为编号为 -1 的存储器区。任何已定义、已使能的区都将优于背景区。 若MPU未使能，处理器将忽略本标志位。
1	HFNMIENA	R/W	0	故障期间使能MPU 该位控制 MPU 在硬故障、NMI 以及 FAULTMASK 处理程序期间的操作。 值 描述 0 不管 ENABLE 位的状态如何，在硬故障、NMI 以及 FAULTMASK 处理程序期间都将自动禁用 MPU。 1 在硬故障、NMI 以及 FAULTMASK 处理程序期间，仍然启用 MPU。 若此标志位置位，而MPU又没有使能，那么其后果将无法预料。
0	ENABLE	R/W	0	MPU启用 值 描述 0 禁用MPU。 1 使能MPU。 若 HFNMIENA 位置位，而 MPU 又没有使能，那么其后果将无法预料。

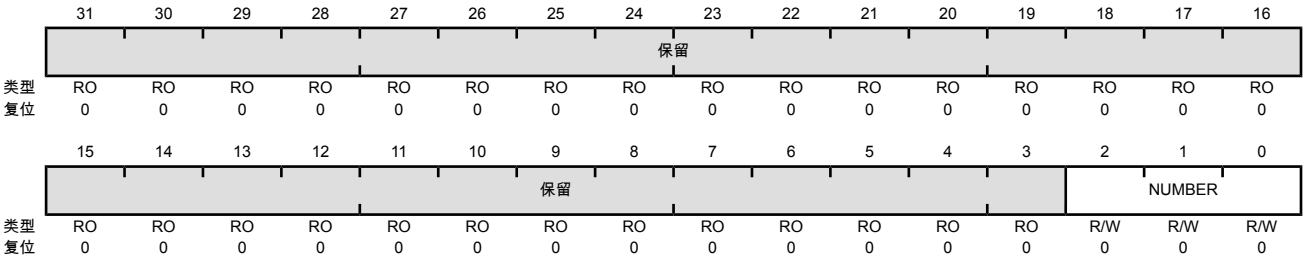
寄存器 82: MPU 区编号寄存器（MPUNUMBER），偏移量 0xD98

注意: 本寄存器只能在特权模式下访问。

MPUNUMBER 寄存器用于选择 MPU 区基地址寄存器 (MPUBASE) 以及 MPU 区属性及大小寄存器 (MPUATTR) 所引用的存储区编号。一般在访问 MPUBASE 及 MPUATTR 寄存器之前，应将需要操作的存储区编号写入本寄存器。但是，该区编号可以通过写入 MPUBASE 寄存器的 VALID 位设置（请参考 166页）来改变。该写入操作会更新 REGION 域的值。

MPU 区编号寄存器 (MPUNUMBER)

基址 0xE000.E000
偏移量 0xD98
类型 R/W, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:3	保留	RO	0x0000.000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
2:0	NUMBER	R/W	0x0	可访问的 MPU 区 该域用于指定 MPUBASE 和 MPUATTR 寄存器参考的 MPU 区。MPU 支持8个存储区。

寄存器 83: MPU 区基地址寄存器（MPUBASE），偏移量 0xD9C

寄存器 84: MPU 区基地址别名寄存器 1（MPUBASE1），偏移量 0xDA4

寄存器 85: MPU 区基地址别名寄存器 2（MPUBASE2），偏移量 0xDAC

寄存器 86: MPU 区基地址别名寄存器 3（MPUBASE3），偏移量 0xDB4

注意: 本寄存器只能在特权模式下访问。

MPUBASE 寄存器用于定义由 MPU 区编号寄存器 (MPUNUMBER) 所指定 MPU 区的基地址，此外也能够直接更新 MPUNUMBER 寄存器的内容。如果想修改当前的存储区编号并连带更新 MPUNUMBER 寄存器内容，应在写 MPUBASE 寄存器时将 VALID 位置位。

ADDR 域是指 MPUBASE 寄存器的 31:*N* 位。(*N*-1):5 位保留。*N* 的值由区大小所决定，而区大小由 MPU 区属性及大小寄存器 (MPUATTR) 的 SIZE 位域决定，其中：

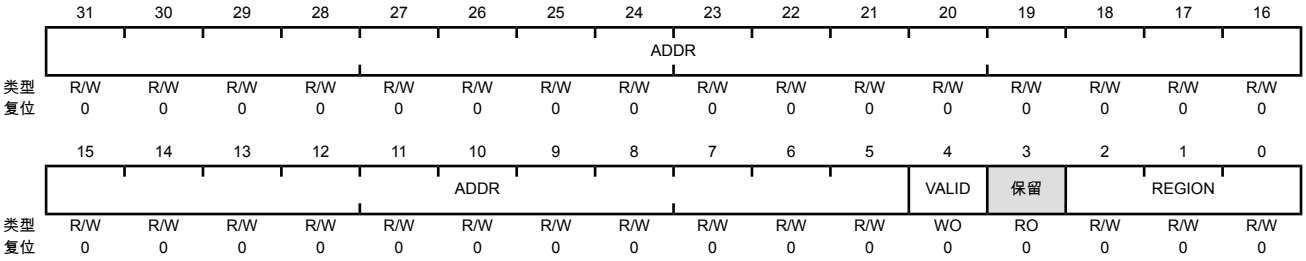
$N = \mathrm{Log}_2$（区大小，以字节为单位）

若在 MPUATTR 寄存器中配置区大小为 4GB，那就没有可用的 ADDR 位域。此时，该存储区将占据处理器的全部寻址空间，其基地址为0x0000 0000。

基地址必须按照存储器区的大小对齐。例如，定义某个区大小为64kB，则其基地址必须按照64kB的倍数对齐，例如可以是0x0001 0000或0x0002 0000。

MPU 区基地址寄存器 (MPUBASE)

基址 0xE000.E000
偏移量 0xD9C
类型 R/W, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:5	ADDR	R/W	0x0000.000	基地址掩码 [31:*N*] 位域包含存储器区的基地址。*N* 的值取决于区大小，如前所述。其余的 (*N*-1):5 位保留。 软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
4	VALID	WO	0	区编号有效 值 描述 0 MPUNUMBER 寄存器不变，处理器按照 MPUNUMBER 寄存器的设置更新存储区的基地址，忽略 REGION 位域的设置。 1 MPUNUMBER 寄存器将更新为 REGION 位域的值，并依此更新 REGION 位域相应存储区的基址。 此标志位的回读值始终为0。
3	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。

位/域	名称	类型	复位	描述
2:0	REGION	R/W	0x0	区编号 写入本位域时，其包含将要写入 MPUNUMBER 寄存器的值。读取本位域时，将返回 MPUNUMBER 寄存器的当前存储区编号。

寄存器 87: MPU 区属性和大小寄存器（MPUATTR），偏移量 0xDA0

寄存器 88: MPU 区属性和大小别名寄存器 1（MPUATTR1），偏移量 0xDA8

寄存器 89: MPU 区属性和大小别名寄存器 2（MPUATTR2），偏移量 0xDB0

寄存器 90: MPU 区属性和大小别名寄存器 3（MPUATTR3），偏移量 0xDB8

注意: 本寄存器只能在特权模式下访问。

MPUATTR 寄存器用于定义由 MPU 区编号寄存器 (MPUNUMBER) 所指定 MPU 区的大小以及存储器属性，并且能够使能该 MPU 区及其子区。

MPUATTR 寄存器可按字或半字对齐访问，其高半字保存区属性，低半字保存区大小、区使能标志以及各子区的使能标志。

MPU 访问权限属性位（XN、AP、TEX、S、C、B）用于控制对相应存储区的访问。假如试图访问某个区域，而该区域并未开放相关权限，那么MPU将会产生一个访问权限故障。

SIZE 位域用于定义由 MPUNUMBER 寄存器所指定的 MPU 存储区的大小，如下所示：

区大小（以字节为单位）= $2^{(SIZE+1)}$

存储区最小允许为 32 字节，对应的 SIZE 位域值为 4。表3-10（168页）列出了 SIZE 域与区大小相关的一些示例值，同时列出了 MPU 区基地址 (MPUBASE) 中对应的 N 值。

表 **3-10. SIZE** 域数值示例

SIZE 位域编码	区大小	**N** 的值[a]	注
00100b (0x4)	32 B	5	允许的最小区大小
01001b (0x9)	1 KB	10	-
10011b (0x13)	1 MB	20	-
11101b (0x1D)	1 GB	30	-
11111b (0x1F)	4 GB	MPUBASE 寄存器中没有有效的 ADDR 域；该区已经占据全部寻址空间。	最大可能的大小

a. 表示 MPUBASE 寄存器（请参考 166页）中的 N 参数。

MPU 区属性和大小寄存器 (MPUATTR)

基址 0xE000.E000
偏移量 0xDA0
类型 R/W, 复位 0x0000.0000

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
域	保留			XN	保留	AP			保留		TEX			S	C	B
类型	RO	RO	RO	R/W	RO	R/W	R/W	R/W	RO	RO	R/W	R/W	R/W	R/W	R/W	R/W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
域	SRD								保留		SIZE					ENABLE
类型	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	RO	RO	R/W	R/W	R/W	R/W	R/W	R/W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位/域	名称	类型	复位	描述
31:29	保留	RO	0x00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。

位/域	名称	类型	复位	描述
28	XN	R/W	0	禁止访问指令 值 描述 0 使能取值操作。 1 禁止取值操作。
27	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
26:24	AP	R/W	0	访问权限 该域的更多使用信息请参考 表3-5（107页）。
23:22	保留	RO	0x0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
21:19	TEX	R/W	0x0	类型扩展掩码 该域的更多使用信息请参考 表3-3（107页）。
18	S	R/W	0	可共享 该位的更多使用信息请参考 表3-3（107页）。
17	C	R/W	0	可高速缓冲性 该位的更多使用信息请参考 表3-3（107页）。
16	B	R/W	0	可缓冲性 该位的更多使用信息请参考 表3-3（107页）。
15:8	SRD	R/W	0x00	子区禁用位 值 描述 0 启用相应子区。 1 禁用相应子区。 存储区小于等于128字节时，将无法再分割为子区。此时写入存储区属性时，应将 SRD 位域配置为 0x00。更多信息参见 “存储子区”一节（106页）。
7:6	保留	RO	0x0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
5:1	SIZE	R/W	0x0	区大小掩码 SIZE 位域用于定义由 MPUNUMBER 寄存器指定的 MPU 存储器区的大小。请参阅 表3-10（168页） 以了解更多信息。
0	ENABLE	R/W	0	区启用 值 描述 0 禁用此存储器区。 1 启用此存储器区。

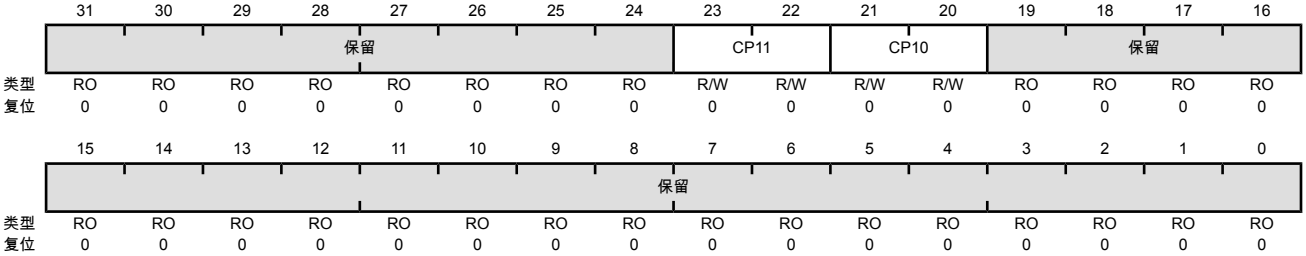
3.7 浮点单元 (FPU) 寄存器描述

本节按照地址偏移量由小到大的顺序，依次罗列并描述浮点单元 (FPU) 的各寄存器。

寄存器 91: 协处理器访问控制（CPAC），偏移量 0xD88

CPAC 寄存器指定协处理器的访问权限。

协处理器访问控制 (CPAC)
基址 0xE000.E000
偏移量 0xD88
类型 R/W, 复位 0x0000.0000

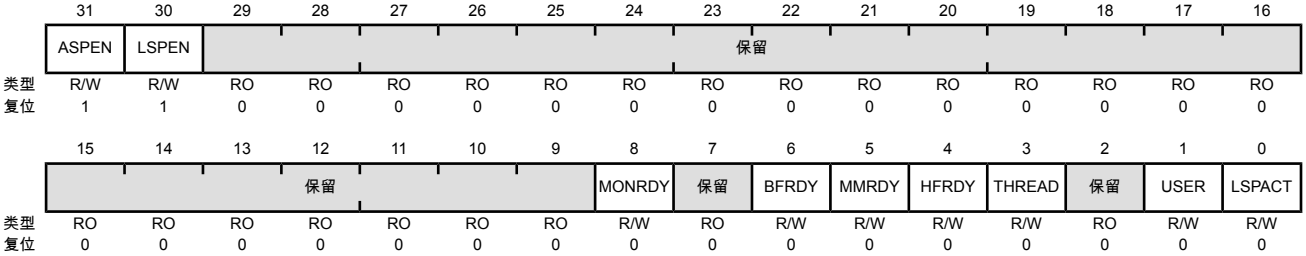


位/域	名称	类型	复位	描述
31:24	保留	RO	0x00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
23:22	CP11	R/W	0x00	CP11 协处理器访问权限 值 描述 0x0 拒绝访问 任何访问尝试都将产生 NOCP 用法故障。 0x1 仅特权级访问 无特权的访问将产生 NOCP 错误。 0x2 保留 任何访问的结果都无法预测。 0x3 完全访问权
21:20	CP10	R/W	0x00	CP10 协处理器访问权限 值 描述 0x0 拒绝访问 任何访问尝试都将产生 NOCP 用法故障。 0x1 仅特权级访问 无特权的访问将产生 NOCP 错误。 0x2 保留 任何访问的结果都无法预测。 0x3 完全访问权
19:0	保留	RO	0x00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。

寄存器 92: 浮点上下文控制（FPCC），偏移量 0xF34

FPCC 寄存器置位或返回 FPU 控制数据。

浮点上下文控制 (FPCC)
基址 0xE000.E000
偏移量 0xF34
类型 R/W, 复位 0xC000.0000



位/域	名称	类型	复位	描述
31	ASPEN	R/W	1	启用自动状态保存 置位后，在执行浮点指令时可使用 CONTROL 寄存器中的 FRACTV 位。在浮点上下文异常进入和退出时，该功能可自动保存和恢复硬件状态。 **重要:** 两个位控制何时启用 FPCA：浮点上下文控制 (FPCC) 寄存器中的 ASPEN 位和辅助控制 (ACTLR) 寄存器中的 DISFPCA 位。
30	LSPEN	R/W	1	启用怠惰状态保存 置位后，启用浮点上下文的怠惰状态保存。
29:9	保留	RO	0x00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
8	MONRDY	R/W	0	监控器就绪 置位后，调试监控器被启用；分配浮点堆栈框时，优先级允许置位 MON_PEND。
7	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
6	BFRDY	R/W	0	总线故障就绪 置位后，总线故障被启用；分配浮点堆栈框时，优先级允许将总线故障处理程序设置为挂起状态。
5	MMRDY	R/W	0	存储器管理故障就绪 置位后，存储器管理被启用；分配浮点堆栈框时，优先级允许将存储器管理处理程序设置为挂起状态。
4	HFRDY	R/W	0	硬故障就绪 置位后，分配浮点堆栈框时，优先级允许将硬故障处理程序设置为挂起状态。
3	THREAD	R/W	0	线程模式 置位后，分配浮点堆栈框时状态为线程模式。
2	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。

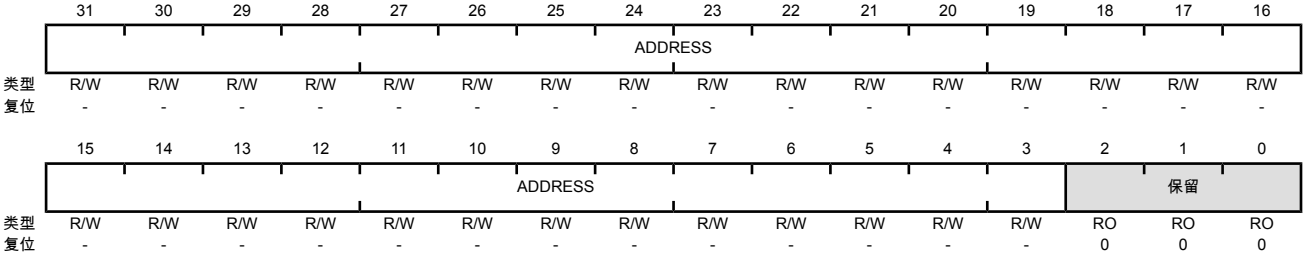
位/域	名称	类型	复位	描述
1	USER	R/W	0	用户特权等级 置位后，分配浮点堆栈框时特权等级为用户。
0	LSPACT	R/W	0	怠惰状态保存有效 置位后，怠惰状态保存有效。浮点堆栈框已分配，但将状态保存到堆栈框出现延迟。

寄存器 93: 浮点上下文访问（FPCA），偏移量 0xF38

FPCA 寄存器包含在异常堆栈框上分配的未被占用的浮点寄存器空间的位置。

浮点上下文访问 (FPCA)

基址 0xE000.E000
偏移量 0xF38
类型 R/W, 复位 -



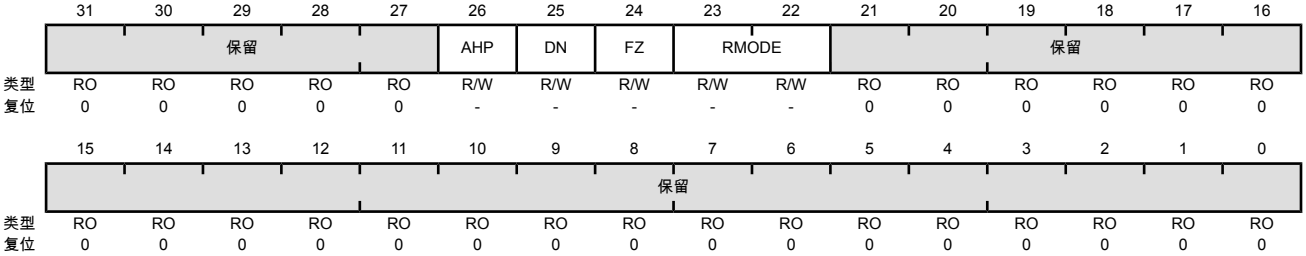
位/域	名称	类型	复位	描述
31:3	ADDRESS	R/W	-	地址 在异常堆栈框上分配的未被占用的浮点寄存器空间的位置。
2:0	保留	RO	0x00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。

寄存器 94: 浮点默认状态控制（FPDSC），偏移量 0xF3C

FPDSC 寄存器包含浮点状态控制 (FPSC) 寄存器的默认值。

浮点默认状态控制 (FPDSC)

基址 0xE000.E000
偏移量 0xF3C
类型 R/W, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:27	保留	RO	0x00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
26	AHP	R/W	-	AHP 位默认值 该位包含 FPSC 寄存器中 AHP 位的默认值。
25	DN	R/W	-	DN 位默认值 该位包含 FPSC 寄存器中 DN 位的默认值。
24	FZ	R/W	-	FZ 位默认值 该位包含 FPSC 寄存器中 FZ 位的默认值。
23:22	RMODE	R/W	-	RMODE 位默认值 该位包含 FPSC 寄存器中 RMODE 位域的默认值。 值 描述 0x0 向最接近值舍入 (RN) 模式 0x1 向正无穷大舍入 (RP) 模式 0x2 向负无穷大舍入 (RM) 模式 0x3 向 0 舍入 (RZ) 模式
21:0	保留	RO	0x00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。

4 JTAG 接口

联合测试行动组 (JTAG) 是一项 IEEE 标准，它定义了数字集成电路的测试访问端口和边界扫描结构，并且提供了一个标准化的串行接口来控制相关的测试逻辑。TAP、指令寄存器 (IR) 和数据寄存器 (DR) 可用来测试组装好的印刷电路板的互连性，并获取元件的制造信息。JTAG 端口还可用于访问和控制测试用设计的特性，如 I/O 管脚的观察和控制、扫描测试以及调试。

JTAG 接口由四个管脚组成：TCK、TMS、TDI 和 TDO。数据串行发送至 TDI 的控制器，然后从 TDO 控制器串行输出。该数据的解析取决于TAP控制器的当前状态。有关 JTAG 端口和 TAP 控制器操作的详细信息，请参考 “IEEE 标准 1149.1-测试访问端口和边界扫描结构”。

TM4C1231H6PM JTAG 控制器与 Cortex-M4F 内核内置的 ARM JTAG 控制器一起工作，这是通过复用这两个 JTAG 控制器的 TDO 输出来实现的。ARM JTAG 指令选择 ARM 的 TDO 输出，而 JTAG 指令选择 TDO 输出。复用器由 JTAG 控制器控制，它可以对 ARM Tiva™ C 系列 控制器和未执行的 JTAG 指令进行全面的编程。

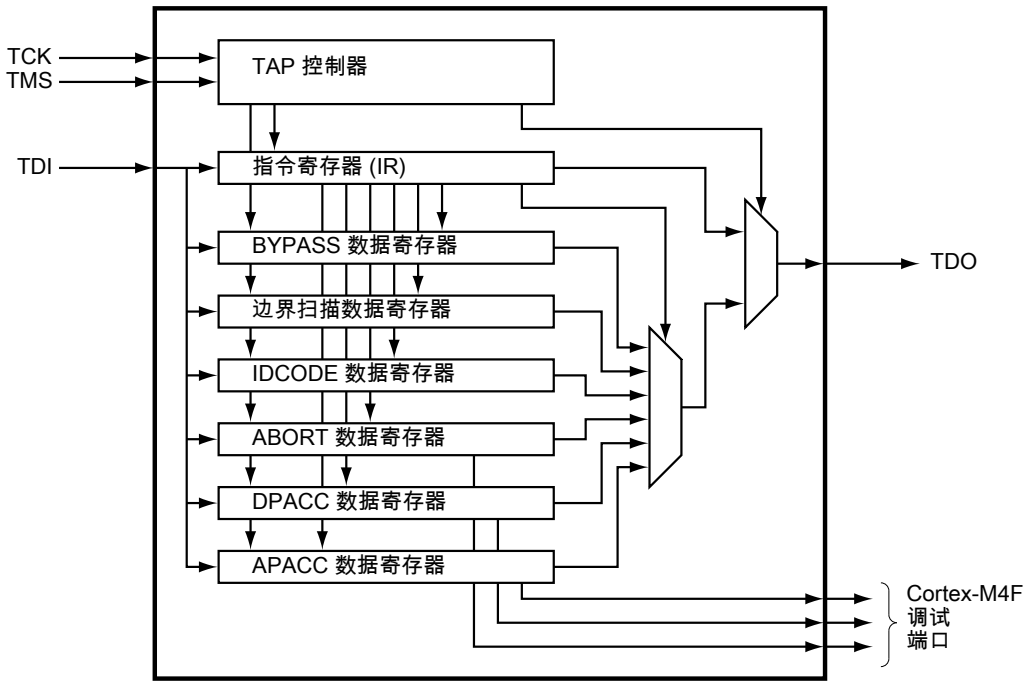
TM4C1231H6PM JTAG 模块具有以下特性：

- IEEE 1149.1-1990 兼容的测试访问端口 (TAP) 控制器
- 4 位指令寄存器 (IR) 链，用于存储 JTAG 指令
- IEEE 标准指令：BYPASS、IDCODE、SAMPLE/PRELOAD 和 EXTEST
- ARM 附加指令：APACC、DPACC 和 ABORT
- 集成的 ARM 串行线调试 (SWD)
  - 串行线 JTAG 调试端口 (SWJ-DP)
  - Flash 修补和断点 (FPB) 单元，用于实现断点操作
  - 数据观察点和触发 (DWT) 单元，用于执行观察点、触发源和系统性能分析
  - 仪表跟踪宏单元 (ITM)，用于支持 printf 型调试
  - 用于指令追踪捕捉的嵌入式追踪宏单元 (ETM)
  - 跟踪端口接口单元 (TPIU) 用作跟踪端口分析仪的桥接

关于 ARM JTAG 控制器的更多信息，可查看“ ARM® Debug Interface V5 Architecture Specification”。

4.1 结构框图

图 **4-1. JTAG** 模块方框图



4.2 信号描述

下表列出了 JTAG/SWD 控制器的外部信号及其功能。JTAG/SWD控制器信号中的某些GPIO信号具有备用功能，但是注意这些管脚的复位状态用于JTAG/SWD功能。JTAG/SWD 控制器信号处于保护状态，要配置成 GPIO，即需要进行特殊处理，请参阅“确认控制”（581页）。表中的“复用管脚/赋值”一栏列出了 JTAG/SWD 控制器信号的 GPIO 管脚布局。将 GPIO 备用功能选择 (GPIOAFSEL) 寄存器（595页）中的 AFSEL 位置位以选择 JTAG/SWD 功能。必须将括号中的数字写入 GPIO 端口控制 (GPIOPCTL) 寄存器（612页）中的 PMCn 域，以便将 JTAG/SWD 信号分配给指定的 GPIO 端口管脚。有关如何配置 GPIO 的更多信息，请参阅“通用输入/输入端口（GPIOs）”（575页）。

表 **4-1. JTAG_SWD_SWO** 信号 **(64LQFP)**

管脚名称	管脚编号	管脚复用/管脚赋值	管脚类型	缓冲区类型[a]	描述
SWCLK	52	PC0 (1)	I	TTL	JTAG/SWD CLK信号。
SWDIO	51	PC1 (1)	I/O	TTL	JTAG TMS及SWDIO信号。
SWO	49	PC3 (1)	O	TTL	JTAG TDO及SWO信号。
TCK	52	PC0 (1)	I	TTL	JTAG/SWD CLK信号。
TDI	50	PC2 (1)	I	TTL	JTAG TDI信号。
TDO	49	PC3 (1)	O	TTL	JTAG TDO及SWO信号。
TMS	51	PC1 (1)	I	TTL	JTAG TMS及SWDIO信号。

a. TTL 表示管脚的电压水平与 TTL 一致。

4.3 功能说明

JTAG 模块的高级概念图如图4-1（176页）所示。JTAG 模块由测试访问端口 (TAP) 控制器和带有并行更新寄存器的串行移位链组成。TAP 控制器是一个简单的状态机，它由 TCK 和 TMS 输入控制。TAP 控制器的当前状态取决于 TMS 管脚在 TCK 信号上升沿所捕获的值的序列。TAP 控制器决定了串行移位链何时捕捉新数据，何时将数据从 TDI 移位到 TDO，以及何时更新并行加载寄存器。TAP控制器的当前状态还决定了正在被访问的是指令寄存器 (IR) 链还是一个数据寄存器 (DR) 链。

带有并行加载寄存器的串行移位链由一个指令寄存器 (IR) 链和多个数据寄存器 (DR) 链组成。加载在并行加载寄存器中的当前指令决定了哪个DR链在TAP控制器排序过程中被捕捉、移位或更新。

某些指令，像 EXTEST，会对当前位于 DR 链中的数据进行操作，但不会捕获、移动或更新任何链。为了确保 TDI 和 TDO 之间的串行通道一直连接，未被执行的指令将会译码成 BYPASS 指令（关于已执行指令的一览表，请参考表4-3（182页））。

有关 JTAG 时序图，请参阅“JTAG and Boundary Scan”（1034页）。

注意: 在所有可能的复位源中，只有上电复位 (POR) 和有效的 RST 输入对 JTAG 模块有影响。管脚配置由 RST 输入和 POR 来复位，但是内部的 JTAG 逻辑只能由 POR 来复位。更多复位信息，请参阅“复位源”（187页）。

4.3.1 JTAG 接口管脚

JTAG 接口由四个标准管脚组成：TCK、TMS、TDI 和 TDO。有关这些管脚的信息，以及它们在上电复位或者 RST 输入导致复位之后的相关状态，请参阅表4-2。下面接着详细介绍各个管脚的信息。

注意: 以下管脚配置为复位后的 JTAG 端口管脚。有关如何重新编程配置这些管脚，请参阅“通用输入/输入端口（GPIOs）”（575页）。

表 **4-2.** 上电复位或 **RST** 生效后的 **JTAG** 端口管脚状态

管脚名称	数据方向	内部上拉	内部下拉	驱动强度	驱动值
TCK	输入	使能	禁能	N/A	N/A
TMS	输入	使能	禁能	N/A	N/A
TDI	输入	使能	禁能	N/A	N/A
TDO	输出	使能	禁能	2-mA 驱动器	高阻

4.3.1.1 测试时钟输入 (TCK)

TCK 管脚是 JTAG 模块的时钟。通过提供该时钟，测试逻辑可以独立于其它系统时钟而单独运行，这个时钟确保菊链的多个JTAG TAP控制器可以在组件之间同步传送串行测试数据。正常工作期间，TCK 通过一个自由运行的时钟（额定占空比为 50%）来驱动。必要时，可以将 TCK 保持为 0 或 1，并持续多个周期。当 TCK 保持为 0 或 1 时，TAP 控制器的状态不发生改变，且 JTAG 指令和数据寄存器中的数据不会丢失。

默认情况下，TCK 管脚的内部上拉电阻在复位后使能，这样可以确保该管脚在没有外部源驱动的情况下不进行计时。只要 TCK 管脚连续被外部源驱动，我们就可以通过关闭内部上拉和下拉电阻来节省内部功耗（请参阅601页 和603页）。

4.3.1.2 测试模式选择 (TMS)

TMS 管脚选择 JTAG TAP 控制器的下一个状态。TMS在 TCK 的上升沿被采样。根据当前的 TAP 状态和 TMS 的采样值进入下一个状态。因为 TMS 管脚在 TCK 的上升沿被采样，所以 “IEEE 标准 1149.1” 认为 TMS 的值在 TCK 的下降沿改变。

保持 TMS 高电平持续 5 个连续的 TCK 周期，将驱使 TAP 控制器状态机进入 Test-Logic-Reset（测试逻辑复位）状态。当TAP控制器进入Test-Logic-Reset状态时，JTAG模块和相关寄存器都将复位

为默认值。这个过程可被用于初始化JTAG控制器。图4-2（179页）中列出了所有 JTAG 测试访问端口状态机。

默认情况下，TMS 管脚的内部上拉电阻在复位后使能。改变 GPIO 端口 C 的上拉电阻设置时，应该确保 PC1/TMS 上的内部上拉电阻保持启用；否则可能会丢失 JTAG 通信（请参阅601页）。

4.3.1.3 测试数据输入 (TDI)

TDI 管脚将一串串行信息传送给 IR 链和 DR 链。TDI在 TCK 的上升沿被采样，并根据当前 TAP 状态和当前指令，将这个数据传送到合适的移位寄存器链。因为 TDI 管脚在 TCK 的上升沿被采样，所以 “IEEE 标准 1149.1” 认为 TDI 的值在 TCK 的下降沿改变。

默认情况下，TDI 管脚的内部上拉电阻在复位后使能。改变 GPIO 端口 C 的上拉电阻设置时，应该确保 PC2/TDI 上的内部上拉电阻保持启用；否则可能会丢失 JTAG 通信（请参阅601页）。

4.3.1.4 测试数据输出 (TDO)

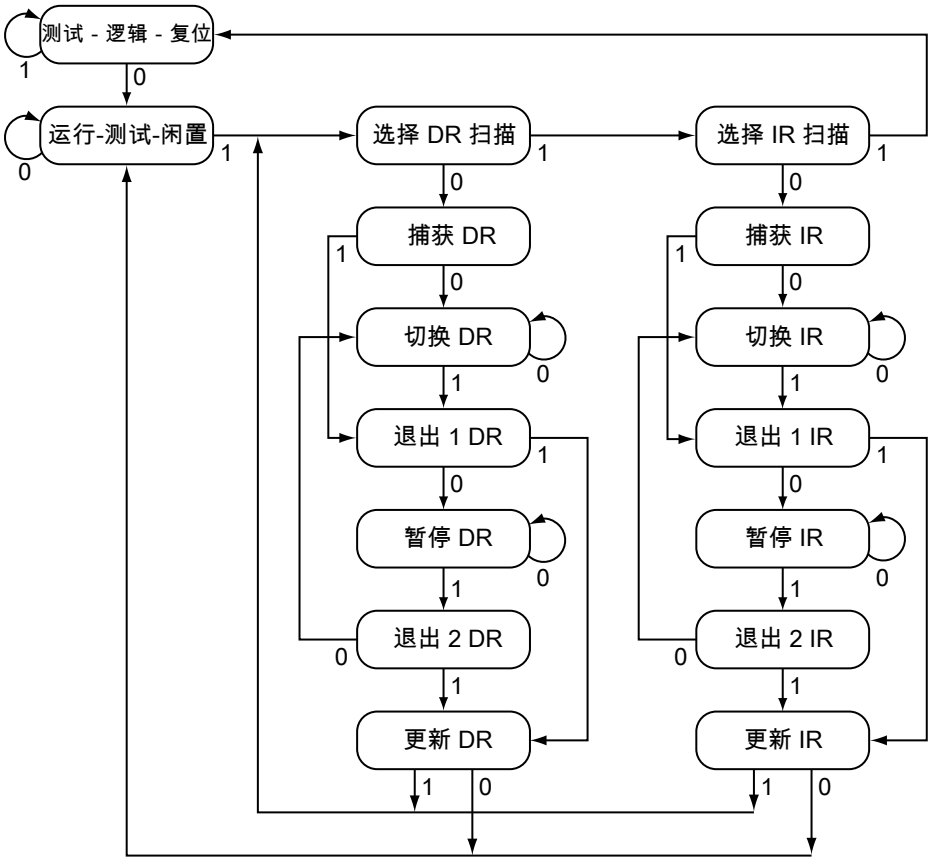
TDO 管脚将一串串行信息从 IR 链或 DR 链输出。TDO 的值取决于当前的 TAP 状态、当前指令、以及正在访问的数据链中的数据。为了在不使用 JTAG 端口时节省功耗，若当前并没有移出数据的活动，TDO 管脚将被放置于停止的驱动状态中。因为在菊链配置中，TDO 可以与另一个控制器的 TDI 管脚相连，所以根据 “IEEE 标准 1149.1”，TDO 的值在 TCK 的下降沿发生变化。

默认情况下，TDO 管脚的内部上拉电阻在复位后使能，这样确保了在不使用 JTAG 端口时，该管脚依然保持在一个不变的逻辑电平。在特定的 TAP 控制状态中，如果高阻输出值尚可接受，那么可以通过关闭内部上拉和内部下拉电阻来节省内部功耗（请参阅601页 和603页）。

4.3.2 JTAG TAP 控制器

JTAG TAP 控制器状态机如图4-2 所示。TAP 控制器状态机在上电复位 (POR) 时将重置为测试-逻辑-复位 (Test-Logic-Reset) 状态。为了在微控制器上电后复位 JTAG 模块，TMS 输入必须保持高电平并持续 5 个 TCK 时钟周期，以此来复位 TAP 控制器和所有相关的 JTAG 链。在 TMS 管脚上发出正确的序列，可以使 JTAG 模块移入新指令、移入数据、或在扩展测试序列期间变为空闲。关于 TAP 控制器功能和每个状态发生的操作的详细情况，请参考 “IEEE 标准 1149.1”。

图 **4-2.** 测试访问端口状态机



4.3.3 移位寄存器

移位寄存器由串行移位寄存器链和并行加载寄存器组成。串行移位寄存器链在 TAP 控制器的 CAPTURE 状态下采样特定的信息，且在 TAP 控制器的 SHIFT 状态下，允许该信息通过 TDO 管脚移出。当采样数据正从 TDO 管脚移出链的同时，新的数据正从 TDI 管脚移入串行移位寄存器。在 TAP 控制器的 UPDATE 状态期间，这些新的数据将被存放到并行加载寄存器中。有关各移位寄存器的详细信息，请参阅“寄存器描述”（182页）。

4.3.4 操作注意事项

在使用JTAG模块时必须考虑某些操作参数。由于JTAG管脚能被编程为GPIO，所以必须考虑这些管脚的线路板配置和复位条件。另外，由于JTAG模块已经集成了ARM串行线调试，所以在这两种操作模式间的切换方法如下所所述。

4.3.4.1 GPIO 功能

当微控制器通过 POR 或者 RST 复位时，JTAG/SWD 端口管脚默认为它们的 JTAG/SWD 配置。在这些 JTAG/SWD 管脚上的默认配置包括：使能数字功能（端口 C GPIO 数字使能寄存器 (GPIODEN) 中的 DEN[3:0] 置位）、使能上拉电阻（端口 C GPIO 上拉选择寄存器 (GPIOPUR) 中的 PUE[3:0] 置位）、禁止下拉电阻（端口 C GPIO 下拉选择寄存器 (GPIOPDR) 中的 PDE[3:0] 清零）、使能备用硬件功能（端口 C GPIO 备用功能选择寄存器 (GPIOAFSEL) 中的 AFSEL[3:0] 置位）。请参阅595页、601页、603页以及606页。

复位后软件可以通过清零端口 C GPIOAFSEL 寄存器的 AFSEL[3:0] 将这些管脚配置为 GPIO。如果用户不需要JTAG/SWD端口用于调试或板级测试，那么这将多提供4个GPIO供设计中使用。

小心 – 用户可以建立一个软件序列来阻止调试器连接到 **TM4C1231H6PM** 微控制器。如果将程序代码加载到 **Flash** 中会立即将 **JTAG** 管脚变成其 **GPIO** 功能，那么在 **JTAG** 管脚功能切换之前，调试器将没有足够的时间去连接和终止控制器。结果调试器可能被锁定在该部分外。通过使用一个基于外部或软件的触发器来恢复**JTAG**功能的软件程序可以避免这个问题。如果未实施软件例程，且器件锁定在此部分以外，则可通过 **TM4C1231H6PM Flash** 编程器的**"**解锁**"**功能解决此问题。有关详细信息，请参阅 **TI** 网站的 **LMFLASHPROGRAMMER** 部分。

GPIO 提交控制寄存器提供了保护层以防止对重要硬件外设的意外编程。系统针对可用作四个 JTAG/SWD 管脚以及 NMI 管脚的 GPIO 管脚提供了保护功能（管脚号请参见"信号表"（1008页））。向 GPIO 备用功能选择 (GPIOAFSEL) 寄存器（请参阅595页）、GPIO 上拉电阻选择 (GPIOPUR) 寄存器（请参阅601页）、GPIO 下拉电阻选择 (GPIOPDR) 寄存器（请参阅603页）以及 GPIO 数字使能 (GPIODEN) 寄存器（请参阅606页）中受保护的位写入数据将不会确认保存，除非 GPIO 锁定 (GPIOLOCK) 寄存器（请参阅608页）没有被锁定，同时 GPIO 确认 (GPIOCR) 寄存器（请参阅609页）中相应的位被置位。

4.3.4.2 JTAG/SWD 通信

由于调试时钟和系统时钟可以在不同的频率运行，所以必须注意要与JTAG/SWD接口保持可靠的通信。在Capture-DR状态，先前处理的结果，如果有，将被返回，同时伴随一个3位的ACK应答。软件应该检测这个ACK应答，以便查看在初始化一个新的处理前，先前的操作是否已经完成。或者，如果系统时钟至少比调试时钟（TCK 或 SWCLK）快 8 倍，那么先前的操作有足够的时间来完成，ACK 位也不需要被检测。

4.3.4.3 恢复一个**"**锁死**"**的微控制器

注意: 执行以下序列将会把"非易失性寄存器编程"（465页）中列出的非易失性寄存器恢复为出厂默认值。执行以下序列将先对 Flash 存储器进行整体擦除，然后恢复非易失性寄存器。

另外，执行以下序列还将擦除 EEPROM，并将其换位写入计数器返回到出厂默认值。

如果软件将任意一个 JTAG/SWD 管脚配置为 GPIO，并且失去与调试器进行通信的能力，那么可以用调试端口解锁序列来恢复微控制器。当微控制器保持在复位时，执行总共10次的JTAG到SWD和SWD到JTAG的切换序列，将会整体擦除Flash存储器。调试端口解锁序列为：

1. 发出和保持 RST 信号。
2. 向器件供电。
3. 执行"JTAG到SWD 切换"一节（181页）中 JTAG 到 SWD 切换序列的第 1 步和第 2 步。
4. 执行"SWD到JTAG 切换"一节（181页）中 SWD 到 JTAG 切换序列的第 1 步和第 2 步。
5. 执行JTAG到SWD切换序列的第1步和第2步。
6. 执行SWD到JTAG切换序列的第1步和第2步。
7. 执行JTAG到SWD切换序列的第1步和第2步。
8. 执行SWD到JTAG切换序列的第1步和第2步。
9. 执行JTAG到SWD切换序列的第1步和第2步。
10. 执行SWD到JTAG切换序列的第1步和第2步。
11. 执行JTAG到SWD切换序列的第1步和第2步。

12. 执行SWD到JTAG切换序列的第1步和第2步。

13. 释放 RST 信号。

14. 等待 400 ms。

15. 微控制器进入上电周期。

4.3.4.4 ARM 串行线调试 (SWD)

为了无缝结合 ARM 串行线调试 (SWD) 功能，串行线调试器必须能够与 Cortex-M4F 内核相连，而无需执行或者了解 JTAG 的运行情况。这可以通过一个SWD报头来实现，这个报头在SWD会话开始前发出。

用来使能SWJ-DP模块的SWD接口的报头在TAP控制器处于Test-Logic-Reset状态时开启。此后，前导码会使 TAP 控制器依次进入以下状态：运行-测试-闲置 (Run Test Idle)、选择 DR (Select DR)、选择 IR (Select IR)、测试 - 逻辑 - 复位 (Test Logic Reset)、测试 - 逻辑 - 复位 (Test Logic Reset)、运行-测试 - 闲置 (Run Test Idle)、运行-测试 - 闲置 (Run Test Idle)、选择 DR (Select DR)、选择 IR (Select IR)、测试 - 逻辑 - 复位 (Test Logic Reset)、测试 - 逻辑 - 复位 (Test Logic Reset)、运行-测试 - 闲置 (Run Test Idle)、运行-测试 - 闲置 (Run Test Idle)、选择 DR (Select DR)、选择 IR (Select IR) 以及测试 - 逻辑 - 复位 (Test Logic Reset) 状态。

通过TAP状态机的这个序列进行步进，将使能SWD接口并禁止JTAG接口。关于该操作和 SWD 接口的更多信息，可查看 “ARM® Debug Interface V5 Architecture Specification”。

由于这个序列是一系列有效的可发出的 JTAG 操作，所以 ARM JTAG TAP 控制器并不完全符合“IEEE 标准 1149.1”。这是 ARM JTAG TAP控制器唯一不完全符合规范的地方。由于TAP控制器在正常操作中该序列出现的可能性很低，所以应该不会影响JTAG接口的正常执行。

JTAG到SWD 切换

为了将调试访问端口 (DAP) 的操作模式由JTAG切换到SWD，外部调试硬件必须向微控制器发送切换报头。用于切换到 SWD 模式的 16 位 TMS/SWDIO 命令定义为 b1110.0111.1001.1110，先发送 LSB。当先发送LSB时，这个命令也可以表示为0xE79E。完整的切换序列应包括对 TCK/SWCLK 和 TMS/SWDIO 信号的下列操作：

1. TCK/SWCLK 为高电平时，发送至少 50 个 TMS/SWDIO 周期，以确保 JTAG 和 SWD 都处于复位状态。

2. 在 TMS/SWDIO 上发送 16 位 JTAG 到 SWD 的切换命令 0xE79E。

3. 在 TCK/SWCLK 为高电平时，发送至少 50 个 TMS/SWDIO 周期，以确保在发送切换序列之前，如果 SWJ-DP 已经在 SWD 模式，那么 SWD 进入线复位状态。

要确认调试访问端口 (DAP) 已切换为串行线调试 (SWD) 的操作模式，需执行 SWD READID 操作。可将 ID 值与器件的已知 ID 进行比较，以确认该切换。

SWD到JTAG 切换

为了将调试访问端口 (DAP) 的操作模式由SWD切换到JTAG，外部调试硬件必须向微控制器发送切换报头。用于切换到 JTAG 模式的 16 位 TMS/SWDIO 命令定义为 b1110.0111.0011.1100，先发送 LSB。当先发送LSB时，这个命令也可以表示为0xE73C。完整的切换序列应包括对 TCK/SWCLK 和 TMS/SWDIO 信号的下列操作：

1. TCK/SWCLK 为高电平时，发送至少 50 个 TMS/SWDIO 周期，以确保 JTAG 和 SWD 都处于复位状态。

2. 在 TMS/SWDIO 上发送 16 位 SWD 到 JTAG 的切换命令 0xE73C。

3. 在 TCK/SWCLK 为高电平时，发送至少 50 个 TMS/SWDIO，以确保在发送切换序列之前，如果 SWJ-DP 已经在 JTAG 模式，那么 JTAG 进入测试逻辑复位状态。

要确认调试访问端口 (DAP) 已切换为 JTAG 操作模式，将 JTAG 指令寄存器 (IR) 设置为 IDCODE 指令，并移出数据寄存器 (DR)。可将 DR 值与器件的已知 IDCODE 进行比较，以确认该切换。

4.4 初始化和配置

在上电复位或外部复位 (RST) 后，JTAG 管脚将被自动配置以进行 JTAG 通信。不需要用户定义的初始化或配置。但是，如果用户应用程序将这些管脚变成GPIO功能，那么在恢复JTAG通信前，这些管脚必须被配置到它们的JTAG功能。为了使管脚返回到它们的 JTAG 功能，需要使用 GPIOAFSEL 寄存器使能 4 个 JTAG 管脚 (PC[3:0]) 的备用功能。另外，要启用备用功能，应将对 4 个 JTAG 管脚 (PC[3:0]) 上 GPIO 管脚配置的任何其他更改都恢复为默认设置。

4.5 寄存器描述

在JTAG TAP控制器里的寄存器或移位寄存器链都没有存储器映射，它们不能通过片上的高级外设总线 (APB) 访问。但是，这些JTAG控制器里的寄存器都可以通过TAP控制器被串行访问。这些寄存器包括指令寄存器和6个数据寄存器。

4.5.1 指令寄存器 (IR)

JTAG TAP 指令寄存器 (IR) 是一个 4 位串行扫描链，它通过一个并行加载寄存器在 JTAG 的 TDI 和 TDO 管脚之间进行连接。当 TAP 控制器处于正确的状态时，相应的位便可移入指令寄存器 (IR)。这些位一旦移入链中并且更新后，就会被解析成当前指令。有关指令寄存器 (IR) 位的译码，请参阅表4-3。下面对每条指令及其相关的数据寄存器进行了详细解释。

表 **4-3. JTAG** 指令寄存器命令

IR[3:0]	指令	描述
0x0	EXTEST	将由SAMPLE/PRELOAD 指令预加载到边界扫描链的值驱动到引脚(pad)上。
0x2	SAMPLE / PRELOAD	捕获当前的I/O值，并在新的预加载数据移入时将采样的值移出边界扫描链。
0x8	ABORT	将数据移入“ARM调试端口中止（Debug Port Abort）寄存器”
0xA	DPACC	将数据移入和移出“ARM DP 访问寄存器”。
0xB	APACC	将数据移入和移出“ARM AC访问寄存器”。
0xE	IDCODE	将 “IEEE 标准 1149.1” 定义的制造信息加载到 IDCODE 链并将它移出。
0xF	旁路	通过一个移位寄存器链将 TDI 与 TDO 相连。
其它	保留	默认为 BYPASS 指令，以确保 TDI 总是连接到 TDO。

4.5.1.1 EXTEST 指令

EXTEST 指令与本身的数据寄存器链没有联系。EXTEST指令使用的是SAMPLE/PRELOAD指令预加载边界扫描数据寄存器里的数据。当EXTEST指令出现在指令寄存器时，在与输出和输出使能相关的边界扫描数据寄存器里的预加载数据被用来驱动GPIO引脚，而不是来自内核的信号来驱动。通过将已知的值驱动到控制器外的测试，该指令可用于验证联通性。当EXTEST指令出现在指令寄存器时，边界扫描数据寄存器可以被访问来采样和移出当前数据，并且加载新的数据到边界扫描数据寄存器。

4.5.1.2 SAMPLE/PRELOAD 指令

SAMPLE/PRELOAD 指令连接 TDI 和 TDO 之间的边界扫描数据寄存器数据链。该指令采样当前引脚的状态以供观察，并且预加载新的测试数据。每个GPIO引脚都有一个相关的输入，输出和输出使能信号。在该指令执行期间，当TAP控制器进入到 Capture DR 状态时，每个GPIO引脚的输入，输出和输出使能信号都会被捕捉。当 TAP 控制器处于 Shift DR 状态时这些采样值被串行移出到 TDO 上，它们可以在各种测试中用于观察和比较。

在这些输入、输出和输出使能信号的采样值被移出边界扫描数据寄存器的同时，新的数据也正通过 TDI 管脚移入边界扫描数据寄存器。一旦新数据被移入边界扫描数据寄存器，当TAP控制器进入 Update DR状态时数据将被保存在并行加载寄存器。这种并行加载寄存器的更新可以将数据预加载到每个与输入，输出和输出使能相关的边界扫描数据寄存器中。这些被预加载的数据可以和EXTEST 指令一起使用，以便将数据送入控制器内，或将数据从控制器内送出。更多信息参见 “边界扫描数据寄存器”（184页）。

4.5.1.3 ABORT 指令

ABORT 指令连接 TDI 和 TDO 之间的相应的 ABORT 数据寄存器链。该指令提供了对ARM调试访问端口 (DAP) 的ABORT寄存器的读和写访问。将适当的数据移入这个数据寄存器可以清除各种错误位，或对之前的请求启动DAP中止。更多信息请参阅“ABORT 数据寄存器”（185页）。

4.5.1.4 DPACC 指令

DPACC 指令连接 TDI 和 TDO 之间的相应的 DPACC 数据寄存器链。该指令提供了对ARM调试访问端口 (DAP) 的DPACC寄存器的读和写访问。将适当的数据移入该寄存器并且从该寄存器读取输出数据，便可以对ARM调试和状态寄存器进行读和写访问。更多信息参见 “DPACC 数据寄存器”（185页）。

4.5.1.5 APACC 指令

APACC 指令连接 TDI 和 TDO 之间的相应的 APACC 数据寄存器链。该指令提供了对ARM调试访问端口 (DAP) 的APACC寄存器的读和写访问。将适当的数据移入该寄存器并且从该寄存器读取输出数据，便可以通过调试端口对内部组件和总线进行读和写访问。更多信息参见 “APACC 数据寄存器”（184页）。

4.5.1.6 IDCODE 指令

IDCODE 指令连接 TDI 和 TDO 之间的相应的 IDCODE 数据寄存器链。该指令提供了有关制造商，器件编号和ARM内核版本等信息。测试设备和调试器可以使用这些信息来自动配置输入和输出数据流。当上电复位 (POR) 被确认，或者进入Test-Logic-Reset状态时，IDCODE 指令默认加载到JTAG 的指令寄存器。更多信息参见 “IDCODE 数据寄存器”（184页）。

4.5.1.7 BYPASS 指令

BYPASS 指令连接 TDI 和 TDO 之间的相应的 BYPASS 数据寄存器链。该指令用来在 TDI 和 TDO 端口之间创建一条长度最短的串行路径。BYPASS数据寄存器是个1位移位寄存器。对于不需要特别测试的组件，通过加载BYPASS指令可以在JTAG扫描链中将它们旁路掉，由此提高了测试效率。更多信息参见 “BYPASS 数据寄存器”（184页）。

4.5.2 数据寄存器

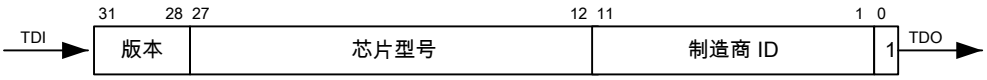
JTAG 模块包含6个数据寄存器。串行数据寄存器链包括：IDCODE、BYPASS、边界扫描、APACC、DPACC 和 ABORT。它们将在下面的小节中进行讨论。

4.5.2.1 IDCODE 数据寄存器

根据 “IEEE 标准 1149.1” 的定义，32 位 IDCODE 数据寄存器的格式如图4-3所示。该标准要求每个与 JTAG 兼容的设备都将 IDCODE 指令或者 BYPASS 指令当作默认指令来执行。IDCODE 数据寄存器的 LSB (最低位)被定义成 1，以将它和 LSB 为 0 的 BYPASS 指令区分开来。这使得自动配置测试工具可以决定哪个指令是默认指令。

JTAG端口主要用于制造商测试组件以及编程开发和调试。为了便于使用自动配置调试工具，IDCODE 指令将输出 0x4BA00477 值。通过这个值，调试器可以自动进行配置，以便在调试过程中能够和 Cortex-M4F 一起正确工作。

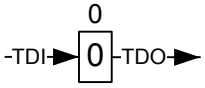
图 **4-3. IDCODE** 寄存器格式



4.5.2.2 BYPASS 数据寄存器

根据 “IEEE 标准 1149.1” 的定义，1 位 BYPASS 数据寄存器的格式如图4-4所示。该标准要求每个与 JTAG 兼容的器件将 BYPASS 指令或者 IDCODE 指令当作默认指令来执行。BYPASS 数据寄存器的 LSB 被定义成 0，以便将它与 LSB 为 1 的 IDCODE 指令区分开来。这使得自动配置测试工具可以决定哪个指令是默认指令。

图 **4-4. BYPASS** 寄存器格式

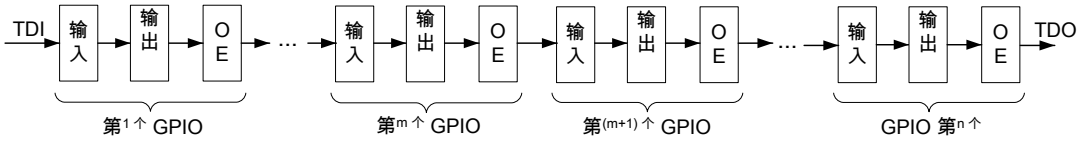


4.5.2.3 边界扫描数据寄存器

边界扫描数据寄存器的格式如图4-5所示。每个GPIO管脚都包含在边界扫描数据寄存器中，这些管脚的顺序为 JTAG 端口管脚排列的逆时针方向。每个GPIO管脚有3个相关的数字信号包含在链中。这些信号是输入，输出和输出使能，它们按照图示顺序排列。

当SAMPLE/PRELOAD指令访问边界扫描数据寄存器时，每个数字引脚的输入，输出和输出使能被采样并且移出链供校验。这些值的采样发生在 TAP 控制器 Capture DR 状态下的 TCK 上升沿。当被采样的数据正从 TAP 控制器 Shift DR 状态下的边界扫描链移出时，新的数据便可以被预先加载到链中，以便 EXTEST 指令使用。EXTEST 指令强制将数据移出控制器。

图 **4-5.** 边界扫描寄存器格式



4.5.2.4 APACC 数据寄存器

关于由 ARM 定义的 35 位 APACC 数据寄存器的格式可参阅 “ARM® Debug Interface V5 Architecture Specification”。

4.5.2.5 DPACC 数据寄存器

关于由 ARM 定义的 35 位 DPACC 数据寄存器的格式可参阅“ARM® Debug Interface V5 Architecture Specification”。

4.5.2.6 ABORT 数据寄存器

关于由 ARM 定义的 35 位 ABORT 数据寄存器的格式可参阅“ARM® Debug Interface V5 Architecture Specification”。

5 系统控制

系统控制配置整个器件的操作，并提供器件信息。可配置的特性包括复位控制、NMI操作、功率控制、时钟控制和低功耗模式。

5.1 信号描述

以下表格列出了系统控制模块的外部信号，并描述了各自的功能。NMI 信号是 GPIO 信号的备用功能，复位后用作 GPIO 信号。NMI 管脚具备提交保护，并需要特定的处理过程才能被配置为任意备用功能或者随后返回 GPIO 功能，请参考“确认控制”（581页）。下表中“管脚复用/赋值”栏给出了 NMI 信号的 GPIO 管脚位置。GPIO 备选功能选择 (GPIOAFSEL) 寄存器 (595页) 里的 AFSEL 位应被置位以选择 NMI 功能。括号里边的数字表示必须编入 GPIO 端口控制 (GPIOPCTL) 寄存器（612页) 中的 PMCn 位域里的编码，用以将 NMI 信号分配给指定的 GPIO 端口管脚。有关如何配置 GPIO 的更多信息，请参阅“通用输入/输入端口（GPIOs）”（575页）。其余的管脚（在管脚复用/赋值栏中标记为“固定”的管脚）都有一个固定的管脚赋值和功能。

表 **5-1.** 系统控制**;** 时钟 信号 **(64LQFP)**

管脚名称	管脚编号	管脚复用/管脚赋值	管脚类型	缓冲区类型[a]	描述
NMI	10 28	PD7 (8) PF0 (8)	I	TTL	不可屏蔽的中断
OSC0	40	固定	I	模拟	主振荡器晶体输入或外部时钟参考输入。
OSC1	41	固定	O	模拟	主振荡器晶体输出。当使用外部单端参考时钟源时，此管脚应悬空。
RST	38	固定	I	TTL	系统复位输入

a. TTL 表示管脚的电压水平与 TTL 一致。

5.2 功能说明

系统控制模块提供以下功能：

- 器件标识，见“器件标识”（186页）
- 本地控制，例如复位（见 “复位控制”（186页）），电源（见 “功率控制”（192页））和时钟控制（见 “时钟控制”（193页））
- 系统控制（运行、睡眠和深度睡眠模式），见 “系统控制”（199页）

5.2.1 器件标识

有些只读寄存器可以向软件提供关于微控制器的信息，比如版本、器件编号、存储器大小以及其器件上的外设存在。器件标识 0 (DID0)（208页）和器件标识 1 (DID1)（210页）寄存器提供关于器件版本、封装、温度范围等的详细信息。从系统控制偏移量 0x300 开始的外设存在寄存器（如看门狗定时器外设存在 (PPWD) 寄存器）提供关于器件中各种类型模块的数量信息。最后，外设属性寄存器的各个外设寄存器空间的偏移量0xFC0处提供了片上外设性能的信息（如GPTM外设属性 (GPTMPP) 寄存器）。之前的器件使用器件功能 (DC0-DC9) 寄存器显示关于外设及其性能的信息。这些器件的寄存器用于向后兼容软件，但不提供之前的器件所不适用的外设信息。

5.2.2 复位控制

这一节讨论复位期间硬件方面的功能以及复位序列之后的系统软件要求。

5.2.2.1 复位源

TM4C1231H6PM 微控制器有 6 个复位源：

1. 上电复位 (POR)（见第188页）。
2. 外部复位输入管脚 (RST) 有效（见第188页）。
3. 以下事件可触发掉电检测：（见第190页）。
   - V_{DD} 低于 BOR0。触发值是 BOR0 的最高 V_{DD} 电压。
   - V_{DD} 低于 BOR1。触发值是 BOR1 的最高 V_{DD} 电压。
4. 软件启动的复位（利用软件复位寄存器）（见第190页）。
5. 违反看门狗定时器复位条件（见第191页）。
6. MOSC 故障（见第192页）。

表5-2 提供了不同复位操作结果的摘要。

表 **5-2.** 复位源

复位源	内核复位？	**JTAG** 复位？	片上外设复位？
上电复位	是	是	是
RST	是	仅管脚配置	是
掉电复位	是	仅管脚配置	是
使用 APINT 寄存器中的 SYSRESREQ 位进行软件系统请求复位。	是	仅管脚配置	是
使用 APINT 寄存器中的 VECTRESET 位进行软件系统请求复位。	是	仅管脚配置	否
软件外设复位	否	仅管脚配置	是[a]
看门狗复位	是	仅管脚配置	是
MOSC失败复位	是	仅管脚配置	是

a. 使用软件复位控制寄存器可在在模块基础上进行编程。

复位后，复位原因 (RESC) 寄存器根据复位原因置位。该寄存器中的位具有粘着特性，经过多个复位序列后仍能保持其状态，内部 POR 复位除外。内部 POR 复位后，RESC 寄存器中除 POR 指示器对应位之外的所有位都清零。RESC 寄存器的位可以通过写 0 来清零。

任何复位内核的复位操作中，通过使用启动配置 (BOOTCFG) 寄存器中配置好的 GPIO 信号，用户可以选择让内核直接执行 ROM 的 Boot Loader 或 Flash 存储器上的应用程序。

复位时，将执行以下序列：

1. 读取 BOOTCFG 寄存器。如果 EN 位被清零，那么执行 ROM 的引导装载程序。
2. 在ROM的Boot Loader中，指定的GPIO管脚的状态与规定的极性相比较，如果管脚状态与规定的极性匹配，那么将 ROM 映射到地址 0x0000.0000 并继续执行 ROM 的 Boot Loader。

3. 如果 EN 位置位或管脚状态与规定的极性不匹配，那么读取地址 0x0000.0004 的数据。如果该地址的数据是 0xFFFF.FFFF，那么将 ROM 映射到地址 0x0000.0000 并继续执行 ROM 的 Boot Loader。

4. 如果地址 0x0000.0004 中的数据不是 0xFFFF.FFFF，堆栈指针 (SP) 将加载 Flash 存储器地址 0x0000.0000 的数据，程序计数器 (PC) 将加载地址 0x0000.0004 的数据。随后用户应用程序开始执行。

注意: 如果设备未能完成初始化阶段，会切换 TDO 输出管脚，以表明设备当前未执行。此功能用于调试。

例如，如果 BOOTCFG 寄存器写入值 0x0000.3C01 并提交，那么复位时将检查 PB7 来决定是否执行 ROM 的 Boot Loader。如果 PB7 是低电平，内核无条件开始执行 ROM 的 Boot Loader。如果 PB7 是高电平，若 0x0000.0004 位置的复位向量不是 0xFFFF.FFFF，那么执行 Flash 存储器的应用程序，若复位向量是0xFFFF.FFFF，则执行ROM的Boot Loader。

5.2.2.2 上电复位 (POR)

注意: JTAG 控制器仅可以通过上电复位重置。

内部上电复位 (POR) 电路监测电源电压 (V_{DD})，并且在电源达到阈值 (V_{VDD_POK}) 时向包括 JTAG 在内的所有内部逻辑产生复位信号。当片上上电复位脉冲结束时，微控制器必须在规定的参数范围内工作（见“Power and Brown-Out”（1036页））。当应用要求使用外部复位信号让微控制器更长时间地保持在复位状态时（相对使用内部 POR 而言），可以使用 RST 输入，具体原因请参阅“外部 RST 管脚”（188页）。

上电复位序列如下：

1. 微控制器等待内部POR变为无效。

2. 内部复位释放，内核从存储器加载初始堆栈指针、初始程序计数器以及由程序计数器指定的第1条指令，然后开始执行。

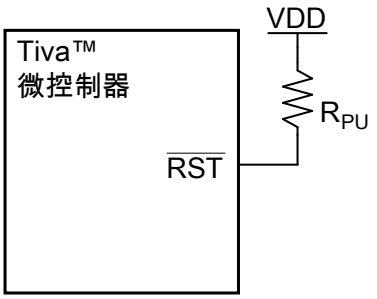
内部 POR 只在微控制器最初上电以及休眠唤醒时激活。上电复位时序如“Power and Brown-Out”（1036页）所示。

5.2.2.3 外部 RST 管脚

注意: 建议 RST 信号的跟踪线路越短越好。确保将连接 RST 信号的任何元件布置得尽可能靠近微控制器。

如果应用中只使用内部 POR 电路，那么 RST 输入必须通过一个可选的上拉电阻（0 至 100 KΩ）连接到电源 (V_{DD})，如图5-1（189页）所示。RST 输入的滤波功能需要最小脉宽，以便复位脉冲被识别，见表21-11（1041页）。

图 **5-1.** 基本 **$\overline{\text{RST}}$** 配置



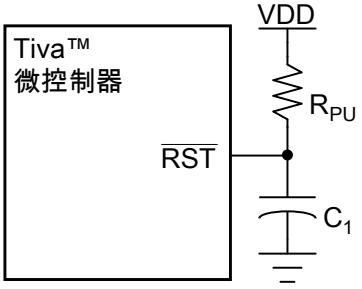
R_{PU} = 0 至 100 kΩ

外部复位管脚 ($\overline{\text{RST}}$) 复位微控制器，包括内核和所有片上外设。外部复位序列如下：

1. 外部复位管脚 ($\overline{\text{RST}}$) 在 T_{MIN} 规定的时间持续有效，然后失效（请参考“Reset”（1041页））。
2. 内部复位释放，内核从存储器加载初始堆栈指针、初始程序计数器以及由程序计数器指定的第1条指令，然后开始执行。

为提高噪声免疫和/或延迟上电复位，$\overline{\text{RST}}$ 输入可以连接至一个 RC 网络，见图5-2（189页）。

图 **5-2.** 延长上电复位时间的外部电路

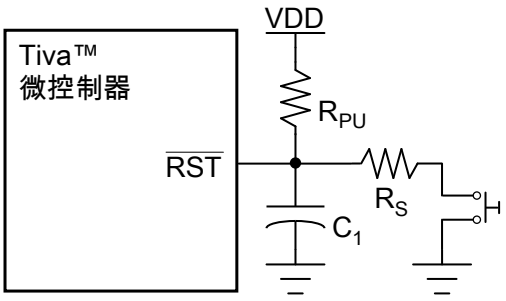


R_{PU} = 1 kΩ 至 100 kΩ

C_1 = 1 nF 至 10 µF

如果应用中需要使用外部复位开关，图5-3（190页）显示了适当的电路来使用。

图 **5-3.** 复位电路由开关控制



典型 R_{PU} = 10 kΩ

典型 R_S = 470 Ω

C_1 = 10 nF

R_{PU} 和 C_1 元件确定了上电延时。

外部复位时序如图21-11（1042页）所示。

5.2.2.4 掉电复位 (BOR)

微控制器提供一个由以下事件触发的掉电检测电路：

- V_{DD} 低于 BOR0。外部电源电压 V_{DD} 低于指定的 V_{DD} BOR0 值。触发值是 BOR0 的最高 V_{DD} 电压。
- V_{DD} 低于 BOR1。外部电源电压 V_{DD} 低于指定的 V_{DD} BOR1 值。触发值是 BOR1 的最高 V_{DD} 电压。

应用通过读取复位原因 (RESC) 寄存器即可识别导致复位的 BOR 事件。检测到掉电条件时，默认条件将产生一次复位。可对 BOR 事件进行编程，以便在上电和掉电复位控制 (PBORCTL) 寄存器的 BOR0 或 BOR1 位清零时产生中断。

掉电复位序列如下：

1. 当 V_{DD} 降至低于 V_{BORnTH} 时，内部 BOR 条件将置位。有关 V_{BORnTH} 值，请参考“Power and Brown-Out”（1036页）。
2. 如果BOR条件存在，内部复位就有效。
3. 内部复位释放，微控制器获取并加载初始堆栈指针、初始程序计数器以及由程序计数器指定的第1条指令，然后开始执行。

掉电复位的效果等同于一次有效的外部 RST 输入，并且该复位将会保持有效，直到 V_{DD} 恢复到正确的电压级别。在复位中断处理程序中可以检查 RESC 寄存器，以确定掉电条件是否是复位的原因，从而使软件确定需要恢复哪些操作。

内部掉电复位时序如“Power and Brown-Out”（1036页）所示。

5.2.2.5 软件复位

软件可以复位一个特别的外设或者对整个微控制器产生一个复位

通过系统控制偏移量 0x500 处开始的外设专用复位寄存器（例如看门狗定时器软件复位 (SRWD) 寄存器），软件可以单独复位各个外设。如果外设对应的位被置位随后清零，那么该外设被复位。

软件可以通过置位应用中断和复位控制 (APINT) 寄存器的 SYSRESREQ 位来复位包括内核在内的整个微控制器。软件启动的系统复位序列如下：

1. 通过置位 SYSRESREQ 位即可产生软件微控制器复位。
2. 内部复位有效。
3. 内部复位释放，微控制器从存储器加载初始堆栈指针、初始程序计数器以及由程序计数器指定的第1条指令，然后开始执行。

软件只可以通过置位 APINT 寄存器的 VECTRESET 位来复位内核。软件启动的内核复位序列如下：

1. 通过置位 VECTRESET 位来启动内核复位。
2. 内部复位有效。
3. 内部复位释放，微控制器从存储器加载初始堆栈指针、初始程序计数器以及由程序计数器指定的第1条指令，然后开始执行。

软件启动的系统复位时序如图21-12（1042页）所示。

5.2.2.6 看门狗定时器复位

看门狗定时器模块的功能是阻止系统挂起。TM4C1231H6PM 微控制器提供两个看门狗定时器模块，以防其中一个看门狗时钟源失效。一个看门狗脱离系统时钟运行，另一个脱离精确内部振荡器 (PIOSC) 运行。除了由于PIOSC看门狗定时器模块在不同的时钟范围，对寄存器的访问必须在它们之间有一个时间延迟外，每个模块以相同的方式工作。看门狗定时器可被配置为在第一次超时的时候向微控制器产生一个中断或不可屏蔽中断，在第二次超时的时候产生一个复位。

在看门狗定时器第一次超时发生后，32 位看门狗计数器会重新加载看门狗定时器加载 (WDTLOAD) 寄存器的值并从这个值开始递减计数。如果在第一次溢出中断被清零之前，定时器向下计数到0，同时复位信号使能，那么看门狗定时器将把复位信号发送给微控制器。看门狗定时器复位序列如下：

1. 看门狗定时器第二次溢出时没有被服务。
2. 内部复位有效。
3. 内部复位释放，微控制器从存储器加载初始堆栈指针、初始程序计数器以及由程序计数器指定的第1条指令，然后开始执行。

关于看门狗定时器模块的更多信息，请参阅“看门狗定时器”（692页）。

看门狗复位时序如图21-13（1042页）所示。

5.2.3 不可屏蔽的中断

微控制器有四个不可屏蔽的中断源 (NMI)：

- NMI 信号有效确认
- 主振荡器校验错误
- 中断控制和状态 (INTCTRL) 寄存器的 NMISET 位（ Cortex™-M4F ）（请参阅137页）。

- 看门狗控制 (WDTCTL) 寄存器的 INTTYPE 位置位时，看门狗模块将发生超时中断（见698页）。

为辨认出中断源，软件必须检查中断原因。

5.2.3.1 NMI 管脚

NMI 信号是 GPIO 端口管脚 PD7 或 PF0 的备用功能。如“通用输入/输入端口（GPIOs）”（575页）所描述的那样，若想将该信号用于中断，则必须启用 GPIO 中的备用功能。注意：启用 NMI 备用功能需要使用 GPIO 锁定和提交功能，正如与 JTAG/SWD 功能相关的 GPIO 端口管脚那样，见609页。高电平激活 NMI 信号；启用的 NMI 信号高于 V_{IH} 会启动 NMI 中断序列。

5.2.3.2 主振荡器校验失败

TM4C1231H6PM 微控制器提供了一个主振荡器校验电路。如果振荡器运行得太快或太慢，该电路会产生一个错误条件。如果主振荡器校验电路被启用并产生一个错误，此时会产生一个上电复位并将控制权交给 NMI 处理程序，或产生中断。MOSCCTL 寄存器的 MOSCIM 位决定将发生的动作。在这两种情况中，系统时钟源自动切换为 PIOSC。发生 MOSC 故障复位时，NMI 处理程序将用于解决主振荡器检验故障，因为可以从通用复位处理程序移除必要代码，加速复位处理过程。通过将主振荡器控制 (MOSCCTL) 寄存器的 CVAL 位置位来启用检测电路。主振荡器校验错误在复位原因 (RESC) 寄存器的主振荡器失败状态位 (MOSCFAIL) 显示。有关主振荡器校验电路动作的详细描述，请参阅“主振荡器校验电路”（198页）。

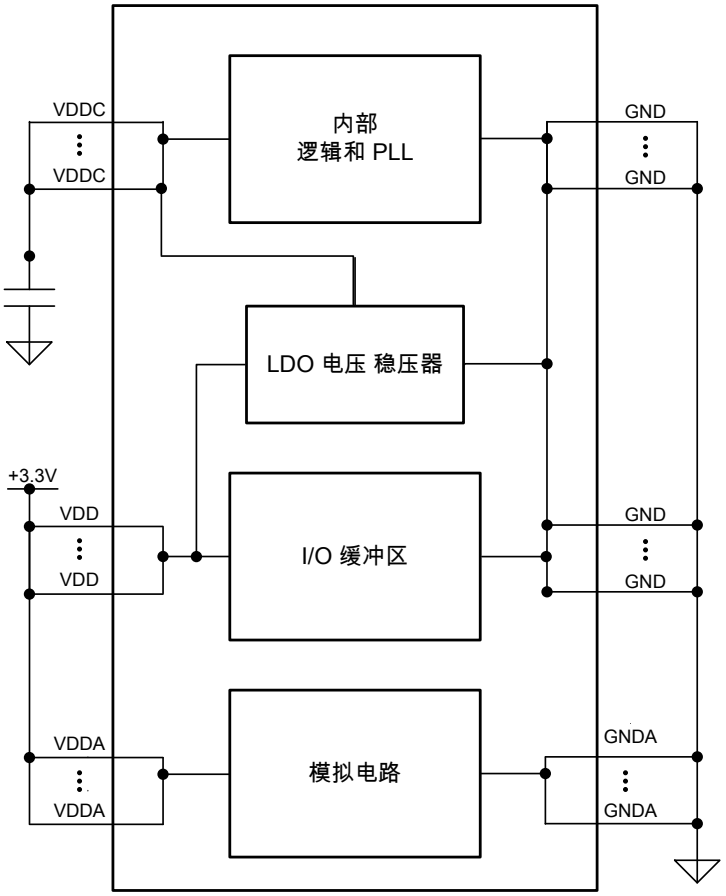
5.2.4 功率控制

TM4C1231H6PM 微控制器提供了一个集成的 LDO 稳压器，可用来对大多数微控制器的内部逻辑提供电源。图5-4显示了功率结构。

不可以使用外部LOD。

注意: VDDA 的电压必须符合表21-5（1031页）中的规格，否则微控制器不能正常工作。VDDA为器件的所有模拟电路供电，包括时钟电路。

图 **5-4.** 功率结构



5.2.5 时钟控制

系统控制决定了该部分的时钟控制。

5.2.5.1 基础时钟源

在微控制器中有多个时钟源可以使用。

- 精确内部振荡器 **(PIOSC).** 精确内部振荡器是一个片上时钟源，在POR期间和之后，微控制器使用该时钟源。它不需要使用任何外部元件，并提供一个 16-MHz 时钟，其校准精度为 ±1%，整个温度范围内的精度为 ±3%（参见“PIOSC Specifications”（1045页））。PIOSC 是为需要精确时钟源并减少系统开销的应用而考虑的。如果需要主振荡器，软件必须在复位后使能主振荡器，并在改变时钟参考前让主振荡器达到稳定。如果休眠模块时间源为 32.768-kHz 振荡器，软件可以根据参考时钟调整内部精确振荡器，以获得更高精确性。不论PIOSC是否是系统时钟源，PIOSC可以被配置为 ADC 时钟源以及 UART 和 SSI 的波特率时钟，见“系统控制”（199页）。

- 主振荡器 **(MOSC).** 主振荡器可通过两种方式提供一个频率精确的时钟源：外部单端时钟源连接到 OSC0 输入管脚，或者外部晶振串接在 OSC0 输入管脚和 OSC1 输出管脚间。如果 PLL 正在使用，晶振的值必须是 5 MHz 到 25 MHz（含）之间的一个支持的频率。如果 PLL 没有被使用，晶振可以是 4 MHz 到 25 MHz 之间的任何一个支持的频率。单端时钟源范围请见表 21-13（1044页）。支持的晶振在 RCC 寄存器的 XTAL 位域列出（见221页）。

- 低频内部振荡器 **(LFIOSC).** 低频内部振荡器适用于深度睡眠省电模式。其频率范围较宽，具体请参阅“Low-Frequency Internal Oscillator (LFIOSC) Specifications”（1045页）。该省电模式受益于精简的内部配电系统，同时也允许MOSC关闭。此外，在深度睡眠模式中时PIOSC可以掉电。
- 休眠模块时钟源. 休眠模块使用连接到 XOSC0 管脚的 32.768-kHz 振荡器计时。32.768-kHz 振荡器可以用于系统时钟，从而消除额外晶体或振荡器的需求。休眠模块时钟源旨在为系统提供实时时钟源，以及为深度睡眠或休眠模式省电提供精确时钟源。

内部系统时钟(SysClk) 源于上述任一时钟源，同时增加两项：主内部PLL输出和四分频的精确内部振荡器 (4 MHz ± 1%)。PLL时钟参考频率必须在范围5 MHz 到 25 MHz (包括)内。表5-3（194页）显示了不同的时钟源如何在系统中使用。

表 **5-3.** 时钟源选项

时钟源	驱动**PLL**？		用作**SysClk**？	
精确内部振荡器	是	BYPASS = 0、OSCSRC = 0x1	是	BYPASS = 1、OSCSRC = 0x1
4分频的精确内部振荡器 (4MHz±1%)	否	-	是	BYPASS = 1、OSCSRC = 0x2
主振荡器	是	BYPASS = 0、OSCSRC = 0x0	是	BYPASS = 1、OSCSRC = 0x0
低频内部振荡器 (LFIOSC)	否	-	是	BYPASS = 1、OSCSRC = 0x3
休眠模块32.768-kHz振荡器	否	-	是	BYPASS = 1、OSCSRC2 = 0x7

5.2.5.2 时钟配置

运行模式时钟配置 (RCC) 寄存器和运行模式时钟配置 2 (RCC2) 寄存器提供对系统时钟的控制。RCC2 寄存器用来扩充域，提供了 RCC 寄存器之外的其它编码。使用时，RCC2 寄存器中域的值被 RCC 寄存器相应域的逻辑使用。值得一提的是，RCC2 提供了更多种类的时钟配置选项。这些寄存器控制下面的时钟功能：

- 睡眠和深度睡眠模式中的时钟源
- 源自PLL或其它时钟源的系统时钟
- 振荡器和PLL的使能/禁止
- 时钟分频
- 晶振输入选择

重要: 写 RCC 寄存器后再写 RCC2 寄存器。

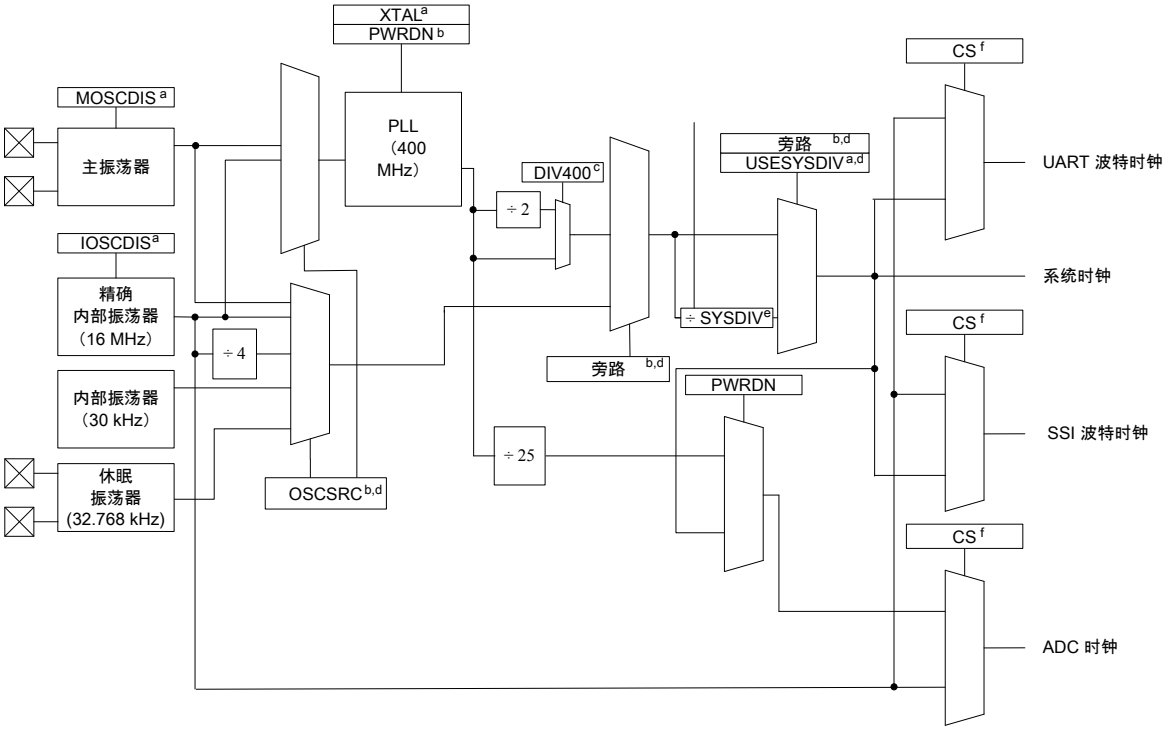
在将系统时钟的基础时钟源变更为 MOSC 时，必须先将 MOSCDIS 位置位，然后再重新选择 MOSC，否则可能偶尔会出现未定义的系统时钟配置。

系统时钟的配置在 EEPROM 操作过程中不可更改。软件必须等到 EEPROM 完成状态 (EEDONE) 寄存器中的 WORKING 位清零之后，才能对系统时钟做出更改。

图5-5 所示为主时钟树逻辑。外设框图由系统时钟信号驱动并且可以独立使能/禁止。如果 PLL 被禁用，可从 PIOSC 或系统时钟中选择 ADC 时钟信号，如果 PLL 被启用，可从分频至 16 MHz 的 PLL 输出中选择 ADC 时钟信号。

注意: 如果 ADC 模块未使用 PIOSC 作为时钟源，系统时钟必须至少为 16 MHz。

图 **5-5.** 主时钟树



注意：
- **a.** 由 RCC 寄存器的位/域提供控制。
- **b.** 由 RCC 寄存器的位/域提供控制，或者如果 RCC2 寄存器位 USERCC2 置位，由 RCC2 寄存器的位/域提供控制。
- **c.** 由 RCC2 寄存器的位/域提供控制。
- **d.** 在深度睡眠模式也可以被 DSLPCLKCFG 控制。
- **e.** 由 RCC 寄存器的 SYSDIV 域提供控制，如果 USERCC2 位置位，由 RCC2 寄存器的 SYSDIV2 域提供控制；如果 USERCC2 和 DIV400 位都置位，则由[SYSDIV2,SYSDIV2LSB]域提供控制。
- **f.** 由 UARTCC、SSICC 和 ADCCC 寄存器域提供控制。

通信时钟源

在上述的主时钟树之外，UART 和 SSI 模块在外设寄存器映射中都拥有时钟控制寄存器（偏移量 0xFC8），以用来选择模块波特率时钟的时钟源。用户可以在作为波特率时钟默认源的系统时钟和 PIOSC 之间进行选择。请注意在使用 PIOSC 作为波特率时，可能有特别注意事项。更多信息，请查看描述模块运行章节的时钟控制寄存器说明。

使用 **SYSDIV** 和 **SYSDIV2** 域

在 RCC 寄存器中，SYSDIV 域规定了哪个分频器用于产生源自 PLL 输出或振荡器源（取决于该寄存器中 BYPASS 位的配置）的系统时钟。当使用PLL时，在应用分频前400MHz的VCO频率先被2分频。表5-4显示了使用 PLL (BYPASS=0) 或另一个时钟源 (BYPASS=1) 时，SYSDIV 编码如何影响系统时钟频率。分频值等于 SYSDIV 编码加 1。表5-3（194页）列出了可能的时钟源。

表 **5-4.** 使用 **SYSDIV** 域可能实现的系统时钟频率

SYSDIV	分频系数	频率 **(BYPASS=0)**	频率 **(BYPASS=1)**	**TivaWare™** 参数符号[a]
0x0	/1	保留	时钟源频率/1	SYSCTL_SYSDIV_1
0x1	/2	保留	时钟源频率/2	SYSCTL_SYSDIV_2
0x2	/3	66.67 MHz	时钟源频率/3	SYSCTL_SYSDIV_3

表 **5-4.** 使用 **SYSDIV** 域可能实现的系统时钟频率（续）

SYSDIV	分频系数	频率 (BYPASS=0)	频率 (BYPASS=1)	TivaWare™ 参数符号[a]
0x3	/4	50 MHz	时钟源频率/4	SYSCTL_SYSDIV_4
0x4	/5	40 MHz	时钟源频率/5	SYSCTL_SYSDIV_5
0x5	/6	33.33 MHz	时钟源频率/6	SYSCTL_SYSDIV_6
0x6	/7	28.57 MHz	时钟源频率/7	SYSCTL_SYSDIV_7
0x7	/8	25 MHz	时钟源频率/8	SYSCTL_SYSDIV_8
0x8	/9	22.22 MHz	时钟源频率/9	SYSCTL_SYSDIV_9
0x9	/10	20 MHz	时钟源频率/10	SYSCTL_SYSDIV_10
0xA	/11	18.18 MHz	时钟源频率/11	SYSCTL_SYSDIV_11
0xB	/12	16.67 MHz	时钟源频率/12	SYSCTL_SYSDIV_12
0xC	/13	15.38 MHz	时钟源频率/13	SYSCTL_SYSDIV_13
0xD	/14	14.29 MHz	时钟源频率/14	SYSCTL_SYSDIV_14
0xE	/15	13.33 MHz	时钟源频率/15	SYSCTL_SYSDIV_15
0xF	/16	12.5 MHz（默认）	时钟源频率/16	SYSCTL_SYSDIV_16

a. 该参数在函数中使用，例如 TivaWare 外设驱动库中的 SysCtlClockSet()。

RCC2 寄存器中的 SYSDIV2 域比 RCC 寄存器中的 SYSDIV 域多两位，这样能提供更大的分频（最高可达 64 分频），同时允许使用更低的系统时钟频率以改善深度睡眠功耗。当使用PLL时，在应用分频前400MHz的VCO频率先被2分频。分频值等于 SYSDIV2 编码加 1。表5-5 显示了使用 PLL(BYPASS2=0)或另一个时钟源(BYPASS2=1)时，SYSDIV2编码如何影响系统时钟频率。表 5-3（194页）列出了可能的时钟源。

表 **5-5.** 使用 **SYSDIV2** 域可能实现的系统时钟频率示例

SYSDIV2	分频系数	频率 (BYPASS2=0)	频率 (BYPASS2=1)	TivaWare 参数符号[a]
0x00	/1	保留	时钟源频率/1	SYSCTL_SYSDIV_1
0x01	/2	保留	时钟源频率/2	SYSCTL_SYSDIV_2
0x02	/3	66.67 MHz	时钟源频率/3	SYSCTL_SYSDIV_3
0x03	/4	50 MHz	时钟源频率/4	SYSCTL_SYSDIV_4
0x04	/5	40 MHz	时钟源频率/5	SYSCTL_SYSDIV_5
...	...	...	...	...
0x09	/10	20 MHz	时钟源频率/10	SYSCTL_SYSDIV_10
...	...	...	...	...
0x3F	/64	3.125 MHz	时钟源频率/64	SYSCTL_SYSDIV_64

a. 该参数在函数中使用，例如 TivaWare 外设驱动库中的 SysCtlClockSet()。

为了使用 PLL 时能有更多的频率选择，器件提供了 DIV400 位和 SYSDIV2LSB 位。当 DIV400 位置位时，第 22 位变成了SYSDIV2 的 LSB。此时，分频值等于 SYSDIV2 编码加 SYSDIV2LSB 再加 1。表5-6显示了DIV400置位后的频率选择。当 DIV400 被清零后，SYSDIV2LSB 将被忽略，系统时钟频率仍然取决于表5-5（196页）所示的频率值。

表 **5-6.** 当 **DIV400=1** 时可能实现的系统时钟频率示例

SYSDIV2	SYSDIV2LSB	分频系数	频率 (BYPASS2=0)[a]	TivaWare 参数符号[b]
0x00	保留	/2	保留	-
0x01	0	/3	保留	-
	1	/4	保留	-

表 5-6. 当 DIV400=1 时可能实现的系统时钟频率示例（续）

SYSDIV2	SYSDIV2LSB	分频系数	频率 (BYPASS2=0)[a]	TivaWare 参数符号[b]
0x02	0	/5	80 MHz	SYSCTL_SYSDIV_2_5
	1	/6	66.67 MHz	SYSCTL_SYSDIV_3
0x03	0	/7	保留	-
	1	/8	50 MHz	SYSCTL_SYSDIV_4
0x04	0	/9	44.44 MHz	SYSCTL_SYSDIV_4_5
	1	/10	40 MHz	SYSCTL_SYSDIV_5
...	...	...	...	...
0x3F	0	/127	3.15 MHz	SYSCTL_SYSDIV_63_5
	1	/128	3.125 MHz	SYSCTL_SYSDIV_64

a. 注意：只有当 BYPASS2=0 时，DIV400 和 SYSDIV2LSB 才有效。

b. 该参数在函数中使用，例如 TivaWare 外设驱动库中的 SysCtlClockSet()。

5.2.5.3 精确内部振荡器操作 (PIOSC)

微控制器上电后运行 PIOSC。如果希望运行其它时钟源，PIOSC 在用于内部功能时必须保持启用。仅可以在深度睡眠模式期间禁用 PIOSC。它可以通过置位 DSLPCLKCFG 寄存器中的 PIOSCPD 位掉电。

PIOSC 生成一个 16-MHz 时钟，其校准精度为 ±1%，整个温度范围内的精度为 ±3%（参见“PIOSC Specifications”（1045页））。在工厂，PIOSC 被设置为 16 MHz。但是，在其它的电压或温度条件下，该频率可以通过软件用下面的三种方法来调整：

- 默认校准：在精确内部振荡器校准 (PIOSCCAL) 寄存器中清零 UTEN 位并置位 UPDATE 位。
- 用户自定义校准：用户可以通过设置 UT 值来调整 PIOSC 频率。随着 UT 值的增大，生成的周期也将增大。为了提交一个新的 UT 值，首先要置位 UTEN 位，接着编程 UT 域，然后置位 UPDATE 位。该调整在几个时钟周期内完成，然后频率突变。
- 使用具有正在工作的 32.768-kHz 时钟源的休眠模块进行自动校准：置位 PIOSCCAL 寄存器中的 CAL 位；校准的结果显示在精确内部振荡器统计 (PIOSCSTAT) 寄存器的 RESULT 域中。校准完成后，使用 CT 域中返回的调整值来调整 PIOSC。

5.2.5.4 用于主振荡器的晶振配置 (MOSC)

主振荡器支持的晶振值：从 4 到 25 MHz。

RCC 寄存器中的 XTAL 位（见221页）描述了可用的晶振选择和默认的编程值。

软件根据晶振值来配置 RCC 寄存器的 XTAL 域。如果设计中使用 PLL，那么 XTAL 域中的值会从内部转换到 PLL 设置中。

5.2.5.5 主PLL 频率配置

主PLL在上电复位期间默认为禁止，如果需要可在稍后通过软件使能。软件指定输出分频来设置系统时钟频率同时使能主PLL来驱动输出。PLL 以 400 MHz 的频率工作，但是在输出分频应用前将先被二分频，除非 RCC2 寄存器中的 DIV400 位置位。

要把 PIOSC 配置为主 PLL 的时钟源，可将运行模式时钟配置 2 (RCC2) 寄存器的 OSCRC2 域编程为 0x1。

如果主振荡器给主 PLL 提供了时钟基准，软件可以从 PLL 频率 n (PLLFREQn) 寄存器（请参考237页）中使用由硬件提供的转换（该转换通常用于对 PLL 进行编程）。内部转换可提供± 1% 的目标 PLL VCO 频率。表21-14（1044页） 显示了实际的PLL频率和给定晶振选择的误差。

在运行模式时钟配置(RCC)寄存器（请参考221页）中的晶振值域(XTAL)描述了可用的晶振选择和默认的 PLLFREQn 寄存器编程。只要 XTAL 域改变，新的设置就会被转换，同时内部 PLL 设置被更新。

5.2.5.6 PLL 模式

- 普通:PLL 倍频输入时钟参考并驱动输出。
- 掉电模式:大部分PLL内部电路被禁止并且PLL不再驱动输出。

使用RCC/RCC2寄存器的域来编程PLL模式（见221页和227页）。

5.2.5.7 PLL 操作

如果 PLL 配置改变，PLL输出频率是不稳定的，直到重新集中(重新锁定)到一个新的设定为止。配置更改和重新锁定之间的时间是 T_{READY}（请参考表21-13（1044页））。在重新锁定期间，受影响的PLL不可用作时钟参考。软件可以查询 PLL 状态 (PLLSTAT) 寄存器的 LOCK 位以确定 PLL 锁定的时间。

PLL 可通过以下方法之一进行更改：

- 更改为 RCC 寄存器的 XTAL 值 - 写入相同的值不会导致重新锁定。
- 使PLL从掉电模式变为正常模式。

使用根据系统时钟计时的计数器测量 T_{READY} 的要求。如果 PLL 上电，递减计数器初值设为 0x200。如果 PLLFREQn 寄存器中的 M 或 N 值发生改变，则计数器将设为 0xC0。此时要提供硬件来保持 PLL 不被用作系统时钟，直到上述更改完成并满足 T_{READY} 条件为止。用户要确保在 RCC/RCC2 寄存器切换到使用 PLL 之前必须有一个稳定的时钟源（例如主振荡器）。

如果主 PLL 启用并且系统时钟一步切换到使用 PLL，系统控制硬件会继续使用 RCC/RCC2 寄存器选择的振荡器作为微控制器的时钟，直到主 PLL 稳定（满足 T_{READY} 时间）为止才更改到 PLL。软件可以使用很多方法来确保系统由主 PLL 提供时钟，包括周期性地检测原始中断状态 (RIS) 寄存器的 PLLLRIS 位，并且启用 PLL 锁定中断。

5.2.5.8 主振荡器校验电路

时钟控制包括能确保主振荡器工作在合适频率的电路。如果频率在相连晶振允许的频带范围外，该电路会监测主振荡器频率和信号。

检测电路通过主振荡器控制 (MOSCCTL) 寄存器的 CVAL位来启用。当该电路启用且检测到错误，同时 MOSCCTL 寄存器中的 MOSCIM 位清零时，硬件会执行下面的序列：

1. 复位原因 (RESC) 寄存器的 MOSCFAIL 位被置位。
2. 系统时钟由主振荡器切换到PIOSC。
3. 一个内部上电复位被启动。
4. 复位不再有效，处理器在复位序列期间直接执行 NMI 处理程序。

如果 MOSCCTL 寄存器中的 MOSCIM 位置位，那么硬件会执行下面的序列：

1. 系统时钟由主振荡器切换到PIOSC。
2. RIS 寄存器中的 MOFRIS 位置位以指示 MOSC 故障。

5.2.6 系统控制

为了节省功耗，外设专用的 RCGCx、SCGCx 和 DCGCx寄存器（例如，RCGCWD）分别控制着微控制器在运行模式、睡眠模式和深度睡眠模式时该外设或系统模块的时钟门控逻辑。这些寄存器位于系统控制寄存器映射中，分别从偏移量 0x600、0x700 和 0x800 处开始。在访问任何模块寄存器前，在 RCGC 寄存器中启用外设模块时钟后必须有 3 个系统时钟的延迟。

重要: 为了支持传统软件，RCGCn、SCGCn 和 DCGCn 寄存器在偏移量为 0x100 - 0x128 时可用。对所有这些传统寄存器进行写操作时，也将对外设专用 RCGCx、SCGCx 和 DCGCx 寄存器中的相应位执行写操作。软件必须使用外设专用寄存器以支持不处于传统寄存器中的模块。建议新软件使用新寄存器并且不依赖于传统操作。

如果软件使用外设专用寄存器对传统外设（如TIMER0）执行写操作，则写操作会产生正确操作，但是该位的值不在传统寄存器中得到反映。通过对传统寄存器的写操作更改的任何位都可以通过对传统寄存器的读操作进行正确回读。如果软件使用传统和外设专用寄存器访问，则必须通过读-修改-写的操作来访问外设专用寄存器，因为该操作仅影响不在传统寄存器中的外设。通过这种方法，外设专用和传统寄存器都具有一致的信息。

微控制器有 4 种级别的操作，定义如下：

- 运行模式
- 睡眠模式
- 深度睡眠模式
- 休眠模式

下面章节将详细描述不同的模式。

小心 **–** 如果 **Cortex-M4F** 调试访问端口 **(DAP)** 已经启用，并且设备从低功耗睡眠或深度睡眠模式中唤醒，内核可能会在所有外设的时钟恢复到运行模式配置前开始执行代码。**DAP**通常由软件工具使能，以便在调试或**Flash**编程时访问**JTAG**或**SWD**接口。如果这种情况发生，当软件访问一个带有无效时钟的外设时会触发一次硬件错误。

软件延时循环可用在从**WFI**指令**(**等待中断**)**唤醒系统的中断程序开始处。这样可以延迟执行可能会产生错误的访问外设寄存器的指令。对于产品软件来说，该循环可以去除，因为在正常执行期间**DAP**几乎不肯能被使能。

由于**DAP**默认为禁止 **(**上电复位**)** ，所以用户也可以让设备按上电周期运行。**DAP** 只有通过 **JTAG** 或 **SWD** 接口才能启用。

5.2.6.1 运行模式

在运行模式，微控制器主动执行代码。运行模式提供了处理器和所有目前被外设专用RCGC 寄存器启用的外设的正常操作。系统时钟可以是包括PLL在内的任何可用的时钟源。

5.2.6.2 睡眠模式

在睡眠模式，运行中的外设时钟频率不变，但是处理器和存储器子系统不使用时钟，所以不再执行代码。睡眠模式是通过 Cortex-M4F 内核执行一条 WFI （等待中断）指令。系统中任何正确配置的中断事件都可以将处理器带回到运行模式。更多详细信息请参阅“电源管理”（93页）。

当自动时钟门控启用时，外设专用 SCGC 寄存器启用的外设时钟被使用（请参考 RCC 寄存器）；当自动时钟门控禁用时，外设专用RCGC 寄存器启用的外设时钟被使用。系统时钟的源和频率与运行模式期间是一样的。

还提供其他睡眠模式，可降低 SRAM 和 Flash 存储器的功耗。但是，较低功耗的模式的睡眠和唤醒速度较慢，请参见“动态电源管理”（201页）了解更多信息。

重要: 在执行 WFI 指令前，软件必须检查 EEPROM 完成状态 (EEDONE) 寄存器的 WORKING 位是否清零，以确保 EEPROM 不忙。

5.2.6.3 深度睡眠模式

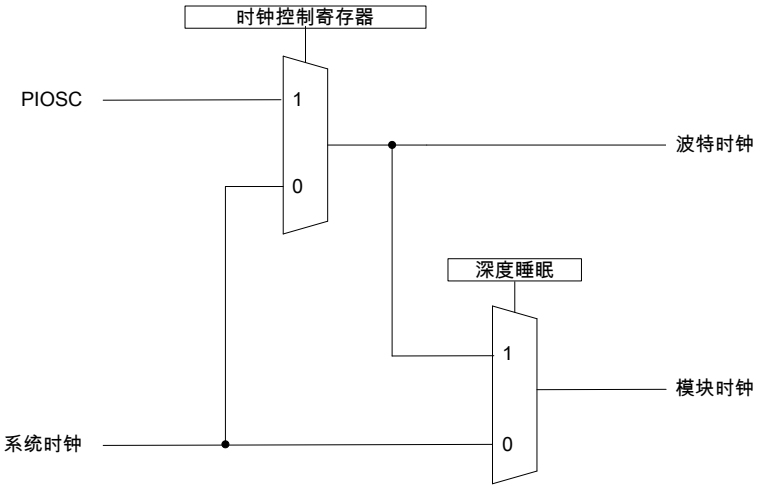
在深度睡眠模式中，除了正在停止的处理器时钟之外，有效外设的时钟频率可以改变（取决于深度睡眠模式的时钟配置）。中断可以让微控制器从睡眠模式返回到运行模式；代码请求可以进入睡眠模式。要进入深度睡眠模式，首先置位系统控制 (SYSCTRL) 寄存器的 SLEEPDEEP 位（见143页），然后执行一条 WFI 指令。系统中任何正确配置的中断事件都可以将处理器带回到运行模式。更多详细信息请参阅“电源管理”（93页）。

这些 Cortex-M4F 处理器内核和存储器子系统在深度睡眠模式中不计时。当自动时钟门控启用时，外设专用 DSCGC 寄存器启用的外设时钟被使用（请参考 RCC 寄存器）；当自动时钟门控禁用时，外设专用 RCGC 寄存器启用的外设时钟被使用。系统时钟源在寄存器 DSLPCLKCFG 中规定。当使用 DSLPCLKCFG 寄存器时，内部振荡器源将上电，如果有必要，其它时钟会掉电。如果执行 WFI 指令时 PLL 正在运行，硬件会让 PLL 掉电并将 RCC/RCC2 寄存器中的 SYSDIV 域分别改写为 /16 或 /64，具体由 DSLPCLKCFG 寄存器的 DSDIVORIDE 设置决定。执行 WFI 指令时，USB PLL 不会掉电当深度睡眠退出事件发生时，硬件会把系统时钟带回到深度睡眠模式开始时的源和频率，然后使能在深度睡眠期间停止的时钟。如果 PIOSC 被用作PLL的参考时钟源，在深度睡眠期间它会继续提供时钟。请参阅231页。

重要: 在执行 WFI 指令前，软件必须检查 EEPROM 完成状态 (EEDONE) 寄存器的 WORKING 位是否清零，以确保 EEPROM 不忙。

要实现尽可能低的深度睡眠功耗，以及无需为时钟更改而重新配置外设即可从外设唤醒处理器的能力，一些通信模块在模式寄存器空间中的偏移量0xFC8处提供了时钟控制寄存器。时钟控制寄存器中CS域允许用户选择PIOSC作为模块波特率时钟的时钟源。微控制器进入深度睡眠模式时，PIOSC也成为模块时钟的源，允许发送和接收 FIFO 在该部分处于深度睡眠时继续操作。图5-6（200页）显示了如何选择时钟。

图 **5-6.** 模块时钟选择



还提供其他深度睡眠模式，可降低SRAM和Flash存储器的功耗。但是，较低功耗的模式的深度睡眠和唤醒速度较慢，请参见“动态电源管理”（201页）了解更多信息。

5.2.6.4 动态电源管理

除睡眠和深度睡眠模式以及片上模块的时钟门控之外，还提供几种功率模式选择，以在睡眠和深度睡眠模式时让 LDO、Flash 存储器和 SRAM 进入不同级别的节能模式。注：这些功能可能不适用于所有器件；系统属性 (SYSPROP) 寄存器提供关于给定 MCU 是否支持某一模式的信息。以下寄存器提供如下功能：

- LDO 睡眠功率控制 (LDOSPCTL)：控制睡眠模式中的 LDO 值
- LDO 深度睡眠功率控制 (LDODPCTL)：控制深度睡眠模式中的 LDO 值
- LDO 睡眠电源校准 (LDOSPCAL)：为睡眠模式中的 LDO 值提供工厂建议
- LDO 深度睡眠电源校准 (LDODPCAL)：为深度睡眠模式中的 LDO 值提供工厂建议
- 睡眠功率配置寄存器 (SLPPWRCFG)：控制睡眠模式下，Flash 存储器和 SRAM 的节电模式
- 睡眠功率配置寄存器 (SLPPWRCFG)：控制深度睡眠模式下，Flash 存储器和 SRAM 的节电模式
- 深度睡眠时钟配置 (DSLPCLKCFG)：控制深度睡眠模式的计时
- 睡眠/深度睡眠功率模式状态 (SDPMST)：提供关于不同省电事件的状态信息

LDO 功率控制

注意: 当通过 JTAG 连接器件时，睡眠或深度睡眠的 LDO 控制设置不可用，且不会被应用。

用户可以使用 LDOSPCTL 寄存器（见242页）或 LDODPCTL 寄存器（见245页）动态请求提高或降低 LDO 电压水平以平衡功率/性能。要降低 LDO 电平，软件必须在 RCC/RCC2（适用于睡眠模式）或 DSLPCLKCFG（适用于深度睡眠模式）中为降低后的 LDO 值配置系统时钟，然后再请求降低 LDO。

LDO 功率校准寄存器 LDOSPCAL 和 LDODPCAL 为不同模式的 LDO 提供建议的值。如果软件请求的 LDO 值太低或太高，则不能接受该值并且会在 SDPMST 寄存器中报告该错误。

下表显示了相对于配置的 LDO 电压的最高系统时钟频率和 PIOSC 最高频率。

工作电压 **(LDO)**	最高系统时钟频率	**PIOSC**
1,2	80 MHz	16 MHz
0,9	20 MHz	16 MHz

Flash 存储器和 SRAM 功率控制

在睡眠或深度睡眠模式期间，Flash 存储器可以处于默认的主动模式或低功率模式；SRAM 可以处于默认的主动模式、待机模式或低功率模式。各种情况中的主动模式可为睡眠和唤醒提供最快的速度，但是消耗更多能量。低功率模式提供最低功耗，但是睡眠或唤醒需要较长时间。

配置系统属性 (SYSPROP) 寄存器的 SRAMSM 位可使 SRAM 禁用所有电源管理功能。此配置的工作方式与传统 Stellaris® 器件一样，可提供最快的睡眠和唤醒速度，但是在睡眠和深度睡眠模式中消耗的功率最大。其他电源选项均为保留模式和具有较低 SRAM 电压的保留模式。具有较低 SRAM 电压的 SRAM 保留模式提供最低功耗，但是睡眠或唤醒需要最长时间。可以使用 SLPPWRCFG 和 DSLPPWRCFG 寄存器针对 Flash 寄存器和 SRAM 对这些模式进行单独配置。

以下节能选项在睡眠和深度睡眠模式中可用：

- 可以根据外设专用 SCGC 或 DCGC 寄存器中的设置为时钟安装门控。

- 在深度睡眠模式中，可以使用 DSLPCLKCFG 寄存器更改时钟源并关闭 PIOSC 的电源（如果没有活动外设需要使用它）。以下选项在睡眠模式中不可用：
- 使用 LDOSPCTL 或 LDODPCTL 寄存器可以更改 LDO 电压。
- 可以将 Flash 存储器置入低功率模式。
- 可以将 SRAM 置入待机或低功率模式。

SDPMST 寄存器提供发出动态电源管理指令后的结果。如果它在运行，还提供可以通过调试器或内核查看的一些实时状态。这些事件不会触发中断，仅提供信息以帮助调整电源管理软件。状态寄存器在每个提供错误检测的动态电源管理事件请求开始时被写入。尚无机制清零这些位；它们在下一个事件时被覆盖。实时提供实时数据，并无事件记录该信息。

5.2.6.5 休眠模式

这种模式下，微控制器主要功能部件的电源关断，只有休眠模块的电路在工作。需要一个外部唤醒事件或RTC事件来使微控制器回到运行模式。然后 Cortex-M4F 处理器和休眠模块之外的外设经历一个正常的“上电”序列，处理器开始运行代码。软件通过检查休眠模块寄存器可以确定微控制器是否已经从休眠模式重新启动。关于休眠模式操作的更多信息，请参考“休眠模块”（429页）。

5.3 初始化和配置

对 PLL 的配置可通过直接对 RCC/RCC2 寄存器执行写操作来实现。如果RCC2寄存器正被使用，那么必须置位USERCC2位，合适的RCC2位/域被使用。成功改变基于PLL的系统时钟所需的步骤如下：

1. 通过置位 RCC 寄存器的 BYPASS 位并清零 USESYS 位来旁路 PLL 和系统时钟分频器，从而配置微控制器在“原始”时钟源运行，在切换系统时钟到 PLL 前允许新的 PLL 配置有效。
2. 选择晶振值 (XTAL) 和振荡器源 (OSCSRC)，清零 RCC/RCC2 的 PWRDN 位。设置 XTAL 域可以自动为相应晶振提供有效的 PLL 配置数据，清零 PWRDN 位可以给 PLL 及其输出供电并启用它们。
3. 在 RCC/RCC2 中选择需要的系统分频器 (SYSDIV) 并在 RCC 中置位 USESYS 位。SYSDIV 域决定了微控制器的系统频率。
4. 通过查询原始中断状态 (RIS) 寄存器的 PLLLRIS 位来等待 PLL 被锁定。
5. 通过清零 RCC/RCC2 的 BYPASS 位来启用 PLL。

5.4 寄存器映射

表 5-7（203页）列出了按功能分组的系统控制寄存器。列出的偏移量是相对于系统控制基址 0x400F.E000而言的寄存器地址的 16 进制增量。

注意：系统控制寄存器空间中未被使用的空间都是为将来或内部使用保留的。软件不得修改任何保留的存储器地址。

有关系统控制寄存器空间中定义的其它 Flash 和 ROM 寄存器的描述，请参阅“内部存储器”（458页）。

表 5-7. 系统控制 寄存器映射

偏移量	名称	类型	复位	描述	见页面
系统控制寄存器					
0x000	DID0	RO	-	器件标识 0	208
0x004	DID1	RO	-	器件标识寄存器 1	210
0x030	PBORCTL	R/W	0x0000.7FFF	掉电复位控制	212
0x050	RIS	RO	0x0000.0000	原始中断状态	213
0x054	IMC	R/W	0x0000.0000	中断屏蔽控制	215
0x058	MISC	R/W1C	0x0000.0000	屏蔽的中断状态和清除	217
0x05C	RESC	R/W	-	复位原因	219
0x060	RCC	R/W	0x0780.3AD1	运行模式时钟配置	221
0x06C	GPIOHBCTL	R/W	0x0000.7E00	GPIO 高性能总线控制	225
0x070	RCC2	R/W	0x07C0.6810	运行模式时钟配置 2	227
0x07C	MOSCCTL	R/W	0x0000.0000	主振荡器控制	230
0x144	DSLPCLKCFG	R/W	0x0780.0000	深度睡眠时钟配置	231
0x14C	SYSPROP	RO	0x0000.1D31	系统属性寄存器	233
0x150	PIOSCCAL	R/W	0x0000.0000	精确内部振荡器校准	235
0x154	PIOSCSTAT	RO	0x0000.0040	精确内部振荡器统计寄存器	236
0x160	PLLFREQ0	RO	0x0000.0032	PLL 频率寄存器 0	237
0x164	PLLFREQ1	RO	0x0000.0001	PLL 频率寄存器 1	238
0x168	PLLSTAT	RO	0x0000.0000	PLL 状态寄存器	239
0x188	SLPPWRCFG	R/W	0x0000.0000	睡眠功率配置寄存器	240
0x18C	DSLPPWRCFG	R/W	0x0000.0000	深度睡眠功率配置寄存器	241
0x1B4	LDOSPCTL	R/W	0x0000.0018	LDO 睡眠功率控制寄存器	242
0x1B8	LDOSPCAL	RO	0x0000.1818	LDO 睡眠功率校准寄存器	244
0x1BC	LDODPCTL	R/W	0x0000.0012	LDO 深度睡眠功率控制寄存器	245
0x1C0	LDODPCAL	RO	0x0000.1212	LDO 深度睡眠功率校准寄存器	247
0x1CC	SDPMST	RO	0x0000.0000	睡眠/深度睡眠功率模式状态寄存器	248
0x300	PPWD	RO	0x0000.0003	看门狗定时器外设存在寄存器	251
0x304	PPTIMER	RO	0x0000.003F	16/32 位通用定时器外设存在寄存器	252
0x308	PPGPIO	RO	0x0000.003F	通用输入/输出外设存在寄存器	254
0x30C	PPDMA	RO	0x0000.0001	微型直接存储器访问外设存在寄存器	257
0x314	PPHIB	RO	0x0000.0001	休眠外设存在寄存器	258
0x318	PPUART	RO	0x0000.00FF	通用异步收发器外设存在寄存器	259
0x31C	PPSSI	RO	0x0000.000F	同步串行接口外设存在寄存器	261

表 **5-7.** 系统控制 寄存器映射（续）

偏移量	名称	类型	复位	描述	见页面
0x320	PPI2C	RO	0x0000.000F	内部集成电路外设存在寄存器	262
0x328	PPUSB	RO	0x0000.0000	通用串行总线外设存在寄存器	264
0x334	PPCAN	RO	0x0000.0001	控制器局域网外设存在寄存器	265
0x338	PPADC	RO	0x0000.0003	模数转换器外设存在寄存器	266
0x33C	PPACMP	RO	0x0000.0001	模拟比较器外设存在寄存器	267
0x340	PPPWM	RO	0x0000.0000	脉宽调解器外设存在寄存器	268
0x344	PPQEI	RO	0x0000.0000	正交编码器接口外设存在寄存器	269
0x358	PPEEPROM	RO	0x0000.0001	EEPROM 外设存在寄存器	270
0x35C	PPWTIMER	RO	0x0000.003F	32/64 位宽通用定时器外设存在寄存器	271
0x500	SRWD	R/W	0x0000.0000	看门狗定时器软件复位寄存器	273
0x504	SRTIMER	R/W	0x0000.0000	16/32 位通用定时器软件复位寄存器	274
0x508	SRGPIO	R/W	0x0000.0000	通用输入/输出软件复位寄存器	276
0x50C	SRDMA	R/W	0x0000.0000	微型直接存储器访问软件复位寄存器	278
0x514	SRHIB	R/W	0x0000.0000	休眠软件复位寄存器	279
0x518	SRUART	R/W	0x0000.0000	通用异步收发器软件复位寄存器	280
0x51C	SRSSI	R/W	0x0000.0000	同步串行接口软件复位寄存器	282
0x520	SRI2C	R/W	0x0000.0000	内部集成电路软件复位寄存器	284
0x534	SRCAN	R/W	0x0000.0000	控制器局域网软件复位寄存器	286
0x538	SRADC	R/W	0x0000.0000	模数转换器软件复位寄存器	287
0x53C	SRACMP	R/W	0x0000.0000	模数比较器软件复位寄存器	288
0x558	SREEPROM	R/W	0x0000.0000	EEPROM 软件复位寄存器	289
0x55C	SRWTIMER	R/W	0x0000.0000	32/64 位宽通用定时器软件复位寄存器	290
0x600	RCGCWD	R/W	0x0000.0000	看门狗定时器运行模式时钟门控控制寄存器	292
0x604	RCGCTIMER	R/W	0x0000.0000	16/32 位通用定时器运行模式时钟门控控制寄存器	293
0x608	RCGCGPIO	R/W	0x0000.0000	通用输入/输出运行模式时钟门控控制寄存器	295
0x60C	RCGCDMA	R/W	0x0000.0000	微型直接存储器访问运行模式时钟门控控制寄存器	297
0x614	RCGCHIB	R/W	0x0000.0001	休眠运行模式时钟门控控制寄存器	298
0x618	RCGCUART	R/W	0x0000.0000	通用异步收发器运行模式时钟门控控制寄存器	299
0x61C	RCGCSSI	R/W	0x0000.0000	同步串行接口运行模式时钟门控控制寄存器	301
0x620	RCGCI2C	R/W	0x0000.0000	内部集成电路运行模式时钟门控控制寄存器	303
0x634	RCGCCAN	R/W	0x0000.0000	控制器局域网运行模式时钟门控控制寄存器	305
0x638	RCGCADC	R/W	0x0000.0000	模数转换器运行模式时钟门控控制寄存器	306
0x63C	RCGCACMP	R/W	0x0000.0000	模拟比较器运行模式时钟门控控制寄存器	307

表 5-7. 系统控制 寄存器映射（续）

偏移量	名称	类型	复位	描述	见页面
0x658	RCGCEEPROM	R/W	0x0000.0000	EEPROM 运行模式时钟门控控制寄存器	308
0x65C	RCGCWTIMER	R/W	0x0000.0000	32/64 位宽通用定时器运行模式时钟门控控制寄存器	309
0x700	SCGCWD	R/W	0x0000.0000	看门狗定时器睡眠模式时钟门控控制寄存器	311
0x704	SCGCTIMER	R/W	0x0000.0000	16/32 位通用定时器睡眠模式时钟门控控制寄存器	312
0x708	SCGCGPIO	R/W	0x0000.0000	通用输入/输出睡眠模式时钟门控控制寄存器	314
0x70C	SCGCDMA	R/W	0x0000.0000	微型直接存储器访问睡眠模式时钟门控控制寄存器	316
0x714	SCGCHIB	R/W	0x0000.0001	休眠睡眠模式时钟门控控制寄存器	317
0x718	SCGCUART	R/W	0x0000.0000	通用异步收发器睡眠模式时钟门控控制寄存器	318
0x71C	SCGCSSI	R/W	0x0000.0000	同步串行接口睡眠模式时钟门控控制寄存器	320
0x720	SCGCI2C	R/W	0x0000.0000	内部集成电路睡眠模式时钟门控控制寄存器	322
0x734	SCGCCAN	R/W	0x0000.0000	控制器局域网睡眠模式时钟门控控制寄存器	324
0x738	SCGCADC	R/W	0x0000.0000	模数转换器睡眠模式时钟门控控制寄存器	325
0x73C	SCGCACMP	R/W	0x0000.0000	模拟比较器睡眠模式时钟门控控制寄存器	326
0x758	SCGCEEPROM	R/W	0x0000.0000	EEPROM 睡眠模式时钟门控控制寄存器	327
0x75C	SCGCWTIMER	R/W	0x0000.0000	32/64 位宽通用定时器睡眠模式时钟门控控制寄存器	328
0x800	DCGCWD	R/W	0x0000.0000	看门狗定时器深度睡眠模式时钟门控控制寄存器	330
0x804	DCGCTIMER	R/W	0x0000.0000	16/32 位通用定时器深度睡眠模式时钟门控控制寄存器	331
0x808	DCGCGPIO	R/W	0x0000.0000	通用输入/输出深度睡眠模式时钟门控控制寄存器	333
0x80C	DCGCDMA	R/W	0x0000.0000	微型直接存储器访问深度睡眠模式时钟门控控制寄存器	335
0x814	DCGCHIB	R/W	0x0000.0001	休眠深度睡眠模式时钟门控控制寄存器	336
0x818	DCGCUART	R/W	0x0000.0000	通用异步收发器深度睡眠模式时钟门控控制寄存器	337
0x81C	DCGCSSI	R/W	0x0000.0000	同步串行接口深度睡眠模式时钟门控控制寄存器	339
0x820	DCGCI2C	R/W	0x0000.0000	内部集成电路深度睡眠模式时钟门控控制寄存器	341
0x834	DCGCCAN	R/W	0x0000.0000	控制器局域网深度睡眠模式时钟门控控制寄存器	343
0x838	DCGCADC	R/W	0x0000.0000	模数转换器深度睡眠模式时钟门控控制寄存器	344
0x83C	DCGCACMP	R/W	0x0000.0000	模拟比较器深度睡眠模式时钟门控控制寄存器	345
0x858	DCGCEEPROM	R/W	0x0000.0000	EEPROM 深度睡眠模式时钟门控控制寄存器	346
0x85C	DCGCWTIMER	R/W	0x0000.0000	32/64 位宽通用定时器深度睡眠模式时钟门控控制寄存器	347
0xA00	PRWD	RO	0x0000.0000	看门狗定时器外设就绪寄存器	349
0xA04	PRTIMER	RO	0x0000.0000	16/32 位通用定时器外设就绪寄存器	350
0xA08	PRGPIO	RO	0x0000.0000	通用输入/输出外设就绪寄存器	352
0xA0C	PRDMA	RO	0x0000.0000	微型直接存储器访问外设就绪寄存器	354
0xA14	PRHIB	RO	0x0000.0001	休眠外设就绪寄存器	355

表 **5-7.** 系统控制 寄存器映射（续）

偏移量	名称	类型	复位	描述	见页面
0xA18	PRUART	RO	0x0000.0000	通用异步收发器外设就绪寄存器	356
0xA1C	PRSSI	RO	0x0000.0000	同步串行接口外设就绪寄存器	358
0xA20	PRI2C	RO	0x0000.0000	内部集成电路外设就绪寄存器	360
0xA34	PRCAN	RO	0x0000.0000	控制器局域网外设就绪寄存器	362
0xA38	PRADC	RO	0x0000.0000	模数转换器外设就绪寄存器	363
0xA3C	PRACMP	RO	0x0000.0000	模拟比较器外设就绪寄存器	364
0xA58	PREEPROM	RO	0x0000.0000	EEPROM 外设就绪寄存器	365
0xA5C	PRWTIMER	RO	0x0000.0000	32/64 位宽通用定时器外设就绪寄存器	366
系统控制传统寄存器					
0x008	DC0	RO	0x007F.007F	器件功能寄存器 0	368
0x010	DC1	RO	0x1103.2FFF	器件功能寄存器 1	370
0x014	DC2	RO	0x030F.F037	器件功能寄存器 2	373
0x018	DC3	RO	0xBFFF.0FC0	器件功能寄存器 3	375
0x01C	DC4	RO	0x0004.F03F	器件功能寄存器 4	379
0x020	DC5	RO	0x0000.0000	器件功能寄存器 5	381
0x024	DC6	RO	0x0000.0000	器件功能寄存器 6	383
0x028	DC7	RO	0xFFFF.FFFF	器件功能寄存器 7	384
0x02C	DC8	RO	0x0FFF.0FFF	器件功能寄存器 8	387
0x040	SRCR0	RO	0x0000.0000	软件复位控制寄存器 0	390
0x044	SRCR1	RO	0x0000.0000	软件复位控制寄存器 1	392
0x048	SRCR2	RO	0x0000.0000	软件复位控制寄存器 2	394
0x100	RCGC0	RO	0x0000.0040	运行模式时钟门控控制寄存器 0	396
0x104	RCGC1	RO	0x0000.0000	运行模式时钟门控控制寄存器 1	399
0x108	RCGC2	RO	0x0000.0000	运行模式时钟门控控制寄存器 2	402
0x110	SCGC0	RO	0x0000.0040	睡眠模式时钟门控控制寄存器 0	404
0x114	SCGC1	RO	0x0000.0000	睡眠模式时钟门控控制寄存器 1	406
0x118	SCGC2	RO	0x0000.0000	睡眠模式时钟门控控制寄存器 2	409
0x120	DCGC0	RO	0x0000.0040	深度睡眠模式时钟门控控制寄存器 0	411
0x124	DCGC1	RO	0x0000.0000	深度睡眠模式时钟门控控制寄存器 1	413
0x128	DCGC2	RO	0x0000.0000	深度睡眠模式时钟门控控制寄存器 2	416
0x190	DC9	RO	0x00FF.00FF	器件功能寄存器 9	418
0x1A0	NVMSTAT	RO	0x0000.0001	非易失性存储器信息寄存器	420

5.5 系统控制寄存器描述

所有给出的地址都是相对于 0x400F.E000 的系统控制基址而言的。“系统控制传统寄存器描述”（367页）中列出了仅为支持传统软件而提供的寄存器。

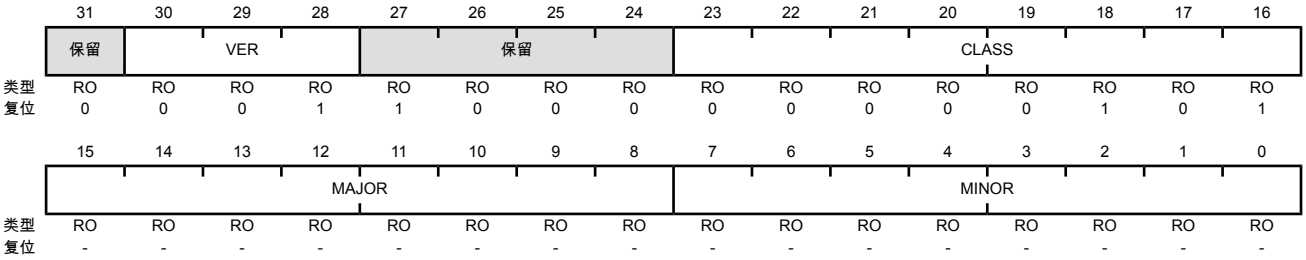
寄存器 1: 器件标识 0（DID0），偏移量 0x000

本寄存器标识了微控制器的版本。每个微控制器由 DID0 寄存器中的 CLASS 域和 DID1 寄存器中的 PARTNO 域的值组合起来进行专门标识。MAJOR 和 MINOR 位域标识了模具版本号。MAJOR 和 MINOR 位域共同标识了器件版本号。

MAJOR 位域值	MINOR 位域值	模具版本	器件版本
0x0	0x0	A0	1
0x0	0x1	A1	2
0x0	0x2	A2	3
0x0	0x3	A3	4
0x1	0x0	B0	5
0x1	0x1	B1	6
0x1	0x2	B2	7

器件标识 0 (DID0)

基址 0x400F.E000
偏移量 0x000
类型 RO, 复位 -



位/域	名称	类型	复位	描述
31	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
30:28	VER	RO	0x01	DID0版本 该域定义了 DID0 寄存器格式的版本。版本号是数字的。VER 域的值编码如下（其它所有编码保留）： 值 描述 0x1 DID0 寄存器格式的第二版。
27:24	保留	RO	0x08	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
23:16	CLASS	RO	0x05	器件分类 CLASS 域的值用于标识内部设计。根据内部设计，所有微控制器的掩码组在特定的产品线中产生。CLASS 域的值随新产品线而改变，随 fab 过程的改变而改变（例如重新映射或收缩），或者在任何 MAJOR或 MINOR 域需要与之前微控制器有差异的地方改变。CLASS 域的值编码如下（所有其它的编码保留）： 值 描述 0x05 Tiva™ TM4C123x 微控制器

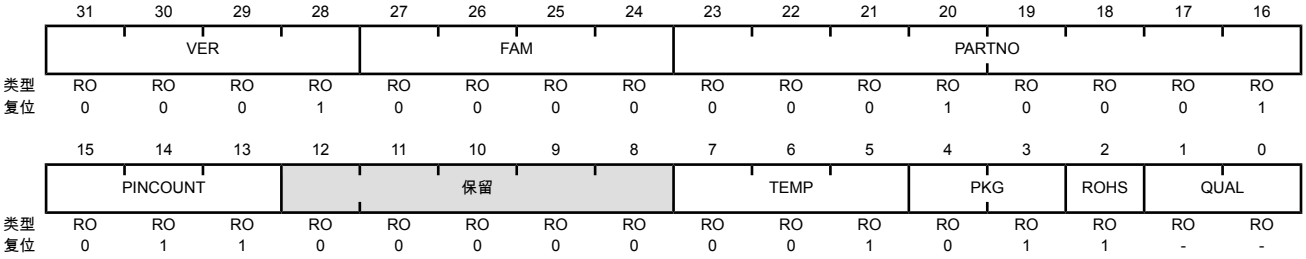
位/域	名称	类型	复位	描述
15:8	MAJOR	RO	-	主模具版本 这个域指定了微控制器的主版本号。主版本反映了设计的基本层的改变。该域编码如下： 值 描述 0x0 版本 A（初值设备） 0x1 版本 B（第一个基本层版本） 0x2 版本 B（第二个基本层版本） 依此类推。
7:0	MINOR	RO	-	次模具版本 这个域指定了微控制器的次版本号。次版本反映了设计的金属层的改变。MAJOR 域改变时 MINOR 域的值将复位。这个域的值是数字，编码如下： 值 描述 0x0 原始器件，或者一个主版本更新。 0x1 第一个金属层改变。 0x2 第二个金属层改变。 依此类推。

寄存器 2: 器件标识寄存器 1（DID1），偏移量 0x004

该寄存器标识了器件系列、器件号、温度范围、管脚数和封装类型。每个微控制器由 DID0 寄存器中的 CLASS 域和 DID1 寄存器中的 PARTNO 域的值组合起来进行专门标识。

器件标识寄存器 1 (DID1)

基址 0x400F.E000
偏移量 0x004
类型 RO, 复位 -



位/域	名称	类型	复位	描述
31:28	VER	RO	0x1	DID1版本 该域定义了 DID1 寄存器格式的版本。版本号是数字的。VER 域的值编码如下（其它所有编码保留）： 值 描述 0x0 最初的 DID1 寄存器格式定义，表示 Stellaris LM3Snnn 器件。 0x1 DID1 寄存器格式的第二版。
27:24	FAM	RO	0x0	产品系列 该域提供了微处理器产品组中的器件系列标识。该值编码如下（所有其它编码保留）： 值 描述 0x0 Tiva™ C 系列 微控制器和传统 Stellaris 微控制器，即所含外部器件的型号以 TM4C、LM4F 或 LM3S 开头的所有器件。
23:16	PARTNO	RO	0x11	器件型号 该域提供了该产品在系列中的器件号。显示的复位值指示 TM4C1231H6PM 微控制器。
15:13	PINCOUNT	RO	0x3	封装管脚数 该域规定了该器件封装的管脚数。该值编码如下（所有其它编码保留）： 值 描述 0x0 保留 0x1 保留 0x2 100 管脚封装 0x3 64 管脚封装 0x4 144 管脚封装 0x5 157 管脚封装 0x6 168 管脚封装

位/域	名称	类型	复位	描述
12:8	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
7:5	TEMP	RO	0x1	温度范围 该域规定了器件的温度等级。该值编码如下（所有其它编码保留）： 值 描述 0x0 保留 0x1 工业温度范围(-40°C ~ 85°C) 0x2 扩展温度范围 (-40°C ~ 105°C) 0x3 适用于工业温度范围（-40°C 到 85°C）和扩展温度范围（-40°C 到 105°C）的器件。特定订购编号请参阅“封装信息”（1070页）。
4:3	PKG	RO	0x1	封装类型 该域规定封装类型。该值编码如下（所有其它编码保留）： 值 描述 0x0 保留 0x1 LQFP 封装 0x2 BGA 封装
2	ROHS	RO	0x1	RoHS一致 该位说明了器件是否符合 RoHS 标准。值为 1 表示符合。
1:0	QUAL	RO	-	认证情况 该域规定了器件的合格状态。该值编码如下（所有其它编码保留）： 值 描述 0x0 工程样片(未经认证） 0x1 试制产品（未经认证） 0x2 完全通过认证

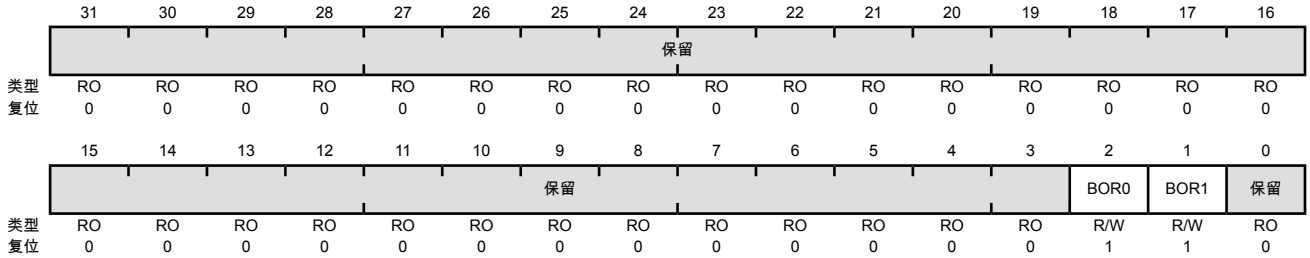
寄存器 3: 掉电复位控制（PBORCTL），偏移量 0x030

该寄存器用来控制初始上电复位之后的复位条件。

注意: BOR 电压值和中点完全来自于模拟。这些值尚无法确定，且可能随时发生修改。

掉电复位控制 (PBORCTL)

基址 0x400F.E000
偏移量 0x030
类型 R/W, 复位 0x0000.7FFF



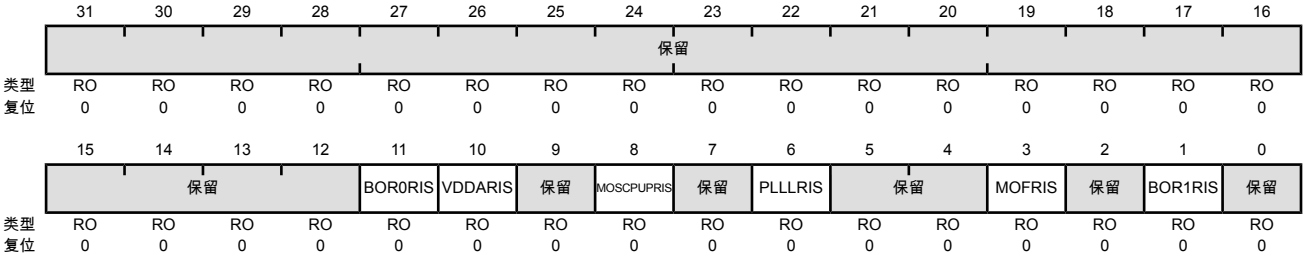
位/域	名称	类型	复位	描述
31:3	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
2	BOR0	R/W	1	VDD 低于 BOR0 事件动作 VDD BOR0 跳变值为 3.02V +/- 90mv。 值 描述 0 一次 BOR0 事件导致一次中断并发送给中断控制器。 1 一次 BOR0 事件导致微控制器复位。
1	BOR1	R/W	1	VDD 低于 BOR1 事件动作 VDD BOR1 跳变值为 2.88V +/- 90mv。 值 描述 0 一次 BOR1 事件导致一次中断并发送给中断控制器。 1 一次 BOR1 事件导致微控制器复位。
0	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。

寄存器 4: 原始中断状态（RIS），偏移量 0x050

该寄存器显示系统控制原始中断的状态。如果中断屏蔽控制 (IMC) 寄存器相应的位被置位，就会产生一个中断发送到中断控制器。向屏蔽中断状态和清除 (MISC) 寄存器相应的位写 1 可以清除中断状态位。

原始中断状态 (RIS)

基址 0x400F.E000
偏移量 0x050
类型 RO, 复位 0x0000.0000



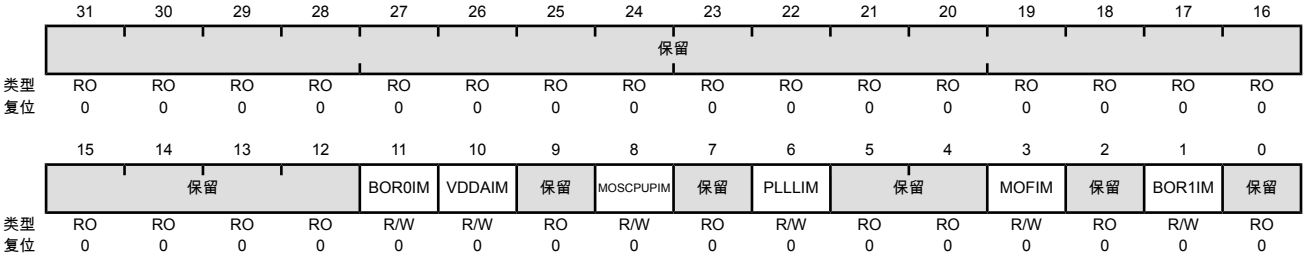
位/域	名称	类型	复位	描述
31:12	保留	RO	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
11	BOR0RIS	RO	0	VDD 处于 BOR0 原始中断状态 值 描述 1 VDD BOR0 条件当前起作用。 0 VDD BOR0 条件当前不起作用。 注意：为让 BOR0 事件引发中断，必须清零 PBORCTL 寄存器的 BOR0 位。 通过在 MISC 寄存器的 BORMIS 位写 1 可以清除这个位。
10	VDDARIS	RO	0	VDDA 电源正常事件原始中断状态 值 描述 1 VDDA 处于适当的工作电压。 0 VDDA 未处于适当的工作电压。 通过在 MISC 寄存器的 VDDAMIS 位写 1 可以清除这个位。
9	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
8	MOSCPUPRIS	RO	0	MOSC上电原始中断状态 值 描述 1 MOSC 经过充足的时间到达预期的频率。这个上电时间值显示为 T_{MOSC_START}。 0 MOSC没有经过充足的时间到达预期的频率。 通过在 MISC 寄存器的 MOSCPUPMIS 位写 1 可以清除这个位。
7	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。

位/域	名称	类型	复位	描述
6	PLLLRIS	RO	0	PLL锁定原始中断状态 值 描述 1 PLL 定时器已经达到了 T_{READY}，表明 PLL 经过了充足的时间锁定。 0 PLL 定时器没有达到 T_{READY}。 通过在 MISC 寄存器的 PLLLMIS 位写 1 可以清除这个位。
5:4	保留	RO	0x0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
3	MOFRIS	RO	0	主振荡器故障原始中断状态 值 描述 1 MOSCCTL 寄存器中的 MOSCIM 位置位并且主振荡器发生故障。 0 主振荡器未发生故障。 通过在 MISC 寄存器的 MOFMIS 位写 1 可以清除这个位。
2	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
1	BOR1RIS	RO	0	VDD 处于 BOR1 原始中断状态 值 描述 1 VDD BOR1 条件当前起作用。 0 VDD BOR1 条件当前不起作用。 注意：为让 BOR1 事件引发中断，必须清零 PBORCTL 寄存器的 BOR1 位。 通过在 MISC 寄存器的 BOR1MIS 位写 1 可以清除这个位。
0	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。

寄存器 5: 中断屏蔽控制（IMC），偏移量 0x054

该寄存器包含了系统控制原始中断的屏蔽位。如果原始中断状态 (RIS) 寄存器中的相应位被置位，那么这个寄存器的某位置位所表示的原始中断会被发送到中断控制器。

中断屏蔽控制 (IMC)
基址 0x400F.E000
偏移量 0x054
类型 R/W, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:12	保留	RO	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
11	BOR0IM	R/W	0	VDD 处于 BOR0 中断屏蔽 值 描述 1 当 RIS 寄存器中的 BOR0RIS 位被置位时，向中断控制器发送一个中断。 0 BOR0RIS 中断被抑制，不会发送到中断控制器。
10	VDDAIM	R/W	0	VDDA 电源正常中断屏蔽 值 描述 1 当 RIS 寄存器中的 VDDARIS 位被置位时，向中断控制器发送一个中断。 0 VDDARIS 中断被抑制，不会发送到中断控制器。
9	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
8	MOSCPUPIM	R/W	0	MOSC上电中断屏蔽 值 描述 1 当RIS寄存器中的MOSCPUPRIS位被置位时，向中断控制器发送一个中断。 0 MOSCPUPRIS中断被抑制，不会发送到中断控制器。
7	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。

位/域	名称	类型	复位	描述
6	PLLLIM	R/W	0	PLL锁定的中断屏蔽 值 描述 1 当 RIS 寄存器中的 PLLLRIS 位被置位时，向中断控制器发送一个中断。 0 PLLLRIS 中断被抑制，不会发送到中断控制器。
5:4	保留	RO	0x0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
3	MOFIM	R/W	0	主振荡器故障中断屏蔽 值 描述 1 当 RIS 寄存器中的 MOFRIS 位被置位时，向中断控制器发送一个中断。 0 MOFRIS 中断被抑制，中断不会发送到中断控制器。
2	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
1	BOR1IM	R/W	0	VDD 低于 BOR1 中断屏蔽 值 描述 1 当 RIS 寄存器中的 BOR1RIS 位被置位时，向中断控制器发送一个中断。 0 BOR1RIS 中断被抑制，不会发送到中断控制器。
0	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。

寄存器 6: 屏蔽的中断状态和清除（MISC），偏移量 0x058

读该寄存器时，寄存器给出原始中断状态 (RIS) 寄存器中相应中断的当前屏蔽的状态值。所有的位都是 R/W1C，所以对一个位写入 1 可以清零 RIS寄存器中相应的原始中断位（见213页）。

屏蔽的中断状态和清除 (MISC)

基址 0x400F.E000
偏移量 0x058
类型 R/W1C, 复位 0x0000.0000

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
域	保留															
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
域	保留				BOR0MIS	VDDAMIS	保留	MOSCPUPMIS	保留	PLLLMIS	保留		MOFMIS	保留	BOR1MIS	保留
类型	RO	RO	RO	RO	R/W1C	R/W1C	RO	R/W1C	RO	R/W1C	RO	RO	RO	RO	R/W1C	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位/域	名称	类型	复位	描述
31:12	保留	RO	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
11	BOR0MIS	R/W1C	0	VDD 低于 BOR0 屏蔽中断状态 值 描述 1 当读取该位时，值为 1 表示因为发生 BOR0 条件而触发了一个非屏蔽中断。 对这个位写 1 可清零该位和 RIS 寄存器的 BOR0RIS 位。 0 当读取该位时，值为 0 表示没有发生 BOR0 条件。 写 0 对该位状态没有影响。
10	VDDAMIS	R/W1C	0	VDDA 电源正常屏蔽中断状态 值 描述 1 当读取该位返回 1 时，表示由于 VDDA 低于正常工作电压而触发了一个非屏蔽中断。 对这个位写 1 可清零该位和 RIS 寄存器的 VDDARIS 位。 0 当读取该位为 0 时，表示 VDDA 电压正常。 写 0 对该位状态没有影响。
9	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
8	MOSCPUPMIS	R/W1C	0	MOSC上电屏蔽的中断状态 值 描述 1 当读取该位时，值为1表示一个非屏蔽中断被标记，这是由于MOSC PLL 经过了充足的时间锁定而引起的。 对这个位写 1 可清零该位和 RIS 寄存器的 MOSCPUPRIS 位。 0 当读取该位时，值为0表示MOSC PLL 没有经过了充足的时间锁定。 写 0 对该位状态没有影响。

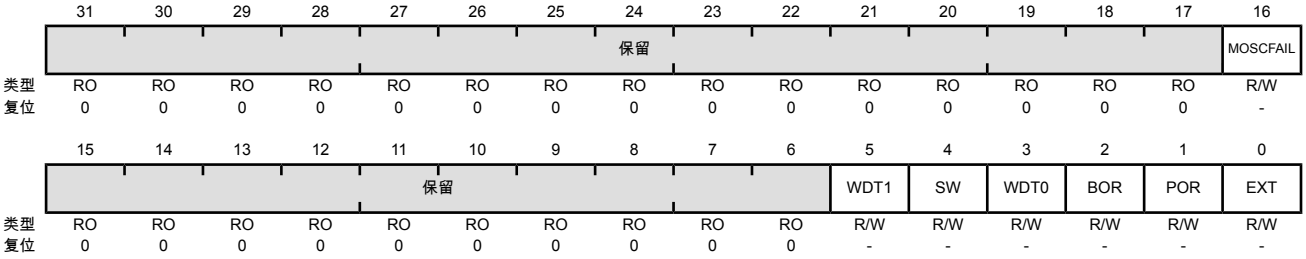
位/域	名称	类型	复位	描述
7	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
6	PLLLMIS	R/W1C	0	PLL 锁定屏蔽的中断状态 值 描述 1 当读取该位时，值为1表示一个非屏蔽中断被标记，这是由于PLL经过了充足的时间锁定而引起的。 对这个位写 1 可清零该位和 RIS 寄存器的 PLLLRIS 位。 0 当读取该位时，值为0表示 PLL 没有经过了充足的时间锁定。 写 0 对该位状态没有影响。
5:4	保留	RO	0x0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
3	MOFMIS	RO	0	主振荡器故障屏蔽中断状态 值 描述 1 当读取该位为1时，表示一个非屏蔽中断信号被发出，原因是主振荡器发生故障。 对这个位写 1 可清零该位和 RIS 寄存器的 MOFRIS 位。 0 当读取该位时，值为0表示主振荡器没有发生故障。 写 0 对该位状态没有影响。
2	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
1	BOR1MIS	R/W1C	0	VDD 处于 BOR1 屏蔽中断状态 值 描述 1 当读取该位时，值为 1 表示因为发生 BOR1 条件而触发了一个非屏蔽中断。 对这个位写 1 可清零该位和 RIS 寄存器的 BOR1RIS 位。 0 当读取该位时，值为 0 表示没有发生 BOR1 条件。 写 0 对该位状态没有影响。
0	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。

寄存器 7: 复位原因（RESC），偏移量 0x05C

复位后该寄存器随复位原因而置位。该寄存器中的位具有粘着性，经过多个复位序列后仍保持它们的状态，但上电复位除外。在上电复位时，RESC 寄存器中除 POR 外的所有其它位都被清零。

复位原因 (RESC)

基址 0x400F.E000
偏移量 0x05C
类型 R/W, 复位 -



位/域	名称	类型	复位	描述
31:17	保留	RO	0x000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
16	MOSCFAIL	R/W	-	MOSC失败复位 值 描述 1 当读取该位时，该位表示 MOSC 电路被启用作时钟验证，但是 MOSCCTL 寄存器中的 MOSCIM 位清零时失败，于是产生了一次复位事件。 0 当读取该位时，值为0表示从之前的上电复位开始MOSC失败没有产生过复位。 对这个位写0可将它清零。
15:6	保留	RO	0x00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
5	WDT1	R/W	-	看门狗定时器1复位 值 描述 1 当读取该位时，该值表示看门狗定时器 1 超时并产生了一次复位。 0 当读取该位时，该值表示从之前的上电复位开始看门狗定时器 1 没有产生过复位。 对这个位写0可将它清零。
4	SW	R/W	-	软件复位 值 描述 1 当读取该位时，值为1表示软件复位引起了一次复位事件。 0 当读取该位时，值为0表示从之前的上电复位开始软件复位没有产生过复位。 对这个位写0可将它清零。

位/域	名称	类型	复位	描述
3	WDT0	R/W	-	看门狗定时器0复位 值 描述 1 当读取该位时，该值表示看门狗定时器 0 超时并产生了一次复位。 0 当读取该位时，该值表示从之前的上电复位开始看门狗定时器 0 没有产生过复位。 对这个位写0可将它清零。
2	BOR	R/W	-	掉电复位 值 描述 1 当读取该位时，值为 1 表示掉电（BOR0 或 BOR1）复位引起了一次复位事件。 0 当读取该位时，值为 0 表示自之前的上电复位后，未因掉电（BOR0 或 BOR1）复位导致复位事件。 对这个位写0可将它清零。
1	POR	R/W	-	上电复位 值 描述 1 当读取该位时，值为1表示上电复位引起了一次复位事件。 0 当读取该位时，值为0表示上电复位没有产生过复位。 对这个位写0可将它清零。
0	EXT	R/W	-	外部复位 值 描述 1 当读取该位时，值为 1 表示外部复位（RST 有效）引起了一次复位事件。 0 当读取该位时，值为 0 表示自之前的上电复位后，未因数外部复位（RST 有效）导致复位事件。 对这个位写0可将它清零。

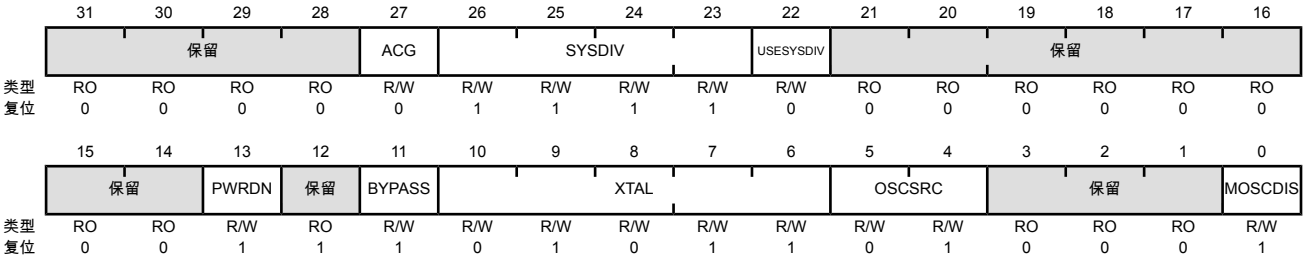
寄存器 8: 运行模式时钟配置（RCC），偏移量 0x060

该寄存器中的位配置系统时钟和振荡器。

重要: 写 RCC 寄存器后再写 RCC2 寄存器。

运行模式时钟配置 (RCC)

基址 0x400F.E000
偏移量 0x060
类型 R/W, 复位 0x0780.3AD1



位/域	名称	类型	复位	描述
31:28	保留	RO	0x0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
27	ACG	R/W	0	自动始终门控 该位规定了如果微控制器进入了睡眠模式或深度睡眠模式，系统是否分别使用睡眠模式时钟门控控制 (SCGCn) 寄存器和深度睡眠模式时钟门控控制 (DCGCn) 寄存器。 值 描述 1 当微控制器处于一种睡眠模式时，SCGCn 或 DCGCn 寄存器用来控制分配给外设的时钟。当微控制器处于一种睡眠模式时，SCGCn 和 DCGCn 寄存器使得没有使用的外设耗费更少的功耗。 0 当微控制器进入一种睡眠模式时，将使用运行模式时钟门控控制 (RCGCn) 寄存器。 在运行模式中，总是使用 RCGCn 寄存器来控制时钟。
26:23	SYSDIV	R/W	0xF	系统时钟分频值 规定了使用哪个分频值来根据 PLL 输出或振荡器源产生系统时钟（取决于该寄存器中 BYPASS 位如何配置）。关于位编码的信息参见表 5-4（195页）。 如果 SYSDIV 的值小于 MINSYSDIV（见370页），并且 PLL 正在被使用，那么 MINSYSDIV 值被作为分频值使用。 如果 PLL 没有被使用，SYSDIV 值可以小于 MINSYSDIV。
22	USESYSDIV	R/W	0	启用系统时钟分频器 值 描述 1 系统时钟分频器用作系统时钟源。当PLL被选作源时，将强制使用系统时钟分频器。 如果 RCC2 寄存器中的 USERCC2 位被置位，那么会使用 RCC2 寄存器中的 SYSDIV2 域来作为系统时钟分频值，而不是这个寄存器中的 SYSDIV 域。 0 系统时钟不分频使用。

位/域	名称	类型	复位	描述
21:14	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
13	PWRDN	R/W	1	PLL 掉电 值 描述 1 PLL 掉电。在该位置位前，必须注意要确保有另外的时钟起作用，并且 BYPASS 位被置位。 0 PLL正常工作。
12	保留	RO	1	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
11	BYPASS	R/W	1	PLL旁路 值 描述 1 系统时钟来自 OSC 源并且被 SYSDIV 规定的分频值分频。 0 系统时钟是被 SYSDIV 规定的分频值分频的 PLL 输出时钟。 编程指南参见表5-4（195页）。 注意: ADC必须使用PLL作为时钟或直接使用16MHz时钟源以便正常工作。

位/域	名称	类型	复位	描述
10:6	XTAL	R/W	0x0B	晶体值 该域规定了与主振荡器连接的晶振值。该域的编码在下面提供。 值 不使用PLL时的晶体频率（MHz） 使用PLL时的晶体频率（MHz） 0x00-0x5 保留 0x06 4 MHz 保留 0x07 4.096 MHz 保留 0x08 4.9152 MHz 保留 0x09 5 MHz 0x0A 5.12 MHz 0x0B 6 MHz 0x0C 6.144 MHz 0x0D 7.3728 MHz 0x0E 8 MHz 0x0F 8.192 MHz 0x10 10.0 MHz 0x11 12.0 MHz 0x12 12.288 MHz 0x13 13.56 MHz 0x14 14.31818 MHz 0x15 16.0 MHz 0x16 16.384 MHz 0x17 18.0 MHz 0x18 20.0 MHz 0x19 24.0 MHz 0x1A 25.0 MHz
5:4	OSCSRC	R/W	0x1	振荡源 选择 OSC 的输入源。该值为： 值 输入源 0x0 主振荡器 主振荡器 0x1 PIOSC 精确内部振荡器 （默认） 0x2 PIOSC/4 精确内部振荡器/4 0x3 LFIOSC 低频内部振荡器 有关其他振荡器源，请参阅 RCC2 寄存器。
3:1	保留	RO	0x0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。

位/域	名称	类型	复位	描述
0	MOSCDIS	R/W	1	主振荡器禁止 值 描述 1 主振荡器禁止(默认)。 0 主振荡器使能。

寄存器 9: GPIO 高性能总线控制（GPIOHBCTL），偏移量 0x06C

该寄存器控制着哪个内部总线可用来访问每个GPIO端口。当某位清零时，相应的GPIO端口穿过老的高级外设总线 (APB) 并通过APB存储器槽 (aperture) 被访问。当某位置位时，相应的GPIO端口穿过高级高性能总线 (AHB) 并通过AHB存储器槽被访问。每个GPIO端口都可以独立的被配置使用AHB或APB，但是只可以通过一个槽被访问。AHB 总线能比 APB 总线提供更好的连续访问性能。对于启用用于 AHB 访问的端口，存储器映射中的地址槽会改变（见表 10-6（584页））。

重要: 端口 K-N 和 P-Q 仅适用于 AHB 总线，因此相应位复位为 1。如果其中一个位清零，则禁用相应的端口。如果这些端口中的任何一个正在使用，应用使用读-修改-写的操作来更改该寄存器的值，以便这些端口保持启用。

GPIO 高性能总线控制 (GPIOHBCTL)

基址 0x400F.E000
偏移量 0x06C
类型 R/W, 复位 0x0000.7E00

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	保留															
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	保留										PORTF	PORTE	PORTD	PORTC	PORTB	PORTA
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W	R/W	R/W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位/域	名称	类型	复位	描述
31:6	保留	RO	0x0000.0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
5	PORTF	R/W	0	端口F高级高性能总线 该位为端口F定义了存储器槽。 值 描述 1 高级高性能总线 (AHB)。 0 高级外设总线 (APB)。这个总线是传统总线。
4	PORTE	R/W	0	端口E高级高性能总线 该位为端口E定义了存储器槽。 值 描述 1 高级高性能总线 (AHB)。 0 高级外设总线 (APB)。这个总线是传统总线。
3	PORTD	R/W	0	端口D高级高性能总线 该位为端口D定义了存储器槽。 值 描述 1 高级高性能总线 (AHB)。 0 高级外设总线 (APB)。这个总线是传统总线。

位/域	名称	类型	复位	描述
2	PORTC	R/W	0	端口C高级高性能总线 该位为端口C定义了存储器槽。 值 描述 1 高级高性能总线 (AHB)。 0 高级外设总线 (APB)。这个总线是传统总线。
1	PORTB	R/W	0	端口B高级高性能总线 该位为端口B定义了存储器槽。 值 描述 1 高级高性能总线 (AHB)。 0 高级外设总线 (APB)。这个总线是传统总线。
0	PORTA	R/W	0	端口A高级高性能总线 该位为端口A定义了存储器槽。 值 描述 1 高级高性能总线 (AHB)。 0 高级外设总线 (APB)。这个总线是传统总线。

寄存器 10: 运行模式时钟配置 2（RCC2），偏移量 0x070

该寄存器替代了类似RCC的寄存器域，如表5-8 所示。当USERCC2位置位时，允许使用RCC2寄存器的扩展功能，同时也提供方法来向下兼容以前的器件。每个取代 RCC 域的 RCC2 域都位于相同的 LSB 位位置；但是，某些 RCC2 域比相应的 RCC 域更大。

表 **5-8.** 替代 **RCC** 域的 **RCC2** 域

RCC2 域...	替代 **RCC** 域
SYSDIV2，位 [28:23]	SYSDIV，位 [26:23]
PWRDN2，位 [13]	PWRDN，位 [13]
BYPASS2，位 [11]	BYPASS，位 [11]
OSCSRC2，位 [6:4]	OSCSRC，位 [5:4]

重要: 写 RCC 寄存器后再写 RCC2 寄存器。

运行模式时钟配置 2 (RCC2)

基址 0x400F.E000
偏移量 0x070
类型 R/W, 复位 0x07C0.6810

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
域	USERCC2	DIV400	保留	SYSDIV2						SYSDIV2LSB	保留					
类型	R/W	R/W	RO	R/W	R/W	R/W	R/W	R/W	R/W	R/W	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	1	1	1	1	1	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
域	保留		PWRDN2	保留	BYPASS2	保留				OSCSRC2			保留			
类型	RO	RO	R/W	RO	R/W	RO	RO	RO	RO	R/W	R/W	R/W	RO	RO	RO	RO
复位	0	0	1	0	1	0	0	0	0	0	0	1	0	0	0	0

位/域	名称	类型	复位	描述
31	USERCC2	R/W	0	使用 RCC2 值 描述 1 RCC2 寄存器域代替 RCC 寄存器域。 0 RCC 寄存器域被使用，同时 RCC2 域被忽略。
30	DIV400	R/W	0	将PLL分为 400MHz vs 200MHz 该位和 SYSDIV2LSB 位配合使用可提供更多频率选择。 值 描述 1 附加在SYSDIV2域的SYSDIV2LSB位之后可创建一个7位的分频器，以使用400 MHz的PLL输出，参见表5-6（196页）。 0 使用 SYSDIV2 本来的值并应用到被预分频的 200MHz PLL 输出。编程指南参见表5-5（196页）。
29	保留	RO	0x0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。

位/域	名称	类型	复位	描述
28:23	SYSDIV2	R/W	0x0F	系统时钟分频值2 规定了使用哪个分频值来根据PLL输出或振荡器源产生系统时钟（取决于该寄存器中 BYPASS2 位如何配置）。当 RCC 寄存器中的 USESYSDIV 位和本寄存器中的 USERCC2 位都置位时，SYSDIV2 用作分频值。编程指南参见表5-5（196页）。
22	SYSDIV2LSB	R/W	1	SYSDIV2 的附加 LSB 当 DIV400 置位时，该位变成 SYSDIV2 的 LSB。如果 DIV400 清零，该位不使用。编程指南参见表5-5（196页）。 该位只有在 DIV400 置位时才能置位和清零。
21:14	保留	RO	0x0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
13	PWRDN2	R/W	1	掉电 PLL 2 值 描述 1 PLL 掉电。 0 PLL 工作正常。
12	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
11	BYPASS2	R/W	1	PLL 旁路 2 值 描述 1 系统时钟来自 OSC 源并且被 SYSDIV2 规定的分频值分频。 0 系统时钟是被 SYSDIV2 规定的分频值分频的 PLL 输出时钟。 编程指南参见表5-5（196页）。 注意: ADC必须使用PLL作为时钟或直接使用16MHz时钟源以便正常工作。
10:7	保留	RO	0x0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
6:4	OSCSRC2	R/W	0x1	振荡器源 2 选择 OSC 的输入源。该值为： 值 描述 0x0 主振荡器 主振荡器 0x1 PIOSC 精确内部振荡器 0x2 PIOSC/4 精确内部振荡器/4 0x3 LFIOSC 低频内部振荡器 0x4-0x6 保留 0x7 32.768 kHz 32.768-kHz 外部振荡器

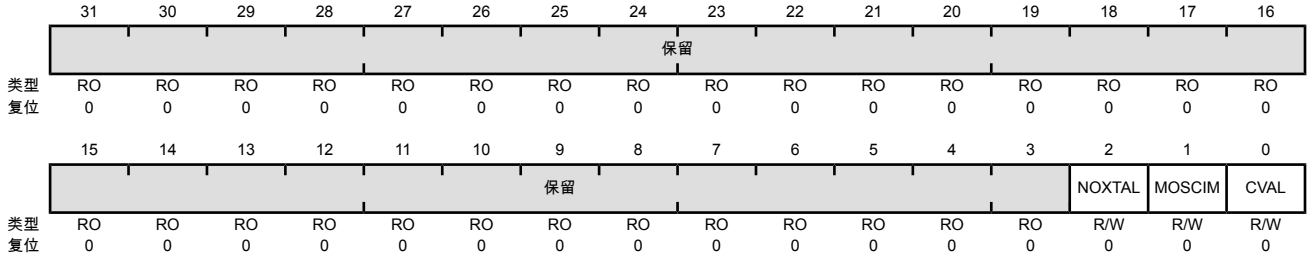
位/域	名称	类型	复位	描述
3:0	保留	RO	0x0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。

寄存器 11: 主振荡器控制（MOSCCTL），偏移量 0x07C

该寄存器提供对主振荡器功能的控制，包括启用 MOSC 时钟验证电路、MOSC 发生故障时需采取什么行动以及是否连接晶振。当使能时，该电路监测MOSC的频率以确认振荡器工作在规定的范围内。如果时钟启用后变得无效，微控制器会发出一个上电复位给 NMI 处理程序并重新启动它，或产生一个中断。

主振荡器控制 (MOSCCTL)

基址 0x400F.E000
偏移量 0x07C
类型 R/W, 复位 0x0000.0000



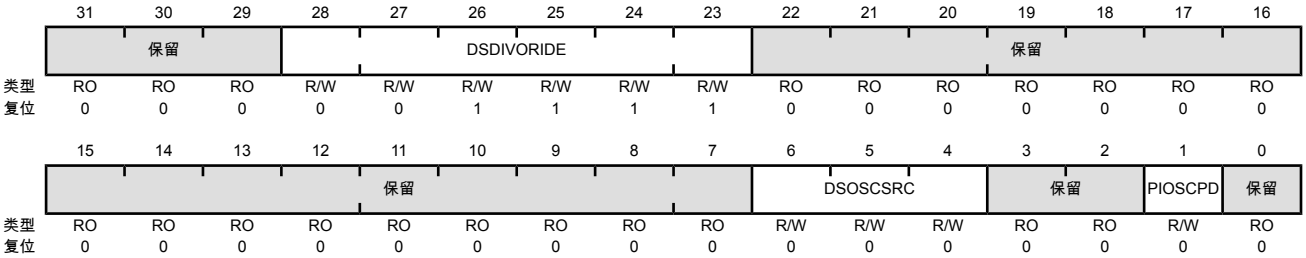
位/域	名称	类型	复位	描述
31:3	保留	RO	0x0000.000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
2	NOXTAL	R/W	0	未连接晶体 值 描述 1 应在晶振或外部振荡器未连接到OSC0 和OSC1 输入时置位该位，以减少功耗。 0 应在晶体或外部振荡器连接到 OSC0 和 OSC1 输入时清零该位，不论是否使用或关闭 MOSC。
1	MOSCIM	R/W	0	MOSC 故障动作 值 描述 1 如果 MOSC 发生故障，则根据 RIS 寄存器中的 MOFRIS 位的指示会产生一个中断。 0 如果 MOSC 发生故障，则会产生 MOSC 故障复位，并且重新启动到 NMI 处理程序。 不论采取什么动作，如果 MOSC 发生故障，振荡器源会自动切换至 PIOSC。
0	CVAL	R/W	0	MOSC时钟确认 值 描述 1 MOSC监测电路使能。 0 MOSC监测电路禁止。

寄存器 12: 深度睡眠时钟配置（DSLPCLKCFG），偏移量 0x144

这个寄存器为深度睡眠模式的硬件控制提供了配置信息。

深度睡眠时钟配置 (DSLPCLKCFG)

基址 0x400F.E000
偏移量 0x144
类型 R/W, 复位 0x0780.0000



位/域	名称	类型	复位	描述
31:29	保留	RO	0x0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
28:23	DSDIVORIDE	R/W	0x0F	分频器域替换 如果当PLL正运行时使能深度睡眠模式，PLL将被禁止。在深度睡眠期间，该6位域包含了一个系统分频器域来代替 RCC 寄存器中的 SYSDIV 域或 RCC2 寄存器中的 SYSDIV2 域。该分频器用于 DSOSCSRC 域选择的源。 值 描述 0x0 /1 0x1 /2 0x2 /3 0x3 /4 ... ... 0x3F /64
22:7	保留	RO	0x000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。

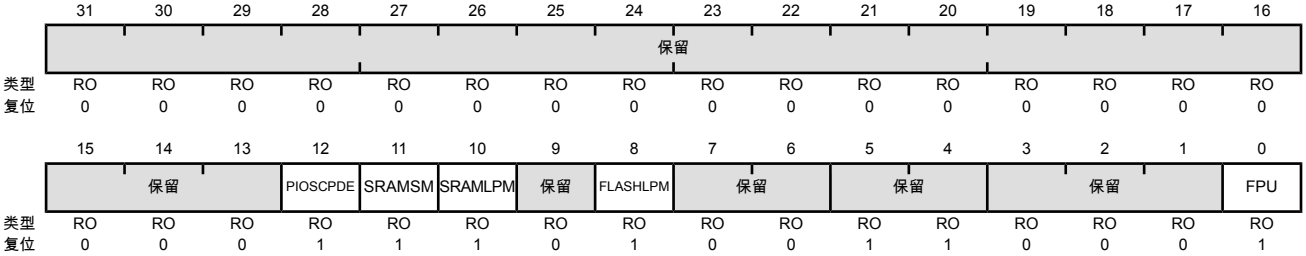
位/域	名称	类型	复位	描述
6:4	DSOSCSRC	R/W	0x0	时钟源 规定了在深度睡眠模式期间时钟源。 值 描述 0x0 主振荡器 使用主振荡器用作源。要将 MOSC 用作深度睡眠模式的时钟源，还必须通过运行模式时钟配置 (RCC) 寄存器将 MOSC 配置为运行模式的时钟源。 注意: 如果PIOSC正用作PLL的时钟参考，那么在深度睡眠模式PIOSC代替MOSC作为时钟源。 0x1 PIOSC 使用精确内部16MHz振荡器作为源。 0x2 保留 0x3 LFIOSC 使用低频内部振荡器作为源。 0x4-0x6 保留 0x7 32.768 kHz 使用休眠模块 32.768-kHz 外部振荡器作为源。
3:2	保留	RO	0x0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
1	PIOSCPD	R/W	0	PIOSC 掉电请求 允许软件请求在深度睡眠模式关闭 PIOSC。如果在深度睡眠期间启用的外设需要使用 PIOSC，则 PIOSC 将掉电，但是 SDPMST 寄存器中的 PPDW 位将产生一个警告。如果不能使 PIOSC 掉电，则 SDPMST 寄存器中的 PPDERR 位将报告错误。 该位仅在 SYSPROP 寄存器的 PIOSCPDE 位置位时才能使 PIOSC 掉电。 值 描述 0 没有动作。 1 软件请求在深度睡眠模式中关闭 PIOSC。
0	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。

寄存器 13: 系统属性寄存器（SYSPROP），偏移量 0x14C

该寄存器提供关于某些系统控制属性在微控制器上是否存在的信息。

系统属性寄存器 (SYSPROP)

基址 0x400F.E000
偏移量 0x14C
类型 RO, 复位 0x0000.1D31



位/域	名称	类型	复位	描述
31:13	保留	RO	0x0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
12	PIOSCPDE	RO	0x1	PIOSC 关闭存在 该位决定是否可以在深度睡眠模式中设置 DSLPCLKCFG 寄存器中的 PIOSCPD 位以使 PIOSC 掉电。 值 描述 0 PIOSCPD 位的状态将忽略。 1 在深度睡眠模式中可以置位 PIOSCPD 位以使 PIOSC 掉电。
11	SRAMSM	RO	0x1	SRAM 睡眠/深度睡眠待机模式存在 该位决定是否可以配置 SLPPWRCFG 和 DSLPPWRCFG 寄存器中的 SRAMPM 域，以便在睡眠或深度睡眠模式中将 SRAM 置入待机模式。 值 描述 0 SRAMPM 域中的值 0x1 将被忽略。 1 可以配置 SRAMPM 域，以便在睡眠或深度睡眠模式中将 SRAM 置入待机模式。
10	SRAMLPM	RO	0x1	SRAM 睡眠/深度睡眠低功率模式存在 该位决定是否可以配置 SLPPWRCFG 和 DSLPPWRCFG 寄存器中的 SRAMPM 域，以便在睡眠或深度睡眠模式中将 SRAM 置入低功率模式。 值 描述 0 SRAMPM 域中的值 0x3 将忽略。 1 可以配置 SRAMPM 域，以便在睡眠或深度睡眠模式中将 SRAM 置入低功率模式。
9	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。

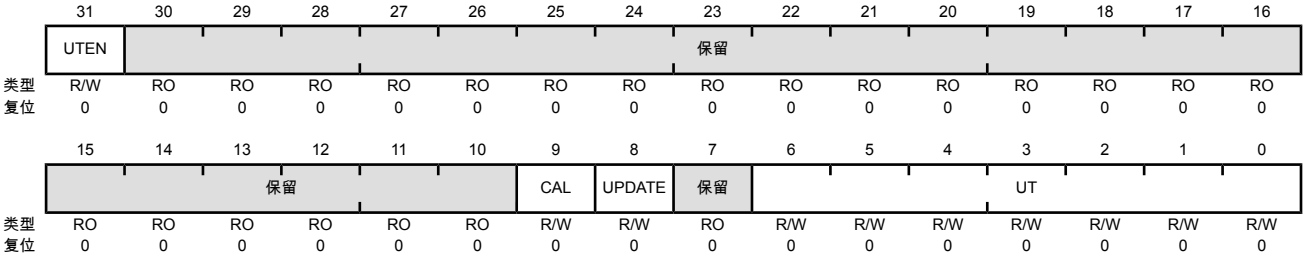
位/域	名称	类型	复位	描述
8	FLASHLPM	RO	0x1	Flash 存储器睡眠/深度睡眠低功率模式存在 该位决定是否可以配置 SLPPWRCFG 和 DSLPPWRCFG 寄存器中的 FLASHPM 域，以便在睡眠或深度睡眠模式中将 Flash 存储器置入低功率模式。 值 描述 0 FLASHPM 域中的值 0x2 将忽略。 1 可以配置 FLASHPM 域，以便在睡眠或深度睡眠模式中将 Flash 存储器置入低功率模式。
7:6	保留	RO	0x0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
5:4	保留	RO	0x3	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
3:1	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
0	FPU	RO	0x1	FPU 存在 该位指示 FPU 在 CortexM4 内核中是否存在。 值 描述 0 FPU 不存在。 1 FPU 存在。

寄存器 14: 精确内部振荡器校准（PIOSCCAL），偏移量 0x150

该寄存器用来更新或重新校准精确内部振荡器。请注意，必须将 32.768-kHz 振荡器用作休眠模块时钟源，以便用户能校准 PIOSC。

精确内部振荡器校准 (PIOSCCAL)

基址 0x400F.E000
偏移量 0x150
类型 R/W, 复位 0x0000.0000



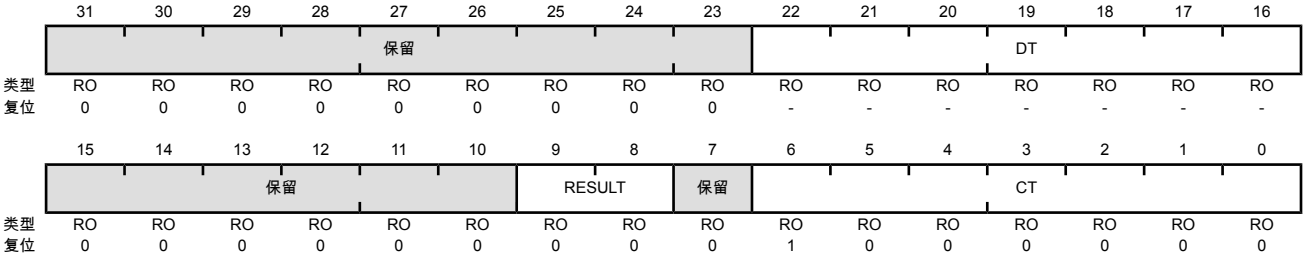
位/域	名称	类型	复位	描述
31	UTEN	R/W	0	使用用户校准值 值 描述 1 该寄存器的[6:0]位校准值用于任何更新校准的操作。 0 出厂校准值用于更新校准的操作。
30:10	保留	RO	0x0000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
9	CAL	R/W	0	开始校准 值 描述 1 开始对 PIOSC 进行新的校准。结果显示在 PIOSCSTAT 寄存器中。校准完成后，操作中产生的校准值在 PIOSC 中有效。不论校准通过或是失败，结果都将替代之前所有的更新校准操作。 0 没有动作。 该位在置位后自动清零。
8	UPDATE	R/W	0	更新校准 值 描述 1 通过 PIOSCSTAT 寄存器中的 UT 位或 DT 位更新 PIOSC 校准值。与 UTEN 一起使用。 0 没有动作。 该位在更新后自动清零。
7	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
6:0	UT	R/W	0x0	用户校准值 用户校准值，可被加载到PIOSC。 关于校准 PIOSC 的更多信息，请参阅“主PLL 频率配置”（197页）。

寄存器 15: 精确内部振荡器统计寄存器（PIOSCSTAT），偏移量 0x154

该寄存器提供关于 PIOSC 校准的用户信息。请注意，必须将 32.768-kHz 振荡器用作休眠模块时钟源，以便用户能校准 PIOSC。

精确内部振荡器统计寄存器 (PIOSCSTAT)

基址 0x400F.E000
偏移量 0x154
类型 RO, 复位 0x0000.0040



位/域	名称	类型	复位	描述
31:23	保留	RO	0x00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
22:16	DT	RO	-	默认校准值 该域包含默认校准值。该值在每次完全上电后装入到 PIOSC 中。
15:10	保留	RO	0x0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
9:8	RESULT	RO	0	校准结果 值 描述 0x0 尚未尝试校准。 0x1 完成的上次校准操作符合 1% 的精度。 0x2 失败的上次校准操作符合 1% 的精度。 0x3 保留
7	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
6:0	CT	RO	0x40	校准值 该域包含上次校准操作的校准值。出厂校准后，CT 和 DT 相同。

寄存器 16: PLL 频率寄存器 0（PLLFREQ0），偏移量 0x160

该寄存器始终包含提交到系统 PLL 的当前 M 值。

可以使用以下等式计算 PLL 频率：

```
PLL frequency = (XTAL frequency * MDIV) / ((Q + 1) * (N + 1))
```

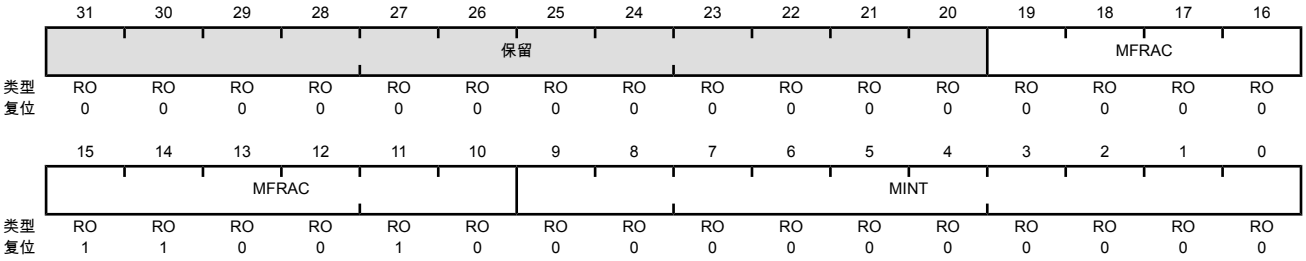
其中

```
MDIV = MINT + (MFRAC / 1024)
```

Q 和 N 值在 PLLFREQ1 寄存器中显示。表21-14（1044页）显示 M、Q 和 N 值，以及由此产生的不同 XTAL 配置的 PLL 频率。

PLL 频率寄存器 0 (PLLFREQ0)

基址 0x400F.E000
偏移量 0x160
类型 RO, 复位 0x0000.0032



位/域	名称	类型	复位	描述
31:20	保留	RO	0x000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
19:10	MFRAC	RO	0x32	PLL M 分数值 该域包含 PLL M 值的整数值。
9:0	MINT	RO	0x00	PLL M 整数值 该域包含 PLL M 值的整数值。

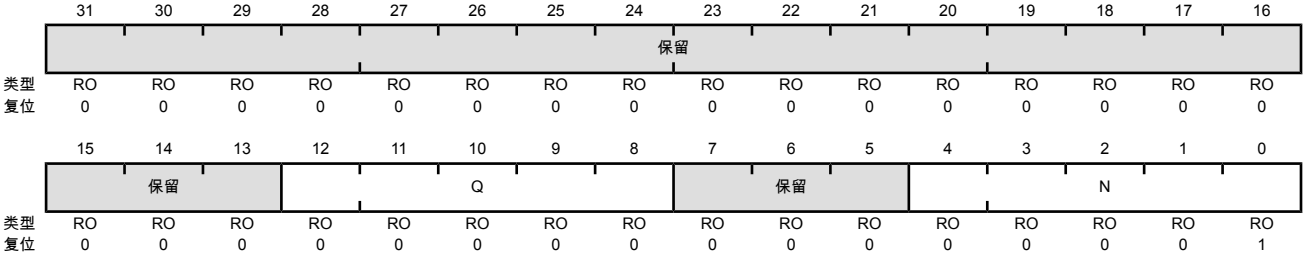
寄存器 17: PLL 频率寄存器 1（PLLFREQ1），偏移量 0x164

该寄存器始终包含提交到系统 PLL 的当前 Q 和 N 值。

M 值在 PLLFREQ0 寄存器中显示。表21-14（1044页）显示 M、Q 和 N 值，以及由此产生的不同 XTAL 配置的 PLL 频率。

PLL 频率寄存器 1 (PLLFREQ1)

基址 0x400F.E000
偏移量 0x164
类型 RO, 复位 0x0000.0001



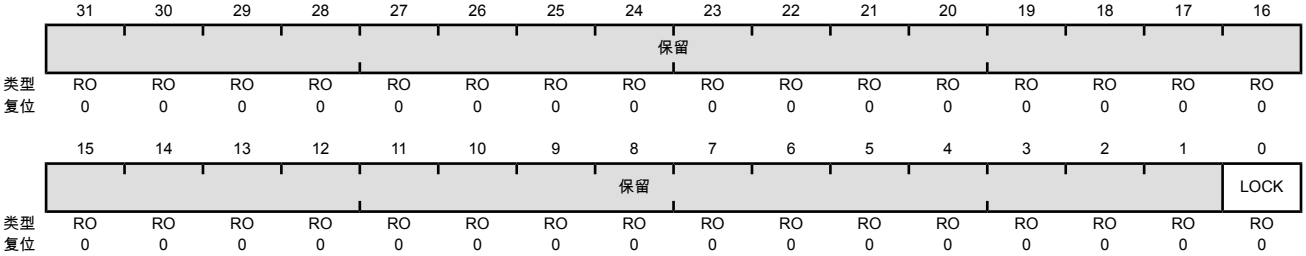
位/域	名称	类型	复位	描述
31:13	保留	RO	0x0000.0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
12:8	Q	RO	0x0	PLL Q 值 该域包含 PLL Q 值。
7:5	保留	RO	0x0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
4:0	N	RO	0x1	PLL N 值 该域包含 PLL N 值。

寄存器 18: PLL 状态寄存器（PLLSTAT），偏移量 0x168

该寄存器显示 PLL 锁定的直接状态。

PLL 状态寄存器 (PLLSTAT)

基址 0x400F.E000
偏移量 0x168
类型 RO, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:1	保留	RO	0x0000.000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
0	LOCK	RO	0x0	PLL 锁定

值 描述

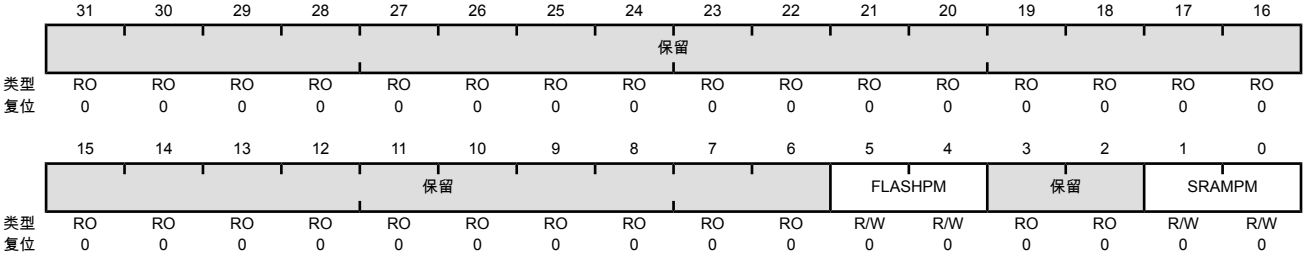
1 PLL 上电并锁定。

0 PLL 掉电或尚未锁定。

寄存器 19: 睡眠功率配置寄存器（SLPPWRCFG），偏移量 0x188

该寄存器提供睡眠模式中的 SRAM 和 Flash 存储器的功率控制配置信息。

睡眠功率配置寄存器 (SLPPWRCFG)
基址 0x400F.E000
偏移量 0x188
类型 R/W, 复位 0x0000.0000



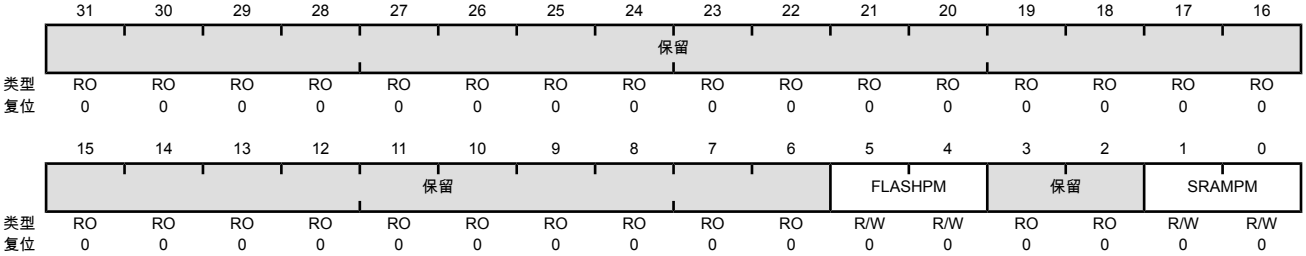
位/域	名称	类型	复位	描述
31:6	保留	RO	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
5:4	FLASHPM	R/W	0x0	Flash 功率模式 值 描述 0x0 主动模式 Flash 存储器不处于低功率模式。在微控制器处于睡眠模式时，该模式提供最快的睡眠和唤醒速度，但功耗也是最高。 0x1 保留 0x2 低功率模式 Flash 存储器处于低功率模式。该模式提供较低的功耗，但是在离开睡眠模式时需要更多时间。 0x3 保留
3:2	保留	RO	0x0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
1:0	SRAMPM	R/W	0x0	SRAM 功率模式 当微控制器处于深度睡眠模式时，该域控制片上 SRAM 的低功率模式。 值 描述 0x0 主动模式 SRAM 不处于低功率模式。在微控制器处于睡眠模式时，该模式提供最快的睡眠和唤醒速度，但功耗也是最高。 0x1 待机模式 在睡眠模式中时 SRAM 处于待机模式。 0x2 保留 0x3 低功率模式 SRAM 处于低功率模式。在睡眠模式中，该模式提供最慢的睡眠和唤醒速度，但功耗最低。

寄存器 20: 深度睡眠功率配置寄存器（DSLPPWRCFG），偏移量 0x18C

该寄存器在深度睡眠模式中时提供 SRAM 和 Flash 存储器的功率控制配置信息。

深度睡眠功率配置寄存器 (DSLPPWRCFG)

基址 0x400F.E000
偏移量 0x18C
类型 R/W, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:6	保留	RO	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
5:4	FLASHPM	R/W	0x0	Flash 功率模式 值 描述 0x0 主动模式 Flash 存储器不处于低功率模式。在微控制器处于深度睡眠模式时，该模式提供最快的睡眠和唤醒速度，但功耗也是最高。 0x1 保留 0x2 低功率模式 Flash 存储器处于低功率模式。该模式提供较低的功耗，但是在离开深度睡眠模式时需要更多时间。 0x3 保留
3:2	保留	RO	0x0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
1:0	SRAMPM	R/W	0x0	SRAM 功率模式 当微控制器处于深度睡眠模式时，该域控制片上 SRAM 的低功率模式。 值 描述 0x0 主动模式 SRAM 不处于低功率模式。在微控制器处于深度睡眠模式时，该模式提供最快的睡眠和唤醒速度，但功耗也是最高。 0x1 待机模式 在深度睡眠模式中时 SRAM 处于待机模式。 0x2 保留 0x3 低功率模式 SRAM 处于低功率模式。在深度睡眠模式中，该模式提供最慢的睡眠和唤醒速度，但功耗最低。

寄存器 21: LDO 睡眠功率控制寄存器（LDOSPCTL），偏移量 0x1B4

该寄存器用于指定睡眠模式中的 LDO 输出电压。不管对 VADJEN 位指定了什么，对 VLDO 位域执行写入操作都对 LDO 输出电压没有任何效果。LDO 输出电压固定为建议的出厂复位值。

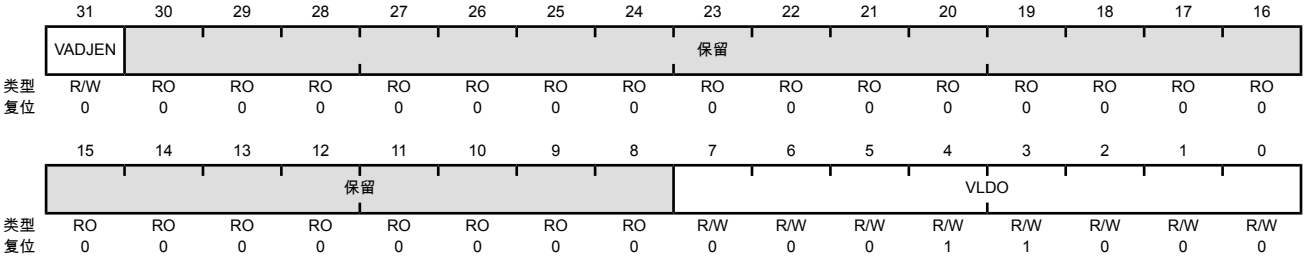
下表显示了相对于配置的 LDO 电压的最高系统时钟频率和 PIOSC 最高频率。

工作电压 **(LDO)**	最高系统时钟频率	**PIOSC**
1,2	80 MHz	16 MHz
0,9	20 MHz	16 MHz

注意：
- 如果调试器自上一次掉电复位后已连接，那么 LDO 在睡眠/深度睡眠模式下将不会自动调整。
- 如果调整了 LDO 电压，在从睡眠或者深度睡眠模式中唤醒时，将需要额外花费 4 us。

LDO 睡眠功率控制寄存器 (LDOSPCTL)

基址 0x400F.E000
偏移量 0x1B4
类型 R/W, 复位 0x0000.0018



位/域	名称	类型	复位	描述
31	VADJEN	R/W	0	电压调整启用 该位启用 VLDO 域的值，用于指定睡眠模式中的 LDO 的输出电压。 值 描述 0 在睡眠模式中，将 LDO 输出电压设置为出厂默认值。VLDO 域的值不影响 LDO 操作。 1 睡眠模式中的 LDO 输出值通过 VLDO 域中的值配置。
30:8	保留	RO	0x000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。

位/域	名称	类型	复位	描述
7:0	VLDO	R/W	0x18	LDO输出电压 该域提供运行模式中的 LDO 输出电压的程序控制。该域的值仅在 VADJEN 位置位时用于 LDO 电压。 要最小化睡眠模式消耗的能量，建议将 LDO 输出电压配置为等于或低于默认的 1.2 V。

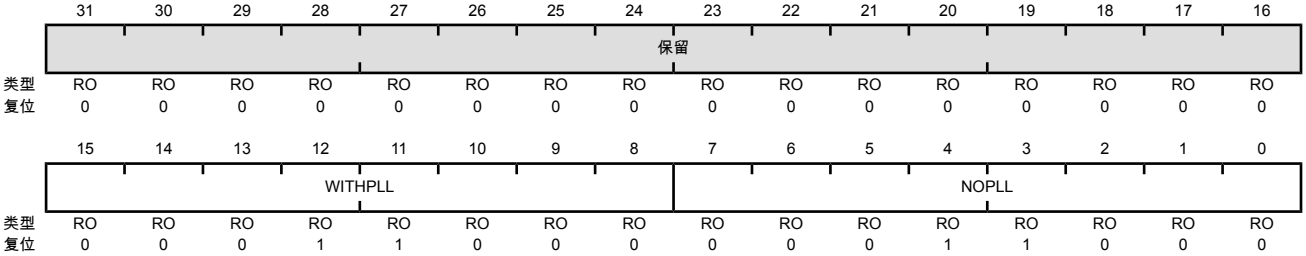
值	描述
0x12	0.90 V
0x13	0.95 V
0x14	1.00 V
0x15	1.05 V
0x16	1.10 V
0x17	1.15 V
0x18	1.20 V
0x19 - 0xFF	保留

寄存器 22: LDO 睡眠功率校准寄存器（LDOSPCAL），偏移量 0x1B8

该寄存器提供出厂设定值，这些值是在睡眠模式中用于 LDOSPCTL 寄存器的 VLDO 域的建议值。

LDO 睡眠功率校准寄存器 (LDOSPCAL)

基址 0x400F.E000
偏移量 0x1B8
类型 RO, 复位 0x0000.1818



位/域	名称	类型	复位	描述
31:16	保留	RO	0x0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
15:8	WITHPLL	RO	0x18	使用 PLL 的睡眠模式 该域中的值为使用 PLL 时对 LDOSPCTL 寄存器的 VLDO 域的建议值。该值提供建议的最低 LDO 输出电压，用于最大指定值时的 PLL。
7:0	NOPLL	RO	0x18	不使用 PLL 的睡眠模式 该域中的值为不使用 PLL 时对 LDOSPCTL 寄存器的 VLDO 域的建议值。该值提供建议的最低 LDO 输出电压，以在没有 PLL 时使用。

寄存器 23: LDO 深度睡眠功率控制寄存器（LDODPCTL），偏移量 0x1BC

该寄存器用于指定深度睡眠模式中的 LDO 输出电压。不管对 VADJEN 位指定了什么，对 VLDO 位域执行写入操作都对 LDO 输出电压没有任何效果。LDO 输出电压固定为建议的出厂复位值。

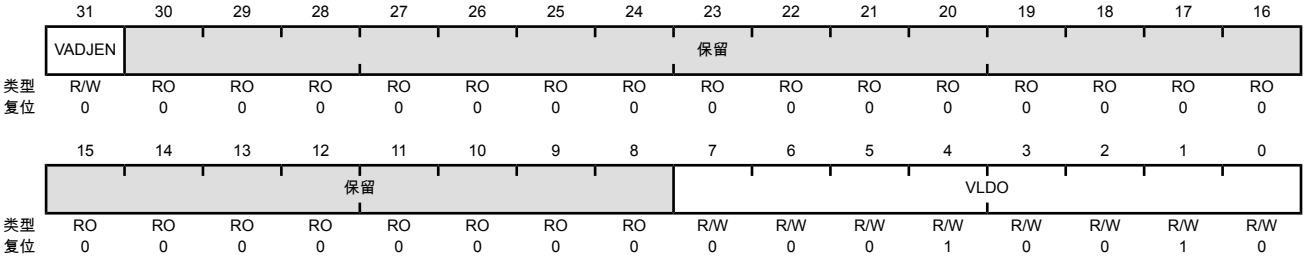
下表显示了相对于配置的 LDO 电压的最高系统时钟频率和 PIOSC 最高频率。

工作电压 **(LDO)**	最高系统时钟频率	**PIOSC**
1,2	80 MHz	16 MHz
0,9	20 MHz	16 MHz

注意：
- 如果调试器自上一次掉电复位后已连接，那么 LDO 在睡眠/深度睡眠模式下将不会自动调整。
- 如果调整了 LDO 电压，在从睡眠或者深度睡眠模式中唤醒时，将需要额外花费 4 us。

LDO 深度睡眠功率控制寄存器 (LDODPCTL)

基址 0x400F.E000
偏移量 0x1BC
类型 R/W, 复位 0x0000.0012



位/域	名称	类型	复位	描述
31	VADJEN	R/W	0	电压调整启用 该位启用VLDO域的值，用于指定深度睡眠模式中的LDO的输出电压。 值 描述 0 在深度睡眠模式中，将 LDO 输出电压设置为出厂默认值。VLDO 域的值不影响 LDO 操作。 1 深度睡眠模式中的 LDO 输出值通过 VLDO 域中的值配置。
30:8	保留	RO	0x000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。

位/域	名称	类型	复位	描述
7:0	VLDO	R/W	0x12	LDO输出电压 该域提供运行模式中的 LDO 输出电压的程序控制。该域的值仅在 VADJEN 位置位时用于 LDO 电压。 要最小化深度睡眠模式消耗的能量，建议将 LDO 输出电压配置为默认的 0.90 V。

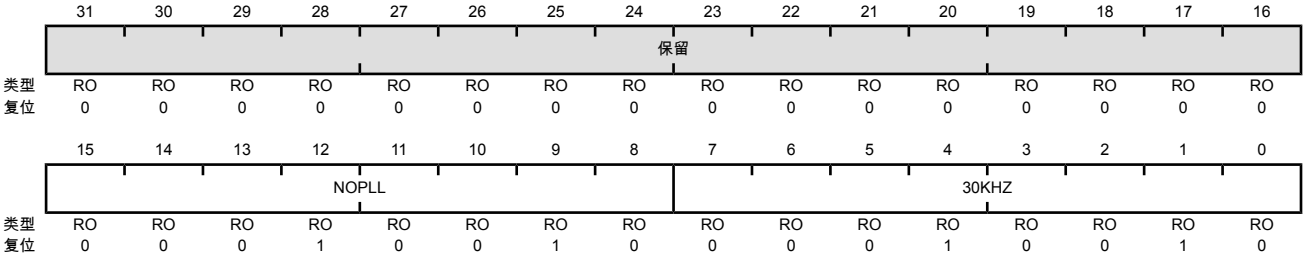
值	描述
0x12	0.90 V
0x13	0.95 V
0x14	1.00 V
0x15	1.05 V
0x16	1.10 V
0x17	1.15 V
0x18	1.20 V
0x19 - 0xFF	保留

寄存器 24: LDO 深度睡眠功率校准寄存器（LDODPCAL），偏移量 0x1C0

该寄存器提供出厂设定值，这些值是在深度睡眠模式中用于 LDOPCTL 寄存器的 VLDO 域的建议值。

LDO 深度睡眠功率校准寄存器 (LDODPCAL)

基址 0x400F.E000
偏移量 0x1C0
类型 RO, 复位 0x0000.1212



位/域	名称	类型	复位	描述
31:16	保留	RO	0x0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
15:8	NOPLL	RO	0x12	不使用 PLL 的深度睡眠模式 该寄存器中的值为不使用 PLL 时对 LDODPCTL 寄存器的 VLDO 域的建议值。该值为系统时钟提供建议的最低 LDO 输出电压。
7:0	30KHZ	RO	0x12	使用 IOSC 的深度睡眠模式 该寄存器中的值为不使用 PLL 时对 LDODPCTL 寄存器的 VLDO 域的建议值。该值为低频内部振荡器提供建议的最低 LDO 输出电压。

寄存器 25: 睡眠/深度睡眠功率模式状态寄存器（SDPMST），偏移量 0x1CC

如果该寄存器正在运行，则它能提供关于睡眠和深度睡眠功率模式的状态信息，以及可以通过调试器或内核查看的一些实时状态。这些事件不会触发中断，并且表示提供可以帮助调整电源管理软件的信息。状态寄存器在引发任何错误检测结果的每个动态电源管理事件请求开始时被写入。尚无机制清零这些位；它们在下一个事件时被覆盖。LDOUA、FLASHLP、LOWPWR、PRACT 位提供实时数据，并无事件记录该信息。

睡眠/深度睡眠功率模式状态寄存器 (SDPMST)

基址 0x400F.E000
偏移量 0x1CC
类型 RO, 复位 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	保留												LDOUA	FLASHLP	LOWPWR	PRACT
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	保留								PPDW	LMAXERR	保留	LSMINERR	LDMINERR	PPDERR	FPDERR	SPDERR
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位/域	名称	类型	复位	描述
31:20	保留	RO	0x000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
19	LDOUA	RO	0	LDO 更新有效 值 描述 1 LDO 电平正在更改。 0 LDO 电平并未更改。
18	FLASHLP	RO	0	低功率状态的 Flash 存储器 值 描述 1 如 SLPPWRCFG 或 DSLPPWRCFG 寄存器中的设置，Flash 存储器当前处于低功率状态。 0 Flash 存储器当前处于活动状态。
17	LOWPWR	RO	0	睡眠或深度睡眠模式 值 描述 1 微控制器当前处于睡眠或深度睡眠模式，并且处于等待中断或处于上电的过程中。该位的状态不受 Flash 存储器或 SRAM 功率状态的影响。 0 微控制器当前处于运行模式中。

位/域	名称	类型	复位	描述
16	PRACT	RO	0	睡眠或深度睡眠功率请求有效 值 描述 1 微控制器当前处于深度睡眠模式或处于睡眠模式，并且根据 SLPPWRCFG 寄存器的配置，将 SRAM 和/或 Flash 存储器置入较低功率模式的请求当前有效。 0 功率请求无效。
15:8	保留	RO	0x00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
7	PPDW	RO	0	PIOSC 关闭请求警告 值 描述 1 由于软件已请求使用 DSLPCLKCFG 寄存器中的 PIOSCPD 位在深度睡眠期间关闭 PIOSC，而外设要求该其在深度睡眠中保持工作，因此发出警告。不论是否出现警告，都将关闭 PIOSC。 0 无错误。
6	LMAXERR	RO	0	VLDO 值超出最大值错误 值 描述 1 由于软件通过 LDOSPCTL 或 LDODPCTL 寄存器中的 VLDO 位请求将 LDO 电压升高到其最大允许值以上，因此出现错误。 此时，LDO 被设置为出厂默认值。 0 无错误。
5	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
4	LSMINERR	RO	0	睡眠模式中 VLDO 值低于最小错误 值 描述 1 由于软件通过 LDOSPCTL 寄存器中的 VLDO 位请求将 LDO 电压降低到其最小允许值以下，因此出现错误。 在这种情况下，LDO 电压在进入睡眠模式时不发生更改。 0 无错误。
3	LDMINERR	RO	0	深度睡眠模式中 VLDO 值低于最小值错误 值 描述 1 由于软件通过 LDODPCTL 寄存器中的 VLDO 位请求将 LDO 电压降低到其最小允许值以下，因此出现错误。 在这种情况下，LDO 电压在进入深度睡眠模式时不发生更改。 0 无错误。

位/域	名称	类型	复位	描述
2	PPDERR	RO	0	PIOSC 关闭请求错误 值 描述 1 由于软件已请求应在深度睡眠期间关闭 PIOSC，但实际无法关闭 PIOSC，因此出现错误。 在这种情况下，在进入深度睡眠模式时关闭 PIOSC。 0 无错误。
1	FPDERR	RO	0	Flash 存储器关闭请求错误 值 描述 1 由于软件通过 SLPPWRCFG 或 DSLPPWRCFG 寄存器中的 FLASHPM 域请求进入不可用的 Flash 存储器掉电模式，因此出现错误。 0 无错误。
0	SPDERR	RO	0	SRAM 关闭请求错误 值 描述 1 由于软件通过 SLPPWRCFG 或 DSLPPWRCFG 寄存器中的 SRAMPM 域请求进入不可用的 SRAM 存储器掉电模式，因此出现错误。 0 无错误。

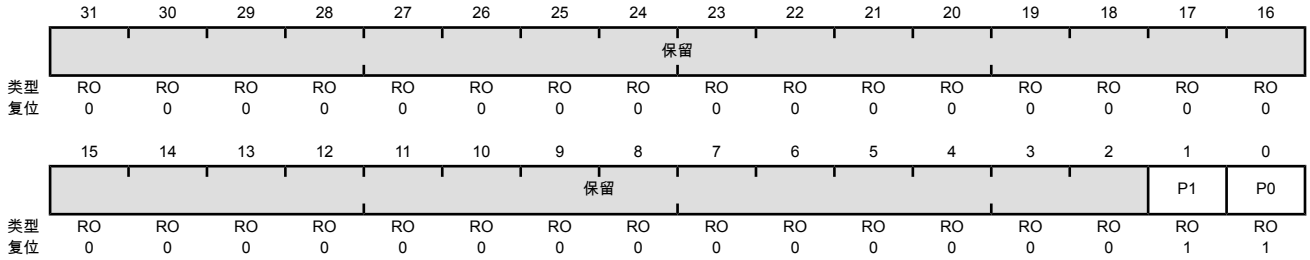
寄存器 26: 看门狗定时器外设存在寄存器（PPWD），偏移量 0x300

PPWD 寄存器提供关于看门狗模块的软件信息。

重要: 应使用该寄存器确定在该微控制器上执行的看门狗定时器。但是，要支持传统软件，可使用 DC1 寄存器。读取 DC1 寄存器可正确识别传统模块是否存在。

看门狗定时器外设存在寄存器 (PPWD)

基址 0x400F.E000
偏移量 0x300
类型 RO, 复位 0x0000.0003



位/域	名称	类型	复位	描述
31:2	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
1	P1	RO	0x1	看门狗定时器 1 存在 值 描述 1 看门狗模块 1 存在。 0 看门狗模块 1 不存在。
0	P0	RO	0x1	看门狗定时器 0 存在 值 描述 1 看门狗模块 0 存在。 0 看门狗模块 0 不存在。

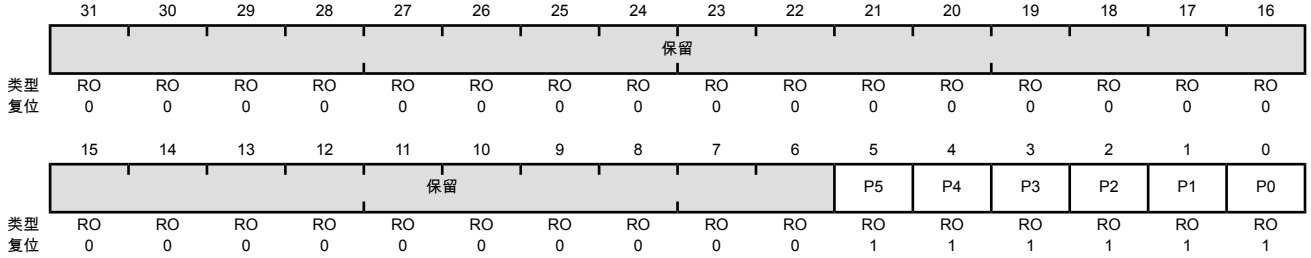
寄存器 27: 16/32 位通用定时器外设存在寄存器（PPTIMER），偏移量 0x304

PPTIMER 寄存器提供关于 16/32 位通用定时器模块的软件信息。

重要: 应使用该寄存器确定在该微控制器上执行的定时器。但是，要支持传统软件，可使用 DC2 寄存器。读取 DC2 寄存器可正确识别传统模块是否存在。软件必须使用该寄存器来确定不受 DC2 寄存器支持的模块是否存在。

16/32 位通用定时器外设存在寄存器 (PPTIMER)

基址 0x400F.E000
偏移量 0x304
类型 RO, 复位 0x0000.003F



位/域	名称	类型	复位	描述
31:6	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
5	P5	RO	0x1	16/32 位通用定时器 5 存在 值 描述 1 16/32 位通用定时器模块 5 存在。 0 16/32 位通用定时器模块 6 不存在。
4	P4	RO	0x1	16/32 位通用定时器 4 存在 值 描述 1 16/32 位通用定时器模块 4 存在。 0 16/32 位通用定时器模块 4 不存在。
3	P3	RO	0x1	16/32 位通用定时器 3 存在 值 描述 1 16/32 位通用定时器模块 3 存在。 0 16/32 位通用定时器模块 3 不存在。
2	P2	RO	0x1	16/32 位通用定时器 2 存在 值 描述 1 16/32 位通用定时器模块 2 存在。 0 16/32 位通用定时器模块 2 不存在。

位/域	名称	类型	复位	描述
1	P1	RO	0x1	16/32 位通用定时器 1 存在 值 描述 1 16/32 位通用定时器模块 1 存在。 0 16/32 位通用定时器模块 1 不存在。
0	P0	RO	0x1	16/32 位通用定时器 0 存在 值 描述 1 16/32 位通用定时器模块 0 存在。 0 16/32 位通用定时器模块 0 不存在。

寄存器 28: 通用输入/输出外设存在寄存器（PPGPIO），偏移量 0x308

PPGPIO 寄存器提供关于通用输入/输出模块的软件信息。

重要: 应使用该寄存器确定在该微控制器上执行的 GPIO 端口。但是，要支持传统软件，可使用 DC4 寄存器。读取 DC4 寄存器可正确识别传统模块是否存在。软件必须使用该寄存器来确定不受 DC4 寄存器支持的模块是否存在。

通用输入/输出外设存在寄存器 (PPGPIO)

基址 0x400F.E000
偏移量 0x308
类型 RO, 复位 0x0000.003F

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	保留															
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	保留	P14	P13	P12	P11	P10	P9	P8	P7	P6	P5	P4	P3	P2	P1	P0
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1

位/域	名称	类型	复位	描述
31:15	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
14	P14	RO	0x0	GPIO 端口 Q 存在 值 描述 1 GPIO 端口 Q 存在。 0 GPIO 端口 Q 不存在。
13	P13	RO	0x0	GPIO 端口 P 存在 值 描述 1 GPIO 端口 P 存在。 0 GPIO 端口 P 不存在。
12	P12	RO	0x0	GPIO 端口 N 存在 值 描述 1 GPIO 端口 N 存在。 0 GPIO 端口 N 不存在。
11	P11	RO	0x0	GPIO 端口 M 存在 值 描述 1 GPIO 端口 M 存在。 0 GPIO 端口 M 不存在。

位/域	名称	类型	复位	描述
10	P10	RO	0x0	GPIO 端口 L 存在 值 描述 1 GPIO 端口 L 存在。 0 GPIO 端口 L 不存在。
9	P9	RO	0x0	GPIO 端口 K 存在 值 描述 1 GPIO 端口 K 存在。 0 GPIO 端口 K 不存在。
8	P8	RO	0x0	GPIO 端口 J 存在 值 描述 1 GPIO 端口 J 存在。 0 GPIO 端口 J 不存在。
7	P7	RO	0x0	GPIO 端口 H 存在 值 描述 1 GPIO 端口 H 存在。 0 GPIO 端口 H 不存在。
6	P6	RO	0x0	GPIO 端口 G 存在 值 描述 1 GPIO 端口 G 存在。 0 GPIO 端口 G 不存在。
5	P5	RO	0x1	GPIO 端口 F 存在 值 描述 1 GPIO 端口 F 存在。 0 GPIO 端口 F 不存在。
4	P4	RO	0x1	GPIO端口E存在 值 描述 1 GPIO 端口 E 存在。 0 GPIO 端口 E 不存在。

位/域	名称	类型	复位	描述
3	P3	RO	0x1	GPIO端口D存在 值 描述 1 GPIO 端口 D 存在。 0 GPIO 端口 D 不存在。
2	P2	RO	0x1	GPIO端口C存在 值 描述 1 GPIO 端口 C 存在。 0 GPIO 端口 C 不存在。
1	P1	RO	0x1	GPIO端口B存在 值 描述 1 GPIO 端口 B 存在。 0 GPIO 端口 B 不存在。
0	P0	RO	0x1	GPIO端口A存在 值 描述 1 GPIO 端口 A 存在。 0 GPIO 端口 A 不存在。

寄存器 29: 微型直接存储器访问外设存在寄存器（PPDMA），偏移量 0x30C

PPDMA 寄存器提供关于 µDMA 模块的软件信息。

重要: 应使用该寄存器确定是否在该微控制器上执行 µDMA 模块。但是，要支持传统软件，可使用 DC7 寄存器。读取 DC7 寄存器即可正确识别 µDMA 模块是否存在。

微型直接存储器访问外设存在寄存器 (PPDMA)

基址 0x400F.E000
偏移量 0x30C
类型 RO, 复位 0x0000.0001

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	保留															
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	保留															P0
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

位/域	名称	类型	复位	描述
31:1	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
0	P0	RO	0x1	µDMA 模块存在 值 描述 1 µDMA 模块存在。 0 µDMA 模块不存在。

寄存器 30: 休眠外设存在寄存器（PPHIB），偏移量 0x314

PPHIB 寄存器提供关于休眠模块的软件信息。

重要: 应使用该寄存器确定是否在该微控制器上执行休眠模块。但是，要支持传统软件，可使用 DC1 寄存器。读取 DC1 寄存器即可正确识别休眠模块是否存在。

休眠外设存在寄存器 (PPHIB)

基址 0x400F.E000
偏移量 0x314
类型 RO, 复位 0x0000.0001

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	保留															
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	保留															P0
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

位/域	名称	类型	复位	描述
31:1	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
0	P0	RO	0x1	休眠模块存在

值 描述
1 休眠模块存在。
0 休眠模块不存在。

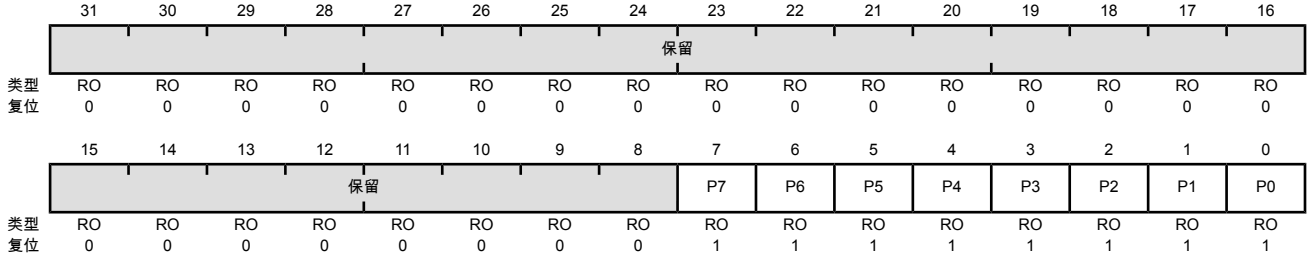
寄存器 31: 通用异步收发器外设存在寄存器（PPUART），偏移量 0x318

PPUART 寄存器提供关于 UART 模块的软件信息。

重要: 应使用该寄存器确定在该微控制器上执行的 UART 模块。但是，要支持传统软件，可使用 DC2 寄存器。读取 DC2 寄存器即可正确识别传统 UART 模块是否存在。软件必须使用该寄存器来确定不受 DC2 寄存器支持的模块是否存在。

通用异步收发器外设存在寄存器 (PPUART)

基址 0x400F.E000
偏移量 0x318
类型 RO, 复位 0x0000.00FF



位/域	名称	类型	复位	描述
31:8	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
7	P7	RO	0x1	UART模块7存在 值 描述 1 UART 模块 7 存在。 0 UART 模块 7 不存在。
6	P6	RO	0x1	UART模块6存在 值 描述 1 UART 模块 6 存在。 0 UART 模块 6 不存在。
5	P5	RO	0x1	UART模块5存在 值 描述 1 UART 模块 5 存在。 0 UART 模块 5 不存在。
4	P4	RO	0x1	UART模块4存在 值 描述 1 UART 模块 4 存在。 0 UART 模块 4 不存在。

位/域	名称	类型	复位	描述
3	P3	RO	0x1	UART模块3存在 值 描述 1 UART 模块 3 存在。 0 UART 模块 3 不存在。
2	P2	RO	0x1	UART模块2存在 值 描述 1 UART 模块 2 存在。 0 UART 模块 2 不存在。
1	P1	RO	0x1	UART模块1存在 值 描述 1 UART 模块 1 存在。 0 UART 模块 1 不存在。
0	P0	RO	0x1	UART模块0存在 值 描述 1 UART 模块 0 存在。 0 UART 模块 0 不存在。

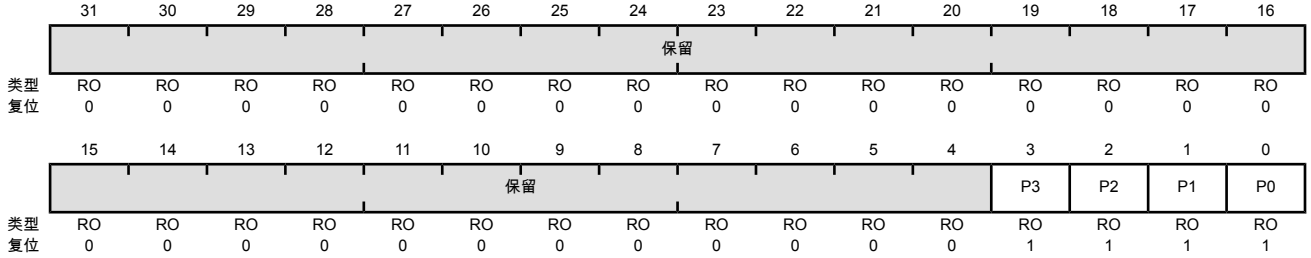
寄存器 32: 同步串行接口外设存在寄存器（PPSSI），偏移量 0x31C

PPSSI 寄存器提供关于 SSI 模块的软件信息。

重要: 应使用该寄存器确定在该微控制器上执行的 SSI 模块。但是，要支持传统软件，可使用 DC2 寄存器。读取 DC2 寄存器即可正确识别传统 SSI 模块是否存在。软件必须使用该寄存器来确定不受 DC2 寄存器支持的模块是否存在。

同步串行接口外设存在寄存器 (PPSSI)

基址 0x400F.E000
偏移量 0x31C
类型 RO, 复位 0x0000.000F



位/域	名称	类型	复位	描述
31:4	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
3	P3	RO	0x1	SSI模块3存在 值 描述 1 SSI 模块 3 存在。 0 SSI 模块 3 不存在。
2	P2	RO	0x1	SSI模块2存在 值 描述 1 SSI 模块 2 存在。 0 SSI 模块 2 不存在。
1	P1	RO	0x1	SSI模块1存在 值 描述 1 SSI 模块 1 存在。 0 SSI 模块 1 不存在。
0	P0	RO	0x1	SSI模块0存在 值 描述 1 SSI 模块 0 存在。 0 SSI 模块 0 不存在。

寄存器 33: 内部集成电路外设存在寄存器（PPI2C），偏移量 0x320

PPI2C 寄存器提供关于 I²C 模块的软件信息。

重要: 应使用该寄存器确定在该微控制器上执行的 I²C 模块。但是，要支持传统软件，可使用 DC2 寄存器。读取 DC2 寄存器即可正确识别传统 I²C 模块是否存在。软件必须使用该寄存器来确定不受 DC2 寄存器支持的模块是否存在。

内部集成电路外设存在寄存器 (PPI2C)

基址 0x400F.E000
偏移量 0x320
类型 RO, 复位 0x0000.000F

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	保留															
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	保留										P5	P4	P3	P2	P1	P0
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1

位/域	名称	类型	复位	描述
31:6	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
5	P5	RO	0x0	I²C 模块 5 存在 值 描述 1 I²C 模块 5 存在。 0 I²C 模块 5 不存在。
4	P4	RO	0x0	I²C 模块 4 存在 值 描述 1 I²C 模块 4 存在。 0 I²C 模块 4 不存在。
3	P3	RO	0x1	I²C 模块 3 存在 值 描述 1 I²C 模块 3 存在。 0 I²C 模块 3 不存在。
2	P2	RO	0x1	I²C 模块 2 存在 值 描述 1 I²C 模块 2 存在。 0 I²C 模块 2 不存在。

位/域	名称	类型	复位	描述
1	P1	RO	0x1	I^2C 模块 1 存在 值 描述 1 I^2C 模块 1 存在。 0 I^2C 模块 1 不存在。
0	P0	RO	0x1	I^2C 模块 0 存在 值 描述 1 I^2C 模块 0 存在。 0 I^2C 模块 0 不存在。

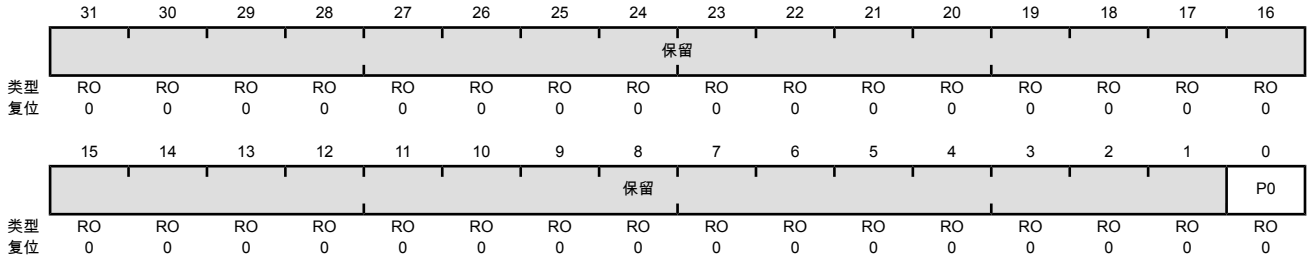
寄存器 34: 通用串行总线外设存在寄存器（PPUSB），偏移量 0x328

PPUSB 寄存器提供关于 USB 模块的软件信息。

重要: 应使用该寄存器确定是否在该微控制器上执行 USB 模块。但是，要支持传统软件，可使用 DC6 寄存器。读取 DC6 寄存器即可正确识别 USB 模块是否存在。

通用串行总线外设存在寄存器 (PPUSB)

基址 0x400F.E000
偏移量 0x328
类型 RO, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:1	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
0	P0	RO	0x0	USB 模块存在 值 描述 1 USB 模块存在。 0 USB 模块不存在。

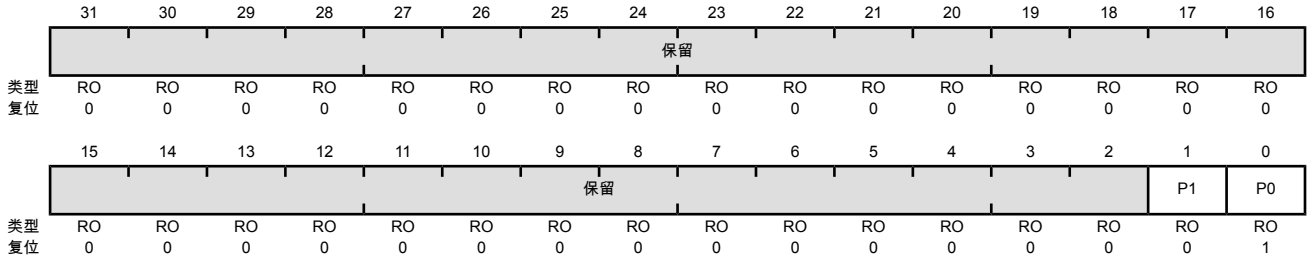
寄存器 35: 控制器局域网外设存在寄存器（PPCAN），偏移量 0x334

PPCAN 寄存器提供关于 CAN 模块的软件信息。

重要: 应使用该寄存器确定在该微控制器上执行的 CAN 模块。但是，要支持传统软件，可使用 DC1 寄存器。读取 DC1 寄存器即可正确识别传统 CAN 模块是否存在。

控制器局域网外设存在寄存器 (PPCAN)

基址 0x400F.E000
偏移量 0x334
类型 RO, 复位 0x0000.0001



位/域	名称	类型	复位	描述
31:2	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
1	P1	RO	0x0	CAN 模块 1 存在 值 描述 1 CAN 模块 1 存在。 0 CAN 模块 1 不存在。
0	P0	RO	0x1	CAN 模块 0 存在 值 描述 1 CAN 模块 0 存在。 0 CAN 模块 0 不存在。

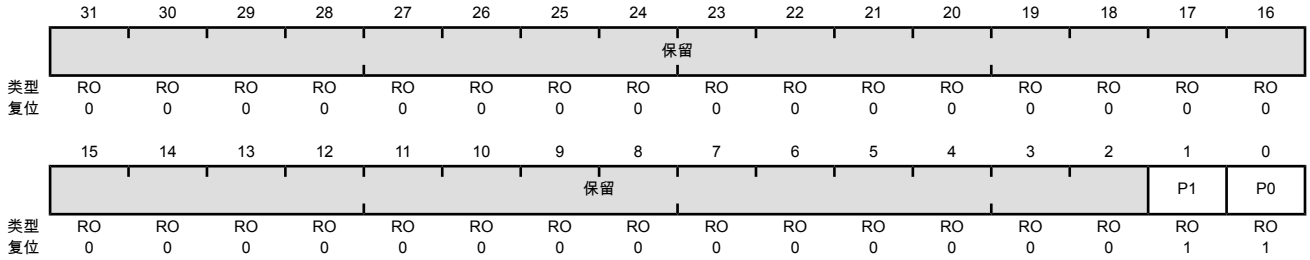
寄存器 36: 模数转换器外设存在寄存器（PPADC），偏移量 0x338

PPADC 寄存器提供关于 ADC 模块的软件信息。

重要: 应使用该寄存器确定在该微控制器上执行的 ADC 模块。但是，要支持传统软件，可使用 DC1 寄存器。读取 DC1 寄存器即可正确识别传统 ADC 模块是否存在。

模数转换器外设存在寄存器 (PPADC)

基址 0x400F.E000
偏移量 0x338
类型 RO, 复位 0x0000.0003



位/域	名称	类型	复位	描述
31:2	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
1	P1	RO	0x1	ADC 模块 1 存在 值 描述 1 ADC 模块 1 存在。 0 ADC 模块 1 不存在。
0	P0	RO	0x1	ADC 模块 0 存在 值 描述 1 ADC 模块 0 存在。 0 ADC 模块 0 不存在。

寄存器 37: 模拟比较器外设存在寄存器（PPACMP），偏移量 0x33C

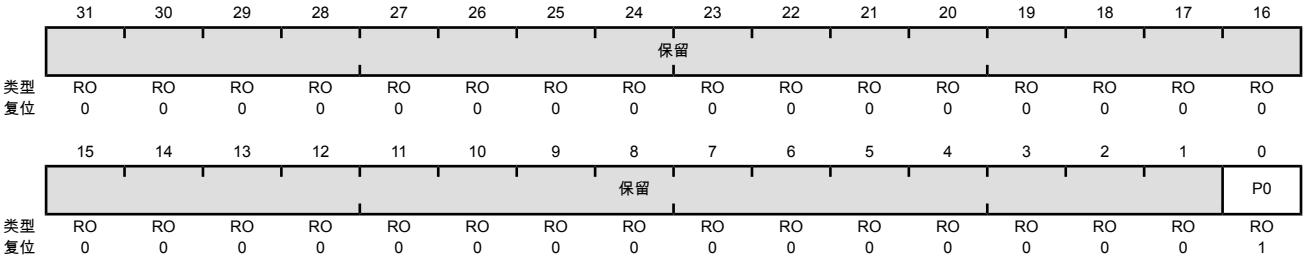
PPACMP 寄存器提供关于模拟比较器模块的软件信息。

重要: 应使用该寄存器确定是否在该微控制器上执行模拟比较器模块。但是，要支持传统软件，可使用 DC2 寄存器。读取 DC2 寄存器即可正确识别模拟比较器模块是否存在。

请注意，模拟比较器外设属性 (ACMPPP) 寄存器可指示模块中包含的模拟比较器块数量。

模拟比较器外设存在寄存器 (PPACMP)

基址 0x400F.E000
偏移量 0x33C
类型 RO, 复位 0x0000.0001



位/域	名称	类型	复位	描述
31:1	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
0	P0	RO	0x1	模拟比较器模块存在 值 描述 1 模拟比较器模块存在。 0 模拟比较器模块不存在。

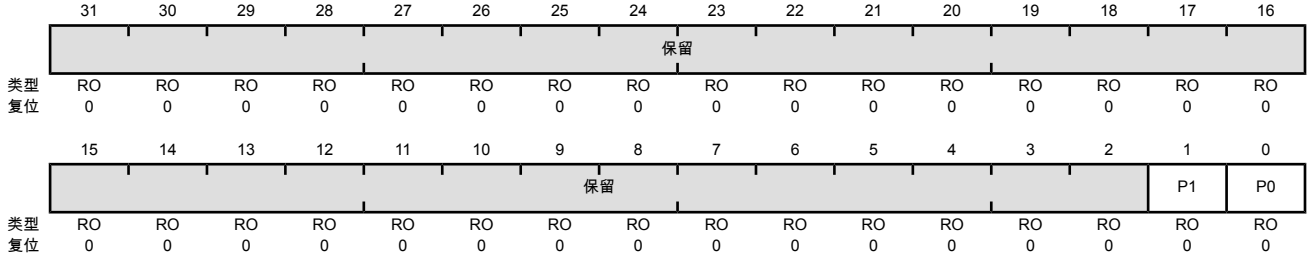
寄存器 38: 脉宽调解器外设存在寄存器（PPPWM），偏移量 0x340

PPPWM 寄存器提供关于 PWM 模块的软件信息。

重要: 应使用该寄存器确定在该微控制器上执行的 PWM 模块。但是，要支持传统软件，可使用 DC1 寄存器。读取 DC1 寄存器即可正确识别传统 PWM 模块是否存在。软件必须使用该寄存器来确定不受 DC1 寄存器支持的模块是否存在。

脉宽调解器外设存在寄存器 (PPPWM)

基址 0x400F.E000
偏移量 0x340
类型 RO, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:2	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
1	P1	RO	0x0	PWM 模块 1 存在 值 描述 1 PWM 模块 1 存在。 0 PWM 模块 1 不存在。
0	P0	RO	0x0	PWM 模块 0 存在 值 描述 1 PWM 模块 0 存在。 0 PWM 模块 0 不存在。

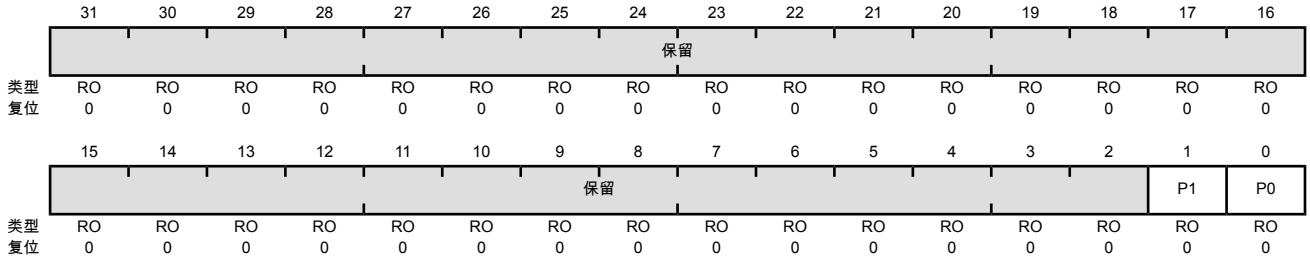
寄存器 39: 正交编码器接口外设存在寄存器（PPQEI），偏移量 0x344

PPQEI 寄存器提供关于 QEI 模块的软件信息。

重要: 应使用该寄存器确定在该微控制器上执行的 QEI 模块。但是，要支持传统软件，可使用 DC2 寄存器。读取 DC2 寄存器即可正确识别传统 QEI 模块是否存在。

正交编码器接口外设存在寄存器 (PPQEI)

基址 0x400F.E000
偏移量 0x344
类型 RO, 复位 0x0000.0000



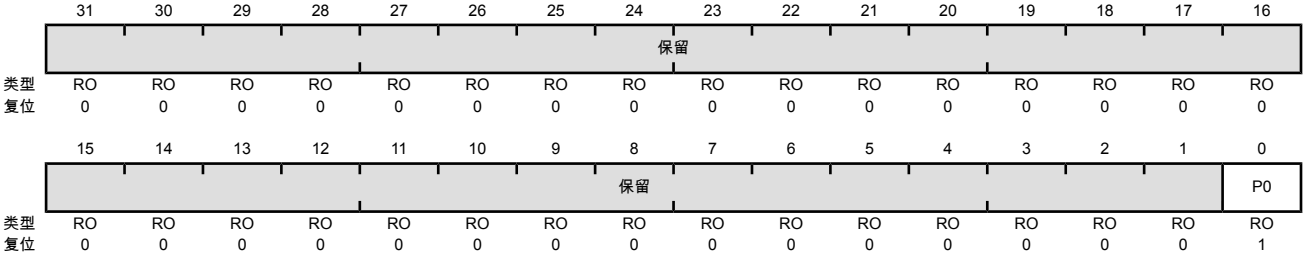
位/域	名称	类型	复位	描述
31:2	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
1	P1	RO	0x0	QEI模块1存在 值 描述 1 QEI 模块 1 存在。 0 QEI 模块 1 不存在。
0	P0	RO	0x0	QEI模块0存在 值 描述 1 QEI 模块 0 存在。 0 QEI 模块 0 不存在。

寄存器 40: EEPROM 外设存在寄存器（PPEEPROM），偏移量 0x358

PPEEPROM 寄存器提供关于 EEPROM 模块的软件信息。

EEPROM 外设存在寄存器 (PPEEPROM)

基址 0x400F.E000
偏移量 0x358
类型 RO, 复位 0x0000.0001



位/域	名称	类型	复位	描述
31:1	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
0	P0	RO	0x1	EEPROM 模块存在

值 描述

1 EEPROM 模块存在。

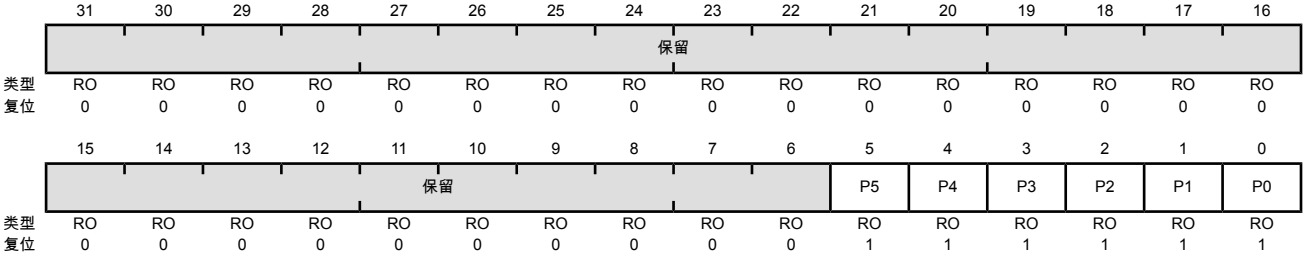
0 EEPROM 模块不存在。

寄存器 41: 32/64 位宽通用定时器外设存在寄存器（PPWTIMER），偏移量 0x35C

PPWTIMER 寄存器提供关于 32/64 位宽通用定时器模块的软件信息。

32/64 位宽通用定时器外设存在寄存器 (PPWTIMER)

基址 0x400F.E000
偏移量 0x35C
类型 RO, 复位 0x0000.003F



位/域	名称	类型	复位	描述
31:6	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
5	P5	RO	0x1	32/64 位宽通用定时器 5 存在 值 描述 1 32/64 位宽通用定时器模块 5 存在。 0 32/64 位宽通用定时器模块 5 不存在。
4	P4	RO	0x1	32/64 位宽通用定时器 4 存在 值 描述 1 32/64 位宽通用定时器模块 4 存在。 0 32/64 位宽通用定时器模块 4 不存在。
3	P3	RO	0x1	32/64 位宽通用定时器 3 存在 值 描述 1 32/64 位宽通用定时器模块 3 存在。 0 32/64 位宽通用定时器模块 3 不存在。
2	P2	RO	0x1	32/64 位宽通用定时器 2 存在 值 描述 1 32/64 位宽通用定时器模块 2 存在。 0 32/64 位宽通用定时器模块 2 不存在。
1	P1	RO	0x1	32/64 位宽通用定时器 1 存在 值 描述 1 32/64 位宽通用定时器模块 1 存在。 0 32/64 位宽通用定时器模块 1 不存在。

位/域	名称	类型	复位	描述
0	P0	RO	0x1	32/64 位宽通用定时器 0 存在 值 描述 1 32/64 位宽通用定时器模块 0 存在。 0 32/64 位宽通用定时器模块 0 不存在。

寄存器 42: 看门狗定时器软件复位寄存器（SRWD），偏移量 0x500

SRWD 寄存器为软件提供复位可用看门狗模块的功能。该寄存器特别为看门狗模块提供与传统软件复位控制 n SRCRn 寄存器相同的功能，并且具有与相应 SRCRn 位相同的位极性。

通过软件使用简单的两步过程对外设进行复位：

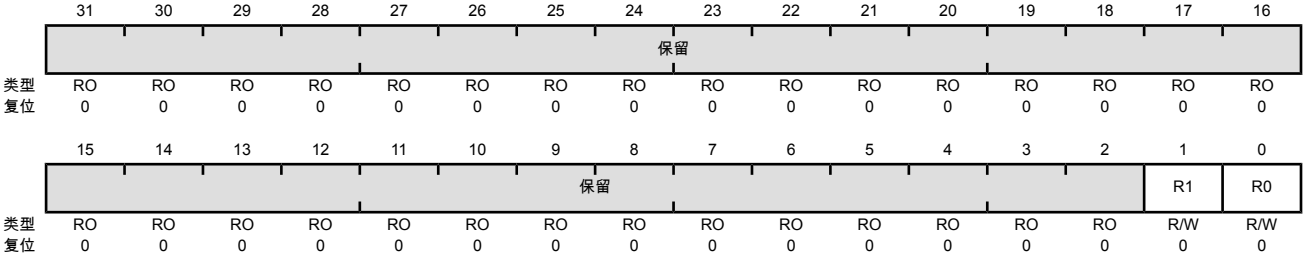
1. 软件置位 SRWD 寄存器中的位。SRWD 位为 1 时，外设保持在复位状态。
2. 软件通过清零 SRWD 位完成复位过程。

从清零 SRWD 位到外设就绪时可能会有延迟。软件可以检查相应的 PRWD 位了解情况。

重要: 应使用该寄存器复位看门狗模块。要支持传统软件，可使用 SRCR0 寄存器。置位 SRCR0 寄存器中的位也会复位相应模块。通过对该 SRCR0 寄存器的写操作更改的任何位都可以在对 SRCR0 寄存器进行读操作时进行正确回读。如果软件使用该寄存器复位传统外设（例如 Watchdog 1），则写操作会产生正确操作，但是该位的值不在 SRCR0 寄存器中得到反映。如果软件使用传统和外设专用寄存器访问，则必须通过读-修改-写的操作来访问外设专用寄存器，因为该操作仅影响不在传统寄存器中的外设。通过这种方法，外设专用和传统寄存器都具有一致的信息。

看门狗定时器软件复位寄存器 (SRWD)

基址 0x400F.E000
偏移量 0x500
类型 R/W, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:2	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
1	R1	R/W	0	看门狗定时器 1 软件复位 值 描述 1 看门狗模块 1 复位。 0 看门狗模块 1 未复位。
0	R0	R/W	0	看门狗定时器 0 软件复位 值 描述 1 看门狗模块 0 复位。 0 看门狗模块 0 未复位。

寄存器 43: 16/32 位通用定时器软件复位寄存器（SRTIMER），偏移量 0x504

SRTIMER 寄存器为软件提供复位可用 16/32 位定时器模块的功能。该寄存器特别为定时器模块提供与传统软件复位控制 n SRCRn 寄存器相同的功能，并且具有与相应 SRCRn 位相同的位极性。

通过软件使用简单的两步过程对外设进行复位：

1. 软件置位 SRTIMER 寄存器中的位。SRTIMER 位为 1 时，外设保持在复位状态。
2. 软件通过清零 SRTIMER 位完成复位过程。

从清零 SRTIMER 位到外设就绪可能会有延迟。软件可以检查相应的 PRTIMER 位以确保操作正确。

重要: 应使用该寄存器复位定时器模块。要支持传统软件，可使用 SRCR1 寄存器。置位 SRCR1 寄存器中的位也会复位相应模块。通过对该 SRCR1 寄存器的写操作更改的任何位都可以在对 SRCR1 寄存器进行读操作时进行正确回读。软件必须使用该寄存器复位不处于传统寄存器中的模块。如果软件使用该寄存器复位传统外设（如 Timer 1），则写操作会产生正确操作，但是该位的值不在 SRCR1 寄存器中得到反映。如果软件使用传统和外设专用寄存器访问，则必须通过读-修改-写的操作来访问外设专用寄存器，因为该操作仅影响不在传统寄存器中的外设。通过这种方法，外设专用和传统寄存器都具有一致的信息。

16/32 位通用定时器软件复位寄存器 (SRTIMER)

基址 0x400F.E000
偏移量 0x504
类型 R/W, 复位 0x0000.0000

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
域	保留															
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
域	保留										R5	R4	R3	R2	R1	R0
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W	R/W	R/W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位/域	名称	类型	复位	描述
31:6	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
5	R5	R/W	0	16/32 位通用定时器 5 软件复位 值 描述 1 16/32 位通用定时器模块 5 复位。 0 16/32 位通用定时器模块 5 未复位。
4	R4	R/W	0	16/32 位通用定时器 4 软件复位 值 描述 1 16/32 位通用定时器模块 4 复位。 0 16/32 位通用定时器模块 4 未复位。

位/域	名称	类型	复位	描述
3	R3	R/W	0	16/32 位通用定时器 3 软件复位 值 描述 1 16/32 位通用定时器模块 3 复位。 0 16/32 位通用定时器模块 3 未复位。
2	R2	R/W	0	16/32 位通用定时器 2 软件复位 值 描述 1 16/32 位通用定时器模块 2 复位。 0 16/32 位通用定时器模块 2 未复位。
1	R1	R/W	0	16/32 位通用定时器 1 软件复位 值 描述 1 16/32 位通用定时器模块 1 复位。 0 16/32 位通用定时器模块 1 未复位。
0	R0	R/W	0	16/32 位通用定时器 0 软件复位 值 描述 1 16/32 位通用定时器模块 0 复位。 0 16/32 位通用定时器模块 0 未复位。

寄存器 44: 通用输入/输出软件复位寄存器（SRGPIO），偏移量 0x508

SRGPIO 寄存器为软件提供复位可用 GPIO 模块的功能。该寄存器特别为 GPIO 模块提供与传统软件复位控制 n SRCRn 寄存器相同的功能，并且具有与相应 SRCRn 位相同的位极性。

通过软件使用简单的两步过程对外设进行复位：

1. 软件置位 SRGPIO 寄存器中的位。SRGPIO 位为 1 时，外设保持在复位状态。
2. 软件通过清零 SRGPIO 位完成复位过程。

从清零 SRGPIO 位到外设就绪时可能会有延迟。软件可以检查相应的 PRGPIO 位以了解情况。

重要: 应使用该寄存器复位 GPIO 模块。要支持传统软件，可使用 SRCR2 寄存器。置位 SRCR2 寄存器中的位也会复位相应模块。通过对该 SRCR2 寄存器的写操作更改的任何位都可以在对 SRCR2 寄存器进行读操作时进行正确回读。软件必须使用该寄存器复位不处于传统寄存器中的模块。如果软件使用该寄存器复位传统外设（如 GPIO A），则写操作会产生正确操作，但是该位的值不在 SRCR2 寄存器中得到反映。如果软件使用传统和外设专用寄存器访问，则必须通过读-修改-写的操作来访问外设专用寄存器，因为该操作仅影响不在传统寄存器中的外设。通过这种方法，外设专用和传统寄存器都具有一致的信息。

通用输入/输出软件复位寄存器 (SRGPIO)

基址 0x400F.E000
偏移量 0x508
类型 R/W, 复位 0x0000.0000

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	保留															
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	保留										R5	R4	R3	R2	R1	R0
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W	R/W	R/W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位/域	名称	类型	复位	描述
31:6	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
5	R5	R/W	0	GPIO 端口 F 软件复位 值 描述 1 GPIO 端口 F 复位。 0 GPIO 端口 F 未复位。
4	R4	R/W	0	GPIO 端口 E 软件复位 值 描述 1 GPIO 端口 E 复位。 0 GPIO 端口 E 未复位。

位/域	名称	类型	复位	描述
3	R3	R/W	0	GPIO 端口 D 软件复位 值 描述 1 GPIO 端口 D 复位。 0 GPIO 端口 D 未复位。
2	R2	R/W	0	GPIO 端口 C 软件复位 值 描述 1 GPIO 端口 C 复位。 0 GPIO 端口 C 未复位。
1	R1	R/W	0	GPIO 端口 B 软件复位 值 描述 1 GPIO 端口 B 复位。 0 GPIO 端口 B 未复位。
0	R0	R/W	0	GPIO 端口 A 软件复位 值 描述 1 GPIO 端口 A 复位。 0 GPIO 端口 A 未复位。

寄存器 45: 微型直接存储器访问软件复位寄存器（SRDMA），偏移量 0x50C

SRDMA 寄存器为软件提供复位可用 µDMA 模块的功能。该寄存器特别为 µDMA 模块提供与传统软件复位控制 n SRCRn 寄存器相同的功能，并且具有与相应 SRCRn 位相同的位极性。

通过软件使用简单的两步过程对外设进行复位：

1. 软件置位 SRDMA 寄存器中的位。SRDMA 位为 1 时，外设保持在复位状态。
2. 软件通过清零 SRDMA 位完成复位过程。

从清零 SRDMA 位到外设就绪时可能会有延迟。软件可以检查相应的 PRDMA 位以了解情况。

重要: 应使用该寄存器复位 µDMA 模块。要支持传统软件，可使用 SRCR2 寄存器。置位 SRCR2 寄存器中的 UDMA 位也会复位 µDMA 模块。如果通过对 SRCR2 寄存器的写操作置位 UDMA 位，它可以在对 SRCR2 寄存器进行读操作时进行正确回读。如果软件使用该寄存器复位 µDMA 模块，则写操作会产生正确操作，但是 UDMA 位的值不在 SRCR2 寄存器中得到反映。如果软件使用传统和外设专用寄存器访问，则必须通过读-修改-写的操作来访问外设专用寄存器，因为该操作仅影响不在传统寄存器中的外设。通过这种方法，外设专用和传统寄存器都具有一致的信息。

微型直接存储器访问软件复位寄存器 (SRDMA)

基址 0x400F.E000
偏移量 0x50C
类型 R/W, 复位 0x0000.0000

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	保留															
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	保留															R0
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位/域	名称	类型	复位	描述
31:1	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
0	R0	R/W	0	µDMA 模块软件复位 值 描述 1 µDMA 模块复位。 0 µDMA 模块未复位。

寄存器 46: 休眠软件复位寄存器（SRHIB），偏移量 0x514

SRHIB 寄存器为软件提供复位可用休眠模块的功能。该寄存器特别为休眠模块提供与传统软件复位控制 n SRCRn 寄存器相同的功能，并且具有与相应 SRCRn 位相同的位极性。

通过软件使用简单的两步过程对外设进行复位：

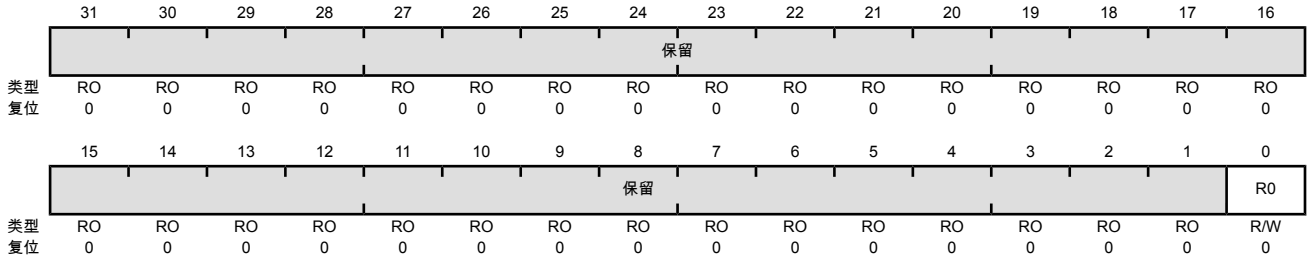
1. 软件置位 SRHIB 寄存器中的位。SRHIB 位为 1 时，外设保持在复位状态。
2. 软件通过清零 SRHIB 位完成复位过程。

从清零 SRHIB 位到外设就绪时可能会有延迟。软件可以检查相应的 PRHIB 位以了解情况。

重要: 应使用该寄存器复位休眠模块。要支持传统软件，可使用 SRCR0 寄存器。置位 SRCR0 寄存器中的 HIB 位也会复位休眠模块。如果通过对 SRCR0 寄存器的写操作置位 HIB 位，它可以在对 SRCR0 寄存器进行读操作时进行正确回读。如果软件使用该寄存器复位休眠模块，则写操作会产生正确操作，但是 HIB 位的值不在 SRCR0 寄存器中得到反映。如果软件使用传统和外设专用寄存器访问，则必须通过读-修改-写的操作来访问外设专用寄存器，因为该操作仅影响不在传统寄存器中的外设。通过这种方法，外设专用和传统寄存器都具有一致的信息。

休眠软件复位寄存器 (SRHIB)

基址 0x400F.E000
偏移量 0x514
类型 R/W, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:1	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
0	R0	R/W	0	休眠模块软件复位 值 描述 1 休眠模块复位。 0 休眠模块未复位。

寄存器 47: 通用异步收发器软件复位寄存器（SRUART），偏移量 0x518

SRUART 寄存器为软件提供复位可用 UART 模块的功能。该寄存器特别为 UART 模块提供与传统软件复位控制 n SRCRn 寄存器相同的功能，并且具有与相应 SRCRn 位相同的位极性。

通过软件使用简单的两步过程对外设进行复位：

1. 软件置位 SRUART 寄存器中的位。SRUART 位为 1 时，外设保持在复位状态。
2. 软件通过清零 SRUART 位完成复位过程。

从清零 SRUART 位到外设就绪时可能会有延迟。软件可以检查相应的 PRUART 位以了解情况。

重要: 应使用该寄存器复位 UART 模块。要支持传统软件，可使用 SRCR1 寄存器。置位 SRCR1 寄存器中的位也会复位相应模块。通过对该 SRCR1 寄存器的写操作更改的任何位都可以在对 SRCR1 寄存器进行读操作时进行正确回读。软件必须使用该寄存器复位不处于传统寄存器中的模块。如果软件使用该寄存器复位传统外设（如 UART0），则写操作会产生正确操作，但是该位的值不在 SRCR1 寄存器中得到反映。如果软件使用传统和外设专用寄存器访问，则必须通过读-修改-写的操作来访问外设专用寄存器，因为该操作仅影响不在传统寄存器中的外设。通过这种方法，外设专用和传统寄存器都具有一致的信息。

通用异步收发器软件复位寄存器 (SRUART)

基址 0x400F.E000
偏移量 0x518
类型 R/W, 复位 0x0000.0000

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	保留															
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	保留								R7	R6	R5	R4	R3	R2	R1	R0
类型	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位/域	名称	类型	复位	描述
31:8	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
7	R7	R/W	0	UART 模块 7 软件复位 值 描述 1 UART 模块 7 复位。 0 UART 模块 7 未复位。
6	R6	R/W	0	UART 模块 6 软件复位 值 描述 1 UART 模块 6 复位。 0 UART 模块 6 未复位。

位/域	名称	类型	复位	描述
5	R5	R/W	0	UART 模块 5 软件复位 值 描述 1 UART 模块 5 复位。 0 UART 模块 5 未复位。
4	R4	R/W	0	UART 模块 4 软件复位 值 描述 1 UART 模块 4 复位。 0 UART 模块 4 未复位。
3	R3	R/W	0	UART 模块 3 软件复位 值 描述 1 UART 模块 3 复位。 0 UART 模块 3 未复位。
2	R2	R/W	0	UART 模块 2 软件复位 值 描述 1 UART 模块 2 复位。 0 UART 模块 2 未复位。
1	R1	R/W	0	UART 模块 1 软件复位 值 描述 1 UART 模块 1 复位。 0 UART 模块 1 未复位。
0	R0	R/W	0	UART 模块 0 软件复位 值 描述 1 UART 模块 0 复位。 0 UART 模块 0 未复位。

寄存器 48: 同步串行接口软件复位寄存器（SRSSI），偏移量 0x51C

SRSSI 寄存器为软件提供复位可用 SSI 模块的功能。该寄存器特别为 SSI 模块提供与传统软件复位控制 n SRCRn 寄存器相同的功能，并且具有与相应 SRCRn 位相同的位极性。

通过软件使用简单的两步过程对外设进行复位：

1. 软件置位 SRSSI 寄存器中的位。SRSSI 位为 1 时，外设保持在复位状态。
2. 软件通过清零 SRSSI 位完成复位过程。

从清零 SRSSI 位到外设就绪时可能会有延迟。软件可以检查相应的 PRSSI 位以了解情况。

重要: 应使用该寄存器复位 SSI 模块。要支持传统软件，可使用 SRCR1 寄存器。置位 SRCR1 寄存器中的位也会复位相应模块。通过对该 SRCR1 寄存器的写操作更改的任何位都可以在对 SRCR1 寄存器进行读操作时进行正确回读。软件必须使用该寄存器复位不处于传统寄存器中的模块。如果软件使用该寄存器复位传统外设（如 SSI0），则写操作会产生正确操作，但是该位的值不在 SRCR1 寄存器中得到反映。如果软件使用传统和外设专用寄存器访问，则必须通过读-修改-写的操作来访问外设专用寄存器，因为该操作仅影响不在传统寄存器中的外设。通过这种方法，外设专用和传统寄存器都具有一致的信息。

同步串行接口软件复位寄存器 (SRSSI)

基址 0x400F.E000
偏移量 0x51C
类型 R/W, 复位 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	保留															
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	保留												R3	R2	R1	R0
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位/域	名称	类型	复位	描述
31:4	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
3	R3	R/W	0	SSI 模块 3 软件复位 值 描述 1 SSI 模块 3 复位。 0 SSI 模块 3 未复位。
2	R2	R/W	0	SSI 模块 2 软件复位 值 描述 1 SSI 模块 2 复位。 0 SSI 模块 2 未复位。

位/域	名称	类型	复位	描述
1	R1	R/W	0	SSI 模块 1 软件复位 值 描述 1 SSI 模块 1 复位。 0 SSI 模块 1 未复位。
0	R0	R/W	0	SSI 模块 0 软件复位 值 描述 1 SSI 模块 0 复位。 0 SSI 模块 0 未复位。

寄存器 49: 内部集成电路软件复位寄存器（SRI2C），偏移量 0x520

SRI2C 寄存器为软件提供复位可用 I^2C 模块的功能。该寄存器特别为 I^2C 模块提供与传统软件复位控制 n SRCRn 寄存器相同的功能，并且具有与相应 SRCRn 位相同的位极性。

通过软件使用简单的两步过程对外设进行复位：

1. 软件置位 SRI2C 寄存器中的位。SRI2C 位为 1 时，外设保持在复位状态。
2. 软件通过清零 SRI2C 位完成复位过程。

从清零 SRI2C 位到外设就绪时可能会有延迟。软件可以检查相应的 PRI2C 位以了解情况。

重要: 应使用该寄存器复位 I^2C 模块。要支持传统软件，可使用 SRCR1 寄存器。置位 SRCR1 寄存器中的位也会复位相应模块。通过对该 SRCR1 寄存器的写操作更改的任何位都可以在对 SRCR1 寄存器进行读操作时进行正确回读。软件必须使用该寄存器复位不处于传统寄存器中的模块。如果软件使用该寄存器复位传统外设（如 I2C0），则写操作会产生正确操作，但是该位的值不在 SRCR1 寄存器中得到反映。如果软件使用传统和外设专用寄存器访问，则必须通过读-修改-写的操作来访问外设专用寄存器，因为该操作仅影响不在传统寄存器中的外设。通过这种方法，外设专用和传统寄存器都具有一致的信息。

内部集成电路软件复位寄存器 (SRI2C)

基址 0x400F.E000
偏移量 0x520
类型 R/W, 复位 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	保留															
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	保留												R3	R2	R1	R0
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位/域	名称	类型	复位	描述
31:4	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
3	R3	R/W	0	I^2C 模块 3 软件复位 值 描述 1 I^2C 模块 3 复位。 0 I^2C 模块 3 未复位。
2	R2	R/W	0	I^2C 模块 2 软件复位 值 描述 1 I^2C 模块 2 复位。 0 I^2C 模块 2 未复位。

位/域	名称	类型	复位	描述
1	R1	R/W	0	I^2C 模块 1 软件复位 值 描述 1 I^2C 模块 1 复位。 0 I^2C 模块 1 未复位。
0	R0	R/W	0	I^2C 模块 0 软件复位 值 描述 1 I^2C 模块 0 复位。 0 I^2C 模块 0 未复位。

寄存器 50: 控制器局域网软件复位寄存器（SRCAN），偏移量 0x534

SRCAN 寄存器为软件提供复位可用 CAN 模块的功能。该寄存器特别为 CAN 模块提供与传统软件复位控制 n SRCRn 寄存器相同的功能，并且具有与相应 SRCRn 位相同的位极性。

通过软件使用简单的两步过程对外设进行复位：

1. 软件置位 SRCAN 寄存器中的位。SRCAN 位为 1 时，外设保持在复位状态。
2. 软件通过清零 SRCAN 位完成复位过程。

从清零 SRCAN 位到外设就绪时可能会有延迟。软件可以检查相应的 PRCAN 位以了解情况。

重要: 应使用该寄存器复位 CAN 模块。要支持传统软件，可使用 SRCR0 寄存器。置位 SRCR0 寄存器中的位也会复位相应模块。通过对该 SRCR0 寄存器的写操作更改的任何位都可以在对 SRCR0 寄存器进行读操作时进行正确回读。如果软件使用该寄存器复位传统外设（如 CAN0），则写操作会产生正确操作，但是该位的值不在 SRCR0 寄存器中得到反映。如果软件使用传统和外设专用寄存器访问，则必须通过读-修改-写的操作来访问外设专用寄存器，因为该操作仅影响不在传统寄存器中的外设。通过这种方法，外设专用和传统寄存器都具有一致的信息。

控制器局域网软件复位寄存器 (SRCAN)

基址 0x400F.E000
偏移量 0x534
类型 R/W, 复位 0x0000.0000

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
域	保留															
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
域	保留															R0
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位/域	名称	类型	复位	描述
31:1	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
0	R0	R/W	0	CAN 模块 0 软件复位 值 描述 1 CAN 模块 0 复位。 0 CAN 模块 0 未复位。

寄存器 51: 模数转换器软件复位寄存器（SRADC），偏移量 0x538

SRADC 寄存器为软件提供复位可用 ADC 模块的功能。该寄存器特别为 ADC 模块提供与传统软件复位控制 n SRCRn 寄存器相同的功能，并且具有与相应 SRCRn 位相同的位极性。

通过软件使用简单的两步过程对外设进行复位：

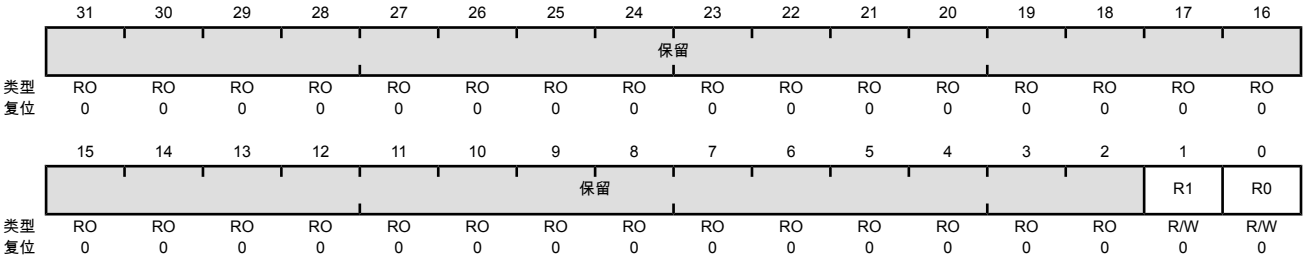
1. 软件置位 SRADC 寄存器中的位。SRADC 位为 1 时，外设保持在复位状态。
2. 软件通过清零 SRADC 位完成复位过程。

从清零 SRADC 位到外设就绪时可能会有延迟。软件可以检查相应的 PRADC 位以确保正确。

重要: 应使用该寄存器复位 ADC 模块。要支持传统软件，可使用 SRCR0 寄存器。置位 SRCR0 寄存器中的位也会复位相应模块。通过对该 SRCR0 寄存器的写操作更改的任何位都可以在对 SRCR0 寄存器进行读操作时进行正确回读。如果软件使用该寄存器复位传统外设（如 ADC0），则写操作会产生正确操作，但是该位的值不在 SRCR0 寄存器中得到反映。如果软件使用传统和外设专用寄存器访问，则必须通过读-修改-写的操作来访问外设专用寄存器，因为该操作仅影响不在传统寄存器中的外设。通过这种方法，外设专用和传统寄存器都具有一致的信息。

模数转换器软件复位寄存器 (SRADC)

基址 0x400F.E000
偏移量 0x538
类型 R/W, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:2	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
1	R1	R/W	0	ADC 模块 1 软件复位 值 描述 1 ADC 模块 1 复位。 0 ADC 模块 1 未复位。
0	R0	R/W	0	ADC 模块 0 软件复位 值 描述 1 ADC 模块 0 复位。 0 ADC 模块 0 未复位。

寄存器 52: 模数比较器软件复位寄存器（SRACMP），偏移量 0x53C

SRACMP 寄存器为软件提供复位可用模拟比较器模块的功能。该寄存器特别为模拟比较器模块提供与传统软件复位控制 n SRCRn 寄存器相同的功能，并且具有与相应 SRCRn 位相同的位极性。

通过软件使用简单的两步过程对模块进行复位：

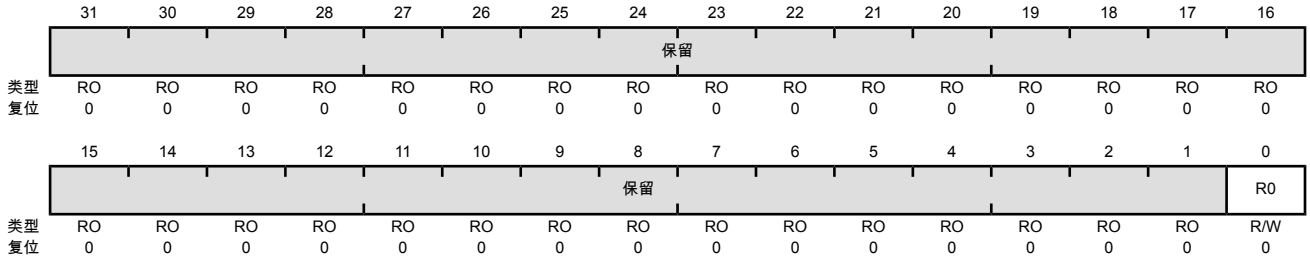
1. 软件置位 SRACMP 寄存器中的位。SRACMP 位为 1 时，模块保持在复位状态。
2. 软件通过清零 SRACMP 位完成复位过程。

从清零 SRACMP 位到模块就绪时可能会有延迟。软件可以检查相应的 PRACMP 位以了解情况。

重要: 应使用该寄存器复位模拟比较器模块。要支持传统软件，可使用 SRCR1 寄存器。置位 SRCR0 寄存器中的任何 COMPn 位也会复位模拟比较器模块。如果通过对 SRCR1 寄存器的写操作置位任何 COMPn 位，它可以在对 SRCR0 寄存器进行读操作时进行正确回读。如果软件使用该寄存器复位模拟比较器模块，则写操作会产生正确操作，但是 SRCR1 寄存器的 COMPn 位不反映 R0 的值。如果软件使用传统和外设专用寄存器访问，则必须通过读-修改-写的操作来访问外设专用寄存器，因为该操作仅影响不在传统寄存器中的外设。通过这种方法，外设专用和传统寄存器都具有一致的信息。

模数比较器软件复位寄存器 (SRACMP)

基址 0x400F.E000
偏移量 0x53C
类型 R/W, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:1	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
0	R0	R/W	0	模拟比较器模块 0 软件复位 值 描述 1 模拟比较器模块复位。 0 模拟比较器模块未复位。

寄存器 53: EEPROM 软件复位寄存器（SREEPROM），偏移量 0x558

SREEPROM 寄存器为软件提供复位可用 EEPROM 模块的功能。

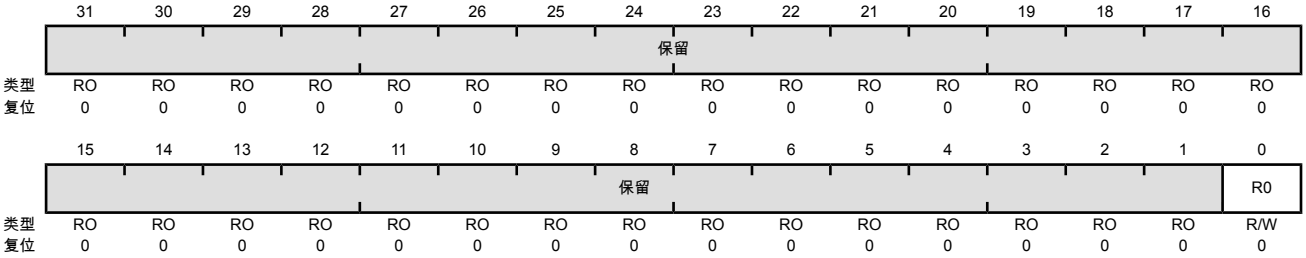
通过软件使用简单的两步过程对外设进行复位：

1. 软件置位 SREEPROM 寄存器中的位。SREEPROM 位为 1 时，外设保持在复位状态。
2. 软件通过清零 SREEPROM 位完成复位过程。

从清零 SREEPROM 位到外设就绪时可能会有延迟。软件可以检查相应的 PREEPROM 位以确保正确。

EEPROM 软件复位寄存器 (SREEPROM)

基址 0x400F.E000
偏移量 0x558
类型 R/W, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:1	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
0	R0	R/W	0	EEPROM 模块软件复位 值 描述 1 EEPROM 模块复位。 0 EEPROM 模块未复位。

寄存器 54: 32/64 位宽通用定时器软件复位寄存器（SRWTIMER），偏移量 0x55C

SRWTIMER 寄存器为软件提供复位可用 32/64 位宽定时器模块的功能。

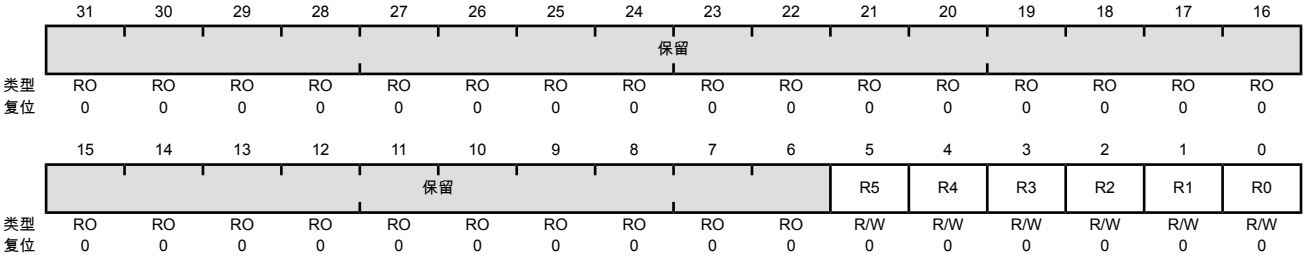
通过软件使用简单的两步过程对外设进行复位：

1. 软件置位 SRWTIMER 寄存器中的位。SRWTIMER 位为 1 时，外设保持在复位状态。
2. 软件通过清零 SRWTIMER 位完成复位过程。

从清零 SRWTIMER 位到外设就绪可能会有延迟。软件可以检查相应的 SRWTIMER 位以确保操作正确。

32/64 位宽通用定时器软件复位寄存器 (SRWTIMER)

基址 0x400F.E000
偏移量 0x55C
类型 R/W, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:6	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
5	R5	R/W	0	32/64 位宽通用定时器 5 软件复位 值 描述 1 32/64 位宽通用定时器模块 5 复位。 0 32/64 位宽通用定时器模块 5 未复位。
4	R4	R/W	0	32/64 位宽通用定时器 4 软件复位 值 描述 1 32/64 位宽通用定时器模块 4 复位。 0 32/64 位宽通用定时器模块 4 未复位。
3	R3	R/W	0	32/64 位宽通用定时器 3 软件复位 值 描述 1 32/64 位宽通用定时器模块 3 复位。 0 32/64 位宽通用定时器模块 3 未复位。

位/域	名称	类型	复位	描述
2	R2	R/W	0	32/64 位宽通用定时器 2 软件复位 值 描述 1 32/64 位宽通用定时器模块 2 复位。 0 32/64 位宽通用定时器模块 2 未复位。
1	R1	R/W	0	32/64 位宽通用定时器 1 软件复位 值 描述 1 32/64 位宽通用定时器模块 1 复位。 0 32/64 位宽通用定时器模块 1 未复位。
0	R0	R/W	0	32/64 位宽通用定时器 0 软件复位 值 描述 1 32/64 位宽通用定时器模块 0 复位。 0 32/64 位宽通用定时器模块 0 未复位。

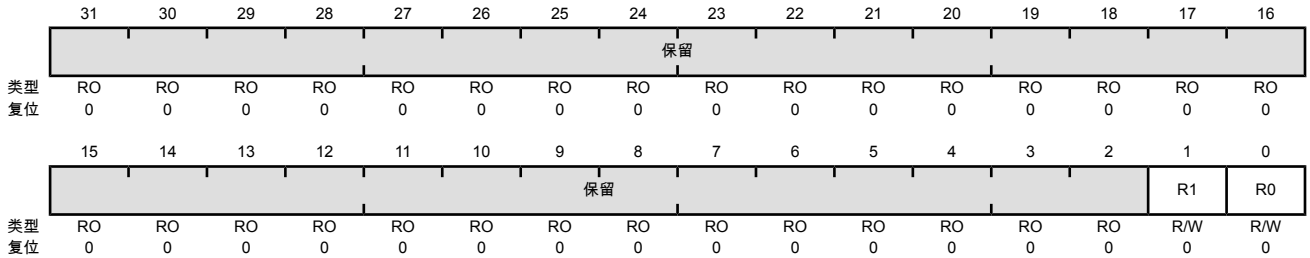
寄存器 55: 看门狗定时器运行模式时钟门控控制寄存器（RCGCWD），偏移量 0x600

RCGCWD 寄存器为软件提供启用和禁用运行模式中看门狗模块的功能。在启用时，为模块提供一个时钟并允许对模块寄存器的访问。在禁用时，时钟禁用以节能，并且对模块寄存器的访问将产生总线错误。该寄存器特别为看门狗模块提供与传统运行模式时钟门控控制 n RCGCn 寄存器相同的功能，并且具有与相应 RCGCn 位相同的位极性。

重要: 应使用该寄存器控制看门狗模块的计时。要支持传统软件，可使用RCGC0 寄存器。对 RCGC0 寄存器执行写操作将同时对该寄存器中的相应位执行写操作。通过对 RCGC0 寄存器的写操作更改的任何位都可以通过对RCGC0 寄存器的读操作进行正确回读。如果软件使用该寄存器对传统外设（如 Watchdog 0）进行写操作，则写操作会产生正确操作，但是该位的值不在RCGC0 寄存器中得到反映。如果软件使用传统和外设专用寄存器访问，则必须通过读-修改-写的操作来访问外设专用寄存器，因为该操作仅影响不在传统寄存器中的外设。通过这种方法，外设专用和传统寄存器都具有一致的信息。

看门狗定时器运行模式时钟门控控制寄存器 (RCGCWD)

基址 0x400F.E000
偏移量 0x600
类型 R/W, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:2	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
1	R1	R/W	0	看门狗定时器 1 运行模式时钟门控控制 值 描述 1 启用运行模式中的看门狗模块 1 并提供时钟。 0 看门狗模块 1 禁用。
0	R0	R/W	0	看门狗定时器 0 运行模式时钟门控控制 值 描述 1 启用运行模式中的看门狗模块 0 并提供时钟。 0 看门狗模块 0 禁用。

寄存器 56: 16/32 位通用定时器运行模式时钟门控控制寄存器（RCGCTIMER），偏移量 0x604

RCGCTIMER 寄存器为软件提供启用和禁用运行模式中的 16/32 位定时器模块的功能。在启用时，为模块提供一个时钟并允许对模块寄存器的访问。在禁用时，时钟禁用以节能，并且对模块寄存器的访问将产生总线错误。该寄存器特别为定时器模块提供与传统运行模式时钟门控控制 n RCGCn 寄存器相同的功能，并且具有与相应 RCGCn 位相同的位极性。

重要: 应使用该寄存器控制定时器模块的计时。要支持传统软件，可使用RCGC1 寄存器。对 RCGC1 寄存器执行写操作将同时对该寄存器中的相应位执行写操作。通过对 RCGC1 寄存器的写操作更改的任何位都可以通过对RCGC1 寄存器的读操作进行正确回读。软件必须使用该寄存器支持不处于传统寄存器中的模块。如果软件使用该寄存器对传统外设（如 Timer 0）进行写操作，则写操作会产生正确操作，但是该位的值不在 RCGC1 寄存器中得到反映。如果软件使用传统和外设专用寄存器访问，则必须通过读-修改-写的操作来访问外设专用寄存器，因为该操作仅影响不在传统寄存器中的外设。通过这种方法，外设专用和传统寄存器都具有一致的信息。

16/32 位通用定时器运行模式时钟门控控制寄存器 (RCGCTIMER)

基址 0x400F.E000
偏移量 0x604
类型 R/W, 复位 0x0000.0000

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	保留															
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	保留										R5	R4	R3	R2	R1	R0
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W	R/W	R/W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位/域	名称	类型	复位	描述
31:6	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
5	R5	R/W	0	16/32 位通用定时器 5 运行模式时钟门控控制 值 描述 1 启用运行模式中的 16/32 位通用定时器模块 5 并提供时钟。 0 16/32 位通用定时器模块 5 禁用。
4	R4	R/W	0	16/32 位通用定时器 4 运行模式时钟门控控制 值 描述 1 启用运行模式中的 16/32 位通用定时器模块 4 并提供时钟。 0 16/32 位通用定时器模块 4 禁用。
3	R3	R/W	0	16/32 位通用定时器 3 运行模式时钟门控控制 值 描述 1 启用运行模式中的 16/32 位通用定时器模块 3 并提供时钟。 0 16/32 位通用定时器模块 3 禁用。

位/域	名称	类型	复位	描述
2	R2	R/W	0	16/32 位通用定时器 2 运行模式时钟门控控制 值 描述 1 启用运行模式中的 16/32 位通用定时器模块 2 并提供时钟。 0 16/32 位通用定时器模块 2 禁用。
1	R1	R/W	0	16/32 位通用定时器 1 运行模式时钟门控控制 值 描述 1 启用运行模式中的 16/32 位通用定时器模块 1 并提供时钟。 0 16/32 位通用定时器模块 1 禁用。
0	R0	R/W	0	16/32 位通用定时器 0 运行模式时钟门控控制 值 描述 1 启用运行模式中的 16/32 位通用定时器模块 0 并提供时钟。 0 16/32 位通用定时器模块 0 禁用。

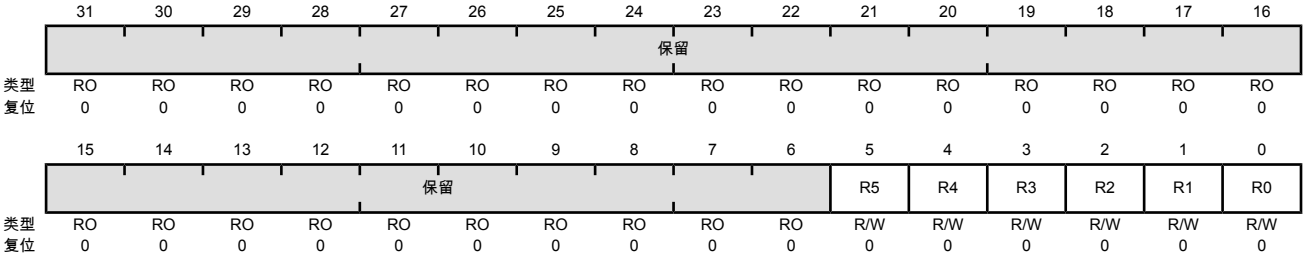
寄存器 57: 通用输入/输出运行模式时钟门控控制寄存器（RCGCGPIO），偏移量 0x608

RCGCGPIO 寄存器为软件提供启用和禁用运行模式中的 GPIO 模块的功能。在启用时，为模块提供一个时钟并允许对模块寄存器的访问。在禁用时，时钟禁用以节能，并且对模块寄存器的访问将产生总线错误。该寄存器特别为看门狗模块提供与传统运行模式时钟门控控制 n RCGCn 寄存器相同的功能，并且具有与相应 RCGCn 位相同的位极性。

重要: 应使用该寄存器控制 GPIO 模块的计时。要支持传统软件，可使用 RCGC2 寄存器。对 RCGC2 寄存器执行写操作将同时对该寄存器中的相应位执行写操作。通过对 RCGC2 寄存器的写操作更改的任何位都可以通过对 RCGC2 寄存器的读操作进行正确回读。软件必须使用该寄存器支持不处于传统寄存器中的模块。如果软件使用该寄存器写入传统外设（如 GPIO A），则写操作会产生正确操作，但是该位的值不在 RCGC2 寄存器中得到反映。如果软件使用传统和外设专用寄存器访问，则必须通过读-修改-写的操作来访问外设专用寄存器，因为该操作仅影响不在传统寄存器中的外设。通过这种方法，外设专用和传统寄存器都具有一致的信息。

通用输入/输出运行模式时钟门控控制寄存器 (RCGCGPIO)

基址 0x400F.E000
偏移量 0x608
类型 R/W, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:6	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
5	R5	R/W	0	GPIO 端口 F 运行模式时钟门控控制 值 描述 1 启用运行模式中的 GPIO 端口 F 并提供时钟。 0 GPIO 端口 F 禁用。
4	R4	R/W	0	GPIO 端口 E 运行模式时钟门控控制 值 描述 1 启用运行模式中的 GPIO 端口 E 并提供时钟。 0 GPIO 端口 E 禁用。
3	R3	R/W	0	GPIO 端口 D 运行模式时钟门控控制 值 描述 1 启用运行模式中的 GPIO 端口 D 并提供时钟。 0 GPIO 端口 D 禁用。

位/域	名称	类型	复位	描述
2	R2	R/W	0	GPIO 端口 C 运行模式时钟门控控制 值 描述 1 启用运行模式中的 GPIO 端口 C 并提供时钟。 0 GPIO 端口 C 禁用。
1	R1	R/W	0	GPIO 端口 B 运行模式时钟门控控制 值 描述 1 启用运行模式中的 GPIO 端口 B 并提供时钟。 0 GPIO 端口 B 禁用。
0	R0	R/W	0	GPIO 端口 A 运行模式时钟门控控制 值 描述 1 启用运行模式中的 GPIO 端口 A 并提供时钟。 0 GPIO 端口 A 禁用。

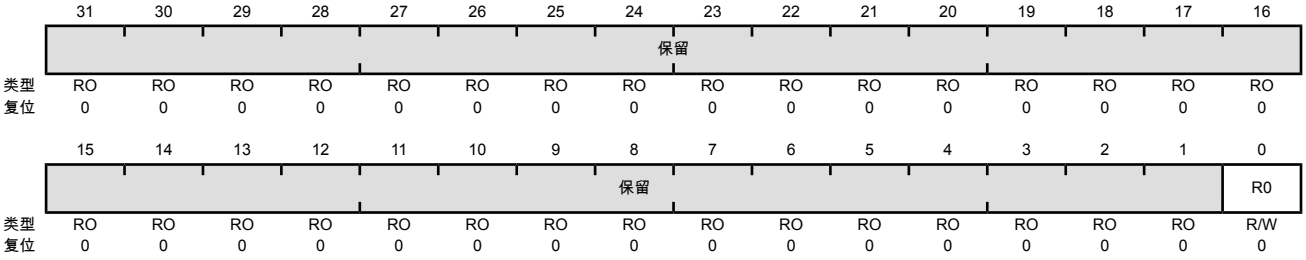
寄存器 58: 微型直接存储器访问运行模式时钟门控控制寄存器（RCGCDMA），偏移量 0x60C

RCGCDMA 寄存器为软件提供启用和禁用运行模式中的 µDMA 模块的功能。在启用时，为模块提供时钟并允许对模块寄存器的访问。在禁用时，时钟禁用以节能，并且对模块寄存器的访问将产生总线错误。该寄存器特别为看门狗模块提供与传统运行模式时钟门控控制 n RCGCn 寄存器相同的功能，并且具有与相应 RCGCn 位相同的位极性。

重要: 应使用该寄存器控制 µDMA 模块的计时。要支持传统软件，可使用 RCGC2 寄存器。对 RCGC2 寄存器中 UDMA 位执行写操作将同时对该寄存器中的 R0 位执行写操作。如果通过对 RCGC2 寄存器的写操作更改 UDMA 位，它可以通过读 RCGC2 寄存器进行正确回读。如果软件使用该寄存器控制 µDMA 模块，则该写操作会产生正确操作，但是 RCGC2 寄存器中的 UDMA 位不会反映 R0 位的值。如果软件使用传统和外设专用寄存器访问，则必须通过读-修改-写的操作来访问外设专用寄存器，因为该操作仅影响不在传统寄存器中的外设。通过这种方法，外设专用和传统寄存器都具有一致的信息。

微型直接存储器访问运行模式时钟门控控制寄存器 (RCGCDMA)

基址 0x400F.E000
偏移量 0x60C
类型 R/W, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:1	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
0	R0	R/W	0	µDMA 模块运行模式时钟门控控制 值 描述 1 启用运行模式中的 µDMA 模块 并提供时钟。 0 µDMA 模块禁用。

寄存器 59: 休眠运行模式时钟门控控制寄存器（RCGCHIB），偏移量 0x614

RCGCHIB 寄存器为软件提供启用和禁用运行模式中休眠模块的功能。在启用时，为模块提供时钟并允许对模块寄存器的访问。在禁用时，时钟禁用以节能，并且对模块寄存器的访问将产生总线错误。该寄存器特别为看门狗模块提供与传统运行模式时钟门控控制 n RCGCn 寄存器相同的功能，并且具有与相应 RCGCn 位相同的位极性。

重要: 应使用该寄存器控制休眠模块的计时。要支持传统软件，可使用 RCGC0 寄存器。对 RCGC0 寄存器中 HIB 位执行写操作将同时对该寄存器中的 R0 位执行写操作。如果通过对 RCGC0 寄存器的写操作更改 HIB 位，它可以通过读 RCGC0 寄存器进行正确回读。如果软件使用该寄存器控制休眠模块，则该写操作会产生正确操作，但是 RCGC0 寄存器中的 HIB 位不会反映 R0 位的值。如果软件使用传统和外设专用寄存器访问，则必须通过读-修改-写的操作来访问外设专用寄存器，因为该操作仅影响不在传统寄存器中的外设。通过这种方法，外设专用和传统寄存器都具有一致的信息。

休眠运行模式时钟门控控制寄存器 (RCGCHIB)

基址 0x400F.E000
偏移量 0x614
类型 R/W, 复位 0x0000.0001

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	保留															
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	保留															R0
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

位/域	名称	类型	复位	描述
31:1	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
0	R0	R/W	1	休眠模块运行模式时钟门控控制 值 描述 1 启用运行模式中的休眠模块并提供时钟。 0 休眠模块禁用。

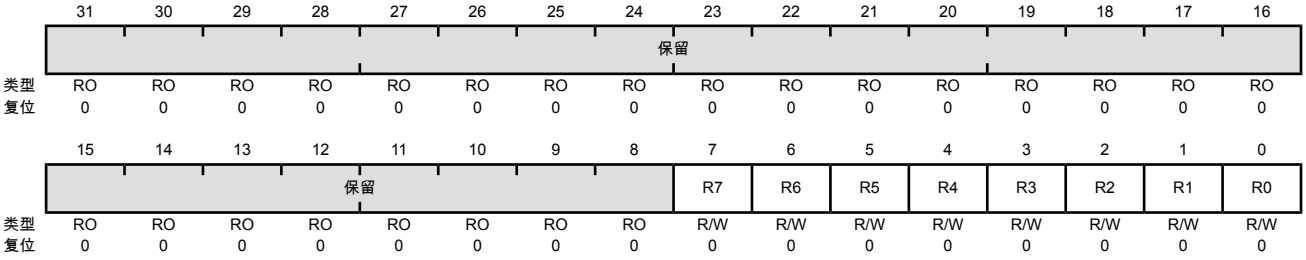
寄存器 60: 通用异步收发器运行模式时钟门控控制寄存器（RCGCUART），偏移量 0x618

RCGCUART 寄存器为软件提供启用和禁用运行模式中的 UART 模块的功能。在启用时，为模块提供一个时钟并允许对模块寄存器的访问。在禁用时，时钟禁用以节能，并且对模块寄存器的访问将产生总线错误。该寄存器特别为看门狗模块提供与传统运行模式时钟门控控制 n RCGCn 寄存器相同的功能，并且具有与相应 RCGCn 位相同的位极性。

重要: 应使用该寄存器控制 UART 模块的计时。要支持传统软件，可使用 RCGC1 寄存器。对 RCGC1 寄存器执行写操作将同时对该寄存器中的相应位执行写操作。通过对 RCGC1 寄存器的写操作更改的任何位都可以通过对 RCGC1 寄存器的读操作进行正确回读。软件必须使用该寄存器支持不处于传统寄存器中的模块。如果软件使用该寄存器对传统外设（如 UART0）进行写操作，则写操作会产生正确操作，但是该位的值不在 RCGC1 寄存器中得到反映。如果软件使用传统和外设专用寄存器访问，则必须通过读-修改-写的操作来访问外设专用寄存器，因为该操作仅影响不在传统寄存器中的外设。通过这种方法，外设专用和传统寄存器都具有一致的信息。

通用异步收发器运行模式时钟门控控制寄存器 (RCGCUART)

基址 0x400F.E000
偏移量 0x618
类型 R/W, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:8	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
7	R7	R/W	0	UART 模块 7 运行模式时钟门控控制 值 描述 1 启用运行模式中的 UART 模块 7 并提供时钟。 0 UART 模块 7 禁用。
6	R6	R/W	0	UART 模块 6 运行模式时钟门控控制 值 描述 1 启用运行模式中的 UART 模块 6 并提供时钟。 0 UART 模块 6 禁用。
5	R5	R/W	0	UART 模块 5 运行模式时钟门控控制 值 描述 1 启用运行模式中的 UART 模块 5 并提供时钟。 0 UART 模块 5 禁用。

位/域	名称	类型	复位	描述
4	R4	R/W	0	UART 模块 4 运行模式时钟门控控制 值 描述 1 启用运行模式中的 UART 模块 4 并提供时钟。 0 UART 模块 4 禁用。
3	R3	R/W	0	UART 模块 3 运行模式时钟门控控制 值 描述 1 启用运行模式中的 UART 模块 3 并提供时钟。 0 UART 模块 3 禁用。
2	R2	R/W	0	UART 模块 2 运行模式时钟门控控制 值 描述 1 启用运行模式中的 UART 模块 2 并提供时钟。 0 UART 模块 2 禁用。
1	R1	R/W	0	UART 模块 1 运行模式时钟门控控制 值 描述 1 启用运行模式中的 UART 模块 1 并提供时钟。 0 UART 模块 1 禁用。
0	R0	R/W	0	UART 模块 0 运行模式时钟门控控制 值 描述 1 启用运行模式中的 UART 模块 0 并提供时钟。 0 UART 模块 0 禁用。

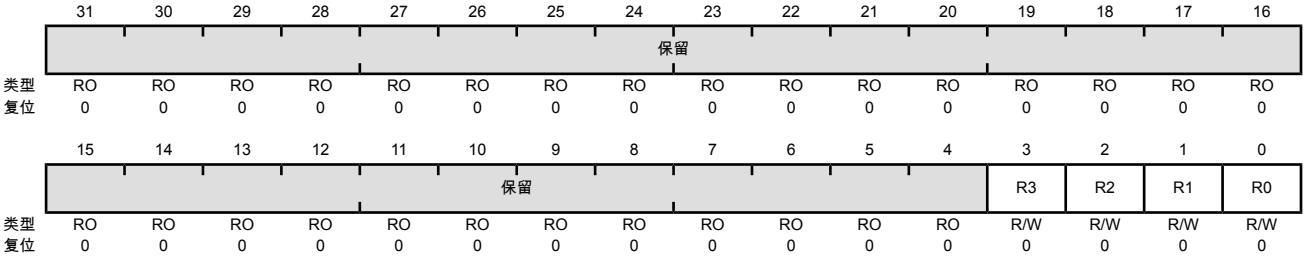
寄存器 61: 同步串行接口运行模式时钟门控控制寄存器（RCGCSSI），偏移量 0x61C

RCGCSSI 寄存器为软件提供启用和禁用运行模式中的 SSI 模块的功能。在启用时，为模块提供一个时钟并允许对模块寄存器的访问。在禁用时，时钟禁用以节能，并且对模块寄存器的访问将产生总线错误。该寄存器特别为看门狗模块提供与传统运行模式时钟门控控制 n RCGCn 寄存器相同的功能，并且具有与相应 RCGCn 位相同的位极性。

重要: 应使用该寄存器控制 SSI 模块的计时。要支持传统软件，可使用 RCGC1 寄存器。对 RCGC1 寄存器执行写操作将同时对该寄存器中的相应位执行写操作。通过对 RCGC1 寄存器的写操作更改的任何位都可以通过对 RCGC1 寄存器的读操作进行正确回读。软件必须使用该寄存器支持不处于传统寄存器中的模块。如果软件使用该寄存器对传统外设（如 SSI0）进行写操作，则写操作会产生正确操作，但是该位的值不在 RCGC1 寄存器中得到反映。如果软件使用传统和外设专用寄存器访问，则必须通过读-修改-写的操作来访问外设专用寄存器，因为该操作仅影响不在传统寄存器中的外设。通过这种方法，外设专用和传统寄存器都具有一致的信息。

同步串行接口运行模式时钟门控控制寄存器 (RCGCSSI)

基址 0x400F.E000
偏移量 0x61C
类型 R/W, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:4	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
3	R3	R/W	0	SSI 模块 3 运行模式时钟门控控制 值 描述 1 启用运行模式中的 SSI 模块 3 并提供时钟。 0 SSI 模块 3 禁用。
2	R2	R/W	0	SSI 模块 2 运行模式时钟门控控制 值 描述 1 启用运行模式中的 SSI 模块 2 并提供时钟。 0 SSI 模块 2 禁用。
1	R1	R/W	0	SSI 模块 1 运行模式时钟门控控制 值 描述 1 启用运行模式中的 SSI 模块 1 并提供时钟。 0 SSI 模块 1 禁用。

位/域	名称	类型	复位	描述
0	R0	R/W	0	SSI 模块 0 运行模式时钟门控控制 值 描述 1 启用运行模式中的 SSI 模块 0 并提供时钟。 0 SSI 模块 0 禁用。

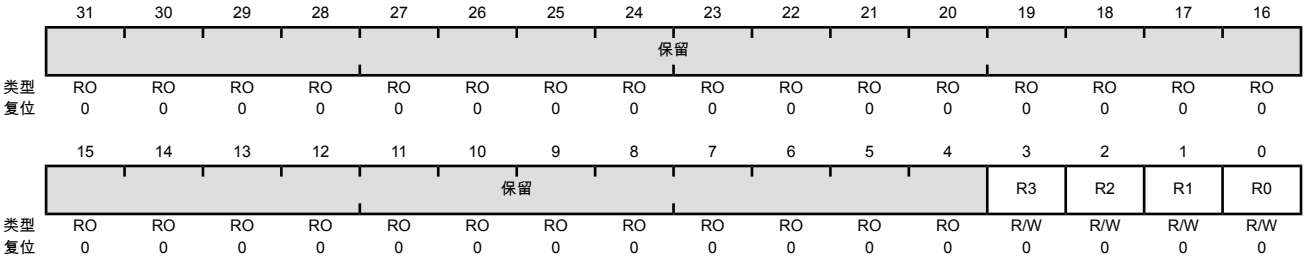
寄存器 62: 内部集成电路运行模式时钟门控控制寄存器（RCGCI2C），偏移量 0x620

RCGCI2C 寄存器为软件提供启用和禁用运行模式中的 I²C 模块的功能。在启用时，为模块提供一个时钟并允许对模块寄存器的访问。在禁用时，时钟禁用以节能，并且对模块寄存器的访问将产生总线错误。该寄存器特别为看门狗模块提供与传统运行模式时钟门控控制 n RCGCn 寄存器相同的功能，并且具有与相应 RCGCn 位相同的位极性。

重要: 应使用该寄存器控制 I²C 模块的计时。要支持传统软件，可使用 RCGC1 寄存器。对 RCGC1 寄存器执行写操作将同时对该寄存器中的相应位执行写操作。通过对 RCGC1 寄存器的写操作更改的任何位都可以通过对 RCGC1 寄存器的读操作进行正确回读。软件必须使用该寄存器支持不处于传统寄存器中的模块。如果软件使用该寄存器对传统外设（如 I2C0）进行写操作，则写操作会产生正确操作，但是该位的值不在 RCGC1 寄存器中得到反映。如果软件使用传统和外设专用寄存器访问，则必须通过读-修改-写的操作来访问外设专用寄存器，因为该操作仅影响不在传统寄存器中的外设。通过这种方法，外设专用和传统寄存器都具有一致的信息。

内部集成电路运行模式时钟门控控制寄存器 (RCGCI2C)

基址 0x400F.E000
偏移量 0x620
类型 R/W, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:4	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
3	R3	R/W	0	I²C 模块 3 运行模式时钟门控控制 值 描述 1 启用运行模式中的 I²C 模块 3 并提供时钟。 0 I²C 模块 3 禁用。
2	R2	R/W	0	I²C 模块 2 运行模式时钟门控控制 值 描述 1 启用运行模式中的 I²C 模块 2 并提供时钟。 0 I²C 模块 2 禁用。
1	R1	R/W	0	I²C 模块 1 运行模式时钟门控控制 值 描述 1 启用运行模式中的 I²C 模块 1 并提供时钟。 0 I²C 模块 1 禁用。

位/域	名称	类型	复位	描述
0	R0	R/W	0	I^2C 模块 0 运行模式时钟门控控制

值	描述
1	启用运行模式中的 I^2C 模块 0 并提供时钟。
0	I^2C 模块 0 禁用。

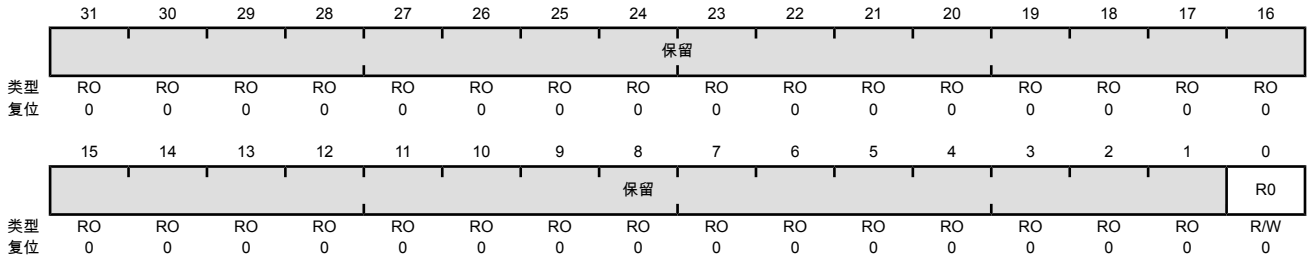
寄存器 63: 控制器局域网运行模式时钟门控控制寄存器（RCGCCAN），偏移量 0x634

RCGCCAN 寄存器为软件提供启用和禁用运行模式中的 CAN 模块的功能。在启用时，为模块提供一个时钟并允许对模块寄存器的访问。在禁用时，时钟禁用以节能，并且对模块寄存器的访问将产生总线错误。该寄存器特别为看门狗模块提供与传统运行模式时钟门控控制 n RCGCn 寄存器相同的功能，并且具有与相应 RCGCn 位相同的位极性。

重要: 应使用该寄存器控制 CAN 模块的计时。要支持传统软件，可使用 RCGC0 寄存器。对 RCGC0 寄存器执行写操作将同时对该寄存器中的相应位执行写操作。通过对 RCGC0 寄存器的写操作更改的任何位都可以通过对 RCGC0 寄存器的读操作进行正确回读。如果软件使用该寄存器对传统外设（如 CAN0）进行写操作，则写操作会产生正确操作，但是该位的值不在 RCGC0 寄存器中得到反映。如果软件使用传统和外设专用寄存器访问，则必须通过读-修改-写的操作来访问外设专用寄存器，因为该操作仅影响不在传统寄存器中的外设。通过这种方法，外设专用和传统寄存器都具有一致的信息。

控制器局域网运行模式时钟门控控制寄存器 (RCGCCAN)

基址 0x400F.E000
偏移量 0x634
类型 R/W, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:1	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
0	R0	R/W	0	CAN 模块 0 运行模式时钟门控控制 值 描述 1 启用运行模式中的 CAN 模块 0 并提供时钟。 0 CAN 模块 0 禁用。

寄存器 64: 模数转换器运行模式时钟门控控制寄存器（RCGCADC），偏移量 0x638

RCGCADC 寄存器为软件提供启用和禁用运行模式中的 ADC 模块的功能。在启用时，为模块提供一个时钟并允许对模块寄存器的访问。在禁用时，时钟禁用以节能，并且对模块寄存器的访问将产生总线错误。该寄存器特别为看门狗模块提供与传统运行模式时钟门控控制 n RCGCn 寄存器相同的功能，并且具有与相应 RCGCn 位相同的位极性。

重要: 应使用该寄存器控制 ADC 模块的计时。要支持传统软件，可使用 RCGC0 寄存器。对 RCGC0 寄存器执行写操作将同时对该寄存器中的相应位执行写操作。通过对 RCGC0 寄存器的写操作更改的任何位都可以通过对 RCGC0 寄存器的读操作进行正确回读。如果软件使用该寄存器对传统外设（如 ADC0）进行写操作，则写操作会产生正确操作，但是该位的值不在 RCGC0 寄存器中得到反映。如果软件使用传统和外设专用寄存器访问，则必须通过读-修改-写的操作来访问外设专用寄存器，因为该操作仅影响不在传统寄存器中的外设。通过这种方法，外设专用和传统寄存器都具有一致的信息。

模数转换器运行模式时钟门控控制寄存器 (RCGCADC)

基址 0x400F.E000
偏移量 0x638
类型 R/W, 复位 0x0000.0000

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
域	保留															
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
域	保留														R1	R0
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位/域	名称	类型	复位	描述
31:2	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
1	R1	R/W	0	ADC 模块 1 运行模式时钟门控控制 值 描述 1 启用运行模式中的 ADC 模块 1 并提供时钟。 0 ADC 模块 1 禁用。
0	R0	R/W	0	ADC 模块 0 运行模式时钟门控控制 值 描述 1 启用运行模式中的 ADC 模块 0 并提供时钟。 0 ADC 模块 0 禁用。

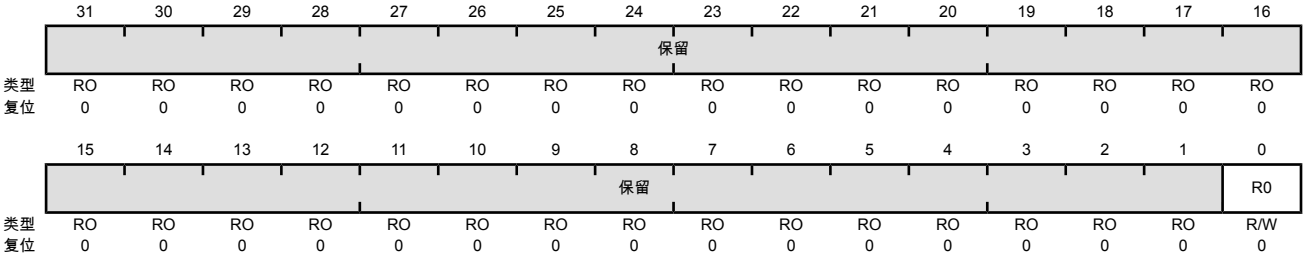
寄存器 65: 模拟比较器运行模式时钟门控控制寄存器（RCGCACMP），偏移量 0x63C

RCGCACMP 寄存器为软件提供启用和禁用运行模式中模拟比较器模块的功能。在启用时，为模块提供时钟并允许对模块寄存器的访问。在禁用时，时钟禁用以节能，并且对模块寄存器的访问将产生总线错误。该寄存器特别为看门狗模块提供与传统运行模式时钟门控控制 n RCGCn 寄存器相同的功能，并且具有与相应 RCGCn 位相同的位极性。

重要: 应使用该寄存器控制模拟比较器模块的计时。要支持传统软件，可使用 RCGC1 寄存器。对 RCGC1 寄存器中 COMPn 位执行写操作将同时对该寄存器中的 R0 位执行写操作。如果通过对 RCGC1 寄存器的写操作置位任何 COMPn 位，它可以在对 RCGC1 寄存器进行读操作时进行正确回读。如果软件使用该寄存器更改模拟比较器模块的计时，则写操作会产生正确操作，但是 RCGC1 寄存器的 COMPn 位不反映值 R0。如果软件使用传统和外设专用寄存器访问，则必须通过读-修改-写的操作来访问外设专用寄存器，因为该操作仅影响不在传统寄存器中的外设。通过这种方法，外设专用和传统寄存器都具有一致的信息。

模拟比较器运行模式时钟门控控制寄存器 (RCGCACMP)

基址 0x400F.E000
偏移量 0x63C
类型 R/W, 复位 0x0000.0000



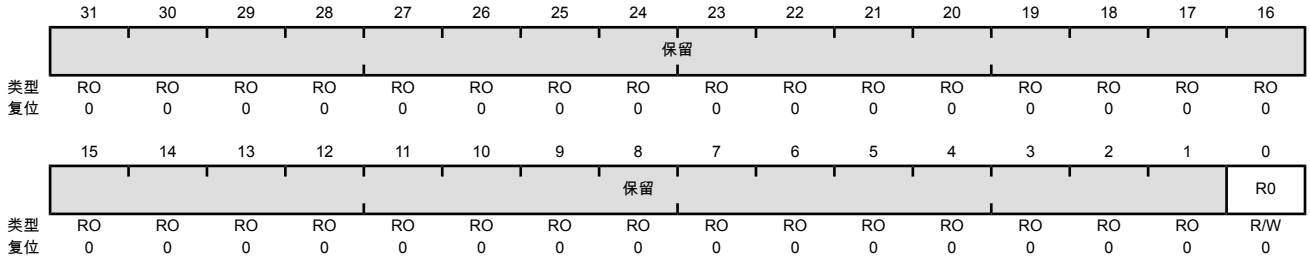
位/域	名称	类型	复位	描述
31:1	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
0	R0	R/W	0	模拟比较器模块 0 运行模式时钟门控控制 值 描述 1 启用运行模式中的模拟比较器模块并提供时钟。 0 模拟比较器模块禁用。

寄存器 66: EEPROM 运行模式时钟门控控制寄存器（RCGCEEPROM），偏移量 0x658

RCGCEEPROM 寄存器为软件提供启用和禁用运行模式中的 EEPROM 模块的功能。在启用时，为模块提供时钟并允许对模块寄存器的访问。在禁用时，时钟禁用以节能，并且对模块寄存器的访问将产生总线错误。

EEPROM 运行模式时钟门控控制寄存器 (RCGCEEPROM)

基址 0x400F.E000
偏移量 0x658
类型 R/W, 复位 0x0000.0000



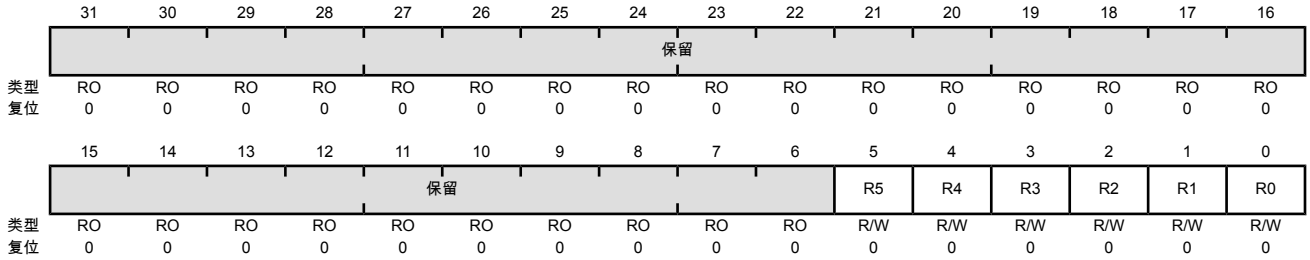
位/域	名称	类型	复位	描述
31:1	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
0	R0	R/W	0	EEPROM 模块运行模式时钟门控控制 值 描述 1 启用运行模式中的 EEPROM 模块并提供时钟。 0 EEPROM 模块禁用。

寄存器 67:32/64 位宽通用定时器运行模式时钟门控控制寄存器（RCGCWTIMER），偏移量 0x65C

RCGCWTIMER 寄存器为软件提供启用和禁用运行模式中的 32/64 位定时器模块的功能。在启用时，为模块提供一个时钟并允许对模块寄存器的访问。在禁用时，时钟禁用以节能，并且对模块寄存器的访问将产生总线错误。该寄存器特别为定时器模块提供与传统运行模式时钟门控控制 n RCGCn 寄存器相同的功能，并且具有与相应 RCGCn 位相同的位极性。

32/64 位宽通用定时器运行模式时钟门控控制寄存器 (RCGCWTIMER)

基址 0x400F.E000
偏移量 0x65C
类型 R/W, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:6	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
5	R5	R/W	0	32/64 位宽通用定时器 5 运行模式时钟门控控制 值 描述 1 启用运行模式中的 32/64 位宽通用定时器模块 5 并提供时钟。 0 32/64 位宽通用定时器模块 5 禁用。
4	R4	R/W	0	32/64 位宽通用定时器 4 运行模式时钟门控控制 值 描述 1 启用运行模式中的 32/64 位宽通用定时器模块 4 并提供时钟。 0 32/64 位宽通用定时器模块 4 禁用。
3	R3	R/W	0	32/64 位宽通用定时器 3 运行模式时钟门控控制 值 描述 1 启用运行模式中的 32/64 位宽通用定时器模块 3 并提供时钟。 0 32/64 位宽通用定时器模块 3 禁用。
2	R2	R/W	0	32/64 位宽通用定时器 2 运行模式时钟门控控制 值 描述 1 启用运行模式中的 32/64 位宽通用定时器模块 2 并提供时钟。 0 32/64 位宽通用定时器模块 2 禁用。

位/域	名称	类型	复位	描述
1	R1	R/W	0	32/64 位宽通用定时器 1 运行模式时钟门控控制 值 描述 1 启用运行模式中的 32/64 位宽通用定时器模块 1 并提供时钟。 0 32/64 位宽通用定时器模块 1 禁用。
0	R0	R/W	0	32/64 位宽通用定时器 0 运行模式时钟门控控制 值 描述 1 启用运行模式中的 32/64 位宽通用定时器模块 0 并提供时钟。 0 32/64 位宽通用定时器模块 0 禁用。

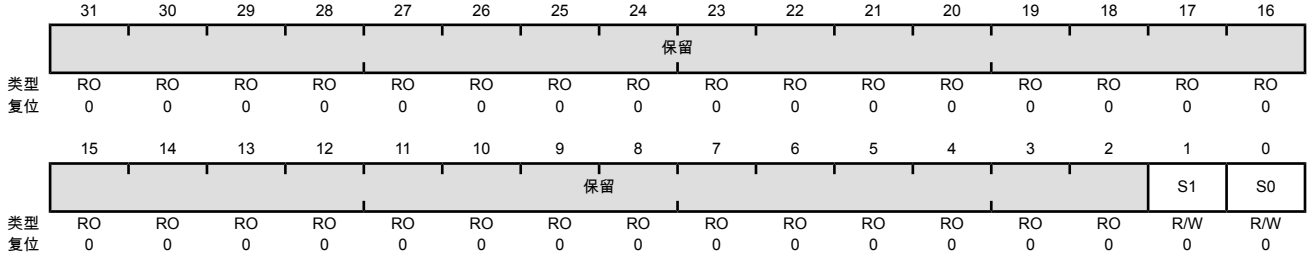
寄存器 68: 看门狗定时器睡眠模式时钟门控控制寄存器（SCGCWD），偏移量 0x700

SCGCWD 寄存器为软件提供启用和禁用睡眠模式中的看门狗模块的功能。在启用时，为模块提供一个时钟。在禁用时，禁用时钟以节能。该寄存器特别为看门狗模块提供与传统睡眠模式时钟门控控制 n SCGCn 寄存器相同的功能，并且具有与相应 SCGCn 位相同的位极性。

重要: 应使用该寄存器控制看门狗模块的计时。要支持传统软件，可使用 SCGC0 寄存器。对 SCGC0 寄存器执行写操作将同时对该寄存器中的相应位执行写操作。通过对 SCGC0 寄存器的写操作更改的任何位都可以通过对 SCGC0 寄存器的读操作进行正确回读。如果软件使用该寄存器对传统外设（如 Watchdog 0）进行写操作，则写操作会产生正确操作，但是该位的值不在 SCGC0 寄存器中得到反映。如果软件使用传统和外设专用寄存器访问，则必须通过读-修改-写的操作来访问外设专用寄存器，因为该操作仅影响不在传统寄存器中的外设。通过这种方法，外设专用和传统寄存器都具有一致的信息。

看门狗定时器睡眠模式时钟门控控制寄存器 (SCGCWD)

基址 0x400F.E000
偏移量 0x700
类型 R/W, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:2	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
1	S1	R/W	0	看门狗定时器 1 睡眠模式时钟门控控制 值 描述 1 启用睡眠模式中的看门狗模块 1 并提供时钟。 0 看门狗模块 1 禁用。
0	S0	R/W	0	看门狗定时器 0 睡眠模式时钟门控控制 值 描述 1 启用睡眠模式中的看门狗模块 0 并提供时钟。 0 看门狗模块 0 禁用。

寄存器 69: 16/32 位通用定时器睡眠模式时钟门控控制寄存器（SCGCTIMER），偏移量 0x704

SCGCTIMER 寄存器为软件提供启用和禁用睡眠模式中的 16/32 位定时器模块的功能。在启用时，为模块提供一个时钟。在禁用时，禁用时钟以节能。该寄存器特别为定时器模块提供与传统睡眠模式时钟门控控制 n SCGCn 寄存器相同的功能，并且具有与相应 SCGCn 位相同的位极性。

重要: 应使用该寄存器控制定时器模块的计时。要支持传统软件，可使用 SCGC1 寄存器。对 SCGC1 寄存器执行写操作将同时对该寄存器中的相应位执行写操作。通过对 SCGC1 寄存器的写操作更改的任何位都可以通过对 SCGC1 寄存器的读操作进行正确回读。软件必须使用该寄存器支持不处于传统寄存器中的模块。如果软件使用该寄存器对传统外设（如 Timer 0）进行写操作，则写操作会产生正确操作，但是该位的值不在 SCGC1 寄存器中得到反映。如果软件使用传统和外设专用寄存器访问，则必须通过读-修改-写的操作来访问外设专用寄存器，因为该操作仅影响不在传统寄存器中的外设。通过这种方法，外设专用和传统寄存器都具有一致的信息。

16/32 位通用定时器睡眠模式时钟门控控制寄存器 (SCGCTIMER)

基址 0x400F.E000
偏移量 0x704
类型 R/W, 复位 0x0000.0000

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
域	保留															
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
域	保留										S5	S4	S3	S2	S1	S0
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W	R/W	R/W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位/域	名称	类型	复位	描述
31:6	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
5	S5	R/W	0	16/32 位通用定时器 5 睡眠模式时钟门控控制 值 描述 1 启用睡眠模式中的 16/32 位通用定时器模块 5 并提供时钟。 0 16/32 位通用定时器模块 5 禁用。
4	S4	R/W	0	16/32 位通用定时器 4 睡眠模式时钟门控控制 值 描述 1 启用睡眠模式中的 16/32 位通用定时器模块 4 并提供时钟。 0 16/32 位通用定时器模块 4 禁用。
3	S3	R/W	0	16/32 位通用定时器 3 睡眠模式时钟门控控制 值 描述 1 启用睡眠模式中的 16/32 位通用定时器模块 3 并提供时钟。 0 16/32 位通用定时器模块 3 禁用。

位/域	名称	类型	复位	描述
2	S2	R/W	0	16/32 位通用定时器 2 睡眠模式时钟门控控制 值 描述 1 启用睡眠模式中的 16/32 位通用定时器模块 2 并提供时钟。 0 16/32 位通用定时器模块 2 禁用。
1	S1	R/W	0	16/32 位通用定时器 1 睡眠模式时钟门控控制 值 描述 1 启用睡眠模式中的 16/32 位通用定时器模块 1 并提供时钟。 0 16/32 位通用定时器模块 1 禁用。
0	S0	R/W	0	16/32 位通用定时器 0 睡眠模式时钟门控控制 值 描述 1 启用睡眠模式中的 16/32 位通用定时器模块 0 并提供时钟。 0 16/32 位通用定时器模块 0 禁用。

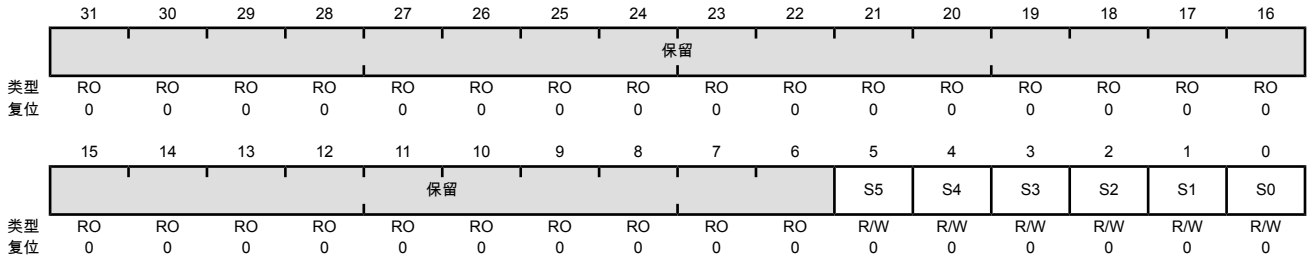
寄存器 70: 通用输入/输出睡眠模式时钟门控控制寄存器（SCGCGPIO），偏移量 0x708

SCGCGPIO 寄存器为软件提供启用和禁用睡眠模式中的 GPIO 模块的功能。在启用时，为模块提供一个时钟。在禁用时，禁用时钟以节能。该寄存器特别为看门狗模块提供与传统睡眠模式时钟门控控制 n SCGCn 寄存器相同的功能，并且具有与相应 SCGCn 位相同的位极性。

重要: 应使用该寄存器控制 GPIO 模块的计时。要支持传统软件，可使用 SCGC2 寄存器。对 SCGC2 寄存器执行写操作将同时对该寄存器中的相应位执行写操作。通过对 SCGC2 寄存器的写操作更改的任何位都可以通过对 SCGC2 寄存器的读操作进行正确回读。软件必须使用该寄存器支持不处于传统寄存器中的模块。如果软件使用该寄存器写入传统外设（如 GPIO A），则写操作会产生正确操作，但是该位的值不在 SCGC2 寄存器中得到反映。如果软件使用传统和外设专用寄存器访问，则必须通过读-修改-写的操作来访问外设专用寄存器，因为该操作仅影响不在传统寄存器中的外设。通过这种方法，外设专用和传统寄存器都具有一致的信息。

通用输入/输出睡眠模式时钟门控控制寄存器 (SCGCGPIO)

基址 0x400F.E000
偏移量 0x708
类型 R/W, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:6	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
5	S5	R/W	0	GPIO 端口 F 睡眠模式时钟门控控制 值 描述 1 启用睡眠模式中的 GPIO 端口 F 并提供时钟。 0 GPIO 端口 F 禁用。
4	S4	R/W	0	GPIO 端口 E 睡眠模式时钟门控控制 值 描述 1 启用睡眠模式中的 GPIO 端口 E 并提供时钟。 0 GPIO 端口 E 禁用。
3	S3	R/W	0	GPIO 端口 D 睡眠模式时钟门控控制 值 描述 1 启用睡眠模式中的 GPIO 端口 D 并提供时钟。 0 GPIO 端口 D 禁用。

位/域	名称	类型	复位	描述
2	S2	R/W	0	GPIO 端口 C 睡眠模式时钟门控控制 值 描述 1 启用睡眠模式中的 GPIO 端口 C 并提供时钟。 0 GPIO 端口 C 禁用。
1	S1	R/W	0	GPIO 端口 B 睡眠模式时钟门控控制 值 描述 1 启用睡眠模式中的 GPIO 端口 B 并提供时钟。 0 GPIO 端口 B 禁用。
0	S0	R/W	0	GPIO 端口 A 睡眠模式时钟门控控制 值 描述 1 启用睡眠模式中的 GPIO 端口 A 并提供时钟。 0 GPIO 端口 A 禁用。

寄存器 71: 微型直接存储器访问睡眠模式时钟门控控制寄存器（SCGCDMA），偏移量 0x70C

SCGCDMA 寄存器为软件提供启用和禁用睡眠模式中的 µDMA 模块的功能。在启用时，为模块提供一个时钟。在禁用时，禁用时钟以节能。该寄存器特别为看门狗模块提供与传统睡眠模式时钟门控控制 n SCGCn 寄存器相同的功能，并且具有与相应 SCGCn 位相同的位极性。

重要: 应使用该寄存器控制 µDMA 模块的计时。要支持传统软件，可使用 SCGC2 寄存器。对 SCGC2 寄存器中 UDMA 位的写操作还对该寄存器中的 S0 位执行写操作。如果通过对 SCGC2 寄存器的写操作更改 UDMA 位，它可以通过读 SCGC2 寄存器进行正确回读。如果软件使用该寄存器控制µDMA模块，则该写操作会产生正确操作，但是SCGC2寄存器中的UDMA位不会反映S0位的值。如果软件使用传统和外设专用寄存器访问，则必须通过读-修改-写的操作来访问外设专用寄存器，因为该操作仅影响不在传统寄存器中的外设。通过这种方法，外设专用和传统寄存器都具有一致的信息。

微型直接存储器访问睡眠模式时钟门控控制寄存器 (SCGCDMA)

基址 0x400F.E000
偏移量 0x70C
类型 R/W, 复位 0x0000.0000

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	保留															
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	保留															S0
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位/域	名称	类型	复位	描述
31:1	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
0	S0	R/W	0	µDMA 模块睡眠模式时钟门控控制 值 描述 1 启用睡眠模式中的 µDMA 模块并提供时钟。 0 µDMA 模块禁用。

寄存器 72: 休眠睡眠模式时钟门控控制寄存器（SCGCHIB），偏移量 0x714

SCGCHIB 寄存器为软件提供启用和禁用睡眠模式中休眠模块的功能。在启用时，为模块提供一个时钟。在禁用时，禁用时钟以节能。该寄存器特别为看门狗模块提供与传统睡眠模式时钟门控控制 n SCGCn 寄存器相同的功能，并且具有与相应 SCGCn 位相同的位极性。

重要: 应使用该寄存器控制休眠模块的计时。要支持传统软件，可使用 SCGC0 寄存器。对 SCGC0 寄存器中 HIB 位的写操作还对该寄存器中的 S0 位执行写操作。如果通过对 SCGC0 寄存器的写操作更改 HIB 位，它可以通过读 SCGC0 寄存器进行正确回读。如果软件使用该寄存器控制休眠模块，则该写操作会产生正确操作，但是 SCGC0 寄存器中的 HIB 位不会反映 S0 位的值。如果软件使用传统和外设专用寄存器访问，则必须通过读-修改-写的操作来访问外设专用寄存器，因为该操作仅影响不在传统寄存器中的外设。通过这种方法，外设专用和传统寄存器都具有一致的信息。

休眠睡眠模式时钟门控控制寄存器 (SCGCHIB)

基址 0x400F.E000
偏移量 0x714
类型 R/W, 复位 0x0000.0001

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	保留															
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	保留															S0
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

位/域	名称	类型	复位	描述
31:1	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
0	S0	R/W	1	休眠模块睡眠模式时钟门控控制 值 描述 1 启用睡眠模式中的休眠模块并提供时钟。 0 休眠模块禁用。

寄存器 73: 通用异步收发器睡眠模式时钟门控控制寄存器（SCGCUART），偏移量 0x718

SCGCUART 寄存器为软件提供启用和禁用睡眠模式中的 UART 模块的功能。在启用时，为模块提供一个时钟。在禁用时，禁用时钟以节能。该寄存器特别为看门狗模块提供与传统睡眠模式时钟门控控制 n SCGCn 寄存器相同的功能，并且具有与相应 SCGCn 位相同的位极性。

重要: 应使用该寄存器控制UART模块的计时。要支持传统软件，可使用SCGC1寄存器。对SCGC1 寄存器执行写操作将同时对该寄存器中的相应位执行写操作。通过对 SCGC1 寄存器的写操作更改的任何位都可以通过对 SCGC1 寄存器的读操作进行正确回读。软件必须使用该寄存器支持不处于传统寄存器中的模块。如果软件使用该寄存器对传统外设（如 UART0）进行写操作，则写操作会产生正确操作，但是该位的值不在 SCGC1 寄存器中得到反映。如果软件使用传统和外设专用寄存器访问，则必须通过读-修改-写的操作来访问外设专用寄存器，因为该操作仅影响不在传统寄存器中的外设。通过这种方法，外设专用和传统寄存器都具有一致的信息。

通用异步收发器睡眠模式时钟门控控制寄存器 (SCGCUART)

基址 0x400F.E000
偏移量 0x718
类型 R/W, 复位 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	保留															
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	保留								S7	S6	S5	S4	S3	S2	S1	S0
类型	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位/域	名称	类型	复位	描述
31:8	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
7	S7	R/W	0	UART 模块 7 睡眠模式时钟门控控制 值 描述 1 启用睡眠模式中的 UART 模块 7 并提供时钟。 0 UART 模块 7 禁用。
6	S6	R/W	0	UART 模块 6 睡眠模式时钟门控控制 值 描述 1 启用睡眠模式中的 UART 模块 6 并提供时钟。 0 UART 模块 6 禁用。
5	S5	R/W	0	UART 模块 5 睡眠模式时钟门控控制 值 描述 1 启用睡眠模式中的 UART 模块 5 并提供时钟。 0 UART 模块 5 禁用。

位/域	名称	类型	复位	描述
4	S4	R/W	0	UART 模块 4 睡眠模式时钟门控控制 值 描述 1 启用睡眠模式中的 UART 模块 4 并提供时钟。 0 UART 模块 4 禁用。
3	S3	R/W	0	UART 模块 3 睡眠模式时钟门控控制 值 描述 1 启用睡眠模式中的 UART 模块 3 并提供时钟。 0 UART 模块 3 禁用。
2	S2	R/W	0	UART 模块 2 睡眠模式时钟门控控制 值 描述 1 启用睡眠模式中的 UART 模块 2 并提供时钟。 0 UART 模块 2 禁用。
1	S1	R/W	0	UART 模块 1 睡眠模式时钟门控控制 值 描述 1 启用睡眠模式中的 UART 模块 1 并提供时钟。 0 UART 模块 1 禁用。
0	S0	R/W	0	UART 模块 0 睡眠模式时钟门控控制 值 描述 1 启用睡眠模式中的 UART 模块 0 并提供时钟。 0 UART 模块 0 禁用。

寄存器 74: 同步串行接口睡眠模式时钟门控控制寄存器（SCGCSSI），偏移量 0x71C

SCGCSSI 寄存器为软件提供启用和禁用睡眠模式中的 SSI 模块的功能。在启用时，为模块提供一个时钟。在禁用时，禁用时钟以节能。该寄存器特别为看门狗模块提供与传统睡眠模式时钟门控控制 n SCGCn 寄存器相同的功能，并且具有与相应 SCGCn 位相同的位极性。

重要: 应使用该寄存器控制 SSI 模块的计时。要支持传统软件，可使用 SCGC1 寄存器。对 SCGC1 寄存器执行写操作将同时对该寄存器中的相应位执行写操作。通过对 SCGC1 寄存器的写操作更改的任何位都可以通过对 SCGC1 寄存器的读操作进行正确回读。软件必须使用该寄存器支持不处于传统寄存器中的模块。如果软件使用该寄存器对传统外设（如 SSI0）进行写操作，则写操作会产生正确操作，但是该位的值不在 SCGC1 寄存器中得到反映。如果软件使用传统和外设专用寄存器访问，则必须通过读-修改-写的操作来访问外设专用寄存器，因为该操作仅影响不在传统寄存器中的外设。通过这种方法，外设专用和传统寄存器都具有一致的信息。

同步串行接口睡眠模式时钟门控控制寄存器 (SCGCSSI)

基址 0x400F.E000
偏移量 0x71C
类型 R/W, 复位 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	保留															
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	保留												S3	S2	S1	S0
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位/域	名称	类型	复位	描述
31:4	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
3	S3	R/W	0	SSI 模块 3 睡眠模式时钟门控控制 值 描述 1 启用睡眠模式中的 SSI 模块 3 并提供时钟。 0 SSI 模块 3 禁用。
2	S2	R/W	0	SSI 模块 2 睡眠模式时钟门控控制 值 描述 1 启用睡眠模式中的 SSI 模块 2 并提供时钟。 0 SSI 模块 2 禁用。
1	S1	R/W	0	SSI 模块 1 睡眠模式时钟门控控制 值 描述 1 启用睡眠模式中的 SSI 模块 1 并提供时钟。 0 SSI 模块 1 禁用。

位/域	名称	类型	复位	描述
0	S0	R/W	0	SSI 模块 0 睡眠模式时钟门控控制 值 描述 1 启用睡眠模式中的 SSI 模块 0 并提供时钟。 0 SSI 模块 0 禁用。

寄存器 75: 内部集成电路睡眠模式时钟门控控制寄存器（SCGCI2C），偏移量 0x720

SCGCI2C 寄存器为软件提供启用和禁用睡眠模式中的 I²C 模块的功能。在启用时，为模块提供一个时钟。在禁用时，禁用时钟以节能。该寄存器特别为看门狗模块提供与传统睡眠模式时钟门控控制 n SCGCn 寄存器相同的功能，并且具有与相应 SCGCn 位相同的位极性。

重要: 应使用该寄存器控制 I²C 模块的计时。要支持传统软件，可使用 SCGC1 寄存器。对 SCGC1 寄存器执行写操作将同时对该寄存器中的相应位执行写操作。通过对 SCGC1 寄存器的写操作更改的任何位都可以通过对 SCGC1 寄存器的读操作进行正确回读。软件必须使用该寄存器支持不处于传统寄存器中的模块。如果软件使用该寄存器对传统外设（如 I²C0）进行写操作，则写操作会产生正确操作，但是该位的值不在 SCGC1 寄存器中得到反映。如果软件使用传统和外设专用寄存器访问，则必须通过读-修改-写的操作来访问外设专用寄存器，因为该操作仅影响不在传统寄存器中的外设。通过这种方法，外设专用和传统寄存器都具有一致的信息。

内部集成电路睡眠模式时钟门控控制寄存器 (SCGCI2C)

基址 0x400F.E000
偏移量 0x720
类型 R/W, 复位 0x0000.0000

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
域	保留															
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
域	保留												S3	S2	S1	S0
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位/域	名称	类型	复位	描述
31:4	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
3	S3	R/W	0	I²C 模块 3 睡眠模式时钟门控控制 值 描述 1 启用睡眠模式中的 I²C 模块 3 并提供时钟。 0 I²C 模块 3 禁用。
2	S2	R/W	0	I²C 模块 2 睡眠模式时钟门控控制 值 描述 1 启用睡眠模式中的 I²C 模块 2 并提供时钟。 0 I²C 模块 2 禁用。
1	S1	R/W	0	I²C 模块 1 睡眠模式时钟门控控制 值 描述 1 启用睡眠模式中的 I²C 模块 1 并提供时钟。 0 I²C 模块 1 禁用。

位/域	名称	类型	复位	描述
0	S0	R/W	0	I^2C 模块 0 睡眠模式时钟门控控制 值 描述 1 启用睡眠模式中的 I^2C 模块 0 并提供时钟。 0 I^2C 模块 0 禁用。

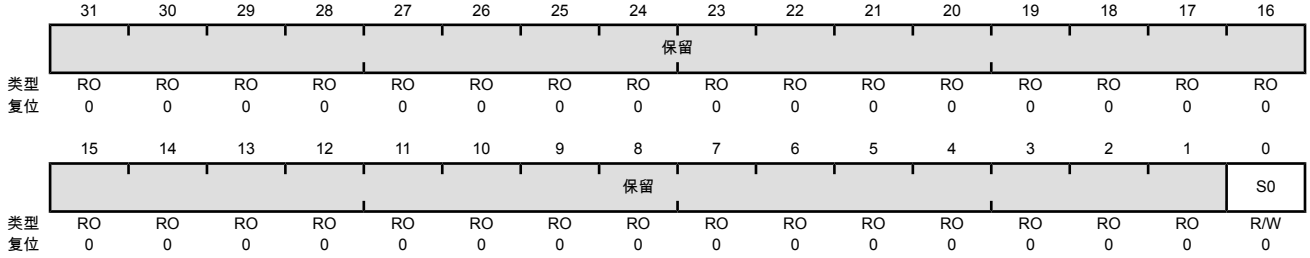
寄存器 76: 控制器局域网睡眠模式时钟门控控制寄存器（SCGCCAN），偏移量 0x734

SCGCCAN 寄存器为软件提供启用和禁用睡眠模式中的 CAN 模块的功能。在启用时，为模块提供一个时钟。在禁用时，禁用时钟以节能。该寄存器特别为看门狗模块提供与传统睡眠模式时钟门控控制 n SCGCn 寄存器相同的功能，并且具有与相应 SCGCn 位相同的位极性。

重要: 应使用该寄存器控制 CAN 模块的计时。要支持传统软件，可使用 SCGC0 寄存器。对 SCGC0 寄存器执行写操作将同时对该寄存器中的相应位执行写操作。通过对 SCGC0 寄存器的写操作更改的任何位都可以通过对 SCGC0 寄存器的读操作进行正确回读。如果软件使用该寄存器对传统外设（如 CAN0）进行写操作，则写操作会产生正确操作，但是该位的值不在 SCGC0 寄存器中得到反映。如果软件使用传统和外设专用寄存器访问，则必须通过读-修改-写的操作来访问外设专用寄存器，因为该操作仅影响不在传统寄存器中的外设。通过这种方法，外设专用和传统寄存器都具有一致的信息。

控制器局域网睡眠模式时钟门控控制寄存器 (SCGCCAN)

基址 0x400F.E000
偏移量 0x734
类型 R/W, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:1	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
0	S0	R/W	0	CAN 模块 0 睡眠模式时钟门控控制 值 描述 1 启用睡眠模式中的 CAN 模块 0 并提供时钟。 0 CAN 模块 0 禁用。

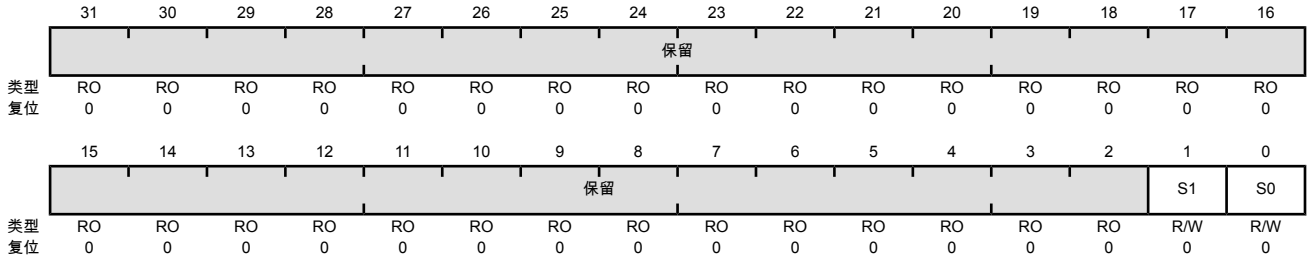
寄存器 77: 模数转换器睡眠模式时钟门控控制寄存器（SCGCADC），偏移量 0x738

SCGCADC 寄存器为软件提供启用和禁用睡眠模式中的 ADC 模块的功能。在启用时，为模块提供一个时钟。在禁用时，禁用时钟以节能。该寄存器特别为看门狗模块提供与传统睡眠模式时钟门控控制 n SCGCn 寄存器相同的功能，并且具有与相应 SCGCn 位相同的位极性。

重要: 应使用该寄存器控制 ADC 模块的计时。要支持传统软件，可使用 SCGC0 寄存器。对 SCGC0 寄存器执行写操作将同时对该寄存器中的相应位执行写操作。通过对 SCGC0 寄存器的写操作更改的任何位都可以通过对 SCGC0 寄存器的读操作进行正确回读。如果软件使用该寄存器对传统外设（如 ADC0）进行写操作，则写操作会产生正确操作，但是该位的值不在 SCGC0 寄存器中得到反映。如果软件使用传统和外设专用寄存器访问，则必须通过读-修改-写的操作来访问外设专用寄存器，因为该操作仅影响不在传统寄存器中的外设。通过这种方法，外设专用和传统寄存器都具有一致的信息。

模数转换器睡眠模式时钟门控控制寄存器 (SCGCADC)

基址 0x400F.E000
偏移量 0x738
类型 R/W, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:2	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
1	S1	R/W	0	ADC 模块 1 睡眠模式时钟门控控制 值 描述 1 启用睡眠模式中的 ADC 模块 1 并提供时钟。 0 ADC 模块 1 禁用。
0	S0	R/W	0	ADC 模块 0 睡眠模式时钟门控控制 值 描述 1 启用睡眠模式中的 ADC 模块 0 并提供时钟。 0 ADC 模块 0 禁用。

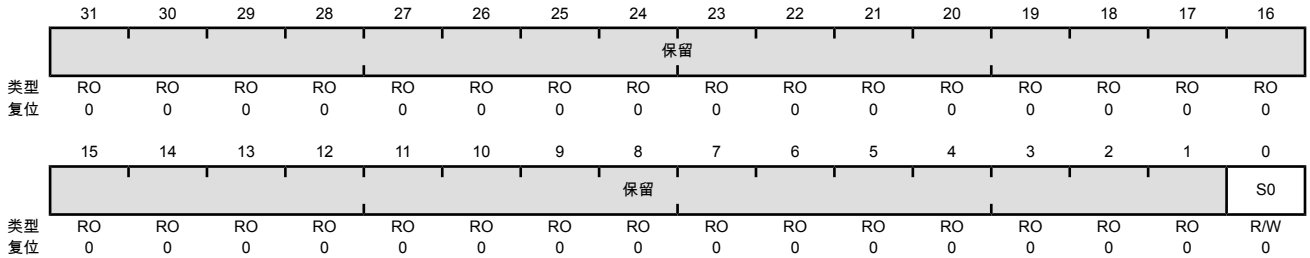
寄存器 78: 模拟比较器睡眠模式时钟门控控制寄存器（SCGCACMP），偏移量 0x73C

SCGCACMP 寄存器为软件提供启用和禁用睡眠模式中模拟比较器模块的功能。在启用时，为模块提供一个时钟。在禁用时，禁用时钟以节能。该寄存器特别为看门狗模块提供与传统睡眠模式时钟门控控制 n SCGCn 寄存器相同的功能，并且具有与相应 SCGCn 位相同的位极性。

重要: 应使用该寄存器控制模拟比较器模块的计时。要支持传统软件，可使用 SCGC1 寄存器。对 SCGC1 寄存器中 COMPn 位的写操作还对该寄存器中的 S0 位执行写操作。如果通过对 SCGC1 寄存器的写操作置位任何 COMPn 位，它可以在对 SCGC1 寄存器进行读操作时进行正确回读。如果软件使用该寄存器更改模拟比较器模块的计时，则写操作会产生正确操作，但是 SCGC1 寄存器的 COMPn 位不反映值 S0。如果软件使用传统和外设专用寄存器访问，则必须通过读-修改-写的操作来访问外设专用寄存器，因为该操作仅影响不在传统寄存器中的外设。通过这种方法，外设专用和传统寄存器都具有一致的信息。

模拟比较器睡眠模式时钟门控控制寄存器 (SCGCACMP)

基址 0x400F.E000
偏移量 0x73C
类型 R/W, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:1	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
0	S0	R/W	0	模拟比较器模块 0 睡眠模式时钟门控控制 值 描述 1 启用睡眠模式中的模拟比较器模块并提供时钟。 0 模拟比较器模块禁用。

寄存器 79: EEPROM 睡眠模式时钟门控控制寄存器（SCGCEEPROM），偏移量 0x758

SCGCEEPROM 寄存器为软件提供启用和禁用睡眠模式中的 EEPROM 模块的功能。在启用时，为模块提供一个时钟。在禁用时，禁用时钟以节能。

EEPROM 睡眠模式时钟门控控制寄存器 (SCGCEEPROM)

基址 0x400F.E000
偏移量 0x758
类型 R/W, 复位 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	保留															
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	保留															S0
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位/域	名称	类型	复位	描述
31:1	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
0	S0	R/W	0	EEPROM 模块睡眠模式时钟门控控制 值 描述 1 启用睡眠模式中的 EEPROM 模块并提供时钟。 0 EEPROM 模块禁用。

寄存器 80: 32/64 位宽通用定时器睡眠模式时钟门控控制寄存器（SCGCWTIMER），偏移量 0x75C

SCGCWTIMER 寄存器为软件提供启用和禁用睡眠模式中的 32/64 位定时器模块的功能。在启用时，为模块提供一个时钟。在禁用时，禁用时钟以节能。该寄存器特别为定时器模块提供与传统睡眠模式时钟门控控制 n SCGCn 寄存器相同的功能，并且具有与相应 SCGCn 位相同的位极性。

32/64 位宽通用定时器睡眠模式时钟门控控制寄存器 (SCGCWTIMER)

基址 0x400F.E000
偏移量 0x75C
类型 R/W, 复位 0x0000.0000

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	保留															
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	保留										S5	S4	S3	S2	S1	S0
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W	R/W	R/W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位/域	名称	类型	复位	描述
31:6	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
5	S5	R/W	0	32/64 位宽通用定时器 5 睡眠模式时钟门控控制 值 描述 1 启用睡眠模式中的 32/64 位宽通用定时器模块 5 并提供时钟。 0 32/64 位宽通用定时器模块 5 禁用。
4	S4	R/W	0	32/64 位宽通用定时器 4 睡眠模式时钟门控控制 值 描述 1 启用睡眠模式中的 32/64 位宽通用定时器模块 4 并提供时钟。 0 32/64 位宽通用定时器模块 4 禁用。
3	S3	R/W	0	32/64 位宽通用定时器 3 睡眠模式时钟门控控制 值 描述 1 启用睡眠模式中的 32/64 位宽通用定时器模块 3 并提供时钟。 0 32/64 位宽通用定时器模块 3 禁用。
2	S2	R/W	0	32/64 位宽通用定时器 2 睡眠模式时钟门控控制 值 描述 1 启用睡眠模式中的 32/64 位宽通用定时器模块 2 并提供时钟。 0 32/64 位宽通用定时器模块 2 禁用。

位/域	名称	类型	复位	描述
1	S1	R/W	0	32/64 位宽通用定时器 1 睡眠模式时钟门控控制 值 描述 1 启用睡眠模式中的 32/64 位宽通用定时器模块 1 并提供时钟。 0 32/64 位宽通用定时器模块 1 禁用。
0	S0	R/W	0	32/64 位宽通用定时器 0 睡眠模式时钟门控控制 值 描述 1 启用睡眠模式中的 32/64 位宽通用定时器模块 0 并提供时钟。 0 32/64 位宽通用定时器模块 0 禁用。

寄存器 81: 看门狗定时器深度睡眠模式时钟门控控制寄存器（DCGCWD），偏移量 0x800

DCGCWD 寄存器为软件提供启用和禁用深度睡眠模式中看门狗模块的功能。在启用时，为模块提供一个时钟。在禁用时，禁用时钟以节能。该寄存器特别为看门狗模块提供与传统深度睡眠模式时钟门控控制 n DCGCn 寄存器相同的功能，并且具有与相应 DCGCn 位相同的位极性。

重要: 应使用该寄存器控制看门狗模块的计时。要支持传统软件，可使用 DCGC0 寄存器。对 DCGC0 寄存器执行写操作将同时对该寄存器中的相应位执行写操作。通过对 DCGC0 寄存器的写操作更改的任何位都可以通过对 DCGC0 寄存器的读操作进行正确回读。如果软件使用该寄存器对传统外设（如 Watchdog 0）进行写操作，则写操作会产生正确操作，但是该位的值不在 DCGC0 寄存器中得到反映。如果软件使用传统和外设专用寄存器访问，则必须通过读-修改-写的操作来访问外设专用寄存器，因为该操作仅影响不在传统寄存器中的外设。通过这种方法，外设专用和传统寄存器都具有一致的信息。

看门狗定时器深度睡眠模式时钟门控控制寄存器 (DCGCWD)

基址 0x400F.E000
偏移量 0x800
类型 R/W, 复位 0x0000.0000

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	保留															
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	保留														D1	D0
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位/域	名称	类型	复位	描述
31:2	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
1	D1	R/W	0	看门狗定时器 1 深度睡眠模式时钟门控控制 值 描述 1 启用深度睡眠模式中的看门狗模块 1 并提供时钟。 0 看门狗模块 1 禁用。
0	D0	R/W	0	看门狗定时器 0 深度睡眠模式时钟门控控制 值 描述 1 启用深度睡眠模式中的看门狗模块 0 并提供时钟。 0 看门狗模块 0 禁用。

寄存器 82: 16/32 位通用定时器深度睡眠模式时钟门控控制寄存器（DCGCTIMER），偏移量 0x804

DCGCTIMER 寄存器为软件提供启用和禁用深度睡眠模式中的 16/32 位定时器模块的功能。在启用时，为模块提供一个时钟。在禁用时，禁用时钟以节能。该寄存器特别为定时器模块提供与传统深度睡眠模式时钟门控控制 n DCGCn 寄存器相同的功能，并且具有与相应 DCGCn 位相同的位极性。

重要: 应使用该寄存器控制定时器模块的计时。要支持传统软件，可使用 DCGC1 寄存器。对 DCGC1 寄存器执行写操作将同时对该寄存器中的相应位执行写操作。通过对 DCGC1 寄存器的写操作更改的任何位都可以通过对 DCGC1 寄存器的读操作进行正确回读。软件必须使用该寄存器支持不处于传统寄存器中的模块。如果软件使用该寄存器对传统外设（如 Timer 0）进行写操作，则写操作会产生正确操作，但是该位的值不在 DCGC1 寄存器中得到反映。如果软件使用传统和外设专用寄存器访问，则必须通过读-修改-写的操作来访问外设专用寄存器，因为该操作仅影响不在传统寄存器中的外设。通过这种方法，外设专用和传统寄存器都具有一致的信息。

16/32 位通用定时器深度睡眠模式时钟门控控制寄存器 (DCGCTIMER)

基址 0x400F.E000
偏移量 0x804
类型 R/W, 复位 0x0000.0000

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	保留															
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	保留										D5	D4	D3	D2	D1	D0
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W	R/W	R/W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位/域	名称	类型	复位	描述
31:6	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
5	D5	R/W	0	16/32 位通用定时器 5 深度睡眠模式时钟门控控制 值 描述 1 启用深度睡眠模式中的 16/32 位通用定时器模块 5 并提供时钟。 0 16/32 位通用定时器模块 5 禁用。
4	D4	R/W	0	16/32 位通用定时器 4 深度睡眠模式时钟门控控制 值 描述 1 启用深度睡眠模式中的 16/32 位通用定时器模块 4 并提供时钟。 0 16/32 位通用定时器模块 4 禁用。
3	D3	R/W	0	16/32 位通用定时器 3 深度睡眠模式时钟门控控制 值 描述 1 启用深度睡眠模式中的 16/32 位通用定时器模块 3 并提供时钟。 0 16/32 位通用定时器模块 3 禁用。

位/域	名称	类型	复位	描述
2	D2	R/W	0	16/32 位通用定时器 2 深度睡眠模式时钟门控控制 值 描述 1 启用深度睡眠模式中的 16/32 位通用定时器模块 2 并提供时钟。 0 16/32 位通用定时器模块 2 禁用。
1	D1	R/W	0	16/32 位通用定时器 1 深度睡眠模式时钟门控控制 值 描述 1 启用深度睡眠模式中的 16/32 位通用定时器模块 1 并提供时钟。 0 16/32 位通用定时器模块 1 禁用。
0	D0	R/W	0	16/32 位通用定时器 0 深度睡眠模式时钟门控控制 值 描述 1 启用深度睡眠模式中的 16/32 位通用定时器模块 0 并提供时钟。 0 16/32 位通用定时器模块 0 禁用。

寄存器 83: 通用输入/输出深度睡眠模式时钟门控控制寄存器（DCGCGPIO），偏移量 0x808

DCGCGPIO 寄存器为软件提供启用和禁用深度睡眠模式中的 GPIO 模块的功能。在启用时，为模块提供一个时钟。在禁用时，禁用时钟以节能。该寄存器特别为看门狗模块提供与传统深度睡眠模式时钟门控控制 n DCGCn 寄存器相同的功能，并且具有与相应 DCGCn 位相同的位极性。

重要: 应使用该寄存器控制 GPIO 模块的计时。要支持传统软件，可使用 DCGC2 寄存器。对 DCGC2 寄存器执行写操作将同时对该寄存器中的相应位执行写操作。通过对 DCGC2 寄存器的写操作更改的任何位都可以通过对 DCGC2 寄存器的读操作进行正确回读。软件必须使用该寄存器支持不处于传统寄存器中的模块。如果软件使用该寄存器写入传统外设（如 GPIO A），则写操作会产生正确操作，但是该位的值不在 DCGC2 寄存器中得到反映。如果软件使用传统和外设专用寄存器访问，则必须通过读-修改-写的操作来访问外设专用寄存器，因为该操作仅影响不在传统寄存器中的外设。通过这种方法，外设专用和传统寄存器都具有一致的信息。

通用输入/输出深度睡眠模式时钟门控控制寄存器 (DCGCGPIO)

基址 0x400F.E000
偏移量 0x808
类型 R/W, 复位 0x0000.0000

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
域	保留															
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
域	保留										D5	D4	D3	D2	D1	D0
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W	R/W	R/W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位/域	名称	类型	复位	描述
31:6	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
5	D5	R/W	0	GPIO 端口 F 深度睡眠模式时钟门控控制 值 描述 1 启用深度睡眠模式中的 GPIO 端口 F 并提供时钟。 0 GPIO 端口 F 禁用。
4	D4	R/W	0	GPIO 端口 E 深度睡眠模式时钟门控控制 值 描述 1 启用深度睡眠模式中的 GPIO 端口 E 并提供时钟。 0 GPIO 端口 E 禁用。
3	D3	R/W	0	GPIO 端口 D 深度睡眠模式时钟门控控制 值 描述 1 启用深度睡眠模式中的 GPIO 端口 D 并提供时钟。 0 GPIO 端口 D 禁用。

位/域	名称	类型	复位	描述
2	D2	R/W	0	GPIO 端口 C 深度睡眠模式时钟门控控制 值 描述 1 启用深度睡眠模式中的 GPIO 端口 C 并提供时钟。 0 GPIO 端口 C 禁用。
1	D1	R/W	0	GPIO 端口 B 深度睡眠模式时钟门控控制 值 描述 1 启用深度睡眠模式中的 GPIO 端口 B 并提供时钟。 0 GPIO 端口 B 禁用。
0	D0	R/W	0	GPIO 端口 A 深度睡眠模式时钟门控控制 值 描述 1 启用深度睡眠模式中的 GPIO 端口 A 并提供时钟。 0 GPIO 端口 A 禁用。

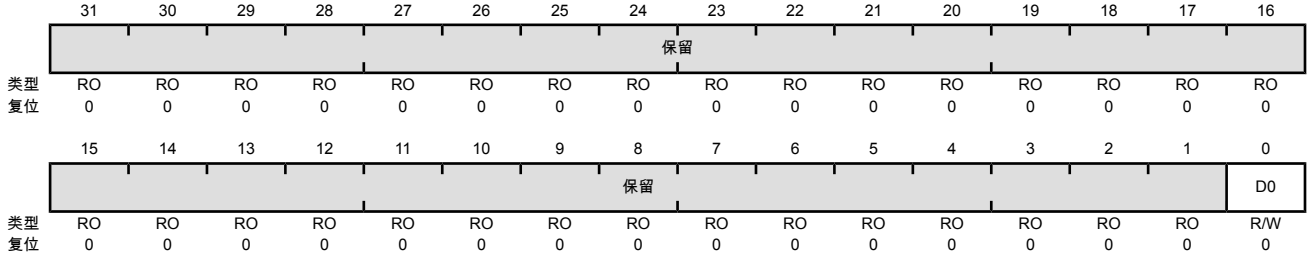
寄存器 84: 微型直接存储器访问深度睡眠模式时钟门控控制寄存器（DCGCDMA），偏移量 0x80C

DCGCDMA 寄存器为软件提供启用和禁用深度睡眠模式中的 µDMA 模块的功能。在启用时，为模块提供一个时钟。在禁用时，禁用时钟以节能。该寄存器特别为看门狗模块提供与传统深度睡眠模式时钟门控控制 n DCGCn 寄存器相同的功能，并且具有与相应 DCGCn 位相同的位极性。

重要: 应使用该寄存器控制 µDMA 模块的计时。要支持传统软件，可使用 DCGC2 寄存器。对 DCGC2 寄存器中 UDMA 位的写操作还对该寄存器中的 D0 位执行写操作。如果通过对 DCGC2 寄存器的写操作更改 UDMA 位，它可以通过读 DCGC2 寄存器进行正确回读。如果软件使用该寄存器控制 µDMA 模块，则该写操作会产生正确操作，但是 DCGC2 寄存器中的 UDMA 位不会反映 D0 位的值。如果软件使用传统和外设专用寄存器访问，则必须通过读-修改-写的操作来访问外设专用寄存器，因为该操作仅影响不在传统寄存器中的外设。通过这种方法，外设专用和传统寄存器都具有一致的信息。

微型直接存储器访问深度睡眠模式时钟门控控制寄存器 (DCGCDMA)

基址 0x400F.E000
偏移量 0x80C
类型 R/W, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:1	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
0	D0	R/W	0	µDMA 模块深度睡眠模式时钟门控控制 值 描述 1 启用深度睡眠模式中的 µDMA 模块并提供时钟。 0 µDMA 模块禁用。

寄存器 85: 休眠深度睡眠模式时钟门控控制寄存器（DCGCHIB），偏移量 0x814

DCGCHIB 寄存器为软件提供启用和禁用深度睡眠模式中休眠模块的功能。在启用时，为模块提供一个时钟。在禁用时，禁用时钟以节能。该寄存器特别为看门狗模块提供与传统深度睡眠模式时钟门控控制 n DCGCn 寄存器相同的功能，并且具有与相应 DCGCn 位相同的位极性。

重要: 应使用该寄存器控制休眠模块的计时。要支持传统软件，可使用 DCGC0 寄存器。对 DCGC0 寄存器中 HIB 位的写操作还对该寄存器中的 D0 位执行写操作。如果通过对 DCGC0 寄存器的写操作更改 HIB 位，它可以通过读 DCGC0 寄存器进行正确回读。如果软件使用该寄存器控制休眠模块，则该写操作会产生正确操作，但是DCGC0 寄存器中的 HIB 位不会反映 D0 位的值。如果软件使用传统和外设专用寄存器访问，则必须通过读-修改-写的操作来访问外设专用寄存器，因为该操作仅影响不在传统寄存器中的外设。通过这种方法，外设专用和传统寄存器都具有一致的信息。

休眠深度睡眠模式时钟门控控制寄存器 (DCGCHIB)

基址 0x400F.E000
偏移量 0x814
类型 R/W, 复位 0x0000.0001

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	保留															
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	保留															D0
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

位/域	名称	类型	复位	描述
31:1	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
0	D0	R/W	1	休眠模块深度睡眠模式时钟门控控制 值 描述 1 启用深度睡眠模式中的休眠模块并提供时钟。 0 休眠模块禁用。

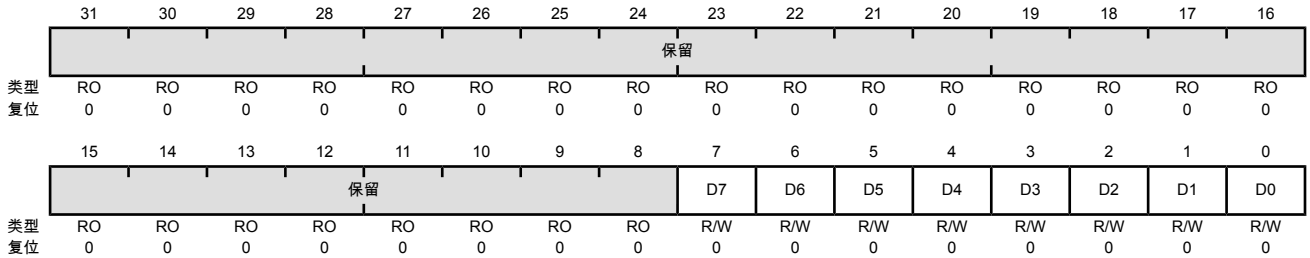
寄存器 86: 通用异步收发器深度睡眠模式时钟门控控制寄存器（DCGCUART），偏移量 0x818

DCGCUART 寄存器为软件提供启用和禁用深度睡眠模式中的 UART 模块的功能。在启用时，为模块提供一个时钟。在禁用时，禁用时钟以节能。该寄存器特别为看门狗模块提供与传统深度睡眠模式时钟门控控制 n DCGCn 寄存器相同的功能，并且具有与相应 DCGCn 位相同的位极性。

重要: 应使用该寄存器控制 UART 模块的计时。要支持传统软件，可使用 DCGC1 寄存器。对 DCGC1 寄存器执行写操作将同时对该寄存器中的相应位执行写操作。通过对 DCGC1 寄存器的写操作更改的任何位都可以通过对 DCGC1 寄存器的读操作进行正确回读。软件必须使用该寄存器支持不处于传统寄存器中的模块。如果软件使用该寄存器对传统外设（如 UART0）进行写操作，则写操作会产生正确操作，但是该位的值不在 DCGC1 寄存器中得到反映。如果软件使用传统和外设专用寄存器访问，则必须通过读-修改-写的操作来访问外设专用寄存器，因为该操作仅影响不在传统寄存器中的外设。通过这种方法，外设专用和传统寄存器都具有一致的信息。

通用异步收发器深度睡眠模式时钟门控控制寄存器 (DCGCUART)

基址 0x400F.E000
偏移量 0x818
类型 R/W, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:8	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
7	D7	R/W	0	UART 模块 7 深度睡眠模式时钟门控控制 值 描述 1 启用深度睡眠模式中的 UART 模块 7 并提供时钟。 0 UART 模块 7 禁用。
6	D6	R/W	0	UART 模块 6 深度睡眠模式时钟门控控制 值 描述 1 启用深度睡眠模式中的 UART 模块 6 并提供时钟。 0 UART 模块 6 禁用。
5	D5	R/W	0	UART 模块 5 深度睡眠模式时钟门控控制 值 描述 1 启用深度睡眠模式中的 UART 模块 5 并提供时钟。 0 UART 模块 5 禁用。

位/域	名称	类型	复位	描述
4	D4	R/W	0	UART 模块 4 深度睡眠模式时钟门控控制 值 描述 1 启用深度睡眠模式中的 UART 模块 4 并提供时钟。 0 UART 模块 4 禁用。
3	D3	R/W	0	UART 模块 3 深度睡眠模式时钟门控控制 值 描述 1 启用深度睡眠模式中的 UART 模块 3 并提供时钟。 0 UART 模块 3 禁用。
2	D2	R/W	0	UART 模块 2 深度睡眠模式时钟门控控制 值 描述 1 启用深度睡眠模式中的 UART 模块 2 并提供时钟。 0 UART 模块 2 禁用。
1	D1	R/W	0	UART 模块 1 深度睡眠模式时钟门控控制 值 描述 1 启用深度睡眠模式中的 UART 模块 1 并提供时钟。 0 UART 模块 1 禁用。
0	D0	R/W	0	UART 模块 0 深度睡眠模式时钟门控控制 值 描述 1 启用深度睡眠模式中的 UART 模块 0 并提供时钟。 0 UART 模块 0 禁用。

寄存器 87: 同步串行接口深度睡眠模式时钟门控控制寄存器（DCGCSSI），偏移量 0x81C

DCGCSSI 寄存器为软件提供启用和禁用深度睡眠模式中的 SSI 模块的功能。在启用时，为模块提供一个时钟。在禁用时，禁用时钟以节能。该寄存器特别为看门狗模块提供与传统深度睡眠模式时钟门控控制 n DCGCn 寄存器相同的功能，并且具有与相应 DCGCn 位相同的位极性。

重要: 应使用该寄存器控制 SSI 模块的计时。要支持传统软件，可使用 DCGC1 寄存器。对 DCGC1 寄存器执行写操作将同时对该寄存器中的相应位执行写操作。通过对 DCGC1 寄存器的写操作更改的任何位都可以通过对 DCGC1 寄存器的读操作进行正确回读。软件必须使用该寄存器支持不处于传统寄存器中的模块。如果软件使用该寄存器对传统外设（如 SSI0）进行写操作，则写操作会产生正确操作，但是该位的值不在 DCGC1 寄存器中得到反映。如果软件使用传统和外设专用寄存器访问，则必须通过读-修改-写的操作来访问外设专用寄存器，因为该操作仅影响不在传统寄存器中的外设。通过这种方法，外设专用和传统寄存器都具有一致的信息。

同步串行接口深度睡眠模式时钟门控控制寄存器 (DCGCSSI)

基址 0x400F.E000
偏移量 0x81C
类型 R/W, 复位 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	保留															
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	保留												D3	D2	D1	D0
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位/域	名称	类型	复位	描述
31:4	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
3	D3	R/W	0	SSI 模块 3 深度睡眠模式时钟门控控制 值 描述 1 启用深度睡眠模式中的 SSI 模块 3 并提供时钟。 0 SSI 模块 3 禁用。
2	D2	R/W	0	SSI 模块 2 深度睡眠模式时钟门控控制 值 描述 1 启用深度睡眠模式中的 SSI 模块 2 并提供时钟。 0 SSI 模块 2 禁用。
1	D1	R/W	0	SSI 模块 1 深度睡眠模式时钟门控控制 值 描述 1 启用深度睡眠模式中的 SSI 模块 1 并提供时钟。 0 SSI 模块 1 禁用。

位/域	名称	类型	复位	描述
0	D0	R/W	0	SSI 模块 0 深度睡眠模式时钟门控控制 值 描述 1 启用深度睡眠模式中的 SSI 模块 0 并提供时钟。 0 SSI 模块 0 禁用。

寄存器 88: 内部集成电路深度睡眠模式时钟门控控制寄存器（DCGCI2C），偏移量 0x820

DCGCI2C 寄存器为软件提供启用和禁用深度睡眠模式中的 I^2C 模块的功能。在启用时，为模块提供一个时钟。在禁用时，禁用时钟以节能。该寄存器特别为看门狗模块提供与传统深度睡眠模式时钟门控控制 n DCGCn 寄存器相同的功能，并且具有与相应 DCGCn 位相同的位极性。

重要: 应使用该寄存器控制 I^2C 模块的计时。要支持传统软件，可使用 DCGC1 寄存器。对 DCGC1 寄存器执行写操作将同时对该寄存器中的相应位执行写操作。通过对 DCGC1 寄存器的写操作更改的任何位都可以通过对 DCGC1 寄存器的读操作进行正确回读。软件必须使用该寄存器支持不处于传统寄存器中的模块。如果软件使用该寄存器对传统外设（如 I^2C0）进行写操作，则写操作会产生正确操作，但是该位的值不在 DCGC1 寄存器中得到反映。如果软件使用传统和外设专用寄存器访问，则必须通过读-修改-写的操作来访问外设专用寄存器，因为该操作仅影响不在传统寄存器中的外设。通过这种方法，外设专用和传统寄存器都具有一致的信息。

内部集成电路深度睡眠模式时钟门控控制寄存器 (DCGCI2C)

基址 0x400F.E000
偏移量 0x820
类型 R/W, 复位 0x0000.0000

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
域	保留															
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
域	保留												D3	D2	D1	D0
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位/域	名称	类型	复位	描述
31:4	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
3	D3	R/W	0	I^2C 模块 3 深度睡眠模式时钟门控控制 值 描述 1 启用深度睡眠模式中的 I^2C 模块 3 并提供时钟。 0 I^2C 模块 3 禁用。
2	D2	R/W	0	I^2C 模块 2 深度睡眠模式时钟门控控制 值 描述 1 启用深度睡眠模式中的 I^2C 模块 2 并提供时钟。 0 I^2C 模块 2 禁用。
1	D1	R/W	0	I^2C 模块 1 深度睡眠模式时钟门控控制 值 描述 1 启用深度睡眠模式中的 I^2C 模块 1 并提供时钟。 0 I^2C 模块 1 禁用。

位/域	名称	类型	复位	描述
0	D0	R/W	0	I^2C 模块 0 深度睡眠模式时钟门控控制 值 描述 1 启用深度睡眠模式中的 I^2C 模块 0 并提供时钟。 0 I^2C 模块 0 禁用。

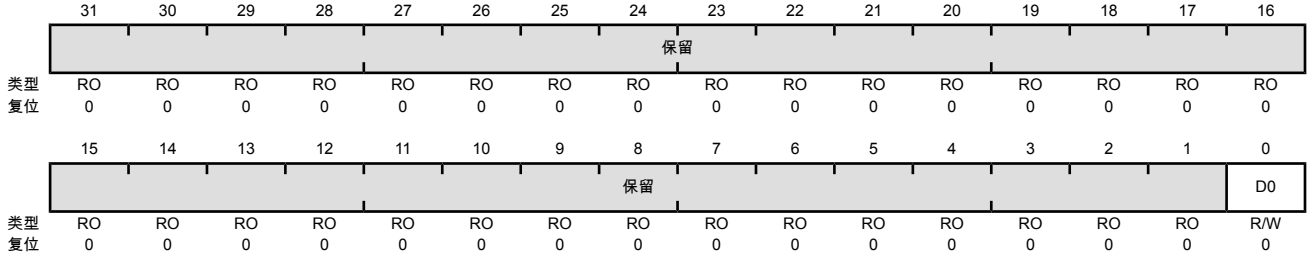
寄存器 89: 控制器局域网深度睡眠模式时钟门控控制寄存器（DCGCCAN），偏移量 0x834

DCGCCAN 寄存器为软件提供启用和禁用深度睡眠模式中的 CAN 模块的功能。在启用时，为模块提供一个时钟。在禁用时，禁用时钟以节能。该寄存器特别为看门狗模块提供与传统深度睡眠模式时钟门控控制 n DCGCn 寄存器相同的功能，并且具有与相应 DCGCn 位相同的位极性。

重要: 应使用该寄存器控制 CAN 模块的计时。要支持传统软件，可使用 DCGC0 寄存器。对 DCGC0 寄存器执行写操作将同时对该寄存器中的相应位执行写操作。通过对 DCGC0 寄存器的写操作更改的任何位都可以通过对 DCGC0 寄存器的读操作进行正确回读。如果软件使用该寄存器对传统外设（如 CAN0）进行写操作，则写操作会产生正确操作，但是该位的值不在 DCGC0 寄存器中得到反映。如果软件使用传统和外设专用寄存器访问，则必须通过读-修改-写的操作来访问外设专用寄存器，因为该操作仅影响不在传统寄存器中的外设。通过这种方法，外设专用和传统寄存器都具有一致的信息。

控制器局域网深度睡眠模式时钟门控控制寄存器 (DCGCCAN)

基址 0x400F.E000
偏移量 0x834
类型 R/W, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:1	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
0	D0	R/W	0	CAN 模块 0 深度睡眠模式时钟门控控制 值 描述 1 启用深度睡眠模式中的 CAN 模块 0 并提供时钟。 0 CAN 模块 0 禁用。

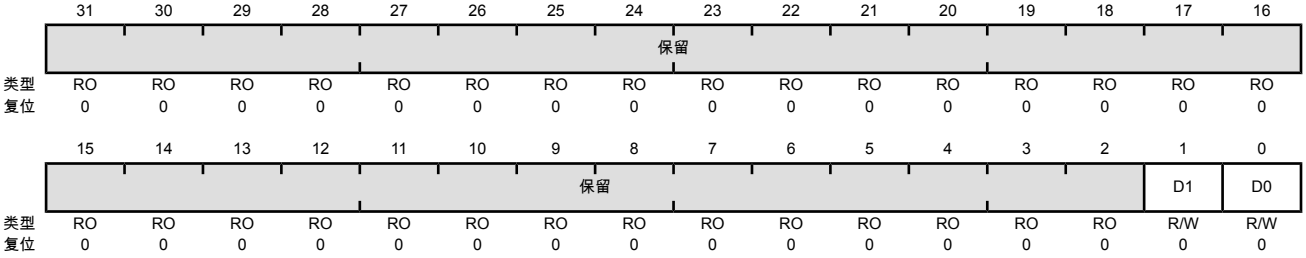
寄存器 90: 模数转换器深度睡眠模式时钟门控控制寄存器（DCGCADC），偏移量 0x838

DCGCADC 寄存器为软件提供启用和禁用深度睡眠模式中的 ADC 模块的功能。在启用时，为模块提供一个时钟。在禁用时，禁用时钟以节能。该寄存器特别为看门狗模块提供与传统深度睡眠模式时钟门控控制 n DCGCn 寄存器相同的功能，并且具有与相应 DCGCn 位相同的位极性。

重要: 应使用该寄存器控制 ADC 模块的计时。要支持传统软件，可使用 DCGC0 寄存器。对 DCGC0 寄存器执行写操作将同时对该寄存器中的相应位执行写操作。通过对 DCGC0 寄存器的写操作更改的任何位都可以通过对 DCGC0 寄存器的读操作进行正确回读。如果软件使用该寄存器对传统外设（如 ADC0）进行写操作，则写操作会产生正确操作，但是该位的值不在 DCGC0 寄存器中得到反映。如果软件使用传统和外设专用寄存器访问，则必须通过读-修改-写的操作来访问外设专用寄存器，因为该操作仅影响不在传统寄存器中的外设。通过这种方法，外设专用和传统寄存器都具有一致的信息。

模数转换器深度睡眠模式时钟门控控制寄存器 (DCGCADC)

基址 0x400F.E000
偏移量 0x838
类型 R/W, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:2	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
1	D1	R/W	0	ADC 模块 1 深度睡眠模式时钟门控控制 值 描述 1 启用深度睡眠模式中的 ADC 模块 1 并提供时钟。 0 ADC 模块 1 禁用。
0	D0	R/W	0	ADC 模块 0 深度睡眠模式时钟门控控制 值 描述 1 启用深度睡眠模式中的 ADC 模块 0 并提供时钟。 0 ADC 模块 0 禁用。

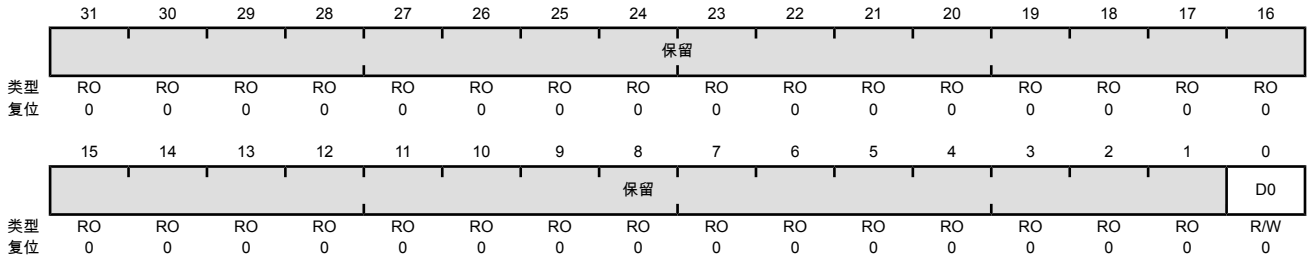
寄存器 91: 模拟比较器深度睡眠模式时钟门控控制寄存器（DCGCACMP），偏移量 0x83C

DCGCACMP 寄存器为软件提供启用和禁用深度睡眠模式中模拟比较器模块的功能。在启用时，为模块提供一个时钟。在禁用时，禁用时钟以节能。该寄存器特别为看门狗模块提供与传统深度睡眠模式时钟门控控制 n DCGCn 寄存器相同的功能，并且具有与相应 DCGCn 位相同的位极性。

重要: 应使用该寄存器控制模拟比较器模块的计时。要支持传统软件，可使用 DCGC1 寄存器。对 DCGC1 寄存器中 COMPn 位的写操作还对该寄存器中的 D0 位执行写操作。如果通过对 DCGC1 寄存器的写操作置位任何 COMPn 位，它可以在对 DCGC1 寄存器进行读操作时进行正确回读。如果软件使用该寄存器更改模拟比较器模块的计时，则写操作会产生正确操作，但是 DCGC1 寄存器的 COMPn 位不反映值 D0。如果软件使用传统和外设专用寄存器访问，则必须通过读-修改-写的操作来访问外设专用寄存器，因为该操作仅影响不在传统寄存器中的外设。通过这种方法，外设专用和传统寄存器都具有一致的信息。

模拟比较器深度睡眠模式时钟门控控制寄存器 (DCGCACMP)

基址 0x400F.E000
偏移量 0x83C
类型 R/W, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:1	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
0	D0	R/W	0	模拟比较器模块 0 深度睡眠模式时钟门控控制 值 描述 1 启用深度睡眠模式中的模拟比较器模块并提供时钟。 0 模拟比较器模块禁用。

寄存器 92: EEPROM 深度睡眠模式时钟门控控制寄存器（DCGCEEPROM），偏移量 0x858

DCGCEEPROM 寄存器为软件提供启用和禁用深度睡眠模式中的 EEPROM 模块的功能。在启用时，为模块提供一个时钟。在禁用时，禁用时钟以节能。

EEPROM 深度睡眠模式时钟门控控制寄存器 (DCGCEEPROM)

基址 0x400F.E000
偏移量 0x858
类型 R/W, 复位 0x0000.0000

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	保留															
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	保留															D0
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位/域	名称	类型	复位	描述
31:1	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
0	D0	R/W	0	EEPROM 模块深度睡眠模式时钟门控控制

值 描述

1 启用 EEPROM 模式中的休眠模块并提供时钟。

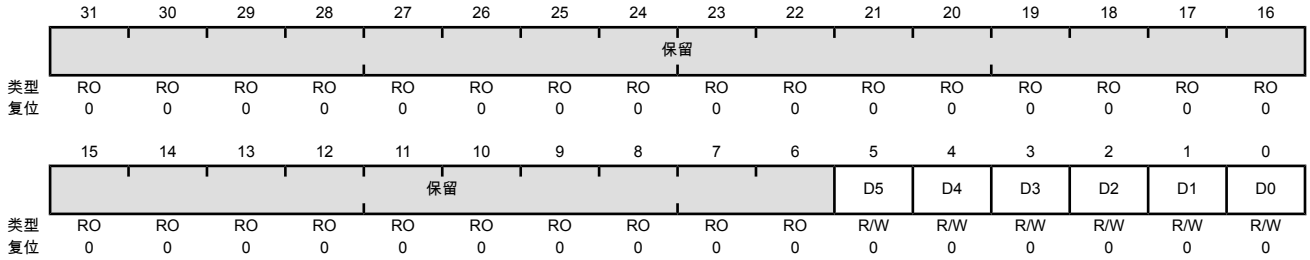
0 EEPROM 模块禁用。

寄存器 93: 32/64 位宽通用定时器深度睡眠模式时钟门控控制寄存器（DCGCWTIMER），偏移量 0x85C

DCGCWTIMER 寄存器为软件提供启用和禁用深度睡眠模式中的 32/64 位宽定时器模块的功能。在启用时，为模块提供一个时钟。在禁用时，禁用时钟以节能。该寄存器特别为定时器模块提供与传统深度睡眠模式时钟门控控制 n DCGCn 寄存器相同的功能，并且具有与相应 DCGCn 位相同的位极性。

32/64 位宽通用定时器深度睡眠模式时钟门控控制寄存器 (DCGCWTIMER)

基址 0x400F.E000
偏移量 0x85C
类型 R/W, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:6	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
5	D5	R/W	0	32/64 位宽通用定时器 5 深度睡眠模式时钟门控控制 值 描述 1 启用深度睡眠模式中的 32/64 位宽通用定时器模块 5 并提供时钟。 0 32/64 位宽通用定时器模块 5 禁用。
4	D4	R/W	0	32/64 位宽通用定时器 4 深度睡眠模式时钟门控控制 值 描述 1 启用深度睡眠模式中的 32/64 位宽通用定时器模块 4 并提供时钟。 0 32/64 位宽通用定时器模块 4 禁用。
3	D3	R/W	0	32/64 位宽通用定时器 3 深度睡眠模式时钟门控控制 值 描述 1 启用深度睡眠模式中的 32/64 位宽通用定时器模块 3 并提供时钟。 0 32/64 位宽通用定时器模块 3 禁用。
2	D2	R/W	0	32/64 位宽通用定时器 2 深度睡眠模式时钟门控控制 值 描述 1 启用深度睡眠模式中的 32/64 位宽通用定时器模块 2 并提供时钟。 0 32/64 位宽通用定时器模块 2 禁用。

位/域	名称	类型	复位	描述
1	D1	R/W	0	32/64 位宽通用定时器 1 深度睡眠模式时钟门控控制 值 描述 1 启用深度睡眠模式中的 32/64 位宽通用定时器模块 1 并提供时钟。 0 32/64 位宽通用定时器模块 1 禁用。
0	D0	R/W	0	32/64 位宽通用定时器 0 深度睡眠模式时钟门控控制 值 描述 1 启用深度睡眠模式中的 32/64 位宽通用定时器模块 0 并提供时钟。 0 32/64 位宽通用定时器模块 0 禁用。

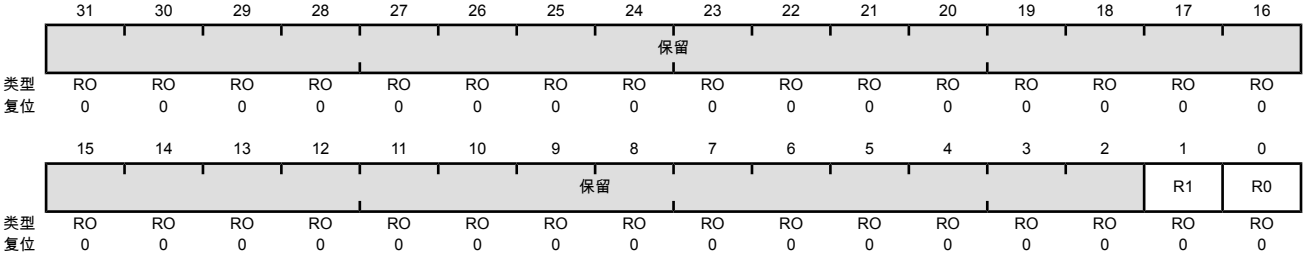
寄存器 94: 看门狗定时器外设就绪寄存器（PRWD），偏移量 0xA00

PRWD 寄存器指示看门狗模块是否就绪，可由软件在功率状态发生更改、运行模式计时或复位之后访问。如果相应 RCGCWD 位发生更改，则运行模式计时更改开始。如果相应 SRWD 位从 0 更改为 1，则复位随之改变。

PRWD 位由于任何上述事件清零，并且不会再次置位，直到模块完全通电、启用并内部复位。

看门狗定时器外设就绪寄存器 (PRWD)

基址 0x400F.E000
偏移量 0xA00
类型 RO, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:2	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
1	R1	RO	0	看门狗定时器 1 外设就绪 值 描述 1 看门狗模块 1 已可以访问。 0 看门狗模块 1 尚不可以访问。未使用时钟、未通电或处于完成复位序列的过程中。
0	R0	RO	0	看门狗定时器 0 外设就绪 值 描述 1 看门狗模块 0 已可以访问。 0 看门狗模块 0 尚不可以访问。未使用时钟、未通电或处于完成复位序列的过程中。

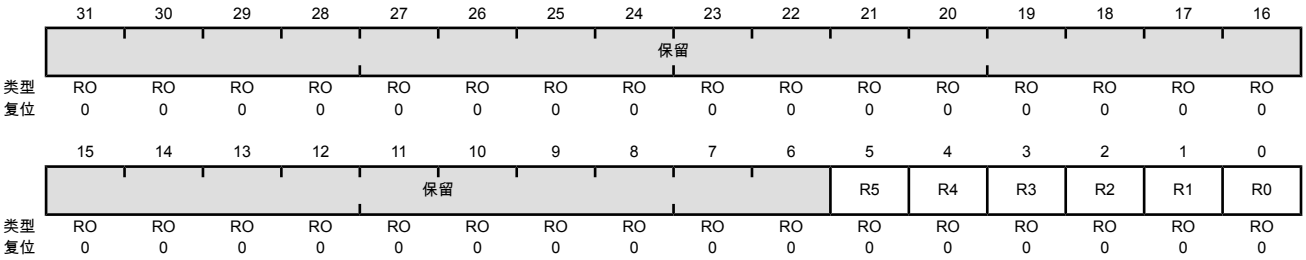
寄存器 95: 16/32 位通用定时器外设就绪寄存器（PRTIMER），偏移量 0xA04

PRTIMER 寄存器指示定时器模块是否就绪，可由软件在功率状态发生更改、运行模式计时或复位之后访问。如果相应 RCGCTIMER 位发生更改，则运行模式计时随之改变。如果相应 SRTIMER 位从 0 更改为 1，则复位随之改变。

发生任何上述事件时，PRTIMER 位将清零，并且不会再次置位，直到模块完全通电、启用并内部复位。

16/32 位通用定时器外设就绪寄存器 (PRTIMER)

基址 0x400F.E000
偏移量 0xA04
类型 RO, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:6	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
5	R5	RO	0	16/32 位通用定时器 5 外设就绪 值 描述 1 16/32 位定时器模块 5 已可以访问。 0 16/32 位定时器模块 5 尚不可以访问。未使用时钟、未通电或处于完成复位序列的过程中。
4	R4	RO	0	16/32 位通用定时器 4 外设就绪 值 描述 1 16/32 位定时器模块 4 已可以访问。 0 16/32 位定时器模块 4 尚不可以访问。未使用时钟、未通电或处于完成复位序列的过程中。
3	R3	RO	0	16/32 位通用定时器 3 外设就绪 值 描述 1 16/32 位定时器模块 3 已可以访问。 0 16/32 位定时器模块 3 尚不可以访问。未使用时钟、未通电或处于完成复位序列的过程中。

位/域	名称	类型	复位	描述
2	R2	RO	0	16/32 位通用定时器 2 外设就绪 值 描述 1 16/32 位定时器模块 2 已可以访问。 0 16/32 位定时器模块 2 尚不可以访问。未使用时钟、未通电或处于完成复位序列的过程中。
1	R1	RO	0	16/32 位通用定时器 1 外设就绪 值 描述 1 16/32 位定时器模块 1 已可以访问。 0 16/32 位定时器模块 1 尚不可以访问。未使用时钟、未通电或处于完成复位序列的过程中。
0	R0	RO	0	16/32 位通用定时器 0 外设就绪 值 描述 1 16/32 位定时器模块 0 已可以访问。 0 16/32 位定时器模块 0 尚不可以访问。未使用时钟、未通电或处于完成复位序列的过程中。

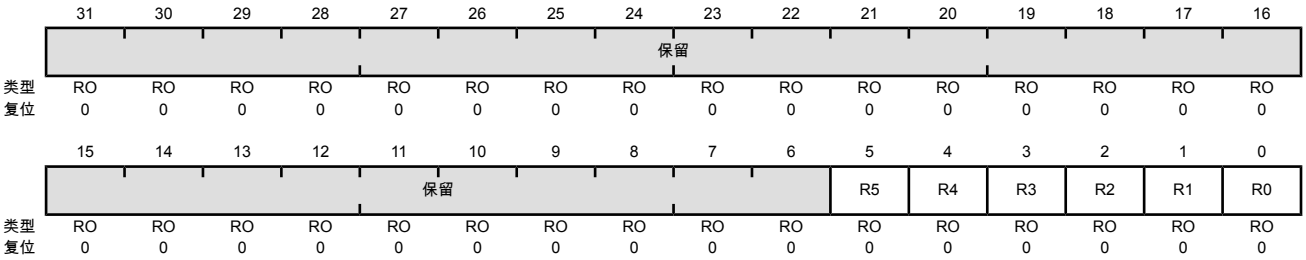
寄存器 96: 通用输入/输出外设就绪寄存器（PRGPIO），偏移量 0xA08

PRGPIO 寄存器指示 GPIO 模块是否就绪，可由软件在功率状态发生更改、运行模式计时或复位之后访问。如果相应 RCGCGPIO 位发生更改，则运行模式计时随之改变。如果相应 SRGPIO 位从 0 更改为 1，则复位随之改变。

发生任何上述事件时，PRGPIO 位将清零，并且不会再次置位，直到模块完全通电、启用并内部复位。

通用输入/输出外设就绪寄存器 (PRGPIO)

基址 0x400F.E000
偏移量 0xA08
类型 RO, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:6	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
5	R5	RO	0	GPIO 端口 F 外设就绪 值 描述 1 GPIO 端口 F 已可以访问。 0 GPIO 端口 F 尚不可以访问。未使用时钟、未通电或处于完成复位序列的过程中。
4	R4	RO	0	GPIO 端口 E 外设就绪 值 描述 1 GPIO 端口 E 已可以访问。 0 GPIO 端口 E 尚不可以访问。未使用时钟、未通电或处于完成复位序列的过程中。
3	R3	RO	0	GPIO 端口 D 外设就绪 值 描述 1 GPIO 端口 D 已可以访问。 0 GPIO 端口 D 尚不可以访问。未使用时钟、未通电或处于完成复位序列的过程中。

位/域	名称	类型	复位	描述
2	R2	RO	0	GPIO 端口 C 外设就绪 值 描述 1 GPIO 端口 C 已可以访问。 0 GPIO 端口 C 尚不可以访问。未使用时钟、未通电或处于完成复位序列的过程中。
1	R1	RO	0	GPIO 端口 B 外设就绪 值 描述 1 GPIO 端口 B 已可以访问。 0 GPIO 端口 B 尚不可以访问。未使用时钟、未通电或处于完成复位序列的过程中。
0	R0	RO	0	GPIO 端口 A 外设就绪 值 描述 1 GPIO 端口 A 已可以访问。 0 GPIO 端口 A 尚不可以访问。未使用时钟、未通电或处于完成复位序列的过程中。

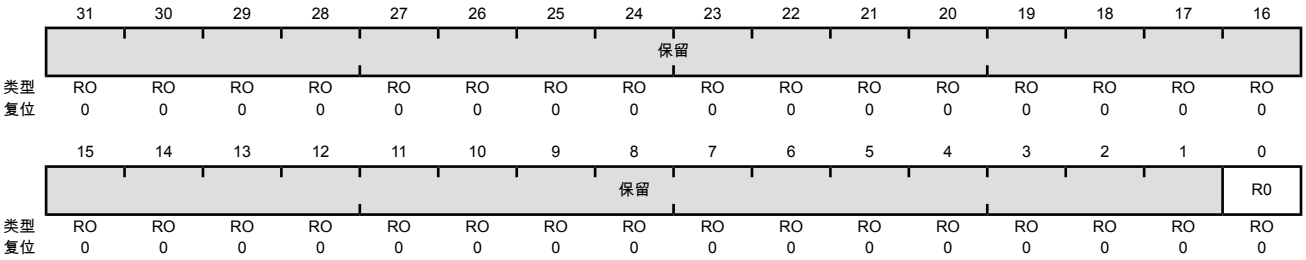
寄存器 97: 微型直接存储器访问外设就绪寄存器（PRDMA），偏移量 0xA0C

PRDMA 寄存器指示 µDMA 模块是否就绪，可由软件在功率状态发生更改、运行模式计时或复位之后访问。如果相应 RCGCDMA 位发生更改，则运行模式计时随之改变。如果相应 SRDMA 位从 0 更改为 1，则复位随之改变。

发生任何上述事件时，PRDMA 位将清零，并且不会再次置位，直到模块完全通电、启用并内部复位。

微型直接存储器访问外设就绪寄存器 (PRDMA)

基址 0x400F.E000
偏移量 0xA0C
类型 RO, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:1	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
0	R0	RO	0	µDMA 模块外设就绪 值 描述 1 µDMA 模块已可以访问。 0 µDMA 模块尚不可以访问。未使用时钟、未通电或处于完成复位序列的过程中。

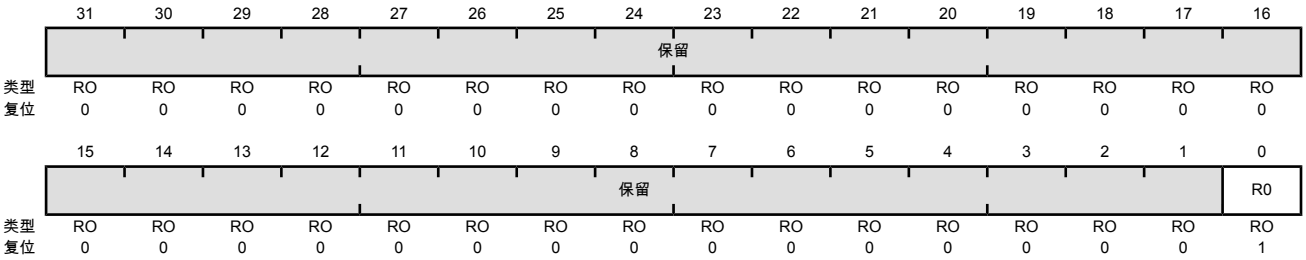
寄存器 98: 休眠外设就绪寄存器（PRHIB），偏移量 0xA14

PRHIB 寄存器指示休眠模块是否就绪，可由软件在功率状态发生更改、运行模式计时或复位之后访问。如果相应 RCGCHIB 位发生更改，则运行模式计时随之改变。如果相应 SRHIB 位从 0 更改为 1，则复位随之改变。

发生任何上述事件时，PRHIB 位将清零，并且不会再次置位，直到模块完全通电、启用并内部复位。

休眠外设就绪寄存器 (PRHIB)

基址 0x400F.E000
偏移量 0xA14
类型 RO, 复位 0x0000.0001



位/域	名称	类型	复位	描述
31:1	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
0	R0	RO	1	休眠模块外设就绪 值 描述 1 休眠模块已可以访问。 0 休眠模块尚不可以访问。未使用时钟、未通电或处于完成复位序列的过程中。

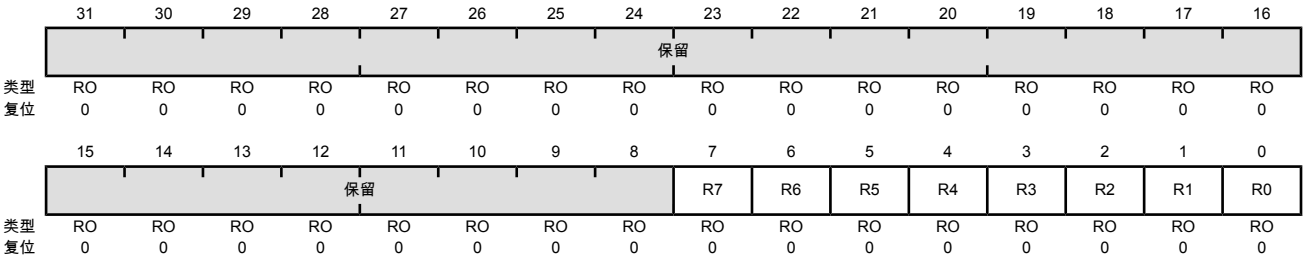
寄存器 99: 通用异步收发器外设就绪寄存器（PRUART），偏移量 0xA18

PRUART 寄存器指示UART 模块是否就绪，可由软件在功率状态发生更改、运行模式计时或复位之后访问。如果相应 RCGCUART 位发生更改，则运行模式计时随之改变。如果相应 SRUART 位从 0 更改为 1，则复位随之改变。

发生任何上述事件时，PRUART 位将清零，并且不会再次置位，直到模块完全通电、启用并内部复位。

通用异步收发器外设就绪寄存器 (PRUART)

基址 0x400F.E000
偏移量 0xA18
类型 RO, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:8	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
7	R7	RO	0	UART 模块 7 外设就绪 值 描述 1 UART 模块 7 已可以访问。 0 UART 模块 7 尚不可以访问。未使用时钟、未通电或处于完成复位序列的过程中。
6	R6	RO	0	UART 模块 6 外设就绪 值 描述 1 UART 模块 6 已可以访问。 0 UART 模块 6 尚不可以访问。未使用时钟、未通电或处于完成复位序列的过程中。
5	R5	RO	0	UART 模块 5 外设就绪 值 描述 1 UART 模块 5 已可以访问。 0 UART 模块 5 尚不可以访问。未使用时钟、未通电或处于完成复位序列的过程中。

位/域	名称	类型	复位	描述
4	R4	RO	0	UART 模块 4 外设就绪 值 描述 1 UART 模块 4 已可以访问。 0 UART 模块 4 尚不可以访问。未使用时钟、未通电或处于完成复位序列的过程中。
3	R3	RO	0	UART 模块 3 外设就绪 值 描述 1 UART 模块 3 已可以访问。 0 UART 模块 3 尚不可以访问。未使用时钟、未通电或处于完成复位序列的过程中。
2	R2	RO	0	UART 模块 2 外设就绪 值 描述 1 UART 模块 2 已可以访问。 0 UART 模块 2 尚不可以访问。未使用时钟、未通电或处于完成复位序列的过程中。
1	R1	RO	0	UART 模块 1 外设就绪 值 描述 1 UART 模块 1 已可以访问。 0 UART 模块 1 尚不可以访问。未使用时钟、未通电或处于完成复位序列的过程中。
0	R0	RO	0	UART 模块 0 外设就绪 值 描述 1 UART 模块 0 已可以访问。 0 UART 模块 0 尚不可以访问。未使用时钟、未通电或处于完成复位序列的过程中。

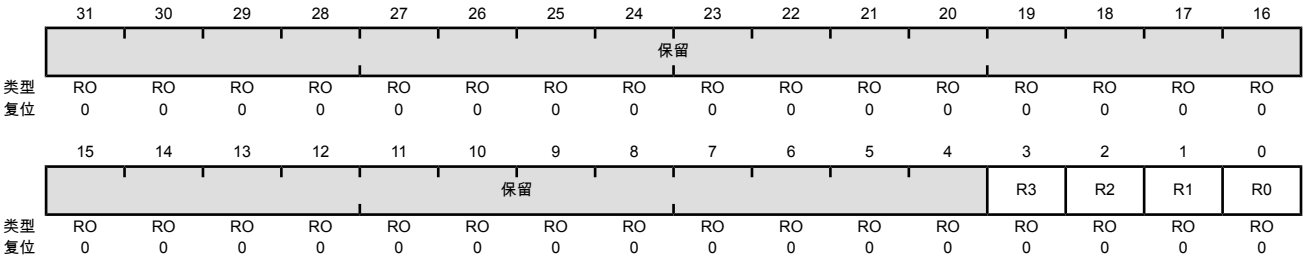
寄存器 100: 同步串行接口外设就绪寄存器（PRSSI），偏移量 0xA1C

PRSSI 寄存器指示 SSI 模块是否就绪，可由软件在功率状态发生更改、运行模式计时或复位之后访问。如果相应 RCGCSSI 位发生更改，则运行模式计时随之改变。如果相应 SRSSI 位从 0 更改为 1，则复位随之改变。

发生任何上述事件时，PRSSI 位将清零，并且不会再次置位，直到模块完全通电、启用并内部复位。

同步串行接口外设就绪寄存器 (PRSSI)

基址 0x400F.E000
偏移量 0xA1C
类型 RO, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:4	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
3	R3	RO	0	SSI 模块 3 外设就绪 值 描述 1 SSI 模块 3 已可以访问。 0 SSI 模块 3 尚不可以访问。未使用时钟、未通电或处于完成复位序列的过程中。
2	R2	RO	0	SSI 模块 2 外设就绪 值 描述 1 SSI 模块 2 已可以访问。 0 SSI 模块 2 尚不可以访问。未使用时钟、未通电或处于完成复位序列的过程中。
1	R1	RO	0	SSI 模块 1 外设就绪 值 描述 1 SSI 模块 1 已可以访问。 0 SSI 模块 1 尚不可以访问。未使用时钟、未通电或处于完成复位序列的过程中。

位/域	名称	类型	复位	描述
0	R0	RO	0	SSI 模块 0 外设就绪

值 描述

1 SSI 模块 0 已可以访问。

0 SSI 模块 0 尚不可以访问。未使用时钟、未通电或处于完成复位序列的过程中。

寄存器 101: 内部集成电路外设就绪寄存器（PRI2C），偏移量 0xA20

PRI2C 寄存器指示 I²C 模块是否就绪，可由软件在功率状态发生更改、运行模式计时或复位之后访问。如果相应 RCGCI2C 位发生更改，则运行模式计时随之改变。如果相应 SRI2C 位从 0 更改为 1，则复位随之改变。

发生任何上述事件时，PRI2C 位将清零，并且不会再次置位，直到模块完全通电、启用并内部复位。

内部集成电路外设就绪寄存器 (PRI2C)

基址 0x400F.E000
偏移量 0xA20
类型 RO, 复位 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	保留															
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	保留												R3	R2	R1	R0
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位/域	名称	类型	复位	描述
31:4	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
3	R3	RO	0	I²C 模块 3 外设就绪 值 描述 1 I²C 模块 3 已可以访问。 0 I²C 模块 3 尚不可以访问。未使用时钟、未通电或处于完成复位序列的过程中。
2	R2	RO	0	I²C 模块 2 外设就绪 值 描述 1 I²C 模块 2 已可以访问。 0 I²C 模块 2 尚不可以访问。未使用时钟、未通电或处于完成复位序列的过程中。
1	R1	RO	0	I²C 模块 1 外设就绪 值 描述 1 I²C 模块 1 已可以访问。 0 I²C 模块 1 尚不可以访问。未使用时钟、未通电或处于完成复位序列的过程中。

位/域	名称	类型	复位	描述
0	R0	RO	0	I^2C 模块 0 外设就绪

值 描述

1 I^2C 模块 0 已可以访问。

0 I^2C 模块 0 尚不可以访问。未使用时钟、未通电或处于完成复位序列的过程中。

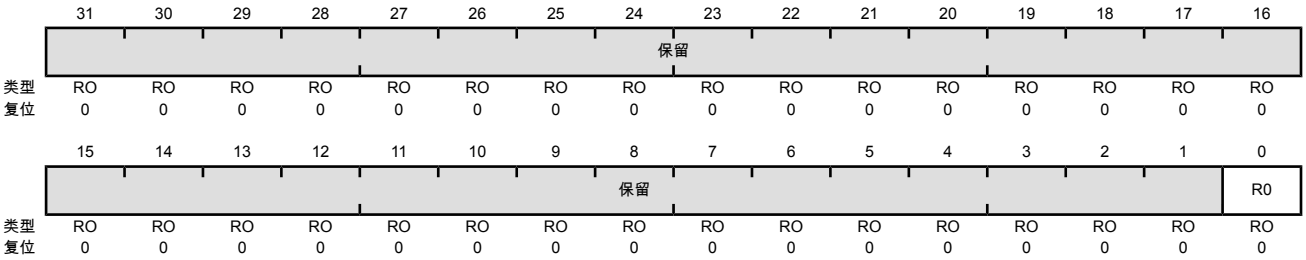
寄存器 102: 控制器局域网外设就绪寄存器（PRCAN），偏移量 0xA34

PRCAN 寄存器指示 CAN 模块是否就绪，可由软件在功率状态发生更改、运行模式计时或复位之后访问。如果相应 RCGCCAN 位发生更改，则运行模式计时随之改变。如果相应 SRCAN 位从 0 更改为 1，则复位随之改变。

发生任何上述事件时，PRCAN 位将清零，并且不会再次置位，直到模块完全通电、启用并内部复位。

控制器局域网外设就绪寄存器 (PRCAN)

基址 0x400F.E000
偏移量 0xA34
类型 RO, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:1	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
0	R0	RO	0	CAN 模块 0 外设就绪 值 描述 1 CAN 模块 0 已可以访问。 0 CAN 模块 0 尚不可以访问。未使用时钟、未通电或处于完成复位序列的过程中。

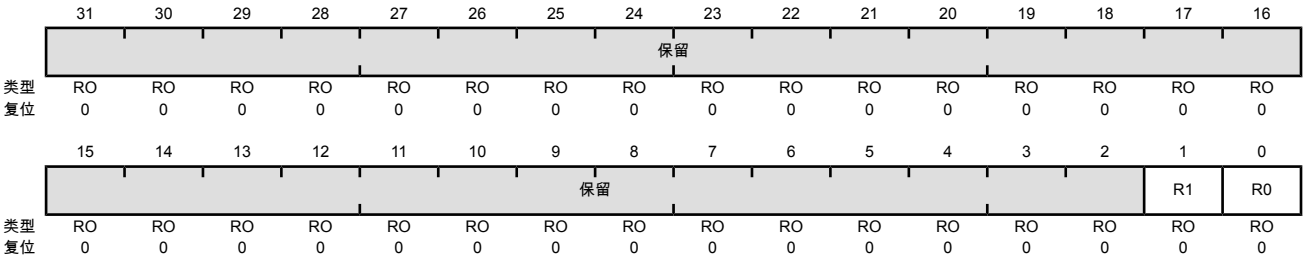
寄存器 103: 模数转换器外设就绪寄存器（PRADC），偏移量 0xA38

PRADC 寄存器指示 ADC 模块是否就绪，可由软件在功率状态发生更改、运行模式计时或复位之后访问。如果相应 RCGCADC 位发生更改，则运行模式计时随之改变。如果相应 SRADC 位从 0 更改为 1，则复位随之改变。

发生任何上述事件时，PRADC 位将清零，并且不会再次置位，直到模块完全通电、启用并内部复位。

模数转换器外设就绪寄存器 (PRADC)

基址 0x400F.E000
偏移量 0xA38
类型 RO, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:2	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
1	R1	RO	0	ADC 模块 1 外设就绪 值 描述 1 ADC 模块 1 已可以访问。 0 ADC 模块 1 尚不可以访问。未使用时钟、未通电或处于完成复位序列的过程中。
0	R0	RO	0	ADC 模块 0 外设就绪 值 描述 1 ADC 模块 0 已可以访问。 0 ADC 模块 0 尚不可以访问。未使用时钟、未通电或处于完成复位序列的过程中。

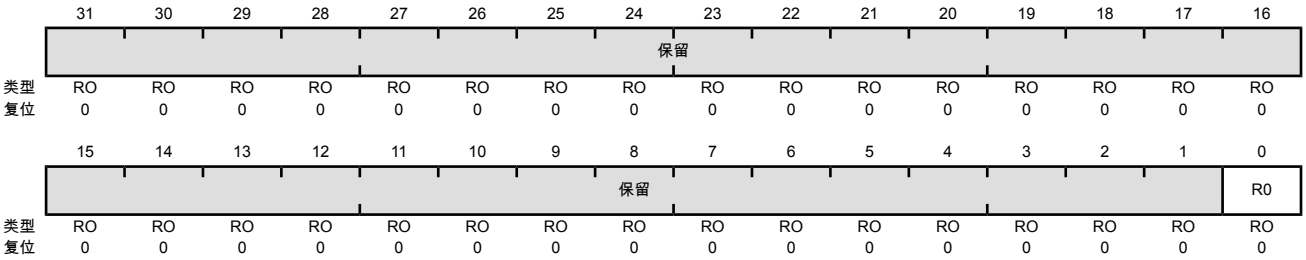
寄存器 104: 模拟比较器外设就绪寄存器（PRACMP），偏移量 0xA3C

PRACMP 寄存器指示模拟比较器模块是否就绪，可由软件在功率状态发生更改、运行模式计时或复位之后访问。如果相应 RCGCACMP 位发生更改，则运行模式计时随之改变。如果相应 SRACMP 位从 0 更改为 1，则复位随之改变。

发生任何上述事件时，PRACMP 位将清零，并且不会再次置位，直到模块完全通电、启用并内部复位。

模拟比较器外设就绪寄存器 (PRACMP)

基址 0x400F.E000
偏移量 0xA3C
类型 RO, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:1	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
0	R0	RO	0	模拟比较器模块 0 外设就绪 值 描述 1 模拟比较器模块已可以访问。 0 模拟比较器模块尚不可以访问。未使用时钟、未通电或处于完成复位序列的过程中。

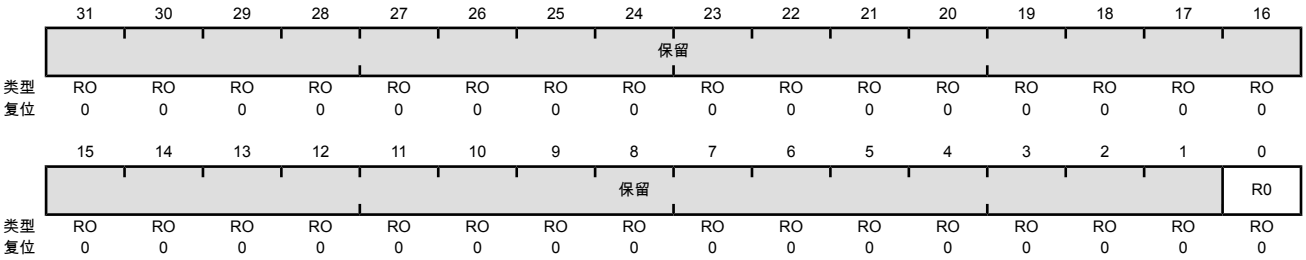
寄存器 105: EEPROM 外设就绪寄存器（PREEPROM），偏移量 0xA58

PREEPROM 寄存器指示 EEPROM 模块是否就绪，可由软件在功率状态发生更改、运行模式计时或复位之后访问。如果相应 RCGCEEPROM 位发生更改，则运行模式计时随之改变。如果相应 SREEPROM 位从 0 更改为 1，则复位随之改变。

发生任何上述事件时，PREEPROM 位将清零，并且不会再次置位，直到模块完全通电、启用并内部复位。

EEPROM 外设就绪寄存器 (PREEPROM)

基址 0x400F.E000
偏移量 0xA58
类型 RO, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:1	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
0	R0	RO	0	EEPROM 模块外设就绪 值 描述 1 EEPROM 模块已可以访问。 0 EEPROM 模块尚不可以访问。未使用时钟、未通电或处于完成复位序列的过程中。

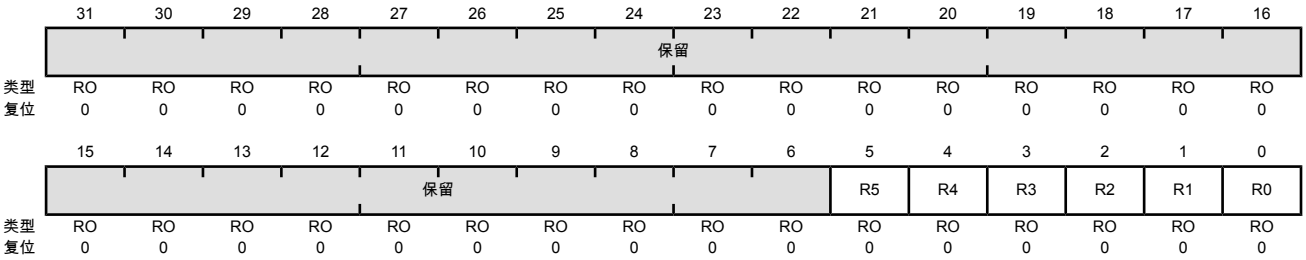
寄存器 106: 32/64 位宽通用定时器外设就绪寄存器（PRWTIMER），偏移量 0xA5C

PRWTIMER 寄存器指示定时器模块是否就绪，可由软件在功率状态发生更改、运行模式计时或复位之后访问。如果相应RCGCWTIMER位发生更改，则运行模式计时随之改变。如果相应SRWTIMER位从 0 更改为 1，则复位随之改变。

发生任何上述事件时，PRWTIMER 位将清零，并且不会再次置位，直到模块完全通电、启用并内部复位。

32/64 位宽通用定时器外设就绪寄存器 (PRWTIMER)

基址 0x400F.E000
偏移量 0xA5C
类型 RO, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:6	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
5	R5	RO	0	32/64 位宽通用定时器 5 外设就绪 值 描述 1 32/64 位宽定时器模块 5 已可以访问。 0 32/64 位宽定时器模块 5 尚不可以访问。未使用时钟、未通电或处于完成复位序列的过程中。
4	R4	RO	0	32/64 位宽通用定时器 4 外设就绪 值 描述 1 32/64 位宽定时器模块 4 已可以访问。 0 32/64 位宽定时器模块 4 尚不可以访问。未使用时钟、未通电或处于完成复位序列的过程中。
3	R3	RO	0	32/64 位宽通用定时器 3 外设就绪 值 描述 1 32/64 位宽定时器模块 3 已可以访问。 0 32/64 位宽定时器模块 3 尚不可以访问。未使用时钟、未通电或处于完成复位序列的过程中。

位/域	名称	类型	复位	描述
2	R2	RO	0	32/64 位宽通用定时器 2 外设就绪 值 描述 1 32/64 位宽定时器模块 2 已可以访问。 0 32/64 位宽定时器模块 2 尚不可以访问。未使用时钟、未通电或处于完成复位序列的过程中。
1	R1	RO	0	32/64 位宽通用定时器 1 外设就绪 值 描述 1 32/64 位宽定时器模块 1 已可以访问。 0 32/64 位宽定时器模块 1 尚不可以访问。未使用时钟、未通电或处于完成复位序列的过程中。
0	R0	RO	0	32/64 位宽通用定时器 0 外设就绪 值 描述 1 32/64 位宽定时器模块 0 已可以访问。 0 32/64 位宽定时器模块 0 尚不可以访问。未使用时钟、未通电或处于完成复位序列的过程中。

5.6 系统控制传统寄存器描述

所有给出的地址都是相对于 0x400F.E000 的系统控制基址而言的。

重要: 本节中的寄存器仅用于实现传统软件支持；应使用“系统控制寄存器描述”（207页）中的寄存器。

寄存器 107: 器件功能寄存器 0（DC0），偏移量 0x008

该传统寄存器由器件预定义且可用于校验特性。

重要: 该寄存器用于实现传统软件支持。

应使用 Flash 大小 (FSIZE) 和 SRAM 大小 (SSIZE) 寄存器确定该微控制器的存储器大小。读 DC0 即可正确识别传统存储器大小，但是软件必须使用 FSIZE 和 SSIZE 来确定以下未列出的存储器的大小。

器件功能寄存器 0 (DC0)

基址 0x400F.E000
偏移量 0x008
类型 RO, 复位 0x007F.007F

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	SRAMSZ															
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	FLASHSZ															
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1

位/域	名称	类型	复位	描述
31:16	SRAMSZ	RO	0x7F	SRAM大小 显示了片上 SRAM 的大小。 值 描述 0x7 2KB SRAM 0xF 4 KB SRAM 0x17 6 KB SRAM 0x1F 8 KB SRAM 0x2F 12 KB SRAM 0x3F 16 KB SRAM 0x4F 20 KB SRAM 0x5F 24 KB SRAM 0x7F 32 KB SRAM

位/域	名称	类型	复位	描述
15:0	FLASHSZ	RO	0x7F	Flash大小 表示片内 Flash 存储器的大小。 值 描述 0x3 8KB Flash 0x7 16 KB Flash 0xF 32 KB Flash 0x1F 64 KB Flash 0x2F 96 KB Flash 0x3F 128 KB Flash 0x5F 192 KB Flash 0x7F 256 KB Flash

寄存器 108: 器件功能寄存器 1（DC1），偏移量 0x010

该寄存器由器件预定义且可用于校验特性。如果该寄存器中的某位是零，那么对应模块不存在。RCGC0、SCGC0、DCGC0 寄存器以及外设专用的 RCGC、SCGC 和 DCGC 寄存器中的对应位不能置位。

重要: 该寄存器用于实现传统软件支持。

应使用外设存在寄存器确定在该微控制器上执行的模块。读 DC1 即可正确识别传统模块是否存在，但是软件必须使用外设存在寄存器来确定不受 DCn 寄存器支持的模块是否存在。

同样，应使用 ADC 外设属性 (ADCPP) 寄存器确定最大 ADC 采样率，以及温度传感器是否存在。但是，要支持传统软件，可使用 MAXADCnSPD 域和 TEMPSNS 位。读 DC1 即可正确识别传统比率的最大 ADC 采样率，以及温度传感器是否存在。

器件功能寄存器 1 (DC1)

基址 0x400F.E000
偏移量 0x010
类型 RO, 复位 0x1103.2FFF

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
域	保留			WDT1	保留		CAN1	CAN0	保留		PWM1	PWM0	保留		ADC1	ADC0
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	1	0	0	0	1	0	0	0	0	0	0	1	1

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
域	MINSYSDIV				MAXADC1SPD		MAXADC0SPD		MPU	HIB	TEMPSNS	PLL	WDT0	SWO	SWD	JTAG
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	1	0	1	1	1	1	1	1	1	1	1	1	1	1

位/域	名称	类型	复位	描述
31:29	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
28	WDT1	RO	0x1	看门狗定时器1存在 置位表示看门狗定时器 1 存在。
27:26	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
25	CAN1	RO	0x0	CAN 模块 1 存在 置位表示 CAN 单元 1 存在。
24	CAN0	RO	0x1	CAN 模块 0 存在 置位表示 CAN 单元 0 存在。
23:22	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
21	PWM1	RO	0x0	PWM 模块 1 存在 置位表示PWM模块存在。
20	PWM0	RO	0x0	PWM 模块 0 存在 置位表示PWM模块存在。
19:18	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。

位/域	名称	类型	复位	描述
17	ADC1	RO	0x1	ADC 模块 1 存在 置位表示 ADC 模块 1 存在。
16	ADC0	RO	0x1	ADC 模块 0 存在 置位表示 ADC 模块 0 存在。
15:12	MINSYSDIV	RO	0x2	系统时钟分频器 用于系统时钟的最小 4 位分频器。复位值取决于硬件。查看 RCC 寄存器可获知如何使用 SYSDIV 位来改变系统时钟。 值 描述 0x1 指定一个 PLL 2.5 分频的 80-MHz CPU 时钟。 0x2 指定一个 PLL 3 分频的 66-MHz CPU 时钟。 0x3 指定一个 PLL 4 分频的 50-MHz CPU 时钟。 0x4 指定一个 PLL 5 分频的 40-MHz CPU 时钟。 0x7 指定一个 PLL 8 分频的 25-MHz 时钟。 0x9 指定一个 PLL 10 分频的 20-MHz 时钟。
11:10	MAXADC1SPD	RO	0x3	ADC1 最大速率 该域规定了 ADC 采样数据的最大速率。 值 描述 0x3 1M 采样/秒 0x2 500K 采样/秒 0x1 250K 采样/秒 0x0 125K 采样/秒
9:8	MAXADC0SPD	RO	0x3	ADC0最快速度 该域规定了 ADC 采样数据的最大速率。 值 描述 0x3 1M 采样/秒 0x2 500K 采样/秒 0x1 250K 采样/秒 0x0 125K 采样/秒
7	MPU	RO	0x1	MPU存在 置位表示 Cortex-M4F 存储器保护单元 (MPU) 模块存在。关于 MPU 的详细信息可查看“Cortex-M4F 外设”一章。
6	HIB	RO	0x1	休眠模块存在 置位表示休眠模块存在。
5	TEMPSNS	RO	0x1	温度传感器存在标志。 置位表示片上温度传感器存在。
4	PLL	RO	0x1	PLL存在标志。 置位表示片上锁相环 (PLL) 存在。

位/域	名称	类型	复位	描述
3	WDT0	RO	0x1	看门狗定时器 0 存在 置位表示看门狗定时器 0 存在。
2	SWO	RO	0x1	SWO跟踪端口存在 置位表示串行线输出 (SWO) 跟踪端口存在。
1	SWD	RO	0x1	SWD存在标志 置位表示串行线调试器 (SWD) 存在。
0	JTAG	RO	0x1	JTAG存在标志 置位表示 JTAG 调试器接口存在。

寄存器 109: 器件功能寄存器 2（DC2），偏移量 0x014

该寄存器由器件预定义且可用于校验特性。如果该寄存器中的某位是零，那么对应模块不存在。RCGC1、SCGC1、DCGC1 寄存器以及外设专用的 RCGC、SCGC 和 DCGC 寄存器中的对应位不能置位。

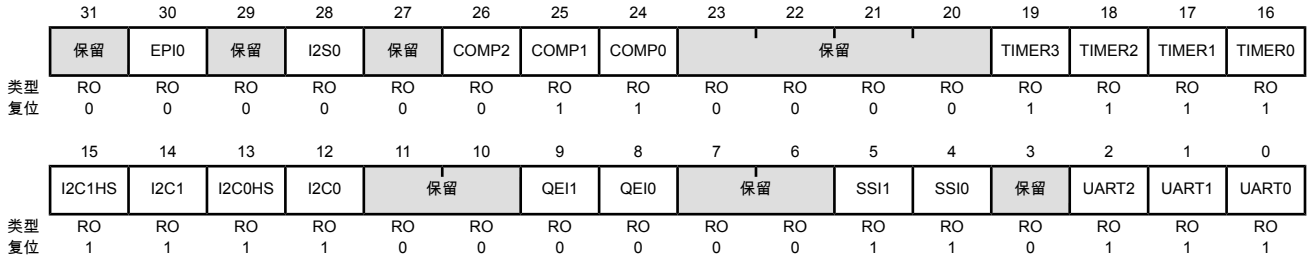
重要: 该寄存器用于实现传统软件支持。

应使用外设存在寄存器确定在该微控制器上执行的模块。读 DC2 即可正确识别传统模块是否存在，但是软件必须使用外设存在寄存器来确定不受 DCn 寄存器支持的模块是否存在。

请注意：模拟比较器外设存在 (PPACMP) 寄存器可识别模拟比较器模块是否存在。模拟比较器外设属性 (ACMPPP) 寄存器可指示模块中存在的模拟比较器块数量。

器件功能寄存器 2 (DC2)

基址 0x400F.E000
偏移量 0x014
类型 RO, 复位 0x030F.F037



位/域	名称	类型	复位	描述
31	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
30	EPI0	RO	0x0	EPI模块0存在 置位表示EPI模块0存在。
29	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
28	I2S0	RO	0x0	I2S模块0存在 置位表示I2S模块0存在。
27	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
26	COMP2	RO	0x0	模拟比较器2存在 置位表示模拟比较器2存在。
25	COMP1	RO	0x1	模拟比较器1存在 置位表示模拟比较器1存在。
24	COMP0	RO	0x1	模拟比较器0存在 置位表示模拟比较器0存在。
23:20	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。

位/域	名称	类型	复位	描述
19	TIMER3	RO	0x1	定时器模块3存在 置位表示通用定时器模块3存在。
18	TIMER2	RO	0x1	定时器模块2存在 置位表示通用定时器模块2存在。
17	TIMER1	RO	0x1	定时器模块1存在 置位表示通用定时器模块1存在。
16	TIMER0	RO	0x1	定时器模块0存在 置位表示通用定时器模块0存在。
15	I2C1HS	RO	0x1	I2C模块1速度 置位表示 I2C 模块 1 可以在高速模式中操作。
14	I2C1	RO	0x1	I2C模块1存在 置位表示I2C模块1存在。
13	I2C0HS	RO	0x1	I2C模块0速度 置位表示 I2C 模块 0 可以在高速模式中操作。
12	I2C0	RO	0x1	I2C模块0存在 置位表示I2C模块0存在。
11:10	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
9	QEI1	RO	0x0	QEI模块1存在 置位表示QEI模块1存在。
8	QEI0	RO	0x0	QEI模块0存在 置位表示QEI模块0存在。
7:6	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
5	SSI1	RO	0x1	SSI模块1存在 置位表示SSI模块1存在。
4	SSI0	RO	0x1	SSI模块0存在 置位表示SSI模块0存在。
3	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
2	UART2	RO	0x1	UART模块2存在 置位表示UART模块2存在。
1	UART1	RO	0x1	UART模块1存在 置位表示UART模块1存在。
0	UART0	RO	0x1	UART模块0存在 置位表示UART模块0存在。

寄存器 110: 器件功能寄存器 3（DC3），偏移量 0x018

该寄存器由器件预定义且可用于校验特性。如果该寄存器中的某位是零，那么对应功能不存在。

重要: 该寄存器用于实现传统软件支持。

对于一些模块来说，应使用外设驻留外设属性寄存器确定在该微控制器上可用的管脚。读 DC3 即可正确识别传统管脚是否存在，但是软件必须使用外设属性寄存器来确定不受 DCn 寄存器支持的管脚是否存在。

器件功能寄存器 3 (DC3)

基址 0x400F.E000
偏移量 0x018
类型 RO, 复位 0xBFFF.0FC0

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	32KHZ	保留	CCP5	CCP4	CCP3	CCP2	CCP1	CCP0	ADC0AIN7	ADC0AIN6	ADC0AIN5	ADC0AIN4	ADC0AIN3	ADC0AIN2	ADC0AIN1	ADC0AIN0
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	PWMFAULT	C2O	C2PLUS	C2MINUS	C1O	C1PLUS	C1MINUS	C0O	C0PLUS	C0MINUS	PWM5	PWM4	PWM3	PWM2	PWM1	PWM0
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	1	1	1	1	1	1	0	0	0	0	0	0

位/域	名称	类型	复位	描述
31	32KHZ	RO	0x1	32KHz输入时钟可用 置位表示一个偶数的CCP管脚存在并且可用作一个32KHz输入时钟。 注意: GPTMPP 寄存器不提供该信息。
30	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
29	CCP5	RO	0x1	T2CCP1 管脚存在 置位时表示存在捕获/比较/PWM 管脚 T2CCP1。 注意: GPTMPP 寄存器不提供该信息。
28	CCP4	RO	0x1	T2CCP0 管脚存在 置位时表示存在捕获/比较/PWM 管脚 T2CCP0。 注意: GPTMPP 寄存器不提供该信息。
27	CCP3	RO	0x1	T1CCP1 管脚存在 置位时表示存在捕获/比较/PWM 管脚 T1CCP1。 注意: GPTMPP 寄存器不提供该信息。
26	CCP2	RO	0x1	T1CCP0 管脚存在 置位时表示存在捕获/比较/PWM 管脚 T1CCP0。 注意: GPTMPP 寄存器不提供该信息。
25	CCP1	RO	0x1	T0CCP1 管脚存在 置位时表示存在捕获/比较/PWM 管脚 T0CCP1。 注意: GPTMPP 寄存器不提供该信息。

位/域	名称	类型	复位	描述
24	CCP0	RO	0x1	T0CCP0 管脚存在 置位时表示存在捕获/比较/PWM 管脚 T0CCP0。 注意: GPTMPP 寄存器不提供该信息。
23	ADC0AIN7	RO	0x1	ADC 模块0 AIN7管脚存在 置位表示ADC模块0的输入管脚7存在。 注意: ADCPP 寄存器中的 CH 域提供该信息。
22	ADC0AIN6	RO	0x1	ADC 模块0 AIN6管脚存在 置位表示ADC模块0的输入管脚6存在。 注意: ADCPP 寄存器中的 CH 域提供该信息。
21	ADC0AIN5	RO	0x1	ADC 模块0 AIN5管脚存在 置位表示ADC模块0的输入管脚5存在。 注意: ADCPP 寄存器中的 CH 域提供该信息。
20	ADC0AIN4	RO	0x1	ADC 模块0 AIN4管脚存在 置位表示ADC模块0的输入管脚4存在。 注意: ADCPP 寄存器中的 CH 域提供该信息。
19	ADC0AIN3	RO	0x1	ADC 模块0 AIN3管脚存在 置位表示ADC模块0的输入管脚3存在。 注意: ADCPP 寄存器中的 CH 域提供该信息。
18	ADC0AIN2	RO	0x1	ADC 模块0 AIN2管脚存在 置位表示ADC模块0的输入管脚2存在。 注意: ADCPP 寄存器中的 CH 域提供该信息。
17	ADC0AIN1	RO	0x1	ADC 模块0 AIN1管脚存在 置位表示ADC模块0的输入管脚1存在。 注意: ADCPP 寄存器中的 CH 域提供该信息。
16	ADC0AIN0	RO	0x1	ADC 模块0 AIN0管脚存在 置位表示ADC模块0的输入管脚0存在。 注意: ADCPP 寄存器中的 CH 域提供该信息。
15	PWMFAULT	RO	0x0	PWM 故障管脚存在 置位表示 PWM 故障管脚存在。查看 DC5 可了解该器件的特定错误管脚。 注意: PWMPP 寄存器中的 FCNT 域提供该信息。
14	C2O	RO	0x0	C2o管脚存在 置位表示模拟比较器 2 输出管脚存在。 注意: ACMPPP 寄存器中的 C2O 位提供该信息。
13	C2PLUS	RO	0x0	C2+ 管脚存在标志 置位表示模拟比较器 2(+) 输入管脚存在。 注意: 模拟比较器 2 存在时管脚存在。

位/域	名称	类型	复位	描述
12	C2MINUS	RO	0x0	C2-管脚存在标志 置位表示模拟比较器 2(-) 输入管脚存在。 注意: 模拟比较器 2 存在时管脚存在。
11	C1O	RO	0x1	C1o管脚存在 置位表示模拟比较器 1 输出管脚存在。 注意: ACMPPP 寄存器中的 C1O 位提供该信息。
10	C1PLUS	RO	0x1	C1+管脚存在标志 置位表示模拟比较器 1(+) 输入管脚存在。 注意: 模拟比较器 1 存在时管脚存在。
9	C1MINUS	RO	0x1	C1-管脚存在标志 置位表示模拟比较器 1(-) 输入管脚存在。 注意: 模拟比较器 1 存在时管脚存在。
8	C0O	RO	0x1	C0o管脚存在 置位表示模拟比较器 0 输出管脚存在。 注意: ACMPPP 寄存器中的 C0O 位提供该信息。
7	C0PLUS	RO	0x1	C0+管脚存在标志 置位表示模拟比较器 0(+) 输入管脚存在。 注意: 模拟比较器 0 存在时管脚存在。
6	C0MINUS	RO	0x1	C0-管脚存在标志 置位表示模拟比较器 0(-) 输入管脚存在。 注意: 模拟比较器 0 存在时管脚存在。
5	PWM5	RO	0x0	PWM5管脚存在 置位表示PWM管脚5存在。 注意: PWMPP 寄存器中的 GCNT 域提供该信息。
4	PWM4	RO	0x0	PWM4管脚存在 置位表示PWM管脚4存在。 注意: PWMPP 寄存器中的 GCNT 域提供该信息。
3	PWM3	RO	0x0	PWM3管脚存在 置位表示PWM管脚3存在。 注意: PWMPP 寄存器中的 GCNT 域提供该信息。
2	PWM2	RO	0x0	PWM2管脚存在 置位表示PWM管脚2存在。 注意: PWMPP 寄存器中的 GCNT 域提供该信息。
1	PWM1	RO	0x0	PWM1管脚存在 置位表示PWM管脚1存在。 注意: PWMPP 寄存器中的 GCNT 域提供该信息。

位/域	名称	类型	复位	描述
0	PWM0	RO	0x0	PWM0管脚存在 置位表示PWM管脚0存在。 注意: PWMPP 寄存器中的 GCNT 域提供该信息。

寄存器 111: 器件功能寄存器 4（DC4），偏移量 0x01C

该寄存器由器件预定义且可用于校验特性。如果该寄存器中的某位是零，那么对应模块不存在。RCGC2、SCGC2、DCGC2 寄存器以及外设专用的 RCGC、SCGC 和 DCGC 寄存器中的对应位不能置位。

重要: 该寄存器用于实现传统软件支持。

应使用外设存在寄存器确定在该微控制器上执行的模块。读 DC4 即可正确识别传统模块是否存在，但是软件必须使用外设存在寄存器来确定不受 DCn 寄存器支持的模块是否存在。

应使用外设驻留外设属性寄存器确定在该微控制器上可用的管脚和功能。读取 DC4 即可正确识别传统管脚或功能是否存在。软件必须使用外设属性寄存器来确定不受 DCn 寄存器支持的管脚或功能是否存在。

器件功能寄存器 4 (DC4)

基址 0x400F.E000
偏移量 0x01C
类型 RO, 复位 0x0004.F03F

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	保留	EPHY0	保留	EMAC0	保留			E1588	保留					PICAL	保留	
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	CCP7	CCP6	UDMA	ROM	保留			GPIOJ	GPIOH	GPIOG	GPIOF	GPIOE	GPIOD	GPIOC	GPIOB	GPIOA
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	1	1	1	1	0	0	0	0	0	0	1	1	1	1	1	1

位/域	名称	类型	复位	描述
31	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
30	EPHY0	RO	0x0	以太网 PHY 层 0 存在 置位表示以太网 PHY 层 0 存在。
29	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
28	EMAC0	RO	0x0	以太网 MAC 层 0 存在 置位表示以太网 MAC 层 0 存在。
27:25	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
24	E1588	RO	0x0	1588 可用 置位表示以太网 MAC 层 0 是 1588 可用。
23:19	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
18	PICAL	RO	0x1	PIOSC 规格 置位表示可以通过软件校准 PIOSC。
17:16	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。

位/域	名称	类型	复位	描述
15	CCP7	RO	0x1	T3CCP1 管脚存在 置位时表示存在捕获/比较/PWM 管脚 T3CCP1。 注意: GPTMPP 寄存器不提供该信息。
14	CCP6	RO	0x1	T3CCP0 管脚存在 置位时表示存在捕获/比较/PWM 管脚 T3CCP0。 注意: GPTMPP 寄存器不提供该信息。
13	UDMA	RO	0x1	微 DMA 模块存在 置位表示微 DMA 模块存在。
12	ROM	RO	0x1	内部 ROM 代码存在 置位表示内部代码ROM存在。
11:9	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
8	GPIOJ	RO	0x0	GPIO 端口 J 存在 置位表示GPIO端口J存在。
7	GPIOH	RO	0x0	GPIO 端口 H 存在 置位表示 GPIO 端口 H 存在。
6	GPIOG	RO	0x0	GPIO 端口 G 存在 置位表示 GPIO 端口 G 存在。
5	GPIOF	RO	0x1	GPIO 端口 F 存在 置位表示 GPIO 端口 F 存在。
4	GPIOE	RO	0x1	GPIO端口E存在 置位表示GPIO端口E存在。
3	GPIOD	RO	0x1	GPIO端口D存在 置位表示GPIO端口D存在。
2	GPIOC	RO	0x1	GPIO端口C存在 置位表示GPIO端口C存在。
1	GPIOB	RO	0x1	GPIO端口B存在 置位表示GPIO端口B存在。
0	GPIOA	RO	0x1	GPIO端口A存在 置位表示GPIO端口A存在。

寄存器 112: 器件功能寄存器 5（DC5），偏移量 0x020

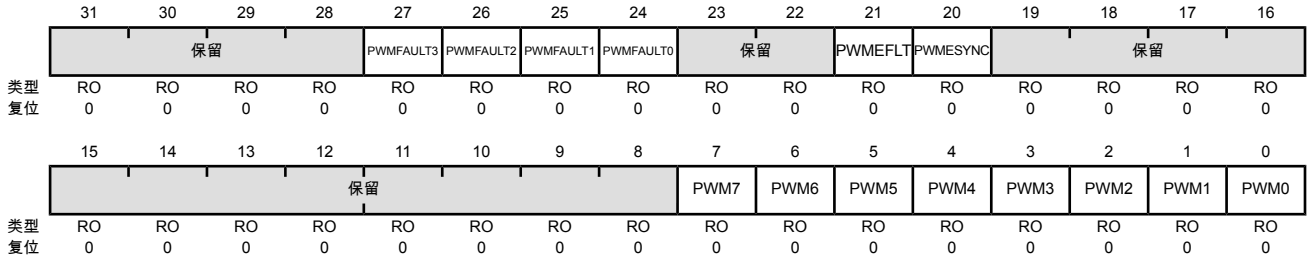
该寄存器由器件预定义且可用于校验 PWM 特性。如果该寄存器中的某位是零，那么对应模块不存在。

重要: 该寄存器用于实现传统软件支持。

应使用 PWM 外设属性 (PWMPP) 寄存器确定在 PWM 模块上可用的管脚和功能。读该寄存器即可正确识别传统管脚或功能是否存在。软件必须使用 PWMPP 寄存器来确定不受 DCn 寄存器支持的管脚或功能是否存在。

器件功能寄存器 5 (DC5)

基址 0x400F.E000
偏移量 0x020
类型 RO, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:28	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
27	PWMFAULT3	RO	0x0	PWM 错误 3 管脚存在 置位表示 PWM 错误 3 管脚存在。
26	PWMFAULT2	RO	0x0	PWM 错误 2 管脚存在 置位表示 PWM 错误 2 管脚存在。
25	PWMFAULT1	RO	0x0	PWM 错误 1 管脚存在 置位表示 PWM 错误 1 管脚存在。
24	PWMFAULT0	RO	0x0	PWM 错误 0 管脚存在 置位表示 PWM 错误 0 管脚存在。
23:22	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
21	PWMEFLT	RO	0x0	PWM 扩展错误激活 置位表示 PWM 扩展错误特性是激活的。
20	PWMESYNC	RO	0x0	PWM 扩展 SYNC 激活 置位表示 PWM 扩展 SYNC 特性是激活的。
19:8	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
7	PWM7	RO	0x0	PWM7 管脚存在 置位表示PWM管脚7存在。

位/域	名称	类型	复位	描述
6	PWM6	RO	0x0	PWM6 管脚存在 置位表示PWM管脚6存在。
5	PWM5	RO	0x0	PWM5管脚存在 置位表示PWM管脚5存在。
4	PWM4	RO	0x0	PWM4管脚存在 置位表示PWM管脚4存在。
3	PWM3	RO	0x0	PWM3管脚存在 置位表示PWM管脚3存在。
2	PWM2	RO	0x0	PWM2管脚存在 置位表示PWM管脚2存在。
1	PWM1	RO	0x0	PWM1管脚存在 置位表示PWM管脚1存在。
0	PWM0	RO	0x0	PWM0管脚存在 置位表示PWM管脚0存在。

寄存器 113: 器件功能寄存器 6（DC6），偏移量 0x024

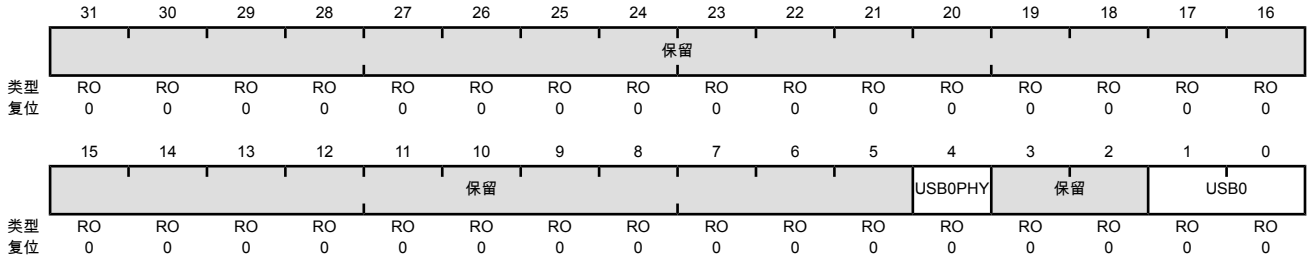
该寄存器由器件预定义且可用于校验特性。如果该寄存器中的某位是零，那么对应模块不存在。RCGC0、SCGC0 和 DCGC0 寄存器中的对应位不能置位。

重要: 该寄存器用于实现传统软件支持。

应使用 USB 外设属性 (USBPP) 寄存器确定在 USB 模块上可用的功能。读该寄存器即可正确识别传统功能是否存在。软件必须使用 USBPP 寄存器来确定不受 DCn 寄存器支持的管脚或功能是否存在。

器件功能寄存器 6 (DC6)

基址 0x400F.E000
偏移量 0x024
类型 RO, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:5	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
4	USB0PHY	RO	0x0	USB 模块 0 PHY 存在 置位表示 USB 模块 0 PHY 存在。
3:2	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
1:0	USB0	RO	0x0	USB 模块0存在 该域表示 USB 模块 0 存在并规定了它的功能。 sysValue 描述 0x0 NA USB0 不存在。 0x1 DEVICE USB0 仅是器件。 0x2 HOST USB0 是器件或主机。 0x3 OTG USB0 是 OTG。

寄存器 114: 器件功能寄存器 7（DC7），偏移量 0x028

该寄存器由器件预定义且可用于校验 µDMA 通道特性。值为 1 表示该通道在这个器件上可用，值为 0 表示该通道只在这个系列的其它器件上可用。通道可以拥有多个赋值，请参考“通道分配”（513页）了解更多信息。

重要: 该寄存器用于实现传统软件支持。DMA 状态 (DMASTAT) 寄存器的 DMACHANS 位域表示 DMA 通道的数量。

器件功能寄存器 7 (DC7)

基址 0x400F.E000
偏移量 0x028
类型 RO, 复位 0xFFFF.FFFF

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
域	保留	DMACH30	DMACH29	DMACH28	DMACH27	DMACH26	DMACH25	DMACH24	DMACH23	DMACH22	DMACH21	DMACH20	DMACH19	DMACH18	DMACH17	DMACH16
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
域	DMACH15	DMACH14	DMACH13	DMACH12	DMACH11	DMACH10	DMACH9	DMACH8	DMACH7	DMACH6	DMACH5	DMACH4	DMACH3	DMACH2	DMACH1	DMACH0
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

位/域	名称	类型	复位	描述
31	保留	RO	0x1	DMA 通道 31 置位表示 µDMA 通道 31 可用。
30	DMACH30	RO	0x1	DMA 通道 30 置位表示 µDMA 通道 30 可用。
29	DMACH29	RO	0x1	DMA 通道 29 置位表示 µDMA 通道 29 可用。
28	DMACH28	RO	0x1	DMA 通道 28 置位表示 µDMA 通道 28 可用。
27	DMACH27	RO	0x1	DMA 通道 27 置位表示 µDMA 通道 27 可用。
26	DMACH26	RO	0x1	DMA 通道 26 置位表示 µDMA 通道 26 可用。
25	DMACH25	RO	0x1	DMA 通道 25 置位表示 µDMA 通道 25 可用。
24	DMACH24	RO	0x1	DMA 通道 24 置位表示 µDMA 通道 24 可用。
23	DMACH23	RO	0x1	DMA 通道 23 置位表示 µDMA 通道 23 可用。
22	DMACH22	RO	0x1	DMA 通道 22 置位表示 µDMA 通道 22 可用。

位/域	名称	类型	复位	描述
21	DMACH21	RO	0x1	DMA 通道 21 置位表示 µDMA 通道 21 可用。
20	DMACH20	RO	0x1	DMA 通道 20 置位表示 µDMA 通道 20 可用。
19	DMACH19	RO	0x1	DMA 通道 19 置位表示 µDMA 通道 19 可用。
18	DMACH18	RO	0x1	DMA 通道 18 置位表示 µDMA 通道 18 可用。
17	DMACH17	RO	0x1	DMA 通道 17 置位表示 µDMA 通道 17 可用。
16	DMACH16	RO	0x1	DMA 通道 16 置位表示 µDMA 通道 16 可用。
15	DMACH15	RO	0x1	DMA 通道 15 置位表示 µDMA 通道 15 可用。
14	DMACH14	RO	0x1	DMA 通道 14 置位表示 µDMA 通道 14 可用。
13	DMACH13	RO	0x1	DMA 通道 13 置位表示 µDMA 通道 13 可用。
12	DMACH12	RO	0x1	DMA 通道 12 置位表示 µDMA 通道 12 可用。
11	DMACH11	RO	0x1	DMA 通道 11 置位表示 µDMA 通道 11 可用。
10	DMACH10	RO	0x1	DMA 通道 10 置位表示 µDMA 通道 10 可用。
9	DMACH9	RO	0x1	DMA 通道 9 置位表示 µDMA 通道 9 可用。
8	DMACH8	RO	0x1	DMA 通道 8 置位表示 µDMA 通道 8 可用。
7	DMACH7	RO	0x1	DMA 通道 7 置位表示 µDMA 通道 7 可用。
6	DMACH6	RO	0x1	DMA 通道 6 置位表示 µDMA 通道 6 可用。
5	DMACH5	RO	0x1	DMA 通道 5 置位表示 µDMA 通道 5 可用。
4	DMACH4	RO	0x1	DMA 通道 4 置位表示 µDMA 通道 4 可用。

位/域	名称	类型	复位	描述
3	DMACH3	RO	0x1	DMA 通道 3 置位表示 µDMA 通道 3 可用。
2	DMACH2	RO	0x1	DMA 通道 2 置位表示 µDMA 通道 2 可用。
1	DMACH1	RO	0x1	DMA 通道 1 置位表示 µDMA 通道 1 可用。
0	DMACH0	RO	0x1	DMA 通道 0 置位表示 µDMA 通道 0 可用。

寄存器 115: 器件功能寄存器 8 （DC8），偏移量 0x02C

该寄存器由器件预定义且可用于校验特性。

重要: 该寄存器用于实现传统软件支持。

应使用 ADC 外设属性 (ADCPP) 寄存器确定在 ADC 模块上可用的输入通道数量。读该寄存器即可正确识别传统通道是否存在，但是软件必须使用 ADCPP 寄存器来确定不受 DCn 寄存器支持的通道是否存在。

器件功能寄存器 8 (DC8)

基址 0x400F.E000
偏移量 0x02C
类型 RO, 复位 0x0FFF.0FFF

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ADC1AIN15	ADC1AIN14	ADC1AIN13	ADC1AIN12	ADC1AIN11	ADC1AIN10	ADC1AIN9	ADC1AIN8	ADC1AIN7	ADC1AIN6	ADC1AIN5	ADC1AIN4	ADC1AIN3	ADC1AIN2	ADC1AIN1	ADC1AIN0
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ADC0AIN15	ADC0AIN14	ADC0AIN13	ADC0AIN12	ADC0AIN11	ADC0AIN10	ADC0AIN9	ADC0AIN8	ADC0AIN7	ADC0AIN6	ADC0AIN5	ADC0AIN4	ADC0AIN3	ADC0AIN2	ADC0AIN1	ADC0AIN0
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1

位/域	名称	类型	复位	描述
31	ADC1AIN15	RO	0x0	ADC 模块 1 AIN15 管脚存在 置位表示ADC模块1的输入管脚15存在。
30	ADC1AIN14	RO	0x0	ADC 模块1 AIN14 管脚存在 置位表示ADC模块1的输入管脚14存在。
29	ADC1AIN13	RO	0x0	ADC 模块1 AIN13 管脚存在 置位表示ADC模块1的输入管脚13存在。
28	ADC1AIN12	RO	0x0	ADC 模块1 AIN12 管脚存在 置位表示ADC模块1的输入管脚12存在。
27	ADC1AIN11	RO	0x1	ADC 模块1 AIN11 管脚存在 置位表示ADC模块1的输入管脚11存在。
26	ADC1AIN10	RO	0x1	ADC 模块1 AIN10 管脚存在 置位表示ADC模块1的输入管脚10存在。
25	ADC1AIN9	RO	0x1	ADC 模块 1 AIN9 管脚存在 置位表示ADC模块1的输入管脚9存在。
24	ADC1AIN8	RO	0x1	ADC 模块 1 AIN8 管脚存在 置位表示ADC模块1的输入管脚8存在。
23	ADC1AIN7	RO	0x1	ADC 模块1 AIN7管脚存在 置位表示ADC模块1的输入管脚7存在。
22	ADC1AIN6	RO	0x1	ADC 模块1 AIN6管脚存在 置位表示ADC模块1的输入管脚6存在。

位/域	名称	类型	复位	描述
21	ADC1AIN5	RO	0x1	ADC 模块1 AIN5管脚存在 置位表示ADC模块1的输入管脚5存在。
20	ADC1AIN4	RO	0x1	ADC 模块1 AIN4管脚存在 置位表示ADC模块1的输入管脚4存在。
19	ADC1AIN3	RO	0x1	ADC 模块1 AIN3管脚存在 置位表示ADC模块1的输入管脚3存在。
18	ADC1AIN2	RO	0x1	ADC 模块1 AIN2管脚存在 置位表示ADC模块1的输入管脚2存在。
17	ADC1AIN1	RO	0x1	ADC 模块1 AIN1管脚存在 置位表示ADC模块1的输入管脚1存在。
16	ADC1AIN0	RO	0x1	ADC 模块1 AIN0管脚存在 置位表示ADC模块1的输入管脚0存在。
15	ADC0AIN15	RO	0x0	ADC 模块 0 AIN15 管脚存在 置位表示ADC模块0的输入管脚15存在。
14	ADC0AIN14	RO	0x0	ADC 模块0 AIN14 管脚存在 置位表示ADC模块0的输入管脚14存在。
13	ADC0AIN13	RO	0x0	ADC 模块0 AIN13 管脚存在 置位表示ADC模块0的输入管脚13存在。
12	ADC0AIN12	RO	0x0	ADC 模块0 AIN12 管脚存在 置位表示ADC模块0的输入管脚12存在。
11	ADC0AIN11	RO	0x1	ADC 模块0 AIN11 管脚存在 置位表示ADC模块0的输入管脚11存在。
10	ADC0AIN10	RO	0x1	ADC 模块0 AIN10 管脚存在 置位表示ADC模块0的输入管脚10存在。
9	ADC0AIN9	RO	0x1	ADC 模块 0 AIN9 管脚存在 置位表示ADC模块0的输入管脚9存在。
8	ADC0AIN8	RO	0x1	ADC 模块 0 AIN8 管脚存在 置位表示ADC模块0的输入管脚8存在。
7	ADC0AIN7	RO	0x1	ADC 模块0 AIN7管脚存在 置位表示ADC模块0的输入管脚7存在。
6	ADC0AIN6	RO	0x1	ADC 模块0 AIN6管脚存在 置位表示ADC模块0的输入管脚6存在。
5	ADC0AIN5	RO	0x1	ADC 模块0 AIN5管脚存在 置位表示ADC模块0的输入管脚5存在。
4	ADC0AIN4	RO	0x1	ADC 模块0 AIN4管脚存在 置位表示ADC模块0的输入管脚4存在。

位/域	名称	类型	复位	描述
3	ADC0AIN3	RO	0x1	ADC 模块0 AIN3管脚存在 置位表示ADC模块0的输入管脚3存在。
2	ADC0AIN2	RO	0x1	ADC 模块0 AIN2管脚存在 置位表示ADC模块0的输入管脚2存在。
1	ADC0AIN1	RO	0x1	ADC 模块0 AIN1管脚存在 置位表示ADC模块0的输入管脚1存在。
0	ADC0AIN0	RO	0x1	ADC 模块0 AIN0管脚存在 置位表示ADC模块0的输入管脚0存在。

寄存器 116: 软件复位控制寄存器 0（SRCR0），偏移量 0x040

该寄存器允许模块被单独复位。写入该寄存器的值被器件功能 1 (DC1) 寄存器的位屏蔽。

重要: 该寄存器用于实现传统软件支持。

应将外设专用软件复位寄存器（如 SRWD）用于复位特定的外设。写该传统寄存器的同时，也将写外设专用寄存器中的相应位。通过对该传统寄存器的写操作更改的任何位都可以通过对该寄存器的读操作进行正确回读。软件必须使用外设专用寄存器以支持不处于传统寄存器中的模块。如果软件使用外设专用寄存器对传统外设（如看门狗 1）执行写操作，则写操作会产生正确操作，但是该位的值不在该寄存器中得到反映。如果软件使用传统和外设专用寄存器访问，则必须通过读-修改-写的操作来访问外设专用寄存器，因为该操作仅影响不在传统寄存器中的外设。通过这种方法，外设专用和传统寄存器都具有一致的信息。

软件复位控制寄存器 0 (SRCR0)

基址 0x400F.E000
偏移量 0x040
类型 RO, 复位 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	保留			WDT1	保留			CAN0	保留						ADC1	ADC0
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	保留									HIB	保留		WDT0	保留		
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位/域	名称	类型	复位	描述
31:29	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
28	WDT1	RO	0x0	WDT1 复位控制 当该位置位时，看门狗定时器模块 1 复位。所有内部数据丢失，同时寄存器回到复位状态。该位在置位后必须手动清零。
27:25	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
24	CAN0	RO	0x0	CAN0复位控制 当该位置位时，CAN 模块 0 复位。所有内部数据丢失，同时寄存器回到复位状态。该位在置位后必须手动清零。
23:18	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
17	ADC1	RO	0x0	ADC1复位控制 当该位置位时，ADC模块1复位。所有内部数据丢失，同时寄存器回到复位状态。该位在置位后必须手动清零。
16	ADC0	RO	0x0	ADC0复位控制 当该位置位时，ADC模块0复位。所有内部数据丢失，同时寄存器回到复位状态。该位在置位后必须手动清零。
15:7	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。

位/域	名称	类型	复位	描述
6	HIB	RO	0x0	HIB 复位控制 当该位置位时，休眠模块复位。所有内部数据丢失，同时寄存器回到复位状态。该位在置位后必须手动清零。
5:4	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
3	WDT0	RO	0x0	WDT0复位控制 当该位置位时，看门狗定时器模块 0 复位。所有内部数据丢失，同时寄存器回到复位状态。该位在置位后必须手动清零。
2:0	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。

寄存器 117: 软件复位控制寄存器 1（SRCR1），偏移量 0x044

该寄存器允许模块被单独复位。写入该寄存器的值被器件功能 2 (DC2) 寄存器的位屏蔽。

重要: 该寄存器用于实现传统软件支持。

应将外设专用软件复位寄存器（如 SRTIMER）用于复位特定的外设。写该寄存器的同时，也将写外设专用寄存器中的相应位。通过对该寄存器的写操作更改的任何位都可以通过对该寄存器的读操作进行正确回读。软件必须使用外设专用寄存器以支持不处于传统寄存器中的模块。如果软件使用外设专用寄存器对传统外设（如TIMER0）执行写操作，则写操作会产生正确操作，但是该位的值不在该寄存器中得到反映。如果软件使用传统和外设专用寄存器访问，则必须通过读-修改-写的操作来访问外设专用寄存器，因为该操作仅影响不在传统寄存器中的外设。通过这种方法，外设专用和传统寄存器都具有一致的信息。

请注意：软件复位模拟比较器 (SRACMP) 寄存器仅有一个位来设置模拟比较器模块。复位该模块将会复位所有块。如果任何COMPn位置位，则整个模拟比较器复位。不能单独复位各模块。

软件复位控制寄存器 1 (SRCR1)

基址 0x400F.E000
偏移量 0x044
类型 RO, 复位 0x0000.0000

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
域	保留						COMP1	COMP0	保留				TIMER3	TIMER2	TIMER1	TIMER0
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
域	保留	I2C1	保留	I2C0	保留						SSI1	SSI0	保留	UART2	UART1	UART0
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位/域	名称	类型	复位	描述
31:26	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
25	COMP1	RO	0x0	模拟比较器1复位控制 当该位置位时，模拟比较器模块1复位。所有内部数据丢失，同时寄存器回到复位状态。该位在置位后必须手动清零。
24	COMP0	RO	0x0	模拟比较器0复位控制 当该位置位时，模拟比较器模块0复位。所有内部数据丢失，同时寄存器回到复位状态。该位在置位后必须手动清零。
23:20	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
19	TIMER3	RO	0x0	定时器3复位控制 定时器3复位控制。当该位置位时，通用定时器模块3复位。所有内部数据丢失，同时寄存器回到复位状态。该位在置位后必须手动清零。
18	TIMER2	RO	0x0	定时器2复位控制 当该位置位时，通用定时器模块2复位。所有内部数据丢失，同时寄存器回到复位状态。该位在置位后必须手动清零。

位/域	名称	类型	复位	描述
17	TIMER1	RO	0x0	定时器1复位控制 当该位置位时，通用定时器模块1复位。所有内部数据丢失，同时寄存器回到复位状态。该位在置位后必须手动清零。
16	TIMER0	RO	0x0	定时器0复位控制 当该位置位时，通用定时器模块0复位。所有内部数据丢失，同时寄存器回到复位状态。该位在置位后必须手动清零。
15	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
14	I2C1	RO	0x0	I2C1复位控制 当该位置位时，I2C模块1复位。所有内部数据丢失，同时寄存器回到复位状态。该位在置位后必须手动清零。
13	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
12	I2C0	RO	0x0	I2C0复位控制 当该位置位时，I2C模块0复位。所有内部数据丢失，同时寄存器回到复位状态。该位在置位后必须手动清零。
11:6	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
5	SSI1	RO	0x0	SSI1复位控制 当该位置位时，SSI模块1复位。所有内部数据丢失，同时寄存器回到复位状态。该位在置位后必须手动清零。
4	SSI0	RO	0x0	SSI0复位控制 当该位置位时，SSI模块0复位。所有内部数据丢失，同时寄存器回到复位状态。该位在置位后必须手动清零。
3	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
2	UART2	RO	0x0	UART2复位控制 当该位置位时，UART模块2复位。所有内部数据丢失，同时寄存器回到复位状态。该位在置位后必须手动清零。
1	UART1	RO	0x0	UART1复位控制 当该位置位时，UART模块1复位。所有内部数据丢失，同时寄存器回到复位状态。该位在置位后必须手动清零。
0	UART0	RO	0x0	UART0复位控制 当该位置位时，UART模块0复位。所有内部数据丢失，同时寄存器回到复位状态。该位在置位后必须手动清零。

寄存器 118: 软件复位控制寄存器 2（SRCR2），偏移量 0x048

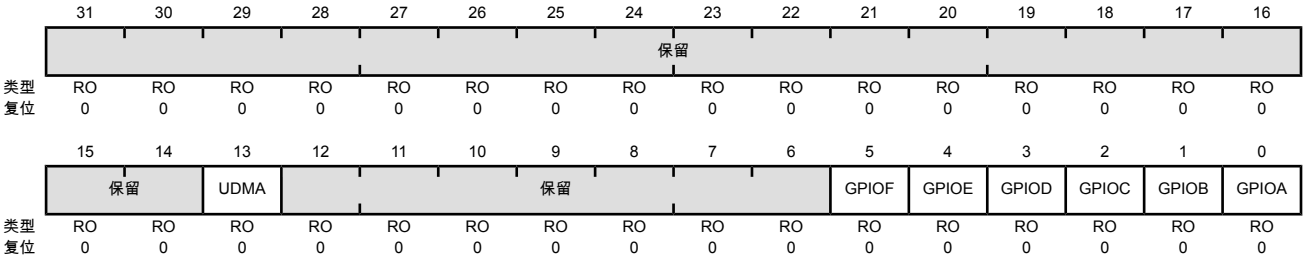
该寄存器允许模块被单独复位。写入该寄存器的值被器件功能 4 (DC4) 寄存器的位屏蔽。

重要: 该寄存器用于实现传统软件支持。

应将外设专用软件复位寄存器（如 SRDMA）用于复位特定的外设。写该传统寄存器的同时，也将写外设专用寄存器中的相应位。通过对该寄存器的写操作更改的任何位都可以通过对该寄存器的读操作进行正确回读。软件必须使用外设专用寄存器以支持不处于传统寄存器中的模块。如果软件使用外设专用寄存器对传统外设（如 µDMA）执行写操作，则写操作会产生正确操作，但是该位的值不在该寄存器中得到反映。如果软件使用传统和外设专用寄存器访问，则必须通过读-修改-写的操作来访问外设专用寄存器，因为该操作仅影响不在传统寄存器中的外设。通过这种方法，外设专用和传统寄存器都具有一致的信息。

软件复位控制寄存器 2 (SRCR2)

基址 0x400F.E000
偏移量 0x048
类型 RO, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:14	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
13	UDMA	RO	0x0	微DMA复位控制 当该位置位时，uDMA模块0复位。所有内部数据丢失，同时寄存器回到复位状态。该位在置位后必须手动清零。
12:6	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
5	GPIOF	RO	0x0	端口F复位控制 当该位置位时，端口F模块复位。所有内部数据丢失，同时寄存器回到复位状态。该位在置位后必须手动清零。
4	GPIOE	RO	0x0	端口E复位控制 当该位置位时，端口E模块复位。所有内部数据丢失，同时寄存器回到复位状态。该位在置位后必须手动清零。
3	GPIOD	RO	0x0	端口D复位控制 当该位置位时，端口D模块复位。所有内部数据丢失，同时寄存器回到复位状态。该位在置位后必须手动清零。
2	GPIOC	RO	0x0	端口C复位控制 当该位置位时，端口C模块复位。所有内部数据丢失，同时寄存器回到复位状态。该位在置位后必须手动清零。

位/域	名称	类型	复位	描述
1	GPIOB	RO	0x0	端口B复位控制 当该位置位时，端口B模块复位。所有内部数据丢失，同时寄存器回到复位状态。该位在置位后必须手动清零。
0	GPIOA	RO	0x0	端口A复位控制 当该位置位时，端口A模块复位。所有内部数据丢失，同时寄存器回到复位状态。该位在置位后必须手动清零。

寄存器 119: 运行模式时钟门控控制寄存器 0（RCGC0），偏移量 0x100

该寄存器控制正常运行模式的时钟门控逻辑。每一位都控制一个给定的接口、功能或模块的时钟启用。如果置位表示该模块接收到时钟并运行。否则，该单元不使用时钟并且被禁止（节能）。如果这些单元不使用时钟，对它们的读或写都会产生一个总线故障。除非特别说明，否则这些位的复位状态为 0（不使用时钟），以便禁止所有功能单元。应用所需的端口需要通过软件来启用。注意这些寄存器除了含有对接口、功能或模块进行控制的位以外，还可能含有其它位，这样可保证与其它产品系列以及将来的期间实现合理的代码兼容。RCGC0 是运行操作的时钟配置寄存器，SCGC0 是睡眠操作的时钟配置寄存器，DCGC0 是深度睡眠操作的时钟配置寄存器。置位运行模式时钟配置 (RCC) 寄存器的 ACG 位说明系统使用睡眠模式。请注意，在访问该模块中任何寄存器前，启用模块时钟后必须有 3 个系统时钟的延迟。

重要: 该寄存器用于实现传统软件支持。

应将外设专用运行模块时钟门控控制寄存器（如 RCGCWD）用于复位特定的外设。写该传统寄存器的同时，也将写外设专用寄存器中的相应位。通过对该寄存器的写操作更改的任何位都可以通过对该寄存器的读操作进行正确回读。软件必须使用外设专用寄存器以支持不处于传统寄存器中的模块。如果软件使用外设专用寄存器对传统外设（如看门狗 1）执行写操作，则写操作会产生正确操作，但是该位的值不在该寄存器中得到反映。如果软件使用传统和外设专用寄存器访问，则必须通过读-修改-写的操作来访问外设专用寄存器，因为该操作仅影响不在传统寄存器中的外设。通过这种方法，外设专用和传统寄存器都具有一致的信息。

同样，应将 ADC 外设配置 (ADCPC) 寄存器用于配置 ADC 采样率。但是，要支持传统软件，可使用 MAXADCnSPD 域。写这些传统域的同时，也将写外设专用寄存器中的相应域。如果通过对该寄存器进行写操作更改一个域，则可以通过对该寄存器进行读操作对它进行正确回读。软件必须使用外设专用寄存器以支持该寄存器中不可用的比率。如果软件使用外设专用寄存器设置 ADC 比率，则写操作会产生正确操作，但是该域的值不在该寄存器中得到反映。如果软件使用传统和外设专用寄存器访问，则必须通过读-修改-写的操作来访问外设专用寄存器，因为该操作仅影响不在传统寄存器中的外设。通过这种方法，外设专用和传统寄存器都具有一致的信息。

运行模式时钟门控控制寄存器 0 (RCGC0)

基址 0x400F.E000
偏移量 0x100
类型 RO, 复位 0x0000.0040

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
域	保留			WDT1	保留			CAN0	保留						ADC1	ADC0
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
域	保留				MAXADC1SPD		MAXADC0SPD		保留	HIB	保留		WDT0	保留		
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0

位/域	名称	类型	复位	描述
31:29	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
28	WDT1	RO	0x0	WDT1 时钟门控控制 该位控制看门狗定时器模块 1 的时钟门控。如果置位表示该模块接收到时钟并运行。否则该模块不使用时钟并被禁用。如果该模块不使用时钟，对它的读或写都会产生一个总线故障。

位/域	名称	类型	复位	描述
27:25	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
24	CAN0	RO	0x0	CAN0时钟门控控制 这个位控制 CAN 模块 0 的时钟门控。如果置位表示该模块接收到时钟并运行。否则该模块不使用时钟并被禁用。如果该模块不使用时钟，对它的读或写都会产生一个总线故障。
23:18	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
17	ADC1	RO	0x0	ADC1 时钟门控控制 这个位控制 SAR ADC 模块 1 的时钟选通如果置位表示该模块接收到时钟并运行。否则该模块不使用时钟并被禁用。如果该模块不使用时钟，对它的读或写都会产生一个总线故障。
16	ADC0	RO	0x0	ADC0时钟门控控制 这个位控制 ADC 模块 0 的时钟门控。如果置位表示该模块接收到时钟并运行。否则该模块不使用时钟并被禁用。如果该模块不使用时钟，对它的读或写都会产生一个总线故障。
15:12	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
11:10	MAXADC1SPD	RO	0x0	ADC1 采样速率 该域设置 ADC 模块 1 的数据采样速率。不能把速率设置得超过最大速率。采样速率可通过如下方式对 MAXADC1SPD 位置位来设定（其它所有编码保留）： 值 描述 0x3 1M 采样/秒 0x2 500K 采样/秒 0x1 250K 采样/秒 0x0 125K 采样/秒
9:8	MAXADC0SPD	RO	0x0	ADC0 采样速率 该域设置 ADC0 模块的数据采样速率。不能把速率设置得超过最大速率。采样速率可通过如下对 MAXADC0SPD 位的设置而设定（其它所有编码保留）： 值 描述 0x3 1M 采样/秒 0x2 500K 采样/秒 0x1 250K 采样/秒 0x0 125K 采样/秒
7	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
6	HIB	RO	0x1	HIB 时钟门控控制 该位控制休眠模块的时钟门控。如果置位表示该模块接收到时钟并运行。否则该模块不使用时钟并被禁用。如果该模块不使用时钟，对它的读或写都会产生一个总线故障。

位/域	名称	类型	复位	描述
5:4	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
3	WDT0	RO	0x0	WDT0 时钟门控控制 该位控制看门狗定时器模块 0 的时钟门控。如果置位表示该模块接收到时钟并运行。否则该模块不使用时钟并被禁用。如果该模块不使用时钟，对它的读或写都会产生一个总线故障。
2:0	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。

寄存器 120: 运行模式时钟门控控制寄存器 1（RCGC1），偏移量 0x104

该寄存器控制正常运行模式的时钟门控逻辑。每一位都控制一个给定的接口、功能或模块的时钟启用。如果置位表示该模块接收到时钟并运行。否则，该单元不使用时钟并且被禁止（节能）。如果这些单元不使用时钟，对它们的读或写都会产生一个总线故障。除非特别说明，否则这些位的复位状态为 0（不使用时钟），以便禁止所有功能单元。应用所需的端口需要通过软件来启用。注意这些寄存器除了含有对接口、功能或模块进行控制的位以外，还可能含有其它位，这样可保证与其它产品系列以及将来的期间实现合理的代码兼容。RCGC1 是运行操作的时钟配置寄存器，SCGC1 是睡眠操作的时钟配置寄存器，DCGC1 是深度睡眠操作的时钟配置寄存器。置位运行模式时钟配置 (RCC) 寄存器的 ACG 位说明系统使用睡眠模式。请注意，在访问该模块中任何寄存器前，启用模块时钟后必须有 3 个系统时钟的延迟。

重要: 该寄存器用于实现传统软件支持。

应将外设专用运行模式时钟门控控制寄存器（如 RCGCTIMER）用于复位特定的外设。写该传统寄存器的同时，也将写外设专用寄存器中的相应位。通过对该寄存器的写操作更改的任何位都可以通过对该寄存器的读操作进行正确回读。软件必须使用外设专用寄存器以支持不处于传统寄存器中的模块。如果软件使用外设专用寄存器对传统外设（如 Timer 0）执行写操作，则写操作会产生正确操作，但是该位的值不在该寄存器中得到反映。如果软件使用传统和外设专用寄存器访问，则必须通过读-修改-写的操作来访问外设专用寄存器，因为该操作仅影响不在传统寄存器中的外设。通过这种方法，外设专用和传统寄存器都具有一致的信息。

运行模式时钟门控控制寄存器 1 (RCGC1)

基址 0x400F.E000
偏移量 0x104
类型 RO, 复位 0x0000.0000

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
域	保留						COMP1	COMP0	保留				TIMER3	TIMER2	TIMER1	TIMER0
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
域	保留	I2C1	保留	I2C0	保留						SSI1	SSI0	保留	UART2	UART1	UART0
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位/域	名称	类型	复位	描述
31:26	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
25	COMP1	RO	0x0	模拟比较器1时钟门控 这个位控制模拟比较器 1 的时钟门控。如果置位表示该模块接收到时钟并运行。否则该模块不使用时钟并被禁用。如果该模块不使用时钟，对它的读或写都会产生一个总线故障。
24	COMP0	RO	0x0	模拟比较器0时钟门控 这个位控制模拟比较器 0 的时钟门控。如果置位表示该模块接收到时钟并运行。否则该模块不使用时钟并被禁用。如果该模块不使用时钟，对它的读或写都会产生一个总线故障。
23:20	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。

位/域	名称	类型	复位	描述
19	TIMER3	RO	0x0	定时器3时钟门控控制 这个位控制通用定时器模块 3 的时钟门控。如果置位表示该模块接收到时钟并运行。否则该模块不使用时钟并被禁用。如果该模块不使用时钟，对它的读或写都会产生一个总线故障。
18	TIMER2	RO	0x0	定时器2时钟门控控制 这个位控制通用定时器模块 2 的时钟门控。如果置位表示该模块接收到时钟并运行。否则该模块不使用时钟并被禁用。如果该模块不使用时钟，对它的读或写都会产生一个总线故障。
17	TIMER1	RO	0x0	定时器1时钟门控控制 这个位控制通用定时器模块 1 的时钟门控。如果置位表示该模块接收到时钟并运行。否则该模块不使用时钟并被禁用。如果该模块不使用时钟，对它的读或写都会产生一个总线故障。
16	TIMER0	RO	0x0	定时器0时钟门控控制 这个位控制通用定时器模块 0 的时钟门控。如果置位表示该模块接收到时钟并运行。否则该模块不使用时钟并被禁用。如果该模块不使用时钟，对它的读或写都会产生一个总线故障。
15	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
14	I2C1	RO	0x0	I2C1时钟门控控制 这个位控制 I2C 模块 1 的时钟门控。如果置位表示该模块接收到时钟并运行。否则该模块不使用时钟并被禁用。如果该模块不使用时钟，对它的读或写都会产生一个总线故障。
13	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
12	I2C0	RO	0x0	I2C0时钟门控控制 这个位控制 I2C 模块 0 的时钟门控。如果置位表示该模块接收到时钟并运行。否则该模块不使用时钟并被禁用。如果该模块不使用时钟，对它的读或写都会产生一个总线故障。
11:6	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
5	SSI1	RO	0x0	SSI1时钟门控控制 这个位控制 SSI 模块 1 的时钟门控。如果置位表示该模块接收到时钟并运行。否则该模块不使用时钟并被禁用。如果该模块不使用时钟，对它的读或写都会产生一个总线故障。
4	SSI0	RO	0x0	SSI0时钟门控控制 这个位控制 SSI 模块 0 的时钟门控。如果置位表示该模块接收到时钟并运行。否则该模块不使用时钟并被禁用。如果该模块不使用时钟，对它的读或写都会产生一个总线故障。
3	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
2	UART2	RO	0x0	UART2时钟门控控制 这个位控制 UART 模块 2 的时钟门控。如果置位表示该模块接收到时钟并运行。否则该模块不使用时钟并被禁用。如果该模块不使用时钟，对它的读或写都会产生一个总线故障。

位/域	名称	类型	复位	描述
1	UART1	RO	0x0	UART1时钟门控控制 这个位控制 UART 模块 1 的时钟门控。如果置位表示该模块接收到时钟并运行。否则该模块不使用时钟并被禁用。如果该模块不使用时钟，对它的读或写都会产生一个总线故障。
0	UART0	RO	0x0	UART0时钟门控控制 这个位控制 UART 模块 0 的时钟门控。如果置位表示该模块接收到时钟并运行。否则该模块不使用时钟并被禁用。如果该模块不使用时钟，对它的读或写都会产生一个总线故障。

寄存器 121: 运行模式时钟门控控制寄存器 2（RCGC2），偏移量 0x108

该寄存器控制正常运行模式的时钟门控逻辑。每一位都控制一个给定的接口、功能或模块的时钟启用。如果置位表示该模块接收到时钟并运行。否则，该单元不使用时钟并且被禁止（节能）。如果这些单元不使用时钟，对它们的读或写都会产生一个总线故障。除非特别说明，否则这些位的复位状态为 0（不使用时钟），以便禁止所有功能单元。应用所需的端口需要通过软件来启用。注意这些寄存器除了含有对接口、功能或模块进行控制的位以外，还可能含有其它位，这样可保证与其它产品系列以及将来的期间实现合理的代码兼容。RCGC2 是运行操作的时钟配置寄存器，SCGC2 是睡眠操作的时钟配置寄存器，DCGC2 是深度睡眠操作的时钟配置寄存器。置位运行模式时钟配置 (RCC) 寄存器的 ACG 位说明系统使用睡眠模式。请注意，在访问该模块中任何寄存器前，启用模块时钟后必须有 3 个系统时钟的延迟。

重要: 该寄存器用于实现传统软件支持。

应将外设专用运行模块时钟门控控制寄存器（如 RCGCDMA）用于复位特定的外设。写该传统寄存器的同时，也将写外设专用寄存器中的相应位。通过对该寄存器的写操作更改的任何位都可以通过对该寄存器的读操作进行正确回读。软件必须使用外设专用寄存器以支持不处于传统寄存器中的模块。如果软件使用外设专用寄存器对传统外设（如 µDMA）执行写操作，则写操作会产生正确操作，但是该位的值不在该寄存器中得到反映。如果软件使用传统和外设专用寄存器访问，则必须通过读-修改-写的操作来访问外设专用寄存器，因为该操作仅影响不在传统寄存器中的外设。通过这种方法，外设专用和传统寄存器都具有一致的信息。

运行模式时钟门控控制寄存器 2 (RCGC2)

基址 0x400F.E000
偏移量 0x108
类型 RO, 复位 0x0000.0000

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	保留															
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	保留		UDMA	保留							GPIOF	GPIOE	GPIOD	GPIOC	GPIOB	GPIOA
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位/域	名称	类型	复位	描述
31:14	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
13	UDMA	RO	0x0	微DMA时钟门控控制 该位控制微DMA的时钟门控。如果置位表示该模块接收到时钟并运行。否则该模块不使用时钟并被禁用。如果该模块不使用时钟，对它的读或写都会产生一个总线故障。
12:6	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
5	GPIOF	RO	0x0	端口F时钟门控控制 该位控制端口F的时钟门控。如果置位表示该模块接收到时钟并运行。否则该模块不使用时钟并被禁用。如果该模块不使用时钟，对它的读或写都会产生一个总线故障。

位/域	名称	类型	复位	描述
4	GPIOE	RO	0x0	端口E时钟门控控制 端口E时钟门控控制。该位控制端口E的时钟门控。如果置位表示该模块接收到时钟并运行。否则该模块不使用时钟并被禁用。如果该模块不使用时钟，对它的读或写都会产生一个总线故障。
3	GPIOD	RO	0x0	端口D时钟门控控制 端口 D 时钟门控控制。该位控制端口 D 的时钟门控。如果置位表示该模块接收到时钟并运行。否则该模块不使用时钟并被禁用。如果该模块不使用时钟，对它的读或写都会产生一个总线故障。
2	GPIOC	RO	0x0	端口C时钟门控控制 该位控制端口C的时钟门控。如果置位表示该模块接收到时钟并运行。否则该模块不使用时钟并被禁用。如果该模块不使用时钟，对它的读或写都会产生一个总线故障。
1	GPIOB	RO	0x0	端口B时钟门控控制 该位控制端口B的时钟门控。如果置位表示该模块接收到时钟并运行。否则该模块不使用时钟并被禁用。如果该模块不使用时钟，对它的读或写都会产生一个总线故障。
0	GPIOA	RO	0x0	端口A时钟门控控制 该位控制端口A的时钟门控。如果置位表示该模块接收到时钟并运行。否则该模块不使用时钟并被禁用。如果该模块不使用时钟，对它的读或写都会产生一个总线故障。

寄存器 122: 睡眠模式时钟门控控制寄存器 0（SCGC0），偏移量 0x110

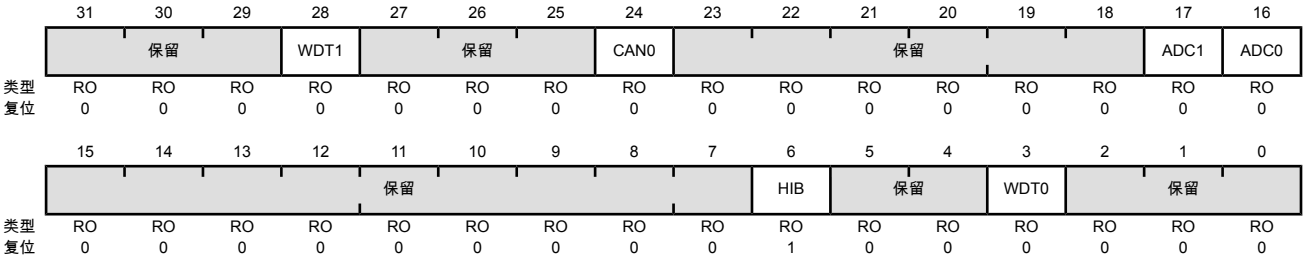
该寄存器控制睡眠模式的时钟门控逻辑。每一位都控制一个给定的接口、功能或模块的时钟启用。如果置位表示该模块接收到时钟并运行。否则，该单元不使用时钟并且被禁止（节能）。如果这些单元不使用时钟，对它们的读或写都会产生一个总线故障。除非特别说明，否则这些位的复位状态为 0（不使用时钟），以便禁止所有功能单元。应用所需的端口需要通过软件来启用。注意这些寄存器除了含有对接口、功能或模块进行控制的位以外，还可能含有其它位，这样可保证与其它产品系列以及将来的期间实现合理的代码兼容。RCGC0 是运行操作的时钟配置寄存器，SCGC0 是睡眠操作的时钟配置寄存器，DCGC0 是深度睡眠操作的时钟配置寄存器。置位运行模式时钟配置 (RCC) 寄存器的 ACG 位说明系统使用睡眠模式。

重要: 该寄存器用于实现传统软件支持。

应将外设专用睡眠模块时钟门控控制寄存器（如 SCGCWD）用于复位特定的外设。写该传统寄存器的同时，也将写外设专用寄存器中的相应位。通过对该寄存器的写操作更改的任何位都可以通过对该寄存器的读操作进行正确回读。软件必须使用外设专用寄存器以支持不处于传统寄存器中的模块。如果软件使用外设专用寄存器对传统外设（如看门狗 1）执行写操作，则写操作会产生正确操作，但是该位的值不在该寄存器中得到反映。如果软件使用传统和外设专用寄存器访问，则必须通过读-修改-写的操作来访问外设专用寄存器，因为该操作仅影响不在传统寄存器中的外设。通过这种方法，外设专用和传统寄存器都具有一致的信息。

睡眠模式时钟门控控制寄存器 0 (SCGC0)

基址 0x400F.E000
偏移量 0x110
类型 RO, 复位 0x0000.0040



位/域	名称	类型	复位	描述
31:29	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
28	WDT1	RO	0x0	WDT1 时钟门控控制 该位控制看门狗定时器模块 1 的时钟门控。如果置位表示该模块接收到时钟并运行。否则该模块不使用时钟并被禁用。如果该模块不使用时钟，对它的读或写都会产生一个总线故障。
27:25	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
24	CAN0	RO	0x0	CAN0时钟门控控制 这个位控制 CAN 模块 0 的时钟门控。如果置位表示该模块接收到时钟并运行。否则该模块不使用时钟并被禁用。如果该模块不使用时钟，对它的读或写都会产生一个总线故障。
23:18	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。

位/域	名称	类型	复位	描述
17	ADC1	RO	0x0	ADC1 时钟门控控制 这个位控制 ADC 模块 1 的时钟门控。如果置位表示该模块接收到时钟并运行。否则该模块不使用时钟并被禁用。如果该模块不使用时钟，对它的读或写都会产生一个总线故障。
16	ADC0	RO	0x0	ADC0时钟门控控制 这个位控制 ADC 模块 0 的时钟门控。如果置位表示该模块接收到时钟并运行。否则该模块不使用时钟并被禁用。如果该模块不使用时钟，对它的读或写都会产生一个总线故障。
15:7	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
6	HIB	RO	0x1	HIB 时钟门控控制 该位控制休眠模块的时钟门控。如果置位表示该模块接收到时钟并运行。否则该模块不使用时钟并被禁用。如果该模块不使用时钟，对它的读或写都会产生一个总线故障。
5:4	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
3	WDT0	RO	0x0	WDT0 时钟门控控制 该位控制看门狗定时器模块 0 的时钟门控。如果置位表示该模块接收到时钟并运行。否则该模块不使用时钟并被禁用。如果该模块不使用时钟，对它的读或写都会产生一个总线故障。
2:0	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。

寄存器 123: 睡眠模式时钟门控控制寄存器 1（SCGC1），偏移量 0x114

该寄存器控制睡眠模式的时钟门控逻辑。每一位都控制一个给定的接口、功能或模块的时钟启用。如果置位表示该模块接收到时钟并运行。否则，该单元不使用时钟并且被禁止（节能）。如果这些单元不使用时钟，对它们的读或写都会产生一个总线故障。除非特别说明，否则这些位的复位状态为 0（不使用时钟），以便禁止所有功能单元。应用所需的端口需要通过软件来启用。注意这些寄存器除了含有对接口、功能或模块进行控制的位以外，还可能含有其它位，这样可保证与其它产品系列以及将来的期间实现合理的代码兼容。RCGC1 是运行操作的时钟配置寄存器，SCGC1 是睡眠操作的时钟配置寄存器，DCGC1 是深度睡眠操作的时钟配置寄存器。置位运行模式时钟配置 (RCC) 寄存器的 ACG 位说明系统使用睡眠模式。

重要: 该寄存器用于实现传统软件支持。

应将外设专用睡眠模式时钟门控控制寄存器（如 SCGCTIMER）用于复位特定的外设。写该传统寄存器的同时，也将写外设专用寄存器中的相应位。通过对该寄存器的写操作更改的任何位都可以通过对该寄存器的读操作进行正确回读。软件必须使用外设专用寄存器以支持不处于传统寄存器中的模块。如果软件使用外设专用寄存器对传统外设（如 Timer 0）执行写操作，则写操作会产生正确操作，但是该位的值不在该寄存器中得到反映。如果软件使用传统和外设专用寄存器访问，则必须通过读-修改-写的操作来访问外设专用寄存器，因为该操作仅影响不在传统寄存器中的外设。通过这种方法，外设专用和传统寄存器都具有一致的信息。

睡眠模式时钟门控控制寄存器 1 (SCGC1)

基址 0x400F.E000
偏移量 0x114
类型 RO, 复位 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	保留						COMP1	COMP0	保留				TIMER3	TIMER2	TIMER1	TIMER0
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	保留	I2C1	保留	I2C0	保留						SSI1	SSI0	保留	UART2	UART1	UART0
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位/域	名称	类型	复位	描述
31:26	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
25	COMP1	RO	0x0	模拟比较器1时钟门控 这个位控制模拟比较器 1 的时钟门控。如果置位表示该模块接收到时钟并运行。否则该模块不使用时钟并被禁用。如果该模块不使用时钟，对它的读或写都会产生一个总线故障。
24	COMP0	RO	0x0	模拟比较器0时钟门控 这个位控制模拟比较器 0 的时钟门控。如果置位表示该模块接收到时钟并运行。否则该模块不使用时钟并被禁用。如果该模块不使用时钟，对它的读或写都会产生一个总线故障。
23:20	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。

位/域	名称	类型	复位	描述
19	TIMER3	RO	0x0	定时器3时钟门控控制 这个位控制通用定时器模块 3 的时钟门控。如果置位表示该模块接收到时钟并运行。否则该模块不使用时钟并被禁用。如果该模块不使用时钟，对它的读或写都会产生一个总线故障。
18	TIMER2	RO	0x0	定时器2时钟门控控制 这个位控制通用定时器模块 2 的时钟门控。如果置位表示该模块接收到时钟并运行。否则该模块不使用时钟并被禁用。如果该模块不使用时钟，对它的读或写都会产生一个总线故障。
17	TIMER1	RO	0x0	定时器1时钟门控控制 这个位控制通用定时器模块 1 的时钟门控。如果置位表示该模块接收到时钟并运行。否则该模块不使用时钟并被禁用。如果该模块不使用时钟，对它的读或写都会产生一个总线故障。
16	TIMER0	RO	0x0	定时器0时钟门控控制 这个位控制通用定时器模块 0 的时钟门控。如果置位表示该模块接收到时钟并运行。否则该模块不使用时钟并被禁用。如果该模块不使用时钟，对它的读或写都会产生一个总线故障。
15	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
14	I2C1	RO	0x0	I2C1时钟门控控制 这个位控制 I2C 模块 1 的时钟门控。如果置位表示该模块接收到时钟并运行。否则该模块不使用时钟并被禁用。如果该模块不使用时钟，对它的读或写都会产生一个总线故障。
13	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
12	I2C0	RO	0x0	I2C0时钟门控控制 这个位控制 I2C 模块 0 的时钟门控。如果置位表示该模块接收到时钟并运行。否则该模块不使用时钟并被禁用。如果该模块不使用时钟，对它的读或写都会产生一个总线故障。
11:6	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
5	SSI1	RO	0x0	SSI1时钟门控控制 这个位控制 SSI 模块 1 的时钟门控。如果置位表示该模块接收到时钟并运行。否则该模块不使用时钟并被禁用。如果该模块不使用时钟，对它的读或写都会产生一个总线故障。
4	SSI0	RO	0x0	SSI0时钟门控控制 这个位控制 SSI 模块 0 的时钟门控。如果置位表示该模块接收到时钟并运行。否则该模块不使用时钟并被禁用。如果该模块不使用时钟，对它的读或写都会产生一个总线故障。
3	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
2	UART2	RO	0x0	UART2时钟门控控制 这个位控制 UART 模块 2 的时钟门控。如果置位表示该模块接收到时钟并运行。否则该模块不使用时钟并被禁用。如果该模块不使用时钟，对它的读或写都会产生一个总线故障。

位/域	名称	类型	复位	描述
1	UART1	RO	0x0	UART1时钟门控控制 这个位控制 UART 模块 1 的时钟门控。如果置位表示该模块接收到时钟并运行。否则该模块不使用时钟并被禁用。如果该模块不使用时钟，对它的读或写都会产生一个总线故障。
0	UART0	RO	0x0	UART0时钟门控控制 这个位控制 UART 模块 0 的时钟门控。如果置位表示该模块接收到时钟并运行。否则该模块不使用时钟并被禁用。如果该模块不使用时钟，对它的读或写都会产生一个总线故障。

寄存器 124: 睡眠模式时钟门控控制寄存器 2（SCGC2），偏移量 0x118

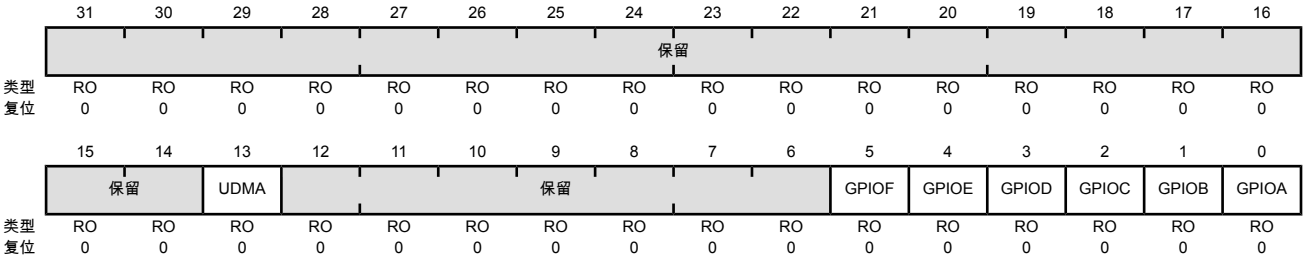
该寄存器控制睡眠模式的时钟门控逻辑。每一位都控制一个给定的接口、功能或模块的时钟启用。如果置位表示该模块接收到时钟并运行。否则，该单元不使用时钟并且被禁止（节能）。如果这些单元不使用时钟，对它们的读或写都会产生一个总线故障。除非特别说明，否则这些位的复位状态为 0（不使用时钟），以便禁止所有功能单元。应用所需的端口需要通过软件来启用。注意这些寄存器除了含有对接口、功能或模块进行控制的位以外，还可能含有其它位，这样可保证与其它产品系列以及将来的期间实现合理的代码兼容。RCGC2 是运行操作的时钟配置寄存器，SCGC2 是睡眠操作的时钟配置寄存器，DCGC2 是深度睡眠操作的时钟配置寄存器。置位运行模式时钟配置 (RCC) 寄存器的 ACG 位说明系统使用睡眠模式。

重要: 该寄存器用于实现传统软件支持。

应将外设专用睡眠模块时钟门控控制寄存器（如 SCGCDMA）用于复位特定的外设。写该传统寄存器的同时，也将写外设专用寄存器中的相应位。通过对该寄存器的写操作更改的任何位都可以通过对该寄存器的读操作进行正确回读。软件必须使用外设专用寄存器以支持不处于传统寄存器中的模块。如果软件使用外设专用寄存器对传统外设（如 µDMA）执行写操作，则写操作会产生正确操作，但是该位的值不在该寄存器中得到反映。如果软件使用传统和外设专用寄存器访问，则必须通过读-修改-写的操作来访问外设专用寄存器，因为该操作仅影响不在传统寄存器中的外设。通过这种方法，外设专用和传统寄存器都具有一致的信息。

睡眠模式时钟门控控制寄存器 2 (SCGC2)

基址 0x400F.E000
偏移量 0x118
类型 RO, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:14	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
13	UDMA	RO	0x0	微DMA时钟门控控制 该位控制微DMA的时钟门控。如果置位表示该模块接收到时钟并运行。否则该模块不使用时钟并被禁用。如果该模块不使用时钟，对它的读或写都会产生一个总线故障。
12:6	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
5	GPIOF	RO	0x0	端口F时钟门控控制 该位控制端口F的时钟门控。如果置位表示该模块接收到时钟并运行。否则该模块不使用时钟并被禁用。如果该模块不使用时钟，对它的读或写都会产生一个总线故障。

位/域	名称	类型	复位	描述
4	GPIOE	RO	0x0	端口E时钟门控控制 端口E时钟门控控制。该位控制端口E的时钟门控。如果置位表示该模块接收到时钟并运行。否则该模块不使用时钟并被禁用。如果该模块不使用时钟，对它的读或写都会产生一个总线故障。
3	GPIOD	RO	0x0	端口D时钟门控控制 端口 D 时钟门控控制。该位控制端口 D 的时钟门控。如果置位表示该模块接收到时钟并运行。否则该模块不使用时钟并被禁用。如果该模块不使用时钟，对它的读或写都会产生一个总线故障。
2	GPIOC	RO	0x0	端口C时钟门控控制 该位控制端口C的时钟门控。如果置位表示该模块接收到时钟并运行。否则该模块不使用时钟并被禁用。如果该模块不使用时钟，对它的读或写都会产生一个总线故障。
1	GPIOB	RO	0x0	端口B时钟门控控制 该位控制端口B的时钟门控。如果置位表示该模块接收到时钟并运行。否则该模块不使用时钟并被禁用。如果该模块不使用时钟，对它的读或写都会产生一个总线故障。
0	GPIOA	RO	0x0	端口A时钟门控控制 该位控制端口A的时钟门控。如果置位表示该模块接收到时钟并运行。否则该模块不使用时钟并被禁用。如果该模块不使用时钟，对它的读或写都会产生一个总线故障。

寄存器 125: 深度睡眠模式时钟门控控制寄存器 0（DCGC0），偏移量 0x120

该寄存器控制深度睡眠模式的时钟门控逻辑。每一位都控制一个给定的接口、功能或模块的时钟启用。如果置位表示该模块接收到时钟并运行。否则，该单元不使用时钟并且被禁止（节能）。如果这些单元不使用时钟，对它们的读或写都会产生一个总线故障。除非特别说明，否则这些位的复位状态为 0（不使用时钟），以便禁止所有功能单元。应用所需的端口需要通过软件来启用。注意这些寄存器除了含有对接口、功能或模块进行控制的位以外，还可能含有其它位，这样可保证与其它产品系列以及将来的期间实现合理的代码兼容。RCGC0 是运行操作的时钟配置寄存器，SCGC0 是睡眠操作的时钟配置寄存器，DCGC0 是深度睡眠操作的时钟配置寄存器。置位运行模式时钟配置 (RCC) 寄存器的 ACG 位说明系统使用睡眠模式。

重要: 该寄存器用于实现传统软件支持。

应将外设专用深度睡眠模块时钟门控控制寄存器（如 DCGCWD）用于复位特定的外设。写该传统寄存器的同时，也将写外设专用寄存器中的相应位。通过对该寄存器的写操作更改的任何位都可以通过对该寄存器的读操作进行正确回读。软件必须使用外设专用寄存器以支持不处于传统寄存器中的模块。如果软件使用外设专用寄存器对传统外设（如看门狗 1）执行写操作，则写操作会产生正确操作，但是该位的值不在该寄存器中得到反映。如果软件使用传统和外设专用寄存器访问，则必须通过读-修改-写的操作来访问外设专用寄存器，因为该操作仅影响不在传统寄存器中的外设。通过这种方法，外设专用和传统寄存器都具有一致的信息。

深度睡眠模式时钟门控控制寄存器 0 (DCGC0)

基址 0x400F.E000
偏移量 0x120
类型 RO, 复位 0x0000.0040

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
域	保留			WDT1	保留			CAN0	保留						ADC1	ADC0
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
域	保留									HIB	保留		WDT0	保留		
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0

位/域	名称	类型	复位	描述
31:29	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
28	WDT1	RO	0x0	WDT1 时钟门控控制 该位控制看门狗定时器模块 1 的时钟门控。如果置位表示该模块接收到时钟并运行。否则该模块不使用时钟并被禁用。如果该模块不使用时钟，对它的读或写都会产生一个总线故障。
27:25	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
24	CAN0	RO	0x0	CAN0时钟门控控制 这个位控制 CAN 模块 0 的时钟门控。如果置位表示该模块接收到时钟并运行。否则该模块不使用时钟并被禁用。如果该模块不使用时钟，对它的读或写都会产生一个总线故障。
23:18	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。

位/域	名称	类型	复位	描述
17	ADC1	RO	0x0	ADC1 时钟门控控制 这个位控制 ADC 模块 1 的时钟门控。如果置位表示该模块接收到时钟并运行。否则该模块不使用时钟并被禁用。如果该模块不使用时钟，对它的读或写都会产生一个总线故障。
16	ADC0	RO	0x0	ADC0时钟门控控制 这个位控制 ADC 模块 0 的时钟门控。如果置位表示该模块接收到时钟并运行。否则该模块不使用时钟并被禁用。如果该模块不使用时钟，对它的读或写都会产生一个总线故障。
15:7	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
6	HIB	RO	0x1	HIB 时钟门控控制 该位控制休眠模块的时钟门控。如果置位表示该模块接收到时钟并运行。否则该模块不使用时钟并被禁用。如果该模块不使用时钟，对它的读或写都会产生一个总线故障。
5:4	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
3	WDT0	RO	0x0	WDT0 时钟门控控制 该位控制看门狗定时器模块 0 的时钟门控。如果置位表示该模块接收到时钟并运行。否则该模块不使用时钟并被禁用。如果该模块不使用时钟，对它的读或写都会产生一个总线故障。
2:0	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。

寄存器 126: 深度睡眠模式时钟门控控制寄存器 1（DCGC1），偏移量 0x124

该寄存器控制深度睡眠模式的时钟门控逻辑。每一位都控制一个给定的接口、功能或模块的时钟启用。如果置位表示该模块接收到时钟并运行。否则，该单元不使用时钟并且被禁止（节能）。如果这些单元不使用时钟，对它们的读或写都会产生一个总线故障。除非特别说明，否则这些位的复位状态为 0（不使用时钟），以便禁止所有功能单元。应用所需的端口需要通过软件来启用。注意这些寄存器除了含有对接口、功能或模块进行控制的位以外，还可能含有其它位，这样可保证与其它产品系列以及将来的期间实现合理的代码兼容。RCGC1 是运行操作的时钟配置寄存器，SCGC1 是睡眠操作的时钟配置寄存器，DCGC1 是深度睡眠操作的时钟配置寄存器。置位运行模式时钟配置 (RCC) 寄存器的 ACG 位说明系统使用睡眠模式。

重要: 该寄存器用于实现传统软件支持。

应将外设专用深度睡眠模式时钟门控控制寄存器（如 DCGCTIMER）用于复位特定的外设。写该传统寄存器的同时，也将写外设专用寄存器中的相应位。通过对该寄存器的写操作更改的任何位都可以通过对该寄存器的读操作进行正确回读。软件必须使用外设专用寄存器以支持不处于传统寄存器中的模块。如果软件使用外设专用寄存器对传统外设（如 Timer 0）执行写操作，则写操作会产生正确操作，但是该位的值不在该寄存器中得到反映。如果软件使用传统和外设专用寄存器访问，则必须通过读-修改-写的操作来访问外设专用寄存器，因为该操作仅影响不在传统寄存器中的外设。通过这种方法，外设专用和传统寄存器都具有一致的信息。

深度睡眠模式时钟门控控制寄存器 1 (DCGC1)

基址 0x400F.E000
偏移量 0x124
类型 RO, 复位 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	保留						COMP1	COMP0	保留				TIMER3	TIMER2	TIMER1	TIMER0
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	保留	I2C1	保留	I2C0	保留						SSI1	SSI0	保留	UART2	UART1	UART0
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位/域	名称	类型	复位	描述
31:26	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
25	COMP1	RO	0x0	模拟比较器1时钟门控 这个位控制模拟比较器 1 的时钟门控。如果置位表示该模块接收到时钟并运行。否则该模块不使用时钟并被禁用。如果该模块不使用时钟，对它的读或写都会产生一个总线故障。
24	COMP0	RO	0x0	模拟比较器0时钟门控 这个位控制模拟比较器 0 的时钟门控。如果置位表示该模块接收到时钟并运行。否则该模块不使用时钟并被禁用。如果该模块不使用时钟，对它的读或写都会产生一个总线故障。
23:20	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。

位/域	名称	类型	复位	描述
19	TIMER3	RO	0x0	定时器3时钟门控控制 这个位控制通用定时器模块 3 的时钟门控。如果置位表示该模块接收到时钟并运行。否则该模块不使用时钟并被禁用。如果该模块不使用时钟，对它的读或写都会产生一个总线故障。
18	TIMER2	RO	0x0	定时器2时钟门控控制 这个位控制通用定时器模块 2 的时钟门控。如果置位表示该模块接收到时钟并运行。否则该模块不使用时钟并被禁用。如果该模块不使用时钟，对它的读或写都会产生一个总线故障。
17	TIMER1	RO	0x0	定时器1时钟门控控制 这个位控制通用定时器模块 1 的时钟门控。如果置位表示该模块接收到时钟并运行。否则该模块不使用时钟并被禁用。如果该模块不使用时钟，对它的读或写都会产生一个总线故障。
16	TIMER0	RO	0x0	定时器0时钟门控控制 这个位控制通用定时器模块 0 的时钟门控。如果置位表示该模块接收到时钟并运行。否则该模块不使用时钟并被禁用。如果该模块不使用时钟，对它的读或写都会产生一个总线故障。
15	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
14	I2C1	RO	0x0	I2C1时钟门控控制 这个位控制 I2C 模块 1 的时钟门控。如果置位表示该模块接收到时钟并运行。否则该模块不使用时钟并被禁用。如果该模块不使用时钟，对它的读或写都会产生一个总线故障。
13	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
12	I2C0	RO	0x0	I2C0时钟门控控制 这个位控制 I2C 模块 0 的时钟门控。如果置位表示该模块接收到时钟并运行。否则该模块不使用时钟并被禁用。如果该模块不使用时钟，对它的读或写都会产生一个总线故障。
11:6	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
5	SSI1	RO	0x0	SSI1时钟门控控制 这个位控制 SSI 模块 1 的时钟门控。如果置位表示该模块接收到时钟并运行。否则该模块不使用时钟并被禁用。如果该模块不使用时钟，对它的读或写都会产生一个总线故障。
4	SSI0	RO	0x0	SSI0时钟门控控制 这个位控制 SSI 模块 0 的时钟门控。如果置位表示该模块接收到时钟并运行。否则该模块不使用时钟并被禁用。如果该模块不使用时钟，对它的读或写都会产生一个总线故障。
3	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
2	UART2	RO	0x0	UART2时钟门控控制 这个位控制 UART 模块 2 的时钟门控。如果置位表示该模块接收到时钟并运行。否则该模块不使用时钟并被禁用。如果该模块不使用时钟，对它的读或写都会产生一个总线故障。

位/域	名称	类型	复位	描述
1	UART1	RO	0x0	UART1时钟门控控制 这个位控制 UART 模块 1 的时钟门控。如果置位表示该模块接收到时钟并运行。否则该模块不使用时钟并被禁用。如果该模块不使用时钟，对它的读或写都会产生一个总线故障。
0	UART0	RO	0x0	UART0时钟门控控制 这个位控制 UART 模块 0 的时钟门控。如果置位表示该模块接收到时钟并运行。否则该模块不使用时钟并被禁用。如果该模块不使用时钟，对它的读或写都会产生一个总线故障。

寄存器 127: 深度睡眠模式时钟门控控制寄存器 2（DCGC2），偏移量 0x128

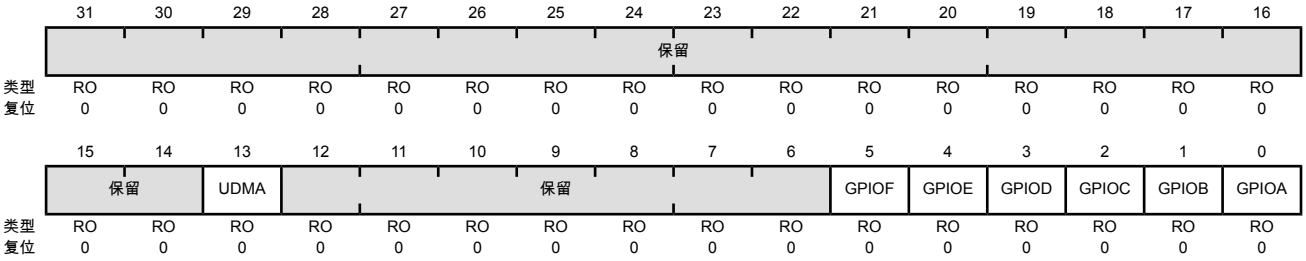
该寄存器控制深度睡眠模式的时钟门控逻辑。每一位都控制一个给定的接口、功能或模块的时钟启用。如果置位表示该模块接收到时钟并运行。否则，该单元不使用时钟并且被禁止（节能）。如果这些单元不使用时钟，对它们的读或写都会产生一个总线故障。除非特别说明，否则这些位的复位状态为 0（不使用时钟），以便禁止所有功能单元。应用所需的端口需要通过软件来启用。注意这些寄存器除了含有对接口、功能或模块进行控制的位以外，还可能含有其它位，这样可保证与其它产品系列以及将来的期间实现合理的代码兼容。RCGC2 是运行操作的时钟配置寄存器，SCGC2 是睡眠操作的时钟配置寄存器，DCGC2 是深度睡眠操作的时钟配置寄存器。置位运行模式时钟配置 (RCC) 寄存器的 ACG 位说明系统使用睡眠模式。

重要: 该寄存器用于实现传统软件支持。

应将外设专用深度睡眠模块时钟门控控制寄存器（如 DCGCDMA）用于复位特定的外设。写该传统寄存器的同时，也将写外设专用寄存器中的相应位。通过对该寄存器的写操作更改的任何位都可以通过对该寄存器的读操作进行正确回读。软件必须使用外设专用寄存器以支持不处于传统寄存器中的模块。如果软件使用外设专用寄存器对传统外设（如 µDMA）执行写操作，则写操作会产生正确操作，但是该位的值不在该寄存器中得到反映。如果软件使用传统和外设专用寄存器访问，则必须通过读-修改-写的操作来访问外设专用寄存器，因为该操作仅影响不在传统寄存器中的外设。通过这种方法，外设专用和传统寄存器都具有一致的信息。

深度睡眠模式时钟门控控制寄存器 2 (DCGC2)

基址 0x400F.E000
偏移量 0x128
类型 RO, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:14	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
13	UDMA	RO	0x0	微DMA时钟门控控制 该位控制微DMA的时钟门控。如果置位表示该模块接收到时钟并运行。否则该模块不使用时钟并被禁用。如果该模块不使用时钟，对它的读或写都会产生一个总线故障。
12:6	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
5	GPIOF	RO	0x0	端口F时钟门控控制 该位控制端口F的时钟门控。如果置位表示该模块接收到时钟并运行。否则该模块不使用时钟并被禁用。如果该模块不使用时钟，对它的读或写都会产生一个总线故障。

位/域	名称	类型	复位	描述
4	GPIOE	RO	0x0	端口E时钟门控控制 端口E时钟门控控制。该位控制端口E的时钟门控。如果置位表示该模块接收到时钟并运行。否则该模块不使用时钟并被禁用。如果该模块不使用时钟，对它的读或写都会产生一个总线故障。
3	GPIOD	RO	0x0	端口D时钟门控控制 端口 D 时钟门控控制。该位控制端口 D 的时钟门控。如果置位表示该模块接收到时钟并运行。否则该模块不使用时钟并被禁用。如果该模块不使用时钟，对它的读或写都会产生一个总线故障。
2	GPIOC	RO	0x0	端口C时钟门控控制 该位控制端口C的时钟门控。如果置位表示该模块接收到时钟并运行。否则该模块不使用时钟并被禁用。如果该模块不使用时钟，对它的读或写都会产生一个总线故障。
1	GPIOB	RO	0x0	端口B时钟门控控制 该位控制端口B的时钟门控。如果置位表示该模块接收到时钟并运行。否则该模块不使用时钟并被禁用。如果该模块不使用时钟，对它的读或写都会产生一个总线故障。
0	GPIOA	RO	0x0	端口A时钟门控控制 该位控制端口A的时钟门控。如果置位表示该模块接收到时钟并运行。否则该模块不使用时钟并被禁用。如果该模块不使用时钟，对它的读或写都会产生一个总线故障。

寄存器 128: 器件功能寄存器 9（DC9），偏移量 0x190

该寄存器由器件预定义且可用于校验 ADC 数字比较器特性。

重要: 该寄存器用于实现传统软件支持。

应使用 ADC 外设属性 (ADCPP) 寄存器确定在 ADC 模块上可用的数字比较器数量。读取该寄存器即可正确识别传统比较器是否存在。软件必须使用 ADCPP 寄存器来确定不受 DCn 寄存器支持的比较器是否存在。

器件功能寄存器 9 (DC9)

基址 0x400F.E000
偏移量 0x190
类型 RO, 复位 0x00FF.00FF

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	保留								ADC1DC7	ADC1DC6	ADC1DC5	ADC1DC4	ADC1DC3	ADC1DC2	ADC1DC1	ADC1DC0
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	保留								ADC0DC7	ADC0DC6	ADC0DC5	ADC0DC4	ADC0DC3	ADC0DC2	ADC0DC1	ADC0DC0
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1

位/域	名称	类型	复位	描述
31:24	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
23	ADC1DC7	RO	0x1	ADC1 DC7 存在 置位表示 ADC 模块 1 的数字比较器 7 存在。
22	ADC1DC6	RO	0x1	ADC1 DC6 存在 置位表示 ADC 模块 1 的数字比较器 6 存在。
21	ADC1DC5	RO	0x1	ADC1 DC5存在 置位表示 ADC 模块 1 的数字比较器 5 存在。
20	ADC1DC4	RO	0x1	ADC1 DC4存在 置位表示 ADC 模块 1 的数字比较器 4 存在。
19	ADC1DC3	RO	0x1	ADC1 DC3存在 置位表示 ADC 模块 1 的数字比较器 3 存在。
18	ADC1DC2	RO	0x1	ADC1 DC2存在 置位表示 ADC 模块 1 的数字比较器 2 存在。
17	ADC1DC1	RO	0x1	ADC1 DC1存在 置位表示 ADC 模块 1 的数字比较器 1 存在。
16	ADC1DC0	RO	0x1	ADC1 DC0存在 置位表示 ADC 模块 1 的数字比较器 0 存在。
15:8	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。

位/域	名称	类型	复位	描述
7	ADC0DC7	RO	0x1	ADC0 DC7存在 置位表示 ADC 模块 0 的数字比较器 7 存在。
6	ADC0DC6	RO	0x1	ADC0 DC6存在 置位表示 ADC 模块 0 的数字比较器 6 存在。
5	ADC0DC5	RO	0x1	ADC0 DC5存在 置位表示 ADC 模块 0 的数字比较器 5 存在。
4	ADC0DC4	RO	0x1	ADC0 DC4存在 置位表示 ADC 模块 0 的数字比较器 4 存在。
3	ADC0DC3	RO	0x1	ADC0 DC3存在 置位表示 ADC 模块 0 的数字比较器 3 存在。
2	ADC0DC2	RO	0x1	ADC0 DC2存在 置位表示 ADC 模块 0 的数字比较器 2 存在。
1	ADC0DC1	RO	0x1	ADC0 DC1存在 置位表示 ADC 模块 0 的数字比较器 1 存在。
0	ADC0DC0	RO	0x1	ADC0 DC0存在 置位表示 ADC 模块 0 的数字比较器 0 存在。

寄存器 129: 非易失性存储器信息寄存器（NVMSTAT），偏移量 0x1A0

该寄存器由器件预定义且可用于校验特性。

重要: 该寄存器用于实现传统软件支持。

应使用 ROM 第三方软件 (ROMSWMAP) 寄存器确定该微控制器的片上 ROM 中是否存在第三方软件。读取寄存器中的 TPSW 位可以正确地识别是否存在传统第三方软件。软件应使用 ROMSWMAP 寄存器来识别不在传统器件上的软件。

非易失性存储器信息寄存器 (NVMSTAT)

基址 0x400F.E000
偏移量 0x1A0
类型 RO, 复位 0x0000.0001

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	保留															
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	保留															FWB
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

位/域	名称	类型	复位	描述
31:1	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
0	FWB	RO	0x1	32 个字的 Flash 写缓存器可用 置位表示 32 个字的 Flash 存储器写缓存器特性可用。

6 系统异常模块

该模块是处理系统级 Cortex-M4 FPU 异常情况的 AHB 外设。对于寄存器映射到此存储器槽的功能，如果器件不提供该功能，则写入相关寄存器的所有值都将被忽略，读取值返回零。

6.1 功能说明

系统异常模块提供系统级中断的控制和状态。由于在发送到中断控制器前，所有中断事件会进行一次逻辑或操作，因此在任何给定的时间，系统异常模块只能向控制器发送一个中断请求。通过读取系统异常屏蔽的中断状态 (SYSEXCMIS) 寄存器，软件可以在一个中断服务例程中处理多个中断事件。对系统异常中断屏蔽 (SYSEXCIM) 寄存器中相应的中断屏蔽位进行置位，可以定义能够触发控制器级中断的中断事件。如果不使用中断，通过系统异常原始中断状态 (SYSEXCRIS) 寄存器可随时查看原始中断状态。向系统异常中断清零 (SYSEXCIC) 寄存器的相应位写 1，即可（为 SYSEXCMIS 寄存器和 SYSEXCRIS 寄存器）清除中断状态。

6.2 寄存器映射

表 6-1（421页）列出了系统异常模块寄存器。表中列出的偏移量是寄存器地址相对于 0x400F.9000 的系统异常基址的 16 进制增量。

注意: 未使用的系统异常寄存器空间保留用于未来使用或内部使用。软件不得修改任何保留的存储器地址。

表 **6-1.** 系统异常 寄存器映射

偏移量	名称	类型	复位	描述	见页面
0x000	SYSEXCRIS	RO	0x0000.0000	系统异常原始中断状态	422
0x004	SYSEXCIM	R/W	0x0000.0000	系统异常中断屏蔽	424
0x008	SYSEXCMIS	RO	0x0000.0000	系统异常屏蔽的中断状态	426
0x00C	SYSEXCIC	W1C	0x0000.0000	系统异常中断清零	428

6.3 寄存器描述

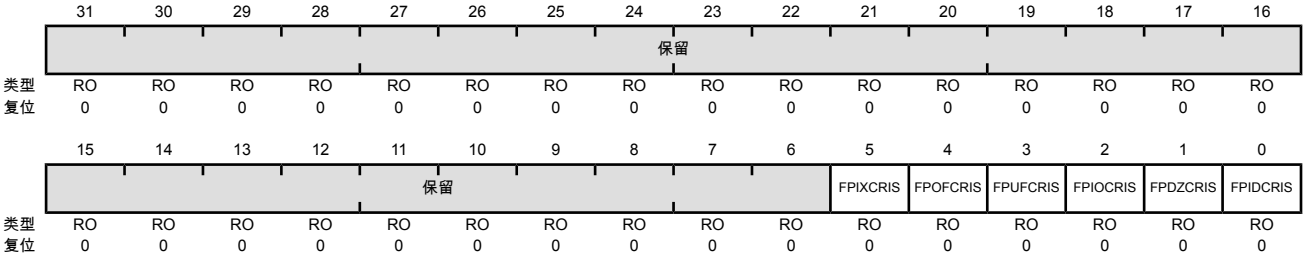
所有给出的地址都是相对于 0x400F.9000 系统异常基址而言的。

寄存器 1: 系统异常原始中断状态（SYSEXCRIS），偏移量 0x000

SYSEXCRIS 寄存器是原始中断状态寄存器。当读取本寄存器时，返回当前各个中断的原始状态。对本寄存器写操作无效。

系统异常原始中断状态 (SYSEXCRIS)

基址 0x400F.9000
偏移量 0x000
类型 RO, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:6	保留	RO	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
5	FPIXCRIS	RO	0	浮点不精确异常原始中断状态 值 描述 0 无中断 1 发生了浮点不精确异常。 通过在 SYSEXCIC 寄存器的 IXCIC 位写 1 可以清除这个位。
4	FPOFCRIS	RO	0	浮点上溢异常原始中断状态 值 描述 0 无中断 1 发生了浮点上溢异常。 通过在 SYSEXCIC 寄存器的 OFCIC 位写 1 可以清除这个位。
3	FPUFCRIS	RO	0	浮点下溢异常原始中断状态 值 描述 0 无中断 1 发生了浮点下溢异常。 通过在 SYSEXCIC 寄存器的 UFCIC 位写 1 可以清除这个位。
2	FPIOCRIS	RO	0	浮点无效操作原始中断状态 值 描述 0 无中断 1 发生了浮点无效操作异常。 通过在 SYSEXCIC 寄存器的 IOCIC 位写 1 可以清除这个位。

位/域	名称	类型	复位	描述
1	FPDZCRIS	RO	0	浮点除 0 异常原始中断状态 值 描述 0 无中断 1 发生了浮点除 0 异常。 通过在 SYSEXCIC 寄存器的 DZCIC 位写 1 可以清除这个位。
0	FPIDCRIS	RO	0	浮点输入反常异常原始中断状态 值 描述 0 无中断 1 发生了浮点输入反常异常。 通过在 SYSEXCIC 寄存器的 IDCIC 位写 1 可以清除这个位。

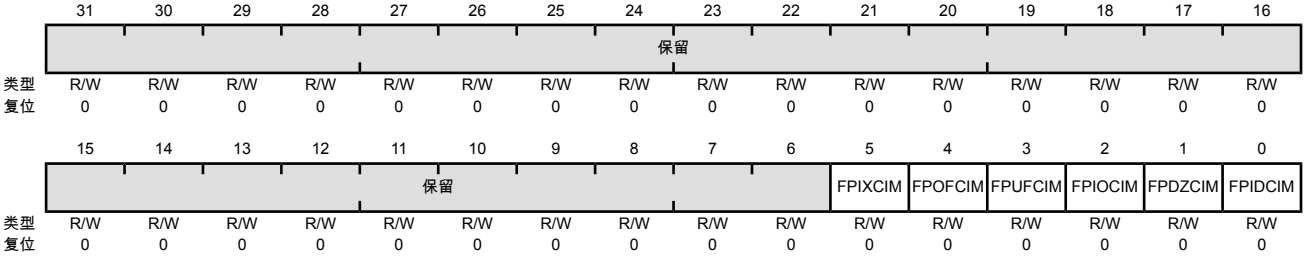
寄存器 2: 系统异常中断屏蔽（SYSEXCIM），偏移量 0x004

SYSEXCIM 寄存器是中断屏蔽置位/清零寄存器。

读本寄存器时，返回各中断的当前掩码状态。对某个位进行置位时，可将相应的原始中断信号传递到中断控制器。将某个位清零时，可阻止该原始中断信号传递到中断控制器。

系统异常中断屏蔽 (SYSEXCIM)

基址 0x400F.9000
偏移量 0x004
类型 R/W, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:6	保留	R/W	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
5	FPIXCIM	R/W	0	浮点不精确异常中断屏蔽 值 描述 0 FPIXCRIS 中断被抑制，不发送到中断控制器。 1 当 SYSEXCRIS 寄存器中的 FPISCRIS 位置位时，向中断控制器发送中断。
4	FPOFCIM	R/W	0	浮点上溢异常中断屏蔽 值 描述 0 FPOFCIS 中断被抑制，不发送到中断控制器。 1 当 SYSEXCRIS 寄存器中的 FPOFCRIS 位置位时，向中断控制器发送中断。
3	FPUFCIM	R/W	0	浮点下溢异常中断屏蔽 值 描述 0 FPUFCRIS 中断被抑制，不发送到中断控制器。 1 当 SYSEXCRIS 寄存器中的 FPUFCRIS 位置位时，向中断控制器发送中断。
2	FPIOCIM	R/W	0	浮点无效操作中断屏蔽 值 描述 0 FPIOCRIS 中断被抑制，不发送到中断控制器。 1 当 SYSEXCRIS 寄存器中的 FPIOCRIS 位置位时，向中断控制器发送中断。

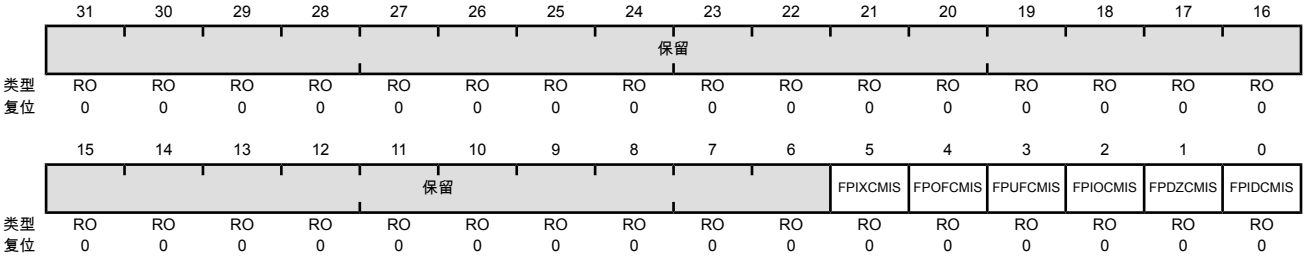
位/域	名称	类型	复位	描述
1	FPDZCIM	R/W	0	浮点除 0 异常中断屏蔽 值 描述 0 FPDZCRIS 中断被抑制，不发送到中断控制器。 1 当 SYSEXCRIS 寄存器中的 FPDZCRIS 位置位时，向中断控制器发送中断。
0	FPIDCIM	R/W	0	浮点输入反常异常中断屏蔽 值 描述 0 FPIDCRIS 中断被抑制，不发送到中断控制器。 1 当SYSEXCRIS寄存器中的FPIDCRIS位置位时，向中断控制器发送中断。

寄存器 3: 系统异常屏蔽的中断状态（SYSEXCMIS），偏移量 0x008

SYSEXCMIS 寄存器是屏蔽的中断状态寄存器。读取时，该寄存器给出相应中断的当前屏蔽状态值。对本寄存器写操作无效。

系统异常屏蔽的中断状态 (SYSEXCMIS)

基址 0x400F.9000
偏移量 0x008
类型 RO, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:6	保留	RO	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
5	FPIXCMIS	RO	0	浮点不精确异常屏蔽的中断状态 值 描述 0 中断没有发生或者被屏蔽。 1 由于不精确异常，因此发出未屏蔽的中断信号。 通过在 SYSEXCIC 寄存器的 FPIXCIC 位写 1 可以清除这个位。
4	FPOFCMIS	RO	0	浮点上溢异常屏蔽的中断状态 值 描述 0 中断没有发生或者被屏蔽。 1 由于上溢异常，因此发出未屏蔽的中断信号。 通过在 SYSEXCIC 寄存器的 FPOFCIC 位写 1 可以清除这个位。
3	FPUFCMIS	RO	0	浮点下溢异常屏蔽的中断状态 值 描述 0 中断没有发生或者被屏蔽。 1 由于下溢异常，因此发出未屏蔽的中断信号。 通过在 SYSEXCIC 寄存器的 FPUFCIC 位写 1 可以清除这个位。
2	FPIOCMIS	RO	0	浮点无效操作屏蔽的中断状态 值 描述 0 中断没有发生或者被屏蔽。 1 由于无效操作，因此发出未屏蔽的中断信号。 通过在 SYSEXCIC 寄存器的 FPIOCIC 位写 1 可以清除这个位。

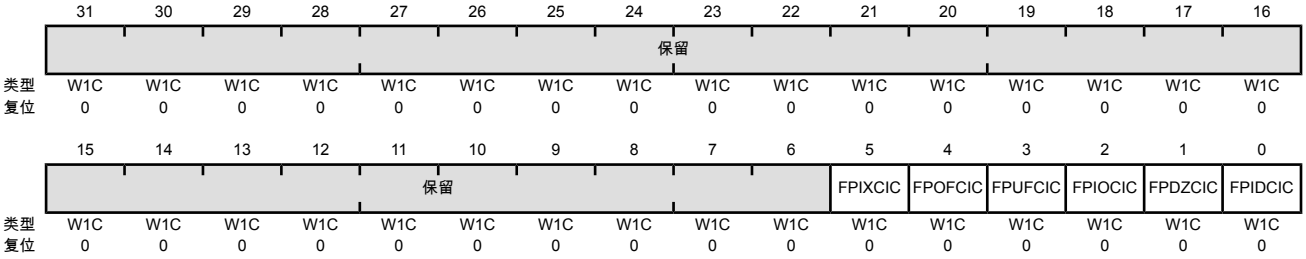
位/域	名称	类型	复位	描述
1	FPDZCMIS	RO	0	浮点除 0 异常屏蔽的中断状态 值 描述 0 中断没有发生或者被屏蔽。 1 由于除 0 异常，因此发出未屏蔽的中断信号。 通过在 SYSEXCIC 寄存器的 FPDZCIC 位写 1 可以清除这个位。
0	FPIDCMIS	RO	0	浮点输入反常异常屏蔽的中断状态 值 描述 0 中断没有发生或者被屏蔽。 1 由于输入反常异常，因此发出未屏蔽的中断信号。 通过在 SYSEXCIC 寄存器的 FPIDCIC 位写 1 可以清除这个位。

寄存器 4: 系统异常中断清零（SYSEXCIC），偏移量 0x00C

SYSEXCIC 寄存器是中断清零寄存器。写入 1 时，相应的中断（包括原始中断，如果启用了屏蔽中断，则还包括屏蔽中断）被清除。写入 0 不起任何作用。

系统异常中断清零 (SYSEXCIC)

基址 0x400F.9000
偏移量 0x00C
类型 W1C, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:6	保留	W1C	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
5	FPIXCIC	W1C	0	浮点不精确异常中断清零 向该位写 1 可将 SYSEXCRIS 寄存器的 FPIXCRIS 位和 SYSEXCMIS 寄存器的 FPIXCMIS 位清零。
4	FPOFCIC	W1C	0	浮点上溢异常中断清零 向该位写 1 可将 SYSEXCRIS 寄存器的 FPOFCRIS 位和 SYSEXCMIS 寄存器的 FPOFCMIS 位清零。
3	FPUFCIC	W1C	0	浮点下溢异常中断清零 向该位写 1 可将 SYSEXCRIS 寄存器的 FPUFCRIS 位和 SYSEXCMIS 寄存器的 FPUFCMIS 位清零。
2	FPIOCIC	W1C	0	浮点无效操作中断清零 向该位写 1 可将清 SYSEXCRIS 寄存器的 FPIOCRIS 位和 SYSEXCMIS 寄存器的 FPIOCMIS 位清零。
1	FPDZCIC	W1C	0	浮点除 0 异常中断清零 向该位写 1 可将 SYSEXCRIS 寄存器的 FPDZCRIS 位和 SYSEXCMIS 寄存器的 FPDZCMIS 位清零。
0	FPIDCIC	W1C	0	浮点输入反常值异常中断清零 向该位写 1 可将 SYSEXCRIS 寄存器的 FPIDCRIS 位和 SYSEXCMIS 寄存器的 FPIDCMIS 位清零。

7 休眠模块

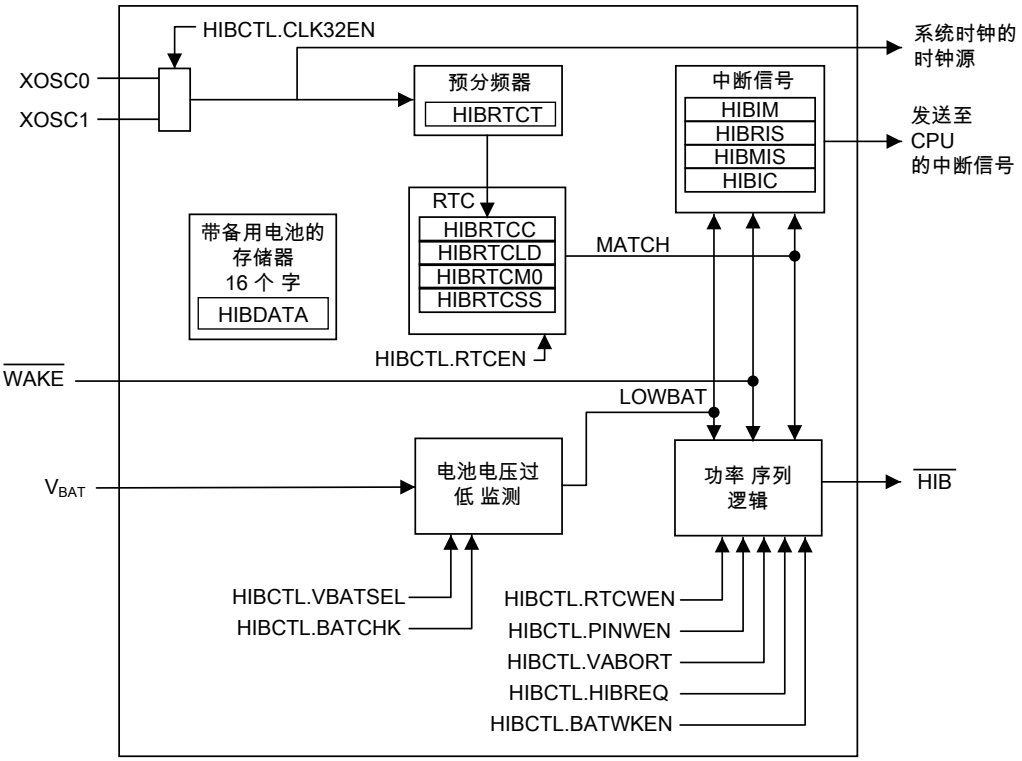
休眠模块用于管理电源的关断和恢复，以便降低系统功耗。当处理器和外设空闲时，电源可以完全关断，只维持休眠模块的供电。电源供电可以利用外部信号触发恢复，也可以利用内置实时时钟 (RTC)，在经过一段特定的时间之后恢复。休眠模块可以由外部电池或者辅助电源单独供电。

休眠模块具有以下特性：

- 32 位实时秒计数器 (RTC)，其时钟分辨率是 1/32,768 秒；以及一个 15 位亚秒计数器
  - 32 位的 RTC 秒匹配寄存器，以及一个 15 位的亚秒匹配寄存器（其时钟分辨率是 1/32,768 秒），用于定时唤醒和产生中断
  - RTC 预分频器调整，对时钟速率进行良好地调节
- 电源控制的两种机制：
  - 使用离散的外部稳压器控制系统电源
  - 使用在寄存器控制下的内部开关控制片上电源
- 使用外部信号用作唤醒的专用管脚
- 只要 V_{DD} 或 V_{BAT} 有效，RTC 运行的内存和休眠所占内存就一直有效
- 电池电量低检测、发出信号和中断发生；在电量低时提供可选的唤醒操作
- GPIO 管脚的状态在休眠过程中可保持不变
- 时钟源可以是 32.768 KHz 的外部晶振或振荡器
- 16 个 32 位字的带备用电池存储器，用于在休眠过程中保存状态
- 可为以下事件设置中断信号：
  - RTC匹配
  - 外部唤醒
  - 电池电量过低

7.1 结构框图

图 **7-1.** 休眠模块结构图



7.2 信号描述

以下表格列出了休眠模块的外部信号，并且分别描述了它们的功能。

表 **7-1.** 休眠 信号 **(64LQFP)**

管脚名称	管脚编号	管脚复用/管脚赋值	管脚类型	缓冲区类型[a]	描述
GNDX	35	固定	-	电源	休眠震荡器的接地。当使用晶振时钟源时，该管脚应连接到数字接地和晶振负载电容。使用外部震荡器时，该管脚应连接到数字接地。
HIB	33	固定	O	TTL	该输出指示处理器是否处于休眠模式。
VBAT	37	固定	-	电源	休眠模块的电源供应源它通常连接到电池的正极端并用作备用电池/休眠模块电源供应器的电源。
WAKE	32	固定	I	TTL	当有效时外部输入将处理器从休眠模式中唤醒。
XOSC0	34	固定	I	模拟	休眠模块晶体振荡器输入或外部时钟参考输入。请注意休眠模块 RTC 采用 32.768-kHz 晶体或 32.768-kHz 振荡器。
XOSC1	36	固定	O	模拟	休眠模块晶体振荡器输出。当使用外部单端参考时钟源时，此管脚应悬空。

a. TTL 表示管脚的电压水平与 TTL 一致。

7.3 功能说明

休眠模块具有两种电源控制机制：

- 第一种机制是在保持 I/O 管脚的电源供电（VDD3ON 模式）时，利用内部开关来控制 Cortex-M4F 以及大多数模拟和数字功能的电源通断。
- 第二种机制是利用控制信号 (HIB) 来控制微处理器的电源。该信号可以控制外部电压稳压器的开启或者关断。

休眠模块的电源采用动态方式确定。休眠模块的电源取以下较大值：主电压源 (V_{DD}) 或者电池/辅助电压源 (V_{BAT})。休眠模块还具有单独的时钟源，当系统时钟断电之后，该时钟源可以继续维持实时时钟 (RTC) 功能。休眠模式可以通过以下两种方法之一进入：

- 用户将休眠控制 (HIBCTL) 寄存器的 HIBREQ 位置位，从而启动休眠
- 应用有效的 V_{BAT} 时，电源接受仲裁，不再使用 V_{DD}。

休眠状态期间，当外部管脚 (WAKE) 被置位或者内部 RTC 达到特定值时，休眠模块就会给外部电源稳压器发出信号，指示恢复系统供电。休眠模块还可以监测电池电压是否过低。如果监测到这种情况，休眠模块就不会进入休眠状态，或者就从休眠状态中唤醒。

从休眠状态中唤醒时，HIB 信号将被清零。恢复 V_{DD} 电压将引起上电复位 (POR)。从 WAKE 信号被置位到代码开始执行的时间就等于唤醒时间 ($t_{WAKE_TO_HIB}$) + 上电复位时间 (T_{POR})。

7.3.1 寄存器访问间隙

因为休眠模块具有单独的时钟域，所以对休眠寄存器进行写入操作时，各次访问之间必须存在时间差。该延迟时间即 $t_{HIB_REG_ACCESS}$，因此软件必须把这个延迟时间插入休眠寄存器的连续写入活动之间，或者插入连续写入和读取活动之间。HIBMIS 寄存器中的 WC 中断用于告知应用程序休眠模块寄存器什么时候可以访问。或者，软件可以利用休眠控制 (HIBCTL) 寄存器中的 WRC 位来确保所需时间差是否已经过去。WRC位表明了软件是否可以开始安全的读或写寄存器。软件将轮询 HIBCTL 以了解 WRC 是否置位，然后再访问休眠寄存器。

连续读取休眠模块寄存器没有时序限制。读取活动将以外设时钟的最高速率进行。

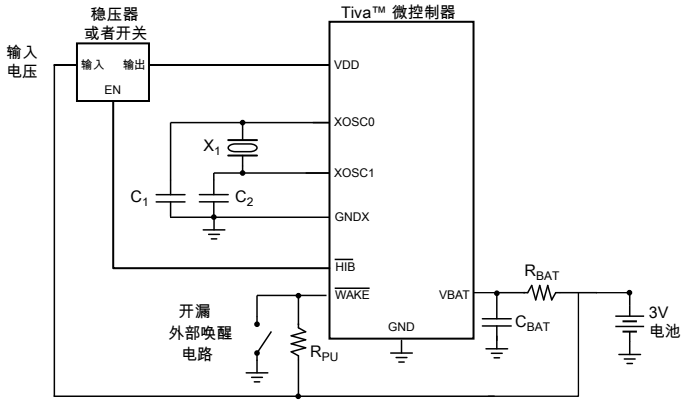
7.3.2 休眠时钟源

在使用休眠模块的系统中，休眠模块必须根据独立于主系统时钟之外的外部时钟源计时，即便没有使用 RTC 功能也是如此。外部振荡器或者晶振可以作为休眠模块的时钟源。要使用晶振，应将 32.768-kHz 晶振连接到 XOSC0 和 XOSC1 管脚上。或者，可以直接在 XOSC0 管脚上连接一个 32.768-kHz 振荡器，而保持 XOSC1 悬空。请注意，32.768-kHz 振荡器的电压振幅一定要小于 V_{BAT}，否则，在休眠状态下，休眠模块就会从振荡器（而不是 V_{BAT}）中取电。请参阅图7-2（432页）和图7-3（432页）。

将 HIBCTL 寄存器中的 CLK32EN 位置位即可启用休眠时钟源。访问任何休眠模块寄存器之前，必须将 CLK32EN 位置位。如果采用晶振作为时钟源，那么在写入 CLK32EN 位之后，软件必须设置 t_{HIBOSC_START} 延迟时间，然后方可访问休眠模块寄存器。晶振将在该延迟时间内加电并稳定下来。如果采用外部振荡器做时钟源，则无需设置延迟时间。使用外部时钟源时，应将 HIBCTL 寄存器的 OSCBYP 位置位。当使用晶振时钟源时，GNDX 管脚应连接到数字接地和晶振负载电容，如图 7-2（432页）所示。使用外部时钟源时，GNDX 管脚应连接到数字接地。

注意: 在下图中，建议将参数 R_{BAT} 和 C_{BAT} 分别设置为 51Ω ±5% 和 0.1µF ±5%。更多信息参见 “Hibernation Module”（1052页）。

图 **7-2.** 使用晶振作为休眠模块的时钟源（单一电池源）



注意: 部分器件可能不提供 GNDX 信号。参见“信号表”（1008页），了解您器件特定的管脚。

X_1 = 晶振频率为 f_{XOSC_XTAL}。

$C_{1,2}$ = 晶振厂家提供的负载电容说明中的电容值。

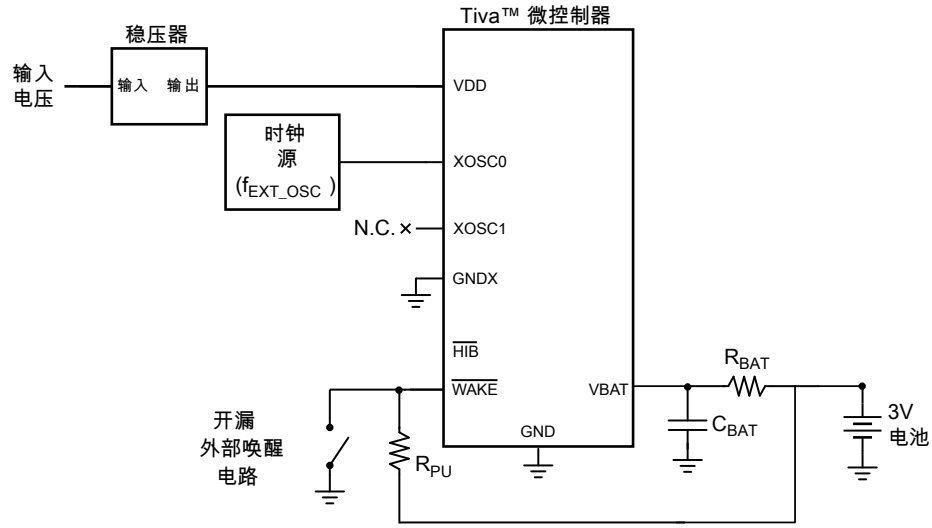
R_{PU} = 上拉电阻为 200 kΩ

R_{BAT} = 51Ω ±5%

C_{BAT} = 0.1µF ±20%

具体参数值请参阅“Hibernation Clock Source Specifications”（1045页）。

图 **7-3.** 在 **VDD3ON** 模式中使用专用振荡器作为休眠模块的时钟源



注意: 部分器件可能不提供GNDX、WAKE或者HIB信号。参见“信号表”（1008页），了解您器件特定的管脚。

R_{PU} = 上拉电阻为 1 MΩ

R_{BAT} = 51Ω ±5%

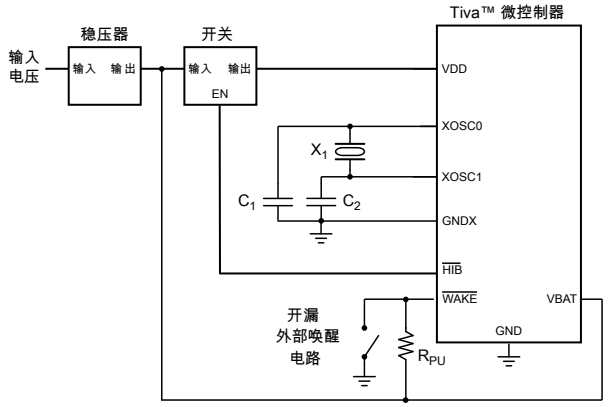
C_{BAT} = 0.1µF ±20%

7.3.3 系统实现

使用休眠模块时可以实现多种不同的系统配置：

- 使用单一电池源，电池提供 V_{DD} 和 V_{BAT}，如 图7-2（432页） 所示。
- 使用VDD3ON模式，V_{DD}在休眠时仍然带电，从而允许GPIO管脚保留其状态，如图7-3（432页）所示。在此模式中，V_{DDC}从内部断电。重新上电时，GPIO保留值将释放，GPIO将初始化为默认值。
- V_{DD} 和 V_{BAT} 使用独立源。在此模式中，我们需要部署其他电路方可在不使用电池或者电池耗尽的情况下启动系统。
- 使用稳压器提供V_{DD}和V_{BAT}，并通过HIB启用开关，从而在休眠时移除V_{DD}，如 图7-4（433页）所示。

图 **7-4. V_{DD} 和 V_{BAT} 使用稳压器**



注意: 部分器件可能不提供 GNDX 信号。参见“信号表”（1008页），了解您器件特定的管脚。

添加外部电容到V_{BAT}源降低了低电池电压测量的准确性，应尽量避免。本节提及的原理图只显示了休眠模块管脚（而非整个系统）的连接。

如果应用程序不要求使用休眠模块，请参考“未用管脚的处理”（1028页）。在这种情况下，必须将运行模式时钟门控控制 0 (RCGC0) 和休眠运行模式时钟门控控制 (RCGCHIB) 寄存器中的 HIB 位清零，以禁用休眠模块的系统时钟信号，同时让休眠模块的寄存器拒绝访问。

7.3.4 电池管理

重要: 系统级问题可能会影响电池电压低监测电路的准确性。设计者应当考虑电池的类型、放电特性以及在电池电压测量过程中的测试负载。

休眠模块可以通过 VBAT 管脚由电池或者辅助电源单独供电。该模块可以监视电池的电压水平，并可检测电池电压何时低于 V_{LOWBAT}。通过 HIBCTL 寄存器的 VBATSEL 域，可将该电压阈值设在 1.9 V 和 2.5 V 之间。用户还可将该模块设置为当电池电压低于此阈值时，系统不进入休眠状态。此外，处于休眠状态时，该模块将检测电池电压。通过 HIBCTL 寄存器的 BATWKEN 位，微控制器可配置为当电池电压低于此阈值时即从休眠状态中唤醒。

休眠模块用于检测低电池电压状态，并在发生此状态时将休眠原始中断状态 (HIBRIS) 寄存器的 LOWBAT 位置位。如果 HIBCTL 寄存器中的 VABORT 位也被置位，那么当检测到电池电压过低

时，模块就不会进入休眠状态。用户还可将该模块设置为当电池电压低于此阈值时，模块即产生中断（请参阅“中断和状态”（437页））。

注意：休眠模块由电压较高的电源（V_{BAT} 或 V_{DD}）供电。因此，必须对电路进行设计，以保证在额定条件下 V_{DD} 的高于 V_{BAT}，否则即使 V_{DD} 可用，休眠模块也只由电池供电。

7.3.5 实时时钟

RTC 模块用于记录实时时间。RTC 能够以秒计数器模式工作。32.768 kHz 时钟源和 15 位预分频器可将时钟频率降至 1 Hz。1 Hz 时钟用于让 32 位计数器递增计数，并记录秒数。可通过配置匹配寄存器将系统从休眠中唤醒或产生中断。此外，用户还可使用软件通过软件修正寄存器对振荡器的误差进行补偿。

7.3.5.1 RTC 计数器 - 秒/亚秒模式

RTC 的时钟信号由休眠模块的其中一个 32.768-kHz 时钟源提供。休眠 RTC 计数器 (HIBRTCC) 寄存器可显示秒数。休眠 RTC 亚秒 (HIBRTCSS) 寄存器可为需要低于一秒分频的应用提供更高的时间精度。

置位 HIBCTL 寄存器的 RTCEN 位即可启用 RTC。RTC 计数器和亚秒计数器在 RTCEN 位置位后立即开始计数。两个计数器都采用递增计数模式运行。只要启用了 RTC 并存在有效的 V_{BAT}，RTC 就会继续计数，而不论 V_{DD} 是否存在或该部件是否处于休眠模式。

向休眠 RTC 加载 (HIBRTCLD) 寄存器执行写入操作即可置位 HIBRTCC 寄存器。写入 HIBRTCLD 寄存器即可将 HIBRTCSS 寄存器的 15 位亚秒计数器域 RTCSSC 清零。要确保有效读取 RTC 值，应首先读取 HIBRTCC 寄存器，然后读取 HIBRTCSS 寄存器的 RTCSSC 域，再重新读取 HIBRTCC 寄存器。如果 HIBRTCC 寄存器两次读取的数值一样，则读取有效。按照这个程序，可以防止 HIBRTCC 寄存器因 RTCSSC 域读为 1 而重新计数所造成的应用出错。将 HIBIM 寄存器的 RTCAL0 位置位即可让 RTC 生成警报。发生 RTC 匹配时，系统将生成中断并显示在 HIBRIS 寄存器中。请参阅 “RTC 匹配 - 秒/亚秒模式”（434页）以了解更多信息。

启用 RTC 后，仅冷 POR 可复位 RTC 寄存器。在冷 POR 中，V_{BAT} 和 V_{DD} 被移除。如果在启用 RTC 时发生其他类型的复位，例如外部 $\overline{RST}$ 置位或者 BOR 复位，则 RTC 不会复位。只要不启用 RTC 和外部唤醒管脚，任何类型的系统复位都可将 RTC 寄存器复位。

7.3.5.2 RTC 匹配 - 秒/亚秒模式

休眠模块包含一个 32 位匹配寄存器 HIBRTCM0，可用于与 RTC 32 位计数器 HIBRTCC 的值进行比较。这个匹配寄存器也适用于亚秒计数器。HIBRTCSS 寄存器的 15 位域 (RTCSSM) 将与 15 位亚秒计数器的值对比。发生匹配时，HIBRIS 寄存器的 RTCALT0 位将置位。对于使用休眠模式的应用，处理器可设置为在 HIBCTL 寄存器的RTCWEN 位置位时从休眠中唤醒。处理器还可设置为在 HIBIM 寄存器的 RTCALT0 位置位时生成中断，并发送给中断控制器。

匹配中断的产生优先于中断清零。因此，如果 HIBRTCC 和 HIBRTCM0 的值相等，那么对休眠中断清除 (HIBIC) 寄存器的 RTCALT0 位的写入操作不会将 RTCALT0 位清零。可通过多种方法避免发生此事，例如在写入 HIBIC 之前向 HIBRTCLD 寄存器写入新值，以将 RTCALT0 清零。再如，将 HIBCTL 寄存器的 RTCEN 位清零然后置位，以禁用并重新启用 RTC。

注意: 在匹配事件有效时发出休眠请求将立即唤醒模块。当 RTCWEN 位置位且 HIBRIS 寄存器的 RTCALT0 位置位，同时 HIBCTL 寄存器的 HIBREQ 位写 1 时即发生此情况。要避免这种情况，可以在将 HIBREQ 位置位之前，通过向 HIBIC 寄存器相应位写 1 来清零 HIBRIS 的 RTCAL0 位。再如，将 HIBCTL 寄存器的 RTCEN 位清零然后置位，以禁用并重新启用 RTC。

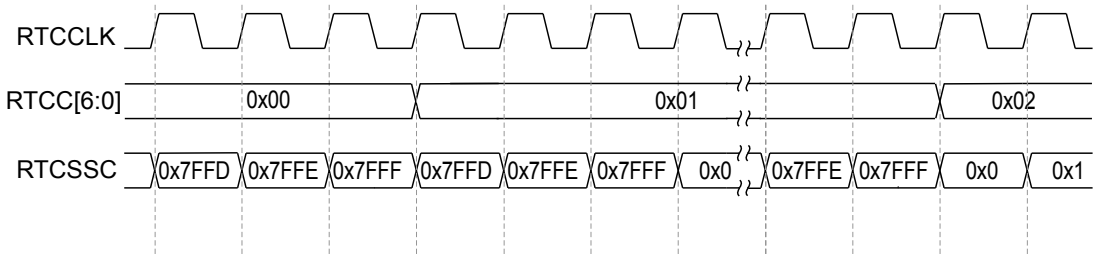
7.3.5.3 RTC 修正

预分频器修正寄存器 HIBRTCT 可以用来补偿时钟源的误差。该寄存器的标称值为 0x7FFF。当 HIBRTCC 寄存器中的 [5:0] 位由 0x00 变成 0x01 时，在 RTC 计数器模式中，该默认值每隔 64 秒就会调整一次，以便将输入时钟分频。通过此配置，软件可以调整预分频器修正寄存器的数值（使其大于或者小于 0x7FFF），以便精确修正时钟速率。要降低 RTC 时钟速率，就要让预分频器修正寄存器的值大于 0x7FFF；要增加 RTC 时钟速率，就要让预分频器修正寄存器的值小于 0x7FFF。

使用接近 HIBRTCSS 寄存器亚秒匹配值的修正值时，必须小心谨慎。使用大于 0x7FFF 的修正值时，可能导致同一计数器值发生两个匹配中断。此外，使用低于 0x7FFF 的修正值时，可能丢失匹配中断。

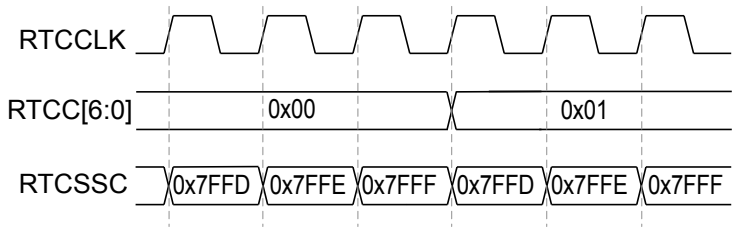
修正值大于 0x7FFF 时，如果 HIBRTCSS 寄存器的 RTCSSC 值达到 0x7FFF，则 RTCC 值从 0x0 增加到 0x1，而 RTCSSC 值要减去修正量。RTCSSC 值在重新回到 0x0 并开始下次递增计数前，先会再次递增计数至 0x7FFF。如果匹配值在该范围以内，匹配中断将触发两次。例如，如 图 7-5（435页）所示，如果 RTCM0=0x1 且 RTCSSM=0x7FFD 时配置匹配中断，将触发两个中断。

图 **7-5. TRIM** 值为 **0x8002** 时的计数器行为



修正值小于 0x7FFF 时，RTCSSC 值从 0x7FFF 增加到修正值，而 RTCC 值从 0x0 增加到 0x1。如果匹配值在该范围以内，匹配中断将不会触发。例如，如 图7-6（435页）所示，如果RTCM0=0x1 且 RTCSSM=0x2 时配置匹配中断，绝不会触发中断。

图 **7-6. TRIM** 值为 **0x7FFC** 时的计数器行为



7.3.6 带备用电池的存储器

休眠模块包含 16 个 32 位字的存储器，并采用电池或者辅助电源供电，因此它们在休眠期间也会保持通电。处理器软件可以在休眠之前将状态信息保存在该存储器中，并在唤醒时恢复。用户可以通过 HIBDATA 寄存器访问带备用电池的存储器。如果 V_{DD} 和 V_{BAT} 都被移除，HIBDATA 寄存器中的内容将丢失。

7.3.7 电源控制：使用 HIB

重要: 使用 HIB 控制电源时，需要对休眠模块进行特殊的系统设置，因为它会试图关断微控制器所有其他部分的电源。所有连接到芯片的系统信号和电源都必须驱动输出至 0 V，或者通过由 HIB 控制的相同稳压器关闭电源。

休眠模块利用 HIB 管脚控制微控制器的电源。该管脚会连接到为微控制器和其他电路提供 3.3 V 电源的外部稳压器的启用信号端。当 HIB 信号被休眠模块置位以后，外部稳压器将关闭，而不再为微处理器和其他原本由外部稳压器供电的系统部位供电。但是，休眠模块仍然从 V_{BAT} 电源获得电能，直到发生唤醒事件。将 HIB 清零即可恢复微处理器的供电，进而使外部稳压器恢复为芯片供电。

7.3.8 以 VDD3ON 模式管理电源

在休眠模式中，休眠模块还可以用来切断芯片内部模块的电源。处于此状态时，如果 HIBCTL 寄存器的 VDD3ON 位置位，那么所有管脚都会保持休眠之前的状态。举个例子，输入管脚还是输入功能；输出驱动高电平仍保持驱动高电平，等等。在 VDD3ON 模式中有一些需要注意的重要程序和功能项目：

- 在 VDD3ON 模式中，稳压器应将休眠模式下的微控制器电源保持在 3.3 V。当 HIBCTL 寄存器的 RETCLR 位清零时，GPIO 保留将被禁用。

7.3.9 启动休眠

当 HIBCTL 寄存器中的 HIBREQ 位被置位，休眠功能就会启动。如果尚未使用 HIBCTL 寄存器的 PINWEN 或 RTCWEN 位配置唤醒条件，休眠请求将被忽略。如果 HIBREQ 位置位时正在进行 Flash 存储器写入操作，那么互锁功能就会延迟系统进入休眠模式，直到写入操作完成。此外，如果电池电压低于 HIBCTL 寄存器 VBATSEL 域设置的阈值电压，休眠请求将被忽略。

7.3.10 从休眠模式唤醒

通过将 HIBCTL 寄存器中的 PINWEN 位置位，休眠模块就会被外部 WAKE 管脚唤醒。将 RTCWEN 位置位，休眠模块就会被 RTC 匹配唤醒。请注意，WAKE 管脚使用休眠模块的内部电压作为逻辑 1 的参考电压。

休眠模块也可配置为在发生以下事件时从休眠中唤醒：

- RTC 匹配唤醒事件
- 电池电压低唤醒事件

将 HIBCTL 寄存器的 RTCWEN 位置位后，当 HIBRTCC 寄存器的值与 HIBRTCM0 寄存器的值匹配，并且 RTCSSC 域的值与 HIBRTCSS 寄存器中 RTCSSM 域的值匹配时，系统即可从休眠中唤醒。

要在发生电池电压低事件时让系统从休眠中唤醒，必须将 HIBCTL 寄存器的 BATWKEN 位置位。根据该配置，在休眠期间，系统每 512 秒就会检查一次电池电压。如果电压低于 VBATSEL 域指定的电压值，HIBRIS 寄存器中的 LOWBAT 中断位就会被置位。

在发生外部唤醒、外部复位、或者 RTC 匹配时，休眠模块的唤醒将延迟，直到 V_{DD} 高于指定的最小电压值（请参阅表21-5（1031页））。

当休眠模块被唤醒时，微控制器会进行正常的上电复位。该复位不会重置休眠模块，但会重置微控制器的其他部分。通过观察中断状态寄存器（请参阅“中断和状态”（437页））以及查询带备用电池的存储器中的状态数据（请参阅“带备用电池的存储器”（435页）），软件可以监测到休眠唤醒上电。

7.3.11 仲裁性电源移除

在 CLK32EN 位和以下任何一个位置位时，如果 V_{DD} 被仲裁移除，微控制器会进入休眠：

- HIBCTL 寄存器中的 PINWEN 位
- HIBCTL 寄存器中的 RTCEN 位

重新上电时，微控制器从休眠状态唤醒。

如果 CLK32EN 位已置位但 PINWEN 和 位均清零，微控制器仍然会在掉电时进入休眠模式，但当重新接通电压 V_{DD} 时，MCU 将执行冷上电复位，且休眠模块将复位。如果 CLK32EN 位没有置位且 V_{DD} 被仲裁移除，该部件只是关机并在重新上电时执行冷上电复位。

如果仲裁移除 V_{DD} 时正在进行 Flash 存储器或 HIBDATA寄存器写入操作，必须在重新应用 V_{DD} 时重试此写入操作。

7.3.12 中断和状态

当以下条件出现时，休眠模块可以产生中断：

- $\overline{\text{WAKE}}$ 管脚有效确认
- RTC匹配
- 检测到低电池电压
- 写入完成/可以写入
- 外部 $\overline{\text{RESET}}$ 管脚有效确认
- 对唤醒启用的外部 GPIO 管脚（端口）的有效确认

所有的中断信号在相或后被发送到中断控制器，因此在给定的时间，休眠模块只能向控制器产生一个中断请求。软件中断处理器可以通过读取 休眠屏蔽中断状态 (HIBMIS) 寄存器来处理多个中断事件。软件还可以通过访问 HIBRIS 寄存器，随时读取休眠模块的状态。该寄存器会显示所有挂起的事件。从休眠模式中唤醒后，该寄存器可以用来判断唤醒条件是上述事件之一还是电源断开。

通过设置休眠中断屏蔽 (HIBIM) 寄存器中相应的位，即可配置将触发中断信号的事件。通过修改休眠中断清除 (HIBIC) 寄存器中相应的位，即可清除挂起的中断。

7.4 初始化和配置

休眠模块有几种不同的配置。下面各小节说明了不同情况下推荐的编程步骤。因为休眠模块以较低频率运行，并与根据系统时钟运行的微控制器的其他部分同步，因此软件在写入寄存器（请参阅“寄存器访问间隙”（431页））之后，必须存在一个延迟时间 $t_{HIB_REG_ACCESS}$。HIBMIS 寄存器中的 WC 中断用于告知应用程序休眠模块寄存器什么时候可以访问。

7.4.1 初始化

休眠模块启用系统时钟即可完成复位过程，但是如果模块的系统时钟被禁用，那么必须重新启用系统时钟，即便不使用 RTC 功能。请参阅298页。

如果使用 32.768-kHz 晶振作为休眠模块的时钟源，请遵循以下步骤：

1. 向 HIBIM 寄存器写入 0x0000.0010，以便启用 WC 中断。
2. 向 HIBCTL 寄存器写入 0x40（偏移量 0x10）以启用振荡器输入。

3. 等待 HIBMIS 寄存器中的 WC 中断被触发以后，然后再对休眠模块进行其他操作。

如果使用单端 32.768-kHz 振荡器作为休眠模块时钟源，请遵循以下步骤：

1. 向 HIBIM 寄存器写入 0x0000.0010，以便启用 WC 中断。
2. 在 HIBCTL 寄存器中写入 0x0001.0040（偏移量 0x10）以启用振荡器输入，并绕过芯片内部的振荡器。
3. 等待 HIBMIS 寄存器中的 WC 中断被触发以后，然后再对休眠模块进行其他操作。

以上步骤只需要在整个系统第一次初始化的时候执行。如果微控制器已处于休眠状态，那么休眠模块已经上电，无需执行以上步骤。软件可以通过观察 HIBCTL 寄存器中的 CLK32EN 位监测休眠模块和时钟是否已经上电。

表7-2（438页） 说明了在正常运行以及休眠状态下，位的各种设置与时钟功能的关系。

表 **7-2.** 休眠模块的时钟运行

CLK32EN	PINWEN	RTCWEN	RTCEN	正常运行	休眠
0	X	X	X	休眠模块禁用	休眠模块禁用
1	0	0	1	RTC 匹配功能启用。	无休眠
1	0	1	1	模块计时	将 RTC 匹配作为唤醒事件
1	1	0	0	模块计时	休眠期间时钟模块断电；外部唤醒事件发生时，时钟模块上电。
1	1	0	1	模块计时	休眠期间时钟模块依旧供电，以实现 RTC 功能。根据外部事件唤醒。
1	1	1	1	模块计时	RTC 匹配或者外部唤醒事件，以先发生者为准。

7.4.2 RTC 匹配功能（无休眠）

请按照以下步骤配置休眠模块的 RTC 匹配功能：

1. 在 HIBCTL 寄存器中写入 0x0000.0040（偏移量 0x010）以启用 32.768-kHz 休眠振荡器。
2. 在 HIBRTCM0 寄存器中写入所需的 RTC 匹配值（偏移量 0x004）；并在 HIBRTCSS 寄存器的 RTCSSM 域中写入该 RTC 匹配值（偏移量 0x028）。
3. 将要求的 RTC 加载值写入 HIBRTCLD 寄存器（偏移量 0x00C）。
4. 在 HIBIM 寄存器中的 RTCALT0 写入所需的 RTC 匹配中断掩码（偏移量 0x014）。
5. 在 HIBCTL 寄存器中写入 0x0000.0041（偏移量 0x010）让 RTC 开始计数。

7.4.3 RTC 匹配/唤醒

请使用以下步骤实现 RTC 匹配和休眠模块的唤醒功能：

1. 在 HIBCTL 寄存器中写入 0x0000.0040（偏移量 0x010）以启用 32.768-kHz 休眠振荡器。
2. 在 HIBRTCM0 寄存器中写入所需的 RTC 匹配值（偏移量 0x004）；并在 HIBRTCSS 寄存器的 RTCSSM 域中写入该 RTC 匹配值（偏移量 0x028）。
3. 将要求的 RTC 加载值写入 HIBRTCLD 寄存器（偏移量 0x00C）。该写入会将 15 位亚秒计数器清零。

4. 在 HIBDATA 寄存器中写入需要在休眠的时候保留的任意数据（偏移量 0x030-0x06F）。

5. 在 HIBCTL 寄存器中写入 0x0000.004B，偏移量 0x010，以设置外部唤醒功能，并开始休眠序列。

7.4.4 外部唤醒

如果要使用外部 WAKE 管脚作为微控制器的唤醒源，请遵循以下步骤：

1. 在 HIBCTL 寄存器中写入 0x0000.0040（偏移量 0x010）以启用 32.768-kHz 休眠振荡器。

2. 在 HIBDATA 寄存器中写入需要在休眠的时候保留的任意数据（偏移量 0x030-0x06F）。

3. 在 HIBCTL 寄存器中写入 0x0000.0052（偏移量 0x010），以启用外部唤醒功能，并开始休眠序列。

7.4.5 RTC 或外部唤醒

1. 在 HIBCTL 寄存器中写入 0x0000.0040（偏移量 0x010）以启用 32.768-kHz 休眠振荡器。

2. 在 HIBRTCM0 寄存器中写入所需的 RTC 匹配值（偏移量 0x004）；并在 HIBRTCSS 寄存器的 RTCSSM 域中写入该 RTC 匹配值（偏移量 0x028）。

3. 将要求的 RTC 加载值写入 HIBRTCLD 寄存器（偏移量 0x00C）。该写入会将 15 位亚秒计数器清零。

4. 在 HIBDATA 寄存器中写入需要在休眠的时候保留的任意数据（偏移量 0x030-0x06F）。

5. 在 HIBCTL 寄存器中写入 0x0000.005B，偏移量 0x010，以设置 RTC 匹配/外部唤醒功能，并开始休眠序列。

7.5 寄存器映射

表 7-3列出了休眠寄存器的信息。所有给出的地址都是相对于休眠模块基址 0x400F.C000而言的。请注意，对寄存器编程之前，必须启用休眠模块的系统时钟（请参阅298页）。启用休眠模块时钟之后，必须经过 3 个系统时钟的延迟才能访问休眠模块的寄存器。另外，在访问其他任何休眠模块寄存器之前，必须将 HIBCTL 寄存器的 CLK32EN 位置位。

注意: 休眠模块寄存器处于休眠模块的时钟域内，具有特殊的时序要求。软件应该利用 HIBCTL 寄存器中的 WRC 位来确保所需的时间差是否已经过去。如果 WRC 位被清零，那么任何写入访问都不起作用。请参阅“寄存器访问间隙”（431页）。

重要: 休眠模块寄存器在两种情况下会复位：

1. 任何类型的系统复位可令防篡改模块复位（前提是 HIBCTL 寄存器的 RTCEN 和 PINWEN 位清零）。

2. 当 V_{DD} 和 V_{BAT} 电源被移除时发生冷 POR 也可使之复位。

任何其他复位条件都会被休眠模块忽略。

表 **7-3.** 休眠模块 寄存器映射

偏移量	名称	类型	复位	描述	见页面
0x000	HIBRTCC	RO	0x0000.0000	休眠 RTC 计数器寄存器	441

表 **7-3.** 休眠模块 寄存器映射（续）

偏移量	名称	类型	复位	描述	见页面
0x004	HIBRTCM0	R/W	0xFFFF.FFFF	休眠 RTC 匹配寄存器 0	442
0x00C	HIBRTCLD	R/W	0x0000.0000	休眠 RTC 加载寄存器	443
0x010	HIBCTL	R/W	0x8000.2000	休眠控制寄存器	444
0x014	HIBIM	R/W	0x0000.0000	休眠中断屏蔽寄存器	448
0x018	HIBRIS	RO	0x0000.0000	休眠原始中断状态寄存器	450
0x01C	HIBMIS	RO	0x0000.0000	休眠屏蔽中断状态寄存器	452
0x020	HIBIC	R/W1C	0x0000.0000	休眠中断清除寄存器	454
0x024	HIBRTCT	R/W	0x0000.7FFF	休眠 RTC 修正寄存器	455
0x028	HIBRTCSS	R/W	0x0000.0000	休眠 RTC 亚秒寄存器	456
0x030-0x06F	HIBDATA	R/W	-	休眠数据寄存器	457

7.6 寄存器描述

本节的剩余部分按照地址偏移量的数字顺序列出和描述了休眠模块寄存器。

寄存器 1: 休眠 RTC 计数器寄存器（HIBRTCC），偏移量 0x000

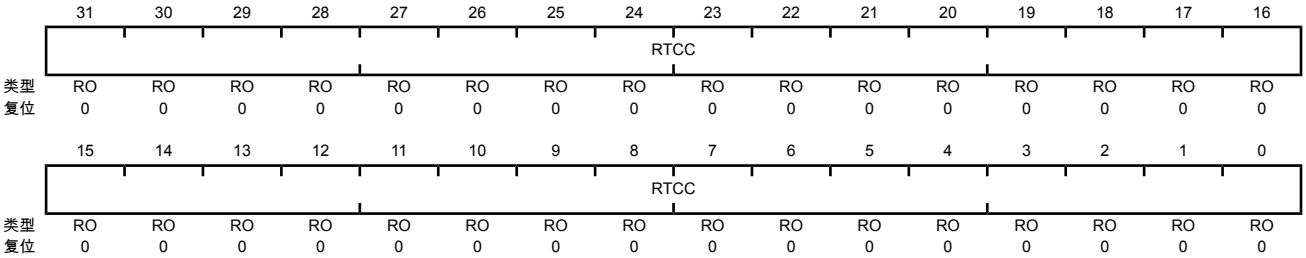
该寄存器是RTC计数器的当前32位值。

RTC 计数器由一个 32 位秒计数器和一个 15 位亚秒计数器组成。休眠模块复位的同时，RTC 计数器也会复位。用户可以通过 HIBRTCLD 寄存器将 RTC 32 位秒计数器置位。当 32 位秒计数器被置位 后，15 位亚秒计数器就会被清零。

要得到 RTC 值，可以首先读取 HIBRTCC 寄存器，再读取 RTCSSC 寄存器的 HIBRTCSS 域，然后再次读取 HIBRTCC 寄存器。如果 HIBRTCC 寄存器两次读取的数值一样，则读取有效。

休眠 RTC 计数器寄存器 (HIBRTCC)

基址 0x400F.C000
偏移量 0x000
类型 RO, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:0	RTCC	RO	0x0000.0000	RTC 计数器 读取返回 32 位计数器数值。因为 RTC 已经启用，所以该数值代表延迟的秒数。该寄存器为只读状态。使用 HIBRTCLD 寄存器可改变该值。

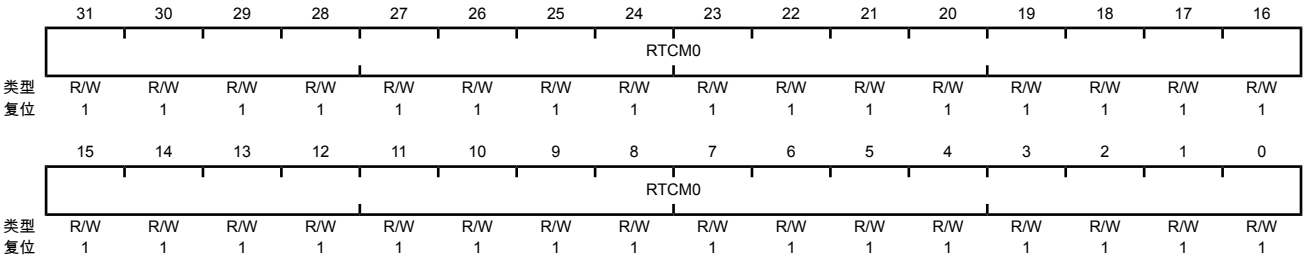
寄存器 2: 休眠 RTC 匹配寄存器 0（HIBRTCM0），偏移量 0x004

该寄存器是 RTC 计数器的一个 32 位秒匹配寄存器。15 位亚秒匹配数值存储在 HIBRTCSS 寄存器的 RTCSSC 域中，它可以与该寄存器配合使用，以得到更精确的时间匹配。

注意: 休眠模块寄存器处于休眠模块的时钟域内，具有特殊的时序要求。软件应该利用 HIBCTL 寄存器中的 WRC 位来确保所需的时间差是否已经过去。如果 WRC 位被清零，那么任何写入访问都不起作用。请参阅“寄存器访问间隙”（431页）。

休眠 RTC 匹配寄存器 0 (HIBRTCM0)

基址 0x400F.C000
偏移量 0x004
类型 R/W, 复位 0xFFFF.FFFF



位/域	名称	类型	复位	描述
31:0	RTCM0	R/W	0xFFFF.FFFF	RTC 匹配 0 将数值写入 RTC 匹配寄存器。 读时返回当前的匹配值。

寄存器 3: 休眠 RTC 加载寄存器（HIBRTCLD），偏移量 0x00C

该寄存器用来向 RTC 计数器加载一个 32 位数值。当该寄存器被写入时，加载立即发生。当该寄存器写入数值时，15 位亚秒计数器也会被清零。

注意: 休眠模块寄存器处于休眠模块的时钟域内，具有特殊的时序要求。软件应该利用 HIBCTL 寄存器中的 WRC 位来确保所需的时间差是否已经过去。如果 WRC 位被清零，那么任何写入访问都不起作用。请参阅“寄存器访问间隙”（431页）。

休眠 RTC 加载寄存器 (HIBRTCLD)

基址 0x400F.C000
偏移量 0x00C
类型 R/W, 复位 0x0000.0000

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
域	RTCLD															
类型	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
域	RTCLD															
类型	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位/域	名称	类型	复位	描述
31:0	RTCLD	R/W	0x0000.0000	RTC 加载 写入活动会将当前值加载到 RTC 计数器 (RTCC) 中。 读时返回32位的装载值。

寄存器 4: 休眠控制寄存器（HIBCTL），偏移量 0x010

这个寄存器是休眠模块的控制寄存器。在发生休眠事件之前，该寄存器必须最后写入。如果在将 HIBREQ 位置位之后再写入其他寄存器，那么就不能确保在休眠发生之前完成数据的写入。

注意: 此寄存器的写入操作有特殊的时间要求。软件应利用 HIBCTL 寄存器中的 WRC 位来确保所需的同步已经发生。如果 WRC 位清零，那么对此寄存器的任何写入都将被忽略。可随时读取。

休眠控制寄存器 (HIBCTL)

基址 0x400F.C000
偏移量 0x010
类型 R/W, 复位 0x8000.2000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	WRC	保留													OSCDRV	OSCBYP
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W
复位	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	保留	VBATSEL		保留		BATCHK	BATWKEN	VDD3ON	VABORT	CLK32EN	保留	PINWEN	RTCWEN	保留	HIBREQ	RTCEN
类型	RO	R/W	R/W	RO	RO	R/W	R/W	R/W	R/W	R/W	RO	R/W	R/W	RO	R/W	R/W
复位	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0

位/域	名称	类型	复位	描述
31	WRC	RO	1	写入完成/可以写入 值 描述 0 该接口正在处理优先级高的写入，处于繁忙状态。当 WRC 是 0 时，任何写入操作都会导致不确定的行为。 1 该接口可以接受写入。 软件必须在写入请求之间查询该位，并延迟写入活动直到 WRC=1，以确保操作正确。可配置一个中断来表明 WRC 已完成。 该位的名称 WRC 表示“写入完成”，是该位的默认用法(写入访问之间)。然而，因为该位无需复位即可置位，所以这个名称还表示“可以写入”，说明该接口可以通过软件写入数值。软件可以利用这个区别在复位时监测哪种编程比较合适：0 = 需要软件延迟循环；1 = WRC 节奏可用。
30:18	保留	RO	0x000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
17	OSCDRV	R/W	0	振荡器驱动能力 该位用来补偿过大或者过小的滤波电容。 注意: 休眠振荡器开始之后，该位不应该被改变。振荡器运行时，如果用户改变了这个值，振荡器可能会不稳定。 值 描述 0 低电平驱动强度已启用，12 pF。 1 高电平驱动强度已启用，24 pF。

位/域	名称	类型	复位	描述
16	OSCBYP	R/W	0	振荡器旁路 值 描述 0 内部 32.768-kHz 休眠振荡器已启用。当使用外部 32.768-kHz 晶振时，该位应该被清零。 1 内部 32.768-kHz 休眠振荡器已禁用，并且已断电。当使用连接到 XOSC0 的单端振荡器时，该位应该置位。
15	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
14:13	VBATSEL	R/W	0x1	选择低电池电压比较器 该域确定检查电池状态时所用的电池电压值。如果电池电压低于指定值，HIBRIS 寄存器中的 LOWBAT 中断位就会被置位。 值 描述 0x0 1.9 V 0x1 2.1 V（默认） 0x2 2.3 V 0x3 2.5 V
12:11	保留	RO	0x0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
10	BATCHK	R/W	0	检查电池状态 值 描述 0 读取到该值表示低电池电压比较器周期没有激活。 写入 0 未生效。 1 读取到该值则表示低电池电压比较器周期尚未完成。 将此位置位即可启动低电池电压比较器周期。如果电池电压低于 VBATSEL 域指定的电压值，HIBRIS 寄存器中的 LOWBAT 中断位就会被置位。电池检测过程中，休眠请求会被延迟。
9	BATWKEN	R/W	0	电池电压过低唤醒 值 描述 0 模块不会自动检查电池电压。电池电压过低不会将微处理器从休眠状态中唤醒。 1 如果该位置位，在休眠期间，系统每 512 秒就会检查一次电池电压。 如果电压低于 VBATSEL 域指定的电压值，微控制器就会从休眠状态唤醒，HIBRIS 寄存器中的 LOWBAT 中断位会被置位。

位/域	名称	类型	复位	描述
8	VDD3ON	R/W	0	VDD 上电 值 描述 0 内部开关没有使用。HIB 信号应该用来控制外部开关或者稳压器。 1 内部开关控制芯片模块的电源（VDD3ON 模式）。 请注意，在休眠模式下，不管 VDD3ON 位的状态如何，HIB 信号都会被置位。因此，当 VDD3ON 置位时，HIB 信号不得连接至 3.3V 稳压器，而 3.3V 电源应该继续保持连接。如果此位在休眠中置位，所有管脚都将保持进入休眠前的状态。举个例子，输入管脚还是输入功能；输出驱动高电平仍保持驱动高电平，等等。
7	VABORT	R/W	0	启用电源切断终止 值 描述 0 不管电池电压处于何种水平，微控制器都会进入休眠状态。 1 该位被置位 之后，模块会在进入休眠之前检查电池电压。如果 V_{BAT} 低于 VBATSEL 指定的电压值，那么微控制器不会进入休眠状态。
6	CLK32EN	R/W	0	启用时钟 要使用休眠模块，必须启用该位。 值 描述 0 休眠模块时钟源已禁用。 1 休眠模块时钟源已启用。
5	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
4	PINWEN	R/W	0	外部 WAKE 管脚启用 值 描述 0 WAKE 管脚状态对休眠没有影响。 1 WAKE 管脚高电平会把微控制器从休眠状态唤醒。
3	RTCWEN	R/W	0	RTC 唤醒启用 值 描述 0 RTC 匹配事件对休眠没有影响。 1 RTC 匹配事件（HIBRTCC 寄存器的值与 HIBRTCM0 寄存器的值匹配，而且 RTCSSC 域的值与 HIBRTCSS 寄存器中 RTCSSM 域的值匹配）会将微控制器从休眠状态中唤醒。
2	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。

位/域	名称	类型	复位	描述
1	HIBREQ	R/W	0	休眠请求 值 描述 0 无休眠请求。 1 将该位置位，启动休眠。 唤醒事件之后，该位会被硬件自动清零。 如果 PINWEN 和 RTCWEN 位都已清零，将会忽略休眠请求。 如果 BATCHK 位被置位，那么休眠请求就会被推迟。
0	RTCEN	R/W	0	RTC 计时器启用 值 描述 0 休眠模块 RTC 功能已禁用。 1 休眠模块 RTC 功能已启用。

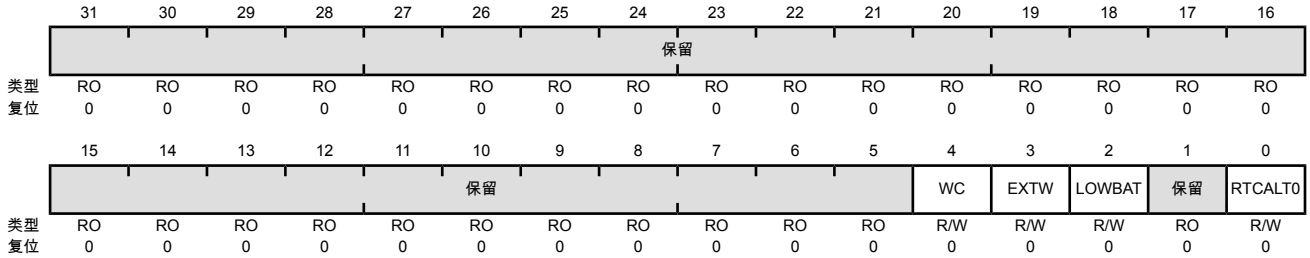
寄存器 5: 休眠中断屏蔽寄存器（HIBIM），偏移量 0x014

这个寄存器是休眠模块中断源的中断屏蔽寄存器。该寄存器中的每一位都可以屏蔽休眠原始中断状态 (HIBRIS) 寄存器中相应的位。如果该位没有被屏蔽，那么中断信号就会发送到中断控制器。如果该位被屏蔽，那么中断信号就不会发送到中断控制器。可先将 HIBIM 寄存器的 WC 位置位，再将 HIBCTL 寄存器的 CLK32EN 位置位。从而软件可以通过 WC 中断触发信号检测 RTCOSC 时钟于何时变得稳定（这一时间可能超过 1 秒）。在 CLK32EN 位置位之前，如果 WC 位被置位，那么该掩码不会在休眠周期中保留，除非再将该位写入一次。

注意: 该寄存器的WC位位于系统时钟域内，因而对该位的写入操作能及时生效，并能在HIBCTL寄存器的 CLK32EN 位置位之前完成。

休眠中断屏蔽寄存器 (HIBIM)

基址 0x400F.C000
偏移量 0x014
类型 R/W, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:5	保留	RO	0x0000.000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
4	WC	R/W	0	外部写入完成/可以中断屏蔽 值 描述 0 WC 中断被抑制，不会发送到中断控制器。 1 当 HIBRIS 寄存器中的 WC 位被置位 时，中断信号就会发送到中断控制器。
3	EXTW	R/W	0	外部唤醒中断屏蔽 值 描述 0 EXTW 中断被抑制，不会发送到中断控制器。 1 当 HIBRIS 寄存器中的 EXTW 位被置位 时，中断信号就会发送到中断控制器。
2	LOWBAT	R/W	0	电池电压过低中断屏蔽 值 描述 0 LOWBAT 中断被抑制，不会发送到中断控制器。 1 当 HIBRIS 寄存器中的 LOWBAT 位被置位 时，中断信号就会发送到中断控制器。
1	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。

位/域	名称	类型	复位	描述
0	RTCALT0	R/W	0	RTC 警报 0 中断屏蔽

值 描述

0 RTCALT0 中断被抑制，不会发送到中断控制器。

1 当 HIBRIS 寄存器中的 RTCALT0 位被置位 时，中断信号就会发送到中断控制器。

寄存器 6: 休眠原始中断状态寄存器（HIBRIS），偏移量 0x018

这个寄存器是休眠模块中断源的原始中断状态。通过将 HIBIM 寄存器中相应的位清零，即可屏蔽所有位。当某一位被屏蔽时，中断信号不会发送到中断控制器。通过将 1 写入休眠中断清除 (HIBIC) 寄存器中相应的位或进入休眠，即可将该寄存器中的所有位清零。

注意: 由于 V_{DD} 仲裁电源丢失，该寄存器的这些位不会指示休眠状态.如果 LOWBAT 位在电源丢失前置位，当电源恢复时，它将保持置位状态。此外，EXTW 位将在离开休眠状态时自动清零，因此如果此位在电源丢失前置位，相关事件将在电源恢复后丢失。

休眠原始中断状态寄存器 (HIBRIS)

基址 0x400F.C000
偏移量 0x018
类型 RO, 复位 0x0000.0000

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	保留															
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	保留											WC	EXTW	LOWBAT	保留	RTCALT0
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位/域	名称	类型	复位	描述
31:5	保留	RO	0x0000.000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
4	WC	RO	0	写入完成/可以读取原始中断状态 值 描述 0 HIBCTL 寄存器中的 WRC 位尚未置位。 1 HIBCTL 寄存器中的 WRC 位已经被置位。 在 HIBIC 寄存器中的 WC 位写入 1 即可将该位清零。
3	EXTW	RO	0	外部唤醒原始中断状态 注意：WAKE 信号将在休眠模块记录中断后清零。 值 描述 0 WAKE 管脚没有被置高电平。 1 WAKE 管脚已被置高电平。 在 HIBIC 寄存器中的 EXTW 位写入 1 即可将该位清零。
2	LOWBAT	RO	0	低电池电压原始中断状态 值 描述 0 电池电压还没有低于 V_{LOWBAT}。 1 电池电压低于 V_{LOWBAT}。 将 HIBIC 寄存器中的 LOWBAT 位置位即可将该位清零。
1	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。

位/域	名称	类型	复位	描述
0	RTCALT0	RO	0	RTC 警报 0 原始中断状态 值 描述 0 不匹配 1 HIBRTCC 寄存器的值与 HIBRTCM0 寄存器的值匹配，而且 RTCSSC 域的值与 HIBRTCSS 寄存器中 RTCSSM 域的值匹配。 在 HIBIC 寄存器中的 RTCALT0 位写入 1 即可将该位清零。

寄存器 7: 休眠屏蔽中断状态寄存器（HIBMIS），偏移量 0x01C

这个寄存器是休眠模块中断源的可屏蔽中断的状态。该寄存器中的位是 HIBRIS 寄存器和 HIBIM 寄存器中相应的位相与得到的。如果两个相应的位均被置位，且寄存器中位也被置位，那么中断信号将会发送到中断控制器。

休眠屏蔽中断状态寄存器 (HIBMIS)

基址 0x400F.C000
偏移量 0x01C
类型 RO, 复位 0x0000.0000

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
域	保留															
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
域	保留											WC	EXTW	LOWBAT	保留	RTCALT0
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位/域	名称	类型	复位	描述
31:5	保留	RO	0x0000.000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
4	WC	RO	0	写入完成/可以读取屏蔽中断状态 值 描述 0 WRC 位没有被置位 或者中断被屏蔽。 1 由于 WRC 位被置位，系统产生了一个没有屏蔽的中断信号。 在 HIBIC 寄存器中的 WC 位写入 1 即可将该位清零。
3	EXTW	RO	0	外部唤醒可屏蔽中断状态 值 描述 0 外部唤醒中断还没有发生，或者被屏蔽。 1 由于WAKE管脚被置高电平，系统产生了一个没有屏蔽的中断信号。 在 HIBIC 寄存器中的 EXTW 位写入 1 即可将该位清零。
2	LOWBAT	RO	0	低电池电压可屏蔽中断状态 值 描述 0 电池电压过低中断没有发生，或者被屏蔽。 1 由于电池电压过低，系统产生了一个没有屏蔽的中断信号。 将 HIBIC 寄存器中的 LOWBAT 位置位即可将该位清零。
1	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。

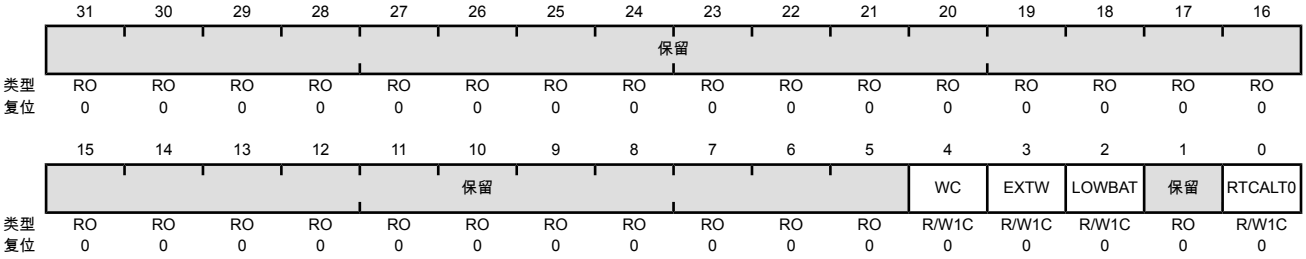
位/域	名称	类型	复位	描述
0	RTCALT0	RO	0	RTC 警报 0 屏蔽中断状态 值 描述 0 RTC 匹配中断没有发生或者被屏蔽。 1 由于发生 RTC 匹配，系统产生了一个没有屏蔽的中断信号。 在 HIBIC 寄存器中的 RTCALT0 位写入 1 即可将该位清零。

寄存器 8: 休眠中断清除寄存器（HIBIC），偏移量 0x020

这个寄存器是休眠模块中断源的中断写1清除寄存器。向 HIBRIS 寄存器中相应的位写入 1，相应的中断就会被清除。

休眠中断清除寄存器 (HIBIC)

基址 0x400F.C000
偏移量 0x020
类型 R/W1C, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:5	保留	RO	0x0000.000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
4	WC	R/W1C	0	写入完成/可以进行中断清除 向该位写入 1 会将 HIBRIS 和 HIBMIS 寄存器中的 WC 位清零。 读取该位将返回原始中断状态。
3	EXTW	R/W1C	0	外部唤醒中断清除 向该位写入 1 会将 HIBRIS 和 HIBMIS 寄存器中的 EXTW 位清零。 读取该位将返回原始中断状态。
2	LOWBAT	R/W1C	0	电池电压低中断清除 向该位写入 1 会将 HIBRIS 和 HIBMIS 寄存器中的 LOWBAT 位清零。 读取该位将返回原始中断状态。
1	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
0	RTCALT0	R/W1C	0	RTC 警报 0 屏蔽中断清除 向该位写入 1 会将 HIBRIS 和 HIBMIS 寄存器中的 RTCALT0 位清零。 读取该位将返回原始中断状态。 注意: 如果 RTC 值和 HIBRTCM0 寄存器/RTCMSS 域值相等，那么计时器中断源不能被清零。匹配中断的优先级比中断清零的优先级高。

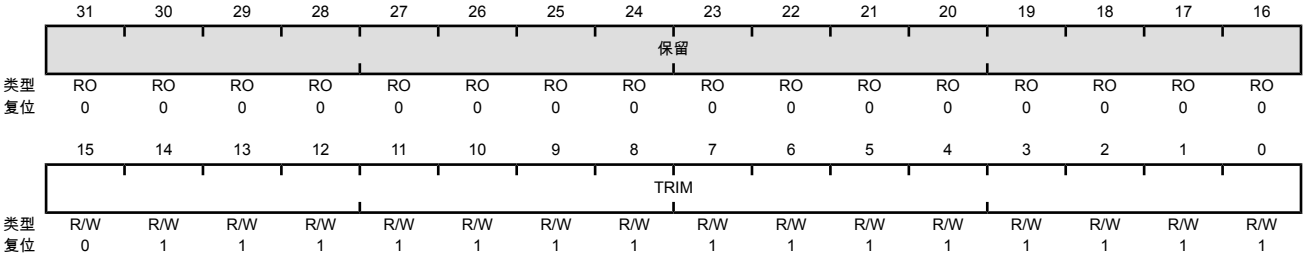
寄存器 9: 休眠 RTC 修正寄存器（HIBRTCT），偏移量 0x024

该寄存器中包含的数值用来修正 RTC 时钟预分频器。它表示计算得到的下溢数值，该数值用于修正周期。在 RTC 模式中，它可以表示为 0x7FFF ± N 时钟周期，其中 N 是每隔 64 秒需要增加或者减少的时钟周期的数量。

注意: 休眠模块寄存器处于休眠模块的时钟域内，具有特殊的时序要求。软件应该利用 HIBCTL 寄存器中的 WRC 位来确保所需的时间差是否已经过去。如果 WRC 位被清零，那么任何写入访问都不起作用。请参阅“寄存器访问间隙”（431页）。

休眠 RTC 修正寄存器 (HIBRTCT)

基址 0x400F.C000
偏移量 0x024
类型 R/W, 复位 0x0000.7FFF



位/域	名称	类型	复位	描述
31:16	保留	RO	0x0000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
15:0	TRIM	R/W	0x7FFF	RTC 修正值 在 RTC 计数器模式中，该数值每隔 64 秒就会加载到 RTC 预分频器中。 它用于调整 RTC 的时钟速率，补偿时钟源的漂移和误差。可以利用软件来调整补偿值，即增加或者减小默认值 0x7FFF。增大该数值会降低 RTC 的速率，而减小该数值会增加 RTC 的速率。

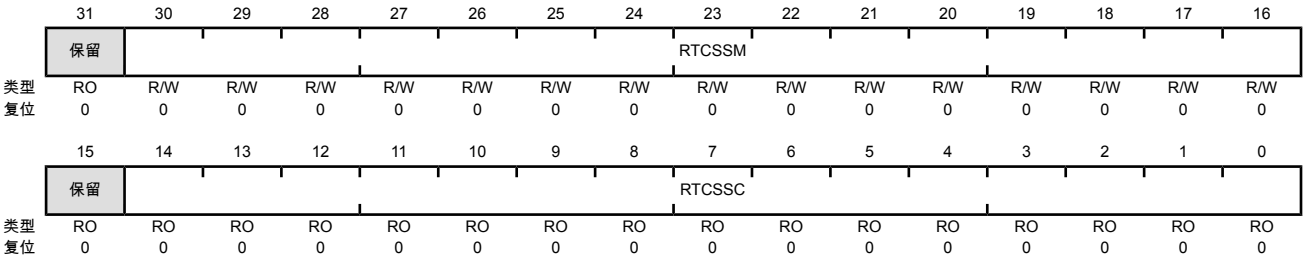
寄存器 10: 休眠 RTC 亚秒寄存器（HIBRTCSS），偏移量 0x028

该寄存器包含 RTC 亚秒计数器和匹配值。要得到 RTC 值，可以首先读取 HIBRTCC 寄存器，再读取 RTCSSC 寄存器的 HIBRTCSS 域，然后再次读取 HIBRTCC 寄存器。如果 HIBRTCC 寄存器两次读取的数值一样，则读取有效。

注意: 休眠模块寄存器处于休眠模块的时钟域内，具有特殊的时序要求。软件应该利用 HIBCTL 寄存器中的 WRC 位来确保所需的时间差是否已经过去。如果 WRC 位被清零，那么任何写入访问都不起作用。请参阅“寄存器访问间隙”（431页）。

休眠 RTC 亚秒寄存器 (HIBRTCSS)

基址 0x400F.C000
偏移量 0x028
类型 R/W, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
30:16	RTCSSM	R/W	0x0000	RTC 亚秒匹配 写入活动将该值加载到 RTC 亚秒匹配寄存器中，单位 1/32,768 秒。 读取返回当前匹配值，单位 1/32,768 秒。
15	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
14:0	RTCSSC	RO	0x0000	RTC 亚秒计数 读取返回亚秒计数值，单位 1/32,768 秒。

寄存器 11: 休眠数据寄存器（HIBDATA），偏移量 0x030-0x06F

这个地址空间用作一个 16x32 位的存储器（64 个字节）。它可以由系统处理器加载以存储状态信息，并且它可使用电池供电，能够在电源关断期间保持状态。

注意: 休眠模块寄存器处于休眠模块的时钟域内，具有特殊的时序要求。软件应该利用 HIBCTL 寄存器中的 WRC 位来确保所需的时间差是否已经过去。如果 WRC 位被清零，那么任何写入访问都不起作用。请参阅“寄存器访问间隙”（431页）。

注意: 如果仲裁移除 V_{DD} 时正在进行 HIBDATA 寄存器写操作，必须在重新应用 V_{DD} 电压后重试写操作。

休眠数据寄存器 (HIBDATA)

基址 0x400F.C000
偏移量 0x030-0x06F
类型 R/W, 复位 -

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	RTD															
类型	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	RTD															
类型	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

位/域	名称	类型	复位	描述
31:0	RTD	R/W	-	休眠模块 NV 数据

8 内部存储器

TM4C1231H6PM 微控制器带有 32 KB 位带 SRAM、内部 ROM256 KB Flash 存储器以及 2KB EEPROM。Flash存储器控制器提供了一个友好的用户接口，使Flash编程成为一项简单的任务。Flash 存储器由 1 KB 可单独擦除的块构成，且能够以 2 KB 大小的块为单位对其应用存储器保护。EEPROM 模块提供了一个定义明确的寄存器接口，既可以用随机读取和写入模式访问 EEPROM，也可以用滚动或者顺序访问模式。密码模型允许应用程序锁定一个或者更多的 EEPROM 模块，来控制 16-字边界的访问。

8.1 结构框图

图8-1（458页）展示了内部SRAM、ROM 和 Flash 存储器模块和控制逻辑。图中的虚线框表示处于系统控制模块中的寄存器。

图 **8-1.** 内部存储器结构图

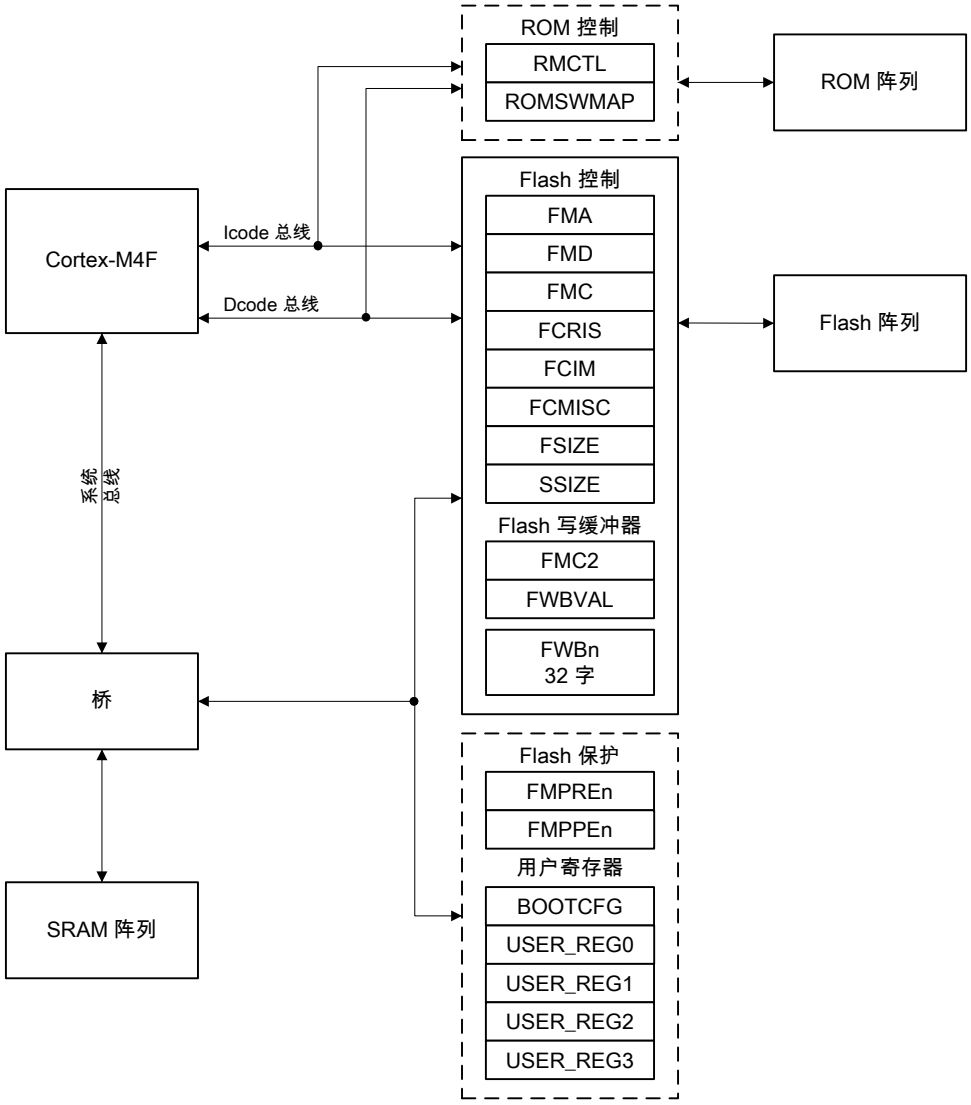
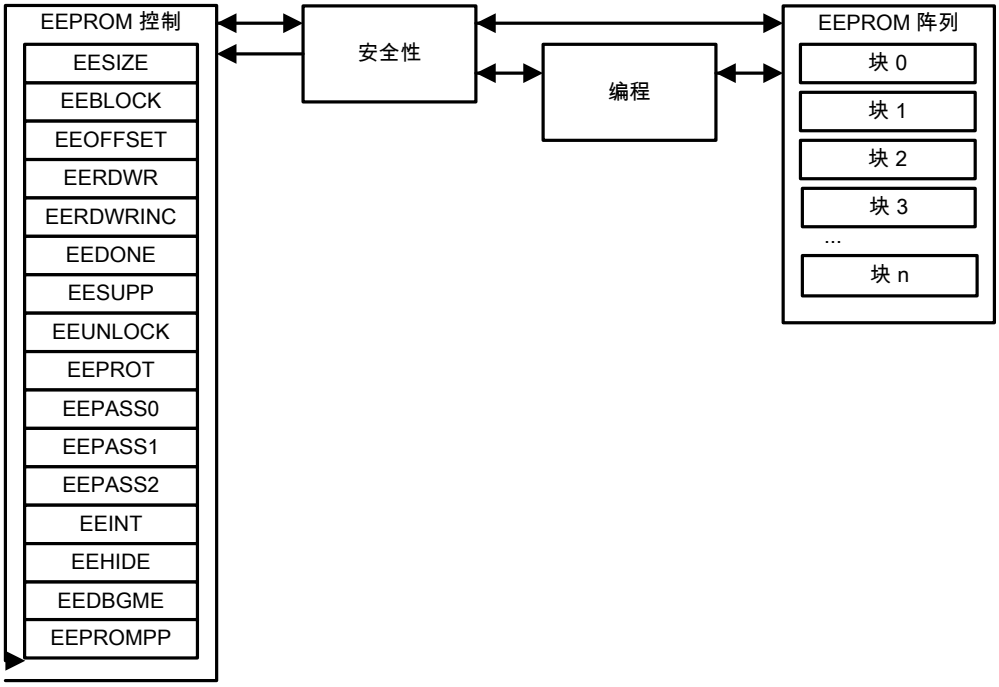


图8-2（459页）说明了内部 EEPROM 模块和控制逻辑。EEPROM 模块连接到 AHB 总线。

图 8-2. EEPROM 结构图



8.2 功能说明

本节描述了 SRAM、ROM Flash 以及 EEPROM 存储器的功能。

注意: μDMA 控制器可以将数据转移到片上 SRAM，也可以从片上 SRAM 将数据转移出。但是，由于 Flash 存储器和 ROM 位于不同的内部总线，所以 μDMA 不能从 Flash 存储器或 ROM 转移数据。

8.2.1 SRAM

TM4C1231H6PM 器件的内部 SRAM 位于器件存储器映射的地址 0x2000.0000。为了减少读 - 修改 - 写（RMW）操作的时间，ARM 在处理器中引入了位带 (bit-banding) 技术。在位带启用的处理器中，存储器映射的特定区域 (SRAM 和外设空间) 能够使用地址别名，在单个原子操作中访问各个位。位带基址位于 0x2200.0000。

位带别名可以使用下面的公式计算：

位带别名 = 位带基址 + （字节偏移量 * 32） + （位编号 * 4）

例如，如果要修改地址0x2000.1000的第3位，位带别名的计算如下：

```
0x2200.0000 + (0x1000 * 32) + (3 * 4) = 0x2202.000C
```

通过计算得出的位带别名，对地址 0x2202.000C 执行读/写操作的指令可以直接访问地址 0x2000.1000 处字节的第 3 位。

关于位带的详细信息，请参考“位带区”（79页）。

注意: SRAM 使用两个32位的SRAW存储区来实现功能 (分离的SRAM阵列)。存储区被这样分开，以便一个包含所有偶数字 (偶存储区)，另一个包含所有奇数字 (奇存储区)。对同一个存储区执行读访问后立即执行写访问，中间会引起一个单时钟周期的停顿。但是对一个存储区执行读访问后对另一个存储区执行写访问，则可以在连续时钟周期内进行而不引起延迟。

8.2.2 ROM

TM4C1231H6PM 器件的内部 ROM 位于器件存储器映射的地址 0x0100.0000。有关 ROM 内容的详细信息，请参阅 “TM4C1231H6PM ROM User’s Guide”。

ROM包含下面几部分：

- TivaWare™ 引导装载程序和向量表
- TivaWare 为产品特定的外设和接口而发行的外设驱动库 (DriverLib)
- 高级加密标准 (AES) 密码表
- 循环冗余检验 (CRC) 错误检测功能

引导装载程序用作初始化程序的装载器(当Flash存储器为空时)，也可以作为一种应用——初始的固件升级机制(通过回调引导装载程序)。应用程序可以调用ROM中外设驱动库的API，以减少对Flash存储器的需求，释放Flash存储器空间用于其它目的(如应用程序增加的特性)。高级加密标准 (AES) 是美国政府使用的公开定义的加密标准。循环冗余检验 (CRC) 技术可用来确认一个数据块的内容是否与先前检验的相同。

8.2.2.1 引导装载程序概述

以下 TivaWare 引导加载程序用来将代码下载到设备的 Flash 存储器中，而不需要使用调试接口。内核复位时，通过使用启动配置 (BOOTCFG) 寄存器中配置好的端口 A-H 的 GPIO 信号，用户可以选择让内核直接执行 ROM 的引导装载程序或 Flash 存储器上的应用程序（请参阅508页）。

复位时，将执行以下序列：

1. 读取 BOOTCFG 寄存器。如果 EN 位被清零，那么执行 ROM 的引导装载程序。
2. 在ROM的Boot Loader中，指定的GPIO管脚的状态与规定的极性相比较，如果管脚状态与规定的极性匹配，那么将 ROM 映射到地址 0x0000.0000 并继续执行 ROM 的 Boot Loader。
3. 如果 EN 位被置位或者管脚状态与指定的极性不符，那么地址 0x0000.0004 的数据就会被读取。如果该数据是 0xFFFF.FFFF，那么 ROM 被映射到地址 0x0000.0000，系统继续执行 ROM 引导装载程序。
4. 如果地址 0x0000.0004 中的数据不是 0xFFFF.FFFF，堆栈指针 (SP) 将装载 Flash 存储器地址 0x0000.0000 的数据，程序计数器 (PC) 将装载地址 0x0000.0004 的数据。随后用户应用程序开始执行。

引导装载程序使用一个简单的封装接口提供与设备的同步通信。由于引导装载程序不能使能PLL，所以它的速度由内部振荡器 (PIOSC) 频率决定。下面的串行接口可以使用：

- UART0
- SSI0
- I^2C0
- USB

UART0、SSI0 和 I2C0 接口的数据格式和通信协议都相同。

注意: 引导加载程序的 Flash 存储器驻留版本也支持 CAN。

有关引导装载程序的详细信息，请参阅 “TivaWare™ Boot Loader for C Series User's Guide（文献编号 SPMU301）”。USB 引导加载程序使用标准的器件固件升级 USB 设备分类。

在 **ROM** 中使用 **UART** 引导加载程序的注意事项

U0Tx 不会由 ROM 引导装载程序驱动，直到完成自动波特率处理。如果 U0Tx 在此期间浮动，则所连接的接收器可能发现信号跳变，这会被其 UART 解读为有效字符。为应对这种情况，对 U0Tx 添加上拉或下拉，为信号提供指定状态，直到 ROM 引导装载程序开始驱动 U0Tx。上拉为更优选择，因为它表示 UART 空闲，而下拉表示中止状态。

8.2.2.2 TivaWare 外设驱动库

以下 TivaWare 外设驱动库包含一个叫做 `driverlib/rom.h` 的文件，它帮助调用 ROM 中的外设驱动库函数。有关每个函数的详细描述，请参阅 “TM4C1231H6PM ROM User’s Guide”。有关调用 ROM 函数和使用 `driverlib/rom.h` 的详细信息，请参阅 “TivaWare™ Peripheral Driver Library for C Series User's Guide（文献编号 SPMU298）” 的“使用 ROM”章节。同时提供了 `driverlib/rom_map.h` 报头文件，以提高使用不同 Tiva™ C 系列 器件时的移植性。不同的器件在 ROM 中可能具有不同的 DriverLib 函数子集。`driverlib/rom_map.h` 报头文件使用创建时间标签将函数调用路由到 ROM（如果这些函数在指定器件上可用），否则，它将路由到函数的 Flash 驻留版本。

ROM起始处的表格指示了ROM提供的API的入口指针。通过这些表格访问API提供了可扩展性，因为API的位置可能会在将来的ROM版本中改变，而API表格不会变。该表格被分为两级，主表格包含的每个指针对应一个外设，该外设指向二级表格。二级表格包含的每个指针对应一个与外设相关的API。主表格的位置在 0x0100.0010，恰好在 ROM 中的 Cortex-M4F 向量表后面。

有关 DriverLib 函数的详细信息，请参阅 “TivaWare™ Peripheral Driver Library for C Series User's Guide（文献编号 SPMU298）”。

增加的API可用于图像和USB功能，但不会预装载到ROM。以下 TivaWare 图像库提供了一系列基本的图像设置，并提供一个小组件，以在装有 Tiva™ C 系列 微处理器且带图形显示的路线板上建立图形用户接口。有关详细信息，请参阅 “TivaWare™ Graphics Library for C Series User's Guide（文献编号 SPMU300）”。

8.2.2.3 高级加密标准 (AES) 密码表

AES 是一种强大的加密方法，拥有不错的性能和大小。AES 在硬件和软件方面都很快，它非常容易使用，并且只需要很少的存储空间。AES 可理想的用于预先排列好密钥的应用，如在加工或配置过程中的设置好。XySSL AES 使用的4个数据表都在ROM中提供。第一个是正向的S-box代换表，第二个是反向的S-box代换表，第三个是正向的多项式表，最后一个是反向的多项式表。有关 AES 的详细信息，请参阅 “TM4C1231H6PM ROM User’s Guide”。

8.2.2.4 循环冗余检验 (CRC) 错误检测

CRC 技术可用来确认信息的正确接收 (在传送中没有丢失或改变)，用来确认解压后的数据，用来证实Flash存储器的内容没有更改，以及其它数据需要被确认的情况。CRC 优于简单的校验和 (例如异或所有的位)，因为它更容易捕捉到变化。有关 CRC 的详细信息，请参阅 “TM4C1231H6PM ROM User’s Guide”。

8.2.3 Flash 存储器

在系统时钟速度为40 MHz或以下时，Flash存储器是单周期读取的。Flash 存储器由一系列 1 KB 的块组织在一起，这些块可以被单独擦除。可以单独对一个 32 位的字进行编程，把当前为 1 的位变为 0。另外，可通过一个写入缓冲区对 Flash 存储器中的 32 个连续字进行编程，所需时间仅是单独对每个字编程的一半。擦除一个块会将块中的所有位都复位为 1。1KB的块可以配成一系列2KB的

块，2KB块可以被单独保护。该保护允许块被标记为只读或只执行，以提供不同等级的代码保护。只读块不能被擦除或编程，块的内容受保护不能修改。只执行块不能被擦除或编程，只能通过控制器取指机制来读取它的内容，块的内容受保护而不能被控制器或调试器读取。

8.2.3.1 预取指缓冲器

Flash 存储器控制器有一个预取指缓冲器，当 CPU 频率大于 40 MHz 时它将自动启用。在此模式下，Flash存储器以一半系统时钟的工作。每个时钟周期，预取指缓冲器获取两个32位字，这样在代码线性执行时允许取指不处于等待状态。取指缓冲器包含一个分支推断机制，它可以辨认出分支从而避免因读取下一对字而增加的额外等待状态。并且，短循环分支经常保持在缓冲器中。因此，一些分支可以没有等待状态而执行。其它分支会引起一个单独的等待状态。

8.2.3.2 Flash 存储器保护

用户可以在四对 32 位宽存储器中使用两种 Flash 存储器保护形式，以 2-KB Flash 存储器块为单位。FMPPEn 和 FMPREn 寄存器的各个位控制各种保护形式的策略（每个块控制一种策略）。

- Flash 存储器保护编程启用 (FMPPEn)：如果某个位被置位，那么就可以对相应的块进行编程（写入）或擦除。如果被清零，对应的块不能更改。
- Flash 存储器保护读取启用 (FMPREn)：如果某个位被置位，那么软件或者调试器就可以对相应的块执行或读取。如果被清零，对应的块只能被执行，块的内容禁止被作为数据读取。

这些策略可以进行组合，如表8-1（462页） 所示。

表 **8-1. Flash** 存储器保护策略组合

FMPPEn	FMPREn	保护
0	0	只执行保护模块只能被执行，不能被写入或擦除。这种模式用来保护代码。
1	0	模块可以被写入、擦除或执行，不能被读取。这种组合很少使用。
0	1	只读保护。模块可以被读取或执行，但不能写入或擦除。这种模式用来锁定模块防止对其进行进一步的修改，但允许对其执行任意的读或执行访问。
1	1	无保护。模块可以被写入、擦除、执行或读取。

对 Flash 存储器的读取保护块（FMPREn 位被置位）进行读取访问是被禁止的，会产生一次总线故障。对 Flash 存储器的编程保护块（FMPPEn 位被置位）进行编程或擦除访问是被禁止的。也可以选择产生一个中断（将 Flash 控制器中断屏蔽 (FCIM) 寄存器中的 AMASK 位置位），以在开发和调试阶段提醒软件开发者注意软件错误。

在 FMPREn 和 FMPPEn 寄存器的出厂设置中，所有已经实现的存储器组所对应的位的值为 1。这种设置实现了一种开放式的访问和可编程性的策略。寄存器的位可通过清零特定寄存器的位来改变。这种改变立即生效，但不是永久的，等到寄存器被提交（保存）以后，位的改变就是永久性的。如果一个位从1变为0且没有提交，那么它可以通过执行一段上电复位序列来恢复。这些更改需要用 Flash 存储器控制 (FMC) 寄存器来提交。有关这些位的编程详情请参考 “非易失性寄存器编程”（465页）。

8.2.3.3 只执行保护

只执行保护可防止对受保护的 flash 存储器块进行读写操作。当器件需要调试功能，但一部分应用空间必须禁止外部访问时，即可使用此模式。此模式的一个典型应用示例是：某公司在销售 Tiva™ C 系列 器件时预先写入了其专有软件，同时允许最终用户向 flash 存储器的未保护区域添加定制代码（例如在电机控制模块的 flash 存储器中设置一个可定制的电机配置区）。

文字数据增加了这种保护机制的复杂度。编译并链接 C 代码后，编译器通常会将文字数据（常量等）置于函数之间的文本区域。在运行过程中，用户可通过 LDR 指令访问文字数据，该指令根据 PC 相关的存储器地址加载存储器中的数据。执行 LDR 指令将在 Cortex-M3 的 DCode 总线上产生

读取通信。这种读取通信将受只执行保护机制保护。如果访问的块标记为仅执行，则通信将被阻止，常量数据将不会加载到处理器，从而相应操作不会正确执行。因此，如果使用只执行保护，就需要以不同的方式处理每个文字数据。我们提供三种处理方式：

1. 使用这样一个编译器：该编译器可将文字数据收集到一个单独区域内，这个单独区域则存在于一个或多个可读取型 flash 存储块中。请注意，LDR 指令可能使用一个 PC 相关的地址（此时，文字池不得位于偏移范围之外），或者软件可能保留一个寄存器，以指向文字池的基址。另外，LDR 偏移是相对于文字池的起始位置而言的。
2. 使用可通过算术指令直接数据和后续计算生成文字数据的编译器。
3. 如果编译器不支持上述两种方法，则以汇编语言的方式使用其中任何一种方法。

8.2.3.4 只读保护

只读保护可防止 flash 块中的内容被重新设置，但允许通过处理器或调试接口进行访问。注：如果 FMPREn 位已清零，指向 Flash 存储器模块的所有读访问都被禁止，包括任何数据访问。必须注意的是，不得将所要求的数据存储在相关 FMPREn 位已清零的 Flash 存储器模块中。

只读模式并不会阻止对既有程序的读取访问，但它可防止内容被意外（或恶意）擦除或编程。当调试接口永久禁用时，只读模式对于引导装载程序等实用程序将特别实用。在这样的配置组合中，可为 Flash 存储器提供访问控制的引导装载程序将受到保护，不会被擦除或者修改。

8.2.3.5 永久禁用调试

在灵敏度极高的应用中，可永久禁用处理器和外设的调试接口，以阻止通过 JTAG 或 SWD 接口对器件进行任何形式的访问。禁用调试接口后，仍可执行标准的 IEEE 指令（例如边界扫描操作），但无法访问处理器和外设。

引导配置 (BOOTCFG) 寄存器的 DBG0 和 DBG1 位用于控制是否打开调试接口。

如果永久禁用调试接口，则必须提供另外一些机制，比如引导装载程序，以便用户安装更新或者修复漏洞。调试接口一旦禁用即无法恢复。

8.2.3.6 中断信号

Flash 控制器在检测到下列状态时会产生中断：

- 编程中断 - 当编程或擦除动作完成时发出的信号。
- 访问中断 - 当对受相应 FMPPEn 位保护的 2 KB 块存储器尝试编程或擦除操作时发出的信号。

能够触发控制器级中断的事件在 Flash 控制器屏蔽中断状态 (FCMIS) 寄存器（请参考479页）中定义，将相应的 MASK 位置位即可启用该中断。如果不使用中断，原始的中断状态总可以通过 Flash 控制器原始中断状态 (FCRIS) 寄存器（请参考477页）进行查看。

对 Flash 控制器屏蔽中断状态和清除 (FCMISC) 寄存器（请参考481页）中相应的位写 1 可以清除相应的中断（适用于 FCMIS 和 FCRIS 寄存器）。

8.2.3.7 Flash 存储器编程

德州仪器 Tiva™ C 系列 设备为 Flash 存储器编程提供了一个友好的用户接口。所有的擦除/编程操作都通过 3 个寄存器来处理：Flash 存储器地址 (FMA)、Flash 存储器数据 (FMD) 和 Flash 存储器控制 (FMC)。注意，如果微控制器的调试功能没有激活而处在“锁死”状态，必须执行一段恢复序列来再激活调试模块。请参阅“恢复一个“锁死”的微控制器”（180页）。

在Flash存储器操作(写、页擦除或整体擦除)过程中，对它进行访问是禁止的。所以指令和按字取指都将延迟到Flash存储器操作完成。如果在Flash存储器操作过程中需要执行指令，那么代码必须放置在SRAM上并在SRAM上执行。

注意: 对 Flash 存储器进行编程时，必须考虑存储器的以下特性：

- 只有擦除才能将位从 0 变成 1。
- 写入操作只能将位从 1 变成 0。如果写入操作试图将 0 变成 1，那么该写入操作失败，不会改变任何位的状态。
- 进入睡眠或者深度睡眠模式（使用等待中断指令 WFI）之前即可开始执行 flash 操作。进入睡眠或者深度睡眠模式之后也可执行这些操作。如果访问 EEPROM 之后发生 Flash 编程/擦除事件，则 Flash 事件将在从睡眠/深度睡眠模式唤醒后开始执行并执行完毕。

8.2.3.8 基本编程/擦除操作

编程 **1** 个 **32** 位字

1. 将源数据写入 FMD 寄存器。
2. 将目标地址写入 FMA 寄存器。
3. 将 Flash 存储器写入密钥和 WRITE 位写入 FMC 寄存器。要写入 Flash 存储器，必须将值 0xA442 或 0x71D5 写入 WRKEY 域中，具体取决于 BOOTCFG 寄存器 KEY 位的值。
4. 查询 FMC 寄存器，直至 WRITE 位被清零。

执行一个 **1 KB** 页的擦除

1. 将页地址写入 FMA 寄存器。
2. 将 Flash 存储器写入密钥和 ERASE 位写入 FMC 寄存器。要写入 Flash 存储器，必须将值 0xA442 或 0x71D5 写入 WRKEY 域中，具体取决于 BOOTCFG 寄存器 KEY 位的值。
3. 查询 FMC 寄存器，直到 ERASE 位被清零，或者通过 FCIM 寄存器的 PMASK 位启用编程中断。

执行一次 **Flash** 存储器的整体擦除

1. 将 Flash 存储器写入密钥和 MERASE 位写入 FMC 寄存器。要写入 Flash 存储器，必须将值 0xA442 或 0x71D5 写入 WRKEY 域中，具体取决于 BOOTCFG 寄存器 KEY 位的值。
2. 查询 FMC 寄存器，直到 MERASE 位被清零，或者通过 FCIM 寄存器中的 PMASK 位启用编程中断。

8.2.3.9 32 字 Flash 存储器写缓冲器

通过 32 字的写入缓冲器可以对两个 32 位字同时编程，加快 Flash 存储器的写入访问速度，从而可以使用上述方法在处理 16 字的同时对 32 字进行编程。被缓存的数据写入 Flash 写缓冲器 (FWBn) 寄存器。

这些寄存器与 Flash 存储器是 32 字对齐的，所以 FWB0 寄存器对应 FMA 中的地址（FMA 的 [6:0] 位都是 0）。FWB1 寄存器对应 FMA + 0x4 中的地址，后面以此类推。只有上次缓存 Flash 存储器写入操作之后更过新的 FWBn 寄存器才会被写入。Flash 写缓冲器有效 (FWBVAL) 寄存器显示了从上次缓存 Flash 存储器写操作之后，哪些寄存器已经被写入。该寄存器包含的位对应 32 个 FWBn

寄存器，其中 FWBVAL 的第 [n] 位对应 FWBn。如果 FWBVAL 寄存器的某位被置位，那么相应的 FWBn 寄存器已经被更新了。

用一次单独被缓冲的 **Flash** 存储器写操作来编程 **32** 个字

1. 将源数据写入 FWBn 寄存器。
2. 将目标地址写入 FMA 寄存器。该地址必须是一个 32 字对齐的地址（即 FMA 的 [6:0] 必须全是 0）。
3. 将 Flash 存储器写入密钥和 WRBUF 位写入 FMC2 寄存器。要写入 Flash 存储器，必须将值 0xA442 或 0x71D5 写入 WRKEY 域中，具体取决于 BOOTCFG 寄存器 KEY 位的值。
4. 查询 FMC2 寄存器，直到 WRBUF 位被清零，或者等待 PMIS 中断信号发出。

8.2.3.10 非易失性寄存器编程

注意: 引导配置 (BOOTCFG) 寄存器需要执行一次上电复位 (POR)，然后才会让提交的更改生效。

本节讨论如何更新 表8-2（466页） 显示的 Flash 存储器自身中的寄存器。这些寄存器驻留在与主 Flash存储器阵列分离的空间，并且不受擦除或整体擦除的影响。除启动配置 (BOOTCFG) 寄存器外，这些寄存器中的设置可进行写操作，还可校验他们的功能并在提交前读回他们的值，此时他们是非易失性的。如果某个寄存器的一个值尚未提交，那么一个上电复位即可恢复上次提交的值或默认值（如果寄存器从未提交过值）。其他类型的复位不起任何作用。一旦寄存器的内容被提交，唯一能恢复出厂默认值的办法就是执行 “恢复一个“锁死”的微控制器”（180页） 中描述的操作。

要对非易失性寄存器进行写操作：

- 这些寄存器中的位只能由 1 变为 0。
- 对于除 BOOTCFG 寄存器之外的所有寄存器，将数据写入寄存器描述中提供的寄存器地址。对于 BOOTCFG 寄存器，将数据写入 FMD 寄存器。
- 寄存器可进行读操作以校验其内容。要校验存储在 BOOTCFG 寄存器中的内容，需读取 FMD 寄存器。读取 BOOTCFG 寄存器将返回之前提交的值或（如果寄存器从未提交过值）默认值。
- 新值在所有寄存器中立即生效，但 BOOTCFG 寄存器例外，该寄存器的新值在提交前并不会存储在寄存器中。
- 在提交寄存器值之前，一个上电复位将恢复上次提交的值或默认值（如果寄存器从未提交过值）。

要提交新值到非易失性寄存器：

- 如上所述写入数据。
- 将 表8-2（466页） 显示的值写入 FMA 寄存器。
- 写入 Flash 存储器写入密钥，并将 FMC 寄存器的 COMT 位置位。这些值必须同时写入 FMC 寄存器中。
- 提交非易失性寄存器的时序与常规 Flash 存储器的写操作相同，由 表21-26（1053页） 中所示的 T_{PROG64} 定义。软件可轮询 FMC 寄存器的 COMT 位以确定操作何时完成，或者可将 FCIM 寄存器的 PMASK 位置位以启用中断。
- 提交 BOOTCFG 寄存器时，如果尝试将已提交为 0 的位提交为 1，FCRIS 寄存器的 INVDRIS 位会被置位。

- 一旦提交该值，上电复位将不会对寄存器内容产生任何影响。
- 对 BOOTCFG 寄存器的更改在下次上电复位后生效。
- 一旦将 NW 位更改为 0 并提交，就无法对 BOOTCFG 寄存器进行更多更改。

重要: 提交过后，这些寄存器只能通过“恢复一个“锁死”的微控制器”（180页）中描述的操作恢复到他们的出厂设置。由该操作引起的主 Flash 存储器阵列的整体擦除操作发生在这些寄存器的恢复操作之前。

表 **8-2.** 用户可编程的 **Flash** 存储器驻留寄存器

被提交的寄存器	**FMA** 值	数据源
FMPRE0	0x0000.0000	FMPRE0
FMPRE1	0x0000.0002	FMPRE1
FMPRE2	0x0000.0004	FMPRE2
FMPRE3	0x0000.0006	FMPRE3
FMPPE0	0x0000.0001	FMPPE0
FMPPE1	0x0000.0003	FMPPE1
FMPPE2	0x0000.0005	FMPPE2
FMPPE3	0x0000.0007	FMPPE3
USER_REG0	0x8000.0000	USER_REG0
USER_REG1	0x8000.0001	USER_REG1
USER_REG2	0x8000.0002	USER_REG2
USER_REG3	0x8000.0003	USER_REG3
BOOTCFG	0x7510.0000	FMD

8.2.4 EEPROM

TM4C1231H6PM 微控制器包含 1 个 EEPROM 单元，其特性如下：

- 可用 2K 字节的存储器，即 512 32 位字
- 32 个块区，每区 16 字（64 字节）
- 内置的换位写入技术
- 每个模块的访问保护
- 整个外设的锁定保护选项和每个块的锁定保护一样，都使用 32 位到 96 位的解锁代码（根据应用的需要选择）
- 支持写完成中断，避免轮询
- 每个 2 页面块可进行 500K 次写操作（按周期使用固定偏移量对隔页进行写操作时）到 15M 次操作（在两个页面之间循环时）。

8.2.4.1 功能说明

EEPROM 模块提供了一个定义明确的寄存器接口，既可以用随机读取和写入模式访问 EEPROM，也可以用滚动或者顺序访问模式。

保护机制可以对 EEPROM 进行锁定，在很多情况下能够阻止不必要的写入或者读取操作。密码模型允许应用程序锁定一个或者更多的 EEPROM 模块，来控制 16-字边界的访问。

重要: 系统时钟的配置在 EEPROM 操作过程中不可更改。软件必须等到 EEPROM 完成状态 (EEDONE) 寄存器中的 WORKING 位清零之后，才能对系统时钟做出更改。

模块

EEPROM 中有 32 个大小为 16 字的块。可以读取字节和半字，而且访问不必发生在字边界。整个字被读取，任何不需要的数据将被忽略。他们只在字的别名处可写入。要写入字节，需要读取字值，修改相应的字节，并重新写回字。

每个块都可以用块选择寄存器进行寻址，地址是 EEPROM 中的偏移量。每个字也可以在块中进行偏移量寻址。

当前块由 EEPROM 当前块 (EEBLOCK) 寄存器选择。当前地址偏移量由 EEPROM 当前偏移量 (EEOFFSET) 寄存器选择并验证有效性。应用程序可以随时对 EEOFFSET 寄存器进行写入，而且当 EEPROM 读写加 1 (EERDWRINC) 寄存器被访问时，它还会自动递增。然而，EERDWRINC 寄存器不会增加块数量，而是在块中换行。

每个块可以单独保护。读取应用程序无权访问的块会返回 0xFFFF.FFFF。对应用程序无权访问的块进行写入操作会导致 EEDONE 寄存器出错。

时序注意事项

启用或者复位 EEPROM 模块之后，软件必须等到 EEDONE 寄存器中的 WORKING 位清零之后才能访问各 EEPROM 寄存器。

如果 EEPROM 进行写入的同时，Flash 存储器在写入或擦除，EEPROM 进程可能被 Flash 存储器写入/擦除操作中断，并在完成 Flash 存储器写入后继续。该操作可能更改 EEPROM 操作所需时间。

EEPROM 操作必须在进入睡眠或深度睡眠模式之前完成。在发布 WFI 指令进入睡眠或深度睡眠之前，应先校验 EEPROM 完成状态 (EEDONE) 寄存器以确保完成 EEPROM 操作。

读取块内的字交以直接速度进行，意思是如果系统时钟速度比 EEPROM 的速度快，那么等待状态将自动产生。读取访问时间在表21-27（1053页）中做了定义。

对 EEOFFSET 寄存器进行写入操作也不会招致任何错误。

对 EEBLOCK 寄存器的写入操作不会产生延迟，但是在写入 EEBLOCK 之后访问块中的数据会延迟 4 个时钟。这个时间用来装载块的具体信息。

对块中的字进行写入操作也会产生延迟，延迟时间各不相同。应用程序可以利用中断来得知什么时候写入操作完成，或者可以查询 EEDONE 寄存器来了解写入完成状态。EEPROM 的写入时序和擦除时序都是可变的，其中擦除时间比大多数外部 EEPROM 的写入时间都短。

锁定和密码

EEPROM 可以在模块级和块级被锁定。锁定功能由存储在 EEPROM 密码 (EEPASSn) 寄存器中的密码控制，该密码可以是任何 32 位到 96 位之间的值，但不能全是 1。块 0 是主块，它的密码可以保护控制寄存器以及其他的块。还可以用块密码对每个块进行进一步保护。

如果块 0 具有密码，那么在复位时整个模块都会被锁定。锁定规则如下：块 0 解锁以后，从块 1 到块 31 可访问，而块 0 受自己的保护位保护。因此，EEBLOCK 寄存器不能从 0 开始改变，除非块 0 被解除锁定。

任何块（包括块 0）有了密码以后，都可以根据锁定或没有锁定控制块是否可访问。一般来说，在锁定时，这种锁定保护可以用来阻止写入访问或者阻止写入和读取访问。

复位时所有密码保护的块都会被锁定。要解锁块，必须将正确的密码写入EEPROM解锁(EEUNLOCK)寄存器。写入密码时，应使用 EEPASSn 寄存器将其写入 1 到 3 次，以形成 32 位、64 位或 96 位的注册密码。用来配置 EEPASS0 寄存器的值必须最后写入。比如，对于一个 96 位的密码，用来配置 EEPASS2 寄存器的数值必须先写入，然后是 EEPASS1 和 EEPASS0 寄存器的数值。在 EEUNLOCK 寄存器中写入 0xFFFF.FFFF 可以重新锁定块或模块，因为 0xFFFF.FFFF 不是有效密码。

保护和访问控制

保护位为每个块提供了单独的读写控制，可以实现以下多种保护模式：

- 没有密码：任何时间都能读取和写入。在没有密码时，该模式是默认设置。
- 没有密码：可以读取，但是不能写入。
- 有密码：可读取，只有在密码解锁后才能写入。在有密码时，该模式是默认设置。
- 有密码：在锁定时只能读取，或者只能写入。
- 有密码：在锁定时只读，但是不能写入。

另外，访问保护可以通过处理器模式实现。这种配置允许仅管理员访问或者管理员和用户访问，是默认模式。管理员访问模式还可以阻止 µDMA 和调试器的访问。

另外，主块可以用来控制保护机制本身的访问保护。如果块 0 的访问控制是仅管理员访问模式，那么整个模块可以用管理员模式访问。另外，块 0 的保护级别为整个 EEPROM 设置了最低的保护级别。比如，如果 EEPROT 寄存器中的 PROT 域针对块 0 设置成了 0x1，那么块 1 的 PROT 域只能是 0x1、0x2 或者 0x3，而不能是 0x0。

请注意，如果块 0 有密码而且没有解锁，那么从块 1 到块 31 均不能读写。如果块 0 有一个主密码，那么块 0 或者某个块的最严格的保护设置会同样适用于其他的块。

隐藏块

隐藏可以提供暂时的保护。除了块 0，其他块都可以隐藏，这样可以阻止对隐藏块的访问，直到下一次复位。

这种机制可以允许引导程序或者初始化过程访问某些数据，而所有其他进一步的访问都不能访问这些数据。因为引导和初始化程序控制应用程序的功能，所以隐藏块在调试功能禁用时可以提供强大的数据隔离。

典型应用模型：初始化代码中包含密码、密钥和/或哈希表，用以验证其他的应用程序。一旦执行，该块将被隐藏，直到下次复位并重新输入初始化代码之后才能访问。

电源和复位安全

一旦 EEDONE 寄存器显示一个位置被成功写入数据，那么该数据会一直保持，直到该位置被重新写入。EEDONE 寄存器显示写入完成以后，将不存在电源或者复位冲突。

中断控制

EEPROM 模块允许在写入完成时产生中断，从而无需进行轮询。该中断可以用来驱动应用程序的 ISR（中断服务程序），ISR 可写入更多的字或者验证操作完成。不管是由错误还是成功编程或擦除操作导致，该中断机制可以在 EEDONE 寄存器工作或停止时随时使用。该中断机制适用于数据写入、密码寄存器和保护寄存器写入、EEPROM 支持控制和状态 (EESUPP) 寄存器强制擦除，以及 EEPROM 调试整体擦除 (EEDGBME) 寄存器整体擦除。EEPROM 中断用 Flash 存储器中断向量给内核发信号。软件可以通过检测 Flash 控制器屏蔽中断状态和清除 (FCMISC) 寄存器的第 2 位来确定中断源是否是 EEPROM。

工作原理

EEPROM 使用采用 EEPROM 类型单元的传统 Flash 存储块模型操作，但是使用扇区擦除。另外，当需要时，页中复制的字允许多于 50 万个擦除周期，这意味着每个字都有一个最新版本。因此，写入操作在一个新的位置创建了一个新版本的字，原来的数值就会过时。

每个扇区包含两个块。每个块包含一个活动副本和六个冗余副本的位置。密码、保护位和控制数据都存储在页中。

当页中存储最新字的空间不够时，会启用一个复制缓冲区。复制缓冲区可复制每个块中最新的字。原来的页将被擦除。最后，复制缓冲区的内容被复制回页中。这种机制确保掉电的时候数据不会丢失，即使是在操作期间。EEPROM 机制跟踪所有状态信息，提供全面的安全性和保护。尽管不太可能发生错误，但是特定环境下编程还是可能产生错误，比如，编程过程中电压过低。在这些情况下，可以用 EESUPP 寄存器来完成一个操作，请参阅“编程中的错误”一节（469页）。

手动复制缓冲区擦除

复制缓冲区只有在主块满的时候才用，因为一个字已经写入七次，没有更多的空间来存储它的最新版本。在这种情况下，该块中所有字的最新版本都被复制到复制缓冲区，以便主块安全擦除，提供掉电安全保护。如果复制缓冲区自身满了的话，那么它必须首先被擦除，但这样会额外延长操作时间。对复制缓冲区执行手动擦除以后，将来的写入访问即可节约时间。如果复制缓冲区必须被擦除，那么应将 EESUPP 寄存器中的 EREQ 位置位。如果这样，应用程序可以设置 START 位，以强制在方便的时候进行擦除。EEDONE 和 EEINT 寄存器可以用来监测完成情况。

调试整体擦除

EEPROM 调试整体擦除功能使得开发人员能够对 EEPROM 执行整体擦除要正确执行整体擦除，当前不得存在活动的 EEPROM 操作。完成上一次 EEPROM 操作后，应用程序必须确保任何 EEPROM 寄存器均未更新，包括在不执行实际读写操作的情况下修改 EEBLOCK 和 EEOFFSET 寄存器。要禁止这些操作，应用程序必须将 EEPROM 软件复位 (SREEPROM) 寄存器的 R0 位置位，以将 EEPROM 模块复位，并等待 EEPROM 完成状态 (EEDONE) 寄存器的 WORKING 位清零，然后将 EEPROM 调试整体擦除 (EEDBGME) 寄存器的 ME 位置位，以此来启用调试整体擦除功能。

编程中的错误

数据写入、密码设定、保护设定以及复制缓冲区擦除等操作可以同时执行多种操作。比如，一次普通的写入操作将执行两种写入操作：控制字写入和数据写入。如果控制字成功写入，但是数据写入失败（比如，由于电压降低），整个写入也失败：这会显示在 EEDONE 寄存器中。失败和纠正措施根据操作类型分类：

- 如果因控制字成功写入但是数据没有写入而导致普通写入失败，那么将在系统稳定（比如说，电压稳定）以后会重复一次该操作，以作为安全措施。在重试之后，控制字和写入数据将移动到下一个位置。
- 如果密码或者保护写入失败，那么将在系统稳定以后也会重复一次该操作，以作为安全措施。在制造或者设置模式以外的模式中写入多字密码写入时必须小心，要保证所有的字连续写入。否则即必须对部分密码解锁，以提供恢复功能。
- 如果字写入需要将块写入复制缓冲区中，那么在接下来的操作中可能失败或者掉电。发生错误时，控制字机制可跟踪 EEPROM 处于哪一步。如果没有完成，EESUPP 寄存器会显示部分完成，可以写入 EESUPPSTART 位来允许它继续执行，直到完成。
- 如果复制缓冲区擦除失败或者擦除时掉电，那么 EESUPP 寄存器会显示操作没有完成，并允许该操作重新开始。

复位后，在开始写入任何数据到 EEPROM 之前，软件必须读取 EESUPP 寄存器并校验是否存在任何错误状态，该状态可能显示系统由于电压过低而复位，当前正在执行写入或擦除操作。如果

PRETRY 位或 ERETRY 位被置位，外设应置位，方法是将 EEPROM 软件复位 (SREEPROM) 寄存器的 R0 位置位并清零，然后等待 EEDONE 寄存器的 WORKING 位清零，之后再重新校验 EESUPP 寄存器的错误指示。此过程应允许 EEPROM 从写入或擦除错误中恢复。在少数情况下，EESUPP 寄存器在执行此操作后可能继续显示错误，此时应重复该复位操作。从错误中恢复后，该应用程序应重新写入发生初次故障时正在进行编程的数据。

耐久性

耐久性以元块为单位，每个元块包含两个块。耐久性用两种方法测量：

1. 对于应用程序来说是可执行的写入次数。
2. 对于微控制器来说是元块上可执行的擦除次数。

在第二种测量方法中，写入次数取决于如何执行写入操作。例如：

- 一个字可以写入 50 万次，但是这些写入会影响该字所在的元块。因此，写入一个字 50 万次以后再向旁边的字写入 50 万次就无法保证能够成功。为了确保成功，这些字应该进行更多的并行写入。
- 所有的字可以在一次扫描中写入，超过 50 万次的扫描会把所有的字更新超过 50 万次。
- 不同的字可以这样写入：任何或者所有的字写入超过 50 万次，但是每个字的写入数量相同。比如，偏移量 0 写入三次，偏移量 1 写入两次，偏移量 2 写入四次，偏移量 1 写入两次，然后偏移量 0 再次写入。因此，在该顺序结束时，所有的三个偏移量都是四次写入。这种七次写入的平衡最大程度地增加了同一个元块中不同字的耐久性。

8.2.4.2 EEPROM 初始化及配置

在写入任何 EEPROM 寄存器之前，必须启用 EEPROM 模块的时钟，请参考 308页。

标准设置如下：

- 块 0 有密码。
- 块 0 可读取，但是只有解锁后才能写入。
- 块 0 有一个 ID 以及其他公共数据。

这种配置中，ID 任何时候都可读，但是 EEPROM 剩下的内容被锁定，应用程序不能访问。当可以访问 EEPROM 的应用程序部分选择解锁块 0 时，余下的块才可访问。

8.3 寄存器映射

表 8-3（470页）列出了 ROM 控制器寄存器和 Flash 存储器以及控制寄存器。表中列出的偏移量是相对于特定存储器控制器基址的 16 进制增量。Flash 存储器寄存器偏移量是相对于 Flash 存储器控制基址 0x400F.D000而言的。EEPROM 寄存器偏移量是相对于 EEPROM 基址 0x400A.F000而言的。ROM 和 Flash 存储器保护寄存器偏移量是相对于系统控制基址 0x400F.E000而言的。

表 **8-3. Flash** 寄存器映射

偏移量	名称	类型	复位	描述	见页面
Flash 存储器寄存器（**Flash** 控制偏移量）					
0x000	FMA	R/W	0x0000.0000	Flash 存储器地址	473

表 8-3. Flash 寄存器映射（续）

偏移量	名称	类型	复位	描述	见页面
0x004	FMD	R/W	0x0000.0000	Flash 存储器数据寄存器	474
0x008	FMC	R/W	0x0000.0000	Flash 存储器控制	475
0x00C	FCRIS	RO	0x0000.0000	Flash 控制器原始中断状态	477
0x010	FCIM	R/W	0x0000.0000	Flash 控制器中断屏蔽	479
0x014	FCMISC	R/W1C	0x0000.0000	Flash控制器可屏蔽中断的状态和清除	481
0x020	FMC2	R/W	0x0000.0000	Flash存储器控制2	483
0x030	FWBVAL	R/W	0x0000.0000	Flash 写缓冲器有效	484
0x100 - 0x17C	FWBn	R/W	0x0000.0000	Flash 写缓冲器n	485
0xFC0	FSIZE	RO	0x0000.007F	Flash 容量寄存器	486
0xFC4	SSIZE	RO	0x0000.007F	SRAM 大小寄存器	487
0xFCC	ROMSWMAP	RO	0x0000.0000	ROM 软件映射寄存器	488
EEPROM 寄存器（**EEPROM** 控制偏移量）					
0x000	EESIZE	RO	0x0020.0200	EEPROM 大小信息寄存器	489
0x004	EEBLOCK	R/W	0x0000.0000	EEPROM 当前块寄存器	490
0x008	EEOFFSET	R/W	0x0000.0000	EEPROM 当前寄存器	491
0x010	EERDWR	R/W	-	EEPROM 读写寄存器	492
0x014	EERDWRINC	R/W	-	EEPROM 读写加 1 寄存器	493
0x018	EEDONE	RO	0x0000.0000	EEPROM 完成状态寄存器	494
0x01C	EESUPP	R/W	-	EEPROM 支持控制和状态寄存器	496
0x020	EEUNLOCK	R/W	-	EEPROM 解锁寄存器	498
0x030	EEPROT	R/W	0x0000.0000	EEPROM 保护寄存器	499
0x034	EEPASS0	R/W	-	EEPROM 密码寄存器	500
0x038	EEPASS1	R/W	-	EEPROM 密码寄存器	500
0x03C	EEPASS2	R/W	-	EEPROM 密码寄存器	500
0x040	PWM0CTL	R/W	0x0000.0000	EEPROM 中断寄存器	501
0x050	EEHIDE	R/W	0x0000.0000	EEPROM 块隐藏寄存器	502
0x080	EEDBGME	R/W	0x0000.0000	EEPROM 调试整体擦除寄存器	503
0xFC0	EEPROMPP	RO	0x0000.001F	EEPROM 外设属性寄存器	504
存储器寄存器（系统控制偏移量**)**					
0x0F0	RMCTL	R/W1C	-	ROM 控制	505
0x130	FMPRE0	R/W	0xFFFF.FFFF	Flash 存储器保护读取启用寄存器 0	506
0x200	FMPRE0	R/W	0xFFFF.FFFF	Flash 存储器保护读取启用寄存器 0	506

表 8-3. Flash 寄存器映射（续）

偏移量	名称	类型	复位	描述	见页面
0x134	FMPPE0	R/W	0xFFFF.FFFF	Flash 存储器保护编程启用寄存器 0	507
0x400	FMPPE0	R/W	0xFFFF.FFFF	Flash 存储器保护编程启用寄存器 0	507
0x1D0	BOOTCFG	RO	0xFFFF.FFFE	启动配置	508
0x1E0	USER_REG0	R/W	0xFFFF.FFFF	用户寄存器 0	511
0x1E4	USER_REG1	R/W	0xFFFF.FFFF	用户寄存器 1	511
0x1E8	USER_REG2	R/W	0xFFFF.FFFF	用户寄存器 2	511
0x1EC	USER_REG3	R/W	0xFFFF.FFFF	用户寄存器 3	511
0x204	FMPRE1	R/W	0xFFFF.FFFF	Flash 存储器保护读取启用寄存器 1	506
0x208	FMPRE2	R/W	0xFFFF.FFFF	Flash 存储器保护读取启用寄存器 2	506
0x20C	FMPRE3	R/W	0xFFFF.FFFF	Flash 存储器保护读取启用寄存器 3	506
0x404	FMPPE1	R/W	0xFFFF.FFFF	Flash 存储器保护编程启用寄存器 1	507
0x408	FMPPE2	R/W	0xFFFF.FFFF	Flash 存储器保护编程启用寄存器 2	507
0x40C	FMPPE3	R/W	0xFFFF.FFFF	Flash 存储器保护编程启用寄存器 3	507

8.4 Flash 存储器寄存器描述 (Flash 控制偏移量)

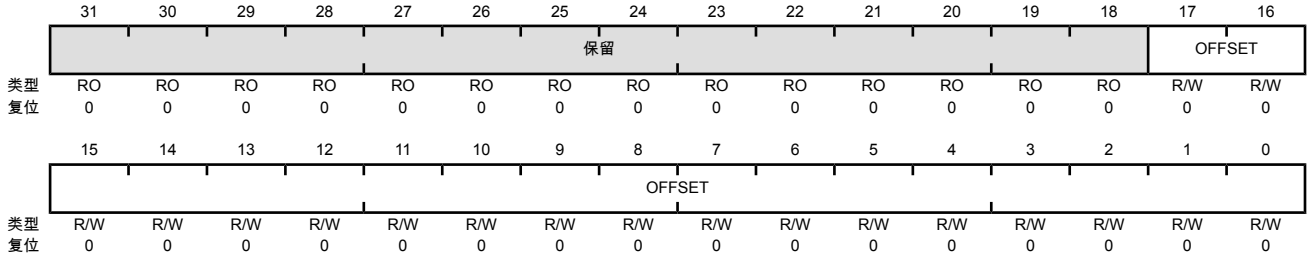
本节按照地址偏移的数字顺序排列和描述Flash寄存器。本节的寄存器地址偏移量都是相对于 Flash 控制基址 0x400F.D000 而言的。

寄存器 1: Flash 存储器地址（FMA），偏移量 0x000

在单字写操作过程中，该寄存器含有一个 4 字节对齐的地址并指定在哪里写入数据。在使用写缓冲区进行写入操作时，此寄存器含有一个 128 字节（32 字）对齐的地址，用以指定开始写入 32 字块。在擦除操作过程中，该寄存器含有一个 1 KB 对齐的 CPU 字节地址，并负责指定哪个块将被擦除。注意必须要符合对齐的要求，否则操作的结果将不可预知。

Flash 存储器地址 (FMA)

基址 0x400F.D000
偏移量 0x000
类型 R/W, 复位 0x0000.0000



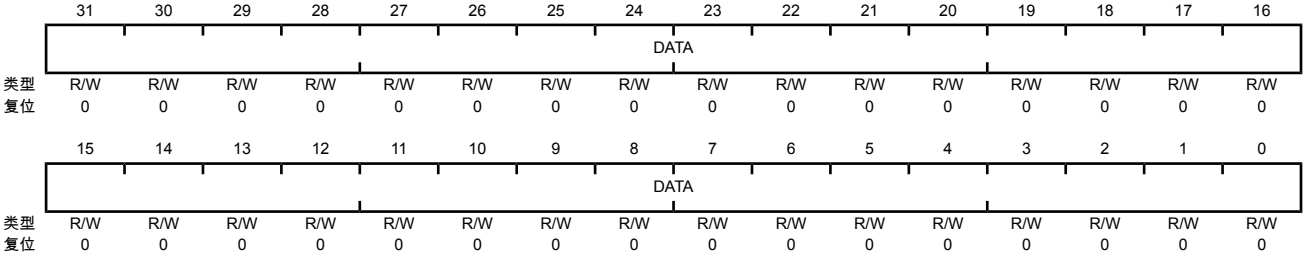
位/域	名称	类型	复位	描述
31:18	保留	RO	0x0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
17:0	OFFSET	R/W	0x0	地址偏移量。 执行操作的 Flash 存储器地址偏移量，非易失性寄存器除外（有关该域值的详细内容请参考 “非易失性寄存器编程”（465页））。

寄存器 2: Flash 存储器数据寄存器（FMD），偏移量 0x004

该寄存器中包含的是编程周期中被写入的数据。该寄存器在擦除周期中不使用。

Flash 存储器数据寄存器 (FMD)

基址 0x400F.D000
偏移量 0x004
类型 R/W, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:0	DATA	R/W	0x0000.0000	数据值 写操作的数据值。

寄存器 3: Flash 存储器控制（FMC），偏移量 0x008

当该寄存器被写入时，Flash 存储器控制器将对 Flash 存储器地址 (FMA) 寄存器指定的位置启动适当周期数目的访问操作（请参阅473页）。如果访问是写入操作，那么 Flash 存储器数据 (FMD) 寄存器中的数据将会写入指定地址（请参阅474页）。

该寄存器是启动存储器操作过程中最后被写入的寄存器。该寄存器低字节的4个控制位用来启动存储器操作。

注意不要设置多个控制位使得操作的结果不可预知。

Flash 存储器控制 (FMC)
基址 0x400F.D000
偏移量 0x008
类型 R/W, 复位 0x0000.0000

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	WRKEY															
类型	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	保留												COMT	MERASE	ERASE	WRITE
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位/域	名称	类型	复位	描述
31:16	WRKEY	WO	0x0000	Flash存储器写密钥 该域包含一个写密钥，用于最大限度的减少对Flash的意外写入。要写入 Flash 存储器，必须将值 0xA442 或 0x71D5 写入此域中，具体取决于 BOOTCFG 寄存器 KEY 位的值。如果没有这个 WRKEY 值，那么向 FMC 寄存器中的写入操作将被忽略。读取该域将返回0。
15:4	保留	RO	0x00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
3	COMT	R/W	0	提交寄存器值 该位用于提交对Flash存储器驻留寄存器的写入，并监视提交过程。 值 描述 0 写 0 对该位状态没有影响。 读取该位为0表示之前的提交访问完成。 1 置位该位可以向Flash存储器驻留寄存器提交（写入）寄存器的值。 读取该位为1表示之前的提交访问没有完成。 更多 Flash 存储器驻留寄存器的信息，请参考 “非易失性寄存器编程”（465页）。

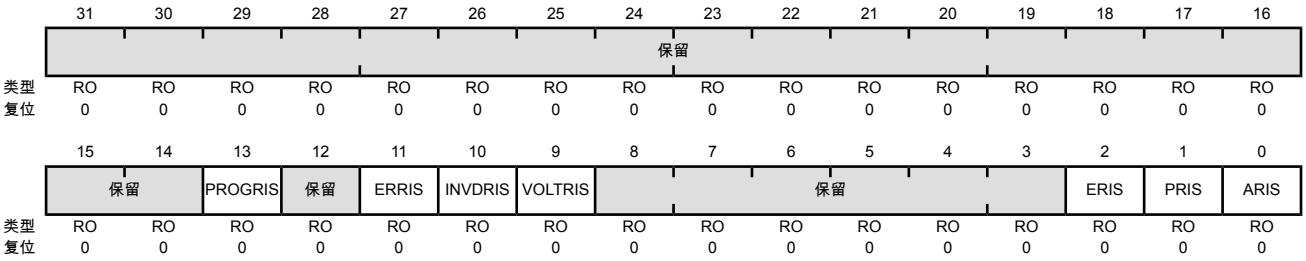
位/域	名称	类型	复位	描述
2	MERASE	R/W	0	整体擦除Flash存储器 该位用于整体擦除Flash主存储器，并监视擦除过程。 值 描述 0 写 0 对该位状态没有影响。 读取该位为 0 表示之前的整体擦除操作完成。 1 置位该位可以擦除Flash主存储器。 读取该位为 1 表示之前的整体擦除操作没有完成。 要了解擦除时间，请参考 “Flash Memory and EEPROM”（1053页）。
1	ERASE	R/W	0	擦除Flash存储器的页 该位用于擦除Flash存储器的页，并监视擦除过程。 值 描述 0 写 0 对该位状态没有影响。 读取该位为 0 表示之前的页擦除操作完成。 1 置位该位可以擦除 Flash 存储器中由 FMA 寄存器内容指定的页。 读取该位为 1 表示之前的页擦除操作没有完成。 要了解擦除时间，请参考 “Flash Memory and EEPROM”（1053页）。
0	WRITE	R/W	0	写一个字到Flash存储器 该位用于写一个字到Flash存储器，并监视写入过程。 值 描述 0 写 0 对该位状态没有影响。 读取该位为 0 表示之前的写入更新操作完成。 1 置位该位可以将 FMD 寄存器的值存储到 Flash 存储器中 FMA 寄存器内容指定的位置。 读取该位为 1 表示写入更新操作没有完成。 要了解编程信息，请参考 “Flash Memory and EEPROM”（1053页）。

寄存器 4: Flash 控制器原始中断状态（FCRIS），偏移量 0x00C

该寄存器指示Flash控制器有一个中断状态。如果 FCIM 寄存器中的某位被置位，那么相应的中断就会发送到中断控制器。

Flash 控制器原始中断状态 (FCRIS)

基址 0x400F.D000
偏移量 0x00C
类型 RO, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:14	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
13	PROGRIS	RO	0	程序验证错误原始中断状态 值 描述 0 未产生中断 1 因为程序操作失败，所以会产生一个挂起的中断。当使用 Flash 写入缓冲区时发生错误，软件必须监测受影响的字，以确定哪里发生了错误。 将 FCMISC 寄存器中的 PROGMISC 位置位可以将该位清零。
12	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
11	ERRIS	RO	0	擦除验证错误原始中断状态 值 描述 0 未产生中断 1 因为擦除操作验证失败，所以会产生一个挂起的中断。当使用 Flash 写入缓冲区时发生错误，软件必须监测受影响的字，以确定哪里发生了错误。 向 FCMISC 寄存器中的 ERMISC 位写 1 可以将该位清零。
10	INVDRIS	RO	0	无效数据原始中断状态 值 描述 0 未产生中断 1 某个位之前编程为 0，而现在的操作试图将其编程为 1，所以会产生一个挂起的中断。 向 FCMISC 寄存器中的 INVMISC 位写 1 可以将该位清零。

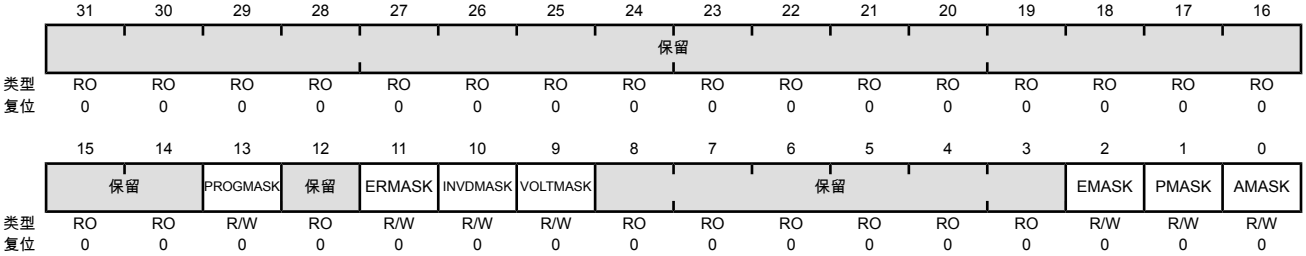
位/域	名称	类型	复位	描述
9	VOLTRIS	RO	0	泵激电压原始中断状态 值 描述 0 未产生中断 1 在 Flash 操作过程中，泵的调制电压超出范围，操作终止，所以会产生一个挂起的中断。 向 FCMISC 寄存器中的 VOLTMISC 位写 1 可以将该位清零。
8:3	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
2	ERIS	RO	0	EEPROM 原始中断状态 该位提供 EEPROM 操作状态。 值 描述 0 未产生 EEPROM 中断。 1 发生了 EEPROM 中断。 向 FCMISC 寄存器中的 EMISC 位写 1 可以将该位清零。
1	PRIS	RO	0	编程的原始中断状态 该位给出编程周期的状态，编程周期是指通过 FMC 或者 FMC2 寄存器位（请参考 475页 和 483页）产生的写入或擦除操作。 值 描述 0 编程周期或者擦除周期没有完成。 1 编程周期或者擦除周期已经完成。 当 FCIM 寄存器的 PMASK 位置位时，该位状态会发送给中断控制器。 向 FCMISC 寄存器中的 PMISC 位写 1 可将该位清零。
0	ARIS	RO	0	访问的原始中断状态 值 描述 0 没有不合适的编程或擦除试图访问存储器。 1 试图对存储器块进行的编程或擦除行为与 FMPPEn 寄存器设定的保护策略相矛盾。 当 FCIM 寄存器的 AMASK 位置位时，该位状态会发送给中断控制器。 向 FCMISC 寄存器的 AMISC 位写 1 可将该位清零。

寄存器 5: Flash 控制器中断屏蔽（FCIM），偏移量 0x010

该寄存器控制Flash控制器是否向控制器产生中断。

Flash 控制器中断屏蔽 (FCIM)

基址 0x400F.D000
偏移量 0x010
类型 R/W, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:14	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
13	PROGMASK	R/W	0	PROGVER 中断屏蔽 值 描述 0 PROGRIS 中断被抑制，不会发送到中断控制器。 1 当 PROGRIS 位置位时，向中断控制器发送一个中断。
12	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
11	ERMASK	R/W	0	ERVER 中断屏蔽 值 描述 0 ERRIS 中断被抑制，不会发送到中断控制器。 1 当 ERRIS 位置位时，向中断控制器发送一个中断。
10	INVDMASK	R/W	0	无效数据中断屏蔽 值 描述 0 INVDRIS 中断被抑制，不会发送到中断控制器。 1 当 INVDRIS 位置位时，向中断控制器发送一个中断。
9	VOLTMASK	R/W	0	VOLT 中断屏蔽 值 描述 0 VOLTRIS 中断被抑制，不会发送到中断控制器。 1 当 VOLTRIS 位置位时，向中断控制器发送一个中断。
8:3	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。

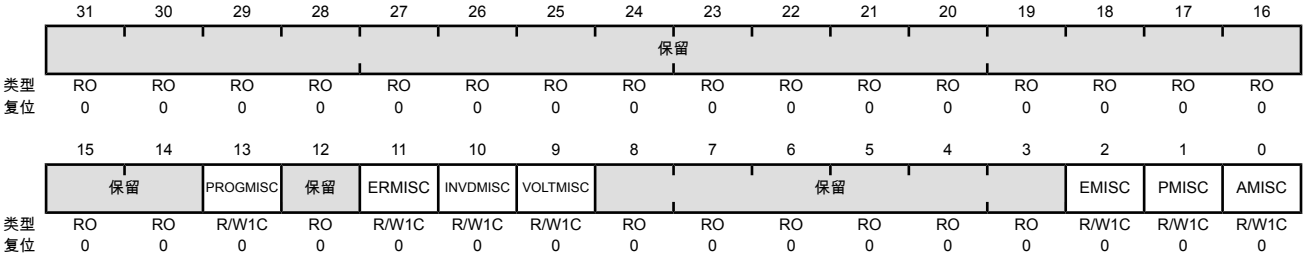
位/域	名称	类型	复位	描述
2	EMASK	R/W	0	EEPROM 中断屏蔽 值 描述 0 ERIS 中断被抑制，不会发送到中断控制器。 1 当 ERIS 位置位时，向中断控制器发送一个中断。
1	PMASK	R/W	0	编程中断屏蔽 该位控制着编程原始中断状态向中断控制器的报告。 值 描述 0 PRIS 中断被抑制，不会发送到中断控制器。 1 当 PRIS 位置位时，向中断控制器发送一个中断。
0	AMASK	R/W	0	访问中断屏蔽 该位控制着访问原始中断状态向中断控制器的报告。 值 描述 0 ARIS 中断被抑制，不会发送到中断控制器。 1 当 ARIS 位置位时，向中断控制器发送一个中断。

寄存器 6: Flash控制器可屏蔽中断的状态和清除（FCMISC），偏移量 0x014

该寄存器提供两个功能。首先，它可以通过指示哪个中断源或哪些中断源正在发出中断信号来报告中断产生的原因。其次，它提供清除中断报告的办法。

Flash控制器可屏蔽中断的状态和清除 (FCMISC)

基址 0x400F.D000
偏移量 0x014
类型 R/W1C, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:14	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
13	PROGMISC	R/W1C	0	PROGVER 屏蔽中断状态和清除 值 描述 0 读取该位为 0 表示中断没有发生。 写 0 对该位状态没有影响。 1 读取该位为 1 表示发出了非屏蔽中断信号。 对这个位写 1 清零 PROGMISC 位以及 FCRIS 寄存器的 PROGRIS 位（参考 477页）。
12	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
11	ERMISC	R/W1C	0	ERVER 屏蔽中断状态和清除 值 描述 0 读取该位为 0 表示中断没有发生。 写 0 对该位状态没有影响。 1 读取该位为 1 表示发出了非屏蔽中断信号。 对这个位写 1 清零 ERMISC 位以及 FCRIS 寄存器的 ERRIS 位（参考 477页）。
10	INVDMISC	R/W1C	0	无效数据屏蔽中断状态和清除 值 描述 0 读取该位为 0 表示中断没有发生。 写 0 对该位状态没有影响。 1 读取该位为 1 表示发出了非屏蔽中断信号。 对这个位写 1 清零 INVDMISC 位以及 FCRIS 寄存器的 INVDRIS 位（参考 477页）。

位/域	名称	类型	复位	描述
9	VOLTMISC	R/W1C	0	VOLT 屏蔽中断状态和清除 值 描述 0 读取该位为 0 表示中断没有发生。 写 0 对该位状态没有影响。 1 读取该位为 1 表示发出了非屏蔽中断信号。 对这个位写 1 清零 VOLTMISC 位以及 FCRIS 寄存器的 VOLTRIS 位（参考 477页）。
8:3	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
2	EMISC	R/W1C	0	EEPROM 屏蔽中断状态和清除 值 描述 0 读取该位为 0 表示中断没有发生。 写 0 对该位状态没有影响。 1 读取该位为 1 表示发出了非屏蔽中断信号。 对这个位写 1 清零 EMISC 位以及 FCRIS 寄存器的 ERIS 位（参考 477页）。
1	PMISC	R/W1C	0	编程可屏蔽的中断状态和清除 值 描述 0 当读取该位为0时，表示没有发生编程周期完成中断。 写 0 对该位状态没有影响。 1 当读取该位为1时，表示一个非屏蔽中断信号被发出，原因是一个编程周期完成。 对这个位写 1 清零 PMISC 位以及 FCRIS 寄存器的 PRIS 位（参考 477页）。
0	AMISC	R/W1C	0	访问可屏蔽的中断状态和清除 值 描述 0 读取该位为 0 表示不适当中断没有发生。 写 0 对该位状态没有影响。 1 当该位读取为 1 时，表示发出了一个非屏蔽中断信号，原因是试图对存储器块进行的编程或擦除行为与 FMPPEn 寄存器为该存储器块设置的保护策略相矛盾。 对这个位写 1 清零 AMISC 位以及 FCRIS 寄存器的 ARIS 位（参考 477页）。

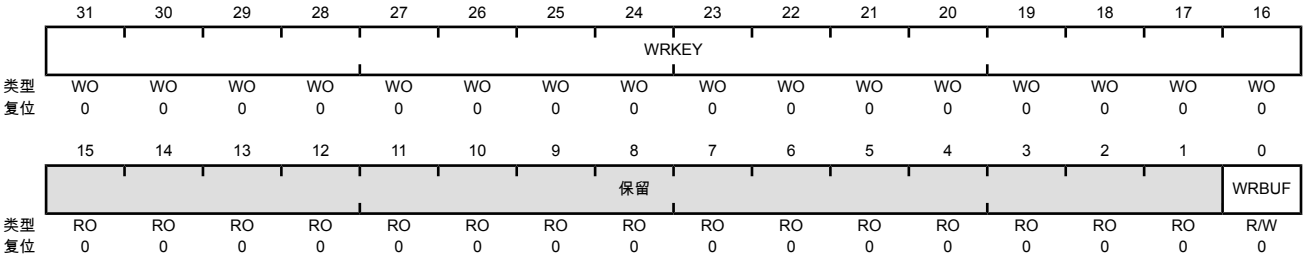
寄存器 7: Flash存储器控制2（FMC2），偏移量 0x020

当该寄存器被写入时，Flash 存储器控制器将对 Flash 存储器地址 (FMA) 寄存器指定的位置启动适当周期数目的访问操作（请参阅473页）。如果访问是写操作，那么 Flash 写缓冲器 (FWB) 寄存器中的数据将会写入指定地址。

该寄存器是启动存储器操作过程中最后被写入的寄存器。

Flash存储器控制2 (FMC2)

基址 0x400F.D000
偏移量 0x020
类型 R/W, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:16	WRKEY	WO	0x0000	Flash存储器写密钥 该域包含一个写密钥，用于最大限度的减少对Flash的意外写入。要写入 Flash 存储器，必须将值 0xA442 或 0x71D5 写入此域中，具体取决于 BOOTCFG 寄存器 KEY 位的值。如果没有这个 WRKEY 值，那么向 FMC2 寄存器中的写入操作将被忽略。读取该域将返回0。
15:1	保留	RO	0x000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
0	WRBUF	R/W	0	缓冲的 Flash 存储器写入 被缓存的Flash存储器写入 值 描述 0 写 0 对该位状态没有影响。 当读取该位为0时，表示之前的被缓存的Flash存储器写访问完成。 1 置位该位将会把 FWBn 寄存器中存储的数据写入到 FMA 寄存器内容所指定的位置。 当读取该位为1时，表示之前的被缓存的Flash存储器写访问没有完成。 要了解编程信息，请参考 “Flash Memory and EEPROM”（1053页）。

寄存器 8: Flash 写缓冲器有效（FWBVAL），偏移量 0x030

该寄存器的位显示了哪个 FWBn 寄存器自从上次 Flash 存储器写缓冲器的写操作后，又被处理器写入过。值为1的寄存器会在下一次Flash存储器写缓冲器写操作中被写入。该寄存器在写操作后被硬件清零。写操作中违反保护也会清该状态。

在被写入操作清零之后，软件可以通过将 FWB[n] 位置 1 来将同样的 32 个字编程到不同的 Flash 存储器位置。于是接着的写操作将使用与上次写操作相同的数据。另外，如果某个 FWBn 寄存器的改变不应该写入 Flash 存储器，当下一次写操作发生时，软件可以清零相应的 FWB[n] 位以保留当前的数据。

Flash 写缓冲器有效 (FWBVAL)

基址 0x400F.D000
偏移量 0x030
类型 R/W, 复位 0x0000.0000

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	FWB[n]															
类型	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	FWB[n]															
类型	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位/域	名称	类型	复位	描述
31:0	FWB[n]	R/W	0x0	Flash存储器写缓冲器

值	描述
0	相应的 FWBn 寄存器没有被写入新的数据。
1	相应的 FWBn 寄存器已经在上次缓冲器写操作之后更新，并准备好写入 Flash 存储器。

第 0 位对应 FWB0，偏移量 0x100，第 31 位对应 FWB31，偏移量 0x13C。

寄存器 9: Flash 写缓冲器n（FWBn），偏移量 0x100 - 0x17C

这32个寄存器包含的数据将在一次被缓存的Flash存储器写操作中写入Flash存储器。偏移量选择其中的一个32位寄存器。只有自上一次缓存 Flash 存储器写操作后被更新的 FWBn 寄存器才可以写入 Flash 存储器，所以没有必要为了写 1 或 2 个字而写满所有的寄存器存储区。FWBn 寄存器写入 Flash 存储器时，FWB0 寄存器对应 FMA 中包含的地址。FWB1 写入地址 FMA+0x4，以此类推。注意只有为 0 的数据位才会导致 Flash 存储器被修改。数据位为1将保持Flash存储器位的内容为它之前的值。

Flash 写缓冲器n (FWBn)

基址 0x400F.D000
偏移量 0x100 - 0x17C
类型 R/W, 复位 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	DATA															
类型	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	DATA															
类型	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位/域	名称	类型	复位	描述
31:0	DATA	R/W	0x0000.0000	数据 将要写入Flash存储器的数据。

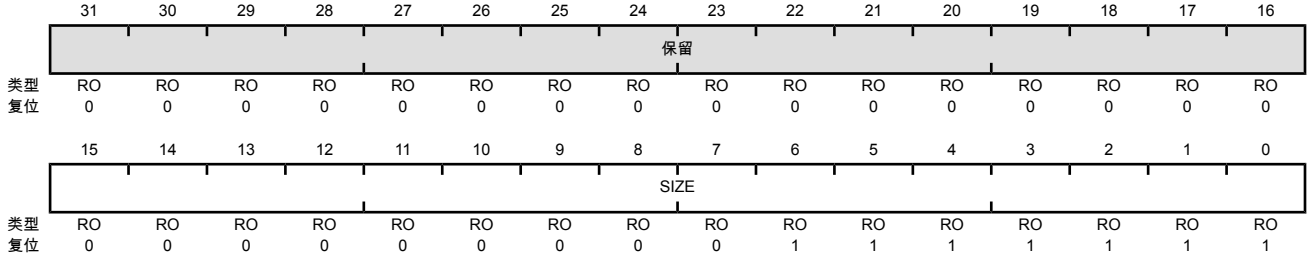
寄存器 10: Flash 容量寄存器（FSIZE），偏移量 0xFC0

该寄存器表示片内 Flash 存储器的容量大小。

重要: 该寄存器应该用来决定微控制器处理的 Flash 存储器容量大小。然而，为了支持传统软件，可以使用 DC0 寄存器。读取 DC0 寄存器能够了解传统存储器的准确容量。软件必须使用 FSIZE 寄存器来确定没有在 DC0 寄存器说明中列出的存储器的大小。

Flash 容量寄存器 (FSIZE)

基址 0x400F.D000
偏移量 0xFC0
类型 RO, 复位 0x0000.007F



位/域	名称	类型	复位	描述
31:16	保留	RO	0x0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
15:0	SIZE	RO	0x7F	Flash大小 表示片内 Flash 存储器的大小。 值 描述 0x007F 256 KB Flash

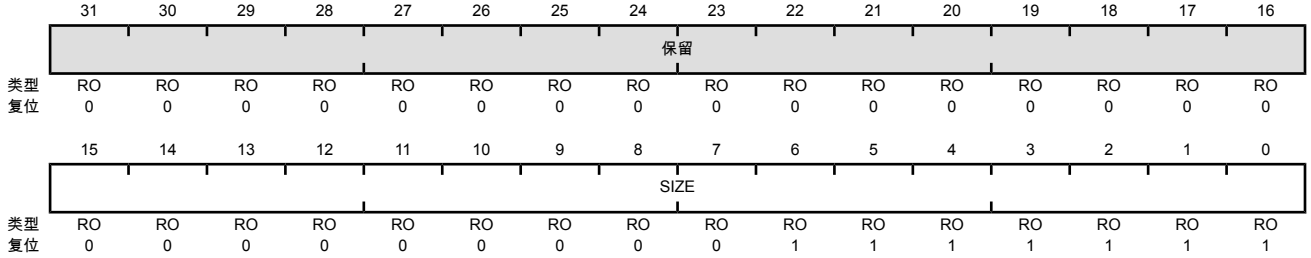
寄存器 11: SRAM 大小寄存器（SSIZE），偏移量 0xFC4

这个寄存器用来指示片内 SRAM 的大小。

重要: 该寄存器应该用来决定微控制器处理的 SRAM 容量大小。然而，为了支持传统软件，可以使用 DC0 寄存器。读取 DC0 寄存器能够了解传统存储器的准确容量。软件必须使用 SSIZE 寄存器来确定没有在 DC0 寄存器说明中列出的存储器的大小。

SRAM 大小寄存器 (SSIZE)

基址 0x400F.D000
偏移量 0xFC4
类型 RO, 复位 0x0000.007F



位/域	名称	类型	复位	描述
31:16	保留	RO	0x0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
15:0	SIZE	RO	0x7F	SRAM大小 显示了片上 SRAM 的大小。 值 描述 0x007F 32 KB SRAM

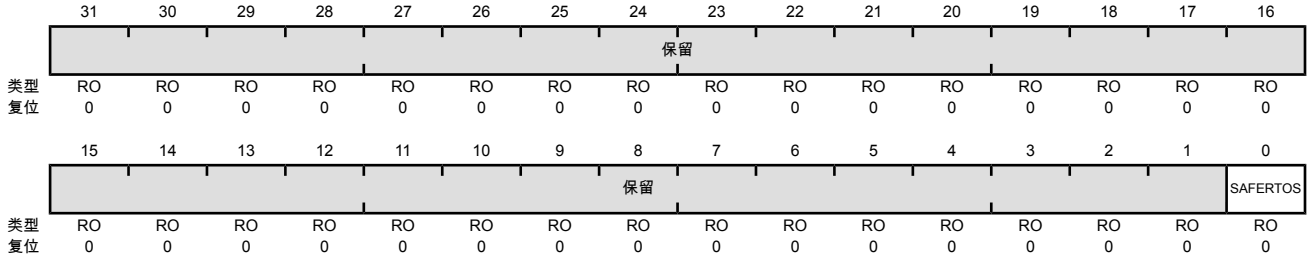
寄存器 12: ROM 软件映射寄存器（ROMSWMAP），偏移量 0xFCC

该寄存器用来指示片内 ROM 中第三方软件的存在。

重要: 该寄存器用来确定微控制器中片内 ROM 中是否存在第三方软件。然而，为了支持传统软件，可以使用 NVMSTAT 寄存器。读取 NVMSTAT 寄存器中的 TPSW 位可以正确地识别是否存在传统第三方软件。软件应使用 ROMSWMAP 寄存器来识别不在传统器件上的软件。

ROM 软件映射寄存器 (ROMSWMAP)

基址 0x400F.D000
偏移量 0xFCC
类型 RO, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:1	保留	RO	0x0000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
0	SAFERTOS	RO	0x0	SafeRTOS 存在 值 描述 0 SafeRTOS 没有在片内 ROM 中。 1 SafeRTOS 在片内 ROM 中。

8.5 EEPROM 寄存器描述（EEPROM 偏移量）

本节按照地址偏移量由小到大的顺序依次详细介绍各 EEPROM 寄存器。本节中的寄存器地址偏移量都是相对于 EEPROM 基址 0x400A.F000 而言的。

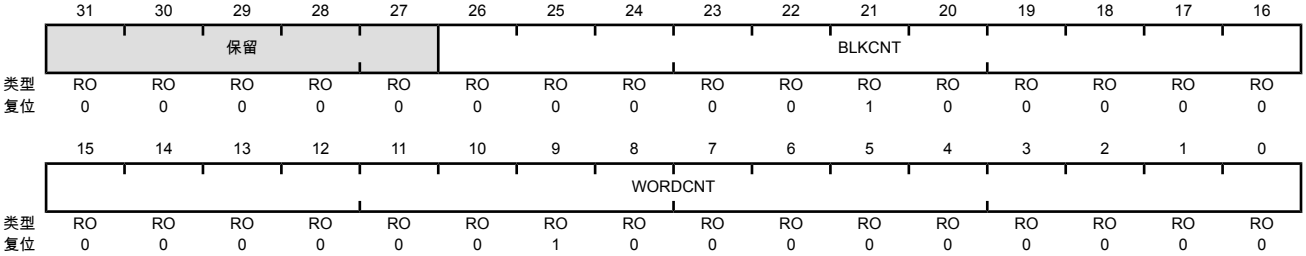
注意配置这些寄存器之前必须启用 EEPROM 模块的时钟（请参考 308页）。在启用 EEPROM 模块时钟以后，必须等待三个系统时钟才能访问 EEPROM 模块的寄存器。另外，启用或者复位 EEPROM 模块之后，软件必须等到 EEDONE 寄存器中的 WORKING 位清零之后才能访问 EEPROM 寄存器。

寄存器 13: EEPROM 大小信息寄存器（EESIZE），偏移量 0x000

EESIZE 寄存器表示 EEPROM 中 16 字块和 32 位字的数量。

EEPROM 大小信息寄存器 (EESIZE)

基址 0x400A.F000
偏移量 0x000
类型 RO, 复位 0x0020.0200



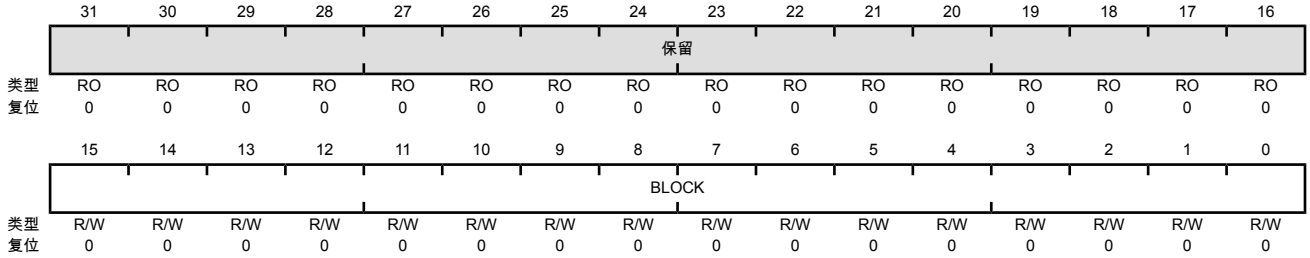
位/域	名称	类型	复位	描述
31:27	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
26:16	BLKCNT	RO	0x20	16 字块的数量 该域中的编码值描述了 EEPROM 中的 16 字块的数量。
15:0	WORDCNT	RO	0x200	32 位字的数量 该域中的编码值描述了 EEPROM 中的 32 位字的数量。

寄存器 14: EEPROM 当前块寄存器（EEBLOCK），偏移量 0x004

EEBLOCK 寄存器用于为随后的读取、写入和保护控制选择 EEPROM 块。该值是一个 EEPROM 的块偏移量，因此第一个块是 0，第二个块是 1，以此类推。每个块包含 16 个字。试图设置无效块会引起 BLOCK 域配置为 0。软件可以在 BLOCK 域被写入后对之进行读取，以验证被访问的块是否有效。不存在的块和通过 EEHIDE 寄存器隐藏的块都属于无效块。请注意，块 0 不能被隐藏。

EEPROM 当前块寄存器 (EEBLOCK)

基址 0x400A.F000
偏移量 0x004
类型 R/W, 复位 0x0000.0000



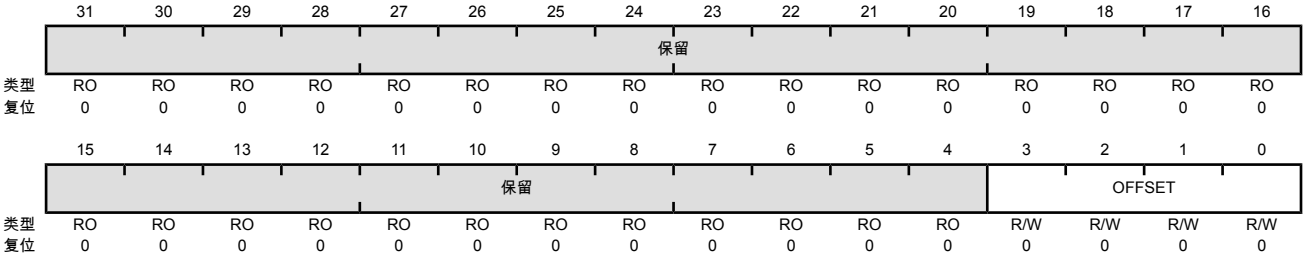
位/域	名称	类型	复位	描述
31:16	保留	RO	0x00000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
15:0	BLOCK	R/W	0x0000	当前块 该域指定随后访问的 EEPROM 中的块。一旦该域被配置，读写寄存器将使用 EEOFFSET 寄存器选择块中的字指向指定的块。另外，保护和解锁寄存器用于选定的块。该寄存器能够写入的最大值由块的数量决定，EESIZE 寄存器会显示这个值。如果写入的值大于块的最大数量或者对锁定的块写入都会引起该域变成 0。

寄存器 15: EEPROM 当前寄存器（EEOFFSET），偏移量 0x008

EEOFFSET 寄存器用于选择由 EEBLOCK 寄存器指定的块中待读取或写入的 EEPROM 字。该值是块中的字偏移量。因为访问 EERDWRINC 寄存器会改变该偏移量，所以软件可以通过读取这个寄存器的值来判断当前的偏移量。

EEPROM 当前寄存器 (EEOFFSET)

基址 0x400A.F000
偏移量 0x008
类型 R/W, 复位 0x0000.0000



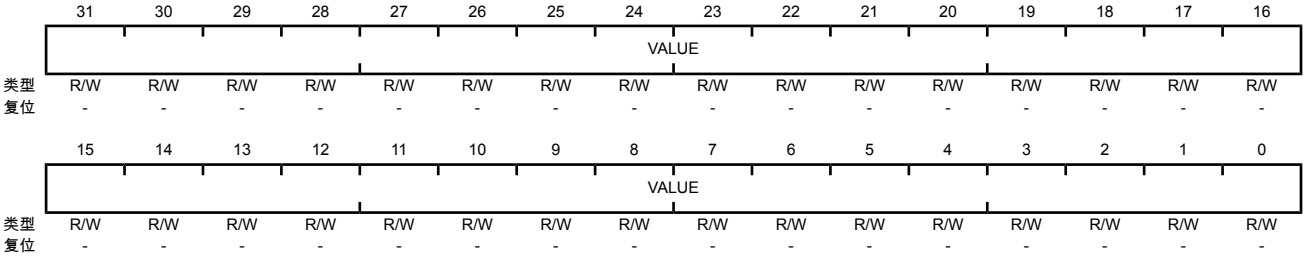
位/域	名称	类型	复位	描述
31:4	保留	RO	0x0000.000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
3:0	OFFSET	R/W	0x0	当前地址偏移量 该值是EEBLOCK寄存器所指定块的当前地址的偏移量。配置之后，读写寄存器EERDRWR和EERDWRINC都将根据那个地址工作。偏移量会由 EERDWRINC 寄存器自动递增并在块中循环，这意味着偏移量会从 15 变成 0。

寄存器 16: EEPROM 读写寄存器（EERDWR），偏移量 0x010

EERDWR 寄存器用来读取或者写入 EEBLOCK 和 EEOFFSET 寄存器指向的 EEPROM 字。如果保护或者访问规则不允许访问，那么该操作处理如下：如果不允许读取，任何情况都返回值 0xFFFF.FFFF；如果不允许写入，EEDONE 被配置为显示错误。

EEPROM 读写寄存器 (EERDWR)

基址 0x400A.F000
偏移量 0x010
类型 R/W, 复位 -



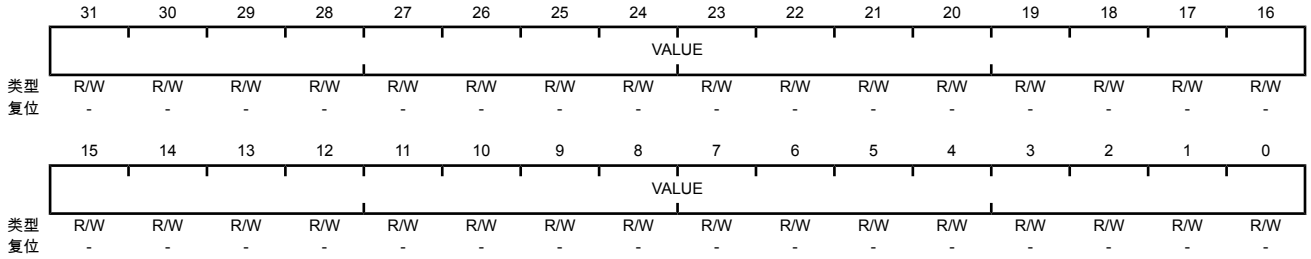
位/域	名称	类型	复位	描述
31:0	VALUE	R/W	-	EEPROM 读取或写入数据 读取时，该域包含 EEOFFSET 指向的字中的数值。写入时，该域包含将要存储到 EEOFFSET 指向的字中的数值。对于写操作，配置该域会开始写入过程。如果保护和访问规则不允许读取，那么返回的数值全部为 1。如果保护和访问规则不允许写入，那么写入操作失败，而且 EEDONE 寄存器会显示这个失败。

寄存器 17: EEPROM 读写加 1 寄存器（EERDWRINC），偏移量 0x014

EERDWR 寄存器用来读取或者写入 EEBLOCK 和 EEOFFSET 寄存器指向的 EEPROM 字，然后使 EEOFFSET 寄存器中的 OFFSET 域递增。如果保护或者访问规则不允许访问，那么该操作处理如下：如果不允许读取，任何情况都返回值 0xFFFF.FFFF；如果不允许写入，EEDONE 被配置为显示错误。不管什么情况，OFFSET 域都会递增。如果到达最后数值，那么 OFFSET 回到 0，指向第一个字。

EEPROM 读写加 1 寄存器 (EERDWRINC)

基址 0x400A.F000
偏移量 0x014
类型 R/W, 复位 -



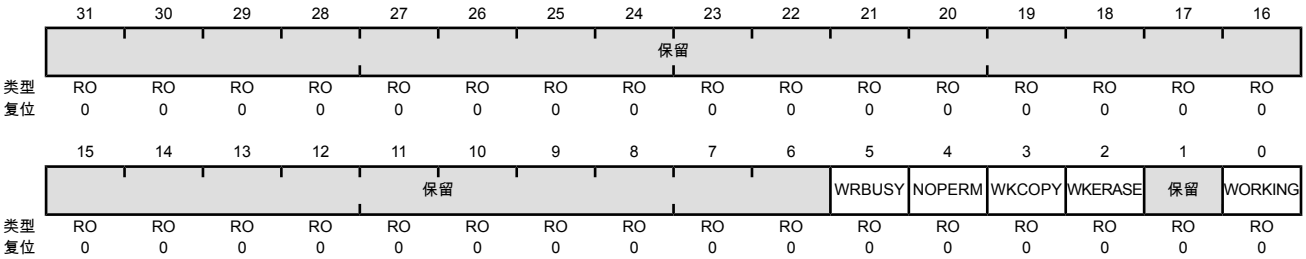
位/域	名称	类型	复位	描述
31:0	VALUE	R/W	-	EEPROM 以递增的方式读取或写入数据 读取时，该域包含 EEOFFSET 指向的字中的数值。写入时，该域包含将要存储到 EEOFFSET 指向的字中的数值。对于写操作，配置该域会开始写入过程。如果保护和访问规则不允许读取，那么返回的数值全部为 1。如果保护和访问规则不允许写入，那么写入操作失败，而且 EEDONE 寄存器会显示这个失败。 不管有没有错误，EEOFFSET 寄存器中的 OFFSET 域将递增 1，如果达到最大值，该数值会返回重新计数。

寄存器 18: EEPROM 完成状态寄存器（EEDONE），偏移量 0x018

EEDONE 寄存器显示以下操作是否成功：使用 EERDWR 或者 EERDWRINC 寄存器进行的写入操作、使用 EEPROT 寄存器进行的保护设置、使用 EEPASS 寄存器进行的密码记录、使用 EESUPP 寄存器进行的复制缓冲区擦除或程序重试以及使用 EEDBGME 寄存器进行的调试整体擦除。EEDONE 寄存器可以和 EEINT 寄存器一起产生中断，报告状态。普通用法是查询 EEDONE 寄存器或者在中断触发之后读取这个寄存器。EEDONE 位 0 被置位时，表示该操作仍在进行中。如果 EEDONE 位 0 清零，那么 EEDONE 中的值表示完成状态。如果 EEDONE 为 0，那么写入操作成功完成。如果 EEDONE!=0，那么发生错误，并由已置位的位给出错误源。如果发生错误，应采取纠正措施，请参考 496页。

EEPROM 完成状态寄存器 (EEDONE)

基址 0x400A.F000
偏移量 0x018
类型 RO, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:6	保留	RO	0x0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
5	WRBUSY	RO	0	写忙碌标志 值 描述 0 无错误 1 当写入正在进行时尝试访问 EEPROM。
4	NOPERM	RO	0	没有许可的写入操作 值 描述 0 无错误 1 没有许可的情况下试图写入。由于块被锁定，写入操作与设定的访问保护相违背，或者在写入密码后再次尝试写入密码，都会产生错误。
3	WKCOPY	RO	0	进行复制 值 描述 0 EEPROM 没有复制。 1 写入操作正在进行，等到EEPROM 向或者从复制缓冲区复制数据。

位/域	名称	类型	复位	描述
2	WKERASE	RO	0	进行擦除 值 描述 0 EEPROM 没有擦除。 1 写入操作正在进行，最初的块在复制之后正在被擦除。
1	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
0	WORKING	RO	0	EEPROM 忙 值 描述 0 EEPROM 没有工作。 1 EEPROM 正在执行请求的操作。

寄存器 19: EEPROM 支持控制和状态寄存器（EESUPP），偏移量 0x01C

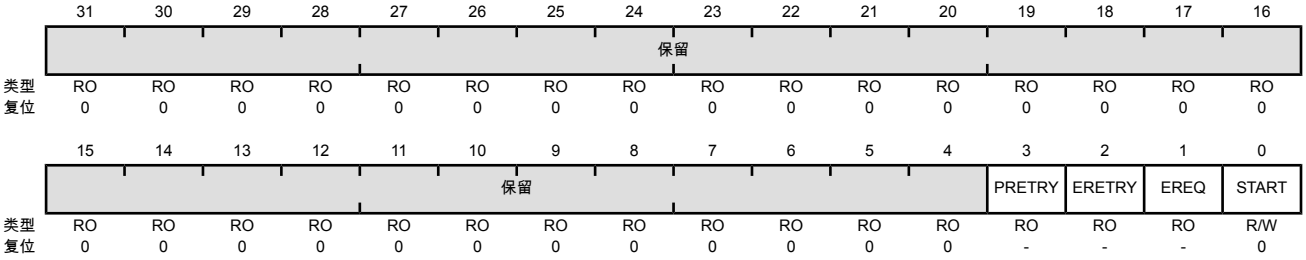
EESUPP 寄存器显示了是否因为以下原因而需要内部操作：内部复制缓冲区必须被擦除，或者发生了编程失败使操作必须终止。这些条件在下面做了解释，详情可以参考 “手动复制缓冲区擦除”一节（469页） 和 “编程中的错误”一节（469页）。

- 如果内部复制缓冲区由于装满，在下一次应用时必须被擦除，那么 EREQ 位被置位。为了避免下次写入时等待复制缓冲区被擦除的延迟，可以将该寄存器的 START 位置位，手动擦除缓冲区。
- 如果 PRETRY 或 ERETRY 位置位，则表示必须完成某项操作，将 START 置位可以重新执行该操作。
- 成功完成此前失败的操作后，PRETRY 和 ERETRY 位将自动清零。

这些位在复位时不会改变，所以复位前所有发生的条件在复位后还可以指示当时情况。

EEPROM 支持控制和状态寄存器 (EESUPP)

基址 0x400A.F000
偏移量 0x01C
类型 R/W, 复位 -



位/域	名称	类型	复位	描述
31:4	保留	RO	0x0000.000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
3	PRETRY	RO	-	编程必须重试 值 描述 0 编程没有失败完成。 1 不管哪个方向的复制编程没有完成，必须将 START 位置位，重新进行编程操作。
2	ERETRY	RO	-	擦除必须重试 值 描述 0 擦除没有失败。 1 擦除操作没有完成，必须将 START 位置位，重新进行擦除操作。如果失败的擦除是因为擦除主缓冲区，那么在擦除成功以后再执行一次复制。

位/域	名称	类型	复位	描述
1	EREQ	RO	-	擦除请求 值 描述 0 复制缓冲区有剩余空间。 1 需要复制缓冲区擦除操作。
0	START	R/W	0	开始擦除 如果 PRETRY 或 ERETRY 位被置位，那么将该位置位会启动错误恢复。如果 PRETRY 和 ERETRY 位都被清零，那么将该位置位就会开始擦除复制缓冲区（如果 EREQ 位被置位）。如果该寄存器中的其他位都没有置位，那么将该位置位没有任何用处。该位置位以后，EEDONE 寄存器中的 WORKING 位也被置位，并在操作完成以后清零。另外，EEINT 寄存器可以在操作完成后产生一个中断。 如果操作过程中该位被置位，那么写入无效。 操作完成时，START 位自动被清零。

寄存器 20: EEPROM 解锁寄存器（EEUNLOCK），偏移量 0x020

EEUNLOCK 寄存器可以用密码解锁整个 EEPROM 或者单个块。如果 EEPASSn 寄存器为 EEBLOCK 寄存器选择的块记录了密码，那么就需要解锁操作。如果块 0 有密码，它也会锁定其他块，使其无法访问，但是它可以使用自己的保护机制，比如，可以读取，但是锁定时不能写入。另外，如果块 0 有密码，那么在解锁其他块之前必须先解锁块 0。

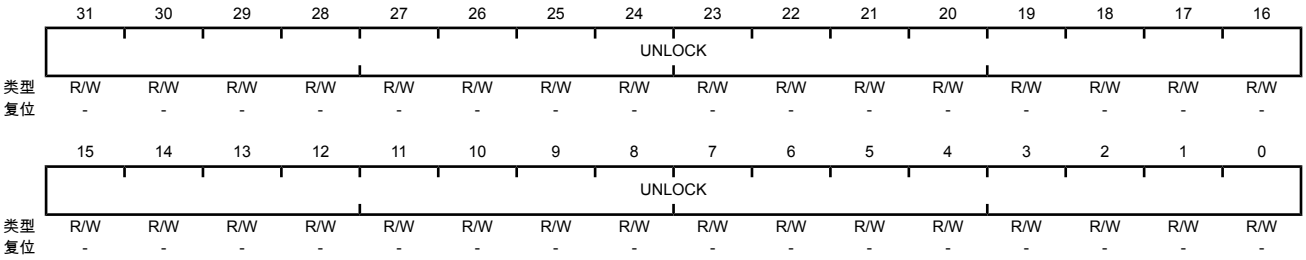
EEUNLOCK 寄存器需要被写入 1 到 3 次，以便形成 EEPASSn 寄存器记录的 32 位、64 位或者 96 位密码。用来配置 EEPASS0 寄存器的值必须最后写入。比如，对于一个 96 位的密码，用来配置 EEPASS2 寄存器的数值必须先写入，然后是 EEPASS1 和 EEPASS0 寄存器的数值。将该寄存器写入 0xFFFF.FFFF 会重新锁定整个 EEPROM 或者块。

如果向该寄存器写入无效数值，那么块保持锁定状态。EEPROM 的锁定状态可以通过读取 EEUNLOCK 寄存器确定。如果设置了多字密码，写入的字的数量不正确，那么将该寄存器写入 0xFFFF.FFFF 会恢复 EEPROM 的锁定状态，此时可以重新尝试适当的解锁顺序。

请注意，内部逻辑会阻止任何试图寻找正确密码或者密码长度的电子或者时间攻击。

EEPROM 解锁寄存器 (EEUNLOCK)

基址 0x400A.F000
偏移量 0x020
类型 R/W, 复位 -



位/域	名称	类型	复位	描述
31:0	UNLOCK	R/W	-	EEPROM 解锁

值	描述
0	EEPROM 被锁定。
1	EEPROM 被解锁。

如果 EEBLOCK 寄存器参考的块有密码，或者主块（块 0）有密码，EEPROM 被锁定。向该寄存器写入密码可以解锁。锁定的块和 EEPROM 在下次锁定和下次复位之前一直被锁定。将该寄存器写入 0xFFFF.FFFF 可以再锁定一次。

寄存器 21: EEPROM 保护寄存器（EEPROT），偏移量 0x030

EEPROT 寄存器用来设定或者读取由 EEBLOCK 寄存器选定的当前块的保护。保护和访问控制用来决定什么时候块的内容可以读写。块 0 的保护级别为整个 EEPROM 设置了最低的保护级别。比如，如果 PROT 域为块 0 设置成了 0x1，那么块 1 的 PROT 域只能是 0x1、0x2 或者 0x3，不能是 0x0。

EEPROM 保护寄存器 (EEPROT)

基址 0x400A.F000
偏移量 0x030
类型 R/W, 复位 0x0000.0000

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
域	保留															
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
域	保留												ACC	PROT		
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位/域	名称	类型	复位	描述
31:4	保留	RO	0x0000.000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
3	ACC	R/W	0	访问控制 值 描述 0 用户和管理代码都可以访问 EEPROM 的此块。 1 只有管理模式才能访问 EEPROM 的块。µDMA 和调试都不能访问 EEPROM。 如果该位置位，那么只有管理模式才能访问块 0。
2:0	PROT	R/W	0x0	保护控制 保护位控制哪些内容需要读取和写入 EEBLOCK 寄存器指定的块，或者如果块 0 被选中，那就是读取和写入所有的块。下面的值都是允许的： 值 描述 0x0 该设置是默认设置。如果没有密码，那么块不受保护，可以读写。 如果有密码，那么块可读，但只有解锁后才能写入。 0x1 如果有密码，那么块只能在解锁时读取或写入。 没有密码时，这个值没有意义。 0x2 如果没有密码，那么块可读，但不能写入。 如果有密码，那么块只在解锁时可读，但在任何情况下都不能写入。 0x3 保留

寄存器 22: EEPROM 密码寄存器（EEPASS0），偏移量 0x034

寄存器 23: EEPROM 密码寄存器（EEPASS1），偏移量 0x038

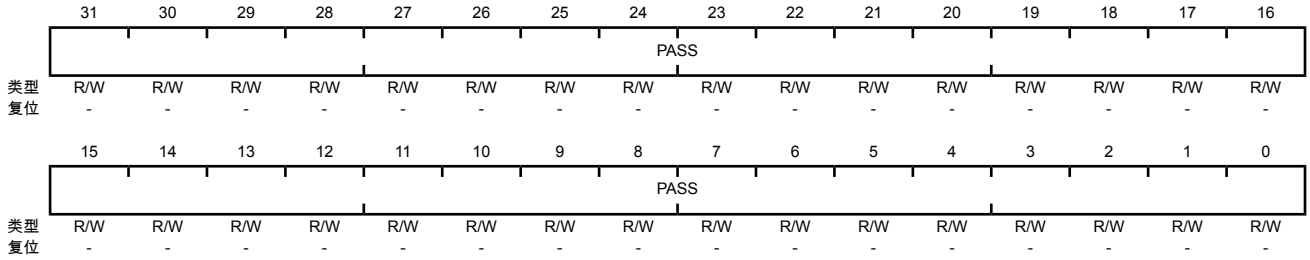
寄存器 24: EEPROM 密码寄存器（EEPASS2），偏移量 0x03C

EEPASSn 寄存器用来为块设置密码。每个密码只能设置一次，而且不能改变。密码可以是 32 位、64 位或者 96 位。每个字密码可以是任意 32 位数值，除了 0xFFFF.FFFF（全为 1）。要设置密码，将 EEPASS0 寄存器写入一个数值，但不能是 0xFFFF.FFFF。写入完成后，如 EEDONE 寄存器所指示的，应用程序可选择向 EEPASS1 寄存器写入一个值，但不能是 0xFFFF.FFFF。写入完成后，应用程序可选择向 EEPASS2 寄存器写入一个值，但不能是 0xFFFF.FFFF，以此创建一个 96 位的密码。这些寄存器不必连续写入，EEPASS1 和 EEPASS2 寄存器可以以后再写入。根据写入寄存器的数量，解锁代码也需要同样数量的字来解锁。

注意: 写入密码之后，块其实还没有被锁定，直到复位或者 EEUNLOCK 被写入 0xFFFF.FFFF 之后才真正被锁定。

EEPROM 密码寄存器（EEPASSn）

基址 0x400A.F000
偏移量 0x034
类型 R/W, 复位 -



位/域	名称	类型	复位	描述
31:0	PASS	R/W	-	密码 如果这个块关联了密码，那么该寄存器读取返回 0x1，如果块没有密码，那么读取返回 0x0。当读取返回 0x0 时，对本寄存器执行写操作就会设定密码。当读取返回 0x1 时，对该寄存器的写入操作被忽略，EEDONE 寄存器中的 NOPERM 被置位。

寄存器 25: EEPROM 中断寄存器（PWM0CTL），偏移量 0x040

EEINT 寄存器用来控制 EEPROM 写入完成以后是否产生中断，EEDONE 寄存器中的值由 0x1 变成其他值说明写入完成。如果该寄存器中的 INT 位被置位，那么当 EEDONE 寄存器中的值从 0x1 变成别的值时（因为 Flash 存储器和 EEPROM 共享中断向量），Flash 控制器原始中断状态 (FCRIS) 寄存器中的 ERIS 位会被置位。

EEPROM 中断寄存器 (PWM0CTL)

基址 0x400A.F000
偏移量 0x040
类型 R/W, 复位 0x0000.0000

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	保留															
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	保留															INT
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位/域	名称	类型	复位	描述
31:1	保留	RO	0x0000.000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
0	INT	R/W	0	中断启用

值	描述
0	未产生中断。
1	当 EEDONE 寄存器由 1 变成 0 或者发生错误时，会产生一个中断。EEDONE 寄存器显示了对偏移量地址写入之后以及对密码和保护位写入之后的状态。

寄存器 26: EEPROM 块隐藏寄存器（EEHIDE），偏移量 0x050

EEHIDE 寄存器用来隐藏一个或者多个块，不包括块 0。一旦被隐藏，该块就不能访问，下一次复位之后才能访问。该模式允许初始化代码访问其他应用程序看不见的数据。该寄存器还提供额外的安全保护，因为在代码或者数据找不到密码。

EEPROM 块隐藏寄存器 (EEHIDE)

基址 0x400A.F000
偏移量 0x050
类型 R/W, 复位 0x0000.0000

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
域	Hn															
类型	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
域	Hn															保留
类型	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位/域	名称	类型	复位	描述
31:1	Hn	R/W	0x0000.000	隐藏块

值 描述

0 相应的块没有隐藏。

1 如果块序号与该位的序号相同，那么这个块被隐藏。隐藏块不能被访问，EEBLOCK 寄存器中的 OFFSET 值不能设定为该块的序号。如果试图将 OFFSET 域配置到隐藏的块，那么 EEBLOCK 寄存器就会被清零。

任何试图将该寄存器中已置位的位清零的操作都会被忽略。

0	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。

寄存器 27: EEPROM 调试整体擦除寄存器（EEDBGME），偏移量 0x080

EEDBGME 寄存器用于整体擦除 EEPROM 块，让其返回到出厂默认状态。该寄存器仅用于调试和测试过程，不用在生产环境中。擦除必须使用安全的方法。首先擦除所有的数据，然后擦除保护机制。该寄存器只能由内核在管理模式下写入。当启用之后，该寄存器也可以由 TM4C1231H6PM 调试控制器写入。这种机制需要密钥，以避免意外使用。请注意，如果擦除过程掉电，必须重新使用该机制来完成操作。永久断电不会暴露受保护的安全数据。

为开始整体擦除，整个寄存器必须写入 0xE37B.0001。在擦除过程中该寄存器读取返回 0x1，擦除完成以后读取返回 0x0。EEDONE 寄存器在擦除开始时设为 0x1，在整体擦除结束或者错误发生时变为 0x0。

注：整体擦除 EEPROM 块表示换位写入计数器也将复位到出厂设置。

EEPROM 调试整体擦除寄存器 (EEDBGME)

基址 0x400A.F000
偏移量 0x080
类型 R/W, 复位 0x0000.0000

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	KEY															
类型	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

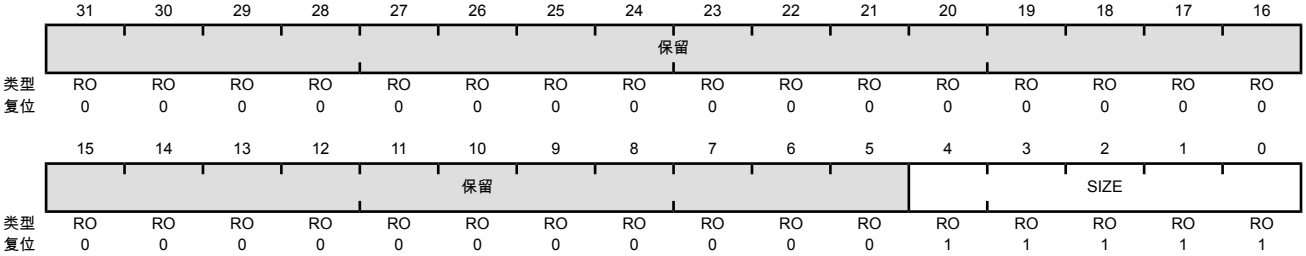
位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	保留															ME
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位/域	名称	类型	复位	描述
31:16	KEY	WO	0x0000	擦除密钥 该域必须写入 0xE37B，ME 域才能有效。
15:1	保留	RO	0x000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
0	ME	R/W	0	整体擦除 值 描述 0 没有动作。 1 该位为 1 时，EEPROM 会被整体擦除。在 EEPROM 整体擦除完成之前，该位会一直为 1。

寄存器 28: EEPROM 外设属性寄存器（EEPROMPP），偏移量 0xFC0

EEPROMPP 寄存器显示了这个部分的 EEPROM 的容量大小。

EEPROM 外设属性寄存器 (EEPROMPP)
基址 0x400A.F000
偏移量 0xFC0
类型 RO, 复位 0x0000.001F



位/域	名称	类型	复位	描述
31:5	保留	RO	0x0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
4:0	SIZE	RO	0x1F	2 KB 大小 EEPROM

8.6 存储器寄存器描述 (系统控制偏移量)

本节的剩余部分按照地址偏移的数字顺序列出并描述了系统控制地址空间中的寄存器。本节中的寄存器地址偏移量都是相对于系统控制基址 0x400F.E000 而言的。

寄存器 29: ROM 控制（RMCTL），偏移量 0x0F0

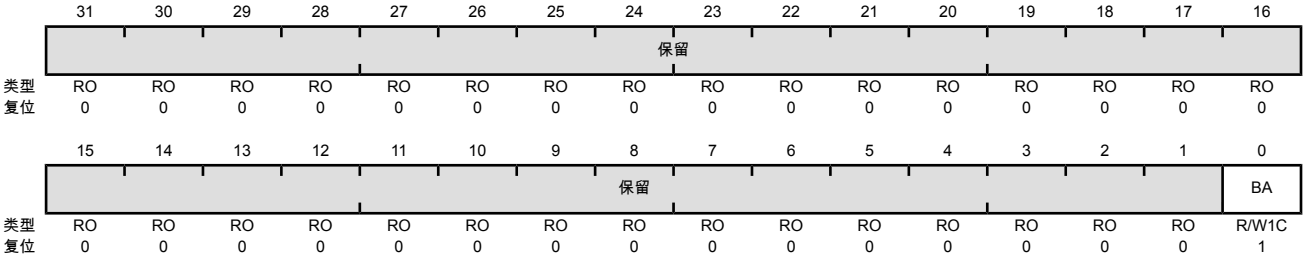
该寄存器提供了对ROM控制器状态的控制。该寄存器的偏移量是相对于系统控制基址 0x400F.E000 而言的。

复位时，将执行以下序列：

1. 读取 BOOTCFG 寄存器。如果 EN 位被清零，那么执行 ROM 的引导装载程序。
2. 在ROM的Boot Loader中，指定的GPIO管脚的状态与规定的极性相比较，如果管脚状态与规定的极性匹配，那么将 ROM 映射到地址 0x0000.0000 并继续执行 ROM 的 Boot Loader。
3. 如果 EN 位被置位或者管脚状态与指定的极性不符，那么地址 0x0000.0004 的数据就会被读取。如果该数据是 0xFFFF.FFFF，那么 ROM 被映射到地址 0x0000.0000，系统继续执行 ROM 引导装载程序。
4. 如果地址 0x0000.0004 中的数据不是 0xFFFF.FFFF，堆栈指针 (SP) 将装载 Flash 存储器地址 0x0000.0000 的数据，程序计数器 (PC) 将装载地址 0x0000.0004 的数据。随后用户应用程序开始执行。

ROM 控制 (RMCTL)

基址 0x400F.E000
偏移量 0x0F0
类型 R/W1C, 复位 -



位/域	名称	类型	复位	描述
31:1	保留	RO	0x0000.000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
0	BA	R/W1C	1	启动别名 值 描述 0 Flash存储器在地址0x0。 1 微控制器的ROM出现在地址0x0。 对该位的位置写入1可以清零该位。

寄存器 30: Flash 存储器保护读取启用寄存器 0（FMPRE0），偏移量 0x130 和 0x200

寄存器 31: Flash 存储器保护读取启用寄存器 1（FMPRE1），偏移量 0x204

寄存器 32: Flash 存储器保护读取启用寄存器 2（FMPRE2），偏移量 0x208

寄存器 33: Flash 存储器保护读取启用寄存器 3（FMPRE3），偏移量 0x20C

注意: FMPPE0 寄存器采用别名，具备向后兼容性。

注意: 偏移量是相对于系统控制基址 0x400F.E000 的。

这个寄存器存放的是每个 2 KB flash 块的只读保护位（FMPPEn 存放只执行位）。

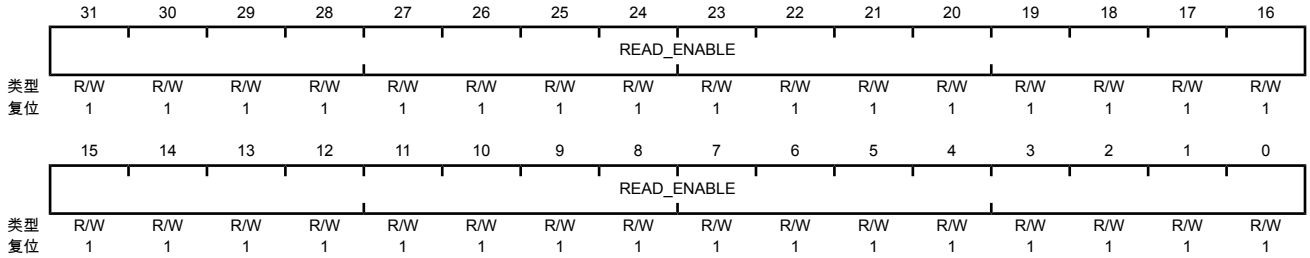
这个寄存器在上电复位序列加载。对所有已执行的 2 KB 模块来说，FMPREn 和 FMPPEn 寄存器在出厂时都被设置为 1。这样可以实现一种开放的访问和编程策略。可以通过写特定的寄存器位来更改这个寄存器的位。但是，这个寄存器是R/W0，用户只能将保护位从1变为0 (不可以从0变为1)。这种改变不是永久的，直到寄存器被提交 (保存)，此时位的改变是永久性的。如果一个位从1变为0且没有提交，那么它可以通过执行一段上电复位序列来恢复。显示的复位值只适用于上电复位，任何其它类型的复位都不会影响这个寄存器。一旦提交，恢复该寄存器出厂默认值的唯一方法是执行“恢复一个“锁死”的微控制器”（180页） 中详述的序列。

每个 FMPREn 寄存器控制一个 64 k 的 Flash 块。有关详细信息，请参阅“Flash 存储器保护”（462页）。

- FMPRE0：0 到 64 KB
- FMPRE1：65 到 128 KB
- FMPRE2：129 到 192 KB
- FMPRE3：193 到 256 KB

Flash 存储器保护读取启用寄存器 n（FMPREn）

基址 0x400F.E000
偏移量 0x130 和 0x200
类型 R/W, 复位 0xFFFF.FFFF



位/域	名称	类型	复位	描述
31:0	READ_ENABLE	R/W	0xFFFF.FFFF	Flash 读使能 每个位可以把 2 KB 的 Flash 块配置成只读。 这些策略可以进行组合，如表8-1（462页） 所示。

寄存器 34: Flash 存储器保护编程启用寄存器 0（FMPPE0），偏移量 0x134 和 0x400

寄存器 35: Flash 存储器保护编程启用寄存器 1（FMPPE1），偏移量 0x404

寄存器 36: Flash 存储器保护编程启用寄存器 2（FMPPE2），偏移量 0x408

寄存器 37: Flash 存储器保护编程启用寄存器 3（FMPPE3），偏移量 0x40C

注意: FMPPE0 寄存器采用别名，具备向后兼容性。

注意: 偏移量是相对于系统控制基址0x400F.E000的。

该寄存器存放的是 2KB flash 块的只执行保护位（FMPREn 存放只读保护位）。

这个寄存器在上电复位序列加载。在 FMPREn 和 FMPPEn 寄存器的出厂设置中，所有已经实现的存储器组所对应的位的值为 1。这样可以实现一种开放的访问和编程策略。可以通过写特定的寄存器位来更改这个寄存器的位。但是，这个寄存器是R/W0，用户只能将保护位从1变为0 (不可以从0变为1)。这种改变不是永久的，直到寄存器被提交 (保存)，此时位的改变是永久性的。如果一个位从1变为0且没有提交，那么它可以通过执行一段上电复位序列来恢复。显示的复位值只适用于上电复位，任何其它类型的复位都不会影响这个寄存器。一旦提交，恢复该寄存器出厂默认值的唯一方法是执行“恢复一个“锁死”的微控制器”（180页）中详述的序列。有关详细信息，请参阅“Flash 存储器保护”（462页）。

每个 FMPPEn 寄存器控制一个 64 k 的 Flash 块。有关详细信息，请参阅“Flash 存储器保护”（462页）。

- FMPPE0：0 到 64 KB
- FMPPE1：65 到 128 KB
- FMPPE2：129 到 192 KB
- FMPPE3：193 到 256 KB

Flash 存储器保护编程启用寄存器 n（FMPPEn）

基址 0x400F.E000
偏移量 0x134 和 0x400
类型 R/W, 复位 0xFFFF.FFFF

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	PROG_ENABLE															
类型	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	PROG_ENABLE															
类型	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

位/域	名称	类型	复位	描述
31:0	PROG_ENABLE	R/W	0xFFFF.FFFF	Flash 编程使能 每个位可以把 2 KB 的 Flash 块配置成只执行。 这些策略可以进行组合，如表8-1（462页）所示。

寄存器 38: 启动配置（BOOTCFG），偏移量 0x1D0

注意: 偏移量是相对于系统控制基址 0x400F.E000 的。

注意: 引导配置 (BOOTCFG) 寄存器需要执行一次上电复位 (POR)，然后才会让提交的更改生效。

此寄存器无法直接写入，而应通过 FMD 寄存器写入，如“非易失性寄存器编程”（465页）所述。该寄存器提供了对GPIO管脚的配置来使能ROM引导装载程序，还提供了一次性写入机制来禁止外部调试器访问设备。复位时，通过使用在该寄存器中配置的 A-Q 端口 GPIO 信号，用户可以选择让内核直接执行 ROM 引导加载程序或执行 Flash 存储器中的应用程序。复位时，将执行以下序列：

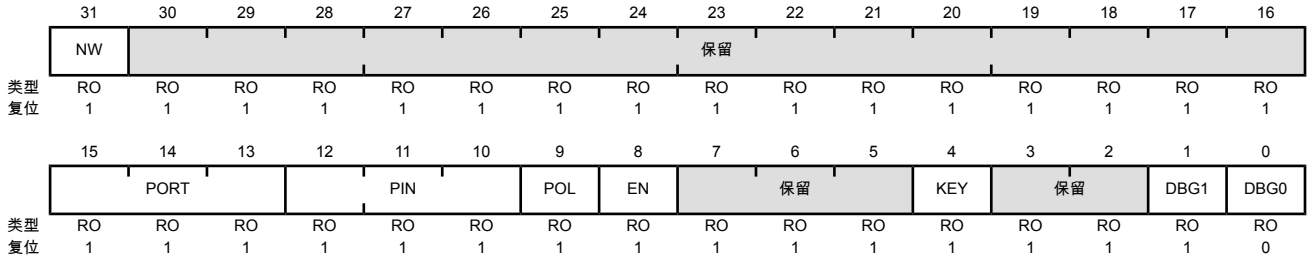
1. 读取 BOOTCFG 寄存器。如果 EN 位被清零，那么执行 ROM 的引导装载程序。
2. 在ROM的Boot Loader中，指定的GPIO管脚的状态与规定的极性相比较，如果管脚状态与规定的极性匹配，那么将 ROM 映射到地址 0x0000.0000 并继续执行 ROM 的 Boot Loader。
3. 如果 EN 位被置位或者管脚状态与指定的极性不符，那么地址 0x0000.0004 的数据就会被读取。如果该数据是 0xFFFF.FFFF，那么 ROM 被映射到地址 0x0000.0000，系统继续执行 ROM 引导装载程序。
4. 如果地址 0x0000.0004 中的数据不是 0xFFFF.FFFF，堆栈指针 (SP) 将装载 Flash 存储器地址 0x0000.0000 的数据，程序计数器 (PC) 将装载地址 0x0000.0004 的数据。随后用户应用程序开始执行。

出厂时 DBG0 位设为 0，DBG1 设为 1，默认启用外部调试器。将 DBG1 位清零后，从下一次设备上电周期开始，任何外部调试器都将无法访问设备。NW 位表示可将寄存器的位从 1 改为 0。

在上电序列后，可以用 FMC 寄存器的 COMT 位来提交确认寄存器的值，然后寄存器的值变为非易失性保存下来。在提交之前，这些位只能由 1 变为 0。寄存器尚未提交时，显示的复位值只适用于上电复位；任何其它类型的复位都不会影响这个寄存器。一旦提交，该寄存器在上电复位时保留自身的值。一旦提交，恢复该寄存器出厂默认值的唯一方法是执行“恢复一个“锁死”的微控制器”（180页）中详述的序列。

启动配置 (BOOTCFG)

基址 0x400F.E000
偏移量 0x1D0
类型 RO, 复位 0xFFFF.FFFE



位/域	名称	类型	复位	描述
31	NW	RO	1	没有被写 置位时，此位表示可将寄存器的值从 1 改为 0。清零时，此位规定不能更改该寄存器的内容。
30:16	保留	RO	0xFFFF	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。

位/域	名称	类型	复位	描述
15:13	PORT	RO	0x7	启动GPIO端口 该域选择复位时使能ROM引导装载程序的GPIO端口。 值 描述 0x0 端口 A 0x1 端口 B 0x2 端口 C 0x3 端口 D 0x4 端口 E 0x5 端口 F 0x6 端口 G 0x7 端口 H
12:10	PIN	RO	0x7	启动GPIO管脚 该域选择复位时使能ROM引导装载程序的GPIO端口引脚编号。 值 描述 0x0 管脚 0 0x1 管脚 1 0x2 管脚 2 0x3 管脚 3 0x4 管脚 4 0x5 管脚 5 0x6 管脚 6 0x7 管脚 7
9	POL	RO	1	启动GPIO极性 该位为1，选择GPIO管脚的高电平在复位时使能ROM引导装载程序。该位为0，选择GPIO管脚的低电平。
8	EN	RO	1	启动GPIO使能 清零该位将使能GPIO管脚在复位时用于使能ROM引导装载程序。若该位被置位时，地址 0x0000.0004 的内容将被检测，来查看 Flash 存储器是否被编程。如果内容不是0xFFFF.FFFF，内核会从Flash存储器执行。如果Flash没有被编程，内核会从ROM执行。
7:5	保留	RO	0x7	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
4	KEY	RO	1	KEY 选择 此位用于选择 0xA442 或 0x71D5 作为 FMC/FMC2 寄存器的 WRKEY 值。 值 描述 0 将 0x71D5 用作 FMC/FMC2 寄存器的 WRKEY 值。使用 0xA442 密钥时，向 FMC/FMC2 寄存器的写入将被忽略。 1 将 0xA442 用作 FMC/FMC2 寄存器的 WRKEY 值。使用 0x71D5 密钥时，向 FMC/FMC2 寄存器的写入将被忽略。

位/域	名称	类型	复位	描述
3:2	保留	RO	0x3	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
1	DBG1	RO	1	调试控制1 为使调试功能可用，DBG1 位必须为 1 并且 DBG0 必须为 0。
0	DBG0	RO	0	调试控制0 为使调试功能可用，DBG1 位必须为 1 并且 DBG0 必须为 0。

寄存器 39: 用户寄存器 0（USER_REG0），偏移量 0x1E0

寄存器 40: 用户寄存器 1（USER_REG1），偏移量 0x1E4

寄存器 41: 用户寄存器 2（USER_REG2），偏移量 0x1E8

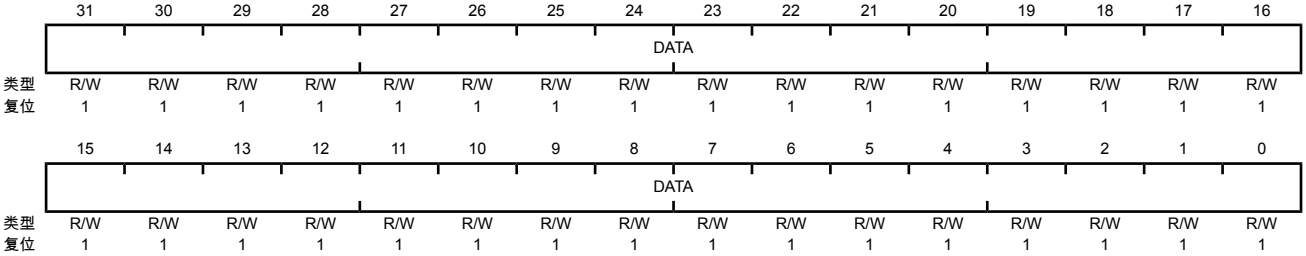
寄存器 42: 用户寄存器 3（USER_REG3），偏移量 0x1EC

注意: 偏移量是相对于系统控制基址 0x400F.E000 的。

每个此类寄存器都可提供 32 位用户定义的非易失性数据。这些寄存器中的位只能由 1 变为 0。寄存器尚未提交时，显示的复位值只适用于上电复位；任何其它类型的复位都不会影响这个寄存器。一旦提交，该寄存器在上电复位时保留自身的值。一旦提交，恢复该寄存器出厂默认值的唯一方法是执行 “恢复一个“锁死”的微控制器”（180页） 中详述的序列。

用户寄存器 n（USER_REGn）

基址 0x400F.E000
偏移量 0x1E0
类型 R/W, 复位 0xFFFF.FFFF



位/域	名称	类型	复位	描述
31:0	DATA	R/W	0xFFFF.FFFF	用户数据 包含用户数据值。该域的位全部初始化为 1，且提交后，该域将通过上电复位保留自身的值。

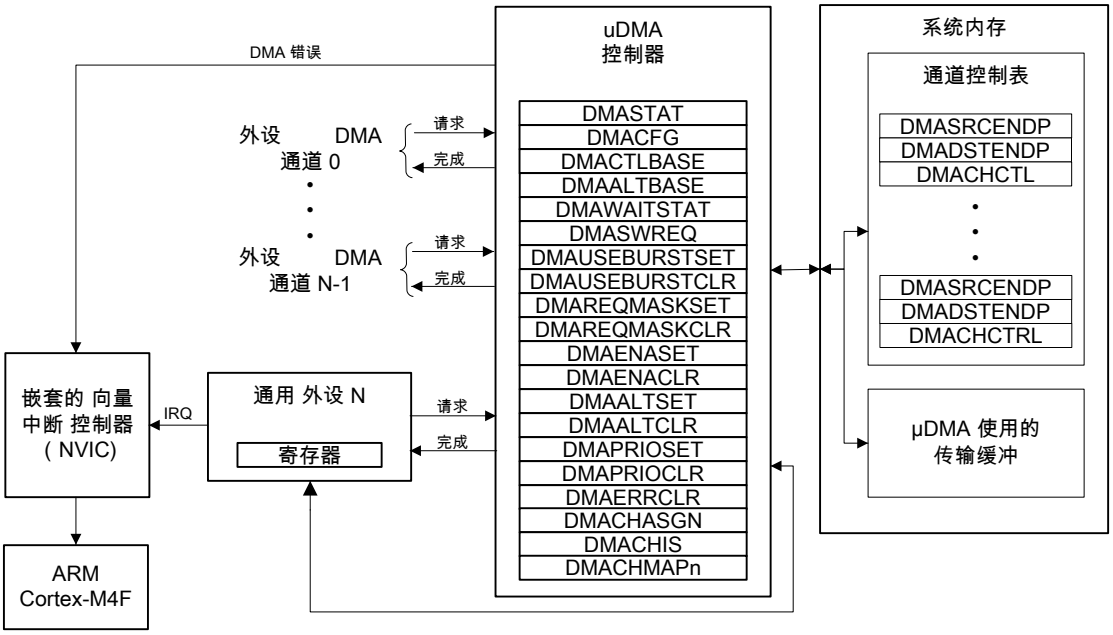
9 微型直接存储器访问（µDMA）

TM4C1231H6PM 微控制器内置一个直接存储器访问（Direct Memory Access，简写为DMA）控制器，我们称之为微型 DMA (µDMA) 控制器。µDMA 控制器所提供的工作方式能够分载 Cortex™-M4F 处理器参与的数据传输任务，从而使处理器得到更加高效的利用和腾出更多的总线带宽。µDMA 控制器能够自动执行存储器与外设之间的数据传输。片上每个支持 µDMA 功能的外设都有专用的 µDMA 通道，通过合理的编程配置，当外设需要时能够自动在外设和存储器之间传输数据。µDMA 控制器具有以下特性：

- ARM® PrimeCell® 32 通道的可配置 µDMA 控制器；
- 支持存储器到存储器、存储器到外设、外设到存储器的 µDMA 传输，包括：
  - 基本模式，用于简单的传输需求
  - 乒乓模式，用于实现持续数据流
  - 散聚模式，借助一个可编程的任务列表，由单个请求触发多达 256 个指定传输
- 高度灵活的可配置的通道配置；
  - 各通道均可独立配置、独立操作
  - 每个支持 µDMA 功能的片上模块都有其专用通道
  - 灵活的通道分配
  - 对于双向模块，为其接收和发送各提供一个通道
  - 专用的软件通道，可由软件启动 µDMA 传输
  - 每通道都可分别配置优先级
  - 可选配置：任一通道均可用作软件启动传输
- 优先级分为两级；
- 通过优化设计，改进了 µDMA 控制器与处理器内核之间的总线访问性能：
  - 当内核不访问总线时，µDMA 控制器即可占用总线
  - RAM条带处理
  - 外设总线分段
- 支持 8 位、16 位或 32 位数据宽度
- 待传输数目可编程为 2 的整数幂，有效范围 1 到 1024
- 源地址及目的地址可自动递增，递增单位可以是字节、半字、字、不递增
- 可屏蔽的外设请求
- 传输结束中断，且每个通道有独立的中断

9.1 结构框图

图 **9-1. µDMA** 结构图



9.2 功能说明

µDMA 控制器是一种使用方便、配置灵活的 DMA 控制器，用于同微控制器的 Cortex-M4F 处理器内核配合以实现高效工作。µDMA控制器支持多种数据宽度以及地址递增机制，各DMA通道之间具有不同的优先级，还提供了多种传输模式，能够通过预编程实现十分复杂的自动传输流程。µDMA 控制器对总线的占用总是次于处理器内核，因此绝不影响处理器的总线会话。由于 µDMA 控制器只会在总线空闲时占用总线，因此它提供的数据传输带宽非常独立，不会影响系统其它部分的正常运行。此外总线架构还经过了优化，增强了处理器内核与 µDMA 控制器高效共享片上总线的能力，从而大大提高了性能。优化的内容包括 RAM 条带处理以及外设总线分段，在大多数情况下允许处理器内核和 µDMA 控制器同时访问总线并执行数据传输。

µDMA 控制器可以将数据转移到片上 SRAM，也可以从片上 SRAM 将数据转移出。但是，由于 Flash 存储器和 ROM 位于不同的内部总线，所以 µDMA 不能从 Flash 存储器或 ROM 转移数据。

µDMA 控制器为每种支持 µDMA 的外设功能都提供了专用的通道，可以各自独立进行配置。µDMA 控制器的配置方法比较独特，是通过系统存储器中的通道控制结构体进行配置的，并且该结构体由处理器维护。除支持简单传输模式之外，µDMA 控制器也支持更加“复杂”的传输模式：在收到某个单次传输请求后，按照建立在存储器中的任务列表，可以执行向/从指定地址发送/接收指定大小数据块的传输流程。µDMA 控制器还支持以乒乓缓冲的方式实现与外设之间的持续数据流。

每个通道还能配置仲裁数目。所谓仲裁数目，是指 µDMA 控制器在重新仲裁总线优先级之前，以猝发方式传输的数据单元数目。借助仲裁数目的配置，当外设产生一个 µDMA 服务请求后，即可精准地操控与外设之间传输多少个数据单元。

9.2.1 通道分配

使用 DMA 通道映射选择 n (DMACHMAPn) 寄存器中的 4 位分配域可以为每个 µDMA 通道分配最多 5 种可能的通道分配方式。

表9-1（514页）示出了 μDMA 通道映射。在“编码”列中示出了各 DMACHMAPn 位域的编码。编码 0x5 - 0xF 是保留的。要支持使用 DMA 通道分配 (DMACHASGN) 寄存器的传统软件，编码0 要与清零的 DMACHASGN 位相等，且 编码1 要与置位的 DMACHASGN 位相等。在读取 DMACHASGN 寄存器时，如果相应的 DMACHMAPn 寄存器位域等于 0，则读取的位域返回值为 0；否则，返回值为 1（如果相应的 DMACHMAPn 寄存器位域不为 0）。表中的类型标识栏用于表示特定外设是使用单个请求 (S)，猝发请求 (B) 还是两者都使用。

注意: 表中注明为“软件”的通道可用于未来进行外设扩展。现在这些通道仅软件可访问，不可连接外设。30号通道是软件专用通道。

表 **9-1. μDMA** 通道分配

编码	0		1		2		3		4	
通道编号	外设	类型	外设	类型	外设	类型	外设	类型	外设	类型
0	软件	B	UART2 RX	SB	软件	B	通用定时器 4A	B	软件	B
1	软件	B	UART2 TX	SB	软件	B	通用定时器 4B	B	软件	B
2	软件	B	通用定时器 3A	B	软件	B	软件	B	软件	B
3	软件	B	通用定时器 3B	B	软件	B	软件	B	软件	B
4	软件	B	通用定时器 2A	B	软件	B	GPIO A	B	软件	B
5	软件	B	通用定时器 2B	B	软件	B	GPIO B	B	软件	B
6	软件	B	通用定时器 2A	B	UART5 RX	SB	GPIO C	B	软件	B
7	软件	B	通用定时器 2B	B	UART5 TX	SB	GPIO D	B	软件	B
8	UART0 RX	SB	UART1 RX	SB	软件	B	通用定时器 5A	B	软件	B
9	UART0 TX	SB	UART1 TX	SB	软件	B	通用定时器 5B	B	软件	B
10	SSI0 RX	SB	SSI1 RX	SB	UART6 RX	SB	GPWideTimer 0A	B	软件	B
11	SSI0 TX	SB	SSI1 TX	SB	UART6 TX	SB	GPWideTimer 0B	B	软件	B
12	软件	B	UART2 RX	SB	SSI2 RX	SB	GPWideTimer 1A	B	软件	B
13	软件	B	UART2 TX	SB	SSI2 TX	SB	GPWideTimer 1B	B	软件	B
14	ADC0 SS0	B	通用定时器 2A	B	SSI3 RX	SB	GPIO E	B	软件	B
15	ADC0 SS1	B	通用定时器 2B	B	SSI3 TX	SB	GPIO F	B	软件	B
16	ADC0 SS2	B	软件	B	UART3 RX	SB	GPWideTimer 2A	B	软件	B
17	ADC0 SS3	B	软件	B	UART3 TX	SB	GPWideTimer 2B	B	软件	B
18	通用定时器 0A	B	通用定时器 1A	B	UART4 RX	SB	GPIO B	B	软件	B
19	通用定时器 0B	B	通用定时器 1B	B	UART4 TX	SB	软件	B	软件	B
20	通用定时器 1A	B	软件	B	UART7 RX	SB	软件	B	软件	B
21	通用定时器 1B	B	软件	B	UART7 TX	SB	软件	B	软件	B
22	UART1 RX	SB	软件	B	软件	B	软件	B	软件	B
23	UART1 TX	SB	软件	B	软件	B	软件	B	软件	B
24	SSI1 RX	SB	ADC1 SS0	B	软件	B	GPWideTimer 3A	B	软件	B
25	SSI1 TX	SB	ADC1 SS1	B	软件	B	GPWideTimer 3B	B	软件	B
26	软件	B	ADC1 SS2	B	软件	B	GPWideTimer 4A	B	软件	B
27	软件	B	ADC1 SS3	B	软件	B	GPWideTimer 4B	B	软件	B
28	软件	B	软件	B	软件	B	GPWideTimer 5A	B	软件	B
29	软件	B	软件	B	软件	B	GPWideTimer 5B	B	软件	B
30	软件	B	软件	B	软件	B	软件	B	软件	B
31	保留	B	保留	B	保留	B	保留	B	保留	B

9.2.2 优先级

每个通道 µDMA 的优先级由通道的序号以及通道的优先级标志位所决定。第0号µDMA通道的优先级最高；通道的序号越大，其优先级越低。每个 µDMA 通道都有一个可设置的优先级标志位，由此可分为默认优先级和高优先级。若某个通道的优先级位置位，则该通道将具有高优先级，其优先于所有未将此标志位置位的通道。假如有多个通道都设为高优先级，那么仍将按照通道序号区分其相互的优先级。

通道的优先级位可通过 DMA 通道优先置位 (DMAPRIOSET) 寄存器置位，通过 DMA 通道优先清除 (DMAPRIOCLR) 寄存器清零。

9.2.3 仲裁数目

当某个 µDMA 通道请求传输时，µDMA 控制器将对所有发出请求的通道进行仲裁，并且向其中优先级最高的通道提供服务。一旦开始传输后，将持续传输一定数量的数据，之后再对发出请求的通道进行仲裁。每个通道的仲裁数目都是可设置的，其有效范围为1~1024个数据单元。当 µDMA 控制器按照仲裁数目传输了若干个数据单元之后，随后将检查所有发出请求的通道，并向其中优先级最高的通道提供服务。

如果某个优先级较低的 µDMA 通道仲裁数目设置得太大，那么高优先级通道的传输延迟将可能增加，因为 µDMA 控制器需要等待低优先级的猝发传输完全结束之后才会重新进行仲裁，检查是否存在更高优先级的请求。基于以上原因，建议低优先级通道的仲裁数目不应设得太大，这样可以充分保障系统对高优先级µDMA通道的响应速度。

仲裁数目也可以形象地看做一个猝发的大小。仲裁数目就是获得控制权后以猝发形式连续传输的数据单元数。请注意这里所说的“仲裁”是指 µDMA 通道优先级的仲裁，而非总线的仲裁。在竞争总线时，处理器内核始终优于 µDMA 控制器。此外，只要处理器需要在同一总线上执行总线交互，µDMA 控制器都将失去总线控制权；即便在猝发传输的过程中，µDMA 控制器也将被暂时中断。

9.2.4 请求类型

µDMA 控制器可响应来自外设的两种请求：单次请求或猝发请求。每种外设可能支持其中一种或两种类型。单次请求表明外设已准备好传输一个数据单元，猝发请求表明外设已准备好传输多个数据单元。

取决于外设发出的是单次请求或猝发请求，µDMA 控制器的响应也将有所不同。假如同时产生了单次请求和猝发请求，而且 µDMA 通道已按照猝发请求建立，那么优先响应猝发请求。表9-2（515页）列出了各种外设对这两种请求类型的支持情况。

表 **9-2.** 所支持的请求类型

外设	产生单次请求的事件	产生猝发请求的事件
ADC	无	半空的 FIFO
通用定时器	无	触发事件
GPIO 触发	原始中断脉冲	无
SSI TX	TX FIFO未满	TX FIFO深度（固定为4）
SSI RX	RX FIFO非空	RX FIFO 深度（固定为 4）
UART3 TX	TX FIFO未满	TX FIFO深度（可配置）
UART RX	RX FIFO非空	RX FIFO深度（可配置）

9.2.4.1 请求类型

当检测到单次请求、并且没有猝发请求时，µDMA 控制器将传输一个数据单元，传输完成后停止并等待其它请求。

9.2.4.2 猝发请求

当检测到猝发请求后，μDMA 控制器将执行猝发传输，传输数目是以下两者的较小值：仲裁数目；尚未传输完的数据单元数。因此，仲裁数目应与外设发出猝发请求时所包含的数据单元数相同。例如，UART 模块可基于 FIFO 触发深度产生猝发请求。此时仲裁数目应与满足触发深度条件后 FIFO 能够传输的数据单元数相同。猝发传输一旦启动就必须运行到结束，期间即使有更高优先级通道的请求也无法中断。猝发传输所需的时间通常都比数量相同、单次触发的用时总和要短。

实际使用中应尽可能地采用猝发传输，尽量避免单次传输。例如，某些数据天生就只有在作为一个数据块共同传输时才有意义，每次传输一点则毫无用处。通过 DMA 通道采用猝发置位寄存器 (DMAUSEBURSTSET) 可以禁用单次请求。当把此寄存器中对应于某个通道的标志位置位后，μDMA 控制器将只响应该通道的猝发请求。

9.2.5 通道配置

μDMA 控制器采用系统内存中保存一个控制表，表中包含若干个通道控制结构体。每个 μDMA 通道在控制表中可能有一个或两个结构体。控制表中的每个结构体都包含：源指针、目的指针、待传输数目、传输模式。控制表可以定义到系统内存中的任意位置，但必须保证其连续并且按1024字节边界对齐。

表9-3（516页）列出了内存中通道控制表的内容分布布局。每个通道在控制表中都可能包含一个或两个结构体：主控制结构体及副控制结构体。在控制表中，所有主控制结构体的都在表的前半部分，所有副控制结构体都在表的下半部分。在较简单的传输模式中，对传输的连续性要求不高，允许在每次传输结束后再重新配置、重新启动。这种情况一般不需要副控制结构体，因此内存中只需放置表的前半部分，而后半部分所占用的内存可用作其它用途。如果采用更加复杂的传输模式（例如乒乓模式或散聚模式），那就需要用到副控制结构体，此时整个控制表都必须加载到内存中。

控制表中任何未用到的内存块都可留给应用程序使用，包括任何应用程序未用的通道的控制结构体，以及各个通道中未用到的控制字。

表 **9-3.** 控制结构体的存储器映射

偏移量	通道
0x0	通道 0 主功能
0x10	通道 1 主功能
...	...
0x1F0	通道 31 主功能
0x200	通道0副功能
0x210	通道1副功能
...	...
0x3F0	通道31副功能

表9-4 列出了控制表中单个控制结构体项的内容。每个控制结构体项都按照 16 字节边界对齐。每个结构体项由 4 个长整型项组成：源末指针、目的末指针、控制字以及一个未用的长整型项。末指针就是指向传输过程最末一个单元地址的指针（包含其本身）。假如源地址或目的地址并不自动递增（例如外设的寄存器），那么指针应当指向待传输的地址。

表 **9-4.** 通道控制结构体

偏移量	描述
0x000	源末指针
0x004	目的末指针
0x008	控制字

表 **9-4.** 通道控制结构体（续）

偏移量	描述
0x00C	未用

控制字包含以下的位域：

- 源/目的数据宽度
- 源/目的地址增量
- 总线重新仲裁之前传输的数目（仲裁数目）
- 待传输的数据单元总数
- 采用猝发传输标志
- 传输模式

关于控制字及其各个位域的详细介绍，请参阅“µDMA 通道控制结构体”（534页）。µDMA 控制器在传输执行期间自动更新待传输大小位域以及传输模式位域。当传输结束后，待传输数目将为0，传输模式将变为“已停止”。由于控制字是由 µDMA 控制器自动修改的，因此在每次新建传输之前必须手动配置。源末指针和目的末指针不会被自动修改，所以只要源地址或目的地址不变，就无需再进行配置。

在启动传输之前，必须将 DMA 通道启用置位 (DMAENASET) 寄存器中的相应标志位置位，启用 µDMA 通道。当需要禁用某个通道时，应将 DMA 通道使能清除寄存器 (DMAENACLR) 中的相应标志位置位。当某个 µDMA 传输结束后，控制器会自动禁用该通道。

9.2.6 传输模式

µDMA 控制器支持多种传输模式。前两种模式支持简单的单次传输。后面几种复杂的模式能够实现持续数据流。

9.2.6.1 停止模式

停止模式虽然是控制字中传输模式位域的有效值，但实际上这并不是一种真正的传输模式。当控制字中的传输模式是停止模式时，µDMA 控制器并不会对此通道进行任何传输，并且一旦该通道启用，µDMA 控制器还会自动禁用该通道。在任何 µDMA 传输结束后，µDMA 控制器都会自动将通道控制字的传输模式位域改写为停止模式。

9.2.6.2 基本模式

在基本模式下，只要有待传输的数据单元，并且收到了传输请求，µDMA 控制器便会执行传输。这种模式适用于那些只要有数据可传输就产生 µDMA 请求信号的外设。如果请求是瞬时的（即使整个传输尚未完成也并不保持），则不得采用基本模式。举例来说，如果将某个通道设为基本模式，并且采用软件启动，则启动时只会创建一个瞬时请求；此时传输的数目等于 DMA 通道控制字 (DMACHCTL) 寄存器中 ARBSIZE 位域所指定的数目，即使还有更多数据需要传输也将停止。

在基本模式下，当所有数据单元传输完成后，µDMA 控制器自动将该通道置为停止模式。

9.2.6.3 自动模式

自动模式与基本模式类似，区别在于：每当收到一个传输请求后，传输过程会一直持续到整个传输结束，即使 µDMA 请求已经消失（瞬时请求）也会持续完成。这种模式非常适用于软件触发的传输过程。一般来说外设都不使用自动模式。

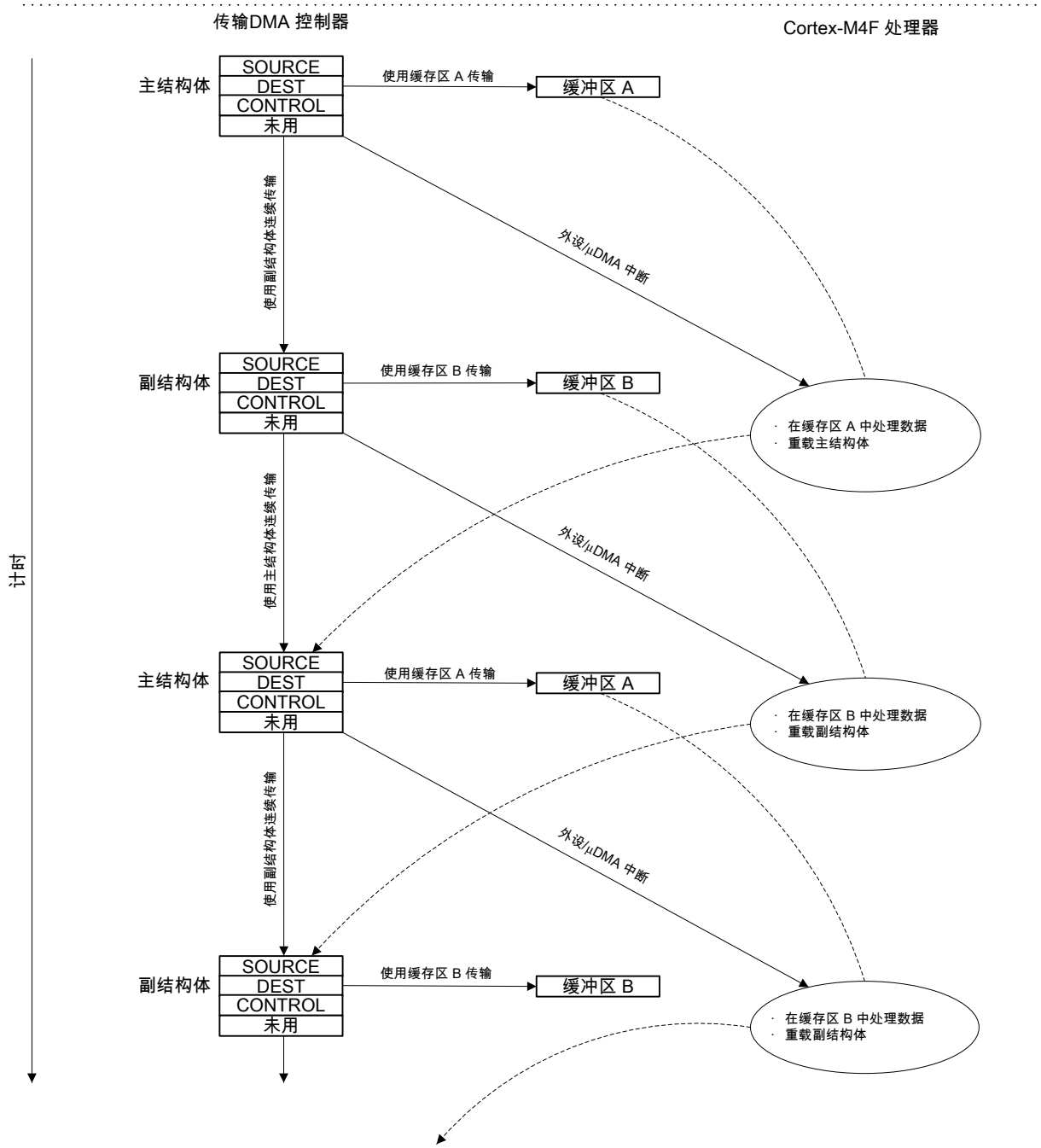
在自动模式下，当所有数据单元传输完成后，µDMA 控制器自动将该通道置为停止模式。

9.2.6.4 乒乓

乒乓模式用于实现内存与外设之间连续不断的数据流。要使用乒乓模式，必须同时配置主数据结构体和副数据结构体。两个结构体均用于实现存储器与外设之间的数据传输，均由处理器建立。传输过程首先从主控制结构体开始。当主控制结构体所配置的传输过程结束后，μDMA 控制器自动载入副控制结构体并按其配置继续传输。每当这时都会产生一个中断，处理器可以对刚刚结束传输过程的数据结构体进行重新配置。于是乎，主/副控制结构体交替在缓冲区与外设之间搬运数据，周而复始，川流不息。

图9-2（519页）描绘出了乒乓模式下的操作示例。

图 9-2. 乒乓式 µDMA 数据会话的示例



9.2.6.5 存储器散聚

存储器散聚模式是一种较为复杂的工作模式。通常在搬运数据块时，其数据源和数据目的都是线性分布的；但有时必须将内存中某块连续的数据分散传递到几个不同的位置，或将内存中几个不同位置的数据块汇聚传递到同一个位置连续放置，此时就应当采用散聚模式。举例来说，内存中可能存储有数条遵从某种通信协议的报文，那么就可以利用 µDMA 的汇聚模式将几个报文的有效数据内容依次读出、并连续保存到内存缓冲中的指定位置（有效内容拼装）。

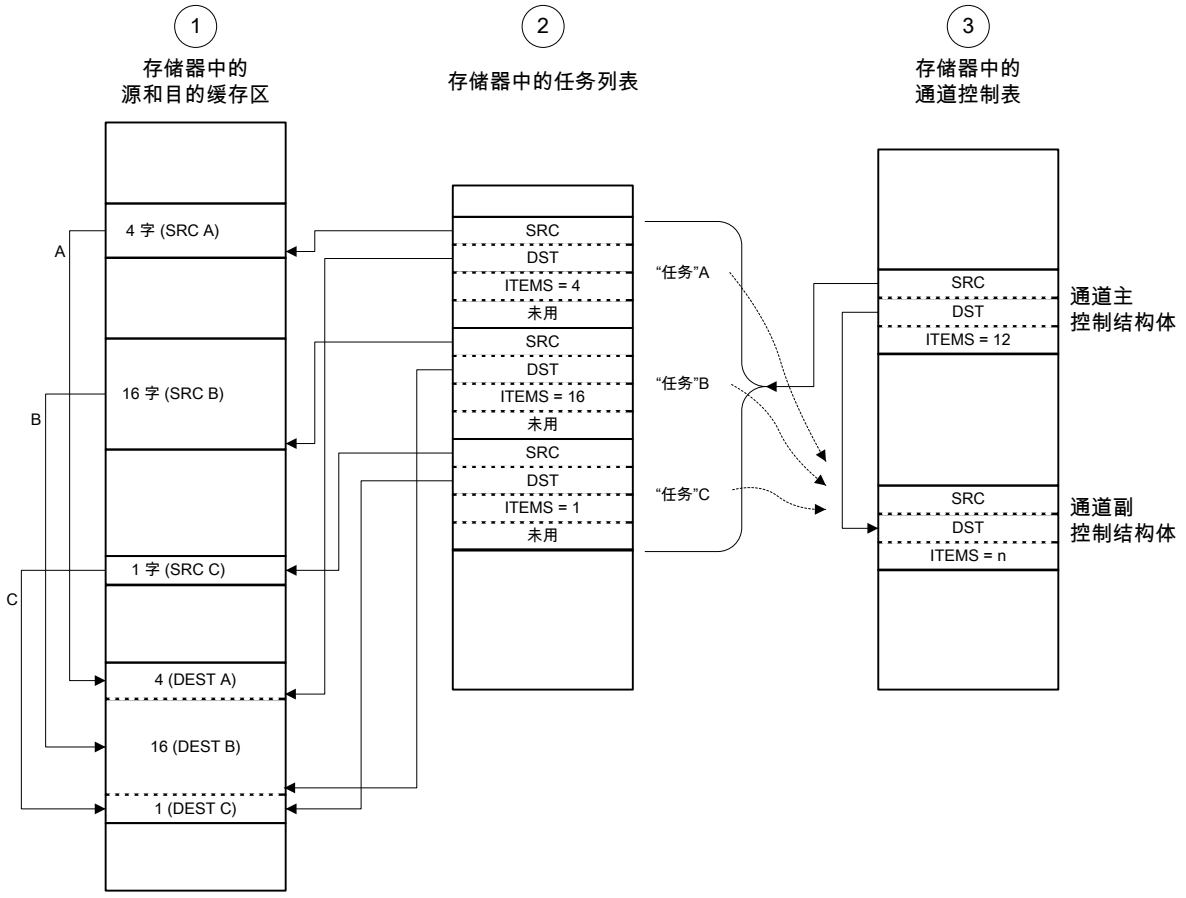
在存储器散聚模式下，主控制结构体的工作是按照内存中一个表的内容配置副控制结构体。这个表由处理器软件建立，包含若干个控制结构体，每个控制结构体中包含能够实现特定传输的源末指针、目的末指针、控制字。每个控制结构体项的控制字中必须将传输模式设置为散聚模式。主传输流程依次将表中的控制结构体项拷贝到副控制结构体中，随后予以执行。μDMA 控制器就这样交替切换：每次用主控制结构体从列表中将下一个传输流程配置拷贝到副控制结构体中，然后切换到副控制结构体执行相应的传输任务。在列表的最末一个控制结构体项中，应将其控制字编程为采用自动传输模式。这样在执行最后一个传输过程时是自动模式，μDMA 控制器在执行完成后将停止此通道的运行。只有当最后一次传输过程也结束后，才会产生结束中断。如果让控制表最后一个控制结构体项拷贝覆盖主控制结构体，使其重新指向列表的起始位置（或指向一个新的列表），就可以让整个列表始终不停循环工作。此外通过编辑控制表内容，也可以触发一个或多个其它通道执行传输：比较直接的方式是编辑产生一个写操作、以软件触发其它通道；也可以采用间接的方式，通过设法让某个外设动作而产生 μDMA 请求。

按照这种方式对 μDMA 控制器进行配置，即可基于一个 μDMA 请求执行一组最多 256 个指定的传输。

图9-3（521页）和 图9-4（522页）描绘出按照存储器散聚模式工作的示例。这个例子演示的是汇集操作：将分别位于内存中三个不同缓冲区的数据拷贝到同一个缓冲区中并连续放置。图9-3（521页）描绘出应用程序应如何在内存中建立一个 μDMA 任务列表，控制器按照该列表执行三组来自内存中不同位置的拷贝操作。通道的主控制结构体负责将控制结构体项从任务列表中拷贝出来，并填充到副控制结构体中。

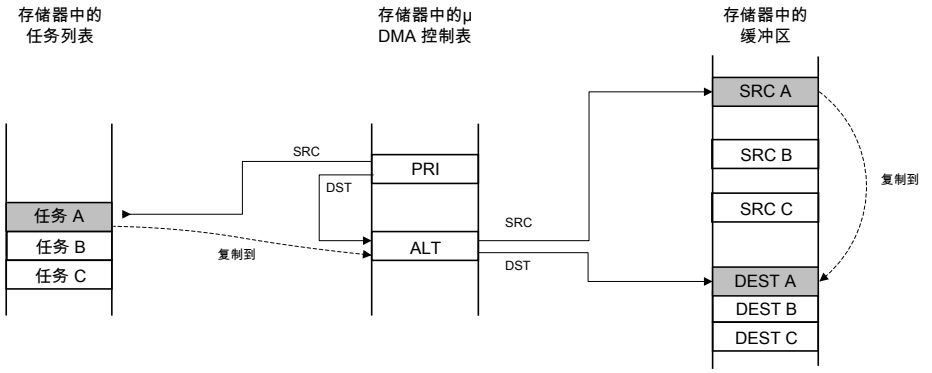
图9-4（522页）描绘出 μDMA 控制器执行三组拷贝操作的序列。首先，μDMA 控制器按照主控制结构体工作，将任务A载入到副控制结构体中。随后 μDMA 控制器切换到副控制结构体，按照任务 A 从源缓冲区 A 拷贝数据到目的缓冲区。随后，μDMA 控制器再次按照主控制结构体工作，将任务 B 载入到副控制结构体中，并按照副控制结构体执行任务 B 的拷贝操作。对于任务C也同样重复以上步骤。

图 **9-3.** 存储器散聚模式：创建及配置



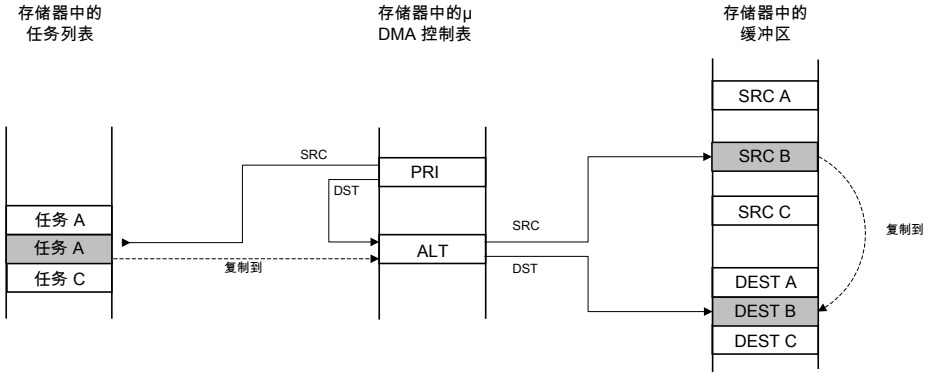
注：
1. 应用程序需要将存储器中三个不同的位置的若干个数据单元拷贝到一个缓冲区中并顺序组合.
2. 应用程序在存储器中建立 µDMA“任务列表”，表中包括 3 个 µDMA 控制“任务”的指针以及控制配置。
3. 应用程序设置通道的主控制结构体，每次将一个任务的配置拷贝 到副控制结构体中，并且接下来由 µDMA 控制器予以执行。
4.任务列表中的 SRC 和 DST 指针必须指向相应缓冲区中的最后位置。

图 9-4. 存储器散聚模式：μDMA 复制序列



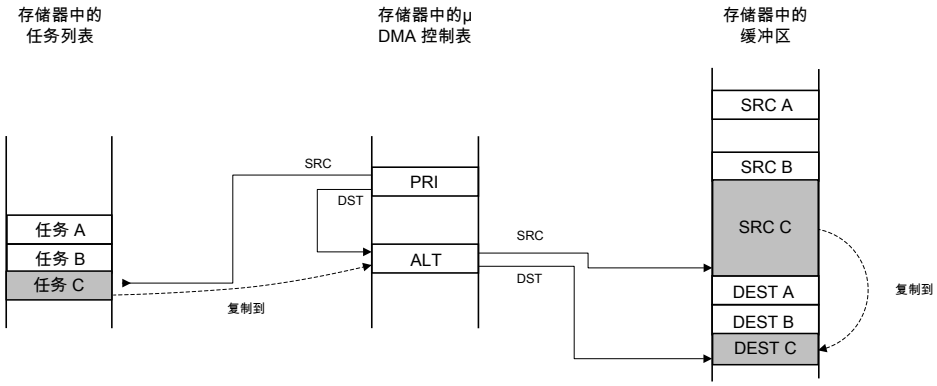
μDMA 控制器按照通道的主控制结构体工作， 将任务 A 的配置复制到 通道的 副控制结构体中。

然后，通过通道的副控制结构体， 将数据从源缓冲区 A 复制到目标 缓冲区。



μDMA 控制器按照通道的主控制结构体工作， 将任务 B 配置复制到 通道的 副控制结构体中。

然后，通过通道的副控制结构体， 将数据从源缓冲区 B 复制到目标 缓冲区。



μDMA 控制器按照通道的主控制结构体工作， 将任务 C 配置复制到 通道的 副控制结构体中。

然后，通过通道的副控制结构体， 将数据从源缓存区 C 复制到目标 缓冲区。

9.2.6.6 外设散聚

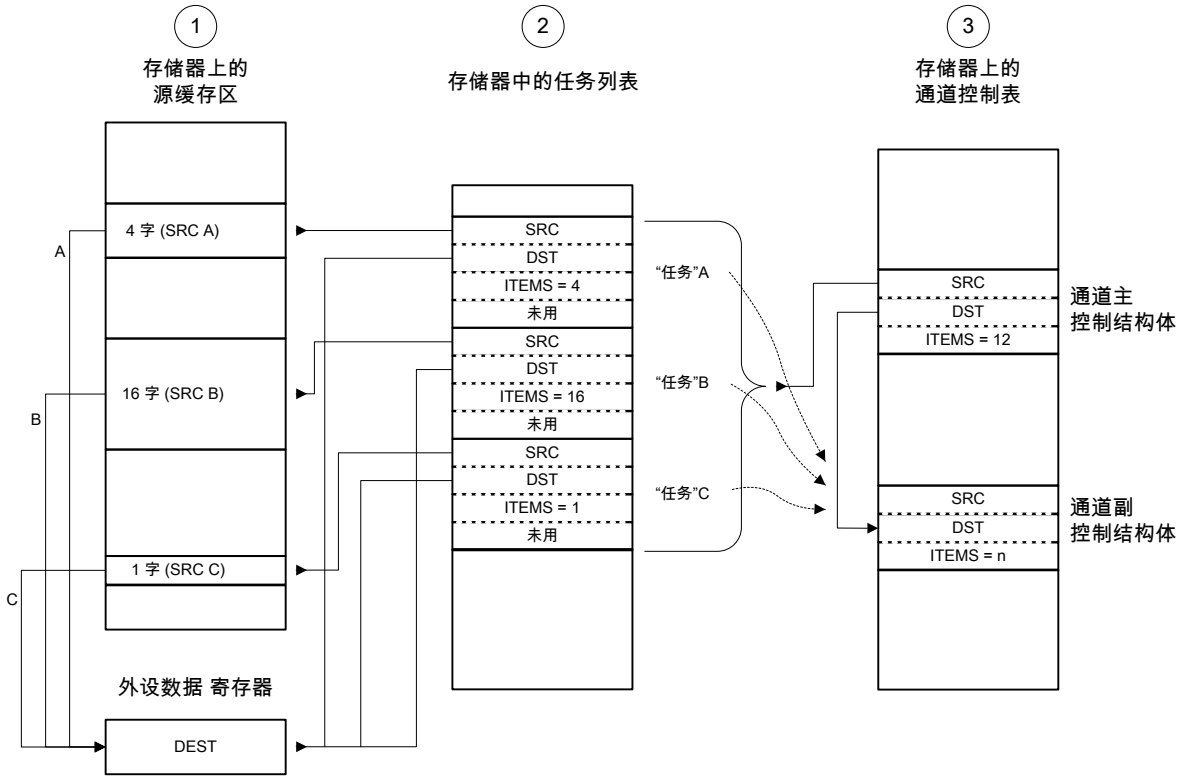
外设散聚模式与存储器散聚模式非常相似，只不过传输过程是由产生 µDMA 请求的外设控制的。当 µDMA 控制器检测到有来自外设的请求后，将通过主控制结构体从控制表中拷贝一个控制结构体项填充到副控制结构体中，随后执行其传输过程。此次传输过程结束后，只有当外设再次产生 µDMA 请求后，才会开始下一个传输过程。仅当外设产生请求时，µDMA 控制器才会继续执行控制表中的传输任务，直至完成最后一次传输。只有当最后一次传输过程也结束后，才会产生结束中断。

按照这种方式对 µDMA 控制器进行配置，只要外设准备好传输数据，就可以在内存的若干指定地址与外设之间传输数据。

图9-5（524页）和图9-6（525页）描绘出按照外设散聚模式工作的示例。这个例子演示的是汇集操作：将分别位于内存中三个不同位置的数据拷贝到同一个外设数据寄存器中。图9-5（524页）描绘出应用程序应如何在内存中建立一个 µDMA 任务列表，控制器按照该列表执行三组来自内存中不同位置的拷贝操作。通道的主控制结构体负责将控制结构体项从任务列表中拷贝出来，并填充到副控制结构体中。

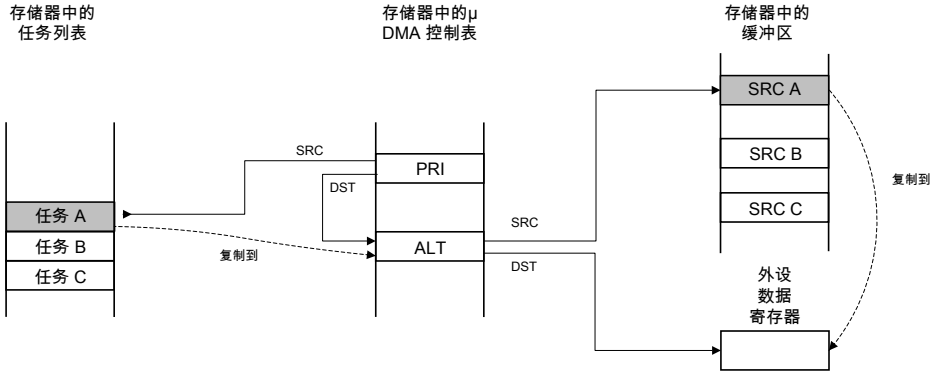
图9-6（525页）描绘出 µDMA 控制器执行三组拷贝操作的序列。首先，µDMA控制器按照主控制结构体工作，将任务A载入到副控制结构体中。随后µDMA控制器切换到副控制结构体，按照任务A从源缓冲区A拷贝数据到外设数据寄存器。随后，µDMA控制器再次按照主任务结构体工作，将任务B载入到副控制结构体中，并按照副控制结构体执行任务B的拷贝操作。对于任务C也同样重复以上步骤。

图 9-5. 外设散聚模式：创建及配置



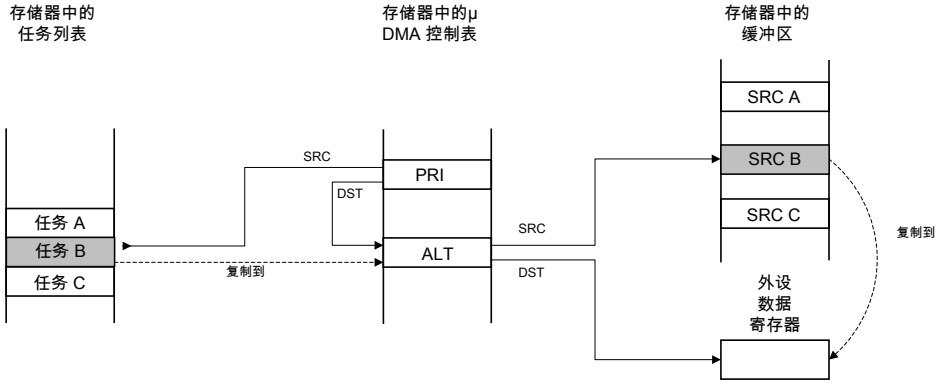
注：
1. 应用程序需要将内存中三个不同位置的若干个数据单元复制到一个外设 数据寄存器。
2. 应用程序在存储器中建立 μDMA“任务列表”，表中包括 3 个 μDMA 控制“任务”指针以及控制配置。
3. 应用程序设置通道的主控制结构体，每次将一个任务的配置复制到副控制结构体中，
并且接下来由 μDMA 控制器予以执行。

图 9-6. 外设散聚模式：μDMA 复制序列



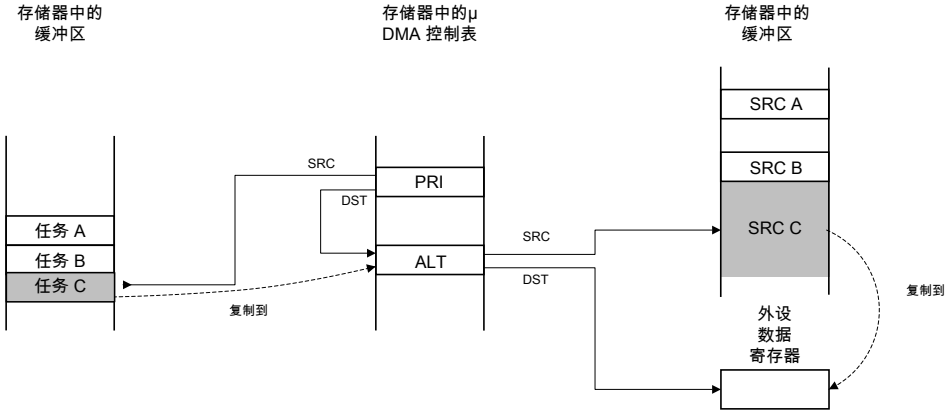
μDMA 控制器按照通道的主控制结构体工作， 将任务 A 的配置复制到 通道的 副控制结构体中。

然后，通过通道的副控制结构体， 将数据从源缓冲区 A 复制到外设 数据寄存器。



μDMA 控制器按照通道的主控制结构体工作， 将任务 B 的配置复制到 通道的 副控制结构体中。

然后，通过通道的副控制结构体， 将数据从源缓冲区 B 复制到外设 数据寄存器。



μDMA 控制器按照通道的主控制结构体工作， 将任务 C 的配置复制到 通道的 副控制结构体中。

然后，通过通道的副控制结构体， 将数据从源缓冲区 C 复制到外设 数据寄存器。

9.2.7 待传输数目及增量

μDMA 控制器支持传输宽度为 8 位、16 位或 32 位的数据。对于任何传输，都必须保障源数据宽度与目的数据宽度一致。源地址及目的地址可以按字节、半字或字自动递增，也可以设置为不自动递增。源地址增量及目的地址增量相互无关，设置地址增量时只要保证其大于等于数据宽度即可。例如，当传输8位宽的数据单元时，将地址增量设置为整字（32位）也是允许的。待传输的数据在内存中必须按照数据宽度（8位、16位或32位）对齐。

表9-5 列出了从某个支持 8 位数据的外设进行读操作时的配置。

表 **9-5. μDMA** 读操作实例：**8** 位外设

位域	配置
源数据宽度	8 位
目的数据宽度	8 位
源地址增量	不递增
目的地址增量	字节
源末指针	外设读 FIFO 寄存器
目的末指针	内存中数据缓冲区的末尾

9.2.8 外设接口

如果某个外设支持μDMA 功能，则当其作好传输数据的准备时，将可产生一个单次请求信号和/或一个猝发请求信号（见表9-2（515页））。请求信号可通过 DMA 通道请求屏蔽置位寄存器 (DMAREQMASKSET) 禁用，也可通过 DMA 通道请求屏蔽清除寄存器 (DMAREQMASKCLR) 使能。若某个通道的请求屏蔽位置位，则禁用（屏蔽）该通道的 μDMA 请求信号。假如并未屏蔽该请求信号，并且 μDMA 通道已经正确配置并且以及启用，则当外设产生请求信号时，μDMA 控制器将开始传输过程。

注意: 当使用 μDMA 与外设进行数据通信时，外设必须禁用所有到 NVIC 的中断。

当 μDMA 传输结束后，μDMA 控制器产生一个中断，详见 “中断及错误”（526页）。

关于某种外设与 μDMA 控制器如何相互配合工作的详细信息，请参阅该类型外设 DMA 操作的相关章节。

9.2.9 软件请求

在 32 个 μDMA 通道中有一个是专用于软件启动的传输过程的。当此通道 μDMA 传输结束时，还有专用的中断予以指示。要想正确使用软件启动的 μDMA 传输，应首先配置并使能传输过程，之后通过 DMA 通道软件请求寄存器 (DMASWREQ) 发送软件请求。请注意，基于软件的μDMA传输应当采用自动传输模式。

通过 DMASWREQ 寄存器也可以启动任意可用软件通道的传输。假如在某个外设的 μDMA 通道上采用软件启动请求，那么当传输结束时，结束中断将在该外设的中断向量处产生，而非软件中断向量。只要某个外设并不用到 μDMA 数据传输，任何外设通道都可以用于软件传输请求。

9.2.10 中断及错误

根据外设的情况，μDMA 可以在一个完整的传输结束时或在 FIFO 或缓存达到特定水平时显示传输完成（参见表9-2（515页）和各个外设章节）。当某个 μDMA 传输过程结束时，μDMA 控制器将在相应外设的中断向量处产生一个结束中断。因此，假如某个外设采用 μDMA 传输数据，并且启用了该外设的中断，那么中断处理函数中必须包含对 μDMA 传输结束中断的相关处理。假如传输过程使用了软件μDMA通道，那么结束中断将在专用的软件μDMA中断向量上产生（参见表9-6（527页））。

当启用某外设的 µDMA 后，µDMA 控制器将禁止该外设的普通传输中断传递到中断控制器，不过这些中断的状态仍然能在外设的中断寄存器中查询到。因此，当采用 µDMA 传输大量数据时，中断控制器并不会随着数据流从外设频繁收到中断，而是只在数据传输过程结束后收到一个中断。请注意，未屏蔽的外设错误中断仍会正常发送到中断控制器。

当某一 µDMA 通道发出完成中断，在 DMA 通道中断状态 (DMACHIS) 寄存器中对应该外设通道的 CHIS 位被置位（参见 561页）。外设中断处理代码也可以通过该寄存器来确定中断是由 µDMA 通道造成的，还是由外设中断寄存器上报的出错事件造成的。当中断处理程序激活后，µDMA 控制器所发出的完成中断请求会自动清除。

若 µDMA 控制器在尝试进行数据传输时遇到了总线错误或存储器保护错误，将会自动关闭出错的 µDMA 通道，并且在 µDMA 错误中断向量处产生中断。处理器可以通过读取 DMA 总线错误清除寄存器 (DMAERRCLR) 来确定是否有需要处理的错误。一旦产生错误则 ERRCLR 标志位将置位。向 ERRCLR 位写 1 即可清除错误状态。

表9-6 列出了 µDMA 控制器专用的中断。

表 **9-6. µDMA** 中断分配

中断	分配
46	µDMA软件通道传输中断
47	µDMA错误中断

9.3 初始化和配置

9.3.1 模块初始化

在使用 µDMA 控制器之前，必须先在系统控制模块中将其启用，并且在外设中启用 µDMA 功能。此外，还应当先设置好通道控制结构体的位置。

系统初始化期间应执行一遍下面的步骤：

1. 使用 RCGCDMA 寄存器启用 µDMA 时钟（参见 297页）。
2. 通过将 DMA 配置 (DMACFG) 中的 MASTEREN 位置位，启用 µDMA 控制器。
3. 向 DMA 通道控制基指针寄存器 (DMACTLBASE) 写入控制表的基地址,可以对通道控制表的位置编程。基地址必须按照1024字节对齐。

9.3.2 存储器到存储器传输的配置

第 30 号 µDMA 通道是专用的软件启动传输通道。不过，只要相关外设不使用µDMA功能，那么任何通道都可以用于软件启动、存储器到存储器的传输。

9.3.2.1 配置通道属性

首先我们应当配置通道属性：

1. 将 DMA 通道优先置位寄存器 (DMAPRIOSET) 的第 30 位置位，即可将通道设为高优先级；将 DMA 通道优先清除寄存器 (DMAPRIOCLR) 的第 30 位置位，即可将通道设为默认优先级。
2. 将 DMA 通道主副清除寄存器 (DMAALTCLR) 的第 30 位置位，可为此次传输选择主通道控制结构体。
3. 将 DMA 通道采用猝发清除 (DMAUSEBURSTCLR) 寄存器中的第 30 位置位，可允许 µDMA 控制器既能响应单次请求也能响应猝发请求。

4. 将 DMA 通道请求屏蔽清零 (DMAREQMASKCLR) 寄存器中的第 30 位置位，以允许 μDMA 控制器识别该通道的请求。

9.3.2.2 配置通道控制结构体

下面来配置通道控制结构体。

本示例需要实现的功能是：从某个内存缓冲区向另一缓冲区传输 256 个字。采用第30号通道进行软件启动传输，其控制结构体在控制表中的偏移量为0x1E0。通道 30 的通道控制结构体的偏移量见表9-7。

表 **9-7.** 第 **30** 号通道的通道控制结构体偏移量

偏移量	描述
控制表基地址 +0x1E0	第30号通道源末指针
控制表基地址 +0x1E4	第 30 号通道目的末指针
控制表基地址 +0x1E8	第 30 号通道控制字

配置源和目的参数

源末指针和目的末指针都应当指向传输过程最后一次传输的地址（其本身包含在内）。

1. 向偏移量0x1E0处的源末指针写入：源缓冲地址+0x3FC（0xFF * 4）；

2. 向偏移量0x1E4处的目的末指针写入：目的缓冲地址+0x3FC（0xFF * 4）；

至于偏移量 0x1E8 处的控制字，必须按照 表9-8 进行编程。

表 **9-8.** 存储器传输示例的通道控制字配置

DMACHCTL 中的位域	位	值	描述
DSTINC	31:30	2	目标地址按32位自动递增
DSTSIZE	29:28	2	目标数据宽度为32位
SRCINC	27:26	2	源地址按32位自动递增
SRCSIZE	25:24	2	源数据宽度为32位
保留	23:18	0	保留
ARBSIZE	17:14	3	传输8个数据单元后仲裁
XFERSIZE	13:4	255	总共传输256个单元
NXTUSEBURST	3	0	对本传输类型无意义
XFERMODE	2:0	2	采用自动请求传输模式

9.3.2.3 启动传输过程

完成通道配置后，即可启动传输过程：

1. 将 DMA 通道使能置位寄存器 (DMAENASET) 的第 30 位置位，即可使能通道。

2. 将 DMA 通道软件请求寄存器 (DMASWREQ) 的第 30 位置位，产生传输请求。

随后就会开始 μDMA 传输。倘若同时开启了相关的中断，那么当传输过程全部结束后还会产生中断事件通知处理器。如果需要，还需通过读取 DMAENASET 寄存器中的第 30 位来检查状态。当传输完成后，此位自动清零。此外也可通过读通道控制字（偏移量 0x1E8）的 XFERMODE 位域来检查传输状态。当传输完成后，此位自动清零。

9.3.3 外设简单发送的配置

在下面的示例中，我们要配置 µDMA 控制器，将缓冲区中的数据发送给某个外设。该外设具有发送 FIFO，且触发深度为 4。此示例中的外设占用 µDMA 第 7 号通道。

9.3.3.1 配置通道属性

首先我们应当配置通道属性：

1. 配置 DMA 通道优先置位寄存器 (DMAPRIOSET) 的第 7 位，即可将通道设为高优先级；将 DMA 通道优先清除寄存器 (DMAPRIOCLR) 的第 7 位置位，即可将通道设为默认优先级。
2. 将 DMA 通道主副清除寄存器 (DMAALTCLR) 的第 7 位置位，为此次传输选择主通道控制结构体。
3. 将 DMA 通道采用猝发清除 (DMAUSEBURSTCLR) 寄存器中的第 7 位置位，允许 µDMA 控制器既能响应单次请求也能响应猝发请求。
4. 将 DMA 通道请求屏蔽清零 (DMAREQMASKCLR) 寄存器中的第 7 位置位，以允许 µDMA 控制器识别该通道的请求。

9.3.3.2 配置通道控制结构体

本示例需要实现的功能是：从某个内存缓冲区经过第 7 号通道向某个外设的发送 FIFO 寄存器传输 64 个字节。第 7 号通道的控制结构体在控制表中的偏移量为 0x070。通道 7 的通道控制结构体的偏移量见 表9-9。

表 **9-9.** 第 **7** 号通道的通道控制结构体偏移量

偏移量	描述
控制表基地址 +0x070	第7号通道源末指针
控制表基地址 +0x074	第 7 号通道目的末指针
控制表基地址 +0x078	第 7 号通道控制字

配置源和目的参数

源末指针和目的末指针都应当指向传输过程最后一次传输的地址（其本身包含在内）。由于外设指针是固定的，因此只需指向外设的数据寄存器即可。

1. 向偏移量0x070处的源末指针写入：源缓冲地址+0x3F；
2. 向偏移量0x074处的目的末指针写入：外设的发送FIFO寄存器地址；

至于偏移量 0x078 处的控制字，应按照 表9-10 进行编程。

表 **9-10.** 外设传输示例的通道控制字配置

DMACHCTL 中的位域	位	值	描述
DSTINC	31:30	3	目标地址不自动递增
DSTSIZE	29:28	0	目标数据宽度为8位
SRCINC	27:26	0	源地址按8位自动递增
SRCSIZE	25:24	0	源数据宽度为8位
保留	23:18	0	保留
ARBSIZE	17:14	2	传输4个数据单元后仲裁

表 **9-10.** 外设传输示例的通道控制字配置（续）

DMACHCTL 中的位域	位	值	描述
XFERSIZE	13:4	63	总共传输64个单元
NXTUSEBURST	3	0	对本传输类型无意义
XFERMODE	2:0	1	采用基本传输模式

注意**:** 注：在这个示例中，外设产生的是单次请求还是猝发请求并不重要。由于外设本身具有发送 FIFO，并且在深度达到 4 时触发，因此将仲裁数目设为 4。即使外设真的产生猝发请求，那么传输 4 字节也正好符合 FIFO 的容限。假如外设产生的是单次请求（即FIFO中仍然有空位），那么将每次传输1个字节。假如应用程序要求必须按猝发方式传输，那么应当将 DMA 通道采用猝发置位寄存器 (DMAUSEBURSTSET) 中管辖通道猝发的 SET[7] 置位。

9.3.3.3 启动传输过程

完成通道配置后，即可启动传输过程：

1. 将 DMA 通道使能置位寄存器 (DMAENASET) 的第 7 位置位，即可使能通道。

随后 μDMA 控制器即可经由第 7 号通道进行传输。每当外设产生 μDMA 请求后，控制器就会向其传输若干数据。当全部64个字节传输完成后，传输过程才会结束。传输过程结束后 μDMA 控制器将自动禁用该通道，并将通道控制字的 XFERMODE 位清零（停止模式）。可以通过读取 DMA 通道启用置位 (DMAENASET) 寄存器中的第 7 位来检查传输状态。当传输完成后，此位自动清零。此外也可通过通道控制字（偏移量 0x078）的 XFERMODE 位域来检查传输状态。当传输完成后，此位自动清零。

假如使能了该外设的中断，那么当整个传输过程结束时，外设中断处理程序将收到中断信号。

9.3.4 外设乒乓接收的配置

在下面的示例中，我们要配置 μDMA 控制器，从某个外设连续接收 8 位数据，并保存到一对 64 字节的缓冲区中。该外设具有发送 FIFO，且触发深度为 8。此示例中的外设占用 μDMA 第 8 号通道。

9.3.4.1 配置通道属性

首先我们应当配置通道属性：

1. 配置 DMA 通道优先置位寄存器 (DMAPRIOSET) 的第 8 位，即可将通道设为高优先级；将 DMA 通道优先清除寄存器 (DMAPRIOCLR) 的第 7 位置位，即可将通道设为默认优先级。

2. 将 DMA 通道主副清除寄存器 (DMAALTCLR) 的第 8 位置位，为此次传输选择主通道控制结构体。

3. 将 DMA 通道采用猝发清除 (DMAUSEBURSTCLR) 寄存器中的第 8 位置位，允许 μDMA 控制器既能响应单次请求也能响应猝发请求。

4. 将 DMA 通道请求屏蔽清零 (DMAREQMASKCLR) 寄存器中的第 8 位置位，以允许 μDMA 控制器识别该通道的请求。

9.3.4.2 配置通道控制结构体

下面来配置通道控制结构体，本示例需要实现的功能是：从外设的接收FIFO向两个分别为64字节的缓冲区传输若干字节。接收数据时，当一个缓冲区装满后，μDMA 控制器自动切换到向另一个缓冲区填充收到的数据。

要想实现乒乓式缓冲，必须同时使用该通道的主控制结构体和副控制结构体。第8号通道的主控制结构体在控制表中的偏移量为0x080，副控制结构体在控制表中的偏移量为0x280。通道 8 的通道控制结构体的偏移量见 表9-11。

表 **9-11.** 第 **8** 号通道的主控制结构体及副控制结构体偏移量

偏移量	描述
控制表基地址 +0x080	第 8 号通道主源末指针
控制表基地址 +0x084	第 8 号通道主目的末指针
控制表基地址 +0x088	第 8 号通道主控制字
控制表基地址 +0x280	第 8 号通道副源末指针
控制表基地址 +0x284	第 8 号通道副目的末指针
控制表基地址 +0x288	第 8 号通道副控制字

配置源和目的参数

源末指针和目的末指针都应当指向传输过程最后一次传输的地址（其本身包含在内）。由于外设指针是固定的，因此只需指向外设的数据寄存器即可。主控制结构体和副控制结构体中的指针都必须进行配置。

1. 向偏移量0x080处的主源末指针写入：外设的接收缓冲地址；
2. 向偏移量 0x084 处的主目的末指针写入：乒乓缓冲区A地址 +0x3F。
3. 向偏移量0x280处的副源末指针写入：外设的接收缓冲地址；
4. 向偏移量0x284处的副目的末指针写入：乒乓缓冲区B地址+0x3F；

至于偏移量0x088处的主控制字和0x288处的副控制字，应按照下面的方式编程：

1. 按照对偏移量 0x088 处的主控制字进行编程。
2. 按照表9-12 对偏移量 0x288 处的副通道控制字进行编程。

表 **9-12.** 外设乒乓接收示例的通道控制字配置

DMACHCTL 中的位域	位	值	描述
DSTINC	31:30	0	目标地址按8位自动递增
DSTSIZE	29:28	0	目标数据宽度为8位
SRCINC	27:26	3	源地址不自动递增
SRCSIZE	25:24	0	源数据宽度为8位
保留	23:18	0	保留
ARBSIZE	17:14	3	传输8个数据单元后仲裁
XFERSIZE	13:4	63	总共传输64个单元
NXTUSEBURST	3	0	对本传输类型无意义
XFERMODE	2:0	3	采用乒乓传输模式

注意: 注：在这个示例中，外设产生的是单次请求还是猝发请求并不重要。由于外设本身具有发送 FIFO，并且在深度达到 8 时触发，因此将仲裁数目设为 8。即使外设真的产生猝发请求，那么传输 8 字节也正好符合 FIFO 的容限。假如外设产生的是单次请求（即FIFO中仍然有空位），那么将每次传输1个字节。假如应用程序要求必须按猝发方式传输，那么应当将 DMA 通道采用猝发置位寄存器 (DMAUSEBURSTSET) 中管辖通道猝发的 SET[8] 置位。

9.3.4.3 配置外设中断

当采用 μDMA 的乒乓模式工作时，应当配置中断服务函数。强烈建议通过中断服务函数进行相关处理。不过，乒乓模式也可以采用轮询方式进行相关处理。每当其中一个缓冲区传输完成后即会触发中断。

1. 配置并使能该外设的中断处理函数。

9.3.4.4 启用 μDMA 通道

完成通道配置后，即可启动传输过程：

1. 将 DMA 通道使能置位寄存器 (DMAENASET) 的第 8 位置位，即可使能通道。

9.3.4.5 处理中断

当前已配置并启用了 μDMA 控制器，在第 8 号通道上可进行传输。当外设产生 μDMA 请求后，控制器将按照主控制结构体的配置将数据传输到缓冲区 A。当对缓冲区 A 的主传输流程结束后，控制器将自动切换到副通道控制结构体，并开始将数据搬运到缓冲区 B。与此同时，主通道控制字的模式位域将自动变为“已停止”，中断将挂起。

当产生中断后，中断处理函数首先应确认哪一缓冲区已传输完成；之后自行处理数据或置标志（有中断外的相应代码根据此标志处理缓冲区的数据）。随后设置本缓冲区下一次的传输任务。

依上所述，在中断处理函数中应当：

1. 读取偏移量 0x088 处的主通道控制字，检查其 XFERMODE 位域。若该位域为 0，则表明缓冲区 A 已传输结束。如果缓冲区 A 传输完成，则应当：
   a. 自行处理缓冲区A中刚收到的数据；或置标志表明缓冲区A有已接收完成的数据，由专用的缓冲区处理代码进行处理。
   b. 按照 表9-12（531页） 对偏移量 0x88 处的主控制字进行编程。
2. 读取偏移量 0x288 处的副通道控制字，检查其 XFERMODE 位域。若该位域为 0，则表明缓冲区 B 已传输结束。如果缓冲区 B 传输完成，则应当：
   a. 自行处理缓冲区B中刚收到的数据；或置标志表明缓冲区B有已接收完成的数据，由专用的缓冲区处理代码进行处理；
   b. 按照表9-12（531页） 对偏移量 0x288 处的副通道控制字重新编程。

9.3.5 通道分配的配置

通过 DMACHMAPn 寄存器可更改任一 μDMA 通道的功能分配。本寄存器的每个 4 位域分别对应一个 μDMA 通道。

关于通道的分配参见 表9-1（514页）。

9.4 寄存器映射

表 9-13（533页） 列出了所有 μDMA 通道控制结构体以及相关寄存器。通道控制结构体展示出通道控制表中每一项的详细内容。通道控制表位于系统内存中，其位置由应用程序决定，因此其基地址为 n/a（无预定义值），且寄存器描述带有如此提示。在下表中，通道控制结构体的“偏移量”一列代表该配置字相对于控制表中每个结构体项起始地址的偏移。至于通道控制表在内存中的具体排布，请参阅 “通道配置”（516页） 以及表9-3（516页）。μDMA 寄存器地址是相对于 0x400F.F000的 μDMA 基地址而言的，以十六进制增量的方式给出。在编制 μDMA 模块寄存器之前，注意应先启用 μDMA

模块时钟，参见 297页。µDMA 模块时钟启用后，必须等待至少 3 个系统时钟才可访问 µDMA 模块寄存器。

表 9-13. µDMA 寄存器映射

偏移量	名称	类型	复位	描述	见页面
µDMA 通道控制结构体（从通道控制表基地址的偏移量）					
0x000	DMASRCENDP	R/W	-	DMA 通道源地址末指针寄存器	535
0x004	DMADSTENDP	R/W	-	DMA 通道目的地址末指针寄存器	536
0x008	DMACHCTL	R/W	-	DMA 通道控制字寄存器	537
µDMA 寄存器（从 **µDMA** 基地址的偏移量）					
0x000	DMASTAT	RO	0x001F.0000	DMA 状态寄存器	542
0x004	DMACFG	WO	-	DMA 配置寄存器	544
0x008	DMACTLBASE	R/W	0x0000.0000	DMA 通道控制基指针寄存器	545
0x00C	DMAALTBASE	RO	0x0000.0200	DMA 副通道控制基指针寄存器	546
0x010	DMAWAITSTAT	RO	0x03C3.CF00	DMA 通道等待请求状态寄存器	547
0x014	DMASWREQ	WO	-	DMA 通道软件请求寄存器	548
0x018	DMAUSEBURSTSET	R/W	0x0000.0000	DMA 通道采用猝发置位寄存器	549
0x01C	DMAUSEBURSTCLR	WO	-	DMA 通道采用猝发清除寄存器	550
0x020	DMAREQMASKSET	R/W	0x0000.0000	DMA 通道请求屏蔽置位寄存器	551
0x024	DMAREQMASKCLR	WO	-	DMA 通道请求屏蔽清零寄存器	552
0x028	DMAENASET	R/W	0x0000.0000	DMA 通道启用置位寄存器	553
0x02C	DMAENACLR	WO	-	DMA 通道启用清除寄存器	554
0x030	DMAALTSET	R/W	0x0000.0000	DMA 通道主副置位寄存器	555
0x030	DMAALTCLR	WO	-	DMA 通道主副清除寄存器	556
0x038	DMAPRIOSET	R/W	0x0000.0000	DMA 通道优先置位寄存器	557
0x03C	DMAPRIOCLR	WO	-	DMA 通道优先清零寄存器	558
0x04C	DMAERRCLR	R/W	0x0000.0000	DMA 总线错误清除寄存器	559
0x500	DMACHASGN	R/W	0x0000.0000	DMA 通道分配寄存器	560
0x504	DMACHIS	R/W1C	0x0000.0000	DMA 通道中断状态寄存器	561
0x510	DMACHMAP0	R/W	0x0000.0000	DMA 通道映射选择寄存器 0	562
0x514	DMACHMAP1	R/W	0x0000.0000	DMA 通道映射选择寄存器 1	563
0x518	DMACHMAP2	R/W	0x0000.0000	DMA 通道映射选择寄存器 2	564
0x51C	DMACHMAP3	R/W	0x0000.0000	DMA 通道映射选择寄存器 3	565
0xFD0	DMAPeriphID4	RO	0x0000.0004	DMA 外设标识寄存器 4	570
0xFE0	DMAPeriphID0	RO	0x0000.0030	DMA 外设标识寄存器 0	566
0xFE4	DMAPeriphID1	RO	0x0000.00B2	DMA 外设标识寄存器 1	567

表 9-13. μDMA 寄存器映射（续）

偏移量	名称	类型	复位	描述	见页面
0xFE8	DMAPeriphID2	RO	0x0000.000B	DMA 外设标识寄存器 2	568
0xFEC	DMAPeriphID3	RO	0x0000.0000	DMA 外设标识寄存器 3	569
0xFF0	DMAPCellID0	RO	0x0000.000D	DMA PrimeCell 标识寄存器 0	571
0xFF4	DMAPCellID1	RO	0x0000.00F0	DMA PrimeCell 标识寄存器 1	572
0xFF8	DMAPCellID2	RO	0x0000.0005	DMA PrimeCell 标识寄存器 2	573
0xFFC	DMAPCellID3	RO	0x0000.00B1	DMA PrimeCell 标识寄存器 3	574

9.5 μDMA 通道控制结构体

μDMA 通道控制结构体保存每个 μDMA 通道的传输设置。每个μDMA通道具有两个控制结构体，所有控制结构体共同在系统内存中组成一个控制表。“通道配置”（516页）给出了通道控制表以及通道控制结构体的详细解释。

通道控制表由若干个控制结构体项组成。每个通道都有一个主控制结构体和一个副控制结构体。主控制结构体位于偏移量 0x0、0x10、0x20，依此类推。副控制结构体位于偏移量 0x200、0x210、0x220，依此类推。

寄存器 1: DMA 通道源地址末指针寄存器（DMASRCENDP），偏移量 0x000

DMA 通道源地址末指针 (DMASRCENDP) 是通道控制结构体的一部分，用于定义 µDMA 传输的源地址。

µDMA 控制器可以将数据转移到片上 SRAM，也可以从片上 SRAM 将数据转移出。但是，由于 Flash 存储器和 ROM 位于不同的内部总线，所以 µDMA 控制器不能向/从 Flash 存储器或 ROM 转移数据。

注意: 此处的偏移量是指针对系统内存中控制结构体的基地址，而不是 µDMA 模块的基地址。

DMA 通道源地址末指针寄存器 (DMASRCENDP)

基址 n/a
偏移量 0x000
类型 R/W, 复位 -

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ADDR															
类型	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ADDR															
类型	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

位/域	名称	类型	复位	描述
31:0	ADDR	R/W	-	源地址末指针 此位域的内容是 µDMA 传输源数据块中最后一个数据单元的地址（其本身包含在内）。假如源地址不递增（DMACHCTL 寄存器的 SRCINC 位域为 0x3），那么此位域将指向源地址本身（例如外设的数据寄存器）。

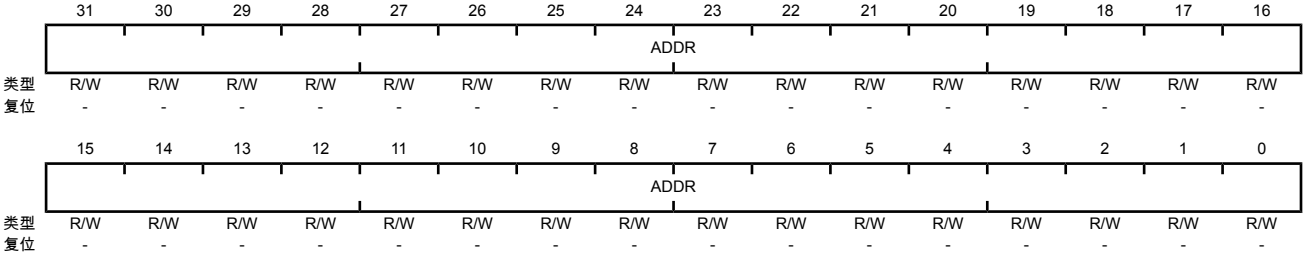
寄存器 2: DMA 通道目的地址末指针寄存器（DMADSTENDP），偏移量 0x004

DMA 通道目的地址末指针 (DMADSTENDP) 是通道控制结构体的一部分，用于定义 μDMA 传输的目的地地址。

注意: 此处的偏移量是指针对系统内存中控制结构体的基地址，而不是 μDMA 模块的基地址。

DMA 通道目的地址末指针寄存器 (DMADSTENDP)

基址 n/a
偏移量 0x004
类型 R/W, 复位 -



位/域	名称	类型	复位	描述
31:0	ADDR	R/W	-	目的地址末指针 此位域的内容是 μDMA 传输目的数据块中最后一个数据单元的地址（其本身包含在内）。假如目的地址不递增（DMACHCTL 寄存器的 DSTINC 位域为 0x3），那么此位域将指向目的地址本身（例如外设的数据寄存器）。

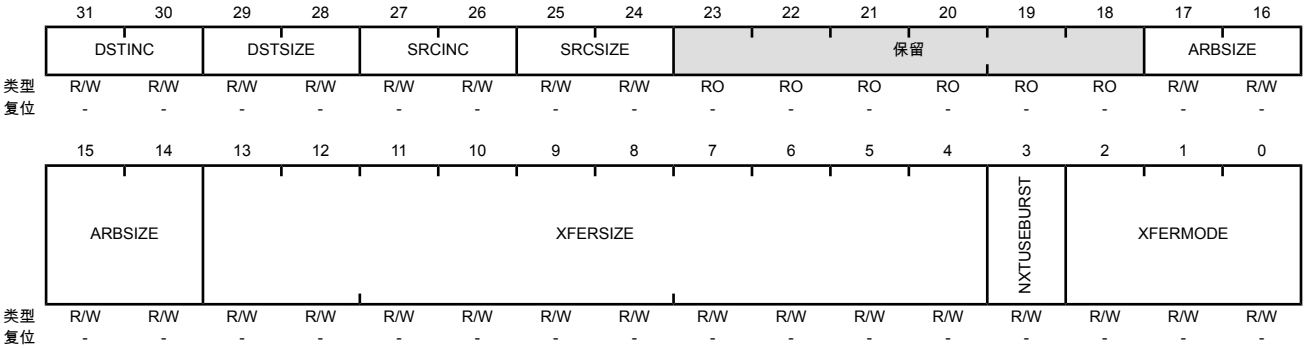
寄存器 3: DMA 通道控制字寄存器（DMACHCTL），偏移量 0x008

DMA 通道控制字 (DMACHCTL) 是通道控制结构体的一部分，用于指定 µDMA 传输的参数。

注意: 此处的偏移量是指针对系统内存中控制结构体的基地址，而不是 µDMA 模块的基地址。

DMA 通道控制字寄存器 (DMACHCTL)

基址 n/a
偏移量 0x008
类型 R/W, 复位 -



位/域	名称	类型	复位	描述
31:30	DSTINC	R/W	-	目的地址增量 此位域用于配置目的地址自动递增时的增量。 地址增量必须大于等于目的数据宽度 (DSTSIZE)。 值 描述 0x0 字节 按照8位地址递增。 0x1 半字 按照16位地址递增。 0x2 字 按照32位地址递增。 0x3 不递增 通道地址始终等于该通道目的地址末指针 (DMADSTENDP) 的值
29:28	DSTSIZE	R/W	-	目的数据宽度 此位域用于配置目的数据宽度。 注意: DSTSIZE 必须与 SRCSIZE 相等。 值 描述 0x0 字节 传输的每个数据单元为8位。 0x1 半字 传输的每个数据单元为16位。 0x2 字 传输的每个数据单元为32位。 0x3 保留

位/域	名称	类型	复位	描述
27:26	SRCINC	R/W	-	源地址增量 此位域用于配置源地址自动递增时的增量。 地址增量必须大于等于源数据宽度 (SRCSIZE)。 值 描述 0x0 字节 按照8位地址递增。 0x1 半字 按照16位地址递增。 0x2 字 按照32位地址递增。 0x3 不递增 通道地址始终等于该通道源地址末指针 (DMASRCENDP) 的值
25:24	SRCSIZE	R/W	-	源数据宽度 此位域用于配置源数据宽度。 注意: DSTSIZE 必须与 SRCSIZE 相等。 值 描述 0x0 字节 传输的每个数据单元为 8 位。 0x1 半字 传输的每个数据单元为 16 位。 0x2 字 传输的每个数据单元为 32 位。 0x3 保留
23:18	保留	RO	-	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。

位/域	名称	类型	复位	描述
17:14	ARBSIZE	R/W	-	仲裁数目 此位域用于配置传输了多少个数据单元后，µDMA 控制器重新进行仲裁。仲裁数目是2的整数幂，其可能的配置如下表所示： 值 描述 0x0 1个单元 µDMA 控制器每传输 1 个数据单元后即重新仲裁。 0x1 2个单元 0x2 4个单元 0x3 8个单元 0x4 16个单元 0x5 32个单元 0x6 64个单元 0x7 128个单元 0x8 256个单元 0x9 512个单元 0xA-0xF 1024个单元 当配置为此数值时，由于待传输数目最多也只能是1024个单元，因此 µDMA 传输过程中将不再仲裁。
13:4	XFERSIZE	R/W	-	待传输数目（减1） 此位域配置要传输的数据单元总数。配置值等于实际要传输的数据单元数减 1，也就是说若此位域的值为 0，则实际需要传输 1 个数据单元。此位域共有10位，最大可能值为1023，因此最多允许传输1024个单元。 待传输数目的单位是数据单元，而不是字节。若数据宽度为32位，则可传输若干个32位宽的字。 µDMA 控制器每次进入仲裁流程之前都会立即更新此位域，因此该位域的实际含义是剩余需要传输的数据单元数。
3	NXTUSEBURST	R/W	-	下一个采用猝发 此位域控制是否在外设散聚操作的最后一次传输任务时自动将管辖猝发的 SET[n] 位置位。通常在最后一次传输时，假如剩余待传输的数据单元数少于仲裁数目，那么 µDMA 控制器会采用单次传输来完成本次数据会话。若此标志位置位，那么控制器将采用猝发传输来完成最后一次传输。

位/域	名称	类型	复位	描述
2:0	XFERMODE	R/W	-	μDMA 传输模式 此位域控制 μDMA 流程的工作模式。关于传输模式，详见 “传输模式”（517页）。 由于此寄存器也位于系统内存中，因此其复位值不定。使能此通道前必须将此位域初始化为0。 值 描述 0x0 停止 0x1 基本 0x2 自动请求 0x3 乒乓 0x4 存储器散聚 0x5 副存储器散聚 0x6 外设散聚 0x7 副外设散聚

XFERMODE 位域的有效值.

停止
通道已停止或配置数据无效。此时不会产生传输。

基本
每次触发后（不论是由外设请求还是软件请求触发），μDMA 控制器都按照 ARBSIZE 位域指定的数目传输若干次。

自动请求
初次请求（不论是外设还是软件产生的请求）即足以完成 XFERSIZE 位域指定内容项的整个传输，无需额外再产生请求信号。

乒乓
这种模式需要同时用到主控制结构体和副控制结构体。当前（主或副）控制结构体的 XFERSIZE 位域指定的数目已经传输完成后，μDMA 控制器将切换到另一个控制结构体工作。只要控制结构体都还是乒乓模式，μDMA控制器就会这样不停切换交替传输下去。当其中一个控制结构体设为非乒乓模式后，乒乓传输流程才会停止。单个控制结构体配置的传输完成后将会产生中断。请参阅“乒乓”（518页）。

存储器散聚
这种模式需要同时用到主控制结构体和副控制结构体，按照一个任务列表依次执行其各个动作。此时，主控制结构体的源地址指针应指向一个任务列表的顶部，主控制结构体负责将任务配置拷贝到副控制结构体中。副控制结构体的 XFERMODE 位域必须配置为 0x5（副存储器散聚）才能正确执行所载入的任务。当任务执行结束后，μDMA切换回主控制结构体，并将任务列表中的下一项任务配置拷贝到副控制结构体中。流程就这样不断进行直至整个任务列表全部完成。最后一项任务的 XFERMODE 位域不得为 0x5。请注意，要想实现永不停歇的循环工作，可将最后一项任务制定为更新主控制结构体，使其重新指向任务列表的顶部，当然也可以指向其它的任务列表。请参阅“存储器散聚”（519页）。

副存储器散聚
当 μDMA 控制器工作于存储器散聚模式时，必须将副控制结构体的工作模式配置为此种模式。

外设散聚

当 µDMA 控制器工作于外设散聚模式时，必须将主控制结构体的工作模式配置为此种模式。此模式下 µDMA 控制器的工作流程与存储器散聚模式基本相同，区别在于：µDMA 控制器每次传输并非按照副控制结构体 XFERSIZE 位域定义的数目执行，而是在每次触发按照 ARBSIZE 位域定义的数目执行，详见基本模式。请参阅“外设散聚”（523页）。

副外设散聚

当 µDMA 控制器工作于外设散聚模式时，必须将副控制结构体的工作模式配置为此种模式。

9.6 µDMA 寄存器描述

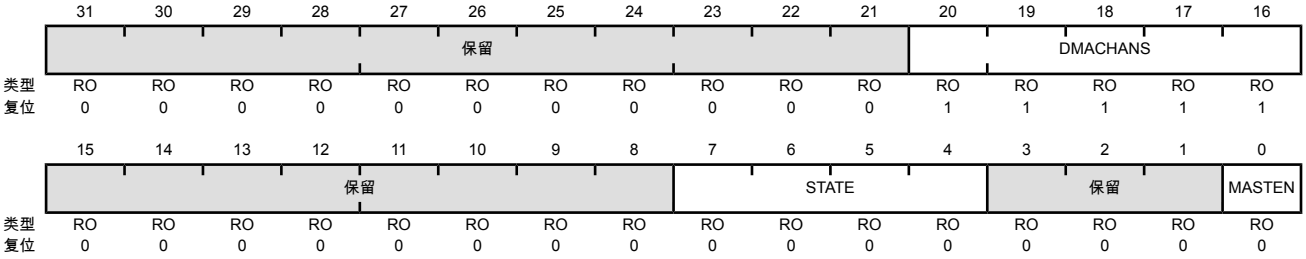
给出的寄存器地址都是相对于 0x400F.F000 的 µDMA 模块基地址而言的。

寄存器 4: DMA 状态寄存器（DMASTAT），偏移量 0x000

DMA 状态 (DMASTAT) 寄存器可返回 μDMA 控制器的当前状态。若 μDMA 控制器处于复位状态，则不能读取本寄存器。

DMA 状态寄存器 (DMASTAT)

基址 0x400F.F000
偏移量 0x000
类型 RO, 复位 0x001F.0000



位/域	名称	类型	复位	描述
31:21	保留	RO	0x000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
20:16	DMACHANS	RO	0x1F	可用的 μDMA 通道数（减 1） 此位域的值等于 μDMA 控制器可用的 μDMA 通道数减 1。复位值 0x1F 表示本器件有 32 个可用的 μDMA 通道。
15:8	保留	RO	0x00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
7:4	STATE	RO	0x0	控制状态机状态 此位域能够体现当前状态机的状态。可能的状态包括： 值 描述 0x0 空闲 0x1 正在读取通道控制数据 0x2 正在读取源末指针 0x3 正在读取目的末指针 0x4 正在从源地址读取数据 0x5 正在向目的地址写数据 0x6 正在等待μDMA请求清除 0x7 正在写入通道控制数据 0x8 已挂起 0x9 已完成 0xA-0xF 未定义
3:1	保留	RO	0x0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。

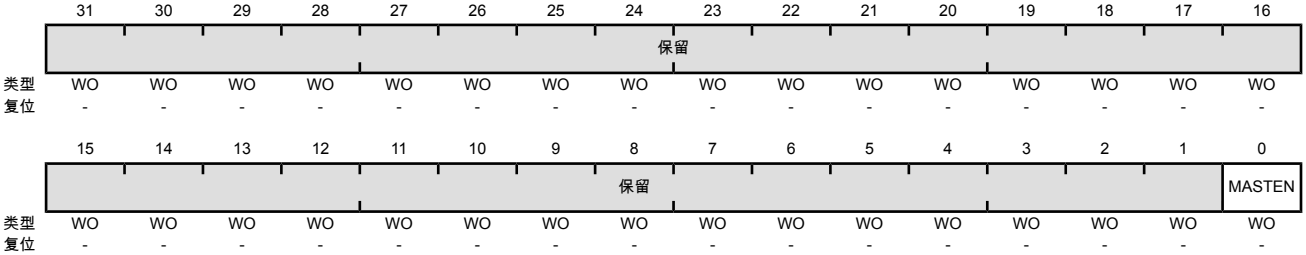
位/域	名称	类型	复位	描述
0	MASTEN	RO	0	主使能状态 值 描述 0 µDMA 控制器已被禁用。 1 µDMA 控制器已被启用。

寄存器 5: DMA 配置寄存器（DMACFG），偏移量 0x004

DMACFG 寄存器用于配置 μDMA 控制器。

DMA 配置寄存器 (DMACFG)

基址 0x400F.F000
偏移量 0x004
类型 WO, 复位 -



位/域	名称	类型	复位	描述
31:1	保留	WO	-	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
0	MASTEN	WO	-	控制器主机启用 值 描述 0 禁用 μDMA 控制器。 1 启用 μDMA 控制器。

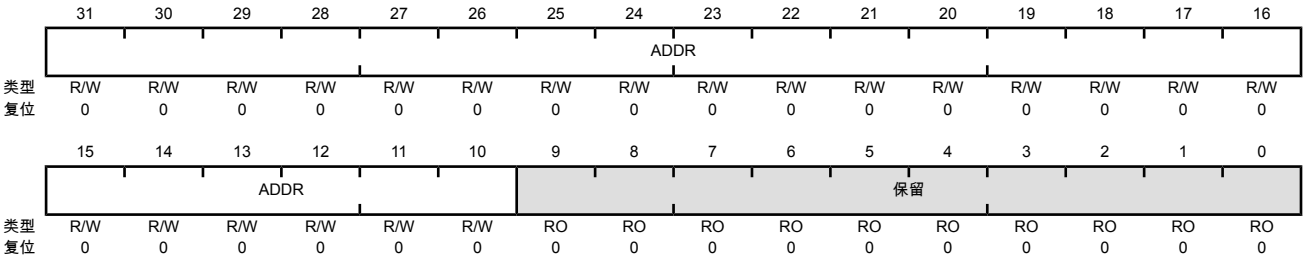
寄存器 6: DMA 通道控制基指针寄存器（DMACTLBASE），偏移量 0x008

DMA 通道控制基指针寄存器 (DMACTLBASE) 必须合理配置，使控制表的基指针指向系统内存中的某个地址。

µDMA 控制器所需的内存大小并不固定，取决于应用程序需要用到的 µDMA 通道数量，以及是否需要用到副控制数据结构体。关于通道控制表的详细信息，请参见“通道配置”（516页）。基地址必须按照1024字节对齐。当 µDMA 处于复位状态时，此寄存器不可读。

DMA 通道控制基指针寄存器 (DMACTLBASE)

基址 0x400F.F000
偏移量 0x008
类型 R/W, 复位 0x0000.0000



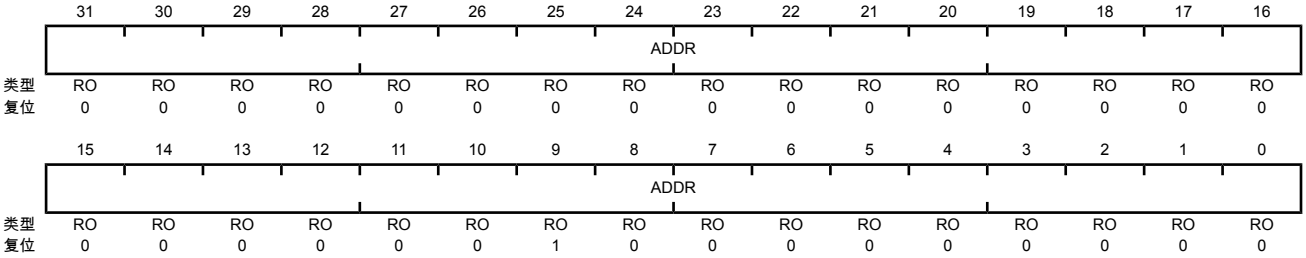
位/域	名称	类型	复位	描述
31:10	ADDR	R/W	0x0000.00	通道控制基地址 此位域包含指向通道控制表的基地址指针。基地址必须按照1024字节对齐。
9:0	保留	RO	0x00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。

寄存器 7: DMA 副通道控制基指针寄存器（DMAALTBASE），偏移量 0x00C

DMA 副通道控制基指针寄存器 (DMAALTBASE) 可返回副通道控制数据的基地址。提供本寄存器是为了方便软件使用，无需自行计算副控制结构体的地址。当 μDMA 处于复位状态时，此寄存器不可读。

DMA 副通道控制基指针寄存器 (DMAALTBASE)

基址 0x400F.F000
偏移量 0x00C
类型 RO, 复位 0x0000.0200



位/域	名称	类型	复位	描述
31:0	ADDR	RO	0x0000.0200	副通道地址指针 此位域包含副控制结构体的基地址。

寄存器 8: DMA 通道等待请求状态寄存器（DMAWAITSTAT），偏移量 0x010

本只读寄存器用于指示出 µDMA 控制器是否正在等待请求。为了提高 µDMA 的性能，可以禁止在外设发出单次请求时触发 µDMA 传输，而是只在外设产生猝发请求时才通过 µDMA 传输。此功能需外设的设计予以支持才能正常使用，软件无论采取什么方式都无法控制。当 µDMA 处于复位状态时，此寄存器不可读。

DMA 通道等待请求状态寄存器 (DMAWAITSTAT)

基址 0x400F.F000
偏移量 0x010
类型 RO, 复位 0x03C3.CF00

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	WAITREQ[n]															
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	1	1	1	1	0	0	0	0	1	1

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	WAITREQ[n]															
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	1	1	0	0	1	1	1	1	0	0	0	0	0	0	0	0

位/域	名称	类型	复位	描述
31:0	WAITREQ[n]	RO	0x03C3.CF00	通道[n]的等待状态 此位域可返回各个通道等待请求的状态。第 0 位对应于第 0 号通道。 值 描述 1 相应通道并未等待请求。 0 相应通道正在等待请求。

寄存器 9: DMA 通道软件请求寄存器（DMASWREQ），偏移量 0x014

DMASWREQ 寄存器的每个位分别对应一个 μDMA 通道。将某个位置位，即会向相应的 μDMA 通道产生一个请求信号。

DMA 通道软件请求寄存器 (DMASWREQ)

基址 0x400F.F000
偏移量 0x014
类型 WO, 复位 -

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	SWREQ[n]															
类型	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO
复位	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SWREQ[n]															
类型	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO
复位	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

位/域	名称	类型	复位	描述
31:0	SWREQ[n]	WO	-	通道[n]的软件请求 此位域用于产生软件请求。第 0 位对应于第 0 号通道。 值 描述 1 向相应通道产生软件请求。 0 不产生软件请求。 当软件请求完成后，相应标志位将自动清零。

寄存器 10: DMA 通道采用猝发置位寄存器（DMAUSEBURSTSET），偏移量 0x018

DMAUSEBURSTSET 寄存器的每个位分别对应一个 µDMA 通道。将某个位置位，即可禁止相应的 µDMA通道响应单次请求，只接受猝发请求。读取本寄存器可返回各个通道采用猝发的状态。

假如待传输数据的数目是仲裁数目（猝发大小）的整数倍，那么当完成最后一次传输后，相应的 SET[n] 位将会清零。假如剩余待传输的数据单元少于仲裁数目（猝发大小），那么 µDMA 控制器会自动将相应的 SET[n] 位清零，并且剩余的数据单元将按照单次请求的方式传输。如果想用猝发方式传输剩余的数据单元，应将相应位再次置位。假如外设不支持猝发请求模型，则此寄存器中的相关位不得置位。

详细信息和实例请参考 “请求类型”（515页）。

DMA 通道采用猝发置位寄存器 (DMAUSEBURSTSET)

基址 0x400F.F000
偏移量 0x018
类型 R/W, 复位 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	SET[n]															
类型	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SET[n]															
类型	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位/域	名称	类型	复位	描述
31:0	SET[n]	R/W	0x0000.0000	通道[n]的采用猝发置位

值	描述
0	µDMA 通道 [n] 既可响应单次请求也响应猝发请求。
1	µDMA 通道 [n] 只响应猝发请求。

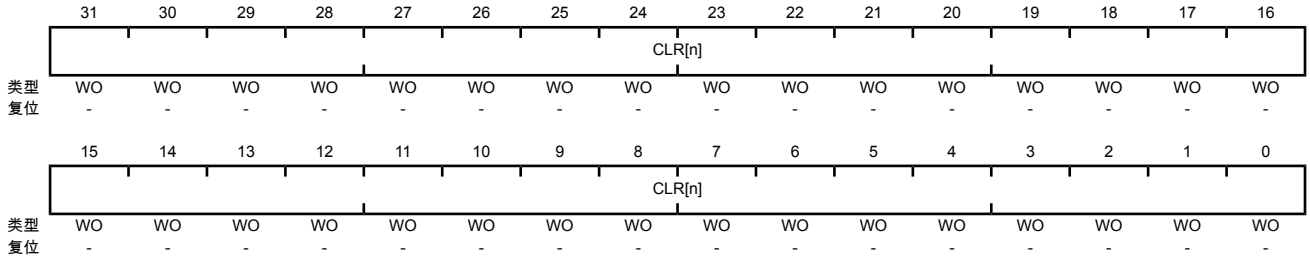
第 0 位对应于第 0 号通道。该位如上所述自动清零。此外，也可以通过将 DMAUSEBURSTCLR 寄存器中相应的 CLR[n] 位置位来手动清零位。

寄存器 11: DMA 通道采用猝发清除寄存器（DMAUSEBURSTCLR），偏移量 0x01C

DMAUSEBURSTCLR 寄存器的每个位分别对应一个 μDMA 通道。将某个位置位，则 DMAUSEBURSTSET 寄存器中相应的 SET[n] 位即会清零。

DMA 通道采用猝发清除寄存器 (DMAUSEBURSTCLR)

基址 0x400F.F000
偏移量 0x01C
类型 WO, 复位 -



位/域	名称	类型	复位	描述
31:0	CLR[n]	WO	-	通道 [n] 的采用猝发清零

值	描述
0	没有影响
1	将某个位置位，即可将 DMAUSEBURSTSET 寄存器中的 SET[n] 清零，也就是说 μDMA 通道 [n] 可响应单次请求和猝发请求。

寄存器 12: DMA 通道请求屏蔽置位寄存器（DMAREQMASKSET），偏移量 0x020

DMAREQMASKSET 寄存器的每个位分别对应一个 µDMA 通道。将某个位置位，相应的 µDMA 通道即不再自动产生请求。读取本寄存器可返回请求屏蔽的状态。当屏蔽某个 µDMA 通道的请求后，外设将无法请求 µDMA 传输。于是该通道便可用于软件启动的 µDMA 传输流程。

DMA 通道请求屏蔽置位寄存器 (DMAREQMASKSET)

基址 0x400F.F000
偏移量 0x020
类型 R/W, 复位 0x0000.0000

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	SET[n]															
类型	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SET[n]															
类型	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位/域	名称	类型	复位	描述
31:0	SET[n]	R/W	0x0000.0000	通道[n]的请求屏蔽置位

值	描述
0	相应通道所关联的外设可以请求 µDMA 传输。
1	相应通道所关联的外设不能请求 µDMA 传输。此时该通道可用于软件启动的µDMA传输。

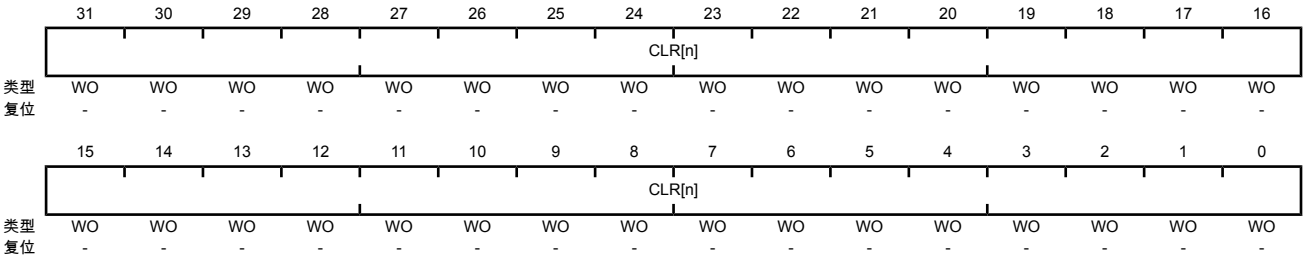
第 0 位对应于第 0 号通道。要想将本寄存器中的某个位清零，需将 DMAREQMASKCLR 寄存器的相应 CLR[n] 位置位。

寄存器 13: DMA 通道请求屏蔽清零寄存器（DMAREQMASKCLR），偏移量 0x024

DMAREQMASKCLR 寄存器的每个位分别对应一个 µDMA 通道。将某个位置位，则 DMAREQMASKSET 寄存器中相应的 SET[n] 位即会清零。

DMA 通道请求屏蔽清零寄存器 (DMAREQMASKCLR)

基址 0x400F.F000
偏移量 0x024
类型 WO, 复位 -



位/域	名称	类型	复位	描述
31:0	CLR[n]	WO	-	通道 [n] 的请求屏蔽清零

值	描述
0	没有影响
1	将某个位置位，即可将 DMAREQMASKSET 寄存器中相应的 SET[n] 清零，也就是说启用了与该通道 [n] 关联的外设，从而请求 µDMA 传输。

寄存器 14: DMA 通道启用置位寄存器（DMAENASET），偏移量 0x028

DMAENASET 寄存器的每个位分别对应一个 µDMA 通道。将某个位置位，即可使能相应的µDMA通道。读取本寄存器可返回各通道的使能状态。假如某个通道已经使能，但是屏蔽了请求（DMAREQMASKSET 寄存器置位），那么这个通道就能用于软件启动的传输。

DMA 通道启用置位寄存器 (DMAENASET)

基址 0x400F.F000
偏移量 0x028
类型 R/W, 复位 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	SET[n]															
类型	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SET[n]															
类型	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位/域	名称	类型	复位	描述
31:0	SET[n]	R/W	0x0000.0000	通道 [n] 的启用设置

值	描述
0	禁用 µDMA 通道 [n]。
1	启用 µDMA 通道 [n]。

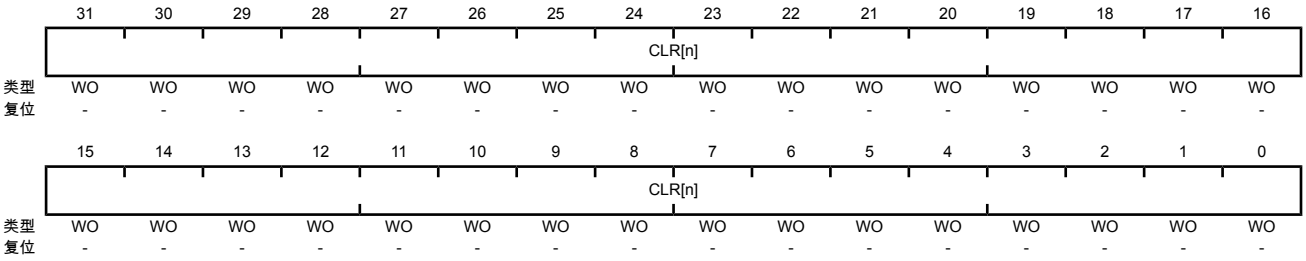
第 0 位对应于第 0 号通道。要想将本寄存器中的某个位清零，需将 DMAENACLR 寄存器的相应 CLR[n] 位置位，或等到 µDMA 传输结束。

寄存器 15: DMA 通道启用清除寄存器（DMAENACLR），偏移量 0x02C

DMAENACLR 寄存器的每个位分别对应一个 μDMA 通道。将某个位置位，则 DMAENASET 寄存器中相应的 SET[n] 位即会清零。

DMA 通道启用清除寄存器 (DMAENACLR)

基址 0x400F.F000
偏移量 0x02C
类型 WO, 复位 -



位/域	名称	类型	复位	描述
31:0	CLR[n]	WO	-	清除通道 [n] 启用清除

值	描述
0	没有影响
1	将某个位置位，即可将 DMAENASET 寄存器中的 SET[n] 清零，也就是说通道 [n] 禁用，无法用于 μDMA 传输。

注意: 当某一通道完成 μDMA 传输时，控制器会禁用它。

寄存器 16: DMA 通道主副置位寄存器（DMAALTSET），偏移量 0x030

DMAALTSET 寄存器的每个位分别对应一个 µDMA 通道。将某个位置位，相应的µDMA通道即会采用其副控制结构体。读取本寄存器可返回各通道使用哪一控制结构体。

DMA 通道主副置位寄存器 (DMAALTSET)

基址 0x400F.F000
偏移量 0x030
类型 R/W, 复位 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	SET[n]															
类型	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SET[n]															
类型	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位/域	名称	类型	复位	描述
31:0	SET[n]	R/W	0x0000.0000	通道[n]的副控制结构体置位

值	描述
0	相应通道使用主控制结构体
1	相应通道使用副控制结构体

第 0 位对应于第 0 号通道。要想将本寄存器中的某个位清零，需将 DMAALTCLR 寄存器的相应 CLR[n] 位置位。

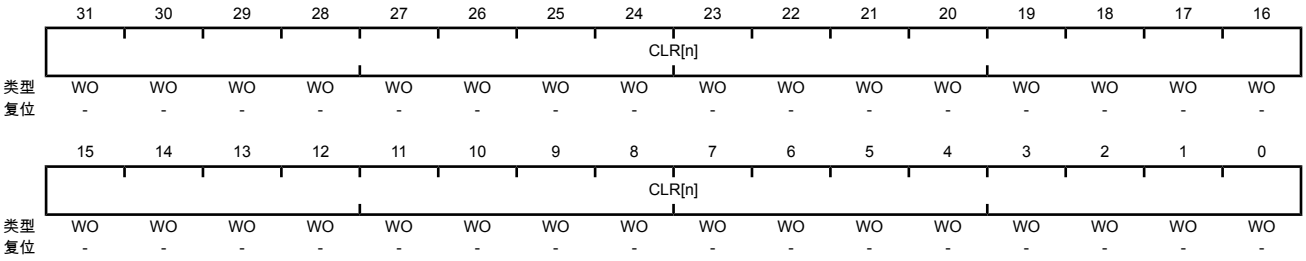
注意: 对于乒乓模式和散聚模式，µDMA控制器会自动将相应位置位，选择副控制结构体。

寄存器 17: DMA 通道主副清除寄存器（DMAALTCLR），偏移量 0x030

DMAALTCLR 寄存器的每个位分别对应一个 μDMA 通道。将某个位置位，即会将 DMAALTSET 寄存器中相应的 SET[n] 位清零。

DMA 通道主副清除寄存器 (DMAALTCLR)

基址 0x400F.F000
偏移量 0x030
类型 WO, 复位 -



位/域	名称	类型	复位	描述
31:0	CLR[n]	WO	-	通道 [n] 的副控制结构体清零

值	描述
0	没有影响
1	将某个位置位，即可将 DMAALTSET 寄存器中相应的 SET[n] 清零，也就是说通道 [n] 正在使用主控制结构体。

注意: 对于乒乓模式和散聚模式，μDMA控制器会自动将相应位置位，选择副控制结构体。

寄存器 18: DMA 通道优先置位寄存器（DMAPRIOSET），偏移量 0x038

DMAPRIOSET 寄存器的每个位分别对应一个 µDMA 通道。将某个位置位，相应的µDMA通道即会成为高优先级。读取本寄存器可获知各通道的优先级。

DMA 通道优先置位寄存器 (DMAPRIOSET)

基址 0x400F.F000
偏移量 0x038
类型 R/W, 复位 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	SET[n]															
类型	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SET[n]															
类型	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位/域	名称	类型	复位	描述
31:0	SET[n]	R/W	0x0000.0000	通道 [n] 的优先级设置

值	描述
0	相应通道为默认优先级
1	相应通道为高优先级

第 0 位对应于第 0 号通道。要想将本寄存器中的某个位清零，需将 DMAPRIOCLR 寄存器的相应 CLR[n] 位置位。

寄存器 19: DMA 通道优先清零寄存器（DMAPRIOCLR），偏移量 0x03C

DMAPRIOCLR 寄存器的每个位分别对应一个 μDMA 通道。将某个位置位，即会将 DMAPRIOSET 寄存器中相应的 SET[n] 位清零。

DMA 通道优先清零寄存器 (DMAPRIOCLR)

基址 0x400F.F000
偏移量 0x03C
类型 WO, 复位 -

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	CLR[n]															
类型	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO
复位	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	CLR[n]															
类型	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO
复位	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

位/域	名称	类型	复位	描述
31:0	CLR[n]	WO	-	通道[n]的优先级清零

值 描述

0 没有影响

1 将某个位置位，即可将 DMAPRIOSET 寄存器中的 SET[n] 清零，也就是说通道 [n] 正在使用默认优先级。

寄存器 20: DMA 总线错误清除寄存器（DMAERRCLR），偏移量 0x04C

DMA 总线错误清除寄存器 (DMAERRCLR) 用于读取并清除 µDMA 总线错误状态。如果 µDMA 控制器在执行传输过程中遭遇总线错误，即会将错误状态标志置位。如果在某个通道上发生了总线错误，µDMA 控制器会自动禁用该通道。其它通道不受影响，仍然可以继续工作。

DMA 总线错误清除寄存器 (DMAERRCLR)

基址 0x400F.F000
偏移量 0x04C
类型 R/W, 复位 0x0000.0000

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	保留															
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	保留															ERRCLR
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W1C
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位/域	名称	类型	复位	描述
31:1	保留	RO	0x0000.000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
0	ERRCLR	R/W1C	0	µDMA 总线错误状态 值 描述 0 无挂起的总线错误。 1 有挂起的总线错误。 此位写 1 清 0。

寄存器 21: DMA 通道分配寄存器（DMACHASGN），偏移量 0x500

DMACHASGN 寄存器的每个位分别对应一个 µDMA 通道。将某个位置位，会选择表9-1（514页）中规定的通道的次功能。

注意: 该寄存器用于支持传统软件。新软件应该使用 DMACHMAPn 寄存器。如果该寄存器的某位清零，则在 DMACHMAPn 寄存器的相应位会被配置为 0x0。如果在该寄存器的某位置位，则相应位域会被配置为 0x1。在读取该寄存器时，如果相应的 DMACHMAPn 寄存器位域等于 0， 则读取的位值为 0；否则，为 1（如果相应的 DMACHMAPn 寄存器位域不为 0）。

DMA 通道分配寄存器 (DMACHASGN)

基址 0x400F.F000
偏移量 0x500
类型 R/W, 复位 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	CHASGN[n]															
类型	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	CHASGN[n]															
类型	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

位/域	名称	类型	复位	描述
31:0	CHASGN[n]	R/W	-	通道 [n] 的分配选择

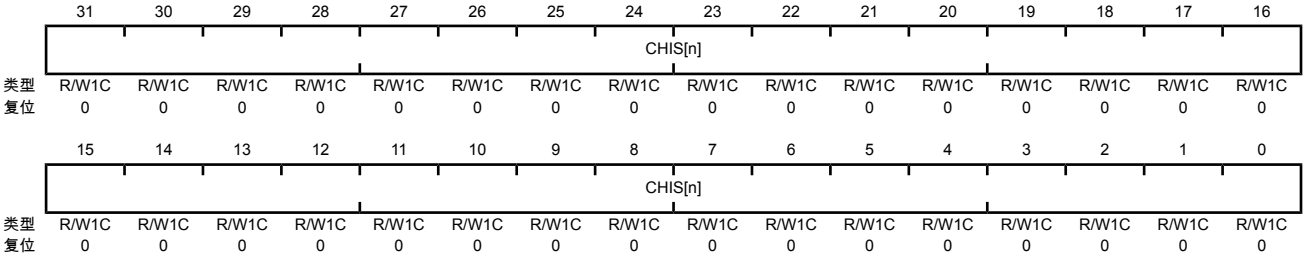
值	描述
0	按照通道的主功能工作。
1	按照通道的次功能工作。

寄存器 22: DMA 通道中断状态寄存器（DMACHIS），偏移量 0x504

DMACHIS 寄存器的每个位分别对应一个 µDMA 通道。当 µDMA 通道产生了一个完成中断时，该位置位。该位写 1 清零。

DMA 通道中断状态寄存器 (DMACHIS)

基址 0x400F.F000
偏移量 0x504
类型 R/W1C, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:0	CHIS[n]	R/W1C	0x0000.0000	通道 [n] 的中断状态

值 描述

1 相应的 µDMA 通道产生中断。

0 相应的 µDMA 通道没有产生中断。

此位写 1 清 0。

寄存器 23: DMA 通道映射选择寄存器 0（DMACHMAP0），偏移量 0x510

DMACHMAP0 寄存器的每个 4 位域用于配置 表9-1（514页） 所描述的 μDMA 通道分配。

注意: 要支持使用 DMA 通道分配 (DMACHASGN) 寄存器的传统软件，值 0x0 等于清零的 DMACHASGN；值 0x1 等于置位的 DMACHASGN 位。

DMA 通道映射选择寄存器 0 (DMACHMAP0)

基址 0x400F.F000
偏移量 0x510
类型 R/W, 复位 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	CH7SEL				CH6SEL				CH5SEL				CH4SEL			
类型	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	CH3SEL				CH2SEL				CH1SEL				CH0SEL			
类型	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位/域	名称	类型	复位	描述
31:28	CH7SEL	R/W	0x00	μDMA 通道 7 源选择 关于通道的分配参见 表9-1（514页）。
27:24	CH6SEL	R/W	0x00	μDMA 通道 6 源选择 关于通道的分配参见 表9-1（514页）。
23:20	CH5SEL	R/W	0x00	μDMA 通道 5 源选择 关于通道的分配参见 表9-1（514页）。
19:16	CH4SEL	R/W	0x00	μDMA 通道 4 源选择 关于通道的分配参见 表9-1（514页）。
15:12	CH3SEL	R/W	0x00	μDMA 通道 3 源选择 关于通道的分配参见 表9-1（514页）。
11:8	CH2SEL	R/W	0x00	μDMA 通道 2 源选择 关于通道的分配参见 表9-1（514页）。
7:4	CH1SEL	R/W	0x00	μDMA 通道 1 源选择 关于通道的分配参见 表9-1（514页）。
3:0	CH0SEL	R/W	0x00	μDMA 通道 0 源选择 关于通道的分配参见 表9-1（514页）。

寄存器 24: DMA 通道映射选择寄存器 1（DMACHMAP1），偏移量 0x514

DMACHMAP1 寄存器的每个 4 位域用于配置表9-1（514页）所描述的 µDMA 通道分配。

注意: 要支持使用 DMA 通道分配 (DMACHASGN) 寄存器的传统软件，值 0x0 等于清零的 DMACHASGN；值 0x1 等于置位的 DMACHASGN 位。

DMA 通道映射选择寄存器 1 (DMACHMAP1)

基址 0x400F.F000
偏移量 0x514
类型 R/W, 复位 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	CH15SEL				CH14SEL				CH13SEL				CH12SEL			
类型	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	CH11SEL				CH10SEL				CH9SEL				CH8SEL			
类型	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位/域	名称	类型	复位	描述
31:28	CH15SEL	R/W	0x00	µDMA 通道 15 源选择 关于通道的分配参见 表9-1（514页）。
27:24	CH14SEL	R/W	0x00	µDMA 通道 14 源选择 关于通道的分配参见 表9-1（514页）。
23:20	CH13SEL	R/W	0x00	µDMA 通道 13 源选择 关于通道的分配参见 表9-1（514页）。
19:16	CH12SEL	R/W	0x00	µDMA 通道 12 源选择 关于通道的分配参见 表9-1（514页）。
15:12	CH11SEL	R/W	0x00	µDMA 通道 11 源选择 关于通道的分配参见 表9-1（514页）。
11:8	CH10SEL	R/W	0x00	µDMA 通道 10 源选择 关于通道的分配参见 表9-1（514页）。
7:4	CH9SEL	R/W	0x00	µDMA 通道 9 源选择 关于通道的分配参见 表9-1（514页）。
3:0	CH8SEL	R/W	0x00	µDMA 通道 8 源选择 关于通道的分配参见 表9-1（514页）。

寄存器 25: DMA 通道映射选择寄存器 2（DMACHMAP2），偏移量 0x518

DMACHMAP2 寄存器的每个 4 位域用于配置表9-1（514页） 所描述的 μDMA 通道分配。

注意: 要支持使用 DMA 通道分配 (DMACHASGN) 寄存器的传统软件，值 0x0 等于清零的 DMACHASGN；值 0x1 等于置位的 DMACHASGN 位。

DMA 通道映射选择寄存器 2 (DMACHMAP2)

基址 0x400F.F000
偏移量 0x518
类型 R/W, 复位 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	CH23SEL				CH22SEL				CH21SEL				CH20SEL			
类型	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	CH19SEL				CH18SEL				CH17SEL				CH16SEL			
类型	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位/域	名称	类型	复位	描述
31:28	CH23SEL	R/W	0x00	μDMA 通道 23 源选择 关于通道的分配参见 表9-1（514页）。
27:24	CH22SEL	R/W	0x00	μDMA 通道 22 源选择 关于通道的分配参见 表9-1（514页）。
23:20	CH21SEL	R/W	0x00	μDMA 通道 21 源选择 关于通道的分配参见 表9-1（514页）。
19:16	CH20SEL	R/W	0x00	μDMA 通道 20 源选择 关于通道的分配参见 表9-1（514页）。
15:12	CH19SEL	R/W	0x00	μDMA 通道 19 源选择 关于通道的分配参见 表9-1（514页）。
11:8	CH18SEL	R/W	0x00	μDMA 通道 18 源选择 关于通道的分配参见 表9-1（514页）。
7:4	CH17SEL	R/W	0x00	μDMA 通道 17 源选择 关于通道的分配参见 表9-1（514页）。
3:0	CH16SEL	R/W	0x00	μDMA 通道 16 源选择 关于通道的分配参见 表9-1（514页）。

寄存器 26: DMA 通道映射选择寄存器 3（DMACHMAP3），偏移量 0x51C

DMACHMAP3 寄存器的每个 4 位域用于配置表9-1（514页）所描述的 µDMA 通道分配。

注意: 要支持使用 DMA 通道分配 (DMACHASGN) 寄存器的传统软件，值 0x0 等于清零的 DMACHASGN；值 0x1 等于置位的 DMACHASGN 位。

DMA 通道映射选择寄存器 3 (DMACHMAP3)

基址 0x400F.F000
偏移量 0x51C
类型 R/W, 复位 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	CH31SEL				CH30SEL				CH29SEL				CH28SEL			
类型	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	CH27SEL				CH26SEL				CH25SEL				CH24SEL			
类型	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

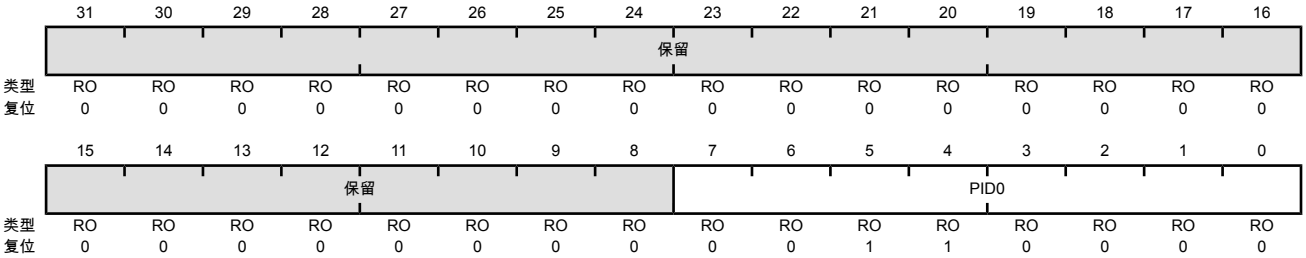
位/域	名称	类型	复位	描述
31:28	CH31SEL	R/W	0x00	µDMA 通道 31 源选择 关于通道的分配参见 表9-1（514页）。
27:24	CH30SEL	R/W	0x00	µDMA 通道 30 源选择 关于通道的分配参见 表9-1（514页）。
23:20	CH29SEL	R/W	0x00	µDMA 通道 29 源选择 关于通道的分配参见 表9-1（514页）。
19:16	CH28SEL	R/W	0x00	µDMA 通道 28 源选择 关于通道的分配参见 表9-1（514页）。
15:12	CH27SEL	R/W	0x00	µDMA 通道 27 源选择 关于通道的分配参见 表9-1（514页）。
11:8	CH26SEL	R/W	0x00	µDMA 通道 26 源选择 关于通道的分配参见 表9-1（514页）。
7:4	CH25SEL	R/W	0x00	µDMA 通道 25 源选择 关于通道的分配参见 表9-1（514页）。
3:0	CH24SEL	R/W	0x00	µDMA 通道 24 源选择 关于通道的分配参见 表9-1（514页）。

寄存器 27: DMA 外设标识寄存器 0（DMAPeriphID0），偏移量 0xFE0

DMAPeriphIDn 寄存器均为硬编码的寄存器，寄存器的位域决定复位值。

DMA 外设标识寄存器 0 (DMAPeriphID0)

基址 0x400F.F000
偏移量 0xFE0
类型 RO, 复位 0x0000.0030



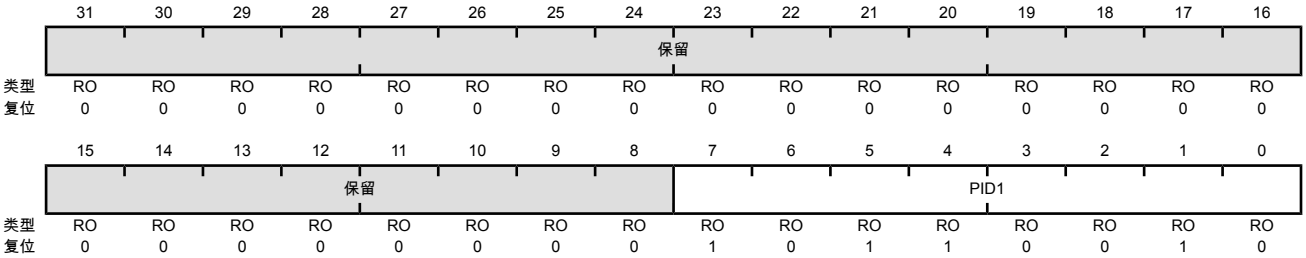
位/域	名称	类型	复位	描述
31:8	保留	RO	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
7:0	PID0	RO	0x30	μDMA 外设标识寄存器 [7:0] 可被软件用来标识该外设的存在与否。

寄存器 28: DMA 外设标识寄存器 1（DMAPeriphID1），偏移量 0xFE4

DMAPeriphIDn 寄存器均为硬编码的寄存器，寄存器的位域决定复位值。

DMA 外设标识寄存器 1 (DMAPeriphID1)

基址 0x400F.F000
偏移量 0xFE4
类型 RO, 复位 0x0000.00B2



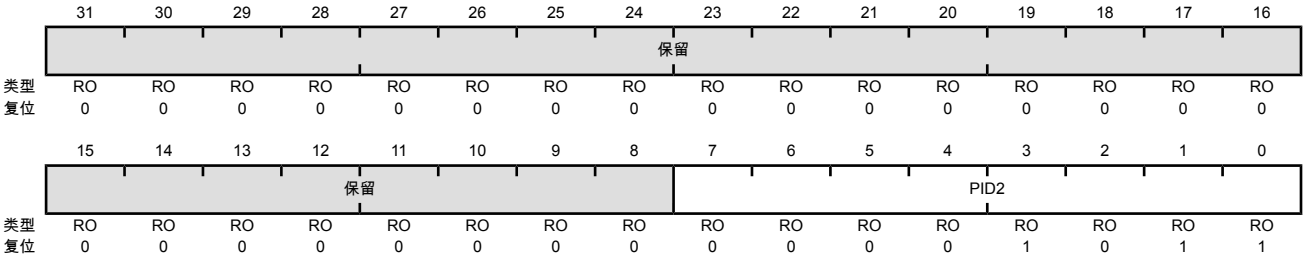
位/域	名称	类型	复位	描述
31:8	保留	RO	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
7:0	PID1	RO	0xB2	µDMA 外设标识寄存器 [15:8] 可被软件用来标识该外设的存在与否。

寄存器 29: DMA 外设标识寄存器 2（DMAPeriphID2），偏移量 0xFE8

DMAPeriphIDn 寄存器均为硬编码的寄存器，寄存器的位域决定复位值。

DMA 外设标识寄存器 2 (DMAPeriphID2)

基址 0x400F.F000
偏移量 0xFE8
类型 RO, 复位 0x0000.000B



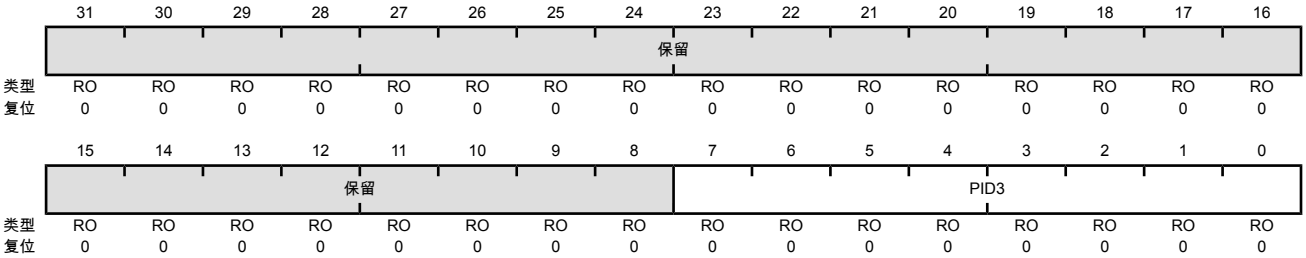
位/域	名称	类型	复位	描述
31:8	保留	RO	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
7:0	PID2	RO	0x0B	μDMA 外设 ID 寄存器 [23:16] 可被软件用来标识该外设的存在与否。

寄存器 30: DMA 外设标识寄存器 3（DMAPeriphID3），偏移量 0xFEC

DMAPeriphIDn 寄存器均为硬编码的寄存器，寄存器的位域决定复位值。

DMA 外设标识寄存器 3 (DMAPeriphID3)

基址 0x400F.F000
偏移量 0xFEC
类型 RO, 复位 0x0000.0000



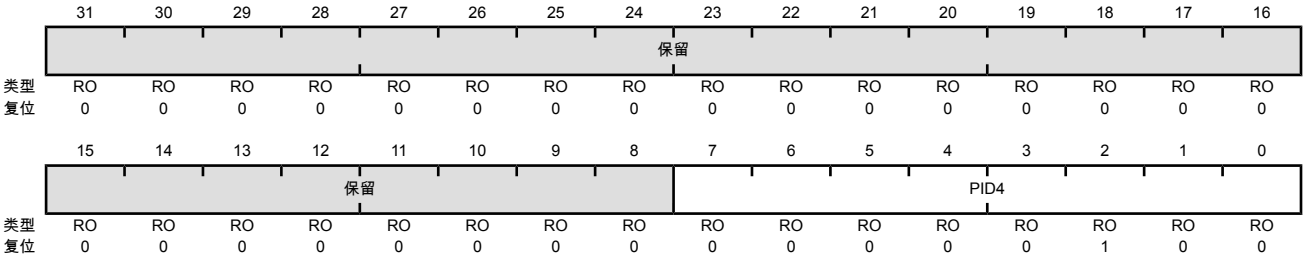
位/域	名称	类型	复位	描述
31:8	保留	RO	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
7:0	PID3	RO	0x00	µDMA 外设标识寄存器 [31:24] 可被软件用来标识该外设的存在与否。

寄存器 31: DMA 外设标识寄存器 4（DMAPeriphID4），偏移量 0xFD0

DMAPeriphIDn 寄存器均为硬编码的寄存器，寄存器的位域决定复位值。

DMA 外设标识寄存器 4 (DMAPeriphID4)

基址 0x400F.F000
偏移量 0xFD0
类型 RO, 复位 0x0000.0004



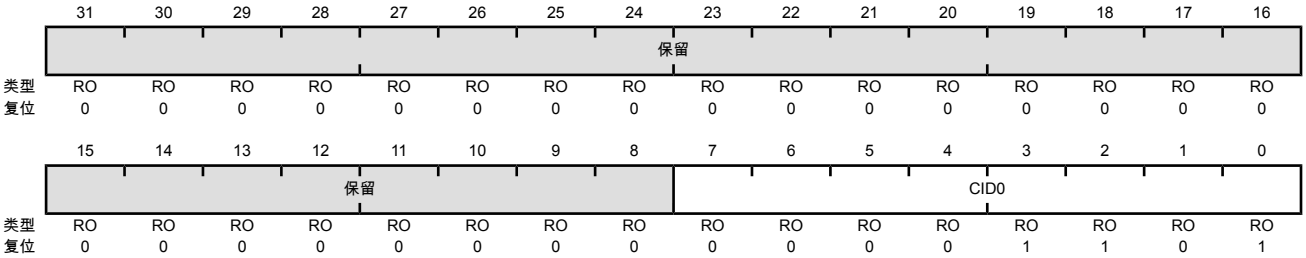
位/域	名称	类型	复位	描述
31:8	保留	RO	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
7:0	PID4	RO	0x04	μDMA外设ID寄存器 可被软件用来标识该外设的存在与否。

寄存器 32: DMA PrimeCell 标识寄存器 0（DMAPCellID0），偏移量 0xFF0

DMAPCellIDn 寄存器均为硬编码的寄存器，寄存器的位域决定复位值。

DMA PrimeCell 标识寄存器 0 (DMAPCellID0)

基址 0x400F.F000
偏移量 0xFF0
类型 RO, 复位 0x0000.000D



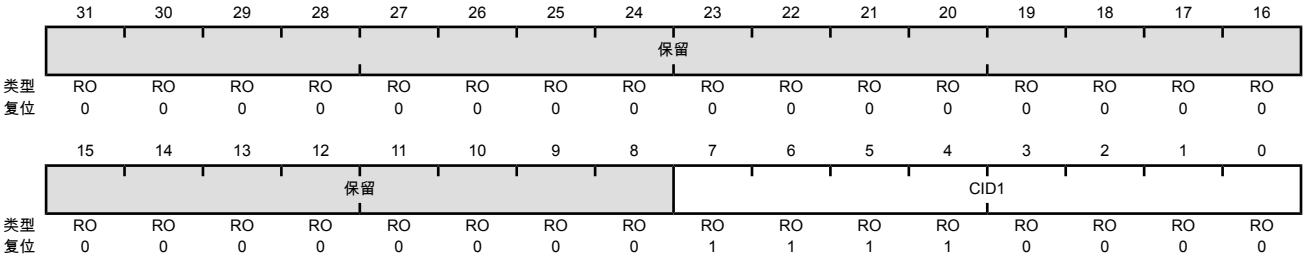
位/域	名称	类型	复位	描述
31:8	保留	RO	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
7:0	CID0	RO	0x0D	µDMA PrimeCell 标识寄存器 [7:0] 为软件提供一个标准的交叉外设识别系统。

寄存器 33: DMA PrimeCell 标识寄存器 1（DMAPCellID1），偏移量 0xFF4

DMAPCellIDn 寄存器均为硬编码的寄存器，寄存器的位域决定复位值。

DMA PrimeCell 标识寄存器 1 (DMAPCellID1)

基址 0x400F.F000
偏移量 0xFF4
类型 RO, 复位 0x0000.00F0



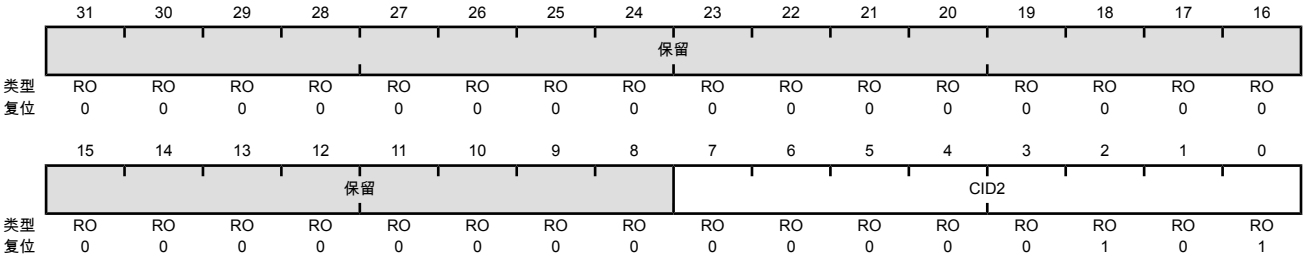
位/域	名称	类型	复位	描述
31:8	保留	RO	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
7:0	CID1	RO	0xF0	μDMA PrimeCell 标识寄存器 [15:8] 为软件提供一个标准的交叉外设识别系统。

寄存器 34: DMA PrimeCell 标识寄存器 2（DMAPCellID2），偏移量 0xFF8

DMAPCellIDn 寄存器均为硬编码的寄存器，寄存器的位域决定复位值。

DMA PrimeCell 标识寄存器 2 (DMAPCellID2)

基址 0x400F.F000
偏移量 0xFF8
类型 RO, 复位 0x0000.0005



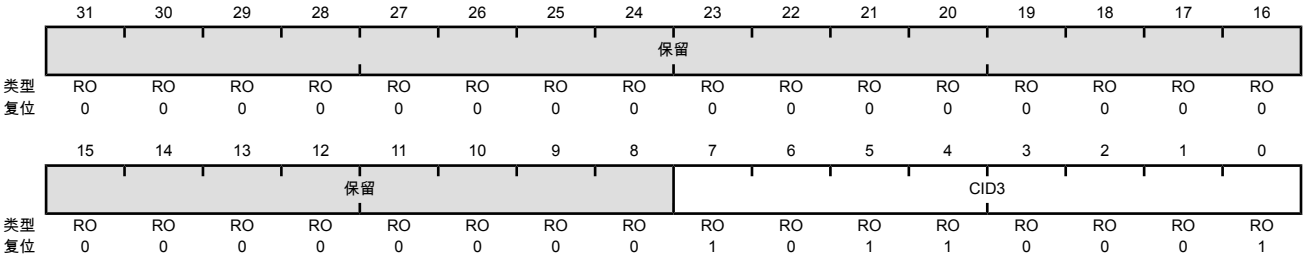
位/域	名称	类型	复位	描述
31:8	保留	RO	0x00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
7:0	CID2	RO	0x05	µDMA PrimeCell ID 寄存器 [23:16] 为软件提供一个标准的交叉外设识别系统。

寄存器 35: DMA PrimeCell 标识寄存器 3（DMAPCellID3），偏移量 0xFFC

DMAPCellIDn 寄存器均为硬编码的寄存器，寄存器的位域决定复位值。

DMA PrimeCell 标识寄存器 3 (DMAPCellID3)

基址 0x400F.F000
偏移量 0xFFC
类型 RO, 复位 0x0000.00B1



位/域	名称	类型	复位	描述
31:8	保留	RO	0x00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
7:0	CID3	RO	0xB1	μDMA PrimeCell 标识寄存器 [31:24] 为软件提供一个标准的交叉外设识别系统。

10 通用输入/输入端口（GPIOs）

GPIO 模块包括 6 个物理 GPIO 块，每个块对应一个单独的 GPIO 端口（端口 A、端口 B、端口 C、端口 D、端口 E、端口 F）。GPIO 模块支持高达 43 个可编程的输入/输出管脚。具体取决于使用的外设。

GPIO模块具有如下特性：

- 高达 43 个 GPIO，具体取决于配置
- 高度灵活的管脚复用，可配置为GPIO或任一外设功能
- 配置为输入模式可承受 5 V 电压
- 端口 A-G 可通过高级外设总线 (APB) 访问
- 快速切换能力，在 AHB 端口每个时钟周期实现一次变化；在 APB 端口每两个时钟周期实现一次变化
- 可编程控制的GPIO中断
  - 产生中断屏蔽
  - 上升沿、下降沿或是双边沿(上升沿和下降沿)触发
  - 高电平或低电平触发
- 读写操作时刻可过地址线进行位屏蔽的操作
- 可用于启动一个 ADC 采样序列或 µDMA 传输
- 在休眠模式中，可以保持管脚的状态
- 配置为数字输入的管脚均为施密特触发
- 可编程控制的GPIO引脚配置
  - 弱上拉或下拉电阻
  - 数字通信时可配置为 2 mA、4 mA 或 8 mA 驱动电流；对于需要大电流的应用，可通过多达四个管脚承载 18 mA
  - 8 mA 驱动电流的斜率控制
  - 开漏启用
  - 数字输入启用

10.1 信号描述

GPIO 信号具有复用硬件功能。下面的表格列出了所有 GPIO 管脚及其模拟和数字复用功能。当配置成输入时，除了 PD4、PD5、PB0 和 PB1 最高可承受 3.6 V 电压，其他所有 GPIO 管脚都可以承受 5 V 电压。将 GPIO 备用功能选择 (GPIOAFSEL) 和 GPIODEN 寄存器中相应的位置位，并使用如下表所示的数字编码配置 GPIO 端口控制 (GPIOPCTL) 寄存器中的 PMCx 位域，即可启用数字复用硬件功能。下表中的模拟信号也能耐受 5 V 电压，通过清零 GPIO 数字使能 (GPIODEN) 寄存

器的 DEN 位可对其进行配置。AINx 模拟信号所具备的内部电路能确保他们不会超过 V_{DD} 的电压（低于表21-1（1029页）规定的最大值），但模拟性能规范仅适用于以下条件：I/O 管脚的输入信号在 $0\ V < V_{IN} < V_{DD}$ 范围之间。请注意，每个管脚必须单独编程；表格中的列并没有任何分组的意思。表中的灰色单元格代表相应GPIO管脚的默认值。

重要: 所有的 GPIO 管脚在复位时都被配置为 GPIO 功能，而且是三态的，即 (GPIOAFSEL=0、GPIODEN=0、GPIOPDR=0、GPIOPUR=0、GPIOPCTL=0)，但是下表中列出的这些管脚除外。上电复位 ($\overline{POR}$) 或确认 $\overline{RST}$ 都会将管脚恢复其默认设置。

表 **10-1.** 具有非 **0** 复位值的 **GPIO** 管脚

引脚	默认值	GPIOAFSEL	GPIODEN	GPIOPDR	GPIOPUR	GPIOPCTL
PA[1:0]	UART0	0	0	0	0	0x1
PA[5:2]	SSI0	0	0	0	0	0x2
PB[3:2]	I^2C0	0	0	0	0	0x3
PA[3:0]	JTAG/SWD	1	1	0	1	0x1

GPIO 提交控制寄存器提供了保护层，以防止对重要硬件信号的意外编程，其中包括 JTAG/SWD 信号和 NMI 信号。就算是不配置成 JTAG/SWD 或 NMI 信号，而是把他们配置成备用功能，这些管脚也必须遵循提交控制过程；请参考 “确认控制”（581页）。

表 **10-2. GPIO** 管脚和复用功能 **(64LQFP)**

IO	管脚	模拟功能	数字功能（GPIOPCTL PMCx 位域编码）[a]										
			1	2	3	4	5	6	7	8	9	14	15
PA0	17	-	U0Rx	-	-	-	-	-	-	-	-	-	-
PA1	18	-	U0Tx	-	-	-	-	-	-	-	-	-	-
PA2	19	-	-	SSI0Clk	-	-	-	-	-	-	-	-	-
PA3	20	-	-	SSI0Fss	-	-	-	-	-	-	-	-	-
PA4	21	-	-	SSI0Rx	-	-	-	-	-	-	-	-	-
PA5	22	-	-	SSI0Tx	-	-	-	-	-	-	-	-	-
PA6	23	-	-	-	I2C1SCL	-	-	-	-	-	-	-	-
PA7	24	-	-	-	I2C1SDA	-	-	-	-	-	-	-	-
PB0	45	-	U1Rx	-	-	-	-	-	T2CCP0	-	-	-	-
PB1	46	-	U1Tx	-	-	-	-	-	T2CCP1	-	-	-	-
PB2	47	-	-	-	I2C0SCL	-	-	-	T3CCP0	-	-	-	-
PB3	48	-	-	-	I2C0SDA	-	-	-	T3CCP1	-	-	-	-
PB4	58	AIN10	-	SSI2Clk	-	-	-	-	T1CCP0	CAN0Rx	-	-	-
PB5	57	AIN11	-	SSI2Fss	-	-	-	-	T1CCP1	CAN0Tx	-	-	-
PB6	1	-	-	SSI2Rx	-	-	-	-	T0CCP0	-	-	-	-
PB7	4	-	-	SSI2Tx	-	-	-	-	T0CCP1	-	-	-	-
PC0	52	-	TCK SWCLK	-	-	-	-	-	T4CCP0	-	-	-	-
PC1	51	-	TMS SWDIO	-	-	-	-	-	T4CCP1	-	-	-	-

表 10-2. GPIO 管脚和复用功能 (64LQFP)（续）

IO	管脚	模拟功能	数字功能（GPIOPCTL PMCx 位域编码）[a]										
			1	2	3	4	5	6	7	8	9	14	15
PC2	50	-	TDI	-	-	-	-	-	T5CCP0	-	-	-	-
PC3	49	-	TDO SWO	-	-	-	-	-	T5CCP1	-	-	-	-
PC4	16	C1-	U4Rx	U1Rx	-	-	-	-	WT0CCP0	U1RTS	-	-	-
PC5	15	C1+	U4Tx	U1Tx	-	-	-	-	WT0CCP1	U1CTS	-	-	-
PC6	14	C0+	U3Rx	-	-	-	-	-	WT1CCP0	-	-	-	-
PC7	13	C0-	U3Tx	-	-	-	-	-	WT1CCP1	-	-	-	-
PD0	61	AIN7	SSI3Clk	SSI1Clk	I2C3SCL	-	-	-	WT2CCP0	-	-	-	-
PD1	62	AIN6	SSI3Fss	SSI1Fss	I2C3SDA	-	-	-	WT2CCP1	-	-	-	-
PD2	63	AIN5	SSI3Rx	SSI1Rx	-	-	-	-	WT3CCP0	-	-	-	-
PD3	64	AIN4	SSI3Tx	SSI1Tx	-	-	-	-	WT3CCP1	-	-	-	-
PD4	43	-	U6Rx	-	-	-	-	-	WT4CCP0	-	-	-	-
PD5	44	-	U6Tx	-	-	-	-	-	WT4CCP1	-	-	-	-
PD6	53	-	U2Rx	-	-	-	-	-	WT5CCP0	-	-	-	-
PD7	10	-	U2Tx	-	-	-	-	-	WT5CCP1	NMI	-	-	-
PE0	9	AIN3	U7Rx	-	-	-	-	-	-	-	-	-	-
PE1	8	AIN2	U7Tx	-	-	-	-	-	-	-	-	-	-
PE2	7	AIN1	-	-	-	-	-	-	-	-	-	-	-
PE3	6	AIN0	-	-	-	-	-	-	-	-	-	-	-
PE4	59	AIN9	U5Rx	-	I2C2SCL	-	-	-	-	CAN0Rx	-	-	-
PE5	60	AIN8	U5Tx	-	I2C2SDA	-	-	-	-	CAN0Tx	-	-	-
PF0	28	-	U1RTS	SSI1Rx	CAN0Rx	-	-	-	T0CCP0	NMI	C0o	-	-
PF1	29	-	U1CTS	SSI1Tx	-	-	-	-	T0CCP1	-	C1o	TRD1	-
PF2	30	-	-	SSI1Clk	-	-	-	-	T1CCP0	-	-	TRD0	-
PF3	31	-	-	SSI1Fss	CAN0Tx	-	-	-	T1CCP1	-	-	TRCLK	-
PF4	5	-	-	-	-	-	-	-	T2CCP0	-	-	-	-

a. 带灰色阴影的数字信号是相应 GPIO 管脚的上电默认值。 本器件不使用编码 10-13。

10.2 功能说明

每个 GPIO 端口都是同一物理模块的独立硬件实例（请参考 图10-1（578页） 和 图10-2（579页））。TM4C1231H6PM 微控制器包含 6 个端口，因此会有 6 个此种物理 GPIO 模块。请注意，并非每个模块都实施了所有的管脚。对于片内外设模块来说，一些 GPIO 管脚可作为 I/O 信号使用。GPIO 管脚的复用硬件功能请参考 表20-5（1024页）。

图 10-1. 数字 I/O 口

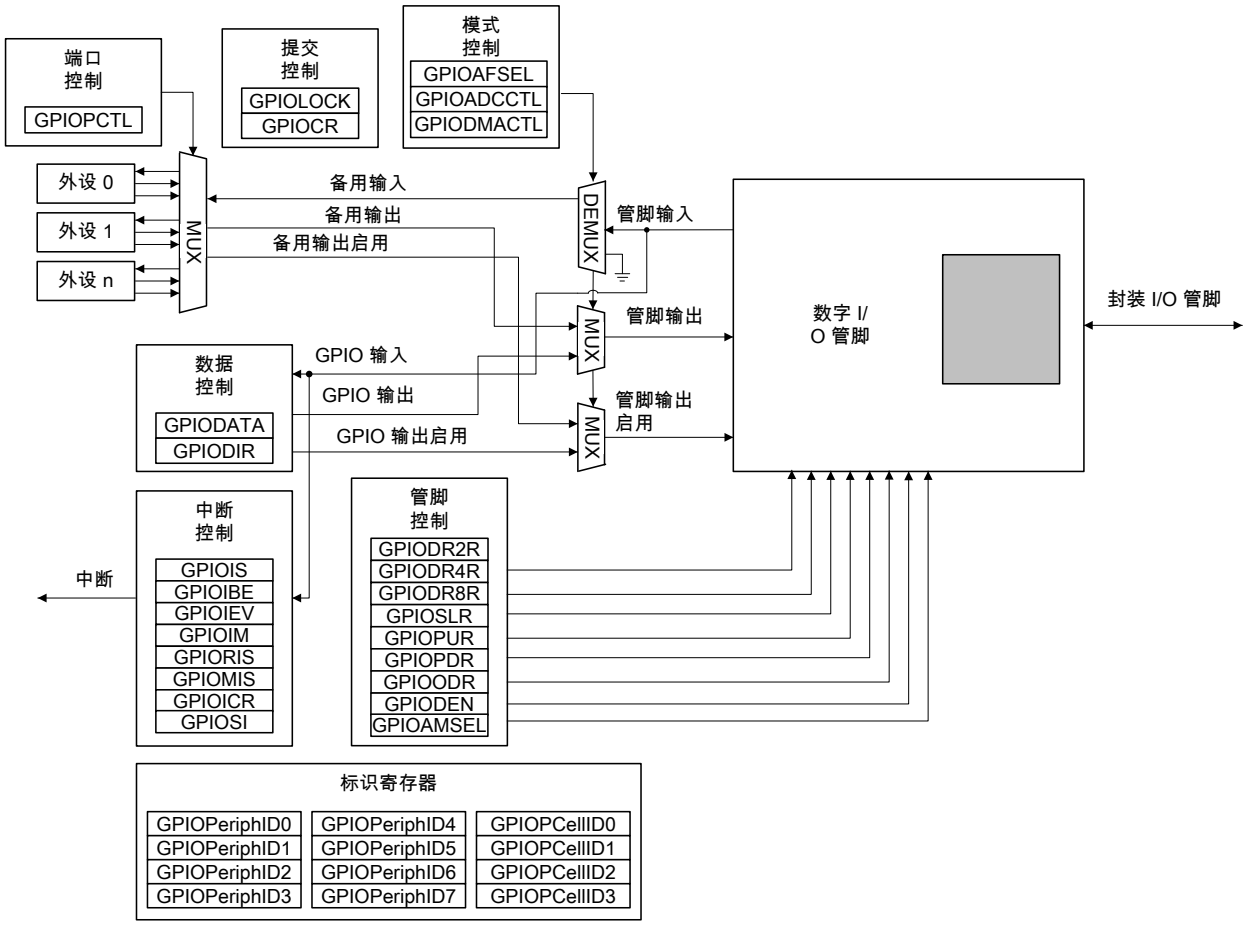
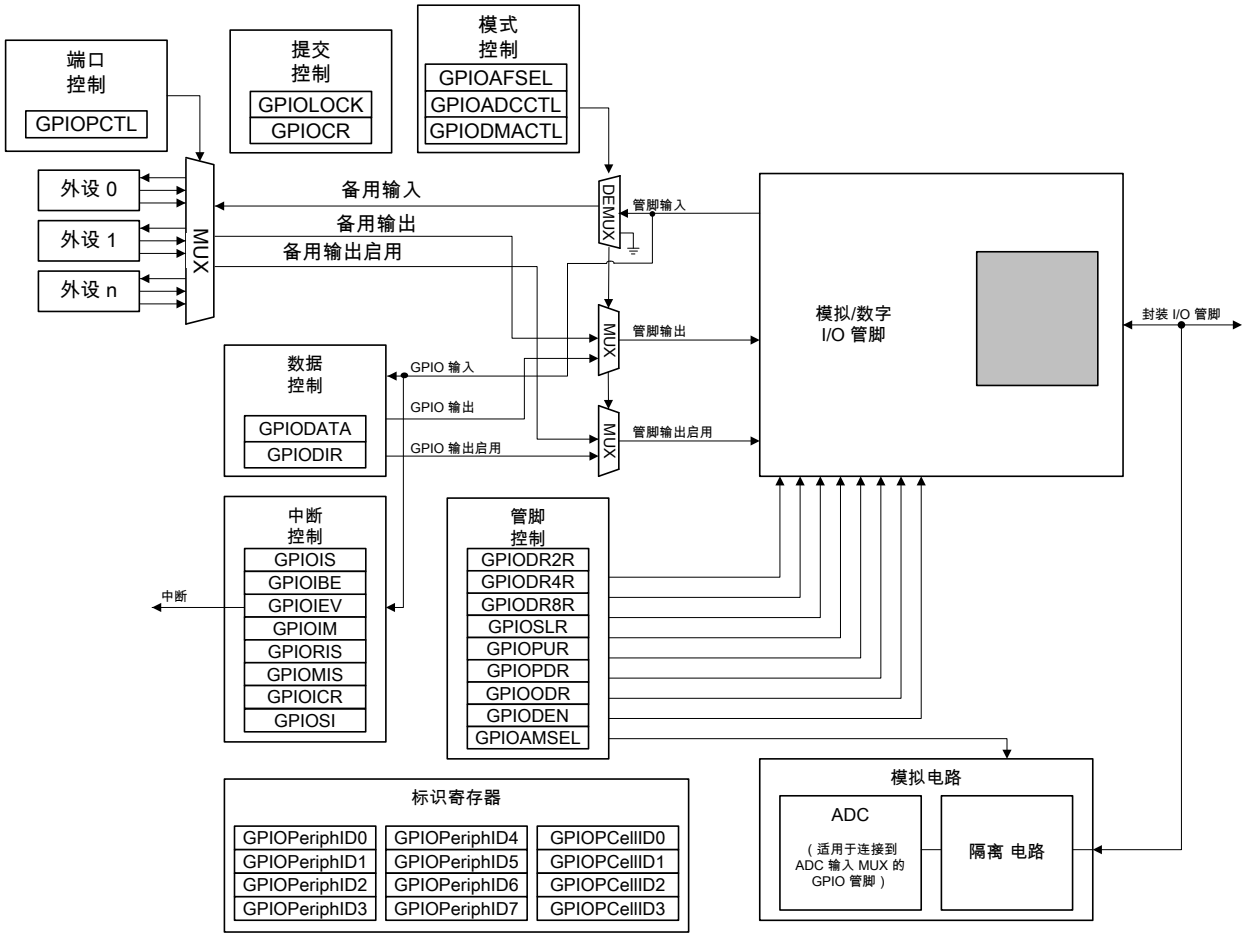

端口
控制
GPIOPCTL
提交
控制
GPIOLOCK
GPIOCR
模式
控制
GPIOAFSEL
GPIOADCCTL
GPIODMACTL
外设 0
外设 1
外设 n
MUX
备用输入
备用输出
备用输出启用
DEMUX
管脚输入
MUX
管脚输出
MUX
管脚输出
启用
数字 I/
O 管脚
封装 I/O 管脚
数据
控制
GPIODATA
GPIODIR
GPIO 输入
GPIO 输出
GPIO 输出启用
中断
控制
中断
GPIOIS
GPIOIBE
GPIOIEV
GPIOIM
GPIORIS
GPIOMIS
GPIOICR
GPIOSI
管脚
控制
GPIODR2R
GPIODR4R
GPIODR8R
GPIOSLR
GPIOPUR
GPIOPDR
GPIOODR
GPIODEN
GPIOAMSEL
标识寄存器
GPIOPeriphID0
GPIOPeriphID1
GPIOPeriphID2
GPIOPeriphID3
GPIOPeriphID4
GPIOPeriphID5
GPIOPeriphID6
GPIOPeriphID7
GPIOPCellID0
GPIOPCellID1
GPIOPCellID2
GPIOPCellID3

图 10-2. 模拟/数字 I/O 口



10.2.1 数据控制

数据控制寄存器允许软件配置 GPIO 的操作模式。当数据寄存器捕获输入数据时，数据方向寄存器将 GPIO 配置为输入；当数据寄存器通过端口输出数据时，数据方向寄存器将 GPIO 配置为输出。

小心 – 用户可以建立一个软件序列来阻止调试器连接到 **TM4C1231H6PM** 微控制器。如果将程序代码加载到 **Flash** 中会立即将 **JTAG** 管脚变成其 **GPIO** 功能，那么在 **JTAG** 管脚功能切换之前，调试器将没有足够的时间去连接和终止控制器。结果调试器可能被锁定在该部分外。通过使用一个基于外部或软件的触发器来恢复**JTAG**功能的软件程序可以避免这个问题。如果未实施软件例程，且器件锁定在此部分以外，则可通过 **TM4C1231H6PM Flash** 编程器的**“**解锁**”**功能解决此问题。有关详细信息，请参阅 **TI** 网站的 **LMFLASHPROGRAMMER** 部分。

10.2.1.1 数据方向操作

GPIO 方向 (GPIODIR) 寄存器（请参考 587页）用来将每个独立的管脚配置为输入或输出当数据方向寄存器里的位被清零时被配置为输入，相应的数据寄存器位便可以捕获并储存GPIO端口的值。当数据方向寄存器里的位被置位时被配置为输出，数据寄存器里相应的位便可以驱动GPIO端口。

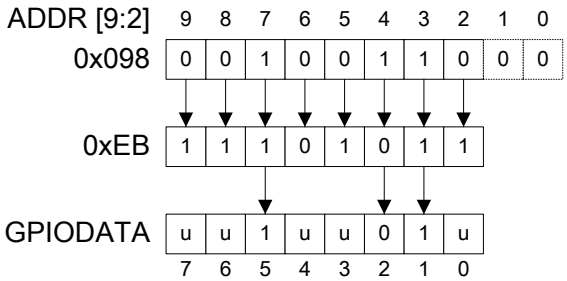
10.2.1.2 数据寄存器的操作

为了提高软件的效率，通过将地址总线的位 [9:2] 用作屏蔽位，可以对 GPIO端口的 GPIO 数据 (GPIODATA) 寄存器（请参考 586页）中的各个位进行修改。通过这种方式软件驱动程序就可以以一条指令修改任何一个GPIO管脚，而不影响其他管脚的状态。这种方式与通过“读-修改-写”来操作GPIO管脚的典型做法不同。为了提供这种特性，GPIODATA 寄存器涵盖了存储器映射中的 256 个单元。

在写入操作中，如果与数据位相关联的地址位被置位，那么 GPIODATA 寄存器的值将发生变化。如果地址位被清零，那么数据位保持不变。

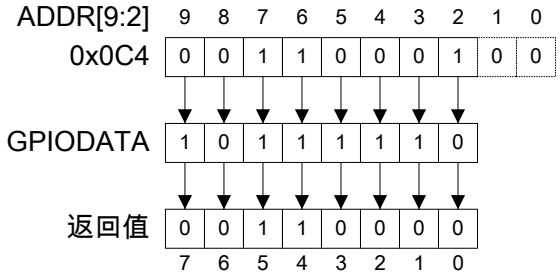
例如，将 0xEB 写入地址 GPIODATA + 0x98 处，结果将如 图10-3 所示。其中，u 表示写入操作没有改变数据。该示例演示了如何写入 GPIODATA 的位 5、2 和 1。

图 **10-3. GPIODATA** 写实例



在读操作过程中，如果与数据位相关联的地址位被置位，那么就可以读取到数据寄存器里的值。如果与数据位相关联的地址位被清零，那么不管数据寄存器里实际值是什么都读做0。例如，读取地址 GPIODATA+0x0C4 处的值，结果如 图10-4 所示。该示例演示了如何读取 GPIODATA 的位 5、4 和 0。

图 **10-4. GPIODATA** 读实例



10.2.2 中断控制

每个 GPIO 端口的中断能力都由 7 个寄存器控制。这些寄存器可以用于选择中断源、极性以及边沿属性。当一个或多个输入引发中断时，只有一个中断输出被送到整个GPIO端口的中断控制器。对于边沿触发，为了让进一步的中断可用，软件必须清除该中断。对于电平触发，必须保持住外部电平的状态才能使控制器识别中断的发生。

以下三个寄存器用来定义中断触发的类型

- GPIO 中断检测 (GPIOIS) 寄存器（请参阅588页）
- GPIO 中断双边沿 (GPIOIBE) 寄存器（请参阅589页）
- GPIO 中断事件 (GPIOIEV) 寄存器（请参阅590页）

通过 GPIO 中断屏蔽 (GPIOIM) 寄存器可以使能或禁能中断（请参阅591页）。

当产生中断条件时，可以在 GPIO 原始中断状态 (GPIORIS) 和 GPIO 屏蔽后的中断状态 (GPIOMIS) 寄存器中观察到中断信号的状态（请参考592页 和593页）。顾名思义，GPIOMIS 寄存器仅显示允许被传送到中断控制器的中断条件。GPIOIS 寄存器则用于指明 GPIO 管脚满足中断条件，但不一定发送到控制器。

对于 GPIO 电平触发中断，产生中断的信号必须保持，直到进入中断服务。当产生逻辑判断的中断致使输入信号无效时，GPIORIS 寄存器中相应的 RIS 位将清零。对于 GPIO 边沿触发中断，向 GPIO 中断清零 (GPIOICR) 寄存器的相应位写 1 即可将 GPIORIS 寄存器的 RIS 位清零（请参阅594页）。相应的 GPIOMIS 位将指示 RIS 位的屏蔽值。

在设置中断控制寄存器（GPIOIS、GPIOIBE 或 GPIOIEV）的时候，应该保持中断的屏蔽状态（GPIOIM 清零）以防止发生意外中断。如果相应的位没有屏蔽，那么向中断控制寄存器中写入任何值都有可能产生伪中断。

10.2.2.1 ADC 触发源

任何 GPIO 管脚都可以通过 GPIO ADC 控制 (GPIOADCCTL) 寄存器配置成 ADC 的外部触发源。如果 GPIO 被配置为非屏蔽的中断管脚（GPIOIM 中相应的位被置位），该端口产生中断时，就会发送一个触发信号到 ADC。如果 ADC 事件多路复用器选择 (ADCEMUX) 寄存器被配置为使用外部触发器，那么将启动 ADC 转换。请参阅748页。

请注意，如果 Port B GPIOADCCTL 寄存器被清零，PB4 也可以用作 ADC 的外部触发信号。此传统模式允许在此微控制器中运行针对上一代器件编写的代码。

10.2.2.2 µDMA 触发源

任何 GPIO 管脚都可以通过 GPIO DMA 控制 (GPIODMACTL) 寄存器配置成 µDMA 的外部触发源。如果 GPIO 被配置为非屏蔽的中断管脚（GPIOIM 中相应的位被置位），就会产生一个针对该端口的中断，并且发送一个外部的触发信号到 µDMA。如果 µDMA 配置为根据 GPIO 信号开始传输数据，那么此时就会启动传输。

10.2.3 模式控制

GPIO引脚既可以被软件控制也可以被硬件控制。软件控制是大部分引脚的默认状态，并且与 GPIO 模式对应，此时的 GPIODATA 寄存器用来读写相应的引脚。当 GPIO 备用功能选择 (GPIOAFSEL) 寄存器（请参考 595页）启用硬件控制时，管脚状态将由它的复用（即外设）功能控制。

更多的管脚复用功能选择由 GPIO 端口控制 (GPIOPCTL) 寄存器提供，该寄存器可以为每个 GPIO 选择其中一个外设功能。关于这些配置的详细信息，请参阅 表20-5（1024页）。

注意: 如果一个管脚被用作 ADC 的输入，那么 GPIOAMSEL 寄存器中相应的位必须置位，禁用模拟隔离电路。

10.2.4 确认控制

GPIO 提交控制寄存器提供了保护层以防止对重要硬件外设的意外编程。系统针对可用作四个 JTAG/SWD 管脚以及 NMI 管脚的 GPIO 管脚提供了保护功能（管脚号请参见“信号表”（1008页））。向 GPIO 备用功能选择 (GPIOAFSEL) 寄存器（请参阅595页）、GPIO 上拉电阻选择 (GPIOPUR) 寄存器（请参阅601页）、GPIO 下拉电阻选择 (GPIOPDR) 寄存器（请参阅603页）以及 GPIO 数字使能 (GPIODEN) 寄存器（请参阅606页）中受保护的位写入数据将不会确认保存，除非 GPIO 锁定 (GPIOLOCK) 寄存器（请参阅608页）没有被锁定，同时 GPIO 确认 (GPIOCR) 寄存器（请参阅609页）中相应的位被置位。

10.2.5 引脚(Pad)控制

可以根据应用程序的要求用软件来配置GPIO管脚。管脚控制寄存器包括GPIODR2R、GPIODR4R、GPIODR8R、GPIOODR、GPIOPUR、GPIOPDR、GPIOSLR 以及 GPIODEN 寄存器。这些寄存器控制着引脚的驱动电流大小，开漏配置，上拉下拉电阻选择，斜率控制和数字输入使能。如果对配置为开漏输出的 GPIO 施加了 5 V 电压，则输出电压将取决于上拉电阻的强度。GPIO 管脚并未配置为输出 5 V 电压。

10.2.6 标识

复位时配置的标识寄存器允许软件将模块当作GPIO块进行检测和识别。标识寄存器包括GPIOPeriphID0-GPIOPeriphID7 寄存器以及 GPIOPCellID0-GPIOPCellID3 寄存器。

10.3 初始化和配置

GPIO模块可以通过两个不同的存储器槽访问。传统的高级外设总线(APB)可向后兼容以前的器件。另外一种是先进高端总线(AHB)，它和APB一样拥有相同的寄存器映射，但是提供了比APB更好的访问性能。但是这两种访问方式只能选择一种使用。为指定 GPIO 端口启用的槽由 GPIOHBCTL 寄存器（请参考 225页）中相应的位来控制。注：GPIO 只可以通过 AHB 槽访问。

要将 GPIO 管脚配置为特殊端口，请按以下步骤操作：

1. 将RCGCGPIO寄存器中相应的位置位即可为端口启用时钟（请参阅295页）。此外，可按相同的方式设置 SCGCGPIO 和 DCGCGPIO 寄存器，以启用睡眠模式和深度睡眠模式的时钟。
2. 设置 GPIODIR 寄存器即可指定 GPIO 端口管脚的方向。写 1 即表示输出，写 0 即表示输入。
3. 通过配置 GPIOAFSEL 寄存器将每个位设置为 GPIO 或备用管脚。如果将某个位设置为备用管脚，则必须根据特定外设的需要设置 GPIOPCTL 寄存器的 PMCx 域。另外还可通过 GPIOADCCTL 和 GPIODMACTL 这两个寄存器将 GPIO 管脚分别设置为 ADC 或 µDMA 触发信号。
4. 通过 GPIODR2R、GPIODR4R 和 GPIODR8R 寄存器设置每个管脚的驱动强度。
5. 通过 GPIOPUR、GPIOPDR 和 GPIOODR 寄存器设置端口中每个管脚的功能：上拉、下拉或者开漏。如有需要，还可通过 GPIOSLR 寄存器设置斜率。
6. 要为 GPIO 管脚启用数字 I/O 功能，应将 GPIODEN 寄存器中相应的 DEN 位置位。要为 GPIO 管脚启用模拟功能（如可用），应将 GPIOAMSEL 寄存器中的 GPIOAMSEL 位置位。
7. 通过 GPIOIS、GPIOIBE、GPIOBE、GPIOEV 和 GPIOIM 寄存器可配置每个端口的类型、事件和中断屏蔽。
8. 软件还可选择将 GPIOLOCK 寄存器中的 LOCK 位置位，以锁定 GPIO 端口管脚的 NMI 和 JTAG/SWD 管脚的配置。

除非另行配置，内部上电复位时，所有的 GPIO 管脚都被配置成无驱动模式（三态）：GPIOAFSEL=0、GPIODEN=0、GPIOPDR=0 且 GPIOPUR=0，但 表10-1（576页）中显示的管脚不在此列。表10-3（583页）列出了GPIO端口的所有可能的配置以及实现这些配置的控制寄存器设置。表10-4（583页）显示了为 GPIO 端口的管脚 2 配置上升沿中断的方法。

表 10-3. GPIO 端口配置示例

配置	GPIO 寄存器位值[a]									
	AFSEL	DIR	ODR	DEN	PUR	PDR	DR2R	DR4R	DR8R	SLR
数字输入(GPIO)	0	0	0	1	?	?	X	X	X	X
数字输出(GPIO)	0	1	0	1	?	?	?	?	?	?
开漏输出(GPIO)	0	1	1	1	X	X	?	?	?	?
开漏输入/输出 (I2CSDA)	1	X	1	1	X	X	?	?	?	?
数字输入 (定时器 CCP)	1	X	0	1	?	?	X	X	X	X
数字输出 (定时器 PWM)	1	X	0	1	?	?	?	?	?	?
数字输入/输出 (SSI)	1	X	0	1	?	?	?	?	?	?
数字输入/输出 (UART)	1	X	0	1	?	?	?	?	?	?
模拟输入（比较器）	0	0	0	0	0	0	X	X	X	X
数字输出（比较器）	1	X	0	1	?	?	?	?	?	?

a. X=忽略（无关位）

？代表是0或1由具体情况决定，取决于配置

表 10-4. GPIO 中断配置示例

寄存器	期望的中断触发事件	管脚 2 位的值[a]							
		7	6	5	4	3	2	1	0
GPIOIS	0-边沿触发 1=电平	X	X	X	X	X	0	X	X
GPIOIBE	0-单边沿触发 1=双边沿	X	X	X	X	X	0	X	X
GPIOIEV	0 = 低电平，或下降沿 1 = 高电平，或上升沿	X	X	X	X	X	1	X	X
GPIOIM	0-屏蔽 1=不屏蔽	0	0	0	0	0	1	0	0

a. X=忽略（无关位）

10.4 寄存器映射

表 10-6（584页）列出了 GPIO 寄存器。每一个 GPIO 端口都可通过两种总线槽访问。传统的高级外设总线 (APB) 可向后兼容以前的器件。另外一种是先进高端总线 (AHB)，它和 APB 总线一样拥有相同的寄存器映射，但是提供了更好的连续访问性能。

重要: 本章的 GPIO 寄存器在每个 GPIO 块中都是相同的，但是根据块的不同，8 个位可能并不是全部与 GPIO 端口相连。向未连接的位写数据没有任何效果，而读取未连接的位的数据没有任何意义。有关本器件提供的 GPIO，请参阅“信号描述”（575页）。

以下偏移量代表该寄存器相对于 GPIO 端口基址的十六进制增量地址：

- GPIO 端口 A (APB)：0x4000.4000
- GPIO 端口 A (AHB)：0x4005.8000
- GPIO 端口 B (APB)：0x4000.5000
- GPIO 端口 B (AHB)：0x4005.9000
- GPIO 端口 C (APB)：0x4000.6000

- GPIO 端口 C (AHB)：0x4005.A000
- GPIO 端口 D (APB)：0x4000.7000
- GPIO 端口 D (AHB)：0x4005.B000
- GPIO 端口 E (APB)：0x4002.4000
- GPIO 端口 E (AHB)：0x4005.C000
- GPIO 端口 F (APB)：0x4002.5000
- GPIO 端口 F (AHB)：0x4005.D000

注意配置这些寄存器之前必须先启用 GPIO 模块的时钟（请参阅295页）。启用 GPIO 模块时钟以后，必须等待三个系统时钟才能访问 GPIO 模块的寄存器。

重要: 所有的 GPIO 管脚在复位时都被配置为 GPIO 功能，而且是三态的，即 (GPIOAFSEL=0、GPIODEN=0、GPIOPDR=0、GPIOPUR=0、GPIOPCTL=0)，但是下表中列出的这些管脚除外。上电复位 (POR) 或确认 RST 都会将管脚恢复其默认设置。

表 **10-5.** 具有非 **0** 复位值的 **GPIO** 管脚

引脚	默认值	GPIOAFSEL	GPIODEN	GPIOPDR	GPIOPUR	GPIOPCTL
PA[1:0]	UART0	0	0	0	0	0x1
PA[5:2]	SSI0	0	0	0	0	0x2
PB[3:2]	I^2C0	0	0	0	0	0x3
PA[3:0]	JTAG/SWD	1	1	0	1	0x1

GPIO 提交控制寄存器提供了保护层，以防止对重要硬件信号的意外编程，其中包括 JTAG/SWD 信号和 NMI 信号。就算是不配置成 JTAG/SWD 或 NMI 信号，而是把他们配置成备用功能，这些管脚也必须遵循提交控制过程；请参考 “确认控制”（581页）。

除了 NMI 管脚和四个 JTAG/SWD 管脚之外，所有 GPIO 管脚的 GPIOCR 寄存器的默认类型是 RO（管脚号请参见“信号表”（1008页））。GPIOCR 寄存器当前仅保护这六个 GPIO 管脚。鉴于此，相应 GPIO 端口的寄存器类型为 R/W。

除了 NMI 管脚和四个 JTAG/SWD 管脚之外，所有 GPIO 管脚的 GPIOCR 寄存器的默认复位值是 0x0000.00FF（管脚号请参见“信号表”（1008页））。为了确保 JTAG 和 NMI 管脚不会被意外地编程为 GPIO 管脚，这些管脚默认是锁定的，以防止犯错。鉴于此，相应端口的 GPIOCR 默认复位值发生变化。

表 **10-6. GPIO** 触发 寄存器映射

偏移量	名称	类型	复位	描述	见页面
0x000	GPIODATA	R/W	0x0000.0000	GPIO 数据寄存器	586
0x400	GPIODIR	R/W	0x0000.0000	GPIO 方向寄存器	587
0x404	GPIOIS	R/W	0x0000.0000	GPIO 中断检测寄存器	588
0x408	GPIOIBE	R/W	0x0000.0000	GPIO 中断双边沿	589
0x40C	GPIOIEV	R/W	0x0000.0000	GPIO 中断事件寄存器	590
0x410	GPIOIM	R/W	0x0000.0000	GPIO 中断屏蔽寄存器	591
0x414	GPIORIS	RO	0x0000.0000	GPIO 原始中断状态寄存器	592
0x418	GPIOMIS	RO	0x0000.0000	GPIO 屏蔽中断状态寄存器	593
0x41C	GPIOICR	W1C	0x0000.0000	GPIO 中断清除寄存器	594

表 10-6. GPIO 触发 寄存器映射（续）

偏移量	名称	类型	复位	描述	见页面
0x420	GPIOAFSEL	R/W	-	GPIO 备用功能选择寄存器	595
0x500	GPIODR2R	R/W	0x0000.00FF	GPIO 2-mA 驱动选择寄存器	597
0x504	GPIODR4R	R/W	0x0000.0000	GPIO 4-mA 驱动选择寄存器	598
0x508	GPIODR8R	R/W	0x0000.0000	GPIO 8-mA 驱动选择寄存器	599
0x50C	GPIOODR	R/W	0x0000.0000	GPIO 开漏选择寄存器	600
0x510	GPIOPUR	R/W	-	GPIO 上拉电阻选择寄存器	601
0x514	GPIOPDR	R/W	0x0000.0000	GPIO 下拉电阻选择寄存器	603
0x518	GPIOSLR	R/W	0x0000.0000	GPIO 斜率控制选择寄存器	605
0x51C	GPIODEN	R/W	-	GPIO 数字使能寄存器	606
0x520	GPIOLOCK	R/W	0x0000.0001	GPIO 锁定寄存器	608
0x524	GPIOCR	-	-	GPIO 确认寄存器	609
0x528	GPIOAMSEL	R/W	0x0000.0000	GPIO 模拟选择寄存器	611
0x52C	GPIOPCTL	R/W	-	GPIO端口控制寄存器	612
0x530	GPIOADCCTL	R/W	0x0000.0000	GPIO ADC 控制寄存器	614
0x534	GPIODMACTL	R/W	0x0000.0000	GPIO DMA 控制寄存器	615
0xFD0	GPIOPeriphID4	RO	0x0000.0000	GPIO 外设标识寄存器 4	616
0xFD4	GPIOPeriphID5	RO	0x0000.0000	GPIO 外设标识寄存器 5	617
0xFD8	GPIOPeriphID6	RO	0x0000.0000	GPIO 外设标识寄存器 6	618
0xFDC	GPIOPeriphID7	RO	0x0000.0000	GPIO 外设标识寄存器 7	619
0xFE0	GPIOPeriphID0	RO	0x0000.0061	GPIO 外设标识寄存器 0	620
0xFE4	GPIOPeriphID1	RO	0x0000.0000	GPIO 外设标识寄存器 1	621
0xFE8	GPIOPeriphID2	RO	0x0000.0018	GPIO 外设标识寄存器 2	622
0xFEC	GPIOPeriphID3	RO	0x0000.0001	GPIO 外设标识寄存器 3	623
0xFF0	GPIOPCellID0	RO	0x0000.000D	GPIO PrimeCell 标识寄存器 0	624
0xFF4	GPIOPCellID1	RO	0x0000.00F0	GPIO PrimeCell 标识寄存器 1	625
0xFF8	GPIOPCellID2	RO	0x0000.0005	GPIO PrimeCell 标识寄存器 2	626
0xFFC	GPIOPCellID3	RO	0x0000.00B1	GPIO PrimeCell 标识寄存器 3	627

10.5 寄存器描述

在本章的剩余部分列出并描述了GPIO寄存器，以偏移量递增的序列来描述。

寄存器 1: GPIO 数据寄存器（GPIODATA），偏移量 0x000

GPIODATA 寄存器是数据寄存器。在软件控制模式中，如果通过 GPIO 方向 (GPIODIR) 寄存器（请参考 587页）将各个管脚配置成输出，那么写入 GPIODATA 寄存器的值将被发送到 GPIO 端口管脚。

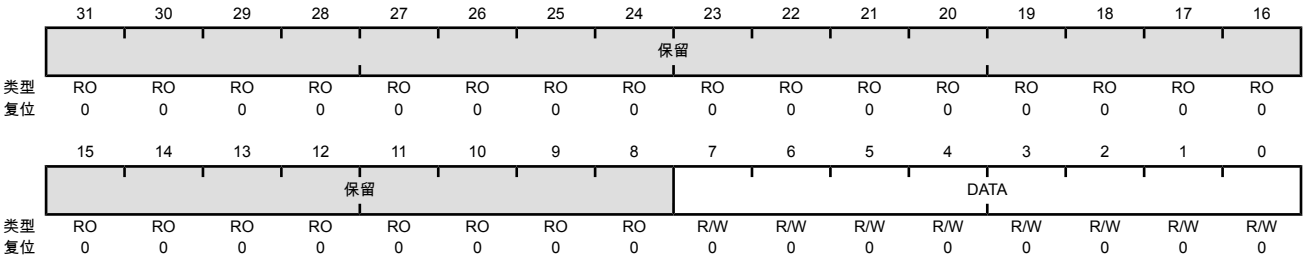
为了对 GPIODATA 寄存器执行写操作，由地址总线位 [9:2] 产生的相关屏蔽位必须置位。否则，该位的值不会被写操作改变。

同样，从该寄存器读取的值由从访问数据寄存器的地址处获取的屏蔽位[9:2]的情况来决定。如果地址屏蔽位为 1，那么读取 GPIODATA 中相应位的值；如果地址屏蔽位为 0，那么不管 GPIODATA 中相应位的值是什么，都会将它们读作 0。

如果相应的管脚被配置成输出，那么读取 GPIODATA 将返回最后写入的位值；或者当这些管脚被配置成输入时，将返回相应的输入管脚上的值。复位时所有的位都是清零的。

GPIO 数据寄存器 (GPIODATA)

GPIO 端口 A (APB) 基址: 0x4000.4000
GPIO 端口 A (AHB) 基址: 0x4005.8000
GPIO 端口 B (APB) 基址: 0x4000.5000
GPIO 端口 B (AHB) 基址: 0x4005.9000
GPIO 端口 C (APB) 基址: 0x4000.6000
GPIO 端口 C (AHB) 基址: 0x4005.A000
GPIO 端口 D (APB) 基址: 0x4000.7000
GPIO 端口 D (AHB) 基址: 0x4005.B000
GPIO 端口 E (APB) 基址: 0x4002.4000
GPIO 端口 E (AHB) 基址: 0x4005.C000
GPIO 端口 F (APB) 基址: 0x4002.5000
GPIO 端口 F (AHB) 基址: 0x4005.D000
偏移量 0x000
类型 R/W, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:8	保留	RO	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
7:0	DATA	R/W	0x00	GPIO数据寄存器 该寄存器被虚拟地映射到地址空间的 256 个单元中。为便于通过单独的驱动器读写这些寄存器，从这些寄存器读取的值和写入这些寄存器的值可以通过八条地址线 [9:2] 屏蔽。读取该寄存器将返回其当前状态。写入该寄存器仅影响那些没有被 ADDR[9:2] 屏蔽的位和被配置成输出的位。关于读写操作的实例，请参考 “数据寄存器的操作”（580页）。

寄存器 2: GPIO 方向寄存器（GPIODIR），偏移量 0x400

GPIODIR 寄存器是数据方向寄存器。GPIODIR 寄存器中的某位置位会将相应的管脚配置成输出，而清零的话就是配置为输入。复位时，所有的位都被清零，意味着所有GPIO端口默认为输入状态。

GPIO 方向寄存器 (GPIODIR)

GPIO 端口 A (APB) 基址: 0x4000.4000
GPIO 端口 A (AHB) 基址: 0x4005.8000
GPIO 端口 B (APB) 基址: 0x4000.5000
GPIO 端口 B (AHB) 基址: 0x4005.9000
GPIO 端口 C (APB) 基址: 0x4000.6000
GPIO 端口 C (AHB) 基址: 0x4005.A000
GPIO 端口 D (APB) 基址: 0x4000.7000
GPIO 端口 D (AHB) 基址: 0x4005.B000
GPIO 端口 E (APB) 基址: 0x4002.4000
GPIO 端口 E (AHB) 基址: 0x4005.C000
GPIO 端口 F (APB) 基址: 0x4002.5000
GPIO 端口 F (AHB) 基址: 0x4005.D000
偏移量 0x400
类型 R/W, 复位 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	保留															
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	保留								DIR							
类型	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

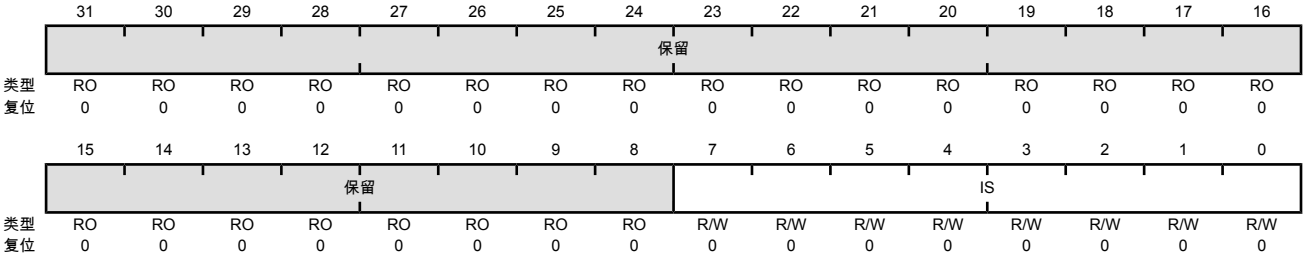
位/域	名称	类型	复位	描述
31:8	保留	RO	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
7:0	DIR	R/W	0x00	GPIO数据方向寄存器 值 描述 0 相应的引脚为输入。 1 相应的引脚为输出。

寄存器 3: GPIO 中断检测寄存器（GPIOIS），偏移量 0x404

GPIOIS 是中断检测寄存器。GPIOIS 中的位置位时，相应的管脚被配置为电平触发，清零时配置为边沿触发。复位时所有的位都是清零的。

GPIO 中断检测寄存器 (GPIOIS)

GPIO 端口 A (APB) 基址: 0x4000.4000
GPIO 端口 A (AHB) 基址: 0x4005.8000
GPIO 端口 B (APB) 基址: 0x4000.5000
GPIO 端口 B (AHB) 基址: 0x4005.9000
GPIO 端口 C (APB) 基址: 0x4000.6000
GPIO 端口 C (AHB) 基址: 0x4005.A000
GPIO 端口 D (APB) 基址: 0x4000.7000
GPIO 端口 D (AHB) 基址: 0x4005.B000
GPIO 端口 E (APB) 基址: 0x4002.4000
GPIO 端口 E (AHB) 基址: 0x4005.C000
GPIO 端口 F (APB) 基址: 0x4002.5000
GPIO 端口 F (AHB) 基址: 0x4005.D000
偏移量 0x404
类型 R/W, 复位 0x0000.0000



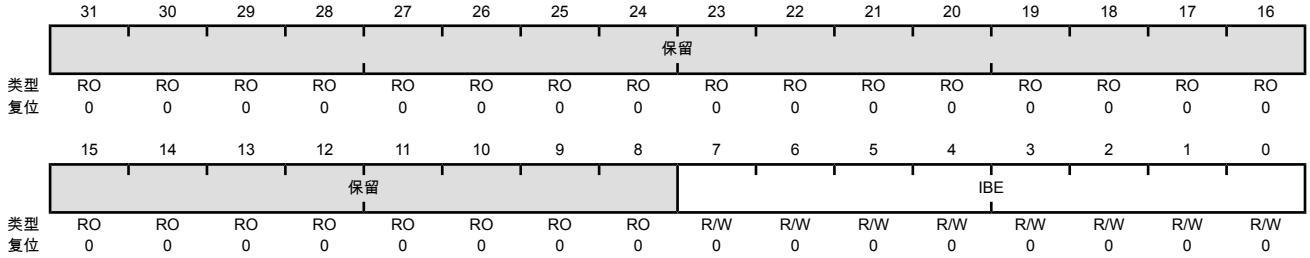
位/域	名称	类型	复位	描述
31:8	保留	RO	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
7:0	IS	R/W	0x00	GPIO中断检测类型 值 描述 0 相应的引脚为边沿触发。 1 相应的引脚为电平触发。

寄存器 4: GPIO 中断双边沿（GPIOIBE），偏移量 0x408

GPIOIBE 寄存器允许双边沿触发中断。当 GPIO 中断检测 (GPIOIS) 寄存器（请参考 588页）相应的位置位时，表示边沿触发。此时，如果 GPIOIBE 寄存器中相应的位置位，则表示双边沿触发，即检测上升沿和下降沿，不必考虑 GPIO 中断事件 (GPIOIEV) 寄存器（请参考 590页）的设置如何。将其中的位清零会将引脚配置为由 GPIOIEV 寄存器控制。复位时所有的位都是清零的。

GPIO 中断双边沿 (GPIOIBE)

GPIO 端口 A (APB) 基址: 0x4000.4000
GPIO 端口 A (AHB) 基址: 0x4005.8000
GPIO 端口 B (APB) 基址: 0x4000.5000
GPIO 端口 B (AHB) 基址: 0x4005.9000
GPIO 端口 C (APB) 基址: 0x4000.6000
GPIO 端口 C (AHB) 基址: 0x4005.A000
GPIO 端口 D (APB) 基址: 0x4000.7000
GPIO 端口 D (AHB) 基址: 0x4005.B000
GPIO 端口 E (APB) 基址: 0x4002.4000
GPIO 端口 E (AHB) 基址: 0x4005.C000
GPIO 端口 F (APB) 基址: 0x4002.5000
GPIO 端口 F (AHB) 基址: 0x4005.D000
偏移量 0x408
类型 R/W, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:8	保留	RO	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
7:0	IBE	R/W	0x00	GPIO双边沿检测类型

值 描述

0 中断发生由 GPIO 中断事件 (GPIOIEV) 寄存器控制（请参阅590 页）。

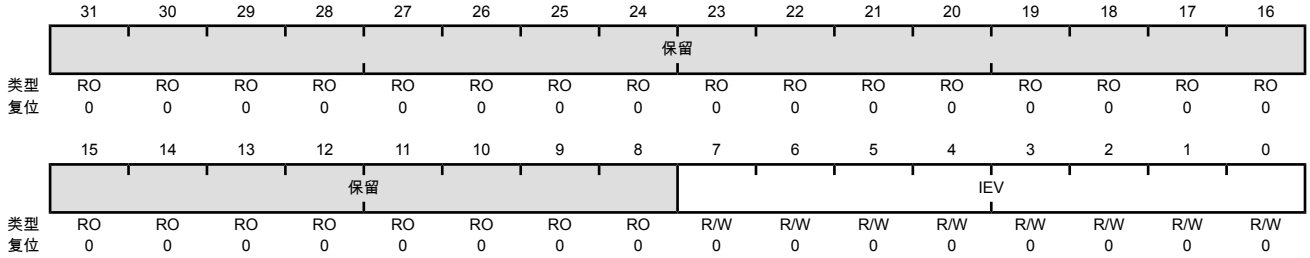
1 双边沿触发中断。

寄存器 5: GPIO 中断事件寄存器（GPIOIEV），偏移量 0x40C

GPIOIEV 寄存器是中断事件寄存器。当 GPIOIEV 寄存器中某位置位时，相应的管脚由上升沿或是高电平触发中断，具体由 GPIO 中断检测 (GPIOIS) 寄存器（请参考 588页）中的位控制。清零一个位则会将引脚配置为由下降沿或是低电平触发，具体取决于 GPIOIS 寄存器中相应位的值。复位时所有的位都是清零的。

GPIO 中断事件寄存器 (GPIOIEV)

GPIO 端口 A (APB) 基址: 0x4000.4000
GPIO 端口 A (AHB) 基址: 0x4005.8000
GPIO 端口 B (APB) 基址: 0x4000.5000
GPIO 端口 B (AHB) 基址: 0x4005.9000
GPIO 端口 C (APB) 基址: 0x4000.6000
GPIO 端口 C (AHB) 基址: 0x4005.A000
GPIO 端口 D (APB) 基址: 0x4000.7000
GPIO 端口 D (AHB) 基址: 0x4005.B000
GPIO 端口 E (APB) 基址: 0x4002.4000
GPIO 端口 E (AHB) 基址: 0x4005.C000
GPIO 端口 F (APB) 基址: 0x4002.5000
GPIO 端口 F (AHB) 基址: 0x4005.D000
偏移量 0x40C
类型 R/W, 复位 0x0000.0000



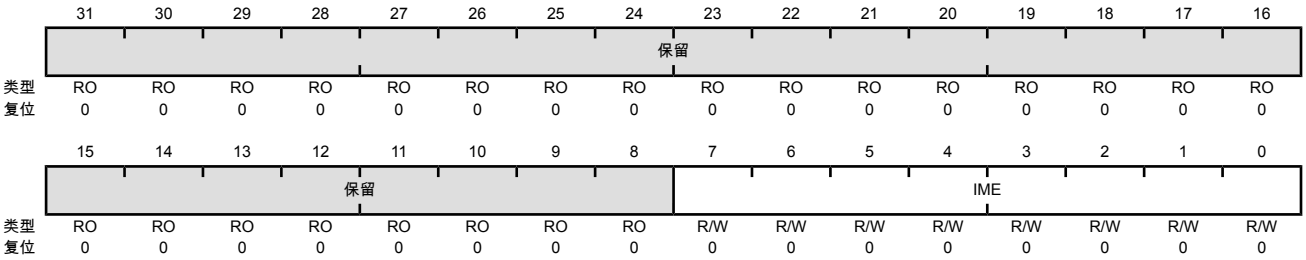
位/域	名称	类型	复位	描述
31:8	保留	RO	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
7:0	IEV	R/W	0x00	GPIO中断事件 值 描述 0 下降沿或是低电平触发。 1 下降沿或是高电平触发。

寄存器 6: GPIO 中断屏蔽寄存器（GPIOIM），偏移量 0x410

GPIOIM 是中断屏蔽寄存器。当 GPIOIM 寄存器置位时，相应的位产生的中断允许被送到联合中断信号上的中断控制器。清零GPIOIM寄存器的位则相应的管脚产生的中断不会被送到中断控制器。复位时所有的位都是清零的。

GPIO 中断屏蔽寄存器 (GPIOIM)

GPIO 端口 A (APB) 基址: 0x4000.4000
GPIO 端口 A (AHB) 基址: 0x4005.8000
GPIO 端口 B (APB) 基址: 0x4000.5000
GPIO 端口 B (AHB) 基址: 0x4005.9000
GPIO 端口 C (APB) 基址: 0x4000.6000
GPIO 端口 C (AHB) 基址: 0x4005.A000
GPIO 端口 D (APB) 基址: 0x4000.7000
GPIO 端口 D (AHB) 基址: 0x4005.B000
GPIO 端口 E (APB) 基址: 0x4002.4000
GPIO 端口 E (AHB) 基址: 0x4005.C000
GPIO 端口 F (APB) 基址: 0x4002.5000
GPIO 端口 F (AHB) 基址: 0x4005.D000
偏移量 0x410
类型 R/W, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:8	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
7:0	IME	R/W	0x00	GPIO中断屏蔽

值 描述

0 相应管脚的中断被屏蔽。

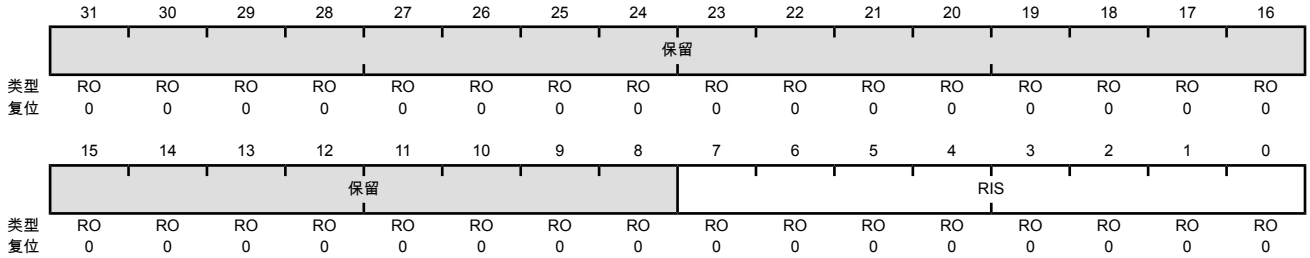
1 相应管教的中断不屏蔽，被发送到中断控制器。

寄存器 7: GPIO 原始中断状态寄存器（GPIORIS），偏移量 0x414

GPIORIS 是原始中断状态寄存器。当一个管脚上发生中断时GPIORIS寄存器被置位。当 GPIO 中断屏蔽 (GPIOIM) 寄存器（请参考 591页）中的某位置位时，相应的中断被送到中断控制器。读出来的某位为零则表示相应的位未发生中断。对于 GPIO 电平触发中断，产生中断的信号必须保持，直到进入中断服务。当产生逻辑判断的中断致使输入信号无效时，GPIORIS 寄存器中相应的 RIS 位将清零。对于 GPIO 边沿触发中断，向 GPIO 中断清零 (GPIOICR) 寄存器的相应位写 1 即可将 GPIORIS 寄存器的的 RIS 位清零。相应的 GPIOMIS 位将指示 RIS 位的屏蔽值。

GPIO 原始中断状态寄存器 (GPIORIS)

GPIO 端口 A (APB) 基址: 0x4000.4000
GPIO 端口 A (AHB) 基址: 0x4005.8000
GPIO 端口 B (APB) 基址: 0x4000.5000
GPIO 端口 B (AHB) 基址: 0x4005.9000
GPIO 端口 C (APB) 基址: 0x4000.6000
GPIO 端口 C (AHB) 基址: 0x4005.A000
GPIO 端口 D (APB) 基址: 0x4000.7000
GPIO 端口 D (AHB) 基址: 0x4005.B000
GPIO 端口 E (APB) 基址: 0x4002.4000
GPIO 端口 E (AHB) 基址: 0x4005.C000
GPIO 端口 F (APB) 基址: 0x4002.5000
GPIO 端口 F (AHB) 基址: 0x4005.D000
偏移量 0x414
类型 RO, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:8	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
7:0	RIS	RO	0x00	GPIO原始中断状态 值 描述 0 相应的管脚上未产生中断。 1 相应的管教上已经有中断产生。 对于边沿触发中断，可通过向 GPIOICR 寄存器的相应位写 1 来将此位清零。 对于 GPIO 电平触发中断，此位在电平无效时清零。

寄存器 8: GPIO 屏蔽中断状态寄存器（GPIOMIS），偏移量 0x418

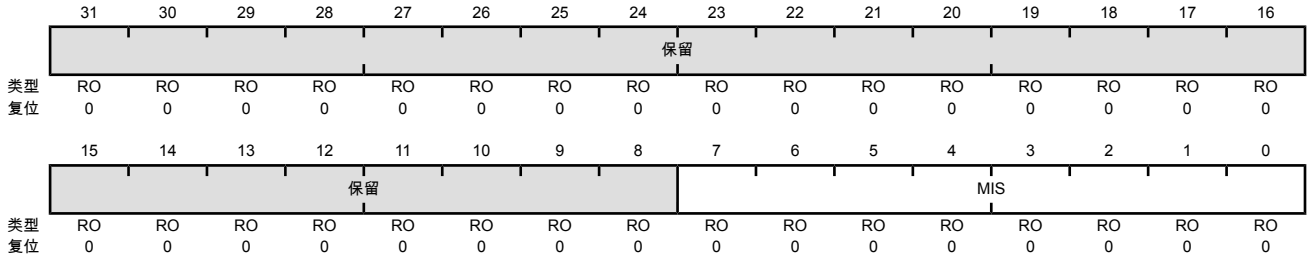
GPIOMIS 是屏蔽中断状态寄存器。如果该寄存器中的某个位置位，则说明相应的中断已经发送到中断控制器。如果某位清零，表示没有产生中断，或者中断被屏蔽。

请注意，如果 Port B GPIOADCCTL 寄存器被清零，PB4 也可以用作 ADC 的外部触发信号。此传统模式允许在此微控制器中运行针对上一代器件编写的代码。

GPIOMIS 是屏蔽后的中断状态寄存器。

GPIO 屏蔽中断状态寄存器 (GPIOMIS)

GPIO 端口 A (APB) 基址: 0x4000.4000
GPIO 端口 A (AHB) 基址: 0x4005.8000
GPIO 端口 B (APB) 基址: 0x4000.5000
GPIO 端口 B (AHB) 基址: 0x4005.9000
GPIO 端口 C (APB) 基址: 0x4000.6000
GPIO 端口 C (AHB) 基址: 0x4005.A000
GPIO 端口 D (APB) 基址: 0x4000.7000
GPIO 端口 D (AHB) 基址: 0x4005.B000
GPIO 端口 E (APB) 基址: 0x4002.4000
GPIO 端口 E (AHB) 基址: 0x4005.C000
GPIO 端口 F (APB) 基址: 0x4002.5000
GPIO 端口 F (AHB) 基址: 0x4005.D000
偏移量 0x418
类型 RO, 复位 0x0000.0000



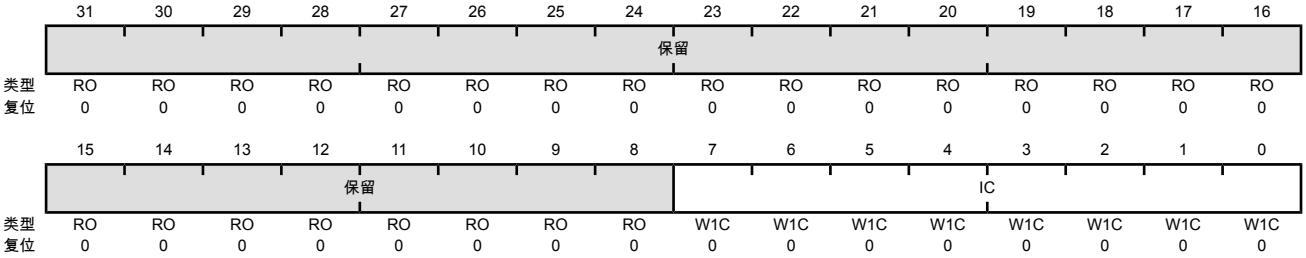
位/域	名称	类型	复位	描述
31:8	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
7:0	MIS	RO	0x00	GPIO屏蔽后的中断状态 值 描述 0 相应的管脚上未产生中断或中断已经被屏蔽。 1 相应的管教上已经有中断产生。 对于边沿触发中断，可通过向 GPIOICR 寄存器的相应位写 1 来将此位清零。 对于 GPIO 电平触发中断，此位在电平无效时清零。

寄存器 9: GPIO 中断清除寄存器（GPIOICR），偏移量 0x41C

GPIOICR 是中断清除寄存器。对于边沿触发中断，向 GPIOICR 寄存器的 IC 位写 1 即可将 GPIORIS 和 GPIOMIS 寄存器中相应的位清零。如果是电平触发中断，该寄存器中的 IC 位没有效果。此外，向 GPIOICR 寄存器的任何位写 0 均没有效果。

GPIO 中断清除寄存器 (GPIOICR)

GPIO 端口 A (APB) 基址: 0x4000.4000
GPIO 端口 A (AHB) 基址: 0x4005.8000
GPIO 端口 B (APB) 基址: 0x4000.5000
GPIO 端口 B (AHB) 基址: 0x4005.9000
GPIO 端口 C (APB) 基址: 0x4000.6000
GPIO 端口 C (AHB) 基址: 0x4005.A000
GPIO 端口 D (APB) 基址: 0x4000.7000
GPIO 端口 D (AHB) 基址: 0x4005.B000
GPIO 端口 E (APB) 基址: 0x4002.4000
GPIO 端口 E (AHB) 基址: 0x4005.C000
GPIO 端口 F (APB) 基址: 0x4002.5000
GPIO 端口 F (AHB) 基址: 0x4005.D000
偏移量 0x41C
类型 W1C, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:8	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
7:0	IC	W1C	0x00	GPIO中断清除 值 描述 0 没有任何效果。 1 清除GPIORIS和GPIOMIS寄存器里相应的位。

寄存器 10: GPIO 备用功能选择寄存器（GPIOAFSEL），偏移量 0x420

GPIOAFSEL 寄存器是模式控制选择寄存器。如果某位清零则表示该位被GPIO寄存器用作GPIO工能。置位则表示相关的管脚被用作相应的外设功能。每个 GPIO 上都有几个不同的复用外设功能。可以通过 GPIO 端口控制 (GPIOPCTL) 寄存器来选择其中的一个可能的功能。表20-5（1024页）详细列出了每个 GPIO 管脚上的复用功能。复位时，下表中没有列出的端口对应的该寄存器的值为 0x0000.0000。

重要: 所有的 GPIO 管脚在复位时都被配置为 GPIO 功能，而且是三态的，即 (GPIOAFSEL=0、GPIODEN=0、GPIOPDR=0、GPIOPUR=0、GPIOPCTL=0)，但是下表中列出的这些管脚除外。上电复位 (POR) 或确认 RST 都会将管脚恢复其默认设置。

表 **10-7.** 具有非 **0** 复位值的 **GPIO** 管脚

引脚	默认值	GPIOAFSEL	GPIODEN	GPIOPDR	GPIOPUR	GPIOPCTL
PA[1:0]	UART0	0	0	0	0	0x1
PA[5:2]	SSI0	0	0	0	0	0x2
PB[3:2]	I^2C0	0	0	0	0	0x3
PA[3:0]	JTAG/SWD	1	1	0	1	0x1

GPIO 提交控制寄存器提供了保护层，以防止对重要硬件信号的意外编程，其中包括 JTAG/SWD 信号和 NMI 信号。就算是不配置成 JTAG/SWD 或 NMI 信号，而是把他们配置成备用功能，这些管脚也必须遵循提交控制过程；请参考 “确认控制”（581页）。

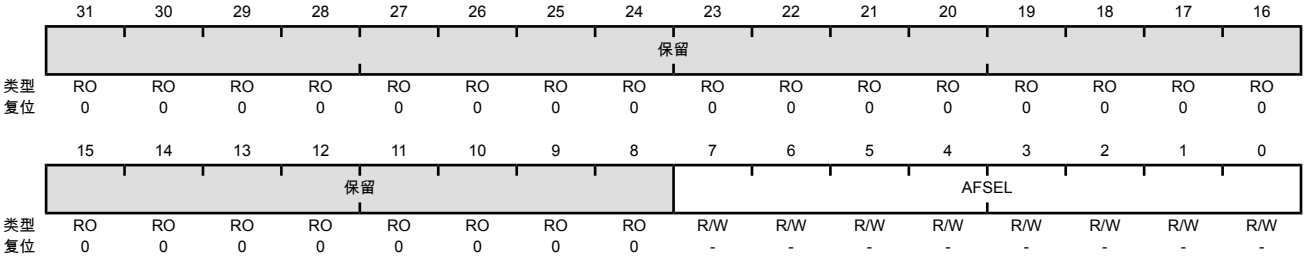
小心 – 用户可以建立一个软件序列来阻止调试器连接到 **TM4C1231H6PM** 微控制器。如果将程序代码加载到 **Flash** 中会立即将 **JTAG** 管脚变成其 **GPIO** 功能，那么在 **JTAG** 管脚功能切换之前，调试器将没有足够的时间去连接和终止控制器。结果调试器可能被锁定在该部分外。通过使用一个基于外部或软件的触发器来恢复**JTAG**功能的软件程序可以避免这个问题。如果未实施软件例程，且器件锁定在此部分以外，则可通过 **TM4C1231H6PM Flash** 编程器的**“解锁”**功能解决此问题。有关详细信息，请参阅 **TI** 网站的 **LMFLASHPROGRAMMER** 部分。

GPIO 提交控制寄存器提供了保护层以防止对重要硬件外设的意外编程。系统针对可用作四个 JTAG/SWD 管脚以及 NMI 管脚的 GPIO 管脚提供了保护功能（管脚号请参见“信号表”（1008页））。向 GPIO 备用功能选择 (GPIOAFSEL) 寄存器（请参阅595页）、GPIO 上拉电阻选择 (GPIOPUR) 寄存器（请参阅601页）、GPIO 下拉电阻选择 (GPIOPDR) 寄存器（请参阅603页）以及 GPIO 数字使能 (GPIODEN) 寄存器（请参阅606页）中受保护的位写入数据将不会确认保存，除非 GPIO 锁定 (GPIOLOCK) 寄存器（请参阅608页）没有被锁定，同时 GPIO 确认 (GPIOCR) 寄存器（请参阅609页）中相应的位被置位。

当使用 I^2C 模块时，除了要设置 GPIOAFSEL 寄存器中的 I^2C 时钟和数据管脚之外，还必须要用 GPIO 开漏选择 (GPIOODR) 寄存器将数据管脚设置成开漏（请参阅“初始化和配置”（582页）中的例子）。

GPIO 备用功能选择寄存器 (GPIOAFSEL)

GPIO 端口 A (APB) 基址: 0x4000.4000
GPIO 端口 A (AHB) 基址: 0x4005.8000
GPIO 端口 B (APB) 基址: 0x4000.5000
GPIO 端口 B (AHB) 基址: 0x4005.9000
GPIO 端口 C (APB) 基址: 0x4000.6000
GPIO 端口 C (AHB) 基址: 0x4005.A000
GPIO 端口 D (APB) 基址: 0x4000.7000
GPIO 端口 D (AHB) 基址: 0x4005.B000
GPIO 端口 E (APB) 基址: 0x4002.4000
GPIO 端口 E (AHB) 基址: 0x4005.C000
GPIO 端口 F (APB) 基址: 0x4002.5000
GPIO 端口 F (AHB) 基址: 0x4005.D000
偏移量 0x420
类型 R/W, 复位 -



位/域	名称	类型	复位	描述
31:8	保留	RO	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
7:0	AFSEL	R/W	-	GPIO备用功能选择

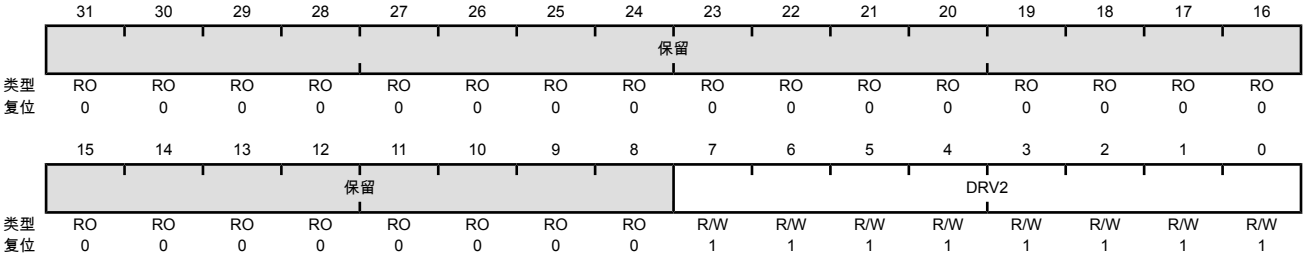
值	描述
0	相应的管脚为受GPIO寄存器控制的GPIO功能。
1	相应的管脚为相应的外设功能。 对于未在表10-1（576页） 中列出的 GPIO 端口，该寄存器的复位值为 0x0000.0000。

寄存器 11: GPIO 2-mA 驱动选择寄存器（GPIODR2R），偏移量 0x500

GPIODR2R 是 2-mA 驱动控制寄存器。每个引脚都可以单独配置，而不会影响其他的引脚。当 GPIO 信号的 DRV2 位置位时，GPIODR4R 寄存器里的 DRV4 和 GPIODR8R 寄存器里的 DRV8 被硬件自动清零。默认所有的引脚都是2-mA的驱动。

GPIO 2-mA 驱动选择寄存器 (GPIODR2R)

GPIO 端口 A (APB) 基址: 0x4000.4000
GPIO 端口 A (AHB) 基址: 0x4005.8000
GPIO 端口 B (APB) 基址: 0x4000.5000
GPIO 端口 B (AHB) 基址: 0x4005.9000
GPIO 端口 C (APB) 基址: 0x4000.6000
GPIO 端口 C (AHB) 基址: 0x4005.A000
GPIO 端口 D (APB) 基址: 0x4000.7000
GPIO 端口 D (AHB) 基址: 0x4005.B000
GPIO 端口 E (APB) 基址: 0x4002.4000
GPIO 端口 E (AHB) 基址: 0x4005.C000
GPIO 端口 F (APB) 基址: 0x4002.5000
GPIO 端口 F (AHB) 基址: 0x4005.D000
偏移量 0x500
类型 R/W, 复位 0x0000.00FF



位/域	名称	类型	复位	描述
31:8	保留	RO	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
7:0	DRV2	R/W	0xFF	输出2mA电流驱动使能

值 描述

0 相应 GPIO 管脚的驱动电流由 GPIODR4R 或 GPIODR8R 寄存器控制。

1 相应的管脚电流为2mA驱动。

将 GPIODR4 寄存器或 GPIODR8 寄存器中的某位置位，都会将相应的 2-mA 启用位清零。如果通过 APB 存储器槽访问 GPIO，那么在写入后的第二个时钟周期，这种改变生效。如果使用 AHB 访问，这种改变在下一个时钟周期生效。

寄存器 12: GPIO 4-mA 驱动选择寄存器（GPIODR4R），偏移量 0x504

GPIODR4R 是 4-mA 驱动控制寄存器。每个引脚都可以单独配置，而不会影响其他的引脚。当 GPIO 信号的 DRV4 置位时，GPIODR2R 寄存器里的 DRV2 和 GPIODR8R 寄存器里的 DRV8 被硬件自动清零。

GPIO 4-mA 驱动选择寄存器 (GPIODR4R)

GPIO 端口 A (APB) 基址: 0x4000.4000
GPIO 端口 A (AHB) 基址: 0x4005.8000
GPIO 端口 B (APB) 基址: 0x4000.5000
GPIO 端口 B (AHB) 基址: 0x4005.9000
GPIO 端口 C (APB) 基址: 0x4000.6000
GPIO 端口 C (AHB) 基址: 0x4005.A000
GPIO 端口 D (APB) 基址: 0x4000.7000
GPIO 端口 D (AHB) 基址: 0x4005.B000
GPIO 端口 E (APB) 基址: 0x4002.4000
GPIO 端口 E (AHB) 基址: 0x4005.C000
GPIO 端口 F (APB) 基址: 0x4002.5000
GPIO 端口 F (AHB) 基址: 0x4005.D000
偏移量 0x504
类型 R/W, 复位 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	保留															
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	保留								DRV4							
类型	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位/域	名称	类型	复位	描述
31:8	保留	RO	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
7:0	DRV4	R/W	0x00	输出4mA电流驱动使能

值 描述

0 相应的管脚驱动由 GPIODR2R 或 GPIODR8R 寄存器控制。

1 相应的管脚电流为4mA驱动。

将 GPIODR2 寄存器或 GPIODR8 寄存器中的某位置位，都会将相应的 4-mA 启用位清零。如果通过 APB 存储器槽访问 GPIO，那么在写入后的第二个时钟周期，这种改变生效。如果使用 AHB 访问，这种改变在下一个时钟周期生效。

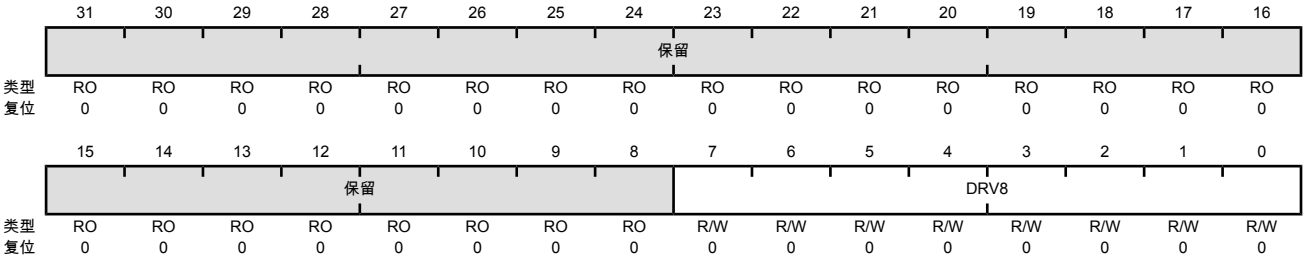
寄存器 13: GPIO 8-mA 驱动选择寄存器（GPIODR8R），偏移量 0x508

GPIODR8R 是 8-mA 驱动控制寄存器。每个引脚都可以单独配置，而不会影响其他的引脚。当 GPIO 信号的 DRV8 置位时，GPIODR2R 寄存器里的 DRV2 和 GPIODR4R 寄存器里的 DRV4 被硬件自动清零。8-mA驱动还可被用在高电流驱动的应用上。

注意: 8-mA 驱动和大电流驱动在配置上没有区别。额外的电流来自 V_{OH}/V_{OL} 电平转换。更多信息请参考 “Recommended Operating Conditions”（1031页）。

GPIO 8-mA 驱动选择寄存器 (GPIODR8R)

GPIO 端口 A (APB) 基址: 0x4000.4000
GPIO 端口 A (AHB) 基址: 0x4005.8000
GPIO 端口 B (APB) 基址: 0x4000.5000
GPIO 端口 B (AHB) 基址: 0x4005.9000
GPIO 端口 C (APB) 基址: 0x4000.6000
GPIO 端口 C (AHB) 基址: 0x4005.A000
GPIO 端口 D (APB) 基址: 0x4000.7000
GPIO 端口 D (AHB) 基址: 0x4005.B000
GPIO 端口 E (APB) 基址: 0x4002.4000
GPIO 端口 E (AHB) 基址: 0x4005.C000
GPIO 端口 F (APB) 基址: 0x4002.5000
GPIO 端口 F (AHB) 基址: 0x4005.D000
偏移量 0x508
类型 R/W, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:8	保留	RO	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
7:0	DRV8	R/W	0x00	输出8mA电流驱动使能

值 描述

0 相应的 GPIO 管脚驱动由 GPIODR2R 或 GPIODR4R 寄存器控制。

1 相应的管脚电流为8mA驱动。

将 GPIODR2 寄存器或 GPIODR4 寄存器中的某位置位，都会将相应的 8-mA 启用位清零。如果通过 APB 存储器槽访问 GPIO，那么在写入后的第二个时钟周期，这种改变生效。如果使用 AHB 访问，这种改变在下一个时钟周期生效。

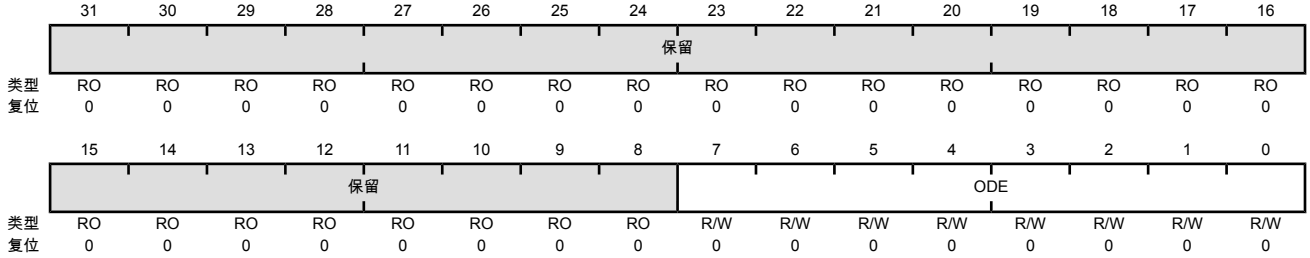
寄存器 14: GPIO 开漏选择寄存器（GPIOODR），偏移量 0x50C

GPIOODR 是开漏控制寄存器。置位则使能相应管脚的开漏功能。当管脚的开漏功能启用的时候，GPIO 数字使能 (GPIODEN) 寄存器（请参考 606页）中相应的位也必须置位。为了达到所需的下降时间，需要将驱动强度和斜率控制寄存器（GPIODR2R、GPIODR4R、GPIODR8R 和 GPIOSLR）中的相应位置位。如果 GPIODIR 寄存器的某位被清零，那么相应的 GPIO 是输入。如果 GPIO 配置成输入，而同时又选择了开漏，那么该 GPIO 还是输入，开漏选择不会生效，GPIO 变成输出以后才开漏才有效。

当使用 I^2C 模块时，除了将管脚配置成开漏功能，GPIO 备用功能选择 (GPIOAFSEL) 寄存器中的 I^2C 时钟和数据管脚位必须置位（请参阅“初始化和配置”（582页） 中的示例）。

GPIO 开漏选择寄存器 (GPIOODR)

GPIO 端口 A (APB) 基址: 0x4000.4000
GPIO 端口 A (AHB) 基址: 0x4005.8000
GPIO 端口 B (APB) 基址: 0x4000.5000
GPIO 端口 B (AHB) 基址: 0x4005.9000
GPIO 端口 C (APB) 基址: 0x4000.6000
GPIO 端口 C (AHB) 基址: 0x4005.A000
GPIO 端口 D (APB) 基址: 0x4000.7000
GPIO 端口 D (AHB) 基址: 0x4005.B000
GPIO 端口 E (APB) 基址: 0x4002.4000
GPIO 端口 E (AHB) 基址: 0x4005.C000
GPIO 端口 F (APB) 基址: 0x4002.5000
GPIO 端口 F (AHB) 基址: 0x4005.D000
偏移量 0x50C
类型 R/W, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:8	保留	RO	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
7:0	ODE	R/W	0x00	GPIO开漏选择 值 描述 0 不使能相应管脚的开漏功能。 1 使能相应管脚的开漏功能。

寄存器 15: GPIO 上拉电阻选择寄存器（GPIOPUR），偏移量 0x510

GPIOPUR 寄存器是上拉控制寄存器。当置位时，相应的管脚使能弱上拉电阻。将 GPIOPUR 中的位置位会自动将 GPIO 下拉电阻选择 (GPIOPDR) 寄存器（请参考 603页）中相应的位清零。该寄存器的写入操作受到 GPIOCR 寄存器的保护。GPIOCR 寄存器中某位被清零时可以防止写入该寄存器中相应的位。

重要: 所有的 GPIO 管脚在复位时都被配置为 GPIO 功能，而且是三态的，即 (GPIOAFSEL=0、GPIODEN=0、GPIOPDR=0、GPIOPUR=0、GPIOPCTL=0)，但是下表中列出的这些管脚除外。上电复位 (POR) 或确认 RST 都会将管脚恢复其默认设置。

表 **10-8.** 具有非 **0** 复位值的 **GPIO** 管脚

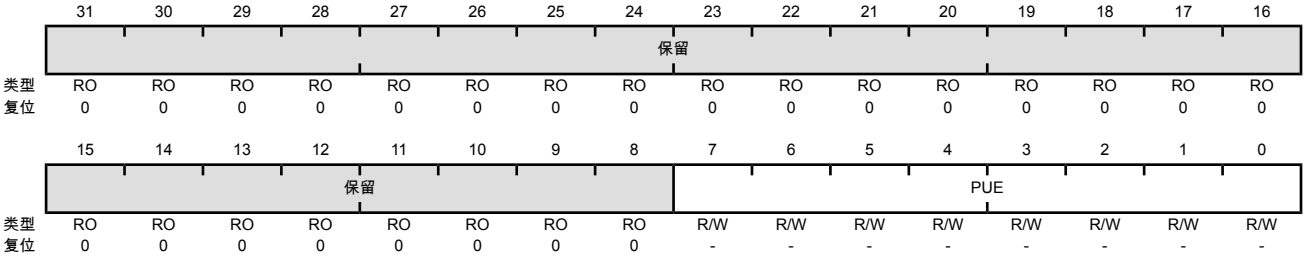
引脚	默认值	GPIOAFSEL	GPIODEN	GPIOPDR	GPIOPUR	GPIOPCTL
PA[1:0]	UART0	0	0	0	0	0x1
PA[5:2]	SSI0	0	0	0	0	0x2
PB[3:2]	I^2C0	0	0	0	0	0x3
PA[3:0]	JTAG/SWD	1	1	0	1	0x1

GPIO 提交控制寄存器提供了保护层，以防止对重要硬件信号的意外编程，其中包括 JTAG/SWD 信号和 NMI 信号。就算是不配置成 JTAG/SWD 或 NMI 信号，而是把他们配置成备用功能，这些管脚也必须遵循提交控制过程；请参考 “确认控制”（581页）。

注意: GPIO 提交控制寄存器提供了保护层以防止对重要硬件外设的意外编程。系统针对可用作四个 JTAG/SWD 管脚以及 NMI 管脚的 GPIO 管脚提供了保护功能（管脚号请参见“信号表”（1008页））。向 GPIO 备用功能选择 (GPIOAFSEL) 寄存器（请参阅595页）、GPIO 上拉电阻选择 (GPIOPUR) 寄存器（请参阅601页）、GPIO 下拉电阻选择 (GPIOPDR) 寄存器（请参阅603页）以及 GPIO 数字使能 (GPIODEN) 寄存器（请参阅606页）中受保护的位写入数据将不会确认保存，除非 GPIO 锁定 (GPIOLOCK) 寄存器（请参阅608页）没有被锁定，同时 GPIO 确认 (GPIOCR) 寄存器（请参阅609页）中相应的位被置位。

GPIO 上拉电阻选择寄存器 (GPIOPUR)

GPIO 端口 A (APB) 基址: 0x4000.4000
GPIO 端口 A (AHB) 基址: 0x4005.8000
GPIO 端口 B (APB) 基址: 0x4000.5000
GPIO 端口 B (AHB) 基址: 0x4005.9000
GPIO 端口 C (APB) 基址: 0x4000.6000
GPIO 端口 C (AHB) 基址: 0x4005.A000
GPIO 端口 D (APB) 基址: 0x4000.7000
GPIO 端口 D (AHB) 基址: 0x4005.B000
GPIO 端口 E (APB) 基址: 0x4002.4000
GPIO 端口 E (AHB) 基址: 0x4005.C000
GPIO 端口 F (APB) 基址: 0x4002.5000
GPIO 端口 F (AHB) 基址: 0x4005.D000
偏移量 0x510
类型 R/W, 复位 -



位/域	名称	类型	复位	描述
31:8	保留	RO	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
7:0	PUE	R/W	-	GPIO上拉电阻选择 值 描述 0 相应管脚的弱上拉电阻被禁用。 1 相应管脚的弱上拉电阻被启用。 将 GPIOPDR 寄存器中的位置位会将 GPIOPUR 寄存器中相应的位清零。如果通过 APB 存储器槽访问 GPIO，那么在写入后的第二个时钟周期，这种改变生效。如果使用 AHB 访问，这种改变在下一个时钟周期生效。 对于未在表10-1（576页） 中列出的 GPIO 端口，该寄存器的复位值为 0x0000.0000。

寄存器 16: GPIO 下拉电阻选择寄存器（GPIOPDR），偏移量 0x514

GPIOPDR 是下拉电阻控制寄存器。置位时相应的管脚下拉电阻启用。置位 GPIOPDR 中的位会使 GPIO上拉电阻选择 (GPIOPUR) 寄存器（见 601页）中相应的位自动清零

重要: 所有的GPIO管脚在复位时都被配置为GPIO功能，而且是三态的，即(GPIOAFSEL=0、GPIODEN=0、GPIOPDR=0、GPIOPUR=0、GPIOPCTL=0)，但是下表中列出的这些管脚除外。上电复位 (POR) 或确认 RST 都会将管脚恢复其默认设置。

表 **10-9.** 具有非 **0** 复位值的 **GPIO** 管脚

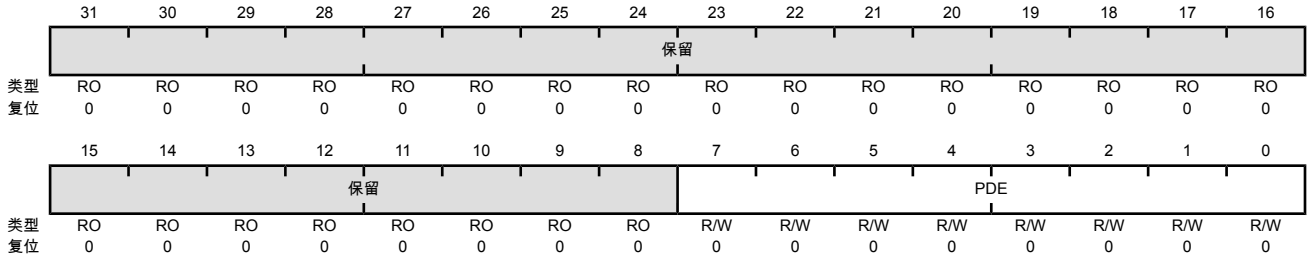
引脚	默认值	GPIOAFSEL	GPIODEN	GPIOPDR	GPIOPUR	GPIOPCTL
PA[1:0]	UART0	0	0	0	0	0x1
PA[5:2]	SSI0	0	0	0	0	0x2
PB[3:2]	I^2C0	0	0	0	0	0x3
PA[3:0]	JTAG/SWD	1	1	0	1	0x1

GPIO 提交控制寄存器提供了保护层，以防止对重要硬件信号的意外编程，其中包括 JTAG/SWD 信号和 NMI 信号。就算是不配置成 JTAG/SWD 或 NMI 信号，而是把他们配置成备用功能，这些管脚也必须遵循提交控制过程；请参考 “确认控制”（581页）。

注意: GPIO提交控制寄存器提供了保护层以防止对重要硬件外设的意外编程。系统针对可用作四个 JTAG/SWD 管脚以及 NMI 管脚的 GPIO 管脚提供了保护功能（管脚号请参见“信号表”（1008页））。向GPIO备用功能选择 (GPIOAFSEL) 寄存器（请参阅595页）、GPIO 上拉电阻选择 (GPIOPUR) 寄存器（请参阅601页）、GPIO 下拉电阻选择 (GPIOPDR) 寄存器（请参阅603页）以及GPIO数字使能 (GPIODEN) 寄存器（请参阅606页）中受保护的位写入数据将不会确认保存，除非 GPIO 锁定 (GPIOLOCK) 寄存器（请参阅608页）没有被锁定，同时 GPIO 确认 (GPIOCR) 寄存器（请参阅609页）中相应的位被置位。

GPIO 下拉电阻选择寄存器 (GPIOPDR)

GPIO 端口 A (APB) 基址: 0x4000.4000
GPIO 端口 A (AHB) 基址: 0x4005.8000
GPIO 端口 B (APB) 基址: 0x4000.5000
GPIO 端口 B (AHB) 基址: 0x4005.9000
GPIO 端口 C (APB) 基址: 0x4000.6000
GPIO 端口 C (AHB) 基址: 0x4005.A000
GPIO 端口 D (APB) 基址: 0x4000.7000
GPIO 端口 D (AHB) 基址: 0x4005.B000
GPIO 端口 E (APB) 基址: 0x4002.4000
GPIO 端口 E (AHB) 基址: 0x4005.C000
GPIO 端口 F (APB) 基址: 0x4002.5000
GPIO 端口 F (AHB) 基址: 0x4005.D000
偏移量 0x514
类型 R/W, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:8	保留	RO	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。

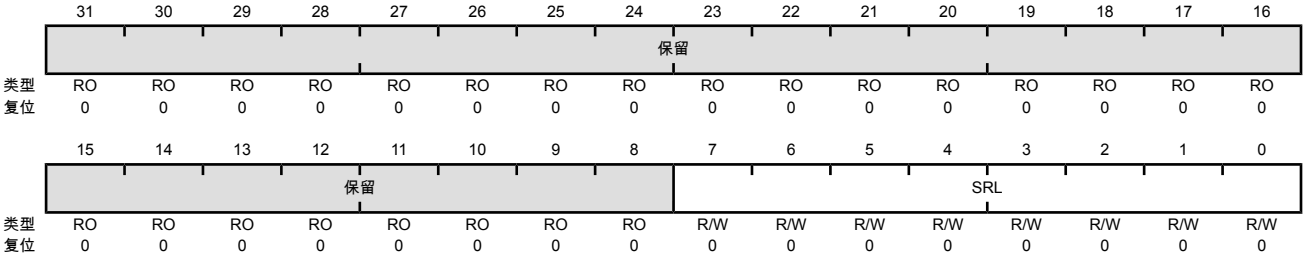
位/域	名称	类型	复位	描述
7:0	PDE	R/W	0x00	GPIO下拉电阻选择 值 描述 0 相应管脚的弱下拉电阻被禁用。 1 相应管脚的弱下拉电阻被启用。 将 GPIOPUR 寄存器中的位置位会将 GPIOPDR 寄存器中相应的位清零。如果通过 APB 存储器槽访问 GPIO，那么在写入后的第二个时钟周期，这种改变生效。如果使用 AHB 访问，这种改变在下一个时钟周期生效。

寄存器 17: GPIO 斜率控制选择寄存器（GPIOSLR），偏移量 0x518

GPIOSLR 是斜率控制寄存器。仅当使用 8-mA 驱动电流时才可使用斜率控制。可通过 GPIO 8-mA 驱动选择 (GPIODR8R) 寄存器选择驱动电流的强度。

GPIO 斜率控制选择寄存器 (GPIOSLR)

GPIO 端口 A (APB) 基址: 0x4000.4000
GPIO 端口 A (AHB) 基址: 0x4005.8000
GPIO 端口 B (APB) 基址: 0x4000.5000
GPIO 端口 B (AHB) 基址: 0x4005.9000
GPIO 端口 C (APB) 基址: 0x4000.6000
GPIO 端口 C (AHB) 基址: 0x4005.A000
GPIO 端口 D (APB) 基址: 0x4000.7000
GPIO 端口 D (AHB) 基址: 0x4005.B000
GPIO 端口 E (APB) 基址: 0x4002.4000
GPIO 端口 E (AHB) 基址: 0x4005.C000
GPIO 端口 F (APB) 基址: 0x4002.5000
GPIO 端口 F (AHB) 基址: 0x4005.D000
偏移量 0x518
类型 R/W, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:8	保留	RO	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
7:0	SRL	R/W	0x00	斜率限制启用（8-mA 驱动） 值 描述 0 不使能管脚的斜率控制 1 使能管脚的斜率控制

寄存器 18: GPIO 数字使能寄存器（GPIODEN），偏移量 0x51C

注意: 配置为数字输入的管脚均为施密特触发

GPIODEN 是数字启用寄存器。默认情况下，除了下面列出的引脚外，所有的引脚复位时都被配置为非驱动(三态)的。它们的数字功能是禁止的。它们不去动管脚上的逻辑值，而且也不允许管脚上的电压进入GPIO接收器。为了将该管脚用作数字输入或输出（GPIO 或备用功能），必须将相应的 GPIODEN 位置位。

重要: 所有的GPIO管脚在复位时都被配置为GPIO功能，而且是三态的，即(GPIOAFSEL=0、GPIODEN=0、GPIOPDR=0、GPIOPUR=0、GPIOPCTL=0)，但是下表中列出的这些管脚除外。上电复位 ($\overline{POR}$) 或确认 $\overline{RST}$ 都会将管脚恢复其默认设置。

表 **10-10.** 具有非 **0** 复位值的 **GPIO** 管脚

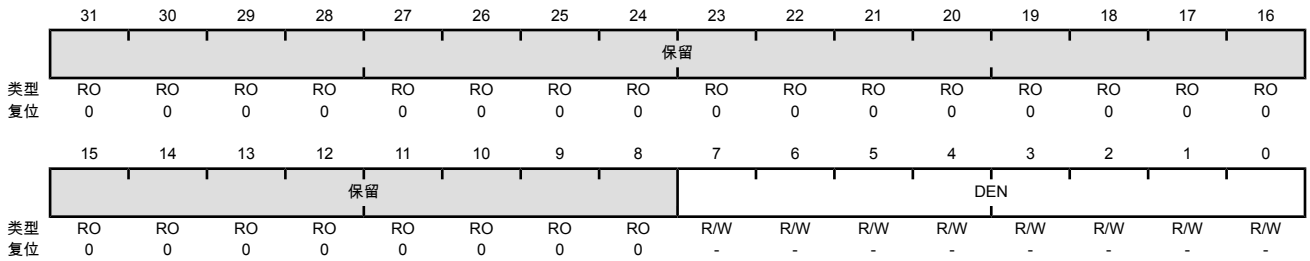
引脚	默认值	GPIOAFSEL	GPIODEN	GPIOPDR	GPIOPUR	GPIOPCTL
PA[1:0]	UART0	0	0	0	0	0x1
PA[5:2]	SSI0	0	0	0	0	0x2
PB[3:2]	I^2C0	0	0	0	0	0x3
PA[3:0]	JTAG/SWD	1	1	0	1	0x1

GPIO 提交控制寄存器提供了保护层，以防止对重要硬件信号的意外编程，其中包括 JTAG/SWD 信号和 NMI 信号。就算是不配置成 JTAG/SWD 或 NMI 信号，而是把他们配置成备用功能，这些管脚也必须遵循提交控制过程；请参考 “确认控制”（581页）。

注意: GPIO提交控制寄存器提供了保护层以防止对重要硬件外设的意外编程。系统针对可用作四个 JTAG/SWD 管脚以及 NMI 管脚的 GPIO 管脚提供了保护功能（管脚号请参见“信号表”（1008页））。向GPIO备用功能选择 (GPIOAFSEL) 寄存器（请参阅595页）、GPIO 上拉电阻选择 (GPIOPUR) 寄存器（请参阅601页）、GPIO 下拉电阻选择 (GPIOPDR) 寄存器（请参阅603页）以及GPIO数字使能 (GPIODEN) 寄存器（请参阅606页）中受保护的位写入数据将不会确认保存，除非 GPIO 锁定 (GPIOLOCK) 寄存器（请参阅608页）没有被锁定，同时 GPIO 确认 (GPIOCR) 寄存器（请参阅609页）中相应的位被置位。

GPIO 数字使能寄存器 (GPIODEN)

GPIO 端口 A (APB) 基址: 0x4000.4000
GPIO 端口 A (AHB) 基址: 0x4005.8000
GPIO 端口 B (APB) 基址: 0x4000.5000
GPIO 端口 B (AHB) 基址: 0x4005.9000
GPIO 端口 C (APB) 基址: 0x4000.6000
GPIO 端口 C (AHB) 基址: 0x4005.A000
GPIO 端口 D (APB) 基址: 0x4000.7000
GPIO 端口 D (AHB) 基址: 0x4005.B000
GPIO 端口 E (APB) 基址: 0x4002.4000
GPIO 端口 E (AHB) 基址: 0x4005.C000
GPIO 端口 F (APB) 基址: 0x4002.5000
GPIO 端口 F (AHB) 基址: 0x4005.D000
偏移量 0x51C
类型 R/W, 复位 -



位/域	名称	类型	复位	描述
31:8	保留	RO	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
7:0	DEN	R/W	-	数字启用 值 描述 0 不使能管脚的数字功能。 1 启用管脚的数字功能。 对于未在表10-1（576页） 中列出的 GPIO 端口，该寄存器的复位值为 0x0000.0000。

寄存器 19: GPIO 锁定寄存器（GPIOLOCK），偏移量 0x520

GPIOLOCK 寄存器可以启用对 GPIOCR 寄存器（请参考 609页）的写入访问。写入 0x4C4F.434B 到 GPIOLOCK 寄存器可解锁 GPIOCR 寄存器。写任何其他的数值到 GPIOLOCK 寄存器可以恢复锁定状态。读取 GPIOLOCK 寄存器将返回锁定状态，而不是先前写入的 32 位值。因此，当写入访问被禁用或锁定时，读取 GPIOLOCK 寄存器返回 0x00000001。当写入访问被启用或解锁时，读取 GPIOLOCK 寄存器返回 0x00000000。

GPIO 锁定寄存器 (GPIOLOCK)

GPIO 端口 A (APB) 基址: 0x4000.4000
GPIO 端口 A (AHB) 基址: 0x4005.8000
GPIO 端口 B (APB) 基址: 0x4000.5000
GPIO 端口 B (AHB) 基址: 0x4005.9000
GPIO 端口 C (APB) 基址: 0x4000.6000
GPIO 端口 C (AHB) 基址: 0x4005.A000
GPIO 端口 D (APB) 基址: 0x4000.7000
GPIO 端口 D (AHB) 基址: 0x4005.B000
GPIO 端口 E (APB) 基址: 0x4002.4000
GPIO 端口 E (AHB) 基址: 0x4005.C000
GPIO 端口 F (APB) 基址: 0x4002.5000
GPIO 端口 F (AHB) 基址: 0x4005.D000
偏移量 0x520
类型 R/W, 复位 0x0000.0001

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	LOCK															
类型	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	LOCK															
类型	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

位/域	名称	类型	复位	描述
31:0	LOCK	R/W	0x0000.0001	GPIO锁定 写入 0x4C4F.434B 可解锁 GPIO 确认 (GPIOCR) 寄存器，使之允许写入；写入其他数值或对 GPIOCR 寄存器写操作将恢复其锁定状态，以防止更新此寄存器。 读取该寄存器返回值具有如下的含义 值 描述 0x1 GPIOCR 寄存器处于锁定状态，不能被修改。 0x0 GPIOCR 寄存器处于解锁状态，可以被修改。

寄存器 20: GPIO 确认寄存器（GPIOCR），偏移量 0x524

GPIOCR 是确认寄存器。当写入 GPIOAFSEL、GPIOPUR、GPIOPDR 和 GPIODEN 寄存器时，GPIOCR 寄存器的值决定了这些寄存器中哪些位将被确认。如果 GPIOCR 寄存器中的某个位清零，那么写入 GPIOAFSEL、GPIOPUR、GPIOPDR 或 GPIODEN 寄存器中相应位的数据将不被确认，其原来的值将被保留。如果 GPIOCR 寄存器中的某个位置位，那么写入 GPIOAFSEL、GPIOPUR、GPIOPDR 或 GPIODEN 寄存器中相应位的数据将被确认，寄存器保存新的值。

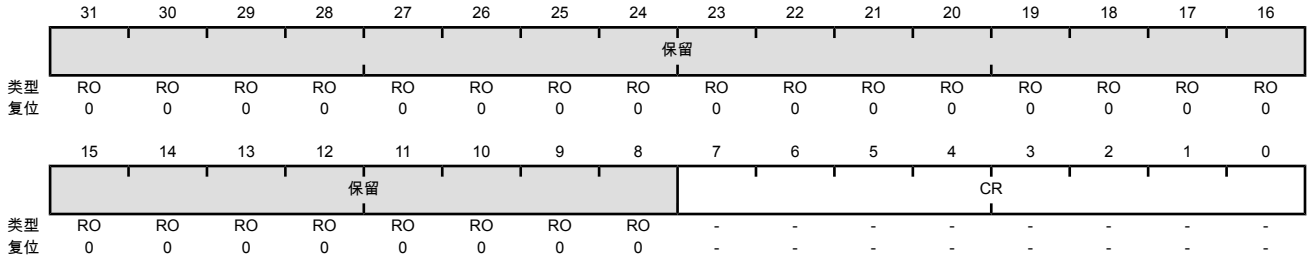
GPIOCR 寄存器的内容只有在 GPIOLOCK 寄存器解锁时才能被修改。如果 GPIOLOCK 寄存器被锁定，那么写 GPIOCR 寄存器的操作将被忽略。

重要: 该寄存器用于防止对那些控制 NMI 和 JTAG/SWD 调试硬件连接的寄存器进行意外编程。通过将 GPIOCR 寄存器中针对 NMI 和 JTAG/SWD 管脚的位初始化为 0（管脚号请参见“信号表”（1008页）），NMI 和 JTAG/SWD 调试端口只能通过对 GPIOLOCK、GPIOCR 以及其他相应寄存器的一系列写操作来转换为 GPIO。

因为这种保护当前只在 NMI 和 JTAG/SWD 管脚上执行（管脚号请参见“信号表”（1008页）），所以 GPIOCR 寄存器中所有其它位都不能写入 0x0。这些位被硬连接为 0x1，以确保总是可以为其它管脚的 GPIOAFSEL、GPIOPUR、GPIOPDR 或 GPIODEN 寄存器位确认新值。

GPIO 确认寄存器 (GPIOCR)

GPIO 端口 A (APB) 基址: 0x4000.4000
GPIO 端口 A (AHB) 基址: 0x4005.8000
GPIO 端口 B (APB) 基址: 0x4000.5000
GPIO 端口 B (AHB) 基址: 0x4005.9000
GPIO 端口 C (APB) 基址: 0x4000.6000
GPIO 端口 C (AHB) 基址: 0x4005.A000
GPIO 端口 D (APB) 基址: 0x4000.7000
GPIO 端口 D (AHB) 基址: 0x4005.B000
GPIO 端口 E (APB) 基址: 0x4002.4000
GPIO 端口 E (AHB) 基址: 0x4005.C000
GPIO 端口 F (APB) 基址: 0x4002.5000
GPIO 端口 F (AHB) 基址: 0x4005.D000
偏移量 0x524
类型 -, 复位 -



位/域	名称	类型	复位	描述
31:8	保留	RO	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。

位/域	名称	类型	复位	描述
7:0	CR	-	-	GPIO确认

值	描述
0	相应的 GPIOAFSEL、GPIOPUR、GPIOPDR 或 GPIODEN 位不能写入。
1	相应的 GPIOAFSEL、GPIOPUR、GPIOPDR 或 GPIODEN 位可以写入。

注意: 除了 NMI 管脚和四个 JTAG/SWD 管脚之外，所有 GPIO 管脚的 GPIOCR 寄存器的默认类型是 RO（管脚号请参见“信号表”（1008页））。GPIOCR 寄存器当前仅保护这六个 GPIO 管脚。鉴于此，相应 GPIO 端口的寄存器类型为 R/W。

除了 NMI 管脚和四个 JTAG/SWD 管脚之外，所有 GPIO 管脚的 GPIOCR 寄存器的默认复位值是 0x0000.00FF（管脚号请参见“信号表”（1008页））。为了确保 JTAG 和 NMI 管脚不会被意外地编程为 GPIO 管脚，这些管脚默认是锁定的，以防止犯错。鉴于此，相应端口的 GPIOCR 默认复位值发生变化。

寄存器 21: GPIO 模拟选择寄存器（GPIOAMSEL），偏移量 0x528

重要: 该寄存器只能用于那些可以用作 ADC AINx 输入的端口和管脚。

如果有引脚被用作了 ADC 的输入管脚，那么 GPIOAMSEL 寄存器相应的位必须置位，以禁用模拟隔离电流。

GPIOAMSEL 寄存器控制着统一 I/O 管脚模拟侧的隔离电路。因为 GPIO 引脚可能会被 5 V 电源驱动或受模拟操作的影响，所以在不使用模拟电路的模拟功能时需要将之与管脚隔离。

该寄存器的每一位控制着相应引脚的隔离电路。要了解哪些 GPIO 管脚可以用作 ADC 功能，请参考 表20-5（1024页）。

GPIO 模拟选择寄存器 (GPIOAMSEL)

GPIO 端口 A (APB) 基址: 0x4000.4000
GPIO 端口 A (AHB) 基址: 0x4005.8000
GPIO 端口 B (APB) 基址: 0x4000.5000
GPIO 端口 B (AHB) 基址: 0x4005.9000
GPIO 端口 C (APB) 基址: 0x4000.6000
GPIO 端口 C (AHB) 基址: 0x4005.A000
GPIO 端口 D (APB) 基址: 0x4000.7000
GPIO 端口 D (AHB) 基址: 0x4005.B000
GPIO 端口 E (APB) 基址: 0x4002.4000
GPIO 端口 E (AHB) 基址: 0x4005.C000
GPIO 端口 F (APB) 基址: 0x4002.5000
GPIO 端口 F (AHB) 基址: 0x4005.D000
偏移量 0x528
类型 R/W, 复位 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	保留															
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	保留												GPIOAMSEL			
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位/域	名称	类型	复位	描述
31:4	保留	RO	0x0000.000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
3:0	GPIOAMSEL	R/W	0x0	GPIO模拟模块选择

值 描述

0 GPIO模拟功能使能，隔离不使能，引脚使用模拟功能。

1 GPIO模拟功能不使能，模拟电路被隔离，引脚为GPIO配置寄存器指定的功能。

注意: 该寄存器和位只适用于那些通过通用 I/O 口共享模拟功能的管脚。

该寄存器对所有引脚的复位状态都是0。

寄存器 22: GPIO端口控制寄存器（GPIOPCTL），偏移量0x52C

GPIOCTL 寄存器是 GPIO 端口控制寄存器。它和 GPIOAFSEL 一起决定了在备用功能模式下管脚使用那一路外设。GPIOAFSEL 寄存器大部分的位在复位时是清零的，所以 GPIO 管脚大部分被默认配置为 GPIO 功能。GPIOAFSEL 寄存器中某位置位时，表示相应的 GPIO 管脚由相关外设控制。GPIOPCTL 寄存器可以为每个 GPIO 管脚选择使用哪个外设功能，因此在信号定义时提供了很高的灵活性。该寄存器中位域编码的更多信息，请参考 表20-5（1024页）。复位时，下表中没有列出的端口对应的该寄存器的值为0x0000.0000。

注意: 如果输入外设的特定信号被分配给两个不同的 GPIO 端口管脚，则该信号分配给编号更小的一个端口，而忽略大编号端口的信号分配。如果来自外设的特定输出信号被分配给两个不同的 GPIO 端口管脚，则此信号将同时输出至两个管脚。不建议将来自外设的输出信号分配给两个不同的 GPIO 管脚。

重要: 所有的 GPIO 管脚在复位时都被配置为 GPIO 功能，而且是三态的，即 (GPIOAFSEL=0、GPIODEN=0、GPIOPDR=0、GPIOPUR=0、GPIOPCTL=0)，但是下表中列出的这些管脚除外。上电复位 (POR) 或确认 RST 都会将管脚恢复其默认设置。

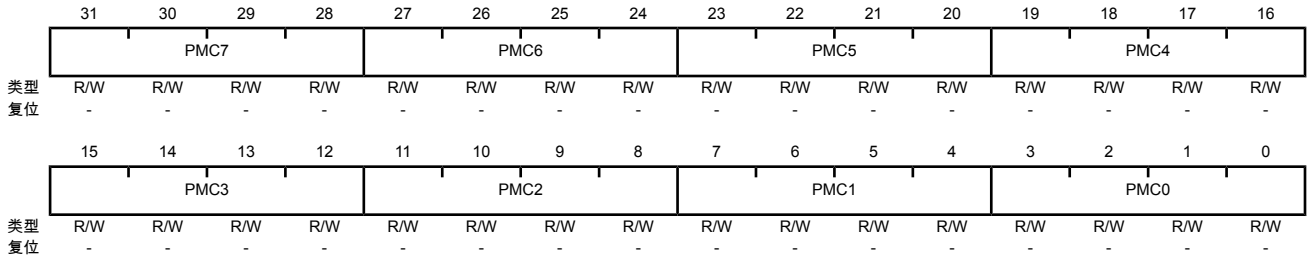
表 **10-11.** 具有非 **0** 复位值的 **GPIO** 管脚

引脚	默认值	GPIOAFSEL	GPIODEN	GPIOPDR	GPIOPUR	GPIOPCTL
PA[1:0]	UART0	0	0	0	0	0x1
PA[5:2]	SSI0	0	0	0	0	0x2
PB[3:2]	I^2C0	0	0	0	0	0x3
PA[3:0]	JTAG/SWD	1	1	0	1	0x1

GPIO 提交控制寄存器提供了保护层，以防止对重要硬件信号的意外编程，其中包括 JTAG/SWD 信号和 NMI 信号。就算是不配置成 JTAG/SWD 或 NMI 信号，而是把他们配置成备用功能，这些管脚也必须遵循提交控制过程；请参考 “确认控制”（581页）。

GPIO端口控制寄存器 (GPIOPCTL)

GPIO 端口 A (APB) 基址: 0x4000.4000
GPIO 端口 A (AHB) 基址: 0x4005.8000
GPIO 端口 B (APB) 基址: 0x4000.5000
GPIO 端口 B (AHB) 基址: 0x4005.9000
GPIO 端口 C (APB) 基址: 0x4000.6000
GPIO 端口 C (AHB) 基址: 0x4005.A000
GPIO 端口 D (APB) 基址: 0x4000.7000
GPIO 端口 D (AHB) 基址: 0x4005.B000
GPIO 端口 E (APB) 基址: 0x4002.4000
GPIO 端口 E (AHB) 基址: 0x4005.C000
GPIO 端口 F (APB) 基址: 0x4002.5000
GPIO 端口 F (AHB) 基址: 0x4005.D000
偏移量 0x52C
类型 R/W, 复位 -



位/域	名称	类型	复位	描述
31:28	PMC7	R/W	-	端口复用控制器7 该区域的值控制着端口7的配置
27:24	PMC6	R/W	-	端口复用控制器6 该区域的值控制着端口6的配置
23:20	PMC5	R/W	-	端口复用控制器5 该区域的值控制着端口5的配置
19:16	PMC4	R/W	-	端口复用控制器4 该区域的值控制着端口4的配置
15:12	PMC3	R/W	-	端口复用控制器3 该区域的值控制着端口3的配置
11:8	PMC2	R/W	-	端口复用控制器2 该区域的值控制着端口2的配置
7:4	PMC1	R/W	-	端口复用控制器1 该区域的值控制着端口1的配置
3:0	PMC0	R/W	-	端口复用控制器0 该区域的值控制着端口0的配置

寄存器 23: GPIO ADC 控制寄存器（GPIOADCCTL），偏移量 0x530

该寄存器用来将 GPIO 管脚配置成 ADC 的触发源。

请注意，如果 Port B GPIOADCCTL 寄存器被清零，PB4 也可以用作 ADC 的外部触发信号。此传统模式允许在此微控制器中运行针对上一代器件编写的代码。

GPIO ADC 控制寄存器 (GPIOADCCTL)

GPIO 端口 A (APB) 基址: 0x4000.4000
GPIO 端口 A (AHB) 基址: 0x4005.8000
GPIO 端口 B (APB) 基址: 0x4000.5000
GPIO 端口 B (AHB) 基址: 0x4005.9000
GPIO 端口 C (APB) 基址: 0x4000.6000
GPIO 端口 C (AHB) 基址: 0x4005.A000
GPIO 端口 D (APB) 基址: 0x4000.7000
GPIO 端口 D (AHB) 基址: 0x4005.B000
GPIO 端口 E (APB) 基址: 0x4002.4000
GPIO 端口 E (AHB) 基址: 0x4005.C000
GPIO 端口 F (APB) 基址: 0x4002.5000
GPIO 端口 F (AHB) 基址: 0x4005.D000
偏移量 0x530
类型 R/W, 复位 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	保留															
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	保留								ADCEN							
类型	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位/域	名称	类型	复位	描述
31:8	保留	RO	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
7:0	ADCEN	R/W	0x00	ADC 触发启用

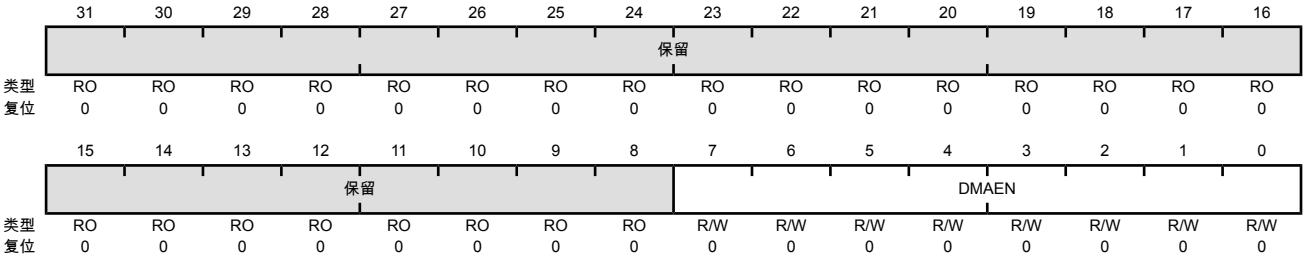
值	描述
0	相应管脚没有用来触发 ADC。
1	相应管脚用来触发 ADC。

寄存器 24: GPIO DMA 控制寄存器（GPIODMACTL），偏移量 0x534

该寄存器用来将 GPIO 管脚配置成 µDMA 的触发源。

GPIO DMA 控制寄存器 (GPIODMACTL)

GPIO 端口 A (APB) 基址: 0x4000.4000
GPIO 端口 A (AHB) 基址: 0x4005.8000
GPIO 端口 B (APB) 基址: 0x4000.5000
GPIO 端口 B (AHB) 基址: 0x4005.9000
GPIO 端口 C (APB) 基址: 0x4000.6000
GPIO 端口 C (AHB) 基址: 0x4005.A000
GPIO 端口 D (APB) 基址: 0x4000.7000
GPIO 端口 D (AHB) 基址: 0x4005.B000
GPIO 端口 E (APB) 基址: 0x4002.4000
GPIO 端口 E (AHB) 基址: 0x4005.C000
GPIO 端口 F (APB) 基址: 0x4002.5000
GPIO 端口 F (AHB) 基址: 0x4005.D000
偏移量 0x534
类型 R/W, 复位 0x0000.0000



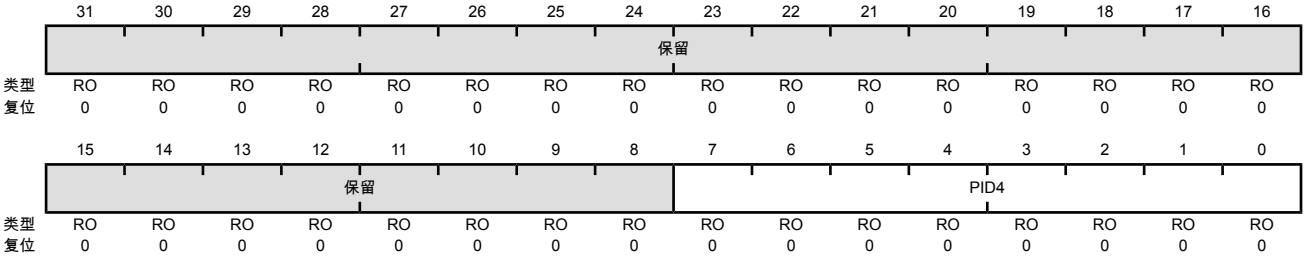
位/域	名称	类型	复位	描述
31:8	保留	RO	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
7:0	DMAEN	R/W	0x00	µDMA 触发启用 值 描述 0 相应管脚没有用来触发 µDMA。 1 相应管脚用来触发 µDMA。

寄存器 25: GPIO 外设标识寄存器 4（GPIOPeriphID4），偏移量 0xFD0

GPIOPeriphID4、GPIOPeriphID5、GPIOPeriphID6 和 GPIOPeriphID7 寄存器在概念上可以看作一个 32 位寄存器；每个寄存器包含 32 位寄存器的 8 个位，可被软件用来标识外设。

GPIO 外设标识寄存器 4 (GPIOPeriphID4)

GPIO 端口 A (APB) 基址: 0x4000.4000
GPIO 端口 A (AHB) 基址: 0x4005.8000
GPIO 端口 B (APB) 基址: 0x4000.5000
GPIO 端口 B (AHB) 基址: 0x4005.9000
GPIO 端口 C (APB) 基址: 0x4000.6000
GPIO 端口 C (AHB) 基址: 0x4005.A000
GPIO 端口 D (APB) 基址: 0x4000.7000
GPIO 端口 D (AHB) 基址: 0x4005.B000
GPIO 端口 E (APB) 基址: 0x4002.4000
GPIO 端口 E (AHB) 基址: 0x4005.C000
GPIO 端口 F (APB) 基址: 0x4002.5000
GPIO 端口 F (AHB) 基址: 0x4005.D000
偏移量 0xFD0
类型 RO, 复位 0x0000.0000



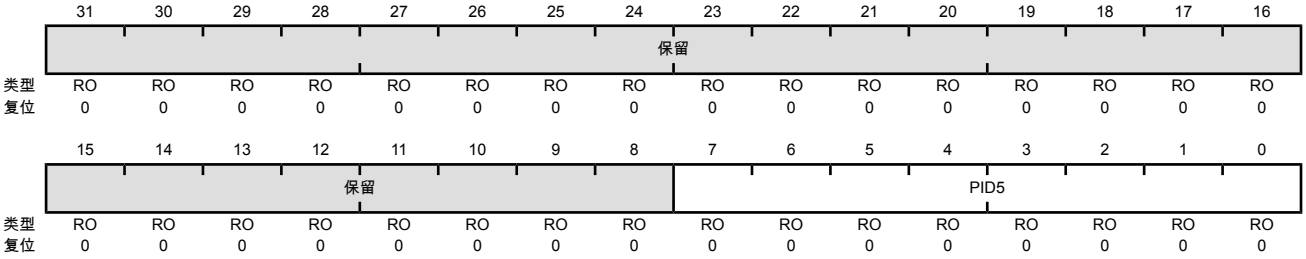
位/域	名称	类型	复位	描述
31:8	保留	RO	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
7:0	PID4	RO	0x00	GPIO外设标识寄存器[7..0]

寄存器 26: GPIO 外设标识寄存器 5（GPIOPeriphID5），偏移量 0xFD4

GPIOPeriphID4、GPIOPeriphID5、GPIOPeriphID6 和 GPIOPeriphID7 寄存器在概念上可以看作一个 32 位寄存器；每个寄存器包含 32 位寄存器的 8 个位，可被软件用来标识外设。

GPIO 外设标识寄存器 5 (GPIOPeriphID5)

GPIO 端口 A (APB) 基址: 0x4000.4000
GPIO 端口 A (AHB) 基址: 0x4005.8000
GPIO 端口 B (APB) 基址: 0x4000.5000
GPIO 端口 B (AHB) 基址: 0x4005.9000
GPIO 端口 C (APB) 基址: 0x4000.6000
GPIO 端口 C (AHB) 基址: 0x4005.A000
GPIO 端口 D (APB) 基址: 0x4000.7000
GPIO 端口 D (AHB) 基址: 0x4005.B000
GPIO 端口 E (APB) 基址: 0x4002.4000
GPIO 端口 E (AHB) 基址: 0x4005.C000
GPIO 端口 F (APB) 基址: 0x4002.5000
GPIO 端口 F (AHB) 基址: 0x4005.D000
偏移量 0xFD4
类型 RO, 复位 0x0000.0000



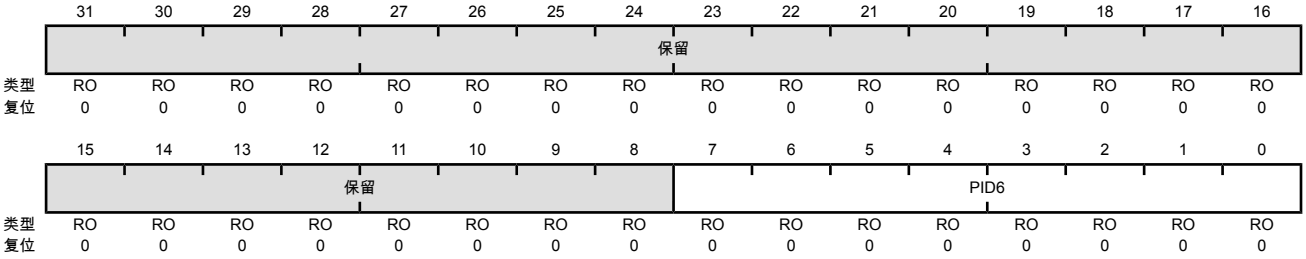
位/域	名称	类型	复位	描述
31:8	保留	RO	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
7:0	PID5	RO	0x00	GPIO外设标识寄存器[15..8]

寄存器 27: GPIO 外设标识寄存器 6（GPIOPeriphID6），偏移量0xFD8

GPIOPeriphID4、GPIOPeriphID5、GPIOPeriphID6 和 GPIOPeriphID7 寄存器在概念上可以看作一个 32 位寄存器；每个寄存器包含 32 位寄存器的 8 个位，可被软件用来标识外设。

GPIO 外设标识寄存器 6 (GPIOPeriphID6)

GPIO 端口 A (APB) 基址: 0x4000.4000
GPIO 端口 A (AHB) 基址: 0x4005.8000
GPIO 端口 B (APB) 基址: 0x4000.5000
GPIO 端口 B (AHB) 基址: 0x4005.9000
GPIO 端口 C (APB) 基址: 0x4000.6000
GPIO 端口 C (AHB) 基址: 0x4005.A000
GPIO 端口 D (APB) 基址: 0x4000.7000
GPIO 端口 D (AHB) 基址: 0x4005.B000
GPIO 端口 E (APB) 基址: 0x4002.4000
GPIO 端口 E (AHB) 基址: 0x4005.C000
GPIO 端口 F (APB) 基址: 0x4002.5000
GPIO 端口 F (AHB) 基址: 0x4005.D000
偏移量 0xFD8
类型 RO, 复位 0x0000.0000



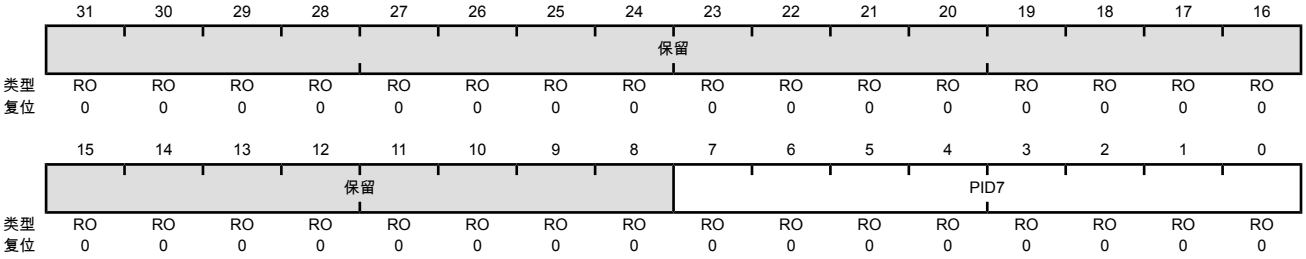
位/域	名称	类型	复位	描述
31:8	保留	RO	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
7:0	PID6	RO	0x00	GPIO 外设 ID 寄存器 [23:16]

寄存器 28: GPIO 外设标识寄存器 7（GPIOPeriphID7），偏移量 0xFDC

GPIOPeriphID4、GPIOPeriphID5、GPIOPeriphID6 和 GPIOPeriphID7 寄存器在概念上可以看作一个 32 位寄存器；每个寄存器包含 32 位寄存器的 8 个位，可被软件用来标识外设。

GPIO 外设标识寄存器 7 (GPIOPeriphID7)

GPIO 端口 A (APB) 基址: 0x4000.4000
GPIO 端口 A (AHB) 基址: 0x4005.8000
GPIO 端口 B (APB) 基址: 0x4000.5000
GPIO 端口 B (AHB) 基址: 0x4005.9000
GPIO 端口 C (APB) 基址: 0x4000.6000
GPIO 端口 C (AHB) 基址: 0x4005.A000
GPIO 端口 D (APB) 基址: 0x4000.7000
GPIO 端口 D (AHB) 基址: 0x4005.B000
GPIO 端口 E (APB) 基址: 0x4002.4000
GPIO 端口 E (AHB) 基址: 0x4005.C000
GPIO 端口 F (APB) 基址: 0x4002.5000
GPIO 端口 F (AHB) 基址: 0x4005.D000
偏移量 0xFDC
类型 RO, 复位 0x0000.0000



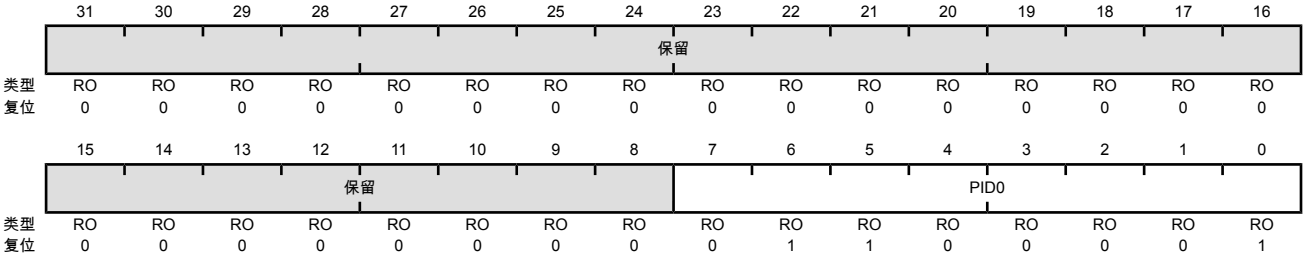
位/域	名称	类型	复位	描述
31:8	保留	RO	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
7:0	PID7	RO	0x00	GPIO外设标识寄存器[31..24]

寄存器 29: GPIO 外设标识寄存器 0（GPIOPeriphID0），偏移量 0xFE0

GPIOPeriphID0、GPIOPeriphID1、GPIOPeriphID2 和 GPIOPeriphID3 寄存器在概念上可以看作一个 32 位寄存器；每个寄存器包含 32 位寄存器的 8 个位，可被软件用来标识外设。

GPIO 外设标识寄存器 0 (GPIOPeriphID0)

GPIO 端口 A (APB) 基址: 0x4000.4000
GPIO 端口 A (AHB) 基址: 0x4005.8000
GPIO 端口 B (APB) 基址: 0x4000.5000
GPIO 端口 B (AHB) 基址: 0x4005.9000
GPIO 端口 C (APB) 基址: 0x4000.6000
GPIO 端口 C (AHB) 基址: 0x4005.A000
GPIO 端口 D (APB) 基址: 0x4000.7000
GPIO 端口 D (AHB) 基址: 0x4005.B000
GPIO 端口 E (APB) 基址: 0x4002.4000
GPIO 端口 E (AHB) 基址: 0x4005.C000
GPIO 端口 F (APB) 基址: 0x4002.5000
GPIO 端口 F (AHB) 基址: 0x4005.D000
偏移量 0xFE0
类型 RO, 复位 0x0000.0061



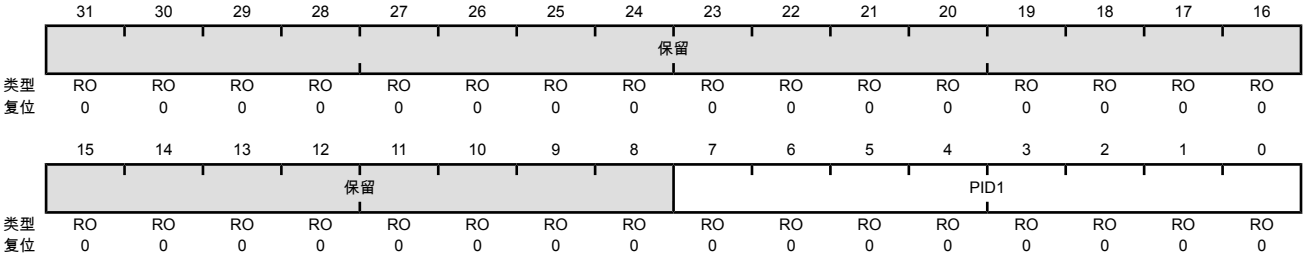
位/域	名称	类型	复位	描述
31:8	保留	RO	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
7:0	PID0	RO	0x61	GPIO外设标识寄存器[7..0] 可被软件用来标识该外设的存在与否。

寄存器 30: GPIO 外设标识寄存器 1（GPIOPeriphID1），偏移量 0xFE4

GPIOPeriphID0、GPIOPeriphID1、GPIOPeriphID2 和 GPIOPeriphID3 寄存器在概念上可以看作一个 32 位寄存器；每个寄存器包含 32 位寄存器的 8 个位，可被软件用来标识外设。

GPIO 外设标识寄存器 1 (GPIOPeriphID1)

GPIO 端口 A (APB) 基址: 0x4000.4000
GPIO 端口 A (AHB) 基址: 0x4005.8000
GPIO 端口 B (APB) 基址: 0x4000.5000
GPIO 端口 B (AHB) 基址: 0x4005.9000
GPIO 端口 C (APB) 基址: 0x4000.6000
GPIO 端口 C (AHB) 基址: 0x4005.A000
GPIO 端口 D (APB) 基址: 0x4000.7000
GPIO 端口 D (AHB) 基址: 0x4005.B000
GPIO 端口 E (APB) 基址: 0x4002.4000
GPIO 端口 E (AHB) 基址: 0x4005.C000
GPIO 端口 F (APB) 基址: 0x4002.5000
GPIO 端口 F (AHB) 基址: 0x4005.D000
偏移量 0xFE4
类型 RO, 复位 0x0000.0000



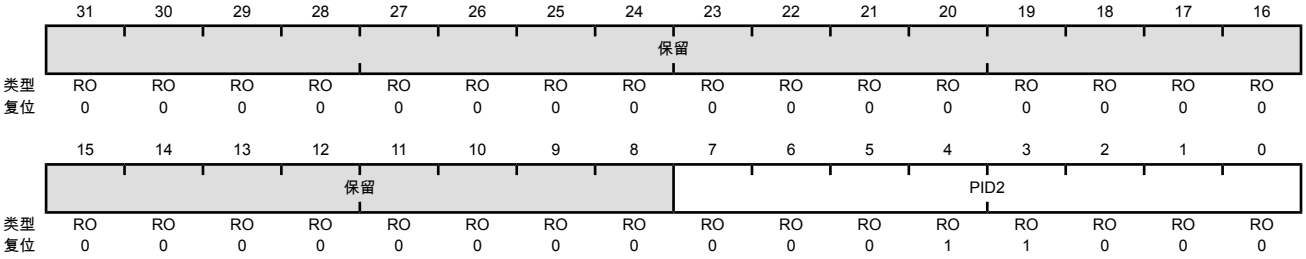
位/域	名称	类型	复位	描述
31:8	保留	RO	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
7:0	PID1	RO	0x00	GPIO外设标识寄存器[15..8] 可被软件用来标识该外设的存在与否。

寄存器 31: GPIO 外设标识寄存器 2（GPIOPeriphID2），偏移量 0xFE8

GPIOPeriphID0、GPIOPeriphID1、GPIOPeriphID2 和 GPIOPeriphID3 寄存器在概念上可以看作一个 32 位寄存器；每个寄存器包含 32 位寄存器的 8 个位，可被软件用来标识外设。

GPIO 外设标识寄存器 2 (GPIOPeriphID2)

GPIO 端口 A (APB) 基址: 0x4000.4000
GPIO 端口 A (AHB) 基址: 0x4005.8000
GPIO 端口 B (APB) 基址: 0x4000.5000
GPIO 端口 B (AHB) 基址: 0x4005.9000
GPIO 端口 C (APB) 基址: 0x4000.6000
GPIO 端口 C (AHB) 基址: 0x4005.A000
GPIO 端口 D (APB) 基址: 0x4000.7000
GPIO 端口 D (AHB) 基址: 0x4005.B000
GPIO 端口 E (APB) 基址: 0x4002.4000
GPIO 端口 E (AHB) 基址: 0x4005.C000
GPIO 端口 F (APB) 基址: 0x4002.5000
GPIO 端口 F (AHB) 基址: 0x4005.D000
偏移量 0xFE8
类型 RO, 复位 0x0000.0018



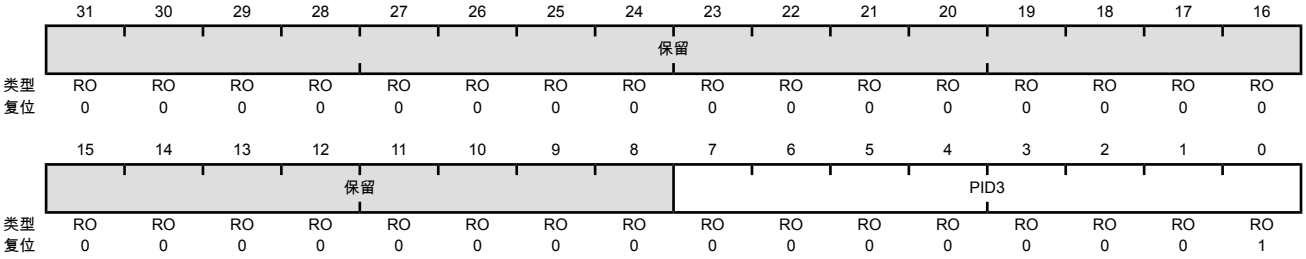
位/域	名称	类型	复位	描述
31:8	保留	RO	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
7:0	PID2	RO	0x18	GPIO 外设 ID 寄存器 [23:16] 可被软件用来标识该外设的存在与否。

寄存器 32: GPIO 外设标识寄存器 3（GPIOPeriphID3），偏移量 0xFEC

GPIOPeriphID0、GPIOPeriphID1、GPIOPeriphID2 和 GPIOPeriphID3 寄存器在概念上可以看作一个 32 位寄存器；每个寄存器包含 32 位寄存器的 8 个位，可被软件用来标识外设。

GPIO 外设标识寄存器 3 (GPIOPeriphID3)

GPIO 端口 A (APB) 基址: 0x4000.4000
GPIO 端口 A (AHB) 基址: 0x4005.8000
GPIO 端口 B (APB) 基址: 0x4000.5000
GPIO 端口 B (AHB) 基址: 0x4005.9000
GPIO 端口 C (APB) 基址: 0x4000.6000
GPIO 端口 C (AHB) 基址: 0x4005.A000
GPIO 端口 D (APB) 基址: 0x4000.7000
GPIO 端口 D (AHB) 基址: 0x4005.B000
GPIO 端口 E (APB) 基址: 0x4002.4000
GPIO 端口 E (AHB) 基址: 0x4005.C000
GPIO 端口 F (APB) 基址: 0x4002.5000
GPIO 端口 F (AHB) 基址: 0x4005.D000
偏移量 0xFEC
类型 RO, 复位 0x0000.0001



位/域	名称	类型	复位	描述
31:8	保留	RO	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
7:0	PID3	RO	0x01	GPIO外设标识寄存器[31..24] 可被软件用来标识该外设的存在与否。

寄存器 33: GPIO PrimeCell 标识寄存器 0（GPIOPCellID0），偏移量 0xFF0

GPIOPCellID0、GPIOPCellID1、GPIOPCellID2 和 GPIOPCellID3 这四个寄存器都是 8 位寄存器，在概念上可以将它们看作一个 32 位寄存器。寄存器将作为一个标准的交叉外设（cross-peripheral）标识系统来使用。

GPIO PrimeCell 标识寄存器 0 (GPIOPCellID0)

GPIO 端口 A (APB) 基址: 0x4000.4000
GPIO 端口 A (AHB) 基址: 0x4005.8000
GPIO 端口 B (APB) 基址: 0x4000.5000
GPIO 端口 B (AHB) 基址: 0x4005.9000
GPIO 端口 C (APB) 基址: 0x4000.6000
GPIO 端口 C (AHB) 基址: 0x4005.A000
GPIO 端口 D (APB) 基址: 0x4000.7000
GPIO 端口 D (AHB) 基址: 0x4005.B000
GPIO 端口 E (APB) 基址: 0x4002.4000
GPIO 端口 E (AHB) 基址: 0x4005.C000
GPIO 端口 F (APB) 基址: 0x4002.5000
GPIO 端口 F (AHB) 基址: 0x4005.D000
偏移量 0xFF0
类型 RO, 复位 0x0000.000D

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	保留															
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	保留								CID0							
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	1

位/域	名称	类型	复位	描述
31:8	保留	RO	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
7:0	CID0	RO	0x0D	GPIOPrimeCell标识寄存器[7..0] 为软件提供一个标准的交叉外设识别系统。

寄存器 34: GPIO PrimeCell 标识寄存器 1（GPIOPCellID1），偏移量 0xFF4

GPIOPCellID0、GPIOPCellID1、GPIOPCellID2 和 GPIOPCellID3 这四个寄存器都是 8 位寄存器，在概念上可以将它们看作一个 32 位寄存器。寄存器将作为一个标准的交叉外设（cross-peripheral）标识系统来使用。

GPIO PrimeCell 标识寄存器 1 (GPIOPCellID1)

GPIO 端口 A (APB) 基址: 0x4000.4000
GPIO 端口 A (AHB) 基址: 0x4005.8000
GPIO 端口 B (APB) 基址: 0x4000.5000
GPIO 端口 B (AHB) 基址: 0x4005.9000
GPIO 端口 C (APB) 基址: 0x4000.6000
GPIO 端口 C (AHB) 基址: 0x4005.A000
GPIO 端口 D (APB) 基址: 0x4000.7000
GPIO 端口 D (AHB) 基址: 0x4005.B000
GPIO 端口 E (APB) 基址: 0x4002.4000
GPIO 端口 E (AHB) 基址: 0x4005.C000
GPIO 端口 F (APB) 基址: 0x4002.5000
GPIO 端口 F (AHB) 基址: 0x4005.D000
偏移量 0xFF4
类型 RO, 复位 0x0000.00F0

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	保留															
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	保留								CID1							
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	0

位/域	名称	类型	复位	描述
31:8	保留	RO	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
7:0	CID1	RO	0xF0	GPIOPrimeCell标识寄存器[15..8] 为软件提供一个标准的交叉外设识别系统。

寄存器 35: GPIO PrimeCell 标识寄存器 2（GPIOPCellID2），偏移量 0xFF8

GPIOPCellID0、GPIOPCellID1、GPIOPCellID2 和 GPIOPCellID3 这四个寄存器都是 8 位寄存器，在概念上可以将它们看作一个 32 位寄存器。寄存器将作为一个标准的交叉外设（cross-peripheral）标识系统来使用。

GPIO PrimeCell 标识寄存器 2 (GPIOPCellID2)

GPIO 端口 A (APB) 基址: 0x4000.4000
GPIO 端口 A (AHB) 基址: 0x4005.8000
GPIO 端口 B (APB) 基址: 0x4000.5000
GPIO 端口 B (AHB) 基址: 0x4005.9000
GPIO 端口 C (APB) 基址: 0x4000.6000
GPIO 端口 C (AHB) 基址: 0x4005.A000
GPIO 端口 D (APB) 基址: 0x4000.7000
GPIO 端口 D (AHB) 基址: 0x4005.B000
GPIO 端口 E (APB) 基址: 0x4002.4000
GPIO 端口 E (AHB) 基址: 0x4005.C000
GPIO 端口 F (APB) 基址: 0x4002.5000
GPIO 端口 F (AHB) 基址: 0x4005.D000
偏移量 0xFF8
类型 RO, 复位 0x0000.0005

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
保留															
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留								CID2							
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

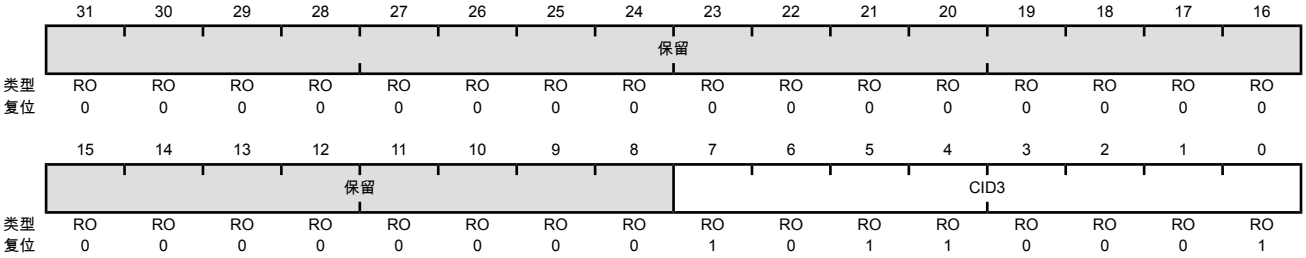
位/域	名称	类型	复位	描述
31:8	保留	RO	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
7:0	CID2	RO	0x05	GPIO PrimeCell ID 寄存器 [23:16] 为软件提供一个标准的交叉外设识别系统。

寄存器 36: GPIO PrimeCell 标识寄存器 3（GPIOPCellID3），偏移量 0xFFC

GPIOPCellID0、GPIOPCellID1、GPIOPCellID2 和 GPIOPCellID3 这四个寄存器都是 8 位寄存器，在概念上可以将它们看作一个 32 位寄存器。寄存器将作为一个标准的交叉外设（cross-peripheral）标识系统来使用。

GPIO PrimeCell 标识寄存器 3 (GPIOPCellID3)

GPIO 端口 A (APB) 基址: 0x4000.4000
GPIO 端口 A (AHB) 基址: 0x4005.8000
GPIO 端口 B (APB) 基址: 0x4000.5000
GPIO 端口 B (AHB) 基址: 0x4005.9000
GPIO 端口 C (APB) 基址: 0x4000.6000
GPIO 端口 C (AHB) 基址: 0x4005.A000
GPIO 端口 D (APB) 基址: 0x4000.7000
GPIO 端口 D (AHB) 基址: 0x4005.B000
GPIO 端口 E (APB) 基址: 0x4002.4000
GPIO 端口 E (AHB) 基址: 0x4005.C000
GPIO 端口 F (APB) 基址: 0x4002.5000
GPIO 端口 F (AHB) 基址: 0x4005.D000
偏移量 0xFFC
类型 RO, 复位 0x0000.00B1



位/域	名称	类型	复位	描述
31:8	保留	RO	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
7:0	CID3	RO	0xB1	GPIOPrimeCell标识寄存器[31..24] 为软件提供一个标准的交叉外设识别系统。

11 通用定时器

可编程定时器可对驱动定时器输入管脚的外部事件进行计数或定时。TM4C1231H6PM 通用定时器模块 (GPTM) 包含 6 个 16/32 位 GPTM 模块和 6 个 32/64 位宽 GPTM 模块。每个 16/32 位 GPTM 模块提供两个 16 位的定时器/计数器（称作 Timer A 和 Timer B），用户可以将它们配置成独立运行的定时器或事件计数器，或将它们连接成一个 32 位定时器或一个 32 位实时时钟 (RTC)。每个 32/64 位宽 GPTM 模块为 Timer A 和 Timer B 提供 32 位定时器，可以将它们连接成一个 64 位定时器来运行。它还可以触发微型直接内存访问传输 (µDMA)。

此外，如果在周期和单次触发模式下发生超时，定时器还可用于触发模数转换 (ADC)。由于所有通用定时器的触发信号在到达ADC模块前一起进行或操作，因而只需使用一个定时器来触发ADC事件。

通用定时器 (GPT) 模块是 Tiva™ C 系列 微控制器。其它定时器资源还包括系统定时器 (SysTick)（见101）。

通用定时器模块 (GPTM) 包含 6 个 16/32 位 GPTM 模块和 6 个 32/64 位宽 GPTM 模块，并提供以下功能选项：

- 16/32 位运行模式：
  - 16 位或 32 位可编程的单次定时器
  - 16 位或 32 位可编程的周期定时器
  - 具有 8 位预分频的 16 位通用定时器
  - 当有 32.768 KHz 的外部时钟源时可作为 32 位的实时时钟
  - 16 位输入沿计数或定时捕获模式，并带 8 位的预分频器
  - 带 8 位预分频器的 16 位 PWM 模式以及软件编程实现的 PWM 信号反相输出
- 32/64 位运行模式：
  - 32 位或 64 位可编程的单次定时器
  - 32 位或 64 位可编程的周期定时器
  - 具有 16 位预分频的 32 位通用定时器
  - 当有 32.768 KHz 的外部时钟源时可作为 64 位的实时时钟
  - 带有 16 位预分频器的 32 位输入沿计数或定时捕获模块
  - 带有 16 位预分频器 的 32 位 PWM 模式以及软件编程实现的 PWM 信号反相输出
- 可以向上或向下计数
- 十二个 16/32 位捕获比较 PWM 管脚 (CCP)
- 十二个 32/64 位捕捉比较 PWM 管脚 (CCP)
- 菊花链式的定时器模块允许一个定时器开始计时多路时钟事件
- 定时器同步功能允许所选的定时器在同一时钟周期开始计数

- 模数转换(ADC)触发器
- 当调试时，CPU 出现暂停标识时，用户可以停止定时器事件（包括 RTC 模式）
- 可以确定从产生定时器中断到进入中断服务程序所经过的时间
- 用微型直接内存访问 (µDMA) 有效的传输数据
  - 每个定时器具有专用通道
  - 定时器中断响应突发请求

11.1 结构框图

在结构图中，可用的特定捕获比较 PWM (CCP) 管脚取决于 TM4C1231H6PM 器件。请参见表 11-1（629页）了解可用的 CCP 管脚和它们的定时器分配。

图 **11-1. GPTM** 模块的结构图

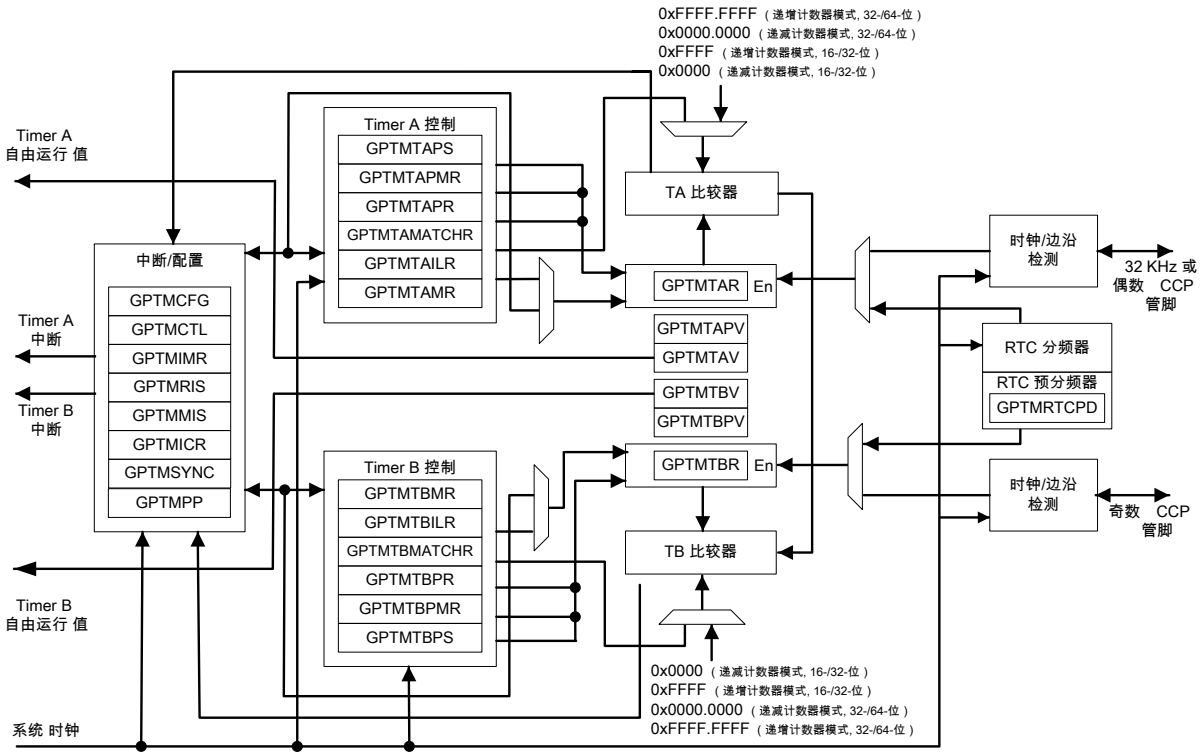


表 **11-1.** 可用的 **CCP** 管脚

定时器	递增/递减计数器	偶数CCP管脚	奇数CCP管脚
16/32 位 Timer 0	Timer A	T0CCP0	-
	Timer B	-	T0CCP1
16/32 位 Timer 1	Timer A	T1CCP0	-
	Timer B	-	T1CCP1
16/32 位 Timer 2	Timer A	T2CCP0	-
	Timer B	-	T2CCP1

表 **11-1.** 可用的 **CCP** 管脚（续）

定时器	递增/递减计数器	偶数CCP管脚	奇数CCP管脚
16/32 位 Timer 3	Timer A	T3CCP0	-
	Timer B	-	T3CCP1
16/32 位 Timer 4	Timer A	T4CCP0	-
	Timer B	-	T4CCP1
16/32 位 Timer 5	Timer A	T5CCP0	-
	Timer B	-	T5CCP1
32/64 位宽 Timer 0	Timer A	WT0CCP0	-
	Timer B	-	WT0CCP1
32/64 位宽 Timer 1	Timer A	WT1CCP0	-
	Timer B	-	WT1CCP1
32/64 位宽 Timer 2	Timer A	WT2CCP0	-
	Timer B	-	WT2CCP1
32/64 位宽 Timer 3	Timer A	WT3CCP0	-
	Timer B	-	WT3CCP1
32/64 位宽 Timer 4	Timer A	WT4CCP0	-
	Timer B	-	WT4CCP1
32/64 位宽 Timer 5	Timer A	WT5CCP0	-
	Timer B	-	WT5CCP1

11.2 信号描述

以下表格列出了通用定时器模块的外部信号，并描述了各自的功能。通用定时器引脚可能是默认为GPIO功能的GPIO引脚的备用功能。下表中的“引脚复用/分配”一列给出了可作为定时器的引脚。GPIO 备用功能选择 (GPIOAFSEL) 寄存器 (595页) 里的 AFSEL 位应被置位以选择通用定时器功能。括号中的数字是必须写入 GPIO 端口控制 (GPIOPCTL) 寄存器 (612页) 里的 PMCn 域的编码，以便将通用定时器信号配置为指定的 GPIO 端口管脚。有关如何配置 GPIO 的更多信息，请参阅“通用输入/输入端口（GPIOs）”（575页）。

表 **11-2.** 通用定时器 信号 **(64LQFP)**

管脚名称	管脚编号	管脚复用/管脚赋值	管脚类型	缓冲区类型[a]	描述
T0CCP0	1 28	PB6 (7) PF0 (7)	I/O	TTL	16/32 位 Timer 0 捕获/比较/PWM 0。
T0CCP1	4 29	PB7 (7) PF1 (7)	I/O	TTL	16/32 位 Timer 0 捕获/比较/PWM 1。
T1CCP0	30 58	PF2 (7) PB4 (7)	I/O	TTL	16/32 位 Timer 1 捕获/比较/PWM 0。
T1CCP1	31 57	PF3 (7) PB5 (7)	I/O	TTL	16/32 位 Timer 1 捕获/比较/PWM 1。
T2CCP0	5 45	PF4 (7) PB0 (7)	I/O	TTL	16/32 位 Timer 2 捕获/比较/PWM 0。
T2CCP1	46	PB1 (7)	I/O	TTL	16/32 位 Timer 2 捕获/比较/PWM 1。
T3CCP0	47	PB2 (7)	I/O	TTL	16/32 位 Timer 3 捕获/比较/PWM 0。
T3CCP1	48	PB3 (7)	I/O	TTL	16/32 位 Timer 3 捕获/比较/PWM 1。
T4CCP0	52	PC0 (7)	I/O	TTL	16/32 位 Timer 4 捕获/比较/PWM 0。

表 **11-2.** 通用定时器 信号 **(64LQFP)**（续）

管脚名称	管脚编号	管脚复用/管脚赋值	管脚类型	缓冲区类型[a]	描述
T4CCP1	51	PC1 (7)	I/O	TTL	16/32 位 Timer 4 捕获/比较/PWM 1。
T5CCP0	50	PC2 (7)	I/O	TTL	16/32 位 Timer 5 捕获/比较/PWM 0。
T5CCP1	49	PC3 (7)	I/O	TTL	16/32 位 Timer 5 捕获/比较/PWM 1。
WT0CCP0	16	PC4 (7)	I/O	TTL	32/64 位宽 Timer 0 捕获/比较/PWM 0。
WT0CCP1	15	PC5 (7)	I/O	TTL	32/64 位宽 Timer 0 捕获/比较/PWM 1。
WT1CCP0	14	PC6 (7)	I/O	TTL	32/64 位宽 Timer 1 捕获/比较/PWM 0。
WT1CCP1	13	PC7 (7)	I/O	TTL	32/64 位宽 Timer 1 捕获/比较/PWM 1。
WT2CCP0	61	PD0 (7)	I/O	TTL	32/64 位宽 Timer 2 捕获/比较/PWM 0。
WT2CCP1	62	PD1 (7)	I/O	TTL	32/64 位宽 Timer 2 捕获/比较/PWM 1。
WT3CCP0	63	PD2 (7)	I/O	TTL	32/64 位宽 Timer 3 捕获/比较/PWM 0。
WT3CCP1	64	PD3 (7)	I/O	TTL	32/64 位宽 Timer 3 捕获/比较/PWM 1。
WT4CCP0	43	PD4 (7)	I/O	TTL	32/64 位宽 Timer 4 捕获/比较/PWM 0。
WT4CCP1	44	PD5 (7)	I/O	TTL	32/64 位宽 Timer 4 捕获/比较/PWM 1。
WT5CCP0	53	PD6 (7)	I/O	TTL	32/64 位宽 Timer 5 捕获/比较/PWM 0。
WT5CCP1	10	PD7 (7)	I/O	TTL	32/64 位宽 Timer 5 捕获/比较/PWM 1。

a. TTL 表示管脚的电压水平与 TTL 一致。

11.3 功能说明

每个 GPTM 模块的主要元件包括两个自由运行的递增/递减计数器（称作 Timer A 和 Timer B）、两个预分频器寄存器、两个匹配寄存器、两个预分频器匹配寄存器、两个影子寄存器、两个加载/初始化寄存器以及与它们相关的控制功能。GPTM的准确功能可由软件来控制，并通过寄存器接口进行配置。Timer A 和 Timer B 可以独立使用，在这种情况下，它们拥有针对 16/32 位 GPTM 模块的 16 位的计数范围和针对 32/64 位宽 GPTM 模块的 32 位的计数范围。此外，可以将 Timer A 和 Timer B 连在一起为 16/32 位 GPTM 模块提供 32 位的计数范围，以及为 32/64 位宽 GPTM 模块提供 64 位的计数范围。请注意仅可以在单独使用定时器时使用预分频器。

各个 GPTM 模块的可用模式在 表11-3（631页）中显示。请注意在单次触发或周期模式下递减计数时，预分频器用作真预分频器并且包含计数的最低位。在单次触发或周期模式下递增计数时，预分频器用作定时器扩展并且包含计数的最高位。在输入边沿计数、输入边沿计时和 PWM 模式下，预分频器总是用作定时器扩展，而不论计数方向。

表 **11-3.** 通用定时器功能

模式	定时器使用	计数方向	计数器大小		预分频器大小[a]		预分频器行为（计数方向）
			16/32 位 **GPTM**	**32/64** 位宽 **GPTM**	**16/32** 位 **GPTM**	**32/64** 位宽 **GPTM**	
单次触发	独立	递增或递减	16 位	32 位	8 位	16 位	定时器扩展（递增），预分频器（递减）
	连接	递增或递减	32 位	64 位	-	-	N/A
周期	独立	递增或递减	16 位	32 位	8 位	16 位	定时器扩展（递增），预分频器（递减）
	连接	递增或递减	32 位	64 位	-	-	N/A
RTC	连接	递增	32 位	64 位	-	-	N/A
边沿计数	独立	递增或递减	16 位	32 位	8 位	16 位	定时器扩展（两者均有）

表 **11-3.** 通用定时器功能（续）

模式	定时器使用	计数方向	计数器大小		预分频器大小[a]		预分频器行为（计数方向）
			16/32 位 **GPTM**	**32/64** 位宽 **GPTM**	**16/32** 位 **GPTM**	**32/64** 位宽 **GPTM**	
边沿时间	独立	递增或递减	16 位	32 位	8 位	16 位	定时器扩展（两者均有）
PWM	独立	递减	16 位	32 位	8 位	16 位	定时器扩展

a. 仅可以在单独使用定时器时使用预分频器

软件使用 GPTM 配置 (GPTMCFG) 寄存器（见650页）、GPTM Timer A 模式 (GPTMTAMR) 寄存器（见651页）和 GPTM Timer B 模式 (GPTMTBMR) 寄存器（见654页）配置 GPTM。处于其中的一个连接模式中时，Timer A 和 Timer B 仅能在一个模式中运行。但是，在独立模式中配置时，可以在任何独立模式的组合中自由配置 Timer A 和 Timer B。

11.3.1 GPTM复位条件

GPTM模块复位后处于未激活状态，所有控制寄存器均被清零，同时进入默认状态。计数器的 Timer A 和 Timer B 均初始化至全部为 1，其相应的寄存器为：

- 加载寄存器：
  - GPTM Timer A 间隔加载 (GPTMTAILR) 寄存器（请参阅674页）
  - GPTM Timer B 间隔加载 (GPTMTBILR) 寄存器（请参阅675页）
- 影子寄存器：
  - GPTM Timer A 值 (GPTMTAV) 寄存器（请参阅684页）
  - GPTM Timer B 值 (GPTMTBV) 寄存器（请参阅685页）

以下预分频计数器均初始化至全部为 0：

- GPTM Timer A 预分频 (GPTMTAPR) 寄存器（请参阅678页）
- GPTM Timer B 预分频 (GPTMTBPR) 寄存器（请参阅679页）
- GPTM Timer A 预分频快照 (GPTMTAPS) 寄存器（请参阅687页）
- GPTM Timer B 预分频快照 (GPTMTBPS) 寄存器（请参阅688页）
- GPTM Timer A 预分频值 (GPTMTAPV) 寄存器（请参阅689页）
- GPTM Timer B 预分频值 (GPTMTBPV) 寄存器（请参阅690页）

11.3.2 定时器模式

此部分描述了各种定时器模式的运行。在连接模式中使用 Timer A 和 Timer B 时，仅必须使用 Timer A 控制和状态位；不需要使用 Timer B 控制和状态位。通过向 GPTM 配置 (GPTMCFG) 寄存器写入 0x4，可将 GPTM 配置为独立/分离模式（见650页）。在以下部分中，变量“n”用于位域和寄存器名称中，表示 Timer A 函数或 Timer B 函数。在本节中，递减计数模式中的超时事件为 0x0，而在递增模式中为 GPTM Timer n 间隔加载寄存器 (GPTMTnILR) 和可选的 GPTM Timer n 预分频寄存器 (GPTMTnPMR) 中的数值。

11.3.2.1 单次触发/周期定时器模式

选择单次触发模式还是周期模式由写入 GPTM Timer n 模式 (GPTMTnMR) 寄存器中 TnMR 域的值来决定（见651页）。定时器是递增计数还是递减计数由 GPTMTnMR 寄存器中的 TnCDIR 位来决定。

当软件置位了 GPTM 控制 (GPTMCTL) 寄存器（见657页）中的 TnEN 位，定时器开始从 0x0 开始递增计数或从预加载的值开始递减计数。另外，如果 GPTMTnMR 寄存器中的 TnWOT 被置位，一旦TnEN位被置位，定时器就会等待一个触发来开始计数（详情请参阅“等待触发模式”（642页））。表11-4（633页） 显示启用定时器时加载到定时器寄存器的值。

表 **11-4.** 单次触发或周期模式下启用定时器时的计数器值

寄存器	递减模式	递增模式
GPTMTnR	GPTMTnILR	0x0
GPTMTnV	连接模式下为 GPTMTnILR；独立模式下为 GPTMTnPR 与 GPTMTnILR 的组合	0x0
GPTMTnPS	独立模式下为 GPTMTnPR；连接模式下不可用	独立模式下为 0x0；连接模式下不可用
GPTMTnPV	独立模式下为 GPTMTnPR；连接模式下不可用	独立模式下为 0x0；连接模式下不可用

当定时器递减计数并且到达超时事件 (0x0) 时，定时器将会在下一个时钟周期从 GPTMTnILR 和 GPTMTnPR 寄存器重新加载其初值。当定时器递增计数并达到超时事件（GPTMTnILR 和可选的 GPTMTnPR 寄存器中的数值）时，定时器重新加载 0x0。如果配置为单次触发定时器，定时器将停止计数，并且将 GPTMCTL 寄存器中的 TnEN 位清零。如果配置为周期定时器，则定时器会在下一个时钟周期开始再次计数。

在周期、快照模式中（GPTMTnMR 寄存器中的 TnMR 域为 0x2，且 TnSNAPS 位置位），发生超时事件时的定时器值被加载到 GPTMTnR 寄存器，而预分频器的值被加载到 GPTMTnPS 寄存器。自由运行计数器的值显示在 GPTMTnV 寄存器中，自由运行预分频器的值显示在 GPTMTnPV 寄存器中。通过这种方法，软件能校验快照值和自由运行定时器的当前值，从而确定从发生中断到进入中断服务程序之间所用时间。定时器配置为单次触发模式时，快照模式不可用。

除了重新装载计数值，当达到超时时通用定时器就会触发并产生中断。GPTM将 GPTM 原始中断状态 (GPTMRIS) 寄存器（见666页）中的 TnTORIS 位置位，并保持该值直到向 GPTM 中断清除 (GPTMICR) 寄存器（见672页）执行写操作将其清零。如果在 GPTM 中断屏蔽 (GPTMIMR) 寄存器（见663页）中启用超时中断，GPTM 还将 GPTM 屏蔽的中断状态 (GPTMMIS) 寄存器（见669页）中的 TnTOMIS 位置位。将 GPTM Timer n 模式 (GPTMTnMR) 寄存器的 TACINTD 位置位即可完全禁用超时中断。此时，即便是 GPTMRIS 寄存器中的 TnTORIS 位也不会置位。

通过将 GPTMTnMR 寄存器中的 TnMIE 位置位，当定时器的值与加载到 GPTM Timer n 匹配 (GPTMTnMATCHR) 和 GPTM Timer n 预分频匹配值 (GPTMTnPMR) 寄存器的值相等时，也能产生中断条件。该中断和超时中断具有同样的状态、屏蔽和清零方式，但使用匹配中断位实现功能（例如，原始中断状态通过 GPTM 原始中断状态 (GPTMRIS) 寄存器的 TnMRIS 位监控）。注：中断状态位并不会由硬件更新，除非GPTMTnMR寄存器的TnMIE位已置位，这与超时中断的行为不同。通过将 GPTMCTL 中的 TnOTE 位置位以启用 ADC 触发。ADC 触发器启用时，仅单次触发或周期超时事件可致使 ADC 触发器有效。配置并启用相应的 µDMA 通道即可启用 µDMA 触发器。请参阅“通道配置”（516页）。

如果软件在计数器递减的过程中更新了 GPTMTnILR 或 GPTMTnPR 寄存器，计数器会在下一个时钟周期加载新值，并且如果 GPTMTnMR 寄存器中的 TnILD 位清零，则从新值开始继续计数。如果TnILD位置位，计数器在下一个超时后加载新值。如果软件在计数器递增的过程中更新了GPTMTnILR或 GPTMTnPR 寄存器，超时事件会在下一个时钟周期更改为新值。如果软件在计数器递增或递减的过程中更新了 GPTM Timer n 值 (GPTMTnV) 寄存器的值，计数器将会在下个时钟周期载入这个新值并从这个新值开始递增或递减。如果软件更新了 GPTMTnMATCHR 或 GPTMTnPMR 寄存器，

新值将在下一个时钟周期得到反映（如果GPTMTnMR寄存器的TnMRSU位已清零）。如果TnMRSU位置位，新值将在下一个超时后再生效。

在 64 位模式中使用 32/64 位宽定时器模块时，必须以 “访问连接的 32/64 位宽 GPTM 寄存器值”（644页 ） 中描述的方法访问某些寄存器。

如果 GPTMCTL 寄存器的 TnSTALL 位置位且 GPTMCTL 寄存器的 RTCEN 位不置位，定时器将会在调试器停止处理器时冻结计数。当处理器运行的时候定时器将会继续计数。如果将 RTCEN 位置位，则 TnSTALL 位不会在调试器停止处理器时冻结计数。

下面的表格显示了在使用预分频器时 16 位自由运行的定时器的各种配置。所有的值都是以时钟频率 n80-MHz（时钟周期 Tc=12.5 ns）作为标准进行计算。预分频器只能在 16/32 位定时器配置为 16 位模式和 32/64 位定时器配置为 32 位模式时使用。

表 **11-5.** 带预分频器的 **16** 位定时器配置

预分频（**8** 位值）	定时器时钟 **(Tc)** 编号[a]	最大时间	单位
00000000	1	0.8192	ms
00000001	2	1.6384	ms
00000010	3	2.4576	ms
----------	--	--	--
11111101	254	208.0768	ms
11111110	255	208.896	ms
11111111	256	209.7152	ms

a. Tc表示时钟周期。

下面的表格显示了在使用预分频器时 32 位自由运行的定时器（配置为 32/64 位模式）的各种配置。所有的值都是以时钟频率 n80-MHz（时钟周期 Tc=12.5 ns）作为标准进行计算。

表 **11-6.** 带预分频器配置的 **32** 位定时器（配置为 **32/64** 位模式）

预分频（**16** 位值）	定时器时钟 **(Tc)** 编号[a]	最大时间	单位
0x0000	1	53.687	s
0x0001	2	107.374	s
0x0002	3	214.748	s
----------	--	--	--
0xFFFD	65534	0.879	10^6 s
0xFFFE	65535	1.759	10^6 s
0xFFFF	65536	3.518	10^6 s

a. Tc表示时钟周期。

11.3.2.2 实时时钟模式

在实时时钟 (RTC) 模式中，Timer A 和 Timer B 寄存器连在一起被配置为递增计数器。复位后，首次选择 RTC 模式时，计数器加载的值为 0x1。所有后续加载的值必须写入 GPTM Timer n 间隔加载 (GPTMTnILR) 寄存器（请参阅674页 ）。如果 GPTMTnILR 寄存器加载了一个新值，则计数器将从该值开始计数，并在达到固定值 0xFFFFFFFF 时返回初始值并重新计数。表11-7（634页 ） 显示启用定时器时加载到定时器寄存器的值。

表 **11-7. RTC** 模式下启用定时器时的计数器值

寄存器	递减模式	递增模式
GPTMTnR	不可用	0x1

表 **11-7. RTC** 模式下启用定时器时的计数器值（续）

寄存器	递减模式	递增模式
GPTMTnV	不可用	0x1
GPTMTnPS	不可用	不可用
GPTMTnPV	不可用	不可用

在 RTC 模式中，要求 CCP0 输入时钟为 32.768 KHz。然后将时钟信号分频为 1 Hz，将其传送给计数器的输入端。

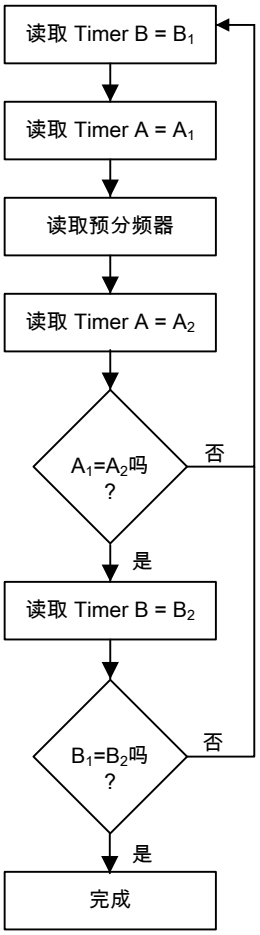
在软件写 GPTMCTL 寄存器中的 TAEN 位时，计数器从其预加载的值 0x1 开始递增计数。如果当前的计数值与 GPTMTnMATCHR 寄存器中预加载的值相匹配，GPTM 将使 GPTMRIS 中的 RTCRIS 位生效，并且继续计数，直到发生硬件复位或者软件将其禁用（通过清零 TAEN 位）。当定时器值达到终端计数时，定时器将会返回到 0x0 继续递增计数。如果 RTC 中断在 GPTMIMR 寄存器中被启用的话，GPTM 也会将 GPTMMIS 中的 RTCMIS 位置位，并且产生一个控制器中断。通过写 GPTMICR 寄存器中的 RTCCINT 位将状态标记清除。

在此模式中，GPTMTnR 和 GPTMTnV 寄存器总是具有相同值。

在 RTC 模式中使用 32/64 位宽定时器模块时，必须以“访问连接的 32/64 位宽 GPTM 寄存器值”（644页）中描述的方法访问某些寄存器。

RTC 预分频器的值可通过 GPTM RTC 预分频 (GPTMRTCPD) 寄存器读取。要确保 RTC 值的一致性，软件应遵循图11-2（636页）中的详细过程。

图 **11-2.** 读取 **RTC** 值



除了产生中断外，RTC 还可产生 µDMA 触发信号。通过配置并启用合适的 µDMA 通道来启用 µDMA 触发。请参阅“通道配置”（516页）。

11.3.2.3 输入边沿计数模式下：

注意: 对于上升沿检测，输入信号必须在上升沿到达高电平后至少保持两个系统时钟周期。同样，对于下降沿检测，输入信号在到达下降沿的低电平后至少保持两个系统时钟周期。有鉴于此，边沿检测输入的最大频率为系统频率的 1/4。

在边沿计数模式中，定时器被配置为 24 位或 48 位递增或递增或递减计数器，包括带有存储在 GPTM Timer n 预分频 (GPTMTnPR) 寄存器中的高计数值和 GPTMTnR 寄存器低位的可选预分频器。在此模式中，定时器能够捕获三种事件类型：上升沿、下降沿、或上升/下降沿。要想使用定时器的边沿计数模式，必须把 GPTMTnMR 寄存器 TnCMR 位清零。定时器计数的边沿类型由 GPTMCTL 寄存器中的 TnEVENT 位域的值决定。在递减模式的初始化过程中，需对 GPTMTnMATCHR 和 GPTMTnPMR 寄存器进行配置，使 GPTMTnILR 和 GPTMTnPR 寄存器以及 GPTMTnMATCHR 和 GPTMTnPMR 寄存器之间的差值等于必须计算的边沿事件的数目。在递增计数模式中，定时器从 0x0 开始计数到 GPTMTnMATCHR 和 GPTMTnPMR 寄存器中的值。请注意：执行递增计数时，GPTMTnPR 和 GPTMTnILR 的值必须大于 GPTMTnPMR 和 GPTMTnMATCHR 的值。表 11-8（637页）显示了定时器寄存器在启用定时器后加载的值。

表 **11-8.** 输入边沿计数模式下启用定时器时的计数器值

寄存器	递减模式	递增模式
GPTMTnR	GPTMTnPR 与 GPTMTnILR 组合	0x0
GPTMTnV	GPTMTnPR 与 GPTMTnILR 组合	0x0
GPTMTnPS	GPTMTnPR	0x0
GPTMTnPV	GPTMTnPR	0x0

当软件写 GPTM 控制 (GPTMCTL) 寄存器的 TnEN 位时，定时器将启用并用于事件捕获。CCP 管脚上每输入一个事件，计数器的值就递减或递增 1，直到事件计数的值与 GPTMTnMATCHR 和 GPTMTnPMR 的值匹配。计数匹配时，GPTM 让 GPTM 原始中断状态 (GPTMRIS) 寄存器中的 CnMRIS 位有效，并保持该值直到向 GPTM 中断清除 (GPTMICR) 寄存器执行写操作将其清零。如果在 GPTM 中断屏蔽 (GPTMIMR) 寄存器中启用捕获模式匹配中断，GPTM 还会将 GPTM 屏蔽的中断状态 (GPTMMIS) 寄存器中的 CnMMIS 位置位。在此模式中，GPTMTnR 和 GPTMTnPS 寄存器用于保存输入事件的计数，而 GPTMTnV 和 GPTMTnPV 寄存器用于保存自由运行定时器和自由运行预分频器的值。在递增计数模式中，输入事件的当前计数同时保存在 GPTMTnR 和 GPTMTnV 寄存器中。

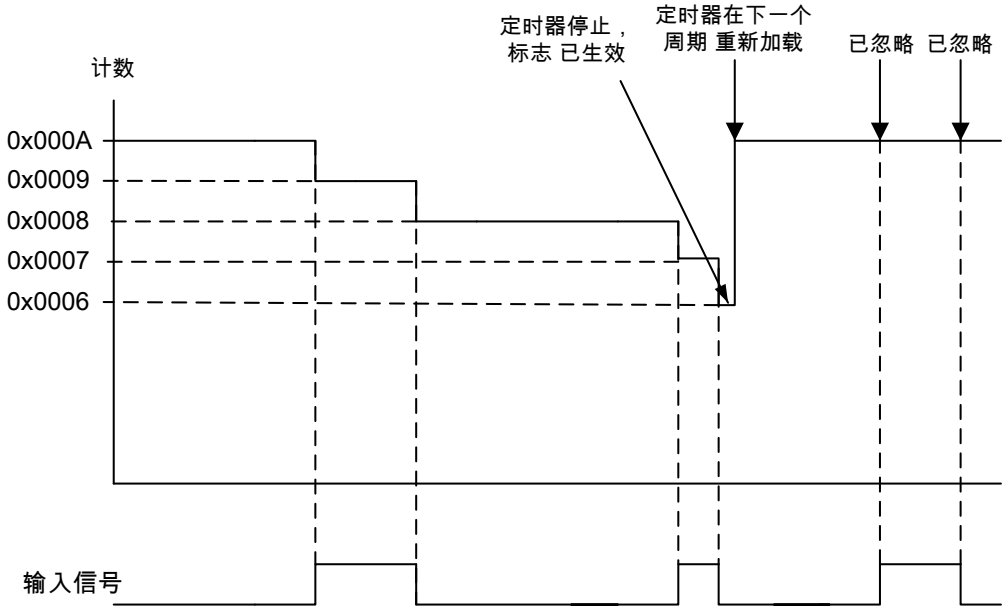
除产生中断之外，还可以产生 µDMA 触发。通过配置并启用合适的 µDMA 通道来启用 µDMA 触发。请参阅“通道配置”（516页）。

在递减模式中达到匹配值后，计数器使用 GPTMTnILR 和 GPTMTnPR 寄存器中的值执行重装操作，并且由于 GPTM 自动将 GPTMCTL 寄存器的 TnEN 清零，因此计数器停止计数。一旦事件计数值满足要求，接下来的所有事件都将被忽略，直到通过软件重新将 TnEN启用。在递增模式中，定时器重新加载 0x0 并继续计数。

图11-3（637页）显示了输入边沿计数模式的工作情况。在这种情况下，定时器的初值设置为 GPTMnILR = 0x000A，匹配值 GPTMnMATCHR = 0x0006。因此，需计数 4 个边沿事件。计数器配置为检测输入信号的上升/下降沿。

注：在当前计数值与 GPTMTnMATCHR 寄存器中的值匹配之后，定时器自动将 TnEN 位清零，因此最后两个边沿没有计算在内。

图 **11-3.** 输入边沿计数模式实例，递减计数



11.3.2.4 输入边沿计时模式

注意: 对于上升沿检测，输入信号必须在上升沿到达高电平后至少保持两个系统时钟周期。同样，下降沿检测在到达下降沿的低电平后必须至少保持两个系统时钟周期。有鉴于此，边沿检测输入的最大频率为系统频率的 1/4。

在边沿定时模式中，定时器被配置为 24 位或 48 位递增或递减计数器，包括带有存储在 GPTMTnPR 寄存器中的高定时器值和 GPTMTnILR 寄存器低位的可选预分频器。在此模式中，在递减计数时，定时器被初始化为 GPTMTnILR 和 GPTMTnPR 寄存器中加载的数值，递增计数时为 0x0。定时器能够捕获三种事件类型：上升沿、下降沿、或上升/下降沿。要想使用定时器的边沿定时模式，则必须把 GPTMTnMR 寄存器 TnCMR 位清零。定时器捕获的边沿类型由 GPTMCTL 寄存器中的 TnEVENT 位域的值决定。表11-9（638页） 显示启用定时器时加载到定时器寄存器的值。

表 **11-9.** 输入事件计数模式下启用定时器时的计数器值

寄存器	递减模式	递增模式
TnR	GPTMTnILR	0x0
TnV	GPTMTnILR	0x0
TnPS	GPTMTnPR	0x0
TnPV	GPTMTnPR	0x0

在软件写 GPTMCTL 寄存器的 TnEN 位时，定时器将启用并用于事件捕获。在检测到所选的输入事件时，GPTMTnR 和 GPTMTnPS 寄存器将捕获定时器计数器的当前值，且该值可通过微控制器来读取。然后 GPTM 让 GPTM 原始中断状态 (GPTMRIS) 寄存器中的 CnERIS 位有效，并保持该值直到向 GPTM 中断清除 (GPTMICR) 寄存器执行写操作将其清零。如果在 GPTM 中断屏蔽 (GPTMIMR) 寄存器中启用捕获模式事件中断，GPTM 还会将 GPTM 屏蔽的中断状态 (GPTMMIS) 寄存器中的 CnEMIS 位置位。这种模式下，GPTMTnR 和 GPTMTnPS 寄存器将保存发生选定输入事件的时间，而 GPTMTnV 和 GPTMTnPV 寄存器将保存自由运行定时器和自由运行预分频器的值。读取这些寄存器可以判定从发生中断到进入 ISR（中断服务程序）所用时间。

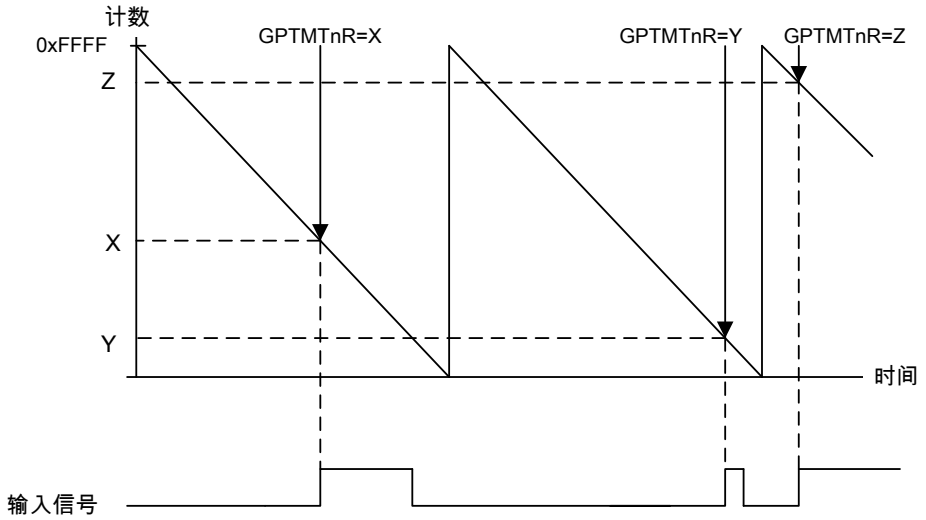
除产生中断之外，还可以产生 µDMA 触发。通过配置相应的 µDMA 通道即可启用 µDMA 触发信号。请参阅“通道配置”（516页）。

当捕获到一个事件时，定时器并不停止计数。它会继续计数直到 TnEN 位被清零。当定时器达到超时值时，在递增模式中重新加载 0x0，在递减模式中重新加载来自 GPTMTnILR 和 GPTMTnPR 寄存器的值。

图11-4（639页） 显示了输入边沿定时模式的工作原理。在图中，假设计数器的开始至是默认的 0xFFFF，并且定时器被配置为捕捉上升沿事件。

每当检测到上升沿事件时，当前计数值便加载到 GPTMTnR 和 GPTMTnPS 寄存器中，且该值一直保持在寄存器中直到检测到下一个上升沿（在此上升沿处，新的计数值加载到 GPTMTnR 和 GPTMTnPS 寄存器中）。

图 **11-4. 16** 位输入边沿计时模式实例



注意: 在边沿定时模式下工作时，计数器在启用预分频器时按模 2^{24} 计数，在未启用时预分频器时按 2^{16} 计数。如果边沿有可能比计数更长，则可执行周期定时器模式中配置的另一定时器，以确保检测到丢失的边沿。周期定时器的配置须确保满足以下条件：

- 周期定时器的运行周期与边沿定时定时器一致
- 周期定时器中断的优先级高于边沿定时超时中断。
- 如果启动了周期定时器中断服务例程，软件必须检查是否存在挂起的边沿定时中断。如果有，计数器的值必须减 1，然后再用于计算事件的快照时间。

11.3.2.5 PWM 模式

通用定时器支持简单的PWM生成模式。在 PWM 模式中，定时器被配置为 24 位或 48 位递减计数器，初值由 GPTMTnILR 和 GPTMTnPR 寄存器定义。在这种模式下，PWM周期和频率是同步事件，因此，保障了无毛刺。将 GPTMTnMR 寄存器的 TnAMS 位设为 0x1、TnCMR 位设为 0x0、TnMR 域设为 0x2 即可启用 PWM 模式表11-10（639页） 显示启用定时器时加载到定时器寄存器的值。

表 **11-10. PWM** 模式下启用定时器时的计数器值

寄存器	递减模式	递增模式
GPTMTnR	GPTMTnILR	不可用
GPTMTnV	GPTMTnILR	不可用
GPTMTnPS	GPTMTnPR	不可用
GPTMTnPV	GPTMTnPR	不可用

在软件写 GPTMCTL 寄存器的 TnEN 位时，计数器开始递减计数，直到计数值到达 0x0。另外，如果 GPTMTnMR 寄存器中的 TnWOT 被置位，一旦 TnEN 位被置位，定时器就会等待一个触发来开始计数（详情请参阅“等待触发模式”（642页））。在周期模式的下一个计数周期，计数器将 GPTMTnILR 和 GPTMTnPR 寄存器中的值重新载入，作为它的初值，并继续计数直到计数器因软件将 GPTMCTL 寄存器的 TnEN 位清零而被禁用。定时器能够基于三种事件类型产生中断。这三种

事件类型为：上升沿、下降沿或上升/下降沿。该事件通过 GPTMCTL 寄存器的 TnEVENT 域配置，而中断通过设置 GPTMTnMR 寄存器的 TnPWMIE 位启用。发生该事件时，将 GPTM 原始中断状态 (GPTMRIS) 寄存器中的 CnERIS 位置位，并保持该值直到向 GPTM 中断清除 (GPTMICR) 寄存器执行写操作将其清零。如果在 GPTM 中断屏蔽 (GPTMIMR) 寄存器中启用捕获模式事件中断，GPTM 还会将 GPTM 屏蔽的中断状态 (GPTMMIS) 寄存器中的 CnEMIS 位置位。注：中断状态位不会更新，除非 TnPWMIE 位已置位。

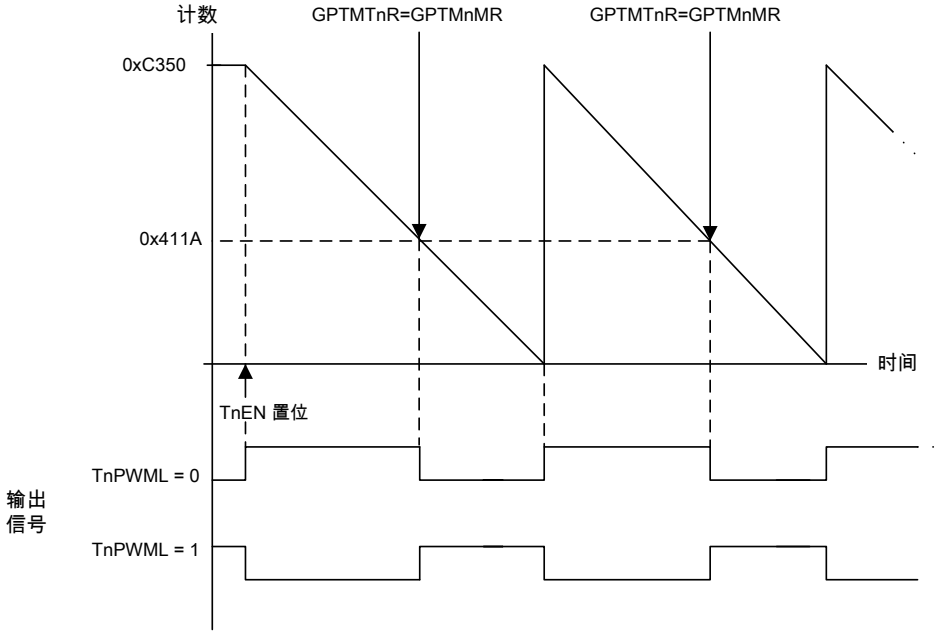
在此模式中，GPTMTnR 和 GPTMTnV 寄存器总是具有相同的值，GPTMPnPS 和 GPTMTnPV 寄存器亦然。

当计数器的值与 GPTMTnILR和 GPTMTnPR 寄存器的值（计数器的初始状态）相等时，输出 PWM 信号生效，当计数器的值与 GPTMTnMATCHR 和 GPTMTnPMR 寄存器的值相等时，输出 PWM 信号失效。通过将 GPTMCTL 寄存器的TnPWML 位置位，软件可实现将输出 PWM 信号反相的功能。

注意: 如果启用了 PWM 输出反相，将翻转边沿检测中断行为。因此，如果上升沿中断触发已置位，且 PWM 反相生成上升沿，将不存在有效的事件触发中断。相反，中断在 PWM 信号的下降沿生成。

图11-5（640页）显示了在输入时钟为 50 MHz 以及 TnPWML 为 0 的情况下，如何产生周期为 1ms、占空比为 66% 的输出 PWM（TnPWML =1 时，占空比为 33%）。在这个例子中，初值 GPTMTnILR = 0xC350，匹配值 GPTMTnMATCHR = 0x411A。

图 **11-5. 16-**位**PWM**模式实例



使用 GPTMSYNC 寄存器同步定时器时，定时器必须进行适当配置，以避免在 CCP 输出上出现故障。GPTMTnMR 寄存器的 PLO 和 MRSU 位必须置位。图11-6（641页）显示在 PLO 和 MRSU 位被置位并且 GPTMTnMATCHR 值大于 GPTMTnILR 值时，CCP 输出如何操作。

图 **11-6. CCP** 输出，**GPTMTnMATCHR > GPTMTnILR**

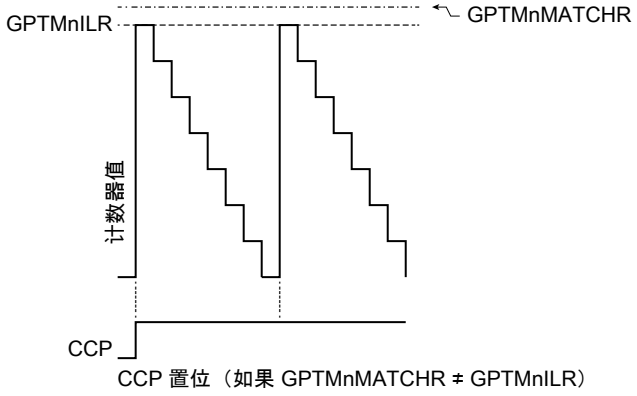


图11-7（641页）显示在 PLO 和 MRSU 位被置位并且 GPTMTnMATCHR 值等于 GPTMTnILR 值时，CCP 输出如何操作。此时，如果 PLO 位等于 0，CCP 信号在载入 GPTMTnILR 值时拉高，匹配基本上将被忽略。

图 **11-7. CCP** 输出，**GPTMTnMATCHR = GPTMTnILR**

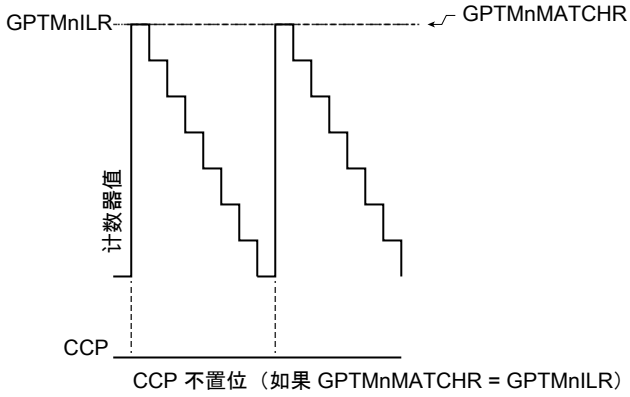
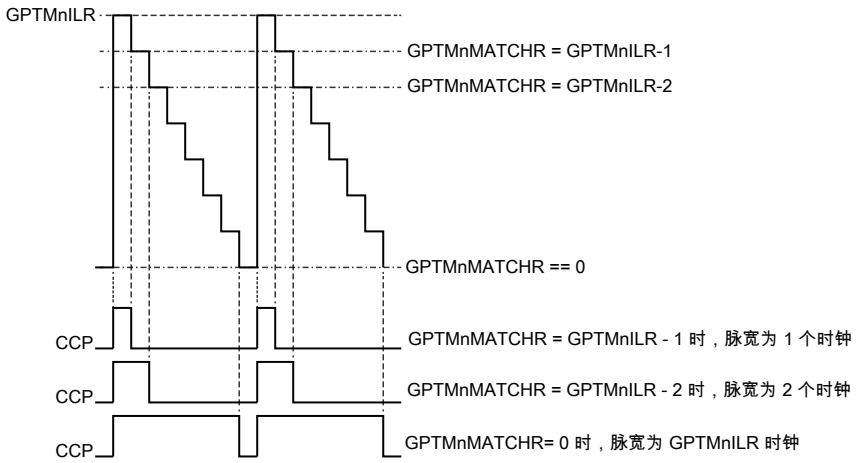


图11-8（642页）显示在 PLO 和 MRSU 位被置位并且 GPTMTnILR 大于 GPTMTnMATCHR 值时，CCP 输出如何操作。

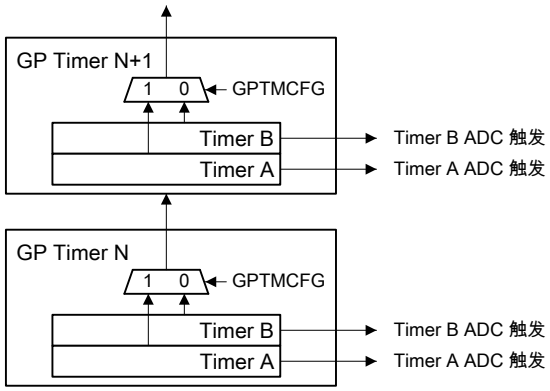
图 **11-8. CCP** 输出，**GPTMTnILR > GPTMTnMATCHR**



11.3.3 等待触发模式

等待触发模式，是允许使用菊花链式的定时器模块。比如，一旦配置好，一个单独的定时器可以通过定时器触发来初始化多路时钟事件。通过置位 GPTMTnMR 寄存器里的 TnWOT 位来启用等待触发模式。当 TnWOT 置位时，Timer N+1 只有在等到菊花链中它的上一个定时器 (Timer N) 发生超时事件时，才会开始计数。菊花链的配置一般是如下形式的：GPTM1跟着GPTM0，GPTM2跟着GPTM1，…等等。如果 Timer A 被配置为 32 位（16/32 位模式）或 64 位（32/64 位宽模式）定时器（通过 GPTMCFG 寄存器的 GPTMCFG 域控制），则触发下一模块的 Timer A。如果 Timer A 被配置为 16 位（16/32 位模式）或 32 位（32/64 位宽模式）定时器，则触发同一模块中的 Timer B，由 Timer B 触发下一模块的 Timer A。必须注意：GPTM0 的 TAWOT 位永远不会置位。图 11-9（642页） 显示了 GPTMCFG 位如何影响菊花链。这些功能对单次触发、周期和 PWM 模式都是有效的。

图 **11-9.** 定时器菊花链



11.3.4 同步通用定时器模块

GPTM0 模块中的 GPTM 同步控制 (GPTMSYNC) 寄存器可以用于同步所选的定时器，以便同时开始计数。将 GPTMSYNC 寄存器的某位置位将导致相关定时器执行超时事件的动作。在同步定时器时未产生中断。如果正在连接模式中使用定时器，仅 Timer A 的位必须在 GPTMSYNC 寄存器中置位。

注意: 要正常使用此功能，所有定时器必须设置相同的时钟源。

表11-11（643页）显示在各种定时器模式中同步定时器时所执行超时事件的动作。

表 **11-11. GPTM** 模式的超时动作

模式	计数目录	超时动作
32 和 64 位单次触发（连接定时器）	—	N/A
32 和 64 位周期（连接定时器）	递减	计数值 = ILR
	递增	计数值 = 0
32 和 64 位 RTC（连接定时器）	递增	计数值 = 0
16 和 32 位单次触发（独立/分离定时器）	—	N/A
16 和 32 位周期（独立/分离定时器）	递减	计数值 = ILR
	递增	计数值 = 0
16 和 32 位边沿计数（独立/分离定时器）	递减	计数值 = ILR
	递增	计数值 = 0
16 和 32 位边沿计时（独立/分离定时器）	递减	计数值 = ILR
	递增	计数值 = 0
16 和 32 位 PWM	递减	计数值 = ILR

11.3.5 DMA 操作

每一个定时器都有一个专用的 µDMA 通道，并且可以给 µDMA 控制器提供一个请求信号。该请求信号是突发类型的，并且在定时器原始中断发生时都会发生。µDMA 传输的仲裁数目应与每次定时器事件出现时应该传输的数据单元数相同。

例如，要传输256个项目，每10ms传输8个。配置一个定时器为周期定时器，每10ms发生一次中断。配置 µDMA 传输总量为 256 个项目，每次突发传送 8 个项目。每次定时器超时，µDMA 控制器就会传输 8 个项目，直到所有 256 个项目传输完毕。

不需要其他特殊步骤就能使定时器用于 µDMA 操作。关于给 µDMA 控制器编程的更多详情，请参阅 “微型直接存储器访问（µDMA）”（512页）。

11.3.6 访问连接的 16/32 位 GPTM 寄存器值

通过向 GPTM 配置 (GPTMCFG) 寄存器中的 GPTMCFG 位域写入 0x0 或 0x1，可将 GPTM 配置为连接模式。在这两种配置下，某些 16/32 位 GPTM 寄存器会连接在一起形成伪 32 位寄存器。这些寄存器包括：

- GPTM Timer A 间隔加载 (GPTMTAILR) 寄存器 [15:0]，见674页
- GPTM Timer B 间隔加载 (GPTMTBILR) 寄存器 [15:0]，见675页
- GPTM Timer A (GPTMTAR) 寄存器 [15:0]，见682页
- GPTM Timer B (GPTMTBR) 寄存器 [15:0]，见683页
- GPTM Timer A 值 (GPTMTAV) 寄存器 [15:0]，见684页
- GPTM Timer B 值 (GPTMTBV) 寄存器 [15:0]，见685页
- GPTM Timer A 匹配 (GPTMTAMATCHR) 寄存器 [15:0]，见676页

- GPTM Timer B 匹配 (GPTMTBMATCHR) 寄存器 [15:0]，见677页

在 32 位模式下，GPTM 把对 GPTMTAILR 的 32 位写访问转换为对 GPTMTAILR 和 GPTMTBILR 的写访问。这样，写操作最终的字顺序为：

```
GPTMTBILR[15:0]:GPTMTAILR[15:0]
```

同样，对 GPTMTAR 的 32 位读操作返回的值为：

```
GPTMTBR[15:0]:GPTMTAR[15:0]
```

对 GPTMTAV 的 32 位读操作返回的值为：

```
GPTMTBV[15:0]:GPTMTAV[15:0]
```

11.3.7 访问连接的 32/64 位宽 GPTM 寄存器值

在 32/64 位宽 GPTM 模块上，连接的寄存器值（64 位和 48 位）并不适合作为大于处理器内核的总线宽度访问的位宽。在连接定时器模式和独立的定时器模式中，在使用预分频器时，软件必须针对该值执行原子访问，以使之一致。在读取大于 32 位的定时器值时，软件应遵循以下步骤：

1. 读取合适的 Timer B 寄存器或预分频器寄存器。
2. 读取相应的 Timer A 寄存器。
3. 重新读取 Timer B 寄存器或预分频器寄存器。
4. 对比首次和第二次读取的 Timer B 或预分频器值。如果它们相同，定时器的值一致。如果它们不同，再次重复步骤 1-4，使它们相同。

以下伪代码说明了该过程：

```
high = timer_high;
low = timer_low;
if (high != timer_high);  //low overflowed into high
{
    high = timer_high;
    low = timer_low;
}
```

必须以此方法读取的寄存器如下所示：

- 64 位读取
  - GPTMTAV 和 GPTMTBV
  - GPTMTAR 和 GPTMTBR
- 48 位读取
  - GPTMTAR 和 GPTMTAPS

– GPTMTBR 和 GPTMTBPS

– GPTMTAV 和 GPTMTAPV

– GPTMTBV 和 GPTMTBPV

同样，还必须通过在写入低位之前写入高位来执行写操作，如下所示：

1. 写入合适的 Timer B 寄存器或预分频器寄存器。

2. 写入相应的 Timer A 寄存器。

必须以此方法写入的寄存器如下所示：

■ 64 位写入

– GPTMTAV 和 GPTMTBV

– GPTMTAMATCHR 和 GPTMTBMATCHR

– GPTMTAILR 和 GPTMTBILR

■ 48 位写入

– GPTMTAV 和 GPTMTAPV

– GPTMTBV 和 GPTMTBPV

– GPTMTAMATCHR 和 GPTMTAPMR

– GPTMTBMATCHR 和 GPTMTBPMR

– GPTMTAILR 和 GPTMTAPR

– GPTMTBILR 和 GPTMTBPR

在写入 64 位值时，如果连续两次写入到以上“64 位写入”标题下列出的任何寄存器中，不论寄存器是在 Timer A 或 Timer B 中，或者如果在写入 Timer B 中相应的寄存器之前写入寄存器 Timer A，那么系统将通过 GPTMRIS 寄存器中的 WUERIS 位报告错误。如果该错误未被屏蔽，即可引发中断。请注意，因为预分频器的使用是可选项，所以该错误并未报告给预分频器寄存器。因此，程序员必须小心遵循上文所述的协议。

11.4 初始化和配置

要使用 GPTM，必须将 RCGCTIMER或 RCGCWTIMER 寄存器中相应的 TIMERn 位置位（请参阅293页 和309页）。如果使用任何 CCP 管脚，必须通过 RCGCGPIO 寄存器启用相关 GPIO 模块的时钟（见295页）。要了解要启用哪一个 GPIO 端口，请参考表20-4（1019页）。在 GPIOPCTL 寄存器中配置 PMCn 域，以将 CCP 信号分配到合适的管脚（见612页和 表20-5（1024页））。

本节提供了所支持的定时器模式的初始化和配置的例子。

11.4.1 单次触发/周期定时器模式

将 GPTM 配置为单次触发和周期模式的步骤如下：

1. 确保先禁用定时器（将 GPTMCTL 寄存器的 TnEN 位清零），然后再进行更改操作。

2. 向 GPTM 配置 (GPTMCFG) 寄存器写入 0x0000.0000。
3. 配置 GPTM Timer n 模式寄存器 (GPTMTnMR) 的 TnMR 域：
   a. 写入 0x1 设为单次触发模式。
   b. 写入0x2 设为周期模式。
4. （可选）配置 GPTMTnMR 寄存器中的 TnSNAPS、TnWOT、TnMTE 和 TnCDIR 位，以选择是否捕获超时自由运行定时器的值、使用外部触发来启动计数、配置一个额外的触发或中断，以及递增还是递减计数。
5. 将初值加载到 GPTM Timer n 间隔加载 (GPTMTnILR) 寄存器。
6. 如果需要中断，将 GPTM 中断屏蔽 (GPTMIMR) 寄存器里相应的位置位。
7. 在 GPTMCTL 寄存器中将 TnEN 位置位来启用定时器并开始计数。
8. 查询 GPTMRIS 寄存器或等待产生中断（如果已启用）。在这两种情况下，通过向 GPTM 中断清除 (GPTMICR) 寄存器里相应的位写 1 来清除状态标记。

如果 GPTMTAMR 寄存器中的 TnMIE 位置位，GPTMRIS 寄存器中的 RTCRIS 位将置位，并且定时器继续计数。在单次触发模式中，定时器在到达超时事件时停止计数。要重新使能定时器需要重复上述步骤。在周期模式中，发生超时事件后定时器将会重新载入并继续计数。

11.4.2 实时时钟 (RTC) 模式

要使用32位实时时钟，定时器需要在CCP的偶数管脚上有一个32.768KHz的输入信号。要使用RTC请遵循如下步骤：

1. 在执行任何更改之前，先将定时器禁用（把 TAEN 位清零）。
2. 如果定时器在此之前以其他模式运行，则应先清除 GPTM 定时器 n 模式 (GPTMTnMR) 寄存器中的所有残余设置位，然后再重新配置。
3. 向 GPTM 配置 (GPTMCFG) 寄存器写入 0x0000.0001。
4. 向 GPTM Timer n 匹配寄存器 (GPTMTnMATCHR) 写入匹配值。
5. 根据需要将 GPTM 控制 (GPTMCTL) 寄存器的 RTCEN 位和 TnSTALL 位置位或清零。
6. 如果需要中断，将 GPTM 中断屏蔽 (GPTMIMR) 寄存器的 RTCIM 位置位。
7. 置位 GPTMCTL 寄存器的 TAEN 位来启用定时器并开始计数。

当定时器计数等于 GPTMTnMATCHR 寄存器中的值时，GPTM 确认 GPTMRIS 寄存器中的 RTCRIS 位，并且继续计数，直到 Timer A 被禁用或发生硬件复位。通过写 GPTMICR 寄存器的 RTCCINT 位来清除中断。请注意：如果 GPTMTnILR 寄存器加载了新值，则定时器将从该新值开始继续计数，并在达到 0xFFFF.FFFF 后返回初始值重新计数。

11.4.3 输入边沿计数模式下：

将定时器通过如下步骤配置为输入边沿计数模式：

1. 在执行任何更改之前，先将定时器禁用（把 TnEN 位清零）。

2. 向 GPTM 配置 (GPTMCFG) 寄存器写入 0x0000.0004。
3. 在 GPTM 定时器模式 (GPTMTnMR) 寄存器中，向 TnCMR 位域写入 0x0，向 TnMR 位域写入 0x3。
4. 在 GPTM 控制 (GPTMCTL) 寄存器里的 TnEVENT 位域写入相应的值来设置时钟捕获的事件类型。
5. 如果使用预分频器，向 GPTM Timer n 预分频 (GPTMTnPR) 寄存器中写入预分频值。
6. 给 GPTM Timer n 间隔加载 (GPTMTnILR) 寄存器加载定时器初值。
7. 向 GPTM Timer n 预分频匹配 (GPTMTnPMR) 寄存器加载预分频器匹配值（如有）。
8. 向 GPTM Timer n 匹配 (GPTMTnMATCHR) 寄存器加载事件计数。请注意：执行递增计数时，GPTMTnPR 和 GPTMTnILR 的值必须大于 GPTMTnPMR 和 GPTMTnMATCHR 的值。
9. 如果需要中断，将 GPTM 中断屏蔽 (GPTMIMR) 寄存器的 CnMIM 位置位。
10. 置位 GPTMCTL 寄存器的 TnEN 位可启用定时器并开始等待边沿事件。
11. 查询 GPTMRIS 寄存器的 CnMRIS 位，或者等待发生中断（如果已启用）。在这两个情况下，通过向 GPTM 中断清除 (GPTMICR) 寄存器中的 CnMCINT 位写入 1 来清除状态标志。

在输入边沿计数模式中递减计数时，定时器在检测到定义好的边沿事件数之后停止。需确保 TnEN 位清零并重复#4（647页） 到#9（647页） 才能重新启用定时器。

11.4.4 输入边沿定时模式

将定时器配置为输入边沿定时模式的步骤如下：

1. 在执行任何更改之前，先将定时器禁用（把 TnEN 位清零）。
2. 向 GPTM 配置 (GPTMCFG) 寄存器写入 0x0000.0004。
3. 在 GPTM 定时器模式 (GPTMTnMR) 寄存器中，向 TnCMR 位域写入 0x1，向 TnMR 位域写入 0x3。
4. 在 GPTM 控制 (GPTMCTL) 寄存器里的 TnEVENT 位域写入相应的值来设置时钟捕获的事件类型。
5. 如果使用预分频器，向 GPTM Timer n 预分频 (GPTMTnPR) 寄存器中写入预分频值。
6. 给 GPTM Timer n 间隔加载 (GPTMTnILR) 寄存器加载定时器初值。
7. 如果需要中断，将 GPTM 中断屏蔽 (GPTMIMR) 寄存器的 CnEIM 位置位。
8. 将 GPTM 控制 (GPTMCTL) 寄存器中的 TnEN 位置位可启用定时器并开始计数。
9. 查询 GPTMRIS 寄存器中的 CnERIS 位或等待发生中断（如果已启用）。在这两种情况下，通过向 GPTM 中断清除 (GPTMICR) 寄存器中的 CnECINT 位写入 1 来清除状态标志。事件发生的时间可以通过读取 GPTM Timer n (GPTMTnR) 寄存器来获得。

在输入边沿定时模式下，定时器在检测到边沿事件之后继续运行，但通过写 GPTMTnILR 寄存器可在任何时候改变定时器间隔。此改变在写操作的下一个周期生效。

11.4.5 PWM 模式

定时器可以通过以下步骤配置为 PWM 模式：

1. 在执行任何更改之前，先将定时器禁用（把 TnEN 位清零）。
2. 向 GPTM 配置 (GPTMCFG) 寄存器写入 0x0000.0004。
3. 在 GPTM 定时器模式 (GPTMTnMR) 寄存器中，将 TnAMS 位设为 0x1，将 TnCMR 位设为 0x0，将 TnMR 域设为 0x2。
4. 在 GPTM 控制 (GPTMCTL) 寄存器的 TnEVENT 位域中，配置 PWM 信号的输出状态（是否需要反相）。
5. 如果使用预分频器，向 GPTM Timer n 预分频 (GPTMTnPR) 寄存器中写入预分频值。
6. 如果使用 PWM 中断，配置 GPTMCTL 寄存器中 TnEVENT 域的中断条件，并且通过将 GPTMTnMR 寄存器中的 TnPWMIE 位置位来启用中断。注：PWM 输出反相时，将翻转边沿检测中断行为（参考 657页）。
7. 给 GPTM Timer n 间隔加载 (GPTMTnILR) 寄存器加载定时器初值。
8. 向 GPTM Timer n 匹配 (GPTMTnMATCHR) 寄存器加载匹配值。
9. 置位 GPTM 控制 (GPTMCTL) 寄存器的 TnEN 位可启用定时器并开始输出 PWM 信号。

在PWM模式下，定时器在产生PWM信号之后继续运行。通过写 GPTMTnILR 寄存器可在任何时候对 PWM 周期进行调整，此改变在写操作的下一个周期生效。

11.5 寄存器映射

表 11-12（649页）列出了 GPTM 寄存器。下表中列出的“偏移量”代表寄存器相对于该定时器基址的十六进制地址增量：

- 16/32 位Timer 0：0x4003.0000
- 16/32 位Timer 1：0x4003.1000
- 16/32 位Timer 2：0x4003.2000
- 16/32 位Timer 3：0x4003.3000
- 16/32 位Timer 4：0x4003.4000
- 16/32 位Timer 5：0x4003.5000
- 32/64 位宽 Timer 0：0x4003.6000
- 32/64 位宽 Timer 1：0x4003.7000
- 32/64 位宽 Timer 2：0x4004.C000
- 32/64 位宽 Timer 3：0x4004.D000
- 32/64 位宽 Timer 4：0x4004.E000
- 32/64 位宽 Timer 5：0x4004.F000

GPTM 外设属性 (GPTMPP) 寄存器中的 SIZE 域可识别模块是具有 16/32 位还是 32/64 位宽定时器。

请注意，在给寄存器编程前必须启用通用定时器模块时钟（请参阅293页 或309页）。在访问任何定时器模块寄存器前，启用定时器模块时钟后必须有 3 个系统时钟的延迟。

表 11-12. 定时器触发 寄存器映射

偏移量	名称	类型	复位	描述	见页面
0x000	GPTMCFG	R/W	0x0000.0000	GPTM 配置寄存器	650
0x004	GPTMTAMR	R/W	0x0000.0000	GPTM Timer A 模式寄存器	651
0x008	GPTMTBMR	R/W	0x0000.0000	GPTM Timer B 模式寄存器	654
0x00C	GPTMCTL	R/W	0x0000.0000	GPTM 控制寄存器	657
0x010	GPTMSYNC	R/W	0x0000.0000	GPTM 同步寄存器	660
0x018	GPTMIMR	R/W	0x0000.0000	GPTM 中断屏蔽寄存器	663
0x01C	GPTMRIS	RO	0x0000.0000	GPTM 原始中断状态寄存器	666
0x020	GPTMMIS	RO	0x0000.0000	GPTM 屏蔽的中断状态寄存器	669
0x024	GPTMICR	W1C	0x0000.0000	GPTM 中断清除寄存器	672
0x028	GPTMTAILR	R/W	0xFFFF.FFFF	GPTM Timer A 间隔加载寄存器	674
0x02C	GPTMTBILR	R/W	-	GPTM Timer B 间隔加载寄存器	675
0x030	GPTMTAMATCHR	R/W	0xFFFF.FFFF	GPTM Timer A 匹配寄存器	676
0x034	GPTMTBMATCHR	R/W	-	GPTM Timer B 匹配寄存器	677
0x038	GPTMTAPR	R/W	0x0000.0000	GPTM Timer A 预分频寄存器	678
0x03C	GPTMTBPR	R/W	0x0000.0000	GPTM Timer B 预分频寄存器	679
0x040	GPTMTAPMR	R/W	0x0000.0000	GPTM TimerA 预分频匹配寄存器	680
0x044	GPTMTBPMR	R/W	0x0000.0000	GPTM TimerB 预分频匹配寄存器	681
0x048	GPTMTAPR	RO	0xFFFF.FFFF	GPTM Timer A 寄存器	682
0x04C	GPTMTBPR	RO	-	GPTM Timer B 寄存器	683
0x050	GPTMTAV	RW	0xFFFF.FFFF	GPTM Timer A 值寄存器	684
0x054	GPTMTBV	RW	-	GPTM Timer B 值寄存器	685
0x058	GPTMRTCPD	RO	0x0000.7FFF	GPTM RTC 预分频寄存器	686
0x05C	GPTMTAPS	RO	0x0000.0000	GPTM Timer A 预分频快照寄存器	687
0x060	GPTMTBPS	RO	0x0000.0000	GPTM Timer B 预分频快照寄存器	688
0x064	GPTMTAPV	RO	0x0000.0000	GPTM Timer A 预分频值寄存器	689
0x068	GPTMTBPV	RO	0x0000.0000	GPTM Timer B 预分频值寄存器	690
0xFC0	GPTMPP	RO	0x0000.0000	GPTM 外设属性寄存器	691

11.6 寄存器描述

本章的其余部分，按照地址的偏移量递增的顺序描述了GPTM的寄存器。

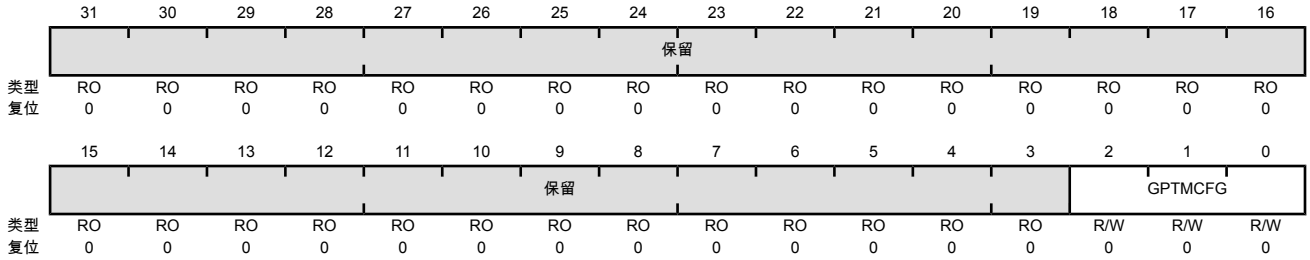
寄存器 1: GPTM 配置寄存器（GPTMCFG），偏移量 0x000

该寄存器对 GPTM 模块的全局操作进行配置。写入该寄存器的值决定 GPTM 处于 32 或 64 位模式（连接定时器）还是处于 16 或 32 位模式（独立、分离定时器）。

重要: 该寄存器中的位应仅在 GPTMCTL 寄存器中的 TAEN 和 TBEN 位清零时发生更改。

GPTM 配置寄存器 (GPTMCFG)

16/32 位Timer 0 基址: 0x4003.0000
16/32 位Timer 1 基址: 0x4003.1000
16/32 位Timer 2 基址: 0x4003.2000
16/32 位Timer 3 基址: 0x4003.3000
16/32 位Timer 4 基址: 0x4003.4000
16/32 位Timer 5 基址: 0x4003.5000
32/64 位宽 Timer 0 基址: 0x4003.6000
32/64 位宽 Timer 1 基址: 0x4003.7000
32/64 位宽 Timer 2 基址: 0x4004.C000
32/64 位宽 Timer 3 基址: 0x4004.D000
32/64 位宽 Timer 4 基址: 0x4004.E000
32/64 位宽 Timer 5 基址: 0x4004.F000
偏移量 0x000
类型 R/W, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:3	保留	RO	0x0000.000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
2:0	GPTMCFG	R/W	0x0	GPTM配置 GPTMCFG 值的定义如下： 值 描述 0x0 对于 16/32 位定时器，该值选择 32 位定时器配置。 对于 32/64 位宽定时器，该值选择 64 位定时器配置。 0x1 对于 16/32 位定时器，该值选择 32 位实时时钟 (RTC) 计数器配置。 对于 32/64 位宽定时器，该值选择 64 位实时时钟 (RTC) 计数器配置。 0x2-0x3 保留 0x4 对于 16/32 位定时器，该值选择 16 位定时器配置。 对于 32/64 位宽定时器，该值选择 32 位定时器配置。 功能由 GPTMTAMR 和 GPTMTBMR 的位 1:0 来控制。 0x5-0x7 保留

寄存器 2: GPTM Timer A 模式寄存器（GPTMTAMR），偏移量 0x004

该寄存器根据 GPTMCFG 寄存器的配置来进一步的配置 GPTM。在 PWM 模式中，将 TAAMS 位置位，将 TACMR 位清零，并把 TAMR 域配置为 0x1 或 0x2。

该寄存器在 Timer A 独立使用时控制它的模式。在 Timer A 和 Timer B 连接在一起时，该寄存器控制 Timer A 和 Timer B 的模式，并忽略 GPTMTBMR 的内容。

重要: 该寄存器中的位应仅可在 GPTMCTL 寄存器中的 TAEN 位清零时发生更改。

GPTM Timer A 模式寄存器 (GPTMTAMR)

16/32 位Timer 0 基址: 0x4003.0000
16/32 位Timer 1 基址: 0x4003.1000
16/32 位Timer 2 基址: 0x4003.2000
16/32 位Timer 3 基址: 0x4003.3000
16/32 位Timer 4 基址: 0x4003.4000
16/32 位Timer 5 基址: 0x4003.5000
32/64 位宽 Timer 0 基址: 0x4003.6000
32/64 位宽 Timer 1 基址: 0x4003.7000
32/64 位宽 Timer 2 基址: 0x4004.C000
32/64 位宽 Timer 3 基址: 0x4004.D000
32/64 位宽 Timer 4 基址: 0x4004.E000
32/64 位宽 Timer 5 基址: 0x4004.F000
偏移量 0x004
类型 R/W, 复位 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	保留															
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	保留				TAPLO	TAMRSU	TAPWMIE	TAILD	TASNAPS	TAWOT	TAMIE	TACDIR	TAAMS	TACMR	TAMR	
类型	RO	RO	RO	RO	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位/域	名称	类型	复位	描述
31:12	保留	RO	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
11	TAPLO	R/W	0	GPTM Timer A PWM 传统操作 值 描述 0 定时器达到 0 后重新载入 GPTMTAILR 时，执行 CCP 管脚拉低的传统操作。 1 定时器达到 0 后重新载入 GPTMTAILR 时，CCP 拉高。 该位仅在 PWM 模式中有效。
10	TAMRSU	R/W	0	GPTM Timer A 匹配寄存器更新 值 描述 0 在下一个时钟周期更新 GPTMTAMATCHR 寄存器和 GPTMTAPR 寄存器（如使用）。 1 在下一次超时时更新 GPTMTAMATCHR 寄存器和 GPTMTAPR 寄存器（如使用）。 当该位被置位时，如果定时器被禁用（TAEN 清零），则定时器启用时，GPTMTAMATCHR 和 GPTMTAPR 将更新。如果定时器暂停（TASTALL 置位），GPTMTAMATCHR 和 GPTMTAPR 将根据该位的配置更新。

位/域	名称	类型	复位	描述
9	TAPWMIE	R/W	0	GPTM Timer A PWM 中断启用 根据 GPTMCTL 寄存器中 TAEVENT 域的定义，在 PWM 模式中，该位启用在上升沿、下降沿或 CCP 输出的双边沿中断功能。 值 描述 0 捕获事件中断已禁用。 1 捕获事件中断已启用。 该位仅在 PWM 模式中有效。
8	TAILD	R/W	0	GPTM Timer A 间隔加载写操作 值 描述 0 在下一个周期根据 GPTMTAILR 寄存器中的值更新 GPTMTAR 和 GPTMTAV 寄存器。同时在下一个周期根据 GPTMTAPR 寄存器的值更新 GPTMTAPS和 GPTMTAPV 寄存器。 1 在下一个周期根据 GPTMTAILR 寄存器中的值更新 GPTMTAR 和 GPTMTAV 寄存器。同时在下一次超时时根据 GPTMTAPR 寄存器的值更新 GPTMTAPS和 GPTMTAPV 寄存器。 请注意该位的状态在递增计数时没有影响。 定时器已启用并正在运行时，适用上面的位描述。当该位被置位时，如果定时器被禁用（TAEN 清零），则定时器启用时，GPTMTAR、GPTMTAV、GPTMTAPs、和 GPTMTAPV 将更新。如果定时器暂停（TASTALL 置位），GPTMTAR 和 GPTMTAPS 将根据该位的配置更新。
7	TASNAPS	R/W	0	GPTM Timer A 快照模式 值 描述 0 禁用快照模式 1 如果 Timer A 被配置为周期模式，Timer A 的实际自由运行、捕获或者快照值将在发生超时、捕获或者快照事件时加载到GPTM Timer A (GPTMTAR) 寄存器。如果使用了定时器预分频器，将把预分频器快照载入 GPTM Timer A (GPTMTAPR)。
6	TAWOT	R/W	0	GPTM Timer A 等待触发 值 描述 0 一旦使能马上开始计数 1 如果 Timer A 启用（GPTMCTL 寄存器中的 TAEN 置位），Timer A 不会马上开始计数，直到它从菊花链中上一个位置的定时器收到一个触发信号，请参阅图11-9（642页）。这些功能对单次触发、周期和 PWM 模式都是有效的。 GPTM 0 Timer A的该位必须清零
5	TAMIE	R/W	0	GPTM Timer A 匹配中断启用 值 描述 0 匹配事件的匹配中断已禁用。 1 在单次触发或者周期模式下，当达到 GPTMTAMATCHR 寄存器中的匹配值时，就会产生中断。

位/域	名称	类型	复位	描述
4	TACDIR	R/W	0	GPTM Timer A 计数方向 值 描述 0 向下递减计数 1 定时器递增计数。在递增计数时，定时器从 0x0 开始。 在 PWM 或 RTC 模式中时，忽略该位的状态。PWM 模式始终递减计数，而 RTC 模式始终递增计数。
3	TAAMS	R/W	0	GPTM Timer A 交替模式选择 TAAMS 的值定义如下： 值 描述 0 捕获或者对比模式已启用。 1 使能PWM模式 注意: 要启用 PWM 模式，必须将 TACMR 位清零并将 TAMR 域配置为 0x1 或 0x2。
2	TACMR	R/W	0	GPTM Timer A 捕获模式 TACMR 的值定义如下： 值 描述 0 边沿计数模式 1 边沿定时模式
1:0	TAMR	R/W	0x0	GTPM Timer A 模式寄存器 TAMR 的值定义如下： 值 描述 0x0 保留 0x1 单次触发定时器模式 0x2 周期定时器模式 0x3 捕获模式 定时器模式基于 GPTMCFG 寄存器 2:0 位定义的定时器配置。

寄存器 3: GPTM Timer B 模式寄存器（GPTMTBMR），偏移量 0x008

该寄存器根据 GPTMCFG 寄存器的配置来进一步的配置 GPTM。在 PWM 模式中，将 TBAMS 位置位，将 TBCMR 位清零，并配置 TBMR 域为 0x1 或 0x2。

该寄存器在 Timer B 独立使用时控制它的模式。在 Timer A 和 Timer B 连接时，该寄存器将被忽略，并且 GPTMTAMR 控制 Timer A 和 Timer B 的模式。

重要: 该寄存器中的位应仅可在 GPTMCTL 寄存器中的 TBEN 位清零时发生更改。

GPTM Timer B 模式寄存器 (GPTMTBMR)

16/32 位Timer 0 基址: 0x4003.0000
16/32 位Timer 1 基址: 0x4003.1000
16/32 位Timer 2 基址: 0x4003.2000
16/32 位Timer 3 基址: 0x4003.3000
16/32 位Timer 4 基址: 0x4003.4000
16/32 位Timer 5 基址: 0x4003.5000
32/64 位宽 Timer 0 基址: 0x4003.6000
32/64 位宽 Timer 1 基址: 0x4003.7000
32/64 位宽 Timer 2 基址: 0x4004.C000
32/64 位宽 Timer 3 基址: 0x4004.D000
32/64 位宽 Timer 4 基址: 0x4004.E000
32/64 位宽 Timer 5 基址: 0x4004.F000
偏移量 0x008
类型 R/W, 复位 0x0000.0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
保留															
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留				TBPLO	TBMRSU	TBPWMIE	TBILD	TBSNAPS	TBWOT	TBMIE	TBCDIR	TBAMS	TBCMR	TBMR	
类型	RO	RO	RO	RO	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位/域	名称	类型	复位	描述
31:12	保留	RO	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
11	TBPLO	R/W	0	GPTM Timer B PWM 传统操作 值 描述 0 定时器达到 0 后重新载入 GPTMTAILR 时，执行 CCP 管脚拉低的传统操作。 1 定时器达到 0 后重新载入 GPTMTAILR 时，CCP 拉高。 该位仅在 PWM 模式中有效。
10	TBMRSU	R/W	0	GPTM Timer B 匹配寄存器更新 值 描述 0 在下一个时钟周期更新 GPTMTBMATCHR 寄存器和 GPTMTBPR 寄存器（如使用）。 1 在下一次超时时更新 GPTMTBMATCHR 寄存器和 GPTMTBPR 寄存器（如使用）。 当该位被置位时，如果定时器被禁用（TBEN 清零），则定时器启用时，GPTMTBMATCHR 和 GPTMTBPR 将更新。如果定时器暂停（TBSTALL 置位），GPTMTBMATCHR 和 GPTMTBPR 将根据该位的配置更新。

位/域	名称	类型	复位	描述
9	TBPWMIE	R/W	0	GPTM Timer B PWM 中断启用 根据 GPTMCTL 寄存器中 TBEVENT 域的定义，在 PWM 模式中，该位启用在上升沿、下降沿或 CCP 输出的双边沿中断功能。 值 描述 0 捕获事件中断已禁用。 1 捕获事件已启用。 该位仅在 PWM 模式中有效。
8	TBILD	R/W	0	GPTM Timer B 间隔加载写操作 值 描述 0 在下一个周期根据 GPTMTBILR 寄存器中的值更新 GPTMTBR 和 GPTMTBV 寄存器。同时在下一个周期根据 GPTMTBPR 寄存器的值更新 GPTMTBPS和 GPTMTBPV 寄存器。 1 在下一个周期根据 GPTMTBILR 寄存器中的值更新 GPTMTBR 和 GPTMTBV 寄存器。同时在下一次超时时根据 GPTMTBPR 寄存器的值更新 GPTMTBPS和 GPTMTBPV 寄存器。 请注意该位的状态在递增计数时没有影响。 定时器已启用并正在运行时，适用上面的位描述。当该位被置位时，如果定时器被禁用（TBEN 清零），则定时器启用时，GPTMTBR、GPTMTBV、GPTMTBPS 和 GPTMTBPV 将更新。如果定时器暂停（TBSTALL 置位），GPTMTBR 和 GPTMTBPS 将根据该位的配置更新。
7	TBSNAPS	R/W	0	GPTM Timer B 快照模式 值 描述 0 禁用快照模式 1 如果 Timer B 被配置为周期模式，Timer B 实际自由运行的值将在发生超时事件时加载到 GPTM Timer B (GPTMTBR) 寄存器。如果使用了定时器预分频器，将把预分频器快照载入 GPTM Timer B (GPTMTBPR)。
6	TBWOT	R/W	0	GPTM 定时器 B 等待触发 值 描述 0 Timer B 一启用就开始计数。 1 如果 Timer B 启用（GPTMCTL 寄存器中的 TBEN 置位），Timer B 不会马上开始计数，直到它从菊花链中上一个位置的定时器收到一个触发信号，请参阅图11-9（642页）。这些功能对单次触发、周期和 PWM 模式都是有效的。
5	TBMIE	R/W	0	GPTM Timer B 匹配中断启用 值 描述 0 匹配事件的匹配中断已禁用。 1 在单次触发或者周期模式下，当达到 GPTMTBMATCHR 寄存器中的匹配值时，就会产生中断。

位/域	名称	类型	复位	描述
4	TBCDIR	R/W	0	GPTM Timer B 计数方向 值 描述 0 向下递减计数 1 定时器递增计数。在递增计数时，定时器从 0x0 开始。 在 PWM 或 RTC 模式中时，忽略该位的状态。PWM 模式始终递减计数，而 RTC 模式始终递增计数。
3	TBAMS	R/W	0	GPTM Timer B 交替模式选择 TBAMS 的值定义如下： 值 描述 0 捕获或者对比模式已启用。 1 使能PWM模式 注意: 要启用 PWM 模式，必须将 TBCMR 位清零，并将 TBMR 域配置为 0x1 或 0x2。
2	TBCMR	R/W	0	GPTM Timer B 捕获模式 TBCMR 的值定义如下： 值 描述 0 边沿计数模式 1 边沿定时模式
1:0	TBMR	R/W	0x0	GTPM Timer B 模式寄存器 TBMR 的值定义如下： 值 描述 0x0 保留 0x1 单次触发定时器模式 0x2 周期定时器模式 0x3 捕获模式 定时器模式基于 GPTMCFG 寄存器 2:0 位定义的定时器配置。

寄存器 4: GPTM 控制寄存器（GPTMCTL），偏移量 0x00C

该寄存器与 GPTMCFG 和 GMTMTnMR 寄存器一起使用，用于对定时器配置进行微调和启用其他功能，比如定时器停止和输出触发。输出触发器可以用来启动ADC模块上的传输。

重要: 该寄存器中的位应仅在相应定时器的 TnEN 位清零时发生更改。

GPTM 控制寄存器 (GPTMCTL)

16/32 位Timer 0 基址: 0x4003.0000
16/32 位Timer 1 基址: 0x4003.1000
16/32 位Timer 2 基址: 0x4003.2000
16/32 位Timer 3 基址: 0x4003.3000
16/32 位Timer 4 基址: 0x4003.4000
16/32 位Timer 5 基址: 0x4003.5000
32/64 位宽 Timer 0 基址: 0x4003.6000
32/64 位宽 Timer 1 基址: 0x4003.7000
32/64 位宽 Timer 2 基址: 0x4004.C000
32/64 位宽 Timer 3 基址: 0x4004.D000
32/64 位宽 Timer 4 基址: 0x4004.E000
32/64 位宽 Timer 5 基址: 0x4004.F000
偏移量 0x00C
类型 R/W, 复位 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	保留															
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	保留	TBPWML	TBOTE	保留	TBEVENT		TBSTALL	TBEN	保留	TAPWML	TAOTE	RTCEN	TAEVENT		TASTALL	TAEN
类型	RO	R/W	R/W	RO	R/W	R/W	R/W	R/W	RO	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位/域	名称	类型	复位	描述
31:15	保留	RO	0x0000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
14	TBPWML	R/W	0	GPTM Timer B PWM 输出水平 TBPWML 的值定义如下： 值 描述 0 输出不受影响 1 输出反向
13	TBOTE	R/W	0	GPTM Timer B输出触发使能 TBOTE 值定义如下： 值 描述 0 Timer B的ADC触发输出被禁止 1 Timer B的ADC触发输出被使能 注意: 要让 ADC 触发信号有效，必须将定时器配置为单次触发或者周期超时模式。GPTM 并不会为匹配事件、对比事件或者对比匹配事件产生触发信号。 此外，必须用 ADCEMUX 寄存器中的 EMn 位启用 ADC，并且选择定时器为触发源（见748页）。
12	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。

位/域	名称	类型	复位	描述
11:10	TBEVENT	R/W	0x0	GPTM Timer B事件模式 TBEVENT 的值定义如下： 值 描述 0x0 上升沿 0x1 下降沿 0x2 保留 0x3 双边沿 注意: 如果启用了 PWM 输出反相，将翻转边沿检测中断行为。因此，如果上升沿中断触发已置位，且PWM反相生成上升沿，将不存在有效的事件触发中断。相反，中断在PWM信号的下降沿生成。
9	TBSTALL	R/W	0	GPTM Timer B停止使能 TBSTALL 的值定义如下： 值 描述 0 当处理器被调试挂起时，Timer B继续计数 1 当处理器被调试挂起时，Timer B停止计数 当处理器正常运行的时候，TBSTALL 位将被忽略。
8	TBEN	R/W	0	GPTM Timer B使能 TBEN 的值定义如下： 值 描述 0 Timer B是禁止的 1 根据 GPTMCFG 寄存器，启用 Timer B 并开始计数或启用捕获逻辑。
7	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
6	TAPWML	R/W	0	GPTM Timer A的PWM输出电平 TAPWML 的值定义如下： 值 描述 0 输出不受影响 1 输出反向

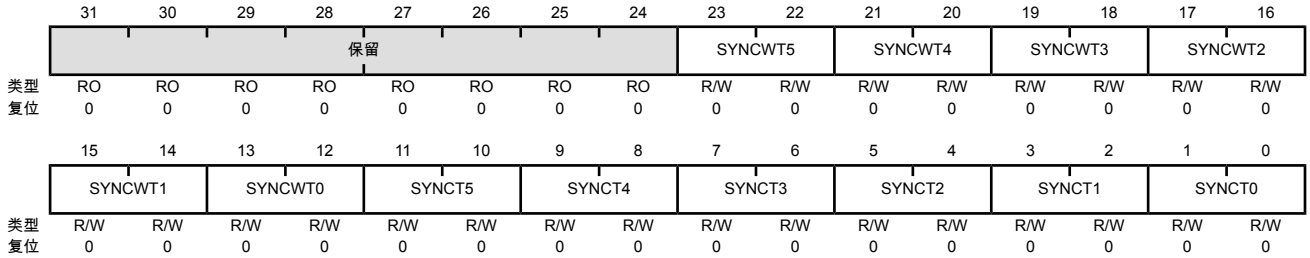
位/域	名称	类型	复位	描述
5	TAOTE	R/W	0	GPTM Timer A输出触发使能 TAOTE 值定义如下： 值 描述 0 Timer A的ADC触发输出被禁止 1 Timer A的ADC触发输出被使能 注意: 要让 ADC 触发信号有效，必须将定时器配置为单次触发或者周期超时模式。GPTM 并不会为匹配事件、对比事件或者对比匹配事件产生触发信号。 此外，必须用 ADCEMUX 寄存器中的 EMn 位启用 ADC，并且选择定时器为触发源（见748页）。
4	RTCEN	R/W	0	GPTM RTC 停止启用 RTCEN 的值定义如下： 值 描述 0 当处理器被调试器停止时，RTC 停止计数。 1 当处理器被调试器停止时，RTC 继续计数。 置位后，RTCEN 位可防止所有操作模式下的定时器停止运行，即使 TnSTALL 已置位。
3:2	TAEVENT	R/W	0x0	GPTM Timer A事件模式 TAEVENT 值定义如下： 值 描述 0x0 上升沿 0x1 下降沿 0x2 保留 0x3 双边沿 注意: 如果启用了 PWM 输出反相，将翻转边沿检测中断行为。因此，如果上升沿中断触发已置位，且PWM反相生成上升沿，将不存在有效的事件触发中断。相反，中断在PWM信号的下降沿生成。
1	TASTALL	R/W	0	GPTM Timer A停止使能 TASTALL 的值定义如下： 值 描述 0 当处理器被调试挂起时，Timer A继续计数 1 当处理器被调试挂起时，Timer A停止计数 当处理器正常运行的时候，TASTALL 位将被忽略。
0	TAEN	R/W	0	GPTM Timer A使能 TAEN 的值定义如下： 值 描述 0 Timer A是禁止的 1 根据 GPTMCFG 寄存器，Timer A 已启用并开始计数，或捕获逻辑已启用。

寄存器 5: GPTM 同步寄存器（GPTMSYNC），偏移量 0x010

注意: 仅在 GPTM 模块 0 上执行该寄存器。

该寄存器允许软件同步多个定时器。

GPTM 同步寄存器 (GPTMSYNC)
16/32 位Timer 0 基址: 0x4003.0000
16/32 位Timer 1 基址: 0x4003.1000
16/32 位Timer 2 基址: 0x4003.2000
16/32 位Timer 3 基址: 0x4003.3000
16/32 位Timer 4 基址: 0x4003.4000
16/32 位Timer 5 基址: 0x4003.5000
32/64 位宽 Timer 0 基址: 0x4003.6000
32/64 位宽 Timer 1 基址: 0x4003.7000
32/64 位宽 Timer 2 基址: 0x4004.C000
32/64 位宽 Timer 3 基址: 0x4004.D000
32/64 位宽 Timer 4 基址: 0x4004.E000
32/64 位宽 Timer 5 基址: 0x4004.F000
偏移量 0x010
类型 R/W, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:24	保留	RO	0x00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
23:22	SYNCWT5	R/W	0x0	同步 GPTM 32/64 位 Timer 5 SYNCWT5 值定义如下： 值 描述 0x0 GPTM 32/64 位 Timer 5 未受影响。 0x1 触发 GPTM 32/64 位 Timer 5 的 Timer A 超时事件。 0x2 触发 GPTM 32/64 位 Timer 5 的 Timer B 超时事件。 0x3 触发 GPTM 32/64 位 Timer 5 的 Timer A 和 Timer B 超时事件。
21:20	SYNCWT4	R/W	0x0	同步 GPTM 32/64 位 Timer 4 SYNCWT4 值定义如下： 值 描述 0x0 GPTM 32/64 位 Timer 4 未受影响。 0x1 触发 GPTM 32/64 位 Timer 4 的 Timer A 超时事件。 0x2 触发 GPTM 32/64 位 Timer 4 的 Timer B 超时事件。 0x3 触发 GPTM 32/64 位 Timer 4 的 Timer A 和 Timer B 超时事件。

位/域	名称	类型	复位	描述
19:18	SYNCWT3	R/W	0x0	同步 GPTM 32/64 位 Timer 3 SYNCWT3 值定义如下： 值 描述 0x0 GPTM 32/64 位 Timer 3 未受影响。 0x1 触发 GPTM 32/64 位 Timer 3 的 Timer A 超时事件。 0x2 触发 GPTM 32/64 位 Timer 3 的 Timer B 超时事件。 0x3 触发 GPTM 32/64 位 Timer 3 的 Timer A 和 Timer B 超时事件。
17:16	SYNCWT2	R/W	0x0	同步 GPTM 32/64 位 Timer 2 SYNCWT2 值定义如下： 值 描述 0x0 GPTM 32/64 位 Timer 2 未受影响。 0x1 触发 GPTM 32/64 位 Timer 2 的 Timer A 超时事件。 0x2 触发 GPTM 32/64 位 Timer 2 的 Timer B 超时事件。 0x3 触发 GPTM 32/64 位 Timer 2 的 Timer A 和 Timer B 超时事件。
15:14	SYNCWT1	R/W	0x0	同步 GPTM 32/64 位 Timer 1 SYNCWT1 值定义如下： 值 描述 0x0 GPTM 32/64 位 Timer 1 未受影响。 0x1 触发 GPTM 32/64 位 Timer 1 的 Timer A 超时事件。 0x2 触发 GPTM 32/64 位 Timer 1 的 Timer B 超时事件。 0x3 触发 GPTM 32/64 位 Timer 1 的 Timer A 和 Timer B 超时事件。
13:12	SYNCWT0	R/W	0x0	同步 GPTM 32/64 位 Timer 0 SYNCWT0 值定义如下： 值 描述 0x0 GPTM 32/64 位 Timer 0 未受影响。 0x1 触发 GPTM 32/64 位 Timer 0 的 Timer A 超时事件。 0x2 触发 GPTM 32/64 位 Timer 0 的 Timer B 超时事件。 0x3 触发 GPTM 32/64 位 Timer 0 的 Timer A 和 Timer B 超时事件。
11:10	SYNCT5	R/W	0x0	同步 GPTM 16/32 位 Timer 5 SYNCT5 值定义如下： 值 描述 0x0 GPTM 16/32 位 Timer 5 未受影响。 0x1 触发 GPTM 16/32 位 Timer 5 的 Timer A 超时事件。 0x2 触发 GPTM 16/32 位 Timer 5 的 Timer B 超时事件。 0x3 触发 GPTM 16/32 位 Timer 5 的 Timer A 和 Timer B 超时事件。

位/域	名称	类型	复位	描述
9:8	SYNCT4	R/W	0x0	同步 GPTM 16/32 位 Timer 4 SYNCT4 值定义如下： 值 描述 0x0 GPTM 16/32 位 Timer 4 未受影响。 0x1 触发 GPTM 16/32 位 Timer 4 的 Timer A 超时事件。 0x2 触发 GPTM 16/32 位 Timer 4 的 Timer B 超时事件。 0x3 触发 GPTM 16/32 位 Timer 4 的 Timer A 和 Timer B 超时事件。
7:6	SYNCT3	R/W	0x0	同步 GPTM 16/32 位 Timer 3 SYNCT3 值定义如下： 值 描述 0x0 GPTM 16/32 位 Timer 3 未受影响。 0x1 触发 GPTM 16/32 位 Timer 3 的 Timer A 超时事件。 0x2 触发 GPTM 16/32 位 Timer 3 的 Timer B 超时事件。 0x3 触发 GPTM 16/32 位 Timer 3 的 Timer A 和 Timer B 超时事件。
5:4	SYNCT2	R/W	0x0	同步 GPTM 16/32 位 Timer 2 SYNCT2 值定义如下： 值 描述 0x0 GPTM 16/32 位 Timer 2 未受影响。 0x1 触发 GPTM 16/32 位 Timer 2 的 Timer A 超时事件。 0x2 触发 GPTM 16/32 位 Timer 2 的 Timer B 超时事件。 0x3 触发 GPTM 16/32 位 Timer 2 的 Timer A 和 Timer B 超时事件。
3:2	SYNCT1	R/W	0x0	同步 GPTM 16/32 位 Timer 1 SYNCT1 值定义如下： 值 描述 0x0 GPTM 16/32 位 Timer 1 未受影响。 0x1 触发 GPTM 16/32 位 Timer 1 的 Timer A 超时事件。 0x2 触发 GPTM 16/32 位 Timer 1 的 Timer B 超时事件。 0x3 触发 GPTM 16/32 位 Timer 1 的 Timer A 和 Timer B 超时事件。
1:0	SYNCT0	R/W	0x0	同步 GPTM 16/32 位 Timer 0 SYNCT0 值定义如下： 值 描述 0x0 GPTM 16/32 位 Timer 0 未受影响。 0x1 触发 GPTM 16/32 位 Timer 0 的 Timer A 超时事件。 0x2 触发 GPTM 16/32 位 Timer 0 的 Timer B 超时事件。 0x3 触发 GPTM 16/32 位 Timer 0 的 Timer A 和 Timer B 超时事件。

寄存器 6: GPTM 中断屏蔽寄存器（GPTMIMR），偏移量 0x018

该寄存器允许软件启用/禁用 GPTM 控制器级中断。置位时启用相应中断，清零时禁用相应中断。

GPTM 中断屏蔽寄存器 (GPTMIMR)

16/32 位Timer 0 基址: 0x4003.0000
16/32 位Timer 1 基址: 0x4003.1000
16/32 位Timer 2 基址: 0x4003.2000
16/32 位Timer 3 基址: 0x4003.3000
16/32 位Timer 4 基址: 0x4003.4000
16/32 位Timer 5 基址: 0x4003.5000
32/64 位宽 Timer 0 基址: 0x4003.6000
32/64 位宽 Timer 1 基址: 0x4003.7000
32/64 位宽 Timer 2 基址: 0x4004.C000
32/64 位宽 Timer 3 基址: 0x4004.D000
32/64 位宽 Timer 4 基址: 0x4004.E000
32/64 位宽 Timer 5 基址: 0x4004.F000
偏移量 0x018
类型 R/W, 复位 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	保留															WUEIM
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	保留				TBMIM	CBEIM	CBMIM	TBTOIM	保留			TAMIM	RTCIM	CAEIM	CAMIM	TATOIM
类型	RO	RO	RO	RO	R/W	R/W	R/W	R/W	RO	RO	RO	R/W	R/W	R/W	R/W	R/W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位/域	名称	类型	复位	描述
31:17	保留	RO	0x0000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
16	WUEIM	R/W	0	32/64 位宽 GPTM 写操作更新错误中断屏蔽 WUEIM 值定义如下： 值 描述 0 中断被禁止 1 中断被使能
15:12	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
11	TBMIM	R/W	0	GPTM Timer B 匹配中断屏蔽 TBMIM 值定义如下： 值 描述 0 中断被禁止 1 中断被使能
10	CBEIM	R/W	0	GPTM Timer B 捕获模式事件中断屏蔽 CBEIM 值定义如下： 值 描述 0 中断被禁止 1 中断被使能

位/域	名称	类型	复位	描述
9	CBMIM	R/W	0	GPTM Timer B 捕获模式匹配中断屏蔽 CBMIM 值定义如下： 值 描述 0 中断被禁止 1 中断被使能
8	TBTOIM	R/W	0	GPTM Timer B超时中断屏蔽 TBTOIM 值定义如下： 值 描述 0 中断被禁止 1 中断被使能
7:5	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
4	TAMIM	R/W	0	GPTM Timer A 匹配中断屏蔽 TAMIM 值定义如下： 值 描述 0 中断被禁止 1 中断被使能
3	RTCIM	R/W	0	GPTM实时时钟中断屏蔽 RTCIM 值定义如下： 值 描述 0 中断被禁止 1 中断被使能
2	CAEIM	R/W	0	GPTM Timer A 捕获模式事件中断屏蔽 CAEIM 值定义如下： 值 描述 0 中断被禁止 1 中断被使能
1	CAMIM	R/W	0	GPTM Timer A 捕获模式匹配中断屏蔽 CAMIM 值定义如下： 值 描述 0 中断被禁止 1 中断被使能

位/域	名称	类型	复位	描述
0	TATOIM	R/W	0	GPTM Timer A 超时中断屏蔽 TATOIM 值定义如下： 值 描述 0 中断被禁止 1 中断被使能

寄存器 7: GPTM 原始中断状态寄存器（GPTMRIS），偏移量 0x01C

该寄存器显示了GPTM内部中断信号的状态。不管是否在 GPTMIMR 寄存器中将中断屏蔽，这些位都会置位。向 GPTMICR 的某一位写 1 可将其中对应位清零。

注意: 通过 GPTM 控制 (GPTMCTL) 寄存器的 TnEN 位禁用再重新启用定时器不会影响 GPTMRIS 寄存器的状态。如果某个应用程序要求所有或部分状态位在重新启用定时器后不继续保留，那么在重新启用定时器之前，应通过 GPTMICR 寄存器将 GPTMRIS 寄存器中的相应位清零。如果未执行该操作，那么一旦重新启用定时器，在 GPTMRIS 寄存器中设置的状态位和未在 GPTMIMR 寄存器中屏蔽的状态位会产生一个中断信号。

GPTM 原始中断状态寄存器 (GPTMRIS)

16/32 位Timer 0 基址: 0x4003.0000
16/32 位Timer 1 基址: 0x4003.1000
16/32 位Timer 2 基址: 0x4003.2000
16/32 位Timer 3 基址: 0x4003.3000
16/32 位Timer 4 基址: 0x4003.4000
16/32 位Timer 5 基址: 0x4003.5000
32/64 位宽 Timer 0 基址: 0x4003.6000
32/64 位宽 Timer 1 基址: 0x4003.7000
32/64 位宽 Timer 2 基址: 0x4004.C000
32/64 位宽 Timer 3 基址: 0x4004.D000
32/64 位宽 Timer 4 基址: 0x4004.E000
32/64 位宽 Timer 5 基址: 0x4004.F000
偏移量 0x01C
类型 RO, 复位 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	保留															WUERIS
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	保留				TBMRIS	CBERIS	CBMRIS	TBTORIS	保留			TAMRIS	RTCRIS	CAERIS	CAMRIS	TATORIS
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位/域	名称	类型	复位	描述
31:17	保留	RO	0x0000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
16	WUERIS	R/W	0	32/64 位宽 GPTM 写操作更新错误原始中断状态 值 描述 0 无错误。 1 Timer A 寄存器或 Timer B 寄存器被连续写入两次，或 Timer A 寄存器在写入相应的 Timer B 寄存器前被写入。
15:12	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
11	TBMRIS	RO	0	GPTM Timer B 匹配原始中断 值 描述 0 匹配值没有到达 1 配置为单次触发或周期模式时，GPTMTBMR 寄存器的 TBMIE 位被置位，并达到 GPTMTBMATCHR 和（可选）GPTMTBPMR 寄存器中的匹配值。 通过向 GPTMICR 寄存器中的 TBMCINT 位写 1 来清零该位。

位/域	名称	类型	复位	描述
10	CBERIS	RO	0	GPTM Timer B 捕获模式事件原始中断 值 描述 0 Timer B 尚未发生捕获模式事件。 1 Timer B 已发生捕获模式事件。子定时器配置为输入边沿计时模式或配置为 PWM 模式（且通过将 GPTMTBMR 的 TBPWMIE 位置位而启用 PWM 中断）时，该中断有效。 通过写 1 到 GPTMICR 寄存器的 CBECINT 位可将该位清零。
9	CBMRIS	RO	0	GPTM Timer B 捕获模式匹配原始中断 值 描述 0 Timer B 尚未发生捕获模式匹配。 1 Timer B 已发生捕获模式匹配。配置为输入边沿计时模式时，如果 GPTMTBR 和 GPTMTBPR 中的值与 GPTMTBMATCHR 和 GPTMTBPMR 中的值相同，则该中断有效。 通过写 1 到 GPTMICR 寄存器的 CBMCINT 位可将该位清零。
8	TBTORIS	RO	0	GPTM Timer B超时原始中断 值 描述 0 Timer B 未超时。 1 Timer B 已超时。单次触发或周期模式定时器达到自身的计数限值（0 或载入 GPTMTBILR 的值，由计数方向决定）时，该中断有效。 通过写 1 到 GPTMICR 寄存器的 TBTOCINT 位可将该位清零。
7:5	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
4	TAMRIS	RO	0	GPTM Timer A 匹配原始中断 值 描述 0 匹配值没有到达 1 配置为单次触发或周期模式时，GPTMTAMR 寄存器的 TAMIE 位被置位，并达到 GPTMTAMATCHR 和GPTMTAPMR 寄存器（可选）中的匹配值。 通过写 1 到 GPTMICR 寄存器的 TAMCINT 位可将该位清零。
3	RTCRIS	RO	0	GPTM实时时钟原始中断 值 描述 0 RTC 事件未发生。 1 RTC 事件已发生。 通过写 1 到 GPTMICR 寄存器的 RTCCINT 位可将该位清零。

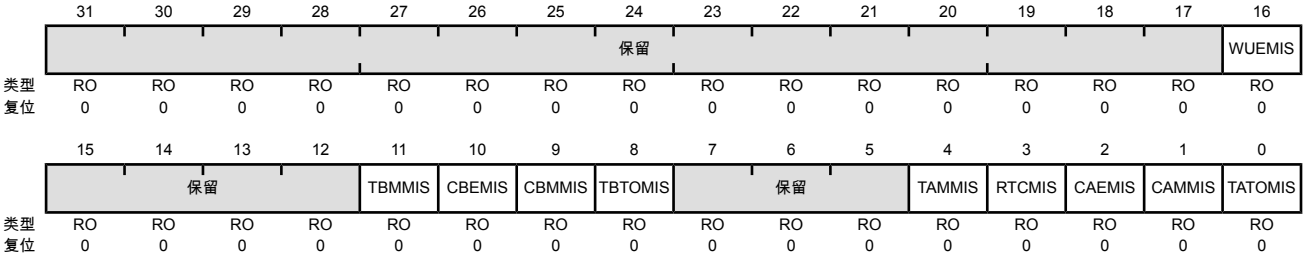
位/域	名称	类型	复位	描述
2	CAERIS	RO	0	GPTM Timer A 捕获模式事件原始中断 值 描述 0 Timer A 尚未发生捕获模式事件。 1 Timer A 已发生捕获模式事件。子定时器配置为输入边沿计时模式或配置为 PWM 模式（且通过将 GPTMTAMR 的 TAPWMIE 位置位而启用 PWM 中断）时，该中断有效。 通过写 1 到 GPTMICR 寄存器的 CAECINT 位可将该位清零。
1	CAMRIS	RO	0	GPTM Timer A 捕获模式匹配原始中断 值 描述 0 Timer A 尚未发生捕获模式匹配。 1 Timer A 已发生捕获模式匹配。配置为输入边沿计时模式时，如果 GPTMTAR 和 GPTMTAPR 中的值与 GPTMTAMATCHR 和 GPTMTAPMR 中的值相同，则该中断有效。 通过写 1 到 GPTMICR 寄存器的 CAMCINT 位可将该位清零。
0	TATORIS	RO	0	GPTM Timer A超时中断清除 值 描述 0 Timer A 未超时。 1 Timer A 已超时。单次触发或周期模式定时器达到自身的计数限值（0 或载入 GPTMTAILR 的值，由计数方向决定）时，该中断有效。 该位通过写 1 到 GPTMICR 寄存器的 TATOCINT 位来清零。

寄存器 8: GPTM 屏蔽的中断状态寄存器（GPTMMIS），偏移量 0x020

该寄存器显示了GPTM控制器级中断的状态。如果没有在 GPTMIMR 寄存器中将中断屏蔽，并在此时出现一个使中断有效的事件，那么该寄存器中相应的位将会置位。通过向 GPTMICR 的对应位写 1 可将所有位清零。

GPTM 屏蔽的中断状态寄存器 (GPTMMIS)

16/32 位Timer 0 基址: 0x4003.0000
16/32 位Timer 1 基址: 0x4003.1000
16/32 位Timer 2 基址: 0x4003.2000
16/32 位Timer 3 基址: 0x4003.3000
16/32 位Timer 4 基址: 0x4003.4000
16/32 位Timer 5 基址: 0x4003.5000
32/64 位宽 Timer 0 基址: 0x4003.6000
32/64 位宽 Timer 1 基址: 0x4003.7000
32/64 位宽 Timer 2 基址: 0x4004.C000
32/64 位宽 Timer 3 基址: 0x4004.D000
32/64 位宽 Timer 4 基址: 0x4004.E000
32/64 位宽 Timer 5 基址: 0x4004.F000
偏移量 0x020
类型 RO, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:17	保留	RO	0x0000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
16	WUEMIS	RO	0	32/64 位宽 GPTM 写操作更新错误屏蔽中断状态 值 描述 0 未屏蔽的写操作更新错误未发生。 1 未屏蔽的写操作更新错误已发生。
15:12	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
11	TBMMIS	RO	0	GPTM Timer B 匹配屏蔽中断 值 描述 0 匹配值没有到达或中断已经被屏蔽 1 未屏蔽的 Timer B 模式匹配中断已发生。 通过向 GPTMICR 寄存器中的 TBMCINT 位写 1 来清零该位。

位/域	名称	类型	复位	描述
10	CBEMIS	RO	0	GPTM Timer B 捕获模式事件屏蔽中断 值 描述 0 捕获器B事件没有发生或中断已经被屏蔽 1 未屏蔽的捕获 B 事件中断 已发生。 通过写 1 到 GPTMICR 寄存器的 CBECINT 位可将该位清零。
9	CBMMIS	RO	0	GPTM Timer B 捕获模式匹配屏蔽中断 值 描述 0 捕获 B 模式匹配值没有到达或中断已经被屏蔽。 1 未屏蔽的捕获 B 匹配中断 已发生。 通过写 1 到 GPTMICR 寄存器的 CBMCINT 位可将该位清零。
8	TBTOMIS	RO	0	GPTM Timer B 超时屏蔽的中断 值 描述 0 捕获器B没有超时或中断已经被屏蔽 1 未屏蔽的 Timer B 超时中断 已发生。 通过写 1 到 GPTMICR 寄存器的 TBTOCINT 位可将该位清零。
7:5	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
4	TAMMIS	RO	0	GPTM Timer A 匹配屏蔽中断 值 描述 0 Timer A 模式匹配值没有到达或中断已经被屏蔽。 1 未屏蔽的 Timer A 模式匹配中断 已发生。 通过写 1 到 GPTMICR 寄存器的 TAMCINT 位可将该位清零。
3	RTCMIS	RO	0	GPTM RTC 屏蔽的中断 值 描述 0 RTC 事件未产生中断，或此中断被屏蔽。 1 未屏蔽的 RTC 事件中断 已发生。 通过写 1 到 GPTMICR 寄存器的 RTCCINT 位可将该位清零。

位/域	名称	类型	复位	描述
2	CAEMIS	RO	0	GPTM Timer A 捕获模式事件屏蔽中断 值 描述 0 捕获 B 事件没有发生或中断已经被屏蔽。 1 未屏蔽的捕获 A 事件中断已发生。 通过写 1 到 GPTMICR 寄存器的 CAECINT 位可将该位清零。
1	CAMMIS	RO	0	GPTM Timer A 捕获模式匹配屏蔽中断 值 描述 0 捕获 A 模式匹配值没有到达或中断已经被屏蔽。 1 未屏蔽的捕获 A 匹配中断已发生。 通过写 1 到 GPTMICR 寄存器的 CAMCINT 位可将该位清零。
0	TATOMIS	RO	0	GPTM Timer A 超时屏蔽的中断 值 描述 0 Timer A 没有超时或中断已经被屏蔽。 1 未屏蔽的 Timer A 超时中断已发生。 该位通过写 1 到 GPTMICR 寄存器的 TATOCINT 位来清零。

寄存器 9: GPTM 中断清除寄存器（GPTMICR），偏移量 0x024

该寄存器用来清零 GPTMRIS 和 GPTMMIS 寄存器的状态位。向该寄存器写 1 可清除 GPTMRIS 和 GPTMMIS 中相应的位。

GPTM 中断清除寄存器 (GPTMICR)

16/32 位Timer 0 基址: 0x4003.0000
16/32 位Timer 1 基址: 0x4003.1000
16/32 位Timer 2 基址: 0x4003.2000
16/32 位Timer 3 基址: 0x4003.3000
16/32 位Timer 4 基址: 0x4003.4000
16/32 位Timer 5 基址: 0x4003.5000
32/64 位宽 Timer 0 基址: 0x4003.6000
32/64 位宽 Timer 1 基址: 0x4003.7000
32/64 位宽 Timer 2 基址: 0x4004.C000
32/64 位宽 Timer 3 基址: 0x4004.D000
32/64 位宽 Timer 4 基址: 0x4004.E000
32/64 位宽 Timer 5 基址: 0x4004.F000
偏移量 0x024
类型 W1C, 复位 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	保留															WUECINT
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	保留				TBMCINT	CBECINT	CBMCINT	TBTOCINT	保留			TAMCINT	RTCCINT	CAECINT	CAMCINT	TATOCINT
类型	RO	RO	RO	RO	W1C	W1C	W1C	W1C	RO	RO	RO	W1C	W1C	W1C	W1C	W1C
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位/域	名称	类型	复位	描述
31:17	保留	RO	0x0000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
16	WUECINT	R/W	0	32/64 位宽 GPTM 写操作更新错误中断清除 向本标志位写 1，即可将 GPTMRIS 寄存器的 WUERIS 位以及 GPTMMIS 寄存器的 WUEMIS 位清零。
15:12	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
11	TBMCINT	W1C	0	GPTM Timer B 匹配中断清零 该位写 1 可清除 GPTMRIS 寄存器的 TBMRIS 位和 GPTMMIS 寄存器 TBMMIS 位。
10	CBECINT	W1C	0	GPTM Timer B 捕获模式事件中断清零 该位写 1 可清除 GPTMRIS 寄存器的 CBERIS 位和 GPTMMIS 寄存器 CBEMIS 位。
9	CBMCINT	W1C	0	GPTM Timer B 捕获模式匹配中断清零 该位写 1 可清除 GPTMRIS 寄存器的 CBMRIS 位和 GPTMMIS 寄存器 CBMMIS 位。
8	TBTOCINT	W1C	0	GPTM Timer B超时中断清除 该位写 1 可清除 GPTMRIS 寄存器的 TBTORIS 位和 GPTMMIS 寄存器 TBTOMIS位。
7:5	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。

位/域	名称	类型	复位	描述
4	TAMCINT	W1C	0	GPTM Timer A 匹配中断清零 该位写 1 可清除 GPTMRIS 寄存器的 TAMRIS 位和 GPTMMIS 寄存器 TAMMIS 位。
3	RTCCINT	W1C	0	GPTM实时时钟中断清除 该位写 1 可清除 GPTMRIS 寄存器的 RTCRIS位和 GPTMMIS 寄存器 RTCMIS位。
2	CAECINT	W1C	0	GPTM Timer A 捕获模式事件中断清零 该位写 1 可清除 GPTMRIS 寄存器的 CAERIS 位和 GPTMMIS 寄存器 CAEMIS 位。
1	CAMCINT	W1C	0	GPTM Timer A 捕获模式匹配中断清零 该位写 1 可清除 GPTMRIS 寄存器的 CAMRIS 位和 GPTMMIS 寄存器 CAMMIS 位。
0	TATOCINT	W1C	0	GPTM Timer A超时中断清除 该位写 1 可清除 GPTMRIS 寄存器的 TATORIS 位和 GPTMMIS 寄存器 TATOMIS 位。

寄存器 10: GPTM Timer A 间隔加载寄存器（GPTMTAILR），偏移量 0x028

当定时器递减计数时，该寄存器为定时器装载初值。当定时器递增计数时，该寄存器为超时事件设置上边沿。

当 16/32 位 GPTM 配置为其中一种 32 位模式时，GPTMTAILR 作为 32 位寄存器使用（高 16 位对应于 GPTM Timer B 间隔加载 (GPTMTBILR) 寄存器的值）。当配置为 16 位时，高 16 位读出来是 0，并且不会影响 GPTMTBILR 寄存器的状态。

当 32/64 位宽 GPTM 配置为其中一种 64 位模式时，GPTMTAILR 包含 64 位计数的 31:0 位，并且 GPTM Timer B 间隔加载 (GPTMTBILR) 寄存器包含 63:32 位。

GPTM Timer A 间隔加载寄存器 (GPTMTAILR)

16/32 位Timer 0 基址: 0x4003.0000
16/32 位Timer 1 基址: 0x4003.1000
16/32 位Timer 2 基址: 0x4003.2000
16/32 位Timer 3 基址: 0x4003.3000
16/32 位Timer 4 基址: 0x4003.4000
16/32 位Timer 5 基址: 0x4003.5000
32/64 位宽 Timer 0 基址: 0x4003.6000
32/64 位宽 Timer 1 基址: 0x4003.7000
32/64 位宽 Timer 2 基址: 0x4004.C000
32/64 位宽 Timer 3 基址: 0x4004.D000
32/64 位宽 Timer 4 基址: 0x4004.E000
32/64 位宽 Timer 5 基址: 0x4004.F000
偏移量 0x028
类型 R/W, 复位 0xFFFF.FFFF

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	TAILR															
类型	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	TAILR															
类型	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

位/域	名称	类型	复位	描述
31:0	TAILR	R/W	0xFFFF.FFFF	GPTM Timer A 间隔加载寄存器 写入该域将为 Timer A 加载计数器，读取该域将返回 GPTMTAILR 寄存器的当前值。

寄存器 11: GPTM Timer B 间隔加载寄存器（GPTMTBILR），偏移量 0x02C

当定时器递减计数时，该寄存器为定时器装载初值。当定时器递增计数时，该寄存器为超时事件设置上边沿。

当 16/32 位 GPTM 配置为其中一种 32 位模式时，该寄存器中 15:0 位的内容被加载到 GPTMTAILR 寄存器的高 16 位。读取该寄存器将返回 Timer B 的当前值并且忽略写操作。在 16 位模式中，15:0 位用于加载值。在两种情况下都保留 31:16 位。

当 32/64 位宽 GPTM 配置为其中一种 64 位模式时，GPTMTAILR 包含 64 位计数的 31:0 位，并且 GPTMTBILR 寄存器包含 63:32 位。

GPTM Timer B 间隔加载寄存器 (GPTMTBILR)

16/32 位Timer 0 基址: 0x4003.0000
16/32 位Timer 1 基址: 0x4003.1000
16/32 位Timer 2 基址: 0x4003.2000
16/32 位Timer 3 基址: 0x4003.3000
16/32 位Timer 4 基址: 0x4003.4000
16/32 位Timer 5 基址: 0x4003.5000
32/64 位宽 Timer 0 基址: 0x4003.6000
32/64 位宽 Timer 1 基址: 0x4003.7000
32/64 位宽 Timer 2 基址: 0x4004.C000
32/64 位宽 Timer 3 基址: 0x4004.D000
32/64 位宽 Timer 4 基址: 0x4004.E000
32/64 位宽 Timer 5 基址: 0x4004.F000
偏移量 0x02C
类型 R/W, 复位 -

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	TBILR															
类型	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	TBILR															
类型	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位	0	0	1	1	0	0	1	0	0	1	1	0	0	1	0	0

位/域	名称	类型	复位	描述
31:0	TBILR	R/W	0x0000.FFFF（适用于 16/32 位） 0xFFFF.FFFF（适用于 32/64 位）	GPTM Timer B 间隔加载寄存器 写入该域将为 Timer B 加载计数器，读取该域将返回 GPTMTBILR 寄存器的当前值。 当 16/32 位 GPTM 处于 32 位模式中，写操作被忽略，读操作返回 GPTMTBILR 的当前值。

寄存器 12: GPTM Timer A 匹配寄存器（GPTMTAMATCHR），偏移量 0x030

该寄存器加载一个匹配值。在单次触发或周期模式中，如果定时器的值等于该寄存器的值则引发一个中断。

在边沿计数模式中，该寄存器和 GPTMTAILR 寄存器一起记录共有多少个边沿事件发生。检测到的边沿数等于 GPTMTAILR 的值减去该寄存器里的值。请注意：在边沿计数模式中执行递增计数时，GPTMTnPR 和 GPTMTnILR 的值必须大于 GPTMTnPMR 和 GPTMTnMATCHR 的值。

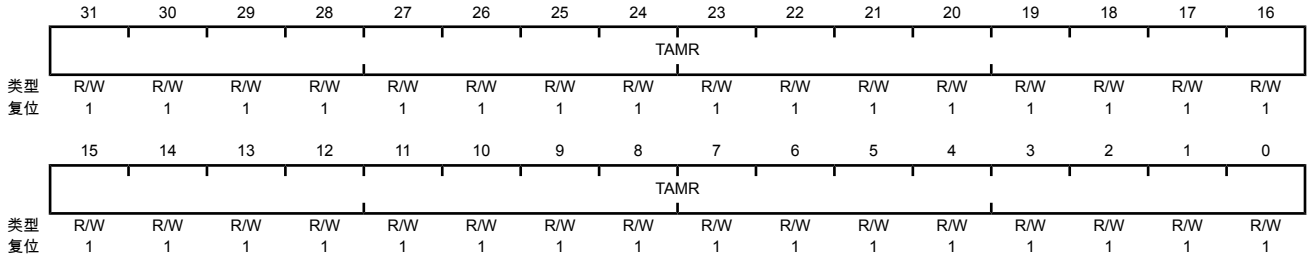
在 PWM 模式中，该寄存器与 GPTMTAILR 寄存器一起决定输出信号的占空比。

当 16/32 位 GPTM 配置为其中一种 32 位模式时，GPTMTAMATCHR 作为 32 位寄存器使用（高 16 位对应于 GPTM Timer B 匹配 (GPTMTBMATCHR) 寄存器的值）。当配置为 16 位时，高 16 位读出来是 0，并且不会影响 GPTMTBMATCHR 寄存器的状态。

当 32/64 位宽 GPTM 配置为其中一种 64 位模式时，GPTMTAMATCHR 包含 64 位匹配值的 31:0 位，并且 GPTM Timer B 匹配 (GPTMTBMATCHR) 寄存器包含 63:32 位。

GPTM Timer A 匹配寄存器 (GPTMTAMATCHR)

16/32 位Timer 0 基址: 0x4003.0000
16/32 位Timer 1 基址: 0x4003.1000
16/32 位Timer 2 基址: 0x4003.2000
16/32 位Timer 3 基址: 0x4003.3000
16/32 位Timer 4 基址: 0x4003.4000
16/32 位Timer 5 基址: 0x4003.5000
32/64 位宽 Timer 0 基址: 0x4003.6000
32/64 位宽 Timer 1 基址: 0x4003.7000
32/64 位宽 Timer 2 基址: 0x4004.C000
32/64 位宽 Timer 3 基址: 0x4004.D000
32/64 位宽 Timer 4 基址: 0x4004.E000
32/64 位宽 Timer 5 基址: 0x4004.F000
偏移量 0x030
类型 R/W, 复位 0xFFFF.FFFF



位/域	名称	类型	复位	描述
31:0	TAMR	R/W	0xFFFF.FFFF	GPTM Timer A 匹配寄存器 该值与 GPTMTAR 寄存器进行比较来确定匹配事件的发生。

寄存器 13: GPTM Timer B 匹配寄存器（GPTMTBMATCHR），偏移量 0x034

该寄存器加载一个匹配值。在单次触发或周期模式中，如果定时器的值等于该寄存器的值则引发一个中断。

在边沿计数模式中，该寄存器和 GPTMTBILR 寄存器一起记录共有多少个边沿事件发生。检测到的边沿数等于 GPTMTBILR 的值减去该寄存器里的值。请注意：在边沿计数模式中执行递增计数时，GPTMTnPR 和 GPTMTnILR 的值必须大于 GPTMTnPMR 和 GPTMTnMATCHR 的值。

在 PWM 模式中，该寄存器与 GPTMTBILR 寄存器一起决定输出信号的占空比。

当 16/32 位 GPTM 配置为其中一种 32 位模式时，该寄存器中 15:0 位的内容被加载到 GPTMTAMATCHR 寄存器的高 16 位。读取该寄存器将返回 Timer B 的匹配值并且忽略写操作。在 16 位模式中，15:0 位用于匹配值。在两种情况下都保留 31:16 位。

当 32/64 位宽 GPTM 配置为其中一种 64 位模式时，GPTMTAMATCHR 包含 64 位匹配值的 31:0 位，并且 GPTMTBMATCHR 寄存器包含 63:32 位。

GPTM Timer B 匹配寄存器 (GPTMTBMATCHR)

16/32 位Timer 0 基址: 0x4003.0000
16/32 位Timer 1 基址: 0x4003.1000
16/32 位Timer 2 基址: 0x4003.2000
16/32 位Timer 3 基址: 0x4003.3000
16/32 位Timer 4 基址: 0x4003.4000
16/32 位Timer 5 基址: 0x4003.5000
32/64 位宽 Timer 0 基址: 0x4003.6000
32/64 位宽 Timer 1 基址: 0x4003.7000
32/64 位宽 Timer 2 基址: 0x4004.C000
32/64 位宽 Timer 3 基址: 0x4004.D000
32/64 位宽 Timer 4 基址: 0x4004.E000
32/64 位宽 Timer 5 基址: 0x4004.F000
偏移量 0x034
类型 R/W, 复位 -

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	TBMR															
类型	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	TBMR															
类型	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位	0	0	1	1	0	0	1	0	0	1	1	0	0	1	0	0

位/域	名称	类型	复位	描述
31:0	TBMR	R/W	0x0000.FFFF（适用于 16/32 位） 0xFFFF.FFFF（适用于 32/64 位）	GPTM Timer B 匹配寄存器 该值与 GPTMTBR 寄存器进行比较来确定匹配事件的发生。

寄存器 14: GPTM Timer A 预分频寄存器（GPTMTAPR），偏移量 0x038

该寄存器允许软件扩展独立使用时定时器的范围。在单次触发或周期递减计数模式中，该寄存器用作定时器计数器的真预分频器。用作真预分频器时，在 GPTMTAR 和 GPTMTAV 寄存器的值递增前，预分频器计数递减到 0。在所有其他独立/分离模式中，该寄存器是定时器计数器上限范围的一个线性扩展，在 16/32 位 GPTM 的 16 位模式下包含位 23:16，在 32/64 位宽 GPTM 的 32 位模式下包含位 47:32。

GPTM Timer A 预分频寄存器 (GPTMTAPR)

16/32 位Timer 0 基址: 0x4003.0000
16/32 位Timer 1 基址: 0x4003.1000
16/32 位Timer 2 基址: 0x4003.2000
16/32 位Timer 3 基址: 0x4003.3000
16/32 位Timer 4 基址: 0x4003.4000
16/32 位Timer 5 基址: 0x4003.5000
32/64 位宽 Timer 0 基址: 0x4003.6000
32/64 位宽 Timer 1 基址: 0x4003.7000
32/64 位宽 Timer 2 基址: 0x4004.C000
32/64 位宽 Timer 3 基址: 0x4004.D000
32/64 位宽 Timer 4 基址: 0x4004.E000
32/64 位宽 Timer 5 基址: 0x4004.F000
偏移量 0x038
类型 R/W, 复位 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	保留															
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	TAPSRH								TAPSR							
类型	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

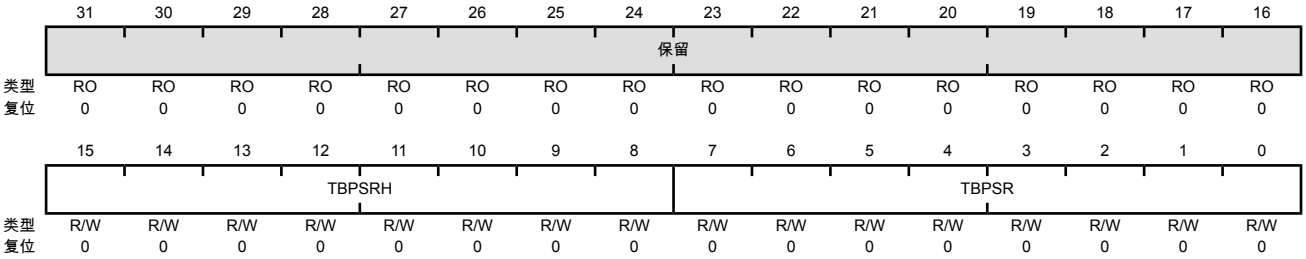
位/域	名称	类型	复位	描述
31:16	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
15:8	TAPSRH	R/W	0x00	GPTM Timer A 预分频高字节 寄存器通过写操作载入该值。执行读操作将返回寄存器的当前值。 对于 16/32 位 GPTM，保留该域。对于 32/64 位宽 GPTM，该域包含 16 位预分频器的高 8 位。 详细信息和实例请参考表11-5（634页）。
7:0	TAPSR	R/W	0x00	GPTM Timer A预分频 寄存器通过写操作载入该值。执行读操作将返回寄存器的当前值。 对于 16/32 位 GPTM，该域包含全部 8 位预分频器。对于 32/64 位宽 GPTM，该域包含 16 位预分频器的低 8 位。 详细信息和实例请参考表11-5（634页）。

寄存器 15: GPTM Timer B 预分频寄存器（GPTMTBPR），偏移量 0x03C

该寄存器允许软件扩展独立使用时定时器的范围。在单次触发或周期递减计数模式中，该寄存器用作定时器计数器的真预分频器。用作真预分频器时，预分频器在 GPTMTBR 和 GPTMTBV 寄存器的值递增前，计数递减到 0。在所有其他独立/分离模式中，该寄存器是定时器计数器上限范围的一个线性扩展，在 16/32 位 GPTM 的 16 位模式下包含位 23:16，在 32/64 位宽 GPTM 的 32 位模式下包含位 47:32。

GPTM Timer B 预分频寄存器 (GPTMTBPR)

16/32 位Timer 0 基址: 0x4003.0000
16/32 位Timer 1 基址: 0x4003.1000
16/32 位Timer 2 基址: 0x4003.2000
16/32 位Timer 3 基址: 0x4003.3000
16/32 位Timer 4 基址: 0x4003.4000
16/32 位Timer 5 基址: 0x4003.5000
32/64 位宽 Timer 0 基址: 0x4003.6000
32/64 位宽 Timer 1 基址: 0x4003.7000
32/64 位宽 Timer 2 基址: 0x4004.C000
32/64 位宽 Timer 3 基址: 0x4004.D000
32/64 位宽 Timer 4 基址: 0x4004.E000
32/64 位宽 Timer 5 基址: 0x4004.F000
偏移量 0x03C
类型 R/W, 复位 0x0000.0000



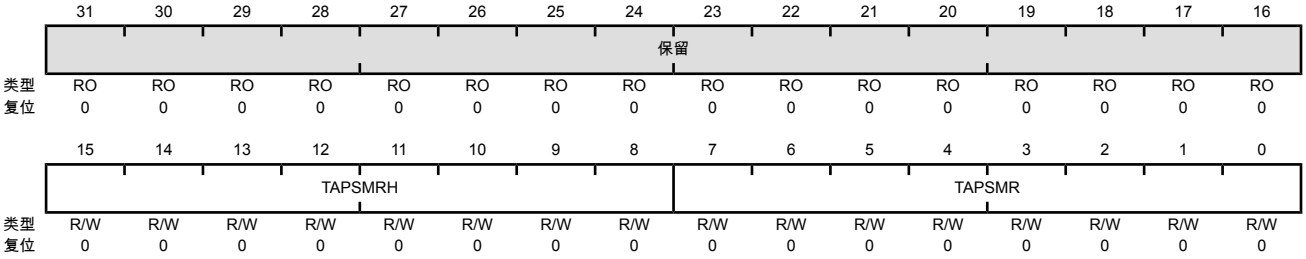
位/域	名称	类型	复位	描述
31:16	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
15:8	TBPSRH	R/W	0x00	GPTM Timer B 预分频高字节 寄存器通过写操作载入该值。执行读操作将返回寄存器的当前值。 对于 16/32 位 GPTM，保留该域。对于 32/64 位宽 GPTM，该域包含 16 位预分频器的高 8 位。 详细信息和实例请参考表11-5（634页）。
7:0	TBPSR	R/W	0x00	GPTM Timer B预分频寄存器 寄存器通过写操作载入该值。执行读操作将返回寄存器的当前值。 对于 16/32 位 GPTM，该域包含全部 8 位预分频器。对于 32/64 位宽 GPTM，该域包含 16 位预分频器的低 8 位。 详细信息和实例请参考表11-5（634页）。

寄存器 16: GPTM TimerA 预分频匹配寄存器（GPTMTAPMR），偏移量 0x040

该寄存器允许软件扩展在单独使用定时器时 GPTMTAMATCHR 的范围。该寄存器保存 16/32 位 GPTM 的 16 位模式中的 23:16 位以及 32/64 位宽 GPTM 的 32 位模式中的 47:32 位。

GPTM TimerA 预分频匹配寄存器 (GPTMTAPMR)

16/32 位Timer 0 基址: 0x4003.0000
16/32 位Timer 1 基址: 0x4003.1000
16/32 位Timer 2 基址: 0x4003.2000
16/32 位Timer 3 基址: 0x4003.3000
16/32 位Timer 4 基址: 0x4003.4000
16/32 位Timer 5 基址: 0x4003.5000
32/64 位宽 Timer 0 基址: 0x4003.6000
32/64 位宽 Timer 1 基址: 0x4003.7000
32/64 位宽 Timer 2 基址: 0x4004.C000
32/64 位宽 Timer 3 基址: 0x4004.D000
32/64 位宽 Timer 4 基址: 0x4004.E000
32/64 位宽 Timer 5 基址: 0x4004.F000
偏移量 0x040
类型 R/W, 复位 0x0000.0000



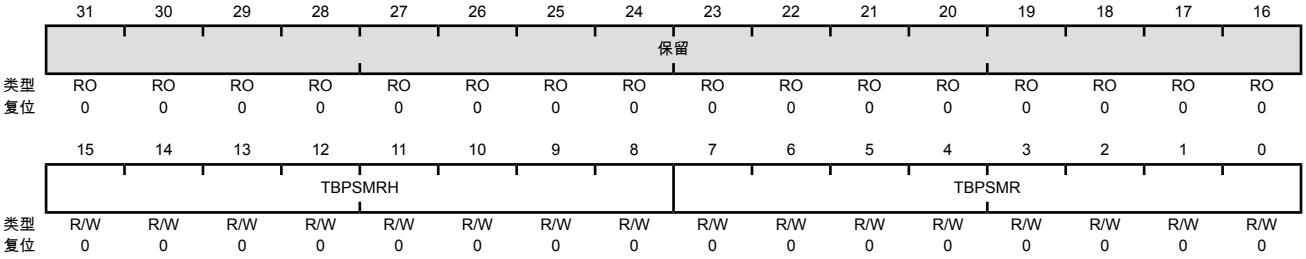
位/域	名称	类型	复位	描述
31:16	保留	RO	0x0000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
15:8	TAPSMRH	R/W	0x00	GPTM Timer A 预分频匹配高字节 该值与 GPTMTAMATCHR 一起使用，以便在使用预分频器的情况下检测定时器匹配事件。 对于 16/32 位 GPTM，保留该域。对于 32/64 位宽 GPTM，该域包含 16 位预分频匹配值的高 8 位。
7:0	TAPSMR	R/W	0x00	GPTM Timer A预分频匹配 该值与 GPTMTAMATCHR 一起使用，以便在使用预分频器的情况下检测定时器匹配事件。 对于 16/32 位 GPTM，该域包含全部 8 位预分频器匹配值。对于 32/64 位宽 GPTM，该域包含 16 位预分频器匹配值的低 8 位。

寄存器 17: GPTM TimerB 预分频匹配寄存器（GPTMTBPMR），偏移量 0x044

该寄存器允许软件扩展在单独使用定时器时 GPTMTBMATCHR 的范围。该寄存器保存 16/32 位 GPTM 的 16 位模式中的 23:16 位以及 32/64 位宽 GPTM 的 32 位模式中的 47:32 位。

GPTM TimerB 预分频匹配寄存器 (GPTMTBPMR)

16/32 位Timer 0 基址: 0x4003.0000
16/32 位Timer 1 基址: 0x4003.1000
16/32 位Timer 2 基址: 0x4003.2000
16/32 位Timer 3 基址: 0x4003.3000
16/32 位Timer 4 基址: 0x4003.4000
16/32 位Timer 5 基址: 0x4003.5000
32/64 位宽 Timer 0 基址: 0x4003.6000
32/64 位宽 Timer 1 基址: 0x4003.7000
32/64 位宽 Timer 2 基址: 0x4004.C000
32/64 位宽 Timer 3 基址: 0x4004.D000
32/64 位宽 Timer 4 基址: 0x4004.E000
32/64 位宽 Timer 5 基址: 0x4004.F000
偏移量 0x044
类型 R/W, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:16	保留	RO	0x0000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
15:8	TBPSMRH	R/W	0x00	GPTM Timer B 预分频匹配高字节 该值与 GPTMTBMATCHR 一起使用，以便在使用预分频器的情况下检测定时器匹配事件。 对于 16/32 位 GPTM，保留该域。对于 32/64 位宽 GPTM，该域包含 16 位预分频匹配值的高 8 位。
7:0	TBPSMR	R/W	0x00	GPTM Timer B预分频匹配寄存器 该值与 GPTMTBMATCHR 一起使用，以便在使用预分频器的情况下检测定时器匹配事件。 对于 16/32 位 GPTM，该域包含全部 8 位预分频器匹配值。对于 32/64 位宽 GPTM，该域包含 16 位预分频器匹配值的低 8 位。

寄存器 18: GPTM Timer A 寄存器（GPTMTAPR），偏移量 0x048

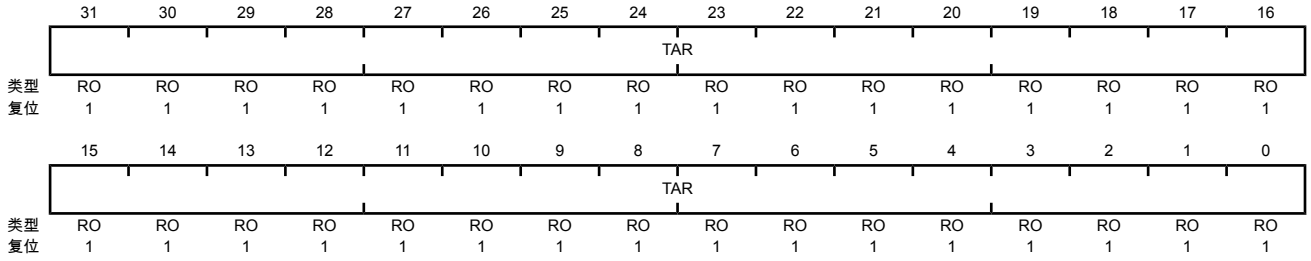
在任何情况下该寄存器显示当前 Timer A 计数器的值，输入边沿计数模式的情况除外。在输入边沿计数模式中，该寄存器记录了已经发生的边沿的个数。在输入边沿计时模式中，该寄存器包含上一次边沿事件发生的时间。

当 16/32 位 GPTM 配置为其中一种 32 位模式时，GPTMTAR 作为 32 位寄存器使用（高 16 位对应于 GPTM Timer B (GPTMTBR) 寄存器的值）。在16 位输入边沿计数、输入边沿计时和 PWM 模式中，15:0 位包含计数器的值，23:16 位包含预分频器高 8 位的值。31:24 位的回读值始终为 0。要在 16 位单次触发和周期模式中读取预分频器的值，读取 GPTMTAV 寄存器中的 [23:16] 位。在周期快照模式中，读取 Timer A 预分频快照 (GPTMTAPS) 寄存器可获得预分频器的值。

当 32/64 位宽 GPTM 配置为其中一种 64 位模式时，GPTMTAR 包含 64 位定时器值的 31:0 位并且 GPTM Timer B (GPTMTBR) 寄存器包含 63:32 位。在 32 位模式中，预分频器的值存储在 GPTM Timer A 预分频快照 (GPTMTAPS) 寄存器中。

GPTM Timer A 寄存器 (GPTMTAPR)

16/32 位Timer 0 基址: 0x4003.0000
16/32 位Timer 1 基址: 0x4003.1000
16/32 位Timer 2 基址: 0x4003.2000
16/32 位Timer 3 基址: 0x4003.3000
16/32 位Timer 4 基址: 0x4003.4000
16/32 位Timer 5 基址: 0x4003.5000
32/64 位宽 Timer 0 基址: 0x4003.6000
32/64 位宽 Timer 1 基址: 0x4003.7000
32/64 位宽 Timer 2 基址: 0x4004.C000
32/64 位宽 Timer 3 基址: 0x4004.D000
32/64 位宽 Timer 4 基址: 0x4004.E000
32/64 位宽 Timer 5 基址: 0x4004.F000
偏移量 0x048
类型 RO, 复位 0xFFFF.FFFF



位/域	名称	类型	复位	描述
31:0	TAR	RO	0xFFFF.FFFF	GPTM Timer A 寄存器 执行读操作将返回 GPTM Timer A 计数寄存器 的当前值，输入边沿计数和计时模式的情况除外。在输入边沿计数模式中，该寄存器记录了已经发生的边沿的个数。在输入边沿计时模式中，该寄存器包含上一次边沿事件发生的时间。

寄存器 19: GPTM Timer B 寄存器（GPTMTBPR），偏移量 0x04C

在任何情况下该寄存器显示当前 Timer B 计数器的值，输入边沿计数模式的情况除外。在输入边沿计数模式中，该寄存器记录了已经发生的边沿的个数。在输入边沿计时模式中，该寄存器包含上一次边沿事件发生的时间。

当 16/32 位 GPTM 配置为其中一种 32 位模式时，该寄存器中 15:0 位的内容被加载到 GPTMTAR 寄存器的高 16 位。读取该寄存器将返回 Timer B 的当前值。在 16 位模式中，15:0 位包含计数器的值，23:16 位包含输入边沿计数、输入边沿计时和 PWM 模式中中预分频器高 8 位的值。31:24 位的回读值始终为 0。要在 16 位单次触发和周期模式中读取预分频器的值，应读取 GPTMTBV 寄存器中的 [23:16] 位。在周期快照模式中，读取 Timer B 预分频快照 (GPTMTBPS) 寄存器可获得预分频器的值。

当 32/64 位宽 GPTM 配置为其中一种 64 位模式时，GPTMTAR 包含 64 位定时器值的 31:0 位并且 GPTM Timer B (GPTMTBR) 寄存器包含 63:32 位。在 32 位模式中，预分频器的值存储在 GPTM Timer B 预分频快照 (GPTMTBPS) 寄存器中。

GPTM Timer B 寄存器 (GPTMTBPR)

16/32 位Timer 0 基址: 0x4003.0000
16/32 位Timer 1 基址: 0x4003.1000
16/32 位Timer 2 基址: 0x4003.2000
16/32 位Timer 3 基址: 0x4003.3000
16/32 位Timer 4 基址: 0x4003.4000
16/32 位Timer 5 基址: 0x4003.5000
32/64 位宽 Timer 0 基址: 0x4003.6000
32/64 位宽 Timer 1 基址: 0x4003.7000
32/64 位宽 Timer 2 基址: 0x4004.C000
32/64 位宽 Timer 3 基址: 0x4004.D000
32/64 位宽 Timer 4 基址: 0x4004.E000
32/64 位宽 Timer 5 基址: 0x4004.F000
偏移量 0x04C
类型 RO, 复位 -

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	TBR															
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	TBR															
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	1	1	0	0	1	0	0	1	1	0	0	1	0	0

位/域	名称	类型	复位	描述
31:0	TBR	RO	0x0000.FFFF（适用于 16/32 位） 0xFFFF.FFFF（适用于 32/64 位）	GPTM Timer B 寄存器 执行读操作将返回 GPTM Timer B 计数寄存器的当前值，输入边沿计数和计时模式的情况除外。在输入边沿计数模式中，该寄存器记录了已经发生的边沿的个数。在输入边沿计时模式中，该寄存器包含上一次边沿事件发生的时间。

寄存器 20: GPTM Timer A 值寄存器（GPTMTAV），偏移量 0x050

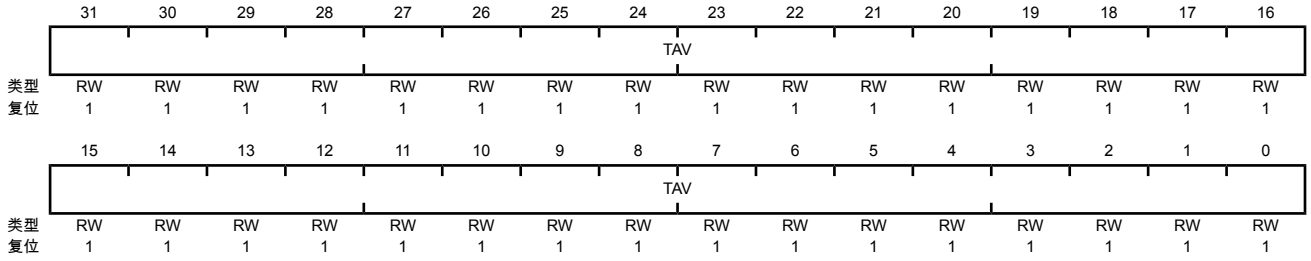
在所有模式下，读取该寄存器返回Timer A自由运行的值。在周期操作模式下配合使用快照特性时，软件可以根据该值来判断从发生中断到进入 ISR（中断服务程序）之间所用时间。向该寄存器写入的值将会在下一个时钟周期加载到 GPTMTAR 寄存器里。

当 16/32 位 GPTM 配置为其中一种 32 位模式时，GPTMTAV 作为 32 位寄存器使用（高 16 位对应于 GPTM Timer B 值 (GPTMTBV) 寄存器的值）。在16 位模式中，15:0 位包含计数器的值，23:16 位包含预分频器当前自由运行的值，该值是输入边沿计数、输入边沿计时、PWM 和单次触发或周期递增计数模式的计数高 8 位。在单次触发或周期递减计数模式中，存储在 23:16 的预分频器是真预分频器，即 23:16 位在 15:0 位的值递减之前递减计数。31:24 位预分频器的回读值始终为 0。

当 32/64 位宽 GPTM 配置为其中一种 64 位模式时，GPTMTAV 包含 64 位定时器值的 31:0 位并且 GPTM Timer B 值 (GPTMTBV) 寄存器包含 63:32 位。在 32 位模式中，预分频器的当前自由运行值存储在 GPTM Timer A 预分频值 (GPTMTAPV) 寄存器.mint 中

GPTM Timer A 值寄存器 (GPTMTAV)

16/32 位Timer 0 基址: 0x4003.0000
16/32 位Timer 1 基址: 0x4003.1000
16/32 位Timer 2 基址: 0x4003.2000
16/32 位Timer 3 基址: 0x4003.3000
16/32 位Timer 4 基址: 0x4003.4000
16/32 位Timer 5 基址: 0x4003.5000
32/64 位宽 Timer 0 基址: 0x4003.6000
32/64 位宽 Timer 1 基址: 0x4003.7000
32/64 位宽 Timer 2 基址: 0x4004.C000
32/64 位宽 Timer 3 基址: 0x4004.D000
32/64 位宽 Timer 4 基址: 0x4004.E000
32/64 位宽 Timer 5 基址: 0x4004.F000
偏移量 0x050
类型 RW, 复位 0xFFFF.FFFF



位/域	名称	类型	复位	描述
31:0	TAV	RW	0xFFFF.FFFF	GPTM Timer A值寄存器 在所有模式下，读操作返回 Timer A 当前自由运行的值。向该寄存器写入的值将会在下一个时钟周期加载到 GPTMTAR 寄存器里。 注意: 在 16 位模式中，仅 GPTMTAV 寄存器的低 16 位可写入新值。向预分频器位写入值不会产生任何效果。

寄存器 21: GPTM Timer B 值寄存器（GPTMTBV），偏移量 0x054

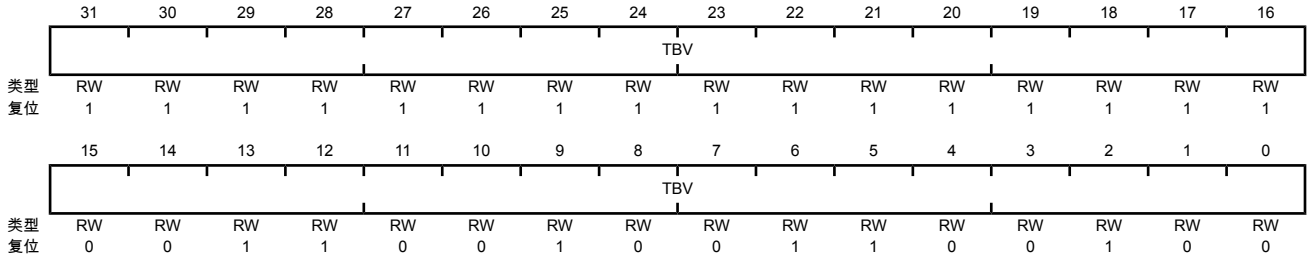
在所有模式下，读取该寄存器返回 Timer B 当前的自由运行值。软件可以根据该值来判断从发生中断到响应中断经历的时间。向该寄存器写入的值将会在下一个时钟周期加载到 GPTMTBR 寄存器里。

当 16/32 位 GPTM 配置为其中一种 32 位模式时，该寄存器中 15:0 位的内容被加载到 GPTMTAV 寄存器的高 16 位。读取该寄存器将返回 Timer B 的当前自由运行值。在 16 位模式中，15:0 位包含计数器的值，23:16 位包含预分频器当前自由运行的值，该值是输入边沿计数、输入边沿计时、PWM 和单次触发或周期递增计数模式的计数高 8 位。在单次触发或周期递减计数模式中，存储在 23:16 的预分频器是真预分频器，即 23:16 位在 15:0 位的值递减之前递减计数。31:24 位预分频器的回读值始终为 0。

当 32/64 位宽 GPTM 配置为其中一种 64 位模式时，GPTMTBV 包含 64 位定时器值的 63:32 位，GPTM Timer A 值 (GPTMTAV) 寄存器包含 31:0 位。在 32 位模式中，预分频器的当前自由运行值存储在 GPTM Timer B 预分频值 (GPTMTBPV) 寄存器中。

GPTM Timer B 值寄存器 (GPTMTBV)

16/32 位Timer 0 基址: 0x4003.0000
16/32 位Timer 1 基址: 0x4003.1000
16/32 位Timer 2 基址: 0x4003.2000
16/32 位Timer 3 基址: 0x4003.3000
16/32 位Timer 4 基址: 0x4003.4000
16/32 位Timer 5 基址: 0x4003.5000
32/64 位宽 Timer 0 基址: 0x4003.6000
32/64 位宽 Timer 1 基址: 0x4003.7000
32/64 位宽 Timer 2 基址: 0x4004.C000
32/64 位宽 Timer 3 基址: 0x4004.D000
32/64 位宽 Timer 4 基址: 0x4004.E000
32/64 位宽 Timer 5 基址: 0x4004.F000
偏移量 0x054
类型 RW, 复位 -



位/域	名称	类型	复位	描述
31:0	TBV	RW	0x0000.FFFF（适用于 16/32 位） 0xFFFF.FFFF（适用于 32/64 位）	GPTM Timer B值寄存器 在所有模式下，读操作返回 Timer A 当前自由运行的值。向该寄存器写入的值将会在下一个时钟周期加载到 GPTMTAR 寄存器里。 注意: 在 16 位模式中，仅 GPTMTBV 寄存器的低 16 位可写入新值。向预分频器位写入值不会产生任何效果。

寄存器 22: GPTM RTC 预分频寄存器（GPTMRTCPD），偏移量 0x058

定时器在 RTC 模式中运行时，该寄存器提供当前的 RTC 预分频器值。软件必须通过连续读取 GPTMTAR、GPTMTBR 和 GPTMRTCPD 寄存器执行原子访问。请参见图11-2（636页）了解更多信息。

GPTM RTC 预分频寄存器 (GPTMRTCPD)

16/32 位Timer 0 基址: 0x4003.0000
16/32 位Timer 1 基址: 0x4003.1000
16/32 位Timer 2 基址: 0x4003.2000
16/32 位Timer 3 基址: 0x4003.3000
16/32 位Timer 4 基址: 0x4003.4000
16/32 位Timer 5 基址: 0x4003.5000
32/64 位宽 Timer 0 基址: 0x4003.6000
32/64 位宽 Timer 1 基址: 0x4003.7000
32/64 位宽 Timer 2 基址: 0x4004.C000
32/64 位宽 Timer 3 基址: 0x4004.D000
32/64 位宽 Timer 4 基址: 0x4004.E000
32/64 位宽 Timer 5 基址: 0x4004.F000
偏移量 0x058
类型 RO, 复位 0x0000.7FFF

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	保留															
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	RTCPD															
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

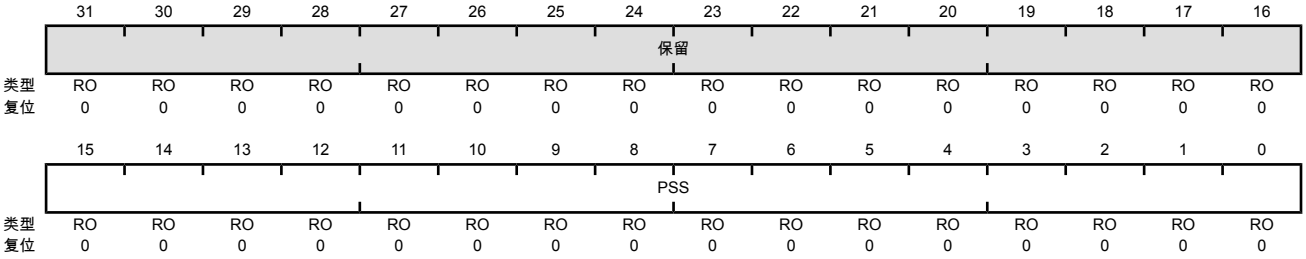
位/域	名称	类型	复位	描述
31:16	保留	RO	0x0000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
15:0	RTCPD	RO	0x0000.7FFF	RTC 预分频计数器值 定时器在 RTC 模式中运行时的当前 RTC 预分频器值。该域在其他定时器模式中无意义。

寄存器 23: GPTM Timer A 预分频快照寄存器（GPTMTAPS），偏移量 0x05C

对于 32/64 位宽 GPTM，该寄存器显示 32 位模式中 Timer A 预分频器的当前值。对于 16/32 位宽 GPTM，该寄存器显示周期快照模式中 Timer A 预分频器的当前值。

GPTM Timer A 预分频快照寄存器 (GPTMTAPS)

16/32 位Timer 0 基址: 0x4003.0000
16/32 位Timer 1 基址: 0x4003.1000
16/32 位Timer 2 基址: 0x4003.2000
16/32 位Timer 3 基址: 0x4003.3000
16/32 位Timer 4 基址: 0x4003.4000
16/32 位Timer 5 基址: 0x4003.5000
32/64 位宽 Timer 0 基址: 0x4003.6000
32/64 位宽 Timer 1 基址: 0x4003.7000
32/64 位宽 Timer 2 基址: 0x4004.C000
32/64 位宽 Timer 3 基址: 0x4004.D000
32/64 位宽 Timer 4 基址: 0x4004.E000
32/64 位宽 Timer 5 基址: 0x4004.F000
偏移量 0x05C
类型 RO, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:16	保留	RO	0x0000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
15:0	PSS	RO	0x0000	GPTM Timer A 预分频器快照 执行读操作将返回 GPTM Timer A 预分频器当前值。

寄存器 24: GPTM Timer B 预分频快照寄存器（GPTMTBPS），偏移量 0x060

对于 32/64 位宽 GPTM，该寄存器显示 32 位模式中 Timer B 预分频器的当前值。对于 16/32 位宽 GPTM，该寄存器显示周期快照模式中 Timer B 预分频器的当前值。

GPTM Timer B 预分频快照寄存器 (GPTMTBPS)

16/32 位Timer 0 基址: 0x4003.0000
16/32 位Timer 1 基址: 0x4003.1000
16/32 位Timer 2 基址: 0x4003.2000
16/32 位Timer 3 基址: 0x4003.3000
16/32 位Timer 4 基址: 0x4003.4000
16/32 位Timer 5 基址: 0x4003.5000
32/64 位宽 Timer 0 基址: 0x4003.6000
32/64 位宽 Timer 1 基址: 0x4003.7000
32/64 位宽 Timer 2 基址: 0x4004.C000
32/64 位宽 Timer 3 基址: 0x4004.D000
32/64 位宽 Timer 4 基址: 0x4004.E000
32/64 位宽 Timer 5 基址: 0x4004.F000
偏移量 0x060
类型 RO, 复位 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	保留															
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	PSS															
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

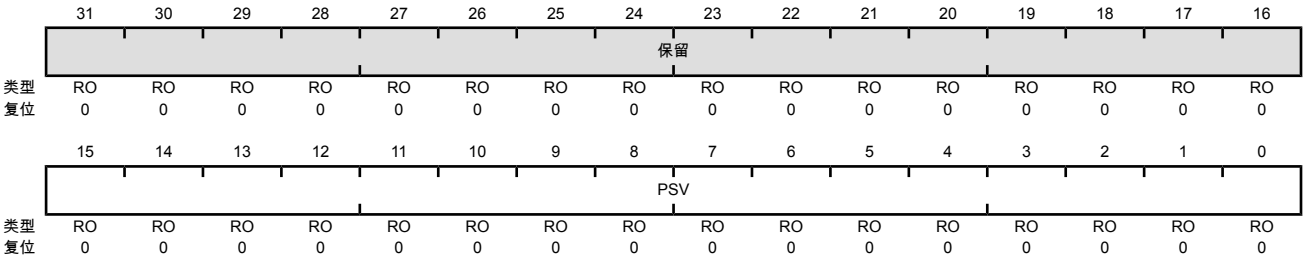
位/域	名称	类型	复位	描述
31:16	保留	RO	0x0000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
15:0	PSS	RO	0x0000	GPTM Timer A 预分频器值 执行读操作将返回 GPTM Timer A 预分频器当前值。

寄存器 25: GPTM Timer A 预分频值寄存器（GPTMTAPV），偏移量 0x064

对于 32/64 位宽 GPTM，该寄存器显示 32 位模式中 Timer A 预分频器的当前自由运行值。结合 GPTMTAV 寄存器，软件可以使用该值来判断从发生中断到响应中断经历的时间。该寄存器不适用于 16/32 位 GPTM 模式。

GPTM Timer A 预分频值寄存器 (GPTMTAPV)

16/32 位Timer 0 基址: 0x4003.0000
16/32 位Timer 1 基址: 0x4003.1000
16/32 位Timer 2 基址: 0x4003.2000
16/32 位Timer 3 基址: 0x4003.3000
16/32 位Timer 4 基址: 0x4003.4000
16/32 位Timer 5 基址: 0x4003.5000
32/64 位宽 Timer 0 基址: 0x4003.6000
32/64 位宽 Timer 1 基址: 0x4003.7000
32/64 位宽 Timer 2 基址: 0x4004.C000
32/64 位宽 Timer 3 基址: 0x4004.D000
32/64 位宽 Timer 4 基址: 0x4004.E000
32/64 位宽 Timer 5 基址: 0x4004.F000
偏移量 0x064
类型 RO, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:16	保留	RO	0x0000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
15:0	PSV	RO	0x0000	GPTM Timer A 预分频器值 执行读操作将返回 Timer A 寄存器的当前自由运行值。

寄存器 26: GPTM Timer B 预分频值寄存器（GPTMTBPV），偏移量 0x068

对于 32/64 位宽 GPTM，该寄存器显示 32 位模式中 Timer B 预分频器的当前自由运行值。结合 GPTMTBV 寄存器，软件可以使用该值来判断从发生中断到响应中断经历的时间。该寄存器不适用于 16/32 位 GPTM 模式。

GPTM Timer B 预分频值寄存器 (GPTMTBPV)

16/32 位Timer 0 基址: 0x4003.0000
16/32 位Timer 1 基址: 0x4003.1000
16/32 位Timer 2 基址: 0x4003.2000
16/32 位Timer 3 基址: 0x4003.3000
16/32 位Timer 4 基址: 0x4003.4000
16/32 位Timer 5 基址: 0x4003.5000
32/64 位宽 Timer 0 基址: 0x4003.6000
32/64 位宽 Timer 1 基址: 0x4003.7000
32/64 位宽 Timer 2 基址: 0x4004.C000
32/64 位宽 Timer 3 基址: 0x4004.D000
32/64 位宽 Timer 4 基址: 0x4004.E000
32/64 位宽 Timer 5 基址: 0x4004.F000
偏移量 0x068
类型 RO, 复位 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	保留															
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	PSV															
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

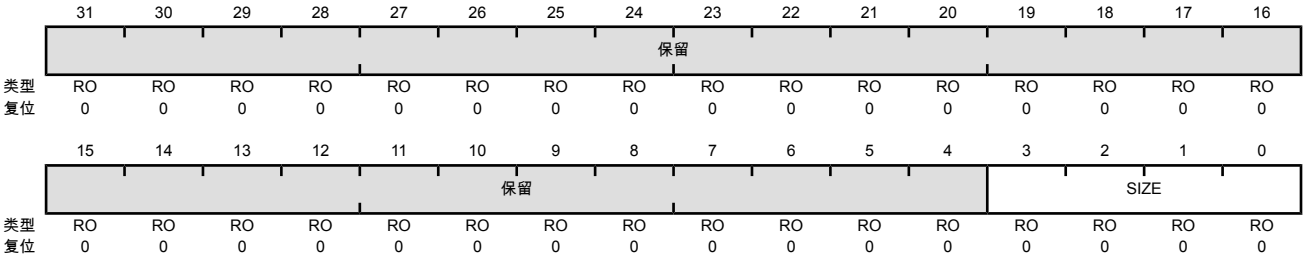
位/域	名称	类型	复位	描述
31:16	保留	RO	0x0000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
15:0	PSV	RO	0x0000	GPTM Timer B 预分频器值 执行读操作将返回 Timer A 寄存器的当前自由运行值。

寄存器 27: GPTM 外设属性寄存器（GPTMPP），偏移量 0xFC0

GPTMPP 寄存器提供关于通用定时器模块属性的信息。

GPTM 外设属性寄存器 (GPTMPP)

16/32 位Timer 0 基址: 0x4003.0000
16/32 位Timer 1 基址: 0x4003.1000
16/32 位Timer 2 基址: 0x4003.2000
16/32 位Timer 3 基址: 0x4003.3000
16/32 位Timer 4 基址: 0x4003.4000
16/32 位Timer 5 基址: 0x4003.5000
32/64 位宽 Timer 0 基址: 0x4003.6000
32/64 位宽 Timer 1 基址: 0x4003.7000
32/64 位宽 Timer 2 基址: 0x4004.C000
32/64 位宽 Timer 3 基址: 0x4004.D000
32/64 位宽 Timer 4 基址: 0x4004.E000
32/64 位宽 Timer 5 基址: 0x4004.F000
偏移量 0xFC0
类型 RO, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:4	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
3:0	SIZE	RO	0x0	计数大小

值	描述
0	Timer A 和 Timer B 计数器为 16 位，各自具有 8 位预分频计数器。
1	Timer A 和 Timer B 计数器为 32 位，各自具有 16 位预分频计数器。

12 看门狗定时器

达到超时值时，看门狗定时器会发出不可屏蔽中断 (NMI)、常规中断或者复位信号。当系统由于软件错误或是由于因外部设备故障而无法按预期的方式响应的时候，使用看门狗定时器可以重新获得控制权。TM4C1231H6PM 微控制器有两个看门狗定时器模块，一个模块使用系统时钟计时（看门狗 定时器 0），另一个模块使用 PIOSC 计时（看门狗 定时器 1）。这两个模块是相同的，只是 WDT1 在不同的时钟域，因此需要同步器。因此，WDT1 在看门狗定时器控制寄存器 (WDTCTL) 里有一个位用来说明 WDT1 寄存器写操作何时完成。在开始新的访问之前可通过该位来确保上一次的访问已经结束。

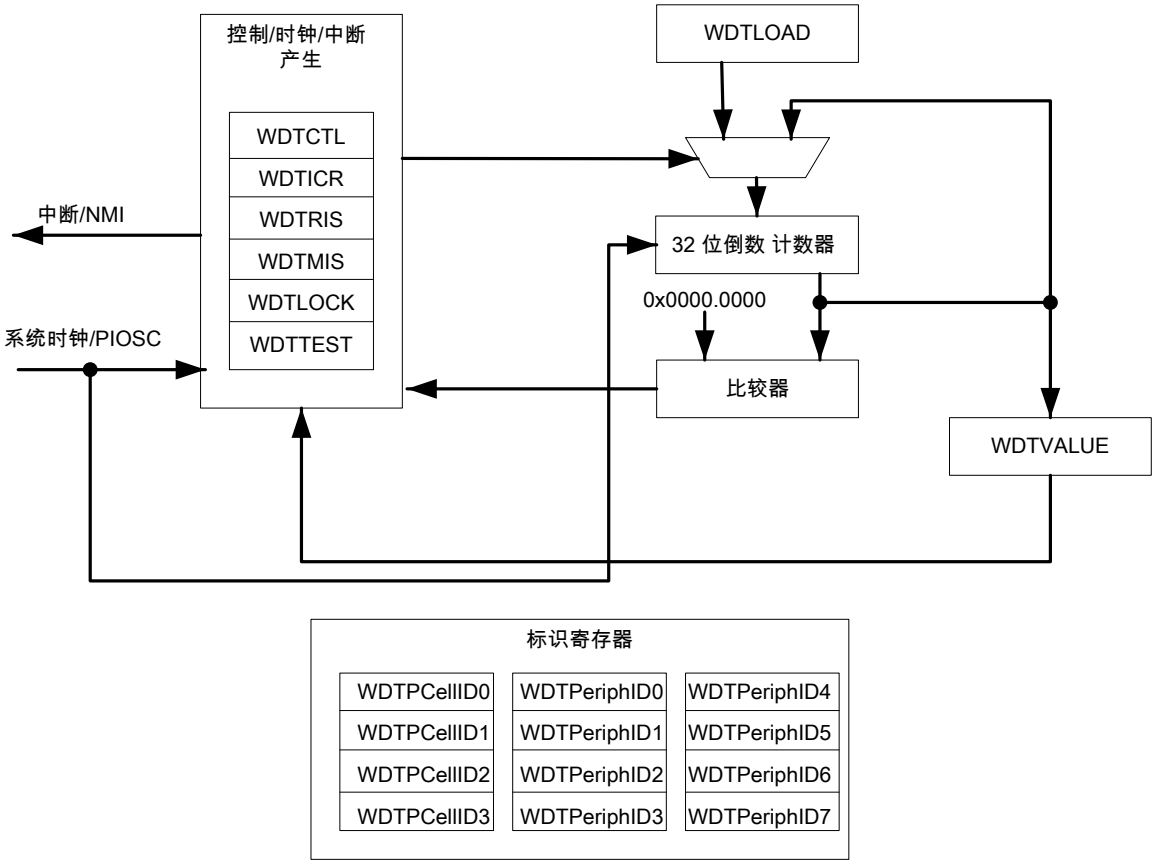
TM4C1231H6PM 控制器有两个具有如下特性的看门狗定时器模块：

- 32位递减并且可编程装载的寄存器
- 独立的看门狗时钟使能
- 带中断屏蔽功能和可选 NMI 功能的可编程中断产生逻辑
- 软件跑飞时保护锁定寄存器
- 复位使能/禁止产生逻辑
- 调试期间，微控制器的 CPU 暂停时，用户可启用的停滞

看门狗定时器可以配置为第一次超时的时候产生中断通知CPU，在第二次超时的时候产生一个重启信号。配置好看门狗定时器后，即可写入锁定寄存器，从而防止定时器配置被意外更改。

12.1 结构框图

图 12-1. WDT模块的结构图



12.2 功能说明

看门狗定时器有一个32为的计数器，当它使能后开始递减计数，当递减到0后就会产生第一个超时信号；在计数器使能后看门狗定时器的中断同时也被使能。使用 WDTCTL 寄存器的 INTTYPE 位可以将看门狗的中断配置为不可屏蔽中断 (NMI)。在第一次超时后，32 位计数器重新装载看门狗定时器装载寄存器 (WDTLOAD) 的值，并从该值开始恢复递减计数。一旦看门狗定时器被配置后，看门狗定时器锁定寄存器 (WDTLOCK) 被写入，从而防止软件意外更改看门狗定时器的配置。

如果在第一次超时中断被清除以前，计数器又递减到 0，并且看门狗定时器将控制寄存器 WDTCTL 里的 RESEN 位置位，使能了复位信号，看门狗定时器将把复位信号通知给系统。如果第二次超时发生之时，第一次超时中断已经清除，32 位计数器将会重新装载 WDTLOAD 寄存器里的值，并从该值开始恢复计数。

如果看门狗定时器正在计数的时候向 WDTLOAD 寄存器写入新值，计数器将会装载新值，并继续计数。

写 WDTLOAD 寄存器并不清除活动的中断。中断必须通过向看门狗中断清除寄存器 (WDTICR) 写入来专门清除。

看门狗的中断和复位信号可以根据需要来使能或禁止。当中断重新使能后，32位计数器将会预装载装载寄存器的值，而不是从它最后的状态开始计数。

复位后，看门狗定时器将默认被禁用。要对设备实现最大程度的看门狗保护，可在复位向量一开始就启用看门狗定时器。

12.2.1 寄存器访问间隙

因为看门狗定时器1(WDT1)模块具有一个独立的时钟域，所有在两次访问寄存器的时候要有一些时间间隙。软件必须保证，在连续的写入寄存器或是连续的读写寄存器操作之间插入足够的延时。WDT1对于连续的读寄存器的操作之间则没有限制。针对 WDT1 的看门狗控制寄存器 (WDTCTL) WRC 位表明所需时间差已过。WRC位表明了软件是否可以开始安全的读或写寄存器。软件在访问另一个寄存器之前必须查询 WDTCTL 寄存器的 WRC 位是否为 1。注意：WDT0没有该限制，因为它是运行在系统时钟下的。

12.3 初始化和配置

要使用 WDT，必须对看门狗定时器运行模式时钟门控控制寄存器 (RCGCWD) 中的 Rn 位进行置位，从而启用外设时钟，请参考292页。

看门狗定时器通过如下步骤来配置：

1. 为 WDTLOAD 寄存器装入所需的定时器负载值。
2. 如果是 WDT1，则等待 WDTCTL 寄存器的 WRC 位被置位。
3. 如果看门狗被配置为触发系统复位，则将 WDTCTL 寄存器里 RESEN 位置位。
4. 如果是 WDT1，则等待 WDTCTL 寄存器的 WRC 位被置位。
5. 将 WDTCTL 寄存器的 INTEN 位置位来使能看门狗，使能中断，并锁定控制寄存器。

如果软件需要锁定所有的看门狗寄存器，则写任意值到WDTLOCK寄存器便可以完全锁定看门狗定时器模块。如要要解锁看门狗寄存器，则需要向WDTLOCK寄存器写入0x1ACC.E551。

要维护看门狗，需要定期将计数值重新载入 WDTLOAD 寄存器以重启计数。如果看门狗维护不及时，可通过 WDTCTL 寄存器中的 INTEN 位启用中断，让处理器尝试实施纠正操作。如果使用 ISR 无法恢复故障，可置位 WDTCTL 中的 RESEN 位来复位系统。

12.4 寄存器映射

表 12-1（694页）列出了看门狗寄存器。所列偏移量是寄存器地址相对于看门狗定时器基址的 16 进制增量，该定时器基址为：

- WDT0：0x4000.0000
- WDT1：0x4000.1000

注意：必须先启用看门狗定时器模块的时钟，然后才能配置寄存器（请参阅292页）。

表 **12-1.** 看门狗定时器 寄存器映射

偏移量	名称	类型	复位	描述	见页面
0x000	WDTLOAD	R/W	0xFFFF.FFFF	看门狗加载寄存器	696
0x004	WDTVALUE	RO	0xFFFF.FFFF	看门狗当前值寄存器	697

表 12-1. 看门狗定时器 寄存器映射（续）

偏移量	名称	类型	复位	描述	见页面
0x008	WDTCTL	R/W	0x0000.0000 (WDT0) 0x8000.0000 (WDT1)	看门狗控制寄存器	698
0x00C	WDTICR	WO	-	看门狗中断清除寄存器	700
0x010	WDTRIS	RO	0x0000.0000	看门狗原始中断状态寄存器	701
0x014	WDTMIS	RO	0x0000.0000	看门狗可屏蔽中断状态寄存器	702
0x418	WDTTEST	R/W	0x0000.0000	看门狗测试寄存器	703
0xC00	WDTLOCK	R/W	0x0000.0000	看门狗锁定寄存器	704
0xFD0	WDTPeriphID4	RO	0x0000.0000	看门狗外设标识寄存器 4	705
0xFD4	WDTPeriphID5	RO	0x0000.0000	看门狗外设标识寄存器 5	706
0xFD8	WDTPeriphID6	RO	0x0000.0000	看门狗外设标识寄存器 6	707
0xFDC	WDTPeriphID7	RO	0x0000.0000	看门狗外设标识寄存器 7	708
0xFE0	WDTPeriphID0	RO	0x0000.0005	看门狗外设标识寄存器 0	709
0xFE4	WDTPeriphID1	RO	0x0000.0018	看门狗外设标识寄存器 1	710
0xFE8	WDTPeriphID2	RO	0x0000.0018	看门狗外设标识寄存器 2	711
0xFEC	WDTPeriphID3	RO	0x0000.0001	看门狗外设标识寄存器 3	712
0xFF0	WDTPCellID0	RO	0x0000.000D	看门狗 PrimeCell 标识寄存器 0	713
0xFF4	WDTPCellID1	RO	0x0000.00F0	看门狗 PrimeCell 标识寄存器 1	714
0xFF8	WDTPCellID2	RO	0x0000.0006	看门狗 PrimeCell 标识寄存器 2	715
0xFFC	WDTPCellID3	RO	0x0000.00B1	看门狗 PrimeCell 标识寄存器 3	716

12.5 寄存器描述

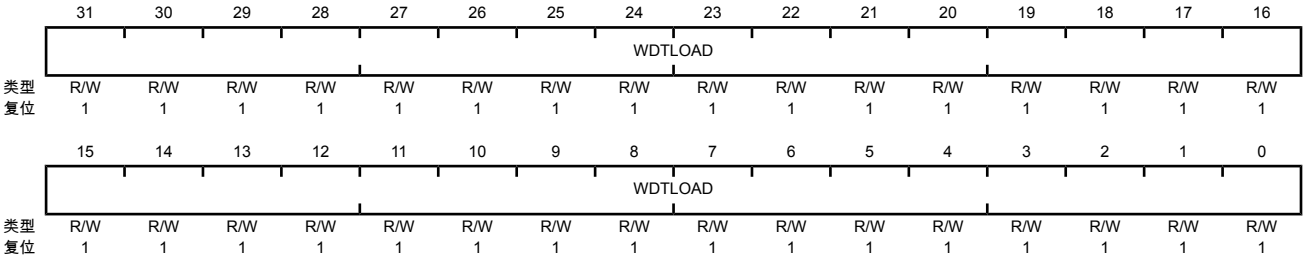
在本章剩余的部分，按寄存器偏移地址递增的顺序列出了看门狗定时器的寄存器，并详细描述了它们。

寄存器 1: 看门狗加载寄存器（WDTLOAD），偏移量 0x000

该寄存器是32位的计数器的32位间隔值。当该寄存器被写入的时候，该值立即被装载并从新值开始递减计数。当 WDTLOAD 寄存器的值装载为 0x0000.0000 时，会立即产生一个中断。

看门狗加载寄存器 (WDTLOAD)

WDT0 基址: 0x4000.0000
WDT1 基址: 0x4000.1000
偏移量 0x000
类型 R/W, 复位 0xFFFF.FFFF



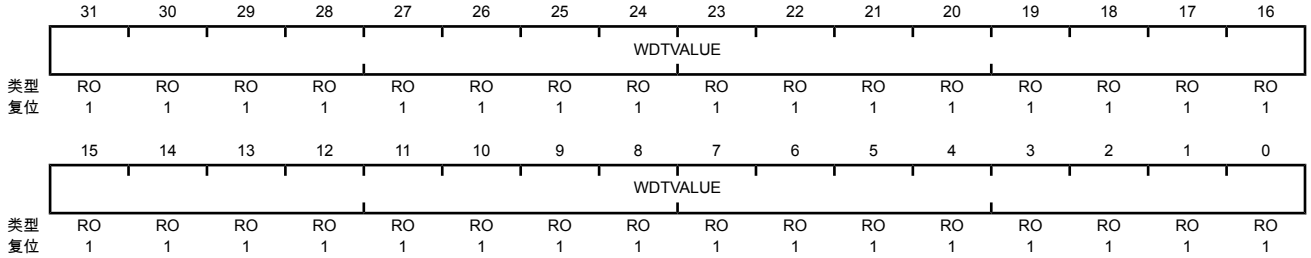
位/域	名称	类型	复位	描述
31:0	WDTLOAD	R/W	0xFFFF.FFFF	看门狗装载值

寄存器 2: 看门狗当前值寄存器（WDTVALUE），偏移量 0x004

该寄存器装载着定时器当前的值。

看门狗当前值寄存器 (WDTVALUE)

WDT0 基址: 0x4000.0000
WDT1 基址: 0x4000.1000
偏移量 0x004
类型 RO, 复位 0xFFFF.FFFF



位/域	名称	类型	复位	描述
31:0	WDTVALUE	RO	0xFFFF.FFFF	看门狗定时器当前值寄存器 32位递减计数器的当前值。

寄存器 3: 看门狗控制寄存器（WDTCTL），偏移量 0x008

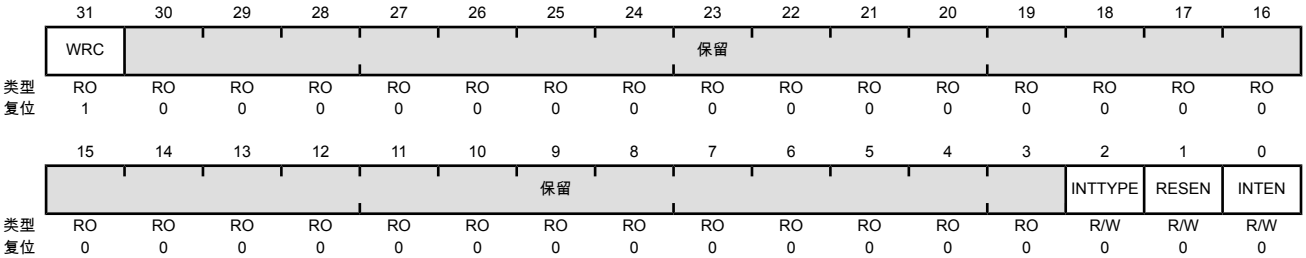
该寄存器是看门狗控制寄存器。看门狗可以产生复位信号(在第二次超时)或在超时时产生中断。

如果对 INTEN 位进行置位，从而启用了看门狗中断，之后对 INTEN 位的所有写入操作都将被忽略。只有两种方式能重新启用对该位的写入：硬件复位，或通过设置看门狗定时器软件复位寄存器(SRWD)的相应位来进行软件复位。

重要: 因为看门狗定时器1(WDT1)模块具有一个独立的时钟域，所有在两次访问寄存器的时候要有一些时间间隙。软件必须保证，在连续的写入寄存器或是连续的读写寄存器操作之间插入足够的延时。WDT1对于连续的读寄存器的操作之间则没有限制。针对 WDT1 的看门狗控制寄存器 (WDTCTL) WRC 位表明所需时间差已过。WRC位表明了软件是否可以开始安全的读或写寄存器。软件在访问另一个寄存器之前必须查询 WDTCTL 寄存器的 WRC 位是否为 1。注意：WDT0 没有该限制，因为它运行在系统时钟下，因此没有 WRC 位。

看门狗控制寄存器 (WDTCTL)

WDT0 基址: 0x4000.0000
WDT1 基址: 0x4000.1000
偏移量 0x008
类型 R/W, 复位 0x0000.0000 (WDT0) 和 0x8000.0000 (WDT1)



位/域	名称	类型	复位	描述
31	WRC	RO	1	写入操作完成 WRC 值定义如下： 值 描述 0 未产生此中断 1 产生了接收FIFO读取错误中断。 注意: 该位是为 WDT0 保留的，其复位值为 0。
30:3	保留	RO	0x000.000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
2	INTTYPE	R/W	0	看门狗中断类型 INTTYPE 值定义如下： 值 描述 0 看门狗定时器中断为标准中断。 1 看门狗定时器中断为不可屏蔽中断。

位/域	名称	类型	复位	描述
1	RESEN	R/W	0	看门狗复位使能 RESEN 值定义如下： 值 描述 0 禁止 1 使能看门狗复位输出
0	INTEN	R/W	0	看门狗中断使能 INTEN 值定义如下： 值 描述 0 禁用中断事件。该位一旦置位，只能由硬件复位或软件复位清零，其中软件复位需要通过设置看门狗定时器软件复位寄存器 (SRWD) 的相应位来触发。 1 启用中断事件。一旦启用，所有的写入操作都会被忽略。 将该位置位可使能看门狗定时器。

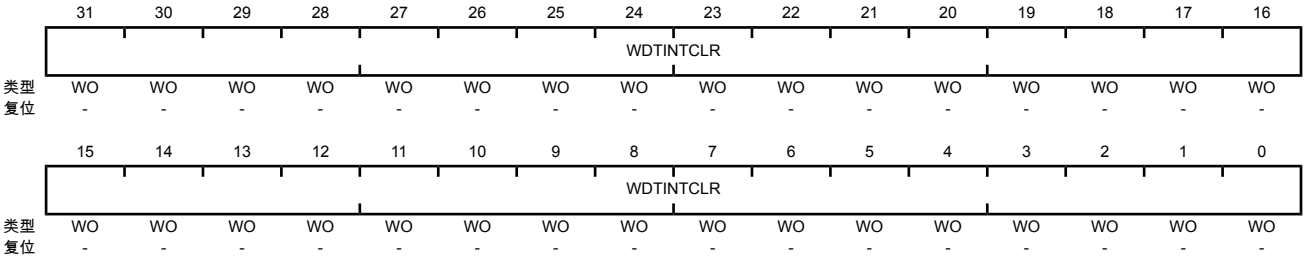
寄存器 4: 看门狗中断清除寄存器（WDTICR），偏移量 0x00C

该寄存器是中断清除寄存器。向该寄存器写入任何值都将清除看门狗中断，并且 32 位计数器将从 WDTLOAD 寄存器重新载入数据。在发生看门狗超时中断时写该寄存器，可以让系统进入看门狗中断服务。读取或复位后该寄存器的值时不确定的。

注意: 使用 WDTLOCK 寄存器锁定看门狗寄存器不会影响 WDTICR 寄存器，并且允许随时处理中断请求。因此，在任何时候写 WDTICR 寄存器会清除 WDTMIS 寄存器，并且 32 位计数器将从 WDTLOAD 寄存器重新载入数据。WDTICR 寄存器只有在中断被触发且需要进入中断服务时才能写入。

看门狗中断清除寄存器 (WDTICR)

WDT0 基址: 0x4000.0000
WDT1 基址: 0x4000.1000
偏移量 0x00C
类型 WO, 复位 -



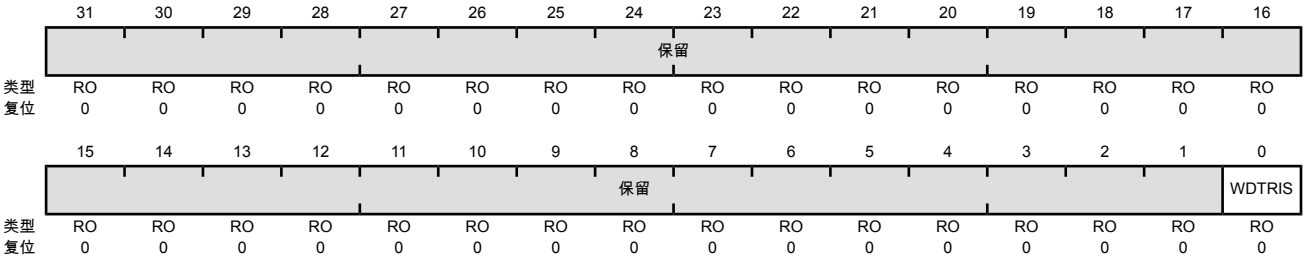
位/域	名称	类型	复位	描述
31:0	WDTINTCLR	WO	-	清除看门狗中断。 向该寄存器写入任何值都将清除看门狗中断，并且 32 位计数器将从 WDTLOAD 寄存器重新载入数据。在发生看门狗超时中断时写该寄存器，可以让系统进入看门狗中断服务。读取或复位后该寄存器的值时不确定的。

寄存器 5: 看门狗原始中断状态寄存器（WDTRIS），偏移量 0x010

该寄存器是看门狗的原始中断状态寄存器。如果中断被屏蔽了，可以通过该寄存器来监视看门狗的中断事件。

看门狗原始中断状态寄存器 (WDTRIS)

WDT0 基址: 0x4000.0000
WDT1 基址: 0x4000.1000
偏移量 0x010
类型 RO, 复位 0x0000.0000



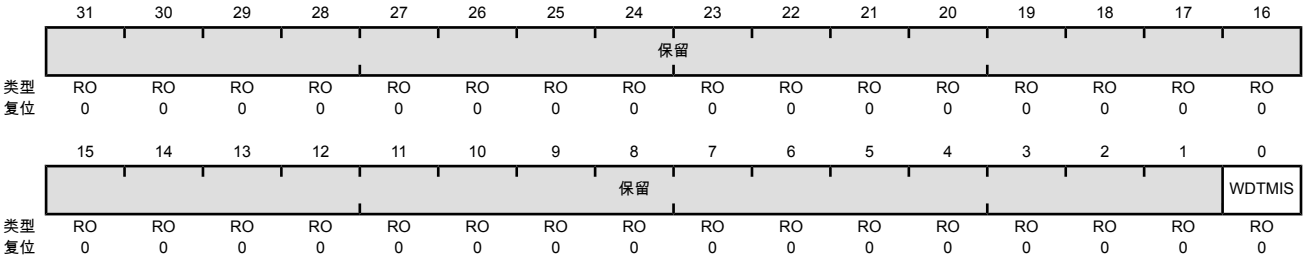
位/域	名称	类型	复位	描述
31:1	保留	RO	0x0000.000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
0	WDTRIS	RO	0	看门狗原始中断状态 值 描述 1 看门狗超时事件已经发生。 0 看门狗没有到达超时

寄存器 6: 看门狗可屏蔽中断状态寄存器（WDTMIS），偏移量 0x014

该寄存器是看门狗屏蔽后的中断状态寄存器。该寄存器的值是原始中断位和看门狗中断启用位逻辑与运算的结果。

看门狗可屏蔽中断状态寄存器 (WDTMIS)

WDT0 基址: 0x4000.0000
WDT1 基址: 0x4000.1000
偏移量 0x014
类型 RO, 复位 0x0000.0000



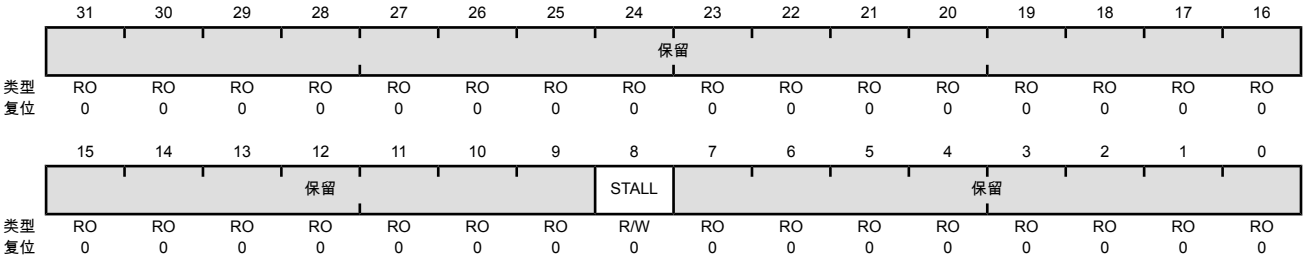
位/域	名称	类型	复位	描述
31:1	保留	RO	0x0000.000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
0	WDTMIS	RO	0	看门狗屏蔽后的中断状态 值 描述 1 看门狗超时事件已经向中断控制器发出信号 0 看门狗没有到达超时或是中断已经被屏蔽

寄存器 7: 看门狗测试寄存器（WDTTEST），偏移量 0x418

进行调试期间，当微控制器使CPU的暂停(Halt)标志有效时的暂停操作(stalling)可由用户通过该寄存器来控制使能。

看门狗测试寄存器 (WDTTEST)

WDT0 基址: 0x4000.0000
WDT1 基址: 0x4000.1000
偏移量 0x418
类型 R/W, 复位 0x0000.0000



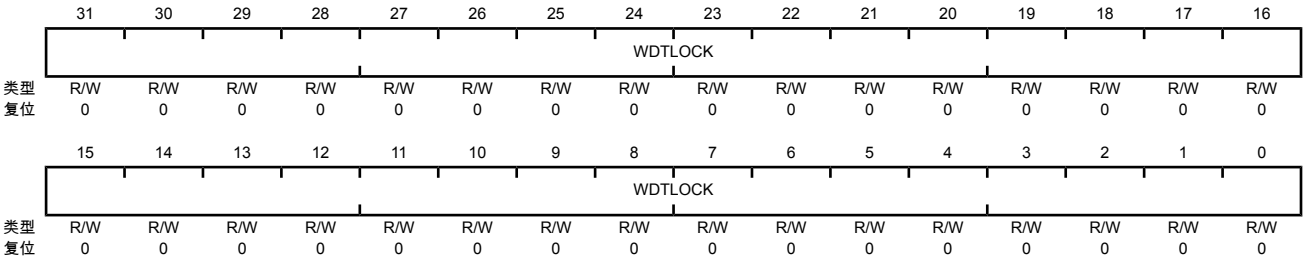
位/域	名称	类型	复位	描述
31:9	保留	RO	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
8	STALL	R/W	0	看门狗停滞使能 值 描述 1 如果微控制器因为调试而停止工作，则看门狗定时器停止计数。一旦微控制器重新启动，看门狗定时器也会恢复计数。 0 如果控制器因为调试停止，看门狗不会停止，将继续计数
7:0	保留	RO	0x00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。

寄存器 8: 看门狗锁定寄存器（WDTLOCK），偏移量 0xC00

向 WDTLOCK 寄存器写入 0x1ACC.E551 允许对其它所有寄存器执行写操作。在 WDTLOCK 寄存器中写入其它任意值则寄存器再次锁定，不能对所有其它寄存器执行写操作，但看门狗测试 (WDTTEST) 寄存器除外。锁定状态将在 2 个时钟周期后生效。读取 WDTLOCK 寄存器会返回锁定的状态，而不是写入的 32 位值。因此，当写入访问被禁止时，读取 WDTLOCK 寄存器将返回 0x0000.0001（这是在已锁定的情况下；否则，返回的值为 0x0000.0000 [未锁定]）。

看门狗锁定寄存器 (WDTLOCK)

WDT0 基址: 0x4000.0000
WDT1 基址: 0x4000.1000
偏移量 0xC00
类型 R/W, 复位 0x0000.0000



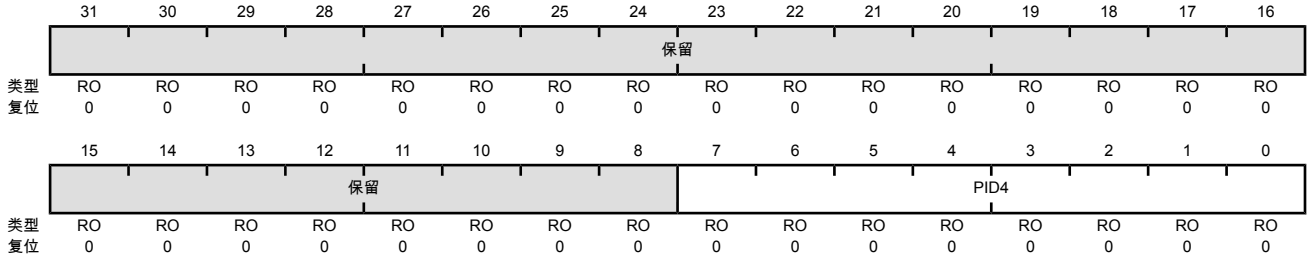
位/域	名称	类型	复位	描述
31:0	WDTLOCK	R/W	0x0000.0000	看门狗中锁定寄存器 写入值 0x1ACC.E551 可解除对看门狗寄存器的写入访问锁定。写入其它任何值会重新应用锁定，防止对寄存器进行更新，但 WDTTEST 寄存器除外。看门狗寄存器锁定时，可避免写 WDTTEST 寄存器。 读取该寄存器返回值具有如下的含义 值 描述 0x0000.0001 锁定 0x0000.0000 未锁定

寄存器 9: 看门狗外设标识寄存器 4（WDTPeriphID4），偏移量 0xFD0

WDTPeriphIDn 寄存器均为硬编码寄存器，寄存器的位域决定复位值。

看门狗外设标识寄存器 4 (WDTPeriphID4)

WDT0 基址: 0x4000.0000
WDT1 基址: 0x4000.1000
偏移量 0xFD0
类型 RO, 复位 0x0000.0000



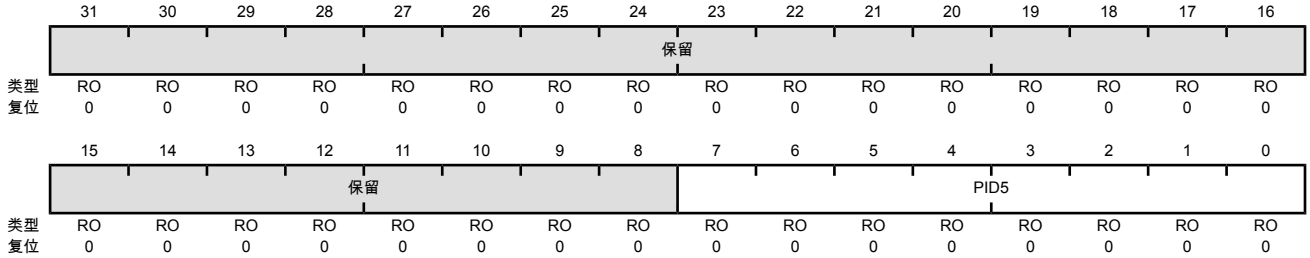
位/域	名称	类型	复位	描述
31:8	保留	RO	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
7:0	PID4	RO	0x00	看门狗外设标识寄存器[7:0]

寄存器 10: 看门狗外设标识寄存器 5（WDTPeriphID5），偏移量 0xFD4

WDTPeriphIDn 寄存器均为硬编码寄存器，寄存器的位域决定复位值。

看门狗外设标识寄存器 5 (WDTPeriphID5)

WDT0 基址: 0x4000.0000
WDT1 基址: 0x4000.1000
偏移量 0xFD4
类型 RO, 复位 0x0000.0000



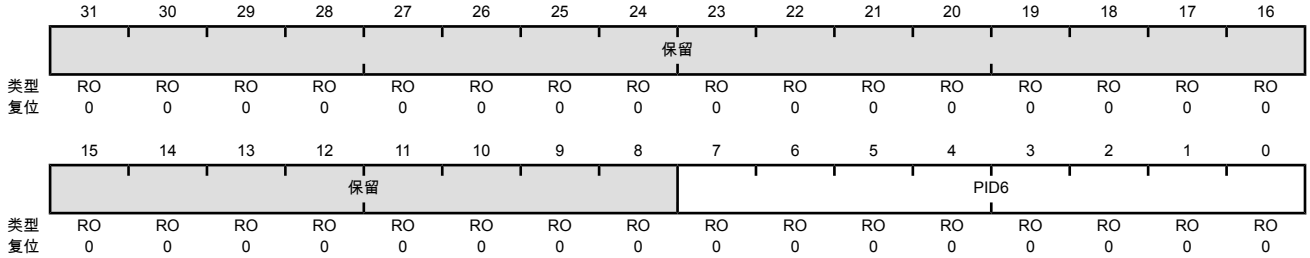
位/域	名称	类型	复位	描述
31:8	保留	RO	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
7:0	PID5	RO	0x00	看门狗外设标识寄存器[15:8]

寄存器 11: 看门狗外设标识寄存器 6（WDTPeriphID6），偏移量 0xFD8

WDTPeriphIDn 寄存器均为硬编码寄存器，寄存器的位域决定复位值。

看门狗外设标识寄存器 6 (WDTPeriphID6)

WDT0 基址: 0x4000.0000
WDT1 基址: 0x4000.1000
偏移量 0xFD8
类型 RO, 复位 0x0000.0000



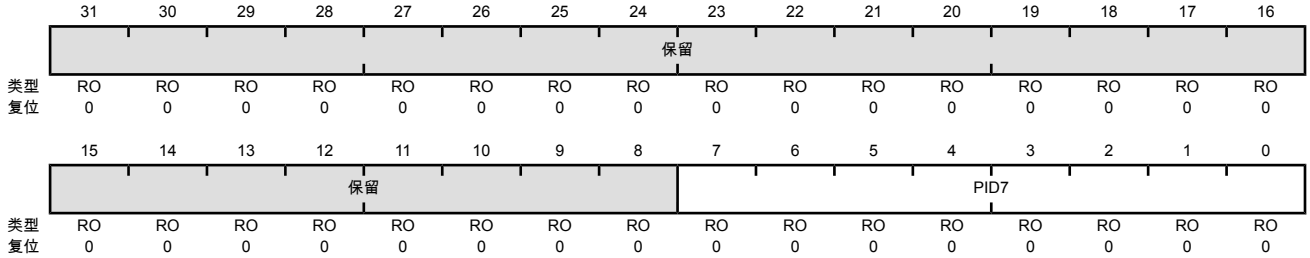
位/域	名称	类型	复位	描述
31:8	保留	RO	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
7:0	PID6	RO	0x00	WDT 外设标识寄存器 [23:16]

寄存器 12: 看门狗外设标识寄存器 7（WDTPeriphID7），偏移量 0xFDC

WDTPeriphIDn 寄存器均为硬编码寄存器，寄存器的位域决定复位值。

看门狗外设标识寄存器 7 (WDTPeriphID7)

WDT0 基址: 0x4000.0000
WDT1 基址: 0x4000.1000
偏移量 0xFDC
类型 RO, 复位 0x0000.0000



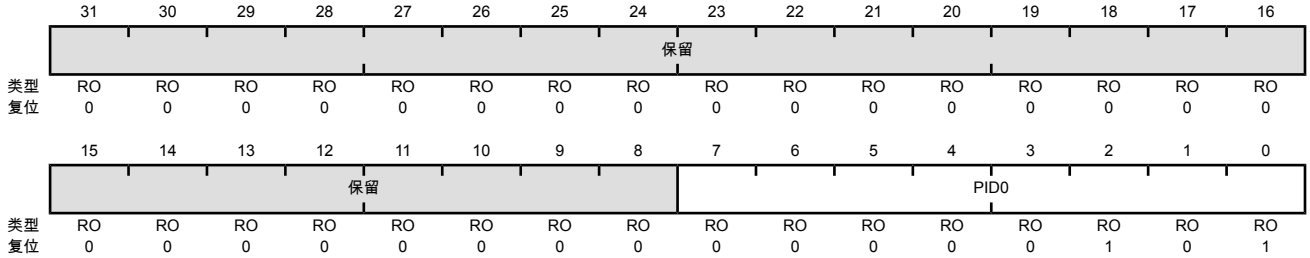
位/域	名称	类型	复位	描述
31:8	保留	RO	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
7:0	PID7	RO	0x00	看门狗外设标识寄存器[31:24]

寄存器 13: 看门狗外设标识寄存器 0（WDTPeriphID0），偏移量 0xFE0

WDTPeriphIDn 寄存器均为硬编码寄存器，寄存器的位域决定复位值。

看门狗外设标识寄存器 0 (WDTPeriphID0)

WDT0 基址: 0x4000.0000
WDT1 基址: 0x4000.1000
偏移量 0xFE0
类型 RO, 复位 0x0000.0005



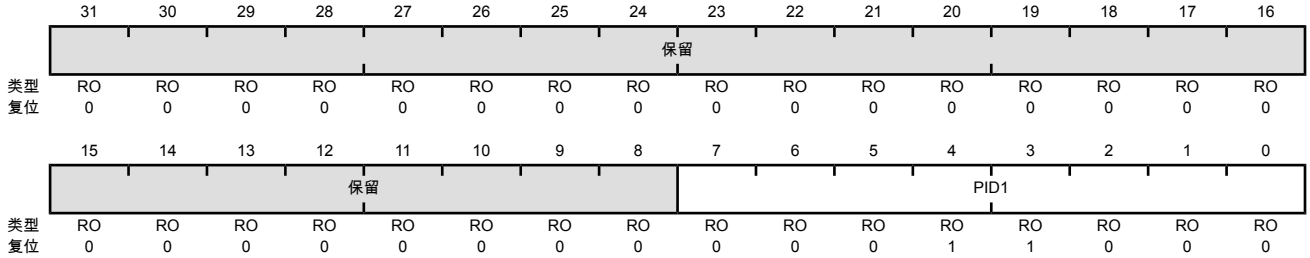
位/域	名称	类型	复位	描述
31:8	保留	RO	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
7:0	PID0	RO	0x05	看门狗外设标识寄存器 [7:0]

寄存器 14: 看门狗外设标识寄存器 1（WDTPeriphID1），偏移量 0xFE4

WDTPeriphIDn 寄存器均为硬编码寄存器，寄存器的位域决定复位值。

看门狗外设标识寄存器 1 (WDTPeriphID1)

WDT0 基址: 0x4000.0000
WDT1 基址: 0x4000.1000
偏移量 0xFE4
类型 RO, 复位 0x0000.0018



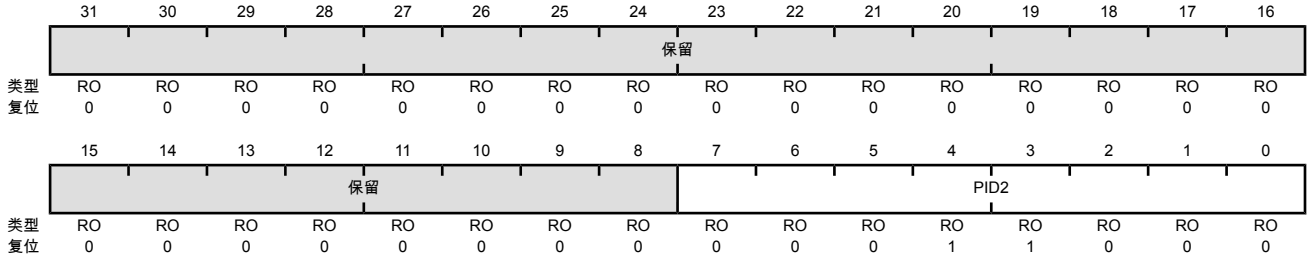
位/域	名称	类型	复位	描述
31:8	保留	RO	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
7:0	PID1	RO	0x18	看门狗外设标识寄存器 [15:8]

寄存器 15: 看门狗外设标识寄存器 2（WDTPeriphID2），偏移量 0xFE8

WDTPeriphIDn 寄存器均为硬编码寄存器，寄存器的位域决定复位值。

看门狗外设标识寄存器 2 (WDTPeriphID2)

WDT0 基址: 0x4000.0000
WDT1 基址: 0x4000.1000
偏移量 0xFE8
类型 RO, 复位 0x0000.0018



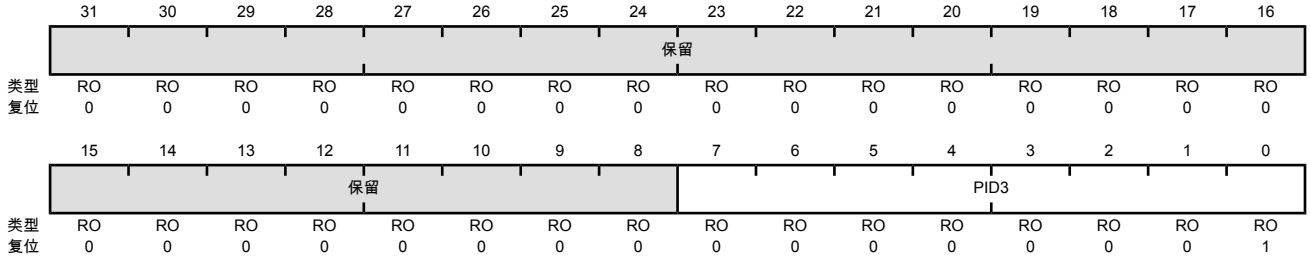
位/域	名称	类型	复位	描述
31:8	保留	RO	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
7:0	PID2	RO	0x18	看门狗外设标记寄存器 [23:16]

寄存器 16: 看门狗外设标识寄存器 3（WDTPeriphID3），偏移量 0xFEC

WDTPeriphIDn 寄存器均为硬编码寄存器，寄存器的位域决定复位值。

看门狗外设标识寄存器 3 (WDTPeriphID3)

WDT0 基址: 0x4000.0000
WDT1 基址: 0x4000.1000
偏移量 0xFEC
类型 RO, 复位 0x0000.0001



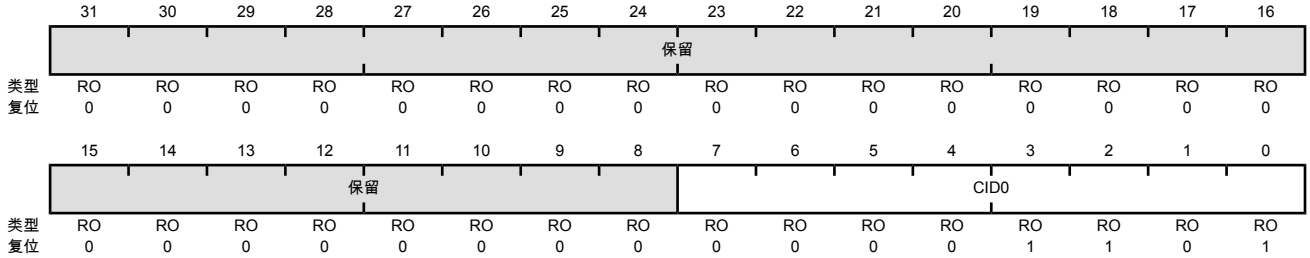
位/域	名称	类型	复位	描述
31:8	保留	RO	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
7:0	PID3	RO	0x01	看门狗外设标识寄存器 [31:24]

寄存器 17: 看门狗 PrimeCell 标识寄存器 0（WDTPCellID0），偏移量 0xFF0

WDTPCellIDn 寄存器均为硬编码寄存器，寄存器内的位域决定了复位值。

看门狗 PrimeCell 标识寄存器 0 (WDTPCellID0)

WDT0 基址: 0x4000.0000
WDT1 基址: 0x4000.1000
偏移量 0xFF0
类型 RO, 复位 0x0000.000D



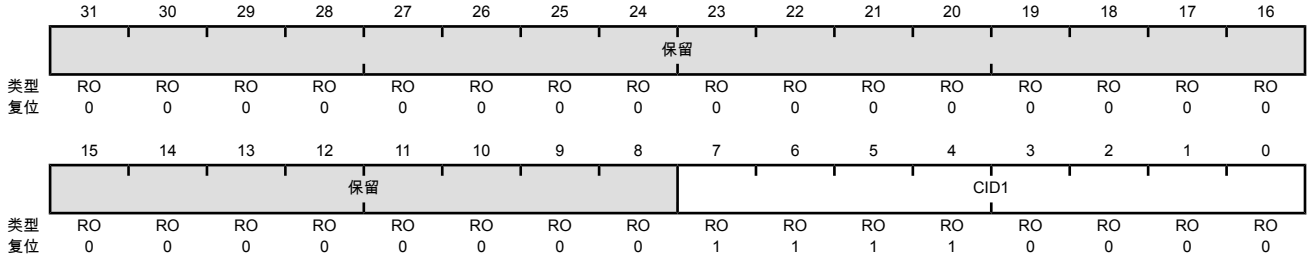
位/域	名称	类型	复位	描述
31:8	保留	RO	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
7:0	CID0	RO	0x0D	看门狗PrimeCell标识寄存器[7:0]

寄存器 18: 看门狗 PrimeCell 标识寄存器 1（WDTPCellID1），偏移量 0xFF4

WDTPCellIDn 寄存器为硬编码，寄存器内的位域决定了复位值。

看门狗 PrimeCell 标识寄存器 1 (WDTPCellID1)

WDT0 基址: 0x4000.0000
WDT1 基址: 0x4000.1000
偏移量 0xFF4
类型 RO, 复位 0x0000.00F0



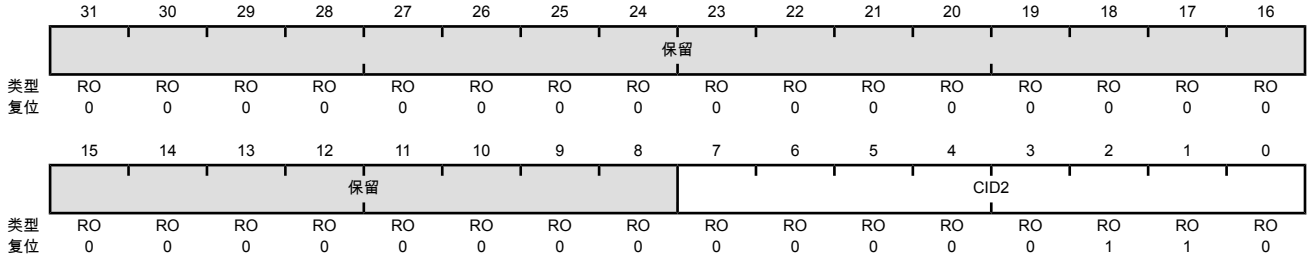
位/域	名称	类型	复位	描述
31:8	保留	RO	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
7:0	CID1	RO	0xF0	看门狗PrimeCell标识寄存器[15:8]

寄存器 19: 看门狗 PrimeCell 标识寄存器 2（WDTPCellID2），偏移量 0xFF8

WDTPCellIDn 寄存器为硬编码，寄存器内的位域决定了复位值。

看门狗 PrimeCell 标识寄存器 2 (WDTPCellID2)

WDT0 基址: 0x4000.0000
WDT1 基址: 0x4000.1000
偏移量 0xFF8
类型 RO, 复位 0x0000.0006



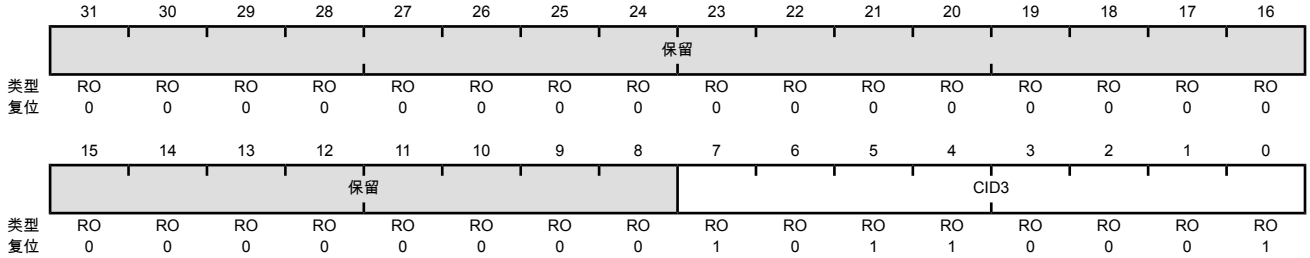
位/域	名称	类型	复位	描述
31:8	保留	RO	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
7:0	CID2	RO	0x06	看门狗 PrimeCell 标识寄存器 [23:16]

寄存器 20: 看门狗 PrimeCell 标识寄存器 3（WDTPCellID3），偏移量 0xFFC

WDTPCellIDn 寄存器为硬编码，寄存器内的位域决定了复位值。

看门狗 PrimeCell 标识寄存器 3 (WDTPCellID3)

WDT0 基址: 0x4000.0000
WDT1 基址: 0x4000.1000
偏移量 0xFFC
类型 RO, 复位 0x0000.00B1



位/域	名称	类型	复位	描述
31:8	保留	RO	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
7:0	CID3	RO	0xB1	看门狗PrimeCell标识寄存器[31:24]

13 模-数转换器（ADC）

模-数转换器（ADC）是一种能够将连续的模拟电压信号转换为离散的数字量的外设。包含两个完全相同的转换器模块，它们共用 12 个输入通道。

该 TM4C1231H6PM ADC 模块的转换分辨率为 12 位，并提供 12 个输入通道和一个内部温度传感器。每个 ADC 模块都包含 4 个可编程的序列发生器，无需控制器干预即可自动完成对多个模拟输入源的采样。每个采样序列发生器都可灵活配置其输入源、触发事件、中断的产生、序列发生器的优先级等内容。此外，还可选择将转换结果转移给数字比较器模块。每个 ADC 模块提供 8 个 数字比较器。每个数字比较器模块内置16 路数字比较器，每路数字比较器均可将 ADC 转换结果数值与 2 个由用户定义的门限值进行比较，以确定信号的工作范围。ADC0 和 ADC1 可各自采用不同的触发源，也可采用相同的触发源；可各自采用不同的模拟输入端，也可采用同一模拟输入端。ADC模块内部还具有移相器，可将采样开始时间（采样点）延后指定的相角。因此当两个ADC模块同时工作时，其采样点既可以配置为同相工作，也可以配置为相互错开一定的相角，详见 “采样相位控制”（721页）。

该 TM4C1231H6PM微处理器提供 2 个 ADC 模块，每个模块都具有以下特性：

- 12 个共用模拟输入通道
- 12 位精度的 ADC
- 可配置为单端输入或差分输入；
- 片上内置温度传感器
- 1M 次/秒的采样率
- 可选的移相器，采样点以采样周期计可延后 22.5° 到 337.5°
- 4 个可编程的采样转换序列发生器，序列长度 1 到 8 个单元不等，且各自带有相应长度的转换结果 FIFO
- 灵活的转换触发控制：
  - 控制器（软件）触发
  - 定时器触发
  - 模拟比较器触发
  - GPIO
- 硬件可对多达 64 个采样值进行平均计算
- 八个数字比较器
- 模拟部分的电源/地与数字部分的电源/地相互独立
- 用微型直接内存访问 (µDMA) 有效的传输数据
  - 每个采样序列发生器各自有专用的通道
  - ADC 模块的 DMA 操作均采用猝发请求

13.1 结构框图

该 TM4C1231H6PM 微控制器内置两个相同的模数转换器 (ADC) 模块。这两个模块（ADC0 和 ADC1）共用相同的 12 个模拟输入通道。两个ADC模块的工作相互独立，因此可同时执行不同的采样序列、随时对任一模拟输入通道进行采样、并各自产生不同的中断和触发事件。图13-1（718页）显示了这两个 ADC 模块是如何与模拟输入端以及系统总线连接的。

图 **13-1.** 两个 **ADC** 模块的连接结构图

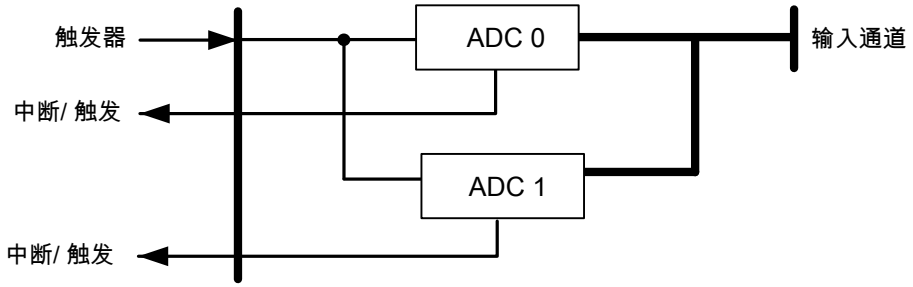
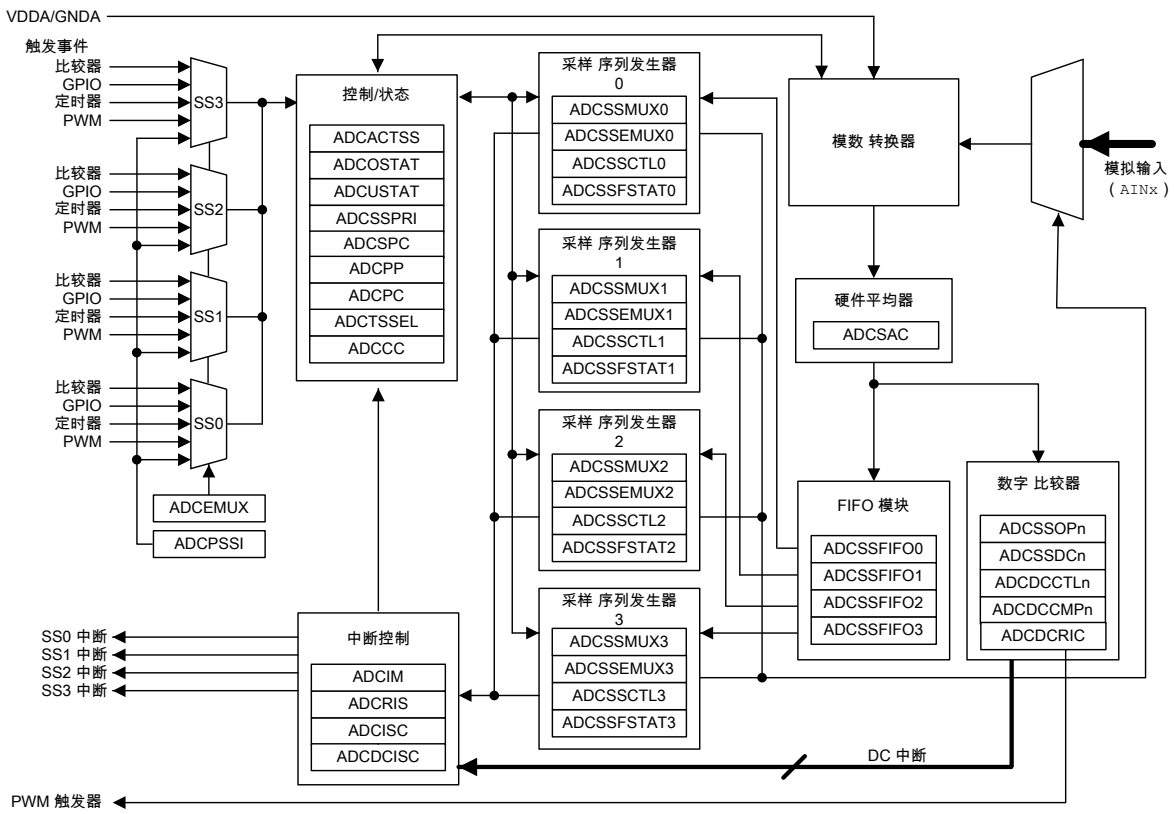


图13-2（718页）显示了 ADC 模块中控制寄存器及数据寄存器的内部配置情况。

图 **13-2. ADC**模块框图



13.2 信号描述

下表列出了与 ADC 模块相关的所有外部信号并逐一描述其功能。AINx 信号是某些 GPIO 信号的模拟功能。表中“复用管脚/赋值”一列是各ADC信号所对应的GPIO管脚。这些信号的配置方式为：将

GPIO 数字输入使能 (GPIODEN) 寄存器中相应的 DEN 位清零，并将 GPIO 模拟模式选择 (GPIOAMSEL) 寄存器中相应的 AMSEL 位置位。有关如何配置 GPIO 的更多信息，请参阅“通用输入/输入端口（GPIOs）”（575页）。

表 **13-1. ADC** 信号 **(64LQFP)**

管脚名称	管脚编号	管脚复用/管脚赋值	管脚类型	缓冲区类型[a]	描述
AIN0	6	PE3	I	模拟	模数转换器输入0。
AIN1	7	PE2	I	模拟	模数转换器输入1。
AIN2	8	PE1	I	模拟	模数转换器输入2。
AIN3	9	PE0	I	模拟	模数转换器输入3。
AIN4	64	PD3	I	模拟	模数转换器输入4。
AIN5	63	PD2	I	模拟	模数转换器输入5。
AIN6	62	PD1	I	模拟	模数转换器输入6。
AIN7	61	PD0	I	模拟	模数转换器输入7。
AIN8	60	PE5	I	模拟	模数转换器输入8。
AIN9	59	PE4	I	模拟	模数转换器输入9。
AIN10	58	PB4	I	模拟	模数转换器输入10。
AIN11	57	PB5	I	模拟	模数转换器输入11。

a. TTL 表示管脚的电压水平与 TTL 一致。

13.3 功能说明

该 TM4C1231H6PM ADC 通过使用一种基于序列的可编程方法来收集采样数据，取代了许多传统 ADC 模块使用的单次采样或双采样的方法。每个*采样序列（Sample Sequence）*均由一组编程的连续（背靠背）采样组成，因此 ADC 模块可以自动从多个输入源采集数据，无需处理器对其重新配置或进行干预。采样序列中的每个采样动作都可灵活编程，可配置的参数包括选择输入源和输入模式（单端输入或差分输入）、采样结束时是否产生中断、是否是队列中最后一个采样动作的标识符等等。此外，若结合 µDMA 工作，ADC模块能够更加高效地从采样序列中获取数据，同时无需 CPU 进行任何干预。

13.3.1 采样序列发生器

采样控制和数据采集都是由采样序列发生器（Sample Sequencer，简写为SS）处理的。所有序列发生器的实现方法都是相同的，区别仅在于能够捕捉的采样数以及FIFO深度有所不同。表 13-2（719页）给出了每个序列发生器可捕获的最大采样数及其相对应的 FIFO 深度。捕捉到的每个采样都要存入 FIFO 中。在本实现方案中，每个 FIFO 单元均为一个 32 位的字，低 12 位包含的是转换结果。

表 **13-2.** 采样序列发生器的采样数和 **FIFO** 深度

序列发生器	采样数	**FIFO** 深度
SS3	1	1
SS2	4	4
SS1	4	4
SS0	8	8

对于指定的采样序列，若以 n 代表其序号，则采样序列 n 中的每个采样动作分别以 ADC 采样序列输入多路复用器选择 (ADCSSMUXn)、及 ADC 采样序列控制 (ADCSSCTLn) 中的 1 个半字节予以定义。该 ADCSSMUXn 用于选择输入管脚，而 ADCSSCTLn 包含采样控制位，这些控制位分别与

参数（例如，温度传感器的选择、中断启用、序列末端和差分输入模式）对应。采样序列发生器可以通过置位 ADC活动采样序列发生器 (ADCACTSS) 寄存器中相应的 ASENn 位进行启用，但也可以在启用之前进行配置。软件可通过置位 ADC 处理器采样序列启动 (ADCPSSI) 寄存器的 SSn 位来启动采样。此外，在配置各个 ADC 模块时，通过设置 ADCPSSI 寄存器的 GSYNC 和 SYNCWAIT 位同时启动多个 ADC 模块的采样序列。关于这些配置位的详细信息，请参阅 756页。

配置采样序列时，允许同一序列中的多个采样动作对同一输入端进行采样。ADCSSCTLn 寄存器中的 IEn 位可针对任意采样动作组合置位，如此可在必要时允许在采样序列的每个采样动作后产生中断。同样，END 位也可在采样序列的任意时刻置位。举例来说，假设使用采样序列 0，那么可在与第 5 个采样动作相关的半字中将 END 位置位，从而使采样序列 0 在完成第 5 个采样动作后结束整个采样序列。

当采样序列执行结束后，可从 ADC 采样序列结果 FIFO (ADCSSFIFOn) 寄存器中读取采样结果数据。ADC模块的FIFO均为简单的环型缓冲区，反复读取同一地址（ADCSSFIFOn）即可依次“弹出”结果数据。为了方便软件调试，通过 ADC 采样序列 FIFO 状态 (ADCSSFSTATn) 寄存器可查询到 FIFO 头指针和尾指针的位置以及 FULL 和 EMPTY 状态标志。如果 FIFO 已满，再进行写操作时，该写操作会失败，该 FIFO 会出现上溢状况。通过 ADCOSTAT 和 ADCUSTAT 寄存器可监控上溢和下溢状态。

13.3.2 模块控制

控制逻辑单元中除采样序列发生器的剩余部分负责执行以下任务：

- 中断的产生
- DMA 操作
- 采样序列按优先级执行
- 触发事件的配置
- 比较器的配置
- 采样相位控制
- 模块计时

大多数的 ADC 控制逻辑都以 16 MHz 的 ADC 时钟速率运行。当系统 XTAL 选择 PLL 时，硬件将自动配置内部的 ADC 分频器，以便按照 16 MHz 频率工作。

13.3.2.1 中断信号

采样序列发生器和数字比较器数字比较器的寄存器配置可以监控产生原始中断的事件，但对中断是否真正发送给中断控制器没有控制权。ADC 模块是否产生中断信号是由 ADC 中断掩码 (ADCIM) 寄存器的 MASK 位决定的。中断状态可以从以下两个位置查询：ADC 原始中断状态寄存器（ADCRIS）显示各个中断信号的原始状态；ADC中断及清除寄存器（ADCISC）显示经 ADCIM 寄存器启用后的实际中断状态。通过向 ADCISC 对应的 IN 位写 1 来清除中断。请注意，数字比较器中断不是通过本寄存器清除的，而是通过向 ADC 数字比较器中断状态及清除寄存器 (ADCDCISC) 的对应位写 1 来清除的。

13.3.2.2 DMA 操作

如果使用 DMA，则每个采样序列发生器能够独立工作，无需微控制器干预或重新配置即可传输数据，从而提高了效率。每个采样序列发生器都可向 µDMA 控制器中相关的专用通道发送请求。ADC 不支持单次的传输请求。当采样序列的中断标志置位时（ADCSSCTLn 寄存器的 IE 位置位）产生猝发传输请求。

µDMA 传输的仲裁大小必须是 2 的整数幂，并且 ADCSSCTLn 寄存器的相关 IE 位必须置位。例如，若 SS0 的 µDMA 通道大小为 4，那么必须将 IE3（第 4 个采样动作）和 IE7（第 8 个采样动作）置位。因此，每 4 个采样动作后会触发 1 次 µDMA 请求。除此之外不需要其它特殊步骤，ADC 模块已经能够进行 µDMA 工作。

关于 µDMA 控制器编程的更多信息，请参阅 “微型直接存储器访问（µDMA）”（512页）。

13.3.2.3 优先级

当同时发生多个采样事件（触发条件）时，将按照 ADC 采样序列器优先级 (ADCSSPRI) 寄存器中的值对它们进行排序和依次处理。优先级的有效值为0~3，其中0代表最高优先级、3代表最低优先级。如果多个活动的采样序列具有相同的优先级，将导致转换结果数据不连续，因此软件必须确保当前活动的所有采样序列各自具有唯一的优先级。

13.3.2.4 采样事件

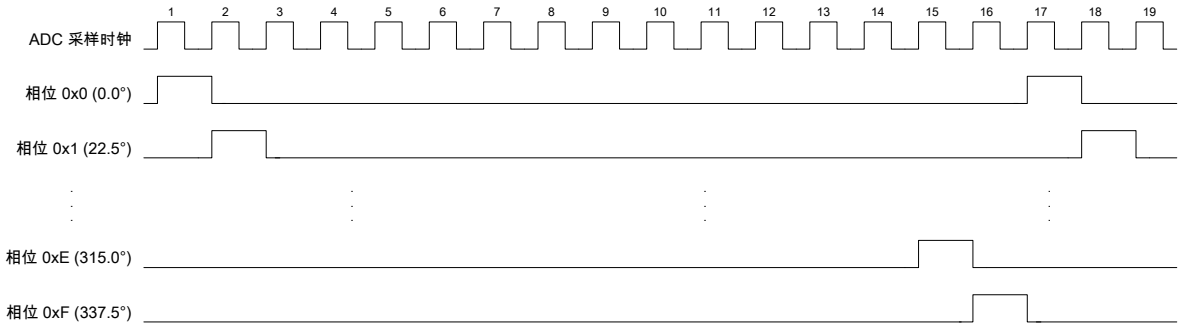
每个采样序列发生器的采样触发条件均通过 ADC 事件多路复用器选择 (ADCEMUX) 寄存器予以定义。触发事件源包括处理器触发（默认）、模拟比较器触发、GPIO ADC 控制 (GPIOADCCTL) 寄存器指定的 GPIO 外部信号触发、通用定时器触发、以及持续采样触发。软件可以将 ADC 处理器采样序列启动 (ADCPSSI) 的 SSx 位置位来启动采样序列。

配置持续采样触发条件时务必慎重。假如某个采样序列的优先级过高，可能导致其它低优先级采样序列始终无法运行。通常，要将使用连续采样模式的采样序列器设置位最低优先级。当输入接口上的电压达到了某一特定值，连续采样可以和数字比较器配合使用以产生中断。

13.3.2.5 采样相位控制

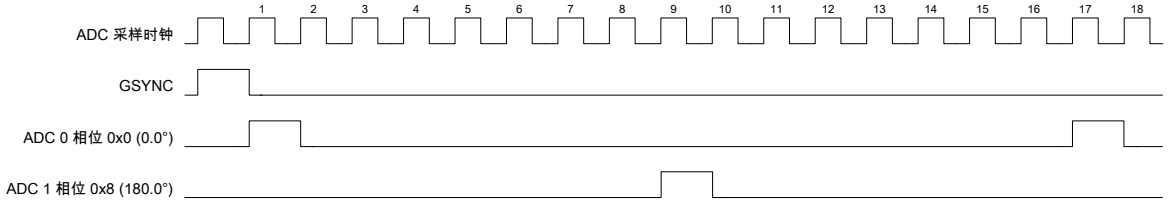
ADC0 和 ADC1 可各自采用不同的触发源，也可采用相同的触发源；可各自采用不同的模拟输入端，也可采用同一模拟输入端。假如两个转换器以相同的采样率工作，其采样点既可以配置为同相，也可以配置为相互错开一定的相角（可实现15种离散的相位差）。采样点延后的相位通过 ADC 采样相位控制 (ADCSPC) 寄存器按 22.5° 逐步递增，最大可递增至 337.5°。图13-3（721页） 显示了 1 Msps 采样率时各种不同的相位关系示例。

图 **13-3. ADC** 采样相位



借助此功能可让单个输入通道实现双倍采样率。将ADC0和ADC1模块配置为采用同一个输入通道，ADC0 模块可以按照标准相位采样（ADCSPC 寄存器的 PHASE = 0）。ADC1 模块可以配置为延后 180° 相位采样（PHASE = 0x8）。通过 ADC 处理器采样序列启动 (ADCPSSI) 寄存器的 GSYNC 和 SYNCWAIT 位可以将两个模块配置位同步运行。然后由软件将来自两个模块的结果数据进行组合，就能在 16 MHz （参见 图13-4（722页））工作频率下实现 1M 的采样率。

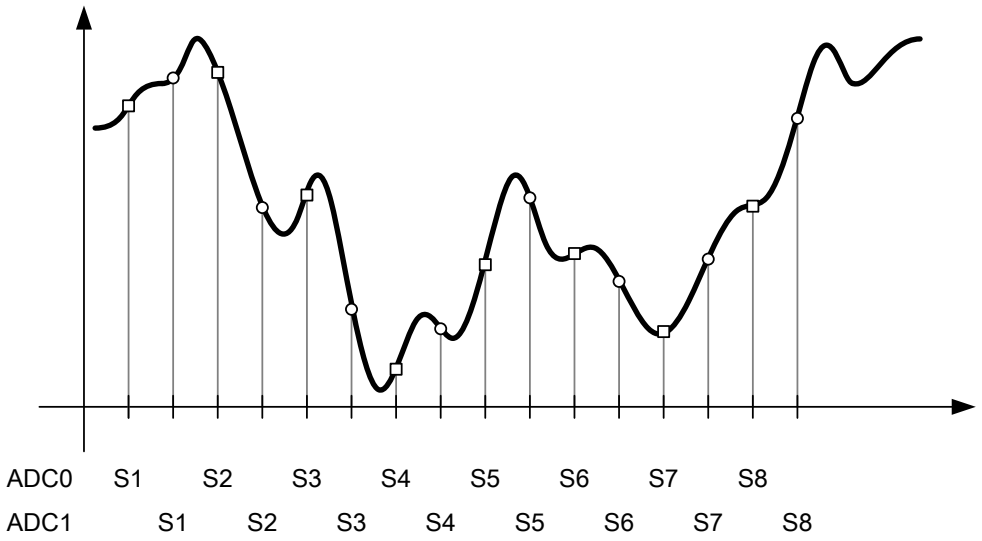
图 **13-4. ADC** 采样率倍增



使用 ADCSPC 寄存器，ADC0 和 ADC1 还能实现许多有趣的应用：

- 不同信号的同步持续采样。两个转换器的采样序列同相进行。
  - ADC 模块 0，ADCSPC = 0x0，采样 AIN0
  - ADC 模块 1，ADCSPC = 0x0，采样 AIN1
- 同一信号的交错采样。两个模块的采样序列交错进行异相采样，当采样率为 1 Msps 时，交错时间为 0.5 µs。在软件将转换结果进行交错组合时，如 图13-5（722页）所示，此配置可将单个输入通道的转换带宽加倍。
  - ADC 模块 0，ADCSPC = 0x0，采样 AIN0
  - ADC 模块 1，ADCSPC = 0x8，采样 AIN0

图 **13-5.** 交错采样



13.3.2.6 模块计时

该模块由一个 16-MHz 的时钟计时，该时钟可通过分频的 PLL 输出、PIOSC 或连接到 MOSC 的外部信号（PLL 处于旁路模式）获得。PLL 工作时，ADC 时钟通过 PLL ÷ 25 得到（默认）。但是，PIOSC 可用于使用 ADC 时钟配置 (ADCCC) 寄存器的模块时钟。要将 PIOSC 用作 ADC 的时钟源，首先应给 PLL 上电，然后通过 ADCCC 寄存器的 CS 位域启用 PIOSC，再禁用 PLL。PLL 处于旁路模式时，连接到 MOSC 的模块时钟源时钟必须为 16 MHz，除非该时钟源使用 PIOSC。要将 MOSC 用作 ADC 的时钟源，首先应给 PLL 上电，然后启用 ADC 模块的时钟，再禁用 PLL，最后

将系统时钟切换至 MOSC。如果 PIOSC 是 ADC 模块的时钟源，则 ADC 模块可以继续以深度睡眠模式工作。

系统时钟的频率必须与 ADC 时钟相同或更高。让所有 ADC 模块使用相同的时钟源有助于在多个转换设备间同步数据采样。可通过 ADCCC 寄存器的 ADC0 对时钟源进行选择和编程。ADC 模块无法以不同的转换速率运行。

13.3.2.7 繁忙状态

ADCACTSS 寄存器的 BUSY 位用于指示 ADC 目前正在紧张地执行转换。如果当前周期或者后面若干周期不存在可启动新转换的触发条件挂起，BUSY 位将为 0。在向模数转换器运行模式时钟门控控制 (RCGCADC) 寄存器写入数据以禁用 ADC 时钟之前，软件必须确认 BUSY 位已清零

13.3.2.8 启用抖动

ADCCTL 寄存器中的 DITHER 位用来减少 ADC 采样时的随机噪音，并将 ADC 运行保持在 “Analog-to-Digital Converter (ADC)” (1058页) 定义的具体性能限制内。在用 ADC 模块采集多个持续样本时，应启用 ADCCTL 寄存器中的 DITHER 位，以及 ADC 采样平均控制 (ADCSAC) 寄存器中的硬件均分功能。复位时，DITHER 位将默认被禁用。

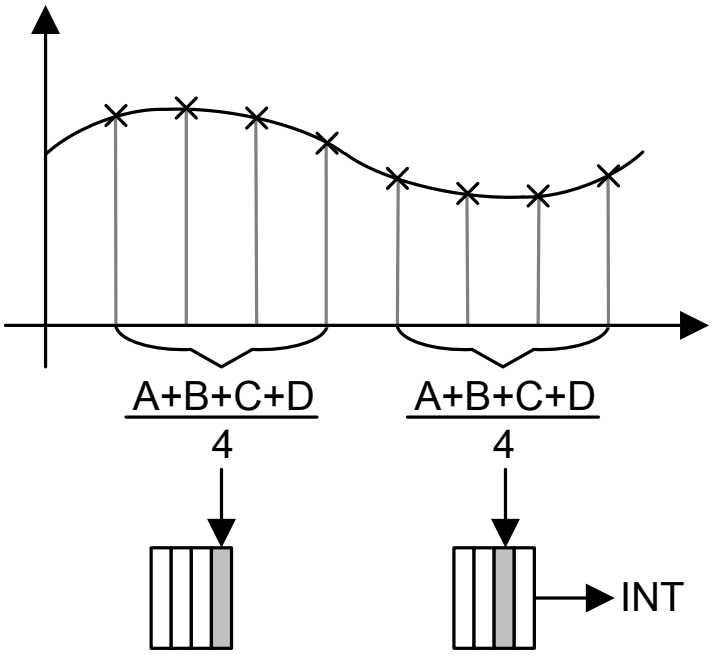
13.3.3 硬件采样平均电路

启用硬件采样平均电路可以获得更高的精度，与此同时付出的代价是吞吐率将成比例地降低。硬件采样平均电路最高可将64次采样结果累加并计算出平均值，以平均值作为单次采样的数据写入序列发生器FIFO的1个单元中。由于是算术平均值，因此吞吐率与求平均值的采样数目成反比。例如，若取16次采样进行平均值计算，那么吞吐率将降为1/16。

默认情况下，硬件采样平均电路是关闭的，转换器捕捉的所有数据直接送入序列发生器的FIFO中。进行平均计算的硬件由 ADC采样平均控制 (ADCSAC) 寄存器进行控制 (见 758页) 。每个ADC模块只有一个平均电路，不论单端输入还是差分输入都会被执行相同的求平均值操作。

图13-6 中显示一个实例。在该实例中，ADCSAC 寄存器设置为 0x2 以进行 4x 硬件过采样；IE1 被置位以提供采样队列；第 2 个平均值储存进 FIFO 后，将产生中断信号。

图 **13-6.** 采样平均的实例

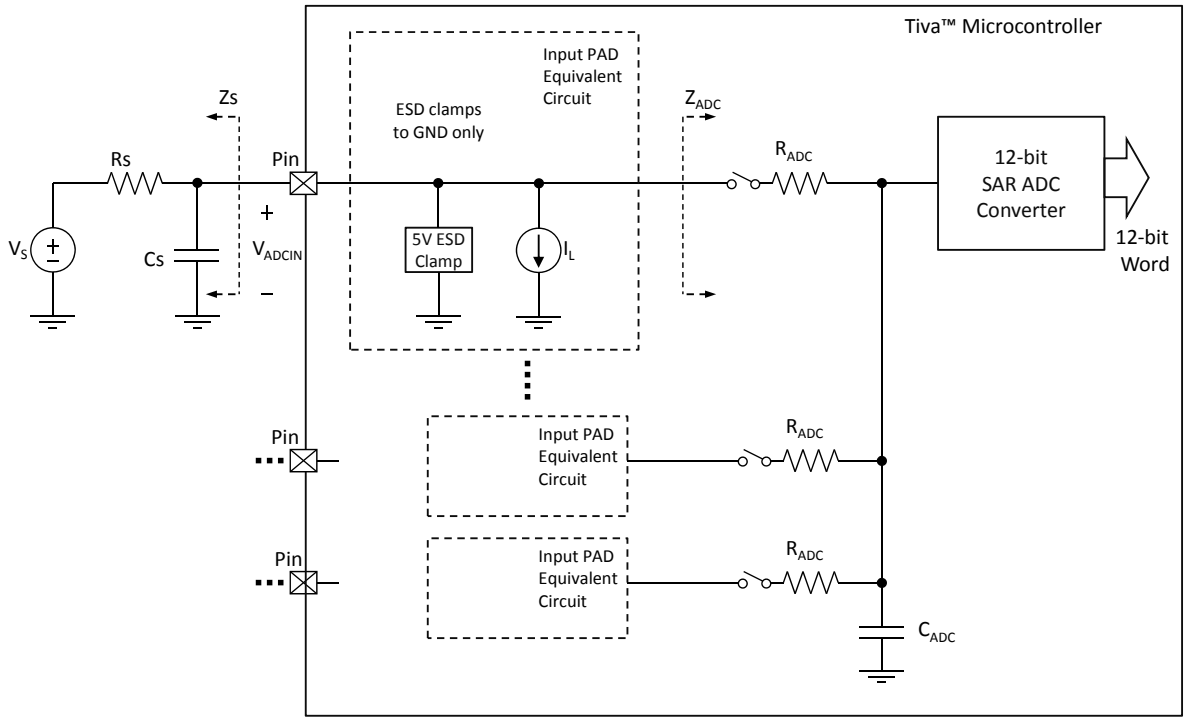


13.3.4 模-数转换器

模-数转换器（ADC）模块采用逐次逼近寄存器（Successive Approximation Register，简写为SAR）架构实现低功耗、高精度的12位A/D转换。该逐次逼近架构使用开关电容阵列执行两种功能：采集和保持信号，提供 12 位 DAC 操作。

图13-7 显示 ADC 输入端等效框图；参数值请参考 “Analog-to-Digital Converter (ADC)”（1058页）。

图 13-7. ADC 输入端等效框图

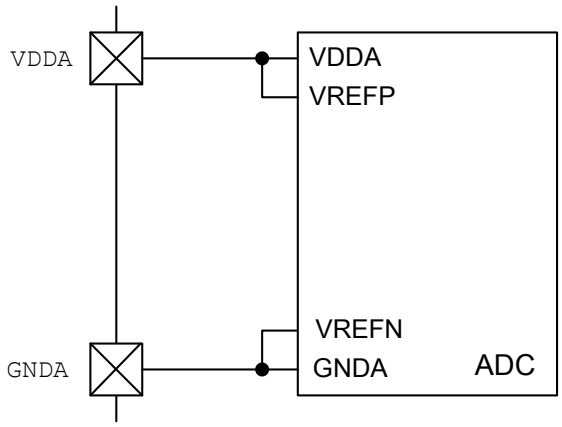


ADC模块同时从3.3V模拟电源和1.2V数字电源取电。在不要求 ADC 转换精度时，可以将 ADC 时钟配置为低功耗（请参阅“系统控制”（199页））。模拟信号通过特殊的平衡输入通道连接到ADC，尽量减少输入信号的失真和串扰。关于 ADC 模块供电及模拟输入的详细信息，请参阅 “Analog-to-Digital Converter (ADC)”（1058页）。

13.3.4.1 参考电压

ADC 使用内部信号 VREFP 和 VREFN 作为参考电压源，以对选定的模拟输入电压进行转换。VREFP 连接到 VDDA 而 VREFN 连接到 GNDA，如 图13-8 所示。

图 13-8. ADC 参考电压

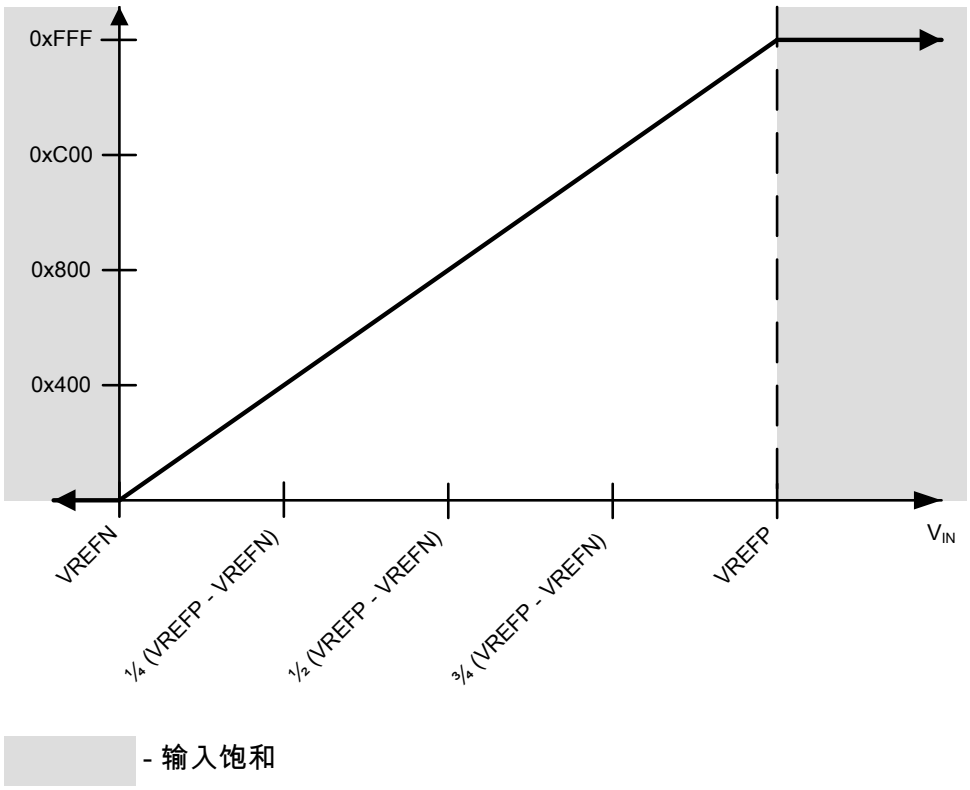


转换值的范围为 0x000~0xFFF。在单端输入模式下，0x000 对应于 VREFN 上的电平；0xFFF 对应于 VREFP 上的电平。通过这样的配置，分辨率就可以使用以下等式计算：

```
mV per ADC code = (VREFP - VREFN) / 4096
```

虽然模拟输入管脚能够处理超出此范围的电压，但为了确保结果精确，模拟输入电源必须处于表 21-32（1058页）规定的限制范围以内。图13-9（726页）中示出了 ADC 转换值与输入模拟电压的函数关系。

图 **13-9. ADC** 转换结果



13.3.5 差分采样

除了传统的单端采样，ADC模块还支持对两个模拟输入通道进行差分采样。要启用差分采样功能，软件必须在某个采样步骤的配置半字节中将 ADCSSCTL0n 寄存器的 Dn 位置位。

若采样序列中某个采样动作配置为差分采样，必须在 ADCSSMUXn 寄存器中配置输入的差分信号对。差分信号对 0 对模拟输入端 0 和 1 进行采样，差分信号对 1 对模拟输入端 2 和 3 进行采样，依此类推（见 表13-3（727页））。ADC 不支持差分信号对的随意组合，如模拟输入端 0 和模拟输入端 3 无法作为一对差分信号输入。

表 **13-3.** 差分采样对

差分信号对	模拟输入
0	0 和 1
1	2 和 3
2	4 和 5
3	6 和 7
4	8 和 9
5	10 和 11

差分模式下采样电压是奇数通道与偶数通道电压的差值：

- 正向输入电压：VIN+ = V_{IN_EVEN}（偶数通道电压）
- 负向输入电压：VIN- = V_{IN_ODD}（奇数通道电压）

差分输入电压定义为：VIN_D = VIN+ - VIN-，因此：

- 若 VIN_D = 0，则转换结果 = 0x800
- 若 VIN_D > 0，则转换结果 > 0x800（范围是 0x800 ~ 0xFFF）
- 若 VIN_D < 0，则转换结果 < 0x800（范围是 0–0x800）

使用差分采样时，还需考虑以下定义：

- 输入共模电压：VIN_{CM} = (VIN+ + VIN-) / 2
- 正向参考电压：VREFP
- 负向参考电压：VREFN
- 差分参考电压：$VREF_D$ = VREFP - VREFN
- 参考共模电压：$VREF_{CM}$ = (VREFP + VREFN) / 2

差分模式具备以下条件时效果最佳：

- V_{IN_EVEN} 和 V_{IN_ODD} 必须在（VREFP 至 VREFN）范围内，否则无法得到有效的转换结果
- 最大可能的差分输入摆幅或最大的差分范围为：-$VREF_D$ 至 +$VREF_D$，因此最大的峰峰差分输入信号为 (+$VREF_D$ - -$VREF_D$) = 2 * $VREF_D$= 2 * (VREFP - VREFN)
- 为了利用最大可能的差分输入摆幅，VIN_{CM} 应非常接近 $VREF_{CM}$，请参考 表21-32（1058页）。

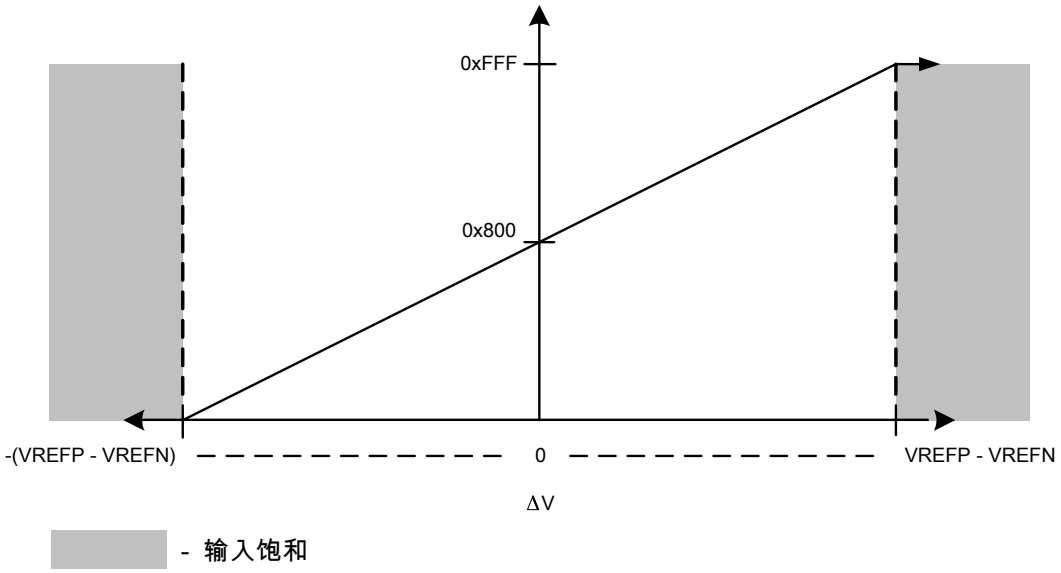
如果 VIN_{CM} 不等于 $VREF_{CM}$，那么在最大或最小电压下，差分输入信号可能减弱，（这是因为任何单端输入都不能大于 VREFP 或小于 VREFN），而且无法实现全摆幅。因此输入电压和参考电压之间的任何共模差异都会限制 ADC 的差分动态范围。

由于最大的峰峰差分信号电压为 2 * (VREFP - VREFN)，ADC 读数表示为：

```
mV per ADC code = (2 *(VREFP - VREFN)) / 4096
```

图13-10 显示如何通过 ADC 读数来表示差分电压 ΔV。

图 **13-10.** 差分电压表达式



13.3.6 内部温度传感器

温度传感器的主要作用有两个：1) 内部温度过高或过低时，向系统给予提示。2) 提供温度测试方法来校准休眠模块 RTC 调整值。

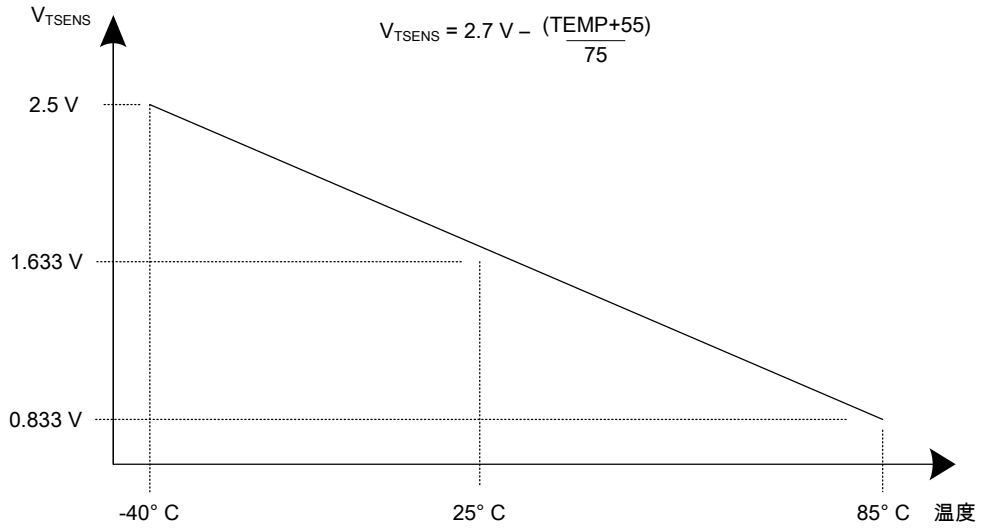
温度传感器没有单独的启用/禁用操作，因为它还关系到带隙参考电压的产生，必须始终启用。该参考电压不仅提供给 ADC 模块，还需要提供给其它所有模拟模块。此外，该温度传感器还带有 3.3 V 的辅助掉电输入功能，可以由休眠模块进行控制。

内部温度传感器将温度测量值转换为电压。此电压值 V_{TSENS} 可以通过以下公式得出（其中 TEMP 指温度，单位 °C）：

V_{TSENS} = 2.7 - ((TEMP + 55) / 75)

这种关系如图13-11（729页）所示。

图 **13-11.** 内部温度传感器特性



通过将 ADCSSCTLn 中的 TSn 位置位，即可在采样队列中得到温度感应器的读数。也可以从温度传感器的ADC结果通过函数转换得到温度读数。以下公式根据 ADC 读数（ADC_{CODE}，定义为 0 至 4095 的一个不带正负号的十进制数）和最大的 ADC 电压范围 (VREFP - VREFN) 计算温度（TEMP，单位 ℃）：

$TEMP = 147.5 - ((75 * (VREFP - VREFN) \times ADC_{CODE}) / 4096)$

13.3.7 数字比较器

ADC通常用于对外部信号采样并监控其数值的变动，确保其保持在给定的范围内。为了实现此监控过程的自动化、减少所需的处理器开销，ADC 模块内置有 每个模块提供 8 个数字比较器。

ADC 转换结果可直接发送给数字比较器，与用户编程的门限进行比较。门限通过 ADC 数字比较器范围寄存器（ADCDCCMPn）配置。ADC 可配置为根据 ADC 是在低值带、中值带还是高值带（可在 ADCDCCMPn 位域进行配置）运行而生成中断。另外可将数字比较器的 4 种工作模式（单次触发，持续触发，迟滞单次触发，迟滞持续触发）应用于中断配置。

13.3.7.1 输出功能

取决于 ADC 采样序列 n 操作 (ADCCSOPn) 寄存器中的 SnDCOP 位的设置，ADC 转换结果可以保存到 ADC 采样序列 FIFO 中，也可以供给数字比较器进行比较。选定的ADC转换结果将被其对应的数字比较器用于监控外部信号。每个数字比较器可以有两种输出功能：处理器中断或 PWM 触发事件。

每种输出功能都有其状态机对被监控的信号实施追踪。中断功能和触发事件功能既可以分别使能，也可以同时使能；两种功能将根据同一转换数据判断其条件是否已经满足、并据此产生相应的输出。

中断信号

将 ADC 数字比较器控制寄存器 (ADCDCCTLn) 的 CIE 位置位即可启用数字比较器的中断功能。此时中断功能状态机开始运行，并监控输入的 ADC 转换结果。当某组条件得到满足并且 ADCIM 寄存器的 DCONSSx 位置 1 时，将向中断控制器发送一个中断。

注意: 任何时刻只允许将 1 个 DCONSSn 位置位。如果置位的 DCONSSn 位过多，会导致 ADCRIS 寄存器的 INRDC 位被屏蔽，并且任何采样序列器的中断线都不再产生中断。如果使用中断信号，建议在交替采样或者采样序列结束时启用中断。

13.3.7.2 工作模式

数字比较器有4种工作模式，能够支持类型广泛的应用、满足各种信号的要求。这4种工作模式分别是：持续触发、单次触发、迟滞持续触发、迟滞单次触发。工作模式通过 ADCDCCTLn 寄存器的 CIM 域选取。

持续触发模式

在持续触发工作模式中，只要ADC转换值满足比较条件即会产生相应的中断或触发事件。因此，如果A/D转换结果处于规定的范围内，将产生一连串的中断或触发事件。

单次触发模式

在单次触发工作模式中，只有当前ADC转换值满足比较条件并且前一个ADC转换值不满足比较条件时，才会产生相应的中断或触发事件。因此，如果A/D转换结果处于规定的范围内，将产生单个中断或触发事件。

迟滞持续触发模式

迟滞持续触发工作模式只能结合低值带或高值带工作，只有跨越中值带进入相反的区域时才会清除迟滞条件。在迟滞持续触发工作模式中，满足以下条件时才会产生相应的中断或触发事件：ADC转换值满足其比较条件，或之前的某个ADC结果满足比较条件，并且迟滞条件尚未清除（ADC转换值尚未落入相反的区域）。因此，在ADC转换值进入相反的区域之前，将不断产生一连串的中断或触发事件。

迟滞单次触发模式

迟滞单次触发工作模式只能结合低值带或高值带工作，只有跨越中值带进入相反的区域时才会清除迟滞条件。在迟滞单次触发工作模式中，满足以下条件时才会产生相应的中断或触发事件：ADC 转换值满足其比较条件，且前一个 ADC 转换值不满足比较条件，且迟滞条件已清除。因此将产生单个中断或触发事件。

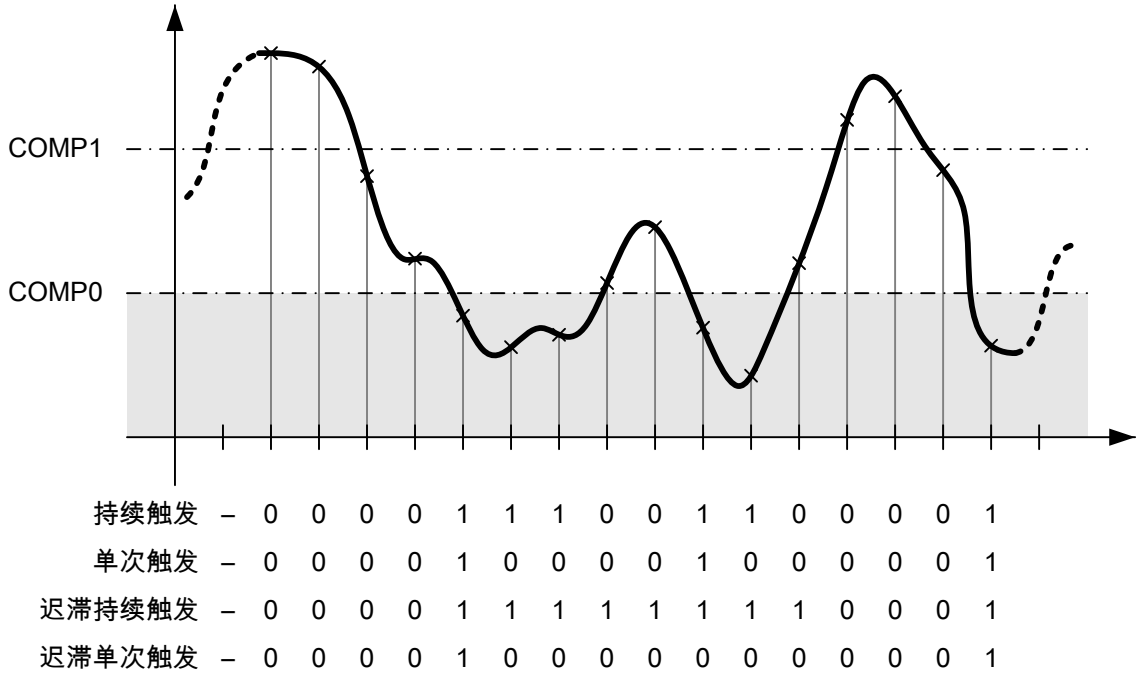
13.3.7.3 功能作用范围

ADC 数字比较器范围 (ADCDCCMPn) 寄存器中的两组比较门限 COMP0 和 COMP1 可将转换结果有效划分为 3 个不同区域。这些区域分别称为低值带（小于 COMP0）、中值带（大于 COMP0 但小于等于 COMP1）以及高值带（大于等于 COMP1）。允许将 COMP0 和 COMP1 编程为相等的值，也就是只划分出两个区域。请注意，COMP1 的值必须始终大于等于 COMP0。若 COMP1 小于 COMP0，其后果将难以预料。

低值带工作

要让数字比较器在低值带内工作，必须将 ADCDCCTLn 寄存器的 CIC 域设为 0x0。此设置会在低值带内按照编程的工作模式产生中断或触发事件。图13-12（731页）中显示了在高值带内各种工作模式下产生中断/触发事件的状态示例。注意，工作模式名称（持续触发、单次触发、迟滞持续触发、迟滞单次触发）之后一列中的“0”表示不产生中断或触发事件信号，“1”表示产生中断或触发事件信号。

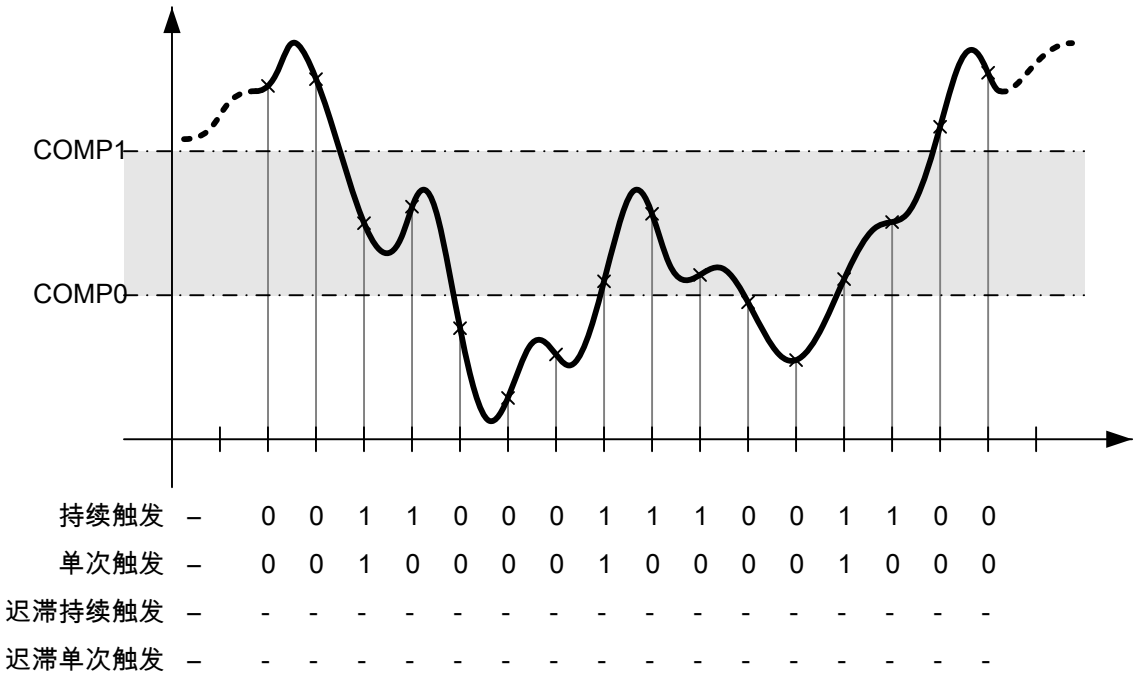
图 **13-12.** 低值带工作（**CIC = 0x0**）



中值带工作

要让数字比较器在中值带内工作，必须将 ADCDCCTLn 寄存器的 CIC 域设为 0x1。此设置会在中值带内按照工作模式产生中断或触发事件。只有持续触发工作模式和单次触发工作模式能够在中值带内工作。图13-13（732页）中显示了在中值带内各种允许的工作模式下产生中断/触发信号的状态示例。注意，工作模式名称（持续触发、单次触发）之后一列中的“0”表示不产生中断或触发事件信号，“1”表示产生中断或触发事件信号。

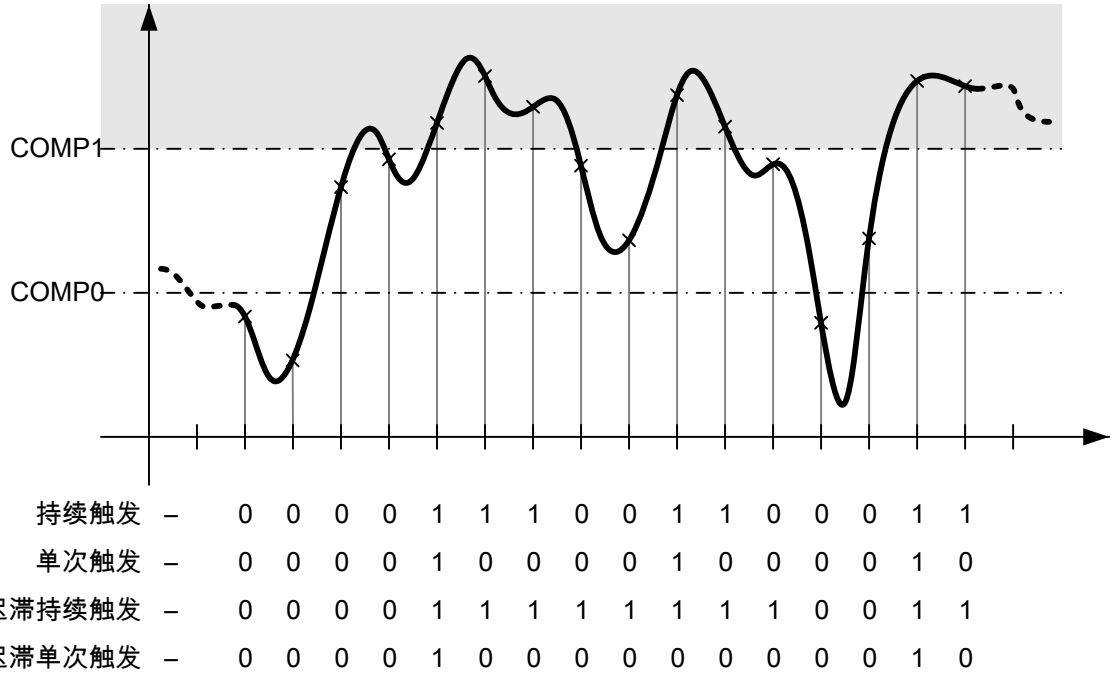
图 **13-13.** 中值带工作（**CIC = 0x1** ）



高值带工作

要让数字比较器在高值带内工作，必须将 ADCDCCTLn 寄存器的 CIC 域设为 0x3。此设置会在高值带内按照工作模式产生中断或触发事件。图13-14（733页）中显示了在高值带内各种允许的工作模式下产生中断/触发信号的状态示例。注意，工作模式名称（持续触发、单次触发、迟滞持续触发、迟滞单次触发）之后一列中的“0”表示不产生中断或触发事件信号，“1”表示产生中断或触发事件信号。

图 **13-14.** 高值带工作（**CIC = 0x3**）



13.4 初始化和配置

要想正常使用 ADC 模块，必须通过 RCC 寄存器（见第 221页）启用 PLL，并且将工作频率编程为 ADC 模块支持的数值。采用不支持的频率有可能造成ADC模块的工作发生错误。

13.4.1 模块初始化

ADC 模块的初始化流程比较简单，只有很少几个步骤：启用 ADC 的时钟、禁用待用模拟输入脚的模拟隔离电路、配置采样序列发生器优先级（如果有必要的话）。

ADC的初始化序列如下所示：

1. 使用 RCGCADC 寄存器启用 ADC 时钟（参见 306页）。
2. 通过 RCGCGPIO 寄存器（参见 295页）启用相应 GPIO 模块的时钟。欲了解需要启用哪些 GPIO 端口，请参阅“信号描述”（718页）。
3. 将 ADC 输入管脚的 AFSEL 位置位（参见 595页）。为了确定哪些 GPIO 需要配置，请参考 表 20-4（1019页）。
4. 通过将 GPIO 数字使能 (GPIODEN) 寄存器中相应 DEN 位清零，将 AINx 管脚配置为模拟输出（请参阅606页）。
5. 通过为 GPIOAMSEL 寄存器（参见 611页）的相应位写 1，禁用待用模拟输入脚的模拟隔离电路。
6. 假如应用有相关需求，则应通过 ADCSSPRI 寄存器重新配置采样序列发生器的优先级。默认配置是采样序列发生器0的优先级最高，采样序列发生器3的优先级最低。

13.4.2 采样序列发生器的配置

与模块的初始化流程相比，采样序列发生器的配置稍微复杂一些，这大概是因为每个采样序列发生器都是完全可编程的。

每个采样序列发生器的配置步骤应如下：

1. 将 ADCACTSS 寄存器的 ASENn 位清零，禁用采样序列发生器。采样序列器不用使能也可以进行配置。不过如果在配置期间禁用采样序列发生器，可以有效防止在此期间因满足触发条件而造成的误执行；
2. 在 ADCEMUX 寄存器中为采样序列发生器配置触发事件；
3. 在 ADCSSMUXn寄存器中为采样序列的每个采样配置相应的输入源。
4. 针对采样序列中的每个采样动作，对 ADCSSCTLn 寄存器中相应半字节的采样控制位进行配置。在配置最后一个半字节时，应确保 END 位置位。如果 END 不置位，将导致不可预测的执行结果。
5. 假如打算采用中断，则应将 ADCIM 寄存器中相应的 MASK 位置位。
6. 将 ADCACTSS 寄存器的 ASENn 位置位，启用采样序列发生器逻辑单元。

13.5 寄存器映射

表 13-4（734页）列出了所有 ADC 寄存器。表中偏移量一列是指相对于 ADC 模块基地址的寄存器地址十六进制增量，两个 ADC 模块的基地址分别为：

- ADC0：0x4003.8000
- ADC1：0x4003.9000

注意配置这些寄存器之前必须启用 ADC 模块的时钟（请参考 306页）。ADC 模块时钟启用后，必须等待至少 3 个系统时钟才可访问 ADC 模块寄存器。

表 **13-4. ADC** 寄存器映射

偏移量	名称	类型	复位	描述	见页面
0x000	ADCACTSS	R/W	0x0000.0000	ADC 有效采样序列发生器寄存器	737
0x004	ADCRIS	RO	0x0000.0000	ADC原始中断状态寄存器	739
0x008	ADCIM	R/W	0x0000.0000	ADC 中断掩码寄存器	741
0x00C	ADCISC	R/W1C	0x0000.0000	ADC 中断状态及清除寄存器	743
0x010	ADCOSTAT	R/W1C	0x0000.0000	ADC 上溢状态寄存器	746
0x014	ADCEMUX	R/W	0x0000.0000	ADC 事件复用选择寄存器	748
0x018	ADCUSTAT	R/W1C	0x0000.0000	ADC 下溢状态寄存器	753
0x020	ADCSSPRI	R/W	0x0000.3210	ADC 采样序列发生器优先级寄存器	754
0x024	ADCSPC	R/W	0x0000.0000	ADC 采样相位控制寄存器	755
0x028	ADCPSSI	R/W	-	ADC 处理器采样序列启动寄存器	756
0x030	ADCSAC	R/W	0x0000.0000	ADC 采样平均控制寄存器	758
0x034	ADCDCISC	R/W1C	0x0000.0000	ADC 数字比较器中断状态及清除寄存器	759

表 13-4. ADC 寄存器映射（续）

偏移量	名称	类型	复位	描述	见页面
0x038	ADCCTL	R/W	0x0000.0000	ADC 控制寄存器	761
0x040	ADCSSMUX0	R/W	0x0000.0000	ADC 采样序列输入复用选择寄存器 0	762
0x044	ADCSSCTL0	R/W	0x0000.0000	ADC 采样序列控制寄存器 0	763
0x048	ADCSSFIFO0	RO	-	ADC 采样序列结果 FIFO 寄存器 0	769
0x04C	ADCSSFSTAT0	RO	0x0000.0100	ADC 采样序列 FIFO 0 状态寄存器	770
0x050	ADCSSOP0	R/W	0x0000.0000	ADC 采样序列寄存器 0	772
0x054	ADCSSDC0	R/W	0x0000.0000	ADC 采样序列数字比较器选择寄存器 0	774
0x060	ADCSSMUX1	R/W	0x0000.0000	ADC 采样序列输入复用选择寄存器 1	776
0x064	ADCSSCTL1	R/W	0x0000.0000	ADC 采样序列控制寄存器 1	777
0x068	ADCSSFIFO1	RO	-	ADC 采样序列结果 FIFO 寄存器 1	769
0x06C	ADCSSFSTAT1	RO	0x0000.0100	ADC 采样序列 FIFO 1 状态寄存器	770
0x070	ADCSSOP1	R/W	0x0000.0000	ADC 采样序列 1 工作寄存器	781
0x074	ADCSSDC1	R/W	0x0000.0000	ADC 采样序列数字比较器选择寄存器 1	782
0x080	ADCSSMUX2	R/W	0x0000.0000	ADC 采样序列输入复用选择寄存器 2	776
0x084	ADCSSCTL2	R/W	0x0000.0000	ADC 采样序列控制寄存器 2	777
0x088	ADCSSFIFO2	RO	-	ADC 采样序列结果 FIFO 寄存器 2	769
0x08C	ADCSSFSTAT2	RO	0x0000.0100	ADC 采样序列 FIFO 2 状态寄存器	770
0x090	ADCSSOP2	R/W	0x0000.0000	ADC 采样序列 2 工作寄存器	781
0x094	ADCSSDC2	R/W	0x0000.0000	ADC 采样序列数字比较器选择寄存器 2	782
0x0A0	ADCSSMUX3	R/W	0x0000.0000	ADC 采样序列输入复用选择寄存器 3	783
0x0A4	ADCSSCTL3	R/W	0x0000.0000	ADC 采样序列控制寄存器 3	784
0x0A8	ADCSSFIFO3	RO	-	ADC 采样序列结果 FIFO 寄存器 3	769
0x0AC	ADCSSFSTAT3	RO	0x0000.0100	ADC 采样序列 FIFO 3 状态寄存器	770
0x0B0	ADCSSOP3	R/W	0x0000.0000	ADC 采样序列器 3 工作寄存器	786
0x0B4	ADCSSDC3	R/W	0x0000.0000	ADC 采样序列 3 数字比较器选择寄存器	787
0xD00	ADCDCRIC	WO	0x0000.0000	ADC 数字比较器复位启动条件寄存器	788
0xE00	ADCDCCTL0	R/W	0x0000.0000	ADC 数字比较器控制寄存器 0	792
0xE04	ADCDCCTL1	R/W	0x0000.0000	ADC 数字比较器控制寄存器 1	792
0xE08	ADCDCCTL2	R/W	0x0000.0000	ADC 数字比较器控制寄存器 2	792
0xE0C	ADCDCCTL3	R/W	0x0000.0000	ADC 数字比较器控制寄存器 3	792
0xE10	ADCDCCTL4	R/W	0x0000.0000	ADC 数字比较器控制寄存器 4	792
0xE14	ADCDCCTL5	R/W	0x0000.0000	ADC 数字比较器控制寄存器 5	792
0xE18	ADCDCCTL6	R/W	0x0000.0000	ADC 数字比较器控制寄存器 6	792

表 13-4. ADC 寄存器映射（续）

偏移量	名称	类型	复位	描述	见页面
0xE1C	ADCDCCTL7	R/W	0x0000.0000	ADC 数字比较器控制寄存器 7	792
0xE40	ADCDCCMP0	R/W	0x0000.0000	ADC 数字比较器范围寄存器 0	794
0xE44	ADCDCCMP1	R/W	0x0000.0000	ADC 数字比较器范围寄存器 1	794
0xE48	ADCDCCMP2	R/W	0x0000.0000	ADC 数字比较器范围寄存器 2	794
0xE4C	ADCDCCMP3	R/W	0x0000.0000	ADC 数字比较器范围寄存器 3	794
0xE50	ADCDCCMP4	R/W	0x0000.0000	ADC 数字比较器范围寄存器 4	794
0xE54	ADCDCCMP5	R/W	0x0000.0000	ADC 数字比较器范围寄存器 5	794
0xE58	ADCDCCMP6	R/W	0x0000.0000	ADC 数字比较器范围寄存器 6	794
0xE5C	ADCDCCMP7	R/W	0x0000.0000	ADC 数字比较器范围寄存器 7	794
0xFC0	ADCPP	RO	0x00B0.20C7	ADC 外设属性寄存器	795
0xFC4	ADCPC	R/W	0x0000.0007	ADC 外设配置寄存器	797
0xFC8	ADCCC	R/W	0x0000.0000	ADC 时钟配置寄存器	798

13.6 寄存器描述

下文将按照地址偏移量的数字顺序列出ADC寄存器，并对这些寄存器进行描述。

寄存器 1: ADC 有效采样序列发生器寄存器（ADCACTSS），偏移量 0x000

本寄存器控制采样序列发生器是否有效。每个采样序列发生器可分别设置启用或禁用。

ADC 有效采样序列发生器寄存器 (ADCACTSS)

ADC0 基址: 0x4003.8000
ADC1 基址: 0x4003.9000
偏移量 0x000
类型 R/W, 复位 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	保留															BUSY
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	保留												ASEN3	ASEN2	ASEN1	ASEN0
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位/域	名称	类型	复位	描述
31:17	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
16	BUSY	RO	0	ADC 繁忙 值 描述 0 ADC 空闲 1 ADC 繁忙
15:4	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
3	ASEN3	R/W	0	ADC SS3启用 值 描述 0 禁用采样序列器3。 1 启用采样序列器3。
2	ASEN2	R/W	0	ADC SS2 启用 值 描述 0 禁用采样序列器2。 1 启用采样序列器2。
1	ASEN1	R/W	0	ADC SS1 启用 值 描述 0 禁用采样序列器1。 1 启用采样序列器1。

位/域	名称	类型	复位	描述
0	ASEN0	R/W	0	ADC SS0 启用

值 描述
0 禁用采样序列器0。
1 启用采样序列器0。

寄存器 2: ADC原始中断状态寄存器（ADCRIS），偏移量 0x004

本寄存器用于指示每个采样序列发生器的原始中断状态。软件可以通过这些标志位查看那些并不向中断控制器发送的中断状态。

ADC原始中断状态寄存器 (ADCRIS)

ADC0 基址: 0x4003.8000
ADC1 基址: 0x4003.9000
偏移量 0x004
类型 RO, 复位 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	保留															INRDC
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	保留												INR3	INR2	INR1	INR0
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位/域	名称	类型	复位	描述
31:17	保留	RO	0x000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
16	INRDC	RO	0	数字比较器原始中断状态 值 描述 0 ADCDCISC 寄存器的所有位均清零。 1 ADCDCISC 寄存器中至少有 1 个位置位，即至少产生了 1 个数字比较器中断。
15:4	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
3	INR3	RO	0	SS3原始中断状态 值 描述 0 未产生中断 1 采样已完成了转换，且 ADCSSCTL3 寄存器中对应的 IEn 位已置位，将启用原始中断。 向 ADCISC 寄存器的 IN3 位写 1，可清除本标志位。
2	INR2	RO	0	SS2原始中断状态 值 描述 0 未产生中断 1 采样已完成了转换，且 ADCSSCTL2 寄存器中对应的 IEn 位已置位，将启用原始中断。 向 ADCISC 寄存器的 IN2 位写 1，可清除本标志位。

位/域	名称	类型	复位	描述
1	INR1	RO	0	SS1原始中断状态 值 描述 0 未产生中断 1 采样已完成了转换，且 ADCSSCTL1 寄存器中对应的 IEn 位已置位，将启用原始中断。 向 ADCISC 寄存器的 IN1 位写 1，可清除本标志位。
0	INR0	RO	0	SS0原始中断状态 值 描述 0 未产生中断 1 采样已完成了转换，且 ADCSSCTL0 寄存器中对应的 IEn 位已置位，将启用原始中断。 向 ADCISC 寄存器的 IN0 位写 1，可清除本标志位。

寄存器 3: ADC 中断掩码寄存器（ADCIM），偏移量 0x008

本寄存器控制采样序列发生器和数字比较器原始中断信号是否能发送给中断控制器。每个原始中断信号都能分别予以掩码。

注意: 任何时刻只允许将 1 个 DCONSSn 位置位。如果置位的 DCONSSn 位过多，会导致 ADCRIS 寄存器的 INRDC 位被屏蔽，并且任何采样序列器的中断线都不再产生中断。如果使用中断信号，建议在交替采样或者采样序列结束时启用中断。

ADC 中断掩码寄存器 (ADCIM)

ADC0 基址: 0x4003.8000
ADC1 基址: 0x4003.9000
偏移量 0x008
类型 R/W, 复位 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	保留												DCONSS3	DCONSS2	DCONSS1	DCONSS0
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	保留												MASK3	MASK2	MASK1	MASK0
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位/域	名称	类型	复位	描述
31:20	保留	RO	0x000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
19	DCONSS3	R/W	0	SS3 数字比较器中断 值 描述 0 数字比较器的状态不影响 SS3 中断状态。 1 数字比较器产生的原始中断信号（ADCRIS 寄存器的 INRDC 位）将从 SS3 中断线发送给中断控制器。
18	DCONSS2	R/W	0	SS2数字比较器中断 值 描述 0 数字比较器的状态不影响 SS2 中断状态。 1 数字比较器产生的原始中断信号（ADCRIS 寄存器的 INRDC 位）将从 SS2 中断线发送给中断控制器。
17	DCONSS1	R/W	0	SS1数字比较器中断 值 描述 0 数字比较器的状态不影响 SS1 中断状态。 1 数字比较器产生的原始中断信号（ADCRIS 寄存器的 INRDC 位）将从 SS1 中断线发送给中断控制器。

位/域	名称	类型	复位	描述
16	DCONSS0	R/W	0	SS0数字比较器中断 值 描述 0 数字比较器的状态不影响 SS0 中断状态。 1 数字比较器产生的原始中断信号（ADCRIS 寄存器的 INRDC 位）将从 SS0 中断线发送给中断控制器。
15:4	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
3	MASK3	R/W	0	SS3中断掩码 值 描述 0 采样序列器3的状态不影响SS3中断状态 1 采样序列器 3 的原始中断信号（ADCRIS 寄存器的 INR3 位）将发送给中断控制器。
2	MASK2	R/W	0	SS2中断掩码 值 描述 0 采样序列器2的状态不影响SS2中断状态 1 采样序列器 2 的原始中断信号（ADCRIS 寄存器的 INR2 位）将发送给中断控制器。
1	MASK1	R/W	0	SS1中断掩码 值 描述 0 采样序列器1的状态不影响SS1中断状态 1 采样序列器 1 的原始中断信号（ADCRIS 寄存器的 INR1 位）将发送给中断控制器。
0	MASK0	R/W	0	SS0中断掩码 值 描述 0 采样序列器0的状态不影响SS0中断状态。 1 采样序列器 0 的原始中断信号（ADCRIS 寄存器的 INR0 位）将发送给中断控制器。

寄存器 4: ADC 中断状态及清除寄存器（ADCISC），偏移量 0x00C

本寄存器能够显示出由采样序列发生器以及数字比较器以及数字比较器产生、已经发送给中断控制器的中断状态，并提供清除采样序列发生器中断状态的机制。读本寄存器时，每个位域实际上是相应的 INR 位和 MASK 位的逻辑与结果。向采样序列器中断位写1，可清除相应的中断。请注意，数字比较器中断是通过向 ADCDCISC 寄存器的对应位写 1 来清除的。假如以软件轮询 ADCRIS 寄存器的方式处理事件（而不是产生中断），那么即使 INn 位并未置 1 也能通过 ADCISC 寄存器来清除 INRn 位。

ADC 中断状态及清除寄存器 (ADCISC)

ADC0 基址: 0x4003.8000
ADC1 基址: 0x4003.9000
偏移量 0x00C
类型 R/W1C, 复位 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	保留												DCINSS3	DCINSS2	DCINSS1	DCINSS0
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	保留												IN3	IN2	IN1	IN0
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W1C	R/W1C	R/W1C	R/W1C
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位/域	名称	类型	复位	描述
31:20	保留	RO	0x000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
19	DCINSS3	RO	0	SS3 数字比较器的中断状态 值 描述 0 未发生中断，或中断被屏蔽。 1 如果 ADCRIS 寄存器的 INRDC 位和 ADCIM 寄存器的 DCONSS3 位都置位，就会向中断控制器产生一个基于电平的中断。 此位写 1 清 0。如果将此位清零，则 ADCRIS 寄存器中的 INRDC 位也会被清零。
18	DCINSS2	RO	0	SS2 数字比较器的中断状态 值 描述 0 未发生中断，或中断被屏蔽。 1 如果 ADCRIS 寄存器的 INRDC 位和 ADCIM 寄存器的 DCONSS2 位都置位，就会向中断控制器产生一个基于电平的中断。 此位写 1 清 0。如果将此位清零，则 ADCRIS 寄存器中的 INRDC 位也会被清零。

位/域	名称	类型	复位	描述
17	DCINSS1	RO	0	SS1数字比较器中断状态 值 描述 0 未发生中断，或中断被屏蔽。 1 如果 ADCRIS 寄存器的 INRDC 位和 ADCIM 寄存器的 DCONSS1 位都置位，就会向中断控制器产生一个基于电平的中断。 此位写 1 清 0。如果将此位清零，则 ADCRIS 寄存器中的 INRDC 位也会被清零。
16	DCINSS0	RO	0	SS0数字比较器中断状态 值 描述 0 未发生中断，或中断被屏蔽。 1 如果 ADCRIS 寄存器的 INRDC 位和 ADCIM 寄存器的 DCONSS0 位都置位，就会向中断控制器产生一个基于电平的中断。 此位写 1 清 0。如果将此位清零，则 ADCRIS 寄存器中的 INRDC 位也会被清零。
15:4	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
3	IN3	R/W1C	0	SS3中断状态及清除 值 描述 0 未发生中断，或中断被屏蔽。 1 ADCRIS 寄存器的 INR3 位以及 ADCIM 寄存器的 MASK3 位都已置位，于是向中断控制器产生基于电平的中断。 对此标志位写 1，即可将其清零。如果将此位清零，则寄存器 ADCRIS 中的 INR3 位也会被清零。
2	IN2	R/W1C	0	SS2中断状态及清除 值 描述 0 未发生中断，或中断被屏蔽。 1 ADCRIS 寄存器的 INR2 位以及 ADCIM 寄存器的 MASK2 位都已置位，于是向中断控制器产生基于电平的中断。 对此标志位写 1，即可将其清零。如果将此位清零，则寄存器 ADCRIS 中的 INR2 位也会被清零。
1	IN1	R/W1C	0	SS1中断状态及清除 值 描述 0 未发生中断，或中断被屏蔽。 1 ADCRIS 寄存器的 INR1 位以及 ADCIM 寄存器的 MASK1 位都已置位，于是向中断控制器产生基于电平的中断。 对此标志位写 1，即可将其清零。如果将此位清零，则寄存器 ADCRIS 中的 INR1 位也会被清零。

位/域	名称	类型	复位	描述
0	IN0	R/W1C	0	SS0中断状态及清除

值 描述

0 未发生中断，或中断被屏蔽。

1 ADCRIS 寄存器的 INR0 位以及 ADCIM 寄存器的 MASK0 位都已置位，于是向中断控制器产生基于电平的中断。

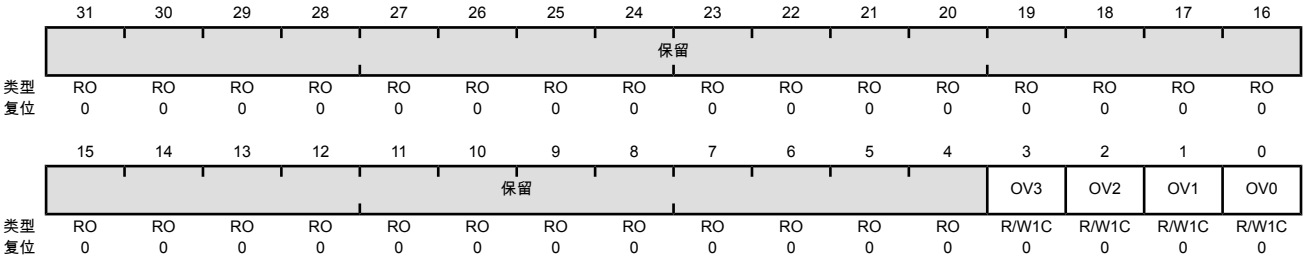
对此标志位写 1，即可将其清零。如果将此位清零，则寄存器 ADCRIS 中的 INR0 位也会被清零。

寄存器 5: ADC 上溢状态寄存器（ADCOSTAT），偏移量 0x010

本寄存器用于指示采样序列发生器FIFO的上溢状况。当软件处理完FIFO上溢状况后，可向相应的位写1予以清除。

ADC 上溢状态寄存器 (ADCOSTAT)

ADC0 基址: 0x4003.8000
ADC1 基址: 0x4003.9000
偏移量 0x010
类型 R/W1C, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:4	保留	RO	0x0000.000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
3	OV3	R/W1C	0	SS3 FIFO上溢 值 描述 0 FIFO 未出现上溢。 1 采样序列发生器 3 的 FIFO 达到了上溢条件，也就是说 FIFO 满且要求进行写操作。当FIFO出现上溢后，最近一次的写操作会丢失。 对此标志位写 1，即可将其清零。
2	OV2	R/W1C	0	SS2 FIFO上溢 值 描述 0 FIFO 未出现上溢。 1 采样序列发生器 2 的 FIFO 达到了上溢条件，也就是说 FIFO 满且要求进行写操作。当FIFO出现上溢后，最近一次的写操作会丢失。 对此标志位写 1，即可将其清零。
1	OV1	R/W1C	0	SS1 FIFO上溢 值 描述 0 FIFO 未出现上溢。 1 采样序列发生器 1 的 FIFO 达到了上溢条件，也就是说 FIFO 满且要求进行写操作。当FIFO出现上溢后，最近一次的写操作会丢失。 对此标志位写 1，即可将其清零。

位/域	名称	类型	复位	描述
0	OV0	R/W1C	0	SS0 FIFO上溢

值 描述

0 FIFO 未出现上溢。

1 采样序列发生器 0 的 FIFO 达到了上溢条件，也就是说 FIFO 满且要求进行写操作。当FIFO出现上溢后，最近一次的写操作会丢失。

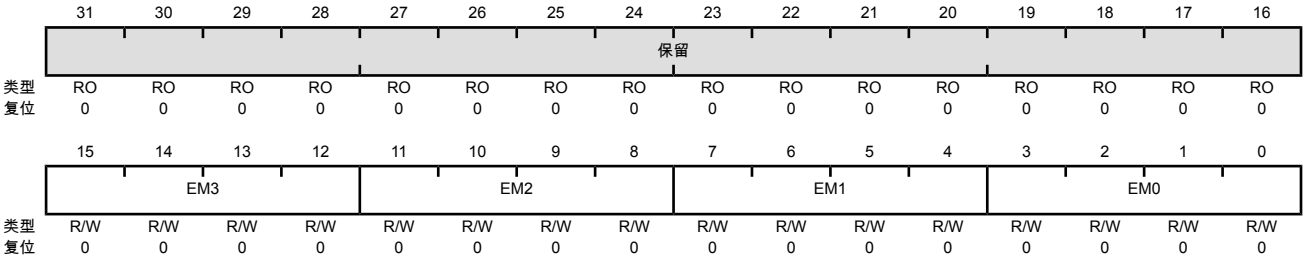
对此标志位写 1，即可将其清零。

寄存器 6: ADC 事件复用选择寄存器（ADCEMUX），偏移量 0x014

ADCEMUX 寄存器用于选择每个采样序列发生器启动采样的事件（触发条件）。每个采样序列器可配置为使用唯一的触发源。

ADC 事件复用选择寄存器 (ADCEMUX)

ADC0 基址: 0x4003.8000
ADC1 基址: 0x4003.9000
偏移量 0x014
类型 R/W, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:16	保留	RO	0x0000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。

位/域	名称	类型	复位	描述
15:12	EM3	R/W	0x0	SS3触发条件选择 该字段选择采样序列发生器3的触发源。 此位域的有效配置包括： 值 事件 0x0 处理器（默认） 将 ADCPSSI 寄存器中的 SSn 位置位，启动该触发器。 0x1 模拟比较器0 该触发器由模拟比较器控制 0 (ACCTL0) 寄存器配置（参考1004页）。 0x2 模拟比较器1 该触发器由模拟比较器控制 1 (ACCTL1) 寄存器配置（参考1004页）。 0x3 保留 0x4 外部（GPIO 管脚） 该触发器连接到相应 GPIO 的 GPIO 中断上（请参阅“ADC触发源”（581页））。 注意: 如果 GPIO 带有作为备用功能的 AINx 信号，则可用于触发 ADC。同样，该管脚也不能同时用作 GPIO 和模拟输入端。 0x5 定时器 此外，还必须通过 GPTMCTL 寄存器的 TnOTE 位来启用该触发器（参考 657页）。 0x6 保留 0x7 保留 0x8 保留 0x9 保留 0xA-0xE 保留 0xF 持续（始终采样）

位/域	名称	类型	复位	描述
11:8	EM2	R/W	0x0	SS2 触发器选择 该字段选择采样序列发生器2的触发源。 此位域的有效配置包括： 值 事件 0x0 处理器（默认） 将 ADCPSSI 寄存器中的 SSn 位置位，启动该触发器。 0x1 模拟比较器0 该触发器由模拟比较器控制 0 (ACCTL0) 寄存器配置（参考1004页）。 0x2 模拟比较器1 该触发器由模拟比较器控制 1 (ACCTL1) 寄存器配置（参考1004页）。 0x3 保留 0x4 外部（GPIO 管脚） 该触发器连接到相应 GPIO 的 GPIO 中断上（请参阅“ADC 触发源”（581页））。 注意: 如果 GPIO 带有作为备用功能的 AINx 信号，则可用于触发 ADC。同样，该管脚也不能同时用作 GPIO 和模拟输入端。 0x5 定时器 此外，还必须通过 GPTMCTL 寄存器的 TnOTE 位来启用该触发器（参考 657页）。 0x6 保留 0x7 保留 0x8 保留 0x9 保留 0xA-0xE 保留 0xF 持续（始终采样）

位/域	名称	类型	复位	描述
7:4	EM1	R/W	0x0	SS1 触发器选择 该字段选择采样序列发生器1的触发源。 此位域的有效配置包括： 值 事件 0x0 处理器（默认） 将 ADCPSSI 寄存器中的 SSn 位置位，启动该触发器。 0x1 模拟比较器0 该触发器由模拟比较器控制 0 (ACCTL0) 寄存器配置（参考 1004页）。 0x2 模拟比较器1 该触发器由模拟比较器控制 1 (ACCTL1) 寄存器配置（参考 1004页）。 0x3 保留 0x4 外部（GPIO 管脚） 该触发器连接到相应 GPIO 的 GPIO 中断上（请参阅“ADC 触发源”（581页））。 注意: 如果 GPIO 带有作为备用功能的 AINx 信号，则可用于触发 ADC。同样，该管脚也不能同时用作 GPIO 和模拟输入端。 0x5 定时器 此外，还必须通过 GPTMCTL 寄存器的 TnOTE 位来启用该触发器（参考 657页）。 0x6 保留 0x7 保留 0x8 保留 0x9 保留 0xA-0xE 保留 0xF 持续（始终采样）

位/域	名称	类型	复位	描述
3:0	EM0	R/W	0x0	SS0 触发器选择 该字段选择采样序列发生器0的触发源 此位域的有效配置包括：

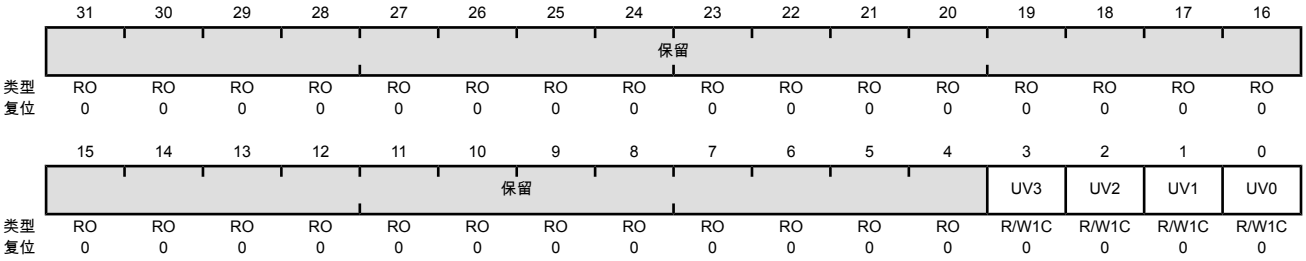
值	事件
0x0	处理器（默认） 将 ADCPSSI 寄存器中的 SSn 位置位，启动该触发器。
0x1	模拟比较器0 该触发器由模拟比较器控制 0 (ACCTL0) 寄存器配置（参考1004页）。
0x2	模拟比较器1 该触发器由模拟比较器控制 1 (ACCTL1) 寄存器配置（参考1004页）。
0x3	保留
0x4	外部（GPIO 管脚） 该触发器连接到相应 GPIO 的 GPIO 中断上（请参阅“ADC 触发源”（581页））。 注意: 如果 GPIO 带有作为备用功能的 AINx 信号，则可用于触发 ADC。同样，该管脚也不能同时用作 GPIO 和模拟输入端。
0x5	定时器 此外，还必须通过 GPTMCTL 寄存器的 TnOTE 位来启用该触发器（参考 657页）。
0x6	保留
0x7	保留
0x8	保留
0x9	保留
0xA-0xE	保留
0xF	持续（始终采样）

寄存器 7: ADC 下溢状态寄存器（ADCUSTAT），偏移量 0x018

本寄存器用于指示采样序列发生器FIFO的下溢状况。当软件处理完FIFO下溢状况后，可向相应的位写1予以清除。

ADC 下溢状态寄存器 (ADCUSTAT)

ADC0 基址: 0x4003.8000
ADC1 基址: 0x4003.9000
偏移量 0x018
类型 R/W1C, 复位 0x0000.0000



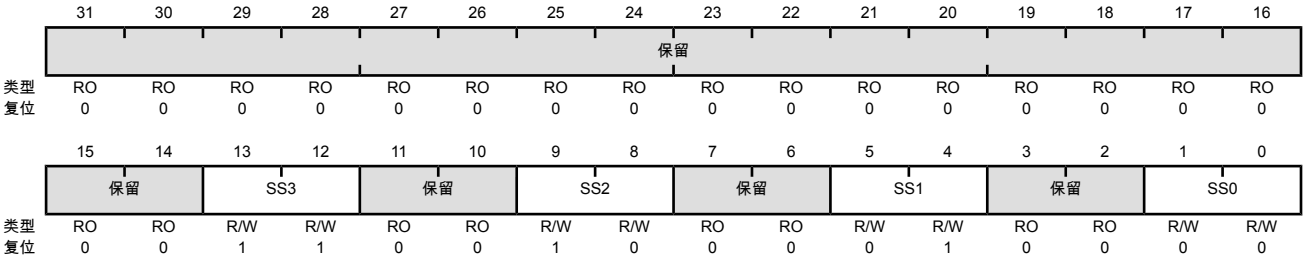
位/域	名称	类型	复位	描述
31:4	保留	RO	0x0000.000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
3	UV3	R/W1C	0	SS3 FIFO下溢 此位域的有效配置如下所示。对此标志位写 1，即可将其清零。 值 描述 0 FIFO未出现下溢。 1 采样序列发生器的 FIFO 达到了下溢条件，也就是说 FIFO 空且要求进行读操作。此次读操作并不会移动 FIFO 指针，返回值为 0。
2	UV2	R/W1C	0	SS2 FIFO下溢 此域的有效配置与对 UV3 域的配置相同。对此标志位写 1，即可将其清零。
1	UV1	R/W1C	0	SS1 FIFO 下溢 此域的有效配置与对 UV3 域的配置相同。对此标志位写 1，即可将其清零。
0	UV0	R/W1C	0	SS0 FIFO 下溢 此域的有效配置与对 UV3 域的配置相同。对此标志位写 1，即可将其清零。

寄存器 8: ADC 采样序列发生器优先级寄存器（ADCSSPRI），偏移量 0x020

本寄存器用于设置各采样序列发生器的优先级。复位后默认情况下采样序列发生器0具有最高优先级、采样序列发生器3具有最低优先级。修改优先级时务必确保每个采样序列发生器具有唯一的优先级，否则ADC可能无法正常工作。

ADC 采样序列发生器优先级寄存器 (ADCSSPRI)

ADC0 基址: 0x4003.8000
ADC1 基址: 0x4003.9000
偏移量 0x020
类型 R/W, 复位 0x0000.3210



位/域	名称	类型	复位	描述
31:14	保留	RO	0x0000.0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
13:12	SS3	R/W	0x3	SS3优先级 此字段包含确定采样序列发生器 3 的优先级编码的二进制编码值。优先级编码 0x0 的优先级最高；0x3 最低。分配给序列发生器的优先级必须被唯一地映射。如果两个或两个以上的域相等，那么 ADC 就不能正常工作。
11:10	保留	RO	0x0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
9:8	SS2	R/W	0x2	SS2 优先级 此字段包含确定采样序列发生器 2 的优先级编码的二进制编码值。优先级编码 0x0 的优先级最高；0x3 最低。分配给序列发生器的优先级必须被唯一地映射。如果两个或两个以上的域相等，那么 ADC 就不能正常工作。
7:6	保留	RO	0x0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
5:4	SS1	R/W	0x1	SS1 优先级 此字段包含确定采样序列发生器 1 的优先级编码的二进制编码值。优先级编码 0x0 的优先级最高；0x3 最低。分配给序列发生器的优先级必须被唯一地映射。如果两个或两个以上的域相等，那么 ADC 就不能正常工作。
3:2	保留	RO	0x0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
1:0	SS0	R/W	0x0	SS0 优先级 此字段包含确定采样序列发生器 0 的优先级编码的二进制编码值。优先级编码 0x0 的优先级最高；0x3 最低。分配给序列发生器的优先级必须被唯一地映射。如果两个或两个以上的域相等，那么 ADC 就不能正常工作。

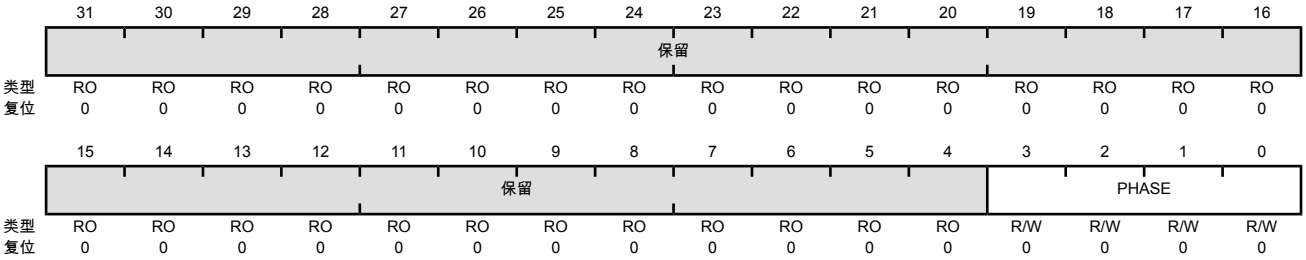
寄存器 9: ADC 采样相位控制寄存器（ADCSPC），偏移量 0x024

本寄存器可设置采样点的相位，从0.0°到337.5°可选。例如，借助于芯片内的两路ADC模块可实现双倍采样率，其方法便是将两路ADC模块配置为对同一模拟输入端采样，并将一路ADC模块采样相位配置为0°、另一路ADC模块采样相位配置为180°。

注意: 当 PHASE 域非 0 时应当小心，因为对 AINx 输入信号的采样延迟可能导致采样结果有别于系统设计的预期。从 ADC 触发到采样的时间会增加，可能会使得响应时间要比预期要长。因此而增加的延迟可能会对系统的设计带来负面影响。设计人员必须慎重考虑此延时的影响。

ADC 采样相位控制寄存器 (ADCSPC)

ADC0 基址: 0x4003.8000
ADC1 基址: 0x4003.9000
偏移量 0x024
类型 R/W, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:4	保留	RO	0x0000.000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
3:0	PHASE	R/W	0x0	相位差 此位域设置采样点与标准采样时间的相位差。

值	描述
0x0	ADC采样滞后0°
0x1	ADC 采样滞后 22.5°
0x2	ADC 采样滞后 45.0°
0x3	ADC 采样滞后 67.5°
0x4	ADC 采样滞后 90.0°
0x5	ADC 采样滞后 112.5°
0x6	ADC 采样滞后 135.0°
0x7	ADC 采样滞后 157.5°
0x8	ADC 采样滞后 180.0°
0x9	ADC 采样滞后 202.5°
0xA	ADC 采样滞后 225.0°
0xB	ADC 采样滞后 247.5°
0xC	ADC 采样滞后 270.0°
0xD	ADC 采样滞后 292.5°
0xE	ADC 采样滞后 315.0°
0xF	ADC 采样滞后 337.5°

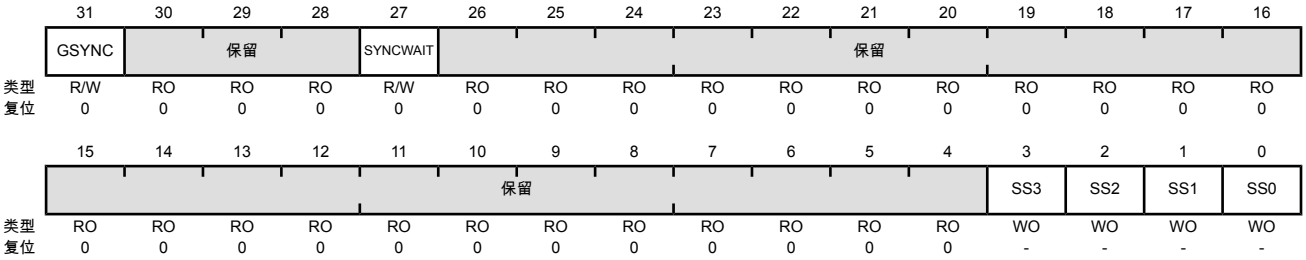
寄存器 10: ADC 处理器采样序列启动寄存器（ADCPSSI），偏移量 0x028

本寄存器提供了由软件启动采样序列工作的手段。采样序列可以各自分别启动，也可以按照任意组合同时启动。当同时触发多个采样序列时，其执行顺序将由 ADCSSPRI 寄存器的优先级编码决定。

本寄存器还能配置并同时启动对所有 ADC 模块的采样。方法是：首先配置第一个 ADC 模块。然后对该模块的 ADCPSSI 寄存器进行写操作。将相应的 SS 位与 SYNCWAIT 位一同置位。其余ADC 模块同样按此流程逐个进行配置。当配置完最后一个 ADC 模块后，其 ADCPSSI 寄存器同样应写入，写操作时应将相应的 SS 位与 GSYNC 位一同置 1。随后所有ADC模块将同时开始按照其配置执行采样。

ADC 处理器采样序列启动寄存器 (ADCPSSI)

ADC0 基址: 0x4003.8000
ADC1 基址: 0x4003.9000
偏移量 0x028
类型 R/W, 复位 -



位/域	名称	类型	复位	描述
31	GSYNC	R/W	0	全局同步 值 描述 0 一旦开始采样则此标志位清零 1 此位用于同时启动多个 ADC 模块的采样。只要有 ADC 模块因 SSn 和 SYNCWAIT 位置位而得以启用，则对本标志位进行写操作时，这些模块将开始采样。
30:28	保留	RO	0x0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
27	SYNCWAIT	R/W	0	同步等待 值 描述 0 启动某个采样序列时，将立即执行。 1 通过该位可允许创建采样序列，但会延迟采样，直到 GSYNC 位置位。
26:4	保留	RO	0x0000.0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。

位/域	名称	类型	复位	描述
3	SS3	WO	-	SS3启动 值 描述 0 没有影响 1 如果采样序列发生器 3 已在 ADCACTSS 寄存器中启用，则将执行采样。 此标志位只能进行写操作；读操作的返回值无意义。
2	SS2	WO	-	SS2 启动 值 描述 0 没有影响 1 如果采样序列发生器 2 已在 ADCACTSS 寄存器中启用，则将执行采样。 此标志位只能进行写操作；读操作的返回值无意义。
1	SS1	WO	-	SS1 启动 值 描述 0 没有影响 1 如果采样序列发生器 1 已在 ADCACTSS 寄存器中启用，则将执行采样。 此标志位只能进行写操作；读操作的返回值无意义。
0	SS0	WO	-	SS0 启动 值 描述 0 没有影响 1 如果采样序列发生器 0 已在 ADCACTSS 寄存器中启用，则将执行采样。 此标志位只能进行写操作；读操作的返回值无意义。

寄存器 11: ADC 采样平均控制寄存器（ADCSAC），偏移量 0x030

本寄存器控制硬件对转换结果进行取平均值时的次数。存储在 FIFO 中的最终转换结果是以指定的 ADC 速率通过 2 ^{AVG} 连续的 ADC 采样进行平均计算而得到的。若 AVG 为 0，则每次采样将直接写入 FIFO，不进行任何平均运算。若 AVG = 6，那么 ADC 模块将连续采集 64 个采样后求其算术平均值，并将计算结果存入 FIFO 中。若 AVG = 7，其结果无法预料。

ADC 采样平均控制寄存器 (ADCSAC)

ADC0 基址: 0x4003.8000
ADC1 基址: 0x4003.9000
偏移量 0x030
类型 R/W, 复位 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	保留															
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	保留													AVG		
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

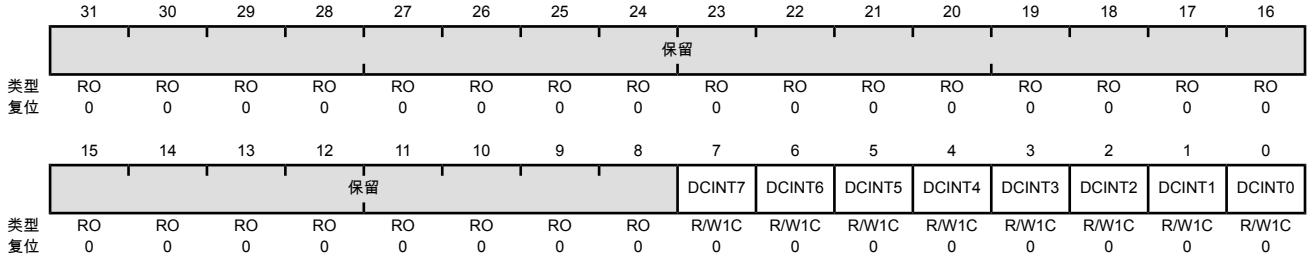
位/域	名称	类型	复位	描述
31:3	保留	RO	0x0000.000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
2:0	AVG	R/W	0x0	硬件平均控制 确定硬件平均计算的 ADC 采样数。该 AVG 域可以是 0 到 6 之间的任意值。如果 AVG = 7，其结果无法预测。 值 描述 0x0 无硬件过采样 0x1 2x 硬件过采样 0x2 4x 硬件过采样 0x3 8x 硬件过采样 0x4 16x 硬件过采样 0x5 32x 硬件过采样 0x6 64x 硬件过采样 0x7 保留

寄存器 12: ADC 数字比较器中断状态及清除寄存器（ADCDCISC），偏移量 0x034

本寄存器可指示数字比较器中断状态，并能对其确认操作。每个数字比较器占用1个位。

ADC 数字比较器中断状态及清除寄存器 (ADCDCISC)

ADC0 基址: 0x4003.8000
ADC1 基址: 0x4003.9000
偏移量 0x034
类型 R/W1C, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:8	保留	RO	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
7	DCINT7	R/W1C	0	数字比较器7中断状态及清除 值 描述 0 无中断。 1 数字比较器7产生中断 对此标志位写 1，即可将其清零。
6	DCINT6	R/W1C	0	数字比较器6中断状态及清除 值 描述 0 无中断。 1 数字比较器6产生中断 对此标志位写 1，即可将其清零。
5	DCINT5	R/W1C	0	数字比较器5中断状态及清除 值 描述 0 无中断。 1 数字比较器5产生中断 对此标志位写 1，即可将其清零。
4	DCINT4	R/W1C	0	数字比较器4中断状态及清除 值 描述 0 无中断。 1 数字比较器4产生中断 对此标志位写 1，即可将其清零。

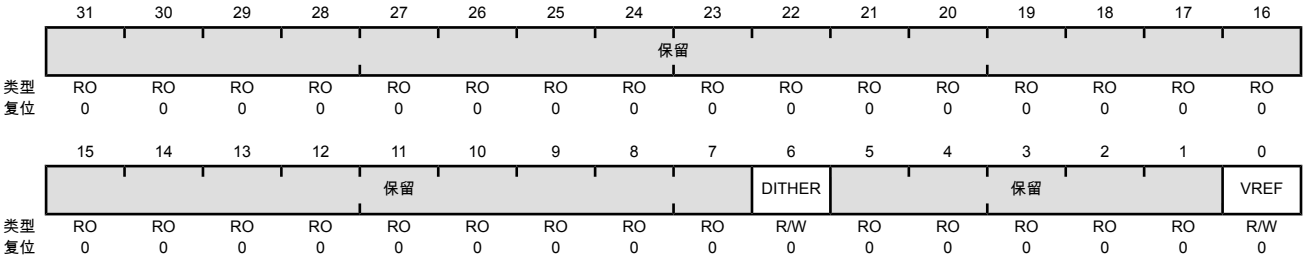
位/域	名称	类型	复位	描述
3	DCINT3	R/W1C	0	数字比较器3中断状态及清除 值 描述 0 无中断。 1 数字比较器3产生中断 对此标志位写 1，即可将其清零。
2	DCINT2	R/W1C	0	数字比较器2中断状态及清除 值 描述 0 无中断。 1 数字比较器2产生中断 对此标志位写 1，即可将其清零。
1	DCINT1	R/W1C	0	数字比较器1中断状态及清除 值 描述 0 无中断。 1 数字比较器1产生中断 对此标志位写 1，即可将其清零。
0	DCINT0	R/W1C	0	数字比较器0中断状态及清除 值 描述 0 无中断。 1 数字比较器0产生中断 对此标志位写 1，即可将其清零。

寄存器 13: ADC 控制寄存器（ADCCTL），偏移量 0x038

本寄存器用于配置参考电压。注意，此寄存器中设置的值会应用到所有ADC模块，不可能设置为一个模块使用内部参考，而另一个模块使用外部参考。

ADC 控制寄存器 (ADCCTL)

ADC0 基址: 0x4003.8000
ADC1 基址: 0x4003.9000
偏移量 0x038
类型 R/W, 复位 0x0000.0000



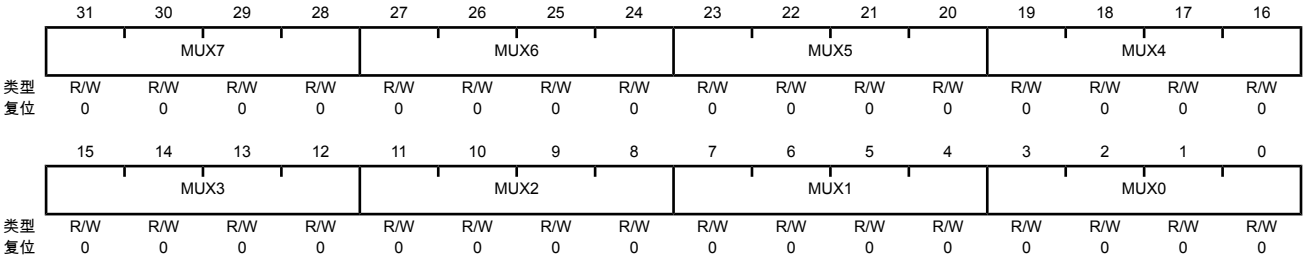
位/域	名称	类型	复位	描述
31:7	保留	RO	0x0000.000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
6	DITHER	R/W	0	启用抖动模式 值 描述 0 禁用抖动模式 1 抖动模式已启用
5:1	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
0	VREF	R/W	0x0	参考电压源选择 值 描述 0x0 VDDA 和 GNDA 是所有 ADC 模块的电压参考。 0x1 保留

寄存器 14: ADC 采样序列输入复用选择寄存器 0（ADCSSMUX0），偏移量 0x040

该寄存器为采样序列发生器0所执行序列的每个采样定义模拟输入配置。本寄存器为 32 位宽，可分别定义 8 个采样的信息。

ADC 采样序列输入复用选择寄存器 0 (ADCSSMUX0)

ADC0 基址: 0x4003.8000
ADC1 基址: 0x4003.9000
偏移量 0x040
类型 R/W, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:28	MUX7	R/W	0x0	第8个采样动作的输入选择 MUX7 字段将在采样序列发生器所执行序列的第 8 个采样中使用。它确定在进行模数转换时采样哪个模拟输入。此处设置的值表示相应的管脚。例如，当 EMUX7 清零时，0x1 表示输入管脚是 AIN1。
27:24	MUX6	R/W	0x0	第7个采样动作的输入选择 MUX6 字段将在采样序列发生器所执行序列的第 7 个采样中使用。它确定在进行模数转换时采样哪个模拟输入。
23:20	MUX5	R/W	0x0	第 6 个采样的输入选择 MUX5 字段将在采样序列发生器所执行序列的第 6 个采样中使用。它确定在进行模数转换时采样哪个模拟输入。
19:16	MUX4	R/W	0x0	第 5 个采样的输入选择 MUX4 字段将在采样序列发生器所执行序列的第 5 个采样中使用。它确定在进行模数转换时采样哪个模拟输入。
15:12	MUX3	R/W	0x0	第 4 个采样的输入选择 MUX3 字段将在采样序列发生器所执行序列的第 4 个采样中使用。它确定在进行模数转换时采样哪个模拟输入。
11:8	MUX2	R/W	0x0	第 3 个采样的输入选择 MUX2 字段将在采样序列发生器所执行序列的第 3 个采样中使用。它确定在进行模数转换时采样哪个模拟输入。
7:4	MUX1	R/W	0x0	第 2 个采样的输入选择 MUX1 字段将在采样序列发生器所执行序列的第 2 个采样中使用。它确定在进行模数转换时采样哪个模拟输入。
3:0	MUX0	R/W	0x0	第 1 个采样的输入选择 MUX0 字段将在采样序列发生器所执行序列的第 1 个采样中使用。它确定在进行模数转换时采样哪个模拟输入。

寄存器 15: ADC 采样序列控制寄存器 0（ADCSSCTL0），偏移量 0x044

本寄存器包含采样序列中每个采样动作的配置信息。在配置一个采样序列时，不管该序列中包含几个有效的采样动作，其最后 1 个有效采样动作的 END 位必须置 1。本寄存器为 32 位宽，可分别定义 8 个采样的信息。

ADC 采样序列控制寄存器 0 (ADCSSCTL0)

ADC0 基址: 0x4003.8000
ADC1 基址: 0x4003.9000
偏移量 0x044
类型 R/W, 复位 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	TS7	IE7	END7	D7	TS6	IE6	END6	D6	TS5	IE5	END5	D5	TS4	IE4	END4	D4
类型	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	TS3	IE3	END3	D3	TS2	IE2	END2	D2	TS1	IE1	END1	D1	TS0	IE0	END0	D0
类型	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位/域	名称	类型	复位	描述
31	TS7	R/W	0	第8个采样动作的温度传感器选择 值 描述 0 在执行采样序列的第 8 个采样动作时读取 ADCSSMUXn 寄存器指定的输入管脚。 1 采样序列的第 8 个采样动作读取温度传感器的内容
30	IE7	R/W	0	第 8 个采样动作的中断使能 值 描述 0 不产生原始中断 1 在第 8 个采样转换结束后，原始中断信号（INR0 位）生效。若 ADCIM 寄存器的 MASK0 位置位，就会向中断控制器提交中断。 允许同一序列中多个采样动作各自产生中断。
29	END7	R/W	0	第 8 个采样动作后序列终止 值 描述 0 序列中的另一个采样是最后采样。 1 第 8 个采样为序列的最后一个采样 可在任意的采样位置结束序列。软件必须在队列中的某个位置处设置一个 ENDn 位。当某一采样已设置了 ENDn 位，其后的采样就不会发出转换请求（即使该域值不为零）。

位/域	名称	类型	复位	描述
28	D7	R/W	0	第 8 个采样动作的差分输入选择 值 描述 0 模拟输入信号为单端信号。 1 该模拟输入是差分采样。相应的 ADCSSMUXn 半字节必须被设置为对编号 i，即成对的输入为“2i”和“2i+1”。 由于温度传感器没有差分选项，因此当 TS7 位置 1 时，此标志位不得置位。
27	TS6	R/W	0	第 7 个采样动作的温度传感器选择 值 描述 0 在执行采样序列的第 7 个采样动作时读取 ADCSSMUXn 寄存器指定的输入管脚。 1 在执行采样序列的第 7 个采样动作时读取温度传感器。
26	IE6	R/W	0	第 7 个采样动作的中断启用 值 描述 0 不产生原始中断 1 在第 7 个采样转换结束后，原始中断信号（INR0 位）生效。若 ADCIM 寄存器的 MASK0 位置位，就会向中断控制器提交中断。 允许同一序列中多个采样动作各自产生中断。
25	END6	R/W	0	第 7 个采样是序列末端 值 描述 0 序列中的另一个采样是最后采样。 1 第 7 个采样为序列中的最后一个采样。 可在任意的采样位置结束序列。软件必须在队列中的某个位置处设置一个 ENDn 位。当某一采样已设置了 ENDn 位，其后的采样就不会发出转换请求（即使该域值不为零）。
24	D6	R/W	0	第 7 个采样动作的差分输入选择 值 描述 0 模拟输入信号为单端信号。 1 该模拟输入是差分采样。相应的 ADCSSMUXn 半字节必须被设置为对编号 i，即成对的输入为“2i”和“2i+1”。 由于温度传感器没有差分选项，因此当 TS6 位置 1 时，此标志位不得置位。
23	TS5	R/W	0	第 6 个采样动作的温度传感器选择 值 描述 0 在执行采样序列的第 6 个采样动作时读取 ADCSSMUXn 寄存器指定的输入管脚。 1 在执行采样序列的第 6 个采样动作时读取温度传感器。

位/域	名称	类型	复位	描述
22	IE5	R/W	0	第 6 个采样动作的中断启用 值 描述 0 不产生原始中断 1 在第 6 个采样转换结束后，原始中断信号 （INR0 位）生效。若 ADCIM 寄存器的 MASK0 位置位，就会向中断控制器提交中断。 允许同一序列中多个采样动作各自产生中断。
21	END5	R/W	0	第 6 个采样是序列末端 值 描述 0 序列中的另一个采样是最后采样。 1 第 6 个采样为序列中的最后一个采样。 可在任意的采样位置结束序列。软件必须在队列中的某个位置处设置一个 ENDn 位。当某一采样已设置了 ENDn 位，其后的采样就不会发出转换请求（即使该域值不为零）。
20	D5	R/W	0	第 6 个采样动作的差分输入选择 值 描述 0 模拟输入信号为单端信号。 1 该模拟输入是差分采样。相应的 ADCSSMUXn 半字节必须被设置为对编号 i，即成对的输入为“2i”和“2i+1”。 由于温度传感器没有差分选项，因此当 TS5 位置 1 时，此标志位不得置位。
19	TS4	R/W	0	第 5 个采样动作的温度传感器选择 值 描述 0 在执行采样序列的第 5 个采样动作时读取 ADCSSMUXn 寄存器指定的输入管脚。 1 在执行采样序列的第 5 个采样动作时读取温度传感器。
18	IE4	R/W	0	第 5 个采样动作的中断启用 值 描述 0 不产生原始中断 1 在第 5 个采样转换结束后，原始中断信号 （INR0 位）生效。若 ADCIM 寄存器的 MASK0 位置位，就会向中断控制器提交中断。 允许同一序列中多个采样动作各自产生中断。
17	END4	R/W	0	第 5 个采样是序列末端 值 描述 0 序列中的另一个采样是最后采样。 1 第 5 个采样为序列中的最后一个采样。 可在任意的采样位置结束序列。软件必须在队列中的某个位置处设置一个 ENDn 位。当某一采样已设置了 ENDn 位，其后的采样就不会发出转换请求（即使该域值不为零）。

位/域	名称	类型	复位	描述
16	D4	R/W	0	第 5 个采样动作的差分输入选择 值 描述 0 模拟输入信号为单端信号。 1 该模拟输入是差分采样。相应的 ADCSSMUXn 半字节必须被设置为对编号 i，即成对的输入为“2i”和“2i+1”。 由于温度传感器没有差分选项，因此当 TS4 位置 1 时，此标志位不得置位。
15	TS3	R/W	0	第 4 个采样动作的温度传感器选择 值 描述 0 在执行采样序列的第 4 个采样动作时读取 ADCSSMUXn 寄存器指定的输入管脚。 1 在执行采样序列的第 4 个采样动作时读取温度传感器。
14	IE3	R/W	0	第 4 个采样动作的中断启用 值 描述 0 不产生原始中断 1 在第 4 个采样转换结束后，原始中断信号（INR0 位）生效。若 ADCIM 寄存器的 MASK0 位置位，就会向中断控制器提交中断。 允许同一序列中多个采样动作各自产生中断。
13	END3	R/W	0	第 4 个采样是序列末端 值 描述 0 序列中的另一个采样是最后采样。 1 第 4 个采样为序列中的最后一个采样。 可在任意的采样位置结束序列。软件必须在队列中的某个位置处设置一个 ENDn 位。当某一采样已设置了 ENDn 位，其后的采样就不会发出转换请求（即使该域值不为零）。
12	D3	R/W	0	第 4 个采样动作的差分输入选择 值 描述 0 模拟输入信号为单端信号。 1 该模拟输入是差分采样。相应的 ADCSSMUXn 半字节必须被设置为对编号 i，即成对的输入为“2i”和“2i+1”。 由于温度传感器没有差分选项，因此当 TS3 位置 1 时，此标志位不得置位。
11	TS2	R/W	0	第 3 个采样动作的温度传感器选择 值 描述 0 在执行采样序列的第 3 个采样动作时读取 ADCSSMUXn 寄存器指定的输入管脚。 1 在执行采样序列的第 3 个采样动作时读取温度传感器。

位/域	名称	类型	复位	描述
10	IE2	R/W	0	第 3 个采样动作的中断启用 值 描述 0 不产生原始中断 1 在第 3 个采样转换结束后，原始中断信号（INR0 位）生效。若 ADCIM 寄存器的 MASK0 位置位，就会向中断控制器提交中断。 允许同一序列中多个采样动作各自产生中断。
9	END2	R/W	0	第 3 个采样是序列末端 值 描述 0 序列中的另一个采样是最后采样。 1 第 3 个采样为序列中的最后一个采样。 可在任意的采样位置结束序列。软件必须在队列中的某个位置处设置一个 ENDn 位。当某一采样已设置了 ENDn 位，其后的采样就不会发出转换请求（即使该域值不为零）。
8	D2	R/W	0	第 3 个采样动作的差分输入选择 值 描述 0 模拟输入信号为单端信号。 1 该模拟输入是差分采样。相应的 ADCSSMUXn 半字节必须被设置为对编号 i，即成对的输入为“2i”和“2i+1”。 由于温度传感器没有差分选项，因此当 TS2 位置 1 时，此标志位不得置位。
7	TS1	R/W	0	第 2 个采样动作的温度传感器选择 值 描述 0 在执行采样序列的第 2 个采样动作时读取 ADCSSMUXn 寄存器指定的输入管脚。 1 在执行采样序列的第 2 个采样动作时读取温度传感器。
6	IE1	R/W	0	第 2 个采样动作的中断启用 值 描述 0 不产生原始中断 1 在第 2 个采样转换结束后，原始中断信号（INR0 位）生效。若 ADCIM 寄存器的 MASK0 位置位，就会向中断控制器提交中断。 允许同一序列中多个采样动作各自产生中断。
5	END1	R/W	0	第 2 个采样是序列末端 值 描述 0 序列中的另一个采样是最后采样。 1 第 2 个采样为序列中的最后一个采样。 可在任意的采样位置结束序列。软件必须在队列中的某个位置处设置一个 ENDn 位。当某一采样已设置了 ENDn 位，其后的采样就不会发出转换请求（即使该域值不为零）。

位/域	名称	类型	复位	描述
4	D1	R/W	0	第 2 个采样动作的差分输入选择 值 描述 0 模拟输入信号为单端信号。 1 该模拟输入是差分采样。相应的 ADCSSMUXn 半字节必须被设置为对编号 i，即成对的输入为“2i”和“2i+1”。 由于温度传感器没有差分选项，因此当 TS1 位置 1 时，此标志位不得置位。
3	TS0	R/W	0	第 1 个采样动作的温度传感器选择 值 描述 0 在执行采样序列的第 1 个采样动作时读取 ADCSSMUXn 寄存器指定的输入管脚。 1 在执行采样序列的第 1 个采样动作时读取温度传感器。
2	IE0	R/W	0	第 1 个采样动作的中断启用 值 描述 0 不产生原始中断 1 在第 1 个采样转换结束后，原始中断信号（INR0 位）生效。若 ADCIM 寄存器的 MASK0 位置位，就会向中断控制器提交中断。 允许同一序列中多个采样动作各自产生中断。
1	END0	R/W	0	第 1 个采样是序列末端 值 描述 0 序列中的另一个采样是最后采样。 1 第 1 个采样为序列中的最后一个采样。 可在任意的采样位置结束序列。软件必须在队列中的某个位置处设置一个 ENDn 位。当某一采样已设置了 ENDn 位，其后的采样就不会发出转换请求（即使该域值不为零）。
0	D0	R/W	0	第 1 个采样动作的差分输入选择 值 描述 0 模拟输入信号为单端信号。 1 该模拟输入是差分采样。相应的 ADCSSMUXn 半字节必须被设置为对编号 i，即成对的输入为“2i”和“2i+1”。 由于温度传感器没有差分选项，因此当 TS0 位置 1 时，此标志位不得置位。

寄存器 16: ADC 采样序列结果 FIFO 寄存器 0（ADCSSFIFO0），偏移量 0x048
寄存器 17: ADC 采样序列结果 FIFO 寄存器 1（ADCSSFIFO1），偏移量 0x068
寄存器 18: ADC 采样序列结果 FIFO 寄存器 2（ADCSSFIFO2），偏移量 0x088
寄存器 19: ADC 采样序列结果 FIFO 寄存器 3（ADCSSFIFO3），偏移量 0x0A8

重要: 本寄存器为读敏感型寄存器。有关详细信息，请参阅寄存器描述部分。

本寄存器用于保存采样序列发生器采集的采样转换结果（ADCSSFIFO0 寄存器用于采样序列发生器 0，ADCSSFIFO1 寄存器用于采样序列发生器 1，ADCSSFIFO2 寄存器用于采样序列发生器 2，ADCSSFIFO3 寄存器用于采样序列发生器 3）。每读1次本寄存器可返回1个转换结果数据，依次返回第1个采样结果、第2个采样结果……依此类推，直到FIFO被读空。如果 FIFO 因未能及时得到软件处理而出现了上溢或下溢状况，可通过 ADCOSTAT 寄存器和 ADCUSTAT 寄存器进行查询。

ADC 采样序列结果 FIFO 寄存器 n（ADCSSFIFOn）

ADC0 基址: 0x4003.8000
ADC1 基址: 0x4003.9000
偏移量 0x048
类型 RO, 复位 -

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	保留															
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	保留				DATA											
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	-	-	-	-	-	-	-	-	-	-	-	-

位/域	名称	类型	复位	描述
31:12	保留	RO	0x0000.0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
11:0	DATA	RO	-	转换结果数据

寄存器 20: ADC 采样序列 FIFO 0 状态寄存器（ADCSSFSTAT0），偏移量 0x04C

寄存器 21: ADC 采样序列 FIFO 1 状态寄存器（ADCSSFSTAT1），偏移量 0x06C

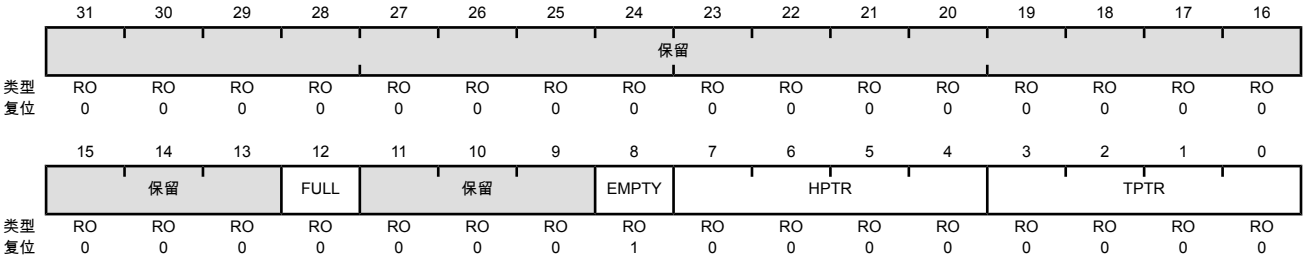
寄存器 22: ADC 采样序列 FIFO 2 状态寄存器（ADCSSFSTAT2），偏移量 0x08C

寄存器 23: ADC 采样序列 FIFO 3 状态寄存器（ADCSSFSTAT3），偏移量 0x0AC

本寄存器为采样序列发生器的状态提供了一个窗口，可藉此查询满/空状态信息以及头指针、尾指针的位置。复位值 0x100 表示 FIFO 为空，其头尾指针都指向索引值 0。ADCSSFSTAT0 寄存器代表 FIFO0 的状态（8 个单元）；ADCSSFSTAT1 寄存器代表 FIFO1 的状态（4 个单元）；ADCSSFSTAT2 寄存器代表 FIFO2 的状态（4 个单元）；ADCSSFSTAT3 寄存器代表 FIFO3 的状态（1 个单元）。

ADC 采样序列 FIFO n 状态寄存器（ADCSSFSTATn）

ADC0 基址: 0x4003.8000
ADC1 基址: 0x4003.9000
偏移量 0x04C
类型 RO, 复位 0x0000.0100



位/域	名称	类型	复位	描述
31:13	保留	RO	0x0000.0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
12	FULL	RO	0	FIFO满 值 描述 0 FIFO未满 1 FIFO已满
11:9	保留	RO	0x0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
8	EMPTY	RO	1	FIFO空 值 描述 0 FIFO非空 1 FIFO空
7:4	HPTR	RO	0x0	FIFO头指针 该字段包含当前 FIFO 的“头”指针索引，也就是下一个要执行的写操作的位。 FIFO0 的有效值是 0x0-0x7；FIFO1 和 FIFO2 的有效值是 0x0-0x3；FIFO3 的有效值是 0x0。

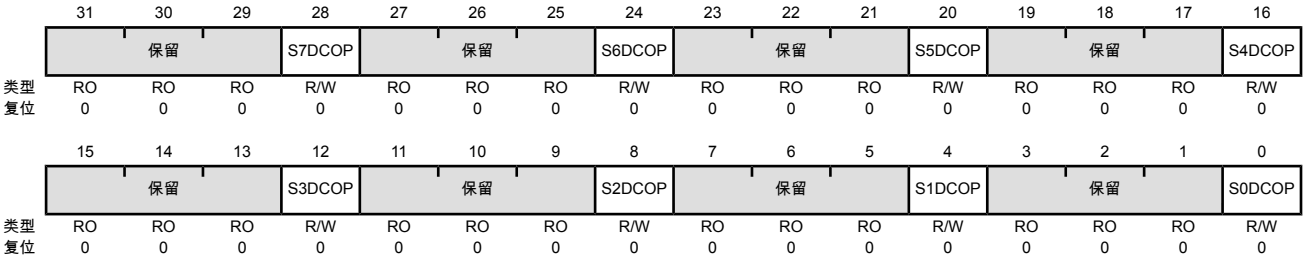
位/域	名称	类型	复位	描述
3:0	TPTR	RO	0x0	FIFO尾指针 该字段包含当前 FIFO 的“尾”指针索引，也就是下一个要读取的位。 FIFO0 的有效值是 0x0-0x7；FIFO1 和 FIFO2 的有效值是 0x0-0x3；FIFO3 的有效值是 0x0。

寄存器 24: ADC 采样序列寄存器 0（ADCSSOP0），偏移量 0x050

本寄存器用于设置如何处理采样序列发生器0的各个采样结果，是保存到采样序列FIFO0中还是传递给数字比较器。

ADC 采样序列寄存器 0 (ADCSSOP0)

ADC0 基址: 0x4003.8000
ADC1 基址: 0x4003.9000
偏移量 0x050
类型 R/W, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:29	保留	RO	0x0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
28	S7DCOP	R/W	0	采样7数字比较器工作 值 描述 0 18 个采样动作的结果保存到采样序列器 FIFO0 中。 1 18个采样动作的结果不写入 FIFO，而是发送给指定的数字比较器（由 ADCSSDC0 寄存器的 S7DCSEL 位域指定）。
27:25	保留	RO	0x0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
24	S6DCOP	R/W	0	采样6数字比较器工作 用于定义第 7 个采样动作，定义方法与 S7DCOP 相同。
23:21	保留	RO	0x0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
20	S5DCOP	R/W	0	采样5数字比较器工作 定义和 S7DCOP 一样，但在第 6 个采样中使用。
19:17	保留	RO	0x0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
16	S4DCOP	R/W	0	采样4数字比较器工作 定义和 S7DCOP 一样，但在第 5 个采样中使用。
15:13	保留	RO	0x0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
12	S3DCOP	R/W	0	采样3数字比较器工作 定义和 S7DCOP 一样，但在第 4 个采样中使用。
11:9	保留	RO	0x0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。

位/域	名称	类型	复位	描述
8	S2DCOP	R/W	0	采样2数字比较器工作 定义和 S7DCOP 一样，但在第 3 个采样中使用。
7:5	保留	RO	0x0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
4	S1DCOP	R/W	0	采样1数字比较器工作 定义和 S7DCOP 一样，但在第 2 个采样中使用。
3:1	保留	RO	0x0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
0	S0DCOP	R/W	0	采样0数字比较器工作 定义和 S7DCOP 一样，但在第 1 个采样中使用。

寄存器 25: ADC 采样序列数字比较器选择寄存器 0（ADCSSDC0），偏移量 0x054

若 ADCSSOP0 寄存器的 SnDCOP 位置位，则通过本寄存器选择将采样序列 0 的指定转换采样结果传递给哪一个数字比较器。

ADC 采样序列数字比较器选择寄存器 0 (ADCSSDC0)

ADC0 基址: 0x4003.8000
ADC1 基址: 0x4003.9000
偏移量 0x054
类型 R/W, 复位 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	S7DCSEL				S6DCSEL				S5DCSEL				S4DCSEL			
类型	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	S3DCSEL				S2DCSEL				S1DCSEL				S0DCSEL			
类型	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位/域	名称	类型	复位	描述
31:28	S7DCSEL	R/W	0x0	采样 7 数字比较器的选择 若 ADCSSOP0 寄存器的 S7DCOP 位置 1，则通过本位域选择将采样序列发生器 0 的第 8 个采样发送给哪一数字比较器（及其相关的控制寄存器集） 注意: 表中未列出的值是保留值。 值 描述 0x0 数字比较器单元 0（ADCDCCMP0 和 ADCDCCTL0） 0x1 数字比较器单元 1（ADCDCCMP1 和 ADCDCCTL1） 0x2 数字比较器单元 2（ADCDCCMP2 和 ADCDCCTL2） 0x3 数字比较器单元 3（ADCDCCMP3 和 ADCDCCTL3） 0x4 数字比较器单元 4（ADCDCCMP4 和 ADCDCCTL4） 0x5 数字比较器单元 5（ADCDCCMP5 和 ADCDCCTL5） 0x6 数字比较器单元 6（ADCDCCMP6 和 ADCDCCTL6） 0x7 数字比较器单元 7（ADCDCCMP7 和 ADCDCCTL7）
27:24	S6DCSEL	R/W	0x0	采样 6 数字比较器的选择 此位域用于定义第 7 个采样，其编码方式与 S7DCSEL 相同。
23:20	S5DCSEL	R/W	0x0	采样 5 数字比较器的选择 此位域的编码方式与 S7DCSEL 相同，但用于第 6 个采样。
19:16	S4DCSEL	R/W	0x0	采样 4 数字比较器的选择 此位域的编码方式与 S7DCSEL 相同，但用于第 5 个采样。
15:12	S3DCSEL	R/W	0x0	采样 3 数字比较器的选择 此位域的编码方式与 S7DCSEL 相同，但用于第 4 个采样。
11:8	S2DCSEL	R/W	0x0	采样 2 数字比较器的选择 此位域的编码方式与 S7DCSEL 相同，但用于第 3 个采样。

位/域	名称	类型	复位	描述
7:4	S1DCSEL	R/W	0x0	采样 1 数字比较器的选择 此位域的编码方式与 S7DCSEL 相同，但用于第 2 个采样。
3:0	S0DCSEL	R/W	0x0	采样 0 数字比较器的选择 此位域的编码方式与 S7DCSEL 相同，但用于第 1 个采样。

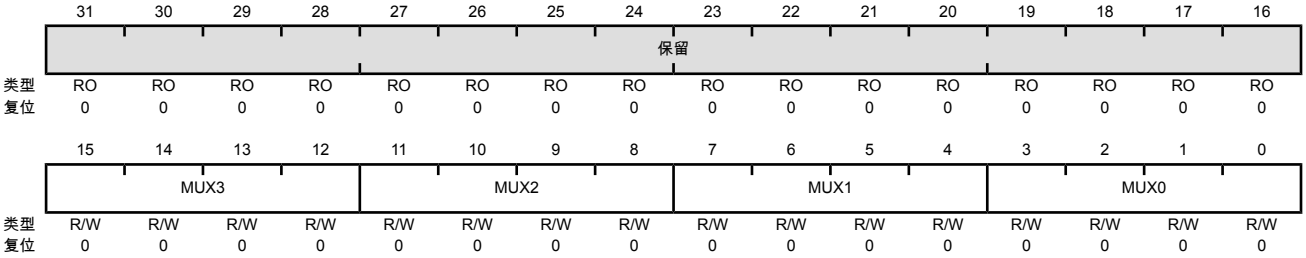
寄存器 26: ADC 采样序列输入复用选择寄存器 1（ADCSSMUX1），偏移量 0x060

寄存器 27: ADC 采样序列输入复用选择寄存器 2（ADCSSMUX2），偏移量 0x080

该寄存器在序列中定义了每个采样的模拟输入配置，使用采样序列发生器 1 或 2 来执行。本寄存器为 16 位宽，可分别定义 4 个采样的信息。详细的位描述请见在 762页 上的 ADCSSMUX0 寄存器。ADCSSMUX1 寄存器配置采样序列发生器 1，ADCSSMUX2 寄存器配置采样序列发生器 2。

ADC 采样序列输入复用选择寄存器 n (ADCSSMUXn)

ADC0 基址: 0x4003.8000
ADC1 基址: 0x4003.9000
偏移量 0x060
类型 R/W, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:16	保留	RO	0x0000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
15:12	MUX3	R/W	0x0	第 4 个采样的输入选择
11:8	MUX2	R/W	0x0	第 3 个采样的输入选择
7:4	MUX1	R/W	0x0	第 2 个采样的输入选择
3:0	MUX0	R/W	0x0	第 1 个采样的输入选择

寄存器 28: ADC 采样序列控制寄存器 1（ADCSSCTL1），偏移量 0x064

寄存器 29: ADC 采样序列控制寄存器 2（ADCSSCTL2），偏移量 0x084

这些寄存器包含在序列中每个采样的配置信息，使用采样序列发生器 1 或 2 来执行。在配置一个采样序列时，不管该序列中包含几个有效的采样动作，其最后一个有效采样动作的 END 位必须置位。本寄存器为16位宽，可分别定义4个采样的信息。详细的位描述请见在 763页 上的 ADCSSCTL0 寄存器。ADCSSCTL1 寄存器配置采样序列发生器 1，ADCSSCTL2寄存器配置采样序列发生器 2。

ADC 采样序列控制寄存器 n (ADCSSCTLn)

ADC0 基址: 0x4003.8000
ADC1 基址: 0x4003.9000
偏移量 0x064
类型 R/W, 复位 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	保留															
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	TS3	IE3	END3	D3	TS2	IE2	END2	D2	TS1	IE1	END1	D1	TS0	IE0	END0	D0
类型	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位/域	名称	类型	复位	描述
31:16	保留	RO	0x0000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
15	TS3	R/W	0	第 4 个采样动作的温度传感器选择 值 描述 0 在执行采样序列的第 4 个采样动作时读取 ADCSSMUXn 寄存器指定的输入管脚。 1 在执行采样序列的第 4 个采样动作时读取温度传感器。
14	IE3	R/W	0	第 4 个采样动作的中断启用 值 描述 0 不产生原始中断 1 在第 4 个采样转换结束后，原始中断信号（INR0 位）生效。若 ADCIM 寄存器的 MASK0 位置位，就会向中断控制器提交中断。 允许同一序列中多个采样动作各自产生中断。
13	END3	R/W	0	第 4 个采样是序列末端 值 描述 0 序列中的另一个采样是最后采样。 1 第 4 个采样为序列中的最后一个采样。 可在任意的采样位置结束序列。软件必须在队列中的某个位置处设置一个 ENDn 位。当某一采样已设置了 ENDn 位，其后的采样就不会发出转换请求（即使该域值不为零）。

位/域	名称	类型	复位	描述
12	D3	R/W	0	第 4 个采样动作的差分输入选择 值 描述 0 模拟输入信号为单端信号。 1 该模拟输入是差分采样。相应的 ADCSSMUXn 半字节必须被设置为对编号 i，即成对的输入为“2i”和“2i+1”。 由于温度传感器没有差分选项，因此当 TS3 位置 1 时，此标志位不得置位。
11	TS2	R/W	0	第 3 个采样动作的温度传感器选择 值 描述 0 在执行采样序列的第 3 个采样动作时读取 ADCSSMUXn 寄存器指定的输入管脚。 1 在执行采样序列的第 3 个采样动作时读取温度传感器。
10	IE2	R/W	0	第 3 个采样动作的中断启用 值 描述 0 不产生原始中断 1 在第 3 个采样转换结束后，原始中断信号（INR0 位）生效。若 ADCIM 寄存器的 MASK0 位置位，就会向中断控制器提交中断。 允许同一序列中多个采样动作各自产生中断。
9	END2	R/W	0	第 3 个采样是序列末端 值 描述 0 序列中的另一个采样是最后采样。 1 第 3 个采样为序列中的最后一个采样。 可在任意的采样位置结束序列。软件必须在队列中的某个位置处设置一个 ENDn 位。当某一采样已设置了 ENDn 位，其后的采样就不会发出转换请求（即使该域值不为零）。
8	D2	R/W	0	第 3 个采样动作的差分输入选择 值 描述 0 模拟输入信号为单端信号。 1 该模拟输入是差分采样。相应的 ADCSSMUXn 半字节必须被设置为对编号 i，即成对的输入为“2i”和“2i+1”。 由于温度传感器没有差分选项，因此当 TS2 位置 1 时，此标志位不得置位。
7	TS1	R/W	0	第 2 个采样动作的温度传感器选择 值 描述 0 在执行采样序列的第 2 个采样动作时读取 ADCSSMUXn 寄存器指定的输入管脚。 1 在执行采样序列的第 2 个采样动作时读取温度传感器。

位/域	名称	类型	复位	描述
6	IE1	R/W	0	第 2 个采样动作的中断启用 值 描述 0 不产生原始中断 1 在第 2 个采样转换结束后，原始中断信号（INR0 位）生效。若 ADCIM 寄存器的 MASK0 位置位，就会向中断控制器提交中断。 允许同一序列中多个采样动作各自产生中断。
5	END1	R/W	0	第 2 个采样是序列末端 值 描述 0 序列中的另一个采样是最后采样。 1 第 2 个采样为序列中的最后一个采样。 可在任意的采样位置结束序列。软件必须在队列中的某个位置处设置一个 ENDn 位。当某一采样已设置了 ENDn 位，其后的采样就不会发出转换请求（即使该域值不为零）。
4	D1	R/W	0	第 2 个采样动作的差分输入选择 值 描述 0 模拟输入信号为单端信号。 1 该模拟输入是差分采样。相应的 ADCSSMUXn 半字节必须被设置为对编号 i，即成对的输入为“2i”和“2i+1”。 由于温度传感器没有差分选项，因此当 TS1 位置 1 时，此标志位不得置位。
3	TS0	R/W	0	第 1 个采样动作的温度传感器选择 值 描述 0 在执行采样序列的第 1 个采样动作时读取 ADCSSMUXn 寄存器指定的输入管脚。 1 在执行采样序列的第 1 个采样动作时读取温度传感器。
2	IE0	R/W	0	第 1 个采样动作的中断启用 值 描述 0 不产生原始中断 1 在第 1 个采样转换结束后，原始中断信号（INR0 位）生效。若 ADCIM 寄存器的 MASK0 位置位，就会向中断控制器提交中断。 允许同一序列中多个采样动作各自产生中断。
1	END0	R/W	0	第 1 个采样是序列末端 值 描述 0 序列中的另一个采样是最后采样。 1 第 1 个采样为序列中的最后一个采样。 可在任意的采样位置结束序列。软件必须在队列中的某个位置处设置一个 ENDn 位。当某一采样已设置了 ENDn 位，其后的采样就不会发出转换请求（即使该域值不为零）。

位/域	名称	类型	复位	描述
0	D0	R/W	0	第 1 个采样动作的差分输入选择 值 描述 0 模拟输入信号为单端信号。 1 该模拟输入是差分采样。相应的 ADCSSMUXn 半字节必须被设置为对编号 i，即成对的输入为“2i”和“2i+1”。 由于温度传感器没有差分选项，因此当 TS0 位置 1 时，此标志位不得置位。

寄存器 30: ADC 采样序列 1 工作寄存器（ADCSSOP1），偏移量 0x070

寄存器 31: ADC 采样序列 2 工作寄存器（ADCSSOP2），偏移量 0x090

本寄存器用于设置如何处理采样序列发生器1或2的各个采样结果，是保存到采样序列FIFO中还是传递给数字比较器。ADCSSOP1 寄存器配置采样序列发生器 1，ADCSSOP2 寄存器配置采样序列发生器 2。

ADC 采样序列 n 工作寄存器 (ADCSSOPn)

ADC0 基址: 0x4003.8000
ADC1 基址: 0x4003.9000
偏移量 0x070
类型 R/W, 复位 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	保留															
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	保留			S3DCOP	保留			S2DCOP	保留			S1DCOP	保留			S0DCOP
类型	RO	RO	RO	R/W	RO	RO	RO	R/W	RO	RO	RO	R/W	RO	RO	RO	R/W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位/域	名称	类型	复位	描述
31:13	保留	RO	0x0000.0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
12	S3DCOP	R/W	0	采样3数字比较器工作 值 描述 0 第 4 个采样储存到采样序列器 FIFOn 中。 1 第 4 个采样会被发送到由 ADCSSDC0n 寄存器中的 S3DCSEL 位指定的数字比较器单元，且该值不会写入 FIFO。
11:9	保留	RO	0x0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
8	S2DCOP	R/W	0	采样2数字比较器工作 定义和 S3DCOP 一样，但在第 3 个采样中使用。
7:5	保留	RO	0x0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
4	S1DCOP	R/W	0	采样1数字比较器工作 定义和 S3DCOP 一样，但在第 2 个采样中使用。
3:1	保留	RO	0x0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
0	S0DCOP	R/W	0	采样0数字比较器工作 定义和 S3DCOP 一样，但在第 1 个采样中使用。

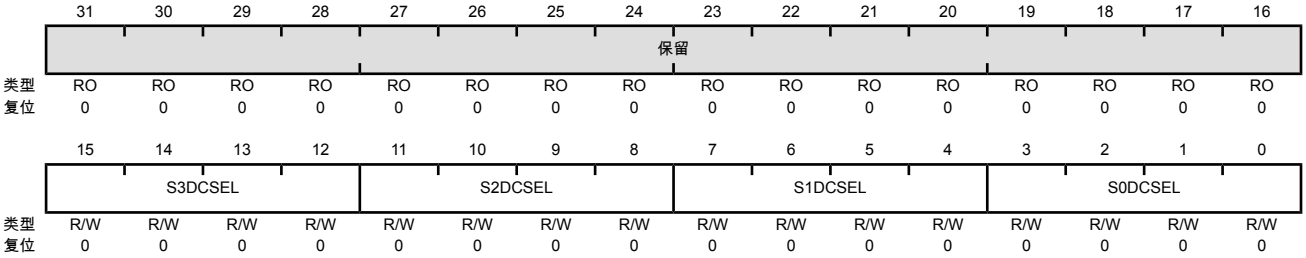
寄存器 32: ADC 采样序列数字比较器选择寄存器 1（ADCSSDC1），偏移量 0x074

寄存器 33: ADC 采样序列数字比较器选择寄存器 2（ADCSSDC2），偏移量 0x094

若 ADCSSOPn 寄存器相应的 SnDCOP 位置 1，则通过本寄存器选择将采样序列 n 的指定转换采样结果传递给哪一个数字比较器。ADCSSDC1 寄存器控制采样序列发生器 1 的选择，ADCSSDC2 寄存器控制采样序列发生器 2 的选择。

ADC 采样序列数字比较器选择寄存器 n (ADCSSDCn)

ADC0 基址: 0x4003.8000
ADC1 基址: 0x4003.9000
偏移量 0x074
类型 R/W, 复位 0x0000.0000



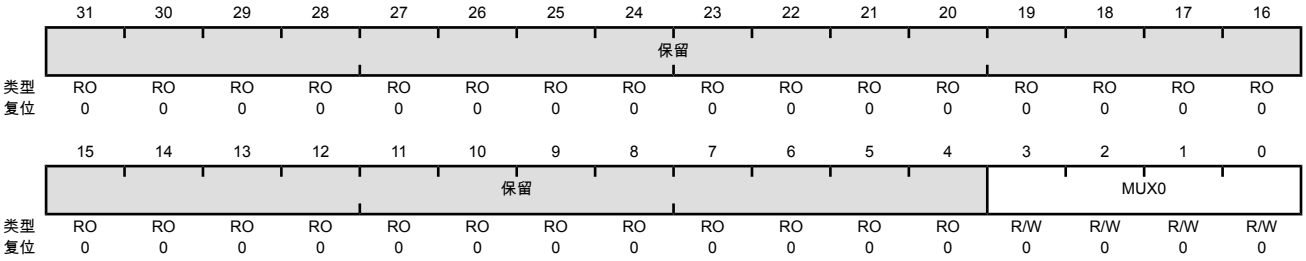
位/域	名称	类型	复位	描述
31:16	保留	RO	0x0000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
15:12	S3DCSEL	R/W	0x0	采样 3 数字比较器的选择 若 ADCSSOPn 寄存器的 S3DCOP 位置位，则通过本位域选择将采样序列器 n 的第 8 个采样发送给哪个数字比较器（及其相关的控制寄存器）。 注意: 表中未列出的值是保留值。 值 描述 0x0 数字比较器单元 0（ADCDCCMP0 和 ADCCCTL0） 0x1 数字比较器单元 1 （ADCDCCMP1 和 ADCCCTL1） 0x2 数字比较器单元 2（ADCDCCMP2 和 ADCCCTL2） 0x3 数字比较器单元 3（ADCDCCMP3 和 ADCCCTL3） 0x4 数字比较器单元 4（ADCDCCMP4 和 ADCCCTL4） 0x5 数字比较器单元 5（ADCDCCMP5 和 ADCCCTL5） 0x6 数字比较器单元 6（ADCDCCMP6 和 ADCCCTL6） 0x7 数字比较器单元 7（ADCDCCMP7 和 ADCCCTL7）
11:8	S2DCSEL	R/W	0x0	采样 2 数字比较器的选择 此位域的编码方式与 S3DCSEL 相同，但用于第 3 个采样。
7:4	S1DCSEL	R/W	0x0	采样 1 数字比较器的选择 此位域的编码方式与 S3DCSEL 相同，但用于第 2 个采样。
3:0	S0DCSEL	R/W	0x0	采样 0 数字比较器的选择 此位域的编码方式与 S3DCSEL 相同，但用于第 1 个采样。

寄存器 34: ADC 采样序列输入复用选择寄存器 3（ADCSSMUX3），偏移量 0x0A0

该寄存器为采样序列发生器 3 所执行的采样定义模拟输入配置。本寄存器为4位宽，只能定义1个采样的信息。详细的位描述请见在 762页 上的 ADCSSMUX0 寄存器。

ADC 采样序列输入复用选择寄存器 3 (ADCSSMUX3)

ADC0 基址: 0x4003.8000
ADC1 基址: 0x4003.9000
偏移量 0x0A0
类型 R/W, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:4	保留	RO	0x0000.000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
3:0	MUX0	R/W	0	第 1 个采样的输入选择

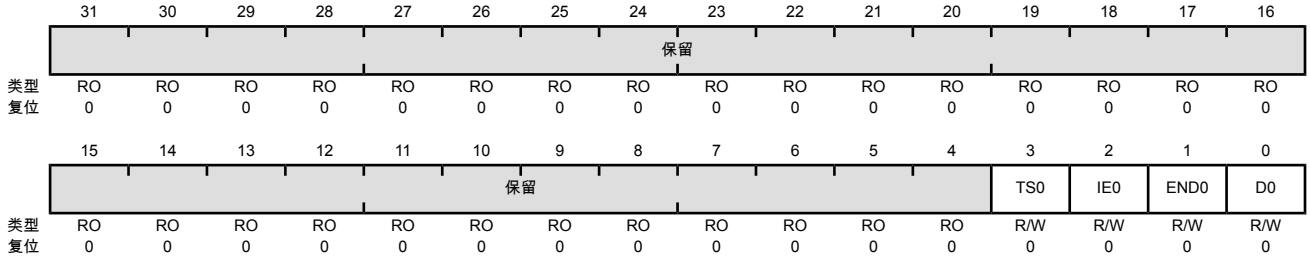
寄存器 35: ADC 采样序列控制寄存器 3（ADCSSCTL3），偏移量 0x0A4

该寄存器包含采样序列发生器 3 所执行的采样配置信息。本寄存器为4位宽，只能定义1个采样的信息。详细的位描述请见在 763页 上的 ADCSSCTL0 寄存器。

注意: 通过此寄存器配置采样序列发生器时，必须将 END0 位置位。

ADC 采样序列控制寄存器 3 (ADCSSCTL3)

ADC0 基址: 0x4003.8000
ADC1 基址: 0x4003.9000
偏移量 0x0A4
类型 R/W, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:4	保留	RO	0x0000.000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
3	TS0	R/W	0	第 1 个采样动作的温度传感器选择 值 描述 0 在执行采样序列的第 1 个采样动作时读取 ADCSSMUXn 寄存器指定的输入管脚。 1 在执行采样序列的第 1 个采样动作时读取温度传感器。
2	IE0	R/W	0	启用采样中断 值 描述 0 不产生原始中断 1 此采样转换结束后，原始中断信号（INR0 位）生效。若 ADCIM 寄存器的 MASK0 位置位，就会向中断控制器提交中断。 允许同一序列中多个采样动作各自产生中断。
1	END0	R/W	0	序列末端 开始执行单个采样序列时，必须将该位置位。 值 描述 0 继续采样和转换。 1 这是序列末端。

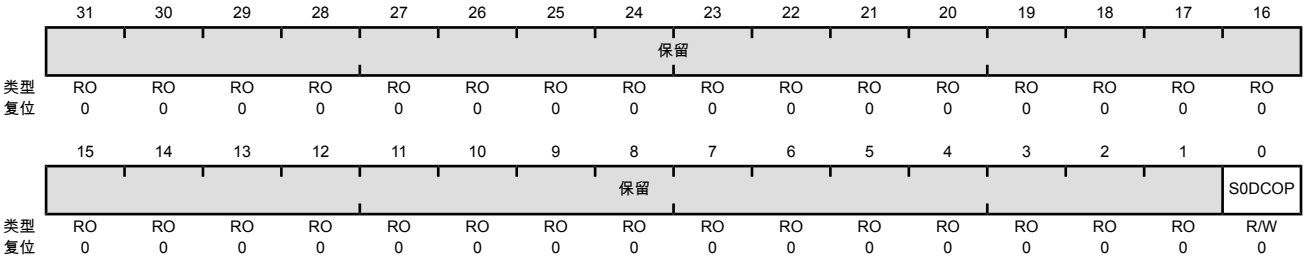
位/域	名称	类型	复位	描述
0	D0	R/W	0	采样动作的差分输入选择 值 描述 0 模拟输入信号为单端信号。 1 该模拟输入是差分采样。相应的 ADCSSMUXn 半字节必须被设置为对编号 i，即成对的输入为“2i”和“2i+1”。 由于温度传感器没有差分选项，因此当 TS0 位置 1 时，此标志位不得置位。

寄存器 36: ADC 采样序列器 3 工作寄存器（ADCSSOP3），偏移量 0x0B0

本寄存器用于设置如何处理采样序列发生器3的各个采样结果，是保存到采样序列FIFO0中还是传递给数字比较器。

ADC 采样序列器 3 工作寄存器 (ADCSSOP3)

ADC0 基址: 0x4003.8000
ADC1 基址: 0x4003.9000
偏移量 0x0B0
类型 R/W, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:1	保留	RO	0x0000.000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
0	S0DCOP	R/W	0	采样0数字比较器工作

值 描述

0 采样会被保存到采样序列 FIFO 3 中。

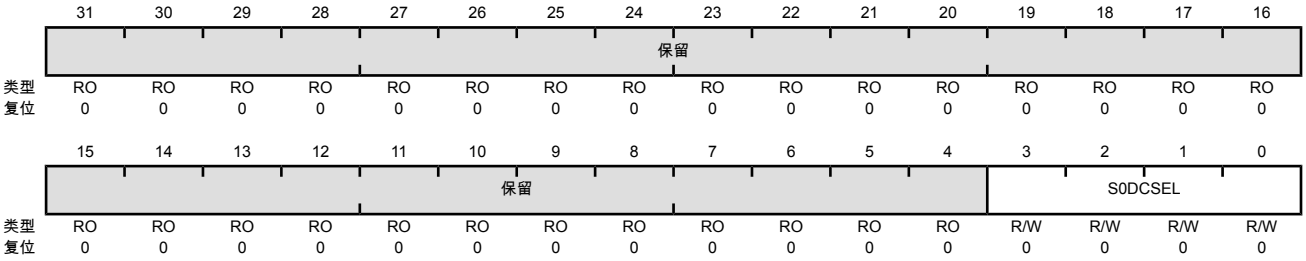
1 采样会被发送到由 ADCSSDC03 寄存器中的 S0DCSEL 位指定的数字比较器单元，且该值不会写入 FIFO。

寄存器 37: ADC 采样序列 3 数字比较器选择寄存器（ADCSSDC3），偏移量 0x0B4

若 ADCSSOP3 寄存器的 SnDCOP 位置位，则通过本寄存器选择将采样序列 3 的指定转换采样结果传递给哪一个数字比较器。

ADC 采样序列 3 数字比较器选择寄存器 (ADCSSDC3)

ADC0 基址: 0x4003.8000
ADC1 基址: 0x4003.9000
偏移量 0x0B4
类型 R/W, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:4	保留	RO	0x0000.000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
3:0	S0DCSEL	R/W	0x0	采样 0 数字比较器的选择 若 ADCSSOP3 寄存器的 S0DCOP 位置位，则通过本位域选择将采样序列器 3 的采样发送给哪个数字比较器（及其相关的控制寄存器）。 注意: 表中未列出的值是保留值。 值 描述 0x0 数字比较器单元 0（ADCDCCMP0 和 ADCCCTL0） 0x1 数字比较器单元 1 （ADCDCCMP1 和 ADCCCTL1） 0x2 数字比较器单元 2（ADCDCCMP2 和 ADCCCTL2） 0x3 数字比较器单元 3（ADCDCCMP3 和 ADCCCTL3） 0x4 数字比较器单元 4（ADCDCCMP4 和 ADCCCTL4） 0x5 数字比较器单元 5（ADCDCCMP5 和 ADCCCTL5） 0x6 数字比较器单元 6（ADCDCCMP6 和 ADCCCTL6） 0x7 数字比较器单元 7（ADCDCCMP7 和 ADCCCTL7）

寄存器 38: ADC 数字比较器复位启动条件寄存器（ADCDCRIC），偏移量 0xD00

本寄存器能够将任一数字比较器中断或触发事件恢复到其初始状态。通过这种方式及时恢复中断，可保障数字比较器中断或触发事件所用到的数据不会已经过期失效。

ADC 数字比较器复位启动条件寄存器 (ADCDCRIC)

ADC0 基址: 0x4003.8000
ADC1 基址: 0x4003.9000
偏移量 0xD00
类型 WO, 复位 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	保留								DCTRIG7	DCTRIG6	DCTRIG5	DCTRIG4	DCTRIG3	DCTRIG2	DCTRIG1	DCTRIG0
类型	RO	RO	RO	RO	RO	RO	RO	RO	WO	WO	WO	WO	WO	WO	WO	WO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	保留								DCINT7	DCINT6	DCINT5	DCINT4	DCINT3	DCINT2	DCINT1	DCINT0
类型	RO	RO	RO	RO	RO	RO	RO	RO	WO	WO	WO	WO	WO	WO	WO	WO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位/域	名称	类型	复位	描述
31:24	保留	RO	0x00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
23	DCTRIG7	WO	0	数字比较器触发事件7 值 描述 0 没有影响 1 将数字比较器7触发事件单元恢复到其初始状态 当触发事件被清除时，此标志位将自动清零。 由于数字比较器采用ADC的当前转换值以及前一转换值来决定何时产生触发事件，所以在启动新的采样序列时，务必将数字比较器恢复到其初始状态，避免仍然使用已失效的数据。该位置位后，软件应当等到该位清零后才可以继续。
22	DCTRIG6	WO	0	数字比较器触发事件6 值 描述 0 没有影响 1 将数字比较器6触发事件单元恢复到其初始状态 当触发事件被清除时，此标志位将自动清零。 由于数字比较器采用ADC的当前转换值以及前一转换值来决定何时产生触发事件，所以在启动新的采样序列时，务必将数字比较器恢复到其初始状态，避免仍然使用已失效的数据。

位/域	名称	类型	复位	描述
21	DCTRIG5	WO	0	数字比较器触发事件5 值 描述 0 没有影响 1 将数字比较器5触发事件单元恢复到其初始状态 当触发事件被清除时，此标志位将自动清零。 由于数字比较器采用ADC的当前转换值以及前一转换值来决定何时产生触发事件，所以在启动新的采样序列时，务必将数字比较器恢复到其初始状态，避免仍然使用已失效的数据。
20	DCTRIG4	WO	0	数字比较器触发事件4 值 描述 0 没有影响 1 将数字比较器4触发事件单元恢复到其初始状态 当触发事件被清除时，此标志位将自动清零。 由于数字比较器采用ADC的当前转换值以及前一转换值来决定何时产生触发事件，所以在启动新的采样序列时，务必将数字比较器恢复到其初始状态，避免仍然使用已失效的数据。
19	DCTRIG3	WO	0	数字比较器触发事件3 值 描述 0 没有影响 1 将数字比较器3触发事件单元恢复到其初始状态 当触发事件被清除时，此标志位将自动清零。 由于数字比较器采用ADC的当前转换值以及前一转换值来决定何时产生触发事件，所以在启动新的采样序列时，务必将数字比较器恢复到其初始状态，避免仍然使用已失效的数据。
18	DCTRIG2	WO	0	数字比较器触发事件2 值 描述 0 没有影响 1 将数字比较器2触发事件单元恢复到其初始状态 当触发事件被清除时，此标志位将自动清零。 由于数字比较器采用ADC的当前转换值以及前一转换值来决定何时产生触发事件，所以在启动新的采样序列时，务必将数字比较器恢复到其初始状态，避免仍然使用已失效的数据。
17	DCTRIG1	WO	0	数字比较器触发事件1 值 描述 0 没有影响 1 将数字比较器1触发事件单元恢复到其初始状态 当触发事件被清除时，此标志位将自动清零。 由于数字比较器采用ADC的当前转换值以及前一转换值来决定何时产生触发事件，所以在启动新的采样序列时，务必将数字比较器恢复到其初始状态，避免仍然使用已失效的数据。

位/域	名称	类型	复位	描述
16	DCTRIG0	WO	0	数字比较器触发事件0 值 描述 0 没有影响 1 将数字比较器0触发事件单元恢复到其初始状态 当触发事件被清除时，此标志位将自动清零。 由于数字比较器采用ADC的当前转换值以及前一转换值来决定何时产生触发事件，所以在启动新的采样序列时，务必将数字比较器恢复到其初始状态，避免仍然使用已失效的数据。
15:8	保留	RO	0x00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
7	DCINT7	WO	0	数字比较器中断7 值 描述 0 没有影响 1 将数字比较器7中断单元恢复到其初始状态 当中断被清除时，此标志位将自动清零。 由于数字比较器采用ADC的当前转换值以及前一转换值来决定何时产生中断，所以在启动新的采样序列时，务必将数字比较器恢复到其初始状态，避免仍然使用已失效的数据。
6	DCINT6	WO	0	数字比较器中断6 值 描述 0 没有影响 1 将数字比较器6中断单元恢复到其初始状态 当中断被清除时，此标志位将自动清零。 由于数字比较器采用ADC的当前转换值以及前一转换值来决定何时产生中断，所以在启动新的采样序列时，务必将数字比较器恢复到其初始状态，避免仍然使用已失效的数据。
5	DCINT5	WO	0	数字比较器中断5 值 描述 0 没有影响 1 将数字比较器5中断单元恢复到其初始状态 当中断被清除时，此标志位将自动清零。 由于数字比较器采用ADC的当前转换值以及前一转换值来决定何时产生中断，所以在启动新的采样序列时，务必将数字比较器恢复到其初始状态，避免仍然使用已失效的数据。

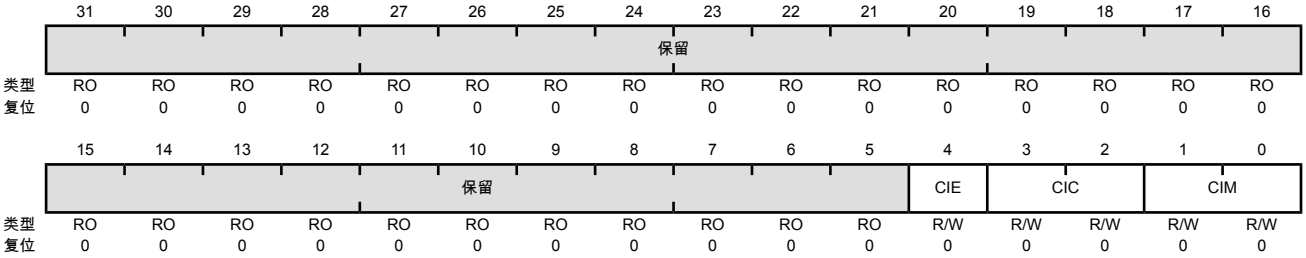
位/域	名称	类型	复位	描述
4	DCINT4	WO	0	数字比较器中断4 值 描述 0 没有影响 1 将数字比较器4中断单元恢复到其初始状态 当中断被清除时，此标志位将自动清零。 由于数字比较器采用ADC的当前转换值以及前一转换值来决定何时产生中断，所以在启动新的采样序列时，务必将数字比较器恢复到其初始状态，避免仍然使用已失效的数据。
3	DCINT3	WO	0	数字比较器中断3 值 描述 0 没有影响 1 将数字比较器3中断单元恢复到其初始状态 当中断被清除时，此标志位将自动清零。 由于数字比较器采用ADC的当前转换值以及前一转换值来决定何时产生中断，所以在启动新的采样序列时，务必将数字比较器恢复到其初始状态，避免仍然使用已失效的数据。
2	DCINT2	WO	0	数字比较器中断2 值 描述 0 没有影响 1 将数字比较器2中断单元恢复到其初始状态 当中断被清除时，此标志位将自动清零。 由于数字比较器采用ADC的当前转换值以及前一转换值来决定何时产生中断，所以在启动新的采样序列时，务必将数字比较器恢复到其初始状态，避免仍然使用已失效的数据。
1	DCINT1	WO	0	数字比较器中断1 值 描述 0 没有影响 1 将数字比较器1中断单元恢复到其初始状态 当中断被清除时，此标志位将自动清零。 由于数字比较器采用ADC的当前转换值以及前一转换值来决定何时产生中断，所以在启动新的采样序列时，务必将数字比较器恢复到其初始状态，避免仍然使用已失效的数据。
0	DCINT0	WO	0	数字比较器中断0 值 描述 0 没有影响 1 将数字比较器0中断单元恢复到其初始状态 当中断被清除时，此标志位将自动清零。 由于数字比较器采用ADC的当前转换值以及前一转换值来决定何时产生中断，所以在启动新的采样序列时，务必将数字比较器恢复到其初始状态，避免仍然使用已失效的数据。

寄存器 39: ADC 数字比较器控制寄存器 0（ADCDCCTL0），偏移量 0xE00
寄存器 40: ADC 数字比较器控制寄存器 1（ADCDCCTL1），偏移量 0xE04
寄存器 41: ADC 数字比较器控制寄存器 2（ADCDCCTL2），偏移量 0xE08
寄存器 42: ADC 数字比较器控制寄存器 3（ADCDCCTL3），偏移量 0xE0C
寄存器 43: ADC 数字比较器控制寄存器 4（ADCDCCTL4），偏移量 0xE10
寄存器 44: ADC 数字比较器控制寄存器 5（ADCDCCTL5），偏移量 0xE14
寄存器 45: ADC 数字比较器控制寄存器 6（ADCDCCTL6），偏移量 0xE18
寄存器 46: ADC 数字比较器控制寄存器 7（ADCDCCTL7），偏移量 0xE1C

本寄存器配置用于产生处理器中断触发事件的比较编码。

ADC 数字比较器控制寄存器 n (ADCDCCTLn)

ADC0 基址: 0x4003.8000
ADC1 基址: 0x4003.9000
偏移量 0xE00
类型 R/W, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:5	保留	RO	0x0000.0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
4	CIE	R/W	0	比较中断启用

值	描述
0	禁用比较中断。ADC 转换结果始终不会产生中断。
1	启用比较中断。根据 CIC 和 CIM 域的设置并通过 ADC 转换数据来判断是否应当产生中断。

位/域	名称	类型	复位	描述
3:2	CIC	R/W	0x0	比较中断条件 当 ADC 转换结果与 COMP0 值以及 COMP1 值进行比较时，此位域定义哪个工作区域产生中断。COMP0 和 COMP1 域均在 ADCDCCMPx 寄存器中定义。 值 描述 0x0 低值带 ADC 结果 < COMP0 ≤ COMP1 0x1 中值带 COMP0 ≤ ADC 结果 < COMP1 0x2 保留 0x3 高值带 COMP0 < COMP1 ≤ ADC 结果
1:0	CIM	R/W	0x0	比较中断模式 此位域定义比较器产生中断的工作模式。 值 描述 0x0 持续触发模式 只要ADC的转换结果处于选定的工作区域内，就产生中断 0x1 单次触发模式 当ADC的转换结果首次处于选定的工作区域内时，将产生一次中断 0x2 迟滞持续触发模式 只要ADC的转换结果处于选定的工作区域内，就会持续产生中断；只有当转换结果到达相反的工作区域内时才会清除迟滞状态。 0x3 迟滞单次触发模式 当 ADC 的转换结果首次处于选定的工作区域内时，将产生一次中断。只有当转换结果到达相反的工作区域时，才会清除迟滞状态。此后才可能再次中断。

寄存器 47: ADC 数字比较器范围寄存器 0（ADCDCCMP0），偏移量 0xE40
寄存器 48: ADC 数字比较器范围寄存器 1（ADCDCCMP1），偏移量 0xE44
寄存器 49: ADC 数字比较器范围寄存器 2（ADCDCCMP2），偏移量 0xE48
寄存器 50: ADC 数字比较器范围寄存器 3（ADCDCCMP3），偏移量 0xE4C
寄存器 51: ADC 数字比较器范围寄存器 4（ADCDCCMP4），偏移量 0xE50
寄存器 52: ADC 数字比较器范围寄存器 5（ADCDCCMP5），偏移量 0xE54
寄存器 53: ADC 数字比较器范围寄存器 6（ADCDCCMP6），偏移量 0xE58
寄存器 54: ADC 数字比较器范围寄存器 7（ADCDCCMP7），偏移量 0xE5C

本寄存器定义两个比较门限值，ADC模块据此判断转换的数据属于哪一工作区域。

注意: 注意：COMP1 域的值必须大于等于 COMP0 域的值，否则可能产生无法预料的后果。

ADC 数字比较器范围寄存器 n (ADCDCCMPn)

ADC0 基址: 0x4003.8000
ADC1 基址: 0x4003.9000
偏移量 0xE40
类型 R/W, 复位 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	保留				COMP1											
类型	RO	RO	RO	RO	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	保留				COMP0											
类型	RO	RO	RO	RO	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

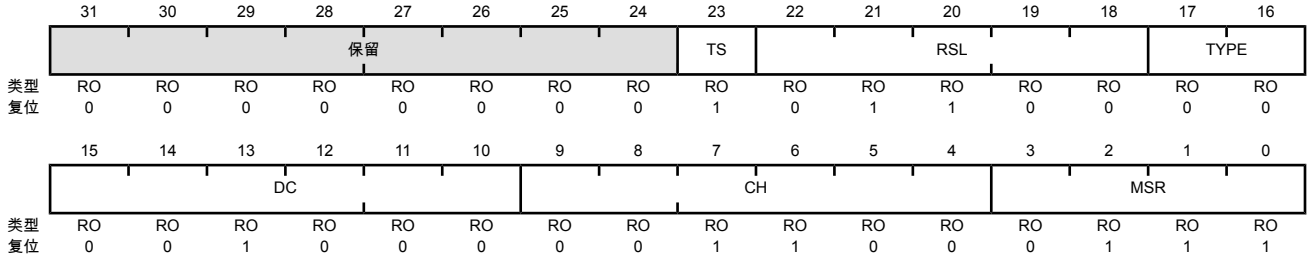
位/域	名称	类型	复位	描述
31:28	保留	RO	0x0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
27:16	COMP1	R/W	0x000	门限1 该域中的值与 ADC 转换结果进行比较。比较的结果用于决定转换结果是否在高值带范围内。 请注意，COMP1 的值必须大于等于 COMP0 的值。
15:12	保留	RO	0x0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
11:0	COMP0	R/W	0x000	门限0 该域中的值与 ADC 转换结果进行比较。比较的结果用于决定转换数据是否在低值带范围内。

寄存器 55: ADC 外设属性寄存器（ADCPP），偏移量 0xFC0

ADCPP 寄存器提供关于 ADC 模块属性的寄存器。

ADC 外设属性寄存器 (ADCPP)

ADC0 基址: 0x4003.8000
ADC1 基址: 0x4003.9000
偏移量 0xFC0
类型 RO, 复位 0x00B0.20C7



位/域	名称	类型	复位	描述
31:24	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
23	TS	RO	0x1	温度传感器 值 描述 0 该 ADC 模块没有温度传感器。 1 该 ADC 模块有一个温度传感器。 该域提供与传统 DC1 寄存器的 TEMPSNS 位相似的功能。
22:18	RSL	RO	0xC	分辨率 该域指定一个最大的二进制数值，用以表示已转换的采样。该域编码为一个二进制的数，范围在 0 到 32 位之间。
17:16	TYPE	RO	0x0	ADC 的架构 值 描述 0x0 SAR 0x1 - 0x3 保留
15:10	DC	RO	0x8	数字比较器计数 该域指定可用于转换器的 ADC 数字比较器的数量。该域编码为一个二进制的数，范围在 0 到 63 之间。 该域提供与传统 DC9 寄存器的 ADCnDCn 位相似的功能。
9:4	CH	RO	0xC	ADC 通道计数 该域指定可用于转换器的 ADC 输入通道的数量。本域编码为一个二进制的数，范围在 0 到 63 之间。 该域提供与传统 DC3 和 DC8 寄存器的 ADCnAINn 位相似的功能。

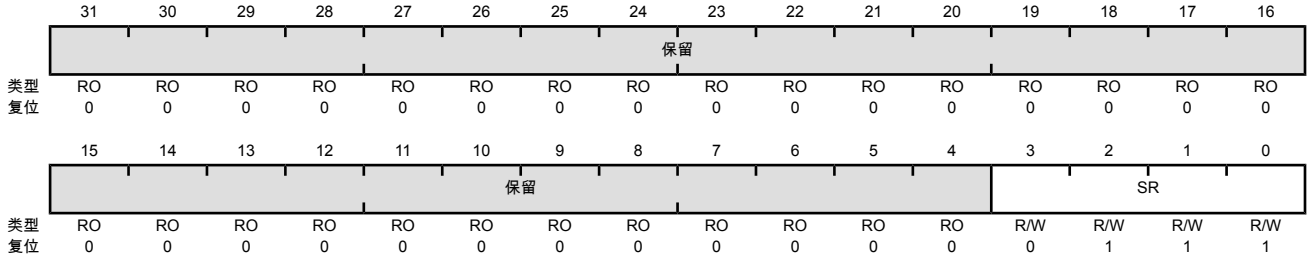
位/域	名称	类型	复位	描述
3:0	MSR	RO	0x7	最大 ADC 采样率 该域指定最大 ADC 转换数/秒。该 MSR 域编码如下： 值 描述 0x0 保留 0x1 125 ksps 0x2 保留 0x3 250 ksps 0x4 保留 0x5 500 ksps 0x6 保留 0x7 1 Msps 0x8 - 0xF 保留

寄存器 56: ADC 外设配置寄存器（ADCPC），偏移量 0xFC4

ADCPC 寄存器提供关于外设配置的信息。

ADC 外设配置寄存器 (ADCPC)

ADC0 基址: 0x4003.8000
ADC1 基址: 0x4003.9000
偏移量 0xFC4
类型 R/W, 复位 0x0000.0007



位/域	名称	类型	复位	描述
31:4	保留	RO	0x0000.0000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
3:0	SR	R/W	0x7	ADC 采样率 该域指定ADC转换数/秒，适用于运行模式、睡眠模式和深度睡眠模式。该域编码是以传统的 RCGC0 寄存器编码为基础的。该可编程的采样速率不能超过由 ADCPP 寄存器中的 MSR 位指定的最大采样速率。该 SR 域编码如下： 值 描述 0x0 保留 0x1 125 ksps 0x2 保留 0x3 250 ksps 0x4 保留 0x5 500 ksps 0x6 保留 0x7 1 Msps 0x8 - 0xF 保留

寄存器 57: ADC 时钟配置寄存器（ADCCC），偏移量 0xFC8

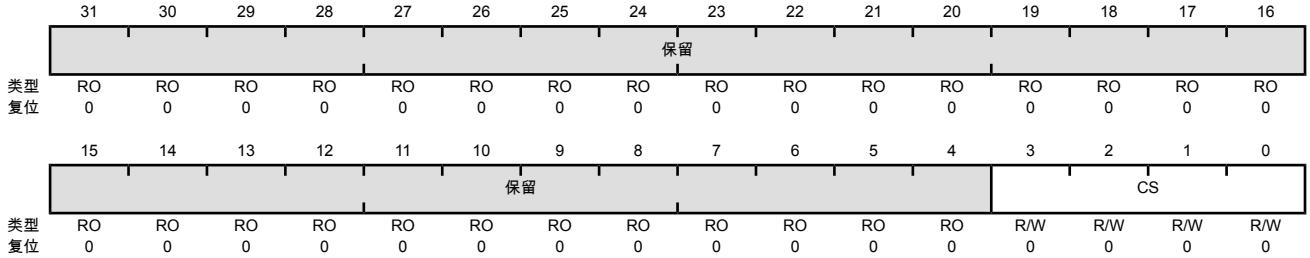
ADCCC 寄存器控制 ADC 模块的时钟源。

要将 PIOSC 用作 ADC 的时钟源，首先应给 PLL 上电，然后通过 CS 位域启用 PIOSC，再禁用 PLL。

要将 MOSC 用作 ADC 的时钟源，首先应给 PLL 上电，然后启用 ADC 模块的时钟，再禁用 PLL，最后将系统时钟切换至 MOSC。

ADC 时钟配置寄存器 (ADCCC)

ADC0 基址: 0x4003.8000
ADC1 基址: 0x4003.9000
偏移量 0xFC8
类型 R/W, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:4	保留	RO	0x0000.000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
3:0	CS	R/W	0	ADC 时钟源 下表指定了可以产生 ADC 时钟输入的时钟源，请参考 图5-5（195页）。

值	描述
0x0	可以是 16 MHz 系统时钟（如果 PLL 被旁路），也可以是通过 PLL ÷ 25 而得到的 16 MHz 时钟（默认）。 请注意：如果不使用 PLL，则系统时钟必须大于等于 16 MHz。
0x1	PIOSC PIOSC 为 ADC 提供 16 MHz 时钟源。如果 PIOSC 用作时钟源，ADC 模块可继续以深度睡眠模式运行。
0x2 - 0xF	保留

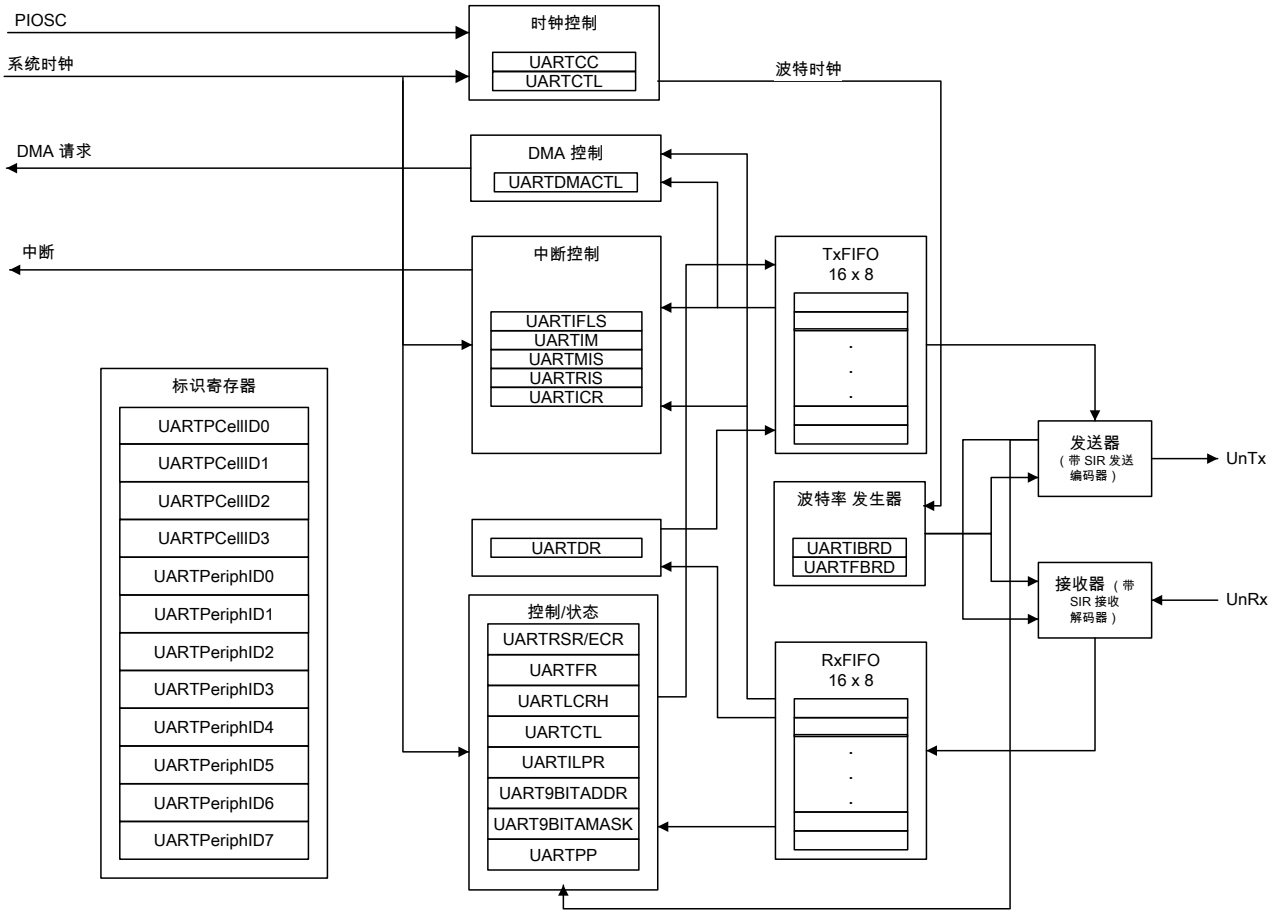
14 通用异步收发器（UART）

TM4C1231H6PM 控制器包括八个具有以下特征的通用异步收发器 (UART)：

- 可编程的波特率发生器，在常规模式（16 分频）下最高可达 5 Mbps，在高速模式（8 分频）下最高可达 10 Mbps
- 相互独立的 16×8 发送 (TX) FIFO和接收 (RX) FIFO，可降低中断服务对 CPU 的占用
- FIFO 长度可编程，包括提供传统双缓冲接口的 1 字节深的操作
- FIFO触发深度有如下级别可选：1/8、1/4、1/2、3/4或7/8；
- 标准的异步通讯位：起始位、停止位、奇偶校验位；
- 线中止的产生与检测；
- 完全可编程的串行接口特性
  - 可包含5、6、7或8个数据位
  - 可产生/检测奇偶校验位，支持偶校验位、奇校验位、粘着校验位或无校验位
  - 可产生1或2个停止位
- IrDA串行红外（SIR）编解码器
  - 可选择采用 IrDA 串行红外 (SIR) 输入输出或普通 UART 输入输出
  - 支持 IrDA SAR 编解码功能，半双工时数据传输率最高 115.2 Kbps
  - 支持标准的 3/16 位时间以及低功耗位时间（1.41~2.23 µs）
  - 可编程的内部时钟发生器，可对参考时钟源进行1~256分频以提供低功耗位时间
- 支持与 ISO 7816 智能卡的通讯
- 调制解调器流量控制和状态（在 UART1 模块上）
- 支持 EIA-485（9 位）
- 提供标准的基于 FIFO 深度的中断以及发送结束中断
- 用微型直接内存访问 (µDMA) 有效的传输数据
  - 相互独立的发送通道和接收通道
  - 当接收 FIFO 中有数据时产生单次请求；当接收 FIFO 到达预设的触发深度时产生猝发请求
  - 当发送 FIFO 中有空闲单元时产生单次请求；当发送 FIFO 到达预设的触发深度时产生猝发请求

14.1 结构框图

图 **14-1. UART模块的结构图**



14.2 信号描述

下表列出了 UART 模块的外部信号并描述了每个信号的功能。UART 信号通常是 GPIO 信号的备选功能，因此这些管脚在复位时默认设置为 GPIO 信号；只有 U0Rx 和 U0Tx 这 2 个管脚默认即为 UART 功能。表中“复用管脚/赋值”一列是各UART信号所对应的GPIO管脚。应对 GPIO 备用功能选择 (GPIOAFSEL) 寄存器（595页）的 AFSEL 位进行置位以选择 UART 功能。括号中的数字表示必须写入 GPIO 端口控制 (GPIOPCTL) 寄存器（612页）中 PMCn 位域的编码，以便向指定的 GPIO 端口引脚分配 UART 信号。有关如何配置 GPIO 的更多信息，请参阅“通用输入/输入端口（GPIOs）”（575页）。

表 **14-1. UART** 信号 **(64LQFP)**

管脚名称	管脚编号	管脚复用/管脚赋值	管脚类型	缓冲区类型[a]	描述
U0Rx	17	PA0 (1)	I	TTL	UART 0模块接收信号。
U0Tx	18	PA1 (1)	O	TTL	UART 0模块发送信号。
U1CTS	15 29	PC5 (8) PF1 (1)	I	TTL	UART 模块 1 CTS（Clear to Send，允许发送）调制解调器流控输入信号。

表 14-1. UART 信号 (64LQFP)（续）

管脚名称	管脚编号	管脚复用/管脚赋值	管脚类型	缓冲区类型[a]	描述
U1RTS	16 28	PC4 (8) PF0 (1)	O	TTL	UART 模块 1 RTS（Request to Send，请求发送）调制解调器流控输出线。
U1Rx	16 45	PC4 (2) PB0 (1)	I	TTL	UART 1模块接收信号。
U1Tx	15 46	PC5 (2) PB1 (1)	O	TTL	UART 1模块发送信号。
U2Rx	53	PD6 (1)	I	TTL	UART 2模块接收信号。
U2Tx	10	PD7 (1)	O	TTL	UART 2模块发送信号。
U3Rx	14	PC6 (1)	I	TTL	UART 3模块接收信号。
U3Tx	13	PC7 (1)	O	TTL	UART 3模块发送信号。
U4Rx	16	PC4 (1)	I	TTL	UART 4模块接收信号。
U4Tx	15	PC5 (1)	O	TTL	UART 4模块发送信号。
U5Rx	59	PE4 (1)	I	TTL	UART 5模块接收信号。
U5Tx	60	PE5 (1)	O	TTL	UART 5模块发送信号。
U6Rx	43	PD4 (1)	I	TTL	UART 6模块接收信号。
U6Tx	44	PD5 (1)	O	TTL	UART 6模块发送信号。
U7Rx	9	PE0 (1)	I	TTL	UART 7模块接收信号。
U7Tx	8	PE1 (1)	O	TTL	UART 7模块发送信号。

a. TTL 表示管脚的电压水平与 TTL 一致。

14.3 功能说明

每个 TM4C1231H6PM UART 可执行“并 - 串”和“串 - 并”转换功能。其功能与 16C550 UART 类似，但两者的寄存器不兼容。

用户可通过 UART 控制 (UARTCTL) 寄存器（见 823页）的 TXE 和 RXE 位对 UART 进行发送和/或接收配置。复位后，发送和接收默认都是使能的。在对任一控制寄存器编程之前，必须将 UART 禁能，这可以通过清零 UARTCTL 寄存器的 UARTEN 位来实现。假如在UART发送或接收期间进行此操作，则UART模块会在当前进行的数据会话结束后才停止运行。

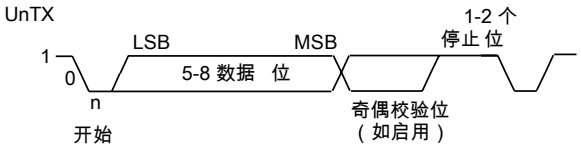
UART模块还包含串行红外（SIR）编解码模块，可直接连接红外收发器实现IrDA SIR物理层。SIR功能通过 UARTCTL 寄存器进行设置。

14.3.1 发送/接收逻辑

发送逻辑单元从发送FIFO取出数据后执行并-串转换。控制逻辑输出串行位流时，最先输出起始位，之后按照控制寄存器的配置依次输出若干数据位（最低有效位在前）、奇偶校验位和停止位。详见图14-2（802页）。

接收逻辑单元在检测到有效的起始脉冲后，对接收到的串行位码流执行串-并转换。在接收过程中还要进行溢出错误检测、奇偶校验、帧错误检测、线中止检测，并将这些状态随数据一同写入接收FIFO中。

图 14-2. UART字符帧



14.3.2 波特率的产生

波特率分频系数是由16位整数部分和6位小数部分组成的22位二进制数。整数部分和小数部分共同确定分频系数，并由此决定位时间。波特率分频值支持小数部分，使得 UART 可以产生各种标准波特率。

16 位整数通过 UART 波特率分频值整数 (UARTIBRD) 寄存器（见 819页）进行加载；而 6 位小数则通过 UART 波特率分频值小数 (UARTFBRD) 寄存器（见 820页）进行加载。波特率分频值 (BRD) 和系统时钟之间具有以下关系（其中 *BRDI* 是 BRD 的整数部分，*BRDF* 是小数部分，之间用一个小数点隔开。）

```
BRD = BRDI + BRDF = UARTSysClk / (ClkDiv * 波特率)
```

式中 UARTSysClk 是连接到 UART 模块的系统时钟，ClkDiv 是一个常数，取值为 16（UARTCTL 寄存器的 HSE = 0 时）或 8（HSE = 1 时）。默认情况下，该系统时钟为“时钟控制”（193页）中描述的主系统时钟。另外，UART 可根据内部精确振荡器 (PIOSC) 计时，不受系统时钟选择的影响。这使 UART 时钟能够独立于系统时钟 PLL 设置编程。请参考 UARTCC 寄存器获取更多详细信息。

6 位小数部分（即写入 UARTFBRD 寄存器 DIVFRAC 位域的数值）的计算方法是：将波特率除数的小数部分乘以 64，之后加 0.5 以抵消舍入误差：

```
UARTFBRD[DIVFRAC] = integer(BRDF * 64 + 0.5)
```

UART 模块产生内部波特率参考时钟，其频率为波特率的 8 或 16 倍（取决于 UARTCTL 寄存器第 5 位 HSE 的设置），分别称为 Baud8 或 Baud16。此参考时钟一方面经过8分频或16分频后产生发送时钟，另一方面在接收过程中用于错误检测。请注意，在 ISO 7816 智能卡模式（在 UARTCTL 寄存器中的 SMART 位置位时）下，HSE 位的状态对时钟产生没有影响。

UARTIBRD 和 UARTFBRD 寄存器与 UART 线控，高字节 (UARTLCRH) 寄存器（见 821页）一起组成一个 30 位内部寄存器。这个内部寄存器只在对 UARTLCRH 寄存器执行写操作时才会更新，因此更改波特率除数之后必须写一次 UARTLCRH 寄存器，更改内容才会生效。

更新波特率寄存器时，有如下4种可能的操作序列：

- 写 UARTIBRD，写 UARTFBRD，写 UARTLCRH；
- 写 UARTFBRD，写 UARTIBRD，写 UARTLCRH；
- 写 UARTIBRD，写 UARTLCRH；
- 写 UARTFBRD，写 UARTLCRH

14.3.3 数据传输

数据在接收或发送时各保存在16字节深的FIFO中，接收FIFO的每个单元还有额外4位保存状态信息。当需要进行发送时，先将数据写入发送FIFO。若 UART 模块已经使能，则将按 UARTLCRH 寄存器所配置的参数开始发送数据帧。UART模块会持续发送数据，直到发送FIFO中没有可发数据为止。数据一经写入发送 FIFO（即，如果该 FIFO 不为空），UART 标志 (UARTFR) 寄存器（见 816页）中的 BUSY 位即会生效，并且在数据发送期间一直保持有效。只有当发送 FIFO 已空、并且最

后 1 个字符（包括停止位）已经从移位寄存器中发出后，BUSY 位才会失效。即使UART模块不再使能，此标志位也能指示出UART是否处于忙状态。

在接收器空闲（UnRx 信号持续为 1）且数据输入变为“低电平”（收到起始位）时，接收计数器开始运行，并且根据 UARTCTL（详见 “发送/接收逻辑”（801页））中的 HSE 位（第 5 位）的设置，在 Baud16 的第八个周期或者 Baud8 的第四个周期对数据进行采样。

如果 UnRx 信号在 Baud16 的第 8 个周期（HSE 清零）或者 Baud 8 的第 4 个周期（HSE 置位）仍然为低电平，则起始位有效且可以识别，否则即忽略该起始位。检测到有效起始位后，会按照设定的数据字符长度和 UARTCTL 中 HSE 位的值，每 16 个 Baud16 周期或每 8 个 Baud8 周期（即：每个位周期）对后续数据位进行一次采样。之后将捕捉并校验奇偶校验位（如果使能了奇偶校验）。数据长度和奇偶校验位在 UARTLCRH 寄存器中设置。

最后，通过UnRx信号为高电平则判定停止位有效，否则视为发生帧错误。若成功接收到一帧数据，则数据和与之相关的错误标志都将保存到接收FIFO中。

14.3.4 串行红外（SIR）

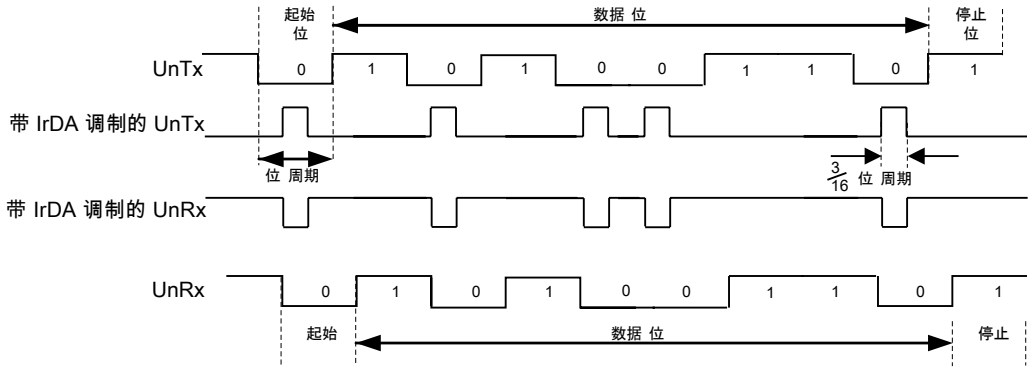
UART模块包含有IrDA串行红外（SAR）编解码模块。IrDA SAR模块能够在异步UART数据流与半双工串行SIR接口之间进行相互转换。不在片上执行任何模拟信号处理。SIR模块的作用是向UART提供数字编码输出和解码输入。使能 SIR 功能后，SIR 模块将通过 UnTx 和 UnRx 管脚实施 SIR 协议。因此这两个管脚应与片外的红外收发器连接，实现完整的IrDA SIR物理层链路。SIR 模块可以接收和发送，但是只能以半双工方式进行红外通信，所以它不能同时进行接收和发送。发送必须停止，然后才能接收数据。IrDA SIR 物理层规定发送和接收之间的最小延迟为 10ms。SIR模块具有两种工作模式：

- IrDA标准模式：输出脚上的逻辑0是一个宽度为3/16位周期（按照选定的波特率）的高脉冲；逻辑1则以静态的低电平输出。这些电平控制红外发送器的驱动装置，逢 0 电平便发送光脉冲。在接收端，接收到的光脉冲给接收器的光敏晶体管基极加电，将其输出拉至低电平并将 UART 输入管脚置为低电平。
- IrDA 低功耗模式：更改 UARTCTL 寄存器（见 823页）的适当位后，发送红外脉冲的宽度将变更为内部产生的 IrLPBaud16 信号的三倍（在额定频率 1.8432MHz 下即为 1.63µs）。

无论设备是处于正常还是低功耗 IrDA 模式，只要在首次检测到低电平后一个 IrLPBaud16 周期后，解码器仍然为低电平，则起始位视为有效。这使得正常模式的 UART 能接收低功耗模式 UART 发来的数据，实现传输小到 1.41 µs 的脉冲。因此，在低功耗和正常模式操作中，UARTILPR 寄存器的 ILPDVSR 域必须进行编程，以使 1.42 MHz < $F_{IrLPBaud16}$ < 2.12 MHz，从而产生 1.41–2.11 µs（是 IrLPBaud16 周期的三倍）的低功耗脉冲。IrLPBaud16 的最低频率须确保脉宽小于一个 IrLPBaud16 时钟周期的脉冲被忽略，但高于 1.4µs 时则为有效脉冲。

图14-3（804页）显示了含有 IrDA 调制和不含 IrDA 调制时的 UART 发送和接收信号。

图 14-3. IrDA 数据调制



无论是IrDA标准模式还是低功耗模式：

- 发送时，UART数据位作为编码的基础；
- 接收时，解码后的位转发给UART接收逻辑单元。

IrDA SIR物理层协议制定了半双工的通信链路标准，其中规定发送和接收之间至少间隔10ms的延时。UART模块本身并不提供此延时，必须由用户的应用软件予以保障。对于很多一体化的红外收发器来说这个延时非常必要，因为从发射LED耦合过来的光功率将导致红外接收电子部分发生偏置甚至饱和。此延时通常称为等待时间或接收器建立时间。

14.3.5 对ISO 7816的支持

UART模块为与ISO 7816智能卡通讯提供一些基本的支持。当 UARTCTL 寄存器的第 3 位 (SMART) 置位时，UnTx 信号被用作位时钟信号,UnRx 信号被用作连接至智能卡的半双工通讯线路。可将 GPIO 信号用于向智能卡发出复位信号。其余智能卡信号应由系统设计提供。此模式下的最大时钟速率是系统时钟/16。

当启用 ISO 7816 模式后，UARTLCRH 寄存器必须设置为：8 位数据字（WLEN 位域 6:5 配置为 0x3），带偶校验位（PEN 置位、EPS 置位）。UART 模块在这种模式下忽略 UARTLCRH 寄存器 STP2 位的设置，自动采用 2 个停止位。

假如在发送期间检测到奇偶校验错误，UnRx将在第二个停止位期间拉低。此时UART将中止当前传输过程、清空发送FIFO并丢弃其中的所有数据，同时产生一个奇偶校验错误中断。软件可以据此检测到发生的异常状况，并重新发送受影响的数据。请注意，在这种情况下UART模块并不会自动重发被丢弃的数据。

14.3.6 对调制解调器握手信号的支持

本节介绍在将 UART 作为数据终端设备 (DTE) 或数据通信设备 (DCE) 连接时，如何为 UART1 配置并使用调制解调器的流控信号。一般来说，调制解调器都是DCE，连接到调制解调器的设备都是DTE。

14.3.6.1 信号

根据 UART 是用作 DCE 还是 DTE，UART1 所提供的状态信号有所不同。用作 DTE 时，调制解调器的流控信号定义如下：

- U1CTS 允许发送信号；

- U1RTS 请求发送信号；

用作 DCE 时，调制解调器的流控信号定义如下：

- U1CTS 请求发送信号；
- U1RTS 允许发送信号；

14.3.6.2 流控

流控既可以通过硬件实现也可以通过软件实现。下面将分别介绍这两种方法：

硬件流控（**RTS/CTS**）

两设备之间的硬件流控需将发送方的 U1RTS 输出端连接到接收方的允许发送输入端，并将接收方的请求发送输出端连接到发送方的 U1CTS 输入端。

U1CTS 输入信号控制发送方。发送方只有在 U1CTS 输入端生效（拉低）的情况下才能发送数据。U1RTS 输出信号指示接收方的 FIFO 状态。U1CTS将保持信号生效,直到达到预编程的水印电平，此时表示接收方 FIFO 无法再存储更多字符。

UARTCTL 寄存器的第 15 位 (CTSEN) 和第 14 位 (RTSEN) 指定流控模式，如表14-2（805页） 中所述。

表 **14-2.** 流控模式

CTSEN	RTSEN	描述
1	1	RTS与CTS流控使能
1	0	仅CTS流控使能
0	1	仅RTS流控使能
0	0	RTS与CTS流控均禁用

请注意当 RTSEN 为 1 时，软件将无法通过 UARTCTL 寄存器的请求发送 (RTS) 位改变 U1RTS 的输出值，因此应当忽略 RTS 位的状态。

软件流控（调制解调器状态中断）

两设备间的软件流控实现方法是通过中断来指示UART的状态。使用 UARTIM 寄存器的 3 位，可为 U1CTS 信号发出中断。可使用 UARTRIS 寄存器和 UARTMIS 寄存器查看原始中断状态和屏蔽中断状态。以上中断可通过 UARTICR 寄存器予以清除。

14.3.7 9 位 UART 模式

UART 提供 9 位模式，可利用 UART9BITADDR 寄存器中的 9BITEN 位将其启用。此功能在 UART 的多分支配置中非常有用，在这种配置下，与多个从机相连的单个主机可以通过某个从机的地址或带地址字节限定符的地址集，与该从机相互通信。所有从机都会在奇偶校验位查找是否有该地址限定符，如果设置了该限定符，则对接收到的字节和预设地址进行比较。如果地址匹配，则继续接收或发送数据。如果地址不匹配，则丢弃该地址字节以及后续的所有数据字节。如果 UART 处于 9 位模式，则接收器在没有奇偶校验的模式下工作。可将该地址预先定义为与接收到的字节匹配，可通过 UART9BITADDR 寄存器进行配置。通过使用 UART9BITAMASK 寄存器中的地址屏蔽，匹配范围可扩大为地址集。默认情况下，UART9BITAMASK 为 0xFF，表示仅匹配指定的地址。

找不到匹配项时，会丢弃其余数据字节和清零的第 9 位。如果找到匹配项，则向 NVIC 发出一个中断信号，以便进一步操作。后续数据字节和清零的第 9 位存放在 FIFO 中。如果在这种情况下启用了 µDMA 和/或 FIFO 操作，软件可以屏蔽该中断，无需处理器干预。9 位模式下的所有发送操作均为数据字节且第9位清零。软件可以将奇偶校验设定改写为粘着奇偶校验，对某个字节启用奇校验，

从而改写将置位（以指示地址）的第 9 位。要让发送时间与正确的奇偶校验设定匹配，可将地址字节作为单次传输发送，而不是突发传输。发送 FIFO 并不含地址/数据位，因此软件应该相应地启用地址位。

14.3.8 FIFO操作

UART 有 2 个 16x8 的 FIFO；一个用于发送，另一个用于接收。这两个 FIFO 都通过 UART 数据 (UARTDR) 寄存器（见 811页）进行访问。对 UARTDR 寄存器执行读操作将返回 12 位的结果，其中包含 8 个数据位和 4 个错误标志位；对 UARTDR 寄存器执行写操作，可将 8 位数据写入发送 FIFO 中。

复位后，两个FIFO默认都是禁用的，其表现如同1字节深的保持寄存器。可通过对 UARTLCRH 中的 FEN 位进行置位，从而启用这两个 FIFO（821页）。

可通过 UART 标志 (UARTFR) 寄存器（见 816页）和 UART 接收状态 (UARTRSR) 寄存器监控 FIFO 的状态。而对空、满和溢出条件的监控则是由硬件来完成的。UARTFR 寄存器包含空和满的标志（TXFE、TXFF、RXFE 和 RXFF 位），而 UARTRSR 寄存器则通过 OE 位指示溢出状态。如果 FIFO 被禁用，将根据 1 字节深的保持寄存器的状态设置空和满标志。

令 FIFO 产生中断的触发点是通过 UART 中断 FIFO 深度选择 (UARTIFLS) 寄存器（见827页）来控制的。两个FIFO可分别配置为不同的触发深度。可选的触发深度包括1/8、1/4、1/2、3/4和7/8。举例来说，若设置接收FIFO的触发深度为1/4，则当UART连续收到4个数据字节后即会产生一个接收中断。复位后两个FIFO的默认触发深度都是1/2。

14.3.9 中断信号

在出现以下状况时UART模块会产生中断：

- 溢出（Overrun）错误
- 线中止错误
- 奇偶校验错误（Parity Error）
- 帧错误
- 接收超时
- 发送（当满足 UARTIFLS 寄存器中 TXIFLSEL 位定义的条件时，或 UARTCTL 寄存器的 EOT 位置 1 并且发送数据的最后 1 位已经从串行移位寄存器发出时）
- 接收（当满足 UARTIFLS 寄存器中 RXIFLSEL 位定义的条件时）

在发送给中断控制器之前，所有中断事件先进行一次逻辑或操作，因此同一时刻不管实际发生了多少中断事件，UART模块都只向中断控制器产生一个中断请求。通过读取 UART 屏蔽中断状态 (UARTMIS) 寄存器（见834页），软件可以在一个中断服务例程中处理多个中断事件。

对 UART 中断屏蔽 (UARTIM) 寄存器（见829页）中相应的 IM 位进行置位，可以定义能够触发控制器级别中断的中断事件。如果不使用中断，总是可通过 UART 原始中断状态 (UARTRIS) 寄存器（见831页）查看原始中断状态。

向 UART 中断清除 (UARTICR) 寄存器（见837页）的相应位写 1，即可（为 UARTMIS 寄存器和 UARTRIS 寄存器）清除中断。

当接收方 FIFO 不为空时，接收超时中断有效，且在一个 32 位周期内（HSE 清零时）或在一个 64 位周期内（HSE 置位时）不再接收数据。接收超时中断既可以自动清除（读出 FIFO 或保持寄存器中的所有数据，使得 FIFO 变为空状态），也可以向 UARTICR 寄存器的相应位写 1 手动清除。

发生以下事件之一时，接收中断将更改状态：

- 如果 FIFO 启用且接收 FIFO 到达设置的触发级别，RXRIS 位被置位。通过从接收 FIFO 读取数据直至其低于触发级别，或向 RXIC 位写 1 清除中断，即可将接收中断清除。
- 如果 FIFO 禁用（拥有一个位置的深度）且接收数据已填充该位置，则 RXRIS 位被置位。通过对接收 FIFO 执行一次读取，或向 RXIC 位写 1 清除中断来清零，即可将接收中断清除。

发生以下事件之一时，发送中断将更改状态：

- 如果 FIFO 启用且发送 FIFO 超出设置的触发水平，TXRIS 位被置位。发送的数据量超过某一水平时将会触发发送中断信号，因此 FIFO 载入的数据量必须超过既定触发水平，否则不会再产生发送中断信号。发送中断通过向发送 FIFO 写入数据直至其高于触发级别来清零，或通过向 TXIC 位写 1 清除中断来清零。
- 如果 FIFO 禁用（拥有一个位置的深度）且发送器单个位置中无数据存在，则 TXRIS 位被置位。发送中断通过对发送 FIFO 执行一次写入来清零，或通过向 TXIC 位写 1 清除中断来清零。

14.3.10 回送操作

可对 UARTCTL 寄存器（见823页）的 LBE 位进行置位，从而使 UART 处于内部回送模式，以便进行诊断和调试。在回送模式下，从 UnTx 输出端发送的数据将被 UnRx 输入端接收。请注意，应先对 LBE 位进行置位，然后再启用 UART。

14.3.11 DMA 操作

UART 向 µDMA 控制器接口提供独立的发送通道和接收通道。UART 的 DMA 操作通过 UART DMA 控制寄存器 (UARTDMACTL) 使能。在使能µDMA操作后，UART模块在接收FIFO或发送FIFO可以传输数据时向接收或发送通道产生µDMA请求。对于接收通道，只要接收 FIFO 中有数据，就会发出单次传输请求。只要接收 FIFO 中的数据量达到或超过 UARTIFLS 寄存器中配置的 FIFO 触发水平，就会发出突发传输请求。对于发送通道，只要发送 FIFO 中至少有一个空位，就会发出单次传输请求。只要发送 FIFO 中所含的字符少于 FIFO 触发水平，就会发出突发请求。µDMA 控制器会根据 DMA 通道的配置自动处理单次和突发 DMA 传输请求。

如需为接收通道启用 DMA 操作，请对 DMA 控制 (UARTDMACTL) 寄存器中的 RXDMAE 位进行置位。如需为发送通道启用 DMA 操作，请对 UARTDMACTL 寄存器中的 TXDMAE 位进行置位。还可以将 UART 配置为在发生接收错误时，令接收通道停止使用 DMA。如果对 UARTDMACR 寄存器的 DMAERR 位进行了置位且发生接收错误，DMA 接收请求会被自动禁用。此状况可通过清除相应的UART错误中断予以解除。

如果 µDMA 已启用，那么当 TX FIFO 或 RX FIFO 达到 UARTIFLS 寄存器中设定的触发点时，控制器会触发一个中断信号。此中断使用 UART 中断向量。因此，如果 UART 操作使用了中断，且启用了 DMA，UART 中断处理函数中必须包含对 µDMA 完成中断的处理。

注意: 要在 UART 串行移位器完成发送时触发一个中断信号，必须将 UARTCTL 寄存器的 EOT 位置位。在此配置下，只要 FIFO 完全变空且包括停止位在内的所有数据均从发送串行移位器发出，就会产生发送中断。这种情况下，系统忽略对 UARTIFLS 寄存器中 TXIFLSEL 位的设置。

请参见“微型直接存储器访问（µDMA）”（512页）以了解有关对µDMA控制器进行配置的更多信息。

14.4 初始化和配置

请按照以下步骤使能并初始化UART模块：

1. 使用 RCGCUART 寄存器（见299页）启用 UART 模块。

2. 通过RCGCGPIO寄存器启用相应GPIO模块的时钟，请参考295页。要了解要启用哪一个GPIO端口，请参考表20-5（1024页）。

3. 将相应管脚的 GPIO AFSEL 位置位（请参阅595页）。为了确定哪些 GPIO 需要配置，请参考表20-4（1019页）。

4. 按所选工作模式分别配置 GPIO 的限流和/或斜率（见597页和605页）。

5. 配置 GPIOPCTL 寄存器的 PMCn 位域，以将 UART 信号赋给相应的管脚（见612页和表20-5（1024页））。

要使用UART，必须将RCGCUART寄存器（299页）中相应的位置位，从而启用外设时钟。另外，相应GPIO模块的时钟也必须通过系统控制模块中的RCGCGPIO寄存器（295页）来启用。要了解要启用哪一个 GPIO 端口，请参考表20-5（1024页）。

本节将详细介绍配置UART模块的步骤。首先，我们假定UART时钟为20MHz，并且希望实现如下规格的UART接口：

- 波特率115200
- 8位数据位
- 1停止位
- 无奇偶校验位
- 禁用FIFO
- 不使用中断

设置 UART 首先要确定的参数就是波特率除数 (BRD)，因为应当先配置 UARTIBRD 寄存器和 UARTFBRD 寄存器，再配置 UARTLCRH 寄存器。而 BRD 可以通过““波特率的产生”（802页）”中描述的等式计算得到：

```
BRD = 20,000,000 / (16 * 115,200) = 10.8507
```

即UARTIBRD寄存器（见819页）的DIVINT位域应设为10（十进制）或0xA。加载到UARTFBRD寄存器（见820页）的值通过以下等式算出：

```
UARTFBRD[DIVFRAC] = integer(0.8507 * 64 + 0.5) = 54
```

现在BRD已经计算出来，那么UART配置可按照如下顺序进行写入：

1. 将 UARTCTL 寄存器的 UARTEN 位清零，禁用 UART 模块；

2. 将 BRD 的整数部分写入 UARTIBRD 寄存器；

3. 将 BRD 的小数部分写入 UARTFBRD 寄存器；

4. 将所需的串行工作参数写入 UARTLCRH 寄存器（在本示例中为 0x0000 0060）；

5. 通过写 UARTCC 寄存器来配置 UART 时钟源。

6. 还可以选择配置μDMA通道（见“微型直接存储器访问（μDMA）”（512页））并在UARTDMACTL寄存器中启用 DMA 选项。

7. 将 UARTCTL 寄存器的 UARTEN 位置位，以使能 UART 模块。

14.5 寄存器映射

表 14-3（809页）列出了 UART 寄存器。下表中列出的“偏移量”代表寄存器相对于该 UART 基址的十六进制地址增量：

- UART0：0x4000.C000
- UART1：0x4000.D000
- UART2：0x4000.E000
- UART3：0x4000.F000
- UART4：0x4001.0000
- UART5：0x4001.1000
- UART6：0x4001.2000
- UART7：0x4001.3000

必须先启用 UART 模块的时钟，然后才能配置寄存器（请参阅299页）。启用 UART 模块的时钟后，必须有 3 个系统时钟单位的延迟，然后才能访问 UART 模块的寄存器。

必须先禁用 UART（见823页中对 UARTCTL 寄存器 UARTEN 位的说明），然后才能对控制寄存器重新编程。如果在接收或发送期间禁用UART，则UART模块会等待当前数据会话完成后再停止运行。

表 **14-3. UART** 寄存器映射

偏移量	名称	类型	复位	描述	见页面
0x000	UARTDR	R/W	0x0000.0000	UART 数据寄存器	811
0x004	UARTRSR/UARTECR	R/W	0x0000.0000	UART 接收状态/错误清除寄存器	813
0x018	UARTFR	RO	0x0000.0090	UART 标志寄存器	816
0x020	UARTILPR	R/W	0x0000.0000	UART IrDA 低功耗寄存器	818
0x024	UARTIBRD	R/W	0x0000.0000	UART 波特率分频值整数寄存器	819
0x028	UARTFBRD	R/W	0x0000.0000	UART 波特率分频值小数寄存器	820
0x02C	UARTLCRH	R/W	0x0000.0000	UART 线控寄存器	821
0x030	UARTCTL	R/W	0x0000.0300	UART 控制寄存器	823
0x034	UARTIFLS	R/W	0x0000.0012	UART 中断 FIFO 深度选择寄存器	827
0x038	UARTIM	R/W	0x0000.0000	UART 中断屏蔽寄存器	829
0x03C	UARTRIS	RO	0x0000.0000	UART 原始中断状态寄存器	831
0x040	UARTMIS	RO	0x0000.0000	UART 屏蔽中断状态寄存器	834
0x044	UARTICR	W1C	0x0000.0000	UART 中断清除寄存器	837
0x048	UARTDMACTL	R/W	0x0000.0000	UART DMA 控制寄存器	839
0x0A4	UART9BITADDR	R/W	0x0000.0000	UART 9 位模式自身地址寄存器	840
0x0A8	UART9BITAMASK	R/W	0x0000.00FF	UART 9 位模式自身地址屏蔽寄存器	841
0xFC0	UARTPP	RO	0x0000.0003	UART 外设属性寄存器	842
0xFC8	UARTCC	R/W	0x0000.0000	UART 时钟配置寄存器	843
0xFD0	UARTPeriphID4	RO	0x0000.0000	UART 外设标识寄存器 4	844

表 14-3. UART 寄存器映射（续）

偏移量	名称	类型	复位	描述	见页面
0xFD4	UARTPeriphID5	RO	0x0000.0000	UART 外设标识寄存器 5	845
0xFD8	UARTPeriphID6	RO	0x0000.0000	UART 外设标识寄存器 6	846
0xFDC	UARTPeriphID7	RO	0x0000.0000	UART 外设标识寄存器 7	847
0xFE0	UARTPeriphID0	RO	0x0000.0060	UART 外设标识寄存器 0	848
0xFE4	UARTPeriphID1	RO	0x0000.0000	UART 外设标识寄存器 1	849
0xFE8	UARTPeriphID2	RO	0x0000.0018	UART 外设标识寄存器 2	850
0xFEC	UARTPeriphID3	RO	0x0000.0001	UART 外设标识寄存器 3	851
0xFF0	UARTPCellID0	RO	0x0000.000D	UART PrimeCell 标识寄存器 0	852
0xFF4	UARTPCellID1	RO	0x0000.00F0	UART PrimeCell 标识寄存器 1	853
0xFF8	UARTPCellID2	RO	0x0000.0005	UART PrimeCell 标识寄存器 2	854
0xFFC	UARTPCellID3	RO	0x0000.00B1	UART PrimeCell 标识寄存器 3	855

14.6 寄存器描述

本章的剩余部分按照地址偏移量由小到大的顺序依次详细介绍各寄存器。

寄存器 1: UART 数据寄存器（UARTDR），偏移量 0x000

重要: 本寄存器为读敏感型寄存器。有关详细信息，请参阅寄存器描述部分。

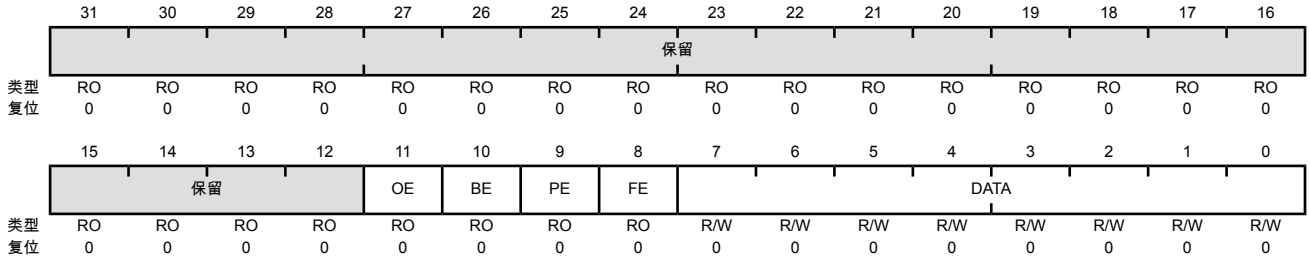
本寄存器是UART模块的数据寄存器，是与FIFO的接口。

当发送数据时，若FIFO已使能，则对本寄存器写入的数据将推入发送FIFO。如果发送FIFO被禁用，则数据仅保存在发送保持寄存器（即发送FIFO的最底部单元）中。对本寄存器执行写操作即会启动UART的发送。

当接收数据时，若FIFO已使能，则收到的数据字节以及4个状态位（线中止错误、帧错误、奇偶校验错误、溢出错误）将推入12位宽的接收FIFO中。如果接收FIFO被禁用，则数据字节和状态位保存在接收保持寄存器（即接收FIFO的最底部单元）中。对本寄存器的读操作即可获取数据字节。

UART 数据寄存器 (UARTDR)

UART0 基址: 0x4000.C000
UART1 基址: 0x4000.D000
UART2 基址: 0x4000.E000
UART3 基址: 0x4000.F000
UART4 基址: 0x4001.0000
UART5 基址: 0x4001.1000
UART6 基址: 0x4001.2000
UART7 基址: 0x4001.3000
偏移量 0x000
类型 R/W, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:12	保留	RO	0x0000.0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
11	OE	RO	0	UART溢出错误 值 描述 0 未发生数据溢出丢失 1 当FIFO已满时又收到新的数据，导致数据丢失
10	BE	RO	0	UART线中止错误 值 描述 0 未检测到线中止状况 1 检测到线中止状况：接收端被拉低的时间过长，超过一个完整字的传输时间（指包含起始位、数据位、奇偶校验位、停止位的完整一帧时间）。 在FIFO模式下，该错误与FIFO顶部的字符有关。发生中止时，只有一个0字符被加载到FIFO。仅在接收数据输入变为1（标记状态）且接收到下一个有效的起始位后，才启用下一字符。

位/域	名称	类型	复位	描述
9	PE	RO	0	UART奇偶校验错误 值 描述 0 未发生奇偶校验错误 1 所收到数据字符的奇偶校验与 UARTLCRH 寄存器第 2 位及第 7 位定义的奇偶校验不符。 在FIFO模式下，该错误与FIFO顶部的字符有关。
8	FE	RO	0	UART帧错误 值 描述 0 未发生帧错误 1 收到的数据帧没有有效的停止位（有效的停止位应当为1）
7:0	DATA	R/W	0x00	接收/发送的数据 要通过UART发送数据，则应写入此位域。 读此位域时，将返回UART收到的数据。

寄存器 2: UART 接收状态/错误清除寄存器（UARTRSR/UARTECR），偏移量 0x004

UARTRSR/UARTECR 寄存器分别是接收状态寄存器/错误清除寄存器。

除了 UARTDR 寄存器所返回的状态位外，也可以从 UARTRSR 寄存器读取接收状态。如果从该寄存器读取状态，则状态信息与读取 UARTRSR 寄存器之前从 UARTDR 读取的内容相对应。而当产生溢出状况时，溢出状态位会立即置位。

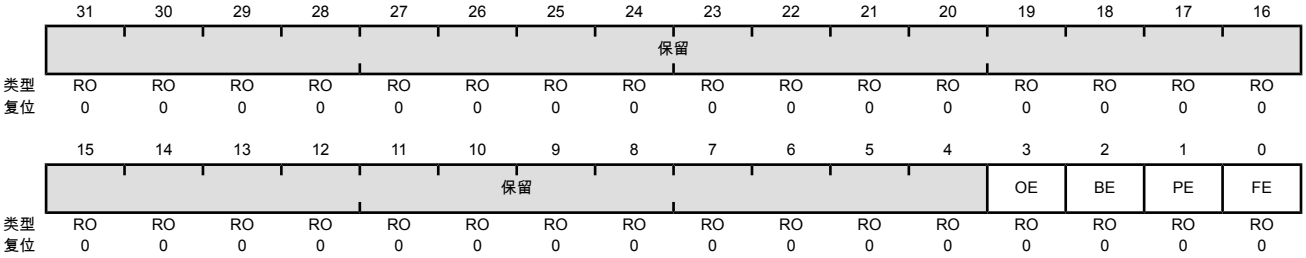
UARTRSR 寄存器不可写。

对 UARTECR 寄存器写入任何值都会清除帧错误、奇偶校验错误、中止错误和溢出错误。复位时本寄存器将清零。

只读状态寄存器

UART 接收状态/错误清除寄存器 (UARTRSR/UARTECR)

UART0 基址: 0x4000.C000
UART1 基址: 0x4000.D000
UART2 基址: 0x4000.E000
UART3 基址: 0x4000.F000
UART4 基址: 0x4001.0000
UART5 基址: 0x4001.1000
UART6 基址: 0x4001.2000
UART7 基址: 0x4001.3000
偏移量 0x004
类型 RO, 复位 0x0000.0000



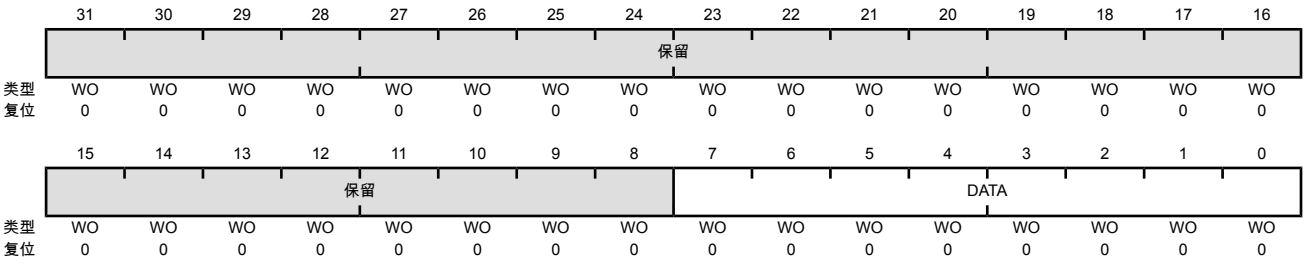
位/域	名称	类型	复位	描述
31:4	保留	RO	0x0000.000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
3	OE	RO	0	UART溢出错误 值 描述 0 未发生数据溢出丢失 1 当FIFO已满时又收到新的数据，导致数据丢失 写 UARTECR 寄存器时此标志位自动清零。 由于 FIFO 已满时不再有数据写入，所以 FIFO 的内容依然有效，只是移位寄存器的内容被覆盖了。此时，CPU 必须读取数据以便将 FIFO 清空。

位/域	名称	类型	复位	描述
2	BE	RO	0	UART线中止错误 值 描述 0 未检测到线中止状况 1 检测到线中止状况：接收端被拉低的时间过长，超过一个完整字的传输时间（指包含起始位、数据位、奇偶校验位、停止位的完整一帧时间）。 向 UARTECR 进行写操作会将该位清零。 在FIFO模式下，该错误与FIFO顶部的字符有关。发生中止时，只有一个0字符被加载到FIFO。仅在接收数据输入变为1（标记状态）且接收到下一个有效的起始位后，才启用下一字符。
1	PE	RO	0	UART奇偶校验错误 值 描述 0 未发生奇偶校验错误 1 所收到数据字符的奇偶校验与 UARTLCRH 寄存器第 2 位及第 7 位定义的奇偶校验不符。 向 UARTECR 进行写操作会将该位清零。
0	FE	RO	0	UART帧错误 值 描述 0 未发生帧错误 1 收到的数据帧没有有效的停止位（有效的停止位应当为1） 向 UARTECR 进行写操作会将该位清零。 在FIFO模式下，该错误与FIFO顶部的字符有关。

只写的错误清除寄存器

UART 接收状态/错误清除寄存器 (UARTRSR/UARTECR)

UART0 基址: 0x4000.C000
UART1 基址: 0x4000.D000
UART2 基址: 0x4000.E000
UART3 基址: 0x4000.F000
UART4 基址: 0x4001.0000
UART5 基址: 0x4001.1000
UART6 基址: 0x4001.2000
UART7 基址: 0x4001.3000
偏移量 0x004
类型 WO, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:8	保留	WO	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
7:0	DATA	WO	0x00	错误清除 对此寄存器写入任何数值都会清除帧错误、奇偶校验错误、线中止错误以及溢出错误的标志位

寄存器 3: UART 标志寄存器（UARTFR），偏移量 0x018

UARTFR 寄存器是标志寄存器。复位后，TXFF、RXFF 和 BUSY 位均为 0；而 TXFE 和 RXFE 位均为 1。CTS 位指示调制解调器流控。请注意，调制解调器位仅对 UART1 有效，对于 UART0 和 UART2，该位是保留的。

UART 标志寄存器 (UARTFR)

UART0 基址: 0x4000.C000
UART1 基址: 0x4000.D000
UART2 基址: 0x4000.E000
UART3 基址: 0x4000.F000
UART4 基址: 0x4001.0000
UART5 基址: 0x4001.1000
UART6 基址: 0x4001.2000
UART7 基址: 0x4001.3000
偏移量 0x018
类型 RO, 复位 0x0000.0090

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	保留															
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	保留								TXFE	RXFF	TXFF	RXFE	BUSY	保留		CTS
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0

位/域	名称	类型	复位	描述
31:8	保留	RO	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
7	TXFE	RO	1	UART发送FIFO空 此标志位的含义取决于 UARTLCRH 寄存器中 FEN 位的状态。 值 描述 0 UART有数据待发送 1 若 FIFO 被禁用 (FEN = 0)，则表明发送保持寄存器为空。 若 FIFO 被使能 (FEN = 1)，则表明发送 FIFO 为空。
6	RXFF	RO	0	UART接收FIFO满 此标志位的含义取决于 UARTLCRH 寄存器中 FEN 位的状态。 值 描述 0 UART还能接收数据 1 若 FIFO 被禁用 (FEN = 0)，则表明接收保持寄存器已满。 若 FIFO 被使能 (FEN = 1)，则表明接收 FIFO 已满。
5	TXFF	RO	0	UART 发送 FIFO 已满 此标志位的含义取决于 UARTLCRH 寄存器中 FEN 位的状态。 值 描述 0 发送器未满。 1 如果 FIFO 处于禁用状态（FEN 为 0），则发送保持寄存器为满。 如果 FIFO 处于启用状态（FEN 为 1），则发送 FIFO 为满。

位/域	名称	类型	复位	描述
4	RXFE	RO	1	UART 接收 FIFO 为空 此标志位的含义取决于 UARTLCRH 寄存器中 FEN 位的状态。 值 描述 0 接收器不为空。 1 如果 FIFO 处于禁用状态（FEN 为 0），则接收保持寄存器为空。 如果 FIFO 处于启用状态（FEN 为 1），则接收 FIFO 为空。
3	BUSY	RO	0	UART忙 值 描述 0 UART未处于忙状态 1 UART 正忙于发送数据。该位一直处于置位状态，直至包括所有停止位在内的全部字节都从移位寄存器发出为止。 一旦发送FIFO不为空时（不管UART是否使能）该位都会置位。
2:1	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
0	CTS	RO	0	请求发送 值 描述 0 没有发出 U1CTS 信号。 1 发出 U1CTS 信号。

寄存器 4: UART IrDA 低功耗寄存器（UARTILPR），偏移量 0x020

UARTILPR 寄存器保存 8 位的低功耗计数器除数，将系统时钟 (SysClk) 除以这个除数，即可得到低功耗 SIR 脉宽时钟。复位时此寄存器清0。

利用 UARTILPR 寄存器中的低功耗除数，将 SysClk 分频后可得到内部的 IrLPBaud16 时钟。在低功耗模式下，SIR 脉冲的宽度是 IrLPBaud16 时钟周期的三倍。低功耗分频系数值按照下式计算：

$ILPDVSR = SysClk / F_{IrLPBaud16}$

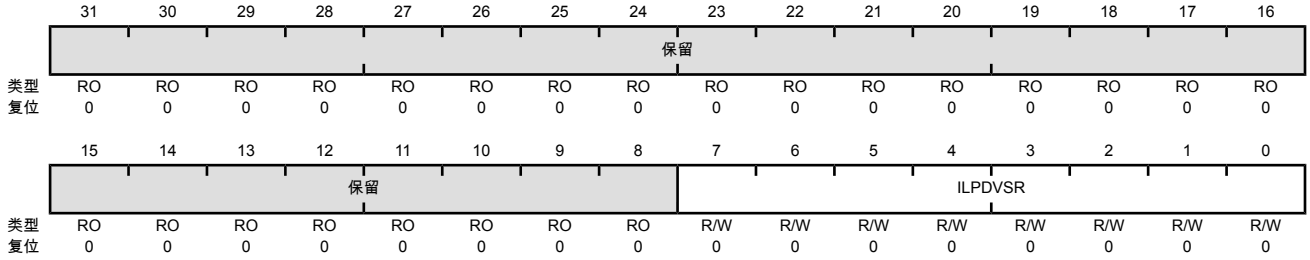
其中 $F_{IrLPBaud16}$ 标称值为 1.8432 MHz。

由于 IrLPBaud16 时钟被用于对发送的数据采样（不论所处模式），因此在低功耗和正常模式下，都必须对 ILPDVSR 域编程，以使 1.42 MHz < $F_{IrLPBaud16}$ < 2.12 MHz，产生 1.41–2.11 µs（是 IrLPBaud16 周期的三倍）的低功耗脉冲。IrLPBaud16 的最低频率须确保脉宽小于一个 IrLPBaud16 时钟周期的脉冲被忽略，但高于 1.4µs 时则为有效脉冲。

注意：　0 是非法值。如果编程为 0，将不会产生 IrLPBaud16 脉冲。

UART IrDA 低功耗寄存器 (UARTILPR)

UART0 基址: 0x4000.C000
UART1 基址: 0x4000.D000
UART2 基址: 0x4000.E000
UART3 基址: 0x4000.F000
UART4 基址: 0x4001.0000
UART5 基址: 0x4001.1000
UART6 基址: 0x4001.2000
UART7 基址: 0x4001.3000
偏移量 0x020
类型 R/W, 复位 0x0000.0000



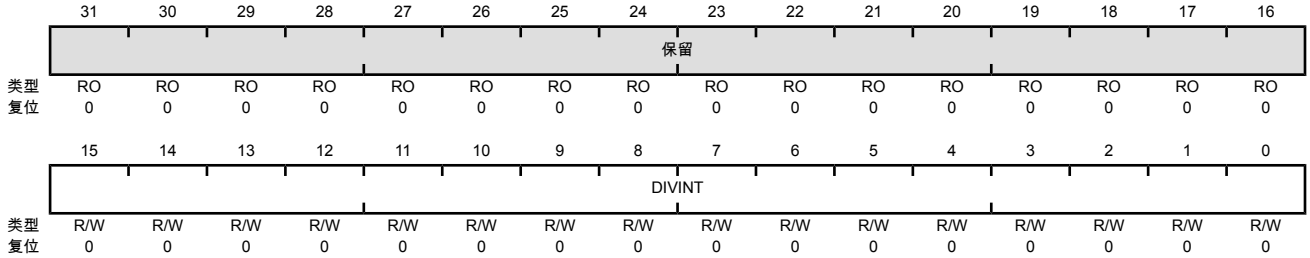
位/域	名称	类型	复位	描述
31:8	保留	RO	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
7:0	ILPDVSR	R/W	0x00	IrDA低功耗分频系数 此位域包含8位的IrDA低功耗分频系数值。

寄存器 5: UART 波特率分频值整数寄存器（UARTIBRD），偏移量 0x024

UARTIBRD 寄存器包含波特率除数的整数部分。复位时本寄存器将清零。允许的最小分频比为 1（即 UARTIBRD = 0），此时将忽略 UARTFBRD 寄存器的设置。更改 UARTIBRD 寄存器后，新的配置需在当前字符的传输/接收结束后才会生效。对波特率除数进行任何修改后，必须写一次 UARTLCRH 寄存器。详细配置信息请参考“波特率的产生”（802页）。

UART 波特率分频值整数寄存器 (UARTIBRD)

UART0 基址: 0x4000.C000
UART1 基址: 0x4000.D000
UART2 基址: 0x4000.E000
UART3 基址: 0x4000.F000
UART4 基址: 0x4001.0000
UART5 基址: 0x4001.1000
UART6 基址: 0x4001.2000
UART7 基址: 0x4001.3000
偏移量 0x024
类型 R/W, 复位 0x0000.0000



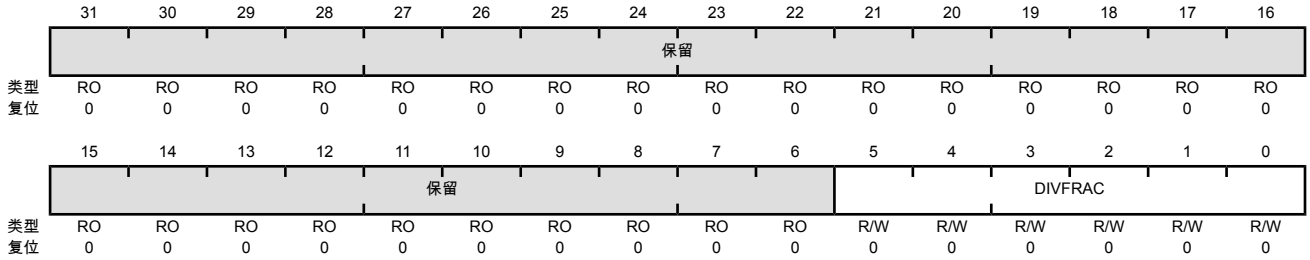
位/域	名称	类型	复位	描述
31:16	保留	RO	0x0000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
15:0	DIVINT	R/W	0x0000	整数波特率除数

寄存器 6: UART 波特率分频值小数寄存器（UARTFBRD），偏移量 0x028

UARTFBRD 寄存器包含波特率除数的小数部分。复位时本寄存器将清零。更改 UARTFBRD 寄存器后，新的配置需在当前字符的传输/接收结束后才会生效。对波特率除数进行任何修改后，必须写一次 UARTLCRH 寄存器。详细配置信息请参考“波特率的产生”（802页）。

UART 波特率分频值小数寄存器 (UARTFBRD)

UART0 基址: 0x4000.C000
UART1 基址: 0x4000.D000
UART2 基址: 0x4000.E000
UART3 基址: 0x4000.F000
UART4 基址: 0x4001.0000
UART5 基址: 0x4001.1000
UART6 基址: 0x4001.2000
UART7 基址: 0x4001.3000
偏移量 0x028
类型 R/W, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:6	保留	RO	0x0000.000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
5:0	DIVFRAC	R/W	0x0	小数波特率除数

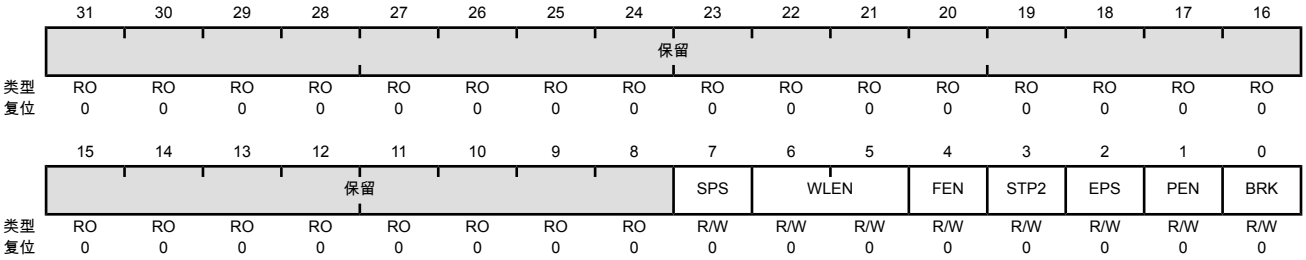
寄存器 7: UART 线控寄存器（UARTLCRH），偏移量 0x02C

UARTLCRH 寄存器称为线控寄存器。数据长度、奇偶校验位、停止位等串行通讯参数都是通过本寄存器设置的。

更改波特率除数（UARTIBRD 寄存器和/或 UARTFBRD 寄存器）后，还必须写一次 UARTLCRH 寄存器。波特率除数寄存器的写选通信号是关联在 UARTLCRH 寄存器的。

UART 线控寄存器 (UARTLCRH)

UART0 基址: 0x4000.C000
UART1 基址: 0x4000.D000
UART2 基址: 0x4000.E000
UART3 基址: 0x4000.F000
UART4 基址: 0x4001.0000
UART5 基址: 0x4001.1000
UART6 基址: 0x4001.2000
UART7 基址: 0x4001.3000
偏移量 0x02C
类型 R/W, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:8	保留	RO	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
7	SPS	R/W	0	UART粘着奇偶校验位选择 如果对 UARTLCRH 的第 1、2 和 7 位进行了置位，则发送奇偶校验位，且校验结果为 0。如果对第 1 和 7 位进行了置位且第 2 位清零，则发送奇偶校验位，且校验结果为 1。 该位清零时，粘着奇偶校验被禁能。
6:5	WLEN	R/W	0x0	UART字长度 此位域定义收发的每帧中包含的数据位数 值 描述 0x0 5 个数据位（默认） 0x1 6 位 0x2 7 位 0x3 8 位
4	FEN	R/W	0	UART使能FIFO 值 描述 0 禁用 FIFO（字符模式）。FIFO 变成 1 字节深的保持寄存器。 1 FIFO使能（接收FIFO和发送FIFO）

位/域	名称	类型	复位	描述
3	STP2	R/W	0	UART双停止位选择 值 描述 0 每帧传输结束时发送1个停止位 1 每帧传输结束时发送双停止位。接收逻辑不会检查接收的是否为双停止位。 在 ISO 7816 智能卡模式下（UARTCTL 寄存器的 SMART 位置 1），强制采用 2 个停止位。
2	EPS	R/W	0	UART偶校验位选择 值 描述 0 帧传输采用奇校验，即数据位和奇偶校验位中“1”的个数应为奇数 1 帧传输采用偶校验，即数据位和奇偶校验位中“1”的个数应为偶数 当奇偶检验被 PEN 位禁止时，该位无效。
1	PEN	R/W	0	UART奇偶校验位使能 值 描述 0 禁用奇偶校验位，并且帧中也不会包含奇偶校验位 1 使能奇偶校验位的生成和校验
0	BRK	R/W	0	UART发送线中止 值 描述 0 正常使用。 1 当前字符发送完毕后，UnTx 信号的输出一直处于低电平状态。为了正确执行中止命令，软件必须令该位处于置位状态，并且持续至少 2 帧（字符周期）。

寄存器 8: UART 控制寄存器（UARTCTL），偏移量 0x030

UARTCTL 寄存器称为控制寄存器。复位后，除了启用发送位 (TXE) 和启用接收位 (RXE) 处于置位状态，其他所有位都清零。

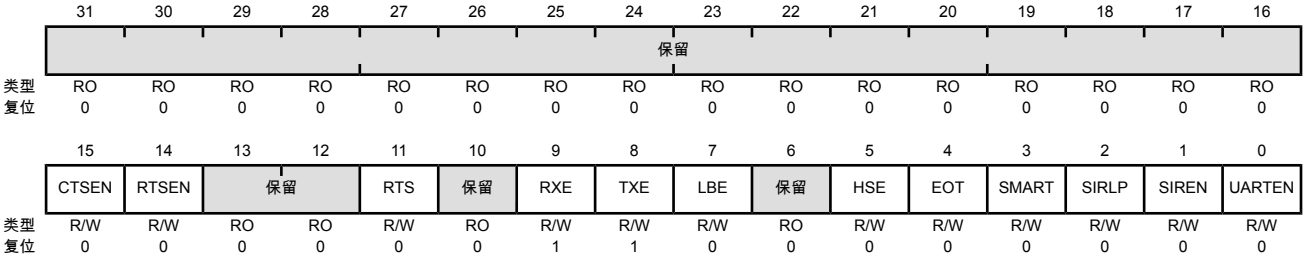
要使能 UART 模块，必须将 UARTEN 置位。假如软件准备对 UART 模块的配置进行更改，则必须先将 UARTEN 清零，之后才能写入更改的配置内容。假如在接收或发送期间禁用UART，则当前数据会话结束后，UART模块才会停止运行。

注意: 启用 UART 后不得修改 UARTCTL 寄存器，否则会产生无法预料的后果。推荐按照下面的步骤来更改 UARTCTL 寄存器的内容。

1. 禁用UART模块；
2. 等待当前数据会话（传输的字符）结束；
3. 通过将线控寄存器 UARTLCRH 的第 4 位 (FEN) 清 0，从而清空发送 FIFO。
4. 修改控制寄存器的内容；
5. 重新使能UART模块。

UART 控制寄存器 (UARTCTL)

UART0 基址: 0x4000.C000
UART1 基址: 0x4000.D000
UART2 基址: 0x4000.E000
UART3 基址: 0x4000.F000
UART4 基址: 0x4001.0000
UART5 基址: 0x4001.1000
UART6 基址: 0x4001.2000
UART7 基址: 0x4001.3000
偏移量 0x030
类型 R/W, 复位 0x0000.0300



位/域	名称	类型	复位	描述
31:16	保留	RO	0x0000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
15	CTSEN	R/W	0	使能允许发送 值 描述 0 禁用CTS硬件流控 1 启用 CTS 硬件流控。仅当 U1CTS 信号有效时才发送数据。

位/域	名称	类型	复位	描述
14	RTSEN	R/W	0	使能请求发送 值 描述 0 禁用RTS硬件流控 1 启用 RTS 硬件流控。仅当接收 FIFO 当前有条目时，才会（通过发出 U1RTS）请求数据。
13:12	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
11	RTS	R/W	0	请求发送 RTSEN 清零后，该位的状态通过 U1RTS 信号反映。如果对 RTSEN 进行置位，则进行写入操作时会忽略该位，进行读取操作时也应忽略该位。
10	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
9	RXE	R/W	1	启用 UART 接收 值 描述 0 禁用 UART 的接收部分。 1 启用 UART 的接收部分。 假如在接收的过程中禁用UART，那么仍然会完成当前数据会话后再停止UART模块。 注意: 要使能接收，还必须将 UARTEN 位置位。
8	TXE	R/W	1	启用 UART 发送 值 描述 0 禁用 UART 的发送部分。 1 启用 UART 的发送部分。 如果UART在发送中途被禁用，它会先发送完当前字符，然后再停止。 注意: 要使能发送，还必须将 UARTEN 位置位。
7	LBE	R/W	0	UART环回使能 值 描述 0 正常工作 1 UnTx 信号在内部连接到 UnRx 信号上。
6	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。

位/域	名称	类型	复位	描述
5	HSE	R/W	0	高速使能 值 描述 0 UART模块按系统时钟的16分频工作 1 UART模块按系统时钟的8分频工作 注意: 使用的系统时钟也取决于波特率除数配置（见819页和820页）。 在 ISO 7816 智能卡模式下（对 SMART 位置位），该位的状态不影响时钟产生。
4	EOT	R/W	0	传输结束 此标志位决定 UARTRIS 寄存器中 TXRIS 位的表现。 值 描述 0 当 UARTIFLS 寄存器定义的发送 FIFO 状况满足时，TXRIS 置位。 1 仅当所有发送的数据（包括停止位）从串行移位寄存器中发送出去后，TXRIS 才能置位。
3	SMART	R/W	0	ISO 7816智能卡支持 值 描述 0 正常工作 1 UART模块按智能卡模式工作 在 ISO 7816 模式下，应用程序必须确保将 UARTLCRH 寄存器配置为 8 位字长 (WLEN = 0x3) 和偶校验（PEN = 1、EPS = 1、SPS = 0）。 在此模式下，会忽略 UARTLCRH 中 STP2 位的值并强制采用 2 个停止位。请注意，在出现奇偶校验错误的情况下，UART 并不支持自动重新发送。如果在发送过程中检测到奇偶校验错误，会中止所有后续发送操作，必须由软件来处理受影响字节或报文的重新发送。
2	SIRLP	R/W	0	UART SIR低功耗模式 此标志位用于选择IrDA编码模式 值 描述 0 低电平位在发送时编码成宽度为3/16个位周期的脉冲 1 UART 在 SIR 低功耗模式下工作。无论选定的比特率是多少，低电平位将以 3 倍于 IrLPBaud16 输入信号周期的脉宽发送。 对该位进行置位可降低功率消耗，但可能会缩短发送距离。更多信息参见 818页。
1	SIREN	R/W	0	启用 UART SIR 值 描述 0 正常工作 1 使能IrDA SIR模块，UART将按照SIR协议进行发送和接收

位/域	名称	类型	复位	描述
0	UARTEN	R/W	0	启用 UART 值 描述 0 禁用 UART。 1 启用 UART。 假如在收发过程中禁用UART，那么仍然会完成当前数据会话后再停止UART模块。

寄存器 9: UART 中断 FIFO 深度选择寄存器（UARTIFLS），偏移量 0x034

UARTIFLS 寄存器称为中断 FIFO 深度选择寄存器。用户可通过本寄存器定义 UARTRIS 寄存器中 TXRIS 和 RXRIS 位触发时的 FIFO 深度。

中断是在FIFO深度从不满足触发条件到满足触发条件的跳变沿产生的。简单来说，中断是在FIFO深度越过触发门限时产生的。例如，若接收FIFO触发深度设为1/2，则当UART模块接收到第9个字符时才会触发中断。

复位后，TXIFLSEL 和 RXIFLSEL 位域均经过配置，以使 FIFO 以 1/2 的深度来触发中断。

UART 中断 FIFO 深度选择寄存器 (UARTIFLS)

UART0 基址: 0x4000.C000
UART1 基址: 0x4000.D000
UART2 基址: 0x4000.E000
UART3 基址: 0x4000.F000
UART4 基址: 0x4001.0000
UART5 基址: 0x4001.1000
UART6 基址: 0x4001.2000
UART7 基址: 0x4001.3000
偏移量 0x034
类型 R/W, 复位 0x0000.0012

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
域	保留															
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
域	保留										RXIFLSEL			TXIFLSEL		
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W	R/W	R/W
复位	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0

位/域	名称	类型	复位	描述
31:6	保留	RO	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
5:3	RXIFLSEL	R/W	0x2	UART接收中断FIFO深度选择 如下设置产生接收中断的触发深度： 值 描述 0x0 RX FIFO深度 ≥ 全满的1/8 0x1 RX FIFO ≥ ¼ 满 0x2 RX FIFO ≥ ½ 满（默认） 0x3 RX FIFO ≥ ¾ 满 0x4 RX FIFO ≥ ⅞ 满 0x5-0x7 保留

位/域	名称	类型	复位	描述
2:0	TXIFLSEL	R/W	0x2	UART 发送中断 FIFO 深度选择 发送中断的触发深度设置如下： 值 描述 0x0 TX FIFO ≤ ⅞ 空 0x1 TX FIFO ≤ ¾ 空 0x2 TX FIFO ≤ ½ 空（默认） 0x3 TX FIFO ≤ ¼ 空 0x4 TX FIFO ≤ ⅛ 空 0x5-0x7 保留 注意: 如果对 UARTCTL（见823页）中的 EOT 位进行置位，只要 FIFO 完全变空且包括停止位在内的所有数据均从发送串行移位器发出，就会产生发送中断。在这种情况下，会忽略 TXIFLSEL 的设置。

寄存器 10: UART 中断屏蔽寄存器（UARTIM），偏移量 0x038

UARTIM 寄存器是中断掩码置位/清零寄存器。

读本寄存器时，返回各中断的当前掩码状态。对某个位进行置位时，可将相应的原始中断信号传递到中断控制器。将某个位清零时，可阻止该原始中断信号传递到中断控制器。

UART 中断屏蔽寄存器 (UARTIM)

UART0 基址: 0x4000.C000
UART1 基址: 0x4000.D000
UART2 基址: 0x4000.E000
UART3 基址: 0x4000.F000
UART4 基址: 0x4001.0000
UART5 基址: 0x4001.1000
UART6 基址: 0x4001.2000
UART7 基址: 0x4001.3000
偏移量 0x038
类型 R/W, 复位 0x0000.0000

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
域	保留															
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
域	保留			9BITIM	保留	OEIM	BEIM	PEIM	FEIM	RTIM	TXIM	RXIM	保留		CTSIM	保留
类型	RO	RO	RO	R/W	RO	R/W	R/W	R/W	R/W	R/W	R/W	R/W	RO	RO	R/W	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位/域	名称	类型	复位	描述
31:13	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
12	9BITIM	R/W	0	9 位模式中断屏蔽 值 描述 0 9BITRIS 中断被抑制，不会发送到中断控制器。 1 当 UARTRIS 寄存器的 9BITRIS 位置位时，会向中断控制器发送中断。
11	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
10	OEIM	R/W	0	UART溢出错误中断掩码 值 描述 0 OERIS 中断被抑制，不会发送到中断控制器。 1 当 UARTRIS 寄存器的 OERIS 位置位时，会向中断控制器发送中断。
9	BEIM	R/W	0	UART线中止错误中断掩码 值 描述 0 BERIS 中断被抑制，不会发送到中断控制器。 1 当 UARTRIS 寄存器的 BERIS 位置位时，会向中断控制器发送中断。

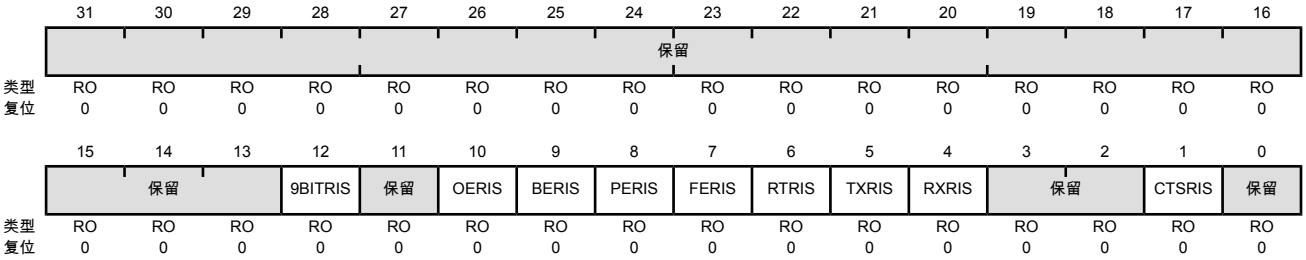
位/域	名称	类型	复位	描述
8	PEIM	R/W	0	UART奇偶校验错误中断掩码 值 描述 0 PERIS 中断被抑制，不会发送到中断控制器。 1 当 UARTRIS 寄存器的 PERIS 位置位时，会向中断控制器发送中断。
7	FEIM	R/W	0	UART帧错误中断掩码 值 描述 0 FERIS 中断被抑制，不会发送到中断控制器。 1 当 UARTRIS 寄存器的 FERIS 位置位时，会向中断控制器发送中断。
6	RTIM	R/W	0	UART接收超时中断掩码 值 描述 0 RTRIS 中断被抑制，不会发送到中断控制器。 1 当 UARTRIS 寄存器的 RTRIS 位置位时，会向中断控制器发送中断。
5	TXIM	R/W	0	UART发送中断掩码 值 描述 0 TXRIS 中断被抑制，不会发送到中断控制器。 1 当 UARTRIS 寄存器的 TXRIS 位置位时，会向中断控制器发送中断。
4	RXIM	R/W	0	UART接收中断掩码 值 描述 0 RXRIS 中断被抑制，不会发送到中断控制器。 1 当 UARTRIS 寄存器的 RXRIS 位置位时，会向中断控制器发送中断。
3:2	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
1	CTSIM	R/W	0	UART允许发送调制解调器中断掩码 值 描述 0 CTSRIS 中断被抑制，不会发送到中断控制器。 1 当 CTSRIS 寄存器的 DCDRIS 位置位时，会向中断控制器发送中断。 该位仅对 UART1 有效，对于 UART0 和 UART2，该位是保留的。
0	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。

寄存器 11: UART 原始中断状态寄存器（UARTRIS），偏移量 0x03C

UARTRIS 寄存器称为原始中断状态寄存器。当读取本寄存器时，返回当前各个中断的原始状态。对本寄存器写操作无效。

UART 原始中断状态寄存器 (UARTRIS)

UART0 基址: 0x4000.C000
UART1 基址: 0x4000.D000
UART2 基址: 0x4000.E000
UART3 基址: 0x4000.F000
UART4 基址: 0x4001.0000
UART5 基址: 0x4001.1000
UART6 基址: 0x4001.2000
UART7 基址: 0x4001.3000
偏移量 0x03C
类型 RO, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:13	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
12	9BITRIS	RO	0	9 位模式原始中断状态 值 描述 0 无中断 1 出现了接收地址匹配。 向 UARTICR 寄存器的 9BITIC 位写入 1 即可将该位清零。
11	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
10	OERIS	RO	0	UART 溢出错误原始中断状态 值 描述 0 无中断 1 发生了溢出错误。 向 UARTICR 寄存器的 OEIC 位写入 1 即可将该位清零。
9	BERIS	RO	0	UART 中止错误原始中断状态 值 描述 0 无中断 1 发生了中止错误。 向 UARTICR 寄存器的 BEIC 位写入 1 即可将该位清零。

位/域	名称	类型	复位	描述
8	PERIS	RO	0	UART 奇偶校验错误原始中断状态 值 描述 0 无中断 1 发生了奇偶校验错误。 向 UARTICR 寄存器的 PEIC 位写入 1 即可将该位清零。
7	FERIS	RO	0	UART 帧错误原始中断状态 值 描述 0 无中断 1 发生了帧错误。 向 UARTICR 寄存器的 FEIC 位写入 1 即可将该位清零。
6	RTRIS	RO	0	UART 接收超时原始中断状态 值 描述 0 无中断 1 发生接收超时。 向 UARTICR 寄存器的 RTIC 位写入 1 即可将本位清零。
5	TXRIS	RO	0	UART 发送原始中断状态 值 描述 0 无中断 1 若 UARTCTL 寄存器的 EOT 位清零，则说明发送 FIFO 深度已经越过了 UARTIFLS 寄存器所定义的门限。 若 EOT 位置位，则说明所有待发送数据和标志的最后一位已经从串行移位寄存器中发出。 FIFO 启用时，通过向 UARTICR 寄存器中的 TXIC 位写 1 或向发送 FIFO 写入数据直至其高于触发深度，以将该位清零，或在 FIFO 禁用时写入一个字节将该位清零。
4	RXRIS	RO	0	UART 接收原始中断状态 值 描述 0 无中断 1 接收 FIFO 深度已经越过了 UARTIFLS 寄存器所定义的门限。 FIFO 启用时，通过向 UARTICR 寄存器中的 RXIC 位写 1 或从接收 FIFO 读取数据直至其低于触发深度，以将该位清零；FIFO 禁用时，通过读取一个字节将该位清零。
3:2	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。

位/域	名称	类型	复位	描述
1	CTSRIS	RO	0	UART 允许发送调制解调器原始中断状态 值 描述 0 无中断 1 用于软件流控的允许发送标志。 向 UARTICR 寄存器的 CTSIC 位写入 1 即可将该位清零。 该位仅对 UART1 有效，对于 UART0 和 UART2，该位是保留的。
0	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。

寄存器 12: UART 屏蔽中断状态寄存器（UARTMIS），偏移量 0x040

UARTMIS 寄存器称为屏蔽中断状态寄存器。读取时，该寄存器给出相应中断的当前屏蔽状态值。对本寄存器写操作无效。

UART 屏蔽中断状态寄存器 (UARTMIS)

UART0 基址: 0x4000.C000
UART1 基址: 0x4000.D000
UART2 基址: 0x4000.E000
UART3 基址: 0x4000.F000
UART4 基址: 0x4001.0000
UART5 基址: 0x4001.1000
UART6 基址: 0x4001.2000
UART7 基址: 0x4001.3000
偏移量 0x040
类型 RO, 复位 0x0000.0000

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
域	保留															
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
域	保留			9BITMIS	保留	OEMIS	BEMIS	PEMIS	FEMIS	RTMIS	TXMIS	RXMIS	保留		CTSMIS	保留
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位/域	名称	类型	复位	描述
31:13	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
12	9BITMIS	RO	0	9 位模式屏蔽中断状态 值 描述 0 中断没有发生或者被屏蔽。 1 由于接收地址匹配，因此发出未屏蔽的中断信号。 向 UARTICR 寄存器的 9BITIC 位写入 1 即可将该位清零。
11	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
10	OEMIS	RO	0	UART溢出错误掩码后中断状态 值 描述 0 中断没有发生或者被屏蔽。 1 已产生溢出错误，并且此中断未被屏蔽 向 UARTICR 寄存器的 OEIC 位写入 1 即可将该位清零。
9	BEMIS	RO	0	UART 中止错误屏蔽中断状态 值 描述 0 中断没有发生或者被屏蔽。 1 由于产生中止错误，因此发出未屏蔽的中断信号。 向 UARTICR 寄存器的 BEIC 位写入 1 即可将该位清零。

位/域	名称	类型	复位	描述
8	PEMIS	RO	0	UART 奇偶校验错误屏蔽中断状态 值 描述 0 中断没有发生或者被屏蔽。 1 由于产生奇偶校验错误，因此发出未屏蔽的中断信号。 向 UARTICR 寄存器的 PEIC 位写入 1 即可将该位清零。
7	FEMIS	RO	0	UART 帧错误屏蔽中断状态 值 描述 0 中断没有发生或者被屏蔽。 1 由于产生帧错误，因此发出未屏蔽的中断信号。 向 UARTICR 寄存器的 FEIC 位写入 1 即可将该位清零。
6	RTMIS	RO	0	UART 接收超时屏蔽中断状态 值 描述 0 中断没有发生或者被屏蔽。 1 由于接收超时，因此发出未屏蔽的中断信号。 向 UARTICR 寄存器的 RTIC 位写入 1 即可将本位清零。
5	TXMIS	RO	0	UART 发送屏蔽中断状态 值 描述 0 中断没有发生或者被屏蔽。 1 若发送 FIFO 深度已经越过了定义的门限（EOT 位清零），或所有待发送数据已经从串行移位寄存器中发出（EOT 位置位），则发出未被屏蔽的中断。 FIFO 启用时，通过向 UARTICR 寄存器中的 TXIC 位写 1 或向发送 FIFO 写入数据直至其高于触发深度，以将该位清零，或在 FIFO 禁用时写入一个字节将该位清零。
4	RXMIS	RO	0	UART 接收屏蔽中断状态 值 描述 0 中断没有发生或者被屏蔽。 1 接收FIFO深度已经越过了UARTIFLS寄存器所定义的门限，并且此中断未被屏蔽 FIFO 启用时，通过向 UARTICR 寄存器中的 RXIC 位写 1 或从接收 FIFO 读取数据直至其低于触发深度，以将该位清零；FIFO 禁用时，通过读取一个字节将该位清零。
3:2	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。

位/域	名称	类型	复位	描述
1	CTSMIS	RO	0	UART 允许发送调制解调器屏蔽中断状态 值 描述 0 中断没有发生或者被屏蔽。 1 由于允许发送，因此发出未屏蔽的中断信号。 向 UARTICR 寄存器的 CTSIC 位写入 1 即可将该位清零。 该位仅对 UART1 有效，对于 UART0 和 UART2，该位是保留的。
0	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。

寄存器 13: UART 中断清除寄存器（UARTICR），偏移量 0x044

UARTICR 寄存器称为中断清除寄存器。写入 1 时，相应的中断（包括原始中断，如果启用了屏蔽中断，则还包括屏蔽中断）被清除。写入 0 不起任何作用。

UART 中断清除寄存器 (UARTICR)

UART0 基址: 0x4000.C000
UART1 基址: 0x4000.D000
UART2 基址: 0x4000.E000
UART3 基址: 0x4000.F000
UART4 基址: 0x4001.0000
UART5 基址: 0x4001.1000
UART6 基址: 0x4001.2000
UART7 基址: 0x4001.3000
偏移量 0x044
类型 W1C, 复位 0x0000.0000

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
域	保留															
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
域	保留			9BITIC	保留	OEIC	BEIC	PEIC	FEIC	RTIC	TXIC	RXIC	保留		CTSMIC	保留
类型	RO	RO	RO	R/W	RO	W1C	W1C	W1C	W1C	W1C	W1C	W1C	RO	RO	W1C	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位/域	名称	类型	复位	描述
31:13	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
12	9BITIC	R/W	0	9 位模式中断清除 向该位写入 1，即可将 UARTRIS 寄存器的 9BITRIS 位以及 UARTMIS 寄存器的 9BITMIS 位清零。
11	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
10	OEIC	W1C	0	溢出错误中断清除 向该位写入 1，即可将 UARTRIS 寄存器的 OERIS 位以及 UARTMIS 寄存器的 OEMIS 位清零。
9	BEIC	W1C	0	中止错误中断清除 向该位写入 1，即可将 UARTRIS 寄存器的 BERIS 位以及 UARTMIS 寄存器的 BEMIS 位清零。
8	PEIC	W1C	0	奇偶校验错误中断清除 向该位写入 1，即可将 UARTRIS 寄存器的 PERIS 位以及 UARTMIS 寄存器的 PEMIS 位清零。
7	FEIC	W1C	0	帧错误中断清除 向该位写入 1，即可将 UARTRIS 寄存器的 FERIS 位以及 UARTMIS 寄存器的 FEMIS 位清零。
6	RTIC	W1C	0	接收超时中断清除 向该位写入 1，即可将 UARTRIS 寄存器的 RTRIS 位以及 UARTMIS 寄存器的 RTMIS 位清零。

位/域	名称	类型	复位	描述
5	TXIC	W1C	0	发送中断清除 向该位写入 1，即可将 UARTRIS 寄存器的 TXRIS 位以及 UARTMIS 寄存器的 TXMIS 位清零。
4	RXIC	W1C	0	接收中断清除 向该位写入 1，即可将 UARTRIS 寄存器的 RXRIS 位以及 UARTMIS 寄存器的 RXMIS 位清零。
3:2	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
1	CTSMIC	W1C	0	UART 允许发送调制解调器中断清除 向该位写入 1，即可将 UARTRIS 寄存器的 CTSRIS 位以及 UARTMIS 寄存器的 CTSMIS 位清零。 该位仅对 UART1 有效，对于 UART0 和 UART2，该位是保留的。
0	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。

寄存器 14: UART DMA 控制寄存器（UARTDMACTL），偏移量 0x048

UARTDMACTL 寄存器称为 DMA 控制寄存器。

UART DMA 控制寄存器 (UARTDMACTL)

UART0 基址: 0x4000.C000
UART1 基址: 0x4000.D000
UART2 基址: 0x4000.E000
UART3 基址: 0x4000.F000
UART4 基址: 0x4001.0000
UART5 基址: 0x4001.1000
UART6 基址: 0x4001.2000
UART7 基址: 0x4001.3000
偏移量 0x048
类型 R/W, 复位 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	保留															
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	保留													DMAERR	TXDMAE	RXDMAE
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

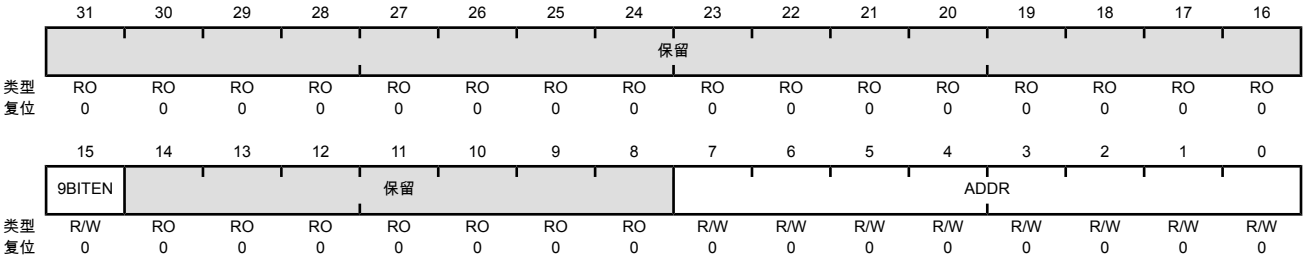
位/域	名称	类型	复位	描述
31:3	保留	RO	0x00000.000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
2	DMAERR	R/W	0	出现错误时的 DMA 值 描述 0 接收错误不影响µDMA请求 1 当产生接收错误时，自动禁用µDMA请求
1	TXDMAE	R/W	0	发送DMA使能 值 描述 0 禁用发送FIFO的µDMA通道 1 使能发送FIFO的µDMA通道
0	RXDMAE	R/W	0	接收 DMA 启用 值 描述 0 禁用接收 FIFO 的 µDMA 通道 1 启用接收 FIFO 的 µDMA 通道

寄存器 15: UART 9 位模式自身地址寄存器（UART9BITADDR），偏移量 0x0A4

UART9BITADDR 寄存器用于在 9 位地址屏蔽 (UART9BITAMASK) 被设置为 0xFF 时，写入应与接收字节匹配的具体地址。该寄存器与UART9BITAMASK配合使用，从而与接收到的地址字节匹配。

UART 9 位模式自身地址寄存器 (UART9BITADDR)

UART0 基址: 0x4000.C000
UART1 基址: 0x4000.D000
UART2 基址: 0x4000.E000
UART3 基址: 0x4000.F000
UART4 基址: 0x4001.0000
UART5 基址: 0x4001.1000
UART6 基址: 0x4001.2000
UART7 基址: 0x4001.3000
偏移量 0x0A4
类型 R/W, 复位 0x0000.0000



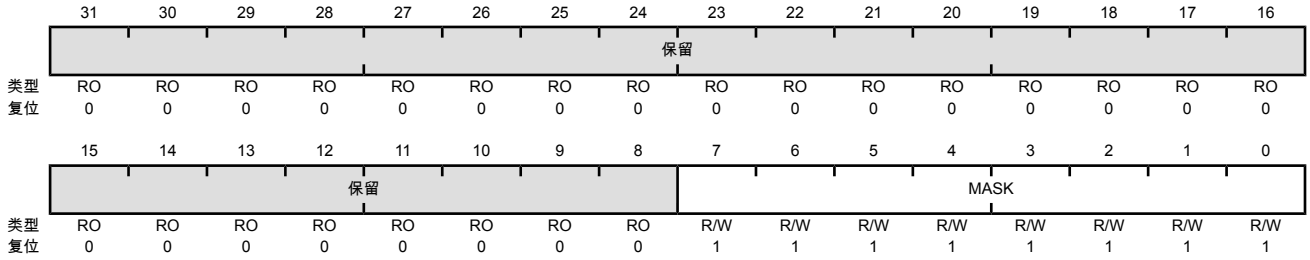
位/域	名称	类型	复位	描述
31:16	保留	RO	0x0000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
15	9BITEN	R/W	0	启用 9 位模式 值 描述 0 禁用 9 位模式。 1 启用 9 位模式。
14:8	保留	RO	0x0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
7:0	ADDR	R/W	0x00	9 位模式的自身地址 此位域中包含当 UART9BITAMASK 被设置为 0xFF 时，应与接收字节相匹配的地址。

寄存器 16: UART 9 位模式自身地址屏蔽寄存器（UART9BITAMASK），偏移量 0x0A8

UART9BITAMASK 寄存器用于为 9 位模式启用地址屏蔽。屏蔽地址位，从而创建一组将与接收到的地址字节相匹配的地址。

UART 9 位模式自身地址屏蔽寄存器 (UART9BITAMASK)

UART0 基址: 0x4000.C000
UART1 基址: 0x4000.D000
UART2 基址: 0x4000.E000
UART3 基址: 0x4000.F000
UART4 基址: 0x4001.0000
UART5 基址: 0x4001.1000
UART6 基址: 0x4001.2000
UART7 基址: 0x4001.3000
偏移量 0x0A8
类型 R/W, 复位 0x0000.00FF



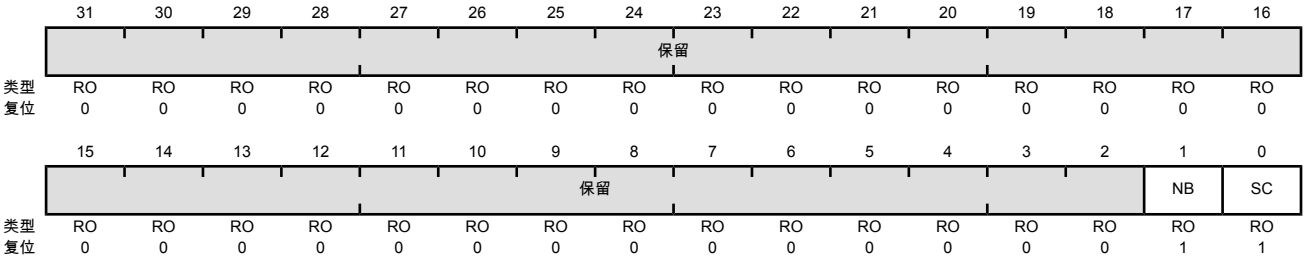
位/域	名称	类型	复位	描述
31:8	保留	RO	0x00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
7:0	MASK	R/W	0xFF	9 位模式的自身地址屏蔽 此位域包含地址屏蔽，可用于创建一组应匹配的地址。

寄存器 17: UART 外设属性寄存器（UARTPP），偏移量 0xFC0

UARTPP 寄存器提供有关 UART 模块属性的信息。

UART 外设属性寄存器 (UARTPP)

UART0 基址: 0x4000.C000
UART1 基址: 0x4000.D000
UART2 基址: 0x4000.E000
UART3 基址: 0x4000.F000
UART4 基址: 0x4001.0000
UART5 基址: 0x4001.1000
UART6 基址: 0x4001.2000
UART7 基址: 0x4001.3000
偏移量 0xFC0
类型 RO, 复位 0x0000.0003



位/域	名称	类型	复位	描述
31:2	保留	RO	0x0000.000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
1	NB	RO	0x1	9 位模式支持 值 描述 0 UART 模块不支持 9 位数据的发送，因而不支持 RS-485。 1 UART 模块支持 9 位数据的发送，从而支持 RS-485。
0	SC	RO	0x1	智能卡支持 值 描述 0 UART 模块不支持智能卡。 1 UART 模块支持智能卡。

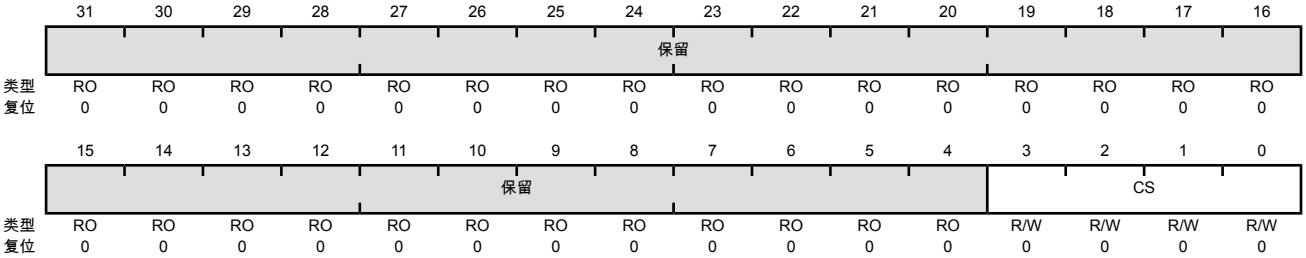
寄存器 18: UART 时钟配置寄存器（UARTCC），偏移量 0xFC8

UARTCC 寄存器控制 UART 模块的波特时钟源。详见 “通信时钟源”一节（195页）。

注意: 如果将 PIOSC 用作 UART 波特时钟，则系统时钟频率在运行模式下必须至少为 9 MHz。

UART 时钟配置寄存器 (UARTCC)

UART0 基址: 0x4000.C000
UART1 基址: 0x4000.D000
UART2 基址: 0x4000.E000
UART3 基址: 0x4000.F000
UART4 基址: 0x4001.0000
UART5 基址: 0x4001.1000
UART6 基址: 0x4001.2000
UART7 基址: 0x4001.3000
偏移量 0xFC8
类型 R/W, 复位 0x0000.0000



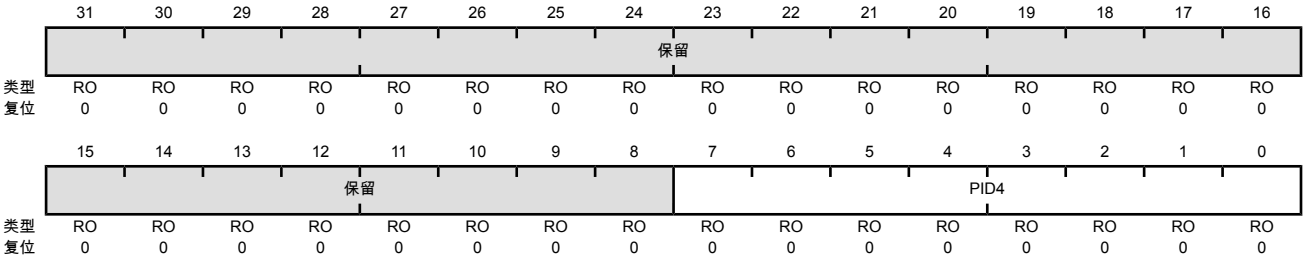
位/域	名称	类型	复位	描述
31:4	保留	RO	0x0000.000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
3:0	CS	R/W	0	UART 波特时钟源 下面的表格指定了产生 UART 波特时钟的源： 值 描述 0x0 系统时钟（基于时钟源和分频因子） 0x1-0x4 保留 0x5 PIOSC 0x5-0xF 保留

寄存器 19: UART 外设标识寄存器 4（UARTPeriphID4），偏移量 0xFD0

UARTPeriphIDn 寄存器均为硬编码的只读寄存器，寄存器的位域决定复位值。

UART 外设标识寄存器 4 (UARTPeriphID4)

UART0 基址: 0x4000.C000
UART1 基址: 0x4000.D000
UART2 基址: 0x4000.E000
UART3 基址: 0x4000.F000
UART4 基址: 0x4001.0000
UART5 基址: 0x4001.1000
UART6 基址: 0x4001.2000
UART7 基址: 0x4001.3000
偏移量 0xFD0
类型 RO, 复位 0x0000.0000



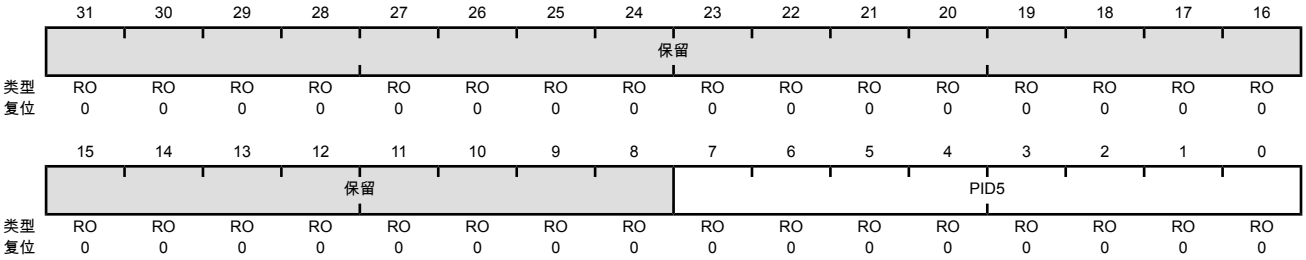
位/域	名称	类型	复位	描述
31:8	保留	RO	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
7:0	PID4	RO	0x00	UART 外设标识寄存器 [7:0] 可被软件用来标识该外设的存在与否。

寄存器 20: UART 外设标识寄存器 5（UARTPeriphID5），偏移量 0xFD4

UARTPeriphIDn 寄存器均为硬编码的只读寄存器，寄存器的位域决定复位值。

UART 外设标识寄存器 5 (UARTPeriphID5)

UART0 基址: 0x4000.C000
UART1 基址: 0x4000.D000
UART2 基址: 0x4000.E000
UART3 基址: 0x4000.F000
UART4 基址: 0x4001.0000
UART5 基址: 0x4001.1000
UART6 基址: 0x4001.2000
UART7 基址: 0x4001.3000
偏移量 0xFD4
类型 RO, 复位 0x0000.0000



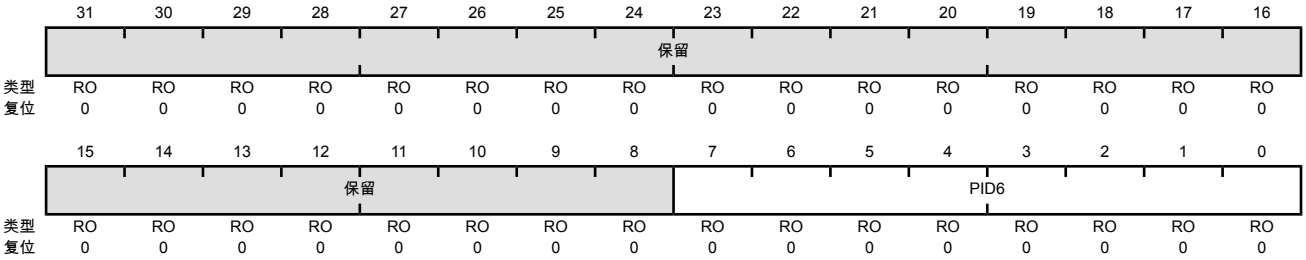
位/域	名称	类型	复位	描述
31:8	保留	RO	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
7:0	PID5	RO	0x00	UART 外设标识寄存器 [15:8] 可被软件用来标识该外设的存在与否。

寄存器 21: UART 外设标识寄存器 6（UARTPeriphID6），偏移量 0xFD8

UARTPeriphIDn 寄存器均为硬编码的只读寄存器，寄存器的位域决定复位值。

UART 外设标识寄存器 6 (UARTPeriphID6)

UART0 基址: 0x4000.C000
UART1 基址: 0x4000.D000
UART2 基址: 0x4000.E000
UART3 基址: 0x4000.F000
UART4 基址: 0x4001.0000
UART5 基址: 0x4001.1000
UART6 基址: 0x4001.2000
UART7 基址: 0x4001.3000
偏移量 0xFD8
类型 RO, 复位 0x0000.0000



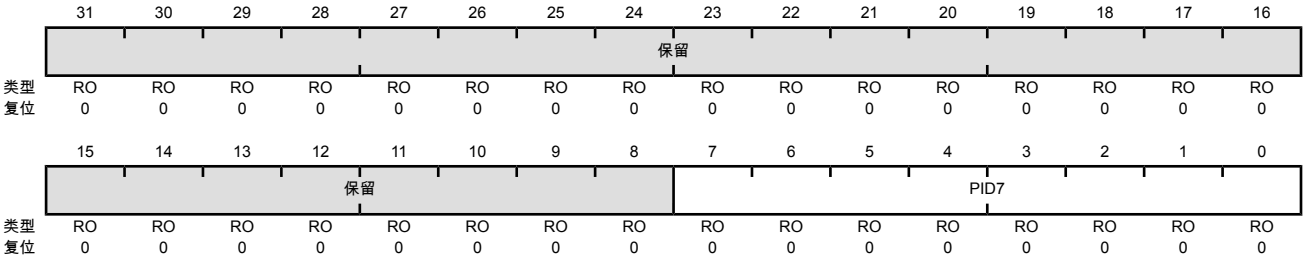
位/域	名称	类型	复位	描述
31:8	保留	RO	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
7:0	PID6	RO	0x00	UART 外设标识寄存器 [23:16] 可被软件用来标识该外设的存在与否。

寄存器 22: UART 外设标识寄存器 7（UARTPeriphID7），偏移量 0xFDC

UARTPeriphIDn 寄存器均为硬编码的只读寄存器，寄存器的位域决定复位值。

UART 外设标识寄存器 7 (UARTPeriphID7)

UART0 基址: 0x4000.C000
UART1 基址: 0x4000.D000
UART2 基址: 0x4000.E000
UART3 基址: 0x4000.F000
UART4 基址: 0x4001.0000
UART5 基址: 0x4001.1000
UART6 基址: 0x4001.2000
UART7 基址: 0x4001.3000
偏移量 0xFDC
类型 RO, 复位 0x0000.0000



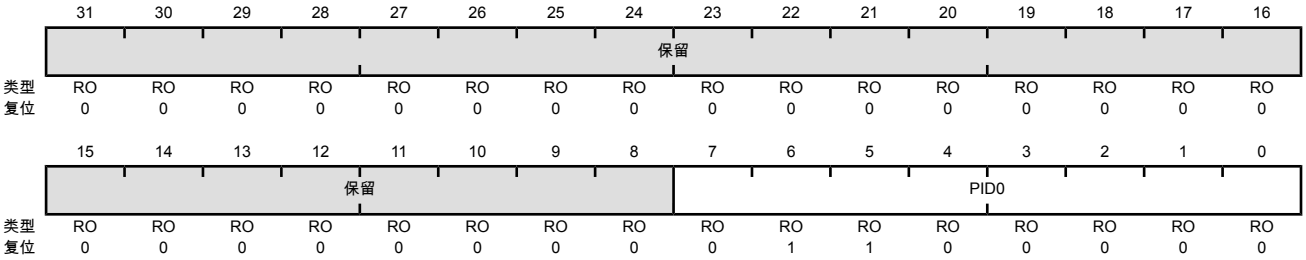
位/域	名称	类型	复位	描述
31:8	保留	RO	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
7:0	PID7	RO	0x00	UART 外设标识寄存器 [31:24] 可被软件用来标识该外设的存在与否。

寄存器 23: UART 外设标识寄存器 0（UARTPeriphID0），偏移量 0xFE0

UARTPeriphIDn 寄存器均为硬编码的只读寄存器，寄存器的位域决定复位值。

UART 外设标识寄存器 0 (UARTPeriphID0)

UART0 基址: 0x4000.C000
UART1 基址: 0x4000.D000
UART2 基址: 0x4000.E000
UART3 基址: 0x4000.F000
UART4 基址: 0x4001.0000
UART5 基址: 0x4001.1000
UART6 基址: 0x4001.2000
UART7 基址: 0x4001.3000
偏移量 0xFE0
类型 RO, 复位 0x0000.0060



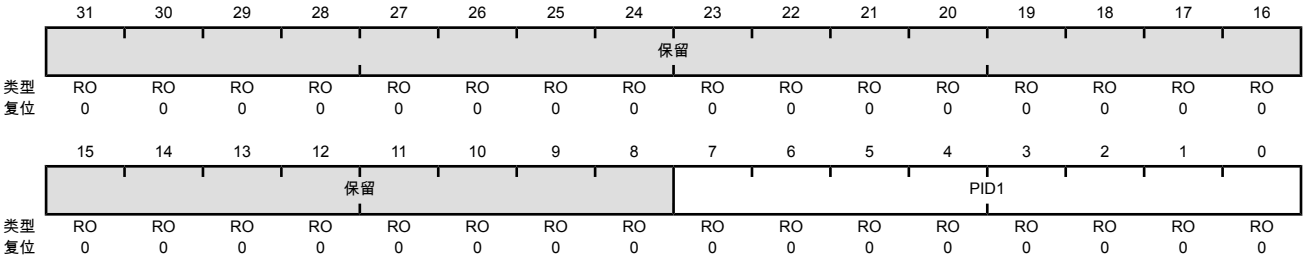
位/域	名称	类型	复位	描述
31:8	保留	RO	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
7:0	PID0	RO	0x60	UART 外设标识寄存器 [7:0] 可被软件用来标识该外设的存在与否。

寄存器 24: UART 外设标识寄存器 1（UARTPeriphID1），偏移量 0xFE4

UARTPeriphIDn 寄存器均为硬编码的只读寄存器，寄存器的位域决定复位值。

UART 外设标识寄存器 1 (UARTPeriphID1)

UART0 基址: 0x4000.C000
UART1 基址: 0x4000.D000
UART2 基址: 0x4000.E000
UART3 基址: 0x4000.F000
UART4 基址: 0x4001.0000
UART5 基址: 0x4001.1000
UART6 基址: 0x4001.2000
UART7 基址: 0x4001.3000
偏移量 0xFE4
类型 RO, 复位 0x0000.0000



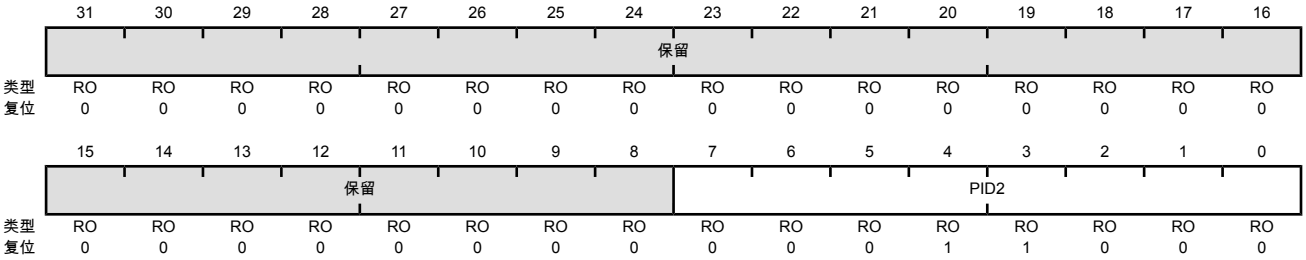
位/域	名称	类型	复位	描述
31:8	保留	RO	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
7:0	PID1	RO	0x00	UART 外设标识寄存器 [15:8] 可被软件用来标识该外设的存在与否。

寄存器 25: UART 外设标识寄存器 2（UARTPeriphID2），偏移量 0xFE8

UARTPeriphIDn 寄存器均为硬编码的只读寄存器，寄存器的位域决定复位值。

UART 外设标识寄存器 2 (UARTPeriphID2)

UART0 基址: 0x4000.C000
UART1 基址: 0x4000.D000
UART2 基址: 0x4000.E000
UART3 基址: 0x4000.F000
UART4 基址: 0x4001.0000
UART5 基址: 0x4001.1000
UART6 基址: 0x4001.2000
UART7 基址: 0x4001.3000
偏移量 0xFE8
类型 RO, 复位 0x0000.0018



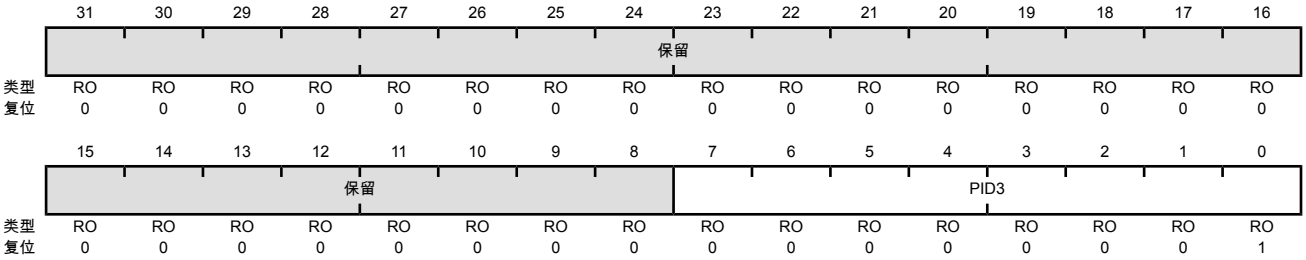
位/域	名称	类型	复位	描述
31:8	保留	RO	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
7:0	PID2	RO	0x18	UART 外设标识寄存器 [23:16] 可被软件用来标识该外设的存在与否。

寄存器 26: UART 外设标识寄存器 3（UARTPeriphID3），偏移量 0xFEC

UARTPeriphIDn 寄存器均为硬编码的只读寄存器，寄存器的位域决定复位值。

UART 外设标识寄存器 3 (UARTPeriphID3)

UART0 基址: 0x4000.C000
UART1 基址: 0x4000.D000
UART2 基址: 0x4000.E000
UART3 基址: 0x4000.F000
UART4 基址: 0x4001.0000
UART5 基址: 0x4001.1000
UART6 基址: 0x4001.2000
UART7 基址: 0x4001.3000
偏移量 0xFEC
类型 RO, 复位 0x0000.0001



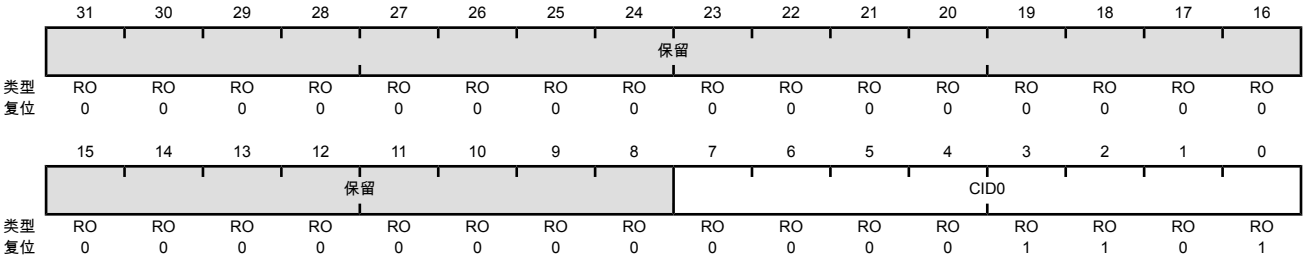
位/域	名称	类型	复位	描述
31:8	保留	RO	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
7:0	PID3	RO	0x01	UART 外设标识寄存器 [31:24] 可被软件用来标识该外设的存在与否。

寄存器 27: UART PrimeCell 标识寄存器 0（UARTPCellID0），偏移量 0xFF0

UARTPCellIDn 寄存器均为硬编码的只读寄存器，寄存器的位域决定复位值。

UART PrimeCell 标识寄存器 0 (UARTPCellID0)

UART0 基址: 0x4000.C000
UART1 基址: 0x4000.D000
UART2 基址: 0x4000.E000
UART3 基址: 0x4000.F000
UART4 基址: 0x4001.0000
UART5 基址: 0x4001.1000
UART6 基址: 0x4001.2000
UART7 基址: 0x4001.3000
偏移量 0xFF0
类型 RO, 复位 0x0000.000D



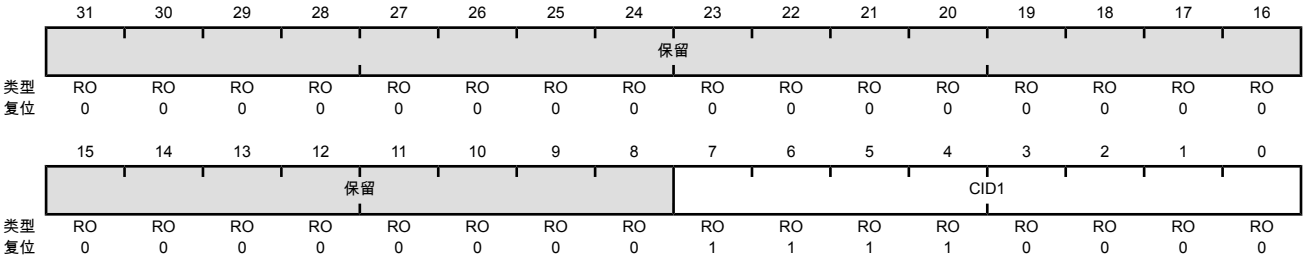
位/域	名称	类型	复位	描述
31:8	保留	RO	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
7:0	CID0	RO	0x0D	UART PrimeCell 标志寄存器 [7:0] 为软件提供一个标准的交叉外设识别系统。

寄存器 28: UART PrimeCell 标识寄存器 1（UARTPCellID1），偏移量 0xFF4

UARTPCellIDn 寄存器均为硬编码的只读寄存器，寄存器的位域决定复位值。

UART PrimeCell 标识寄存器 1 (UARTPCellID1)

UART0 基址: 0x4000.C000
UART1 基址: 0x4000.D000
UART2 基址: 0x4000.E000
UART3 基址: 0x4000.F000
UART4 基址: 0x4001.0000
UART5 基址: 0x4001.1000
UART6 基址: 0x4001.2000
UART7 基址: 0x4001.3000
偏移量 0xFF4
类型 RO, 复位 0x0000.00F0



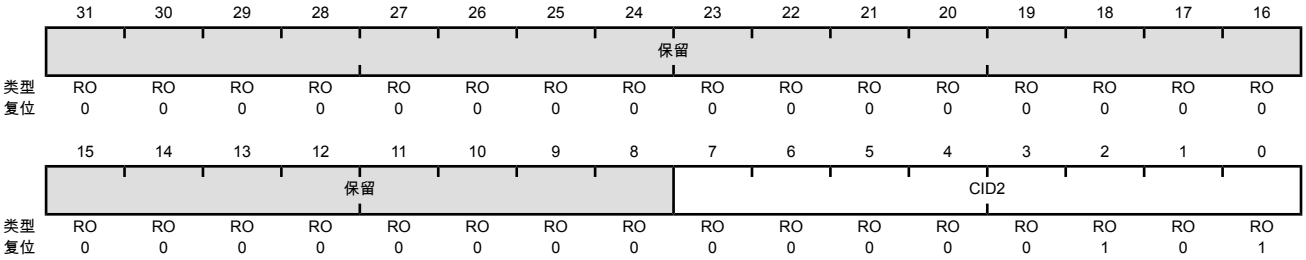
位/域	名称	类型	复位	描述
31:8	保留	RO	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
7:0	CID1	RO	0xF0	UART PrimeCell 标志寄存器 [15:8] 为软件提供一个标准的交叉外设识别系统。

寄存器 29: UART PrimeCell 标识寄存器 2（UARTPCellID2），偏移量 0xFF8

UARTPCellIDn 寄存器均为硬编码的只读寄存器，寄存器的位域决定复位值。

UART PrimeCell 标识寄存器 2 (UARTPCellID2)

UART0 基址: 0x4000.C000
UART1 基址: 0x4000.D000
UART2 基址: 0x4000.E000
UART3 基址: 0x4000.F000
UART4 基址: 0x4001.0000
UART5 基址: 0x4001.1000
UART6 基址: 0x4001.2000
UART7 基址: 0x4001.3000
偏移量 0xFF8
类型 RO, 复位 0x0000.0005



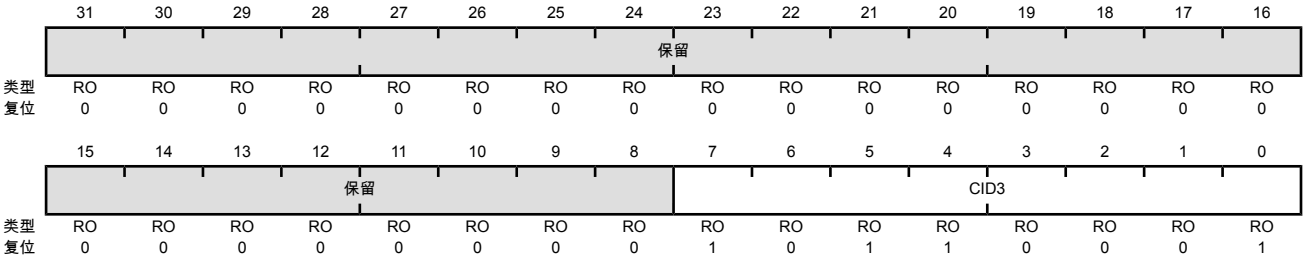
位/域	名称	类型	复位	描述
31:8	保留	RO	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
7:0	CID2	RO	0x05	UART PrimeCell 标志寄存器 [23:16] 为软件提供一个标准的交叉外设识别系统。

寄存器 30: UART PrimeCell 标识寄存器 3（UARTPCellID3），偏移量 0xFFC

UARTPCellIDn 寄存器均为硬编码的只读寄存器，寄存器的位域决定复位值。

UART PrimeCell 标识寄存器 3 (UARTPCellID3)

UART0 基址: 0x4000.C000
UART1 基址: 0x4000.D000
UART2 基址: 0x4000.E000
UART3 基址: 0x4000.F000
UART4 基址: 0x4001.0000
UART5 基址: 0x4001.1000
UART6 基址: 0x4001.2000
UART7 基址: 0x4001.3000
偏移量 0xFFC
类型 RO, 复位 0x0000.00B1



位/域	名称	类型	复位	描述
31:8	保留	RO	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
7:0	CID3	RO	0xB1	UART PrimeCell 标志寄存器 [31:24] 为软件提供一个标准的交叉外设识别系统。

15 同步串行接口（SSI）

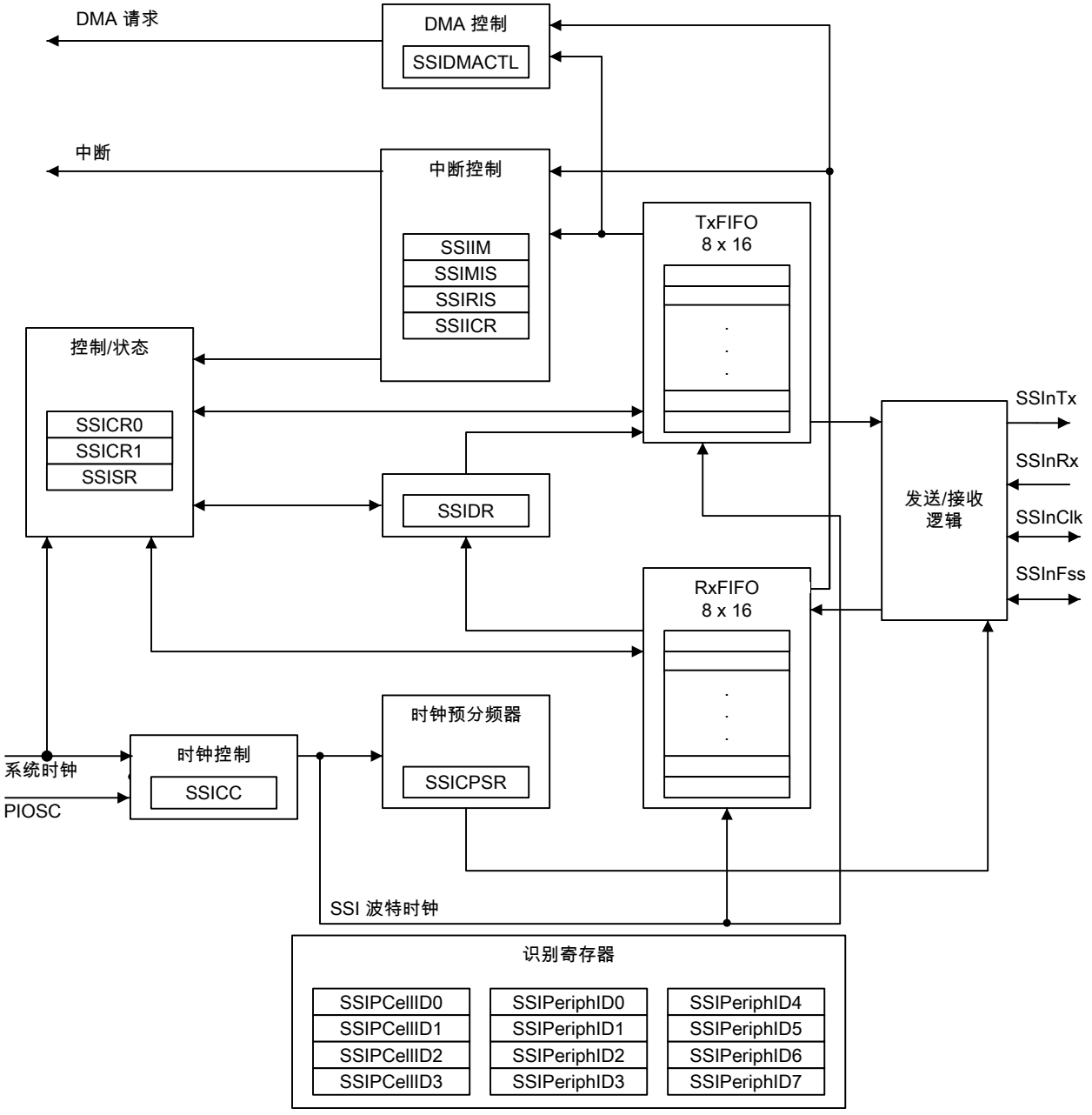
TM4C1231H6PM 微控制器包含 4 个同步串行接口 (SSI) 模块。每个 SSI 模块都能以主机或从机方式与外设器件进行同步串行通信，可以支持飞思卡尔 (Freescale) SPI、MICROWIRE 或者 Texas Instruments 的同步串行接口 (SSI)。

TM4C1231H6PM SSI 模块具有以下特性：

- 提供可编程控制的接口，可与飞思卡尔的 SPI 接口、MICROWIRE 或者 Texas Instruments 同步串行接口相连
- 主机或从机工作方式；
- 可编程的时钟位速率以及预分频器；
- 相互独立的发送 FIFO 和接收 FIFO，二者均为 16 位宽、8 个单元深；
- 可编程的数据帧长度，4位到16位可选；
- 内部环回测试模式，能够很方便地实现诊断/调试；
- 标准 FIFO 中断以及发送结束中断；
- 用微型直接内存访问 (µDMA) 有效的传输数据
  - 相互独立的发送通道和接收通道
  - 当接收 FIFO 中有数据时产生单次请求；当接收 FIFO 中包含 4 个数据单元时产生猝发请求
  - 发送单次请求在 FIFO 有空闲单元时有效；发送猝发请求在有 4 个及以上条目可以写入 FIFO 中时有效

15.1 结构框图

图 **15-1. SSI**模块的结构图



15.2 信号描述

下面的表格列出了 SSI 模块的外部信号，并描述了每个信号的功能。大多数 SSI 信号是某些 GPIO 信号的备用功能，在复位的时候默认为 GPIO 信号。但 SSI0Clk、SSI0Fss、SSI0Rx 和 SSI0Tx 管脚例外，它们将默认为 SSI 功能。下表中“复用管脚/赋值”一栏列出了 SSI 信号可能的 GPIO 管脚布局。将 GPIO 备用功能选择 (GPIOAFSEL) 寄存器（595页）中的 AFSEL 位置位，以便选择 SSI 功能。必须将括号中的数字写入 GPIO 端口控制 (GPIOPCTL) 寄存器（612页）的 PMCn 域中，以便

把 SSI 信号分配给指定 GPIO 端口管脚。有关如何配置 GPIO 的更多信息，请参阅“通用输入/输入端口（GPIOs）”（575页）。

表 **15-1. SSI** 信号 **(64LQFP)**

管脚名称	管脚编号	管脚复用/管脚赋值	管脚类型	缓冲区类型[a]	描述
SSI0Clk	19	PA2 (2)	I/O	TTL	SSI模块0时钟
SSI0Fss	20	PA3 (2)	I/O	TTL	SSI 模块 0 帧信号
SSI0Rx	21	PA4 (2)	I	TTL	SSI模块0接收
SSI0Tx	22	PA5 (2)	O	TTL	SSI模块0发送
SSI1Clk	30 61	PF2 (2) PD0 (2)	I/O	TTL	SSI 模块 1 时钟信号。
SSI1Fss	31 62	PF3 (2) PD1 (2)	I/O	TTL	SSI 模块 1 帧信号。
SSI1Rx	28 63	PF0 (2) PD2 (2)	I	TTL	SSI 模块 1 接收信号。
SSI1Tx	29 64	PF1 (2) PD3 (2)	O	TTL	SSI 模块 1 发送信号。
SSI2Clk	58	PB4 (2)	I/O	TTL	SSI 模块 2 时钟信号。
SSI2Fss	57	PB5 (2)	I/O	TTL	SSI 模块 2 帧信号。
SSI2Rx	1	PB6 (2)	I	TTL	SSI 模块 2 接收信号。
SSI2Tx	4	PB7 (2)	O	TTL	SSI 模块 2 发送信号。
SSI3Clk	61	PD0 (1)	I/O	TTL	SSI 模块 3 时钟信号。
SSI3Fss	62	PD1 (1)	I/O	TTL	SSI 模块 3 帧信号。
SSI3Rx	63	PD2 (1)	I	TTL	SSI 模块 3 接收信号。
SSI3Tx	64	PD3 (1)	O	TTL	SSI 模块 3 发送信号。

a. TTL 表示管脚的电压水平与 TTL 一致。

15.3 功能说明

SSI对从片外器件接收的数据进行串-并转换。CPU可访问数据、控制信息以及状态信息。发送及接收通道均内置FIFO存储器，在发送及接收模式下各自最多能缓冲8个16位数值。SSI 还支持 µDMA 接口。可以将发送和接收 FIFO 配置成 µDMA 模块的目的地址/源地址。将 SSIDMACTL 寄存器中相应的位置位（请参考885页）即可启用 µDMA 操作。

15.3.1 位速率的产生

SSI模块内置可编程的位速率时钟分频器以及预分频器，通过分频产生串行输出时钟。SSI模块支持2MHz或更高的位速率，实际使用时最高位速率通常由片外器件的性能决定。

串行位速率是由输入时钟（SysClk）分频后得到的。首先，使用2～254之间的偶数分频值CPSDVSR对输入时钟进行分频，该值在SSI时钟预分频(SSICPSR)寄存器中设置（请参考878页）。然后再使用1～256之间的一个数（即1＋SCR）对时钟进一步分频，此处的SCR在SSI控制0(SSICR0)寄存器中设置（请参考871页）。

因此输出时钟 SSInClk 的频率为：

```
SSInClk = SysClk / (CPSDVSR * (1 + SCR))
```

注意： 系统时钟或 PIOSC 可用作 SSInClk 的源。当 SSI 时钟配置 (SSICC) 寄存器的 CS 域配置为 0x5 时，PIOSC 被选为源。在主机模式下，系统时钟或PIOSC的速度必须至少是SSInClk

的两倍，而 SSInClk 不能超过 25 MHz。在从机模式下，系统时钟或 PIOSC 的速度必须至少是 SSIClk 的 12 倍，而 SSInClk 不能超过 6.67 MHz。

SSI 的时序参数请参考 “Synchronous Serial Interface (SSI)”（1061页）。

15.3.2 FIFO操作

15.3.2.1 发送FIFO

SSI发送FIFO是一组16位宽、8单元深的先入先出缓冲区。CPU 通过写 SSI 数据 (SSIDR) 寄存器将数据写入发送 FIFO，数据在由发送逻辑读出之前一直保存在发送 FIFO 中（请参考 875页）。

当工作于主机或从机模式时，数据以并行方式写入发送 FIFO，然后进行并 - 串转换并通过 SSInTx 管脚分别发送给片外连接的从设备或主设备。

当工作于从机模式时，SSI模块在每次主设备启动会话后才发送数据。若发送FIFO空，那么在主机启动会话时SSI模块会将发送FIFO中最旧的一个数据发送出去。通过 RCGCSSI 寄存器的 Rn 位启用 SSI 模块时钟后，如果写入发送 FIFO 的有效数据不足 8 个，则 SSI 模块将发送 0。因此，必须依会话要求确保FIFO内包含有效数据。SSI 模块可配置为在 FIFO 空时产生中断或 µDMA 请求。

15.3.2.2 接收FIFO

SSI接收FIFO是一组16位宽、8单元深的先入先出缓冲区。从串行接口接收到的数据在由 CPU 读出之前一直保存在缓冲区中，CPU 通过读 SSIDR 寄存器来访问读 FIFO。

当工作于主机或从机模式时，自 SSInRx 管脚接收的串行数据首先进行保存，而后分别并行载入片外主机或从机接收 FIFO 中。

15.3.3 中断信号

SSI模块可在出现以下情况时产生中断：

- 发送FIFO服务（发送FIFO半满或更低）
- 接收FIFO服务（接收FIFO半满或更多）
- 接收FIFO超时
- 接收FIFO溢出
- 传输结束
- 接收 DMA 传输完成
- 发送 DMA 传输完成

在发送给中断控制器之前，所有中断事件先进行一次逻辑或操作，因此同一时刻不管实际发生了多少SSI中断事件，SSI模块都只向中断控制器产生一个中断请求。将 SSI 中断屏蔽 (SSIIM) 寄存器（请参考879页）中相应的位清零，可以单独屏蔽这四个可屏蔽中断中的任何一个。将相应的屏蔽位置位来启用中断。

SSI模块不但提供组合的中断输出，还分别提供各个中断源的输出，因此在处理中断时既可采用全局中断服务子程序，也可采用模块化的设备驱动程序。动态的发送/接收数据流中断与静态的状态中断相互独立，方便即时响应FIFO触发深度进行读写操作。独立中断源的状态可以查询 SSI 原始中断状态 (SSIRIS) 和 SSI 屏蔽中断状态 (SSIMIS) 寄存器（请分别参考880页和882页）。

接收 FIFO 设有 32 个 SSInClk 时钟周期（不管此时 SSInClk 是否激活）的超时周期，并且只要接收 FIFO 从空状态变为非空状态即会启动。假如接收FIFO在接下来的32个SSIClk周期内再次变为空

状态，超时周期才会中止并复位。因此中断服务程序应在读出接收 FIFO 后及时对 SSI 中断清除寄存器 (SSIIC) 的 RTIC 位写 1，以清除接收 FIFO 超时中断。此清除操作不得执行得太晚，否则有可能中断服务子程序在中断实际清除之前已经返回，此外也可能造成不必要地重复进入中断。

发送结束 (EOT) 中断指明数据已完全发送，只能有效用于主机模式设备/操作。该中断可以用来指示什么时候关闭 SSI 模块的时钟或者进入休眠模式。另外由于数据的发送和接收是同时完成的，此中断也能即时指示接收FIFO中的数据已经就绪，无需等待接收FIFO超时了。

注意: 仅在飞思卡尔 SPI 模式下，即便 FIFO 已满，也可实现每发送一个字节即产生一个 EOT 中断信号。如果集成的从机 SSI 将 EOT 位置 0，而 µDMA 被配置为通过外部回送将该 SSI 中的数据传输到设备所用主机 SSI 中，那么即便 FIFO 已满，也可实现每发送一个字节 SSI 从机就产生一个 EOT 中断信号。

15.3.4 帧格式

根据所设置的数据大小，每个数据帧的长度均在 4 ~ 16 位之间，并且从最高有效位 (MSB) 开始发送。通过对 SSICR0 寄存器中的 FRF 位进行设置，用户可选择三种基本的帧类型：

- Texas Instruments 同步串行
- 飞思卡尔SPI格式
- MICROWIRE格式

对于上述 3 种帧格式，串行时钟 (SSInClk) 在 SSI 空闲时保持不活动状态，只有当数据的发送或接收处于活动状态时，SSInClk 才在设置好的频率下工作。利用 SSInClk 的空闲状态可提供接收超时指示，如果一个超时周期之后接收 FIFO 仍含有数据，则产生超时指示。

对于飞思卡尔 SPI 和 MICROWIRE 帧格式，串行帧 (SSInFss) 管脚为低电平有效，并在整个帧传输过程中保持有效（被下拉）。

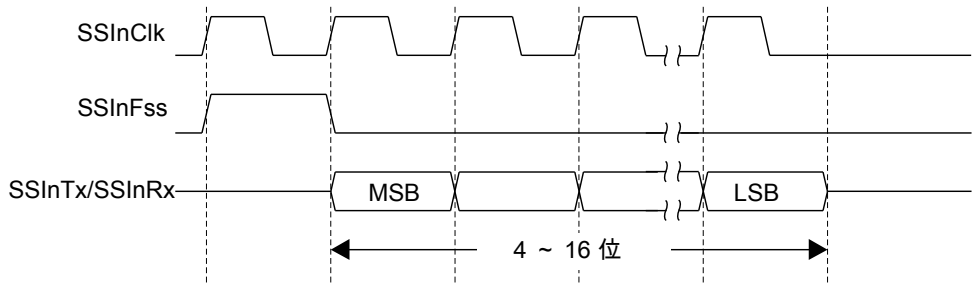
而对于 Texas Instruments 同步串行帧格式，在发送每个帧之前，SSInFss 管脚会发出一个以上升沿开始并持续一个时钟周期的脉冲。在这种帧格式中，SSI 和片外从器件在 SSInClk 的上升沿驱动各自的输出数据，并在下降沿锁存另一个器件的数据。

与另外两种全双工的帧格式不同，MICROWIRE是半双工工作的，其采用特殊的主-从消息技术。在该模式中，当帧开始传输时向片外从机发送一个 8 位的控制报文。在发送过程中，SSI 不会接收到任何输入数据。当控制字发送结束（8位的最后1位发送完成）后，片外从设备即对控制字进行译码。总线等待1个时钟周期之后，片外从设备开始以所请求的数据应答，应答数据长度为4位到16位，于是单次会话的总帧长为13位到25位。

15.3.4.1 Texas Instruments 同步串行帧格式

图15-2（860页）显示了 Texas Instruments 同步串行单次传输的的帧格式。

图 **15-2. TI** 同步串行帧格式（单次传输）

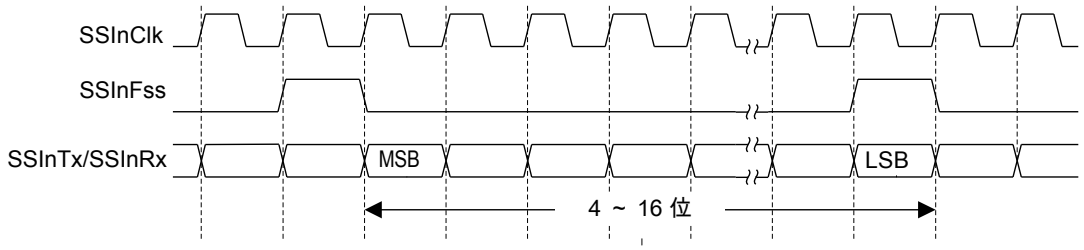


在此模式中，当SSI模块处于空闲状态时，SSInClk和SSInFss强制拉低，SSI发送数据管脚SSInTx被置为三态。一旦发送 FIFO 的底部入口包含数据，SSInFss 就会变为高电平并持续一个 SSInClk 周期。要发送的值也从发送 FIFO 传输到发送逻辑的串行移位寄存器中。在下一个 SSInClk 时钟上升沿，数据帧（长度为 4 到 16 位）的最高有效位移位输出到 SSInTx 管脚上。同样，接收到的数据的 MSB 也通过片外串行从器件移到 SSInRx 管脚上。

然后，SSI和片外串行从器件在SSInClk的每一个下降沿时将数据位逐个移入各自的串行移位器中。在锁存了 LSB 之后的第一个 SSInClk 上升沿，接收数据从串行移位器传输到接收 FIFO。

图15-3（861页）显示了 Texas Instruments 同步串行帧格式的连续的传输情况。

图 **15-3. TI** 同步串行的帧格式（连续传输）



15.3.4.2 飞思卡尔 SPI 帧格式

飞思卡尔 SPI 接口是一个 4 线接口，其中 SSInFss 信号用作从机选择。飞思卡尔 SPI 格式的主要特征是其 SSInClk 信号定义相当灵活，其非活动状态及相位可分别通过 SSISCR0 控制寄存器的 SPO 位以及 SPH 位进行设置。

SPO时钟极性位

当 SPO 时钟极性控制位清零时，它在 SSInClk 管脚上产生稳定的低电平值。如果 SPO 位置位，则在没有进行数据传输的情况下，它在 SSInClk 管脚上产生一个稳定的高电平值。

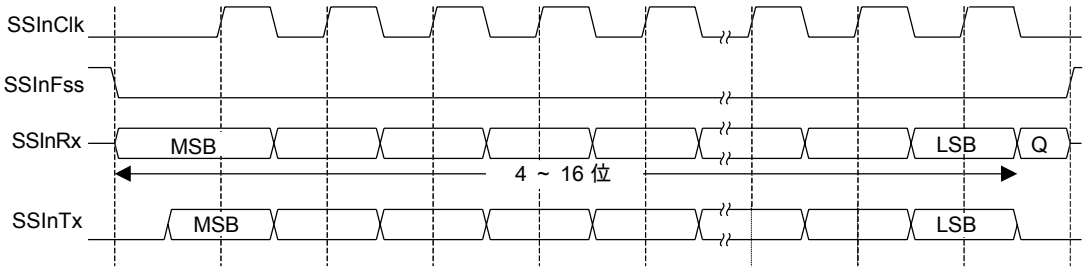
SPH相位控制位

SPH 相位控制位用于选择捕捉数据的时钟沿，允许其改变状态。此标志位将决定如何界定传输过程的首位，即是否在首次捕捉数据之前忽略1次时钟跳变。当 SPH 相位控制位清零时，在第一个时钟边沿转换时捕获数据。如果 SPH 位置位，则在第二个时钟边沿转换时捕获数据。

15.3.4.3 SPO = 0、SPH = 0 时的飞思卡尔 SPI 帧格式

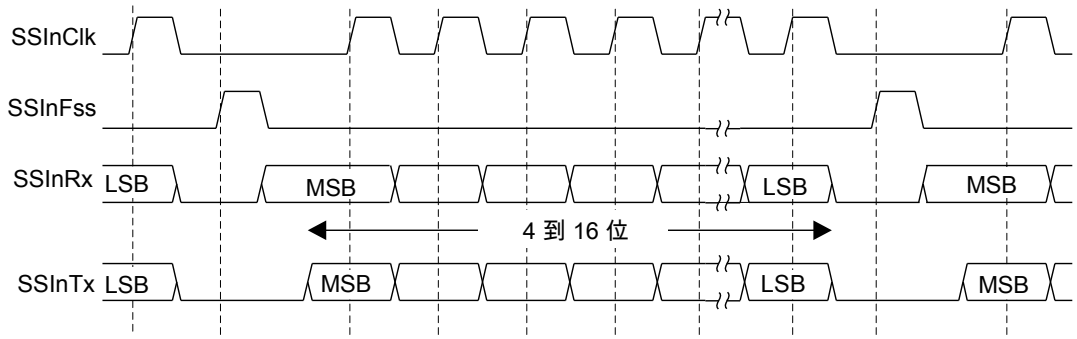
SPO = 0 和 SPH = 0 时，Freescale SPI 帧格式的单次和连续传输信号序列如图15-4（862页）和图15-5（862页）所示。

图 15-4. SPO = 0 和 SPH = 0 时的飞思卡尔 SPI 格式（单次传输）



注意: Q未定义。

图 15-5. SPO = 0 和 SPH = 0 时的飞思卡尔 SPI 格式 (连续传输)



在这种格式配置下，SSI空闲期间：

- SSInClk 强制拉低
- SSInFss 强制拉高
- 发送数据管脚 SSInTx 被仲裁强制拉低
- 若 SSI 模块工作于主机模式，则开启 SSInClk 管脚输出
- 若 SSI 模块工作于从机模式，则关闭 SSInClk 管脚输出

若 SSI 模块使能，并且发送 FIFO 中已经填入有效数据，那么传输过程在 SSInFss 主机信号拉低时开始；这可使从机数据立即传输到主机的 SSInRx 输入线上。与此同时主设备的 SSInTx 输出端口也将使能。

在半个 SSInClk 周期之后，有效的主机数据传输到 SSInTx 管脚。主机和从机数据都已设置好后，则在下半个 SSInClk 周期后，SSInClk 主机时钟管脚变为高电平。

之后双方将在每个 SSInClk 时钟的上升沿捕捉数据、在下降沿进行传输。

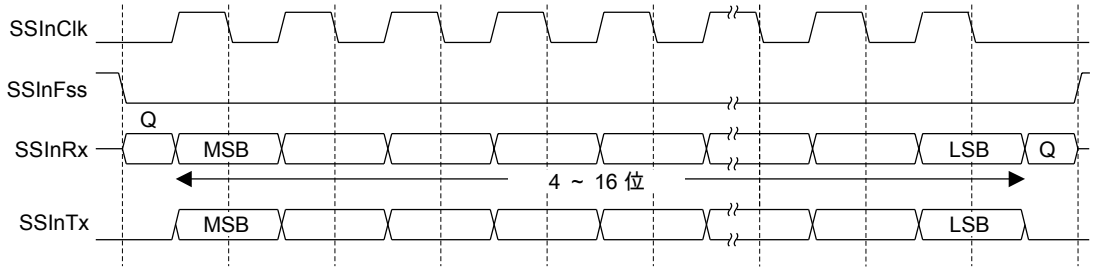
如果传输一个字，则在数据字的所有位都已传输完之后，SSInFss 线在捕获到最后一个位后的一个 SSInClk 周期返回到其空闲的高电平状态。

在背靠背连续传输过程中，必须使SSInFss信号在相邻两次数据传输之间输出高脉冲，因为当SPH位清 0 时，从设备选通管脚将锁住串行外设寄存器中的数据，不允许对其修改。因此主设备必须在相邻两次数据传输之间拉高 SSInFss 管脚，以使能串行外设数据写入操作。当连续传输完成时，SSInFss 管脚将在捕获到最后一位后的一个 SSInClk 周期返回到其空闲状态。

15.3.4.4 SPO = 0、SPH = 1 时的飞思卡尔 SPI 帧格式

SPO = 0 和 SPH = 1 时，飞思卡尔 SPI 帧格式的传输信号序列如图15-6（863页） 所示，该图涵盖了单次和连续传输两种情况。

图 **15-6. SPO = 0、SPH = 1 时的飞思卡尔 SPI 帧格式**



注意: Q未定义。

在这种格式配置下，SSI空闲期间：

- SSInClk 强制拉低
- SSInFss 强制拉高
- 发送数据管脚 SSInTx 被仲裁强制拉低
- 若 SSI 模块工作于主机模式，则开启 SSInClk 管脚输出
- 若 SSI 模块工作于从机模式，则关闭 SSInClk 管脚输出

如果 SSI 启用并且在发送 FIFO 中含有有效的数据，则 SSInFss 主机信号驱动为低电平时发送操作开始。与此同时主设备的 SSInTx 输出也将使能。再过半个 SSInClk 时钟周期后，主设备和从设备的有效数据都已在各自的发送线上就绪。同时，利用一个上升沿跳变将 SSInClk 使能。

之后，数据在 SSInClk 信号的下降沿被捕获，在上升沿进行传输。

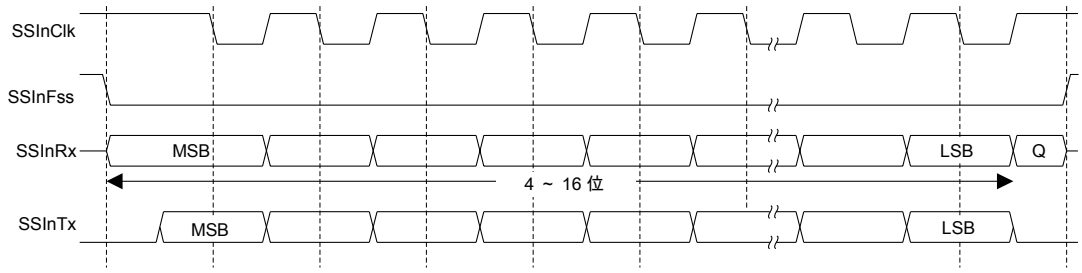
在单字传输过程中，当数据字的所有位传输完成后，SSInFss 管脚将在捕捉最后 1 位（下降沿）后 1 个 SSInClk 周期恢复其空闲的高电平状态。

如果是连续背对背传输，则 SSInFss 管脚在连续的数据字之间保持为低电平，连续传输的结束情况与单字传输的相同。

15.3.4.5 SPO = 1、SPH = 0 时的飞思卡尔 SPI 帧格式

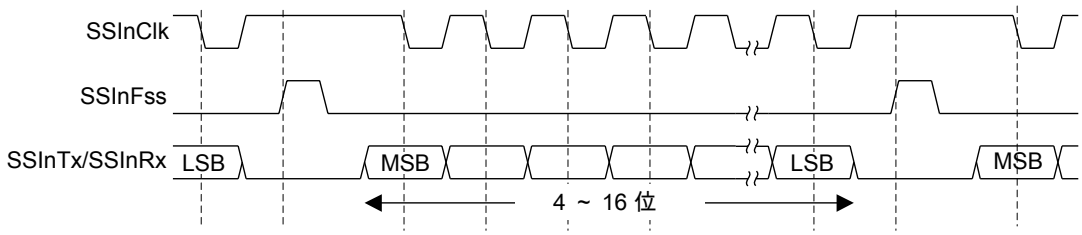
SPO = 1 和 SPH = 0 时，Freescale SPI 帧格式的单次和连续传输信号序列如图15-7（864页）和图15-8（864页）所示。

图 15-7. SPO = 1 和 SPH = 0 时的飞思卡尔 SPI 帧格式（单次传输）



注意: Q未定义。

图 15-8. SPO = 1 和 SPH = 0 时的飞思卡尔 SPI 帧格式（连续传输）



在这种格式配置下，SSI空闲期间：

- SSInClk 强制拉高
- SSInFss 强制拉高
- 发送数据管脚 SSInTx 被仲裁强制拉低
- 若 SSI 模块工作于主机模式，则开启 SSInClk 管脚输出
- 若 SSI 模块工作于从机模式，则关闭 SSInClk 管脚输出

若 SSI 模块使能，并且发送 FIFO 中已经填入有效数据，那么传输过程在 SSInFss 主机信号拉低时开始；这可使从机数据立即传输到主机的 SSInRx 线上。与此同时主设备的 SSInTx 输出端口也将使能。

半个周期之后，有效的主机数据传输到 SSInTx 线上。一旦主机和从机数据都已设置好，再经过半个 SSInClk 周期，SSInClk 主机时钟管脚变为低电平。这表示数据在每个 SSInClk 时钟的下降沿被捕获，在上升沿进行传输。

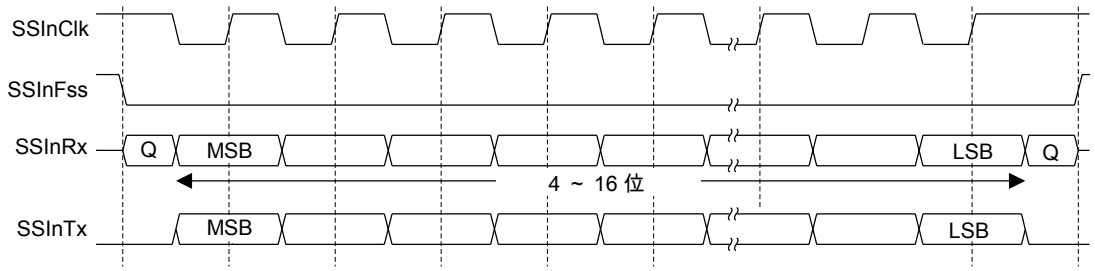
在单字传输过程中，当数据字的所有位传输完成后，SSInFss 管脚将在捕捉最后 1 位（下降沿）后的 1 个 SSInClk 恢复其空闲的高电平状态。

在背靠背连续传输过程中，必须使SSInFss信号在相邻两次数据传输之间输出高脉冲，因为当SPH位清 0 时，从设备选通管脚将锁住串行外设寄存器中的数据，不允许对其修改。因此主设备必须在相邻两次数据传输之间拉高 SSInFss 管脚，以使能串行外设数据写入操作。当连续传输完成时，SSInFss 管脚将在捕获到最后一位后的一个 SSInClk 周期返回到其空闲状态。

15.3.4.6 SPO = 1、SPH = 1 时的飞思卡尔 SPI 帧格式

SPO = 1 和 SPH = 1 时，飞思卡尔 SPI 帧格式的传输信号序列如图15-9（865页）所示，该图涵盖了单次和连续传输两种情况。

图 **15-9. SPO = 1、SPH = 1** 时的飞思卡尔 **SPI** 帧格式



注意: Q未定义。

在这种格式配置下，SSI空闲期间：

- SSInClk 强制拉高
- SSInFss 强制拉高
- 发送数据管脚 SSInTx 被仲裁强制拉低
- 若 SSI 模块工作于主机模式，则开启 SSInClk 管脚输出
- 若 SSI 模块工作于从机模式，则关闭 SSInClk 管脚输出

如果 SSI 启用并且在发送 FIFO 中含有有效的数据，则 SSInFss 主机信号驱动为低电平时发送操作开始。与此同时主设备的 SSInTx 输出端口也将使能。再过半个 SSInClk 周期后，主设备和从设备的数据都已在各自的发送线上就绪。同时，利用下降沿跳变将 SSInClk 使能。之后数据在每个 SSInClk 时钟信号的上升沿被捕获，在下降沿移位输出。

在单字传输过程中，当数据字的所有位传输完成后，SSInFss 管脚将在捕捉最后 1 位（上升沿）后的 1 个 SSInClk 周期恢复其空闲的高电平状态。

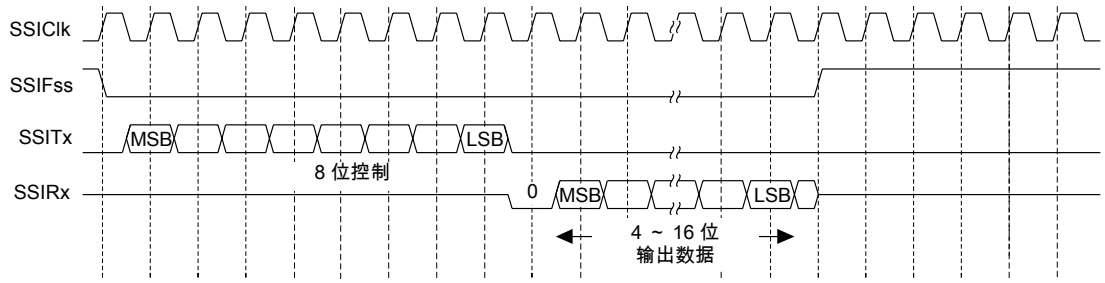
在背靠背连续传输过程中，SSInFss 信号在连续的数据传输过程中始终保持有效（拉低），直到捕捉最后 1 个字的最后 1 位（上升沿）之后像单字传输那样返回其空闲状态。

在背靠背连续传输过程中，SSInFss 管脚在连续的数据字之间保持为低电平，连续传输的结束情况与单字传输的相同。

15.3.4.7 MICROWIRE帧格式

图15-10（865页）显示了单次传输的 MICROWIRE 帧格式图15-11（866页）显示了该格式的连续的传输情况

图 **15-10. MICROWIRE**的帧格式 **(**单帧**)**



MICROWIRE的帧格式与SPI非常相近，区别之处在于MICROWIRE是半双工而非全双工，而且还采用了主-从报文传递技术。每次串行传输都由 SSI 向片外从器件发送 8 位控制字开始。在此传输过程中，SSI 不会接收到输入的数据。当控制字发送结束（8位的最后1位发送完成）后，片外从设备即对控制字进行译码。总线等待1个时钟周期之后，片外从设备开始以所请求的数据应答，应答数据长度为4位到16位，于是单次会话的总帧长为13位到25位。

在这种格式配置下，SSI空闲期间：

- SSInClk 强制拉低
- SSInFss 强制拉高
- 发送数据管脚 SSInTx 被仲裁强制拉低

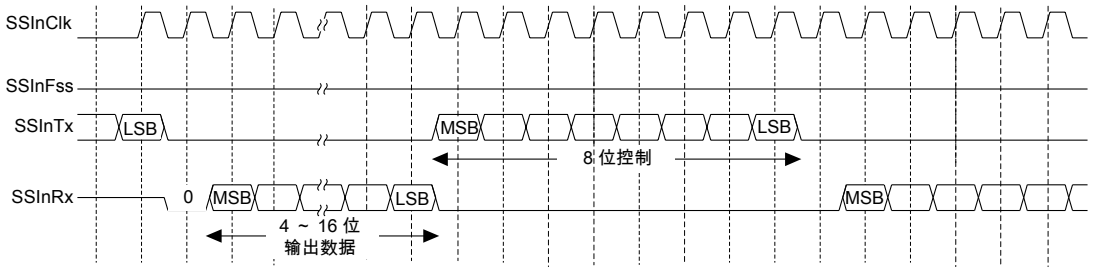
向发送FIFO内写入控制字节即可触发发送。在 SSInFss 的下降沿，发送 FIFO 底部入口包含的值被传输到发送逻辑的串行移位寄存器中，而 8 位控制帧的 MSB 被移出到 SSInTx 管脚上。SSInFss在该控制帧的传输期间保持低电平。SSInRx 管脚在此传输期间保持三态。

片外串行从器件会在 SSInClk 时钟的每个上升沿将每个控制位锁存入串行移位寄存器。在将最后一位锁存之后，从器件在一个时钟周期的等待状态期间对控制字节进行译码，并且从机将数据发送回SSI进行应答。每个数据位在 SSInRx 的下降沿时刻被驱动到 SSInClk 线上。SSI 在 SSInClk 的上升沿时将每个位锁存。在帧尾，对于单次传输，SSInFss 信号在最后 1 位被锁存入接收串行移位寄存器（上升沿）的 1 个时钟周期被拉高，使接收的数据传递至接收 FIFO。

注意: 在接收移位器将 LSB 锁存之后的 SSInClk 的下降沿上或在 SSInFss 管脚变为高电平时，片外从器件能够将接收线置为三态。

连续传输的开始及结束均与单次传输大体相同。但 SSInFss 线持续有效（保持低电平），并且数据传输以背对背的方式产生。下一帧的控制字节紧跟当前接收数据的最低有效位 (LSB)。在当前帧的 LSB 锁存到 SSI 之后，所接收到的每个值在 SSInClk 的下降沿时刻从接收移位器中输出。

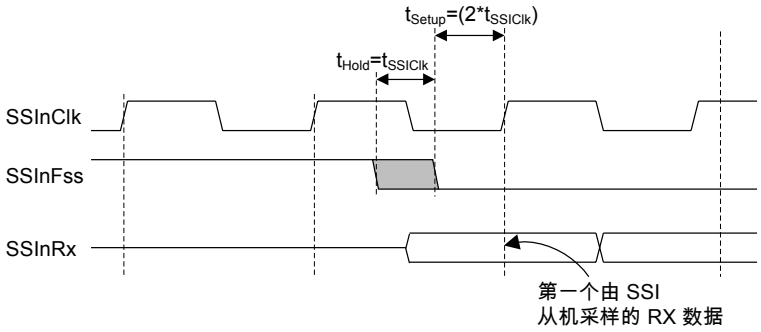
图 **15-11. MICROWIRE**的帧格式 **(**连续传输**)**



在 MICROWIRE 模式中，当 SSInClk 变为低电平之后，SSI 从机在 SSInFss 的上升沿时刻对接收数据的第一个位进行采样。用来驱动自由运行的 SSInClk 的主机必须确保 SSInFss 信号相对于 SSInClk 的上升沿具有足够的建立时间和保持时间裕量。

图15-12（867页）阐明了建立和保持时间要求。相对于 SSI 从机对接收数据的第一位进行采样时所在的 SSInClk 上升沿，SSInFss 的建立时间至少必须是 SSI 操作时钟 (SSInClk) 周期的两倍。相对于该边沿之前的 SSInClk 上升沿，SSInFss 至少必须具有一个 SSInClk 周期的保持时间。

图 **15-12. MICROWIRE** 帧格式，**SSInFss** 输入建立和保持时间要求



15.3.5 DMA 操作

SSI 模块可与 µDMA 控制器接口实现相互独立的发送通道和接收通道。SSI 的 µDMA 操作通过 SSI DMA 控制寄存器 (SSIDMACTL) 使能。在使能µDMA操作后，SSI模块在接收FIFO或发送FIFO可以传输数据时向接收或发送通道产生µDMA请求。对于接收通道，只要接收 FIFO 中有数据，就会发出单次传输请求。如果接收 FIFO 中的数据是 4 个或者更多，就会发出多个连续传输请求。对于发送通道，只要发送 FIFO 中存在一个空位，单次传输请求就会发出。如果发送 FIFO 中的空位有 4 个或者更多，就会发出多个连续传输请求。µDMA 控制器会根据 µDMA 通道的配置自动处理单次传输请求和连续传输请求。为了启用接收通道的 µDMA 操作，寄存器 DMA 控制 (SSIDMACTL) 中的 RXDMAE 位应置位。为了启用发送通道的 µDMA 操作，寄存器 SSIDMACTL 中的 TXDMAE 位应置位。如果已经启用了 µDMA，那么 µDMA 控制器会在传输结束时自动触发中断。此中断产生到 SSI 中断向量。因此，如果启用了 SSI 操作使用的中断和µDMA，必须设计 SSI 中断处理函数来处理 µDMA 的完成中断。

请参见“微型直接存储器访问（µDMA）”（512页）以了解有关对 µDMA 控制器进行配置的更多信息。

15.4 初始化和配置

请按照以下步骤使能并初始化SSI模块：

1. 用 RCGCSSI 寄存器（请参考301页）启用 SSI 模块。
2. 通过 RCGCGPIO 寄存器启用相应 GPIO 模块的时钟，请参考 295页。要了解要启用哪一个 GPIO 端口，请参考表20-5（1024页）。
3. 将相应管脚的 GPIO AFSEL 位置位（请参阅595页）。为了确定哪些 GPIO 需要配置，请参考表20-4（1019页）。
4. 配置 GPIOPCTL 寄存器的 PMCn 域，将 SSI 信号赋给相应的管脚。请参阅612页和表 20-5（1024页）。
5. 对 GPIODEN 寄存器编程，以启用管脚的数字功能。此外，必须配置驱动强度、开漏选择和上拉/下拉功能。请参阅 “通用输入/输入端口（GPIOs）”（575页） 以了解更多信息。

   注意: 上拉可用来避免 SSI 管脚上不必要的切换，该切换会将从机带入错误状态。

对于不同的帧格式，应按照如下步骤配置SSI模块：

1. 应确保 SSICR1 寄存器的 SSE 位在更改配置前清零。
2. 选择SSI模块工作于主机模式还是从机模式：

a. 若工作于主机模式，应将 SSICR1 寄存器配置为 0x0000.0000；

b. 若工作于从机模式（允许输出），应将 SSICR1 寄存器配置为 0x0000.0004；

c. 若工作于从机模式（禁止输出），应将 SSICR1 寄存器配置为 0x0000.000C。

3. 通过写 SSICC 寄存器来配置 SSI 时钟源。

4. 通过写 SSICPSR 寄存器配置时钟预分频除数。

5. 通过 SSICR0 寄存器配置以下内容：

- 串行时钟速率 (SCR)
- 若采用飞思卡尔 SPI 帧格式，需配置时钟相位和时钟极性（SPH、SPO）
- 协议模式：飞思卡尔 SPI、TI SSF、MICROWIRE (FRF)
- 数据长度 (DSS)

6. 另外，还可以配置 µDMA 通道（请参考“微型直接存储器访问（µDMA）”（512页）），并在 SSIDMACTL 寄存器中启用 DMA 选项。

7. 将 SSICR1 寄存器的 SSE 位置位，以使能 SSI 模块。

下面举例予以说明。假定SSI模块要按照如下参数工作：

- 主机模式
- 飞思卡尔SPI格式（SPO = 1，SPH = 1）
- 位速率为1Mbps
- 数据长度8位

假设系统时钟20MHz，于是位速率计算公式为：

```
SSInClk = SysClk / (CPSDVSR * (1 + SCR))
1x10^6  = 20x10^6 / (CPSDVSR * (1 + SCR))
```

在此情况下，如果 CPSDVSR = 0x2，SCR 必须为 0x9。

软件中的配置步骤为：

1. 确保 SSICR1 寄存器的 SSE 位清零。

2. 对 SSICR1 寄存器写入 0x0000.0000；

3. 对 SSICPSR 寄存器写入 0x0000.0002；

4. 对 SSICR0 寄存器写入 0x0000.09C7。

5. 随后将 SSICR1 寄存器的 SSE 位置位，使能 SSI 模块。

15.5 寄存器映射

表 15-2（869页）列出了 SSI 寄存器。表中偏移量一列是指相对于SSI基地址的十六进制地址增量，两个SSI模块的基地址分别为：

- SSI0：0x4000.8000
- SSI1：0x4000.9000
- SSI2：0x4000.A000
- SSI3：0x4000.B000

请注意配置这些寄存器之前必须先启用 SSI 模块时钟（请参考301页）。PRSSI 寄存器的 Rn 位必须读数为 0x1，才能访问任一 SSI 模块寄存器。

注意: 在对任何控制寄存器重新编程前，必须将 SSI 禁能(见 SSICR1 寄存器的 SSE 位)。

表 **15-2. SSI** 寄存器映射

偏移量	名称	类型	复位	描述	见页面
0x000	SSICR0	R/W	0x0000.0000	SSI 控制寄存器0	871
0x004	SSICR1	R/W	0x0000.0000	SSI 控制寄存器1	873
0x008	SSIDR	R/W	0x0000.0000	SSI 数据寄存器	875
0x00C	SSISR	RO	0x0000.0003	SSI 状态寄存器	876
0x010	SSICPSR	R/W	0x0000.0000	SSI 时钟预分频寄存器	878
0x014	SSIIM	R/W	0x0000.0000	SSI 中断屏蔽寄存器	879
0x018	SSIRIS	RO	0x0000.0008	SSI 原始中断状态寄存器	880
0x01C	SSIMIS	RO	0x0000.0000	SSI 屏蔽中断状态寄存器	882
0x020	SSIICR	W1C	0x0000.0000	SSI 中断清除寄存器	884
0x024	SSIDMACTL	R/W	0x0000.0000	SSI DMA 控制寄存器	885
0xFC8	SSICC	R/W	0x0000.0000	SSI 时钟配置寄存器	886
0xFD0	SSIPeriphID4	RO	0x0000.0000	SSI 外设标识寄存器 4	887
0xFD4	SSIPeriphID5	RO	0x0000.0000	SSI 外设标识寄存器 5	888
0xFD8	SSIPeriphID6	RO	0x0000.0000	SSI 外设标识寄存器 6	889
0xFDC	SSIPeriphID7	RO	0x0000.0000	SSI 外设标识寄存器 7	890
0xFE0	SSIPeriphID0	RO	0x0000.0022	SSI 外设标识寄存器 0	891
0xFE4	SSIPeriphID1	RO	0x0000.0000	SSI 外设标识寄存器 1	892
0xFE8	SSIPeriphID2	RO	0x0000.0018	SSI 外设标识寄存器 2	893
0xFEC	SSIPeriphID3	RO	0x0000.0001	SSI 外设标识寄存器 3	894
0xFF0	SSIPCellID0	RO	0x0000.000D	SSI PrimeCell 标识寄存器 0	895
0xFF4	SSIPCellID1	RO	0x0000.00F0	SSI PrimeCell 标识寄存器 1	896
0xFF8	SSIPCellID2	RO	0x0000.0005	SSI PrimeCell 标识寄存器 2	897
0xFFC	SSIPCellID3	RO	0x0000.00B1	SSI PrimeCell 标识寄存器 3	898

15.6 寄存器描述

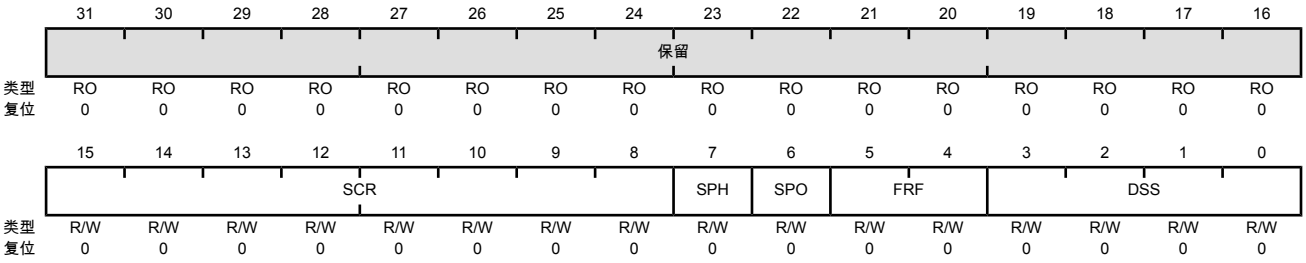
本章的剩余部分按照地址偏移量由小到大的顺序依次详细介绍各寄存器。

寄存器 1: SSI 控制寄存器0（SSICR0），偏移量 0x000

SSICR0 寄存器中包含用来控制 SSI 模块内各种功能的位域。诸如协议模式、时钟速率和数字宽度等功能都在该寄存器中配置。

SSI 控制寄存器0 (SSICR0)

SSI0 基址: 0x4000.8000
SSI1 基址: 0x4000.9000
SSI2 基址: 0x4000.A000
SSI3 基址: 0x4000.B000
偏移量 0x000
类型 R/W, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:16	保留	RO	0x0000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
15:8	SCR	R/W	0x00	SSI串行时钟速率 该域的值用来产生 SSI 的发送和接收位速率位速率： BR=SysClk/(CPSDVSR * (1 + SCR)) 此处，CPSDVSR 为 2-254 之间的一个偶数值，在 SSICPSR 寄存器中设置，SCR 为 0-255 之间的一个值。
7	SPH	R/W	0	SSI串行时钟相位 此位仅用于飞思卡尔SPI格式。 SPH 控制位用来选择捕获数据的时钟边沿，并允许边沿改变状态。该位对第一个传输位影响最大，因为它可以在第一个数据捕获边沿之前允许或不允许一次时钟转换。 值 描述 0 数据在首个时钟跳变沿捕捉 1 数据在第2个时钟跳变沿捕捉
6	SPO	R/W	0	SSI串行时钟极性 值 描述 0 SSInClk 管脚为稳定的低电平值。 1 当无数据传输时，SSInClk 管脚为稳定的高电平值。

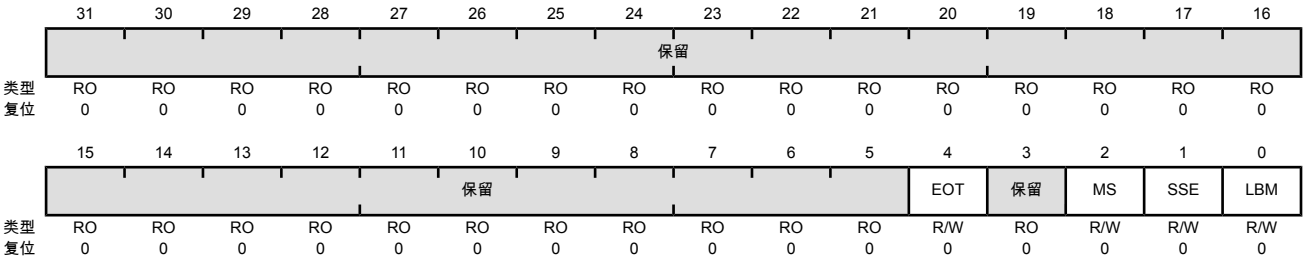
位/域	名称	类型	复位	描述
5:4	FRF	R/W	0x0	SSI帧格式选择 值 帧格式 0x0 飞思卡尔 SPI 帧格式 0x1 Texas Instruments 同步串行帧格式 0x2 MICROWIRE帧格式 0x3 保留
3:0	DSS	R/W	0x0	SSI数据帧长度选择 值 系统数据宽度 0x0-0x2 保留 0x3 传输的每个数据单元为 4 位。 0x4 传输的每个数据单元为 5 位。 0x5 传输的每个数据单元为 6 位。 0x6 传输的每个数据单元为 7 位。 0x7 传输的每个数据单元为 8 位。 0x8 传输的每个数据单元为 9 位。 0x9 传输的每个数据单元为 10 位。 0xA 传输的每个数据单元为 11 位。 0xB 传输的每个数据单元为 12 位。 0xC 传输的每个数据单元为 13 位。 0xD 传输的每个数据单元为 14 位。 0xE 传输的每个数据单元为 15 位。 0xF 传输的每个数据单元为 16 位。

寄存器 2: SSI 控制寄存器1（SSICR1），偏移量 0x004

SSICR1 寄存器中包含用来控制 SSI 模块内各种功能的位域。主机和从机的模式功能由该寄存器控制。

SSI 控制寄存器1 (SSICR1)

SSI0 基址: 0x4000.8000
SSI1 基址: 0x4000.9000
SSI2 基址: 0x4000.A000
SSI3 基址: 0x4000.B000
偏移量 0x004
类型 R/W, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:5	保留	RO	0x0000.0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
4	EOT	R/W	0	传输结束 本位仅对主机模式的设备和操作有效 (MS = 0x0)。 值 描述 0 当发送FIFO内的有效数据不超过4个单元时产生TXRIS中断 1 当传输结束时产生TXRIS中断 注意: 仅在飞思卡尔 SPI 模式下，即便 FIFO 已满，也可实现每发送一个字节即产生一个 EOT 中断信号。如果集成的从机 SSI 将 EOT 位置 0，而 µDMA 被配置为通过外部回送将该 SSI 中的数据传输到设备所用主机 SSI 中，那么即便 FIFO 已满，也可实现每发送一个字节 SSI 从机就产生一个 EOT 中断信号。
3	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
2	MS	R/W	0	SSI主/从选择 此位用于选择主机模式还是从机模式，只有当 SSI 禁用 (SSE=0) 时才能设置。 值 描述 0 SSI模块工作于主机模式 1 SSI模块工作于从机模式

位/域	名称	类型	复位	描述
1	SSE	R/W	0	SSI同步串行接口使能 值 描述 0 禁用SSI模块 1 使能SSI模块 注意: 在对任何控制器重新编程之前必须先将该位清零。
0	LBM	R/W	0	SSI环回模式 值 描述 0 进行正常的串口操作 1 启用环回模式：发送串行移位寄存器的输出端将在芯片内部连接到接收串行移位寄存器的输入端。

寄存器 3: SSI 数据寄存器（SSIDR），偏移量 0x008

重要: 本寄存器为读敏感型寄存器。有关详细信息，请参阅寄存器描述部分。

SSIDR 寄存器是一个 16 位宽的寄存器。当读取 SSIDR 寄存器时，将访问接收 FIFO 中被当前 FIFO 读指针所指向的单元。当SSI接收逻辑单元将输入的数据帧移出后，该数据将放入接收FIFO中由当前FIFO写指针所指向的单元。

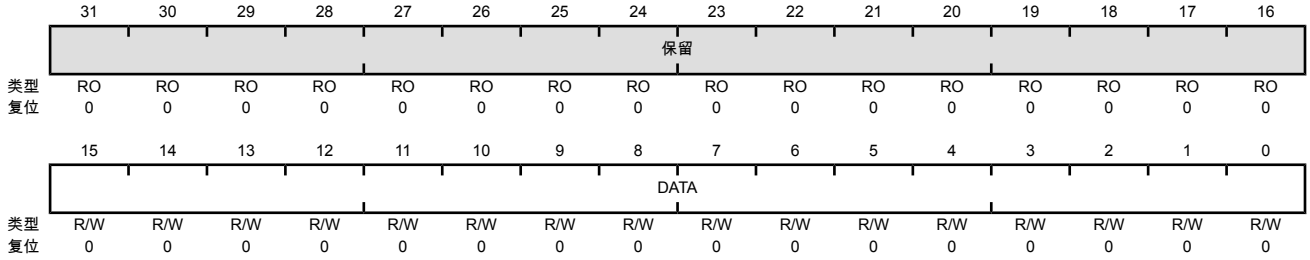
当写入 SSIDR 寄存器时，数据将放入发送 FIFO 中被当前 FIFO 写指针所指向的单元。发送逻辑从发送 FIFO 中一次移出一个数据值。每个数据值被加载到发送串行移位器中，然后以设置好的位速率串行移出到 SSInTx 管脚。

当所选的数据大小小于 16 位时，用户必须将写入发送 FIFO 的数据右对齐。发送逻辑忽略未使用的位。而在接收时，若收到的数据长度不足16位，在接收缓冲中能自动右对齐。

当 SSI 设置为 MICROWIRE 帧格式时，发送数据的默认宽度为 8 位（忽略最高有效字节）。接收数据的宽度由程序员控制。即使 SSICR1 寄存器的 SSE 位清 0，发送 FIFO 和接收 FIFO 也不会清空，因此软件可以在使能 SSI 模块之前先填充发送 FIFO。

SSI 数据寄存器 (SSIDR)

SSI0 基址: 0x4000.8000
SSI1 基址: 0x4000.9000
SSI2 基址: 0x4000.A000
SSI3 基址: 0x4000.B000
偏移量 0x008
类型 R/W, 复位 0x0000.0000



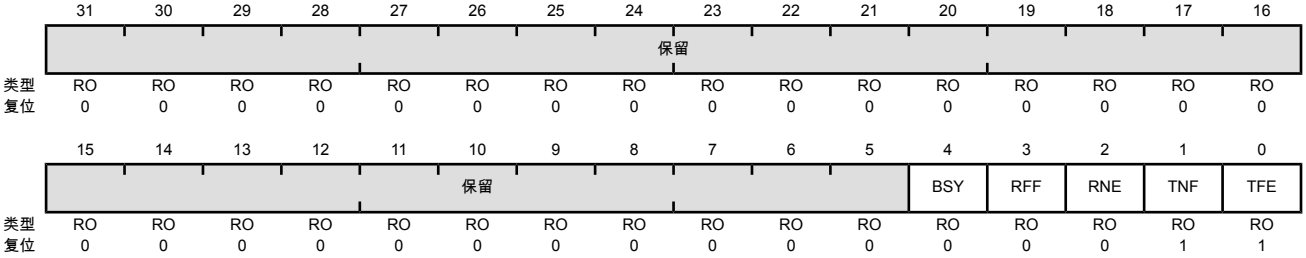
位/域	名称	类型	复位	描述
31:16	保留	RO	0x0000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
15:0	DATA	R/W	0x0000	SSI接收/发送数据 一个读取操作读取接收 FIFO 的数据。一个写入操作将数据写入发送 FIFO。 当 SSI 设置的数据宽度小于 16 位时，软件必须将数据右对齐。发送逻辑将忽略顶部未使用的位。接收逻辑自动右对齐数据。

寄存器 4: SSI 状态寄存器（SSISR），偏移量 0x00C

SSISR 寄存器包含若干标志位，用于指示 FIFO 填充状态和 SSI 忙状态。

SSI 状态寄存器 (SSISR)

SSI0 基址: 0x4000.8000
SSI1 基址: 0x4000.9000
SSI2 基址: 0x4000.A000
SSI3 基址: 0x4000.B000
偏移量 0x00C
类型 RO, 复位 0x0000.0003



位/域	名称	类型	复位	描述
31:5	保留	RO	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
4	BSY	RO	0	SSI忙标志 值 描述 0 SSI 空闲。 1 SSI正在发送/接收数据帧，或发送FIFO非空
3	RFF	RO	0	SSI 接收 FIFO 满 值 描述 0 接收 FIFO 未满 1 接收 FIFO 满
2	RNE	RO	0	SSI FIFO 不为空 值 描述 0 接收 FIFO 为空 1 接收 FIFO 不为空
1	TNF	RO	1	SSI 发送 FIFO 未满 值 描述 0 发送 FIFO 已满 1 发送 FIFO 未满

位/域	名称	类型	复位	描述
0	TFE	RO	1	SSI 发送 FIFO 为空 值 描述 0 发送 FIFO 不为空。 1 发送 FIFO 为空。

寄存器 5: SSI 时钟预分频寄存器（SSICPSR），偏移量 0x010

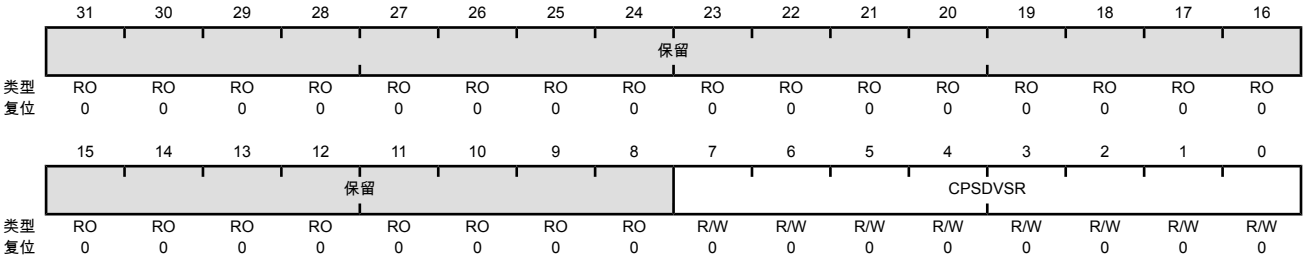
SSICPSR 寄存器指定用于根据系统时钟产生 SSInClk 信号的分频因子。该时钟进一步分频，分频因子为 1 到 256，即 1 + SCR。SCR 通过 SSICR0 寄存器编程配置。因此 SSInClk 频率的计算公式如下：

```
SSInClk = SysClk / (CPSDVSR * (1 + SCR))
```

本寄存器必须写入2到254之间的偶数。所设的值的最低有效位硬编码为 0。如果向该寄存器写入奇数，则该寄存器读操作返回的值中，最低有效位为 0。

SSI 时钟预分频寄存器 (SSICPSR)

SSI0 基址: 0x4000.8000
SSI1 基址: 0x4000.9000
SSI2 基址: 0x4000.A000
SSI3 基址: 0x4000.B000
偏移量 0x010
类型 R/W, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:8	保留	RO	0x00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
7:0	CPSDVSR	R/W	0x00	SSI时钟预分频比 该值必须为 2-254 之间的一个偶数，具体取值由 SSInClk 的频率决定。执行读操作时，LSB 的返回值始终为 0。

寄存器 6: SSI 中断屏蔽寄存器（SSIIM），偏移量 0x014

SSIIM 寄存器是中断屏蔽置位/清零寄存器。本寄存器可读可写，复位后所有位清零。

对本寄存器进行读操作，可获取各个中断的当前屏蔽状态。将某位置位会清除相应的屏蔽，使该中断发送到中断控制器。将某位清零会置位相应的屏蔽，使该中断就不发送到中断控制器。

SSI 中断屏蔽寄存器 (SSIIM)

SSI0 基址: 0x4000.8000
SSI1 基址: 0x4000.9000
SSI2 基址: 0x4000.A000
SSI3 基址: 0x4000.B000
偏移量 0x014
类型 R/W, 复位 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	保留															
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	保留												TXIM	RXIM	RTIM	RORIM
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位/域	名称	类型	复位	描述
31:4	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
3	TXIM	R/W	0	SSI发送FIFO中断屏蔽 值 描述 0 屏蔽发送FIFO中断 1 不屏蔽发送FIFO中断
2	RXIM	R/W	0	SSI 接收 FIFO 中断屏蔽 值 描述 0 接收 FIFO 中断被屏蔽。 1 接收 FIFO 中断没有被屏蔽。
1	RTIM	R/W	0	SSI 接收超时中断屏蔽 值 描述 0 接收 FIFO 超时中断被屏蔽。 1 接收 FIFO 超时中断没有被屏蔽。
0	RORIM	R/W	0	SSI 接收溢出中断屏蔽 值 描述 0 接收 FIFO 溢出中断被屏蔽。 1 接收 FIFO 溢出中断没有被屏蔽。

寄存器 7: SSI 原始中断状态寄存器（SSIRIS），偏移量 0x018

SSIRIS 寄存器为原始中断状态寄存器。读操作时，此寄存器给出屏蔽前的当前原始中断状态值。对本寄存器写操作无效。

SSI 原始中断状态寄存器 (SSIRIS)

SSI0 基址: 0x4000.8000
SSI1 基址: 0x4000.9000
SSI2 基址: 0x4000.A000
SSI3 基址: 0x4000.B000
偏移量 0x018
类型 RO, 复位 0x0000.0008

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
域	保留															
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
域	保留												TXRIS	RXRIS	RTRIS	RORRIS
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0

位/域	名称	类型	复位	描述
31:4	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
3	TXRIS	RO	1	SSI发送FIFO原始中断标志 值 描述 0 无中断。 1 若 SSICR1 寄存器的 EOT 位清零，则表示发送 FIFO 为半空或更少。 若 EOT 置位，则表示发送 FIFO 空，且最后 1 位已经从串行移位器发出。 当发送 FIFO 中有效数据多于 4 个单元（EOT 清零时）或非空（EOT 置位时），该位清零。
2	RXRIS	RO	0	SSI接收FIFO原始中断标志 值 描述 0 无中断。 1 接收FIFO中有效数据不少于4个单元 当接收FIFO中有效数据少于4个单元时，此标志位清零。
1	RTRIS	RO	0	SSI接收超时原始中断标志 值 描述 0 无中断。 1 发生接收超时 当对 SSI 中断清除寄存器 (SSIICR) 的 RTIC 位写入 1 时，本标志位清零。

位/域	名称	类型	复位	描述
0	RORRIS	RO	0	SSI 接收溢出原始中断状态 值 描述 0 无中断。 1 接收 FIEO 发生溢出 当对 SSI 中断清除寄存器 (SSIICR) 的 RORIC 位写入 1 时，该位清零。

寄存器 8: SSI 屏蔽中断状态寄存器（SSIMIS），偏移量 0x01C

SSIMIS 寄存器为屏蔽中断状态寄存器。读取时，该寄存器给出相应中断的当前屏蔽状态值。对本寄存器写操作无效。

SSI 屏蔽中断状态寄存器 (SSIMIS)

SSI0 基址: 0x4000.8000
SSI1 基址: 0x4000.9000
SSI2 基址: 0x4000.A000
SSI3 基址: 0x4000.B000
偏移量 0x01C
类型 RO, 复位 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	保留															
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	保留												TXMIS	RXMIS	RTMIS	RORMIS
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位/域	名称	类型	复位	描述
31:4	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
3	TXMIS	RO	0	SSI 发送 FIFO 屏蔽中断状态 值 描述 0 中断没有发生或者被屏蔽。 1 发送 FIFO 中有效数据超过 4 个单元（EOT 位清零）或者最后 1 位已经通过串口发出（EOT位置位），这两种情况会发出未屏蔽的中断信号。 当发送 FIFO 中有效数据少于 4 个单元（EOT 清零）或非空（EOT 置位）时，该位清零。
2	RXMIS	RO	0	SSI 接收 FIFO 屏蔽中断状态 值 描述 0 中断没有发生或者被屏蔽。 1 由于接收 FIFO 中有效数据超过 4 个单元，因此发出了未屏蔽的中断信号。 当接收FIFO中有效数据少于4个单元时，此标志位清零。
1	RTMIS	RO	0	SSI 接收超时屏蔽中断状态 值 描述 0 中断没有发生或者被屏蔽。 1 接收超时产生中断。 当对 SSI 中断清除寄存器 (SSIICR) 的 RTIC 位写入 1 时，本标志位清零。

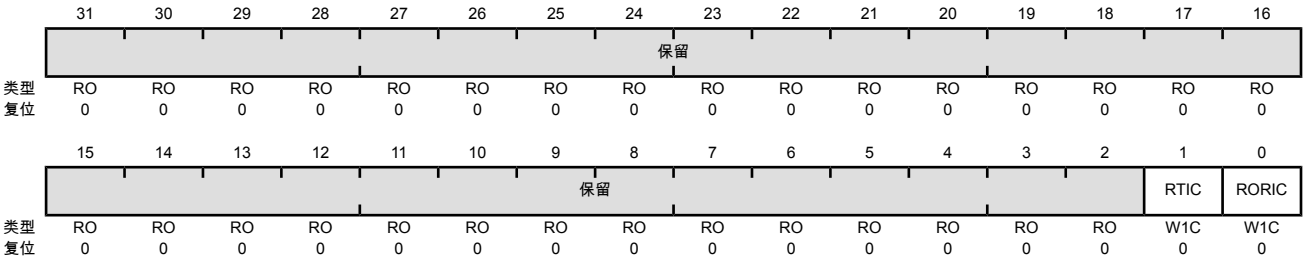
位/域	名称	类型	复位	描述
0	RORMIS	RO	0	SSI 接收溢出屏蔽中断状态 值 描述 0 中断没有发生或者被屏蔽。 1 接收 FIFO 溢出产生中断。 当对 SSI 中断清除寄存器 (SSIICR) 的 RORIC 位写入 1 时，该位清零。

寄存器 9: SSI 中断清除寄存器（SSIICR），偏移量 0x020

SSIICR 寄存器为中断清除寄存器。当对本寄存器的有效位写1时，即会清除相应的中断。写入 0 不起任何作用。

SSI 中断清除寄存器 (SSIICR)

SSI0 基址: 0x4000.8000
SSI1 基址: 0x4000.9000
SSI2 基址: 0x4000.A000
SSI3 基址: 0x4000.B000
偏移量 0x020
类型 W1C, 复位 0x0000.0000



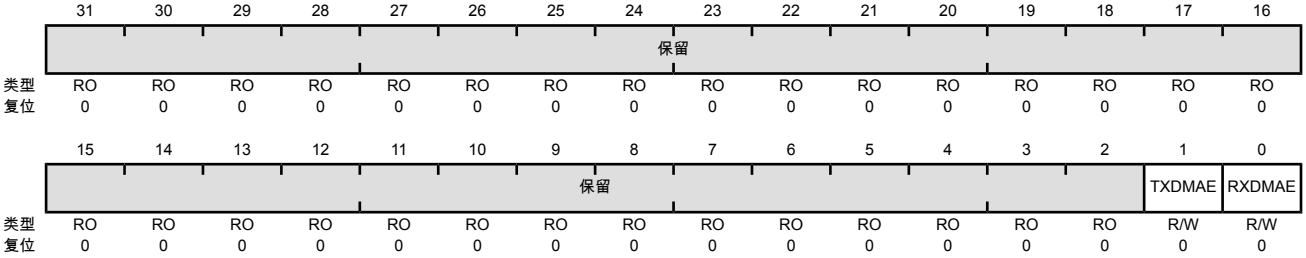
位/域	名称	类型	复位	描述
31:2	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
1	RTIC	W1C	0	SSI接收超时中断清除 对此位写 1，SSIRIS 寄存器的 RTRIS 标志位以及 SSIMIS 寄存器的 RTMIS 标志位将清零。
0	RORIC	W1C	0	SSI 接收溢出中断清除 向该位写入 1 会将 SSIRIS 寄存器的 RORRIS 位和 SSIMIS 寄存器的 RORMIS 位清零。

寄存器 10: SSI DMA 控制寄存器（SSIDMACTL），偏移量 0x024

SSIDMACTL 寄存器为 µDMA 控制寄存器。

SSI DMA 控制寄存器 (SSIDMACTL)

SSI0 基址: 0x4000.8000
SSI1 基址: 0x4000.9000
SSI2 基址: 0x4000.A000
SSI3 基址: 0x4000.B000
偏移量 0x024
类型 R/W, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:2	保留	RO	0x0000.000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
1	TXDMAE	R/W	0	发送DMA使能 值 描述 0 禁用发送FIFO的µDMA通道 1 使能发送FIFO的µDMA通道
0	RXDMAE	R/W	0	接收 DMA 启用 值 描述 0 禁用接收 FIFO 的 µDMA 通道 1 启用接收 FIFO 的 µDMA 通道

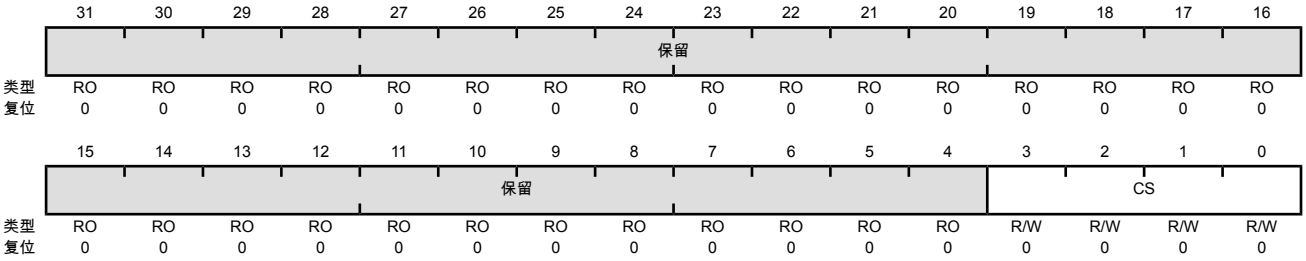
寄存器 11: SSI 时钟配置寄存器（SSICC），偏移量 0xFC8

SSICC 寄存器控制 SSI 模块的波特时钟源。

注意: 如果 PIOSC 用于 SSI 波特时钟，那么在运行模式下系统时钟频率至少为 16 MHz。

SSI 时钟配置寄存器 (SSICC)

SSI0 基址: 0x4000.8000
SSI1 基址: 0x4000.9000
SSI2 基址: 0x4000.A000
SSI3 基址: 0x4000.B000
偏移量 0xFC8
类型 R/W, 复位 0x0000.0000



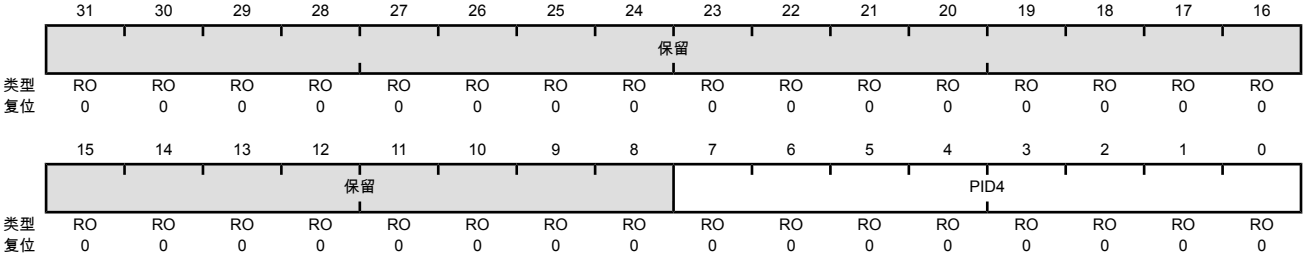
位/域	名称	类型	复位	描述
31:4	保留	RO	0x0000.000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
3:0	CS	R/W	0	SSI 波特时钟源 下面的表格指定了产生 SSI 波特时钟的源： 值 描述 0x0 系统时钟（基于时钟源和分频因子） 0x1-0x4 保留 0x5 PIOSC 0x6 - 0xF 保留

寄存器 12: SSI 外设标识寄存器 4（SSIPeriphID4），偏移量 0xFD0

SSIPeriphIDn 寄存器均为硬编码的只读寄存器，寄存器的位域决定复位值。

SSI 外设标识寄存器 4 (SSIPeriphID4)

SSI0 基址: 0x4000.8000
SSI1 基址: 0x4000.9000
SSI2 基址: 0x4000.A000
SSI3 基址: 0x4000.B000
偏移量 0xFD0
类型 RO, 复位 0x0000.0000



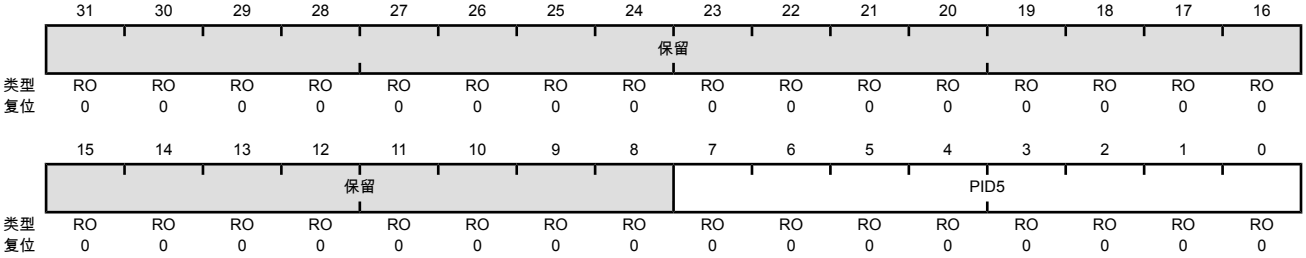
位/域	名称	类型	复位	描述
31:8	保留	RO	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
7:0	PID4	RO	0x00	SSI 外设 ID 寄存器 [7:0] 可被软件用来标识该外设的存在与否。

寄存器 13: SSI 外设标识寄存器 5（SSIPeriphID5），偏移量 0xFD4

SSIPeriphIDn 寄存器均为硬编码的只读寄存器，寄存器的位域决定复位值。

SSI 外设标识寄存器 5 (SSIPeriphID5)

SSI0 基址: 0x4000.8000
SSI1 基址: 0x4000.9000
SSI2 基址: 0x4000.A000
SSI3 基址: 0x4000.B000
偏移量 0xFD4
类型 RO, 复位 0x0000.0000



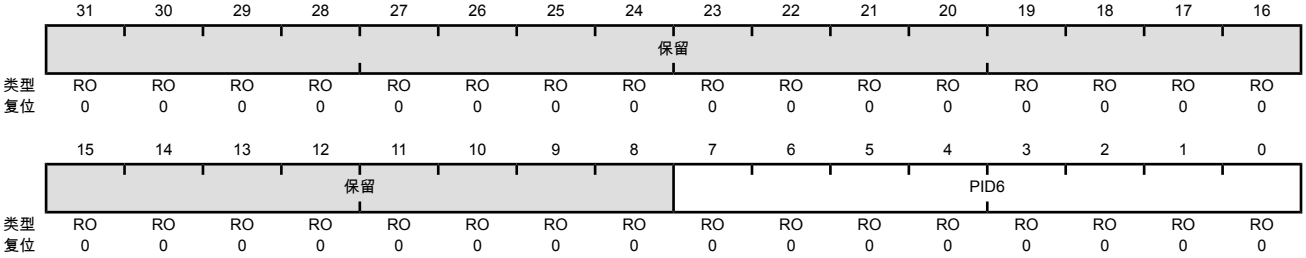
位/域	名称	类型	复位	描述
31:8	保留	RO	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
7:0	PID5	RO	0x00	SSI 外设 ID 寄存器 [15:8] 可被软件用来标识该外设的存在与否。

寄存器 14: SSI 外设标识寄存器 6（SSIPeriphID6），偏移量 0xFD8

SSIPeriphIDn 寄存器均为硬编码的只读寄存器，寄存器的位域决定复位值。

SSI 外设标识寄存器 6 (SSIPeriphID6)

SSI0 基址: 0x4000.8000
SSI1 基址: 0x4000.9000
SSI2 基址: 0x4000.A000
SSI3 基址: 0x4000.B000
偏移量 0xFD8
类型 RO, 复位 0x0000.0000



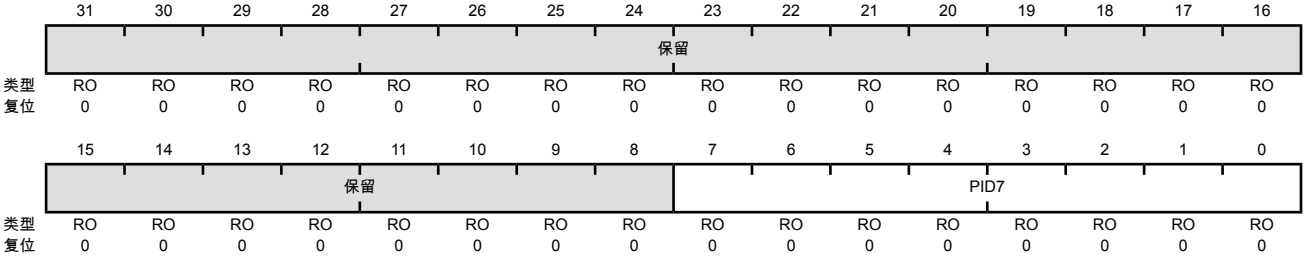
位/域	名称	类型	复位	描述
31:8	保留	RO	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
7:0	PID6	RO	0x00	SSI 外设 ID 寄存器 [23:16] 可被软件用来标识该外设的存在与否。

寄存器 15: SSI 外设标识寄存器 7（SSIPeriphID7），偏移量 0xFDC

SSIPeriphIDn 寄存器均为硬编码的只读寄存器，寄存器的位域决定复位值。

SSI 外设标识寄存器 7 (SSIPeriphID7)

SSI0 基址: 0x4000.8000
SSI1 基址: 0x4000.9000
SSI2 基址: 0x4000.A000
SSI3 基址: 0x4000.B000
偏移量 0xFDC
类型 RO, 复位 0x0000.0000



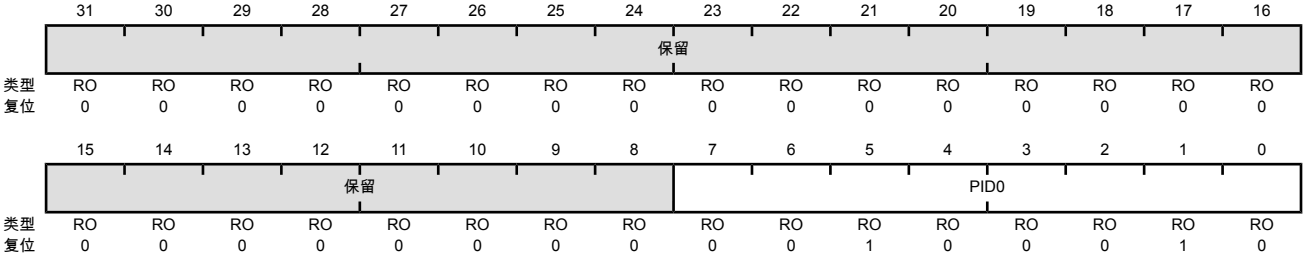
位/域	名称	类型	复位	描述
31:8	保留	RO	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
7:0	PID7	RO	0x00	SSI 外设 ID 寄存器 [31:24] 可被软件用来标识该外设的存在与否。

寄存器 16: SSI 外设标识寄存器 0（SSIPeriphID0），偏移量 0xFE0

SSIPeriphIDn 寄存器均为硬编码的只读寄存器，寄存器的位域决定复位值。

SSI 外设标识寄存器 0 (SSIPeriphID0)

SSI0 基址: 0x4000.8000
SSI1 基址: 0x4000.9000
SSI2 基址: 0x4000.A000
SSI3 基址: 0x4000.B000
偏移量 0xFE0
类型 RO, 复位 0x0000.0022



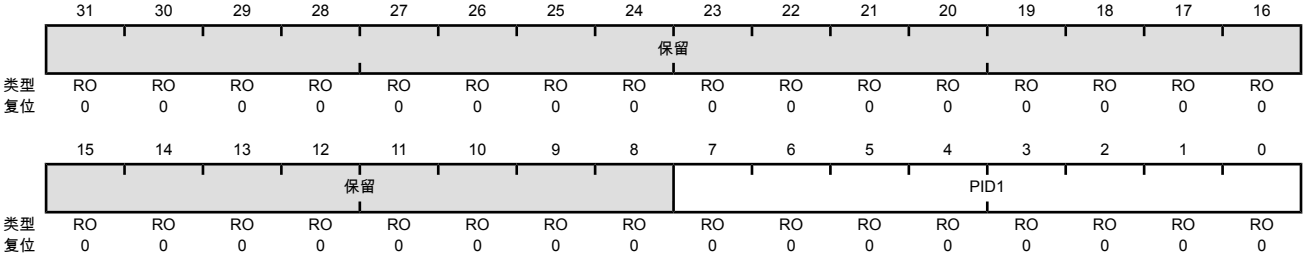
位/域	名称	类型	复位	描述
31:8	保留	RO	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
7:0	PID0	RO	0x22	SSI 外设 ID 寄存器 [7:0] 可被软件用来标识该外设的存在与否。

寄存器 17: SSI 外设标识寄存器 1（SSIPeriphID1），偏移量 0xFE4

SSIPeriphIDn 寄存器均为硬编码的只读寄存器，寄存器的位域决定复位值。

SSI 外设标识寄存器 1 (SSIPeriphID1)

SSI0 基址: 0x4000.8000
SSI1 基址: 0x4000.9000
SSI2 基址: 0x4000.A000
SSI3 基址: 0x4000.B000
偏移量 0xFE4
类型 RO, 复位 0x0000.0000



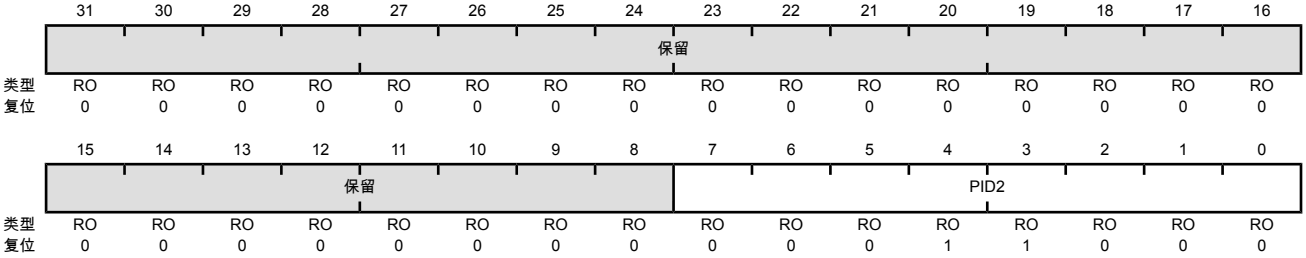
位/域	名称	类型	复位	描述
31:8	保留	RO	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
7:0	PID1	RO	0x00	SSI 外设 ID 寄存器 [15:8] 可被软件用来标识该外设的存在与否。

寄存器 18: SSI 外设标识寄存器 2（SSIPeriphID2），偏移量 0xFE8

SSIPeriphIDn 寄存器均为硬编码的只读寄存器，寄存器的位域决定复位值。

SSI 外设标识寄存器 2 (SSIPeriphID2)

SSI0 基址: 0x4000.8000
SSI1 基址: 0x4000.9000
SSI2 基址: 0x4000.A000
SSI3 基址: 0x4000.B000
偏移量 0xFE8
类型 RO, 复位 0x0000.0018



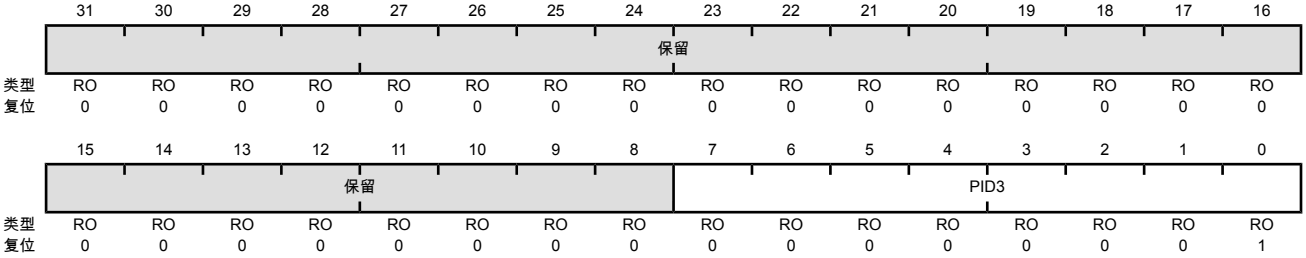
位/域	名称	类型	复位	描述
31:8	保留	RO	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
7:0	PID2	RO	0x18	SSI 外设 ID 寄存器 [23:16] 可被软件用来标识该外设的存在与否。

寄存器 19: SSI 外设标识寄存器 3（SSIPeriphID3），偏移量 0xFEC

SSIPeriphIDn 寄存器均为硬编码的只读寄存器，寄存器的位域决定复位值。

SSI 外设标识寄存器 3 (SSIPeriphID3)

SSI0 基址: 0x4000.8000
SSI1 基址: 0x4000.9000
SSI2 基址: 0x4000.A000
SSI3 基址: 0x4000.B000
偏移量 0xFEC
类型 RO, 复位 0x0000.0001



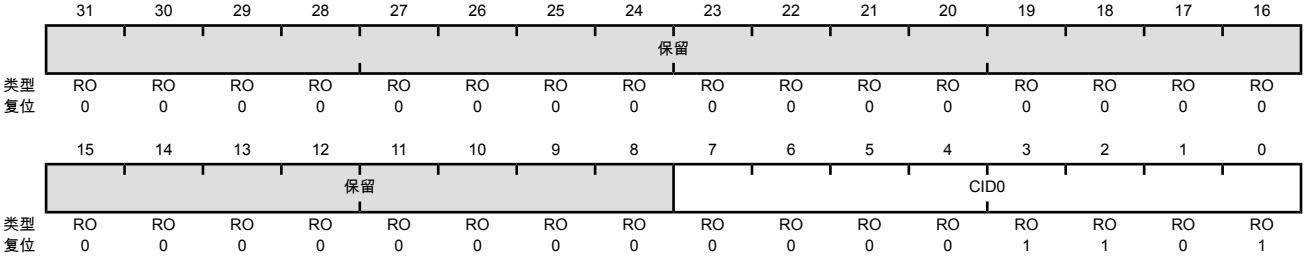
位/域	名称	类型	复位	描述
31:8	保留	RO	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
7:0	PID3	RO	0x01	SSI 外设 ID 寄存器 [31:24] 可被软件用来标识该外设的存在与否。

寄存器 20: SSI PrimeCell 标识寄存器 0（SSIPCellID0），偏移量 0xFF0

SSIPCellIDn 寄存器均为硬编码的只读寄存器，寄存器的位域决定复位值。

SSI PrimeCell 标识寄存器 0 (SSIPCellID0)

SSI0 基址: 0x4000.8000
SSI1 基址: 0x4000.9000
SSI2 基址: 0x4000.A000
SSI3 基址: 0x4000.B000
偏移量 0xFF0
类型 RO, 复位 0x0000.000D



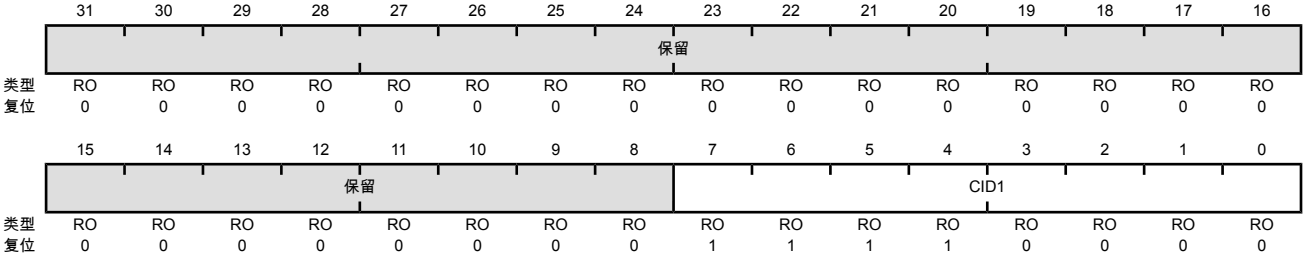
位/域	名称	类型	复位	描述
31:8	保留	RO	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
7:0	CID0	RO	0x0D	SSI PrimeCell 标识寄存器 [7:0] 为软件提供一个标准的交叉外设识别系统。

寄存器 21: SSI PrimeCell 标识寄存器 1（SSIPCellID1），偏移量 0xFF4

SSIPCellIDn 寄存器均为硬编码的只读寄存器，寄存器的位域决定复位值。

SSI PrimeCell 标识寄存器 1 (SSIPCellID1)

SSI0 基址: 0x4000.8000
SSI1 基址: 0x4000.9000
SSI2 基址: 0x4000.A000
SSI3 基址: 0x4000.B000
偏移量 0xFF4
类型 RO, 复位 0x0000.00F0



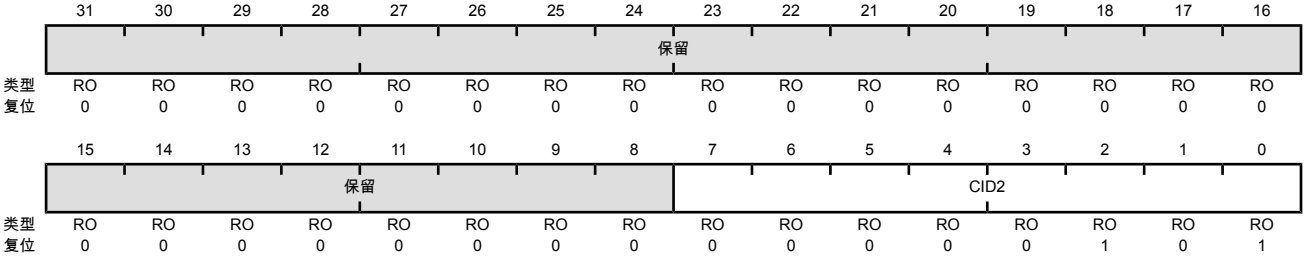
位/域	名称	类型	复位	描述
31:8	保留	RO	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
7:0	CID1	RO	0xF0	SSI PrimeCell 标识寄存器 [15:8] 为软件提供一个标准的交叉外设识别系统。

寄存器 22: SSI PrimeCell 标识寄存器 2（SSIPCellID2），偏移量 0xFF8

SSIPCellIDn 寄存器均为硬编码的只读寄存器，寄存器的位域决定复位值。

SSI PrimeCell 标识寄存器 2 (SSIPCellID2)

SSI0 基址: 0x4000.8000
SSI1 基址: 0x4000.9000
SSI2 基址: 0x4000.A000
SSI3 基址: 0x4000.B000
偏移量 0xFF8
类型 RO, 复位 0x0000.0005



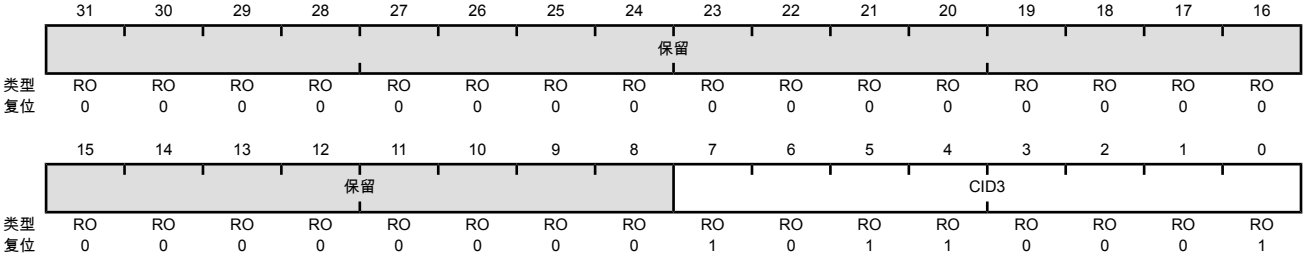
位/域	名称	类型	复位	描述
31:8	保留	RO	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
7:0	CID2	RO	0x05	SSI PrimeCell 标识寄存器 [23:16] 为软件提供一个标准的交叉外设识别系统。

寄存器 23: SSI PrimeCell 标识寄存器 3（SSIPCellID3），偏移量 0xFFC

SSIPCellIDn 寄存器均为硬编码的只读寄存器，寄存器的位域决定复位值。

SSI PrimeCell 标识寄存器 3 (SSIPCellID3)

SSI0 基址: 0x4000.8000
SSI1 基址: 0x4000.9000
SSI2 基址: 0x4000.A000
SSI3 基址: 0x4000.B000
偏移量 0xFFC
类型 RO, 复位 0x0000.00B1



位/域	名称	类型	复位	描述
31:8	保留	RO	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
7:0	CID3	RO	0xB1	SSI PrimeCell 标识寄存器 [31:24] 为软件提供一个标准的交叉外设识别系统。

16 内部集成电路（I²C）接口

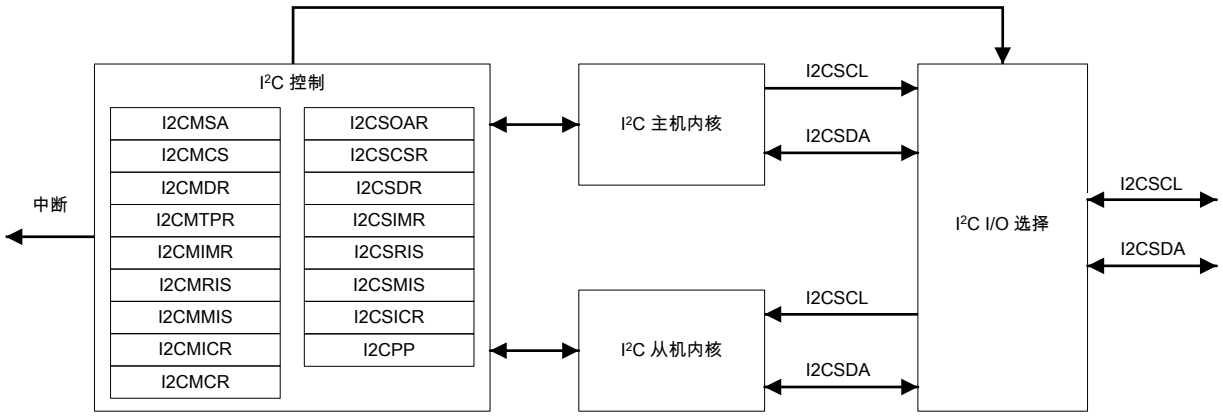
内部集成电路 (I²C) 总线通过双线设计（串行数据线 SDA 和串行时钟线 SCL）提供双向数据传输以及连接外部 I²C 设备的接口，例如串行存储器（RAM 和 ROM）、网络设备、LCD、音频发生器等等。I²C 总线还可在产品开发和制造过程中用于系统测试和诊断。TM4C1231H6PM 微控制器包含能够与总线上的其他 I²C 备进行互动（发送和接收数据）。

TM4C1231H6PM 控制器包含 个 I²C 模块，具有以下特点：

- I²C 总线上的设备可被指定为主机或从机
  - 在主机或从机模式下都支持发送和接受数据
  - 支持它们作为主机和从机的同步操作
- 四种 I²C 模式：
  - 主机发送
  - 主机接收
  - 从机发送
  - 从机接收
- 四种传输速度：
  - 标准 (100 Kbps)
  - 快速 (400 Kbps)
  - 超快速 (1 Kbps)
  - 高速 (3.33 Mbps)
- 时钟低电平超时中断
- 双从机地址功能
- 故障抑制
- 主机和从机产生中断
  - 主机因为传送或接收数据结束(或者是因为错误而取消)产生中断
  - 从机在主机向其发送数据或发出请求时，或检测到START或STOP信号时产生中断。
- 主机带有仲裁和时钟同步功能，支持多主机以及 7 位寻址模式

16.1 结构框图

图 16-1. I²C 结构图



16.2 信号描述

下表列出了 I²C 接口的外部信号及其功能。I²C 接口信号是某些 GPIO 信号的备用功能，复位时将重置为默认的 GPIO 信号，而 I2C0SCL 和 I2CSDA 管脚例外，这两个管脚复位时将重置为默认的 I²C 功能。下表中“复用管脚/赋值”一栏列出了 I²C 信号的 GPIO 管脚的各种可能布局。应将 GPIO 备用功能选择 (GPIOAFSEL) 寄存器（595页）的 AFSEL 置位，以便选择 I²C 功能。必须将括号里的数字写入 GPIO 端口控制 (GPIOPCTL) 寄存器（612页）的 PMCn 域中，以便将 I²C 信号分配给特定的 GPIO 端口管脚。请注意，应通过 GPIO 开漏选择 (GPIOODR) 寄存器将 I2CSDA 管脚设置成开漏。有关如何配置 GPIO 的更多信息，请参阅“通用输入/输入端口（GPIOs）”（575页）。

表 16-1. I2C 信号 (64LQFP)

管脚名称	管脚编号	管脚复用/管脚赋值	管脚类型	缓冲区类型[a]	描述
I2C0SCL	47	PB2 (3)	I/O	OD	I²C 模块 0 时钟。请注意该信号具有有源上拉。不应将相应的端口管脚配置为开漏。
I2C0SDA	48	PB3 (3)	I/O	OD	I²C 模块 0 数据。
I2C1SCL	23	PA6 (3)	I/O	OD	I²C 模块 1 时钟。请注意该信号具有有源上拉。不应将相应的端口管脚配置为开漏。
I2C1SDA	24	PA7 (3)	I/O	OD	I²C 模块 1 数据。
I2C2SCL	59	PE4 (3)	I/O	OD	I²C 模块 2 时钟。请注意该信号具有有源上拉。不应将相应的端口管脚配置为开漏。
I2C2SDA	60	PE5 (3)	I/O	OD	I²C 模块 2 数据。
I2C3SCL	61	PD0 (3)	I/O	OD	I²C 模块 3 时钟。请注意该信号具有有源上拉。不应将相应的端口管脚配置为开漏。
I2C3SDA	62	PD1 (3)	I/O	OD	I²C 模块 3 数据。

a. TTL 表示管脚的电压水平与 TTL 一致。

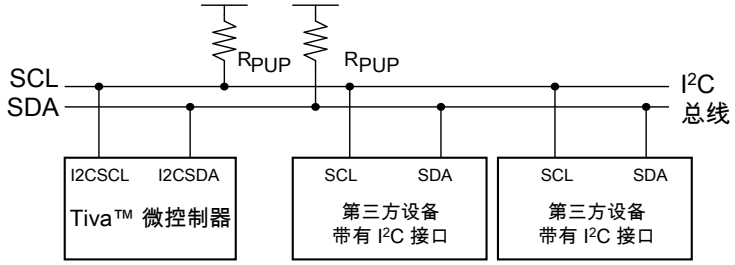
16.3 功能说明

每个 I²C 模块由主机和从机两个功能组成，并通过唯一地址进行标识。主机发起的通信会产生时钟信号SCL。为了实现正确操作，SDA管脚必须配置为开漏信号。由于内部电路支持高速操作，SCL管脚不得配置为开漏信号，即使内部电路可导致其发挥开漏信号的作用。SDA 和 SCL 信号必须使

用上拉电阻连接到正向电源电压。典型的 I2C 总线配置请参阅图16-2。有关如何确定正确操作所需的上拉大小，请参考 “I2C 总线规范和用户手册”。

I2C 时序结构图请参阅“Inter-Integrated Circuit (I2C) Interface”（1064页）。

图 **16-2. I2C** 总线配置



16.3.1 I2C 总线功能概览

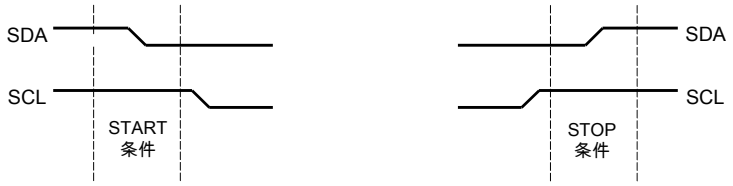
I2C 总线只使用两个信号：SDA 和 SCL。在 I2CSDA 微控制器中，它们分别对应 I2CSCL 和 TM4C1231H6PM。SDA是双向的串行数据线，SCL是双向的串行时钟线。当两根线都处于高电平的时候，总线处于空闲状态。

I2C 总线每次传输的数据长度为九位，其中包括八位数据位和一位应答位。每次传输的字节数（定义为有效 START 和 STOP 条件之间的时间，请参阅“START 和 STOP 条件”（901页））没有限制，但是每个数据字节后面必须紧跟一位应答位，而且数据传输时必须首先传送最高有效位 (MSB) 位。当接收器不能完整接收另一个字节时，它可以保持时钟线SCL为低电平，并迫使发送器进入等待状态。当接收器释放了时钟线SCL的时候，数据传输得以继续进行。

16.3.1.1 START 和 STOP 条件

I2C 总线协议定义了两种状态，以便开始和结束数据传输：START 和 STOP。当 SCL 为高电平时，SDA 线由高到低的跳变被定义为 START 信号；当 SCL 为高电平的时候，SDA 线由低到高的跳变被定义为 STOP 信号。总线在 START 条件之后被视为忙状态，在 STOP 条件之后被视为空闲(free)状态。请参阅图16-3。

图 **16-3. START** 和 **STOP** 条件



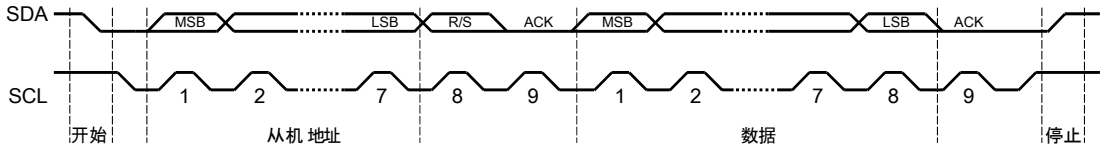
STOP 位决定周期是在数据周期结束时停止，还是继续运行，直到发生重复的 START 条件。要产生单次传输，应在 I2C 主机从机地址 (I2CMSA) 寄存器中写入所需的地址，并将 R/S 位清零，在控制寄存器中写入 ACK= X（0 或 1）、STOP= 1、START=1 以及 RUN= 1，以便执行单次传输并停止。操作完成后（或者因为错误退出），中断管脚将激活，数据可能从 I2C 主机数据 (I2CMDR) 寄存器中读出。I2C 模块以主接收器模式运行时，ACK 位通常会被置位，这会让 I2C 总线控制器在每个字节接收完之后自动发送一个应答。当 I2C 总线控制器无需接收从发送器发送的数据时，该位必须清零。

当此模块以从机模式运行时，I²C 从机原始中断状态 (I2CSRIS) 寄存器中的 STARTRIS 和STOPRIS 位用于监测总线上的开始和停止条件；通过配置 I²C 从机屏蔽中断状态 (I2CSMIS) 寄存器可将 STARTRIS 和 STOPRIS 位转变成控制器中断（前提是启用了中断功能）。

16.3.1.2 带有7位地址的数据格式

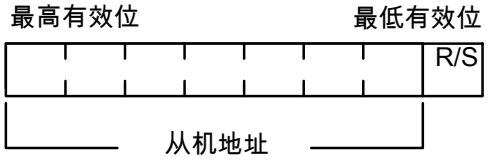
传输数据格式请参阅图16-4。在达到开始条件之后，从机地址将被发送。地址共有 7 位，紧跟着的第 8 位是数据传输方向位（I2CMSA 寄存器的 R/S 位）。R/S 位清零表示传输操作（发送），此位置位表示数据请求（接收）。数据传输总是由主机生成一个停止条件终止的，然而，主机可以在没有产生停止信号的时候，通过再产生一个开始信号和总线上另一个设备的地址，来与另一个设备通信。因此，在一次传输过程中可能会存在各种不同组合的接收/发送格式。

图 **16-4.** 带 **7** 位地址的完整数据传输



第一字节中的前七位即构成从机地址（请参阅图16-5）。第八位确定报文的方向。R/S位的值为0意味着主机将会传输(发送)数据给选定的从机，如果该位的值为1则表明主机将要从从机那接收数据。

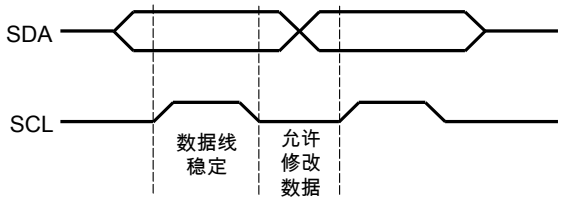
图 **16-5.** 首字节的**R/S**位



16.3.1.3 数据有效性

SDA 线上的数据在时钟的高电平期间必须稳定，只有在 SCL 为低电平的时候，数据线才能改变（请参阅图16-6）。

图 **16-6.** I²C 总线位传输过程中的数据有效性



16.3.1.4 应答

总线上所有传输都要带有应答时钟周期，该时钟周期由主机产生。发送器(可以是主机或从机) 在应答周期过程中释放SDA线，即SDA为高电平。为了响应传输，接收器必须在应答时钟周期过程中拉低SDA。在应答周期内，接收器发出的数据必须遵循数据有效性要求，请参阅“数据有效性”（902页）。

当从机不能响应从机地址时，从机必须将SDA线保持在高电平状态，使得主机可产生停止条件来中止当前的传输。如果主机在传输过程中用作接收器，那么它有责任应答从机发出的每次传输。由于主机控制着传输中的字节数，因此它通过在最后一个数据字节上不产生应答来向从机发送器指示数据的结束。然后从机发送器必须释放SDA线，以便主机可以产生停止条件或重复起始条件。

如果从机需要提供手动的应答或者否定应答，I2C 从机应答控制 (I2CSACKCTL) 寄存器可以让从机对无效数据或无效指令做出否定应答，或者对有效数据或有效指令做出应答。当启用该功能时，MCU 从机模块的 I2C 时钟会在最后一个数据位之后拉低，直到该寄存器写入指定响应。

16.3.1.5 重复开始

I2C 主机模块能够在发生初次传输后执行重复的 START 序列（发送或接收）。

主机发送的重复开始序列如下所述：

1. 设备处于空闲状态时，主机将从机地址写入 I2CMSA 寄存器，并将 R/S 位配置为所需的传输类型。
2. 相关数据将写入 I2CMDR 寄存器。
3. 当 I2CMCS 寄存器的 BUSY 位为 0 时，主机向 I2CMCS 寄存器写入 0x3，以发起传输。
4. 主机不会产生 STOP 条件，但会将另一个从机地址写入 I2CMSA 寄存器，然后写入 0x3，以发起重复的 START。

主机接收的重复开始序列与上述序列类似：

1. 设备处于空闲状态时，主机将从机地址写入 I2CMSA 寄存器，并将 R/S 位配置为所需的传输类型。
2. 主机读取 I2CMDR 寄存器的数据。
3. 当 I2CMCS 寄存器的 BUSY 位为 0 时，主机向 I2CMCS 寄存器写入 0x3，以发起传输。
4. 主机不会产生 STOP 条件，但会将另一个从机地址写入 I2CMSA 寄存器，然后写入 0x3，以发起重复的 START。

有关重复 START 的详细信息，请参阅图16-12（912页） 和图16-13（913页）。

16.3.1.6 时钟低电平超时 (CLTO)

I2C 从机可以将时钟周期性地拉低，以产生较低的位传输速率，从而延长数据传输。I2C 模块有一个 12 位的可编程计数器，它可以跟踪时钟被拉低了多长时间。通过 I2C 主机时钟低电平超时计数 (I2CMCLKOCNT) 寄存器，该计数器的高 8 位可通过软件编程。低四位值为 0x0，用户无法查看。写入 I2CMCLKOCNT 寄存器中 CNTL 的值必须大于 0x01。应用程序能对计数器最高的 8 位进行编程，以反映可接受的传输累计低电平时间。该计数在 START 条件时加载，并且在主机内部总线时钟的每个下降沿进行递减计数。注意：即使总线上的 SCL 被拉低，为此计数器生成的内部总线时钟将一直按编程的 I2C 速度运行。达到终端计数时，主机状态机在 SCL 和 SDA 释放时通过发布 STOP 条件，在总线上强制执行 ABORT。

例如，如果一个 I2C 模块正按 100 kHz 速度运行，由于低四位值为 0x0，将 I2CMCLKOCNT 寄存器编程为 0xDA 会让该值转换为 0xDA0。换言之，也就是 3488 个时钟周期，即在 100 kHz 频率下，时钟低电平累积时间为 34.88 ms。

当到达时钟超时期限时， I2C 主机原始中断状态 (I2CMRIS) 寄存器中的 CLKRIS 位被置位，以便让主机开始纠正措施，解决远程从机状态问题。另外，I2C 主机控制/状态 (I2CMCS) 寄存器中的 CLKTO 位将被置位；在发送 STOP 条件时或者 I2C 主机复位期间，该位将被清零。软件可以读取 I2C 主机总线监视 (I2CMBMON) 寄存器中的 SDA 和 SCL 位以获得 SDA 和 SCL 信号的原始状态，从而帮助确定远程从机的状态。

发生 CLTO 条件时，应用软件必须选择如何尝试恢复总线。大多数应用程序可能会尝试手动切换 I2C 管脚，以强制从机释放时钟信号（另外一种常用的解决方案是强制总线 STOP）。如果在猝发传

输结束前检测到 CLTO，而且主机成功恢复了总线，那么主机硬件将尝试完成挂起的猝发操作。总线上的实际操作取决于总线恢复后的从机状态。如果从机重新进入能够应答主机的状态（基本上总线挂起之前的状态），则会从之前停止的位置继续运行。但是如果从机重新进入复位状态（或者由于主机发出强制 STOP，导致从机进入空闲状态），它可能忽略主机完成猝发操作的尝试，同时 NAK 主机发送或请求的第一个数据字节。

由于从机的操作无法始终准确预测，建议应用程序软件在 CLTO 中断服务例程期间始终写入 I²C 主机配置 (I2CMCR) 寄存器的 STOP 位。这一设置可以将总线恢复后主机接收或发送的数据量限制为单个字节，并且在这个单字节在传输线上时，主机将发出一个 STOP。另一种解决方案是在尝试手动恢复总线之前通过应用程序软件将 I²C 外设复位。这种解决方案能够让 I²C 主机硬件在尝试恢复卡滞总线前重新进入已知的良好（以及空闲）状态，并防止线上意外出现数据。

注意: 主机时钟低电平超时计数器会计算 SCL 被持续拉低的所有时间。如果 SCL 在任何时候失效，主机时钟低电平超时计数器将重新加载 I2CMCLKOCNT 寄存器中的值，并从此值开始递减计数。

16.3.1.7 双地址

I²C 接口支持从机双地址功能。系统提供额外的可编程地址，启用后也可以进行地址匹配。在传统模式中，双地址功能将被禁用，如果地址与 I2CSOAR 寄存器中的 OAR 域相匹配，I²C 从机会在总线上提供应答。在双地址模式下，如果 I2CSOAR 寄存器中的 OAR 域或者 I2CSOAR2 寄存器中的 OAR2 域匹配，I²C 从机会在总线上提供应答。双地址功能通过对 I2CSOAR2 寄存器中的 OAR2EN 位进行编程而启用，且传统地址不会被禁用。

I2CSCSR 寄存器中的 OAR2SEL 位可以显示出应答地址是否是复用地址。该位被清零时，表示处于传统操作，或者无地址匹配。

16.3.1.8 仲裁

只有在总线空闲时，主机才可以启动传输。在 START 条件的最少保持时间内，两个或两个以上的主机都有可能产生 START 条件。在这些情况下，当SCL为高电平时仲裁机制在SDA线上产生。在仲裁过程中，第一个竞争的主机在 SDA 上设置 1（高电平），而另一个主机发送 0（低电平），前者将关闭其数据输出阶段并退出，直至总线再次空闲。

仲裁可以在多个位上发生。仲裁的第一个阶段是比较地址位，如果两个主机都试图寻址相同的器件，则仲裁继续比较数据位。

16.3.1.9 多主机配置中的故障抑制

使用多主机配置时，可将 I²C 主机配置 (I2CMCR) 寄存器的 GFE 位置位，以为 SCL 和 SDA 线路启用故障抑制，并确保信号值正确。使用 I²C 主机配置 2 (I2CMCR2) 寄存器的 GFPW 位可将滤波器配置为不同的滤波宽度。故障抑制值将以缓冲系统时钟数的形式给出。请注意：故障抑制不为零时，所有信号都将在内部延迟。例如，如果 GFPW 设置为 0x7，则在计算预期处理时间时，应加上 31 个时钟。

16.3.2 可用的速度模式

I²C 总线能够以标准模式 (100 kbps)、快速模式 (400 kbps)、超快模式 (1 Mbps) 或者高速模式 (3.33 Mbps) 运行。所选速度模式必须与总线上的其他 I²C 设备相同。

16.3.2.1 标准、快速和超快模式

通过 I²C 主机定时器周期 (I2CMTPR) 寄存器中的数值可以选择标准、快速和超快模式，其 SCL 频率为标准模式 100 kbps、快速模式 400 kbps 或超快模式 1 Mbps。

I²C 时钟速率取决于以下参数：*CLK_PRD*、*TIMER_PRD*、*SCL_LP* 和 *SCL_HP*，其中：

CLK_PRD 是系统时钟周期

SCL_LP 是 SCL 的低电平相位（固定为 6）

SCL_HP 是 SCL 的高电平相位（固定为 4）

TIMER_PRD 是 I2CMTPR 寄存器的编程值（请参考 926页）。通过替换以下公式中的已知变量并算出 *TIMER_PRD* 值即可得到该值。

I^2C 时钟周期计算方法如下：

SCL_PERIOD = 2 × (1 + *TIMER_PRD*) × (*SCL_LP* + *SCL_HP*) × *CLK_PRD*

例如：

CLK_PRD = 50ns

TIMER_PRD = 2

SCL_LP=6

SCL_HP=4

产生的SCL频率是：

1/*SCL_PERIOD* = 333 Khz

表16-2 给出了不同的系统时钟频率下产生标准、快速和超快模式 SCL 频率的定时器周期的示例。

表 **16-2. I^2C 主机定时器周期与速度模式示例**

系统时钟	定时器周期	标准模式	定时器周期	快速模式	定时器周期	超快模式
4 MHz	0x01	100 Kbps	-	-	-	-
6 MHz	0x02	100 Kbps	-	-	-	-
12.5 MHz	0x06	89 Kbps	0x01	312 Kbps	-	-
16.7 MHz	0x08	93 Kbps	0x02	278 Kbps	-	-
20 MHz	0x09	100 Kbps	0x02	333 Kbps	-	-
25 MHz	0x0C	96.2 Kbps	0x03	312 Kbps	-	-
33 MHz	0x10	97.1 Kbps	0x04	330 Kbps	-	-
40 MHz	0x13	100 Kbps	0x04	400 Kbps	0x01	1000 Kbps
50 MHz	0x18	100 Kbps	0x06	357 Kbps	0x02	833 Kbps
80 MHz	0x27	100 Kbps	0x09	400 Kbps	0x03	1000 Kbps

16.3.2.2 高速模式

TM4C1231H6PM I^2C 外设支持在主机和从机模式下高速运行。高速模式的配置方法是将 I^2C 主机控制/状态 (I2CMCS) 寄存器的 HS 位置位。高速模式将以高位速率传输数据，其占空比为 66.6%/33.3%，但是通信和仲裁将以标准、快速或者超快模式的速度进行，用户可以选择其中一种模式。如果 I2CMCS 寄存器中的 HS 位被置位，那么当前模式的上拉功能将启用。

可以使用以下公式选择时钟周期，但是在这种情况下，SCL_LP=2，SCL_HP=1。

SCL_PERIOD = 2 × (1 + *TIMER_PRD*) × (*SCL_LP* + *SCL_HP*) × *CLK_PRD*

例如：

```
CLK_PRD = 25 ns
TIMER_PRD = 1
SCL_LP=2
SCL_HP=1
```

产生的SCL频率是：

```
1/T = 3.33 Mhz
```

表16-3（906页）给出了高速模式下定时器周期和系统时钟的示例。请注意：必须将 I2CMTPR 寄存器的 HS 位置位，以便在高速模式中使用 TPR 值。

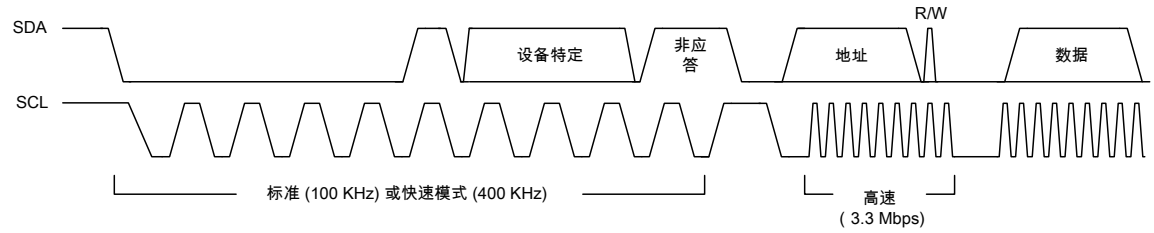
表 **16-3.** 高速模式下 **I²C** 主机定时器周期的示例

系统时钟	定时器周期	传输模式
40 MHz	0x01	3.33 Mbps
50 MHz	0x02	2.77 Mbps
80 MHz	0x03	3.33 Mbps

用作主机时，协议如 图16-7 所示。在开始高速模式传输前，主机负责发送标准模式 (100 Kbps) 或快速模式 (400 Kbps) 的主机代码字节。主机代码字节必须包含 0000.1XXX 格式的数据，并用于告诉从机设备准备进行高速传输。主机代码字节不应发送到从机，而只用于表示即将传输的数据将以更高的数据速率传输。要发送主机代码字节，软件应将主机代码字节的值置入 I2CMSA 寄存器，并将 0x13 写入 I2CMCS 寄存器。这会将 I²C 主机外设置于高速模式，所有之后的传输（直至 STOP 指令）都通过常规的 I2CMCS 命令位在高速数据速率进行，而无需将 I2CMCS 寄存器的 HS 位置位。同样的，仅在主机代码字节中需要将 I2CMCS 寄存器的 HS 位置位。

用作高速从机时，无需使用其他软件。

图 **16-7.** 高速数据格式



注意: 高速模式为 3.4 Mbps，条件是设置了正确的系统时钟频率，且 SCL 和 SDA 线具有适当拉力。

16.3.3 中断信号

I²C 会在发生以下条件时产生中断：

- 主机传输完毕
- 主机仲裁丢失
- 主机发送错误
- 主机总线超时
- 从机接收完成
- 从机请求传输
- 总线检测到结束条件
- 总线检测到开始条件

I²C 主机和 I²C 从机模块具有单独的中断信号。然而两种模式下都能因为多种情况产生中断，但只有一个中断信号进入中断控制器。

16.3.3.1 I²C 主机中断

当通信结束（发送或者接收）、仲裁失败或者通信发生错误时，I²C 主机模块将产生一个中断。要启用 I²C 主机中断，应通过软件将 I²C 主机中断屏蔽 (I2CMIMR) 寄存器中的 IM 位置位。当满足中断条件时，必须通过软件检查 I²C 主机控制/状态 (I2CMCS) 寄存器中的 ERROR 和 ARBLST 位，以验证错误并未发生在最后一个通信期间，以及确保没有输掉仲裁。如果最后的应答信号不是由从机发出，则可断定有错误发生。如果没有检测到错误的发生，并且主机没有丢失仲裁，应用成员就可以处理传输。将 I²C 主机中断清除 (I2CMICR) 寄存器中的 IC 位置位，该中断就会被清除。

如果应用程序无需使用中断，那么通过 I²C 主机原始中断状态 (I2CMRIS) 寄存器可以随时查看原始中断状态。

16.3.3.2 I²C 从机中断

从机模式可以在已接收完数据或是需要从主机接收数据的时候产生中断。将 I²C 从机中断屏蔽 (I2CSIMR) 寄存器中的 DATAIM 位置位即可启用该中断。应通过软件检查 I²C 从机控制/状态 (I2CSCSR) 寄存器中的 RREQ 和 TREQ 位，以确定该模块应该写入（发送）还是读取（接收）来自 I²C 从机数据 (I2CSDR) 寄存器的数据。如果从机模块处于接收状态，并且已经接收到了第一个字节，则 FBR 和 RREQ 位一起置位。将 I²C 从机中断清除 (I2CSICR) 寄存器中的 DATAIC 位置位即可清除该中断。

另外，从机模式时，在检测到开始和结束信号的时候也可以产生中断。要启用这些中断，请将 I²C 从机中断屏蔽 (I2CSIMR) 寄存器中的 STARTIM 和 STOPIM 位置位；要清除这些中断，请将 I²C 从机中断清除 (I2CSICR) 寄存器中的 STOPIC 和 STARTIC 位置位。

如果应用程序无需使用中断，那么通过 I²C 从机原始中断状态 (I2CSRIS) 寄存器可以随时查看原始中断状态。

16.3.4 回送操作

将 I²C 主机配置 (I2CMCR) 寄存器中的 LPBK 位置位即可让 I²C 模块进入内部回送模式，以便进行诊断或者调试工作。在回送模式中，主机的 SDA 和 SCL 信号与从机模块的 SDA 和 SCL 信号绑定，以便在不使用 I/O 接口的情况下对器件进行内部测试。

16.3.5 命令序列流程图

本节描述了主机和从机模式下进行各种类型的 I²C 传输的详细步骤。

16.3.5.1 I²C 主机指令序列

下图显示了 I²C 主机的可用指令序列。

图 **16-8.** 主机单次传输

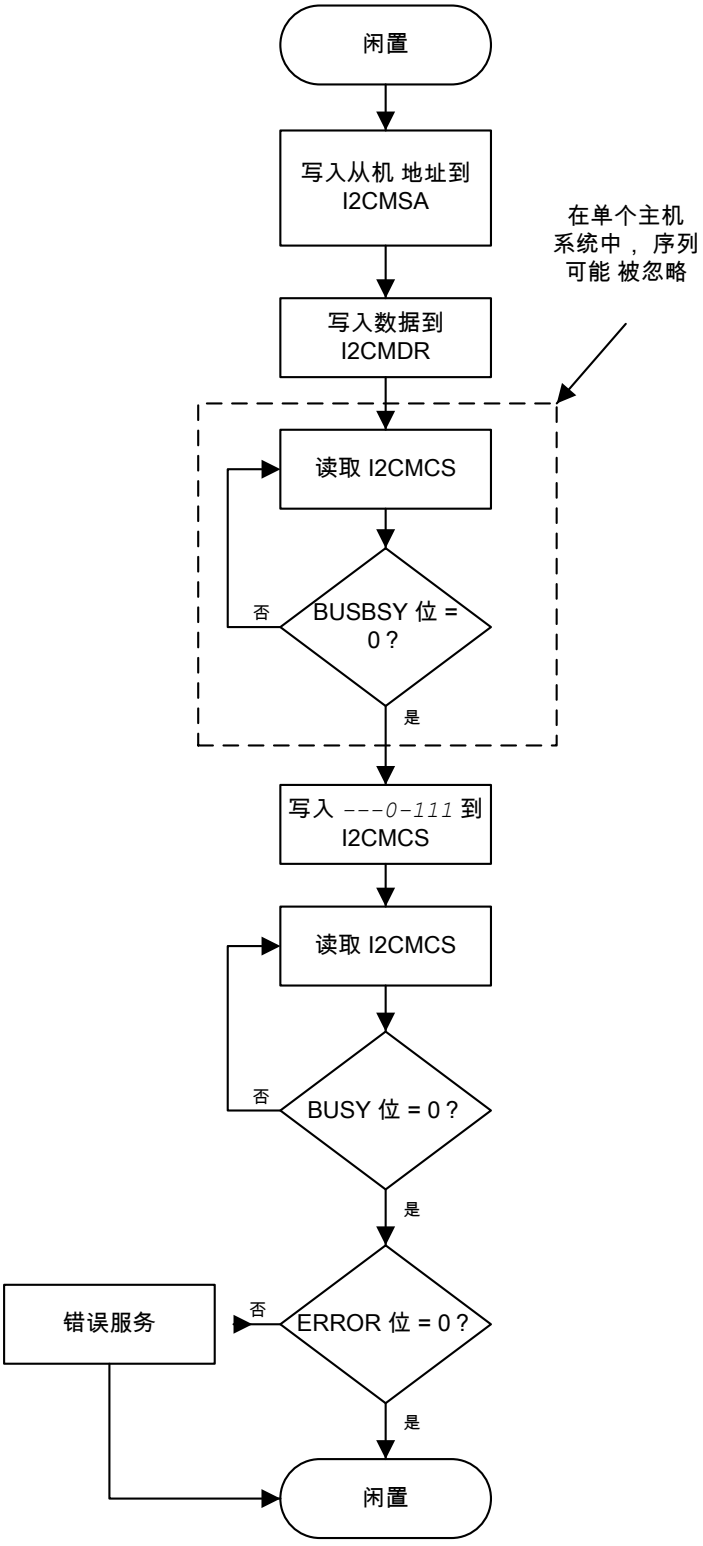

闲置
写入从机 地址到 I2CMSA
在单个主机系统中，序列可能 被忽略
写入数据到 I2CMDR
读取 I2CMCS
否
BUSBSY 位 = 0？
是
写入 ---0-111 到 I2CMCS
读取 I2CMCS
否
BUSY 位 = 0？
是
错误服务
否
ERROR 位 = 0？
是
闲置

图 **16-9.** 主机单次接收

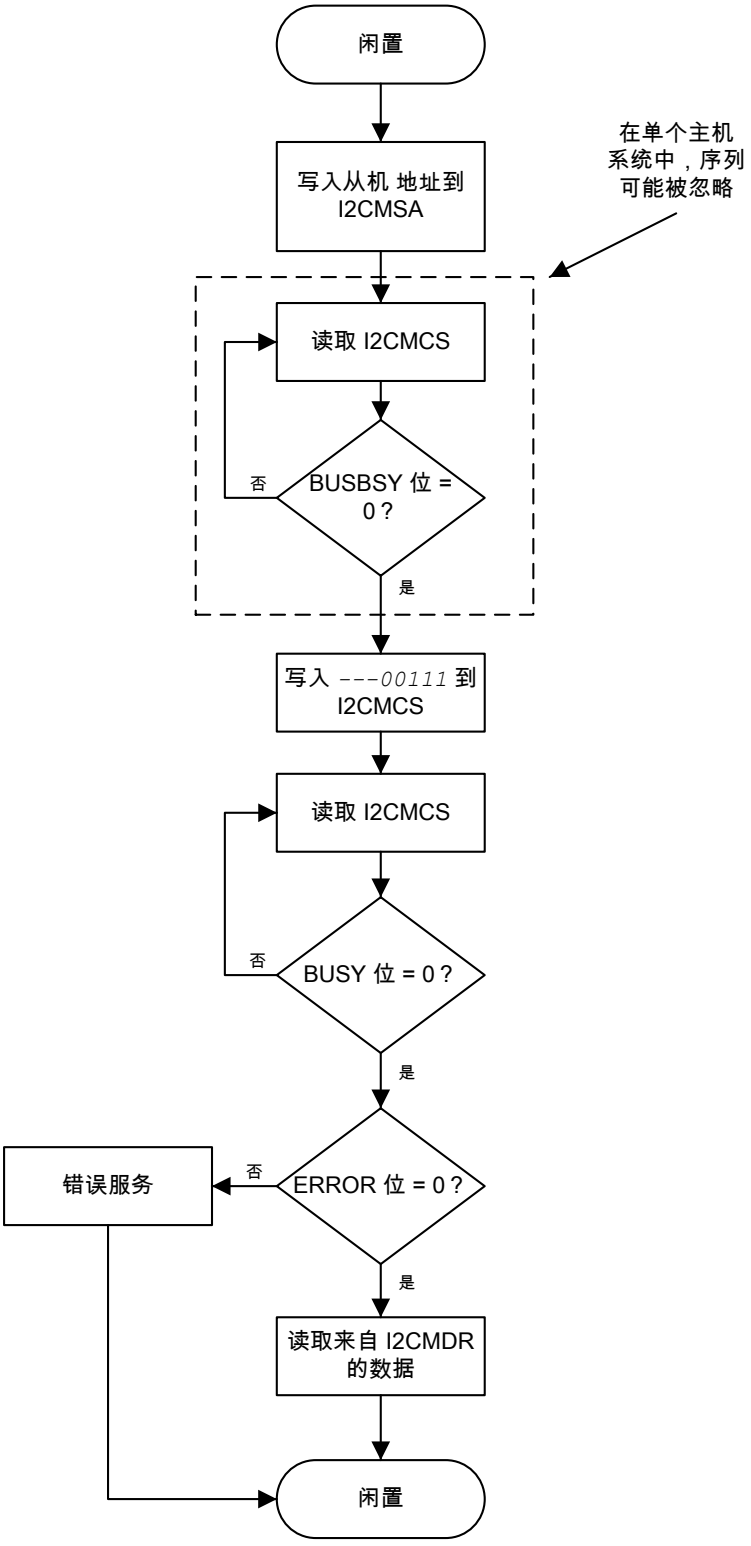

闲置
写入从机 地址到 I2CMSA
在单个主机系统中，序列可能被忽略
读取 I2CMCS
否
BUSBSY 位 = 0？
是
写入 ---00111 到 I2CMCS
读取 I2CMCS
否
BUSY 位 = 0？
是
错误服务
否
ERROR 位 = 0？
是
读取来自 I2CMDR 的数据
闲置

图 16-10. 多数据字节的主机传输

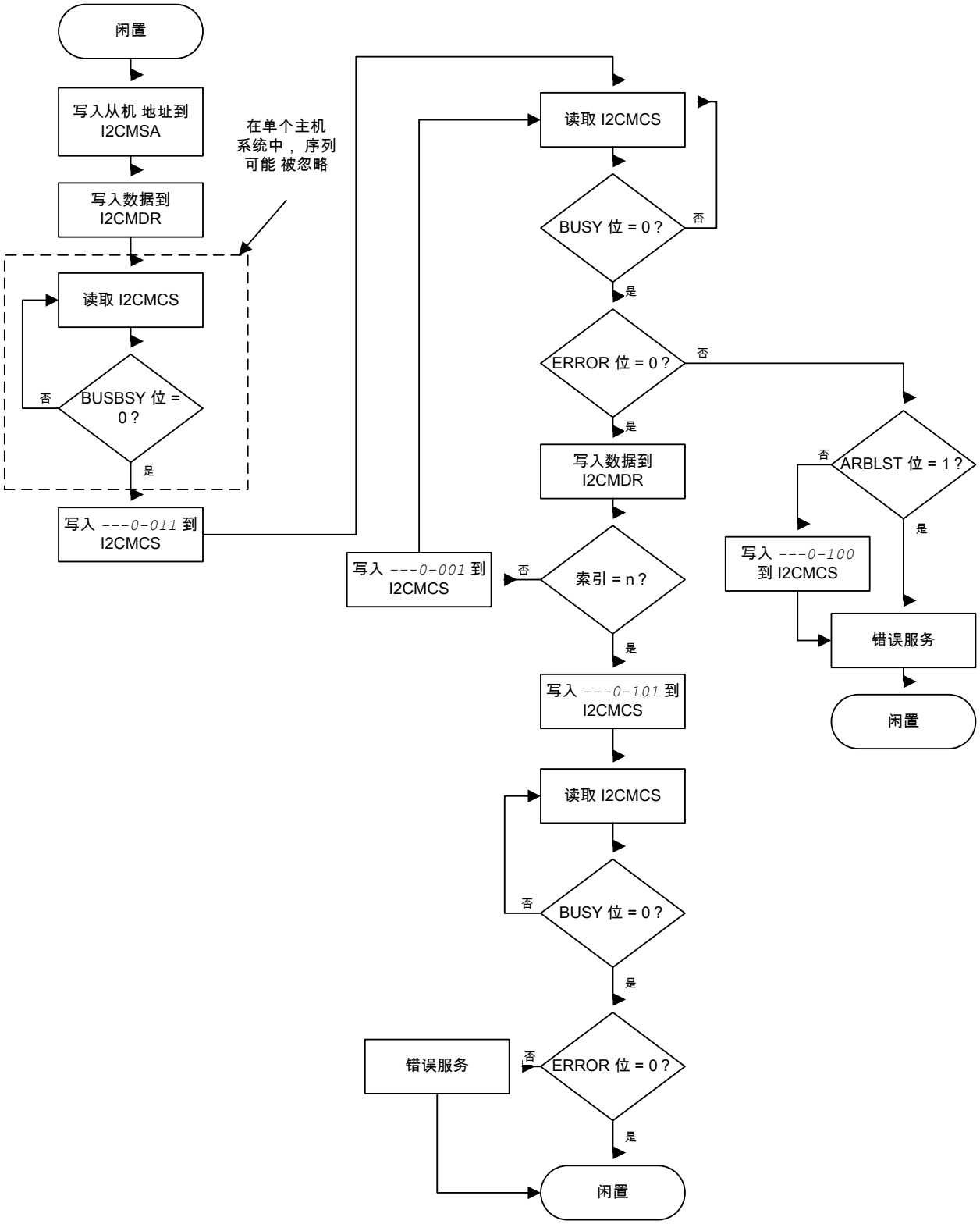


图 16-11. 多数据字节的主机接收

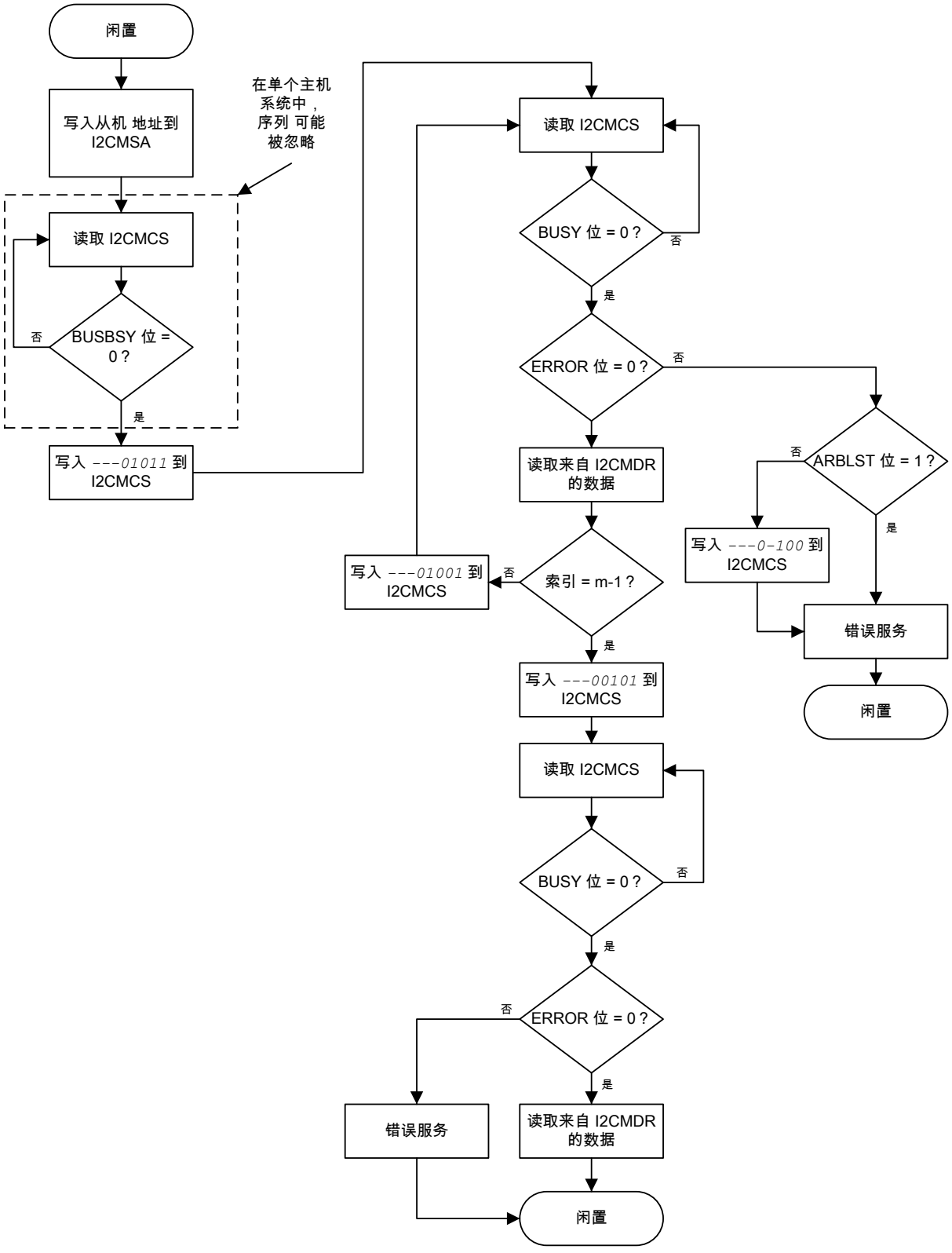


图 **16-12.** 主机传输后以重复开始序列进行的主机接收

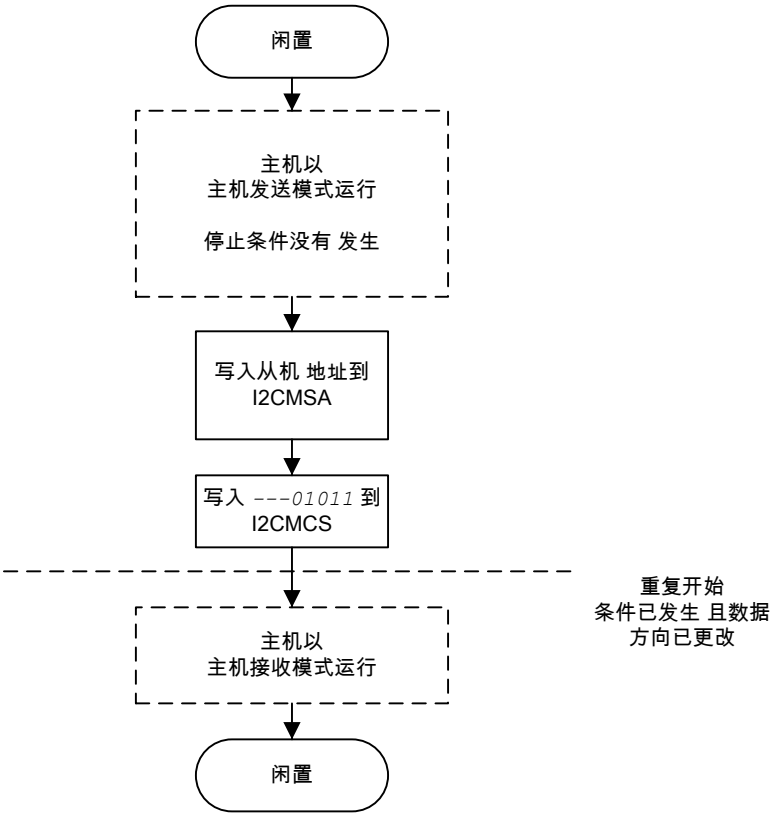


图 16-13. 主机接收后以重复开始序列进行的主机传输

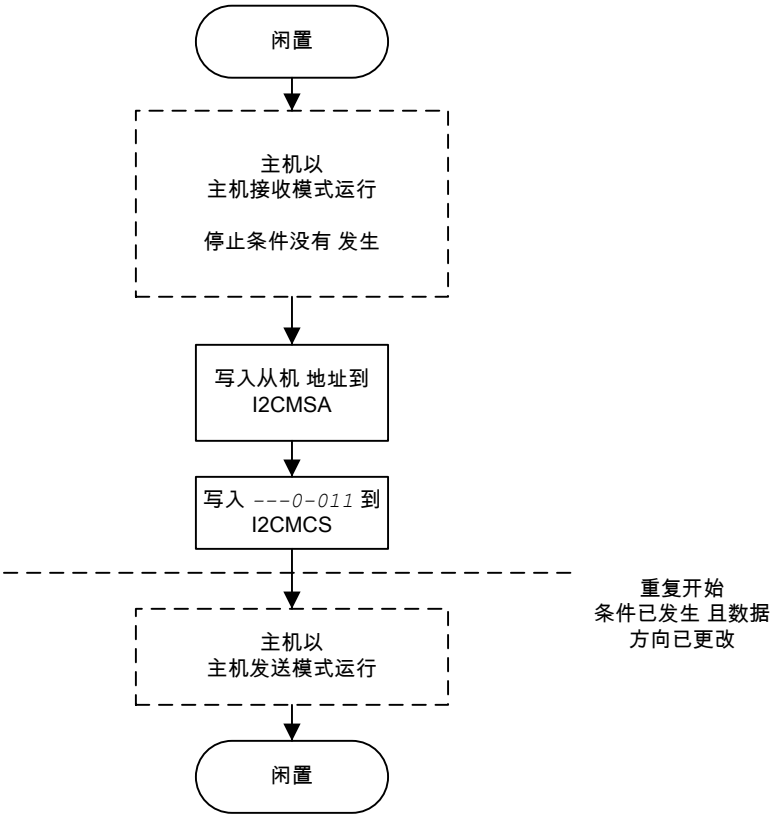
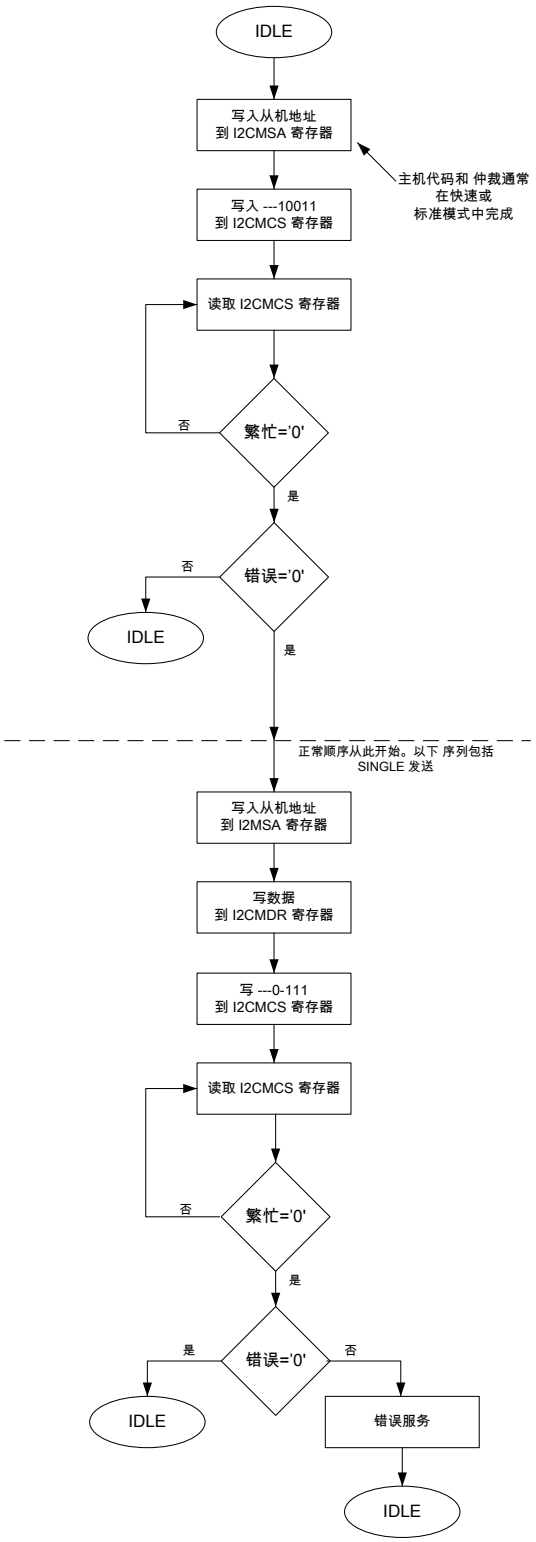


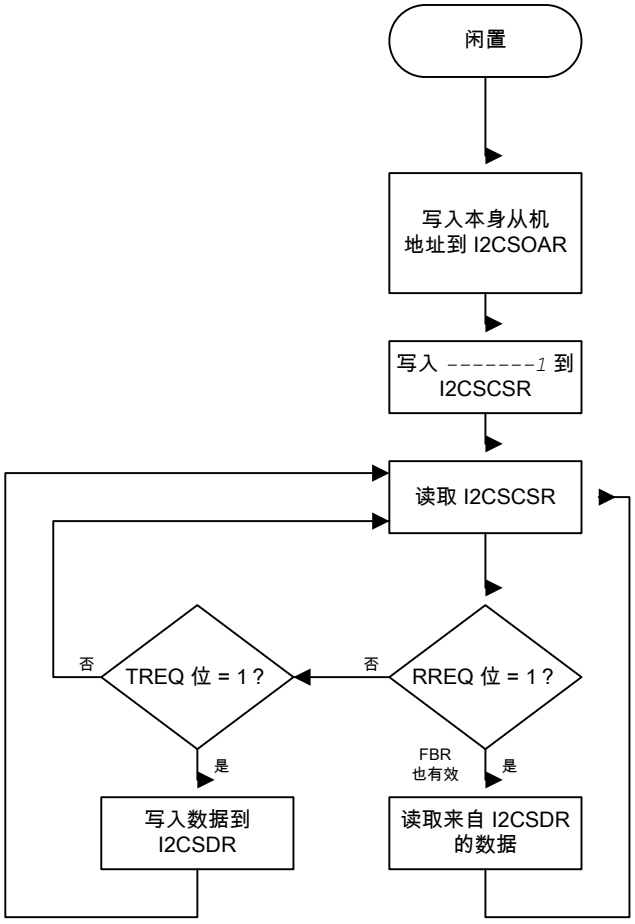
图 16-14. 标准高速模式主机传输



16.3.5.2 I²C 从机指令序列

图16-15（915页）显示了 I²C 从机的可用指令序列。

图 **16-15.** 从机命令序列



16.4 初始化和配置

16.4.1 将 I²C 模块配置为以主机身份传输单字节数据

以下示例显示了如何将 I²C 模块配置为以主机身份传输单字节数据。这里假设系统时钟为20MHz。

1. 用系统控制模块中的 RCGCI2C 寄存器启用 I²C 时钟（请参阅303页）。
2. 用系统控制模块中的 RCGCGPIO 寄存器启用适当的 GPIO 模块时钟（请参阅295页）。要了解要启用哪一个 GPIO 端口，请参考表20-5（1024页）。
3. 在 GPIO 模块中，通过 GPIOAFSEL 寄存器启用相应管脚的复用功能（请参阅 595页）。为了确定哪些 GPIO 需要配置，请参考 表20-4（1019页）。
4. 启用 I2CSDA 管脚以执行开漏操作。请参阅600页。

5. 设置 GPIOPCTL 寄存器中的 PMCn 域，将 I²C 信号分配到适当的管脚。请参阅612页和表20-5（1024页）。
6. 在 I2CMCR 寄存器中写入 0x0000.0010，以初始化 I²C 主机。
7. 通过向 I2CMTPR 寄存器写入正确的值来设置所需的 100 Kbps SCL 时钟速率写入 I2CMTPR 寄存器的值反映了在一个 SCL 时钟周期中系统时钟周期的数目。TPR的值由以下方程决定：

   ```
   TPR = (System Clock/(2*(SCL_LP + SCL_HP)*SCL_CLK))-1;
   TPR = (20MHz/(2*(6+4)*100000))-1;
   TPR = 9
   ```

   向 I2CMTPR 寄存器写入 0x0000.0009。
8. 在 I2CMSA 寄存器中写入 0x0000.0076，以指定从机地址，并指明下一个操作是发送。这会把从机地址设置为 0x3B。
9. 向 I2CMDR 寄存器写入需要传送的数据，以此设置数据寄存器中待发送的数据（字节）。
10. 通过向 I2CMCS 寄存器写入值 0x0000.0007 来开始从主机发送一个字节的数据到从机（STOP、START、RUN）。
11. 等待传输结束（通过轮询 I2CMCS 寄存器的 BUSBSY 位是否已被清零）。
12. 检查 I2CMCS 寄存器中的 ERROR 位，确认传输得到应答。

16.4.2 将 I²C 主机配置为高速模式

将 I²C 主机配置为高速模式的方法：

1. 用系统控制模块中的 RCGCI2C 寄存器启用 I²C 时钟（请参阅303页）。
2. 用系统控制模块中的 RCGCGPIO 寄存器启用适当的 GPIO 模块时钟（请参阅295页）。要了解要启用哪一个 GPIO 端口，请参考表20-5（1024页）。
3. 在 GPIO 模块中，通过 GPIOAFSEL 寄存器启用相应管脚的复用功能（请参阅 595页）。为了确定哪些 GPIO 需要配置，请参考 表20-4（1019页）。
4. 启用 I2CSDA 管脚以执行开漏操作。请参阅600页。
5. 设置 GPIOPCTL 寄存器中的 PMCn 域，将 I²C 信号分配到适当的管脚。请参阅612页和表20-5（1024页）。
6. 在 I2CMCR 寄存器中写入 0x0000.0010，以初始化 I²C 主机。
7. 通过向 I2CMTPR 寄存器写入正确的值来设置所需的 SCL 时钟速率 3.33 Mbps。写入 I2CMTPR 寄存器的值反映了在一个 SCL 时钟周期中系统时钟周期的数目。TPR的值由以下方程决定：

   ```
   TPR = (System Clock/(2*(SCL_LP + SCL_HP)*SCL_CLK))-1;
   TPR = (80 MHz/(2*(2+1)*3330000))-1;
   TPR = 3
   ```

   向 I2CMTPR 寄存器写入 0x0000.0003。
8. 要发送主机代码字节，软件应将主机代码字节的值置入 I2CMSA 寄存器，并将 0x13 写入 I2CMCS 寄存器。

9. 这会将 I2C 主机外设置于高速模式，所有之后的传输（直至 STOP 指令）都通过常规的 I2CMCS 命令位以高速数据速率进行，而无需将 I2CMCS 寄存器的 HS 位置位。

10. 将 I2CMCS 寄存器中的 STOP 位置位，可结束该操作。

11. 等待传输结束（通过轮询 I2CMCS 寄存器的 BUSBSY 位是否已被清零）。

12. 检查 I2CMCS 寄存器中的 ERROR 位，确认传输得到应答。

16.5 寄存器映射

表 16-4（917页）列出了各个 I²C 寄存器。所有给出的地址都相对于 I²C 基础地址：

- I²C 0：0x4002.0000
- I²C 1：0x4002.1000
- I²C 2：0x4002.2000
- I²C 3：0x4002.3000

请注意，在对寄存器编程之前必须启用 I²C 模块的时钟（请参阅303页）。I²C 模块时钟启用之后必须延迟 3 个系统时钟，I²C 模块的寄存器才能访问。

驱动程序库中的 hw_i2c.h 文件 TivaWare™ 将 0x800 作为 I²C 从机寄存器的基础地址。请留意在使用偏移量在 0x800 和 0x818 之间的寄存器时 TivaWare™（适用于 C 系列）将使用从机基础地址，且偏移量介于 0x000 和 0x018 之间。

表 **16-4.** 内部集成电路（**I²C**）接口 寄存器映射

偏移量	名称	类型	复位	描述	见页面
I²C 主机					
0x000	I2CMSA	R/W	0x0000.0000	I2C 主机从机地址寄存器	919
0x004	I2CMCS	R/W	0x0000.0020	I2C 主机控制/状态寄存器	920
0x008	I2CMDR	R/W	0x0000.0000	I2C 主机数据寄存器	925
0x00C	I2CMTPR	R/W	0x0000.0001	I2C 主机定时器周期寄存器	926
0x010	I2CMIMR	R/W	0x0000.0000	I2C 主机中断屏蔽寄存器	927
0x014	I2CMRIS	RO	0x0000.0000	I2C 主机原始中断状态寄存器	928
0x018	I2CMMIS	RO	0x0000.0000	I2C 主机屏蔽中断状态寄存器	929
0x01C	I2CMICR	WO	0x0000.0000	I2C 主机中断清除寄存器	930
0x020	I2CMCR	R/W	0x0000.0000	I2C 主机配置寄存器	931
0x024	I2CMCLKOCNT	R/W	0x0000.0000	I2C 主机时钟低电平超时计数寄存器	932
0x02C	I2CMBMON	RO	0x0000.0003	I2C 主机总线监视寄存器	933
0x038	I2CMCR2	R/W	0x0000.0000	I2C 主机配置 2 寄存器	934
I²C 从机					
0x800	I2CSOAR	R/W	0x0000.0000	I2C 从机本身地址寄存器	935
0x804	I2CSCSR	RO	0x0000.0000	I2C 从机控制/状态寄存器	936
0x808	I2CSDR	R/W	0x0000.0000	I2C 从机数据寄存器	938

表 16-4. 内部集成电路（I²C）接口 寄存器映射（续）

偏移量	名称	类型	复位	描述	见页面
0x80C	I2CSIMR	R/W	0x0000.0000	I2C 从机中断屏蔽	939
0x810	I2CSRIS	RO	0x0000.0000	I2C 从机原始中断状态寄存器	940
0x814	I2CSMIS	RO	0x0000.0000	I2C 从机屏蔽中断状态寄存器	941
0x818	I2CSICR	WO	0x0000.0000	I2C 从机中断清除寄存器	942
0x81C	I2CSOAR2	R/W	0x0000.0000	I2C 从机自身地址寄存器 2	943
0x820	I2CSACKCTL	R/W	0x0000.0000	I2C 从机应答控制寄存器	944
I²C 状态和控制					
0xFC0	I2CPP	RO	0x0000.0001	I2C 外设属性寄存器	945
0xFC4	I2CPC	RO	0x0000.0001	I2C 外设配置寄存器	946

16.6 寄存器描述（I²C 主机）

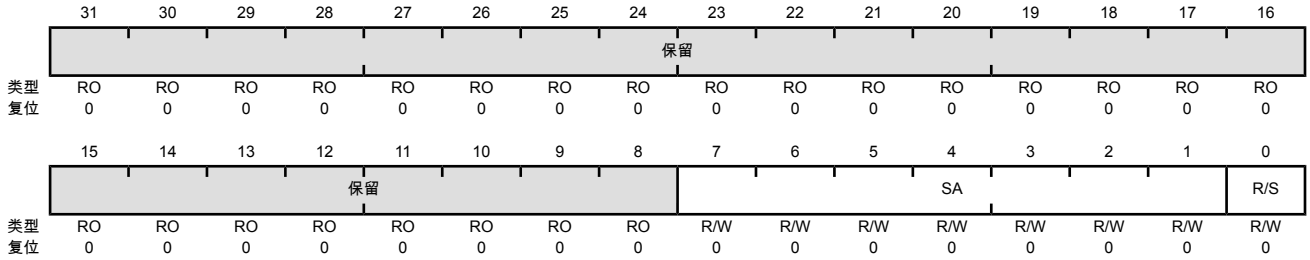
本章的剩余部分按照地址偏移量由小到大的顺序依次详细介绍各 I²C 主机寄存器。

寄存器 1: I²C 主机从机地址寄存器（I2CMSA），偏移量 0x000

该寄存器包含8位：7位地址(A6:A0)和1位接收/发送位。 接收/发送位决定了下一个操作是接收(高)还是发送(低)。

I2C 主机从机地址寄存器 (I2CMSA)

I2C 0 基址: 0x4002.0000
I2C 1 基址: 0x4002.1000
I2C 2 基址: 0x4002.2000
I2C 3 基址: 0x4002.3000
偏移量 0x000
类型 R/W, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:8	保留	RO	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
7:1	SA	R/W	0x00	I²C 从机地址 该字段表示从机地址的A6到A0。
0	R/S	R/W	0	接收/发送 R/S 位可指示下一个操作是接收（高电平）还是发送（低电平）。 值 描述 0 发送 1 接收

寄存器 2: I²C 主机控制/状态寄存器（I2CMCS），偏移量 0x004

该寄存器在读取时访问状态位，在写入时访问控制位。读取时，状态寄存器可指示 I²C 总线控制器的状态。写入时，控制器寄存器可用来设置 I²C 控制器的操作。

START 位用来产生 START 或 REPEATED START 信号。STOP 位决定周期是在数据周期结束时停止，还是继续运行下一个传输周期（可能是重复 START）。要产生单次传输，应在 I²C 主机从机地址 (I2CMSA) 寄存器中写入所需的地址，并将 R/S 位清零，在控制寄存器中写入 ACK= X（0 或 1）、STOP= 1、START=1 以及 RUN= 1，以便执行单次传输并停止。当操作完成时(或者由于错误退出)，中断将激活，数据可能从 I2CMDR 寄存器中读出。I²C 模块以主机接收器模式运行时，ACK 位通常会被置位，这会让 I²C 总线控制器在每个字节接收完成之后自动发送一个应答。当 I²C 总线控制器无需接收从发送器发送的数据时，该位必须清零。

只读状态寄存器

I2C 主机控制/状态寄存器 (I2CMCS)

I2C 0 基址: 0x4002.0000
I2C 1 基址: 0x4002.1000
I2C 2 基址: 0x4002.2000
I2C 3 基址: 0x4002.3000
偏移量 0x004
类型 RO, 复位 0x0000.0020

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
保留															
类型 RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位 0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留								CLKTO	BUSBSY	IDLE	ARBLST	DATACK	ADRACK	ERROR	BUSY
类型 RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位 0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0

位/域	名称	类型	复位	描述
31:8	保留	RO	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
7	CLKTO	RO	0	时钟超时错误 值 描述 0 无时钟超时错误。 1 发生时钟超时错误。 当主机发送停止条件或者 I²C 主机复位时，该位被清零。
6	BUSBSY	RO	0	总线占线 值 描述 0 I²C 总线空闲。 1 I²C 总线占线。 该位根据开始和停止条件变化。

位/域	名称	类型	复位	描述
5	IDLE	RO	1	I^2C 空闲 值 描述 0 I^2C 控制器没有闲置。 1 I^2C 控制器空闲。
4	ARBLST	RO	0	仲裁失败 值 描述 0 I^2C 控制器赢得仲裁。 1 I^2C 控制器输掉仲裁。
3	DATACK	RO	0	应答数据 值 描述 0 发送数据时应答 1 发送数据时不应答
2	ADRACK	RO	0	应答地址 值 描述 0 传输地址得到应答。 1 传输地址没有得到应答。
1	ERROR	RO	0	误差 值 描述 0 在最后的操作中没有检测到错误 1 在最后的操作中已经发生了错误 该错误可能源于从机地址没有得到应答或者传输数据没有得到应答。
0	BUSY	RO	0	I^2C 占线 值 描述 0 控制器空闲。 1 控制器忙。 当 BUSY 置位时，其他的状态位将无效。

只写控制寄存器

I2C 主机控制/状态寄存器 (I2CMCS)

I2C 0 基址: 0x4002.0000
I2C 1 基址: 0x4002.1000
I2C 2 基址: 0x4002.2000
I2C 3 基址: 0x4002.3000
偏移量 0x004
类型 WO, 复位 0x0000.0020

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
域	保留															
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
域	保留											HS	ACK	STOP	START	RUN
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	WO	WO	WO	WO	WO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位/域	名称	类型	复位	描述
31:5	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
4	HS	WO	0	高速使能 值 描述 0 在 I2CMTPR 寄存器中写入相应数值即可选择以标准、快速或者超快模式运行主机。SCL 频率分别是标准模式 100 kbps、快速模式 400 kbps 或超快模式 1 Mpbs。 1 主机以高速模式运行时，传输速率可高达 3.33 Mbps。
3	ACK	WO	0	数据应答允许 值 描述 0 主机在收到数据的时候不自动应答 1 主机会自动应答接收到的数据字节。域解码请参阅表16-5（923页）。
2	STOP	WO	0	产生停止条件 值 描述 0 控制器不生成停止信号 1 控制器产生了停止条件。域解码请参阅表16-5（923页）。
1	START	WO	0	产生开始条件 值 描述 0 控制器不生成开始信号 1 控制器产生或者重复开始条件。域解码请参阅表16-5（923页）。

位/域	名称	类型	复位	描述
0	RUN	WO	0	I^2C 主机启用

值 描述

0 该编码表示主机不可以发送或接收数据。

1 主机可以发送或接收数据。

域解码请参阅表16-5（923页）。

表 **16-5.** 为 **I2CMCS[3:0]** 域写入域解码

当前状态	I2CMSA[0]	I2CMCS[3:0]				描述
	R/S	ACK	STOP	START	RUN	
空闲	0	X[a]	0	1	1	START条件之后跟随SEND(主机进入主机发送状态)
	0	X	1	1	1	START条件之后跟随SEND和STOP条件(主机保持在空闲状态)
	1	0	0	1	1	START条件之后跟随带有非应答的接收操作(主机进入主机接收状态)。
	1	0	1	1	1	START条件之后跟随RECEIVE和STOP条件(主机保持在空闲状态)。
	1	1	0	1	1	START条件之后跟随 RECEIVE(主机进入主机接收状态)。
	1	1	1	1	1	非法
	其余未列出的情况都不执行任何操作					NOP
主机发送	X	X	0	0	1	发送操作(主机仍然在发送状态)
	X	X	1	0	0	STOP条件（主机进入空闲状态）。
	X	X	1	0	1	发送之后跟着是停止条件(主机进入空闲状态)
	0	X	0	1	1	重复START条件之后跟随SEND(主机仍然在发送状态)
	0	X	1	1	1	重复的START条件之后跟随SEND和STOP条件(主机进入空闲状态)。
	1	0	0	1	1	重复START条件之后跟随带有非应答的接收操作（主机进入主机接收状态）。
	1	0	1	1	1	重复START条件之后跟随SEND和STOP条件(主机进入空闲状态)。
	1	1	0	1	1	重复START条件之后跟随RECEIVE（主机进入主机接收状态）。
	1	1	1	1	1	非法。
	其余未列出的情况都不执行任何操作					NOP.

表 16-5. 为 I2CMCS[3:0] 域写入域解码（续）

当前状态	I2CMSA[0]	I2CMCS[3:0]				描述
	R/S	ACK	STOP	START	RUN	
主机接收	X	0	0	0	1	带有非应答的接收操作（主机保持在主机接收状态）。
	X	X	1	0	0	STOP条件（主机进入空闲状态）。[b]
	X	0	1	0	1	RECEIVE 接收之后跟随STOP条件（主机进入空闲状态）。
	X	1	0	0	1	接收操作（主机保持在主机接收状态）。
	X	1	1	0	1	非法。
	1	0	0	1	1	重复的START条件之后跟随带有非应答的接收操作（主机保持在主机接收状态）。
	1	0	1	1	1	重复的START条件之后跟随 RECEIVE和STOP条件（主机进入空闲状态）。
	1	1	0	1	1	重复的START条件之后跟随 RECEIVE（主机保持在主机接收状态）。
	0	X	0	1	1	重复开始条件后紧跟发送操作（主机进入主机发送状态）。
	0	X	1	1	1	重复的START条件之后跟随SEND和STOP条件(主机进入空闲状态)。
	其余未列出的情况都不执行任何操作					NOP.

a. 表格中的X说明该位可以设置成0或1。

b. 在主机接收模式下，STOP条件仅在主机执行数据非应答或从机执行地址非应答后产生。

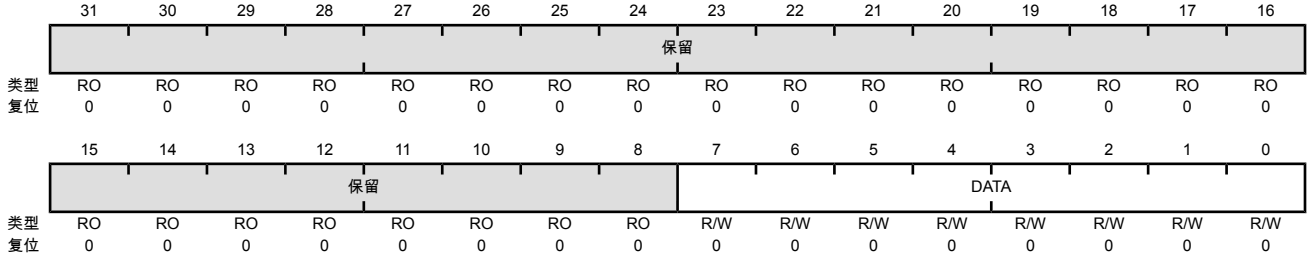
寄存器 3: I²C 主机数据寄存器（I2CMDR），偏移量 0x008

重要: 本寄存器为读敏感型寄存器。有关详细信息，请参阅寄存器描述部分。

在主机发送数据时，该寄存器包含要发送的数据；在主机接收数据时，该寄存器包含接收到的数据。

I2C 主机数据寄存器 (I2CMDR)

I2C 0 基址: 0x4002.0000
I2C 1 基址: 0x4002.1000
I2C 2 基址: 0x4002.2000
I2C 3 基址: 0x4002.3000
偏移量 0x008
类型 R/W, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:8	保留	RO	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
7:0	DATA	R/W	0x00	此字节包含操作期间传输的数据。

寄存器 4: I²C 主机定时器周期寄存器（I2CMTPR），偏移量 0x00C

编程该寄存器以设置 SCL 时钟的定时器周期，并将 SCL 时钟指定为标准或高速模式。

I2C 主机定时器周期寄存器 (I2CMTPR)

I2C 0 基址: 0x4002.0000
I2C 1 基址: 0x4002.1000
I2C 2 基址: 0x4002.2000
I2C 3 基址: 0x4002.3000
偏移量 0x00C
类型 R/W, 复位 0x0000.0001

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	保留															
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	保留								HS	TPR						
类型	RO	RO	RO	RO	RO	RO	RO	RO	WO	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

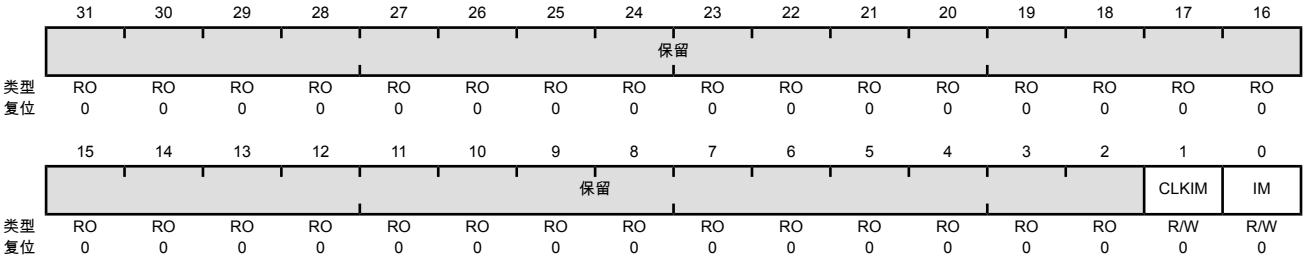
位/域	名称	类型	复位	描述
31:8	保留	RO	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
7	HS	WO	0x0	高速使能 值 描述 0 TPR 设置的 SCL 时钟周期适用于标准模式 (100 Kbps)、快速模式 (400 Kbps) 或超快模式 (1 Mbps)。 1 TPR 设置的 SCL 时钟周期适用于高速模式 (3.33 Mbps)。
6:0	TPR	R/W	0x1	定时器周期 此域将在用于配置 *SCL_PERIOD* 的公式中使用： *SCL_PERIOD* = 2×(1 + TPR)×(*SCL_LP* + *SCL_HP*)×*CLK_PRD* 此处， *SCL_PRD* 是 SCL 线周期（I²C 时钟）。 TPR 是定时器周期寄存器值（范围 1 到 127）。 *SCL_LP* 是 SCL 低电平周期（固定为 6）。 *SCL_HP* 是 SCL 高电平周期（固定为 4）。 *CLK_PRD* 是系统时钟周期，单位 ns。

寄存器 5: I²C 主机中断屏蔽寄存器（I2CMIMR），偏移量 0x010

该寄存器控制是否将原始中断提交到控制器中断。

I2C 主机中断屏蔽寄存器 (I2CMIMR)

I2C 0 基址: 0x4002.0000
I2C 1 基址: 0x4002.1000
I2C 2 基址: 0x4002.2000
I2C 3 基址: 0x4002.3000
偏移量 0x010
类型 R/W, 复位 0x0000.0000



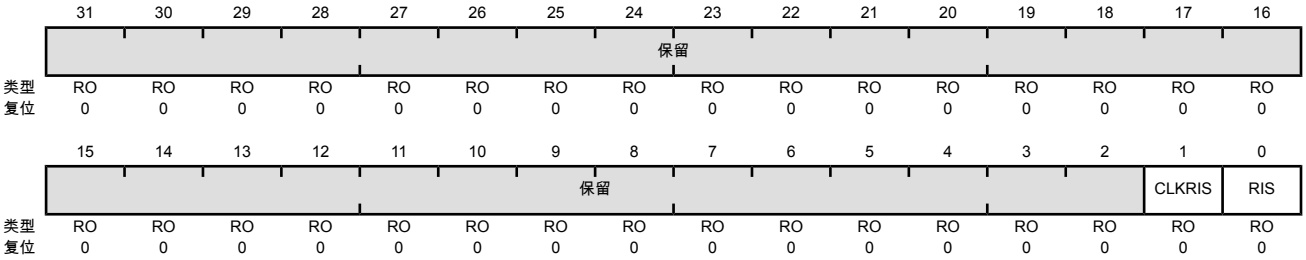
位/域	名称	类型	复位	描述
31:2	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
1	CLKIM	R/W	0	时钟超时中断屏蔽 值 描述 0 CLKRIS 中断被抑制，不会发送到中断控制器。 1 当 I2CMRIS 寄存器中的 CLKRIS 位被置位时，时钟超时中断就会被发送到中断控制器。
0	IM	R/W	0	主机中断屏蔽寄存器 值 描述 0 RIS 中断被抑制，没有发送到中断控制器。 1 当 I2CMRIS 寄存器里的 RIS 位置位时，主机中断发送到中断控制器。

寄存器 6: I²C 主机原始中断状态寄存器（I2CMRIS），偏移量 0x014

该寄存器表示是否有中断正等待处理

I2C 主机原始中断状态寄存器 (I2CMRIS)

I2C 0 基址: 0x4002.0000
I2C 1 基址: 0x4002.1000
I2C 2 基址: 0x4002.2000
I2C 3 基址: 0x4002.3000
偏移量 0x014
类型 RO, 复位 0x0000.0000



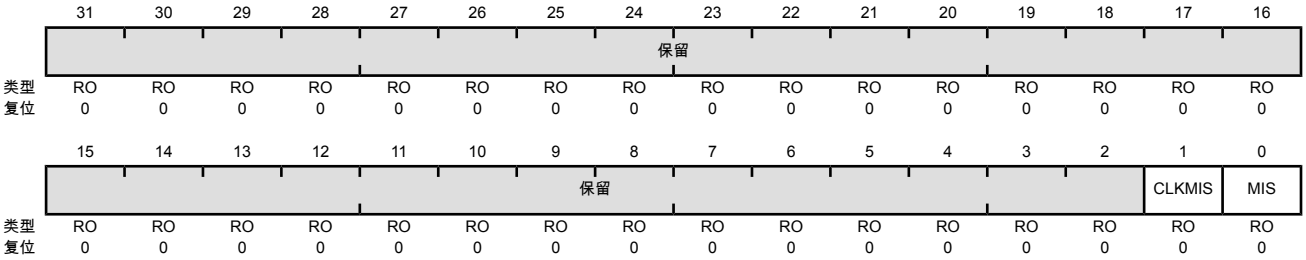
位/域	名称	类型	复位	描述
31:2	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
1	CLKRIS	RO	0	时钟超时原始中断状态 值 描述 0 无中断。 1 时钟超时中断被挂起。 如果在 I2CMICR 寄存器中的 CLKIC 位写入 1，该位会被清零。
0	RIS	RO	0	主机原始中断状态寄存器 值 描述 0 无中断。 1 主机中断被挂起。 通过向 I2CMICR 寄存器的 IC 位写 1 来清除该位。

寄存器 7: I²C 主机屏蔽中断状态寄存器（I2CMMIS），偏移量 0x018

该寄存器表示是否发出中断信号。

I2C 主机屏蔽中断状态寄存器 (I2CMMIS)

I2C 0 基址: 0x4002.0000
I2C 1 基址: 0x4002.1000
I2C 2 基址: 0x4002.2000
I2C 3 基址: 0x4002.3000
偏移量 0x018
类型 RO, 复位 0x0000.0000



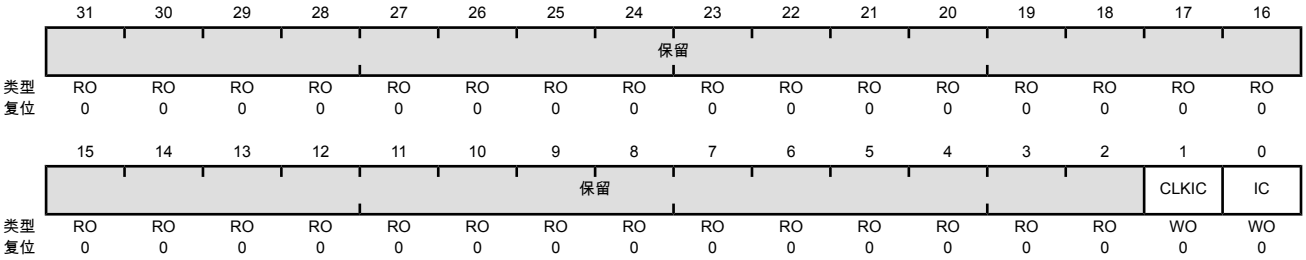
位/域	名称	类型	复位	描述
31:2	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
1	CLKMIS	RO	0	时钟超时屏蔽中断状态 值 描述 0 无中断。 1 未经屏蔽的时钟超时中断信号已经发出，已经挂起。 如果在 I2CMICR 寄存器中的 CLKIC 位写入 1，该位会被清零。
0	MIS	RO	0	屏蔽中断状态 值 描述 0 中断没有发生或者被屏蔽。 1 未经屏蔽的主机中断信号已经发出，已经挂起。 通过向 I2CMICR 寄存器的 IC 位写 1 来清除该位。

寄存器 8: I²C 主机中断清除寄存器（I2CMICR），偏移量 0x01C

该寄存器可清除原始中断和屏蔽中断。

I2C 主机中断清除寄存器 (I2CMICR)

I2C 0 基址: 0x4002.0000
I2C 1 基址: 0x4002.1000
I2C 2 基址: 0x4002.2000
I2C 3 基址: 0x4002.3000
偏移量 0x01C
类型 WO, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:2	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
1	CLKIC	WO	0	时钟超时中断清除 在该位写入 1 即可将 I2CMRIS 寄存器中的 CLKRIS 位和 I2CMMIS 寄存器中的 CLKMIS 位清零。 读取该寄存器的值没有任何意义
0	IC	WO	0	主机中断清除寄存器 向该位写入 1 可清除 I2CMRIS 的 RIS 位和 I2CMMIS 的 MIS 位。 读取该寄存器的值没有任何意义

寄存器 9: I²C 主机配置寄存器（I2CMCR），偏移量 0x020

该寄存器用于配置主机或从机模式、启用故障滤波器，并为测试模式回送设置接口。

I2C 主机配置寄存器 (I2CMCR)

I2C 0 基址: 0x4002.0000
I2C 1 基址: 0x4002.1000
I2C 2 基址: 0x4002.2000
I2C 3 基址: 0x4002.3000
偏移量 0x020
类型 R/W, 复位 0x0000.0000

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
域	保留															
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
域	保留									GFE	SFE	MFE	保留			LPBK
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	RO	RO	RO	R/W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位/域	名称	类型	复位	描述
31:7	保留	RO	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
6	GFE	R/W	0	I²C 故障滤波器启用 值 描述 0 I²C 故障滤波器已禁用。 1 I²C 故障滤波器已启用。 通过 I²C 主机配置 2 (I2CMCR2) 寄存器的 GFPW 位配置脉冲宽度。
5	SFE	R/W	0	I²C 从机功能启用 值 描述 0 从机模式禁用。 1 从机模式启用。
4	MFE	R/W	0	I²C 主机功能启用 值 描述 0 主机模式禁用。 1 主机模式启用。
3:1	保留	RO	0x0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
0	LPBK	R/W	0	I²C 回送 值 描述 0 正常工作 1 控制器处于测试模式回送状态。

寄存器 10: I²C 主机时钟低电平超时计数寄存器（I2CMCLKOCNT），偏移量 0x024

该寄存器包含一个 12 位计数器的高 8 位，可用来保持远程从机拉低时钟的超时限制。计数器的低四位值始终为 0x0，用户无法查看。

注意： 主机时钟低电平超时计数器会计算 SCL 被持续拉低的所有时间。如果 SCL 在任何时候失效，主机时钟低电平超时计数器将重新加载 I2CMCLKOCNT 寄存器中的值，并从此值开始递减计数。

I2C 主机时钟低电平超时计数寄存器 (I2CMCLKOCNT)

I2C 0 基址: 0x4002.0000
I2C 1 基址: 0x4002.1000
I2C 2 基址: 0x4002.2000
I2C 3 基址: 0x4002.3000
偏移量 0x024
类型 R/W, 复位 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	保留															
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	保留								CNTL							
类型	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

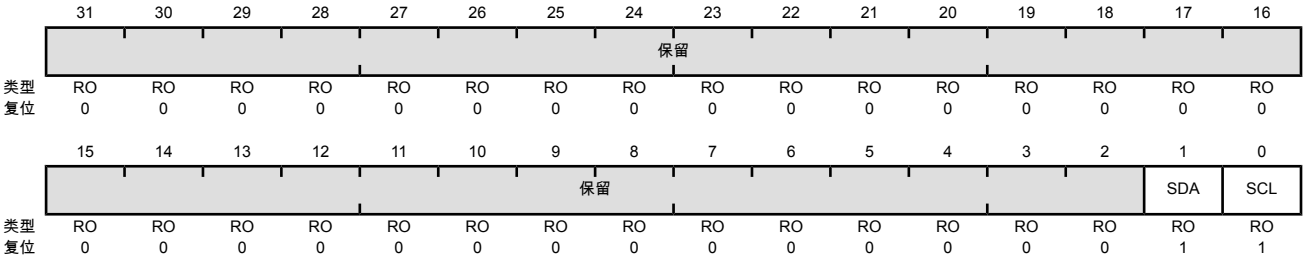
位/域	名称	类型	复位	描述
31:8	保留	RO	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
7:0	CNTL	R/W	0	I²C 主机计数值 该域包含一个 12 位计数器的高 8 位，可用来进行时钟低电平超时计数。 注意： CNTL 值必须大于 0x1。

寄存器 11: I²C 主机总线监视寄存器（I2CMBMON），偏移量 0x02C

该寄存器用来确定 SCL 和 SDA 的信号状态。

I2C 主机总线监视寄存器 (I2CMBMON)

I2C 0 基址: 0x4002.0000
I2C 1 基址: 0x4002.1000
I2C 2 基址: 0x4002.2000
I2C 3 基址: 0x4002.3000
偏移量 0x02C
类型 RO, 复位 0x0000.0003



位/域	名称	类型	复位	描述
31:2	保留	RO	0x0000.000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
1	SDA	RO	1	I²C SDA 状态 值 描述 0 I2CSDA 信号为低电平。 1 I2CSDA 信号为高电平。
0	SCL	RO	1	I²C SCL 状态 值 描述 0 I2CSCL 信号为低电平。 1 I2CSCL 信号为高电平。

寄存器 12: I²C 主机配置 2 寄存器（I2CMCR2），偏移量 0x038

此寄存器用于为故障抑制设置脉冲宽度，以系统时钟数计。

I2C 主机配置 2 寄存器 (I2CMCR2)

I2C 0 基址: 0x4002.0000
I2C 1 基址: 0x4002.1000
I2C 2 基址: 0x4002.2000
I2C 3 基址: 0x4002.3000
偏移量 0x038
类型 R/W, 复位 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	保留															
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	保留									GFPW			保留			
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位/域	名称	类型	复位	描述
31:7	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
6:4	GFPW	R/W	0	I²C 故障滤波器脉冲宽度 此域用于控制 SCL 和 SDA 线路的故障抑制脉冲宽度。故障抑制值以系统时钟数计。 值 描述 0x0 旁路 0x1 1 个时钟 0x2 2 个时钟 0x3 3 个时钟 0x4 4 个时钟 0x5 8 个时钟 0x6 16 个时钟 0x7 31 个时钟
3:0	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。

16.7 寄存器描述（I²C 从机）

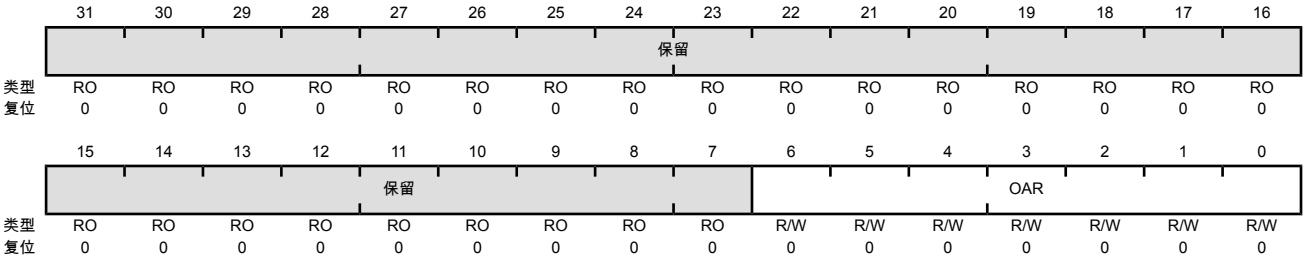
本章的剩余部分按照地址偏移量由小到大的顺序依次详细介绍各 I²C 从机寄存器。

寄存器 13: I²C 从机本身地址寄存器（I2CSOAR），偏移量 0x800

该寄存器包括 7 个地址位，用以识别 I²C 总线上的 TM4C1231H6PM I²C 设备。

I2C 从机本身地址寄存器 (I2CSOAR)

I2C 0 基址: 0x4002.0000
I2C 1 基址: 0x4002.1000
I2C 2 基址: 0x4002.2000
I2C 3 基址: 0x4002.3000
偏移量 0x800
类型 R/W, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:7	保留	RO	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
6:0	OAR	R/W	0x00	I²C 从机本身地址 该字段表示从机地址的A6到A0。

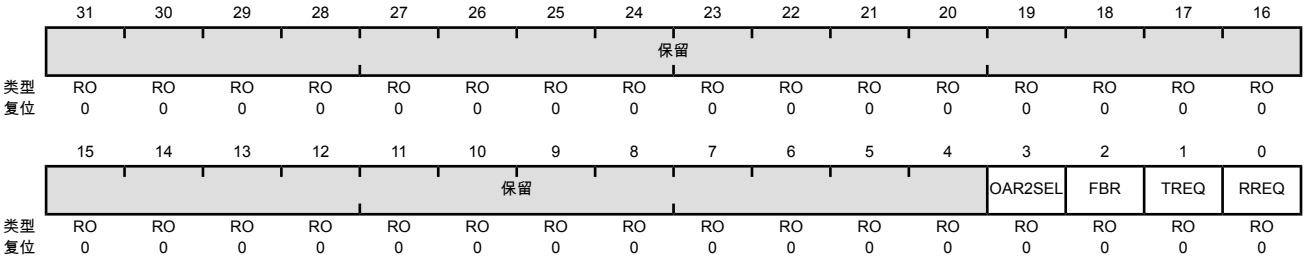
寄存器 14: I²C 从机控制/状态寄存器（I2CSCSR），偏移量 0x804

写入时，该寄存器是控制寄存器；读取时，该寄存器是状态寄存器。

只读状态寄存器

I2C 从机控制/状态寄存器 (I2CSCSR)

I2C 0 基址: 0x4002.0000
I2C 1 基址: 0x4002.1000
I2C 2 基址: 0x4002.2000
I2C 3 基址: 0x4002.3000
偏移量 0x804
类型 RO, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:4	保留	RO	0x0000.000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
3	OAR2SEL	RO	0	OAR2 地址匹配 值 描述 0 地址没有匹配，或者匹配是传统模式。 1 OAR2 地址已匹配，而且得到从机应答。 比较每个地址之后，该位会被重新评估。
2	FBR	RO	0	第一个字节已接收 值 描述 0 还没有接收到第一个字节 1 接收到了紧随从机自身地址的第一个字节。 仅当 RREQ 位置位时该位才有效。并且在 I2CSDR 寄存器数据读出后该位自动清零。 注意: 该位不用于从机发送操作。
1	TREQ	RO	0	发送请求 值 描述 0 没有未完成的发送请求。 1 I²C 控制器被设置成从发送器，正在使用时钟拉低来延迟主机，直到数据写入 I2CSDR 寄存器中。

位/域	名称	类型	复位	描述
0	RREQ	RO	0	接收请求

值 描述

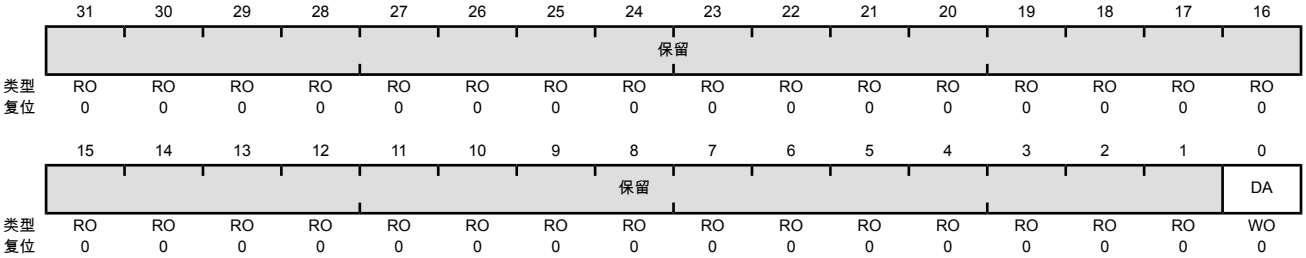
0 没有未完成的接收数据。

1 I^2C 控制器有未完成的来自 I^2C 主机的接收数据，正在使用时钟拉低来延迟主机，直到 I2CSDR 寄存器的数据被读取。

只写控制寄存器

I2C 从机控制/状态寄存器 (I2CSCSR)

I2C 0 基址: 0x4002.0000
I2C 1 基址: 0x4002.1000
I2C 2 基址: 0x4002.2000
I2C 3 基址: 0x4002.3000
偏移量 0x804
类型 WO, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:1	保留	RO	0x0000.000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
0	DA	WO	0	设备激活

值 描述

0 禁用 I^2C 从机操作。

1 启用 I^2C 从机操作。

一旦将该位置位，则在通过写 0 或复位而清零该位前，不得再次将该位置位，否则可能发生传输故障。

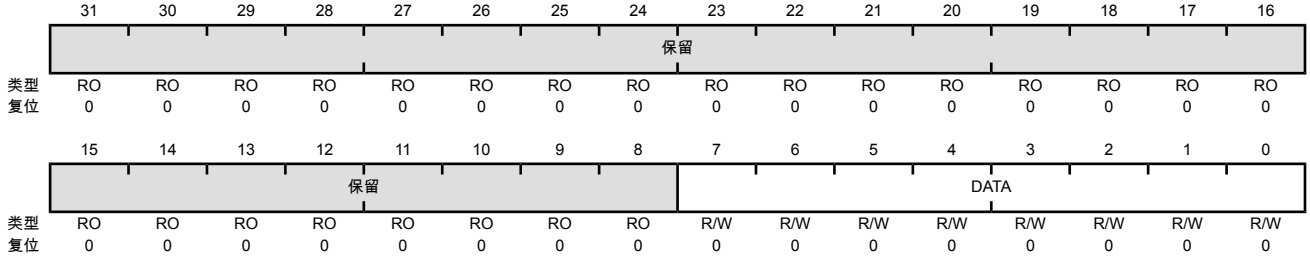
寄存器 15: I²C 从机数据寄存器（I2CSDR），偏移量 0x808

重要: 本寄存器为读敏感型寄存器。有关详细信息，请参阅寄存器描述部分。

该寄存器含有在从机发送状态中准备发送的数据，以及在从机接收状中接收到的数据。

I2C 从机数据寄存器 (I2CSDR)

I2C 0 基址: 0x4002.0000
I2C 1 基址: 0x4002.1000
I2C 2 基址: 0x4002.2000
I2C 3 基址: 0x4002.3000
偏移量 0x808
类型 R/W, 复位 0x0000.0000



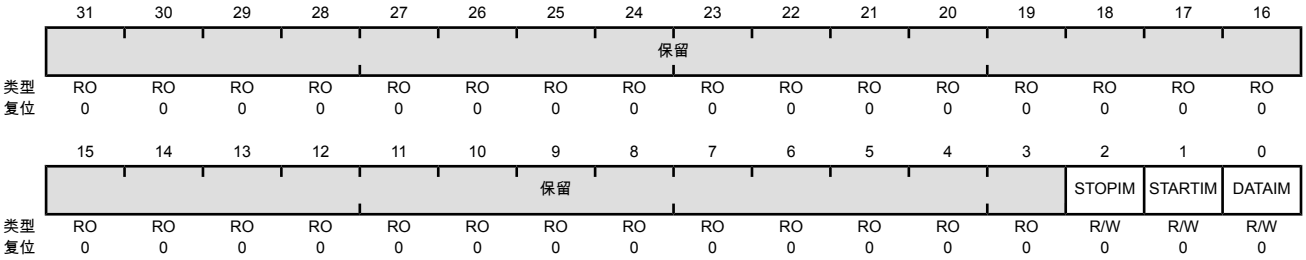
位/域	名称	类型	复位	描述
31:8	保留	RO	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
7:0	DATA	R/W	0x00	要发送的数据 该字段包含了在从机接收或发送操作中传输的数据。

寄存器 16: I²C 从机中断屏蔽（I2CSIMR），偏移量 0x80C

该寄存器控制是否将原始中断提交到控制器中断。

I2C 从机中断屏蔽 (I2CSIMR)

I2C 0 基址: 0x4002.0000
I2C 1 基址: 0x4002.1000
I2C 2 基址: 0x4002.2000
I2C 3 基址: 0x4002.3000
偏移量 0x80C
类型 R/W, 复位 0x0000.0000



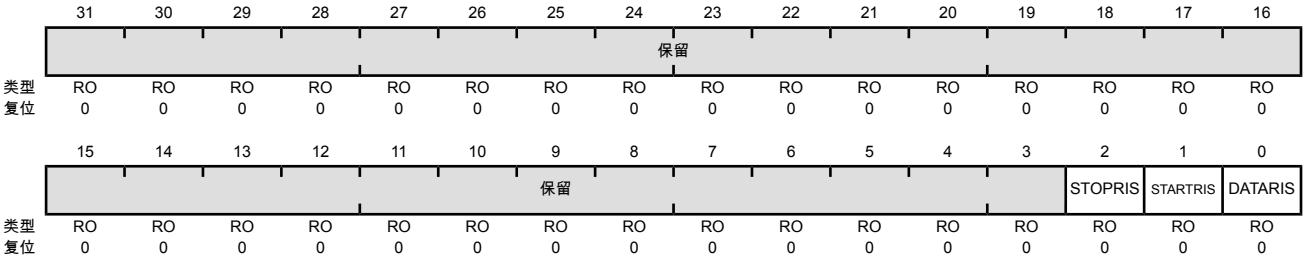
位/域	名称	类型	复位	描述
31:3	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
2	STOPIM	R/W	0	停止条件中断屏蔽 值 描述 0 STOPRIS 中断被抑制，不发送到中断控制器。 1 当 I2CSRIS 寄存器中的 STOPRIS 位置位时，STOP 条件中断被送到中断控制器。
1	STARTIM	R/W	0	开始条件中断屏蔽 值 描述 0 STARTRIS 中断被抑制，不会发送到中断控制器。 1 当 I2CSRIS 寄存器中的 STARTRIS 位置位时，START 条件中断被送到中断控制器。
0	DATAIM	R/W	0	数据中断屏蔽 值 描述 0 DATARIS 中断被抑制，不会发送到中断控制器。 1 当 I2CSRIS 寄存器中的 DATARIS 位置位时，数据接收或数据请求中断被送到中断控制器。

寄存器 17: I²C 从机原始中断状态寄存器（I2CSRIS），偏移量 0x810

该寄存器表示是否有中断正等待处理

I2C 从机原始中断状态寄存器 (I2CSRIS)

I2C 0 基址: 0x4002.0000
I2C 1 基址: 0x4002.1000
I2C 2 基址: 0x4002.2000
I2C 3 基址: 0x4002.3000
偏移量 0x810
类型 RO, 复位 0x0000.0000



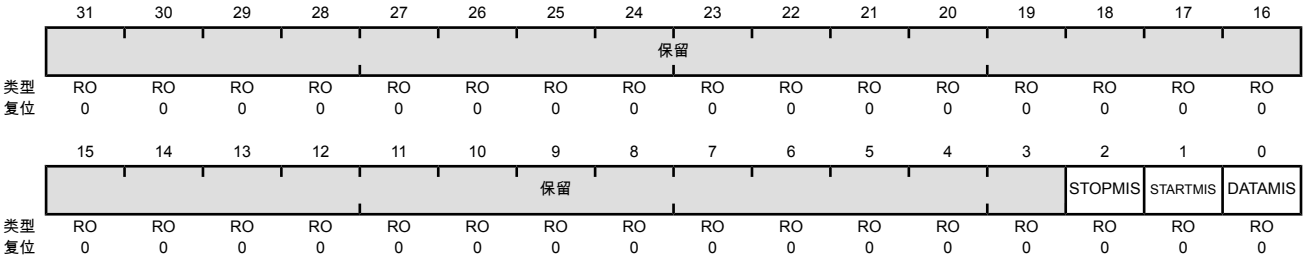
位/域	名称	类型	复位	描述
31:3	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
2	STOPRIS	RO	0	停止条件原始中断状态 值 描述 0 无中断。 1 有一个停止条件中断已挂起。 将 I2CSICR 寄存器中的 STOPIC 位置位，该位就会被清零。
1	STARTRIS	RO	0	开始条件原始中断状态 值 描述 0 无中断。 1 有一个开始条件中断已挂起。 在 I2CSICR 寄存器的 STARTIC 位写入 1 来清除该位。
0	DATARIS	RO	0	数据原始中断状态 值 描述 0 无中断。 1 有一个数据接收或者数据请求中断已挂起。 在 I2CSICR 寄存器中的 DATAIC 位写入 1，该位就会被清零。

寄存器 18: I²C 从机屏蔽中断状态寄存器（I2CSMIS），偏移量 0x814

该寄存器表示是否发出中断信号。

I2C 从机屏蔽中断状态寄存器 (I2CSMIS)

I2C 0 基址: 0x4002.0000
I2C 1 基址: 0x4002.1000
I2C 2 基址: 0x4002.2000
I2C 3 基址: 0x4002.3000
偏移量 0x814
类型 RO, 复位 0x0000.0000



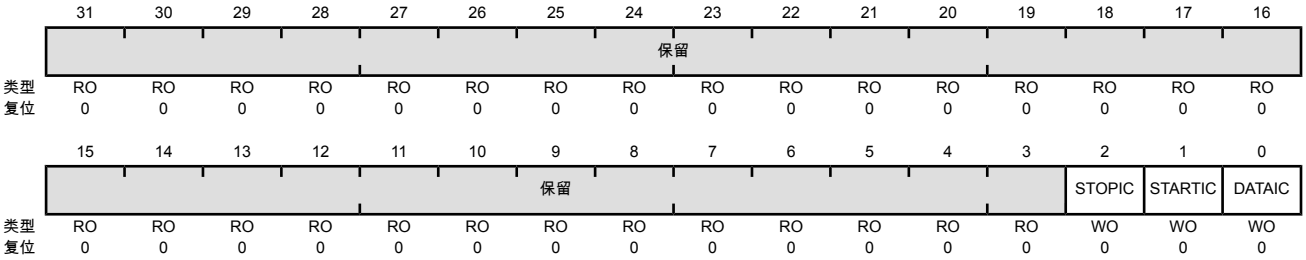
位/域	名称	类型	复位	描述
31:3	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
2	STOPMIS	RO	0	停止条件屏蔽中断状态 值 描述 0 中断没有发生或者被屏蔽。 1 有一个未屏蔽的停止条件中断已挂起。 将 I2CSICR 寄存器中的 STOPIC 位置位，该位就会被清零。
1	STARTMIS	RO	0	开始条件屏蔽中断状态 值 描述 0 中断没有发生或者被屏蔽。 1 有一个未屏蔽的开始条件中断已挂起。 在 I2CSICR 寄存器的 STARTIC 位写入 1 来清除该位。
0	DATAMIS	RO	0	数据屏蔽中断状态 值 描述 0 中断没有发生或者被屏蔽。 1 有一个没有屏蔽的停止条件中断等待处理 在 I2CSICR 寄存器中的 DATAIC 位写入 1，该位就会被清零。

寄存器 19: I²C 从机中断清除寄存器（I2CSICR），偏移量 0x818

该寄存器清除原始的中断。读取该寄存器的值没有任何意义

I2C 从机中断清除寄存器 (I2CSICR)

I2C 0 基址: 0x4002.0000
I2C 1 基址: 0x4002.1000
I2C 2 基址: 0x4002.2000
I2C 3 基址: 0x4002.3000
偏移量 0x818
类型 WO, 复位 0x0000.0000



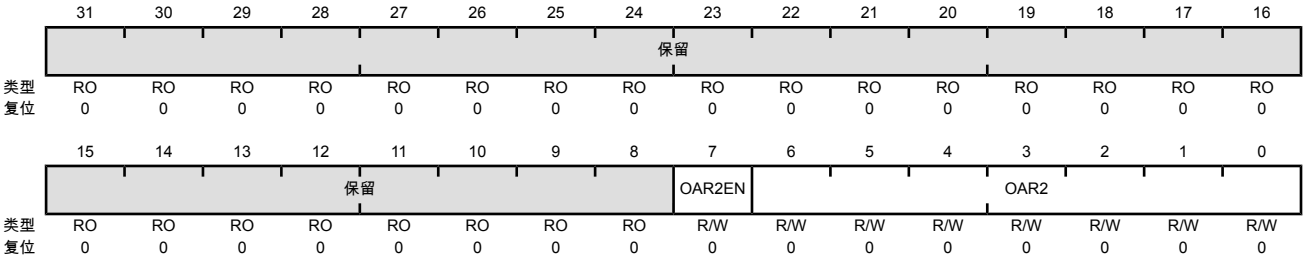
位/域	名称	类型	复位	描述
31:3	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
2	STOPIC	WO	0	停止条件中断清除 向该位写入 1 可将 I2CSRIS 寄存器的 STOPRIS 位和 I2CSMIS 寄存器的 STOPMIS 位清零。 读取该寄存器的值没有任何意义
1	STARTIC	WO	0	开始条件中断清除 在该位写入 1 即可将 I2CSRIS 寄存器中的 STARTRIS 位和 I2CSMIS 寄存器中的 STARTMIS 位清零。 读取该寄存器的值没有任何意义
0	DATAIC	WO	0	数据中断清除 向该位写入 1 可将 I2CSRIS 寄存器的 STOPRIS 位和 I2CSMIS 寄存器的 STOPMIS 位清零。 读取该寄存器的值没有任何意义

寄存器 20: I²C 从机自身地址寄存器 2（I2CSOAR2），偏移量 0x81C

该寄存器包含 7 位地址位，用以识别 I²C 总线上 I²C 设备的复用地址。

I2C 从机自身地址寄存器 2 (I2CSOAR2)

I2C 0 基址: 0x4002.0000
I2C 1 基址: 0x4002.1000
I2C 2 基址: 0x4002.2000
I2C 3 基址: 0x4002.3000
偏移量 0x81C
类型 R/W, 复位 0x0000.0000



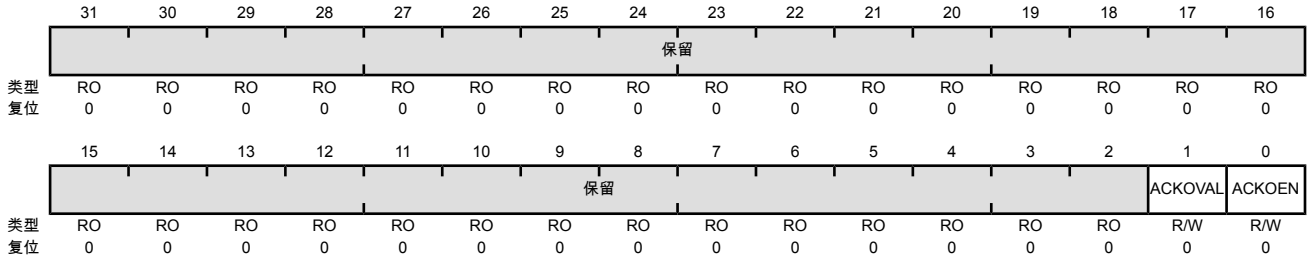
位/域	名称	类型	复位	描述
31:8	保留	RO	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
7	OAR2EN	R/W	0	I²C 从机自身地址 2 启用 值 描述 0 复用地址禁用。 1 启用 OAR2 域中的复用地址功能。
6:0	OAR2	R/W	0x00	I²C 从机自身地址 2 该域指定复用 OAR2 地址。

寄存器 21: I²C 从机应答控制寄存器（I2CSACKCTL），偏移量 0x820

该寄存器能够让 I²C 从机否定应答无效数据和指令，或应答有效数据和指令。最后一个数据位传输完成以后，I²C 时钟被拉低，直到该寄存器写入数据。

I2C 从机应答控制寄存器 (I2CSACKCTL)

I2C 0 基址: 0x4002.0000
I2C 1 基址: 0x4002.1000
I2C 2 基址: 0x4002.2000
I2C 3 基址: 0x4002.3000
偏移量 0x820
类型 R/W, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:2	保留	RO	0x0000.000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
1	ACKOVAL	R/W	0	I²C 从机应答覆盖值 值 描述 0 发送应答，表示数据或者指令有效。 1 发送否定应答，表示数据或者指令无效。
0	ACKOEN	R/W	0	I²C 从机应答取代启用 值 描述 0 没有响应。 1 设备会根据 ACKOVAL 位写入的值发送应答或否定应答。

16.8 寄存器描述（I²C 状态和控制寄存器）

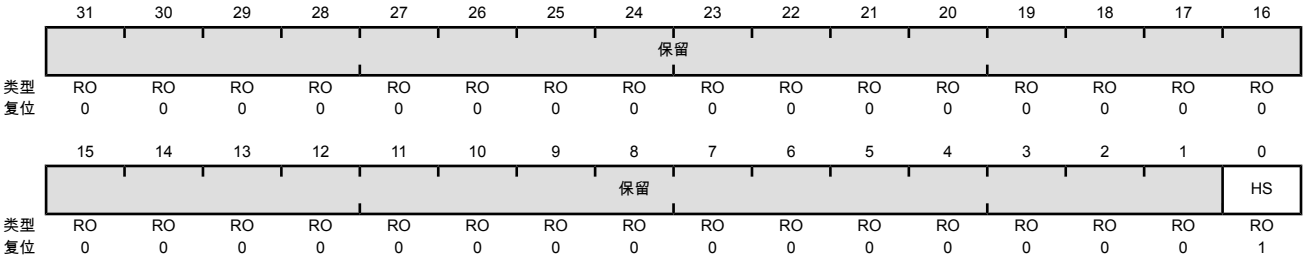
本章的剩余部分按照地址偏移量由小到大的顺序依次详细介绍各 I²C 状态和控制寄存器。

寄存器 22: I²C 外设属性寄存器（I2CPP），偏移量 0xFC0

I2CPP 寄存器提供关于 I²C 模块的属性信息。

I2C 外设属性寄存器 (I2CPP)

I2C 0 基址: 0x4002.0000
I2C 1 基址: 0x4002.1000
I2C 2 基址: 0x4002.2000
I2C 3 基址: 0x4002.3000
偏移量 0xFC0
类型 RO, 复位 0x0000.0001



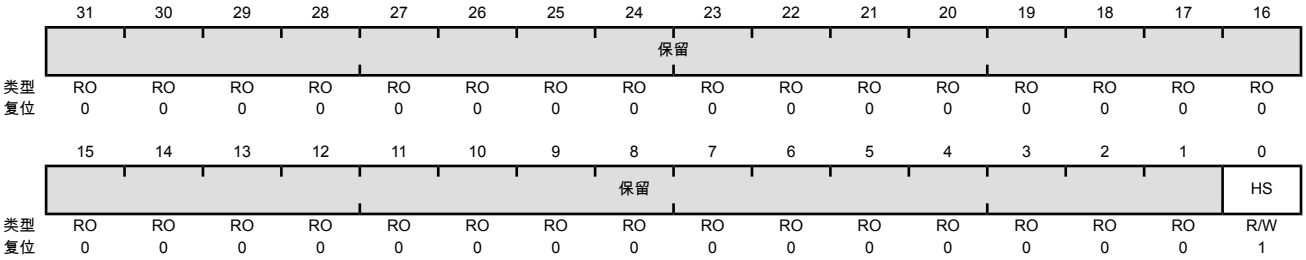
位/域	名称	类型	复位	描述
31:1	保留	RO	0x0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
0	HS	RO	0x1	可以进行高速传输 值 描述 0 该接口可以进行标准、快速或超快操作。 1 该接口可以进行高速操作。

寄存器 23: I²C 外设配置寄存器（I2CPC），偏移量 0xFC4

软件可通过 I2CPC 寄存器启用 I²C 模块提供的各种功能。

I2C 外设配置寄存器 (I2CPC)

I2C 0 基址: 0x4002.0000
I2C 1 基址: 0x4002.1000
I2C 2 基址: 0x4002.2000
I2C 3 基址: 0x4002.3000
偏移量 0xFC4
类型 RO, 复位 0x0000.0001



位/域	名称	类型	复位	描述
31:1	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读-修改-写操作过程中应该保持不变。
0	HS	R/W	1	可以进行高速传输

值 描述

0 该接口设置为进行标准、快速或超快操作。

1 该接口设置为进行高速操作。请注意：该编码只能在 I2CPP 的 HS 位已置位时使用。否则，该编码不可用。

17 控制器局域网（CAN）模块

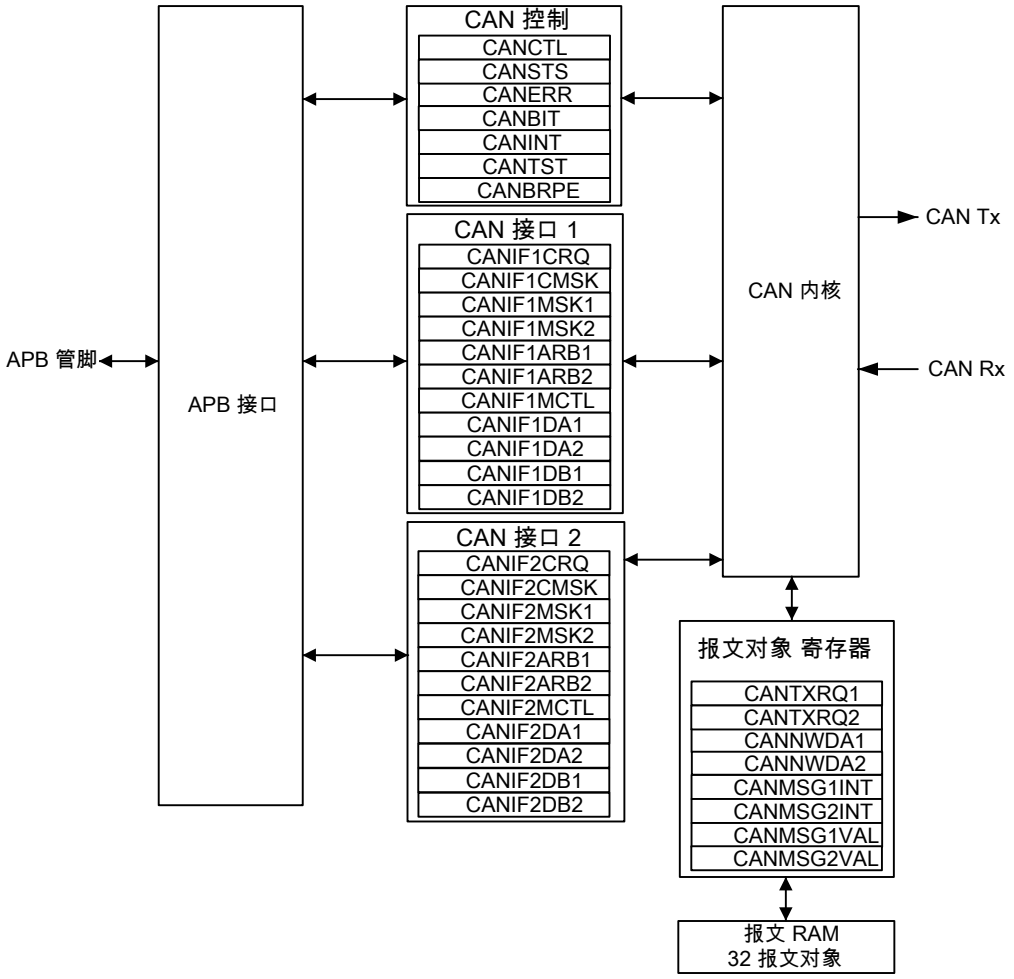
控制器局域网（Controller Area Network，简写为CAN）是一种用于连接电子控制设备（Eletronic Control Unit，简写为ECU）的多主共享型串行总线标准。CAN总线针对抗电磁干扰进行了专门设计，适用于具有较强电磁干扰的环境，不但可以使用与RS-485类似的差分平衡传输线，也可以使用更加可靠的双绞线。CAN总线最初是针对汽车应用而研发的，不过时至今日已经广泛应用于各种嵌入式控制领域（例如工业方面和医疗方面）。CAN总线在总线长度小于40米时最高可达1Mbps位速率。位速率越低则有效通讯距离越远（例如125kbps时通讯距离可达500米）。

TM4C1231H6PM 微控制器包括 1 个 CAN 单元，有如下特性：

- 支持 CAN 总线协议 2.0 A/B
- 位速率最高可达 1 Mbps
- 32 个报文对象，每个报文对象都具有独立的标识符掩码
- 可屏蔽中断；
- 支持禁用自动重新发送（Disable Automatic Retransmission，简写为 DAR）模式，因此可用于时间触发 CAN（Time Triggered CAN，简写为 TTCAN）应用
- 可编程的回送模式，用于实现自检
- 可编程的FIFO模式，能存储多个报文对象
- 提供 CANnTX 和 CANnRX 管脚，可无缝连接片外 CAN 收发器。

17.1 结构框图

图 **17-1. CAN** 控制器结构图



17.2 信号描述

下表列出了 CAN 控制器的外部信号及其功能。CAN控制器信号通常是GPIO信号的备选功能，因此这些管脚在复位时默认设置为GPIO信号。表中“复用管脚/赋值”一列是各CAN信号所对应的GPIO管脚。应将 GPIO 备用功能选择 (GPIOAFSEL) 寄存器（595页）中的 AFSEL 位置位，以便选择 CAN 控制器功能。必须将括号中的数字写入 GPIO 端口控制 (GPIOPCTL) 寄存器（612页）的 PMCn 域中，以便把 CAN 信号分配给指定 GPIO 端口管脚。有关如何配置 GPIO 的更多信息，请参阅“通用输入/输入端口（GPIOs）”（575页）。

表 **17-1.** 控制器局域网 信号 **(64LQFP)**

管脚名称	管脚编号	管脚复用/管脚赋值	管脚类型	缓冲区类型[a]	描述
CAN0Rx	28 58 59	PF0 (3) PB4 (8) PE4 (8)	I	TTL	CAN模块0接收信号。

表 **17-1.** 控制器局域网 信号 **(64LQFP)**（续）

管脚名称	管脚编号	管脚复用/管脚赋值	管脚类型	缓冲区类型[a]	描述
CAN0Tx	31 57 60	PF3 (3) PB5 (8) PE5 (8)	O	TTL	CAN模块0发送信号。

a. TTL 表示管脚的电压水平与 TTL 一致。

17.3 功能说明

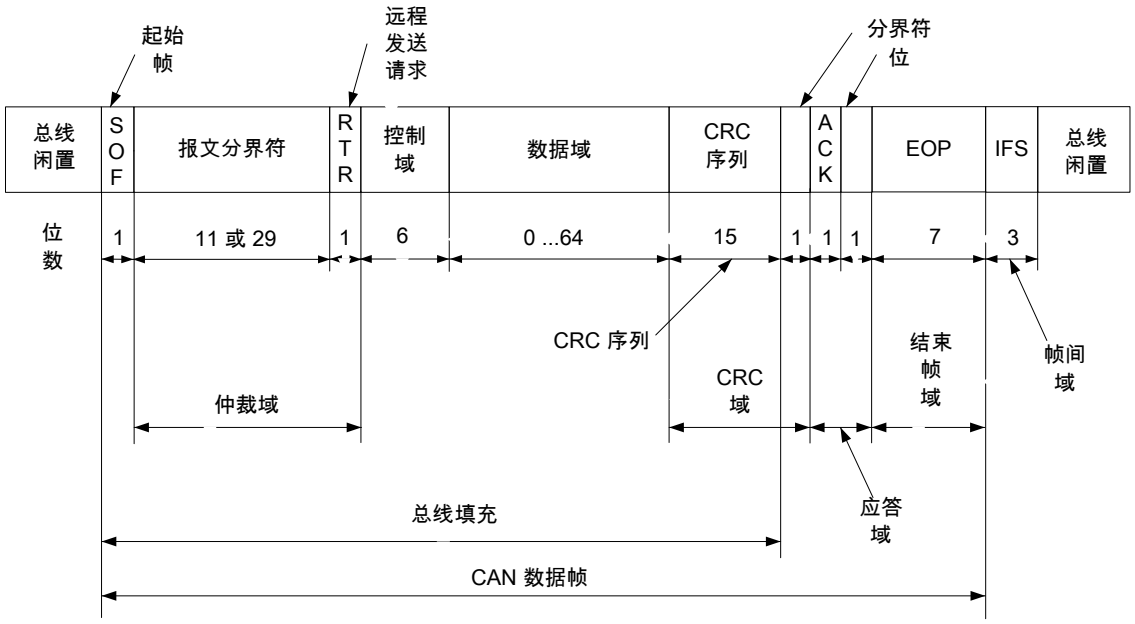
TM4C1231H6PM CAN 控制器支持 CAN 2.0 版协议 (A/B)。支持传输的报文类型包括带有 11 位标识符（标准）或29位标识符（扩展）的数据帧、远程帧、错误帧以及过载帧。通过编程，传输速率最高可达到 1 Mbps。

CAN模块由3个主要部分组成：

- CAN协议控制器及报文处理器
- 报文存储器
- CAN寄存器接口

数据帧中包含要发送的数据，而远程帧不包含任何数据、仅用于向其它节点请求指定的报文对象。CAN 数据帧/远程帧的结构如图17-2 所示。

图 **17-2. CAN** 数据帧/远程帧



协议控制器从CAN总线上接收和发送串行数据，并将数据传递给报文处理器。报文处理器基于当前的过滤设置以及报文对象存储器中的标识符，将适当的报文内容载入与之对应的报文对象。报文处理器还负责根据CAN总线的事件产生中断。

报文对象存储器是一组32个完全相同的存储模块，可为每个报文对象保存其当前的配置、状态以及实际数据。这些报文对象可通过两组CAN报文对象寄存器接口中的任何一组进行访问。

报文存储器在 TM4C1231H6PM 存储器映射中是无法直接访问的，因此 TM4C1231H6PM CAN 控制器提供了间接访问的接口：用户可通过两个CAN接口寄存器组访问报文存储器，以便与报文对象进行通信。读取或写入报文对象时，必须使用这两个接口。两组报文对象接口也可以并行访问CAN控制器报文对象，因此当多个报文对象同时包含亟待处理的新信息时完全可以并行处理。一般我们用一组接口专司发送，另一组接口专司接收。

17.3.1 初始化

要使用 CAN 控制器，必须先通过 RCGC0 寄存器启用外设时钟（请参阅396页）。此外，还必须通过RCGC2 寄存器启用相关 GPIO 模块的时钟（请参阅402页）。要了解要启用哪一个 GPIO 端口，请参考表20-4（1019页）。将相应管脚的 GPIO AFSEL 位置位（请参阅595页）。此后，应配置 GPIOPCTL 寄存器的 PMCn 域，将 CAN 信号赋给相应的管脚。请参阅612页和表20-5（1024页）。

要开始软件初始化，请（通过软件或以硬件复位的方式）将 CAN 控制 (CANCTL) 寄存器的 INIT 位置位，或进入脱离总线状态（当发送器的错误计数值超过 255 时即进入此状态）。将 INIT 位置位后，CAN 总线上的报文传输都将终止，而且 CANnTX 信号将保持为高电平。进入初始化状态并不会改变CAN控制器、报文对象或错误计数器的配置。不过，某些配置寄存器只能在初始化状态下才能访问。

要初始化 CAN 控制器，请将 CAN 位时序 (CANBIT) 寄存器置位，并配置每个报文对象。如果无需使用某个报文对象，请将 CAN IFn 仲裁 2 (CANIFnARB2) 寄存器的 MSGVAL 位清零，即可使其标记为无效。如果需要使用某个报文对象，则应当详细初始化整个报文对象，因为一旦报文对象的某个域包含无效的数据将会导致无法预料的后果。只有将 CANCTL 寄存器的 INIT 和 CCE 位都置位，才能访问 CANBIT 寄存器和 CAN 波特率预分频器扩展 (CANBRPE) 寄存器，以配置位时序。要退出初始化状态，必须将 INIT 位清零。此后，内部比特流处理器（Bit Stream Processor，简写为 BSP）首先等待一个连续11个隐性位序列的产生（表明总线处于空闲状态），依此序列实现与CAN总线数据传输的同步，之后才能参与总线活动、开始报文传输。报文对象随时都可以初始化，与CAN控制器的初始化状态无关。不过，在开始传输报文之前必须保障所有报文对象已经配置有合适的标识符或已经被设置为无效。在正常工作时，如果需要修改某个报文对象的配置，应先将CANIFnARB2寄存器中的 MSGVAL 位清零，暂时将该报文对象设为无效。待修改好配置后，重新将 MSGVAL 位置位，以将报文对象设为有效。

17.3.2 基本操作

CAN 模块提供两组 CAN 接口寄存器（CANIF1x 和 CANIF2x）用于访问报文 RAM 中的报文对象。CAN控制器自动将与报文RAM的数据交互映射为与寄存器的数据交互。两组寄存器相互独立而又完全相同，可用于实现连续会话。一般来说，用一组接口专司发送，另一组接口专司接收。

一旦 CAN 模块完成初始化并且 CANCTL 寄存器中的 INIT 位被清零，CAN 模块将自动与 CAN 总线同步，然后开始传输报文。收到的每个报文都需要经过报文处理器的滤波。如果能够通过滤波，将会保存到由 CAN IFn 命令请求 (CANIFnCRQ) 寄存器中的 MNUM 位所指定的报文对象中。整个报文（包括所有仲裁位、数据长度码以及8个数据字节）都将保存在报文对象中。如果使用标识符屏蔽（CAN IFn 屏蔽 1 和 CAN IFn 屏蔽 2 (CANIFnMSKn) 寄存器中的 MSK 位），那么报文对象中被屏蔽为“无关”的仲裁位可能会被覆盖。

CPU随时可以通过CAN接口寄存器读写任一报文。当出现同时访问的情况时，由报文处理器负责保障数据的一致性。

报文对象的发送是在由管理CAN硬件的软件所管理的。报文对象既可以仅用于单次数据传输、也可以作为永久报文对象实现周期性的响应。永久报文对象的所有仲裁部分及控制部分是预先设好的固定值，只需不断刷新各数据字节。开始发送时，CAN 发送请求 n (CANTXRQn) 寄存器的 TXRQST 位以及 CAN 新数据 n (CANNWDAn) 寄存器的 NEWDAT 位将置位。假如多个要发送的报文都使用同一报文对象（当报文对象不够用时），必须在请求发送报文之前完整配置报文对象。

CAN 控制器允许同时请求发送任意数量的报文对象；当多个报文对象同时等待发送时，发送顺序是按照内部优先级定义的，即按照报文对象的序号 (MNUM) 排序，其中 1 是最高优先级、32 是最低

优先级。报文可以随时刷新或设置为无效，即使其发送请求尚被挂起也是如此。当报文的发送请求已挂起并且尚未开始时，刷新报文内容时将丢弃旧内容。按照报文对象的配置，当收到标识符相同的远程帧时能够自动请求发送报文。

当收到匹配的远程帧时可以自动开始请求发送。要启用该模式，请将 CAN IFn 报文控制 (CANIFnMCTL) 寄存器的 RMTEN 位置位。当收到匹配的远程帧时，相应的 TXRQST 位会置位，并且报文对象将自动发送其数据部分或产生中断，以表示有远程帧的请求。远程帧可以是严格限定的某个报文标识符，也可以是报文对象所定义的一个标识符范围。CAN 屏蔽寄存器 (CANIFnMSKn) 用于配置哪些帧组将被确定为远程帧请求。通过 CANIFnMCTL 寄存器的 UMASK 位启用 CANIFnMSKn 寄存器的 MSK 位，以确定哪些帧为远程帧请求。如果准备以 29 位扩展标识符触发远程帧请求，则应将 CANIFnMSK2 寄存器的 MXTD 位置位。

17.3.3 报文对象的发送

假如CAN模块的内部发送移位寄存器已经准备好装载，并且CAN接口寄存器与报文RAM之前无数据传输，那么已挂起发送请求并且优先级最高的有效报文对象将会被报文处理器装载入发送移位寄存器中，并开始发送流程。该报文对象在 CANNWDAn 寄存器中的 NEWDAT 位将清零。当发送成功后，如果自开始发送以后该数据对象没有写入新的数据，则 CANTXRQn 寄存器的 TXRQST 位将清零。如果 CAN 控制器配置为每成功发送一个报文对象即产生一次中断（CAN IFn 报文控制 (CANIFnMCTL) 寄存器的 TXIE 位置位），那么 CANIFnMCTL 寄存器的 INTPND 位会在每次成功发送报文对象后置位。如果CAN模块丢失仲裁（竞争总线失败）或在发送期间产生错误，该报文将在CAN总线一有空闲时立即重新发送。如果与此同时有更高优先级的报文对象请求发送，那么将按照优先级的顺序依次发送报文。

17.3.4 待发送报文对象的配置

可按照下面的步骤配置要发送的报文对象：

1. 在 CAN IFn 命令屏蔽 (CANIFnCMASK) 寄存器中：
   - 将 WRNRD 位置位，以指定对 CANIFnCMASK 寄存器进行写入操作；通过 MASK 位指定是否将报文对象的 IDMASK、DIR 和 MXTD 发送给 CAN IFn 寄存器；
   - 通过 ARB 位指定是否将报文对象的 ID、DIR、XTD 和 MSGVAL 发送给接口寄存器；
   - 通过 CONTROL 位指定是否将控制位发送给接口寄存器；
   - 通过 CLRINTPND 位指定是否将 CANIFnMCTL 寄存器的 INTPND 位清零；
   - 通过 NEWDAT 位指定是否将 CANNWDAn 寄存器的 NEWDAT 位清零；
   - 通过 DATAA 和 DATAB 位指定要发送哪些数据位；
2. 通过 CANIFnMSK1 寄存器的 MSK[15:0] 位来指定 29 位或 11 位报文标识符中哪些位用于验收滤波。请注意，此寄存器的 MSK[15:0] 位域只对应于 29 位标识符的 [15:0] 位域，而并不适用于 11 位标识符。若此寄存器写入0x00，则所有报文均可通过验收过滤。此外还要注意，要将这些位用于验收滤波，必须将 CANIFnMCTL 寄存器的 UMASK 位置位，以启用这些位。
3. 配置 CANIFnMSK2 寄存器的 MSK[12:0] 位域，定义使用 29 位或 11 位标识符中的哪些仲裁位进行验收滤波。请注意，MSK[12:0] 位域对应于 29 位报文标识符的 [28:16] 位；MSK[12:2] 位域对应于 11 位报文标识符的 [10:0]。通过 MXTD 和 MDIR 位可选择是否将 XTD 和 DIR 位也用于验收滤波。若此寄存器写入0x00，则所有报文均可通过验收过滤。此外还要注意，要将这些位用于验收滤波，必须将 CANIFnMCTL 寄存器的 UMASK 位置位，以启用这些位。

4. 对于 29 位的标识符而言，配置 CANIFnARB1 寄存器的 ID[15:0] 以用于报文标识符的 [15:0] 位，而配置 CANIFnARB2 寄存器的 ID[12:0] 以用于报文标识符的 [28:16] 位。此外应将 XTD 位置位，表示采用扩展标识符；将 DIR 位置位，表示发送；将 MSGVAL 位置位，表示该报文对象有效。

5. 对于 11 位标识符，不考虑 CANIFnARB1 寄存器并配置 CANIFnARB2 寄存器的 ID[12:2] 以用于报文标识符的 [10:0] 位。此外应将 XTD 位清零，表示采用标准标识符；将 DIR 位置位，表示发送；将 MSGVAL 位置位，表示该报文对象有效。

6. 在 CANIFnMCTL 寄存器中：

   - 可选设置：将 UMASK 位置位即可为验收滤波启用掩码（CANIFnMSK1 和 CANIFnMSK2 寄存器中指定的 MSK、MXTD 和 MDIR）；
   - 可选设置：将 TXIE 位置位，使 INTPND 位在成功发送帧后自动置位；
   - 可选设置：将 RMTEN 位置位，这样每当收到匹配的远程帧后 TXRQST 位自动置位，从而允许自动传输；
   - 对于单个报文对象，应将 EOB 位置位；
   - 配置 DLC[3:0] 位域，指定数据帧的长度。配置时应注意不要将 NEWDAT、MSGLST、INTPND 或 TXRQST位置位。

7. 将待发送数据加载到 CAN IFn 数据（CANIFnDA1、CANIFnDA2、CANIFnDB1 和 CANIFnDB2）寄存器中。CAN 数据帧的数据字节 0 将保存在 CANIFnDA1 寄存器的 DATA[7:0] 中。

8. 在 CAN IFn 命令请求 (CANIFnCRQ) 寄存器的 MNUM 域中通过编程填写待发送报文对象的编号。

9. 当所有配置都成功完成后，将 CANIFnMCTL 寄存器的 TXRQST 位置位。此标志位置位后，报文对象就会在总线空闲时按照优先级进行发送了。请注意，如果 CANIFnMCTL 寄存器的 RMTEN 位置位，则在接收到匹配的远程帧后也会自动开始发送报文。

17.3.5 待发送报文对象的刷新

CPU 随时可以通过 CAN 接口寄存器刷新待发送报文对象的数据字节，而且在刷新之前并不一定需要将 CANIFnARB2 寄存器的 MSGVAL 位或者 CANIFnMCTL 寄存器的 TXRQST 位清零。

在将 CANIFnDAn/CANIFnDBn 寄存器的内容传递给报文对象之前，即使只需刷新其中的几个数据字节，也必须保证寄存器中的 4 个字节全都有效。通过 CPU 写入新数据字节时，应将所有 4 个字节写入 CANIFnDAn/CANIFnDBn 寄存器中，或者先将报文对象传输到 CANIFnDAn/CANIFnDBn 寄存器，然后再写入数据字节。

要想只刷新报文对象中的数据，应将 CANIFnMSKn 寄存器的 WRNRD、DATAA 和 DATAB 位置位，接着将刷新数据写入 CANIFnDA1、CANIFnDA2、CANIFnDB1 和 CANIFnDB2 寄存器，然后将报文对象编号写入 CAN IFn 命令请求 (CANIFnCRQ) 寄存器的 MNUM 域。要想尽快开始发送新数据，应将 CANIFnMSKn 寄存器的 TXRQST 位置位。

刷新数据时，如果报文对象已开始发送，那么在发送完成后，CANIFnMCTL 寄存器的 TXRQST 位可能被清零。为了避免出现这种情况，应同时将 CANIFnMCTL 寄存器的 NEWDAT 和 TXRQST 位置位。只要这两个位同时置位，则当开始新的发送过程后，NEWDAT 位将立即清零。

17.3.6 已接收报文对象的接受

当已接收报文的仲裁域和控制域（CANIFnARB2 寄存器的 ID 和 XTD 位以及 CANIFnMCTL 寄存器的 RMTEN 和 DLC[3:0] 位）已经全部移入 CAN 控制器后，控制器的报文处理功能将开始扫描报文 RAM，以搜索与之匹配的有效报文对象。要扫描报文 RAM 以搜索匹配报文对象，控制器将使用滤波功能。此功能通过 CANIFnMSKn 寄存器屏蔽位设置，并通过 CANIFnMCTL 寄存器的 UMASK 位启用。控制器将从报文对象1开始逐个将有效报文对象与收到的报文进行比对，以期在报文RAM中找到匹配的报文对象。当找到匹配的报文对象后，扫描过程就此结束，随后报文处理器将根据收到的是数据帧还是请求帧予以分别处理。

17.3.7 接收数据帧

报文处理器将来自 CAN 控制器接收移位寄存器的报文存储到报文 RAM 中匹配的报文对象中。而数据字节、所有仲裁位和 DLC 位都存储到相应的报文对象中。即使采用了仲裁掩码，数据字节也是与标识符相关联的。CANIFnMCTL 寄存器的 NEWDAT 位置位时，即表示已经收到新数据。CPU在读取此报文对象之后必须将此标志位清零，告诉控制器已经接收处理此报文、该条缓冲可用于接收新的报文了。当 NEWDAT 位已经置位时，假如 CAN 控制器又接收到一条新报文，那么 CANIFnMCTL 寄存器的 MSGLST 位将自动置位，表示前一条数据丢失。假如系统需要在成功接收完一帧后产生中断，则应将 CANIFnMCTL 寄存器的 RXIE 位置位。在这种情况下，当某个报文对象成功接收一帧后，该寄存器的 INTPND 位将自动置位，使得 CANINT 寄存器指向该报文对象。注意：该报文对象的 TXRQST 位应当清零，防止发送远程帧。

17.3.8 接收远程帧

远程帧中不包含数据，而是通知其它节点指示应当发送哪一报文对象。当收到远程帧时，应考虑匹配报文对象的3种不同配置：

表 **17-2.** 报文对象的配置

CANIFnMCTL 寄存器的配置	描述
■ DIR = 1（方向为发送）；通过 CANIFnARB2 寄存器设置 ■ RMTEN = 1（在收到帧时，将 CANIFnMCTL 寄存器的的 TXRQST 位，以启用发送） ■ UMASK = 1 或 0	在接收到匹配的远程帧时，将该报文对象的 TXRQST 位置位。其他报文对象保持不变。控制器将尽快开始自动发送该报文对象的数据。
■ DIR = 1（方向为发送）；通过 CANIFnARB2 寄存器设置 ■ RMTEN = 0（在收到帧时，不改变 CANIFnMCTL 寄存器的 TXRQST 位的设置） ■ UMASK = 0（忽略 CANIFnMSKn 寄存器中的屏蔽设置）	在接收到匹配的远程帧时，该报文对象的 TXRQST 位保持不变；而远程帧被忽略。该远程帧将被禁用，数据不会被传输，且看不到任何能表明该远程帧曾经存在过的迹象。
■ DIR = 1（方向为发送）；通过 CANIFnARB2 寄存器设置 ■ RMTEN = 0（在收到帧时，不改变 CANIFnMCTL 寄存器的 TXRQST 位的设置） ■ UMASK = 1（使用掩码 [CANIFnMSKn 寄存器的 MSK、MXTD 和 MDIR] 进行验收滤波）	在接收到匹配的远程帧时，将该报文对象的 TXRQST 位清零。移位寄存器的仲裁域和控制域 (ID + XTD + RMTEN + DLC) 将存储到报文 RAM 中的报文对象中，并且该报文对象的 NEWDAT 位将置位。报文对象的数据域保持不变；像接收到的数据帧那样处理远程帧。该模式对于来自另一 CAN 器件的远程数据请求来说非常有用，因为其 TM4C1231H6PM 控制器可能没有包含可用的数据。软件必须填充数据并人工响应此帧。

17.3.9 接收/发送优先级

报文对象的接收/发送优先级由报文编号决定。报文对象 1 的优先级最高，报文对象 32 的优先级最低。假如当前挂起的发送请求不止一个，那么将优先发送序号最小的那个报文对象。请注意，这里所说的优先级与CAN总线上通过报文标识符强制实现的总线优先级没有半点关系。举例来说，假如报文对象1和报文对象2同时有待发送的有效报文，那么报文对象1将总是优先发送，不管其报文标识符的总线优先级如何。

17.3.10 接收报文对象的配置

可按照下面的步骤配置接收报文对象：

1. 按照“待发送报文对象的配置”（951页）中所描述的方法对 CAN IFn 命令屏蔽 (CANIFnCMASK) 寄存器进行配置，但 WRNRD 位要被置位，表明是对报文 RAM 的写入操作。

2. 按照“待发送报文对象的配置”（951页）中描述的方法对 CANIFnMSK1 和 CANIFnMSK2 寄存器进行配置，以设置将哪些位用作验收滤波。注意，要将这些位用于验收滤波，必须将 CANIFnMCTL 寄存器的 UMASK 位置位，以启用这些位。

3. 配置 CANIFnMSK2 寄存器的 MSK[12:0] 位域，定义使用 29 位或 11 位标识符中的哪些仲裁位进行验收滤波。请注意，MSK[12:0] 位域对应于 29 位报文标识符的 [28:16] 位；MSK[12:2] 位域对应于 11 位报文标识符的 [10:0]。通过 MXTD 和 MDIR 位可选择是否将 XTD 和 DIR 位也用于验收滤波。若此寄存器写入0x00，则所有报文均可通过验收过滤。此外还要注意，要将这些位用于验收滤波，必须将 CANIFnMCTL 寄存器的 UMASK 位置位，以启用这些位。

4. 按照“待发送报文对象的配置”（951页）中描述的方法配置 CANIFnARB1 和 CANIFnARB2 寄存器的 XTD 和 ID 位以接收报文标识符，并将 MSGVAL 位置位以表示报文有效，同时将 DIR 位清零以表示接收。

5. 在 CANIFnMCTL 寄存器中：

   - 可选设置：将 UMASK 位置位即可为验收滤波启用掩码（CANIFnMSK1 和 CANIFnMSK2 寄存器中指定的 MSK、MXTD 和 MDIR）；
   - 可选设置：将 RXIE 位置位，这样每当成功接收一帧后 INTPND 位将自动置位；
   - 将 RMTEN 位清零，使得 TXRQST 位保持不变；
   - 对于单个报文对象，应将 EOB 位置位；
   - 配置 DLC[3:0] 位域，指定数据帧的长度；

   配置时应注意不要将 NEWDAT、MSGLST、INTPND 或 TXRQST位置位。

6. 通过 CAN IFn 命令请求 (CANIFnCRQ) 寄存器的 MNUM 域设置待接收报文对象的编号。当CAN总线上有匹配的帧时，将立即开始接收报文对象。

当报文处理器向数据对象保存一个数据帧时，会将收到的数据长度码 (DLC) 以及 8 个数据字节写入 CANIFnDA1、CANIFnDA2、CANIFnDB1 和 CANIFnDB2 寄存器。CAN 数据帧的数据字节 0 将保存在 CANIFnDA1 寄存器的 DATA[7:0] 中。假如数据长度码小于8，那么报文对象的剩余数据字节将以随机值予以覆盖。

通过CAN掩码寄存器（CANIFnMSKn）可以允许某个报文对象只接收某些数据帧。CAN 屏蔽寄存器 (CANIFnMSKn) 用于配置哪些帧组将被报文对象接收。通过 CANIFnMCTL 寄存器的 UMASK 位启用 CANIFnMSKn 寄存器的 MSK 位，以确定哪些帧将被接收。如果报文对象仅使用 29 位扩展标识符，则应将 CANIFnMSK2 寄存器中的 MXTD 位置位。

17.3.11 已接收报文对象的处理

报文处理器状态机已足可保障数据的一致性，因此CPU随时都能通过CAN接口寄存器组来读取已接收的报文。

一般来说，CPU 会先向 CANIFnCMSK 寄存器写入 0x007F，随后向 CANIFnCRQ 寄存器写入报文对象的编号。这个操作组合将使已接收的整条报文从报文 RAM 移入报文缓冲寄存器中（CANIFnMSKn、CANIFnARBn 和 CANIFnMCTL 寄存器）。此外，报文 RAM 中的 NEWDAT 和 INTPND 位也会清零，以确认该报文对象已经被读出，并清除该报文对象所挂起的中断。

假如报文对象使用掩码进行验收滤波，那么可以通过 CANIFnARBn 寄存器查询已接收报文在屏蔽操作之前的完整 ID。

CANIFnMCTL 寄存器的 NEWDAT 位能够指示出自从上一次读取此报文对象之后是否又收到了一条新的报文。CANIFnMCTL 寄存器的 MSGLST 位能够指示出自从上一次读取此报文对象之后是否又收到了多条新的报文。MSGLST 位不会自动清零，因此必须在每次读取其状态后通过软件将其清零。

通过运用远程帧，CPU 可以向 CAN 总线的其它节点请求新的数据。如果某个报文对象的方向已经设置为接收，则将该报文对象的 TXRQST 位置位时会以其报文标识符发送一个远程帧。此远程帧将触发其它CAN节点在总线上发送标识符匹配的数据帧。假如在远程帧发送完成之前收到了匹配的数据帧，那么 TXRQST 位将自动复位。这样在CAN总线上的其它节点已经发送数据帧（稍微早于预期的时间）时，可以避免发生数据丢失。

17.3.11.1 FIFO缓冲区的配置

除 CANIFnMCTL 寄存器的 EOB 位以外，属于 FIFO 缓冲区的接收报文对象的配置方式与单个接收报文对象的配置方法完全相同（请参阅“接收报文对象的配置”（954页））。要将2个或2个以上的报文对象连锁到同一个FIFO缓冲区中，这些报文对象的标识符及掩码（如果有的话）都应当编程为相同的值。由于报文对象已经有固定的优先级，因此序号最小的报文对象必然是FIFO缓冲区中的首个报文对象。在 FIFO 中，除了最后一个报文对象外，其它报文对象的 EOB 位必须清零。最后一个报文对象的 EOB 位置位即可表明它是缓冲区中的最后一个单元。

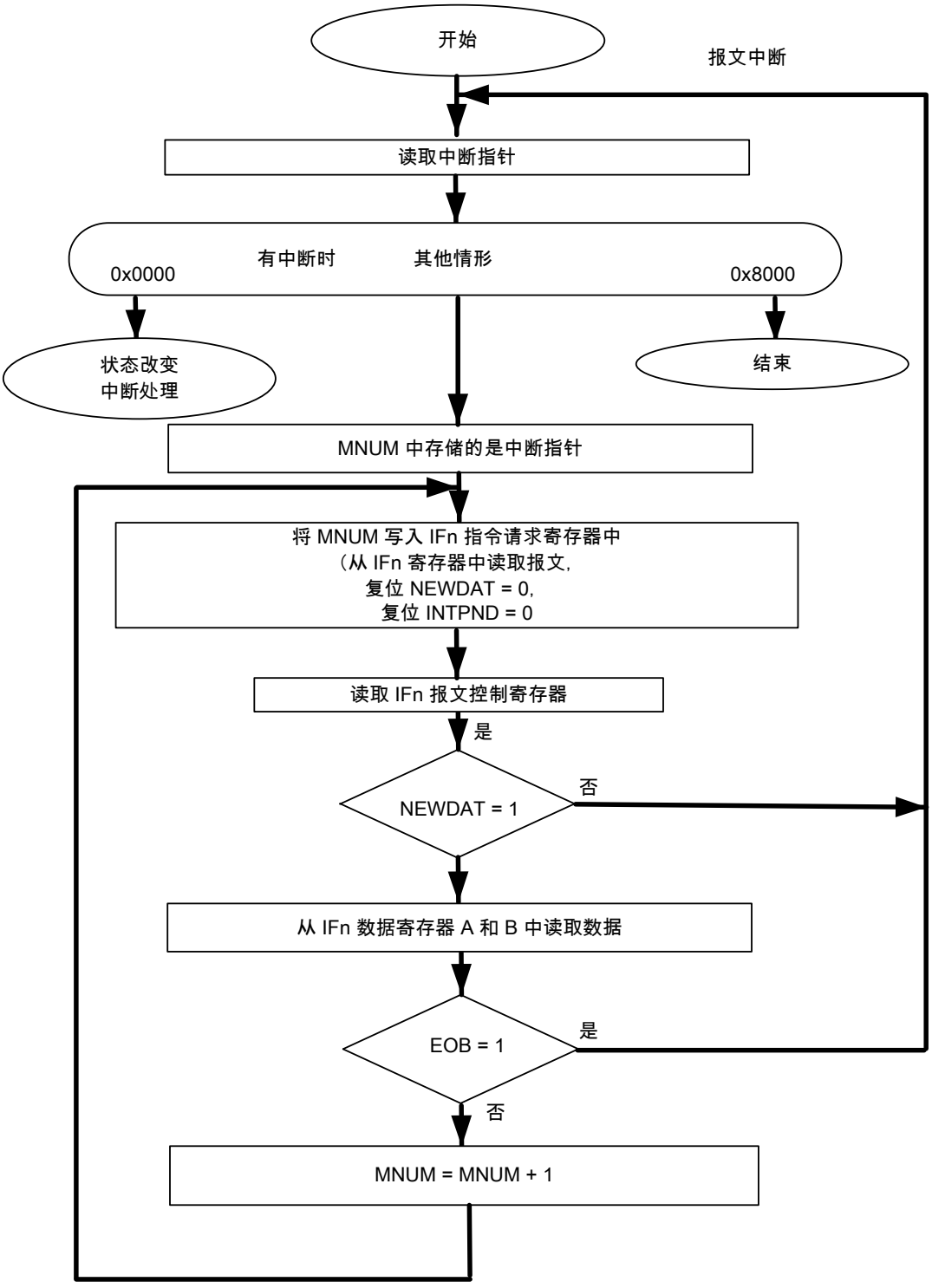
17.3.11.2 采用FIFO缓冲区的报文接收过程

当接收到若干个标识符与FIFO缓冲区匹配的报文时，这些报文将从序号最低的报文对象开始依次写入FIFO缓冲区内。当报文保存到 FIFO 缓冲区中的某个报文对象中时，该报文对象的 CANIFnMCTL 寄存器的 NEWDAT 位便会置位。若在 EOB 清零时将 NEWDAT 位置位，该报文对象将自动锁定，此后报文处理器将无法再对其写入，直到 CPU 将 NEWDAT 位清零后才会解除锁定状态。匹配的报文将依次保存入FIFO缓冲区中，一直到此FIFO缓冲区的最后一个报文对象。当 FIFO 缓冲区填满后，除非已经通过将 NEWDAT 位清零解除锁定前面的所有报文对象，否则所有后来收到的匹配的报文都将写入最后一个报文对象中，因此最后一个报文对象的内容可能会被反复覆盖。

17.3.11.3 FIFO缓冲区的读取

当 CPU 向 CANIFnCRQ 寄存器写入 FIFO 缓冲区中某个报文对象编号以传输其内容时，应将 CANIFnCMSK 寄存器的 TXRQST 和 CLRINTPND 位置位，这样，CANIFnMCTL 寄存器的 NEWDAT 和 INTPEND 位将在读取后自动清零。CANIFnMCTL 寄存器中的这两个位将始终反映报文对象的当前状态，直到用户手动将其清零。要确保FIFO缓冲区能够正确工作，CPU应当按照序号由小到大的顺序依次读取各报文对象的内容。当从 FIFO 缓冲区中读取数据时，用户必须明白：此时接收到的新报文将写入 CANIFnMCTL 寄存器的 NEWDAT 位已被清零的编号最小的报文对象中。因此这之后的FIFO中接收报文的顺序是无法得到保障的。图17-3（956页）描述了 CPU 可如何处理连续进入 FIFO 缓冲区中的一组报文对象。

图 17-3. FIFO 缓冲区中的报文对象



17.3.12 中断的处理

如果多个中断被挂起，CAN 中断 (CANINT) 寄存器将指向优先级最高的挂起中断，而不考虑它们的时间顺序。状态中断具有最高优先级。就报文中断而言，编号最小的报文对象中断具有最高优先级。

通过将相应报文对象的 CANIFnMCTL 寄存器的 INTPND 位清零，或读取 CAN 状态 (CANSTS) 寄存器即可将报文中断清除。状态中断可通过读取 CANSTS 寄存器予以清除。

CANINT 寄存器中的中断标识符 INTID 可表明中断原因。当没有任何中断挂起时，该寄存器的回读值为0x0000。只要 INTID 位域的值不是 0，即表明有中断挂起。如果将 CANCTL 寄存器的 IE 位置位，则中断控制器的中断线将激活。中断线将保持激活状态，直到 INTID 域为 0（所有中断源都已清除：中断的条件已经解除）或 IE 位被清零（也就是禁用 CAN 控制器的中断）时才会取消。

CANINT 寄存器的 INTID 域将指向挂起的具有最高优先级的报文中断。CANCTL 寄存器的 SIE 位用于控制 CANSTS 寄存器中 RXOK、TXOK 和 LEC 位的改变是否会导致中断。CANCTL 寄存器的 EIE 位用于控制 CANSTS 寄存器中 BOFF 和 EWARN 位的改变是否会导致中断。CANCTL 寄存器中的 IE 位用于控制 CAN 控制器产生的中断是否会导致中断控制器中断。即使将 CANCTL 寄存器的 IE 清零，CANINT 寄存器也将自动刷新，不过向 CPU 指出产生了中断。

当 CANINT 寄存器的值为 0x8000 时，表示 CAN 模块刷新了 CANSTS 寄存器（但并不一定改变了其内容）并挂起了中断，因此产生的中断是错误中断或状态中断。对 CANSTS 寄存器执行写入操作即可将其中的 RXOK、TXOK 和 LEC 位清零；但如果想要清除状态中断源，则唯一的办法是读取 CANSTS 寄存器。

在进行中断处理时可通过以下两种方法确定中断源：一种方法是通过读取 CANINT 寄存器的 INTID 位，确定目前挂起的最高优先级中断；另一种方法是通过读取 CAN 报文中断挂起 (CANMSGnINT) 寄存器，查看所有拥有挂起中断的报文对象。

负责读取作为中断源的报文的中断服务程序在读取报文时还可能同时将 CANIFnCMSK 寄存器的 CLRINTPND 位置位，以将报文对象的 INTPND 位清零。一旦 INTPND 位被清零，CANINT 寄存器的内容将包含下一个挂起中断的报文对象编号。

17.3.13 测试模式

CAN模块提供测试模式，方便进行各种诊断。将 CANCTL 寄存器的 TEST 位置位即可进入测试模式。进入测试模式后，可通过 CAN 测试 (CANTST) 寄存器的 TX[1:0]、LBACK、SILENT 和 BASIC 等位域将 CAN 控制器配置成以不同的诊断模式工作。CANTST 寄存器的 RX 位能够监控 CANnRX 管脚状态。若 TEST 位清零，则 CANTST 寄存器的所有功能全部被禁用。

17.3.13.1 安静模式

安静模式下CAN控制器不会发送显性位（确认位、错误帧等等），因此在分析CAN总线数据流的同时不会对CAN总线通讯造成任何影响。将 CANTST 寄存器的 SILENT 位置位即可进入安静模式。在安静模式下，CAN控制器能够接收有效的数据帧和远程帧，但本身只能在CAN总线上发送隐性位，并且无法发送任何报文。即使CAN控制器必须发送显性位（确认位、过载位或有效错误标志位），也仅在控制器内部连接。也就是说只有CAN控制器本身能够监控到发送的显性位，在CAN总线上仍然发送的是隐形位。

17.3.13.2 回送模式

回送模式用于实现自检功能。在回送模式下，CAN 控制器在内部将 CANnTX 和 CANnRX 管脚连接到一起，将自己发送的报文视为接收到的报文并存储到报文缓冲器中（假如这些报文能够顺利通过验收滤波）。将 CANTST 寄存器的 LBACK 位置位即可让 CAN 控制器进入回送模式。为了彻底隔绝外部事件的影响，在回送模式下，CAN 控制器将会忽略应答错误（即在数据帧/远程帧的应答时间内采样到的隐性位）。CANnRX 管脚的实际值会被 CAN 控制器忽略。不过，CANnTX 管脚上仍然能够监控到发送的报文。

17.3.13.3 回送+安静模式

如果混合使用回送模式与安静模式，CAN 控制器既能实现在线自检、也不会影响与 CANnTX 和 CANnRX 管脚连接的 CAN 总线系统的运行。在这种模式下，CANnRX 管脚并不与 CAN 控制器相

连，并且 CANnTX 管脚始终保持隐性位状态。将 CANTST 寄存器的 LBACK 和 SILENT 位同时置位即可进入这种模式。

17.3.13.4 基本模式

所谓基本模式就是CAN控制器不使用报文RAM的工作模式。在基本模式下，CANIF1寄存器组用作发送缓冲。将 CANIF1CRQ 寄存器的 BUSY 位置位即会请求发送 IF1 寄存器组的内容。当 BUSY 位置位时，CANIF1 寄存器即被锁定。此时，BUSY 位可用于指示挂起的发送。只要CAN总线有空闲，CANIF1寄存器组就会把内容载入CAN控制器的移位寄存器并开始发送。发送结束后，BUSY 位将自动清零，并且 CANIF1 寄存器组也将被解除锁定状态。若当前有发送挂起且 CANIF1 寄存器组被锁定，随时都能通过将 CANIF1CRQ 寄存器的 BUSY 位清零而中止发送。如果报文因为丢失仲裁或者发送出错而等待重发，只要 CPU 将 BUSY 位清零，就能禁止自动重发此报文。

CANIF2寄存器组用作接收缓冲。当接收一条报文后，移位寄存器中的内容将不经过任何验收过滤、直接保存到CANIF2寄存器组中。此外，在报文传输期间可以监视移位寄存器的实际内容。每当 CANIF2CRQ 寄存器的 BUSY 位置位（开始读取报文对象）时，移位寄存器的内容都将保存到 CANIF2 寄存器组中。

在基本模式下，所有与报文对象相关的控制位、状态位，以及 CANIFnCMSK 寄存器的控制位都无意义。CANIFnCRQ 寄存器的报文编号也无意义。在 CANIF2MCTL 寄存器中，NEWDAT 和 MSGLST 位保持其原有功能，DLC[3:0] 位域能显示出收到的 DLC，其余控制位均清零。

将 CANTST 寄存器的 BASIC 位置位，即可启用基本模式。

17.3.13.5 发送控制

软件可以直接覆盖 CANnTX 管脚的控制方式，有如下 4 种不同的方式可选：

- CANnTX 由 CAN 控制器控制；
- CANnTX 管脚按照采样点驱动输出，用以监控位时序；
- CANnTX 驱动输出低电平；
- CANnTX 驱动输出高电平；

最后两项功能结合可读的 CAN 接收管脚 CANnRX，可用于校验 CAN 总线物理层是否工作正常。

发送控制功能是通过 CANTST 寄存器的 TX[1:0] 位域进行选择的。CANnTX 信号的三种测试功能可连接所有 CAN 协议功能。TX[1:0] 位必须清零（当选择 CAN 报文传输、回送模式、安静模式或基本模式时）。

17.3.14 位定时配置错误的注意事项

如果在配置CAN的位定时参数时出现微小的错误，即使其并未立即体现出总线故障，也可能会严重降低CAN总线的性能。在很多情况下，CAN总线本身的位同步功能将会掩盖CAN位定时参数配置的不当，将其影响降低到只会偶尔产生错误帧的程度。但在仲裁（竞争总线）时，当两个或两个以上的CAN节点同时试图发送帧时，偏移的采样点可能导致其中一个收发器进入被动错误状态。这种偶发错误非常难于分析，需要对CAN节点内的位同步原理以及CAN节点与CAN总线的相互作用有详尽的了解才能予以断定。

17.3.15 位时间与位速率

CAN总线支持的位速率从1kbps到1000kbps不等。CAN网络中的每个成员都有自己的时钟发生电路。即使各CAN节点所采用的振荡器频率并不相同，也能够通过单独设置各个CAN节点的位定时参数，最终实现位速率的统一。

由于温度/电压波动以及器件本身老化导致频率总会存在微小偏差，因此这些振荡器不可能始终保持稳定。不过，只要偏差保持在指定的振荡器容差范围内，CAN节点都可通过定期与总线上的比特流重新同步的方式为各种位速率进行纠偏。

根据 CAN 规范，位时间分成 4 个时间段（请参阅图17-4（959页））：同步段、传播时间段、相位缓冲段 1 和相位缓冲段 2。每个段由具体、可编程的时间份额数量组成（请参阅表17-3（959页））。时间份额是位时间的基本单位。时间份额的长度 (t_q) 由 CAN 控制器的输入时钟 (fsys) 以及波特率预分频器 (BRP) 共同确定：

t_q = BRP / fsys

其中 fsys 输入时钟源就是通过 RCC 或 RCC2 寄存器设置的系统时钟频率（请参阅221页或227页）。

同步段 (Sync) 是位时间的组成部分，在同步段内期望产生 CAN 总线电平的跳变沿；若跳变沿位于 Sync 之外，则将跳变沿与 Sync 的间隔称为该跳变沿的相位误差。

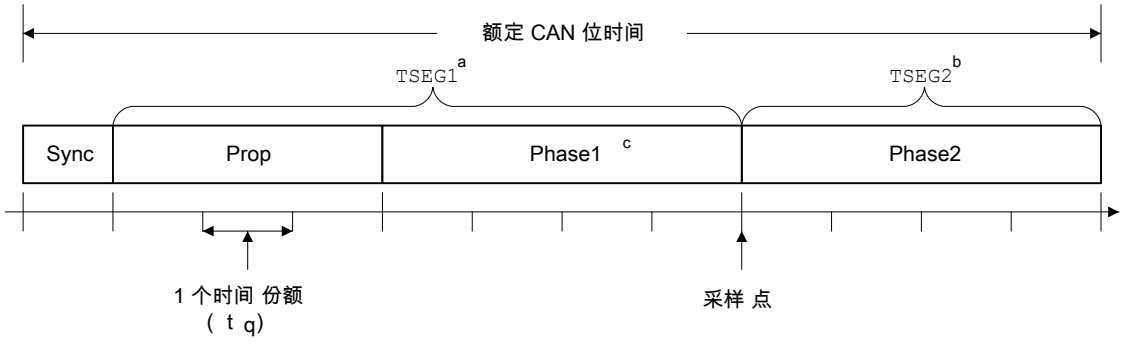
传播段（Prop）用于补偿CAN网络中的物理延迟时间。

相位缓冲段1和相位缓冲段2共同决定采样点在位时间中的位置。

同步跳转宽度（Synchronization Jump Width，简写为SJW）定义了再同步的跳转距离，当出现跳变沿相位误差时，可按该距离在相位缓冲段的有效范围内移动采样点进行补偿。

当给定位速率后，可能有多种不同的位时间配置均可满足速率的要求；但是为了保障CAN网络正常工作，还应当慎重考虑并计算物理延迟时间以及振荡器的容差范围。

图 **17-4. CAN** 的位时间



a. TSEG1 = Prop + Phase1
b. TSEG2 = Phase2
c. Phase1 = Phase2 或 Phase1 + 1 = Phase2

表 **17-3. CAN** 协议范围[a]

参数符号	有效范围	注释
BRP	[1 ..64]	定义时间份额 t_q 的长度。通过 CANBRPE 寄存器可将此范围扩展到 1024。
Sync	1 t_q	固定长度，将总线输入与系统时钟进行同步
Prop	[1 ..8] t_q	补偿CAN总线传输时的物理延时
Phase1	[1 ..8] t_q	根据同步的情况可能会适当延长
Phase2	[1 ..8] t_q	根据同步的情况可能会适当缩短
SJW	[1 ..4] t_q	不得超过任一相位缓冲段的长度

a. 该表描述了CAN协议要求的最小可编程范围。

可通过 CANBIT 寄存器中的两个寄存器字节设置位时序。在 CANBIT 寄存器中，TSEG2、TSEG1、SJW 和 BRP 这 4 个位域的值必须填写为其实际取值减去 1；因此虽然上表中定义的有效范围是 [1..n]，实际对寄存器编程值的有效范围是 [0..n-1]。如此一来，举例来说，同步跳转宽度 SJW（有效范围 [1..4]）在 SJW 位域中只需两个位即可表示。表17-4列出了 CANBIT 寄存器位域值与实际参数的关系。

表 **17-4. CANBIT** 寄存器值

CANBIT 寄存器域	设置
TSEG2	Phase2 - 1
TSEG1	Prop + Phase1 - 1
SJW	SJW - 1
BRP	BRP

因此，位时间的长度为（以编程值计算）：

$[TSEG1 + TSEG2 + 3] \times t_q$

或以实际值表述为：

$[Sync + Prop + Phase1 + Phase2] \times t_q$

CANBIT 寄存器中的数值将作为 CAN 协议控制器的配置输入。波特率预分频器（由 BRP 位域配置）定义了每个时间份额（位时间的基本时间单元）的长度；位时序逻辑（由 TSEG1、TSEG2 和 SJW 配置）定义了每个位时间中包含多少个时间份额。

位时间的处理、采样点的位置计算以及偶发的同步动作均由CAN控制器控制，并且在每个时间份额都要进行判定。

CAN控制器负责将报文转译为帧以及将帧转译为报文。此外，控制器还要负责以下任务：产生/丢弃帧中的固定格式位；插入/解出填充位；计算/校验CRC校验码；执行错误管理；确定采用何种类型的同步。一个位的值是在采样点接收或发送的。信息处理时间 (IPT) 是采样点后控制器做好向 CAN 总线发送下一个位所需的时间。IPT 期间控制器需要进行的工作包括：获取下一个数据位、处理CRC位、判断是否需要位填充以及根据需要产生错误标志或进入空闲状态。

IPT 根据应用程序的不同而不同，但不会长于 2 t_q；CAN 的 IPT 是 0 t_q。IPT长度应作为是Phase2可编程长度的下限值。当发生再同步时，缩短后的Phase2可能会小于IPT，但这并不影响总线时序。

17.3.16 位定时参数的计算

通常在计算位定时参数时，首先应当确定需要达到的位速率或位时间。位时间（即位速率的倒数）必须是系统时钟周期的整数倍。

每个位时间由4到25个时间份额所组成。通过将各个时间段进行组合，就能得到所需的位时间，即重复以下步骤。

首先要确定Prop段。Prop段的长度将决定CAN总线的最大延迟时间，反之也可以根据CAN总线的实际最大延迟时间来确定其长度。为保证CAN总线系统具有可扩展性，应明确定义最长总线距离下节点间的最大延时。Prop 的最终时间应转换为时间份额（舍入到最接近的整数值，即 t_q 的整数倍）。

Sync 段的长度是 1 个 t_q（固定），从而两个相位缓冲段的长度为（位时间 - Prop - 1）t_q。假如剩余的 t_q 数是偶数值，则将平均分配给两个相位缓冲段，即 Phase2 = Phase1；假如是奇数值，则 Phase2 = Phase1 + 1。

还必须考虑 Phase2 的最小额定长度。Phase2 不能比 CAN 控制器的信息处理时间短。其长度范围为 [0..2] t_q，具体视实际的执行情况而定。

同步跳转宽度可选择以下3个值中的最小值：4、Phase1、Phase2。

确定了以上配置参数后，即可根据下面的公式计算振荡器容差范围：

$$(1 - df) \times fnom \leq fosc \leq (1 + df) \times fnom$$

此处，

$$df \leq \frac{(Phase_seg1, Phase_seg2)\min}{2 \times (13 \times tbit - Phase_Seg2)}$$

- df = 振荡器频率的最大容限
- fosc = 实际的振荡器频率
- fnom = 额定的振荡器频率

最大频率范围必须符合下等式

$$df_{max} = 2 \times df \times fnom$$

此处，

- Phase1 和 Phase2 取自 表17-3（959页）
- tbit = 位时间
- dfmax = 两个振荡器之间的最大差异

假如可选的配置参数不止一组，那么应当优先选用振荡器容差范围最大的一组参数。

如果多个系统时钟频率不同的CAN节点需要达到相同的位速率，需要对其各自进行不同的配置。在计算传输时间时，应在整个CAN网络内选取距离最远的节点计算最大延迟时间。

整个CAN系统的振荡器容差范围将由系统中容差范围最小的节点所决定（木桶原理）。

以上计算结果表明：要想得到与协议兼容的位定时参数配置，有时必须在诸多因素中进行取舍；要么必须减小总线长度或位速率，要么必须提高振荡器频率的稳定性。

17.3.16.1 高波特率下的位定时参数计算

在此示例中，假设 CAN 模块输入时钟为 25 MHz，CAN 总线位速率为 1 Mbps。

```
bit time = 1 µs = n * tq = 5 * tq
tq = 200 ns
tq = (Baud rate Prescaler)/CAN Clock
Baud rate Prescaler = tq * CAN Clock
Baud rate Prescaler = 200E-9 * 25E6 = 5

tSync = 1 * tq = 200 ns                \\fixed at 1 time quanta

delay of bus driver 50 ns
```

```
delay of receiver circuit 30 ns
delay of bus line (40m) 220 ns
tProp 400 ns = 2 * tq                  \\400 is next integer multiple of tq

bit time = tSync + tTSeg1 + tTSeg2 = 5 * tq
bit time = tSync + tProp + tPhase 1 + tPhase2
tPhase 1 + tPhase2 = bit time - tSync - tProp
tPhase 1 + tPhase2 = (5 * tq) - (1 * tq) - (2 * tq)
tPhase 1 + tPhase2 = 2 * tq
tPhase1 = 1 * tq
tPhase2 = 1 * tq                       \\tPhase2 = tPhase1

tTSeg1 = tProp + tPhase1
tTSeg1 = (2 * tq) + (1 * tq)
tTSeg1 = 3 * tq

tTSeg2 = tPhase2
tTSeg2 = (Information Processing Time + 1) * tq
tTSeg2 = 1 * tq                         \\Assumes IPT=0

tSJW = 1 * tq                          \\Least of 4, Phase1 and Phase2
```

在上面的例子中，CANBIT 寄存器各位域的值为：

TSEG2	= TSeg2 -1 = 1-1 = 0
TSEG1	= TSeg1 -1 = 3-1 = 2
SJW	= SJW -1 = 1-1 = 0
BRP	= 波特率预分频器 - 1 = 5-1 =4

因此最终写入 CANBIT 寄存器的值是 0x0204。

17.3.16.2 低波特率下的位定时参数计算

在此计算中，假定 CAN 时钟的频率 50 MHz，CAN 总线位速率为 100kbps。

```
bit time = 10 µs = n * tq = 10 * tq
tq = 1 µs
tq = (Baud rate Prescaler)/CAN Clock
Baud rate Prescaler = tq * CAN Clock
Baud rate Prescaler = 1E-6 * 50E6 = 50

tSync = 1 * tq = 1 µs                 \\fixed at 1 time quanta

delay of bus driver 200 ns
```

```
delay of receiver circuit 80 ns
delay of bus line (40m) 220 ns
tProp 1 µs = 1 * tq                    \\1 µs is next integer multiple of tq

bit time = tSync + tTSeg1 + tTSeg2 = 10 * tq
bit time = tSync + tProp + tPhase 1 + tPhase2
tPhase 1 + tPhase2 = bit time - tSync - tProp
tPhase 1 + tPhase2 = (10 * tq) - (1 * tq)  - (1 * tq)
tPhase 1 + tPhase2 = 8 * tq
tPhase1 = 4 * tq
tPhase2 = 4 * tq                       \\tPhase1 = tPhase2

tTSeg1 = tProp + tPhase1
tTSeg1 = (1 * tq) + (4 * tq)
tTSeg1 = 5 * tq
tTSeg2 = tPhase2
tTSeg2 = (Information Processing Time + 4) × tq
tTSeg2 = 4 * tq                        \\Assumes IPT=0

tSJW = 4 * tq                          \\Least of 4, Phase1, and Phase2
```

TSEG2	= TSeg2 -1 = 4-1 = 3
TSEG1	= TSeg1 -1 = 5-1 = 4
SJW	= SJW -1 = 4-1 = 3
BRP	= 波特率预分频器 - 1 = 50-1 =49

因此最终写入 CANBIT 寄存器的值是 0x34F1。

17.4 寄存器映射

表 17-5（963页）列出了各寄存器。表中偏移量一列是指相对于CAN基地址的十六进制地址增量，各个CAN模块的基地址分别为：

- CAN0：0x4004.0000

请注意，在设置寄存器之前，应先启用 CAN 控制器时钟（请参阅305页）。启用 CAN 模块时钟后，必须要延迟 3 个系统时钟才可以访问 CAN 模块寄存器。

表 **17-5. CAN** 寄存器映射

偏移量	名称	类型	复位	描述	见页面
0x000	CANCTL	R/W	0x0000.0001	CAN 控制寄存器	966
0x004	CANSTS	R/W	0x0000.0000	CAN 状态寄存器	968

表 17-5. CAN 寄存器映射（续）

偏移量	名称	类型	复位	描述	见页面
0x008	CANERR	RO	0x0000.0000	CAN 错误计数寄存器	970
0x00C	CANBIT	R/W	0x0000.2301	CAN 位时序寄存器	971
0x010	CANINT	RO	0x0000.0000	CAN 中断寄存器	972
0x014	CANTST	R/W	0x0000.0000	CAN 测试寄存器	973
0x018	CANBRPE	R/W	0x0000.0000	CAN 波特率预分频器扩展寄存器	975
0x020	CANIF1CRQ	R/W	0x0000.0001	CAN IF1 指令请求寄存器	976
0x024	CANIF1CMSK	R/W	0x0000.0000	CAN IF1 指令屏蔽寄存器	977
0x028	CANIF1MSK1	R/W	0x0000.FFFF	CAN IF1 屏蔽寄存器 1	980
0x02C	CANIF1MSK2	R/W	0x0000.FFFF	CAN IF1 屏蔽寄存器 2	981
0x030	CANIF1ARB1	R/W	0x0000.0000	CAN IF1 仲裁寄存器 1	982
0x034	CANIF1ARB2	R/W	0x0000.0000	CAN IF1 仲裁寄存器 2	983
0x038	CANIF1MCTL	R/W	0x0000.0000	CAN IF1 报文控制寄存器	985
0x03C	CANIF1DA1	R/W	0x0000.0000	CAN IF1 数据寄存器 A1	988
0x040	CANIF1DA2	R/W	0x0000.0000	CAN IF1 数据寄存器 A2	988
0x044	CANIF1DB1	R/W	0x0000.0000	CAN IF1 数据寄存器 B1	988
0x048	CANIF1DB2	R/W	0x0000.0000	CAN IF1 数据寄存器 B2	988
0x080	CANIF2CRQ	R/W	0x0000.0001	CAN IF2 指令请求寄存器	976
0x084	CANIF2CMSK	R/W	0x0000.0000	CAN IF2 指令屏蔽寄存器	977
0x088	CANIF2MSK1	R/W	0x0000.FFFF	CAN IF2 屏蔽寄存器 1	980
0x08C	CANIF2MSK2	R/W	0x0000.FFFF	CAN IF2 屏蔽寄存器 2	981
0x090	CANIF2ARB1	R/W	0x0000.0000	CAN IF2 仲裁寄存器 1	982
0x094	CANIF2ARB2	R/W	0x0000.0000	CAN IF2 仲裁寄存器 2	983
0x098	CANIF2MCTL	R/W	0x0000.0000	CAN IF2 报文控制寄存器	985
0x09C	CANIF2DA1	R/W	0x0000.0000	CAN IF2 数据寄存器 A1	988
0x0A0	CANIF2DA2	R/W	0x0000.0000	CAN IF2 数据寄存器 A2	988
0x0A4	CANIF2DB1	R/W	0x0000.0000	CAN IF2 数据寄存器 B1	988
0x0A8	CANIF2DB2	R/W	0x0000.0000	CAN IF2 数据寄存器 B2	988
0x100	CANTXRQ1	RO	0x0000.0000	CAN 传输请求寄存器 1	989
0x104	CANTXRQ2	RO	0x0000.0000	CAN 传输请求寄存器 2	989
0x120	CANNWDA1	RO	0x0000.0000	CAN 新数据寄存器 1	990
0x124	CANNWDA2	RO	0x0000.0000	CAN 新数据寄存器 2	990
0x140	CANMSG1INT	RO	0x0000.0000	CAN 报文 1 中断挂起寄存器	991
0x144	CANMSG2INT	RO	0x0000.0000	CAN 报文 2 中断挂起寄存器	991

表 17-5. CAN 寄存器映射（续）

偏移量	名称	类型	复位	描述	见页面
0x160	CANMSG1VAL	RO	0x0000.0000	CAN 报文 1 有效寄存器	992
0x164	CANMSG2VAL	RO	0x0000.0000	CAN 报文 2 有效寄存器	992

17.5 寄存器描述

本章的剩余部分按照地址偏移量由小到大的顺序依次详细介绍各寄存器。CAN 模块提供两组接口寄存器以用于访问报文 RAM 中的报文对象：CANIF1x 和 CANIF2x。这两组寄存器的功能完全相同，可用于实现连续会话。

寄存器 1: CAN 控制寄存器（CANCTL），偏移量 0x000

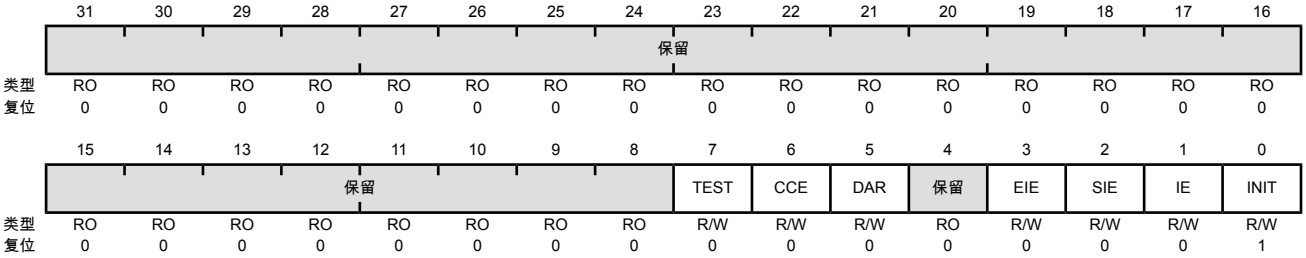
此控制寄存器用于启动CAN模块、使能测试模式以及使能中断。

即使置位或清零 INIT 位，也无法缩短离线恢复序列（见 CAN 2.0 规范）。器件一旦进入离线状态便会将 INIT 位置位，并终止所有总线活动。当 CPU 将 INIT 位清零后，器件需等待总线出现 129 个空闲态后（129 * 11 个连续隐形位）才能恢复正常的总线操作。离线恢复序列结束时，错误管理计数器都会复位。

在 INIT 清零后的等待期间，每当监视到 11 个连续的高位后，都会向 CANSTS 寄存器写入一个 BITERROR0 错误码（LEC 位域 = 0x5），这样 CPU 随时可以检查 CAN 总线是否被拉低或受到持续干扰，同时还能监视离线恢复序列的进度。

CAN 控制寄存器 (CANCTL)

CAN0 基址: 0x4004.0000
偏移量 0x000
类型 R/W, 复位 0x0000.0001



位/域	名称	类型	复位	描述
31:8	保留	RO	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
7	TEST	R/W	0	测试模式启用 值 描述 0 CAN控制器工作在正常模式 1 CAN控制器工作在测试模式
6	CCE	R/W	0	配置修改使能 值 描述 0 禁止对 CANBIT 寄存器写操作 1 若 INIT 位为 1，则允许对 CANBIT 寄存器进入写入操作。
5	DAR	R/W	0	禁用自动重发功能 值 描述 0 允许自动重发发送失败的报文 1 禁用自动重发，报文均只尝试发送一次
4	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。

位/域	名称	类型	复位	描述
3	EIE	R/W	0	出错中断启用 值 描述 0 不产生错误中断 1 每当 CANSTS 寄存器的 BOFF 或 EWARN 标志位发生变化时，将产生一个中断。
2	SIE	R/W	0	状态中断启用 值 描述 0 不产生状态中断 1 成功发送或接收报文或检测到 CAN 总线错误时，即产生一个中断。当 CANSTS 寄存器的 TXOK、RXOK 或 LEC 位发生变化时，将产生一个中断。
1	IE	R/W	0	CAN 中断启用 值 描述 0 中断禁用。 1 中断启用。
0	INIT	R/W	1	初始化 值 描述 0 正常工作 1 初始化开始。

寄存器 2: CAN 状态寄存器（CANSTS），偏移量0x004

重要: 本寄存器为读敏感型寄存器。有关详细信息，请参阅寄存器描述部分。

状态寄存器中包含中断服务所需的信息，例如离线状况、错误计数门限以及错误类型等等。

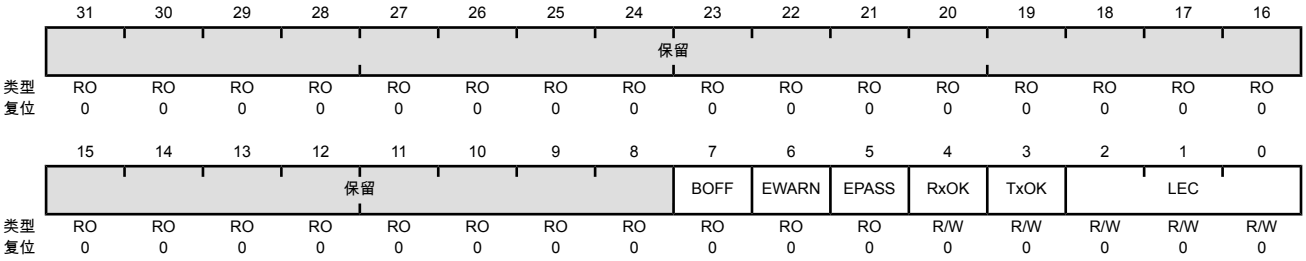
LEC 位域保存的代码能够指示 CAN 总线上一次错误的类型。当成功完成一个报文的传输（接收或发送）后，此位域将自动清除。CPU可以向此位域写入未用的错误码0x7，将来此位域发生变化时可以很容易发现。

若 CAN 控制 (CANCTL) 寄存器中的 BOFF 或 EWARN 位置位，即可产生错误中断；若 RXOK、TXOK 或 LEC 位置位，即可产生状态中断。EPASS 位发生变化或者对 RXOK、TXOK 或 LEC 位的写操作都不会产生中断。

假如有中断挂起，读取 CAN 状态 (CANSTS) 寄存器还会清除 CAN 中断 (CANINT) 寄存器。

CAN 状态寄存器 (CANSTS)

CAN0 基址: 0x4004.0000
偏移量 0x004
类型 R/W, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:8	保留	RO	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
7	BOFF	RO	0	脱离总线状态 值 描述 0 CAN控制器未处于离线状态 1 CAN控制器处于离线状态
6	EWARN	RO	0	警告状态 值 描述 0 两组错误计数器都低于错误警告门限（96次） 1 至少有一组错误计数器已达到错误警告门限（96次）
5	EPASS	RO	0	错误认可 值 描述 0 CAN模块处于主动错误状态，也就是说，接收错误计数或发送错误计数都小于等于127 1 CAN模块处于被动错误状态，也就是说，接收错误计数或发送错误计数至少有一个大于127

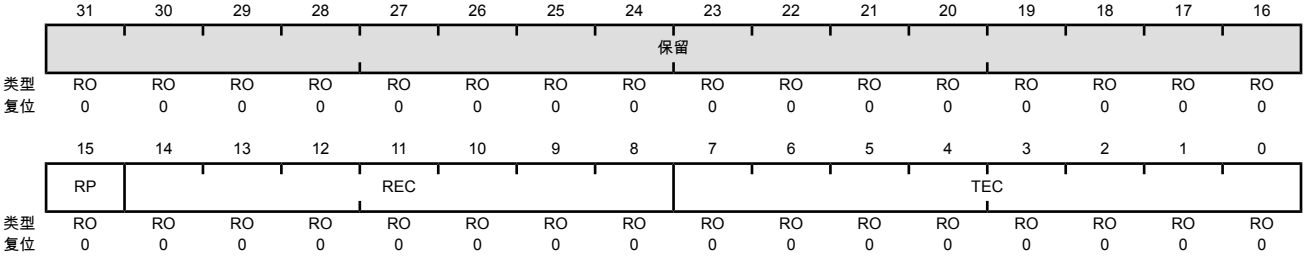
位/域	名称	类型	复位	描述
4	RxOK	R/W	0	接收报文成功 值 描述 0 从本标志位上一次清零以来，并未成功接收到新的报文 1 自从该位上一次清零以来，至少成功接收到一条新的报文。此位与验收滤波结果无关。 本标志位必须手动写0予以清除。
3	TxOK	R/W	0	发送报文成功 值 描述 0 自从该位上一次清零以来，没有报文发送成功。 1 从本标志位上一次清零以来，至少成功发送了一条新的报文，没有错误并且至少有一个节点确认 本标志位必须手动写0予以清除。
2:0	LEC	R/W	0x0	上一次的错误代码 此位域包含CAN总线上次发生的错误代码 值 描述 0x0 无错误 0x1 填充错误 一个序列中有超过5个相同极性的位出现在接收报文中，这是接收报文所不能允许的。 0x2 格式错误 接收帧的固定格式部分包含错误格式。 0x3 应答错误 另一节点没有应答发送的报文。 0x4 位 1 错误 在发送报文时，CAN 控制器对数据线进行监控以探测是否存在冲突。在发送仲裁域时，数据冲突是仲裁协议的一部分。在发送其它帧域时，数据冲突被视为错误。 位1错误表示器件想发送高电平（逻辑1），但是监控到的总线值却是低电平（逻辑0）。 0x5 位 0 错误 第0位错误表示器件企图发送低电平（逻辑0），但是监视到的总线电平为高电平（逻辑1）。 在脱离总线恢复期间，每次监控到含 11 个高电平位的序列后，该状态就会置位。通过对该状态的检查，软件可以在干扰总线的情况下对脱离总线恢复序列的进程进行监控。 0x6 CRC 错误 CRC 校验和在接收报文中是错误的，表示所计算得到的接收值与数据计算得到的CRC值不相同。 0x7 无事件发生 当 LEC 位域是这个值时，表明自上一次对 LEC 位域写 0x7 以来，未产生任何 CAN 总线事件。

寄存器 3: CAN 错误计数寄存器（CANERR），偏移量 0x008

本寄存器包含错误计数值，可用于分析错误的成因。

CAN 错误计数寄存器 (CANERR)

CAN0 基址: 0x4004.0000
偏移量 0x008
类型 RO, 复位 0x0000.0000



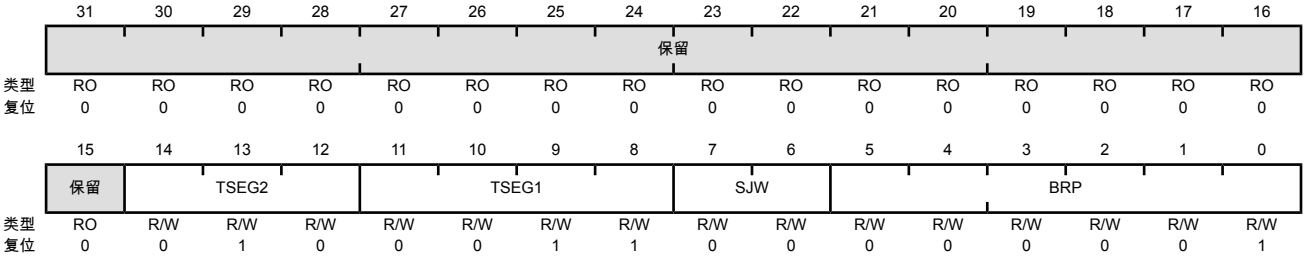
位/域	名称	类型	复位	描述
31:16	保留	RO	0x0000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
15	RP	RO	0	接收到的错误认可 值 描述 0 接收错误计数低于被动错误门限（127或更低） 1 接收错误计数已达到被动错误门限（128或更高）
14:8	REC	RO	0x00	接收错误计数器 此位域包含接收错误计数器的值（0 ~ 127）
7:0	TEC	RO	0x00	发送错误计数器 此位包含发送错误计数器的值（0 到 255）。

寄存器 4: CAN 位时序寄存器（CANBIT），偏移量 0x00C

本寄存器用于定义位时间以及时间份额。写入的数值应按照系统时钟频率进行计算。只有将CANCTL寄存器的CCE位和INIT位置位时，此寄存器才允许写入。更多信息参见“位时间与位速率”（958页）。

CAN 位时序寄存器 (CANBIT)

CAN0 基址: 0x4004.0000
偏移量 0x00C
类型 R/W, 复位 0x0000.2301



位/域	名称	类型	复位	描述
31:15	保留	RO	0x0000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
14:12	TSEG2	R/W	0x2	采样点后的时间段 0x00-0x07：硬件在实际应用中对该值的解释要比此处编程的值大 1。 因此，例如复位 0x2 表示 Phase2 包含 3（即 2 + 1）个位时间份额（请参阅图17-4（959页））。位时间份额由 BRP 域定义。
11:8	TSEG1	R/W	0x3	采样点前的时间段 0x00-0x0F：硬件在实际应用中对该值的解释要比此处编程的值大 1。 因此，例如复位 0x3 表示 Phase1 包含 4（即 3 + 1）个位时间份额（请参阅图17-4（959页））。位时间份额由 BRP 域定义。
7:6	SJW	R/W	0x0	（再）同步跳转宽度 0x00-0x03：硬件在实际应用中对该值的解释要比此处编程的值大 1。 在帧起始 (SOF) 过程中，如果 CAN 控制器检测到相位误差（偏差），它可以通过改变 SJW 的值来调整 TSEG2 或 TSEG1 的长度因此复位 0 调节了 1 个位时间份额的长度。
5:0	BRP	R/W	0x1	波特率预分频系数 该值是通过对振荡器频率分频获得的，用于产生位时间份额。位时间由多个这种份额组成。 0x00-0x3F：硬件在实际应用中对该值的解释要比此处编程的值大 1。 BRP 位域定义每个时间份额由多少个 CAN 时钟周期组成，因此复位值是 2 个时间份额 (1+1)。 CANBRPE 寄存器可对位时间进行进一步细分。

寄存器 5: CAN 中断寄存器（CANINT），偏移量 0x010

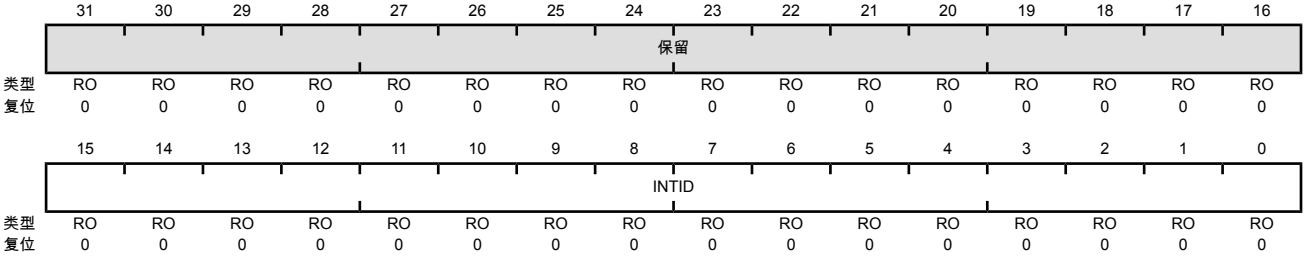
本寄存器包含的内容是中断源。

假如多个中断同时挂起，则 CAN 中断 (CANINT) 寄存器始终指向挂起的最高优先级中断，而并非按照中断产生的顺序。中断在被CPU清除之前将保持挂起。假如 INTID 位域的值不是 0x0000（默认值），并且 CANCTL 寄存器的 IE 位置位，则该中断就是激活的。在 INTID 位域被清除（通过读取 CANSTS 寄存器）之前，或在 CANCTL 寄存器的 IE 位清零之前，中断线将始终保持激活。

注意: 假如有中断挂起，读取 CAN 状态 (CANSTS) 寄存器还会清除 CAN 中断 (CANINT) 寄存器。

CAN 中断寄存器 (CANINT)

CAN0 基址: 0x4004.0000
偏移量 0x010
类型 RO, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:16	保留	RO	0x0000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
15:0	INTID	RO	0x0000	中断标识符 此位域中的数值代表中断源。

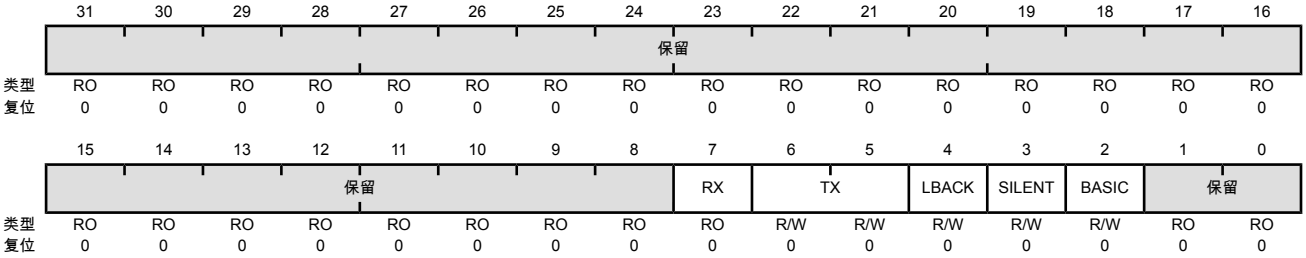
值	描述
0x0000	无中断挂起
0x0001-0x0020	此编号对应的报文对象产生中断
0x0021-0x7FFF	保留
0x8000	状态中断
0x8001-0xFFFF	保留

寄存器 6: CAN 测试寄存器（CANTST），偏移量 0x014

本寄存器用于自检以及访问外部管脚。只有当 CANCTL 寄存器的 TEST 位置位后，才允许对本寄存器进行写操作。用户可以按需组合不同的测试模式，不过应当注意：假如 TX 位域的值不是 0，将可能影响 CAN 发送功能。

CAN 测试寄存器 (CANTST)

CAN0 基址: 0x4004.0000
偏移量 0x014
类型 R/W, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:8	保留	RO	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
7	RX	RO	0	接收观测 值 描述 0 CANnRx 管脚为低电平。 1 CANnRx 管脚为高电平。
6:5	TX	R/W	0x0	发送控制 取代 CANnTx 管脚的控制。 值 描述 0x0 CAN模块控制 CANnTx 由 CAN 模块控制。默认设置。 0x1 采样点 采样点在 CANnTx 信号驱动输出。该模式适用于监控位时序。 0x2 拉低 CANnTx 驱动输出低电平。该模式适用于检查 CAN 总线的线物理层。 0x3 拉高 CANnTx 驱动输出高电平。该模式适用于检查 CAN 总线的线物理层。
4	LBACK	R/W	0	回环模式 值 描述 0 禁用环回模式 1 回送模式启用。在回送模式下，发送器所发出的数据将被路由到接收器。而接收输入将被忽略。

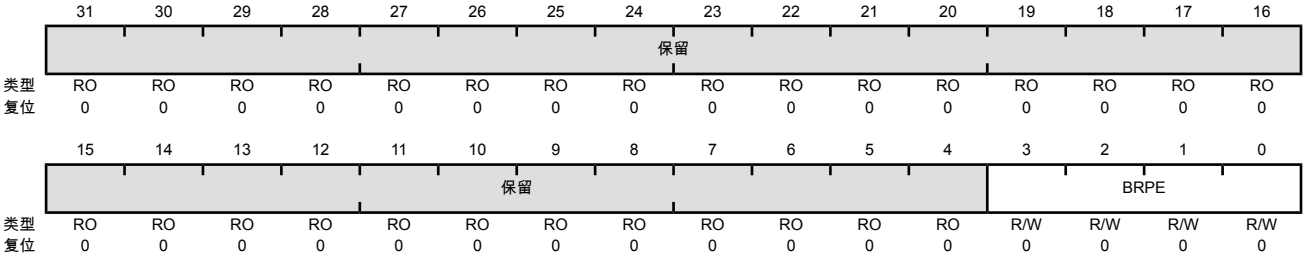
位/域	名称	类型	复位	描述
3	SILENT	R/W	0	安静模式 值 描述 0 安静模式禁用。 1 安静模式启用。在安静模式下，CAN 控制器不发送任何数据，而只监控总线。此模式通常也称为总线监控模式。
2	BASIC	R/W	0	基本模式 值 描述 0 禁用基本模式 1 基本模式启用。在基本模式下，软件应当采用 CANIF1 寄存器作为发送缓冲区，并采用 CANIF2 寄存器作为接收缓冲区。
1:0	保留	RO	0x0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。

寄存器 7: CAN 波特率预分频器扩展寄存器（CANBRPE），偏移量 0x018

本寄存器用于对 CANBIT 寄存器中的 BRP 位划分的位时间进行进一步分频。当 CANCTL 寄存器的 CCE 位置位时，本寄存器即可写入。

CAN 波特率预分频器扩展寄存器 (CANBRPE)

CAN0 基址: 0x4004.0000
偏移量 0x018
类型 R/W, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:4	保留	RO	0x0000.000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
3:0	BRPE	R/W	0x0	波特率预分频系数扩展 0x00-0x0F：最高可将 CANBIT 寄存器中的 BRP 位扩展到 1023。硬件在实际应用中对该位的解释要比 BRPE (MSB) 和 BRP (LSB) 编程的值大 1。

寄存器 8: CAN IF1 指令请求寄存器（CANIF1CRQ），偏移量 0x020

寄存器 9: CAN IF2 指令请求寄存器（CANIF2CRQ），偏移量 0x080

当 CANIF1MCTL 寄存器的 TXRQST 位置位时，如果对本寄存器的 MNUM 域写入某个报文对象的编号，即会启动报文传输。写 MNUM 位域的同时，BUSY 位也会自动置位，表明在 CAN 接口寄存器与内部报文 RAM 之间正在进行数据交换。等待 3 到 6 个 CAN_CLK 周期后，接口寄存器与报文 RAM 之间的传输结束，随后将 BUSY 位清零。

CAN IFn 指令请求寄存器（CANIFnCRQ）

CAN0 基址: 0x4004.0000
偏移量 0x020
类型 R/W, 复位 0x0000.0001

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	保留															
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	BUSY	保留									MNUM					
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W	R/W	R/W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

位/域	名称	类型	复位	描述
31:16	保留	RO	0x0000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
15	BUSY	RO	0	繁忙标志 值 描述 0 当读/写活动结束后，此标志位自动清零 1 当向本寄存器中写入报文对象序号时，此标志位即会自动置位
14:6	保留	RO	0x00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
5:0	MNUM	R/W	0x01	报文序号 在报文 RAM 中的 32 个报文对象中挑出一个进行数据传输。报文对象的编号是 1 到 32。 值 描述 0x00 保留 0 不是有效的报文编号；它被解释为 0x20 或对象 32。 0x01-0x20 报文序号 代表1到32号报文对象 0x21-0x3F 保留 不是有效的报文编号；这些值被替换，并且被解释为0x01-0x1F。

寄存器 10: CAN IF1 指令屏蔽寄存器（CANIF1CMSK），偏移量 0x024

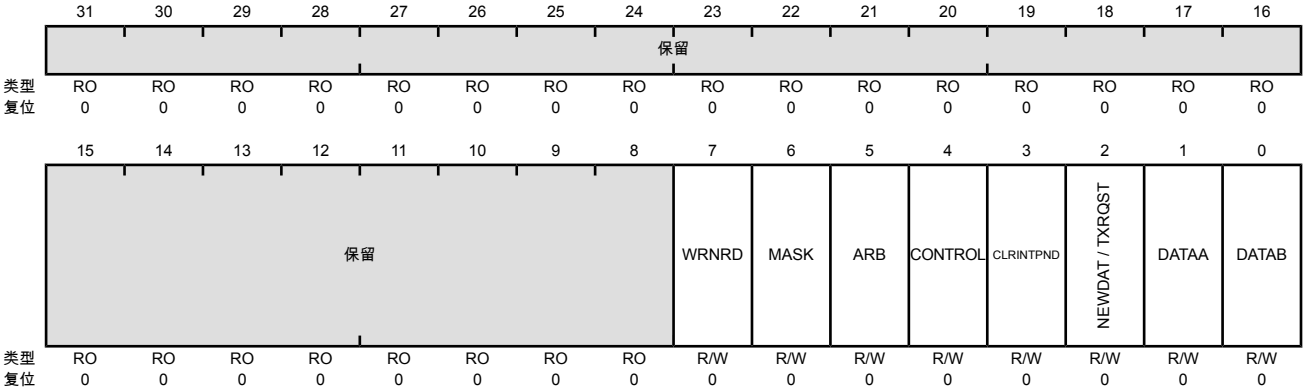
寄存器 11: CAN IF2 指令屏蔽寄存器（CANIF2CMSK），偏移量 0x084

读取指令掩码寄存器，可获取各种功能的当前状态。写入指令掩码寄存器，可以选择传输方向、选择缓冲寄存器组作为数据传输的源或目的。

请注意，若 WRNRD 位清零并且 CLRINTPND及/或 NEWDAT 位置 1，则在读取报文对象缓冲区时，缓冲区中的中断挂起标志及/或新数据标志将被清除。

CAN IFn 指令屏蔽寄存器（CANIFnCMSK）

CAN0 基址: 0x4004.0000
偏移量 0x024
类型 R/W, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:8	保留	RO	0x0000.00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
7	WRNRD	R/W	0	写入，非读取 值 描述 0 将 CANIFnCRQ 寄存器 MNUM 位域所指定的 CAN 报文对象中的数据移动到 CANIFn 寄存器组中。 1 将 CANIFn 寄存器组中的数据移动到 CAN 指令请求 (CANIFnCRQ) 寄存器 MNUM 位所指定的 CAN 报文对象中。 注意: CLRINTPND 和/或 NEWDAT 位置位时，可通过读取报文缓冲区 (WRNRD = 0) 来清除缓冲区中的中断挂起和新数据状况。
6	MASK	R/W	0	访问掩码位 值 描述 0 掩码位无变化 1 将报文对象的 IDMASK + DIR + MXTD 传递给接口寄存器。

位/域	名称	类型	复位	描述
5	ARB	R/W	0	访问仲裁位 值 描述 0 仲裁位无变化 1 将报文对象的 ID + DIR + XTD + MSGVAL 传递给接口寄存器。
4	CONTROL	R/W	0	配置控制位 值 描述 0 控制位无变化 1 将 CANIFnMCTL 寄存器的控制位传递给接口寄存器。
3	CLRINTPND	R/W	0	清除中断挂起位 此标志位的功能取决于 WRNRD 位的设置。 值 描述 0 若 WRNRD 清零，则中断挂起状态从报文缓冲区传递给 CANIFnMCTL 寄存器。 若 WRNRD 置位，报文对象中的 INTPND 位保持不变。 1 若将 WRNRD 位清零，报文缓冲区中的中断挂起状态也会被清除。请注意：传递给 CANIFnMCTL 寄存器的该位的值总是会反映其被清除之前的真实状态。 若 WRNRD 置位，报文对象中的 INTPND 位将被清零。
2	NEWDAT / TXRQST	R/W	0	NEWDAT / TXRQST 位 此标志位的功能取决于 WRNRD 位的设置。 值 描述 0 若 WRNRD 清零，则新数据状态的值将从从报文缓冲区传递给 CANIFnMCTL 寄存器。 若 WRNRD 置位，将不会请求发送。 1 若将 WRNRD 位清零，报文缓冲区中的新数据状态将被清除。请注意：传递给 CANIFnMCTL 寄存器的该位的值总是会反映其被清除之前的真实状态。 若将WRNRD位置位，将会请求发送。请注意：当该位置位时，CANIFnMCTL 寄存器中的 TXRQST 位将被忽略。
1	DATAA	R/W	0	访问数据字节 0 到 3 此标志位的功能取决于 WRNRD 位的设置。 值 描述 0 数据字节0~3不会改变 1 若 WRNRD 清零，则将 CANIFnDA1 寄存器以及 CANIFnDA2 寄存器的数据字节 0-3 传递给报文对象。 若 WRNRD 置位，则将报文对象中的数据字节 0-3 传递给 CANIFnDA1 和 CANIFnDA2 寄存器。

位/域	名称	类型	复位	描述
0	DATAB	R/W	0	访问数据字节 4 到 7 此位的功能取决于 WRNRD 位的配置。 值 描述 0 数据字节4~7不会改变 1 若 WRNRD 清零，则将 CANIFnDA1 寄存器以及 CANIFnDA2 寄存器的数据字节 4-7 传递给报文对象。 若 WRNRD 置位，则将报文对象中的数据字节 4-7 传递给 CANIFnDA1 和 CANIFnDA2 寄存器。

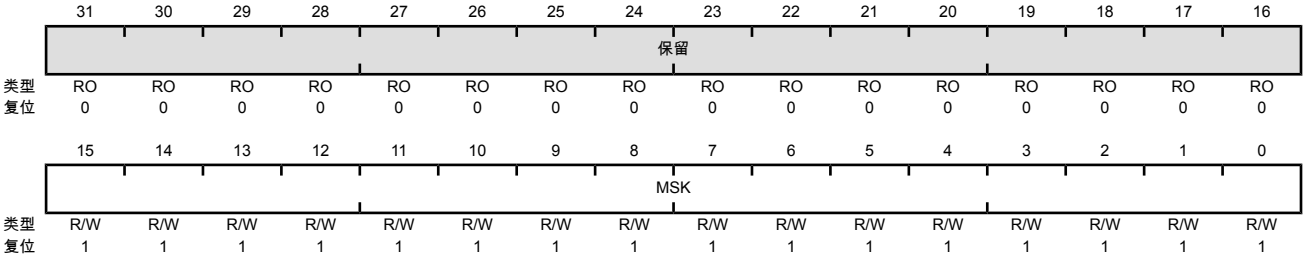
寄存器 12: CAN IF1 屏蔽寄存器 1（CANIF1MSK1），偏移量 0x028

寄存器 13: CAN IF2 屏蔽寄存器 1（CANIF2MSK1），偏移量 0x088

本寄存器中包含掩码信息，以及报文 RAM 中报文对象的数据信息 (CANIFnDAn)、仲裁信息 (CANIFnARBn) 和控制信息 (CANIFnMCTL)。本寄存器中的掩码可结合 CANIFnARBn 寄存器 ID 位域实现验收滤波。其余掩码信息包含在 CANIFnMSK2 寄存器中。

CAN IFn 屏蔽寄存器 1（CANIFnMSK1）

CAN0 基址: 0x4004.0000
偏移量 0x028
类型 R/W, 复位 0x0000.FFFF



位/域	名称	类型	复位	描述
31:16	保留	RO	0x0000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
15:0	MSK	R/W	0xFFFF	标识符掩码 在使用 29 位的标识符时，这些位用作 ID 的 [15:0] 位。CANIFnMSK2 寄存器中的 MSK 域用作 ID 的 [28：16] 位。对于 11 位的报文标识符，本位无意义。 值 描述 0 报文对象中的对应标识符域 (ID) 不能抑制验收滤波中的匹配。 1 报文对象中的对应标识符域 (ID) 用于验收滤波。

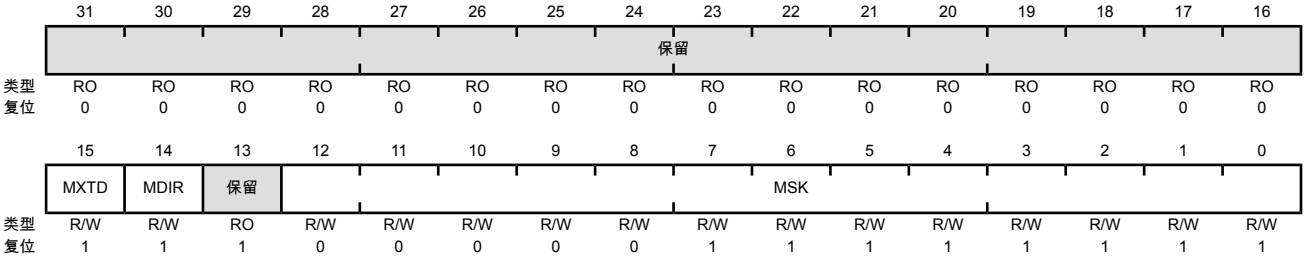
寄存器 14: CAN IF1 屏蔽寄存器 2（CANIF1MSK2），偏移量 0x02C

寄存器 15: CAN IF2 屏蔽寄存器 2（CANIF2MSK2），偏移量 0x08C

本寄存器中包含了 CANIFnMSK1 寄存器中随附的扩展掩码信息。

CAN IFn 屏蔽寄存器 2（CANIFnMSK2）

CAN0 基址: 0x4004.0000
偏移量 0x02C
类型 R/W, 复位 0x0000.FFFF



位/域	名称	类型	复位	描述
31:16	保留	RO	0x0000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
15	MXTD	R/W	1	掩码扩展标识符 值 描述 0 扩展标识符位（CANIFnARB2 寄存器的 XTD 位）不影响验收滤波。 1 扩展标识符位 XTD 也用于验收滤波。
14	MDIR	R/W	1	屏蔽报文方向 值 描述 0 报文方向位（CANIFnARB2 寄存器的 DIR 位）不影响验收滤波 1 报文方向位 DIR 也用于验收滤波
13	保留	RO	1	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
12:0	MSK	R/W	0xFF	标识符掩码 在使用 29 位的标识符时，这些位用作 ID 的 [28:16] 位。CANIFnMSK2 寄存器中的 MSK 域用作 ID 的 [15 : 0] 位。在使用 11 位的标识符时，MSK[12:2] 用作 ID 的 [10:0] 位。 值 描述 0 报文对象中的对应标识符域 (ID) 不用于验收滤波。 1 报文对象中的对应标识符域 (ID) 用于验收滤波。

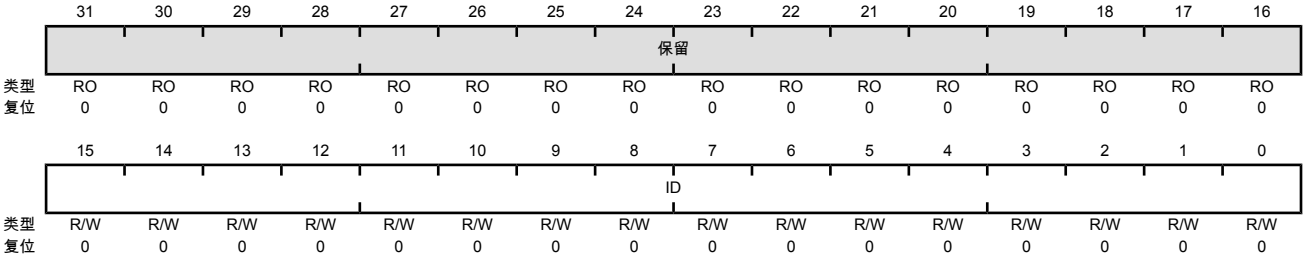
寄存器 16: CAN IF1 仲裁寄存器 1（CANIF1ARB1），偏移量 0x030

寄存器 17: CAN IF2 仲裁寄存器 1（CANIF2ARB1），偏移量 0x090

本寄存器包含用于验收过滤的报文标识符。

CAN IFn 仲裁寄存器 1（CANIFnARB1）

CAN0 基址: 0x4004.0000
偏移量 0x030
类型 R/W, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:16	保留	RO	0x0000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
15:0	ID	R/W	0x0000	报文标识符 此位域结合 CANIFnARB2 寄存器的 ID 位域，可用于创建报文标识符。 对于 29 位报文标识符，CANIFnARB1 寄存器的 [15:0] 位域用于 ID 的 [15:0]，CANIFnARB2 寄存器的 [12:0] 位域用于 ID 的 [28:16]。 对于11位报文标识符，本位域无意义。

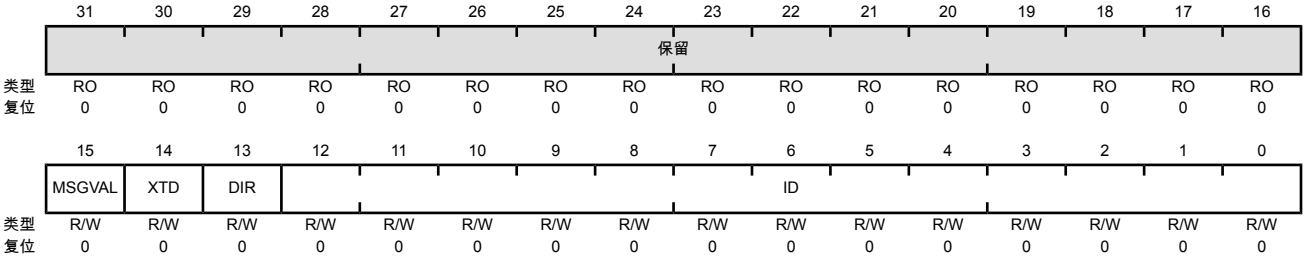
寄存器 18: CAN IF1 仲裁寄存器 2（CANIF1ARB2），偏移量 0x034

寄存器 19: CAN IF2 仲裁寄存器 2（CANIF2ARB2），偏移量 0x094

本寄存器包含用于验收过滤的相关信息。

CAN IFn 仲裁寄存器 2（CANIFnARB2）

CAN0 基址: 0x4004.0000
偏移量 0x034
类型 R/W, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:16	保留	RO	0x0000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
15	MSGVAL	R/W	0	报文有效 值 描述 0 该报文对象被报文处理器忽略 1 该报文对象已经配置完成，并准备由CAN控制器的报文处理器处理管理。 在初始化过程中以及在 CANCTL 寄存器的 INIT 位清零之前，所有未用的报文对象会将该位清零。在 CANIFnARBn 寄存器中的 ID 位、CANIFnARB2 寄存器中的 XTD 和 DIR 位或者 CANIFnMCTL 寄存器中的 DLC 位中的任一位被修改之前，或者不再需要报文对象时，也必须将 MSGVAL 位清零。
14	XTD	R/W	0	扩展标识符 值 描述 0 该报文对象采用11位标准标识符 1 该报文对象采用29位扩展标识符
13	DIR	R/W	0	报文方向 值 描述 0 接收。当 CANIFnMCTL 寄存器中的 TXRQST 位置位时，即表明接收到带有此报文对象标识符的远程帧。当接收到标识符匹配的数据帧时，报文将存储与该报文对象中。 1 发送。当 CANIFnMCTL 寄存器中的 TXRQST 位置位时，相应的报文对象将作为数据帧发送。当接收到标识符匹配的远程帧时，该报文对象的 TXRQST 位将置位（如果 RMTEN = 1）。

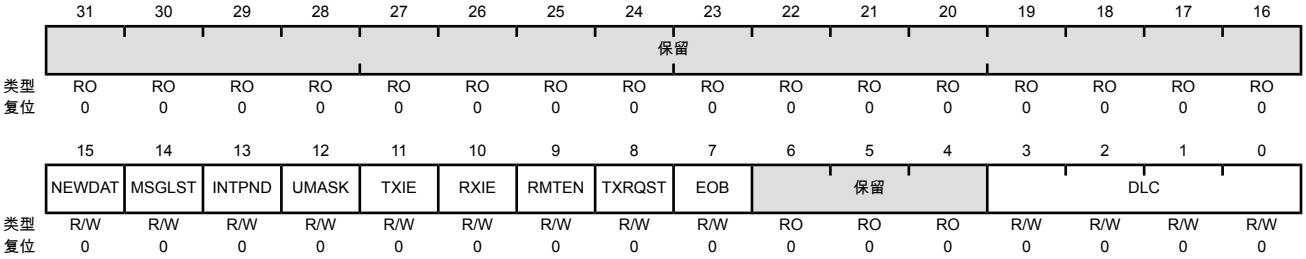
位/域	名称	类型	复位	描述
12:0	ID	R/W	0x000	报文标识符 此位域结合 CANIFnARB2 寄存器的 ID 位域，可用于创建报文标识符。 对于 29 位报文标识符，CANIFnARB1 寄存器的 ID[15:0] 位域用于 ID 的 [15:0]，ID[12:0] 位域用于 ID 的 [28:16]。 在使用 11 位的标识符时，ID[12:2] 将用作 ID 的 [10:0] 位。CANIFnARB1 寄存器的 ID 域会被忽略。

寄存器 20: CAN IF1 报文控制寄存器（CANIF1MCTL），偏移量 0x038
寄存器 21: CAN IF2 报文控制寄存器（CANIF2MCTL），偏移量 0x098

本寄存器包含将要发送给报文RAM的报文对象的控制信息。

CAN IFn 报文控制寄存器（CANIFnMCTL）

CAN0 基址: 0x4004.0000
偏移量 0x038
类型 R/W, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:16	保留	RO	0x0000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
15	NEWDAT	R/W	0	新数据 值 描述 0 此标志位上一次由CPU清零之后，报文控制器并未向此报文对象的数据部分写入新的数据。 1 报文处理器或CPU已经向此报文对象的数据部分写入了新的数据。
14	MSGLST	R/W	0	报文丢失 值 描述 0 此标志位上一次由CPU清零之后，并未丢失新的报文。 1 NEWDAT 位置位时，报文处理器在此对象中又写入了新的报文，导致 CPU 丢失报文。 仅当 CANIFnARB2 寄存器的 DIR 位清零（接收方向）时，此标志位才对报文对象有效。
13	INTPND	R/W	0	中断挂起 值 描述 0 此报文对象不是中断源 1 此报文对象是中断源。如果此时不存在优先级更高的中断，那么 CANINT 寄存器中的中断标识符将指向该报文对象。

位/域	名称	类型	复位	描述
12	UMASK	R/W	0	使用验收掩码 值 描述 0 掩码被忽略 1 使用掩码位（CANIFnMSKn 寄存器的 MSK、MXTD 和 MDIR 位）进行验收滤波。
11	TXIE	R/W	0	传输中断启用 值 描述 0 成功发送一帧后，CANIFnMCTL 寄存器的 INTPND 位不会变化 1 成功发送一帧后，CANIFnMCTL 寄存器的 INTPND 位将自动置位。
10	RXIE	R/W	0	接收中断启用 值 描述 0 成功接收一帧后，CANIFnMCTL 寄存器的 INTPND 位不会变化。 1 成功接收一帧后，CANIFnMCTL 寄存器的 INTPND 位将自动置位。
9	RMTEN	R/W	0	远程启用 值 描述 0 收到远程帧后，CANIFnMCTL 寄存器的 TXRQST 位不会变化。 1 收到远程帧后，CANIFnMCTL 寄存器的 TXRQST 位将自动置位。
8	TXRQST	R/W	0	发送请求 值 描述 0 此报文对象并未等待发送 1 已经请求发送此报文对象，并且尚未发送完成 注意: 若 CANIFnCMSK 寄存器的 WRNRD 和 TXRQST 位置位，则此标志位将被忽略。
7	EOB	R/W	0	缓冲区末端 值 描述 0 此报文对象属于FIFO缓冲区，但不是FIFO的最后一条报文对象 1 此报文对象是FIFO缓冲区的最后一个报文对象，或只是单个缓冲。 该位用于串联两个或多个报文对象（多达32个）以创建FIFO缓冲区。当为单个报文对象（因而不属于 FIFO 缓冲区）时，该位必须被置位。

位/域	名称	类型	复位	描述
6:4	保留	RO	0x0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
3:0	DLC	R/W	0x0	数据长度码

值	描述
0x0-0x8	指定数据帧中的字节数
0x9-0xF	默认数据帧长度为 8 字节

报文对象的 CANIFnMCTL 寄存器中的 DLC 域的定义必须和含相同标识符的其它节点上的相应对象相同。在报文处理器存储数据帧时，它将 DLC 写入由接收报文给定的值中。

寄存器 22: CAN IF1 数据寄存器 A1（CANIF1DA1），偏移量 0x03C
寄存器 23: CAN IF1 数据寄存器 A2（CANIF1DA2），偏移量 0x040
寄存器 24: CAN IF1 数据寄存器 B1（CANIF1DB1），偏移量 0x044
寄存器 25: CAN IF1 数据寄存器 B2（CANIF1DB2），偏移量 0x048
寄存器 26: CAN IF2 数据寄存器 A1（CANIF2DA1），偏移量 0x09C
寄存器 27: CAN IF2 数据寄存器 A2（CANIF2DA2），偏移量 0x0A0
寄存器 28: CAN IF2 数据寄存器 B1（CANIF2DB1），偏移量 0x0A4
寄存器 29: CAN IF2 数据寄存器 B2（CANIF2DB2），偏移量 0x0A8

这些寄存器包含将要发送或刚刚收到的数据。在CAN数据帧中，数据字节0是发送或接收的第一个字节，数据字节7是发送或接收的最后一个字节。在CAN的串行比特流中，每个数据字节都是高位在前的。

CAN IFn 数据寄存器 nn（CANIFnDnn）

CAN0 基址: 0x4004.0000
偏移量 0x03C
类型 R/W, 复位 0x0000.0000

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	保留															
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	DATA															
类型	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位/域	名称	类型	复位	描述
31:16	保留	RO	0x0000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
15:0	DATA	R/W	0x0000	数据 CANIFnDA1 寄存器包含数据字节 1 和 0；CANIFnDA2 包含数据字节 3 和 2；CANIFnDB1 包含数据字节 5 和 4；以及 CANIFnDB2 包含数据字节 7 和 6。

寄存器 30: CAN 传输请求寄存器 1（CANTXRQ1），偏移量 0x100

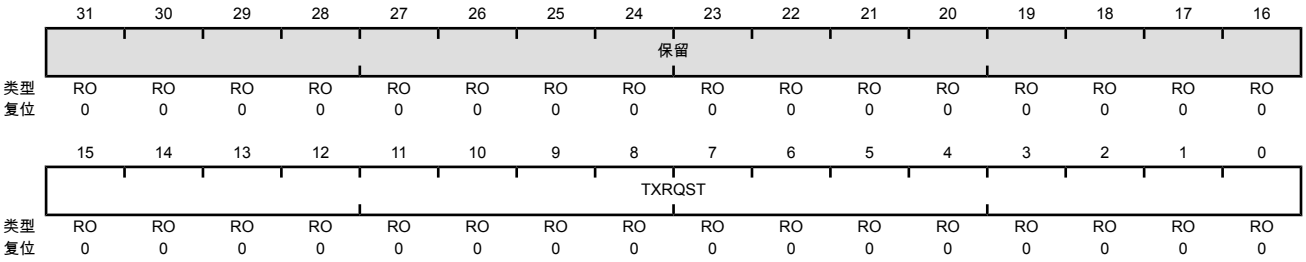
寄存器 31: CAN 传输请求寄存器 2（CANTXRQ2），偏移量 0x104

CANTXRQ1 和 CANTXRQ2 寄存器共同保存 32 个报文对象的 TXRQST 位状态。通过读取这些标志位，CPU可以检查哪些报文对象有挂起的发送请求。某一特定报文对象的 TXRQST 位可以通过以下三种源变更：(1) CPU（通过 CANIFnMCTL 寄存器修改）；(2) 报文处理器状态机（收到远程帧后）；(3) 报文处理器状态机（在成功发送报文后）。

CANTXRQ1 寄存器包含报文 RAM 中前 16 个报文对象的 TXRQST 位；CANTXRQ2 寄存器包含报文 RAM 中后 16 个报文对象的 TXRQST 位。

CAN 传输请求寄存器 n（CANTXRQn）

CAN0 基址: 0x4004.0000
偏移量 0x100
类型 RO, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:16	保留	RO	0x0000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
15:0	TXRQST	RO	0x0000	传输请求位

值	描述
0	相应的报文对象并未等待发送
1	相应的报文对象已请求发送，并且尚未发送完成

寄存器 32: CAN 新数据寄存器 1（CANNWDA1），偏移量 0x120

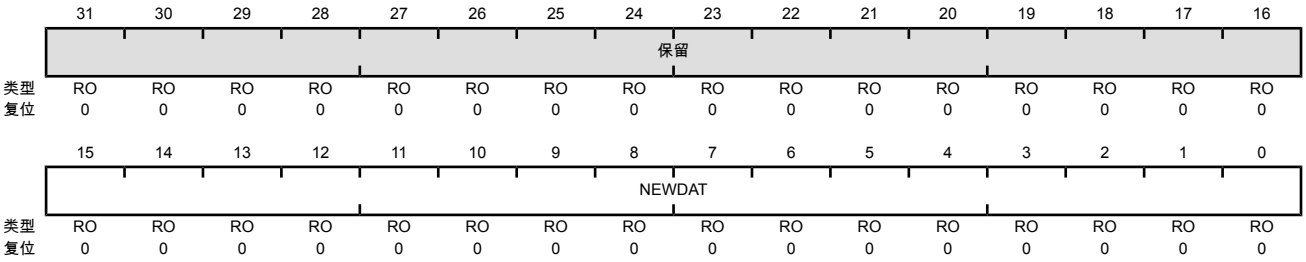
寄存器 33: CAN 新数据寄存器 2（CANNWDA2），偏移量 0x124

CANNWDA1 和 CANNWDA2 寄存器共同保存 32 个报文对象的 NEWDAT 位状态。通过读取这些标志位，CPU可以检查哪些报文对象的数据部分有更新。某一特定报文对象的 NEWDAT 位可以通过以下三种源变更：(1) CPU（通过 CANIFnMCTL 寄存器修改）；(2) 报文处理器状态机（在收到数据帧后）；(3) 报文处理器状态机（在成功发送报文后）。

CANNWDA1 寄存器包含报文 RAM 中前 16 个报文对象的 NEWDAT 位；CANNWDA2 寄存器包含报文 RAM 中后 16 个报文对象的 NEWDAT 位。

CAN 新数据寄存器 n（CANNWDAn）

CAN0 基址: 0x4004.0000
偏移量 0x120
类型 RO, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:16	保留	RO	0x0000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
15:0	NEWDAT	RO	0x0000	新数据位

值	描述
0	自从此标志位上次被CPU清零以来，相应报文对象的数据部分并未写入新的数据
1	报文处理器或CPU向相应报文对象的数据部分写入了新的数据

寄存器 34: CAN 报文 1 中断挂起寄存器（CANMSG1INT），偏移量 0x140

寄存器 35: CAN 报文 2 中断挂起寄存器（CANMSG2INT），偏移量 0x144

CANMSG1INT 和 CANMSG2INT 寄存器共同保存 32 个报文对象的 INTPND 位状态。通过读取这些标志位，CPU可以检查哪些报文对象有挂起的中断请求。某一特定报文对象的 INTPND 位可以通过以下 2 种方式改变：(1) CPU（通过 CANIFnMCTL 寄存器修改）；(2) 报文处理器状态机（在成功接收或发送一帧后）。

该域也可以通过 CANINT 寄存器中编码。

CANMSG1INT 寄存器包含报文 RAM 中前 16 个报文对象的 INTPND 位；CANMSG2INT 寄存器包含报文 RAM 中后 16 个报文对象的 INTPND 位。

CAN 报文 n 中断挂起寄存器（CANMSGnINT）

CAN0 基址: 0x4004.0000
偏移量 0x140
类型 RO, 复位 0x0000.0000

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	保留															
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	INTPND															
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位/域	名称	类型	复位	描述
31:16	保留	RO	0x0000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
15:0	INTPND	RO	0x0000	中断挂起位

值	描述
0	相应报文对象并非中断源
1	相应报文对象是当前挂起的中断源之一

寄存器 36: CAN 报文 1 有效寄存器（CANMSG1VAL），偏移量 0x160

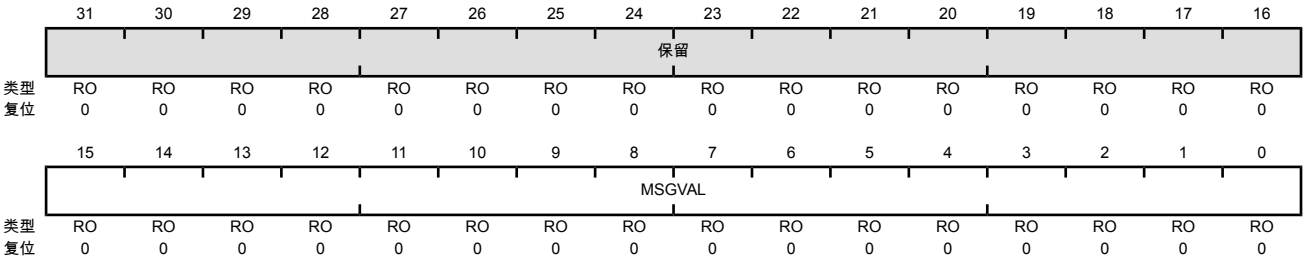
寄存器 37: CAN 报文 2 有效寄存器（CANMSG2VAL），偏移量 0x164

CANMSG1VAL 和 CANMSG2VAL 寄存器共同保存 32 个报文对象的 MSGVAL 位状态。通过读取这些标志位，CPU可以检查哪些报文对象有效。特定报文对象的报文有效位可通过 CANIFnARB2 寄存器修改。

CANMSG1VAL 寄存器包含报文 RAM 中前 16 个报文对象的 MSGVAL 位；CANMSG2VAL 寄存器包含报文 RAM 中后 16 个报文对象的 MSGVAL 位。

CAN 报文 n 有效寄存器（CANMSGnVAL）

CAN0 基址: 0x4004.0000
偏移量 0x160
类型 RO, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:16	保留	RO	0x0000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
15:0	MSGVAL	RO	0x0000	报文有效位

值	描述
0	相应报文对象并未配置，被报文处理器忽略
1	相应报文对象已完成配置，应当由报文处理器处理

18 模拟比较器

模拟比较器是一个外设，它能比较两个模拟电压的大小，并通过自身提供的逻辑输出端将比较结果以信号的形式输出。

注意: 不是所有比较器都可以选择驱动输出管脚。更多信息参见 “信号描述”（994页）。

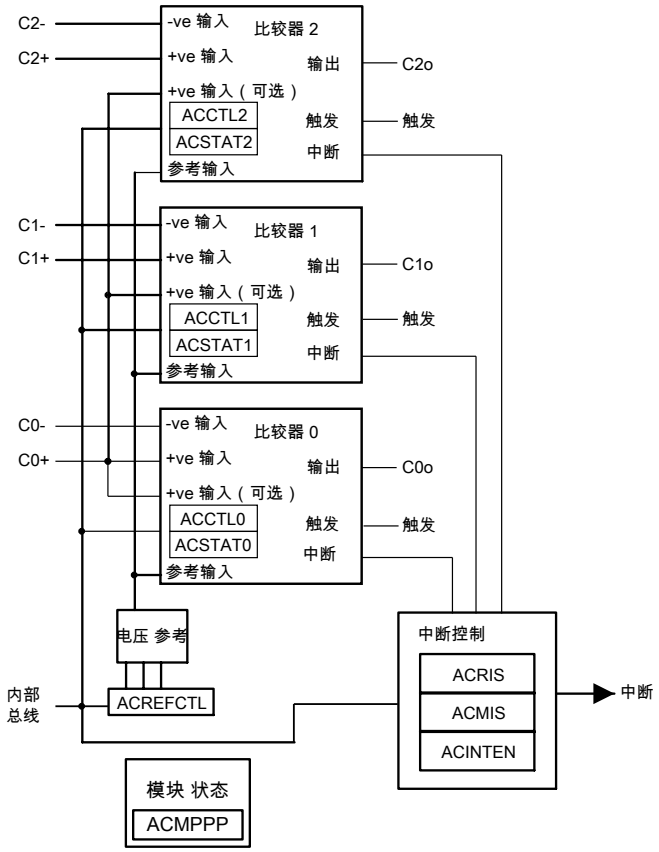
比较器可以向器件管脚提供输出，以替换板上的模拟比较器。比较器也可以通过中断信号示意应用程序在 ADC 中开始采样序列，或者直接触发采样。中断产生逻辑和ADC触发是各自独立的。这就意味着，中断可以在上升沿产生，而ADC在下降沿触发。

TM4C1231H6PM 微控制器提供两个独立集成的模拟比较器，具有如下功能：

- 可以比较外部输入管脚和外部输入管脚或内部可编程的参考电压
- 比较器可将测试电压与下面的其中一种电压相比较
  - 独立的外部参考电压
  - 共用的外部参考电压
  - 共用的内部参考电压

18.1 结构框图

图 **18-1.** 模拟比较器模块的结构图



注意: 此结构图描述了此微控制器产品系列的模拟比较器以及比较器输出的最大数量。本器件的数量可能有所不同。有关本器件的配置，请参阅1006页。

18.2 信号描述

下表中列出了模拟比较器的外部信号及其功能描述。模拟比较器输出信号是某些 GPIO 管脚的复用功能，复位时，它默认为 GPIO 信号。表中列标题名为‘管脚复用/分配’的列列出了可以被用作模拟比较器的管脚。通过将 GPIO 备用功能选择 (GPIOAFSEL) 寄存器 (595页) 中的 AFSEL 位置位来选择模拟比较器功能。括号中的数字是必须写入 GPIO 端口控制 (GPIOPCTL) 寄存器 (612页) 中的 PMCn 位的编码，以便将管脚配置为模拟比较器功能。通过将 GPIO 数字使能 (GPIODEN) 寄存器中的 DEN 位清零即可配置正极和负极的输入信号。有关如何配置 GPIO 的更多信息，请参阅“通用输入/输入端口（GPIOs）”（575页）。

表 **18-1.** 模拟比较器触发 信号 **(64LQFP)**

管脚名称	管脚编号	管脚复用/管脚赋值	管脚类型	缓冲区类型[a]	描述
C0+	14	PC6	I	模拟	模拟比较器0正极输入
C0-	13	PC7	I	模拟	模拟比较器0负极输入
C0o	28	PF0 (9)	O	TTL	模拟比较器0输出端。
C1+	15	PC5	I	模拟	模拟比较器1正极输入

表 **18-1.** 模拟比较器触发 信号 **(64LQFP)**（续）

管脚名称	管脚编号	管脚复用/管脚赋值	管脚类型	缓冲区类型[a]	描述
C1-	16	PC4	I	模拟	模拟比较器1负极输入
C1o	29	PF1 (9)	O	TTL	模拟比较器1输出端。

a. TTL 表示管脚的电压水平与 TTL 一致。

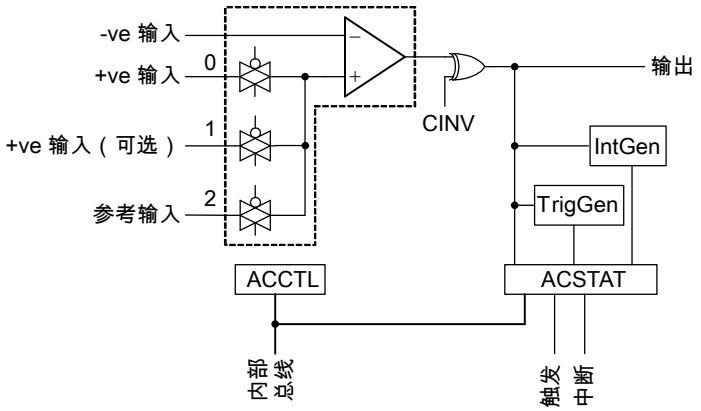
18.3 功能说明

比较器通过比较VIN-和VIN+输入来产生输出VOUT。

```
VIN- < VIN+, VOUT = 1
VIN- > VIN+, VOUT = 0
```

如图18-2（995页）所示，VIN- 的输入源是外部输入 Cn-，其中 n 指模拟比较器编号。除了外部输入 Cn+ 之外，VIN+ 输入源还可以是 C0+ 或者是内部参考源 V_{IREF}。

图 **18-2.** 比较单元的结构



比较器通过模拟比较器控制 (ACCTL) 和模拟比较器状态 (ACSTAT) 这两个状态/控制寄存器来配置。内部参考源通过模拟比较器参考电压控制 (ACREFCTL) 寄存器进行配置。中断状态和控制通过模拟比较器屏蔽中断状态 (ACMIS)、模拟比较器原始中断状态 (ACRIS) 和模拟比较器中断启用 (ACINTEN) 这三个寄存器来控制。

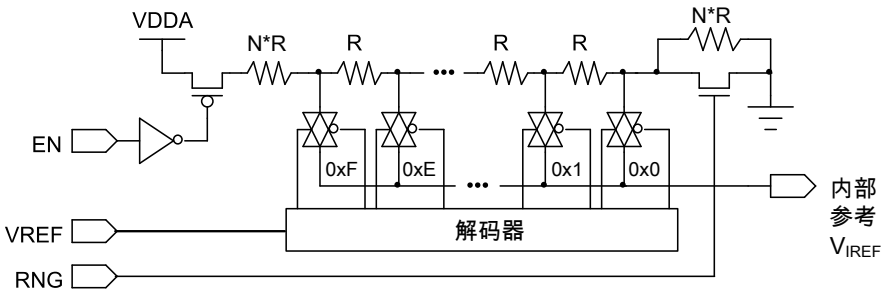
通常情况下，比较器输出被内部用于产生中断，这点受 ACCTL 寄存器的 ISEN 位控制。该输出也可以用来驱动外部管脚 (Cno)，或产生模数转换器 (ADC) 触发信号。

重要: 在使用模拟比较器之前，ACCTL 寄存器的 ASRCP 位必须置位。

18.3.1 内部参考电压编程

内部的参考电压结构如 图18-3（996页）所示。该内部参考电压由配置寄存器 ACREFCTL 控制。

图 **18-3.** 比较器内部参考结构



注意: 在上图中，N*R 代表 R 值的整数倍，它产生 表18-2（996页） 中指定的结果。

内部参考电压可设置为两种模式（低电平或高电平），具体取决于 ACREFCTL 寄存器的 RNG 位。当 RNG 位被清零时，内部参考电压处于高电平模式，而当 RNG 被置位时，内部参考电压处于低电平模式。

在每种模式下，内部参考电压 V_{IREF} 具有 16 个预先设定的阈值或阶跃值。用于与外部输入电压进行比较的阈值可通过 ACREFCTL 寄存器的 VREF 域选择。

在高电平模式下，V_{IREF} 阈值电压始于理想高电平启动电压 V_{DDA}/4.2，并以理想恒电压阶跃 V_{DDA}/29.4 增加。

在低电平模式下，V_{IREF} 阈值电压始于 0 V，并以理想恒电压阶跃 V_{DDA}/22.12 增加。有关每种模式的理想的 V_{IREF} 阶跃电压以及 RNG 和 VREF 域对其有何作用，请参阅 表18-2。

表 **18-2.** 内部参考电压和 **ACREFCTL** 域值

ACREFCTL 寄存器		基于**VREF**位域的输出参考电压
EN 位的值	**RNG** 位的值	
EN=0	RNG=X	任何 VREF 值均为 0 V (GND)。建议使用 RNG = 1 和 VREF = 0 来降低参考接地的噪声。
EN=1	RNG=0	V_{IREF} 高电平：16 个电压阈值，索引为 VREF = 0x0 ..0xF 理想启动电压 (VREF=0)：V_{DDA} / 4.2 理想阶跃电压：V_{DDA}/ 29.4 理想 V_{IREF} 阈值：V_{IREF} (VREF) = V_{DDA} / 4.2 + VREF * (V_{DDA}/ 29.4)，其中 VREF = 0x0 ..0xF 最小和最大 V_{IREF} 阈值请参考 表18-3（997页）。
	RNG=1	V_{IREF} 下限：16 个电压阈值，索引为 VREF = 0x0 ..0xF 理想启动电压 (VREF=0)：0 V 理想阶跃电压：V_{DDA}/ 22.12 理想 V_{IREF} 阈值：V_{IREF} (VREF) = VREF * (V_{DDA}/ 22.12)，其中 VREF = 0x0 ..0xF 最小和最大 V_{IREF} 阈值请参考 表18-4（997页）。

请注意 表18-2 中的值是 V_{IREF} 阈值的理想值。实际上，每个阈值阶跃的这些值都将在最小值和最大值之间变动，具体取决于进程和温度。每个阶跃的最小值和最大值通过以下的公式计算：

- V_{IREF}(VREF) [最小值] = 理想 V_{IREF}(VREF) – （理想阶跃电压 – 2 mV） / 2
- V_{IREF}(VREF) [最大值] = 理想 V_{IREF}(VREF) + （理想阶跃电压 – 2 mV） / 2

高电平和低电平模式下，V_{DDA} = 3.3V 时的最小和最大 V_{IREF} 值示例请见 表18-3 和 表18-4。请注意这些示例只适用于 V_{DDA} = 3.3V；V_{DDA} 发生变化时，数值将按比例增加和减少。

表 18-3. 模拟比较器参考电压特性，V_{DDA} = 3.3V，EN= 1 且 RNG = 0

VREF 值	V_{IREF} 最小值	理想 V_{IREF}	V_{IREF} 最大值	单元
0x0	0,731	0,786	0,841	V
0x1	0,843	0,898	0,953	V
0x2	0,955	1,010	1,065	V
0x3	1,067	1,122	1,178	V
0x4	1,180	1,235	1,290	V
0x5	1,292	1,347	1,402	V
0x6	1,404	1,459	1,514	V
0x7	1,516	1,571	1,627	V
0x8	1,629	1,684	1,739	V
0x9	1,741	1,796	1,851	V
0xA	1,853	1,908	1,963	V
0xB	1,965	2,020	2,076	V
0xC	2,078	2,133	2,188	V
0xD	2,190	2,245	2,300	V
0xE	2,302	2,357	2,412	V
0xF	2,414	2,469	2,525	V

表 18-4. 模拟比较器参考电压特性，V_{DDA} = 3.3V，EN= 1 且 RNG = 1

VREF 值	V_{IREF} 最小值	理想 V_{IREF}	V_{IREF} 最大值	单元
0x0	0,000	0,000	0,074	V
0x1	0,076	0,149	0,223	V
0x2	0,225	0,298	0,372	V
0x3	0,374	0,448	0,521	V
0x4	0,523	0,597	0,670	V
0x5	0,672	0,746	0,820	V
0x6	0,822	0,895	0,969	V
0x7	0,971	1,044	1,118	V
0x8	1,120	1,193	1,267	V
0x9	1,269	1,343	1,416	V
0xA	1,418	1,492	1,565	V
0xB	1,567	1,641	1,715	V
0xC	1,717	1,790	1,864	V
0xD	1,866	1,939	2,013	V
0xE	2,015	2,089	2,162	V
0xF	2,164	2,238	2,311	V

18.4 初始化和配置

下面的例子展示了应如何配置模拟比较器才能从内部寄存器中读回其输出值。

1. 向系统控制模块中的 RCGCACMP 寄存器写入 0x0000.0001 来启用模拟比较器时钟（参见 307 页）。
2. 通过 RCGCGPIO 寄存器（参见 295页）启用相应 GPIO 模块的时钟。欲了解需要启用哪些 GPIO 端口，请参阅表20-5（1024页）。
3. 在GPIO模块中使能GPIO端口，并配置相关的引脚为输入。欲了解需要配置哪些 GPIO，请参阅表20-4（1019页）。
4. 配置寄存器 GPIOPCTL 中 PMCn 域，将模拟比较器输出信号分配给相应的管脚（参见 612页和表20-5（1024页））。
5. 向 ACREFCTL 寄存器写入 0x0000.030C，从而将内部电压参考配置为 1.65 V。
6. 向 ACCTLn 寄存器写入 0x0000.040C，将参考选为比较器的内部电压参考源，并且不将输出翻转。
7. 延迟10µs。
8. 通过读取 ACSTATn 寄存器的 OVAL 值，获得比较器的输出值。

改变比较器负输入信号 C- 的电平以观察 OVAL 值的变化。

18.5 寄存器映射

表 18-5（998页） 中列出了比较器寄存器。表中所列偏移量是寄存器地址相对于模拟比较器基址 0x4003.C000的 16 进制增量。注意，在对这些寄存器编程之前必须先启用模拟比较器的时钟（请参阅307页）。模拟比较器模块时钟启用后，必须等待至少 3 个系统时钟才可访问模拟比较器寄存器。

表 **18-5.** 模拟比较器触发 寄存器映射

偏移量	名称	类型	复位	描述	见页面
0x00	ACMIS	R/W1C	0x0000.0000	模拟比较器屏蔽中断状态寄存器	999
0x010	ACREFCTL	R/W	0x0000.0000	模拟比较器参考电压控制寄存器	1002
0x020	ACSTAT0	RO	0x0000.0000	模拟比较器状态寄存器 0	1003
0x024	ACCTL0	R/W	0x0000.0000	模拟比较器控制寄存器 0	1004
0x04	ACRIS	RO	0x0000.0000	模拟比较器原始中断状态寄存器	1000
0x040	ACSTAT1	RO	0x0000.0000	模拟比较器状态寄存器 1	1003
0x044	ACCTL1	R/W	0x0000.0000	模拟比较器控制寄存器 1	1004
0x08	ACINTEN	R/W	0x0000.0000	模拟比较器中断启用寄存器	1001
0xFC0	ACMPPP	RO	0x0003.0003	模拟比较器外设属性寄存器	1006

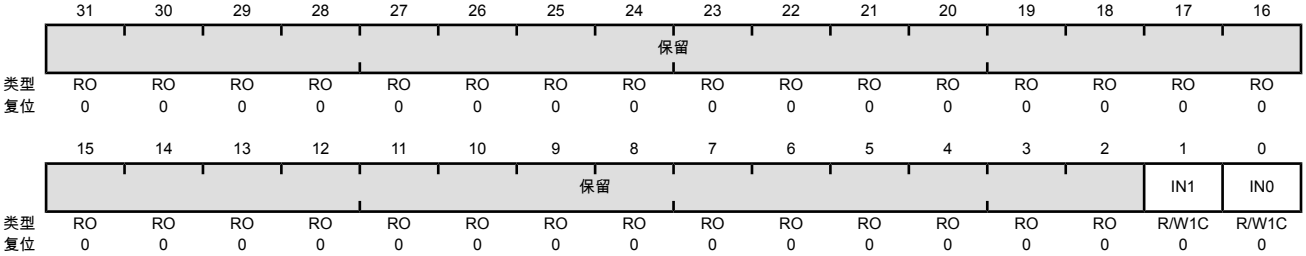
18.6 寄存器描述

本节内容将按地址偏移量的数字顺序来列举并且描述模拟比较器寄存器。

寄存器 1: 模拟比较器屏蔽中断状态寄存器（ACMIS），偏移量 0x00

该寄存器汇总了比较器的中断状态（被屏蔽的）。

模拟比较器屏蔽中断状态寄存器 (ACMIS)
基址 0x4003.C000
偏移量 0x00
类型 R/W1C, 复位 0x0000.0000



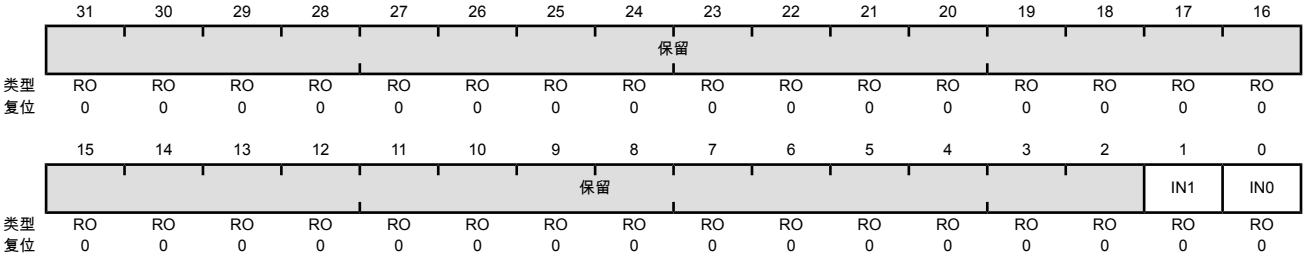
位/域	名称	类型	复位	描述
31:2	保留	RO	0x0000.000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
1	IN1	R/W1C	0	比较器1屏蔽后的中断状态 值 描述 0 未发生中断，或中断被屏蔽。 1 ACRIS 和 ACINTEN 寄存器的 IN1 位置位，向中断控制器发送中断信号。 对此标志位写 1，即可将其清零。设置此位会清除寄存器 ACRIS 中的 IN1 位。
0	IN0	R/W1C	0	比较器0屏蔽后的中断状态 值 描述 0 未发生中断，或中断被屏蔽。 1 ACRIS 和 ACINTEN 的 IN0 位被置位，向中断控制器提交了中断。 对此标志位写 1，即可将其清零。设置此位会清除寄存器 ACRIS中的 IN0 位。

寄存器 2: 模拟比较器原始中断状态寄存器（ACRIS），偏移量 0x04

该寄存器汇总了比较器的（原始）中断状态。必须启用该寄存器中的这些位，才可以使用 ACINTEN 寄存器来产生中断。

模拟比较器原始中断状态寄存器 (ACRIS)

基址 0x4003.C000
偏移量 0x04
类型 RO, 复位 0x0000.0000



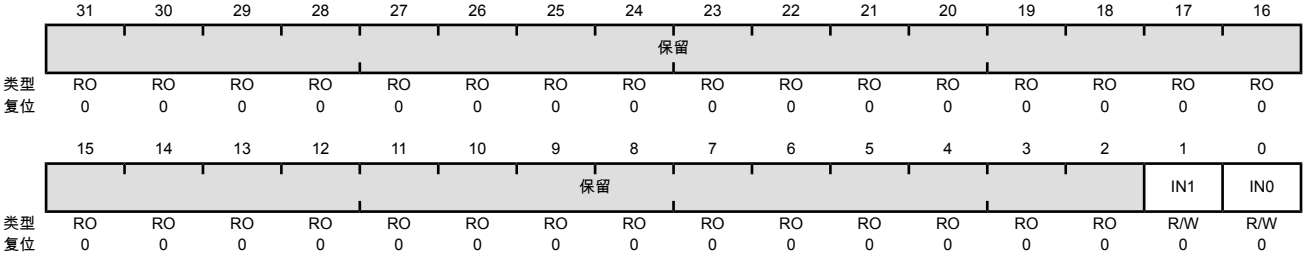
位/域	名称	类型	复位	描述
31:2	保留	RO	0x0000.000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
1	IN1	RO	0	比较器1中断状态 值 描述 0 未产生中断 1 模拟比较器 1 产生了一个通过 ACCTL1 的 ISEN 位来配置的中断。 向 ACMIS 寄存器中 IN1 位写入 1 可将此位清零。
0	IN0	RO	0	比较器0中断状态 值 描述 0 未产生中断 1 模拟比较器 0 产生了一个通过 ACCTL0 的 ISEN 位来配置的中断。 向 ACMIS 寄存器中 IN0 位写入 1 可将此位清零。

寄存器 3: 模拟比较器中断启用寄存器（ACINTEN），偏移量 0x08

该寄存器为比较器启用中断。

模拟比较器中断启用寄存器 (ACINTEN)

基址 0x4003.C000
偏移量 0x08
类型 R/W, 复位 0x0000.0000



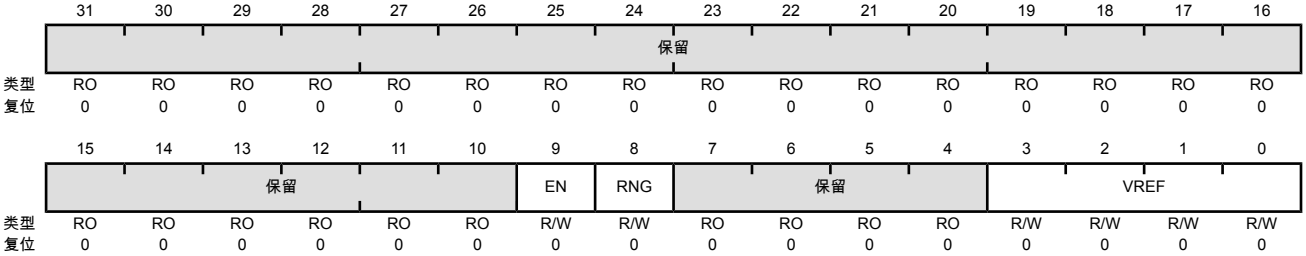
位/域	名称	类型	复位	描述
31:2	保留	RO	0x00	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
1	IN1	R/W	0	比较器 1 中断使能 值 描述 0 比较器 1 中断不影响中断状态。 1 模拟比较器1原始中断信号被送到中断控制器。
0	IN0	R/W	0	比较器 0 中断使能 值 描述 0 比较器 0 中断不影响中断状态。 1 模拟比较器0原始中断信号被送到中断控制器。

寄存器 4: 模拟比较器参考电压控制寄存器（ACREFCTL），偏移量 0x010

该寄存器指示阶梯电阻 (resistor ladder)是否已上电，以及该电阻的范围和接头(tap)。

模拟比较器参考电压控制寄存器 (ACREFCTL)

基址 0x4003.C000
偏移量 0x010
类型 R/W, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:10	保留	RO	0x0000.0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
9	EN	R/W	0	阶梯电阻启用 值 描述 0 阶梯电阻未上电。 1 阶梯电阻上电。阶梯电阻被连接到 V_{DDA}。 复位时，该位将清零，使得内部参考在未使用时仅消耗最小的功率。
8	RNG	R/W	0	阶梯电阻范围 值 描述 0 内部参考电阻的理想阶跃电压为 VDDA / 29.4。 1 内部参考电阻的理想阶跃电压为 VDDA / 22.12。
7:4	保留	RO	0x0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
3:0	VREF	R/W	0x0	阶梯电阻参考电压 VREF 位域指定了通过模拟复用器的阶梯电阻接头。接头位置的电压是可用于比较的内部参考电压。有关输出参考电压的一些例子请见 表 18-2（996页）。

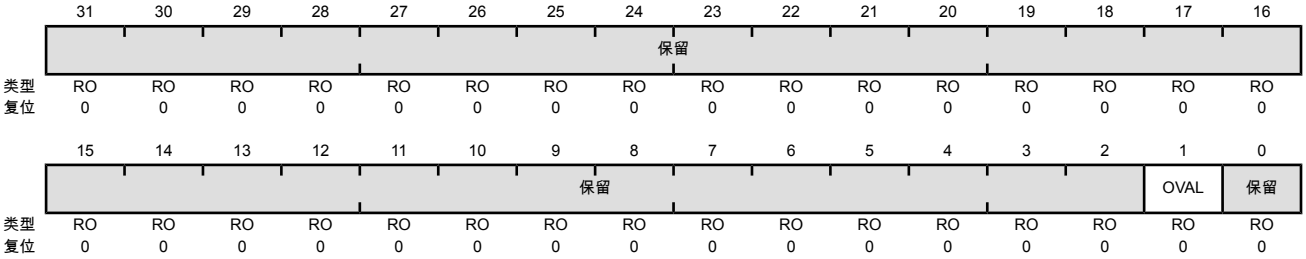
寄存器 5: 模拟比较器状态寄存器 0（ACSTAT0），偏移量 0x020

寄存器 6: 模拟比较器状态寄存器 1（ACSTAT1），偏移量 0x040

这些寄存器指示比较器的当前输出值。

模拟比较器状态寄存器 n (ACSTATn)

基址 0x4003.C000
偏移量 0x020
类型 RO, 复位 0x0000.0000



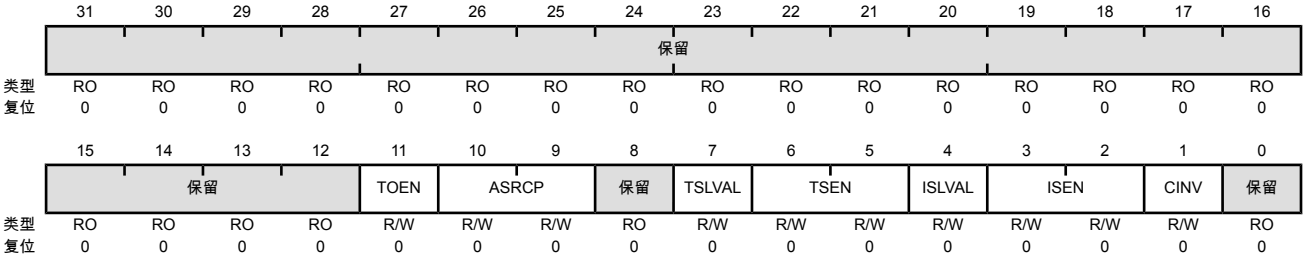
位/域	名称	类型	复位	描述
31:2	保留	RO	0x0000.000	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
1	OVAL	RO	0	比较器输出值 值 描述 0 VIN- > VIN+ 1 VIN- < VIN+ VIN - 是 Cn- 管脚上的电压。VIN+ 是 Cn+ 管脚、C0+ 管脚上的电压，或者由 ACCTL 寄存器的 ASRCP 位所定义的内部参考电压 (V_{IREF})。
0	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。

寄存器 7: 模拟比较器控制寄存器 0（ACCTL0），偏移量 0x024
寄存器 8: 模拟比较器控制寄存器 1（ACCTL1），偏移量 0x044

这些寄存器用来配置比较器的输入和输出。

模拟比较器控制寄存器 n (ACCTLn)

基址 0x4003.C000
偏移量 0x024
类型 R/W, 复位 0x0000.0000



位/域	名称	类型	复位	描述
31:12	保留	RO	0x0000.0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
11	TOEN	R/W	0	触发输出启用 值 描述 0 ADC 事件被阻止，不发送到 ADC 1 ADC 事件被发送到 ADC
10:9	ASRCP	R/W	0x0	模拟正电压源 ASRCP 位域指定了到比较器 VIN+ 端口的输入电压源。该位域的编码如下： 值 描述 0x0 以下引脚的值 Cn+ 0x1 以下引脚的值 C0+ 0x2 内部参考电压 (V_{IREF}) 0x3 保留
8	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
7	TSLVAL	R/W	0	电平触发值 值 描述 0 比较器输出低电平触发ADC事件 1 比较器输出高电平触发 ADC 事件。

位/域	名称	类型	复位	描述
6:5	TSEN	R/W	0x0	触发方式 TSEN 域指定了产生 ADC 事件的比较器输出的检测方式。检测方式如下所述： 值 描述 0x0 电平触发，见 TSLVAL 0x1 下降沿 0x2 上升沿 0x3 上升沿、下降沿均可
4	ISLVAL	R/W	0	电平触发中断值 值 描述 0 比较器输出低电平触发中断。 1 比较器输出高电平触发中断。
3:2	ISEN	R/W	0x0	中断触发方式 ISEN 位域指定了产生中断的比较器输出的检测方式。检测方式如下所述： 值 描述 0x0 电平触发，见 ISLVAL 0x1 下降沿 0x2 上升沿 0x3 上升沿、下降沿均可
1	CINV	R/W	0	比较器输出翻转 值 描述 0 比较器输出不改变 1 比较器输出之前由硬件进行了翻转。
0	保留	RO	0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。

寄存器 9: 模拟比较器外设属性寄存器（ACMPPP），偏移量 0xFC0

ACMPPP 寄存器提供关于模拟比较器模块属性的信息。

模拟比较器外设属性寄存器 (ACMPPP)

基址 0x4003.C000
偏移量 0xFC0
类型 RO, 复位 0x0003.0003

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	保留														C1O	C0O
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	保留														CMP1	CMP0
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1

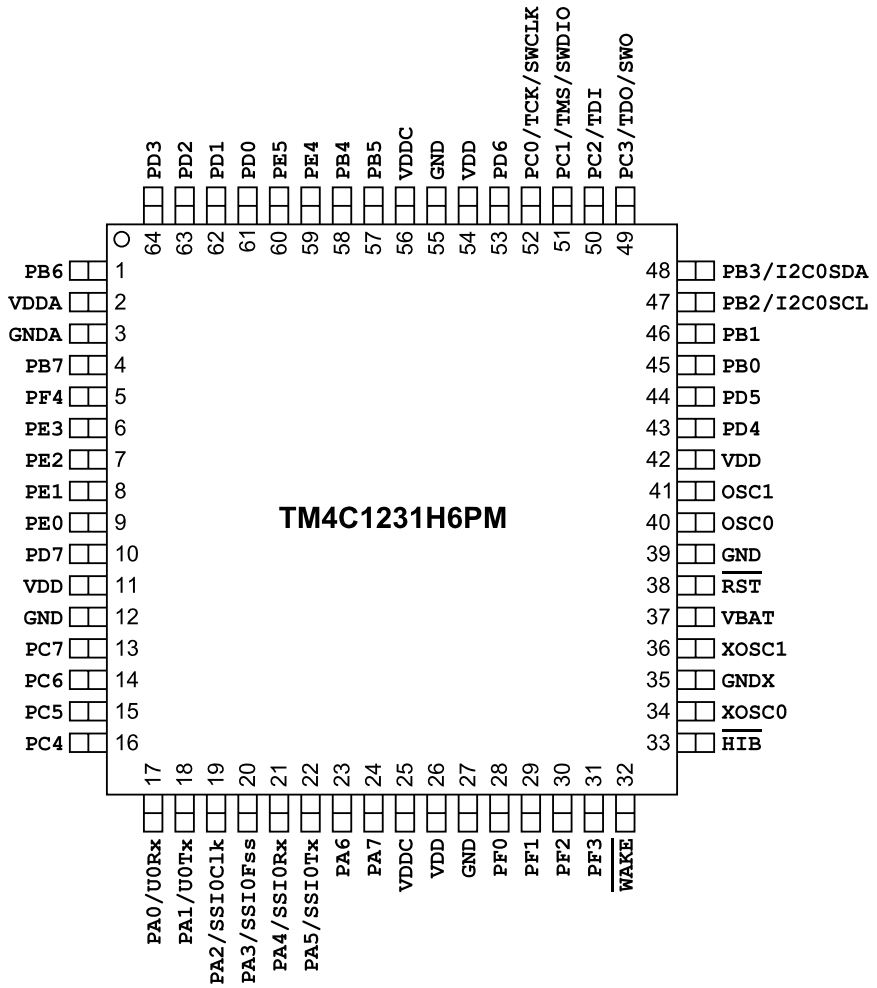
位/域	名称	类型	复位	描述
31:18	保留	RO	0x0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
17	C1O	RO	0x1	存在比较器输出 1 值 描述 0 不存在比较器输出 1。 1 存在比较器输出 1。
16	C0O	RO	0x1	存在比较器输出 0 值 描述 0 不存在比较器输出 0。 1 存在比较器输出 0。
15:2	保留	RO	0x0	软件不应该依赖保留位的值。为了兼容未来的器件，保留位的值在读 - 修改 - 写操作过程中应当保持不变。
1	CMP1	RO	0x1	存在比较器 1 值 描述 0 不存在比较器 1。 1 存在比较器 1。
0	CMP0	RO	0x1	存在比较器 0 值 描述 0 不存在比较器 0。 1 存在比较器 0。

19 管脚图

TM4C1231H6PM 微控制器的管脚图如下所示。

所有 GPIO 信号均通过 GPIO 端口识别，复位时默认为复用功能的除外。在这种情况下，默认的复用功能将标明在GPIO端口名称后面。有关每个管脚可能实现功能的完全列表请参考表20-5（1024页）。

图 **19-1. 64** 管脚 **LQFP** 封装管脚图



20 信号表

本章通过表格列举出每个管脚可配置成的信号。除下表列出的管脚外，其余可配置管脚在复位时的默认配置均为GPIO信号。通过 GPIOAMSEL 寄存器（见 611页）可选用管脚的模拟功能。当需要使用某个 GPIO 管脚的复用数字功能时，应将 GPIOAFSEL 寄存器（见 595页）的相应位置位。更多的管脚复用选项可通过 GPIOPCTL 寄存器（见 612页）的 PMCx 位域实现，通过该位域，可为 GPIO 选定某个可用的外设功能。

重要: 除下表列出的管脚外，其余可配置管脚在复位时的默认配置均为 GPIO 信号。上电复位 (POR) 时，或确认 RST 时，管脚都将恢复其默认设置。

表 **20-1.** 默认为复用功能的 **GPIO** 管脚

GPIO管脚	默认值	GPIOAFSEL位	GPIOPCTL的PMCx位域
PA[1:0]	UART0	0	0x1
PA[5:2]	SSI0	0	0x1
PB[3:2]	I^2C0	0	0x1
PA[3:0]	JTAG/SWD	1	0x3

表20-2（1008页）显示管脚到信号名称的映射，包括信号的功能特性。在每个管脚之后均完整列出其所有可供备选的模拟/数字功能。

表20-3（1014页）按信号名称的字母顺序列出信号。假如某个信号在多个管脚上均可配置，则列出该信号可对应的所有管脚。“复用管脚/赋值”一列是信号所对应的 GPIO 管脚以及应写入 GPIOPCTL 寄存器 PMCx 位域的编码。

表20-4（1019页）按功能将信号分组，GPIO 管脚除外。假如某个信号在多个管脚上均可配置，则列出该信号可对应的所有管脚。

表20-5（1024页）列出了所有 GPIO 管脚及其模拟/数字复用功能。AINx模拟信号都无法耐受 5V 电压，需经过隔离电路才能连接其内部电路。这些信号的配置方式为：将 GPIO 数字使能 (GPIODEN) 寄存器中相应的 DEN 位清零，并将 GPIO 模拟模式选择 (GPIOAMSEL) 寄存器中相应的 AMSEL 位置位。其他模拟信号都可耐受 5V，因此管脚可以直接连接内部电路（C0-、C0+、C1-、C1+）。这些信号的配置方式为：清除寄存器 GPIO 数字使能 (GPIODEN) 寄存器中相应的 DEN 位。而在配置数字信号时，应将 GPIO 备选功能选择寄存器 (GPIOAFSEL) 以及 GPIODEN 寄存器的相应位置位，并按照下表中所列出的数字编码对 GPIO 端口控制寄存器 (GPIOPCTL) 的 PMCx 位域进行适当配置。表中的灰色单元格代表相应GPIO管脚的默认值。

表20-6（1026页）按可用于配置的管脚数量升序列出了所有信号。此表格宜用于根据特定功能来规划管脚的配置。应用笔记《AN01274： Tiva™ C 系列 如何配置微控制器的管脚复用功能》提供了实施管脚复用的概述，解释系统设计人员如何定义管脚配置，并举例说明管脚配置的步骤。

注意: 所有数字输入端均为施密特触发。

20.1 按管脚编号分类的信号

表 **20-2.** 按管脚编号分类的信号

管脚编号	管脚名称	管脚类型	缓冲区类型[a]	描述
1	PB6	I/O	TTL	GPIO端口B第6位。
	SSI2Rx	I	TTL	SSI 模块 2 接收信号。
	T0CCP0	I/O	TTL	16/32 位 Timer 0 捕获/比较/PWM 0。

表 **20-2.** 按管脚编号分类的信号（续）

管脚编号	管脚名称	管脚类型	缓冲区类型[a]	描述
2	VDDA	-	电源	模拟电路（ADC、模拟比较器等）的电源正端。VDDA 应与 VDD 分开，尽量避免 VDD 中携带的噪声影响模拟功能。VDDA不论系统配置如何，管脚的电压必须符合 表21-5（1031页）中的规格。
3	GNDA	-	电源	模拟电路（ADC、 模拟比较器等）的接地参考。这些与 GND 分开，以期尽量避免 VDD 中携带的电噪声影响模拟功能。
4	PB7	I/O	TTL	GPIO端口B第7位。
	SSI2Tx	O	TTL	SSI 模块 2 发送信号。
	T0CCP1	I/O	TTL	16/32 位 Timer 0 捕获/比较/PWM 1。
5	PF4	I/O	TTL	GPIO端口F第4位。
	T2CCP0	I/O	TTL	16/32 位 Timer 2 捕获/比较/PWM 0。
6	PE3	I/O	TTL	GPIO端口E第3位。
	AIN0	I	模拟	模数转换器输入0。
7	PE2	I/O	TTL	GPIO端口E第2位。
	AIN1	I	模拟	模数转换器输入1。
8	PE1	I/O	TTL	GPIO端口E第1位。
	AIN2	I	模拟	模数转换器输入2。
	U7Tx	O	TTL	UART 7模块发送信号。
9	PE0	I/O	TTL	GPIO端口E第0位。
	AIN3	I	模拟	模数转换器输入3。
	U7Rx	I	TTL	UART 7模块接收信号。
10	PD7	I/O	TTL	GPIO端口D第7位。
	NMI	I	TTL	不可屏蔽的中断
	U2Tx	O	TTL	UART 2模块发送信号。
	WT5CCP1	I/O	TTL	32/64 位宽 Timer 5 捕获/比较/PWM 1。
11	VDD	-	电源	I/O和某些逻辑的电源正极。
12	GND	-	电源	逻辑和I/O管脚的地参考。
13	PC7	I/O	TTL	GPIO端口C第7位。
	C0-	I	模拟	模拟比较器0负极输入
	U3Tx	O	TTL	UART 3模块发送信号。
	WT1CCP1	I/O	TTL	32/64 位宽 Timer 1 捕获/比较/PWM 1。
14	PC6	I/O	TTL	GPIO端口C第6位。
	C0+	I	模拟	模拟比较器0正极输入
	U3Rx	I	TTL	UART 3模块接收信号。
	WT1CCP0	I/O	TTL	32/64 位宽 Timer 1 捕获/比较/PWM 0。
15	PC5	I/O	TTL	GPIO端口C第5位。
	C1+	I	模拟	模拟比较器1正极输入
	U1CTS	I	TTL	UART 模块 1 CTS（Clear to Send，允许发送）调制解调器流控输入信号。
	U1Tx	O	TTL	UART 1模块发送信号。
	U4Tx	O	TTL	UART 4模块发送信号。
	WT0CCP1	I/O	TTL	32/64 位宽 Timer 0 捕获/比较/PWM 1。

表 **20-2.** 按管脚编号分类的信号（续）

管脚编号	管脚名称	管脚类型	缓冲区类型[a]	描述
16	PC4	I/O	TTL	GPIO端口C第4位。
	C1-	I	模拟	模拟比较器1负极输入
	U1RTS	O	TTL	UART 模块 1 RTS（Request to Send，请求发送）调制解调器流控输出线。
	U1Rx	I	TTL	UART 1模块接收信号。
	U4Rx	I	TTL	UART 4模块接收信号。
	WT0CCP0	I/O	TTL	32/64 位宽 Timer 0 捕获/比较/PWM 0。
17	PA0	I/O	TTL	GPIO端口A第0位。
	U0Rx	I	TTL	UART 0模块接收信号。
18	PA1	I/O	TTL	GPIO端口A第1位。
	U0Tx	O	TTL	UART 0模块发送信号。
19	PA2	I/O	TTL	GPIO端口A第2位。
	SSI0Clk	I/O	TTL	SSI模块0时钟
20	PA3	I/O	TTL	GPIO端口A第3位。
	SSI0Fss	I/O	TTL	SSI 模块 0 帧信号
21	PA4	I/O	TTL	GPIO端口A第4位。
	SSI0Rx	I	TTL	SSI模块0接收
22	PA5	I/O	TTL	GPIO端口A第5位。
	SSI0Tx	O	TTL	SSI模块0发送
23	PA6	I/O	TTL	GPIO端口A第6位。
	I2C1SCL	I/O	OD	I^2C 模块 1 时钟。请注意该信号具有有源上拉。不应将相应的端口管脚配置为开漏。
24	PA7	I/O	TTL	GPIO端口A第7位。
	I2C1SDA	I/O	OD	I^2C 模块 1 数据。
25	VDDC	-	电源	为主要的逻辑部分（包括处理器内核以及大部分片上外设）供电的电源正端。该管脚上的电压为 1.2 V，由片上 LDO 提供。按照 表 21-12（1043页）中的规定，VDDC管脚只应相互连接或连接外部电容。
26	VDD	-	电源	I/O和某些逻辑的电源正极。
27	GND	-	电源	逻辑和I/O管脚的地参考。
28	PF0	I/O	TTL	GPIO端口F第0位。
	C0o	O	TTL	模拟比较器0输出端。
	CAN0Rx	I	TTL	CAN模块0接收信号。
	NMI	I	TTL	不可屏蔽的中断
	SSI1Rx	I	TTL	SSI 模块 1 接收信号。
	T0CCP0	I/O	TTL	16/32 位 Timer 0 捕获/比较/PWM 0。
	U1RTS	O	TTL	UART 模块 1 RTS（Request to Send，请求发送）调制解调器流控输出线。

表 **20-2.** 按管脚编号分类的信号（续）

管脚编号	管脚名称	管脚类型	缓冲区类型[a]	描述
29	PF1	I/O	TTL	GPIO端口F第1位。
	C1o	O	TTL	模拟比较器1输出端。
	SSI1Tx	O	TTL	SSI 模块 1 发送信号。
	T0CCP1	I/O	TTL	16/32 位 Timer 0 捕获/比较/PWM 1。
	TRD1	O	TTL	跟踪数据信号 1。
	U1CTS	I	TTL	UART 模块 1 CTS（Clear to Send，允许发送）调制解调器流控输入信号。
30	PF2	I/O	TTL	GPIO端口F第2位。
	SSI1Clk	I/O	TTL	SSI 模块 1 时钟信号。
	T1CCP0	I/O	TTL	16/32 位 Timer 1 捕获/比较/PWM 0。
	TRD0	O	TTL	跟踪数据信号 0。
31	PF3	I/O	TTL	GPIO端口F第3位。
	CAN0Tx	O	TTL	CAN模块0发送信号。
	SSI1Fss	I/O	TTL	SSI 模块 1 帧信号。
	T1CCP1	I/O	TTL	16/32 位 Timer 1 捕获/比较/PWM 1。
	TRCLK	O	TTL	跟踪时钟信号。
32	WAKE	I	TTL	当有效时外部输入将处理器从休眠模式中唤醒。
33	HIB	O	TTL	该输出指示处理器是否处于休眠模式。
34	XOSC0	I	模拟	休眠模块晶体振荡器输入或外部时钟参考输入。请注意休眠模块 RTC 采用 32.768-kHz 晶体或 32.768-kHz 振荡器。
35	GNDX	-	电源	休眠震荡器的接地。当使用晶振时钟源时，该管脚应连接到数字接地和晶振负载电容。使用外部震荡器时，该管脚应连接到数字接地。
36	XOSC1	O	模拟	休眠模块晶体振荡器输出。当使用外部单端参考时钟源时，此管脚应悬空。
37	VBAT	-	电源	休眠模块的电源供应源它通常连接到电池的正极端并用作备用电池/休眠模块电源供应器的电源。
38	RST	I	TTL	系统复位输入
39	GND	-	电源	逻辑和I/O管脚的地参考。
40	OSC0	I	模拟	主振荡器晶体输入或外部时钟参考输入。
41	OSC1	O	模拟	主振荡器晶体输出。当使用外部单端参考时钟源时，此管脚应悬空。
42	VDD	-	电源	I/O和某些逻辑的电源正极。
43	PD4	I/O	TTL	GPIO端口D第4位。该管脚不能承受 5V 最高电压。
	U6Rx	I	TTL	UART 6模块接收信号。
	WT4CCP0	I/O	TTL	32/64 位宽 Timer 4 捕获/比较/PWM 0。
44	PD5	I/O	TTL	GPIO端口D第5位。该管脚不能承受 5V 最高电压。
	U6Tx	O	TTL	UART 6模块发送信号。
	WT4CCP1	I/O	TTL	32/64 位宽 Timer 4 捕获/比较/PWM 1。
45	PB0	I/O	TTL	GPIO端口B第0位。该管脚不能承受 5V 最高电压。
	T2CCP0	I/O	TTL	16/32 位 Timer 2 捕获/比较/PWM 0。
	U1Rx	I	TTL	UART 1模块接收信号。

表 **20-2.** 按管脚编号分类的信号（续）

管脚编号	管脚名称	管脚类型	缓冲区类型[a]	描述
46	PB1	I/O	TTL	GPIO端口B第1位。该管脚不能承受 5V 最高电压。
	T2CCP1	I/O	TTL	16/32 位 Timer 2 捕获/比较/PWM 1。
	U1Tx	O	TTL	UART 1模块发送信号。
47	PB2	I/O	TTL	GPIO端口B第2位。
	I2C0SCL	I/O	OD	I^2C 模块 0 时钟。请注意该信号具有有源上拉。不应将相应的端口管脚配置为开漏。
	T3CCP0	I/O	TTL	16/32 位 Timer 3 捕获/比较/PWM 0。
48	PB3	I/O	TTL	GPIO端口B第3位。
	I2C0SDA	I/O	OD	I^2C 模块 0 数据。
	T3CCP1	I/O	TTL	16/32 位 Timer 3 捕获/比较/PWM 1。
49	PC3	I/O	TTL	GPIO端口C第3位。
	SWO	O	TTL	JTAG TDO及SWO信号。
	T5CCP1	I/O	TTL	16/32 位 Timer 5 捕获/比较/PWM 1。
	TDO	O	TTL	JTAG TDO及SWO信号。
50	PC2	I/O	TTL	GPIO端口C第2位。
	T5CCP0	I/O	TTL	16/32 位 Timer 5 捕获/比较/PWM 0。
	TDI	I	TTL	JTAG TDI信号。
51	PC1	I/O	TTL	GPIO端口C第1位。
	SWDIO	I/O	TTL	JTAG TMS及SWDIO信号。
	T4CCP1	I/O	TTL	16/32 位 Timer 4 捕获/比较/PWM 1。
	TMS	I	TTL	JTAG TMS及SWDIO信号。
52	PC0	I/O	TTL	GPIO端口C第0位。
	SWCLK	I	TTL	JTAG/SWD CLK信号。
	T4CCP0	I/O	TTL	16/32 位 Timer 4 捕获/比较/PWM 0。
	TCK	I	TTL	JTAG/SWD CLK信号。
53	PD6	I/O	TTL	GPIO端口D第6位。
	U2Rx	I	TTL	UART 2模块接收信号。
	WT5CCP0	I/O	TTL	32/64 位宽 Timer 5 捕获/比较/PWM 0。
54	VDD	-	电源	I/O和某些逻辑的电源正极。
55	GND	-	电源	逻辑和I/O管脚的地参考。
56	VDDC	-	电源	为主要的逻辑部分（包括处理器内核以及大部分片上外设）供电的电源正端。该管脚上的电压为 1.2 V，由片上 LDO 提供。按照 表 21-12（1043页）中的规定，VDDC管脚只应相互连接或连接外部电容。
57	PB5	I/O	TTL	GPIO端口B第5位。
	AIN11	I	模拟	模数转换器输入11。
	CAN0Tx	O	TTL	CAN模块0发送信号。
	SSI2Fss	I/O	TTL	SSI 模块 2 帧信号。
	T1CCP1	I/O	TTL	16/32 位 Timer 1 捕获/比较/PWM 1。

表 **20-2.** 按管脚编号分类的信号（续）

管脚编号	管脚名称	管脚类型	缓冲区类型[a]	描述
58	PB4	I/O	TTL	GPIO端口B第4位。
	AIN10	I	模拟	模数转换器输入10。
	CAN0Rx	I	TTL	CAN模块0接收信号。
	SSI2Clk	I/O	TTL	SSI 模块 2 时钟信号。
	T1CCP0	I/O	TTL	16/32 位 Timer 1 捕获/比较/PWM 0。
59	PE4	I/O	TTL	GPIO端口E第4位。
	AIN9	I	模拟	模数转换器输入9。
	CAN0Rx	I	TTL	CAN模块0接收信号。
	I2C2SCL	I/O	OD	I^2C 模块 2 时钟。请注意该信号具有有源上拉。不应将相应的端口管脚配置为开漏。
	U5Rx	I	TTL	UART 5模块接收信号。
60	PE5	I/O	TTL	GPIO端口E第5位。
	AIN8	I	模拟	模数转换器输入8。
	CAN0Tx	O	TTL	CAN模块0发送信号。
	I2C2SDA	I/O	OD	I^2C 模块 2 数据。
	U5Tx	O	TTL	UART 5模块发送信号。
61	PD0	I/O	TTL	GPIO端口D第0位。
	AIN7	I	模拟	模数转换器输入7。
	I2C3SCL	I/O	OD	I^2C 模块 3 时钟。请注意该信号具有有源上拉。不应将相应的端口管脚配置为开漏。
	SSI1Clk	I/O	TTL	SSI 模块 1 时钟信号。
	SSI3Clk	I/O	TTL	SSI 模块 3 时钟信号。
	WT2CCP0	I/O	TTL	32/64 位宽 Timer 2 捕获/比较/PWM 0。
62	PD1	I/O	TTL	GPIO端口D第1位。
	AIN6	I	模拟	模数转换器输入6。
	I2C3SDA	I/O	OD	I^2C 模块 3 数据。
	SSI1Fss	I/O	TTL	SSI 模块 1 帧信号。
	SSI3Fss	I/O	TTL	SSI 模块 3 帧信号。
	WT2CCP1	I/O	TTL	32/64 位宽 Timer 2 捕获/比较/PWM 1。
63	PD2	I/O	TTL	GPIO端口D第2位。
	AIN5	I	模拟	模数转换器输入5。
	SSI1Rx	I	TTL	SSI 模块 1 接收信号。
	SSI3Rx	I	TTL	SSI 模块 3 接收信号。
	WT3CCP0	I/O	TTL	32/64 位宽 Timer 3 捕获/比较/PWM 0。
64	PD3	I/O	TTL	GPIO端口D第3位。
	AIN4	I	模拟	模数转换器输入4。
	SSI1Tx	O	TTL	SSI 模块 1 发送信号。
	SSI3Tx	O	TTL	SSI 模块 3 发送信号。
	WT3CCP1	I/O	TTL	32/64 位宽 Timer 3 捕获/比较/PWM 1。

a. TTL 表示管脚的电压水平与 TTL 一致。

20.2 按信号名称分类的信号

表 **20-3.** 按信号名称分类的信号

管脚名称	管脚编号	管脚复用/管脚赋值	管脚类型	缓冲区类型[a]	描述
AIN0	6	PE3	I	模拟	模数转换器输入0。
AIN1	7	PE2	I	模拟	模数转换器输入1。
AIN2	8	PE1	I	模拟	模数转换器输入2。
AIN3	9	PE0	I	模拟	模数转换器输入3。
AIN4	64	PD3	I	模拟	模数转换器输入4。
AIN5	63	PD2	I	模拟	模数转换器输入5。
AIN6	62	PD1	I	模拟	模数转换器输入6。
AIN7	61	PD0	I	模拟	模数转换器输入7。
AIN8	60	PE5	I	模拟	模数转换器输入8。
AIN9	59	PE4	I	模拟	模数转换器输入9。
AIN10	58	PB4	I	模拟	模数转换器输入10。
AIN11	57	PB5	I	模拟	模数转换器输入11。
C0+	14	PC6	I	模拟	模拟比较器0正极输入
C0-	13	PC7	I	模拟	模拟比较器0负极输入
C0o	28	PF0 (9)	O	TTL	模拟比较器0输出端。
C1+	15	PC5	I	模拟	模拟比较器1正极输入
C1-	16	PC4	I	模拟	模拟比较器1负极输入
C1o	29	PF1 (9)	O	TTL	模拟比较器1输出端。
CAN0Rx	28 58 59	PF0 (3) PB4 (8) PE4 (8)	I	TTL	CAN模块0接收信号。
CAN0Tx	31 57 60	PF3 (3) PB5 (8) PE5 (8)	O	TTL	CAN模块0发送信号。
GND	12 27 39 55	固定	-	电源	逻辑和I/O管脚的地参考。
GNDA	3	固定	-	电源	模拟电路（ADC、模拟比较器等）的接地参考。这些与 GND 分开，以期尽量避免 VDD 中携带的电噪声影响模拟功能。
GNDX	35	固定	-	电源	休眠震荡器的接地。当使用晶振时钟源时，该管脚应连接到数字接地和晶振负载电容。使用外部震荡器时，该管脚应连接到数字接地。
HIB	33	固定	O	TTL	该输出指示处理器是否处于休眠模式。
I2C0SCL	47	PB2 (3)	I/O	OD	I^2C 模块 0 时钟。请注意该信号具有有源上拉。不应将相应的端口管脚配置为开漏。
I2C0SDA	48	PB3 (3)	I/O	OD	I^2C 模块 0 数据。
I2C1SCL	23	PA6 (3)	I/O	OD	I^2C 模块 1 时钟。请注意该信号具有有源上拉。不应将相应的端口管脚配置为开漏。
I2C1SDA	24	PA7 (3)	I/O	OD	I^2C 模块 1 数据。
I2C2SCL	59	PE4 (3)	I/O	OD	I^2C 模块 2 时钟。请注意该信号具有有源上拉。不应将相应的端口管脚配置为开漏。

表 **20-3.** 按信号名称分类的信号（续）

管脚名称	管脚编号	管脚复用/管脚赋值	管脚类型	缓冲区类型[a]	描述
I2C2SDA	60	PE5 (3)	I/O	OD	I^2C 模块 2 数据。
I2C3SCL	61	PD0 (3)	I/O	OD	I^2C 模块 3 时钟。请注意该信号具有有源上拉。不应将相应的端口管脚配置为开漏。
I2C3SDA	62	PD1 (3)	I/O	OD	I^2C 模块 3 数据。
NMI	10 28	PD7 (8) PF0 (8)	I	TTL	不可屏蔽的中断
OSC0	40	固定	I	模拟	主振荡器晶体输入或外部时钟参考输入。
OSC1	41	固定	O	模拟	主振荡器晶体输出。当使用外部单端参考时钟源时，此管脚应悬空。
PA0	17	-	I/O	TTL	GPIO端口A第0位。
PA1	18	-	I/O	TTL	GPIO端口A第1位。
PA2	19	-	I/O	TTL	GPIO端口A第2位。
PA3	20	-	I/O	TTL	GPIO端口A第3位。
PA4	21	-	I/O	TTL	GPIO端口A第4位。
PA5	22	-	I/O	TTL	GPIO端口A第5位。
PA6	23	-	I/O	TTL	GPIO端口A第6位。
PA7	24	-	I/O	TTL	GPIO端口A第7位。
PB0	45	-	I/O	TTL	GPIO端口B第0位。该管脚不能承受5V最高电压。
PB1	46	-	I/O	TTL	GPIO端口B第1位。该管脚不能承受5V最高电压。
PB2	47	-	I/O	TTL	GPIO端口B第2位。
PB3	48	-	I/O	TTL	GPIO端口B第3位。
PB4	58	-	I/O	TTL	GPIO端口B第4位。
PB5	57	-	I/O	TTL	GPIO端口B第5位。
PB6	1	-	I/O	TTL	GPIO端口B第6位。
PB7	4	-	I/O	TTL	GPIO端口B第7位。
PC0	52	-	I/O	TTL	GPIO端口C第0位。
PC1	51	-	I/O	TTL	GPIO端口C第1位。
PC2	50	-	I/O	TTL	GPIO端口C第2位。
PC3	49	-	I/O	TTL	GPIO端口C第3位。
PC4	16	-	I/O	TTL	GPIO端口C第4位。
PC5	15	-	I/O	TTL	GPIO端口C第5位。
PC6	14	-	I/O	TTL	GPIO端口C第6位。
PC7	13	-	I/O	TTL	GPIO端口C第7位。
PD0	61	-	I/O	TTL	GPIO端口D第0位。
PD1	62	-	I/O	TTL	GPIO端口D第1位。
PD2	63	-	I/O	TTL	GPIO端口D第2位。
PD3	64	-	I/O	TTL	GPIO端口D第3位。
PD4	43	-	I/O	TTL	GPIO端口D第4位。该管脚不能承受5V最高电压。
PD5	44	-	I/O	TTL	GPIO端口D第5位。该管脚不能承受5V最高电压。
PD6	53	-	I/O	TTL	GPIO端口D第6位。
PD7	10	-	I/O	TTL	GPIO端口D第7位。
PE0	9	-	I/O	TTL	GPIO端口E第0位。

表 **20-3.** 按信号名称分类的信号（续）

管脚名称	管脚编号	管脚复用/管脚赋值	管脚类型	缓冲区类型[a]	描述
PE1	8	-	I/O	TTL	GPIO端口E第1位。
PE2	7	-	I/O	TTL	GPIO端口E第2位。
PE3	6	-	I/O	TTL	GPIO端口E第3位。
PE4	59	-	I/O	TTL	GPIO端口E第4位。
PE5	60	-	I/O	TTL	GPIO端口E第5位。
PF0	28	-	I/O	TTL	GPIO端口F第0位。
PF1	29	-	I/O	TTL	GPIO端口F第1位。
PF2	30	-	I/O	TTL	GPIO端口F第2位。
PF3	31	-	I/O	TTL	GPIO端口F第3位。
PF4	5	-	I/O	TTL	GPIO端口F第4位。
RST	38	固定	I	TTL	系统复位输入
SSI0Clk	19	PA2 (2)	I/O	TTL	SSI模块0时钟
SSI0Fss	20	PA3 (2)	I/O	TTL	SSI 模块 0 帧信号
SSI0Rx	21	PA4 (2)	I	TTL	SSI模块0接收
SSI0Tx	22	PA5 (2)	O	TTL	SSI模块0发送
SSI1Clk	30 61	PF2 (2) PD0 (2)	I/O	TTL	SSI 模块 1 时钟信号。
SSI1Fss	31 62	PF3 (2) PD1 (2)	I/O	TTL	SSI 模块 1 帧信号。
SSI1Rx	28 63	PF0 (2) PD2 (2)	I	TTL	SSI 模块 1 接收信号。
SSI1Tx	29 64	PF1 (2) PD3 (2)	O	TTL	SSI 模块 1 发送信号。
SSI2Clk	58	PB4 (2)	I/O	TTL	SSI 模块 2 时钟信号。
SSI2Fss	57	PB5 (2)	I/O	TTL	SSI 模块 2 帧信号。
SSI2Rx	1	PB6 (2)	I	TTL	SSI 模块 2 接收信号。
SSI2Tx	4	PB7 (2)	O	TTL	SSI 模块 2 发送信号。
SSI3Clk	61	PD0 (1)	I/O	TTL	SSI 模块 3 时钟信号。
SSI3Fss	62	PD1 (1)	I/O	TTL	SSI 模块 3 帧信号。
SSI3Rx	63	PD2 (1)	I	TTL	SSI 模块 3 接收信号。
SSI3Tx	64	PD3 (1)	O	TTL	SSI 模块 3 发送信号。
SWCLK	52	PC0 (1)	I	TTL	JTAG/SWD CLK信号。
SWDIO	51	PC1 (1)	I/O	TTL	JTAG TMS及SWDIO信号。
SWO	49	PC3 (1)	O	TTL	JTAG TDO及SWO信号。
T0CCP0	1 28	PB6 (7) PF0 (7)	I/O	TTL	16/32 位 Timer 0 捕获/比较/PWM 0。
T0CCP1	4 29	PB7 (7) PF1 (7)	I/O	TTL	16/32 位 Timer 0 捕获/比较/PWM 1。
T1CCP0	30 58	PF2 (7) PB4 (7)	I/O	TTL	16/32 位 Timer 1 捕获/比较/PWM 0。
T1CCP1	31 57	PF3 (7) PB5 (7)	I/O	TTL	16/32 位 Timer 1 捕获/比较/PWM 1。
T2CCP0	5 45	PF4 (7) PB0 (7)	I/O	TTL	16/32 位 Timer 2 捕获/比较/PWM 0。

表 **20-3.** 按信号名称分类的信号（续）

管脚名称	管脚编号	管脚复用/管脚赋值	管脚类型	缓冲区类型[a]	描述
T2CCP1	46	PB1 (7)	I/O	TTL	16/32 位 Timer 2 捕获/比较/PWM 1。
T3CCP0	47	PB2 (7)	I/O	TTL	16/32 位 Timer 3 捕获/比较/PWM 0。
T3CCP1	48	PB3 (7)	I/O	TTL	16/32 位 Timer 3 捕获/比较/PWM 1。
T4CCP0	52	PC0 (7)	I/O	TTL	16/32 位 Timer 4 捕获/比较/PWM 0。
T4CCP1	51	PC1 (7)	I/O	TTL	16/32 位 Timer 4 捕获/比较/PWM 1。
T5CCP0	50	PC2 (7)	I/O	TTL	16/32 位 Timer 5 捕获/比较/PWM 0。
T5CCP1	49	PC3 (7)	I/O	TTL	16/32 位 Timer 5 捕获/比较/PWM 1。
TCK	52	PC0 (1)	I	TTL	JTAG/SWD CLK信号。
TDI	50	PC2 (1)	I	TTL	JTAG TDI信号。
TDO	49	PC3 (1)	O	TTL	JTAG TDO及SWO信号。
TMS	51	PC1 (1)	I	TTL	JTAG TMS及SWDIO信号。
TRCLK	31	PF3 (14)	O	TTL	跟踪时钟信号。
TRD0	30	PF2 (14)	O	TTL	跟踪数据信号 0。
TRD1	29	PF1 (14)	O	TTL	跟踪数据信号 1。
U0Rx	17	PA0 (1)	I	TTL	UART 0模块接收信号。
U0Tx	18	PA1 (1)	O	TTL	UART 0模块发送信号。
U1CTS	15 29	PC5 (8) PF1 (1)	I	TTL	UART 模块 1 CTS（Clear to Send，允许发送）调制解调器流控输入信号。
U1RTS	16 28	PC4 (8) PF0 (1)	O	TTL	UART 模块 1 RTS（Request to Send，请求发送）调制解调器流控输出线。
U1Rx	16 45	PC4 (2) PB0 (1)	I	TTL	UART 1模块接收信号。
U1Tx	15 46	PC5 (2) PB1 (1)	O	TTL	UART 1模块发送信号。
U2Rx	53	PD6 (1)	I	TTL	UART 2模块接收信号。
U2Tx	10	PD7 (1)	O	TTL	UART 2模块发送信号。
U3Rx	14	PC6 (1)	I	TTL	UART 3模块接收信号。
U3Tx	13	PC7 (1)	O	TTL	UART 3模块发送信号。
U4Rx	16	PC4 (1)	I	TTL	UART 4模块接收信号。
U4Tx	15	PC5 (1)	O	TTL	UART 4模块发送信号。
U5Rx	59	PE4 (1)	I	TTL	UART 5模块接收信号。
U5Tx	60	PE5 (1)	O	TTL	UART 5模块发送信号。
U6Rx	43	PD4 (1)	I	TTL	UART 6模块接收信号。
U6Tx	44	PD5 (1)	O	TTL	UART 6模块发送信号。
U7Rx	9	PE0 (1)	I	TTL	UART 7模块接收信号。
U7Tx	8	PE1 (1)	O	TTL	UART 7模块发送信号。
VBAT	37	固定	-	电源	休眠模块的电源供应源它通常连接到电池的正极端并用作备用电池/休眠模块电源供应器的电源。
VDD	11 26 42 54	固定	-	电源	I/O和某些逻辑的电源正极。

表 **20-3.** 按信号名称分类的信号（续）

管脚名称	管脚编号	管脚复用/管脚赋值	管脚类型	缓冲区类型[a]	描述
VDDA	2	固定	-	电源	模拟电路（ADC、模拟比较器等）的电源正端。VDDA 应与 VDD 分开，尽量避免 VDD 中携带的噪声影响模拟功能。VDDA不论系统配置如何，管脚的电压必须符合 表21-5（1031页）中的规格。
VDDC	25 56	固定	-	电源	为主要的逻辑部分（包括处理器内核以及大部分片上外设）供电的电源正端。该管脚上的电压为 1.2 V，由片上 LDO 提供。按照 表21-12（1043页）中的规定，VDDC 管脚只应相互连接或连接外部电容。
WAKE	32	固定	I	TTL	当有效时外部输入将处理器从休眠模式中唤醒。
WT0CCP0	16	PC4 (7)	I/O	TTL	32/64 位宽 Timer 0 捕获/比较/PWM 0。
WT0CCP1	15	PC5 (7)	I/O	TTL	32/64 位宽 Timer 0 捕获/比较/PWM 1。
WT1CCP0	14	PC6 (7)	I/O	TTL	32/64 位宽 Timer 1 捕获/比较/PWM 0。
WT1CCP1	13	PC7 (7)	I/O	TTL	32/64 位宽 Timer 1 捕获/比较/PWM 1。
WT2CCP0	61	PD0 (7)	I/O	TTL	32/64 位宽 Timer 2 捕获/比较/PWM 0。
WT2CCP1	62	PD1 (7)	I/O	TTL	32/64 位宽 Timer 2 捕获/比较/PWM 1。
WT3CCP0	63	PD2 (7)	I/O	TTL	32/64 位宽 Timer 3 捕获/比较/PWM 0。
WT3CCP1	64	PD3 (7)	I/O	TTL	32/64 位宽 Timer 3 捕获/比较/PWM 1。
WT4CCP0	43	PD4 (7)	I/O	TTL	32/64 位宽 Timer 4 捕获/比较/PWM 0。
WT4CCP1	44	PD5 (7)	I/O	TTL	32/64 位宽 Timer 4 捕获/比较/PWM 1。
WT5CCP0	53	PD6 (7)	I/O	TTL	32/64 位宽 Timer 5 捕获/比较/PWM 0。
WT5CCP1	10	PD7 (7)	I/O	TTL	32/64 位宽 Timer 5 捕获/比较/PWM 1。
XOSC0	34	固定	I	模拟	休眠模块晶体振荡器输入或外部时钟参考输入。请注意休眠模块 RTC 采用 32.768-kHz 晶体或 32.768-kHz 振荡器。
XOSC1	36	固定	O	模拟	休眠模块晶体振荡器输出。当使用外部单端参考时钟源时，此管脚应悬空。

a. TTL 表示管脚的电压水平与 TTL 一致。

20.3 按功能分类的信号（GPIO 除外）

表 **20-4.** 按功能分类的信号（**GPIO** 除外）

功能	管脚名称	管脚编号	管脚类型	缓冲区类型[a]	描述
ADC	AIN0	6	I	模拟	模数转换器输入0。
	AIN1	7	I	模拟	模数转换器输入1。
	AIN2	8	I	模拟	模数转换器输入2。
	AIN3	9	I	模拟	模数转换器输入3。
	AIN4	64	I	模拟	模数转换器输入4。
	AIN5	63	I	模拟	模数转换器输入5。
	AIN6	62	I	模拟	模数转换器输入6。
	AIN7	61	I	模拟	模数转换器输入7。
	AIN8	60	I	模拟	模数转换器输入8。
	AIN9	59	I	模拟	模数转换器输入9。
	AIN10	58	I	模拟	模数转换器输入10。
	AIN11	57	I	模拟	模数转换器输入11。

表 **20-4.** 按功能分类的信号（**GPIO** 除外）（续）

功能	管脚名称	管脚编号	管脚类型	缓冲区类型[a]	描述
GPIO 触发	PA0	17	I/O	TTL	GPIO端口A第0位。
	PA1	18	I/O	TTL	GPIO端口A第1位。
	PA2	19	I/O	TTL	GPIO端口A第2位。
	PA3	20	I/O	TTL	GPIO端口A第3位。
	PA4	21	I/O	TTL	GPIO端口A第4位。
	PA5	22	I/O	TTL	GPIO端口A第5位。
	PA6	23	I/O	TTL	GPIO端口A第6位。
	PA7	24	I/O	TTL	GPIO端口A第7位。
	PB0	45	I/O	TTL	GPIO端口B第0位。该管脚不能承受5V最高电压。
	PB1	46	I/O	TTL	GPIO端口B第1位。该管脚不能承受5V最高电压。
	PB2	47	I/O	TTL	GPIO端口B第2位。
	PB3	48	I/O	TTL	GPIO端口B第3位。
	PB4	58	I/O	TTL	GPIO端口B第4位。
	PB5	57	I/O	TTL	GPIO端口B第5位。
	PB6	1	I/O	TTL	GPIO端口B第6位。
	PB7	4	I/O	TTL	GPIO端口B第7位。
	PC0	52	I/O	TTL	GPIO端口C第0位。
	PC1	51	I/O	TTL	GPIO端口C第1位。
	PC2	50	I/O	TTL	GPIO端口C第2位。
	PC3	49	I/O	TTL	GPIO端口C第3位。
	PC4	16	I/O	TTL	GPIO端口C第4位。
	PC5	15	I/O	TTL	GPIO端口C第5位。
	PC6	14	I/O	TTL	GPIO端口C第6位。
	PC7	13	I/O	TTL	GPIO端口C第7位。
	PD0	61	I/O	TTL	GPIO端口D第0位。
	PD1	62	I/O	TTL	GPIO端口D第1位。
	PD2	63	I/O	TTL	GPIO端口D第2位。
	PD3	64	I/O	TTL	GPIO端口D第3位。
	PD4	43	I/O	TTL	GPIO端口D第4位。该管脚不能承受5V最高电压。
	PD5	44	I/O	TTL	GPIO端口D第5位。该管脚不能承受5V最高电压。
	PD6	53	I/O	TTL	GPIO端口D第6位。
	PD7	10	I/O	TTL	GPIO端口D第7位。
	PE0	9	I/O	TTL	GPIO端口E第0位。
	PE1	8	I/O	TTL	GPIO端口E第1位。
	PE2	7	I/O	TTL	GPIO端口E第2位。
	PE3	6	I/O	TTL	GPIO端口E第3位。
	PE4	59	I/O	TTL	GPIO端口E第4位。
	PE5	60	I/O	TTL	GPIO端口E第5位。
	PF0	28	I/O	TTL	GPIO端口F第0位。
	PF1	29	I/O	TTL	GPIO端口F第1位。
	PF2	30	I/O	TTL	GPIO端口F第2位。
	PF3	31	I/O	TTL	GPIO端口F第3位。

表 **20-4.** 按功能分类的信号（**GPIO** 除外）（续）

功能	管脚名称	管脚编号	管脚类型	缓冲区类型[a]	描述
	PF4	5	I/O	TTL	GPIO端口F第4位。
I2C	I2C0SCL	47	I/O	OD	I^2C 模块 0 时钟。请注意该信号具有有源上拉。不应将相应的端口管脚配置为开漏。
	I2C0SDA	48	I/O	OD	I^2C 模块 0 数据。
	I2C1SCL	23	I/O	OD	I^2C 模块 1 时钟。请注意该信号具有有源上拉。不应将相应的端口管脚配置为开漏。
	I2C1SDA	24	I/O	OD	I^2C 模块 1 数据。
	I2C2SCL	59	I/O	OD	I^2C 模块 2 时钟。请注意该信号具有有源上拉。不应将相应的端口管脚配置为开漏。
	I2C2SDA	60	I/O	OD	I^2C 模块 2 数据。
	I2C3SCL	61	I/O	OD	I^2C 模块 3 时钟。请注意该信号具有有源上拉。不应将相应的端口管脚配置为开漏。
	I2C3SDA	62	I/O	OD	I^2C 模块 3 数据。
JTAG/SWD/SWO	SWCLK	52	I	TTL	JTAG/SWD CLK信号。
	SWDIO	51	I/O	TTL	JTAG TMS及SWDIO信号。
	SWO	49	O	TTL	JTAG TDO及SWO信号。
	TCK	52	I	TTL	JTAG/SWD CLK信号。
	TDI	50	I	TTL	JTAG TDI信号。
	TDO	49	O	TTL	JTAG TDO及SWO信号。
	TMS	51	I	TTL	JTAG TMS及SWDIO信号。
SSI	SSI0Clk	19	I/O	TTL	SSI模块0时钟
	SSI0Fss	20	I/O	TTL	SSI 模块 0 帧信号
	SSI0Rx	21	I	TTL	SSI模块0接收
	SSI0Tx	22	O	TTL	SSI模块0发送
	SSI1Clk	30 61	I/O	TTL	SSI 模块 1 时钟信号。
	SSI1Fss	31 62	I/O	TTL	SSI 模块 1 帧信号。
	SSI1Rx	28 63	I	TTL	SSI 模块 1 接收信号。
	SSI1Tx	29 64	O	TTL	SSI 模块 1 发送信号。
	SSI2Clk	58	I/O	TTL	SSI 模块 2 时钟信号。
	SSI2Fss	57	I/O	TTL	SSI 模块 2 帧信号。
	SSI2Rx	1	I	TTL	SSI 模块 2 接收信号。
	SSI2Tx	4	O	TTL	SSI 模块 2 发送信号。
	SSI3Clk	61	I/O	TTL	SSI 模块 3 时钟信号。
	SSI3Fss	62	I/O	TTL	SSI 模块 3 帧信号。
	SSI3Rx	63	I	TTL	SSI 模块 3 接收信号。
	SSI3Tx	64	O	TTL	SSI 模块 3 发送信号。

表 **20-4.** 按功能分类的信号（**GPIO** 除外）（续）

功能	管脚名称	管脚编号	管脚类型	缓冲区类型[a]	描述
UART	U0Rx	17	I	TTL	UART 0模块接收信号。
	U0Tx	18	O	TTL	UART 0模块发送信号。
	U1CTS	15 29	I	TTL	UART 模块 1 CTS（Clear to Send，允许发送）调制解调器流控输入信号。
	U1RTS	16 28	O	TTL	UART 模块 1 RTS（Request to Send，请求发送）调制解调器流控输出线。
	U1Rx	16 45	I	TTL	UART 1模块接收信号。
	U1Tx	15 46	O	TTL	UART 1模块发送信号。
	U2Rx	53	I	TTL	UART 2模块接收信号。
	U2Tx	10	O	TTL	UART 2模块发送信号。
	U3Rx	14	I	TTL	UART 3模块接收信号。
	U3Tx	13	O	TTL	UART 3模块发送信号。
	U4Rx	16	I	TTL	UART 4模块接收信号。
	U4Tx	15	O	TTL	UART 4模块发送信号。
	U5Rx	59	I	TTL	UART 5模块接收信号。
	U5Tx	60	O	TTL	UART 5模块发送信号。
	U6Rx	43	I	TTL	UART 6模块接收信号。
	U6Tx	44	O	TTL	UART 6模块发送信号。
	U7Rx	9	I	TTL	UART 7模块接收信号。
	U7Tx	8	O	TTL	UART 7模块发送信号。
休眠	GNDX	35	-	电源	休眠震荡器的接地。当使用晶振时钟源时，该管脚应连接到数字接地和晶振负载电容。使用外部震荡器时，该管脚应连接到数字接地。
	HIB	33	O	TTL	该输出指示处理器是否处于休眠模式。
	VBAT	37	-	电源	休眠模块的电源供应源它通常连接到电池的正极端并用作备用电池/休眠模块电源供应器的电源。
	WAKE	32	I	TTL	当有效时外部输入将处理器从休眠模式中唤醒。
	XOSC0	34	I	模拟	休眠模块晶体振荡器输入或外部时钟参考输入。请注意休眠模块 RTC 采用 32.768-kHz 晶体或 32.768-kHz 振荡器。
	XOSC1	36	O	模拟	休眠模块晶体振荡器输出。当使用外部单端参考时钟源时，此管脚应悬空。
内核	TRCLK	31	O	TTL	跟踪时钟信号。
	TRD0	30	O	TTL	跟踪数据信号 0。
	TRD1	29	O	TTL	跟踪数据信号 1。

表 **20-4.** 按功能分类的信号（**GPIO** 除外）（续）

功能	管脚名称	管脚编号	管脚类型	缓冲区类型[a]	描述
功率	GND	12 27 39 55	-	电源	逻辑和I/O管脚的地参考。
	GNDA	3	-	电源	模拟电路（ADC、模拟比较器等）的接地参考。这些与 GND 分开，以期尽量避免 VDD 中携带的电噪声影响模拟功能。
	VDD	11 26 42 54	-	电源	I/O和某些逻辑的电源正极。
	VDDA	2	-	电源	模拟电路（ADC、模拟比较器等）的电源正端。VDDA 应与 VDD 分开，尽量避免 VDD 中携带的噪声影响模拟功能。VDDA不论系统配置如何，管脚的电压必须符合 表21-5（1031页）中的规格。
	VDDC	25 56	-	电源	为主要的逻辑部分（包括处理器内核以及大部分片上外设）供电的电源正端。该管脚上的电压为 1.2 V，由片上 LDO 提供。按照 表21-12（1043页） 中的规定，VDDC 管脚只应相互连接或连接外部电容。
控制器局域网	CAN0Rx	28 58 59	I	TTL	CAN模块0接收信号。
	CAN0Tx	31 57 60	O	TTL	CAN模块0发送信号。
模拟比较器触发	C0+	14	I	模拟	模拟比较器0正极输入
	C0-	13	I	模拟	模拟比较器0负极输入
	C0o	28	O	TTL	模拟比较器0输出端。
	C1+	15	I	模拟	模拟比较器1正极输入
	C1-	16	I	模拟	模拟比较器1负极输入
	C1o	29	O	TTL	模拟比较器1输出端。
系统控制; 时钟	NMI	10 28	I	TTL	不可屏蔽的中断
	OSC0	40	I	模拟	主振荡器晶体输入或外部时钟参考输入。
	OSC1	41	O	模拟	主振荡器晶体输出。当使用外部单端参考时钟源时，此管脚应悬空。
	RST	38	I	TTL	系统复位输入

表 20-4. 按功能分类的信号（GPIO 除外）（续）

功能	管脚名称	管脚编号	管脚类型	缓冲区类型[a]	描述
通用定时器	T0CCP0	1 28	I/O	TTL	16/32 位 Timer 0 捕获/比较/PWM 0。
	T0CCP1	4 29	I/O	TTL	16/32 位 Timer 0 捕获/比较/PWM 1。
	T1CCP0	30 58	I/O	TTL	16/32 位 Timer 1 捕获/比较/PWM 0。
	T1CCP1	31 57	I/O	TTL	16/32 位 Timer 1 捕获/比较/PWM 1。
	T2CCP0	5 45	I/O	TTL	16/32 位 Timer 2 捕获/比较/PWM 0。
	T2CCP1	46	I/O	TTL	16/32 位 Timer 2 捕获/比较/PWM 1。
	T3CCP0	47	I/O	TTL	16/32 位 Timer 3 捕获/比较/PWM 0。
	T3CCP1	48	I/O	TTL	16/32 位 Timer 3 捕获/比较/PWM 1。
	T4CCP0	52	I/O	TTL	16/32 位 Timer 4 捕获/比较/PWM 0。
	T4CCP1	51	I/O	TTL	16/32 位 Timer 4 捕获/比较/PWM 1。
	T5CCP0	50	I/O	TTL	16/32 位 Timer 5 捕获/比较/PWM 0。
	T5CCP1	49	I/O	TTL	16/32 位 Timer 5 捕获/比较/PWM 1。
	WT0CCP0	16	I/O	TTL	32/64 位宽 Timer 0 捕获/比较/PWM 0。
	WT0CCP1	15	I/O	TTL	32/64 位宽 Timer 0 捕获/比较/PWM 1。
	WT1CCP0	14	I/O	TTL	32/64 位宽 Timer 1 捕获/比较/PWM 0。
	WT1CCP1	13	I/O	TTL	32/64 位宽 Timer 1 捕获/比较/PWM 1。
	WT2CCP0	61	I/O	TTL	32/64 位宽 Timer 2 捕获/比较/PWM 0。
	WT2CCP1	62	I/O	TTL	32/64 位宽 Timer 2 捕获/比较/PWM 1。
	WT3CCP0	63	I/O	TTL	32/64 位宽 Timer 3 捕获/比较/PWM 0。
	WT3CCP1	64	I/O	TTL	32/64 位宽 Timer 3 捕获/比较/PWM 1。
	WT4CCP0	43	I/O	TTL	32/64 位宽 Timer 4 捕获/比较/PWM 0。
	WT4CCP1	44	I/O	TTL	32/64 位宽 Timer 4 捕获/比较/PWM 1。
	WT5CCP0	53	I/O	TTL	32/64 位宽 Timer 5 捕获/比较/PWM 0。
	WT5CCP1	10	I/O	TTL	32/64 位宽 Timer 5 捕获/比较/PWM 1。

a. TTL 表示管脚的电压水平与 TTL 一致。

20.4 GPIO 管脚和复用功能

表 20-5. GPIO 管脚和复用功能

IO	管脚	模拟功能	数字功能（GPIOPCTL PMCx 位域编码）[a]										
			1	2	3	4	5	6	7	8	9	14	15
PA0	17	-	U0Rx	-	-	-	-	-	-	-	-	-	-
PA1	18	-	U0Tx	-	-	-	-	-	-	-	-	-	-
PA2	19	-	-	SSI0Clk	-	-	-	-	-	-	-	-	-
PA3	20	-	-	SSI0Fss	-	-	-	-	-	-	-	-	-
PA4	21	-	-	SSI0Rx	-	-	-	-	-	-	-	-	-
PA5	22	-	-	SSI0Tx	-	-	-	-	-	-	-	-	-
PA6	23	-	-	-	I2C1SCL	-	-	-	-	-	-	-	-

表 20-5. GPIO 管脚和复用功能（续）

IO	管脚	模拟功能	数字功能（GPIOPCTL PMCx 位域编码）[a]										
			1	2	3	4	5	6	7	8	9	14	15
PA7	24	-	-	-	I2C1SDA	-	-	-	-	-	-	-	-
PB0	45	-	U1Rx	-	-	-	-	-	T2CCP0	-	-	-	-
PB1	46	-	U1Tx	-	-	-	-	-	T2CCP1	-	-	-	-
PB2	47	-	-	-	I2C0SCL	-	-	-	T3CCP0	-	-	-	-
PB3	48	-	-	-	I2C0SDA	-	-	-	T3CCP1	-	-	-	-
PB4	58	AIN10	-	SSI2Clk	-	-	-	-	T1CCP0	CAN0Rx	-	-	-
PB5	57	AIN11	-	SSI2Fss	-	-	-	-	T1CCP1	CAN0Tx	-	-	-
PB6	1	-	-	SSI2Rx	-	-	-	-	T0CCP0	-	-	-	-
PB7	4	-	-	SSI2Tx	-	-	-	-	T0CCP1	-	-	-	-
PC0	52	-	TCK SWCLK	-	-	-	-	-	T4CCP0	-	-	-	-
PC1	51	-	TMS SWDIO	-	-	-	-	-	T4CCP1	-	-	-	-
PC2	50	-	TDI	-	-	-	-	-	T5CCP0	-	-	-	-
PC3	49	-	TDO SWO	-	-	-	-	-	T5CCP1	-	-	-	-
PC4	16	C1-	U4Rx	U1Rx	-	-	-	-	WT0CCP0	U1RTS	-	-	-
PC5	15	C1+	U4Tx	U1Tx	-	-	-	-	WT0CCP1	U1CTS	-	-	-
PC6	14	C0+	U3Rx	-	-	-	-	-	WT1CCP0	-	-	-	-
PC7	13	C0-	U3Tx	-	-	-	-	-	WT1CCP1	-	-	-	-
PD0	61	AIN7	SSI3Clk	SSI1Clk	I2C3SCL	-	-	-	WT2CCP0	-	-	-	-
PD1	62	AIN6	SSI3Fss	SSI1Fss	I2C3SDA	-	-	-	WT2CCP1	-	-	-	-
PD2	63	AIN5	SSI3Rx	SSI1Rx	-	-	-	-	WT3CCP0	-	-	-	-
PD3	64	AIN4	SSI3Tx	SSI1Tx	-	-	-	-	WT3CCP1	-	-	-	-
PD4	43	-	U6Rx	-	-	-	-	-	WT4CCP0	-	-	-	-
PD5	44	-	U6Tx	-	-	-	-	-	WT4CCP1	-	-	-	-
PD6	53	-	U2Rx	-	-	-	-	-	WT5CCP0	-	-	-	-
PD7	10	-	U2Tx	-	-	-	-	-	WT5CCP1	NMI	-	-	-
PE0	9	AIN3	U7Rx	-	-	-	-	-	-	-	-	-	-
PE1	8	AIN2	U7Tx	-	-	-	-	-	-	-	-	-	-
PE2	7	AIN1	-	-	-	-	-	-	-	-	-	-	-
PE3	6	AIN0	-	-	-	-	-	-	-	-	-	-	-
PE4	59	AIN9	U5Rx	-	I2C2SCL	-	-	-	-	CAN0Rx	-	-	-
PE5	60	AIN8	U5Tx	-	I2C2SDA	-	-	-	-	CAN0Tx	-	-	-
PF0	28	-	U1RTS	SSI1Rx	CAN0Rx	-	-	-	T0CCP0	NMI	C0o	-	-
PF1	29	-	U1CTS	SSI1Tx	-	-	-	-	T0CCP1	-	C1o	TRD1	-
PF2	30	-	-	SSI1Clk	-	-	-	-	T1CCP0	-	-	TRD0	-
PF3	31	-	-	SSI1Fss	CAN0Tx	-	-	-	T1CCP1	-	-	TRCLK	-
PF4	5	-	-	-	-	-	-	-	T2CCP0	-	-	-	-

a. 带灰色阴影的数字信号是相应 GPIO 管脚的上电默认值。 本器件不使用编码 10-13。

20.5 复用功能的可能的管脚赋值

表 **20-6.** 复用功能的可能的管脚赋值

# 种可能的赋值	复用功能	GPIO 功能
1	AIN0	PE3
	AIN1	PE2
	AIN10	PB4
	AIN11	PB5
	AIN2	PE1
	AIN3	PE0
	AIN4	PD3
	AIN5	PD2
	AIN6	PD1
	AIN7	PD0
	AIN8	PE5
	AIN9	PE4
	C0+	PC6
	C0-	PC7
	C0o	PF0
	C1+	PC5
	C1-	PC4
	C1o	PF1
	I2C0SCL	PB2
	I2C0SDA	PB3
	I2C1SCL	PA6
	I2C1SDA	PA7
	I2C2SCL	PE4
	I2C2SDA	PE5
	I2C3SCL	PD0
	I2C3SDA	PD1
	SSI0Clk	PA2
	SSI0Fss	PA3
	SSI0Rx	PA4
	SSI0Tx	PA5
	SSI2Clk	PB4
	SSI2Fss	PB5
	SSI2Rx	PB6
	SSI2Tx	PB7
	SSI3Clk	PD0
	SSI3Fss	PD1
	SSI3Rx	PD2
	SSI3Tx	PD3
	SWCLK	PC0
	SWDIO	PC1

表 20-6. 复用功能的可能的管脚赋值（续）

# 种可能的赋值	复用功能	GPIO 功能
	SWO	PC3
	T2CCP1	PB1
	T3CCP0	PB2
	T3CCP1	PB3
	T4CCP0	PC0
	T4CCP1	PC1
	T5CCP0	PC2
	T5CCP1	PC3
	TCK	PC0
	TDI	PC2
	TDO	PC3
	TMS	PC1
	TRCLK	PF3
	TRD0	PF2
	TRD1	PF1
	U0Rx	PA0
	U0Tx	PA1
	U2Rx	PD6
	U2Tx	PD7
	U3Rx	PC6
	U3Tx	PC7
	U4Rx	PC4
	U4Tx	PC5
	U5Rx	PE4
	U5Tx	PE5
	U6Rx	PD4
	U6Tx	PD5
	U7Rx	PE0
	U7Tx	PE1
	WT0CCP0	PC4
	WT0CCP1	PC5
	WT1CCP0	PC6
	WT1CCP1	PC7
	WT2CCP0	PD0
	WT2CCP1	PD1
	WT3CCP0	PD2
	WT3CCP1	PD3
	WT4CCP0	PD4
	WT4CCP1	PD5
	WT5CCP0	PD6
	WT5CCP1	PD7

表 **20-6.** 复用功能的可能的管脚赋值（续）

# 种可能的赋值	复用功能	GPIO 功能
2	NMI	PD7 PF0
	SSI1Clk	PD0 PF2
	SSI1Fss	PD1 PF3
	SSI1Rx	PD2 PF0
	SSI1Tx	PD3 PF1
	T0CCP0	PB6 PF0
	T0CCP1	PB7 PF1
	T1CCP0	PB4 PF2
	T1CCP1	PB5 PF3
	T2CCP0	PB0 PF4
	U1CTS	PC5 PF1
	U1RTS	PC4 PF0
	U1Rx	PB0 PC4
	U1Tx	PB1 PC5
3	CAN0Rx	PB4 PE4 PF0
	CAN0Tx	PB5 PE5 PF3

20.6 未用管脚的处理

对于64管脚LQFP封装中的设备，如果在特定的系统中并未用到某些信号，可按照表20-7（1028页）对相应的功能信号进行处理。表中列出了两种选项：一般的处理方法以及推荐的处理方法，其中按照推荐的方法处理有助于降低功耗并改善 EMC 性能。假如系统中并未使用某个功能模块，并且已将其输入端接地，那么必须避免使能该模块的时钟（将 RCGCx 寄存器中对应的标志位置位）。

表 **20-7.** 未用信号的连接（**64** 管脚 **LQFP**）

功能	信号名称	管脚编号	一般的处理方法	推荐的处理方法
GPIO 触发	所有未用的GPIO	-	悬空	GND
休眠	HIB	33	悬空	悬空
	VBAT	37	悬空	VDD
	WAKE	32	悬空	GND
	XOSC0	34	悬空	GND
	XOSC1	36	悬空	悬空
	GNDX	35	GND	GND
无连接	悬空	参考表20-3（1014页）中的 NC 管脚号。	悬空	悬空
系统控制	OSC0	40	悬空	GND
	OSC1	41	悬空	悬空
	RST	38	VDD	上拉，如图5-1（189页）所示。

21 Electrical Characteristics

21.1 Maximum Ratings

The maximum ratings are the limits to which the device can be subjected without permanently damaging the device. Device reliability may be adversely affected by exposure to absolute-maximum ratings for extended periods.

注意: The device is not guaranteed to operate properly at the maximum ratings.

表 21-1. Maximum Ratings

Parameter	Parameter Name[a]	Value		Unit
		Min	Max	
V_{DD}	V_{DD} supply voltage	0	4	V
V_{DDA}	V_{DDA} supply voltage[b]	0	4	V
V_{BAT}	V_{BAT} battery supply voltage	0	4	V
V_{BATRMP}	V_{BAT} battery supply voltage ramp time	0	0.7	V/µs
V_{IN_GPIO}	Input voltage on GPIOs, regardless of whether the microcontroller is powered[cde]	-0.3	5.5	V
	Input voltage for PD4, PD5, PB0 and PB1 when configured as GPIO	-0.3	V_{DD} + 0.3	V
$I_{GPIOMAX}$	Maximum current per output pin	-	25	mA
T_S	Unpowered storage temperature range	-65	150	°C
T_{JMAX}	Maximum junction temperature	-	150	°C

a. Voltages are measured with respect to GND.

b. To ensure proper operation, VDDA must be powered before VDD if sourced from different supplies, or connected to the same supply as VDD. Note that the minimum operating voltage for VDD differs from the minimum operating voltage for VDDA. This change should be accounted for in the system design if both are sourced from the same supply. There is not a restriction on order for powering off.

c. Applies to static and dynamic signals including overshoot.

d. Refer to 图21-16（1055页）for a representation of the ESD protection on GPIOs.

e. For additional details, see the note on GPIO pad tolerance in “GPIO Module Characteristics”（1054页）.

重要: This device contains circuitry to protect the I/Os against damage due to high-static voltages; however, it is advised that normal precautions be taken to avoid application of any voltage higher than maximum-rated voltages to this high-impedance circuit. Reliability of operation is enhanced if unused inputs are connected to an appropriate logic voltage level (see “未用管脚的处理”（1028页）).

表 21-2. ESD Absolute Maximum Ratings

	Parameter	Min	Nom	Max	Unit
Component-Level ESD Stress Voltage[a]	V_{ESDHBM} [b]	-	-	2.0	kV
	V_{ESDCDM} [c]	-	-	500	V

a. Electrostatic discharge (ESD) to measure device sensitivity/immunity to damage caused by electrostatic discharges in device.

b. Level listed is passing level per ANSI/ESDA/JEDEC JS-001. JEDEC document JEP155 states that 500V HBM allows safe manufacturing with a standard ESD control process.

c. Level listed is the passing level per EIA-JEDEC JESD22-C101E. JEDEC document JEP157 states that 250V CDM allows safe manufacturing with a standard ESD control process.

21.2 Operating Characteristics

表 21-3. Temperature Characteristics

Characteristic	Symbol	Value	Unit
Ambient operating temperature range	T_A	-40 to +85	°C
Case operating temperature range	T_C	-40 to +93	°C
Junction operating temperature range	T_J	-40 to +96	°C

表 21-4. Thermal Characteristics[a]

Characteristic	Symbol	Value	Unit
Thermal resistance (junction to ambient)[b]	Θ_{JA}	54.8	°C/W
Thermal resistance (junction to board)[b]	Θ_{JB}	27.5	°C/W
Thermal resistance (junction to case)[b]	Θ_{JC}	15.8	°C/W
Thermal metric (junction to top of package)	Ψ_{JT}	0.7	°C/W
Thermal metric (junction to board)	Ψ_{JB}	27.1	°C/W
Junction temperature formula	T_J	$T_C + (P \bullet \Psi_{JT})$ $T_{PCB} + (P \bullet \Psi_{JB})$[c] $T_A + (P \bullet \Theta_{JA})$[d] $T_B + (P \bullet \Theta_{JB})$[ef]	°C

a. For more details about thermal metrics and definitions, see the “Semiconductor and IC Package Thermal Metrics Application Report” (literature number SPRA953).

b. Junction to ambient thermal resistance (Θ_{JA}), junction to board thermal resistance (Θ_{JB}), and junction to case thermal resistance (Θ_{JC}) numbers are determined by a package simulator.

c. T_{PCB} is the temperature of the board acquired by following the steps listed in the EAI/JESD 51-8 standard summarized in the “Semiconductor and IC Package Thermal Metrics Application Report” (literature number SPRA953).

d. Because Θ_{JA} is highly variable and based on factors such as board design, chip/pad size, altitude, and external ambient temperature, it is recommended that equations containing Ψ_{JT} and Ψ_{JB} be used for best results.

e. T_B is temperature of the board.

f. Θ_{JB} is not a pure reflection of the internal resistance of the package because it includes the resistance of the testing board and environment. It is recommended that equations containing Ψ_{JT} and Ψ_{JB} be used for best results.

21.3 Recommended Operating Conditions

For special high-current applications, the GPIO output buffers may be used with the following restrictions. With the GPIO pins configured as 8-mA output drivers, a total of four GPIO outputs may be used to sink current loads up to 18 mA each. At 18-mA sink current loading, the V_{OL} value is specified as 1.2 V. The high-current GPIO package pins must be selected such that there are only a maximum of two per side of the physical package with the total number of high-current GPIO outputs not exceeding four for the entire package.

表 21-5. Recommended DC Operating Conditions

Parameter	Parameter Name	Min	Nom	Max	Unit
V_{DD}	V_{DD} supply voltage	3.15	3.3	3.63	V
V_{DDA}	V_{DDA} supply voltage	2.97	3.3	3.63	V
V_{DDC}	V_{DDC} supply voltage	1.08	1.2	1.32	V
V_{DDCDS}[ab]	V_{DDC} supply voltage, Deep-sleep mode	1.08	-	1.32	V

a. These values are valid when LDO is in operation.

b. There are peripheral timing restrictions for SSI and LPC in Deep-sleep mode. Please refer to those peripheral characteristic sections for more information.

表 21-6. Recommended GPIO Pad Operating Conditions

Parameter	Parameter Name	Min	Nom	Max	Unit
V_{IH}	GPIO high-level input voltage	0.65 * V_{DD}	-	5.5	V
V_{IL}	GPIO low-level input voltage	0	-	0.35 * V_{DD}	V
V_{HYS}	GPIO input hysteresis	0.2	-	-	V
V_{OH}	GPIO high-level output voltage	2.4	-	-	V
V_{OL}	GPIO low-level output voltage	-	-	0.4	V
I_{OH}	High-level source current, V_{OH}=2.4 V[a]				
	2-mA Drive	2.0	-	-	mA
	4-mA Drive	4.0	-	-	mA
	8-mA Drive	8.0	-	-	mA
I_{OL}	Low-level sink current, V_{OL}=0.4 V[a]				
	2-mA Drive	2.0	-	-	mA
	4-mA Drive	4.0	-	-	mA
	8-mA Drive	8.0	-	-	mA
	8-mA Drive, V_{OL}=1.2 V	18.0	-	-	mA

a. I_O specifications reflect the maximum current where the corresponding output voltage meets the V_{OH}/V_{OL} thresholds. I_O current can exceed these limits (subject to absolute maximum ratings).

表 21-7. GPIO Current Restrictions[a]

Parameter	Parameter Name	Min	Nom	Max	Unit
I_{MAXL}	Cumulative maximum GPIO current per side, left[b]	-	-	30	mA
I_{MAXB}	Cumulative maximum GPIO current per side, bottom[b]	-	-	35	mA
I_{MAXR}	Cumulative maximum GPIO current per side, right[b]	-	-	40	mA
I_{MAXT}	Cumulative maximum GPIO current per side, top[b]	-	-	40	mA

a. Based on design simulations, not tested in production.

b. Sum of sink and source current for GPIOs as shown in 表21-8（1032页）.

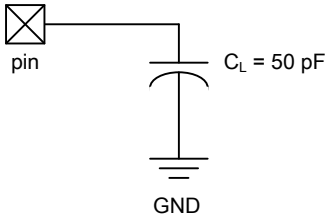
表 21-8. GPIO Package Side Assignments

Side	GPIOs
Left	PB[6-7], PC[4-7], PD7, PE[0-3], PF4
Bottom	PA[0-7], PF[0-3]
Right	PB[0-3], PD[4-5]
Top	PB[4-5], PC[0-3], PD[0-3,6], PE[4-5]

21.4 Load Conditions

Unless otherwise specified, the following conditions are true for all timing measurements.

图 21-1. Load Conditions



21.5 JTAG and Boundary Scan

表 21-9. JTAG Characteristics

Parameter No.	Parameter	Parameter Name	Min	Nom	Max	Unit
J1	F_{TCK}	TCK operational clock frequency[a]	0	-	10	MHz
J2	T_{TCK}	TCK operational clock period	100	-	-	ns
J3	T_{TCK_LOW}	TCK clock Low time	-	$t_{TCK}/2$	-	ns
J4	T_{TCK_HIGH}	TCK clock High time	-	$t_{TCK}/2$	-	ns
J5	T_{TCK_R}	TCK rise time	0	-	10	ns
J6	T_{TCK_F}	TCK fall time	0	-	10	ns
J7	T_{TMS_SU}	TMS setup time to TCK rise	8	-	-	ns
J8	T_{TMS_HLD}	TMS hold time from TCK rise	4	-	-	ns
J9	T_{TDI_SU}	TDI setup time to TCK rise	18	-	-	ns
J10	T_{TDI_HLD}	TDI hold time from TCK rise	4	-	-	ns
J11	T_{TDO_ZDV}	TCK fall to Data Valid from High-Z, 2-mA drive	-	13	35	ns
		TCK fall to Data Valid from High-Z, 4-mA drive		9	26	ns
		TCK fall to Data Valid from High-Z, 8-mA drive		8	26	ns
		TCK fall to Data Valid from High-Z, 8-mA drive with slew rate control		10	29	ns
J12	T_{TDO_DV}	TCK fall to Data Valid from Data Valid, 2-mA drive	-	14	20	ns
		TCK fall to Data Valid from Data Valid, 4-mA drive		10	26	ns
		TCK fall to Data Valid from Data Valid, 8-mA drive		8	21	ns
		TCK fall to Data Valid from Data Valid, 8-mA drive with slew rate control		10	26	ns
J13	T_{TDO_DVZ}	TCK fall to High-Z from Data Valid, 2-mA drive	-	7	16	ns
		TCK fall to High-Z from Data Valid, 4-mA drive		7	16	ns
		TCK fall to High-Z from Data Valid, 8-mA drive		7	16	ns
		TCK fall to High-Z from Data Valid, 8-mA drive with slew rate control		8	19	ns

a. A ratio of at least 8:1 must be kept between the system clock and TCK.

图 21-2. JTAG Test Clock Input Timing

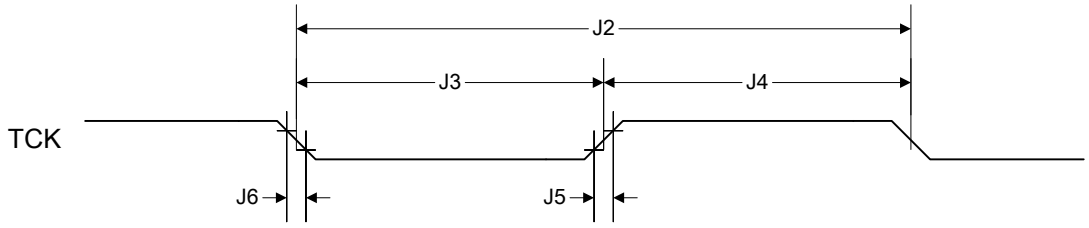
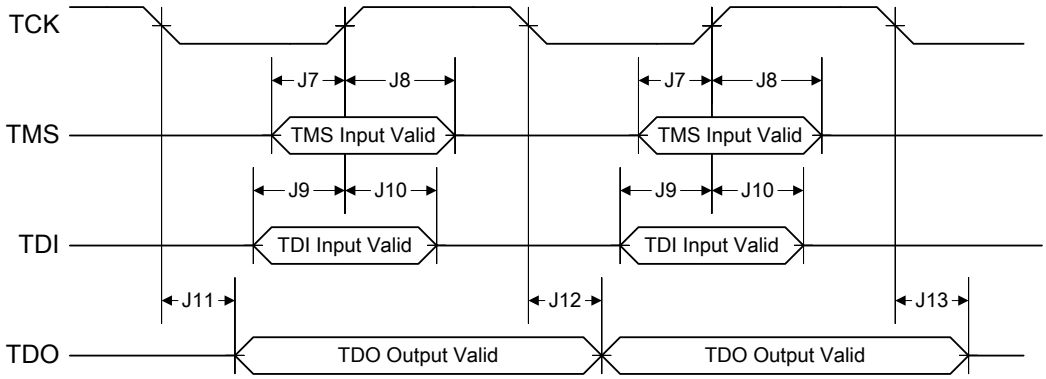


图 21-3. JTAG Test Access Port (TAP) Timing



21.6 Power and Brown-Out

表 21-10. Power-On and Brown-Out Levels

Parameter No.	Parameter	Parameter Name	Min	Nom	Max	Unit
P1	T_{VDDA_RISE}	Analog Supply Voltage (VDDA) Rise Time	-	-	∞	µs
P2	T_{VDD_RISE}	I/O Supply Voltage (VDD) Rise Time	-	-	∞	µs
P3	T_{VDDC_RISE}[a]	Core Supply Voltage (VDDC) Rise Time	12.50	-	50.00	µs
P4	V_{POR}	Power-On Reset Threshold	2.00	2.30	2.60	V
P5	V_{VDDA_POK}	VDDA Power-OK Threshold (Rising Edge)	2.70	2.85	3.00	V
		VDDA Power-OK Threshold (Falling Edge)	2.71	2.80	2.89	V
P6	V_{VDD_POK}[b]	VDD Power-OK Threshold (Rising Edge)	2.85	3.00	3.15	V
		VDD Power-OK Threshold (Falling Edge)	2.70	2.78	2.87	V
P7	V_{VDD_BOR0}	Brown-Out 0 Reset Threshold	2.93	3.02	3.11	V
P8	V_{VDD_BOR1}	Brown-Out 1 Reset Threshold	2.83	2.92	3.01	V
P9	V_{VDDC_POK}	VDDC Power-OK Threshold (Rising Edge)	0.80	0.95	1.10	V
		VDDC Power-OK Threshold (Falling Edge)	0.71	0.80	0.89	V

a. The MIN and MAX value is based on an external filter capacitor load within the range of CLDO. Please refer to “On-Chip Low Drop-Out (LDO) Regulator”（1043页）for the CLDO value.

b. Digital logic, Flash memory, and SRAM are all designed to operate at VDD voltages below 2.70 V. The internal POK reset protects the device from unpredictable operation on power down.

21.6.1 VDDA Levels

The V_{DDA} supply has two monitors:

- Power-On Reset (POR)
- Power-OK (POK)

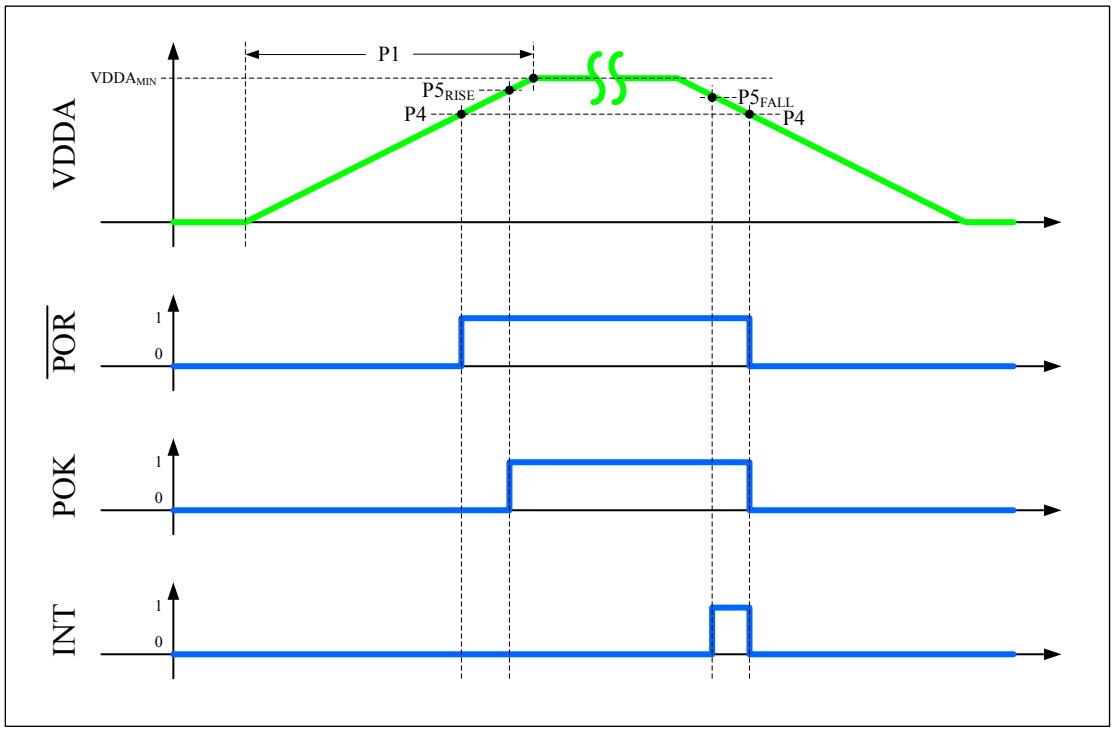
The POR monitor is used to keep the analog circuitry in reset until the V_{DDA} supply has reached the correct range for the analog circuitry to begin operating. The POK monitor is used to keep the digital circuitry in reset until the V_{DDA} power supply is at an acceptable operational level. The digital Power-On Reset (Digital POR) is only released when the Power-On Reset has de-asserted and all of the Power-OK monitors for each of the supplies indicate that power levels are in operational ranges.

Once the V_{DDA} POK monitor has released the digital Power-On Reset on the initial power-up, voltage drops on the V_{DDA} supply will only be reflected in the following bits. The digital Power-On Reset will not be re-asserted.

- VDDARIS bit in the Raw Interrupt Status (RIS) register (see 213页).
- VDDAMIS bit in the Masked Interrupt Status and Clear (MISC) register (see 217页). This bit is set only if the VDDAIM bit in the Interrupt Mask Control (IMC) register has been set.

图21-4（1037页）shows the relationship between V_{DDA}, POR, POK, and an interrupt event.

图 21-4. Power Assertions versus VDDA Levels



21.6.2 VDD Levels

The V_{DD} supply has three monitors:

- Power-OK (POK)
- Brown-Out Reset0 (BOR0)
- Brown-Out Reset1 (BOR1)

The POK monitor is used to keep the digital circuitry in reset until the V_{DD} power supply is at an acceptable operational level. The digital Power-On Reset (Digital POR) is only released when the Power-On Reset has de-asserted and all of the Power-OK monitors for each of the supplies indicate that power levels are in operational ranges. The BOR0 and the BOR1 monitors are used to generate a reset to the device or assert an interrupt if the V_{DD} supply drops below its operational range. The BOR1 monitor's threshold is in between the BOR0 and POK thresholds.

If either a BOR0 event or a BOR1 event occurs, the following bits are affected:

- BOR0RIS or BOR1RIS bits in the Raw Interrupt Status (RIS) register (see 213页).
- BOR0MIS or BOR1MIS bits in the Masked Interrupt Status and Clear (MISC) register (see 217页). These bits are set only if the respective BOR0IM or BOR1IM bits in the Interrupt Mask Control (IMC) register have been set.
- BOR bit in the Reset Cause (RESC) register (see 219页). This bit is set only if either of the BOR0 or BOR1 events have been configured to initiate a reset.

In addition, the following bits control both the BOR0 and BOR1 events:

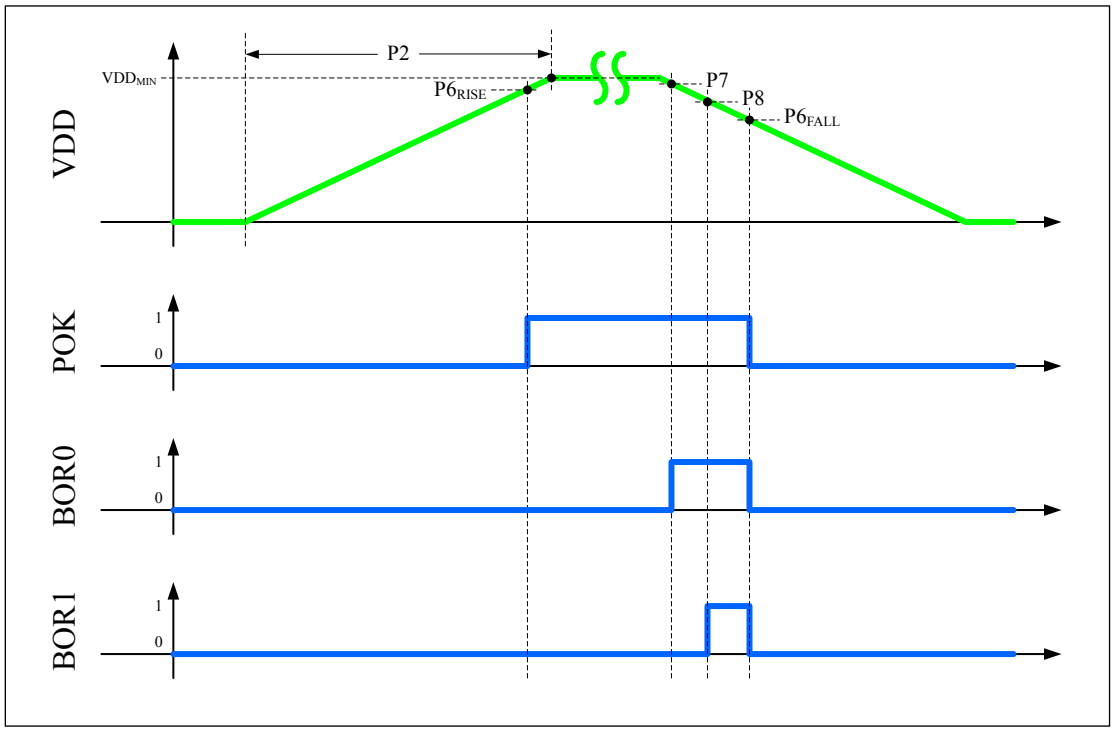
- BOR0IM or BOR1IM bits in the Interrupt Mask Control (IMC) register (see 215页).

- BOR0 or BOR1 bits in the Power-On and Brown-Out Reset Control (PBORCTL) register (see 212页).

图21-5（1038页）shows the relationship between:

- V_{DD}, POK, and a BOR0 event
- V_{DD}, POK, and a BOR1 event

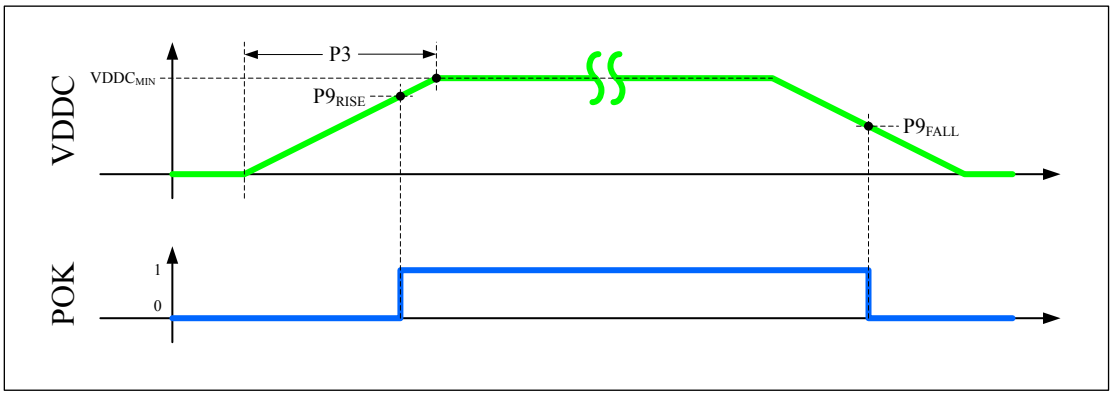
图 21-5. Power and Brown-Out Assertions versus VDD Levels



21.6.3 VDDC Levels

The V_{DDC} supply has one monitor: the Power-OK (POK). The POK monitor is used to keep the digital circuitry in reset until the V_{DDC} power supply is at an acceptable operational level. The digital Power-On Reset (`Digital POR`) is only released when the Power-On Reset has de-asserted and all of the Power-OK monitors for each of the supplies indicate that power levels are in operational ranges. 图21-6（1039页）shows the relationship between POK and V_{DDC}.

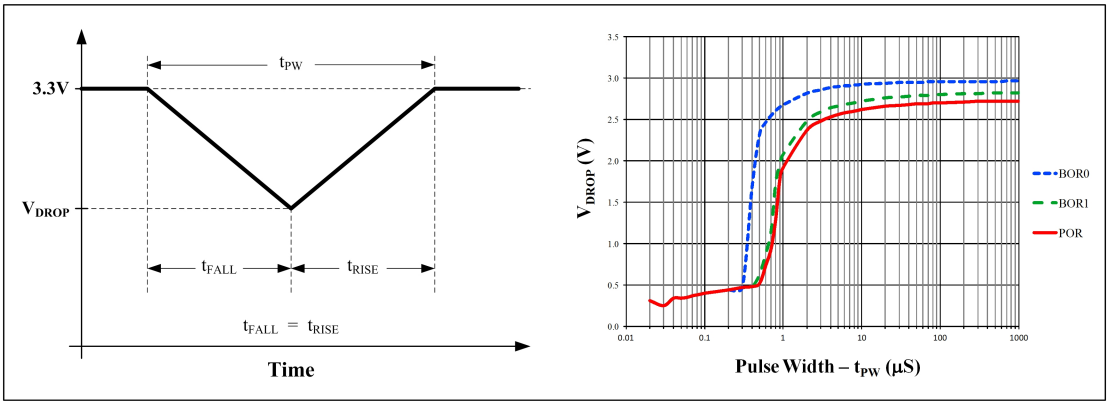
图 21-6. POK assertion vs VDDC



21.6.4 VDD Glitches

图21-7（1039页）shows the response of the BOR0, BOR1, and the POR circuit to glitches on the V_{DD} supply.

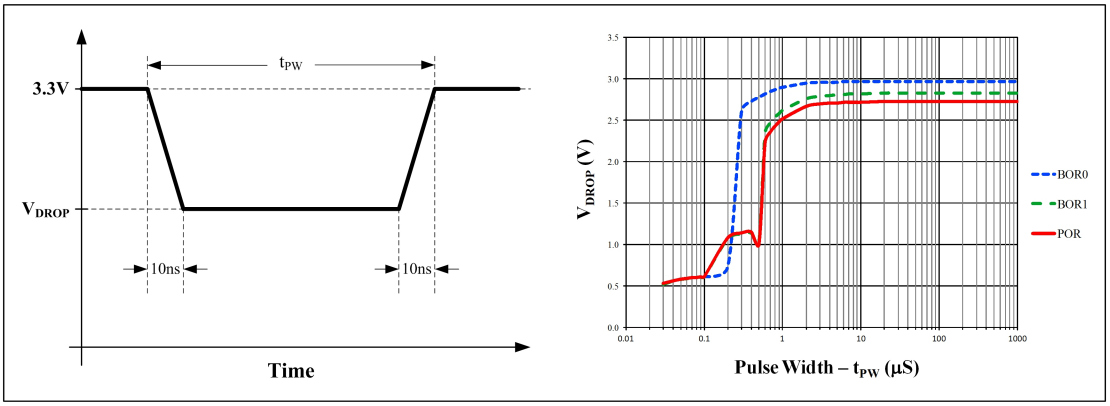
图 21-7. POR-BOR0-BOR1 VDD Glitch Response



21.6.5 VDD Droop Response

图21-8（1040页）shows the response of the BOR0, BOR1, and the POR monitors to a drop on the V_{DD} supply.

图 21-8. POR-BOR0-BOR1 VDD Droop Response



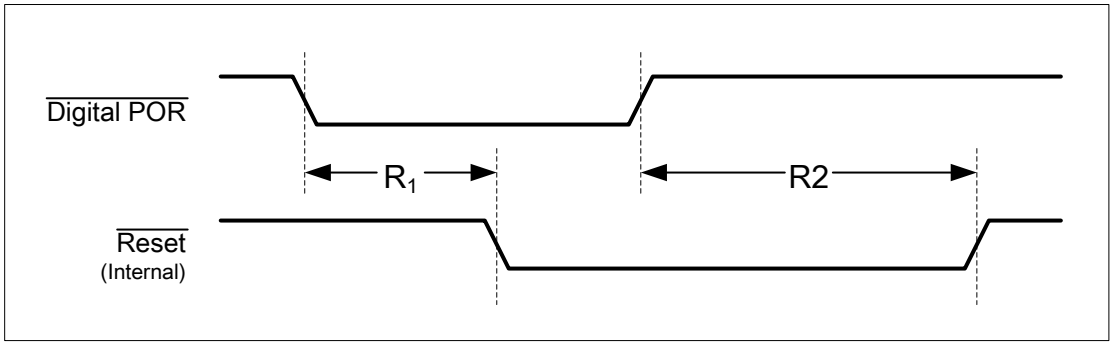
21.7 Reset

表 21-11. Reset Characteristics

Parameter No.	Parameter	Parameter Name	Min	Nom	Max	Unit
R1	$T_{DPORDLY}$	Digital POR to Internal Reset assertion delay[a]	0.80	-	5.35	µs
R2	T_{IRTOUT}	Internal Reset timeout	-	-	500	µs
R3	$T_{BOR0DLY}$	BOR0 to Internal Reset assertion delay[a]	0.25	-	1.95	µs
R3	$T_{BOR1DLY}$	BOR1 to Internal Reset assertion delay[a]	0.75	-	5.95	µs
R4	T_{RSTMIN}	Minimum $\overline{RST}$ pulse width	-	250	-	ns
R5	$T_{IRHWDLY}$	$\overline{RST}$ to Internal Reset assertion delay	-	250	-	ns
R6	T_{IRSWR}	Internal reset timeout after software-initiated system reset	-	2.07	-	µs
R7	T_{IRWDR}	Internal reset timeout after Watchdog reset	-	2.10	-	µs
R8	T_{IRMFR}	Internal reset timeout after MOSC failure reset	-	1.92	-	µs

a. Timing values are dependent on the V_{DD} power-down ramp rate.

图 21-9. Digital Power-On Reset Timing



注意: The digital Power-On Reset is only released when the analog Power-On Reset has de-asserted and all of the Power-OK monitors for each of the supplies indicate that power levels are in operational ranges.

图 21-10. Brown-Out Reset Timing

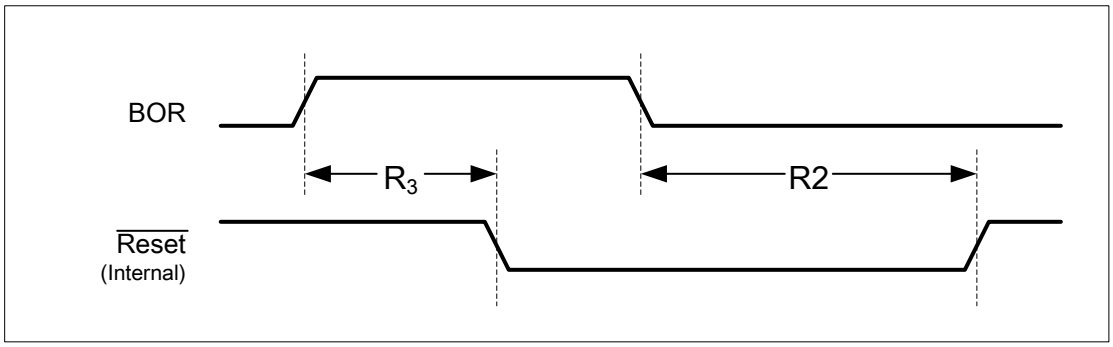


图 21-11. External Reset Timing ($\overline{RST}$)

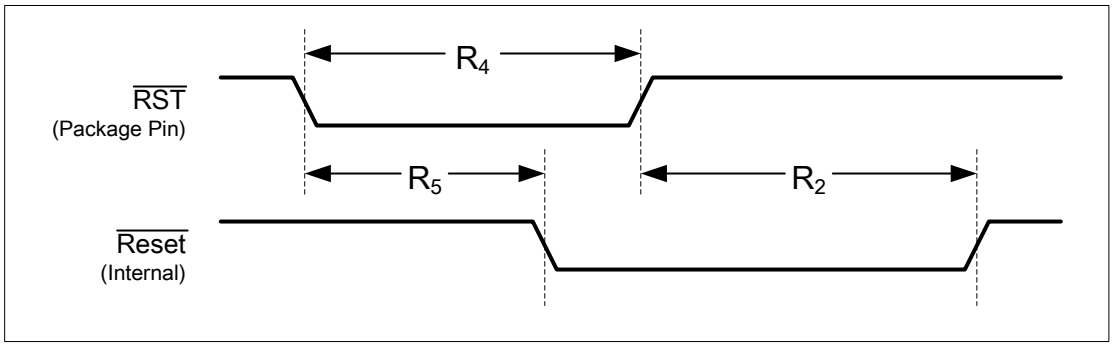


图 21-12. Software Reset Timing

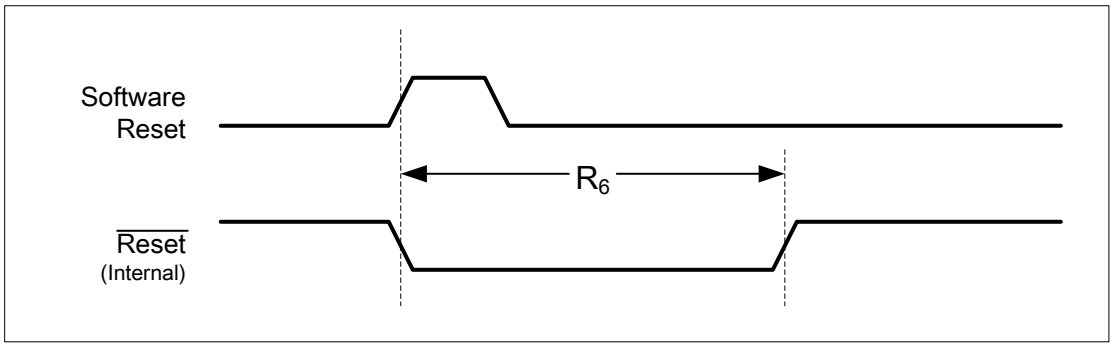


图 21-13. Watchdog Reset Timing

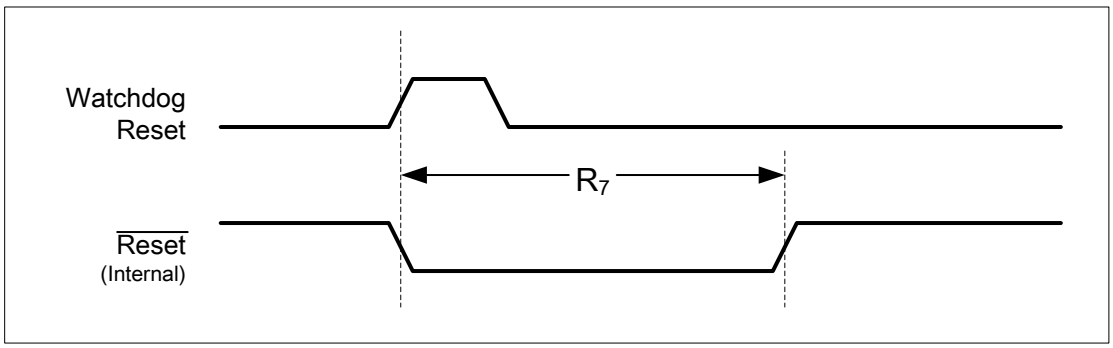
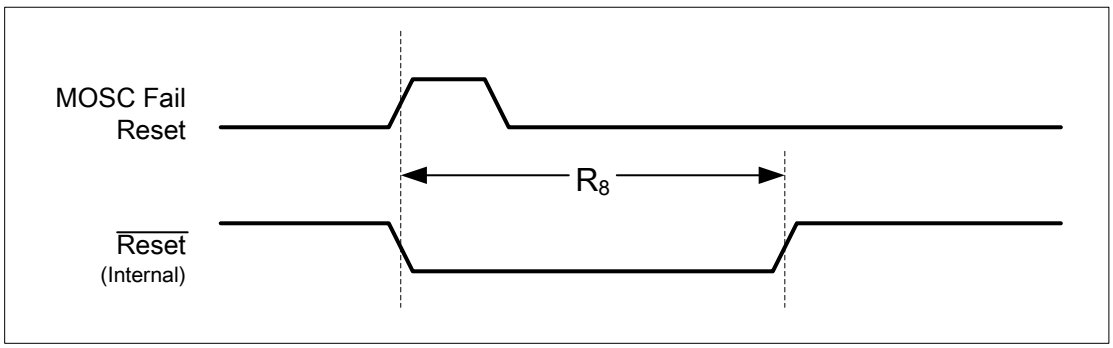


图 21-14. MOSC Failure Reset Timing



21.8 On-Chip Low Drop-Out (LDO) Regulator

表 21-12. LDO Regulator Characteristics

Parameter	Parameter Name	Min	Nom	Max	Unit
C_{LDO}	External filter capacitor size for internal power supply[a]	2.5	-	4.0	µF
ESR	Filter capacitor equivalent series resistance	10	-	100	mΩ
ESL	Filter capacitor equivalent series inductance	-	-	0.5	nH
V_{LDO}	LDO output voltage	1.08	1.2	1.32	V
I_{INRUSH}	Inrush current	50	-	250	mA

a. The capacitor should be connected as close as possible to pin 56.

21.9 Clocks

The following sections provide specifications on the various clock sources and mode.

21.9.1 PLL Specifications

The following tables provide specifications for using the PLL.

表 21-13. Phase Locked Loop (PLL) Characteristics

Parameter	Parameter Name	Min	Nom	Max	Unit
F_{REF_XTAL}	Crystal reference	5[a]	-	25	MHz
F_{REF_EXT}	External clock reference[a]	5[a]	-	25	MHz
F_{PLL}	PLL frequency[b]	-	400	-	MHz
T_{READY}	PLL lock time, enabling the PLL	-	-	512 * (N+1)[c]	reference clocks[d]
	PLL lock time, changing the XTAL field in the RCC/RCC2 register or changing the OSCSRC between MOSC and PIOSC	-	-	128 * (N+1)[c]	reference clocks[d]

a. If the PLL is not used, the minimum input frequency can be 4 MHz.

b. PLL frequency is automatically calculated by the hardware based on the XTAL field of the RCC register. The PLL frequency that is set by the hardware can be calculated using the values in the PLLFREQ0 and PLLFREQ1 registers.

c. N is the value in the N field in the PLLFREQ1 register.

d. A reference clock is the clock period of the crystal being used, which can be MOSC or PIOSC. For example, a 16-MHz crystal connected to MOSC yields a reference clock of 62.5 ns.

表21-14（1044页）shows the actual frequency of the PLL based on the crystal frequency used (defined by the XTAL field in the RCC register).

表 21-14. Actual PLL Frequency

XTAL	Crystal Frequency (MHz)	MINT	MFRAC	Q	N	PLL Multiplier	PLL Frequency (MHz)	Error
0x09	5.0	0x50	0x0	0x0	0x0	80	400	-
0x0A	5.12	0x9C	0x100	0x0	0x1	156.25	400	-
0x0B	6.0	0xC8	0x0	0x0	0x2	200	400	-
0x0C	6.144	0xC3	0x140	0x0	0x2	195.3125	400	-
0x0D	7.3728	0xA2	0x30A	0x0	0x2	162.7598	399.9984	0.0004%
0x0E	8.0	0x32	0x0	0x0	0x0	50	400	-
0x0F	8.192	0xC3	0x140	0x0	0x3	195.3125	400	-
0x10	10.0	0x50	0x0	0x0	0x1	80	400	-
0x11	12.0	0xC8	0x0	0x0	0x5	200	400	-
0x12	12.288	0xC3	0x140	0x0	0x5	195.3125	400	-
0x13	13.56	0xB0	0x3F6	0x0	0x5	176.9902	399.9979	0.0005%
0x14	14.318	0xC3	0x238	0x0	0x6	195.5547	399.9982	0.0005%
0x15	16.0	0x32	0x0	0x0	0x1	50	400	-
0x16	16.384	0xC3	0x140	0x0	0x7	195.3125	400	-
0x17	18	0xC8	0x0	0x0	0x8	200	400	-
0x18	20	0x50	0x0	0x0	0x3	80	400	-
0x19	24	0x32	0x0	0x0	0x2	50	400	-

表 21-14. Actual PLL Frequency（续）

XTAL	Crystal Frequency (MHz)	MINT	MFRAC	Q	N	PLL Multiplier	PLL Frequency (MHz)	Error
0x1A	25	0x50	0x0	0x0	0x4	80	400	-

21.9.2 PIOSC Specifications

表 21-15. PIOSC Clock Characteristics

Parameter	Parameter Name	Min	Nom	Max	Unit
F_{PIOS}[a]	Internal 16-MHz precision oscillator frequency variance (factory calibrated at 25 °C C and 3.3 V) across the specified voltage and temperature range	-	-	±3%	-

a. This parameter value remains valid if recalibration occurs.

21.9.3 Low-Frequency Internal Oscillator (LFIOSC) Specifications

表 21-16. Low-Frequency internal Oscillator Characteristics

Parameter	Parameter Name	Min	Nom	Max	Unit
F_{LFIOSC}	Low-frequency internal oscillator (LFIOSC) frequency	10	33	90	KHz

21.9.4 Hibernation Clock Source Specifications

表 21-17. Hibernation Oscillator Input Characteristics

Parameter	Parameter Name	Min	Nom	Max	Unit
$F_{HIBLFIOSC}$	Hibernation low frequency internal oscillator (HIB LFIOSC) frequency	10	33	90	KHz
C_1, C_2	External load capacitance on XOSC0, XOSC1 pins[a]	12	-	24	pF
C_{PKG}	Device package stray shunt capacitance[a]	-	0.5	-	pF
C_{PCB}	PCB stray shunt capacitance[a]	-	0.5	-	pF
C_0	Crystal shunt capacitance[a]	-	3	-	pF
C_{SHUNT}	Total shunt capacitance[a]	-	-	4	pF
ESR	Crystal effective series resistance, OSCDRV = 0[b]	-	-	50	kΩ
	Crystal effective series resistance, OSCDRV = 1[b]	-	-	75	kΩ
DL	Oscillator output drive level	-	-	0.25	µW
T_{START}	Oscillator startup time, when using a crystal[c]	-	600	1500[d]	ms
V_{IH}[e]	CMOS input high level, when using an external oscillator with Supply > 3.3 V	2.64	-	-	V
	CMOS input high level, when using an external oscillator with 1.8 V ≤ Supply ≤ 3.3 V	0.8 * Supply	-	-	V
V_{IL}[e]	CMOS input low level, when using an external oscillator with 1.8 V ≤ Supply ≤ 3.63 V	-	-	0.2 * Supply	V
V_{HYS}[e]	CMOS input buffer hysteresis, when using an external oscillator with 1.8 V ≤ Supply ≤ 3.63 V	360	960	1390	mV
DC_{HIBOSC_EXT}	External clock reference duty cycle	30	-	70	%

a. See information below table.

b. Crystal ESR specified by crystal manufacturer.

c. Oscillator startup time is specified from the time the oscillator is enabled to when it reaches a stable point of oscillation such that the internal clock is valid.

d. Only valid for recommended supply conditions. Measured with OSCDRV bit set (high drive strength enabled, 24 pF).

e. Specification is relative to the larger of V_{DD} or V_{BAT}.

The load capacitors added on the board, C_1 and C_2, should be chosen such that the following equation is satisfied (see 表21-17（1045页） for typical values).

- C_L = load capacitance specified by crystal manufacturer
- $C_L = (C_1*C_2)/(C_1+C_2) + C_{PKG} + C_{PCB}$
- $C_{SHUNT} = C_{PKG} + C_{PCB} + C_0$ (total shunt capacitance seen across XOSC0, XOSC1)
- C_{PKG}, C_{PCB} as measured across the XOSC0, XOSC1 pins excluding the crystal
- Clear the OSCDRV bit in the Hibernation Control (HIBCTL) register for $C_{1,2} \leq 18$ pF; set the OSCDRV bit for $C_{1,2} > 18$ pF.
- C_0 = Shunt capacitance of crystal specified by the crystal manufacturer

21.9.5 Main Oscillator Specifications

表 21-18. Main Oscillator Input Characteristics

Parameter	Parameter Name	Min	Nom	Max	Unit
F_{MOSC}	Parallel resonance frequency	4[a]	-	25	MHz
C_1, C_2	External load capacitance on OSC0, OSC1 pins[b]	12	-	24	pF
C_{PKG}	Device package stray shunt capacitance[b]	-	0.5	-	pF
C_{PCB}	PCB stray shunt capacitance[b]	-	0.5	-	pF
C_0	Crystal shunt capacitance[bc]	-	4	-	pF
C_{SHUNT}	Total shunt capacitance[b]	-	-	4	pF
ESR	Crystal effective series resistance, 4 MHz[dc]	-	-	300	Ω
	Crystal effective series resistance, 6 MHz[dc]	-	-	200	Ω
	Crystal effective series resistance, 8 MHz[dc]	-	-	130	Ω
	Crystal effective series resistance, 12 MHz[dc]	-	-	120	Ω
	Crystal effective series resistance, 16 MHz[dc]	-	-	100	Ω
	Crystal effective series resistance, 25 MHz[dc]	-	-	50	Ω
DL	Oscillator output drive level[e]	-	OSC_{PWR}	-	mW
T_{START}	Oscillator startup time, when using a crystal[f]	-	-	18	ms
V_{IH}	CMOS input high level, when using an external oscillator	0.65 * V_{DD}	-	V_{DD}	V
V_{IL}	CMOS input low level, when using an external oscillator	GND	-	0.35 * V_{DD}	V
V_{HYS}	CMOS input buffer hysteresis, when using an external oscillator	150	-	-	mV
DC_{OSC_EXT}	External clock reference duty cycle	45	-	55	%

a. 5 MHz is the minimum when using the PLL.

b. See information below table.

c. Crystal vendors can be contacted to confirm these specifications are met for a specific crystal part number if the vendors generic crystal datasheet show limits outside of these specifications.

d. Crystal ESR specified by crystal manufacturer.

e. $OSC_{PWR} = (2 * pi * F_P * C_L * 2.5)^2 * ESR / 2$. An estimation of the typical power delivered to the crystal is based on the C_L, F_P and ESR parameters of the crystal in the circuit as calculated by the OSC_{PWR} equation. Ensure that the value calculated for OSC_{PWR} does not exceed the crystal's drive-level maximum.

f. Oscillator startup time is specified from the time the oscillator is enabled to when it reaches a stable point of oscillation such that the internal clock is valid.

The load capacitors added on the board, C_1 and C_2, should be chosen such that the following equation is satisfied (see 表21-18（1046页） for typical values and 表21-19（1047页） for detailed crystal parameter information).

- C_L = load capacitance specified by crystal manufacturer
- $C_L = (C_1*C_2)/(C_1+C_2) + C_{SHUNT}$
- $C_{SHUNT} = C_0 + C_{PKG} + C_{PCB}$ (total shunt capacitance seen across OSC0, OSC1 crystal inputs)
- C_{PKG}, C_{PCB} = the mutual caps as measured across the OSC0,OSC1 pins excluding the crystal.
- C_0 = Shunt capacitance of crystal specified by the crystal manufacturer

表21-19（1047页） lists part numbers of crystals that have been simulated and confirmed to operate within the specifications in 表21-18（1046页）. Other crystals that have nearly identical crystal parameters can be expected to work as well.

In the table below, the crystal parameters labeled C0, C1 and L1 are values that are obtained from the crystal manufacturer. These numbers are usually a result of testing a relevant batch of crystals on a network analyzer. The parameters labeled ESR, DL and C_L are maximum numbers usually available in the data sheet for a crystal.

The table also includes three columns of Recommended Component Values. These values apply to system board components. C_1 and C_2 are the values in pico Farads of the load capacitors that should be put on each leg of the crystal pins to ensure oscillation at the correct frequency. Rs is the value in kΩ of a resistor that is placed in series with the crystal between the OSC1 pin and the crystal pin. Rs dissipates some of the power so the Max DI crystal parameter is not exceeded. Only use the recommended C_1, C_2, and Rs values with the associated crystal part. The values in the table were used in the simulation to ensure crystal startup and to determine the worst case drive level (WC DI). The value in the WC DI column should not be greater than the Max DI Crystal parameter. The WC DI value can be used to determine if a crystal with similar parameter values but a lower Max DI value is acceptable.

表 21-19. Crystal Parameters

MFG	MFG Part#	Holder	PKG Size (mm x mm)	Freq (MHz)	Crystal Spec	Crystal Parameters						Recommended Component Values			WC DI (µW)
						Typical Values			Max Values						
						C0 (pF)	C1 (fF)	L1 (mH)	ESR (Ω)	Max DI (µW)	C_L (pf)	C_1 (pF)	C_2 (pF)	Rs (kΩ)	
NDK	NX8045GB-4.000M-STD-CJL-5	NX8045GB	8 x 4.5	4	STD-CJL-5	1.00	2.70	598.10	300	500	8	12	12	0	132
FOX	FQ1045A-4	2-SMD	10 x 4.5	4	FQ1045A	1.18	4.05	396.00	150	500	10	14	14	0	103

表 **21-19. Crystal Parameters**（续）

MFG	MFG Part#	Holder	PKG Size (mm x mm)	Freq (MHz)	Crystal Spec	Crystal Parameters						Recommended Component Values			WC DI (µW)
						Typical Values			Max Values						
						C0 (pF)	C1 (fF)	L1 (mH)	ESR (Ω)	Max DI (µW)	C_L (pf)	C_1 (pF)	C_2 (pF)	Rs (kΩ)	
NDK	NX8045GB-5.000M-STD-CSF-4	NX8045GB	8 x 4.5	5	STD-CJL-4	1.00	2.80	356.50	250	500	8	12	12	0	164
NDK	NX8045GB-6.000M-STD-CSF-4	NX8045GB	8 x 4.5	6	STD-CJL-4	1.30	4.10	173.20	250	500	8	12	12	0	214
FOX	FQ1045A-6	2-SMD	10 x 4.5	6	FQ1045A	1.37	6.26	112.30	150	500	10	14	14	0	209
NDK	NX8045GB-8.000M-STD-CSF-6	NX8045GB	8 x 4.5	8	STD-CSF-6	1.00	2.80	139.30	200	500	8	12	12	0	277
FOX	FQ7050B-8	4-SMD	7 x 5	8	FQ7050	1.95	6.69	59.10	80	500	10	14	14	0	217
ECS	ECS-80-16-28A-TR	HC49/US	12.5 x 4.85	8	CSM-4A	1.82	4.90	85.70	80	500	16	24	24	0	298
NDK	NX3225GA-12.000MHZ-STD-CRG-2	NX3225GA	3.2 x 2.5	12	STD-CRG-2	0.70	2.20	81.00	100	200	8	12	12	2.5	147
NDK	NX5032GA-12.000MHZ-LN-CD-1	NX5032GA	5 x 3.2	12	LN-CD-1	0.93	3.12	56.40	120	500	8	12	12	0	362
FOX	FQ5032B-12	4-SMD	5 x 3.2	12	FQ5032	1.16	4.16	42.30	80	500	10	14	14	0	370
NDK	NX3225GA-16.000MHZ-STD-CRG-2	NX3225GA	3.2 x 2.5	16	STD-CRG-2	1.00	2.90	33.90	80	200	8	12	12	2	188
NDK	NX5032GA-16.000MHZ-LN-CD-1	NX5032GA	5 x 3.2	16	LN-CD-1	1.02	3.82	25.90	120	500	8	10	10	0	437
ECS	ECS-160-9-42-CKM-TR	ECX-42	4 x 2.5	16	ECX-42	1.47	3.90	25.84	60	300	9	12	12	0.5	289
NDK	NX3225GA-25.000MHZ-STD-CRG-2	NX3225GA	3.2 x 2.5	25	STD-CRG-2	1.10	4.70	8.70	50	200	8	12	12	2	181
AURIS	Q-25.000M-HC3225/4-F-30-30-E-12-TR	HC3225/4	3.2 x 2.5	25	HC3225	1.58	5.01	8.34	50	500	12	16	16	1	331
FOX	FQ5032B-25	4-SMD	5 x 3.2	25	FQ5032	1.69	7.92	5.13	50	500	10	14	14	0.5	433

表 **21-20. Supported MOSC Crystal Frequencies**

Value	Crystal Frequency (MHz) Not Using the PLL	Crystal Frequency (MHz) Using the PLL
0x00-0x5	reserved	

表 21-20. Supported MOSC Crystal Frequencies（续）

Value	Crystal Frequency (MHz) Not Using the PLL	Crystal Frequency (MHz) Using the PLL
0x06	4 MHz	reserved
0x07	4.096 MHz	reserved
0x08	4.9152 MHz	reserved
0x09	5 MHz	
0x0A	5.12 MHz	
0x0B	6 MHz	
0x0C	6.144 MHz	
0x0D	7.3728 MHz	
0x0E	8 MHz	
0x0F	8.192 MHz	
0x10	10.0 MHz	
0x11	12.0 MHz	
0x12	12.288 MHz	
0x13	13.56 MHz	
0x14	14.31818 MHz	
0x15	16.0 MHz (reset value)	
0x16	16.384 MHz	
0x17	18.0 MHz	
0x18	20.0 MHz	
0x19	24.0 MHz	
0x1A	25.0 MHz	

21.9.6 System Clock Specification with ADC Operation

表 21-21. System Clock Characteristics with ADC Operation

Parameter	Parameter Name	Min	Nom	Max	Unit
F_{sysadc}	System clock frequency when the ADC module is operating (when PLL is bypassed).[a]	15.9952	16	16.0048	MHz

a. Clock frequency (plus jitter) must be stable inside specified range. ADC can be clocked from the PLL, directly from an external clock source, or from the PIOSC, as long as frequency absolute precision is inside specified range.

21.10 Sleep Modes

表 21-22. Sleep Modes AC Characteristics[a]

Parameter No	Parameter	Parameter Name	Min	Nom	Max	Unit
D1	T_{WAKE_S}	Time to wake from interrupt in sleep mode[b]	-	-	2	system clocks
	T_{WAKE_DS}	Time to wake from interrupt in deep-sleep mode, using PIOSC for both Run mode and Deep-sleep mode[b c]	-	1.25	-	µs
		Time to wake from interrupt in deep-sleep mode, using PIOSC for Run mode and LFIOSC for Deep-sleep mode[b c]	-	350	-	µs
D2	$T_{WAKE_PLL_DS}$	Time to wake from interrupt in deep-sleep mode when using the PLL[b]	-	-	T_{READY}	ms

a. Values in this table assume the LFIOSC is the clock source during sleep or deep-sleep mode.

b. Specified from registering the interrupt to first instruction.

c. If the main oscillator is used for run mode, add the main oscillator startup time, T_{START}.

表 21-23. Time to Wake with Respect to Low-Power Modes[ab]

Mode	Run Mode Clock/Frequency	Sleep/Deep-Sleep Mode Clock/Frequency	FLASHPM	SRAMPM	Time to Wake		Unit
					Min	Max	
Sleep	MOSC, PLL on - 80MHz	MOSC, PLL on - 80MHz	0x0	0x0	0.28	0.30	µs
				0x1	33.57	35.00	µs
				0x3	33.75	35.05	µs
			0x2	0x0	105.02	109.23	µs
				0x1	137.85	143.93	µs
				0x3	138.06	143.86	µs

表 21-23. Time to Wake with Respect to Low-Power Modes（续）

Mode	Run Mode Clock/Frequency	Sleep/Deep-Sleep Mode Clock/Frequency	FLASHPM	SRAMPM	Time to Wake		Unit
					Min	Max	
Deep-Sleep	MOSC, PLL on - 80MHz	PIOSC - 16MHz	0x0	0x0	2.47	2.60	µs
				0x1	35.31	36.35	µs
				0x3	35.40	36.76	µs
			0x2	0x0	107.05	111.54	µs
				0x1	139.34	145.64	µs
				0x3	140.41	145.53	µs
	PIOSC - 16MHz	PIOSC - 16MHz	0x0	0x0	2.47	2.61	µs
				0x1	35.25	36.65	µs
				0x3	35.38	36.79	µs
			0x2	0x0	107.43	111.52	µs
				0x1	139.83	145.85	µs
				0x3	139.35	145.54	µs
	PIOSC - 16MHz	LFIOSC, PIOSC off[c] - 30kHz	0x0	0x0	415.06	728.38	µs
				0x1	436.60	740.88	µs
				0x3	433.80	755.32	µs
			0x2	0x0	503.73	812.82	µs
				0x1	537.72	846.23	µs
				0x3	536.10	839.25	µs
	MOSC, PLL on - 80MHz	LFIOSC, PIOSC off[c] - 30kHz	0x0	0x0	18.95	19.55	ms
				0x1	18.94	19.54	ms
				0x3	18.95	19.53	ms
			0x2	0x0	18.95	19.54	ms
				0x1	18.94	19.53	ms
				0x3	18.95	19.54	ms

a. Time from wake event to first instruction of code execution.

b. If the LDO voltage is adjusted, it will take an extra 4 us to wake up from Sleep or Deep-sleep mode.

c. PIOSC is turned off by setting the PIOSCPD bit in the DSLPCLKCFG register.

21.11 Hibernation Module

The Hibernation module requires special system implementation considerations because it is intended to power down all other sections of its host device, refer to “休眠模块”（429页）.

表 21-24. Hibernation Module Battery Characteristics

Parameter	Parameter Name	Min	Nominal	Max	Unit
V_{BAT}	Battery supply voltage	1.8	3.0	3.6[a]	V
V_{BATRMP}[b]	V_{BAT} battery supply voltage ramp time	0	-	0.7	V/µs
V_{LOWBAT}	Low battery detect voltage, VBATSEL=0x0	1.8	1.9	2.0	V
	Low battery detect voltage, VBATSEL=0x1	2.0	2.1	2.2	V
	Low battery detect voltage, VBATSEL=0x2	2.2	2.3	2.4	V
	Low battery detect voltage, VBATSEL=0x3	2.4	2.5	2.6	V

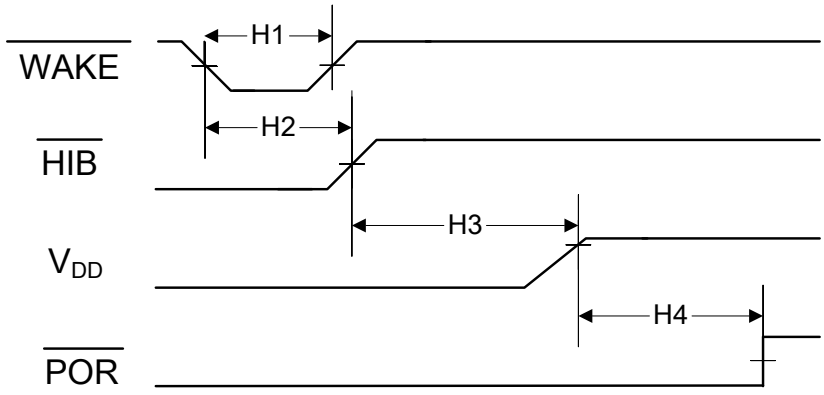
a. To ensure proper functionality, any voltage input within the range of 3.6 V < VBAT ≤ 4 V must be connected through a diode.

b. For recommended V_{BAT} RC circuit values, refer to the diagrams located in“休眠时钟源”（431页）.

表 21-25. Hibernation Module AC Characteristics

Parameter No	Parameter	Parameter Name	Min	Nom	Max	Unit
H1	T_{WAKE}	WAKE assertion time	100	-	-	ns
H2	$T_{WAKE_TO_HIB}$	WAKE assert to HIB desassert (wake up time)	-	-	1	hibernation clock period
H3	T_{VDD_RAMP}	V_{DD} ramp to 3.0 V	-	Depends on characteristics of power supply	-	µs
H4	T_{VDD_CODE}	V_{DD} at 3.0 V to internal POR deassert; first instruction executes	-	-	500	µs

图 21-15. Hibernation Module Timing



21.12 Flash Memory and EEPROM

表 21-26. Flash Memory Characteristics

Parameter	Parameter Name	Min	Nom	Max	Unit
PE_{CYC}	Number of program/erase cycles before failure[a]	100,000	-	-	cycles
T_{RET}	Data retention, -40˚C to +85˚C	20	-	-	years
T_{PROG64}	Program time for double-word-aligned 64 bits of data[b]	30	50	300	µs
T_{ERASE}	Page erase time, <1k cycles	-	8	15	ms
	Page erase time, 10k cycles	-	15	40	ms
	Page erase time, 100k cycles	-	75	500	ms
T_{ME}	Mass erase time, <1k cycles	-	10	25	ms
	Mass erase time, 10k cycles	-	20	70	ms
	Mass erase time, 100k cycles	-	300	2500	ms

a. A program/erase cycle is defined as switching the bits from 1-> 0 -> 1.

b. If programming fewer than 64 bits of data, the programming time is the same. For example, if only 32 bits of data need to be programmed, the other 32 bits are masked off.

表 21-27. EEPROM Characteristics[a]

Parameter	Parameter Name	Min	Nom	Max	Unit
EPE_{CYC}[b]	Number of mass program/erase cycles of a single word before failure[c]	500,000	-	-	cycles
ET_{RET}	Data retention, -40˚C to +85˚C	20	-	-	years
ET_{PROG}	Program time for 32 bits of data - space available	-	110	600	µs
	Program time for 32 bits of data - requires a copy to the copy buffer, copy buffer has space and less than 10% of EEPROM endurance used	-	30	-	ms
	Program time for 32 bits of data - requires a copy to the copy buffer, copy buffer has space and greater than 90% of EEPROM endurance used	-	-	900	ms
	Program time for 32 bits of data - requires a copy to the copy buffer, copy buffer requires an erase and less than 10% of EEPROM endurance used	-	60	-	ms
	Program time for 32 bits of data - requires a copy to the copy buffer, copy buffer requires an erase and greater than 90% of EEPROM endurance used	-	-	1800	ms
ET_{READ}	Read access time	-	4	-	system clocks
ET_{ME}	Mass erase time, <1k cycles	-	8	15	ms
	Mass erase time, 10k cycles	-	15	40	ms
	Mass erase time, 100k cycles	-	75	500	ms

a. Because the EEPROM operates as a background task and does not prevent the CPU from executing from Flash memory, the operation will complete within the maximum time specified provided the EEPROM operation is not stalled by a Flash memory program or erase operation.

b. One word can be written more than 500K times, but these writes impact the endurance of the words in the meta-block that the word is within. Different words can be written such that any or all words can be written more than 500K times when write counts per word stay about the same. See “耐久性”一节（470页） for more information.

c. A program/erase cycle is defined as switching the bits from 1-> 0 -> 1.

21.13 Input/Output Pin Characteristics

21.13.1 GPIO Module Characteristics

注意: All GPIO signals are 5-V tolerant when configured as inputs except for PD4, PD5, PB0 and PB1, which are limited to 3.6 V. See “信号描述”（575页） for more information on GPIO configuration.

注意: GPIO pads are tolerant to 5-V digital inputs without creating reliability issues, as long as the supply voltage, VDD, is present. There are limitations to how long a 5-V input can be present on any given I/O pad if VDD is not present. Not meeting these conditions will affect reliability of the device and affect the GPIO characteristics specifications.

- If the voltage applied to a GPIO pad is in the high voltage range (5V +/- 10%) while VDD is not present, such condition should be allowed for a maximum of 10,000 hours at 27°C or 5,000 hours at 85°C, over the lifetime of the device.
- If the voltage applied to a GPIO pad is in the normal voltage range (3.3V +/- 10%) while VDD is not present or if the voltage applied is in the high voltage range (5V +/- 10%) while VDD is present, there are no constraints on the lifetime of the device.

表 21-28. GPIO Module Characteristics[a]

Parameter	Parameter Name	Min	Nom	Max	Unit
R_{GPIOPU}	GPIO internal pull-up resistor	13	20	30	kΩ
R_{GPIOPD}	GPIO internal pull-down resistor	13	20	35	kΩ
I_{LKG+}	GPIO input leakage current, 0 V ≤ V_{IN} ≤ V_{DD} GPIO pins[b]	-	-	1.0	µA
	GPIO input leakage current, 0 V < V_{IN} ≤ V_{DD}, GPIO pins configured as ADC or analog comparator inputs	-	-	2.0	µA
T_{GPIOR}	GPIO rise time, 2-mA drive[c]	-	14.2	16.1	ns
	GPIO rise time, 4-mA drive[c]		11.9	15.5	ns
	GPIO rise time, 8-mA drive[c]		8.1	11.2	ns
	GPIO rise time, 8-mA drive with slew rate control[c]		9.5	11.8	ns
T_{GPIOF}	GPIO fall time, 2-mA drive[d]	-	25.2	29.4	ns
	GPIO fall time, 4-mA drive[d]		13.3	16.8	ns
	GPIO fall time, 8-mA drive[d]		8.6	11.2	ns
	GPIO fall time, 8-mA drive with slew rate control[d]		11.3	12.9	ns

a. V_{DD} must be within the range specified in 表21-5（1031页）.

b. The leakage current is measured with V_{IN} applied to the corresponding pin(s). The leakage of digital port pins is measured individually. The port pin is configured as an input and the pullup/pulldown resistor is disabled.

c. Time measured from 20% to 80% of V_{DD}.

d. Time measured from 80% to 20% of V_{DD}.

21.13.2 Types of I/O Pins and ESD Protection

With respect to ESD and leakage current, three types of I/O pins exist on the device: Power I/O pins, I/O pins with fail-safe ESD protection (GPIOs other than PD4 and PD5 , and XOSCn pins) and I/O pins with non-fail-safe ESD protection (any non-power, non-GPIO (other than PD4 and PD5) and non-XOSCn pins). This section covers I/O pins with fail-safe ESD protection and I/O pins with

non-fail-safe ESD protection. Power I/O pin voltage and current limitations are specified in “Recommended Operating Conditions”（1031页）.

21.13.2.1 Fail-Safe Pins

GPIOs other than PD4 and PD5, inputs for the Hibernate 32-kHz oscillator (XOSCn), and Hibernate inputs use ESD protection as shown in 图21-16（1055页）.

An unpowered device cannot be parasitically powered through any of these pins. This ESD protection prevents a direct path between these I/O pads and any power supply rails in the device. GPIO/XOSCn pad input voltages should be kept inside the maximum ratings specified in 表21-1（1029页） to ensure current leakage and current injections are within acceptable range. Current leakages and current injection for these pins are specified in 表21-28（1054页）.

图21-16（1055页） shows a diagram of the ESD protection on fail-safe pins.

Some GPIOs when configured as inputs require a strong pull-up resistor to maintain a threshold above the minimum value of VIH during power-on. See 表21-30（1056页）.

图 21-16. ESD Protection on Fail-Safe Pins

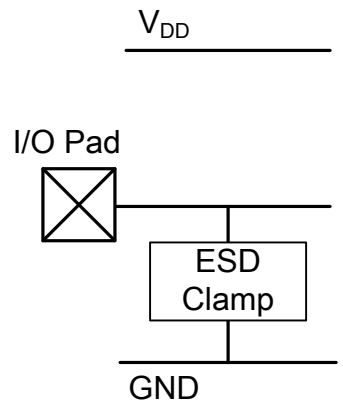


表 21-29. Pad Voltage/Current Characteristics for Fail-Safe Pins[a]

Parameter	Parameter Name	Min	Nom	Max	Unit
I_{LKG+}	GPIO input leakage current, $V_{DD} < V_{IN} \leq 4.5$ V[bb]	-	-	700	µA
	GPIO input leakage current, $4.5 V < V_{IN} \leq 5.5$ V[bc]	-	-	100	µA
I_{LKG-}	GPIO input leakage current, $V_{IN} < -0.3$ V[bd]	-	-	-[e]	µA
	GPIO input leakage current, $-0.3 V \leq V_{IN} < 0$ V[b]	-	-	10	µA
I_{INJ+}	DC injection current, $V_{DD} < V_{IN} \leq 5.5$ V[fg]	-	-	I_{LKG+}	µA
I_{INJ-}	DC injection current, $V_{IN} \leq 0$ V[g]	-	-	0.5	mA

a. VIN must be within the range specified in 表21-1（1029页）.

b. To protect internal circuitry from over-voltage, the GPIOs have an internal voltage clamp that limits internal swings to V_{DD} without affecting swing at the I/O pad. This internal clamp starts turning on while $V_{DD} < V_{IN} < 4.5$ V and causes a somewhat larger (but bounded) current draw. To save power, static input voltages between V_{DD} and 4.5 V should be avoided.

c. Leakage current above maximum voltage ($V_{IN} = 5.5$V) is not guaranteed, this condition is not allowed and can result in permanent damage to the device.

d. Leakage outside the minimum range (-0.3V) is unbounded and must be limited to IINJ- using an external resistor.

e. In this case, I_{LKG-} is unbounded and must be limited to I_{INJ-} using an external resistor.

f. Current injection is internally bounded for GPIOs, and maximum current into the pin is given by ILKG+ for $V_{DD} < V_{IN} < 5.5$ V.

g. If the I/O pad is not voltage limited, it should be current limited (to IINJ+ and IINJ-) if there is any possibility of the pad voltage exceeding the VIO limits (including transient behavior during supply ramp up, or at any time when the part is unpowered).

表 21-30. Fail-Safe GPIOs that Require an External Pull-up

GPIO	Pin	Pull-Up Resistor Value	Unit
PB0	45	1k ≤ R ≤ 10k	Ω
PB1	46	1k ≤ R ≤ 10k	Ω
PE3	6	1k ≤ R ≤ 10k	Ω

21.13.2.2 Non-Fail-Safe Pins

The Main Oscillator (MOSC) crystal connection pins and GPIO pins PD4 and PD5 have ESD protection as shown in 图21-17（1056页）. These pins have a potential path between the I/O pad and an internal power rail if either one of the ESD diodes is accidentally forward biased. The voltage and current of these pins should follow the specifications in 表21-31（1056页） to prevent potential damage to the device. In addition to the specifications outlined in 表21-31（1056页）, it is recommended that the ADC external reference specifications in 表21-32（1058页） be adhered to in order to prevent any gain error.

图21-17（1056页） shows a diagram of the ESD protection on non-fail-safe pins.

图 21-17. ESD Protection on Non-Fail-Safe Pins

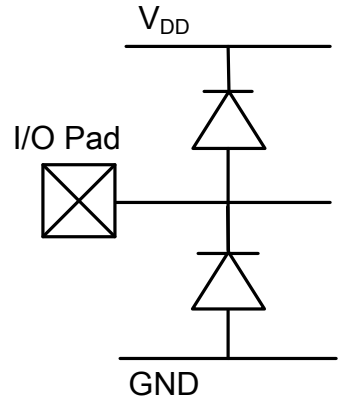


表 21-31. Non-Fail-Safe I/O Pad Voltage/Current Characteristics[abcd]

Parameter	Parameter Name	Min	Nom	Max	Unit
V_{IO}	IO pad voltage limits	-0.3	V_{DD}	V_{DD}+0.3	V
I_{LKG+}	Positive IO leakage for V_{IO} Max[ef]	-	-	10	µA
I_{LKG-}	Negative IO leakage for V_{IO} Min[ef]	-	-	10	µA
I_{INJ+}	Max positive injection[g]	-	-	2	mA
I_{INJ-}	Max negative injection if not voltage protected[g]	-	-	-0.5	mA

a. V_{IN} must be within the range specified in 表21-1（1029页）. Leakage current outside of this maximum voltage is not guaranteed and can result in permanent damage of the device.

b. VDD must be within the range specified in 表21-5（1031页）.

c. To avoid potential damage to the part, either the voltage or current on the ESD-protected, non-Power, non-Hibernate/XOSC input/outputs should be limited externally as shown in this table.

d. I/O pads should be protected if at any point the IO voltage has a possibility of going outside the limits shown in the table. If the part is unpowered, the IO pad Voltage or Current must be limited (as shown in this table) to avoid powering the part through the IO pad, causing potential irreversible damage.

e. This value applies to an I/O pin that is voltage-protected within the Min and Max V_{IO} ratings. Leakage outside the specified voltage range is unbounded and must be limited to I_{INJ-} using an external resistor.

f. MIN and MAX leakage current for the case when the I/O is voltage-protected to VIO Min or VIO Max.

g. If an I/O pin is not voltage-limited, it should be current-limited (to I_{INJ+} and I_{INJ-}) if there is any possibility of the pad voltage exceeding the V_{IO} limits (including transient behavior during supply ramp up, or at any time when the part is unpowered).

21.14 Analog-to-Digital Converter (ADC)

表 21-32. ADC Electrical Characteristics[ab]

Parameter	Parameter Name	Min	Nom	Max	Unit
POWER SUPPLY REQUIREMENTS					
V_{DDA}	ADC supply voltage	2.97	3.3	3.63	V
GNDA	ADC ground voltage	-	0	-	V
VDDA / GNDA VOLTAGE REFERENCE					
C_{REF}	Voltage reference decoupling capacitance	-	1.0 // 0.01[c]	-	µF
ANALOG INPUT					
V_{ADCIN}	Single-ended, full- scale analog input voltage, internal reference[de]	0	-	V_{DDA}	V
	Differential, full-scale analog input voltage, internal reference[df]	$-V_{DDA}$	-	V_{DDA}	V
VIN_{CM}	Input common mode voltage, differential mode[g]	-	-	(VREFP + VREFN) / 2 ± 25	mV
I_L	ADC input leakage current[h]	-	-	2.0	µA
R_{ADC}	ADC equivalent input resistance[h]	-	-	2.5	kΩ
C_{ADC}	ADC equivalent input capacitance[h]	-	-	10	pF
R_S	Analog source resistance[h]	-	-	500	Ω
SAMPLING DYNAMICS					
F_{ADC}	ADC conversion clock frequency[i]	-	16	-	MHz
F_{CONV}	ADC conversion rate		1		MSPS
T_S	ADC sample time	-	250	-	ns
T_C	ADC conversion time[j]		1		µs
T_{LT}	Latency from trigger to start of conversion	-	2	-	ADC clocks
SYSTEM PERFORMANCE when using internal reference					
N	Resolution		12		bits
INL	Integral nonlinearity error, over full input range	-	±1.5	±3.0	LSB
DNL	Differential nonlinearity error, over full input range	-	±0.8	+2.0/-1.0[k]	LSB
E_O	Offset error	-	±5.0	±15.0	LSB
E_G	Gain error[l]	-	±10.0	±30.0	LSB
E_T	Total unadjusted error, over full input range[m]	-	±10.0	±30.0	LSB
DYNAMIC CHARACTERISTICS[no]					
SNR_D	Signal-to-noise-ratio, Differential input, V_{ADCIN}: -20dB FS, 1KHz [p]	70	72	-	dB
SDR_D	Signal-to-distortion ratio, Differential input, V_{ADCIN}: -3dB FS, 1KHz[pqr]	72	75	-	dB
$SNDR_D$	Signal-to-Noise+Distortion ratio, Differential input, V_{ADCIN}: -3dB FS, 1KHz[pst]	68	70	-	dB
SNR_S	Signal-to-noise-ratio, Single-ended input, V_{ADCIN}: -20dB FS, 1KHz[u]	60	65	-	dB

表 21-32. ADC Electrical Characteristics（续）

Parameter	Parameter Name	Min	Nom	Max	Unit
SDR_S	Signal-to-distortion ratio, Single-ended input, V_{ADCIN}: -3dB FS, 1KHz[qr]	70	72	-	dB
$SNDR_S$	Signal-to-Noise+Distortion ratio, Single-ended input, V_{ADCIN}: -3dB FS, 1KHz[stu]	60	63	-	dB
TEMPERATURE SENSOR					
V_{TSENS}	Temperature sensor voltage, junction temperature 25 °C	-	1.633	-	V
S_{TSENS}	Temperature sensor slope	-	-13.3	-	mV/°C
E_{TSENS}	Temperature sensor accuracy[v]	-	-	±5	°C

a. V_{REF+}= 3.3V, F_{ADC}=16 MHz unless otherwise noted.

b. Best design practices suggest that static or quiet digital I/O signals be configured adjacent to sensitive analog inputs to reduce capacitive coupling and cross talk. Analog signals configured adjacent to ADC input channels should meet the same source resistance and bandwidth limitations that apply to the ADC input signals.

c. Two capacitors in parallel.

d. Internal reference is connected directly between V_{DDA} and GNDA (VREFi = V_{DDA} - GNDA). In this mode, E_O, E_G, E_T, and dynamic specifications are adversely affected due to internal voltage drop and noise on V_{DDA} and GNDA.

e. $V_{ADCIN} = V_{INP} - V_{INN}$

f. With signal common mode as $V_{DDA}/2$.

g. This parameter is defined as the average of the differential inputs.

h. As shown in 图21-18（1060页）, R_{ADC} is the total equivalent resistance in the input line all the way up to the sampling node at the input of the ADC.

i. See “System Clock Specification with ADC Operation”（1049页） for full ADC clock frequency specification.

j. ADC conversion time (Tc) includes the ADC sample time (Ts).

k. 12-bit DNL

l. Gain error is measured at max code after compensating for offset. Gain error is equivalent to "Full Scale Error." It can be given in % of slope error, or in LSB, as done here.

m. Total Unadjusted Error is the maximum error at any one code versus the ideal ADC curve. It includes all other errors (offset error, gain error and INL) at any given ADC code.

n. A low-noise environment is assumed in order to obtain values close to spec. The board must have good ground isolation between analog and digital grounds and a clean reference voltage. The input signal must be band-limited to Nyquist bandwidth. No anti-aliasing filter is provided internally.

o. ADC dynamic characteristics are measured using low-noise board design, with low-noise reference voltage (< -74dB noise level in signal BW) and low-noise analog supply voltage. Board noise and ground bouncing couple into the ADC and affect dynamic characteristics. Clean external reference must be used to achieve shown specs.

p. Differential signal with correct common mode, applied between two ADC inputs.

q. SDR = -THD in dB.

r. For higher frequency inputs, degradation in SDR should be expected.

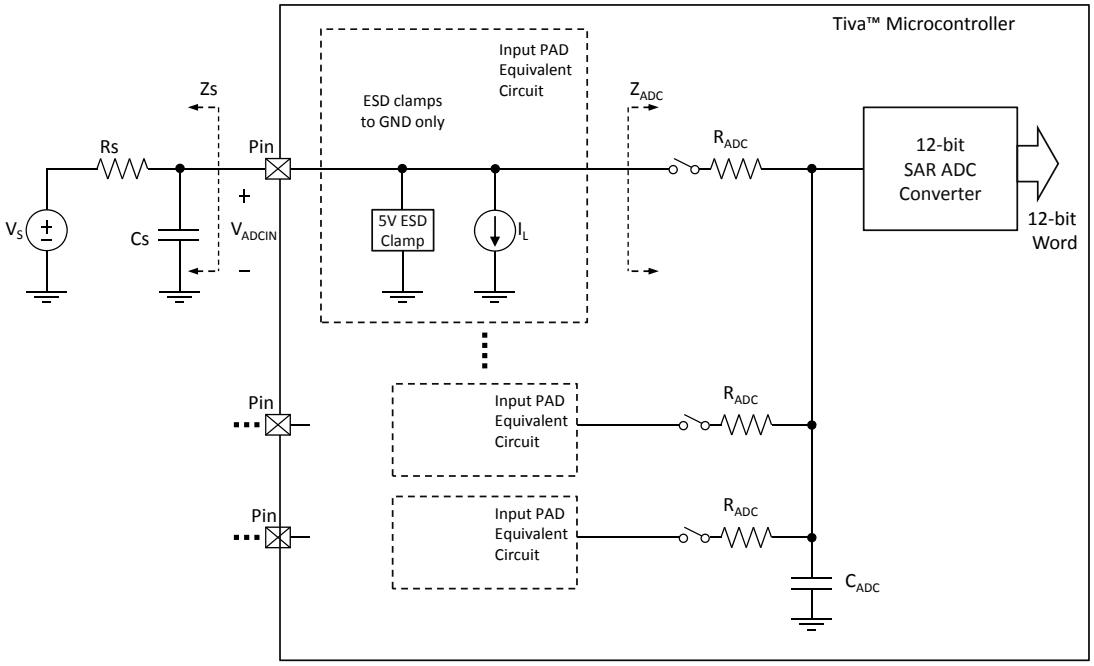
s. SNDR = S/(N+D) = SINAD (in dB)

t. Effective number of bits (ENOB) can be calculated from SNDR: ENOB = (SNDR - 1.76) / 6.02.

u. Single-ended inputs are more sensitive to board and trace noise than differential inputs; SNR and SNDR measurements on single-ended inputs are highly dependent on how clean the test set-up is. If the input signal is not well-isolated on the board, higher noise than specified could potentially be seen at the ADC output.

v. Note that this parameter does not include ADC error.

图 21-18. ADC Input Equivalency Diagram



21.15 Synchronous Serial Interface (SSI)

表 21-33. SSI Characteristics

Parameter No.	Parameter	Parameter Name	Min	Nom	Max	Unit
S1	T_{CLK_PER}	SSIClk cycle time, as master[a]	40	-	-	ns
		SSIClk cycle time, as slave[b]	150	-	-	ns
S2	T_{CLK_HIGH}	SSIClk high time, as master	20	-	-	ns
		SSIClk high time, as slave	75	-	-	ns
S3	T_{CLK_LOW}	SSIClk low time, as master	20	-	-	ns
		SSIClk low time, as slave	75	-	-	ns
S4	T_{CLKR}	SSIClk rise time[c]	1.25	-	-	ns
S5	T_{CLKF}	SSIClk fall time[c]	1.25	-	-	ns
S6	T_{TXDMOV}	Master Mode: Master Tx Data Output (to slave) Valid Time from edge of SSIClk	-	-	15.7	ns
S7	T_{TXDMOH}	Master Mode: Master Tx Data Output (to slave) Hold Time from next SSIClk	0.31	-	-	ns
S8	T_{RXDMS}	Master Mode: Master Rx Data In (from slave) setup time	17.15	-	-	ns
S9	T_{RXDMH}	Master Mode: Master Rx Data In (from slave) hold time	0	-	-	ns
S10	T_{TXDSOV}	Slave Mode: Master Tx Data Output (to Master) Valid Time from edge of SSIClk	-	-	77.74[d]	ns
S11	T_{TXDSOH}	Slave Mode: Slave Tx Data Output (to Master) Hold Time from next SSIClk	55.5[e]	-	-	ns
S12	T_{RXDSSU}	Slave Mode: Rx Data In (from master) setup time	0	-	-	ns
S13	T_{RXDSH}	Slave Mode: Rx Data In (from master) hold time	51.55[f]	-	-	ns

a. In master mode, the system clock must be at least twice as fast as the SSIClk.

b. In slave mode, the system clock must be at least 12 times faster than the SSIClk.

c. Note that the delays shown are using 8-mA drive strength.

d. This MAX value is for the minimum T_{SYSCLK} (12.5 ns). To find the MAX T_{TXDSOV} value for a larger T_{SYSCLK}, use the equation: $4*T_{SYSCLK}+27.74$.

e. This MIN value is for the minimum slave mode T_{SYSCLK} (12.5 ns). To find the MIN T_{TXDSOH} value for a larger T_{SYSCLK}, use the equation: $4*T_{SYSCLK}+5.50$.

f. This MIN value is for the minimum slave mode T_{SYSCLK} (12.5 ns). To find the MIN T_{RXDSH} value for a larger T_{SYSCLK}, use the equation: $4*T_{SYSCLK}+1.55$.

图 21-19. SSI Timing for TI Frame Format (FRF=01), Single Transfer Timing Measurement

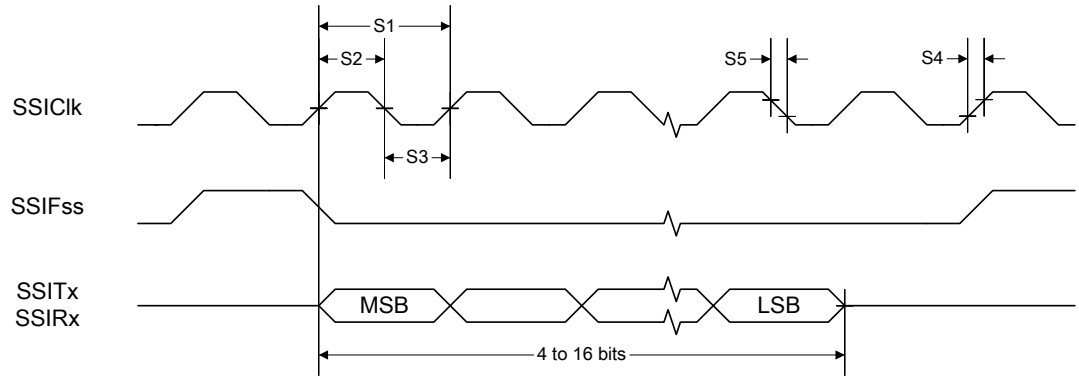


图 21-20. SSI Timing for MICROWIRE Frame Format (FRF=10), Single Transfer

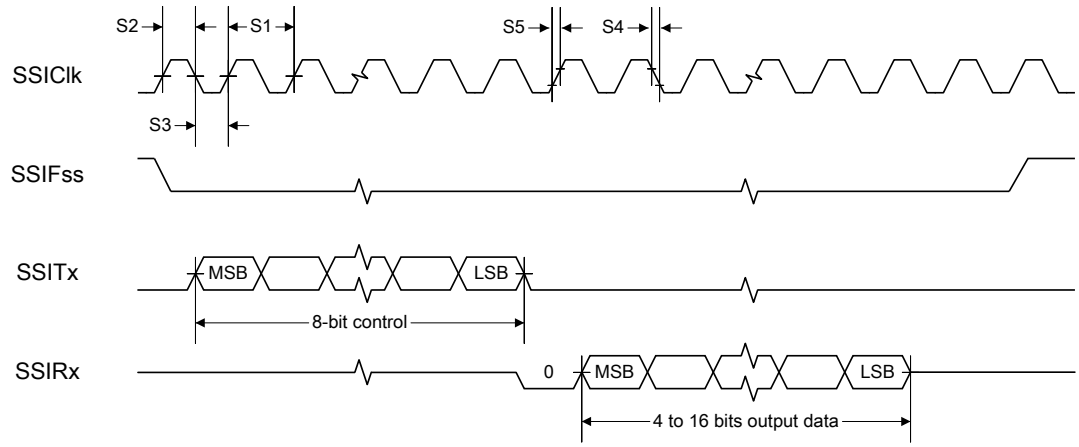


图 21-21. Master Mode SSI Timing for SPI Frame Format (FRF=00), with SPH=1

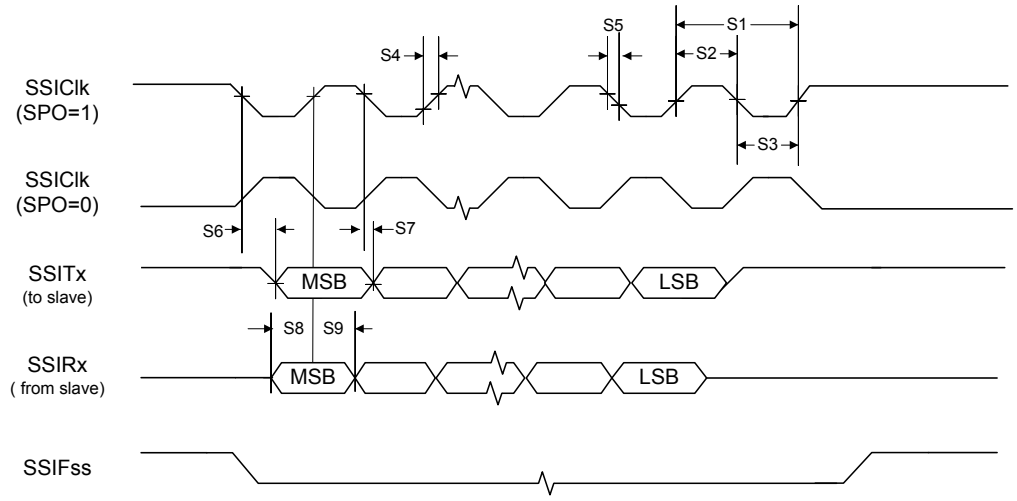
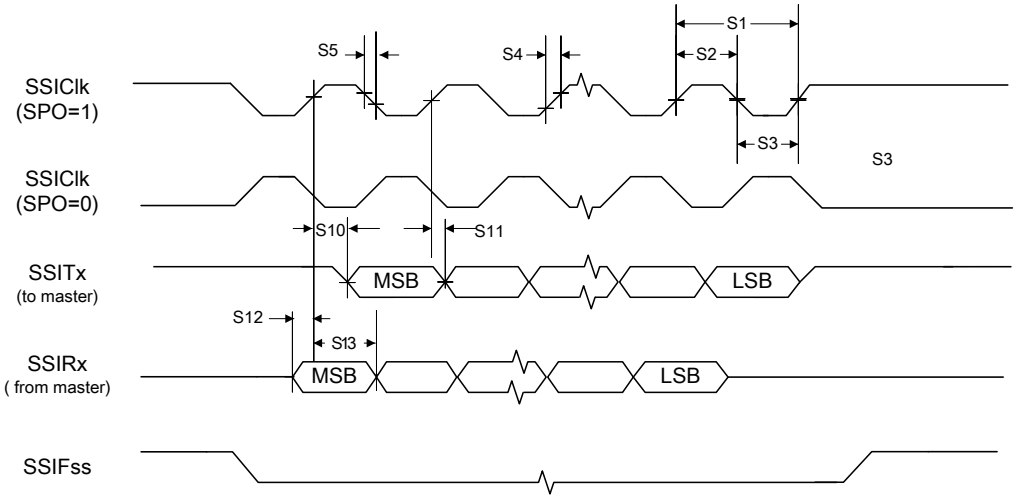


图 21-22. Slave Mode SSI Timing for SPI Frame Format (FRF=00), with SPH=1



21.16 Inter-Integrated Circuit (I^2C) Interface

表 21-34. I^2C Characteristics

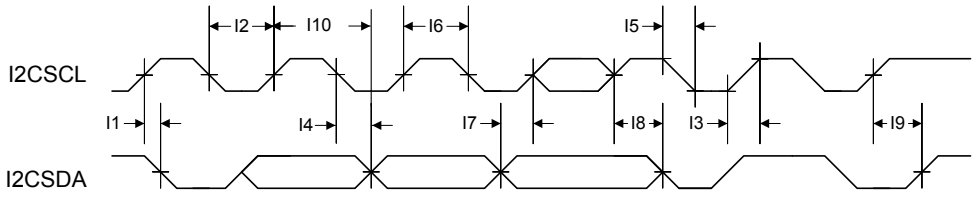
Parameter No.	Parameter	Parameter Name	Min	Nom	Max	Unit
I1[a]	T_{SCH}	Start condition hold time	36	-	-	system clocks
I2[a]	T_{LP}	Clock Low period	36	-	-	system clocks
I3[b]	T_{SRT}	I2CSCL/I2CSDA rise time (V_{IL} =0.5 V to V_{IH} =2.4 V)	-	-	(see note b)	ns
I4[a]	T_{DH}	Data hold time	2	-	-	system clocks
I5[c]	T_{SFT}	I2CSCL/I2CSDA fall time (V_{IH} =2.4 V to V_{IL} =0.5 V)	-	9	10	ns
I6[a]	T_{HT}	Clock High time	24	-	-	system clocks
I7[a]	T_{DS}	Data setup time	18	-	-	system clocks
I8[a]	T_{SCSR}	Start condition setup time (for repeated start condition only)	36	-	-	system clocks
I9[a]	T_{SCS}	Stop condition setup time	24	-	-	system clocks

a. Values depend on the value programmed into the TPR bit in the I^2C Master Timer Period (I2CMTPR) register; a TPR programmed for the maximum I2CSCL frequency (TPR=0x2) results in a minimum output timing as shown in the table above. The I^2C interface is designed to scale the actual data transition time to move it to the middle of the I2CSCL Low period. The actual position is affected by the value programmed into the TPR; however, the numbers given in the above values are minimum values.

b. Because I2CSCL and I2CSDA operate as open-drain-type signals, which the controller can only actively drive Low, the time I2CSCL or I2CSDA takes to reach a high level depends on external signal capacitance and pull-up resistor values.

c. Specified at a nominal 50 pF load.

图 21-23. I^2C Timing



21.17 Analog Comparator

表 21-35. Analog Comparator Characteristics[ab]

Parameter	Parameter Name	Min	Nom	Max	Unit
V_{INP},V_{INN}[c]	Input voltage range	GNDA	-	V_{DDA}	V
V_{CM}	Input common mode voltage range	GNDA	-	V_{DDA}	V
V_{OS}	Input offset voltage	-	±10	±50[d]	mV
I_{INP},I_{INN}	Input leakage current over full voltage range	-	-	2.0	µA
C_{MRR}	Common mode rejection ratio	-	50	-	dB
T_{RT}	Response time	-	-	1.0[e]	µs
T_{MC}	Comparator mode change to Output Valid	-	-	10	µs

a. Best design practices suggest that static or quiet digital I/O signals be configured adjacent to sensitive analog inputs to reduce capacitive coupling and cross talk.

b. To achieve best analog results, the source resistance driving the analog inputs, V_{INP} and V_{INN}, should be kept low.

c. The external voltage inputs to the Analog Comparator are designed to be highly sensitive and can be affected by external noise on the board. For this reason, V_{INP} and V_{INN} must be set to different voltage levels during idle states to ensure the analog comparator triggers are not enabled. If an internal voltage reference is used, it should be set to a mid-supply level. When operating in Sleep/Deep-Sleep modes, the Analog Comparator module external voltage inputs set to different levels (greater than the input offset voltage) to achieve minimum current draw.

d. Measured at VREF=100 mV.

e. Measured at external VREF=100 mV, input signal switching from 75 mV to 125 mV.

表 21-36. Analog Comparator Voltage Reference Characteristics

Parameter	Parameter Name	Min	Nom	Max	Unit
R_{HR}	Resolution in high range	-	V_{DDA}/29.4	-	V
R_{LR}	Resolution in low range	-	V_{DDA}/22.12	-	V
A_{HR}	Absolute accuracy high range	-	-	±R_{HR}/2	V
A_{LR}	Absolute accuracy low range	-	-	±R_{LR}/2	V

表 21-37. Analog Comparator Voltage Reference Characteristics, V_{DDA} = 3.3V, EN= 1, and RNG = 0

VREF Value	V_{IREF} Min	Ideal V_{IREF}	V_{IREF} Max	Unit
0x0	0.731	0.786	0.841	V
0x1	0.843	0.898	0.953	V
0x2	0.955	1.010	1.065	V
0x3	1.067	1.122	1.178	V
0x4	1.180	1.235	1.290	V
0x5	1.292	1.347	1.402	V
0x6	1.404	1.459	1.514	V
0x7	1.516	1.571	1.627	V
0x8	1.629	1.684	1.739	V
0x9	1.741	1.796	1.851	V
0xA	1.853	1.908	1.963	V
0xB	1.965	2.020	2.076	V
0xC	2.078	2.133	2.188	V

表 21-37. Analog Comparator Voltage Reference Characteristics, V_{DDA} = 3.3V, EN= 1, and RNG = 0（续）

VREF Value	V_{IREF} Min	Ideal V_{IREF}	V_{IREF} Max	Unit
0xD	2.190	2.245	2.300	V
0xE	2.302	2.357	2.412	V
0xF	2.414	2.469	2.525	V

表 21-38. Analog Comparator Voltage Reference Characteristics, V_{DDA} = 3.3V, EN= 1, and RNG = 1

VREF Value	V_{IREF} Min	Ideal V_{IREF}	V_{IREF} Max	Unit
0x0	0.000	0.000	0.074	V
0x1	0.076	0.149	0.223	V
0x2	0.225	0.298	0.372	V
0x3	0.374	0.448	0.521	V
0x4	0.523	0.597	0.670	V
0x5	0.672	0.746	0.820	V
0x6	0.822	0.895	0.969	V
0x7	0.971	1.044	1.118	V
0x8	1.120	1.193	1.267	V
0x9	1.269	1.343	1.416	V
0xA	1.418	1.492	1.565	V
0xB	1.567	1.641	1.715	V
0xC	1.717	1.790	1.864	V
0xD	1.866	1.939	2.013	V
0xE	2.015	2.089	2.162	V
0xF	2.164	2.238	2.311	V

21.18 Current Consumption

表 21-39. Current Consumption

Parameter	Parameter Name	Conditions	System Clock		Nom			Max	Unit
			Frequency	Clock Source	-40°C	25°C	85°C	85°C	
I_{DD_RUN}	Run mode (Flash loop)	V_{DD} = 3.3 V V_{DDA} = 3.3 V Peripherals = All ON	80 MHz	MOSC with PLL	45.0	45.1	45.7	54.9	mA
			40 MHz	MOSC with PLL	31.9	32.0	32.7	40.6	mA
			16 MHz	MOSC with PLL	19.6	19.7	20.3	27.6	mA
			16 MHz	PIOSC	17.5	17.6	18.0	25.3	mA
			1 MHz	PIOSC	10.0	10.1	10.5	17.5	mA
		V_{DD} = 3.3 V V_{DDA} = 3.3 V Peripherals = All OFF	80 MHz	MOSC with PLL	24.5	24.7	25.2	31.3	mA
			40 MHz	MOSC with PLL	19.6	19.7	20.4	25.9	mA
			16 MHz	MOSC with PLL	12.1	12.2	12.7	18.7	mA
			16 MHz	PIOSC	10.1	10.1	10.5	16.4	mA
			1 MHz	PIOSC	5.45	5.50	5.98	11.6	mA
	Run mode (SRAM loop)	V_{DD} = 3.3 V V_{DDA} = 3.3 V Peripherals = All ON	80 MHz	MOSC with PLL	34.7	34.9	35.5	44.2	mA
			40 MHz	MOSC with PLL	22.2	22.4	22.9	30.2	mA
			16 MHz	MOSC with PLL	14.7	14.8	15.3	21.8	mA
			16 MHz	PIOSC	12.8	12.9	13.4	19.7	mA
			1 MHz	PIOSC	8.07	8.16	8.61	14.6	mA
		V_{DD} = 3.3 V V_{DDA} = 3.3 V Peripherals = All OFF	80 MHz	MOSC with PLL	15.2	15.3	15.8	21.7	mA
			40 MHz	MOSC with PLL	10.3	10.5	10.9	16.2	mA
			16 MHz	MOSC with PLL	7.32	7.45	7.92	13.0	mA
			16 MHz	PIOSC	5.87	5.96	6.35	13.7	mA
			1 MHz	PIOSC	3.54	3.63	4.07	8.84	mA
I_{DDA}[a]	Run, Sleep and Deep-sleep mode	V_{DD} = 3.3 V V_{DDA} = 3.3 V Peripherals = All ON	-	MOSC with PLL, PIOSC	2.71	2.71	2.71	3.97	mA
	Deep-Sleep mode		30 kHz	LFIOSC	2.54	2.54	2.54	3.68	mA
	Run, Sleep and Deep-sleep mode	V_{DD} = 3.3 V V_{DDA} = 3.3 V Peripherals = All OFF	-	MOSC with PLL, PIOSC, LFIOSC	0.28	0.28	0.29	0.56	mA

表 21-39. Current Consumption（续）

Parameter	Parameter Name	Conditions	System Clock		Nom			Max	Unit
			Frequency	Clock Source	-40°C	25°C	85°C	85°C	
I_{DD_SLEEP}	Sleep mode (FLASHPM = 0x0)	V_{DD} = 3.3 V V_{DDA} = 3.3 V Peripherals = All ON LDO = 1.2 V	80 MHz	MOSC with PLL	29.3	29.5	30.0	38.1	mA
			40 MHz	MOSC with PLL	19.5	19.7	20.2	27.1	mA
			16 MHz	MOSC with PLL	13.6	13.8	14.2	20.6	mA
			16 MHz	PIOSC[b]	11.7	11.8	12.2	18.5	mA
			1 MHz	PIOSC[b]	7.01	7.06	7.93	12.0	mA
		V_{DD} = 3.3 V V_{DDA} = 3.3 V Peripherals = All OFF LDO = 1.2 V	80 MHz	MOSC with PLL	9.60	9.73	10.2	15.4	mA
			40 MHz	MOSC with PLL	7.49	7.60	8.06	13.2	mA
			16 MHz	MOSC with PLL	6.22	6.33	6.78	11.7	mA
			16 MHz	PIOSC[b]	4.28	4.35	4.77	9.52	mA
			1 MHz	PIOSC[b]	3.52	3.59	4.01	8.70	mA
	Sleep mode (FLASHPM = 0x2)	V_{DD} = 3.3 V V_{DDA} = 3.3 V Peripherals = All ON LDO = 1.2 V	80 MHz	MOSC with PLL	28.4	28.6	29.2	37.2	mA
			40 MHz	MOSC with PLL	18.6	18.8	19.3	26.2	mA
			16 MHz	MOSC with PLL	12.7	12.9	13.3	19.7	mA
			16 MHz	PIOSC[b]	10.8	10.9	11.3	17.5	mA
			1 MHz	PIOSC[b]	7.09	7.20	7.67	13.6	mA
		V_{DD} = 3.3 V V_{DDA} = 3.3 V Peripherals = All OFF LDO = 1.2 V	80 MHz	MOSC with PLL	8.66	8.82	9.31	14.5	mA
			40 MHz	MOSC with PLL	6.55	6.69	7.17	12.1	mA
			16 MHz	MOSC with PLL	5.27	5.41	5.89	10.7	mA
			16 MHz	PIOSC[b]	3.34	3.44	3.88	8.65	mA
			1 MHz	PIOSC[b]	2.58	2.67	3.13	7.85	mA

表 21-39. Current Consumption（续）

Parameter	Parameter Name	Conditions	System Clock		Nom			Max	Unit
			Frequency	Clock Source	-40°C	25°C	85°C	85°C	
$I_{DD_DEEPSLEEP}$	Deep-sleep mode (FLASHPM = 0x0)	V_{DD} = 3.3 V V_{DDA} = 3.3 V Peripherals = All ON LDO = 1.2 V	16 MHz	PIOSC	9.29	9.29	9.66	15.9	mA
			30 kHz	LFIOSC	5.10	5.10	5.48	11.2	mA
		V_{DD} = 3.3 V V_{DDA} = 3.3 V Peripherals = All OFF LDO = 1.2 V	16 MHz	PIOSC	3.51	3.51	3.91	8.67	mA
			30 kHz	LFIOSC	2.00	2.00	2.39	7.24	mA
	Deep-sleep mode (FLASHPM = 0x2)	V_{DD} = 3.3 V V_{DDA} = 3.3 V Peripherals = All ON LDO = 1.2 V	16 MHz	PIOSC	8.34	8.36	8.77	14.9	mA
			30 kHz	LFIOSC	4.14	4.18	4.59	10.4	mA
		V_{DD} = 3.3 V V_{DDA} = 3.3 V Peripherals = All OFF LDO = 1.2 V	16 MHz	PIOSC	2.56	2.60	3.02	7.79	mA
			30 kHz	LFIOSC	1.04	1.07	1.49	6.48	mA
I_{HIB_NORTC}	Hibernate mode (external wake, RTC disabled)	V_{BAT} = 3.0 V V_{DD} = 0 V V_{DDA} = 0 V System Clock = OFF Hibernate Module = 32.768 kHz	-	-	1.23	1.38	1.54	5.20	µA
I_{HIB_RTC}	Hibernate mode (RTC enabled)	V_{BAT} = 3.0 V V_{DD} = 0 V V_{DDA} = 0 V System Clock = OFF Hibernate Module = 32.768 kHz	-	-	1.27	1.40	1.69	5.24	µA
I_{HIB_VDD3ON}	Hibernate mode (VDD3ON mode, RTC on)	V_{BAT} = 3.0 V V_{DD} = 3.3 V V_{DDA} = 3.3 V System Clock = OFF Hibernate Module = 32.768 kHz	-	-	3.17	4.49	10.6	28.1	µA
	Hibernate mode (VDD3ON mode, RTC off)	V_{BAT} = 3.0 V V_{DD} = 3.3 V V_{DDA} = 3.3 V System Clock = OFF Hibernate Module = 32.768 kHz	-	-	3.16	4.33	10.4	27.7	µA

a. The value for I_{DDA} is included in the above values for I_{DD_RUN}, I_{DD_SLEEP}, and $I_{DD_DEEPSLEEP}$.

b. Note that if the MOSC is the source of the Run-mode system clock and is powered down in Sleep mode, wake time is increased by T_{MOSC_SETTLE}.

A 封装信息

A.1 可订购器件

注意: 关于可订购器件型号的更多细节，请参见封装选项附录。

图 A-1. 器件型号说明

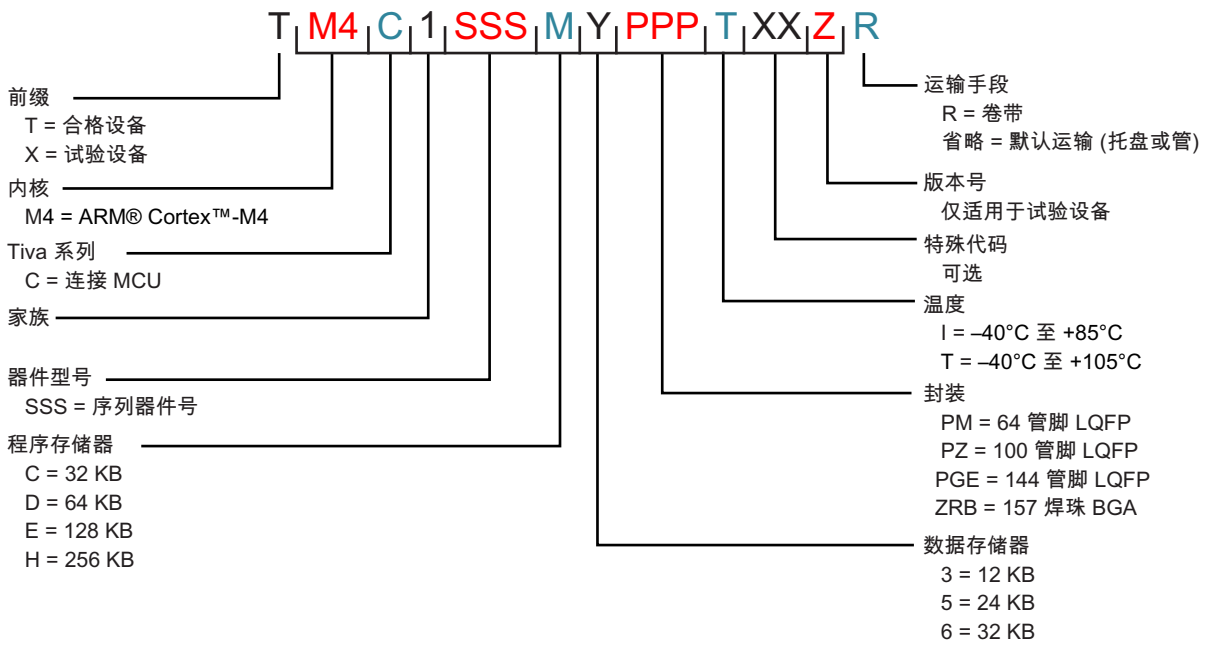


表 A-1. 可订购的器件型号

器件型号	描述
TM4C1231H6PMI	Tiva™ C 系列 TM4C1231H6PM 微控制器工业温度范围 64 管脚 LQFP
TM4C1231H6PMIR	Tiva™ C 系列 TM4C1231H6PM 微控制器工业温度范围 64 管脚 LQFP 卷带

A.2 型号标识

Tiva™ C 系列 微控制器均带有标识号标记，如下图所示。



此标识号标记包含以下信息：

- 第 1 行和第 5 行：内部跟踪编号
- 第 2 行和第 3 行：器件型号

  例如，第二行为 TM4C123G，第三行为 E6PZI，这表示订购器件型号为 TM4C123GE6PZI。器件号中的第一个字母表示产品状态。T表示器件完全合格，已正式投入生产；X表示器件仍处于试验阶段（预生产），需要提供责任豁免声明。预产器件的器件型号中还包含一个版本号。例如，XM4C123G 后面注明 E6PZI5，这表示第 5 版。投产器件的器件型号中不包含版本号。一些器件的温度字符后还会写上内部代码。DID0 寄存器负责标识微控制器的版本。MAJOR 和 MINOR 位域标识了模具版本号。MAJOR 和 MINOR 位域一起标识了 TM4C123x 微控制器器件的版本号。

MAJOR 位域值	MINOR 位域值	模具版本	器件版本
0x0	0x0	A0	1
0x0	0x1	A1	2
0x0	0x2	A2	3
0x0	0x3	A3	4
0x1	0x0	B0	5
0x1	0x1	B1	6
0x1	0x2	B2	7

- 第 4 行：日期代码

  第四行的前两个字符为日期代码，紧随其后的是内部跟踪编号。日期代码 YM 包含两位数，分别表示年份的最后一个数字和月份。例如，第四行的前两个数字为 34，这个日期代码表示 2013 年 4 月。

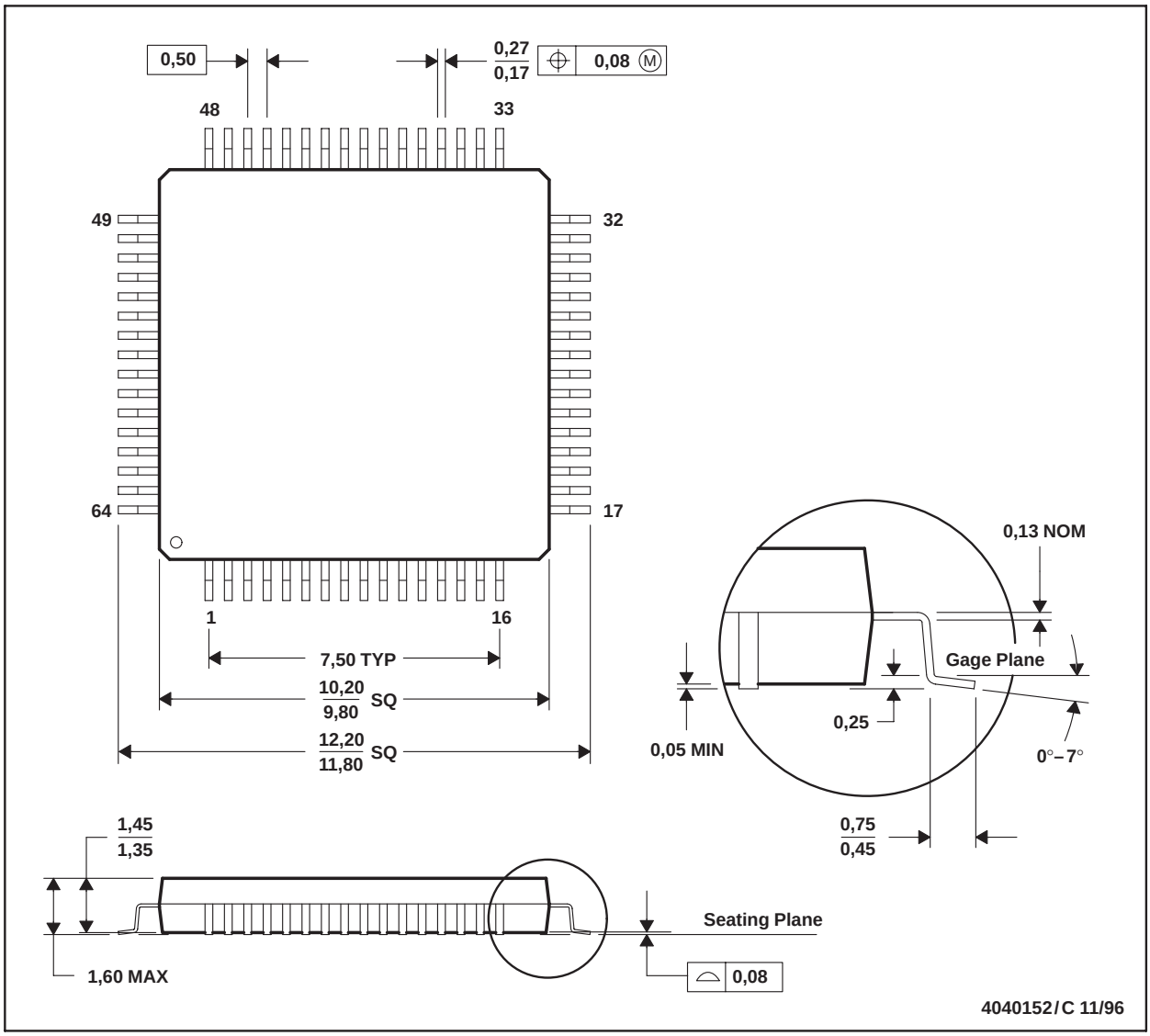
A.3 封装图

图 **A-2. TM4C1231H6PM 64 管脚 LQFP 封装图**

MECHANICAL DATA

MTQF008A – JANUARY 1995 – REVISED DECEMBER 1996

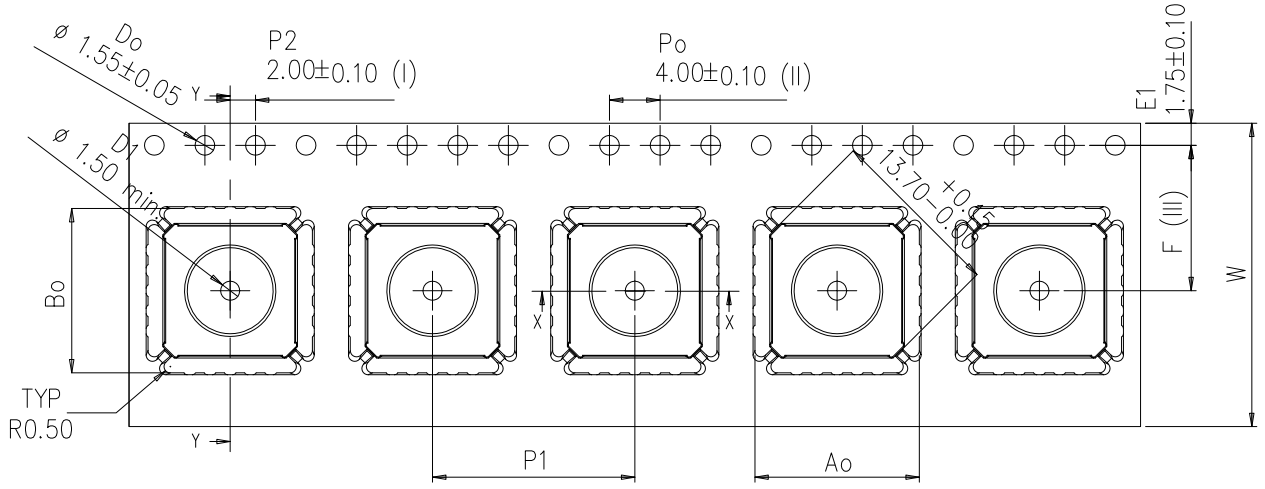
PM (S-PQFP-G64) **PLASTIC QUAD FLATPACK**

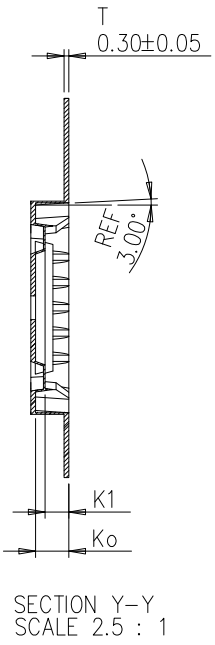


NOTES: A. All linear dimensions are in millimeters.
B. This drawing is subject to change without notice.
C. Falls within JEDEC MS-026
D. May also be thermally enhanced plastic with leads connected to the die pads.

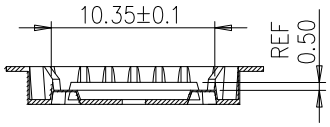
A.4 封装材料

图 A-3. 64 管脚 LQFP PM 封装载带





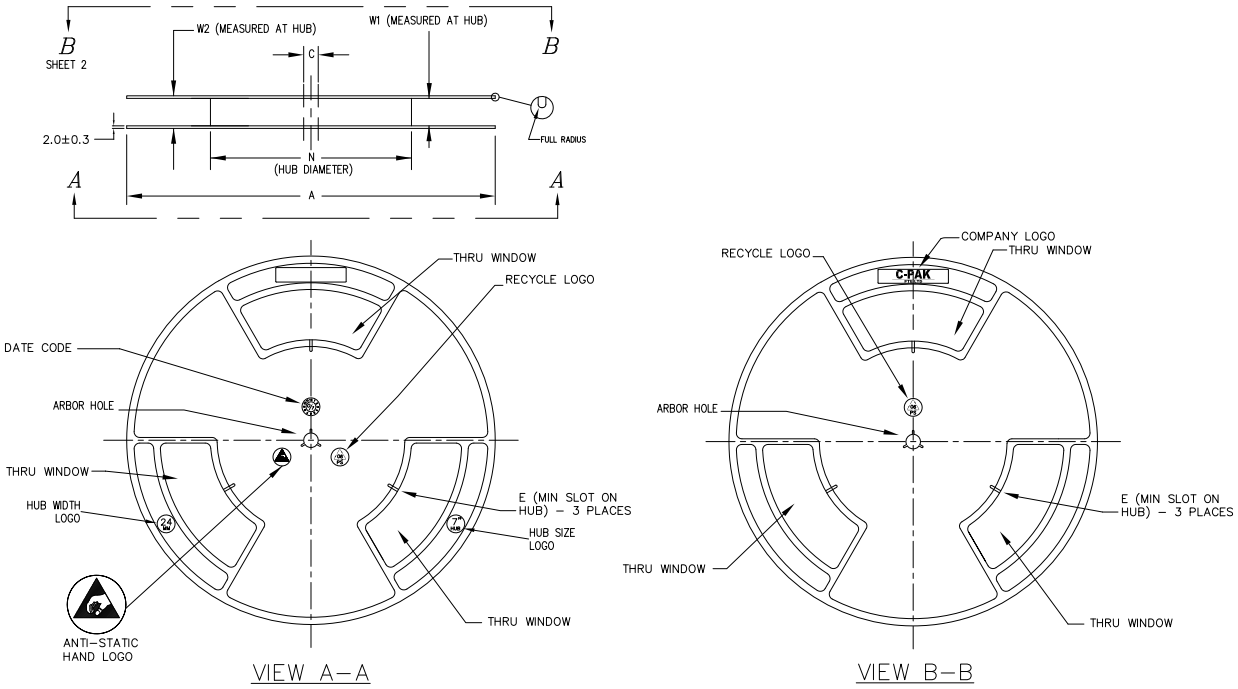
SECTION Y−Y
SCALE 2.5 : 1



SECTION X−X
SCALE 2.5 : 1

Ao	13.00 ±0.10
Bo	13.00 ±0.10
Ko	2.10 ±0.10
K1	1.50 ±0.10
F	11.50 ±0.10
P1	16.00 ±0.10
W	24.00 ±0.30

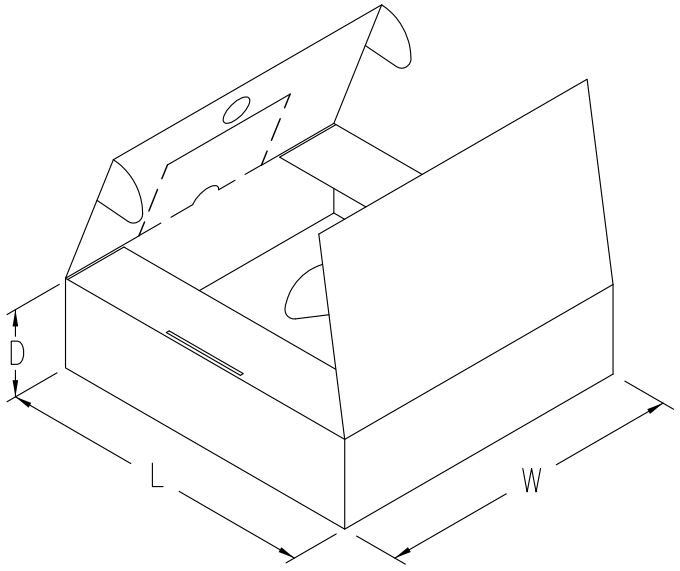
图 A-4. 64 管脚 LQFP PM 封装塑料圆盘



DIMENSIONS (MM)

MAT'L REV LTR	PART NUMBER	C–PAK NUMBER	MATL TYPE	TAPE SIZE	A (MAX.)	B (±0.5)	C (+0.5/–0.2)	D (MIN.)	N (MIN.)	W1 (+2.0,–0.0)	W2 (MAX.)
A	4203746–0005	13E7 W24 BK CP/TI	ASD	24.0	330	1.5	13.0	20.2	178±2.0	24.4	30.4

图 A-5. 64 管脚 LQFP PM 封装用载带圆盘箱



INSIDE DIMENSIONS
INCH [MM]

L = 13.25 [336.6]

W = 13.25 [336.6]

D = 1.625 [41.27]

PACKAGING INFORMATION

Orderable part number	Status (1)	Material type (2)	Package | Pins	Package qty | Carrier	RoHS (3)	Lead finish/ Ball material (4)	MSL rating/ Peak reflow (5)	Op temp (°C)	Part marking (6)
TM4C1231H6PMI	Active	Production	LQFP (PM) | 64	160 | JEDEC TRAY (5+1)	Yes	NIPDAU	Level-3-260C-168 HR	-40 to 85	TM4C1231 H6PMI
TM4C1231H6PMI.B	Active	Production	LQFP (PM) | 64	160 | JEDEC TRAY (5+1)	Yes	NIPDAU	Level-3-260C-168 HR	-40 to 85	TM4C1231 H6PMI
TM4C1231H6PMI7	Active	Production	LQFP (PM) | 64	160 | JEDEC TRAY (10+1)	Yes	NIPDAU	Level-3-260C-168 HR	-40 to 85	TM4C1231 H6PMI7
TM4C1231H6PMI7.B	Active	Production	LQFP (PM) | 64	160 | JEDEC TRAY (10+1)	Yes	NIPDAU	Level-3-260C-168 HR	-40 to 85	TM4C1231 H6PMI7
TM4C1231H6PMI7R	Active	Production	LQFP (PM) | 64	1000 | LARGE T&R	Yes	NIPDAU	Level-3-260C-168 HR	-40 to 85	TM4C1231 H6PMI7
TM4C1231H6PMI7R.B	Active	Production	LQFP (PM) | 64	1000 | LARGE T&R	Yes	NIPDAU	Level-3-260C-168 HR	-40 to 85	TM4C1231 H6PMI7
TM4C1231H6PMI7RG4	Active	Production	LQFP (PM) | 64	1000 | LARGE T&R	Yes	NIPDAU	Level-3-260C-168 HR	-40 to 85	TM4C1231 H6PMI7
TM4C1231H6PMI7RG4.B	Active	Production	LQFP (PM) | 64	1000 | LARGE T&R	Yes	NIPDAU	Level-3-260C-168 HR	-40 to 85	TM4C1231 H6PMI7
TM4C1231H6PMIR	Active	Production	LQFP (PM) | 64	1000 | LARGE T&R	Yes	NIPDAU	Level-3-260C-168 HR	-40 to 85	TM4C1231 H6PMI
TM4C1231H6PMIR.B	Active	Production	LQFP (PM) | 64	1000 | LARGE T&R	Yes	NIPDAU	Level-3-260C-168 HR	-40 to 85	TM4C1231 H6PMI

(1) **Status:** For more details on status, see our product life cycle.

(2) **Material type:** When designated, preproduction parts are prototypes/experimental devices, and are not yet approved or released for full production. Testing and final process, including without limitation quality assurance, reliability performance testing, and/or process qualification, may not yet be complete, and this item is subject to further changes or possible discontinuation. If available for ordering, purchases will be subject to an additional waiver at checkout, and are intended for early internal evaluation purposes only. These items are sold without warranties of any kind.

(3) **RoHS values:** Yes, No, RoHS Exempt. See the TI RoHS Statement for additional information and value definition.

(4) **Lead finish/Ball material:** Parts may have multiple material finish options. Finish options are separated by a vertical ruled line. Lead finish/Ball material values may wrap to two lines if the finish value exceeds the maximum column width.

(5) **MSL rating/Peak reflow:** The moisture sensitivity level ratings and peak solder (reflow) temperatures. In the event that a part has multiple moisture sensitivity ratings, only the lowest level per JEDEC standards is shown. Refer to the shipping label for the actual reflow temperature that will be used to mount the part to the printed circuit board.

(6) **Part marking:** There may be an additional marking, which relates to the logo, the lot trace code information, or the environmental category of the part.

Multiple part markings will be inside parentheses. Only one part marking contained in parentheses and separated by a "~" will appear on a part. If a line is indented then it is a continuation of the previous line and the two combined represent the entire part marking for that device.

Important Information and Disclaimer: The information provided on this page represents TI's knowledge and belief as of the date that it is provided. TI bases its knowledge and belief on information provided by third parties, and makes no representation or warranty as to the accuracy of such information. Efforts are underway to better integrate information from third parties. TI has taken and continues to take reasonable steps to provide representative and accurate information but may not have conducted destructive testing or chemical analysis on incoming materials and chemicals. TI and TI suppliers consider certain information to be proprietary, and thus CAS numbers and other limited information may not be available for release.

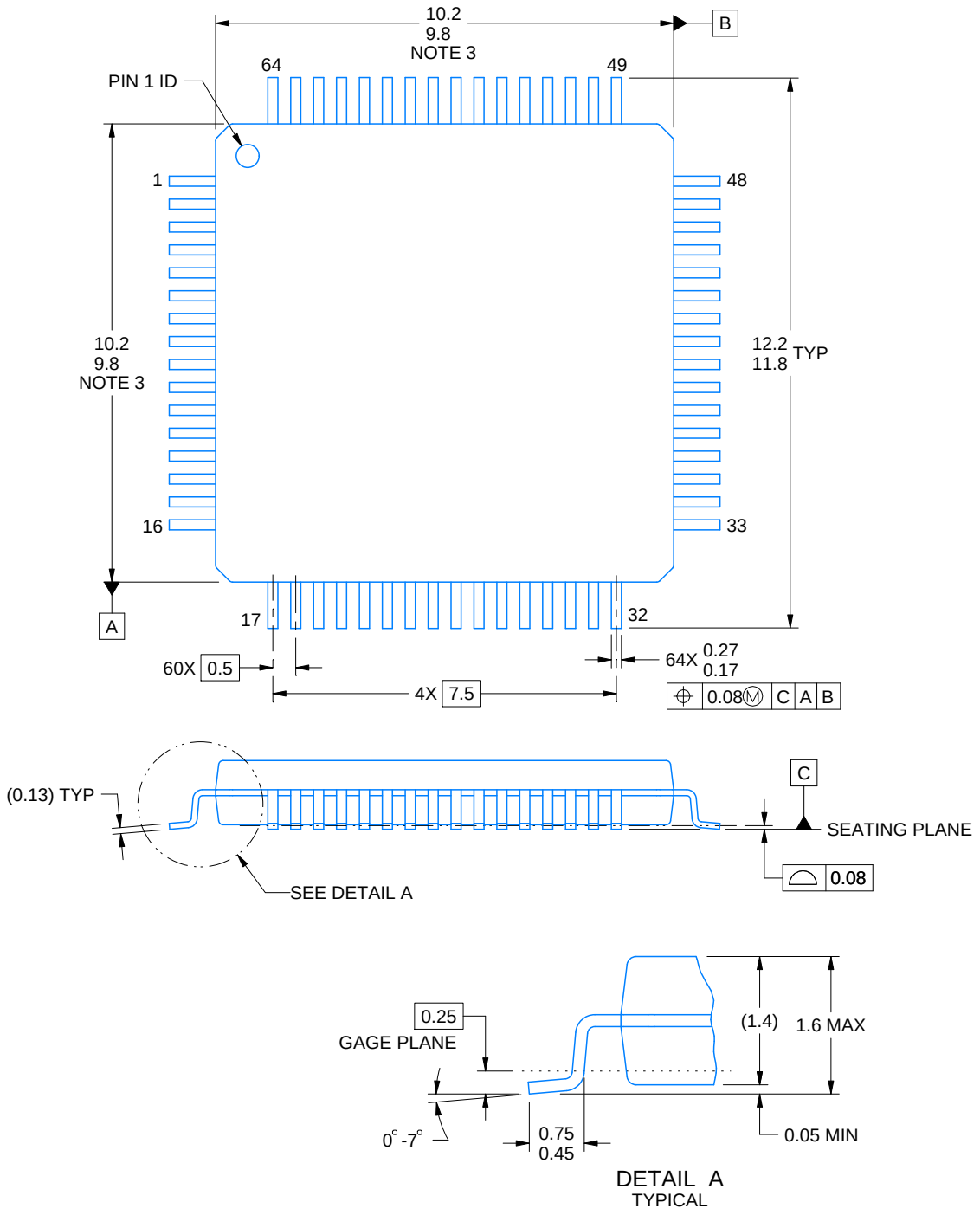
In no event shall TI's liability arising out of such information exceed the total purchase price of the TI part(s) at issue in this document sold by TI to Customer on an annual basis.

PACKAGE OUTLINE

PM0064A

LQFP - 1.6 mm max height

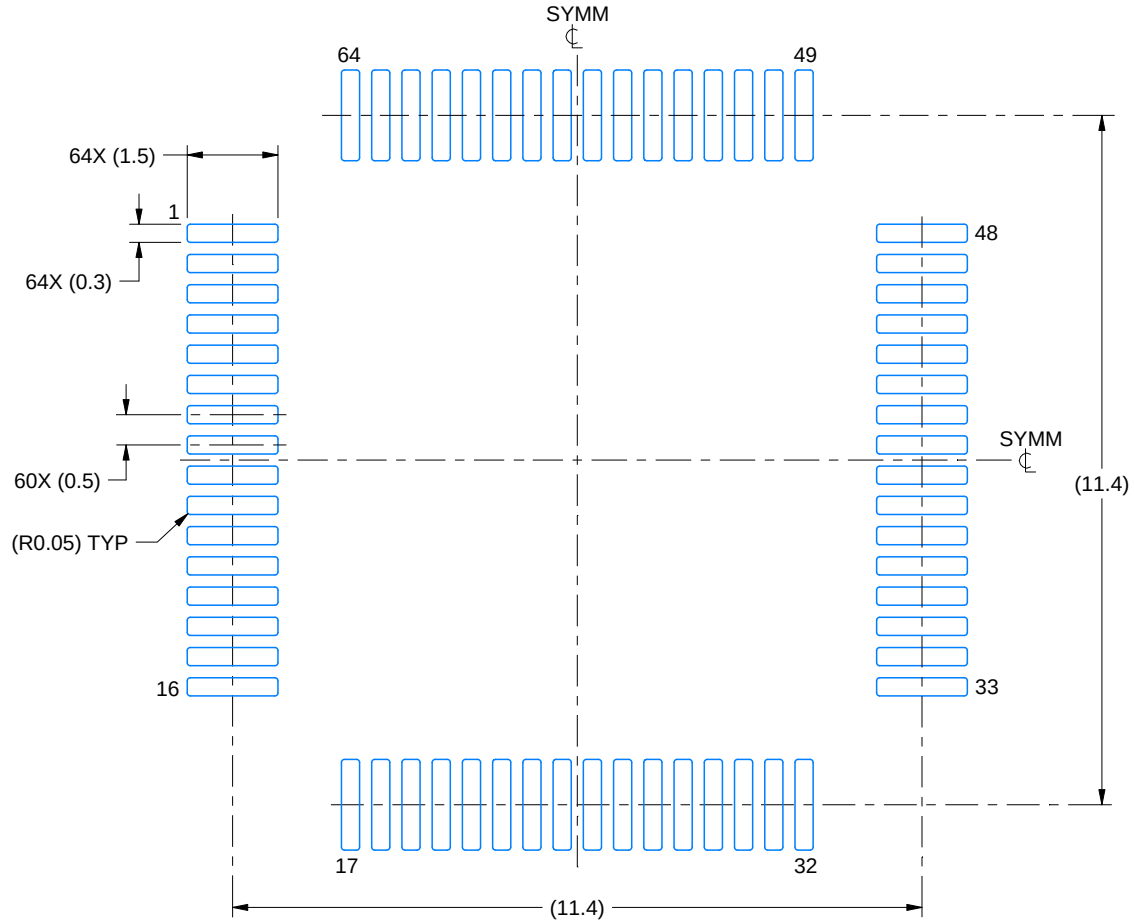
PLASTIC QUAD FLATPACK



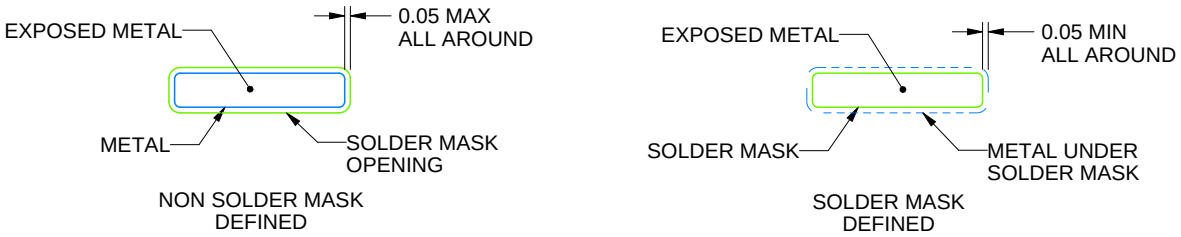
4215162/A 03/2017

NOTES:

1. All linear dimensions are in millimeters. Any dimensions in parenthesis are for reference only. Dimensioning and tolerancing per ASME Y14.5M.
2. This drawing is subject to change without notice.
3. This dimension does not include mold flash, protrusions, or gate burrs. Mold flash, protrusions, or gate burrs shall not exceed 0.15 mm per side.
4. Reference JEDEC registration MS-026.



LAND PATTERN EXAMPLE
EXPOSED METAL SHOWN
SCALE:8X

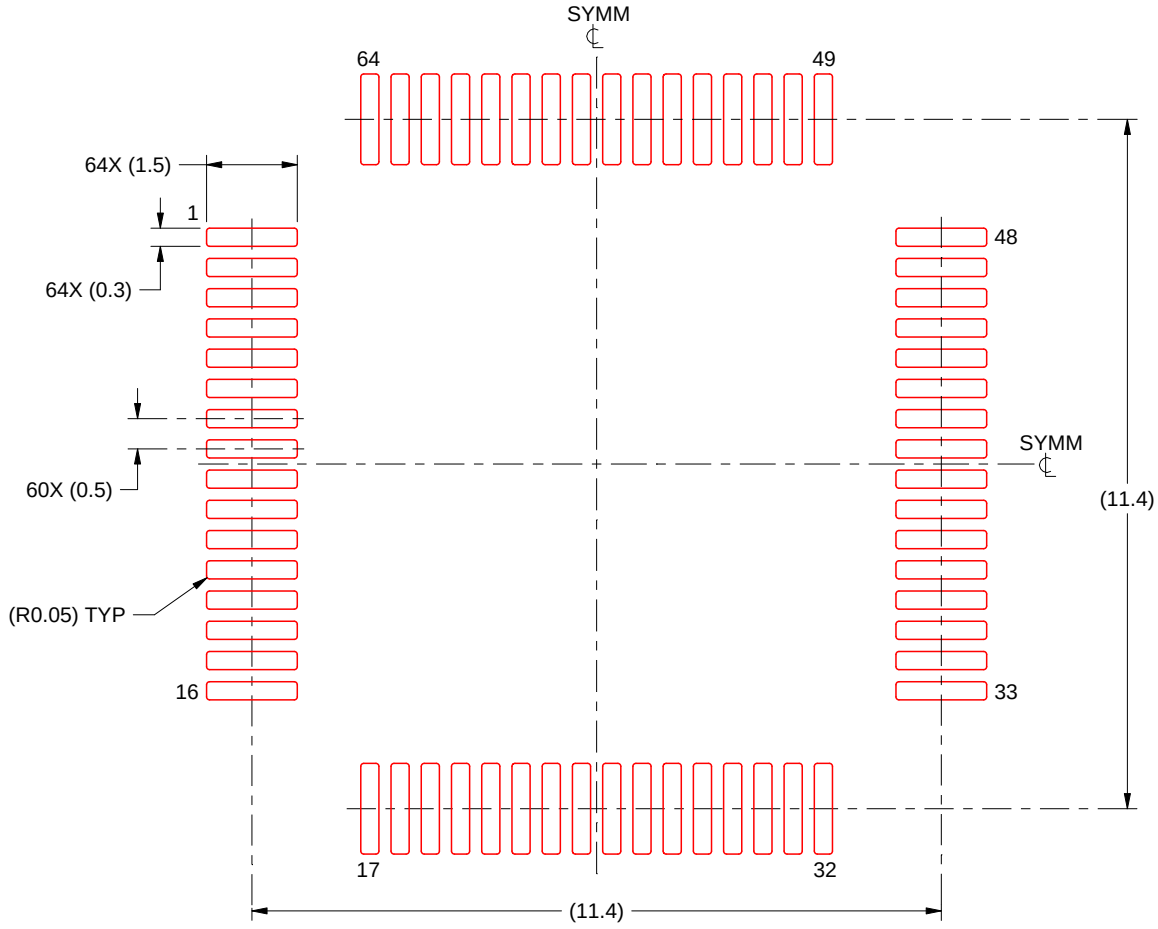


SOLDER MASK DETAILS

4215162/A 03/2017

NOTES: (continued)

5. Publication IPC-7351 may have alternate designs.
6. Solder mask tolerances between and around signal pads can vary based on board fabrication site.
7. For more information, see Texas Instruments literature number SLMA004 (www.ti.com/lit/slma004).



SOLDER PASTE EXAMPLE
BASED ON 0.125 mm THICK STENCIL
SCALE:8X

4215162/A 03/2017

NOTES: (continued)

8. Laser cutting apertures with trapezoidal walls and rounded corners may offer better paste release. IPC-7525 may have alternate design recommendations.
9. Board assembly site may have different recommendations for stencil design.

重要通知和免责声明

TI“按原样”提供技术和可靠性数据（包括数据表）、设计资源（包括参考设计）、应用或其他设计建议、网络工具、安全信息和其他资源，不保证没有瑕疵且不做出任何明示或暗示的担保，包括但不限于对适销性、与某特定用途的适用性或不侵犯任何第三方知识产权的暗示担保。

这些资源可供使用 TI 产品进行设计的熟练开发人员使用。您将自行承担以下全部责任：(1) 针对您的应用选择合适的 TI 产品，(2) 设计、验证并测试您的应用，(3) 确保您的应用满足相应标准以及任何其他安全、安保法规或其他要求。

这些资源如有变更，恕不另行通知。TI 授权您仅可将这些资源用于研发本资源所述的 TI 产品的相关应用。严禁以其他方式对这些资源进行复制或展示。您无权使用任何其他 TI 知识产权或任何第三方知识产权。对于因您对这些资源的使用而对 TI 及其代表造成的任何索赔、损害、成本、损失和债务，您将全额赔偿，TI 对此概不负责。

TI 提供的产品受 TI 销售条款)、TI 通用质量指南 或 ti.com 上其他适用条款或 TI 产品随附的其他适用条款的约束。TI 提供这些资源并不会扩展或以其他方式更改 TI 针对 TI 产品发布的适用的担保或担保免责声明。 除非德州仪器 (TI) 明确将某产品指定为定制产品或客户特定产品，否则其产品均为按确定价格收入目录的标准通用器件。

TI 反对并拒绝您可能提出的任何其他或不同的条款。

版权所有 © 2025，德州仪器 (TI) 公司

最后更新日期：2025 年 10 月