

# **NuMicro® Family**

## **M480 Series**

### **Technical Reference Manual**

*The information described in this document is the exclusive intellectual property of Nuvoton Technology Corporation and shall not be reproduced without permission from Nuvoton.*

*Nuvoton is providing this document only for reference purposes of NuMicro microcontroller based system design. Nuvoton assumes no responsibility for errors or omissions.*

*All data and specifications are subject to change without notice.*

For additional information or questions, please contact: Nuvoton Technology Corporation.

[www.nuvoton.com](http://www.nuvoton.com)

## 目录

1 概述 .....	30
2 特性 .....	31
2.1 NuMicro® M480 特性 .....	31
3 料号信息 .....	40
3.1 概要 .....	40
3.2 封装 .....	40
3.3 NuMicro® M481 基本系列 .....	41
3.4 NuMicro® M482 USB FS OTG 系列 .....	42
3.5 NuMicro® M483 CAN 系列 .....	43
3.6 NuMicro® M484 USB HS OTG 系列 .....	44
3.7 NuMicro® M485 Crypto 系列 .....	45
3.8 NuMicro® M487 以太网系列 .....	46
3.9 NuMicro® M480 命名规则 .....	47
4 引脚配置 .....	48
4.1 引脚配置 .....	48
4.1.1 NuMicro® M481 基本系列 QFN33 引脚图 .....	48
4.1.2 NuMicro® M481 基本系列 LQFP48 引脚图 .....	49
4.1.3 NuMicro® M481 基本系列 LQFP64 引脚图 .....	50
4.1.4 NuMicro® M482 USB FS OTG 系列 QFN33 引脚图 .....	51
4.1.5 NuMicro® M482 USB FS OTG 系列 LQFP48 引脚图 .....	52
4.1.6 NuMicro® M482 USB FS OTG 系列 LQFP64 引脚图 .....	53
4.1.7 NuMicro® M482 USB FS OTG 系列 LQFP128 引脚图 .....	54
4.1.8 NuMicro® M483 CAN 系列 LQFP64 引脚图 .....	55
4.1.9 NuMicro® M483 CAN 系列 LQFP128 引脚图 .....	56
4.1.10 NuMicro® M484 USB HS OTG 系列 LQFP64 引脚图 .....	57
4.1.11 NuMicro® M484 USB HS OTG 系列 带有两组USB LQFP64 引脚图 .....	58
4.1.12 NuMicro® M484 USB HS OTG 系列 LQFP128 引脚图 .....	59
4.1.13 NuMicro® M485 加解密系列 QFN33 引脚图 .....	60
4.1.14 NuMicro® M485 加解密系列 LQFP48 引脚图 .....	61
4.1.15 NuMicro® M485 加解密系列 LQFP64 引脚图 .....	62
4.1.16 NuMicro® M485 加解密系列 LQFP128 引脚图 .....	63
4.1.17 NuMicro® M487 以太网系列 LQFP64 引脚图 .....	64
4.1.18 NuMicro® M487 以太网系列 LQFP128 引脚图 .....	65
4.1.19 NuMicro® M487 以太网系列 LQFP144 引脚图 .....	66
4.2 引脚描述 .....	67

4.2.1 M481 系列引脚描述.....	67
4.2.2 M482 系列引脚描述.....	88
4.2.3 M483 系列引脚描述.....	120
4.2.4 M484 系列引脚描述.....	153
4.2.5 M485 系列引脚描述.....	185
4.2.6 M487 系列引脚描述.....	217
4.2.7 M480 多功能管脚摘要表.....	253
4.2.8 M480 按GPIO分类的多功能引脚摘要表.....	282
<b>5 框图 .....</b>	<b>317</b>
5.1 NuMicro® M480 框图.....	317
<b>6 功能描述 .....</b>	<b>318</b>
6.1 ARM® Cortex®-M4 内核 .....	318
6.2 系统管理.....	321
6.2.1 概述.....	321
6.2.2 系统复位.....	321
6.2.3 系统电源分配.....	326
6.2.4 电源模式和唤醒源.....	327
6.2.5 电源模式转换.....	331
6.2.6 系统内存映射.....	332
6.2.7 SRAM 内存结构.....	335
6.2.8 总线矩阵.....	336
6.2.9 HIRC自动校准.....	337
6.2.10 寄存器映射.....	338
6.2.11 寄存器描述.....	340
6.2.12 系统定时器(SysTick).....	394
6.2.13 NVIC 可嵌套中断向量.....	398
6.2.14 系统控制寄存器.....	428
6.3 时钟控制器.....	437
6.3.1 概述.....	437
6.3.2 时钟发生器.....	439
6.3.3 系统时钟和SysTick 时钟.....	440
6.3.4 外设时钟.....	441
6.3.5 掉电模式时钟.....	442
6.3.6 时钟输出.....	442
6.3.7 USB时钟源 .....	443
6.3.8 寄存器映射.....	444
6.3.9 寄存器描述.....	446
6.4 FMC 存储控制器 .....	495
6.4.1 概述.....	495

6.4.2 FMC特性.....	495
6.4.3 框图.....	495
6.4.4 功能描述.....	497
6.4.5 寄存器映射.....	541
6.4.6 寄存器描述.....	541
6.5 GPIO 通用 I/O .....	570
6.5.1 概述.....	570
6.5.2 特性.....	570
6.5.3 框图.....	571
6.5.4 基本配置.....	571
6.5.5 功能描述.....	572
6.5.6 寄存器映射.....	576
6.5.7 寄存器描述.....	580
6.6 PDMA 控制器 .....	597
6.6.1 概述.....	597
6.6.2 特性.....	597
6.6.3 框图.....	597
6.6.4 基本配置.....	597
6.6.5 功能描述.....	598
6.6.6 寄存器映射.....	604
6.6.7 寄存器描述.....	608
6.7 TMR 定时器控制器 .....	641
6.7.1 概述.....	641
6.7.2 特性.....	641
6.7.3 框图.....	642
6.7.4 基本配置.....	646
6.7.5 定时器功能描述.....	647
6.7.6 PWM 功能描述.....	652
6.7.7 寄存器映射.....	669
6.7.8 寄存器描述.....	674
6.8 WDT 看门狗定时器.....	718
6.8.1 概述.....	718
6.8.2 特性.....	718
6.8.3 框图.....	718
6.8.4 基本配置.....	718
6.8.5 功能描述.....	719
6.8.6 寄存器映射.....	722
6.8.7 寄存器描述.....	723
6.9 WWDT 窗口看门狗定时器 .....	728

6.9.1 概述.....	728
6.9.2 特性.....	728
6.9.3 框图.....	728
6.9.4 基本配置.....	728
6.9.5 功能描述.....	729
6.9.6 寄存器映射.....	733
6.9.7 寄存器描述.....	734
6.10 RTC 实时时钟.....	739
6.10.1概述.....	739
6.10.2特性.....	739
6.10.3框图.....	740
6.10.4基本配置.....	740
6.10.5功能描述.....	742
6.10.6寄存器映射.....	749
6.10.7寄存器描述.....	751
6.11 EPWM 发生器和捕获定时器 .....	786
6.11.1概述.....	786
6.11.2特性.....	786
6.11.3框图.....	787
6.11.4基本配置.....	791
6.11.5功能描述.....	793
6.11.6寄存器映射.....	825
6.11.7寄存器描述.....	831
6.12 BPWM 基本 PWM 发生器和捕获定时器 .....	900
6.12.1概述.....	900
6.12.2特性.....	900
6.12.3框图.....	901
6.12.4基本配置.....	904
6.12.5功能描述.....	905
6.12.6寄存器映射.....	919
6.12.7寄存器描述.....	922
6.13 QEI 正交编码接口.....	955
6.13.1概述.....	955
6.13.2特性.....	955
6.13.3框图.....	955
6.13.4基本配置.....	956
6.13.5功能描述.....	957
6.13.6寄存器映射.....	965
6.13.7寄存器描述.....	966
6.14 ECAP 增强型输入捕捉定时器.....	976

6.14.1 概述.....	976
6.14.2 特性.....	976
6.14.3 框图.....	976
6.14.4 基本配置.....	977
6.14.5 功能描述.....	978
6.14.6 寄存器映射.....	983
6.14.7 寄存器描述.....	984
<b>6.15 UART 接口控制器.....</b>	<b>996</b>
6.15.1 概述.....	996
6.15.2 特性.....	996
6.15.3 框图.....	997
6.15.4 基本配置.....	1000
6.15.5 功能描述.....	1004
6.15.6 寄存器映射.....	1029
6.15.7 寄存器描述.....	1031
<b>6.16 EMAC 以太网控制器.....</b>	<b>1066</b>
6.16.1 概述.....	1066
6.16.2 特性.....	1066
6.16.3 框图.....	1067
6.16.4 基本配置.....	1067
6.16.5 功能描述.....	1068
6.16.6 寄存器映射.....	1083
6.16.7 寄存器描述.....	1086
<b>6.17 智能卡主机接口.....</b>	<b>1138</b>
6.17.1 概述.....	1138
6.17.2 特征.....	1138
6.17.3 框图.....	1138
6.17.4 基本配置.....	1140
6.17.5 功能描述.....	1143
6.17.6 寄存器映射.....	1154
6.17.7 寄存器描述.....	1155
<b>6.18 I<sup>2</sup>S 控制器.....</b>	<b>1181</b>
6.18.1 概述.....	1181
6.18.2 特征.....	1181
6.18.3 框图.....	1181
6.18.4 基本配置.....	1181
6.18.5 功能描述.....	1182
6.18.6 寄存器映射.....	1193
6.18.7 寄存器描述.....	1194
<b>6.19 SPI 接口.....</b>	<b>1212</b>

6.19.1 概述.....	1212
6.19.2 特征.....	1212
6.19.3 框图.....	1213
6.19.4 基本配置.....	1213
6.19.5 功能描述.....	1217
6.19.6 时序图.....	1229
6.19.7 编程示例.....	1230
6.19.8 寄存器映射.....	1233
6.19.9 寄存器描述.....	1234
6.20 QSPI 接口 .....	1254
6.20.1 概述.....	1254
6.20.2 特征.....	1254
6.20.3 框图.....	1254
6.20.4 基本配置.....	1255
6.20.5 功能描述.....	1256
6.20.6 时序图.....	1272
6.20.7 编程示例.....	1273
6.20.8 寄存器映射.....	1276
6.20.9 寄存器描述.....	1277
6.21 SPIM 同步串行接口控制器 (SPI 主机模式).....	1291
6.21.1 概述.....	1291
6.21.2 特性.....	1291
6.21.3 框图.....	1292
6.21.4 基本配置.....	1293
6.21.5 功能描述.....	1294
6.21.6 SPI flash 控制器在 AMBA 系统中映射地址.....	1304
6.21.7 寄存器映射.....	1305
6.21.8 寄存器描述.....	1306
6.22 I2C 接口.....	1324
6.22.1 概述.....	1324
6.22.2 特性.....	1324
6.22.3 框图.....	1324
6.22.4 基本配置.....	1325
6.22.5 功能描述.....	1327
6.22.6 寄存器映射.....	1351
6.22.7 寄存器描述.....	1351
6.23 USCI – 通用串行接口 .....	1374
6.23.1 概述.....	1374
6.23.2 特性.....	1374
6.23.3 框图.....	1374

6.23.4 功能描述.....	1375
6.24 USCI – UART 模式.....	1393
6.24.1 概述.....	1393
6.24.2 特性.....	1393
6.24.3 框图.....	1393
6.24.4 基本配置.....	1394
6.24.5 功能描述.....	1395
6.24.6 寄存器映射.....	1411
6.24.7 寄存器描述.....	1412
6.25 USCI - SPI 模式 .....	1434
6.25.1 概述.....	1434
6.25.2 特性.....	1434
6.25.3 框图.....	1435
6.25.4 基本配置.....	1435
6.25.5 功能描述.....	1437
6.25.6 寄存器映射.....	1451
6.25.7 寄存器描述.....	1452
6.26 USCI - I <sup>2</sup> C 模式 .....	1475
6.26.1 概述.....	1475
6.26.2 特性.....	1475
6.26.3 框图.....	1475
6.26.4 基本配置.....	1476
6.26.5 功能描述.....	1477
6.26.6 寄存器映射.....	1501
6.26.7 寄存器描述.....	1502
6.27 CAN 接口 .....	1520
6.27.1 概述.....	1520
6.27.2 特性.....	1520
6.27.3 框图.....	1520
6.27.4 基本配置.....	1521
6.27.5 功能描述.....	1522
6.27.6 测试模式.....	1523
6.27.7 CAN 通信.....	1525
6.27.8 寄存器映射.....	1543
6.27.9 寄存器描述.....	1548
6.28 SD 卡主机接口 .....	1583
6.28.1 概述.....	1583
6.28.2 特性.....	1583
6.28.3 框图.....	1584

6.28.4 基本配置.....	1585
6.28.5 功能描述.....	1586
6.28.6 寄存器映射.....	1588
6.28.7 寄存器描述.....	1589
6.29 EBI 外部总线接口 .....	1609
6.29.1 概述.....	1609
6.29.2 特性.....	1609
6.29.3 框图.....	1610
6.29.4 基本配置.....	1610
6.29.5 功能描述.....	1612
6.29.6 寄存器映射.....	1622
6.29.7 寄存器描述.....	1623
6.30 USBD USB 1.1设备控制器.....	1627
6.30.1 特性.....	1627
6.30.2 框图.....	1627
6.30.3 基本配置.....	1627
6.30.4 功能描述.....	1628
6.30.5 寄存器映射.....	1632
6.30.6 寄存器描述.....	1636
6.31 HSUSBD 高速 USB2.0 设备接口 .....	1661
6.31.1 概述.....	1661
6.31.2 特性.....	1661
6.31.3 框图.....	1661
6.31.4 基本配置.....	1661
6.31.5 功能描述.....	1662
6.31.6 寄存器映射.....	1664
6.31.7 寄存器描述.....	1669
6.32 USBH USB2.0主机接口 .....	1718
6.32.1 概述.....	1718
6.32.2 特性.....	1718
6.32.3 框图.....	1719
6.32.4 基本配置.....	1719
6.32.5 功能描述.....	1720
6.32.6 寄存器映射.....	1722
6.32.7 寄存器描述.....	1724
6.33 USB OTG .....	1775
6.33.1 概述.....	1775
6.33.2 特性.....	1775
6.33.3 框图.....	1775

6.33.4 基本配置.....	1775
6.33.5 功能描述.....	1776
6.33.6 寄存器映射.....	1778
6.33.7 寄存器描述.....	1780
6.34 HSOTG 高速 USB OTG .....	1789
6.34.1 概述.....	1789
6.34.2 特性.....	1789
6.34.3 框图.....	1789
6.34.4 基本配置.....	1790
6.34.5 功能描述.....	1790
6.34.6 寄存器映射.....	1794
6.34.7 寄存器描述.....	1795
6.35 CRC 控制器.....	1804
6.35.1 概述.....	1804
6.35.2 特性.....	1804
6.35.3 框图.....	1804
6.35.4 基本配置.....	1805
6.35.5 功能描述.....	1805
6.35.6 寄存器映射.....	1807
6.35.7 寄存器描述.....	1808
6.36 CRYPTO 加解密算法加速器 .....	1813
6.36.1 概述.....	1813
6.36.2 特性.....	1813
6.36.3 框图.....	1814
6.36.4 基本配置.....	1815
6.36.5 功能描述.....	1815
6.36.6 寄存器映射.....	1838
6.36.7 寄存器描述.....	1848
6.37 EADC 增强型 12位模数转换器 .....	1915
6.37.1 概述.....	1915
6.37.2 特性.....	1915
6.37.3 框图.....	1916
6.37.4 基本配置.....	1916
6.37.5 功能描述.....	1917
6.37.6 寄存器映射.....	1931
6.37.7 寄存器描述.....	1934
6.38 DAC 数模转换器 .....	1964
6.38.1 概述.....	1964
6.38.2 特性.....	1964

6.38.3 框图.....	1965
6.38.4 基本配置.....	1965
6.38.5 功能描述.....	1966
6.38.6 寄存器映射.....	1971
6.38.7 寄存器描述.....	1972
6.39 ACMP 模拟比较器控制器 .....	1986
6.39.1 概述.....	1986
6.39.2 特性.....	1986
6.39.3 框图.....	1987
6.39.4 基本配置.....	1987
6.39.5 功能描述.....	1988
6.39.6 寄存器映射.....	1993
6.39.7 寄存器描述.....	1994
6.40 OPA 运算放大器.....	2001
6.40.1 概述.....	2001
6.40.2 特性.....	2001
6.40.3 框图.....	2002
6.40.4 基本配置.....	2002
6.40.5 功能描述.....	2003
6.40.6 寄存器映射.....	2005
6.40.7 寄存器描述.....	2006
<b>7 应用电路 .....</b>	<b>2013</b>
7.1 外接Vref电源线路图.....	2013
7.2 内置Vref电源线路图.....	2014
7.3 外接Vref及外接RTC线路图 .....	2015
7.4 外接Vref及内置RTC线路图 .....	2016
7.5 外设应用线路图 .....	2017
<b>8 封装尺寸 .....</b>	<b>2018</b>
8.1 QFN 33L (5x5x0.8 mm <sup>3</sup> Pitch 0.5 mm) .....	2018
8.2 LQFP 48L (7x7x1.4 mm <sup>3</sup> Footprint 2.0mm).....	2019
8.3 LQFP 64L (7x7x1.4 mm <sup>3</sup> footprint 2.0 mm).....	2021
8.4 LQFP 128L (14x14x1.4 mm <sup>3</sup> footprint 2.0 mm) .....	2022
8.5 LQFP 144L (20x20x1.4 mm <sup>3</sup> footprint 2.0 mm) .....	2023
<b>9 缩写 .....</b>	<b>2024</b>
9.1 缩写.....	2024
<b>10 版本历史 .....</b>	<b>2026</b>

## 图目录

图 4.1-1 NuMicro® M481 基本系列 QFN 33引脚图 .....	48
图 4.1-2 NuMicro® M481 基本系列 LQFP 48引脚图 .....	49
图 4.1-3 NuMicro® M481 基本系列 LQFP 64引脚图 .....	50
图 4.1-4 NuMicro® M482 USB FS OTG 系列 QFN 33引脚图.....	51
图 4.1-5 NuMicro® M482 USB FS OTG 系列 LQFP 48引脚图.....	52
图4.1-6 NuMicro® M482 USB FS OTG 系列 LQFP 64引脚图.....	53
图4.1-7 NuMicro® M482 USB FS OTG 系列 LQFP 128引脚图.....	54
图4.1-8 NuMicro® M483 CAN系列LQFP 64引脚图 .....	55
图4.1-9 NuMicro® M483 CAN系列LQFP 128引脚图 .....	56
图4.1-10 NuMicro® M484 USB HS OTG系列LQFP 64引脚图 .....	57
图4.1-11 NuMicro® M484 USB HS OTG系列LQFP 64引脚图 .....	58
图4.1-12 NuMicro® M484 USB HS OTG系列LQFP 128引脚图 .....	59
图4.1-13 NuMicro® M485 加解密系列QFN 33引脚图.....	60
图4.1-14 NuMicro® M485 加解密系列LQFP 48引脚图.....	61
图4.1-15 NuMicro® M485 加解密系列LQFP 64引脚图 .....	62
图4.1-16 NuMicro® M485 加解密系列LQFP 128引脚图 .....	63
图4.1-17 NuMicro® M487 以太网系列LQFP 64引脚图 .....	64
图4.1-18 NuMicro® M487 以太网系列LQFP 128引脚图 .....	65
图 4.1-19 NuMicro® M487 以太网系列LQFP 144引脚图 .....	66
图 5.1-1 NuMicro® M480 框图 .....	317
表 6.1-1 Cortex®-M4 框图 .....	318
图 6.2-1 系统复位源.....	322
图 6.2-2 nRESET 复位时序 .....	324
图 6.2-3 (POR) 上电复位波形.....	324
图 6.2-4 低电压复位 (LVR) 波形 .....	325
图 6.2-5 欠压侦测 (BOD) 波形 .....	326
图 6.2-6 NuMicro® M480 电源分布图 .....	327
图 6.2-7 电源模式状态机 .....	329
图 6.2-8 NuMicro® M480 电源模式图 .....	332
图 6.2-9 SRAM 框图 .....	335
图 6.2-10 SRAM 内存结构 .....	336
图 6.2-11 NuMicro® M480总线矩阵 .....	337
图 6.3-1 时钟发生器全局示意图 .....	438

图 6.3-2时钟发生器框图 .....	439
图 6.3-3系统时钟框图 .....	440
图 6.3-4 HXT停止保护过程 .....	441
图 6.3-5时钟控制框图 .....	441
图 6.3-6时钟输出框图 .....	442
图 6.3-7 USB 时钟源 .....	443
图 6.4-1存储器控制器框图 .....	496
图 6.4-2数据FLASH与APROM共享存储空间(128KB 和 256KB) .....	498
图6.4-3 SPROM 安全模式 .....	506
图 6.4-4 OTP 存储器映射 .....	506
图 6.4-5 KPROM 存储映射 .....	507
图 6.4-6 Flash 存储映射 .....	515
图 6.4-7支持IAP模式的系统存储映射 .....	517
图6.4-8支持IAP的LDROM 模式 .....	518
图 6.4-9支持IAP的APROM 模式 .....	518
图 6.4-10支持IAP的Boot Loader模式 .....	519
图 6.4-11不支持IAP模式的系统存储映射 .....	520
图6.4-12启动源选择 .....	521
图 6.4-13 ISP 流程示例 .....	524
图 6.4-14 ISP 32-bit编程流程 .....	525
图 6.4-15 ISP 64-bit编程流程 .....	526
图 6.4-16多字编程时间 .....	526
图 6.4-17固件(firmware)在SRAM中的多字编程 .....	527
图 6.4-18固件(firmware)在Boot Loader中的多字编程 .....	528
图 6.4-19多字编程流程 .....	529
图6.4-20快速Flash 编程校验流程 .....	530
图6.4-21校验流程 .....	531
图6.4-22 Flash CRC32 Checksum 计算 .....	531
图6.4-23 Flash 访问周期自动调整流程 .....	534
图6.4-24 Flash 安全密匙设置流程 .....	536
图6.4-25 KEY 比较流程 .....	538
图 6.5-1 GPIO控制器框图 .....	571
图 6.5-2推挽输出模式 .....	572
图 6.5-3开漏输出 .....	572

图 6.5-4准双向 I/O 模式.....	573
图 6.5-5 GPIO GPIO 上升沿触发中断 .....	574
图 6.5-6下降沿触发中断 .....	574
图 6.6-1 PDMA 控制器框图 .....	597
图 6.6-2描述符表入口结构.....	598
图 6.6-3基本模式的有限状态机 .....	599
图 6.6-4描述符表格链接单结构 .....	600
图 6.6-5 Scatter-Gather模式的有限状态机 .....	600
图 6.6-6基本模式的单字传输和批量传输举例 .....	601
图 6.6-7 PDMA 通道 0 超时计数器操作例子 .....	602
图 6.6-8 跨步功能块传输 .....	603
图 6.7-1定时器控制器框图.....	642
图 6.7-2定时器控制器的时钟源 .....	643
图 6.7-3 PWM 发生器框图.....	644
图 6.7-4 PWM 系统时钟源控制 .....	644
图 6.7-5 PWM 计数器时钟源控制.....	645
图 6.7-6 PWM 独立模式结构图 .....	645
图 6.7-7 PWM互补模式结构图 .....	646
图 6.7-8连续计数模式 .....	648
图 6.7-9外部捕获模式 .....	649
图 6.7-10外部复位计数器模式 .....	650
图 6.7-11内部定时器触发 .....	651
图 6.7-12定时器间互触发捕获时序 .....	652
图 6.7-13 PWM 上数计数模式的预分频波形 .....	652
图 6.7-14 PWM 上计数模式 .....	653
图 6.7-15 PWM 下计数模式 .....	653
图 6.7-16 PWM 上下计数模式 .....	654
图 6.7-17上下数模式PWM 的比较器事件 .....	655
图 6.7-18上数计数模式周期装载模式 .....	656
图 6.7-19上计数模式立即装载模式 .....	657
图 6.7-20在上下数模式PWM 脉冲的产生 .....	657
图 6.7-21在上数模式PWM 脉冲的产生 .....	658
图 6.7-22在下数模式PWM 脉冲的产生 .....	658
图 6.7-23上数模式和上下数模式PWM暂空比周期0%到100%的例子 .....	659

图 6.7-24 PWM 独立模式输出波形.....	659
图 6.7-25 PWM 互补模式输出波形.....	660
图 6.7-26独立模式下PWMx_CH0 输出控制 .....	660
图 6.7-27互补模式下PWMx_CH0 和 PWMx_CH1 输出控制 .....	660
图 6.7-28死区插入 .....	661
图 6.7-29 PWM 输出屏蔽控制波形.....	661
图 6.7-30刹车管脚噪音滤波器框图.....	662
图 6.7-31 PWMx_CH0 和 PWMx_CH1的刹车事件框图.....	663
图 6.7-32 PWMx_CH0 和 PWMx_CH1的边沿侦测器刹车波形.....	664
图 6.7-33 PWMx_CH0 和 PWMx_CH1的电平侦测器刹车波形.....	665
图 6.7-34刹车源框图.....	666
图 6.7-35系统失败刹车框图 .....	666
图 6.7-36 PWMx_CH0 和 PWMx_CH1 带死区插入的极性控制 .....	667
图 6.7-37 PWM 中断架构图 .....	668
图 6.7-38 PWM 触发 ADC 框图.....	668
图 6.8-1看门狗定时器框图.....	718
图 6.8-2看门狗定时器时钟控制 .....	719
图 6.8-3看门狗定时器定时溢出间隔和复位周期时序图 .....	720
图 6.9-1 WWDT 框图.....	728
图 6.9-2 WWDT 时钟控制 .....	729
图 6.9-3 WWDT 复位和重载动作 .....	730
图 6.9-4当 CNTDAT > CMPDATWWDT 重载计数器 .....	730
图 6.9-5当 WWDT_CNT < WINCMPWWDT 重载计数器 .....	731
图 6.9-6 WWDT 中断和复位信号 .....	731
图 6.10-1 RTC 框图 .....	740
图 6.10-2动态率定义 .....	746
图 6.10-3备用 I/O 控制图 .....	748
图 6.11-1 EPWM发生器简要方块图 .....	787
图 6.11-2 EPWM时钟源控制 .....	788
图 6.11-3 EPWM时钟源控制 .....	789
图 6.11-4 EPWM独立模式架构图 .....	790
图 6.11-5 EPWM互补模式架构图 .....	791
图 6.11-6 EPWM_CH0上计数模式的预分频器波形.....	793
图 6.11-7当设置清计数器 EPWMx计数器的波形 .....	794

图 6.11-8 EPWM上计数器模式.....	794
图 6.11-9 EPWM下计数器模式.....	795
图 6.11-10 EPWM 上下计数器模式.....	796
图 6.11-11 EPWM 在上下计数方式中的比较点事件.....	796
图 6.11-12 EPWM双缓存说明 .....	797
图 6.11-13向上计数方式的周期装载模式 .....	798
图 6.11-14向上计数方式的立即装载模式 .....	799
图 6.11-15向上计数模式窗口装载 .....	800
图 6.11-16上下计数方式的中心装载模式 .....	801
图 6.11-17 EPWM One-shot模式输出波形.....	802
图 6.11-18 EPWM 脉冲产生.....	803
图 6.11-19 EPWM 0% 到 100%占空比脉冲产生.....	803
图 6.11-20 EPWM独立模式波形.....	805
图 6.11-21 EPWM 互补模式波形.....	805
图 6.11-22 EPWM 组功能波形 .....	806
图 6.11-23 EPWM SYNC_IN 噪音滤波框图.....	807
图 6.11-24 EPWM计数器同步功能框图 .....	807
图 6.11-25 EPWM 同步源来自SYNC_IN 信号的同步功能 .....	808
图 6.11-26 EPWMx_CH0 独立模式下的输出控制 .....	809
图 6.11-27 EPWMx_CH0 和 EPWMx_CH1 在互补模式下的输出控制 .....	809
图 6.11-28死区插入 .....	810
图 6.11-29屏蔽控制波形图解.....	810
图 6.11-30刹车噪音滤波器方块图 .....	811
图 6.11-31 EPWMx_CH0 和 EPWMx_CH1 通道组 刹车方块图 .....	812
图 6.11-32 EPWMx_CH0 和 EPWMx_CH1通道组的边沿检测波形 .....	813
图 6.11-33 EPWMx_CH0 和 EPWMx_CH1通道组的电平检测波形 .....	813
图 6.11-34刹车源框图 .....	814
图 6.11-35系统故障刹车框图 .....	814
图 6.11-36 EPWM LEB 功能波形 .....	815
图 6.11-37带上升沿死区插入初始状态和极性控制 .....	816
图 6.11-38 EPWMx_CH0 累加中断波形.....	817
图 6.11-39 EPWMx_CH0 和 EPWMx_CH1组中断架构图 .....	818
图 6.11-40 EPWMx_CH0 和 EPWMx_CH1组触发EADC框图 .....	819
图 6.11-41上下计数方式EPWM触发EADC的时序波形 .....	820

图 6.11-42 EPWM_CH0 和 EPWM_CH1 组触发 DAC 框图 .....	820
图 6.11-43 EPWM_CH0 捕获框图 .....	821
图 6.11-44 捕获操作波形图 .....	822
图 6.11-45 通道0捕获PDMA 操作波形 .....	823
图 6.11-46 累加器 PDMA 功能架构 .....	824
图 6.12-1 BPWM 发生器框图 .....	901
图 6.12-2 BPWM 时钟源控制 .....	902
图 6.12-3 BPWM 时钟源控制 .....	902
图 6.12-4 BPWM 独立模式框架图 .....	903
图 6.12-5 BPWM_CH0 CLKPSC 波形 .....	905
图 6.12-6 BPWM 计数器清空波形 .....	906
图 6.12-7 BPWM Up 向上计数方式 .....	906
图 6.12-8 BPWM 向下计数方式 .....	907
图 6.12-9 BPWM 上下计数方式 .....	907
图 6.12-10 上下计数方式时 BPWM CMPDAT 事件 .....	908
图 6.12-11 向上计数方式下的周期载入模式 .....	909
图 6.12-12 向上计数方式下的立即载入模式 .....	910
图 6.12-13 上下计数方式下的中间载入模式 .....	911
图 6.12-14 BPWM 脉冲发生 (左: 不对称脉冲, 右: 多样脉冲) .....	912
图 6.12-15 BPWM 0% 到 100% 脉冲产生 (左: 上计数器模式, 右: 上下计数器模式) .....	912
图 6.12-16 BPWM_CH0 输出控制的 3 步 .....	913
图 6.12-17 屏蔽功能波形说明 .....	913
图 6.12-18 初始状态和极性控制 .....	914
图 6.12-19 BPWM_CH0 和 BPWM_CH1 的中断架构 .....	915
图 6.12-20 BPWM_CH0 和 BPWM_CH1 触发 EADC 源的框架图 .....	916
图 6.12-21 BPWM CH0~ CH5 触发 EADC 的框架图 .....	916
图 6.12-22 上下计数方式下 BPWM 触发 EADC 的时序波形图 .....	916
图 6.12-23 BPWM_CH0 捕捉框图 .....	917
图 6.12-24 捕捉操作波形图 .....	918
图 6.13-1 QEI 框图 .....	955
图 6.13-2 QEI 时钟控制 .....	956
图 6.13-3 噪声滤波器 .....	957
图 6.13-4 噪声滤波器采样时钟源选择 .....	958
图 6.13-5 QEA/QEB/IDX 通过噪声滤波器所需时序 .....	958

图 6.13-6 X4 计数模式.....	959
图 6.13-7 X2 计数模式.....	960
图 6.13-8 比较操作 .....	961
图 6.13-9 QEI_CNT 重载/复位控制 .....	962
图 6.13-10 捕捉 QEI 计数器的触发控制 .....	962
图 6.13-11 捕捉和锁存 QEI 计数器 .....	963
图 6.13-12 QEI 中断架构图 .....	964
图 6.14-1 输入捕捉定时器/计数器架构 .....	976
图 6.14-2 输入捕捉定时器/计数器时钟源控制 .....	978
图 6.14-3 噪声滤波器采样时钟选择 .....	979
图 6.14-4 输入捕捉定时器/计数器功能框图 .....	980
图 6.14-5 输入捕捉定时器/计数器中断架构图 .....	982
图 6.15-1 UART 时钟控制框图 .....	998
图 6.15-2 UART 结构图.....	999
图 6.15-3 自动波特率检测 .....	1008
图 6.15-4 发送延时操作 .....	1008
图 6.15-5 UART nCTS 唤醒示例 1 .....	1009
图 6.15-6 UART nCTS 唤醒示例 2 .....	1009
图 6.15-7 UART 数据唤醒 .....	1010
图 6.15-8 UART 接收数据 FIFO 达到阈值唤醒 .....	1010
图 6.15-9 UART RS-485 AAD 模式地址匹配唤醒 .....	1011
图 6.15-10 UART 数据接收 FIFO 超时唤醒 .....	1011
图 6.15-11 自动流控框图 .....	1015
图 6.15-12 UART nCTS 自动流控使能 .....	1015
图 6.15-13 UART nRTS 自动流控功能使能 .....	1016
图 6.15-14 UART nRTS 软件控制的自动流控 .....	1016
图 6.15-15 IrDA 控制框图 .....	1017
图 6.15-16 IrDA TX/RX 时序图 .....	1018
图 6.15-17 LIN 帧结构 .....	1018
图 6.15-18 LIN 字节结构 .....	1019
图 6.15-19 LINE 模式暂停检测 .....	1021
图 6.15-20 LIN 帧 ID 和校验格式 .....	1021
图 6.15-21 LIN 同步域测量 .....	1023
图 6.15-22 AR 模式下, SLVEUEN=1 时, UART_BAUD 更新顺序 .....	1024

图 6.15-23 AR 模式下, SLVEUEN=0 时, UART_BAUD 更新顺序0 .....	1024
图 6.15-24 RS-485 nRTS 自动方向控制模式下 nRTS 管脚驱动电平 .....	1026
图 6.15-25 RS-485 nRTS 软件控制时的驱动电平 .....	1027
图 6.15-26 RS-485 帧结构 .....	1028
图 6.16-1 EMAC 框图 .....	1067
图 6.16-2 以太网帧格式 .....	1069
图 6.16-3 64 位参考时间计数器 .....	1070
图 6.16-4 RXDMA 描述符数据结构 .....	1070
图 6.16-5 TXDMA 描述符数据结构 .....	1076
图 6.17-1 SC 时钟控制框图 (时钟控制器含 7 位的预分频计数器) .....	1139
图 6.17-2 SC 控制器框图 .....	1139
图 6.17-3 SC 数据格式 .....	1143
图 6.17-4 SC 激活序列 .....	1144
图 6.17-5 SC 热复位序列 .....	1145
图 6.17-6 SC 释放序列 .....	1146
图 6.17-7 基本操作流程 .....	1147
图 6.17-8 初始化字符 TS .....	1148
图 6.17-9 SC 错误信号 .....	1149
图 6.17-10 发送时块保护时间操作 .....	1152
图 6.17-11 接收时的块保护时间操作 .....	1152
图 6.17-12 扩展保护时间操作 .....	1152
图 6.18-1 I <sup>2</sup> S 控制器框架 .....	1181
表 6.18-2 I <sup>2</sup> C控制器配置 .....	1182
图 6.18-3 I <sup>2</sup> S 时钟控制框图 .....	1183
图 6.18-4 从机模式接口框图 .....	1183
图 6.18-5 I <sup>2</sup> S 通道宽度和数据宽度 (CHWIDTH≤DATWIDTH) .....	1184
图 6.18-6 I <sup>2</sup> S 通道宽度和数据宽度 (CHWIDTH > DATWIDTH) .....	1184
图 6.18-7 I <sup>2</sup> S 数据格式时序图 (FORMAT = 0x0 ; CHWIDTH≤DATWIDTH) .....	1185
图 6.18-8 MSB 对齐数据格式 (FORMAT = 0x1 ; CHWIDTH > DATWIDTH) .....	1185
图 6.18-9 LSB 对齐数据格式 (FORMAT = 0x2 ; CHWIDTH > DATWIDTH) .....	1185
图 6.18-10 标准 PCM 音频时序图 (FORMAT = 0x4 ; CHWIDTH≤DATWIDTH) .....	1186
图 6.18-11 MSB 对齐的 PCM 数据格式 (FORMAT = 0x5 ; CHWIDTH > DATWIDTH) .....	1186
图 6.18-12 LSB 对齐的 PCM 数据格式 (FORMAT = 0x6 ; CHWIDTH > DATWIDTH) .....	1186
图 6.18-13 用 32 位通道块传输 6 通道 24 位音频数据的 TDM 传输 (PCM 标准模式; FORMAT=0x4) .....	1187

图 6.18-14 用 32 位通道块传输 6 通道 24 位音频数据的 TDM 传输 (PCM MSB 对齐; FORMAT=0x5).....	1187
图 6.18-15 用 32 位通道块传输 6 通道 24 位音频数据的 TDM 传输 (PCM LSB 对齐; FORMAT=0x6).....	1187
图 6.18-16 I <sup>2</sup> S 中断.....	1188
图 6.18-17 不同设置下 2 通道音频数据的 FIFO 示意.....	1189
图 6.18-18 不同设置下 4 通道音频数据的 FIFO 示意.....	1190
图 6.18-19 不同设置下 6 通道音频数据的 FIFO 示意 (1) .....	1191
图 6.18-20 不同设置下 6 通道音频数据的 FIFO 示意 (2) .....	1192
图 6.20-1 QSPI 框架图 .....	1255
图 6.21-1 SPIM 方框图 .....	1292
图 6.21-2 SPIM 时序图 .....	1294
图 6.21-3 类型_1 四线模式写的编程命令模式.....	1296
图 6.21-4 类型_2四线模式写的编程命令模式.....	1296
图 6.21-5 类型_3四线模式写的编程命令模式.....	1297
图 6.21-6 连续读模式被禁用下命令为0xEB的快速读四线I/O .....	1298
图 6.21-7连续读模式使能下命令为0xEB的快速读四线 .....	1298
图 6.21-8 DTR/DDR 快速读命令 0x0D .....	1302
图 6.21-9 DTR/DDR快速读双线I/O命令0xBD .....	1302
图 6.21-10 DTR/DDR 快速读四线I/O命令0xED .....	1303
图 6.21-11 SPIM_SS 激活建立时间与SPIM_CLK 的关联和 SPIM_SS 取消选择时间.....	1321
图 6.22-1 I <sup>2</sup> C 控制器框图.....	1325
图 6.22-2 I <sup>2</sup> C 总线时序 .....	1327
图 6.22-3 I <sup>2</sup> C 协议 .....	1328
图 6.22-4 起始和停止状况 .....	1328
图 6.22-5 I <sup>2</sup> C 总线上的位传输 .....	1329
图 6.22-6 I <sup>2</sup> C 总线上的应答信号 .....	1329
图 6.22-7 7位地址情况下主机向从机传输数据 .....	1330
图 6.22-8 7位地址情况下主机向从机读取数据 .....	1330
图 6.22-9 10位地址情况下主机向从机传输数据 .....	1330
图 6.22-10 10位地址情况下主机向从机读取数据 .....	1331
图 6.22-11通过当前状态控制I <sup>2</sup> C 总线 .....	1331
图 6.22-12主机发送模式控制流程 .....	1332
图 6.22-13主机接收模式控制流程 .....	1333
图 6.22-14从机模式控制流程.....	1334

图 6.22-15 GC 模式 .....	1335
图 6.22-16 仲裁丢失 .....	1336
图 6.22-17 总线管理主要元素包协议图 .....	1338
图 6.22-18 7-位可寻址的设备与主机通信 .....	1339
图 6.22-19 7-位可寻址的设备对警报响应地址的响应 .....	1339
图 6.22-20 总线管理警报功能 .....	1340
图 6.22-21 系统管理总线超时时序 .....	1341
图 6.22-22 总线时钟低电平超时时序 .....	1341
图 6.22-23 建立时间错误调整 .....	1343
图 6.22-24 保持时间错误调整 .....	1343
图 6.22-25 I <sup>2</sup> C 数据移位方向 .....	1344
图 6.22-26 I <sup>2</sup> C 超时计数模块框图 .....	1346
图 6.22-27 I <sup>2</sup> C 唤醒的相关信号波形 .....	1347
图 6.22-28 EEPROM 随机读取 .....	1349
图 6.22-29 随机读取协议 .....	1349
图 6.23-1 框图 .....	1374
图 6.23-2 输入调节 USCI <sub>x</sub> _DAT[1:0] 和 USCI <sub>x</sub> _CTL[1:0] .....	1377
图 6.23-3 输入调节 USCI <sub>x</sub> _CLK .....	1377
图 6.23-4 数据缓冲框图 .....	1379
图 6.23-5 数据访问结构 .....	1381
图 6.23-6 传送数据路径 .....	1383
图 6.23-7 接收数据路径 .....	1386
图 6.23-8 相关协议时钟发生器 .....	1387
图 6.23-9 基本时钟分频计数器 .....	1388
图 6.23-10 时序测量计数器框图 .....	1389
图 6.23-11 采样时间计数器 .....	1390
图 6.23-12 事件和中断结构 .....	1391
图 6.24-1 USCI-UART 模式框图 .....	1394
图 6.24-2 全双工通信的 UART 信号连接 .....	1396
图 6.24-3 UART 标准帧格式 .....	1398
图 6.24-4 UART 位时序 (数据采样时间) .....	1404
图 6.24-5 UART 自动波特率控制 .....	1406
图 6.24-6 输入数据唤醒 .....	1407
图 6.24-7 nCTS 唤醒状态1 .....	1407

图 6.24-8 nCTS唤醒状态2 .....	1407
图 6.25-1 SPI 主机模式应用框图 .....	1434
图 6.25-2 SPI 从机模式应用框图 .....	1434
图 6.25-3 USCI SPI 模式框图 .....	1435
图 6.25-44-写 全双工 SPI 通信信号 (主机模式) .....	1437
图 6.25-54-写 全双工 SPI 通信信号 (从机模式) .....	1438
图 6.25-6 不同时钟模式下的SPI通信 (SCLKMODE=0x0) .....	1439
图 6.25-7不同时钟模式下的SPI通信(SCLKMODE=0x1) .....	1439
图 6.25-8不同时钟模式下的SPI通信(SCLKMODE=0x2) .....	1440
图 6.25-9不同时钟模式下的SPI通信(SCLKMODE=0x3) .....	1440
图 6.25-10 MSB优先格式下16位数据长度的一次字传输 .....	1441
图 6.25-11 两组传输间的字暂停 .....	1441
图 6.25-12 自动从机选择 (SUSPITV $\geq$ 0x3) .....	1442
图 6.25-13 自动从机选择(SUSPITV < 0x3) .....	1443
图 6.25-14 一个输出数据通道半双工通信 (SPI 主机模式) .....	1444
图 6.25-15一个输出数据通道半双工通信(SPI 主机模式) .....	1444
图 6.25-16 主机模式下的SPI 时序 .....	1447
图 6.25-17主机模式下的SPI 时序(串行总线时钟的反相位) .....	1448
图 6.25-18从机模式下的SPI 时序 .....	1448
图 6.25-19从机模式下的SPI 时序(串行总线时钟的反相位) .....	1449
图 6.26-1 I <sup>2</sup> C 总线时序 .....	1475
图 6.26-2 USCI I2C模式框图 .....	1476
图 6.26-3 I <sup>2</sup> C 协议 .....	1477
图 6.26-4起始和停止状况 .....	1478
图 6.26-5 I <sup>2</sup> C 总线上的位传输 .....	1479
图 6.26-6总线上的应答信号 .....	1479
图 6.26-7 仲裁丢失 .....	1481
图 6.26-8依据当I <sup>2</sup> C 状态控制I <sup>2</sup> C 总线 .....	1483
图 6.26-9 7位地址情况下主机向从机传输数据 .....	1484
图 6.26-10 7位地址情况下主机向从机读取数据 .....	1484
图 6.26-11 10位地址情况下主机向从机传输数据 .....	1484
图 6.26-12 10位地址情况下主机向从机读取数据 .....	1485
图 6.26-13 7位地址主机发送模式控制流程 .....	1486
图 6.26-14 7位地址主机接收模式控制流程 .....	1487

图 6.26-15 10位地址主机发送模式控制流程 .....	1488
图 6.26-16 10位地址主机接收模式控制流程 .....	1489
图 6.26-17 7位地址从机模式控制流程 .....	1491
图 6.26-18 10位地址从机模式控制流程 .....	1492
图 6.26-19 7位地址的GC模式 .....	1493
图 6.26-20 保持时间错误调整 .....	1496
图 6.26-21 I <sup>2</sup> C 超时计数模块框图 .....	1497
图 6.26-22 EEPROM 随机读取 .....	1499
图 6.26-23 EEPROM随机读取协议 .....	1500
图 6.27-1 CAN 外设框图 .....	1521
图 6.27-2 CAN内核静默模式 .....	1524
图 6.27-3 CAN 内核回环模式 .....	1524
图 6.27-4 CAN内核回环模式与静默模式组合 .....	1525
图 6.27-5 IFn寄存器和报文RAM间的数据传输 .....	1527
图 6.27-6 应用软件处理FIFO缓存 .....	1532
图 6.27-7 位时间 .....	1534
图 6.27-8 传播时间段 .....	1535
图 6.27-9 同步在“late”及“early”边沿 .....	1536
图 6.27-10 过滤显性干扰短脉冲 .....	1537
图 6.27-11 CAN内核CAN协议控制器架构 .....	1539
图 6.28-1 SD主机控制器框图 .....	1584
图 6.28-2 PAD (物理地址描述符) 表格式 .....	1587
图 6.29-1 EBI 框图 .....	1610
图 6.29-2 16-位 EBI 数据宽度与16-位设备的连接 .....	1613
图 6.29-3 8-位 EBI 数据宽度与8-位设备的连接 .....	1613
图 6.29-4 分离模式下16-位 EBI 数据宽度与16-位设备的连接 .....	1614
图 6.29-5 分离模式下8-位 EBI 数据宽度与8-位设备的连接 .....	1614
图 6.29-6 16-位数据宽度的时序控制波形 .....	1616
图 6.29-7 8-位数据宽度时序控制波形 .....	1617
图 6.29-8 16位数据宽度字节写的时序控制波形 .....	1618
图 6.29-9 插入空闲周期的时序控制 .....	1619
图 6.29-10 16位数据宽度在分离模式下的时序控制波形 .....	1620
图 6.29-11 连续数据访问模式的时序控制波形 .....	1621
图 6.30-1 USB 框图 .....	1627

图 6.30-2 唤醒中断操作流程 .....	1629
图 6.30-3 端点SRAM 结构 .....	1630
图 6.30-4 Setup 事务后接着是Data In事务 .....	1631
图 6.30-5 数据输出 .....	1631
图 6.30-6 LPM 状态转换图 .....	1632
图 6.31-1 USB设备控制器框图 .....	1661
图 6.32-1 USB 2.0 FS 主机控制器框图 .....	1719
图 6.33-1 USB OTG 框图 .....	1775
图 6.33-2 USB 设备模式 .....	1776
图 6.33-3 USB 主机模式 .....	1777
图 6.34-1 USB OTG 框图 .....	1790
图 6.34-2 USB设备模式 .....	1791
图 6.34-3 USB主机模式 .....	1792
图 6.35-1 CRC 发生器框图 .....	1805
图 6.35-2 校验和位顺序反转功能框图 .....	1806
图 6.35-3 写数据位顺序反转功能框图 .....	1806
图 6.36-1 加解算法加速器框图 .....	1814
图 6.36-2 PRNG功能框图 .....	1817
图 6.36-3 电子密码本模式 .....	1818
图 6.36-4 密码块链模式 .....	1819
图 6.36-5 密码反馈模式 .....	1820
图 6.36-6 输出反馈模式 .....	1821
图 6.36-7 计数器模式 .....	1822
图 6.36-8 CBC-CS1 加密 .....	1822
图 6.36-9 CBC-CS1 解密 .....	1823
图 6.37-1 ADC 框图 .....	1916
图 6.37-2 模块 0~3 框图 .....	1918
图 6.37-3 模块 4~15 框图 .....	1919
图 6.37-4 模块 16~18 框图 .....	1919
图 6.37-5 EADC时钟控制 .....	1920
图 6.37-6 ADC 转换时序图, n=0~18 .....	1921
图 6.37-7 模块转换优先级仲裁框图 .....	1922
图 6.37-8 选定模块ADC EOC 信号 ADINT0~3中断 .....	1924
图 6.37-9 EPWM触发 ADC开始转换 .....	1924

图 6.37-10 外部触发 ADC 开始转换 .....	1925
图 6.37-11 转换开始延时时间 .....	1926
图 6.37-12 EADC0_ST 防抖时序 .....	1926
图 6.37-13 ADC 延长采样时间时序 .....	1927
图 6.37-14 ADC 转换结果监控逻辑框图 .....	1928
图 6.37-15 ADC 控制器中断 .....	1929
图 6.37-16 ADC 带校准的启动序列 .....	1930
图 6.38-1 数模转换器框图 .....	1965
图 6.38-2 数据保持寄存器格式 .....	1967
图 6.38-3 DAC 转换由软件写触发 .....	1967
图 6.38-4 DAC 转换由硬件触发事件开始 .....	1968
图 6.38-5 DAC0 和 DAC1 组模式和非组模式更新示例 .....	1968
图 6.38-6 DAC PDMA 下溢条件示例 .....	1969
图 6.38-7 软件 PDMA 模式 DAC 连续转换 .....	1969
图 6.38-8 DAC 中断源 .....	1970
图 6.39-1 模拟比较器框图 .....	1987
图 6.39-2 ACMP0 比较器迟滞回差功能 .....	1988
图 6.39-3 窗口锁存模式 .....	1989
图 6.39-4 滤波功能示例 .....	1989
图 6.39-5 比较器控制器中断 .....	1990
图 6.39-6 比较器参考电压框图 .....	1990
图 6.39-7 窗口比较模式示例 .....	1991
图 6.39-8 窗口比较模式示例 .....	1992
图 6.40-1 运算放大器框图 .....	2002

## 表目录

表 6.2-1 寄存器复位值 .....	324
表 6.2-2 电源模式表 .....	328
表 6.2-3 电源模式区别表 .....	328
表 6.2-4 电源模式定义表 .....	329
表 6.2-5 不同电源模式的时钟 .....	329
表 6.2-6 重新进入掉电状态的条件 .....	331
表 6.2-7 片上地址空间分布 .....	335
表 6.2-8 异常类型 .....	399
表 6.2-9 中断向量表 .....	402
表 6.2-10 优先级分组 .....	432
表 6.3-1 PLL 输出频率公式的符号定义 .....	470
表 6.4-1 双-Bank 区块地址范围 .....	498
表 6.4-2 向量映射支持情况 .....	521
表 6.4-3 ISP 命令表 .....	523
表 6.4-4 FMC 用于 Flash 编程的控制寄存器 .....	525
表 6.4-5 在自动调整功能下 Flash 访问最优周期数 .....	533
表 6.4-6 对 CPU 的四种保护的锁定效果表 .....	539
表 6.4-7 对 ICE/ ICP/Writer 的四种保护的锁定效果表 .....	540
表 6.5-1 消抖功能设置表 .....	575
表 6.6-1 通道优先级表 .....	599
表 6.7-1 TIMER01 引脚配置 .....	647
表 6.7-2 TIMER23 引脚配置 .....	647
表 6.7-3 上数计数模式中 PWM 脉冲产生事件优先级 .....	658
表 6.7-4 下数计数模式中 PWM 脉冲产生事件优先级 .....	659
表 6.7-5 上下数计数模式中 PWM 脉冲产生事件优先级 .....	659
表 6.8-1 看门狗定时器定时溢出间隔周期选择 .....	720
表 6.9-1 WWDT 预分频值选择 .....	729
表 6.9-2 CMPDAT 设置限制 .....	732
表 6.10-1 RTC 读/写 使能 .....	742
表 6.10-2 12/24 小时制选择 .....	743
表 6.10-3 上电后寄存器值 .....	745
表 6.10-4 寄存器电源域 .....	745
表 6.10-5 动态样本源选择 .....	746

表 6.10-6 Tamper 控制位效果位对0对的影响 .....	746
表 6.10-7 Tamper 控制位效果位对1和2对的影响.....	747
表 6.11-1 EPWM时钟源控制寄存器设置表 .....	788
表 6.11-2向上计数器EPWM脉冲发生事件优先级.....	804
表 6.11-3向下计数器EPWM脉冲发生事件优先级.....	804
表 6.11-4上下计数器EPWM脉冲产生事件优先级.....	804
表 6.12-1 BPWM 时钟源控制寄存器设置 .....	903
表 6.12-2 向上计数方式下的 BPWM 脉冲发生事件优先级.....	912
表 6.12-3 向下计数方式的 BPWM 脉冲发生事件优先级 .....	912
表 6.12-4 上下计数方式的 BPWM 脉冲发生事件优先级 .....	912
表 6.13-1 计数方向 .....	961
表 6.14-1 一个典型的滤波设置 .....	978
表 6.15-1 NuMicro™ M480 系列 UART 特性.....	997
表 6.15-2 UART 中断 .....	1000
表 6.15-3 UART 接口控制器.....	1004
表 6.15-4 UART 控制器波特率公式表 .....	1005
表 6.15-5 UART 控制器波特率参数设置示例 .....	1005
表 6.15-6 UART 控制器波特率寄存器设置示例.....	1006
表 6.15-7 波特率补偿示例 1 .....	1006
表 6.15-8 波特率补偿示例 2 .....	1007
表 6.15-9 UART 控制器中断源和标志位列表 .....	1013
表 6.15-10 UART 线控的数据位和停止位长度设置 .....	1014
表 6.15-11 UART 线控校验位设置 .....	1014
表 6.15-12 LIN 主机模式报头选择 .....	1019
表 6.16-1 仲裁器仲裁结果 .....	1068
表 6.16-2 RXDMA 描述符字 1 (TSEN (EMAC_TSCTL[0]) 为 0).....	1073
表 6.16-3 RXDMA 描述符字 1 (TSEN (EMAC_TSCTL[0]) 为 1).....	1073
表 6.16-4 RXDMA 描述符字 2 .....	1074
表 6.16-5 RXDMA 描述符字 3 (TSEN (EMAC_TSCTL[0]) 为 0).....	1075
表 6.16-6 RXDMA 描述符字 3 (TSEN (EMAC_TSCTL[0]) 为 1).....	1075
表 6.16-7 TXDMA 描述符字 0 .....	1078
表 6.16-8 TXDMA 描述符字 1 (TSEN (EMAC_TSCTL[0]) is 0) .....	1079
表 6.16-9 TXDMA 描述符字 1 (TSEN (EMAC_TSCTL[0]) is 1) .....	1079
表 6.16-10 TXDMA 描述符字 2 .....	1081

表 6.16-11 TXDMA 描述符字 3 (TSEN (EMAC_TSCTL[0]) 为 0) .....	1082
表 6.16-12 TXDMA 描述符字 3 (TSEN (EMAC_TSCTL[0]) 为 1) .....	1082
表 6.16-13 不同 CAMCMR 设置及可接收的包类型 .....	1088
表 6.16-14 MII 管理帧格式 .....	1103
表 6.16-15 MII 管理功能配置时序 .....	1103
表 6.17-1 SC 主控制器管脚描述 .....	1140
表 6.17-2 UART 管脚描述 .....	1140
表 6.17-3 Timer0/Timer1/Timer2 操作模式 .....	1151
表 6.19-1 SPI 特性对比 .....	1213
表 6.19-2 SPI/I <sup>2</sup> S 接口管脚描述 (SPI0~SPI3) .....	1217
表 6.21-1 SPI flash 控制器在 AMBA 系统中映射地址 .....	1304
表 6.21-2 从机片选电平和激活使能的功能描述 .....	1310
表 6.22-1 SMBus 保留地址 .....	1337
表 6.22-2 I <sup>2</sup> C 传输速率与 PCLK 之间的关系 .....	1342
表 6.22-3 I <sup>2</sup> C 状态码描述 .....	1345
表 6.23-1 不同协议的输入信号 .....	1376
表 6.23-2 不同协议输出信号 .....	1379
表 6.23-3 数据传输事件和中断处理 .....	1391
表 6.23-4 具体协议事件和中断处理 .....	1392
表 6.24-1 UART 协议输入信号 .....	1397
表 6.24-2 UART 协议的输出信号 .....	1398
表 6.25-1 SPI 通信信号 .....	1437
表 6.25-2 串行总线时钟配置 .....	1438
表 6.26-1 I <sup>2</sup> C 传输速率与 PCLK 之间的关系 .....	1496
图 6.26-2 建立时间错误调整 .....	1496
表 6.27-1 初始化传送对象 .....	1529
表 6.27-2 初始化接收对象 .....	1530
表 6.27-3 CAN 位时间参数 .....	1534
表 6.27-4 CAN 寄存器中每个位的功能 .....	1547
表 6.27-5 错误码 .....	1552
表 6.27-6 中断源 .....	1555
表 6.27-7 IF1 及 IF2 报文接口寄存器 .....	1558
表 6.27-8 报文对象在内存中的结构 .....	1572
表 6.28-1 SD0 引脚配置 .....	1585

表 6.28-2 SD1 引脚配置 .....	1586
表 6.29-1 EBI 地址映射 .....	1612
表 6.29-2 时序控制参数 .....	1615
表 6.30-1 USB 链路电源管理 (Lx) 状态 .....	1632
SGEN (HSUSBD_DMACTL[6]) =1使能 DMA 分散收集功能，然后设置 HSUSBD_DMACNT 为8个字节，内存地址和长度会在描述符里重排序，格式如表 6.31-1所示.....	1663
表 6.31-2 分散-收集描述符格式.....	1663
表 6.36-1每个引擎错误条件和错误标识 .....	1815
表 6.36-2 DMA使能位列表 .....	1816
表 6.36-3 DMA 串流位表 .....	1816
表 6.36-4 通道选择位表 .....	1817
表 6.36-5 ECC参数和相应寄存器列表.....	1828
表 6.36-6各种操作所需的输入数据 .....	1829
表 6.36-7轻量级二进制不可约多项式 .....	1833
表 6.37-1解析度与转换周期的关系 .....	1922
表 6.37-2 EADC差分模式通道选择.....	1928
表 6.37-3 EADC省电模式 .....	1929
表 6.37-4 EADC校准启动 .....	1930
表 6.39-1窗口比较逻辑真值表 .....	1991
表 9.1-1 缩写表.....	2025

## 1 概述

NuMicro® M480高效能低功耗的ARM® Cortex®-M4微控制器，支持DSP指令集。工作频率最高可达到192 MHz，最大工作电流为 175 $\mu$ A/ MHz。快闪记忆体（Flash）最大容量为 512 KB，采用双区块（双银行）架构的快闪记忆体支持Over-The-Air升级程序，在系统运行时可以同时更新韧体。SRAM最大容量为 160 KB，其中 32 KB的SRAM 支持快取机制，加速运行外部 SPI Flash 的程序。新导入安全启动（Secure Boot）功能，在开机过程中对韧体进行扫描及验证，一旦发现内容被恶意窜改将立即停止运行内建 4 KB Secure Protection ROM提供一个安全的空间。支持1.8V-3.63V宽电压供电，提供工业级-40°C至105°C的工作温度。

NuMicro® M480整合了多样化的高速传输介面，配置 USB 2.0 高速介面，内建物理层（收发器）并提供装置，主机，OTG等传输模式，配置 USB 2.0 全速介面，内建物理层 提供装置，主机，OTG等传输模式。9个UART介面，其中包含3个智能卡介面。4个复合的SPI / I<sup>2</sup>S介面。1个四位SPI介面。1个SPI Flash介面。3个I<sup>2</sup>C介面。1个I<sup>2</sup>S高传真音效介面。2个SDIO介面。2个CAN 2.0B介面。2个用于马达控制的正交编码器（QEI）介面。1个10/100 Mbps乙太网控制器（MAC） 及2个USCI介面，可自由调配为UART，SPI或I<sup>2</sup>C。支持16路的周边DMA控制器及多达32路PWM（192 MHz / 16位）输出。

提供多种高性能类比周边设备，包括1个12位元16路5MSPS采样率的SAR ADC，2个12位元1MSPS采样率的DAC，2个比较器（Comparator）与3个运算放大器（运算放大器）。

NuMicro® M480内建一个内建硬体加密引擎（包含ECC，AES，DES，3DES，SHA及HMAC），一个乱数产生器（RNG）能产随机金钥

NuMicro® M480全系列包含

- NuMicro® M481Base系列：高性能，低动态功耗，高整合度支持安全启动（Secure Boot）适合感测器集线器应用
- NuMicro® M482USB FS OTG系列：支持USB 2.0全速OTG PHY，并提供装置，主机，OTG等传输模式
- NuMicro® M483CAN系列：提供2组CAN 2.0B总线及
- NuMicro® M484USB HS OTG系列：同时USB 2.0全速OTG PHY，并提供装置，主机，OTG等传输模式
- NuMicro® M485加密系列：内建硬体加密引擎及乱数产生器（RNG）
- NuMicro® M487以太网系列：内建10 / 100M乙太网控制器（MAC），支持RMII，MDC，MDIO，可快速实现网路连接

## 2 特性

### 2.1 NuMicro® M480 特性

#### 核心和系统

<b>ARM® Cortex®-M4</b>	<ul style="list-style-type: none"> <li>- ARM® Cortex®-M4F , 主频 192MHz</li> <li>- 内存保护单元 (MPU)</li> <li>- 16 级中断优先级 (NVIC)</li> <li>- IEEE 754 兼容浮点运算单元(FPU)</li> <li>- DSP 指令扩展, 带硬件除法器及单周期 32 硬件乘法器</li> <li>- 24位系统定时器</li> <li>- 支持可屏蔽中断</li> <li>- 通过WFI/WFE指令指令, 支持低功耗休眠功能</li> </ul>
<b>BOD 欠压检测</b>	<ul style="list-style-type: none"> <li>- 8个电压选择支持中断或复位功能: 3.0V/2.8V/2.6V/2.4V/2.2V/2.0V/1.8V/1.6V</li> </ul>
<b>低压复位</b>	<ul style="list-style-type: none"> <li>- 电压门限: 1.5 V</li> </ul>
<b>安全</b>	<ul style="list-style-type: none"> <li>- 96位唯一序列号 (UID)</li> <li>- 128位用户代码 (UCID)</li> <li>- 内建温度传感器, 精度 1°C</li> </ul>

#### 记忆装置

<b>Boot loader</b>	<ul style="list-style-type: none"> <li>- 32 KB保护启动程序</li> <li>- 支持安全启动功能 SHA-256 和 AES-256 从 APROM, LDROM 和外部 SPI flash</li> <li>- ISP功能支持接口: UART 和 USB</li> <li>- 支持 ISP/IAP 库</li> </ul>
<b>内嵌闪存</b>	<ul style="list-style-type: none"> <li>- 双 bank 512 KB/256 KB 应用代码空间(APROM) 用于代码安全升级</li> <li>- 在连续地址读访问中最大可运行到零等待的192 MHz</li> <li>- 4 KB 加载代码空间 (LDROM)</li> <li>- 8 KB 密钥保护存储空间 (KPROM) 保护固件编程.</li> <li>- 4 K字节安全保护内存 (SPROM) 保护知识产权</li> <li>- 2 K字节单次编程存储器 (OTP) 保护数据安全.</li> <li>- 擦除页大小: 4 KB</li> <li>- 支持快速编程CRC校验.</li> <li>- 支持在系统编程 (ISP), 和在应用编程(IAP)</li> <li>- 可配置启动方式, 包括Boot Loader, 加载代码空间 (LDROM) 或应用代码空间(APROM)</li> <li>- 数据空间大小可配置</li> <li>- 2线 SWD 编程接口</li> <li>- 支持32-bit、64-bit和多字编程功能.</li> </ul>
<b>SRAM内存</b>	<ul style="list-style-type: none"> <li>- 内嵌160/96 KB SRAM</li> <li>- SRAM 分成三部分: bank0 (32 KB), bank1 (96/32 KB) 和 bank2 (32 KB)</li> <li>- Bank0支持硬件奇偶校验并有保持模式</li> </ul>

**CRC 计算单元**

- Bank2 能切换到 SPI Flash 高速缓存cache
- 支持字节、半字、字访问
- 有 PDMA 模式
- 支持四个常用多项式 CRC-CCITT, CRC-8, CRC-16, 和 CRC-32
- 初值及种子值可配
- 支持反向运算
- 支持补码运算
- 支持 8/16/32位运算
- 支持 8位写模式: 1-AHB 时钟操作
- 支持 16位写模式: 2-AHB 时钟操作
- 支持 32位写模式: 4-AHB 时钟操作
- 支持 DMA 完成运算

**PDMA (外设DMA)**

- 16 个传输通道
- 支持基本和分散聚集(catter-Gather)传输模式
- 分散聚集(catter-Gather)传输模式支持环形缓冲管理
- 支持固定模式和循环模式优先级
- 支持一次或突发传输
- 支持字节、半字，字传输
- 源地址和目标地址都支持递增或固定.

**时钟****外部时钟控制**

- 可外接 4~24 MHz 晶体 (HXT)
- 可外接 32.768 kHz 晶体 (LXT) 用于 RTC 和低功耗系统时钟
- 有晶体失效检测功能
- 时钟失效可配置为非屏蔽中断

**内置时钟控制**

- 内建 12 MHz RC 时钟 (HIRC) 精度±2%
- 内建 10 kHz RC时钟 (LIRC) 用于WDT和唤醒功能
- 内建 PLL 可把HIRC或HXT倍频到 480 MHz

**RTC 实时时钟**

- 电池引脚
- RTC时钟源包括低速外部晶体振荡器
- 可保存80 字节数据，并有6个数据清除引脚
- 支持6个可选的静态和动态数据清除引脚
- 可唤醒空闲模式，掉电模式，待命模式，深度休眠模式
- 时钟补偿功能，可在5秒内达到±5ppm 精度
- 有报警功能 (秒分时，日月年)
- RTC定时和报警支持中断
- 闰年计算
- 有 1 Hz时钟输出

**计时器****定时器控制器****TMR 定时器控制器**

- 4 组 32-位定时器，带24位向上计数器和一个8位的预分频计数器
- 提供 单次, 周期, 触发输出 和 连续计数 四种操作模式
- 支持外部管脚捕获事件计数功能
- 支持外部引脚捕获，可用于复位24位向上定时器

- 如果定时器中断信号产生，支持芯片从空闲/掉电模式唤醒

**PWM**

- 8 个独立PWM 输出，16位计数器，12位预分频，最大时钟 192MHz
- 12位死区时间
- 计数方式有：上计数，下计数，上/下计数
- 支持刹车功能
- 对每一个PWM输出管脚支持屏蔽功能和三态使能

**EPWM**

- 12 个独立PWM 输出，16位计数器，12位预分频，最大时钟 192MHz
- 可配成12个输入捕获，16位时间值
- 12位死区时间
- 计数方式有：上计数，下计数，上/下计数
- 可配置成3对互补模式PWM输出
- 有相位同步功能
- 支持刹车功能可自恢复
- 对每一个PWM输出管脚支持屏蔽功能和三态使能
- 可触发 EADC/DAC 启动转换

**BPWM**

- 两个 BPWM模块，支持16位计数器，12位预分频器，最大时钟 192MHz
- 每個模块支持六路 PWM 输出，也可配置成六路输出捕获
- 计数方式有：上计数，下计数，上/下计数
- 同步计数功能
- 可配置成3对互补模式PWM输出
- 有掩码可屏蔽输出
- 可触发 EADC启动转换

**WDT 看门狗**

- 18向上计数器
- 时钟源： LIRC (默认选择), HCLK/2048 和 LXT
- 8 种时间选择 1.6ms ~ 26.0sec (看所选时钟)
- 时钟选LIRC或LXT时，支持唤醒功能
- 时间到后，可复位，即可仅产生中断
- 可选复位延时： 1026、130、18 或 3 WDT\_CLK
- 支持上电复位后即启动计数功能

**WWDT 看门狗**

- 时钟源： HCLK/2048 (默认选择) 和 LIRC
- 带11位预分频器的6位计数器
- 空闲和掉电模式下停止计数

**类比接口****EADC**

- 一组12位精度，19路ADC 输入，采样率5 MSPS
- 输入电压范围: 0~ VREF (最大 AVDD)
- 单端输入模式 16 个通道或差分 8 个通道
- 保证至少 10位精度
- 3个内部通道: V<sub>DD</sub>, 带隙电压, 温度传感器
- 片内参考电压: 1.6V, 2.0V, 2.5V, 和 3.0V.
- 两种节电模式:掉电模式，待命模式
- 支持校正功能.
- 启动转换触发源： 软件，引脚， Timer 0~3 溢出，EPWM 和

	<ul style="list-style-type: none"> <li>- BPWM.</li> <li>- 采样保持时间长短可编程</li> <li>- 19 采样模块</li> <li>- 模块 0~3双缓存</li> <li>- 支持 PDMA 控制</li> </ul>
DAC	<ul style="list-style-type: none"> <li>- 两个 12位DAC, 1MSPS 转换速率</li> <li>- 支持8位模式</li> <li>- 建立时间 8us</li> <li>- 缓存模式最大输出电压: AVDD -0.2V</li> <li>- 启动转换触发源 : 软件, Timer 0~3, EPWM 和DAC外部触发引脚.</li> <li>- 两路 DAC可同步转换</li> <li>- 支持 PDMA 模式</li> </ul>
模拟比较器	<ul style="list-style-type: none"> <li>- 两个轨对轨比较器</li> <li>- 正输入端有四个引脚选择.</li> <li>- 负输入端可选: 引脚、带隙电压源、16级AVDD分压、 VREF 和 DAC 输出</li> <li>- 可选低功耗模式, 速度可编程</li> <li>- 输出变化时可产生中断</li> <li>- 支持唤醒功能</li> <li>- 支持 PWM 的Brake功能, 可逐周期控制 PWM</li> <li>- 支持窗比较模式和窗锁存功能</li> <li>- 斯密特回差电压可编程: 0mV, 10mV, 20mV 和 30mV.</li> </ul>
可选运放	<ul style="list-style-type: none"> <li>- 3 个运放</li> <li>- 输出可通过一个斯密特缓冲后触发中断</li> </ul>
<b>通讯接口</b>	
UART	<ul style="list-style-type: none"> <li>- 六路UART, 波特率最高 17.45MHz</li> <li>- 波特率自动测量功能和补偿功能</li> <li>- 支持低功耗模式, 在系统时钟关闭时, 仍可用 LXT(32.768 KHz) 以 9600bps 工作</li> <li>- 16字节 FIFOs, 可配置触发字节数</li> <li>- 支持硬件流控 CTS, RTS</li> <li>- 支持红外模式 IrDA</li> <li>- UART0, UART1 支持 LIN 功能</li> <li>- 支持 RS-485 9位模式和方向控制</li> <li>- 支持在空闲模式唤醒功能: nCTS唤醒, Rx唤醒, 接收FIFO数据个数到门限值唤醒, RS-485 地址匹配唤醒.</li> <li>- 支持硬件或软件把 RTS 引脚配置成485的收发自动控制</li> <li>- 支持唤醒功能</li> <li>- 可配置8位接收 FIFO 超时定时器</li> <li>- 支持Break错, 帧错, 奇偶校验错, FiFo溢出 检测功能</li> <li>- 支持 PDMA 收发</li> </ul>
智能卡接口	<ul style="list-style-type: none"> <li>- 三个 ISO-7816-3 接口, 兼容 ISO-7816-3 T=0, T=1</li> <li>- 支持 UART功能</li> <li>- 收发各4字节缓存</li> <li>- 位时长可配置 (11 ETU ~ 266 ETU)</li> <li>- 一个24位定时器两个8位定时器可用于复位应答序列 (ATR) 和等待</li> </ul>

	<ul style="list-style-type: none"><li>- 时间定时</li><li>- 支持反向约定</li><li>- 出错后重发功能</li><li>- 硬件附着/分离序列处理功能</li><li>- 硬件热复位序列</li><li>- 拔卡后有硬件分离功能</li></ul>
<b>I2C</b>	<ul style="list-style-type: none"><li>- 3路 I2C，支持主从模式</li><li>- 支持标准速率 (100 kbps), 快速 (400 kbps) 和快速加模式 (1 Mbps) 和高速模式 (3.4 Mbps)</li><li>- 支持10位模式</li><li>- 支持多种速率</li><li>- 从机支持多个地址 (四个从机地址加掩码)</li><li>- 支持 SMBus , PMBus</li><li>- 支持多地址掉电唤醒</li><li>- 支持 DMA传输</li></ul>
<b>SPI 主机</b>	<ul style="list-style-type: none"><li>- 外接SPI 闪存最大32M 字节，支持标准的1位、2位和4位I/O传输模式，最高 96 Mbit/s.</li><li>- 有32KB高速缓存 cache</li><li>- 支持 16位金钥保护代码</li><li>- 代码从SPI到RAM内存间互传输，支持DMA传输</li><li>- 支持 CPU 直接从 SPI 闪存读.</li><li>- 支持 SPI 主机协议，位长可配置为8, 16, 24, 32</li><li>- 突发模式可完成四次连续收发</li></ul>
<b>Quad SPI</b>	<ul style="list-style-type: none"><li>- 一个四线 SPI控制器，支持主从模式，在VDD =2.7V~3.6V最高 96 Mbit/s.</li><li>- 支持2线和4线传输</li><li>- 支持两路半双工传输</li><li>- 支持仅接收模式</li><li>- 收发位长可配置为 8 ~ 32</li><li>- 收发各8级FiFo</li><li>- 高低位在前可配置</li><li>- 支持字节重排序功能</li><li>- 支持字节或字挂起功能</li><li>- 支持 3线模式，无片选</li><li>- 支持 PDMA 模式</li></ul>
<b>SPI</b>	<ul style="list-style-type: none"><li>- 四个 SPI/ I<sup>2</sup>S 控制器，支持主从模式，</li><li>- SPI/ I<sup>2</sup>S收发各4级32位（或8级16位）FiFo</li><li>- 收发都可用 DMA传输</li></ul>
<b>SPI/ I<sup>2</sup>S</b>	<ul style="list-style-type: none"><li>- 最高 96 Mbit/s</li><li>- SPI收发位长可配置为 8 ~ 32</li><li>- SPI高低位在前可配置</li><li>- SPI支持字节重排序功能</li><li>- SPI支持字节或字挂起功能</li><li>- SPI支持半双工传输</li></ul>
<b>I<sup>2</sup>S</b>	<ul style="list-style-type: none"><li>- 支持单声道和立体声</li><li>- 支持位长： 8, 16, 24和 32位</li></ul>

	<ul style="list-style-type: none"> <li>- 支持 PCM 模式 A 和模式 B, I2S 和高位调整格式</li> </ul>
I <sup>2</sup> S	<ul style="list-style-type: none"> <li>- 一个 I<sup>2</sup>S 接口, 可与外部音频CODEC相连</li> <li>- 支持主从模式</li> <li>- 支持最高 192 kHz 的音频采样频率</li> <li>- 支持位长: 8, 16, 24 和 32 位</li> <li>- 支持单声道和立体声</li> <li>- 收发各 16 级 FIFO</li> <li>- 支持 I<sup>2</sup>S 协议: 菲利普标准, 高位和低位调整格式</li> <li>- 支持 PCM 协议: PCM 标准, 高位和低位调整格式</li> <li>- PCM 协议支持 TDM 多通道传输, 通道数可配置为 2, 4, 6, 8</li> <li>- 收发都可用 DMA 传输</li> </ul>
USCI 多用串口	<ul style="list-style-type: none"> <li>- 两个多用串口, 支持 UART, SPI 和 I2C 模式</li> <li>- TX, RX 单字节缓存</li> </ul> <p>UART:</p> <ul style="list-style-type: none"> <li>- 一个发送缓存, 两个接收缓存</li> <li>- 支持硬件流控和可设置的流量控制触发水平</li> <li>- 支持 9 位传输</li> <li>- 通过内建捕获功能可测对方波特率</li> <li>- 支持唤醒功能</li> <li>- 支持 DMA 传输</li> </ul> <p>SPI:</p> <ul style="list-style-type: none"> <li>- 支持主机模式或从机模式</li> <li>- 一个发送缓存, 两个接收缓存</li> <li>- 附带 16 级缓存</li> <li>- 传输位长 4~16 位 (四线模式仅支持 8 ~16)</li> <li>- 高低位在前可配</li> <li>- 支持字挂起</li> <li>- 支持 3 线模式, 无片选</li> <li>- 片选支持唤醒功能</li> <li>- 支持 DMA 传输</li> </ul> <p>I<sup>2</sup>C:</p> <ul style="list-style-type: none"> <li>- 支持主从模式</li> <li>- 一个发送缓存, 两个接收缓存</li> <li>- 支持标准速率 (100 kbps), 快速 (400 kbps) 和高速模式 (1 Mbps)</li> <li>- 支持 10 位地址</li> <li>- 支持 10 位超时</li> <li>- 支持总线监听.</li> <li>- 支持数据线跳变或地址匹配唤醒</li> <li>- 支持多地址识别</li> <li>- 支持设备地址标志</li> <li>- 建立和保持时间可配置</li> </ul>
CAN 2.0	<ul style="list-style-type: none"> <li>- 两个 CAN 控制器</li> <li>- CAN 协议 2.0A/ B</li> <li>- 速率可达 1M bit/s</li> <li>- 32 个消息队列, 每个队列都有消息掩码</li> <li>- 可配置的 FIFO 模式</li> <li>- 对于定时触应用可禁止重发</li> <li>- 支持中断</li> </ul>

	<ul style="list-style-type: none"> <li>- 支持掉电唤醒</li> </ul>
<b>SD 卡接口</b>	<ul style="list-style-type: none"> <li>- 2个SD卡接口, 兼容SD卡v2.0</li> <li>- 支持最高50 MHz, 在3.3V操作下实现200 Mbps。</li> <li>- 支持单数据线和4数据线模式</li> <li>- 3.3V时时钟最高可达48 MHz, 传输速率达 192 Mbps</li> <li>- 专门的 DMA 控制</li> <li>- 支持 SD, SDHC 和 SDIO 卡.</li> <li>- 支持 DMA 的分散聚集模式, 以加速 系统与SD/SDHC/SDIO卡之间的数据传输.</li> </ul>
<b>EBI外总线</b>	<ul style="list-style-type: none"> <li>- 支持三个内存块Bank, 三个支持极性控制的片选引脚</li> <li>- 每个内存块Bank可达 1 M字节, 实际可访问空间依芯片封装而定</li> <li>- 数据位宽支持8/16位</li> <li>- 16条数据线时支持字节写</li> <li>- 支持地址数据复用模式</li> <li>- 支持地址和数据分离模式</li> <li>- 可连接 i80 接口的 LCD</li> <li>- 支持 PDMA 模式</li> </ul>
<b>GPIO</b>	<ul style="list-style-type: none"> <li>- 四种输入输出模式: 准双向, 推挽输出, OD输出, 高阻输入</li> <li>- 输入电平可选TTL/斯密特</li> <li>- 可配置边沿或电平中断</li> <li>- 独立上/下拉电阻控制</li> <li>- 大电流输出驱动</li> <li>- 软件可选速率</li> <li>- 支持5V容限 (除了: PA.8 ~ 15; PB.0 ~ 15; PD.10 ~ 12; PF.2 ~ 5; All USB High Speed PIN; nReset.)</li> </ul>

## 控制界面

	<ul style="list-style-type: none"> <li>- 两路正交编码计数器.</li> </ul>
<b>QEI 正交编码计数接口</b>	<ul style="list-style-type: none"> <li>- 两个计数输入: QEI_A, QEI_B 和一个定位输入 QEI_INDEX</li> <li>- 支持2/4倍自由计数模式和比较计数模式</li> <li>- 与ECAP配合可测量脉宽</li> </ul>
<b>ECAP输入捕获定时器</b>	<ul style="list-style-type: none"> <li>- 支持最大两个输入捕获定时/计数单元, 24位上数定时器/计数器</li> <li>- 每个单元有3个捕获通道, 各自独立时间值锁存器</li> <li>- 计数器支持捕获复位和捕获重装功能.</li> <li>- 可捕获上沿, 下沿, 和双沿都捕获</li> <li>- 支持比较/匹配功能.</li> </ul>

## 进阶介面

	<p>全速 USB 2.0 OTG</p> <ul style="list-style-type: none"> <li>- 支持USB OTG规范2.0版本</li> <li>- 内建全速 USB 2.0 OTG 物理层</li> <li>- 可配置为仅主机, 仅设备, 由ID端选择</li> </ul> <p>全速 USB 2.0 主机</p> <ul style="list-style-type: none"> <li>- 支持USB规范1.1版本</li> <li>- 支持OHCI规范 1.0版本</li> </ul>
--	--

- 支持全速 (12Mbps) 和低速 (1.5Mbps).
- 支持控制传输, 批量传输, 中断传输, 等时传输
- 支持端口路径逻辑将全速, 低速设备接入OHCI 控制器.
- 一个根集线器 Root Hub.
- 支持电源控制和电流检测.
- 实时传输支持DMA 传输.

#### 全速 USB 2.0 设备

- 兼容USB 2.0全速规范
- 总线不活动超3ms后支持总线挂起
- 支持12个端点, 1K字节数据缓存
- 一个中断向量, 四个中断事件: 唤醒中断, 插拔中断, 数据中断和总线中断
- 远程唤醒功能

#### 高速 USB 2.0 OTG

- 支持USB OTG规范2.0版本
- 内建高速 USB 2.0 OTG 物理层
- 可配置为仅主机, 仅设备, 由ID端选择

#### USB 2.0 主机

- 支持USB规范V2.0版本
- 兼容 EHCI V1.0
- 兼容 OHCI V1.0
- 支持 高速(480Mbps), 全速 (12Mbps) , 低速 (1.5Mbps)
- 支持端口路径逻辑将全速, 低速设备接入OHCI 控制器.
- 支持控制, 批量, 中断, 等时和分割传输
- 一个根集线器 Root Hub.
- 集成了一个FS/LS路由控制器
- 内建 DMA

#### 高速 USB 2.0 设备

- 支持USB规范V2.0版本
- 支持12个端点, 每个都可配成批量, 中断, 等时模式输入或输出
- 4096 字节缓存
- 端点数据包最大 1024 字节
- 输入端点三种 控制模式: 自动有效模式, 手支有效模式, 飞行模式
- 支持挂起, 恢复和远程唤醒功能
- 支持 DMA 操作

#### 高速 USB 2.0 OTG

- 支持 IEEE 标准. 802.3 CSMA/CD 协议
- 支持以太网帧时间戳 IEEE Std. 1588 – 2002协议
- 支持半双工或全双工 10 /100 Mbps
- 支持 RMII 接口

#### Ethernet 以太网

- 支持 流控暂停和远程暂停功能
- 支持长帧(超 1518 字节) 和短帧 (低于 64字节) 接收
- MAC 地址识别支持13 CAM
- 支持魔法包唤醒功能
- 支持 MII 管理功能控制外部物理层
- 支持 DMA 功能

---

## 加密界面

---

<b>Elliptic Curve Cryptography (ECC)</b>	<ul style="list-style-type: none"><li>- 硬件 ECC 加速器。</li><li>- 支持192位和256位密钥长度。</li><li>- 支持质数域 GF(p) 和二进制段 GF(2<sup>m</sup>)</li><li>- 支持 NIST P-192, P-224, P-256, P-384 和 P-521</li><li>- 支持 NIST B-163, B-233, B-283, B-409 和 B-571</li><li>- 支持 NIST K-163, K-233, K-283, K-409 和 K-571</li><li>- GF(p) 和 GF(2<sup>m</sup>)支持点乘,加和加倍操作</li><li>- GF(p)支持加, 减, 乘, 和模除操作</li></ul>
<b>Advanced Encryption Standard (AES)</b>	<ul style="list-style-type: none"><li>- 硬件 AES 加速</li><li>- 支持 FIPS NIST 197</li><li>- 支持 128, 192 和 256位密钥</li><li>- 支持 ECB, CBC, CFB, OFB, CTR, CBC-CS1, CBC-CS2, 和 CBC-CS3 模式</li><li>- 兼容 NIST 800 38A</li></ul>
<b>Data Encryption Standard (DES)</b>	<ul style="list-style-type: none"><li>- 硬件 DES 加速</li><li>- 支持 ECB, CBC, CFB, OFB, 和 CTR 模式</li><li>- 支持 FIPS 46-3</li></ul>
<b>Triple Data Encryption Standard (3DES)</b>	<ul style="list-style-type: none"><li>- 硬件 Triple DES 加速</li><li>- 支持 2密钥和3密钥模式</li><li>- 支持 ECB, CBC, CFB, OFB, 和 CTR 模式</li><li>- 支持 FIPS NIST 800-67</li><li>- 执行 X9.52 标准</li></ul>
<b>Secure Hash Algorithm (SHA)</b>	<ul style="list-style-type: none"><li>- 硬件 SHA 加速</li><li>- 支持 FIPS NIST 180, 180-2</li><li>- 支持SHA-160, SHA-224, SHA-256, SHA-384, 他 SHA-512</li><li>- 支持FIPS NIST 180, 180-2</li></ul>
<b>keyed-Hash Message Authentication Code (HMAC)</b>	<ul style="list-style-type: none"><li>- 硬件 HMAC 加速</li><li>- 支持 HMAC-SHA-160, HMAC-SHA-224, HMAC-SHA-256, HMAC-SHA-384, 和HMAC-SHA-512</li><li>- 支持FIPS NIST 180, 180-2</li></ul>

### 3 料号信息

#### 3.1 概要

Part No.	USB FS	USB HS	CAN	Crypto	Ethernet
M481	-	-	-	-	-
M482	√	-	-	-	-
M483	√	√	√	-	-
M484	√	√	-	-	-
M485	√	√	-	√	-
M487	√	√	√	√	√

#### 3.2 封装

Part No.	QFN33	WLCSP49	LQFP48	LQFP64	LQFP128
M481	M481ZGAAE M481ZIDAE	M481LGAAE M481LIDAE	M481SGAAE M481SIDAE		
M482	M482ZIDAE	M482LGAAE M482LIDAE	M482SGAAE M482SIDAE	M482KGAAE M482KIDAE	
M483			M483SGAAE M483SIDAE	M483KIDAE	
M484			M484SGAAE M484SIDAE M484SGAAE2U M484SIDAE2U	M484KIDAE	
M485	M485ZIDAE	M485LIDAE	M485SIDAE	M485KIDAE	
M487			M487SIDAE	M487KIDAE	M487JIDAE

## 3.3 NuMicro® M481 基本系列

PART NUMBER		M481										
		ZGAAE	ZIDAE	LGAAE	LIDAE	SGAAE	SIDAE					
Flash (KB)	256	512	256	512	256	512						
SRAM (KB)	96	160	96	160	96	160						
ISP Loader ROM (KB)	4											
I/O	26		41		52							
32-bit Timer	4											
Tamper	-				1							
RTC	√											
Connectivity	LPUART	6										
	ISO-7816	3										
	SPI Master	1										
	Quad SPI	1										
	SPI/I <sup>2</sup> S	3			4							
	I <sup>2</sup> S	1										
	I <sup>2</sup> C	3										
	USCI	2										
	CAN	-										
	LIN	2										
	SDHC	1			2							
16-bit EPWM												
16-bit BPWM												
QEI	1			2								
ECAP	-			1								
USB 2.0 FS OTG	-											
USB 2.0 HS OTG	-											
12-bit ADC	10	12		16								
12-bit DAC	2											
Analog Comparator	1			2								
Operational Amplifier	1			2								
Ethernet	-											
Cryptography	-											
External Bus Interface	√											
Package	QFN 33	LQFP 48		LQFP 64								

## 3.4 NuMicro® M482 USB FS OTG 系列

PART NUMBER	M482						
	ZIDAE	LGAAE	LIDAE	SGAAE	SIDAE	KGAAE	KIDAE
Flash (KB)	512	256	512	256	512	256	512
SRAM (KB)	160	96	160	96	160	96	160
ISP Loader ROM (KB)				4			
I/O	26	41		52		100	
32-bit Timer				4			
Tamper	-			1		6	
RTC				√			
Connectivity	LPUART			6			
	ISO-7816			3			
	SPI Master			1			
	Quad SPI			1			
	SPI/I <sup>2</sup> S	3	3		4		4
	I <sup>2</sup> S			1			
	I <sup>2</sup> C			3			
	USCI			2			
	CAN			-			
	LIN			2			
	SDHC			2			
16-bit EPWM				12			
16-bit BPWM				12			
QEI	1	2		2		2	
ECAP	-		1			2	
USB 2.0 FS OTG				√			
USB 2.0 HS OTG				-			
12-bit ADC	10	12		16		16	
12-bit DAC				2			
Analog Comparator				2			
Operational Amplifier	1	2		2		3	
Ethernet				-			
Cryptography				-			
External Bus Interface	-	√		√		√	
Package	QFN33	LQFP 48		LQFP 64		LQFP 128	

## 3.5 NuMicro® M483 CAN系列

PART NUMBER		M483		
		SGaAE	SIDAE	KIDAE
Flash (KB)	256	512		512
SRAM (KB)	96	160		160
ISP Loader ROM (KB)			4	
I/O		44		100
32-bit Timer			4	
Tamper		1		6
RTC			√	
Connectivity	LPUART		6	
	ISO-7816		3	
	SPI Master		1	
	Quad SPI		1	
	SPI/I <sup>2</sup> S		4	
	I <sup>2</sup> S		1	
	I <sup>2</sup> C		3	
	USCI		2	
	CAN		2	
	LIN		2	
	SDHC		2	
	16-bit EPWM		12	
	16-bit BPWM		12	
QEI			2	
ECAP	1		2	
USB 2.0 FS OTG	-		√	
USB 2.0 HS OTG			√	
12-bit ADC		16		
12-bit DAC		2		
Analog Comparator			2	
Operational Amplifier	2		3	
Ethernet			-	
Cryptography			-	
External Bus Interface			√	
Package	LQFP 64		LQFP 128	

## 3.6 NuMicro® M484 USB HS OTG系列

PART NUMBER	M484				
	SGAAE	SIDAE	SGAAE2U	SIDAE2U	KIDAE
Flash (KB)	256	512	256	512	512
SRAM (KB)	96	160	96	160	160
ISP Loader ROM (KB)				4	
I/O		44			100
32-bit Timer				4	
Tamper		1			6
RTC				√	
Connectivity	LPUART			6	
	ISO-7816			3	
	SPI Master			1	
	Quad SPI			1	
	SPI/I <sup>2</sup> S			4	
	I <sup>2</sup> S			1	
	I <sup>2</sup> C			3	
	USCI			2	
	CAN			-	
	LIN			2	
	SDHC			2	
16-bit EPWM			12		
16-bit BPWM			12		
QEI			2		
ECAP		1			2
USB 2.0 FS OTG	-			√	
USB 2.0 HS OTG				√	
12-bit ADC			16		
12-bit DAC			2		
Analog Comparator			2		
Operational Amplifier		2			3
Ethernet			-		
Cryptography			-		
External Bus Interface			√		
Package	LQFP 64		LQFP 64		LQFP 128

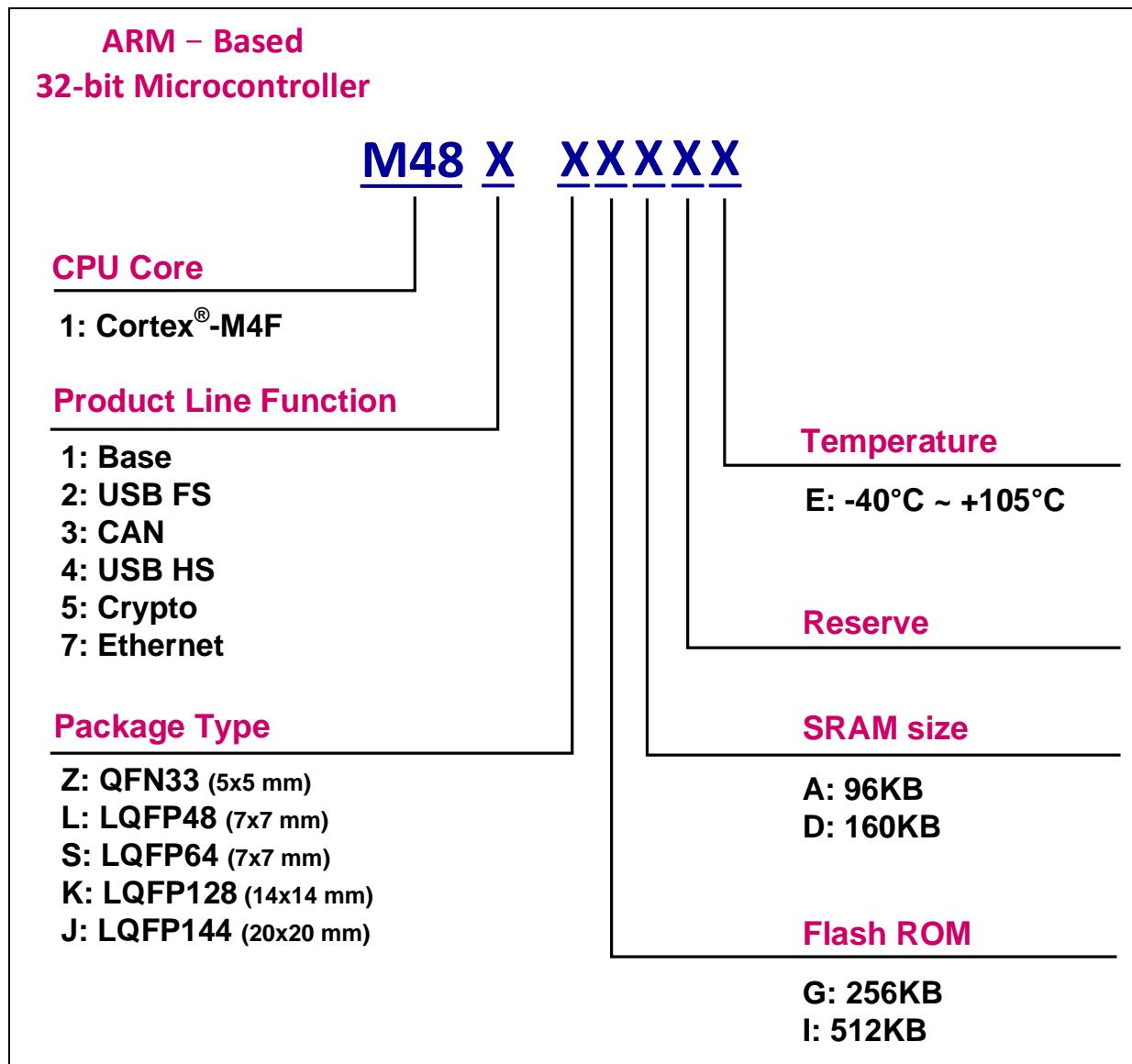
## 3.7 NuMicro® M485 Crypto系列

PART NUMBER	M485				
	ZIDAE	LIDAE	SIDAE	KIDAE	
Flash (KB)			512		
SRAM (KB)			160		
ISP Loader ROM (KB)			4		
I/O	26	41	44	100	
32-bit Timer			4		
Tamper	-	-	1	6	
RTC			√		
Connectivity	LPUART			6	
	ISO-7816			3	
	SPI Master			1	
	Quad SPI			1	
	SPI/I <sup>2</sup> S	3			4
	I <sup>2</sup> S			1	
	I <sup>2</sup> C			3	
	USCI			2	
	CAN			-	
	LIN			2	
SDHC	1			2	
16-bit EPWM			12		
16-bit BPWM			12		
QEI	1			2	
ECAP	-			2	
USB 2.0 FS OTG			√	-	
USB 2.0 HS OTG	-	-	√	√	
12-bit ADC	10	12			16
12-bit DAC			2		
Analog Comparator			2		
Operational Amplifier	1			3	
Ethernet			-		
Cryptography			√		
External Bus Interface	-			√	
Package	QFN33	LQFP 48	LQFP 64	LQFP 128	

## 3.8 NuMicro® M487 以太网系列

PART NUMBER		M487		
		SIDAE	KIDAE	JIDAE
Flash (KB)	512	512	512	512
SRAM (KB)		160		
ISP Loader ROM (KB)		4		
I/O	44	100		114
32-bit Timer		4		
Tamper	2		6	
RTC		√		
Connectivity	UART		6	
	ISO-7816		3	
	SPI Master		1	
	Quad SPI		1	
	SPI/I <sup>2</sup> S	5		6
	I <sup>2</sup> S		1	
	I <sup>2</sup> C		3	
	USCI		2	
	CAN		2	
	LIN		2	
	SDHC		2	
	16-bit EPWM		12	
	16-bit BPWM		12	
	QEI	1		2
	ECAP		2	
USB 2.0 FS OTG	-		√	
USB 2.0 HS OTG			√	
12-bit ADC	12		16	
12-bit DAC		2		
Analog Comparator		2		
Operational Amplifier	2		3	
Ethernet		√		
Cryptography		√		
External Bus Interface	-		√	
Package	LQFP 64	LQFP 128		LQFP 144

## 3.9 NuMicro® M480 命名规则



## 4 引脚配置

### 4.1 引脚配置

#### 4.1.1 NuMicro® M481 基本系列 QFN33 引脚图

对应料号: M481ZGAAE, M481ZIDAE

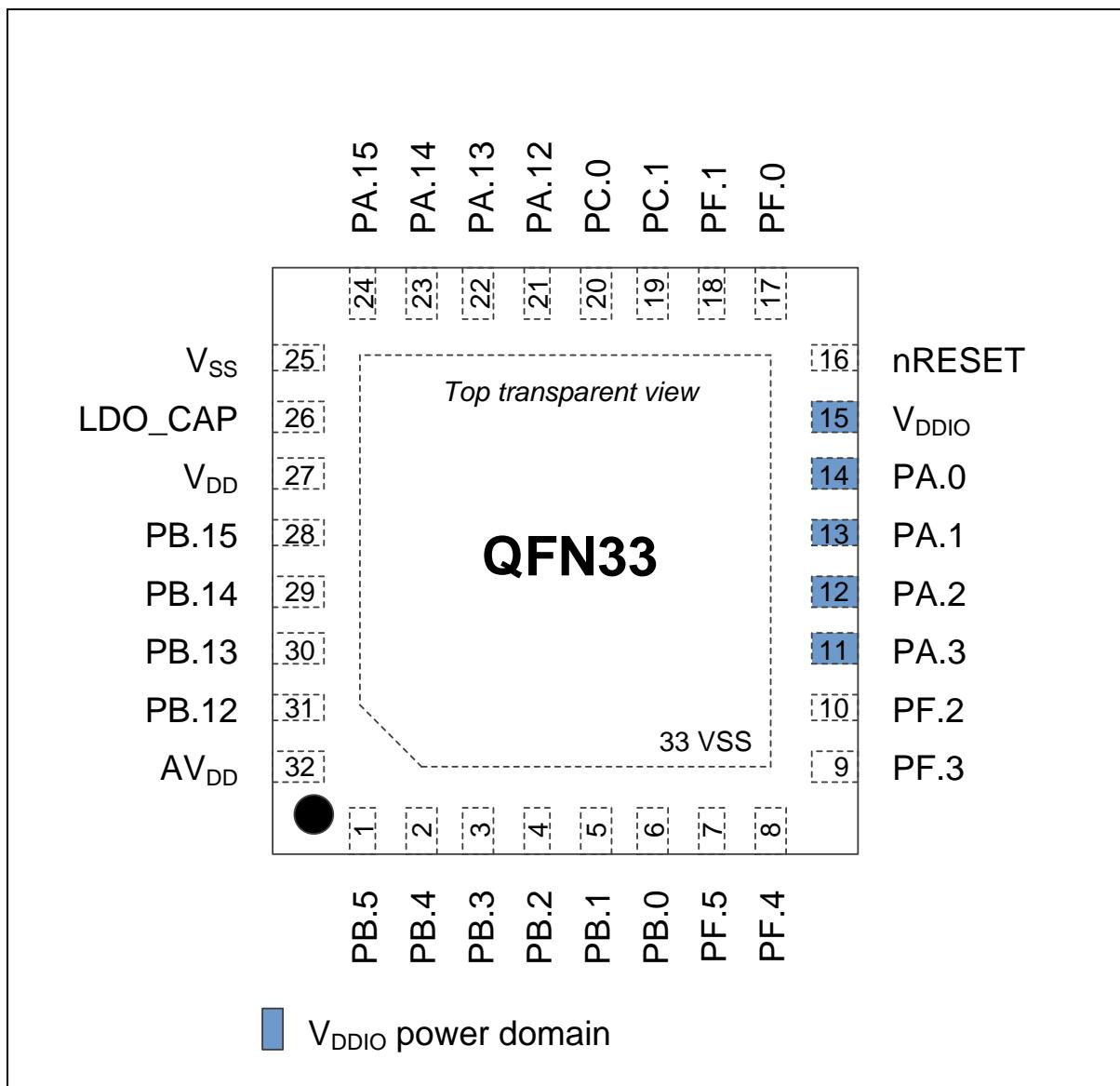


图 4.1-1 NuMicro® M481 基本系列 QFN 33 引脚图

## 4.1.2 NuMicro® M481 基本系列 LQFP48 引脚图

对应料号: M481LGAAE, M481LIDAE

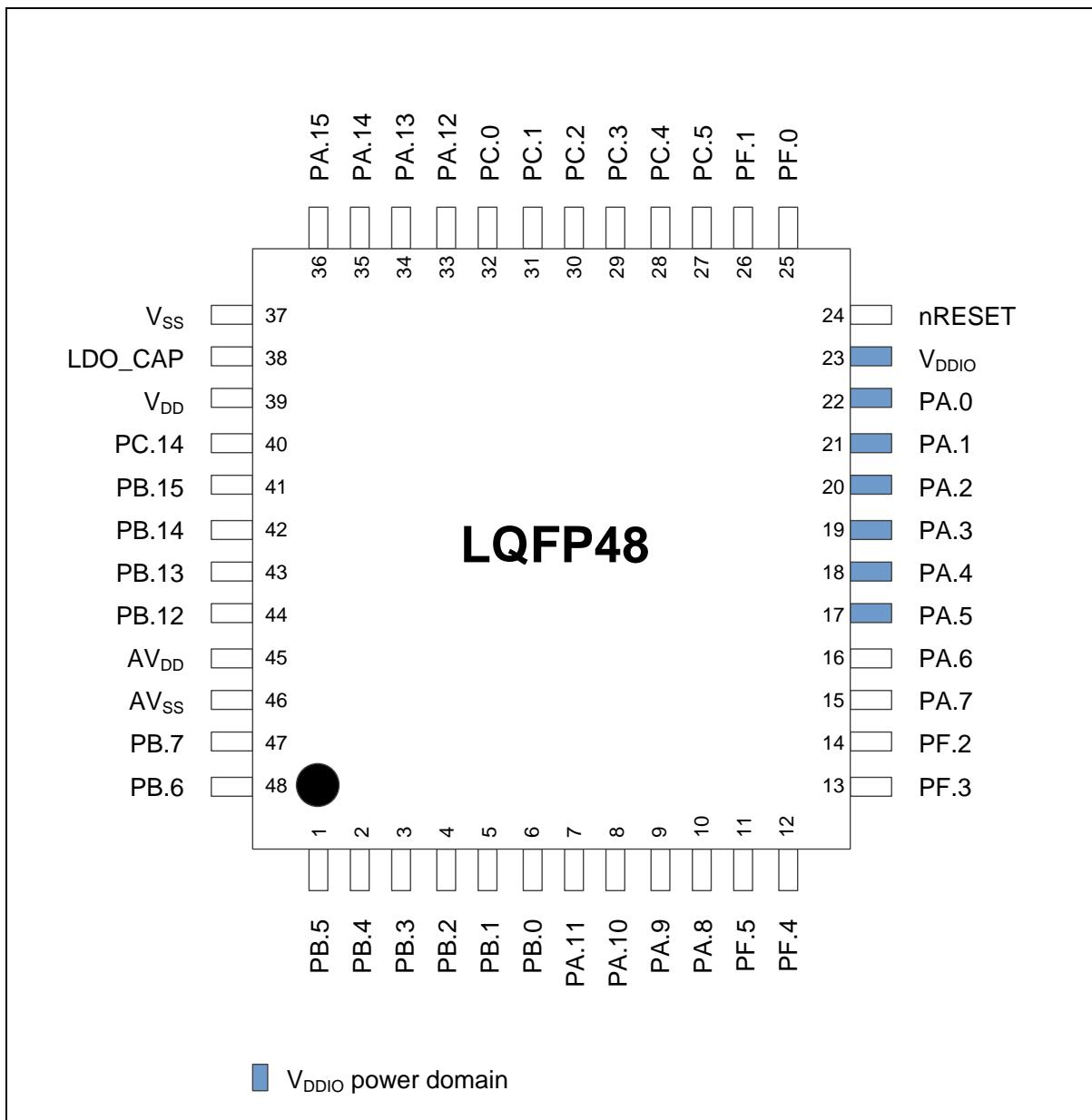


图 4.1-2 NuMicro® M481 基本系列 LQFP 48 引脚图

## 4.1.3 NuMicro® M481 基本系列 LQFP64 引脚图

对应料号: M481SGAAE, M481SIDAE

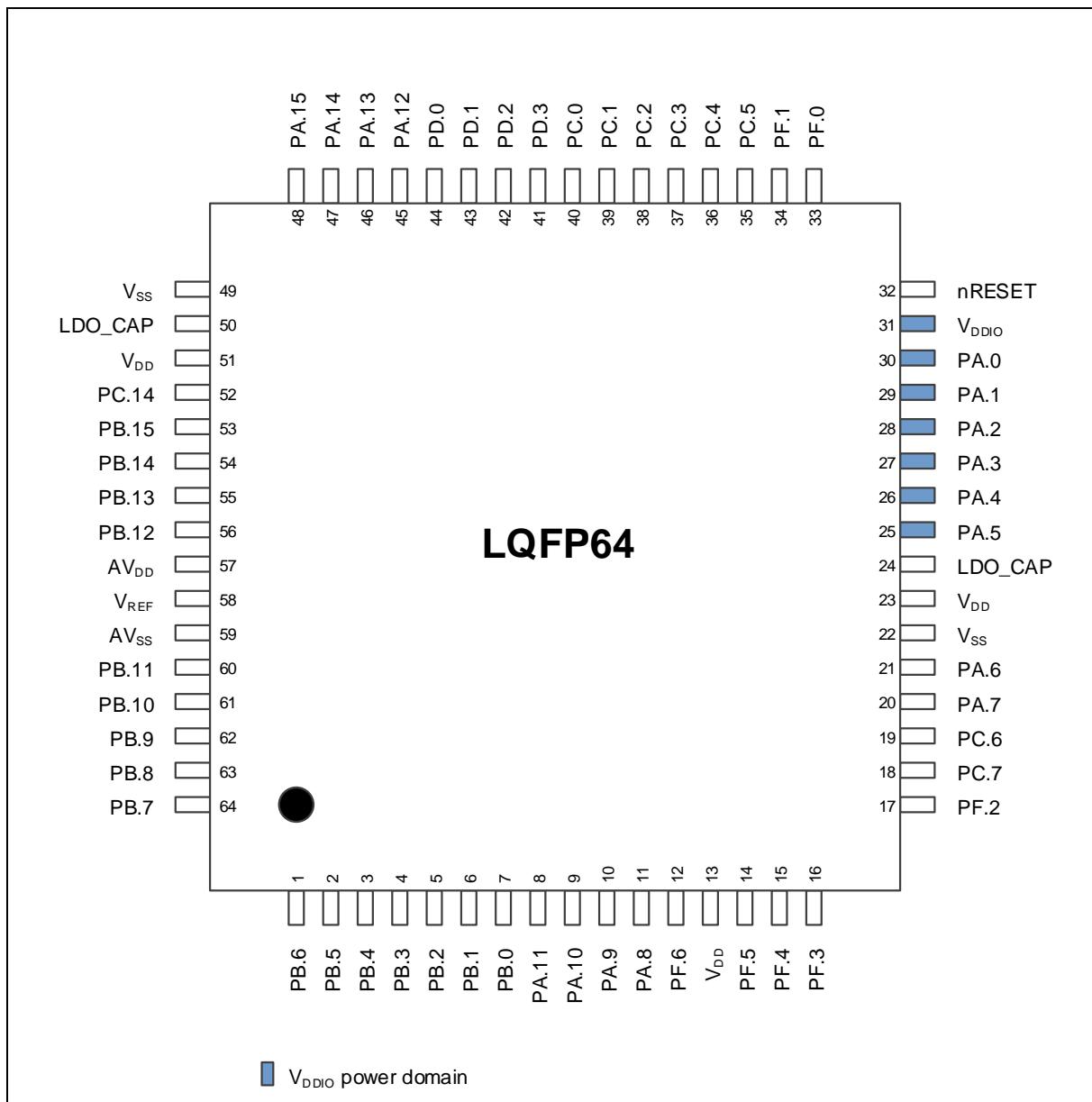


图 4.1-3 NuMicro® M481 基本系列 LQFP 64 引脚图

## 4.1.4 NuMicro® M482 USB FS OTG 系列 QFN33 引脚图

对应料号: M482ZIDAE

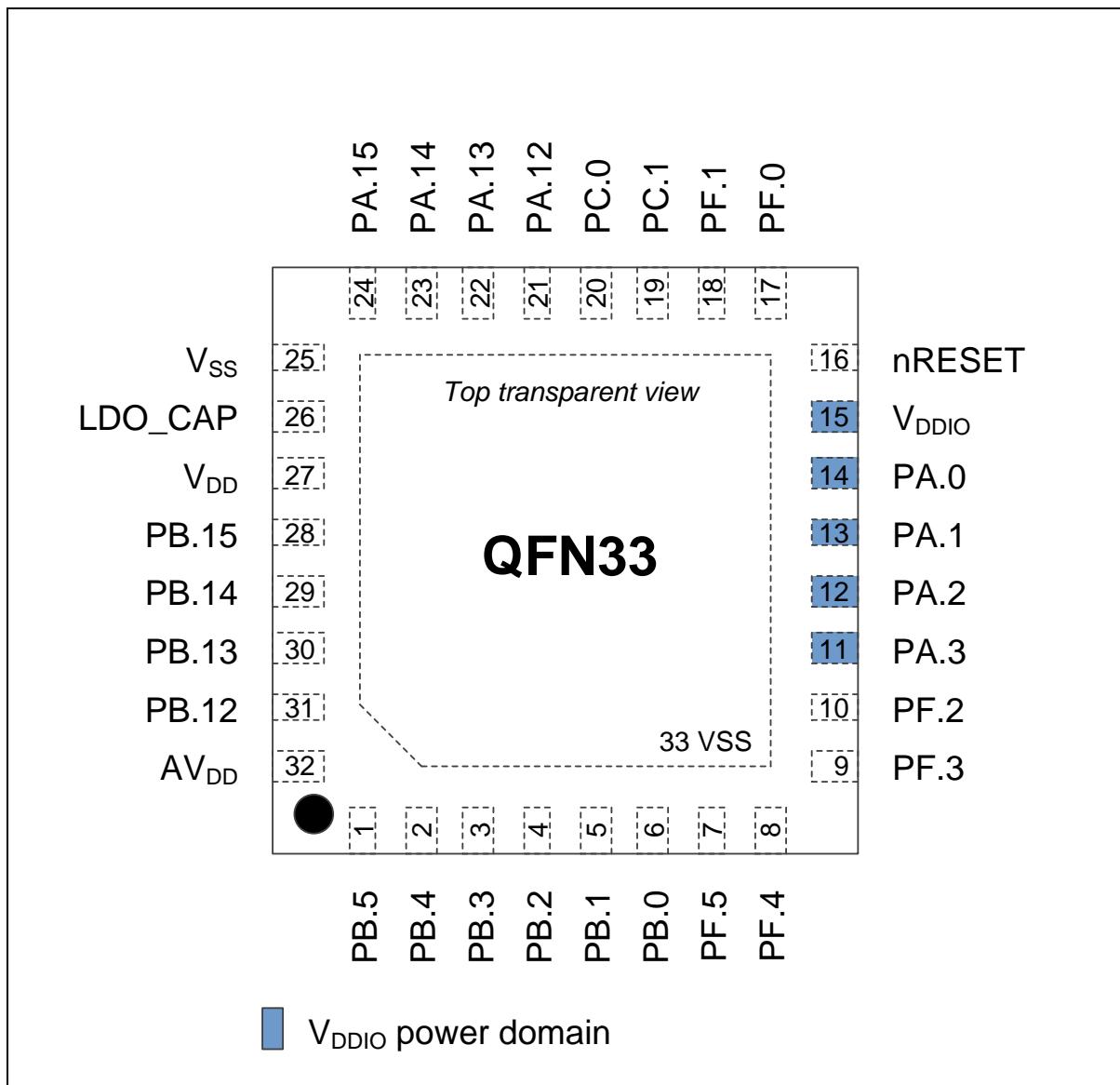


图 4.1-4 NuMicro® M482 USB FS OTG 系列 QFN 33 引脚图

## 4.1.5 NuMicro® M482 USB FS OTG 系列 LQFP48 引脚图

对应料号: M482LGAAE, M482LIDAE

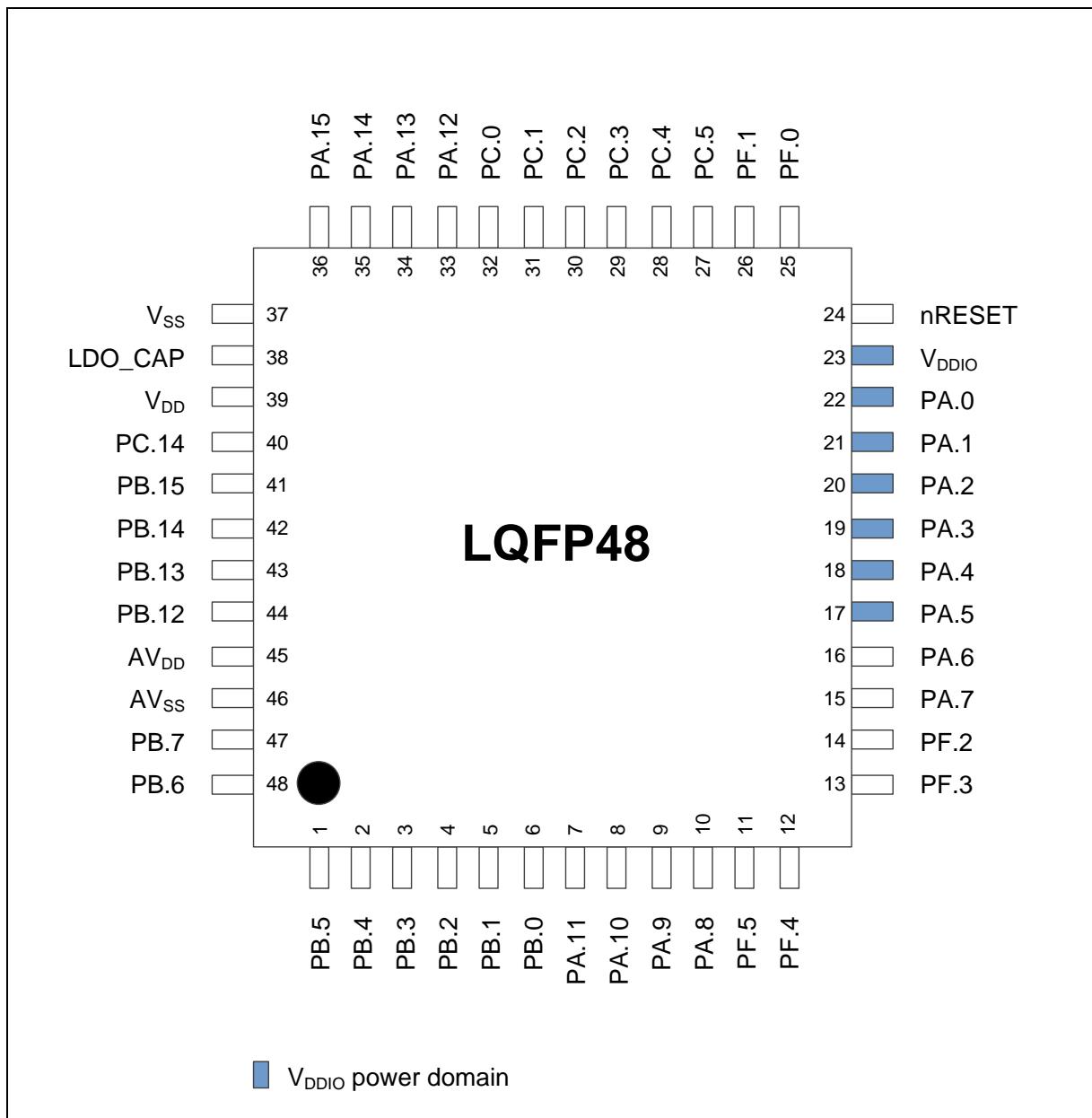


图 4.1-5 NuMicro® M482 USB FS OTG 系列 LQFP 48 引脚图

## 4.1.6 NuMicro® M482 USB FS OTG 系列 LQFP64 引脚图

对应料号: M482SGAAE, M482SIDAE

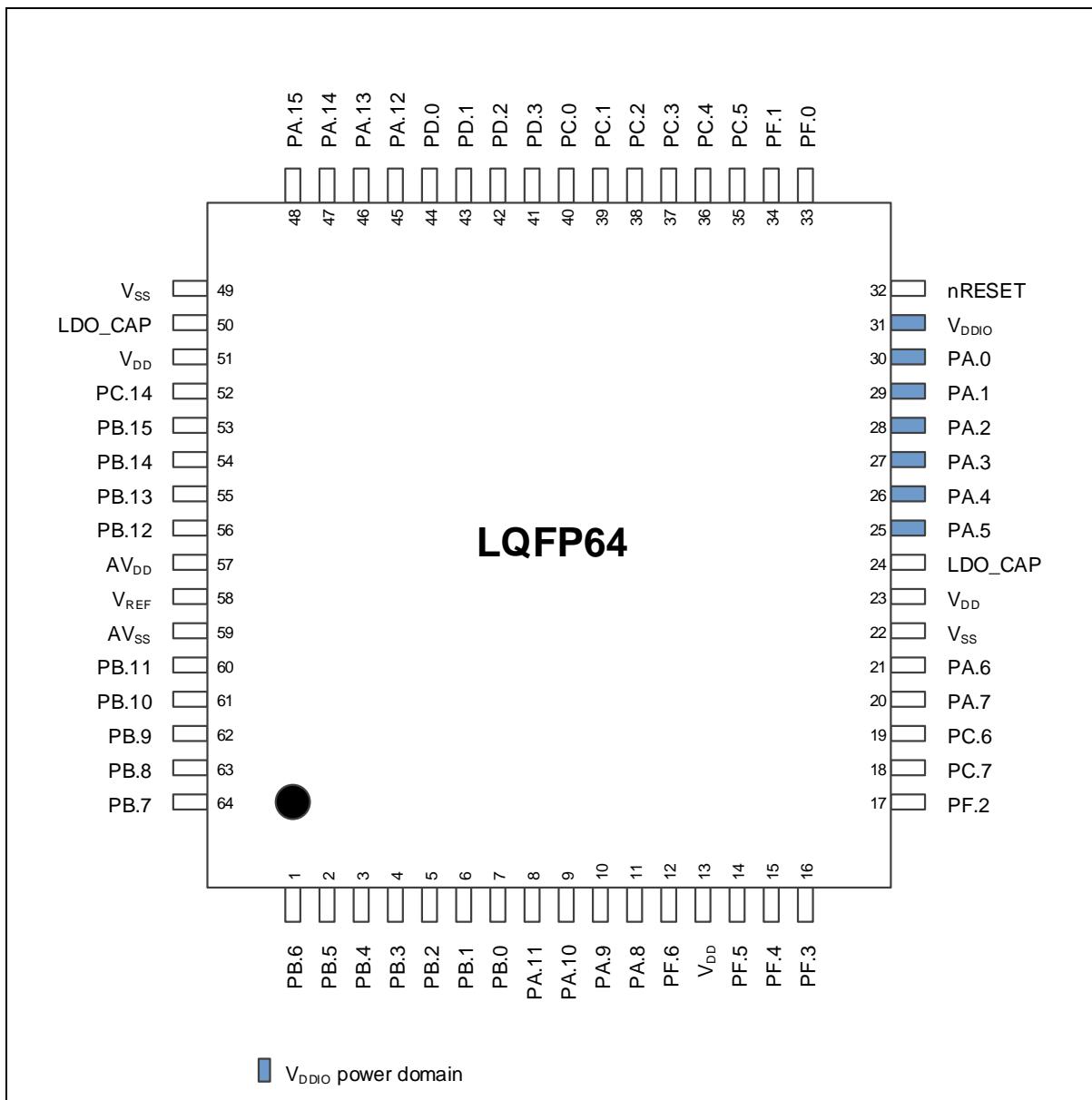


图 4.1-6 NuMicro® M482 USB FS OTG 系列 LQFP 64 引脚图

## 4.1.7 NuMicro® M482 USB FS OTG 系列 LQFP128 引脚图

对应料号: M482KGAAE, M482KIDAE

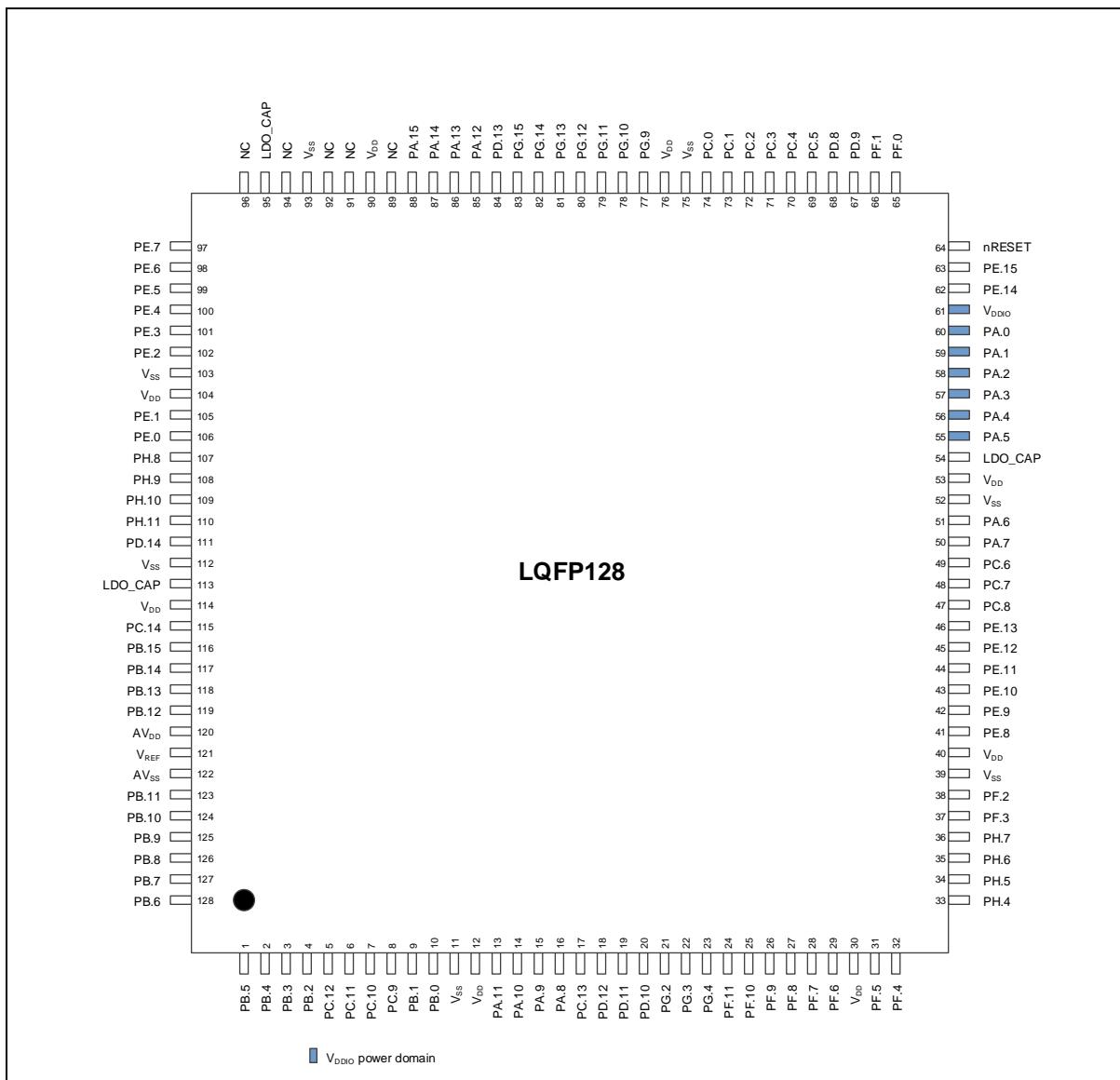


图 4.1-7 NuMicro® M482 USB FS OTG 系列 LQFP 128 引脚图

## 4.1.8 NuMicro® M483 CAN 系列 LQFP64 引脚图

对应料号: M483SGAAE, M483SIDAE

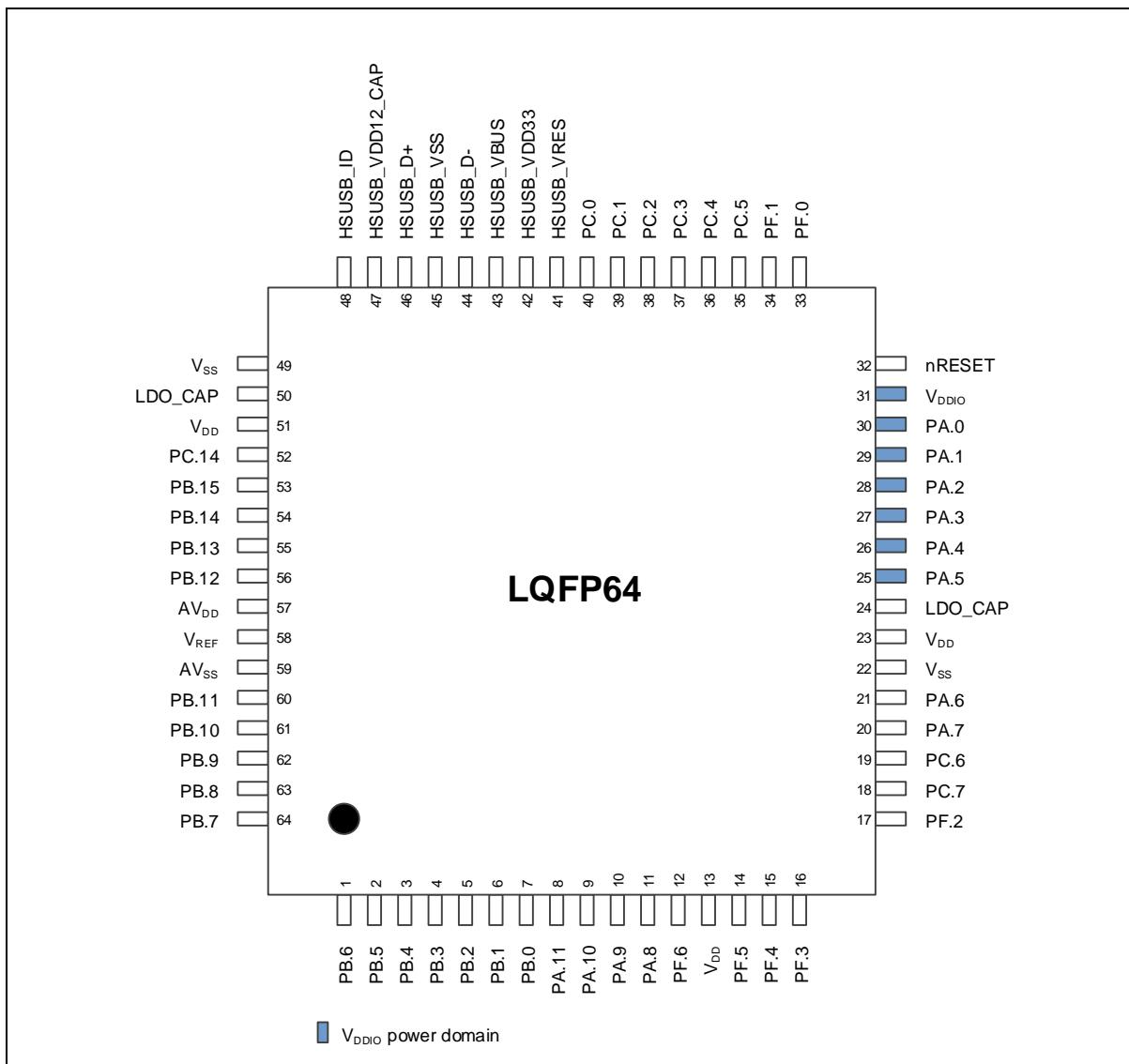


图 4.1-8 NuMicro® M483 CAN 系列 LQFP 64 引脚图

## 4.1.9 NuMicro® M483 CAN 系列 LQFP128 引脚图

对应料号: M483KGAAE, M483KIDAE

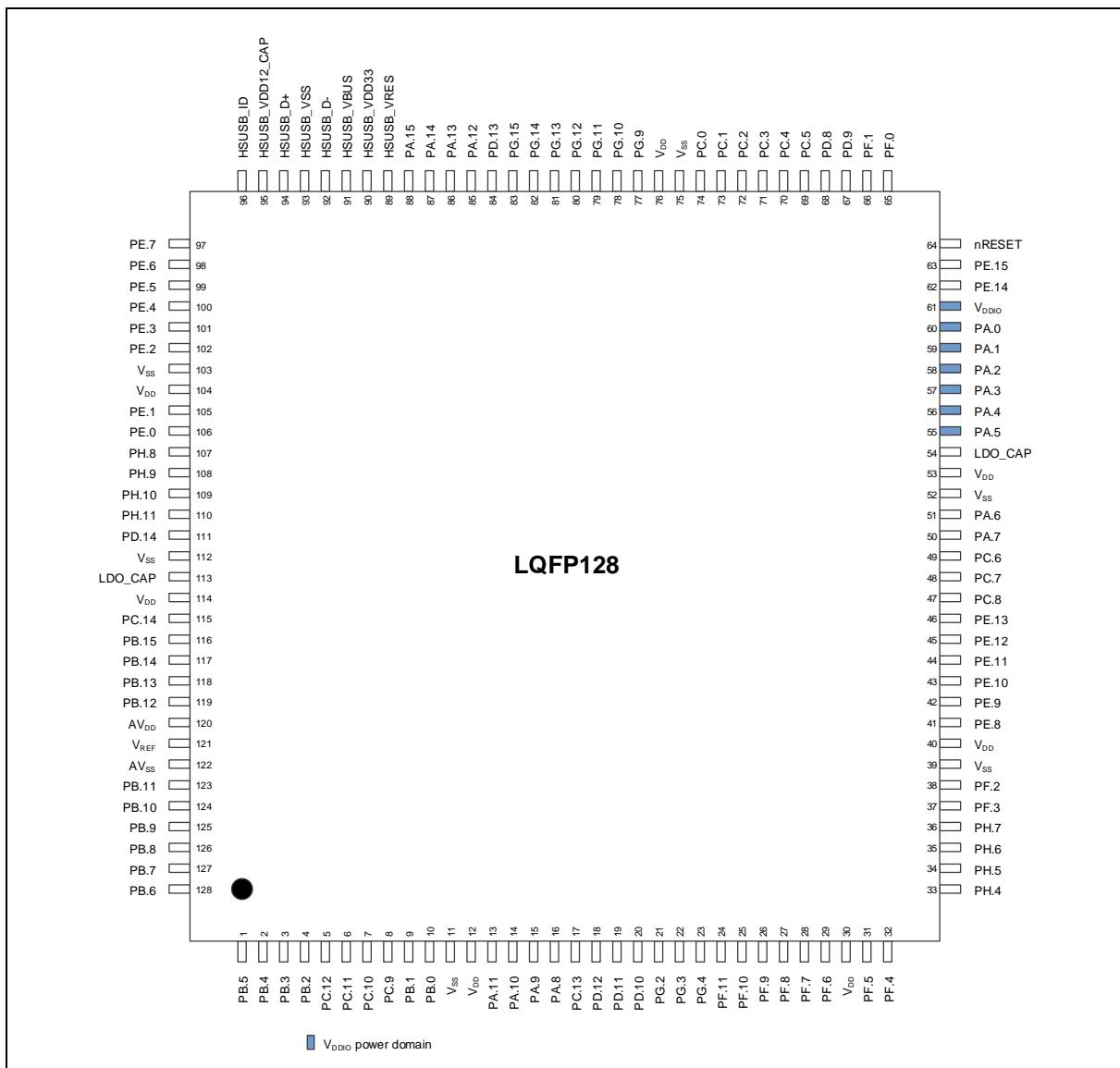


图 4.1-9 NuMicro® M483 CAN 系列 LQFP 128 引脚图

## 4.1.10 NuMicro® M484 USB HS OTG 系列 LQFP64 引脚图

对应料号: M484SGAAE, M484SIDAE

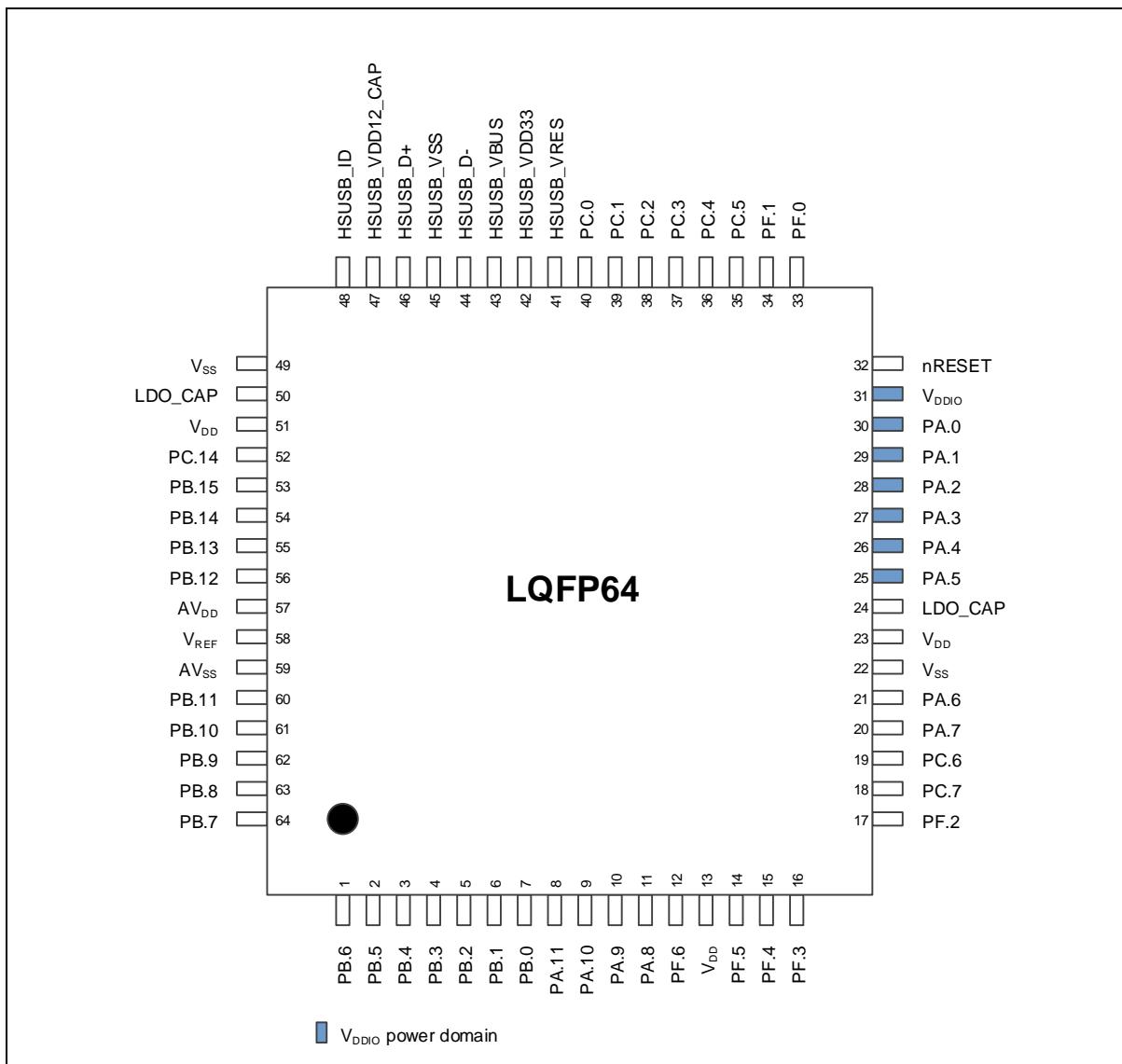


图 4.1-10 NuMicro® M484 USB HS OTG 系列 LQFP 64 引脚图

## 4.1.11 NuMicro® M484 USB HS OTG系列带有两组USB LQFP64引脚图

对应料号: M484SGAAE2U, M484SIDAE2U

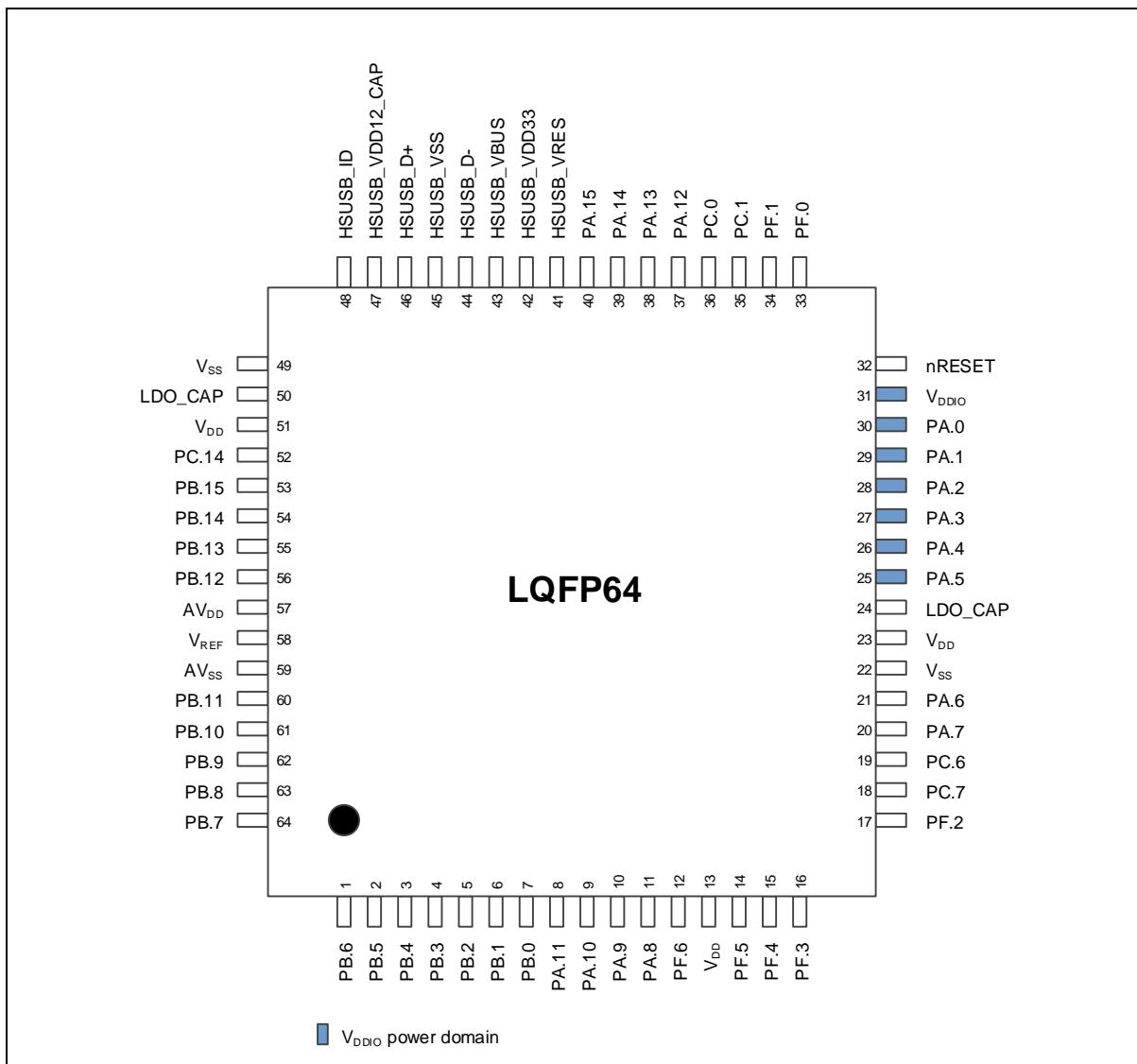


图 4.1-11 NuMicro® M484 USB HS OTG 系列 LQFP 64 引脚图

## 4.1.12 NuMicro® M484 USB HS OTG系列LQFP128引脚图

对应料号: M484KIDAE

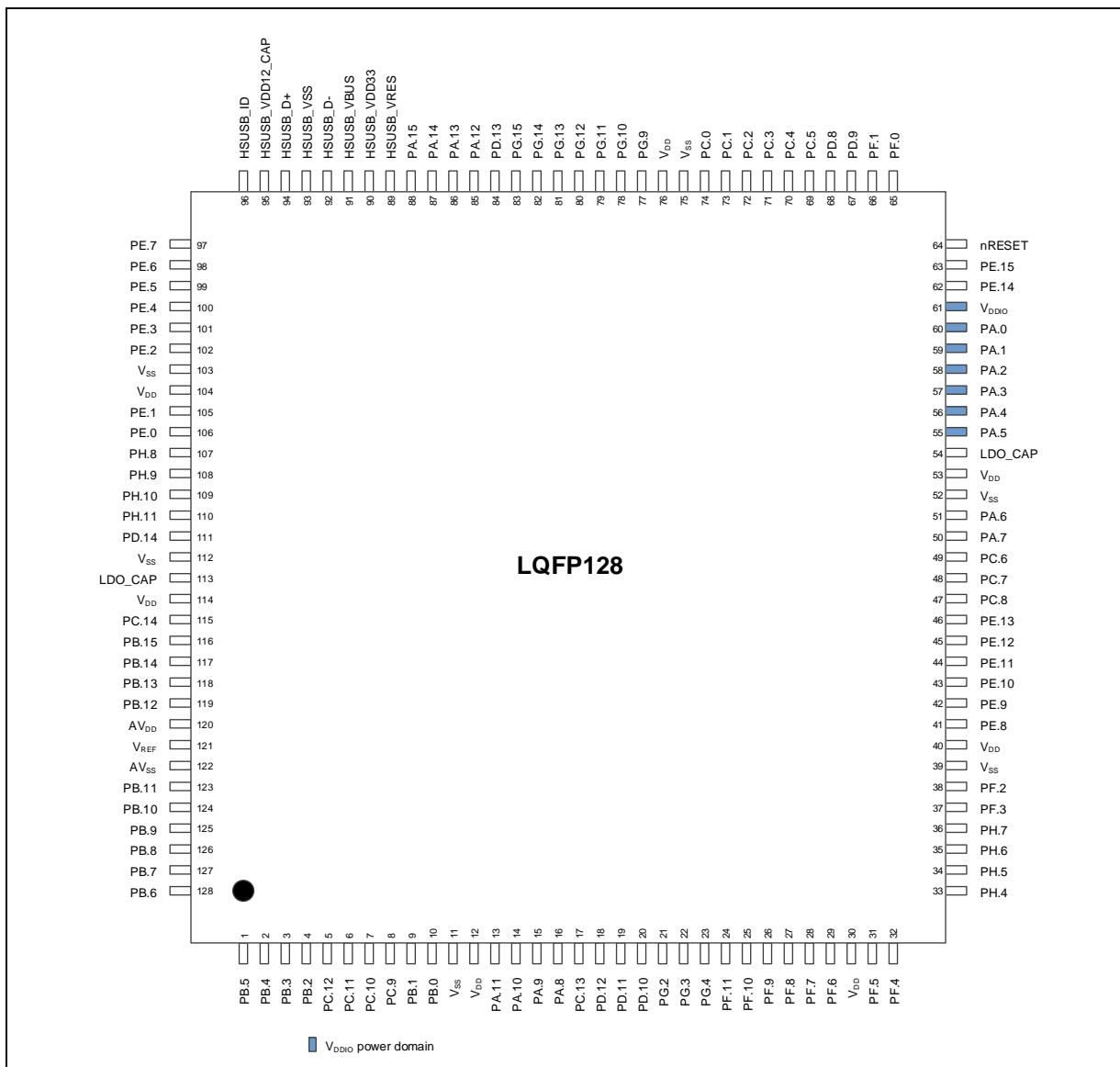


图 4.1-12 NuMicro® M484 USB HS OTG 系列 LQFP 128 引脚图

## 4.1.13 NuMicro® M485 加解密系列 QFN33 引脚图

对应料号: M485ZIDAE

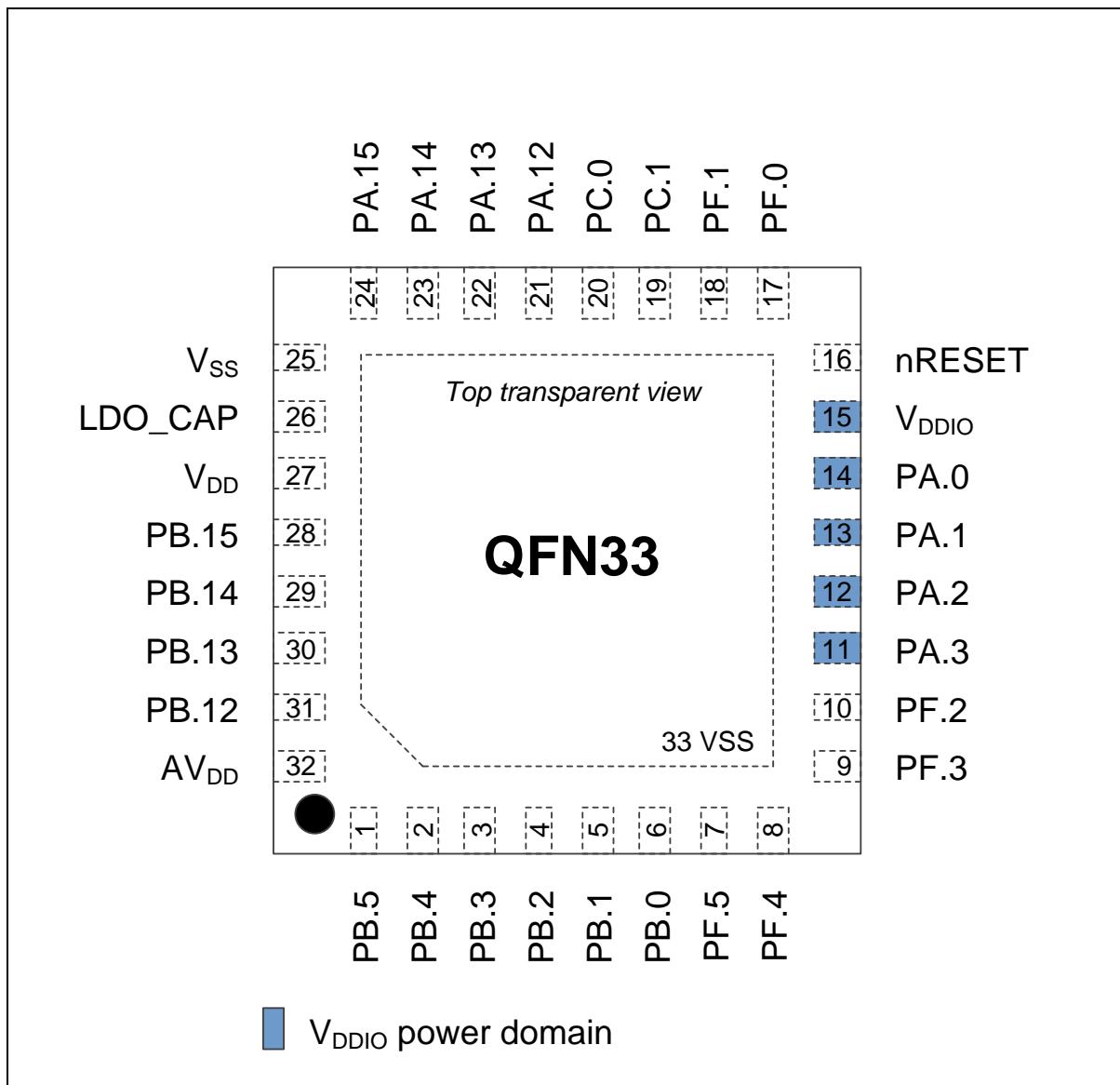


图 4.1-13 NuMicro® M485 加解密系列 QFN 33 引脚图

## 4.1.14 NuMicro® M485 加解密系列 LQFP48 引脚图

对应料号: M485LIDAE

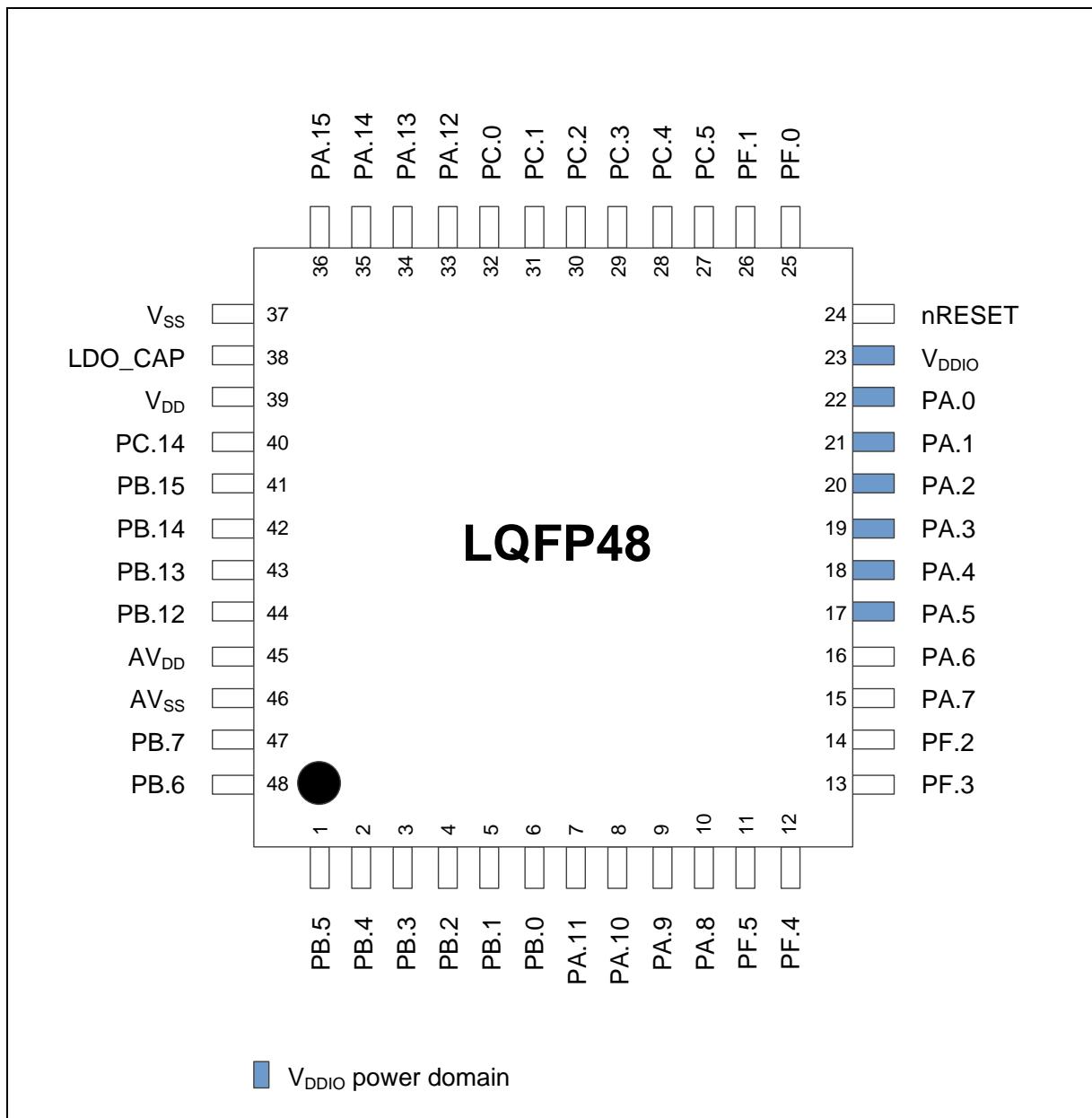


图 4.1-14 NuMicro® M485 加解密系列 LQFP 48 引脚图

## 4.1.15 NuMicro® M485加解密系列LQFP64引脚图

对应料号: M485SGAAE, M485SIDAE

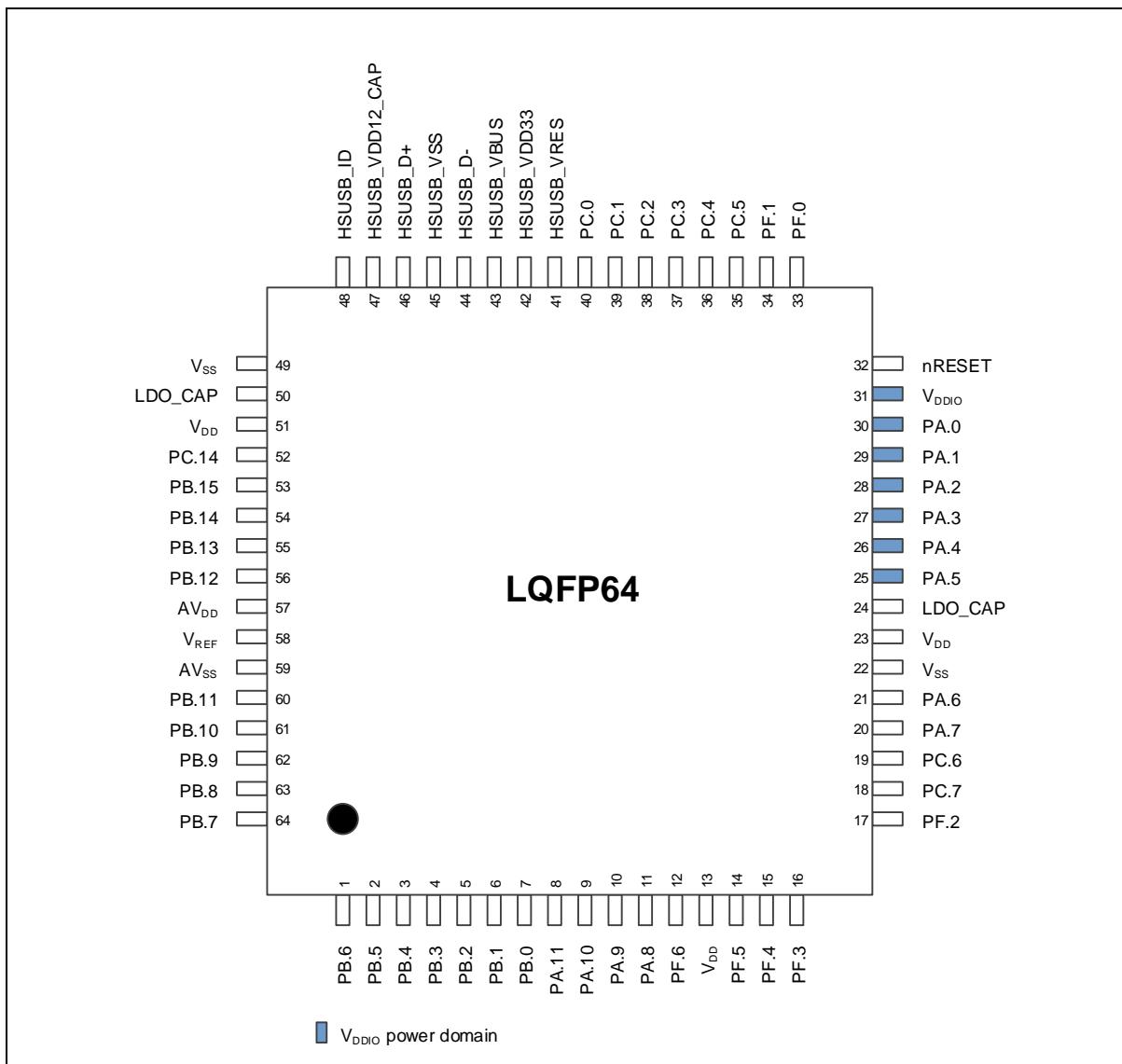


图 4.1-15 NuMicro® M485 加解密系列 LQFP 64 引脚图

## 4.1.16 NuMicro® M485 加解密系列LQFP128引脚图

对应料号: M485KGAAE, M485KIDAE

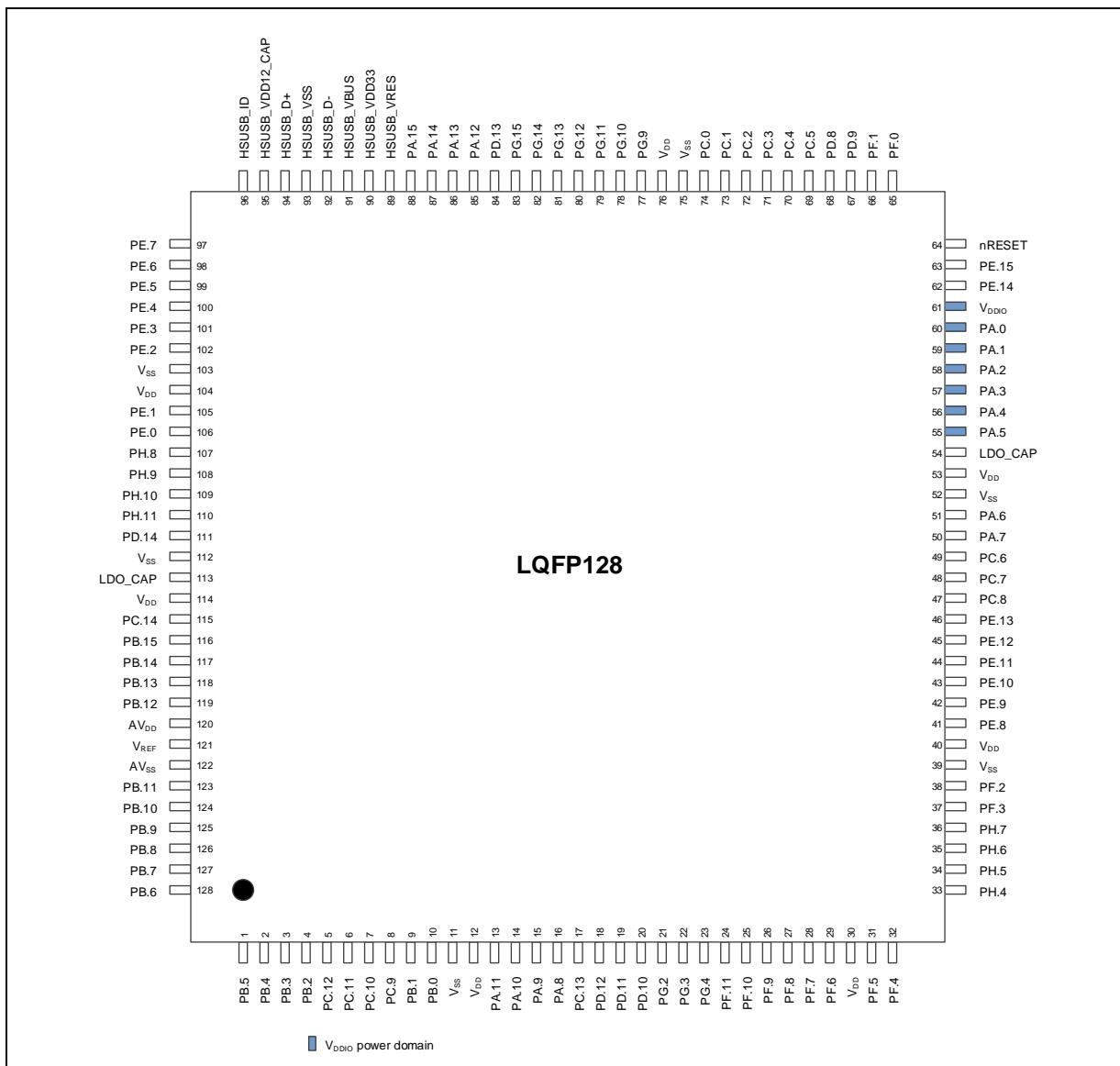


图 4.1-16 NuMicro® M485 加解密系列 LQFP 128 引脚图

## 4.1.17 NuMicro® M487 以太网系列LQFP64引脚图

对应料号: M487SGAAE, M487SIDAE

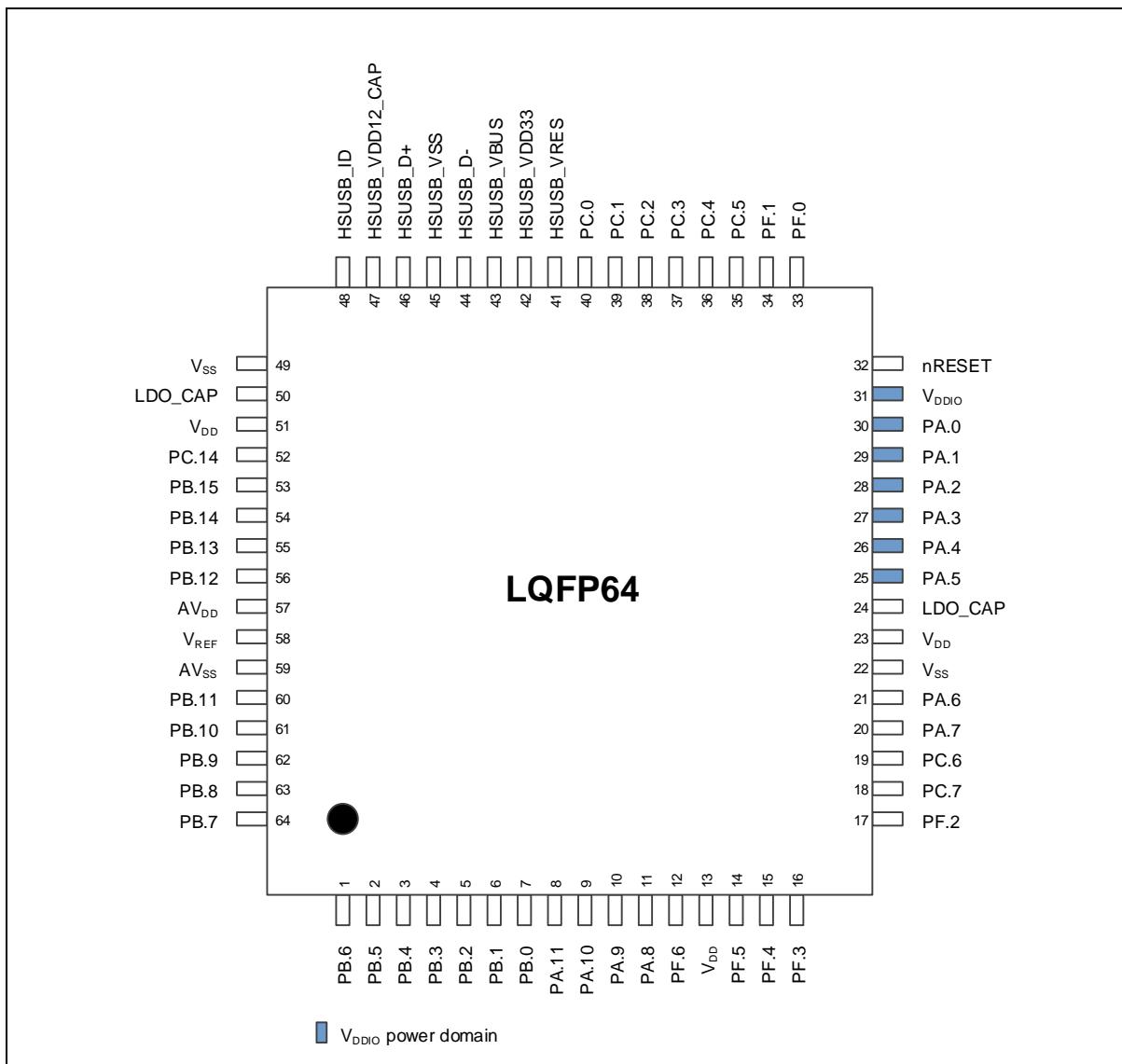


图 4.1-17 NuMicro® M487 以太网系列 LQFP 64 引脚图

## 4.1.18 NuMicro® M487 以太网系列LQFP128引脚图

对应料号: M487KGAAE, M487KIDAE

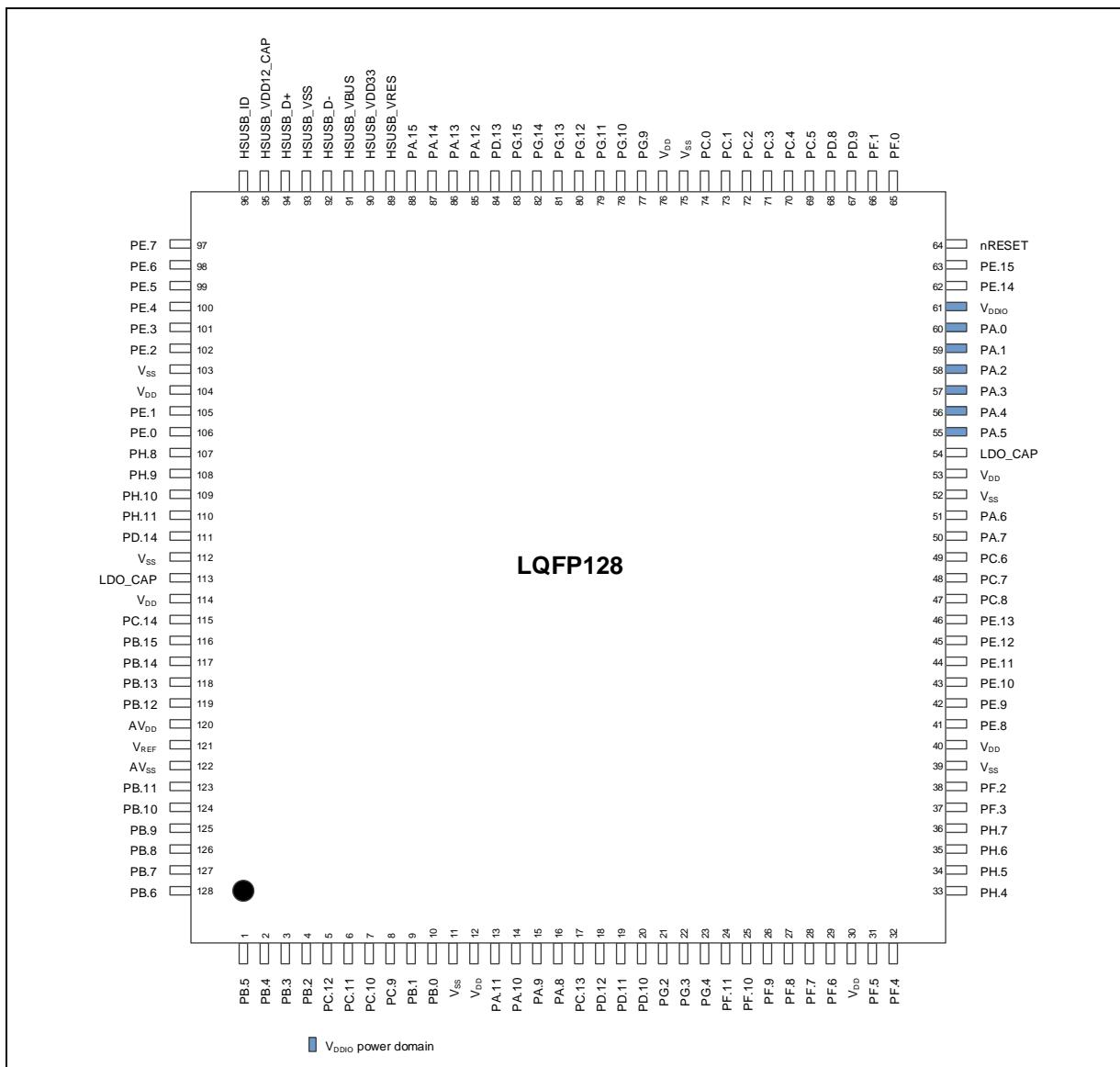


图 4.1-18 NuMicro® M487 以太网系列 LQFP 128 引脚图

## 4.1.19 NuMicro® M487 以太网系列LQFP144引脚图

对应料号: M487JGAAE, M487JIDAE

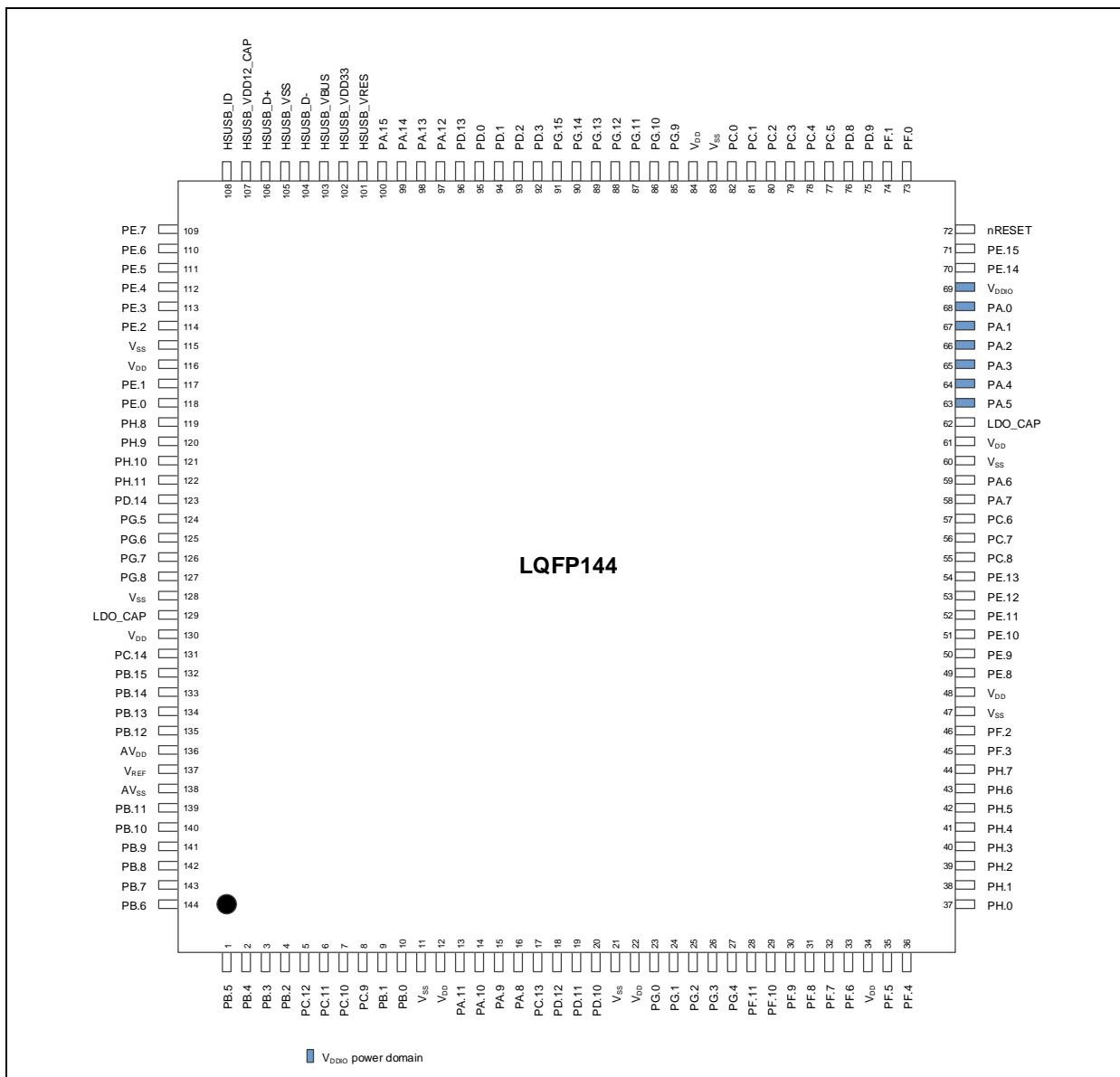


图 4.1-19 NuMicro® M487 以太网系列 LQFP 144 引脚图

## 4.2 引脚描述

### 4.2.1 M481 系列引脚描述

MFP\* = 多功能引脚 (Multi-function pin). (参考 SYS\_GPx\_MFPL 和 SYS\_GPx\_MFPH)

PA.0 MFP0 意思是 SYS\_GPA\_MFPL[3:0] = 0x0.

PA.9 MFP5 意思是 SYS\_GPA\_MFPH[7:4] = 0x5.

32 脚	48 脚	64 脚	引脚名	类型	MFP值	描述
48	1	PB.6	I/O	MFP0	GPIO.	
		EADC0_CH6	A	MFP1	EADC0 通道6模拟输入.	
		EBI_nWRH	O	MFP2	EBI 高字节写使能输出脚	
		USCI1_DAT1	I/O	MFP4	USCI1 数据1脚.	
		UART1_RXD	I	MFP6	UART1 数据接收脚.	
		SD1_CLK	O	MFP7	SD卡接口1 时钟输出脚	
		EBI_nCS1	O	MFP8	EBI 片选 1 输出脚.	
		BPWM1_CH5	I/O	MFP10	BPWM1 通道5 输出/捕获输入.	
		EPWM1_BRAKE1	I	MFP11	EPWM1 刹车1输入脚.	
		EPWM1_CH5	I/O	MFP12	EPWM1 通道5 输出/捕获输入脚.	
1	1	PB.5	I/O	MFP0	GPIO	
		EADC0_CH5	A	MFP1	EADC0 通道5输入脚.	
		ACMP1_N	A	MFP1	模拟比较器1负输入端.	
		EBI_ADR0	O	MFP2	外总线地址0	
		SD0_DAT3	I/O	MFP3	SD卡接口0数据3	
		SPI1_MISO	I/O	MFP5	SPI1 MISO (主输入从输出)	
		I2C0_SCL	I/O	MFP6	I2C0 时钟.	
		UART5_TXD	O	MFP7	UART5 发送端	
		USCI1_CTL0	I/O	MFP8	USCI1 控制0	
		SC0_CLK	O	MFP9	智能卡接口0 时钟	

32 脚	48 脚	64 脚	引脚名	类型	MFP值	描述
			I2S0_BCLK	O	MFP10	I2S0 位时钟
			EPWM0_CH0	I/O	MFP11	EPWM0 通道0
			TM0	I/O	MFP14	定时器0 计数输入/反转输出.
			INT0	I	MFP15	外中断0输入脚
2	2	3	PB.4	I/O	MFP0	GPIO
			EADC0_CH4	A	MFP1	EADC0 通道4输入脚.
			ACMP1_P1	A	MFP1	模拟比较器1正输入端1.
			EBI_ADR1	O	MFP2	外总线地址1
			SD0_DAT2	I/O	MFP3	SD卡接口0数据2
			SPI1_MOSI	I/O	MFP5	SPI1 MOSI (主输出,从输入) .
			I2C0_SDA	I/O	MFP6	I2C0 数据端
			UART5_RXD	I	MFP7	UART5 数据接收.
			USCI1_CTL1	I/O	MFP8	USCI1 控制1
			SC0_DAT	I/O	MFP9	智能卡接口0 数据
			I2S0_MCLK	O	MFP10	I2S0 主时钟.
			EPWM0_CH1	I/O	MFP11	EPWM0 通道1
			TM1	I/O	MFP14	定时器1 计数输入/反转输出
			INT1	I	MFP15	外中断1输入脚
3	3	4	PB.3	I/O	MFP0	GPIO
			EADC0_CH3	A	MFP1	EADC0 通道3输入脚.
			ACMP0_N	A	MFP1	模拟比较器0负输入端.
			EBI_ADR2	O	MFP2	外总线地址2
			SD0_DAT1	I/O	MFP3	SD卡接口0 数据1
			SPI1_CLK	I/O	MFP5	SPI1 时钟
			UART1_TXD	O	MFP6	UART1 发送.
			UART5_nRTS	O	MFP7	UART5 请求发送 RTS
			USCI1_DAT1	I/O	MFP8	USCI1 数据1

32 腳	48 腳	64 腳	引脚名	类型	MFP值	描述
			SC0_RST	O	MFP9	智能卡接口0 复位端.
			I2S0_DI	I	MFP10	I2S0 数据输入
			EPWM0_CH2	I/O	MFP11	EPWM0 通道2
			TM2	I/O	MFP14	定时器2 计数输入/反转输出
			INT2	I	MFP15	外中断2输入脚
			PB.2	I/O	MFP0	GPIO
			EADC0_CH2	A	MFP1	EADC0 通道2输入脚.
			ACMP0_P1	A	MFP1	模拟比较器0正输入端1.
			OPA0_O	A	MFP1	运放0 输出端
			EBI_ADR3	O	MFP2	外总线地址3
			SD0_DAT0	I/O	MFP3	SD卡接口0 数据0
			SPI1_SS	I/O	MFP5	SPI1 从机选择.
			UART1_RXD	I	MFP6	UART1 数据接收.
			UART5_nCTS	I	MFP7	UART5 CTS 清除发送
			USCI1_DAT0	I/O	MFP8	USCI1 数据0.
			SC0_PWR	O	MFP9	智能卡接口0 电源控制端.
			I2S0_DO	O	MFP10	I2S0 数据输出
			EPWM0_CH3	I/O	MFP11	PWM0 通道3
			TM3	I/O	MFP14	定时器3 计数输入/反转输出
			INT3	I	MFP15	外中断3输入脚
			PB.1	I/O	MFP0	GPIO
			EADC0_CH1	A	MFP1	EADC0 通道1输入脚.
			OPA0_N	A	MFP1	运放0 负输入端
			EBI_ADR8	O	MFP2	外总线地址8
			SD0_CLK	O	MFP3	SD卡接口0 时钟
			SPI1_I2SMCLK	I/O	MFP5	SPI1 I2S 主时钟
			SPI3_I2SMCLK	I/O	MFP6	SPI3 I2S 主时钟

32 脚	48 脚	64 脚	引脚名	类型	MFP值	描述
			UART2_TXD	O	MFP7	UART2 发送.
			USCI1_CLK	I/O	MFP8	USCI1 时钟.
			I2C1_SCL	I/O	MFP9	I2C1 时钟.
			I2S0_LRCK	O	MFP10	I2S0 左右声道时钟
			EPWM0_CH4	I/O	MFP11	EPWM0 通道4
			EPWM1_CH4	I/O	MFP12	EPWM1 通道4.
			EPWM0_BRAKE0	I	MFP13	EPWM0 刹车输入引脚0.
			PB.0	I/O	MFP0	GPIO.
			EADC0_CH0	A	MFP1	EADC0 通道0输入脚.
			OPA0_P	A	MFP1	运放0 正输入端
			EBI_ADR9	O	MFP2	外总线地址9
			SD0_CMD	I/O	MFP3	SD卡接口0 命令/响应
			UART2_RXD	I	MFP7	UART2 数据接收.
			SPI0_I2SMCLK	I/O	MFP8	SPI0 I2S 主时钟
			I2C1_SDA	I/O	MFP9	I2C1 数据脚
			EPWM0_CH5	I/O	MFP11	EPWM0 通道5
			EPWM1_CH5	I/O	MFP12	EPWM1 通道5.
			EPWM0_BRAKE1	I	MFP13	EPWM0 刹车输入引脚1
			PA.11	I/O	MFP0	GPIO
			ACMP0_P0	A	MFP1	模拟比较器0正输入端0.
			EBI_nRD	O	MFP2	EBI读.
			SC2_PWR	O	MFP3	智能卡接口2 电源控制脚
			SPI2_SS	I/O	MFP4	SPI2 从机选择.
			SD1_DAT3	I/O	MFP5	SD卡接口1 数据 3.
			USCI0_CLK	I/O	MFP6	USCI0 时钟.
			I2C2_SCL	I/O	MFP7	I2C2 时钟.
			BPWM0_CH0	I/O	MFP9	BPWM0 通道0

32 脚	48 脚	64 脚	引脚名	类型	MFP值	描述
			EPWM0_SYNC_OUT	O	MFP10	EPWM0 计数器同步输出
			TM0_EXT	I/O	MFP13	定时器0 捕获输入.
			DAC1_ST	I	MFP14	DAC1 外部触发输入
			PA.10	I/O	MFP0	GPIO
			ACMP1_P0	A	MFP1	模拟比较器1正输入端0.
			OPA1_O	A	MFP1	运放1 输出脚
			EBI_nWR	O	MFP2	外总线写信号
			SC2_RST	O	MFP3	智能卡接口2 复位 .
			SPI2_CLK	I/O	MFP4	SPI2 时钟
			SD1_DAT2	I/O	MFP5	SD卡接口1 数据 2.
			USCI0_DAT0	I/O	MFP6	USCI0数据线0
			I2C2_SDA	I/O	MFP7	I2C2 数据脚
			BPWM0_CH1	I/O	MFP9	BPWM0 通道1
			QE1_INDEX	I	MFP10	正交编码1 索引输入
			ECAP0_IC0	I	MFP11	增强捕获单元0 输入0.
			TM1_EXT	I/O	MFP13	定时器1事件计数器输入/反转输出
			DAC0_ST	I	MFP14	DAC0 外部触发输入
			PA.9	I/O	MFP0	GPIO.
			OPA1_N	A	MFP1	运放1 负输入端.
			EBI_MCLK	O	MFP2	EBI外部时钟输出脚.
			SC2_DAT	I/O	MFP3	智能卡2 数据脚.
			SPI2_MISO	I/O	MFP4	SPI2 MISO (主机输入, 从机输出) 脚.
			SD1_DAT1	I/O	MFP5	SD卡接口1 数据位1.
			USCI0_DAT1	I/O	MFP6	USCI0 数据1脚.
			UART1_TXD	O	MFP7	UART1 数据发送脚.
			BPWM0_CH2	I/O	MFP9	BPWM0 通道2 输出/捕获输入.
			QE1_A	I	MFP10	正交编码1 计数信号 A 相输入

32 脚	48 脚	64 脚	引脚名	类型	MFP值	描述		
			ECAP0_IC1	I	MFP11	增强型捕获输入单元0输入脚1		
			TM2_EXT	I/O	MFP13	定时器2事件计数器输入/反转输出脚.		
	10	11	PA.8	I/O	MFP0	GPIO.		
			OPA1_P	A	MFP1	运放1 正输入脚.		
			EBI_ALE	O	MFP2	EBI 地址锁存使能输出脚.		
			SC2_CLK	O	MFP3	智能卡 2 时钟脚.		
			SPI2_MOSI	I/O	MFP4	SPI2 MOSI (主机输出, 从机输入) 脚.		
			SD1_DAT0	I/O	MFP5	SD卡接口1 数据位0.		
			USCI0_CTL1	I/O	MFP6	USCI0 控制1 脚..		
			UART1_RXD	I	MFP7	UART1 数据接收脚.		
			BPWM0_CH3	I/O	MFP9	BPWM0 通道3 输出/捕获输入.		
			QEI1_B	I	MFP10	正交编码1 计数信号 B 相输入		
			ECAP0_IC2	I	MFP11	增强型捕获输入单元0输入脚2		
			TM3_EXT	I/O	MFP13	定时器3事件计数器输入/反转输出脚.		
	12		INT4	I	MFP15	外部中断4输入脚.		
			PF.6	I/O	MFP0	GPIO.		
			EBI_ADR19	O	MFP2	EBI 地址总线位19.		
			SC0_CLK	O	MFP3	智能卡 0 时钟脚.		
			I2S0_LRCK	O	MFP4	I2S0 左右声道选择时钟输出脚.		
			SPI0_MOSI	I/O	MFP5	SPI0 MOSI (主机输出, 从机输入) 脚.		
			UART4_RXD	I	MFP6	UART4 数据接收脚.		
			EBI_nCS0	O	MFP7	EBI 片选 0 输出脚.		
	7	11	TAMPER0	I/O	MFP10	TAMPER 倾测循环脚 0.		
			V <sub>DD</sub>	P	MFP0	I/O 端口电源和LDO 电源用于内部 PLL 和数字电路		
			PF.5	I/O	MFP0	GPIO.		
			UART2_RXD	I	MFP2	UART2 数据接收脚.		
			UART2_nCTS	I	MFP4	UART2 清除发送输入脚.		

32 脚	48 脚	64 脚	引脚名	类型	MFP值	描述
			BPWM0_CH4	I/O	MFP8	BPWM0 通道4 输出/捕获输入.
			EPWM0_SYNC_OUT	O	MFP9	EPWM0 计数器同步触发输出脚.
			X32_IN	I	MFP10	外部32.768 kHz 晶振输入脚.
			EADC0_ST	I	MFP11	EADC0 外部触发输入.
8	12	15	PF.4	I/O	MFP0	GPIO.
			UART2_TXD	O	MFP2	UART2 数据发送脚.
			UART2_nRTS	O	MFP4	UART2 请求发送输出脚.
			BPWM0_CH5	I/O	MFP8	BPWM0 通道5 输出/捕获输入.
			X32_OUT	O	MFP10	外部32.768 kHz 晶振输出脚.
9	13	16	PF.3	I/O	MFP0	GPIO.
			EBI_nCS0	O	MFP2	EBI 片选 0 输出脚.
			UART0_TXD	O	MFP3	UART0 数据发送脚.
			I2C0_SCL	I/O	MFP4	I2C0 时钟脚.
			XT1_IN	I	MFP10	外部4~24 MHz (高速) 晶振输入脚.
			BPWM1_CH0	I/O	MFP11	BPWM1 通道0 输出/捕获输入.
10	14	17	PF.2	I/O	MFP0	GPIO.
			EBI_nCS1	O	MFP2	EBI 片选 1 输出脚.
			UART0_RXD	I	MFP3	UART0 数据接收脚.
			I2C0_SDA	I/O	MFP4	I2C0 数据输入/输出引脚.
			QSPI0_CLK	I/O	MFP5	Quad SPI0 串行时钟引脚.
			XT1_OUT	O	MFP10	外部4~24 MHz (高速) 晶振输出引脚.
			BPWM1_CH1	I/O	MFP11	BPWM1 通道1 输出/捕获输入.
		18	PC.7	I/O	MFP0	GPIO.
			EBI_AD9	I/O	MFP2	EBI 地址/数据总线位9.
			SPI1_MISO	I/O	MFP4	SPI1 MISO (主机输入, 从机输出) 脚.
			UART4_TXD	O	MFP5	UART4 数据发送脚.
			SC2_PWR	O	MFP6	智能卡 2 电源脚.

32 脚	48 脚	64 脚	引脚名	类型	MFP值	描述
			UART0_nCTS	I	MFP7	UART0 清除发送输入脚.
			I2C1_SMBAL	O	MFP8	I2C1 SMBus SMBALTER 脚
			EPWM1_CH2	I/O	MFP11	EPWM1 通道2 输出/捕获输入脚.
			BPWM1_CH0	I/O	MFP12	BPWM1 通道0 输出/捕获输入.
			TM0	I/O	MFP14	定时器0事件计数器输入/反转输出脚.
			INT3	I	MFP15	外部中断3输入脚.
		19	PC.6	I/O	MFP0	GPIO.
			EBI_AD8	I/O	MFP2	EBI 地址/数据总线位8.
			SPI1_MOSI	I/O	MFP4	SPI1 MOSI (主机输出, 从机输入) 脚.
			UART4_RXD	I	MFP5	UART4 数据接收脚.
			SC2_RST	O	MFP6	智能卡 2 复位脚.
			UART0_nRTS	O	MFP7	UART0 请求发送输出脚.
			I2C1_SMBSUS	O	MFP8	I2C1 SMBus SMBSUS 脚(PMBus 控制脚)
			EPWM1_CH3	I/O	MFP11	EPWM1 通道3 输出/捕获输入脚.
			BPWM1_CH1	I/O	MFP12	BPWM1 通道1 输出/捕获输入.
			TM1	I/O	MFP14	定时器1事件计数器输入/反转输出脚.
			INT2	I	MFP15	外部中断2输入脚.
		15 20	PA.7	I/O	MFP0	GPIO.
			EBI_AD7	I/O	MFP2	EBI 地址/数据总线位7.
			SPI1_CLK	I/O	MFP4	SPI1 串行时钟引脚.
			SC2_DAT	I/O	MFP6	智能卡2 数据脚.
			UART0_TXD	O	MFP7	UART0 数据发送脚.
			I2C1_SCL	I/O	MFP8	I2C1 时钟脚.
			EPWM1_CH4	I/O	MFP11	EPWM1 通道4 输出/捕获输入脚.
			BPWM1_CH2	I/O	MFP12	BPWM1 通道2 输出/捕获输入.
			ACMP0_WLAT	I	MFP13	模拟比较器0 窗口锁定输入脚
			TM2	I/O	MFP14	定时器2事件计数器输入/反转输出脚.

32 脚	48 脚	64 脚	引脚名	类型	MFP值	描述
			INT1	I	MFP15	外部中断1输入脚.
16	21	PA.6	I/O	MFP0	GPIO.	
		EBI_AD6	I/O	MFP2	EBI 地址/数据总线位6.	
		SPI1_SS	I/O	MFP4	SPI1 从机选择脚.	
		SD1_nCD	I	MFP5	SD卡接口1 卡侦测输入脚	
		SC2_CLK	O	MFP6	智能卡 2 时钟脚.	
		UART0_RXD	I	MFP7	UART0 数据接收脚.	
		I2C1_SDA	I/O	MFP8	I2C1 数据输入/输出引脚.	
		EPWM1_CH5	I/O	MFP11	EPWM1 通道5 输出/捕获输入脚.	
		BPWM1_CH3	I/O	MFP12	BPWM1 通道3 输出/捕获输入.	
		ACMP1_WLAT	I	MFP13	模拟比较器1 窗口锁定输入脚	
		TM3	I/O	MFP14	定时器3事件计数器输入/反转输出脚.	
		INT0	I	MFP15	外部中断0输入脚.	
	22	VSS	P	MFP0	数字电路地.	
	23	VDD	P	MFP0	I/O 端口电源和LDO 电源用于内部 PLL 和数字电路	
	24	LDO_CAP	A	MFP0	LDO 输出脚.	
17	25	PA.5	I/O	MFP0	GPIO.	
		SPIM_D2	I/O	MFP2	SPIM 四线模式数据2 .	
		QSPI0_MISO1	I/O	MFP3	Quad SPI0 MISO1 (主机输入, 从机输出) 脚..	
		SPI1_I2SMCLK	I/O	MFP4	SPI1 I2S 主机时钟输出脚	
		SD1_CMD	I/O	MFP5	SD卡接口1 命令/应答脚	
		SC2_nCD	I	MFP6	智能卡 2 卡侦测脚.	
		UART0_nCTS	I	MFP7	UART0 清除发送输入脚.	
		UART5_TXD	O	MFP8	UART5 数据发送脚.	
		I2C0_SCL	I/O	MFP9	I2C0 时钟脚.	
		BPWM0_CH5	I/O	MFP12	BPWM0 通道5 输出/捕获输入.	
		EPWM0_CH0	I/O	MFP13	EPWM0 通道0 输出/捕获输入脚.	

32 脚	48 脚	64 脚	引脚名	类型	MFP值	描述
			QEIO_INDEX	I	MFP14	正交编码器0索引输入
18	26	PA.4	I/O	MFP0	GPIO.	
		SPI_M3	I/O	MFP2	SPI_M 四线模式数据3 .	
		QSPI0_MOSI1	I/O	MFP3	Quad SPI0 MOSI1 (主机输出, 从机输入) 脚.	
		SPI0_I2SMCLK	I/O	MFP4	SPI0 I2S 主机时钟输出脚	
		SD1_CLK	O	MFP5	SD卡接口1 时钟输出脚	
		SC0_nCD	I	MFP6	智能卡 0 卡侦测脚.	
		UART0_nRTS	O	MFP7	UART0 请求发送输出脚.	
		UART5_RXD	I	MFP8	UART5 数据接收脚.	
		I2C0_SDA	I/O	MFP9	I2C0 数据输入/输出引脚.	
		BPWM0_CH4	I/O	MFP12	BPWM0 通道4 输出/捕获输入.	
11	19	PA.3	I/O	MFP0	GPIO.	
		SPI_M_SS	I/O	MFP2	SPI_M 从机选择脚.	
		QSPI0_SS	I/O	MFP3	Quad SPI0 从机选择脚.	
		SPI0_SS	I/O	MFP4	SPI0 从机选择脚.	
		SD1_DAT3	I/O	MFP5	SD卡接口1 数据位3.	
		SC0_PWR	O	MFP6	智能卡 0 电源脚.	
		UART4_TXD	O	MFP7	UART4 数据发送脚.	
		UART1_TXD	O	MFP8	UART1 数据发送脚.	
		I2C1_SCL	I/O	MFP9	I2C1 时钟脚.	
		BPWM0_CH3	I/O	MFP12	BPWM0 通道3 输出/捕获输入.	
12	20	PA.2	I/O	MFP0	GPIO.	
		SPI_M_CLK	I/O	MFP2	SPI_M 串行时钟引脚.	

32 脚	48 脚	64 脚	引脚名	类型	MFP值	描述
13	21	29	QSPI0_CLK	I/O	MFP3	Quad SPI0 串行时钟引脚.
			SPI0_CLK	I/O	MFP4	SPI0 串行时钟引脚.
			SD1_DAT2	I/O	MFP5	SD卡接口1 数据位2.
			SC0_RST	O	MFP6	智能卡0 复位脚.
			UART4_RXD	I	MFP7	UART4 数据接收脚.
			UART1_RXD	I	MFP8	UART1 数据接收脚.
			I2C1_SDA	I/O	MFP9	I2C1 数据输入/输出引脚.
			BPWM0_CH2	I/O	MFP12	BPWM0 通道2 输出/捕获输入.
			EPWM0_CH3	I/O	MFP13	EPWM0 通道3 输出/捕获输入脚.
			PA.1	I/O	MFP0	GPIO.
14	22	30	SPIM_MISO	I/O	MFP2	SPIM MISO (主机输入, 从机输出) 脚..
			QSPI0_MISO0	I/O	MFP3	Quad SPI0 MISO0 (主机输入, 从机输出) 脚..
			SPI0_MISO	I/O	MFP4	SPI0 MISO (主机输入, 从机输出) 脚.
			SD1_DAT1	I/O	MFP5	SD卡接口1 数据位1.
			SC0_DAT	I/O	MFP6	智能卡0 数据脚.
			UART0_TXD	O	MFP7	UART0 数据发送脚.
			UART1_nCTS	I	MFP8	UART1 清除发送输入脚.
			I2C2_SCL	I/O	MFP9	I2C2 时钟脚.
			BPWM0_CH1	I/O	MFP12	BPWM0 通道1 输出/捕获输入.
			EPWM0_CH4	I/O	MFP13	EPWM0 通道4 输出/捕获输入脚.
			DAC1_ST	I	MFP15	DAC1 外部触发输入.
14	22	30	PA.0	I/O	MFP0	GPIO.
			SPIM_MOSI	I/O	MFP2	SPIM MOSI (主机输出, 从机输入) 脚.
			QSPI0_MOSI0	I/O	MFP3	Quad SPI0 MOSI0 (主机输出, 从机输入) 脚.
			SPI0_MOSI	I/O	MFP4	SPI0 MOSI (主机输出, 从机输入) 脚.
			SD1_DAT0	I/O	MFP5	SD卡接口1 数据位0.
			SC0_CLK	O	MFP6	智能卡0 时钟脚.

32 脚	48 脚	64 脚	引脚名	类型	MFP值	描述
			UART0_RXD	I	MFP7	UART0 数据接收脚.
			UART1_nRTS	O	MFP8	UART1 请求发送输出脚.
			I2C2_SDA	I/O	MFP9	I2C2 数据输入/输出引脚.
			BPWM0_CH0	I/O	MFP12	BPWM0 通道0 输出/捕获输入.
			EPWM0_CH5	I/O	MFP13	EPWM0 通道5 输出/捕获输入脚.
			DAC0_ST	I	MFP15	DAC0 外部触发输入.
15	23	31	VDDIO	P	MFP0	PE.1, PE.8~PE.13的电源.
16	24	32	nRESET	I	MFP0	外部复位输入：低电平有效，带内部上拉.
			PF.0	I/O	MFP0	GPIO.
			UART1_TXD	O	MFP2	UART1 数据发送脚.
			I2C1_SCL	I/O	MFP3	I2C1 时钟脚.
			BPWM1_CH0	I/O	MFP12	BPWM1 通道0 输出/捕获输入.
			ICE_DAT	O	MFP14	串行调试器data 脚.
			PF.1	I/O	MFP0	GPIO.
			UART1_RXD	I	MFP2	UART1 数据接收脚.
			I2C1_SDA	I/O	MFP3	I2C1 数据输入/输出引脚.
			BPWM1_CH1	I/O	MFP12	BPWM1 通道1 输出/捕获输入.
			ICE_CLK	I	MFP14	串行调试器时钟脚.
			PC.5	I/O	MFP0	GPIO.
			EBI_AD5	I/O	MFP2	EBI 地址/数据总线位5.
			SPI_M_D2	I/O	MFP3	SPI_M 四线模式数据2 .
			QSPI0_MISO1	I/O	MFP4	Quad SPI0 MISO1 (主机输入, 从机输出) 脚..
			UART2_TXD	O	MFP8	UART2 数据发送脚.
			I2C1_SCL	I/O	MFP9	I2C1 时钟脚.
			UART4_TXD	O	MFP11	UART4 数据发送脚.
			EPWM1_CH0	I/O	MFP12	EPWM1 通道0 输出/捕获输入脚.
	28	36	PC.4	I/O	MFP0	GPIO.

32 脚	48 脚	64 脚	引脚名	类型	MFP值	描述
			EBI_AD4	I/O	MFP2	EBI 地址/数据总线位4.
			SPIM_D3	I/O	MFP3	SPIM 四线模式数据3 .
			QSPI0_MOSI1	I/O	MFP4	Quad SPI0 MOSI1 (主机输出, 从机输入) 脚.
			SC1_nCD	I	MFP5	智能卡 1 卡侦测脚.
			I2S0_BCLK	O	MFP6	I2S0 位 时钟输出脚.
			SPI1_I2SMCLK	I/O	MFP7	SPI1 I2S 主机时钟输出脚
			UART2_RXD	I	MFP8	UART2 数据接收脚.
			I2C1_SDA	I/O	MFP9	I2C1 数据输入/输出引脚.
			UART4_RXD	I	MFP11	UART4 数据接收脚.
			EPWM1_CH1	I/O	MFP12	EPWM1 通道1 输出/捕获输入脚.
			PC.3	I/O	MFP0	GPIO.
			EBI_AD3	I/O	MFP2	EBI 地址/数据总线位3.
			SPIM_SS	I/O	MFP3	SPIM 从机选择脚.
			QSPI0_SS	I/O	MFP4	Quad SPI0 从机选择脚.
			SC1_PWR	O	MFP5	智能卡 1 电源脚.
			I2S0_MCLK	O	MFP6	I2S0 主机时钟输出脚.
			SPI1_MISO	I/O	MFP7	SPI1 MISO (主机输入, 从机输出) 脚.
			UART2_nRTS	O	MFP8	UART2 请求发送输出脚.
			I2C0_SMBAL	O	MFP9	I2C0 SMBus SMBALTER 脚
			UART3_TXD	O	MFP11	UART3 数据发送脚.
			EPWM1_CH2	I/O	MFP12	EPWM1 通道2 输出/捕获输入脚.
			PC.2	I/O	MFP0	GPIO.
			EBI_AD2	I/O	MFP2	EBI 地址/数据总线位2.
			SPIM_CLK	I/O	MFP3	SPIM 串行时钟引脚.
			QSPI0_CLK	I/O	MFP4	Quad SPI0 串行时钟引脚.
			SC1_RST	O	MFP5	智能卡 1 复位脚.
			I2S0_DI	I	MFP6	I2S0 数据输入脚.

32 脚	48 脚	64 脚	引脚名	类型	MFP值	描述
			SPI1_MOSI	I/O	MFP7	SPI1 MOSI (主机输出, 从机输入) 脚.
			UART2_nCTS	I	MFP8	UART2 清除发送输入脚.
			I2C0_SMBSUS	O	MFP9	I2C0 SMBus SMBSUS 脚(PMBus 控制脚)
			UART3_RXD	I	MFP11	UART3 数据接收脚.
			EPWM1_CH3	I/O	MFP12	EPWM1 通道3 输出/捕获输入脚.
			PC.1	I/O	MFP0	GPIO.
			EBI_AD1	I/O	MFP2	EBI 地址/数据总线位1.
			SPIM_MISO	I/O	MFP3	SPIM MISO (主机输入, 从机输出) 脚..
			QSPI0_MISO0	I/O	MFP4	Quad SPI0 MISO0 (主机输入, 从机输出) 脚..
			SC1_DAT	I/O	MFP5	智能卡1 数据脚.
			I2S0_DO	O	MFP6	I2S0 数据输出脚
			SPI1_CLK	I/O	MFP7	SPI1 串行时钟引脚.
			UART2_TXD	O	MFP8	UART2 数据发送脚.
			I2C0_SCL	I/O	MFP9	I2C0 时钟脚.
			EPWM1_CH4	I/O	MFP12	EPWM1 通道4 输出/捕获输入脚.
			ACMP0_O	O	MFP14	模拟比较器0 输出脚.
			PC.0	I/O	MFP0	GPIO.
			EBI_AD0	I/O	MFP2	EBI 地址/数据总线位0.
			SPIM_MOSI	I/O	MFP3	SPIM MOSI (主机输出, 从机输入) 脚.
			QSPI0_MOSI0	I/O	MFP4	Quad SPI0 MOSI0 (主机输出, 从机输入) 脚.
			SC1_CLK	O	MFP5	智能卡1 时钟脚.
			I2S0_LRCK	O	MFP6	I2S0 左右声道选择时钟输出脚.
			SPI1_SS	I/O	MFP7	SPI1 从机选择脚.
			UART2_RXD	I	MFP8	UART2 数据接收脚.
			I2C0_SDA	I/O	MFP9	I2C0 数据输入/输出引脚.

32 脚	48 脚	64 脚	引脚名	类型	MFP值	描述
		41	PD.3	I/O	MFP0	GPIO.
			EBI_AD10	I/O	MFP2	EBI 地址/数据总线位10.
			USCI0_CTL1	I/O	MFP3	USCI0 控制1 脚..
			SPI0_SS	I/O	MFP4	SPI0 从机选择脚.
			UART3_nRTS	O	MFP5	UART3 请求发送输出脚.
			USCI1_CTL0	I/O	MFP6	USCI1 控制0 脚..
			SC2_PWR	O	MFP7	智能卡 2 电源脚.
			SC1_nCD	I	MFP8	智能卡 1 卡侦测脚.
			UART0_TXD	O	MFP9	UART0 数据发送脚.
		42	PD.2	I/O	MFP0	GPIO.
			EBI_AD11	I/O	MFP2	EBI 地址/数据总线位11.
			USCI0_DAT1	I/O	MFP3	USCI0 数据1脚.
			SPI0_CLK	I/O	MFP4	SPI0 串行时钟引脚.
			UART3_nCTS	I	MFP5	UART3 清除发送输入脚.
			SC2_RST	O	MFP7	智能卡 2 复位脚.
			UART0_RXD	I	MFP9	UART0 数据接收脚.
		43	PD.1	I/O	MFP0	GPIO.
			EBI_AD12	I/O	MFP2	EBI 地址/数据总线位12.
			USCI0_DAT0	I/O	MFP3	USCI0 数据0脚.
			SPI0_MISO	I/O	MFP4	SPI0 MISO (主机输入, 从机输出) 脚.
			UART3_TXD	O	MFP5	UART3 数据发送脚.
			I2C2_SCL	I/O	MFP6	I2C2 时钟脚.
			SC2_DAT	I/O	MFP7	智能卡2 数据脚.
		44	PD.0	I/O	MFP0	GPIO.
			EBI_AD13	I/O	MFP2	EBI 地址/数据总线位13.
			USCI0_CLK	I/O	MFP3	USCI0 时钟脚.
			SPI0_MOSI	I/O	MFP4	SPI0 MOSI (主机输出, 从机输入) 脚.

32 脚	48 脚	64 脚	引脚名	类型	MFP值	描述
21	33	45	UART3_RXD	I	MFP5	UART3 数据接收脚.
			I2C2_SDA	I/O	MFP6	I2C2 数据输入/输出引脚.
			SC2_CLK	O	MFP7	智能卡 2 时钟脚.
			TM2	I/O	MFP14	定时器2事件计数器输入/反转输出脚.
22	34	46	PA.12	I/O	MFP0	GPIO.
			I2S0_BCLK	O	MFP2	I2S0 bit 时钟输出脚.
			UART4_TXD	O	MFP3	UART4 数据发送脚.
			I2C1_SCL	I/O	MFP4	I2C1 时钟脚.
			SPI2_SS	I/O	MFP5	SPI2 从机选择脚.
			SC2_PWR	O	MFP7	智能卡 2 电源脚.
			BPWM1_CH2	I/O	MFP11	BPWM1 通道2 输出/捕获输入.
			QEI1_INDEX	I	MFP12	正交编码器1索引输入
23	35	47	PA.13	I/O	MFP0	GPIO.
			I2S0_MCLK	O	MFP2	I2S0 主机时钟输出脚.
			UART4_RXD	I	MFP3	UART4 数据接收脚.
			I2C1_SDA	I/O	MFP4	I2C1 数据输入/输出引脚.
			SPI2_CLK	I/O	MFP5	SPI2 串行时钟引脚.
			SC2_RST	O	MFP7	智能卡 2 复位脚.
			BPWM1_CH3	I/O	MFP11	BPWM1 通道3 输出/捕获输入.
			QEI1_A	I	MFP12	正交编码1 计数信号 A 相输入

32 脚	48 脚	64 脚	引脚名	类型	MFP值	描述
			QEI1_B	I	MFP12	正交编码1 计数信号 B 相输入
24	36	48	PA.15	I/O	MFP0	GPIO.
			I2S0_DO	O	MFP2	I2S0 数据输出脚
			UART0_RXD	I	MFP3	UART0 数据接收脚.
			SPI2_MOSI	I/O	MFP5	SPI2 MOSI (主机输出, 从机输入) 脚.
			I2C2_SDA	I/O	MFP6	I2C2 数据输入/输出引脚.
			SC2_CLK	O	MFP7	智能卡 2 时钟脚.
			BPWM1_CH5	I/O	MFP11	BPWM1 通道5 输出/捕获输入.
			EPWM0_SYNC_IN	I	MFP12	EPWM0 计数器同步触发输入脚
25	37	49	VSS	P	MFP0	数字电路地.
26	38	50	LDO_CAP	A	MFP0	LDO 输出脚.
27	39	51	VDD	P	MFP0	I/O 端口电源和LDO 电源用于内部 PLL 和数字电路
28	41	53	PC.14	I/O	MFP0	GPIO.
			EBI_AD11	I/O	MFP2	EBI 地址/数据总线位11.
			SC1_nCD	I	MFP3	智能卡 1 卡侦测脚.
			SPI0_I2SMCLK	I/O	MFP4	SPI0 I2S 主机时钟输出脚
			USCI0_CTL0	I/O	MFP5	USCI0 控制0 脚..
			QSPI0_CLK	I/O	MFP6	Quad SPI0 串行时钟引脚.
			EPWM0_SYNC_IN	I	MFP11	EPWM0 计数器同步触发输入脚
			TM1	I/O	MFP13	定时器1事件计数器输入/反转输出脚.
28	41	53	PB.15	I/O	MFP0	GPIO.
			EADC0_CH15	A	MFP1	EADC0 通道15模拟输入.
			EBI_AD12	I/O	MFP2	EBI 地址/数据总线位12.
			SC1_PWR	O	MFP3	智能卡 1 电源脚.
			SPI0_SS	I/O	MFP4	SPI0 从机选择脚.
			USCI0_CTL1	I/O	MFP5	USCI0 控制1 脚..
			UART0_nCTS	I	MFP6	UART0 清除发送输入脚.

32 脚	48 脚	64 脚	引脚名	类型	MFP值	描述
29	42	54	UART3_TXD	O	MFP7	UART3 数据发送脚.
			I2C2_SMBAL	O	MFP8	I2C2 SMBus SMBALTER 脚
			EPWM1_CH0	I/O	MFP11	EPWM1 通道0 输出/捕获输入脚.
			TM0_EXT	I/O	MFP13	定时器0事件计数器输入/反转输出脚.
			PB.14	I/O	MFP0	GPIO.
			EADC0_CH14	A	MFP1	EADC0 通道14模拟输入.
			EBI_AD13	I/O	MFP2	EBI 地址/数据总线位13.
			SC1_RST	O	MFP3	智能卡 1 复位脚.
			SPI0_CLK	I/O	MFP4	SPI0 串行时钟引脚.
			USCI0_DAT1	I/O	MFP5	USCI0 数据1脚.
30	43	55	UART0_nRTS	O	MFP6	UART0 请求发送输出脚.
			UART3_RXD	I	MFP7	UART3 数据接收脚.
			I2C2_SMBSUS	O	MFP8	I2C2 SMBus SMBSUS 脚(PMBus 控制脚)
			EPWM1_CH1	I/O	MFP11	EPWM1 通道1 输出/捕获输入脚.
			TM1_EXT	I/O	MFP13	定时器1事件计数器输入/反转输出脚.
			CLKO	O	MFP14	时钟输出
			PB.13	I/O	MFP0	GPIO.
			EADC0_CH13	A	MFP1	EADC0 通道13模拟输入.
			DAC1_OUT	A	MFP1	DAC1 通道模拟输出
			ACMP0_P3	A	MFP1	模拟比较器0 正输入 3 脚.

32 脚	48 脚	64 脚	引脚名	类型	MFP值	描述
			I2C2_SCL	I/O	MFP8	I2C2 时钟脚.
			EPWM1_CH2	I/O	MFP11	EPWM1 通道2 输出/捕获输入脚.
			TM2_EXT	I/O	MFP13	定时器2事件计数器输入/反转输出脚.
			PB.12	I/O	MFP0	GPIO.
			EADC0_CH12	A	MFP1	EADC0 通道12模拟输入.
			DAC0_OUT	A	MFP1	DAC0 通道模拟输出
			ACMP0_P2	A	MFP1	模拟比较器0 正输入 2 脚.
			ACMP1_P2	A	MFP1	模拟比较器1 正输入 2 脚.
			EBI_AD15	I/O	MFP2	EBI 地址/数据总线位15.
			SC1_CLK	O	MFP3	智能卡 1 时钟脚.
			SPI0_MOSI	I/O	MFP4	SPI0 MOSI (主机输出, 从机输入) 脚.
			USCI0_CLK	I/O	MFP5	USCI0 时钟脚.
			UART0_RXD	I	MFP6	UART0 数据接收脚.
			UART3_nCTS	I	MFP7	UART3 清除发送输入脚.
			I2C2_SDA	I/O	MFP8	I2C2 数据输入/输出引脚.
			SD0_nCD	I	MFP9	SD卡接口0 卡侦测输入脚
			EPWM1_CH3	I/O	MFP11	EPWM1 通道3 输出/捕获输入脚.
			TM3_EXT	I/O	MFP13	定时器3事件计数器输入/反转输出脚.
32	45	57	AVDD	P	MFP0	内部模拟电路电源 .
		58	VREF	A	MFP0	ADC 参考电压输入 注意:该脚需要接一个1uF 电容.
	46	59	AVSS	P	MFP0	模拟电路地
			PB.11	I/O	MFP0	GPIO.
			EADC0_CH11	A	MFP1	EADC0 通道11模拟输入.
			EBI_ADR16	O	MFP2	EBI 地址总线位16.
			UART0_nCTS	I	MFP5	UART0 清除发送输入脚.
			UART4_TXD	O	MFP6	UART4 数据发送脚.

32 脚	48 脚	64 脚	引脚名	类型	MFP值	描述
			I2C1_SCL	I/O	MFP7	I2C1 时钟脚.
			SPI0_I2SMCLK	I/O	MFP9	SPI0 I2S 主机时钟输出脚
			BPWM1_CH0	I/O	MFP10	BPWM1 通道0 输出/捕获输入.
			SPI3_CLK	I/O	MFP11	SPI3 串行时钟引脚.
		61	PB.10	I/O	MFP0	GPIO.
			EADC0_CH10	A	MFP1	EADC0 通道10模拟输入.
			EBI_ADR17	O	MFP2	EBI 地址总线位17.
			USCI1_CTL0	I/O	MFP4	USCI1 控制0 脚..
			UART0_nRTS	O	MFP5	UART0 请求发送输出脚.
			UART4_RXD	I	MFP6	UART4 数据接收脚.
			I2C1_SDA	I/O	MFP7	I2C1 数据输入/输出引脚.
			BPWM1_CH1	I/O	MFP10	BPWM1 通道1 输出/捕获输入.
			SPI3_SS	I/O	MFP11	SPI3 从机选择脚.
		62	PB.9	I/O	MFP0	GPIO.
			EADC0_CH9	A	MFP1	EADC0 通道9模拟输入.
			EBI_ADR18	O	MFP2	EBI 地址总线位18.
			USCI1_CTL1	I/O	MFP4	USCI1 控制1 脚..
			UART0_TXD	O	MFP5	UART0 数据发送脚.
			UART1_nCTS	I	MFP6	UART1 清除发送输入脚.
			I2C1_SMBAL	O	MFP7	I2C1 SMBus SMBALTER 脚
			BPWM1_CH2	I/O	MFP10	BPWM1 通道2 输出/捕获输入.
			SPI3_MISO	I/O	MFP11	SPI3 MISO (主机输入, 从机输出) 脚..
			INT7	I	MFP13	外部中断7输入脚.
		63	PB.8	I/O	MFP0	GPIO.
			EADC0_CH8	A	MFP1	EADC0 通道8模拟输入.
			EBI_ADR19	O	MFP2	EBI 地址总线位19.
			USCI1_CLK	I/O	MFP4	USCI1 时钟脚.

32 脚	48 脚	64 脚	引脚名	类型	MFP值	描述
			UART0_RXD	I	MFP5	UART0 数据接收脚.
			UART1_nRTS	O	MFP6	UART1 请求发送输出脚.
			I2C1_SMBSUS	O	MFP7	I2C1 SMBus SMBSUS 脚(PMBus 控制脚)
			BPWM1_CH3	I/O	MFP10	BPWM1 通道3 输出/捕获输入.
			SPI3_MOSI	I/O	MFP11	SPI3 MOSI (主机输出, 从机输入) 脚.
			INT6	I	MFP13	外部中断6输入脚.
			PB.7	I/O	MFP0	GPIO.
			EADC0_CH7	A	MFP1	EADC0 通道7模拟输入.
			EBI_nWRL	O	MFP2	EBI 低字节写使能输出脚.
			USCI1_DAT0	I/O	MFP4	USCI1 数据0脚.
			UART1_TXD	O	MFP6	UART1 数据发送脚.
			SD1_CMD	I/O	MFP7	SD卡接口1 命令/应答脚
			EBI_nCS0	O	MFP8	EBI 片选0 输出脚.
			BPWM1_CH4	I/O	MFP10	BPWM1 通道4 输出/捕获输入.
			EPWM1_BRAKE0	I	MFP11	EPWM1 刹车0输入脚.
			EPWM1_CH4	I/O	MFP12	EPWM1 通道4 输出/捕获输入脚.
			INT5	I	MFP13	外部中断5输入脚.
			ACMP0_O	O	MFP15	模拟比较器0 输出脚.

## 4.2.2 M482 系列引脚描述

32脚	48脚	64脚	128脚	引脚名	类型	MFP值	描述
1	1	2	1	PB.5	I/O	MFP0	GPIO.
				EADC0_CH5	A	MFP1	EADC0 通道5模拟输入.
				ACMP1_N	A	MFP1	模拟比较器1 负输入脚.
				EBI_ADR0	O	MFP2	EBI 地址总线位0.
				SD0_DAT3	I/O	MFP3	SD卡接口0 数据位3.
				SPI1_MISO	I/O	MFP5	SPI1 MISO (主机输入, 从机输出) 脚.
				I2C0_SCL	I/O	MFP6	I2C0 时钟脚.
				UART5_TXD	O	MFP7	UART5 数据发送脚.
				USCI1_CTL0	I/O	MFP8	USCI1 控制0 脚..
				SC0_CLK	O	MFP9	智能卡 0 时钟脚.
				I2S0_BCLK	O	MFP10	I2S0 位 时钟输出脚.
2	2	3	2	PB.4	I/O	MFP0	GPIO.
				EADC0_CH4	A	MFP1	EADC0 通道4模拟输入.
				ACMP1_P1	A	MFP1	模拟比较器1 正输入 1 脚.
				EBI_ADR1	O	MFP2	EBI 地址总线位1.
				SD0_DAT2	I/O	MFP3	SD卡接口0 数据位2.
				SPI1_MOSI	I/O	MFP5	SPI1 MOSI (主机输出, 从机输入) 脚.
				I2C0_SDA	I/O	MFP6	I2C0 数据输入/输出引脚.
				UART5_RXD	I	MFP7	UART5 数据接收脚.
				USCI1_CTL1	I/O	MFP8	USCI1 控制1 脚..
				SC0_DAT	I/O	MFP9	智能卡0 数据脚.
				I2S0_MCLK	O	MFP10	I2S0 主机时钟输出脚.
				EPWM0_CH1	I/O	MFP11	EPWM0 通道1 输出/捕获输入脚.
				TM1	I/O	MFP14	定时器1事件计数器输入/反转输出脚.
				INT1	I	MFP15	外部中断1输入脚.

32脚	48脚	64脚	128脚	引脚名	类型	MFP值	描述
3	3	4	3	PB.3	I/O	MFP0	GPIO.
				EADC0_CH3	A	MFP1	EADC0 通道3模拟输入.
				ACMP0_N	A	MFP1	模拟比较器0 负输入脚.
				EBI_ADR2	O	MFP2	EBI 地址总线位2.
				SD0_DAT1	I/O	MFP3	SD卡接口0 数据位1.
				SPI1_CLK	I/O	MFP5	SPI1 串行时钟引脚.
				UART1_TXD	O	MFP6	UART1 数据发送脚.
				UART5_nRTS	O	MFP7	UART5 请求发送输出脚.
				USCI1_DAT1	I/O	MFP8	USCI1 数据1脚.
				SC0_RST	O	MFP9	智能卡0 复位脚.
				I2S0_DI	I	MFP10	I2S0 数据输入脚.
				EPWM0_CH2	I/O	MFP11	EPWM0 通道2 输出/捕获输入脚.
				TM2	I/O	MFP14	定时器2事件计数器输入/反转输出脚.
				INT2	I	MFP15	外部中断2输入脚.
4	4	5	4	PB.2	I/O	MFP0	GPIO.
				EADC0_CH2	A	MFP1	EADC0 通道2模拟输入.
				ACMP0_P1	A	MFP1	模拟比较器0 正输入1脚.
				OPA0_O	A	MFP1	运放0 输出脚.
				EBI_ADR3	O	MFP2	EBI 地址总线位3.
				SD0_DAT0	I/O	MFP3	SD卡接口0 数据位0.
				SPI1_SS	I/O	MFP5	SPI1 从机选择脚.
				UART1_RXD	I	MFP6	UART1 数据接收脚.
				UART5_nCTS	I	MFP7	UART5 清除发送输入脚.
				USCI1_DAT0	I/O	MFP8	USCI1 数据0脚.
				SC0_PWR	O	MFP9	智能卡0 电源脚.
				I2S0_DO	O	MFP10	I2S0 数据输出脚
				EPWM0_CH3	I/O	MFP11	EPWM0 通道3 输出/捕获输入脚.
				TM3	I/O	MFP14	定时器3事件计数器输入/反转输出脚.
				INT3	I	MFP15	外部中断3输入脚.

32 脚	48 脚	64 脚	128 脚	引脚名	类型	MFP值	描述
			5	PC.12	I/O	MFP0	GPIO.
				EBI_ADR4	O	MFP2	EBI 地址总线位4.
				UART0_TXD	O	MFP3	UART0 数据发送脚.
				I2C0_SCL	I/O	MFP4	I2C0 时钟脚.
				SPI3_MISO	I/O	MFP6	SPI3 MISO (主机输入, 从机输出) 脚..
				SC0_nCD	I	MFP9	智能卡 0 卡侦测脚.
				ECAP1_IC2	I	MFP11	增强型捕获输入单元1输入脚2
				EPWM1_CH0	I/O	MFP12	EPWM1 通道0 输出/捕获输入脚.
				ACMP0_O	O	MFP14	模拟比较器0 输出脚.
			6	PC.11	I/O	MFP0	GPIO.
				EBI_ADR5	O	MFP2	EBI 地址总线位5.
				UART0_RXD	I	MFP3	UART0 数据接收脚.
				I2C0_SDA	I/O	MFP4	I2C0 数据输入/输出引脚.
				SPI3_MOSI	I/O	MFP6	SPI3 MOSI (主机输出, 从机输入) 脚.
				ECAP1_IC1	I	MFP11	增强型捕获输入单元1输入脚1
				EPWM1_CH1	I/O	MFP12	EPWM1 通道1 输出/捕获输入脚.
				ACMP1_O	O	MFP14	模拟比较器1 输出脚.
			7	PC.10	I/O	MFP0	GPIO.
				EBI_ADR6	O	MFP2	EBI 地址总线位 6.
				SPI3_CLK	I/O	MFP6	SPI3 串行时钟引脚.
				UART3_TXD	O	MFP7	UART3 数据发送脚.
				CAN1_TXD	O	MFP9	CAN1 总线发送输出.
				ECAP1_IC0	I	MFP11	增强型捕获输入单元1输入脚0
				EPWM1_CH2	I/O	MFP12	EPWM1 通道2 输出/捕获输入脚.
			8	PC.9	I/O	MFP0	GPIO.
				EBI_ADR7	O	MFP2	EBI 地址总线位 7.
				SPI3_SS	I/O	MFP6	SPI3 从机选择脚.
				UART3_RXD	I	MFP7	UART3 数据接收脚.
				CAN1_RXD	I	MFP9	CAN1 总线接收输入

32 脚	48 脚	64 脚	128 脚	引脚名	类型	MFP值	描述
				EPWM1_CH3	I/O	MFP12	EPWM1 通道3 输出/捕获输入脚.
5 5 6 9	5 5 6 9	6 6 7 10	9	PB.1	I/O	MFP0	GPIO.
				EADC0_CH1	A	MFP1	EADC0 通道1模拟输入.
				OPA0_N	A	MFP1	运放0 负输入脚.
				EBI_ADR8	O	MFP2	EBI 地址总线位 8.
				SD0_CLK	O	MFP3	SD卡接口0 时钟输出脚
				SPI1_I2SMCLK	I/O	MFP5	SPI1 I2S 主机时钟输出脚
				SPI3_I2SMCLK	I/O	MFP6	SPI3 I2S 主机时钟输出脚
				UART2_TXD	O	MFP7	UART2 数据发送脚.
				USCI1_CLK	I/O	MFP8	USCI1 时钟脚.
				I2C1_SCL	I/O	MFP9	I2C1 时钟脚.
				I2S0_LRCK	O	MFP10	I2S0 左右声道选择时钟输出脚.
				EPWM0_CH4	I/O	MFP11	EPWM0 通道4 输出/捕获输入脚.
				EPWM1_CH4	I/O	MFP12	EPWM1 通道4 输出/捕获输入脚.
				EPWM0_BRAKE0	I	MFP13	EPWM0 刹车0输入脚.
6 6 7 10	6 6 7 10	7 7 8 9	10	PB.0	I/O	MFP0	GPIO.
				EADC0_CH0	A	MFP1	EADC0 通道0模拟输入.
				OPA0_P	A	MFP1	运放0 正输入脚.
				EBI_ADR9	O	MFP2	EBI 地址总线位 9.
				SD0_CMD	I/O	MFP3	SD卡接口0 命令/应答脚
				UART2_RXD	I	MFP7	UART2 数据接收脚.
				SPI0_I2SMCLK	I/O	MFP8	SPI0 I2S 主机时钟输出脚
				I2C1_SDA	I/O	MFP9	I2C1 数据输入/输出引脚.
				EPWM0_CH5	I/O	MFP11	EPWM0 通道5 输出/捕获输入脚.
				EPWM1_CH5	I/O	MFP12	EPWM1 通道5 输出/捕获输入脚.
				EPWM0_BRAKE1	I	MFP13	EPWM0 刹车1输入脚.
			11	VSS	P	MFP0	数字电路地.
			12	V <sub>DD</sub>	P	MFP0	I/O 端口电源和LDO 电源用于内部 PLL 和数字电路
7	8	13		PA.11	I/O	MFP0	GPIO.

32脚	48脚	64脚	128脚	引脚名	类型	MFP值	描述
				ACMP0_P0	A	MFP1	模拟比较器0 正输入 0 脚.
				EBI_nRD	O	MFP2	EBI 读使能输出脚.
				SC2_PWR	O	MFP3	智能卡 2 电源脚.
				SPI2_SS	I/O	MFP4	SPI2 从机选择脚.
				SD1_DAT3	I/O	MFP5	SD卡接口1 数据位3.
				USCI0_CLK	I/O	MFP6	USCI0 时钟脚.
				I2C2_SCL	I/O	MFP7	I2C2 时钟脚.
				BPWM0_CH0	I/O	MFP9	BPWM0 通道0 输出/捕获输入.
				EPWM0_SYNC_OUT	O	MFP10	EPWM0 计数器同步触发输出脚.
				TM0_EXT	I/O	MFP13	定时器0事件计数器输入/反转输出脚.
				DAC1_ST	I	MFP14	DAC1 外部触发输入.
				PA.10	I/O	MFP0	GPIO.
				ACMP1_P0	A	MFP1	模拟比较器1 正输入 0 脚.
				OPA1_O	A	MFP1	运放 1 输出脚.
				EBI_nWR	O	MFP2	EBI 写使能输出脚.
				SC2_RST	O	MFP3	智能卡 2 复位脚.
				SPI2_CLK	I/O	MFP4	SPI2 串行时钟引脚.
				SD1_DAT2	I/O	MFP5	SD卡接口1 数据位2.
				USCI0_DAT0	I/O	MFP6	USCI0 数据0脚.
				I2C2_SDA	I/O	MFP7	I2C2 数据输入/输出引脚.
				BPWM0_CH1	I/O	MFP9	BPWM0 通道1 输出/捕获输入.
				QEII_INDEX	I	MFP10	正交编码器1索引输入
				ECAPO_IC0	I	MFP11	增强型捕获输入单元0输入脚0
				TM1_EXT	I/O	MFP13	定时器1事件计数器输入/反转输出脚.
				DAC0_ST	I	MFP14	DAC0 外部触发输入.
				PA.9	I/O	MFP0	GPIO.
				OPA1_N	A	MFP1	运放1 负输入端.
				EBI_MCLK	O	MFP2	EBI外部时钟输出脚.
				SC2_DAT	I/O	MFP3	智能卡2 数据脚.

32 脚	48 脚	64 脚	128 脚	引脚名	类型	MFP值	描述
				SPI2_MISO	I/O	MFP4	SPI2 MISO (主机输入, 从机输出) 脚.
				SD1_DAT1	I/O	MFP5	SD卡接口1 数据位1.
				USCI0_DAT1	I/O	MFP6	USCI0 数据1脚.
				UART1_TXD	O	MFP7	UART1 数据发送脚.
				BPWM0_CH2	I/O	MFP9	BPWM0 通道2 输出/捕获输入.
				QEI1_A	I	MFP10	正交编码1 计数信号 A 相输入
				ECAPO_IC1	I	MFP11	增强型捕获输入单元0输入脚1
				TM2_EXT	I/O	MFP13	定时器2事件计数器输入/反转输出脚.
				PA.8	I/O	MFP0	GPIO.
				OPA1_P	A	MFP1	运放1 正输入脚.
				EBI_ALE	O	MFP2	EBI 地址锁存使能输出脚.
				SC2_CLK	O	MFP3	智能卡 2 时钟脚.
				SPI2_MOSI	I/O	MFP4	SPI2 MOSI (主机输出, 从机输入) 脚.
				SD1_DAT0	I/O	MFP5	SD卡接口1 数据位0.
				USCI0_CTL1	I/O	MFP6	USCI0 控制1 脚..
				UART1_RXD	I	MFP7	UART1 数据接收脚.
				BPWM0_CH3	I/O	MFP9	BPWM0 通道3 输出/捕获输入.
				QEI1_B	I	MFP10	正交编码1 计数信号 B 相输入
				ECAPO_IC2	I	MFP11	增强型捕获输入单元0输入脚2
				TM3_EXT	I/O	MFP13	定时器3事件计数器输入/反转输出脚.
				INT4	I	MFP15	外部中断4输入脚.
				PC.13	I/O	MFP0	GPIO.
				EBI_ADR10	O	MFP2	EBI 地址总线位10.
				SC2_nCD	I	MFP3	智能卡 2 卡侦测脚.
				SPI2_I2SMCLK	I/O	MFP4	SPI2 I2S 主机时钟输出脚
				CAN1_TXD	O	MFP5	CAN1 总线发送输出.
				USCI0_CTL0	I/O	MFP6	USCI0 控制0 脚..
				UART2_TXD	O	MFP7	UART2 数据发送脚.
				BPWM0_CH4	I/O	MFP9	BPWM0 通道4 输出/捕获输入.

32 脚	48 脚	64 脚	128 脚	引脚名	类型	MFP值	描述
				CLKO	O	MFP13	时钟输出
				EADC0_ST	I	MFP14	EADC0 外部触发输入.
			18	PD.12	I/O	MFP0	GPIO.
				OPA2_O	A	MFP1	运放 2 输出脚.
				EBI_nCS0	O	MFP2	EBI 片选 0 输出脚.
				CAN1_RXD	I	MFP5	CAN1 总线接收输入
				UART2_RXD	I	MFP7	UART2 数据接收脚.
				BPWM0_CH5	I/O	MFP9	BPWM0 通道5 输出/捕获输入.
				QEIO_INDEX	I	MFP10	正交编码器0索引输入
				CLKO	O	MFP13	时钟输出
				EADC0_ST	I	MFP14	EADC0 外部触发输入.
				INT5	I	MFP15	外部中断5输入脚.
			19	PD.11	I/O	MFP0	GPIO.
				OPA2_N	A	MFP1	运放 2 负输入脚.
				EBI_nCS1	O	MFP2	EBI 片选 1 输出脚.
				UART1_TXD	O	MFP3	UART1 数据发送脚.
				CAN0_TXD	O	MFP4	CAN0 总线发送输出.
				QEIO_A	I	MFP10	正交编码0 计数信号 A 相输入
				INT6	I	MFP15	外部中断6输入脚.
			20	PD.10	I/O	MFP0	GPIO.
				OPA2_P	A	MFP1	运放 2 正输入脚.
				EBI_nCS2	O	MFP2	EBI 片选 2 输出脚.
				UART1_RXD	I	MFP3	UART1 数据接收脚.
				CAN0_RXD	I	MFP4	CAN0 总线接收输入
				QEIO_B	I	MFP10	正交编码0 计数信号 B 相输入
				INT7	I	MFP15	外部中断7输入脚.
			21	PG.2	I/O	MFP0	GPIO.
				EBI_ADR11	O	MFP2	EBI 地址总线位11.
				SPI2_SS	I/O	MFP3	SPI2 从机选择脚.

32 脚	48 脚	64 脚	128 脚	引脚名	类型	MFP值	描述
				I2C0_SMBAL	O	MFP4	I2C0 SMBus SMBALTER 脚
				I2C1_SCL	I/O	MFP5	I2C1 时钟脚.
				TM0	I/O	MFP13	定时器0事件计数器输入/反转输出脚.
			22	PG.3	I/O	MFP0	GPIO.
				EBI_ADR12	O	MFP2	EBI 地址总线位12.
				SPI2_CLK	I/O	MFP3	SPI2 串行时钟引脚.
				I2C0_SMBSUS	O	MFP4	I2C0 SMBus SMBSUS 脚(PMBus 控制脚)
				I2C1_SDA	I/O	MFP5	I2C1 数据输入/输出引脚.
				TM1	I/O	MFP13	定时器1事件计数器输入/反转输出脚.
			23	PG.4	I/O	MFP0	GPIO.
				EBI_ADR13	O	MFP2	EBI 地址总线位13.
				SPI2_MISO	I/O	MFP3	SPI2 MISO (主机输入, 从机输出) 脚.
				TM2	I/O	MFP13	定时器2事件计数器输入/反转输出脚.
			24	PF.11	I/O	MFP0	GPIO.
				EBI_ADR14	O	MFP2	EBI 地址总线位14.
				SPI2_MOSI	I/O	MFP3	SPI2 MOSI (主机输出, 从机输入) 脚.
				TAMPER5	I/O	MFP10	TAMPER 侦测循环脚 5.
				TM3	I/O	MFP13	定时器3事件计数器输入/反转输出脚.
			25	PF.10	I/O	MFP0	GPIO.
				EBI_ADR15	O	MFP2	EBI 地址总线位15.
				SC0_nCD	I	MFP3	智能卡 0 卡侦测脚.
				I2S0_BCLK	O	MFP4	I2S0 位 时钟输出脚.
				SPI0_I2SMCLK	I/O	MFP5	SPI0 I2S 主机时钟输出脚
				TAMPER4	I/O	MFP10	TAMPER 侦测循环脚 4.
			26	PF.9	I/O	MFP0	GPIO.
				EBI_ADR16	O	MFP2	EBI 地址总线位16.
				SC0_PWR	O	MFP3	智能卡 0 电源脚.
				I2S0_MCLK	O	MFP4	I2S0 主机时钟输出脚.
				SPI0_SS	I/O	MFP5	SPI0 从机选择脚.

32 脚	48 脚	64 脚	128 脚	引脚名	类型	MFP值	描述	
				TAMPER3	I/O	MFP10	TAMPER 值测循环脚 3.	
			27	PF.8	I/O	MFP0	GPIO.	
				EBI_ADR17	O	MFP2	EBI 地址总线位17.	
				SC0_RST	O	MFP3	智能卡 0 复位脚.	
				I2S0_DI	I	MFP4	I2S0 数据输入脚.	
				SPI0_CLK	I/O	MFP5	SPI0 串行时钟引脚.	
				TAMPER2	I/O	MFP10	TAMPER 值测循环脚 2.	
			28	PF.7	I/O	MFP0	GPIO.	
				EBI_ADR18	O	MFP2	EBI 地址总线位18.	
				SC0_DAT	I/O	MFP3	智能卡0 数据脚.	
				I2S0_DO	O	MFP4	I2S0 数据输出脚	
				SPI0_MISO	I/O	MFP5	SPI0 MISO (主机输入, 从机输出) 脚.	
				UART4_TXD	O	MFP6	UART4 数据发送脚.	
				TAMPER1	I/O	MFP10	TAMPER 值测循环脚 1.	
			12	29	PF.6	I/O	MFP0	GPIO.
					O	MFP2	EBI 地址总线位19.	
					O	MFP3	智能卡 0 时钟脚.	
					O	MFP4	I2S0 左右声道选择时钟输出脚.	
					I/O	MFP5	SPI0 MOSI (主机输出, 从机输入) 脚.	
					I	MFP6	UART4 数据接收脚.	
					O	MFP7	EBI 片选 0 输出脚.	
					I/O	MFP10	TAMPER 值测循环脚 0.	
		13	30	V <sub>DD</sub>	P	MFP0	I/O 端口电源和LDO 电源用于内部 PLL 和数字电路	
7	11	14	31	PF.5	I/O	MFP0	GPIO.	
				UART2_RXD	I	MFP2	UART2 数据接收脚.	
				UART2_nCTS	I	MFP4	UART2 清除发送输入脚.	
				BPWM0_CH4	I/O	MFP8	BPWM0 通道4 输出/捕获输入.	
				EPWM0_SYNC_OUT	O	MFP9	EPWM0 计数器同步触发输出脚.	
				X32_IN	I	MFP10	外部32.768 kHz 晶振输入脚.	

32 脚	48 脚	64 脚	128 脚	引脚名	类型	MFP值	描述
				EADC0_ST	I	MFP11	EADC0 外部触发输入.
8	12	15	32	PF.4	I/O	MFP0	GPIO.
				UART2_TXD	O	MFP2	UART2 数据发送脚.
				UART2_nRTS	O	MFP4	UART2 请求发送输出脚.
				BPWM0_CH5	I/O	MFP8	BPWM0 通道5 输出/捕获输入.
				X32_OUT	O	MFP10	外部32.768 kHz 晶振输出脚.
			33	PH.4	I/O	MFP0	GPIO.
				EBI_ADR3	O	MFP2	EBI 地址总线位3.
				SPI1_MISO	I/O	MFP3	SPI1 MISO (主机输入, 从机输出) 脚.
			34	PH.5	I/O	MFP0	GPIO.
				EBI_ADR2	O	MFP2	EBI 地址总线位2.
				SPI1_MOSI	I/O	MFP3	SPI1 MOSI (主机输出, 从机输入) 脚.
			35	PH.6	I/O	MFP0	GPIO.
				EBI_ADR1	O	MFP2	EBI 地址总线位1.
				SPI1_CLK	I/O	MFP3	SPI1 串行时钟引脚.
			36	PH.7	I/O	MFP0	GPIO.
				EBI_ADR0	O	MFP2	EBI 地址总线位0.
				SPI1_SS	I/O	MFP3	SPI1 从机选择脚.
9	13	16	37	PF.3	I/O	MFP0	GPIO.
				EBI_nCS0	O	MFP2	EBI 片选 0 输出脚.
				UART0_TXD	O	MFP3	UART0 数据发送脚.
				I2C0_SCL	I/O	MFP4	I2C0 时钟脚.
				XT1_IN	I	MFP10	外部4~24 MHz (高速) 晶振输入脚.
				BPWM1_CH0	I/O	MFP11	BPWM1 通道0 输出/捕获输入.
10	14	17	38	PF.2	I/O	MFP0	GPIO.
				EBI_nCS1	O	MFP2	EBI 片选 1 输出脚.
				UART0_RXD	I	MFP3	UART0 数据接收脚.
				I2C0_SDA	I/O	MFP4	I2C0 数据输入/输出引脚.
				QSPI0_CLK	I/O	MFP5	Quad SPI0 串行时钟引脚.

32 脚	48 脚	64 脚	128 脚	引脚名	类型	MFP值	描述
				XT1_OUT	O	MFP10	外部4~24 MHz (高速) 晶振输出引脚.
				BPWM1_CH1	I/O	MFP11	BPWM1 通道1 输出/捕获输入.
			39	VSS	P	MFP0	数字电路地.
			40	VDD	P	MFP0	I/O 端口电源和LDO 电源用于内部 PLL 和数字电路
			41	PE.8	I/O	MFP0	GPIO.
				EBI_ADR10	O	MFP2	EBI 地址总线位10.
				I2S0_BCLK	O	MFP4	I2S0 位 时钟输出脚.
				SPI2_CLK	I/O	MFP5	SPI2 串行时钟引脚.
				USCI1_CTL1	I/O	MFP6	USCI1 控制1 脚..
				UART2_TXD	O	MFP7	UART2 数据发送脚.
				EPWM0_CH0	I/O	MFP10	EPWM0 通道0 输出/捕获输入脚.
				EPWM0_BRAKE0	I	MFP11	EPWM0 刹车0输入脚.
				ECAP0_IC0	I	MFP12	增强型捕获输入单元0输入脚0
				TRACE_CLK	O	MFP14	ETM 追踪 时钟输出脚
			42	PE.9	I/O	MFP0	GPIO.
				EBI_ADR11	O	MFP2	EBI 地址总线位11.
				I2S0_MCLK	O	MFP4	I2S0 主机时钟输出脚.
				SPI2_MISO	I/O	MFP5	SPI2 MISO (主机输入, 从机输出) 脚.
				USCI1_CTL0	I/O	MFP6	USCI1 控制0 脚..
				UART2_RXD	I	MFP7	UART2 数据接收脚.
				EPWM0_CH1	I/O	MFP10	EPWM0 通道1 输出/捕获输入脚.
				EPWM0_BRAKE1	I	MFP11	EPWM0 刹车1输入脚.
				ECAP0_IC1	I	MFP12	增强型捕获输入单元0输入脚1
				TRACE_DATA0	O	MFP14	ETM 追踪数据 0 输出脚
			43	PE.10	I/O	MFP0	GPIO.
				EBI_ADR12	O	MFP2	EBI 地址总线位12.
				I2S0_DI	I	MFP4	I2S0 数据输入脚.
				SPI2_MOSI	I/O	MFP5	SPI2 MOSI (主机输出, 从机输入) 脚.
				USCI1_DAT0	I/O	MFP6	USCI1 数据0脚.

32 脚	48 脚	64 脚	128 脚	引脚名	类型	MFP值	描述
				UART3_TXD	O	MFP7	UART3 数据发送脚.
				EPWM0_CH2	I/O	MFP10	EPWM0 通道2 输出/捕获输入脚.
				EPWM1_BRAKE0	I	MFP11	EPWM1 刹车0输入脚.
				ECAP0_IC2	I	MFP12	增强型捕获输入单元0输入脚2
				TRACE_DATA1	O	MFP14	ETM 追踪数据 1 输出脚
			44	PE.11	I/O	MFP0	GPIO.
				EBI_ADR13	O	MFP2	EBI 地址总线位13.
				I2S0_DO	O	MFP4	I2S0 数据输出脚
				SPI2_SS	I/O	MFP5	SPI2 从机选择脚.
				USCI1_DAT1	I/O	MFP6	USCI1 数据1脚.
				UART3_RXD	I	MFP7	UART3 数据接收脚.
				UART1_nCTS	I	MFP8	UART1 清除发送输入脚.
				EPWM0_CH3	I/O	MFP10	EPWM0 通道3 输出/捕获输入脚.
				EPWM1_BRAKE1	I	MFP11	EPWM1 刹车1输入脚.
				ECAP1_IC2	I	MFP13	增强型捕获输入单元1输入脚2
			45	TRACE_DATA2	O	MFP14	ETM 追踪数据 2 输出脚
				PE.12	I/O	MFP0	GPIO.
				EBI_ADR14	O	MFP2	EBI 地址总线位14.
				I2S0_LRCK	O	MFP4	I2S0 左右声道选择时钟输出脚.
				SPI2_I2SMCLK	I/O	MFP5	SPI2 I2S 主机时钟输出脚
				USCI1_CLK	I/O	MFP6	USCI1 时钟脚.
				UART1_nRTS	O	MFP8	UART1 请求发送输出脚.
				EPWM0_CH4	I/O	MFP10	EPWM0 通道4 输出/捕获输入脚.
				ECAP1_IC1	I	MFP13	增强型捕获输入单元1输入脚1
			46	TRACE_DATA3	O	MFP14	ETM 追踪数据 3 输出脚
				PE.13	I/O	MFP0	GPIO.
				EBI_ADR15	O	MFP2	EBI 地址总线位15.
				I2C0_SCL	I/O	MFP4	I2C0 时钟脚.
				UART4_nRTS	O	MFP5	UART4 请求发送输出脚.

32脚	48脚	64脚	128脚	引脚名	类型	MFP值	描述
				UART1_TXD	O	MFP8	UART1 数据发送脚.
				EPWM0_CH5	I/O	MFP10	EPWM0 通道5 输出/捕获输入脚.
				EPWM1_CH0	I/O	MFP11	EPWM1 通道0 输出/捕获输入脚.
				BPWM1_CH5	I/O	MFP12	BPWM1 通道5 输出/捕获输入.
				ECAP1_IC0	I	MFP13	增强型捕获输入单元1输入脚0
			47	PC.8	I/O	MFP0	GPIO.
				EBI_ADR16	O	MFP2	EBI 地址总线位16.
				I2C0_SDA	I/O	MFP4	I2C0 数据输入/输出引脚.
				UART4_nCTS	I	MFP5	UART4 清除发送输入脚.
				UART1_RXD	I	MFP8	UART1 数据接收脚.
				EPWM1_CH1	I/O	MFP11	EPWM1 通道1 输出/捕获输入脚.
				BPWM1_CH4	I/O	MFP12	BPWM1 通道4 输出/捕获输入.
			18 48	PC.7	I/O	MFP0	GPIO.
				EBI_AD9	I/O	MFP2	EBI 地址/数据总线位9.
				SPI1_MISO	I/O	MFP4	SPI1 MISO (主机输入, 从机输出) 脚.
				UART4_TXD	O	MFP5	UART4 数据发送脚.
				SC2_PWR	O	MFP6	智能卡 2 电源脚.
				UART0_nCTS	I	MFP7	UART0 清除发送输入脚.
				I2C1_SMBAL	O	MFP8	I2C1 SMBus SMBALTER 脚
				EPWM1_CH2	I/O	MFP11	EPWM1 通道2 输出/捕获输入脚.
				BPWM1_CH0	I/O	MFP12	BPWM1 通道0 输出/捕获输入.
				TM0	I/O	MFP14	定时器0事件计数器输入/反转输出脚.
			19 49	INT3	I	MFP15	外部中断3输入脚.
				PC.6	I/O	MFP0	GPIO.
				EBI_AD8	I/O	MFP2	EBI 地址/数据总线位8.
				SPI1_MOSI	I/O	MFP4	SPI1 MOSI (主机输出, 从机输入) 脚.
				UART4_RXD	I	MFP5	UART4 数据接收脚.
				SC2_RST	O	MFP6	智能卡 2 复位脚.
				UART0_nRTS	O	MFP7	UART0 请求发送输出脚.

32 脚	48 脚	64 脚	128 脚	引脚名	类型	MFP值	描述
				I2C1_SMBSUS	O	MFP8	I2C1 SMBus SMBSUS 脚(PMBus 控制脚)
				EPWM1_CH3	I/O	MFP11	EPWM1 通道3 输出/捕获输入脚.
				BPWM1_CH1	I/O	MFP12	BPWM1 通道1 输出/捕获输入.
				TM1	I/O	MFP14	定时器1事件计数器输入/反转输出脚.
				INT2	I	MFP15	外部中断2输入脚.
				PA.7	I/O	MFP0	GPIO.
				EBI_AD7	I/O	MFP2	EBI 地址/数据总线位7.
				SPI1_CLK	I/O	MFP4	SPI1 串行时钟引脚.
				SC2_DAT	I/O	MFP6	智能卡2 数据脚.
				UART0_TXD	O	MFP7	UART0 数据发送脚.
				I2C1_SCL	I/O	MFP8	I2C1 时钟脚.
				EPWM1_CH4	I/O	MFP11	EPWM1 通道4 输出/捕获输入脚.
				BPWM1_CH2	I/O	MFP12	BPWM1 通道2 输出/捕获输入.
				ACMP0_WLAT	I	MFP13	模拟比较器0 窗口锁定输入脚
				TM2	I/O	MFP14	定时器2事件计数器输入/反转输出脚.
				INT1	I	MFP15	外部中断1输入脚.
				PA.6	I/O	MFP0	GPIO.
				EBI_AD6	I/O	MFP2	EBI 地址/数据总线位6.
				SPI1_SS	I/O	MFP4	SPI1 从机选择脚.
				SD1_nCD	I	MFP5	SD卡接口1 卡侦测输入脚
				SC2_CLK	O	MFP6	智能卡 2 时钟脚.
				UART0_RXD	I	MFP7	UART0 数据接收脚.
				I2C1_SDA	I/O	MFP8	I2C1 数据输入/输出引脚.
				EPWM1_CH5	I/O	MFP11	EPWM1 通道5 输出/捕获输入脚.
				BPWM1_CH3	I/O	MFP12	BPWM1 通道3 输出/捕获输入.
				ACMP1_WLAT	I	MFP13	模拟比较器1 窗口锁定输入脚
				TM3	I/O	MFP14	定时器3事件计数器输入/反转输出脚.
				INT0	I	MFP15	外部中断0输入脚.
		22	52	VSS	P	MFP0	数字电路地.

32 脚	48 脚	64 脚	128 脚	引脚名	类型	MFP值	描述
		23	53	VDD	P	MFP0	I/O 端口电源和LDO 电源用于内部 PLL 和数字电路
		24	54	LDO_CAP	A	MFP0	LDO 输出脚.
	17	25	55	PA.5	I/O	MFP0	GPIO.
				SPI_M2	I/O	MFP2	SPI_M2 四线模式数据2 .
				QSPI0_MISO1	I/O	MFP3	Quad SPI0 MISO1 (主机输入, 从机输出) 脚..
				SPI1_I2SMCLK	I/O	MFP4	SPI1 I2S 主机时钟输出脚
				SD1_CMD	I/O	MFP5	SD卡接口1 命令/应答脚
				SC2_nCD	I	MFP6	智能卡 2 卡侦测脚.
				UART0_nCTS	I	MFP7	UART0 清除发送输入脚.
				UART5_TXD	O	MFP8	UART5 数据发送脚.
				I2C0_SCL	I/O	MFP9	I2C0 时钟脚.
				CAN0_TXD	O	MFP10	CAN0 总线发送输出.
				BPWM0_CH5	I/O	MFP12	BPWM0 通道5 输出/捕获输入.
				EPWM0_CH0	I/O	MFP13	EPWM0 通道0 输出/捕获输入脚.
				QEIO_INDEX	I	MFP14	正交编码器0索引输入
	18	26	56	PA.4	I/O	MFP0	GPIO.
				SPI_M3	I/O	MFP2	SPI_M3 四线模式数据3 .
				QSPI0_MOSI1	I/O	MFP3	Quad SPI0 MOSI1 (主机输出, 从机输入) 脚.
				SPI0_I2SMCLK	I/O	MFP4	SPI0 I2S 主机时钟输出脚
				SD1_CLK	O	MFP5	SD卡接口1 时钟输出脚
				SC0_nCD	I	MFP6	智能卡 0 卡侦测脚.
				UART0_nRTS	O	MFP7	UART0 请求发送输出脚.
				UART5_RXD	I	MFP8	UART5 数据接收脚.
				I2C0_SDA	I/O	MFP9	I2C0 数据输入/输出引脚.
				CAN0_RXD	I	MFP10	CAN0 总线接收输入
				BPWM0_CH4	I/O	MFP12	BPWM0 通道4 输出/捕获输入.
				EPWM0_CH1	I/O	MFP13	EPWM0 通道1 输出/捕获输入脚.
				QEIO_A	I	MFP14	正交编码器0 计数信号 A 相输入
11	19	27	57	PA.3	I/O	MFP0	GPIO.

32 脚	48 脚	64 脚	128 脚	引脚名	类型	MFP值	描述
				SPIM_SS	I/O	MFP2	SPIM 从机选择脚.
				QSPI0_SS	I/O	MFP3	Quad SPI0 从机选择脚.
				SPI0_SS	I/O	MFP4	SPI0 从机选择脚.
				SD1_DAT3	I/O	MFP5	SD卡接口1 数据位3.
				SC0_PWR	O	MFP6	智能卡0 电源脚.
				UART4_TXD	O	MFP7	UART4 数据发送脚.
				UART1_TXD	O	MFP8	UART1 数据发送脚.
				I2C1_SCL	I/O	MFP9	I2C1 时钟脚.
				BPWM0_CH3	I/O	MFP12	BPWM0 通道3 输出/捕获输入.
				EPWM0_CH2	I/O	MFP13	EPWM0 通道2 输出/捕获输入脚.
				QEIO_B	I	MFP14	正交编码0 计数信号B 相输入
			58	PA.2	I/O	MFP0	GPIO.
				SPIM_CLK	I/O	MFP2	SPIM 串行时钟引脚.
				QSPI0_CLK	I/O	MFP3	Quad SPI0 串行时钟引脚.
				SPI0_CLK	I/O	MFP4	SPI0 串行时钟引脚.
				SD1_DAT2	I/O	MFP5	SD卡接口1 数据位2.
				SC0_RST	O	MFP6	智能卡0 复位脚.
				UART4_RXD	I	MFP7	UART4 数据接收脚.
				UART1_RXD	I	MFP8	UART1 数据接收脚.
				I2C1_SDA	I/O	MFP9	I2C1 数据输入/输出引脚.
				BPWM0_CH2	I/O	MFP12	BPWM0 通道2 输出/捕获输入.
			59	EPWM0_CH3	I/O	MFP13	EPWM0 通道3 输出/捕获输入脚.
				PA.1	I/O	MFP0	GPIO.
				SPIM_MISO	I/O	MFP2	SPIM MISO (主机输入, 从机输出) 脚..
				QSPI0_MISO0	I/O	MFP3	Quad SPI0 MISO0 (主机输入, 从机输出) 脚..
				SPI0_MISO	I/O	MFP4	SPI0 MISO (主机输入, 从机输出) 脚.
				SD1_DAT1	I/O	MFP5	SD卡接口1 数据位1.
				SC0_DAT	I/O	MFP6	智能卡0 数据脚.
				UART0_TXD	O	MFP7	UART0 数据发送脚.

32 脚	48 脚	64 脚	128 脚	引脚名	类型	MFP值	描述
				UART1_nCTS	I	MFP8	UART1 清除发送输入脚.
				I2C2_SCL	I/O	MFP9	I2C2 时钟脚.
				BPWM0_CH1	I/O	MFP12	BPWM0 通道1 输出/捕获输入.
				EPWM0_CH4	I/O	MFP13	EPWM0 通道4 输出/捕获输入脚.
				DAC1_ST	I	MFP15	DAC1 外部触发输入.
14	22	30	60	PA.0	I/O	MFP0	GPIO.
				SPIM_MOSI	I/O	MFP2	SPIM MOSI (主机输出, 从机输入) 脚.
				QSPI0_MOSIO	I/O	MFP3	Quad SPI0 MOSIO (主机输出, 从机输入) 脚.
				SPI0_MOSI	I/O	MFP4	SPI0 MOSI (主机输出, 从机输入) 脚.
				SD1_DAT0	I/O	MFP5	SD卡接口1 数据位0.
				SC0_CLK	O	MFP6	智能卡 0 时钟脚.
				UART0_RXD	I	MFP7	UART0 数据接收脚.
				UART1_nRTS	O	MFP8	UART1 请求发送输出脚.
				I2C2_SDA	I/O	MFP9	I2C2 数据输入/输出引脚.
				BPWM0_CH0	I/O	MFP12	BPWM0 通道0 输出/捕获输入.
				EPWM0_CH5	I/O	MFP13	EPWM0 通道5 输出/捕获输入脚.
				DAC0_ST	I	MFP15	DAC0 外部触发输入.
15	23	31	61	VDDIO	P	MFP0	PE.1, PE.8~PE.13的电源.
			62	PE.14	I/O	MFP0	GPIO.
				EBI_AD8	I/O	MFP2	EBI 地址/数据总线位8.
				UART2_TXD	O	MFP3	UART2 数据发送脚.
				CAN0_TXD	O	MFP4	CAN0 总线发送输出.
				SD1_nCD	I	MFP5	SD卡接口1 卡侦测输入脚
			63	PE.15	I/O	MFP0	GPIO.
				EBI_AD9	I/O	MFP2	EBI 地址/数据总线位9.
				UART2_RXD	I	MFP3	UART2 数据接收脚.
				CAN0_RXD	I	MFP4	CAN0 总线接收输入
16	24	32	64	nRESET	I	MFP0	外部复位输入：低电平有效，带内部上拉.
17	25	33	65	PF.0	I/O	MFP0	GPIO.

32 脚	48 脚	64 脚	128 脚	引脚名	类型	MFP值	描述
				UART1_TXD	O	MFP2	UART1 数据发送脚.
				I2C1_SCL	I/O	MFP3	I2C1 时钟脚.
				BPWM1_CH0	I/O	MFP12	BPWM1 通道0 输出/捕获输入.
				ICE_DAT	O	MFP14	串行调试器数据输入.
			66	PF.1	I/O	MFP0	GPIO.
				UART1_RXD	I	MFP2	UART1 数据接收脚.
				I2C1_SDA	I/O	MFP3	I2C1 数据输入/输出引脚.
				BPWM1_CH1	I/O	MFP12	BPWM1 通道1 输出/捕获输入.
				ICE_CLK	I	MFP14	串行调试器时钟脚.
			67	PD.9	I/O	MFP0	GPIO.
				EBI_AD7	I/O	MFP2	EBI 地址/数据总线位7.
				I2C2_SCL	I/O	MFP3	I2C2 时钟脚.
				UART2_nCTS	I	MFP4	UART2 清除发送输入脚.
			68	PD.8	I/O	MFP0	GPIO.
				EBI_AD6	I/O	MFP2	EBI 地址/数据总线位6.
				I2C2_SDA	I/O	MFP3	I2C2 数据输入/输出引脚.
				UART2_nRTS	O	MFP4	UART2 请求发送输出脚.
			69	PC.5	I/O	MFP0	GPIO.
				EBI_AD5	I/O	MFP2	EBI 地址/数据总线位5.
				SPIM_D2	I/O	MFP3	SPIM 四线模式数据2 .
				QSPI0_MISO1	I/O	MFP4	Quad SPI0 MISO1 (主机输入, 从机输出) 脚..
				UART2_TXD	O	MFP8	UART2 数据发送脚.
				I2C1_SCL	I/O	MFP9	I2C1 时钟脚.
				CAN0_TXD	O	MFP10	CAN0 总线发送输出.
				UART4_TXD	O	MFP11	UART4 数据发送脚.
				EPWM1_CH0	I/O	MFP12	EPWM1 通道0 输出/捕获输入脚.
			70	PC.4	I/O	MFP0	GPIO.
				EBI_AD4	I/O	MFP2	EBI 地址/数据总线位4.
				SPIM_D3	I/O	MFP3	SPIM 四线模式数据3 .

32 脚	48 脚	64 脚	128 脚	引脚名	类型	MFP值	描述
				QSPI0_MOSI1	I/O	MFP4	Quad SPI0 MOSI1 (主机输出, 从机输入) 脚.
				SC1_nCD	I	MFP5	智能卡 1 卡侦测脚.
				I2S0_BCLK	O	MFP6	I2S0 位时钟输出脚.
				SPI1_I2SMCLK	I/O	MFP7	SPI1 I2S 主机时钟输出脚
				UART2_RXD	I	MFP8	UART2 数据接收脚.
				I2C1_SDA	I/O	MFP9	I2C1 数据输入/输出引脚.
				CAN0_RXD	I	MFP10	CAN0 总线接收输入
				UART4_RXD	I	MFP11	UART4 数据接收脚.
				EPWM1_CH1	I/O	MFP12	EPWM1 通道1 输出/捕获输入脚.
				PC.3	I/O	MFP0	GPIO.
			71	EBI_AD3	I/O	MFP2	EBI 地址/数据总线位3.
				SPIM_SS	I/O	MFP3	SPIM 从机选择脚.
				QSPI0_SS	I/O	MFP4	Quad SPI0 从机选择脚.
				SC1_PWR	O	MFP5	智能卡 1 电源脚.
				I2S0_MCLK	O	MFP6	I2S0 主机时钟输出脚.
				SPI1_MISO	I/O	MFP7	SPI1 MISO (主机输入, 从机输出) 脚.
				UART2_nRTS	O	MFP8	UART2 请求发送输出脚.
				I2C0_SMBAL	O	MFP9	I2C0 SMBus SMBALTER 脚
				CAN1_TXD	O	MFP10	CAN1 总线发送输出.
				UART3_TXD	O	MFP11	UART3 数据发送脚.
				EPWM1_CH2	I/O	MFP12	EPWM1 通道2 输出/捕获输入脚.
			72	PC.2	I/O	MFP0	GPIO.
				EBI_AD2	I/O	MFP2	EBI 地址/数据总线位2.
				SPIM_CLK	I/O	MFP3	SPIM 串行时钟引脚.
				QSPI0_CLK	I/O	MFP4	Quad SPI0 串行时钟引脚.
				SC1_RST	O	MFP5	智能卡 1 复位脚.
				I2S0_DI	I	MFP6	I2S0 数据输入脚.
				SPI1_MOSI	I/O	MFP7	SPI1 MOSI (主机输出, 从机输入) 脚.
				UART2_nCTS	I	MFP8	UART2 清除发送输入脚.

32 脚	48 脚	64 脚	128 脚	引脚名	类型	MFP值	描述
				I2C0_SMBSUS	O	MFP9	I2C0 SMBus SMBSUS 脚(PMBus 控制脚)
				CAN1_RXD	I	MFP10	CAN1 总线接收输入
				UART3_RXD	I	MFP11	UART3 数据接收脚.
				EPWM1_CH3	I/O	MFP12	EPWM1 通道3 输出/捕获输入脚.
19	31	39	73	PC.1	I/O	MFP0	GPIO.
				EBI_AD1	I/O	MFP2	EBI 地址/数据总线位1.
				SPIM_MISO	I/O	MFP3	SPIM MISO (主机输入, 从机输出) 脚..
				QSPI0_MISO0	I/O	MFP4	Quad SPI0 MISO0 (主机输入, 从机输出) 脚..
				SC1_DAT	I/O	MFP5	智能卡1 数据脚.
				I2S0_DO	O	MFP6	I2S0 数据输出脚
				SPI1_CLK	I/O	MFP7	SPI1 串行时钟引脚.
				UART2_TXD	O	MFP8	UART2 数据发送脚.
				I2C0_SCL	I/O	MFP9	I2C0 时钟脚.
				EPWM1_CH4	I/O	MFP12	EPWM1 通道4 输出/捕获输入脚.
20	32	40	74	ACMP0_O	O	MFP14	模拟比较器0 输出脚.
				PC.0	I/O	MFP0	GPIO.
				EBI_AD0	I/O	MFP2	EBI 地址/数据总线位0.
				SPIM_MOSI	I/O	MFP3	SPIM MOSI (主机输出, 从机输入) 脚.
				QSPI0_MOSI0	I/O	MFP4	Quad SPI0 MOSI0 (主机输出, 从机输入) 脚.
				SC1_CLK	O	MFP5	智能卡 1 时钟脚.
				I2S0_LRCK	O	MFP6	I2S0 左右声道选择时钟输出脚.
				SPI1_SS	I/O	MFP7	SPI1 从机选择脚.
				UART2_RXD	I	MFP8	UART2 数据接收脚.
				I2C0_SDA	I/O	MFP9	I2C0 数据输入/输出引脚.
				EPWM1_CH5	I/O	MFP12	EPWM1 通道5 输出/捕获输入脚.
				ACMP1_O	O	MFP14	模拟比较器1 输出脚.
				VSS	P	MFP0	数字电路地.
				VDD	P	MFP0	I/O 端口电源和LDO 电源用于内部 PLL 和数字电路
				PG.9	I/O	MFP0	GPIO.

32 脚	48 脚	64 脚	128 脚	引脚名	类型	MFP值	描述
				EBI_AD0	I/O	MFP2	EBI 地址/数据总线位0.
				SD1_DAT3	I/O	MFP3	SD卡接口1 数据位3.
				SPIM_D2	I/O	MFP4	SPIM 四线模式数据2 .
				BPWM0_CH5	I/O	MFP12	BPWM0 通道5 输出/捕获输入.
			78	PG.10	I/O	MFP0	GPIO.
				EBI_AD1	I/O	MFP2	EBI 地址/数据总线位1.
				SD1_DAT2	I/O	MFP3	SD卡接口1 数据位2.
				SPIM_D3	I/O	MFP4	SPIM 四线模式数据3 .
				BPWM0_CH4	I/O	MFP12	BPWM0 通道4 输出/捕获输入.
			79	PG.11	I/O	MFP0	GPIO.
				EBI_AD2	I/O	MFP2	EBI 地址/数据总线位2.
				SD1_DAT1	I/O	MFP3	SD卡接口1 数据位1.
				SPIM_SS	I/O	MFP4	SPIM 从机选择脚.
				BPWM0_CH3	I/O	MFP12	BPWM0 通道3 输出/捕获输入.
			80	PG.12	I/O	MFP0	GPIO.
				EBI_AD3	I/O	MFP2	EBI 地址/数据总线位3.
				SD1_DAT0	I/O	MFP3	SD卡接口1 数据位0.
				SPIM_CLK	I/O	MFP4	SPIM 串行时钟引脚.
				BPWM0_CH2	I/O	MFP12	BPWM0 通道2 输出/捕获输入.
			81	PG.13	I/O	MFP0	GPIO.
				EBI_AD4	I/O	MFP2	EBI 地址/数据总线位4.
				SD1_CMD	I/O	MFP3	SD卡接口1 命令/应答脚
				SPIM_MISO	I/O	MFP4	SPIM MISO (主机输入, 从机输出) 脚..
				BPWM0_CH1	I/O	MFP12	BPWM0 通道1 输出/捕获输入.
			82	PG.14	I/O	MFP0	GPIO.
				EBI_AD5	I/O	MFP2	EBI 地址/数据总线位5.
				SD1_CLK	I/O	MFP3	SD卡接口1 时钟输出脚
				SPIM_MOSI	I/O	MFP4	SPIM MOSI (主机输出, 从机输入) 脚.
				BPWM0_CH0	I/O	MFP12	BPWM0 通道0 输出/捕获输入.

32 脚	48 脚	64 脚	128 脚	引脚名	类型	MFP值	描述
			83	PG.15	I/O	MFP0	GPIO.
				SD1_nCD	I	MFP3	SD卡接口1 卡侦测输入脚
				CLKO	O	MFP14	时钟输出
				EADC0_ST	I	MFP15	EADC0 外部触发输入.
			41	PD.3	I/O	MFP0	General purpose digital I/O pin.
				EBI_AD10	I/O	MFP2	EBI address/data bus bit 10.
				USCI0_CTL1	I/O	MFP3	USCI0 control 1 pin.
				SPI0_SS	I/O	MFP4	SPI0 slave select pin.
				UART3_nRTS	O	MFP5	UART3 request to Send output pin.
				USCI1_CTL0	I/O	MFP6	USCI1 control 0 pin.
				SC2_PWR	O	MFP7	Smart Card 2 power pin.
				SC1_nCD	I	MFP8	Smart Card 1 card detect pin.
				UART0_TXD	O	MFP9	UART0 data transmitter output pin.
			42	PD.2	I/O	MFP0	General purpose digital I/O pin.
				EBI_AD11	I/O	MFP2	EBI address/data bus bit 11.
				USCI0_DAT1	I/O	MFP3	USCI0 data 1 pin.
				SPI0_CLK	I/O	MFP4	SPI0 serial clock pin.
				UART3_nCTS	I	MFP5	UART3 clear to Send input pin.
				SC2_RST	O	MFP7	Smart Card 2 reset pin.
				UART0_RXD	I	MFP9	UART0 data receiver input pin.
			43	PD.1	I/O	MFP0	General purpose digital I/O pin.
				EBI_AD12	I/O	MFP2	EBI address/data bus bit 12.
				USCI0_DAT0	I/O	MFP3	USCI0 data 0 pin.
				SPI0_MISO	I/O	MFP4	SPI0 MISO (Master In, Slave Out) pin.
				UART3_TXD	O	MFP5	UART3 data transmitter output pin.
				I2C2_SCL	I/O	MFP6	I2C2 clock pin.
				SC2_DAT	I/O	MFP7	Smart Card 2 data pin.
			44	PD.0	I/O	MFP0	General purpose digital I/O pin.
				EBI_AD13	I/O	MFP2	EBI address/data bus bit 13.
				USCI0_CLK	I/O	MFP3	USCI0 clock pin.
				SPI0_MOSI	I/O	MFP4	SPI0 MOSI (Master Out, Slave In) pin.

32 脚	48 脚	64 脚	128 脚	引脚名	类型	MFP值	描述
				UART3_RXD	I	MFP5	UART3 data receiver input pin.
				I2C2_SDA	I/O	MFP6	I2C2 data input/output pin.
				SC2_CLK	O	MFP7	Smart Card 2 clock pin.
				TM2	I/O	MFP14	Timer2 event counter input/toggle output pin.
			84	PD.13	I/O	MFP0	GPIO.
				EBI_AD10	I/O	MFP2	EBI 地址/数据总线位10.
				SD0_nCD	I	MFP3	SD卡接口0 卡侦测输入脚
				SPI0_I2SMCLK	I/O	MFP4	SPI0 I2S 主机时钟输出脚
				SPI1_I2SMCLK	I/O	MFP5	SPI1 I2S 主机时钟输出脚
				SC2_nCD	I	MFP7	智能卡 2 卡侦测脚.
				PA.12	I/O	MFP0	GPIO.
				I2S0_BCLK	O	MFP2	I2S0 位 时钟输出脚.
				UART4_TXD	O	MFP3	UART4 数据发送脚.
				I2C1_SCL	I/O	MFP4	I2C1 时钟脚.
			85	SPI2_SS	I/O	MFP5	SPI2 从机选择脚.
				CAN0_TXD	O	MFP6	CAN0 总线发送输出.
				SC2_PWR	O	MFP7	智能卡 2 电源脚.
				BPWM1_CH2	I/O	MFP11	BPWM1 通道2 输出/捕获输入.
				QEI1_INDEX	I	MFP12	正交编码器1索引输入
				USB_VBUS	P	MFP14	来自USB主机或HUB的电源
				PA.13	I/O	MFP0	GPIO.
				I2S0_MCLK	O	MFP2	I2S0 主机时钟输出脚.
				UART4_RXD	I	MFP3	UART4 数据接收脚.
				I2C1_SDA	I/O	MFP4	I2C1 数据输入/输出引脚.
			86	SPI2_CLK	I/O	MFP5	SPI2 串行时钟引脚.
				CAN0_RXD	I	MFP6	CAN0 总线接收输入
				SC2_RST	O	MFP7	智能卡 2 复位脚.
				BPWM1_CH3	I/O	MFP11	BPWM1 通道3 输出/捕获输入.
				QEI1_A	I	MFP12	正交编码1 计数信号 A 相输入

32 脚	48 脚	64 脚	128 脚	引脚名	类型	MFP值	描述
				USB_D-	A	MFP14	USB 差分信号D+.
23	35	47	87	PA.14	I/O	MFP0	GPIO.
				I2S0_DI	I	MFP2	I2S0 数据输入脚.
				UART0_TXD	O	MFP3	UART0 数据发送脚.
				SPI2_MISO	I/O	MFP5	SPI2 MISO (主机输入, 从机输出) 脚.
				I2C2_SCL	I/O	MFP6	I2C2 时钟脚.
				SC2_DAT	I/O	MFP7	智能卡2 数据脚.
				BPWM1_CH4	I/O	MFP11	BPWM1 通道4 输出/捕获输入.
				QEI1_B	I	MFP12	正交编码1 计数信号 B 相输入
				USB_D+	A	MFP14	USB 差分信号D-.
24	36	48	88	PA.15	I/O	MFP0	GPIO.
				I2S0_DO	O	MFP2	I2S0 数据输出脚
				UART0_RXD	I	MFP3	UART0 数据接收脚.
				SPI2_MOSI	I/O	MFP5	SPI2 MOSI (主机输出, 从机输入) 脚.
				I2C2_SDA	I/O	MFP6	I2C2 数据输入/输出引脚.
				SC2_CLK	O	MFP7	智能卡 2 时钟脚.
				BPWM1_CH5	I/O	MFP11	BPWM1 通道5 输出/捕获输入.
				EPWM0_SYNC_IN	I	MFP12	EPWM0 计数器同步触发输入脚
				USB_OTG_ID	I	MFP14	USB_ID.
		90	VDD	P	MFP0	I/O	端口电源和LDO 电源用于内部 PLL 和数字电路
		93	VSS	P	MFP0	I/O	数字电路地.
		97	PE.7	I/O	MFP0	I/O	GPIO.
				SD0_CMD	I/O	MFP3	SD卡接口0 命令/应答脚
				SPIIM_D2	I/O	MFP4	SPIIM 四线模式数据2 .
				UART5_TXD	O	MFP8	UART5 数据发送脚.
				CAN1_TXD	O	MFP9	CAN1 总线发送输出.
				QEI1_INDEX	I	MFP11	正交编码器1索引输入
				EPWM0_CH0	I/O	MFP12	EPWM0 通道0 输出/捕获输入脚.
				BPWM0_CH5	I/O	MFP13	BPWM0 通道5 输出/捕获输入.

32 脚	48 脚	64 脚	128 脚	引脚名	类型	MFP值	描述
			98	PE.6	I/O	MFP0	GPIO.
				SD0_CLK	O	MFP3	SD卡接口0 时钟输出脚
				SPIM_D3	I/O	MFP4	SPIM 四线模式数据3 .
				SPI3_I2SMCLK	I/O	MFP5	SPI3 I2S 主机时钟输出脚
				SC0_nCD	I	MFP6	智能卡 0 卡侦测脚.
				USCI0_CTL0	I/O	MFP7	USCI0 控制0 脚..
				UART5_RXD	I	MFP8	UART5 数据接收脚.
				CAN1_RXD	I	MFP9	CAN1 总线接收输入
				QEI1_A	I	MFP11	正交编码1 计数信号 A 相输入
				EPWM0_CH1	I/O	MFP12	EPWM0 通道1 输出/捕获输入脚.
			99	PE.5	I/O	MFP0	GPIO.
				EBI_nRD	O	MFP2	EBI 读使能输出脚.
				SD0_DAT3	I/O	MFP3	SD卡接口0 数据位3.
				SPIM_SS	I/O	MFP4	SPIM 从机选择脚.
				SPI3_SS	I/O	MFP5	SPI3 从机选择脚.
				SC0_PWR	O	MFP6	智能卡 0 电源脚.
				USCI0_CTL1	I/O	MFP7	USCI0 控制1 脚..
				QEI1_B	I	MFP11	正交编码1 计数信号 B 相输入
				EPWM0_CH2	I/O	MFP12	EPWM0 通道2 输出/捕获输入脚.
				BPWM0_CH3	I/O	MFP13	BPWM0 通道3 输出/捕获输入.
			100	PE.4	I/O	MFP0	GPIO.
				EBI_nWR	O	MFP2	EBI 写使能输出脚.
				SD0_DAT2	I/O	MFP3	SD卡接口0 数据位2.
				SPIM_CLK	I/O	MFP4	SPIM 串行时钟引脚.
				SPI3_CLK	I/O	MFP5	SPI3 串行时钟引脚.
				SC0_RST	O	MFP6	智能卡 0 复位脚.
				USCI0_DAT1	I/O	MFP7	USCI0 数据1脚.
				QEIO_INDEX	I	MFP11	正交编码器0索引输入

32 脚	48 脚	64 脚	128 脚	引脚名	类型	MFP值	描述
				EPWM0_CH3	I/O	MFP12	EPWM0 通道3 输出/捕获输入脚.
				BPWM0_CH2	I/O	MFP13	BPWM0 通道2 输出/捕获输入.
			101	PE.3	I/O	MFP0	GPIO.
				EBI_MCLK	O	MFP2	EBI外部时钟输出脚.
				SD0_DAT1	I/O	MFP3	SD卡接口0 数据位1.
				SPI1_MISO	I/O	MFP4	SPI1 MISO (主机输入, 从机输出) 脚..
				SPI3_MISO	I/O	MFP5	SPI3 MISO (主机输入, 从机输出) 脚..
				SC0_DAT	I/O	MFP6	智能卡0 数据脚.
				USCI0_DAT0	I/O	MFP7	USCI0 数据0脚.
				QEIO_A	I	MFP11	正交编码0 计数信号 A 相输入
				EPWM0_CH4	I/O	MFP12	EPWM0 通道4 输出/捕获输入脚.
				BPWM0_CH1	I/O	MFP13	BPWM0 通道1 输出/捕获输入.
			102	PE.2	I/O	MFP0	GPIO.
				EBI_ALE	O	MFP2	EBI 地址锁存使能输出脚.
				SD0_DAT0	I/O	MFP3	SD卡接口0 数据位0.
				SPI1_MOSI	I/O	MFP4	SPI1 MOSI (主机输出, 从机输入) 脚.
				SPI3_MOSI	I/O	MFP5	SPI3 MOSI (主机输出, 从机输入) 脚.
				SC0_CLK	O	MFP6	智能卡 0 时钟脚.
				USCI0_CLK	I/O	MFP7	USCI0 时钟脚.
				QEIO_B	I	MFP11	正交编码0 计数信号 B 相输入
				EPWM0_CH5	I/O	MFP12	EPWM0 通道5 输出/捕获输入脚.
				BPWM0_CH0	I/O	MFP13	BPWM0 通道0 输出/捕获输入.
			103	VSS	P	MFP0	数字电路地.
				VDD	P	MFP0	I/O 端口电源和LDO 电源用于内部 PLL 和数字电路
				PE.1	I/O	MFP0	GPIO.
				EBI_AD10	I/O	MFP2	EBI 地址/数据总线位10.
				QSPI0_MISO0	I/O	MFP3	Quad SPI0 MISO0 (主机输入, 从机输出) 脚..
			105	SC2_DAT	I/O	MFP4	智能卡2 数据脚.
				I2S0_BCLK	O	MFP5	I2S0 位 时钟输出脚.

32 脚	48 脚	64 脚	128 脚	引脚名	类型	MFP值	描述
				SPI1_MISO	I/O	MFP6	SPI1 MISO (主机输入, 从机输出) 脚.
				UART3_TXD	O	MFP7	UART3 数据发送脚.
				I2C1_SCL	I/O	MFP8	I2C1 时钟脚.
				UART4_nCTS	I	MFP9	UART4 清除发送输入脚.
			106	PE.0	I/O	MFP0	GPIO.
				EBI_AD11	I/O	MFP2	EBI 地址/数据总线位11.
				QSPI0_MOSI0	I/O	MFP3	Quad SPI0 MOSI0 (主机输出, 从机输入) 脚.
				SC2_CLK	O	MFP4	智能卡 2 时钟脚.
				I2S0_MCLK	O	MFP5	I2S0 主机时钟输出脚.
				SPI1_MOSI	I/O	MFP6	SPI1 MOSI (主机输出, 从机输入) 脚.
				UART3_RXD	I	MFP7	UART3 数据接收脚.
				I2C1_SDA	I/O	MFP8	I2C1 数据输入/输出引脚.
				UART4_nRTS	O	MFP9	UART4 请求发送输出脚.
			107	PH.8	I/O	MFP0	GPIO.
				EBI_AD12	I/O	MFP2	EBI 地址/数据总线位12.
				QSPI0_CLK	I/O	MFP3	Quad SPI0 串行时钟引脚.
				SC2_PWR	O	MFP4	智能卡 2 电源脚.
				I2S0_DI	I	MFP5	I2S0 数据输入脚.
				SPI1_CLK	I/O	MFP6	SPI1 串行时钟引脚.
				UART3_nRTS	O	MFP7	UART3 请求发送输出脚.
				I2C1_SMBAL	O	MFP8	I2C1 SMBus SMBALTER 脚
				I2C2_SCL	I/O	MFP9	I2C2 时钟脚.
				UART1_TXD	O	MFP10	UART1 数据发送脚.
			108	PH.9	I/O	MFP0	GPIO.
				EBI_AD13	I/O	MFP2	EBI 地址/数据总线位13.
				QSPI0_SS	I/O	MFP3	Quad SPI0 从机选择脚.
				SC2_RST	O	MFP4	智能卡 2 复位脚.
				I2S0_DO	O	MFP5	I2S0 数据输出脚
				SPI1_SS	I/O	MFP6	SPI1 从机选择脚.

32 脚	48 脚	64 脚	128 脚	引脚名	类型	MFP值	描述
				UART3_nCTS	I	MFP7	UART3 清除发送输入脚.
				I2C1_SMBSUS	O	MFP8	I2C1 SMBus SMBSUS 脚(PMBus 控制脚)
				I2C2_SDA	I/O	MFP9	I2C2 数据输入/输出引脚.
				UART1_RXD	I	MFP10	UART1 数据接收脚.
			109	PH.10	I/O	MFP0	GPIO.
				EBI_AD14	I/O	MFP2	EBI 地址/数据总线位14.
				QSPI0_MISO1	I/O	MFP3	Quad SPI0 MISO1 (主机输入, 从机输出) 脚..
				SC2_nCD	I	MFP4	智能卡 2 卡侦测脚.
				I2S0_LRCK	O	MFP5	I2S0 左右声道选择时钟输出脚.
				SPI1_I2SMCLK	I/O	MFP6	SPI1 I2S 主机时钟输出脚
				UART4_TXD	O	MFP7	UART4 数据发送脚.
				UART0_TXD	O	MFP8	UART0 数据发送脚.
			110	PH.11	I/O	MFP0	GPIO.
				EBI_AD15	I/O	MFP2	EBI 地址/数据总线位15.
				QSPI0_MOSI1	I/O	MFP3	Quad SPI0 MOSI1 (主机输出, 从机输入) 脚.
				UART4_RXD	I	MFP7	UART4 数据接收脚.
				UART0_RXD	I	MFP8	UART0 数据接收脚.
				EPWM0_CH5	I/O	MFP11	EPWM0 通道5 输出/捕获输入脚.
			111	PD.14	I/O	MFP0	GPIO.
				EBI_nCS0	O	MFP2	EBI 片选 0 输出脚.
				SPI3_I2SMCLK	I/O	MFP3	SPI3 I2S 主机时钟输出脚
				SC1_nCD	I	MFP4	智能卡 1 卡侦测脚.
				EPWM0_CH4	I/O	MFP11	EPWM0 通道4 输出/捕获输入脚.
25	37	49	112	VSS	P	MFP0	数字电路地.
26	38	50	113	LDO_CAP	A	MFP0	LDO 输出脚.
27	39	51	114	VDD	P	MFP0	I/O 端口电源和LDO 电源用于内部 PLL 和数字电路
			115	PC.14	I/O	MFP0	GPIO.
				EBI_AD11	I/O	MFP2	EBI 地址/数据总线位11.
				SC1_nCD	I	MFP3	智能卡 1 卡侦测脚.

32 脚	48 脚	64 脚	128 脚	引脚名	类型	MFP值	描述
				SPI0_I2SMCLK	I/O	MFP4	SPI0 I2S 主机时钟输出脚
				USCI0_CTL0	I/O	MFP5	USCI0 控制0 脚..
				QSPI0_CLK	I/O	MFP6	Quad SPI0 串行时钟引脚.
				EPWM0_SYNC_IN	I	MFP11	EPWM0 计数器同步触发输入脚
				TM1	I/O	MFP13	定时器1事件计数器输入/反转输出脚.
				USB_VBUS_ST	I	MFP14	USB 外部 VBUS 管理器状态脚.
28	41	53	116	PB.15	I/O	MFP0	GPIO.
				EADC0_CH15	A	MFP1	EADC0 通道15模拟输入.
				EBI_AD12	I/O	MFP2	EBI 地址/数据总线位12.
				SC1_PWR	O	MFP3	智能卡 1 电源脚.
				SPI0_SS	I/O	MFP4	SPI0 从机选择脚.
				USCI0_CTL1	I/O	MFP5	USCI0 控制1 脚..
				UART0_nCTS	I	MFP6	UART0 清除发送输入脚.
				UART3_TXD	O	MFP7	UART3 数据发送脚.
				I2C2_SMBAL	O	MFP8	I2C2 SMBus SMBALTER 脚
				EPWM1_CH0	I/O	MFP11	EPWM1 通道0 输出/捕获输入脚.
				TM0_EXT	I/O	MFP13	定时器0事件计数器输入/反转输出脚.
				USB_VBUS_EN	O	MFP14	USB 外部 VBUS 管理器使能脚.
29	41	54	117	PB.14	I/O	MFP0	GPIO.
				EADC0_CH14	A	MFP1	EADC0 通道14模拟输入.
				EBI_AD13	I/O	MFP2	EBI 地址/数据总线位13.
				SC1_RST	O	MFP3	智能卡 1 复位脚.
				SPI0_CLK	I/O	MFP4	SPI0 串行时钟引脚.
				USCI0_DAT1	I/O	MFP5	USCI0 数据1脚.
				UART0_nRTS	O	MFP6	UART0 请求发送输出脚.
				UART3_RXD	I	MFP7	UART3 数据接收脚.
				I2C2_SMBSUS	O	MFP8	I2C2 SMBus SMBSUS 脚(PMBus 控制脚)
				EPWM1_CH1	I/O	MFP11	EPWM1 通道1 输出/捕获输入脚.
				TM1_EXT	I/O	MFP13	定时器1事件计数器输入/反转输出脚.

32 脚	48 脚	64 脚	128 脚	引脚名	类型	MFP值	描述
				CLKO	O	MFP14	时钟输出
30	43	55	118	PB.13	I/O	MFP0	GPIO.
				EADC0_CH13	A	MFP1	EADC0 通道13模拟输入.
				DAC1_OUT	A	MFP1	DAC1 通道模拟输出
				ACMP0_P3	A	MFP1	模拟比较器0 正输入 3 脚.
				ACMP1_P3	A	MFP1	模拟比较器1 正输入 3 脚.
				EBI_AD14	I/O	MFP2	EBI 地址/数据总线位14.
				SC1_DAT	I/O	MFP3	智能卡1 数据脚.
				SPI0_MISO	I/O	MFP4	SPI0 MISO (主机输入, 从机输出) 脚.
				USCI0_DAT0	I/O	MFP5	USCI0 数据0脚.
				UART0_TXD	O	MFP6	UART0 数据发送脚.
				UART3_nRTS	O	MFP7	UART3 请求发送输出脚.
				I2C2_SCL	I/O	MFP8	I2C2 时钟脚.
				EPWM1_CH2	I/O	MFP11	EPWM1 通道2 输出/捕获输入脚.
				TM2_EXT	I/O	MFP13	定时器2事件计数器输入/反转输出脚.
31	44	56	119	PB.12	I/O	MFP0	GPIO.
				EADC0_CH12	A	MFP1	EADC0 通道12模拟输入.
				DAC0_OUT	A	MFP1	DAC0 通道模拟输出
				ACMP0_P2	A	MFP1	模拟比较器0 正输入 2 脚.
				ACMP1_P2	A	MFP1	模拟比较器1 正输入 2 脚.
				EBI_AD15	I/O	MFP2	EBI 地址/数据总线位15.
				SC1_CLK	O	MFP3	智能卡 1 时钟脚.
				SPI0_MOSI	I/O	MFP4	SPI0 MOSI (主机输出, 从机输入) 脚.
				USCI0_CLK	I/O	MFP5	USCI0 时钟脚.
				UART0_RXD	I	MFP6	UART0 数据接收脚.
				UART3_nCTS	I	MFP7	UART3 清除发送输入脚.
				I2C2_SDA	I/O	MFP8	I2C2 数据输入/输出引脚.
				SD0_nCD	I	MFP9	SD卡接口0 卡侦测输入脚
				EPWM1_CH3	I/O	MFP11	EPWM1 通道3 输出/捕获输入脚.

32 脚	48 脚	64 脚	128 脚	引脚名	类型	MFP值	描述
				TM3_EXT	I/O	MFP13	定时器3事件计数器输入/反转输出脚.
32	45	57	120	AVDD	P	MFP0	内部模拟电路电源 .
		58	121	VREF	A	MFP0	ADC 参考电压输入. 注意: 该脚需要接一个1uF 电容.
	46	59	122	AVSS	P	MFP0	模拟电路地
		60	123	PB.11	I/O	MFP0	GPIO.
				EADC0_CH11	A	MFP1	EADC0 通道11模拟输入.
				EBI_ADR16	O	MFP2	EBI 地址总线位16.
				UART0_nCTS	I	MFP5	UART0 清除发送输入脚.
				UART4_TXD	O	MFP6	UART4 数据发送脚.
				I2C1_SCL	I/O	MFP7	I2C1 时钟脚.
				CAN0_TXD	O	MFP8	CAN0 总线发送输出.
				SPI0_I2SMCLK	I/O	MFP9	SPI0 I2S 主机时钟输出脚
				BPWM1_CH0	I/O	MFP10	BPWM1 通道0 输出/捕获输入.
				SPI3_CLK	I/O	MFP11	SPI3 串行时钟引脚.
		61	124	PB.10	I/O	MFP0	GPIO.
				EADC0_CH10	A	MFP1	EADC0 通道10模拟输入.
				EBI_ADR17	O	MFP2	EBI 地址总线位17.
				USCI1_CTL0	I/O	MFP4	USCI1 控制0 脚..
				UART0_nRTS	O	MFP5	UART0 请求发送输出脚.
				UART4_RXD	I	MFP6	UART4 数据接收脚.
				I2C1_SDA	I/O	MFP7	I2C1 数据输入/输出引脚.
				CAN0_RXD	I	MFP8	CAN0 总线接收输入
				BPWM1_CH1	I/O	MFP10	BPWM1 通道1 输出/捕获输入.
				SPI3_SS	I/O	MFP11	SPI3 从机选择脚.
		62	125	PB.9	I/O	MFP0	GPIO.
				EADC0_CH9	A	MFP1	EADC0 通道9模拟输入.
				EBI_ADR18	O	MFP2	EBI 地址总线位18.
				USCI1_CTL1	I/O	MFP4	USCI1 控制1 脚..

32 脚	48 脚	64 脚	128 脚	引脚名	类型	MFP值	描述	
				UART0_TXD	O	MFP5	UART0 数据发送脚.	
				UART1_nCTS	I	MFP6	UART1 清除发送输入脚.	
				I2C1_SMBAL	O	MFP7	I2C1 SMBus SMBALTER 脚	
				BPWM1_CH2	I/O	MFP10	BPWM1 通道2 输出/捕获输入.	
				SPI3_MISO	I/O	MFP11	SPI3 MISO (主机输入, 从机输出) 脚..	
				INT7	I	MFP13	外部中断7输入脚.	
		63	126	PB.8	I/O	MFP0	GPIO.	
				EADC0_CH8	A	MFP1	EADC0 通道8模拟输入.	
				EBI_ADR19	O	MFP2	EBI 地址总线位19.	
				USCI1_CLK	I/O	MFP4	USCI1 时钟脚.	
				UART0_RXD	I	MFP5	UART0 数据接收脚.	
				UART1_nRTS	O	MFP6	UART1 请求发送输出脚.	
				I2C1_SMBSUS	O	MFP7	I2C1 SMBus SMBSUS 脚(PMBus 控制脚)	
				BPWM1_CH3	I/O	MFP10	BPWM1 通道3 输出/捕获输入.	
				SPI3_MOSI	I/O	MFP11	SPI3 MOSI (主机输出, 从机输入) 脚.	
				INT6	I	MFP13	外部中断6输入脚.	
		47	64	127	PB.7	I/O	MFP0	GPIO.
					EADC0_CH7	A	MFP1	EADC0 通道7模拟输入.
					EBI_nWRL	O	MFP2	EBI 低字节写使能输出脚.
					USCI1_DAT0	I/O	MFP4	USCI1 数据0脚.
					CAN1_TXD	O	MFP5	CAN1 总线发送输出.
					UART1_TXD	O	MFP6	UART1 数据发送脚.
					SD1_CMD	I/O	MFP7	SD卡接口1 命令/应答脚
					EBI_nCS0	O	MFP8	EBI 片选 0 输出脚.
					BPWM1_CH4	I/O	MFP10	BPWM1 通道4 输出/捕获输入.
					EPWM1_BRAKE0	I	MFP11	EPWM1 刹车0输入脚.
					EPWM1_CH4	I/O	MFP12	EPWM1 通道4 输出/捕获输入脚.
					INT5	I	MFP13	外部中断5输入脚.
					USB_VBUS_ST	I	MFP14	USB 外部 VBUS 管理器状态脚.

32脚	48脚	64脚	128脚	引脚名	类型	MFP值	描述
				ACMP0_O	O	MFP15	模拟比较器0 输出脚.
		1	128	PB.6	I/O	MFP0	GPIO.
				EADC0_CH6	A	MFP1	EADC0 通道6模拟输入.
				EBI_nWRH	O	MFP2	EBI 高字节写使能输出脚
				USCI1_DAT1	I/O	MFP4	USCI1 数据1脚.
				CAN1_RXD	I	MFP5	CAN1 总线接收输入
				UART1_RXD	I	MFP6	UART1 数据接收脚.
				SD1_CLK	O	MFP7	SD卡接口1 时钟输出脚
				EBI_nCS1	O	MFP8	EBI 片选 1 输出脚.
				BPWM1_CH5	I/O	MFP10	BPWM1 通道5 输出/捕获输入.
				EPWM1_BRAKE1	I	MFP11	EPWM1 刹车1输入脚.
				EPWM1_CH5	I/O	MFP12	EPWM1 通道5 输出/捕获输入脚.
				INT4	I	MFP13	外部中断4输入脚.
				USB_VBUS_EN	O	MFP14	USB 外部 VBUS 管理器使能脚.
				ACMP1_O	O	MFP15	模拟比较器1 输出脚.

#### 4.2.3 M483 系列引脚描述

64脚	128脚	引脚名称	类型	MFP	描述
2	1	PB.5	I/O	MFP0	GPIO.
		EADC0_CH5	A	MFP1	EADC0 通道5模拟输入.
		ACMP1_N	A	MFP1	模拟比较器1 负输入脚.
		EBI_ADR0	O	MFP2	EBI 地址总线位0.
		SD0_DAT3	I/O	MFP3	SD卡接口0 数据位3.
		SPI1_MISO	I/O	MFP5	SPI1 MISO (主机输入, 从机输出) 脚.

64 脚	128 脚	引脚名称	类型	MFP	描述
		I2C0_SCL	I/O	MFP6	I2C0 时钟脚.
		UART5_TXD	O	MFP7	UART5 数据发送脚.
		USCI1_CTL0	I/O	MFP8	USCI1 控制0 脚..
		SC0_CLK	O	MFP9	智能卡0 时钟脚.
		I2S0_BCLK	O	MFP10	I2S0 位 时钟输出脚.
		EPWM0_CH0	I/O	MFP11	EPWM0 通道0 输出/捕获输入脚.
		TM0	I/O	MFP14	定时器0事件计数器输入/反转输出脚.
		INT0	I	MFP15	外部中断0输入脚.
	3 2	PB.4	I/O	MFP0	GPIO.
		EADC0_CH4	A	MFP1	EADC0 通道4模拟输入.
		ACMP1_P1	A	MFP1	模拟比较器1 正输入 1 脚.
		EBI_ADR1	O	MFP2	EBI 地址总线位1.
		SD0_DAT2	I/O	MFP3	SD卡接口0 数据位2.
		SPI1_MOSI	I/O	MFP5	SPI1 MOSI (主机输出, 从机输入) 脚.
		I2C0_SDA	I/O	MFP6	I2C0 数据输入/输出引脚.
		UART5_RXD	I	MFP7	UART5 数据接收脚.
		USCI1_CTL1	I/O	MFP8	USCI1 控制1 脚..
		SC0_DAT	I/O	MFP9	智能卡0 数据脚.
		I2S0_MCLK	O	MFP10	I2S0 主机时钟输出脚.
		EPWM0_CH1	I/O	MFP11	EPWM0 通道1 输出/捕获输入脚.
		TM1	I/O	MFP14	定时器1事件计数器输入/反转输出脚.
		INT1	I	MFP15	外部中断1输入脚.
	4 3	PB.3	I/O	MFP0	GPIO.
		EADC0_CH3	A	MFP1	EADC0 通道3模拟输入.
		ACMP0_N	A	MFP1	模拟比较器0 负输入脚.
		EBI_ADR2	O	MFP2	EBI 地址总线位2.
		SD0_DAT1	I/O	MFP3	SD卡接口0 数据位1.
		SPI1_CLK	I/O	MFP5	SPI1 串行时钟引脚.
		UART1_RXD	O	MFP6	UART1 数据发送脚.

64脚	128脚	引脚名称	类型	MFP	描述
		UART5_nRTS	O	MFP7	UART5 请求发送输出脚.
		USCI1_DAT1	I/O	MFP8	USCI1 数据1脚.
		SC0_RST	O	MFP9	智能卡0复位脚.
		I2S0_DI	I	MFP10	I2S0 数据输入脚.
		EPWM0_CH2	I/O	MFP11	EPWM0 通道2输出/捕获输入脚.
		TM2	I/O	MFP14	定时器2事件计数器输入/反转输出脚.
		INT2	I	MFP15	外部中断2输入脚.
	5 4	PB.2	I/O	MFP0	GPIO.
		EADC0_CH2	A	MFP1	EADC0 通道2模拟输入.
		ACMP0_P1	A	MFP1	模拟比较器0正输入1脚.
		OPA0_O	A	MFP1	运放0输出脚.
		EBI_ADR3	O	MFP2	EBI 地址总线位3.
		SD0_DAT0	I/O	MFP3	SD卡接口0数据位0.
		SPI1_SS	I/O	MFP5	SPI1 从机选择脚.
		UART1_RXD	I	MFP6	UART1 数据接收脚.
		UART5_nCTS	I	MFP7	UART5 清除发送输入脚.
		USCI1_DAT0	I/O	MFP8	USCI1 数据0脚.
		SC0_PWR	O	MFP9	智能卡0电源脚.
		I2S0_DO	O	MFP10	I2S0 数据输出脚
		EPWM0_CH3	I/O	MFP11	EPWM0 通道3输出/捕获输入脚.
		TM3	I/O	MFP14	定时器3事件计数器输入/反转输出脚.
		INT3	I	MFP15	外部中断3输入脚.
	5	PC.12	I/O	MFP0	GPIO.
		EBI_ADR4	O	MFP2	EBI 地址总线位4.
		UART0_TXD	O	MFP3	UART0 数据发送脚.
		I2C0_SCL	I/O	MFP4	I2C0 时钟脚.
		SPI3_MISO	I/O	MFP6	SPI3 MISO (主机输入, 从机输出) 脚..
		SC0_nCD	I	MFP9	智能卡0卡侦测脚.
		ECAP1_IC2	I	MFP11	增强型捕获输入单元1输入脚2

64脚	128脚	引脚名称	类型	MFP	描述
		EPWM1_CH0	I/O	MFP12	EPWM1 通道0 输出/捕获输入脚.
		ACMP0_O	O	MFP14	模拟比较器0 输出脚.
	6	PC.11	I/O	MFP0	GPIO.
		EBI_ADR5	O	MFP2	EBI 地址总线位5.
		UART0_RXD	I	MFP3	UART0 数据接收脚.
		I2C0_SDA	I/O	MFP4	I2C0 数据输入/输出引脚.
		SPI3_MOSI	I/O	MFP6	SPI3 MOSI (主机输出, 从机输入) 脚.
		ECAP1_IC1	I	MFP11	增强型捕获输入单元1输入脚1
		EPWM1_CH1	I/O	MFP12	EPWM1 通道1 输出/捕获输入脚.
		ACMP1_O	O	MFP14	模拟比较器1 输出脚.
	7	PC.10	I/O	MFP0	GPIO.
		EBI_ADR6	O	MFP2	EBI 地址总线位 6.
		SPI3_CLK	I/O	MFP6	SPI3 串行时钟引脚.
		UART3_TXD	O	MFP7	UART3 数据发送脚.
		CAN1_TXD	O	MFP9	CAN1 总线发送输出.
		ECAP1_IC0	I	MFP11	增强型捕获输入单元0输入脚0
		EPWM1_CH2	I/O	MFP12	EPWM1 通道2 输出/捕获输入脚.
	8	PC.9	I/O	MFP0	GPIO.
		EBI_ADR7	O	MFP2	EBI 地址总线位 7.
		SPI3_SS	I/O	MFP6	SPI3 从机选择脚.
		UART3_RXD	I	MFP7	UART3 数据接收脚.
		CAN1_RXD	I	MFP9	CAN1 总线接收输入
		EPWM1_CH3	I/O	MFP12	EPWM1 通道3 输出/捕获输入脚.
	9	PB.1	I/O	MFP0	GPIO.
		EADC0_CH1	A	MFP1	EADC0 通道1模拟输入.
		OPA0_N	A	MFP1	运放 0 负输入脚.
		EBI_ADR8	O	MFP2	EBI 地址总线位 8.
		SD0_CLK	O	MFP3	SD卡接口0 时钟输出脚
		SPI1_I2SMCLK	I/O	MFP5	SPI1 I2S 主机时钟输出脚

64 脚	128 脚	引脚名称	类型	MFP	描述
		SPI3_I2SMCLK	I/O	MFP6	SPI3 I2S 主机时钟输出脚
		UART2_TXD	O	MFP7	UART2 数据发送脚.
		USCI1_CLK	I/O	MFP8	USCI1 时钟脚.
		I2C1_SCL	I/O	MFP9	I2C1 时钟脚.
		I2S0_LRCK	O	MFP10	I2S0 左右声道选择时钟输出脚.
		EPWM0_CH4	I/O	MFP11	EPWM0 通道4 输出/捕获输入脚.
		EPWM1_CH4	I/O	MFP12	EPWM1 通道4 输出/捕获输入脚.
		EPWM0_BRAKE0	I	MFP13	EPWM0 刹车0输入脚.
	7 10	PB.0	I/O	MFP0	GPIO.
		EADC0_CH0	A	MFP1	EADC0 通道0模拟输入.
		OPA0_P	A	MFP1	运放0 正输入脚.
		EBI_ADR9	O	MFP2	EBI 地址总线位 9.
		SD0_CMD	I/O	MFP3	SD卡接口0 命令/应答脚
		UART2_RXD	I	MFP7	UART2 数据接收脚.
		SPI0_I2SMCLK	I/O	MFP8	SPI0 I2S 主机时钟输出脚
		I2C1_SDA	I/O	MFP9	I2C1 数据输入/输出引脚.
		EPWM0_CH5	I/O	MFP11	EPWM0 通道5 输出/捕获输入脚.
		EPWM1_CH5	I/O	MFP12	EPWM1 通道5 输出/捕获输入脚.
	11	V <sub>ss</sub>	P	MFP0	数字电路地.
		V <sub>DD</sub>	P	MFP0	I/O 端口电源和LDO 电源用于内部 PLL 和数字电路
	8 13	PA.11	I/O	MFP0	GPIO.
		ACMP0_P0	A	MFP1	模拟比较器0 正输入 0 脚.
		EBI_nRD	O	MFP2	EBI 读使能输出脚.
		SC2_PWR	O	MFP3	智能卡 2 电源脚.
		SPI2_SS	I/O	MFP4	SPI2 从机选择脚.
		SD1_DAT3	I/O	MFP5	SD卡接口1 数据位3.
		USCI0_CLK	I/O	MFP6	USCI0 时钟脚.
		I2C2_SCL	I/O	MFP7	I2C2 时钟脚.

64 脚	128 脚	引脚名称	类型	MFP	描述
		BPWM0_CH0	I/O	MFP9	BPWM0 通道0 输出/捕获输入.
		EPWM0_SYNC_OUT	O	MFP10	EPWM0 计数器同步触发输出脚.
		TM0_EXT	I/O	MFP13	定时器0事件计数器输入/反转输出脚.
		DAC1_ST	I	MFP14	DAC1 外部触发输入.
9	14	PA.10	I/O	MFP0	GPIO.
		ACMP1_P0	A	MFP1	模拟比较器1 正输入 0 脚.
		OPA1_O	A	MFP1	运放 1 输出脚.
		EBI_nWR	O	MFP2	EBI 写使能输出脚.
		SC2_RST	O	MFP3	智能卡 2 复位脚.
		SPI2_CLK	I/O	MFP4	SPI2 串行时钟引脚.
		SD1_DAT2	I/O	MFP5	SD卡接口1 数据位2.
		USCI0_DAT0	I/O	MFP6	USCI0 数据0脚.
		I2C2_SDA	I/O	MFP7	I2C2 数据输入/输出引脚.
		BPWM0_CH1	I/O	MFP9	BPWM0 通道1 输出/捕获输入.
		QE1_INDEX	I	MFP10	正交编码器1索引输入
		ECAP0_IC0	I	MFP11	增强型捕获输入单元0输入脚0
		TM1_EXT	I/O	MFP13	定时器1事件计数器输入/反转输出脚.
		DAC0_ST	I	MFP14	DAC0 外部触发输入.
10	15	PA.9	I/O	MFP0	GPIO.
		OPA1_N	A	MFP1	运放1 负输入端.
		EBI_MCLK	O	MFP2	EBI外部时钟输出脚.
		SC2_DAT	I/O	MFP3	智能卡2 数据脚.
		SPI2_MISO	I/O	MFP4	SPI2 MISO (主机输入, 从机输出) 脚.
		SD1_DAT1	I/O	MFP5	SD卡接口1 数据位1.
		USCI0_DAT1	I/O	MFP6	USCI0 数据1脚.
		UART1_TXD	O	MFP7	UART1 数据发送脚.
		BPWM0_CH2	I/O	MFP9	BPWM0 通道2 输出/捕获输入.
		QE1_A	I	MFP10	正交编码1 计数信号 A 相输入
		ECAP0_IC1	I	MFP11	增强型捕获输入单元0输入脚1

64 脚	128 脚	引脚名称	类型	MFP	描述
		TM2_EXT	I/O	MFP13	定时器2事件计数器输入/反转输出脚.
11	16	PA.8	I/O	MFP0	GPIO.
		OPA1_P	A	MFP1	运放1 正输入脚.
		EBI_ALE	O	MFP2	EBI 地址锁存使能输出脚.
		SC2_CLK	O	MFP3	智能卡 2 时钟脚.
		SPI2_MOSI	I/O	MFP4	SPI2 MOSI (主机输出, 从机输入) 脚.
		SD1_DAT0	I/O	MFP5	SD卡接口1 数据位0.
		USCI0_CTL1	I/O	MFP6	USCI0 控制1 脚..
		UART1_RXD	I	MFP7	UART1 数据接收脚.
		BPWM0_CH3	I/O	MFP9	BPWM0 通道3 输出/捕获输入.
		QEI1_B	I	MFP10	正交编码1 计数信号 B 相输入
		ECAP0_IC2	I	MFP11	增强型捕获输入单元0输入脚2
		TM3_EXT	I/O	MFP13	定时器3事件计数器输入/反转输出脚.
		INT4	I	MFP15	外部中断4输入脚.
17	17	PC.13	I/O	MFP0	GPIO.
		EBI_ADR10	O	MFP2	EBI 地址总线位10.
		SC2_nCD	I	MFP3	智能卡 2 卡侦测脚.
		SPI2_I2SMCLK	I/O	MFP4	SPI2 I2S 主机时钟输出脚
		CAN1_TXD	O	MFP5	CAN1 总线发送输出.
		USCI0_CTL0	I/O	MFP6	USCI0 控制0 脚..
		UART2_TXD	O	MFP7	UART2 数据发送脚.
		BPWM0_CH4	I/O	MFP9	BPWM0 通道4 输出/捕获输入.
		CLKO	O	MFP13	时钟输出
		EADC0_ST	I	MFP14	EADC0 外部触发输入.
18	18	PD.12	I/O	MFP0	GPIO.
		OPA2_O	A	MFP1	运放 2 输出脚.
		EBI_nCS0	O	MFP2	EBI 片选 0 输出脚.
		CAN1_RXD	I	MFP5	CAN1 总线接收输入
		UART2_RXD	I	MFP7	UART2 数据接收脚.

64 脚	128 脚	引脚名称	类型	MFP	描述
		BPWM0_CH5	I/O	MFP9	BPWM0 通道5 输出/捕获输入.
		QEIO_INDEX	I	MFP10	正交编码器0索引输入
		CLKO	O	MFP13	时钟输出
		EADC0_ST	I	MFP14	EADC0 外部触发输入.
		INT5	I	MFP15	外部中断5输入脚.
	19	PD.11	I/O	MFP0	GPIO.
		OPA2_N	A	MFP1	运放 2 负输入脚.
		EBI_nCS1	O	MFP2	EBI 片选 1 输出脚.
		UART1_TXD	O	MFP3	UART1 数据发送脚.
		CAN0_TXD	O	MFP4	CAN0 总线发送输出.
		QEIO_A	I	MFP10	正交编码0 计数信号 A 相输入
		INT6	I	MFP15	外部中断6输入脚.
	20	PD.10	I/O	MFP0	GPIO.
		OPA2_P	A	MFP1	运放 2 正输入脚.
		EBI_nCS2	O	MFP2	EBI 片选 2 输出脚.
		UART1_RXD	I	MFP3	UART1 数据接收脚.
		CAN0_RXD	I	MFP4	CAN0 总线接收输入
		QEIO_B	I	MFP10	正交编码0 计数信号 B 相输入
		INT7	I	MFP15	外部中断7输入脚.
	21	PG.2	I/O	MFP0	GPIO.
		EBI_ADR11	O	MFP2	EBI 地址总线位11.
		SPI2_SS	I/O	MFP3	SPI2 从机选择脚.
		I2C0_SMBAL	O	MFP4	I2C0 SMBus SMBALTER 脚
		I2C1_SCL	I/O	MFP5	I2C1 时钟脚.
		TM0	I/O	MFP13	定时器0事件计数器输入/反转输出脚.
	22	PG.3	I/O	MFP0	GPIO.
		EBI_ADR12	O	MFP2	EBI 地址总线位12.
		SPI2_CLK	I/O	MFP3	SPI2 串行时钟引脚.
		I2C0_SMBSUS	O	MFP4	I2C0 SMBus SMBSUS 脚(PMBus 控制脚)

64 脚	128 脚	引脚名称	类型	MFP	描述
		I2C1_SDA	I/O	MFP5	I2C1 数据输入/输出引脚.
		TM1	I/O	MFP13	定时器1事件计数器输入/反转输出脚.
	23	PG.4	I/O	MFP0	GPIO.
		EBI_ADR13	O	MFP2	EBI 地址总线位13.
		SPI2_MISO	I/O	MFP3	SPI2 MISO (主机输入, 从机输出) 脚.
		TM2	I/O	MFP13	定时器2事件计数器输入/反转输出脚.
	24	PF.11	I/O	MFP0	GPIO.
		EBI_ADR14	O	MFP2	EBI 地址总线位14.
		SPI2_MOSI	I/O	MFP3	SPI2 MOSI (主机输出, 从机输入) 脚.
		TAMPER5	I/O	MFP10	TAMPER 侦测循环脚 5.
		TM3	I/O	MFP13	定时器3事件计数器输入/反转输出脚.
	25	PF.10	I/O	MFP0	GPIO.
		EBI_ADR15	O	MFP2	EBI 地址总线位15.
		SC0_nCD	I	MFP3	智能卡 0 卡侦测脚.
		I2S0_BCLK	O	MFP4	I2S0 位时钟输出脚.
		SPI0_I2SMCLK	I/O	MFP5	SPI0 I2S 主机时钟输出脚
		TAMPER4	I/O	MFP10	TAMPER 侦测循环脚 4.
	26	PF.9	I/O	MFP0	GPIO.
		EBI_ADR16	O	MFP2	EBI 地址总线位16.
		SC0_PWR	O	MFP3	智能卡 0 电源脚.
		I2S0_MCLK	O	MFP4	I2S0 主机时钟输出脚.
		SPI0_SS	I/O	MFP5	SPI0 从机选择脚.
		TAMPER3	I/O	MFP10	TAMPER 侦测循环脚 3.
	27	PF.8	I/O	MFP0	GPIO.
		EBI_ADR17	O	MFP2	EBI 地址总线位17.
		SC0_RST	O	MFP3	智能卡 0 复位脚.
		I2S0_DI	I	MFP4	I2S0 数据输入脚.
		SPI0_CLK	I/O	MFP5	SPI0 串行时钟引脚.
		TAMPER2	I/O	MFP10	TAMPER 侦测循环脚 2.

64 脚	128 脚	引脚名称	类型	MFP	描述
28	28	PF.7	I/O	MFP0	GPIO.
		EBI_ADR18	O	MFP2	EBI 地址总线位18.
		SC0_DAT	I/O	MFP3	智能卡0 数据脚.
		I2S0_DO	O	MFP4	I2S0 数据输出脚
		SPI0_MISO	I/O	MFP5	SPI0 MISO (主机输入, 从机输出) 脚.
		UART4_TXD	O	MFP6	UART4 数据发送脚.
		TAMPER1	I/O	MFP10	TAMPER 侦测循环脚 1.
12	29	PF.6	I/O	MFP0	GPIO.
		EBI_ADR19	O	MFP2	EBI 地址总线位19.
		SC0_CLK	O	MFP3	智能卡 0 时钟脚.
		I2S0_LRCK	O	MFP4	I2S0 左右声道选择时钟输出脚.
		SPI0_MOSI	I/O	MFP5	SPI0 MOSI (主机输出, 从机输入) 脚.
		UART4_RXD	I	MFP6	UART4 数据接收脚.
		EBI_nCS0	O	MFP7	EBI 片选 0 输出脚.
		TAMPER0	I/O	MFP10	TAMPER 侦测循环脚 0.
13	30	VDD	P	MFP0	I/O 端口电源和LDO 电源用于内部 PLL 和数字电路
14	31	PF.5	I/O	MFP0	GPIO.
		UART2_RXD	I	MFP2	UART2 数据接收脚.
		UART2_nCTS	I	MFP4	UART2 清除发送输入脚.
		BPWM0_CH4	I/O	MFP8	BPWM0 通道4 输出/捕获输入.
		EPWM0_SYNC_OUT	O	MFP9	EPWM0 计数器同步触发输出脚.
		X32_IN	I	MFP10	外部32.768 kHz 晶振输入脚.
		EADC0_ST	I	MFP11	EADC0 外部触发输入.
15	32	PF.4	I/O	MFP0	GPIO.
		UART2_TXD	O	MFP2	UART2 数据发送脚.
		UART2_nRTS	O	MFP4	UART2 请求发送输出脚.
		BPWM0_CH5	I/O	MFP8	BPWM0 通道5 输出/捕获输入.
		X32_OUT	O	MFP10	外部32.768 kHz 晶振输出脚.
	33	PH.4	I/O	MFP0	GPIO.

64 脚	128 脚	引脚名称	类型	MFP	描述
		EBI_ADR3	O	MFP2	EBI 地址总线位3.
		SPI1_MISO	I/O	MFP3	SPI1 MISO (主机输入, 从机输出) 脚.
	34	PH.5	I/O	MFP0	GPIO.
		EBI_ADR2	O	MFP2	EBI 地址总线位2.
		SPI1_MOSI	I/O	MFP3	SPI1 MOSI (主机输出, 从机输入) 脚.
	35	PH.6	I/O	MFP0	GPIO.
		EBI_ADR1	O	MFP2	EBI 地址总线位1.
		SPI1_CLK	I/O	MFP3	SPI1 串行时钟引脚.
	36	PH.7	I/O	MFP0	GPIO.
		EBI_ADR0	O	MFP2	EBI 地址总线位0.
		SPI1_SS	I/O	MFP3	SPI1 从机选择脚.
	37	PF.3	I/O	MFP0	GPIO.
		EBI_nCS0	O	MFP2	EBI 片选 0 输出脚.
		UART0_TXD	O	MFP3	UART0 数据发送脚.
		I2C0_SCL	I/O	MFP4	I2C0 时钟脚.
		XT1_IN	I	MFP10	外部4~24 MHz (高速) 晶振输入脚.
		BPWM1_CH0	I/O	MFP11	BPWM1 通道0 输出/捕获输入.
	38	PF.2	I/O	MFP0	GPIO.
		EBI_nCS1	O	MFP2	EBI 片选 1 输出脚.
		UART0_RXD	I	MFP3	UART0 数据接收脚.
		I2C0_SDA	I/O	MFP4	I2C0 数据输入/输出引脚.
		QSPI0_CLK	I/O	MFP5	Quad SPI0 串行时钟引脚.
		XT1_OUT	O	MFP10	外部4~24 MHz (高速) 晶振输出引脚.
		BPWM1_CH1	I/O	MFP11	BPWM1 通道1 输出/捕获输入.
	39	V <sub>ss</sub>	P	MFP0	数字电路地.
	40	V <sub>DD</sub>	P	MFP0	I/O 端口电源和LDO 电源用于内部 PLL 和数字电路
	41	PE.8	I/O	MFP0	GPIO.
		EBI_ADR10	O	MFP2	EBI 地址总线位10.
		I2S0_BCLK	O	MFP4	I2S0 位 时钟输出脚.

64 脚	128 脚	引脚名称	类型	MFP	描述
		SPI2_CLK	I/O	MFP5	SPI2 串行时钟引脚.
		USCI1_CTL1	I/O	MFP6	USCI1 控制1 脚..
		UART2_TXD	O	MFP7	UART2 数据发送脚.
		EPWM0_CH0	I/O	MFP10	EPWM0 通道0 输出/捕获输入脚.
		EPWM0_BRAKE0	I	MFP11	EPWM0 刹车0输入脚.
		ECAP0_IC0	I	MFP12	增强型捕获输入单元0输入脚0
		TRACE_CLK	O	MFP14	ETM 追踪时钟输出脚
		PE.9	I/O	MFP0	GPIO.
		EBI_ADR11	O	MFP2	EBI 地址总线位11.
		I2S0_MCLK	O	MFP4	I2S0 主机时钟输出脚.
		SPI2_MISO	I/O	MFP5	SPI2 MISO (主机输入, 从机输出) 脚.
		USCI1_CTL0	I/O	MFP6	USCI1 控制0 脚..
		UART2_RXD	I	MFP7	UART2 数据接收脚.
		EPWM0_CH1	I/O	MFP10	EPWM0 通道1 输出/捕获输入脚.
		EPWM0_BRAKE1	I	MFP11	EPWM0 刹车1输入脚.
		ECAP0_IC1	I	MFP12	增强型捕获输入单元0输入脚1
		TRACE_DATA0	O	MFP14	ETM 追踪数据 0 输出脚
		PE.10	I/O	MFP0	GPIO.
		EBI_ADR12	O	MFP2	EBI 地址总线位12.
		I2S0_DI	I	MFP4	I2S0 数据输入脚.
		SPI2_MOSI	I/O	MFP5	SPI2 MOSI (主机输出, 从机输入) 脚.
		USCI1_DAT0	I/O	MFP6	USCI1 数据0脚.
		UART3_TXD	O	MFP7	UART3 数据发送脚.
		EPWM0_CH2	I/O	MFP10	EPWM0 通道2 输出/捕获输入脚.
		EPWM1_BRAKE0	I	MFP11	EPWM1 刹车0输入脚.
		ECAP0_IC2	I	MFP12	增强型捕获输入单元0输入脚2
		TRACE_DATA1	O	MFP14	ETM 追踪数据 1 输出脚
		PE.11	I/O	MFP0	GPIO.
		EBI_ADR13	O	MFP2	EBI 地址总线位13.

64 脚	128 脚	引脚名称	类型	MFP	描述
		I2S0_DO	O	MFP4	I2S0 数据输出脚
		SPI2_SS	I/O	MFP5	SPI2 从机选择脚.
		USCI1_DAT1	I/O	MFP6	USCI1 数据1脚.
		UART3_RXD	I	MFP7	UART3 数据接收脚.
		UART1_nCTS	I	MFP8	UART1 清除发送输入脚.
		EPWM0_CH3	I/O	MFP10	EPWM0 通道3 输出/捕获输入脚.
		EPWM1_BRAKE1	I	MFP11	EPWM1 刹车1输入脚.
		ECAP1_IC2	I	MFP13	增强型捕获输入单元1输入脚2
		TRACE_DATA2	O	MFP14	ETM 追踪数据 2 输出脚
	45	PE.12	I/O	MFP0	GPIO.
		EBI_ADR14	O	MFP2	EBI 地址总线位14.
		I2S0_LRCK	O	MFP4	I2S0 左右声道选择时钟输出脚.
		SPI2_I2SMCLK	I/O	MFP5	SPI2 I2S 主机时钟输出脚
		USCI1_CLK	I/O	MFP6	USCI1 时钟脚.
		UART1_nRTS	O	MFP8	UART1 请求发送输出脚.
		EPWM0_CH4	I/O	MFP10	EPWM0 通道4 输出/捕获输入脚.
		ECAP1_IC1	I	MFP13	增强型捕获输入单元1输入脚1
		TRACE_DATA3	O	MFP14	ETM 追踪数据 3 输出脚
	46	PE.13	I/O	MFP0	GPIO.
		EBI_ADR15	O	MFP2	EBI 地址总线位15.
		I2C0_SCL	I/O	MFP4	I2C0 时钟脚.
		UART4_nRTS	O	MFP5	UART4 请求发送输出脚.
		UART1_TXD	O	MFP8	UART1 数据发送脚.
		EPWM0_CH5	I/O	MFP10	EPWM0 通道5 输出/捕获输入脚.
		EPWM1_CH0	I/O	MFP11	EPWM1 通道0 输出/捕获输入脚.
		BPWM1_CH5	I/O	MFP12	BPWM1 通道5 输出/捕获输入.
		ECAP1_IC0	I	MFP13	增强型捕获输入单元1输入脚0
	47	PC.8	I/O	MFP0	GPIO.
		EBI_ADR16	O	MFP2	EBI 地址总线位16.

64 脚	128 脚	引脚名称	类型	MFP	描述
		I2C0_SDA	I/O	MFP4	I2C0 数据输入/输出引脚.
		UART4_nCTS	I	MFP5	UART4 清除发送输入脚.
		UART1_RXD	I	MFP8	UART1 数据接收脚.
		EPWM1_CH1	I/O	MFP11	EPWM1 通道1 输出/捕获输入脚.
		BPWM1_CH4	I/O	MFP12	BPWM1 通道4 输出/捕获输入.
		PC.7	I/O	MFP0	GPIO.
		EBI_AD9	I/O	MFP2	EBI 地址/数据总线位9.
		SPI1_MISO	I/O	MFP4	SPI1 MISO (主机输入, 从机输出) 脚.
		UART4_TXD	O	MFP5	UART4 数据发送脚.
		SC2_PWR	O	MFP6	智能卡 2 电源脚.
		UART0_nCTS	I	MFP7	UART0 清除发送输入脚.
		I2C1_SMBAL	O	MFP8	I2C1 SMBus SMBALTER 脚
		EPWM1_CH2	I/O	MFP11	EPWM1 通道2 输出/捕获输入脚.
		BPWM1_CH0	I/O	MFP12	BPWM1 通道0 输出/捕获输入.
		TM0	I/O	MFP14	定时器0事件计数器输入/反转输出脚.
		INT3	I	MFP15	外部中断3输入脚.
		PC.6	I/O	MFP0	GPIO.
		EBI_AD8	I/O	MFP2	EBI 地址/数据总线位8.
		SPI1_MOSI	I/O	MFP4	SPI1 MOSI (主机输出, 从机输入) 脚.
		UART4_RXD	I	MFP5	UART4 数据接收脚.
		SC2_RST	O	MFP6	智能卡 2 复位脚.
		UART0_nRTS	O	MFP7	UART0 请求发送输出脚.
		I2C1_SMBSUS	O	MFP8	I2C1 SMBus SMBSUS 脚(PMBus 控制脚)
		EPWM1_CH3	I/O	MFP11	EPWM1 通道3 输出/捕获输入脚.
		BPWM1_CH1	I/O	MFP12	BPWM1 通道1 输出/捕获输入.
		TM1	I/O	MFP14	定时器1事件计数器输入/反转输出脚.
		INT2	I	MFP15	外部中断2输入脚.
		PA.7	I/O	MFP0	GPIO.
		EBI_AD7	I/O	MFP2	EBI 地址/数据总线位7.

64 脚	128 脚	引脚名称	类型	MFP	描述
		SPI1_CLK	I/O	MFP4	SPI1 串行时钟引脚.
		SC2_DAT	I/O	MFP6	智能卡2 数据脚.
		UART0_TXD	O	MFP7	UART0 数据发送脚.
		I2C1_SCL	I/O	MFP8	I2C1 时钟脚.
		EPWM1_CH4	I/O	MFP11	EPWM1 通道4 输出/捕获输入脚.
		BPWM1_CH2	I/O	MFP12	BPWM1 通道2 输出/捕获输入.
		ACMP0_WLAT	I	MFP13	模拟比较器0 窗口锁定输入脚
		TM2	I/O	MFP14	定时器2事件计数器输入/反转输出脚.
		INT1	I	MFP15	外部中断1输入脚.
	21	PA.6	I/O	MFP0	GPIO.
		EBI_AD6	I/O	MFP2	EBI 地址/数据总线位6.
		SPI1_SS	I/O	MFP4	SPI1 从机选择脚.
		SD1_nCD	I	MFP5	SD卡接口1 卡侦测输入脚
		SC2_CLK	O	MFP6	智能卡 2 时钟脚.
		UART0_RXD	I	MFP7	UART0 数据接收脚.
		I2C1_SDA	I/O	MFP8	I2C1 数据输入/输出引脚.
		EPWM1_CH5	I/O	MFP11	EPWM1 通道5 输出/捕获输入脚.
		BPWM1_CH3	I/O	MFP12	BPWM1 通道3 输出/捕获输入.
		ACMP1_WLAT	I	MFP13	模拟比较器1 窗口锁定输入脚
		TM3	I/O	MFP14	定时器3事件计数器输入/反转输出脚.
		INT0	I	MFP15	外部中断0输入脚.
22	52	Vss	P	MFP0	数字电路地.
23	53	VDD	P	MFP0	I/O 端口电源和LDO 电源用于内部 PLL 和数字电路
24	54	LDO_CAP	A	MFP0	LDO 输出脚.
	25	PA.5	I/O	MFP0	GPIO.
		SPIM_D2	I/O	MFP2	SPIM 四线模式数据2 .
		QSPI0_MISO1	I/O	MFP3	Quad SPI0 MISO1 (主机输入, 从机输出) 脚..
		SPI1_I2SMCLK	I/O	MFP4	SPI1 I2S 主机时钟输出脚
		SD1_CMD	I/O	MFP5	SD卡接口1 命令/应答脚

<b>64 脚</b>	<b>128 脚</b>	<b>引脚名称</b>	<b>类型</b>	<b>MFP</b>	<b>描述</b>
		SC2_nCD	I	MFP6	智能卡 2 卡侦测脚.
		UART0_nCTS	I	MFP7	UART0 清除发送输入脚.
		UART5_TXD	O	MFP8	UART5 数据发送脚.
		I2C0_SCL	I/O	MFP9	I2C0 时钟脚.
		CAN0_TXD	O	MFP10	CAN0 总线发送输出.
		BPWM0_CH5	I/O	MFP12	BPWM0 通道5 输出/捕获输入.
		EPWM0_CH0	I/O	MFP13	EPWM0 通道0 输出/捕获输入脚.
		QEIO_INDEX	I	MFP14	正交编码器0索引输入
26	56	PA.4	I/O	MFP0	GPIO.
		SPIM_D3	I/O	MFP2	SPIM 四线模式数据3 .
		QSPI0_MOSI1	I/O	MFP3	Quad SPI0 MOSI1 (主机输出, 从机输入) 脚.
		SPI0_I2SMCLK	I/O	MFP4	SPI0 I2S 主机时钟输出脚
		SD1_CLK	O	MFP5	SD卡接口1 时钟输出脚
		SC0_nCD	I	MFP6	智能卡 0 卡侦测脚.
		UART0_nRTS	O	MFP7	UART0 请求发送输出脚.
		UART5_RXD	I	MFP8	UART5 数据接收脚.
		I2C0_SDA	I/O	MFP9	I2C0 数据输入/输出引脚.
		CAN0_RXD	I	MFP10	CAN0 总线接收输入
		BPWM0_CH4	I/O	MFP12	BPWM0 通道4 输出/捕获输入.
		EPWM0_CH1	I/O	MFP13	EPWM0 通道1 输出/捕获输入脚.
		QEIO_A	I	MFP14	正交编码0 计数信号 A 相输入
27	57	PA.3	I/O	MFP0	GPIO.
		SPIM_SS	I/O	MFP2	SPIM 从机选择脚.
		QSPI0_SS	I/O	MFP3	Quad SPI0 从机选择脚.
		SPI0_SS	I/O	MFP4	SPI0 从机选择脚.
		SD1_DAT3	I/O	MFP5	SD卡接口1 数据位3.
		SC0_PWR	O	MFP6	智能卡 0 电源脚.
		UART4_TXD	O	MFP7	UART4 数据发送脚.
		UART1_TXD	O	MFP8	UART1 数据发送脚.

<b>64 脚</b>	<b>128 脚</b>	<b>引脚名称</b>	<b>类型</b>	<b>MFP</b>	<b>描述</b>
		I2C1_SCL	I/O	MFP9	I2C1 时钟脚.
		BPWM0_CH3	I/O	MFP12	BPWM0 通道3 输出/捕获输入.
		EPWM0_CH2	I/O	MFP13	EPWM0 通道2 输出/捕获输入脚.
		QEIO_B	I	MFP14	正交编码0 计数信号 B 相输入
	28	PA.2	I/O	MFP0	GPIO.
		SPI_M_CLK	I/O	MFP2	SPI_M 串行时钟引脚.
		QSPI0_CLK	I/O	MFP3	Quad SPI0 串行时钟引脚.
		SPI0_CLK	I/O	MFP4	SPI0 串行时钟引脚.
		SD1_DAT2	I/O	MFP5	SD卡接口1 数据位2.
		SC0_RST	O	MFP6	智能卡0 复位脚.
		UART4_RXD	I	MFP7	UART4 数据接收脚.
		UART1_RXD	I	MFP8	UART1 数据接收脚.
		I2C1_SDA	I/O	MFP9	I2C1 数据输入/输出引脚.
		BPWM0_CH2	I/O	MFP12	BPWM0 通道2 输出/捕获输入.
		EPWM0_CH3	I/O	MFP13	EPWM0 通道3 输出/捕获输入脚.
		PA.1	I/O	MFP0	GPIO.
	29	SPI_M_MISO	I/O	MFP2	SPI_M MISO (主机输入, 从机输出) 脚..
		QSPI0_MISO0	I/O	MFP3	Quad SPI0 MISO0 (主机输入, 从机输出) 脚..
		SPI0_MISO	I/O	MFP4	SPI0 MISO (主机输入, 从机输出) 脚.
		SD1_DAT1	I/O	MFP5	SD卡接口1 数据位1.
		SC0_DAT	I/O	MFP6	智能卡0 数据脚.
		UART0_TXD	O	MFP7	UART0 数据发送脚.
		UART1_nCTS	I	MFP8	UART1 清除发送输入脚.
		I2C2_SCL	I/O	MFP9	I2C2 时钟脚.
		BPWM0_CH1	I/O	MFP12	BPWM0 通道1 输出/捕获输入.
		EPWM0_CH4	I/O	MFP13	EPWM0 通道4 输出/捕获输入脚.
		DAC1_ST	I	MFP15	DAC1 外部触发输入.
		PA.0	I/O	MFP0	GPIO.
	30	SPI_M_MOSI	I/O	MFP2	SPI_M MOSI (主机输出, 从机输入) 脚.

64 脚	128 脚	引脚名称	类型	MFP	描述
		QSPI0_MOSIO	I/O	MFP3	Quad SPI0 MOSIO (主机输出, 从机输入) 脚.
		SPI0_MOSI	I/O	MFP4	SPI0 MOSI (主机输出, 从机输入) 脚.
		SD1_DAT0	I/O	MFP5	SD卡接口1 数据位0.
		SC0_CLK	O	MFP6	智能卡0时钟脚.
		UART0_RXD	I	MFP7	UART0 数据接收脚.
		UART1_nRTS	O	MFP8	UART1 请求发送输出脚.
		I2C2_SDA	I/O	MFP9	I2C2 数据输入/输出引脚.
		BPWM0_CH0	I/O	MFP12	BPWM0 通道0 输出/捕获输入.
		EPWM0_CH5	I/O	MFP13	EPWM0 通道5 输出/捕获输入脚.
		DAC0_ST	I	MFP15	DAC0 外部触发输入.
31	61	VDDIO	P	MFP0	PE.1, PE.8~PE.13的电源.
	62	PE.14	I/O	MFP0	GPIO.
		EBI_AD8	I/O	MFP2	EBI 地址/数据总线位8.
		UART2_TXD	O	MFP3	UART2 数据发送脚.
		CAN0_TXD	O	MFP4	CAN0 总线发送输出.
		SD1_nCD	I	MFP5	SD卡接口1 卡侦测输入脚
	63	PE.15	I/O	MFP0	GPIO.
		EBI_AD9	I/O	MFP2	EBI 地址/数据总线位9.
		UART2_RXD	I	MFP3	UART2 数据接收脚.
		CAN0_RXD	I	MFP4	CAN0 总线接收输入
32	64	nRESET	I	MFP0	外部复位输入: 低电平有效, 带内部上拉.
	65	PF.0	I/O	MFP0	GPIO.
		UART1_TXD	O	MFP2	UART1 数据发送脚.
		I2C1_SCL	I/O	MFP3	I2C1 时钟脚.
		BPWM1_CH0	I/O	MFP12	BPWM1 通道0 输出/捕获输入.
		ICE_DAT	O	MFP14	串行调试器数据脚.
	66	PF.1	I/O	MFP0	GPIO.
		UART1_RXD	I	MFP2	UART1 数据接收脚.
		I2C1_SDA	I/O	MFP3	I2C1 数据输入/输出引脚.

<b>64 脚</b>	<b>128 脚</b>	<b>引脚名称</b>	<b>类型</b>	<b>MFP</b>	<b>描述</b>
		BPWM1_CH1	I/O	MFP12	BPWM1 通道1 输出/捕获输入.
		ICE_CLK	I	MFP14	串行调试器时钟脚.
	67	PD.9	I/O	MFP0	GPIO.
		EBI_AD7	I/O	MFP2	EBI 地址/数据总线位7.
		I2C2_SCL	I/O	MFP3	I2C2 时钟脚.
		UART2_nCTS	I	MFP4	UART2 清除发送输入脚.
	68	PD.8	I/O	MFP0	GPIO.
		EBI_AD6	I/O	MFP2	EBI 地址/数据总线位6.
		I2C2_SDA	I/O	MFP3	I2C2 数据输入/输出引脚.
		UART2_nRTS	O	MFP4	UART2 请求发送输出脚.
	35	PC.5	I/O	MFP0	GPIO.
		EBI_AD5	I/O	MFP2	EBI 地址/数据总线位5.
		SPIM_D2	I/O	MFP3	SPIM 四线模式数据2 .
		QSPI0_MISO1	I/O	MFP4	Quad SPI0 MISO1 (主机输入, 从机输出) 脚..
		UART2_TXD	O	MFP8	UART2 数据发送脚.
		I2C1_SCL	I/O	MFP9	I2C1 时钟脚.
		CAN0_TXD	O	MFP10	CAN0 总线发送输出.
		UART4_TXD	O	MFP11	UART4 数据发送脚.
		EPWM1_CH0	I/O	MFP12	EPWM1 通道0 输出/捕获输入脚.
	36	PC.4	I/O	MFP0	GPIO.
		EBI_AD4	I/O	MFP2	EBI 地址/数据总线位4.
		SPIM_D3	I/O	MFP3	SPIM 四线模式数据3 .
		QSPI0_MOSI1	I/O	MFP4	Quad SPI0 MOSI1 (主机输出, 从机输入) 脚.
		SC1_nCD	I	MFP5	智能卡 1 卡侦测脚.
		I2S0_BCLK	O	MFP6	I2S0 bit 时钟输出脚.
		SPI1_I2SMCLK	I/O	MFP7	SPI1 I2S 主机时钟输出脚
		UART2_RXD	I	MFP8	UART2 数据接收脚.
		I2C1_SDA	I/O	MFP9	I2C1 数据输入/输出引脚.
		CAN0_RXD	I	MFP10	CAN0 总线接收输入

<b>64 脚</b>	<b>128 脚</b>	<b>引脚名称</b>	<b>类型</b>	<b>MFP</b>	<b>描述</b>
		UART4_RXD	I	MFP11	UART4 数据接收脚.
		EPWM1_CH1	I/O	MFP12	EPWM1 通道1 输出/捕获输入脚.
37	71	PC.3	I/O	MFP0	GPIO.
		EBI_AD3	I/O	MFP2	EBI 地址/数据总线位3.
		SPIM_SS	I/O	MFP3	SPIM 从机选择脚.
		QSPI0_SS	I/O	MFP4	Quad SPI0 从机选择脚.
		SC1_PWR	O	MFP5	智能卡 1 电源脚.
		I2S0_MCLK	O	MFP6	I2S0 主机时钟输出脚.
		SPI1_MISO	I/O	MFP7	SPI1 MISO (主机输入, 从机输出) 脚.
		UART2_nRTS	O	MFP8	UART2 请求发送输出脚.
		I2C0_SMBAL	O	MFP9	I2C0 SMBus SMBALTER 脚
		CAN1_TXD	O	MFP10	CAN1 总线发送输出.
		UART3_TXD	O	MFP11	UART3 数据发送脚.
		EPWM1_CH2	I/O	MFP12	EPWM1 通道2 输出/捕获输入脚.
38	72	PC.2	I/O	MFP0	GPIO.
		EBI_AD2	I/O	MFP2	EBI 地址/数据总线位2.
		SPIM_CLK	I/O	MFP3	SPIM 串行时钟引脚.
		QSPI0_CLK	I/O	MFP4	Quad SPI0 串行时钟引脚.
		SC1_RST	O	MFP5	智能卡 1 复位脚.
		I2S0_DI	I	MFP6	I2S0 数据输入脚.
		SPI1_MOSI	I/O	MFP7	SPI1 MOSI (主机输出, 从机输入) 脚.
		UART2_nCTS	I	MFP8	UART2 清除发送输入脚.
		I2C0_SMBSUS	O	MFP9	I2C0 SMBus SMBSUS 脚(PMBus 控制脚)
		CAN1_RXD	I	MFP10	CAN1 总线接收输入
		UART3_RXD	I	MFP11	UART3 数据接收脚.
		EPWM1_CH3	I/O	MFP12	EPWM1 通道3 输出/捕获输入脚.
39	73	PC.1	I/O	MFP0	GPIO.
		EBI_AD1	I/O	MFP2	EBI 地址/数据总线位1.
		SPIM_MISO	I/O	MFP3	SPIM MISO (主机输入, 从机输出) 脚..

64 脚	128 脚	引脚名称	类型	MFP	描述
40	74	QSPI0_MISO0	I/O	MFP4	Quad SPI0 MISO0 (主机输入, 从机输出) 脚..
		SC1_DAT	I/O	MFP5	智能卡1 数据脚.
		I2S0_DO	O	MFP6	I2S0 数据输出脚
		SPI1_CLK	I/O	MFP7	SPI1 串行时钟引脚.
		UART2_TXD	O	MFP8	UART2 数据发送脚.
		I2C0_SCL	I/O	MFP9	I2C0 时钟脚.
		EPWM1_CH4	I/O	MFP12	EPWM1 通道4 输出/捕获输入脚.
		ACMP0_O	O	MFP14	模拟比较器0 输出脚.
		PC.0	I/O	MFP0	GPIO.
		EBI_AD0	I/O	MFP2	EBI 地址/数据总线位0.
		SPIM_MOSI	I/O	MFP3	SPIM MOSI (主机输出, 从机输入) 脚.
		QSPI0_MOSI0	I/O	MFP4	Quad SPI0 MOSI0 (主机输出, 从机输入) 脚.
		SC1_CLK	O	MFP5	智能卡 1 时钟脚.
		I2S0_LRCK	O	MFP6	I2S0 左右声道选择时钟输出脚.
		SPI1_SS	I/O	MFP7	SPI1 从机选择脚.
		UART2_RXD	I	MFP8	UART2 数据接收脚.
		I2C0_SDA	I/O	MFP9	I2C0 数据输入/输出引脚.
		EPWM1_CH5	I/O	MFP12	EPWM1 通道5 输出/捕获输入脚.
		ACMP1_O	O	MFP14	模拟比较器1 输出脚.
		V <sub>ss</sub>	P	MFP0	数字电路地.
		V <sub>DD</sub>	P	MFP0	I/O 端口电源和LDO 电源用于内部 PLL 和数字电路
		PG.9	I/O	MFP0	GPIO.
		EBI_AD0	I/O	MFP2	EBI 地址/数据总线位0.
		SD1_DAT3	I/O	MFP3	SD卡接口1 数据位3.
		SPIM_D2	I/O	MFP4	SPIM 四线模式数据2 .
		BPWM0_CH5	I/O	MFP12	BPWM0 通道5 输出/捕获输入.
		PG.10	I/O	MFP0	GPIO.
		EBI_AD1	I/O	MFP2	EBI 地址/数据总线位1.
		SD1_DAT2	I/O	MFP3	SD卡接口1 数据位2.

64 脚	128 脚	引脚名称	类型	MFP	描述
		SPI_M_D3	I/O	MFP4	SPI_M 四线模式数据3.
		BPWM0_CH4	I/O	MFP12	BPWM0 通道4 输出/捕获输入.
	79	PG.11	I/O	MFP0	GPIO.
		EBI_AD2	I/O	MFP2	EBI 地址/数据总线位2.
		SD1_DAT1	I/O	MFP3	SD卡接口1 数据位1.
		SPI_M_SS	I/O	MFP4	SPI_M 从机选择脚.
		BPWM0_CH3	I/O	MFP12	BPWM0 通道3 输出/捕获输入.
	80	PG.12	I/O	MFP0	GPIO.
		EBI_AD3	I/O	MFP2	EBI 地址/数据总线位3.
		SD1_DAT0	I/O	MFP3	SD卡接口1 数据位0.
		SPI_M_CLK	I/O	MFP4	SPI_M 串行时钟引脚.
		BPWM0_CH2	I/O	MFP12	BPWM0 通道2 输出/捕获输入.
	81	PG.13	I/O	MFP0	GPIO.
		EBI_AD4	I/O	MFP2	EBI 地址/数据总线位4.
		SD1_CMD	I/O	MFP3	SD卡接口1 命令/应答脚
		SPI_M_MISO	I/O	MFP4	SPI_M MISO (主机输入, 从机输出) 脚..
		BPWM0_CH1	I/O	MFP12	BPWM0 通道1 输出/捕获输入.
	82	PG.14	I/O	MFP0	GPIO.
		EBI_AD5	I/O	MFP2	EBI 地址/数据总线位5.
		SD1_CLK	O	MFP3	SD卡接口1 时钟输出脚
		SPI_M_MOSI	I/O	MFP4	SPI_M MOSI (主机输出, 从机输入) 脚.
		BPWM0_CH0	I/O	MFP12	BPWM0 通道0 输出/捕获输入.
	83	PG.15	I/O	MFP0	GPIO.
		SD1_nCD	I	MFP3	SD卡接口1 卡侦测输入脚
		CLKO	O	MFP14	时钟输出
		EADC0_ST	I	MFP15	EADC0 外部触发输入.
	84	PD.13	I/O	MFP0	GPIO.
		EBI_AD10	I/O	MFP2	EBI 地址/数据总线位10.
		SD0_nCD	I	MFP3	SD卡接口0 卡侦测输入脚

64 脚	128 脚	引脚名称	类型	MFP	描述
		SPI0_I2SMCLK	I/O	MFP4	SPI0 I2S 主机时钟输出脚
		SPI1_I2SMCLK	I/O	MFP5	SPI1 I2S 主机时钟输出脚
		SC2_nCD	I	MFP7	智能卡 2 卡侦测脚.
	85	PA.12	I/O	MFP0	GPIO.
		I2S0_BCLK	O	MFP2	I2S0 位 时钟输出脚.
		UART4_TXD	O	MFP3	UART4 数据发送脚.
		I2C1_SCL	I/O	MFP4	I2C1 时钟脚.
		SPI2_SS	I/O	MFP5	SPI2 从机选择脚.
		CAN0_TXD	O	MFP6	CAN0 总线发送输出.
		SC2_PWR	O	MFP7	智能卡 2 电源脚.
		BPWM1_CH2	I/O	MFP11	BPWM1 通道2 输出/捕获输入.
		QEI1_INDEX	I	MFP12	正交编码器1索引输入
		USB_VBUS	P	MFP14	来自USB主机或HUB的电源
	86	PA.13	I/O	MFP0	GPIO.
		I2S0_MCLK	O	MFP2	I2S0 主机时钟输出脚.
		UART4_RXD	I	MFP3	UART4 数据接收脚.
		I2C1_SDA	I/O	MFP4	I2C1 数据输入/输出引脚.
		SPI2_CLK	I/O	MFP5	SPI2 串行时钟引脚.
		CAN0_RXD	I	MFP6	CAN0 总线接收输入
		SC2_RST	O	MFP7	智能卡 2 复位脚.
		BPWM1_CH3	I/O	MFP11	BPWM1 通道3 输出/捕获输入.
		QEI1_A	I	MFP12	正交编码1 计数信号 A 相输入
		USB_D-	A	MFP14	USB 差分信号D+.
	87	PA.14	I/O	MFP0	GPIO.
		I2S0_DI	I	MFP2	I2S0 数据输入脚.
		UART0_TXD	O	MFP3	UART0 数据发送脚.
		SPI2_MISO	I/O	MFP5	SPI2 MISO (主机输入, 从机输出) 脚.
		I2C2_SCL	I/O	MFP6	I2C2 时钟脚.
		SC2_DAT	I/O	MFP7	智能卡2 数据脚.

64 脚	128 脚	引脚名称	类型	MFP	描述
		BPWM1_CH4	I/O	MFP11	BPWM1 通道4 输出/捕获输入.
		QEI1_B	I	MFP12	正交编码1 计数信号 B 相输入
		USB_D+	A	MFP14	USB 差分信号D-.
	88	PA.15	I/O	MFP0	GPIO.
		I2S0_DO	O	MFP2	I2S0 数据输出脚
		UART0_RXD	I	MFP3	UART0 数据接收脚.
		SPI2_MOSI	I/O	MFP5	SPI2 MOSI (主机输出, 从机输入) 脚.
		I2C2_SDA	I/O	MFP6	I2C2 数据输入/输出引脚.
		SC2_CLK	O	MFP7	智能卡 2 时钟脚.
		BPWM1_CH5	I/O	MFP11	BPWM1 通道5 输出/捕获输入.
		EPWM0_SYNC_IN	I	MFP12	EPWM0 计数器同步触发输入脚
		USB_OTG_ID	I	MFP14	USB_ID.
41	89	HSUSB_VRES	A	MFP0	HSUSB 模块参考电阻
42	90	HSUSB_VDD33	P	MFP0	HSUSB VDD33电源
43	91	HSUSB_VBUS	P	MFP0	HSUSB 来自USB主机或HUB的电源
44	92	HSUSB_D-	A	MFP0	HSUSB 差分信号D-.
45	93	HSUSB_VSS	P	MFP0	HSUSB地
46	94	HSUSB_D+	A	MFP0	HSUSB 差分信号D+.
47	95	HSUSB_VDD12_CAP	A	MFP0	HSUSB 内部电源管理器输出1.2V 去耦脚. 注意: 该脚需要接一个1uF 电容.
48	96	HSUSB_ID	I	MFP0	HSUSB ID.
	97	PE.7	I/O	MFP0	GPIO.
		SD0_CMD	I/O	MFP3	SD卡接口0 命令/应答脚
		SPIM_D2	I/O	MFP4	SPIM 四线模式数据2 .
		UART5_TXD	O	MFP8	UART5 数据发送脚.
		CAN1_TXD	O	MFP9	CAN1 总线发送输出.
		QEI1_INDEX	I	MFP11	正交编码器1索引输入
		EPWM0_CH0	I/O	MFP12	EPWM0 通道0 输出/捕获输入脚.
		BPWM0_CH5	I/O	MFP13	BPWM0 通道5 输出/捕获输入.
	98	PE.6	I/O	MFP0	GPIO.

64 脚	128 脚	引脚名称	类型	MFP	描述
		SD0_CLK	O	MFP3	SD卡接口0 时钟输出脚
		SPI_M_D3	I/O	MFP4	SPI_M 四线模式数据3 .
		SPI3_I2SMCLK	I/O	MFP5	SPI3 I2S 主机时钟输出脚
		SC0_nCD	I	MFP6	智能卡 0 卡侦测脚.
		USCI0_CTL0	I/O	MFP7	USCI0 控制0 脚..
		UART5_RXD	I	MFP8	UART5 数据接收脚.
		CAN1_RXD	I	MFP9	CAN1 总线接收输入
		QEI1_A	I	MFP11	正交编码1 计数信号 A 相输入
		EPWM0_CH1	I/O	MFP12	EPWM0 通道1 输出/捕获输入脚.
		BPWM0_CH4	I/O	MFP13	BPWM0 通道4 输出/捕获输入.
	99	PE.5	I/O	MFP0	GPIO.
		EBI_nRD	O	MFP2	EBI 读使能输出脚.
		SD0_DAT3	I/O	MFP3	SD卡接口0 数据位3.
		SPI_M_SS	I/O	MFP4	SPI_M 从机选择脚.
		SPI3_SS	I/O	MFP5	SPI3 从机选择脚.
		SC0_PWR	O	MFP6	智能卡 0 电源脚.
		USCI0_CTL1	I/O	MFP7	USCI0 控制1 脚..
		QEI1_B	I	MFP11	正交编码1 计数信号 B 相输入
		EPWM0_CH2	I/O	MFP12	EPWM0 通道2 输出/捕获输入脚.
		BPWM0_CH3	I/O	MFP13	BPWM0 通道3 输出/捕获输入.
	100	PE.4	I/O	MFP0	GPIO.
		EBI_nWR	O	MFP2	EBI 写使能输出脚.
		SD0_DAT2	I/O	MFP3	SD卡接口0 数据位2.
		SPI_M_CLK	I/O	MFP4	SPI_M 串行时钟引脚.
		SPI3_CLK	I/O	MFP5	SPI3 串行时钟引脚.
		SC0_RST	O	MFP6	智能卡 0 复位脚.
		USCI0_DAT1	I/O	MFP7	USCI0 数据1脚.
		QEIO_INDEX	I	MFP11	正交编码器0索引输入
		EPWM0_CH3	I/O	MFP12	EPWM0 通道3 输出/捕获输入脚.

<b>64 脚</b>	<b>128 脚</b>	引脚名称	类型	MFP	描述
		BPWM0_CH2	I/O	MFP13	BPWM0 通道2 输出/捕获输入.
101	101	PE.3	I/O	MFP0	GPIO.
		EBI_MCLK	O	MFP2	EBI外部时钟输出脚.
		SD0_DAT1	I/O	MFP3	SD卡接口0 数据位1.
		SPI1_MISO	I/O	MFP4	SPI1 MISO (主机输入, 从机输出) 脚..
		SPI3_MISO	I/O	MFP5	SPI3 MISO (主机输入, 从机输出) 脚..
		SC0_DAT	I/O	MFP6	智能卡0 数据脚.
		USCI0_DAT0	I/O	MFP7	USCI0 数据0脚.
		QEIO_A	I	MFP11	正交编码0 计数信号 A 相输入
		EPWM0_CH4	I/O	MFP12	EPWM0 通道4 输出/捕获输入脚.
		BPWM0_CH1	I/O	MFP13	BPWM0 通道1 输出/捕获输入.
102	102	PE.2	I/O	MFP0	GPIO.
		EBI_ALE	O	MFP2	EBI 地址锁存使能输出脚.
		SD0_DAT0	I/O	MFP3	SD卡接口0 数据位0.
		SPI1_MOSI	I/O	MFP4	SPI1 MOSI (主机输出, 从机输入) 脚.
		SPI3_MOSI	I/O	MFP5	SPI3 MOSI (主机输出, 从机输入) 脚.
		SC0_CLK	O	MFP6	智能卡0 时钟脚.
		USCI0_CLK	I/O	MFP7	USCI0 时钟脚.
		QEIO_B	I	MFP11	正交编码0 计数信号 B 相输入
		EPWM0_CH5	I/O	MFP12	EPWM0 通道5 输出/捕获输入脚.
		BPWM0_CH0	I/O	MFP13	BPWM0 通道0 输出/捕获输入.
103	V <sub>ss</sub>	P	MFP0		数字电路地.
104	V <sub>DD</sub>	P	MFP0		I/O 端口电源和LDO 电源用于内部 PLL 和数字电路
105	105	PE.1	I/O	MFP0	GPIO.
		EBI_AD10	I/O	MFP2	EBI 地址/数据总线位10.
		QSPI0_MISO0	I/O	MFP3	Quad SPI0 MISO0 (主机输入, 从机输出) 脚..
		SC2_DAT	I/O	MFP4	智能卡2 数据脚.
		I2S0_BCLK	O	MFP5	I2S0 位 时钟输出脚.
		SPI1_MISO	I/O	MFP6	SPI1 MISO (主机输入, 从机输出) 脚.

<b>64 脚</b>	<b>128 脚</b>	引脚名称	类型	MFP	描述
		UART3_TXD	O	MFP7	UART3 数据发送脚.
		I2C1_SCL	I/O	MFP8	I2C1 时钟脚.
		UART4_nCTS	I	MFP9	UART4 清除发送输入脚.
	106	PE.0	I/O	MFP0	GPIO.
		EBI_AD11	I/O	MFP2	EBI 地址/数据总线位11.
		QSPI0_MOSI0	I/O	MFP3	Quad SPI0 MOSI0 (主机输出, 从机输入) 脚.
		SC2_CLK	O	MFP4	智能卡 2 时钟脚.
		I2S0_MCLK	O	MFP5	I2S0 主机时钟输出脚.
		SPI1_MOSI	I/O	MFP6	SPI1 MOSI (主机输出, 从机输入) 脚.
		UART3_RXD	I	MFP7	UART3 数据接收脚.
		I2C1_SDA	I/O	MFP8	I2C1 数据输入/输出引脚.
		UART4_nRTS	O	MFP9	UART4 请求发送输出脚.
	107	PH.8	I/O	MFP0	GPIO.
		EBI_AD12	I/O	MFP2	EBI 地址/数据总线位12.
		QSPI0_CLK	I/O	MFP3	Quad SPI0 串行时钟引脚.
		SC2_PWR	O	MFP4	智能卡 2 电源脚.
		I2S0_DI	I	MFP5	I2S0 数据输入脚.
		SPI1_CLK	I/O	MFP6	SPI1 串行时钟引脚.
		UART3_nRTS	O	MFP7	UART3 请求发送输出脚.
		I2C1_SMBAL	O	MFP8	I2C1 SMBus SMBALTER 脚
		I2C2_SCL	I/O	MFP9	I2C2 时钟脚.
	108	UART1_TXD	O	MFP10	UART1 数据发送脚.
		PH.9	I/O	MFP0	GPIO.
		EBI_AD13	I/O	MFP2	EBI 地址/数据总线位13.
		QSPI0_SS	I/O	MFP3	Quad SPI0 从机选择脚.
		SC2_RST	O	MFP4	智能卡 2 复位脚.
		I2S0_DO	O	MFP5	I2S0 数据输出脚
		SPI1_SS	I/O	MFP6	SPI1 从机选择脚.
		UART3_nCTS	I	MFP7	UART3 清除发送输入脚.

64 脚	128 脚	引脚名称	类型	MFP	描述
		I2C1_SMBSUS	O	MFP8	I2C1 SMBus SMBSUS 脚(PMBus 控制脚)
		I2C2_SDA	I/O	MFP9	I2C2 数据输入/输出引脚.
		UART1_RXD	I	MFP10	UART1 数据接收脚.
	109	PH.10	I/O	MFP0	GPIO.
		EBI_AD14	I/O	MFP2	EBI 地址/数据总线位14.
		QSPI0_MISO1	I/O	MFP3	Quad SPI0 MISO1 (主机输入, 从机输出) 脚..
		SC2_nCD	I	MFP4	智能卡 2 卡侦测脚.
		I2S0_LRCK	O	MFP5	I2S0 左右声道选择时钟输出脚.
		SPI1_I2SMCLK	I/O	MFP6	SPI1 I2S 主机时钟输出脚
		UART4_TXD	O	MFP7	UART4 数据发送脚.
		UART0_TXD	O	MFP8	UART0 数据发送脚.
	110	PH.11	I/O	MFP0	GPIO.
		EBI_AD15	I/O	MFP2	EBI 地址/数据总线位15.
		QSPI0_MOSI1	I/O	MFP3	Quad SPI0 MOSI1 (主机输出, 从机输入) 脚.
		UART4_RXD	I	MFP7	UART4 数据接收脚.
		UART0_RXD	I	MFP8	UART0 数据接收脚.
		EPWM0_CH5	I/O	MFP11	EPWM0 通道5 输出/捕获输入脚.
	111	PD.14	I/O	MFP0	GPIO.
		EBI_nCS0	O	MFP2	EBI 片选 0 输出脚.
		SPI3_I2SMCLK	I/O	MFP3	SPI3 I2S 主机时钟输出脚
		SC1_nCD	I	MFP4	智能卡 1 卡侦测脚.
		EPWM0_CH4	I/O	MFP11	EPWM0 通道4 输出/捕获输入脚.
49	112	Vss	P	MFP0	数字电路地.
50	113	LDO_CAP	A	MFP0	LDO 输出脚.
51	114	VDD	P	MFP0	I/O 端口电源和LDO 电源用于内部 PLL 和数字电路
	115	PC.14	I/O	MFP0	GPIO.
		EBI_AD11	I/O	MFP2	EBI 地址/数据总线位11.
		SC1_nCD	I	MFP3	智能卡 1 卡侦测脚.
		SPI0_I2SMCLK	I/O	MFP4	SPI0 I2S 主机时钟输出脚

64 脚	128 脚	引脚名称	类型	MFP	描述
		USCI0_CTL0	I/O	MFP5	USCI0 控制0 脚..
		QSPI0_CLK	I/O	MFP6	Quad SPI0 串行时钟引脚.
		EPWM0_SYNC_IN	I	MFP11	EPWM0 计数器同步触发输入脚
		TM1	I/O	MFP13	定时器1事件计数器输入/反转输出脚.
		USB_VBUS_ST	I	MFP14	USB 外部 VBUS 管理器状态脚.
		HSUSB_VBUS_ST	I	MFP15	HSUSB 外部 VBUS 管理器状态脚.
	53 116	PB.15	I/O	MFP0	GPIO.
		EADC0_CH15	A	MFP1	EADC0 通道15模拟输入.
		EBI_AD12	I/O	MFP2	EBI 地址/数据总线位12.
		SC1_PWR	O	MFP3	智能卡 1 电源脚.
		SPI0_SS	I/O	MFP4	SPI0 从机选择脚.
		USCI0_CTL1	I/O	MFP5	USCI0 控制1 脚..
		UART0_nCTS	I	MFP6	UART0 清除发送输入脚.
		UART3_TXD	O	MFP7	UART3 数据发送脚.
		I2C2_SMBAL	O	MFP8	I2C2 SMBus SMBALTER 脚
		EPWM1_CH0	I/O	MFP11	EPWM1 通道0 输出/捕获输入脚.
		TM0_EXT	I/O	MFP13	定时器0事件计数器输入/反转输出脚.
		USB_VBUS_EN	O	MFP14	USB 外部 VBUS 管理器使能脚.
		HSUSB_VBUS_EN	O	MFP15	HSUSB 外部 VBUS 管理器使能脚.
	54 117	PB.14	I/O	MFP0	GPIO.
		EADC0_CH14	A	MFP1	EADC0 通道14模拟输入.
		EBI_AD13	I/O	MFP2	EBI 地址/数据总线位13.
		SC1_RST	O	MFP3	智能卡 1 复位脚.
		SPI0_CLK	I/O	MFP4	SPI0 串行时钟引脚.
		USCI0_DAT1	I/O	MFP5	USCI0 数据1脚.
		UART0_nRTS	O	MFP6	UART0 请求发送输出脚.
		UART3_RXD	I	MFP7	UART3 数据接收脚.
		I2C2_SMBSUS	O	MFP8	I2C2 SMBus SMBSUS 脚(PMBus 控制脚)
		EPWM1_CH1	I/O	MFP11	EPWM1 通道1 输出/捕获输入脚.

64 脚	128 脚	引脚名称	类型	MFP	描述
		TM1_EXT	I/O	MFP13	定时器1事件计数器输入/反转输出脚.
		CLKO	O	MFP14	时钟输出
55	118	PB.13	I/O	MFP0	GPIO.
		EADC0_CH13	A	MFP1	EADC0 通道13模拟输入.
		DAC1_OUT	A	MFP1	DAC1 通道模拟输出
		ACMP0_P3	A	MFP1	模拟比较器0 正输入 3 脚.
		ACMP1_P3	A	MFP1	模拟比较器1 正输入 3 脚.
		EBI_AD14	I/O	MFP2	EBI 地址/数据总线位14.
		SC1_DAT	I/O	MFP3	智能卡1 数据脚.
		SPI0_MISO	I/O	MFP4	SPI0 MISO (主机输入, 从机输出) 脚.
		USCI0_DAT0	I/O	MFP5	USCI0 数据0脚.
		UART0_TXD	O	MFP6	UART0 数据发送脚.
		UART3_nRTS	O	MFP7	UART3 请求发送输出脚.
		I2C2_SCL	I/O	MFP8	I2C2 时钟脚.
		EPWM1_CH2	I/O	MFP11	EPWM1 通道2 输出/捕获输入脚.
		TM2_EXT	I/O	MFP13	定时器2事件计数器输入/反转输出脚.
56	119	PB.12	I/O	MFP0	GPIO.
		EADC0_CH12	A	MFP1	EADC0 通道12模拟输入.
		DAC0_OUT	A	MFP1	DAC0 通道模拟输出
		ACMP0_P2	A	MFP1	模拟比较器0 正输入 2 脚.
		ACMP1_P2	A	MFP1	模拟比较器1 正输入 2 脚.
		EBI_AD15	I/O	MFP2	EBI 地址/数据总线位15.
		SC1_CLK	O	MFP3	智能卡 1 时钟脚.
		SPI0_MOSI	I/O	MFP4	SPI0 MOSI (主机输出, 从机输入) 脚.
		USCI0_CLK	I/O	MFP5	USCI0 时钟脚.
		UART0_RXD	I	MFP6	UART0 数据接收脚.
		UART3_nCTS	I	MFP7	UART3 清除发送输入脚.
		I2C2_SDA	I/O	MFP8	I2C2 数据输入/输出引脚.
		SD0_nCD	I	MFP9	SD卡接口0 卡侦测输入脚

64 脚	128 脚	引脚名称	类型	MFP	描述
		EPWM1_CH3	I/O	MFP11	EPWM1 通道3 输出/捕获输入脚.
		TM3_EXT	I/O	MFP13	定时器3事件计数器输入/反转输出脚.
57	120	AV <sub>DD</sub>	P	MFP0	内部模拟电路电源 .
58	121	V <sub>REF</sub>	A	MFP0	ADC 参考电压输入. 注意: 该脚需要接一个1uF 电容.
59	122	AV <sub>ss</sub>	P	MFP0	模拟电路地
		PB.11	I/O	MFP0	GPIO.
		EADC0_CH11	A	MFP1	EADC0 通道11模拟输入.
		EBI_ADR16	O	MFP2	EBI 地址总线位16.
		UART0_nCTS	I	MFP5	UART0 清除发送输入脚.
		UART4_TXD	O	MFP6	UART4 数据发送脚.
		I2C1_SCL	I/O	MFP7	I2C1 时钟脚.
		CAN0_TXD	O	MFP8	CAN0 总线发送输出.
		SPI0_I2SMCLK	I/O	MFP9	SPI0 I2S 主机时钟输出脚
		BPWM1_CH0	I/O	MFP10	BPWM1 通道0 输出/捕获输入.
		SPI3_CLK	I/O	MFP11	SPI3 串行时钟引脚.
		HSUSB_VBUS_ST	I	MFP14	HSUSB 外部 VBUS 管理器状态脚.
		PB.10	I/O	MFP0	GPIO.
		EADC0_CH10	A	MFP1	EADC0 通道10模拟输入.
		EBI_ADR17	O	MFP2	EBI 地址总线位17.
		USCI1_CTL0	I/O	MFP4	USCI1 控制0 脚..
		UART0_nRTS	O	MFP5	UART0 请求发送输出脚.
		UART4_RXD	I	MFP6	UART4 数据接收脚.
		I2C1_SDA	I/O	MFP7	I2C1 数据输入/输出引脚.
		CAN0_RXD	I	MFP8	CAN0 总线接收输入
		BPWM1_CH1	I/O	MFP10	BPWM1 通道1 输出/捕获输入.
		SPI3_SS	I/O	MFP11	SPI3 从机选择脚.
		HSUSB_VBUS_EN	O	MFP14	HSUSB 外部 VBUS 管理器使能脚.
62	125	PB.9	I/O	MFP0	GPIO.

64 脚	128 脚	引脚名称	类型	MFP	描述
63	126	EADC0_CH9	A	MFP1	EADC0 通道9模拟输入.
		EBI_ADR18	O	MFP2	EBI 地址总线位18.
		USCI1_CTL1	I/O	MFP4	USCI1 控制1 脚..
		UART0_TXD	O	MFP5	UART0 数据发送脚.
		UART1_nCTS	I	MFP6	UART1 清除发送输入脚.
		I2C1_SMBAL	O	MFP7	I2C1 SMBus SMBALTER 脚
		BPWM1_CH2	I/O	MFP10	BPWM1 通道2 输出/捕获输入.
		SPI3_MISO	I/O	MFP11	SPI3 MISO (主机输入, 从机输出) 脚..
		INT7	I	MFP13	外部中断7输入脚.
64	127	PB.8	I/O	MFP0	GPIO.
		EADC0_CH8	A	MFP1	EADC0 通道8模拟输入.
		EBI_ADR19	O	MFP2	EBI 地址总线位19.
		USCI1_CLK	I/O	MFP4	USCI1 时钟脚.
		UART0_RXD	I	MFP5	UART0 数据接收脚.
		UART1_nRTS	O	MFP6	UART1 请求发送输出脚.
		I2C1_SMBSUS	O	MFP7	I2C1 SMBus SMBSUS 脚(PMBus 控制脚)
		BPWM1_CH3	I/O	MFP10	BPWM1 通道3 输出/捕获输入.
		SPI3_MOSI	I/O	MFP11	SPI3 MOSI (主机输出, 从机输入) 脚.
		INT6	I	MFP13	外部中断6输入脚.
64	127	PB.7	I/O	MFP0	GPIO.
		EADC0_CH7	A	MFP1	EADC0 通道7模拟输入.
		EBI_nWRL	O	MFP2	EBI 低字节写使能输出脚.
		USCI1_DAT0	I/O	MFP4	USCI1 数据0脚.
		CAN1_TXD	O	MFP5	CAN1 总线发送输出.
		UART1_TXD	O	MFP6	UART1 数据发送脚.
		SD1_CMD	I/O	MFP7	SD卡接口1 命令/应答脚
		EBI_nCS0	O	MFP8	EBI 片选 0 输出脚.
		BPWM1_CH4	I/O	MFP10	BPWM1 通道4 输出/捕获输入.
		EPWM1_BRAKE0	I	MFP11	EPWM1 刹车0输入脚.

64 脚	128 脚	引脚名称	类型	MFP	描述
		EPWM1_CH4	I/O	MFP12	EPWM1 通道4 输出/捕获输入脚.
		INT5	I	MFP13	外部中断5输入脚.
		USB_VBUS_ST	I	MFP14	USB 外部 VBUS 管理器状态脚.
		ACMP0_O	O	MFP15	模拟比较器0 输出脚.
1	128	PB.6	I/O	MFP0	GPIO.
		EADC0_CH6	A	MFP1	EADC0 通道6模拟输入.
		EBI_nWRH	O	MFP2	EBI 高字节写使能输出脚
		USCI1_DAT1	I/O	MFP4	USCI1 数据1脚.
		CAN1_RXD	I	MFP5	CAN1 总线接收输入
		UART1_RXD	I	MFP6	UART1 数据接收脚.
		SD1_CLK	O	MFP7	SD卡接口1 时钟输出脚
		EBI_nCS1	O	MFP8	EBI 片选 1 输出脚.
		BPWM1_CH5	I/O	MFP10	BPWM1 通道5 输出/捕获输入.
		EPWM1_BRAKE1	I	MFP11	EPWM1 刹车1输入脚.
		EPWM1_CH5	I/O	MFP12	EPWM1 通道5 输出/捕获输入脚.
		INT4	I	MFP13	外部中断4输入脚.
		USB_VBUS_EN	O	MFP14	USB 外部 VBUS 管理器使能脚.
		ACMP1_O	O	MFP15	模拟比较器1 输出脚.

## 4.2.4 M484 系列引脚描述

64 Pin	64 Pin 2 USB	128 Pin	引脚名	类型	MFP值	描述
2	2	1	PB.5	I/O	MFP0	GPIO.
			EADC0_CH5	A	MFP1	EADC0 通道5模拟输入.
			ACMP1_N	A	MFP1	模拟比较器1 负输入脚.
			EBI_ADR0	O	MFP2	EBI 地址总线位0.
			SD0_DAT3	I/O	MFP3	SD卡接口0 数据位3.
			SPI1_MISO	I/O	MFP5	SPI1 MISO (主机输入, 从机输出) 脚.
			I2C0_SCL	I/O	MFP6	I2C0 时钟脚.
			UART5_TXD	O	MFP7	UART5 数据发送脚.
			USCI1_CTL0	I/O	MFP8	USCI1 控制0 脚..
			SC0_CLK	O	MFP9	智能卡0 时钟脚.
			I2S0_BCLK	O	MFP10	I2S0 位时钟输出脚.
			EPWM0_CH0	I/O	MFP11	EPWM0 通道0 输出/捕获输入脚.
			TM0	I/O	MFP14	定时器0事件计数器输入/反转输出脚.
			INT0	I	MFP15	外部中断0输入脚.
3	3	2	PB.4	I/O	MFP0	GPIO.
			EADC0_CH4	A	MFP1	EADC0 通道4模拟输入.
			ACMP1_P1	A	MFP1	模拟比较器1 正输入 1 脚.
			EBI_ADR1	O	MFP2	EBI 地址总线位1.
			SD0_DAT2	I/O	MFP3	SD卡接口0 数据位2.
			SPI1_MOSI	I/O	MFP5	SPI1 MOSI (主机输出, 从机输入) 脚.
			I2C0_SDA	I/O	MFP6	I2C0 数据输入/输出引脚.
			UART5_RXD	I	MFP7	UART5 数据接收脚.
			USCI1_CTL1	I/O	MFP8	USCI1 控制1 脚..
			SC0_DAT	I/O	MFP9	智能卡0 数据脚.
			I2S0_MCLK	O	MFP10	I2S0 主机时钟输出脚.
			EPWM0_CH1	I/O	MFP11	EPWM0 通道1 输出/捕获输入脚.
			TM1	I/O	MFP14	定时器1事件计数器输入/反转输出脚.
			INT1	I	MFP15	外部中断1输入脚.

64 Pin	64 Pin 2 USB	128 Pin	引脚名	类型	MFP值	描述
4	4	3	PB.3	I/O	MFP0	GPIO.
			EADC0_CH3	A	MFP1	EADC0 通道3模拟输入.
			ACMP0_N	A	MFP1	模拟比较器0 负输入脚.
			EBI_ADR2	O	MFP2	EBI 地址总线位2.
			SD0_DAT1	I/O	MFP3	SD卡接口0 数据位1.
			SPI1_CLK	I/O	MFP5	SPI1 串行时钟引脚.
			UART1_TXD	O	MFP6	UART1 数据发送脚.
			UART5_nRTS	O	MFP7	UART5 请求发送输出脚.
			USCI1_DAT1	I/O	MFP8	USCI1 数据1脚.
			SC0_RST	O	MFP9	智能卡0 复位脚.
			I2S0_DI	I	MFP10	I2S0 数据输入脚.
			EPWM0_CH2	I/O	MFP11	EPWM0 通道2 输出/捕获输入脚.
			TM2	I/O	MFP14	定时器2事件计数器输入/反转输出脚.
			INT2	I	MFP15	外部中断2输入脚.
5	5	4	PB.2	I/O	MFP0	GPIO.
			EADC0_CH2	A	MFP1	EADC0 通道2模拟输入.
			ACMP0_P1	A	MFP1	模拟比较器0 正输入 1 脚.
			OPA0_O	A	MFP1	运放0 输出脚.
			EBI_ADR3	O	MFP2	EBI 地址总线位3.
			SD0_DAT0	I/O	MFP3	SD卡接口0 数据位0.
			SPI1_SS	I/O	MFP5	SPI1 从机选择脚.
			UART1_RXD	I	MFP6	UART1 数据接收脚.
			UART5_nCTS	I	MFP7	UART5 清除发送输入脚.
			USCI1_DAT0	I/O	MFP8	USCI1 数据0脚.
			SC0_PWR	O	MFP9	智能卡0 电源脚.
			I2S0_DO	O	MFP10	I2S0 数据输出脚
			EPWM0_CH3	I/O	MFP11	EPWM0 通道3 输出/捕获输入脚.
			TM3	I/O	MFP14	定时器3事件计数器输入/反转输出脚.
			INT3	I	MFP15	外部中断3输入脚.

64 Pin	64 Pin 2 USB	128 Pin	引脚名	类型	MFP值	描述
		5	PC.12	I/O	MFP0	GPIO.
			EBI_ADR4	O	MFP2	EBI 地址总线位4.
			UART0_TXD	O	MFP3	UART0 数据发送脚.
			I2C0_SCL	I/O	MFP4	I2C0 时钟脚.
			SPI3_MISO	I/O	MFP6	SPI3 MISO (主机输入, 从机输出) 脚..
			SC0_nCD	I	MFP9	智能卡 0 卡侦测脚.
			ECAP1_IC2	I	MFP11	增强型捕获输入单元1输入脚2
			EPWM1_CH0	I/O	MFP12	EPWM1 通道0 输出/捕获输入脚.
			ACMP0_O	O	MFP14	模拟比较器0 输出脚.
		6	PC.11	I/O	MFP0	GPIO.
			EBI_ADR5	O	MFP2	EBI 地址总线位5.
			UART0_RXD	I	MFP3	UART0 数据接收脚.
			I2C0_SDA	I/O	MFP4	I2C0 数据输入/输出引脚.
			SPI3_MOSI	I/O	MFP6	SPI3 MOSI (主机输出, 从机输入) 脚.
			ECAP1_IC1	I	MFP11	增强型捕获输入单元1输入脚1
			EPWM1_CH1	I/O	MFP12	EPWM1 通道1 输出/捕获输入脚.
			ACMP1_O	O	MFP14	模拟比较器1 输出脚.
			PC.10	I/O	MFP0	GPIO.
		7	EBI_ADR6	O	MFP2	EBI 地址总线位 6.
			SPI3_CLK	I/O	MFP6	SPI3 串行时钟引脚.
			UART3_TXD	O	MFP7	UART3 数据发送脚.
			ECAP1_IC0	I	MFP11	增强型捕获输入单元1输入脚0
			EPWM1_CH2	I/O	MFP12	EPWM1 通道2 输出/捕获输入脚.
			PC.9	I/O	MFP0	GPIO.
		8	EBI_ADR7	O	MFP2	EBI 地址总线位 7.
			SPI3_SS	I/O	MFP6	SPI3 从机选择脚.
			UART3_RXD	I	MFP7	UART3 数据接收脚.
			EPWM1_CH3	I/O	MFP12	EPWM1 通道3 输出/捕获输入脚.
			PB.1	I/O	MFP0	GPIO.

64 Pin	64 Pin 2 USB	128 Pin	引脚名	类型	MFP值	描述
			EADC0_CH1	A	MFP1	EADC0 通道1模拟输入.
			OPA0_N	A	MFP1	运放0 负输入脚.
			EBI_ADR8	O	MFP2	EBI 地址总线位 8.
			SD0_CLK	O	MFP3	SD卡接口0 时钟输出脚
			SPI1_I2SMCLK	I/O	MFP5	SPI1 I2S 主机时钟输出脚
			SPI3_I2SMCLK	I/O	MFP6	SPI3 I2S 主机时钟输出脚
			UART2_TXD	O	MFP7	UART2 数据发送脚.
			USCI1_CLK	I/O	MFP8	USCI1 时钟脚.
			I2C1_SCL	I/O	MFP9	I2C1 时钟脚.
			I2S0_LRCK	O	MFP10	I2S0 左右声道选择时钟输出脚.
			EPWM0_CH4	I/O	MFP11	EPWM0 通道4 输出/捕获输入脚.
			EPWM1_CH4	I/O	MFP12	EPWM1 通道4 输出/捕获输入脚.
			EPWM0_BRAKE0	I	MFP13	EPWM0 刹车0输入脚.
		7	PB.0	I/O	MFP0	GPIO.
			EADC0_CH0	A	MFP1	EADC0 通道0模拟输入.
			OPA0_P	A	MFP1	运放0 正输入脚.
			EBI_ADR9	O	MFP2	EBI 地址总线位 9.
			SD0_CMD	I/O	MFP3	SD卡接口0 命令/应答脚
			UART2_RXD	I	MFP7	UART2 数据接收脚.
			SPI0_I2SMCLK	I/O	MFP8	SPI0 I2S 主机时钟输出脚
			I2C1_SDA	I/O	MFP9	I2C1 数据输入/输出引脚.
			EPWM0_CH5	I/O	MFP11	EPWM0 通道5 输出/捕获输入脚.
			EPWM1_CH5	I/O	MFP12	EPWM1 通道5 输出/捕获输入脚.
		8	PA.11	I/O	MFP0	GPIO.
			ACMP0_P0	A	MFP1	模拟比较器0 正输入 0 脚.
			EBI_nRD	O	MFP2	EBI 读使能输出脚.
			V <sub>ss</sub>	P	MFP0	数字电路地.
		12	V <sub>DD</sub>	P	MFP0	I/O 端口电源和LDO 电源用于内部 PLL 和数字电路

64 Pin	64 Pin 2 USB	128 Pin	引脚名	类型	MFP值	描述
			SC2_PWR	O	MFP3	智能卡 2 电源脚.
			SPI2_SS	I/O	MFP4	SPI2 从机选择脚.
			SD1_DAT3	I/O	MFP5	SD卡接口1 数据位3.
			USCI0_CLK	I/O	MFP6	USCI0 时钟脚.
			I2C2_SCL	I/O	MFP7	I2C2 时钟脚.
			BPWM0_CH0	I/O	MFP9	BPWM0 通道0 输出/捕获输入.
			EPWM0_SYNC_OUT	O	MFP10	EPWM0 计数器同步触发输出脚.
			TM0_EXT	I/O	MFP13	定时器0事件计数器输入/反转输出脚.
			DAC1_ST	I	MFP14	DAC1 外部触发输入.
		14	PA.10	I/O	MFP0	GPIO.
			ACMP1_P0	A	MFP1	模拟比较器1 正输入 0 脚.
			OPA1_O	A	MFP1	运放 1 输出脚.
			EBI_nWR	O	MFP2	EBI 写使能输出脚.
			SC2_RST	O	MFP3	智能卡 2 复位脚.
			SPI2_CLK	I/O	MFP4	SPI2 串行时钟引脚.
			SD1_DAT2	I/O	MFP5	SD卡接口1 数据位2.
			USCI0_DAT0	I/O	MFP6	USCI0 数据0脚.
			I2C2_SDA	I/O	MFP7	I2C2 数据输入/输出引脚.
			BPWM0_CH1	I/O	MFP9	BPWM0 通道1 输出/捕获输入.
			QE1_INDEX	I	MFP10	正交编码器1索引输入
			ECAP0_IC0	I	MFP11	增强型捕获输入单元0输入脚0
			TM1_EXT	I/O	MFP13	定时器1事件计数器输入/反转输出脚.
			DAC0_ST	I	MFP14	DAC0 外部触发输入.
		15	PA.9	I/O	MFP0	GPIO.
			OPA1_N	A	MFP1	运放1 负输入端.
			EBI_MCLK	O	MFP2	EBI外部时钟输出脚.
			SC2_DAT	I/O	MFP3	智能卡2 数据脚.
			SPI2_MISO	I/O	MFP4	SPI2 MISO (主机输入, 从机输出) 脚.
			SD1_DAT1	I/O	MFP5	SD卡接口1 数据位1.

64 Pin	64 Pin 2 USB	128 Pin	引脚名	类型	MFP值	描述
			USCI0_DAT1	I/O	MFP6	USCI0 数据1脚.
			UART1_TXD	O	MFP7	UART1 数据发送脚.
			BPWM0_CH2	I/O	MFP9	BPWM0 通道2 输出/捕获输入.
			QEI1_A	I	MFP10	正交编码1 计数信号 A 相输入
			ECAP0_IC1	I	MFP11	增强型捕获输入单元0输入脚1
			TM2_EXT	I/O	MFP13	定时器2事件计数器输入/反转输出脚.
		16	PA.8	I/O	MFP0	GPIO.
			OPA1_P	A	MFP1	运放1 正输入脚.
			EBI_ALE	O	MFP2	EBI 地址锁存使能输出脚.
			SC2_CLK	O	MFP3	智能卡 2 时钟脚.
			SPI2_MOSI	I/O	MFP4	SPI2 MOSI (主机输出, 从机输入) 脚.
			SD1_DAT0	I/O	MFP5	SD卡接口1 数据位0.
			USCI0_CTL1	I/O	MFP6	USCI0 控制1 脚..
			UART1_RXD	I	MFP7	UART1 数据接收脚.
			BPWM0_CH3	I/O	MFP9	BPWM0 通道3 输出/捕获输入.
			QEI1_B	I	MFP10	正交编码1 计数信号 B 相输入
			ECAP0_IC2	I	MFP11	增强型捕获输入单元0输入脚2
			TM3_EXT	I/O	MFP13	定时器3事件计数器输入/反转输出脚.
			INT4	I	MFP15	外部中断4输入脚.
			PC.13	I/O	MFP0	GPIO.
			EBI_ADR10	O	MFP2	EBI 地址总线位10.
			SC2_nCD	I	MFP3	智能卡 2 卡侦测脚.
			SPI2_I2SMCLK	I/O	MFP4	SPI2 I2S 主机时钟输出脚
			USCI0_CTL0	I/O	MFP6	USCI0 控制0 脚..
			UART2_TXD	O	MFP7	UART2 数据发送脚.
			BPWM0_CH4	I/O	MFP9	BPWM0 通道4 输出/捕获输入.
			CLKO	O	MFP13	时钟输出
			EADC0_ST	I	MFP14	EADC0 外部触发输入.
		18	PD.12	I/O	MFP0	GPIO.

64 Pin	64 Pin 2 USB	128 Pin	引脚名	类型	MFP值	描述
			OPA2_O	A	MFP1	运放 2 输出脚.
			EBI_nCS0	O	MFP2	EBI 片选 0 输出脚.
			UART2_RXD	I	MFP7	UART2 数据接收脚.
			BPWM0_CH5	I/O	MFP9	BPWM0 通道5 输出/捕获输入.
			QEIO_INDEX	I	MFP10	正交编码器0索引输入
			CLKO	O	MFP13	时钟输出
			EADC0_ST	I	MFP14	EADC0 外部触发输入.
			INT5	I	MFP15	外部中断5输入脚.
		19	PD.11	I/O	MFP0	GPIO.
			OPA2_N	A	MFP1	运放 2 负输入脚.
			EBI_nCS1	O	MFP2	EBI 片选 1 输出脚.
			UART1_TXD	O	MFP3	UART1 数据发送脚.
			QEIO_A	I	MFP10	正交编码0 计数信号 A 相输入
			INT6	I	MFP15	外部中断6输入脚.
		20	PD.10	I/O	MFP0	GPIO.
			OPA2_P	A	MFP1	运放 2 正输入脚.
			EBI_nCS2	O	MFP2	EBI 片选 2 输出脚.
			UART1_RXD	I	MFP3	UART1 数据接收脚.
			QEIO_B	I	MFP10	正交编码0 计数信号 B 相输入
			INT7	I	MFP15	外部中断7输入脚.
		21	PG.2	I/O	MFP0	GPIO.
			EBI_ADR11	O	MFP2	EBI 地址总线位11.
			SPI2_SS	I/O	MFP3	SPI2 从机选择脚.
			I2C0_SMBAL	O	MFP4	I2C0 SMBus SMBALTER 脚
			I2C1_SCL	I/O	MFP5	I2C1 时钟脚.
			TM0	I/O	MFP13	定时器0事件计数器输入/反转输出脚.
		22	PG.3	I/O	MFP0	GPIO.
			EBI_ADR12	O	MFP2	EBI 地址总线位12.
			SPI2_CLK	I/O	MFP3	SPI2 串行时钟引脚.

64 Pin	64 Pin 2 USB	128 Pin	引脚名	类型	MFP值	描述
			I2C0_SMBSUS	O	MFP4	I2C0 SMBus SMBSUS 脚(PMBus 控制脚)
			I2C1_SDA	I/O	MFP5	I2C1 数据输入/输出引脚.
			TM1	I/O	MFP13	定时器1事件计数器输入/反转输出脚.
		23	PG.4	I/O	MFP0	GPIO.
			EBI_ADR13	O	MFP2	EBI 地址总线位13.
			SPI2_MISO	I/O	MFP3	SPI2 MISO (主机输入, 从机输出) 脚.
			TM2	I/O	MFP13	定时器2事件计数器输入/反转输出脚.
		24	PF.11	I/O	MFP0	GPIO.
			EBI_ADR14	O	MFP2	EBI 地址总线位14.
			SPI2_MOSI	I/O	MFP3	SPI2 MOSI (主机输出, 从机输入) 脚.
			TAMPER5	I/O	MFP10	TAMPER 侦测循环脚 5.
			TM3	I/O	MFP13	定时器3事件计数器输入/反转输出脚.
		25	PF.10	I/O	MFP0	GPIO.
			EBI_ADR15	O	MFP2	EBI 地址总线位15.
			SC0_nCD	I	MFP3	智能卡 0 卡侦测脚.
			I2S0_BCLK	O	MFP4	I2S0 位 时钟输出脚.
			SPI0_I2SMCLK	I/O	MFP5	SPI0 I2S 主机时钟输出脚
			TAMPER4	I/O	MFP10	TAMPER 侦测循环脚 4.
		26	PF.9	I/O	MFP0	GPIO.
			EBI_ADR16	O	MFP2	EBI 地址总线位16.
			SC0_PWR	O	MFP3	智能卡 0 电源脚.
			I2S0_MCLK	O	MFP4	I2S0 主机时钟输出脚.
			SPI0_SS	I/O	MFP5	SPI0 从机选择脚.
			TAMPER3	I/O	MFP10	TAMPER 侦测循环脚 3.
		27	PF.8	I/O	MFP0	GPIO.
			EBI_ADR17	O	MFP2	EBI 地址总线位17.
			SC0_RST	O	MFP3	智能卡 0 复位脚.
			I2S0_DI	I	MFP4	I2S0 数据输入脚.
			SPI0_CLK	I/O	MFP5	SPI0 串行时钟引脚.

64 Pin	64 Pin 2 USB	128 Pin	引脚名	类型	MFP值	描述
			TAMPER2	I/O	MFP10	TAMPER 倾测循环脚 2.
		28	PF.7	I/O	MFP0	GPIO.
			EBI_ADR18	O	MFP2	EBI 地址总线位18.
			SC0_DAT	I/O	MFP3	智能卡0 数据脚.
			I2S0_DO	O	MFP4	I2S0 数据输出脚
			SPI0_MISO	I/O	MFP5	SPI0 MISO (主机输入, 从机输出) 脚.
			UART4_TXD	O	MFP6	UART4 数据发送脚.
			TAMPER1	I/O	MFP10	TAMPER 倾测循环脚 1.
12	12	29	PF.6	I/O	MFP0	GPIO.
			EBI_ADR19	O	MFP2	EBI 地址总线位19.
			SC0_CLK	O	MFP3	智能卡 0 时钟脚.
			I2S0_LRCK	O	MFP4	I2S0 左右声道选择时钟输出脚.
			SPI0_MOSI	I/O	MFP5	SPI0 MOSI (主机输出, 从机输入) 脚.
			UART4_RXD	I	MFP6	UART4 数据接收脚.
			EBI_nCS0	O	MFP7	EBI 片选 0 输出脚.
			TAMPER0	I/O	MFP10	TAMPER 倾测循环脚 0.
13	13	30	VDD	P	MFP0	I/O 端口电源和LDO 电源用于内部 PLL 和数字电路
14	14	31	PF.5	I/O	MFP0	GPIO.
			UART2_RXD	I	MFP2	UART2 数据接收脚.
			UART2_nCTS	I	MFP4	UART2 清除发送输入脚.
			BPWM0_CH4	I/O	MFP8	BPWM0 通道4 输出/捕获输入.
			EPWM0_SYNC_OUT	O	MFP9	EPWM0 计数器同步触发输出脚.
			X32_IN	I	MFP10	外部32.768 kHz 晶振输入脚.
			EADC0_ST	I	MFP11	EADC0 外部触发输入.
15	15	32	PF.4	I/O	MFP0	GPIO.
			UART2_TXD	O	MFP2	UART2 数据发送脚.
			UART2_nRTS	O	MFP4	UART2 请求发送输出脚.
			BPWM0_CH5	I/O	MFP8	BPWM0 通道5 输出/捕获输入.
			X32_OUT	O	MFP10	外部32.768 kHz 晶振输出脚.

64 Pin	64 Pin 2 USB	128 Pin	引脚名	类型	MFP值	描述
		33	PH.4	I/O	MFP0	GPIO.
			EBI_ADR3	O	MFP2	EBI 地址总线位3.
			SPI1_MISO	I/O	MFP3	SPI1 MISO (主机输入, 从机输出) 脚.
		34	PH.5	I/O	MFP0	GPIO.
			EBI_ADR2	O	MFP2	EBI 地址总线位2.
			SPI1_MOSI	I/O	MFP3	SPI1 MOSI (主机输出, 从机输入) 脚.
		35	PH.6	I/O	MFP0	GPIO.
			EBI_ADR1	O	MFP2	EBI 地址总线位1.
			SPI1_CLK	I/O	MFP3	SPI1 串行时钟引脚.
		36	PH.7	I/O	MFP0	GPIO.
			EBI_ADR0	O	MFP2	EBI 地址总线位0.
			SPI1_SS	I/O	MFP3	SPI1 从机选择脚.
	16	37	PF.3	I/O	MFP0	GPIO.
			EBI_nCS0	O	MFP2	EBI 片选 0 输出脚.
			UART0_TXD	O	MFP3	UART0 数据发送脚.
			I2C0_SCL	I/O	MFP4	I2C0 时钟脚.
			XT1_IN	I	MFP10	外部4~24 MHz (高速) 晶振输入脚.
			BPWM1_CH0	I/O	MFP11	BPWM1 通道0 输出/捕获输入.
	17	38	PF.2	I/O	MFP0	GPIO.
			EBI_nCS1	O	MFP2	EBI 片选 1 输出脚.
			UART0_RXD	I	MFP3	UART0 数据接收脚.
			I2C0_SDA	I/O	MFP4	I2C0 数据输入/输出引脚.
			QSPI0_CLK	I/O	MFP5	Quad SPI0 串行时钟引脚.
			XT1_OUT	O	MFP10	外部4~24 MHz (高速) 晶振输出引脚.
			BPWM1_CH1	I/O	MFP11	BPWM1 通道1 输出/捕获输入.
		39	V <sub>SS</sub>	P	MFP0	数字电路地.
		40	V <sub>DD</sub>	P	MFP0	I/O 端口电源和LDO 电源用于内部 PLL 和数字电路
		41	PE.8	I/O	MFP0	GPIO.
			EBI_ADR10	O	MFP2	EBI 地址总线位10.
			I2S0_BCLK	O	MFP4	I2S0 位 时钟输出脚.

64 Pin	64 Pin 2 USB	128 Pin	引脚名	类型	MFP值	描述
			SPI2_CLK	I/O	MFP5	SPI2 串行时钟引脚.
			USCI1_CTL1	I/O	MFP6	USCI1 控制1 脚..
			UART2_TXD	O	MFP7	UART2 数据发送脚.
			EPWM0_CH0	I/O	MFP10	EPWM0 通道0 输出/捕获输入脚.
			EPWM0_BRAKE0	I	MFP11	EPWM0 刹车0输入脚.
			ECAP0_IC0	I	MFP12	增强型捕获输入单元0输入脚0
			TRACE_CLK	O	MFP14	ETM 追踪 时钟输出脚
		42	PE.9	I/O	MFP0	GPIO.
			EBI_ADR11	O	MFP2	EBI 地址总线位11.
			I2S0_MCLK	O	MFP4	I2S0 主机时钟输出脚.
			SPI2_MISO	I/O	MFP5	SPI2 MISO (主机输入, 从机输出) 脚.
			USCI1_CTL0	I/O	MFP6	USCI1 控制0 脚..
			UART2_RXD	I	MFP7	UART2 数据接收脚.
			EPWM0_CH1	I/O	MFP10	EPWM0 通道1 输出/捕获输入脚.
			EPWM0_BRAKE1	I	MFP11	EPWM0 刹车1输入脚.
			ECAP0_IC1	I	MFP12	增强型捕获输入单元0输入脚1
			TRACE_DATA0	O	MFP14	ETM 追踪数据 0 输出脚
		43	PE.10	I/O	MFP0	GPIO.
			EBI_ADR12	O	MFP2	EBI 地址总线位12.
			I2S0_DI	I	MFP4	I2S0 数据输入脚.
			SPI2_MOSI	I/O	MFP5	SPI2 MOSI (主机输出, 从机输入) 脚.
			USCI1_DAT0	I/O	MFP6	USCI1 数据0脚.
			UART3_TXD	O	MFP7	UART3 数据发送脚.
			EPWM0_CH2	I/O	MFP10	EPWM0 通道2 输出/捕获输入脚.
			EPWM1_BRAKE0	I	MFP11	EPWM1 刹车0输入脚.
			ECAP0_IC2	I	MFP12	增强型捕获输入单元0输入脚2
			TRACE_DATA1	O	MFP14	ETM 追踪数据 1 输出脚
		44	PE.11	I/O	MFP0	GPIO.
			EBI_ADR13	O	MFP2	EBI 地址总线位13.

64 Pin	64 Pin 2 USB	128 Pin	引脚名	类型	MFP值	描述
			I2S0_DO	O	MFP4	I2S0 数据输出脚
			SPI2_SS	I/O	MFP5	SPI2 从机选择脚.
			USCI1_DAT1	I/O	MFP6	USCI1 数据1脚.
			UART3_RXD	I	MFP7	UART3 数据接收脚.
			UART1_nCTS	I	MFP8	UART1 清除发送输入脚.
			EPWM0_CH3	I/O	MFP10	EPWM0 通道3 输出/捕获输入脚.
			EPWM1_BRAKE1	I	MFP11	EPWM1 刹车1输入脚.
			ECAP1_IC2	I	MFP13	增强型捕获输入单元1输入脚2
			TRACE_DATA2	O	MFP14	ETM 追踪数据 2 输出脚
		45	PE.12	I/O	MFP0	GPIO.
			EBI_ADR14	O	MFP2	EBI 地址总线位14.
			I2S0_LRCK	O	MFP4	I2S0 左右声道选择时钟输出脚.
			SPI2_I2SMCLK	I/O	MFP5	SPI2 I2S 主机时钟输出脚
			USCI1_CLK	I/O	MFP6	USCI1 时钟脚.
			UART1_nRTS	O	MFP8	UART1 请求发送输出脚.
			EPWM0_CH4	I/O	MFP10	EPWM0 通道4 输出/捕获输入脚.
			ECAP1_IC1	I	MFP13	增强型捕获输入单元1输入脚1
			TRACE_DATA3	O	MFP14	ETM 追踪数据 3 输出脚
		46	PE.13	I/O	MFP0	GPIO.
			EBI_ADR15	O	MFP2	EBI 地址总线位15.
			I2C0_SCL	I/O	MFP4	I2C0 时钟脚.
			UART4_nRTS	O	MFP5	UART4 请求发送输出脚.
			UART1_TXD	O	MFP8	UART1 数据发送脚.
			EPWM0_CH5	I/O	MFP10	EPWM0 通道5 输出/捕获输入脚.
			EPWM1_CH0	I/O	MFP11	EPWM1 通道0 输出/捕获输入脚.
			BPWM1_CH5	I/O	MFP12	BPWM1 通道5 输出/捕获输入.
			ECAP1_IC0	I	MFP13	增强型捕获输入单元1输入脚0
		47	PC.8	I/O	MFP0	GPIO.
			EBI_ADR16	O	MFP2	EBI 地址总线位16.

<b>64 Pin</b>	<b>64 Pin 2 USB</b>	<b>128 Pin</b>	<b>引脚名</b>	<b>类型</b>	<b>MFP值</b>	<b>描述</b>
			I2C0_SDA	I/O	MFP4	I2C0 数据输入/输出引脚.
			UART4_nCTS	I	MFP5	UART4 清除发送输入脚.
			UART1_RXD	I	MFP8	UART1 数据接收脚.
			EPWM1_CH1	I/O	MFP11	EPWM1 通道1 输出/捕获输入脚.
			BPWM1_CH4	I/O	MFP12	BPWM1 通道4 输出/捕获输入.
		48	PC.7	I/O	MFP0	GPIO.
			EBI_AD9	I/O	MFP2	EBI 地址/数据总线位9.
			SPI1_MISO	I/O	MFP4	SPI1 MISO (主机输入, 从机输出) 脚.
			UART4_TXD	O	MFP5	UART4 数据发送脚.
			SC2_PWR	O	MFP6	智能卡 2 电源脚.
			UART0_nCTS	I	MFP7	UART0 清除发送输入脚.
			I2C1_SMBAL	O	MFP8	I2C1 SMBus SMBALTER 脚
			EPWM1_CH2	I/O	MFP11	EPWM1 通道2 输出/捕获输入脚.
			BPWM1_CH0	I/O	MFP12	BPWM1 通道0 输出/捕获输入.
			TM0	I/O	MFP14	定时器0事件计数器输入/反转输出脚.
		49	INT3	I	MFP15	外部中断3输入脚.
			PC.6	I/O	MFP0	GPIO.
			EBI_AD8	I/O	MFP2	EBI 地址/数据总线位8.
			SPI1_MOSI	I/O	MFP4	SPI1 MOSI (主机输出, 从机输入) 脚.
			UART4_RXD	I	MFP5	UART4 数据接收脚.
			SC2_RST	O	MFP6	智能卡 2 复位脚.
			UART0_nRTS	O	MFP7	UART0 请求发送输出脚.
			I2C1_SMBSUS	O	MFP8	I2C1 SMBus SMBSUS 脚(PMBus 控制脚)
			EPWM1_CH3	I/O	MFP11	EPWM1 通道3 输出/捕获输入脚.
			BPWM1_CH1	I/O	MFP12	BPWM1 通道1 输出/捕获输入.
		50	TM1	I/O	MFP14	定时器1事件计数器输入/反转输出脚.
			INT2	I	MFP15	外部中断2输入脚.
		29	PA.7	I/O	MFP0	GPIO.
			EBI_AD7	I/O	MFP2	EBI 地址/数据总线位7.

64 Pin	64 Pin 2 USB	128 Pin	引脚名	类型	MFP值	描述
			SPI1_CLK	I/O	MFP4	SPI1 串行时钟引脚.
			SC2_DAT	I/O	MFP6	智能卡2 数据脚.
			UART0_TXD	O	MFP7	UART0 数据发送脚.
			I2C1_SCL	I/O	MFP8	I2C1 时钟脚.
			EPWM1_CH4	I/O	MFP11	EPWM1 通道4 输出/捕获输入脚.
			BPWM1_CH2	I/O	MFP12	BPWM1 通道2 输出/捕获输入.
			ACMP0_WLAT	I	MFP13	模拟比较器0 窗口锁定输入脚
			TM2	I/O	MFP14	定时器2事件计数器输入/反转输出脚.
			INT1	I	MFP15	外部中断1输入脚.
			PA.6	I/O	MFP0	GPIO.
		51	EBI_AD6	I/O	MFP2	EBI 地址/数据总线位6.
			SPI1_SS	I/O	MFP4	SPI1 从机选择脚.
			SD1_nCD	I	MFP5	SD卡接口1 卡侦测输入脚
			SC2_CLK	O	MFP6	智能卡 2 时钟脚.
			UART0_RXD	I	MFP7	UART0 数据接收脚.
			I2C1_SDA	I/O	MFP8	I2C1 数据输入/输出引脚.
			EPWM1_CH5	I/O	MFP11	EPWM1 通道5 输出/捕获输入脚.
			BPWM1_CH3	I/O	MFP12	BPWM1 通道3 输出/捕获输入.
			ACMP1_WLAT	I	MFP13	模拟比较器1 窗口锁定输入脚
			TM3	I/O	MFP14	定时器3事件计数器输入/反转输出脚.
		52	V <sub>SS</sub>	P	MFP0	数字电路地.
			V <sub>DD</sub>	P	MFP0	I/O 端口电源和LDO 电源用于内部 PLL 和数字电路
24	24	54	LDO_CAP	A	MFP0	LDO 输出脚.
		55	PA.5	I/O	MFP0	GPIO.
			SPIM_D2	I/O	MFP2	SPIM 四线模式数据2 .
			QSPI0_MISO1	I/O	MFP3	Quad SPI0 MISO1 (主机输入, 从机输出) 脚..
			SPI1_I2SMCLK	I/O	MFP4	SPI1 I2S 主机时钟输出脚
			SD1_CMD	I/O	MFP5	SD卡接口1 命令/应答脚

64 Pin	64 Pin 2 USB	128 Pin	引脚名	类型	MFP值	描述
			SC2_nCD	I	MFP6	智能卡 2 卡侦测脚.
			UART0_nCTS	I	MFP7	UART0 清除发送输入脚.
			UART5_TXD	O	MFP8	UART5 数据发送脚.
			I2C0_SCL	I/O	MFP9	I2C0 时钟脚.
			BPWM0_CH5	I/O	MFP12	BPWM0 通道5 输出/捕获输入.
			EPWM0_CH0	I/O	MFP13	EPWM0 通道0 输出/捕获输入脚.
			QEIO_INDEX	I	MFP14	正交编码器0索引输入
		56	PA.4	I/O	MFP0	GPIO.
			SPI_M_D3	I/O	MFP2	SPI_M 四线模式数据3 .
			QSPI0_MOSI1	I/O	MFP3	Quad SPI0 MOSI1 (主机输出, 从机输入) 脚.
			SPI0_I2SMCLK	I/O	MFP4	SPI0 I2S 主机时钟输出脚
			SD1_CLK	O	MFP5	SD卡接口1 时钟输出脚
			SC0_nCD	I	MFP6	智能卡 0 卡侦测脚.
			UART0_nRTS	O	MFP7	UART0 请求发送输出脚.
			UART5_RXD	I	MFP8	UART5 数据接收脚.
			I2C0_SDA	I/O	MFP9	I2C0 数据输入/输出引脚.
			BPWM0_CH4	I/O	MFP12	BPWM0 通道4 输出/捕获输入.
			EPWM0_CH1	I/O	MFP13	EPWM0 通道1 输出/捕获输入脚.
			QEIO_A	I	MFP14	正交编码0 计数信号 A 相输入
		57	PA.3	I/O	MFP0	GPIO.
			SPI_M_SS	I/O	MFP2	SPI_M 从机选择脚.
			QSPI0_SS	I/O	MFP3	Quad SPI0 从机选择脚.
			SPI0_SS	I/O	MFP4	SPI0 从机选择脚.
			SD1_DAT3	I/O	MFP5	SD卡接口1 数据位3.
			SC0_PWR	O	MFP6	智能卡 0 电源脚.
			UART4_TXD	O	MFP7	UART4 数据发送脚.
			UART1_TXD	O	MFP8	UART1 数据发送脚.
			I2C1_SCL	I/O	MFP9	I2C1 时钟脚.
			BPWM0_CH3	I/O	MFP12	BPWM0 通道3 输出/捕获输入.

64 Pin	64 Pin 2 USB	128 Pin	引脚名	类型	MFP值	描述
			EPWM0_CH2	I/O	MFP13	EPWM0 通道2 输出/捕获输入脚.
			QEIO_B	I	MFP14	正交编码0 计数信号 B 相输入
28	28	58	PA.2	I/O	MFP0	GPIO.
			SPIM_CLK	I/O	MFP2	SPIM 串行时钟引脚.
			QSPI0_CLK	I/O	MFP3	Quad SPI0 串行时钟引脚.
			SPI0_CLK	I/O	MFP4	SPI0 串行时钟引脚.
			SD1_DAT2	I/O	MFP5	SD卡接口1 数据位2.
			SC0_RST	O	MFP6	智能卡0 复位脚.
			UART4_RXD	I	MFP7	UART4 数据接收脚.
			UART1_RXD	I	MFP8	UART1 数据接收脚.
			I2C1_SDA	I/O	MFP9	I2C1 数据输入/输出引脚.
			BPWM0_CH2	I/O	MFP12	BPWM0 通道2 输出/捕获输入.
29	29	59	EPWM0_CH3	I/O	MFP13	EPWM0 通道3 输出/捕获输入脚.
			PA.1	I/O	MFP0	GPIO.
			SPIM_MISO	I/O	MFP2	SPIM MISO (主机输入, 从机输出) 脚..
			QSPI0_MISO0	I/O	MFP3	Quad SPI0 MISO0 (主机输入, 从机输出) 脚..
			SPI0_MISO	I/O	MFP4	SPI0 MISO (主机输入, 从机输出) 脚.
			SD1_DAT1	I/O	MFP5	SD卡接口1 数据位1.
			SC0_DAT	I/O	MFP6	智能卡0 数据脚.
			UART0_TXD	O	MFP7	UART0 数据发送脚.
			UART1_nCTS	I	MFP8	UART1 清除发送输入脚.
			I2C2_SCL	I/O	MFP9	I2C2 时钟脚.
			BPWM0_CH1	I/O	MFP12	BPWM0 通道1 输出/捕获输入.
			EPWM0_CH4	I/O	MFP13	EPWM0 通道4 输出/捕获输入脚.
30	30	60	DAC1_ST	I	MFP15	DAC1 外部触发输入.
			PA.0	I/O	MFP0	GPIO.
			SPIM_MOSI	I/O	MFP2	SPIM MOSI (主机输出, 从机输入) 脚.
			QSPI0_MOSI0	I/O	MFP3	Quad SPI0 MOSI0 (主机输出, 从机输入) 脚.
			SPI0_MOSI	I/O	MFP4	SPI0 MOSI (主机输出, 从机输入) 脚.

64 Pin	64 Pin 2 USB	128 Pin	引脚名	类型	MFP值	描述
			SD1_DAT0	I/O	MFP5	SD卡接口1 数据位0.
			SC0_CLK	O	MFP6	智能卡 0 时钟脚.
			UART0_RXD	I	MFP7	UART0 数据接收脚.
			UART1_nRTS	O	MFP8	UART1 请求发送输出脚.
			I2C2_SDA	I/O	MFP9	I2C2 数据输入/输出引脚.
			BPWM0_CH0	I/O	MFP12	BPWM0 通道0 输出/捕获输入.
			EPWM0_CH5	I/O	MFP13	EPWM0 通道5 输出/捕获输入脚.
			DAC0_ST	I	MFP15	DAC0 外部触发输入.
31	31	61	V <sub>DDIO</sub>	P	MFP0	PE.1, PE.8~PE.13的电源.
		62	PE.14	I/O	MFP0	GPIO.
			EBI_AD8	I/O	MFP2	EBI 地址/数据总线位8.
			UART2_TXD	O	MFP3	UART2 数据发送脚.
			SD1_nCD	I	MFP5	SD卡接口1 卡侦测输入脚
		63	PE.15	I/O	MFP0	GPIO.
			EBI_AD9	I/O	MFP2	EBI 地址/数据总线位9.
			UART2_RXD	I	MFP3	UART2 数据接收脚.
32	32	64	nRESET	I	MFP0	外部复位输入：低电平有效，带内部上拉.
		65	PF.0	I/O	MFP0	GPIO.
			UART1_TXD	O	MFP2	UART1 数据发送脚.
			I2C1_SCL	I/O	MFP3	I2C1 时钟脚.
			BPWM1_CH0	I/O	MFP12	BPWM1 通道0 输出/捕获输入.
			ICE_DAT	O	MFP14	串行调试器数据 脚.
		66	PF.1	I/O	MFP0	GPIO.
			UART1_RXD	I	MFP2	UART1 数据接收脚.
			I2C1_SDA	I/O	MFP3	I2C1 数据输入/输出引脚.
			BPWM1_CH1	I/O	MFP12	BPWM1 通道1 输出/捕获输入.
			ICE_CLK	I	MFP14	串行调试器时钟脚.
		67	PD.9	I/O	MFP0	GPIO.
			EBI_AD7	I/O	MFP2	EBI 地址/数据总线位7.

64 Pin	64 Pin 2 USB	128 Pin	引脚名	类型	MFP值	描述
			I2C2_SCL	I/O	MFP3	I2C2 时钟脚.
			UART2_nCTS	I	MFP4	UART2 清除发送输入脚.
		68	PD.8	I/O	MFP0	GPIO.
			EBI_AD6	I/O	MFP2	EBI 地址/数据总线位6.
			I2C2_SDA	I/O	MFP3	I2C2 数据输入/输出引脚.
			UART2_nRTS	O	MFP4	UART2 请求发送输出脚.
		69	PC.5	I/O	MFP0	GPIO.
			EBI_AD5	I/O	MFP2	EBI 地址/数据总线位5.
			SPIM_D2	I/O	MFP3	SPIM 四线模式数据2 .
			QSPI0_MISO1	I/O	MFP4	Quad SPI0 MISO1 (主机输入, 从机输出) 脚..
			UART2_TXD	O	MFP8	UART2 数据发送脚.
			I2C1_SCL	I/O	MFP9	I2C1 时钟脚.
			UART4_TXD	O	MFP11	UART4 数据发送脚.
			EPWM1_CH0	I/O	MFP12	EPWM1 通道0 输出/捕获输入脚.
		70	PC.4	I/O	MFP0	GPIO.
			EBI_AD4	I/O	MFP2	EBI 地址/数据总线位4.
			SPIM_D3	I/O	MFP3	SPIM 四线模式数据3 .
			QSPI0_MOSI1	I/O	MFP4	Quad SPI0 MOSI1 (主机输出, 从机输入) 脚.
			SC1_nCD	I	MFP5	智能卡 1 卡侦测脚.
			I2S0_BCLK	O	MFP6	I2S0 位 时钟输出脚.
			SPI1_I2SMCLK	I/O	MFP7	SPI1 I2S 主机时钟输出脚
			UART2_RXD	I	MFP8	UART2 数据接收脚.
			I2C1_SDA	I/O	MFP9	I2C1 数据输入/输出引脚.
			UART4_RXD	I	MFP11	UART4 数据接收脚.
		71	EPWM1_CH1	I/O	MFP12	EPWM1 通道1 输出/捕获输入脚.
			PC.3	I/O	MFP0	GPIO.
			EBI_AD3	I/O	MFP2	EBI 地址/数据总线位3.
			SPIM_SS	I/O	MFP3	SPIM 从机选择脚.
			QSPI0_SS	I/O	MFP4	Quad SPI0 从机选择脚.

64 Pin	64 Pin 2 USB	128 Pin	引脚名	类型	MFP值	描述	
			SC1_PWR	O	MFP5	智能卡 1 电源脚.	
			I2S0_MCLK	O	MFP6	I2S0 主机时钟输出脚.	
			SPI1_MISO	I/O	MFP7	SPI1 MISO (主机输入, 从机输出) 脚.	
			UART2_nRTS	O	MFP8	UART2 请求发送输出脚.	
			I2C0_SMBAL	O	MFP9	I2C0 SMBus SMBALTER 脚	
			UART3_TXD	O	MFP11	UART3 数据发送脚.	
			EPWM1_CH2	I/O	MFP12	EPWM1 通道2 输出/捕获输入脚.	
		72	PC.2	I/O	MFP0	GPIO.	
			EBI_AD2	I/O	MFP2	EBI 地址/数据总线位2.	
			SPIM_CLK	I/O	MFP3	SPIM 串行时钟引脚.	
			QSPI0_CLK	I/O	MFP4	Quad SPI0 串行时钟引脚.	
			SC1_RST	O	MFP5	智能卡 1 复位脚.	
			I2S0_DI	I	MFP6	I2S0 数据输入脚.	
			SPI1_MOSI	I/O	MFP7	SPI1 MOSI (主机输出, 从机输入) 脚.	
			UART2_nCTS	I	MFP8	UART2 清除发送输入脚.	
			I2C0_SMBSUS	O	MFP9	I2C0 SMBus SMBSUS 脚(PMBus 控制脚)	
			UART3_RXD	I	MFP11	UART3 数据接收脚.	
			EPWM1_CH3	I/O	MFP12	EPWM1 通道3 输出/捕获输入脚.	
	39	35	73	PC.1	I/O	MFP0	GPIO.
				EBI_AD1	I/O	MFP2	EBI 地址/数据总线位1.
				SPIM_MISO	I/O	MFP3	SPIM MISO (主机输入, 从机输出) 脚..
				QSPI0_MISO0	I/O	MFP4	Quad SPI0 MISO0 (主机输入, 从机输出) 脚..
				SC1_DAT	I/O	MFP5	智能卡1 数据脚.
				I2S0_DO	O	MFP6	I2S0 数据输出脚
				SPI1_CLK	I/O	MFP7	SPI1 串行时钟引脚.
				UART2_TXD	O	MFP8	UART2 数据发送脚.
				I2C0_SCL	I/O	MFP9	I2C0 时钟脚.
				EPWM1_CH4	I/O	MFP12	EPWM1 通道4 输出/捕获输入脚.
				ACMP0_O	O	MFP14	模拟比较器0 输出脚.

64 Pin	64 Pin 2 USB	128 Pin	引脚名	类型	MFP值	描述
40	36	74	PC.0	I/O	MFP0	GPIO.
			EBI_AD0	I/O	MFP2	EBI 地址/数据总线位0.
			SPIM_MOSI	I/O	MFP3	SPIM MOSI (主机输出, 从机输入) 脚.
			QSPI0_MOSI0	I/O	MFP4	Quad SPI0 MOSI0 (主机输出, 从机输入) 脚.
			SC1_CLK	O	MFP5	智能卡 1 时钟脚.
			I2S0_LRCK	O	MFP6	I2S0 左右声道选择时钟输出脚.
			SPI1_SS	I/O	MFP7	SPI1 从机选择脚.
			UART2_RXD	I	MFP8	UART2 数据接收脚.
			I2C0_SDA	I/O	MFP9	I2C0 数据输入/输出引脚.
			EPWM1_CH5	I/O	MFP12	EPWM1 通道5 输出/捕获输入脚.
		75	V <sub>SS</sub>	P	MFP0	数字电路地.
		76	V <sub>DD</sub>	P	MFP0	I/O 端口电源和LDO 电源用于内部 PLL 和数字电路
		77	PG.9	I/O	MFP0	GPIO.
			EBI_AD0	I/O	MFP2	EBI 地址/数据总线位0.
			SD1_DAT3	I/O	MFP3	SD卡接口1 数据位3.
			SPIM_D2	I/O	MFP4	SPIM 四线模式数据2 .
			BPWM0_CH5	I/O	MFP12	BPWM0 通道5 输出/捕获输入.
		78	PG.10	I/O	MFP0	GPIO.
			EBI_AD1	I/O	MFP2	EBI 地址/数据总线位1.
			SD1_DAT2	I/O	MFP3	SD卡接口1 数据位2.
			SPIM_D3	I/O	MFP4	SPIM 四线模式数据3 .
			BPWM0_CH4	I/O	MFP12	BPWM0 通道4 输出/捕获输入.
		79	PG.11	I/O	MFP0	GPIO.
			EBI_AD2	I/O	MFP2	EBI 地址/数据总线位2.
			SD1_DAT1	I/O	MFP3	SD卡接口1 数据位1.
			SPIM_SS	I/O	MFP4	SPIM 从机选择脚.
			BPWM0_CH3	I/O	MFP12	BPWM0 通道3 输出/捕获输入.
		80	PG.12	I/O	MFP0	GPIO.

64 Pin	64 Pin 2 USB	128 Pin	引脚名	类型	MFP值	描述
			EBI_AD3	I/O	MFP2	EBI 地址/数据总线位3.
			SD1_DAT0	I/O	MFP3	SD卡接口1 数据位0.
			SPI_M_CLK	I/O	MFP4	SPI_M 串行时钟引脚.
			BPWM0_CH2	I/O	MFP12	BPWM0 通道2 输出/捕获输入.
		81	PG.13	I/O	MFP0	GPIO.
			EBI_AD4	I/O	MFP2	EBI 地址/数据总线位4.
			SD1_CMD	I/O	MFP3	SD卡接口1 命令/应答脚
			SPI_M_MISO	I/O	MFP4	SPI_M MISO (主机输入, 从机输出) 脚..
			BPWM0_CH1	I/O	MFP12	BPWM0 通道1 输出/捕获输入.
		82	PG.14	I/O	MFP0	GPIO.
			EBI_AD5	I/O	MFP2	EBI 地址/数据总线位5.
			SD1_CLK	O	MFP3	SD卡接口1 时钟输出脚
			SPI_M_MOSI	I/O	MFP4	SPI_M MOSI (主机输出, 从机输入) 脚.
			BPWM0_CH0	I/O	MFP12	BPWM0 通道0 输出/捕获输入.
		83	PG.15	I/O	MFP0	GPIO.
			SD1_nCD	I	MFP3	SD卡接口1 卡侦测输入脚
			CLKO	O	MFP14	时钟输出
			EADC0_ST	I	MFP15	EADC0 外部触发输入.
		84	PD.13	I/O	MFP0	GPIO.
			EBI_AD10	I/O	MFP2	EBI 地址/数据总线位10.
			SD0_nCD	I	MFP3	SD卡接口0 卡侦测输入脚
			SPI0_I2SMCLK	I/O	MFP4	SPI0 I2S 主机时钟输出脚
			SPI1_I2SMCLK	I/O	MFP5	SPI1 I2S 主机时钟输出脚
			SC2_nCD	I	MFP7	智能卡 2 卡侦测脚.
	37	85	PA.12	I/O	MFP0	GPIO.
			I2S0_BCLK	O	MFP2	I2S0 位 时钟输出脚.
			UART4_TXD	O	MFP3	UART4 数据发送脚.
			I2C1_SCL	I/O	MFP4	I2C1 时钟脚.
			SPI2_SS	I/O	MFP5	SPI2 从机选择脚.

64 Pin	64 Pin 2 USB	128 Pin	引脚名	类型	MFP值	描述
			SC2_PWR	O	MFP7	智能卡 2 电源脚.
			BPWM1_CH2	I/O	MFP11	BPWM1 通道2 输出/捕获输入.
			QEI1_INDEX	I	MFP12	正交编码器1索引输入
			USB_VBUS	P	MFP14	来自USB主机或HUB的电源
		38	PA.13	I/O	MFP0	GPIO.
			I2S0_MCLK	O	MFP2	I2S0 主机时钟输出脚.
			UART4_RXD	I	MFP3	UART4 数据接收脚.
			I2C1_SDA	I/O	MFP4	I2C1 数据输入/输出引脚.
			SPI2_CLK	I/O	MFP5	SPI2 串行时钟引脚.
			SC2_RST	O	MFP7	智能卡 2 复位脚.
			BPWM1_CH3	I/O	MFP11	BPWM1 通道3 输出/捕获输入.
			QEI1_A	I	MFP12	正交编码1 计数信号 A 相输入
			USB_D-	A	MFP14	USB 差分信号D+.
		39	PA.14	I/O	MFP0	GPIO.
			I2S0_DI	I	MFP2	I2S0 数据输入脚.
			UART0_TXD	O	MFP3	UART0 数据发送脚.
			SPI2_MISO	I/O	MFP5	SPI2 MISO (主机输入, 从机输出) 脚.
			I2C2_SCL	I/O	MFP6	I2C2 时钟脚.
			SC2_DAT	I/O	MFP7	智能卡2 数据脚.
			BPWM1_CH4	I/O	MFP11	BPWM1 通道4 输出/捕获输入.
			QEI1_B	I	MFP12	正交编码1 计数信号 B 相输入
			USB_D+	A	MFP14	USB 差分信号D-.
		40	PA.15	I/O	MFP0	GPIO.
			I2S0_DO	O	MFP2	I2S0 数据输出脚
			UART0_RXD	I	MFP3	UART0 数据接收脚.
			SPI2_MOSI	I/O	MFP5	SPI2 MOSI (主机输出, 从机输入) 脚.
			I2C2_SDA	I/O	MFP6	I2C2 数据输入/输出引脚.
			SC2_CLK	O	MFP7	智能卡 2 时钟脚.
			BPWM1_CH5	I/O	MFP11	BPWM1 通道5 输出/捕获输入.

64 Pin	64 Pin 2 USB	128 Pin	引脚名	类型	MFP值	描述
			EPWM0_SYNC_IN	I	MFP12	EPWM0 计数器同步触发输入脚
			USB_OTG_ID	I	MFP14	USB_ID.
41	41	89	HSUSB_VRES	A	MFP0	HSUSB 模块参考电阻
42	42	90	HSUSB_VDD33	P	MFP0	HSUSB VDD33电源
43	43	91	HSUSB_VBUS	P	MFP0	HSUSB 来自USB主机或HUB的电源
44	44	92	HSUSB_D-	A	MFP0	HSUSB 差分信号D+.
45	45	93	HSUSB_VSS	P	MFP0	HSUSB地
46	46	94	HSUSB_D+	A	MFP0	HSUSB 差分信号D-.
47	47	95	HSUSB_VDD12_CAP	A	MFP0	HSUSB 内部电压管理器输出1.2V 去耦脚. 注意: 该脚需要接一个1uF 电容.
48	48	96	HSUSB_ID	I	MFP0	HSUSB ID.
		97	PE.7	I/O	MFP0	GPIO.
			SD0_CMD	I/O	MFP3	SD卡接口0 命令/应答脚
			SPIM_D2	I/O	MFP4	SPIM 四线模式数据2 .
			UART5_TXD	O	MFP8	UART5 数据发送脚.
			QEI1_INDEX	I	MFP11	正交编码器1索引输入
			EPWM0_CH0	I/O	MFP12	EPWM0 通道0 输出/捕获输入脚.
			BPWM0_CH5	I/O	MFP13	BPWM0 通道5 输出/捕获输入.
		98	PE.6	I/O	MFP0	GPIO.
			SD0_CLK	O	MFP3	SD卡接口0 时钟输出脚
			SPIM_D3	I/O	MFP4	SPIM 四线模式数据3 .
			SPI3_I2SMCLK	I/O	MFP5	SPI3 I2S 主机时钟输出脚
			SC0_nCD	I	MFP6	智能卡 0 卡侦测脚.
			USCI0_CTL0	I/O	MFP7	USCI0 控制0 脚..
			UART5_RXD	I	MFP8	UART5 数据接收脚.
			QEI1_A	I	MFP11	正交编码1 计数信号 A 相输入
			EPWM0_CH1	I/O	MFP12	EPWM0 通道1 输出/捕获输入脚.
			BPWM0_CH4	I/O	MFP13	BPWM0 通道4 输出/捕获输入.
		99	PE.5	I/O	MFP0	GPIO.
			EBI_nRD	O	MFP2	EBI 读使能输出脚.

64 Pin	64 Pin 2 USB	128 Pin	引脚名	类型	MFP值	描述
			SD0_DAT3	I/O	MFP3	SD卡接口0 数据位3.
			SPI_M_SS	I/O	MFP4	SPI_M 从机选择脚.
			SPI3_SS	I/O	MFP5	SPI3 从机选择脚.
			SC0_PWR	O	MFP6	智能卡0 电源脚.
			USCI0_CTL1	I/O	MFP7	USCI0 控制1 脚..
			QEI1_B	I	MFP11	正交编码1 计数信号B 相输入
			EPWM0_CH2	I/O	MFP12	EPWM0 通道2 输出/捕获输入脚.
			BPWM0_CH3	I/O	MFP13	BPWM0 通道3 输出/捕获输入.
		100	PE.4	I/O	MFP0	GPIO.
			EBI_nWR	O	MFP2	EBI 写使能输出脚.
			SD0_DAT2	I/O	MFP3	SD卡接口0 数据位2.
			SPI_M_CLK	I/O	MFP4	SPI_M 串行时钟引脚.
			SPI3_CLK	I/O	MFP5	SPI3 串行时钟引脚.
			SC0_RST	O	MFP6	智能卡0 复位脚.
			USCI0_DAT1	I/O	MFP7	USCI0 数据1脚.
			QEI0_INDEX	I	MFP11	正交编码器0索引输入
			EPWM0_CH3	I/O	MFP12	EPWM0 通道3 输出/捕获输入脚.
			BPWM0_CH2	I/O	MFP13	BPWM0 通道2 输出/捕获输入.
		101	PE.3	I/O	MFP0	GPIO.
			EBI_MCLK	O	MFP2	EBI外部时钟输出脚.
			SD0_DAT1	I/O	MFP3	SD卡接口0 数据位1.
			SPI_M_MISO	I/O	MFP4	SPI_M MISO (主机输入, 从机输出) 脚..
			SPI3_MISO	I/O	MFP5	SPI3 MISO (主机输入, 从机输出) 脚..
			SC0_DAT	I/O	MFP6	智能卡0 数据脚.
			USCI0_DAT0	I/O	MFP7	USCI0 数据0脚.
			QEI0_A	I	MFP11	正交编码0 计数信号A 相输入
			EPWM0_CH4	I/O	MFP12	EPWM0 通道4 输出/捕获输入脚.
			BPWM0_CH1	I/O	MFP13	BPWM0 通道1 输出/捕获输入.
		102	PE.2	I/O	MFP0	GPIO.

64 Pin	64 Pin 2 USB	128 Pin	引脚名	类型	MFP值	描述
			EBI_ALE	O	MFP2	EBI 地址锁存使能输出脚.
			SD0_DAT0	I/O	MFP3	SD卡接口0 数据位0.
			SPI_MOSI	I/O	MFP4	SPI_MOSI (主机输出, 从机输入) 脚.
			SPI3_MOSI	I/O	MFP5	SPI3 MOSI (主机输出, 从机输入) 脚.
			SC0_CLK	O	MFP6	智能卡 0 时钟脚.
			USCI0_CLK	I/O	MFP7	USCI0 时钟脚.
			QEIO_B	I	MFP11	正交编码0 计数信号 B 相输入
			EPWM0_CH5	I/O	MFP12	EPWM0 通道5 输出/捕获输入脚.
			BPWM0_CH0	I/O	MFP13	BPWM0 通道0 输出/捕获输入.
		103	V <sub>ss</sub>	P	MFP0	数字电路地.
		104	V <sub>dd</sub>	P	MFP0	I/O 端口电源和LDO 电源用于内部 PLL 和数字电路
		105	PE.1	I/O	MFP0	GPIO.
			EBI_AD10	I/O	MFP2	EBI 地址/数据总线位10.
			QSPI0_MISO0	I/O	MFP3	Quad SPI0 MISO0 (主机输入, 从机输出) 脚..
			SC2_DAT	I/O	MFP4	智能卡2 数据脚.
			I2S0_BCLK	O	MFP5	I2S0 位 时钟输出脚.
			SPI1_MISO	I/O	MFP6	SPI1 MISO (主机输入, 从机输出) 脚.
			UART3_TXD	O	MFP7	UART3 数据发送脚.
			I2C1_SCL	I/O	MFP8	I2C1 时钟脚.
			UART4_nCTS	I	MFP9	UART4 清除发送输入脚.
		106	PE.0	I/O	MFP0	GPIO.
			EBI_AD11	I/O	MFP2	EBI 地址/数据总线位11.
			QSPI0_MOSI0	I/O	MFP3	Quad SPI0 MOSI0 (主机输出, 从机输入) 脚.
			SC2_CLK	O	MFP4	智能卡 2 时钟脚.
			I2S0_MCLK	O	MFP5	I2S0 主机时钟输出脚.
			SPI1_MOSI	I/O	MFP6	SPI1 MOSI (主机输出, 从机输入) 脚.
			UART3_RXD	I	MFP7	UART3 数据接收脚.
			I2C1_SDA	I/O	MFP8	I2C1 数据输入/输出引脚.
			UART4_nRTS	O	MFP9	UART4 请求发送输出脚.

64 Pin	64 Pin 2 USB	128 Pin	引脚名	类型	MFP值	描述
		107	PH.8	I/O	MFP0	GPIO.
			EBI_AD12	I/O	MFP2	EBI 地址/数据总线位12.
			QSPI0_CLK	I/O	MFP3	Quad SPI0 串行时钟引脚.
			SC2_PWR	O	MFP4	智能卡 2 电源脚.
			I2S0_DI	I	MFP5	I2S0 数据输入脚.
			SPI1_CLK	I/O	MFP6	SPI1 串行时钟引脚.
			UART3_nRTS	O	MFP7	UART3 请求发送输出脚.
			I2C1_SMBAL	O	MFP8	I2C1 SMBus SMBALTER 脚
			I2C2_SCL	I/O	MFP9	I2C2 时钟脚.
			UART1_TXD	O	MFP10	UART1 数据发送脚.
		108	PH.9	I/O	MFP0	GPIO.
			EBI_AD13	I/O	MFP2	EBI 地址/数据总线位13.
			QSPI0_SS	I/O	MFP3	Quad SPI0 从机选择脚.
			SC2_RST	O	MFP4	智能卡 2 复位脚.
			I2S0_DO	O	MFP5	I2S0 数据输出脚
			SPI1_SS	I/O	MFP6	SPI1 从机选择脚.
			UART3_nCTS	I	MFP7	UART3 清除发送输入脚.
			I2C1_SMBSUS	O	MFP8	I2C1 SMBus SMBSUS 脚(PMBus 控制脚)
			I2C2_SDA	I/O	MFP9	I2C2 数据输入/输出引脚.
			UART1_RXD	I	MFP10	UART1 数据接收脚.
		109	PH.10	I/O	MFP0	GPIO.
			EBI_AD14	I/O	MFP2	EBI 地址/数据总线位14.
			QSPI0_MISO1	I/O	MFP3	Quad SPI0 MISO1 (主机输入, 从机输出) 脚..
			SC2_nCD	I	MFP4	智能卡 2 卡侦测脚.
			I2S0_LRCK	O	MFP5	I2S0 左右声道选择时钟输出脚.
			SPI1_I2SMCLK	I/O	MFP6	SPI1 I2S 主机时钟输出脚
			UART4_TXD	O	MFP7	UART4 数据发送脚.
			UART0_TXD	O	MFP8	UART0 数据发送脚.
		110	PH.11	I/O	MFP0	GPIO.

64 Pin	64 Pin 2 USB	128 Pin	引脚名	类型	MFP值	描述
			EBI_AD15	I/O	MFP2	EBI 地址/数据总线位15.
			QSPI0_MOSI1	I/O	MFP3	Quad SPI0 MOSI1 (主机输出, 从机输入) 脚.
			UART4_RXD	I	MFP7	UART4 数据接收脚.
			UART0_RXD	I	MFP8	UART0 数据接收脚.
			EPWM0_CH5	I/O	MFP11	EPWM0 通道5 输出/捕获输入脚.
		111	PD.14	I/O	MFP0	GPIO.
			EBI_nCS0	O	MFP2	EBI 片选 0 输出脚.
			SPI3_I2SMCLK	I/O	MFP3	SPI3 I2S 主机时钟输出脚
			SC1_nCD	I	MFP4	智能卡 1 卡侦测脚.
			EPWM0_CH4	I/O	MFP11	EPWM0 通道4 输出/捕获输入脚.
49	49	112	V <sub>ss</sub>	P	MFP0	数字电路地.
50	50	113	LDO_CAP	A	MFP0	LDO 输出脚.
51	51	114	V <sub>DD</sub>	P	MFP0	I/O 端口电源和LDO 电源用于内部 PLL 和数字电路
		115	PC.14	I/O	MFP0	GPIO.
			EBI_AD11	I/O	MFP2	EBI 地址/数据总线位11.
			SC1_nCD	I	MFP3	智能卡 1 卡侦测脚.
			SPI0_I2SMCLK	I/O	MFP4	SPI0 I2S 主机时钟输出脚
			USCI0_CTL0	I/O	MFP5	USCI0 控制0 脚..
			QSPI0_CLK	I/O	MFP6	Quad SPI0 串行时钟引脚.
			EPWM0_SYNC_IN	I	MFP11	EPWM0 计数器同步触发输入脚
			TM1	I/O	MFP13	定时器1事件计数器输入/反转输出脚.
			USB_VBUS_ST	I	MFP14	USB 外部 VBUS 管理器状态脚.
		116	HSUSB_VBUS_ST	I	MFP15	HSUSB 外部 VBUS 管理器状态脚.
			PB.15	I/O	MFP0	GPIO.
			EADC0_CH15	A	MFP1	EADC0 通道15模拟输入.
			EBI_AD12	I/O	MFP2	EBI 地址/数据总线位12.
			SC1_PWR	O	MFP3	智能卡 1 电源脚.
			SPI0_SS	I/O	MFP4	SPI0 从机选择脚.
			USCI0_CTL1	I/O	MFP5	USCI0 控制1 脚..

64 Pin	64 Pin 2 USB	128 Pin	引脚名	类型	MFP值	描述
			UART0_nCTS	I	MFP6	UART0 清除发送输入脚.
			UART3_TXD	O	MFP7	UART3 数据发送脚.
			I2C2_SMBAL	O	MFP8	I2C2 SMBus SMBALTER 脚
			EPWM1_CH0	I/O	MFP11	EPWM1 通道0 输出/捕获输入脚.
			TM0_EXT	I/O	MFP13	定时器0事件计数器输入/反转输出脚.
			USB_VBUS_EN	O	MFP14	USB 外部 VBUS 管理器使能脚.
			HSUSB_VBUS_EN	O	MFP15	HSUSB 外部 VBUS 管理器使能脚.
		54	PB.14	I/O	MFP0	GPIO.
			EADC0_CH14	A	MFP1	EADC0 通道14模拟输入.
			EBI_AD13	I/O	MFP2	EBI 地址/数据总线位13.
			SC1_RST	O	MFP3	智能卡 1 复位脚.
			SPI0_CLK	I/O	MFP4	SPI0 串行时钟引脚.
			USCI0_DAT1	I/O	MFP5	USCI0 数据1脚.
			UART0_nRTS	O	MFP6	UART0 请求发送输出脚.
			UART3_RXD	I	MFP7	UART3 数据接收脚.
			I2C2_SMBSUS	O	MFP8	I2C2 SMBus SMBSUS 脚(PMBus 控制脚)
			EPWM1_CH1	I/O	MFP11	EPWM1 通道1 输出/捕获输入脚.
			TM1_EXT	I/O	MFP13	定时器1事件计数器输入/反转输出脚.
			CLKO	O	MFP14	时钟输出
		55	PB.13	I/O	MFP0	GPIO.
			EADC0_CH13	A	MFP1	EADC0 通道13模拟输入.
			DAC1_OUT	A	MFP1	DAC1 通道模拟输出
			ACMP0_P3	A	MFP1	模拟比较器0 正输入 3 脚.
			ACMP1_P3	A	MFP1	模拟比较器1 正输入 3 脚.
			EBI_AD14	I/O	MFP2	EBI 地址/数据总线位14.
			SC1_DAT	I/O	MFP3	智能卡1 数据脚.
			SPI0_MISO	I/O	MFP4	SPI0 MISO (主机输入, 从机输出) 脚.
			USCI0_DAT0	I/O	MFP5	USCI0 数据0脚.
			UART0_TXD	O	MFP6	UART0 数据发送脚.

64 Pin	64 Pin 2 USB	128 Pin	引脚名	类型	MFP值	描述
			UART3_nRTS	O	MFP7	UART3 请求发送输出脚.
			I2C2_SCL	I/O	MFP8	I2C2 时钟脚.
			EPWM1_CH2	I/O	MFP11	EPWM1 通道2 输出/捕获输入脚.
			TM2_EXT	I/O	MFP13	定时器2事件计数器输入/反转输出脚.
56	56	119	PB.12	I/O	MFP0	GPIO.
			EADC0_CH12	A	MFP1	EADC0 通道12模拟输入.
			DAC0_OUT	A	MFP1	DAC0 通道模拟输出
			ACMP0_P2	A	MFP1	模拟比较器0 正输入 2 脚.
			ACMP1_P2	A	MFP1	模拟比较器1 正输入 2 脚.
			EBI_AD15	I/O	MFP2	EBI 地址/数据总线位15.
			SC1_CLK	O	MFP3	智能卡 1 时钟脚.
			SPI0_MOSI	I/O	MFP4	SPI0 MOSI (主机输出, 从机输入) 脚.
			USCI0_CLK	I/O	MFP5	USCI0 时钟脚.
			UART0_RXD	I	MFP6	UART0 数据接收脚.
			UART3_nCTS	I	MFP7	UART3 清除发送输入脚.
			I2C2_SDA	I/O	MFP8	I2C2 数据输入/输出引脚.
			SD0_nCD	I	MFP9	SD卡接口0 卡侦测输入脚
			EPWM1_CH3	I/O	MFP11	EPWM1 通道3 输出/捕获输入脚.
			TM3_EXT	I/O	MFP13	定时器3事件计数器输入/反转输出脚.
57	57	120	AV <sub>DD</sub>	P	MFP0	内部模拟电路电源 .
58	58	121	V <sub>REF</sub>	A	MFP0	ADC 参考电压输入. 注意: 该脚需要接一个1uF 电容.
59	59	122	AV <sub>SS</sub>	P	MFP0	模拟电路地
		123	PB.11	I/O	MFP0	GPIO.
			EADC0_CH11	A	MFP1	EADC0 通道11模拟输入.
			EBI_ADR16	O	MFP2	EBI 地址总线位16.
			UART0_nCTS	I	MFP5	UART0 清除发送输入脚.
			UART4_TXD	O	MFP6	UART4 数据发送脚.
			I2C1_SCL	I/O	MFP7	I2C1 时钟脚.

64 Pin	64 Pin 2 USB	128 Pin	引脚名	类型	MFP值	描述
			SPI0_I2SMCLK	I/O	MFP9	SPI0 I2S 主机时钟输出脚
			BPWM1_CH0	I/O	MFP10	BPWM1 通道0 输出/捕获输入.
			SPI3_CLK	I/O	MFP11	SPI3 串行时钟引脚.
			HSUSB_VBUS_ST	I	MFP14	HSUSB 外部 VBUS 管理器状态脚.
61	61	124	PB.10	I/O	MFP0	GPIO.
			EADC0_CH10	A	MFP1	EADC0 通道10模拟输入.
			EBI_ADR17	O	MFP2	EBI 地址总线位17.
			USCI1_CTL0	I/O	MFP4	USCI1 控制0 脚..
			UART0_nRTS	O	MFP5	UART0 请求发送输出脚.
			UART4_RXD	I	MFP6	UART4 数据接收脚.
			I2C1_SDA	I/O	MFP7	I2C1 数据输入/输出引脚.
			BPWM1_CH1	I/O	MFP10	BPWM1 通道1 输出/捕获输入.
			SPI3_SS	I/O	MFP11	SPI3 从机选择脚.
			HSUSB_VBUS_EN	O	MFP14	HSUSB 外部 VBUS 管理器使能脚.
62	62	125	PB.9	I/O	MFP0	GPIO.
			EADC0_CH9	A	MFP1	EADC0 通道9模拟输入.
			EBI_ADR18	O	MFP2	EBI 地址总线位18.
			USCI1_CTL1	I/O	MFP4	USCI1 控制1 脚..
			UART0_TXD	O	MFP5	UART0 数据发送脚.
			UART1_nCTS	I	MFP6	UART1 清除发送输入脚.
			I2C1_SMBAL	O	MFP7	I2C1 SMBus SMBALTER 脚
			BPWM1_CH2	I/O	MFP10	BPWM1 通道2 输出/捕获输入.
			SPI3_MISO	I/O	MFP11	SPI3 MISO (主机输入, 从机输出) 脚..
			INT7	I	MFP13	外部中断7输入脚.
63	63	126	PB.8	I/O	MFP0	GPIO.
			EADC0_CH8	A	MFP1	EADC0 通道8模拟输入.
			EBI_ADR19	O	MFP2	EBI 地址总线位19.
			USCI1_CLK	I/O	MFP4	USCI1 时钟脚.
			UART0_RXD	I	MFP5	UART0 数据接收脚.

64 Pin	64 Pin 2 USB	128 Pin	引脚名	类型	MFP值	描述
		128	UART1_nRTS	O	MFP6	UART1 请求发送输出脚.
			I2C1_SMBSUS	O	MFP7	I2C1 SMBus SMBSUS 脚(PMBus 控制脚)
			BPWM1_CH3	I/O	MFP10	BPWM1 通道3 输出/捕获输入.
			SPI3_MOSI	I/O	MFP11	SPI3 MOSI (主机输出, 从机输入) 脚.
			INT6	I	MFP13	外部中断6输入脚.
	64	127	PB.7	I/O	MFP0	GPIO.
			EADC0_CH7	A	MFP1	EADC0 通道7模拟输入.
			EBI_nWRL	O	MFP2	EBI 低字节写使能输出脚.
			USCI1_DAT0	I/O	MFP4	USCI1 数据0脚.
			UART1_TXD	O	MFP6	UART1 数据发送脚.
			SD1_CMD	I/O	MFP7	SD卡接口1 命令/应答脚
			EBI_nCS0	O	MFP8	EBI 片选 0 输出脚.
			BPWM1_CH4	I/O	MFP10	BPWM1 通道4 输出/捕获输入.
			EPWM1_BRAKE0	I	MFP11	EPWM1 刹车0输入脚.
			EPWM1_CH4	I/O	MFP12	EPWM1 通道4 输出/捕获输入脚.
			INT5	I	MFP13	外部中断5输入脚.
			USB_VBUS_ST	I	MFP14	USB 外部 VBUS 管理器状态脚.
			ACMP0_O	O	MFP15	模拟比较器0 输出脚.
	1	128	PB.6	I/O	MFP0	GPIO.
			EADC0_CH6	A	MFP1	EADC0 通道6模拟输入.
			EBI_nWRH	O	MFP2	EBI 高字节写使能输出脚
			USCI1_DAT1	I/O	MFP4	USCI1 数据1脚.
			UART1_RXD	I	MFP6	UART1 数据接收脚.
			SD1_CLK	O	MFP7	SD卡接口1 时钟输出脚
			EBI_nCS1	O	MFP8	EBI 片选 1 输出脚.
			BPWM1_CH5	I/O	MFP10	BPWM1 通道5 输出/捕获输入.
			EPWM1_BRAKE1	I	MFP11	EPWM1 刹车1输入脚.
			EPWM1_CH5	I/O	MFP12	EPWM1 通道5 输出/捕获输入脚.
			INT4	I	MFP13	外部中断4输入脚.

64 Pin	64 Pin 2 USB	128 Pin	引脚名	类型	MFP值	描述
			USB_VBUS_EN	O	MFP14	USB 外部 VBUS 管理器使能脚.
			ACMP1_O	O	MFP15	模拟比较器1 输出脚.

## 4.2.5 M485 系列引脚描述

32脚	48脚	64脚	128脚	引脚名	类型	MFP值	描述
1	1	2	1	PB.5	I/O	MFP0	GPIO.
				EADC0_CH5	A	MFP1	EADC0 通道5模拟输入.
				ACMP1_N	A	MFP1	模拟比较器1 负输入脚.
				EBI_ADR0	O	MFP2	EBI 地址总线位0.
				SD0_DAT3	I/O	MFP3	SD卡接口0 数据位3.
				SPI1_MISO	I/O	MFP5	SPI1 MISO (主机输入, 从机输出) 脚.
				I2C0_SCL	I/O	MFP6	I2C0 时钟脚.
				UART5_TXD	O	MFP7	UART5 数据发送脚.
				USCI1_CTL0	I/O	MFP8	USCI1 控制0 脚..
				SC0_CLK	O	MFP9	智能卡 0 时钟脚.
				I2S0_BCLK	O	MFP10	I2S0 位 时钟输出脚.
				EPWM0_CH0	I/O	MFP11	EPWM0 通道0 输出/捕获输入脚.
				TM0	I/O	MFP14	定时器0事件计数器输入/反转输出脚.
				INT0	I	MFP15	外部中断0输入脚.
2	2	3	2	PB.4	I/O	MFP0	GPIO.
				EADC0_CH4	A	MFP1	EADC0 通道4模拟输入.
				ACMP1_P1	A	MFP1	模拟比较器1 正输入 1 脚.
				EBI_ADR1	O	MFP2	EBI 地址总线位1.
				SD0_DAT2	I/O	MFP3	SD卡接口0 数据位2.
				SPI1_MOSI	I/O	MFP5	SPI1 MOSI (主机输出, 从机输入) 脚.
				I2C0_SDA	I/O	MFP6	I2C0 数据输入/输出引脚.
				UART5_RXD	I	MFP7	UART5 数据接收脚.
				USCI1_CTL1	I/O	MFP8	USCI1 控制1 脚..
				SC0_DAT	I/O	MFP9	智能卡0 数据脚.
				I2S0_MCLK	O	MFP10	I2S0 主机时钟输出脚.
				EPWM0_CH1	I/O	MFP11	EPWM0 通道1 输出/捕获输入脚.
				TM1	I/O	MFP14	定时器1事件计数器输入/反转输出脚.
				INT1	I	MFP15	外部中断1输入脚.

32脚	48脚	64脚	128脚	引脚名	类型	MFP值	描述
3	3	4	3	PB.3	I/O	MFP0	GPIO.
				EADC0_CH3	A	MFP1	EADC0 通道3模拟输入.
				ACMP0_N	A	MFP1	模拟比较器0 负输入脚.
				EBI_ADR2	O	MFP2	EBI 地址总线位2.
				SD0_DAT1	I/O	MFP3	SD卡接口0 数据位1.
				SPI1_CLK	I/O	MFP5	SPI1 串行时钟引脚.
				UART1_TXD	O	MFP6	UART1 数据发送脚.
				UART5_nRTS	O	MFP7	UART5 请求发送输出脚.
				USCI1_DAT1	I/O	MFP8	USCI1 数据1脚.
				SC0_RST	O	MFP9	智能卡0 复位脚.
				I2S0_DI	I	MFP10	I2S0 数据输入脚.
				EPWM0_CH2	I/O	MFP11	EPWM0 通道2 输出/捕获输入脚.
				TM2	I/O	MFP14	定时器2事件计数器输入/反转输出脚.
				INT2	I	MFP15	外部中断2输入脚.
4	4	5	4	PB.2	I/O	MFP0	GPIO.
				EADC0_CH2	A	MFP1	EADC0 通道2模拟输入.
				ACMP0_P1	A	MFP1	模拟比较器0 正输入1脚.
				OPA0_O	A	MFP1	运放0 输出脚.
				EBI_ADR3	O	MFP2	EBI 地址总线位3.
				SD0_DAT0	I/O	MFP3	SD卡接口0 数据位0.
				SPI1_SS	I/O	MFP5	SPI1 从机选择脚.
				UART1_RXD	I	MFP6	UART1 数据接收脚.
				UART5_nCTS	I	MFP7	UART5 清除发送输入脚.
				USCI1_DAT0	I/O	MFP8	USCI1 数据0脚.
				SC0_PWR	O	MFP9	智能卡0 电源脚.
				I2S0_DO	O	MFP10	I2S0 数据输出脚
				EPWM0_CH3	I/O	MFP11	EPWM0 通道3 输出/捕获输入脚.
				TM3	I/O	MFP14	定时器3事件计数器输入/反转输出脚.
				INT3	I	MFP15	外部中断3输入脚.

32脚	48脚	64脚	128脚	引脚名	类型	MFP值	描述
			5	PC.12	I/O	MFP0	GPIO.
				EBI_ADR4	O	MFP2	EBI 地址总线位4.
				UART0_TXD	O	MFP3	UART0 数据发送脚.
				I2C0_SCL	I/O	MFP4	I2C0 时钟脚.
				SPI3_MISO	I/O	MFP6	SPI3 MISO (主机输入, 从机输出) 脚..
				SC0_nCD	I	MFP9	智能卡 0 卡侦测脚.
				ECAP1_IC2	I	MFP11	增强型捕获输入单元1输入脚2
				EPWM1_CH0	I/O	MFP12	EPWM1 通道0 输出/捕获输入脚.
				ACMP0_O	O	MFP14	模拟比较器0 输出脚.
			6	PC.11	I/O	MFP0	GPIO.
				EBI_ADR5	O	MFP2	EBI 地址总线位5.
				UART0_RXD	I	MFP3	UART0 数据接收脚.
				I2C0_SDA	I/O	MFP4	I2C0 数据输入/输出引脚.
				SPI3_MOSI	I/O	MFP6	SPI3 MOSI (主机输出, 从机输入) 脚.
				ECAP1_IC1	I	MFP11	增强型捕获输入单元1输入脚1
				EPWM1_CH1	I/O	MFP12	EPWM1 通道1 输出/捕获输入脚.
				ACMP1_O	O	MFP14	模拟比较器1 输出脚.
			7	PC.10	I/O	MFP0	GPIO.
				EBI_ADR6	O	MFP2	EBI 地址总线位 6.
				SPI3_CLK	I/O	MFP6	SPI3 串行时钟引脚.
				UART3_TXD	O	MFP7	UART3 数据发送脚.
				CAN1_TXD	O	MFP9	CAN1 总线发送输出.
				ECAP1_IC0	I	MFP11	增强型捕获输入单元1输入脚0
				EPWM1_CH2	I/O	MFP12	EPWM1 通道2 输出/捕获输入脚.
			8	PC.9	I/O	MFP0	GPIO.
				EBI_ADR7	O	MFP2	EBI 地址总线位 7.
				SPI3_SS	I/O	MFP6	SPI3 从机选择脚.
				UART3_RXD	I	MFP7	UART3 数据接收脚.
				CAN1_RXD	I	MFP9	CAN1 总线接收输入

32 脚	48 脚	64 脚	128 脚	引脚名	类型	MFP值	描述
				EPWM1_CH3	I/O	MFP12	EPWM1 通道3 输出/捕获输入脚.
5	5	6	9	PB.1	I/O	MFP0	GPIO.
				EADC0_CH1	A	MFP1	EADC0 通道1模拟输入.
				OPA0_N	A	MFP1	运放0 负输入脚.
				EBI_ADR8	O	MFP2	EBI 地址总线位 8.
				SD0_CLK	O	MFP3	SD卡接口0 时钟输出脚
				SPI1_I2SMCLK	I/O	MFP5	SPI1 I2S 主机时钟输出脚
				SPI3_I2SMCLK	I/O	MFP6	SPI3 I2S 主机时钟输出脚
				UART2_TXD	O	MFP7	UART2 数据发送脚.
				USCI1_CLK	I/O	MFP8	USCI1 时钟脚.
				I2C1_SCL	I/O	MFP9	I2C1 时钟脚.
				I2S0_LRCK	O	MFP10	I2S0 左右声道选择时钟输出脚.
				EPWM0_CH4	I/O	MFP11	EPWM0 通道4 输出/捕获输入脚.
				EPWM1_CH4	I/O	MFP12	EPWM1 通道4 输出/捕获输入脚.
				EPWM0_BRAKE0	I	MFP13	EPWM0 刹车0输入脚.
6	6	7	10	PB.0	I/O	MFP0	GPIO.
				EADC0_CH0	A	MFP1	EADC0 通道0模拟输入.
				OPA0_P	A	MFP1	运放0 正输入脚.
				EBI_ADR9	O	MFP2	EBI 地址总线位 9.
				SD0_CMD	I/O	MFP3	SD卡接口0 命令/应答脚
				UART2_RXD	I	MFP7	UART2 数据接收脚.
				SPI0_I2SMCLK	I/O	MFP8	SPI0 I2S 主机时钟输出脚
				I2C1_SDA	I/O	MFP9	I2C1 数据输入/输出引脚.
				EPWM0_CH5	I/O	MFP11	EPWM0 通道5 输出/捕获输入脚.
				EPWM1_CH5	I/O	MFP12	EPWM1 通道5 输出/捕获输入脚.
				EPWM0_BRAKE1	I	MFP13	EPWM0 刹车1输入脚.
			11	VSS	P	MFP0	数字电路地.
			12	VDD	P	MFP0	I/O 端口电源和LDO 电源用于内部 PLL 和数字电路
	7	8	13	PA.11	I/O	MFP0	GPIO.

32 脚	48 脚	64 脚	128 脚	引脚名	类型	MFP值	描述	
				ACMP0_P0	A	MFP1	模拟比较器0 正输入 0 脚.	
				EBI_nRD	O	MFP2	EBI 读使能输出脚.	
				SC2_PWR	O	MFP3	智能卡 2 电源脚.	
				SPI2_SS	I/O	MFP4	SPI2 从机选择脚.	
				SD1_DAT3	I/O	MFP5	SD卡接口1 数据位3.	
				USCI0_CLK	I/O	MFP6	USCI0 时钟脚.	
				I2C2_SCL	I/O	MFP7	I2C2 时钟脚.	
				BPWM0_CH0	I/O	MFP9	BPWM0 通道0 输出/捕获输入.	
				EPWM0_SYNC_OUT	O	MFP10	EPWM0 计数器同步触发输出脚.	
				TM0_EXT	I/O	MFP13	定时器0事件计数器输入/反转输出脚.	
				DAC1_ST	I	MFP14	DAC1 外部触发输入.	
			8 9 14	PA.10	I/O	MFP0	GPIO.	
				ACMP1_P0	A	MFP1	模拟比较器1 正输入 0 脚.	
				OPA1_O	A	MFP1	运放 1 输出脚.	
				EBI_nWR	O	MFP2	EBI 写使能输出脚.	
				SC2_RST	O	MFP3	智能卡 2 复位脚.	
				SPI2_CLK	I/O	MFP4	SPI2 串行时钟引脚.	
				SD1_DAT2	I/O	MFP5	SD卡接口1 数据位2.	
				USCI0_DAT0	I/O	MFP6	USCI0 数据0脚.	
				I2C2_SDA	I/O	MFP7	I2C2 数据输入/输出引脚.	
				BPWM0_CH1	I/O	MFP9	BPWM0 通道1 输出/捕获输入.	
				QEII_INDEX	I	MFP10	正交编码器1索引输入	
				ECAPO_IC0	I	MFP11	增强型捕获输入单元0输入脚0	
			9 10 15	TM1_EXT	I/O	MFP13	定时器1事件计数器输入/反转输出脚.	
				DAC0_ST	I	MFP14	DAC0 外部触发输入.	
				PA.9	I/O	MFP0	GPIO.	
				OPA1_N	A	MFP1	运放1 负输入端.	
				EBI_MCLK	O	MFP2	EBI外部时钟输出脚.	
				SC2_DAT	I/O	MFP3	智能卡2 数据脚.	

32 脚	48 脚	64 脚	128 脚	引脚名	类型	MFP值	描述
				SPI2_MISO	I/O	MFP4	SPI2 MISO (主机输入, 从机输出) 脚.
				SD1_DAT1	I/O	MFP5	SD卡接口1 数据位1.
				USCI0_DAT1	I/O	MFP6	USCI0 数据1脚.
				UART1_TXD	O	MFP7	UART1 数据发送脚.
				BPWM0_CH2	I/O	MFP9	BPWM0 通道2 输出/捕获输入.
				QEI1_A	I	MFP10	正交编码1 计数信号 A 相输入
				ECAPO_IC1	I	MFP11	增强型捕获输入单元0输入脚1
				TM2_EXT	I/O	MFP13	定时器2事件计数器输入/反转输出脚.
				PA.8	I/O	MFP0	GPIO.
				OPA1_P	A	MFP1	运放1 正输入脚.
				EBI_ALE	O	MFP2	EBI 地址锁存使能输出脚.
				SC2_CLK	O	MFP3	智能卡 2 时钟脚.
				SPI2_MOSI	I/O	MFP4	SPI2 MOSI (主机输出, 从机输入) 脚.
				SD1_DAT0	I/O	MFP5	SD卡接口1 数据位0.
				USCI0_CTL1	I/O	MFP6	USCI0 控制1 脚..
				UART1_RXD	I	MFP7	UART1 数据接收脚.
				BPWM0_CH3	I/O	MFP9	BPWM0 通道3 输出/捕获输入.
				QEI1_B	I	MFP10	正交编码1 计数信号 B 相输入
				ECAPO_IC2	I	MFP11	增强型捕获输入单元0输入脚2
				TM3_EXT	I/O	MFP13	定时器3事件计数器输入/反转输出脚.
				INT4	I	MFP15	外部中断4输入脚.
				PC.13	I/O	MFP0	GPIO.
				EBI_ADR10	O	MFP2	EBI 地址总线位10.
				SC2_nCD	I	MFP3	智能卡 2 卡侦测脚.
				SPI2_I2SMCLK	I/O	MFP4	SPI2 I2S 主机时钟输出脚
				CAN1_TXD	O	MFP5	CAN1 总线发送输出.
				USCI0_CTL0	I/O	MFP6	USCI0 控制0 脚..
				UART2_TXD	O	MFP7	UART2 数据发送脚.
				BPWM0_CH4	I/O	MFP9	BPWM0 通道4 输出/捕获输入.

32 脚	48 脚	64 脚	128 脚	引脚名	类型	MFP值	描述
				CLKO	O	MFP13	时钟输出
				EADC0_ST	I	MFP14	EADC0 外部触发输入.
			18	PD.12	I/O	MFP0	GPIO.
				OPA2_O	A	MFP1	运放 2 输出脚.
				EBI_nCS0	O	MFP2	EBI 片选 0 输出脚.
				CAN1_RXD	I	MFP5	CAN1 总线接收输入
				UART2_RXD	I	MFP7	UART2 数据接收脚.
				BPWM0_CH5	I/O	MFP9	BPWM0 通道5 输出/捕获输入.
				QEIO_INDEX	I	MFP10	正交编码器0索引输入
				CLKO	O	MFP13	时钟输出
				EADC0_ST	I	MFP14	EADC0 外部触发输入.
			19	INT5	I	MFP15	外部中断5输入脚.
				PD.11	I/O	MFP0	GPIO.
				OPA2_N	A	MFP1	运放 2 负输入脚.
				EBI_nCS1	O	MFP2	EBI 片选 1 输出脚.
				UART1_TXD	O	MFP3	UART1 数据发送脚.
				CAN0_TXD	O	MFP4	CAN0 总线发送输出.
				QEIO_A	I	MFP10	正交编码0 计数信号 A 相输入
			20	INT6	I	MFP15	外部中断6输入脚.
				PD.10	I/O	MFP0	GPIO.
				OPA2_P	A	MFP1	运放 2 正输入脚.
				EBI_nCS2	O	MFP2	EBI 片选 2 输出脚.
				UART1_RXD	I	MFP3	UART1 数据接收脚.
				CAN0_RXD	I	MFP4	CAN0 总线接收输入
				QEIO_B	I	MFP10	正交编码0 计数信号 B 相输入
			21	INT7	I	MFP15	外部中断7输入脚.
				PG.2	I/O	MFP0	GPIO.
				EBI_ADR11	O	MFP2	EBI 地址总线位11.
				SPI2_SS	I/O	MFP3	SPI2 从机选择脚.

32 脚	48 脚	64 脚	128 脚	引脚名	类型	MFP值	描述
				I2C0_SMBAL	O	MFP4	I2C0 SMBus SMBALTER 脚
				I2C1_SCL	I/O	MFP5	I2C1 时钟脚.
				TM0	I/O	MFP13	定时器0事件计数器输入/反转输出脚.
			22	PG.3	I/O	MFP0	GPIO.
				EBI_ADR12	O	MFP2	EBI 地址总线位12.
				SPI2_CLK	I/O	MFP3	SPI2 串行时钟引脚.
				I2C0_SMBSUS	O	MFP4	I2C0 SMBus SMBSUS 脚(PMBus 控制脚)
				I2C1_SDA	I/O	MFP5	I2C1 数据输入/输出引脚.
				TM1	I/O	MFP13	定时器1事件计数器输入/反转输出脚.
			23	PG.4	I/O	MFP0	GPIO.
				EBI_ADR13	O	MFP2	EBI 地址总线位13.
				SPI2_MISO	I/O	MFP3	SPI2 MISO (主机输入, 从机输出) 脚.
				TM2	I/O	MFP13	定时器2事件计数器输入/反转输出脚.
			24	PF.11	I/O	MFP0	GPIO.
				EBI_ADR14	O	MFP2	EBI 地址总线位14.
				SPI2_MOSI	I/O	MFP3	SPI2 MOSI (主机输出, 从机输入) 脚.
				TAMPER5	I/O	MFP10	TAMPER 侦测循环脚 5.
				TM3	I/O	MFP13	定时器3事件计数器输入/反转输出脚.
			25	PF.10	I/O	MFP0	GPIO.
				EBI_ADR15	O	MFP2	EBI 地址总线位15.
				SC0_nCD	I	MFP3	智能卡 0 卡侦测脚.
				I2S0_BCLK	O	MFP4	I2S0 位 时钟输出脚.
				SPI0_I2SMCLK	I/O	MFP5	SPI0 I2S 主机时钟输出脚
				TAMPER4	I/O	MFP10	TAMPER 侦测循环脚 4.
			26	PF.9	I/O	MFP0	GPIO.
				EBI_ADR16	O	MFP2	EBI 地址总线位16.
				SC0_PWR	O	MFP3	智能卡 0 电源脚.
				I2S0_MCLK	O	MFP4	I2S0 主机时钟输出脚.
				SPI0_SS	I/O	MFP5	SPI0 从机选择脚.

32 脚	48 脚	64 脚	128 脚	引脚名	类型	MFP值	描述	
				TAMPER3	I/O	MFP10	TAMPER 侦测循环脚 3.	
			27	PF.8	I/O	MFP0	GPIO.	
				EBI_ADR17	O	MFP2	EBI 地址总线位17.	
				SC0_RST	O	MFP3	智能卡 0 复位脚.	
				I2S0_DI	I	MFP4	I2S0 数据输入脚.	
				SPI0_CLK	I/O	MFP5	SPI0 串行时钟引脚.	
				TAMPER2	I/O	MFP10	TAMPER 侦测循环脚 2.	
			28	PF.7	I/O	MFP0	GPIO.	
				EBI_ADR18	O	MFP2	EBI 地址总线位18.	
				SC0_DAT	I/O	MFP3	智能卡0 数据脚.	
				I2S0_DO	O	MFP4	I2S0 数据输出脚	
				SPI0_MISO	I/O	MFP5	SPI0 MISO (主机输入, 从机输出) 脚.	
				UART4_TXD	O	MFP6	UART4 数据发送脚.	
				TAMPER1	I/O	MFP10	TAMPER 侦测循环脚 1.	
			12	29	PF.6	I/O	MFP0	GPIO.
					O	MFP2	EBI 地址总线位19.	
					O	MFP3	智能卡 0 时钟脚.	
					O	MFP4	I2S0 左右声道选择时钟输出脚.	
					I/O	MFP5	SPI0 MOSI (主机输出, 从机输入) 脚.	
					I	MFP6	UART4 数据接收脚.	
					O	MFP7	EBI 片选 0 输出脚.	
					I/O	MFP10	TAMPER 侦测循环脚 0.	
		13	30	V <sub>DD</sub>	P	MFP0	I/O 端口电源和LDO 电源用于内部 PLL 和数字电路	
7	11	14	31	PF.5	I/O	MFP0	GPIO.	
				UART2_RXD	I	MFP2	UART2 数据接收脚.	
				UART2_nCTS	I	MFP4	UART2 清除发送输入脚.	
				BPWM0_CH4	I/O	MFP8	BPWM0 通道4 输出/捕获输入.	
				EPWM0_SYNC_OUT	O	MFP9	EPWM0 计数器同步触发输出脚.	
				X32_IN	I	MFP10	外部32.768 kHz 晶振输入脚.	

32 脚	48 脚	64 脚	128 脚	引脚名	类型	MFP值	描述
				EADC0_ST	I	MFP11	EADC0 外部触发输入.
8	12	15	32	PF.4	I/O	MFP0	GPIO.
				UART2_TXD	O	MFP2	UART2 数据发送脚.
				UART2_nRTS	O	MFP4	UART2 请求发送输出脚.
				BPWM0_CH5	I/O	MFP8	BPWM0 通道5 输出/捕获输入.
				X32_OUT	O	MFP10	外部32.768 kHz 晶振输出脚.
			33	PH.4	I/O	MFP0	GPIO.
				EBI_ADR3	O	MFP2	EBI 地址总线位3.
				SPI1_MISO	I/O	MFP3	SPI1 MISO (主机输入, 从机输出) 脚.
			34	PH.5	I/O	MFP0	GPIO.
				EBI_ADR2	O	MFP2	EBI 地址总线位2.
				SPI1_MOSI	I/O	MFP3	SPI1 MOSI (主机输出, 从机输入) 脚.
			35	PH.6	I/O	MFP0	GPIO.
				EBI_ADR1	O	MFP2	EBI 地址总线位1.
				SPI1_CLK	I/O	MFP3	SPI1 串行时钟引脚.
			36	PH.7	I/O	MFP0	GPIO.
				EBI_ADR0	O	MFP2	EBI 地址总线位0.
				SPI1_SS	I/O	MFP3	SPI1 从机选择脚.
9	13	16	37	PF.3	I/O	MFP0	GPIO.
				EBI_nCS0	O	MFP2	EBI 片选 0 输出脚.
				UART0_TXD	O	MFP3	UART0 数据发送脚.
				I2C0_SCL	I/O	MFP4	I2C0 时钟脚.
				XT1_IN	I	MFP10	外部4~24 MHz (高速) 晶振输入脚.
				BPWM1_CH0	I/O	MFP11	BPWM1 通道0 输出/捕获输入.
10	14	17	38	PF.2	I/O	MFP0	GPIO.
				EBI_nCS1	O	MFP2	EBI 片选 1 输出脚.
				UART0_RXD	I	MFP3	UART0 数据接收脚.
				I2C0_SDA	I/O	MFP4	I2C0 数据输入/输出引脚.
				QSPI0_CLK	I/O	MFP5	Quad SPI0 串行时钟引脚.

32 脚	48 脚	64 脚	128 脚	引脚名	类型	MFP值	描述
				XT1_OUT	O	MFP10	外部4~24 MHz (高速) 晶振输出引脚.
				BPWM1_CH1	I/O	MFP11	BPWM1 通道1 输出/捕获输入.
			39	VSS	P	MFP0	数字电路地.
			40	VDD	P	MFP0	I/O 端口电源和LDO 电源用于内部 PLL 和数字电路
			41	PE.8	I/O	MFP0	GPIO.
				EBI_ADR10	O	MFP2	EBI 地址总线位10.
				I2S0_BCLK	O	MFP4	I2S0 位 时钟输出脚.
				SPI2_CLK	I/O	MFP5	SPI2 串行时钟引脚.
				USCI1_CTL1	I/O	MFP6	USCI1 控制1 脚..
				UART2_TXD	O	MFP7	UART2 数据发送脚.
				EPWM0_CH0	I/O	MFP10	EPWM0 通道0 输出/捕获输入脚.
				EPWM0_BRAKE0	I	MFP11	EPWM0 刹车0输入脚.
				ECAPO_IC0	I	MFP12	增强型捕获输入单元0输入脚0
				TRACE_CLK	O	MFP14	ETM 追踪 时钟输出脚
			42	PE.9	I/O	MFP0	GPIO.
				EBI_ADR11	O	MFP2	EBI 地址总线位11.
				I2S0_MCLK	O	MFP4	I2S0 主机时钟输出脚.
				SPI2_MISO	I/O	MFP5	SPI2 MISO (主机输入, 从机输出) 脚.
				USCI1_CTL0	I/O	MFP6	USCI1 控制0 脚..
				UART2_RXD	I	MFP7	UART2 数据接收脚.
				EPWM0_CH1	I/O	MFP10	EPWM0 通道1 输出/捕获输入脚.
				EPWM0_BRAKE1	I	MFP11	EPWM0 刹车1输入脚.
				ECAPO_IC1	I	MFP12	增强型捕获输入单元0输入脚1
				TRACE_DATA0	O	MFP14	ETM 追踪数据 0 输出脚
			43	PE.10	I/O	MFP0	GPIO.
				EBI_ADR12	O	MFP2	EBI 地址总线位12.
				I2S0_DI	I	MFP4	I2S0 数据输入脚.
				SPI2_MOSI	I/O	MFP5	SPI2 MOSI (主机输出, 从机输入) 脚.
				USCI1_DAT0	I/O	MFP6	USCI1 数据0脚.

32 脚	48 脚	64 脚	128 脚	引脚名	类型	MFP值	描述
				UART3_TXD	O	MFP7	UART3 数据发送脚.
				EPWM0_CH2	I/O	MFP10	EPWM0 通道2 输出/捕获输入脚.
				EPWM1_BRAKE0	I	MFP11	EPWM1 刹车0输入脚.
				ECAP0_IC2	I	MFP12	增强型捕获输入单元0输入脚2
				TRACE_DATA1	O	MFP14	ETM 追踪数据 1 输出脚
			44	PE.11	I/O	MFP0	GPIO.
				EBI_ADR13	O	MFP2	EBI 地址总线位13.
				I2S0_DO	O	MFP4	I2S0 数据输出脚
				SPI2_SS	I/O	MFP5	SPI2 从机选择脚.
				USCI1_DAT1	I/O	MFP6	USCI1 数据1脚.
				UART3_RXD	I	MFP7	UART3 数据接收脚.
				UART1_nCTS	I	MFP8	UART1 清除发送输入脚.
				EPWM0_CH3	I/O	MFP10	EPWM0 通道3 输出/捕获输入脚.
				EPWM1_BRAKE1	I	MFP11	EPWM1 刹车1输入脚.
				ECAP1_IC2	I	MFP13	增强型捕获输入单元1输入脚2
			45	TRACE_DATA2	O	MFP14	ETM 追踪数据 2 输出脚
				PE.12	I/O	MFP0	GPIO.
				EBI_ADR14	O	MFP2	EBI 地址总线位14.
				I2S0_LRCK	O	MFP4	I2S0 左右声道选择时钟输出脚.
				SPI2_I2SMCLK	I/O	MFP5	SPI2 I2S 主机时钟输出脚
				USCI1_CLK	I/O	MFP6	USCI1 时钟脚.
				UART1_nRTS	O	MFP8	UART1 请求发送输出脚.
				EPWM0_CH4	I/O	MFP10	EPWM0 通道4 输出/捕获输入脚.
				ECAP1_IC1	I	MFP13	增强型捕获输入单元1输入脚1
			46	TRACE_DATA3	O	MFP14	ETM 追踪数据 3 输出脚
				PE.13	I/O	MFP0	GPIO.
				EBI_ADR15	O	MFP2	EBI 地址总线位15.
				I2C0_SCL	I/O	MFP4	I2C0 时钟脚.
				UART4_nRTS	O	MFP5	UART4 请求发送输出脚.

32 脚	48 脚	64 脚	128 脚	引脚名	类型	MFP值	描述
				UART1_TXD	O	MFP8	UART1 数据发送脚.
				EPWM0_CH5	I/O	MFP10	EPWM0 通道5 输出/捕获输入脚.
				EPWM1_CHO	I/O	MFP11	EPWM1 通道0 输出/捕获输入脚.
				BPWM1_CH5	I/O	MFP12	BPWM1 通道5 输出/捕获输入.
				ECAP1_IC0	I	MFP13	增强型捕获输入单元1输入脚0
			47	PC.8	I/O	MFP0	GPIO.
				EBI_ADR16	O	MFP2	EBI 地址总线位16.
				I2C0_SDA	I/O	MFP4	I2C0 数据输入/输出引脚.
				UART4_nCTS	I	MFP5	UART4 清除发送输入脚.
				UART1_RXD	I	MFP8	UART1 数据接收脚.
				EPWM1_CH1	I/O	MFP11	EPWM1 通道1 输出/捕获输入脚.
				BPWM1_CH4	I/O	MFP12	BPWM1 通道4 输出/捕获输入.
		18	48	PC.7	I/O	MFP0	GPIO.
				EBI_AD9	I/O	MFP2	EBI 地址/数据总线位9.
				SPI1_MISO	I/O	MFP4	SPI1 MISO (主机输入, 从机输出) 脚.
				UART4_TXD	O	MFP5	UART4 数据发送脚.
				SC2_PWR	O	MFP6	智能卡 2 电源脚.
				UART0_nCTS	I	MFP7	UART0 清除发送输入脚.
				I2C1_SMBAL	O	MFP8	I2C1 SMBus SMBALTER 脚
				EPWM1_CH2	I/O	MFP11	EPWM1 通道2 输出/捕获输入脚.
				BPWM1_CHO	I/O	MFP12	BPWM1 通道0 输出/捕获输入.
				TM0	I/O	MFP14	定时器0事件计数器输入/反转输出脚.
		19	49	INT3	I	MFP15	外部中断3输入脚.
				PC.6	I/O	MFP0	GPIO.
				EBI_AD8	I/O	MFP2	EBI 地址/数据总线位8.
				SPI1_MOSI	I/O	MFP4	SPI1 MOSI (主机输出, 从机输入) 脚.
				UART4_RXD	I	MFP5	UART4 数据接收脚.
				SC2_RST	O	MFP6	智能卡 2 复位脚.
				UART0_nRTS	O	MFP7	UART0 请求发送输出脚.

32 脚	48 脚	64 脚	128 脚	引脚名	类型	MFP值	描述
				I2C1_SMBSUS	O	MFP8	I2C1 SMBus SMBSUS 脚(PMBus 控制脚)
				EPWM1_CH3	I/O	MFP11	EPWM1 通道3 输出/捕获输入脚.
				BPWM1_CH1	I/O	MFP12	BPWM1 通道1 输出/捕获输入.
				TM1	I/O	MFP14	定时器1事件计数器输入/反转输出脚.
				INT2	I	MFP15	外部中断2输入脚.
				PA.7	I/O	MFP0	GPIO.
				EBI_AD7	I/O	MFP2	EBI 地址/数据总线位7.
				SPI1_CLK	I/O	MFP4	SPI1 串行时钟引脚.
				SC2_DAT	I/O	MFP6	智能卡2 数据脚.
				UART0_TXD	O	MFP7	UART0 数据发送脚.
				I2C1_SCL	I/O	MFP8	I2C1 时钟脚.
				EPWM1_CH4	I/O	MFP11	EPWM1 通道4 输出/捕获输入脚.
				BPWM1_CH2	I/O	MFP12	BPWM1 通道2 输出/捕获输入.
				ACMP0_WLAT	I	MFP13	模拟比较器0 窗口锁定输入脚
				TM2	I/O	MFP14	定时器2事件计数器输入/反转输出脚.
				INT1	I	MFP15	外部中断1输入脚.
				PA.6	I/O	MFP0	GPIO.
				EBI_AD6	I/O	MFP2	EBI 地址/数据总线位6.
				SPI1_SS	I/O	MFP4	SPI1 从机选择脚.
				SD1_nCD	I	MFP5	SD卡接口1 卡侦测输入脚
				SC2_CLK	O	MFP6	智能卡 2 时钟脚.
				UART0_RXD	I	MFP7	UART0 数据接收脚.
				I2C1_SDA	I/O	MFP8	I2C1 数据输入/输出引脚.
				EPWM1_CH5	I/O	MFP11	EPWM1 通道5 输出/捕获输入脚.
				BPWM1_CH3	I/O	MFP12	BPWM1 通道3 输出/捕获输入.
				ACMP1_WLAT	I	MFP13	模拟比较器1 窗口锁定输入脚
				TM3	I/O	MFP14	定时器3事件计数器输入/反转输出脚.
				INT0	I	MFP15	外部中断0输入脚.
		22	52	VSS	P	MFP0	数字电路地.

32 脚	48 脚	64 脚	128 脚	引脚名	类型	MFP值	描述
		23	53	VDD	P	MFP0	I/O 端口电源和LDO 电源用于内部 PLL 和数字电路
		24	54	LDO_CAP	A	MFP0	LDO 输出脚.
	17	25	55	PA.5	I/O	MFP0	GPIO.
				SPI_M2	I/O	MFP2	SPI_M2 四线模式数据2 .
				QSPI0_MISO1	I/O	MFP3	Quad SPI0 MISO1 (主机输入, 从机输出) 脚..
				SPI1_I2SMCLK	I/O	MFP4	SPI1 I2S 主机时钟输出脚
				SD1_CMD	I/O	MFP5	SD卡接口1 命令/应答脚
				SC2_nCD	I	MFP6	智能卡 2 卡侦测脚.
				UART0_nCTS	I	MFP7	UART0 清除发送输入脚.
				UART5_TXD	O	MFP8	UART5 数据发送脚.
				I2C0_SCL	I/O	MFP9	I2C0 时钟脚.
				CAN0_TXD	O	MFP10	CAN0 总线发送输出.
				BPWM0_CH5	I/O	MFP12	BPWM0 通道5 输出/捕获输入.
				EPWM0_CH0	I/O	MFP13	EPWM0 通道0 输出/捕获输入脚.
				QEIO_INDEX	I	MFP14	正交编码器0索引输入
	18	26	56	PA.4	I/O	MFP0	GPIO.
				SPI_M3	I/O	MFP2	SPI_M3 四线模式数据3 .
				QSPI0_MOSI1	I/O	MFP3	Quad SPI0 MOSI1 (主机输出, 从机输入) 脚.
				SPI0_I2SMCLK	I/O	MFP4	SPI0 I2S 主机时钟输出脚
				SD1_CLK	O	MFP5	SD卡接口1 时钟输出脚
				SC0_nCD	I	MFP6	智能卡 0 卡侦测脚.
				UART0_nRTS	O	MFP7	UART0 请求发送输出脚.
				UART5_RXD	I	MFP8	UART5 数据接收脚.
				I2C0_SDA	I/O	MFP9	I2C0 数据输入/输出引脚.
				CAN0_RXD	I	MFP10	CAN0 总线接收输入
				BPWM0_CH4	I/O	MFP12	BPWM0 通道4 输出/捕获输入.
				EPWM0_CH1	I/O	MFP13	EPWM0 通道1 输出/捕获输入脚.
				QEIO_A	I	MFP14	正交编码器0 计数信号 A 相输入
11	19	27	57	PA.3	I/O	MFP0	GPIO.

32脚	48脚	64脚	128脚	引脚名	类型	MFP值	描述
				SPIM_SS	I/O	MFP2	SPIM 从机选择脚.
				QSPI0_SS	I/O	MFP3	Quad SPI0 从机选择脚.
				SPI0_SS	I/O	MFP4	SPI0 从机选择脚.
				SD1_DAT3	I/O	MFP5	SD卡接口1 数据位3.
				SC0_PWR	O	MFP6	智能卡0 电源脚.
				UART4_TXD	O	MFP7	UART4 数据发送脚.
				UART1_TXD	O	MFP8	UART1 数据发送脚.
				I2C1_SCL	I/O	MFP9	I2C1 时钟脚.
				BPWM0_CH3	I/O	MFP12	BPWM0 通道3 输出/捕获输入.
				EPWM0_CH2	I/O	MFP13	EPWM0 通道2 输出/捕获输入脚.
				QEIO_B	I	MFP14	正交编码0 计数信号B 相输入
12	20	28	58	PA.2	I/O	MFP0	GPIO.
				SPIM_CLK	I/O	MFP2	SPIM 串行时钟引脚.
				QSPI0_CLK	I/O	MFP3	Quad SPI0 串行时钟引脚.
				SPI0_CLK	I/O	MFP4	SPI0 串行时钟引脚.
				SD1_DAT2	I/O	MFP5	SD卡接口1 数据位2.
				SC0_RST	O	MFP6	智能卡0 复位脚.
				UART4_RXD	I	MFP7	UART4 数据接收脚.
				UART1_RXD	I	MFP8	UART1 数据接收脚.
				I2C1_SDA	I/O	MFP9	I2C1 数据输入/输出引脚.
				BPWM0_CH2	I/O	MFP12	BPWM0 通道2 输出/捕获输入.
13	21	29	59	PA.1	I/O	MFP0	GPIO.
				SPIM_MISO	I/O	MFP2	SPIM MISO (主机输入, 从机输出) 脚..
				QSPI0_MISO0	I/O	MFP3	Quad SPI0 MISO0 (主机输入, 从机输出) 脚..
				SPI0_MISO	I/O	MFP4	SPI0 MISO (主机输入, 从机输出) 脚.
				SD1_DAT1	I/O	MFP5	SD卡接口1 数据位1.
				SC0_DAT	I/O	MFP6	智能卡0 数据脚.
				UART0_TXD	O	MFP7	UART0 数据发送脚.

32 脚	48 脚	64 脚	128 脚	引脚名	类型	MFP值	描述
				UART1_nCTS	I	MFP8	UART1 清除发送输入脚.
				I2C2_SCL	I/O	MFP9	I2C2 时钟脚.
				BPWM0_CH1	I/O	MFP12	BPWM0 通道1 输出/捕获输入.
				EPWM0_CH4	I/O	MFP13	EPWM0 通道4 输出/捕获输入脚.
				DAC1_ST	I	MFP15	DAC1 外部触发输入.
14	22	30	60	PA.0	I/O	MFP0	GPIO.
				SPIM_MOSI	I/O	MFP2	SPIM MOSI (主机输出, 从机输入) 脚.
				QSPI0_MOSIO	I/O	MFP3	Quad SPI0 MOSIO (主机输出, 从机输入) 脚.
				SPI0_MOSI	I/O	MFP4	SPI0 MOSI (主机输出, 从机输入) 脚.
				SD1_DAT0	I/O	MFP5	SD卡接口1 数据位0.
				SC0_CLK	O	MFP6	智能卡 0 时钟脚.
				UART0_RXD	I	MFP7	UART0 数据接收脚.
				UART1_nRTS	O	MFP8	UART1 请求发送输出脚.
				I2C2_SDA	I/O	MFP9	I2C2 数据输入/输出引脚.
				BPWM0_CH0	I/O	MFP12	BPWM0 通道0 输出/捕获输入.
				EPWM0_CH5	I/O	MFP13	EPWM0 通道5 输出/捕获输入脚.
				DAC0_ST	I	MFP15	DAC0 外部触发输入.
15	23	31	61	VDDIO	P	MFP0	PE.1, PE.8~PE.13的电源.
			62	PE.14	I/O	MFP0	GPIO.
				EBI_AD8	I/O	MFP2	EBI 地址/数据总线位8.
				UART2_TXD	O	MFP3	UART2 数据发送脚.
				CAN0_TXD	O	MFP4	CAN0 总线发送输出.
				SD1_nCD	I	MFP5	SD卡接口1 卡侦测输入脚
			63	PE.15	I/O	MFP0	GPIO.
				EBI_AD9	I/O	MFP2	EBI 地址/数据总线位9.
				UART2_RXD	I	MFP3	UART2 数据接收脚.
				CAN0_RXD	I	MFP4	CAN0 总线接收输入
16	24	32	64	nRESET	I	MFP0	外部复位输入: 低电平有效, 带内部上拉.
17	25	33	65	PF.0	I/O	MFP0	GPIO.

32脚	48脚	64脚	128脚	引脚名	类型	MFP值	描述
				UART1_TXD	O	MFP2	UART1 数据发送脚.
				I2C1_SCL	I/O	MFP3	I2C1 时钟脚.
				BPWM1_CH0	I/O	MFP12	BPWM1 通道0 输出/捕获输入.
				ICE_DAT	O	MFP14	串行调试器数据输入.
			66	PF.1	I/O	MFP0	GPIO.
				UART1_RXD	I	MFP2	UART1 数据接收脚.
				I2C1_SDA	I/O	MFP3	I2C1 数据输入/输出引脚.
				BPWM1_CH1	I/O	MFP12	BPWM1 通道1 输出/捕获输入.
				ICE_CLK	I	MFP14	串行调试器时钟脚.
			67	PD.9	I/O	MFP0	GPIO.
				EBI_AD7	I/O	MFP2	EBI 地址/数据总线位7.
				I2C2_SCL	I/O	MFP3	I2C2 时钟脚.
				UART2_nCTS	I	MFP4	UART2 清除发送输入脚.
			68	PD.8	I/O	MFP0	GPIO.
				EBI_AD6	I/O	MFP2	EBI 地址/数据总线位6.
				I2C2_SDA	I/O	MFP3	I2C2 数据输入/输出引脚.
				UART2_nRTS	O	MFP4	UART2 请求发送输出脚.
			69	PC.5	I/O	MFP0	GPIO.
				EBI_AD5	I/O	MFP2	EBI 地址/数据总线位5.
				SPIM_D2	I/O	MFP3	SPIM 四线模式数据2 .
				QSPI0_MISO1	I/O	MFP4	Quad SPI0 MISO1 (主机输入, 从机输出) 脚..
				UART2_TXD	O	MFP8	UART2 数据发送脚.
				I2C1_SCL	I/O	MFP9	I2C1 时钟脚.
				CAN0_TXD	O	MFP10	CAN0 总线发送输出.
				UART4_TXD	O	MFP11	UART4 数据发送脚.
				EPWM1_CH0	I/O	MFP12	EPWM1 通道0 输出/捕获输入脚.
			70	PC.4	I/O	MFP0	GPIO.
				EBI_AD4	I/O	MFP2	EBI 地址/数据总线位4.
				SPIM_D3	I/O	MFP3	SPIM 四线模式数据3 .

32 脚	48 脚	64 脚	128 脚	引脚名	类型	MFP值	描述
				QSPI0_MOSI1	I/O	MFP4	Quad SPI0 MOSI1 (主机输出, 从机输入) 脚.
				SC1_nCD	I	MFP5	智能卡 1 卡侦测脚.
				I2S0_BCLK	O	MFP6	I2S0 位时钟输出脚.
				SPI1_I2SMCLK	I/O	MFP7	SPI1 I2S 主机时钟输出脚
				UART2_RXD	I	MFP8	UART2 数据接收脚.
				I2C1_SDA	I/O	MFP9	I2C1 数据输入/输出引脚.
				CAN0_RXD	I	MFP10	CAN0 总线接收输入
				UART4_RXD	I	MFP11	UART4 数据接收脚.
				EPWM1_CH1	I/O	MFP12	EPWM1 通道1 输出/捕获输入脚.
				PC.3	I/O	MFP0	GPIO.
				EBI_AD3	I/O	MFP2	EBI 地址/数据总线位3.
				SPIM_SS	I/O	MFP3	SPIM 从机选择脚.
				QSPI0_SS	I/O	MFP4	Quad SPI0 从机选择脚.
				SC1_PWR	O	MFP5	智能卡 1 电源脚.
				I2S0_MCLK	O	MFP6	I2S0 主机时钟输出脚.
				SPI1_MISO	I/O	MFP7	SPI1 MISO (主机输入, 从机输出) 脚.
				UART2_nRTS	O	MFP8	UART2 请求发送输出脚.
				I2C0_SMBAL	O	MFP9	I2C0 SMBus SMBALTER 脚
				CAN1_TXD	O	MFP10	CAN1 总线发送输出.
				UART3_TXD	O	MFP11	UART3 数据发送脚.
				EPWM1_CH2	I/O	MFP12	EPWM1 通道2 输出/捕获输入脚.
				PC.2	I/O	MFP0	GPIO.
				EBI_AD2	I/O	MFP2	EBI 地址/数据总线位2.
				SPIM_CLK	I/O	MFP3	SPIM 串行时钟引脚.
				QSPI0_CLK	I/O	MFP4	Quad SPI0 串行时钟引脚.
				SC1_RST	O	MFP5	智能卡 1 复位脚.
				I2S0_DI	I	MFP6	I2S0 数据输入脚.
				SPI1_MOSI	I/O	MFP7	SPI1 MOSI (主机输出, 从机输入) 脚.
				UART2_nCTS	I	MFP8	UART2 清除发送输入脚.

32脚	48脚	64脚	128脚	引脚名	类型	MFP值	描述
				I2C0_SMBSUS	O	MFP9	I2C0 SMBus SMBSUS 脚(PMBus 控制脚)
				CAN1_RXD	I	MFP10	CAN1 总线接收输入
				UART3_RXD	I	MFP11	UART3 数据接收脚.
				EPWM1_CH3	I/O	MFP12	EPWM1 通道3 输出/捕获输入脚.
19	31	39	73	PC.1	I/O	MFP0	GPIO.
				EBI_AD1	I/O	MFP2	EBI 地址/数据总线位1.
				SPIM_MISO	I/O	MFP3	SPIM MISO (主机输入, 从机输出) 脚..
				QSPI0_MISO0	I/O	MFP4	Quad SPI0 MISO0 (主机输入, 从机输出) 脚..
				SC1_DAT	I/O	MFP5	智能卡1 数据脚.
				I2S0_DO	O	MFP6	I2S0 数据输出脚
				SPI1_CLK	I/O	MFP7	SPI1 串行时钟引脚.
				UART2_TXD	O	MFP8	UART2 数据发送脚.
				I2C0_SCL	I/O	MFP9	I2C0 时钟脚.
				EPWM1_CH4	I/O	MFP12	EPWM1 通道4 输出/捕获输入脚.
20	32	40	74	ACMP0_O	O	MFP14	模拟比较器0 输出脚.
				PC.0	I/O	MFP0	GPIO.
				EBI_AD0	I/O	MFP2	EBI 地址/数据总线位0.
				SPIM_MOSI	I/O	MFP3	SPIM MOSI (主机输出, 从机输入) 脚.
				QSPI0_MOSI0	I/O	MFP4	Quad SPI0 MOSI0 (主机输出, 从机输入) 脚.
				SC1_CLK	O	MFP5	智能卡 1 时钟脚.
				I2S0_LRCK	O	MFP6	I2S0 左右声道选择时钟输出脚.
				SPI1_SS	I/O	MFP7	SPI1 从机选择脚.
				UART2_RXD	I	MFP8	UART2 数据接收脚.
				I2C0_SDA	I/O	MFP9	I2C0 数据输入/输出引脚.
				EPWM1_CH5	I/O	MFP12	EPWM1 通道5 输出/捕获输入脚.
				ACMP1_O	O	MFP14	模拟比较器1 输出脚.
				VSS	P	MFP0	数字电路地.
				VDD	P	MFP0	I/O 端口电源和LDO 电源用于内部 PLL 和数字电路
				PG.9	I/O	MFP0	GPIO.

32 脚	48 脚	64 脚	128 脚	引脚名	类型	MFP值	描述
				EBI_AD0	I/O	MFP2	EBI 地址/数据总线位0.
				SD1_DAT3	I/O	MFP3	SD卡接口1 数据位3.
				SPI_M_D2	I/O	MFP4	SPI_M 四线模式数据2 .
				BPWM0_CH5	I/O	MFP12	BPWM0 通道5 输出/捕获输入.
			78	PG.10	I/O	MFP0	GPIO.
				EBI_AD1	I/O	MFP2	EBI 地址/数据总线位1.
				SD1_DAT2	I/O	MFP3	SD卡接口1 数据位2.
				SPI_M_D3	I/O	MFP4	SPI_M 四线模式数据3 .
				BPWM0_CH4	I/O	MFP12	BPWM0 通道4 输出/捕获输入.
			79	PG.11	I/O	MFP0	GPIO.
				EBI_AD2	I/O	MFP2	EBI 地址/数据总线位2.
				SD1_DAT1	I/O	MFP3	SD卡接口1 数据位1.
				SPI_M_SS	I/O	MFP4	SPI_M 从机选择脚.
				BPWM0_CH3	I/O	MFP12	BPWM0 通道3 输出/捕获输入.
			80	PG.12	I/O	MFP0	GPIO.
				EBI_AD3	I/O	MFP2	EBI 地址/数据总线位3.
				SD1_DAT0	I/O	MFP3	SD卡接口1 数据位0.
				SPI_M_CLK	I/O	MFP4	SPI_M 串行时钟引脚.
				BPWM0_CH2	I/O	MFP12	BPWM0 通道2 输出/捕获输入.
			81	PG.13	I/O	MFP0	GPIO.
				EBI_AD4	I/O	MFP2	EBI 地址/数据总线位4.
				SD1_CMD	I/O	MFP3	SD卡接口1 命令/应答脚
				SPI_M_MISO	I/O	MFP4	SPI_M MISO (主机输入, 从机输出) 脚..
				BPWM0_CH1	I/O	MFP12	BPWM0 通道1 输出/捕获输入.
			82	PG.14	I/O	MFP0	GPIO.
				EBI_AD5	I/O	MFP2	EBI 地址/数据总线位5.
				SD1_CLK	I/O	MFP3	SD卡接口1 时钟输出脚
				SPI_M_MOSI	I/O	MFP4	SPI_M MOSI (主机输出, 从机输入) 脚.
				BPWM0_CH0	I/O	MFP12	BPWM0 通道0 输出/捕获输入.

32 脚	48 脚	64 脚	128 脚	引脚名	类型	MFP值	描述
			83	PG.15	I/O	MFP0	GPIO.
				SD1_nCD	I	MFP3	SD卡接口1 卡侦测输入脚
				CLKO	O	MFP14	时钟输出
				EADC0_ST	I	MFP15	EADC0 外部触发输入.
			84	PD.13	I/O	MFP0	GPIO.
				EBI_AD10	I/O	MFP2	EBI 地址/数据总线位10.
				SD0_nCD	I	MFP3	SD卡接口0 卡侦测输入脚
				SPI0_I2SMCLK	I/O	MFP4	SPI0 I2S 主机时钟输出脚
				SPI1_I2SMCLK	I/O	MFP5	SPI1 I2S 主机时钟输出脚
				SC2_nCD	I	MFP7	智能卡 2 卡侦测脚.
			85	PA.12	I/O	MFP0	GPIO.
				I2S0_BCLK	O	MFP2	I2S0 位 时钟输出脚.
				UART4_TXD	O	MFP3	UART4 数据发送脚.
				I2C1_SCL	I/O	MFP4	I2C1 时钟脚.
				SPI2_SS	I/O	MFP5	SPI2 从机选择脚.
				CAN0_TXD	O	MFP6	CAN0 总线发送输出.
				SC2_PWR	O	MFP7	智能卡 2 电源脚.
				BPWM1_CH2	I/O	MFP11	BPWM1 通道2 输出/捕获输入.
				QE11_INDEX	I	MFP12	正交编码器1索引输入
			86	USB_VBUS	P	MFP14	来自USB主机或HUB的电源
				PA.13	I/O	MFP0	GPIO.
				I2S0_MCLK	O	MFP2	I2S0 主机时钟输出脚.
				UART4_RXD	I	MFP3	UART4 数据接收脚.
				I2C1_SDA	I/O	MFP4	I2C1 数据输入/输出引脚.
				SPI2_CLK	I/O	MFP5	SPI2 串行时钟引脚.
				CAN0_RXD	I	MFP6	CAN0 总线接收输入
				SC2_RST	O	MFP7	智能卡 2 复位脚.
				BPWM1_CH3	I/O	MFP11	BPWM1 通道3 输出/捕获输入.
				QE11_A	I	MFP12	正交编码1 计数信号 A 相输入

32 脚	48 脚	64 脚	128 脚	引脚名	类型	MFP值	描述
				USB_D-	A	MFP14	USB 差分信号D+.
23	35	87	PA.14 I2S0_DI UART0_TXD SPI2_MISO I2C2_SCL SC2_DAT BPWM1_CH4 QE11_B USB_D+	I/O	MFP0	GPIO.	
				I	MFP2	I2S0 数据输入脚.	
				O	MFP3	UART0 数据发送脚.	
				I/O	MFP5	SPI2 MISO (主机输入, 从机输出) 脚.	
				I/O	MFP6	I2C2 时钟脚.	
				I/O	MFP7	智能卡2 数据脚.	
				I/O	MFP11	BPWM1 通道4 输出/捕获输入.	
				I	MFP12	正交编码1 计数信号 B 相输入	
				A	MFP14	USB 差分信号D-.	
24	36	88	PA.15 I2S0_DO UART0_RXD SPI2_MOSI I2C2_SDA SC2_CLK BPWM1_CH5 EPWM0_SYNC_IN USB_OTG_ID	I/O	MFP0	GPIO.	
				O	MFP2	I2S0 数据输出脚	
				I	MFP3	UART0 数据接收脚.	
				I/O	MFP5	SPI2 MOSI (主机输出, 从机输入) 脚.	
				I/O	MFP6	I2C2 数据输入/输出引脚.	
				O	MFP7	智能卡 2 时钟脚.	
				I/O	MFP11	BPWM1 通道5 输出/捕获输入.	
				I	MFP12	EPWM0 计数器同步触发输入脚	
				I	MFP14	USB_ID.	
	41	89	HSUSB_VRES	A	MFP0	HSUSB 模块参考电阻	
	42	90	HSUSB_VDD33	P	MFP0	HSUSB VDD33电源	
	43	91	HSUSB_VBUS	P	MFP0	HSUSB 来自USB主机或HUB的电源	
	44	92	HSUSB_D-	A	MFP0	HSUSB 差分信号D+.	
	45	93	HSUSB_VSS	P	MFP0	HSUSB地	
	46	94	HSUSB_D+	A	MFP0	HSUSB 差分信号D-.	
	47	95	HSUSB_VDD12_CAP	A	MFP0	HSUSB 内部电压管理器输出1.2V 去耦脚. 注意: 该脚需要接一个1uF 电容.	
	48	96	HSUSB_ID	I	MFP0	HSUSB ID.	
		97	PE.7	I/O	MFP0	GPIO.	
			SD0_CMD	I/O	MFP3	SD卡接口0 命令/应答脚	

32脚	48脚	64脚	128脚	引脚名	类型	MFP值	描述
				SPI_M_D2	I/O	MFP4	SPI_M 四线模式数据2 .
				UART5_TXD	O	MFP8	UART5 数据发送脚.
				CAN1_TXD	O	MFP9	CAN1 总线发送输出.
				QEI1_INDEX	I	MFP11	正交编码器1索引输入
				EPWM0_CH0	I/O	MFP12	EPWM0 通道0 输出/捕获输入脚.
				BPWM0_CH5	I/O	MFP13	BPWM0 通道5 输出/捕获输入.
			98	PE.6	I/O	MFP0	GPIO.
				SD0_CLK	O	MFP3	SD卡接口0 时钟输出脚
				SPI_M_D3	I/O	MFP4	SPI_M 四线模式数据3 .
				SPI3_I2SMCLK	I/O	MFP5	SPI3 I2S 主机时钟输出脚
				SC0_nCD	I	MFP6	智能卡 0 卡侦测脚.
				USCI0_CTL0	I/O	MFP7	USCI0 控制0 脚..
				UART5_RXD	I	MFP8	UART5 数据接收脚.
				CAN1_RXD	I	MFP9	CAN1 总线接收输入
				QEI1_A	I	MFP11	正交编码1 计数信号 A 相输入
				EPWM0_CH1	I/O	MFP12	EPWM0 通道1 输出/捕获输入脚.
			99	BPWM0_CH4	I/O	MFP13	BPWM0 通道4 输出/捕获输入.
				PE.5	I/O	MFP0	GPIO.
				EBI_nRD	O	MFP2	EBI 读使能输出脚.
				SD0_DAT3	I/O	MFP3	SD卡接口0 数据位3.
				SPI_M_SS	I/O	MFP4	SPI_M 从机选择脚.
				SPI3_SS	I/O	MFP5	SPI3 从机选择脚.
				SC0_PWR	O	MFP6	智能卡 0 电源脚.
				USCI0_CTL1	I/O	MFP7	USCI0 控制1 脚..
				QEI1_B	I	MFP11	正交编码1 计数信号 B 相输入
				EPWM0_CH2	I/O	MFP12	EPWM0 通道2 输出/捕获输入脚.
			100	BPWM0_CH3	I/O	MFP13	BPWM0 通道3 输出/捕获输入.
				PE.4	I/O	MFP0	GPIO.
				EBI_nWR	O	MFP2	EBI 写使能输出脚.

32脚	48脚	64脚	128脚	引脚名	类型	MFP值	描述
				SD0_DAT2	I/O	MFP3	SD卡接口0 数据位2.
				SPIM_CLK	I/O	MFP4	SPIM 串行时钟引脚.
				SPI3_CLK	I/O	MFP5	SPI3 串行时钟引脚.
				SC0_RST	O	MFP6	智能卡0 复位脚.
				USCI0_DAT1	I/O	MFP7	USCI0 数据1脚.
				QEIO_INDEX	I	MFP11	正交编码器0索引输入
				EPWM0_CH3	I/O	MFP12	EPWM0 通道3 输出/捕获输入脚.
				BPWM0_CH2	I/O	MFP13	BPWM0 通道2 输出/捕获输入.
		101		PE.3	I/O	MFP0	GPIO.
				EBI_MCLK	O	MFP2	EBI外部时钟输出脚.
				SD0_DAT1	I/O	MFP3	SD卡接口0 数据位1.
				SPIM_MISO	I/O	MFP4	SPIM MISO (主机输入, 从机输出) 脚..
				SPI3_MISO	I/O	MFP5	SPI3 MISO (主机输入, 从机输出) 脚..
				SC0_DAT	I/O	MFP6	智能卡0 数据脚.
				USCI0_DAT0	I/O	MFP7	USCI0 数据0脚.
				QEIO_A	I	MFP11	正交编码0 计数信号A 相输入
				EPWM0_CH4	I/O	MFP12	EPWM0 通道4 输出/捕获输入脚.
				BPWM0_CH1	I/O	MFP13	BPWM0 通道1 输出/捕获输入.
		102		PE.2	I/O	MFP0	GPIO.
				EBI_ALE	O	MFP2	EBI 地址锁存使能输出脚.
				SD0_DAT0	I/O	MFP3	SD卡接口0 数据位0.
				SPIM_MOSI	I/O	MFP4	SPIM MOSI (主机输出, 从机输入) 脚.
				SPI3_MOSI	I/O	MFP5	SPI3 MOSI (主机输出, 从机输入) 脚.
				SC0_CLK	O	MFP6	智能卡0 时钟脚.
				USCI0_CLK	I/O	MFP7	USCI0 时钟脚.
				QEIO_B	I	MFP11	正交编码0 计数信号B 相输入
				EPWM0_CH5	I/O	MFP12	EPWM0 通道5 输出/捕获输入脚.
				BPWM0_CH0	I/O	MFP13	BPWM0 通道0 输出/捕获输入.
			103	VSS	P	MFP0	数字电路地.

32 脚	48 脚	64 脚	128 脚	引脚名	类型	MFP值	描述
			104	VDD	P	MFP0	I/O 端口电源和LDO 电源用于内部 PLL 和数字电路
			105	PE.1	I/O	MFP0	GPIO.
				EBI_AD10	I/O	MFP2	EBI 地址/数据总线位10.
				QSPI0_MISO0	I/O	MFP3	Quad SPI0 MISO0 (主机输入, 从机输出) 脚..
				SC2_DAT	I/O	MFP4	智能卡2 数据脚.
				I2S0_BCLK	O	MFP5	I2S0 位 时钟输出脚.
				SPI1_MISO	I/O	MFP6	SPI1 MISO (主机输入, 从机输出) 脚.
				UART3_TXD	O	MFP7	UART3 数据发送脚.
				I2C1_SCL	I/O	MFP8	I2C1 时钟脚.
				UART4_nCTS	I	MFP9	UART4 清除发送输入脚.
			106	PE.0	I/O	MFP0	GPIO.
				EBI_AD11	I/O	MFP2	EBI 地址/数据总线位11.
				QSPI0_MOSI0	I/O	MFP3	Quad SPI0 MOSI0 (主机输出, 从机输入) 脚.
				SC2_CLK	O	MFP4	智能卡 2 时钟脚.
				I2S0_MCLK	O	MFP5	I2S0 主机时钟输出脚.
				SPI1_MOSI	I/O	MFP6	SPI1 MOSI (主机输出, 从机输入) 脚.
				UART3_RXD	I	MFP7	UART3 数据接收脚.
				I2C1_SDA	I/O	MFP8	I2C1 数据输入/输出引脚.
				UART4_nRTS	O	MFP9	UART4 请求发送输出脚.
			107	PH.8	I/O	MFP0	GPIO.
				EBI_AD12	I/O	MFP2	EBI 地址/数据总线位12.
				QSPI0_CLK	I/O	MFP3	Quad SPI0 串行时钟引脚.
				SC2_PWR	O	MFP4	智能卡 2 电源脚.
				I2S0_DI	I	MFP5	I2S0 数据输入脚.
				SPI1_CLK	I/O	MFP6	SPI1 串行时钟引脚.
				UART3_nRTS	O	MFP7	UART3 请求发送输出脚.
				I2C1_SMBAL	O	MFP8	I2C1 SMBus SMBALTER 脚
				I2C2_SCL	I/O	MFP9	I2C2 时钟脚.
				UART1_TXD	O	MFP10	UART1 数据发送脚.

32 脚	48 脚	64 脚	128 脚	引脚名	类型	MFP值	描述
			108	PH.9	I/O	MFP0	GPIO.
				EBI_AD13	I/O	MFP2	EBI 地址/数据总线位13.
				QSPI0_SS	I/O	MFP3	Quad SPI0 从机选择脚.
				SC2_RST	O	MFP4	智能卡 2 复位脚.
				I2S0_DO	O	MFP5	I2S0 数据输出脚
				SPI1_SS	I/O	MFP6	SPI1 从机选择脚.
				UART3_nCTS	I	MFP7	UART3 清除发送输入脚.
				I2C1_SMBSUS	O	MFP8	I2C1 SMBus SMBSUS 脚(PMBus 控制脚)
				I2C2_SDA	I/O	MFP9	I2C2 数据输入/输出引脚.
				UART1_RXD	I	MFP10	UART1 数据接收脚.
			109	PH.10	I/O	MFP0	GPIO.
				EBI_AD14	I/O	MFP2	EBI 地址/数据总线位14.
				QSPI0_MISO1	I/O	MFP3	Quad SPI0 MISO1 (主机输入, 从机输出) 脚..
				SC2_nCD	I	MFP4	智能卡 2 卡侦测脚.
				I2S0_LRCK	O	MFP5	I2S0 左右声道选择时钟输出脚.
				SPI1_I2SMCLK	I/O	MFP6	SPI1 I2S 主机时钟输出脚
				UART4_TXD	O	MFP7	UART4 数据发送脚.
				UART0_TXD	O	MFP8	UART0 数据发送脚.
			110	PH.11	I/O	MFP0	GPIO.
				EBI_AD15	I/O	MFP2	EBI 地址/数据总线位15.
				QSPI0_MOSI1	I/O	MFP3	Quad SPI0 MOSI1 (主机输出, 从机输入) 脚.
				UART4_RXD	I	MFP7	UART4 数据接收脚.
				UART0_RXD	I	MFP8	UART0 数据接收脚.
				EPWM0_CH5	I/O	MFP11	EPWM0 通道5 输出/捕获输入脚.
			111	PD.14	I/O	MFP0	GPIO.
				EBI_nCS0	O	MFP2	EBI 片选 0 输出脚.
				SPI3_I2SMCLK	I/O	MFP3	SPI3 I2S 主机时钟输出脚
				SC1_nCD	I	MFP4	智能卡 1 卡侦测脚.
				EPWM0_CH4	I/O	MFP11	EPWM0 通道4 输出/捕获输入脚.

32 脚	48 脚	64 脚	128 脚	引脚名	类型	MFP值	描述
25	37	49	112	VSS	P	MFP0	数字电路地.
26	38	50	113	LDO_CAP	A	MFP0	LDO 输出脚.
27	39	51	114	VDD	P	MFP0	I/O 端口电源和LDO 电源用于内部 PLL 和数字电路
28	41	53	116	PC.14	I/O	MFP0	GPIO.
				EBI_AD11	I/O	MFP2	EBI 地址/数据总线位11.
				SC1_nCD	I	MFP3	智能卡 1 卡侦测脚.
				SPI0_I2SMCLK	I/O	MFP4	SPI0 I2S 主机时钟输出脚
				USCI0_CTL0	I/O	MFP5	USCI0 控制0 脚..
				QSPI0_CLK	I/O	MFP6	Quad SPI0 串行时钟引脚.
				EPWM0_SYNC_IN	I	MFP11	EPWM0 计数器同步触发输入脚
				TM1	I/O	MFP13	定时器1事件计数器输入/反转输出脚.
				USB_VBUS_ST	I	MFP14	USB 外部 VBUS 管理器状态脚.
				HSUSB_VBUS_ST	I	MFP15	HSUSB 外部 VBUS 管理器状态脚.
29	42	54	117	PB.15	I/O	MFP0	GPIO.
				EADC0_CH15	A	MFP1	EADC0 通道15模拟输入.
				EBI_AD12	I/O	MFP2	EBI 地址/数据总线位12.
				SC1_PWR	O	MFP3	智能卡 1 电源脚.
				SPI0_SS	I/O	MFP4	SPI0 从机选择脚.
				USCI0_CTL1	I/O	MFP5	USCI0 控制1 脚..
				UART0_nCTS	I	MFP6	UART0 清除发送输入脚.
				UART3_TXD	O	MFP7	UART3 数据发送脚.
				I2C2_SMBAL	O	MFP8	I2C2 SMBus SMBALTER 脚
				EPWM1_CHO	I/O	MFP11	EPWM1 通道0 输出/捕获输入脚.
				TM0_EXT	I/O	MFP13	定时器0事件计数器输入/反转输出脚.
				USB_VBUS_EN	O	MFP14	USB 外部 VBUS 管理器使能脚.
				HSUSB_VBUS_EN	O	MFP15	HSUSB 外部 VBUS 管理器使能脚.

32 脚	48 脚	64 脚	128 脚	引脚名	类型	MFP值	描述
				SC1_RST	O	MFP3	智能卡 1 复位脚.
				SPI0_CLK	I/O	MFP4	SPI0 串行时钟引脚.
				USCI0_DAT1	I/O	MFP5	USCI0 数据1脚.
				UART0_nRTS	O	MFP6	UART0 请求发送输出脚.
				UART3_RXD	I	MFP7	UART3 数据接收脚.
				I2C2_SMBSUS	O	MFP8	I2C2 SMBus SMBSUS 脚(PMBus 控制脚)
				EPWM1_CH1	I/O	MFP11	EPWM1 通道1 输出/捕获输入脚.
				TM1_EXT	I/O	MFP13	定时器1事件计数器输入/反转输出脚.
				CLKO	O	MFP14	时钟输出
				PB.13	I/O	MFP0	GPIO.
				EADC0_CH13	A	MFP1	EADC0 通道13模拟输入.
				DAC1_OUT	A	MFP1	DAC1 通道模拟输出
				ACMP0_P3	A	MFP1	模拟比较器0 正输入 3 脚.
				ACMP1_P3	A	MFP1	模拟比较器1 正输入 3 脚.
				EBI_AD14	I/O	MFP2	EBI 地址/数据总线位14.
				SC1_DAT	I/O	MFP3	智能卡1 数据脚.
				SPI0_MISO	I/O	MFP4	SPI0 MISO (主机输入, 从机输出) 脚.
				USCI0_DAT0	I/O	MFP5	USCI0 数据0脚.
				UART0_TXD	O	MFP6	UART0 数据发送脚.
				UART3_nRTS	O	MFP7	UART3 请求发送输出脚.
				I2C2_SCL	I/O	MFP8	I2C2 时钟脚.
				EPWM1_CH2	I/O	MFP11	EPWM1 通道2 输出/捕获输入脚.
				TM2_EXT	I/O	MFP13	定时器2事件计数器输入/反转输出脚.
				PB.12	I/O	MFP0	GPIO.
				EADC0_CH12	A	MFP1	EADC0 通道12模拟输入.
				DAC0_OUT	A	MFP1	DAC0 通道模拟输出
				ACMP0_P2	A	MFP1	模拟比较器0 正输入 2 脚.
				ACMP1_P2	A	MFP1	模拟比较器1 正输入 2 脚.
				EBI_AD15	I/O	MFP2	EBI 地址/数据总线位15.

32 脚	48 脚	64 脚	128 脚	引脚名	类型	MFP值	描述
				SC1_CLK	O	MFP3	智能卡 1 时钟脚.
				SPI0_MOSI	I/O	MFP4	SPI0 MOSI (主机输出, 从机输入) 脚.
				USCI0_CLK	I/O	MFP5	USCI0 时钟脚.
				UART0_RXD	I	MFP6	UART0 数据接收脚.
				UART3_nCTS	I	MFP7	UART3 清除发送输入脚.
				I2C2_SDA	I/O	MFP8	I2C2 数据输入/输出引脚.
				SD0_nCD	I	MFP9	SD卡接口0 卡侦测输入脚
				EPWM1_CH3	I/O	MFP11	EPWM1 通道3 输出/捕获输入脚.
				TM3_EXT	I/O	MFP13	定时器3事件计数器输入/反转输出脚.
32	45	57	120	AVDD	P	MFP0	内部模拟电路电源 .
		58	121	VREF	A	MFP0	ADC 参考电压输入. 注意: 该脚需要接一个1uF 电容.
	46	59	122	AVSS	P	MFP0	模拟电路地
				PB.11	I/O	MFP0	GPIO.
				EADC0_CH11	A	MFP1	EADC0 通道11模拟输入.
				EBI_ADR16	O	MFP2	EBI 地址总线位16.
				UART0_nCTS	I	MFP5	UART0 清除发送输入脚.
				UART4_TXD	O	MFP6	UART4 数据发送脚.
				I2C1_SCL	I/O	MFP7	I2C1 时钟脚.
				CAN0_TXD	O	MFP8	CAN0 总线发送输出.
				SPI0_I2SMCLK	I/O	MFP9	SPI0 I2S 主机时钟输出脚
				BPWM1_CH0	I/O	MFP10	BPWM1 通道0 输出/捕获输入.
				SPI3_CLK	I/O	MFP11	SPI3 串行时钟引脚.
				HSUSB_VBUS_ST	I	MFP14	HSUSB 外部 VBUS 管理器状态脚.
				PB.10	I/O	MFP0	GPIO.
				EADC0_CH10	A	MFP1	EADC0 通道10模拟输入.
				EBI_ADR17	O	MFP2	EBI 地址总线位17.
				USCI1_CTL0	I/O	MFP4	USCI1 控制0 脚..
				UART0_nRTS	O	MFP5	UART0 请求发送输出脚.

32 脚	48 脚	64 脚	128 脚	引脚名	类型	MFP值	描述
				UART4_RXD	I	MFP6	UART4 数据接收脚.
				I2C1_SDA	I/O	MFP7	I2C1 数据输入/输出引脚.
				CAN0_RXD	I	MFP8	CAN0 总线接收输入
				BPWM1_CH1	I/O	MFP10	BPWM1 通道1 输出/捕获输入.
				SPI3_SS	I/O	MFP11	SPI3 从机选择脚.
				HSUSB_VBUS_EN	O	MFP14	HSUSB 外部 VBUS 管理器使能脚.
		62	125	PB.9	I/O	MFP0	GPIO.
				EADC0_CH9	A	MFP1	EADC0 通道9模拟输入.
				EBI_ADR18	O	MFP2	EBI 地址总线位18.
				USCI1_CTL1	I/O	MFP4	USCI1 控制1 脚..
				UART0_TXD	O	MFP5	UART0 数据发送脚.
				UART1_nCTS	I	MFP6	UART1 清除发送输入脚.
				I2C1_SMBAL	O	MFP7	I2C1 SMBus SMBALTER 脚
				BPWM1_CH2	I/O	MFP10	BPWM1 通道2 输出/捕获输入.
				SPI3_MISO	I/O	MFP11	SPI3 MISO (主机输入, 从机输出) 脚..
				INT7	I	MFP13	外部中断7输入脚.
		63	126	PB.8	I/O	MFP0	GPIO.
				EADC0_CH8	A	MFP1	EADC0 通道8模拟输入.
				EBI_ADR19	O	MFP2	EBI 地址总线位19.
				USCI1_CLK	I/O	MFP4	USCI1 时钟脚.
				UART0_RXD	I	MFP5	UART0 数据接收脚.
				UART1_nRTS	O	MFP6	UART1 请求发送输出脚.
				I2C1_SMBSUS	O	MFP7	I2C1 SMBus SMBSUS 脚(PMBus 控制脚)
				BPWM1_CH3	I/O	MFP10	BPWM1 通道3 输出/捕获输入.
				SPI3_MOSI	I/O	MFP11	SPI3 MOSI (主机输出, 从机输入) 脚.
				INT6	I	MFP13	外部中断6输入脚.
	47	64	127	PB.7	I/O	MFP0	GPIO.
				EADC0_CH7	A	MFP1	EADC0 通道7模拟输入.
				EBI_nWRL	O	MFP2	EBI 低字节写使能输出脚.

32 脚	48 脚	64 脚	128 脚	引脚名	类型	MFP值	描述
				USCI1_DAT0	I/O	MFP4	USCI1 数据0脚.
				CAN1_TXD	O	MFP5	CAN1 总线发送输出.
				UART1_TXD	O	MFP6	UART1 数据发送脚.
				SD1_CMD	I/O	MFP7	SD卡接口1 命令/应答脚
				EBI_nCS0	O	MFP8	EBI 片选 0 输出脚.
				BPWM1_CH4	I/O	MFP10	BPWM1 通道4 输出/捕获输入.
				EPWM1_BRAKE0	I	MFP11	EPWM1 刹车0输入脚.
				EPWM1_CH4	I/O	MFP12	EPWM1 通道4 输出/捕获输入脚.
				INT5	I	MFP13	外部中断5输入脚.
				USB_VBUS_ST	I	MFP14	USB 外部 VBUS 管理器状态脚.
				ACMP0_O	O	MFP15	模拟比较器0 输出脚.
		1	128	PB.6	I/O	MFP0	GPIO.
				EADC0_CH6	A	MFP1	EADC0 通道6模拟输入.
				EBI_nWRH	O	MFP2	EBI 高字节写使能输出脚
				USCI1_DAT1	I/O	MFP4	USCI1 数据1脚.
				CAN1_RXD	I	MFP5	CAN1 总线接收输入
				UART1_RXD	I	MFP6	UART1 数据接收脚.
				SD1_CLK	O	MFP7	SD卡接口1 时钟输出脚
				EBI_nCS1	O	MFP8	EBI 片选 1 输出脚.
				BPWM1_CH5	I/O	MFP10	BPWM1 通道5 输出/捕获输入.
				EPWM1_BRAKE1	I	MFP11	EPWM1 刹车1输入脚.
				EPWM1_CH5	I/O	MFP12	EPWM1 通道5 输出/捕获输入脚.
				INT4	I	MFP13	外部中断4输入脚.
				USB_VBUS_EN	O	MFP14	USB 外部 VBUS 管理器使能脚.
				ACMP1_O	O	MFP15	模拟比较器1 输出脚.

## 4.2.6 M487 系列引脚描述

64 脚	128 脚	144 脚	引脚名称	类型	MFP	描述
2	1	1	PB.5	I/O	MFP0	GPIO.
			EADC0_CH5	A	MFP1	EADC0 通道5模拟输入.
			ACMP1_N	A	MFP1	模拟比较器1 负输入脚.
			EBI_ADR0	O	MFP2	EBI 地址总线位0.
			SD0_DAT3	I/O	MFP3	SD卡接口0 数据位3.
			EMAC_RMII_REFCLK	I	MFP4	EMAC RMII 参考时钟输入脚.
			SPI1_MISO	I/O	MFP5	SPI1 MISO (主机输入, 从机输出) 脚.
			I2C0_SCL	I/O	MFP6	I2C0 时钟脚.
			UART5_TXD	O	MFP7	UART5 数据发送脚.
			USCI1_CTL0	I/O	MFP8	USCI1 控制0 脚..
			SC0_CLK	O	MFP9	智能卡0 时钟脚.
			I2S0_BCLK	O	MFP10	I2S0 位时钟输出脚.
			EPWM0_CH0	I/O	MFP11	EPWM0 通道0 输出/捕获输入脚.
			TM0	I/O	MFP14	定时器0事件计数器输入/反转输出脚.
			INT0	I	MFP15	外部中断0输入脚.
3	2	2	PB.4	I/O	MFP0	GPIO.
			EADC0_CH4	A	MFP1	EADC0 通道4模拟输入.
			ACMP1_P1	A	MFP1	模拟比较器1 正输入 1 脚.
			EBI_ADR1	O	MFP2	EBI 地址总线位1.
			SD0_DAT2	I/O	MFP3	SD卡接口0 数据位2.
			EMAC_RMII_RXD0	I	MFP4	EMAC RMII 接收数据总线位 0.
			SPI1_MOSI	I/O	MFP5	SPI1 MOSI (主机输出, 从机输入) 脚.
			I2C0_SDA	I/O	MFP6	I2C0 数据输入/输出引脚.
			UART5_RXD	I	MFP7	UART5 数据接收脚.
			USCI1_CTL1	I/O	MFP8	USCI1 控制1 脚..
			SC0_DAT	I/O	MFP9	智能卡0 数据脚.
			I2S0_MCLK	O	MFP10	I2S0 主机时钟输出脚.
			EPWM0_CH1	I/O	MFP11	EPWM0 通道1 输出/捕获输入脚.

64 脚	128 脚	144 脚	引脚名称	类型	MFP	描述
			TM1	I/O	MFP14	定时器1事件计数器输入/反转输出脚.
			INT1	I	MFP15	外部中断1输入脚.
4	3	3	PB.3	I/O	MFP0	GPIO.
			EADC0_CH3	A	MFP1	EADC0 通道3模拟输入.
			ACMP0_N	A	MFP1	模拟比较器0 负输入脚.
			EBI_ADR2	O	MFP2	EBI 地址总线位2.
			SD0_DAT1	I/O	MFP3	SD卡接口0 数据位1.
			EMAC_RMII_RXD1	I	MFP4	EMAC RMII 接收数据总线位 1.
			SPI1_CLK	I/O	MFP5	SPI1 串行时钟引脚.
			UART1_TXD	O	MFP6	UART1 数据发送脚.
			UART5_nRTS	O	MFP7	UART5 请求发送输出脚.
			USCI1_DAT1	I/O	MFP8	USCI1 数据1脚.
			SC0_RST	O	MFP9	智能卡 0 复位脚.
			I2S0_DI	I	MFP10	I2S0 数据输入脚.
			EPWM0_CH2	I/O	MFP11	EPWM0 通道2 输出/捕获输入脚.
			TM2	I/O	MFP14	定时器2事件计数器输入/反转输出脚.
			INT2	I	MFP15	外部中断2输入脚.
5	4	4	PB.2	I/O	MFP0	GPIO.
			EADC0_CH2	A	MFP1	EADC0 通道2模拟输入.
			ACMP0_P1	A	MFP1	模拟比较器0 正输入 1 脚.
			OPA0_O	A	MFP1	运放 0 输出脚.
			EBI_ADR3	O	MFP2	EBI 地址总线位3.
			SD0_DAT0	I/O	MFP3	SD卡接口0 数据位0.
			EMAC_RMII_CRSDV	I	MFP4	EMAC RMII 载波侦听/接收数据输入脚.
			SPI1_SS	I/O	MFP5	SPI1 从机选择脚.
			UART1_RXD	I	MFP6	UART1 数据接收脚.
			UART5_nCTS	I	MFP7	UART5 清除发送输入脚.
			USCI1_DAT0	I/O	MFP8	USCI1 数据0脚.
			SC0_PWR	O	MFP9	智能卡 0 电源脚.

<b>64 脚</b>	<b>128 脚</b>	<b>144 脚</b>	引脚名称	类型	MFP	描述
			I2S0_DO	O	MFP10	I2S0 数据输出脚.
			EPWM0_CH3	I/O	MFP11	EPWM0 通道3 输出/捕获输入脚.
			TM3	I/O	MFP14	定时器3事件计数器输入/反转输出脚.
			INT3	I	MFP15	外部中断3输入脚.
			PC.12	I/O	MFP0	GPIO.
			EBI_ADR4	O	MFP2	EBI 地址总线位4.
			UART0_TXD	O	MFP3	UART0 数据发送脚.
			I2C0_SCL	I/O	MFP4	I2C0 时钟脚.
			SPI3_MISO	I/O	MFP6	SPI3 MISO (主机输入, 从机输出) 脚..
			SC0_nCD	I	MFP9	智能卡 0 卡侦测脚.
			ECAP1_IC2	I	MFP11	增强型捕获输入单元1输入脚2
			EPWM1_CH0	I/O	MFP12	EPWM1 通道0 输出/捕获输入脚.
			ACMP0_O	O	MFP14	模拟比较器0 输出脚.
			PC.11	I/O	MFP0	GPIO.
			EBI_ADR5	O	MFP2	EBI 地址总线位5.
			UART0_RXD	I	MFP3	UART0 数据接收脚.
			I2C0_SDA	I/O	MFP4	I2C0 数据输入/输出引脚.
			SPI3_MOSI	I/O	MFP6	SPI3 MOSI (主机输出, 从机输入) 脚.
			ECAP1_IC1	I	MFP11	增强型捕获输入单元1输入脚1
			EPWM1_CH1	I/O	MFP12	EPWM1 通道1 输出/捕获输入脚.
			ACMP1_O	O	MFP14	模拟比较器1 输出脚.
			PC.10	I/O	MFP0	GPIO.
			EBI_ADR6	O	MFP2	EBI 地址总线位 6.
			SPI3_CLK	I/O	MFP6	SPI3 串行时钟引脚.
			UART3_TXD	O	MFP7	UART3 数据发送脚.
			CAN1_TXD	O	MFP9	CAN1 总线发送输出.
			ECAP1_IC0	I	MFP11	增强型捕获输入单元1输入脚0
			EPWM1_CH2	I/O	MFP12	EPWM1 通道2 输出/捕获输入脚.
	8	8	PC.9	I/O	MFP0	GPIO.

64 脚	128 脚	144 脚	引脚名称	类型	MFP	描述
			EBI_ADR7	O	MFP2	EBI 地址总线位 7.
			SPI3_SS	I/O	MFP6	SPI3 从机选择脚.
			UART3_RXD	I	MFP7	UART3 数据接收脚.
			CAN1_RXD	I	MFP9	CAN1 总线接收输入
			EPWM1_CH3	I/O	MFP12	EPWM1 通道3 输出/捕获输入脚.
			PB.1	I/O	MFP0	GPIO.
			EADC0_CH1	A	MFP1	EADC0 通道1模拟输入.
			OPA0_N	A	MFP1	运放0 负输入脚.
			EBI_ADR8	O	MFP2	EBI 地址总线位 8.
			SD0_CLK	O	MFP3	SD卡接口0 时钟输出脚
			EMAC_RMII_RXERR	I	MFP4	EMAC RMII接收数据错误输入脚.
			SPI1_I2SMCLK	I/O	MFP5	SPI1 I2S 主机时钟输出脚
			SPI3_I2SMCLK	I/O	MFP6	SPI3 I2S 主机时钟输出脚
			UART2_TXD	O	MFP7	UART2 数据发送脚.
			USCI1_CLK	I/O	MFP8	USCI1 时钟脚.
			I2C1_SCL	I/O	MFP9	I2C1 时钟脚.
			I2S0_LRCK	O	MFP10	I2S0 左右声道选择时钟输出脚.
			EPWM0_CH4	I/O	MFP11	EPWM0 通道4 输出/捕获输入脚.
			EPWM1_CH4	I/O	MFP12	EPWM1 通道4 输出/捕获输入脚.
			EPWM0_BRAKE0	I	MFP13	EPWM0 刹车0输入脚.
			PB.0	I/O	MFP0	GPIO.
			EADC0_CH0	A	MFP1	EADC0 通道0模拟输入.
			OPA0_P	A	MFP1	运放0 正输入脚.
			EBI_ADR9	O	MFP2	EBI 地址总线位 9.
			SD0_CMD	I/O	MFP3	SD卡接口0 命令/应答脚
			UART2_RXD	I	MFP7	UART2 数据接收脚.
			SPI0_I2SMCLK	I/O	MFP8	SPI0 I2S 主机时钟输出脚
			I2C1_SDA	I/O	MFP9	I2C1 数据输入/输出引脚.
			EPWM0_CH5	I/O	MFP11	EPWM0 通道5 输出/捕获输入脚.

<b>64 脚</b>	<b>128 脚</b>	<b>144 脚</b>	引脚名称	类型	MFP	描述
			EPWM1_CH5	I/O	MFP12	EPWM1 通道5 输出/捕获输入脚.
			EPWM0_BRAKE1	I	MFP13	EPWM0 刹车1输入脚.
	11	11	V <sub>SS</sub>	P	MFP0	数字电路地.
	12	12	V <sub>DD</sub>	P	MFP0	I/O 端口电源和LDO 电源用于内部 PLL 和数字电路
8	13	13	PA.11	I/O	MFP0	GPIO.
			ACMP0_P0	A	MFP1	模拟比较器0 正输入 0 脚.
			EBI_nRD	O	MFP2	EBI 读使能输出脚.
			SC2_PWR	O	MFP3	智能卡 2 电源脚.
			SPI2_SS	I/O	MFP4	SPI2 从机选择脚.
			SD1_DAT3	I/O	MFP5	SD卡接口1 数据位3.
			USCI0_CLK	I/O	MFP6	USCI0 时钟脚.
			I2C2_SCL	I/O	MFP7	I2C2 时钟脚.
			BPWM0_CH0	I/O	MFP9	BPWM0 通道0 输出/捕获输入.
			EPWM0_SYNC_OUT	O	MFP10	EPWM0 计数器同步触发输出脚.
			TM0_EXT	I/O	MFP13	定时器0事件计数器输入/反转输出脚.
			DAC1_ST	I	MFP14	DAC1 外部触发输入.
			PA.10	I/O	MFP0	GPIO.
9	14	14	ACMP1_P0	A	MFP1	模拟比较器1 正输入 0 脚.
			OPA1_O	A	MFP1	运放 1 输出脚.
			EBI_nWR	O	MFP2	EBI 写使能输出脚.
			SC2_RST	O	MFP3	智能卡 2 复位脚.
			SPI2_CLK	I/O	MFP4	SPI2 串行时钟引脚.
			SD1_DAT2	I/O	MFP5	SD卡接口1 数据位2.
			USCI0_DAT0	I/O	MFP6	USCI0 数据0脚.
			I2C2_SDA	I/O	MFP7	I2C2 数据输入/输出引脚.
			BPWM0_CH1	I/O	MFP9	BPWM0 通道1 输出/捕获输入.
			QE1_INDEX	I	MFP10	正交编码器1索引输入
			ECAP0_IC0	I	MFP11	增强型捕获输入单元0输入脚0
			TM1_EXT	I/O	MFP13	定时器1事件计数器输入/反转输出脚.

<b>64 脚</b>	<b>128 脚</b>	<b>144 脚</b>	引脚名称	类型	MFP	描述
			DAC0_ST	I	MFP14	DAC0 外部触发输入.
10	15	15	PA.9	I/O	MFP0	GPIO.
			OPA1_N	A	MFP1	运放1 负输入端.
			EBI_MCLK	O	MFP2	EBI外部时钟输出脚.
			SC2_DAT	I/O	MFP3	智能卡2 数据脚.
			SPI2_MISO	I/O	MFP4	SPI2 MISO (主机输入, 从机输出) 脚.
			SD1_DAT1	I/O	MFP5	SD卡接口1 数据位1.
			USCI0_DAT1	I/O	MFP6	USCI0 数据1脚.
			UART1_TXD	O	MFP7	UART1 数据发送脚.
			BPWM0_CH2	I/O	MFP9	BPWM0 通道2 输出/捕获输入.
			QEI1_A	I	MFP10	正交编码1 计数信号 A 相输入
11	16	16	ECAP0_IC1	I	MFP11	增强型捕获输入单元0输入脚1
			TM2_EXT	I/O	MFP13	定时器2事件计数器输入/反转输出脚.
			PA.8	I/O	MFP0	GPIO.
			OPA1_P	A	MFP1	运放1 正输入脚.
			EBI_ALE	O	MFP2	EBI 地址锁存使能输出脚.
			SC2_CLK	O	MFP3	智能卡 2 时钟脚.
			SPI2_MOSI	I/O	MFP4	SPI2 MOSI (主机输出, 从机输入) 脚.
			SD1_DAT0	I/O	MFP5	SD卡接口1 数据位0.
			USCI0_CTL1	I/O	MFP6	USCI0 控制1 脚..
			UART1_RXD	I	MFP7	UART1 数据接收脚.
			BPWM0_CH3	I/O	MFP9	BPWM0 通道3 输出/捕获输入.
			QEI1_B	I	MFP10	正交编码1 计数信号 B 相输入
			ECAP0_IC2	I	MFP11	增强型捕获输入单元0输入脚2
			TM3_EXT	I/O	MFP13	定时器3事件计数器输入/反转输出脚.
			INT4	I	MFP15	外部中断4输入脚.
			PC.13	I/O	MFP0	GPIO.
			EBI_ADR10	O	MFP2	EBI 地址总线位10.
			SC2_nCD	I	MFP3	智能卡 2 卡侦测脚.

<b>64 脚</b>	<b>128 脚</b>	<b>144 脚</b>	引脚名称	类型	MFP	描述
			SPI2_I2SMCLK	I/O	MFP4	SPI2 I2S 主机时钟输出脚
			CAN1_TXD	O	MFP5	CAN1 总线发送输出.
			USCI0_CTL0	I/O	MFP6	USCI0 控制0 脚..
			UART2_TXD	O	MFP7	UART2 数据发送脚.
			BPWM0_CH4	I/O	MFP9	BPWM0 通道4 输出/捕获输入.
			CLKO	O	MFP13	时钟输出
			EADC0_ST	I	MFP14	EADC0 外部触发输入.
			PD.12	I/O	MFP0	GPIO.
			OPA2_O	A	MFP1	运放 2 输出脚.
			EBI_nCS0	O	MFP2	EBI 片选 0 输出脚.
			CAN1_RXD	I	MFP5	CAN1 总线接收输入
			UART2_RXD	I	MFP7	UART2 数据接收脚.
			BPWM0_CH5	I/O	MFP9	BPWM0 通道5 输出/捕获输入.
			QEIO_INDEX	I	MFP10	正交编码器0索引输入
			CLKO	O	MFP13	时钟输出
			EADC0_ST	I	MFP14	EADC0 外部触发输入.
			INT5	I	MFP15	外部中断5输入脚.
			PD.11	I/O	MFP0	GPIO.
			OPA2_N	A	MFP1	运放 2 负输入脚.
			EBI_nCS1	O	MFP2	EBI 片选 1 输出脚.
			UART1_TXD	O	MFP3	UART1 数据发送脚.
			CAN0_TXD	O	MFP4	CAN0 总线发送输出.
			QEIO_A	I	MFP10	正交编码0 计数信号 A 相输入
			INT6	I	MFP15	外部中断6输入脚.
			PD.10	I/O	MFP0	GPIO.
			OPA2_P	A	MFP1	运放 2 正输入脚.
			EBI_nCS2	O	MFP2	EBI 片选 2 输出脚.
			UART1_RXD	I	MFP3	UART1 数据接收脚.
			CAN0_RXD	I	MFP4	CAN0 总线接收输入

<b>64 脚</b>	<b>128 脚</b>	<b>144 脚</b>	引脚名称	类型	MFP	描述
			QEIO_B	I	MFP10	正交编码0 计数信号 B 相输入
			INT7	I	MFP15	外部中断7输入脚.
		21	V <sub>SS</sub>	P	MFP0	数字电路地.
		22	V <sub>DD</sub>	P	MFP0	I/O 端口电源和LDO 电源用于内部 PLL 和数字电路
			PG.0	I/O	MFP0	GPIO.
			EBI_ADR8	O	MFP2	EBI 地址总线位 8.
			I2C0_SCL	I/O	MFP4	I2C0 时钟脚.
			I2C1_SMBAL	O	MFP5	I2C1 SMBus SMBALTER 脚
			UART2_RXD	I	MFP6	UART2 数据接收脚.
			CAN1_TXD	O	MFP7	CAN1 总线发送输出.
			UART1_TXD	O	MFP8	UART1 数据发送脚.
			PG.1	I/O	MFP0	GPIO.
			EBI_ADR9	O	MFP2	EBI 地址总线位 9.
			SPI2_I2SMCLK	I/O	MFP3	SPI2 I2S 主机时钟输出脚
			I2C0_SDA	I/O	MFP4	I2C0 数据输入/输出引脚.
			I2C1_SMBSUS	O	MFP5	I2C1 SMBus SMBSUS 脚(PMBus 控制脚)
			UART2_TXD	O	MFP6	UART2 数据发送脚.
			CAN1_RXD	I	MFP7	CAN1 总线接收输入
			UART1_RXD	I	MFP8	UART1 数据接收脚.
			PG.2	I/O	MFP0	GPIO.
			EBI_ADR11	O	MFP2	EBI 地址总线位 11.
			SPI2_SS	I/O	MFP3	SPI2 从机选择脚.
			I2C0_SMBAL	O	MFP4	I2C0 SMBus SMBALTER 脚
			I2C1_SCL	I/O	MFP5	I2C1 时钟脚.
			TM0	I/O	MFP13	定时器0事件计数器输入/反转输出脚.
			PG.3	I/O	MFP0	GPIO.
			EBI_ADR12	O	MFP2	EBI 地址总线位 12.
			SPI2_CLK	I/O	MFP3	SPI2 串行时钟引脚.
			I2C0_SMBSUS	O	MFP4	I2C0 SMBus SMBSUS 脚(PMBus 控制脚)

64 脚	128 脚	144 脚	引脚名称	类型	MFP	描述
			I2C1_SDA	I/O	MFP5	I2C1 数据输入/输出引脚.
			TM1	I/O	MFP13	定时器1事件计数器输入/反转输出脚.
	23	27	PG.4	I/O	MFP0	GPIO.
			EBI_ADR13	O	MFP2	EBI 地址总线位13.
			SPI2_MISO	I/O	MFP3	SPI2 MISO (主机输入, 从机输出) 脚.
			TM2	I/O	MFP13	定时器2事件计数器输入/反转输出脚.
	24	28	PF.11	I/O	MFP0	GPIO.
			EBI_ADR14	O	MFP2	EBI 地址总线位14.
			SPI2_MOSI	I/O	MFP3	SPI2 MOSI (主机输出, 从机输入) 脚.
			TAMPER5	I/O	MFP10	TAMPER 倾测循环脚 5.
			TM3	I/O	MFP13	定时器3事件计数器输入/反转输出脚.
	25	29	PF.10	I/O	MFP0	GPIO.
			EBI_ADR15	O	MFP2	EBI 地址总线位15.
			SC0_nCD	I	MFP3	智能卡 0 卡侦测脚.
			I2S0_BCLK	O	MFP4	I2S0 位 时钟输出脚.
			SPI0_I2SMCLK	I/O	MFP5	SPI0 I2S 主机时钟输出脚
			TAMPER4	I/O	MFP10	TAMPER 倾测循环脚 4.
	26	30	PF.9	I/O	MFP0	GPIO.
			EBI_ADR16	O	MFP2	EBI 地址总线位16.
			SC0_PWR	O	MFP3	智能卡 0 电源脚.
			I2S0_MCLK	O	MFP4	I2S0 主机时钟输出脚.
			SPI0_SS	I/O	MFP5	SPI0 从机选择脚.
			TAMPER3	I/O	MFP10	TAMPER 倾测循环脚 3.
	27	31	PF.8	I/O	MFP0	GPIO.
			EBI_ADR17	O	MFP2	EBI 地址总线位17.
			SC0_RST	O	MFP3	智能卡 0 复位脚.
			I2S0_DI	I	MFP4	I2S0 数据输入脚.
			SPI0_CLK	I/O	MFP5	SPI0 串行时钟引脚.
			TAMPER2	I/O	MFP10	TAMPER 倾测循环脚 2.

<b>64 脚</b>	<b>128 脚</b>	<b>144 脚</b>	引脚名称	类型	MFP	描述
28	32	32	PF.7	I/O	MFP0	GPIO.
			EBI_ADR18	O	MFP2	EBI 地址总线位18.
			SC0_DAT	I/O	MFP3	智能卡0 数据脚.
			I2S0_DO	O	MFP4	I2S0 数据输出脚
			SPI0_MISO	I/O	MFP5	SPI0 MISO (主机输入, 从机输出) 脚.
			UART4_TXD	O	MFP6	UART4 数据发送脚.
			TAMPER1	I/O	MFP10	TAMPER 倾测循环脚 1.
12	29	33	PF.6	I/O	MFP0	GPIO.
			EBI_ADR19	O	MFP2	EBI 地址总线位19.
			SC0_CLK	O	MFP3	智能卡0 时钟脚.
			I2S0_LRCK	O	MFP4	I2S0 左右声道选择时钟输出脚.
			SPI0_MOSI	I/O	MFP5	SPI0 MOSI (主机输出, 从机输入) 脚.
			UART4_RXD	I	MFP6	UART4 数据接收脚.
			EBI_nCS0	O	MFP7	EBI 片选0 输出脚.
13	30	34	V <sub>DD</sub>	P	MFP0	I/O 端口电源和LDO 电源用于内部 PLL 和数字电路
			PF.5	I/O	MFP0	GPIO.
14	31	35	UART2_RXD	I	MFP2	UART2 数据接收脚.
			UART2_nCTS	I	MFP4	UART2 清除发送输入脚.
			BPWM0_CH4	I/O	MFP8	BPWM0 通道4 输出/捕获输入.
			EPWM0_SYNC_OUT	O	MFP9	EPWM0 计数器同步触发输出脚.
			X32_IN	I	MFP10	外部32.768 kHz 晶振输入脚.
			EADC0_ST	I	MFP11	EADC0 外部触发输入.
			PF.4	I/O	MFP0	GPIO.
15	32	36	UART2_TXD	O	MFP2	UART2 数据发送脚.
			UART2_nRTS	O	MFP4	UART2 请求发送输出脚.
			BPWM0_CH5	I/O	MFP8	BPWM0 通道5 输出/捕获输入.
			X32_OUT	O	MFP10	外部32.768 kHz 晶振输出脚.
			PH.0	I/O	MFP0	GPIO.

64 脚	128 脚	144 脚	引脚名称	类型	MFP	描述
			EBI_ADR7	O	MFP2	EBI 地址总线位 7.
			UART5_TXD	O	MFP4	UART5 数据发送脚.
			TM0_EXT	I/O	MFP13	定时器0事件计数器输入/反转输出脚.
		38	PH.1	I/O	MFP0	GPIO.
			EBI_ADR6	O	MFP2	EBI 地址总线位 6.
			UART5_RXD	I	MFP4	UART5 数据接收脚.
			TM1_EXT	I/O	MFP13	定时器1事件计数器输入/反转输出脚.
		39	PH.2	I/O	MFP0	GPIO.
			EBI_ADR5	O	MFP2	EBI 地址总线位5.
			UART5_nRTS	O	MFP4	UART5 请求发送输出脚.
			UART4_TXD	O	MFP5	UART4 数据发送脚.
			I2C0_SCL	I/O	MFP6	I2C0 时钟脚.
			TM2_EXT	I/O	MFP13	定时器2事件计数器输入/反转输出脚.
		40	PH.3	I/O	MFP0	GPIO.
			EBI_ADR4	O	MFP2	EBI 地址总线位4.
			SPI1_I2SMCLK	I/O	MFP3	SPI1 I2S 主机时钟输出脚
			UART5_nCTS	I	MFP4	UART5 清除发送输入脚.
			UART4_RXD	I	MFP5	UART4 数据接收脚.
			I2C0_SDA	I/O	MFP6	I2C0 数据输入/输出引脚.
			TM3_EXT	I/O	MFP13	定时器3事件计数器输入/反转输出脚.
		33	PH.4	I/O	MFP0	GPIO.
			EBI_ADR3	O	MFP2	EBI 地址总线位3.
			SPI1_MISO	I/O	MFP3	SPI1 MISO (主机输入, 从机输出) 脚.
		34	PH.5	I/O	MFP0	GPIO.
			EBI_ADR2	O	MFP2	EBI 地址总线位2.
			SPI1_MOSI	I/O	MFP3	SPI1 MOSI (主机输出, 从机输入) 脚.
		35	PH.6	I/O	MFP0	GPIO.
			EBI_ADR1	O	MFP2	EBI 地址总线位1.
			SPI1_CLK	I/O	MFP3	SPI1 串行时钟引脚.

<b>64 脚</b>	<b>128 脚</b>	<b>144 脚</b>	引脚名称	类型	MFP	描述
	36	44	PH.7	I/O	MFP0	GPIO.
			EBI_ADR0	O	MFP2	EBI 地址总线位0.
			SPI1_SS	I/O	MFP3	SPI1 从机选择脚.
	37	45	PF.3	I/O	MFP0	GPIO.
			EBI_nCS0	O	MFP2	EBI 片选 0 输出脚.
			UART0_RXD	O	MFP3	UART0 数据发送脚.
			I2C0_SCL	I/O	MFP4	I2C0 时钟脚.
			XT1_IN	I	MFP10	外部4~24 MHz (高速) 晶振输入脚.
			BPWM1_CH0	I/O	MFP11	BPWM1 通道0 输出/捕获输入.
	38	46	PF.2	I/O	MFP0	GPIO.
			EBI_nCS1	O	MFP2	EBI 片选 1 输出脚.
			UART0_RXD	I	MFP3	UART0 数据接收脚.
			I2C0_SDA	I/O	MFP4	I2C0 数据输入/输出引脚.
			QSPI0_CLK	I/O	MFP5	Quad SPI0 串行时钟引脚.
			XT1_OUT	O	MFP10	外部4~24 MHz (高速) 晶振输出引脚.
			BPWM1_CH1	I/O	MFP11	BPWM1 通道1 输出/捕获输入.
	39	47	V <sub>SS</sub>	P	MFP0	数字电路地.
	40	48	V <sub>DD</sub>	P	MFP0	I/O 端口电源和LDO 电源用于内部 PLL 和数字电路
	41	49	PE.8	I/O	MFP0	GPIO.
			EBI_ADR10	O	MFP2	EBI 地址总线位10.
			EMAC_RMII_MDC	O	MFP3	EMAC RMII PHY 管理 时钟输出脚.
			I2S0_BCLK	O	MFP4	I2S0 位 时钟输出脚.
			SPI2_CLK	I/O	MFP5	SPI2 串行时钟引脚.
			USCI1_CTL1	I/O	MFP6	USCI1 控制1 脚..
			UART2_RXD	O	MFP7	UART2 数据发送脚.
			EPWM0_CH0	I/O	MFP10	EPWM0 通道0 输出/捕获输入脚.
			EPWM0_BRAKE0	I	MFP11	EPWM0 刹车0输入脚.
			ECAP0_IC0	I	MFP12	增强型捕获输入单元0输入脚0
			TRACE_CLK	O	MFP14	ETM Trace 时钟输出脚

64 脚	128 脚	144 脚	引脚名称	类型	MFP	描述
42	50	42	PE.9	I/O	MFP0	GPIO.
			EBI_ADR11	O	MFP2	EBI 地址总线位11.
			EMAC_RMII_MDIO	I/O	MFP3	EMAC RMII PHY 管理数据脚.
			I2S0_MCLK	O	MFP4	I2S0 主机时钟输出脚.
			SPI2_MISO	I/O	MFP5	SPI2 MISO (主机输入, 从机输出) 脚.
			USCI1_CTL0	I/O	MFP6	USCI1 控制0 脚..
			UART2_RXD	I	MFP7	UART2 数据接收脚.
			EPWM0_CH1	I/O	MFP10	EPWM0 通道1 输出/捕获输入脚.
			EPWM0_BRAKE1	I	MFP11	EPWM0 刹车1输入脚.
			ECAP0_IC1	I	MFP12	增强型捕获输入单元0输入脚1
43	51	43	追踪_DATA0	O	MFP14	ETM 追踪数据 0 输出脚
			PE.10	I/O	MFP0	GPIO.
			EBI_ADR12	O	MFP2	EBI 地址总线位12.
			EMAC_RMII_TXD0	O	MFP3	EMAC RMII 发送数据总线位 0.
			I2S0_DI	I	MFP4	I2S0 数据输入脚.
			SPI2_MOSI	I/O	MFP5	SPI2 MOSI (主机输出, 从机输入) 脚.
			USCI1_DAT0	I/O	MFP6	USCI1 数据0脚.
			UART3_TXD	O	MFP7	UART3 数据发送脚.
			EPWM0_CH2	I/O	MFP10	EPWM0 通道2 输出/捕获输入脚.
			EPWM1_BRAKE0	I	MFP11	EPWM1 刹车0输入脚.
44	52	44	ECAP0_IC2	I	MFP12	增强型捕获输入单元0输入脚2
			TRACE_DATA1	O	MFP14	ETM 追踪数据 1 输出脚
			PE.11	I/O	MFP0	GPIO.
			EBI_ADR13	O	MFP2	EBI 地址总线位13.
			EMAC_RMII_TXD1	O	MFP3	EMAC RMII发送数据总线位1.
			I2S0_DO	O	MFP4	I2S0 数据输出脚
			SPI2_SS	I/O	MFP5	SPI2 从机选择脚.

<b>64 脚</b>	<b>128 脚</b>	<b>144 脚</b>	引脚名称	类型	MFP	描述
			UART1_nCTS	I	MFP8	UART1 清除发送输入脚.
			EPWM0_CH3	I/O	MFP10	EPWM0 通道3 输出/捕获输入脚.
			EPWM1_BRAKE1	I	MFP11	EPWM1 刹车1输入脚.
			ECAP1_IC2	I	MFP13	增强型捕获输入单元1输入脚2
			TRACE_DATA2	O	MFP14	ETM 追踪数据 2 输出脚
			PE.12	I/O	MFP0	GPIO.
			EBI_ADR14	O	MFP2	EBI 地址总线位14.
			EMAC_RMII_TXEN	O	MFP3	EMAC RMII 发送使能 输出脚.
			I2S0_LRCK	O	MFP4	I2S0 左右声道选择时钟输出脚.
			SPI2_I2SMCLK	I/O	MFP5	SPI2 I2S 主机时钟输出脚
			USCI1_CLK	I/O	MFP6	USCI1 时钟脚.
			UART1_nRTS	O	MFP8	UART1 请求发送输出脚.
			EPWM0_CH4	I/O	MFP10	EPWM0 通道4 输出/捕获输入脚.
			ECAP1_IC1	I	MFP13	增强型捕获输入单元1输入脚1
			TRACE_DATA3	O	MFP14	ETM 追踪数据 3 输出脚
			PE.13	I/O	MFP0	GPIO.
			EBI_ADR15	O	MFP2	EBI 地址总线位15.
			EMAC_PPS	O	MFP3	EMAC 秒脉冲 输出脚.
			I2C0_SCL	I/O	MFP4	I2C0 时钟脚.
			UART4_nRTS	O	MFP5	UART4 请求发送输出脚.
			UART1_TXD	O	MFP8	UART1 数据发送脚.
			EPWM0_CH5	I/O	MFP10	EPWM0 通道5 输出/捕获输入脚.
			EPWM1_CH0	I/O	MFP11	EPWM1 通道0 输出/捕获输入脚.
			BPWM1_CH5	I/O	MFP12	BPWM1 通道5 输出/捕获输入.
			ECAP1_IC0	I	MFP13	增强型捕获输入单元1输入脚0
			PC.8	I/O	MFP0	GPIO.
			EBI_ADR16	O	MFP2	EBI 地址总线位16.
			EMAC_RMII_REFCLK	I	MFP3	EMAC RMII 参考时钟输入脚.
			I2C0_SDA	I/O	MFP4	I2C0 数据输入/输出引脚.

64 脚	128 脚	144 脚	引脚名称	类型	MFP	描述
			UART4_nCTS	I	MFP5	UART4 清除发送输入脚.
			UART1_RXD	I	MFP8	UART1 数据接收脚.
			EPWM1_CH1	I/O	MFP11	EPWM1 通道1 输出/捕获输入脚.
			BPWM1_CH4	I/O	MFP12	BPWM1 通道4 输出/捕获输入.
			PC.7	I/O	MFP0	GPIO.
			EBI_AD9	I/O	MFP2	EBI 地址/数据总线位9.
			EMAC_RMII_RXD0	I	MFP3	EMAC RMII 接收数据总线位 0.
			SPI1_MISO	I/O	MFP4	SPI1 MISO (主机输入, 从机输出) 脚.
			UART4_TXD	O	MFP5	UART4 数据发送脚.
			SC2_PWR	O	MFP6	智能卡 2 电源脚.
			UART0_nCTS	I	MFP7	UART0 清除发送输入脚.
			I2C1_SMBAL	O	MFP8	I2C1 SMBus SMBALTER 脚
			EPWM1_CH2	I/O	MFP11	EPWM1 通道2 输出/捕获输入脚.
			BPWM1_CH0	I/O	MFP12	BPWM1 通道0 输出/捕获输入.
			TM0	I/O	MFP14	定时器0事件计数器输入/反转输出脚.
			INT3	I	MFP15	外部中断3输入脚.
			PC.6	I/O	MFP0	GPIO.
			EBI_AD8	I/O	MFP2	EBI 地址/数据总线位8.
			EMAC_RMII_RXD1	I	MFP3	EMAC RMII 接收数据总线位 1.
			SPI1_MOSI	I/O	MFP4	SPI1 MOSI (主机输出, 从机输入) 脚.
			UART4_RXD	I	MFP5	UART4 数据接收脚.
			SC2_RST	O	MFP6	智能卡 2 复位脚.
			UART0_nRTS	O	MFP7	UART0 请求发送输出脚.
			I2C1_SMBSUS	O	MFP8	I2C1 SMBus SMBSUS 脚(PMBus 控制脚)
			EPWM1_CH3	I/O	MFP11	EPWM1 通道3 输出/捕获输入脚.
			BPWM1_CH1	I/O	MFP12	BPWM1 通道1 输出/捕获输入.
			TM1	I/O	MFP14	定时器1事件计数器输入/反转输出脚.
			INT2	I	MFP15	外部中断2输入脚.
20	50	58	PA.7	I/O	MFP0	GPIO.

64 脚	128 脚	144 脚	引脚名称	类型	MFP	描述
			EBI_AD7	I/O	MFP2	EBI 地址/数据总线位7.
			EMAC_RMII_CRSDV	I	MFP3	EMAC RMII 载波侦听/接收数据输入脚.
			SPI1_CLK	I/O	MFP4	SPI1 串行时钟引脚.
			SC2_DAT	I/O	MFP6	智能卡2 数据脚.
			UART0_TXD	O	MFP7	UART0 数据发送脚.
			I2C1_SCL	I/O	MFP8	I2C1 时钟脚.
			EPWM1_CH4	I/O	MFP11	EPWM1 通道4 输出/捕获输入脚.
			BPWM1_CH2	I/O	MFP12	BPWM1 通道2 输出/捕获输入.
			ACMP0_WLAT	I	MFP13	模拟比较器0 窗口锁定输入脚
			TM2	I/O	MFP14	定时器2事件计数器输入/反转输出脚.
			INT1	I	MFP15	外部中断1输入脚.
			PA.6	I/O	MFP0	GPIO.
			EBI_AD6	I/O	MFP2	EBI 地址/数据总线位6.
			EMAC_RMII_RXERR	I	MFP3	EMAC RMII 接收数据错误输入脚
			SPI1_SS	I/O	MFP4	SPI1 从机选择脚.
			SD1_nCD	I	MFP5	SD卡接口1 卡侦测输入脚
			SC2_CLK	O	MFP6	智能卡 2 时钟脚.
			UART0_RXD	I	MFP7	UART0 数据接收脚.
			I2C1_SDA	I/O	MFP8	I2C1 数据输入/输出引脚.
			EPWM1_CH5	I/O	MFP11	EPWM1 通道5 输出/捕获输入脚.
			BPWM1_CH3	I/O	MFP12	BPWM1 通道3 输出/捕获输入.
			ACMP1_WLAT	I	MFP13	模拟比较器1 窗口锁定输入脚
			TM3	I/O	MFP14	定时器3事件计数器输入/反转输出脚.
			INT0	I	MFP15	外部中断0输入脚.
22	52	60	V <sub>SS</sub>	P	MFP0	数字电路地.
23	53	61	V <sub>DD</sub>	P	MFP0	I/O 端口电源和LDO 电源用于内部 PLL 和数字电路
24	54	62	LDO_CAP	A	MFP0	LDO 输出脚.
			PA.5	I/O	MFP0	GPIO.
			SPI_M_D2	I/O	MFP2	SPI_M 四线模式数据2 .

64 脚	128 脚	144 脚	引脚名称	类型	MFP	描述
			QSPI0_MISO1	I/O	MFP3	Quad SPI0 MISO1 (主机输入, 从机输出) 脚..
			SPI1_I2SMCLK	I/O	MFP4	SPI1 I2S 主机时钟输出脚
			SD1_CMD	I/O	MFP5	SD卡接口1 命令/应答脚
			SC2_nCD	I	MFP6	智能卡 2 卡侦测脚.
			UART0_nCTS	I	MFP7	UART0 清除发送输入脚.
			UART5_TXD	O	MFP8	UART5 数据发送脚.
			I2C0_SCL	I/O	MFP9	I2C0 时钟脚.
			CAN0_TXD	O	MFP10	CAN0 总线发送输出.
			BPWM0_CH5	I/O	MFP12	BPWM0 通道5 输出/捕获输入.
			EPWM0_CH0	I/O	MFP13	EPWM0 通道0 输出/捕获输入脚.
			QEIO_INDEX	I	MFP14	正交编码器0索引输入
			PA.4	I/O	MFP0	GPIO.
			SPIM_D3	I/O	MFP2	SPIM 四线模式数据3 .
			QSPI0_MOSI1	I/O	MFP3	Quad SPI0 MOSI1 (主机输出, 从机输入) 脚.
			SPI0_I2SMCLK	I/O	MFP4	SPI0 I2S 主机时钟输出脚
			SD1_CLK	O	MFP5	SD卡接口1 时钟输出脚
			SC0_nCD	I	MFP6	智能卡 0 卡侦测脚.
			UART0_nRTS	O	MFP7	UART0 请求发送输出脚.
			UART5_RXD	I	MFP8	UART5 数据接收脚.
			I2C0_SDA	I/O	MFP9	I2C0 数据输入/输出引脚.
			CAN0_RXD	I	MFP10	CAN0 总线接收输入
			BPWM0_CH4	I/O	MFP12	BPWM0 通道4 输出/捕获输入.
			EPWM0_CH1	I/O	MFP13	EPWM0 通道1 输出/捕获输入脚.
			QEIO_A	I	MFP14	正交编码0 计数信号 A 相输入
			PA.3	I/O	MFP0	GPIO.
			SPIM_SS	I/O	MFP2	SPIM 从机选择脚.
			QSPI0_SS	I/O	MFP3	Quad SPI0 从机选择脚.
			SPI0_SS	I/O	MFP4	SPI0 从机选择脚.
			SD1_DAT3	I/O	MFP5	SD卡接口1 数据位3.

64 脚	128 脚	144 脚	引脚名称	类型	MFP	描述
			SC0_PWR	O	MFP6	智能卡0电源脚.
			UART4_TXD	O	MFP7	UART4数据发送脚.
			UART1_TXD	O	MFP8	UART1数据发送脚.
			I2C1_SCL	I/O	MFP9	I2C1时钟脚.
			BPWM0_CH3	I/O	MFP12	BPWM0通道3输出/捕获输入.
			EPWM0_CH2	I/O	MFP13	EPWM0通道2输出/捕获输入脚.
			QEIO_B	I	MFP14	正交编码0计数信号B相输入
			PA.2	I/O	MFP0	GPIO.
			SPIIM_CLK	I/O	MFP2	SPIIM串行时钟引脚.
			QSPI0_CLK	I/O	MFP3	Quad SPI0串行时钟引脚.
			SPI0_CLK	I/O	MFP4	SPI0串行时钟引脚.
			SD1_DAT2	I/O	MFP5	SD卡接口1数据位2.
			SC0_RST	O	MFP6	智能卡0复位脚.
			UART4_RXD	I	MFP7	UART4数据接收脚.
			UART1_RXD	I	MFP8	UART1数据接收脚.
			I2C1_SDA	I/O	MFP9	I2C1数据输入/输出引脚.
			BPWM0_CH2	I/O	MFP12	BPWM0通道2输出/捕获输入.
			EPWM0_CH3	I/O	MFP13	EPWM0通道3输出/捕获输入脚.
			PA.1	I/O	MFP0	GPIO.
			SPIIM_MISO	I/O	MFP2	SPIIM MISO(主机输入,从机输出)脚..
			QSPI0_MISO0	I/O	MFP3	Quad SPI0 MISO0(主机输入,从机输出)脚..
			SPI0_MISO	I/O	MFP4	SPI0 MISO(主机输入,从机输出)脚..
			SD1_DAT1	I/O	MFP5	SD卡接口1数据位1.
			SC0_DAT	I/O	MFP6	智能卡0数据脚.
			UART0_TXD	O	MFP7	UART0数据发送脚.
			UART1_nCTS	I	MFP8	UART1清除发送输入脚.
			I2C2_SCL	I/O	MFP9	I2C2时钟脚.
			BPWM0_CH1	I/O	MFP12	BPWM0通道1输出/捕获输入.
			EPWM0_CH4	I/O	MFP13	EPWM0通道4输出/捕获输入脚.

64 脚	128 脚	144 脚	引脚名称	类型	MFP	描述
			DAC1_ST	I	MFP15	DAC1 外部触发输入.
30	60	68	PA.0	I/O	MFP0	GPIO.
			SPI_MOSI	I/O	MFP2	SPI_MOSI (主机输出, 从机输入) 脚.
			QSPI0_MOSIO	I/O	MFP3	Quad SPI0 MOSIO (主机输出, 从机输入) 脚.
			SPI0_MOSI	I/O	MFP4	SPI0 MOSI (主机输出, 从机输入) 脚.
			SD1_DAT0	I/O	MFP5	SD卡接口1 数据位0.
			SC0_CLK	O	MFP6	智能卡0时钟脚.
			UART0_RXD	I	MFP7	UART0 数据接收脚.
			UART1_nRTS	O	MFP8	UART1 请求发送输出脚.
			I2C2_SDA	I/O	MFP9	I2C2 数据输入/输出引脚.
			BPWM0_CH0	I/O	MFP12	BPWM0 通道0 输出/捕获输入.
31	61	69	EPWM0_CH5	I/O	MFP13	EPWM0 通道5 输出/捕获输入脚.
			DAC0_ST	I	MFP15	DAC0 外部触发输入.
			V <sub>DDIO</sub>	P	MFP0	PE.1, PE.8~PE.13的电源.
			PE.14	I/O	MFP0	GPIO.
			EBI_AD8	I/O	MFP2	EBI 地址/数据总线位8.
62	70	70	UART2_TXD	O	MFP3	UART2 数据发送脚.
			CAN0_TXD	O	MFP4	CAN0 总线发送输出.
			SD1_nCD	I	MFP5	SD卡接口1 卡侦测输入脚
			PE.15	I/O	MFP0	GPIO.
			EBI_AD9	I/O	MFP2	EBI 地址/数据总线位9.
63	71	71	UART2_RXD	I	MFP3	UART2 数据接收脚.
			CAN0_RXD	I	MFP4	CAN0 总线接收输入
			nRESET	I	MFP0	外部复位输入: 低电平有效, 带内部上拉.
			PF.0	I/O	MFP0	GPIO.
33	65	73	UART1_TXD	O	MFP2	UART1 数据发送脚.
			I2C1_SCL	I/O	MFP3	I2C1 时钟脚.
			BPWM1_CH0	I/O	MFP12	BPWM1 通道0 输出/捕获输入.
			ICE_DAT	O	MFP14	串行调试器数据脚.

<b>64 脚</b>	<b>128 脚</b>	<b>144 脚</b>	引脚名称	类型	MFP	描述
34	66	74	PF.1	I/O	MFP0	GPIO.
			UART1_RXD	I	MFP2	UART1 数据接收脚.
			I2C1_SDA	I/O	MFP3	I2C1 数据输入/输出引脚.
			BPWM1_CH1	I/O	MFP12	BPWM1 通道1 输出/捕获输入.
			ICE_CLK	I	MFP14	串行调试器时钟脚.
	67	75	PD.9	I/O	MFP0	GPIO.
			EBI_AD7	I/O	MFP2	EBI 地址/数据总线位7.
			I2C2_SCL	I/O	MFP3	I2C2 时钟脚.
			UART2_nCTS	I	MFP4	UART2 清除发送输入脚.
	68	76	PD.8	I/O	MFP0	GPIO.
			EBI_AD6	I/O	MFP2	EBI 地址/数据总线位6.
			I2C2_SDA	I/O	MFP3	I2C2 数据输入/输出引脚.
			UART2_nRTS	O	MFP4	UART2 请求发送输出脚.
	35	69	PC.5	I/O	MFP0	GPIO.
			EBI_AD5	I/O	MFP2	EBI 地址/数据总线位5.
			SPIM_D2	I/O	MFP3	SPIM 四线模式数据2 .
			QSPI0_MISO1	I/O	MFP4	Quad SPI0 MISO1 (主机输入, 从机输出) 脚..
			UART2_TXD	O	MFP8	UART2 数据发送脚.
			I2C1_SCL	I/O	MFP9	I2C1 时钟脚.
			CAN0_TXD	O	MFP10	CAN0 总线发送输出.
			UART4_TXD	O	MFP11	UART4 数据发送脚.
	36	70	PC.4	I/O	MFP0	GPIO.
			EBI_AD4	I/O	MFP2	EBI 地址/数据总线位4.
			SPIM_D3	I/O	MFP3	SPIM 四线模式数据3 .
			QSPI0_MOSI1	I/O	MFP4	Quad SPI0 MOSI1 (主机输出, 从机输入) 脚..
			SC1_nCD	I	MFP5	智能卡 1 卡侦测脚.
			I2S0_BCLK	O	MFP6	I2S0 位 时钟输出脚.
			SPI1_I2SMCLK	I/O	MFP7	SPI1 I2S 主机时钟输出脚

<b>64 脚</b>	<b>128 脚</b>	<b>144 脚</b>	引脚名称	类型	MFP	描述
37	71	79	UART2_RXD	I	MFP8	UART2 数据接收脚.
			I2C1_SDA	I/O	MFP9	I2C1 数据输入/输出引脚.
			CAN0_RXD	I	MFP10	CAN0 总线接收输入
			UART4_RXD	I	MFP11	UART4 数据接收脚.
			EPWM1_CH1	I/O	MFP12	EPWM1 通道1 输出/捕获输入脚.
38	72	80	PC.3	I/O	MFP0	GPIO.
			EBI_AD3	I/O	MFP2	EBI 地址/数据总线位3.
			SPIM_SS	I/O	MFP3	SPIM 从机选择脚.
			QSPI0_SS	I/O	MFP4	Quad SPI0 从机选择脚.
			SC1_PWR	O	MFP5	智能卡 1 电源脚.
			I2S0_MCLK	O	MFP6	I2S0 主机时钟输出脚.
			SPI1_MISO	I/O	MFP7	SPI1 MISO (主机输入, 从机输出) 脚.
			UART2_nRTS	O	MFP8	UART2 请求发送输出脚.
			I2C0_SMBAL	O	MFP9	I2C0 SMBus SMBALTER 脚
			CAN1_TXD	O	MFP10	CAN1 总线发送输出.
			UART3_TXD	O	MFP11	UART3 数据发送脚.
			EPWM1_CH2	I/O	MFP12	EPWM1 通道2 输出/捕获输入脚.
38	72	80	PC.2	I/O	MFP0	GPIO.
			EBI_AD2	I/O	MFP2	EBI 地址/数据总线位2.
			SPIM_CLK	I/O	MFP3	SPIM 串行时钟引脚.
			QSPI0_CLK	I/O	MFP4	Quad SPI0 串行时钟引脚.
			SC1_RST	O	MFP5	智能卡 1 复位脚.
			I2S0_DI	I	MFP6	I2S0 数据输入脚.
			SPI1_MOSI	I/O	MFP7	SPI1 MOSI (主机输出, 从机输入) 脚.
			UART2_nCTS	I	MFP8	UART2 清除发送输入脚.
			I2C0_SMBSUS	O	MFP9	I2C0 SMBus SMBSUS 脚 (PMBus 控制脚)
			CAN1_RXD	I	MFP10	CAN1 总线接收输入
			UART3_RXD	I	MFP11	UART3 数据接收脚.
			EPWM1_CH3	I/O	MFP12	EPWM1 通道3 输出/捕获输入脚.

<b>64 脚</b>	<b>128 脚</b>	<b>144 脚</b>	引脚名称	类型	MFP	描述
39	73	81	PC.1	I/O	MFP0	GPIO.
			EBI_ADI	I/O	MFP2	EBI 地址/数据总线位1.
			SPIM_MISO	I/O	MFP3	SPIM MISO (主机输入, 从机输出) 脚..
			QSPI0_MISO0	I/O	MFP4	Quad SPI0 MISO0 (主机输入, 从机输出) 脚..
			SC1_DAT	I/O	MFP5	智能卡1 数据脚.
			I2S0_DO	O	MFP6	I2S0 数据输出脚
			SPI1_CLK	I/O	MFP7	SPI1 串行时钟引脚.
			UART2_TXD	O	MFP8	UART2 数据发送脚.
			I2C0_SCL	I/O	MFP9	I2C0 时钟脚.
			EPWM1_CH4	I/O	MFP12	EPWM1 通道4 输出/捕获输入脚.
			ACMP0_O	O	MFP14	模拟比较器0 输出脚.
40	74	82	PC.0	I/O	MFP0	GPIO.
			EBI_ADO	I/O	MFP2	EBI 地址/数据总线位0.
			SPIM_MOSI	I/O	MFP3	SPIM MOSI (主机输出, 从机输入) 脚.
			QSPI0_MOSI0	I/O	MFP4	Quad SPI0 MOSI0 (主机输出, 从机输入) 脚.
			SC1_CLK	O	MFP5	智能卡 1 时钟脚.
			I2S0_LRCK	O	MFP6	I2S0 左右声道选择时钟输出脚.
			SPI1_SS	I/O	MFP7	SPI1 从机选择脚.
			UART2_RXD	I	MFP8	UART2 数据接收脚.
			I2C0_SDA	I/O	MFP9	I2C0 数据输入/输出引脚.
			EPWM1_CH5	I/O	MFP12	EPWM1 通道5 输出/捕获输入脚.
			ACMP1_O	O	MFP14	模拟比较器1 输出脚.
	75	83	V <sub>ss</sub>	P	MFP0	数字电路地.
	76	84	V <sub>DD</sub>	P	MFP0	I/O 端口电源和LDO 电源用于内部 PLL 和数字电路
	77	85	PG.9	I/O	MFP0	GPIO.
			EBI_ADO	I/O	MFP2	EBI 地址/数据总线位0.
			SD1_DAT3	I/O	MFP3	SD 卡接口1 数据位3.
			SPIM_D2	I/O	MFP4	SPIM 四线模式数据2 .
			BPWM0_CH5	I/O	MFP12	BPWM0 通道5 输出/捕获输入.

<b>64 脚</b>	<b>128 脚</b>	<b>144 脚</b>	引脚名称	类型	MFP	描述
	78	86	PG.10	I/O	MFP0	GPIO.
			EBI_AD1	I/O	MFP2	EBI 地址/数据总线位1.
			SD1_DAT2	I/O	MFP3	SD卡接口1 数据位2.
			SPIM_D3	I/O	MFP4	SPIM 四线模式数据3 .
			BPWM0_CH4	I/O	MFP12	BPWM0 通道4 输出/捕获输入.
	79	87	PG.11	I/O	MFP0	GPIO.
			EBI_AD2	I/O	MFP2	EBI 地址/数据总线位2.
			SD1_DAT1	I/O	MFP3	SD卡接口1 数据位1.
			SPIM_SS	I/O	MFP4	SPIM 从机选择脚.
			BPWM0_CH3	I/O	MFP12	BPWM0 通道3 输出/捕获输入.
	80	88	PG.12	I/O	MFP0	GPIO.
			EBI_AD3	I/O	MFP2	EBI 地址/数据总线位3.
			SD1_DAT0	I/O	MFP3	SD卡接口1 数据位0.
			SPIM_CLK	I/O	MFP4	SPIM 串行时钟引脚.
			BPWM0_CH2	I/O	MFP12	BPWM0 通道2 输出/捕获输入.
	81	89	PG.13	I/O	MFP0	GPIO.
			EBI_AD4	I/O	MFP2	EBI 地址/数据总线位4.
			SD1_CMD	I/O	MFP3	SD卡接口1 命令/应答脚
			SPIM_MISO	I/O	MFP4	SPIM MISO (主机输入, 从机输出) 脚..
			BPWM0_CH1	I/O	MFP12	BPWM0 通道1 输出/捕获输入.
	82	90	PG.14	I/O	MFP0	GPIO.
			EBI_AD5	I/O	MFP2	EBI 地址/数据总线位5.
			SD1_CLK	O	MFP3	SD卡接口1 时钟输出脚
			SPIM_MOSI	I/O	MFP4	SPIM MOSI (主机输出, 从机输入) 脚.
			BPWM0_CH0	I/O	MFP12	BPWM0 通道0 输出/捕获输入.
	83	91	PG.15	I/O	MFP0	GPIO.
			SD1_nCD	I	MFP3	SD卡接口1 卡侦测输入脚
			CLKO	O	MFP14	时钟输出
			EADC0_ST	I	MFP15	EADC0 外部触发输入.

<b>64 脚</b>	<b>128 脚</b>	<b>144 脚</b>	引脚名称	类型	MFP	描述
		92	PD.3	I/O	MFP0	GPIO.
			EBI_AD10	I/O	MFP2	EBI 地址/数据总线位10.
			USCI0_CTL1	I/O	MFP3	USCI0 控制1 脚..
			SPI0_SS	I/O	MFP4	SPI0 从机选择脚.
			UART3_nRTS	O	MFP5	UART3 请求发送输出脚.
			USCI1_CTL0	I/O	MFP6	USCI1 控制0 脚..
			SC2_PWR	O	MFP7	智能卡 2 电源脚.
			SC1_nCD	I	MFP8	智能卡 1 卡侦测脚.
			UART0_TXD	O	MFP9	UART0 数据发送脚.
		93	PD.2	I/O	MFP0	GPIO.
			EBI_AD11	I/O	MFP2	EBI 地址/数据总线位11.
			USCI0_DAT1	I/O	MFP3	USCI0 数据1脚.
			SPI0_CLK	I/O	MFP4	SPI0 串行时钟引脚.
			UART3_nCTS	I	MFP5	UART3 清除发送输入脚.
			SC2_RST	O	MFP7	智能卡 2 复位脚.
			UART0_RXD	I	MFP9	UART0 数据接收脚.
		94	PD.1	I/O	MFP0	GPIO.
			EBI_AD12	I/O	MFP2	EBI 地址/数据总线位12.
			USCI0_DAT0	I/O	MFP3	USCI0 数据0脚.
			SPI0_MISO	I/O	MFP4	SPI0 MISO (主机输入, 从机输出) 脚.
			UART3_TXD	O	MFP5	UART3 数据发送脚.
			I2C2_SCL	I/O	MFP6	I2C2 时钟脚.
			SC2_DAT	I/O	MFP7	智能卡2 数据脚.
		95	PD.0	I/O	MFP0	GPIO.
			EBI_AD13	I/O	MFP2	EBI 地址/数据总线位13.
			USCI0_CLK	I/O	MFP3	USCI0 时钟脚.
			SPI0_MOSI	I/O	MFP4	SPI0 MOSI (主机输出, 从机输入) 脚.
			UART3_RXD	I	MFP5	UART3 数据接收脚.
			I2C2_SDA	I/O	MFP6	I2C2 数据输入/输出引脚.

64 脚	128 脚	144 脚	引脚名称	类型	MFP	描述
			SC2_CLK	O	MFP7	智能卡 2 时钟脚.
			TM2	I/O	MFP14	定时器2事件计数器输入/反转输出脚.
	84	96	PD.13	I/O	MFP0	GPIO.
			EBI_AD10	I/O	MFP2	EBI 地址/数据总线位10.
			SD0_nCD	I	MFP3	SD卡接口0 卡侦测输入脚
			SPI0_I2SMCLK	I/O	MFP4	SPI0 I2S 主机时钟输出脚
			SPI1_I2SMCLK	I/O	MFP5	SPI1 I2S 主机时钟输出脚
			SC2_nCD	I	MFP7	智能卡 2 卡侦测脚.
	85	97	PA.12	I/O	MFP0	GPIO.
			I2S0_BCLK	O	MFP2	I2S0 位 时钟输出脚.
			UART4_TXD	O	MFP3	UART4 数据发送脚.
			I2C1_SCL	I/O	MFP4	I2C1 时钟脚.
			SPI2_SS	I/O	MFP5	SPI2 从机选择脚.
			CAN0_TXD	O	MFP6	CAN0 总线发送输出.
			SC2_PWR	O	MFP7	智能卡 2 电源脚.
			BPWM1_CH2	I/O	MFP11	BPWM1 通道2 输出/捕获输入.
			QEI1_INDEX	I	MFP12	正交编码器1索引输入
	86	98	USB_VBUS	P	MFP14	来自USB主机或HUB的电源
			PA.13	I/O	MFP0	GPIO.
			I2S0_MCLK	O	MFP2	I2S0 主机时钟输出脚.
			UART4_RXD	I	MFP3	UART4 数据接收脚.
			I2C1_SDA	I/O	MFP4	I2C1 数据输入/输出引脚.
			SPI2_CLK	I/O	MFP5	SPI2 串行时钟引脚.
			CAN0_RXD	I	MFP6	CAN0 总线接收输入
			SC2_RST	O	MFP7	智能卡 2 复位脚.
			BPWM1_CH3	I/O	MFP11	BPWM1 通道3 输出/捕获输入.
			QEI1_A	I	MFP12	正交编码1 计数信号 A 相输入
	87	99	USB_D-	A	MFP14	USB 差分信号D+.
			PA.14	I/O	MFP0	GPIO.

64 脚	128 脚	144 脚	引脚名称	类型	MFP	描述
			I2S0_DI	I	MFP2	I2S0 数据输入脚.
			UART0_TXD	O	MFP3	UART0 数据发送脚.
			SPI2_MISO	I/O	MFP5	SPI2 MISO (主机输入, 从机输出) 脚.
			I2C2_SCL	I/O	MFP6	I2C2 时钟脚.
			SC2_DAT	I/O	MFP7	智能卡2 数据脚.
			BPWM1_CH4	I/O	MFP11	BPWM1 通道4 输出/捕获输入.
			QEI1_B	I	MFP12	正交编码1 计数信号 B 相输入
			USB_D+	A	MFP14	USB 差分信号D-.
			PA.15	I/O	MFP0	GPIO.
			I2S0_DO	O	MFP2	I2S0 数据输出脚
			UART0_RXD	I	MFP3	UART0 数据接收脚.
			SPI2_MOSI	I/O	MFP5	SPI2 MOSI (主机输出, 从机输入) 脚.
			I2C2_SDA	I/O	MFP6	I2C2 数据输入/输出引脚.
			SC2_CLK	O	MFP7	智能卡 2 时钟脚.
			BPWM1_CH5	I/O	MFP11	BPWM1 通道5 输出/捕获输入.
			EPWM0_SYNC_IN	I	MFP12	EPWM0 计数器同步触发输入脚
			USB_OTG_ID	I	MFP14	USB_ID.
41	89	101	HSUSB_VRES	A	MFP0	HSUSB 模块参考电阻
42	90	102	HSUSB_VDD33	P	MFP0	HSUSB VDD33电源
43	91	103	HSUSB_VBUS	P	MFP0	HSUSB 来自USB主机或HUB的电源
44	92	104	HSUSB_D-	A	MFP0	HSUSB 差分信号D+.
45	93	105	HSUSB_VSS	P	MFP0	HSUSB地
46	94	106	HSUSB_D+	A	MFP0	HSUSB 差分信号D-.
47	95	107	HSUSB_VDD12_CAP	A	MFP0	HSUSB 内部电源管理输出1.2V 去耦脚 注: 该脚需要连接一个1uF 电容.
48	96	108	HSUSB_ID	I	MFP0	HSUSB ID.
			PE.7	I/O	MFP0	GPIO.
			SD0_CMD	I/O	MFP3	SD卡接口0 命令/应答脚
			SPIM_D2	I/O	MFP4	SPIM 四线模式数据2.

<b>64 脚</b>	<b>128 脚</b>	<b>144 脚</b>	引脚名称	类型	MFP	描述
			UART5_TXD	O	MFP8	UART5 数据发送脚.
			CAN1_TXD	O	MFP9	CAN1 总线发送输出.
			QEI1_INDEX	I	MFP11	正交编码器1索引输入
			EPWM0_CH0	I/O	MFP12	EPWM0 通道0 输出/捕获输入脚.
			BPWM0_CH5	I/O	MFP13	BPWM0 通道5 输出/捕获输入.
		98	PE.6	I/O	MFP0	GPIO.
			SD0_CLK	O	MFP3	SD卡接口0 时钟输出脚
			SPIM_D3	I/O	MFP4	SPIM 四线模式数据3 .
			SPI3_I2SMCLK	I/O	MFP5	SPI3 I2S 主机时钟输出脚
			SC0_nCD	I	MFP6	智能卡 0 卡侦测脚.
			USCI0_CTL0	I/O	MFP7	USCI0 控制0 脚..
			UART5_RXD	I	MFP8	UART5 数据接收脚.
			CAN1_RXD	I	MFP9	CAN1 总线接收输入
			QEI1_A	I	MFP11	正交编码1 计数信号 A 相输入
			EPWM0_CH1	I/O	MFP12	EPWM0 通道1 输出/捕获输入脚.
		99	PE.5	I/O	MFP0	GPIO.
			EBI_nRD	O	MFP2	EBI 读使能输出脚.
			SD0_DAT3	I/O	MFP3	SD卡接口0 数据位3.
			SPIM_SS	I/O	MFP4	SPIM 从机选择脚.
			SPI3_SS	I/O	MFP5	SPI3 从机选择脚.
			SC0_PWR	O	MFP6	智能卡 0 电源脚.
			USCI0_CTL1	I/O	MFP7	USCI0 控制1 脚..
			QEI1_B	I	MFP11	正交编码1 计数信号 B 相输入
			EPWM0_CH2	I/O	MFP12	EPWM0 通道2 输出/捕获输入脚.
			BPWM0_CH3	I/O	MFP13	BPWM0 通道3 输出/捕获输入.
		100	PE.4	I/O	MFP0	GPIO.
			EBI_nWR	O	MFP2	EBI 写使能输出脚.
			SD0_DAT2	I/O	MFP3	SD卡接口0 数据位2.

64 脚	128 脚	144 脚	引脚名称	类型	MFP	描述
			SPI_M_CLK	I/O	MFP4	SPI_M 串行时钟引脚.
			SPI3_M_CLK	I/O	MFP5	SPI3 串行时钟引脚.
			SC0_RST	O	MFP6	智能卡 0 复位脚.
			USCI0_DAT1	I/O	MFP7	USCI0 数据1脚.
			QEIO_INDEX	I	MFP11	正交编码器0索引输入
			EPWM0_CH3	I/O	MFP12	EPWM0 通道3 输出/捕获输入脚.
			BPWM0_CH2	I/O	MFP13	BPWM0 通道2 输出/捕获输入.
			PE.3	I/O	MFP0	GPIO.
			EBI_MCLK	O	MFP2	EBI外部时钟输出脚.
			SD0_DAT1	I/O	MFP3	SD卡接口0 数据位1.
			SPI_M_MISO	I/O	MFP4	SPI_M MISO (主机输入, 从机输出) 脚..
			SPI3_M_MISO	I/O	MFP5	SPI3 MISO (主机输入, 从机输出) 脚..
			SC0_DAT	I/O	MFP6	智能卡0 数据脚.
			USCI0_DAT0	I/O	MFP7	USCI0 数据0脚.
			QEIO_A	I	MFP11	正交编码0 计数信号 A 相输入
			EPWM0_CH4	I/O	MFP12	EPWM0 通道4 输出/捕获输入脚.
			BPWM0_CH1	I/O	MFP13	BPWM0 通道1 输出/捕获输入.
			PE.2	I/O	MFP0	GPIO.
			EBI_ALE	O	MFP2	EBI 地址锁存使能输出脚.
			SD0_DAT0	I/O	MFP3	SD卡接口0 数据位0.
			SPI_M_MOSI	I/O	MFP4	SPI_M MOSI (主机输出, 从机输入) 脚.
			SPI3_M_MOSI	I/O	MFP5	SPI3 MOSI (主机输出, 从机输入) 脚.
			SC0_CLK	O	MFP6	智能卡 0 时钟脚.
			USCI0_CLK	I/O	MFP7	USCI0 时钟脚.
			QEIO_B	I	MFP11	正交编码0 计数信号 B 相输入
			EPWM0_CH5	I/O	MFP12	EPWM0 通道5 输出/捕获输入脚.
			BPWM0_CH0	I/O	MFP13	BPWM0 通道0 输出/捕获输入.
	103	115	V <sub>SS</sub>	P	MFP0	数字电路地.
	104	116	V <sub>DD</sub>	P	MFP0	I/O 端口电源和LDO 电源用于内部 PLL 和数字电路

64 脚	128 脚	144 脚	引脚名称	类型	MFP	描述
105	117	PE.1	I/O	MFP0	GPIO.	
		EBI_AD10	I/O	MFP2	EBI 地址/数据总线位10.	
		QSPI0_MISO0	I/O	MFP3	Quad SPI0 MISO0 (主机输入, 从机输出) 脚..	
		SC2_DAT	I/O	MFP4	智能卡2 数据脚.	
		I2S0_BCLK	O	MFP5	I2S0 位 时钟输出脚.	
		SPI1_MISO	I/O	MFP6	SPI1 MISO (主机输入, 从机输出) 脚.	
		UART3_TXD	O	MFP7	UART3 数据发送脚.	
		I2C1_SCL	I/O	MFP8	I2C1 时钟脚.	
		UART4_nCTS	I	MFP9	UART4 清除发送输入脚.	
106	118	PE.0	I/O	MFP0	GPIO.	
		EBI_AD11	I/O	MFP2	EBI 地址/数据总线位11.	
		QSPI0_MOSI0	I/O	MFP3	Quad SPI0 MOSI0 (主机输出, 从机输入) 脚.	
		SC2_CLK	O	MFP4	智能卡 2 时钟脚.	
		I2S0_MCLK	O	MFP5	I2S0 主机时钟输出脚.	
		SPI1_MOSI	I/O	MFP6	SPI1 MOSI (主机输出, 从机输入) 脚.	
		UART3_RXD	I	MFP7	UART3 数据接收脚.	
		I2C1_SDA	I/O	MFP8	I2C1 数据输入/输出引脚.	
		UART4_nRTS	O	MFP9	UART4 请求发送输出脚.	
107	119	PH.8	I/O	MFP0	GPIO.	
		EBI_AD12	I/O	MFP2	EBI 地址/数据总线位12.	
		QSPI0_CLK	I/O	MFP3	Quad SPI0 串行时钟引脚.	
		SC2_PWR	O	MFP4	智能卡 2 电源脚.	
		I2S0_DI	I	MFP5	I2S0 数据输入脚.	
		SPI1_CLK	I/O	MFP6	SPI1 串行时钟引脚.	
		UART3_nRTS	O	MFP7	UART3 请求发送输出脚.	
		I2C1_SMBAL	O	MFP8	I2C1 SMBus SMBALTER 脚	
		I2C2_SCL	I/O	MFP9	I2C2 时钟脚.	
	108	120	PH.9	I/O	MFP0	GPIO.

64 脚	128 脚	144 脚	引脚名称	类型	MFP	描述
			EBI_AD13	I/O	MFP2	EBI 地址/数据总线位13.
			QSPI0_SS	I/O	MFP3	Quad SPI0 从机选择脚.
			SC2_RST	O	MFP4	智能卡 2 复位脚.
			I2S0_DO	O	MFP5	I2S0 数据输出脚
			SPI1_SS	I/O	MFP6	SPI1 从机选择脚.
			UART3_nCTS	I	MFP7	UART3 清除发送输入脚.
			I2C1_SMBSUS	O	MFP8	I2C1 SMBus SMBSUS 脚(PMBus 控制脚)
			I2C2_SDA	I/O	MFP9	I2C2 数据输入/输出引脚.
			UART1_RXD	I	MFP10	UART1 数据接收脚.
			PH.10	I/O	MFP0	GPIO.
			EBI_AD14	I/O	MFP2	EBI 地址/数据总线位14.
			QSPI0_MISO1	I/O	MFP3	Quad SPI0 MISO1 (主机输入, 从机输出) 脚..
			SC2_nCD	I	MFP4	智能卡 2 卡侦测脚.
			I2S0_LRCK	O	MFP5	I2S0 左右声道选择时钟输出脚.
			SPI1_I2SMCLK	I/O	MFP6	SPI1 I2S 主机时钟输出脚
			UART4_TXD	O	MFP7	UART4 数据发送脚.
			UART0_TXD	O	MFP8	UART0 数据发送脚.
			PH.11	I/O	MFP0	GPIO.
			EBI_AD15	I/O	MFP2	EBI 地址/数据总线位15.
			QSPI0_MOSI1	I/O	MFP3	Quad SPI0 MOSI1 (主机输出, 从机输入) 脚.
			UART4_RXD	I	MFP7	UART4 数据接收脚.
			UART0_RXD	I	MFP8	UART0 数据接收脚.
			EPWM0_CH5	I/O	MFP11	EPWM0 通道5 输出/捕获输入脚.
			PD.14	I/O	MFP0	GPIO.
			EBI_nCS0	O	MFP2	EBI 片选 0 输出脚.
			SPI3_I2SMCLK	I/O	MFP3	SPI3 I2S 主机时钟输出脚
			SC1_nCD	I	MFP4	智能卡 1 卡侦测脚.
			EPWM0_CH4	I/O	MFP11	EPWM0 通道4 输出/捕获输入脚.
			PG.5	I/O	MFP0	GPIO.

64 脚	128 脚	144 脚	引脚名称	类型	MFP	描述
			EBI_nCS1	O	MFP2	EBI 片选 1 输出脚.
			SPI3_SS	I/O	MFP3	SPI3 从机选择脚.
			SC1_PWR	O	MFP4	智能卡 1 电源脚.
			EPWM0_CH3	I/O	MFP11	EPWM0 通道3 输出/捕获输入脚.
		125	PG.6	I/O	MFP0	GPIO.
			EBI_nCS2	O	MFP2	EBI 片选 2 输出脚.
			SPI3_CLK	I/O	MFP3	SPI3 串行时钟引脚.
			SC1_RST	O	MFP4	智能卡 1 复位脚.
			EPWM0_CH2	I/O	MFP11	EPWM0 通道2 输出/捕获输入脚.
		126	PG.7	I/O	MFP0	GPIO.
			EBI_nWRL	O	MFP2	EBI 低字节写使能输出脚.
			SPI3_MISO	I/O	MFP3	SPI3 MISO (主机输入, 从机输出) 脚..
			SC1_DAT	I/O	MFP4	智能卡1 数据脚.
			EPWM0_CH1	I/O	MFP11	EPWM0 通道1 输出/捕获输入脚.
		127	PG.8	I/O	MFP0	GPIO.
			EBI_nWRH	O	MFP2	EBI 高字节写使能输出脚
			SPI3_MOSI	I/O	MFP3	SPI3 MOSI (主机输出, 从机输入) 脚.
			SC1_CLK	O	MFP4	智能卡 1 时钟脚.
			EPWM0_CH0	I/O	MFP11	EPWM0 通道0 输出/捕获输入脚.
49	112	128	V <sub>ss</sub>	P	MFP0	数字电路地.
50	113	129	LDO_CAP	A	MFP0	LDO 输出脚.
51	114	130	V <sub>DD</sub>	P	MFP0	I/O 端口电源和LDO 电源用于内部 PLL 和数字电路
		131	PC.14	I/O	MFP0	GPIO.
			EBI_AD11	I/O	MFP2	EBI 地址/数据总线位11.
			SC1_nCD	I	MFP3	智能卡 1 卡侦测脚.
			SPI0_I2SMCLK	I/O	MFP4	SPI0 I2S 主机时钟输出脚
			USCI0_CTL0	I/O	MFP5	USCI0 控制0 脚..
			QSPI0_CLK	I/O	MFP6	Quad SPI0 串行时钟引脚.
			EPWM0_SYNC_IN	I	MFP11	EPWM0 计数器同步触发输入脚

<b>64 脚</b>	<b>128 脚</b>	<b>144 脚</b>	引脚名称	类型	MFP	描述
			TM1	I/O	MFP13	定时器1事件计数器输入/反转输出脚.
			USB_VBUS_ST	I	MFP14	USB 外部 VBUS 管理器状态脚
			HSUSB_VBUS_ST	I	MFP15	HSUSB 外部 VBUS 管理器状态脚.
53	116	132	PB.15	I/O	MFP0	GPIO.
			EADC0_CH15	A	MFP1	EADC0 通道15模拟输入.
			EBI_AD12	I/O	MFP2	EBI 地址/数据总线位12.
			SC1_PWR	O	MFP3	智能卡 1 电源脚.
			SPI0_SS	I/O	MFP4	SPI0 从机选择脚.
			USCI0_CTL1	I/O	MFP5	USCI0 控制1 脚..
			UART0_nCTS	I	MFP6	UART0 清除发送输入脚.
			UART3_TXD	O	MFP7	UART3 数据发送脚.
			I2C2_SMBAL	O	MFP8	I2C2 SMBus SMBALTER 脚
			EPWM1_CH0	I/O	MFP11	EPWM1 通道0 输出/捕获输入脚.
			TM0_EXT	I/O	MFP13	定时器0事件计数器输入/反转输出脚.
			USB_VBUS_EN	O	MFP14	USB 外部 VBUS 管理器使能脚.
			HSUSB_VBUS_EN	O	MFP15	HSUSB 外部 VBUS 管理器使能脚.
54	117	133	PB.14	I/O	MFP0	GPIO.
			EADC0_CH14	A	MFP1	EADC0 通道14模拟输入.
			EBI_AD13	I/O	MFP2	EBI 地址/数据总线位13.
			SC1_RST	O	MFP3	智能卡 1 复位脚.
			SPI0_CLK	I/O	MFP4	SPI0 串行时钟引脚.
			USCI0_DAT1	I/O	MFP5	USCI0 数据1脚.
			UART0_nRTS	O	MFP6	UART0 请求发送输出脚.
			UART3_RXD	I	MFP7	UART3 数据接收脚.
			I2C2_SMBSUS	O	MFP8	I2C2 SMBus SMBSUS 脚(PMBus 控制脚)
			EPWM1_CH1	I/O	MFP11	EPWM1 通道1 输出/捕获输入脚.
			TM1_EXT	I/O	MFP13	定时器1事件计数器输入/反转输出脚.
			CLKO	O	MFP14	时钟输出
55	118	134	PB.13	I/O	MFP0	GPIO.

64 脚	128 脚	144 脚	引脚名称	类型	MFP	描述
56	119	135	EADC0_CH13	A	MFP1	EADC0 通道13模拟输入.
			DAC1_OUT	A	MFP1	DAC1 通道模拟输出
			ACMP0_P3	A	MFP1	模拟比较器0 正输入 3 脚.
			ACMP1_P3	A	MFP1	模拟比较器1 正输入 3 脚.
			EBI_AD14	I/O	MFP2	EBI 地址/数据总线位14.
			SC1_DAT	I/O	MFP3	智能卡1 数据脚.
			SPI0_MISO	I/O	MFP4	SPI0 MISO (主机输入, 从机输出) 脚.
			USCI0_DAT0	I/O	MFP5	USCI0 数据0脚.
			UART0_TXD	O	MFP6	UART0 数据发送脚.
			UART3_nRTS	O	MFP7	UART3 请求发送输出脚.
			I2C2_SCL	I/O	MFP8	I2C2 时钟脚.
			EPWM1_CH2	I/O	MFP11	EPWM1 通道2 输出/捕获输入脚.
			TM2_EXT	I/O	MFP13	定时器2事件计数器输入/反转输出脚.
57	120	136	PB.12	I/O	MFP0	GPIO.
			EADC0_CH12	A	MFP1	EADC0 通道12模拟输入.
			DAC0_OUT	A	MFP1	DAC0 通道模拟输出
			ACMP0_P2	A	MFP1	模拟比较器0 正输入 2 脚.
			ACMP1_P2	A	MFP1	模拟比较器1 正输入 2 脚.
			EBI_AD15	I/O	MFP2	EBI 地址/数据总线位15.
			SC1_CLK	O	MFP3	智能卡 1 时钟脚.
			SPI0_MOSI	I/O	MFP4	SPI0 MOSI (主机输出, 从机输入) 脚.
			USCI0_CLK	I/O	MFP5	USCI0 时钟脚.
			UART0_RXD	I	MFP6	UART0 数据接收脚.
			UART3_nCTS	I	MFP7	UART3 清除发送输入脚.
			I2C2_SDA	I/O	MFP8	I2C2 数据输入/输出引脚.
			SD0_nCD	I	MFP9	SD卡接口0 卡侦测输入脚
			EPWM1_CH3	I/O	MFP11	EPWM1 通道3 输出/捕获输入脚.
			TM3_EXT	I/O	MFP13	定时器3事件计数器输入/反转输出脚.
AV <sub>DD</sub>			P	MFP0	内部模拟电路电源 .	

<b>64 脚</b>	<b>128 脚</b>	<b>144 脚</b>	引脚名称	类型	MFP	描述
58	121	137	V <sub>REF</sub>	A	MFP0	ADC 参考电压输入. 注意: 该脚需要接一个 1uF 电容.
59	122	138	AV <sub>SS</sub>	P	MFP0	模拟电路地
60	123	139	PB.11	I/O	MFP0	GPIO.
			EADC0_CH11	A	MFP1	EADC0 通道11模拟输入.
			EBI_ADR16	O	MFP2	EBI 地址总线位16.
			EMAC_RMII_MDC	O	MFP3	EMAC RMII PHY 管理 时钟输出脚.
			UART0_nCTS	I	MFP5	UART0 清除发送输入脚.
			UART4_TXD	O	MFP6	UART4 数据发送脚.
			I2C1_SCL	I/O	MFP7	I2C1 时钟脚.
			CAN0_TXD	O	MFP8	CAN0 总线发送输出.
			SPI0_I2SMCLK	I/O	MFP9	SPI0 I2S 主机时钟输出脚
			BPWM1_CH0	I/O	MFP10	BPWM1 通道0 输出/捕获输入.
			SPI3_CLK	I/O	MFP11	SPI3 串行时钟引脚.
			HSUSB_VBUS_ST	I	MFP14	HSUSB 外部 VBUS 管理器状态脚.
61	124	140	PB.10	I/O	MFP0	GPIO.
			EADC0_CH10	A	MFP1	EADC0 通道10模拟输入.
			EBI_ADR17	O	MFP2	EBI 地址总线位17.
			EMAC_RMII_MDIO	I/O	MFP3	EMAC RMII PHY 管理数据输入.
			USCI1_CTL0	I/O	MFP4	USCI1 控制0 脚..
			UART0_nRTS	O	MFP5	UART0 请求发送输出脚.
			UART4_RXD	I	MFP6	UART4 数据接收脚.
			I2C1_SDA	I/O	MFP7	I2C1 数据输入/输出引脚.
			CAN0_RXD	I	MFP8	CAN0 总线接收输入
			BPWM1_CH1	I/O	MFP10	BPWM1 通道1 输出/捕获输入.
			SPI3_SS	I/O	MFP11	SPI3 从机选择脚.
			HSUSB_VBUS_EN	O	MFP14	HSUSB 外部 VBUS 管理器使能脚.
62	125	141	PB.9	I/O	MFP0	GPIO.
			EADC0_CH9	A	MFP1	EADC0 通道9模拟输入.

64 脚	128 脚	144 脚	引脚名称	类型	MFP	描述
63	126	142	EBI_ADR18	O	MFP2	EBI 地址总线位18.
			EMAC_RMII_TXD0	O	MFP3	EMAC RMII 发送数据总线位 0.
			USCI1_CTL1	I/O	MFP4	USCI1 控制1 脚..
			UART0_TXD	O	MFP5	UART0 数据发送脚.
			UART1_nCTS	I	MFP6	UART1 清除发送输入脚.
			I2C1_SMBAL	O	MFP7	I2C1 SMBus SMBALTER 脚
			BPWM1_CH2	I/O	MFP10	BPWM1 通道2 输出/捕获输入.
			SPI3_MISO	I/O	MFP11	SPI3 MISO (主机输入, 从机输出) 脚..
			INT7	I	MFP13	外部中断7输入脚.
			PB.8	I/O	MFP0	GPIO.
64	127	143	EADC0_CH8	A	MFP1	EADC0 通道8模拟输入.
			EBI_ADR19	O	MFP2	EBI 地址总线位19.
			EMAC_RMII_TXD1	O	MFP3	EMAC RMII 发送数据总线位 1.
			USCI1_CLK	I/O	MFP4	USCI1 时钟脚.
			UART0_RXD	I	MFP5	UART0 数据接收脚.
			UART1_nRTS	O	MFP6	UART1 请求发送输出脚.
			I2C1_SMBSUS	O	MFP7	I2C1 SMBus SMBSUS 脚(PMBus 控制脚)
			BPWM1_CH3	I/O	MFP10	BPWM1 通道3 输出/捕获输入.
			SPI3_MOSI	I/O	MFP11	SPI3 MOSI (主机输出, 从机输入) 脚.
			INT6	I	MFP13	外部中断6输入脚.
64	127	143	PB.7	I/O	MFP0	GPIO.
			EADC0_CH7	A	MFP1	EADC0 通道7模拟输入.
			EBI_nWRL	O	MFP2	EBI 低字节写使能输出脚.
			EMAC_RMII_TXEN	O	MFP3	EMAC RMII 发送使能 输出脚.
			USCI1_DAT0	I/O	MFP4	USCI1 数据0脚.
			CAN1_TXD	O	MFP5	CAN1 总线发送输出.
			UART1_TXD	O	MFP6	UART1 数据发送脚.
			SD1_CMD	I/O	MFP7	SD卡接口1 命令/应答脚
			EBI_nCS0	O	MFP8	EBI 片选 0 输出脚.

64 脚	128 脚	144 脚	引脚名称	类型	MFP	描述
			BPWM1_CH4	I/O	MFP10	BPWM1 通道4 输出/捕获输入.
			EPWM1_BRAKE0	I	MFP11	EPWM1 刹车0输入脚.
			EPWM1_CH4	I/O	MFP12	EPWM1 通道4 输出/捕获输入脚.
			INT5	I	MFP13	外部中断5输入脚.
			USB_VBUS_ST	I	MFP14	USB 外部 VBUS 管理器状态脚.
			ACMP0_O	O	MFP15	模拟比较器0 输出脚.
1	128	144	PB.6	I/O	MFP0	GPIO.
			EADC0_CH6	A	MFP1	EADC0 通道6模拟输入.
			EBI_nWRH	O	MFP2	EBI 高字节写使能输出脚
			EMAC_PPS	O	MFP3	EMAC 秒脉冲 输出脚.
			USCI1_DAT1	I/O	MFP4	USCI1 数据1脚.
			CAN1_RXD	I	MFP5	CAN1 总线接收输入
			UART1_RXD	I	MFP6	UART1 数据接收脚.
			SD1_CLK	O	MFP7	SD卡接口1 时钟输出脚
			EBI_nCS1	O	MFP8	EBI 片选 1 输出脚.
			BPWM1_CH5	I/O	MFP10	BPWM1 通道5 输出/捕获输入.
			EPWM1_BRAKE1	I	MFP11	EPWM1 刹车1输入脚.
			EPWM1_CH5	I/O	MFP12	EPWM1 通道5 输出/捕获输入脚.
			INT4	I	MFP13	外部中断4输入脚.
			USB_VBUS_EN	O	MFP14	USB 外部 VBUS 管理器使能脚.
			ACMP1_O	O	MFP15	模拟比较器1 输出脚.

## 4.2.7 M480 多功能管脚摘要表

组	引脚名称	GPIO	MFP	类型	描述
ACMP0	ACMP0_N	PB.3	MFP1	A	模拟比较器0 负输入脚.
	ACMP0_O	PC.12	MFP14	O	模拟比较器0 输出脚.
		PC.1	MFP14	O	
		PB.7	MFP15	O	
	ACMP0_P0	PA.11	MFP1	A	模拟比较器0 正输入 0 脚.
	ACMP0_P1	PB.2	MFP1	A	模拟比较器0 正输入 1 脚.
	ACMP0_P2	PB.12	MFP1	A	模拟比较器0 正输入 2 脚.
	ACMP0_P3	PB.13	MFP1	A	模拟比较器0 正输入 3 脚.
ACMP1	ACMP1_N	PB.5	MFP1	A	模拟比较器1 负输入脚.
	ACMP1_O	PC.11	MFP14	O	模拟比较器1 输出脚.
		PC.0	MFP14	O	
		PB.6	MFP15	O	
	ACMP1_P0	PA.10	MFP1	A	模拟比较器1 正输入 0 脚.
	ACMP1_P1	PB.4	MFP1	A	模拟比较器1 正输入 1 脚.
	ACMP1_P2	PB.12	MFP1	A	模拟比较器1 正输入 2 脚.
	ACMP1_P3	PB.13	MFP1	A	模拟比较器1 正输入 3 脚.
BPWM0	BPWM0_CH0	PA.11	MFP9	I/O	BPWM0 通道0 输出/捕获输入.
		PA.0	MFP12	I/O	
		PG.14	MFP12	I/O	
		PE.2	MFP13	I/O	
	BPWM0_CH1	PA.10	MFP9	I/O	BPWM0 通道1 输出/捕获输入.
		PA.1	MFP12	I/O	
		PG.13	MFP12	I/O	
		PE.3	MFP13	I/O	
	BPWM0_CH2	PA.9	MFP9	I/O	BPWM0 通道2 输出/捕获输入.
		PA.2	MFP12	I/O	
		PG.12	MFP12	I/O	
		PE.4	MFP13	I/O	

组	引脚名称	GPIO	MFP	类型	描述
BPWM1	BPWM0_CH3	PA.8	MFP9	I/O	BPWM0 通道3 输出/捕获输入.
		PA.3	MFP12	I/O	
		PG.11	MFP12	I/O	
		PE.5	MFP13	I/O	
	BPWM0_CH4	PC.13	MFP9	I/O	BPWM0 通道4 输出/捕获输入.
		PF.5	MFP8	I/O	
		PA.4	MFP12	I/O	
		PG.10	MFP12	I/O	
		PE.6	MFP13	I/O	
	BPWM0_CH5	PD.12	MFP9	I/O	BPWM0 通道5 输出/捕获输入.
		PF.4	MFP8	I/O	
		PA.5	MFP12	I/O	
		PG.9	MFP12	I/O	
		PE.7	MFP13	I/O	
	BPWM1_CH0	PF.3	MFP11	I/O	BPWM1 通道0 输出/捕获输入.
		PC.7	MFP12	I/O	
		PF.0	MFP12	I/O	
		PB.11	MFP10	I/O	
	BPWM1_CH1	PF.2	MFP11	I/O	BPWM1 通道1 输出/捕获输入.
		PC.6	MFP12	I/O	
		PF.1	MFP12	I/O	
		PB.10	MFP10	I/O	
	BPWM1_CH2	PA.7	MFP12	I/O	BPWM1 通道2 输出/捕获输入.
		PA.12	MFP11	I/O	
		PB.9	MFP10	I/O	
	BPWM1_CH3	PA.6	MFP12	I/O	BPWM1 通道3 输出/捕获输入.
		PA.13	MFP11	I/O	
		PB.8	MFP10	I/O	
	BPWM1_CH4	PC.8	MFP12	I/O	BPWM1 通道4 输出/捕获输入.
		PA.14	MFP11	I/O	
		PB.7	MFP10	I/O	
	BPWM1_CH5	PE.13	MFP12	I/O	BPWM1 通道5 输出/捕获输入.

组	引脚名称	GPIO	MFP	类型	描述
		PA.15	MFP11	I/O	
		PB.6	MFP10	I/O	
CAN0	CAN0_RXD	PD.10	MFP4	I	CAN0 总线接收输入
		PA.4	MFP10	I	
		PE.15	MFP4	I	
		PC.4	MFP10	I	
		PA.13	MFP6	I	
		PB.10	MFP8	I	
	CAN0_TXD	PD.11	MFP4	O	CAN0 总线发送输出.
		PA.5	MFP10	O	
		PE.14	MFP4	O	
		PC.5	MFP10	O	
		PA.12	MFP6	O	
		PB.11	MFP8	O	
CAN1	CAN1_RXD	PC.9	MFP9	I	CAN1 总线接收输入
		PD.12	MFP5	I	
		PG.1	MFP7	I	
		PC.2	MFP10	I	
		PE.6	MFP9	I	
		PB.6	MFP5	I	
	CAN1_TXD	PC.10	MFP9	O	CAN1 总线发送输出.
		PC.13	MFP5	O	
		PG.0	MFP7	O	
		PC.3	MFP10	O	
		PE.7	MFP9	O	
		PB.7	MFP5	O	
CLKO	CLKO	PC.13	MFP13	O	时钟输出
		PD.12	MFP13	O	
		PG.15	MFP14	O	
		PB.14	MFP14	O	
DAC0	DAC0_OUT	PB.12	MFP1	A	DAC0 通道模拟输出

组	引脚名称	GPIO	MFP	类型	描述
	DAC0_ST	PA.10	MFP14	I	DAC0 外部触发输入.
		PA.0	MFP15	I	
DAC1	DAC1_OUT	PB.13	MFP1	A	DAC1 通道模拟输出
	DAC1_ST	PA.11	MFP14	I	DAC1 外部触发输入.
		PA.1	MFP15	I	
EADC0	EADC0_CH0	PB.0	MFP1	A	EADC0 通道0模拟输入.
	EADC0_CH1	PB.1	MFP1	A	EADC0 通道1模拟输入.
	EADC0_CH2	PB.2	MFP1	A	EADC0 通道2模拟输入.
	EADC0_CH3	PB.3	MFP1	A	EADC0 通道3模拟输入.
	EADC0_CH4	PB.4	MFP1	A	EADC0 通道4模拟输入.
	EADC0_CH5	PB.5	MFP1	A	EADC0 通道5模拟输入.
	EADC0_CH6	PB.6	MFP1	A	EADC0 通道6模拟输入.
	EADC0_CH7	PB.7	MFP1	A	EADC0 通道7模拟输入.
	EADC0_CH8	PB.8	MFP1	A	EADC0 通道8模拟输入.
	EADC0_CH9	PB.9	MFP1	A	EADC0 通道9模拟输入.
	EADC0_CH10	PB.10	MFP1	A	EADC0 通道10模拟输入.
	EADC0_CH11	PB.11	MFP1	A	EADC0 通道11模拟输入.
	EADC0_CH12	PB.12	MFP1	A	EADC0 通道12模拟输入.
	EADC0_CH13	PB.13	MFP1	A	EADC0 通道13模拟输入.
	EADC0_CH14	PB.14	MFP1	A	EADC0 通道14模拟输入.
	EADC0_CH15	PB.15	MFP1	A	EADC0 通道15模拟输入.
	EADC0_ST	PC.13	MFP14	I	EADC0 外部触发输入.
		PD.12	MFP14	I	
		PF.5	MFP11	I	
		PG.15	MFP15	I	
EBI	EBI_AD0	PC.0	MFP2	I/O	EBI 地址/数据总线位0.
		PG.9	MFP2	I/O	
	EBI_AD1	PC.1	MFP2	I/O	EBI 地址/数据总线位1.
		PG.10	MFP2	I/O	
	EBI_AD2	PC.2	MFP2	I/O	EBI 地址/数据总线位2.

组	引脚名称	GPIO	MFP	类型	描述
		PG.11	MFP2	I/O	
EBI_AD3	PC.3	MFP2	I/O	EBI 地址/数据总线位3.	
	PG.12	MFP2	I/O		
EBI_AD4	PC.4	MFP2	I/O	EBI 地址/数据总线位4.	
	PG.13	MFP2	I/O		
EBI_AD5	PC.5	MFP2	I/O	EBI 地址/数据总线位5.	
	PG.14	MFP2	I/O		
EBI_AD6	PA.6	MFP2	I/O	EBI 地址/数据总线位6.	
	PD.8	MFP2	I/O		
EBI_AD7	PA.7	MFP2	I/O	EBI 地址/数据总线位7.	
	PD.9	MFP2	I/O		
EBI_AD8	PC.6	MFP2	I/O	EBI 地址/数据总线位8.	
	PE.14	MFP2	I/O		
EBI_AD9	PC.7	MFP2	I/O	EBI 地址/数据总线位9.	
	PE.15	MFP2	I/O		
EBI_AD10	PD.3	MFP2	I/O	EBI 地址/数据总线位10.	
	PD.13	MFP2	I/O		
	PE.1	MFP2	I/O		
EBI_AD11	PD.2	MFP2	I/O	EBI 地址/数据总线位11.	
	PE.0	MFP2	I/O		
	PC.14	MFP2	I/O		
EBI_AD12	PD.1	MFP2	I/O	EBI 地址/数据总线位12.	
	PH.8	MFP2	I/O		
	PB.15	MFP2	I/O		
EBI_AD13	PD.0	MFP2	I/O	EBI 地址/数据总线位13.	
	PH.9	MFP2	I/O		
	PB.14	MFP2	I/O		
EBI_AD14	PH.10	MFP2	I/O	EBI 地址/数据总线位14.	
	PB.13	MFP2	I/O		
EBI_AD15	PH.11	MFP2	I/O	EBI 地址/数据总线位15.	
	PB.12	MFP2	I/O		
EBI_ADR0	PB.5	MFP2	O	EBI 地址总线位0.	

组	引脚名称	GPIO	MFP	类型	描述
		PH.7	MFP2	O	
	EBI_ADR1	PB.4	MFP2	O	EBI 地址总线位1.
		PH.6	MFP2	O	
	EBI_ADR2	PB.3	MFP2	O	EBI 地址总线位2.
		PH.5	MFP2	O	
	EBI_ADR3	PB.2	MFP2	O	EBI 地址总线位3.
		PH.4	MFP2	O	
	EBI_ADR4	PC.12	MFP2	O	EBI 地址总线位4.
		PH.3	MFP2	O	
	EBI_ADR5	PC.11	MFP2	O	EBI 地址总线位5.
		PH.2	MFP2	O	
	EBI_ADR6	PC.10	MFP2	O	EBI 地址总线位 6.
		PH.1	MFP2	O	
	EBI_ADR7	PC.9	MFP2	O	EBI 地址总线位 7.
		PH.0	MFP2	O	
	EBI_ADR8	PB.1	MFP2	O	EBI 地址总线位 8.
		PG.0	MFP2	O	
	EBI_ADR9	PB.0	MFP2	O	EBI 地址总线位 9.
		PG.1	MFP2	O	
	EBI_ADR10	PC.13	MFP2	O	EBI 地址总线位10.
		PE.8	MFP2	O	
	EBI_ADR11	PG.2	MFP2	O	EBI 地址总线位11.
		PE.9	MFP2	O	
	EBI_ADR12	PG.3	MFP2	O	EBI 地址总线位12.
		PE.10	MFP2	O	
	EBI_ADR13	PG.4	MFP2	O	EBI 地址总线位13.
		PE.11	MFP2	O	
	EBI_ADR14	PF.11	MFP2	O	EBI 地址总线位14.
		PE.12	MFP2	O	
	EBI_ADR15	PF.10	MFP2	O	EBI 地址总线位15.
		PE.13	MFP2	O	
	EBI_ADR16	PF.9	MFP2	O	EBI 地址总线位16.

组	引脚名称	GPIO	MFP	类型	描述
		PC.8	MFP2	O	
		PB.11	MFP2	O	
EBI_ADR17	PF.8	MFP2	O	EBI 地址总线位17.	
		PB.10	MFP2	O	
EBI_ADR18	PF.7	MFP2	O	EBI 地址总线位18.	
		PB.9	MFP2	O	
EBI_ADR19	PF.6	MFP2	O	EBI 地址总线位19.	
		PB.8	MFP2	O	
EBI_ALE	PA.8	MFP2	O	EBI 地址锁存使能输出脚.	
		PE.2	MFP2	O	
EBI_MCLK	PA.9	MFP2	O	EBI外部时钟输出脚.	
		PE.3	MFP2	O	
EBI_nCS0	PD.12	MFP2	O	EBI 片选 0 输出脚.	
		PF.6	MFP7	O	
		PF.3	MFP2	O	
		PD.14	MFP2	O	
		PB.7	MFP8	O	
EBI_nCS1	PD.11	MFP2	O	EBI 片选 1 输出脚.	
		PF.2	MFP2	O	
		PG.5	MFP2	O	
		PB.6	MFP8	O	
EBI_nCS2	PD.10	MFP2	O	EBI 片选 2 输出脚.	
		PG.6	MFP2	O	
EBI_nRD	PA.11	MFP2	O	EBI 读使能输出脚.	
		PE.5	MFP2	O	
EBI_nWR	PA.10	MFP2	O	EBI 写使能输出脚.	
		PE.4	MFP2	O	
EBI_nWRH	PG.8	MFP2	O	EBI 高字节写使能输出脚	
		PB.6	MFP2	O	
EBI_nWRL	PG.7	MFP2	O	EBI 低字节写使能输出脚.	
		PB.7	MFP2	O	
ECAP0	ECAP0_IC0	PA.10	MFP11	I	增强型捕获输入单元0输入脚0

组	引脚名称	GPIO	MFP	类型	描述
	ECAP0_IC1	PE.8	MFP12	I	
		PA.9	MFP11	I	增强型捕获输入单元0输入脚1
	ECAP0_IC2	PE.9	MFP12	I	
		PA.8	MFP11	I	增强型捕获输入单元0输入脚2
ECAP1	ECAP1_IC0	PC.10	MFP11	I	增强型捕获输入单元1输入脚0
		PE.13	MFP13	I	
	ECAP1_IC1	PC.11	MFP11	I	增强型捕获输入单元1输入脚1
		PE.12	MFP13	I	
	ECAP1_IC2	PC.12	MFP11	I	增强型捕获输入单元1输入脚2
		PE.11	MFP13	I	
EMAC	EMAC_RMII_MDC	PE.8	MFP3	O	EMAC RMII PHY 管理 时钟输出脚.
		PB.11	MFP3	O	
	EMAC_RMII_MDIO	PE.9	MFP3	I/O	EMAC RMII PHY 管理数据脚.
		PB.10	MFP3	I/O	
	EMAC_RMII_RXD0	PB.4	MFP4	I	EMAC RMII 接收数据总线位 0.
		PC.7	MFP3	I	
	EMAC_RMII_RXD1	PB.3	MFP4	I	EMAC RMII 接收数据总线位 1.
		PC.6	MFP3	I	
	EMAC_RMII_CRSDV	PB.2	MFP4	I	EMAC RMII 载波侦听/接收数据输入脚.
		PA.7	MFP3	I	
	EMAC_RMII_RXER	PB.1	MFP4	I	EMAC RMII 接收数据错误输入脚.
		PA.6	MFP3	I	
	EMAC_RMII_TXD0	PE.10	MFP3	O	EMAC RMII 发送数据总线位 0.
		PB.9	MFP3	O	
	EMAC_RMII_TXD1	PE.11	MFP3	O	EMAC RMII 发送数据总线位 1.
		PB.8	MFP3	O	
	EMAC_RMII_TXEN	PE.12	MFP3	O	EMAC RMII 发送使能 输出脚.
		PB.7	MFP3	O	
	EMAC_PPS	PE.13	MFP3	O	EMAC 秒脉冲 输出脚.
		PB.6	MFP3	O	
	EMAC_RMII_REF	PB.5	MFP4	I	EMAC RMII 参考时钟输入脚.

组	引脚名称	GPIO	MFP	类型	描述
	LK	PC.8	MFP3	I	
EPWM0	EPWM0_BRAKE0	PB.1	MFP13	I	EPWM0 刹车0输入脚.
		PE.8	MFP11	I	
	EPWM0_BRAKE1	PB.0	MFP13	I	EPWM0 刹车1输入脚.
		PE.9	MFP11	I	
	EPWM0_CH0	PB.5	MFP11	I/O	EPWM0 通道0 输出/捕获输入脚.
		PE.8	MFP10	I/O	
		PA.5	MFP13	I/O	
		PE.7	MFP12	I/O	
		PG.8	MFP11	I/O	
	EPWM0_CH1	PB.4	MFP11	I/O	EPWM0 通道1 输出/捕获输入脚.
		PE.9	MFP10	I/O	
		PA.4	MFP13	I/O	
		PE.6	MFP12	I/O	
		PG.7	MFP11	I/O	
	EPWM0_CH2	PB.3	MFP11	I/O	EPWM0 通道2 输出/捕获输入脚.
		PE.10	MFP10	I/O	
		PA.3	MFP13	I/O	
		PE.5	MFP12	I/O	
		PG.6	MFP11	I/O	
	EPWM0_CH3	PB.2	MFP11	I/O	EPWM0 通道3 输出/捕获输入脚.
		PE.11	MFP10	I/O	
		PA.2	MFP13	I/O	
		PE.4	MFP12	I/O	
		PG.5	MFP11	I/O	
	EPWM0_CH4	PB.1	MFP11	I/O	EPWM0 通道4 输出/捕获输入脚.
		PE.12	MFP10	I/O	
		PA.1	MFP13	I/O	
		PE.3	MFP12	I/O	
		PD.14	MFP11	I/O	
	EPWM0_CH5	PB.0	MFP11	I/O	EPWM0 通道5 输出/捕获输入脚.
		PE.13	MFP10	I/O	

组	引脚名称	GPIO	MFP	类型	描述
EPWM1		PA.0	MFP13	I/O	EPWM1 通道0 输出/捕获输入脚.
		PE.2	MFP12	I/O	
		PH.11	MFP11	I/O	
	EPWM0_SYNC_IN	PA.15	MFP12	I	
		PC.14	MFP11	I	
	EPWM0_SYNC_OUT	PA.11	MFP10	O	
		PF.5	MFP9	O	
	EPWM1_BRAKE0	PE.10	MFP11	I	
		PB.7	MFP11	I	
	EPWM1_BRAKE1	PE.11	MFP11	I	
		PB.6	MFP11	I	
	EPWM1_CH0	PC.12	MFP12	I/O	
		PE.13	MFP11	I/O	
		PC.5	MFP12	I/O	
		PB.15	MFP11	I/O	
	EPWM1_CH1	PC.11	MFP12	I/O	
		PC.8	MFP11	I/O	
		PC.4	MFP12	I/O	
		PB.14	MFP11	I/O	
	EPWM1_CH2	PC.10	MFP12	I/O	
		PC.7	MFP11	I/O	
		PC.3	MFP12	I/O	
		PB.13	MFP11	I/O	
	EPWM1_CH3	PC.9	MFP12	I/O	
		PC.6	MFP11	I/O	
		PC.2	MFP12	I/O	
		PB.12	MFP11	I/O	
	EPWM1_CH4	PB.1	MFP12	I/O	
		PA.7	MFP11	I/O	
		PC.1	MFP12	I/O	
		PB.7	MFP12	I/O	
	EPWM1_CH5	PB.0	MFP12	I/O	EPWM1 通道5 输出/捕获输入脚.

组	引脚名称	GPIO	MFP	类型	描述
		PA.6	MFP11	I/O	
		PC.0	MFP12	I/O	
		PB.6	MFP12	I/O	
HSUSB	HSUSB_VBUS_EN	PB.15	MFP15	O	HSUSB 外部 VBUS 管理器使能脚.
		PB.10	MFP14	O	
	HSUSB_VBUS_ST	PC.14	MFP15	I	HSUSB 外部 VBUS 管理器状态脚.
		PB.11	MFP14	I	
I2C0	I2C0_SCL	PB.5	MFP6	I/O	I2C0 时钟脚.
		PC.12	MFP4	I/O	
		PG.0	MFP4	I/O	
		PH.2	MFP6	I/O	
		PF.3	MFP4	I/O	
		PE.13	MFP4	I/O	
		PA.5	MFP9	I/O	
		PC.1	MFP9	I/O	
	I2C0_SDA	PD.7	MFP4	I/O	I2C0 数据输入/输出引脚.
		PB.4	MFP6	I/O	
		PC.11	MFP4	I/O	
		PG.1	MFP4	I/O	
		PH.3	MFP6	I/O	
		PF.2	MFP4	I/O	
		PC.8	MFP4	I/O	
		PA.4	MFP9	I/O	
	I2C0_SMBAL	PC.0	MFP9	I/O	I2C0 SMBus SMBALTER 脚
		PD.6	MFP4	I/O	
	I2C0_SMBSUS	PG.2	MFP4	O	I2C0 SMBus SMBSUS 脚(PMBus 控制脚)
		PC.3	MFP9	O	
I2C1	I2C1_SCL	PG.3	MFP4	O	I2C1 时钟脚.
		PC.2	MFP9	O	
		PB.1	MFP9	I/O	
		PG.2	MFP5	I/O	
		PA.7	MFP8	I/O	

组	引脚名称	GPIO	MFP	类型	描述
		PA.3	MFP9	I/O	
		PF.0	MFP3	I/O	
		PC.5	MFP9	I/O	
		PD.5	MFP4	I/O	
		PA.12	MFP4	I/O	
		PE.1	MFP8	I/O	
		PB.11	MFP7	I/O	
	I2C1_SDA	PB.0	MFP9	I/O	I2C1 数据输入/输出引脚.
		PG.3	MFP5	I/O	
		PA.6	MFP8	I/O	
		PA.2	MFP9	I/O	
		PF.1	MFP3	I/O	
		PC.4	MFP9	I/O	
		PD.4	MFP4	I/O	
		PA.13	MFP4	I/O	
		PE.0	MFP8	I/O	
		PB.10	MFP7	I/O	
	I2C1_SMBAL	PG.0	MFP5	O	I2C1 SMBus SMBALTER 脚
		PC.7	MFP8	O	
		PH.8	MFP8	O	
		PB.9	MFP7	O	
	I2C1_SMBSUS	PG.1	MFP5	O	I2C1 SMBus SMBSUS 脚(PMBus 控制脚)
		PC.6	MFP8	O	
		PH.9	MFP8	O	
		PB.8	MFP7	O	
I2C2	I2C2_SCL	PA.11	MFP7	I/O	I2C2 时钟脚.
		PA.1	MFP9	I/O	
		PD.9	MFP3	I/O	
		PD.1	MFP6	I/O	
		PA.14	MFP6	I/O	
		PH.8	MFP9	I/O	
		PB.13	MFP8	I/O	

组	引脚名称	GPIO	MFP	类型	描述
I2S0	I2C2_SDA	PA.10	MFP7	I/O	I2C2 数据输入/输出引脚.
		PA.0	MFP9	I/O	
		PD.8	MFP3	I/O	
		PD.0	MFP6	I/O	
		PA.15	MFP6	I/O	
		PH.9	MFP9	I/O	
		PB.12	MFP8	I/O	
	I2C2_SMBAL	PB.15	MFP8	O	I2C2 SMBus SMBALTER 脚
	I2C2_SMBSUS	PB.14	MFP8	O	I2C2 SMBus SMBSUS 脚(PMBus 控制脚)
I2S0	I2S0_BCLK	PB.5	MFP10	O	I2S0 位时钟输出脚.
		PF.10	MFP4	O	
		PE.8	MFP4	O	
		PC.4	MFP6	O	
		PA.12	MFP2	O	
		PE.1	MFP5	O	
	I2S0_DI	PB.3	MFP10	I	I2S0 数据输入脚.
		PF.8	MFP4	I	
		PE.10	MFP4	I	
		PC.2	MFP6	I	
		PA.14	MFP2	I	
	I2S0_DO	PH.8	MFP5	I	I2S0 数据输出脚
		PB.2	MFP10	O	
		PF.7	MFP4	O	
		PE.11	MFP4	O	
		PC.1	MFP6	O	
		PA.15	MFP2	O	
	I2S0_LRCK	PH.9	MFP5	O	I2S0 左右声道选择时钟输出脚.
		PB.1	MFP10	O	
		PF.6	MFP4	O	
		PE.12	MFP4	O	
		PC.0	MFP6	O	

组	引脚名称	GPIO	MFP	类型	描述
I2S0_MCLK		PH.10	MFP5	O	I2S0 主机时钟输出脚.
		PB.4	MFP10	O	
		PF.9	MFP4	O	
		PE.9	MFP4	O	
		PC.3	MFP6	O	
		PA.13	MFP2	O	
ICE	ICE_CLK	PF.1	MFP14	I	串行调试器时钟脚.
	ICE_DAT	PF.0	MFP14	O	串行调试器数据 脚.
INT0	INT0	PB.5	MFP15	I	外部中断0输入脚.
		PA.6	MFP15	I	
INT1	INT1	PB.4	MFP15	I	外部中断1输入脚.
		PA.7	MFP15	I	
INT2	INT2	PB.3	MFP15	I	外部中断2输入脚.
		PC.6	MFP15	I	
INT3	INT3	PB.2	MFP15	I	外部中断3输入脚.
		PC.7	MFP15	I	
INT4	INT4	PA.8	MFP15	I	外部中断4输入脚.
		PB.6	MFP13	I	
INT5	INT5	PD.12	MFP15	I	外部中断5输入脚.
		PB.7	MFP13	I	
INT6	INT6	PD.11	MFP15	I	外部中断6输入脚.
		PB.8	MFP13	I	
INT7	INT7	PD.10	MFP15	I	外部中断7输入脚.
		PB.9	MFP13	I	
OPA0	OPA0_N	PB.1	MFP1	A	运放 0 负输入脚.
	OPA0_O	PB.2	MFP1	A	运放 0 输出脚.
	OPA0_P	PB.0	MFP1	A	运放0 正输入脚.
OPA1	OPA1_N	PA.9	MFP1	A	运放1 负输入端.
	OPA1_O	PA.10	MFP1	A	运放 1 输出脚.
	OPA1_P	PA.8	MFP1	A	运放1 正输入脚.

组	引脚名称	GPIO	MFP	类型	描述
OPA2	OPA2_N	PD.11	MFP1	A	运放 2 负输入脚.
	OPA2_O	PD.12	MFP1	A	运放 2 输出脚.
	OPA2_P	PD.10	MFP1	A	运放 2 正输入脚.
QEIO	QEIO_A	PD.11	MFP10	I	正交编码0 计数信号 A 相输入
		PA.4	MFP14	I	
		PE.3	MFP11	I	
	QEIO_B	PD.10	MFP10	I	正交编码0 计数信号 B 相输入
		PA.3	MFP14	I	
		PE.2	MFP11	I	
QEI1	QEI1_A	PD.12	MFP10	I	正交编码器0索引输入
		PA.5	MFP14	I	
		PE.4	MFP11	I	
	QEI1_B	PA.9	MFP10	I	正交编码1 计数信号 A 相输入
		PA.13	MFP12	I	
		PE.6	MFP11	I	
SC0	SC0_CLK	PA.8	MFP10	I	正交编码1 计数信号 B 相输入
		PA.14	MFP12	I	
		PE.5	MFP11	I	
		PA.10	MFP10	I	
	SC0_DAT	PA.12	MFP12	I	正交编码器1索引输入
		PE.7	MFP11	I	
		PB.5	MFP9	O	智能卡 0 时钟脚.
	SC0_PWR	PF.6	MFP3	O	
		PA.0	MFP6	O	
		PE.2	MFP6	O	
		PB.4	MFP9	I/O	智能卡0 数据脚.
	SC0_PWR	PF.7	MFP3	I/O	
		PA.1	MFP6	I/O	
		PE.3	MFP6	I/O	
		PB.2	MFP9	O	智能卡 0 电源脚.
		PF.9	MFP3	O	

组	引脚名称	GPIO	MFP	类型	描述
SC1		PA.3	MFP6	O	智能卡 0 复位脚.
		PE.5	MFP6	O	
	SC0_RST	PB.3	MFP9	O	
		PF.8	MFP3	O	
		PA.2	MFP6	O	
		PE.4	MFP6	O	
	SC0_nCD	PC.12	MFP9	I	
		PF.10	MFP3	I	
		PA.4	MFP6	I	
		PE.6	MFP6	I	
	SC1_CLK	PC.0	MFP5	O	智能卡 1 时钟脚.
		PD.4	MFP8	O	
		PG.8	MFP4	O	
		PB.12	MFP3	O	
	SC1_DAT	PC.1	MFP5	I/O	智能卡1 数据脚.
		PD.5	MFP8	I/O	
		PG.7	MFP4	I/O	
		PB.13	MFP3	I/O	
	SC1_PWR	PC.3	MFP5	O	智能卡 1 电源脚.
		PD.7	MFP8	O	
		PG.5	MFP4	O	
		PB.15	MFP3	O	
	SC1_RST	PC.2	MFP5	O	智能卡 1 复位脚.
		PD.6	MFP8	O	
		PG.6	MFP4	O	
		PB.14	MFP3	O	
	SC1_nCD	PC.4	MFP5	I	智能卡 1 卡侦测脚.
		PD.3	MFP8	I	
		PD.14	MFP4	I	
		PC.14	MFP3	I	
SC2	SC2_CLK	PA.8	MFP3	O	智能卡 2 时钟脚.
		PA.6	MFP6	O	

组	引脚名称	GPIO	MFP	类型	描述
SD0	SC2_DAT	PD.0	MFP7	O	智能卡2 数据脚.
		PA.15	MFP7	O	
		PE.0	MFP4	O	
	SC2_PWR	PA.9	MFP3	I/O	
		PA.7	MFP6	I/O	
		PD.1	MFP7	I/O	
		PA.14	MFP7	I/O	
		PE.1	MFP4	I/O	
	SC2_RST	PA.11	MFP3	O	智能卡2 电源脚.
		PC.7	MFP6	O	
		PD.3	MFP7	O	
		PA.12	MFP7	O	
		PH.8	MFP4	O	
	SC2_nCD	PA.10	MFP3	O	智能卡2 复位脚.
		PC.6	MFP6	O	
		PD.2	MFP7	O	
		PA.13	MFP7	O	
	SD0_CLK	PH.9	MFP4	O	智能卡2 卡侦测脚.
		PC.13	MFP3	I	
		PA.5	MFP6	I	
		PD.13	MFP7	I	
	SD0_CMD	PH.10	MFP4	I	SD卡接口0 时钟输出脚
		PB.1	MFP3	O	
	SD0_DAT0	PE.6	MFP3	O	SD卡接口0 命令/应答脚
		PB.0	MFP3	I/O	
	SD0_DAT1	PE.7	MFP3	I/O	SD卡接口0 数据位0.
		PB.2	MFP3	I/O	
	SD0_DAT2	PE.2	MFP3	I/O	SD卡接口0 数据位1.
		PB.3	MFP3	I/O	
		PE.3	MFP3	I/O	SD卡接口0 数据位2.
		PB.4	MFP3	I/O	
		PE.4	MFP3	I/O	

组	引脚名称	GPIO	MFP	类型	描述
SD1	SD0_DAT3	PB.5	MFP3	I/O	SD卡接口0 数据位3.
		PE.5	MFP3	I/O	
	SD0_nCD	PD.13	MFP3	I	SD卡接口0 卡侦测输入脚
		PB.12	MFP9	I	
QSPI0	SD1_CLK	PA.4	MFP5	O	SD卡接口1 时钟输出脚
		PG.14	MFP3	O	
		PB.6	MFP7	O	
	SD1_CMD	PA.5	MFP5	I/O	SD卡接口1 命令/应答脚
		PG.13	MFP3	I/O	
		PB.7	MFP7	I/O	
	SD1_DAT0	PA.8	MFP5	I/O	SD卡接口1 数据位0.
		PA.0	MFP5	I/O	
		PG.12	MFP3	I/O	
	SD1_DAT1	PA.9	MFP5	I/O	SD卡接口1 数据位1.
		PA.1	MFP5	I/O	
		PG.11	MFP3	I/O	
	SD1_DAT2	PA.10	MFP5	I/O	SD卡接口1 数据位2.
		PA.2	MFP5	I/O	
		PG.10	MFP3	I/O	
	SD1_DAT3	PA.11	MFP5	I/O	SD卡接口1 数据位3.
		PA.3	MFP5	I/O	
		PG.9	MFP3	I/O	
	SD1_nCD	PA.6	MFP5	I	SD卡接口1 卡侦测输入脚
		PE.14	MFP5	I	
		PG.15	MFP3	I	
QSPI0	QSPI0_CLK	PF.2	MFP5	I/O	Quad SPI0 串行时钟引脚.
		PA.2	MFP3	I/O	
		PC.2	MFP4	I/O	
		PH.8	MFP3	I/O	
		PC.14	MFP6	I/O	
QSPI0	QSPI0_MISO0	PA.1	MFP3	I/O	Quad SPI0 MISO0 (主机输入, 从机输出)脚..
		PC.1	MFP4	I/O	

组	引脚名称	GPIO	MFP	类型	描述
SPI0	QSPI0_MISO1	PE.1	MFP3	I/O	
		PA.5	MFP3	I/O	Quad SPI0 MISO1 (主机输入, 从机输出)脚..
		PC.5	MFP4	I/O	
	QSPI0_MOSI0	PH.10	MFP3	I/O	
		PA.0	MFP3	I/O	Quad SPI0 MOSI0 (主机输出, 从机输入)脚.
		PC.0	MFP4	I/O	
	QSPI0_MOSI1	PE.0	MFP3	I/O	
		PA.4	MFP3	I/O	Quad SPI0 MOSI1 (主机输出, 从机输入)脚.
		PC.4	MFP4	I/O	
	QSPI0_SS	PH.11	MFP3	I/O	
		PA.3	MFP3	I/O	Quad SPI0 从机选择脚.
		PC.3	MFP4	I/O	
		PH.9	MFP3	I/O	
SPI0	SPI0_CLK	PF.8	MFP5	I/O	SPI0 串行时钟引脚.
		PA.2	MFP4	I/O	
		PD.2	MFP4	I/O	
		PB.14	MFP4	I/O	
	SPI0_I2SMCLK	PB.0	MFP8	I/O	SPI0 I2S 主机时钟输出脚
		PF.10	MFP5	I/O	
		PA.4	MFP4	I/O	
		PD.13	MFP4	I/O	
		PC.14	MFP4	I/O	
		PB.11	MFP9	I/O	
SPI0	SPI0_MISO	PF.7	MFP5	I/O	SPI0 MISO (主机输入, 从机输出) 脚.
		PA.1	MFP4	I/O	
		PD.1	MFP4	I/O	
		PB.13	MFP4	I/O	
	SPI0_MOSI	PF.6	MFP5	I/O	SPI0 MOSI (主机输出, 从机输入) 脚.
		PA.0	MFP4	I/O	
		PD.0	MFP4	I/O	
		PB.12	MFP4	I/O	
	SPI0_SS	PF.9	MFP5	I/O	SPI0 从机选择脚.

组	引脚名称	GPIO	MFP	类型	描述
		PA.3	MFP4	I/O	
		PD.3	MFP4	I/O	
		PB.15	MFP4	I/O	
SPI1	SPI1_CLK	PB.3	MFP5	I/O	SPI1 串行时钟引脚.
		PH.6	MFP3	I/O	
		PA.7	MFP4	I/O	
		PC.1	MFP7	I/O	
		PD.5	MFP5	I/O	
		PH.8	MFP6	I/O	
	SPI1_I2SMCLK	PB.1	MFP5	I/O	SPI1 I2S 主机时钟输出脚
		PH.3	MFP3	I/O	
		PA.5	MFP4	I/O	
		PC.4	MFP7	I/O	
		PD.13	MFP5	I/O	
		PH.10	MFP6	I/O	
	SPI1_MISO	PB.5	MFP5	I/O	SPI1 MISO (主机输入, 从机输出) 脚.
		PH.4	MFP3	I/O	
		PC.7	MFP4	I/O	
		PC.3	MFP7	I/O	
		PD.7	MFP5	I/O	
		PE.1	MFP6	I/O	
	SPI1_MOSI	PB.4	MFP5	I/O	SPI1 MOSI (主机输出, 从机输入) 脚.
		PH.5	MFP3	I/O	
		PC.6	MFP4	I/O	
		PC.2	MFP7	I/O	
		PD.6	MFP5	I/O	
		PE.0	MFP6	I/O	
	SPI1_SS	PB.2	MFP5	I/O	SPI1 从机选择脚.
		PH.7	MFP3	I/O	
		PA.6	MFP4	I/O	
		PC.0	MFP7	I/O	
		PD.4	MFP5	I/O	

组	引脚名称	GPIO	MFP	类型	描述
		PH.9	MFP6	I/O	
SPI2	SPI2_CLK	PA.10	MFP4	I/O	SPI2 串行时钟引脚.
		PG.3	MFP3	I/O	
		PE.8	MFP5	I/O	
		PA.13	MFP5	I/O	
	SPI2_I2SMCLK	PC.13	MFP4	I/O	SPI2 I2S 主机时钟输出脚
		PG.1	MFP3	I/O	
		PE.12	MFP5	I/O	
	SPI2_MISO	PA.9	MFP4	I/O	SPI2 MISO (主机输入, 从机输出) 脚.
		PG.4	MFP3	I/O	
		PE.9	MFP5	I/O	
		PA.14	MFP5	I/O	
	SPI2_MOSI	PA.8	MFP4	I/O	SPI2 MOSI (主机输出, 从机输入) 脚.
		PF.11	MFP3	I/O	
		PE.10	MFP5	I/O	
		PA.15	MFP5	I/O	
	SPI2_SS	PA.11	MFP4	I/O	SPI2 从机选择脚.
		PG.2	MFP3	I/O	
		PE.11	MFP5	I/O	
		PA.12	MFP5	I/O	
SPI3	SPI3_CLK	PC.10	MFP6	I/O	SPI3 串行时钟引脚.
		PE.4	MFP5	I/O	
		PG.6	MFP3	I/O	
		PB.11	MFP11	I/O	
	SPI3_I2SMCLK	PB.1	MFP6	I/O	SPI3 I2S 主机时钟输出脚
		PE.6	MFP5	I/O	
		PD.14	MFP3	I/O	
	SPI3_MISO	PC.12	MFP6	I/O	SPI3 MISO (主机输入, 从机输出) 脚..
		PE.3	MFP5	I/O	
		PG.7	MFP3	I/O	
		PB.9	MFP11	I/O	
	SPI3_MOSI	PC.11	MFP6	I/O	SPI3 MOSI (主机输出, 从机输入) 脚.

组	引脚名称	GPIO	MFP	类型	描述
SPIM		PE.2	MFP5	I/O	SPI3 从机选择脚.
		PG.8	MFP3	I/O	
		PB.8	MFP11	I/O	
	SPI3_SS	PC.9	MFP6	I/O	
		PE.5	MFP5	I/O	
		PG.5	MFP3	I/O	
		PB.10	MFP11	I/O	
	SPIM_CLK	PA.2	MFP2	I/O	SPIM 串行时钟引脚.
		PC.2	MFP3	I/O	
		PG.12	MFP4	I/O	
		PE.4	MFP4	I/O	
	SPIM_D2	PA.5	MFP2	I/O	SPIM 四线模式数据2 .
		PC.5	MFP3	I/O	
		PG.9	MFP4	I/O	
		PE.7	MFP4	I/O	
	SPIM_D3	PA.4	MFP2	I/O	SPIM 四线模式数据3 .
		PC.4	MFP3	I/O	
		PG.10	MFP4	I/O	
		PE.6	MFP4	I/O	
	SPIM_MISO	PA.1	MFP2	I/O	SPIM MISO (主机输入, 从机输出) 脚..
		PC.1	MFP3	I/O	
		PG.13	MFP4	I/O	
		PE.3	MFP4	I/O	
	SPIM_MOSI	PA.0	MFP2	I/O	SPIM MOSI (主机输出, 从机输入) 脚.
		PC.0	MFP3	I/O	
		PG.14	MFP4	I/O	
		PE.2	MFP4	I/O	
	SPIM_SS	PA.3	MFP2	I/O	SPIM 从机选择脚.
		PC.3	MFP3	I/O	
		PG.11	MFP4	I/O	
		PE.5	MFP4	I/O	

组	引脚名称	GPIO	MFP	类型	描述
TAMPER0	TAMPER0	PF.6	MFP10	I/O	TAMPER 偷测循环脚 0.
TAMPER1	TAMPER1	PF.7	MFP10	I/O	TAMPER 偷测循环脚 1.
TAMPER2	TAMPER2	PF.8	MFP10	I/O	TAMPER 偷测循环脚 2.
TAMPER3	TAMPER3	PF.9	MFP10	I/O	TAMPER 偷测循环脚 3.
TAMPER4	TAMPER4	PF.10	MFP10	I/O	TAMPER 偷测循环脚 4.
TAMPER5	TAMPER5	PF.11	MFP10	I/O	TAMPER 偷测循环脚 5.
TM0	TM0	PB.5	MFP14	I/O	定时器0事件计数器输入/反转输出脚.
		PG.2	MFP13	I/O	
		PC.7	MFP14	I/O	
TM0_EXT	TM0_EXT	PA.11	MFP13	I/O	定时器0事件计数器输入/反转输出脚.
		PH.0	MFP13	I/O	
		PB.15	MFP13	I/O	
TM1	TM1	PB.4	MFP14	I/O	定时器1事件计数器输入/反转输出脚.
		PG.3	MFP13	I/O	
		PC.6	MFP14	I/O	
		PC.14	MFP13	I/O	
TM1_EXT	TM1_EXT	PA.10	MFP13	I/O	定时器1事件计数器输入/反转输出脚.
		PH.1	MFP13	I/O	
		PB.14	MFP13	I/O	
TM2	TM2	PB.3	MFP14	I/O	定时器2事件计数器输入/反转输出脚.
		PG.4	MFP13	I/O	
		PA.7	MFP14	I/O	
		PD.0	MFP14	I/O	
TM2_EXT	TM2_EXT	PA.9	MFP13	I/O	定时器2事件计数器输入/反转输出脚.
		PH.2	MFP13	I/O	
		PB.13	MFP13	I/O	
TM3	TM3	PB.2	MFP14	I/O	定时器3事件计数器输入/反转输出脚.
		PF.11	MFP13	I/O	
		PA.6	MFP14	I/O	
TM3_EXT	TM3_EXT	PA.8	MFP13	I/O	定时器3事件计数器输入/反转输出脚.
		PH.3	MFP13	I/O	

组	引脚名称	GPIO	MFP	类型	描述
		PB.12	MFP13	I/O	
TRACE	TRACE_CLK	PE.8	MFP14	O	ETM 追踪 时钟输出脚
	TRACE_DATA0	PE.9	MFP14	O	ETM 追踪数据 0 输出脚
	TRACE_DATA1	PE.10	MFP14	O	ETM 追踪数据 1 输出脚
	TRACE_DATA2	PE.11	MFP14	O	ETM 追踪数据 2 输出脚
	TRACE_DATA3	PE.12	MFP14	O	ETM 追踪数据 3 输出脚
UART0	UART0_RXD	PC.11	MFP3	I	UART0 数据接收脚.
		PF.2	MFP3	I	
		PA.6	MFP7	I	
		PA.0	MFP7	I	
		PD.2	MFP9	I	
		PA.15	MFP3	I	
		PH.11	MFP8	I	
		PB.12	MFP6	I	
		PB.8	MFP5	I	
	UART0_TXD	PC.12	MFP3	O	UART0 数据发送脚.
		PF.3	MFP3	O	
		PA.7	MFP7	O	
		PA.1	MFP7	O	
		PD.3	MFP9	O	
		PA.14	MFP3	O	
		PH.10	MFP8	O	
		PB.13	MFP6	O	
		PB.9	MFP5	O	
	UART0_nCTS	PC.7	MFP7	I	UART0 清除发送输入脚.
		PA.5	MFP7	I	
		PB.15	MFP6	I	
		PB.11	MFP5	I	
	UART0_nRTS	PC.6	MFP7	O	UART0 请求发送输出脚.
		PA.4	MFP7	O	
		PB.14	MFP6	O	

组	引脚名称	GPIO	MFP	类型	描述
		PB.10	MFP5	O	
UART1	UART1_RXD	PB.2	MFP6	I	UART1 数据接收脚.
		PA.8	MFP7	I	
		PD.10	MFP3	I	
		PG.1	MFP8	I	
		PC.8	MFP8	I	
		PA.2	MFP8	I	
		PF.1	MFP2	I	
		PD.6	MFP3	I	
		PH.9	MFP10	I	
		PB.6	MFP6	I	
	UART1_TXD	PB.3	MFP6	O	UART1 数据发送脚.
		PA.9	MFP7	O	
		PD.11	MFP3	O	
		PG.0	MFP8	O	
		PE.13	MFP8	O	
		PA.3	MFP8	O	
		PF.0	MFP2	O	
		PD.7	MFP3	O	
		PH.8	MFP10	O	
		PB.7	MFP6	O	
	UART1_nCTS	PE.11	MFP8	I	UART1 清除发送输入脚.
		PA.1	MFP8	I	
		PB.9	MFP6	I	
	UART1_nRTS	PE.12	MFP8	O	UART1 请求发送输出脚.
		PA.0	MFP8	O	
		PB.8	MFP6	O	
UART2	UART2_RXD	PB.0	MFP7	I	UART2 数据接收脚.
		PD.12	MFP7	I	
		PG.0	MFP6	I	
		PF.5	MFP2	I	
		PE.9	MFP7	I	

组	引脚名称	GPIO	MFP	类型	描述
UART3		PE.15	MFP3	I	UART2 数据发送脚.
		PC.4	MFP8	I	
		PC.0	MFP8	I	
	UART2_TXD	PB.1	MFP7	O	
		PC.13	MFP7	O	
		PG.1	MFP6	O	
		PF.4	MFP2	O	
		PE.8	MFP7	O	
	UART2_nCTS	PE.14	MFP3	O	
		PC.5	MFP8	O	
		PC.1	MFP8	O	
	UART2_nRTS	PF.5	MFP4	I	UART2 清除发送输入脚.
		PD.9	MFP4	I	
		PC.2	MFP8	I	
	UART3_RXD	PF.4	MFP4	O	UART2 请求发送输出脚.
		PD.8	MFP4	O	
		PC.3	MFP8	O	
	UART3_TXD	PC.9	MFP7	I	UART3 数据接收脚.
		PE.11	MFP7	I	
		PC.2	MFP11	I	
		PD.0	MFP5	I	
		PE.0	MFP7	I	
		PB.14	MFP7	I	
	UART3_nCTS	PC.10	MFP7	O	UART3 数据发送脚.
		PE.10	MFP7	O	
		PC.3	MFP11	O	
		PD.1	MFP5	O	
		PE.1	MFP7	O	
		PB.15	MFP7	O	
	UART3_nRTS	PD.2	MFP5	I	UART3 清除发送输入脚.
		PH.9	MFP7	I	
		PB.12	MFP7	I	

组	引脚名称	GPIO	MFP	类型	描述
	UART3_nRTS	PD.3	MFP5	O	UART3 请求发送输出脚.
		PH.8	MFP7	O	
		PB.13	MFP7	O	
UART4	UART4_RXD	PF.6	MFP6	I	UART4 数据接收脚.
		PH.3	MFP5	I	
		PC.6	MFP5	I	
		PA.2	MFP7	I	
		PC.4	MFP11	I	
		PA.13	MFP3	I	
		PH.11	MFP7	I	
		PB.10	MFP6	I	
	UART4_TXD	PF.7	MFP6	O	UART4 数据发送脚.
		PH.2	MFP5	O	
		PC.7	MFP5	O	
		PA.3	MFP7	O	
		PC.5	MFP11	O	
		PA.12	MFP3	O	
		PH.10	MFP7	O	
		PB.11	MFP6	O	
	UART4_nCTS	PC.8	MFP5	I	UART4 清除发送输入脚.
		PE.1	MFP9	I	
	UART4_nRTS	PE.13	MFP5	O	UART4 请求发送输出脚.
		PE.0	MFP9	O	
UART5	UART5_RXD	PB.4	MFP7	I	UART5 数据接收脚.
		PH.1	MFP4	I	
		PA.4	MFP8	I	
		PE.6	MFP8	I	
	UART5_TXD	PB.5	MFP7	O	UART5 数据发送脚.
		PH.0	MFP4	O	
		PA.5	MFP8	O	
		PE.7	MFP8	O	
	UART5_nCTS	PB.2	MFP7	I	UART5 清除发送输入脚.

组	引脚名称	GPIO	MFP	类型	描述
UART5_nRTS	PH.3	MFP4	I		
	PB.3	MFP7	O		UART5 请求发送输出脚.
	PH.2	MFP4	O		
USB	USB_D+	PA.14	MFP14	A	USB 差分信号D-.
	USB_D-	PA.13	MFP14	A	USB 差分信号D+.
	USB_OTG_ID	PA.15	MFP14	I	USB_ID.
	USB_VBUS	PA.12	MFP14	P	来自USB主机或HUB的电源
	USB_VBUS_EN	PB.15	MFP14	O	USB 外部 VBUS 管理器使能脚.
		PB.6	MFP14	O	
	USB_VBUS_ST	PD.4	MFP14	I	USB 外部 VBUS 管理器状态脚.
		PC.14	MFP14	I	
		PB.7	MFP14	I	
USCI0	USCI0_CLK	PA.11	MFP6	I/O	USCI0 时钟脚.
		PD.0	MFP3	I/O	
		PE.2	MFP7	I/O	
		PB.12	MFP5	I/O	
	USCI0_CTL0	PC.13	MFP6	I/O	USCI0 控制0 脚..
		PD.4	MFP3	I/O	
		PE.6	MFP7	I/O	
		PC.14	MFP5	I/O	
	USCI0_CTL1	PA.8	MFP6	I/O	USCI0 控制1 脚..
		PD.3	MFP3	I/O	
		PE.5	MFP7	I/O	
		PB.15	MFP5	I/O	
	USCI0_DAT0	PA.10	MFP6	I/O	USCI0 数据0脚.
		PD.1	MFP3	I/O	
		PE.3	MFP7	I/O	
		PB.13	MFP5	I/O	
	USCI0_DAT1	PA.9	MFP6	I/O	USCI0 数据1脚.
		PD.2	MFP3	I/O	
		PE.4	MFP7	I/O	

组	引脚名称	GPIO	MFP	类型	描述
		PB.14	MFP5	I/O	
USCI1	USCI1_CLK	PB.1	MFP8	I/O	USCI1 时钟脚.
		PE.12	MFP6	I/O	
		PD.7	MFP6	I/O	
		PB.8	MFP4	I/O	
	USCI1_CTL0	PB.5	MFP8	I/O	USCI1 控制0 脚..
		PE.9	MFP6	I/O	
		PD.3	MFP6	I/O	
		PB.10	MFP4	I/O	
	USCI1_CTL1	PB.4	MFP8	I/O	USCI1 控制1 脚..
		PE.8	MFP6	I/O	
		PD.4	MFP6	I/O	
		PB.9	MFP4	I/O	
	USCI1_DAT0	PB.2	MFP8	I/O	USCI1 数据0脚.
		PE.10	MFP6	I/O	
		PD.5	MFP6	I/O	
		PB.7	MFP4	I/O	
	USCI1_DAT1	PB.3	MFP8	I/O	USCI1 数据1脚.
		PE.11	MFP6	I/O	
		PD.6	MFP6	I/O	
		PB.6	MFP4	I/O	
X32	X32_IN	PF.5	MFP10	I	外部32.768 kHz 晶振输入脚.
	X32_OUT	PF.4	MFP10	O	外部32.768 kHz 晶振输出脚.
XT1	XT1_IN	PF.3	MFP10	I	外部4~24 MHz (高速) 晶振输入脚.
	XT1_OUT	PF.2	MFP10	O	外部4~24 MHz (高速) 晶振输出引脚.

## 4.2.8 M480 按GPIO分类的多功能引脚摘要表

	引脚名称	类型	MFP	描述
PA.0	PA.0	I/O	MFP0	GPIO.
	SPI_MOSI	I/O	MFP2	SPI_MOSI (主机输出, 从机输入) 脚.
	QSPI0_MOSIO	I/O	MFP3	Quad SPI0 MOSIO (主机输出, 从机输入) 脚.
	SPI0_MOSI	I/O	MFP4	SPI0 MOSI (主机输出, 从机输入) 脚.
	SD1_DAT0	I/O	MFP5	SD卡接口1 数据位0.
	SC0_CLK	O	MFP6	智能卡0 时钟脚.
	UART0_RXD	I	MFP7	UART0 数据接收脚.
	UART1_nRTS	O	MFP8	UART1 请求发送输出脚.
	I2C2_SDA	I/O	MFP9	I2C2 数据输入/输出引脚.
	BPWM0_CH0	I/O	MFP12	BPWM0 通道0 输出/捕获输入.
	EPWM0_CH5	I/O	MFP13	EPWM0 通道5 输出/捕获输入脚.
	DAC0_ST	I	MFP15	DAC0 外部触发输入.
PA.1	PA.1	I/O	MFP0	GPIO.
	SPI_MISO	I/O	MFP2	SPI_MISO (主机输入, 从机输出) 脚..
	QSPI0_MISO0	I/O	MFP3	Quad SPI0 MISO0 (主机输入, 从机输出) 脚..
	SPI0_MISO	I/O	MFP4	SPI0 MISO (主机输入, 从机输出) 脚.
	SD1_DAT1	I/O	MFP5	SD卡接口1 数据位1.
	SC0_DAT	I/O	MFP6	智能卡0 数据脚.
	UART0_TXD	O	MFP7	UART0 数据发送脚.
	UART1_nCTS	I	MFP8	UART1 清除发送输入脚.
	I2C2_SCL	I/O	MFP9	I2C2 时钟脚.
	BPWM0_CH1	I/O	MFP12	BPWM0 通道1 输出/捕获输入.
	EPWM0_CH4	I/O	MFP13	EPWM0 通道4 输出/捕获输入脚.
	DAC1_ST	I	MFP15	DAC1 外部触发输入.
PA.2	PA.2	I/O	MFP0	GPIO.
	SPI_MCLK	I/O	MFP2	SPI_MCLK 串行时钟引脚.
	QSPI0_CLK	I/O	MFP3	QSPI0_CLK 串行时钟引脚.
	SPI0_CLK	I/O	MFP4	SPI0 串行时钟引脚.
	SD1_DAT2	I/O	MFP5	SD卡接口1 数据位2.

	引脚名称	类型	MFP	描述
PA.3	SC0_RST	O	MFP6	智能卡0复位脚.
	UART4_RXD	I	MFP7	UART4数据接收脚.
	UART1_RXD	I	MFP8	UART1数据接收脚.
	I2C1_SDA	I/O	MFP9	I2C1数据输入/输出引脚.
	BPWM0_CH2	I/O	MFP12	BPWM0通道2输出/捕获输入.
	EPWM0_CH3	I/O	MFP13	EPWM0通道3输出/捕获输入脚.
PA.4	PA.3	I/O	MFP0	GPIO.
	SPI_M_SS	I/O	MFP2	SPI_M从机选择脚.
	QSPI0_SS	I/O	MFP3	Quad SPI0从机选择脚.
	SPI0_SS	I/O	MFP4	SPI0从机选择脚.
	SD1_DAT3	I/O	MFP5	SD卡接口1数据位3.
	SC0_PWR	O	MFP6	智能卡0电源脚.
	UART4_TXD	O	MFP7	UART4数据发送脚.
	UART1_TXD	O	MFP8	UART1数据发送脚.
	I2C1_SCL	I/O	MFP9	I2C1时钟脚.
	BPWM0_CH3	I/O	MFP12	BPWM0通道3输出/捕获输入.
	EPWM0_CH2	I/O	MFP13	EPWM0通道2输出/捕获输入脚.
	QEIO_B	I	MFP14	正交编码0计数信号B相输入
PA.4	PA.4	I/O	MFP0	GPIO.
	SPI_M_D3	I/O	MFP2	SPI_M四线模式数据3.
	QSPI0_MOSI1	I/O	MFP3	Quad SPI0 MOSI1(主机输出,从机输入)脚.
	SPI0_I2SMCLK	I/O	MFP4	SPI0 I2S主机时钟输出脚
	SD1_CLK	O	MFP5	SD卡接口1时钟输出脚
	SC0_nCD	I	MFP6	智能卡0卡侦测脚.
	UART0_nRTS	O	MFP7	UART0请求发送输出脚.
	UART5_RXD	I	MFP8	UART5数据接收脚.
	I2C0_SDA	I/O	MFP9	I2C0数据输入/输出引脚.
	CAN0_RXD	I	MFP10	CAN0总线接收输入
	BPWM0_CH4	I/O	MFP12	BPWM0通道4输出/捕获输入.
	EPWM0_CH1	I/O	MFP13	EPWM0通道1输出/捕获输入脚.

	引脚名称	类型	MFP	描述
	QEIO_A	I	MFP14	正交编码0 计数信号 A 相输入
PA.5	PA.5	I/O	MFP0	GPIO.
	SPI_MISO2	I/O	MFP2	SPI_MISO2 四线模式数据2 .
	QSPI0_MISO1	I/O	MFP3	Quad SPI0 MISO1 (主机输入, 从机输出) 脚..
	SPI1_I2SMCLK	I/O	MFP4	SPI1 I2S 主机时钟输出脚
	SD1_CMD	I/O	MFP5	SD卡接口1 命令/应答脚
	SC2_nCD	I	MFP6	智能卡 2 卡侦测脚.
	UART0_nCTS	I	MFP7	UART0 清除发送输入脚.
	UART5_TXD	O	MFP8	UART5 数据发送脚.
	I2C0_SCL	I/O	MFP9	I2C0 时钟脚.
	CAN0_TXD	O	MFP10	CAN0 总线发送输出.
	BPWM0_CH5	I/O	MFP12	BPWM0 通道5 输出/捕获输入.
	EPWM0_CH0	I/O	MFP13	EPWM0 通道0 输出/捕获输入脚.
	QEIO_INDEX	I	MFP14	正交编码器0索引输入
PA.6	PA.6	I/O	MFP0	GPIO.
	EBI_AD6	I/O	MFP2	EBI 地址/数据总线位6.
	EMAC_RMII_RXERR	I	MFP3	EMAC RMII 接收数据错误输入脚
	SPI1_SS	I/O	MFP4	SPI1 从机选择脚.
	SD1_nCD	I	MFP5	SD卡接口1 卡侦测输入脚
	SC2_CLK	O	MFP6	智能卡 2 时钟脚.
	UART0_RXD	I	MFP7	UART0 数据接收脚.
	I2C1_SDA	I/O	MFP8	I2C1 数据输入/输出引脚.
	EPWM1_CH5	I/O	MFP11	EPWM1 通道5 输出/捕获输入脚.
	BPWM1_CH3	I/O	MFP12	BPWM1 通道3 输出/捕获输入.
	ACMP1_WLAT	I	MFP13	模拟比较器1 窗口锁定输入脚
	TM3	I/O	MFP14	定时器3事件计数器输入/反转输出脚.
PA.7	INT0	I	MFP15	外部中断0输入脚.
	PA.7	I/O	MFP0	GPIO.
	EBI_AD7	I/O	MFP2	EBI 地址/数据总线位7.
	EMAC_RMII_CRSDV	I	MFP3	EMAC RMII 载波侦听/接收数据输入脚.

	引脚名称	类型	MFP	描述
PA.8	SPI1_CLK	I/O	MFP4	SPI1 串行时钟引脚.
	SC2_DAT	I/O	MFP6	智能卡2 数据脚.
	UART0_TXD	O	MFP7	UART0 数据发送脚.
	I2C1_SCL	I/O	MFP8	I2C1 时钟脚.
	EPWM1_CH4	I/O	MFP11	EPWM1 通道4 输出/捕获输入脚.
	BPWM1_CH2	I/O	MFP12	BPWM1 通道2 输出/捕获输入.
	ACMP0_WLAT	I	MFP13	模拟比较器0 窗口锁定输入脚
	TM2	I/O	MFP14	定时器2事件计数器输入/反转输出脚.
	INT1	I	MFP15	外部中断1输入脚.
	PA.8	I/O	MFP0	GPIO.
PA.9	OPA1_P	A	MFP1	运放1 正输入脚.
	EBI_ALE	O	MFP2	EBI 地址锁存使能输出脚.
	SC2_CLK	O	MFP3	智能卡 2 时钟脚.
	SPI2_MOSI	I/O	MFP4	SPI2 MOSI (主机输出, 从机输入) 脚.
	SD1_DAT0	I/O	MFP5	SD卡接口1 数据位0.
	USCI0_CTL1	I/O	MFP6	USCI0 控制1 脚..
	UART1_RXD	I	MFP7	UART1 数据接收脚.
	BPWM0_CH3	I/O	MFP9	BPWM0 通道3 输出/捕获输入.
	QE11_B	I	MFP10	正交编码1 计数信号 B 相输入
	ECAP0_IC2	I	MFP11	增强型捕获输入单元0输入脚2
	TM3_EXT	I/O	MFP13	定时器3事件计数器输入/反转输出脚.
	INT4	I	MFP15	外部中断4输入脚.
PA.9	PA.9	I/O	MFP0	GPIO.
	OPA1_N	A	MFP1	运放1 负输入端.
	EBI_MCLK	O	MFP2	EBI外部时钟输出脚.
	SC2_DAT	I/O	MFP3	智能卡2 数据脚.
	SPI2_MISO	I/O	MFP4	SPI2 MISO (主机输入, 从机输出) 脚.
	SD1_DAT1	I/O	MFP5	SD卡接口1 数据位1.
	USCI0_DAT1	I/O	MFP6	USCI0 数据1脚.
	UART1_TXD	O	MFP7	UART1 数据发送脚.

	引脚名称	类型	MFP	描述
PA.10	BPWM0_CH2	I/O	MFP9	BPWM0 通道2 输出/捕获输入.
	QEI1_A	I	MFP10	正交编码1 计数信号 A 相输入
	ECAP0_IC1	I	MFP11	增强型捕获输入单元0输入脚1
	TM2_EXT	I/O	MFP13	定时器2事件计数器输入/反转输出脚.
PA.11	PA.10	I/O	MFP0	GPIO.
	ACMP1_P0	A	MFP1	模拟比较器1 正输入 0 脚.
	OPA1_O	A	MFP1	运放 1 输出脚.
	EBI_nWR	O	MFP2	EBI 写使能输出脚.
	SC2_RST	O	MFP3	智能卡 2 复位脚.
	SPI2_CLK	I/O	MFP4	SPI2 串行时钟引脚.
	SD1_DAT2	I/O	MFP5	SD卡接口1 数据位2.
	USCI0_DAT0	I/O	MFP6	USCI0 数据0脚.
	I2C2_SDA	I/O	MFP7	I2C2 数据输入/输出引脚.
	BPWM0_CH1	I/O	MFP9	BPWM0 通道1 输出/捕获输入.
	QEI1_INDEX	I	MFP10	正交编码器1索引输入
	ECAP0_IC0	I	MFP11	增强型捕获输入单元0输入脚0
PA.11	TM1_EXT	I/O	MFP13	定时器1事件计数器输入/反转输出脚.
	DAC0_ST	I	MFP14	DAC0 外部触发输入.
	PA.11	I/O	MFP0	GPIO.
	ACMP0_P0	A	MFP1	模拟比较器0 正输入 0 脚.
	EBI_nRD	O	MFP2	EBI 读使能输出脚.
	SC2_PWR	O	MFP3	智能卡 2 电源脚.
	SPI2_SS	I/O	MFP4	SPI2 从机选择脚.
	SD1_DAT3	I/O	MFP5	SD卡接口1 数据位3.
	USCI0_CLK	I/O	MFP6	USCI0 时钟脚.
	I2C2_SCL	I/O	MFP7	I2C2 时钟脚.
	BPWM0_CH0	I/O	MFP9	BPWM0 通道0 输出/捕获输入.
	EPWM0_SYNC_OUT	O	MFP10	EPWM0 计数器同步触发输出脚.
PA.11	TM0_EXT	I/O	MFP13	定时器0事件计数器输入/反转输出脚.
	DAC1_ST	I	MFP14	DAC1 外部触发输入.

	引脚名称	类型	MFP	描述
PA.12	PA.12	I/O	MFP0	GPIO.
	I2S0_BCLK	O	MFP2	I2S0 位时钟输出脚.
	UART4_TXD	O	MFP3	UART4 数据发送脚.
	I2C1_SCL	I/O	MFP4	I2C1 时钟脚.
	SPI2_SS	I/O	MFP5	SPI2 从机选择脚.
	CAN0_TXD	O	MFP6	CAN0 总线发送输出.
	SC2_PWR	O	MFP7	智能卡 2 电源脚.
	BPWM1_CH2	I/O	MFP11	BPWM1 通道2 输出/捕获输入.
	QEI1_INDEX	I	MFP12	正交编码器1索引输入
	USB_VBUS	P	MFP14	来自USB主机或HUB的电源
PA.13	PA.13	I/O	MFP0	GPIO.
	I2S0_MCLK	O	MFP2	I2S0 主机时钟输出脚.
	UART4_RXD	I	MFP3	UART4 数据接收脚.
	I2C1_SDA	I/O	MFP4	I2C1 数据输入/输出引脚.
	SPI2_CLK	I/O	MFP5	SPI2 串行时钟引脚.
	CAN0_RXD	I	MFP6	CAN0 总线接收输入
	SC2_RST	O	MFP7	智能卡 2 复位脚.
	BPWM1_CH3	I/O	MFP11	BPWM1 通道3 输出/捕获输入.
	QEI1_A	I	MFP12	正交编码1 计数信号 A 相输入
	USB_D-	A	MFP14	USB 差分信号D-.
PA.14	PA.14	I/O	MFP0	GPIO.
	I2S0_DI	I	MFP2	I2S0 数据输入脚.
	UART0_TXD	O	MFP3	UART0 数据发送脚.
	SPI2_MISO	I/O	MFP5	SPI2 MISO (主机输入, 从机输出) 脚.
	I2C2_SCL	I/O	MFP6	I2C2 时钟脚.
	SC2_DAT	I/O	MFP7	智能卡2 数据脚.
	BPWM1_CH4	I/O	MFP11	BPWM1 通道4 输出/捕获输入.
	QEI1_B	I	MFP12	正交编码1 计数信号 B 相输入
	USB_D+	A	MFP14	USB 差分信号D+.
PA.15	PA.15	I/O	MFP0	GPIO.

	引脚名称	类型	MFP	描述
	I2S0_DO	O	MFP2	I2S0 数据输出脚
	UART0_RXD	I	MFP3	UART0 数据接收脚.
	SPI2_MOSI	I/O	MFP5	SPI2 MOSI (主机输出, 从机输入) 脚.
	I2C2_SDA	I/O	MFP6	I2C2 数据输入/输出引脚.
	SC2_CLK	O	MFP7	智能卡 2 时钟脚.
	BPWM1_CH5	I/O	MFP11	BPWM1 通道5 输出/捕获输入.
	EPWM0_SYNC_IN	I	MFP12	EPWM0 计数器同步触发输入脚
	USB_OTG_ID	I	MFP14	USB_ID.
PB.0	PB.0	I/O	MFP0	GPIO.
	EADC0_CH0	A	MFP1	EADC0 通道0模拟输入.
	OPA0_P	A	MFP1	运放0 正输入脚.
	EBI_ADR9	O	MFP2	EBI 地址总线位 9.
	SD0_CMD	I/O	MFP3	SD卡接口0 命令/应答脚
	UART2_RXD	I	MFP7	UART2 数据接收脚.
	SPI0_I2SMCLK	I/O	MFP8	SPI0 I2S 主机时钟输出脚
	I2C1_SDA	I/O	MFP9	I2C1 数据输入/输出引脚.
	EPWM0_CH5	I/O	MFP11	EPWM0 通道5 输出/捕获输入脚.
	EPWM1_CH5	I/O	MFP12	EPWM1 通道5 输出/捕获输入脚.
	EPWM0_BRAKE1	I	MFP13	EPWM0 刹车1输入脚.
PB.1	PB.1	I/O	MFP0	GPIO.
	EADC0_CH1	A	MFP1	EADC0 通道1模拟输入.
	OPA0_N	A	MFP1	运放0 负输入脚.
	EBI_ADR8	O	MFP2	EBI 地址总线位 8.
	SD0_CLK	O	MFP3	SD卡接口0 时钟输出脚
	EMAC_RMII_RXERR	I	MFP4	EMAC RMII 接收数据错误输入脚
	SPI1_I2SMCLK	I/O	MFP5	SPI1 I2S 主机时钟输出脚
	SPI3_I2SMCLK	I/O	MFP6	SPI3 I2S 主机时钟输出脚
	UART2_TXD	O	MFP7	UART2 数据发送脚.
	USCI1_CLK	I/O	MFP8	USCI1 时钟脚.
	I2C1_SCL	I/O	MFP9	I2C1 时钟脚.

	引脚名称	类型	MFP	描述
PB.2	I2S0_LRCK	O	MFP10	I2S0 左右声道选择时钟输出脚.
	EPWM0_CH4	I/O	MFP11	EPWM0 通道4 输出/捕获输入脚.
	EPWM1_CH4	I/O	MFP12	EPWM1 通道4 输出/捕获输入脚.
	EPWM0_BRAKE0	I	MFP13	EPWM0 刹车0输入脚.
PB.3	PB.2	I/O	MFP0	GPIO.
	EADC0_CH2	A	MFP1	EADC0 通道2模拟输入.
	ACMP0_P1	A	MFP1	模拟比较器0 正输入 1 脚.
	OPA0_O	A	MFP1	运放 0 输出脚.
	EBI_ADR3	O	MFP2	EBI 地址总线位3.
	SD0_DAT0	I/O	MFP3	SD卡接口0 数据位0.
	EMAC_RMII_CRSDV	I	MFP4	EMAC RMII 载波侦听/接收数据输入脚.
	SPI1_SS	I/O	MFP5	SPI1 从机选择脚.
	UART1_RXD	I	MFP6	UART1 数据接收脚.
	UART5_nCTS	I	MFP7	UART5 清除发送输入脚.
	USCI1_DAT0	I/O	MFP8	USCI1 数据0脚.
	SC0_PWR	O	MFP9	智能卡 0 电源脚.
	I2S0_DO	O	MFP10	I2S0 数据输出脚
	EPWM0_CH3	I/O	MFP11	EPWM0 通道3 输出/捕获输入脚.
	TM3	I/O	MFP14	定时器3事件计数器输入/反转输出脚.
	INT3	I	MFP15	外部中断3输入脚.
PB.3	PB.3	I/O	MFP0	GPIO.
	EADC0_CH3	A	MFP1	EADC0 通道3模拟输入.
	ACMP0_N	A	MFP1	模拟比较器0 负输入脚.
	EBI_ADR2	O	MFP2	EBI 地址总线位2.
	SD0_DAT1	I/O	MFP3	SD卡接口0 数据位1.
	EMAC_RMII_RXD1	I	MFP4	EMAC RMII 接收数据总线位 1.
	SPI1_CLK	I/O	MFP5	SPI1 串行时钟引脚.
	UART1_TXD	O	MFP6	UART1 数据发送脚.
	UART5_nRTS	O	MFP7	UART5 请求发送输出脚.
	USCI1_DAT1	I/O	MFP8	USCI1 数据1脚.

	引脚名称	类型	MFP	描述
PB.4	SC0_RST	O	MFP9	智能卡0复位脚.
	I2S0_DI	I	MFP10	I2S0数据输入脚.
	EPWM0_CH2	I/O	MFP11	EPWM0通道2输出/捕获输入脚.
	TM2	I/O	MFP14	定时器2事件计数器输入/反转输出脚.
	INT2	I	MFP15	外部中断2输入脚.
PB.5	PB.4	I/O	MFP0	GPIO.
	EADC0_CH4	A	MFP1	EADC0通道4模拟输入.
	ACMP1_P1	A	MFP1	模拟比较器1正输入1脚.
	EBI_ADR1	O	MFP2	EBI地址总线位1.
	SD0_DAT2	I/O	MFP3	SD卡接口0数据位2.
	EMAC_RMII_RXD0	I	MFP4	EMAC RMII接收数据总线位0.
	SPI1_MOSI	I/O	MFP5	SPI1 MOSI(主机输出,从机输入)脚.
	I2C0_SDA	I/O	MFP6	I2C0数据输入/输出引脚.
	UART5_RXD	I	MFP7	UART5数据接收脚.
	USCI1_CTL1	I/O	MFP8	USCI1控制1脚..
	SC0_DAT	I/O	MFP9	智能卡0数据脚.
	I2S0_MCLK	O	MFP10	I2S0主机时钟输出脚.
	EPWM0_CH1	I/O	MFP11	EPWM0通道1输出/捕获输入脚.
	TM1	I/O	MFP14	定时器1事件计数器输入/反转输出脚.
	INT1	I	MFP15	外部中断1输入脚.
PB.5	PB.5	I/O	MFP0	GPIO.
	EADC0_CH5	A	MFP1	EADC0通道5模拟输入.
	ACMP1_N	A	MFP1	模拟比较器1负输入脚.
	EBI_ADR0	O	MFP2	EBI地址总线位0.
	SD0_DAT3	I/O	MFP3	SD卡接口0数据位3.
	EMAC_RMII_REFCLK	I	MFP4	EMAC RMII参考时钟输入脚.
	SPI1_MISO	I/O	MFP5	SPI1 MISO(主机输入,从机输出)脚.
	I2C0_SCL	I/O	MFP6	I2C0时钟脚.
	UART5_TXD	O	MFP7	UART5数据发送脚.
	USCI1_CTL0	I/O	MFP8	USCI1控制0脚..

	引脚名称	类型	MFP	描述
PB.6	SC0_CLK	O	MFP9	智能卡0时钟脚.
	I2S0_BCLK	O	MFP10	I2S0位时钟输出脚.
	EPWM0_CH0	I/O	MFP11	EPWM0通道0输出/捕获输入脚.
	TM0	I/O	MFP14	定时器0事件计数器输入/反转输出脚.
	INT0	I	MFP15	外部中断0输入脚.
PB.7	PB.6	I/O	MFP0	GPIO.
	EADC0_CH6	A	MFP1	EADC0通道6模拟输入.
	EBI_nWRH	O	MFP2	EBI高字节写使能输出脚
	EMAC_PPS	O	MFP3	EMAC秒脉冲输出脚.
	USCI1_DAT1	I/O	MFP4	USCI1数据1脚.
	CAN1_RXD	I	MFP5	CAN1总线接收输入
	UART1_RXD	I	MFP6	UART1数据接收脚.
	SD1_CLK	O	MFP7	SD卡接口1时钟输出脚
	EBI_nCS1	O	MFP8	EBI片选1输出脚.
	BPWM1_CH5	I/O	MFP10	BPWM1通道5输出/捕获输入.
	EPWM1_BRAKE1	I	MFP11	EPWM1刹车1输入脚.
	EPWM1_CH5	I/O	MFP12	EPWM1通道5输出/捕获输入脚.
	INT4	I	MFP13	外部中断4输入脚.
	USB_VBUS_EN	O	MFP14	USB外部VBUS管理器使能脚.
	ACMP1_O	O	MFP15	模拟比较器1输出脚.
PB.7	PB.7	I/O	MFP0	GPIO.
	EADC0_CH7	A	MFP1	EADC0通道7模拟输入.
	EBI_nWRL	O	MFP2	EBI低字节写使能输出脚.
	EMAC_RMII_TXEN	O	MFP3	EMAC RMII Transmit Enable输出脚.
	USCI1_DAT0	I/O	MFP4	USCI1数据0脚.
	CAN1_TXD	O	MFP5	CAN1总线发送输出.
	UART1_TXD	O	MFP6	UART1数据发送脚.
	SD1_CMD	I/O	MFP7	SD卡接口1命令/应答脚
	EBI_nCS0	O	MFP8	EBI片选0输出脚.
	BPWM1_CH4	I/O	MFP10	BPWM1通道4输出/捕获输入.

	引脚名称	类型	MFP	描述
PB.8	EPWM1_BRAKE0	I	MFP11	EPWM1 刹车0输入脚.
	EPWM1_CH4	I/O	MFP12	EPWM1 通道4 输出/捕获输入脚.
	INT5	I	MFP13	外部中断5输入脚.
	USB_VBUS_ST	I	MFP14	USB 外部 VBUS 管理器状态脚.
	ACMP0_O	O	MFP15	模拟比较器0 输出脚.
PB.9	PB.8	I/O	MFP0	GPIO.
	EADC0_CH8	A	MFP1	EADC0 通道8模拟输入.
	EBI_ADR19	O	MFP2	EBI 地址总线位19.
	EMAC_RMII_TXD1	O	MFP3	EMAC RMII 发送数据总线位 1.
	USCI1_CLK	I/O	MFP4	USCI1 时钟脚.
	UART0_RXD	I	MFP5	UART0 数据接收脚.
	UART1_nRTS	O	MFP6	UART1 请求发送输出脚.
	I2C1_SMBSUS	O	MFP7	I2C1 SMBus SMBSUS 脚(PMBus 控制脚)
	BPWM1_CH3	I/O	MFP10	BPWM1 通道3 输出/捕获输入.
	SPI3_MOSI	I/O	MFP11	SPI3 MOSI (主机输出, 从机输入) 脚.
PB.10	INT6	I	MFP13	外部中断6输入脚.
	PB.9	I/O	MFP0	GPIO.
	EADC0_CH9	A	MFP1	EADC0 通道9模拟输入.
	EBI_ADR18	O	MFP2	EBI 地址总线位18.
	EMAC_RMII_TXD0	O	MFP3	EMAC RMII 发送数据总线位 0.
	USCI1_CTL1	I/O	MFP4	USCI1 控制1 脚..
	UART0_TXD	O	MFP5	UART0 数据发送脚.
	UART1_nCTS	I	MFP6	UART1 清除发送输入脚.
	I2C1_SMBAL	O	MFP7	I2C1 SMBus SMBALTER 脚
	BPWM1_CH2	I/O	MFP10	BPWM1 通道2 输出/捕获输入.
	SPI3_MISO	I/O	MFP11	SPI3 MISO (主机输入, 从机输出) 脚..
	INT7	I	MFP13	外部中断7输入脚.
	PB.10	I/O	MFP0	GPIO.
PB.11	EADC0_CH10	A	MFP1	EADC0 通道10模拟输入.
	EBI_ADR17	O	MFP2	EBI 地址总线位17.

	引脚名称	类型	MFP	描述
PB.11	EMAC_RMII_MDIO	I/O	MFP3	EMAC RMII PHY 管理数据脚.
	USCI1_CTL0	I/O	MFP4	USCI1 控制0 脚..
	UART0_nRTS	O	MFP5	UART0 请求发送输出脚.
	UART4_RXD	I	MFP6	UART4 数据接收脚.
	I2C1_SDA	I/O	MFP7	I2C1 数据输入/输出引脚.
	CAN0_RXD	I	MFP8	CAN0 总线接收输入
	BPWM1_CH1	I/O	MFP10	BPWM1 通道1 输出/捕获输入.
	SPI3_SS	I/O	MFP11	SPI3 从机选择脚.
	HSUSB_VBUS_EN	O	MFP14	HSUSB 外部 VBUS 管理器使能脚.
	PB.11	I/O	MFP0	GPIO.
PB.12	EADC0_CH11	A	MFP1	EADC0 通道11模拟输入.
	EBI_ADR16	O	MFP2	EBI 地址总线位16.
	EMAC_RMII_MDC	O	MFP3	EMAC RMII PHY 管理 时钟输出脚.
	UART0_nCTS	I	MFP5	UART0 清除发送输入脚.
	UART4_TXD	O	MFP6	UART4 数据发送脚.
	I2C1_SCL	I/O	MFP7	I2C1 时钟脚.
	CAN0_TXD	O	MFP8	CAN0 总线发送输出.
	SPI0_I2SMCLK	I/O	MFP9	SPI0 I2S 主机时钟输出脚
	BPWM1_CH0	I/O	MFP10	BPWM1 通道0 输出/捕获输入.
	SPI3_CLK	I/O	MFP11	SPI3 串行时钟引脚.
	HSUSB_VBUS_ST	I	MFP14	HSUSB 外部 VBUS 管理器状态脚.
PB.12	PB.12	I/O	MFP0	GPIO.
	EADC0_CH12	A	MFP1	EADC0 通道12模拟输入.
	DAC0_OUT	A	MFP1	DAC0 通道模拟输出
	ACMP0_P2	A	MFP1	模拟比较器0 正输入 2 脚.
	ACMP1_P2	A	MFP1	模拟比较器1 正输入 2 脚.
	EBI_AD15	I/O	MFP2	EBI 地址/数据总线位15.
	SC1_CLK	O	MFP3	智能卡 1 时钟脚.
	SPI0_MOSI	I/O	MFP4	SPI0 MOSI (主机输出, 从机输入) 脚.
	USCI0_CLK	I/O	MFP5	USCI0 时钟脚.

	引脚名称	类型	MFP	描述
PB.13	UART0_RXD	I	MFP6	UART0 数据接收脚.
	UART3_nCTS	I	MFP7	UART3 清除发送输入脚.
	I2C2_SDA	I/O	MFP8	I2C2 数据输入/输出引脚.
	SD0_nCD	I	MFP9	SD卡接口0 卡侦测输入脚
	EPWM1_CH3	I/O	MFP11	EPWM1 通道3 输出/捕获输入脚.
	TM3_EXT	I/O	MFP13	定时器3事件计数器输入/反转输出脚.
PB.14	PB.13	I/O	MFP0	GPIO.
	EADC0_CH13	A	MFP1	EADC0 通道13模拟输入.
	DAC1_OUT	A	MFP1	DAC1 通道模拟输出
	ACMP0_P3	A	MFP1	模拟比较器0 正输入 3 脚.
	ACMP1_P3	A	MFP1	模拟比较器1 正输入 3 脚.
	EBI_AD14	I/O	MFP2	EBI 地址/数据总线位14.
	SC1_DAT	I/O	MFP3	智能卡1 数据脚.
	SPI0_MISO	I/O	MFP4	SPI0 MISO (主机输入, 从机输出) 脚.
	USCI0_DAT0	I/O	MFP5	USCI0 数据0脚.
	UART0_TXD	O	MFP6	UART0 数据发送脚.
	UART3_nRTS	O	MFP7	UART3 请求发送输出脚.
	I2C2_SCL	I/O	MFP8	I2C2 时钟脚.
	EPWM1_CH2	I/O	MFP11	EPWM1 通道2 输出/捕获输入脚.
	TM2_EXT	I/O	MFP13	定时器2事件计数器输入/反转输出脚.
PB.15	PB.14	I/O	MFP0	GPIO.
	EADC0_CH14	A	MFP1	EADC0 通道14模拟输入.
	EBI_AD13	I/O	MFP2	EBI 地址/数据总线位13.
	SC1_RST	O	MFP3	智能卡 1 复位脚.
	SPI0_CLK	I/O	MFP4	SPI0 串行时钟引脚.
	USCI0_DAT1	I/O	MFP5	USCI0 数据1脚.
	UART0_nRTS	O	MFP6	UART0 请求发送输出脚.
	UART3_RXD	I	MFP7	UART3 数据接收脚.
	I2C2_SMBSUS	O	MFP8	I2C2 SMBus SMBSUS 脚(PMBus 控制脚)
	EPWM1_CH1	I/O	MFP11	EPWM1 通道1 输出/捕获输入脚.

	引脚名称	类型	MFP	描述
	TM1_EXT	I/O	MFP13	定时器1事件计数器输入/反转输出脚.
	CLKO	O	MFP14	时钟输出
PB.15	PB.15	I/O	MFP0	GPIO.
	EADC0_CH15	A	MFP1	EADC0 通道15模拟输入.
	EBI_AD12	I/O	MFP2	EBI 地址/数据总线位12.
	SC1_PWR	O	MFP3	智能卡 1 电源脚.
	SPI0_SS	I/O	MFP4	SPI0 从机选择脚.
	USCI0_CTL1	I/O	MFP5	USCI0 控制1 脚..
	UART0_nCTS	I	MFP6	UART0 清除发送输入脚.
	UART3_TXD	O	MFP7	UART3 数据发送脚.
	I2C2_SMBAL	O	MFP8	I2C2 SMBus SMBALTER 脚
	EPWM1_CH0	I/O	MFP11	EPWM1 通道0 输出/捕获输入脚.
	TM0_EXT	I/O	MFP13	定时器0事件计数器输入/反转输出脚.
	USB_VBUS_EN	O	MFP14	USB 外部 VBUS 管理器使能脚.
	HSUSB_VBUS_EN	O	MFP15	HSUSB 外部 VBUS 管理器使能脚.
PC.0	PC.0	I/O	MFP0	GPIO.
	EBI_AD0	I/O	MFP2	EBI 地址/数据总线位0.
	SPIM_MOSI	I/O	MFP3	SPIM MOSI (主机输出, 从机输入) 脚.
	QSPI0_MOSIO	I/O	MFP4	Quad SPI0 MOSIO (主机输出, 从机输入) 脚.
	SC1_CLK	O	MFP5	智能卡 1 时钟脚.
	I2S0_LRCK	O	MFP6	I2S0 左右声道选择时钟输出脚.
	SPI1_SS	I/O	MFP7	SPI1 从机选择脚.
	UART2_RXD	I	MFP8	UART2 数据接收脚.
	I2C0_SDA	I/O	MFP9	I2C0 数据输入/输出引脚.
	EPWM1_CH5	I/O	MFP12	EPWM1 通道5 输出/捕获输入脚.
	ACMP1_O	O	MFP14	模拟比较器1 输出脚.
PC.1	PC.1	I/O	MFP0	GPIO.
	EBI_AD1	I/O	MFP2	EBI 地址/数据总线位1.
	SPIM_MISO	I/O	MFP3	SPIM MISO (主机输入, 从机输出) 脚..
	QSPI0_MISO0	I/O	MFP4	Quad SPI0 MISO0 (主机输入, 从机输出) 脚..

	引脚名称	类型	MFP	描述
PC.2	SC1_DAT	I/O	MFP5	智能卡1 数据脚.
	I2S0_DO	O	MFP6	I2S0 数据输出脚
	SPI1_CLK	I/O	MFP7	SPI1 串行时钟引脚.
	UART2_TXD	O	MFP8	UART2 数据发送脚.
	I2C0_SCL	I/O	MFP9	I2C0 时钟脚.
	EPWM1_CH4	I/O	MFP12	EPWM1 通道4 输出/捕获输入脚.
	ACMP0_O	O	MFP14	模拟比较器0 输出脚.
	PC.2	I/O	MFP0	GPIO.
	EBI_AD2	I/O	MFP2	EBI 地址/数据总线位2.
	SPIM_CLK	I/O	MFP3	SPIM 串行时钟引脚.
	QSPI0_CLK	I/O	MFP4	Quad SPI0 串行时钟引脚.
	SC1_RST	O	MFP5	智能卡 1 复位脚.
	I2S0_DI	I	MFP6	I2S0 数据输入脚.
	SPI1_MOSI	I/O	MFP7	SPI1 MOSI (主机输出, 从机输入) 脚.
PC.3	UART2_nCTS	I	MFP8	UART2 清除发送输入脚.
	I2C0_SMBSUS	O	MFP9	I2C0 SMBus SMBSUS 脚(PMBus 控制脚)
	CAN1_RXD	I	MFP10	CAN1 总线接收输入
	UART3_RXD	I	MFP11	UART3 数据接收脚.
	EPWM1_CH3	I/O	MFP12	EPWM1 通道3 输出/捕获输入脚.
	PC.3	I/O	MFP0	GPIO.
	EBI_AD3	I/O	MFP2	EBI 地址/数据总线位3.
	SPIM_SS	I/O	MFP3	SPIM 从机选择脚.
	QSPI0_SS	I/O	MFP4	Quad SPI0 从机选择脚.
	SC1_PWR	O	MFP5	智能卡 1 电源脚.

	引脚名称	类型	MFP	描述
	EPWM1_CH2	I/O	MFP12	EPWM1 通道2 输出/捕获输入脚.
PC.4	PC.4	I/O	MFP0	GPIO.
	EBI_AD4	I/O	MFP2	EBI 地址/数据总线位4.
	SPIIM_D3	I/O	MFP3	SPIIM 四线模式数据3 .
	QSPI0_MOSI1	I/O	MFP4	Quad SPI0 MOSI1 (主机输出, 从机输入) 脚.
	SC1_nCD	I	MFP5	智能卡 1 卡侦测脚.
	I2S0_BCLK	O	MFP6	I2S0 位 时钟输出脚.
	SPI1_I2SMCLK	I/O	MFP7	SPI1 I2S 主机时钟输出脚
	UART2_RXD	I	MFP8	UART2 数据接收脚.
	I2C1_SDA	I/O	MFP9	I2C1 数据输入/输出引脚.
	CAN0_RXD	I	MFP10	CAN0 总线接收输入
PC.5	UART4_RXD	I	MFP11	UART4 数据接收脚.
	EPWM1_CH1	I/O	MFP12	EPWM1 通道1 输出/捕获输入脚.
	PC.5	I/O	MFP0	GPIO.
	EBI_AD5	I/O	MFP2	EBI 地址/数据总线位5.
	SPIIM_D2	I/O	MFP3	SPIIM 四线模式数据2 .
	QSPI0_MISO1	I/O	MFP4	Quad SPI0 MISO1 (主机输入, 从机输出) 脚..
	UART2_TXD	O	MFP8	UART2 数据发送脚.
	I2C1_SCL	I/O	MFP9	I2C1 时钟脚.
	CAN0_TXD	O	MFP10	CAN0 总线发送输出.
PC.6	UART4_TXD	O	MFP11	UART4 数据发送脚.
	EPWM1_CH0	I/O	MFP12	EPWM1 通道0 输出/捕获输入脚.
	PC.6	I/O	MFP0	GPIO.
	EBI_AD8	I/O	MFP2	EBI 地址/数据总线位8.
	EMAC_RMII_RXD1	I	MFP3	EMAC RMII 接收数据总线位 1.
	SPI1_MOSI	I/O	MFP4	SPI1 MOSI (主机输出, 从机输入) 脚.
	UART4_RXD	I	MFP5	UART4 数据接收脚.
	SC2_RST	O	MFP6	智能卡 2 复位脚.

	引脚名称	类型	MFP	描述
PC.7	EPWM1_CH3	I/O	MFP11	EPWM1 通道3 输出/捕获输入脚.
	BPWM1_CH1	I/O	MFP12	BPWM1 通道1 输出/捕获输入.
	TM1	I/O	MFP14	定时器1事件计数器输入/反转输出脚.
	INT2	I	MFP15	外部中断2输入脚.
PC.8	PC.7	I/O	MFP0	GPIO.
	EBI_AD9	I/O	MFP2	EBI 地址/数据总线位9.
	EMAC_RMII_RXD0	I	MFP3	EMAC RMII 接收数据总线位 0.
	SPI1_MISO	I/O	MFP4	SPI1 MISO (主机输入, 从机输出) 脚.
	UART4_TXD	O	MFP5	UART4 数据发送脚.
	SC2_PWR	O	MFP6	智能卡 2 电源脚.
	UART0_nCTS	I	MFP7	UART0 清除发送输入脚.
	I2C1_SMBAL	O	MFP8	I2C1 SMBus SMBALTER 脚
	EPWM1_CH2	I/O	MFP11	EPWM1 通道2 输出/捕获输入脚.
	BPWM1_CH0	I/O	MFP12	BPWM1 通道0 输出/捕获输入.
	TM0	I/O	MFP14	定时器0事件计数器输入/反转输出脚.
	INT3	I	MFP15	外部中断3输入脚.
PC.9	PC.8	I/O	MFP0	GPIO.
	EBI_ADR16	O	MFP2	EBI 地址总线位16.
	EMAC_RMII_REFCLK	I	MFP3	EMAC RMII 参考时钟输入脚.
	I2C0_SDA	I/O	MFP4	I2C0 数据输入/输出引脚.
	UART4_nCTS	I	MFP5	UART4 清除发送输入脚.
	UART1_RXD	I	MFP8	UART1 数据接收脚.
	EPWM1_CH1	I/O	MFP11	EPWM1 通道1 输出/捕获输入脚.
	BPWM1_CH4	I/O	MFP12	BPWM1 通道4 输出/捕获输入.
PC.10	PC.9	I/O	MFP0	GPIO.
	EBI_ADR7	O	MFP2	EBI 地址总线位 7.
	SPI3_SS	I/O	MFP6	SPI3 从机选择脚.
	UART3_RXD	I	MFP7	UART3 数据接收脚.
	CAN1_RXD	I	MFP9	CAN1 总线接收输入
	EPWM1_CH3	I/O	MFP12	EPWM1 通道3 输出/捕获输入脚.

	引脚名称	类型	MFP	描述
PC.10	PC.10	I/O	MFP0	GPIO.
	EBI_ADR6	O	MFP2	EBI 地址总线位 6.
	SPI3_CLK	I/O	MFP6	SPI3 串行时钟引脚.
	UART3_TXD	O	MFP7	UART3 数据发送脚.
	CAN1_TXD	O	MFP9	CAN1 总线发送输出.
	ECAP1_IC0	I	MFP11	增强型捕获输入单元1输入脚0
	EPWM1_CH2	I/O	MFP12	EPWM1 通道2 输出/捕获输入脚.
PC.11	PC.11	I/O	MFP0	GPIO.
	EBI_ADR5	O	MFP2	EBI 地址总线位5.
	UART0_RXD	I	MFP3	UART0 数据接收脚.
	I2C0_SDA	I/O	MFP4	I2C0 数据输入/输出引脚.
	SPI3_MOSI	I/O	MFP6	SPI3 MOSI (主机输出, 从机输入) 脚.
	ECAP1_IC1	I	MFP11	增强型捕获输入单元1输入脚1
	EPWM1_CH1	I/O	MFP12	EPWM1 通道1 输出/捕获输入脚.
	ACMP1_O	O	MFP14	模拟比较器1 输出脚.
PC.12	PC.12	I/O	MFP0	GPIO.
	EBI_ADR4	O	MFP2	EBI 地址总线位4.
	UART0_TXD	O	MFP3	UART0 数据发送脚.
	I2C0_SCL	I/O	MFP4	I2C0 时钟脚.
	SPI3_MISO	I/O	MFP6	SPI3 MISO (主机输入, 从机输出) 脚..
	SC0_nCD	I	MFP9	智能卡 0 卡侦测脚.
	ECAP1_IC2	I	MFP11	增强型捕获输入单元1输入脚2
	EPWM1_CH0	I/O	MFP12	EPWM1 通道0 输出/捕获输入脚.
	ACMP0_O	O	MFP14	模拟比较器0 输出脚.
PC.13	PC.13	I/O	MFP0	GPIO.
	EBI_ADR10	O	MFP2	EBI 地址总线位10.
	SC2_nCD	I	MFP3	智能卡 2 卡侦测脚.
	SPI2_I2SMCLK	I/O	MFP4	SPI2 I2S 主机时钟输出脚
	CAN1_TXD	O	MFP5	CAN1 总线发送输出.
	USCI0_CTL0	I/O	MFP6	USCI0 控制0 脚..

	引脚名称	类型	MFP	描述
PC.14	UART2_TXD	O	MFP7	UART2 数据发送脚.
	BPWM0_CH4	I/O	MFP9	BPWM0 通道4 输出/捕获输入.
	CLKO	O	MFP13	时钟输出
	EADC0_ST	I	MFP14	EADC0 外部触发输入.
PD.0	PC.14	I/O	MFP0	GPIO.
	EBI_AD11	I/O	MFP2	EBI 地址/数据总线位11.
	SC1_nCD	I	MFP3	智能卡 1 卡侦测脚.
	SPI0_I2SMCLK	I/O	MFP4	SPI0 I2S 主机时钟输出脚
	USCI0_CTL0	I/O	MFP5	USCI0 控制0 脚..
	QSPI0_CLK	I/O	MFP6	Quad SPI0 串行时钟引脚.
	EPWM0_SYNC_IN	I	MFP11	EPWM0 计数器同步触发输入脚
	TM1	I/O	MFP13	定时器1事件计数器输入/反转输出脚.
	USB_VBUS_ST	I	MFP14	USB 外部 VBUS 管理器状态脚.
	HSUSB_VBUS_ST	I	MFP15	HSUSB 外部 VBUS 管理器状态脚.
PD.1	PD.0	I/O	MFP0	GPIO.
	EBI_AD13	I/O	MFP2	EBI 地址/数据总线位13.
	USCI0_CLK	I/O	MFP3	USCI0 时钟脚.
	SPI0_MOSI	I/O	MFP4	SPI0 MOSI (主机输出, 从机输入) 脚.
	UART3_RXD	I	MFP5	UART3 数据接收脚.
	I2C2_SDA	I/O	MFP6	I2C2 数据输入/输出引脚.
	SC2_CLK	O	MFP7	智能卡 2 时钟脚.
	TM2	I/O	MFP14	定时器2事件计数器输入/反转输出脚.
PD.2	PD.1	I/O	MFP0	GPIO.
	EBI_AD12	I/O	MFP2	EBI 地址/数据总线位12.
	USCI0_DAT0	I/O	MFP3	USCI0 数据0脚.
	SPI0_MISO	I/O	MFP4	SPI0 MISO (主机输入, 从机输出) 脚.
	UART3_TXD	O	MFP5	UART3 数据发送脚.
	I2C2_SCL	I/O	MFP6	I2C2 时钟脚.
	SC2_DAT	I/O	MFP7	智能卡2 数据脚.
PD.2	PD.2	I/O	MFP0	GPIO.

	引脚名称	类型	MFP	描述
PD.3	EBI_AD11	I/O	MFP2	EBI 地址/数据总线位11.
	USCI0_DAT1	I/O	MFP3	USCI0 数据1脚.
	SPI0_CLK	I/O	MFP4	SPI0 串行时钟引脚.
	UART3_nCTS	I	MFP5	UART3 清除发送输入脚.
	SC2_RST	O	MFP7	智能卡 2 复位脚.
	UART0_RXD	I	MFP9	UART0 数据接收脚.
PD.4	PD.3	I/O	MFP0	GPIO.
	EBI_AD10	I/O	MFP2	EBI 地址/数据总线位10.
	USCI0_CTL1	I/O	MFP3	USCI0 控制1 脚..
	SPI0_SS	I/O	MFP4	SPI0 从机选择脚.
	UART3_nRTS	O	MFP5	UART3 请求发送输出脚.
	USCI1_CTL0	I/O	MFP6	USCI1 控制0 脚..
	SC2_PWR	O	MFP7	智能卡 2 电源脚.
	SC1_nCD	I	MFP8	智能卡 1 卡侦测脚.
	UART0_TXD	O	MFP9	UART0 数据发送脚.
PD.5	PD.4	I/O	MFP0	GPIO.
	USCI0_CTL0	I/O	MFP3	USCI0 控制0 脚..
	I2C1_SDA	I/O	MFP4	I2C1 数据输入/输出引脚.
	SPI1_SS	I/O	MFP5	SPI1 从机选择脚.
	USCI1_CTL1	I/O	MFP6	USCI1 控制1 脚..
	SC1_CLK	O	MFP8	智能卡 1 时钟脚.
	USB_VBUS_ST	I	MFP14	USB 外部 VBUS 管理器状态脚.
PD.6	PD.5	I/O	MFP0	GPIO.
	I2C1_SCL	I/O	MFP4	I2C1 时钟脚.
	SPI1_CLK	I/O	MFP5	SPI1 串行时钟引脚.
	USCI1_DAT0	I/O	MFP6	USCI1 数据0脚.
	SC1_DAT	I/O	MFP8	智能卡1 数据脚.
PD.7	PD.6	I/O	MFP0	GPIO.
	UART1_RXD	I	MFP3	UART1 数据接收脚.
	I2C0_SDA	I/O	MFP4	I2C0 数据输入/输出引脚.

	引脚名称	类型	MFP	描述
PD.7	SPI1_MOSI	I/O	MFP5	SPI1 MOSI (主机输出, 从机输入) 脚.
	USCI1_DAT1	I/O	MFP6	USCI1 数据1脚.
	SC1_RST	O	MFP8	智能卡 1 复位脚.
PD.8	PD.7	I/O	MFP0	GPIO.
	UART1_TXD	O	MFP3	UART1 数据发送脚.
	I2C0_SCL	I/O	MFP4	I2C0 时钟脚.
	SPI1_MISO	I/O	MFP5	SPI1 MISO (主机输入, 从机输出) 脚.
	USCI1_CLK	I/O	MFP6	USCI1 时钟脚.
	SC1_PWR	O	MFP8	智能卡 1 电源脚.
PD.9	PD.8	I/O	MFP0	GPIO.
	EBI_AD6	I/O	MFP2	EBI 地址/数据总线位6.
	I2C2_SDA	I/O	MFP3	I2C2 数据输入/输出引脚.
	UART2_nRTS	O	MFP4	UART2 请求发送输出脚.
PD.10	PD.9	I/O	MFP0	GPIO.
	EBI_AD7	I/O	MFP2	EBI 地址/数据总线位7.
	I2C2_SCL	I/O	MFP3	I2C2 时钟脚.
	UART2_nCTS	I	MFP4	UART2 清除发送输入脚.
	PD.10	I/O	MFP0	GPIO.
	OPA2_P	A	MFP1	运放 2 正输入脚.
	EBI_nCS2	O	MFP2	EBI 片选 2 输出脚.
PD.11	UART1_RXD	I	MFP3	UART1 数据接收脚.
	CAN0_RXD	I	MFP4	CAN0 总线接收输入
	QEI0_B	I	MFP10	正交编码0 计数信号 B 相输入
	INT7	I	MFP15	外部中断7输入脚.
	PD.11	I/O	MFP0	GPIO.
	OPA2_N	A	MFP1	运放 2 负输入脚.
PD.12	EBI_nCS1	O	MFP2	EBI 片选 1 输出脚.
	UART1_TXD	O	MFP3	UART1 数据发送脚.
	CAN0_TXD	O	MFP4	CAN0 总线发送输出.
	QEI0_A	I	MFP10	正交编码0 计数信号 A 相输入

	引脚名称	类型	MFP	描述
	INT6	I	MFP15	外部中断6输入脚.
PD.12	PD.12	I/O	MFP0	GPIO.
	OPA2_O	A	MFP1	运放 2 输出脚.
	EBI_nCS0	O	MFP2	EBI 片选 0 输出脚.
	CAN1_RXD	I	MFP5	CAN1 总线接收输入
	UART2_RXD	I	MFP7	UART2 数据接收脚.
	BPWM0_CH5	I/O	MFP9	BPWM0 通道5 输出/捕获输入.
	QEIO_INDEX	I	MFP10	正交编码器0索引输入
	CLKO	O	MFP13	时钟输出
	EADC0_ST	I	MFP14	EADC0 外部触发输入.
	INT5	I	MFP15	外部中断5输入脚.
PD.13	PD.13	I/O	MFP0	GPIO.
	EBI_AD10	I/O	MFP2	EBI 地址/数据总线位10.
	SD0_nCD	I	MFP3	SD卡接口0 卡侦测输入脚
	SPI0_I2SMCLK	I/O	MFP4	SPI0 I2S 主机时钟输出脚
	SPI1_I2SMCLK	I/O	MFP5	SPI1 I2S 主机时钟输出脚
	SC2_nCD	I	MFP7	智能卡 2 卡侦测脚.
PD.14	PD.14	I/O	MFP0	GPIO.
	EBI_nCS0	O	MFP2	EBI 片选 0 输出脚.
	SPI3_I2SMCLK	I/O	MFP3	SPI3 I2S 主机时钟输出脚
	SC1_nCD	I	MFP4	智能卡 1 卡侦测脚.
	EPWM0_CH4	I/O	MFP11	EPWM0 通道4 输出/捕获输入脚.
PE.0	PE.0	I/O	MFP0	GPIO.
	EBI_AD11	I/O	MFP2	EBI 地址/数据总线位11.
	QSPI0_MOSI0	I/O	MFP3	Quad SPI0 MOSI0 (主机输出, 从机输入) 脚.
	SC2_CLK	O	MFP4	智能卡 2 时钟脚.
	I2S0_MCLK	O	MFP5	I2S0 主机时钟输出脚.
	SPI1_MOSI	I/O	MFP6	SPI1 MOSI (主机输出, 从机输入) 脚.
	UART3_RXD	I	MFP7	UART3 数据接收脚.
	I2C1_SDA	I/O	MFP8	I2C1 数据输入/输出引脚.

	引脚名称	类型	MFP	描述
	UART4_nRTS	O	MFP9	UART4 请求发送输出脚.
PE.1	PE.1	I/O	MFP0	GPIO.
	EBI_AD10	I/O	MFP2	EBI 地址/数据总线位10.
	QSPI0_MISO0	I/O	MFP3	Quad SPI0 MISO0 (主机输入, 从机输出) 脚..
	SC2_DAT	I/O	MFP4	智能卡2 数据脚.
	I2S0_BCLK	O	MFP5	I2S0 位时钟输出脚.
	SPI1_MISO	I/O	MFP6	SPI1 MISO (主机输入, 从机输出) 脚.
	UART3_TXD	O	MFP7	UART3 数据发送脚.
	I2C1_SCL	I/O	MFP8	I2C1 时钟脚.
	UART4_nCTS	I	MFP9	UART4 清除发送输入脚.
PE.2	PE.2	I/O	MFP0	GPIO.
	EBI_ALE	O	MFP2	EBI 地址锁存使能输出脚.
	SD0_DAT0	I/O	MFP3	SD卡接口0 数据位0.
	SPIM_MOSI	I/O	MFP4	SPIM MOSI (主机输出, 从机输入) 脚.
	SPI3_MOSI	I/O	MFP5	SPI3 MOSI (主机输出, 从机输入) 脚.
	SC0_CLK	O	MFP6	智能卡0 时钟脚.
	USCI0_CLK	I/O	MFP7	USCI0 时钟脚.
	QEIO_B	I	MFP11	正交编码0 计数信号B相输入
	EPWM0_CH5	I/O	MFP12	EPWM0 通道5 输出/捕获输入脚.
	BPWM0_CH0	I/O	MFP13	BPWM0 通道0 输出/捕获输入.
PE.3	PE.3	I/O	MFP0	GPIO.
	EBI_MCLK	O	MFP2	EBI外部时钟输出脚.
	SD0_DAT1	I/O	MFP3	SD卡接口0 数据位1.
	SPIM_MISO	I/O	MFP4	SPIM MISO (主机输入, 从机输出) 脚..
	SPI3_MISO	I/O	MFP5	SPI3 MISO (主机输入, 从机输出) 脚..
	SC0_DAT	I/O	MFP6	智能卡0 数据脚.
	USCI0_DAT0	I/O	MFP7	USCI0 数据0脚.
	QEIO_A	I	MFP11	正交编码0 计数信号A相输入
	EPWM0_CH4	I/O	MFP12	EPWM0 通道4 输出/捕获输入脚.
	BPWM0_CH1	I/O	MFP13	BPWM0 通道1 输出/捕获输入.

	引脚名称	类型	MFP	描述
PE.4	PE.4	I/O	MFP0	GPIO.
	EBI_nWR	O	MFP2	EBI 写使能输出脚.
	SD0_DAT2	I/O	MFP3	SD卡接口0 数据位2.
	SPIM_CLK	I/O	MFP4	SPIM 串行时钟引脚.
	SPI3_CLK	I/O	MFP5	SPI3 串行时钟引脚.
	SC0_RST	O	MFP6	智能卡 0 复位脚.
	USCI0_DAT1	I/O	MFP7	USCI0 数据1脚.
	QEI0_INDEX	I	MFP11	正交编码器0索引输入
	EPWM0_CH3	I/O	MFP12	EPWM0 通道3 输出/捕获输入脚.
	BPWM0_CH2	I/O	MFP13	BPWM0 通道2 输出/捕获输入.
PE.5	PE.5	I/O	MFP0	GPIO.
	EBI_nRD	O	MFP2	EBI 读使能输出脚.
	SD0_DAT3	I/O	MFP3	SD卡接口0 数据位3.
	SPIM_SS	I/O	MFP4	SPIM 从机选择脚.
	SPI3_SS	I/O	MFP5	SPI3 从机选择脚.
	SC0_PWR	O	MFP6	智能卡 0 电源脚.
	USCI0_CTL1	I/O	MFP7	USCI0 控制1 脚..
	QEI1_B	I	MFP11	正交编码1 计数信号 B 相输入
	EPWM0_CH2	I/O	MFP12	EPWM0 通道2 输出/捕获输入脚.
	BPWM0_CH3	I/O	MFP13	BPWM0 通道3 输出/捕获输入.
PE.6	PE.6	I/O	MFP0	GPIO.
	SD0_CLK	O	MFP3	SD卡接口0 时钟输出脚
	SPIM_D3	I/O	MFP4	SPIM 四线模式数据3 .
	SPI3_I2SMCLK	I/O	MFP5	SPI3 I2S 主机时钟输出脚
	SC0_nCD	I	MFP6	智能卡 0 卡侦测脚.
	USCI0_CTL0	I/O	MFP7	USCI0 控制0 脚..
	UART5_RXD	I	MFP8	UART5 数据接收脚.
	CAN1_RXD	I	MFP9	CAN1 总线接收输入
	QEI1_A	I	MFP11	正交编码1 计数信号 A 相输入
	EPWM0_CH1	I/O	MFP12	EPWM0 通道1 输出/捕获输入脚.

	引脚名称	类型	MFP	描述
	BPWM0_CH4	I/O	MFP13	BPWM0 通道4 输出/捕获输入.
PE.7	PE.7	I/O	MFP0	GPIO.
	SD0_CMD	I/O	MFP3	SD卡接口0 命令/应答脚
	SPI_M_D2	I/O	MFP4	SPI_M 四线模式数据2 .
	UART5_TXD	O	MFP8	UART5 数据发送脚.
	CAN1_TXD	O	MFP9	CAN1 总线发送输出.
	QEII_INDEX	I	MFP11	正交编码器1索引输入
	EPWM0_CH0	I/O	MFP12	EPWM0 通道0 输出/捕获输入脚.
	BPWM0_CH5	I/O	MFP13	BPWM0 通道5 输出/捕获输入.
PE.8	PE.8	I/O	MFP0	GPIO.
	EBI_ADR10	O	MFP2	EBI 地址总线位10.
	EMAC_RMII_MDC	O	MFP3	EMAC RMII PHY 管理 时钟输出脚.
	I2S0_BCLK	O	MFP4	I2S0 位 时钟输出脚.
	SPI2_CLK	I/O	MFP5	SPI2 串行时钟引脚.
	USCI1_CTL1	I/O	MFP6	USCI1 控制1 脚..
	UART2_TXD	O	MFP7	UART2 数据发送脚.
	EPWM0_CH0	I/O	MFP10	EPWM0 通道0 输出/捕获输入脚.
	EPWM0_BRAKE0	I	MFP11	EPWM0 刹车0输入脚.
	ECAPO_IC0	I	MFP12	增强型捕获输入单元0输入脚0
PE.9	TRACE_CLK	O	MFP14	ETM 追踪 时钟输出脚
	PE.9	I/O	MFP0	GPIO.
	EBI_ADR11	O	MFP2	EBI 地址总线位11.
	EMAC_RMII_MDIO	I/O	MFP3	EMAC RMII PHY 管理数据脚.
	I2S0_MCLK	O	MFP4	I2S0 主机时钟输出脚.
	SPI2_MISO	I/O	MFP5	SPI2 MISO (主机输入, 从机输出) 脚.
	USCI1_CTL0	I/O	MFP6	USCI1 控制0 脚..
	UART2_RXD	I	MFP7	UART2 数据接收脚.
	EPWM0_CH1	I/O	MFP10	EPWM0 通道1 输出/捕获输入脚.
	EPWM0_BRAKE1	I	MFP11	EPWM0 刹车1输入脚.
	ECAPO_IC1	I	MFP12	增强型捕获输入单元0输入脚1

	引脚名称	类型	MFP	描述
	TRACE_DATA0	O	MFP14	ETM 追踪数据 0 输出脚
PE.10	PE.10	I/O	MFP0	GPIO.
	EBI_ADR12	O	MFP2	EBI 地址总线位12.
	EMAC_RMII_TXD0	O	MFP3	EMAC RMII 发送数据总线位 0.
	I2S0_DI	I	MFP4	I2S0 数据输入脚.
	SPI2_MOSI	I/O	MFP5	SPI2 MOSI (主机输出, 从机输入) 脚.
	USCI1_DAT0	I/O	MFP6	USCI1 数据0脚.
	UART3_TXD	O	MFP7	UART3 数据发送脚.
	EPWM0_CH2	I/O	MFP10	EPWM0 通道2 输出/捕获输入脚.
	EPWM1_BRAKE0	I	MFP11	EPWM1 刹车0输入脚.
	ECAP0_IC2	I	MFP12	增强型捕获输入单元0输入脚2
	TRACE_DATA1	O	MFP14	ETM 追踪数据 1 输出脚
PE.11	PE.11	I/O	MFP0	GPIO.
	EBI_ADR13	O	MFP2	EBI 地址总线位13.
	EMAC_RMII_TXD1	O	MFP3	EMAC RMII 发送数据总线位 1.
	I2S0_DO	O	MFP4	I2S0 数据输出脚
	SPI2_SS	I/O	MFP5	SPI2 从机选择脚.
	USCI1_DAT1	I/O	MFP6	USCI1 数据1脚.
	UART3_RXD	I	MFP7	UART3 数据接收脚.
	UART1_nCTS	I	MFP8	UART1 清除发送输入脚.
	EPWM0_CH3	I/O	MFP10	EPWM0 通道3 输出/捕获输入脚.
	EPWM1_BRAKE1	I	MFP11	EPWM1 刹车1输入脚.
PE.12	ECAP1_IC2	I	MFP13	增强型捕获输入单元1输入脚2
	TRACE_DATA2	O	MFP14	ETM 追踪数据 2 输出脚
	PE.12	I/O	MFP0	GPIO.
	EBI_ADR14	O	MFP2	EBI 地址总线位14.
	EMAC_RMII_TXEN	O	MFP3	EMAC RMII 发送使能 输出脚.
	I2S0_LRCK	O	MFP4	I2S0 左右声道选择时钟输出脚.
	SPI2_I2SMCLK	I/O	MFP5	SPI2 I2S 主机时钟输出脚
	USCI1_CLK	I/O	MFP6	USCI1 时钟脚.

	引脚名称	类型	MFP	描述
PE.13	UART1_nRTS	O	MFP8	UART1 请求发送输出脚.
	EPWM0_CH4	I/O	MFP10	EPWM0 通道4 输出/捕获输入脚.
	ECAP1_IC1	I	MFP13	增强型捕获输入单元1输入脚1
	TRACE_DATA3	O	MFP14	ETM 追踪数据 3 输出脚
PE.14	PE.13	I/O	MFP0	GPIO.
	EBI_ADR15	O	MFP2	EBI 地址总线位15.
	EMAC_PPS	O	MFP3	EMAC 秒脉冲 输出脚.
	I2C0_SCL	I/O	MFP4	I2C0 时钟脚.
	UART4_nRTS	O	MFP5	UART4 请求发送输出脚.
	UART1_TXD	O	MFP8	UART1 数据发送脚.
	EPWM0_CH5	I/O	MFP10	EPWM0 通道5 输出/捕获输入脚.
	EPWM1_CH0	I/O	MFP11	EPWM1 通道0 输出/捕获输入脚.
	BPWM1_CH5	I/O	MFP12	BPWM1 通道5 输出/捕获输入.
	ECAP1_IC0	I	MFP13	增强型捕获输入单元1输入脚0
PE.15	PE.14	I/O	MFP0	GPIO.
	EBI_AD8	I/O	MFP2	EBI 地址/数据总线位8.
	UART2_TXD	O	MFP3	UART2 数据发送脚.
	CAN0_TXD	O	MFP4	CAN0 总线发送输出.
	SD1_nCD	I	MFP5	SD卡接口1 卡侦测输入脚
PF.0	PE.15	I/O	MFP0	GPIO.
	EBI_AD9	I/O	MFP2	EBI 地址/数据总线位9.
	UART2_RXD	I	MFP3	UART2 数据接收脚.
	CAN0_RXD	I	MFP4	CAN0 总线接收输入
PF.1	PF.0	I/O	MFP0	GPIO.
	UART1_TXD	O	MFP2	UART1 数据发送脚.
	I2C1_SCL	I/O	MFP3	I2C1 时钟脚.
	BPWM1_CH0	I/O	MFP12	BPWM1 通道0 输出/捕获输入.
	ICE_DAT	O	MFP14	串行调试器数据 脚.
PF.1	PF.1	I/O	MFP0	GPIO.
	UART1_RXD	I	MFP2	UART1 数据接收脚.

	引脚名称	类型	MFP	描述
PF.2	I2C1_SDA	I/O	MFP3	I2C1 数据输入/输出引脚.
	BPWM1_CH1	I/O	MFP12	BPWM1 通道1 输出/捕获输入.
	ICE_CLK	I	MFP14	串行调试器时钟脚.
PF.3	PF.2	I/O	MFP0	GPIO.
	EBI_nCS1	O	MFP2	EBI 片选 1 输出脚.
	UART0_RXD	I	MFP3	UART0 数据接收脚.
	I2C0_SDA	I/O	MFP4	I2C0 数据输入/输出引脚.
	QSPI0_CLK	I/O	MFP5	Quad SPI0 串行时钟引脚.
	XT1_OUT	O	MFP10	外部4~24 MHz (高速) 晶振输出引脚.
	BPWM1_CH1	I/O	MFP11	BPWM1 通道1 输出/捕获输入.
PF.4	PF.3	I/O	MFP0	GPIO.
	EBI_nCS0	O	MFP2	EBI 片选 0 输出脚.
	UART0_TXD	O	MFP3	UART0 数据发送脚.
	I2C0_SCL	I/O	MFP4	I2C0 时钟脚.
	XT1_IN	I	MFP10	外部4~24 MHz (高速) 晶振输入脚.
PF.5	PF.4	I/O	MFP0	GPIO.
	UART2_TXD	O	MFP2	UART2 数据发送脚.
	UART2_nRTS	O	MFP4	UART2 请求发送输出脚.
	BPWM0_CH5	I/O	MFP8	BPWM0 通道5 输出/捕获输入.
	X32_OUT	O	MFP10	外部32.768 kHz 晶振输出脚.
PF.6	PF.5	I/O	MFP0	GPIO.
	UART2_RXD	I	MFP2	UART2 数据接收脚.
	UART2_nCTS	I	MFP4	UART2 清除发送输入脚.
	BPWM0_CH4	I/O	MFP8	BPWM0 通道4 输出/捕获输入.
	EPWM0_SYNC_OUT	O	MFP9	EPWM0 计数器同步触发输出脚.
	X32_IN	I	MFP10	外部32.768 kHz 晶振输入脚.
	EADC0_ST	I	MFP11	EADC0 外部触发输入.
PF.6	PF.6	I/O	MFP0	GPIO.
	EBI_ADR19	O	MFP2	EBI 地址总线位19.

	引脚名称	类型	MFP	描述
PF.8	SC0_CLK	O	MFP3	智能卡 0 时钟脚.
	I2S0_LRCK	O	MFP4	I2S0 左右声道选择时钟输出脚.
	SPI0_MOSI	I/O	MFP5	SPI0 MOSI (主机输出, 从机输入) 脚.
	UART4_RXD	I	MFP6	UART4 数据接收脚.
	EBI_nCS0	O	MFP7	EBI 片选 0 输出脚.
	TAMPER0	I/O	MFP10	TAMPER 倾测循环脚 0.
PF.9	PF.7	I/O	MFP0	GPIO.
	EBI_ADR18	O	MFP2	EBI 地址总线位18.
	SC0_DAT	I/O	MFP3	智能卡0 数据脚.
	I2S0_DO	O	MFP4	I2S0 数据输出脚
	SPI0_MISO	I/O	MFP5	SPI0 MISO (主机输入, 从机输出) 脚.
	UART4_TXD	O	MFP6	UART4 数据发送脚.
PF.10	TAMPER1	I/O	MFP10	TAMPER 倾测循环脚 1.
	PF.8	I/O	MFP0	GPIO.
	EBI_ADR17	O	MFP2	EBI 地址总线位17.
	SC0_RST	O	MFP3	智能卡 0 复位脚.
	I2S0_DI	I	MFP4	I2S0 数据输入脚.
	SPI0_CLK	I/O	MFP5	SPI0 串行时钟引脚.
PF.11	TAMPER2	I/O	MFP10	TAMPER 倾测循环脚 2.
	PF.9	I/O	MFP0	GPIO.
	EBI_ADR16	O	MFP2	EBI 地址总线位16.
	SC0_PWR	O	MFP3	智能卡 0 电源脚.
	I2S0_MCLK	O	MFP4	I2S0 主机时钟输出脚.
	SPI0_SS	I/O	MFP5	SPI0 从机选择脚.
PF.12	TAMPER3	I/O	MFP10	TAMPER 倾测循环脚 3.
	PF.10	I/O	MFP0	GPIO.
	EBI_ADR15	O	MFP2	EBI 地址总线位15.
	SC0_nCD	I	MFP3	智能卡 0 卡倾测脚.
	I2S0_BCLK	O	MFP4	I2S0 位 时钟输出脚.
PF.13	SPI0_I2SMCLK	I/O	MFP5	SPI0 I2S 主机时钟输出脚

	引脚名称	类型	MFP	描述
	TAMPER4	I/O	MFP10	TAMPER 侦测循环脚 4.
PF.11	PF.11	I/O	MFP0	GPIO.
	EBI_ADR14	O	MFP2	EBI 地址总线位14.
	SPI2_MOSI	I/O	MFP3	SPI2 MOSI (主机输出, 从机输入) 脚.
	TAMPER5	I/O	MFP10	TAMPER 侦测循环脚 5.
	TM3	I/O	MFP13	定时器3事件计数器输入/反转输出脚.
PG.0	PG.0	I/O	MFP0	GPIO.
	EBI_ADR8	O	MFP2	EBI 地址总线位 8.
	I2C0_SCL	I/O	MFP4	I2C0 时钟脚.
	I2C1_SMBAL	O	MFP5	I2C1 SMBus SMBALTER 脚
	UART2_RXD	I	MFP6	UART2 数据接收脚.
	CAN1_TXD	O	MFP7	CAN1 总线发送输出.
	UART1_TXD	O	MFP8	UART1 数据发送脚.
PG.1	PG.1	I/O	MFP0	GPIO.
	EBI_ADR9	O	MFP2	EBI 地址总线位 9.
	SPI2_I2SMCLK	I/O	MFP3	SPI2 I2S 主机时钟输出脚
	I2C0_SDA	I/O	MFP4	I2C0 数据输入/输出引脚.
	I2C1_SMBSUS	O	MFP5	I2C1 SMBus SMBSUS 脚(PMBus 控制脚)
	UART2_TXD	O	MFP6	UART2 数据发送脚.
	CAN1_RXD	I	MFP7	CAN1 总线接收输入
	UART1_RXD	I	MFP8	UART1 数据接收脚.
PG.2	PG.2	I/O	MFP0	GPIO.
	EBI_ADR11	O	MFP2	EBI 地址总线位11.
	SPI2_SS	I/O	MFP3	SPI2 从机选择脚.
	I2C0_SMBAL	O	MFP4	I2C0 SMBus SMBALTER 脚
	I2C1_SCL	I/O	MFP5	I2C1 时钟脚.
	TM0	I/O	MFP13	定时器0事件计数器输入/反转输出脚.
PG.3	PG.3	I/O	MFP0	GPIO.
	EBI_ADR12	O	MFP2	EBI 地址总线位12.
	SPI2_CLK	I/O	MFP3	SPI2 串行时钟引脚.

	引脚名称	类型	MFP	描述
PG.4	I2C0_SMBSUS	O	MFP4	I2C0 SMBus SMBSUS 脚(PMBus 控制脚)
	I2C1_SDA	I/O	MFP5	I2C1 数据输入/输出引脚.
	TM1	I/O	MFP13	定时器1事件计数器输入/反转输出脚.
PG.5	PG.4	I/O	MFP0	GPIO.
	EBI_ADR13	O	MFP2	EBI 地址总线位13.
	SPI2_MISO	I/O	MFP3	SPI2 MISO (主机输入, 从机输出) 脚.
	TM2	I/O	MFP13	定时器2事件计数器输入/反转输出脚.
PG.6	PG.5	I/O	MFP0	GPIO.
	EBI_nCS1	O	MFP2	EBI 片选 1 输出脚.
	SPI3_SS	I/O	MFP3	SPI3 从机选择脚.
	SC1_PWR	O	MFP4	智能卡 1 电源脚.
	EPWM0_CH3	I/O	MFP11	EPWM0 通道3 输出/捕获输入脚.
PG.7	PG.6	I/O	MFP0	GPIO.
	EBI_nCS2	O	MFP2	EBI 片选 2 输出脚.
	SPI3_CLK	I/O	MFP3	SPI3 串行时钟引脚.
	SC1_RST	O	MFP4	智能卡 1 复位脚.
	EPWM0_CH2	I/O	MFP11	EPWM0 通道2 输出/捕获输入脚.
PG.8	PG.7	I/O	MFP0	GPIO.
	EBI_nWRL	O	MFP2	EBI 低字节写使能输出脚.
	SPI3_MISO	I/O	MFP3	SPI3 MISO (主机输入, 从机输出) 脚..
	SC1_DAT	I/O	MFP4	智能卡1 数据脚.
	EPWM0_CH1	I/O	MFP11	EPWM0 通道1 输出/捕获输入脚.
PG.9	PG.8	I/O	MFP0	GPIO.
	EBI_nWRH	O	MFP2	EBI 高字节写使能输出脚
	SPI3_MOSI	I/O	MFP3	SPI3 MOSI (主机输出, 从机输入) 脚.
	SC1_CLK	O	MFP4	智能卡 1 时钟脚.
	EPWM0_CH0	I/O	MFP11	EPWM0 通道0 输出/捕获输入脚.
PG.10	PG.9	I/O	MFP0	GPIO.
	EBI_AD0	I/O	MFP2	EBI 地址/数据总线位0.
	SD1_DAT3	I/O	MFP3	SD卡接口1 数据位3.

	引脚名称	类型	MFP	描述
	SPI_M_D2	I/O	MFP4	SPI_M 四线模式数据2 .
	BPWM0_CH5	I/O	MFP12	BPWM0 通道5 输出/捕获输入.
PG.10	PG.10	I/O	MFP0	GPIO.
	EBI_AD1	I/O	MFP2	EBI 地址/数据总线位1.
	SD1_DAT2	I/O	MFP3	SD卡接口1 数据位2.
	SPI_M_D3	I/O	MFP4	SPI_M 四线模式数据3 .
	BPWM0_CH4	I/O	MFP12	BPWM0 通道4 输出/捕获输入.
PG.11	PG.11	I/O	MFP0	GPIO.
	EBI_AD2	I/O	MFP2	EBI 地址/数据总线位2.
	SD1_DAT1	I/O	MFP3	SD卡接口1 数据位1.
	SPI_M_SS	I/O	MFP4	SPI_M 从机选择脚.
	BPWM0_CH3	I/O	MFP12	BPWM0 通道3 输出/捕获输入.
PG.12	PG.12	I/O	MFP0	GPIO.
	EBI_AD3	I/O	MFP2	EBI 地址/数据总线位3.
	SD1_DAT0	I/O	MFP3	SD卡接口1 数据位0.
	SPI_M_CLK	I/O	MFP4	SPI_M 串行时钟引脚.
	BPWM0_CH2	I/O	MFP12	BPWM0 通道2 输出/捕获输入.
PG.13	PG.13	I/O	MFP0	GPIO.
	EBI_AD4	I/O	MFP2	EBI 地址/数据总线位4.
	SD1_CMD	I/O	MFP3	SD卡接口1 命令/应答脚
	SPI_M_MISO	I/O	MFP4	SPI_M MISO (主机输入, 从机输出) 脚..
	BPWM0_CH1	I/O	MFP12	BPWM0 通道1 输出/捕获输入.
PG.14	PG.14	I/O	MFP0	GPIO.
	EBI_AD5	I/O	MFP2	EBI 地址/数据总线位5.
	SD1_CLK	O	MFP3	SD卡接口1 时钟输出脚
	SPI_M_MOSI	I/O	MFP4	SPI_M MOSI (主机输出, 从机输入) 脚.
	BPWM0_CH0	I/O	MFP12	BPWM0 通道0 输出/捕获输入.
PG.15	PG.15	I/O	MFP0	GPIO.
	SD1_nCD	I	MFP3	SD卡接口1 卡侦测输入脚
	CLKO	O	MFP14	时钟输出

	引脚名称	类型	MFP	描述
	EADC0_ST	I	MFP15	EADC0 外部触发输入.
PH.0	PH.0	I/O	MFP0	GPIO.
	EBI_ADR7	O	MFP2	EBI 地址总线位 7.
	UART5_TXD	O	MFP4	UART5 数据发送脚.
	TM0_EXT	I/O	MFP13	定时器0事件计数器输入/反转输出脚.
PH.1	PH.1	I/O	MFP0	GPIO.
	EBI_ADR6	O	MFP2	EBI 地址总线位 6.
	UART5_RXD	I	MFP4	UART5 数据接收脚.
	TM1_EXT	I/O	MFP13	定时器1事件计数器输入/反转输出脚.
PH.2	PH.2	I/O	MFP0	GPIO.
	EBI_ADR5	O	MFP2	EBI 地址总线位5.
	UART5_nRTS	O	MFP4	UART5 请求发送输出脚.
	UART4_TXD	O	MFP5	UART4 数据发送脚.
	I2C0_SCL	I/O	MFP6	I2C0 时钟脚.
	TM2_EXT	I/O	MFP13	定时器2事件计数器输入/反转输出脚.
PH.3	PH.3	I/O	MFP0	GPIO.
	EBI_ADR4	O	MFP2	EBI 地址总线位4.
	SPI1_I2SMCLK	I/O	MFP3	SPI1 I2S 主机时钟输出脚
	UART5_nCTS	I	MFP4	UART5 清除发送输入脚.
	UART4_RXD	I	MFP5	UART4 数据接收脚.
	I2C0_SDA	I/O	MFP6	I2C0 数据输入/输出引脚.
	TM3_EXT	I/O	MFP13	定时器3事件计数器输入/反转输出脚.
PH.4	PH.4	I/O	MFP0	GPIO.
	EBI_ADR3	O	MFP2	EBI 地址总线位3.
	SPI1_MISO	I/O	MFP3	SPI1 MISO (主机输入, 从机输出) 脚.
PH.5	PH.5	I/O	MFP0	GPIO.
	EBI_ADR2	O	MFP2	EBI 地址总线位2.
	SPI1_MOSI	I/O	MFP3	SPI1 MOSI (主机输出, 从机输入) 脚.
PH.6	PH.6	I/O	MFP0	GPIO.
	EBI_ADR1	O	MFP2	EBI 地址总线位1.

	引脚名称	类型	MFP	描述
	SPI1_CLK	I/O	MFP3	SPI1 串行时钟引脚.
PH.7	PH.7	I/O	MFP0	GPIO.
	EBI_ADR0	O	MFP2	EBI 地址总线位0.
	SPI1_SS	I/O	MFP3	SPI1 从机选择脚.
PH.8	PH.8	I/O	MFP0	GPIO.
	EBI_AD12	I/O	MFP2	EBI 地址/数据总线位12.
	QSPI0_CLK	I/O	MFP3	Quad SPI0 串行时钟引脚.
	SC2_PWR	O	MFP4	智能卡 2 电源脚.
	I2S0_DI	I	MFP5	I2S0 数据输入脚.
	SPI1_CLK	I/O	MFP6	SPI1 串行时钟引脚.
	UART3_nRTS	O	MFP7	UART3 请求发送输出脚.
	I2C1_SMBAL	O	MFP8	I2C1 SMBus SMBALTER 脚
	I2C2_SCL	I/O	MFP9	I2C2 时钟脚.
	UART1_TXD	O	MFP10	UART1 数据发送脚.
PH.9	PH.9	I/O	MFP0	GPIO.
	EBI_AD13	I/O	MFP2	EBI 地址/数据总线位13.
	QSPI0_SS	I/O	MFP3	Quad SPI0 从机选择脚.
	SC2_RST	O	MFP4	智能卡 2 复位脚.
	I2S0_DO	O	MFP5	I2S0 数据输出脚
	SPI1_SS	I/O	MFP6	SPI1 从机选择脚.
	UART3_nCTS	I	MFP7	UART3 清除发送输入脚.
	I2C1_SMBSUS	O	MFP8	I2C1 SMBus SMBSUS 脚(PMBus 控制脚)
	I2C2_SDA	I/O	MFP9	I2C2 数据输入/输出引脚.
	UART1_RXD	I	MFP10	UART1 数据接收脚.
PH.10	PH.10	I/O	MFP0	GPIO.
	EBI_ADI4	I/O	MFP2	EBI 地址/数据总线位14.
	QSPI0_MISO1	I/O	MFP3	Quad SPI0 MISO1 (主机输入, 从机输出) 脚..
	SC2_nCD	I	MFP4	智能卡 2 卡侦测脚.
	I2S0_LRCK	O	MFP5	I2S0 左右声道选择时钟输出脚.
	SPI1_I2SMCLK	I/O	MFP6	SPI1 I2S 主机时钟输出脚

	引脚名称	类型	MFP	描述
	UART4_TXD	O	MFP7	UART4 数据发送脚.
	UART0_TXD	O	MFP8	UART0 数据发送脚.
PH.11	PH.11	I/O	MFP0	GPIO.
	EBI_AD15	I/O	MFP2	EBI 地址/数据总线位15.
	QSPI0_MOSI1	I/O	MFP3	Quad SPI0 MOSI1 (主机输出, 从机输入) 脚.
	UART4_RXD	I	MFP7	UART4 数据接收脚.
	UART0_RXD	I	MFP8	UART0 数据接收脚.
	EPWM0_CH5	I/O	MFP11	EPWM0 通道5 输出/捕获输入脚.

## 5 框图

### 5.1 NuMicro® M480 框图

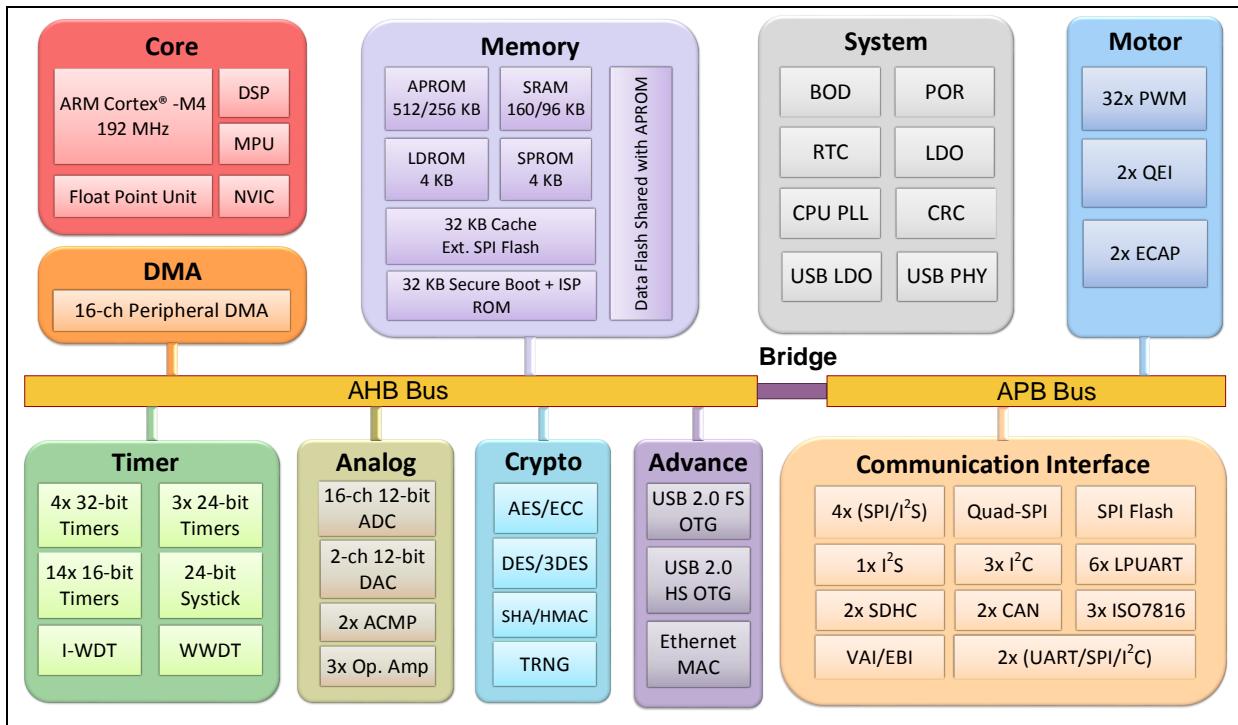


图 5.1-1 NuMicro® M480 框图

## 6 功能描述

### 6.1 ARM® Cortex®-M4 内核

Cortex®-M4 处理器是多级流水线、32位精简指令处理器，三个 AMBA AHB-Lite 接口达到最佳的并行处理，包含一个 NVIC 中断向量控制器。处理器有一个可选的硬件调试单元，可执行与其它 Cortex-M 处理器兼容的 Rhumb 指令。两种处理模式：线程模式 Thread 和句柄模式 Handler。进入异常时就进入句柄模式——异常只能在这种模式下返回。复位后是线程模式，异常返回后也可进入线程模式。M480 的内核是 Cortex®-M4F，又多了一个浮点运算功能。请参考 Cortex®-M4 和 Cortex®-M4F 处理器。表 6.1-1 是功能框图。

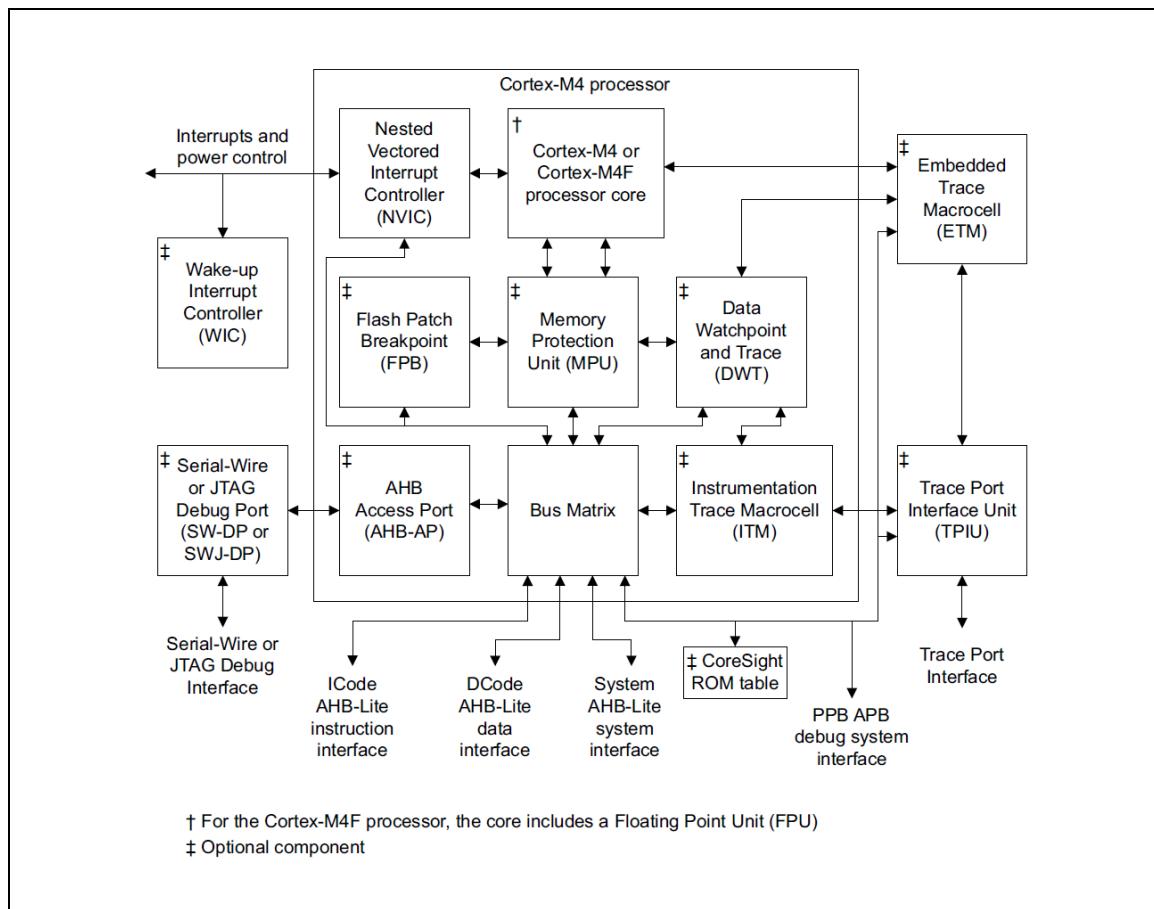


表 6.1-1 Cortex®-M4 框图

**Cortex®-M4 处理器特征:**

- 低逻辑站数量, 加速中断处理功能:
  - Thumb指令集, 请参考 *ARMv7-M Architecture Reference Manual*
  - Banked 栈指针
  - 有除法指令, SDIV , UDIV
  - 句柄和线程模式
  - Thumb 指令和调试功能
  - 支持中断/继续指令 LDM, STM, PUSH, 和 POP
  - 中断上下文保存/退出恢复, 参考 *Interrupt Service Routine (ISR) entry and exit*
  - 支持 ARMv6 大端和小端数据访问
  - 支持ARMv6非对齐访问
- Cortex®-M4F的浮点单元(FPU) 功能:
  - 单精度浮点运算, 32位指令
  - 组合了乘/累加用于增加精度(用 MAC)
  - 硬件支持转换, 加, 减, 带累加的乘, 除, 和平方根运算
  - 支持 IEEE 环绕模式
  - 32个32位寄存器, 可组成16个双字寄存器-
  - 三级流水线
- NVIC可嵌套中断向量控制器, 与处理器紧密接合, 加速中断响应时间, 功能包括:
  - 外中断可配置为 1 ~ 240
  - 优先先级可以有 3 ~ 8 位
  - 优先级可动态改变
  - 可配置优先级组, 组内相互之间不抢先
  - 支持尾链和高优先级先处理功能.
  - 硬件入栈上下文程状态寄存器
  - 支持唤醒功能
- MPU内存保护单元:
  - 可管理8个内存区域
  - 分区可使能禁止读写功能
  - 其它区域有缺省属性
- 调试单元:
  - 可调试访问所有内存空间和寄存器.
  - SWD接口或 JTAG 接口
  - 可选的FLASH断点功能 FPB

- 可选的数据观察和跟踪单元 (DWT)
  - 可选的指令跟踪单元 ITM 支持 printf() 风格的调试
  - 可选的跟踪单元 TPIU 可用于跟踪分析, 包含单线输出接口模式
  - 可选的嵌入跟踪单元 ETM 可跟踪指令
- 总线接口:
    - 三条AHB-Lite总线: 指令总线, 数据总线, 系统总线
    - 基于 APB 总线的各外设单独总线 PPB
    - 支持位段读写.
    - 内存访问对齐
    - 写缓存
    - 多处理器系统独立传输

## 6.2 系统管理

### 6.2.1 概述

系统管理包括以下部分：

- 系统复位
- 系统电源分布
- SRAM 内存结构
- 系统定时器(SysTick)
- 可嵌套中断向量控制(NVIC)
- 系统控制寄存器

### 6.2.2 系统复位

下面事件可以触发系统复位，可以通过读取SYS\_RSTSTS 寄存器以知道复位源是哪个。硬件复位源来自外设信号，软件复位通过设置控制寄存器触发。

- 硬件复位源
  - 上电复位POR
  - 引脚 nRESET 复位
  - WDT/WWDT 复位
  - 低压复位(LVR)
  - 欠压复位 BOD
  - CPU锁死复位 (Lockup)
- 软件复位源
  - 芯片复位：CHIPRST (SYS\_IPRST0[0])写1，复位整个芯片。
  - MCU 复位，SYSRESETREQ (AIRCR[2])写1，复位CPU和外设，从APROM 或 LDROM 启动不变。
  - CPU 复位，CPURST (SYS\_IPRST0[1])写1，仅复位 Cortex®-M4 内核，外设不复位。

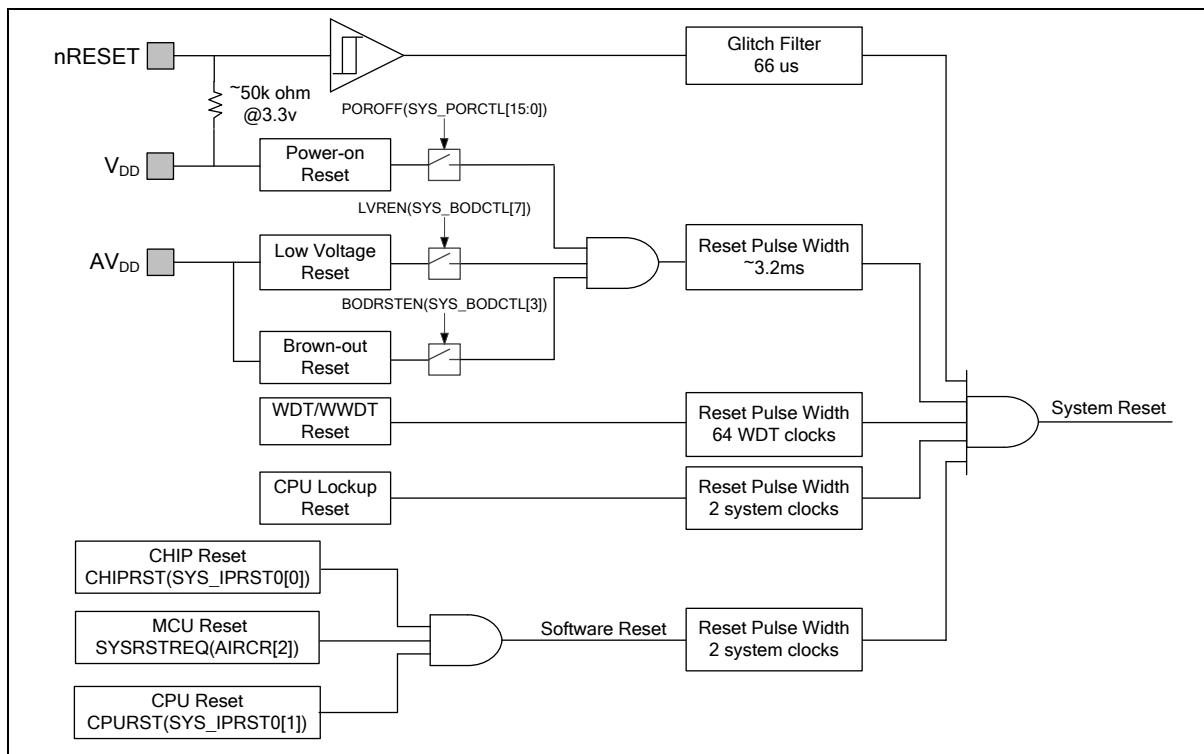


图 6.2-1 系统复位源

共 9 个复位源。见 表 6.2-1

复位源寄存器	上电复位 POR	引脚复位 NRESET	WDT	LVR	BOD	CPU锁定 Lockup	CHIP	MCU	CPU
SYS_RSTSTS	Bit 0 = 1	Bit 1 = 1	Bit 2 = 1	Bit 3 = 1	Bit 4 = 1	Bit 8 = 1	Bit 0 = 1	Bit 5 = 1	Bit 7 = 1
CHIPRST (SYS_IPRST0[0])	0x0	-	-	-	-	-	-	-	-
BODEN (SYS_BODCTL[0])	从 CONFIG0 加载到此寄 存器位	从 CONFIG0 加载 到此 寄存器位	从 CONFIG0 加载 到此 寄存器位	从 CONFIG0 加载 到此 寄存器位	-	从 CONFIG0 加载 到此 寄存器位	从 CONFIG0 加载 到此 寄存器位	从 CONFIG0 加载 到此 寄存器位	-
BODVLL (SYS_BODCTL[2:1])									
BODRSTEN (SYS_BODCTL[3])									
HXTEN (CLK_PWRCTL[0])	从 CONFIG0 加载到此寄 存器位	从 CONFIG0 加载 到此 寄存器位							
LXTEN (CLK_PWRCTL[1])	0x0	-	-	-	-	-	-	-	-
WDTCKEN (CLK_APBCLK0[0])	0x1	-	0x1	-	-	-	0x1	-	-
HCLKSEL	从 CONFIG0	从 CONFIG0	从 CONFIG0	从 CONFIG0	从 CONFIG0	从 CONFIG0	从 CONFIG0	从 CONFIG0	-

(CLK_CLKSEL0[2:0])	加载到此寄存器位	加载到此寄存器位	加载到此寄存器位	加载到此寄存器位	加载到此寄存器位	加载到此寄存器位	加载到此寄存器位	加载到此寄存器位	加载到此寄存器位	
WDTSEL (CLK_CLKSEL1[1:0])	0x3	0x3	-	-	-	-	-	-	-	-
HXTSTB (CLK_STATUS[0])	0x0	-	-	-	-	-	-	-	-	-
LXTSTB (CLK_STATUS[1])	0x0	-	-	-	-	-	-	-	-	-
PLLSTB (CLK_STATUS[2])	0x0	-	-	-	-	-	-	-	-	-
HIRCSTB (CLK_STATUS[4])	0x0	-	-	-	-	-	-	-	-	-
CLKSFAIL (CLK_STATUS[7])	0x0	0x0	-	-	-	-	-	-	-	-
RSTEN (WDT_CTL[1])	从 CONFIG0 加载到此寄 存器位	从 CONFIG0 加载到此寄 存器位	从 CONFIG0 加载到此寄 存器位	从 CONFIG0 加载到此寄 存器位	从 CONFIG0 加载到此寄 存器位	-	从 CONFIG0 加载到此寄 存器位	-	-	-
WDTEN (WDT_CTL[7])										
WDT_CTL except bit 1 and bit 7.	0x0700	0x0700	0x0700	0x0700	0x0700	-	0x0700	-	-	-
WDT_ALTCTL	0x0000	0x0000	0x0000	0x0000	0x0000	-	0x0000	-	-	-
WWDT_RLDCNT	0x0000	0x0000	0x0000	0x0000	0x0000	-	0x0000	-	-	-
WWDT_CTL	0x3F0800	0x3F0800	0x3F0800	0x3F0800	0x3F0800	-	0x3F0800	-	-	-
WWDT_STATUS	0x0000	0x0000	0x0000	0x0000	0x0000	-	0x0000	-	-	-
WWDT_CNT	0x3F	0x3F	0x3F	0x3F	0x3F	-	0x3F	-	-	-
BS (FMC_ISPCTL[1])	从 CONFIG0 加载到此寄 存器位	从 CONFIG0 加载到此寄 存器位	从 CONFIG0 加载到此寄 存器位	从 CONFIG0 加载到此寄 存器位	从 CONFIG0 加载到此寄 存器位	-	从 CONFIG0 加载到此寄 存器位	-	-	-
BL (FMC_ISPCTL[16])										
FMC_DFBA	从 CONFIG1 加载到此寄 存器位	从 CONFIG1 加载到此寄 存器位	从 CONFIG1 加载到此寄 存器位	从 CONFIG1 加载到此寄 存器位	从 CONFIG1 加载到此寄 存器位	-	从 CONFIG1 加载到此寄 存器位	-	-	-
CBS (FMC_ISPSTS[2:1])	从 CONFIG0 加载到此寄 存器位	从 CONFIG0 加载到此寄 存器位	从 CONFIG0 加载到此寄 存器位	从 CONFIG0 加载到此寄 存器位	从 CONFIG0 加载到此寄 存器位	-	从 CONFIG0 加载到此寄 存器位	-	-	-
VECMAP (FMC_ISPSTS[23:9])	从 CONFIG0 加载到此寄 存器位	从 CONFIG0 加载到此寄 存器位	从 CONFIG0 加载到此寄 存器位	从 CONFIG0 加载到此寄 存器位	从 CONFIG0 加载到此寄 存器位	-	从 CONFIG0 加载到此寄 存器位	-	-	-
Other Peripheral Registers	复位值							-		

FMC Registers	复位值
注意: '-' 表示保持原值	

表 6.2-1 寄存器复位值

## 6.2.2.1 nRESET 复位

任何时候都可以将引脚 nRESET 持续拉低至  $0.2 V_{DD}$  以下 66us，异步复位芯片。引脚电压升至  $0.7 V_{DD}$  以上后，再保持 66us 后，PINRF(SYS\_RSTSTS[1]) 置 1。复位时序见图 6.2-2。

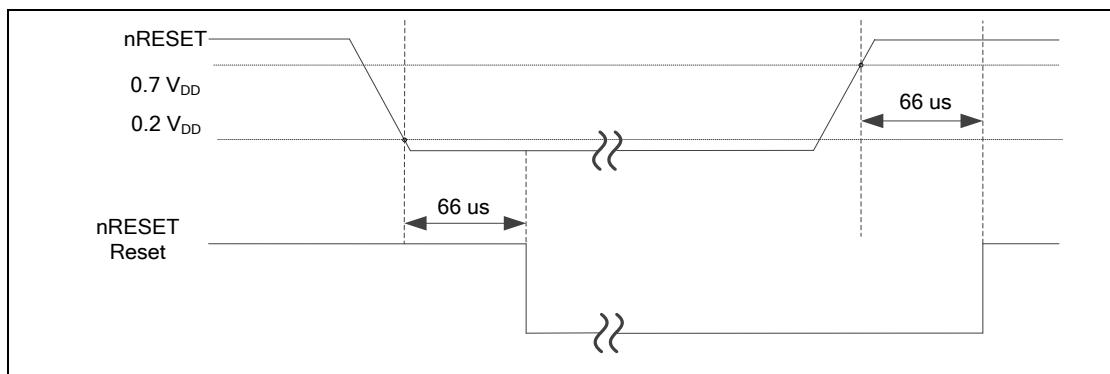


图 6.2-2 nRESET 复位时序

## 6.2.2.2 上电复位 (POR)

上电后 POR 电路让芯片保持复位，直到电压升至可工作的幅度，位 PORF(SYS\_RSTSTS[0]) 置 1 表明发生了上电，此位写 1 清 0。复位时序见图 6.2-3。

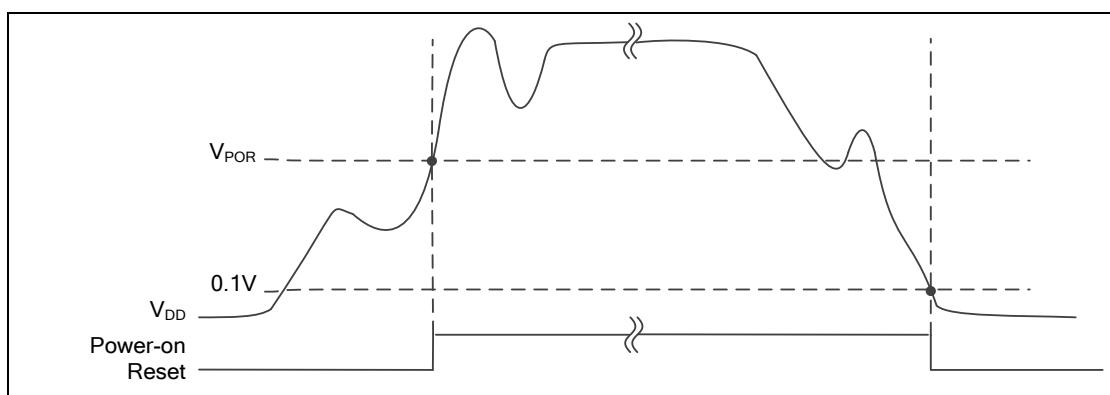


图 6.2-3 (POR) 上电复位波形

## 6.2.2.3 低电压复位 (LVR)

如果 LVREN (SYS\_BODCTL[7]) = 1 使能了低压复位，200us 后低压复位将开始工作。如果检测到  $V_{DD}$  电压低于  $V_{LVR}$  超过 LVRDGSEL (SYS\_BODCTL[14:12]) 配置的防抖时间，将产生低压复位，直到电压高于  $V_{LVR}$  的时间超过 LVRDGSEL (SYS\_BODCTL[14:12]) 配置的防抖时间后，CPU 退出复位开始工作。缺省值低压复位使能但防抖时间为 0。复位波形见图 6.2-4。

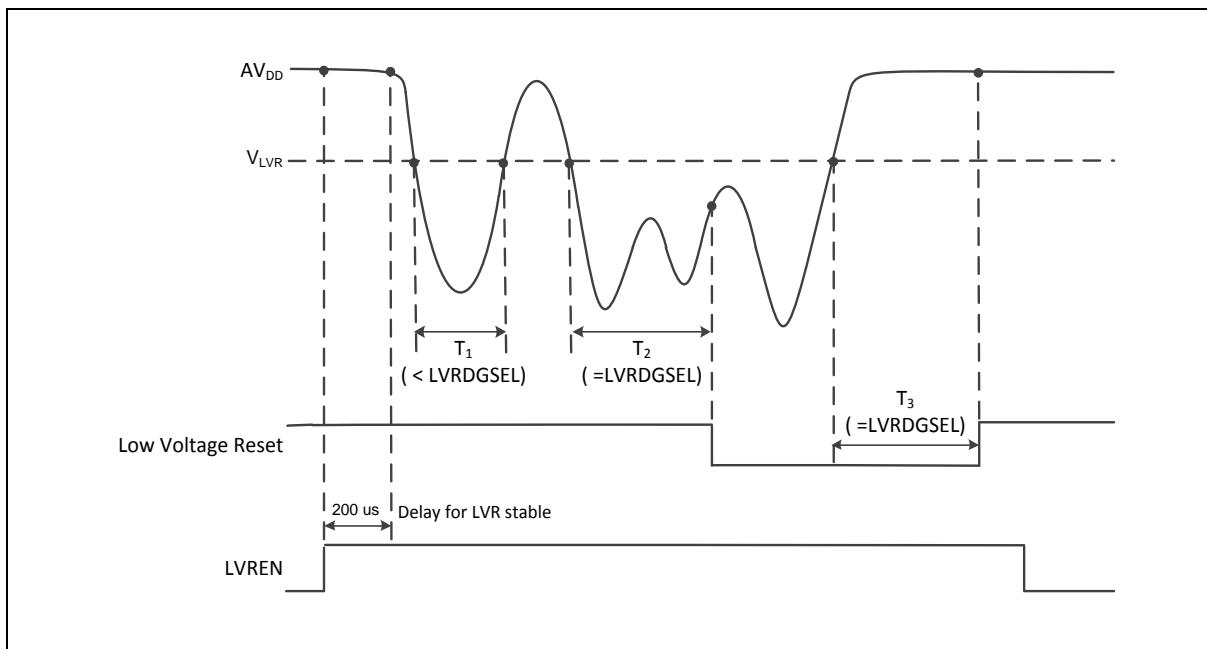


图 6.2-4 低电压复位 (LVR) 波形

#### 6.2.2.4 欠压复位(BOD 复位)

配置BODEN (SYS\_BODCTL[0])可使能欠压复位功能，若  $AV_{DD}$  低于BODVL (SYS\_BODCTL [18:16]) 配置的电压  $V_{BOD}$  的时间，超过BODDGSEL (SYS\_BODCTL[10:8])的值,将产生复位，直到电压高于  $V_{BOD}$  的时间长于 BODDGSEL 的值后，退出复位开始工作。上电后或芯片复位后 BODEN, BODVL 和 BODRSTEN (SYS\_BODCTL[3]) 的缺省值由CONFIG0的对应位加载。图 6.2-5 是复位时序波形

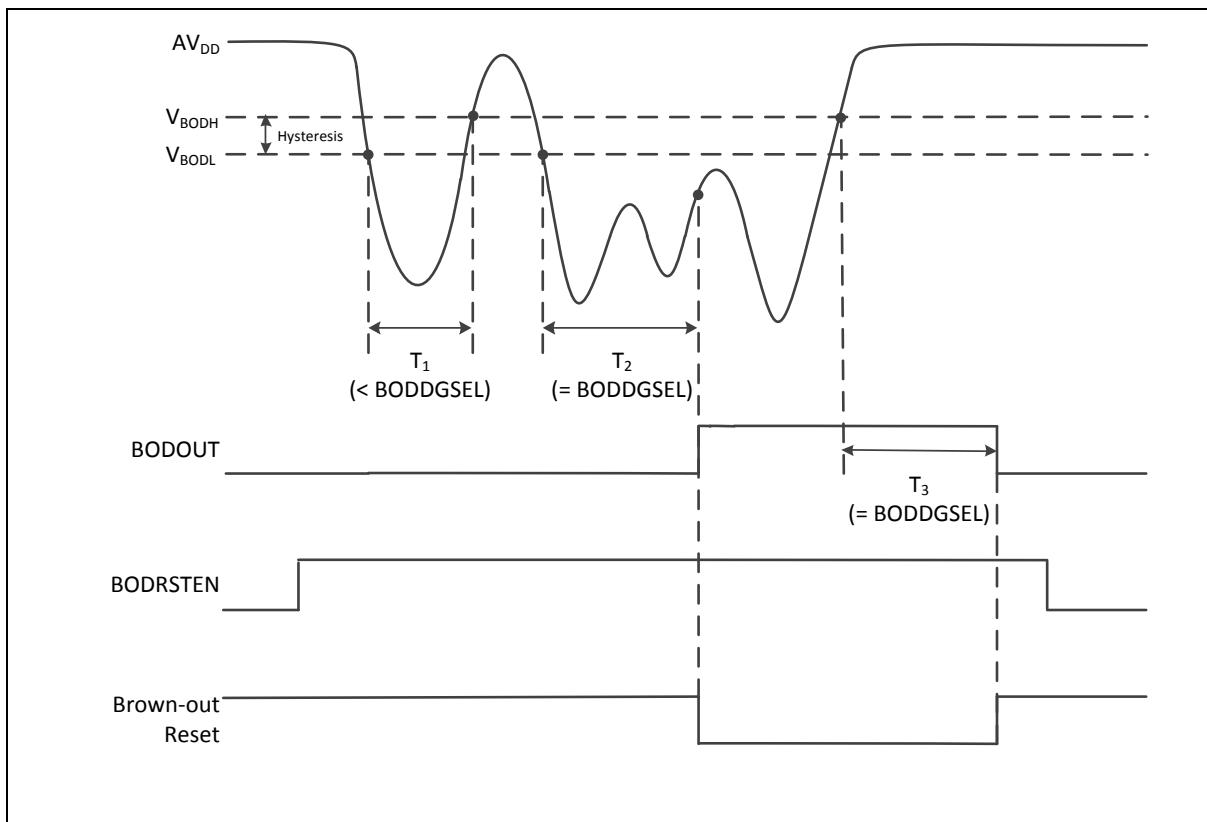


图 6.2-5 欠压侦测(BOD)波形

#### 6.2.2.5 WDT复位

如果系统失控，WDT定时器溢出会产生复位，复位源寄存器WDTRF(SYS\_RSTSTS[2])置1指示产生了WDT复位。

#### 6.2.2.6 CPU 锁死复位

非调试状态出现硬件错误异常 hardfault 将产生 CPU 锁死复位。Debug 调试状态这个复位源将被忽略。

#### 6.2.2.7 CPU 复位, 芯片复位和 MCU 复位

CPURST(SYS\_IPRST0[1])写1产生Cortex®-M4内核复位，所有外设都不复位，保持原工作状态不变。

CHIPRST(SYS\_IPRST0[1])写1产生芯片复位，与重新上电一样。启动选项 BS(FMC\_ISPCTL[1]) 将从 CONFIG0 重新加载

SYSRESETREQ(AIRCR[2]) 写1产生 MCU 复位，与芯片复位不同的是，不会由CONFIG0加载 BS(FMC\_ISPCTL[1]) 所以从 APROM 还是从 LDROM 启动将不变。

### 6.2.3 系统电源分配

电源的四个部分：

- 模拟部分的电源由 AV<sub>DD</sub> 和 AV<sub>SS</sub> 提供
- 数字部分包括GPIO口和内核1.8 V由 V<sub>DD</sub> 和 V<sub>SS</sub> 供电

- USB 部分由VBUS 供电
- RTC部分和它的80个字节内存由 V<sub>DD</sub> 供电

靠近片内 LDO 和 VDD33的输出引脚，要有个电容。AV<sub>DD</sub> 的数值要与V<sub>DD</sub>一样. 图 6.2-6 是 M480电源分布框图.

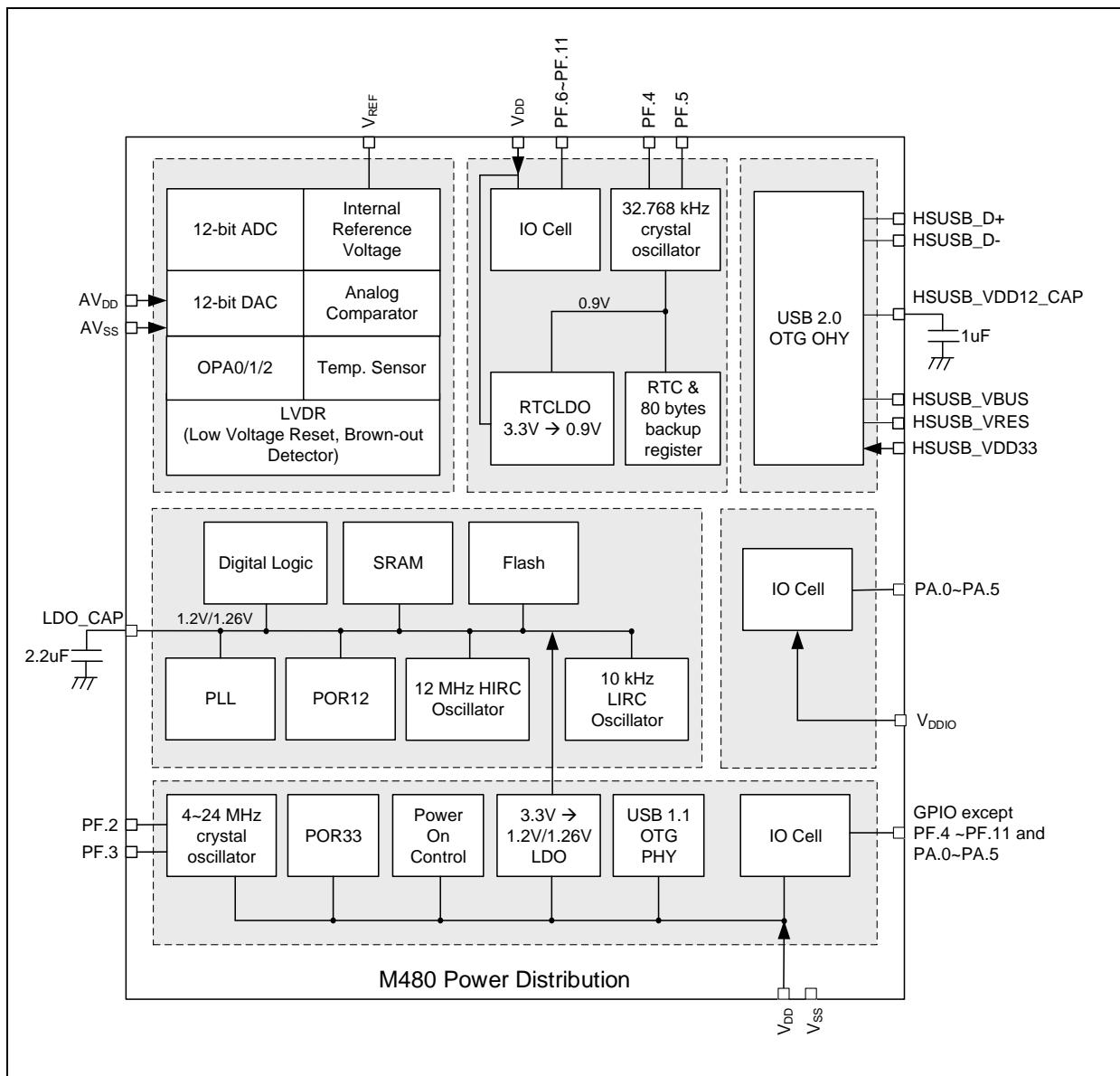


图 6.2-6 NuMicro® M480 电源分布图

#### 6.2.4 电源模式和唤醒源

NuMicro® M480 系列支持几种操作模式来降低功耗，如表 6.2-2 所列

模式	CPU 操作最大速度 (MHz)	LDO_CAP (V)	时钟禁止
正常模式	160	1.20	所有时钟通过控制寄存器禁止
增强模式	192	1.26	所有时钟通过控制寄存器禁止

空闲模式	CPU 进入睡眠模式	1.20/1.26	仅 CPU 时钟禁止
快速唤醒掉电模式 (FWPD)	CPU 进入深度睡眠模式	1.20/1.26	除了LIRC/LXT外，大部分时钟禁止。在时钟源选择 LIRC/LXT 的前提下，仅 RTC/WDT/Timer/UART 外设时钟仍然使能。
正常掉电模式 (NPD)	CPU 进入深度睡眠模式	1.20/1.26	除了LIRC/LXT外，大部分时钟禁止。在时钟源选择 LIRC/LXT 的前提下，仅 RTC/WDT/Timer/UART 外设时钟仍然使能。
低耗电掉电模式 (LLPD)	CPU 进入深度睡眠模式	0.9	除了LIRC/LXT外，大部分时钟禁止。在时钟源选择 LIRC/LXT 的前提下，仅 RTC/WDT/Timer/UART 外设时钟仍然使能。
待机掉电模式 0 (SPD0) <sup>[1]</sup>	关机	悬浮不定	仅 LIRC/LXT 仍然使能，用于 RTC 功能和唤醒定时器。
待机掉电模式 1 (SPD1) <sup>[1]</sup>	关机	悬浮不定	仅 LIRC/LXT 仍然使能，用于 RTC 功能和唤醒定时器。
深度掉电模式 (DPD)	关机	悬浮不定	仅 LIRC 仍然使能，用于 RTC 功能和唤醒定时器。

表 6.2-2 电源模式表

## 注意：

1. 在进入 SPD0/1 模式之前，用户必须打开 LIRC。

表 6.2-3 列举了进入不同工作模式的方法。上电后是正常工作模式。通过设置 SLEEPDEEP (SCR[2]), PDEN (CLK\_PWRCTL[7]) 和 PDMSEL (CLK\_PMUCTL[2:0]) 和执行 WFI 指令用户可以进入各模式。

寄存器/指令模式	SLEEPDEEP (SCR[2])	PDEN (CLK_PWRCTL[7])	PDMSEL (CLK_PMUCTL[2:0])
快速唤醒掉电模式	1	1	2
正常掉电模式	1	1	0
低耗电掉电模式	1	1	1
待机掉电模式 0	1	1	4
待机掉电模式 1	1	1	5
深度掉电模式	1	1	6

表 6.2-3 电源模式区别表

在空闲模式和掉电模式有几个唤醒源。表 6.2-4 列举了各个模式下可用的时钟。

电源模式	正常模式	空闲模式	掉电模式
定义	CPU 在工作状态	CPU 在睡眠状态	CPU 在睡眠状态且除 LXT 和 LIRC 外所有时钟停止。SRAM 数据保持。
进入条件	系统复位后芯片工作在正常模式	CPU 执行 WFI 指令	使能 CPU 休眠和掉电功能后，执行指令 WFI。

唤醒源	N/A	All 中断	RTC, WDT, I²C, Timer, UART, BOD, GPIO, EINT, USCI, USBD, ACMP 和 BOD.
可用时钟	All	除 CPU时钟外的所有时钟	LXT 和 LIRC
唤醒后	N/A	CPU 返回正常模式	CPU返回正常模式

表 6.2-4 电源模式定义表

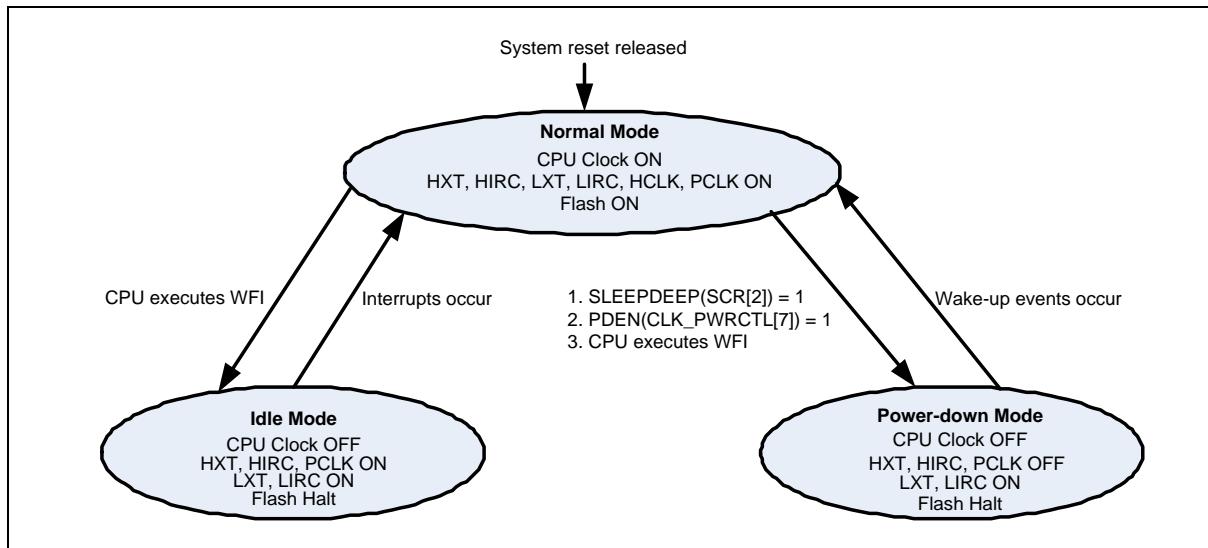


图 6.2-7 电源模式状态机

	空闲模式	NPD, LLPD, FWPD	SPD0, SPD1,	DPD
HXT	ON	停止	停止	停止
HIRC	ON	停止	停止	停止
LXT	ON	ON/OFF <sup>[1]</sup>	ON/OFF <sup>[1]</sup>	停止
LIRC	ON	ON/OFF <sup>[2]</sup>	ON/OFF <sup>[2]</sup>	ON/OFF <sup>[2]</sup>
PLL	ON	停止	停止	停止
HCLK/PCLK	ON	停止	停止	停止
CPU	停止	停止	停止	停止
SRAM 保持	ON	ON	停止	停止
FLASH	停止	停止	停止	停止
TIMER	ON	ON/OFF <sup>[3]</sup>	ON/OFF <sup>[3]</sup>	停止
WDT	ON	ON/OFF <sup>[4]</sup>	停止	停止
RTC	ON	ON/OFF <sup>[5]</sup>	ON/OFF <sup>[5]</sup>	停止
UART	ON	ON/OFF <sup>[6]</sup>	停止	停止
其他	ON	停止	停止	停止

表 6.2-5 不同电源模式的时钟

**注意：**

1. LXT ON 或 OFF 取决于在正常模式时软件的设置
2. LIRC ON 或 OFF 取决于在正常模式时软件的设置 .
3. 是否 TIMER 时钟源选择为LIRC/LXT 且 LIRC/LXT 使能.
4. 是否WDT时钟源选择为LIRC 且LIRC使能
5. 是否RTC时钟源选择为LXT且LXT使能.
6. 是否UART时钟源选择为LXT且LXT使能

**正常掉电模式(NPD)下的唤醒源：**

RTC, WDT, I<sup>2</sup>C, Timer, UART, USCI, BOD, EBOD, GPIO, USBD, 和 ACMP都可以让掉电状态的 CPU 唤醒到正常工作模式

表 6.2-6 列举了如何再次进入掉电状态

在设置 PDEN(CLK\_PWRCTL[7]) 和执行 WFI 进入掉电之前，需等这些条件满足

唤醒源	唤醒条件	省电模式			再次进入省电模式的条件
		NPD/ FWPD/ LLPD	SPD0/1	DPD	
BOD	BOD 复位 / 中断	V	-	-	软件对 BODIF (SYS_BODCTL[4])写1清0后 .
	BOD复位	-	V	-	当进入SPD模式， 软件对BODWK (CLK_PMUSTS[13])写1清0 后.
LVR	LVR 复位	V	-	-	软件对LVRF (SYS_RSTSTS[3])写1清0 后.
		-	V	-	当进入SPD模式， 软件对LVRWK (CLK_PMUSTS[12])写1清0 后.
POR	POR 复位	V	V	-	软件对PORF (SYS_RSTSTS[0])写1清0 后.
INT	外中断	V	-	-	软件写1清零 Px_INTSRC[n] 之后
GPIO	GPIO 中断	V	-	-	软件写1清零Px_INTSRC[n] 之后
GPIO(PA~PD) Wake-up pin	上或下跳沿, 64个引脚	-	V	-	进入SPD模式， 位GPxWK(CLK_PMUSTS[11:8])清0后
GPIO(PC.0) Wake-up pin	上或下跳沿, 1个引脚	-	-	V	进入DPD模式， 位PINWK(CLK_PMUSTS[1]) 清0后
TIMER	定时器中断	V	-	-	软件写1清零 TWKF (TIMERx_INTSTS[1]) 和 TIF (TIMERx_INTSTS[0])后
Wakeup timer	唤醒定时器溢出后	-	V	V	当进入SPD或DPD模式， 软件对TMRWK (CLK_PMUSTS[1])写1清0 后.
WDT	WDT 中断	V	-	-	软件写1清零WKF (WDT_CTL[5]) (写保护)后.
RTC	闹钟中断	V	-	-	软件写1清零ALMIF (RTC_INTSTS[0])后
	周期节拍中断	V	-	-	软件写1清零TICKIF (RTC_INTSTS[1])后

	闹钟唤醒	-	V	-	当进入SPD模式，软件对RTCKW (CLK_PMUSTS[2])写1清0后。
	周期节拍唤醒	-	V	-	当进入SPD模式，软件对RTCKW (CLK_PMUSTS[2])写1清0后
	Tamper事件唤醒	-	V	-	当进入SPD模式，软件对RTCKW (CLK_PMUSTS[2])写1清0后
UART	nCTS 唤醒	V	-	-	软件写1清零CTSWKF (UARTx_WKSTS[0])后。
	RX 数据唤醒	V	-	-	软件写1清零DATWKF (UARTx_WKSTS[1])后。
	接收 FIFO 数据个数到门限唤醒	V	-	-	软件写1清零RFRTWKF (UARTx_WKSTS[2])后。
	RS-485自动地址识别匹配唤醒	V	-	-	软件写1清零RS485WKF (UARTx_WKSTS[3])后。
	接收 FIFO 门限，超时唤醒	V	-	-	软件写1清零TOUTWKF (UARTx_WKSTS[4])后。
USCI UART	CTS 跳变	V	-	-	软件写1清零WKF (UUART_WKSTS[0])后。
	数据跳变	V	-	-	软件写1清零WKF (UUART_WKSTS[0])后。
USCI I2C	数据跳变	V	-	-	软件写1清零WKF (UI2C_WKSTS[0])后。
	地址匹配	V	-	-	软件写1清零WKAKDONE (UI2C_PROTSTS[16]后, 然后写1清0 WKF (UI2C_WKSTS[0])).
USCI SPI	SS 跳变	V	-	-	软件写1清零WKF (USPI_WKSTS[0])后。
I <sup>2</sup> C	地址匹配唤醒	V	-	-	软件写1清零WKAKDONE (I2C_WKSTS[1])后, 然后写1清0 WKIF(I2C_WKSTS[0]).
USBD	远程唤醒	V	-	-	软件写1清零BUSIF (USBD_INTSTS[0])后。
ACMP	比较器掉电唤醒中断	V	-	-	软件写1清零WKIF0 (ACMP_STATUS[8]) 和 WKIF1 (ACMP_STATUS[9])后。
ACMP	比较器输出改变	-	V	-	当进入SPD模式，软件写1清零ACMPWK (CLK_PMUSTS[14]) 后

表 6.2-6 重新进入掉电状态的条件

## 6.2.5 电源模式转换

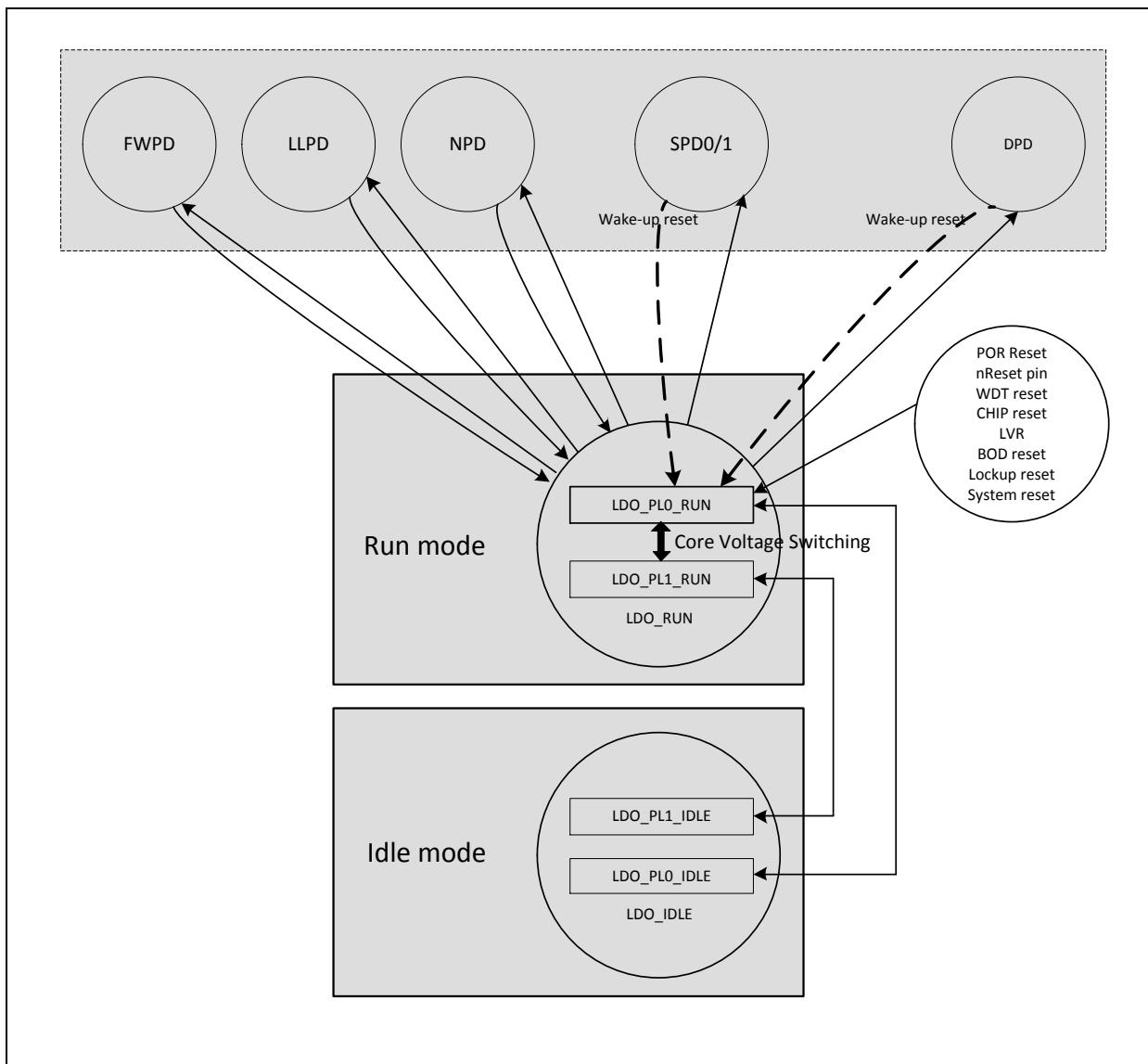


图 6.2-8 NuMicro® M480 电源模式图

### 6.2.6 系统内存映射

M480 可访问 4G 字节空间，表 6.2-7 是空间定义，M480 仅支持小端数据格式。

地址空间	符号	描述
<b>Flash 和 SRAM</b>		
0x0000_0000 – 0x0003_FFFF	FLASH_BA	FLASH 区 (256KB)
0x0000_0000 – 0x0007_FFFF	FLASH_BA	FLASH 区 (512KB)
0x0800_0000 – 0x09FF_FFFF	SPIM_BA	SPIM 区 (32MB)
0x2000_0000 – 0x2000_7FFF	SRAM0_BA	SRAM 区 (32KB)
0x2000_8000 – 0x2001_FFFF	SRAM1_BA	SRAM 区 (96KB)
0x2002_0000 – 0x2002_7FFF	SRAM2_BA	仅用于 CPU 和 SPIM 高速缓存 SRAM 区 (32KB)

0x6000_0000 – 0x6FFF_FFFF	EXTMEM_BA	外接内存区 (256MB)
<b>外设区 (0x4000_0000 – 0x400F_FFFF)</b>		
0x4000_0000 – 0x4000_01FF	SYS_BA	系统控制寄存器
0x4000_0200 – 0x4000_02FF	CLK_BA	时钟控制寄存器
0x4000_0300 – 0x4000_03FF	NMI_BA	非屏蔽中断控制寄存器
0x4000_4000 – 0x4000_4FFF	GPIO_BA	引脚输入输出控制寄存器
0x4000_7000 – 0x4000_7FFF	SPIM_BA	SPIM 控制寄存器
0x4000_8000 – 0x4000_8FFF	PDMA_BA	外设 DMA 控制寄存器
0x4000_9000 – 0x4000_9FFF	USBH_BA	USB 主机端 控制寄存器
0x4000_B000 – 0x4000_BFFF	EMAC_BA	以太网 MAC 控制寄存器
0x4000_C000 – 0x4000_CFFF	FMC_BA	闪存控制寄存器
0x4000_D000 – 0x4000_DFFF	SDH0_BA	SD卡接口0 控制寄存器
0x4000_E000 – 0x4000_EFFF	SDH1_BA	SD卡接口1 控制寄存器
0x4001_0000 – 0x4001_0FFF	EBI_BA	外总线接口控制寄存器
0x4001_9000 – 0x4001_9FFF	HSUSBD_BA	高速USB设备端控制寄存器
0x4001_A000 – 0x4001_AFFF	HSUSBH_BA	高速USB主机端控制寄存器
0x4003_1000 – 0x4003_1FFF	CRC_BA	CRC 运算控制寄存器
0x4003_E000 – 0x4003_EFFF	SWDC_BA	烧录仿真接口 SWD 控制寄存器
0x4003_F000 – 0x4003_FFFF	ETMC_BA	ETM 控制寄存器
0x5008_0000 – 0x5008_0FFF	CRYP_BA	加解码算法控制寄存器
0x4004_0000 – 0x4004_0FFF	WDT_BA	看门狗控制寄存器
0x4004_1000 – 0x4004_1FFF	RTC_BA	实时时钟 (RTC) 控制寄存器
0x4004_3000 – 0x4004_3FFF	EADC_BA	增强的ADC (EADC) 控制寄存器
0x4004_5000 – 0x4004_5FFF	ACMP01_BA	模拟比较器 0/1 控制寄存器
0x4004_6000 – 0x4004_6FFF	OPA_BA	运放 控制寄存器
0x4004_7000 – 0x4004_7FFF	DAC_BA	DAC 控制寄存器
0x4004_8000 – 0x4004_8FFF	I2S0_BA	I <sup>2</sup> S0 控制寄存器
0x4004_D000 – 0x4004_DFFF	OTG_BA	OTG 控制寄存器
0x4004_F000 – 0x4004_FFFF	HSOTG_BA	高速 OTG 控制寄存器
0x4005_0000 – 0x4005_0FFF	TMR01_BA	Timer0/Timer1 控制寄存器
0x4005_1000 – 0x4005_1FFF	TMR23_BA	Timer2/Timer3 控制寄存器
0x4005_8000 – 0x4005_8FFF	EPWM0_BA	EPWM0 控制寄存器
0x4005_9000 – 0x4005_9FFF	EPWM1_BA	EPWM1 控制寄存器

0x4005_A000 – 0x4005_AFFF	BPWM0_BA	BPWM0 控制寄存器
0x4005_B000 – 0x4005_BFFF	BPWM1_BA	BPWM1 控制寄存器
0x4006_0000 – 0x4006_0FFF	QSPI0_BA	QSPI0 控制寄存器
0x4006_1000 – 0x4006_1FFF	SPI0_BA	SPI0 控制寄存器
0x4006_2000 – 0x4006_2FFF	SPI1_BA	SPI1 控制寄存器
0x4006_3000 – 0x4006_3FFF	SPI2_BA	SPI2 控制寄存器
0x4006_4000 – 0x4006_4FFF	SPI3_BA	SPI3 控制寄存器
0x4007_0000 – 0x4007_0FFF	UART0_BA	UART0 控制寄存器
0x4007_1000 – 0x4007_1FFF	UART1_BA	UART1 控制寄存器
0x4007_2000 – 0x4007_2FFF	UART2_BA	UART2 控制寄存器
0x4007_3000 – 0x4007_3FFF	UART3_BA	UART3 控制寄存器
0x4007_4000 – 0x4007_4FFF	UART4_BA	UART4 控制寄存器
0x4007_5000 – 0x4007_5FFF	UART5_BA	UART5 控制寄存器
0x4008_0000 – 0x4008_0FFF	I2C0_BA	I <sup>2</sup> C0 控制寄存器
0x4008_1000 – 0x4008_1FFF	I2C1_BA	I <sup>2</sup> C1 控制寄存器
0x4008_2000 – 0x4008_2FFF	I2C2_BA	I <sup>2</sup> C2 控制寄存器
0x4009_0000 – 0x4009_0FFF	SC0_BA	智能卡接口 0 控制寄存器
0x4009_1000 – 0x4009_1FFF	SC1_BA	智能卡接口 1 控制寄存器
0x4009_2000 – 0x4009_2FFF	SC2_BA	智能卡接口 2 控制寄存器
0x4009_3000 – 0x4009_3FFF	SC3_BA	智能卡接口 3 控制寄存器
0x400A_0000 – 0x400A_0FFF	CAN0_BA	CAN0 控制寄存器
0x400A_1000 – 0x400A_1FFF	CAN1_BA	CAN1 控制寄存器
0x400B_0000 – 0x400B_0FFF	QEI0_BA	QEI0 控制寄存器
0x400B_1000 – 0x400B_1FFF	QEI1_BA	QEI1 控制寄存器
0x400B_4000 – 0x400B_4FFF	ECAP0_BA	ECAP0 控制寄存器
0x400B_5000 – 0x400B_5FFF	ECAP1_BA	ECAP1 控制寄存器
0x400C_0000 – 0x400C_0FFF	USBD_BA	USB设备端控制寄存器
0x400D_0000 – 0x400D_0FFF	USCI0_BA	USCI0 控制寄存器
0x400D_1000 – 0x400D_1FFF	USCI1_BA	USCI1 控制寄存器
<b>系统控制区 (0xE000_E000 ~ 0xE000_EFFF)</b>		
0xE000_E010 – 0xE000_E0FF	SCS_BA	系统定时器控制寄存器
0xE000_E100 – 0xE000_ECFF	SCS_BA	外中断控制寄存器
0xE000_ED00 – 0xE000_ED8F	SCS_BA	系统控制控制寄存器

表 6.2-7 片上地址空间分布

### 6.2.7 SRAM 内存结构

160 K字节的 SRAM 分成三个区块: bank0 32 KB, bank1 96 KB, bank2 32 KB. 三个区块, 地址连续, bank0 有奇偶校验, bank2 与SPIM 高速缓存cache共享, 可切换到外 SPI Flash cache 空间。

- 160 K字节
- 支持字节 / 半字 / 字的写入
- bank0 可独立访问
- bank0支持奇偶校验
- 支持地址溢出出错响应
- 首地址可重映射到 0x1000\_0000

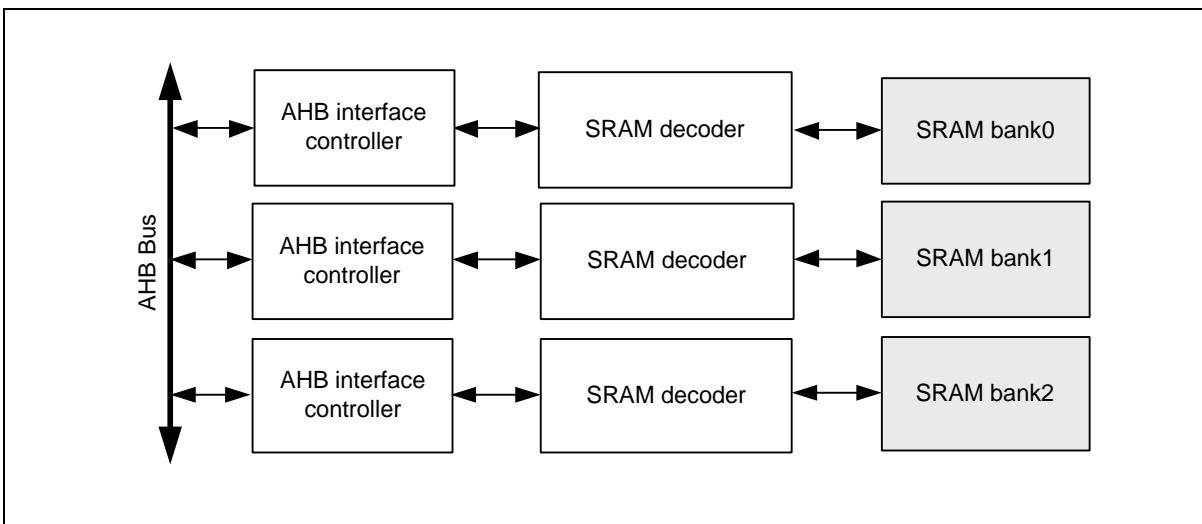


图 6.2-9 SRAM 框图

图 6.2-9 SRAM 是SRAM框图, bank0 地址范围 0x2000\_0000 ~ 0x2000\_7FFF. bank1地址范围 0x2000\_8000 ~ 0x2001\_FFFF. bank2地址范围0x2002\_0000 ~ 0x2002\_7FFF. CPU若读写地址范围 0x2002\_8000 ~ 0x3FFF\_FFFF这些非法空间, 会产生 hardfault。

每个空间都可以从地址 0x2000\_0000 重映射到 0x1000\_0000。也可以从地址 0x1000\_0000 ~ 0x1000\_7FFF CPU 读写 bank0, 也可以从地址0x1000\_8000~0x1001\_FFFF读写 bank1, 也可以从地址0x1002\_0000 ~ 0x1002\_7FFF读写 bank2。

CCMEN(SPIM\_CTL1[2]) = 0时, SRAM bank2切换成外部 SPI Flash 高速缓存, 此时SRAM bank2 不能再当做通用SRAM, 读写这个区域会返回HRESP AHB错误。

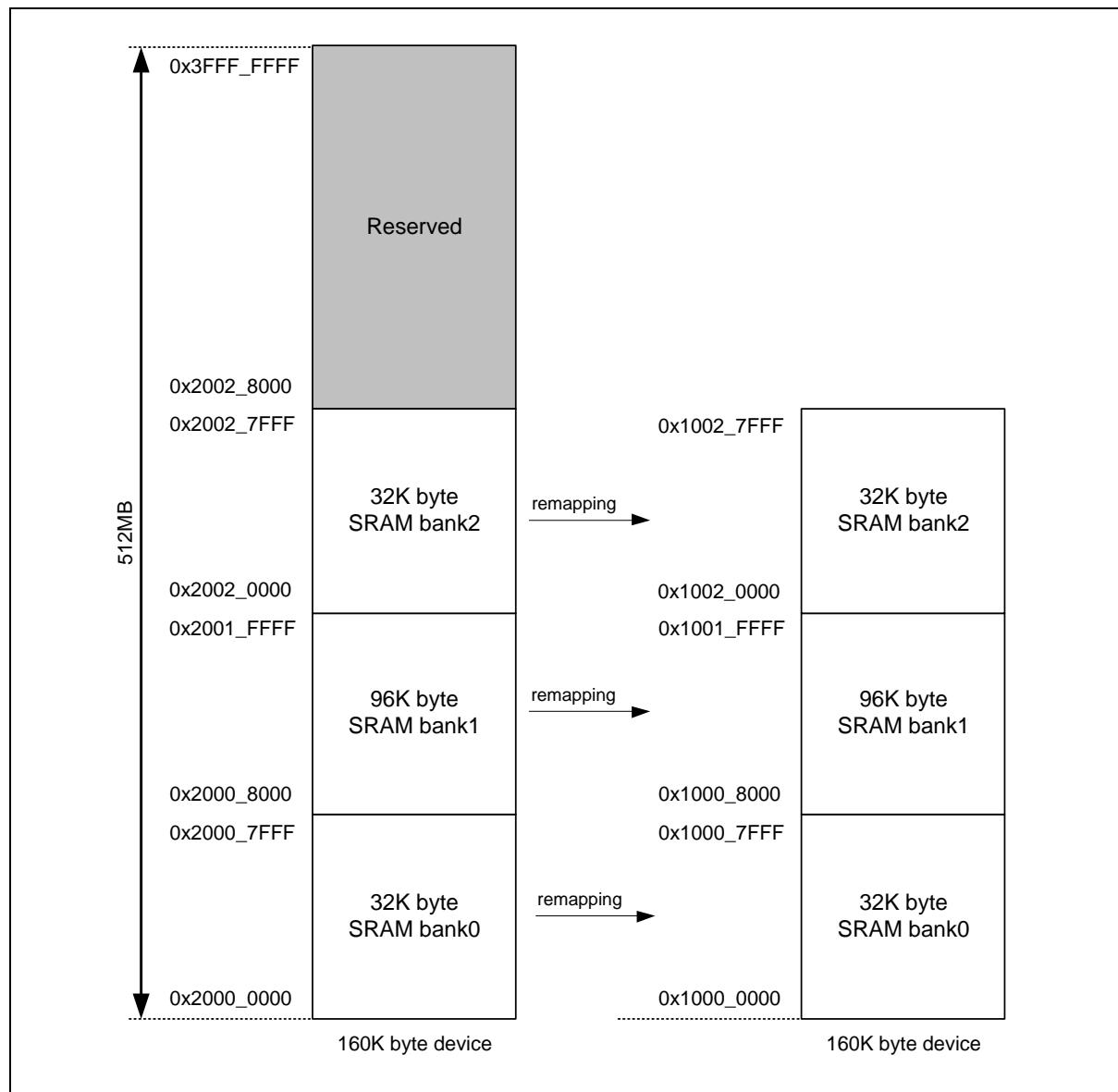


图 6.2-10 SRAM 内存结构

SRAM bank0 有奇偶校验功能，读出时的奇偶校验值若与写入时不一样， PERRIF (SYS\_SRAM\_STATUS[0]) 将置1， SYS\_SRAM\_ERRADDR 记录产生奇偶错的地址。若 PERRIEN (SYS\_SRAM\_INTCTL[0])=1. 将产生中断。在PERRIF(SYS\_SRAM\_STATUS[0]) 写1清0之前，芯片将不再检测奇偶校验错。

### 6.2.8 总线矩阵

M480 支持总线促裁.总线仲法则裁由 INTACTEN (SYS\_AHBMCTL[0]) 决定， CPU 有最高总线优先级。

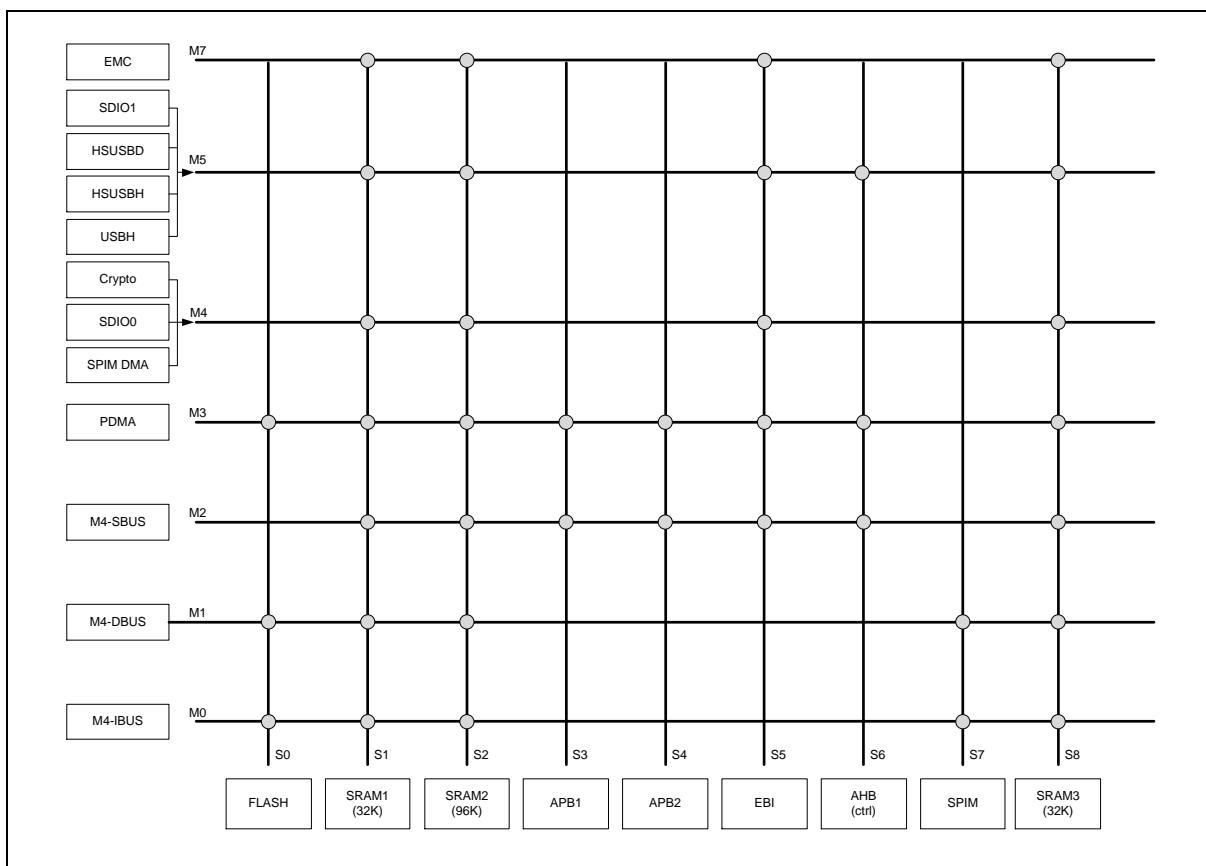


图 6.2-11 NuMicro® M480 总线矩阵

### 6.2.9 HIRC自动校准

位REFCKSEL (SYS\_IRCTCTL[10])配置时钟参考源选择LXT (32768 Hz), 还是USB SOF (Start-Of-Frame), 再让 FREQSEL (SYS\_IRCTCTL[1:0]) = “10”, 就使能了HIRC自动校准功能, 当位FREQLOCK (SYS\_IRCTISTS[8])=1时, 表明 HIRC 已达到全温度范围内 $12\text{MHz} \pm 0.25\%$ 的精度。

### 6.2.10 寄存器映射

R:只读, W: 只写, R/W: 可读写

寄存器	地址偏移	读/写	描述	复位值
<b>SYS 基址:</b>				
<b>SYS_BA = 0x4000_0000</b>				
<b>SYS_P DID</b>	SYS_BA+0x00	只读	芯片型号	0x0045_1A93 <sup>[1]</sup>
<b>SYS_RSTSTS</b>	SYS_BA+0x04	读/写	复位源寄存器	0x0000_0043
<b>SYS_IPRST0</b>	SYS_BA+0x08	读/写	外设复位控制寄存器0	0x0000_0000
<b>SYS_IPRST1</b>	SYS_BA+0x0C	读/写	外设复位控制寄存器1	0x0000_0000
<b>SYS_IPRST2</b>	SYS_BA+0x10	读/写	外设复位控制寄存器2	0x0000_0000
<b>SYS_BODCTL</b>	SYS_BA+0x18	读/写	BOD 欠压控制寄存器	0x000X_038X
<b>SYS_IVSCTL</b>	SYS_BA+0x1C	读/写	片内电压控制寄存器	0x0000_0000
<b>SYS_PORCTL</b>	SYS_BA+0x24	读/写	上电复位控制寄存器	0x0000_00XX
<b>SYS_VREFCTL</b>	SYS_BA+0x28	读/写	参考电压V <sub>REF</sub> 控制寄存器	0x0000_0200
<b>SYS_USBPHY</b>	SYS_BA+0x2C	读/写	USB 物理层控制寄存器	0x0003_0007
<b>SYS_GPA_MFPL</b>	SYS_BA+0x30	读/写	GPIOA 低位功能选择控制寄存器	0x0000_0000
<b>SYS_GPA_MFPH</b>	SYS_BA+0x34	读/写	GPIOA 高位功能选择控制寄存器	0x0000_0000
<b>SYS_GPB_MFPL</b>	SYS_BA+0x38	读/写	GPIOB 低位功能选择控制寄存器	0x0000_0000
<b>SYS_GPB_MFPH</b>	SYS_BA+0x3C	读/写	GPIOB高位功能选择控制寄存器	0x0000_0000
<b>SYS_GPC_MFPL</b>	SYS_BA+0x40	读/写	GPIOC 低位功能选择控制寄存器	0x0000_0000
<b>SYS_GPC_MFPH</b>	SYS_BA+0x44	读/写	GPIOC高位功能选择控制寄存器	0x0000_0000
<b>SYS_GPD_MFPL</b>	SYS_BA+0x48	读/写	GPIOD 低位功能选择控制寄存器	0x0000_0000
<b>SYS_GPD_MFPH</b>	SYS_BA+0x4C	读/写	GPIOD高位功能选择控制寄存器	0x0000_0000
<b>SYS_GPE_MFPL</b>	SYS_BA+0x50	读/写	GPIOE 低位功能选择控制寄存器	0x0000_0000
<b>SYS_GPE_MFPH</b>	SYS_BA+0x54	读/写	GPIOE高位功能选择控制寄存器	0x0000_0000
<b>SYS_GPF_MFPL</b>	SYS_BA+0x58	读/写	GPIOF 低位功能选择控制寄存器	0x0000_00EE
<b>SYS_GPF_MFPH</b>	SYS_BA+0x5C	读/写	GPIOF高位功能选择控制寄存器	0x0000_0000
<b>SYS_GPG_MFPL</b>	SYS_BA+0x60	读/写	GPIOG低位功能选择控制寄存器	0x0000_0000
<b>SYS_GPG_MFPH</b>	SYS_BA+0x64	读/写	GPIOG高位功能选择控制寄存器	0x0000_0000
<b>SYS_GPH_MFPL</b>	SYS_BA+0x68	读/写	GPIOH低位功能选择控制寄存器	0x0000_0000
<b>SYS_GPH_MFPH</b>	SYS_BA+0x6C	读/写	GPIOH高位功能选择控制寄存器	0x0000_0000

<b>SYS_GPA_MFOS</b>	SYS_BA+0x80	读/写	GPIOA 输出模式选择寄存器	0x0000_0000
<b>SYS_GPB_MFOS</b>	SYS_BA+0x84	读/写	GPIOB输出模式选择寄存器	0x0000_0000
<b>SYS_GPC_MFOS</b>	SYS_BA+0x88	读/写	GPIOC输出模式选择寄存器	0x0000_0000
<b>SYS_GPD_MFOS</b>	SYS_BA+0x8C	读/写	GPIOD输出模式选择寄存器	0x0000_0000
<b>SYS_GPE_MFOS</b>	SYS_BA+0x90	读/写	GPIOE输出模式选择寄存器	0x0000_0000
<b>SYS_GPF_MFOS</b>	SYS_BA+0x94	读/写	GPIOF输出模式选择寄存器	0x0000_0000
<b>SYS_GPG_MFOS</b>	SYS_BA+0x98	读/写	GPIOG输出模式选择寄存器	0x0000_0000
<b>SYS_GPH_MFOS</b>	SYS_BA+0x9C	读/写	GPIOH输出模式选择寄存器	0x0000_0000
<b>SYS_SRAM_INTC_TL</b>	SYS_BA+0xC0	读/写	SRAM中断使能控制寄存器	0x0000_0000
<b>SYS_SRAM_STAT_US</b>	SYS_BA+0xC4	读/写	SRAM奇偶错状态寄存器	0x0000_0000
<b>SYS_SRAM_ERR_ADDR</b>	SYS_BA+0xC8	只读	SRAM奇偶错地址寄存器	0x0000_0000
<b>SYS_SRAM_BIST_CTL</b>	SYS_BA+0xD0	读/写	SRAM BIST测试控制寄存器	0x0000_0000
<b>SYS_SRAM_BIST_STS</b>	SYS_BA+0xD4	只读	SRAM BIST 测试状态寄存器	0x00xx_00xx
<b>SYS_IRCTCTL</b>	SYS_BA+0xF0	读/写	HIRC 校准控制寄存器	0x0000_0000
<b>SYS_IRCTIEN</b>	SYS_BA+0xF4	读/写	HIRC 校准中断使能寄存器	0x0000_0000
<b>SYS_IRCTISTS</b>	SYS_BA+0xF8	读/写	HIRC 校准中断状态寄存器	0x0000_0000
<b>SYS_REGLCTL</b>	SYS_BA+0x100	读/写	寄存器锁定控制寄存器	0x0000_0000
<b>SYS_PLCTL</b>	SYS_BA+0x1F8	读/写	内核电压控制	0x0000_0000
<b>SYS_PLSTS</b>	SYS_BA+0x1FC	读/写	内核电压状态寄存器	0x0000_010X
<b>SYS_AHBMCTL</b>	SYS_BA+0x400	读/写	AHB 总线优先级控制寄存器	0x0000_0001

### 6.2.11 寄存器描述

#### SYS\_PDID 芯片型号寄存器

寄存器	偏移	R/W	描述	复位值
SYS_PDID	SYS_BA+0x00	只读	芯片型号寄存器	0x0045_1A93 <sup>[1]</sup>

[1] 每个型号唯一。

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

位	描述
[31:0]	PDID 芯片型号(只读)

**SYS\_RSTSTS 系统复位状态寄存器**

寄存器	偏移	R/W	描述	复位值
SYS_RSTSTS	SYS_BA+0x04	R/W	复位源指示寄存器	0x0000_0043

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
CPURF	Reserved	SYSRF	BODRF	LVRF	WDTRF	PINRF	PORF

位	描述	
[31:9]	Reserved	保留.
[8]	CPULKRF	<b>CPU锁死复位</b> 0 = 未发生CPU锁死复位。 1 = 发生CPU锁死复位 <b>注意:</b> 写1清0 <b>注意2:</b> 连ICE时，此位置1但不会发生复位
[7]	CPURF	<b>CPU复位</b> 0 = 未产生下述复位 1 = 复位原因是对CPURST (SYS_IPRST0[1])写1，使内核和 FMC 复位了 <b>注意:</b> 写1清0
[6]	Reserved	保留.
[5]	SYSRF	<b>系统复位</b> 0 = 未产生下述复位 1 = 复位原因是对位SYSRESETREQ(AIRCR[2](地址 0xE000ED0C) 写1造成的。这个复位不会加载CONFIG0/1 <b>注意:</b> 写1清0
[4]	BODRF	<b>BOD 欠压复位</b> 当电压低于BOD的设定值，又使能BOD复位时，此位将置1. 0 = 未发生欠压复位. 1 = 发生欠压复位. <b>注意:</b> 写1清0

位	描述	
[3]	<b>LVRF</b>	<b>LVR 低压复位</b> 0 = 未发生低压复位 1 = 发生低压复位 <b>注意:</b> 写1清0
[2]	<b>WDTRF</b>	<b>WDT看门狗复位</b> 0 = 未发生下面情况复位 1 = 看门狗或窗看门狗复位 <b>注意1:</b> 写1清0 <b>注意2:</b> 若是看门狗复位， RSTF(WDT_CTL[2]) 置1. 若是窗看门狗复位 WWDTRF (WWDT_STATUS[1]) 置1
[1]	<b>PINRF</b>	<b>NRESET引脚复位</b> 0 = 未发生下面情况复位 1 = 发生复位引脚nRESET拉低复位 <b>注意:</b> 写1清0
[0]	<b>PORF</b>	<b>上电复位</b> 芯片重新上电，或对 CHIPRST (SYS_IPRST0[0]) 写1软件使能芯片复位后，此位置1 0 = 未发生下面两种复位 1 = 上电复位或软件芯片复位. <b>注意:</b> 写1清0

**SYS\_IPRST0 外设复位控制寄存器0**

对某位写1对应外设异步复位，软件写回0后对应外设开始正常工作

寄存器	偏移	读/写	描述	复位值
SYS_IPRST0	SYS_BA+0x08	读/写	外设复位控制寄存器 0	0x0000_0000

31	30	29	28	27	26	25	24
ETMCRST	SWDCRST						Reserved
23	22	21	20	19	18	17	16
						SDH1RST	HSUSBHRST
15	14	13	12	11	10	9	8
Reserved	SPIMRST	Reserved	CRYPTORST	Reserved	HSUSBDRST		Reserved
7	6	5	4	3	2	1	0
CRCRST	SDH0RST	EMACRST	Reserved	EBIRST	PDMARST	CPURST	CHIPRST

位	描述	
[31]	ETMCRST	<b>ETM 复位控制</b> 0 = ETM 正常工作 1 = 让 ETM 复位
[30]	SWDCRST	<b>SWD复位控制</b> 0 = SWD 正常工作 1 = 让 SWD 复位
[29:18]	Reserved	保留.
[17]	SDH1RST	<b>SD卡接口1复位控制(写保护位)</b> 0 = SD卡接口1正常工作 1 = 让SD卡接口1 复位 <b>注意:</b> 此位的写是受保护的, 请参考寄存器 SYS_REGLCTL
[16]	HSUSBHRST	<b>高速USB主机端复位控制 (写保护位)</b> 0 = 高速 USB主机控制器正常工作 1 = 让高速 USB主机控制器复位 <b>注意:</b> 此位的写是受保护的, 请参考寄存器 SYS_REGLCTL
[15]	Reserved	保留.
[14]	SPIMRST	<b>SPIM复位控制</b> 0 = SPIM 正常工作 1 = 让 SPIM 复位
[13]	Reserved	保留.
[12]	CRYPTORST	<b>CRYPT 复位 (写保护位)</b> 0 = CRYPT正常工作

		1 = 让CRYPT复位 <b>注意:</b> 此位的写是受保护的, 请参考寄存器 SYS_REGLCTL
[11]	<b>Reserved</b>	保留.
[10]	<b>HSUSBDRST</b>	<b>高速USB设备复位(写保护位)</b> 0 = HSUSBD正常工作 1 = 让HSUSBD复位.
[9:8]	<b>Reserved</b>	保留.
[7]	<b>CRCRST</b>	<b>CRC复位(写保护位)</b> 0 = CRC正常工作. 1 = 让CRC运算模块复位 <b>注意:</b> 此位的写是受保护的, 请参考寄存器 SYS_REGLCTL
[6]	<b>SDH0RST</b>	<b>SD卡接口0复位(写保护位)</b> 0 = SDHOST0正常工作. 1 = 让SDHOST0复位 <b>注意:</b> 此位的写是受保护的, 请参考寄存器 SYS_REGLCTL.
[5]	<b>EMACRST</b>	<b>EMAC复位(写保护位)</b> 0 = EMAC正常工作. 1 = 让EMAC 复位 <b>注意:</b> 此位的写是受保护的, 请参考寄存器 SYS_REGLCTL
[4]	<b>Reserved</b>	保留.
[3]	<b>EBIRST</b>	<b>EBI复位(写保护位)</b> 0 = EBI 正常工作. 1 = 让 EBI 复位. <b>注意:</b> 此位的写是受保护的, 请参考寄存器 SYS_REGLCTL
[2]	<b>PDMARST</b>	<b>PDMA 复位(写保护位)</b> 0 = PDMA 正常工作 1 = 让 PDMA 复位 <b>注意:</b> 此位的写是受保护的, 请参考寄存器 SYS_REGLCTL
[1]	<b>CPURST</b>	<b>CPU复位(写保护位)</b> 0 = CPU正常工作 1 = 让CPU复位, 两个时钟后, 自动回0 <b>注意:</b> 此位的写是受保护的, 请参考寄存器 SYS_REGLCTL
[0]	<b>CHIPRST</b>	<b>芯片复位(写保护位)</b> CHIPRST和SYSRESETREQ(AIRCR[2])的不同, 请参考 7.2.2章节 0 = 芯片正常工作. 1 = 让芯片复位, 和重新上电一样, 两个时钟后此位自动回0. <b>注意:</b> 此位的写是受保护的, 请参考寄存器 SYS_REGLCTL

**SYS\_IPRST1 外设复位控制寄存器1**

对某位写1对应外设异步复位，软件写回0后对应外设才正常工作

寄存器	偏移	读/写	描述	复位值			
SYS_IPRST1	SYS_BA+0x0C	读/写	外设复位控制寄存器 1	0x0000_0000			

31	30	29	28	27	26	25	24
Reserved		I2S0RST	EADCRST	USBDRST	Reserved	CAN1RST	CAN0RST
23	22	21	20	19	18	17	16
Reserved		UART5RST	UART4RST	UART3RST	UART2RST	UART1RST	UART0RST
15	14	13	12	11	10	9	8
SPI2RST	SPI1RST	SPI0RST	QSPI0RST	Reserved	I2C2RST	I2C1RST	I2C0RST
7	6	5	4	3	2	1	0
ACMP01RST	Reserved	TMR3RST	TMR2RST	TMR1RST	TMR0RST	GPIORST	Reserved

位	描述	
[31:30]	Reserved	保留.
[29]	I2S0RST	<b>I<sup>2</sup>S0 复位</b> 0 = I <sup>2</sup> S0 正常工作 1 = 让 I <sup>2</sup> S0 复位.
[28]	EADCRST	<b>EADC 复位</b> 0 = EADC正常工作 1 = 让 EADC复位
[27]	USBDRST	<b>USBD 复位</b> 0 = USBD正常工作 1 = 让 USBD复位.
[26]	Reserved	保留.
[25]	CAN1RST	<b>CAN1 复位</b> 0 = CAN1正常工作 1 = 让 CAN1复位
[24]	CAN0RST	<b>CAN0 复位</b> 0 = CAN0正常工作 1 = 让 CAN0复位.
[23:22]	Reserved	保留.
[21]	UART5RST	<b>UART5 复位</b> 0 = UART5正常工作 1 = 让 UART5 复位

[20]	<b>UART4RST</b>	<b>UART4 复位</b> 0 = UART4正常工作 1 = 让 UART4复位
[19]	<b>UART3RST</b>	<b>UART3 复位</b> 0 = UART3正常工作 1 = 让 UART3 复位
[18]	<b>UART2RST</b>	<b>UART2 复位</b> 0 = UART2正常工作 1 = 让 UART2 复位.
[17]	<b>UART1RST</b>	<b>UART1 复位</b> 0 = UART1正常工作 1 = 让 UART1 复位
[16]	<b>UART0RST</b>	<b>UART0 复位</b> 0 = UART0正常工作 1 = 让 UART0 复位
[15]	<b>SPI2RST</b>	<b>SPI2 复位</b> 0 = SPI2正常工作. 1 = 让 SPI2 复位.
[14]	<b>SPI1RST</b>	<b>SPI1 复位</b> 0 = SPI1正常工作 1 = 让 SPI1 复位
[13]	<b>SPI0RST</b>	<b>SPI0 复位</b> 0 = SPI0正常工作 1 = 让 SPI0 复位
[12]	<b>QSPI0RST</b>	<b>QSPI0 复位</b> 0 = QSPI0正常工作. 1 = 让 QSPI0 复位
[11]	<b>Reserved</b>	保留.
[10]	<b>I2C2RST</b>	<b>I2C2 复位</b> 0 = I2C2 正常工作 1 =让 I2C2 复位.
[9]	<b>I2C1RST</b>	<b>I2C1 复位</b> 0 = I2C1 正常工作 1 =让 I2C1 复位.
[8]	<b>I2C0RST</b>	<b>I2C0 复位</b> 0 = I2C0正常工作. 1 =让 I2C0 复位.
[7]	<b>ACMP01RST</b>	<b>比较器复位</b> 0 = 比较器 0/1 正常工作 1 = 让比较器 0/1 复位.

[6]	<b>Reserved</b>	保留.
[5]	<b>TMR3RST</b>	<b>Timer3 复位</b> 0 = Timer3正常工作. 1 = 让 Timer3 复位
[4]	<b>TMR2RST</b>	<b>Timer2 复位</b> 0 = Timer2正常工作 1 = 让 Timer2 复位
[3]	<b>TMR1RST</b>	<b>Timer1 复位</b> 0 = Timer1正常工作 1 = 让 Timer1 复位
[2]	<b>TMR0RST</b>	<b>Timer0 复位</b> 0 = Timer0正常工作 1 = 让 Timer0 复位
[1]	<b>GPIORST</b>	<b>GPIO 复位</b> 0 = GPIO正常工作 1 = 让 GPIO 复位
[0]	<b>Reserved</b>	保留.

**SYS\_IPRST2 外设复位控制寄存器2**

对某位写1对应外设异步复位，软件写回0后对应外设才正常工作

寄存器	偏移	读/写	描述				复位值
SYS_IPRST2	SYS_BA+0x10	读/写	外设复位控制寄存器 2				0x0000_0000

31	30	29	28	27	26	25	24
Reserved	OPARST	Reserved		ECAP1RST	ECAP0RST	Reserved	
23	22	21	20	19	18	17	16
QEI1RST	QEI0RST	Reserved		BPWM1RST	BPWM0RST	EPWM1RST	EPWM0RST
15	14	13	12	11	10	9	8
Reserved			DACRST	Reserved		USCI1RST	USCI0RST
7	6	5	4	3	2	1	0
Reserved	SPI3RST	Reserved			SC2RST	SC1RST	SC0RST

位	描述	
[31]	Reserved	保留.
[30]	OPARST	运放 复位 0 = OPA 正常工作 1 = 让 OPA 复位
[29:28]	Reserved	保留.
[27]	ECAP1RST	ECAP1 复位 0 = ECAP1 正常工作 1 = 让 ECAP1 复位
[26]	ECAP0RST	ECAP0 复位 0 = ECAP0 正常工作 1 = 让 ECAP0 复位
[25:24]	Reserved	保留.
[23]	QEI1RST	QEI1 复位 0 = QEI1 正常工作 1 = 让 QEI1 复位
[22]	QEI0RST	QEI0 复位 0 = QEI0 正常工作 1 = 让 QEI0 复位
[21:20]	Reserved	保留.
[19]	BPWM1RST	BPWM1 复位 0 = BPWM1 正常工作

		1 = 让BPWM1 复位
[18]	<b>BPWM0RST</b>	<b>BPWM0 复位</b> 0 = BPWM0 正常工作 1 = 让 BPWM0 复位
[17]	<b>EPWM1RST</b>	<b>EPWM1 复位</b> 0 = EPWM1 正常工作 1 = 让 EPWM1 复位
[16]	<b>EPWM0RST</b>	<b>EPWM0 复位</b> 0 = EPWM0 正常工作 1 = 让 EPWM0 复位
[15:13]	<b>Reserved</b>	保留.
[12]	<b>DACRST</b>	<b>DAC 复位</b> 0 = DAC 正常工作 1 = 让 DAC 复位
[11:10]	<b>Reserved</b>	保留.
[9]	<b>USCI1RST</b>	<b>USCI1 复位</b> 0 = USCI1 正常工作 1 = 让 USCI1 复位
[8]	<b>USCI0RST</b>	<b>USCI0 复位</b> 0 = USCI0 正常工作 1 = 让 USCI0 复位
[7]	<b>Reserved</b>	保留.
[6]	<b>SPI3RST</b>	<b>SPI3 复位</b> 0 = SPI3 正常工作 1 = 让 SPI3 复位
[5:3]	<b>Reserved</b>	保留
[2]	<b>SC2RST</b>	<b>SC2 复位</b> 0 = SC2 正常工作 1 = 让 SC2 复位
[1]	<b>SC1RST</b>	<b>SC1 复位</b> 0 = SC1 正常工作 1 = 让 SC1复位.
[0]	<b>SC0RST</b>	<b>SC0 复位</b> 0 = SC0 正常工作 1 = 让 SC0 复位

**SYS\_BODCTL 欠压复位 BOD 控制寄存器**

寄存器	偏移	读/写	描述	复位值
SYS_BODCTL	SYS_BA+0x18	读/写	BOD 控制寄存器	0x000X_038X

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved					BODVL		
15	14	13	12	11	10	9	8
Reserved	LVRDGSEL			Reserved	BODDGSEL		
7	6	5	4	3	2	1	0
LVREN	BODOUT	BODLPM	BODIF	BODRSTEN	Reserved		BODEN

位	描述	
[31:19]	Reserved	保留.
[18:16]	BODVL	<p><b>BOD电压门限 (写保护位)</b></p> <p>上电缺省值由 CBOV (CONFIG0 [23:21]) 加载进来.</p> <p>000 = BOD电压 = 1.6V.      001 = BOD电压 = 1.8V.      010 = BOD电压 = 2.0V.      011 = BOD电压 = 2.2V.      100 = BOD电压 = 2.4V.      101 = BOD电压 = 2.6V.      110 = BOD电压 = 2.8V.      111 = BOD电压 = 3.0V.</p> <p><b>注意:</b> 此位的写是受保护的, 请参考寄存器 SYS_REGLCTL</p>
[15]	Reserved	保留.
[14:12]	LVRDGSEL	<p><b>LVR 防抖时间 (写保护位)</b></p> <p>000 = 无防抖 (缺省值)      001 = 防抖时间 4 HCLK.      010 = 防抖时间 8 HCLK.      011 = 防抖时间 16 HCLK      100 = 防抖时间 32 HCLK.      101 = 防抖时间 64 HCLK.      110 = 防抖时间 128 HCLK      111 = 防抖时间 256 HCLK.</p> <p><b>注意:</b> 此位的写是受保护的, 请参考寄存器 SYS_REGLCTL</p>
[11]	Reserved	保留.

位	描述
[10:8]	<b>BODDGSEL</b>  <b>BOD 防抖时间 (写保护位)</b> 000 = BOD 防抖时间一个 RC10K 时钟. 001 = 防抖时间 4 HCLK. 010 = 防抖时间 8 HCLK 011 = 防抖时间 16 HCLK 100 = 防抖时间 32 HCLK 101 = 防抖时间 64 HCLK 110 = 防抖时间 128 HCLK. 111 = 防抖时间 256 HCLK <b>注意:</b> 此位的写是受保护的, 请参考寄存器 SYS_REGLCTL
[7]	<b>LVREN</b>  <b>LVR低压复位使能 (写保护位)</b> 0 = 关 LVR 低压复位功能. 1 = 使能低压复位功能. (缺省) <b>注意1:</b> LVR使能后, 将在LVR输出稳定100us后有效(缺省状态) <b>注意2:</b> 此位的写是受保护的, 请参考寄存器 SYS_REGLCTL
[6]	<b>BODOUT</b>  <b>BOD检测电路的输出值</b> 0 = 输出值为 0。这表明V <sub>DD</sub> 高于BODVL的设定值, 或者BOD被禁止 1 = BOD输出 1. 表明V <sub>DD</sub> 低于BODVL的设定值
[5]	<b>BODLPM</b>  <b>BOD低耗电模式 (写保护位)</b> 0 = BOD 正常模式(缺省) 耗电约100uA. 1 = BOD 低耗电模式, 耗电约10uA. <b>注意1:</b> 此位的写是受保护的, 请参考寄存器 SYS_REGLCTL
[4]	<b>BODIF</b>  <b>BOD中断标志</b> 0 = 未发生下述情况 1 = 当BOD电路检测到V <sub>DD</sub> 由高到低跨过BODVL设定电压时, 或者由低到高跨过时, 此位置1, 若使能了中断, 会申请中断。 <b>注意:</b> 写1清 0.
[3]	<b>BODRSTEN</b>  <b>BOD复位使能 (写保护位)</b> 缺省值由CBORST(CONFIG0[20])加载进来 . 0 = BOD输出触发中断. 1 = BOD输出会复位系统. <b>注意1:</b> 若BODEN =1使能了BOD, 并且 BODRSTEN =1使能了BOD复位功能, BOD电路检测到欠压时(BODOUT输出高)会复位芯片。 若BODEN =1使能了BOD, 并且 BODRSTEN =0使能了BOD中断功能, BOD电路检测到欠压时(BODOUT输出高)将申请中断直到BODEN=0关闭BOD电路, 或者关闭BOD中断向量 <b>注意2:</b> 此位的写是受保护的, 请参考寄存器 SYS_REGLCTL
[2:1]	<b>Reserved</b> 保留.

位	描述	
[0]	<b>BODEN</b>	<b>BOD使能 (写保护位)</b> 缺省值由 CBODEN (CONFIG0 [19])加载进来 0 = BOD被禁止. 1 = 使能BOD. <b>注意:</b> 此位的写是受保护的, 请参考寄存器 SYS_REGLCTL

SYS\_IVSCTL 片内电压控制

寄存器	偏移	读/写	描述	复位值
SYS_IVSCTL	SYS_BA+0x1C	读/写	片内电压控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved						VBATUGEN	VTEMPEN

位	描述	
[31:8]	Reserved	保留.
[7:2]	Reserved	保留.
[1]	VBATUGEN	<b>VDD 缓冲使能</b> 0 = 关闭 VDD 缓冲 (缺省). 1 = 使能 VDD 缓冲, 由 ADC 测VDD时, 必须使能
[0]	VTEMPEN	<b>温度传感器使能</b> 0 = 温度传感器关闭 (缺省). 1 = 使能温度传感器, 可从GPC.9得到输出值.

**SYS PORCTL 上电复位控制寄存器**

寄存器	偏移	读/写	描述	复位值
SYS_PORCTL	SYS_BA+0x24	读/写	上电复位控制寄存器	0x0000_00XX

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
POROFF							
7	6	5	4	3	2	1	0
POROFF							

位	描述	
[31:16]	Reserved	保留.
[15:0]	POROFF	<p><b>上电复位使能位(写保护位)</b>            上电后，POR电路给一个复位信号，芯片开始工作，但干扰可能让POR再次复位芯片，向此地址写0x5AA5就关闭了POR复位功能。</p> <p>向此地址写其它值，或芯片因其它原因复位后，POR复位功能又被使能。其它复位源包括：nRESET复位，看门狗复位，LVR低压复位，BOD欠压复位，ICE 仿真器复位，或软件复位</p> <p><b>注意：</b>此位的写是受保护的，请参考寄存器 SYS_REGLCTL</p>

**SYS\_VREFCTL 参考电压控制寄存器**

寄存器	偏移	读/写	描述	复位值
SYS_VREFCTL	SYS_BA+0x28	读/写	VREF 参考电压控制寄存器	0x0000_0200

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
PRELOAD_SEL		Reserved		VREFCTL			

位	描述	
[31:8]	Reserved	保留.
[7:6]	PRELOAD_SEL	预加载时间(写保护位) 00 = 60us , 0.1uF 电容 01 = 310us, 1uF电容 10 = 1270us, 4.7uF电容 11 = 2650us, 10uF电容
[5]	Reserved	保留.
[4:0]	VREFCTL	V <sub>REF</sub> 电压选择(写保护位) 00000 = V <sub>REF</sub> 由外接电压决定. 00011 = V <sub>REF</sub> 是内部 1.6V. 00111 = V <sub>REF</sub> 是内部2.0V. 01011 = V <sub>REF</sub> 是内部2.5V. 01111 = V <sub>REF</sub> 是内部3.0V. 其他 = 保留. <b>注意:</b> 此位的写是受保护的, 请参考寄存器 SYS_REGLCTL

**SYS USBPHY USBPHY控制寄存器**

寄存器	偏移	读/写	描述	复位值
SYS_USBPHY	SYS_BA+0x2C	读/写	USB PHY控制寄存器	0x0003_0007

31	30	29	28	27	26	25	24
Reserved						HSUSBACT	HSUSBEN
23	22	21	20	19	18	17	16
Reserved						HSUSBROLE	
15	14	13	12	11	10	9	8
Reserved						USBEN	
7	6	5	4	3	2	1	0
Reserved						SBO	USBROLE

位	描述	
[31:26]	Reserved	保留.
[25]	HSUSBACT	<p><b>HSUSB PHY 激活控制</b>            该位用于控制HSUSB PHY 在复位状态还是激活状态            0 = HSUSB PHY 在复位状态.            1 = HSUSB PHY 在激活状态.  <b>注意:</b> 在设置 HSUSBEN (SYS_USBPHY[24]) 使能 HSUSB PHY后，用户应该在改成激活模式前，保持HSUSB PHY 在复位模式至少10uS 。</p>
[24]	HSUSBEN	<p><b>HSUSB PHY 使能(写保护位)</b>            该位用于使能/禁止 HSUSB PHY.            0 = HSUSB PHY 禁止.            1 = HSUSB PHY 使能.  <b>注意:</b> 此位的写是受保护的，请参考寄存器 SYS_REGLCTL</p>
[23:18]	Reserved	保留.
[17:16]	HSUSBROLE	<p><b>HSUSB 模式选择(写保护位)</b>            00 = 标准 HS USB 设备模式            01 = 标准 HS USB 主机模式.            10 = 由 ID 引脚决定.            11 = 保留.  <b>注意:</b> 此位的写是受保护的，请参考寄存器 SYS_REGLCTL</p>
[15:9]	Reserved	保留.
[8]	USBEN	<p><b>USB PHY</b>            0 = USB PHY 禁止.            1 = USB PHY 使能.  <b>注意:</b> 此位的写是受保护的，请参考寄存器 SYS_REGLCTL</p>

[7:3]	<b>Reserved</b>	保留.
[2]	<b>SBO</b>	<b>注意:</b> 此位写1, 否则结果未知
[1:0]	<b>USBROLE</b>	<b>USB模式选择 (写保护位)</b> 00 = 标准 USB 设备模式 01 = 标准 USB 主机模式. 10 = 由 ID 引脚决定. 11 = 保留. <b>注意:</b> 此位的写是受保护的, 请参考寄存器 SYS_REGLCTL

**SYS GPA MFPL GPIOA低位功能选择**

寄存器	偏移	读/写	描述	复位值
SYS_GPA_MFPL	SYS_BA+0x30	读/写	GPIOA 低位功能选择	0x0000_0000

31	30	29	28	27	26	25	24
PA7MFP				PA6MFP			
23	22	21	20	19	18	17	16
PA5MFP				PA4MFP			
15	14	13	12	11	10	9	8
PA3MFP				PA2MFP			
7	6	5	4	3	2	1	0
PA1MFP				PA0MFP			

位	描述	
[31:28]	PA7MFP	PA.7多功能选择
[27:24]	PA6MFP	PA.6多功能选择
[23:20]	PA5MFP	PA.5多功能选择
[19:16]	PA4MFP	PA.4多功能选择
[15:12]	PA3MFP	PA.3多功能选择
[11:8]	PA2MFP	PA.2多功能选择
[7:4]	PA1MFP	PA.1多功能选择
[3:0]	PA0MFP	PA.0多功能选择

**SYS GPA MFPH GPIOA高位功能选择**

寄存器	偏移	读/写	描述	复位值
SYS_GPA_MFPH	SYS_BA+0x34	读/写	GPIOA 高位功能选择	0x0000_0000

31	30	29	28	27	26	25	24
PA15MFP				PA14MFP			
23	22	21	20	19	18	17	16
PA13MFP				PA12MFP			
15	14	13	12	11	10	9	8
PA11MFP				PA10MFP			
7	6	5	4	3	2	1	0
PA9MFP				PA8MFP			

位	描述	
[31:28]	PA15MFP	PA.15多功能选择
[27:24]	PA14MFP	PA.14多功能选择
[23:20]	PA13MFP	PA.13多功能选择
[19:16]	PA12MFP	PA.12多功能选择
[15:12]	PA11MFP	PA.11多功能选择
[11:8]	PA10MFP	PA.10多功能选择
[7:4]	PA9MFP	PA.9多功能选择
[3:0]	PA8MFP	PA.8多功能选择

**SYS\_GPB\_MFPL GPIOB低位功能选择**

寄存器	偏移	读/写	描述	复位值
SYS_GPB_MFPL	SYS_BA+0x38	读/写	GPIOB低位功能选择	0x0000_0000

31	30	29	28	27	26	25	24
PB7MFP				PB6MFP			
23	22	21	20	19	18	17	16
PB5MFP				PB4MFP			
15	14	13	12	11	10	9	8
PB3MFP				PB2MFP			
7	6	5	4	3	2	1	0
PB1MFP				PB0MFP			

位	描述	
[31:28]	PB7MFP	PB.7多功能选择
[27:24]	PB6MFP	PB.6多功能选择
[23:20]	PB5MFP	PB.5多功能选择
[19:16]	PB4MFP	PB.4多功能选择
[15:12]	PB3MFP	PB.3多功能选择
[11:8]	PB2MFP	PB.2多功能选择
[7:4]	PB1MFP	PB.1多功能选择
[3:0]	PB0MFP	PB.0多功能选择

**SYS\_GPB\_MFPH GPIOB高位功能选择**

寄存器	偏移	读/写	描述	复位值
SYS_GPB_MFPH	SYS_BA+0x3C	读/写	GPIOB高位功能选择	0x0000_0000

31	30	29	28	27	26	25	24
PB15MFP				PB14MFP			
23	22	21	20	19	18	17	16
PB13MFP				PB12MFP			
15	14	13	12	11	10	9	8
PB11MFP				PB10MFP			
7	6	5	4	3	2	1	0
PB9MFP				PB8MFP			

位	描述	
[31:28]	PB15MFP	PB.15多功能选择
[27:24]	PB14MFP	PB.14多功能选择
[23:20]	PB13MFP	PB.13多功能选择
[19:16]	PB12MFP	PB.12多功能选择
[15:12]	PB11MFP	PB.11多功能选择
[11:8]	PB10MFP	PB.10多功能选择
[7:4]	PB9MFP	PB.9多功能选择
[3:0]	PB8MFP	PB.8多功能选择

SYS\_GPC\_MFPL GPIOC低位功能选择

寄存器	偏移	读/写	描述	复位值
SYS_GPC_MFPL	SYS_BA+0x40	读/写	GPIOC低位功能选择	0x0000_0000

31	30	29	28	27	26	25	24
PC7MFP				PC6MFP			
23	22	21	20	19	18	17	16
PC5MFP				PC4MFP			
15	14	13	12	11	10	9	8
PC3MFP				PC2MFP			
7	6	5	4	3	2	1	0
PC1MFP				PC0MFP			

位	描述	
[31:28]	PC7MFP	PC.7多功能选择
[27:24]	PC6MFP	PC.6多功能选择
[23:20]	PC5MFP	PC.5多功能选择
[19:16]	PC4MFP	PC.4多功能选择
[15:12]	PC3MFP	PC.3多功能选择
[11:8]	PC2MFP	PC.2多功能选择
[7:4]	PC1MFP	PC.1多功能选择
[3:0]	PC0MFP	PC.0多功能选择

**SYS\_GPC\_MFPH GPIOC高位功能选择**

寄存器	偏移	读/写	描述	复位值
SYS_GPC_MFPH	SYS_BA+0x44	读/写	GPIOC高位功能选择	0x0000_0000

31	30	29	28	27	26	25	24
PC15MFP				PC14MFP			
23	22	21	20	19	18	17	16
PC13MFP				PC12MFP			
15	14	13	12	11	10	9	8
PC11MFP				PC10MFP			
7	6	5	4	3	2	1	0
PC9MFP				PC8MFP			

位	描述	
[31:28]	PC15MFP	PC.15多功能选择
[27:24]	PC14MFP	PC.14多功能选择
[23:20]	PC13MFP	PC.13多功能选择
[19:16]	PC12MFP	PC.12多功能选择
[15:12]	PC11MFP	PC.11多功能选择
[11:8]	PC10MFP	PC.10多功能选择
[7:4]	PC9MFP	PC.9多功能选择
[3:0]	PC8MFP	PC.8多功能选择

**SYS\_GPD\_MFPL GPIOD低位功能选择**

寄存器	偏移	读/写	描述	复位值
SYS_GPD_MFPL	SYS_BA+0x48	读/写	GPIOD低位功能选择	0x0000_0000

31	30	29	28	27	26	25	24
PD7MFP				PD6MFP			
23	22	21	20	19	18	17	16
PD5MFP				PD4MFP			
15	14	13	12	11	10	9	8
PD3MFP				PD2MFP			
7	6	5	4	3	2	1	0
PD1MFP				PD0MFP			

位	描述	
[31:28]	PD7MFP	PD.7多功能选择
[27:24]	PD6MFP	PD.6多功能选择
[23:20]	PD5MFP	PD.5多功能选择
[19:16]	PD4MFP	PD.4多功能选择
[15:12]	PD3MFP	PD.3多功能选择
[11:8]	PD2MFP	PD.2多功能选择
[7:4]	PD1MFP	PD.1多功能选择
[3:0]	PD0MFP	PD.0多功能选择

**SYS\_GPD\_MFPH GPIOD高位功能选择**

寄存器	偏移	读/写	描述	复位值
SYS_GPD_MFPH	SYS_BA+0x4C	读/写	GPIOD高位功能选择	0x0000_0000

31	30	29	28	27	26	25	24
PD15MFP				PD14MFP			
23	22	21	20	19	18	17	16
PD13MFP				PD12MFP			
15	14	13	12	11	10	9	8
PD11MFP				PD10MFP			
7	6	5	4	3	2	1	0
PD9MFP				PD8MFP			

位	描述	
[31:28]	PD15MFP	PD.15多功能选择
[27:24]	PD14MFP	PD.14多功能选择
[23:20]	PD13MFP	PD.13多功能选择
[19:16]	PD12MFP	PD.12多功能选择
[15:12]	PD11MFP	PD.11多功能选择
[11:8]	PD10MFP	PD.10多功能选择
[7:4]	PD9MFP	PD.9多功能选择
[3:0]	PD8MFP	PD.8多功能选择

**SYS\_GPE\_MFPL GPIOE低位功能选择**

寄存器	偏移	读/写	描述	复位值
SYS_GPE_MFPL	SYS_BA+0x50	读/写	GPIOE低位功能选择	0x0000_0000

31	30	29	28	27	26	25	24
PE7MFP				PE6MFP			
23	22	21	20	19	18	17	16
PE5MFP				PE4MFP			
15	14	13	12	11	10	9	8
PE3MFP				PE2MFP			
7	6	5	4	3	2	1	0
PE1MFP				PE0MFP			

位	描述	
[31:28]	PE7MFP	PE.7多功能选择
[27:24]	PE6MFP	PE.6多功能选择
[23:20]	PE5MFP	PE.5多功能选择
[19:16]	PE4MFP	PE.4多功能选择
[15:12]	PE3MFP	PE.3多功能选择
[11:8]	PE2MFP	PE.2多功能选择
[7:4]	PE1MFP	PE.1多功能选择
[3:0]	PE0MFP	PE.0多功能选择

**SYS\_GPE\_MFPH GPIOE高位功能选择**

寄存器	偏移	读/写	描述	复位值
SYS_GPE_MFPH	SYS_BA+0x54	读/写	GPIOE高位功能选择	0x0000_0000

31	30	29	28	27	26	25	24
PE15MFP				PE14MFP			
23	22	21	20	19	18	17	16
PE13MFP				PE12MFP			
15	14	13	12	11	10	9	8
PE11MFP				PE10MFP			
7	6	5	4	3	2	1	0
PE9MFP				PE8MFP			

位	描述	
[31:28]	PE15MFP	PE.15多功能选择
[27:24]	PE14MFP	PE.14多功能选择
[23:20]	PE13MFP	PE.13多功能选择
[19:16]	PE12MFP	PE.12多功能选择
[15:12]	PE11MFP	PE.11多功能选择
[11:8]	PE10MFP	PE.10多功能选择
[7:4]	PE9MFP	PE.9多功能选择
[3:0]	PE8MFP	PE.8多功能选择

SYS\_GPF\_MFPL GPIOF低位功能选择

寄存器	偏移	读/写	描述	复位值
SYS_GPF_MFPL	SYS_BA+0x58	读/写	GPIOF低位功能选择	0x0000_00EE

31	30	29	28	27	26	25	24
PF7MFP				PF6MFP			
23	22	21	20	19	18	17	16
PF5MFP				PF4MFP			
15	14	13	12	11	10	9	8
PF3MFP				PF2MFP			
7	6	5	4	3	2	1	0
PF1MFP				PF0MFP			

位	描述	
[31:28]	PF7MFP	PF.7多功能选择
[27:24]	PF6MFP	PF.6多功能选择
[23:20]	PF5MFP	PF.5多功能选择
[19:16]	PF4MFP	PF.4多功能选择
[15:12]	PF3MFP	PF.3多功能选择
[11:8]	PF2MFP	PF.2多功能选择
[7:4]	PF1MFP	PF.1多功能选择
[3:0]	PF0MFP	PF.0多功能选择

**SYS\_GPF\_MFPH GPIOF高位功能选择**

寄存器	偏移	读/写	描述	复位值
SYS_GPF_MFPH	SYS_BA+0x5C	读/写	GPIOF高位功能选择	0x0000_0000

31	30	29	28	27	26	25	24
PF15MFP				PF14MFP			
23	22	21	20	19	18	17	16
PF13MFP				PF12MFP			
15	14	13	12	11	10	9	8
PF11MFP				PF10MFP			
7	6	5	4	3	2	1	0
PF9MFP				PF8MFP			

位	描述	
[31:28]	PF15MFP	PF.15多功能选择
[27:24]	PF14MFP	PF.14多功能选择
[23:20]	PF13MFP	PF.13多功能选择
[19:16]	PF12MFP	PF.12多功能选择
[15:12]	PF11MFP	PF.11多功能选择
[11:8]	PF10MFP	PF.10多功能选择
[7:4]	PF9MFP	PF.9多功能选择
[3:0]	PF8MFP	PF.8多功能选择

SYS\_GPG\_MFPL GPIOG低位功能选择

寄存器	偏移	读/写	描述	复位值
SYS_GPG_MFP_L	SYS_BA+0x60	读/写	GPIOG低位功能选择	0x0000_0000

31	30	29	28	27	26	25	24
PG7MFP				PG6MFP			
23	22	21	20	19	18	17	16
PG5MFP				PG4MFP			
15	14	13	12	11	10	9	8
PG3MFP				PG2MFP			
7	6	5	4	3	2	1	0
PG1MFP				PG0MFP			

位	描述	
[31:28]	PG7MFP	PG.7多功能选择
[27:24]	PG6MFP	PG.6多功能选择
[23:20]	PG5MFP	PG.5多功能选择
[19:16]	PG4MFP	PG.4多功能选择
[15:12]	PG3MFP	PG.3多功能选择
[11:8]	PG2MFP	PG.2多功能选择
[7:4]	PG1MFP	PG.1多功能选择
[3:0]	PG0MFP	PG.0多功能选择

**SYS\_GPG\_MFPH GPIOG高位功能选择**

寄存器	偏移	读/写	描述	复位值
SYS_GPG_MFPH	SYS_BA+0x64	读/写	GPIOG高位功能选择	0x0000_0000

31	30	29	28	27	26	25	24
PG15MFP				PG14MFP			
23	22	21	20	19	18	17	16
PG13MFP				PG12MFP			
15	14	13	12	11	10	9	8
PG11MFP				PG10MFP			
7	6	5	4	3	2	1	0
PG9MFP				PG8MFP			

位	描述	
[31:28]	PG15MFP	PG.15多功能选择
[27:24]	PG14MFP	PG.14多功能选择
[23:20]	PG13MFP	PG.13多功能选择
[19:16]	PG12MFP	PG.12多功能选择
[15:12]	PG11MFP	PG.11多功能选择
[11:8]	PG10MFP	PG.10多功能选择
[7:4]	PG9MFP	PG.9多功能选择
[3:0]	PG8MFP	PG.8多功能选择

SYS\_GPH\_MFPL GPIOH低位功能选择

寄存器	偏移	读/写	描述	复位值
SYS_GPH_MFPL	SYS_BA+0x68	读/写	GPIOH低位功能选择	0x0000_0000

31	30	29	28	27	26	25	24
PH7MFP				PH6MFP			
23	22	21	20	19	18	17	16
PH5MFP				PH4MFP			
15	14	13	12	11	10	9	8
PH3MFP				PH2MFP			
7	6	5	4	3	2	1	0
PH1MFP				PH0MFP			

位	描述	
[31:28]	PH7MFP	PH.7多功能选择
[27:24]	PH6MFP	PH.6多功能选择
[23:20]	PH5MFP	PH.5多功能选择
[19:16]	PH4MFP	PH.4多功能选择
[15:12]	PH3MFP	PH.3多功能选择
[11:8]	PH2MFP	PH.2多功能选择
[7:4]	PH1MFP	PH.1多功能选择
[3:0]	PH0MFP	PH.0多功能选择

**SYS\_GPH\_MFPH高位功能选择**

寄存器	偏移	读/写	描述	复位值
SYS_GPH_MFPH	SYS_BA+0x6C	读/写	GPIOH 高位功能选择	0x0000_0000

31	30	29	28	27	26	25	24
PH15MFP				PH14MFP			
23	22	21	20	19	18	17	16
PH13MFP				PH12MFP			
15	14	13	12	11	10	9	8
PH11MFP				PH10MFP			
7	6	5	4	3	2	1	0
PH9MFP				PH8MFP			

位	描述	
[31:28]	PH15MFP	PH.15多功能选择
[27:24]	PH14MFP	PH.14多功能选择
[23:20]	PH13MFP	PH.13多功能选择
[19:16]	PH12MFP	PH.12多功能选择
[15:12]	PH11MFP	PH.11多功能选择
[11:8]	PH10MFP	PH.10多功能选择
[7:4]	PH9MFP	PH.9多功能选择
[3:0]	PH8MFP	PH.8多功能选择

SYS\_GPx\_MFOS 引脚输出模式选择

寄存器	偏移	读/写	描述	复位值
SYS_GPA_MFOS	SYS_BA+0x80	读/写	GPIOA引脚输出模式选择	0x0000_0000
SYS_GPB_MFOS	SYS_BA+0x84	读/写	GPIOB引脚输出模式选择	0x0000_0000
SYS_GPC_MFOS	SYS_BA+0x88	读/写	GPIOC引脚输出模式选择	0x0000_0000
SYS_GPD_MFOS	SYS_BA+0x8C	读/写	GPIOD引脚输出模式选择	0x0000_0000
SYS_GPE_MFOS	SYS_BA+0x90	读/写	GPIOE引脚输出模式选择	0x0000_0000
SYS_GPF_MFOS	SYS_BA+0x94	读/写	GPIOF引脚输出模式选择	0x0000_0000
SYS_GPG_MFOS	SYS_BA+0x98	读/写	GPIOG引脚输出模式选择	0x0000_0000
SYS_GPH_MFOS	SYS_BA+0x9C	读/写	GPIOH引脚输出模式选择	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
MFOS							
7	6	5	4	3	2	1	0
MFOS							

位	描述	
[31:16]	Reserved	保留.
[n] n=0,1..15	MFOS	<b>GPIOA-h Pin[n] 引脚输出模式选择</b> 0 = 引脚 为推挽输出模式 1 = 引脚 为开漏输出模式. <b>注意:</b> 对于 A/B/E/G, n最大=15 对于 C/D, n最大=14. 对于 F/H, n最大=11

SYS\_SRAM\_INTCTL SRAM奇偶校验错误中断使能控制

寄存器	偏移	读/写	描述				复位值
SYS_SRAM_INTCTL	SYS_BA+0xC0	读/写	SRAM 奇偶校验错误中断使能控制				0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved							PERRIEN

位	描述	
[31:1]	Reserved	保留.
[0]	PERRIEN	<b>SRAM 奇偶校验错误中断使能控制</b> 0 = 禁止SRAM奇偶校验出错中断. 1 = 使能 SRAM奇偶校验出错中断.

**SYS SRAM STATUS SRAM奇偶校验状态**

寄存器	偏移	读/写	描述	复位值
SYS_SRAM_STATUS	SYS_BA+0xC4	读/写	SRAM奇偶校验状态	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved							PERRIF

位	描述	
[31:1]	Reserved	保留.
[0]	PERRIF	<b>SRAM 奇偶校验错误标志</b> 0 = SRAM 无奇偶校验错. 1 = SRAM奇偶校验出错. <b>注意:</b> 写1清0

**SYS\_SRAM\_ERRADDR SRAM奇偶校验出错地址**

寄存器	偏移	读/写	描述	复位值
SYS_SRAM_ERRADDR	SYS_BA+0xC8	只读	SRAM奇偶校验出错地址	0x0000_0000

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

位	描述
[31:0]	ERRADDR 系统 SRAM 校验出错地址

**SYS SRAM BISTCTL SRAM自测控制**

寄存器	偏移	读/写	描述	复位值
SYS_SRAM_BIS TCTL	SYS_BA+0xD0	读/写	SRAM自测控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
SRB1S5	SRB1S4	SRB1S3	SRB1S2	SRB1S1	SRB1S0	SRB0S1	SRB0S0
15	14	13	12	11	10	9	8
Reserved						HSUSBHBIST	HSUSBDBIST
7	6	5	4	3	2	1	0
PDMABIST	EMCBIST	SPIMBIST	USBBIST	CANBIST	CRBIST	SRBIST1	SRBIST0

位	描述	
[31:24]	Reserved	保留.
[23]	SRB1S5	<p><b>SRAM Bank1 5区选中自测 (写保护位)</b>            0 = 自测未选中 SRAM bank1 第6个 16KB            1 = 自测选中 SRAM bank1 第6个 16KB  <b>注意:</b> 此位的写是受保护的, 请参考寄存器 SYS_REGLCTL  <b>注意:</b> 自测时至少选中一个16K区域.</p>
[22]	SRB1S4	<p><b>SRAM Bank1 4区选中自测 (写保护位)</b>            0 = 自测未选中 SRAM bank1 第5个 16KB            1 = 自测选中 SRAM bank1 第5个 16KB  <b>注意:</b> 此位的写是受保护的, 请参考寄存器 SYS_REGLCTL  <b>注意:</b> 自测时至少选中一个16K区域.</p>
[21]	SRB1S3	<p><b>SRAM Bank1 3区选中自测 (写保护位)</b>            0 = 自测未选中 SRAM bank1 第4个 16KB            1 = 自测选中 SRAM bank1 第4个 16KB  <b>注意:</b> 此位的写是受保护的, 请参考寄存器 SYS_REGLCTL  <b>注意:</b> 自测时至少选中一个16K区域.</p>
[20]	SRB1S2	<p><b>SRAM Bank1 2区选中自测 (写保护位)</b>            0 = 自测未选中 SRAM bank1 第3个 16KB            1 = 自测选中 SRAM bank1 第3个 16KB  <b>注意:</b> 此位的写是受保护的, 请参考寄存器 SYS_REGLCTL  <b>注意:</b> 自测时至少选中一个16K区域.</p>
[19]	SRB1S1	<p><b>SRAM Bank1 1区选中自测 (写保护位)</b>            0 = 自测未选中 SRAM bank1 第2个 16KB</p>

		1 = 自测选中 SRAM bank1 第2个 16KB <b>注意:</b> 此位的写是受保护的, 请参考寄存器 SYS_REGLCTL <b>注意:</b> 自测时至少选中一个16K区域.
[18]	<b>SRB1S0</b>	<b>SRAM Bank1 0区选中自测 (写保护位)</b> 0 = 自测未选中 SRAM bank1 第1个 16KB 1 = 自测选中 SRAM bank1 第1个 16KB <b>注意:</b> 此位的写是受保护的, 请参考寄存器 SYS_REGLCTL <b>注意:</b> 自测时至少选中一个16K区域.
[17]	<b>SRB0S1</b>	<b>SRAM Bank0 1区选中自测 (写保护位)</b> 0 = 自测未选中 SRAM bank1 第2个 16KB 1 = 自测选中 SRAM bank1 第2个 16KB <b>注意:</b> 此位的写是受保护的, 请参考寄存器 SYS_REGLCTL <b>注意:</b> 自测时至少选中一个16K区域.
[16]	<b>SRB0S0</b>	<b>SRAM Bank0 0区选中自测 (写保护位)</b> 0 = 自测未选中 SRAM bank1 第1个 16KB 1 = 自测选中 SRAM bank1 第1个 16KB <b>注意:</b> 此位的写是受保护的, 请参考寄存器 SYS_REGLCTL <b>注意:</b> 自测时至少选中一个16K区域.
[15:10]	<b>Reserved</b>	保留.
[9]	<b>HSUSBHBIST</b>	<b>HSUSBH自测使能位 (写保护位)</b> 0 = 关闭HSUSBH RAM 自测. 1 = 开启HSUSBH RAM 自测 <b>注意:</b> 此位的写是受保护的, 请参考寄存器 SYS_REGLCTL
[8]	<b>HSUSBDDBIST</b>	<b>HSUSBD自测使能位 (写保护位)</b> 0 = 关闭HSUSBD RAM 自测. 1 = 开启HSUSBD RAM 自测 <b>注意:</b> 此位的写是受保护的, 请参考寄存器 SYS_REGLCTL
[7]	<b>PDMABIST</b>	<b>PDMA 自测使能位 (写保护位)</b> 0 = 关闭PDMA RAM 自测. 1 = 开启 PDMA RAM 自测 <b>注意:</b> 此位的写是受保护的, 请参考寄存器 SYS_REGLCTL
[6]	<b>EMCBIST</b>	<b>EMC 自测使能位 (写保护位)</b> 0 = 关闭EMC RAM 自测. 1 = 开启 EMC RAM 自测. <b>注意:</b> 此位的写是受保护的, 请参考寄存器 SYS_REGLCTL.
[5]	<b>SPIMBIST</b>	<b>SPIM 自测使能位 (写保护位)</b> 0 = 关闭SPIM RAM 自测. 1 = 开启 SPIM RAM 自测. <b>注意:</b> 此位的写是受保护的, 请参考寄存器 SYS_REGLCTL
[4]	<b>USBBIST</b>	<b>USB 自测使能位 (写保护位)</b> 0 = 关闭 USB RAM 自测.

		1 =开启 USB RAM 自测. <b>注意:</b> 此位的写是受保护的, 请参考寄存器 SYS_REGLCTL
[3]	<b>CANBIST</b>	<b>CAN</b> 自测使能位 (写保护位) 0 =关闭 CAN RAM 自测.. 1 =开启 CAN RAM 自测. <b>注意:</b> 此位的写是受保护的, 请参考寄存器 SYS_REGLCTL
[2]	<b>CRBIST</b>	<b>CACHE</b> 自测使能位 (写保护位) 0 =关闭 CACHE RAM 自测. 1 =开启 CACHE RAM 自测. <b>注意:</b> 此位的写是受保护的, 请参考寄存器 SYS_REGLCTL
[1]	<b>SRBIST1</b>	<b>SRAM</b> 自测使能位 (写保护位) 0 =关闭 SRAM bank1 RAM 自测. 1 =开启 SRAM bank1 RAM 自测. <b>注意:</b> 此位的写是受保护的, 请参考寄存器 SYS_REGLCTL
[0]	<b>SRBIST0</b>	<b>SRAM</b> 自测使能位(写保护位) 0 = 关闭 SRAM bank0 RAM 自测. 1 = 开启 SRAM bank0 RAM 自测. <b>注意:</b> 此位的写是受保护的, 请参考寄存器 SYS_REGLCTL

**SYS SRAM\_BISTSTS SRAM自测状态**

寄存器	偏移	读/写	描述	复位值
SYS_SRAM_BISTSTS	SYS_BA+0xD4	只读	系统 SRAM 自测状态寄存器	0x00xx_00xx

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved			USBBEND	CANBEND	CRBEND	SRBEND1	SRBEND0
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved			USBBEF	CANBEF	CRBISTEF	SRBISTEF1	SRBISTEF0

位	描述	
[31:21]	Reserved	保留.
[20]	USBBEND	<b>USB SRAM 自测结束</b> 0 = USB SRAM 正在自测 1 = USB SRAM 自测结束.
[19]	CANBEND	<b>CAN SRAM 自测结束</b> 0 = CAN SRAM 正在自测. 1 = CAN SRAM 自测结束.
[18]	CRBEND	<b>CACHE SRAM 自测结束</b> 0 = CACHE RAM 正在自测 1 = CACHE RAM 自测结束.
[17]	SRBEND1	<b>2<sup>nd</sup> SRAM 自测结束</b> 0 = 2 <sup>nd</sup> SRAM 正在自测. 1 = 2 <sup>nd</sup> SRAM 自测结束.
[16]	SRBEND0	<b>1<sup>st</sup> SRAM 自测结束</b> 0 = 1 <sup>st</sup> SRAM 正在自测 1 = 1 <sup>st</sup> SRAM 自测结束.
[15:5]	Reserved	保留.
[4]	USBBEF	<b>USB SRAM 自测失败</b> 0 = USB SRAM 自测通过, 没有错误. 1 = USB SRAM 自测失败
[3]	CANBEF	<b>CAN SRAM 自测失败</b> 0 = CAN SRAM 自测通过, 没有错误

		1 = CAN SRAM 自测失败.
[2]	<b>CRBISTEF</b>	<b>CACHE SRAM</b> 自测失败 0 = CACHE RAM 自测通过, 没有错误 1 = CACHE RAM 自测失败.
[1]	<b>SRBISTEF1</b>	<b>2<sup>nd</sup> SRAM</b> 自测失败 0 = 2nd SRAM 自测通过, 没有错误. 1 = 2nd SRAM BIST 自测失败
[0]	<b>SRBISTEF0</b>	<b>1<sup>st</sup> SRAM</b> 自测失败 0 = 1 <sup>st</sup> SRAM 自测通过, 没有错误. 1 = 1 <sup>st</sup> SRAM 自测失败.

**SYS\_IRCTCTL\_HIRC校准控制寄存器**

寄存器	偏移	读/写	描述	复位值
SYS_IRCTCTL	SYS_BA+0xF0	读/写	HIRC校准控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved					REFCKSEL	Reserved	CESTOPEN
7	6	5	4	3	2	1	0
RETRYCNT		LOOPSEL		Reserved		FREQSEL	

位	描述	
[31:11]	Reserved	保留.
[10]	REFCKSEL	<p>校准参考时钟选择            0 = HIRC 校准参考时钟选 LXT (32.768 kHz).            1 = HIRC校准 参考时钟选 USB SOF (Start-Of-Frame)  <b>注意:</b> 测试模式 HIRC校准参考时钟是 20Khz.</p>
[9]	Reserved	保留.
[8]	CESTOPEN	<p>时钟错误停止校准使能位            0 =若时钟错误, 保持校准继续            1 = 若时钟错误, 停止校准</p>
[7:6]	RETRYCNT	<p>校准更新次数限制            这个值决定了在 HIRC 锁定之前, HIRC计算更新几次            一旦HIRC锁定, 内部校准更新次数计数器会复位            如果校准值更新计数器达到这个限制值且HIRC频率仍然没有锁定, 自动校准操作会禁止且 FREQSEL会清0            00 = 校准重试次数限制在 64 次            01 = 校准重试次数限制在 128 次.            10 = 校准重试次数限制在 256 次            11 = 校准重试次数限制在 512 次</p>
[5:4]	LOOPSEL	<p>配置几个参考时钟做一次校准计算            00 = 4 个参考时钟做一次校准计算.            01 = 8 个参考时钟做一次校准计算            10 = 16 个参考时钟做一次校准计算            11 = 32 个参考时钟做一次校准计算</p>
[3:2]	Reserved	保留.

[1:0]	<b>FREQSEL</b>	校准频率选择 CESTOPEN =1时, 若校准次数已到, 时钟仍未锁定, 这个值将自动回 00 00 = 关闭自动校准 01 = 使能 HIRC 自动校准到 12 MHz. 10 = 保留.. 11 = 保留.
-------	----------------	---

**SYS\_IRCTIEN HIRC校准中断使能寄存器**

寄存器	偏移	读/写	描述	复位值
SYS_IRCTIEN	SYS_BA+0xF4	读/写	HIRC 校准中断使能寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved					CLKEIEN	TFAILIEN	Reserved

位	描述	
[31:3]	Reserved	保留.
[2]	CLKEIEN	<p>时钟错误中断使能位 32768Hz 和 HIRC 一旦偏离到不合理的值， 如果CLKERRIF(SYS_IRCTSTS[2])=1， 此位控制是否允许中断。 0 = 禁止 CLKERRIF(SYS_IRCTSTS[2]) 状态 触发中断 1 = 使能 CLKERRIF(SYS_IRCTSTS[2]) 状态 触发中断</p>
[1]	TFAILIEN	<p>校准失败中断使能位 此位控制一旦校准失败 TFAILIF(SYS_IRCTISTS[1]) =1， 是否允许中断 0 = 禁止 TFAILIF(SYS_IRCTISTS [1]) 触发中断 1 = 使能 TFAILIF(SYS_IRCTISTS [1]) 触发中断</p>
[0]	Reserved	保留.

**SYS\_IRCTISTS HIRC校准中断状态寄存器**

寄存器	偏移	读/写	描述	复位值
SYS_IRCTISTS	SYS_BA+0xF8	读/写	HIRC校准中断状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved					CLKERRIF	TFAILIF	FREQLOCK

位	描述	
[31:3]	Reserved	保留.
[2]	CLKERRIF	<b>时钟错误中断状态</b> 一旦 32768Hz 或 HIRC 偏离到不合理的值，此位将置1 如果CLKEIEN(SYS_IRCTIEN[2])=1使能中断，此位一旦置1将触发中断 0 = 时钟准确 1 = 时钟不准 此位置1校准将停止，此时如果CESTOPEN(SYS_IRCTCTL[8])=1， FREQSEL(SYS_IRCTCTL[1:0])将回到00 <b>注意:</b> 写1清0
[1]	TFAILIF	<b>校准失败中断标志</b> 此位指示校准计算次数达RETRYCNT次以后，HIRC仍不准确。 如果TFAILIEN(SYS_IRCTIEN[1])=1允许中断，一旦此位置1将触发中断。 0 = 计算次数还未达到上限 1 = 计算次数已到但时钟仍未锁定，FREQSEL(SYS_IRCTCTL[1:0])回00，校准停止 <b>注意:</b> 写1清0
[0]	FREQLOCK	<b>HIRC频率锁定状态</b> 此位不触发中断 若使能了校准功能，且已锁定，此位写1清0后，又会自动回到1 0 = HIRC未锁定在12 MHz. 1 = HIRC锁定在12 MHz.

**SYS\_REGLCTL 解锁寄存器**

为防止代码误写，或寄存器数据被干扰，上电后某些寄存器的值是锁定不可写的。需向此寄存器(地址0x4000\_0100)连续写“59h”，“16h”“88h”三个值，才能解锁，任何其它值，或写的顺序不对，或写的过程被打断没有连续写，都会解锁不成功。

读出值，位0若为1表明解锁成功。

上锁，只需向此寄存器写一个任意其它值即可。

寄存器	偏移	读/写	描述	复位值
SYS_REGLCTL	SYS_BA+0x100	读/写	解锁寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
REGLCTL							

位	描述
[31:16]	<b>Reserved</b> 保留.
[7:0]	<b>REGLCTL</b> <b>解锁代码 (只写)</b> 解锁，需依次写入 0x59, 0x16, 0x88，解锁后REGLCTL位置1，写保护寄存器可正常写。
[0]	<b>成功解锁指示位 (只读)</b> 0 = 已上锁，对上锁的寄存器，写入操作将无效。 1 = 解锁成功。 上锁的寄存器有以下几个：  SYS_IPRST0 地址0x4000_0008 SYS_ALTCTL 地址0x4000_0014 SYS_BODCTL 地址0x4000_0018 SYS_PORCTL 地址0x4000_0024 SYS_VREFCTL 地址0x4000_0028 SYS_USBPHY 地址0x4000_002C SYS_SRAM_BISTCTL 地址0x4000_00D0

SYS_PLCTL	地址0x4000_01F8
CLK_PWRCTL	地址0x4000_0200
CLK_APBCLK0	地址0x4000_0208
CLK_CLKSEL0	地址0x4000_0210
CLK_CLKSEL1	地址0x4000_0214
CLK_PLLCTL	地址0x4000_0240
CLK_PMUCTL	地址0x4000_0290
NMIEN	地址0x4000_0300
AHBMCTL	地址0x4000_0400
FMC_FTCTL	地址0x4000_5018
FMC_ICPCMD	地址0x4000_501C
FMC_ISPCTL	地址0x4000_C000
FMC_ISPTRG	地址0x4000_C010
FMC_ISPSTS	地址0x4000_C040
FMC_CYCCTL	地址0x4000_C04C
FMC_KPKEYTRG	地址0x4000_C05C
FMC_KPKEYSTS	地址0x4000_C060
WDT_CTL	地址0x4004_0000
WDT_ALTCTL	地址0x4004_0004
TIMER0_CTL	地址0x4005_0000
TIMER1_CTL	地址0x4005_0100
TIMER2_CTL	地址0x4005_1000
TIMER3_CTL	地址0x4005_1100
TIMER0_PWMCTL	地址0x4005_0040

	TIMER1_PWMCTL	地址0x4005_0140
	TIMER2_PWMCTL	地址0x4005_1040
	TIMER3_PWMCTL	地址0x4005_1140
	TIMER0_PWMDTCTL	地址0x4005_0058
	TIMER1_PWMDTCTL	地址0x4005_0158
	TIMER2_PWMDTCTL	地址0x4005_1058
	TIMER3_PWMDTCTL	地址0x4005_1158
	TIMER0_PWMBRKCTL	地址0x4005_0070
	TIMER1_PWMBRKCTL	地址0x4005_0170
	TIMER2_PWMBRKCTL	地址0x4005_1070
	TIMER3_PWMBRKCTL	地址0x4005_1170
	TIMER0_PWMSWBRK	地址0x4005_007C
	TIMER1_PWMSWBRK	地址0x4005_017C
	TIMER2_PWMSWBRK	地址0x4005_107C
	TIMER3_PWMSWBRK	地址0x4005_117C
	TIMER0_PWMINTEN1	地址0x4005_0084
	TIMER1_PWMINTEN1	地址0x4005_0184
	TIMER2_PWMINTEN1	地址0x4005_1084
	TIMER3_PWMINTEN1	地址0x4005_1184
	TIMER0_PWMINTSTS1	地址0x4005_008C
	TIMER1_PWMINTSTS1	地址0x4005_018C
	TIMER2_PWMINTSTS1	地址0x4005_108C
	TIMER3_PWMINTSTS1	地址0x4005_118C
	EPWM_CTL0	地址0x4005_8000/0x4005_9000
	EPWM_CTL1	地址0x4005_8000/0x4005_9000
	EPWM_DTCTL0_1	地址0x4005_8070/0x4005_9070
	EPWM_DTCTL2_3	地址0x4005_8074/0x4005_9074
	EPWM_DTCTL4_5	地址0x4005_8078/0x4005_9078
	EPWM_BRKCTL0_1	地址0x4005_80C8/0x4005_90C8
	EPWM_BRKCTL2_3	地址0x4005_80CC/0x4005_90CC
	EPWM_BRKCTL4_5	地址0x4005_80D0/0x4005_90D0
	EPWM_SWBRK	地址0x4005_80DC/0x4005_90DC
	EPWM_INTEN1	地址0x4005_80E4/0x4005_90E4
	EPWM_INTSTS1	地址0x4005_80EC/0x4005_90EC

		BPWM_CTL0 地址0x4005_A000/0x4005_B000
		SYST_VAL 地址0xE000_E018

SYS\_PLCTL 电源模式控制

寄存器	偏移	读/写	描述	复位值
SYS_PLCTL	SYS_BA+0x1F8	读/写	电源模式控制	0x0000_0000

31	30	29	28	27	26	25	24
LVSPRD							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved						PVSEL	

位	描述	
[31:24]	<b>LVSPRD</b>	<b>LDO电压调整周期 (写保护位)</b> 内核电压改变时，配置每改变一步的停留时间。 $CVSPRD = 0$ , 停留 1us. 其它值时，每步停留时间 = $(CVSPRD + 1) * 1\mu s$ .
[23:22]	<b>Reserved</b>	保留.
[21:16]	<b>LVSSTEP</b>	<b>LDO电压增加步数 (写保护位)</b> 该域用于设置LDO电压改变时的电压步进值，单位是10mV $LDO\ 电压增加 = (CVSSTEP + 1) * 10mV$ .
[15:2]	<b>Reserved</b>	保留.
[1:0]	<b>PVSEL</b>	<b>内核电压选择 (写保护位)</b> 00 = 电源模式设定PL0. 01 = 电源模式设定PL1. 其他 = 保留.

**SYS\_PLSTS 电源模式状态**

寄存器	偏移	读/写	描述				复位值
SYS_PLSTS	SYS_BA+0x1FC	读/写	电源模式状态				0x0000_010X

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved						PLSTATUS	
7	6	5	4	3	2	1	0
Reserved							
PLCBUSY							

位	描述	
[31:10]	<b>Reserved</b>	保留.
[9:8]	<b>PLSTATUS</b>	<p>电压状态 (只读)            该域反映当前的电压            00 = 电源模式设定PL0.            01 = 电源模式设定PL1.            其他 = 保留</p>
[7:1]	<b>Reserved</b>	保留.
[0]	<b>PLCBUSY</b>	<p>电压改变指示 (只读)            0 = 电压改变完成.            1 = 电压正在改变</p>

**SYS\_AHBMCTL AHB 总线矩阵优先级控制**

寄存器	偏移	读/写	描述	复位值
SYS_AHBMCTL	SYS_BA+0x400	读/写	AHB总线矩阵优先级控制	0x0000_0001

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved							INTACTEN

位	描述	
[31:1]	Reserved	保留.
[0]	INTACTEN	<p>Cortex M4 内核最高AHB总线优先级使能位 (写保护位)            0 = 轮换模式.            1 = 中断发生时, Cortex®-M4 CPU 有最高优先级.</p> <p><b>注意:</b> 此位的写是受保护的, 请参考寄存器 SYS_REGLCTL</p>

### 6.2.12 系统定时器(SysTick)

Cortex®-M4 内建一个24位下计数定时器，可用于RTOS的节拍时钟，也可当做一个通用定时器用。

复位后定时器的当前值SYST\_VAL随机。定时器使能后，当前计数寄存器 SYST\_VAL就开始下计数，计到0后重新装入周期值SYST\_LOAD，并且置位 COUNTFLAG ——此位读清0.

详细请参考 “ARM® Cortex™-M4 Technical Reference Manual” 和 “ARM® v6-M Architecture Reference Manual”.

#### 6.2.12.1 系统定时器控制寄存器映射

R: 只读, W: 只写, R/W: 可读写

寄存器	偏移	R/W	描述	复位值
<b>SYST 基地址:</b> <b>SCS_BA = 0xE000_E000</b>				
<b>SYST_CTRL</b>	SCS_BA+0x10	R/W	系统定时器控制和状态寄存器	0x0000_0000
<b>SYST_LOAD</b>	SCS_BA+0x14	R/W	系统定时器周期寄存器	0xFFFF_FFFF
<b>SYST_VAL</b>	SCS_BA+0x18	R/W	系统定时器当前值寄存器	0xFFFF_FFFF

## 6.2.12.2 系统定时器寄存器描述

**SYST\_CTRL SysTick控制和状态寄存器**

寄存器	偏移	读/写	描述	复位值
<b>SYST_CTRL</b>	SCS_BA+0x10	读/写	SysTick 控制和状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved					CLKSRC	TICKINT	ENABLE

位	描述	
[31:17]	<b>Reserved</b>	保留.
[16]	<b>COUNTFLAG</b>	<b>System Tick计数器标志</b> 当前计数器值1变0时，此位置1 读清0，写当前值寄存器 SYST_VAL 此位也会清0
[15:3]	<b>Reserved</b>	保留.
[2]	<b>CLKSRC</b>	<b>计数时钟源选择</b> 0 = 选外部参考时钟(可选项). 1 = 对内核时钟计数
[1]	<b>TICKINT</b>	<b>System Tick中断使能位</b> 0 = 计数到0 不申请中断，软件可查寻 COUNTFLAG得知是否计数跨0 1 = 计数到0申请中断，软件写0到当前值寄存器不会申请中断
[0]	<b>ENABLE</b>	<b>计数使能位</b> 0 = 停止计数 1 = 开始计数

**SYST LOAD 周期寄存器**

寄存器	偏移	读/写	描述	复位值
SYST_LOAD	SCS_BA+0x14	读/写	周期寄存器	0xFFFF_FFFF

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
RELOAD							
15	14	13	12	11	10	9	8
RELOAD							
7	6	5	4	3	2	1	0
RELOAD							

位	描述	
[31:24]	Reserved	保留.
[23:0]	RELOAD	周期值 每次 SYST_VAL 计数到0后，重装入此值，再下计数。

**SYST\_VAL** 当前值寄存器

寄存器	偏移	读/写	描述	复位值
<b>SYST_VAL</b>	SCS_BA+0x18	读/写	当前值寄存器	0xXXXX_XXXX

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
CURRENT							
15	14	13	12	11	10	9	8
CURRENT							
7	6	5	4	3	2	1	0
CURRENT							

位	描述	
[31:24]	<b>Reserved</b>	保留.
[23:0]	<b>CURRENT</b>	<b>当前值</b> 读: 计数的当前值。 写: 写任何值, 都清0当前值寄存器.

### 6.2.13 NVIC 可嵌套中断向量

NVIC 与内核紧密结合，以便快速处理中断。特权模式可读写 NVIC 的所有寄存器。用户模式下可软件触发一个中断。NVIC 寄存器放在 SCS 空间，可按字节，半字和字读写。NVIC 寄存器与调试寄存器都按小端格式读写。

NVIC 支持：

- 支持 1-240 个中断。
- 优先级 0-16，数值越小优先级越高。
- 支持电平和边沿触发中断
- 优先级可动态调整。
- 优先级可分组。同组内不抢先。
- 支持非屏蔽中断 NMI
- 支持休眠模式

发生中断时，处理器自动入栈中断入口的状态，中断返回时处理器自动出栈该状态，不需要顶层的指令处理。

#### 6.2.13.1 异常模式和中断向量表

表 6.2-8 列举了 M480 中断向量。软件可配置 16 个优先级：0x00 最高，0xF0 最低。L 缺省值为 0x00。“Reset”，“NMI” 和 “Hard Fault”的优先级是不可配置的，比 0 还优先。

发生中断时，处理器从向量地址得到中断代码的首地址。复位后向量表在 0x00000000。特权模式下可改写 VTOR 改变向量表的首地址：范围 0x00000080~0x3FFFFF80

向量表包含了复位后栈指针的初值和所有中断代码的首地址，其排列如下表：

异常类型	向量号	向量地址	优先级
复位 Reset	1	0x00000004	-3
非屏蔽中断 NMI	2	0x00000008	-2
硬件错误 Hard Fault	3	0x0000000C	-1
内存管理错误 Memory Manager Fault	4	0x00000010	可配置
总线错误 Bus Fault	5	0x00000014	可配置
用法错误 Usage Fault	6	0x00000018	可配置
保留	7 ~ 10		保留
服务调用 SVC	11	0x0000002C	可配置
调试监测 Debug Monitor	12	0x00000030	可配置
保留	13		保留
挂起服务 PendSV	14	0x00000038	可配置
系统定时器 SysTick	15	0x0000003C	可配置

中断 (IRQ0 ~ IRQ)	16 ~ 111	0x00000000 + (Vector Number)*4	可配置
-----------------	----------	-----------------------------------	-----

表 6.2-8 异常类型

向量序号 (中断寄存器中的位)	中断序号 (中断寄存器中的位)	中断名称	描述
0 ~ 15	-	-	系统异常
16	0	BODOUT	BOD 欠压中断
17	1	IRC_INT	IRC 校准中断
18	2	PWRWU_INT	时钟控制和唤醒状态中断
19	3	SRAM_PERR	SRAM 校验错中断
20	4	CLKFAIL	时钟失效中断
21	5	Reserved	保留
22	6	RTC_INT	实时时钟中断
23	7	TAMPER_INT	清除中断
24	8	WDT_INT	看门狗中断
25	9	WWDT_INT	窗看门狗中断
26	10	EINT0	外中断 PA.0, PD.2 或 PE.4
27	11	EINT1	外中断 PB.0, PD.3 或 PE.5
28	12	EINT2	外中断 PC.0
29	13	EINT3	外中断 PD.0
30	14	EINT4	外中断 PE.0
31	15	EINT5	外中断 PF.0
32	16	GPA_INT	外中断 PA[15:0]
33	17	GPB_INT	外中断 PB[15:0]
34	18	GPC_INT	外中断 PC[15:0]
35	19	GPD_INT	外中断 PD[15:0]
36	20	GPE_INT	外中断 PE[15:0]
37	21	GPF_INT	外中断 PF[15:0]
38	22	QSPI0_INT	QSPI0 中断
39	23	SPI0_INT	SPI0 中断
40	24	BRAKE0_INT	PWM0 急停中断
41	25	PWM0_P0_INT	PWM0_P0 中断

42	26	PWM0_P1_INT	PWM0_P1 中斷
43	27	PWM0_P2_INT	PWM0_P2 中斷
44	28	BRAKE1_INT	PWM1 急停中斷
45	29	PWM1_P0_INT	PWM1_P0 中斷
46	30	PWM1_P1_INT	PWM1_P1 中斷
47	31	PWM1_P2_INT	PWM1_P2 中斷
48	32	TMR0_INT	Timer 0 中斷
49	33	TMR1_INT	Timer 1 中斷
50	34	TMR2_INT	Timer 2 中斷
51	35	TMR3_INT	Timer 3 中斷
52	36	UART0_INT	UART0 中斷
53	37	UART1_INT	UART1 中斷
54	38	I2C0_INT	I2C0 中斷
55	39	I2C1_INT	I2C1 中斷
56	40	PDMA_INT	PDMA 中斷
57	41	DAC_INT	DAC 中斷
58	42	EADC0_INT	EADC 中斷0
59	43	EADC1_INT	EADC 中斷1
60	44	ACMP01_INT	ACMP01 中斷
61	45	Reserved	保留
62	46	EADC2_INT	EADC 中斷2
63	47	EADC3_INT	EADC 中斷3
64	48	UART2_INT	UART2 中斷
65	49	UART3_INT	UART3 中斷
66	50	Reserved	保留
67	51	SPI1_INT	SPI1 中斷
68	52	SPI2_INT	SPI2 中斷
69	53	USBD_INT	USB 设备中断
70	54	USBH_INT	USB 主机 中斷
71	55	USBOTG_INT	USB OTG 中斷
72	56	CAN0_INT	CAN0 中斷
73	57	CAN1_INT	CAN1 中斷
74	58	SC0_INT	智能卡接口0中断

75	59	SC1_INT	智能卡接口1中断
76	60	SC2_INT	智能卡接口2中断
77	61	SC3_INT	智能卡接口3中断
78	62	SPI3_INT	SPI3 中断
80	64	SDHOST0_INT	SD卡接口 中断
81	65	HSUSBD_INT	HSUSBD 中断
82	66	EMAC_TX	以太网 MAC 发送中断
83	67	EMAC_RX	以太网 MAC 接收中断
84	68	I2S0_INT	I2S0 中断
85	69	Reserved	保留
86	70	OPA0_INT	运放0 中断
87	71	CRYPTO	加密模块中断
88	72	GPG_INT	外中断 PG[15:0]
89	73	EINT6	外中断 PG.0
90	74	UART4_INT	UART4 中断
91	75	UART5_INT	UART5 中断
92	76	USCI0_INT	USCI0 中断
93	77	USCI1_INT	USCI1 中断
94	78	BPWM0_INT	BPWM0 中断
95	79	BPWM1_INT	BPWM1 中断
96	80	SPIM_INT	主控SPI 中断
97	81	Reserved	图像采集中断
98	82	I2C2_INT	I2C2 中断
99	83	Reserved	保留
100	84	QEI0_INT	QEI0 中断
101	85	QEI1_INT	QEI1 中断
102	86	ECAP0_INT	ECAP0 中断
103	87	ECAP1_INT	ECAP1 中断
105	88	GPH_INT	外中断 PH[?:0]
105	89	EINT7	外中断PH.?
106	90	SDHOST1_INT	SD接口1 中断
107	91	Reserved	保留
108	92	HSUSBH_INT	HSUSBH 中断

109	93	HSOTG_INT	HSOTG 中断

表 6.2-9 中断向量表

### 6.2.13.2 操作描述

对ISER对应位写1可以使能NVIC中断，对ICER对应位写1可以禁止NVIC中断。读二者任意一个都可得到中断的使能状态。若某个中断被禁止，中断挂起位被置1时，中断不会被响应。复位或中断返回会清除中断挂起位。

**NVIC** 中断挂起位——通过对ISPR写1置1，通过对ICPR写1清0，读二者任一个都可得挂起状态——被清0时，不会影响正在响应的中断状态。

**NVIC**中断的优先级，可以通过一个32位寄存器的8位域来控制(每个字控制四个中断的优先级)

## 6.2.13.3 NVIC 控制寄存器

**R:** 只读, **W:** 只写, **R/W:** 可读写

寄存器	偏移	R/W	描述	复位值
<b>NVIC 基址:</b>				
<b>NVIC_BA = 0xE000_E100</b>				
<b>NVIC_ISER0</b>	NVIC_BA+0x00	R/W	IRQ0 ~ IRQ71 设置使能控制寄存器	0x0000_0000
<b>NVIC_ISER1</b>	NVIC_BA+0x04	R/W	IRQ0 ~ IRQ71设置使能控制寄存器	0x0000_0000
<b>NVIC_ISER2</b>	NVIC_BA+0x08	R/W	IRQ0 ~ IRQ71设置使能控制寄存器	0x0000_0000
<b>NVIC_ICER0</b>	NVIC_BA+0x80	R/W	IRQ0 ~ IRQ71 清除使能控制寄存	0x0000_0000
<b>NVIC_ICER1</b>	NVIC_BA+0x84	R/W	IRQ0 ~ IRQ71清除使能控制寄存	0x0000_0000
<b>NVIC_ICER2</b>	NVIC_BA+0x88	R/W	IRQ0 ~ IRQ71清除使能控制寄存	0x0000_0000
<b>NVIC_ISPR0</b>	NVIC_BA+0x100	R/W	IRQ0 ~ IRQ71 设置挂起控制寄存器	0x0000_0000
<b>NVIC_ISPR1</b>	NVIC_BA+0x104	R/W	IRQ0 ~ IRQ71设置挂起控制寄存器	0x0000_0000
<b>NVIC_ISPR2</b>	NVIC_BA+0x108	R/W	IRQ0 ~ IRQ71设置挂起控制寄存器	0x0000_0000
<b>NVIC_ICPR0</b>	NVIC_BA+0x180	R/W	IRQ0 ~ IRQ71 清除挂起控制寄存器	0x0000_0000
<b>NVIC_ICPR1</b>	NVIC_BA+0x184	R/W	IRQ0 ~ IRQ71清除挂起控制寄存器	0x0000_0000
<b>NVIC_ICPR2</b>	NVIC_BA+0x188	R/W	IRQ0 ~ IRQ71清除挂起控制寄存器	0x0000_0000
<b>NVIC_IABR0</b>	NVIC_BA+0x200	R/W	IRQ0 ~ IRQ71 中断响应状态位寄存器	0x0000_0000
<b>NVIC_IABR1</b>	NVIC_BA+0x204	R/W	IRQ0 ~ IRQ71中断响应状态位寄存器	0x0000_0000
<b>NVIC_IABR2</b>	NVIC_BA+0x208	R/W	IRQ0 ~ IRQ71中断响应状态位寄存器	0x0000_0000
<b>NVIC_IPRn n=0,1..17</b>	NVIC_BA+0x300 +0x4*n	R/W	IRQ0 ~ IRQ71 中断优先级寄存器	0x0000_0000
<b>STIR</b>	NVIC_BA+0xE00	R/W	中断软件触发寄存器	0x0000_0000

**NVIC ISER0 IRQ0 ~ IRQ71 设置使能控制器**

寄存器	偏移地址	R/W	描述	复位值
NVIC_ISER0	0xE000E100	R/W	IRQ0 ~ IRQ71 设置使能控制寄存器	0x0000_0000

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

位	描述	
[31:0]	SETENA	中断设置使能位 写操作: 0 = 无效. 1 = 中断使能. 读操作: 0 = 中断禁止. 1 = 中断使能.

NVIC ISER1 IRQ0 ~ IRQ71 设置使能控制器

寄存器	偏移地址	R/W	描述	复位值
NVIC_ISER0	0xE000E104	R/W	IRQ0 ~ IRQ71 设置使能控制寄存器	0x0000_0000

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

位	描述	
[31:0]	SETENA	中断设置使能位 写操作: 0 = 无效. 1 = 中断使能. 读操作: 0 = 中断禁止. 1 = 中断使能.

**NVIC ISER2 IRQ0 ~ IRQ71 设置使能控制器**

寄存器	偏移地址	R/W	描述	复位值
NVIC_ISER2	0xE000E108	R/W	IRQ0 ~ IRQ71设置使能控制寄存器	0x0000_0000

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

位	描述	
[31:0]	SETENA	中断设置使能位 写操作: 0 = 无效. 1 = 中断使能. 读操作: 0 = 中断禁止. 1 = 中断使能.

**NVIC ICER0 IRQ0 ~ IRQ71 清除使能控制寄存器**

寄存器	偏移地址	R/W	描述	复位值
NVIC_ICER0	0xE000E180	R/W	IRQ0 ~ IRQ71 清除使能控制寄存器	0x0000_0000

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

位	描述	
[31:0]	CALENA	中断清除使能位 写: 0 = 无效. 1 = 禁止中断. 读: 0 = 中断被禁止. 1 = 中断已使能.

**NVIC ICER1 IRQ0 ~ IRQ71 清除使能控制寄存器**

寄存器	偏移地址	R/W	描述	复位值
NVIC_ICER1	0xE000E184	R/W	IRQ0 ~ IRQ71清除使能控制寄存器	0x0000_0000

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

位	描述	
[31:0]	CALENA	中断清除使能位 写: 0 = 无效. 1 = 禁止中断. 读: 0 = 中断被禁止. 1 = 中断已使能.

**NVIC ICER2 IRQ0 ~ IRQ71 清除使能控制寄存器**

寄存器	偏移地址	R/W	描述	复位值
NVIC_ICER2	0xE000E188	R/W	IRQ0 ~ IRQ71清除使能控制寄存器	0x0000_0000

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

位	描述	
[31:0]	CALENA	中断清除使能位 写: 0 = 无效. 1 = 禁止中断. 读: 0 = 中断被禁止. 1 = 中断已使能.

**NVIC ISPR0 IRQ0 ~ IRQ71 设置挂起控制寄存器**

寄存器	偏移地址	R/W	描述	复位值
NVIC_ISPR0	0xE000E200	R/W	IRQ0 ~ IRQ71 设置挂起控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
<b>SETPEND</b>							
23	22	21	20	19	18	17	16
<b>SETPEND</b>							
15	14	13	12	11	10	9	8
<b>SETPEND</b>							
7	6	5	4	3	2	1	0
<b>SETPEND</b>							

位	描述
[31:0]	<b>SETPEND</b> 中断挂起置1位 写: 0 = 无效. 1 = 挂起中断. 读: 0 = 中断未被挂起. 1 = 中断已挂起

**NVIC ISPR1 IRQ0 ~ IRQ71 设置挂起控制寄存器**

寄存器	偏移地址	R/W	描述	复位值
NVIC_ISPR1	0xE000E204	R/W	IRQ0 ~ IRQ71设置挂起控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
<b>SETPEND</b>							
23	22	21	20	19	18	17	16
<b>SETPEND</b>							
15	14	13	12	11	10	9	8
<b>SETPEND</b>							
7	6	5	4	3	2	1	0
<b>SETPEND</b>							

位	描述	
[31:0]	<b>SETPEND</b>	中断挂起置1位 写: 0 = 无效. 1 = 挂起中断. 读: 0 = 中断未被挂起. 1 = 中断已挂起

**NVIC ISPR2 IRQ0 ~ IRQ71 设置挂起控制寄存器**

寄存器	偏移地址	R/W	描述	复位值
NVIC_ISPR2	0xE000E208	R/W	IRQ0 ~ IRQ71设置挂起控制寄存器	0x0000_0000

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

位	描述	
[31:0]	SETPEND	中断挂起置1位 写: 0 = 无效. 1 = 挂起中断. 读: 0 = 中断未被挂起. 1 = 中断已挂起

NVIC\_ICPR0 IRQ0 ~ IRQ71 清除挂起控制寄存器

寄存器	偏移地址	R/W	描述	复位值
NVIC_ICPR0	0xE000E280	R/W	IRQ0 ~ IRQ71 清除挂起控制寄存器	0x0000_0000

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

位	描述
[31:0]	<p><b>CALPEND</b></p> <p>中断挂起清除位</p> <p>写:</p> <p>0 = 无效.</p> <p>1 = 清除 中断挂起.状态</p> <p>读:</p> <p>0 = 中断没挂起.</p> <p>1 = 中断已挂起.</p>

**NVIC\_ICPR1 IRQ0 ~ IRQ71 清除挂起控制寄存器**

寄存器	偏移地址	R/W	描述	复位值
NVIC_ICPR1	0xE000E284	R/W	IRQ0 ~ IRQ71 清除挂起控制寄存器	0x0000_0000

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

位	描述	
[31:0]	CALPEND	中断挂起清除位 写: 0 = 无效. 1 = 清除 中断挂起.状态 读: 0 = 中断没挂起. 1 = 中断已挂起.

**NVIC\_ICPR2 IRQ0 ~ IRQ71 清除挂起控制寄存器**

寄存器	偏移地址	R/W	描述	复位值
NVIC_ICPR2	0xE000E288	R/W	IRQ0 ~ IRQ71清除挂起控制寄存器	0x0000_0000

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

位	描述	
[31:0]	CALPEND	中断挂起清除位 写: 0 = 无效. 1 = 清除 中断挂起.状态 读: 0 = 中断没挂起. 1 = 中断已挂起.

**NVIC\_IABR0 中断活动状态寄存器 IRQ0 ~ IRQ71**

寄存器	偏移地址	R/W	描述	复位值
NVIC_IABR0	0xE000E300	R/W	中断活动状态寄存器 IRQ0 ~ IRQ71	0x0000_0000

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

位	描述	
[31:0]	ACTIVE	中断活动标志 0 = 中断未激活 1 = 中断已激活.

**NVIC\_IABR1 中断活动状态寄存器 IRQ0 ~ IRQ71**

寄存器	偏移地址	R/W	描述	复位值
NVIC_IABR1	0xE000E304	R/W	中断活动状态寄存器 IRQ0 ~ IRQ71	0x0000_0000

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

位	描述		
[31:0]	ACTIVE	中断活动标志 0 = 中断未激活 1 = 中断已激活	

**NVIC\_IABR2 中断活动状态寄存器IRQ0 ~ IRQ71**

寄存器	偏移地址	R/W	描述	复位值
NVIC_IABR2	0xE000E308	R/W	中断活动状态寄存器IRQ0 ~ IRQ71	0x0000_0000

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

位	描述	
[31:0]	ACTIVE	中断活动标志 0 = 中断未激活 1 = 中断已激活.

**NVIC\_IPRn IRQ0 ~ IRQ71 中断优先级寄存器**

寄存器	偏移地址	R/W	描述	复位值
NVIC_IPRn n=0,1..17	0xE000E400 +0x4*n	R/W	IRQ0 ~ IRQ71 中断优先级寄存器	0x0000_0000

31	30	29	28	27	26	25	24
PRI_4n_3				Reserved			
23	22	21	20	19	18	17	16
PRI_4n_2				Reserved			
15	14	13	12	11	10	9	8
PRI_4n_1				Reserved			
7	6	5	4	3	2	1	0
PRI_4n_0				Reserved			

位	描述	
[31:28]	PRI_4n_3	IRQ_4n+3的优先级 “0”最高，“15”最低
[27:24]	Reserved	保留
[23:20]	PRI_4n_2	IRQ_4n+2的优先级 “0”最高，“15”最低
[19:16]	Reserved	保留
[15:12]	PRI_4n_1	IRQ_4n+1的优先级 “0”最高，“15”最低
[11:8]	Reserved	保留
[7:4]	PRI_4n_0	IRQ_4n+0 的优先级 “0”最高，“15”最低
[3:0]	Reserved	保留.

**STIR 中断软件触发寄存器**

寄存器	偏移地址	R/W	描述	复位值
STIR	0xE000EF00	R/W	中断软件触发寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
INTID							

位	描述	
[31:9]	Reserved	保留.
[8:0]	INTID	中断 ID 写 STIR 软件触发一个中断，范围0~0x63，如写0x03，就是触发IRQ3

## 6.2.13.4 NMI 控制寄存器

**R:** 只读, **W:** 只写, **R/W:** 可读写

寄存器	偏移地址	R/W	描述	复位值
<b>NMI 基地址:</b> <b>NMI_BA = 0x4000_0300</b>				
<b>NMIEN</b>	NMI_BA+0x00	R/W	NMI 中断源使能寄存器	0x0000_0000
<b>NMISTS</b>	NMI_BA+0x04	R	NMI 中断源状态寄存器	0x0000_0000

**NMIEN 不可屏蔽中断源使能寄存器**

寄存器	偏移地址	R/W	描述				复位值
NMIEN	NMI_BA+0x00	R/W	NMI 不可屏蔽中断源使能寄存器				0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
UART1_INT	UART0_INT	EINT5	EINT4	EINT3	EINT2	EINT1	EINT0
7	6	5	4	3	2	1	0
TAMPER_INT	RTC_INT	Reserved	CLKFAIL	SRAM_PERR	PWRWU_INT	IRC_INT	BODOUT

位	描述
[31:16]	<b>Reserved</b> 保留.
[15]	<b>UART1_INT</b> <b>UART1 非屏蔽使能位 (写保护位)</b> 0 = 禁止 UART1 做为非屏蔽中断源. 1 = 使能 UART1 做为非屏蔽中断源 <b>注意:</b> 此位的写是受保护的, 请参考寄存器 SYS_REGLCTL
[14]	<b>UART0_INT</b> <b>UART0 非屏蔽使能位 (写保护位)</b> 0 = 禁止 UART0 做为非屏蔽中断源 1 = 使能 UART0 做为非屏蔽中断源. <b>注意:</b> 此位的写是受保护的, 请参考寄存器 SYS_REGLCTL
[13]	<b>EINT5</b> <b>外中断 PF.0 非屏蔽使能位 (写保护位)</b> 0 = 禁止外中断 PF.0 做为非屏蔽中断源. 1 = 使能外中断 PF.0 做为非屏蔽中断源 <b>注意:</b> 此位的写是受保护的, 请参考寄存器 SYS_REGLCTL.
[12]	<b>EINT4</b> <b>外中断 PE.0 非屏蔽使能位(写保护位)</b> 0 = 禁止外中断 PE.0 做为非屏蔽中断源. 1 = 使能外中断 PE.0 做为非屏蔽中断源. <b>注意:</b> 此位的写是受保护的, 请参考寄存器 SYS_REGLCTL
[11]	<b>EINT3</b> <b>外中断 PD.0 非屏蔽使能位 (写保护位)</b> 0 = 禁止外中断 PD.0 做为非屏蔽中断源. 1 = 使能外中断 PD.0 做为非屏蔽中断源. <b>注意:</b> 此位的写是受保护的, 请参考寄存器 SYS_REGLCTL
[10]	<b>EINT2</b> <b>外中断 PC.0 非屏蔽使能位 (写保护位)</b> 0 = 禁止外中断 PC.0 做为非屏蔽中断源. 1 = 使能外中断 PC.0 做为非屏蔽中断源.

		<b>注意:</b> 此位的写是受保护的, 请参考寄存器 SYS_REGLCTL.
[9]	EINT1	<p><b>外中断 PB.0, PD.3 或 PE.5 非屏蔽使能位 (写保护位)</b></p> <p>0 = 禁止外中断 PB.0, PD.3 或 PE.5 做为非屏蔽中断源. 1 = 使能外中断 PB.0, PD.3 或 PE.5 做为非屏蔽中断源.</p> <p><b>注意:</b> 此位的写是受保护的, 请参考寄存器 SYS_REGLCTL</p>
[8]	EINT0	<p><b>外中断 PA.0, PD.2 或 PE.4 非屏蔽使能位 (写保护位)</b></p> <p>0 = 禁止外中断 PA.0, PD.2 或 PE.4 做为非屏蔽中断源. 1 = 使能外中断 PA.0, PD.2 或 PE.4 做为非屏蔽中断源.</p> <p><b>注意:</b> 此位的写是受保护的, 请参考寄存器 SYS_REGLCTL</p>
[7]	TAMPER_INT	<p><b>清除引脚非屏蔽使能位 (写保护位)</b></p> <p>0 = 禁止 清除引脚 做为非屏蔽中断源 1 = 使能 清除引脚 做为非屏蔽中断源.</p> <p><b>注意:</b> 此位的写是受保护的, 请参考寄存器 SYS_REGLCTL</p>
[6]	RTC_INT	<p><b>RTC非屏蔽使能位(写保护位)</b></p> <p>0 = 禁止 RTC 做为非屏蔽中断源. 1 = 使能 RTC 做为非屏蔽中断源</p> <p><b>注意:</b> 此位的写是受保护的, 请参考寄存器 SYS_REGLCTL.</p>
[5]	Reserved	保留
[4]	CLKFAIL	<p><b>时钟失效非屏蔽使能位 (写保护位)</b></p> <p>0 = 禁止 时钟失效 做为非屏蔽中断源 1 = 使能 时钟失效 做为非屏蔽中断源</p> <p><b>注意:</b> 此位的写是受保护的, 请参考寄存器 SYS_REGLCTL</p>
[3]	SRAM_PERR	<p><b>SRAM奇偶校验错非屏蔽使能位 (写保护位)</b></p> <p>0 = 禁止 SRAM奇偶校验错 做为非屏蔽中断源 1 = 使能 SRAM 奇偶校验错 做为非屏蔽中断源.</p> <p><b>注意:</b> 此位的写是受保护的, 请参考寄存器 SYS_REGLCTL.</p>
[2]	PWRWU_INT	<p><b>掉电唤醒非屏蔽使能位 (写保护位)</b></p> <p>0 = 禁止 掉电唤醒 做为非屏蔽中断源. 1 = 使能 掉电唤醒 做为非屏蔽中断源.</p> <p><b>注意:</b> 此位的写是受保护的, 请参考寄存器 SYS_REGLCTL</p>
[1]	IRC_INT	<p><b>IRC 校准非屏蔽使能位 (写保护位)</b></p> <p>0 = 禁止 IRC 校准中断做为非屏蔽中断源. 1 = 使能 IRC 校准中断做为非屏蔽中断源.</p> <p><b>注意:</b> 此位的写是受保护的, 请参考寄存器 SYS_REGLCTL</p>
[0]	BODOUT	<p><b>BOD 欠压非屏蔽使能位 (写保护位)</b></p> <p>0 = 禁止 BOD 做为非屏蔽中断源. 1 = 使能 BOD 做为非屏蔽中断源.</p> <p><b>注意:</b> 此位的写是受保护的, 请参考寄存器 SYS_REGLCTL.</p>

**NMISTS 非屏蔽中断源状态寄存器**

寄存器	偏移地址	R/W	描述	复位值
NMISTS	NMI_BA+0x04	R	NMI 非屏蔽中断源状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
UART1_INT	UART0_INT	EINT5	EINT4	EINT3	EINT2	EINT1	EINT0
7	6	5	4	3	2	1	0
TAMPER_INT	RTC_INT	Reserved	CLKFAIL	SRAM_PERR	PWRWU_INT	IRC_INT	BODOUT

位	描述	
[31:16]	Reserved	保留
[15]	UART1_INT	<b>UART1 中断标志位 (只读)</b> 0 = UART1 未发生中断 1 = UART1 发生中断
[14]	UART0_INT	<b>UART0 中断标志位 (只读)</b> 0 = UART1 未发生中断 1 = UART1 发生中断.
[13]	EINT5	<b>外中断 PF.0 中断标志位 (只读)</b> 0 = 外中断PF.0 未发生 1 = 发生外中断PF.0 中断
[12]	EINT4	<b>外中断 PE.0中断标志位 (只读)</b> 0 = 外中断 PE.0 未发生. 1 =发生外中断PE.0 中断.
[11]	EINT3	<b>外中断 PD.0 中断标志位 (只读)</b> 0 =外中断 PD.0 未发生 . 1 =发生外中断PD.0 中断.
[10]	EINT2	<b>外中断 PC.0 中断标志位 (只读)</b> 0 =外中断 PC.0 未发生. 1 =发生外中断PC.0 中断.
[9]	EINT1	<b>外中断 PB.0, PD.3 或 PE.5 中断标志位 (只读)</b> 0 =外中断PB.0, PD.3 或 PE.5 未发生. 1 =发生外中断 PB.0, PD.3 或 PE.5 中断
[8]	EINT0	<b>外中断 PA.0, PD.2 或 PE.4 中断标志位 (只读)</b>

		0 = 外中断PA.0, PD.2 或 PE.4 未发生. 1 = 发生外中断 PA.0, PD.2 或 PE.4 中断
[7]	<b>TAMPER_INT</b>	<b>清除引脚中断标志位 (只读)</b> 0 = 未发生清除引脚中断 1 = 发生了清除引脚中断.
[6]	<b>RTC_INT</b>	<b>RTC中断标志位 (只读)</b> 0 = 未发生RTC 中断. 1 = 发生了RTC中断.
[5]	<b>Reserved</b>	保留
[4]	<b>CLKFAIL</b>	<b>时钟失效中断标志位 (只读)</b> 0 = 未发生时钟失效中断 1 = 发生了时钟失效中断.
[3]	<b>SRAM_PERR</b>	<b>SRAM 奇偶核验中断标志位 (只读)</b> 0 = 未发生SRAM奇偶核验中断. 1 = 发生了SRAM 奇偶核验中断.
[2]	<b>PWRWU_INT</b>	<b>掉电唤醒中断标志位 (只读)</b> 0 = 未发生掉电唤醒中断 1 = 发生了掉电唤醒中断
[1]	<b>IRC_INT</b>	<b>IRC 校准中断标志位 (只读)</b> 0 = 未发生HIRC 校准中断. 1 = 发生了HIRC 校准中断.
[0]	<b>BODOUT</b>	<b>BOD 欠压中断标志位 (只读)</b> 0 = 未发生 BOD 欠压中断 1 = 发生了 BOD 欠压中断

## 6.2.13.5 AHB总线矩阵优先级控制寄存器

**R:** 只读, **W:** 只写, **R/W:** 可读写

寄存器	偏移地址	R/W	描述	复位值
<b>AHB 基地址:</b> <b>AHB_BA = 0x4000_0400</b>				
<b>AHBMCTL</b> 0x40000400    R/W    AHB 总线矩阵优先级控制寄存器    0x0000_0001				

**AHBMCTL 总线矩阵优先级控制寄存器**

寄存器	偏移地址	R/W	描述	复位值
AHBMCTL	0x40000400	R/W	AHB 总线矩阵优先级控制寄存器	0x0000_0001

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved							INTACTEN

位	描述	
[31:1]	Reserved	保留.
[0]	INTACTEN	<p>M4 内核最高优先级使能位 (写保护位)            0 = 最高优先级轮换方式            1 = 中断发生时 M4 内核有最高优先级使用总线.  <b>注意:</b> 此位的写是受保护的, 请参考寄存器 SYS_REGLCTL</p>

### 6.2.14 系统控制寄存器

Cortex®-M4 状态和操作模式，包括 CPUID, 内核中断优先级和电源控制，由系统控制寄存器管理。

详细请参考 “ARM® Cortex™-M4 Technical Reference Manual” 和 “ARM® v6-M Architecture Reference Manual”

**R:** 只读, **W:** 只写, **R/W:** 可读写

寄存器	偏移地址	R/W	描述	复位值
<b>SCR 基址:</b> <b>SCS_BA = 0xE000_E000</b>				
<b>ICSR</b>	SCS_BA+0xD04	R/W	中断控制和状态寄存器	0x0000_0000
<b>AIRCR</b>	SCS_BA+0xD0C	R/W	应用中断和复位控制寄存器	0xFA05_0000
<b>SCR</b>	SCS_BA+0xD10	R/W	系统控制寄存器	0x0000_0000
<b>SHPR1</b>	SCS_BA+0xD18	R/W	系统中断优先级寄存器1	0x0000_0000
<b>SHPR2</b>	SCS_BA+0xD1C	R/W	系统中断优先级寄存器2	0x0000_0000
<b>SHPR3</b>	SCS_BA+0xD20	R/W	系统中断优先级寄存器3	0x0000_0000

## ICSR 中断控制状态寄存器

寄存器	偏移地址	R/W	描述					复位值
ICSR	SCS_BA+0xD04	R/W	中断控制和状态寄存器					0x0000_0000

31	30	29	28	27	26	25	24
NMIPENDSET	Reserved		PENDSVSET	PENDSVRTC_CAL	PENDSTSET	PENDSTRTC_CAL	Reserved
23	22	21	20	19	18	17	16
ISRPREEMPT	ISRPENDIN_G	Reserved				VECTPENDING	
15	14	13	12	11	10	9	8
VECTPENDING				RETTOBASE	Reserved		
7	6	5	4	3	2	1	0
Reserved	VECTACTIVE						

位	描述
[31]	<b>NMIPENDSET</b> <b>NMI 中断挂起位</b> 写: 0 = 无效. 1 = 申请 NMI 中断. 读: 0 = 无 NMI 中断. 1 = NMI 中断等待执行. <b>注意:</b> NMI 中断优先级最高, 写1后立即去执行NMI中断代码清0此位。此意味着只有在执行NMI中断代码时, 此位写1再读, 才能读到1。
[30:29]	Reserved 保留.
[28]	<b>PENDSVSET</b> <b>PendSV中断挂起位</b> 写: 0 = 无效. 1 = 挂起 PendSV 中断 读: 0 = PendSV 没挂起. 1 = PendSV 已挂起. <b>注意:</b> 此位写1是挂起 PendSV 的唯一方式.
[27]	<b>PENDSVRTC_CAL</b> <b>PendSV中断清除位 (只写)</b> 写: 0 = 无效. 1 = 清除 PendSV 的挂起. <b>注意:</b> 欲清除 PENDSV 位, 必须“写 0 到 PENDSVSET 且写 1 到 PENDSVRTC_CAL”在同一时间。

[26]	<b>PENDSTSET</b>	<b>SysTick中断挂起位</b> 写: 0 = 无效. 1 = 挂起SysTick中断. 读: 0 = SysTick 没挂起. 1 = SysTick已挂起.
[25]	<b>PENDSTRTC_CAL</b>	<b>SysTick中断清除位（只写）</b> 写: 0 = 无效. 1 = 清除 SysTick 的挂起状态 <b>注意:</b> 欲清除 PENDST位，必须“写 0 到 PENDSTSET 且写 1 到 PENDSTRTC_CAL”在同一时间
[24]	<b>Reserved</b>	保留
[23]	<b>ISRPREEMPT</b>	<b>中断抢先位(只读)</b> 若为1，调试停止又开始执行后，会去执行中断代码。
[22]	<b>ISRPENDING</b>	<b>中断挂起标志, NMI 和 Faults 除外(只读)</b> 0 = 无中断挂起. 1 = 有中断挂起.
[21:18]	<b>Reserved</b>	保留
[17:12]	<b>VECTPENDING</b>	<b>使能且挂起中断的优先级最高的中断号</b> 0 = 无挂起的中断. 非0 = 使能且挂起的最高优先级的中断号. BASEPRI 和 FAULTMASK 会影响这个结果，但 PRIMASK 不会
[11]	<b>RETTTOBASE</b>	<b>抢先的异常标志</b> 指示是否有抢先的异常 0 = 在执行抢先的中断 1 = 无抢先的中断，或当前中断是唯一活动的中断
[10:7]	<b>Reserved</b>	保留
[6:0]	<b>VECTACTIVE</b>	<b>当前活动的异常或中断</b> 0 = 线程模式. 非0 = 目前活动的异常或中断.

AIRCR 应用中断复位控制寄存器

寄存器	偏移地址	R/W	描述	复位值
AIRCR	SCS_BA+0xD0C	R/W	应用中断和复位控制寄存器	0xFA05_0000

31	30	29	28	27	26	25	24
VECTORKEY							
23	22	21	20	19	18	17	16
VECTORKEY							
15	14	13	12	11	10	9	8
ENDIANNES	Reserved				PRIGROUP		
7	6	5	4	3	2	1	0
Reserved					SYSRESETREQ	VECTCLRACITIVE	VECTRESET

位	描述	
[31:16]	VECTORKEY	访问钥匙 写此寄存器时这部分必须为 0x05FA, 否则结果不可预知。 如此可防止此寄存器数据被意外改写.
[15]	ENDIANNES	数据顺序 0 = 小端格式. 1 = 大端格式
[14:11]	Reserved	保留.
[10:8]	PRIGROUP	中断优先级组 这个值决定了组优先级和子优先级的划分
[7:3]	Reserved	保留.
[2]	SYSRESETREQ	系统复位 写1会让系统复位。复位后此位自动回0
[1]	VECTCLRACTIVE	异常活动状态清除位 写1会清除中断的所有活动状态 此位只写且只能在内核停止时写。 <b>注意:</b> 这是仿真器重新初始化栈区的方法
[0]	VECTRESET	保留.

PRIGROUP	二进制小数点	组优先级位	子优先级位	组优先级数	子优先级数
0b000	bxxxxxxxx.y	[7:1]	[0]	128	2
0b001	bxxxxxx.yy	[7:2]	[1:0]	64	4
0b010	bxxxxx.yyy	[7:3]	[2:0]	32	8
0b011	bxxxx.yyyy	[7:4]	[3:0]	16	16
0b100	bxxx.yyyyy	[7:5]	[4:0]	8	32
0b101	bxx.yyyyyy	[7:6]	[5:0]	4	64
0b110	bx.yyyyyyy	[7]	[6:0]	2	128
0b111	b.yyyyyyyy	None	[7:0]	1	256

表 6.2-10 优先级分组

**SCR 系统控制寄存器**

寄存器	偏移地址	R/W	描述	复位值
SCR	SCS_BA+0xD10	R/W	系统控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved			SEVONPEND	Reserved	SLEEPDEEP	SLEEPONEXIT	Reserved

位	描述	
[31:5]	Reserved	保留.
[4]	SEVONPEND	.挂起事件的唤醒功能选择 0 = 仅使能的中断或事件唤醒 休眠中的CPU. 1 =所有的中断或事件都唤醒CPU，包括未使能的。 如果一个中断或事件被 挂起，可以唤醒 WFE状态的处理器。若处理器未在等事件，事件会被注册并影响下次的 WFE. 处理器可以被外部事件或执行指令SEV 唤醒
[3]	Reserved	保留.
[2]	SLEEPDEEP	休眠模式选择. 0 = 休眠. 1 = 深度休眠.
[1]	SLEEPONEXIT	返回立即休眠使能位 0 = 中断返回时不再休眠. 1 = 中断返回时立即休眠或深度休眠，不会执行main()中的代码
[0]	Reserved	保留.

**SHPR1 系统中断优先级寄存器1**

寄存器	偏移地址	R/W	描述	复位值
SHPR1	SCS_BA+0xD18	R/W	系统中断优先级寄存器1	0x0000_0000

31	30	29	28	27	26	25	24
<b>PRI_11</b>		<b>Reserved</b>					
23	22	21	20	19	18	17	16
<b>PRI_6</b>							
15	14	13	12	11	10	9	8
<b>PRI_5</b>							
7	6	5	4	3	2	1	0
<b>PRI_4</b>							

位	描述	
[31:24]	<b>Reserved</b>	Reserved.
[23:16]	<b>PRI_6</b>	系统中断6的优先级, 用法错中断UsageFault
[15:8]	<b>PRI_5</b>	系统中断5的优先级, 总线错中断BusFault
[7:0]	<b>PRI_4</b>	系统中断4的优先级, 内存错中断 MemManage

**SHPR2 系统中断优先级寄存器2**

寄存器	偏移地址	R/W	描述	复位值
SHPR2	SCS_BA+0xD1C	R/W	系统中断优先级寄存器2	0x0000_0000

31	30	29	28	27	26	25	24
PRI_11	Reserved						
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved							

位	描述	
[31:30]	PRI_11	系统中断11 – SVCall的优先级 “0”最高, “3”最低
[29:0]	Reserved	保留.

## SHPR3 系统中断优先级寄存器3

寄存器	偏移地址	R/W	描述	复位值
SHPR3	SCS_BA+0xD20	R/W	系统中断优先级寄存器3	0x0000_0000

31	30	29	28	27	26	25	24
<b>PRI_15</b>		Reserved					
23	22	21	20	19	18	17	16
<b>PRI_14</b>		Reserved					
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved							

位	描述	
[31:30]	<b>PRI_15</b>	系统中断 15 – SysTick系统定时器的优先级 “0”最高, “3”最低
[29:24]	<b>Reserved</b>	保留.
[23:22]	<b>PRI_14</b>	系统中断14 – PendSV的中断优先级 “0”最高, “3”最低
[21:0]	<b>Reserved</b>	保留.

## 6.3 时钟控制器

### 6.3.1 概述

时钟控制器为整个芯片提供时钟源，包括系统时钟和所有外围设备时钟。该控制器还通过单独时钟的开或关，时钟源选择和分频器来进行功耗控制。在CPU使能低功耗PDEN(CLK\_PWRCTL[7])位和Cortex®-M4内核执行WFI指令后，芯片进入低功耗模式。直到唤醒中断发生，芯片才会退出低功耗模式。在低功耗模式下，时钟控制器会关闭外部4~24MHz高速晶振和内部12MHz高速RC振荡器，以降低整个系统功耗。下图所示各模块时钟发生器和时钟源的简图。

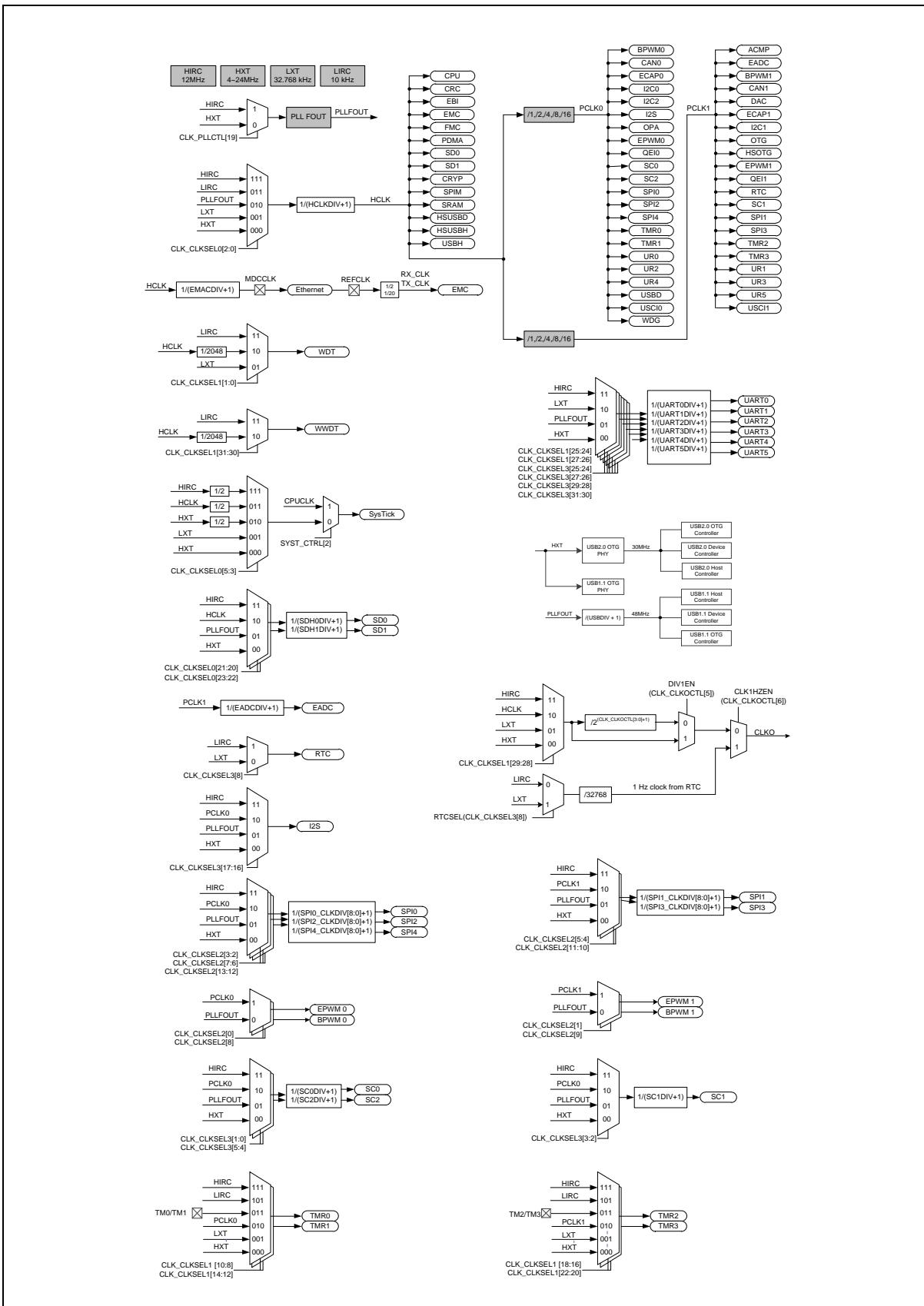


图 6.3-1 时钟发生器全局示意图

### 6.3.2 时钟发生器

时钟发生器由如下5个时钟源组成:

- 外部32.768 kHz低速晶振 (LXT)
- 外部4~24 MHz高速晶振(HXT)
- 可编程的PLL输出时钟频率(PLLFOUT), PLL 由外部 4~24 MHz 晶振或内部 12 MHz 振荡器提供时钟源
- 内部12 MHz高速振荡器(HIRC)
- 内部10 kHz低速RC振荡器(LIRC)

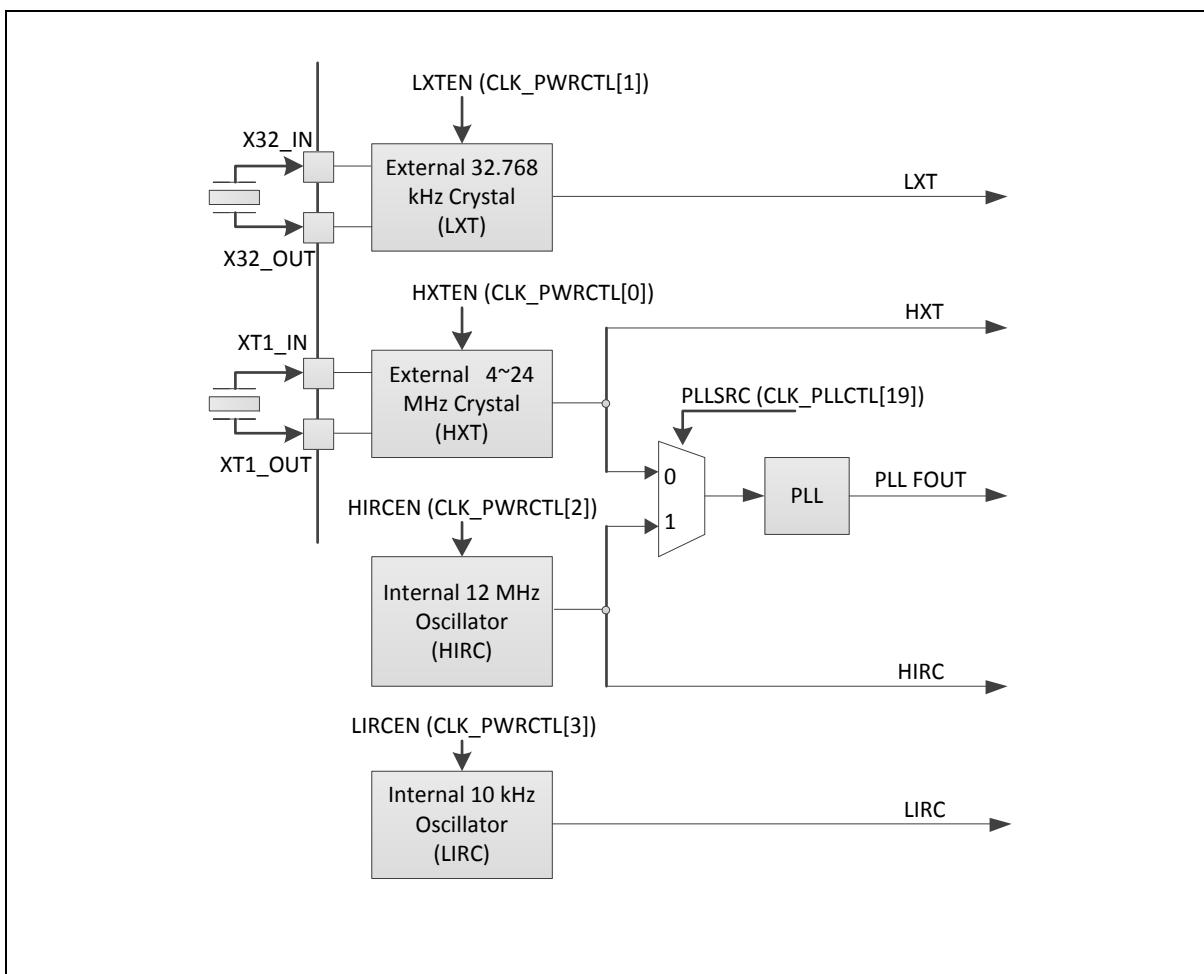


图 6.3-2 时钟发生器框图

### 6.3.3 系统时钟和SysTick 时钟

系统时钟有 5 个可选时钟源，由时钟发生器产生。时钟源切换取决于寄存器 HCLKSEL (CLK\_CLKSEL0[2:0])。其框图如下所示。

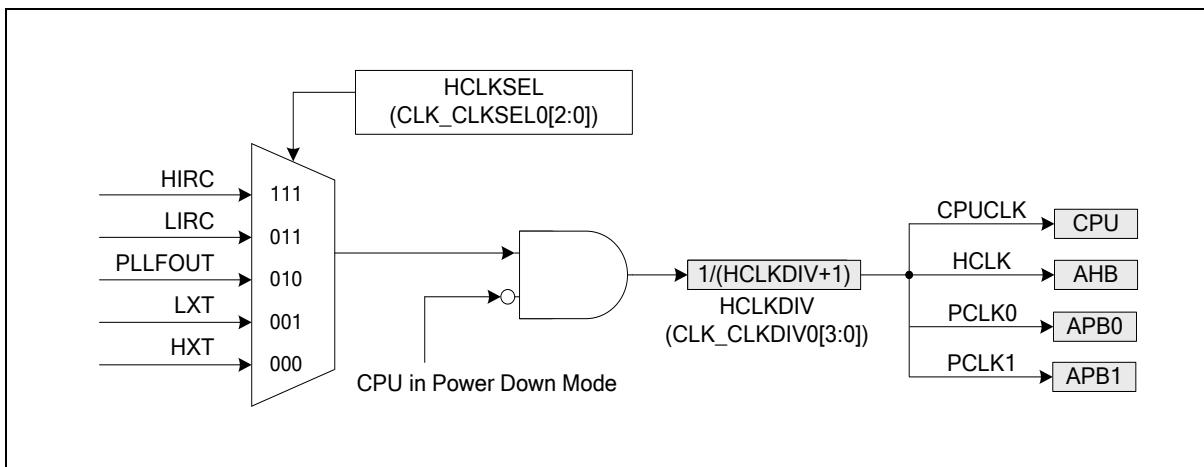


图 6.3-3 系统时钟框图

有两个时钟失败检测器来观察HXT和LXT时钟源的状况，而且都有独立的使能和中断控制设置。当HXT检测器使能时，HIRC时钟也自动使能。当LXT检测器使能时，LIRC时钟也自动使能。

当HXT时钟检测器使能，如果检测到HXT时钟停止且满足下面条件：系统的时钟源来自HXT或者系统时钟源来自PLL(PLL的输入时钟为HXT)，系统时钟将自动切换到HIRC。如果HXT时钟停止条件检测到，HXTFIF (CLK\_CLKDSTS[0])将被设置为1，此时如果HXTFIE (CLK\_CLKDCTL[5])有置位，芯片将进入中断。用户可以试着去恢复HXT，通过禁止HXT和重新使能HXT来确认时钟稳定标志位是否已设置为1。如果HXT时钟稳定标志位有设置为1，这就意味着HXT在重新使能后已恢复震荡，此时用户可以重新把系统时钟切换到HXT了。

HXT时钟停止检测和系统时钟切换到HIRC的过程如下图所示：

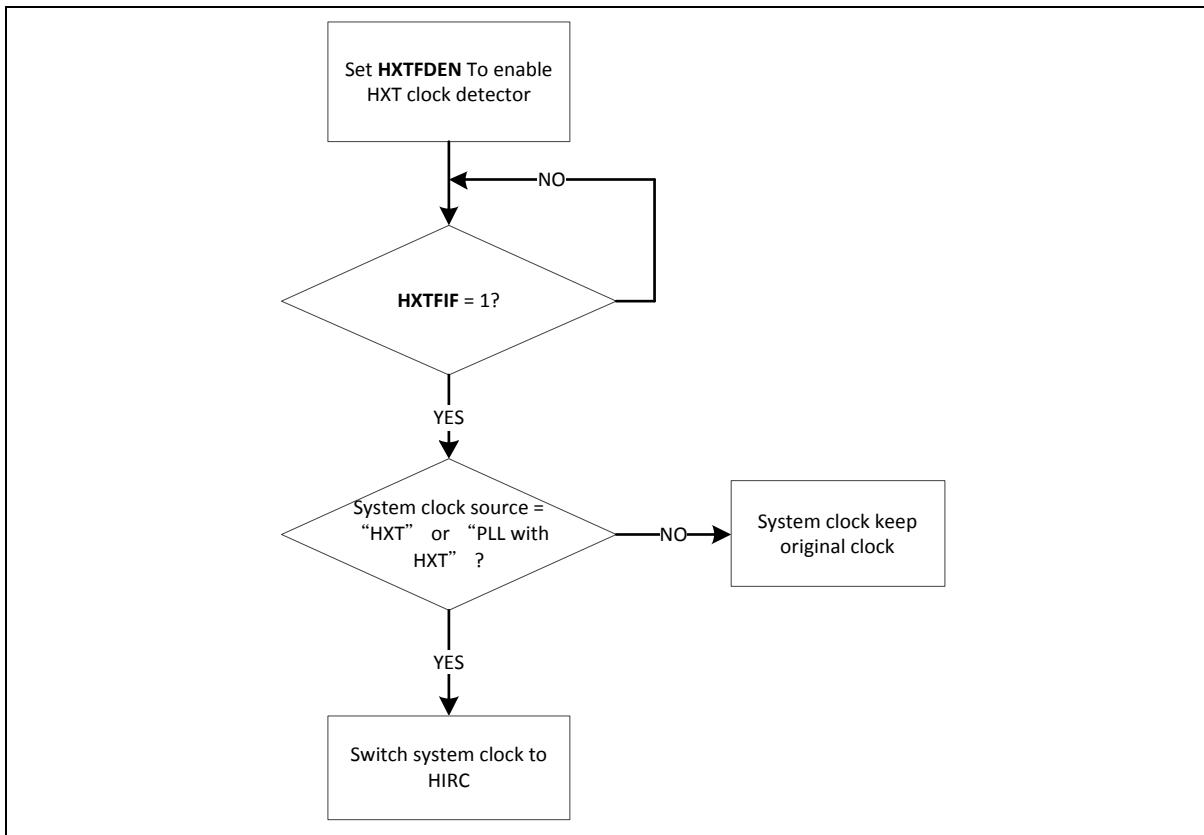


图 6.3-4 HXT 停止保护过程

Cortex®-M4内核的SysTick时钟源可以选择CPU时钟或外部时钟(SYST\_CTRL[2])。如果使用外部时钟，SysTick时钟(STCLK)有5个可选时钟源。时钟源切换取决于寄存器STCLKSEL(CLK\_CLKSEL0[5:3])。其框图如下所示：

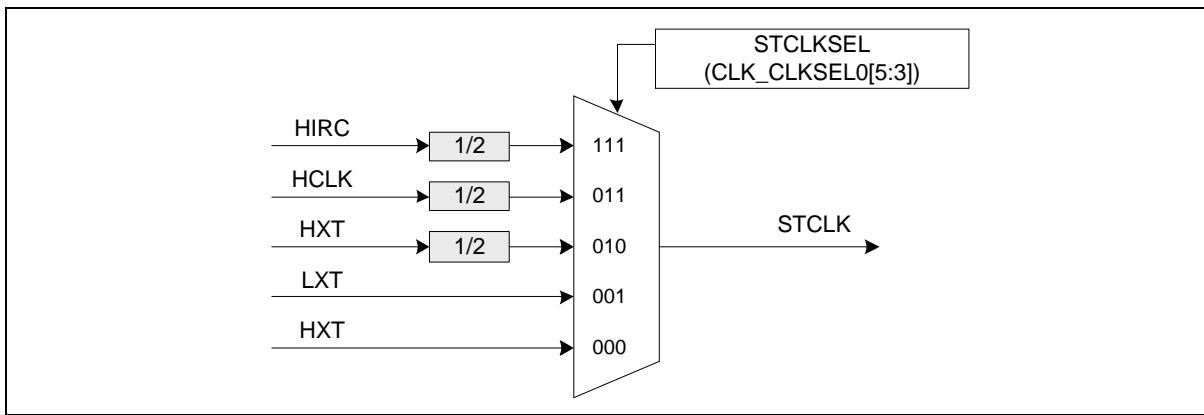


图 6.3-5 时钟控制框图

### 6.3.4 外设时钟

外设时钟可以有不同的时钟源做切换设置。主要取决于不同的外设。请参看寄存器CLK\_CLKSEL1、CLK\_CLKSEL2和CLK\_CLKSEL3的描述。

### 6.3.5 掉电模式时钟

当芯片进入掉电模式，系统时钟和一些时钟源以及一些外设时钟将被关闭。也有一些时钟源与外设时钟仍在工作。

如下时钟仍在工作：

- 时钟发生器
  - 10 kHz内部低速RC振荡器
  - 32.768 kHz外部低速晶振时钟
- 外设时钟 (当模块的时钟源来自32.768 kHz外部低速晶振时钟或10 kHz内部低速RC振荡器)

### 6.3.6 时钟输出

该设备带有一个 $2^N$ 的若干次幂的频率分频器，该分频器由16个链式的二分频器组成的移位寄存器构成。其中哪一级的值被输出，由一个16选1的多路转换器选择，该多路转换器接到CLKO管脚上。因此共有16种的时钟分频选择，分频范围从 $F_{in}/2^1$ 到 $F_{in}/2^{16}$ ，此处 $F_{in}$ 是到时钟分频器的时钟输入频率。

输出公式： $F_{out} = F_{in}/2^{(N+1)}$ ，其中  $F_{in}$  为输入时钟频率， $F_{out}$  为时钟分频器输出频率， $N$  为FREQSEL (CLK\_CLKOCTL[3:0])中的4位值。

往CLKOEN (CLK\_CLKOCTL[4])写1，分级计数器开始计数。往CLKOEN (CLK\_CLKOCTL[4])写0，分级计数器持续计数，直到分频时钟达到低电平并会保持在低电平状态。

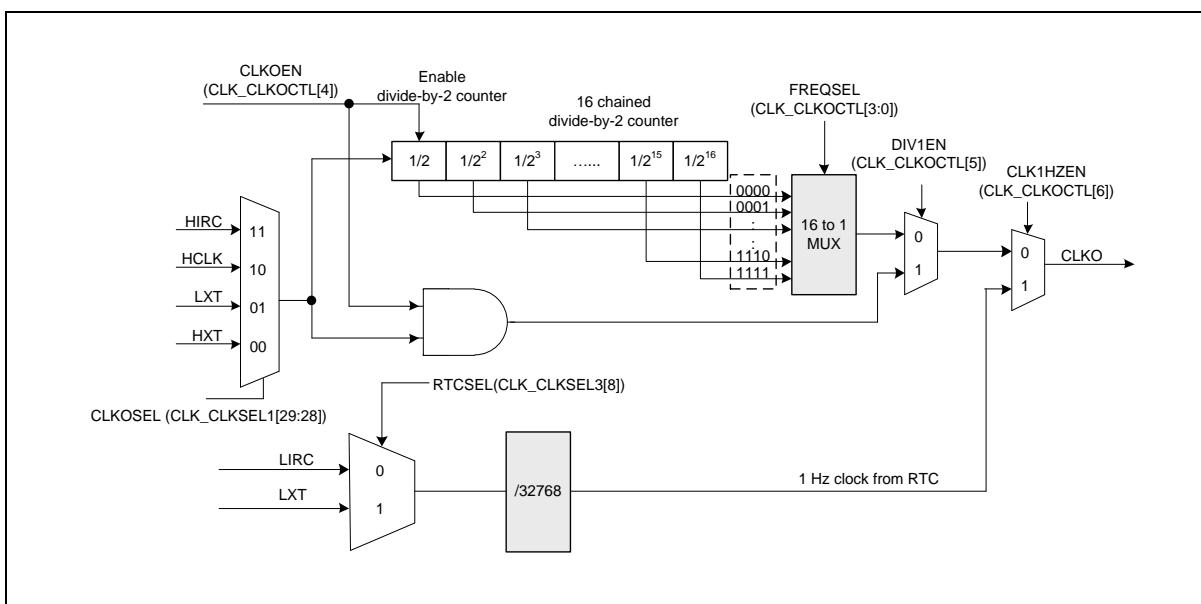


图 6.3-6 时钟输出框图

### 6.3.7 USB时钟源

USB 1.0 和 2.0系统的时钟源来自USB2.0 PHY时钟或可编程PLL的输出。产生的时钟如下图所示：

PLL2是时钟分频器输出频率，输出公式是： $(480 \text{ MHz}) / 2 / (\text{PLL2DIV} + 1)$ 。

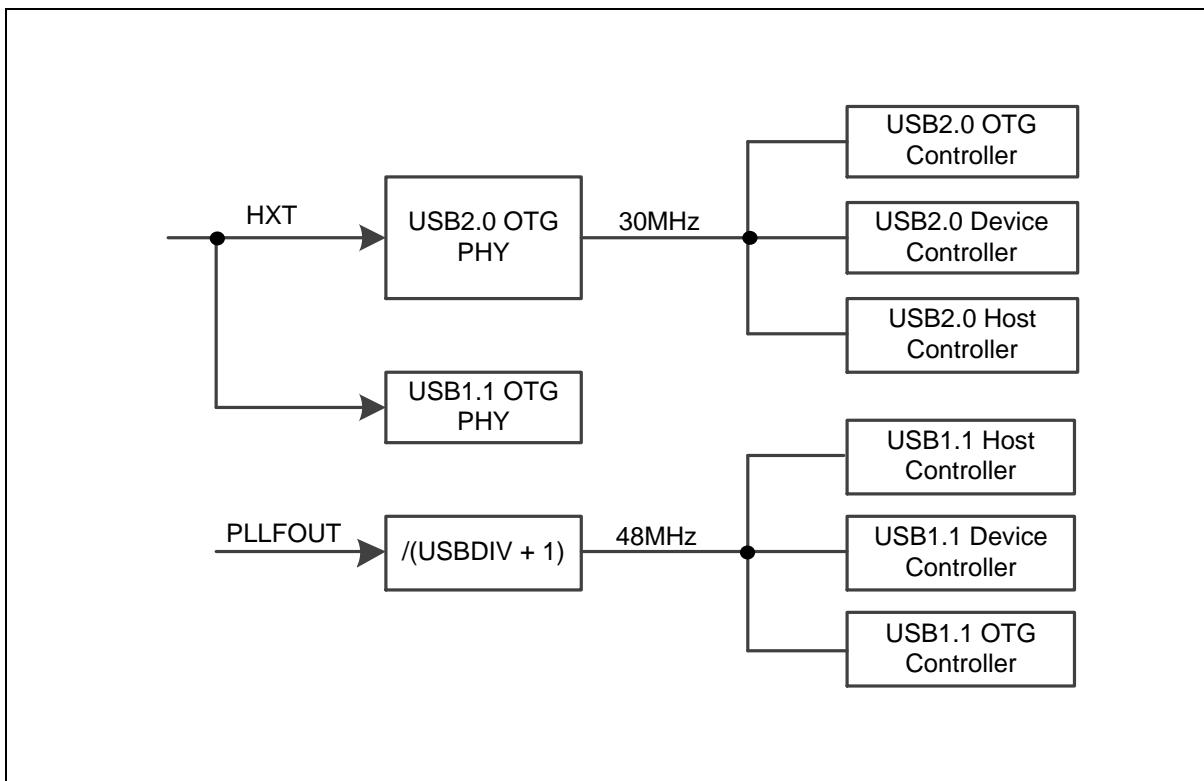


图 6.3-7 USB 时钟源

### 6.3.8 寄存器映射

R: 只读, W: 只写, R/W: 读/写

寄存器	偏移地址	R/W	描述	复位值
<b>CLK 基地址:</b>				
<b>CLK_BA = 0x4000_0200</b>				
<b>CLK_PWRCTL</b>	CLK_BA+0x00	R/W	系统掉电控制寄存器	0x0000_XX1X
<b>CLK_AHBCLK</b>	CLK_BA+0x04	R/W	AHB 设备时钟使能控制寄存器	0x0000_8004
<b>CLK_APBCLK0</b>	CLK_BA+0x08	R/W	APB 设备时钟使能控制寄存器 0	0x0000_0001
<b>CLK_APBCLK1</b>	CLK_BA+0x0C	R/W	APB 设备时钟使能控制寄存器 1	0x0000_0000
<b>CLK_CLKSEL0</b>	CLK_BA+0x10	R/W	时钟源选择控制寄存器 0	0x00F3_013X
<b>CLK_CLKSEL1</b>	CLK_BA+0x14	R/W	时钟源选择控制寄存器 1	0xBF77_7703
<b>CLK_CLKSEL2</b>	CLK_BA+0x18	R/W	时钟源选择控制寄存器 2	0x0000_03AB
<b>CLK_CLKSEL3</b>	CLK_BA+0x1C	R/W	时钟源选择控制寄存器 3	0xFF00_003F
<b>CLK_CLKDIV0</b>	CLK_BA+0x20	R/W	时钟分频数目寄存器 0	0x0000_0000
<b>CLK_CLKDIV1</b>	CLK_BA+0x24	R/W	时钟分频数目寄存器 1	0x0000_0000
<b>CLK_CLKDIV3</b>	CLK_BA+0x2C	R/W	时钟分频数目寄存器 3	0x0000_0000
<b>CLK_CLKDIV4</b>	CLK_BA+0x30	R/W	时钟分频数目寄存器 4	0x0000_0000
<b>CLK_PCLKDIV</b>	CLK_BA+0x34	R/W	APB 时钟分频器寄存器	0x0000_0000
<b>CLK_PLLCTL</b>	CLK_BA+0x40	R/W	PLL 控制寄存器	0x0005_C02E
<b>CLK_STATUS</b>	CLK_BA+0x50	R	时钟状态监测寄存器	0x0000_00XX
<b>CLK_CLKOCTL</b>	CLK_BA+0x60	R/W	时钟输出控制寄存器	0x0000_0000
<b>CLK_CLKDCTL</b>	CLK_BA+0x70	R/W	时钟失败检测器控制寄存器	0x0000_0000
<b>CLK_CLKDSTS</b>	CLK_BA+0x74	R/W	时钟失败检测器状态寄存器	0x0000_0000
<b>CLK_CDUPB</b>	CLK_BA+0x78	R/W	时钟频率检测器上边界寄存器	0x0000_0000
<b>CLK_CDLOWB</b>	CLK_BA+0x7C	R/W	时钟频率检测器下边界寄存器	0x0000_0000
<b>CLK_PMUCTL</b>	CLK_BA+0x90	R/W	电源管理控制寄存器	0x0000_0080
<b>CLK_PMUSTS</b>	CLK_BA+0x94	R/W	电源管理状态寄存器	0x0000_0000
<b>CLK_LDOCTL</b>	CLK_BA+0x98	R/W	LDO控制寄存器	0x0000_0000
<b>CLK_SWKDBCTL</b>	CLK_BA+0x9C	R/W	待机掉电唤醒抖控制寄存器	0x0000_0000
<b>CLK_PASWKCTL</b>	CLK_BA+0xA0	R/W	GPA 待机掉电唤醒控制寄存器	0x0000_0000
<b>CLK_PBSWKCTL</b>	CLK_BA+0xA4	R/W	GPB 待机掉电唤醒控制寄存器	0x0000_0000

<b>CLK_PCSWKCTL</b>	CLK_BA+0xA8	R/W	GPC 待机掉电唤醒控制寄存器	0x0000_0000
<b>CLK_PDSWKCTL</b>	CLK_BA+0xAC	R/W	GPD 待机掉电唤醒控制寄存器	0x0000_0000
<b>CLK_IOPDCTL</b>	CLK_BA+0xB0	R/W	GPIO 待机掉电控制寄存器	0x0000_0000

## 6.3.9 寄存器描述

CLK\_PWRCTL 系统掉电控制寄存器

寄存器	偏移地址	R/W	描述	复位值
CLK_PWRCTL	CLK_BA+0x00	R/W	系统掉电控制寄存器	0x0000_XX1X

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved						HIRCSTBS	
15	14	13	12	11	10	9	8
Reserved		HXTTBEN	HXTSELTYP	HXTGAIN		Reserved	
7	6	5	4	3	2	1	0
PDEN	PDWKIF	PDWKIEN	PDWKDLY	LIRCEN	HIRCEN	LXTEN	HXTEN

位	描述	
[31:18]	Reserved	保留
[17:16]	HIRCSTBS	<p><b>HIRC 稳定判断周期选择 (写保护)</b>            该位写保护，操作前需要进行解锁操作            00 = HIRC 稳定周期= 64 clocks.            01 = HIRC 稳定周期= 24 clocks.            其它值：保留</p>
[15:14]	Reserved	保留
[13]	HXTTBEN	<p><b>HXT 晶振 TURBO 模式 (写保护)</b>            该位写保护，操作前需要进行解锁操作            0 = HXT 晶振 TURBO 模式禁止.            1 = HXT 晶振 TURBO 模式使能..</p>
[12]	HXTSELTYP	<p><b>HXT 晶振类别选择位(写保护)</b>            该位写保护，操作前需要进行解锁操作            0 =选择 INV类型.            1 =选择GM.类型  <b>Note:</b> 该位写保护，参看SYS_REGLCTL寄存器</p>
[11:10]	HXTGAIN	<p><b>HXT 增益控制位(写保护)</b>            该位写保护，操作前需要进行解锁操作.            增益控制用来放大晶振的增益，来确保晶振工作正常。增益控制使能时比增益关闭时，晶振需消耗更多的能量。            00 = HXT频率低于8 MHz.            01 = HXT频率在8 MHz 到 12 MHz之间            10 = HXT频率在12 MHz 到 16 MHz之间</p>

		11 = HXT频率高于16 MHz <b>注意:</b> 该位写保护, 参看SYS_REGLCTL寄存器
[9:8]	<b>Reserved</b>	保留.
[7]	<b>PDEN</b>	<p><b>系统掉电模式使能位(写保护)</b> 当该位被置1, 掉电模式使能, 芯片保持运行状态直到CPU睡眠模式也激活, 然后芯片进入掉电模式。 当芯片从掉电模式中被唤醒, 该位自动清零。用户需要重新设置该位才能进入下一次的掉电模式。 在掉电模式下, HXT 和 HIRC将被禁用, 但是LXT 和 LIRC不受掉电模式的控制。 在掉电模式下, PLL 与系统时钟将被禁用, 忽略时钟源选择。如果外设时钟源为LXT 或 LIRC, 则外设的时钟不受掉电模式的控制。</p> <p>0 = 在CPU 休眠命令 WFI后, 芯片不会进入掉电模式 1 = 在CPU 休眠命令 WFI后, 芯片进入掉电模式 <b>注意:</b> 该位写保护, 参看SYS_REGLCTL寄存器</p>
[6]	<b>PDWKIF</b>	<p><b>芯片掉电模式唤醒中断状态</b> 该位由掉电唤醒事件置位, 表示从掉电模式恢复过来。 任何唤醒源发生, 该标志置位。请参考电源模式和唤醒源章节。 <b>注意1:</b> 该位写 1清零。 <b>注意2:</b> 只有在PDWKIEN (CLK_PWRCTL[5])置1后, 该位才工作。</p>
[5]	<b>PDWKIEN</b>	<p><b>掉电模式唤醒中断使能位 (写保护)</b> 0 = 禁用掉电模式唤醒中断 1 = 使能掉电模式唤醒中断. <b>注意1:</b> 当PDWKIF 和 PDWKIEN 都为1时, 该中断将产生 <b>注意2:</b> 该位写保护, 参看SYS_REGLCTL寄存器</p>
[4]	<b>PDWKDLY</b>	<p><b>使能唤醒延时计数器 (写保护)</b> 当芯片从掉电模式中被唤醒时, 时钟控制将会延迟一定的时钟周期以等待系统时钟稳定。 当芯片工作在外部 4~24 MHz高速晶振时, 延迟时钟周期为 4096 个时钟周期; 当芯片工作在内部 12 MHz 高速振荡器时, 延迟时钟周期为 256 个时钟周期。 0 = 禁用时钟周期的延迟. 1 = 使能时钟周期的延迟 <b>注意:</b> 该位写保护, 参看SYS_REGLCTL寄存器</p>
[3]	<b>LIRCEN</b>	<p><b>LIRC使能位 (写保护)</b> 0 = 禁用10 kHz 内部低速 RC 振荡器(LIRC) 1 = 使能10 kHz 内部低速 RC 振荡器(LIRC) <b>注意:</b> 该位写保护, 参看SYS_REGLCTL寄存器</p>
[2]	<b>HIRCEN</b>	<p><b>HIRC使能位 (写保护)</b> 0 = 禁用12 MHz 内部高速RC振荡器(HIRC) 1 = 使能12 MHz 内部高速RC振荡器(HIRC) <b>注意:</b> 该位写保护, 参看SYS_REGLCTL寄存器</p>
[1]	<b>LXTEN</b>	<p><b>LXT 使能位 (写保护)</b> 0 = 禁用32.768 kHz 外部低速晶振(LXT) 1 = 使能32.768 kHz 外部低速晶振(LXT) <b>注意:</b> 该位写保护, 参看SYS_REGLCTL寄存器</p>

[0]	HXTEN	<b>HXT 使能位（写保护）</b> 该位的默认值由 flash 控制器的用户配置寄存器CONFIG0 [26:24] 设置。当默认的时钟源为 HXT 时，该位自动置1。 0 = 禁用4 ~ 24 MHz 外部高速晶振(HXT) 1 = 使能4~24 MHz 外部高速晶振(HXT) <b>注意:</b> 该位写保护，参看SYS_REGLCTL 寄存器
-----	-------	---

CLK\_AHBCLK AHB 设备时钟使能控制寄存器

该寄存器的各位用于使能/禁用系统时钟， AHB总线设备时钟。

寄存器	偏移量	R/W	描述	复位值
CLK_AHBCLK	CLK_BA+0x04	R/W	AHB设备时钟使能控制寄存器	0x0000_8004

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved						SDH1CKEN	USBHCKEN
15	14	13	12	11	10	9	8
FMCIDLE	SPIMCKEN	Reserved	CRYPTOCKE_N	Reserved	HSUSBDCKE_N	Reserved	
7	6	5	4	3	2	1	0
CRCCKEN	SDH0CKEN	EMACCKEN	Reserved	EBICKEN	ISPCKEN	PDMACKEN	Reserved

位	描述
[31:18]	Reserved 保留
[17]	<b>SDH1CKEN</b> SD1控制器时钟使能位 0 = SD1 引擎时钟禁止 1 = SD1 引擎时钟使能
[16]	<b>USBHCKEN</b> USB HOST 控制器时钟使能位 0 = USB HOST 外设时钟禁止 1 = USB HOST外设时钟使能
[15]	<b>FMCIDLE</b> 空闲模式时, Flash 内存控制器时钟使能位 0 = 芯片工作于空闲模式时, 禁止FMC 外设时钟 1 = 芯片工作于空闲模式时, 使能FMC 外设时钟
[14]	<b>SPIMCKEN</b> SPIM 控制器时钟使能位 0 = SPIM 控制器时钟禁止 1 = SPIM 控制器时钟使能
[13]	Reserved 保留
[12]	<b>CRYPTOCKEN</b> 加解密算法加速器时钟使能位 0 = 加解密加速器时钟禁止 1 = 加解密加速器时钟使能
[11]	Reserved 保留
[10]	<b>HSUSBDCKEN</b> HSUSB Device 时钟使能位 0 = HSUSB device 控制器时钟禁止 1 = HSUSB device 控制器时钟使能

[9:8]	Reserved	保留
[7]	<b>CRCCKEN</b>	<b>CRC</b> 产生控制器时钟使能位 0 = CRC 外设时钟禁止 1 = CRC外设时钟使能
[6]	<b>SDH0CKEN</b>	<b>SD0</b> 控制器时钟使能位 0 = SD0 引擎时钟禁止 1 = SD0 引擎时钟使能
[5]	<b>EMACCKEN</b>	以太网控制器时钟使能位 0 = 以太网控制器引擎时钟禁止 1 = 以太网控制器引擎时钟使能
[4]	Reserved	保留
[3]	<b>EBICKEN</b>	<b>EBI</b> 控制器时钟使能位 0 = EBI 外设时钟禁止 1 = EBI 外设时钟使能
[2]	<b>ISPCKEN</b>	<b>Flash ISP</b> 控制器时钟使能位 0 = Flash ISP 外设时钟禁止. 1 = Flash ISP 外设时钟使能
[1]	<b>PDMACKEN</b>	<b>PDMA</b> 控制器时钟使能位 0 = PDMA 外设时钟禁止. 1 = PDMA 外设时钟使能
[0]	Reserved	保留

**CLK\_APBCLK0 APB设备时钟使能控制寄存器 0**

该寄存器各位用于使能/禁用外设控制器时钟.

寄存器	偏移量	R/W	描述				复位值
<b>CLK_APBCLK0</b>	CLK_BA+0x08	R/W	APB设备时钟使能控制寄存器 0				0x0000_0001

31	30	29	28	27	26	25	24
Reserved	HSOTGCKEN	I2S0CKEN	EADCCKEN	USBDCKEN	OTGCKEN	CAN1CKEN	CAN0CKEN
23	22	21	20	19	18	17	16
Reserved		UART5CKEN	UART4CKEN	UART3CKEN	UART2CKEN	UART1CKEN	UART0CKEN
15	14	13	12	11	10	9	8
SPI2CKEN	SPI1CKEN	SPI0CKEN	QSPI0CKEN	Reserved	I2C2CKEN	I2C1CKEN	I2C0CKEN
7	6	5	4	3	2	1	0
ACMP01CKEN	CLKOCKEN	TMR3CKEN	TMR2CKEN	TMR1CKEN	TMR0CKEN	RTCKEN	WDTCKEN

位	描述
[31]	Reserved 保留
[30]	<b>HSOTGCKEN</b> HSUSB OTG 时钟使能位 0 = HSUSB OTG 时钟禁止 1 = HSUSB OTG 时钟使能
[29]	<b>I2S0CKEN</b> I <sup>2</sup> S0 时钟使能位 0 = I <sup>2</sup> S0 时钟禁止. 1 = I <sup>2</sup> S0 时钟使能
[28]	<b>EADCCKEN</b> 增强型模数转换器 (EADC) 时钟使能位 0 = EADC 时钟禁止 1 = EADC 时钟使能
[27]	<b>USBDCKEN</b> USB Device 时钟使能位 0 = USB Device 时钟禁止 1 = USB Device 时钟使能
[26]	<b>OTGCKEN</b> USB OTG 时钟使能位 0 = USB OTG 时钟禁止 1 = USB OTG 时钟使能.
[25]	<b>CAN1CKEN</b> CAN1 时钟使能位 0 = CAN1 时钟禁止. 1 = CAN1 时钟使能
[24]	<b>CAN0CKEN</b> CAN0 时钟使能位 0 = CAN0 时钟禁止.. 1 = CAN0 时钟使能.

[23:22]	<b>Reserved</b>	保留
[21]	<b>UART5CKEN</b>	<b>UART5 时钟使能位</b> 0 = UART5 时钟禁止 1 = UART5 时钟使能
[20]	<b>UART4CKEN</b>	<b>UART4 时钟使能位</b> 0 = UART4 时钟禁止 1 = UART4 时钟使能.
[19]	<b>UART3CKEN</b>	<b>UART3 时钟使能位</b> 0 = UART3 时钟禁止 1 = UART3 时钟使能.
[18]	<b>UART2CKEN</b>	<b>UART2 时钟使能位</b> 0 = UART2 时钟禁止 1 = UART2 时钟使能
[17]	<b>UART1CKEN</b>	<b>UART1 时钟使能位</b> 0 = UART1 时钟禁止 1 = UART1 时钟使能
[16]	<b>UART0CKEN</b>	<b>UART0 时钟使能位</b> 0 = UART0 时钟禁止 1 = UART0 时钟使能
[15]	<b>SPI2CKEN</b>	<b>SPI2 时钟使能位</b> 0 = SPI2 时钟禁止 1 = SPI2 时钟使能
[14]	<b>SPI1CKEN</b>	<b>SPI1 时钟使能位</b> 0 = SPI1 时钟禁止 1 = SPI1 时钟使能
[13]	<b>SPI0CKEN</b>	<b>SPI0 时钟使能位</b> 0 = SPI0 时钟禁止. 1 = SPI0 时钟使能
[12]	<b>QSPI0CKEN</b>	<b>QSPI0 时钟使能位</b> 0 = QSPI0 时钟禁止. 1 = QSPI0 时钟使能.
[11]	<b>Reserved</b>	保留
[10]	<b>I2C2CKEN</b>	<b>I2C2 时钟使能位</b> 0 = I2C2 时钟禁止. 1 = I2C2 时钟使能
[9]	<b>I2C1CKEN</b>	<b>I2C1 时钟使能位</b> 0 = I2C1 时钟禁止. 1 = I2C1 时钟使能
[8]	<b>I2C0CKEN</b>	<b>I2C0 时钟使能位</b> 0 = I2C0 时钟禁止.

		1 = I2C0 时钟使能
[7]	<b>ACMP01CKEN</b>	模拟比较器 0/1 时钟使能位 0 = 模拟比较器 0/1 时钟禁止 1 = 模拟比较器 0/1 时钟使能.
[6]	<b>CLKOCKEN</b>	<b>CLKO</b> 时钟使能位 0 = CLKO 时钟禁止. 1 = CLKO 时钟使能
[5]	<b>TMR3CKEN</b>	<b>Timer3</b> 时钟使能位 0 = Timer3 时钟禁止. 1 = Timer3 时钟使能
[4]	<b>TMR2CKEN</b>	<b>Timer2</b> 时钟使能位 0 = Timer2 时钟禁止 1 = Timer2 时钟使能.
[3]	<b>TMR1CKEN</b>	<b>Timer1</b> 时钟使能位 0 = Timer1 时钟禁止 1 = Timer1 时钟使能
[2]	<b>TMR0CKEN</b>	<b>Timer0</b> 时钟使能位 0 = Timer0 时钟禁止. 1 = Timer0 时钟使能
[1]	<b>RTCCKEN</b>	<b>实时时钟 (Real-Time-Clock) APB 接口时钟使能位</b> 该位仅用于控制 RTC APB 时钟，RTC 外设的时钟源由RTCSEL(CLK_CLKSEL3[8])寄存器设置，可以选择32.768 kHz外部低速晶振或内部10kHz低速RC振荡器。 0 = RTC 时钟禁止 1 = RTC 时钟使能.
[0]	<b>WDTCKEN</b>	<b>看门狗定时器时钟使能位 (写保护)</b> 0 = 看门狗定时器时钟禁止 1 = 看门狗定时器时钟使能 <b>注意:</b> 该位写保护，参看SYS_REGLCTL寄存器

**CLK\_APBCLK1 APB设备时钟使能控制寄存器1**

该寄存器各位用于使能/禁用外设控制器时钟.

寄存器	偏移量	R/W	描述	复位值
CLK_APBCLK1	CLK_BA+0x0C	R/W	APB 设备时钟使能控制寄存器1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved	OPACKEN	Reserved		ECAP1CKEN	ECAP0CKEN	Reserved	
23	22	21	20	19	18	17	16
QEI1CKEN	QEIOCKEN	Reserved		BPWM1CKEN	BPWM0CKEN	EPWM1CKEN	EPWM0CKEN
15	14	13	12	11	10	9	8
Reserved			DACCKEN	Reserved		USCI1CKEN	USCI0CKEN
7	6	5	4	3	2	1	0
Reserved	SPI3CKEN	Reserved			SC2CKEN	SC1CKEN	SC0CKEN

位	描述
[31]	Reserved 保留
[30]	OP 放大器 (OPA) 时钟使能位 0 = OPA 时钟禁止. 1 = OPA 时钟使能
[29:28]	Reserved 保留
[27]	ECAP1CKEN ECAP1 时钟使能位 0 = ECAP1 时钟禁止 1 = ECAP1 时钟使能
[26]	ECAP0CKEN ECAP0 时钟使能位 0 = ECAP0 时钟禁止. 1 = ECAP0 时钟使能
[25:24]	Reserved 保留
[23]	QEI1CKEN QEI1 时钟使能位 0 = QEI1 时钟禁止. 1 = QEI1 时钟使能
[22]	QEIOCKEN QEIO 时钟使能位 0 = QEIO 时钟禁止 1 = QEIO 时钟使能
[21:20]	Reserved 保留
[19]	BPWM1CKEN BPWM1 时钟使能位 0 = BPWM1 时钟禁止

		1 = BPWM1 时钟使能
[18]	<b>BPWM0CKEN</b>	<b>BPWM0</b> 时钟使能位 0 = BPWM0 时钟禁止 1 = BPWM0 时钟使能.
[17]	<b>EPWM1CKEN</b>	<b>EPWM1</b> 时钟使能位 0 = EPWM1 时钟禁止 1 = EPWM1 时钟使能
[16]	<b>EPWM0CKEN</b>	<b>EPWM0</b> 时钟使能位 0 = EPWM0 时钟禁止 1 = EPWM0 时钟使能
[15:13]	<b>Reserved</b>	保留
[12]	<b>DACCKEN</b>	<b>DAC</b> 时钟使能位 0 = DAC 时钟禁止 1 = DAC 时钟使能
[11:10]	<b>Reserved</b>	保留
[9]	<b>USCI1CKEN</b>	<b>USCI1</b> 时钟使能位 0 = USCI1 时钟禁止 1 = USCI1 时钟使能.
[8]	<b>USCI0CKEN</b>	<b>USCI0</b> 时钟使能位 0 = USCI0 时钟禁止. 1 = USCI0 时钟使能
[7]	<b>Reserved</b>	保留
[6]	<b>SPI3CKEN</b>	<b>SPI3</b> 时钟使能位 0 = SPI3 时钟禁止. 1 = SPI3 时钟使能
[5:3]	<b>Reserved</b>	保留
[2]	<b>SC2CKEN</b>	<b>SC2</b> 时钟使能位 0 = SC2 时钟禁止 1 = SC2 时钟使能.
[1]	<b>SC1CKEN</b>	<b>SC1</b> 时钟使能位 0 = SC1 时钟禁止. 1 = SC1 时钟使能
[0]	<b>SC0CKEN</b>	<b>SC0</b> 时钟使能位 0 = SC0 时钟禁止 1 = SC0 时钟使能

CLK\_CLKSEL0 时钟源选择控制寄存器 0

寄存器	偏移量	R/W	描述	复位值
CLK_CLKSEL0	CLK_BA+0x10	R/W	时钟源选择控制寄存器0	0x00F3_013F

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
SDH1SEL							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved		STCLKSEL			HCLKSEL		

位	描述	
[31:22]	<b>Reserved</b>	保留.
[23:22]	<b>SDH1SEL</b>	<p><b>SD1 引擎时钟源选择 (写保护)</b>            00 = 时钟源来自 HXT 时钟.            01 = 时钟源来自 PLL 时钟..            10 = 时钟源来自 HCLK.            11 = 时钟源来自 HIRC 时钟.</p>
[21:20]	<b>SDH0SEL</b>	<p><b>SD0 引擎时钟源选择 (写保护)</b>            00 = 时钟源来自 HXT 时钟            01 = 时钟源来自 PLL 时钟.            10 = 时钟源来自 HCLK            11 = 时钟源来自 HIRC 时钟.</p>
[19:6]	<b>Reserved</b>	保留.
[5:3]	<b>STCLKSEL</b>	<p><b>Cortex®-M4 SysTick时钟源选择(写保护)</b>            如果 SYST_CTRL[2]=0, SysTick 使用下列时钟源.            000 = 时钟源来自HXT.            001 = 时钟源来自LXT.            010 = 时钟源来自HXT/2            011 = 时钟源来自HCLK/2.            111 = 时钟源来自HIRC/2.  <b>注意1:</b> 如果 SysTick 时钟源不是来自 HCLK (例如 SYST_CTRL [2] = 0), SysTick 时钟源必须小于或等于HCLK/2  <b>注意2:</b> 该位写保护, 参看SYS_REGLCTL寄存器</p>
[2:0]	<b>HCLKSEL</b>	<b>HCLK 时钟源选择(写保护)</b> 在时钟切换前, 相关时钟源 (预选和新选) 必须打开。

		<p>000 = 时钟来自 HXT. 001 = 时钟来自 LXT. 010 = 时钟来自 PLL. 011 = 时钟来自 LIRC. 111 = 时钟来自 HIRC. 其它 = 保留 <b>注:</b> 该位写保护, 参看SYS_REGLCTL 寄存器</p>
--	--	---

**CLK CLKSEL1 时钟源选择控制寄存器 1**

在切换时钟前，两个时钟源（预选与新选）都必须打开。

寄存器	偏移量	R/W	描述	复位值
CLK_CLKSEL1	CLK_BA+0x14	R/W	时钟源选择控制寄存器1	0xBF77_7703

31	30	29	28	27	26	25	24
WWDTSEL		CLKOSEL		UART1SEL		UART0SEL	
23	22	21	20	19	18	17	16
Reserved	TMR3SEL		Reserved	TMR2SEL		TMR1SEL	
15	14	13	12	11	10	9	8
Reserved	TMR0SEL		Reserved	WDTSEL		Reserved	
7	6	5	4	3	2	1	0
Reserved				WDTSEL			

位	描述
[31:30]	<b>WWDTSEL</b> 窗口看门狗定时器时钟源选择 10 = 时钟源为HCLK/2048. 11 = 时钟源为10 kHz 内部低速振荡器时钟(LIRC). 其它 = 保留.
[29:28]	<b>CLKOSEL</b> 时钟输出分频器时钟源选择 00 = 时钟源为外部 4~24 MHz 高速晶振时钟(HXT) 01 = 时钟源为外部 32.768 kHz 低速晶振时钟(LXT) 10 = 时钟源为 HCLK. 11 = 时钟源为内部 12 MHz 高速振荡器时钟(HIRC).
[27:26]	<b>UART1SEL</b> UART1 时钟源选择 00 = 时钟源为外部 4~24 MHz 高速晶振时钟(HXT). 01 = 时钟源为 PLL. 10 = 时钟源为外部 32.768 kHz 低速晶振时钟(LXT) 11 = 时钟源为内部 12 MHz 高速振荡器时钟(HIRC)
[25:24]	<b>UART0SEL</b> UART0 时钟源选择 00 = 时钟源为外部 4~24 MHz 高速晶振时钟(HXT). 01 = 时钟源为 PLL. 10 = 时钟源为外部 32.768 kHz 低速晶振时钟(LXT) 11 = 时钟源为内部 12 MHz 高速振荡器时钟(HIRC)
[23]	<b>Reserved</b> 保留
[22:20]	<b>TMR3SEL</b> TIMER3 时钟源选择 000 = 时钟源为外部 4~24 MHz 高速 晶振时钟(HXT) 001 = 时钟源为外部 32.768 kHz 低速晶振时钟(LXT)

		010 = 时钟源为 PCLK1.. 011 = 时钟源为外部时钟 TM3 管脚. 101 = 时钟源为10 kHz 内部低速振荡器时钟(LIRC) 111 = 时钟源为内部 12 MHz 高速振荡器时钟(HIRC) 其它 = 保留
[19]	<b>Reserved</b>	保留
[18:16]	<b>TMR2SEL</b>	<b>TIMER2 时钟源选择</b> 000 = 时钟源为外部 4~24 MHz高速 晶振时钟(HXT). 001 = 时钟源为外部 32.768 kHz 低速晶振时钟(LXT) 010 = 时钟源为 PCLK1 011 = 时钟源为外部时钟 TM2 管脚. 101 = 时钟源为10 kHz 内部低速振荡器时钟(LIRC) 111 = 时钟源为内部 12 MHz 高速振荡器时钟(HIRC) 其它 = 保留.
[15]	<b>Reserved</b>	保留
[14:12]	<b>TMR1SEL</b>	<b>TIMER1 时钟源选择</b> 000 = 时钟源为外部 4~24 MHz高速 晶振时钟(HXT). 001 = 时钟源为外部 32.768 kHz 低速晶振时钟(LXT) 010 = 时钟源为 PCLK0 011 = 时钟源为外部时钟 TM1 管脚. 101 = 时钟源为10 kHz 内部低速振荡器时钟(LIRC). 111 = 时钟源为内部 12 MHz 高速振荡器时钟(HIRC). 其它 = 保留
[11]	<b>Reserved</b>	保留
[10:8]	<b>TMR0SEL</b>	<b>TIMER0 时钟源选择</b> 000 = 时钟源为外部 4~24 MHz高速 晶振时钟(HXT) 001 = 时钟源为外部 32.768 kHz 低速晶振时钟(LXT). 010 = 时钟源为 PCLK0.. 011 = 时钟源为外部时钟 TM0 管脚. 101 = 时钟源为10 kHz 内部低速振荡器时钟(LIRC) 111 = 时钟源为内部 12 MHz 高速振荡器时钟(HIRC). 其它 = 保留
[7:2]	<b>Reserved</b>	保留
[1:0]	<b>WDTSEL</b>	<b>看门狗定时器时钟源选择(写保护)</b> 00 = 保留. 01 = 时钟源为外部 32.768 kHz 低速晶振时钟(LXT) 10 = 时钟源为 HCLK/2048. 11 = 时钟源为10 kHz 内部低速振荡器时钟(LIRC) <b>注意:</b> 该位写保护, 参看SYS_REGLCTL寄存器.

**CLK CLKSEL2 时钟源选择控制寄存器 2**

在切换时钟前，两个时钟源（预选与新选）都必须打开。

寄存器	偏移量	R/W	描述	复位值
CLK_CLKSEL2	CLK_BA+0x18	R/W	时钟源选择控制寄存器 2	0x0000_03AB

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved		SPI3SEL		SPI2SEL		BPWM1SEL	BPWM0SEL
7	6	5	4	3	2	1	0
SPI1SEL		SPI0SEL		QSPI0SEL		EPWM1SEL	EPWM0SEL

位	描述	
[31:14]	Reserved	保留
[13:12]	SPI3SEL	<b>SPI3 时钟源选择</b> 00 = 时钟源为外部 4~24 MHz 高速晶振时钟(HXT) 01 = 时钟源为 PLL. 10 = 时钟源为 PCLK0. 11 = 时钟源为内部 12 MHz 高速振荡器时钟(HIRC)
[11:10]	SPI2SEL	<b>SPI2 时钟源选择</b> 00 = 时钟源为外部 4~24 MHz 高速晶振时钟(HXT). 01 = 时钟源为 PLL. 10 = 时钟源为 PCLK1 11 = 时钟源为内部 12 MHz 高速振荡器时钟(HIRC)
[9]	BPWM1SEL	<b>BPWM1 时钟源选择</b> BPWM1 的外设时钟源由BPWM1SEL.来定义. 0 = 时钟源为PLL 1 = 时钟源为PCLK1.
[8]	BPWM0SEL	<b>BPWM0 时钟源选择</b> BPWM0 的外设时钟源由BPWM0SEL.来定义 0 = 时钟源为PLL. 1 = 时钟源为PCLK0.
[7:6]	SPI1SEL	<b>SPI1 时钟源选择</b> 00 = 时钟源为外部 4~24 MHz 高速晶振时钟(HXT) 01 = 时钟源为 PLL. 10 = 时钟源为 PCLK0

		11 = 时钟源为内部 12 MHz 高速振荡器时钟(HIRC).
[5:4]	<b>SPI0SEL</b>	<b>SPI0 时钟源选择</b> 00 = 时钟源为外部 4~24 MHz 高速晶振时钟(HXT). 01 = 时钟源为 PLL 10 = 时钟源为 PCLK1 11 = 时钟源为内部 12 MHz 高速振荡器时钟(HIRC).
[3:2]	<b>QSPI0SEL</b>	<b>QSPI0 时钟源选择</b> 00 = 时钟源为外部 4~24 MHz 高速晶振时钟(HXT) 01 = 时钟源为 PLL. 10 = 时钟源为 PCLK0. 11 = 时钟源为内部 12 MHz 高速振荡器时钟(HIRC)
[1]	<b>EPWM1SEL</b>	<b>EPWM1 时钟源选择</b> EPWM1 的外设时钟源由EPWM1SEL.来定义. 0 = 时钟源为PLL. 1 = 时钟源为PCLK1.
[0]	<b>EPWM0SEL</b>	<b>EPWM0 时钟源选择</b> EPWM0 的外设时钟源由EPWM0SEL.来定义. 0 = 时钟源为PLL. 1 = 时钟源为PCLK0.

**CLK\_CLKSEL3 时钟源选择控制寄存器3**

在时钟切换之前，必须打开相关的时钟源（预选和新选）。

寄存器	偏移量	R/W	描述	复位值
CLK_CLKSEL3	CLK_BA+0x1C	R/W	时钟源选择控制寄存器 3	0xFF00_003F

31	30	29	28	27	26	25	24
UART5SEL		UART4SEL		UART3SEL		UART2SEL	
23	22	21	20	19	18	17	16
Reserved						I2S0SEL	
15	14	13	12	11	10	9	8
Reserved							RTCSEL
7	6	5	4	3	2	1	0
Reserved		SC2SEL		SC1SEL		SC0SEL	

位	描述
[31:30]	<b>UART5SEL</b> <b>UART5 时钟源选择</b> 00 = 时钟源为外部 4~24 MHz 高速晶振时钟(HXT). 01 = 时钟源为 PLL. 10 = 时钟源为外部 32.768 kHz 低速晶振时钟(LXT) 11 = 时钟源为内部 12 MHz 高速振荡器时钟(HIRC).
[29:28]	<b>UART4SEL</b> <b>UART4 时钟源选择</b> 00 = 时钟源为外部 4~24 MHz 高速晶振时钟(HXT). 01 = 时钟源为 PLL. 10 = 时钟源为外部 32.768 kHz 低速晶振时钟(LXT). 11 = 时钟源为内部 12 MHz 高速振荡器时钟(HIRC).
[27:26]	<b>UART3SEL</b> <b>UART3 时钟源选择</b> 00 = 时钟源为外部 4~24 MHz 高速晶振时钟(HXT) 01 = 时钟源为 PLL. 10 = 时钟源为外部 32.768 kHz 低速晶振时钟(LXT) 11 = 时钟源为内部 12 MHz 高速振荡器时钟(HIRC).
[25:24]	<b>UART2SEL</b> <b>UART2 时钟源选择</b> 00 = 时钟源为外部 4~24 MHz 高速晶振时钟(HXT) 01 = 时钟源为 PLL. 10 = 时钟源为外部 32.768 kHz 低速晶振时钟(LXT). 11 = 时钟源为内部 12 MHz 高速振荡器时钟(HIRC).
[23:18]	<b>Reserved</b> 保留
[17:16]	<b>I2S0SEL</b> <b>I<sup>2</sup>S0 时钟源选择</b> 00 = 时钟源为外部 4~24 MHz 高速晶振时钟(HXT).

		01 = 时钟源为 PLL. 10 = 时钟源为 PCLK0 11 = 时钟源为内部 12 MHz 高速振荡器时钟(HIRC).
[15:9]	<b>Reserved</b>	保留
[8]	<b>RTCSEL</b>	<b>RTC 时钟源选择</b> 0 = 时钟源为外部 32.768 kHz 低速晶振时钟(LXT) 1 = 时钟源为内部 10 kHz 低速振荡器时钟 (LIRC).
[7:6]	<b>Reserved</b>	保留
[5:4]	<b>SC2SEL</b>	<b>SC2 时钟源选择</b> 00 = 时钟源为外部 4~24 MHz 高速晶振时钟(HXT).. 01 = 时钟源为 PLL. 10 = 时钟源为 PCLK0 11 = 时钟源为内部 12 MHz 高速振荡器时钟(HIRC).
[3:2]	<b>SC1SEL</b>	<b>SC0 时钟源选择</b> 00 = 时钟源为外部 4~24 MHz 高速晶振时钟(HXT).. 01 = 时钟源为 PLL 10 = 时钟源为 PCLK1. 11 = 时钟源为内部 12 MHz 高速振荡器时钟(HIRC).
[1:0]	<b>SC0SEL</b>	<b>SC0 时钟源选择</b> 00 = 时钟源为外部 4~24 MHz 高速晶振时钟(HXT).. 01 = 时钟源为 PLL 10 = 时钟源为 PCLK0 11 = 时钟源为内部 12 MHz 高速振荡器时钟(HIRC).

**CLK CLKDIV0 时钟分频数寄存器 0**

寄存器	偏移量	R/W	描述	复位值
CLK_CLKDIV0	CLK_BA+0x20	R/W	时钟分频数寄存器 0	0x0000_0000

31	30	29	28	27	26	25	24
SDH0DIV							
23	22	21	20	19	18	17	16
EADCDIV							
15	14	13	12	11	10	9	8
UART1DIV				UART0DIV			
7	6	5	4	3	2	1	0
USBDIV				HCLKDIV			

位	描述	
[31:24]	<b>SDH0DIV</b>	<b>SD0 时钟除频数, 时钟源来自 SD0 时钟源</b> SD0 时钟频率 = (SD0 时钟源频率) / (SDH0DIV + 1).
[23:16]	<b>EADCDIV</b>	<b>EADC 时钟除频数, 时钟源来自 EADC 时钟源</b> EADC 时钟频率 = (EADC 时钟源频率) / (EADCDIV + 1).
[15:12]	<b>UART1DIV</b>	<b>UART1 时钟除频数, 时钟源来自 UART1 时钟源</b> UART1 时钟频率 = (UART1 时钟源频率) / (UART1DIV + 1).
[11:8]	<b>UART0DIV</b>	<b>UART0 时钟除频数, 时钟源来自 UART0 时钟源</b> UART0 时钟频率 = (UART0 时钟源频率) / (UART0DIV + 1).
[7:4]	<b>USBDIV</b>	<b>USB 时钟除频数, 时钟源来自 PLL 时钟</b> USB 时钟频率 = (PLL 频率) / (USBDIV + 1).
[3:0]	<b>HCLKDIV</b>	<b>HCLK 时钟除频数, 时钟源来自 HCLK 时钟源</b> HCLK 时钟频率 = (HCLK 时钟源频率) / (HCLKDIV + 1).

CLK CLKDIV1 时钟分频数寄存器1

寄存器	偏移量	R/W	描述	复位值
CLK_CLKDIV1	CLK_BA+0x24	R/W	时钟分频数寄存器1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
SC2DIV							
15	14	13	12	11	10	9	8
SC1DIV							
7	6	5	4	3	2	1	0
SC0DIV							

位	描述	
[31:24]	Reserved	保留.
[23:16]	SC2DIV	SC2时钟的除频数, 时钟源来自SC2时钟源 SC2 时钟频率= (SC2时钟源频率) / (SC2DIV + 1).
[15:8]	SC1DIV	SC1时钟的除频数, 时钟源来自SC1时钟源 SC1时钟频率= (SC1时钟源频率) / (SC1DIV + 1).
[7:0]	SC0DIV	SC0时钟的除频数, 时钟源来自SC0时钟源 SC0时钟频率= (SC0时钟源频率) / (SC0DIV + 1).

CLK\_CLKDIV3 时钟分频数寄存器 3

寄存器	偏移量	R/W	描述	复位值
CLK_CLKDIV3	CLK_BA+0x2C	R/W	时钟分频数寄存器3	0x0000_0000

31	30	29	28	27	26	25	24
SDH1DIV							
23	22	21	20	19	18	17	16
EMACDIV							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved							

位	描述	
[31:24]	<b>SDH1DIV</b>	SC1时钟的除频数，时钟源来自SC1时钟源 SD1 时钟频率= (SD1时钟源频率) / (SDH1DIV + 1).
[23:16]	<b>EMACDIV</b>	以太网时钟的除频数，时钟源来自HCLK EMAC MDCLK 时钟频率 = (HCLK) / (EMACDIV + 1).
[15:0]	<b>Reserved</b>	保留.

CLK\_CLKDIV4 时钟分频数寄存器4

寄存器	偏移量	R/W	描述	复位值
CLK_CLKDIV4	CLK_BA+0x30	R/W	时钟分频数寄存器 4	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
UART5DIV				UART4DIV			
7	6	5	4	3	2	1	0
UART3DIV				UART2DIV			

位	描述	
[31:16]	Reserved	保留
[15:12]	UART5DIV	UART5 时钟除频数, 时钟源来自UART5时钟源 UART5 时钟频率= (UART5 时钟源频率) / (UART5DIV + 1).
[11:8]	UART4DIV	UART4 时钟除频数, 时钟源来自UART4时钟源 UART4 时钟频率= (UART4 时钟源频率) / (UART4DIV + 1).
[7:4]	UART3DIV	UART3 时钟除频数, 时钟源来自UART3时钟源 UART3 时钟频率= (UART3 时钟源频率) / (UART3DIV + 1).
[3:0]	UART2DIV	UART2 时钟除频数, 时钟源来自UART2时钟源 UART2 时钟频率= (UART2 时钟源频率) / (UART2DIV + 1).

CLK\_PCLKDIV APB 时钟分频器寄存器

寄存器	偏移量	R/W	描述	复位值
CLK_PCLKDIV	CLK_BA+0x34	R/W	APB 时钟分频器寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved	APB1DIV			Reserved	APB0DIV		

位	描述	
[31:7]	Reserved	保留.
[6:4]	APB1DIV	<b>APB1 时钟分频器</b> APB1 时钟可以被分频，时钟源来自 HCLK。 000: PCLK1 = HCLK. 001: PCLK1 = 1/2 HCLK. 010: PCLK1 = 1/4 HCLK. 011: PCLK1 = 1/8 HCLK. 100: PCLK1 = 1/16 HCLK. 其他:保留.
[3]	Reserved	保留.
[2:0]	APB0DIV	<b>APB0 时钟分频器</b> APB0 时钟可以被分频，时钟源来自 HCLK。 000: PCLK0 = HCLK. 001: PCLK0 = 1/2 HCLK. 010: PCLK0 = 1/4 HCLK. 011: PCLK0 = 1/8 HCLK. 100: PCLK0 = 1/16 HCLK. 其他:保留.

CLK\_PLLCTL PLL 控制寄存器

寄存器	偏移量	R/W	描述	复位值
CLK_PLLCTL	CLK_BA+0x40	R/W	PLL控制寄存器	0x0005_C02E

31	30	29	28	27	26	25	24
Reserved			BANDSEL	Reserved			
23	22	21	20	19	18	17	16
STBSEL	Reserved			PLLSRC	OE	BP	PD
15	14	13	12	11	10	9	8
OUTDIV		INDIV					FBDIV
7	6	5	4	3	2	1	0
FBDIV							

位	描述	
[31:29]	Reserved	保留
[28]	BANDSEL	<b>PLL 频带选择 (写保护)</b> 0 = PLL低频带选择 (FVCO 范围是 200MHz ~ 400MHz) 1 = PLL高频带选择. (FVCO范围是400MHz ~ 500MHz) <b>注意:</b> 该位写保护, 参看SYS_REGLCTL寄存器
[23]	STBSEL	<b>PLL稳定计数个数选择 (写保护)</b> 0 = PLL 稳定时间为6144个 PLL 时钟源周期(适合时钟源频率等于或小于12MHz).. 1 = PLL 稳定时间为12288个PLL 时钟源周期 (适合时钟源频率大于12 MHz). <b>注意:</b> 该位写保护, 参看SYS_REGLCTL寄存器。
[22:20]	Reserved	保留.
[19]	PLLSRC	<b>PLL 时钟源选择(写保护)</b> 0 = PLL 时钟源为外部 4~24 MHz 高速晶振(HXT) 1 = PLL 时钟源为内部12 MHz 高速振荡器(HIRC) <b>注意:</b> 该位写保护, 参看SYS_REGLCTL寄存器。
[18]	OE	<b>PLL OE (FOUT 使能) 管脚控制 (写保护)</b> 0 = PLL FOUT 使能 1 = PLL FOUT 固定为低 <b>注意:</b> 该位写保护, 参看SYS_REGLCTL寄存器。
[17]	BP	<b>PLL 旁路控制 (写保护)</b> 0 = PLL 正常模式 (默认) 1 = PLL 时钟输出与PLL时钟源(FIN)输入相同 <b>注意:</b> 该位写保护, 参看SYS_REGLCTL寄存器。.
[16]	PD	掉电模式 (写保护)

		如果设置寄存器CLK_PWRCTL的PDEN位为1，PLL将进入掉电模式。 0 = PLL 正常模式 1 = PLL 进入掉电模式（默认） <b>注意：</b> 该位写保护，参看SYS_REGLCTL寄存器。
[15:14]	OUTDIV	<b>PLL 输出分频控制(写保护)</b> 参考下表公式 <b>注意：</b> 该位写保护，参看SYS_REGLCTL寄存器。
[13:9]	INDIV	<b>PLL 输入分频控制(写保护)</b> 参考下表公式 <b>注意：</b> 该位写保护，参看SYS_REGLCTL寄存器。.
[8:0]	FBDIV	<b>PLL 反馈分频控制(写保护)</b> 参考下表公式 <b>注意：</b> 该位写保护，参看SYS_REGLCTL寄存器。

输出时钟频率公式：

$$FREF = FIN \times \frac{1}{NR}$$

$$FVCO = FIN \times \frac{2 * NF}{NR}$$

$$FOUT = FIN \times \frac{2 * NF}{NR} \times \frac{1}{NO}$$

这里的 FREF是用于PFD（相位频率侦测器）的比较频率.

为了PLL在正常模式更好的工作，下面约束条件必须满足：

$$4 \text{ MHz} \leq FREF \leq 8 \text{ MHz}$$

$$200 \text{ MHz} \leq FVCO \leq 500 \text{ MHz}$$

$$50 \text{ MHz} \leq FOUT \leq 500 \text{ MHz}$$

符号	描述
FOUT	输出时钟频率
FIN	输入（参考）时钟频率
NR	输入分频 (INDIV + 1)
NF	反馈分频 (FBDIV + 2)
NO	OUTDIV = "00" OUTDIV = "01" OUTDIV = "10" OUTDIV = "11": NO = 4
	: NO = 1 : NO = 2 : NO = 2

表 6.3-1 PLL 输出频率公式的符号定义



**CLK\_STATUS 时钟状态监测寄存器**

该寄存器各位用于监控芯片时钟源是否稳定，时钟切换是否失败。

寄存器	偏移量	R/W	描述	复位值
CLK_STATUS	CLK_BA+0x50	R	时钟状态监测寄存器	0x0000_00XX

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
CLKSFAIL	Reserved	PLL2STB	HIRCSTB	LIRCSTB	PLLSTB	LXTSTB	HXTSTB

位	描述
[31:8]	Reserved 保留.
[7]	<b>CLKSFAIL</b> 时钟切换失败标志（只读） 当用户切换系统时钟源时，该位被更新。如果切换目标时钟稳定，该位将被设置为0。如果切换目标时钟不稳定，该位将被设置为1。 0 = 时钟切换成功 1 = 时钟切换失败 <b>注意：</b> 往该位写1，会清除该位到0
[6]	Reserved 保留.
[5]	<b>PLL2STB</b> 内部 PLL2 时钟源稳定标志（只读） 0 = 内部 PLL2 时钟不稳定或禁止. 1 = 内部 PLL2 时钟稳定. <b>注意：</b> 该位只读.
[4]	<b>HIRCSTB</b> HIRC时钟源稳定标志（只读） 0 = 12 MHz 内部高速RC 振荡器(HIRC) 时钟不稳定或者禁用 1 = 12 MHz 内部高速RC 振荡器(HIRC) 时钟使能并稳定
[3]	<b>LIRCSTB</b> LIRC 时钟源稳定标志（只读） 0 = 内部10 kHz低速RC振荡器 (LIRC) 时钟不稳定或者禁用 1 = 内部10 kHz低速RC振荡器(LIRC) 时钟使能并稳定
[2]	<b>PLLSTB</b> 内部 PLL 时钟源稳定标志(只读) 0 = 内部 PLL时钟不稳定或者禁用 1 = 内部PLL 时钟使能并稳定
[1]	<b>LXTSTB</b> LXT 时钟源稳定标志（只读）

		0 = 外部32.768 kHz低速晶振(LXT) 时钟不稳定或者禁用 1 = 外部32.768 kHz低速晶振(LXT) 时钟使能并稳定
[0]	HXTSTB	<b>HXT 时钟源稳定标志 (只读)</b> 0 = 外部4~24 MHz高速晶振(HXT) 时钟不稳定或者禁用 1 = 外部4~24 MHz高速晶振(HXT) 时钟使能并稳定

CLK\_CLKOCTL 时钟输出控制寄存器

寄存器	偏移量	R/W	描述	复位值
CLK_CLKOCTL	CLK_BA+0x60	R/W	时钟输出控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved	CLK1HZEN	DIV1EN	CLKOEN	FREQSEL			

位	描述	
[31:7]	Reserved	保留.
[6]	CLK1HZEN	时钟输出 1Hz 使能位 0 = 禁止32.768 kHz频率补偿输出 1Hz 时钟。 1 = 使能32.768 kHz频率补偿输出 1Hz 时钟。
[5]	DIV1EN	时钟输出分频 1使能位 0 = 时钟输出的时钟来自被FREQSEL分频的时钟源 1 = 时钟输出的时钟来自时钟源
[4]	CLKOEN	时钟输出使能位 0 = 禁用时钟输出功能 1 = 使能时钟输出功能
[3:0]	FREQSEL	时钟输出频率选择位 输出频率的公式: $F_{out} = F_{in}/2^{(N+1)}$ . $F_{in}$ 为输入时钟频率 $F_{out}$ 为分频器输出时钟频率 N为FREQSEL[3:0] 的4位值。

CLK CLKDCTL 时钟失败侦测器控制寄存器

寄存器	偏移量	R/W	描述	复位值
CLK_CLKDCTL_L	CLK_BA+0x70	R/W	时钟失败检测器控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved						HXTFQIEN	HXTFQDEN
15	14	13	12	11	10	9	8
Reserved		LXTFIEN	LXTFDEN	Reserved			
7	6	5	4	3	2	1	0
Reserved		HXTFIEN	HXTFDEN	Reserved			

位	描述	
[31:18]	<b>Reserved</b>	保留.
[17]	<b>HXTFQIEN</b>	<b>HXT 时钟频率范围监测中断使能位</b> 0 = 禁止外部4~24 MHz高速晶振(HXT) 时钟频率范围监测失败中断 1 = 使能外部4~24 MHz高速晶振(HXT) 时钟频率范围监测失败中断
[16]	<b>HXTFQDEN</b>	<b>HXT 时钟频率监测使能位</b> 0 = 禁止外部4~24 MHz高速晶振(HXT)时钟频率范围监测. 1 = 使能外部4~24 MHz高速晶振(HXT)时钟频率范围监测
[15:14]	<b>Reserved</b>	保留.
[13]	<b>LXTFIEN</b>	<b>LXT 时钟失败中断使能位</b> 0 = 禁止外部32.768 kHz低速晶振(LXT)时钟失败中断 1 = 使能外部32.768 kHz低速晶振(LXT)时钟失败中断.
[12]	<b>LXTFDEN</b>	<b>LXT 时钟失败检测器使能位</b> 0 = 禁止外部32.768 kHz低速晶振(LXT)时钟失败检测器. 1 = 使能外部32.768 kHz低速晶振(LXT)时钟失败检测器.
[11:6]	<b>Reserved</b>	保留.
[5]	<b>HXTFIEN</b>	<b>HXT 时钟失败中断使能位</b> 0 = 禁止外部4~24 MHz高速晶振(HXT)时钟失败中断 1 = 使能外部4~24 MHz高速晶振(HXT)时钟失败中断.
[4]	<b>HXTFDEN</b>	<b>HXT 时钟失败检测器使能位</b> 0 = 禁止外部4~24 MHz高速晶振(HXT)时钟失败检测器. 1 = 使能外部4~24 MHz高速晶振(HXT)时钟失败检测器
[3:0]	<b>Reserved</b>	保留

CLK\_CLKDSTS 时钟失败侦测器状态寄存器

寄存器	偏移量	R/W	描述	复位值
<b>CLK_CLKDSTS</b>	CLK_BA+0x74	R/W	时钟失败侦测器状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved						LXTFIF	HXTFIF

位	描述	
[31:9]	<b>Reserved</b>	保留.
[8]	<b>HXTFQIF</b>	<b>HXT 时钟频率范围监测中断标志位</b> 0 = 外部4~24 MHz高速晶振(HXT)时钟正常 1 = 外部4~24 MHz高速晶振(HXT)时钟频率异常 <b>注意:</b> 该位写1清除.
[7:2]	<b>Reserved</b>	保留.
[1]	<b>LXTFIF</b>	<b>LXT 时钟失败中断标志位</b> 0 = 外部32.768 kHz低速晶振(LXT)时钟正常 1 = 外部32.768 kHz低速晶振(LXT)时钟停止 <b>注意:</b> 该位写1清除
[0]	<b>HXTFIF</b>	<b>HXT 时钟失败中断标志位</b> 0 = 外部4~24 MHz高速晶振(HXT)时钟正常 1 = 外部4~24 MHz高速晶振(HXT)时钟停止 <b>注意:</b> 该位写1清除

CLK\_CDUPB 时钟频率范围检测器上边界寄存器

寄存器	偏移量	R/W	描述	复位值
CLK_CDUPB	CLK_BA+0x78	R/W	时钟频率检测器上边界寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved						UPERBD	
7	6	5	4	3	2	1	0
UPERBD							

位	描述	
[31:10]	Reserved	保留.
[9:0]	UPERBD	<b>HXT 时钟频率范围检测器上边界值</b> 该位定义频率范围监测窗口的最大值 当HXT频率监测值高于该寄存器的值时， HXT频率范围检测器中断标志将置1.

**CLK\_CDLOWB** 时钟频率范围检测器下边界寄存器

寄存器	偏移量	R/W	描述	复位值
CLK_CDLOWB	CLK_BA+0x7c	R/W	时钟频率范围检测器下边界寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved						LOWERBD	
7	6	5	4	3	2	1	0
LOWERBD							

位	描述	
[31:10]	Reserved	保留.
[9:0]	LOWERBD	<b>HXT</b> 时钟频率范围检测器下边界 该位定义频率范围监测窗口的最小值 当HXT频率监测值低于该寄存器的值时， HXT频率范围检测器中断标志将置1.

**CLK\_PMUCTL 电源管理控制寄存器**

寄存器	偏移量	R/W	描述	复位值
CLK_PMUCTL	CLK_BA+0x90	R/W	电源管理控制寄存器	0x0000_0080

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
RTCWKEN	Reserved				ACMPSPWK	WKPINEN	
15	14	13	12	11	10	9	8
Reserved				WKTMRIS			WKTMRREN
7	6	5	4	3	2	1	0
Reserved				PDMSEL			

位	描述	
[31:24]	Reserved	保留.
[23]	RTCWKEN	<b>RTC 唤醒使能 (写保护)</b> 0 = 禁止RTC在深度睡眠模式或空闲模式唤醒系统 1 = 使能RTC在深度睡眠模式或空闲模式唤醒系统 <b>注:</b> 该位是写保护位。参考SYS_REGLCTL寄存器
[22:19]	Reserved	保留.
[18]	ACMPSPWK	<b>ACMP 待机掉电模式唤醒使能 (写保护)</b> 0 = 禁止ACMP在待机掉电模式唤醒系统 1 = 使能ACMP在待机掉电模式唤醒系统 <b>注:</b> 该位是写保护位。参考SYS_REGLCTL寄存器
[17:16]	WKPINEN	<b>引脚唤醒使能 (写保护)</b> 00 = 禁止引脚在深度睡眠模式唤醒系统 01 = 使能引脚上升沿在深度睡眠模式唤醒系统. 10 = 使能引脚下降沿在深度睡眠模式唤醒系统.. 11 = 使能引脚下降沿和上升沿都可以在深度睡眠模式唤醒系统 <b>注:</b> 该位是写保护位。参考SYS_REGLCTL寄存器
[15:12]	Reserved	保留.
[11:9]	WKTMRIS	<b>唤醒定时器超时间隔选择 (写保护)</b> 当芯片在DPD/SPD模式, 这些位控制唤醒定时器超时间隔。 000 = 超时间隔是128个 OSC10K 时钟 (12.8 ms). 001 = 超时间隔是256个 OSC10K 时钟(25.6 ms). 010 = 超时间隔是512个 OSC10K 时钟 (51.2 ms). 011 = 超时间隔是1024个 OSC10K 时钟 (102.4ms)

		<p>100 =超时间隔是4096个 OSC10K 时钟 (409.6ms).      101 =超时间隔是8192个 OSC10K 时钟(819.2ms).      110 =超时间隔是16384个 OSC10K 时钟 (1638.4ms).      111 =超时间隔是65536个 OSC10K 时钟 (6553.6ms).  <b>注:</b> 该位是写保护位。参考SYS_REGLCTL寄存器</p>
[8]	<b>WKTMRREN</b>	<p><b>唤醒定时器使能 (写保护)</b>      0 = 在 DPD/SPD 模式, 唤醒定时器禁止.      1 = 在 DPD/SPD 模式, 唤醒定时器使能.  <b>注:</b> 该位是写保护位。参考SYS_REGLCTL寄存器.</p>
[7:3]	<b>Reserved</b>	保留.
[2:0]	<b>PDMSEL</b>	<p><b>掉电模式选择 (写保护)</b>      这些位控制芯片当CPU执行WFI/WFE指令时的掉电模式等级选择。      000 = 选择掉电模式. (NPD)      001 = 选择低漏电掉电模式 (LLPD).      010 =选择快速唤醒掉电模式 (FWPD).      011 = 保留.      100 = 选择待机掉电模式0 (SPD0) (SRAM 保持).      101 =选择待机掉电模式1 (SPD1).      110 = 选择深度掉电模式 (DPD).      111 = 保留.  <b>注:</b> 该位是写保护位。参考SYS_REGLCTL寄存器.</p>

## CLK\_P MUSTS 电源管理状态寄存器

寄存器	偏移量	R/W	描述	复位值
CLK_P MUSTS	CLK_BA+0x94	R/W	电源管理状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
CLRWK	Reserved						
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved	ACMPWK	BODWK	LVRWK	GPDWK	GPCWK	GPBWK	GPAWK
7	6	5	4	3	2	1	0
Reserved					RTCWK	TMRWK	PINWK

位	描述	
[31]	CLRWK	<b>清唤醒标志</b> 0 = 不清标志。 1= 清除所有唤醒标志
[30:15]	Reserved	保留
[14]	ACMPWK	<b>ACMP 唤醒标志 (只读)</b> 该标志表示设备被一个ACMP转变请求从待机掉电模式（SPD）唤醒。当进入SPD模式，该标志被清除。
[13]	BODWK	<b>BOD唤醒标志 (只读)</b> 该标志表示设备被一个BOD发生请求从待机掉电模式（SPD）唤醒。当进入SPD模式，该标志被清除。
[12]	LVRWK	<b>LVR唤醒标志 (只读)</b> 该标志表示设备被一个LVR发生请求从待机掉电模式（SPD）唤醒。当进入SPD模式，该标志被清除。
[11]	GPDWK	<b>GPD唤醒标志 (只读)</b> 该标志表示设备被GPD组管脚的一个跳变请求从待机掉电模式（SPD）唤醒。当进入SPD模式，该标志被清除。
[10]	GPCWK	<b>GPC唤醒标志 (只读)</b> 该标志表示设备被GPC组管脚的一个跳变请求从待机掉电模式（SPD）唤醒。当进入SPD模式，该标志被清除。
[9]	GPBWK	<b>GPB唤醒标志 (只读)</b> 该标志表示设备被GPB组管脚的一个跳变请求从待机掉电模式（SPD）唤醒。当进入SPD模式，该标志被清除。
[8]	GPAWK	<b>GPA唤醒标志 (只读)</b> 该标志表示设备被GPA组管脚的一个跳变请求从待机掉电模式（SPD）唤醒。当进入SPD模式，该标志被清除。

[7:3]	<b>Reserved</b>	保留.
[2]	<b>RTCWK</b>	<b>RTC唤醒标志 (只读)</b> 该标志表示设备被RTC闹钟请求从待机掉电模式（SPD）唤醒。当进入SPD模式，该标志被清除。
[1]	<b>TMRWK</b>	<b>Timer唤醒标志 (只读)</b> 该标志表示设备被唤醒定时器超时请求从深度掉电模式(DPD)或待机掉电模式（SPD）唤醒。当进入DPD或SPD模式，该标志被清除。
[0]	<b>PINWK</b>	<b>Pin唤醒标志 (只读)</b> 该标志表示设备被唤醒管脚(GPC.0)的跳变请求从深度掉电模式(DPD)唤醒。当进入DPD模式，该标志被清除。

LDO Control Register (CLK\_LDOCTL)

寄存器	偏移量	R/W	描述	复位值
CLK_LDOCTL	CLK_BA+0x98	R/W	LDO状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved					PDBIASEN	Reserved	
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved							

Bits	Description	
[31:19]	Reserved	保留.
[18]	PDBIASEN	<p><b>Power-down Bias Enable Bit</b></p> <p>0 = 保留. 1 = Power-down bias Enabled.</p> <p><b>Note:</b> This bit should set to 1 before chip enter power-down mode.</p>
[17:0]	Reserved	保留.



CLK\_SWKDBCTL 待机掉电唤醒消抖控制寄存器

寄存器	偏移量	R/W	描述	复位值
CLK_SWKDBCTL	CLK_BA+0x9C	R/W	待机掉电唤醒消抖控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved				SWKDBCLKSEL			

位	描述	
[31:4]	Reserved	保留.
[3:0]	SWKDBCLKSEL	<p>待机掉电 唤醒消抖采样周期选择</p> <p>0000 = 每 1个时钟采样一次唤醒输入.      0001 = 每 2个时钟采样一次唤醒输入      0010 = 每 4个时钟采样一次唤醒输入.      0011 = 每 8个时钟采样一次唤醒输入.      0100 = 每 16个时钟采样一次唤醒输入.      0101 = 每 32个时钟采样一次唤醒输入.      0110 = 每 64个时钟采样一次唤醒输入.      0111 = 每 128个时钟采样一次唤醒输入.      1000 = 每 256个时钟采样一次唤醒输入.      1001 = 每2*256个时钟采样一次唤醒输入.      1010 = 每4*256个时钟采样一次唤醒输入.      1011 = 每8*256个时钟采样一次唤醒输入.      1100 = 每16*256个时钟采样一次唤醒输入.      1101 = 每32*256个时钟采样一次唤醒输入.      1110 = 每64*256个时钟采样一次唤醒输入.      1111 = 每128*256个时钟采样一次唤醒输入..</p> <p>注: 消抖计数器时钟源是内部 10 kHz 低速RC 振荡器 (LIRC).</p>

**CLK\_PASWKCTL GPA 待机掉电唤醒控制寄存器**

寄存器	偏移量	R/W	描述	复位值
CLK_PASWKCTL	CLK_BA+0xA0	R/W	GPA 待机掉电唤醒控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
WKPSEL				Reserved	PFWKEN	PRWKEN	WKEN

位	描述	
[31:9]	Reserved	保留.
[8]	DBEN	<p><b>GPA 输入信号消抖使能位</b>            DBEN位用来使能对应IO口消抖功能，如果输入信号脉冲宽度不能被连续的两个消抖采样周期采样到，输入信号的变化被认为是信号抖动，不会触发唤醒。消抖时钟源是内部 10 kHz 低速RC 振荡器 (LIRC)。            0 = 待机掉电唤醒引脚消抖功能禁止            1 = 待机掉电唤醒引脚消抖功能使能.            消抖功能仅对边沿触发有效</p>
[7:4]	WKPSEL	<p><b>GPA 待机掉电唤醒管脚选择</b>            0000 = GPA.0 唤醒功能使能.            0001 = GPA.1唤醒功能使能.            0010 = GPA.2唤醒功能使能.            0011 = GPA.3唤醒功能使能.            0100 = GPA.4唤醒功能使能.            0101 = GPA.5唤醒功能使能.            0110 = GPA.6唤醒功能使能.            0111 = GPA.7唤醒功能使能.            1000 = GPA.8唤醒功能使能.            1001 = GPA.9唤醒功能使能.            1010 = GPA.10唤醒功能使能.            1011 = GPA.11唤醒功能使能.            1100 = GPA.12唤醒功能使能.            1101 = GPA.13唤醒功能使能.            1110 = GPA.14唤醒功能使能.            1111 = GPA.15唤醒功能使能.</p>

[3]	<b>Reserved</b>	保留.
[2]	<b>PFWKEN</b>	管脚下降沿唤醒使能位 0 = GPA 组管脚下降沿唤醒功能禁止. 1 = GPA 组管脚下降沿唤醒功能使能
[1]	<b>PRWKEN</b>	管脚上升沿唤醒使能位 0 = GPA组管脚上升沿唤醒功能禁止 1 = GPA 组管脚上升沿唤醒功能使能
[0]	<b>WKEN</b>	待机掉电管脚唤醒使能位 0 = GPA组管脚唤醒功能禁止 1 = GPA 组管脚唤醒功能使能.

**CLK\_PBSWKCTL GPB待机掉电唤醒控制寄存器**

寄存器	偏移量	R/W	描述	复位值
CLK_PBSWKCTL	CLK_BA+0xA4	R/W	GPB待机掉电唤醒控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
WKPSEL				Reserved	PFWKEN	PRWKEN	WKEN

位	描述	
[31:9]	Reserved	保留.
[8]	DBEN	<p><b>GPB 输入信号消抖使能位</b>            DBEN位用来使能对应IO口消抖功能，如果输入信号脉冲宽度不能被连续的两个消抖采样周期采样到，输入信号的变化被认为是信号抖动，不会触发唤醒。消抖时钟源是内部 10 kHz 低速RC 振荡器 (LIRC)。            0 = 待机掉电唤醒引脚消抖功能禁止            1 = 待机掉电唤醒引脚消抖功能使能.            消抖功能仅对边沿触发有效</p>
[7:4]	WKPSEL	<p><b>GPB 待机掉电唤醒管脚选择</b>            0000 = GPB.0 唤醒功能使能.            0001 = GPB.1唤醒功能使能.            0010 = GPB.2唤醒功能使能.            0011 = GPB.3唤醒功能使能.            0100 = GPB.4唤醒功能使能.            0101 = GPB.5唤醒功能使能.            0110 = GPB.6唤醒功能使能.            0111 = GPB.7唤醒功能使能.            1000 = GPB.8唤醒功能使能.            1001 = GPB.9唤醒功能使能.            1010 = GPB.10唤醒功能使能.            1011 = GPB.11唤醒功能使能.            1100 = GPB.12唤醒功能使能.            1101 = GPB.13唤醒功能使能.            1110 = GPB.14唤醒功能使能.            1111 = GPB.15唤醒功能使能..</p>

[3]	<b>Reserved</b>	保留.
[2]	<b>PFWKEN</b>	管脚下降沿唤醒使能位 0 = GPB 组管脚下降沿唤醒功能禁止. 1 = GPB 组管脚下降沿唤醒功能使能..
[1]	<b>PRWKEN</b>	管脚上升沿唤醒使能位 0 = GPB 组管脚上升沿唤醒功能禁止 1 = GPB 组管脚上升沿唤醒功能使能
[0]	<b>WKEN</b>	待机掉电管脚唤醒使能位 0 = GPB 组管脚唤醒功能禁止 1 = GPB 组管脚唤醒功能使能.

**CLK\_PCSWKCTL GPC 待机掉电唤醒控制寄存器**

寄存器	偏移量	R/W	描述	复位值
CLK_PCSWKCTL	CLK_BA+0xA8	R/W	GPC 待机掉电唤醒控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
WKPSEL				Reserved	PFWKEN	PRWKEN	WKEN

位	描述	
[31:9]	Reserved	保留.
[8]	DBEN	<p><b>GPC 输入信号消抖使能位</b>            DBEN位用来使能对应IO口消抖功能，如果输入信号脉冲宽度不能被连续的两个消抖采样周期采样到，输入信号的变化被认为是信号抖动，不会触发唤醒。消抖时钟源是内部 10 kHz 低速RC 振荡器 (LIRC)。            0 = 待机掉电唤醒引脚消抖功能禁止.            1 = 待机掉电唤醒引脚消抖功能使能.            消抖功能仅对边沿触发有效.</p>
[7:4]	WKPSEL	<p><b>GPB 待机掉电唤醒管脚选择</b>            0000 = GPC.0 唤醒功能使能.            0001 = GPC.1 唤醒功能使能.            0010 = GPC.2 唤醒功能使能.            0011 = GPC.3 唤醒功能使能.            0100 = GPC.4 唤醒功能使能.            0101 = GPC.5 唤醒功能使能.            0110 = GPC.6 唤醒功能使能.            0111 = GPC.7 唤醒功能使能.            1000 = GPC.8 唤醒功能使能.            1001 = GPC.9 唤醒功能使能.            1010 = GPC.10 唤醒功能使能.            1011 = GPC.11 唤醒功能使能.            1100 = GPC.12 唤醒功能使能.            1101 = GPC.13 唤醒功能使能.            1110 = GPC.14 唤醒功能使能.            1111 = GPC.15 唤醒功能使能.</p>

[3]	<b>Reserved</b>	保留
[2]	<b>PFWKEN</b>	管脚下降沿唤醒使能位 0 = GPC 组管脚下降沿唤醒功能禁止. 1 = GPC组管脚下降沿唤醒功能使能.
[1]	<b>PRWKEN</b>	管脚上升沿唤醒使能位 0 = GPC组管脚上升沿唤醒功能禁止 1 = GPC 组管脚上升沿唤醒功能使能
[0]	<b>WKEN</b>	待机掉电管脚唤醒使能位 0 = GPC组管脚唤醒功能禁止 1 = GPC 组管脚唤醒功能使能

**CLK\_PDSWKCTL GPD 待机掉电唤醒控制寄存器**

寄存器	偏移量	R/W	描述	复位值
CLK_PDSWKCTL	CLK_BA+0xAC	R/W	GPD待机掉电唤醒控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
WKPSEL				Reserved	PFWKEN	PRWKEN	WKEN

位	描述	
[31:9]	Reserved	保留
[8]	DBEN	<p><b>GPD 输入信号消抖使能位</b>            DBEN位用来使能对应IO口消抖功能，如果输入信号脉冲宽度不能被连续的两个消抖采样周期采样到，输入信号的变化被认为是信号抖动，不会触发唤醒。消抖时钟源是内部 10 kHz 低速RC 振荡器 (LIRC)。            0 = 待机掉电唤醒引脚消抖功能禁止            1 = 待机掉电唤醒引脚消抖功能使能。            消抖功能仅对边沿触发有效。</p>
[7:4]	WKPSEL	<p><b>GPD 待机掉电唤醒管脚选择</b>            0000 = GPD.0 唤醒功能使能.            0001 = GPD.1 唤醒功能使能.            0010 = GPD.2 唤醒功能使能.            0011 = GPD.3 唤醒功能使能.            0100 = GPD.4 唤醒功能使能.            0101 = GPD.5 唤醒功能使能.            0110 = GPD.6 唤醒功能使能.            0111 = GPD.7 唤醒功能使能.            1000 = GPD.8 唤醒功能使能.            1001 = GPD.9 唤醒功能使能.            1010 = GPD.10 唤醒功能使能.            1011 = GPD.11 唤醒功能使能.            1100 = GPD.12 唤醒功能使能.            1101 = GPD.13 唤醒功能使能.            1110 = GPD.14 唤醒功能使能.            1111 = GPD.15 唤醒功能使能.</p>

[3]	<b>Reserved</b>	保留
[2]	<b>PFWKEN</b>	管脚下降沿唤醒使能位 0 = GPD 组管脚下降沿唤醒功能禁止. 1 = GPD组管脚下降沿唤醒功能使能.
[1]	<b>PRWKEN</b>	管脚上升沿唤醒使能位 0 = GPD组管脚上升沿唤醒功能禁止 1 = GPD 组管脚上升沿唤醒功能使能
[0]	<b>WKEN</b>	待机掉电管脚唤醒使能位 0 = GPD组管脚唤醒功能禁止 1 = GPD 组管脚唤醒功能使能.

CLK\_IOPDCTL GPIO 待机掉电控制寄存器

寄存器	偏移量	R/W	描述	复位值
CLK_IOPDCTL	CLK_BA+0xB0	R/W	GPIO 待机掉电控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved							IOHR

位	描述	
[31:1]	Reserved	保留.
[0]	IOHR	<b>GPIO 保持释放</b> 当GPIO进入待机掉电模式，所有I/O状态保持正常操作时的状态。在芯片从待机掉电模式唤醒后，I/O仍然保持状态直到用户设置该位释放I/O保持的状态。 该位由硬件自动清0

## 6.4 FMC 存储控制器

### 6.4.1 概述

FMC配备芯片内嵌的双bank flash(BANK0 和 BANK1)，用于存储应用程序和存储基于应用的一些数据，数据Flash可配置大小。BANK0 和 BANK1都有64K/128K/256K 字节空间，所以总的应用存储器(APROM) 大小是128K/256K/512K。在BANK0，一个用户配置区，用于系统初始化，一个4K字节的引导存储器(LDROM)，用于在系统编程(ISP)功能。在BANK1，一个4K字节的安全保护存储器(SPROM)可以隐藏用户的程序。一个3K字节的单次编程存储器(OTP) 用于存储需要单次编程的数据。一个32K的Boot Loader，内建ISP功能。一个零等待周期的4K高速缓存cache，用于提高对flash的访问性能。支持在应用编程(IAP)，更新flash程序后，执行引导程序和用户程序之间切换时，无需复位。

### 6.4.2 FMC特性

- 支持双bank flash用于安全固件升级
- 支持 128K/256K/512K字节应用程序存储空间(APROM).
- 支持4K字节引导存储器(LDROM).
- 支持双bank flash 中镜像SPROM，在写入其他ROM时读取其他SPROM代码
- 支持 4 K字节安全保护存储器(SPROM) 用于隐藏用户的程序.
- 支持可配置大小的数据Flash
- 支持16字节用户配置区，用于控制系统初始化
- 支持 3 K字节单次编程存储器 (OTP).
- 对所有内嵌的flash，支持 4 K字节页擦除 .
- Boot Loader内建了在系统编程(ISP)功能
- 支持安全启动功能用于代码存储和可靠性需求。
- 支持安全密匙保护功能，用于保护APROM, LDROM, SPROM, 用户配置区和KROM
- 支持 32-bit/64-bit 和多字 flash 编程功能.
- 支持快速flash编程校验功能.
- 支持 CRC32 校验和计算功能.
- 支持flash 位全为1校验功能
- 支持在系统编程(ISP) /在应用编程(IAP)来更新片上Flash
- 支持cache存储器来提高flash访问效率和减少功耗
- 支持自动调整flash访问周期功能来优化flash访问性能

### 6.4.3 框图

存储控制器(FMC)包括AHB从接口， cache， boot loader， flash 控制寄存器， flash初始化控制器， Flash操作控制和片上flash。 Flash存储控制器框图如下所示。

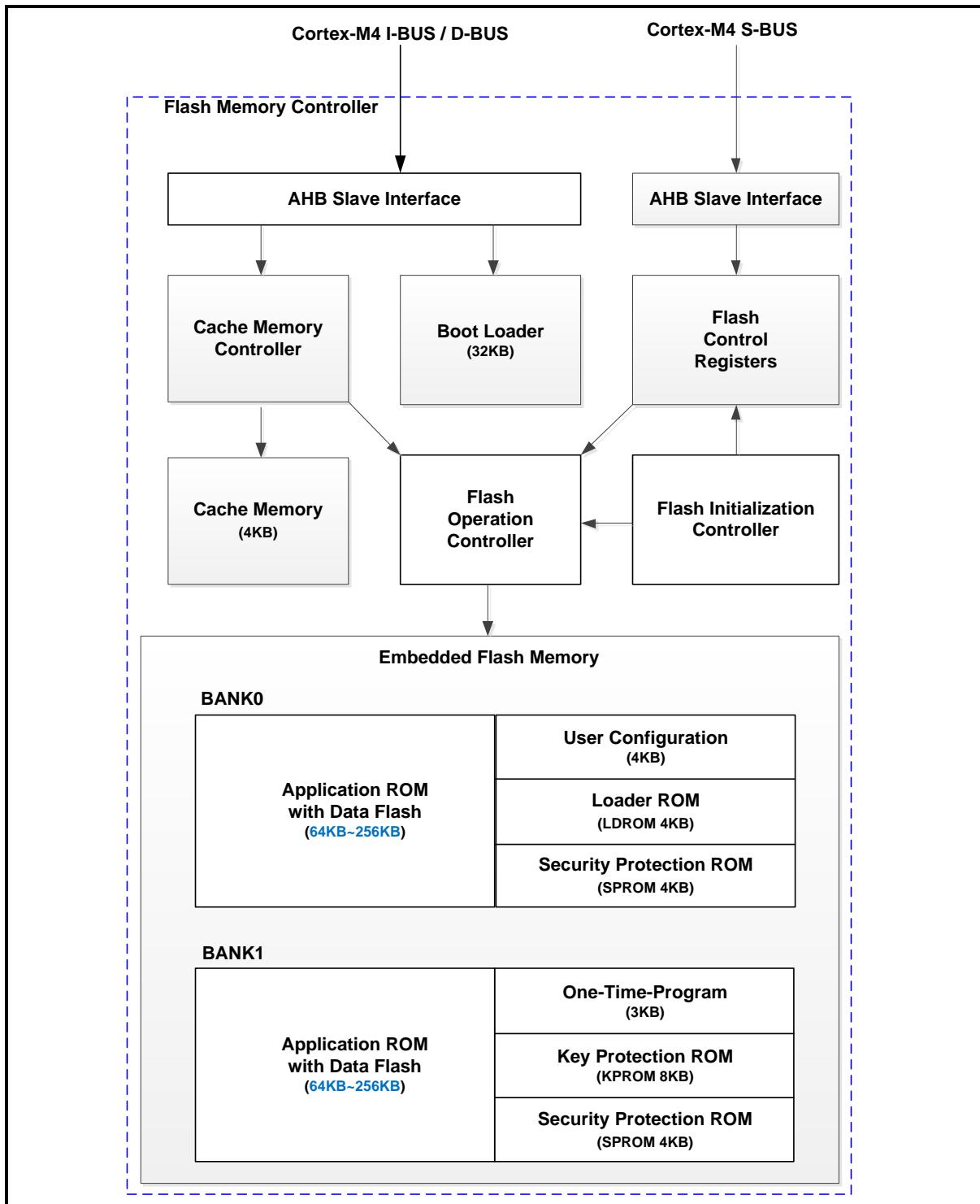


图 6.4-1 存储器控制器框图

**AHB 从接口**

在flash存储控制器中有两个AHB从接口，一个是来自Cortex®-M4 I-总线与D-总线，用于指令和数据读取；另一个是来自Cortex®-M4 S-总线，用于flash控制寄存器的访问，也用于ISP寄存器的访问。

### Cache 存储控制器

一个零等待周期的4 K字节cache，位于Cortex®-M4 CPU和片上flash之间。cache存储控制器提高了flash的访问效率并降低了功耗。

### Boot Loader

Boot Loader 大小是32K字节，包括了内建 ISP功能来更新片上flash。Boot Loader的内容是只读的，不可编程。

### Flash 控制寄存器

所有的ISP控制和状态寄存器都在flash控制寄存器中。详细的寄存器描述会在寄存器介绍章节介绍。

### Flash 初始化控制器

当芯片上电或复位，flash初始化控制器将开始自动访问flash，并且检测flash的稳定，重载用户配置 内容到flash控制寄存器用于系统初始化。

### Flash 操作控制器

对flash操作，例如flash 擦除，flash 编程和读flash，有明确的控制时序。Flash操作控制器在收到cache 存储控制器，flash 控制寄存器和flash初始化控制器的请求后，产生了这些控制时序。

### 片上 Flash 存储器

片上flash存储器是用于存储用户应用程序和参数的。它包括了用户配置区，4K字节的LDROM，两个4KB SPROM, 3KB OTP,8KB KPROM和包含数据flash的128KB/256KB/512KB APROM. 页擦除的flash大小是4KB，最小可编程位大小是32位。

#### 6.4.4 功能描述

FMC 功能包括存储器组织，启动选择，安全启动，IAP, ISP, 片上flash编程，和checksum 计算。在存储器组织中也介绍了flash存储器映射和系统存储器映射。

##### 6.4.4.1 存储器组织

FMC 存储包含了片上flash和boot loader。双bank片上存储器是可编程的，包括APROM, LDROM, SPROM, 数据Flash和用户配置区, OTP 和 KPROM。Boot Loader是一个掩膜的存储器带ISP启动代码，支持固件更新，启动控制，安全控制和固件执行。 地址映射包括flash存储映射和5个系统地址映射：支持IAP功能的LDROM，不支持IAP功能的LDROM，支持IAP功能的APROM，不支持IAP功能的APROM,以及支持IAP功能的Boot Loader

BANK	Flash 存储区块	地址范围
0	APROM 为 512KB	0x00_0000 ~ 0x03_ffff
	APROM为256KB	0x00_0000 ~ 0x01_ffff
	APROM为128KB	0x00_0000 ~ 0x00_ffff
	用户配置区	0x30_0000 ~ 0x30_000f
	LDROM	0x10_0000 ~ 0x10_0fff
	SPROM	0x20_0000 ~ 0x20_0fff
1	APROM为512KB	0x04_0000 ~ 0x07_ffff
	APROM为256KB	0x02_0000 ~ 0x03_ffff

	APROM为128KB	0x01_0000 ~ 0x01_ffff
	OTP	0x31_0000 ~ 0x31_0bff
	KPROM-KEY	0x30_1000 ~ 0x30_11ff
	KPROM - KPCNTROM	0x30_1200 ~ 0x30_1fff

表 6.4-1 双-Bank 区块地址范围

### LDROM APROM 和 Data Flash

LDROM 是用于通过引导程序，来执行在系统编程(ISP)的功能。LDROM是4KB的片上flash存储器，flash地址是从0x0010\_0000到0x0010\_0FFF。APROM是用户应用程序的主要存储器。APROM 大小是128KB/256KB/512KB。数据Flash是用于存储应用参数的（不是指令）。数据FLASH与APROM共享存储空间，大小可配置，数据Flash的基地址由寄存器DFBA (CONFIG1[19:0])设定。所有片上flash存储器，页擦除的大小是4KB。

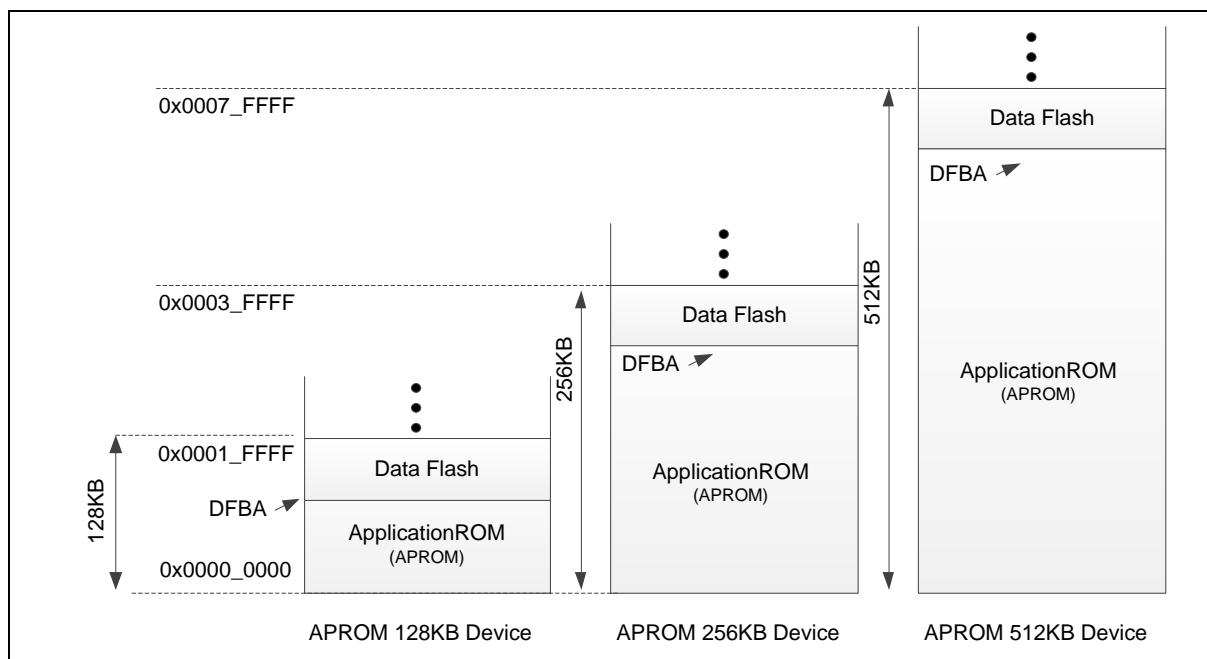


图 6.4-2 数据 FLASH 与 APROM 共享存储空间(128KB 和 256KB)

### 用户配置区

用户配置区是内部可编程的配置区域，用于启动选项，比如flash安全锁，启动选择，欠电压电平设置和数据flash基地址。用户配置区的作用类似保险丝用于上电时的缺省设置。在上电的时候，用户配置区设定会加载到相应的控制寄存器。用户可以通过不同的应用需求来设定寄存器。用户配置区可以通过ISP方式更新，位于地址 0x0030\_0000，有四个32位寄存器 (CONFIG0, CONFIG1, CONFIG2 和 CONFIG3)。用户配置所有的更改将在系统重启后生效。

**CONFIG0 (地址 = 0x0030\_0000)**

31	30	29	28	27	26	25	24
CWDTEN[2]	CWDTPDEN	Reserved		CFGXT1	Reserved		
23	22	21	20	19	18	17	16
CBOV			CBORST	CBODEN	Reserved		
15	14	13	12	11	10	9	8
Reserved			SPLCAEN	ICELOCK	CIOINI	Reserved	
7	6	5	4	3	2	1	0
CBS		MBS	CWDTE[1:0]		Reserved	LOCK	DFEN

位	描述
[31]	<b>CWDTEN[2]</b> 看门狗定时器硬件使能位  当看门狗定时器硬件使能功能有效，上电后，看门狗使能位WDTEN (WDT_CTL[7])和看门狗复位使能位RSTEN (WDT_CTL[1])会自动为1。看门狗的时钟源为LIRC，LIRC不能被禁用。  <b>CWDTEN[2:0]</b> 在寄存器 CONFIG0的相应位[31][4][3], 011 = WDT 看门狗硬件使能功能有效。除芯片进入掉电模式以外，WDT时钟一直开启。当芯片进入掉电模式，如果寄存器CWDTPDEN=0，WDT 时钟是一直开启的，如果寄存器CWDTPDEN =1，WDT时钟由寄存器LIRCEN (CLK_PWRCTL[3])控制。请参考寄存器CWDTPDEN的位描述。 111 = WDT 硬件使能功能无效。 其他 = WDT 硬件使能功能有效。WDT时钟一直开启。
[30]	<b>CWDTPDEN</b> 看门狗时钟掉电使能  0 = 当芯片进入掉电模式时，看门狗时钟保持使能 1 = 当芯片进入掉电模式时，看门狗时钟由LIRCEN (CLK_PWRCTL[3])控制.  注: 该位只有在CWDTEN[2:0]=011时才起作用
[29:28]	<b>Reserved</b> 保留.
[27]	<b>CFGXT1</b> <b>PF[4:3]多功能选择</b> 0 = PF[4:3]管脚配置为GPIO. 1 = PF[4:3]管脚配置成外部4~24 MHz高速晶振(HXT).
[26:24]	<b>Reserved</b> 保留.
[23:21]	<b>CBOV</b> 欠压电压选择 000 = 欠压电压是 1.6V. 001 =欠压电压是1.8V. 010 =欠压电压是2.0V. 011 =欠压电压是2.2V. 100 =欠压电压是2.4V. 101 =欠压电压是2.6V. 110 =欠压电压是2.8V. 111 =欠压电压是3.0V.

[20]	<b>CBORST</b>	欠压复位使能 0 = 上电后, 使能欠压复位 1 = 上电后, 禁用欠压复位
[19]	<b>CBODEN</b>	欠压侦测器使能位 0= 上电后, 欠压侦测器使能. 1=上电后, 欠压侦测器禁止
[18:13]	<b>Reserved</b>	保留.
[12]	<b>SPLCAEN</b>	<b>SPROM 锁死可缓存使能位</b> 0 = SPROM 的内容不可缓存 1 = SPROM 的内容可以缓存.
[11]	<b>ICELOCK</b>	<b>ICE 锁死位</b> 该位仅用于禁止 ICE 功能。用户可以使用它配合LOCK (CONFIG0[1]) 位 或 ALOCK (CONFIG2[7:0]) 位来增加系统的安全等级。 0 = ICE 功能禁止. 1 = ICE 功能使能.
[10]	<b>CIOINI</b>	<b>I/O 初始状态选择</b> 0 = 上电后所有GPIO默认为准双向模式 1 = 上电后所有GPIO默认为三态模式
[9:8]	<b>Reserved</b>	保留.
[7:6]	<b>CBS</b>	<b>芯片启动选择</b> 当CBS[0] = 0, LDROM的基地址映射到0x100000, APROM的基地址映射到0x0. 用户既可以访问APROM也可以访问LDROM, 无需启动切换。换句话说, 如果是IAP模式, 存放在LDROM与APROM的代码可以互相调用。 当MBS =0, CBS[0]值会被强制为0。 00=由LDROM启动支持IAP功能 01=由LDROM启动不支持IAP功能 10=由APROM启动支持IAP功能 11=由APROM启动不支持IAP功能 注: 当CBS[0] = 1并且MBS = 1, BS (FMC_ISPCTL[1])只用于启动切换的控制。 当CBS[0] = 0或MBS = 0, VECMAP (FMC_ISPSTS[23:9])只用于地址0x0~0x1ff重新映射。
[5]	<b>MBS</b>	<b>Boot Loader 启动选择</b> 0 = 从Boot loader启动, CBS设定无效。 1 = 依据CBS设定数值, 从APROM或LDROM启动。 注: 当CBS[0] = 1并且MBS = 1, BS (FMC_ISPCTL[1])只用于启动切换的控制。 当CBS[0] = 0或MBS = 0, VECMAP (FMC_ISPSTS[23:9])只用于地址0x0~0x1ff重新映射。

[4:3]	<b>CWDTEN</b>	<b>看门狗定时器硬件使能位</b> 当看门狗定时器硬件使能功能有效，上电后，看门狗使能位WDTEN (WDT_CTL[7])和看门狗复位使能位RSTEN (WDT_CTL[1])会自动为1。看门狗的时钟源为LIRC，LIRC不能被禁用。 <b>CWDTEN[2:0]</b> 在寄存器 CONFIG0的相应位[31][4][3], 011 = WDT 看门狗硬件使能功能有效。除芯片进入掉电模式以外，WDT时钟一直开启。当芯片进入掉电模式，如果寄存器CWDTPDEN=0，WDT 时钟是一直开启的，如果寄存器CWDTPDEN =1，WDT时钟由寄存器LIRCEN (CLK_PWRCTL[3])控制。请参考寄存器CWDTPDEN的位描述。 111 = WDT 硬件使能功能无效。 其他 = WDT 硬件使能功能有效。WDT时钟一直开启。
[2]	<b>Reserved</b>	保留.
[1]	<b>LOCK</b>	<b>加密锁控制</b> 0 = 加密FLASH数据 1 = 如果ALOCK(CONFIG2[7:0]) 和 SBLOCK (CONFIG2[15:8]) 不是 0x5A., 解除Flash 数据加密 当flash数据被LOCK加密，用户可以在lock效果表中找到FMC的Lock效果
[0]	<b>DFEN</b>	<b>数据FLASH使能位</b> 当DFEN=0时 APROM与数据flash共享，数据flash的基地址由DFBA (CONFIG1[19:0])设定。 0 = 使能数据FLASH 1 = 禁用数据FLASH.

注:如果是保留位，必须是1

**CONFIG1 (地址 = 0x0030\_0004)**

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved				DFBA			
15	14	13	12	11	10	9	8
DFBA							
7	6	5	4	3	2	1	0
DFBA							

位	描述	
[31:20]	<b>Reserved</b>	保留.
[19:0]	<b>DFBA</b>	数据Flash基地址 只有当DFEN (CONFIG0[0])=0时，该寄存器才工作。如果DFEN (CONFIG0[0])=1，数据Flash基地址由用户定义。因为片上FLASH擦除单位为4K字节，所以强制bit11-0位为0。

注:如果是保留位，必须是1

**CONFIG2 (地址 = 0x0030\_0008)**

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
SBLOCK							
7	6	5	4	3	2	1	0
ALOCK							

位	描述	
[31:16]	<b>Reserved</b>	保留.
[15:8]	<b>SBLOCK</b>	<p><b>安全启动锁控制</b>            0x5A = 如果 LOCK (CONFIG0[1]) 是 1 且 ALOCK (CONFIG2[7:0]) 是 0x5A，安全启动功能禁止，存储器内容是未锁状态。            其他 = 安全启动功能使能且 LOCK/ALOCK 也是锁死的。            当安全启动功能使能，FMC会从Boot Loader启动，带IAP模式（也就是CONFIG0中CBS[0]的值被指定为0）。然后执行安全启动。            当 SBLOCK ≠0x5A，连ICE，仅芯片擦除的ISP命令可以通过ICE执行，当flash数据被SBLOCK位锁定，用户必须在上锁效果表中查找FMC上锁的效果。.</p>
[7:0]	<b>ALOCK</b>	<p><b>高级安全锁控制</b>            0x5A = 如果LOCK (CONFIG0[1]) 是 1，Flash存储器内容是未锁的。            其他 = Flash 存储器内容是锁定的。            当flash数据被ALOCK位锁定，用户必须在上锁效果表中查找FMC上锁的效果。</p>

注：如果是保留位，必须是1

**CONFIG3 (地址= 0x0030\_000C)**

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved		SPIMPSL			Reserved		UART1PSL

位	描述	
[31:6]	Reserved	保留.
[5:4]	SPIMPSL	<b>Boot Loader SPIM 多功能管脚选择</b> 00: SPIM_CLK (PA.2), SPIM_SS (PA.3), SPIM_MISO (PA.1), SPIM_MOSI (PA.0) 01: SPIM_CLK (PC.2), SPIM_SS (PC.3), SPIM_MISO (PC.0), SPIM_MOSI (PC.1) 10: SPIM_CLK (PG.12), SPIM_SS (PG.11), SPIM_MISO (PG.13), SPIM_MOSI (PG.14) 11: SPIM_CLK (PE.4), SPIM_SS (PE.5), SPIM_MISO (PE.3), SPIM_MOSI (PE.2)
[3:2]	Reserved	保留.
[1:0]	UART1PSL	<b>Boot Loader UART1多功能管脚选择</b> 00: UART1_TXD (PB.3), UART1_RXD (PB.2) 01: UART1_TXD (PA.9), UART1_RXD (PA.8) 10: UART1_TXD (PA.3), UART1_RXD (PA.2) 11: UART1_TXD (PB.7), UART1_RXD (PB.6)

注: 当芯片复位时, 默认值来自CONFIG3 [1: 0]

注: 如果是保留位, 必须是1

### 安全程序空间(SPROM)

该芯片提供安全程序空间用于用户存储安全应用方面的指令。这块安全空间是4K字节，地址是0x20\_0000 ~ 0x20\_0FFF。这块空间只能被“页擦除”命令擦除。SPROM最后一个字节(地址:0x0020\_0FFF)是用来标识该空间是否加密的。相关信息见下图。SPROM有三种模式, (1) SPROM 非安全模式 (最后一个字节是0xFF), (2)SPROM 调试模式 (最后一个字节是0xAA), (3) SPROM 安全模式 (最后一个字节不是0xFF 或 0xAA)

为了在安全模式隐藏SPROM代码, CPU仅可以执行从SPROM获取指令, 然后获取SPROM数据, 任何CPU直接数据访问SPROM, 没有执行从SPROM获取指令, CPU都会得到全部是相同(0xffff\_ffff).的数据。为了安全保护, 当Cortex®-M4 ICE (In-Circuit-Emulator) 正在SPROM安全代码调试, 任何ICE在SPROM程序设置的断点和通过ICE或PDMA取出SPROM的安全代码都是非法的。

SPROM安全标志显示在SCODE (FMC\_ISPSTS[31]), 该标志在SPROM页擦除操作完成后被清0。如果SPROM的最后一个字节不是0xFF, 在flash初始化的时候该标志被置位。此外用户也可以通过写1到寄存器FMC\_ISPSTS[31]来置位SCODE。在执行SPROM页擦除之前, SPUEN (FMC\_ISPCTL[2])必须置1, 且在SPROM安全模式ISPDAT[31:0]寄存器必须设置为0x0055AA03。

为了实现安全固件更新, 我们分配一个SPROM页带安全应用的相同内容在每个bank中, 以确保安全应用可以在任何bank中执行。当同时更新固件代码和访问SPROM, FMC会在任何空闲的bank中返回SPROM的指令。请注意当在任何SPROM执行ISP flash页擦除命令功能或flash数据编程功能, 所有的SPROM会被自动地擦除或编程。

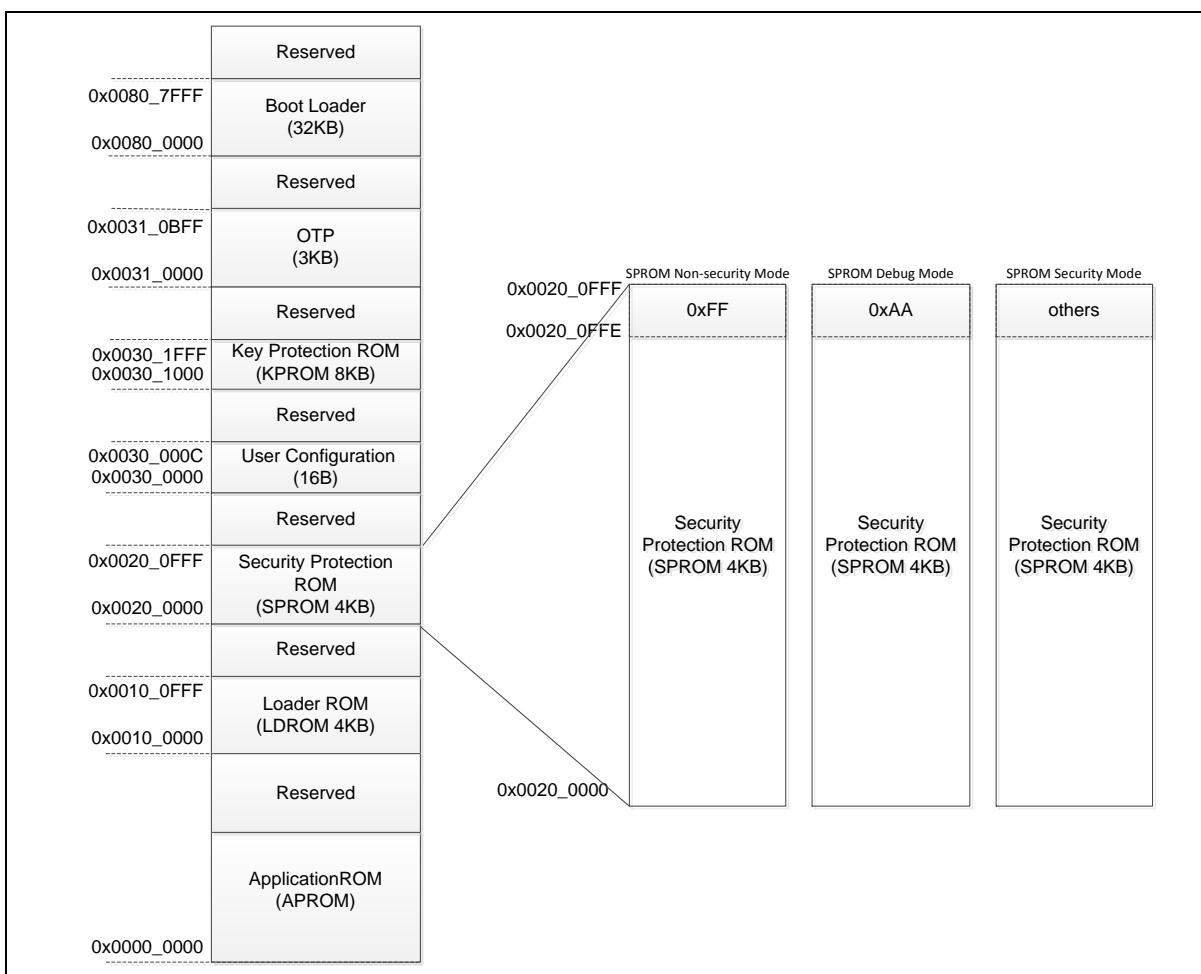


图 6.4-3 SPROM 安全模式

### 单次编程存储器(OTP)

有一个单次编程存储器(OTP)用来保存重要信息，该区域不允许用户再次修改，OTP是3K字节大小，地址空间是0x31\_0000 ~ 0x31\_0BFF。OTP数据最大空间是2K字节，地址是0x31\_0000 ~ 0x31\_07FF，每64位OTP数据有一个32位的“LOCK”位，地址从0x31\_0800 到 0x31\_0BFF，如下图所示。

“LOCK BIT”的目的是记录在OTP被编程的地址是否锁定，锁定  $\text{LOCK BIT} \neq 0xFFFF\_FFFF$ ，未锁定  $\text{LOCK BIT} = 0xFFFF\_FFFF$ 。例如，当LOCK BIT0不是0xFFFFFFFF，不管其内容是否全为1，OTP0不能再次被编程。“flash 页擦除 /批量擦除命令/ FLASH 64-bit 编程/ FLASH 多字编程”永远不允许在OTP执行。在从0x31\_0000 到 0x31\_07FF更新OTP的内容之前，用户必须先检查“LOCK BIT”。完成OTP数据编程之后，用户必须写一个非0xFFFF\_FFFF的数据到之前编程数据的“LOCK BIT”，来确保不能再次修改。OTP通过ISP命令是只读的，CPU直接读是不可读的。

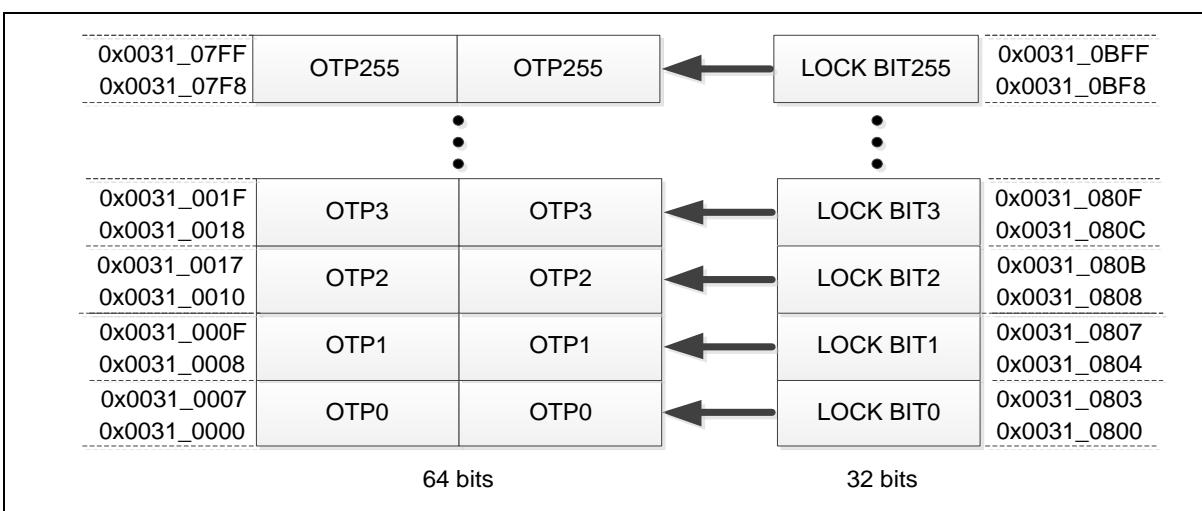


图 6.4-4 OTP 存储器映射

### 密钥保护存储(KPROM)

密钥保护存储(KPROM)是用来存储96位的安全密钥及限制不匹配的密钥访问。KPROM包含28 bytes，地址在0x30\_1000 ~ 0x30\_1017及0x30\_1200 ~ 0x30\_1203，在KEYLOCK (FMC\_KPKEYSTS [1])没有被锁定时可以编程及擦除该区域。KPROM不能通过ISP/ICE/ICP /烧录器 命令和接口直接读取。如果需要校验KPROM的数据，只能通过匹配密钥操作来校验安全密匙 (KPKEY0ROM, KPKEY1ROM, 和 KPKEY2ROM)的数据。KPROM内存映射如下图。

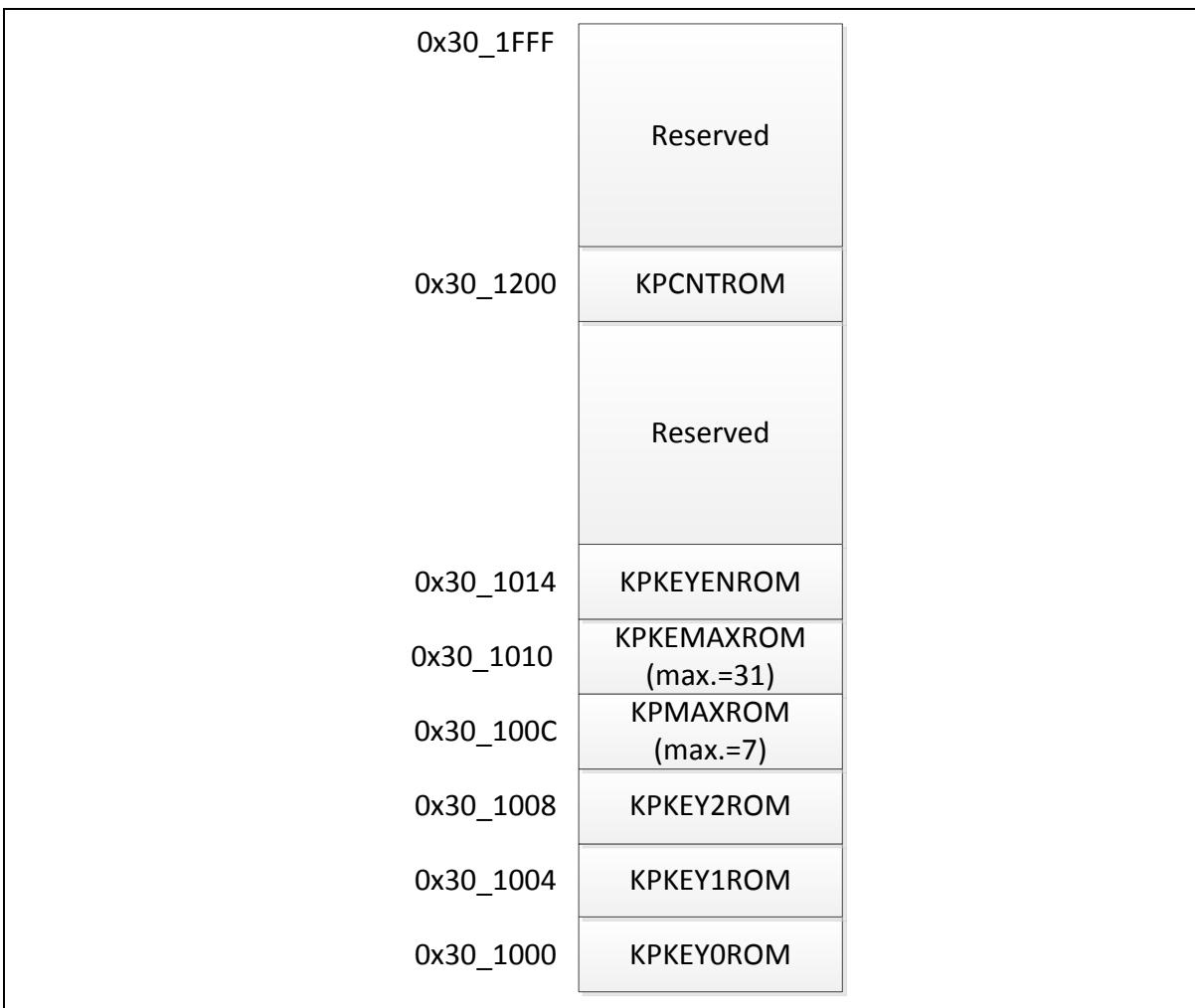


图 6.4-5 KPROM 存储映射

KPKEY0ROM (地址= 0x0030\_1000)

31	30	29	28	27	26	25	24
KPKEY0ROM							
23	22	21	20	19	18	17	16
KPKEY0ROM							
15	14	13	12	11	10	9	8
KPKEY0ROM							
7	6	5	4	3	2	1	0
KPKEY0ROM							

位	描述	
[31:0]	KPKEY0ROM	<b>KEY #0 (只写)</b> KPKEY0ROM是96位安全密钥的前32位。KEYLOCK (FMC_KPKEYSTS [1])没有被锁定时，该区域可以被擦除和编程。KPKEY0ROM不能直接读取。

注意：这些位默认为1。

KPKEY1ROM (地址= 0x0030\_1004)

31	30	29	28	27	26	25	24
KPKEY1ROM							
23	22	21	20	19	18	17	16
KPKEY1ROM							
15	14	13	12	11	10	9	8
KPKEY1ROM							
7	6	5	4	3	2	1	0
KPKEY1ROM							

位	描述	
[31:0]	KPKEY1ROM	KEY #1 (只写) KPKEY1ROM是96位安全密钥的第二个32位。KEYLOCK (FMC_KPKEYSTS [1])没有被锁定时，该区域可以被擦除和编程。KPKEY1ROM不能直接读取。.

KPKEY2ROM (地址 = 0x0030\_1008)

31	30	29	28	27	26	25	24
KPKEY2ROM							
23	22	21	20	19	18	17	16
KPKEY2ROM							
15	14	13	12	11	10	9	8
KPKEY2ROM							
7	6	5	4	3	2	1	0
KPKEY2ROM							

位	描述	
[31:0]	KPKEY2ROM	KEY #2 (只写) KPKEY2ROM是96位安全密钥的第二个32位。KEYLOCK (FMC_KPKEYSTS [1])没有被锁定时，该区域可以被擦除和编程。KPKEY2ROM不能直接读取。

**KPMAXROM (地址 = 0x0030\_100C)**

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved					<b>KPMAXROM</b>		

位	描述	
[31:3]	<b>Reserved</b>	保留.
[2:0]	<b>KPMAXROM</b>	<p><b>密钥尝试的最大错误上电次数(只写)</b></p> <p>KPMAXROM在安全密钥不匹配的情况下，KPMAXROM限制了尝试的上电次数。上电时或复位后，KPMAXROM会被拷贝到寄存器 KPMAX (FMC_KPCNT [10:8])。KEYLOCK (FMC_KPKEYSTS [1])没有被锁定时，KPMAXROM可以被擦除和编程。KPMAXROM不能直接读取。</p>

**KPKEMAXROM (地址 = 0x0030\_1010)**

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved			<b>KPKEMAXROM</b>				

位	描述	
[31:5]	<b>Reserved</b>	保留.
[4:0]	<b>KPKEMAXROM</b>	<p>每次上电密钥尝试最大错误次数(只写)</p> <p>KPKEMAXROM在安全密钥不匹配的情况下，KPKEMAXROM限制了每次上电时的尝试次数。上电时或复位后，KPKEMAXROM会被拷贝到寄存器KPKEMAX(FMC_KPKECNT[12:8])。KEYLOCK (FMC_KPKEYSTS [1])没有被锁定时，KPKEMAXROM可以被擦除和编程。KPKEMAXROM不能直接读取。</p>

**KPKEYENROM (地址 = 0x0030\_1014)**

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
KPKEYENROM							

位	描述	
[31:8]	Reserved	保留.
[7:0]	KPKEYENROM	<p><b>KEYLOCK使能控制位(只写)</b> 在上电时或复位后, KEYLOCK (FMC_KPKEYSTS [1]) 可以由KPKEYENROM使能。KEYLOCK (FMC_KPKEYSTS [1])没有被锁定时, KPKEYENROM可以被擦除和编程但不能直接读取。</p> <p>0xFF=上电时或复位后, 寄存器KEYLOCK(FMC_KPKEYSTS [1])清0。 其他=上电时或复位后, 寄存器KEYLOCK(FMC_KPKEYSTS [1])置1。KPROM, LDROM 和 APROM写保护。 若KEYENROM[0]=0, CONFIG写保护。 若KEYENROM[1]=0, SPROM写保护。</p>

注: 当KPKEYENROM通过ISP功能编程为非0xFF值时, KEYLOCK (FMC\_KPKEYSTS [1]) 将立即启用。

**KPCNTROM (地址 = 0x0030\_1200)**

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
KPCNTROM							

位	描述																			
[31:8]	<b>Reserved</b>	保留.																		
[7:0]	<b>KPCNTROM</b>	<p><b>密钥错误尝试上电计数(只写)</b></p> <p>KPCNTROM会记录上电/断电时的密钥不匹配次数。第一次密钥不匹配时，FMC会清除一位(由KPCNT解码得到)。当密钥匹配时，KPCNTROM会被硬件擦除且不能直接读出。</p> <p>KPCNTROM(8位)在上电或复位后编码成3位之后存储在KPCNT(FMC_KPCNT[2:0])，如下所示：</p> <p>KPCNTROM → KPCNT</p> <table> <tbody> <tr><td>1111_1111</td><td>→ 000</td></tr> <tr><td>1111_1110</td><td>→ 001</td></tr> <tr><td>1111_1100</td><td>→ 010</td></tr> <tr><td>1111_1000</td><td>→ 011</td></tr> <tr><td>1111_0000</td><td>→ 100</td></tr> <tr><td>1110_0000</td><td>→ 101</td></tr> <tr><td>1100_0000</td><td>→ 110</td></tr> <tr><td>1000_0000</td><td>→ 111</td></tr> <tr><td>其他</td><td>→ 111</td></tr> </tbody> </table>	1111_1111	→ 000	1111_1110	→ 001	1111_1100	→ 010	1111_1000	→ 011	1111_0000	→ 100	1110_0000	→ 101	1100_0000	→ 110	1000_0000	→ 111	其他	→ 111
1111_1111	→ 000																			
1111_1110	→ 001																			
1111_1100	→ 010																			
1111_1000	→ 011																			
1111_0000	→ 100																			
1110_0000	→ 101																			
1100_0000	→ 110																			
1000_0000	→ 111																			
其他	→ 111																			

**Flash 内存映射**

Flash存储映射和系统存储映射不同。系统存储映射是CPU用来从FMC存储取代码和数据。而Flash存储映射是ISP用于读取、编程及擦写FMC存储。下图是Flash存储映射.

	Reserved	Reserved	Reserved
0x0080_7FFF	Boot Loader (32KB)	Boot Loader (32KB)	Boot Loader (32KB)
0x0080_0000	Reserved	Reserved	Reserved
0x0031_0BFF	OTP (3KB)	OTP (3KB)	OTP (3KB)
0x0031_0000	Reserved	Reserved	Reserved
0x0030_2FFF	Key Protection (KROM 8KB)	Key Protection (KROM 8KB)	Key Protection (KROM 8KB)
0x0030_1000	Reserved	Reserved	Reserved
0x0030_000F	User Configuration (16B)	User Configuration (16B)	User Configuration (16B)
0x0030_0000	Reserved	Reserved	Reserved
0x0020_0FFF	Security Protection ROM (SPROM 4KB)	Security Protection ROM (SPROM 4KB)	Security Protection ROM (SPROM 4KB)
0x0020_0000	Reserved	Reserved	Reserved
0x0010_0FFF	Loader ROM (LDROM 4KB)	Loader ROM (LDROM 4KB)	Loader ROM (LDROM 4KB)
0x0010_0000	Reserved	Reserved	Reserved
0x0007_FFFF	Reserved	Reserved	ApplicationROM (APROM)
0x0001_FFFF	ApplicationROM (APROM)	ApplicationROM (APROM)	ApplicationROM (APROM)
0x0000_0000			
	APROM 128KB Device	APROM 256KB Device	APROM 512KB Device

图 6.4-6 Flash 存储映射

### 支持 IAP 模式的系统存储映射

在 CPU 访问 FMC 存储器获取代码或数据的时候，用到系统存储器映射。Boot Loader (0x0080\_0000~0x0080\_7FFF)、SPROM(0x0020\_0000~0x0020\_0FFF) 和 LDROM(0x0010\_0000~0x0010\_0FFF) 在 flash 中的地址映射相同。数据 flash 与 APROM 共享，数据 flash 的基址由 CONFIG1 设定。在 flash 初始化时，CONFIG1 的内容被加载到 DFBA(数据 Flash 基址寄存器)。

当 APROM 大小分别是 64KB/128KB/256KB/384KB/512KB 时，

DFBA~(0x0000\_FFFF/0x0001\_FFFF/0x0003\_FFFF/ 0x0005\_FFFF/0x0007\_FFFF) 是 Cortex®-M4 数据存取的数据Flash区，0x0000\_0200~(DFBA-1)是Cortex®-M4指令存取的APROM区。

系统存储向量的地址在0x0000\_0000 到 0x0000\_01FF。

下图表示支持IAP模式的系统存储映射

在CPU启动期间，APROM, LDROM和Boot loader可以映射到系统存储向量区。当芯片启动的时候，有三种支持IAP的系统存储器映射模式：(1)支持IAP的LDROM (2)支持IAP的APROM (3)支持IAP的Boot loader。

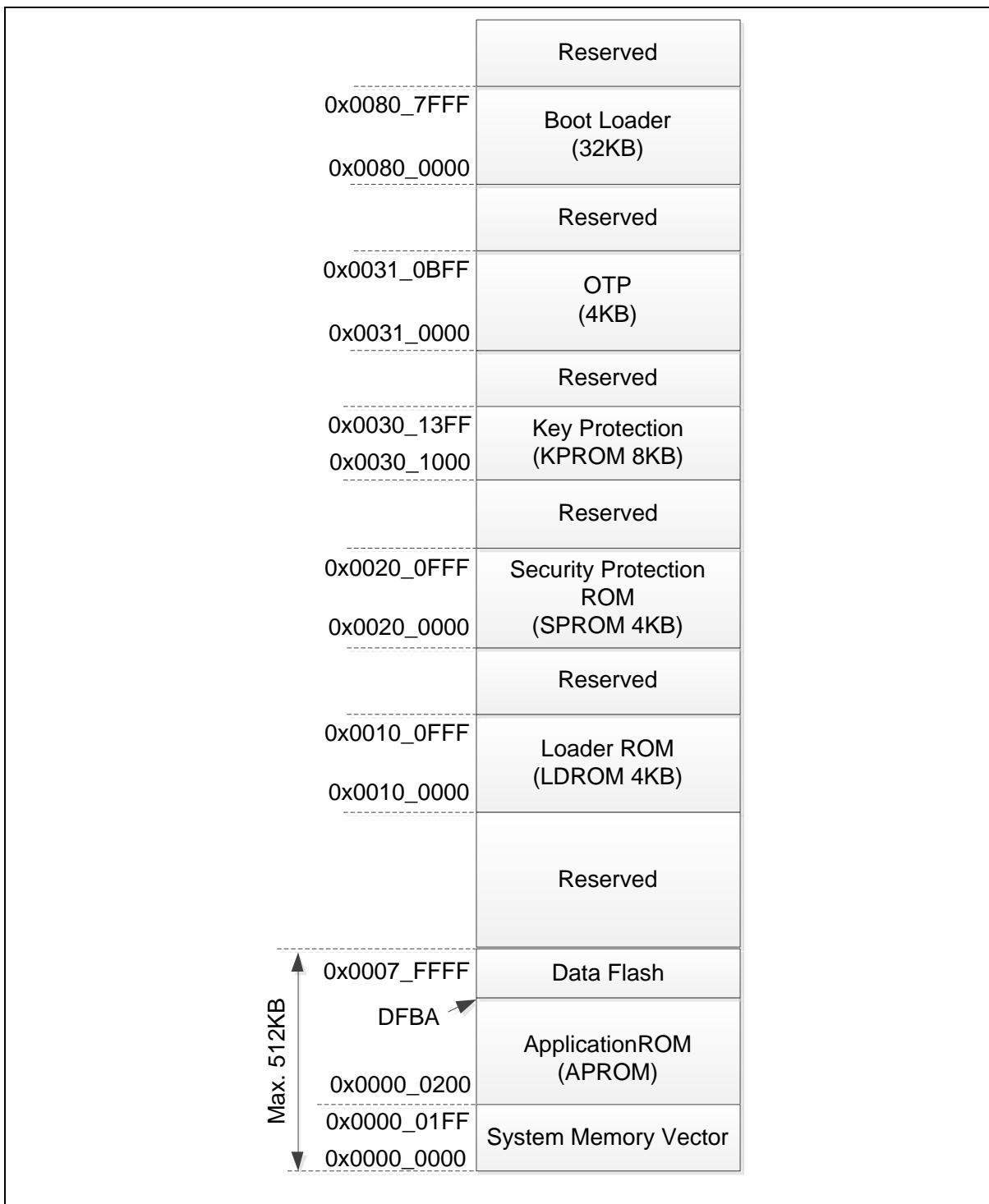


图 6.4-7 支持 IAP 模式的系统存储映射

支持IAP的LDROM模式，LDROM (0x0010\_0000~0x0010\_01FF)映射到系统存储向量区，用于Cortex®-M4指令或数据存取。下图表示支持IAP的LDROM模式的存储映射。

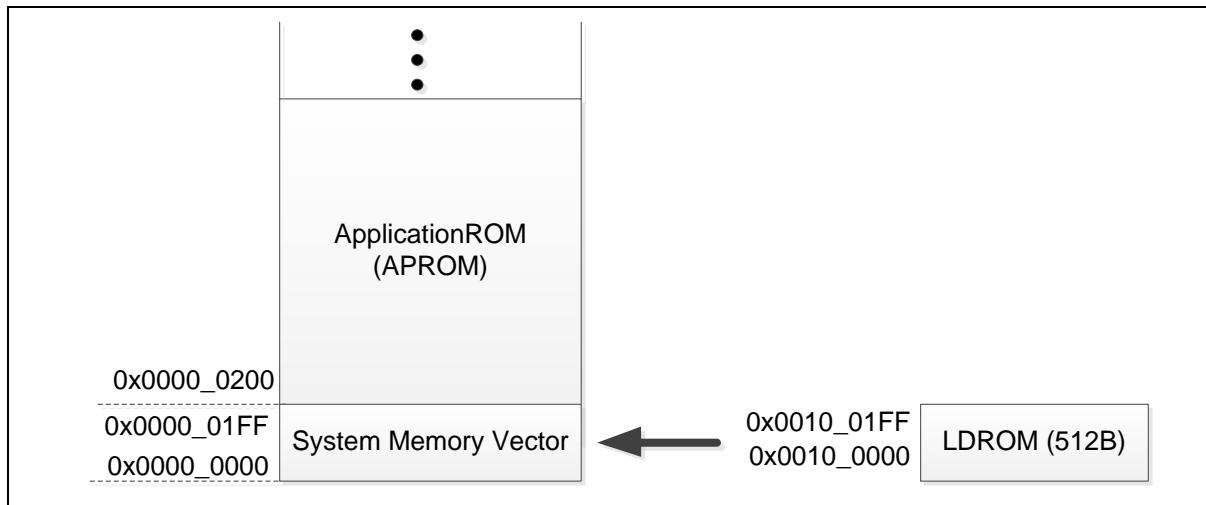


图 6.4-8 支持 IAP 的 LDROM 模式

支持 IAP 的 APROM 模式，APROM (0x0000\_0000~0x0000\_01FF) 映射到系统存储向量区，用于 Cortex®-M4 指令或数据存取。下图表示支持 IAP 的 APROM 模式的存储映射。

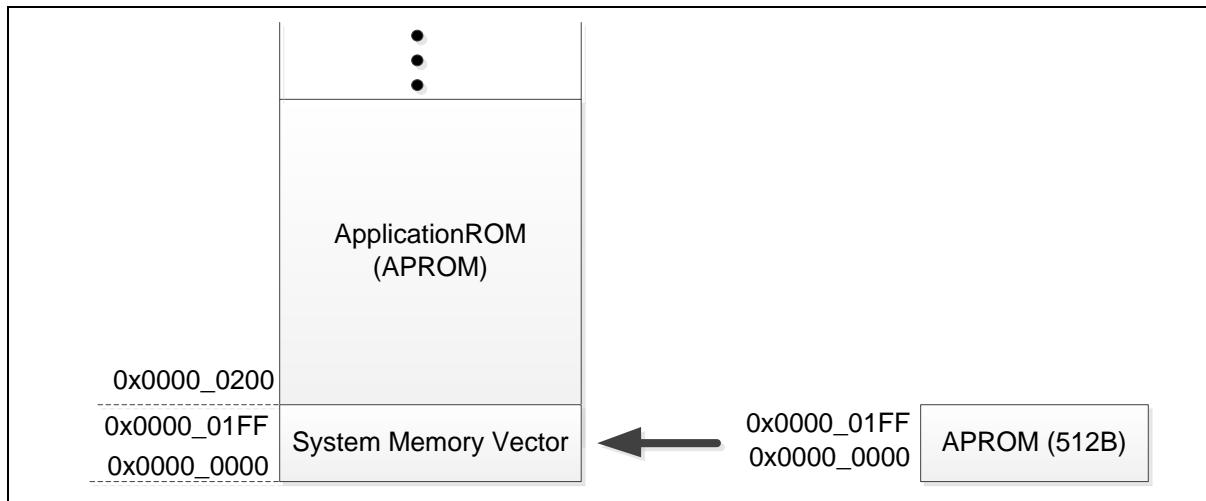


图 6.4-9 支持 IAP 的 APROM 模式

支持 IAP 的 Boot Loader 模式，Boot Loader (0x0080\_0000~0x0080\_01FF) 映射到系统存储向量区，用于 Cortex®-M4 指令或数据存取。下图表示支持 IAP 的 Boot Loader 模式的存储映射。

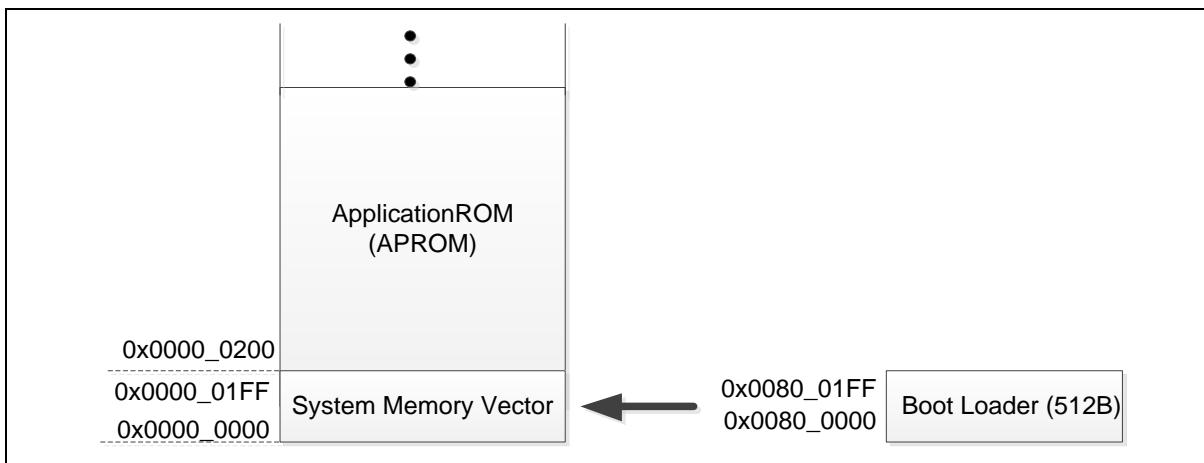


图 6.4-10 支持 IAP 的 Boot Loader 模式

在支持IAP的系统存储映射模式中，当CPU运行时，APROM, LDROM, 和Boot Loader可以重新映射到系统存储向量区。用户可以写重映射的目标地址到寄存器FMC\_ISPADDR，然后通过“向量重映射”指令(0x2E)触发ISP流程。在寄存器VECMAP (FMC\_ISPSTS[23:9])，显示最终的系统存储向量映射地址。

#### 不支持 IAP 功能的系统内存映射

不支持IAP功能的系统内存映射，CPU仍然可以访问Boot Loader(0x0080\_0000~0x0080\_7FFF)和SPROM(0x0020\_0000~0x0020\_0FFF)，但不支持系统内存向量映射。在芯片启动时，有两种不支持IAP的系统内存映射：(1) 不支持IAP功能的LDROM模式 (2) 不支持IAP功能的APROM模式。在不支持IAP功能的LDROM模式中，LDROM 基址地址映射到0x0000\_0000，CPU 编程不能访问APROM。在不支持IAP功能的APROM模式中，APROM的基址地址映射到0x0000\_0000，CPU编程不能访问LDROM。数据Flash与APROM共享，数据Flash的基址地址由CONFIG1设定。在flash初始化期间，CONFIG1内容被加载到DFBA(数据Flash地址)。当APROM大小分别是 64KB/128KB/256KB/384KB/512KB时，DFBA~(0x0000\_FFFF/0x0001\_FFFF/0x0003\_FFFF/0x0005\_FFFF /0x0007\_FFFF) 是Cortex®-M4的数据Flash区，0x0000\_0000~(DFBA-1)是Cortex®-M4的APROM区。下图表示不支持IAP模式的系统存储映射。

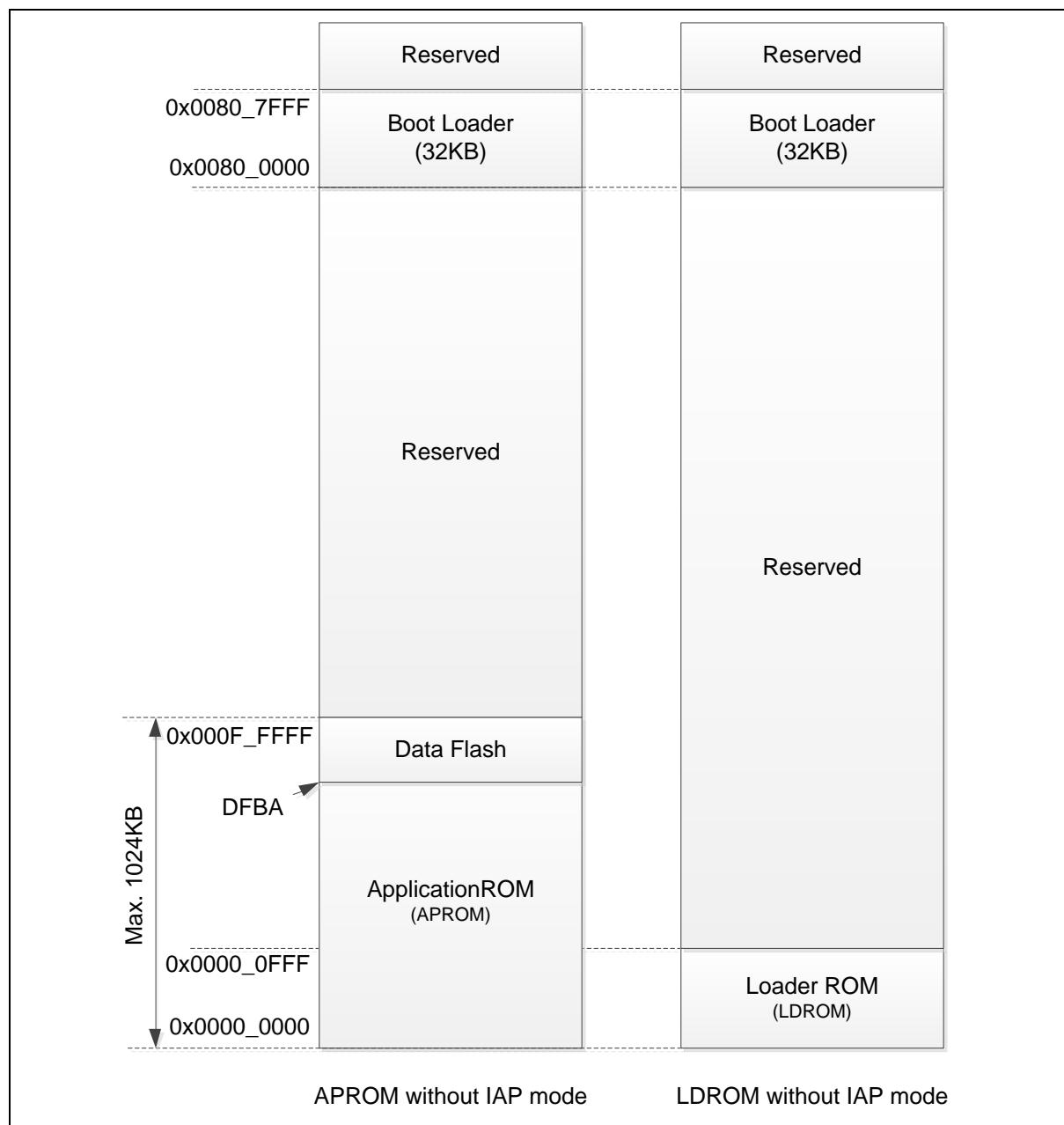


图 6.4-11 不支持 IAP 模式的系统存储映射

#### 6.4.4.2 启动选择

NuMicro™ M480 提供了五种启动方式供用户选择，包括支持 IAP 功能的 LDROM，不支持 IAP 功能的 LDROM，支持 IAP 功能的 APROM，不支持 IAP 功能的 APROM，支持 IAP 功能的 Boot Loader。用户可以用下面三种方法来让 CPU 从 Boot loader 中取指令。PF.6 管脚在复位管脚电平上升期间为低电平、MBS (CONFIG0[5]) 为低或 SBLOCK (CONFIG2[15:8]) 不是 0x5A。其他情况（例如 PF.6 不连接，PF.6 是有内部上拉的）启动来源和系统内存映射由 CBS (CONFIG0[7:6]) 和 MBS (CONFIG0[5]) 设置。

启动源选择如下图，每一个启动选项是否支持向量映射如下表。

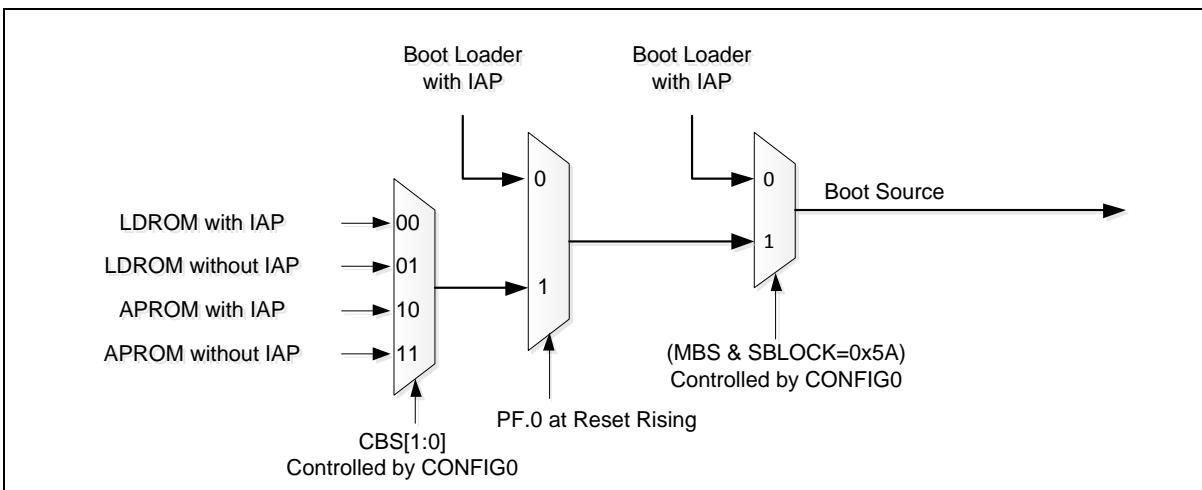


图 6.4-12 启动源选择

MBS	CBS[1:0]	启动选择/系统存储映射	向量映射是否支持
1	00	LDROM 支持 IAP	√
1	01	LDROM 不支持 IAP	-
1	10	APROM 支持 IAP	√
1	11	APROM 不支持 IAP	-
0	XX	Boot Loader 支持 IAP	√

表 6.4-2 向量映射支持情况

#### 6.4.4.3 在应用编程(IAP)

NuMicro M480 系列提供了在应用编程(IAP)功能，用户可以切换APROM, LDROM 和 Boot Loader之间的代码。用户可以使能IAP功能通过设定芯片的启动位寄存器CBS (CONFIG0[7:6]) 等于10 或00，或者MBS (CONFIG0[5]) 等于 0，或者SBLOCK (CONFIG2[15:8]) 是一个非 0x5A 的值。(如果MSB(CONFIG0[5])=0或执行安全启动，CBS[0] (FMC\_ISPSTS[1])将显示为0，并且忽略CONFIG0[6]的设定)。

当支持IAP功能的芯片启动模式使能，任何可以执行的代码(512字节对齐)允许随时映射到系统内存向量。用户可以改变重映射地址到FMC\_ISPADDR，然后用”向量重映射”命令触发ISP流程。

#### 6.4.4.4 在系统编程(ISP)

NuMicro™ M480系列支持在系统编程(ISP)模式，可以通过软件控制重新烧写片上flash。使用ISP可以在

目标板上直接编程，固件与多种接口相搭配，如UART, USB, I2C, SPI, 和 CAN (依据各芯片特性).使得用户可以使用多种方法来更新APROM的代码。ISP功能操作的目标flash存储空间不能跨bank，下表列出了所有ISP命令。

ISP 对片上flash操作提供如下功能。

- 支持flash页擦除功能
- 支持flash数据编程
- 支持读flash数据功能
- 支持读公司ID功能
- 支持读设备ID功能
- 支持读UID功能
- 支持内存CRC32 checksum计算功能
- 支持flash位全为1校验功能
- 支持系统内存向量重映射功能

### ISP 命令

ISP 命令	FMC_ISPCMD	FMC_ISPADDR	FMC_ISPDAT FMC_MPDATA0~FMC_MPDATA3
FLASH 页擦除	0x22	FLASH存储器的有效地址 它必须是页（4K字节）地址对齐。注意 FMC_ISPADDR[11:0] 会被忽略。.	N/A
FLASH Bank 擦除	0x23	目标bank的APROM的有效地址 注意 FMC_ISPADDR[15:0] 会被忽略。	N/A
FLASH 块擦除	0x25	APROM的有效地址 它必须是4页（16K字节）地址对齐 注意 FMC_ISPADDR[13:0] 会被忽略。	N/A
FLASH 批量擦除 (该命令仅当 MERASE(CONFIG0[13]) 位 = 0.时有效)	0x26	0x0000_0000	N/A
FLASH 32-位编程	0x21	FLASH存储器的有效地址	FMC_ISPDAT : 编程数据
FLASH 64-位编程	0x61	FLASH存储器的有效地址	FMC_MPDATA0: 编程数据的低位 (LSB) FMC_MPDATA1: 编程数据的高位 (MSB)
FLASH 多字 (word) 编程	0x27	FLASH存储器的有效地址	FMC_MPDATA0: 第一个编程数据 FMC_MPDATA1: 第二个编程数据 FMC_MPDATA2: 第三个编程数据 FMC_MPDATA3: 第四个编程数据

FLASH 读	0x00	FLASH存储器的有效地址	FMC_ISPDAT: 返回数据
FLASH 64-位 读	0x40	FLASH存储器的有效地址	FMC_ISPDAT: FMC_ISPADDR 的返回数据 FMC_MPDATA0: FMC_ISPADDR 的返回数据 FMC_MPDATA1: FMC_ISPADDR+4的返回数据
读公司 ID	0x0B	0x0000_0000	FMC_ISPDAT: 0x0000_00DA
读设备 ID	0x0C	0x0000_0000	FMC_ISPDAT: 返回设备 ID
读 CRC32 Checksum	0x0D	0x0000_0000	FMC_ISPDAT: 返回 Checksum
执行 CRC32 Checksum 计算	0x2D	内存组织有效起始地址 它必须由4K字节/页排列	FMC_ISPDAT: 要计算的内存大小，必须 4 K字节 对齐
读flash位全为1校验结果	0x08	保持flash位全为1校验的地址	FMC_ISPDAT: 返回的结果 0xA110_0000 : Flash位不全为1 0xA11F_FFFF: Flash位全为1
执行flash位全为1校验	0x28	内存组织的有效起始地址 必须 4 K字节 页对齐	FMC_ISPDAT: flash大小 必须 4 Kbytes 对齐
读UID	0x04	0x0000_0000	FMC_ISPDAT: U ID 字0
		0x0000_0004	FMC_ISPDAT:U ID 字1
		0x0000_0008	FMC_ISPDAT: UID 字2
向量重映射	0x2E	APROM,LDRAM 或 boot loader 有效地址 它必须512字节地址对齐	N/A

表 6.4-3 ISP 命令表

### ISP 流程

FMC控制器提供了片上flash内存的读，擦除和编程操作。一些FMC控制器的寄存器是写保护的，所以在设定之前要解锁。

在解锁了保护寄存器之后，用户需要设定FMC\_ISPCTL控制寄存器来决定更新LDROM、 APROM、 SPROM或配置区，然后设定ISPEN (FMC\_ISPCTL[0]) 来使能ISP功能。

一旦FMC\_ISPCTL寄存器被设置成功，用户可以设定FMC\_ISPCMD(参考上述ISP命令列表)来完成相应的操作。设置FMC\_ISPADDR作为flash内存的目标地址。FMC\_ISPDAT根据命令FMC\_ISPCMD，可以

设定为要编程的数据或作为读数据的返回数据。ISP流程如下图所示。

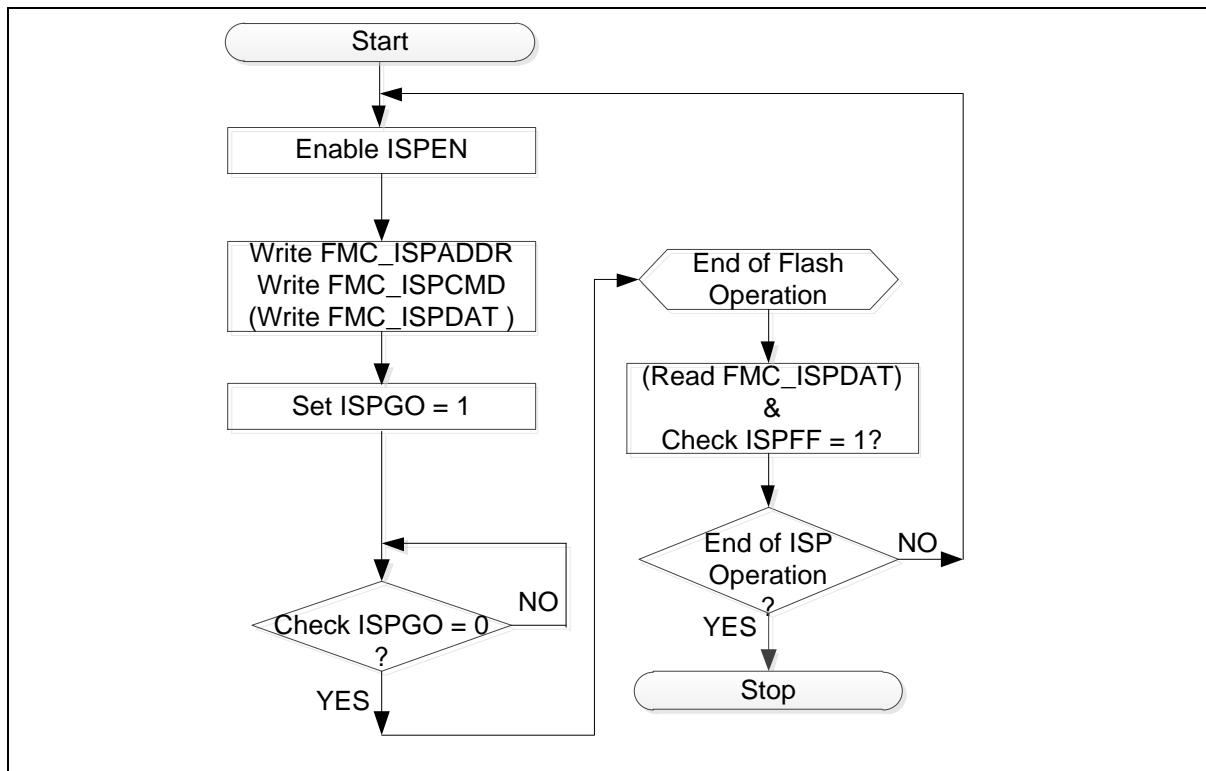


图 6.4-13 ISP 流程示例

最终，设定 ISPGO (FMC\_ISPTRG[0]) 寄存器来执行 ISP 功能。当 ISP 功能启动，ISPBUSY(FMC\_ISPSTS[0]) 和 MPBUSY(FMC\_MPSTS[0]) 会被置 1。当 ISP 功能完成以后，ISPGO(FMC\_ISPTRG[0]), ISPBUSY(FMC\_ISPSTS[0]) 和 MPBUSY(FMC\_MPSTS[0]) 自动清 0。

ISP完成后，几个错误条件需要检查。如果出现错误，ISP操作就不会生效并且ISP失败标志被置位。ISPFF(FMC\_ISPSTS[6])标志只由软件清除。即使ISPFF(FMC\_ISPSTS[6])保持1，也可以开启下一个ISP流程。因此，建议在ISP操作完成后检查ISPFF(FMC\_ISPSTS[6])位，如果被置1要清0。

当ISPGO(FMC\_ISPTRG[0])置位，然后CPU访问相同的bank，CPU将一直等到ISP操作完成。这个时期内外围设备跟通常一样保持运行。任何中断请求都不会响应直到CPU完成ISP操作。当ISP操作完成，ISPGO位将被硬件自动清0。用户可以通过ISPGO(FMC\_ISPTRG[0])位来检查ISP操作是否完成。

当CPU访问操作和ISP命令执行在不同的bank，CPU 和 ISP命令可以并行工作。

#### 6.4.4.5 片上Flash 编程

NuMicro M480提供了32位，64位和多字节flash存储器编程功能来加速flash更新流程。

下表列出了FMC控制寄存器对应的编程功能。

寄存器	描述	32-bit 编程	64-bit 编程	多字节编程
FMC_ISPCTL	ISP 控制寄存器	✓	✓	✓
FMC_ISPADDR	ISP 地址寄存器	✓	✓	✓

FMC_ISPDAT	ISP 数据寄存器	✓	N/A	N/A
FMC_ISPCMD	ISP 命令寄存器	0x21	0x61	0x27
FMC_ISPTRG	ISP 触发寄存器	✓	✓	✓
FMC_ISPSTS	ISP 状态寄存器	✓	✓	N/A
FMC_MPDATA0	ISP Data0 寄存器	N/A	✓	✓
FMC_MPDATA1	ISP Data1 寄存器	N/A	✓	✓
FMC_MPDATA2	ISP Data2 寄存器	N/A	N/A	✓
FMC_MPDATA3	ISP Data3 寄存器	N/A	N/A	✓
FMC_MPSTA	ISP 多字编程状态	N/A	N/A	✓
FMC_MPADDR	ISP 多字编程地址	N/A	N/A	✓

表 6.4-4 FMC 用于 Flash 编程的控制寄存器

**64-bit 编程**

NuMicro™ M480 系列 64-位编程功能比 32-位编程功能快。FMC\_ISPDAT 是 32-位编程数据寄存器。在 64-位编程中，有两个编程数据寄存器，一个是低位 FMC\_MPDATA0，另一个是高位 FMC\_MPDATA1，ISP 命令是 0x61，其他寄存器与 32-位编程寄存器相同。下图是 ISP 的 32-位/64-位编程流程。

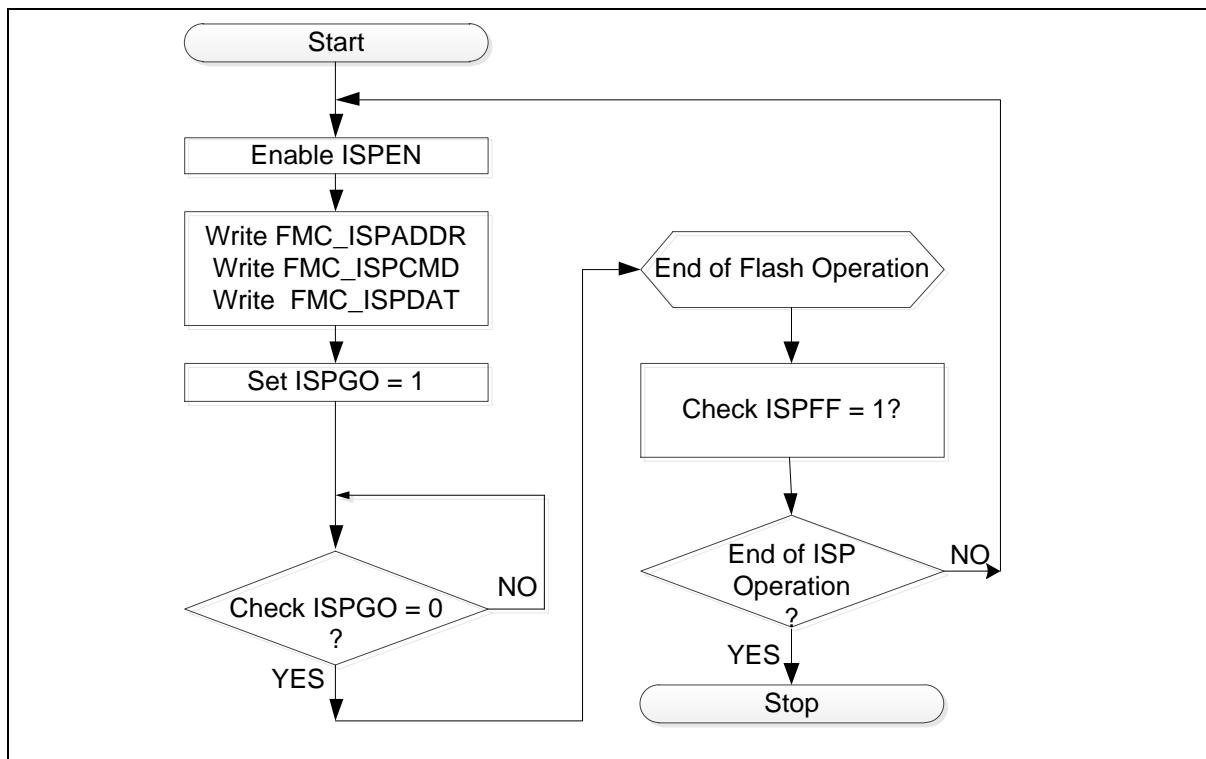


图 6.4-14 ISP 32-bit 编程流程

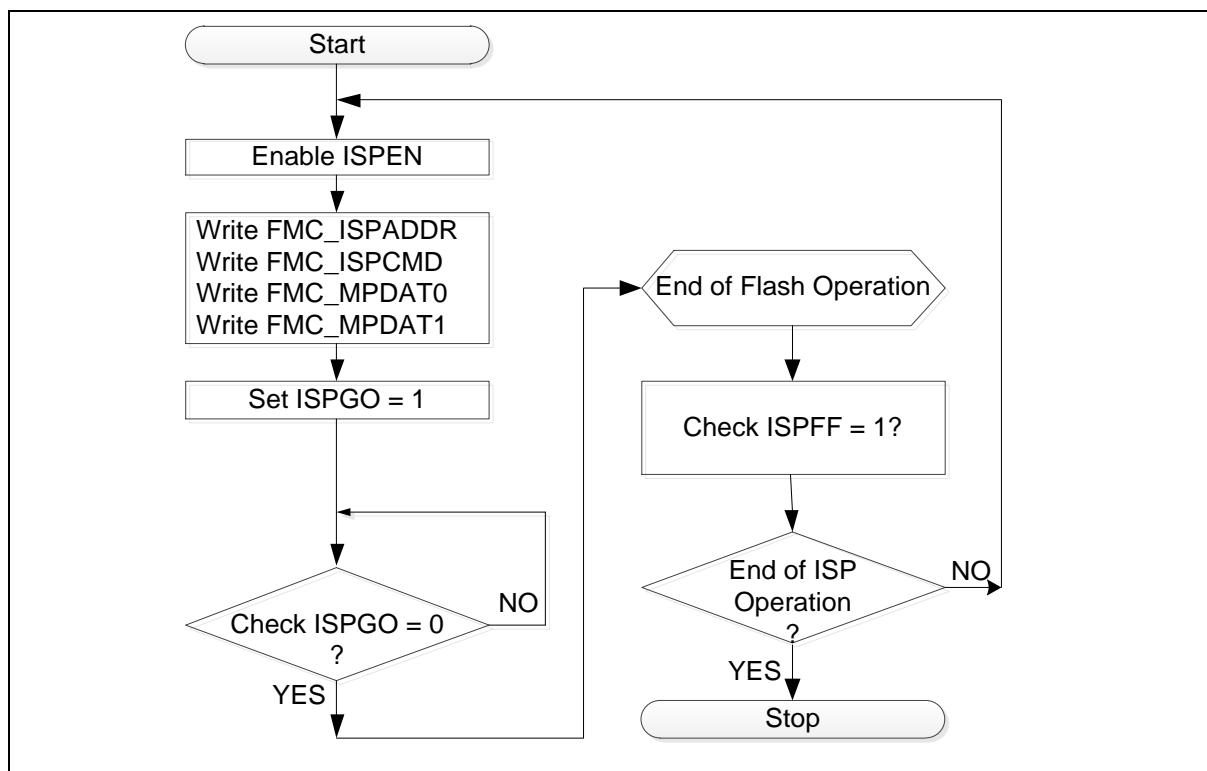


图 6.4-15 ISP 64-bit 编程流程

### 多字编程

NuMicro™ M480 支持多字编程功能来加速flash的更新速度。最大的编程长度是512字节，最小的编程长度是8字节。如果程序大于8字节，多字编程是最快的编程方式，因为每次操作只需一次flash建立时间与保持时间。下图比较了每种编程的时间。

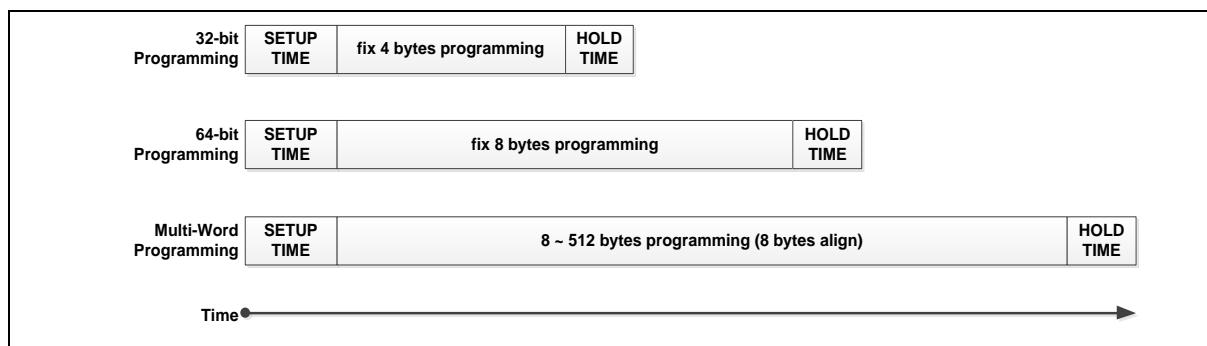


图 6.4-16 多字编程时间

在多字编程操作中，Cortex®-M4需要监测编程缓冲器的空闲状态。CPU需要持续准备下一次的编程数据。多次编程固件(firmware)不可以放在APROM或者LDROM中，因为CPU取指令不能停止。固件(firmware)应存放在Boot loader或片上SRAM中，用于避免CPU停止。

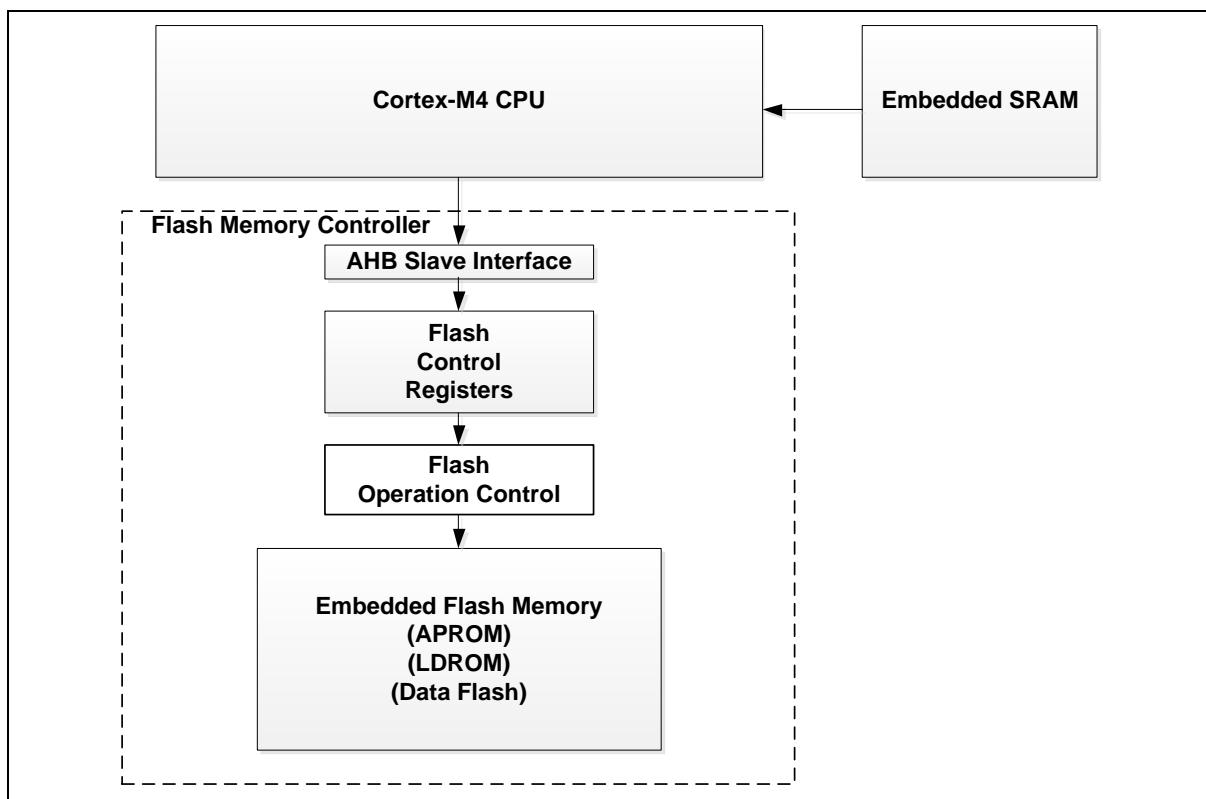


图 6.4-17 固件(firmware)在 SRAM 中的多字编程

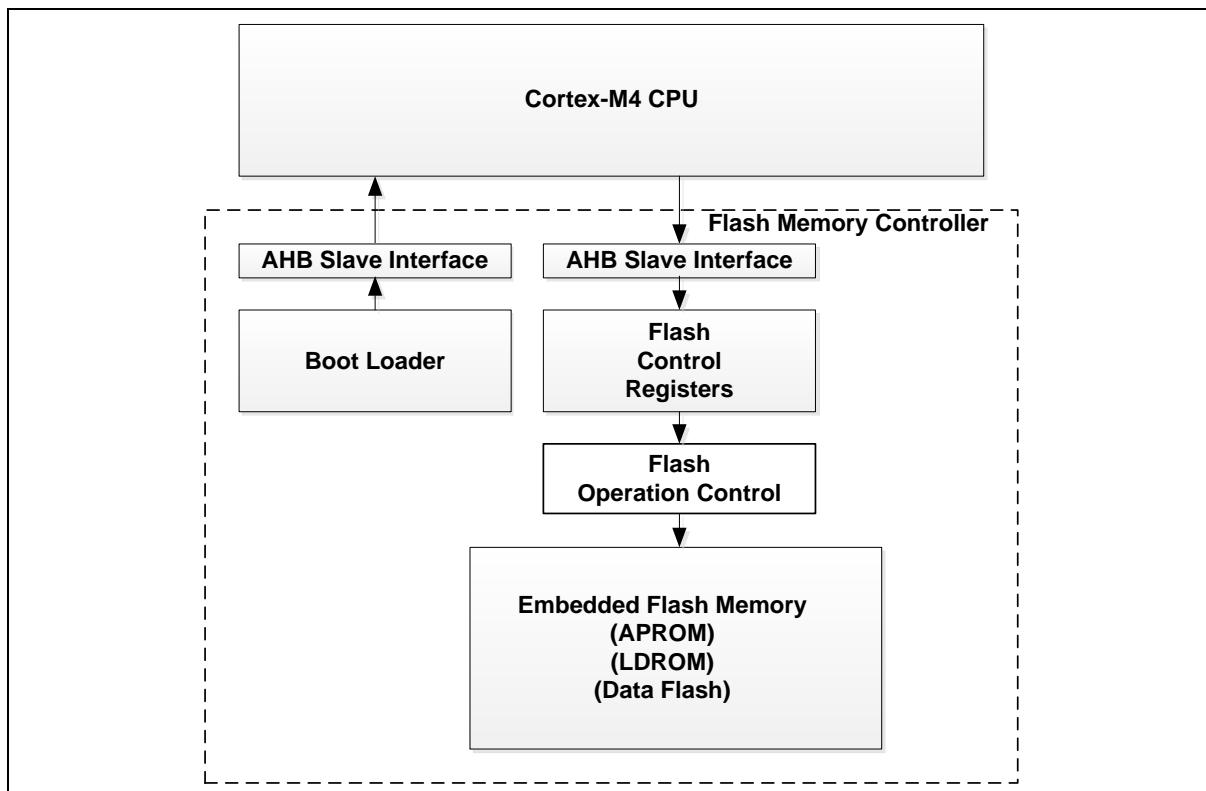


图 6.4-18 固件(firmware)在 Boot Loader 中的多字编程

多字编程的流程如下。ISP的起始地址(FMC\_ISPADDR)是8字节对齐的，FMC\_ISPADDR[2:0]应为0，ISPDAT0(FMC\_MPDATA0)是第一个数据字(偏移量 0x0), ISPDAT1(FMC\_MPDATA1)是第二个数据字(偏移量 0x4), ISPDAT2(FMC\_MPDATA2)是第三个数据字(偏移量 0x8), ISPDAT3(FMC\_MPDATA3)是第四个数据字(偏移量 0xC)。如果ISP起始地址FMC\_ISPADDR [3] =0, 第一个字数据应放在FMC\_MPDATA0, 第二个字数据应放在FMC\_MPDATA1, 第三个字数据应放在FMC\_MPDATA2, 第四个字数据应放在FMC\_MPDATA3。如果ISP起始地址FMC\_ISPADDR [3] =1, 第一个字数据应放在FMC\_MPDATA2, 第二个字数据应放在 FMC\_MPDATA3, 第三个字数据应放在 FMC\_MPDATA0, 第四个字数据应放在 FMC\_MPDATA1.最大的可编程大小是512字节，并以512字节地址对齐。当FMC控制器在多字编程操作，CPU需要监测缓冲器的状态D3~D0(FMC\_MPSTS[7:4])以及MPBUSY (FMC\_MPSTS[0])来等待缓冲器清空((D1,D0)=00 或 (D3,D2)=00)。然后CPU需要及时更新下一次的编程数据(FMC\_MPDATA0, FMC\_MPDATA1, FMC\_MPDATA2 和 FMC\_MPDATA3)。否则，FMC控制器将退出多字编程操作(MPBUSY (FMC\_MPSTS[0]) = 0). 如果CPU无法及时更新数据( MPBUSY (FMC\_MPSTS[0]) =0), CPU需要重新开始一个新的多字编程来延续流程，FMC\_MPADDR提供了上一次编程地址的信息。结尾操作，CPU必须检查ISPFF (FMC\_MPSTS[2])来确认多字操作是否已经成功的完成了。

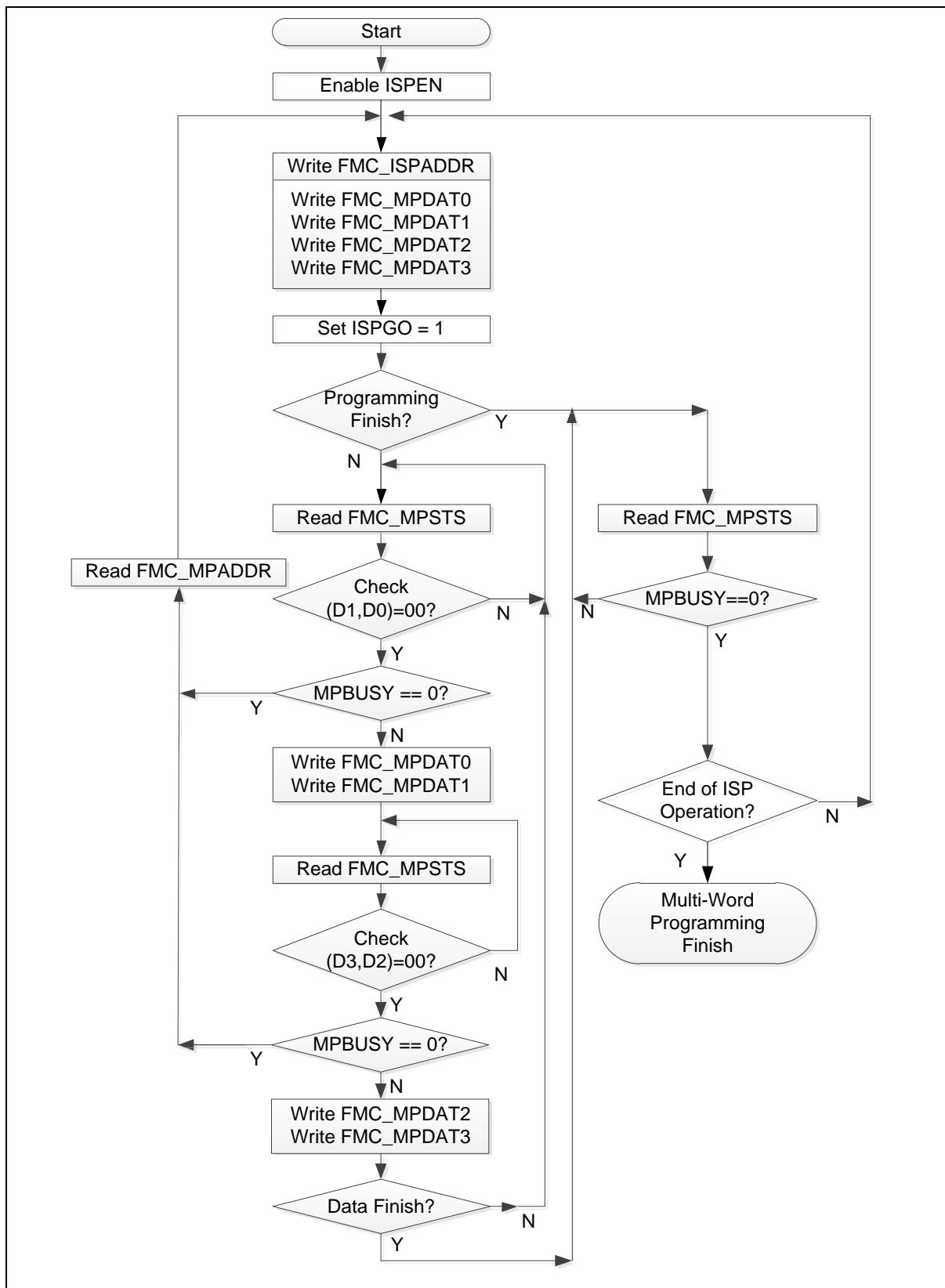


图 6.4-19 多字编程流程

## 6.4.4.6 快速Flash编程校验

在传统的flash编程操作，控制器接收到编程触发事件后控制片上flash编程时序，如下图所示。

NuMicro™ M480 提供快速flash编程验证功能，由硬件实现，可以节省CPU读回和比较的时间。当片上flash已经编程结束，硬件控制器访问flash做数据的读取和比较。最终，比较的结果会存在寄存器PGFF (FMC\_ISPSTS[5]。如果输入的编程数据与输出的数据不同，PGFF置1. 这个标志将保持，直到软件清0或有一个新的擦除操作。

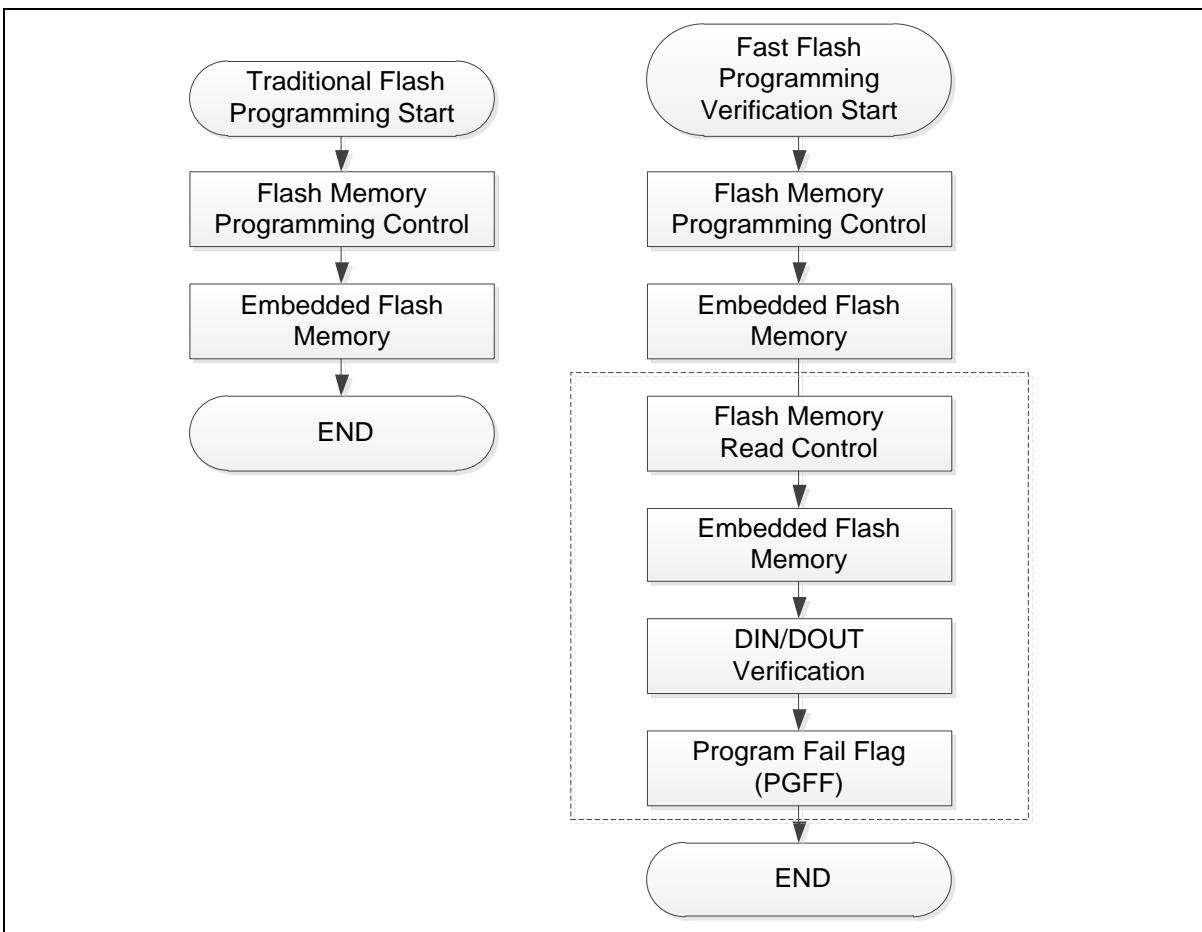


图 6.4-20 快速 Flash 编程校验流程

在传统的flash更新操作，flash存储器有三个步骤来完成flash存储器更新流程，(1) Flash 擦除 (2) Flash 编程 (3) Flash 读回所有数据校验。NuMicro™ M480 系列，只读 FMC\_ISPSTS 来检查步骤(3) 的PGFF 标志，不做数据的读回确认。下图对比了传统编程校验流程和快速编程校验流程。

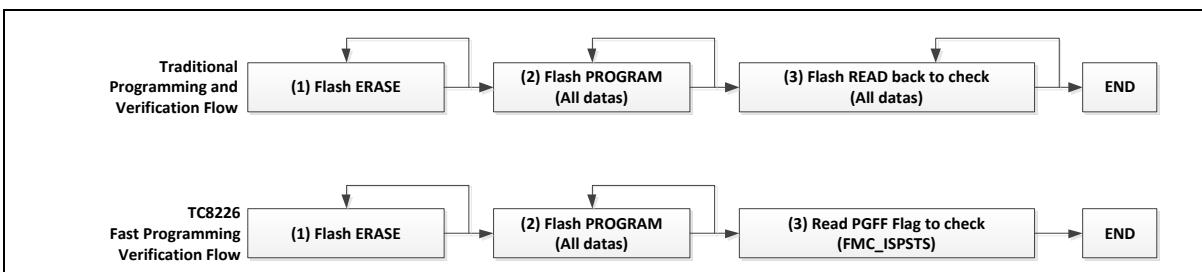


图 6.4-21 校验流程

快速的flash校验功能，用于32-位编程和64-位编程操作，但是不适合多-字编程操作，原因是多-字编程操作是片上flash HV (高电压)持续编程。

#### 6.4.4.7 CRC32 Checksum计算

NuMicro™ M480支持CRC32 checksum计算功能来帮助用户迅速检查APROM, LDROM, SPROM 和 Boot Loader内容。CRC32多项式是：

$$\text{CRC-32: } X^{32} + X^{26} + X^{23} + X^{22} + X^{16} + X^{12} + X^{11} + X^{10} + X^8 + X^7 + X^5 + X^4 + X^2 + X + 1$$

种子= 0xFFFF\_FFFF

CRC32 checksum 计算流程如下图。

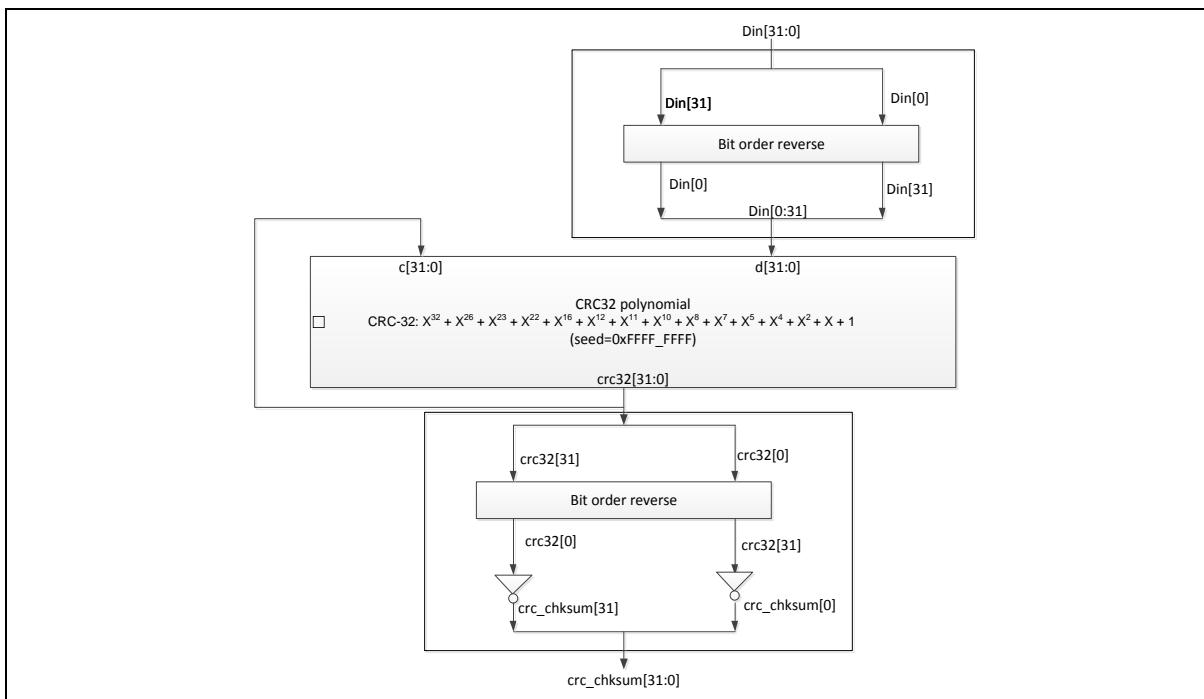


图 6.4-22 Flash CRC32 Checksum 计算

完成CRC32 checksum 计算有三个步骤.

1. “执行内存CRC32 Checksum”ISP 操作
1. “读内存CRC32Checksum”ISP 操作
2. 读取FMC\_ISPDAT寄存器得到checksum数据.

在步骤1， 用户需设置内存的起始地址(FMC\_ISPADDR)和大小(FMC\_ISPDAT)，两者都要4K字节对齐， 大小需 $\geq 4$  K字节， 起始地址适用于APROM, LDROM, SPROM 和 Boot Loader。

在步骤2， FMC\_ISPADDR需与步骤1保持一致。

在步骤3， checksum是从寄存器FMC\_ISPDAT读出。如果checksum是0x0000\_0000， 有两种可能性：

(1) 还在Checksum计算过程中, (2) 地址和大小超出了设备限制。

当 SPROM 被锁定为安全代码, 用户不能直接读SPROM内容, 用户可以使用 CRC32 checksum 功能来检查 SPROM 内容是否正确。

注意： CRC32 checksum 的范围在一次操作中不能跨bank。

#### 6.4.4.8 Flash 全为1校验

NuMicro™ M480 系列支持flash全为1校验功能来帮助用户快速检查APROM, LDROM, 和 SPROM在flash擦除操作后一个内存块内容是否为空白。

完成flash全为1的校验有两或三个步骤:

两步骤流程:

1. “运行 Flash 全为1校验” ISP操作
1. 读 ALLONE(FMC\_ISPSTA[7])位获取校验结果
  - ALLONE : 1, 校验存储区块中所有flash位都是1.
  - ALLONE : 0, 校验存储区块中所有flash位不全都是1.

三步骤流程:

1. “运行 Flash 全为1校验” ISP操作
2. 执行ISP “读 Flash 全为1结果” 操作
3. 读 FMC\_ISPDAT 获取校验结果
  - FMC\_ISPDATA : 0xA11F\_FFFF, 校验存储区块中所有flash位都是1.
  - FMC\_ISPDATA : 0xA110\_0000, 校验存储区块中所有flash位不全都是1

在步骤 1 中, 用户需设置内存的起始地址(FMC\_ISPADDR)和大小(FMC\_ISPDAT)用于校验, 两者都要4K字节对齐, 大小需 $\geq$  4 K字节, 起始地址适用于APROM, LDROM和SPROM。

在步骤 2 中, FMC\_ISPADDR 需与步骤1保持一致。

注意： 执行全为1校验 的范围在一次操作中不能跨bank。

#### 6.4.4.9 Flash 访问周期自动调整

NuMicro™ M480系列支持flash访问周期自动调整功能。当系统时钟 (HCLK) 改变, 用户不需要手动设置flash访问周期, 硬件会监测HCLK的频率且产生一个最优的周期数给flash控制器来获得最好的性能。

任何HCLK和分频器的寄存器有更新, 都是自动调整触发的事件, 包括HCLKSEL(CLK\_CLKSEL0), CLK\_PLLCTL, 和 HCLKDIV(CLK\_CLKDIV0)。当FADIS(FMC\_CYCCTL[8])置位, 且侦测到一个事件。FMC会暂时设置最大数 (也就是8)到寄存器CYCLE (FMC\_CYCCTL[3:0])使flash的访问不受时钟改变的影响。HIRC时钟对于自动调整产生一个用于HCLK计数的精确周期是必须的, 如果想禁止该功能, 用户可以设置FADIS(FMC\_CYCCTL[8])为0且确保FCYCDIS(FMC\_ISPSTS[4])被设置为1。侦测到的HCLK频率和最优的周期数如下表所示。

由于HIRC有误差(12 MHz  $\pm$  2%)且HCLK 和 HIRC的关系异步的, FMC可能在同一个HCLK时钟频率下产生两个可能的最优周期数。

HCLK时钟频率	最优周期数
0MHz ~27MHz	1
25MHz~52MHz	2
49MHz~79MHz	3
75MHz~107MHz	4
102MHz~134MHz	5
128MHz~162MHz	6
155MHz~193MHz	7
>185MHz	8

表 6.4-5 在自动调整功能下 Flash 访问最优周期数

flash 访问周期自动调整流程如下图所示

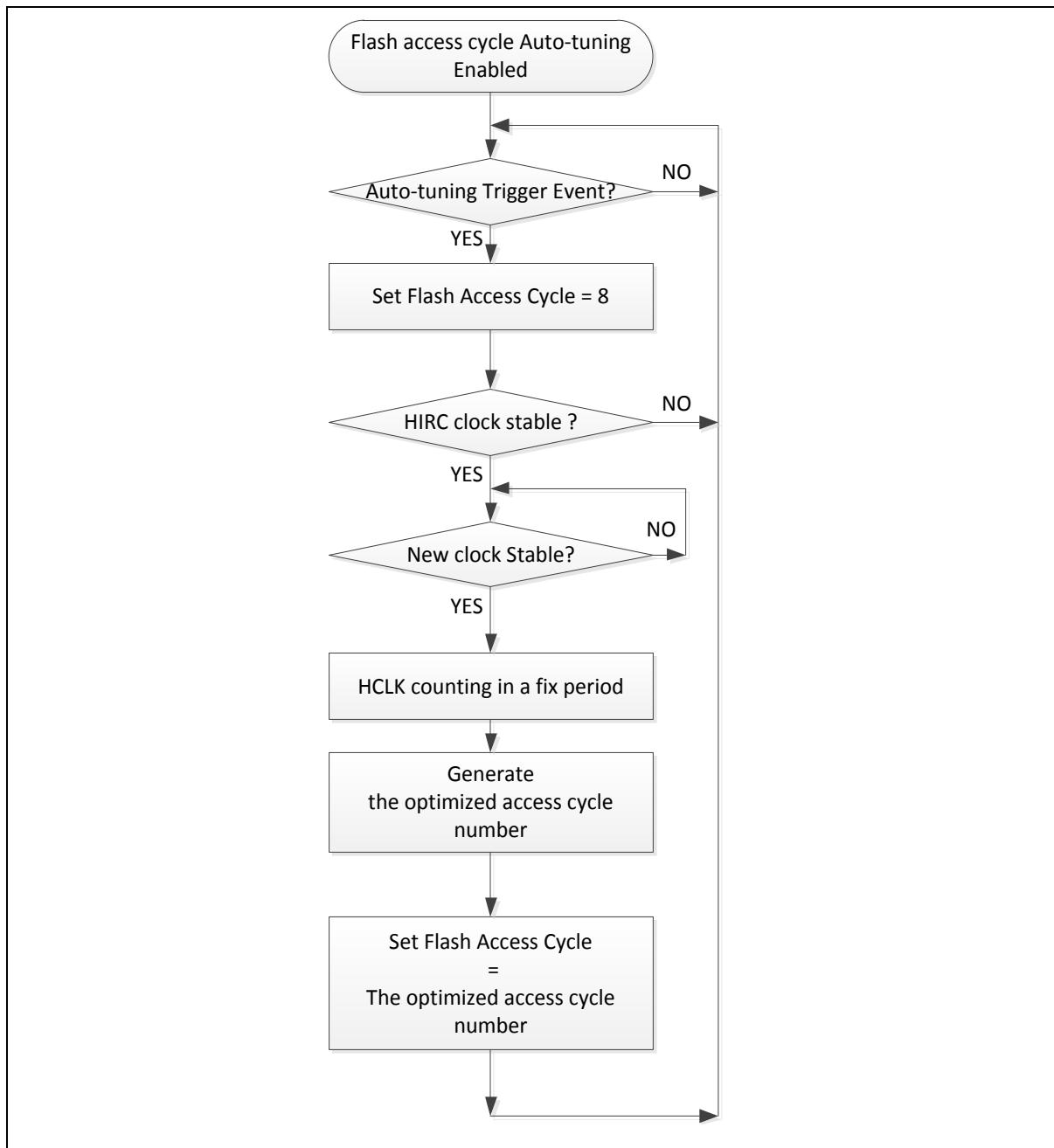


图 6.4-23 Flash 访问周期自动调整流程

#### 6.4.4.10 安全密匙保护

NuMicro™ M480 系列支持安全密匙保护功能来保护 APROM, LDROM, SPROM, CONFIG 和 KPROM 的数据用于编程和擦除的命令。当 KEYLOCK (FMC\_KPKEYSTS [1]) 激活, KPROM, LDROM 和 APROM (Data Flash 除外) 都在写保护状态。基于 KPKEYENROM[1:0] 的值, SPROM 和 CONFIG 的写保护是可选的。任何编程和页擦除操作都是非法的, 整个芯片批量擦除除外。如果 APROM, LDROM, SPROM, CONFIG 和 KPROM 需要修改, 必须先执行正确的密匙比较操作来清除 KEYLOCK

(FMC\_KPKEYSTS [1])的状态。当flash数据被KEYLOCK锁死，用户可以在锁定效果表中查找FMC的锁定效果。

### 安全密匙设置流程

安全密匙设置流程如下图所示，所有安全密匙信息存储在KPROM。首先KPROM (8KB)必须执行擦除操作来禁止安全密匙，然后写三个安全密匙到KPKEY0ROM, KPKEY1ROM 和 KPKEY2ROM。继续写KPMAXROM 和 KPKEMAXROM来定义安全密匙比较重试的限制次数。如果安全密匙比较不匹配，KPMAXROM用来限制上电的次数。KPKEMAXROM用来限制每次上电密匙不匹配操作的次数。最终，写一个非0xFF的值到KPKEYENROM来使能安全密匙保护功能。KPROM数据是不可读的，数据不能被读回检查。检查安全密匙的唯一方法是执行密匙比较操作。

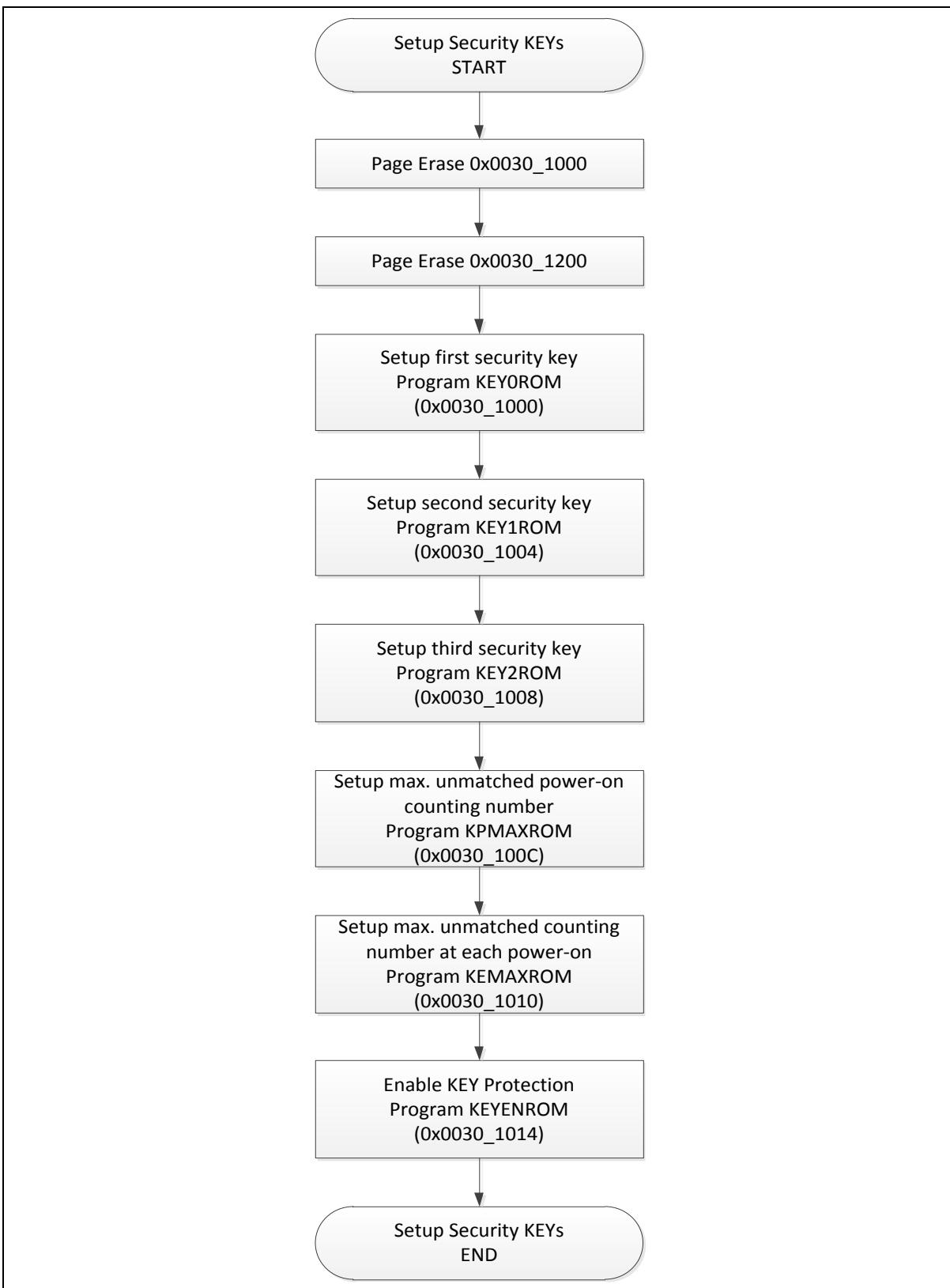


图 6.4-24 Flash 安全密匙设置流程

### 密匙比较操作流程

密钥比较功能用来验证KPROM里的用户密钥和安全密钥。具体操作见下图。

1. 向寄存器FMC\_KPKEY0/FMC\_KPKEY1/FMC\_KPKEY2写入用户密钥
2. 置KPKEYGO (FMC\_KPKEYTRG [0])位为1开始密钥比较操作
3. FMC控制器读取KPROM里的安全密钥并与用户密钥作对比

若密钥匹配，不匹配计数寄存器(FMC\_KPKECNT)会清零，KPROM的KPCNTROM页会被擦除。

若密钥不匹配，不匹配计数寄存器(FMC\_KPKECNT 和 FMC\_KPCNT)加1，KPROM的KPCNTROM页会被写0。

4. 硬件更新状态标识和比较结果到FMC\_KPKEYSTS寄存器供用户读取

如果密钥匹配，KEYMATCH(FMC\_KPKEYSTS [2])位置1，KEYLOCK(FMC\_KPKEYSTS [1])清零。KEYLOCK会保持0大概10~30分钟。当KEYFLAG(FMC\_KPKEYSTS [4])激活时，KEYLOCK会自动返回1。

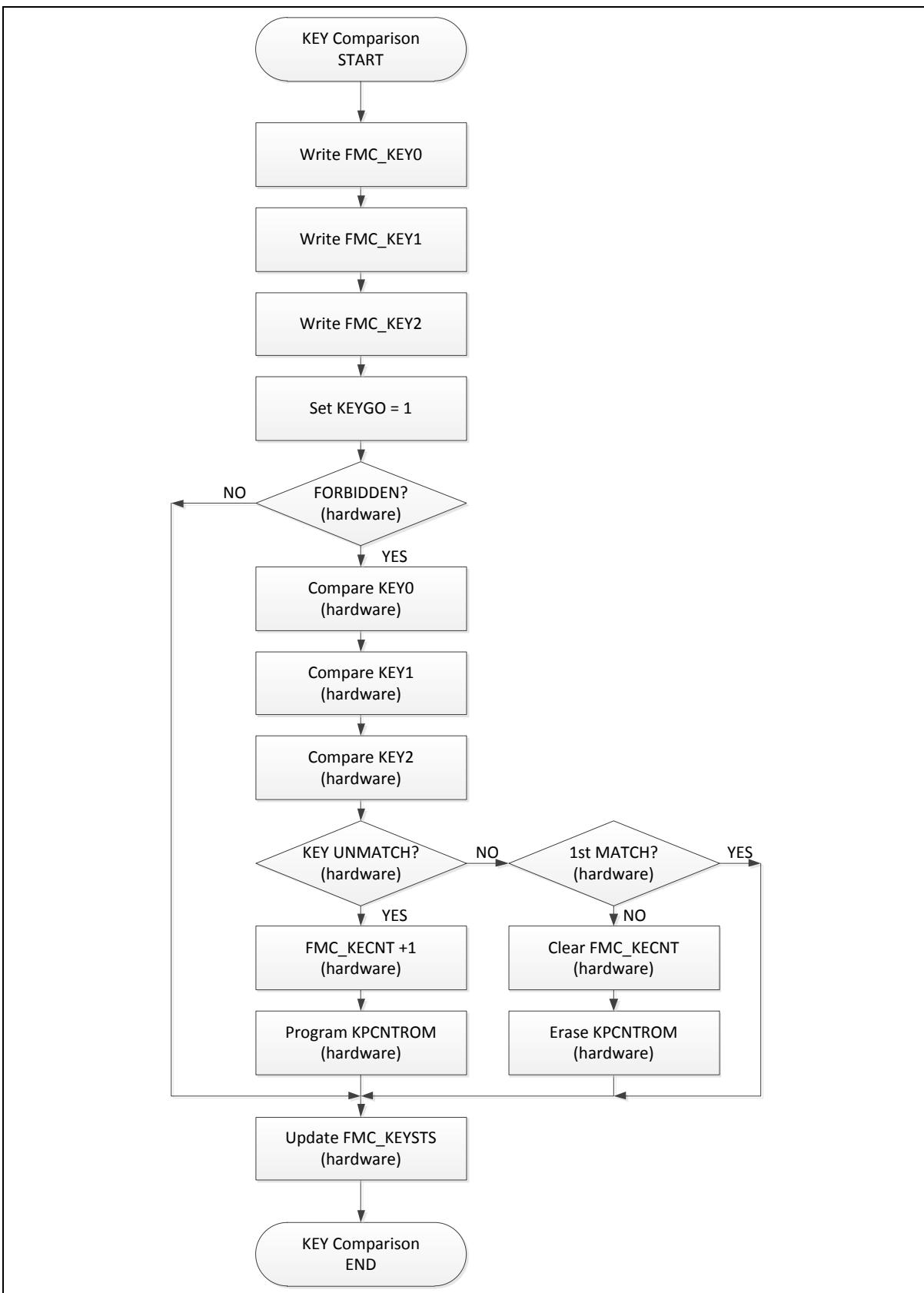


图 6.4-25 KEY 比较流程

## 6.4.4.11 锁定效果表

NuMicroTM M480支持四种保护，包括安全锁控制（也就是 CONFIG0[1]中的LOCK 和CONFIG2[7:0]中的ALOCK）、安全密匙保护功能（也就是 KPROM 和 KEYLOCK）、安全启动（也就是 CONFIG2[15:8]中的SBLOCK）和 安全保护存储（也就是 SPROM 和 SPLOCK）。在这个章节M480准备了一些锁定效果表，帮助用户理解在APROM, LDROM, SPROM CONFIG和其他以上四种对CPU, ICE 和 ICP/Writer的保护的锁定效果。

LOCK/ALOCK	OFF		ON		OFF		OFF		OFF					
KEYLOCK	OFF		OFF		OFF		OFF		ON					
KPKEYENROM[1:0]	X		X		X		X		00	01	10			
SBLOCK	OFF		OFF		ON		OFF		OFF					
SPLOCK	OFF		OFF		OFF		ON		OFF					
APROM	R	W	R	W	R	W	R	W	R	-	R	-	R	-
Data Flash	R	W	R	W	R	W	R	W	R	W	R	W	R	W
LDROM	R	W	R	W	R	W	R	W	R	-	R	-	R	-
SPROM	R	W	R	W	R	W	R <sub>1</sub>	W <sub>1</sub>	R	-	R	-	R	W
CFG	R	W	R	W	R	W	R	W	R	-	R	W	R	-
KPROM	-	W	-	W	-	W	-	W	-	-	-	-	-	-
OTP	R	W <sub>2</sub>	R	W <sub>2</sub>	R	W <sub>2</sub>	R	W <sub>2</sub>	R	W <sub>2</sub>	R	W <sub>2</sub>	R	W <sub>2</sub>
Secure Boot Key	-	W	-	W	-	W	-	W	-	W	-	W	-	W
Boot Loader	R	-	R	-	R	-	R	-	R	-	R	-	R	-
DID/UID/Checksum	R	-	R	-	R	-	R	-	R	-	R	-	R	-

表 6.4-6 对 CPU 的四种保护的锁定效果表

LOCK/ALOCK	OFF		ON		OFF		OFF		OFF					
KEYLOCK	OFF		OFF		OFF		OFF		ON					
KPKEYENROM[1:0]	X		X		X		X		00	01	10			
SBLOCK	OFF		OFF		ON		OFF		OFF					
SPLOCK	OFF		OFF		OFF		ON		OFF					
APROM	R	W	-	-	-	-	R	W	R	-	R	-	R	-
Data Flash	R	W	-	-	-	-	R	W	R	W	R	W	R	W
LDROM	R	W	-	-	-	-	R	W	R	-	R	-	R	-
SPROM	R	W	-	-	-	-	D	-	R	-	R	-	R	W
CFG	R	W	R	-	R	-	R	W	R	-	R	W	R	-
OTP	R	W <sub>2</sub>	-	-	-	-	R	W <sub>2</sub>	R	W <sub>2</sub>	R	W <sub>2</sub>	R	W <sub>2</sub>
KPROM	-	W	-	-	-	-	-	W	-	-	-	-	-	-
Secure Boot Key	-	W	-	-	-	-	-	W	-	W	-	W	-	W

Boot Loader	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
DID/UID/Checksum	R	-	R	-	R	-	R	-	R	-	R	-	R	-	R

表 6.4-7 对 ICE/ ICP/Writer 的四种保护的锁定效果表

注意：

1. 符号“R”意思是可读，符号“W”意思是可写。
2. 符号“X”意思是不管值是多少。
3. 符号“R<sub>1</sub>”意思是 SPROM 对于 CPU 指令可读，但是对于 CPU 数据或 ISP 命令不可读。符号“W<sub>1</sub>”意思是 SPROM 不能编程，但是可以擦除。符号“D”意思是 SPROM 可以在调试模式用于 ICE (不可读，返回 0x0000\_0000)。
4. 符号“W<sub>2</sub>”意思是 OTP 仅当对应的“LOCK BIT”是 0xFFFF\_FFFF 时可写。
5. SBLOCK = LOCK/ALOCK 对于 CPU 和 ICP/Writer。
6. 当 SBLOCK 不是 0x5A，对于 ICE 所有操作 (除了批量擦除) 是无效的。
7. 批量擦除(也就是整个芯片擦除)可以在任何时间任何地址执行，SPROM 除外。
8. 当任何保护与另外一种保护冲突，FMC 会采用最严格的一个保护。例如，当 KEYLOCK 打开，对于 ICE APROM 是可读的，一旦 KEYLOCK 和 LOCK 都打开对于 ICE APROM 是不可读的。
9. 对于 CPU, ICE 和 ICP/Writer, KPROM 和安全启动密匙是不可读的。
10. CE 和 ICP/Writer, Boot loader 是不可读的。

#### 6.4.5 寄存器映射

R: 只读, W: 只写, R/W: 读/写

寄存器	偏移量	R/W	描述	复位值
<b>FMC 基地址</b>				
<b>FMC_BA = 0x4000_C000</b>				
<b>FMC_ISPCTL</b>	FMC_BA+0x00	R/W	ISP 控制寄存器	0x0000_0000
<b>FMC_ISPADDR</b>	FMC_BA+0x04	R/W	ISP 地址寄存器	0x0000_0000
<b>FMC_ISPDATA</b>	FMC_BA+0x08	R/W	ISP 数据寄存器	0x0000_0000
<b>FMC_ISPCMD</b>	FMC_BA+0x0C	R/W	ISP 命令寄存器	0x0000_0000
<b>FMC_ISPTRG</b>	FMC_BA+0x10	R/W	ISP 触发控制寄存器	0x0000_0000
<b>FMC_DFBA</b>	FMC_BA+0x14	R	Data Flash 基地址	0xXXXX_XXXX
<b>FMC_ISPSTS</b>	FMC_BA+0x40	R/W	ISP 状态寄存器	0x0000_0000
<b>FMC_CYCCTL</b>	FMC_BA+0x4C	R/W	Flash 访问周期控制寄存器	0x0000_0000
<b>FMC_KPKEY0</b>	FMC_BA+0x50	W	KPROM KEY0 数据寄存器	0x0000_0000
<b>FMC_KPKEY1</b>	FMC_BA+0x54	W	KPROM KEY1数据寄存器	0x0000_0000
<b>FMC_KPKEY2</b>	FMC_BA+0x58	W	KPROM KEY2数据寄存器	0x0000_0000
<b>FMC_KPKEYTRG</b>	FMC_BA+0x5C	R/W	KPROM密钥比较触发控制寄存器	0x0000_0000
<b>FMC_KPKEYSTS</b>	FMC_BA+0x60	R/W	KPROM密钥比较状态寄存器	0x0000_0000
<b>FMC_KPKEYCNT</b>	FMC_BA+0x64	R	KPROM密钥不匹配计数寄存器	0x0000_XX00
<b>FMC_KPCNT</b>	FMC_BA+0x68	R	KPROM密钥不匹配上电计数寄存器	0x0000_0X00
<b>FMC_MPDATA0</b>	FMC_BA+0x80	R/W	ISP Data0 寄存器	0x0000_0000
<b>FMC_MPDATA1</b>	FMC_BA+0x84	R/W	ISP Data1寄存器	0x0000_0000
<b>FMC_MPDATA2</b>	FMC_BA+0x88	R/W	ISP Data2寄存器	0x0000_0000
<b>FMC_MPDATA3</b>	FMC_BA+0x8C	R/W	ISP Data3寄存器	0x0000_0000
<b>FMC_MPSTS</b>	FMC_BA+0xC0	R	ISP 多字编程状态寄存器	0x0000_0000
<b>FMC_MPADDR</b>	FMC_BA+0xC4	R	ISP多字编程地址寄存器	0x0000_0000

#### 6.4.6 寄存器描述

##### FMC\_ISPCTL ISP 控制寄存器

寄存器	偏移量	R/W	描述	复位值
-----	-----	-----	----	-----

FMC_ISPCTL	FMC_BA+0x00	R/W	ISP 控制寄存器	0x0000_0000			
------------	-------------	-----	-----------	-------------	--	--	--

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved	ISPFF	LDUEN	CFGUEN	APUEN	SPUEN	BS	ISPEN

位	描述	
[31:17]	Reserved	保留.
[16]	BL	<p><b>Boot Loader 启动 (写保护)</b></p> <p>该位被初始化 为MBS (CONFIG0[5])的取反值。除了CPU 复位或系统复位，任何复位BL都将重载初值。该位用于检查芯片是否从 Boot Loader 启动。当更新FMC_ISPCTL寄存器的时候，用户应该保持该位的初始值。</p> <p>0 = 从 APROM 或 LDROM启动. 1 = 从 Boot Loader.启动 <b>注:</b>该位写保护，参见SYS_REGLCTL寄存器</p>
[15]	Reserved	保留.
[14:12]	Reserved	保留
[11]	Reserved	保留
[10:8]	Reserved	保留.
[7]	Reserved	保留

[6]	<b>ISPFF</b>	<p><b>ISP失败标识(写保护)</b></p> <p>当已触发的ISP符合以下条件之一，该由硬件位置1: 该位写1清0.</p> <ul style="list-style-type: none"> <li>(1) APUEN为0时，APROM写APROM</li> <li>(2) LDUEN为0时，LDROM写LDROM</li> <li>(3) CFGUEN为0时，CONFIG被擦写或编程</li> <li>(4) SPUEN为0时，SPROM被擦写或编程</li> <li>(5) SPROM安全模式下，SPROM被编程</li> <li>(6) ICE连接下的锁定模式，进行页擦除</li> <li>(7) 欠压检测到低电压时，擦除或编程</li> <li>(8) 目标地址非法，例如超出有效范围</li> <li>(9) 无效的ISP命令</li> <li>(10) 向量表地址映射到了SPROM区域</li> <li>(11) KEYLOCK为1时，KROM被擦写或编程</li> <li>(12) KEYLOCK为1时，APROM(不含Data Flash) 被擦写或编程</li> <li>(13) KEYLOCK为1时，LDROM被擦写或编程</li> <li>(14) KEYLOCK 为 1 且 KEYENROM[1:0] 为1时，SPROM被擦写或编程</li> <li>(15) KEYLOCK 为 1 且 KEYENROM[1:0] 为1时，CONFIG被擦写或编程</li> <li>(16) SBLOCK 不是 0x5A时，ICE连接状态下的无效操作 (整片擦除外)</li> <li>(17) ICE连接状态下，读boot loader的任何内容。</li> </ul> <p>注:该位写保护，参见SYS_REGLCTL寄存器</p>
[5]	<b>LDUEN</b>	<p><b>LDROM 更新使能位 (写保护)</b></p> <p>0 = LDROM 不能被更新. 1 = LDROM 可以被更新.</p> <p>注:该位写保护，参见SYS_REGLCTL寄存器</p>
[4]	<b>CFGUEN</b>	<p><b>CONFIG更新使能位(写保护)</b></p> <p>0 = CONFIG 不能被更新 1 = CONFIG可以被更新</p> <p>注:该位写保护，参见寄存器SYS_REGLCTL</p>
[3]	<b>APUEN</b>	<p><b>APROM更新使能位(写保护)</b></p> <p>0 = 当芯片运行在APROM时，APROM不能被更新 1 = 当芯片运行在APROM时，APROM可以被更新</p> <p>注:该位写保护，参见寄存器SYS_REGLCTL</p>
[2]	<b>SPUEN</b>	<p><b>SPROM 更新使能位(写保护)</b></p> <p>0 = SPROM不能被更新 1 = SPROM可以被更新</p> <p>注:该位写保护，参见寄存器SYS_REGLCTL</p>

[1]	<b>BS</b>	<b>启动选择(写保护)</b> 置位/清零该位选择下次由LDROM/APROM启动。 该位也可以作为MCU启动状态标识位，用来检查芯片启动源。 除了CPU复位和系统复位，任何其他复位后，该位都会被初始化为寄存器CBS[1](CONFIG0[7])的反转值 0 = 当 MBS (CONFIG0[5]) 是 1，从APROM启动 1 = 当 MBS (CONFIG0[5]) 是 1，从LDROM启动 <b>注:</b> 该位写保护，参见寄存器SYS_REGLCTL
[0]	<b>ISPEN</b>	<b>ISP使能位(写保护)</b> ISP使能位。设置该位使能ISP功能。 0 = ISP功能禁止 1 = ISP功能使能 <b>注:</b> 该位写保护，参见寄存器SYS_REGLCTL

FMC\_ISPADDR ISP 地址

寄存器	偏移量	R/W	描述	复位值
FMC_ISPADDR	FMC_BA+0x04	R/W	ISP 地址寄存器	0x0000_0000

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

位	描述	
[31:0]	ISPADDR	<p><b>ISP 地址</b></p> <p>NuMicro™ M480 系列内嵌FLASH, ISP 32-位操作时, ISPADDR[1:0]必须为00 (字对齐 (4-字节).) ; ISP 64-位操作时, ISPADDR[2:0]必须为000 (双字对齐 (8-字节).) .</p> <p>对于CRC32 Checksum计算命令, 该地址是flash checksum计算的起始地址, 需要4K字节对齐。</p>

**FMC\_ISPDAT ISP 数据寄存器**

寄存器	偏移量	R/W	描述	复位值
FMC_ISPDAT	FMC_BA+0x08	R/W	ISP 数据寄存器	0x0000_0000

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

位	描述	
[31:0]	ISPDAT	<p><b>ISP 数据</b></p> <p>ISP写操作之前，写数据到该寄存器</p> <p>ISP读操作后，可从该寄存器读数据</p> <p>当 ISPFF (FMC_ISPCTL[6]) 是 1, ISPDAT = 0xffff_ffff</p> <p>执行CRC32 Checksum计算时，ISPDAT代表内存大小(字节)，4 K字节对齐。</p> <p>读ISP Checksum命令时，ISPDAT代表checksum的计算结果。</p> <p>ISPDAT = 0x0000_0000, 代表(1) 还在checksum计算中，或(2) checksum 计算的内存范围出错。</p>

**FMC\_ISPCMD ISP 命令寄存器**

寄存器	偏移量	R/W	描述	复位值
FMC_ISPCMD	FMC_BA+0x0C	R/W	ISP 命令寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved	CMD						

位	描述	
[31:7]	Reserved	保留
[6:0]	CMD	<p><b>ISP 命令</b>          命令表如下:</p> <p>0x00=读FLASH.          0x04= 读 UID.          0x08= 读 Flash 位全为1校验结果.          0x0B=读公司 ID.          0x0C=读设备ID.          0x0D=读Checksum.          0x21= FLASH 32-位编程          0x22= FLASH 页擦除., 擦除两个bank中的任意页, OTP除外。          0x23= FLASH Bank 擦除, 擦除BANK0 或 BANK1中APROM的所有页          0x25= FLASH 块擦除, 擦除BANK0 或 BANK1中APROM的四页 (四页对齐)          0x27= FLASH 多-字编程.          0x28= 执行 Flash 位全为1校验          0x2D= 执行 Checksum 计算          0x2E= 向量重映射          0x40= FLASH 64-位 读.          0x61= FLASH 64-位编程          其他是无效指令</p>

FMC\_ISPTRG ISP 触发控制寄存器

寄存器	偏移量	R/W	描述				复位值
FMC_ISPTRG	FMC_BA+0x10	R/W	ISP 触发控制寄存器				0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved							ISPGO

位	描述	
[31:1]	Reserved	保留.
[0]	ISPGO	<p><b>启动ISP触发寄存器（写保护）</b>            写 1 开始ISP操作，当ISP操作结束后，该位由硬件自动清零。            0 = ISP 操作结束            1 = ISP 正在执行  <b>注：</b> 该位是写保护位。参考SYS_REGLCTL寄存器。</p>

**FMC\_DFBA Data Flash起始地址寄存器**

寄存器	偏移量	R/W	描述	复位值
FMC_DFBA	FMC_BA+0x14	R	Data Flash 起始地址	0xFFFF_FFFF

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

位	描述
[31:0]	<b>DFBA</b> <b>数据FLASH起始地址</b> 该寄存器为数据FLASH开始地址寄存器，只读。 数据flash与APROM共用存储空间。从CONFIG1加载寄存器内容。 当DFEN (CONFIG0[0]) =0该寄存器有效。

**FMC\_ISPSTS ISP 状态寄存器**

寄存器	偏移量	R/W	描述	复位值
FMC_ISPSTS	FMC_BA+0x40	R/W	ISP 状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
SCODE	Reserved						
23	22	21	20	19	18	17	16
VECMAP							
15	14	13	12	11	10	9	8
VECMAP							Reserved
7	6	5	4	3	2	1	0
ALLONE	ISPFF	PGFF	FCYCDIS	MBS	CBS		ISPBUSY

位	描述
[31]	<b>SCODE</b> 安全代码上锁标志 当在flash初始化的时候侦测到SPROM的安全代码是上锁的，该位由硬件置1。或者软件写1到该位使安全代码上锁，该位由SPROM页擦除操作清0。 0 = 安全代码未上锁。 1 = 安全代码上锁。
[30:24]	Reserved 保留。
[23:9]	<b>VECMAP</b> 向量页映射地址（只读） 当前 flash 地址空间0x0000_0000~0x0000_01FF映射到地址{VECMAP[14:0], 9'h000} ~ {VECMAP[14:0], 9'h1FF}
[8]	Reserved 保留
[7]	<b>ALLONE</b> Flash 全为1校验标志 在执行flash位全为1校验后，如果flash位全为1，该位由硬件置1，如果不是全为1，该位由硬件清0。该位也可以通过写1清0。 0 =在执行flash位全为1校验后，所有flash位不全为1 1 =在执行flash位全为1校验后，所有flash位全为1

[6]	<b>ISPFF</b>	<p><b>ISP失败标识(写保护)</b></p> <p>该位是 ISPFF (FMC_ISPCTL[6])的镜像，需要写1到FMC_ISPCTL[6] 或 FMC_ISPSTS[6]来清0。当已触发的ISP符合以下条件之一，该由硬件位置1：</p> <ul style="list-style-type: none"> <li>(1) APUEN为0时，APROM写APROM</li> <li>(2) LDUEN为0时，LDROM写LDROM</li> <li>(3) CFGUEN为0时，CONFIG被擦写或编程</li> <li>(4) SPUEN为0时，SPROM被擦写或编程</li> <li>(5) SPROM安全模式下，SPROM被编程</li> <li>(6) ICE连接下的锁定模式，进行页擦除</li> <li>(7) 欠压检测到低电压时，擦除或编程</li> <li>(8) 目标地址非法，例如超出有效范围</li> <li>(9) 无效的ISP命令</li> <li>(10) 向量表地址映射到了SPROM区域</li> <li>(11) KEYLOCK为1时，KROM被擦写或编程</li> <li>(12) KEYLOCK为1时，APROM(不包含Data Flash) 被擦写或编程</li> <li>(13) KEYLOCK为1时，LDROM被擦写或编程</li> <li>(14) KEYLOCK 为 1 且 KEYENROM[1:0] 为1时，SPROM被擦写或编程</li> <li>(15) KEYLOCK 为 1 且 KEYENROM[1:0] 为1时，CONFIG被擦写或编程</li> <li>(16) SBLOCK 不是 0x5A时，ICE连接状态下的无效操作 (整片擦除除外)</li> <li>(17) ICE连接状态下，读boot loader的任何内容。</li> </ul> <p>注：该位写保护，参见SYS_REGLCTL寄存器</p>
[5]	<b>PGFF</b>	<p><b>快速校验标志 (只读)</b></p> <p>如果ISP校验数据不匹配时，该位置1。当执行ISP flash擦除或读CID操作时，该位被清零。</p> <p>0 = Flash 编程成功 1 = Flash编程失败。编程数据与flash内存的数据不同。</p>
[4]	<b>FCYCDIS</b>	<p><b>Flash 访问周期自动调整禁止标志 (只读)</b></p> <p>如果 flash 访问周期自动调整功能禁止，该位置1。自动调整功能通过FADIS(FMC_CYCCTL[8])位禁止或HIRC时钟没有稳定禁止。</p> <p>0 = Flash访问周期自动调整使能。 1 = Flash访问周期自动调整禁止</p>
[3]	<b>MBS</b>	<p><b>从Boot Loader启动的标志 (只读)</b></p> <p>除了 CPU 复位(CPU= 1)及 system 复位 (SYS)外的任何复位发生时，该位由MBS (CONFIG0[5])初始化。</p> <p>0 = 从Boot Loader启动。 1 = 从LDROM/APROM启动 (参考CBS位设定)</p>
[2:1]	<b>CBS</b>	<p><b>启动选择配置 (只读)</b></p> <p>除了 CPU 复位 (CPU= 1)及系统 复位 (SYS)外的任何复位发生时，该位由 CBS (CONFIG0[7:6])初始化。</p> <p>当MBS (FMC_ISPSTS[3])= 1时，如下功能生效。</p> <p>00 = LDROM支持 IAP模式 01 = LDROM不支持 IAP模式 10 = APROM 支持IAP模式 11 = APROM 不支持IAP模式</p>

[0]	<b>ISPBUSY</b>	<b>ISP 忙标志位（只读）</b> 写1开始ISP流程，当ISP操作结束后，该位由硬件自动清零。 该位是ISPGO(FMC_ISPTRG[0]) 的镜像 0 = ISP 操作结束 1 = ISP 正在进行
-----	----------------	--

**FMC\_CYCCTL Flash 访问周期控制寄存器**

寄存器	偏移量	R/W	描述	复位值
FMC_CYCCTL	FMC_BA+0x4C	R/W	Flash 访问周期控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved				CYCLE			

位	描述	
[31:9]	Reserved	保留
[8]	FADIS	<p><b>Flash 访问周期自动调整禁止控制 (写保护)</b></p> <p>设置该位禁止flash访问周期自动调整功能</p> <p>0 = Flash访问周期自动调整使能 1 = Flash访问周期自动调整禁止.</p> <p>注:该位写保护, 参见SYS_REGLCTL寄存器</p>
[7:4]	Reserved	保留.

[3:0]	<b>CYCLE</b>	<p><b>Flash 访问周期控制 (写保护)</b></p> <p>当FCYCDIS (FMC_ISPSTS[4]) 是 0, 该寄存器由硬件自动更新, 当自动调整功能禁止 (FCYCDIS (FMC_ISPSTS[4])) 是 1, 该寄存器由软件更新。</p> <p>0000 = CPU 访问零等待, flash 访问周期是 1;.</p> <p>HCLK 工作频率范围是 &lt;27MHz, cache由硬件禁止.</p> <p>0001 =如果没用cache, CPU 访问等待1个周期; flash访问周期是1;.</p> <p>HCLK 工作频率范围是 &lt;27MHz</p> <p>0010 = 如果没用cache, CPU 访问等待2个周期; flash访问周期是2;.</p> <p>最优的 HCLK 工作 频率范围是27~54 MHz</p> <p>0011 =如果没用cache, CPU 访问等待3个周期; flash访问周期是3;.</p> <p>最优的 HCLK 工作 频率范围是 54~81MHz</p> <p>0100 =如果没用cache, CPU 访问等待4个周期; flash访问周期是4;.</p> <p>最优的 HCLK 工作 频率范围是81~108MHz</p> <p>0101 =如果没用cache, CPU 访问等待5个周期; flash访问周期是5;.</p> <p>最优的 HCLK 工作 频率范围是108~135MHz</p> <p>0110 =如果没用cache, CPU 访问等待6个周期; flash访问周期是6;.</p> <p>最优的 HCLK 工作 频率范围是135~162MHz</p> <p>0111 =如果没用cache, CPU 访问等待7个周期; flash访问周期是7;.</p> <p>最优的 HCLK 工作 频率范围是162~192MHz</p> <p>1000 =如果没用cache, CPU 访问等待8个周期; flash访问周期是8;.</p> <p>最优的 HCLK 工作 频率范围是&gt;192MHz</p> <p><b>注:</b>该位写保护, 参见SYS_REGLCTL寄存器</p>
-------	--------------	---

**FMC\_KPKEY0 KPROM KEY0 数据寄存器**

寄存器	偏移量	R/W	描述	复位值
FMC_KPKEY0	FMC_BA+0x50	W	KPROM KEY0 数据寄存器	0x0000_0000

31	30	29	28	27	26	25	24
KPKEY0							
23	22	21	20	19	18	17	16
KPKEY0							
15	14	13	12	11	10	9	8
KPKEY0							
7	6	5	4	3	2	1	0
KPKEY0							

位	描述	
[31:0]	KPKEY0	KPROM KEY0 数据 (只写) 在密匙比较操作之前，写 KPKEY0 数据到该寄存器

**FMC\_KPKEY1 KPROM KEY1数据寄存器**

寄存器	偏移量	R/W	描述	复位值
FMC_KPKEY1	FMC_BA+0x54	W	KPROM KEY1数据寄存器	0x0000_0000

31	30	29	28	27	26	25	24
KPKEY1							
23	22	21	20	19	18	17	16
KPKEY1							
15	14	13	12	11	10	9	8
KPKEY1							
7	6	5	4	3	2	1	0
KPKEY1							

位	描述	
[31:0]	KPKEY1	<b>KPROM KEY1 数据 (只写)</b> 在密匙比较操作之前，写 KPKEY1 数据到该寄存器

**FMC\_KPKEY2 KPROM KEY2数据寄存器**

寄存器	偏移量	R/W	描述	复位值
FMC_KPKEY2	FMC_BA+0x58	W	KPROM KEY2数据寄存器	0x0000_0000

31	30	29	28	27	26	25	24
KKPEY2							
23	22	21	20	19	18	17	16
KPKEY2							
15	14	13	12	11	10	9	8
KPKEY2							
7	6	5	4	3	2	1	0
KPKEY2							

位	描述	
[31:0]	KPKEY2	<b>KPROM KEY2 数据 (只写)</b> 在密匙比较操作之前，写 KPKEY2 数据到该寄存器

**FMC\_KPKEYTRG KPROM KEY 比较触发控制寄存器**

寄存器	偏移量	R/W	描述	复位值
<b>FMC_KPKEYTRG</b>	FMC_BA+0x5C	R/W	KPROM KEY 比较触发控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved						TCEN	KPKEYGO

位	描述	
[31:2]	<b>Reserved</b>	保留
[1]	<b>TCEN</b>	<p><b>超时计数使能 (写保护)</b>            0 = 超时计数禁止。            1 = 在密匙比较完成后，如果输入密匙匹配，超时计数使能。            超时时间至少10分钟，平均20分钟左右。</p> <p><b>注:</b>该位写保护，参见SYS_REGLCTL寄存器.</p>
[0]	<b>KPKEYGO</b>	<p><b>KPROM 密匙比较开始触发器 (写保护)</b>            写1开始密匙比较操作，当密匙比较操作完成，该位会由硬件自动清0。当FORBID(FMC_KPKEYSTS [3])为0，该触发器操作有效。</p> <p>0 = 密匙比较操作完成            1 = 密匙比较操作正在进行</p> <p><b>注:</b>该位写保护，参见SYS_REGLCTL寄存器</p>

**FMC\_KPKEYSTS KPROM 密匙 状态寄存器**

寄存器	偏移量	R/W	描述	复位值
FMC_KPKEYSTS	FMC_BA+0x60	R/W	KPROM 密匙比较状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved	SPFLAG	CFGFLAG	KEYFLAG	FORBID	KEYMATCH	KEYLOCK	KEYBUSY

位	描述	
[31:7]	Reserved	保留.
[6]	SPFLAG	<b>SPROM写保护使能位(只读)</b> KEYENROM[1]为0时，上电及复位会置位该位。擦写KPROM时，硬件会清0该位。 KEYENROM[1]写0时，该位由硬件置1. 0 = SPROM 写保护关闭 1 = SPROM 写保护使能
[5]	CFGFLAG	<b>CONFIG写保护使能位(只读)</b> KEYENROM[0]为0时，上电及复位会置位该位。擦写KPROM时，硬件会清0该位。 KEYENROM[0]写0时，该位由硬件置1. 0 = CONFIG写保护关闭 1 = CONFIG写保护使能
[4]	KEYFLAG	<b>密钥保护使能标志(只读)</b> 上电时或复位后，若KEYENROM[7:0]不为0xFF，该位置1位。KPROM被擦写时，该位由硬件自动清零。KEYENROM被写入非0xFF时，硬件自动置该位为1. 0 = 安全密匙保护关闭 1 = 安全密匙保护使能
[3]	FORBID	<b>密钥比较禁止标识(只读)</b> 当KPKECNT (FMC_KPKECNT [4:0])大于KPKEMAX (FMC_KPKEY0[12:8]) 或者 KPCNT (FMC_KPCNT [2:0])大于KPMAX(FMC_KPCNT[10:8])时，该位置1. 0 = 密钥比较未禁止 1 = 密钥比较禁止， KEYGO[FMC_KEYTRG[0]]不能触发。

[2]	<b>KEYMATCH</b>	<p><b>密钥比较标识(只读)</b></p> <p>密钥比较操作后, 如果KEY0、KEY1和KEY2同KPROM里的96位密钥相匹配, 该位置1。密钥不匹配时, 该位清零。</p> <p>如下情况时, 该位也会清零:</p> <ul style="list-style-type: none"> <li>● CPU写1到KEYLOCK(FMC_KPKEYSTS[1])</li> <li>● 超时事件</li> <li>● KPROM被擦除</li> <li>● KEYENROM写入非0xFF的值</li> <li>● 掉电模式下</li> </ul> <p>0 = KEY0、KEY1和KEY2与KPROM的设置不匹配 1 = KEY0、KEY1和KEY2与KPROM的设置相匹配</p>
[1]	<b>KEYLOCK</b>	<p><b>密钥锁定标识</b></p> <p>KEYMATCH(FMC_KEYSTS[2])为0时, 该位置1。安全密钥保护时KEYMATCH为1, 则清零该位。批量擦除后, 用户必须复位或上电/断电来清0该位。如下状况时, 该位置1:</p> <ul style="list-style-type: none"> <li>● CPU写1到KEYLOCK(FMC_KPKEYSTS[1])</li> <li>● 上电或复位时, KEYFLAG(FMC_KPKEYSTS[4])为1</li> <li>● KEYENROM写入非0xFF的值</li> <li>● 超时事件</li> <li>● FORBID(FMC_KPKEYSTS[3])为1</li> </ul> <p>0 = KPROM, LDROM和APROM(不包含Data Flash)不处于写保护状态 1 = KPROM, LDROM和APROM(不包含Data Flash)处于写保护状态 SPROM写保护取决于SPFLAG CONFIG写保护取决于CFGFLAG</p>
[0]	<b>KEYBUSY</b>	<p><b>密钥比较忙标识(只读)</b></p> <p>0 = 密钥比较结束 1 = 密钥比较进行中</p>

## FMC\_KPKEYCNT KPROM密钥不匹配计数寄存器

寄存器	偏移量	R/W	描述	复位值
FMC_KPKEYCNT	FMC_BA+0x64	R	KPROM密钥不匹配计数寄存器	0x0000_XX00

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved		KPKEMAX					
7	6	5	4	3	2	1	0
Reserved		KPKECNT					

位	描述	
[31:14]	Reserved	保留
[13:8]	KPKEMAX	<p>每次上电错误密钥尝试次数最大值(只读)</p> <p>KPKEMAX 是每次上电错误密钥尝试次数的最大值。当擦除或编程 KPROM 的 KPKEMAXROM , KPKEMAX 的值会被更新。KPKEMAX 用来限制 KPKECNT(FMC_KPKEY0[5:0]) 的最大计数值。KPKECNT 大于 KPKEMAX 时，FORBID(FMC_KPKEYSTS [3])会被置1。.</p>
[7:6]	Reserved	保留
[5:0]	KPKECNT	<p>每次上电密钥错误尝试次数(只读)</p> <p>安全密钥保护下，密钥错误则KPKECNT加一。系统上电或密钥匹配成功，KECNT清零。</p>

**FMC\_KPCNT KPROM密钥不匹配上电次数寄存器**

寄存器	偏移量	R/W	描述	复位值
FMC_KPCNT	FMC_BA+0x68	R	KPROM密钥不匹配上电次数寄存器	0x0000_0X00

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved				KPMAX			
7	6	5	4	3	2	1	0
Reserved				KPCNT			

位	描述	
[31:12]	Reserved	保留
[11:8]	KPMAX	<b>密钥尝试上电次数最大值(只读)</b> KPMAX是密钥尝试错误的上电次数最大值。当擦除或写KPROM的KPMaxROM， KPMax的值也会被更新。 KPMax用来限制KPCNT(FMC_KPCNT [3:0])的最大计数值。 KPCNT大于KPMax时， FORBID(FMC_KPKEYSTS [3])会被置1。
[7:4]	Reserved	保留.
[3:0]	KPCNT	<b>密钥错误上电次数(只读)</b> KPCNT是安全密钥保护下密钥错误的上电次数。密钥匹配成功， KPCNT清零。

**FMC\_MPDATA0 ISP 数据 0 寄存器**

寄存器	偏移量	R/W	描述	复位值
FMC_MPDATA0	FMC_BA+0x80	R/W	ISP 数据0 寄存器	0x0000_0000

31	30	29	28	27	26	25	24
ISPDAT0							
23	22	21	20	19	18	17	16
ISPDAT0							
15	14	13	12	11	10	9	8
ISPDAT0							
7	6	5	4	3	2	1	0
ISPDAT0							

位	描述	
[31:0]	<b>ISPDAT0</b>	ISP 数据 0 该寄存器是对于32位/64位/多字编程的第一个32位。也是FMC_ISPDAT的镜像寄存器，这两个寄存器保持一致。

**FMC\_MPDAT1 ISP 数据 1 寄存器**

寄存器	偏移量	R/W	描述	复位值
FMC_MPDAT1	FMC_BA+0x84	R/W	ISP 数据1 寄存器	0x0000_0000

31	30	29	28	27	26	25	24
ISPDAT1							
23	22	21	20	19	18	17	16
ISPDAT1							
15	14	13	12	11	10	9	8
ISPDAT1							
7	6	5	4	3	2	1	0
ISPDAT1							

位	描述	
[31:0]	ISPDAT1	ISP 数据 1 该寄存器是对于64位/多字编程的第二个32位。

**FMC\_MPDAT2 ISP 数据 2 寄存器**

寄存器	偏移量	R/W	描述	复位值
FMC_MPDAT2	FMC_BA+0x88	R/W	ISP 数据2 寄存器	0x0000_0000

31	30	29	28	27	26	25	24
ISPDAT2							
23	22	21	20	19	18	17	16
ISPDAT2							
15	14	13	12	11	10	9	8
ISPDAT2							
7	6	5	4	3	2	1	0
ISPDAT2							

位	描述		
[31:0]	ISPDAT2	ISP 数据 2	该寄存器是对于多字编程的第三个32位。

**FMC\_MPDAT3 ISP 数据 3 寄存器**

寄存器	偏移量	R/W	描述	复位值
FMC_MPDAT3	FMC_BA+0x8C	R/W	ISP 数据3 寄存器	0x0000_0000

31	30	29	28	27	26	25	24
ISPDAT3							
23	22	21	20	19	18	17	16
ISPDAT3							
15	14	13	12	11	10	9	8
ISPDAT3							
7	6	5	4	3	2	1	0
ISPDAT3							

位	描述		
[31:0]	ISPDAT3	ISP 数据 3	该寄存器是对于多字编程的第四个32位。

## FMC\_MPSTS ISP多字编程状态寄存器

寄存器	偏移量	R/W	描述	复位值
FMC_MPSTS	FMC_BA+0xC0	R	ISP 多字编程状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
D3	D2	D1	D0	Reserved	ISPFF	PPGO	MPBUSY

位	描述	
[31:8]	<b>Reserved</b>	保留
[7]	<b>D3</b>	<b>ISP数据3 标志 (只读)</b> 当FMC_MPDAT3被写入数据时为1。当FMC_MPDAT3内的资料写入到flash的动作完成时，这个位会自动清零。 0 = FMC_MPDAT3寄存器为空，编程到flash已经结束。 1 = 数据已写入FMC_MPDAT3，但还未编程到flash中。
[6]	<b>D2</b>	<b>ISP数据2 标志 (只读)</b> 当FMC_MPDAT2被写入数据时为1。当FMC_MPDAT2内的资料写入到flash的动作完成时，这个位会自动清零。 0 = FMC_MPDAT2寄存器为空，编程到flash已经结束。 1 = 数据已写入FMC_MPDAT2，但还未编程到flash中。
[5]	<b>D1</b>	<b>ISP数据1 标志 (只读)</b> 当FMC_MPDAT1被写入数据时为1。当FMC_MPDAT1内的资料写入到flash的动作完成时，这个位会自动清零。 0 = FMC_MPDAT1寄存器为空，编程到flash已经结束。 1 = 数据已写入FMC_MPDAT1，但还未编程到flash中。
[4]	<b>D0</b>	<b>ISP数据0 标志 (只读)</b> 当FMC_MPDAT0被写入数据时为1。当FMC_MPDAT0内的资料写入到flash的动作完成时，这个位会自动清零。 0 = FMC_MPDAT0寄存器为空，编程到flash已经结束。 1 = 数据已写入FMC_MPDAT0，但还未编程到flash中。.
[3]	<b>Reserved</b>	保留

[2]	<b>ISPFF</b>	<b>ISP 失败标志 (只读)</b> 该位是寄存器ISPFF (FMC_ISPCTL[6])的镜像，需向FMC_ISPCTL[6] 或FMC_ISPSTS[6]写1清零。当已启动的ISP符合下面的任何一个条件时，该位由硬件置1： (1) APUEN等于0时，APROM写APROM. (2) LDUEN等于0时，LDROM写LDROM. (3) CFGUEN等于0时，CONFIG被擦除或编程. (4) SPUEN等于0时，SPROM被擦除或编程 (5) 在 SPROM 安全模式，SPROM 被编程 (6) ICE 连接状态中，LOCK模式时，下页擦除命令 (7) 检测到欠压时，下擦除或编程命令 (8) 定义地址无效，比如超出正常范围 (9) 无效的ISP命令 (10) 向量地址映射到SPROM区.
[1]	<b>PPGO</b>	<b>ISP 多字编程状态 (只读)</b> 0 = ISP 多字编程操作未进行 1 = ISP 多字编程操作正在执行
[0]	<b>MPBUSY</b>	<b>ISP多字编程忙标志(只读)</b> 写1到ISPGO开始ISP多字编程操作，当ISP多字操作完成后，该位由硬件自动清零。 该位是ISPGO(FMC_ISPTRG[0]寄存器的镜像。 0 = ISP 多字编程操作完成 1 = 正在执行ISP 多字编程操作

**FMC\_MPADDR ISP 多字编程地址寄存器**

寄存器	偏移量	R/W	描述	复位值
FMC_MPADDR	FMC_BA+0xC4	R	ISP多字编程地址寄存器	0x0000_0000

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

位	描述		
[31:0]	MPADDR	ISP多字编程地址	当ISPGO标志为1时， MPADDR是ISP 多字编程操作的地址。当ISP多字编程完成后，MPADDR会保留ISP的最终地址。

## 6.5 GPIO 通用 I/O

### 6.5.1 概述

NuMicro™ NUC480 系列多达118 个通用I/O管脚和其他功能管脚共享，这取决于芯片的配置。118个管脚分配在PA, PB, PC, PD, PE, PF, PG 和 PH这8个端口上。PA, PB, PE和PG有16个管脚, PC, PD有15个管脚, PF, PH有12个管脚。每个管脚都是独立的, 都有相应的寄存器位来控制管脚功能模式与数据。

I/O管脚的I/O类型可由软件独立地配置为输入, 推挽式的输出, 开漏输出或准双向模式。复位之后, 所有管脚的 I/O 管脚类型取决于CIOINI (CONFIG0[10])的设置。每个I/O管脚有一个阻值为50 kΩ左右的弱上拉电阻接到VDD 上, VDD范围从1.8 V 到 3.6 V。

### 6.5.2 特性

- 四种 I/O 模式:
  - 准双向模式
  - 推挽输出
  - 开漏输出
  - 高阻态输入
- 可选TTL/施密特 触发输入
- I/O可以配置为边沿/电平触发的中断源
- 支持高驱动力及高翻转速率的I/O模式。
- 通过CIOINI (CONFIG0[10])可配置所有I/O复位之后的默认模式。
  - 如果CIOIN = 0, 芯片复位后所有的GPIO管脚是准双向模式
  - 如果CIOIN = 1, 芯片复位后所有的GPIO管脚是高阻输入模式
- I/O管脚仅在准双向模式, 内部上拉电阻才使能。
- 使能管脚中断功能将也使能了唤醒功能。

## 6.5.3 框图

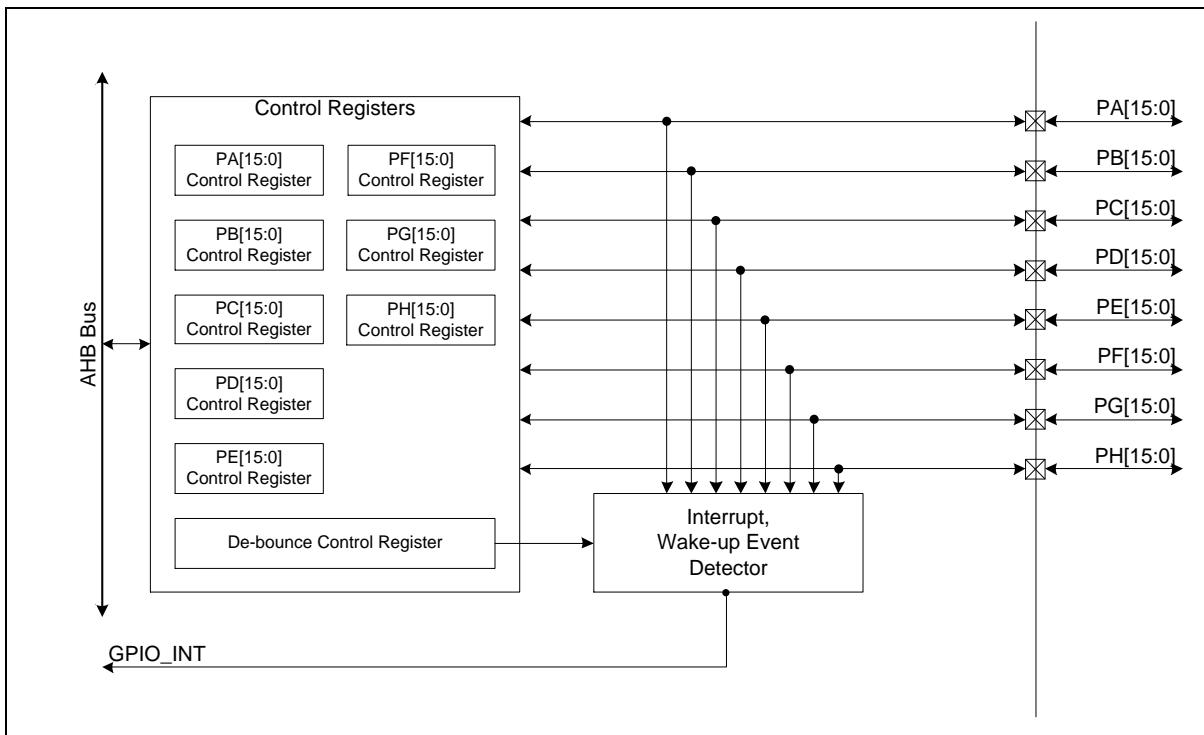


图 6.5-1 GPIO 控制器框图

注: The PC.15/PD.15/PF.12/PF.13/PF.14/PF.15/PH.12/PH.13/PH.14/PH.15 管脚不可用。

## 6.5.4 基本配置

- 复位配置
  - 在GPIO\_RST (SYS\_IPRST1[1]) 中 复位GPIO
- 管脚配置

组	管脚名称	GPIO	MFP
INT0	INT0	PA.6, PB.5	MFP15
INT1	INT1	PA.7, PB.4	MFP15
INT2	INT2	PB.3, PC.6	MFP15
INT3	INT3	PB.2, PC.7	MFP15
INT4	INT4	PB.6	MFP13
		PA.8	MFP15
INT5	INT5	PB.7	MFP13
		PD.12	MFP15
INT6	INT6	PB.8	MFP13
		PD.11	MFP15
INT7	INT7	PB.9	MFP13

	PD.10	MFP15
--	-------	-------

### 6.5.5 功能描述

#### 6.5.5.1 输入模式

设置 MODEn ( $Px\_MODE[2n+1:2n]$ ) 为 00,  $Px.n$  管脚为输入模式, I/O管脚为三态（高阻）, 没有输出驱动能力。管脚 ( $Px\_PIN[n]$ ) 的值反映相应端口的状态。

#### 6.5.5.2 推挽输出模式

下图为推挽输出模式。设置 MODEn ( $Px\_MODE[2n+1:2n]$ ) 为 01,  $Px.n$  管脚为推挽输出模式, I/O支持数字输出功能, 有拉/灌电流能力。DOUT ( $Px\_DOUT[n]$ ) 相应位bit[n]的值被驱动到相应管脚上。

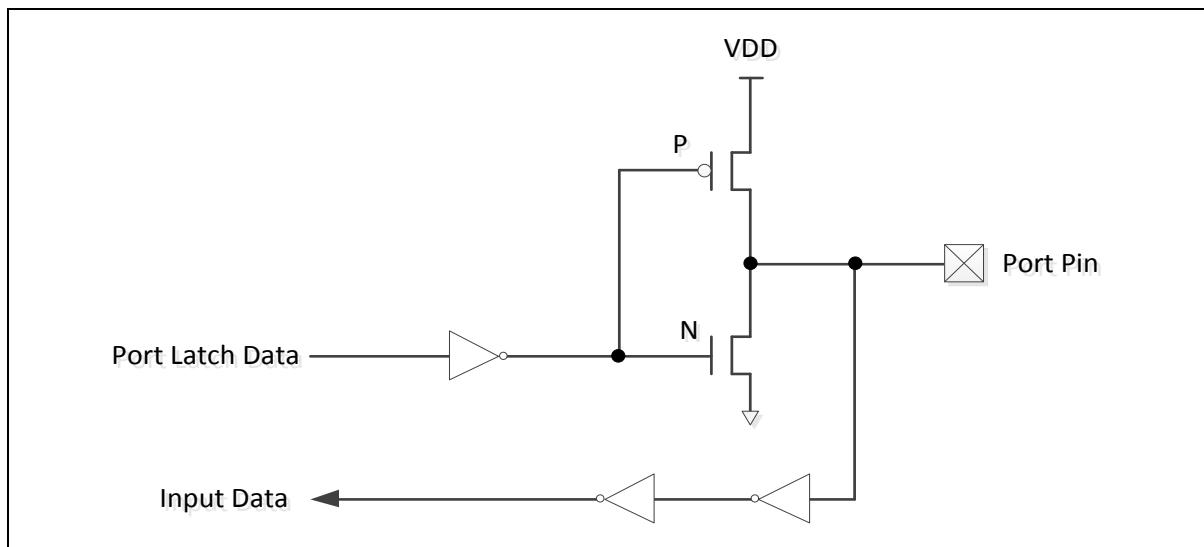


图 6.5-2 推挽输出模式

#### 6.5.5.3 开漏输出模式

下图为开漏输出模式。设置 MODEn ( $Px\_MODE[2n+1:2n]$ ) 为 10,  $Px.n$  管脚为开漏模式, 且I/O管脚数字输出功能仅支持灌电流, 要驱动到高电平需要一个外加上拉电阻。如果DOUT ( $Px\_DOUT[n]$ ) 相应位为‘0’, 管脚上输出低; 如果DOUT ( $Px\_DOUT[n]$ ) 相应位为‘1’, 该管脚输出高由外部上拉电阻控制。

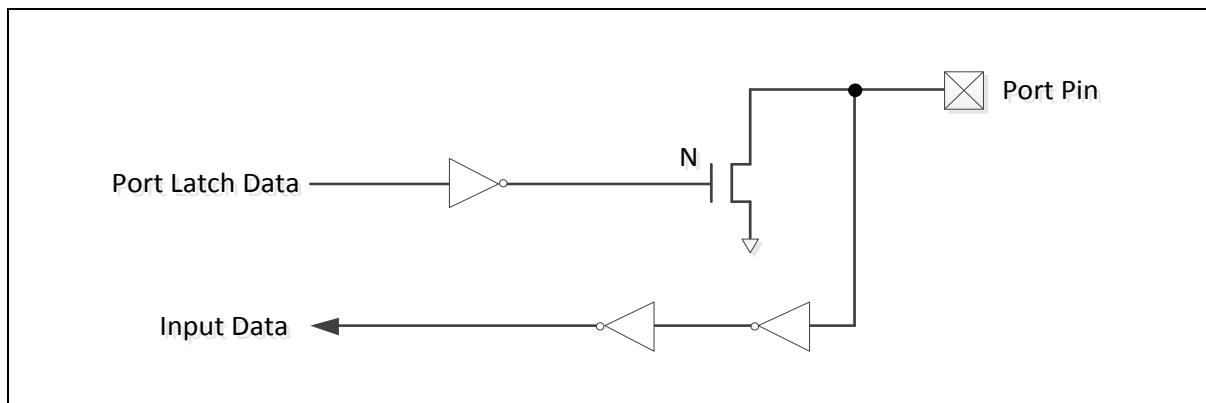


图 6.5-3 开漏输出

#### 6.5.5.4 准双向模式

下图为准双向模式。设置MODEn (Px\_MODE[2n+1:2n])为11，Px.n 管脚为准双向模式，I/O同时支持数字输出和输入功能，但拉电流能力仅达数百uA。要实现数字输入，需要先将DOUT (Px\_DOUT[n])相应位置1。准双向输出是80C51 及其派生产品常见的模式。若DOUT (Px\_DOUT[n])相应位bit[n]为‘0’，管脚上输出为“低”。若DOUT (Px\_DOUT[n])相应位bit[n]为‘1’，该管脚将检测管脚值。若管脚值为高，没有任何动作，若管脚值为低，在该管脚上将驱动2个时钟周期的高电平，然后禁止强输出驱动，其后管脚状态由内部上拉电阻控制。注意：准双向模式输出电流的大小仅有200 uA到30 uA(相应VDD的电压从5.0 V到2.5 V)。

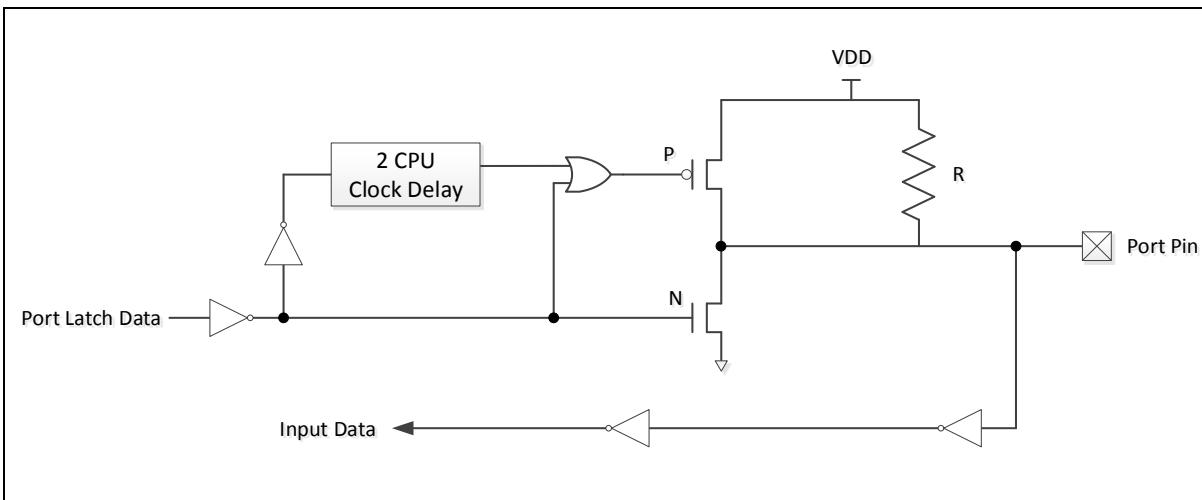


图 6.5-4 准双向 I/O 模式

**注意:**当 GPIO 被设置为准双向模式时且输出低电平，会额外消耗 65uA。

#### 6.5.5.5 GPIO 中断和唤醒功能

每个 GPIO 管脚都可以通过 RHIEN (Px\_INTEN[n+16])/FLIEN (Px\_INTEN[n]) 位 和 TYPE (Px\_INTTYPE[n]) 设置成芯片的中断源。有五种中断条件可以设置：低电平触发、高电平触发、下降沿触发和上升沿触发以及上升与下降沿同时触发。当芯片进入空闲/掉电模式时，GPIO也可以唤醒系统。设置GPIO为唤醒触发的条件与GPIO中断触发的条件相同。

#### 6.5.5.6 GPIO 消抖功能

GPIO消抖功能可以用来采样每一个GPIO管脚的中断输入，防止因干扰产生不想要的中断。GPIO消抖功能仅支持边沿触发。支持三种边沿触发条件，下降沿触发 FLIEN (Px\_INTEN[n]) 和上升沿触发 RHIEN (Px\_INTEN[n+16]) 以及上升与下降沿同时触发 TYPE (Px\_INTTYPE[n])。如果用户想使用消抖功能，消抖使能控制寄存器 Px\_DBEN 必须设置。消抖时钟源可以是 HCLK 或 LIRC (10kHz)，有 DBCLKSRC (Px\_DBCTL[4]) 的设置决定。DBCLKSEL (Px\_DBCTL[3:0]) 寄存器可以控制采样周期。

图 6.5-5 为GPIO上升沿触发中断。两个有效采样信号的时间间隔由 DBCLKSRC (Px\_DBCTL[4]) 和 DBCLKSEL (Px\_DBCTL[3:0]) 决定。IO口的有效数据需要采样两次。对于上升沿设置，如果引脚的状态在设置DBEN (Px\_DBEN) 之前是低。当产生一个管脚高有效数据，中断会产生。但是如果在设置DBEN (Px\_DBEN) 之前管脚状态是高，当先产生一个低有效数据，然后产生一个高有效数据时，中断会产生。下降沿触发中断和上升沿触发相反，如图 6.5-6。.

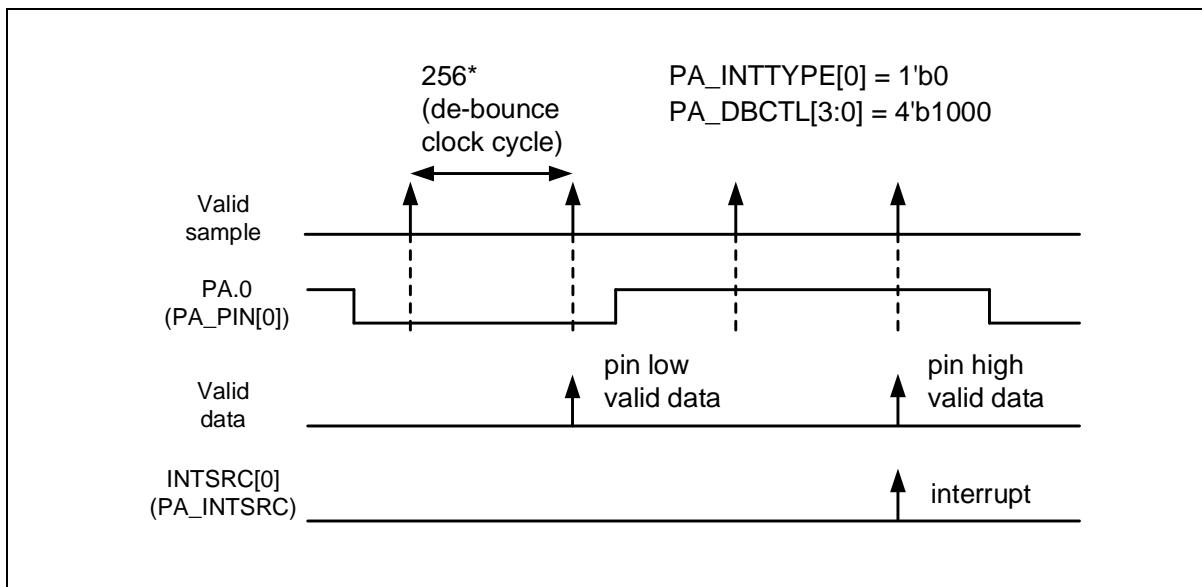


图 6.5-5 GPIO 上升沿触发中断

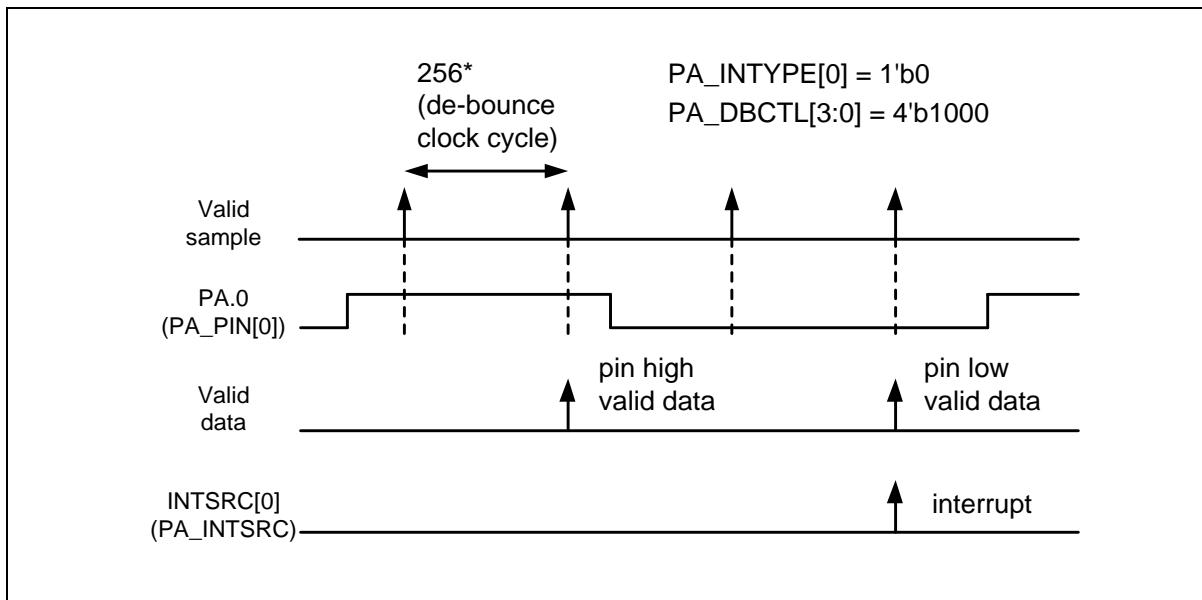


图 6.5-6 下降沿触发中断

GPIO消抖功能也支持在掉电模式下工作，下表为在不同系统状态下消抖功能支持的情况。通过设置 DBENn(Px\_DBEN[n]) 和 DBCLKSRC(Px\_DBCTL[4]) 为 1，可以使消抖功能在掉电模式下工作。DBCLKSEL (Px\_DBCTL[3:0]) 可以设置用来控制GPIO唤醒系统的消抖时间。

系统状态	DBEN	DBCLKSRC	描述
正常模式 / 空闲模式	0	0	不支持消抖功能
		1	不支持消抖功能

	1	0	支持消抖功能使用 HCLK 时钟
		1	支持消抖功能使用 LIRC (10 kHz) 时钟
Power Down Mode	0	0	不支持消抖功能
		1	不支持消抖功能
	1	0	不支持消抖功能
		1	支持消抖功能使用 LIRC (10 kHz) 时钟

表 6.5-1 消抖功能设置表

## 6.5.6 寄存器映射

R: 只读, W: 只写, R/W: 可读写

寄存器	偏移量	R/W	描述	复位值
<b>GPIO 基地址:</b>				
<b>GPIO_BA = 0x4000_4000</b>				
<b>PA_MODE</b>	GPIO_BA+0x000	R/W	PA I/O 模式控制	0xXXXX_XXXX
<b>PA_DINOFF</b>	GPIO_BA+0x004	R/W	PA 数字输入通道禁止控制	0x0000_0000
<b>PA_DOUT</b>	GPIO_BA+0x008	R/W	PA 数据输出值	0x0000_FFFF
<b>PA_DATMSK</b>	GPIO_BA+0x00C	R/W	PA数据输出写屏蔽	0x0000_0000
<b>PA_PIN</b>	GPIO_BA+0x010	R	PA管脚状态值	0x0000_XXXX
<b>PA_DBEN</b>	GPIO_BA+0x014	R/W	PA 消抖使能寄存器	0x0000_0000
<b>PA_INTTYPE</b>	GPIO_BA+0x018	R/W	PA 中断模式控制	0x0000_0000
<b>PA_INTEN</b>	GPIO_BA+0x01C	R/W	PA 中断使能控制	0x0000_0000
<b>PA_INTSRC</b>	GPIO_BA+0x020	R/W	PA 中断源标志	0x0000_XXXX
<b>PA_SMTEN</b>	GPIO_BA+0x024	R/W	PA 施密特 触发输入使能寄存器	0x0000_0000
<b>PA_SLEWCTL</b>	GPIO_BA+0x028	R/W	PA 高翻转速率控制寄存器	0x0000_0000
<b>PA_PUSEL</b>	GPIO_BA+0x030	R/W	PA 上拉和下拉选择寄存器	0x0000_0000
<b>PB_MODE</b>	GPIO_BA+0x040	R/W	PB I/O 模式控制	0xXXXX_XXXX
<b>PB_DINOFF</b>	GPIO_BA+0x044	R/W	PB 数字输入通道禁止控制	0x0000_0000
<b>PB_DOUT</b>	GPIO_BA+0x048	R/W	PB 数据输出值	0x0000_FFFF
<b>PB_DATMSK</b>	GPIO_BA+0x04C	R/W	PB数据输出写屏蔽	0x0000_0000
<b>PB_PIN</b>	GPIO_BA+0x050	R	PB管脚状态值	0x0000_XXXX
<b>PB_DBEN</b>	GPIO_BA+0x054	R/W	PB 消抖使能寄存器	0x0000_0000
<b>PB_INTTYPE</b>	GPIO_BA+0x058	R/W	PB 中断模式控制	0x0000_0000
<b>PB_INTEN</b>	GPIO_BA+0x05C	R/W	PB 中断使能控制	0x0000_0000
<b>PB_INTSRC</b>	GPIO_BA+0x060	R/W	PB 中断源标志	0x0000_XXXX
<b>PB_SMTEN</b>	GPIO_BA+0x064	R/W	PB施密特触发输入使能寄存器	0x0000_0000
<b>PB_SLEWCTL</b>	GPIO_BA+0x068	R/W	PB 高翻转速率控制寄存器	0x0000_0000
<b>PB_PUSEL</b>	GPIO_BA+0x070	R/W	PB 上拉和下拉选择寄存器	0x0000_0000
<b>PC_MODE</b>	GPIO_BA+0x080	R/W	PC I/O 模式控制	0xXXXX_XXXX
<b>PC_DINOFF</b>	GPIO_BA+0x084	R/W	PC数字输入通道禁止控制	0x0000_0000

<b>PC_DOUT</b>	GPIO_BA+0x088	R/W	PC 数据输出值	0x0000_FFFF
<b>PC_DATMSK</b>	GPIO_BA+0x08C	R/W	PC数据输出写屏蔽	0x0000_0000
<b>PC_PIN</b>	GPIO_BA+0x090	R	PC管脚状态值	0x0000_XXXX
<b>PC_DBEN</b>	GPIO_BA+0x094	R/W	PC 消抖使能寄存器	0x0000_0000
<b>PC_INTTYPE</b>	GPIO_BA+0x098	R/W	PC 中断模式控制	0x0000_0000
<b>PC_INTEN</b>	GPIO_BA+0x09C	R/W	PC 中断使能控制	0x0000_0000
<b>PC_INTSRC</b>	GPIO_BA+0x0A0	R/W	PC 中断源标志	0x0000_XXXX
<b>PC_SMTEN</b>	GPIO_BA+0x0A4	R/W	PC施密特触发输入使能寄存器	0x0000_0000
<b>PC_SLEWCTL</b>	GPIO_BA+0x0A8	R/W	PC 高翻转速率控制寄存器	0x0000_0000
<b>PC_PUSEL</b>	GPIO_BA+0x0B0	R/W	PC 上拉和下拉选择寄存器	0x0000_0000
<b>PD_MODE</b>	GPIO_BA+0x0C0	R/W	PD I/O 模式控制	0xXXXX_XXXX
<b>PD_DINOFF</b>	GPIO_BA+0x0C4	R/W	PD 数字输入通道禁止控制	0x0000_0000
<b>PD_DOUT</b>	GPIO_BA+0x0C8	R/W	PD 数据输出值	0x0000_FFFF
<b>PD_DATMSK</b>	GPIO_BA+0x0CC	R/W	PD数据输出写屏蔽	0x0000_0000
<b>PD_PIN</b>	GPIO_BA+0x0D0	R	PD管脚状态值	0x0000_XXXX
<b>PD_DBEN</b>	GPIO_BA+0x0D4	R/W	PD 消抖使能寄存器	0x0000_0000
<b>PD_INTTYPE</b>	GPIO_BA+0x0D8	R/W	PD 中断模式控制	0x0000_0000
<b>PD_INTEN</b>	GPIO_BA+0x0DC	R/W	PD 中断使能控制	0x0000_0000
<b>PD_INTSRC</b>	GPIO_BA+0x0E0	R/W	PD 中断源标志	0x0000_XXXX
<b>PD_SMTEN</b>	GPIO_BA+0x0E4	R/W	PD施密特触发输入使能寄存器	0x0000_0000
<b>PD_SLEWCTL</b>	GPIO_BA+0x0E8	R/W	PD 高翻转速率控制寄存器	0x0000_0000
<b>PD_PUSEL</b>	GPIO_BA+0x0F0	R/W	PD 上拉和下拉选择寄存器	0x0000_0000
<b>PE_MODE</b>	GPIO_BA+0x100	R/W	PE I/O 模式控制	0xXXXX_XXXX
<b>PE_DINOFF</b>	GPIO_BA+0x104	R/W	PE 数字输入通道禁止控制	0x0000_0000
<b>PE_DOUT</b>	GPIO_BA+0x108	R/W	PE 数据输出值	0x0000_FFFF
<b>PE_DATMSK</b>	GPIO_BA+0x10C	R/W	PE数据输出写屏蔽	0x0000_0000
<b>PE_PIN</b>	GPIO_BA+0x110	R	PE管脚状态值	0x0000_XXXX
<b>PE_DBEN</b>	GPIO_BA+0x114	R/W	PE 消抖使能寄存器	0x0000_0000
<b>PE_INTTYPE</b>	GPIO_BA+0x118	R/W	PE 中断模式控制	0x0000_0000
<b>PE_INTEN</b>	GPIO_BA+0x11C	R/W	PE 中断使能控制	0x0000_0000
<b>PE_INTSRC</b>	GPIO_BA+0x120	R/W	PE 中断源标志	0x0000_XXXX

<b>PE_SMTEN</b>	GPIO_BA+0x124	R/W	PE施密特触发输入使能寄存器	0x0000_0000
<b>PE_SLEWCTL</b>	GPIO_BA+0x128	R/W	PE高翻转速率控制寄存器	0x0000_0000
<b>PE_PUSEL</b>	GPIO_BA+0x130	R/W	PE上拉和下拉选择寄存器	0x0000_0000
<b>PF_MODE</b>	GPIO_BA+0x140	R/W	PF I/O 模式控制	0xXXXX_XXXX
<b>PF_DINOFF</b>	GPIO_BA+0x144	R/W	PF 数字输入通道禁止控制	0x0000_0000
<b>PF_DOUT</b>	GPIO_BA+0x148	R/W	PF 数据输出值	0x0000_FFFF
<b>PF_DATMSK</b>	GPIO_BA+0x14C	R/W	PF数据输出写屏蔽	0x0000_0000
<b>PF_PIN</b>	GPIO_BA+0x150	R	PF管脚状态值	0x0000_XXXX
<b>PF_DBEN</b>	GPIO_BA+0x154	R/W	PF 消抖使能寄存器	0x0000_0000
<b>PF_INTTYPE</b>	GPIO_BA+0x158	R/W	PF 中断模式控制	0x0000_0000
<b>PF_INTEN</b>	GPIO_BA+0x15C	R/W	PF 中断使能控制	0x0000_0000
<b>PF_INTSRC</b>	GPIO_BA+0x160	R/W	PF 中断源标志	0x0000_XXXX
<b>PF_SMTEN</b>	GPIO_BA+0x164	R/W	PF施密特触发输入使能寄存器	0x0000_0000
<b>PF_SLEWCTL</b>	GPIO_BA+0x168	R/W	PF高翻转速率控制寄存器	0x0000_0000
<b>PF_PUSEL</b>	GPIO_BA+0x170	R/W	PF上拉和下拉选择寄存器	0x0000_0000
<b>PG_MODE</b>	GPIO_BA+0x180	R/W	PG I/O 模式控制	0xXXXX_XXXX
<b>PG_DINOFF</b>	GPIO_BA+0x184	R/W	PG 数字输入通道禁止控制	0x0000_0000
<b>PG_DOUT</b>	GPIO_BA+0x188	R/W	PG 数据输出值	0x0000_1FFF
<b>PG_DATMSK</b>	GPIO_BA+0x18C	R/W	PG数据输出写屏蔽	0x0000_0000
<b>PG_PIN</b>	GPIO_BA+0x190	R	PG管脚状态值	0x0000_XXXX
<b>PG_DBEN</b>	GPIO_BA+0x194	R/W	PG 消抖使能寄存器	0x0000_0000
<b>PG_INTTYPE</b>	GPIO_BA+0x198	R/W	PG 中断模式控制	0x0000_0000
<b>PG_INTEN</b>	GPIO_BA+0x19C	R/W	PG 中断使能控制	0x0000_0000
<b>PG_INTSRC</b>	GPIO_BA+0x1A0	R/W	PG 中断源标志	0x0000_XXXX
<b>PG_SMTEN</b>	GPIO_BA+0x1A4	R/W	PG施密特触发输入使能寄存器	0x0000_0000
<b>PG_SLEWCTL</b>	GPIO_BA+0x1A8	R/W	PG高翻转速率控制寄存器	0x0000_0000
<b>PG_PUSEL</b>	GPIO_BA+0x1B0	R/W	PG上拉和下拉选择寄存器	0x0000_0000
<b>PH_MODE</b>	GPIO_BA+0x1C0	R/W	PH I/O 模式控制	0xXXXX_XXXX
<b>PH_DINOFF</b>	GPIO_BA+0x1C4	R/W	PH 数字输入通道禁止控制	0x0000_0000
<b>PH_DOUT</b>	GPIO_BA+0x1C8	R/W	PH 数据输出值	0x0000_1FFF

<b>PH_DATMSK</b>	GPIO_BA+0x1CC	R/W	PH数据输出写屏蔽	0x0000_0000
<b>PH_PIN</b>	GPIO_BA+0x1D0	R	PH管脚状态值	0x0000_XXXX
<b>PH_DBEN</b>	GPIO_BA+0x1D4	R/W	PH 消抖使能寄存器	0x0000_0000
<b>PH_INTTYPE</b>	GPIO_BA+0x1D8	R/W	PH 中断模式控制	0x0000_0000
<b>PH_INTEN</b>	GPIO_BA+0x1DC	R/W	PH 中断使能控制	0x0000_0000
<b>PH_INTSRC</b>	GPIO_BA+0x1E0	R/W	PH 中断源标志	0x0000_XXXX
<b>PH_SMTEN</b>	GPIO_BA+0x1E4	R/W	PH施密特触发输入使能寄存器	0x0000_0000
<b>PH_SLEWCTL</b>	GPIO_BA+0x1E8	R/W	PH高翻转速率控制寄存器	0x0000_0000
<b>PH_PUSEL</b>	GPIO_BA+0x1F0	R/W	PH上拉和下拉选择寄存器	0x0000_0000
<b>GPIO_DBCTL</b>	GPIO_BA+0x440	R/W	中断消抖控制寄存器	0x0000_0020
<b>PAn_PDIO n=0,1..15</b>	GPIO_BA+0x800+(0x04 * n)	R/W	GPIO PA.n 管脚数据输入/输出寄存器	0x0000_000X
<b>PBn_PDIO n=0,1..15</b>	GPIO_BA+0x840+(0x04 * n)	R/W	GPIO PB.n管脚数据输入/输出寄存器	0x0000_000X
<b>PCn_PDIO n=0,1..15</b>	GPIO_BA+0x880+(0x04 * n)	R/W	GPIO PC.n管脚数据输入/输出寄存器	0x0000_000X
<b>PDn_PDIO n=0,1..15</b>	GPIO_BA+0x8C0+(0x04 * n)	R/W	GPIO PD.n管脚数据输入/输出寄存器	0x0000_000X
<b>PEn_PDIO n=0,1..15</b>	GPIO_BA+0x900+(0x04 * n)	R/W	GPIO PE.n管脚数据输入/输出寄存器	0x0000_000X
<b>PFn_PDIO n=0,1..15</b>	GPIO_BA+0x940+(0x04 * n)	R/W	GPIO PF.n管脚数据输入/输出寄存器	0x0000_000X
<b>PGn_PDIO n=0,1..12</b>	GPIO_BA+0x980+(0x04 * n)	R/W	GPIO PG.n管脚数据输入/输出寄存器	0x0000_000X
<b>PHn_PDIO n=0,1..12</b>	GPIO_BA+0x9C0+(0x04 * n)	R/W	GPIO PH.n管脚数据输入/输出寄存器	0x0000_000X

## 6.5.7 寄存器描述

Px\_MODE 端口 A-H I/O 模式控制

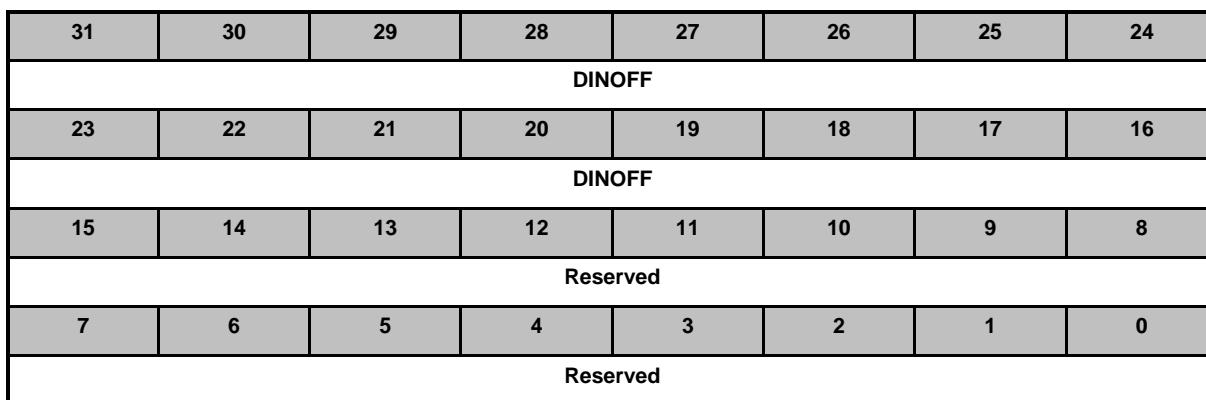
寄存器	偏移量	R/W	描述	复位值
<b>PA_MODE</b>	GPIO_BA+0x000	R/W	PA I/O 模式控制	0xFFFF_FFFF
<b>PB_MODE</b>	GPIO_BA+0x040	R/W	PB I/O模式控制	0xFFFF_FFFF
<b>PC_MODE</b>	GPIO_BA+0x080	R/W	PC I/O模式控制	0xFFFF_FFFF
<b>PD_MODE</b>	GPIO_BA+0x0C0	R/W	PD I/O模式控制	0xFFFF_FFFF
<b>PE_MODE</b>	GPIO_BA+0x100	R/W	PE I/O模式控制	0xFFFF_FFFF
<b>PF_MODE</b>	GPIO_BA+0x140	R/W	PF I/O模式控制	0xFFFF_FFFF
<b>PG_MODE</b>	GPIO_BA+0x180	R/W	PG I/O模式控制	0xFFFF_FFFF
<b>PH_MODE</b>	GPIO_BA+0x1C0	R/W	PH I/O模式控制	0xFFFF_FFFF

31	30	29	28	27	26	25	24
<b>MODE15</b>		<b>MODE14</b>		<b>MODE13</b>		<b>MODE12</b>	
23	22	21	20	19	18	17	16
<b>MODE11</b>		<b>MODE10</b>		<b>MODE9</b>		<b>MODE8</b>	
15	14	13	12	11	10	9	8
<b>MODE7</b>		<b>MODE6</b>		<b>MODE5</b>		<b>MODE4</b>	
7	6	5	4	3	2	1	0
<b>MODE3</b>		<b>MODE2</b>		<b>MODE1</b>		<b>MODE0</b>	

位	描述	
[2n+1:2n] n=0,1..15	<b>MODEn</b>	<p>端口 A-H I/O Pin[N] 模式控制            决定GPIO Px.n的I/O 类型.            00 = Px.n 管脚为高阻输入模式            01 = Px.n 管脚为推挽输出模式            10 = Px.n 管脚为开漏输出模式            11 = Px.n 管脚为准双向模式</p> <p><b>注意1:</b>            由CIOINI (CONFIG0 [10]).决定初始值, 如果CIOINI置0, 默认值是0xFFFF_FFFF上电后所有端口是准双向模式。如果CIOINI置1, 默认值是0x0000_0000, 上电后所有端口是输入模式。</p> <p><b>注意2:</b>            端口A/B/E/G: 最大. n=15            端口C/D: 最大. n=14            端口F/H: 最大. n=11</p>

## Px\_DINOFF 端口 A-H 数字输入通道关闭控制

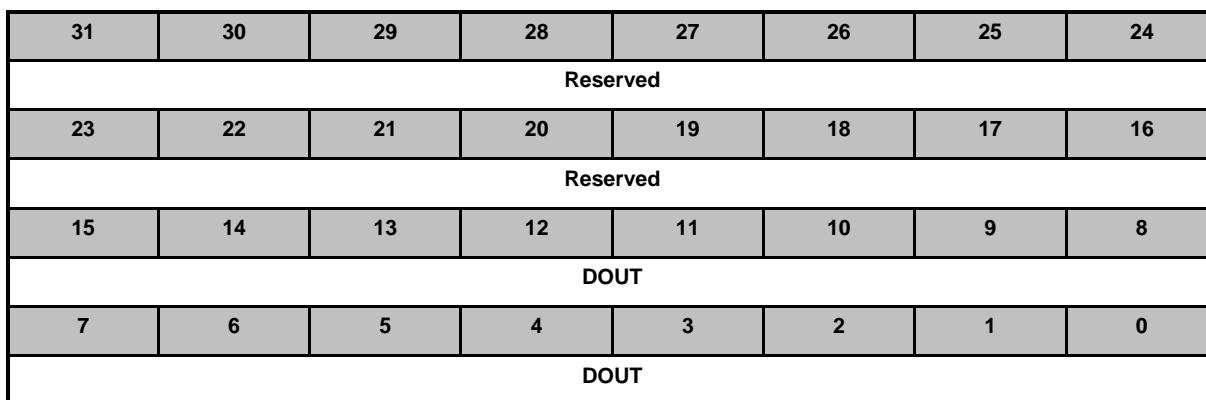
寄存器	偏移量	R/W	描述	复位值
PA_DINOFF	GPIO_BA+0x004	R/W	PA 数字输入通道关闭控制	0x0000_0000
PB_DINOFF	GPIO_BA+0x044	R/W	PB数字输入通道关闭控制	0x0000_0000
PC_DINOFF	GPIO_BA+0x084	R/W	PC数字输入通道关闭控制	0x0000_0000
PD_DINOFF	GPIO_BA+0x0C4	R/W	PD数字输入通道关闭控制	0x0000_0000
PE_DINOFF	GPIO_BA+0x104	R/W	PE数字输入通道关闭控制	0x0000_0000
PF_DINOFF	GPIO_BA+0x144	R/W	PF数字输入通道关闭控制	0x0000_0000
PG_DINOFF	GPIO_BA+0x184	R/W	PG数字输入通道关闭控制	0x0000_0000
PH_DINOFF	GPIO_BA+0x1C4	R/W	PH数字输入通道关闭控制	0x0000_0000



位	描述	
[n+16] n=0,1..15	DINOFF[n]	<p><b>端口A-H Pin[N] 关闭数字输入通道使能</b>            用于控制GPIO的数字输入通路是否使能。如果输入为模拟信号，用户可以关闭输入通道防止漏电。</p> <p>0 = 使能IO数据输入通道            1 = 关闭IO的数字输入通道(数字输入拉低)</p> <p><b>注意:</b>            端口A/B/E/G: 最大. n=15            端口C/D: 最大. n=14            端口F/H: 最大. n=11</p>
[15:0]	Reserved	保留

## Px\_DOUT 端口 A-H 数据输出值

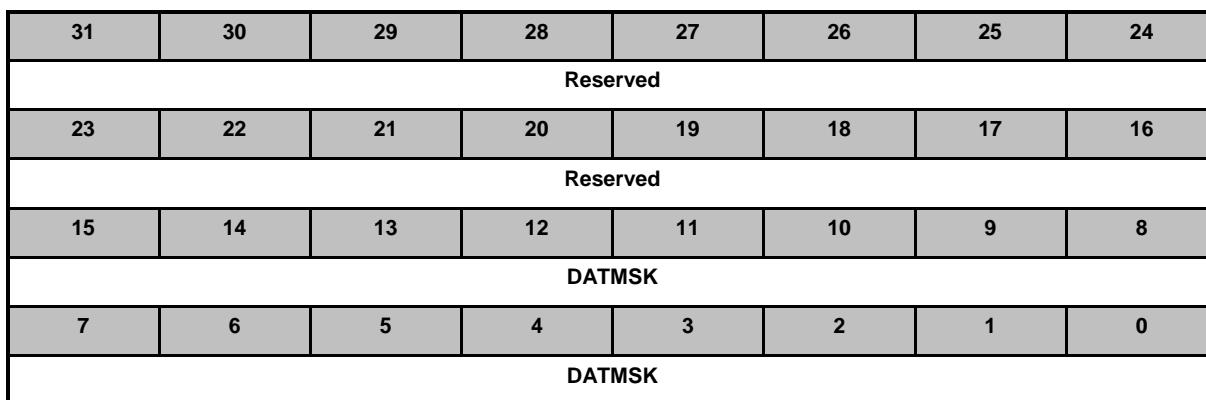
寄存器	偏移量	R/W	描述	复位值
PA_DOUT	GPIO_BA+0x008	R/W	PA 数据输出值	0x0000_FFFF
PB_DOUT	GPIO_BA+0x048	R/W	PB数据输出值	0x0000_FFFF
PC_DOUT	GPIO_BA+0x088	R/W	PC数据输出值	0x0000_FFFF
PD_DOUT	GPIO_BA+0x0C8	R/W	PD数据输出值	0x0000_FFFF
PE_DOUT	GPIO_BA+0x108	R/W	PE数据输出值	0x0000_FFFF
PF_DOUT	GPIO_BA+0x148	R/W	PF数据输出值	0x0000_FFFF
PG_DOUT	GPIO_BA+0x188	R/W	PG数据输出值	0x0000_1FFF
PH_DOUT	GPIO_BA+0x1C8	R/W	PH数据输出值	0x0000_1FFF



位	描述	
[31:16]	Reserved	保留
[n] n=0,1..15	DOUT[n]	<p><b>端口A-H Pin[n] 输出值</b>            在GPIO配置成推挽输出, 开漏和准双向模式时, 控制GPIO相应管脚的状态.            0 = GPIO配置成推挽输出, 开漏和准双向模式时, Px.n为低            1 = GPIO配置成推挽输出, 准双向模式时, Px.n为高  <b>注意:</b>            端口A/B/E/G: 最大. n=15            端口C/D: 最大. n=14            端口F/H: 最大. n=11         </p>

## Px\_DATMSK 端口 A-H 数据输出写屏蔽

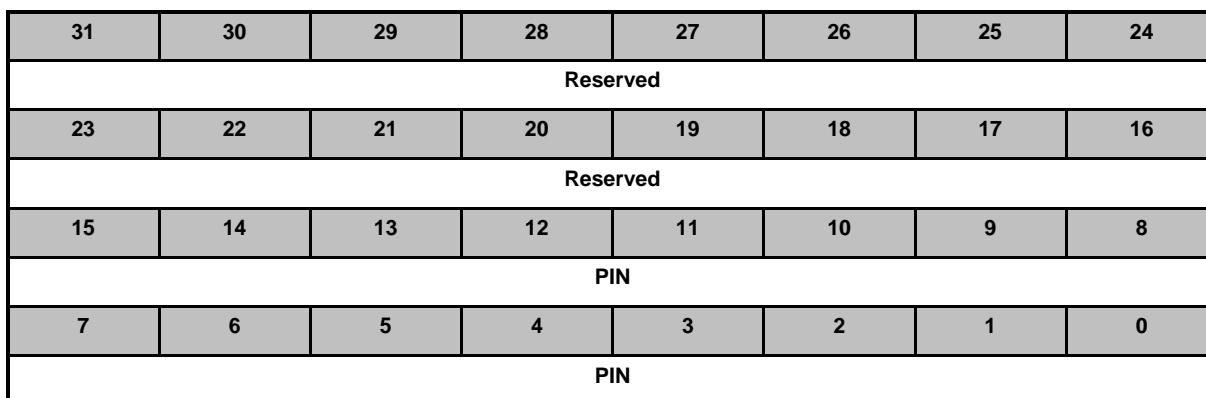
寄存器	偏移量	R/W	描述	复位值
PA_DATMSK	GPIO_BA+0x00C	R/W	PA 数据输出写屏蔽	0x0000_0000
PB_DATMSK	GPIO_BA+0x04C	R/W	PB数据输出写屏蔽	0x0000_0000
PC_DATMSK	GPIO_BA+0x08C	R/W	PC数据输出写屏蔽	0x0000_0000
PD_DATMSK	GPIO_BA+0x0CC	R/W	PD数据输出写屏蔽	0x0000_0000
PE_DATMSK	GPIO_BA+0x10C	R/W	PE数据输出写屏蔽	0x0000_0000
PF_DATMSK	GPIO_BA+0x14C	R/W	PF数据输出写屏蔽	0x0000_0000
PG_DATMSK	GPIO_BA+0x18C	R/W	PG数据输出写屏蔽	0x0000_0000
PH_DATMSK	GPIO_BA+0x1CC	R/W	PH数据输出写屏蔽	0x0000_0000



位	描述	
[31:8]	Reserved	保留.
[n] n=0,1..15	DATMSK[n]	<p><b>端口 A-H Pin[n] 数据输出写屏蔽</b>            用于保护对应DOUT (Px_DOUT[n])位，. 当设置DATMSK (Px_DATMSK[n])为‘1’时，相应DOUT (Px_DOUT[n])位被保护，写信号被屏蔽时，写数据到收保护的位会被忽略。            0 = 相应的DOUT (Px_DOUT[n])位可以被更新            1 = 保护相应的DOUT (Px_DOUT[n])位</p> <p><b>注意1：</b> 该功能只保护相应的DOUT (Px_DOUT[n])位，不保护(PDIO (Pxn_PDIO[0]))相应位。</p> <p><b>注意2：</b>            端口A/B/E/G: 最大. n=15            端口C/D: 最大. n=14            端口F/H: 最大. n=11</p>

**Px\_PIN 端口 A-H 管脚状态值**

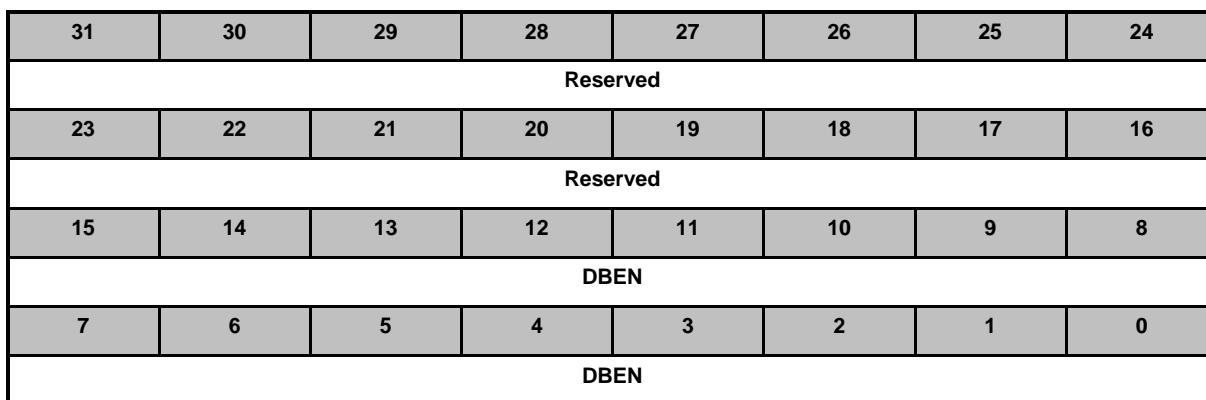
寄存器	偏移量	R/W	描述	复位值
<b>PA_PIN</b>	GPIO_BA+0x010	R	PA管脚状态值	0x0000_XXXX
<b>PB_PIN</b>	GPIO_BA+0x050	R	PB管脚状态值	0x0000_XXXX
<b>PC_PIN</b>	GPIO_BA+0x090	R	PC管脚状态值	0x0000_XXXX
<b>PD_PIN</b>	GPIO_BA+0x0D0	R	PD管脚状态值	0x0000_XXXX
<b>PE_PIN</b>	GPIO_BA+0x110	R	PE管脚状态值	0x0000_XXXX
<b>PF_PIN</b>	GPIO_BA+0x150	R	PF管脚状态值	0x0000_XXXX
<b>PG_PIN</b>	GPIO_BA+0x190	R	PG管脚状态值	0x0000_XXXX
<b>PH_PIN</b>	GPIO_BA+0x1D0	R	PH管脚状态值	0x0000_XXXX



位	描述	
[31:16]	Reserved	保留
[n] n=0,1..15	PIN[n]	<p><b>端口 A-H Pin[n]管脚数据</b></p> <p>这些位的值为各个GPIO Px.n管脚真实状态的反映。如果值为'1'，表示相应管脚状态为高，否则为低</p> <p><b>注意：</b></p> <p>端口A/B/E/G：最大. n=15</p> <p>端口C/D：最大. n=14</p> <p>端口F/H：最大. n=11</p>

**Px\_DBEN 端口 A-H 消抖使能控制寄存器**

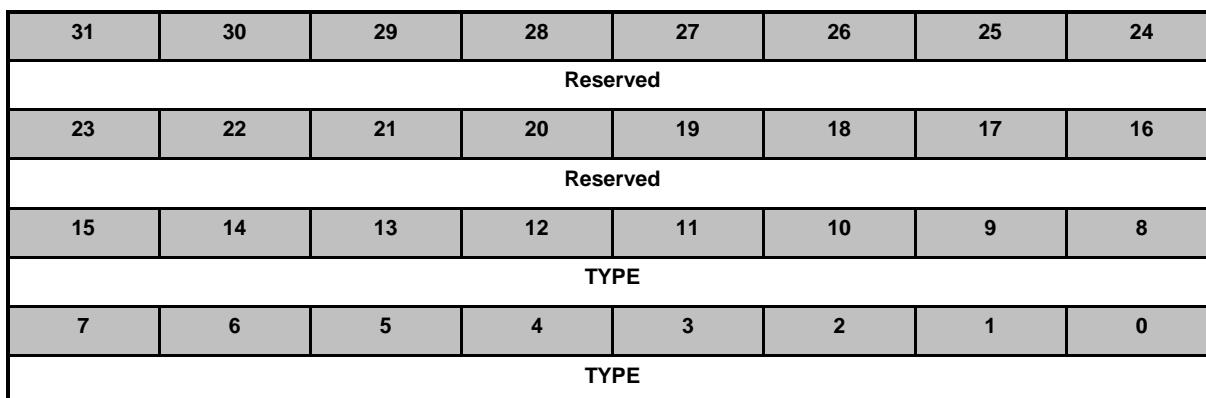
寄存器	偏移量	R/W	描述	复位值
PA_DBEN	GPIO_BA+0x014	R/W	PA 消抖使能控制寄存器	0x0000_0000
PB_DBEN	GPIO_BA+0x054	R/W	PB消抖使能控制寄存器	0x0000_0000
PC_DBEN	GPIO_BA+0x094	R/W	PC消抖使能控制寄存器	0x0000_0000
PD_DBEN	GPIO_BA+0x0D4	R/W	PD消抖使能控制寄存器	0x0000_0000
PE_DBEN	GPIO_BA+0x114	R/W	PE消抖使能控制寄存器	0x0000_0000
PF_DBEN	GPIO_BA+0x154	R/W	PF消抖使能控制寄存器	0x0000_0000
PG_DBEN	GPIO_BA+0x194	R/W	PG消抖使能控制寄存器	0x0000_0000
PH_DBEN	GPIO_BA+0x1D4	R/W	PH消抖使能控制寄存器	0x0000_0000



位	描述	
[31:16]	Reserved	保留
[n] n=0,1..15	DBEN[n]	<p><b>端口A-H Pin[n] 输入信号去抖动使能位</b></p> <p>DBEN[n]用于使能相应位的消抖功能。如果输入信号脉冲宽度不能被两个连续的消抖采样周期所采样，则被视为信号抖动，从而不触发中断。消抖时钟源由DBCLKSRC(GPIO_DBCTL [4])控制，一个消抖周期长度由DBCLKSEL (GPIO_DBCTL [3:0])控制。</p> <p>0 = 禁用Px.n消抖功能 1 = 使能Px.n消抖功能</p> <p>消抖功能对于边沿触发中断有效，对于电平触发中断模式，消抖功能使能位不起作用。</p> <p><b>注意：</b></p> <p>端口A/B/E/G: 最大. n=15 端口C/D: 最大. n=14 端口F/H: 最大. n=11</p>

## Px\_INTTYPE 端口 A-H 中断模式控制

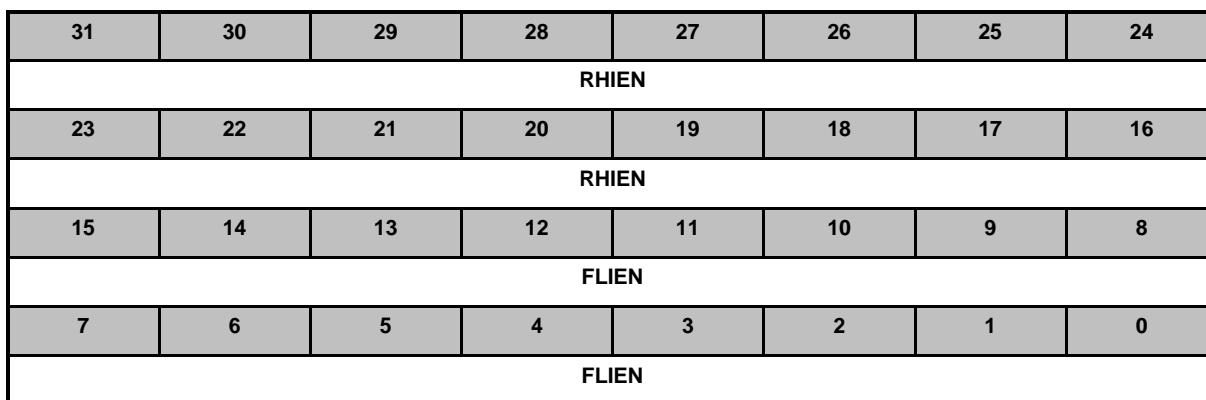
寄存器	偏移量	R/W	描述	复位值
PA_INTTYPE	GPIO_BA+0x018	R/W	PA 中断触发模式控制	0x0000_0000
PB_INTTYPE	GPIO_BA+0x058	R/W	PB中断触发模式控制	0x0000_0000
PC_INTTYPE	GPIO_BA+0x098	R/W	PC中断触发模式控制	0x0000_0000
PD_INTTYPE	GPIO_BA+0x0D8	R/W	PD中断触发模式控制	0x0000_0000
PE_INTTYPE	GPIO_BA+0x118	R/W	PE中断触发模式控制	0x0000_0000
PF_INTTYPE	GPIO_BA+0x158	R/W	PF中断触发模式控制	0x0000_0000
PG_INTTYPE	GPIO_BA+0x198	R/W	PG中断触发模式控制	0x0000_0000
PH_INTTYPE	GPIO_BA+0x1D8	R/W	PH中断触发模式控制	0x0000_0000



位	描述	
[31:16]	Reserved	保留
[n] n=0,1..15	TYPE[n]	<p>端口A-H Pin[n] 边沿或电平检测中断触发模式控制</p> <p>TYPE (Px_INTTYPE[n]) 位用于控制电平触发或边沿触发中断。若为边沿触发中断，触发源可由去抖动控制，如果是电平触发中断，输入源由一个HCLK时钟采样并产生中断。</p> <p>0 = 边沿触发中断 1 = 电平触发中断</p> <p>如果设置管脚为电平触发模式，则在寄存器 RHIEN (Px_INTEN[n+16])/FLIEN (Px_INTEN[n])中，只能设置一个电平(高电平或者低电平)；若设置了两个电平都触发中断，则设置被忽略，不会产生中断</p> <p>去抖动功能对于边沿触发中断有效，对于电平触发中断无效</p> <p><b>注意：</b></p> <p>端口A/B/E/G：最大. n=15 端口C/D：最大. n=14 端口F/H：最大. n=11.</p>

## Px\_INTEN 端口 A-H 中断使能控制寄存器

寄存器	偏移量	R/W	描述	复位值
PA_INTEN	GPIO_BA+0x01C	R/W	PA 中断使能控制寄存器	0x0000_0000
PB_INTEN	GPIO_BA+0x05C	R/W	PB中断使能控制寄存器	0x0000_0000
PC_INTEN	GPIO_BA+0x09C	R/W	PC中断使能控制寄存器	0x0000_0000
PD_INTEN	GPIO_BA+0x0DC	R/W	PD中断使能控制寄存器	0x0000_0000
PE_INTEN	GPIO_BA+0x11C	R/W	PE中断使能控制寄存器	0x0000_0000
PF_INTEN	GPIO_BA+0x15C	R/W	PF中断使能控制寄存器	0x0000_0000
PG_INTEN	GPIO_BA+0x19C	R/W	PG中断使能控制寄存器	0x0000_0000
PH_INTEN	GPIO_BA+0x1DC	R/W	PH中断使能控制寄存器	0x0000_0000



位	描述
[n+16] n=0,1..15	<p><b>RHIEN[n]</b></p> <p>端口A-H Pin[n] 输入上升沿或输入高电平中断使能位</p> <p>RHIEN (Px_INTEN[n+16])用于使能相应GPIO Px.n输入的中断。置‘1’也使能了管脚唤醒功能</p> <p>当设置RHIEN (Px_INTEN[n+16])位为‘1’：</p> <p>如果中断是电平触发模式(TYPE (Px_INTTYPE[n]) = 1)，输入Px.n的状态为高电平时，产生中断。</p> <p>如果中断是边沿触发模式(TYPE (Px_INTTYPE[n]) =0)，输入Px.n的状态由低电平到高电平变化时，产生中断。</p> <p>1 = 使能Px.n高电平或由低电平到高电平变化的中断 0 = 禁用Px.n高电平或由低电平到高电平变化的中断</p> <p><b>注意：</b></p> <p>端口A/B/E/G： 最大. n=15 端口C/D： 最大. n=14 端口F/H： 最大. n=11.</p>
[n] n=0,1..15	<p><b>FLIEN[n]</b></p> <p>使能端口A-H Pin[n] 输入下降沿或输入低电平的中断</p> <p>FLIEN (Px_INTEN[n])用于使能相应GPIO Px.n输入的中断。置 ‘1’ 也使能了管脚唤醒功能</p>

	<p>当设置FLIEN (Px_INTEN[n])位为‘1’:</p> <p>如果中断是电平触发模式(TYPE (Px_INTTYPE[n]) = 1),, 输入Px.n的状态为低电平时, 产生中断.</p> <p>如果中断是边沿触发模式(TYPE (Px_INTTYPE[n]) = 0),, 输入Px.n的状态由高电平到低电平变化时, 产生中断.</p> <p>0 = 禁用Px.n低电平或由高电平到低电平变化的中断</p> <p>1 = 使能Px.n低电平或由高电平到低电平变化的中断</p> <p><b>注意:</b></p> <p>端口A/B/E/G: 最大. n=15</p> <p>端口C/D: 最大. n=14</p> <p>端口F/H: 最大. n=11</p>
--	--

Px\_INTSRC 端口 A-H 中断源标志

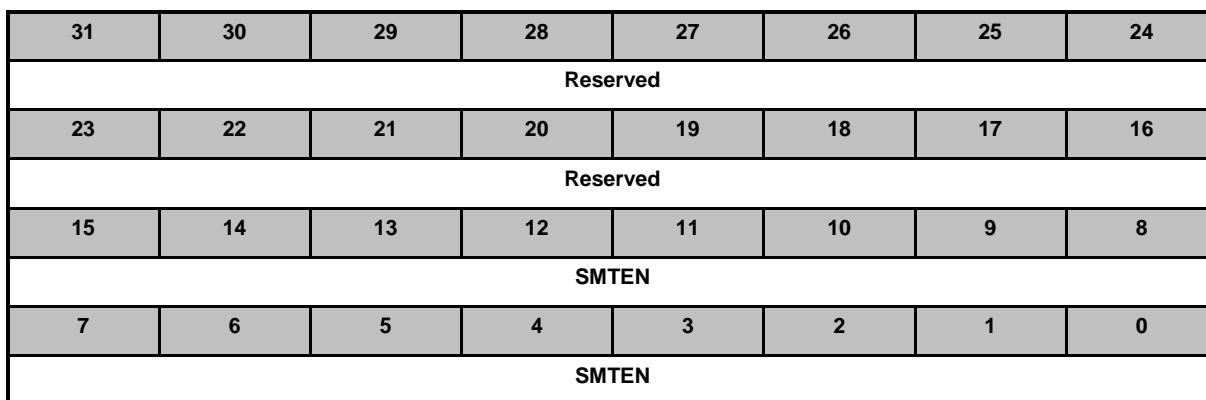
寄存器	偏移量	R/W	描述	复位值
PA_INTSRC	GPIO_BA+0x020	R/W	PA 中断源标志	0x0000_XXXX
PB_INTSRC	GPIO_BA+0x060	R/W	PB中断源标志	0x0000_XXXX
PC_INTSRC	GPIO_BA+0x0A0	R/W	PC中断源标志	0x0000_XXXX
PD_INTSRC	GPIO_BA+0x0E0	R/W	PD中断源标志	0x0000_XXXX
PE_INTSRC	GPIO_BA+0x120	R/W	PE中断源标志	0x0000_XXXX
PF_INTSRC	GPIO_BA+0x160	R/W	PF中断源标志	0x0000_XXXX
PG_INTSRC	GPIO_BA+0x1A0	R/W	PG中断源标志	0x0000_XXXX
PH_INTSRC	GPIO_BA+0x1E0	R/W	PH中断源标志	0x0000_XXXX

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
INTSRC							
7	6	5	4	3	2	1	0
INTSRC							

位	描述	
[31:16]	Reserved	保留.
[n] n=0,1..15	INTSRC[n]	<p>端口A-H Pin[n]中断触发源标志</p> <p>写：</p> <p>0= 无动作</p> <p>1= 清相应的挂起中断标志</p> <p>读：</p> <p>0 = Px.n没有中断</p> <p>1 = Px.n产生中断</p> <p>注意：</p> <p>端口A/B/E/G: 最大. n=15</p> <p>端口C/D: 最大. n=14</p> <p>端口F/H: 最大. n=11</p>

## Px\_SMTEN 端口 A-H 输入施密特触发使能寄存器

寄存器	偏移量	R/W	描述	复位值
PA_SMTEN	GPIO_BA+0x024	R/W	PA 输入施密特触发使能寄存器	0x0000_0000
PB_SMTEN	GPIO_BA+0x064	R/W	PB输入施密特触发使能寄存器	0x0000_0000
PC_SMTEN	GPIO_BA+0x0A4	R/W	PC输入施密特触发使能寄存器	0x0000_0000
PD_SMTEN	GPIO_BA+0x0E4	R/W	PD输入施密特触发使能寄存器	0x0000_0000
PE_SMTEN	GPIO_BA+0x124	R/W	PE输入施密特触发使能寄存器	0x0000_0000
PF_SMTEN	GPIO_BA+0x164	R/W	PF输入施密特触发使能寄存器	0x0000_0000
PG_SMTEN	GPIO_BA+0x1A4	R/W	PG输入施密特触发使能寄存器	0x0000_0000
PH_SMTEN	GPIO_BA+0x1E4	R/W	PH输入施密特触发使能寄存器	0x0000_0000



位	描述	
[31:16]	Reserved	保留
[n] n=0,1..15	SMTEN[n]	<p>端口 A-H Pin[n] 施密特触发输入使能位            0 = 禁用 Px.n 施密特触发输入功能            1 = 使能Px.n 施密特触发输入功能</p> <p>注意：            端口A/B/E/G: 最大. n=15            端口C/D: 最大. n=14            端口F/H: 最大. n=11</p>

## Px\_SLEWCTL 端口 A-H 高翻转速率控制寄存器

寄存器	偏移地址	R/W	描述	复位值
PA_SLEWCTL	GPIO_BA+0x028	R/W	PA 高翻转速率控制寄存器	0x0000_0000
PB_SLEWCTL	GPIO_BA+0x068	R/W	PB 高翻转速率控制寄存器	0x0000_0000
PC_SLEWCTL	GPIO_BA+0x0A8	R/W	PC 高翻转速率控制寄存器	0x0000_0000
PD_SLEWCTL	GPIO_BA+0x0E8	R/W	PD 高翻转速率控制寄存器	0x0000_0000
PE_SLEWCTL	GPIO_BA+0x128	R/W	PE 高翻转速率控制寄存器	0x0000_0000
PF_SLEWCTL	GPIO_BA+0x168	R/W	PF 高翻转速率控制寄存器	0x0000_0000
PG_SLEWCTL	GPIO_BA+0x1A8	R/W	PG 高翻转速率控制寄存器	0x0000_0000
PH_SLEWCTL	GPIO_BA+0x1E8	R/W	PH 高翻转速率控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
HSREN15		HSREN14		HSREN13		HSREN12	
23	22	21	20	19	18	17	16
HSREN11		HSREN10		HSREN9		HSREN8	
15	14	13	12	11	10	9	8
HSREN07		HSREN6		HSREN5		HSREN4	
7	6	5	4	3	2	1	0
HSREN03		HSREN2		HSREN1		HSREN0	

位	描述	
[2n+1:2n] n=0,1..15	HSRENn	<p>端口 A-H Pin[n] 高翻转速率控制</p> <p>00 = Px.n 输出是正常翻转速度模式 (2.7V 时最大 40 MHz).</p> <p>01 = Px.n 输出是高翻转速度模式 (2.7V 时最大 80 MHz).</p> <p>10 = Px.n 输出是快速翻转速度模式 (2.7V 时最大 100 MHz ).</p> <p>11 = 保留.</p> <p><b>注意:</b></p> <p>端口 A/B/E/G: 最大. n=15</p> <p>端口 C/D: 最大. n=14</p> <p>端口 F/H: 最大. n=11</p>

## Px\_PUSEL 端口 A-H 上拉和下拉选择寄存器

寄存器	偏移地址	R/W	描述	复位值
PA_PUSEL	GPIO_BA+0x030	R/W	PA上拉和下拉选择寄存器	0x0000_0000
PB_PUSEL	GPIO_BA+0x070	R/W	PB上拉和下拉选择寄存器	0x0000_0000
PC_PUSEL	GPIO_BA+0x0B0	R/W	PC上拉和下拉选择寄存器	0x0000_0000
PD_PUSEL	GPIO_BA+0x0F0	R/W	PD上拉和下拉选择寄存器	0x0000_0000
PE_PUSEL	GPIO_BA+0x130	R/W	PE上拉和下拉选择寄存器	0x0000_0000
PF_PUSEL	GPIO_BA+0x170	R/W	PF上拉和下拉选择寄存器	0x0000_0000
PG_PUSEL	GPIO_BA+0x1B0	R/W	PG上拉和下拉选择寄存器	0x0000_0000
PH_PUSEL	GPIO_BA+0x1F0	R/W	PH上拉和下拉选择寄存器	0x0000_0000

31	30	29	28	27	26	25	24
PUSEL15		PUSEL14		PUSEL13		PUSEL12	
23	22	21	20	19	18	17	16
PUSEL11		PUSEL10		PUSEL9		PUSEL8	
15	14	13	12	11	10	9	8
PUSEL7		PUSEL6		PUSEL5		PUSEL4	
7	6	5	4	3	2	1	0
PUSEL3		PUSEL2		PUSEL1		PUSEL0	

Bits	Description	
[2n+1:2n] n=0,1..15	PUSELn	<p>端口 A-H Pin[n] 上拉和下拉使能寄存器</p> <p>决定Px.n 管脚每一个I/O上拉/下拉是否使能</p> <p>00 = Px.n 上拉和下拉禁止.</p> <p>01 = Px.n 上拉使能.</p> <p>10 = Px.n 下拉使能.</p> <p><b>注意1:</b></p> <p>基本上，上拉控制和下拉控制有如下限制</p> <p>仅当MODEn设置成三态或开漏模式时，单独上拉控制才有效。</p> <p>仅当MODEn设置成三态模式。单独下拉控制才有效。</p> <p>在三态模式，上拉和下拉都设置成1，保持I/O在三态模式。</p> <p><b>注意2:</b></p> <p>端口A/B/E/G: 最大. n=15</p> <p>端口C/D: 最大. n=14</p> <p>端口F/H: 最大. n=11</p>

**GPIO\_DBCTL 中断消抖控制寄存器**

寄存器	偏移地址	R/W	描述	复位值
GPIO_DBCTL	GPIO_BA+0x440	R/W	中断消抖控制寄存器	0x0000_0020

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved		ICLKON	DBCLKSRC	DBCLKSEL			

位	描述	
[31:6]	Reserved	保留
[5]	ICLKON	<p><b>中断时钟 开 模式</b></p> <p>0 = 边沿侦测电路仅在 I/O管脚对应的RHIEN (Px_INTEN[n+16])/FLIEN (Px_INTEN[n])位置 1有效.</p> <p>1 = 复位之后所有 I/O边沿侦测电路使能.</p> <p><b>注意:</b> 如果没有相关的特定应用, 建议关掉时钟以减少耗电</p>
[4]	DBCLKSRC	<p><b>消抖计数器时钟源选择</b></p> <p>0 = 去抖动计数器时钟源为 HCLK</p> <p>1 = 去抖动计数器时钟源为内部 10 KHz (LIRC) 时钟</p>

位	描述
[3:0]	<b>DBCLKSEL</b> 消抖采样周期选择 0000 = 采样中断信号，每1个时钟一次 0001 = 采样中断信号，每2个时钟一次 0010 = 采样中断信号，每4个时钟一次 0011 = 采样中断信号，每8个时钟一次 0100 = 采样中断信号，每16个时钟一次 0101 = 采样中断信号，每32个时钟一次 0110 = 采样中断信号，每64个时钟一次 0111 = 采样中断信号，每128个时钟一次 1000 = 采样中断信号，每256个时钟一次 1001 = 采样中断信号，每 $2^*256$ 个时钟一次. 1010 = 采样中断信号，每 $4^*256$ 个时钟一次. 1011 = 采样中断信号，每 $8^*256$ 个时钟一次 1100 = 采样中断信号，每 $16^*256$ 个时钟一次 1101 = 采样中断信号，每 $32^*256$ 个时钟一次 1110 = 采样中断信号，每 $64^*256$ 个时钟一次 1111 = 采样中断信号，每 $128^*256$ 个时钟一次

Pxn\_PDIO GPIO Px.n 管脚数据/输入输出寄存器

寄存器	偏移地址	R/W	描述	复位值
PA <sub>n</sub> _PDIO n=0,1..15	GPIO_BA+0x800+(0x04 * n)	R/W	GPIO PA.n管脚数据输入/输出寄存器	0x0000_000X
PB <sub>n</sub> _PDIO n=0,1..15	GPIO_BA+0x840+(0x04 * n)	R/W	GPIO PB.n 管脚数据输入/输出寄存器	0x0000_000X
PC <sub>n</sub> _PDIO n=0,1..15	GPIO_BA+0x880+(0x04 * n)	R/W	GPIO PC.n 管脚数据输入/输出寄存器	0x0000_000X
PD <sub>n</sub> _PDIO n=0,1..15	GPIO_BA+0x8C0+(0x04 * n)	R/W	GPIO PD.n 管脚数据输入/输出寄存器	0x0000_000X
PE <sub>n</sub> _PDIO n=0,1..15	GPIO_BA+0x900+(0x04 * n)	R/W	GPIO PE.n 管脚数据输入/输出寄存器	0x0000_000X
PF <sub>n</sub> _PDIO n=0,1..15	GPIO_BA+0x940+(0x04 * n)	R/W	GPIO PF.n 管脚数据输入/输出寄存器	0x0000_000X
PG <sub>n</sub> _PDIO n=0,1..12	GPIO_BA+0x980+(0x04 * n)	R/W	GPIO PG.n 管脚数据输入/输出寄存器	0x0000_000X
PH <sub>n</sub> _PDIO n=0,1..12	GPIO_BA+0x9C0+(0x04 * n)	R/W	GPIO PH.n 管脚数据输入/输出寄存器	0x0000_000X

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved							PDIO

位	描述	
[31:1]	Reserved	保留
[0]	PDIO	<p><b>GPIO Px.n管脚 数据输入/输出控制</b>  写该位可以控制一个GPIO管脚的输出值  0 = 设置相应GPIO管脚为低  1 = 设置相应GPIO管脚为高  读该寄存器得到GPIO管脚状态  例如：写PA0_PDIO 即把值写到DOUT (Px_DOUT[0]) 位上，读PA0_PDIO 即读取PIN (PA_PIN[0])的值。  <b>注意1：</b>写操作不受DATMSK (Px_DATMSK[n])影响</p>

		<p><b>注意2:</b></p> <p>端口A/B/E/G: 最大. n=15 端口C/D: 最大. n=14 端口F/H: 最大. n=11</p>
--	--	---

## 6.6 PDMA 控制器

### 6.6.1 概述

直接存储器存取(PDMA)用于高速数据传输。PDMA控制器可以从一个地址到另一个地址传输数据，无需CPU介入。这样做的好处是减少CPU的工作量，把节省下的CPU资源做其他应用。PDMA控制器包含16个通道，每个通道支持内存和外设之间的数据传输和内存与内存之间的数据传输。

### 6.6.2 特性

- 支持16个可独立配置的通道
- 支持2种优先级选择：固定优先级(fixed priority)或轮循调度优先级(round-robin priority))
- 传输数据宽度支持8位，16位，32位
- 支持源地址与目的地址方向递增或固定，支持字节，半字，字传输。
- 支持软件，SPI, UART, DAC, ADC 和 PWM请求
- 支持Scatter-Gather模式，通过描述符链表执行灵活的数据传输。
- 支持单一和批量传输类型
- 通道0和通道1支持超时功能
- 通道0到通道5支持跨步功能

### 6.6.3 框图

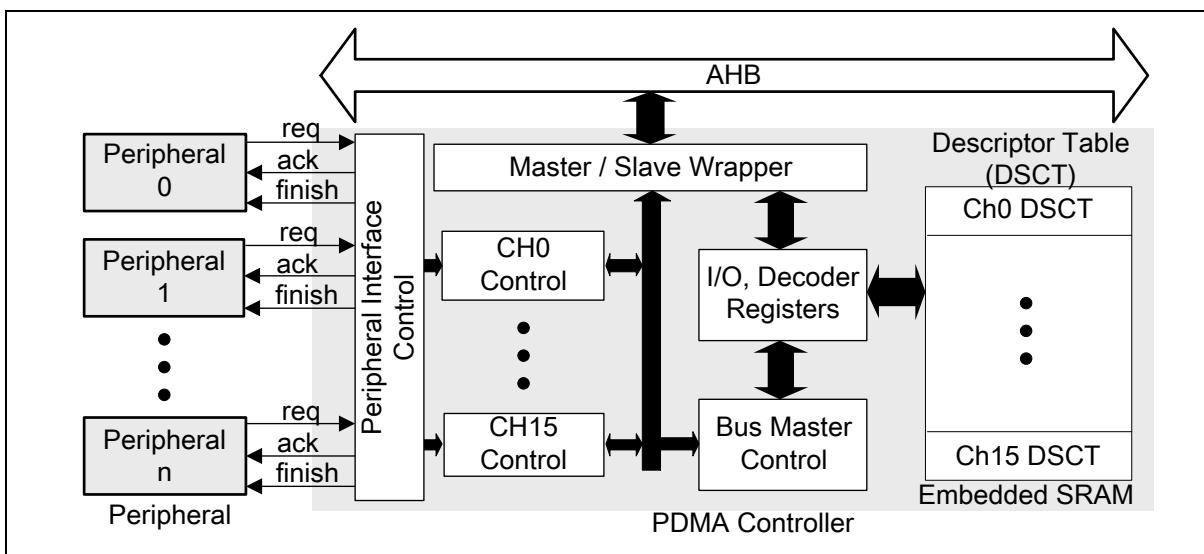


图 6.6-1 PDMA 控制器框图

### 6.6.4 基本配置

- 时钟源配置
  - 在PDMACKEN (CLK\_AHBCLOCK [1])中，使能 PDMA 控制器时钟。
- 复位配置

- 在PDMARST (SYS\_IPRST0[2])中，复位PDMA控制器。

### 6.6.5 功能描述

直接存储器存取(PDMA)控制器可以从一个地址到另一个地址传输数据，无需CPU介入。PDMA有16个独立通道，因为每次只有一个通道工作，因此，PDMA控制器支持两种通道优先级：固定优先级和调度优先级(round-robin priority)，PDMA控制器通道执行的优先级是从高到低的。PDMA控制器支持两种运行模式：基本模式和Scatter-gather模式。基本模式用于按照一个传输描述符表格传输数据。Scatter-gather模式对于每个PDMA都有多个传输描述符表格，所以PDMA控制器通过这些表格，实现灵活的数据传输，传输描述符表数据结构包含了传送信息，例如：传输源地址，传输目的地址，传输计数，批量传输数据大小，传输类型和操作模式。下图是描述符表(DSCT)数据结构。

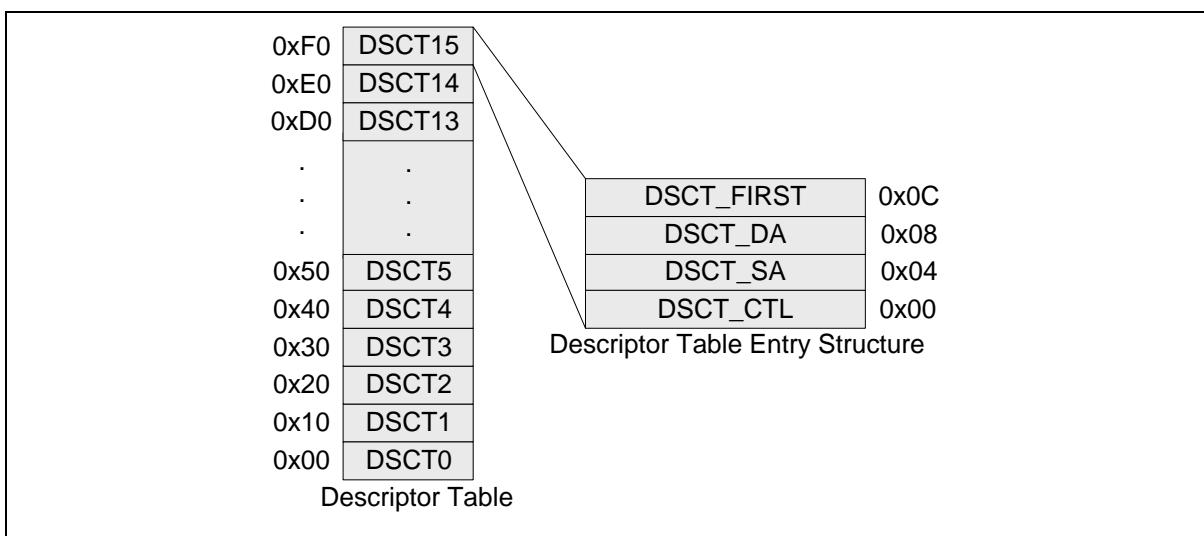


图 6.6-2 描述符表入口结构

PDMA控制器也支持单一和批量数据的传输类型，请求源可以是软件请求或接口请求。内存之间的数据传输是使用软件请求。单一传输的意思是软件或接口准备好传输一个数据(每个数据需要一次请求)，批量传输的意思是软件或接口将传输多个数据(多个数据仅需一次请求)。

#### 6.6.5.1 通道优先级

PDMA控制器支持两级通道优先等级，包括固定优先级和轮循调度优先级(round-robin priority)。固定优先级比轮循调度优先级(round-robin priority)的优先级别更高。如果多路通道设定为固定优先级或调度优先级(round-robin priority)，高通道的优先级别也高。优先级原则如下表。

PDMA_PRISET	通道号	优先级设定	优先级递减顺序
1	15	通道15, 固定优先级	最高
1	14	通道14, 固定优先级	---
---	---	---	---
1	0	通道0, 固定优先级	---
0	15	通道15, 轮循调度优先级(Round-Robin Priority)	---
0	14	通道14, 轮循调度优先级(Round-Robin Priority)	---

---	---	---	---
0	0	通道0, 轮循调度优先级(Round-Robin Priority)	最低

表 6.6-1 通道优先级表

#### 6.6.5.2 PDMA 操作模式

PDMA 支持两种操作模式，包括：基本模式和 Scatter-Gather 模式。

##### 基本模式

基本模式用于执行一个描述符表格的传输模式。该模式用于内存与内存之间的数据传输或接口与内存之间的数据传输。PDMA 控制器操作模式可以通过寄存器 OPMODE (PDMA\_DSCTn\_CTL[1:0], n 代表 PDMA 通道数) 设定，默认设置是在空闲状态(OPMODE (PDMA\_DSCTn\_CTL[1:0]) = 0x0)，推荐用户设定描述符表为空闲状态。如果操作模式不是空闲状态，用户重新配置通道设定可能会引起操作错误。

用户必须填写传输计数寄存器 TXCNT (PDMA\_DSCTn\_CTL31:16]) 和传输宽度选择寄存器 TXWIDTH (PDMA\_DSCTn\_CTL[13:12]), 目的地址递增大小寄存器 DAINC (PDMA\_DSCTn\_CTL[11:10]), 源地址递增大小寄存器 SAINC (PDMA\_DSCTn\_CTL[9:8]), 批量大小寄存器 BURSIZE (PDMA\_DSCTn\_CTL[6:4]) 和传输类型寄存器 TXTYPE(PDMA\_DSCTn\_CTL[2]), 然后 PDMA 控制器将在接收到请求信号后执行传输操作。如果相应的 PDMA 中断位 INTENn (PDMA\_INTEN[15:0]) 使能，传输完成后将给 CPU 产生一个中断，操作模式将被更新为空闲模式，如下图。如果软件配置操作模式为空闲状态，PDMA 控制器不会执行任何传输，并清除这个操作请求。如果相应的 PDMA 中断位被使能，完成这个任务后也会给 CPU 产生中断。

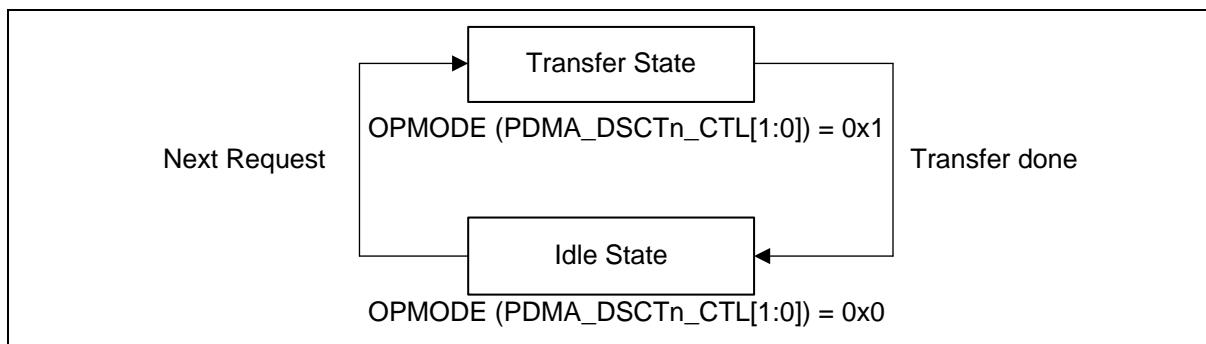


图 6.6-3 基本模式的有限状态机

##### Scatter-Gather 模式

Scatter-Gather 模式是一个复杂模式，通过如下图描述符链表设定，可以实现灵活的数据传输。通过该模式，用户可以实现外设的回环访问。用于多路 PDMA 任务或用于系统内存的不同位置(非相邻位置)的数据搬移。Scatter-gather 模式仅需要一个请求来完成所有表格入口任务，直到最后一个任务。这也意味着 Scatter-Gather 模式仅用于在内存与内存之间传输数据，没有握手信号。

在 Scatter-Gather 模式，这个表用于跳转到下一个表的入口。第一个任务不会做数据搬移操作。如果相应的 PDMA 中断位使能，且 TBINTDIS (PDMA\_DSCTn\_CTL[7])=0，完成每个任务后，将给 CPU 产生一个中断，(任务完成，TBINTDIS 位为“0”，会置位相应的寄存器 TDIFn (PDMA\_TDSTS[15:0]) 标志，如果 TBINTDIS 位为“1”，禁用 TDIFn)。

如果触发了通道 15，在 Scatter-Gather(OPTION (OPMODE (PDMA\_DSCTn\_CTL[1:0]) = 0x2) 模式，硬件将把寄存器 PDMA\_DSCTn\_FIRST (链接地址) 和 PDMA\_SCATBA (基址) 相加来获取真正的 PDMA 任务地址信息。例如，基地址是 0x2000\_0000 (PDMA\_SCATBA 寄存器中仅 MSB 16 位有效)，目前链接地址是 0x0000\_0100 (在寄存器 PDMA\_DSCTn\_FIRST 中，仅 LSB 16 位(不包括[1:0]) 有效)，那

么，下一个DSCT的起始地址是0x2000\_0100。

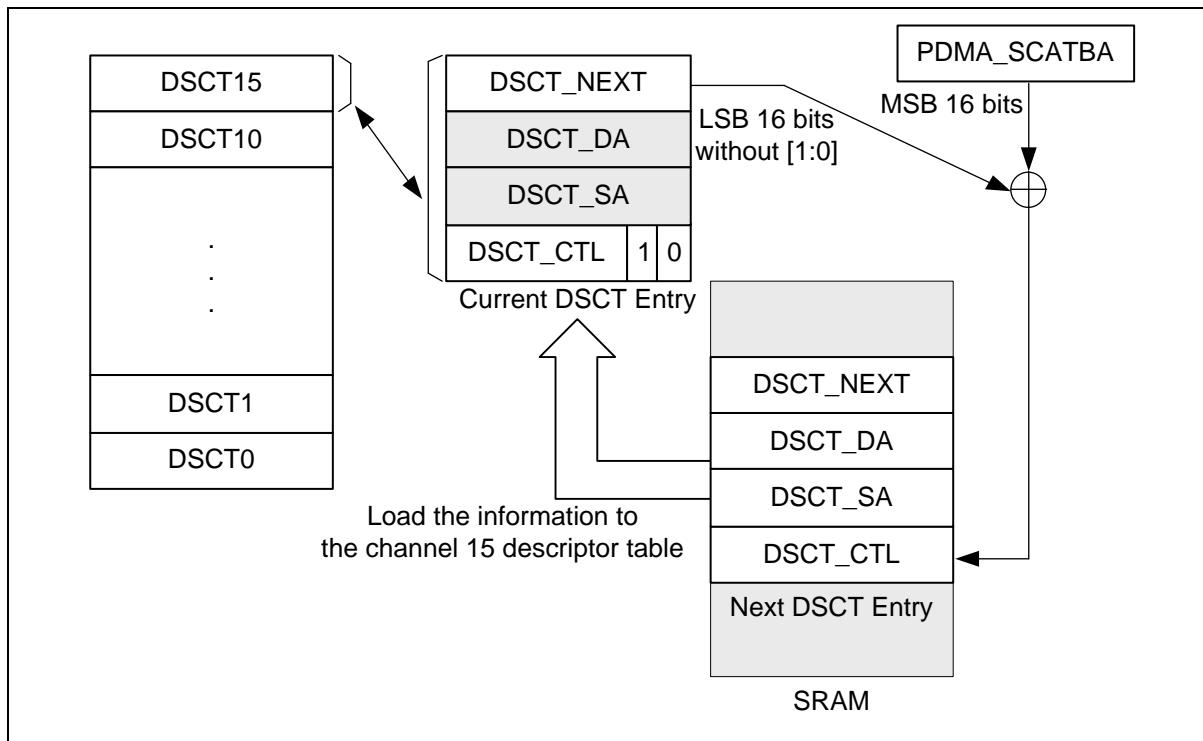


图 6.6-4 描述符表格链接单结构

上述链接列表操作是Scatter-Gather模式的DSCT状态。下图所示，当加载信息结束后，通过该信息将自动进入搬移数据状态并开始传输。然而，如果下一个PDMA信息也是Scatter-Gather模式，当前任务结束后，硬件将抓下一个PDMA块信息。Scatter-Gather模式只有在PDMA控制器操作模式切换到基本模式并完成最后一次传输或者直接切换到空闲状态后才会结束。

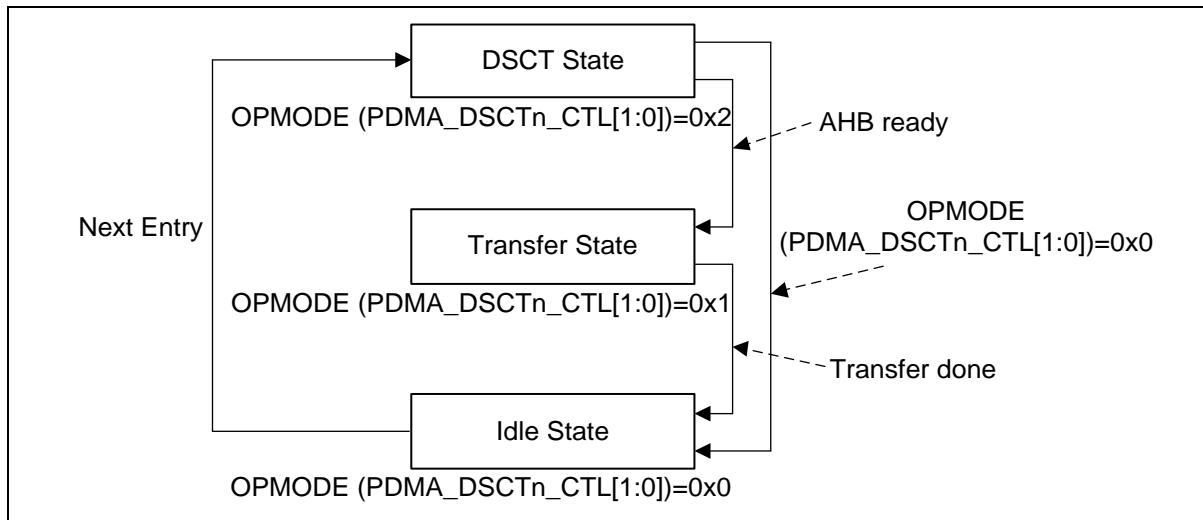


图 6.6-5 Scatter-Gather 模式的有限状态机

### 6.6.5.3 传输类型

PDMA 控制器支持两种传输模式：单一传输和批量传输模式，通过寄存器 TXTYPE (PDMA\_DSCTn\_CTL[2]) 设定。

当PDMA控制器运行在单一传输模式，每搬移一个数据需要一次请求，当搬移一次数据，寄存器TXCNT (PDMA\_DSCTn\_CTL[31:16])会减1，直到寄存器TXCNT (PDMA\_DSCTn\_CTL[31:16])为0，搬移才会完成。在该模式，寄存器BURSIZE (PDMA\_DSCTn\_CTL[6:4])不用于控制搬移数据大小。BURSIZE (PDMA\_DSCTn\_CTL[6:4])数值是固定的。

在批量搬移模式，PDMA控制器搬移TXCNT (PDMA\_DSCTn\_CTL[31:16])个数据，仅需一次请求。当搬移BURSIZE (PDMA\_DSCTn\_CTL[6:4])个数据后，寄存器TXCNT (PDMA\_DSCTn\_CTL[31:16])中的BURSIZE数目会减去BURSIZE。直到TXCNT (PDMA\_DSCTn\_CTL[31:16])递减为0，搬移数据才完成。注意批量传输模式仅用于PDMA控制器在内存与内存之间做批量数据的搬移，对内存到外设或外设到内存传输，用户需要用单次请求模式请注意，PDMA在闪存和存储器之间传输数据应该在MCU进入空闲或掉电模式之前完成，以防止访问错误的数据

下图是基本传输模式的单一传输和批量传输模式举例。在这个例子，通道1使用单一传输模式，TXCNT (PDMA\_DSCTn\_CTL[31:16]) = 127。通道0使用批量传输模式，BURSIZE (PDMA\_DSCTn\_CTL[6:4]) = 128 并且TXCNT (PDMA\_DSCTn\_CTL[31:16]) = 255。操作顺序如下：

1. 通道0与通道1同时接收到触发信号。
2. 默认通道1的优先级高于通道0；PDMA控制器将先加载通道1的描述符表并执行。但通道1是单一传输模式，所以PDMA控制器仅传输一个数据。
3. 然后，PDMA控制器切换到通道0并加载通道0的描述符表。通道0是批量传输模式，大小是128个。所以，PDMA控制器将传输128个数据。
4. 当通道0搬运128个数据，通道1得到另一个请求信号，所以当通道0搬运完128个数据，PDMA控制器将切回通道1，来搬运通道1的下一个数据。
5. 当通道1搬运完数据，PDMA控制器切换到低优先级的通道0来继续搬运下一组128个数据。
6. 当通道0完成256次数据搬移，通道1完成128次数据搬移，PDMA将完成数据搬移。

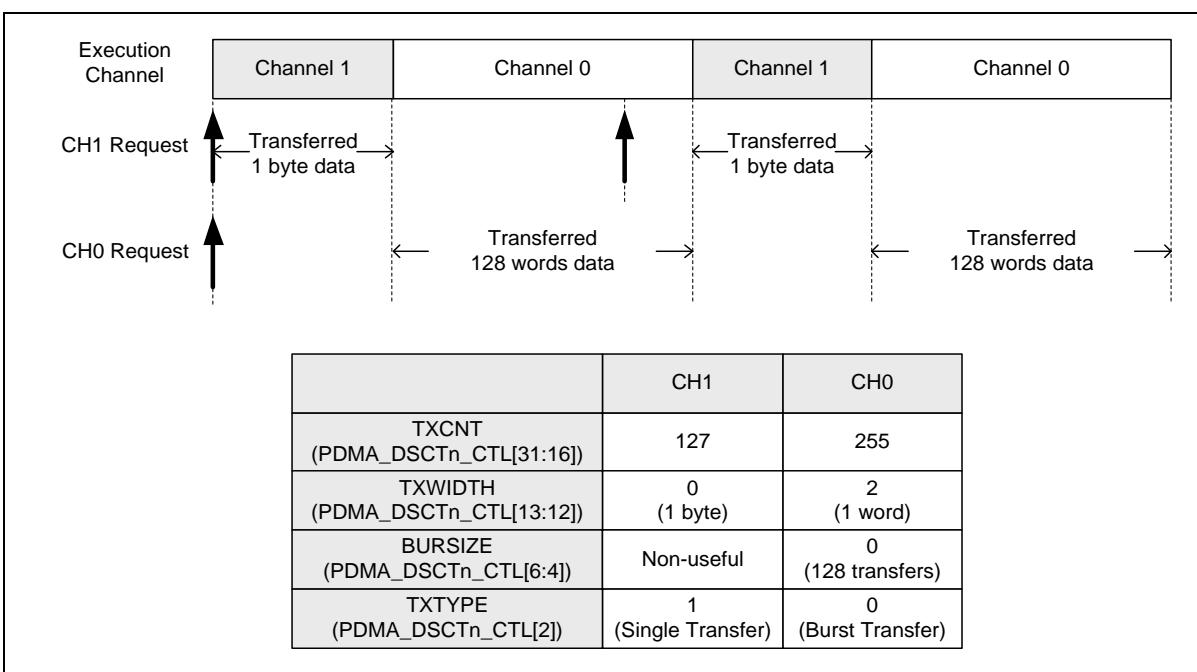


图 6.6-6 基本模式的单字传输和批量传输举例

#### 6.6.5.4 通道超时

仅PDMA通道0和通道1支持超时功能。当传输通道使能，且选择到外设，对应通道超时TOUTENn

(PDMA\_TOUTEN [n], n=0,1)功能使能，然后当通道接收到来自外设的触发信号，通道对应的超时计数器开始从0向上计数。

超时计数器基于HCLK预分频的输出，这由对应通道的TOUTPSCn (PDMA\_TOUTPSC [2+4n:4n], n=0,1)来设置。如果超时计数器从0开始向上计数到对应通道的TOCn (PDMA\_TOCO\_1 [16(n+1)-1:16n], n=0,1)的值，当对应的TOUTIENn (PDMA\_TOUTIEN [n], n=0,1)使能，PDMA控制器会产生中断信号。当超时发送，对应通道的REQTOFn (PDMA\_INTSTS [n+8], n=0,1)会被置位，表示通道超时发生。

当计数器计数到TOCn (PDMA\_TOCO\_1 [16(n+1)-1:16n], n=0,1)，接收到触发信号超时功能禁止或芯片进入掉电模式。超时计数器复位到0。

图 6.6-7 是超时计数器工作的例子，操作顺序如下描述：

1. 当通过设置TOUTEN0(PDMA\_TOUTEN[0])为1使能超时功能，通道0超时计数器还没有开始计数。
2. 当接收到第一个外设请求。超时计数器从0开始计数到TOC0(PDMA\_TOCO\_1[15:0])的值。
3. 超时计数器由接收到第二个外设请求复位。
4. 当超时计数器数到5，通道0请求超时标志(REQTOF0(PDMA\_INTSTS[8]))被置高。超时计数器保持从0到5计数，用户可以清REQTOF0标志，然后通过查询REQTOF0标志检查下次超时的发生。
5. 当超时功能禁止超时计数器复位为0。

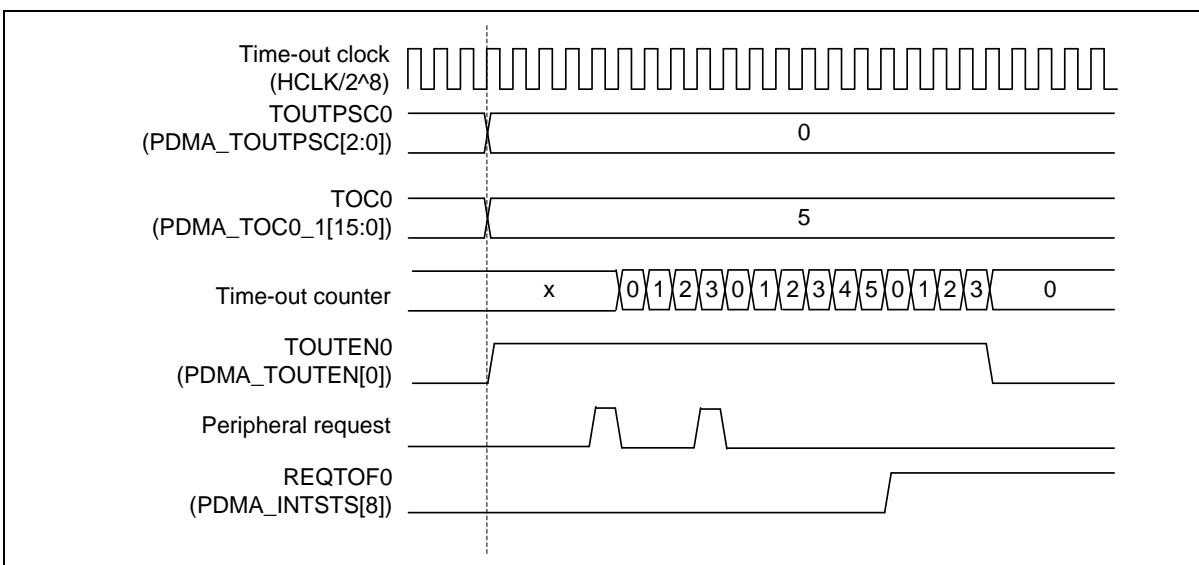


图 6.6-7 PDMA 通道 0 超时计数器操作例子

#### 6.6.5.5 跨步功能

PDMA支持通道0到通道5这6个通道的跨步功能。跨步功能可以从一个地址传输数据到另一个地址，且支持带跨步的块传输。当操作在跨步功能，传输地址可以固定或连续递增。

设置 STRIDE\_EN (PDMA\_DSCTn\_CTL[15]) 位使能跨步功能，然后写一个有效的源地址到 PDMA\_DSCTn\_SA 寄存器，一个源地址偏移计数到 SASOL (PDMA\_ASOCRn[15:0]) 寄存器，一个目的地址到 PDMA\_DSCTn\_DA 寄存器，一个目的地址偏移计数到 DASOL (PDMA\_ASOCRn[31:16])，写一个传输计数到 TXCNT (PDMA\_DSCTn\_CTL) 寄存器，一个跨步传输计数到 STC (PDMA\_STCn[15:0])。

接下来触发SWREQn (PDMA\_SWREQ[5:0])。DMA会继续传输直到TXCNT (PDMA\_DSCTn\_CTL)下数到0。下图表示在源内存和目的内存之间块传输的关系。跨步功能也支持外设到内存或内存到外设传输。

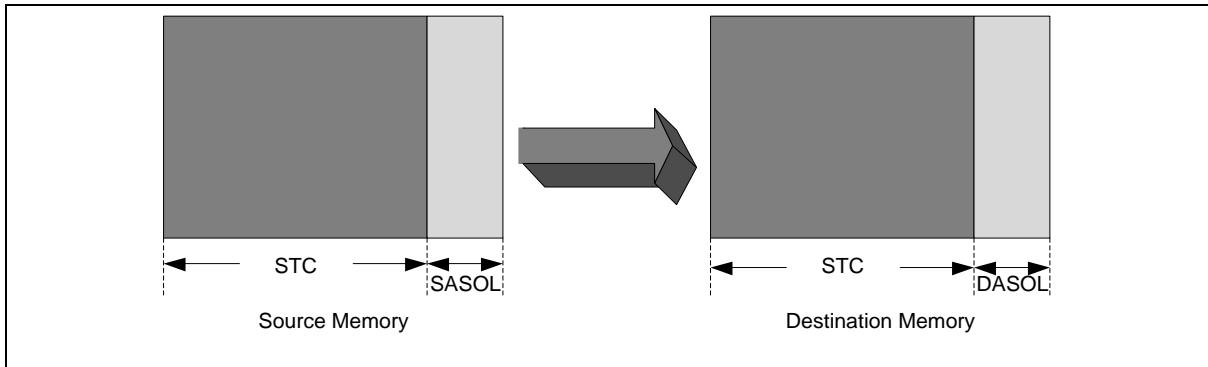


图 6.6-8 跨步功能块传输

### 6.6.6 寄存器映射

R: 只读, W: 只写, R/W: 可读写

寄存器	偏移量	R/W	描述	复位值
<b>PDMA基地址:</b>				
PDMA_BA	= 0x4000_8000			
DSCT_CTL_BA	= 0x4000_8000			
DSCT_SA_BA	= 0x4000_8004			
DSCT_DA_BA	= 0x4000_8008			
DSCT_NEXT_BA	= 0x4000_800c			
CURSCAT_BA	= 0x4000_8100			
PDMA_DSCTn_CTL n = 0,1..15	DSCT_CTL_BA + 0x10 * n	R/W	PDMA通道n描述符表控制寄存器	0xXXXX_XXXX
PDMA_DSCTn_SA n = 0,1..15	DSCT_SA_BA + 0x10 * n	R/W	PDMA通道n源地址寄存器	0xXXXX_XXXX
PDMA_DSCTn_DA n = 0,1..15	DSCT_DA_BA + 0x10 * n	R/W	PDMA通道n目的地址寄存器	0xXXXX_XXXX
PDMA_DSCTn_NEXT n = 0,1..15	DSCT_NEXT_BA + 0x10 * n	R/W	PDMA通道n的第一个Scatter-Gather描述符表偏移地址	0xXXXX_XXXX
PDMA_CURSCATn n = 0,1..15	CURSCAT_BA + 0x004 * n	R	PDMA通道n的目前Scatter-Gather描述符表地址	0xXXXX_XXXX
PDMA_CHCTL	PDMA_BA + 0x400	R/W	PDMA通道控制寄存器	0x0000_0000
PDMA_PAUSE	PDMA_BA + 0x404	W	PDMA 传输停止控制寄存器	0x0000_0000
PDMA_SWREQ	PDMA_BA + 0x408	W	PDMA软件请求寄存器	0x0000_0000
PDMA_TRGSTS	PDMA_BA + 0x40C	R	PDMA 通道请求状态寄存器	0x0000_0000
PDMA_PRISET	PDMA_BA + 0x410	R/W	PDMA 固定优先级设定寄存器	0x0000_0000
PDMA_PRICLR	PDMA_BA + 0x414	W	PDMA 固定优先级清除寄存器	0x0000_0000
PDMA_INTEN	PDMA_BA + 0x418	R/W	PDMA 中断使能寄存器	0x0000_0000
PDMA_INTSTS	PDMA_BA + 0x41C	R/W	PDMA 中断状态寄存器	0x0000_0000
PDMA_ABTESTS	PDMA_BA + 0x420	R/W	PDMA 通道读/写目标终止标志寄存器	0x0000_0000
PDMA_TDSTS	PDMA_BA + 0x424	R/W	PDMA 通道传输完成标志寄存器	0x0000_0000
PDMA_ALIGN	PDMA_BA + 0x428	R/W	PDMA 传输对齐状态寄存器	0x0000_0000
PDMA_TACTSTS	PDMA_BA + 0x42C	R	PDMA 传输激活标志寄存器	0x0000_0000
PDMA_TOUTPSC	PDMA_BA + 0x430	R/W	PDMA 超时预分频器寄存器	0x0000_0000
PDMA_TOUTEN	PDMA_BA + 0x434	R/W	PDMA 超时使能寄存器	0x0000_0000
PDMA_TOUTIEN	PDMA_BA + 0x438	R/W	PDMA 超时中断使能寄存器	0x0000_0000

<b>PDMA_SCATBA</b>	PDMA_BA + 0x43C	R/W	PDMA Scatter-Gather描述符表基地址寄存器	0x2000_0000
<b>PDMA_TOCO_1</b>	PDMA_BA + 0x440	R/W	PDMA 超时计数器通道1和通道0寄存器	0xFFFF_FFFF
<b>PDMA_CHRST</b>	PDMA_BA + 0x460	R/W	PDMA 通道复位寄存器	0x0000_0000
<b>PDMA_SPI</b>	PDMA_BA + 0x464	R/W	PDMA SPI性能提升寄存器	0x0000_0000
<b>PDMA_REQSEL0_3</b>	PDMA_BA + 0x480	R/W	PDMA 请求源选择寄存器0	0x0000_0000
<b>PDMA_REQSEL4_7</b>	PDMA_BA + 0x484	R/W	PDMA请求源选择寄存器1	0x0000_0000
<b>PDMA_REQSEL8_11</b>	PDMA_BA + 0x488	R/W	PDMA请求源选择寄存器2	0x0000_0000
<b>PDMA_REQSEL12_15</b>	PDMA_BA + 0x48C	R/W	PDMA请求源选择寄存器3	0x0000_0000
<b>PDMA_STCR0</b>	PDMA_BA + 0x500	R/W	PDMA 通道 0的跨步传输计数寄存器	0x0000_0000
<b>PDMA_ASOCR0</b>	PDMA_BA + 0x504	R/W	PDMA 通道 0的地址跨步偏移量寄存器	0x0000_0000
<b>PDMA_STCR1</b>	PDMA_BA + 0x508	R/W	PDMA 通道 1的跨步传输计数寄存器	0x0000_0000
<b>PDMA_ASOCR1</b>	PDMA_BA + 0x50C	R/W	PDMA 通道 1的地址跨步偏移量寄存器	0x0000_0000
<b>PDMA_STCR2</b>	PDMA_BA + 0x510	R/W	PDMA 通道 2的跨步传输计数寄存器	0x0000_0000
<b>PDMA_ASOCR2</b>	PDMA_BA + 0x514	R/W	PDMA 通道 2的地址跨步偏移量寄存器	0x0000_0000
<b>PDMA_STCR3</b>	PDMA_BA + 0x518	R/W	PDMA 通道 3的跨步传输计数寄存器	0x0000_0000
<b>PDMA_ASOCR3</b>	PDMA_BA + 0x51C	R/W	PDMA 通道 3的地址跨步偏移量寄存器	0x0000_0000
<b>PDMA_STCR4</b>	PDMA_BA + 0x520	R/W	PDMA 通道 4的跨步传输计数寄存器	0x0000_0000
<b>PDMA_ASOCR4</b>	PDMA_BA + 0x524	R/W	PDMA 通道 4的地址跨步偏移量寄存器	0x0000_0000
<b>PDMA_STCR5</b>	PDMA_BA + 0x528	R/W	PDMA 通道 5的跨步传输计数寄存器	0x0000_0000
<b>PDMA_ASOCR5</b>	PDMA_BA + 0x52C	R/W	PDMA 通道 5的地址跨步偏移量寄存器	0x0000_0000

寄存器	偏移地址	R/W	描述	复位值
<b>PDMA基地址:</b>				
<b>PDMA_BA = 0x4000_8000</b>				
<b>DSCT_CTL_BA = 0x4000_8000</b>				
<b>DSCT_SA_BA = 0x4000_8004</b>				
<b>DSCT_DA_BA = 0x4000_8008</b>				
<b>DSCT_NEXT_BA = 0x4000_800c</b>				
<b>CURSCAT_BA = 0x4000_8100</b>				
<b>PDMA_DSCTn_CTL n = 0,1..15</b>	DSCT_CTL_BA + 0x10 * n	R/W	PDMA通道n描述符表控制寄存器	0xFFFF_FFFF

<b>PDMA_DSCTn_SA n = 0,1..15</b>	DSCT_SA_BA + 0x10 * n	R/W	PDMA通道n源地址寄存器	0xXXXX_XXXX
<b>PDMA_DSCTn_DA n = 0,1..15</b>	DSCT_DA_BA + 0x10 * n	R/W	PDMA通道n目的地址寄存器	0xXXXX_XXXX
<b>PDMA_DSCTn_NE XT n = 0,1..15</b>	DSCT_NEXT_BA + 0x10 * n	R/W	PDMA通道n的第一个Scatter-Gather描述符表偏移地址	0xXXXX_XXXX
<b>PDMA_CURSCATn n = 0,1..15</b>	CURSCAT_BA + 0x004 * n	R	PDMA通道n的目前Scatter-Gather描述符表地址	0xXXXX_XXXX
<b>PDMA_CHCTL</b>	PDMA_BA + 0x400	R/W	PDMA通道控制寄存器	0x0000_0000
<b>PDMA_PAUSE</b>	PDMA_BA + 0x404	W	PDMA 传输停止控制寄存器	0x0000_0000
<b>PDMA_SWREQ</b>	PDMA_BA + 0x408	W	PDMA软件请求寄存器	0x0000_0000
<b>PDMA_TRGSTS</b>	PDMA_BA + 0x40C	R	PDMA 通道请求状态寄存器	0x0000_0000
<b>PDMA_PRISET</b>	PDMA_BA + 0x410	R/W	PDMA 固定优先级设定寄存器	0x0000_0000
<b>PDMA_PRICLR</b>	PDMA_BA + 0x414	W	PDMA 固定优先级清除寄存器	0x0000_0000
<b>PDMA_INTEN</b>	PDMA_BA + 0x418	R/W	PDMA中断使能寄存器	0x0000_0000
<b>PDMA_INTSTS</b>	PDMA_BA + 0x41C	R/W	PDMA 中断状态寄存器	0x0000_0000
<b>PDMA_ABTSSTS</b>	PDMA_BA + 0x420	R/W	PDMA 通道读/写目标终止标志寄存器	0x0000_0000
<b>PDMA_TDSTS</b>	PDMA_BA + 0x424	R/W	PDMA 通道传输完成标志寄存器	0x0000_0000
<b>PDMA_ALIGN</b>	PDMA_BA + 0x428	R/W	PDMA 传输对齐状态寄存器	0x0000_0000
<b>PDMA_TACTSTS</b>	PDMA_BA + 0x42C	R	PDMA 传输激活标志寄存器	0x0000_0000
<b>PDMA_TOUTPSC</b>	PDMA_BA + 0x430	R/W	PDMA 超时预分频器寄存器	0x0000_0000
<b>PDMA_TOUTEN</b>	PDMA_BA + 0x434	R/W	PDMA 超时使能寄存器	0x0000_0000
<b>PDMA_TOUTIEN</b>	PDMA_BA + 0x438	R/W	PDMA 超时中断使能寄存器	0x0000_0000
<b>PDMA_SCATBA</b>	PDMA_BA + 0x43C	R/W	PDMA Scatter-Gather描述符表基址寄存器	0x2000_0000
<b>PDMA_TOCO_1</b>	PDMA_BA + 0x440	R/W	PDMA 超时计数器通道1和通道0寄存器	0xFFFF_FFFF
<b>PDMA_CHRST</b>	PDMA_BA + 0x460	R/W	PDMA 通道复位寄存器	0x0000_0000
<b>PDMA_SPI</b>	PDMA_BA + 0x464	R/W	PDMA SPI性能提升寄存器	0x0000_0000
<b>PDMA_REQSEL0_3</b>	PDMA_BA + 0x480	R/W	PDMA 请求源选择寄存器0	0x0000_0000
<b>PDMA_REQSEL4_7</b>	PDMA_BA + 0x484	R/W	PDMA请求源选择寄存器1	0x0000_0000
<b>PDMA_REQSEL8_11</b>	PDMA_BA + 0x488	R/W	PDMA请求源选择寄存器2	0x0000_0000
<b>PDMA_REQSEL12_15</b>	PDMA_BA + 0x48C	R/W	PDMA请求源选择寄存器3	0x0000_0000
<b>PDMA_STCR0</b>	PDMA_BA + 0x500	R/W	PDMA 通道 0的跨步传输计数寄存器	0x0000_0000
<b>PDMA_ASOCR0</b>	PDMA_BA + 0x504	R/W	PDMA 通道 0的地址跨步偏移量寄存器	0x0000_0000

PDMA_STCR1	PDMA_BA + 0x508	R/W	PDMA 通道 1的跨步传输计数寄存器	0x0000_0000
PDMA_ASOCR1	PDMA_BA + 0x50C	R/W	PDMA 通道 1的地址跨步偏移量寄存器	0x0000_0000
PDMA_STCR2	PDMA_BA + 0x510	R/W	PDMA 通道 2的跨步传输计数寄存器	0x0000_0000
PDMA_ASOCR2	PDMA_BA + 0x514	R/W	PDMA 通道 2的地址跨步偏移量寄存器	0x0000_0000
PDMA_STCR3	PDMA_BA + 0x518	R/W	PDMA 通道 3的跨步传输计数寄存器	0x0000_0000
PDMA_ASOCR3	PDMA_BA + 0x51C	R/W	PDMA 通道 3的地址跨步偏移量寄存器	0x0000_0000
PDMA_STCR4	PDMA_BA + 0x520	R/W	PDMA 通道 4的跨步传输计数寄存器	0x0000_0000
PDMA_ASOCR4	PDMA_BA + 0x524	R/W	PDMA 通道 4的地址跨步偏移量寄存器	0x0000_0000
PDMA_STCR5	PDMA_BA + 0x528	R/W	PDMA 通道 5的跨步传输计数寄存器	0x0000_0000
PDMA_ASOCR5	PDMA_BA + 0x52C	R/W	PDMA 通道 5的地址跨步偏移量寄存器	0x0000_0000

### 6.6.7 寄存器描述

#### PDMA\_DSCTn\_CTL 描述符表控制寄存器

寄存器	偏移地址	R/W	描述	复位值
PDMA_DSCTn_CTL	DSCT_CTL_BA + 0x10 * n	R/W	PDMA 通道 n 的描述符表控制寄存器	0xXXXX_XXXX

31	30	29	28	27	26	25	24
TXCNT							
23	22	21	20	19	18	17	16
TXCNT							
15	14	13	12	11	10	9	8
STRIDEEN	Reserved	TXWIDTH		DAINC		SAINC	
7	6	5	4	3	2	1	0
TBINTDIS	BURSIZE			Reserved	TXTYPE	OPMODE	

Bits	Description
[31:16]	<b>TXCNT</b> <b>传输计数</b> TXCNT 代表 PDMA 传输的请求数，实际的传输数是(TXCNT + 1);最大的传输计数是 16384，依据 TXWIDTH 寄存器设定，每次传输数据单位可以是字节，半-字，或字。 <b>注意：</b> 当 PDMA 完成每次数据传输，该寄存器计数会立刻递减。
[15]	<b>STRIDEEN</b> <b>跨步模式使能位</b> 0 = 跨步传输模式禁止。 1 = 跨步传输模式使能。
[14]	<b>Reserved</b> <b>传输应答选择</b> 0 = 当传输完成，传输应答 1 = 当 PDMA 得到传输数据，传输应答
[13:12]	<b>TXWIDTH</b> <b>传输宽度选择</b> 该寄存器用于数据传输宽度设定 00 = 每次操作数据传输宽度是一个字节(8位) 01 = 每次操作数据传输宽度是半-字(16位) 10 = 每次操作数据传输宽度是字(32位) 11 = 保留 <b>注意：</b> PDMA 传输源地址(PDMA_DSCT_SA)和 PDMA 传输目的地址(PDMA_DSCT_DA)应根据 TXWIDTH 的设定对齐。
[11:10]	<b>DAINC</b> <b>目的地址增加</b> 该寄存器用于设定目的地址的递增量大小 11 = 无增量(固定地址) 其他= 增量的大小依据寄存器 TXWIDTH 设定
[9:8]	<b>SAINC</b> <b>源地址增加</b>

Bits	Description
	<p>该寄存器用于设定源地址递增量的大小            11 = 无增量(固定地址).            其他 = 增量的大小依据寄存器TXWIDTH设定</p>
[7]	<p><b>TBINTDIS</b>  <b>禁用表中断</b>            该寄存器是用于是否使能表中断。当PDMA完成传输任务，如果TBINTDIS位使能，将不会产生中断。            0 = 表中断使能            1 = 禁用表中断</p>
[6:4]	<p><b>BURSIZE</b>  <b>批量大小</b>            000 = 128 个数据传输            001 = 64 个数据传输            010 = 32 个数据传输            011 = 16 个数据传输            100 = 8 个数据传输            101 = 4 个数据传输            110 = 2 个数据传输            111 = 1 个数据传输  <b>注意:</b> 这个寄存器只用于批量传输模式</p>
[3]	<b>Reserved</b> 保留.
[2]	<p><b>TXTYPE</b>  <b>传输类型</b>            0 = 批量传输类型            1 = 单一传输类型</p>
[1:0]	<p><b>OPMODE</b>  <b>PDMA 工作模式选择</b>            00 = 空闲状态: 通道停止或该表格完成，当PDMA完成通道表格任务，OPMODE会自动清零到空闲状态。            01 = 基本模式: 描述符表格只有一个任务。当这个任务完成， PDMA_INTSTS[n]将被置位。            10 = Scatter-Gather模式: 在该模式，用户必须把下一个描述符表格地址输入寄存器PDMA_DSCT_FIRST，PDMA控制机将忽略这次任务，然后加载执行下一个任务。            11 = 保留  <b>注意:</b> 在填写传输任务到描述符表之前，用户必须要检查描述符表是否完成。</p>

PDMA\_DSCTn\_SA 起始源地址寄存器

寄存器	偏移地址	R/W	描述	复位值
PDMA_DSCTn_SA	DSCT_SA_BA + 0x10 * n	R/W	PDMA 通道n 源地址寄存器	0xXXXX_XXXX

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

位	描述
[31:0]	SA PDMA 传输源地址寄存器 该寄存器是 32-位 PDMA 的源地址。

PDMA\_DSCTn\_DA 目的地址寄存器

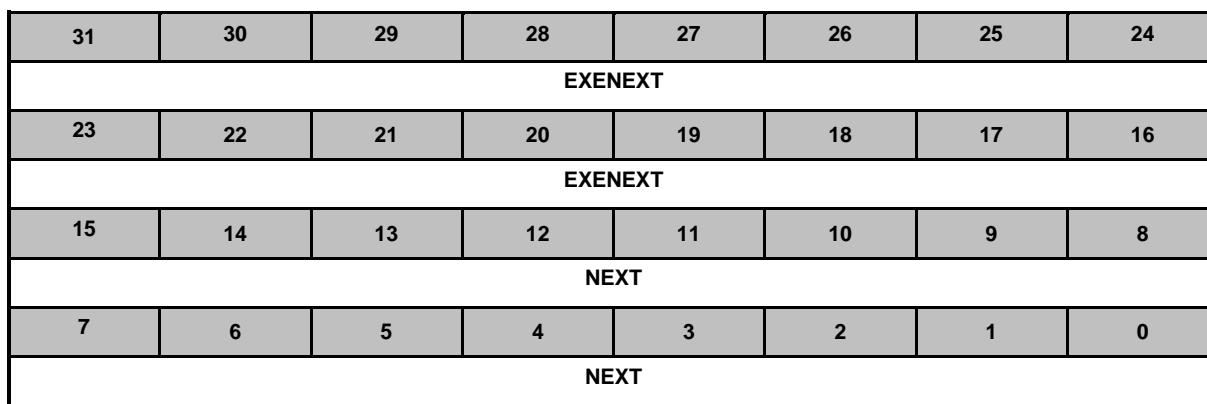
寄存器	偏移地址	R/W	描述	复位值
PDMA_DSCTn_DA	DSCT_DA_BA + 0x10 * n	R/W	PDMA 通道 n 目的地址	0xXXXX_XXXX

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

位	描述
[31:0]	DA PDMA传输目的地址寄存器 该寄存器是 32-位 PDMA 控制器的目的地址寄存器

PDMA\_DSCTn\_NEXT 下一个Scatter-Gather描述表偏移地址

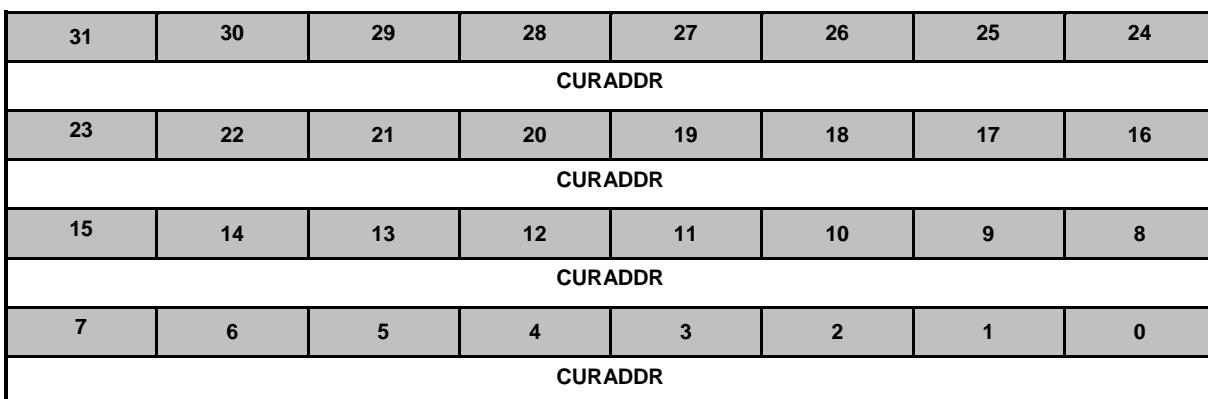
寄存器	偏移地址	R/W	描述	复位值
PDMA_DSCTn_NEXT	DSCT_NEXT_BA 0x10 * n	+R/W	PDMA 通道 n 下一个Scatter-Gather描述符表偏移地址	0xXXXX_XXXX



位	描述	
[31:16]	EXENEXT	<b>PDMA再下一个描述符表偏移地址</b> 该寄存器是系统内存的再下一个描述符表的偏移地址。 <b>注意:</b> 该域写操作无效
[15:0]	NEXT	<b>PDMA下一个描述符表偏移地址</b> 该寄存器是系统内存的下一个描述符表的偏移地址。 <b>写操作:</b> 如果系统内存的基地址是0x2000_0000 (PDMA_SCATBA), 且下一个描述符表地址是从0x2000_0100开始, 该域必须填0x0100。 <b>读操作:</b> 当工作在scatter-gather模式, 最后两位FIRST[1:0]保留。 <b>注意1:</b> 下一个描述符表地址必须是字对齐 <b>注意2:</b> 在填写传输任务到描述符表格之前, 用户必须检查描述符表是否完成。

PDMA\_CURSCATn 当前 Scatter-gather 描述符表格地址

寄存器	偏移地址	R/W	描述	复位值
PDMA_CURSCATn	CURSCAT_BA + 0x004 * n	R	PDMA通道n的当前Scatter-Gather描述符表地址	0xXXXX_XXXX



位	描述
[31:0]	<b>CURADDR</b> PDMA 当前描述地址寄存器(只读) 该寄存器是PDMA控制器的当前32位描述符地址。 <b>注意:</b> 该寄存器只读, 仅用于 Scatter-Gather 模式来表示当前描述符地址。

PDMA\_CHCTL 通道控制寄存器

寄存器	偏移地址	R/W	描述				复位值
PDMA_CHCTL	PDMA_BA + 0x400	R/W	PDMA 通道控制寄存器				0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
CHEN15	CHEN14	CHEN13	CHEN12	CHEN11	CHEN10	CHEN9	CHEN8
7	6	5	4	3	2	1	0
CHEN7	CHEN6	CHEN5	CHEN4	CHEN3	CHEN2	CHEN1	CHEN0

位	描述	
[31:16]	Reserved	保留.
[n] n=0,1..15	CHENn	<p><b>PDMA 通道使能位</b>            该位为"1"是使能PDMA通道n操作，如果不使能该位，该通道无法工作。            0 = 禁用 PDMA通道 [n]            1 = 使能 PDMA通道 [n]</p> <p><b>注意：</b> PDMA_PAUSE 或 PDMA_CHRST寄存器对应的位被置位，会清0该位。</p>

## PDMA\_PAUSE PDMA传输暂停控制寄存器

寄存器	偏移地址	R/W	描述	复位值
PDMA_PAUSE	PDMA_BA + 0x404	W	PDMA Transfer Pause Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
PAUSE15	PAUSE14	PAUSE13	PAUSE12	PAUSE11	PAUSE10	PAUSE9	PAUSE8
7	6	5	4	3	2	1	0
PAUSE7	PAUSE6	PAUSE5	PAUSE4	PAUSE3	PAUSE2	PAUSE1	PAUSE0

位	描述	
[31:16]	Reserved	保留.
[n] n=0,1..15	PAUSEn	<p><b>PDMA 传输暂停控制寄存器(只写)</b></p> <p>用户可以暂停PDMA传输，通过PAUSEn位设定。当软件设定PAUSEn位，将停止正在传输数据的通道，然后清除通道使能位CHEN(PDMA_CHCTL [n], n=0,1..7)及传输请求标志。如果重新使能暂停的通道，剩余的传输会继续。</p> <p><b>位设置:</b></p> <p>0 = 无效 1 = 停止PDMA的通道[n]传输。</p>

PDMA\_SWREQ PDMA 软件请求寄存器

寄存器	偏移地址	R/W	描述	复位值
PDMA_SWREQ	PDMA_BA + 0x408	W	PDMA 软件请求寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
SWREQ15	SWREQ14	SWREQ13	SWREQ12	SWREQ11	SWREQ10	SWREQ9	SWREQ8
7	6	5	4	3	2	1	0
SWREQ7	SWREQ6	SWREQ5	SWREQ4	SWREQ3	SWREQ2	SWREQ1	SWREQ0

位	描述	
[31:16]	Reserved	保留.
[n] n=0,1..15	SWREQn	<p><b>PDMA 软件请求寄存器 (只写)</b></p> <p>设置该位为"1"会在PDMA通道n产生一个软件请求。</p> <p>0 = 无效 1 = 产生一个软件请求</p> <p><b>注意1:</b> 用户可以通过读寄存器PDMA_TRGSTS来知道哪个通道被触发，触发标志可通过软件请求或外设请求触发。</p> <p><b>注意2:</b> 如果用户没有使能相关的PDMA通道，软件请求会被忽略。</p>

PDMA\_TRGSTS PDMA通道请求状态控制器

寄存器	偏移地址	R/W	描述	复位值
PDMA_TRGSTS	PDMA_BA + 0x40C	R	PDMA通道请求状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
REQSTS15	REQSTS14	REQSTS13	REQSTS12	REQSTS11	REQSTS10	REQSTS9	REQSTS8
7	6	5	4	3	2	1	0
REQSTS7	REQSTS6	REQSTS5	REQSTS4	REQSTS3	REQSTS2	REQSTS1	REQSTS0

位	描述	
[31:16]	Reserved	保留.
[n] n=0,1..15	REQSTS <sub>n</sub>	<p><b>PDMA 通道请求状态 (只读)</b></p> <p>这个标志用来表示通道n是否有软件或外设请求。当PDMA控制器完成通道传输，该位会被自动清除。</p> <p>0 = PDMA 通道n无请求 1 = PDMA 通道n有请求</p> <p><b>注意：</b>如果软件通过设定PDMA_PAUSE或PDMA_CHRST 寄存器暂停或复位每个PDMA传输，该位会在完成当前传输后自动清零。</p>

**PDMA\_PRISET PDMA固定优先级设定寄存器**

寄存器	偏移地址	R/W	描述	复位值
PDMA_PRISET	PDMA_BA + 0x410	R/W	PDMA固定优先级设定寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
FPRISET15	FPRISET14	FPRISET13	FPRISET12	FPRISET11	FPRISET10	FPRISET9	FPRISET8
7	6	5	4	3	2	1	0
FPRISET7	FPRISET6	FPRISET5	FPRISET4	FPRISET3	FPRISET2	FPRISET1	FPRISET0

位	描述	
[31:16]	Reserved	保留.
[n] n=0,1..15	FPRISETn	<p><b>PDMA 固定优先级设定寄存器</b>          设置该位为"1" 来使能固定优先级的等级</p> <p><b>写操作:</b>          0 = 无效          1 = 设定PDMA通道n为固定优先级通道</p> <p><b>读操作:</b>          0 = PDMA是轮循调度(round-robin)优先级          1 = PDMA是固定优先级</p> <p><b>注意:</b>该位只设定为固定优先级, 通过设定寄存器PDMA_PRICLR来清除固定优先级.</p>

PDMA\_PRICLR PDMA固定优先级清除寄存器

寄存器	偏移地址	R/W	描述	复位值
PDMA_PRICLR	PDMA_BA + 0x414	W	PDMA固定优先级清除寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
FPRICLR15	FPRICLR14	FPRICLR13	FPRICLR12	FPRICLR11	FPRICLR10	FPRICLR9	FPRICLR8
7	6	5	4	3	2	1	0
FPRICLR7	FPRICLR6	FPRICLR5	FPRICLR4	FPRICLR3	FPRICLR2	FPRICLR1	FPRICLR0

位	描述	
[31:16]	Reserved	保留.
[n] n=0,1..15	FPRICLRn	<p><b>PDMA 固定优先级清除寄存器 (只写)</b>          设定该位为1来清除固定优先级等级          0 = 无效          1 = 清除PDMA通道n的固定优先等级设定。  <b>注意:</b> 用户可以通过读寄存器PDMA_PRISET来知道当前通道的优先等级。</p>

## PDMA\_INTEN PDMA中断使能寄存器

寄存器	偏移地址	R/W	描述	复位值
PDMA_INTEN	PDMA_BA + 0x418	R/W	PDMA中断使能寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
INTEN15	INTEN14	INTEN13	INTEN12	INTEN11	INTEN10	INTEN9	INTEN8
7	6	5	4	3	2	1	0
INTEN7	INTEN6	INTEN5	INTEN4	INTEN3	INTEN2	INTEN1	INTEN0

位	描述	
[31:16]	Reserved	保留.
[n] n=0,1..15	INTENn	<b>PDMA 中断使能寄存器</b> 该寄存器用于使能PDMA的通道n中断 0 = 禁用PDMA通道n的中断 1 = 使能PDMA通道n的中断

PDMA\_INTSTS PDMA中断状态寄存器

寄存器	偏移地址	R/W	描述				复位值
PDMA_INTSTS	PDMA_BA + 0x41C	R/W	PDMA中断状态寄存器				0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved						REQTOF1	REQTOF0
7	6	5	4	3	2	1	0
Reserved					ALIGNF	TDIF	ABTIF

位	描述	
[31:10]	<b>Reserved</b>	保留.
[9]	<b>REQTOF1</b>	<b>通道1请求超时标志</b> 该标志表示PDMA控制器已经等待了外设请求由PDMA_TOC1定义的时间周期，用户可以写1清除这些位。 0 = 没有请求超时. 1 = 外设请求超时.
[8]	<b>REQTOF0</b>	<b>通道0请求超时标志</b> 该标志表示PDMA控制器已经等待了外设请求由PDMA_TOC0定义的时间周期，用户可以写1清除这些位。 0 = 没有请求超时. 1 = 外设请求超时.
[7:3]	<b>Reserved</b>	保留.
[2]	<b>ALIGNF</b>	<b>传输对齐中断标志(只读)</b> 0 = PDMA通道源地址和目的地址都遵循传输宽度的设定对齐 1 = PDMA通道源地址和目的地址没有遵循传输宽度的设定对齐
[1]	<b>TDIF</b>	<b>传输完成中断标志(只读)</b> 该位指示PDMA控制器已经完成了数据传输； 用户可以读取寄存器PDMA_TDSTS来获知哪个通道完成了数据传输。 0 = PDMA通道未完成了数据传输 1 = PDMA通道已经完成了数据传输
[0]	<b>ABTIF</b>	<b>PDMA 读/写目标终止中断标志(只读)</b> 该位指示PDMA的传输数据目标存在终止错误； 软件可以读寄存器PDMA_ABSTS来找出哪个通道存在目标终止错误。 0 = 未收到AHB总线错误响应

位	描述
	1 = 收到AHB总线错误响应

PDMA\_ABTSR PDMA通道读/写目标终止标志寄存器

寄存器	偏移地址	R/W	描述	复位值
PDMA_ABTSR	PDMA_BA + 0x420	R/W	PDMA通道读/写目标终止标志寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
ABTIF15	ABTIF14	ABTIF13	ABTIF12	ABTIF11	ABTIF10	ABTIF9	ABTIF8
7	6	5	4	3	2	1	0
ABTIF7	ABTIF6	ABTIF5	ABTIF4	ABTIF3	ABTIF2	ABTIF1	ABTIF0

位	描述	
[31:16]	Reserved	保留.
[n] n=0,1..15	ABTIFn	<b>PDMA 读/写目标终止中断状态标志</b> 该位指示哪个PDMA控制器存在目标终止错误；用户可以写1清除这些位。 0 = 当通道n传输数据时，未收到AHB总线错误响应。 1 = 当通道n传输数据时，收到AHB总线错误响应。

注：如果目标中止，REQSRC 应设置为 0 以禁用外设请求。

**PDMA\_TDSTS PDMA通道传输数据完成标志寄存器**

寄存器	偏移地址	R/W	描述	复位值
PDMA_TDSTS	PDMA_BA + 0x424	R/W	PDMA通道传输完成标志寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
TDIF15	TDIF14	TDIF13	TDIF12	TDIF11	TDIF10	TDIF9	TDIF8
7	6	5	4	3	2	1	0
TDIF7	TDIF6	TDIF5	TDIF4	TDIF3	TDIF2	TDIF1	TDIF0

位	描述	
[31:16]	Reserved	保留.
[n] n=0,1..15	TDIFn	传输完成标志寄存器 该位指示PDMA控制器通道传输数据是否完成，用户可以写1清除这些位。 0 = PDMA 通道传输数据未完成。 1 = PDMA 通道传输数据完成。

PDMA\_ALIGN PDMA 传输对齐状态寄存器

寄存器	偏移地址	R/W	描述	复位值
PDMA_ALIGN	PDMA_BA + 0x428	R/W	PDMA 传输对齐状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
ALIGN15	ALIGN14	ALIGN13	ALIGN12	ALIGN11	ALIGN10	ALIGN9	ALIGN8
7	6	5	4	3	2	1	0
ALIGN7	ALIGN6	ALIGN5	ALIGN4	ALIGN3	ALIGN2	ALIGN1	ALIGN0

位	描述	
[31:16]	Reserved	保留.
[n] n=0,1..15	ALIGNn	<p>传输对齐标志寄存器</p> <p>0 = PDMA 通道源地址和目的地址都遵循传输宽度的设定对齐</p> <p>1 = PDMA 通道源地址和目的地址没有遵循传输宽度的设定对齐</p> <p>注:源地址和目标地址必须对齐。</p>

PDMA\_TACTSTS PDMA传输激活标志寄存器

寄存器	偏移地址	R/W	描述	复位值
PDMA_TACTSTS	PDMA_BA + 0x42C	R	PDMA 传输激活标志寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
TXACTF15	TXACTF14	TXACTF13	TXACTF12	TXACTF11	TXACTF10	TXACTF9	TXACTF8
7	6	5	4	3	2	1	0
TXACTF7	TXACTF6	TXACTF5	TXACTF4	TXACTF3	TXACTF2	TXACTF1	TXACTF0

位	描述	
[31:16]	Reserved	保留.
[n] n=0,1..15	TXACTFn	<p>传输激活标志寄存器(只读)</p> <p>该位用于指示哪个PDMA通道正在传输</p> <p>0 = PDMA通道没有在传输数据</p> <p>1 = PDMA正在传输数据</p>

PDMA\_TOUTPSC PDMA 超时预分频器寄存器

寄存器	偏移地址	R/W	描述	复位值
PDMA_TOUTPSC	PDMA_BA + 0x430	R/W	PDMA 超时预分频器寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved	TOUTPSC1			Reserved	TOUTPSC0		

位	描述	
[31:7]	Reserved	保留.
[6:4]	TOUTPSC1	<b>PDMA 通道 1 超时时钟源预分频位</b> 000 = PDMA 通道 1 超时时钟源是 HCLK/2 <sup>8</sup> . 001 = PDMA 通道 1 超时时钟源是 HCLK/2 <sup>9</sup> . 010 = PDMA 通道 1 超时时钟源是 HCLK/2 <sup>10</sup> . 011 = PDMA 通道 1 超时时钟源是 HCLK/2 <sup>11</sup> . 100 = PDMA 通道 1 超时时钟源是 HCLK/2 <sup>12</sup> . 101 = PDMA 通道 1 超时时钟源是 HCLK/2 <sup>13</sup> . 110 = PDMA 通道 1 超时时钟源是 HCLK/2 <sup>14</sup> . 111 = PDMA 通道 1 超时时钟源是 HCLK/2 <sup>15</sup> .
[3]	Reserved	保留.
[2:0]	TOUTPSC0	<b>PDMA 通道 0超时时钟源预分频位</b> 000 = PDMA 通道 0 超时时钟源是 HCLK/2 <sup>8</sup> . 001 = PDMA 通道 0 超时时钟源是 HCLK/2 <sup>9</sup> . 010 = PDMA 通道 0 超时时钟源是 HCLK/2 <sup>10</sup> . 011 = PDMA 通道 0 超时时钟源是 HCLK/2 <sup>11</sup> . 100 = PDMA 通道 0 超时时钟源是 HCLK/2 <sup>12</sup> . 101 = PDMA 通道 0 超时时钟源是 HCLK/2 <sup>13</sup> . 110 = PDMA 通道 0 超时时钟源是 HCLK/2 <sup>14</sup> . 111 = PDMA 通道 0 超时时钟源是 HCLK/2 <sup>15</sup> .

**PDMA\_TOUTEN PDMA 超时使能寄存器**

寄存器	偏移地址	R/W	描述	复位值
PDMA_TOUTEN	PDMA_BA + 0x434	R/W	PDMA 超时使能寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved						TOUTEN1	TOUTEN0

位	描述	
[31:2]	Reserved	保留.
[n] n=0,1	TOUTENn	<b>PDMA 超时使能位</b> 0 = PDMA 通道 n 超时功能禁止. 1 = PDMA 通道 n 超时功能使能

PDMA\_TOUTIEN PDMA 超时中断使能寄存器

寄存器	偏移地址	R/W	描述				复位值
PDMA_TOUTIEN	PDMA_BA + 0x438	R/W	PDMA 超时中断使能寄存器				0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved						TOUTIEN1	TOUTIENO

位	描述	
[31:2]	Reserved	保留.
[n] n=0,1	TOUTIENn	PDMA 超时中断使能位 0 = PDMA 通道 n 超时中断禁止. 1 = PDMA 通道 n 超时中断使能

PDMA\_SCATBA PDMA Scatter-gather描述符表基地址寄存器

寄存器	偏移地址	R/W	描述				复位值
PDMA_SCATBA	PDMA_BA + 0x43C	R/W	PDMA Scatter-Gather描述符表基地址寄存器				0x2000_0000

31	30	29	28	27	26	25	24
SCATBA							
23	22	21	20	19	18	17	16
SCATBA							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved							

位	描述	
[31:16]	<b>SCATBA</b>	<p><b>PDMA Scatter-Gather 描述符表基地址寄存器</b></p> <p>在Scatter-Gather模式，该寄存器是基地址用于计算下一个链接地址。</p> <p>下一个链接地址是：</p> <p>下一个链接地址= PDMA_SCATBA + PDMA_DSCT_FIRST.</p> <p>注意：仅在Scatter-Gather模式有效</p>
[15:0]	<b>Reserved</b>	保留.

PDMA\_TOC0\_1 PDMA 超时周期计数器寄存器

寄存器	偏移地址	R/W	描述	复位值
PDMA_TOC0_1	PDMA_BA + 0x440	R/W	PDMA 超时计数器通道1和通道0寄存器	0xFFFF_FFFF

31	30	29	28	27	26	25	24
TOC1							
23	22	21	20	19	18	17	16
TOC1							
15	14	13	12	11	10	9	8
TOC0							
7	6	5	4	3	2	1	0
TOC0							

位	描述	
[31:16]	TOC1	通道 1超时计数器 该域控制通道1超时周期时间，计数单位是基于10 kHz时钟。
[15:0]	TOC0	通道 0超时计数器 该域控制通道0超时周期时间，计数单位是基于10 kHz时钟。

PDMA\_CHRST PDMA 通道复位寄存器

寄存器	偏移地址	R/W	描述	复位值
PDMA_CHRST	PDMA_BA + 0x460	R/W	PDMA 通道复位寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
CH15RST	CH14RST	CH13RST	CH12RST	CH11RST	CH10RST	CH9RST	CH8RST
7	6	5	4	3	2	1	0
CH7RST	CH6RST	CH5RST	CH4RST	CH3RST	CH2RST	CH1RST	CH0RST

位	描述	
[31:16]	Reserved	保留.
[15:0]	CHnRST	<b>通道 N 复位</b> 0 = 对应通道n不复位 1 = 对应通道n复位

PDMA\_REQSEL0\_3 PDMA请求源选择寄存器0

寄存器	偏移地址	R/W	描述	复位值
PDMA_REQSEL0_3	PDMA_BA + 0x480	R/W	PDMA请求源选择寄存器0	0x0000_0000

31	30	29	28	27	26	25	24
Reserved	REQSRC3						
23	22	21	20	19	18	17	16
Reserved	REQSRC2						
15	14	13	12	11	10	9	8
Reserved	REQSRC1						
7	6	5	4	3	2	1	0
Reserved	REQSRC0						

位	描述	
[31]	Reserved	保留.
[30:24]	REQSRC3	<b>通道3请求源选择</b> 该域定义哪个外设连接到PDMA通道3。用户可以通过设定寄存器REQSRC3配置外设。 <b>注意:</b> 通道配置与寄存器REQSRC0相同。请参考REQSRC0说明。
[23]	Reserved	保留.
[22:16]	REQSRC2	<b>通道2请求源选择</b> 该域定义哪个外设连接到PDMA通道2。用户可以通过设定寄存器REQSRC2配置外设。 <b>注意:</b> 通道配置与寄存器REQSRC0相同。请参考REQSRC0说明。
[15]	Reserved	保留.
[14:8]	REQSRC1	<b>通道1请求源选择</b> 该域定义哪个外设连接到PDMA通道1。用户可以通过设定寄存器REQSRC1配置外设。 <b>注意:</b> 通道配置与寄存器REQSRC0相同。请参考REQSRC0说明。
[7]	Reserved	保留.
[6:0]	REQSRC0	<b>通道0请求源选择</b> 该域定义哪个外设连接到PDMA通道0。用户可以通过设定寄存器REQSRC0配置外设。 0 = 禁止 PDMA 外设请求。 1 = 保留。 2 = 通道连接到 USB_TX. 3 = 通道连接到USB_RX. 4 = 通道连接到UART0_TX. 5 = 通道连接到UART0_RX. 6 = 通道连接到UART1_TX. 7 = 通道连接到UART1_RX.

位	描述
	8 =通道连接到UART2_TX. 9 =通道连接到UART2_RX. 10=通道连接到UART3_TX. 11 =通道连接到UART3_RX. 12 =通道连接到UART4_TX. 13 =通道连接到UART4_RX. 14 =通道连接到UART5_TX. 15 =通道连接到UART5_RX. 16 =通道连接到USCI0_TX. 17 =通道连接到USCI0_RX. 18 =通道连接到USCI1_TX. 19 =通道连接到USCI1_RX. 20 =通道连接到QSPI0_TX. 21 =通道连接到QSPI0_RX. 22 =通道连接到SPI0_TX. 23 =通道连接到SPI0_RX. 24 =通道连接到SPI1_TX. 25 =通道连接到SPI1_RX. 26 =通道连接到SPI2_TX. 27 =通道连接到SPI2_RX. 28 =通道连接到SPI3_TX. 29 =通道连接到SPI3_RX. 30 = 保留. 31 = 保留. 32 =通道连接到PWM0_P1_RX. 33 =通道连接到PWM0_P2_RX. 34 =通道连接到PWM0_P3_RX. 35 =通道连接到PWM1_P1_RX. 36 =通道连接到PWM1_P2_RX. 37 =通道连接到PWM1_P3_RX. 38 =通道连接到I2C0_TX. 39 =通道连接到I2C0_RX. 40 =通道连接到I2C1_TX. 41 =通道连接到I2C1_RX. 42 =通道连接到I2C2_TX. 43 =通道连接到I2C2_RX. 44 =通道连接到I2S0_TX. 45 =通道连接到I2S0_RX. 46 =通道连接到TMR0. 47 =通道连接到TMR1. 48 =通道连接到TMR2. 49 =通道连接到TMR3. 50 =通道连接到ADC_RX. 51 =通道连接到DAC0_TX.

位	描述
	<p>52 =通道连接到DAC1_TX. 53 =通道连接到PWM0_CH0_TX. 54 =通道连接到PWM0_CH1_TX. 55 =通道连接到PWM0_CH2_TX. 56 =通道连接到PWM0_CH3_TX. 57 =通道连接到PWM0_CH4_TX. 58 =通道连接到PWM0_CH5_TX. 59 =通道连接到PWM1_CH0_TX. 60 =通道连接到PWM1_CH1_TX. 61 =通道连接到PWM1_CH2_TX. 62 =通道连接到PWM1_CH3_TX. 63 =通道连接到PWM1_CH4_TX. 64 =通道连接到PWM1_CH5_TX. 65 =通道连接到ETMC_RX. 其他 = 保留.</p> <p><b>注意1:</b> 一个外设不可以同时连接到两个通道。</p> <p><b>注意2:</b> 当内存与内存之间数据传输时，该寄存器无效。</p>

## PDMA\_REQSEL4\_7 PDMA请求源选择寄存器1

寄存器	偏移地址	R/W	描述	复位值
PDMA_REQSEL4_7	PDMA_BA + 0x484	R/W	PDMA请求源选择寄存器1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved	REQSRC7						
23	22	21	20	19	18	17	16
Reserved	REQSRC6						
15	14	13	12	11	10	9	8
Reserved	REQSRC5						
7	6	5	4	3	2	1	0
Reserved	REQSRC4						

位	描述	
[31]	Reserved	保留.
[30:24]	REQSRC7	<b>通道7请求源选择</b> 该域定义哪个外设连接到PDMA通道7。用户可以通过设定寄存器REQSRC7配置外设。 <b>注意:</b> 通道配置与寄存器REQSRC0相同。请参考REQSRC0说明。
[23]	Reserved	保留.
[22:16]	REQSRC6	<b>通道6请求源选择</b> 该域定义哪个外设连接到PDMA通道6。用户可以通过设定寄存器REQSRC6配置外设。 <b>注意:</b> 通道配置与寄存器REQSRC0相同。请参考REQSRC0说明。
[15]	Reserved	保留.
[14:8]	REQSRC5	<b>通道5请求源选择</b> 该域定义哪个外设连接到PDMA通道5。用户可以通过设定寄存器REQSRC5配置外设。 <b>注意:</b> 通道配置与寄存器REQSRC0相同。请参考REQSRC0说明。
[7]	Reserved	保留.
[6:0]	REQSRC4	<b>通道4请求源选择</b> 该域定义哪个外设连接到PDMA通道4。用户可以通过设定寄存器REQSRC4配置外设。 <b>注意:</b> 通道配置与寄存器REQSRC0相同。请参考REQSRC0说明。

PDMA\_REQSEL8\_11 PDMA 请求源选择寄存器2

寄存器	偏移地址	R/W	描述	复位值
PDMA_REQSEL8_11	PDMA_BA + 0x488	R/W	PDMA 请求源选择寄存器2	0x0000_0000

31	30	29	28	27	26	25	24
Reserved	REQSRC11						
23	22	21	20	19	18	17	16
Reserved	REQSRC10						
15	14	13	12	11	10	9	8
Reserved	REQSRC9						
7	6	5	4	3	2	1	0
Reserved	REQSRC8						

位	描述	
[31]	Reserved	保留.
[30:24]	REQSRC11	<b>通道11请求源选择</b> 该域定义哪个外设连接到PDMA通道11.. 用户可以通过设定寄存器REQSRC11配置外设。 <b>注意:</b> 通道配置与寄存器REQSRC0相同。请参考REQSRC0说明。
[23]	Reserved	保留.
[22:16]	REQSRC10	<b>通道10请求源选择</b> 该域定义哪个外设连接到PDMA通道10.. 用户可以通过设定寄存器REQSRC10配置外设。 <b>注意:</b> 通道配置与寄存器REQSRC0相同。请参考REQSRC0说明。
[15]	Reserved	保留.
[14:8]	REQSRC9	<b>通道9请求源选择</b> 该域定义哪个外设连接到PDMA通道9.. 用户可以通过设定寄存器REQSRC9配置外设。 <b>注意:</b> 通道配置与寄存器REQSRC0相同。请参考REQSRC0说明。
[7]	Reserved	保留.
[6:0]	REQSRC8	<b>通道8请求源选择</b> 该域定义哪个外设连接到PDMA通道8.. 用户可以通过设定寄存器REQSRC8配置外设。 <b>注意:</b> 通道配置与寄存器REQSRC0相同。请参考REQSRC0说明。

## PDMA\_REQSEL12\_15 PDMA 请求源选择寄存器3

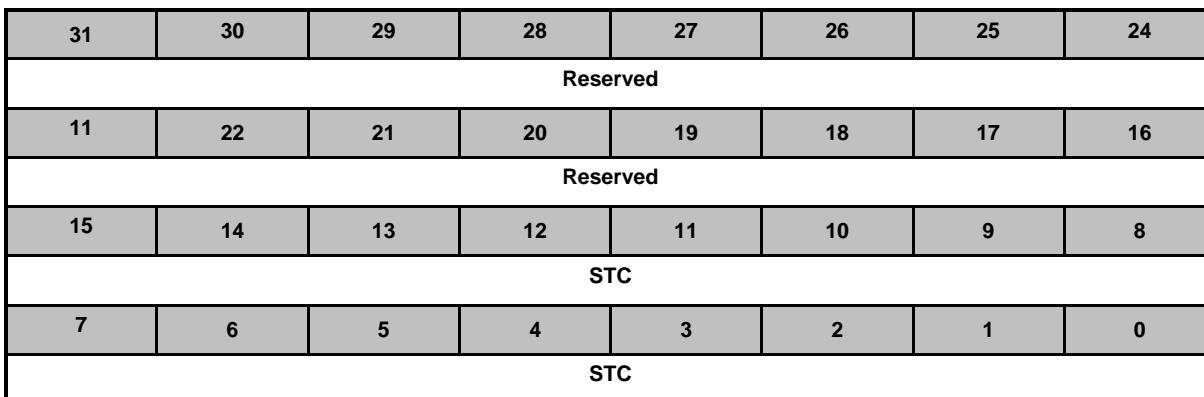
寄存器	偏移地址	R/W	描述	复位值
PDMA_REQSEL12_15	PDMA_BA + 0x48C	R/W	PDMA 请求源选择寄存器3	0x0000_0000

31	30	29	28	27	26	25	24
Reserved	REQSRC15						
23	22	21	20	19	18	17	16
Reserved	REQSRC14						
15	14	13	12	11	10	9	8
Reserved	REQSRC13						
7	6	5	4	3	2	1	0
Reserved	REQSRC12						

位	描述	
[31]	Reserved	保留.
[30:24]	REQSRC15	<b>通道15请求源选择</b> 该域定义哪个外设连接到PDMA通道15。用户可以通过设定寄存器REQSRC15配置外设。 <b>注意:</b> 通道配置与寄存器REQSRC0相同。请参考REQSRC0说明。
[23]	Reserved	保留.
[22:16]	REQSRC14	<b>通道14请求源选择</b> 该域定义哪个外设连接到PDMA通道14。用户可以通过设定寄存器REQSRC14配置外设。 <b>注意:</b> 通道配置与寄存器REQSRC0相同。请参考REQSRC0说明。
[15]	Reserved	保留.
[14:8]	REQSRC13	<b>通道13请求源选择</b> 该域定义哪个外设连接到PDMA通道13。用户可以通过设定寄存器REQSRC13配置外设。 <b>注意:</b> 通道配置与寄存器REQSRC0相同。请参考REQSRC0说明。
[7]	Reserved	保留.
[6:0]	REQSRC12	<b>通道12请求源选择</b> 该域定义哪个外设连接到PDMA通道12。用户可以通过设定寄存器REQSRC12配置外设。 <b>注意:</b> 通道配置与寄存器REQSRC0相同。请参考REQSRC0说明。

**PDMA\_STCRn PDMA 跨步传输计数寄存器n**

寄存器	偏移地址	R/W	描述	复位值
PDMA_STCR0	PDMA_BA + 0x500	R/W	PDMA 通道 0的跨步传输计数寄存器	0x0000_0000
PDMA_STCR1	PDMA_BA + 0x508	R/W	PDMA 通道 1的跨步传输计数寄存器	0x0000_0000
PDMA_STCR2	PDMA_BA + 0x510	R/W	PDMA 通道 2的跨步传输计数寄存器	0x0000_0000
PDMA_STCR3	PDMA_BA + 0x518	R/W	PDMA 通道 3的跨步传输计数寄存器	0x0000_0000
PDMA_STCR4	PDMA_BA + 0x520	R/W	PDMA 通道 4的跨步传输计数寄存器	0x0000_0000
PDMA_STCR5	PDMA_BA + 0x528	R/W	PDMA 通道 5的跨步传输计数寄存器	0x0000_0000



位	描述	
[31:16]	Reserved	保留.
[15:0]	STC	<b>PDMA 跨步传输计数</b> 这 16-bit 寄存器定义每一行跨步传输计数值

PDMA\_ASOCRn PDMA 地址跨步偏移量控制寄存器n

寄存器	偏移地址	R/W	描述	复位值
PDMA_ASOCR0	PDMA_BA + 0x504	R/W	PDMA 通道 0 的地址跨步偏移量寄存器	0x0000_0000
PDMA_ASOCR1	PDMA_BA + 0x50C	R/W	PDMA 通道 1 的地址跨步偏移量寄存器	0x0000_0000
PDMA_ASOCR2	PDMA_BA + 0x514	R/W	PDMA 通道 2 的地址跨步偏移量寄存器	0x0000_0000
PDMA_ASOCR3	PDMA_BA + 0x51C	R/W	PDMA 通道 3 的地址跨步偏移量寄存器	0x0000_0000
PDMA_ASOCR4	PDMA_BA + 0x524	R/W	PDMA 通道 4 的地址跨步偏移量寄存器	0x0000_0000
PDMA_ASOCR5	PDMA_BA + 0x52C	R/W	PDMA 通道 5 的地址跨步偏移量寄存器	0x0000_0000

31	30	29	28	27	26	25	24
DASOL							
11	22	21	20	19	18	17	16
DASOL							
15	14	13	12	11	10	9	8
SASOL							
7	6	5	4	3	2	1	0
SASOL							

位	描述	
[31:16]	DASOL	<b>VDMA 目的地址跨步偏移长度</b> 这 16-bit 寄存器定义目的地址每一行跨步传输偏移计数值
[15:0]	SASOL	<b>VDMA 源地址跨步偏移长度</b> 这 16-bit 寄存器定义源地址每一行跨步传输偏移计数值

## 6.7 TMR 定时器控制器

### 6.7.1 概述

定时器控制器包含 4 组 32-位定时器，TIMER0~TIMER3，提供用户便捷的计数定时功能。定时器可执行很多功能，如频率测量，时间延迟，时钟发生，外部输入管脚事件计数和外部捕获管脚脉宽测量等。

定时器计数器也提供四个PWM产生器，每个PWM产生器支持两个PWM输出通道，可以是独立模式或互补模式。PWM的输出状态可以通过管脚屏蔽控制，极性控制和刹车控制。支持死区发生器。

### 6.7.2 特性

#### 6.7.2.1 定时器功能特性

- 4 组 32-位定时器，带24位向上计数器和一个8位的预分频计数器
- 每个定时器都有独立的时钟源
- 提供 单次, 周期, 触发输出 和 连续计数 四种操作模式
- 通过CNT (TIMERx\_CNT[23:0])可读取内部 24 位向上计数器的值
- 支持事件计数功能
- 通过CAPDAT (TIMERx\_CAP[23:0])可读取24-bit 捕获值
- 支持外部管脚捕获，可用于脉宽测量
- 支持外部引脚捕获，可用于复位24位向上定时器
- 如果定时器中断信号产生，支持芯片从空闲/掉电模式唤醒
- 支持Timer0 ~ Timer3超时中断信号或捕获中断信号触发PWM, ADC, DAC 和 PDMA 功能
- 支持当ACMP输出信号跳变触发内部捕获
- 支持定时器间互触发模式
- 支持事件计数源来自内部USB SOF信号

#### 6.7.2.2 PWM 功能特性

- 支持最大时钟频率到PCLK
- 支持PWM独立模式带两个输出通道
- 支持PWM互补模式带成对的输出通道
  - 12-bit 死区事件插入，带 12-bit 预分频
- 支持 12-bit 预分频，从 1 到 4096
- 支持 16-bit PWM 计数器
  - 上数，下数和上下数计数操作模式
  - 单次或自动重载计数器操作模式
- 对每一个PWM输出管脚支持屏蔽功能和三态使能
- 支持刹车功能
  - 刹车源来自管脚，模拟比较器和系统安全事件（时钟失败、欠压侦测、SRAM校验错误和CPU锁死）

- 对刹车源，支持刹车管脚噪音滤波控制
- 边沿检测刹车源控制刹车状态直到刹车状态清除
- 电平侦测刹车源用于在刹车条件移除后自动恢复功能。
- 支持在下面事件中断：
  - PWM 0点、周期点、上数比较点或下数比较点事件
  - 刹车条件发生
- 支持在下面事件触发ADC：
  - PWM 0点、周期点、上数比较点或下数比较点事件

### 6.7.3 框图

定时器控制器和时钟控制框图如下图所示

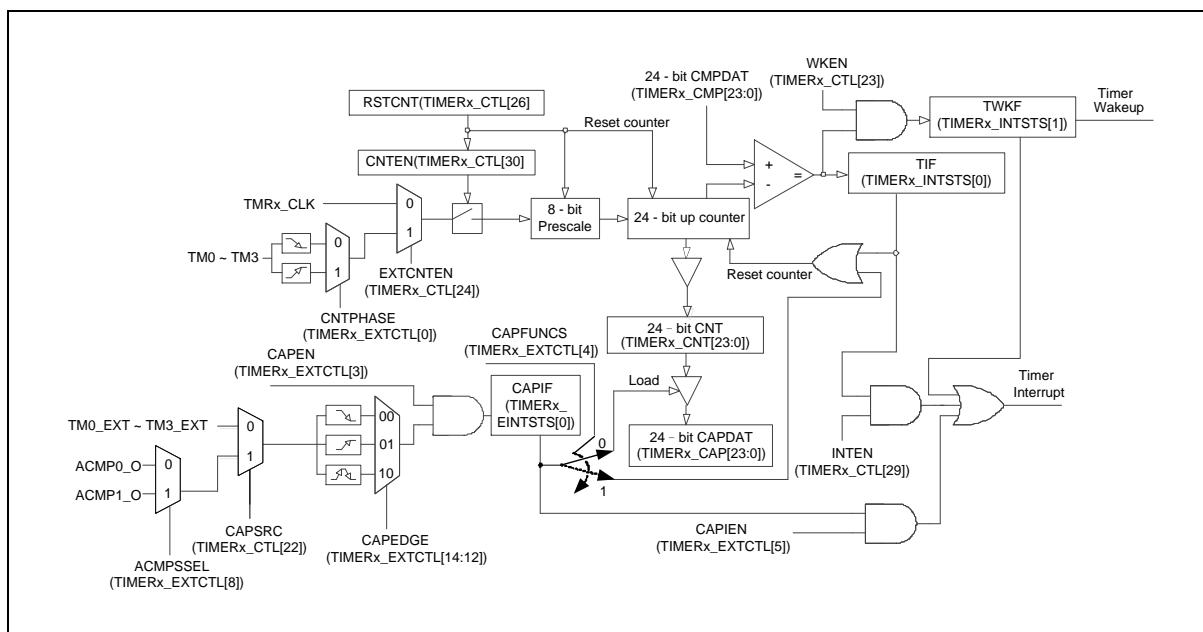


图 6.7-1 定时器控制器框图

设置 FUNMODE (TIMERx\_ALTCTL[0]) 为0使能定时器模式。在定时器模式，定时器0~定时器3的时钟源可以在寄存器TMRxCKEN (CLK\_APBCLK0[5:2])使能，在寄存器TMR0SEL (CLK\_CLKSEL1[10:8]) (定时器0)，TMR1SEL (CLK\_CLKSEL1[14:12]) (定时器1)，TMR2SEL (CLK\_CLKSEL1[18:16]) (定时器2)，TMR3SEL (CLK\_CLKSEL1[22:20]) (定时器3)选择不同时钟频率。如下图。

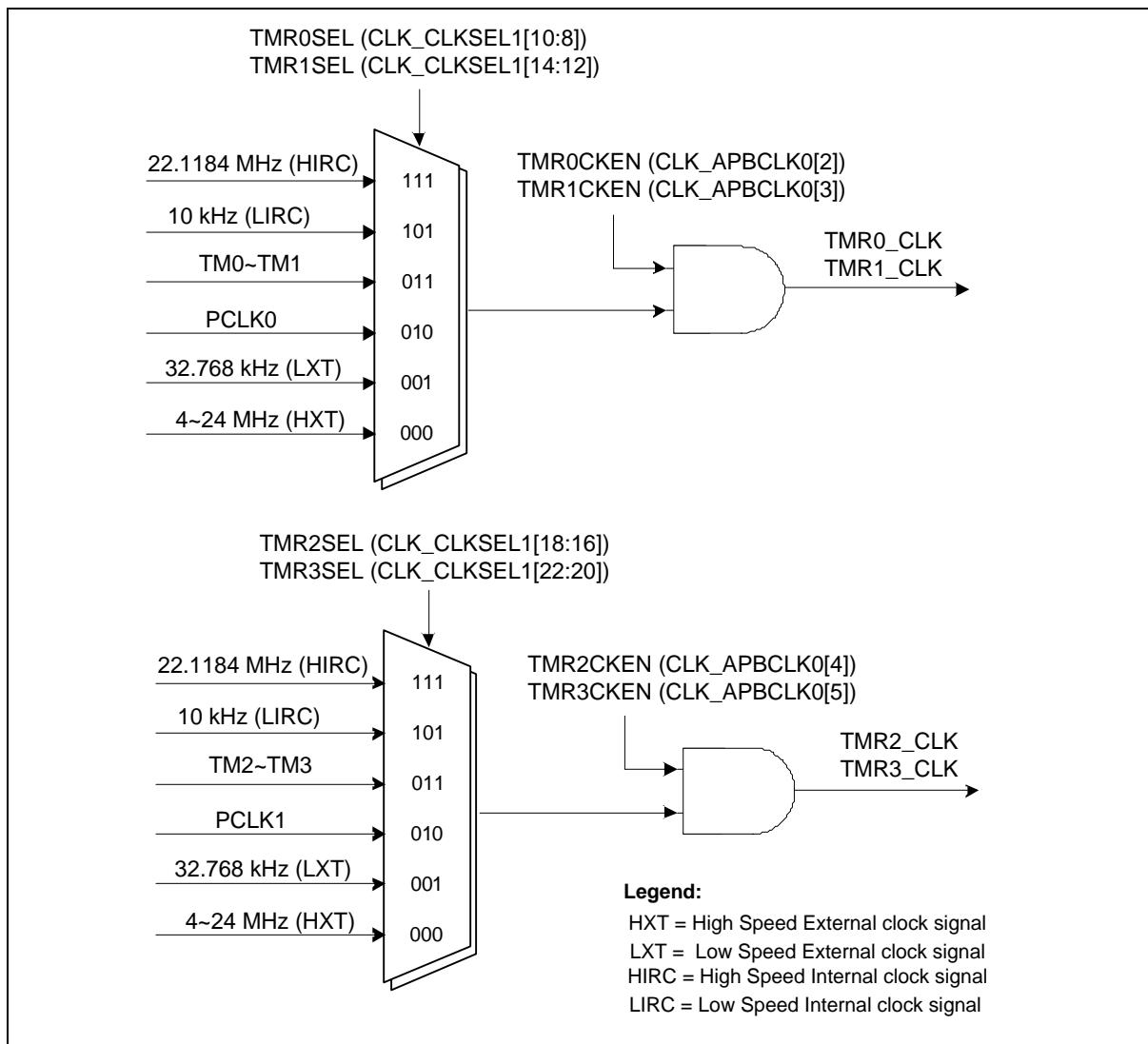


图 6.7-2 定时器控制器的时钟源

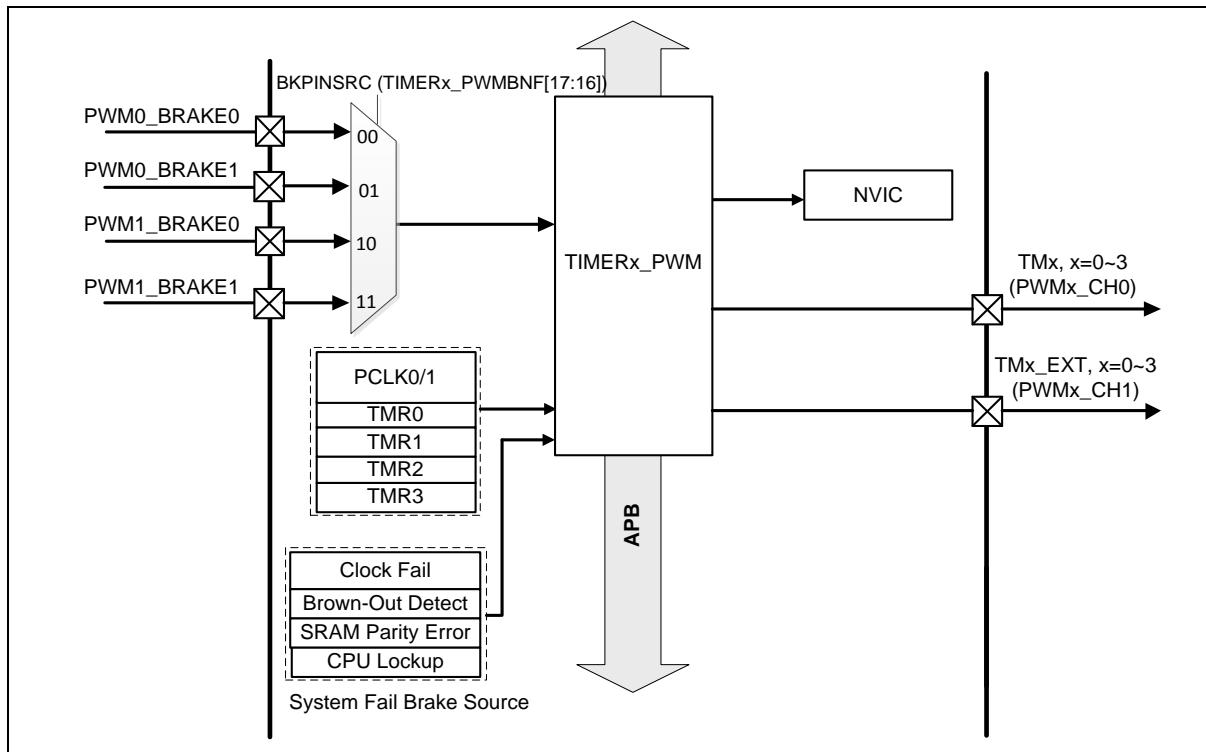


图 6.7-3 PWM 发生器框图

设置 FUNMODE (TIMERx\_ALTCTL[0]) 为 1 使能 PWM 模式。在 PWM 模式，定时器 0~定时器 3 的时钟源可以在寄存器 TMRxCKEN (CLK\_APBCLK0[5:2]) 使能，TMR0\_CLK 和 TMR1\_CLK 时钟源固定为 PCLK0，TMR2\_CLK 和 TMR3\_CLK 时钟源固定为 PCLK1。PWM 系统时钟频率是 PCLKx 的频率，如图 6.7-4。

PWM 计数器的时钟源 (TIMERx\_PWMCLK) 可以从 PWM 系统时钟 (TMRx\_CLK) 或定时器中断事件 (TMRx\_INT) 中选，如图 6.7-5..

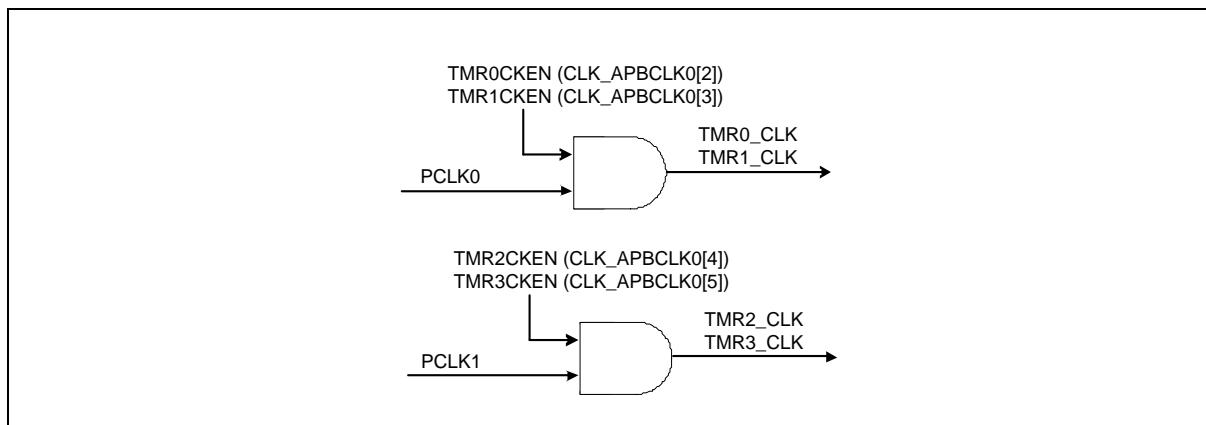


图 6.7-4 PWM 系统时钟源控制

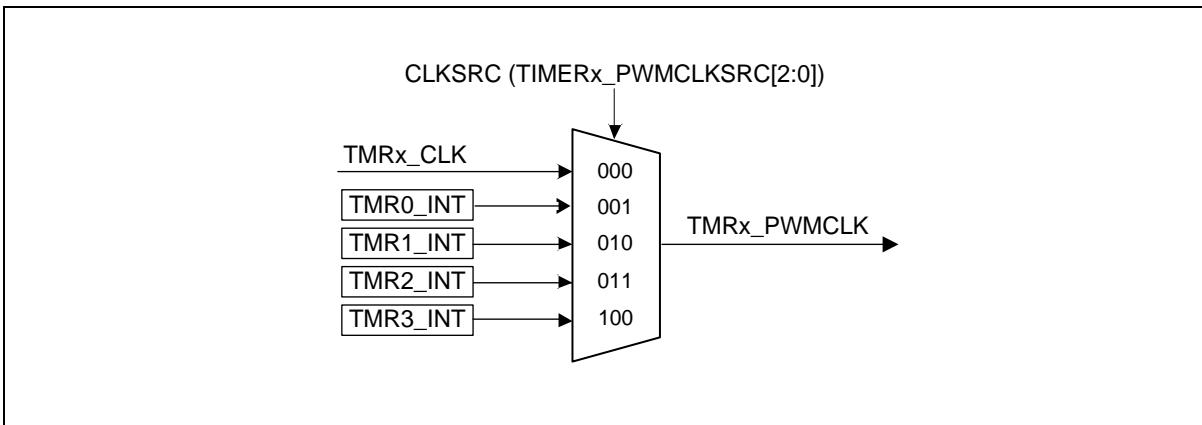


图 6.7-5 PWM 计数器时钟源控制

图 6.7-6 和图 6.7-7 表示 PWM 独立模式和互补模式的结构。每个 PWM 发生器，独立模式和互补模式都支持 PWM<sub>x</sub>\_CH0 和 PWM<sub>x</sub>\_CH1 输出通道。

当计数器数到 0，PERIOD (TIMER<sub>x</sub>\_PWMPERIOD[15:0]) 或等于 CMP (TIMER<sub>x</sub>\_PWMCMPDAT[15:0])，相关事件产生。这些事件传到对应的 PWM 发生器产生 PWM 脉冲，中断信号，和启动 ADC 转换的触发信号。输出控制模块用于决定 PWM 脉冲的输出。输出控制模块的刹车功能也产生中断事件。死区控制仅在 PWM 互补模式有效。

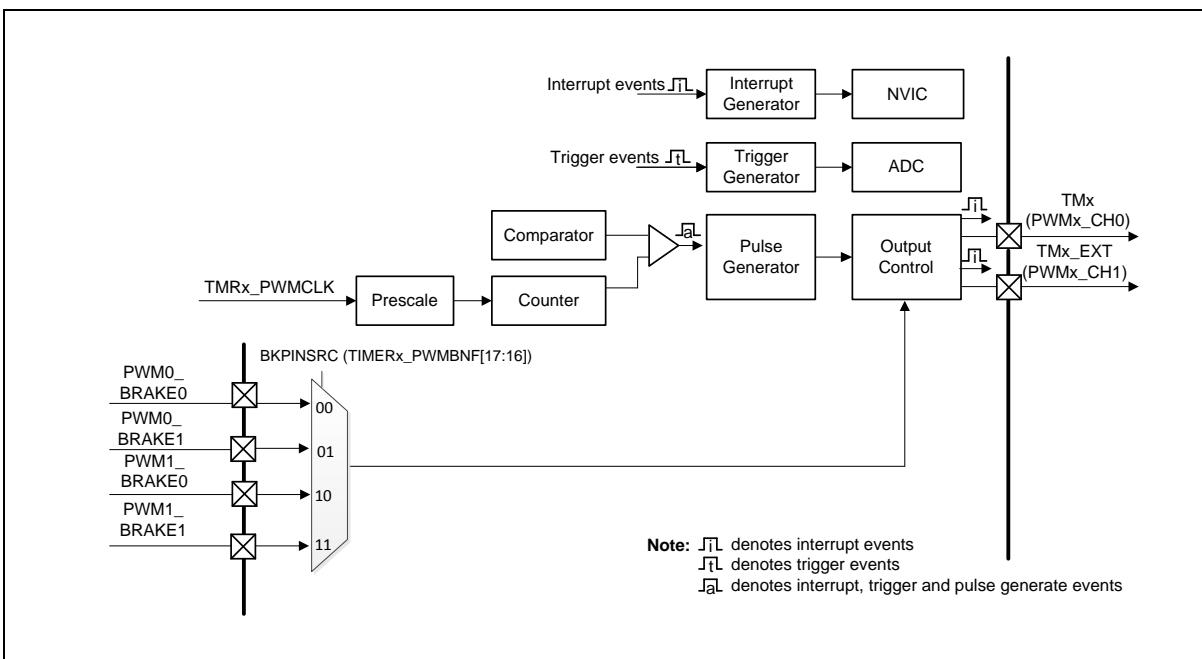


图 6.7-6 PWM 独立模式结构图

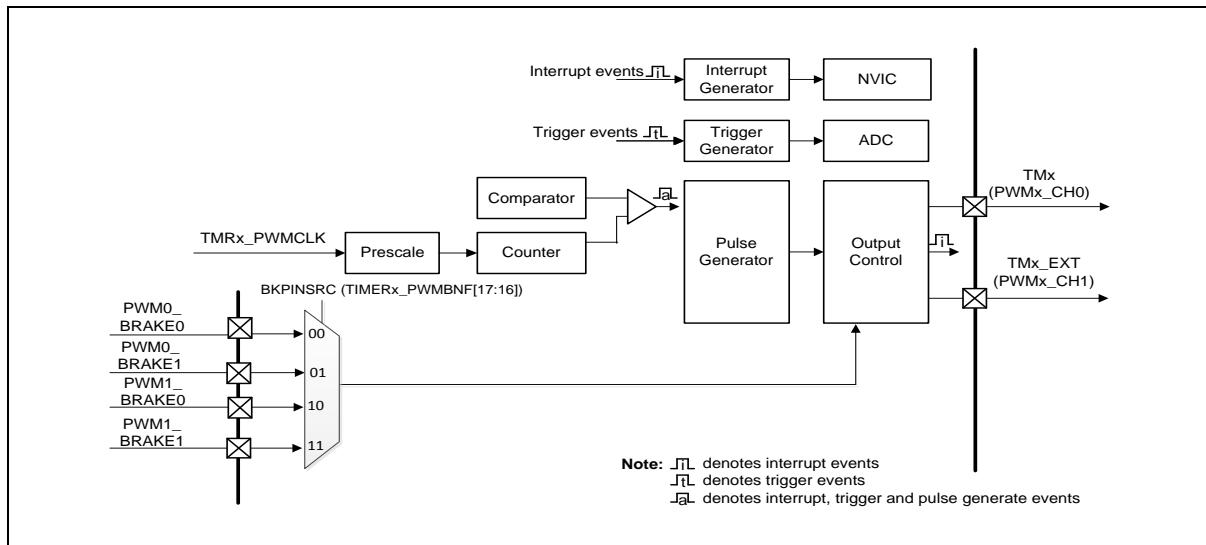


图 6.7-7 PWM 互补模式结构图

#### 6.7.4 基本配置

设置 FUNMODE (TIMERx\_ALTCTL[0]) 为 0 使能定时器模式。在定时器模式，定时器 0~定时器 3 的时钟源可以在寄存器 TMRxCKEN (CLK\_APBCLK0[5:2]) 使能，在寄存器 TMR0SEL (CLK\_CLKSEL1[10:8]) (定时器 0)，TMR1SEL (CLK\_CLKSEL1[14:12]) (定时器 1)，TMR2SEL (CLK\_CLKSEL1[18:16]) (定时器 2)，TMR3SEL (CLK\_CLKSEL1[22:20]) (定时器 3) 选择不同时钟频率。

设置 FUNMODE (TIMERx\_ALTCTL[0]) 为 1 使能 PWM 模式。在 PWM 模式，定时器 0~定时器 3 的时钟源可以在寄存器 TMRxCKEN (CLK\_APBCLK0[5:2]) 使能，TMR0\_CLK 和 TMR1\_CLK 时钟源固定为 PCLK0，TMR2\_CLK 和 TMR3\_CLK 时钟源固定为 PCLK1。

##### 6.7.4.1 TIMER01 基本配置

- 时钟源配置
  - 在 TMR0CKEN (CLK\_APBCLK0[2]) 中使能 TIMER0 外设时钟
  - 在 TMR1CKEN (CLK\_APBCLK0[3]) 中使能 TIMER1 外设时钟
- 复位配置
  - 在 TMR0RST (SYS\_IPRST2[2]) 中复位 TIMER0 控制器
  - 在 TMR1RST (SYS\_IPRST2[3]) 中复位 TIMER1 控制器
- 管脚配置

定时器组	管脚名称	GPIO	MFP
TM0	TM0	PG.2	MFP13
		PB.5, PC.7	MFP14
	TM0_EXT	PA.11, PB.15, PH.0	MFP13
TM1	TM1	PC.14, PG.3	MFP13
		PB.4, PC.6	MFP14
	TM1_EXT	PA.10, PB.14, PH.1	MFP13

表 6.7-1 TIMER01 引脚配置

#### 6.7.4.2 TIMER23 基本配置

- 时钟源配置
  - 在 TMR2CKEN (CLK\_APBCLK0[4]) 中使能 TIMER2
  - 在 TMR2CKEN (CLK\_APBCLK0[5]) 中使能 TIMER3
- 复位配置
  - 在 TMR2RST (SYS\_IPRST2[4]) 中复位 TIMER2
  - 在 TMR3RST (SYS\_IPRST2[5]) 中复位 TIMER3
- 管脚配置

定时器组	管脚名称	GPIO	MFP
TM2	TM2	PG.4	MFP13
		PA.7, PB.3, PD.0	MFP14
	TM2_EXT	PA.9, PB.13, PH.2	MFP13
TM3	TM3	PF.11	MFP13
		PA.6, PB.2	MFP14
	TM3_EXT	PA.8, PB.12, PH.3	MFP13

表 6.7-2 TIMER23 引脚配置

#### 6.7.5 定时器功能描述

##### 6.7.5.1 定时器中断标志

定时器控制器支持两个中断标志：一个是TIF(TIMERx\_INTSTS[0])标志，该标志在当定时器计数器值 CNT (TIMERx\_CNT[23:0])与定时器比较值 CMPDAT (TIMERx\_CMP[23:0])相匹配时置位，另一个是CAPIF (TIMERx\_EINTSTS[0]) 标志，该标志在当 TMx \_EXT 管脚的变化与 CAPEDGE (TIMERx\_EXTCTL[14:12])的设置一致时置位。TWKF (TIMERx\_INTSTS[1])位表示中断唤醒标志。

##### 6.7.5.2 定时器计数模式

定时器控制器提供四种定时器计数模式：单次，周期，触发输出和连续计数模式

##### 6.7.5.3 单次模式

如果定时器工作在单周期 (one-shot) 模式(TIMERx\_CTL[28:27]为00)，且CNTEN (TIMERx\_CTL[30])置1，则定时器的计数器开始计数。一旦 CNT (TIMERx\_CNT[23:0]) 计数器的值达到 CMPDAT (TIMERx\_CMP[23:0])的值时，TIF (TIMERx\_INTSTS[0])标志将变为1，CNT的值和 CNTEN位将由定时器控制器自动清零，然后定时器计数操作停止。与此同时，如果INTEN (TIMERx\_CTL[29])位使能，则定时器中断信号产生并送到 NVIC通知CPU。

通过RSTACT (TIMERx\_CNT[31])，用户可以监视计数器复位操作是否生效。设置ICEDEBUG (TIMERx\_CTL[31]) 为1禁止 ICE 调试模式对定时器计数的影响。

##### 6.7.5.4 周期模式

如果定时器工作在周期 (periodic) 模式(TIMERx\_CTL[28:27]为01)且CNTEN (TIMERx\_CTL[30])置1，则定时器的计数器开始向上计数。一旦 CNT (TIMERx\_CNT[23:0]) 计数器的值达到 CMPDAT (TIMERx\_CMP[23:0])的值时，TIF (TIMERx\_INTSTS[0])标志将变为1，CNT的值将由定时器控制器自

动清零，然后定时器重新计数。与此同时，如果INTEN (TIMERx\_CTL[29])使能，则定时器中断信号产生并送到 NVIC 通知 CPU。在该模式，定时器控制器周期性地操作计数和与CMPDAT的值比较，直到CNTEN位由软件清0。

可以通过设置PERIOSEL (TIMERx\_CTL[20])位选择在周期模式下的定时器操作。

#### 6.7.5.5 触发输出模式

如果定时器工作在触发输出模式(TIMERx\_CTL[28:27]为10)且CNTEN (TIMERx\_CTL[30])位置1，则定时器的计数器开始计数。触发输出模式的计数操作大部分与周期模式是一样的，除了该模式当TIF (TIMERx\_INTSTS[0])位设置时，有相关的TM0 ~ TM3管脚会输出信号，因此，管脚TM0 ~ TM3上的触发输出信号以 50% 的占空比周期反复改变。

#### 6.7.5.6 连续计数模式

如果定时器工作在连续计数 (continuous counting) 模式 (TIMERx\_CTL[28:27] 为 11) 且 CNTEN (TIMERx\_CTL[30]) 位置1，则定时器的计数器开始计数。一旦 CNT (TIMERx\_CNT[23:0]) 的值达到 CMPDAT (TIMERx\_CMP[23:0]) 的值时，TIF (TIMERx\_INTSTS[0]) 标志将变为1，但 CNT 的值继续保持向上计数。与此同时，如果INTEN (TIMERx\_CTL[29])使能，则定时器中断信号产生并送到 NVIC 通知 CPU。在该模式，用户可以立刻改变不同的CMPDAT值，而不需要停止定时器计数和重新开始定时器计数。

例如，CMPDAT的值设置为 80。当CNT 达到 80时,TIF标志将被置1，定时器计数器继续计数，而且CNT的值将不回到0，而是继续计数，81, 82, 83, … 到  $(2^{24}-1)$ ，然后再一次 0, 1, 2, 3, … 到  $2^{24}-1$ ，如此往复。接下来，如果软件改变CMPDAT的值为200并且清除TIF标志位，当CNT的值达到200时，TIF标志将再次变为1。最后，软件改变CMPDAT的值为500并且清除TIF标志，当CNT的值达到500时，TIF标志将再次变为1。

在该模式，计数器计数是连续的。所以该操作模式叫做连续计数模式。

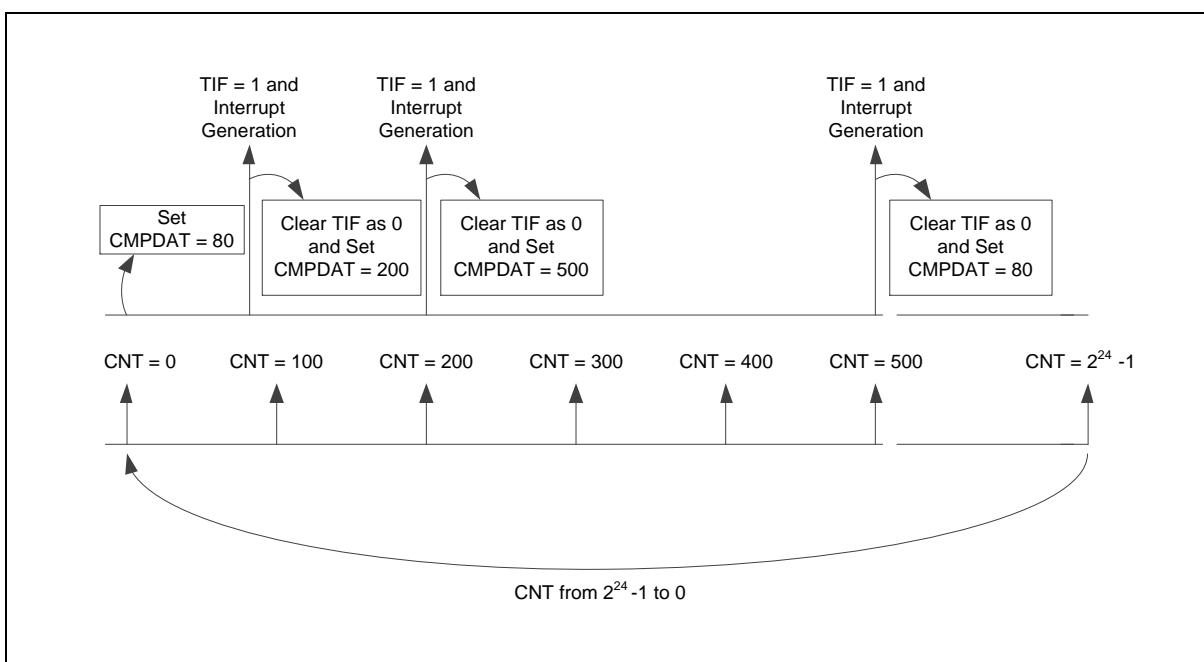


图 6.7-8 连续计数模式

#### 6.7.5.7 事件计数模式

定时器控制器也提供这样的应用，能对输入事件(来自管脚TM<sub>x</sub> ( $x=0\sim3$ ))计数并将事件的次数反映到 CNT (TIMERx\_CNT[23:0]) 的值。也可以称为事件计数功能。该功能下，EXTCNTEN

(**TIMERx\_CTL[24]**)位需置位并且定时器外设时钟源必须设为PCLK。

如果**ECNTSSEL** (**TIMERx\_EXTCTL[16]**) 是 0，事件计数源来自**TMx**管脚。软件可以通过**CNTDBEN** (**TIMERx\_EXTCTL[7]**)位来使能或关闭**TMx**管脚消抖电路。如果**TMx**管脚的消抖电路关闭，输入事件频率必须少于1/3 PCLKxHCLK，如果消抖电路打开，输入事件的频率须小于1/8 PCLKxHCLK，以保证**CNT**的值是正确的。软件也可以通过设置**CNTPHASE** (**TIMERx\_EXTCTL[0]**)来选择**TMx**管脚边沿检测的相位。

事件计数模式下，定时器计数操作模式可以设置为单次，周期，和连续计数模式来计算来自**TMx**管脚的输入事件**CNT** (**TIMERx\_CNT[23:0]**)的值。

如果**ECNTSSEL** (**TIMERx\_EXTCTL[16]**) 是 1，事件计数源由USB设备侦测帧起始包(SOF)产生。请参考USB设备规格。

#### 6.7.5.8 外部捕获模式

事件捕获功能是当检测到**TMx \_EXT**管脚( $x=0\sim3$ )边沿电平有变化时，**CNT** (**TIMERx\_CNT[23:0]**)值会加载到**CAPDAT** (**TIMERx\_CAP[23:0]**)。在该模式下，需把**CAPFUNCS** (**TIMERx\_EXTCTL[4]**)位设置为0，用来选择**TMx \_EXT**变化时用作事件捕获功能，而且定时器外设时钟源必须设为PCLK。

如果**CAPSRC** (**TIMERx\_CTL[22]**) 是 0，捕获事件由**TMx \_EXT**管脚跳变触发。软件可以通过**CAPDBEN** (**TIMERx\_EXTCTL[6]**)位来使能或关闭**TMx \_EXT**管脚消抖电路。在**TMx \_EXT**的消抖电路关闭时，**TMx \_EXT**管脚的转变频率必须少于1/3 PCLKxHCLK，在**TMx \_EXT**的消抖电路打开时，**TMx \_EXT**管脚的转变频率必须少于1/8 PCLKxHCLK，以保证捕获功能能够正常工作。软件也可以通过设置**CAPEDGE** (**TIMERx\_EXTCTL[14:12]**)位来选择**TMx \_EXT**管脚的边沿转变检测方式。

在事件捕获模式，软件不用考虑定时器计数器工作模式的选择，只有当检测到**TMx \_EXT**管脚有边沿变化时捕获事件才会发生。

可以通过**CAPIEN** (**TIMERx\_EXTCTL[5]**)使能捕捉中断功能。当 **TMx \_EXT** 引脚上的电平变化符合设置时，**CAPIF**位将会被置1。

如果CPU不清除**CAPIF**状态标志，用户应知道此时的定时器会保持**TIMERx\_CAP**寄存器的值不变，且不会保存新的捕获值。

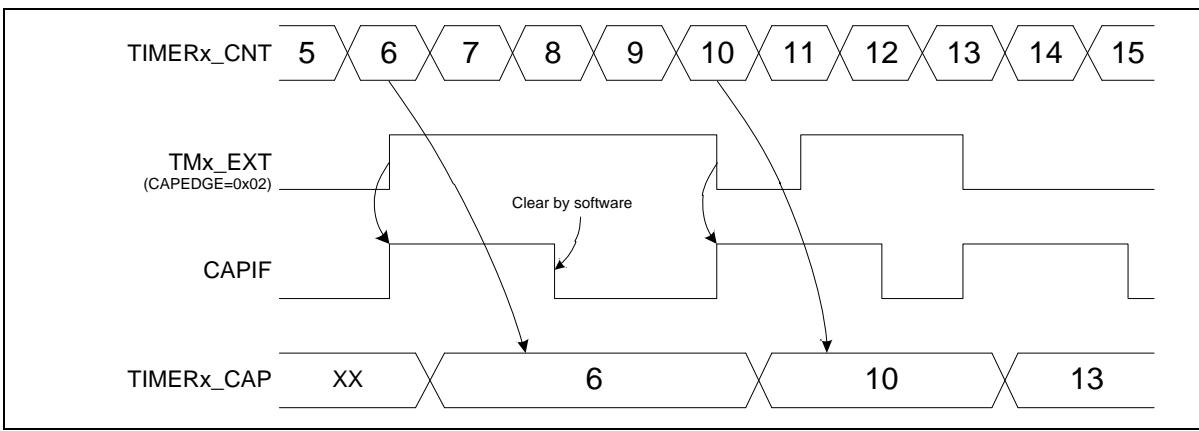


图 6.7-9 外部捕获模式

当**CAPSRC**(**TIMERx\_CTL[22]**) 是1，捕获事件可以由ACMP0 (**ACMPSSEL** (**TIMERx\_EXTCTL[8]**) =0 ) 或ACMP1 (**ACMPSSEL** (**TIMERx\_EXTCTL[8]**) =1) 上的内部输出信号触发。

#### 6.7.5.9 外部复位计数器模式

当捕获事件产生，定时器同样提供捕获事件复位计数器功能来复位**CNT** (**TIMERx\_CNT[23:0]**)的值。在该模式，**CAPFUNCS** (**TIMERx\_EXTCTL[4]**)位必须设置为1来选择**TMx \_EXT**转变或内部ACMPx输出信

号时用作为事件复位计数器。

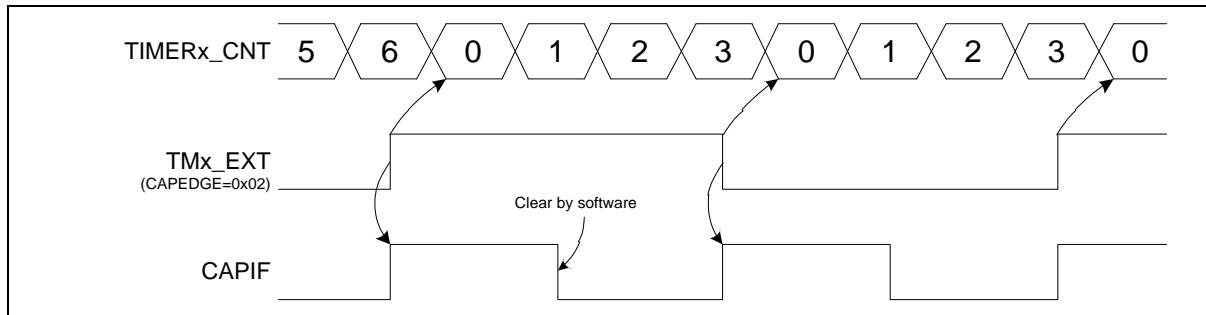


图 6.7-10 外部复位计数器模式

#### 6.7.5.10 定时器触发功能

定时器控制器提供定时器超时溢出中断或捕获中断来触发PWM, ADC, DAC 和 PDMA。如果TRGSSEL (TIMERx\_TRGCTL[0])为0, 超时溢出中断信号用于触发PWM, ADC, DAC 和 PDMA。如果TRGSSEL (TIMERx\_TRGCTL[0])为1, 捕获中断信号用于触发PWM, ADC, DAC 和 PDMA。

当TRGPWM (TIMERx\_TRGCTL[1])被置1, 如果定时中断信号产生, 定时器控制器将产生一个触发脉冲作为PWM外部时钟源。

当TRGADC (TIMERx\_TRGCTL[2])被置1, 如果定时中断信号产生, 定时器控制器将触发ADC开始转换。

当TRGDAC (TIMERx\_TRGCTL[3])被置1, 如果定时中断信号产生, 定时器控制器将触发DAC开始转换。

当TRGPDMA (TIMERx\_TRGCTL[4]) 被置1, 如果定时中断信号产生, 定时器控制器将触发PDMA。

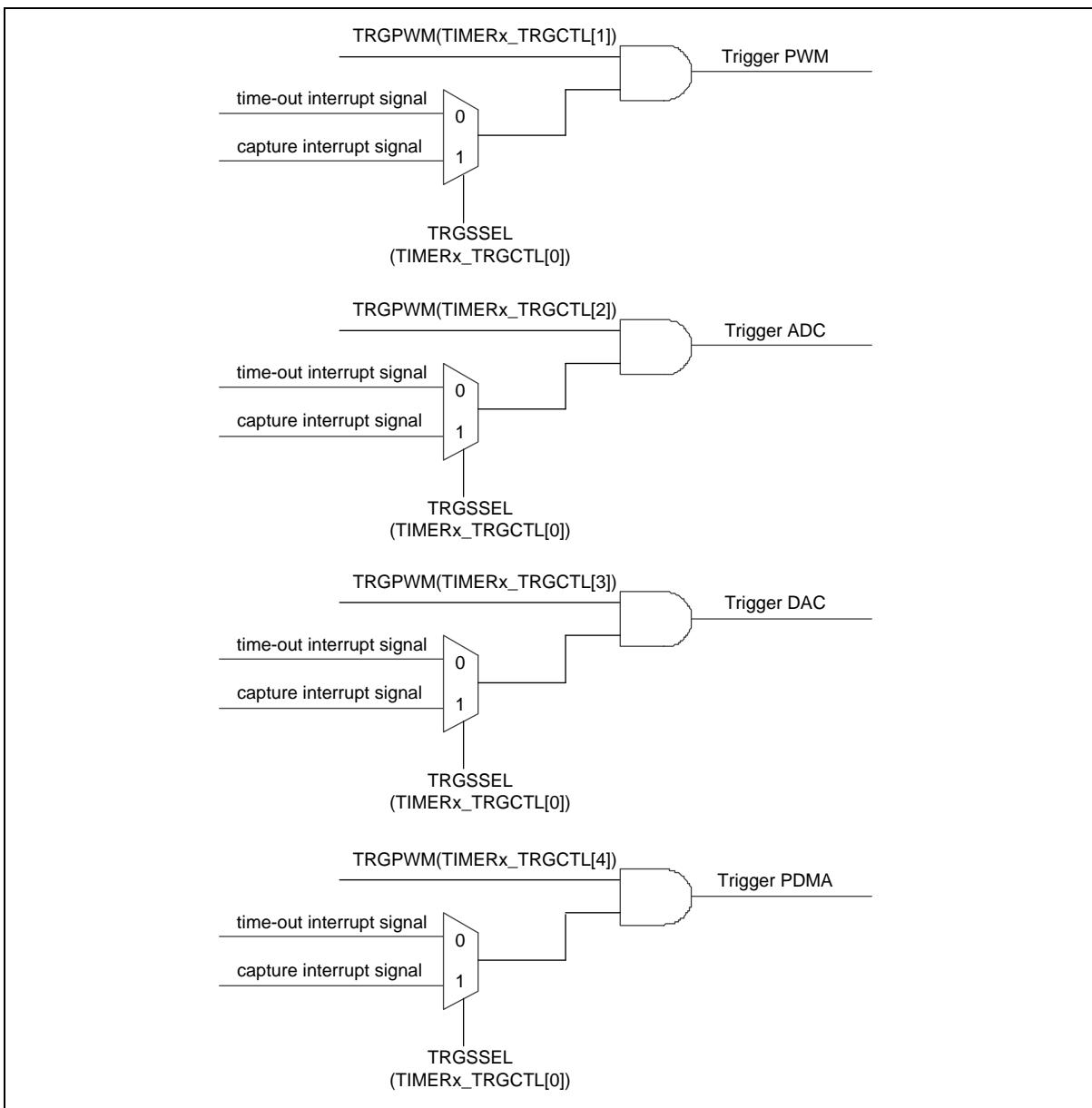


图 6.7-11 内部定时器触发

#### 6.7.5.11 定时器间互触发捕获模式

在该模式下，Timer0/2将会被强制工作在计数器模式，每有一个外部事件计数一次，并会在特定的时刻产生相应的信号 (INTR\_TMR\_TRG) 触发Timer1/3开始或停止计数。而Timer1/3被强制工作在捕获模式，由Timer0/2计数器状态触发开始或停止计数。

设置 Timer0 定时器间互触发捕获使能，触发计数捕获功能强制在Timer1。设置 Timer2 定时器间互触发捕获使能，触发计数捕获功能强制在Timer3。

#### 开始触发

Timer0/2的INTRGEN (TIMERx\_CTL[19])被置位，当Timer0/2计数器的值从 0x0 变为 0x1 时，Timer0/2产生一个上升沿触发信号。Timer1/3计数器则自动立即开始计数。

### 停止触发

当 Timer0/2 CNT 达到 Timer0/2 CMPDAT 值，Timer0/2 会产生一个下降跳变信号。Timer0/2 计数器模式功能被禁止，INTRGEN (TIMERx\_CTL[19]) 被硬件清0。然后 Timer1/3 停止计数，同时 Timer1/3 CNT 的值保存到 Timer1/3 CAPDAT (TIMERx\_CAP[23:0])。

用户使用定时器间互触发模式测量外部事件(TMx)的周期会更精确。图 6.7-12 是定时器间互触发模式，定时器0用作事件计数，定时器1用作触发计数捕获模式的例子。

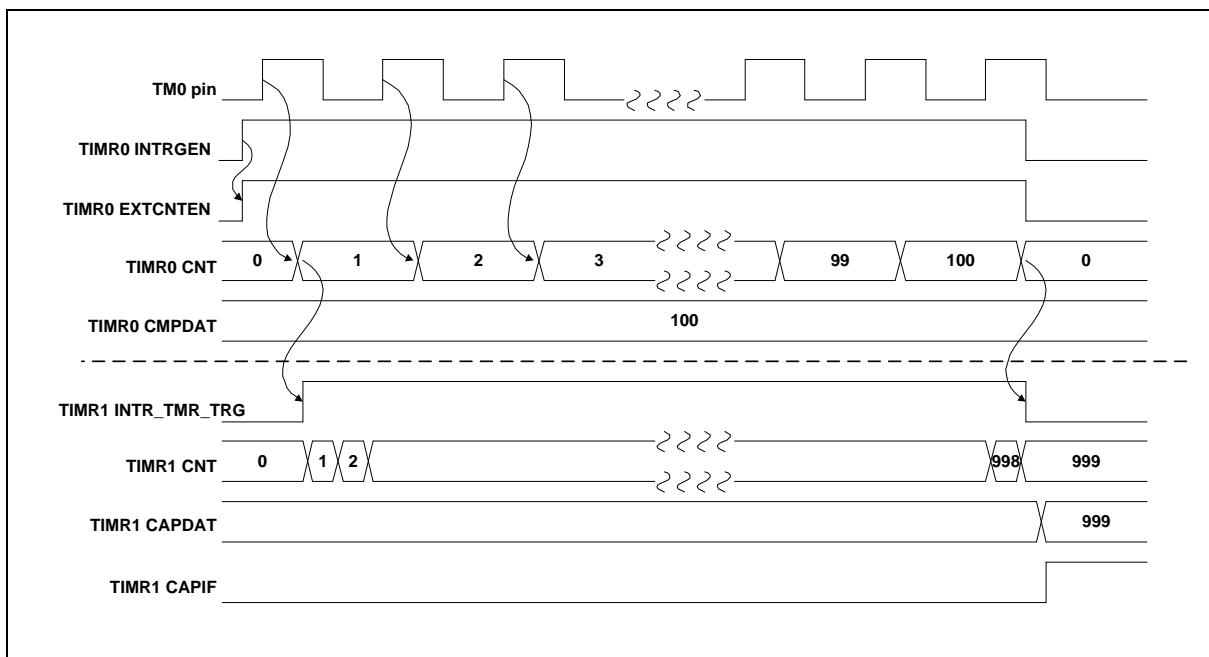


图 6.7-12 定时器间互触发捕获时序

## 6.7.6 PWM 功能描述

### 6.7.6.1 PWM 预分频

PWM 预分频用于分频时钟源，PWM 时钟被 (CLKPSC + 1) 分频。通过 CLKPSC (TIMERx\_PWMCLKPSC[11:0]) 设置分频值。图 6.7-13 是在上数计数模式 PWM 预分频的波形。

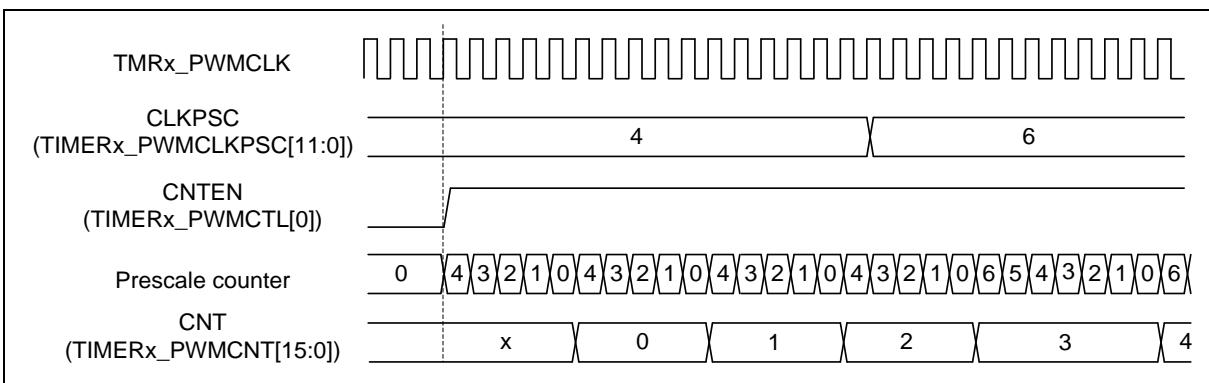


图 6.7-13 PWM 上数计数模式的预分频波形

### 6.7.6.2 PWM 计数器

PWM 支持三种计数器操作模式，上数，下数和上下数计数模式。

#### 6.7.6.3 上计数模式

当PWM设置为上计数模式，CNTTYPE (TIMERx\_PWMCTL[2:1])为0x0，计数器开始从0到PERIOD (TIMERx\_PWMPERIOD[15:0])向上计数。当前计数值可以从CNT (TIMERx\_PWMCNT[15:0]).读出。当计数器和预分频器计数到0将产生零点事件，当计数到PERIOD且预分频器计数到0将产生周期点事件。下图是一个向上计数器例子，PWM周期时间=  $(\text{PERIOD}+1) * (\text{CLKPSC}+1) * \text{TMRx\_PWMCLK}$ 。

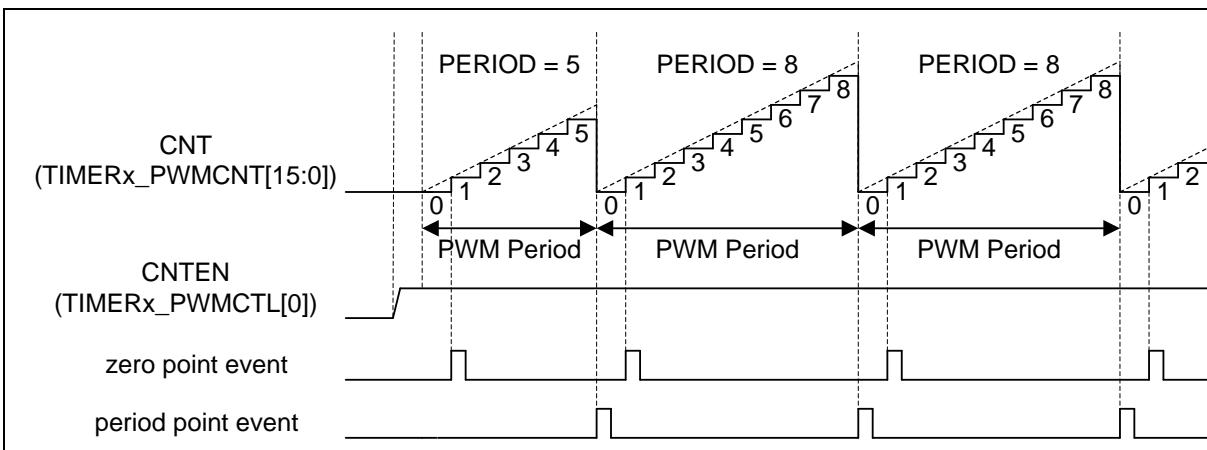


图 6.7-14 PWM 上计数模式

#### 6.7.6.4 下计数模式

当PWM设置为下计数模式，CNTTYPE (TIMERx\_PWMCTL[2:1])为0x1，计数器开始从PERIOD到0向下计数。当前计数值可以从CNT (TIMERx\_PWMCNT[15:0]).读出。当计数器和预分频器计数到0将产生零点事件，当计数到PERIOD且预分频器计数到0将产生周期点事件。下图是一个向下计数器例子，PWM周期时间=  $(\text{PERIOD}+1) * (\text{CLKPSC}+1) * \text{TMRx\_PWMCLK}$ 。

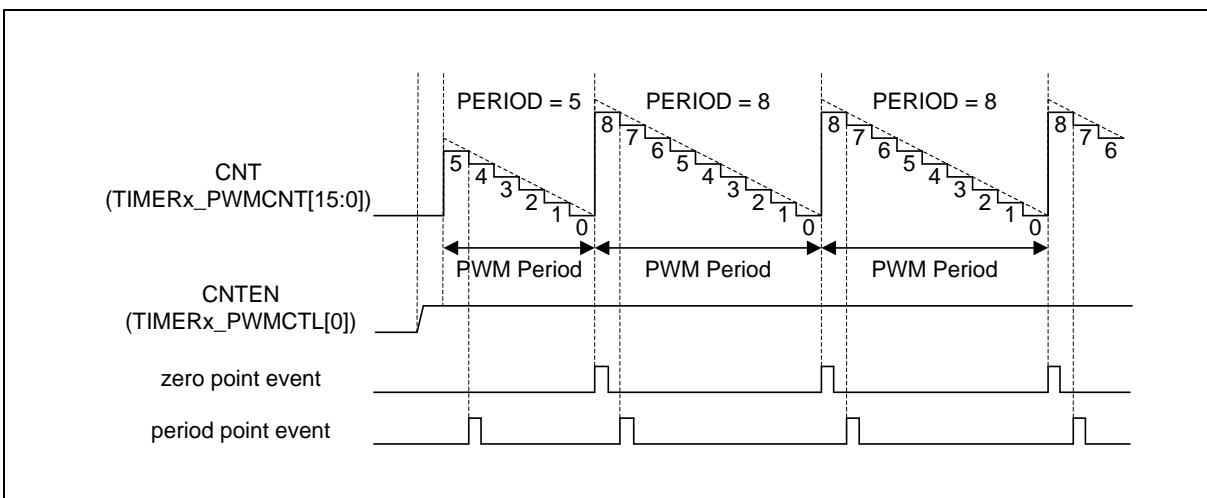


图 6.7-15 PWM 下计数模式

### 6.7.6.5 上下计数模式

当PWM设置为上下计数模式，CNTTYPE (TIMERx\_PWMCTL[2:1])为0x2，计数器开始从0到PERIOD向上计数然后再向下计数到0。当前计数值可以从CNT (TIMERx\_PWMCNT[15:0])读出。当计数器和预分频器计数到0将产生零点事件，当计数到PERIOD且预分频器计数到0将产生中心点事件。下图是一个上下计数器例子，PWM周期时间 =  $(2 * \text{PERIOD}) * (\text{CLKPSC}+1) * \text{TMRx\_PWMCLK}$ 。DIRF (TIMERx\_PWMCNT[16])是计数器的方向指示标志，该标志为1表示向上计数，为0表示向下计数。

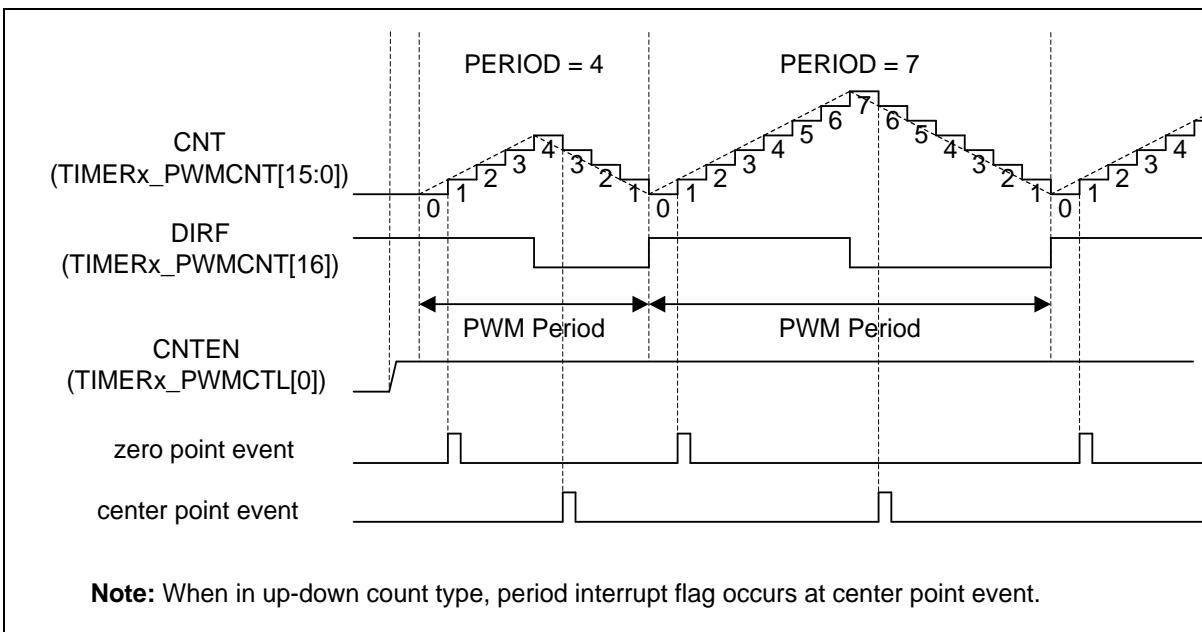


图 6.7-16 PWM 上下计数模式

### 6.7.6.6 PWM 计数器操作模式

PWM计数器支持两种操作模式：单次模式和自动重载模式。如果CNTMODE (TIMERx\_PWMCTL[3])=1，计数器工作在单次模式，如果CNTMODE (TIMERx\_PWMCTL[3])=0计数器工作在自动重载模式。

在这两种模式中，CMP (TIMERx\_PWMCMPPDAT[15:0]) 和 PERIOD (TIMERx\_PWMPERIOD[15:0]) 应先写入，然后设置CNTEN (TIMERx\_PWMCTL[0]) 为1开始计数器工作。

在单次模式，PWM计数器值会在PWM周期完成后，根据计数模式重载为默认值。用户可以写CMP来继续单次操作产生下一个单次脉冲，无论当前单次计数器是正在运行还是已完成。

在自动重载模式，PWM计数器是连续运行，如果在自动重载模式用户设置PERIOD为0 在一个PWM周期完成后，PWM计数器值会根据计数模式重载为默认值。

### 6.7.6.7 PWM 比较器

CMP (TIMERx\_PWMCMPPDAT[15:0])是PWM的比较寄存器，CMP的值会连续不断地和对应的计数器值比较。当计数器等于CMP，PWM产生一个比较点事件。该事件产生PWM输出脉冲，中断信号或触发ADC开始转换。在上下数模式，一个PWM周期会产生两个事件。如下图所示。CMPPU是上数比较点事件，CMPPD是下数比较点事件。

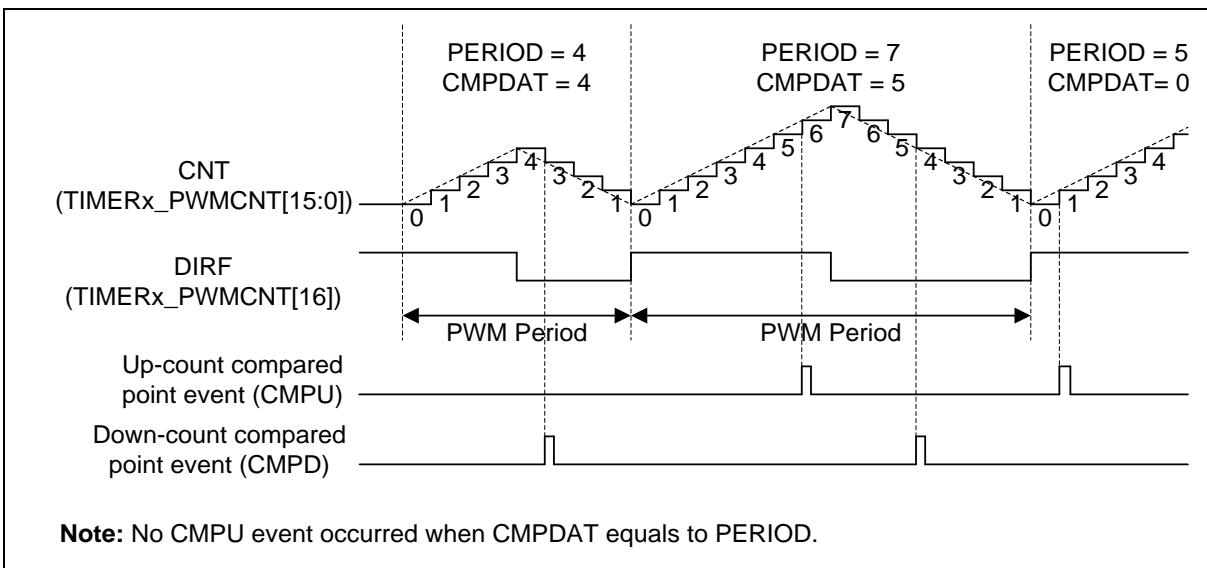


图 6.7-17 上下数模式 PWM 的比较器事件

#### 6.7.6.8 周期装载模式

当 IMMLDEN (TIMERx\_PWMCTL[9]) 位置 0，PWM 工作在周期装载模式。PBUF (TIMERx\_PWM\_PBUF[15:0]) 是 PERIOD 的缓存寄存器。CMPBUF (TIMERx\_PWM\_CMPBUF[15:0]) 是 CMP 的缓存寄存器。在周期装载模式，当每一个 PWM 周期完成。PERIOD (TIMERx\_PWM\_PERIOD[15:0]) 和 CMP (TIMERx\_PWM\_CMPDAT[15:0])。下图为上数计数模式周期装载时序。这里的 PERIOD DATA0 表示 PERIOD 的初始值，PERIOD DATA1 表示 PERIOD 数据第一次更新的值，依次类推。CMP 也遵循这个规则。下面步骤是图 6.7-18 的顺序。

1. 在点1，用户写 CMP DATA1 到 CMP
2. 在点2，PWM 周期结束的时候，周期装载 CMP DATA1 到 CMPBUF
3. 在点3，用户写 PERIOD DATA1 到 PERIOD.
4. 在点4，PWM 周期结束的时候，周期装载 PERIOD DATA1 到 PBUF
5. 在点5，用户写 PERIOD DATA2 到 PERIOD.,
6. 在点6，PWM 周期结束的时候，周期装载 PERIOD DATA2 到 PBUF

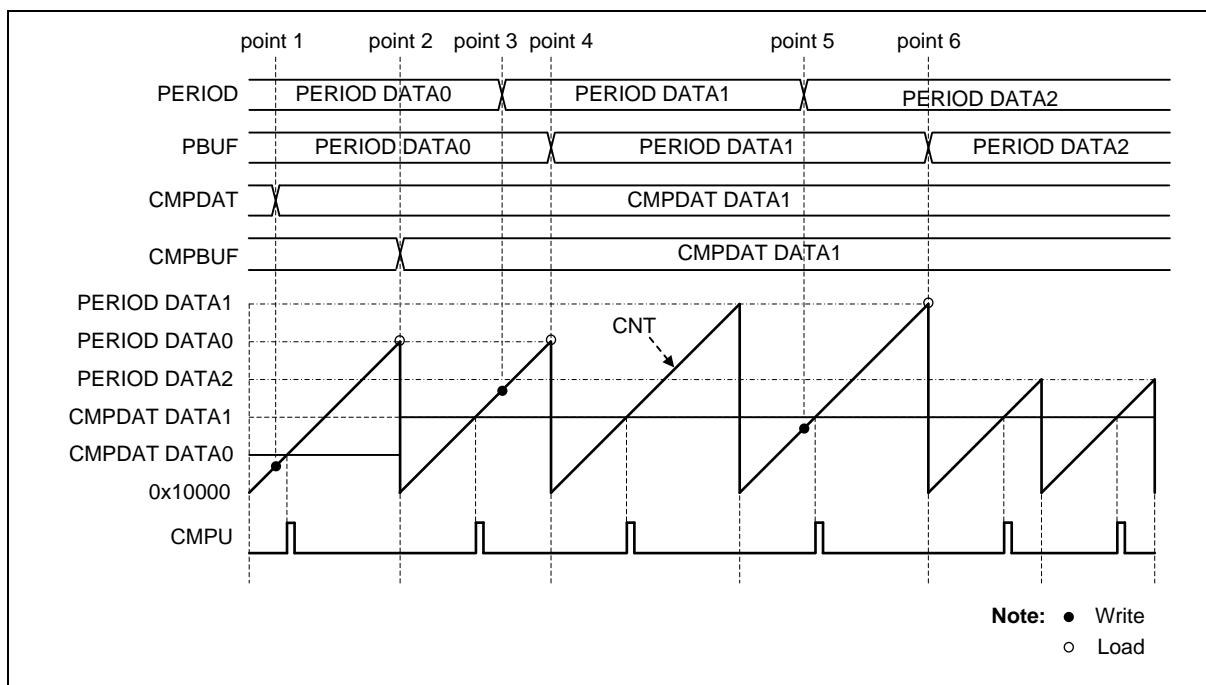


图 6.7-18 上数计数模式周期装载模式

#### 6.7.6.9 立即加载模式

当 IMMLDEN (TIMERx\_PWMCTL[9]) 位置 1，PWM工作在立即装载模式。当用户更新PERIOD (TIMERx\_PWMPeriod[15:0]) 或 CMP (TIMERx\_PWMcmpdat[15:0])，PERIOD 或 CMP 会在当前计数器计数完成后被装载到PBUF (TIMERx\_PWMpbuf[15:0]) 或 CMPBUF (TIMERx\_PWMcmpbuf[15:0])。如果更新PERIOD值小于当前计数器的值，计数器会循环计数。下面步骤是图 6.7-19的顺序。

1. 用户在点1写 CMP DATA1 且硬件会在当前计数器计数完成后装载CMP DATA1 到 CMPBUF
2. 用户在点2写PERIOD DATA1 且 PERIOD DATA1 大于当前计数器的值，PWM计数器会继续计数直到等于PERIOD DATA1 以完成一个PWM周期。
3. 用户在点3写PERIOD DATA2 且 PERIOD DATA2小于当前计数器值，PWM计数器继续计数到最大计数器值0xFFFF，且从0x10000 到 PERIOD DATA2 循环完成一个PWM周期。

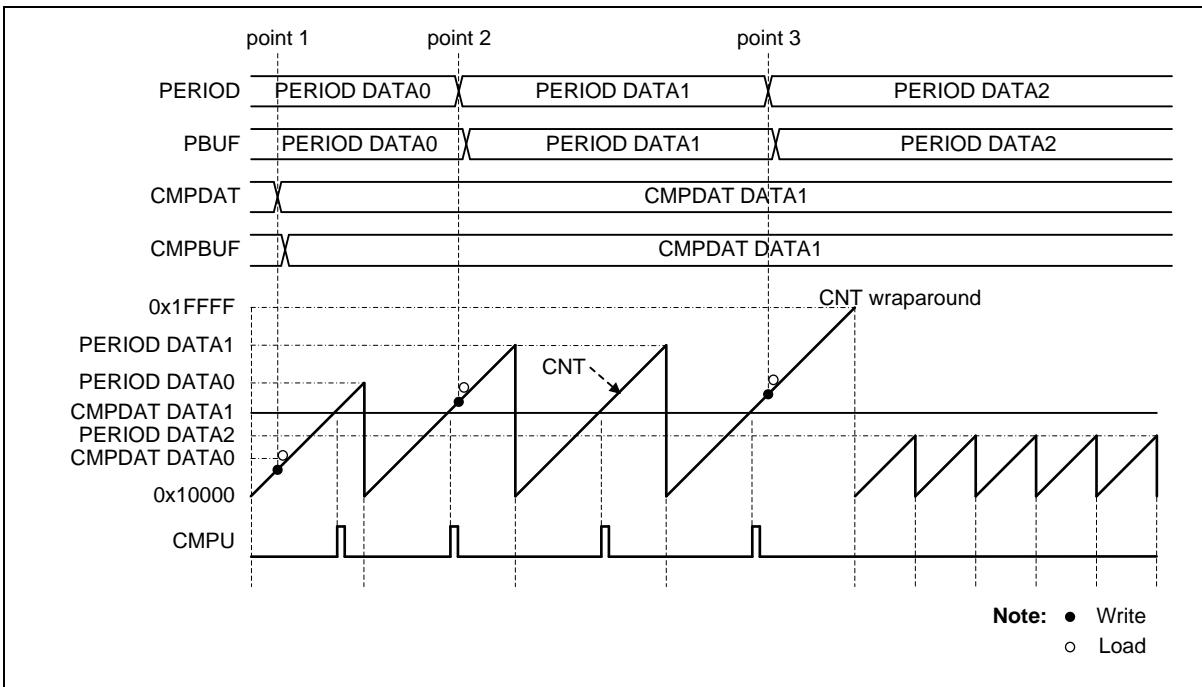


图 6.7-19 上计数模式立即装载模式

#### 6.7.6.10 PWM 脉冲产生器

PWM 脉冲产生器用计数器和比较器事件产生 PWM 输出脉冲。在上数或下数计数模式，事件有0点事件，周期点事件。在上下数计数模式有中心点事件，在三种计数模式下，都有计数器等于比较器点事件。

每一个事件点可以产生 PWM 输出波形。图 6.7-20, 图 6.7-21 和 图 6.7-22.表示不同的计数模式。

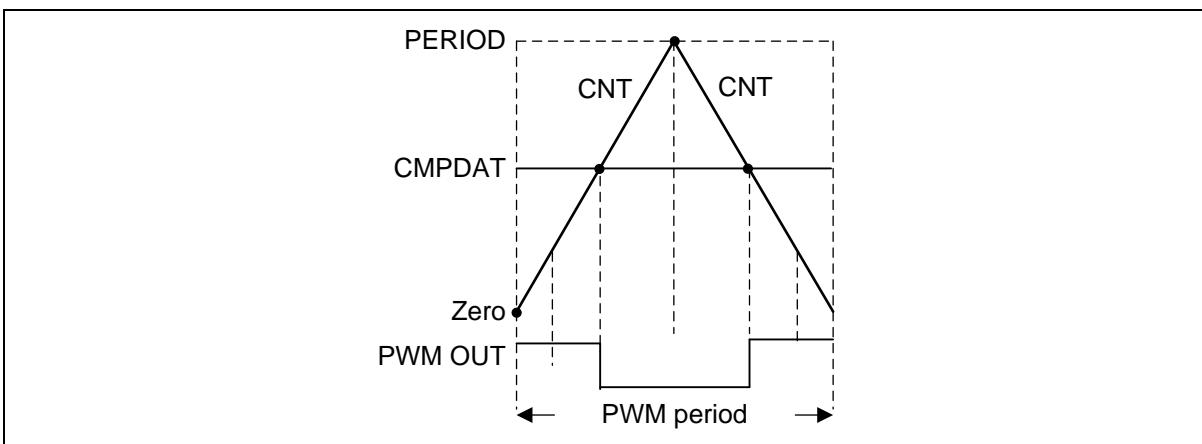


图 6.7-20 在上下数模式 PWM 脉冲的产生

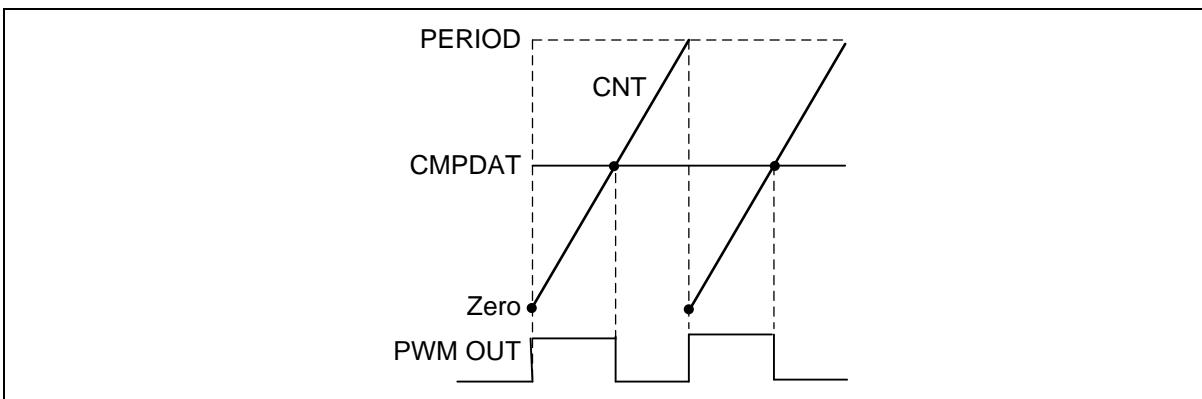


图 6.7-21 在上数模式 PWM 脉冲的产生

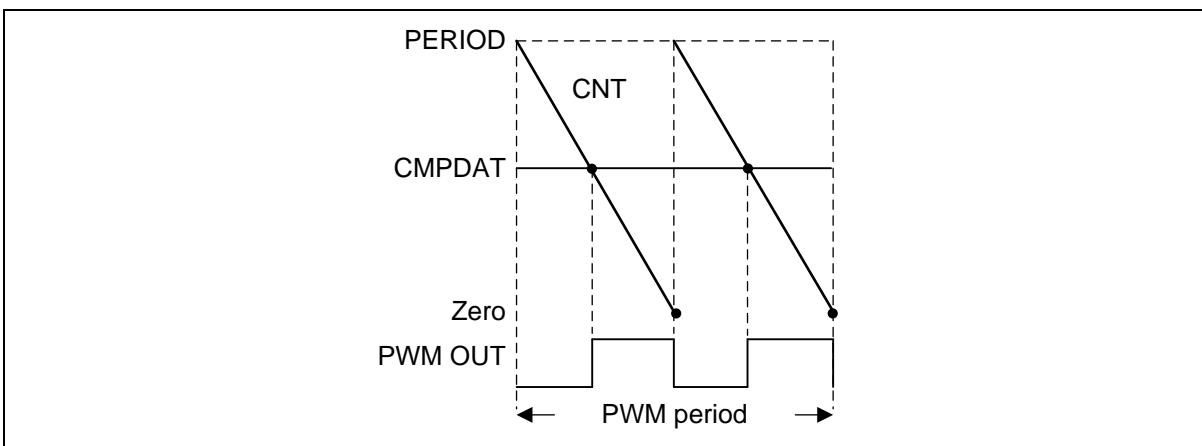


图 6.7-22 在下数模式 PWM 脉冲的产生

PWM事件可能有时同时产生，所以应该关注下表在不同模式下，事件的优先级。

优先级	0点和 CMPU 点事件 ( $CMP = 0$ )	PWM 输出
1 (高)	向上比较事件	低
2 (低)	0点事件	高

表 6.7-3 上数计数模式中 PWM 脉冲产生事件优先级

优先级	0点和 CMPD 点事件 ( $CMP = 0$ )	PWM 输出
1 (高)	0点事件	低
2 (低)	向下比较事件	高

优先级	周期点和 CMPD 点事件 ( $CMP = PERIOD$ )	PWM 输出
1 (高)	向下比较事件	高

2 (低)	周期事件	低
-------	------	---

表 6.7-4 下数计数模式中 PWM 脉冲产生事件优先级

优先级	CMPU 和 CMPD 点事件 ( $CMP = PERIOD$ )	PWM 输出
1 (高)	向下比较事件	高
2 (低)	向上比较事件	低

表 6.7-5 上下数计数模式中 PWM 脉冲产生事件优先级

根据事件优先级限制，仅在上数和上下数计数模式才支持0% 和 100% 暂空比周期的PWM输出。下图是在上数模式和上下数模式PWM占空比周期从0% 到 100%的例子。这里的PERIOD是4，CMP的值不同。

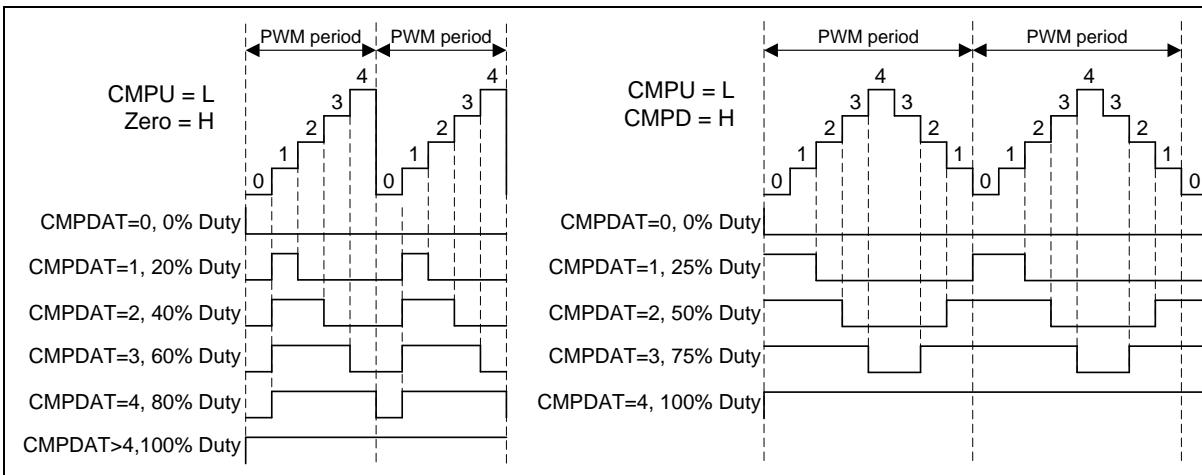


图 6.7-23 上数模式和上下数模式 PWM 暂空比周期 0% 到 100% 的例子

#### 6.7.6.11 PWM 输出模式

PWM支持两种输出模式，独立模式可以用于直流电机系统；互补模式带死区插入的功能，可以用于交流感应电机和永磁同步电机。

#### 6.7.6.12 独立模式

当 OUTMODE (TIMERx\_PWMCTL[16]) 位置0，PWM输出工作在独立模式。在该模式下，PWMx\_CH0 和 PWMx\_CH1可以输出相同的波形，如下图。

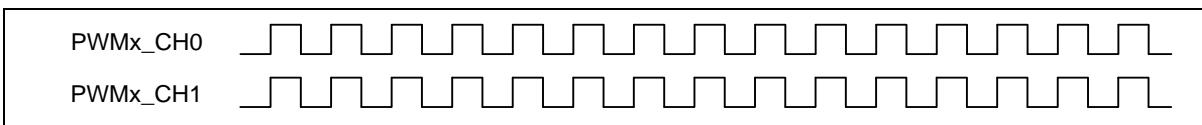


图 6.7-24 PWM 独立模式输出波形

#### 6.7.6.13 互补模式

当 OUTMODE (TIMERx\_PWMCTL[16]) 位置1，PWM输出工作在互补模式。在该模式下，PWMx\_CH0 和 PWMx\_CH1必须总是输出互补的波形，如下图。

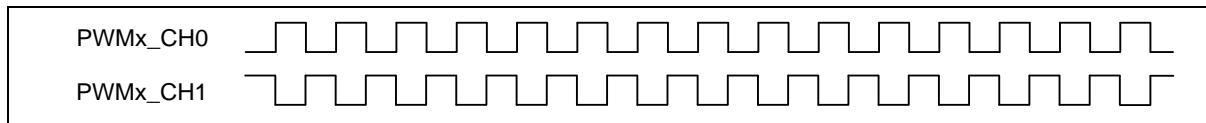


图 6.7-25 PWM 互补模式输出波形

#### 6.7.6.14 PWM 输出控制

PWM脉冲产生后，在独立模式有四个步骤控制输出波形，在互补模式有五步控制输出波形，用户可以设置POEN0 (TIMERx\_PWMPOEN[0]) 和 POEN1 (TIMERx\_PWMPOEN[1]) 为1使能PWMx\_CH0 和 PWMx\_CH1输出波形。

独立模式中，有屏蔽，刹车，管脚极性和输出使能控制输出波形，如下图。

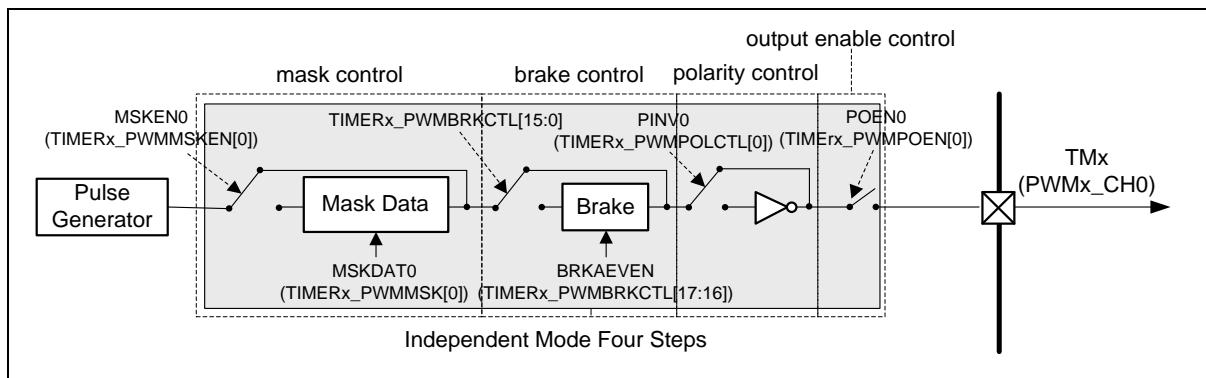


图 6.7-26 独立模式下 PWMx\_CH0 输出控制

互补模式中，多一个死区控制，其他四个步骤和独立模式一样。如下图控制PWMx\_CH0 和 PWMx\_CH1输出。

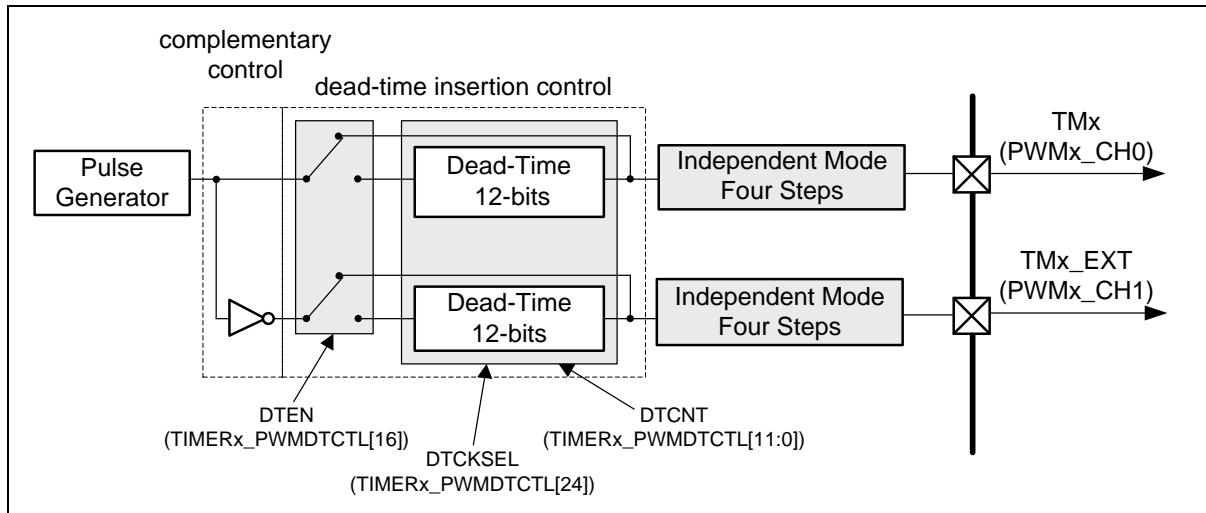


图 6.7-27 互补模式下 PWMx\_CH0 和 PWMx\_CH1 输出控制

#### 6.7.6.15 死区插入控制

互补应用中，互补通道也许用来驱动外部设备例如开关电源，在互补输出通道间死区发生器插入一个低电平间隔。通过设置DTEN (TIMERx\_PWMDTCTL[16])位来使能死区功能，DTCNT (TIMERx\_PWMDTCTL[11:0]) 和 DTCKSEL (TIMERx\_PWMDTCTL[24])用于控制死区间隔，死区

时间可以通过以下公式计算：

死区时间间隔 =  $(DTCNT + 1) * TMRx\_PWMCLK$  周期 (如果 DTCKSEL 是 0)

死区时间间隔 =  $(DTCNT + 1) * TMRx\_PWMCLK * (CLKPSC + 1)$  周期 (如果 DTCKSEL 是 1)

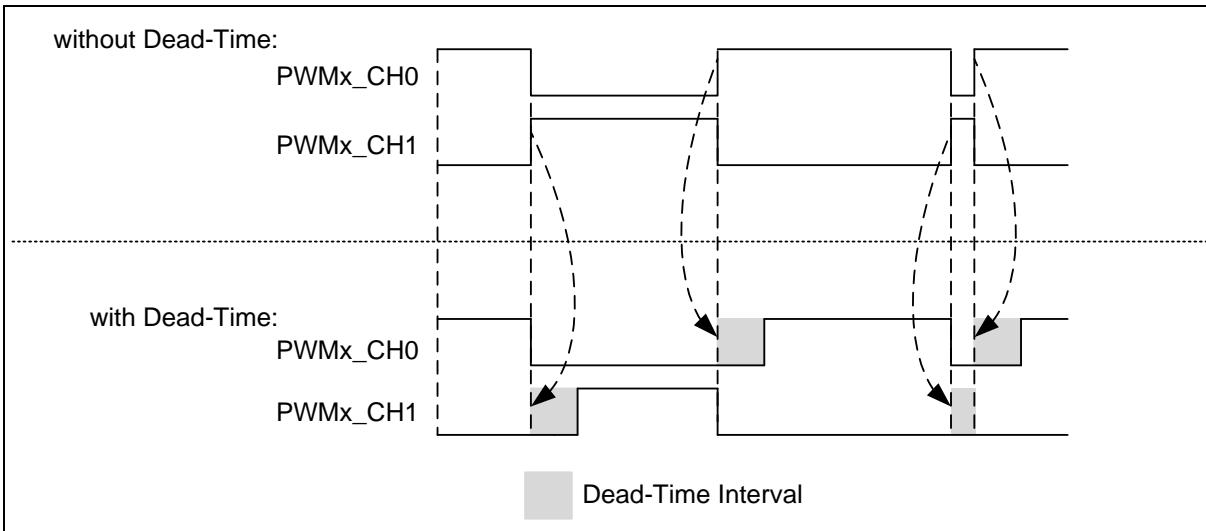


图 6.7-28 死区插入

#### 6.7.6.16 PWM 屏蔽输出控制

通过设置 MSKEN0/1 (TIMERx\_PWMMSKEN[1:0]) 和 MSKDAT0/1 (TIMERx\_PWMMSK[1:0])，PWMx\_CH0/CH1 输出值可以被屏蔽到指定的逻辑状态。当控制各种电子换向电机(ECM)如BLDC，PWM输出屏蔽功能是很有用的。下图是 PWMx\_CH0 和 PWMx\_CH1 中 PWM 屏蔽控制的例子。

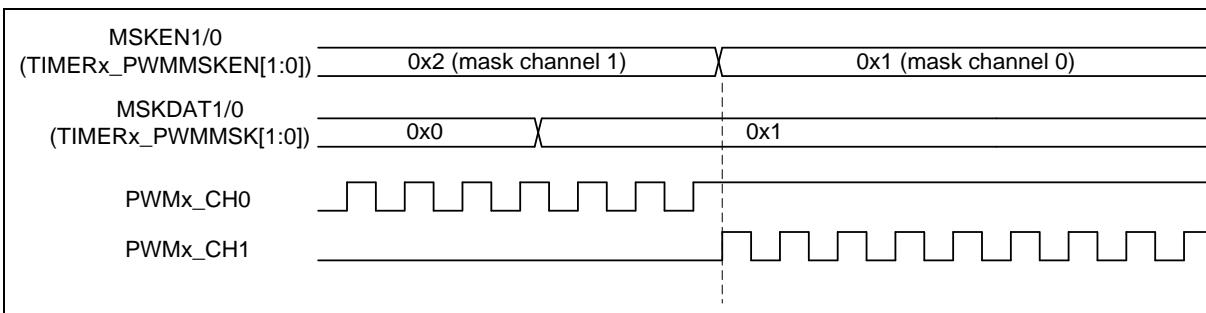


图 6.7-29 PWM 输出屏蔽控制波形

#### 6.7.6.17 PWM 刹车控制

每一个 PWM 产生器支持一个外部输入刹车管脚作为 PWM 刹车事件源。用户可以通过 BKPINSRC (TIMERx\_PWMBNF[17:16]), PWMx\_BRAKEy ( $x=0,1$ ,  $y=0,1$ ) 选择刹车管脚源。有一个 3 位噪音滤波计数器来滤波外部刹车管脚的信号。用户可以设置 BRKNFEN (TIMERx\_PWMBNF[0]) 位来使能刹车管脚噪音滤波功能且噪音滤波采样时钟可以通过设置 BRKNFSEL (TIMERx\_PWMBNF[3:1]) 来选择以适应不同的噪音特性。通过设置 BRKFCNT (TIMERx\_PWMBNF[6:4]), 用户可以定义多少个采样时钟周期来识别有效的刹车信号边沿。此外刹车管脚极性可以通过设置 BRKPINV (TIMERx\_PWMBNF[7]) 来反转。设置 BRKPINV 为 0, 当 PWMx\_BRAKEy 管脚状态从低到高, 刹车事件发生。设置 BRKPINV 为 1, 当 PWMx\_BRAKEy 管脚状态从高到低, 刹车事件发生。

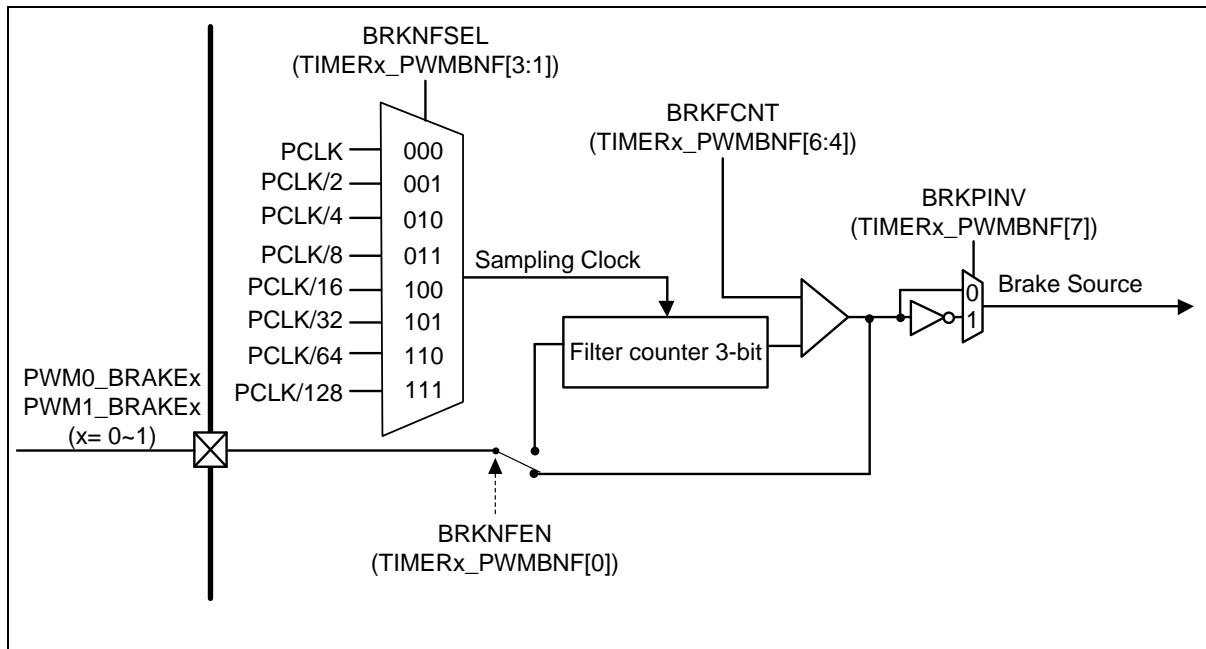


图 6.7-30 刹车管脚噪音滤波器框图

用户可以设置BRKAEVEN (TIMERx\_PWMFRKCTL[17:16]) 控制PWM<sub>x</sub>\_CH0刹车事件发生时的输出状态，设置BRKAODD (TIMERx\_PWMFRKCTL[19:18])控制PWM<sub>x</sub>\_CH1刹车事件发生时的输出状态。当刹车事件发生，有两个刹车侦测器源，边沿侦测刹车源和电平侦测刹车源。下图为PWM<sub>x</sub>\_CH0 和 PWM<sub>x</sub>\_CH1的刹车事件框图。

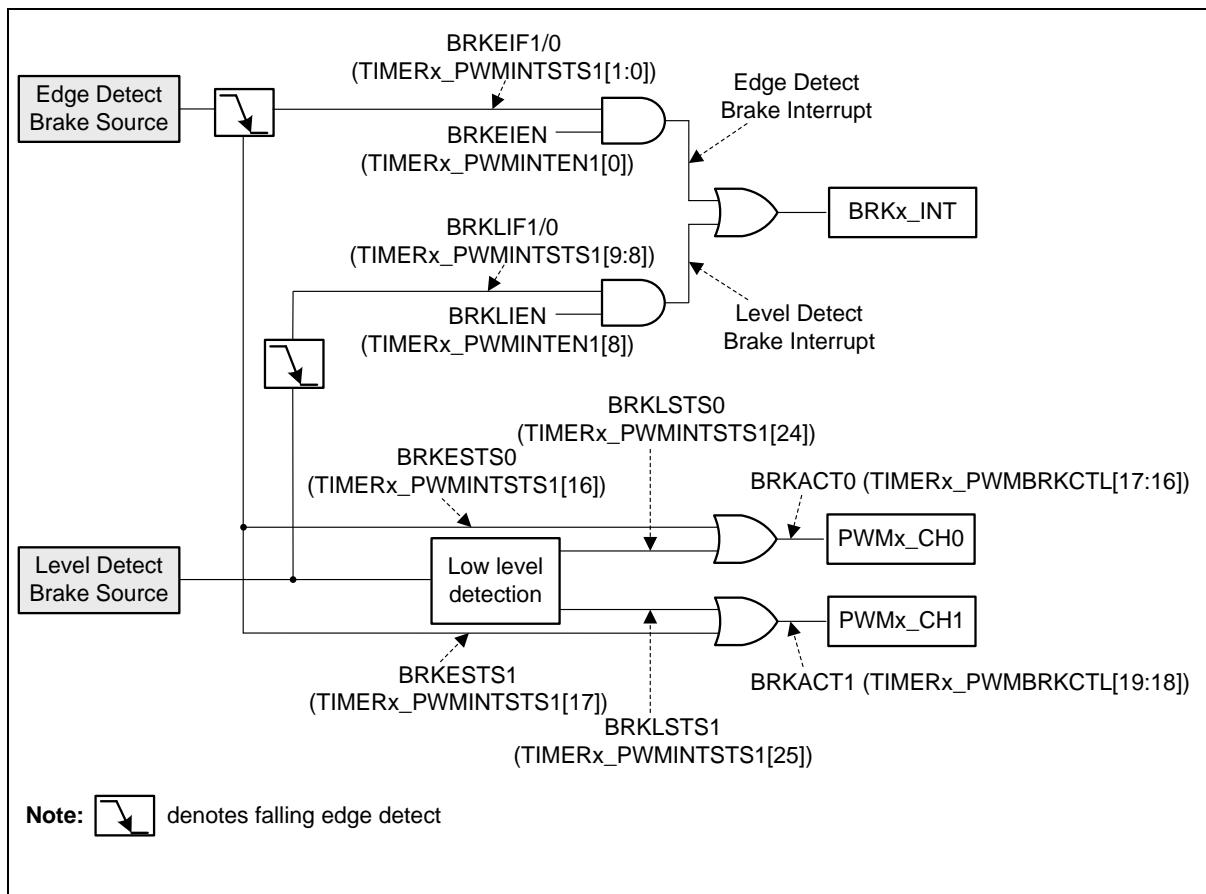


图 6.7-31 PWMx\_CH0 和 PWMx\_CH1 的刹车事件框图

当边沿侦测器侦测到刹车信号，刹车功能产生中断状态（ $\text{PWMx\_CH1}/0$  是  $\text{BRKEIF1}/0$  ( $\text{TIMERx\_PWMINTSTS1}[1:0]$ )）和刹车事件状态（ $\text{PWMx\_CH1}/0$  是  $\text{BRKESTS1}/0$  ( $\text{TIMERx\_PWMINTSTS1}[17:16]$ )）。中断状态  $\text{BRKEIF1}/0$  通过写1清0，刹车事件状态  $\text{BRKESTS1}/0$  会保持直到当对应的  $\text{BRKEIF1}/0$  标志已经清除且 PWM 产生器重新开始正常输出，下一个 PWM 周期开始。

下图是  $\text{PWMx\_CH0}$  和  $\text{PWMx\_CH1}$  边沿侦测器刹车波形的例子。在这个情况中，边沿侦测刹车源发生两次刹车事件。第一次刹车事件发生  $\text{BRKEIF0}$  和  $\text{BRKEIF1}$  标志都被置位，且  $\text{BRKESTS0}$  和  $\text{BRKESTS1}$  状态也被置位表示  $\text{PWMx\_CH0}$  和  $\text{PWMx\_CH1}$  的刹车状态。对于第一个发生的事件，用户写1清除  $\text{BRKEIF0}$ ，之后，当下一个 PWM 周期开始  $\text{BRKESTS0}$  由硬件清0， $\text{PWMx\_CH0}$  输出正常的波形，即使边沿刹车事件仍然在发生。同时  $\text{BRKESTS1}$  保持1且  $\text{PWMx\_CH1}$  在刹车状态保持输出低。第二次事件也触发相同的标志，但是在那时候用户写1清除  $\text{BRKEIF1}$ ，之后在下一个 PWM 周期开始后  $\text{PWMx\_CH1}$  正常输出。

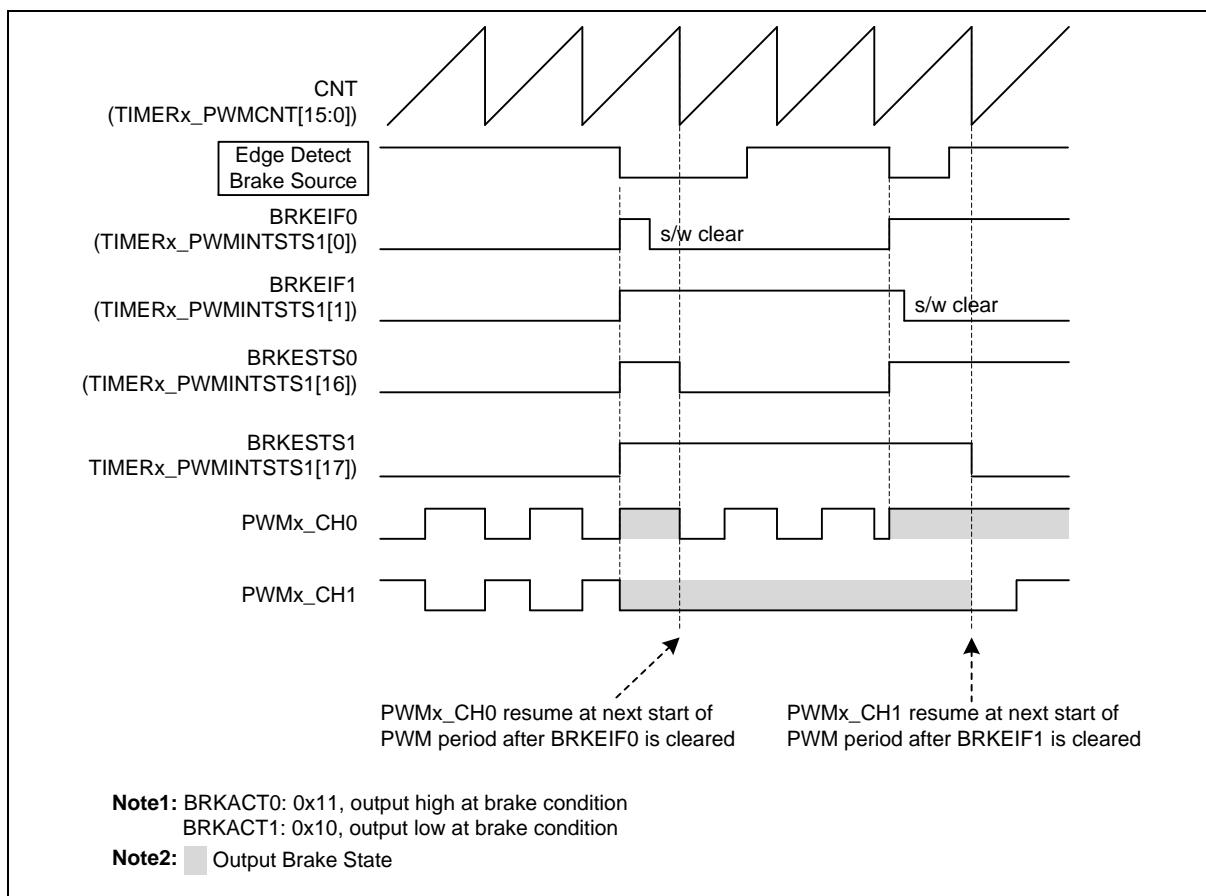


图 6.7-32 PWMx\_CH0 和 PWMx\_CH1 的边沿侦测器刹车波形

当电平侦测器侦测到刹车信号，刹车功能产生中断状态（PWMx\_CH1/0 是 BRKLIF1/0 (TIMERx\_PWMINTSTS1[9:8])）和刹车事件状态（PWMx\_CH1/0 是 BRKLSTS1/0 (TIMERx\_PWMINTSTS1[25:24])）。中断状态BRKLIF1/0通过写1清0，刹车事件状态BRKLSTS1/0仅在当前周期完成后且刹车条件移除清0，然后当下一个PWM周期开始PWM产生器重新开始正常输出。

下图是PWMx\_CH0 和 PWMx\_CH1电平侦测器刹车波形的例子。在这个情况中，BRKLIF0 和 BRKLIF1 仅表示刹车事件已发生。写1清除该标志，不影响BRKLSTS0 和 BRKLSTS1的刹车事件状态。BRKLSTS0 和 BRKLSTS1刹车状态都会在当刹车条件移除（无论BRKLIF0 和 BRKLIF1状态怎样），下一个PWM周期开始后自动清除。

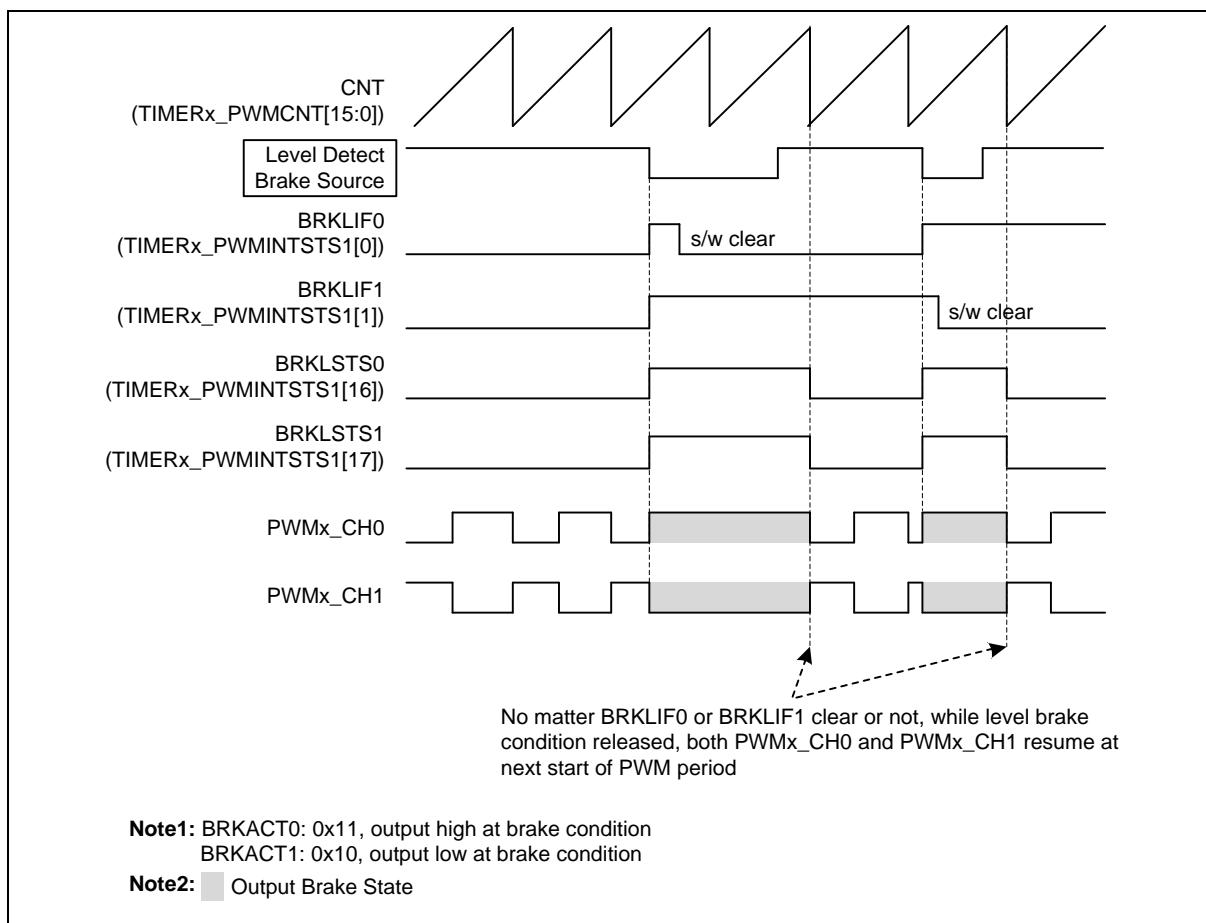


图 6.7-33 PWMx\_CH0 和 PWMx\_CH1 的电平侦测器刹车波形

两种侦测器侦测相同的五个刹车源，如下图。一个来自 PWM<sub>x</sub>\_BRAKE<sub>y</sub> (x=0,1 and y=0,1) 外部输入信号，两个来自内部ACMP比较器信号，一个来自系统失败事件和一个来自软件触发刹车事件。仅当内部 ACMP0\_O 或 ACMP1\_O 信号从低到高的时候ACMP刹车源才会被侦测到。

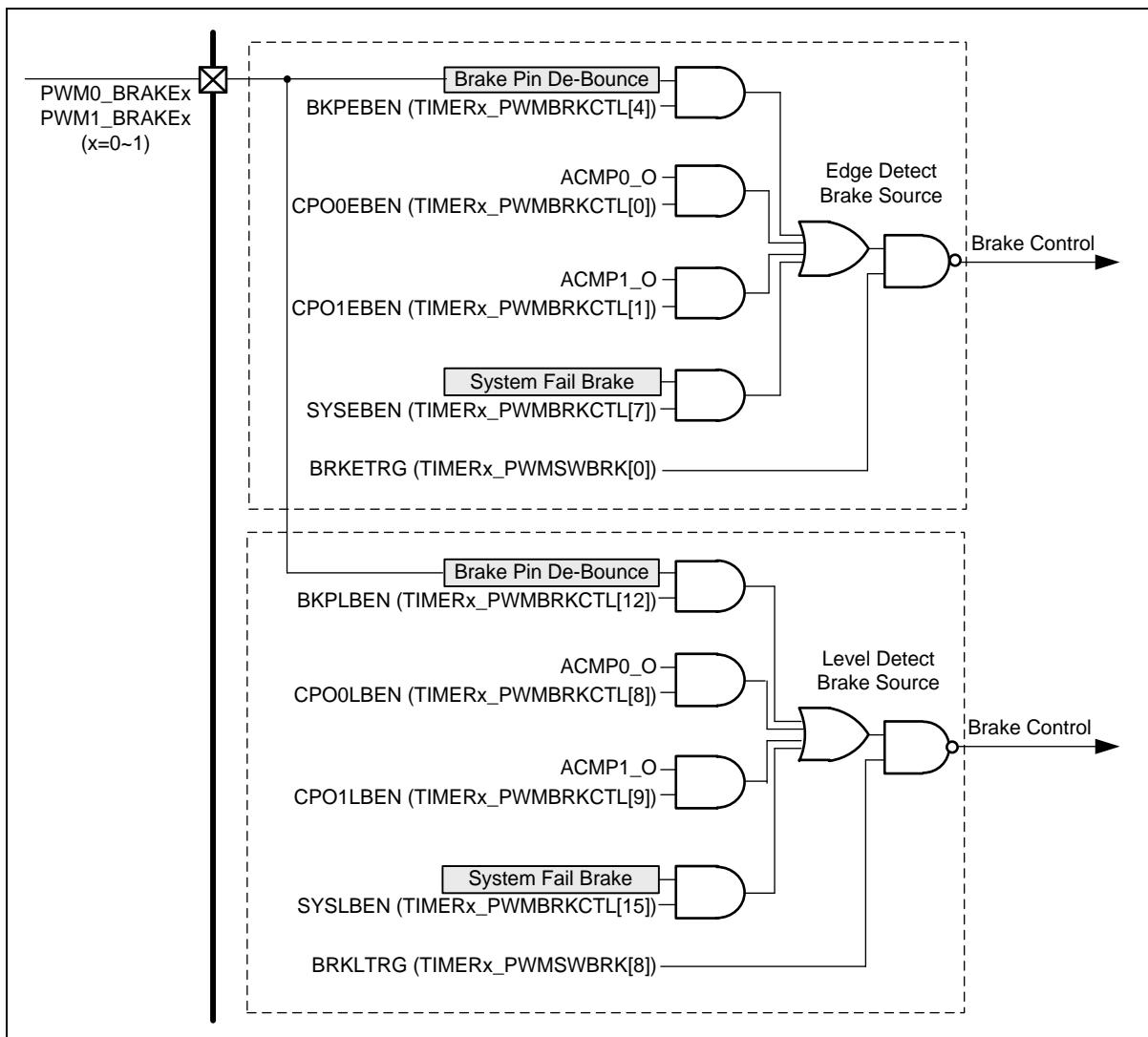


图 6.7-34 刹车源框图

在上面表述的刹车源中，来自系统失败事件的刹车源可以指定为不同的系统失败条件中的某一种。这些条件包括时钟失败、BOD侦测、SRAM校验错误和CPU锁死，如下图。

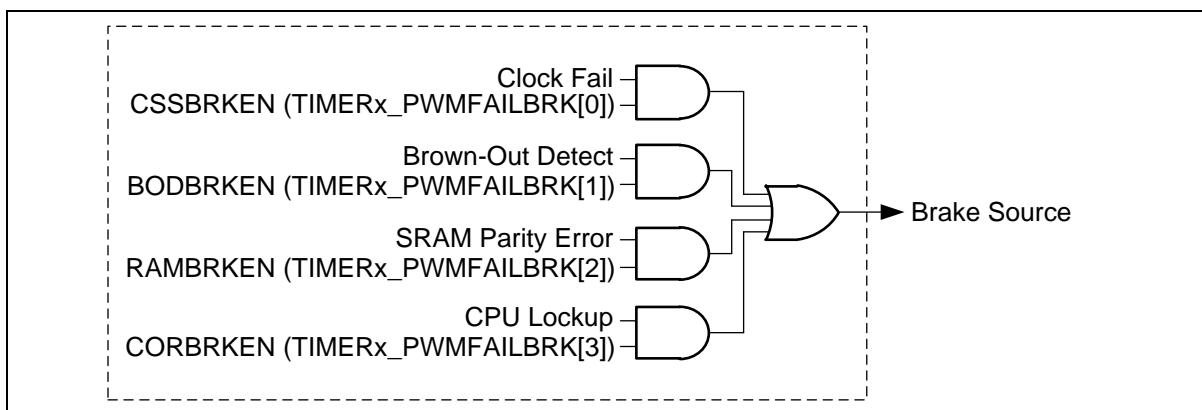


图 6.7-35 系统失败刹车框图

### 6.7.6.18 极性控制

PWM<sub>x</sub>\_CH0 和 PWM<sub>x</sub>\_CH1各自都有一个独立的极性控制器来控制PWM输出极性。用户可以控制 PWM<sub>x</sub>\_CH0（在 PINV0 (TIMER<sub>x</sub>\_PWMPOLCTL[0])) 和 PWM<sub>x</sub>\_CH1（在 PINV1 (TIMER<sub>x</sub>\_PWMPOLCTL[1])) 的极性状态。下图为 PWM<sub>x</sub>\_CH0 和 PWM<sub>x</sub>\_CH1 带极性控制和死区插入的输出。

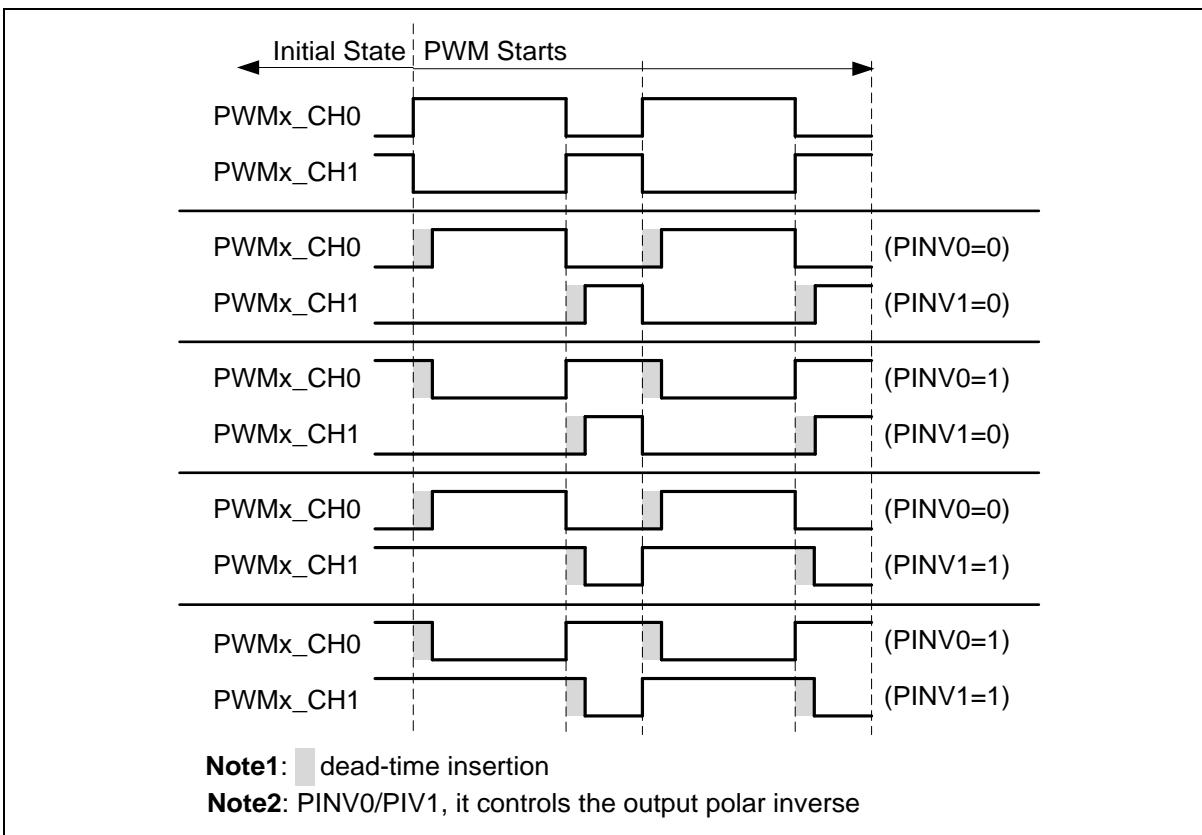


图 6.7-36 PWM<sub>x</sub>\_CH0 和 PWM<sub>x</sub>\_CH1 带死区插入的极性控制

### 6.7.6.19 PWM 中断产生器

每一个PWM都有独立的中断，如下图。

PWM中断(PWM<sub>x</sub>\_INT)来自PWM互补对事件。计数器可以产生零点中断标志ZIF (TIMER<sub>x</sub>\_PWMINSTS0[0]) 和周期点中断标志PIF (TIMER<sub>x</sub>\_PWMINSTS0[1])。当计数器等于比较值CMP (TIMER<sub>x</sub>\_PWMCMPDAT[15:0]), 不同的中断标志会被触发, 这取决于计数方向。如果计数器和CMP匹配发生在上数方向, 比较器上中断标志CMPIUF (TIMER<sub>x</sub>\_PWMINSTS0[2])被置位, 如果匹配发生在下数方向, 比较器下中断标志CMPDIF (TIMER<sub>x</sub>\_PWMINSTS0[3])被置位。如果对应中断使能位置位, 中断触发事件也会产生中断信号。当PWM刹车事件发生, 根据PWM刹车设置, 相关的中断事件会触发。

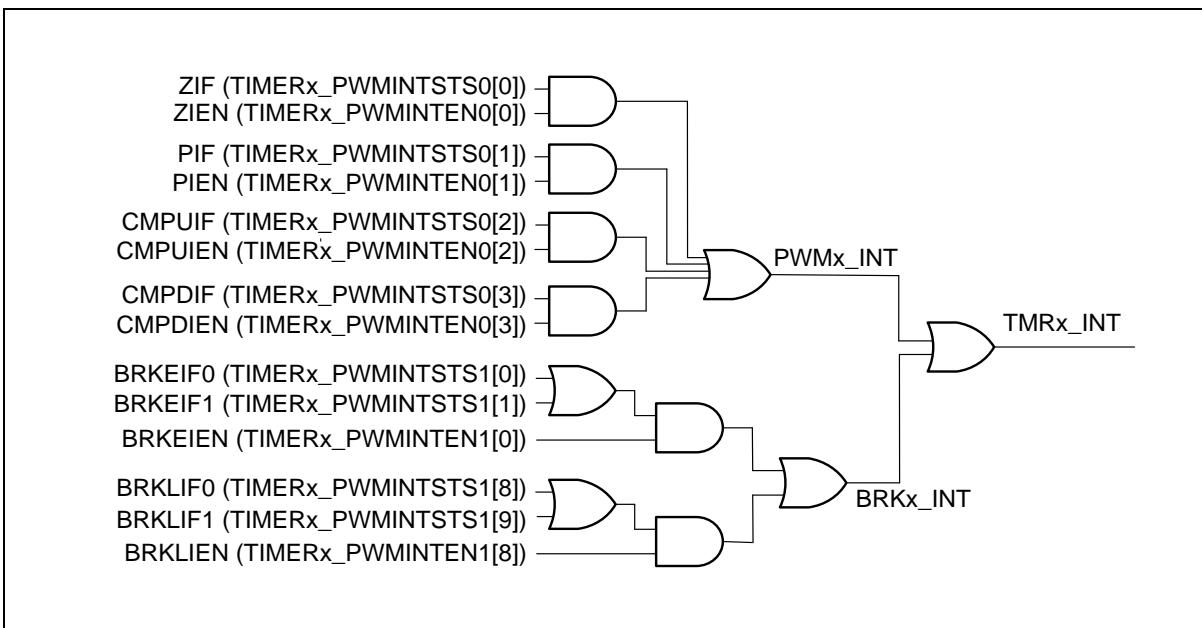


图 6.7-37 PWM 中断架构图

#### 6.7.6.20 PWM 触发 ADC 产生器

PWM计数事件是ADC转换触发源之一。PWM counter event can be one of the ADC conversion trigger source. 当 TRGEN (TIMERx\_PWMADCTS [7]) 被使能后，通过设置 TRGSEL (TIMERx\_PWMADCTS[3:0]) 选择 PWM计数事件触发ADC转换。

有五个PWM计数器事件可以选择作为触发源启动ADC转换，如下图。

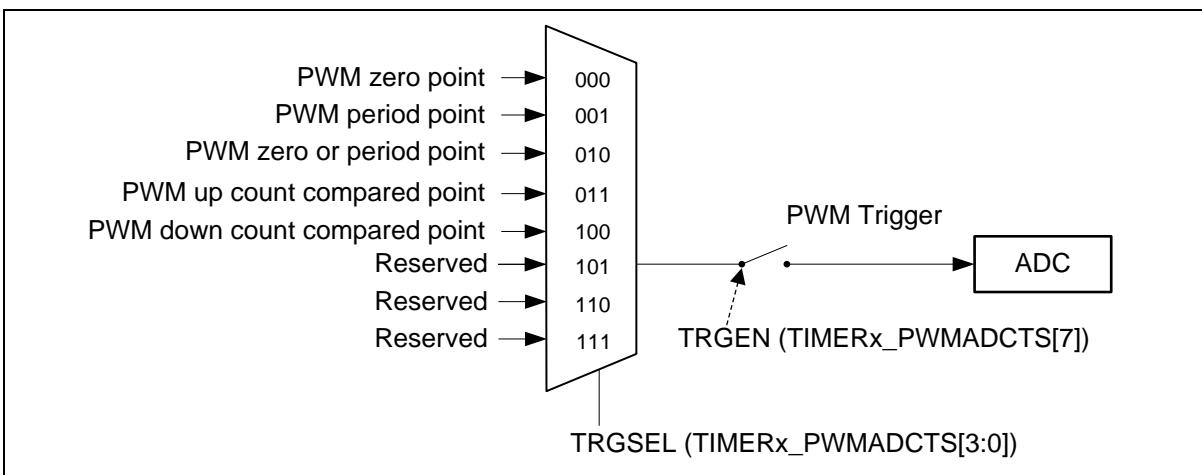


图 6.7-38 PWM 触发 ADC 框图

### 6.7.7 寄存器映射

**R:** 只读, **W:** 只写, **R/W:** 可读写

寄存器	偏移地址	R/W	描述	复位值
<b>TIMER 基地址:</b>				
<b>TMR01_BA = 0x4005_0000</b>				
<b>TMR23_BA = 0x4005_1000</b>				
<b>TIMER0_CTL</b>	TMR01_BA+0x00	R/W	Timer0 控制寄存器	0x0000_0005
<b>TIMER0_CMP</b>	TMR01_BA+0x04	R/W	Timer0 比较寄存器	0x0000_0000
<b>TIMER0_INTS_TS</b>	TMR01_BA+0x08	R/W	Timer0 中断状态寄存器	0x0000_0000
<b>TIMER0_CNT</b>	TMR01_BA+0x0C	R/W	Timer0 数据寄存器	0x0000_0000
<b>TIMER0_CAP</b>	TMR01_BA+0x10	R	Timer0 捕获数据寄存器	0x0000_0000
<b>TIMER0_EXT_CTL</b>	TMR01_BA+0x14	R/W	Timer0 外部控制寄存器	0x0000_0000
<b>TIMER0_EINT_STS</b>	TMR01_BA+0x18	R/W	Timer0 外部中断状态寄存器	0x0000_0000
<b>TIMER0_TRG_CTL</b>	TMR01_BA+0x1C	R/W	Timer0 触发控制寄存器	0x0000_0000
<b>TIMER0_ALT_CTL</b>	TMR01_BA+0x20	R/W	Timer0 选择控制寄存器	0x0000_0000
<b>TIMER0_PWM_CTL</b>	TMR01_BA+0x40	R/W	Timer0 PWM 控制寄存器	0x0000_0000
<b>TIMER0_PWM_CLKSRC</b>	TMR01_BA+0x44	R/W	Timer0 PWM 计数器时钟源寄存器	0x0000_0000
<b>TIMER0_PWM_CLKPSC</b>	TMR01_BA+0x48	R/W	Timer0 PWM 计数器时钟预分频寄存器	0x0000_0000
<b>TIMER0_PWM_CNTCLR</b>	TMR01_BA+0x4C	R/W	Timer0 PWM 清计数器寄存器	0x0000_0000
<b>TIMER0_PWM_PERIOD</b>	TMR01_BA+0x50	R/W	Timer0 PWM 周期寄存器	0x0000_0000
<b>TIMER0_PWM_CMPDAT</b>	TMR01_BA+0x54	R/W	Timer0 PWM 比较寄存器	0x0000_0000
<b>TIMER0_PWM_DTCTL</b>	TMR01_BA+0x58	R/W	Timer0 PWM 死区控制寄存器	0x0000_0000
<b>TIMER0_PWM_CNT</b>	TMR01_BA+0x5C	R	Timer0 PWM 计数器寄存器	0x0000_0000
<b>TIMER0_PWM_MSKEN</b>	TMR01_BA+0x60	R/W	Timer0 PWM 输出屏蔽使能寄存器	0x0000_0000
<b>TIMER0_PWM_MSK</b>	TMR01_BA+0x64	R/W	Timer0 PWM 输出屏蔽数据控制寄存器	0x0000_0000
<b>TIMER0_PWM_BNF</b>	TMR01_BA+0x68	R/W	Timer0 PWM 刹车管脚噪音滤波寄存器	0x0000_0000
<b>TIMER0_PWM_FAILBRK</b>	TMR01_BA+0x6C	R/W	Timer0 PWM 系统失败刹车控制寄存器	0x0000_0000

<b>TIMER0_PWM_BRKCTL</b>	TMR01_BA+0x70	R/W	Timer0 PWM 刹车控制寄存器	0x0000_0000
<b>TIMER0_PWM_POLCTL</b>	TMR01_BA+0x74	R/W	Timer0 PWM 管脚输出极性控制寄存器	0x0000_0000
<b>TIMER0_PWM_POEN</b>	TMR01_BA+0x78	R/W	Timer0 PWM 管脚输出使能寄存器	0x0000_0000
<b>TIMER0_PWM_SWBRK</b>	TMR01_BA+0x7C	W	Timer0 PWM 软件触发刹车控制 寄存器	0x0000_0000
<b>TIMER0_PWM_INTENO</b>	TMR01_BA+0x80	R/W	Timer0 PWM 中断使能寄存器 0	0x0000_0000
<b>TIMER0_PWM_INTEN1</b>	TMR01_BA+0x84	R/W	Timer0 PWM中断使能寄存器1	0x0000_0000
<b>TIMER0_PWM_INTSTS0</b>	TMR01_BA+0x88	R/W	Timer0 PWM 中断状态寄存器 0	0x0000_0000
<b>TIMER0_PWM_INTSTS1</b>	TMR01_BA+0x8C	R/W	Timer0 PWM中断状态寄存器1	0x0000_0000
<b>TIMER0_PWM_ADCTS</b>	TMR01_BA+0x90	R/W	Timer0 PWM ADC 触发源选择控制寄存器	0x0000_0000
<b>TIMER0_PWM_SCTL</b>	TMR01_BA+0x94	R/W	Timer0 PWM 同步控制寄存器	0x0000_0000
<b>TIMER0_PWM_STRG</b>	TMR01_BA+0x98	W	Timer0 PWM 同步触发寄存器	0x0000_0000
<b>TIMER0_PWM_STATUS</b>	TMR01_BA+0x9C	R/W	Timer0 PWM 状态寄存器	0x0000_0000
<b>TIMER0_PWM_PBUF</b>	TMR01_BA+0xA0	R	Timer0 PWM 周期缓存寄存器	0x0000_0000
<b>TIMER0_PWM_CMPBUF</b>	TMR01_BA+0xA4	R	Timer0 PWM 比较器缓存寄存器	0x0000_0000
<b>TIMER1_CTL</b>	TMR01_BA+0x100	R/W	Timer1 控制寄存器	0x0000_0005
<b>TIMER1_CMP</b>	TMR01_BA+0x104	R/W	Timer1比较寄存器	0x0000_0000
<b>TIMER1_INTS_TS</b>	TMR01_BA+0x108	R/W	Timer1中断状态寄存器	0x0000_0000
<b>TIMER1_CNT</b>	TMR01_BA+0x10C	R/W	Timer1数据寄存器	0x0000_0000
<b>TIMER1_CAP</b>	TMR01_BA+0x110	R	Timer1捕获数据寄存器	0x0000_0000
<b>TIMER1_EXT_CTL</b>	TMR01_BA+0x114	R/W	Timer1外部控制寄存器	0x0000_0000
<b>TIMER1_EINT_STS</b>	TMR01_BA+0x118	R/W	Timer1外部中断状态寄存器	0x0000_0000
<b>TIMER1_TRG_CTL</b>	TMR01_BA+0x11C	R/W	Timer1触发控制寄存器	0x0000_0000
<b>TIMER1_ALT_CTL</b>	TMR01_BA+0x120	R/W	Timer1选择控制寄存器	0x0000_0000
<b>TIMER1_PWM_CTL</b>	TMR01_BA+0x140	R/W	Timer1 PWM 控制寄存器	0x0000_0000
<b>TIMER1_PWM</b>	TMR01_BA+0x144	R/W	Timer1 PWM 计数器时钟源寄存器	0x0000_0000

CLKSRC				
TIMER1_PWM_CLKPSC	TMR01_BA+0x148	R/W	Timer1 PWM 计数器时钟预分频寄存器	0x0000_0000
TIMER1_PWM_CNTCLR	TMR01_BA+0x14C	R/W	Timer1 PWM 清计数器寄存器	0x0000_0000
TIMER1_PWM_PERIOD	TMR01_BA+0x150	R/W	Timer1 PWM 周期寄存器	0x0000_0000
TIMER1_PWM_CMPDAT	TMR01_BA+0x154	R/W	Timer1 PWM 比较寄存器	0x0000_0000
TIMER1_PWM_DTCTL	TMR01_BA+0x158	R/W	Timer1 PWM 死区控制寄存器	0x0000_0000
TIMER1_PWM_CNT	TMR01_BA+0x15C	R	Timer1 PWM 计数器寄存器	0x0000_0000
TIMER1_PWM_MSKEN	TMR01_BA+0x160	R/W	Timer1 PWM 输出屏蔽使能寄存器	0x0000_0000
TIMER1_PWM_MSK	TMR01_BA+0x164	R/W	Timer1 PWM 输出屏蔽数据控制寄存器	0x0000_0000
TIMER1_PWM_BNF	TMR01_BA+0x168	R/W	Timer1 PWM 刹车管脚噪音滤波寄存器	0x0000_0000
TIMER1_PWM_FAILBRK	TMR01_BA+0x16C	R/W	Timer1 PWM 系统失败刹车控制寄存器	0x0000_0000
TIMER1_PWM_BRKCTL	TMR01_BA+0x170	R/W	Timer1 PWM 刹车控制寄存器	0x0000_0000
TIMER1_PWM_POLCTL	TMR01_BA+0x174	R/W	Timer1 PWM 管脚输出极性控制寄存器	0x0000_0000
TIMER1_PWM_POEN	TMR01_BA+0x178	R/W	Timer1 PWM 管脚输出使能寄存器	0x0000_0000
TIMER1_PWM_SWBRK	TMR01_BA+0x17C	W	Timer1 PWM 软件触发刹车控制 寄存器	0x0000_0000
TIMER1_PWM_INTENO0	TMR01_BA+0x180	R/W	Timer1 PWM 中断使能寄存器 0	0x0000_0000
TIMER1_PWM_INTEN1	TMR01_BA+0x184	R/W	Timer1 PWM中断使能寄存器1	0x0000_0000
TIMER1_PWM_INTSTS0	TMR01_BA+0x188	R/W	Timer1 PWM 中断状态寄存器 0	0x0000_0000
TIMER1_PWM_INTSTS1	TMR01_BA+0x18C	R/W	Timer1 PWM中断状态寄存器1	0x0000_0000
TIMER1_PWM_ADCTS	TMR01_BA+0x190	R/W	Timer1 PWM ADC 触发源选择控制寄存器	0x0000_0000
TIMER1_PWM_SCTL	TMR01_BA+0x194	R/W	Timer1 PWM 同步控制寄存器	0x0000_0000
TIMER1_PWM_SSTRG	TMR01_BA+0x198	W	Timer1 PWM 同步触发寄存器	0x0000_0000
TIMER1_PWM_STATUS	TMR01_BA+0x19C	R/W	Timer1 PWM 状态寄存器	0x0000_0000
TIMER1_PWM_PBUF	TMR01_BA+0x1A0	R	Timer1 PWM 周期缓存寄存器	0x0000_0000

<b>TIMER1_PWM_CMPBUF</b>	TMR01_BA+0x1A4	R	Timer1 PWM 比较器缓存寄存器	0x0000_0000
<b>TIMER2_CTL</b>	TMR23_BA+0x00	R/W	Timer2 控制寄存器	0x0000_0005
<b>TIMER2_CMP</b>	TMR23_BA+0x04	R/W	Timer2比较寄存器	0x0000_0000
<b>TIMER2_INTS_TS</b>	TMR23_BA+0x08	R/W	Timer2中断状态寄存器	0x0000_0000
<b>TIMER2_CNT</b>	TMR23_BA+0x0C	R/W	Timer2数据寄存器	0x0000_0000
<b>TIMER2_CAP</b>	TMR23_BA+0x10	R	Timer2捕获数据寄存器	0x0000_0000
<b>TIMER2_EXT_CTL</b>	TMR23_BA+0x14	R/W	Timer2外部控制寄存器	0x0000_0000
<b>TIMER2_EINT_STS</b>	TMR23_BA+0x18	R/W	Timer2外部中断状态寄存器	0x0000_0000
<b>TIMER2_TRG_CTL</b>	TMR23_BA+0x1C	R/W	Timer2触发控制寄存器	0x0000_0000
<b>TIMER2_ALT_CTL</b>	TMR23_BA+0x20	R/W	Timer2选择控制寄存器	0x0000_0000
<b>TIMER2_PWM_CTL</b>	TMR23_BA+0x40	R/W	Timer2 PWM 控制寄存器	0x0000_0000
<b>TIMER2_PWM_CLKSRC</b>	TMR23_BA+0x44	R/W	Timer2 PWM 计数器时钟源寄存器	0x0000_0000
<b>TIMER2_PWM_CLKPSC</b>	TMR23_BA+0x48	R/W	Timer2 PWM 计数器时钟预分频寄存器	0x0000_0000
<b>TIMER2_PWM_CNTCLR</b>	TMR23_BA+0x4C	R/W	Timer2 PWM 清计数器寄存器	0x0000_0000
<b>TIMER2_PWM_PERIOD</b>	TMR23_BA+0x50	R/W	Timer2 PWM 周期寄存器	0x0000_0000
<b>TIMER2_PWM_CMPDAT</b>	TMR23_BA+0x54	R/W	Timer2 PWM 比较寄存器	0x0000_0000
<b>TIMER2_PWM_DTCTL</b>	TMR23_BA+0x58	R/W	Timer2 PWM 死区控制寄存器	0x0000_0000
<b>TIMER2_PWM_CNT</b>	TMR23_BA+0x5C	R	Timer2 PWM 计数器寄存器	0x0000_0000
<b>TIMER2_PWM_MSKEN</b>	TMR23_BA+0x60	R/W	Timer2 PWM 输出屏蔽使能寄存器	0x0000_0000
<b>TIMER2_PWM_MSK</b>	TMR23_BA+0x64	R/W	Timer2 PWM 输出屏蔽数据控制寄存器	0x0000_0000
<b>TIMER2_PWM_BNF</b>	TMR23_BA+0x68	R/W	Timer2 PWM 刹车管脚噪音滤波寄存器	0x0000_0000
<b>TIMER2_PWM_FAILBRK</b>	TMR23_BA+0x6C	R/W	Timer2 PWM 系统失败刹车控制寄存器	0x0000_0000
<b>TIMER2_PWM_BRKCTL</b>	TMR23_BA+0x70	R/W	Timer2 PWM 刹车控制寄存器	0x0000_0000
<b>TIMER2_PWM_POLCTL</b>	TMR23_BA+0x74	R/W	Timer2 PWM 管脚输出极性控制寄存器	0x0000_0000
<b>TIMER2_PWM</b>	TMR23_BA+0x78	R/W	Timer2 PWM 管脚输出使能寄存器	0x0000_0000

POEN				
TIMER2_PWM_SWBRK	TMR23_BA+0x7C	W	Timer2 PWM 软件触发刹车控制 寄存器	0x0000_0000
TIMER2_PWM_INTENO	TMR23_BA+0x80	R/W	Timer2 PWM 中断使能寄存器 0	0x0000_0000
TIMER2_PWM_INTEN1	TMR23_BA+0x84	R/W	Timer2 PWM中断使能寄存器1	0x0000_0000
TIMER2_PWM_INTSTS0	TMR23_BA+0x88	R/W	Timer2 PWM 中断状态寄存器 0	0x0000_0000
TIMER2_PWM_INTSTS1	TMR23_BA+0x8C	R/W	Timer2 PWM中断状态寄存器1	0x0000_0000
TIMER2_PWM_ADCTS	TMR23_BA+0x90	R/W	Timer2 PWM ADC 触发源选择控制寄存器	0x0000_0000
TIMER2_PWM_SCTL	TMR23_BA+0x94	R/W	Timer2 PWM 同步控制寄存器	0x0000_0000
TIMER2_PWM_STRG	TMR23_BA+0x98	W	Timer2 PWM 同步触发寄存器	0x0000_0000
TIMER2_PWM_STATUS	TMR23_BA+0x9C	R/W	Timer2 PWM 状态寄存器	0x0000_0000
TIMER2_PWM_PBUF	TMR23_BA+0xA0	R	Timer2 PWM 周期缓存寄存器	0x0000_0000
TIMER2_PWM_CMPBUF	TMR23_BA+0xA4	R	Timer2 PWM 比较器缓存寄存器	0x0000_0000
TIMER3_CTL	TMR23_BA+0x100	R/W	Timer3 控制寄存器	0x0000_0005
TIMER3_CMP	TMR23_BA+0x104	R/W	Timer3比较寄存器	0x0000_0000
TIMER3_INTS_TS	TMR23_BA+0x108	R/W	Timer3中断状态寄存器	0x0000_0000
TIMER3_CNT	TMR23_BA+0x10C	R/W	Timer3数据寄存器	0x0000_0000
TIMER3_CAP	TMR23_BA+0x110	R	Timer3捕获数据寄存器	0x0000_0000
TIMER3_EXT_CTL	TMR23_BA+0x114	R/W	Timer3外部控制寄存器	0x0000_0000
TIMER3_EINT_STS	TMR23_BA+0x118	R/W	Timer3外部中断状态寄存器	0x0000_0000
TIMER3_TRG_CTL	TMR23_BA+0x11C	R/W	Timer3触发控制寄存器	0x0000_0000
TIMER3_ALT_CTL	TMR23_BA+0x120	R/W	Timer3选择控制寄存器	0x0000_0000
TIMER3_PWM_CTL	TMR23_BA+0x140	R/W	Timer3 PWM 控制寄存器	0x0000_0000
TIMER3_PWM_CLKSRC	TMR23_BA+0x144	R/W	Timer3 PWM 计数器时钟源寄存器	0x0000_0000
TIMER3_PWM_CLKPSC	TMR23_BA+0x148	R/W	Timer3 PWM 计数器时钟预分频寄存器	0x0000_0000
TIMER3_PWM_CNTCLR	TMR23_BA+0x14C	R/W	Timer3 PWM 清计数器寄存器	0x0000_0000

<b>TIMER3_PWM_PERIOD</b>	TMR23_BA+0x150	R/W	Timer3 PWM 周期寄存器	0x0000_0000
<b>TIMER3_PWM_CMPDAT</b>	TMR23_BA+0x154	R/W	Timer3 PWM 比较寄存器	0x0000_0000
<b>TIMER3_PWM_DTCTL</b>	TMR23_BA+0x158	R/W	Timer3 PWM 死区控制寄存器	0x0000_0000
<b>TIMER3_PWM_CNT</b>	TMR23_BA+0x15C	R	Timer3 PWM 计数器寄存器	0x0000_0000
<b>TIMER3_PWM_MSKEN</b>	TMR23_BA+0x160	R/W	Timer3 PWM 输出屏蔽使能寄存器	0x0000_0000
<b>TIMER3_PWM_MSK</b>	TMR23_BA+0x164	R/W	Timer3 PWM 输出屏蔽数据控制寄存器	0x0000_0000
<b>TIMER3_PWM_BNF</b>	TMR23_BA+0x168	R/W	Timer3 PWM 刹车管脚噪音滤波寄存器	0x0000_0000
<b>TIMER3_PWM_FAILBRK</b>	TMR23_BA+0x16C	R/W	Timer3 PWM 系统失败刹车控制寄存器	0x0000_0000
<b>TIMER3_PWM_BRKCTL</b>	TMR23_BA+0x170	R/W	Timer3 PWM 刹车控制寄存器	0x0000_0000
<b>TIMER3_PWM_POLCTL</b>	TMR23_BA+0x174	R/W	Timer3 PWM 管脚输出极性控制寄存器	0x0000_0000
<b>TIMER3_PWM_POEN</b>	TMR23_BA+0x178	R/W	Timer3 PWM 管脚输出使能寄存器	0x0000_0000
<b>TIMER3_PWM_SWBRK</b>	TMR23_BA+0x17C	W	Timer3 PWM 软件触发刹车控制 寄存器	0x0000_0000
<b>TIMER3_PWM_INTENO</b>	TMR23_BA+0x180	R/W	Timer3 PWM 中断使能寄存器 0	0x0000_0000
<b>TIMER3_PWM_INTEN1</b>	TMR23_BA+0x184	R/W	Timer3 PWM中断使能寄存器1	0x0000_0000
<b>TIMER3_PWM_INTSTS0</b>	TMR23_BA+0x188	R/W	Timer3 PWM 中断状态寄存器 0	0x0000_0000
<b>TIMER3_PWM_INTSTS1</b>	TMR23_BA+0x18C	R/W	Timer3 PWM中断状态寄存器1	0x0000_0000
<b>TIMER3_PWM_ADCTS</b>	TMR23_BA+0x190	R/W	Timer3 PWM ADC 触发源选择控制寄存器	0x0000_0000
<b>TIMER3_PWM_SCTL</b>	TMR23_BA+0x194	R/W	Timer3 PWM 同步控制寄存器	0x0000_0000
<b>TIMER3_PWM_SSTRG</b>	TMR23_BA+0x198	W	Timer3 PWM 同步触发寄存器	0x0000_0000
<b>TIMER3_PWM_STATUS</b>	TMR23_BA+0x19C	R/W	Timer3 PWM 状态寄存器	0x0000_0000
<b>TIMER3_PWM_PBUF</b>	TMR23_BA+0x1A0	R	Timer3 PWM 周期缓存寄存器	0x0000_0000
<b>TIMER3_PWM_CMPBUF</b>	TMR23_BA+0x1A4	R	Timer3 PWM 比较器缓存寄存器	0x0000_0000

## 6.7.8 寄存器描述

**TIMERx\_CTL** 定时器控制寄存器

寄存器	偏移地址	R/W	描述	复位值
TIMER0_CTL	TMR01_BA+0x00	R/W	Timer0 控制寄存器	0x0000_0005
TIMER1_CTL	TMR01_BA+0x100	R/W	Timer1控制寄存器	0x0000_0005
TIMER2_CTL	TMR23_BA+0x00	R/W	Timer2控制寄存器	0x0000_0005
TIMER3_CTL	TMR23_BA+0x100	R/W	Timer3控制寄存器	0x0000_0005

31	30	29	28	27	26	25	24
ICEDEBUG	CNTEN	INTEN	OPMODE		Reserved	ACTSTS	EXTCNTEN
23	22	21	20	19	18	17	16
WKEN	CAPSRC	TGLPINSEL	PERIOSEL	INTRGEN	Reserved		
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
PSC							

位	描述
[31]	<b>ICEDEBUG</b> 仿真器 (ICE) 调试模式响应禁止位（写保护位） 0 = 仿真器调试模式响应影响定时器计数。 当CPU被ICE停住时，定时器计数器将被固定住。 1 = 仿真器调试模式响应禁用。 无论CPU是否被ICE停住，定时器计数器将持续计数下去。 <b>注意：</b> 该位为写保护位。详情请参考SYS_REGLCTL 寄存器。
[30]	<b>CNTEN</b> 定时器计数使能位 0 = 停止/暂停计数 1 = 开始计数 <b>注意1：</b> 在停止状态，设置CNTEN为 1 将使能 24-位向上计数器从上次停止的计数值继续计数。 <b>注意2：</b> 在 one-shot 模式下(TIMER_CTL[28:27] = 00)，当相应的定时中断标志TIF (TIMERx_INTSTS[0])产生时，该位由硬件自动清零。 <b>注意3：</b> 设置使能/禁止该位需要2 * TMR_CLK个时钟周期生效。用户可以读ACTSTS (TIMERx_CTL[25])来检查使能/禁止命令是否完成。
[29]	<b>INTEN</b> 定时器中断使能位 0 = 禁用定时器超时中断 1 = 使能定时器超时中断 <b>注意：</b> 如果该位使能，当定时器中断标志TIF置 1，定时器中断信号产生并通知CPU。
[28:27]	<b>OPMODE</b> 定时器计数模式选择 00 = 定时器工作在单次触发模式 (one-shot)

		01 = 定时器工作在周期模式 (Periodic) 10 = 定时器工作在触发模式 (Toggle) 11 = 定时器工作在连续计数模式(Continuous Counting)
[26]	<b>Reserved</b>	保留.
[25]	<b>ACTSTS</b>	定时器激活状态位 (只读) 该位表示24位向上计数器的状态。 0 = 定时器计数器未激活 1 = 定时器计数器激活
[24]	<b>EXTCNTEN</b>	事件计数模式使能位 该位用来使能外部计数管脚功能。 0 = 禁用事件计数器模式 1 = 使能事件计数器模式 <b>注意:</b> 当定时器用作事件计数器, 该位需要设置成1.并选择PCLK作为定时器时钟源。
[23]	<b>WKEN</b>	唤醒功能使能位 如果该位置1, 当定时器中断标志TIF (TIMERx_INTSTS[0])被置1, 且INTEN (TIMERx_CTL[29])被使能, 定时器中断信号将产生一个唤醒触发事件给CPU 0 = 如果定时器中断信号产生, 唤醒功能禁止 1 = 如果定时器中断信号产生, 唤醒功能使能
[22]	<b>CAPSRC</b>	捕获管脚源选择 0 = 捕获功能源自TMx_EXT (x= 0~3) 管脚. 1 = 捕获功能源自内部ACMP输出信号, 用户可以设置ACMPSSEL (TIMERx_EXTCTL[8]) 来决定哪一个内部ACMP输出信号作为定时器捕获源。
[21]	<b>TGLPINSEL</b>	触发输出管脚选择 0 = 触发模式输出到TMx脚(定时器事件计数管脚). 1 = 触发模式输出到TMx_EXT (定时器外部捕获管脚).
[20]	<b>PERIOSEL</b>	周期模式行为选择使能位 0 = 在周期模式行为选择禁止 当定时器正在周期模式运行的时候, 用户更新CMPDAT, CNT会复位到默认值。 CNT会被复位到默认值 1 = 在周期模式行为选择使能 当定时器正在周期模式运行的时候, 用户更新CMPDAT有如下限制: 如果CMPDAT值大于CNT, CMPDAT会被更新且CNT保持继续计数。 如果CMPDAT值等于CNT, 定时器超时中断会立即产生。 如果CMPDAT值小于CNT, CNT会复位到默认值。
[19]	<b>INTRGEN</b>	定时器间互触发模式使能控制 设置该位会使能定时器间互触发捕获功能 定时器0/2会工作在事件计数模式, 计数外部时钟源或事件。 定时器1/3会工作在捕获功能的触发计数模式 0 = 定时器间互触发捕获模式禁止. 1 = 定时器间互触发捕获模式使能 <b>注意:</b> 对于 Timer1/3, 该位被忽略, 读回总是0。
[18:8]	<b>Reserved</b>	保留.

[7:0]	<b>PSC</b>	<b>预分频计数器</b> 定时器的输入时钟源或事件计数源被PSC +1预分频，然后再输入到定时器。如果 该位为0，表示不分频。 <b>注意：</b> 更新预分频计数器值会复位内部8位预分频计数器和24位上数计数器值
-------	------------	--

## TIMERx\_CMP 定时器比较寄存器

寄存器	偏移地址	R/W	描述	复位值
TIMER0_CMP	TMR01_BA+0x04	R/W	Timer0比较寄存器	0x0000_0000
TIMER1_CMP	TMR01_BA+0x104	R/W	Timer1比较寄存器	0x0000_0000
TIMER2_CMP	TMR23_BA+0x04	R/W	Timer2比较寄存器	0x0000_0000
TIMER3_CMP	TMR23_BA+0x104	R/W	Timer3比较寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
CMPDAT							
15	14	13	12	11	10	9	8
CMPDAT							
7	6	5	4	3	2	1	0
CMPDAT							

位	描述	
[31:24]	Reserved	保留.
[23:0]	CMPDAT	<p><b>定时器比较值</b>          CMPDAT是24位比较寄存器。当内部 24位向上计数器的值等于CMPDAT的值时，TIF (TIMERx_INTSTS[0] 定时器中断标志)将被置1。</p> <p>超时溢出周期 = (定时器输入时钟周期) * (8-bit PSC + 1) * (24-bit CMPDAT)</p> <p><b>注意1：</b>不能向CMPDAT里写 0x0 或 0x1，否则内核将运行到未知状态。</p> <p><b>注意2：</b>当定时器工作在 连续计数 模式，即使软件写一个新的值到CMPDAT，24位向上计数定时器将保持继续计数。如果定时器工作在其他模式，如果软件写一个新的值到CMPDAT，定时器将使用新比较值并退出当前计数，重新从0开始计数。</p>

**TIMERx\_INTSTS 定时器中断状态寄存器**

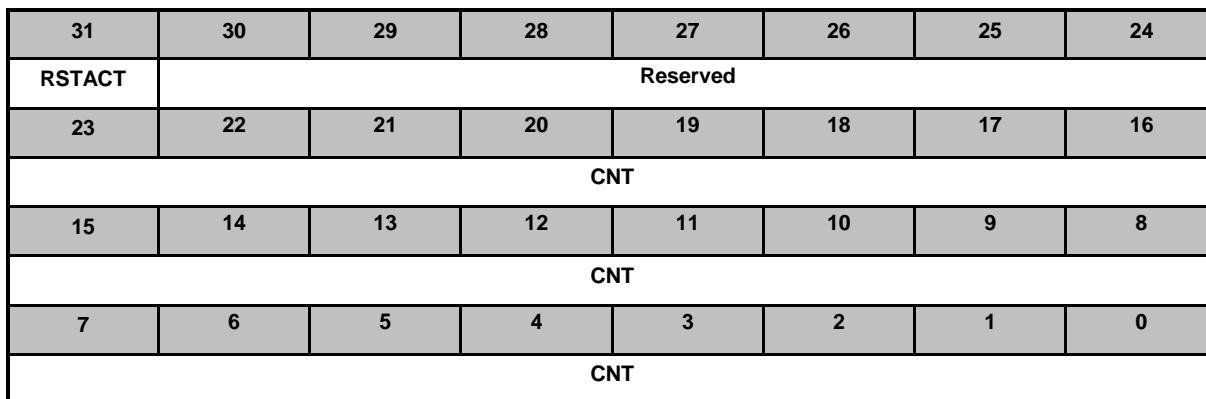
寄存器	偏移地址	R/W	描述	复位值
<b>TIMER0_INTSTS</b>	TMR01_BA+0x08	R/W	Timer0中断状态寄存器	0x0000_0000
<b>TIMER1_INTSTS</b>	TMR01_BA+0x108	R/W	Timer1中断状态寄存器	0x0000_0000
<b>TIMER2_INTSTS</b>	TMR23_BA+0x08	R/W	Timer2中断状态寄存器	0x0000_0000
<b>TIMER3_INTSTS</b>	TMR23_BA+0x108	R/W	Timer3中断状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved						TWKF	TIF

位	描述	
[31:2]	<b>Reserved</b>	保留.
[1]	<b>TWKF</b>	<p><b>定时器唤醒标志</b>            该位表示定时器的中断唤醒标志状态。            0 = 定时器不会引起CPU 唤醒            1 = 如果定时器超时中断信号产生， CPU 从空闲或掉电模式唤醒  <b>注意:</b> 该位必须通过软件写1清0</p>
[0]	<b>TIF</b>	<p><b>定时器中断标志</b>            当内部 24-位向上计数定时器CNT (TIMERx_CNT[23:0])的值与定时器比较值CMPDAT (TIMERx_CMP[23:0])匹配时，该位会显示其中断状态。.            0 = 无影响            1 = 计数定时器与CMPDAT的值相匹配  <b>注意:</b> 该位写 1 清零。</p>

**TIMERx\_CNT 定时器数据寄存器**

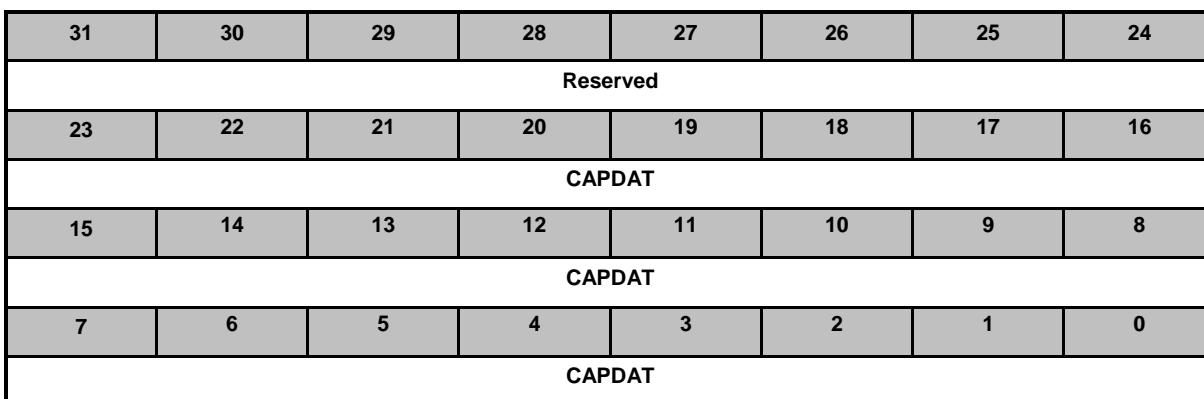
寄存器	偏移地址	R/W	描述	复位值
<b>TIMER0_CNT</b>	TMR01_BA+0x0C	R/W	Timer0数据寄存器	0x0000_0000
<b>TIMER1_CNT</b>	TMR01_BA+0x10C	R/W	Timer1数据寄存器	0x0000_0000
<b>TIMER2_CNT</b>	TMR23_BA+0x0C	R/W	Timer2数据寄存器	0x0000_0000
<b>TIMER3_CNT</b>	TMR23_BA+0x10C	R/W	Timer3数据寄存器	0x0000_0000



位	描述	
[31]	<b>RSTACT</b>	<b>定时器 数据寄存器复位生效 (只读)</b> 该位表示计数器复位操作是否生效 当用户写CNT寄存器，定时器开始复位内部24位定时器上数计数器到0，且重载8位预分频计数器，同时定时器置位该标志为1表示计数器复位操作正在进行。一旦计数器复位操作完成，该位自动清0。 0 = 复位操作已完成。 1 = 通过写TIMERx_CNT触发的复位操作正在进行
[30:24]	<b>Reserved</b>	保留。
[23:0]	<b>CNT</b>	<b>定时器数据寄存器</b> 读操作。 读该寄存器，获取 CNT 值，例如： 如果 EXTCNTEN (TIMERx_CTL[24] ) 是 0，用户可以读CNT的值以获取当前24位计数器值。 如果 EXTCNTEN (TIMERx_CTL[24] ) 是 1，用户可以读CNT的值以获取当前24位事件输入计数器值。 写操作。 写任何值到该寄存器会复位当前CNT值为0，且重载内部8位预分频计数器。

**TIMERx\_CAP 定时器捕获数据寄存器**

寄存器	偏移地址	R/W	描述	复位值
TIMER0_CAP	TMR01_BA+0x10	R	Timer0捕获数据寄存器	0x0000_0000
TIMER1_CAP	TMR01_BA+0x110	R	Timer1捕获数据寄存器	0x0000_0000
TIMER2_CAP	TMR23_BA+0x10	R	Timer2捕获数据寄存器	0x0000_0000
TIMER3_CAP	TMR23_BA+0x110	R	Timer3捕获数据寄存器	0x0000_0000



位	描述	
[31:24]	Reserved	保留.
[23:0]	CAPDAT	<b>定时器捕获数据寄存器</b> 当 CAPEN (TIMERx_EXTCTL[3]) 位被置1, CAPFUNCS (TIMERx_EXTCTL[4])位为0, TMx _EXT 管脚的变化与CAPEEDGE (TIMERx_EXTCTL[14:12])设定相匹配时, CAPIF (TIMERx_EINTSTS[0]) 将被置1且当前定时器计数器 CNT (TIMERx_CNT[23:0]) 的值将被自动载入到CAPDAT.

**TIMERx\_EXTCTL 定时器外部控制寄存器**

寄存器	偏移地址	R/W	描述	复位值
TIMER0_EXT_CTL	TMR01_BA+0x14	R/W	Timer0外部控制寄存器	0x0000_0000
TIMER1_EXT_CTL	TMR01_BA+0x114	R/W	Timer1外部控制寄存器	0x0000_0000
TIMER2_EXT_CTL	TMR23_BA+0x14	R/W	Timer2外部控制寄存器	0x0000_0000
TIMER3_EXT_CTL	TMR23_BA+0x114	R/W	Timer3外部控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved	CAPEDGE			Reserved			ACMPSEL
7	6	5	4	3	2	1	0
CNTDBEN	CAPDBEN	CAPIEN	CAPFUNCS	CAPEN	Reserved		CNTPHASE

位	描述	
[31:17]	Reserved	保留.
[16]	ECNTSSEL	事件计数器源选择，用于触发事件计数器功能 0 = 事件计数器输入源来自TMx (x= 0~3) 管脚. 1 = 事件计数器输入源来自 USB 内部 SOF 输出信号.
[15]	Reserved	保留.
[14:12]	CAPEDGE	定时器外部捕获管脚边沿侦测 当第一个捕获事件产生，CNT (TIMERx_CNT[23:0])会被复位为0，且第一个CAPDAT (TIMERx_CAP[23:0])应该是0。 000 = 当在TMx_EXT (x= 0~3)管脚侦测到下降沿，捕获事件发生。 001 = 当在TMx_EXT (x= 0~3)管脚侦测到上升沿，捕获事件发生。 010 = 当在TMx_EXT (x= 0~3)管脚侦测到下降沿和上升沿，捕获事件都发生。第一个捕获事件发生在下降沿。 011 = 当在TMx_EXT (x= 0~3)管脚侦测到下降沿和上升沿，捕获事件都发生。第一个捕获事件发生在上升沿。 110 = 第一个捕获事件发生在下降沿，接下来的捕获事件发生上升沿。 111 = 第一个捕获事件发生在上升沿，接下来的捕获事件发生下降沿。 100, 101 = 保留.
[11:9]	Reserved	保留.

[8]	<b>ACMPSSEL</b>	用于触发捕获功能的ACMP 源选择 0 = 捕获功能源来自内部 ACMP0 输出信号 1 = 捕获功能源来自内部 ACMP1 输出信号 <b>注意:</b> 仅当 CAPSRC (TIMERx_CTL[22]) 是 1, 这些位有效。.
[7]	<b>CNTDBEN</b>	定时器外部计数器输入管脚消抖使能位 0 = TMx (x= 0~3)管脚消抖禁用 1 = TMx (x= 0~3)管脚消抖使能 <b>注意:</b> 如果该位使能, TMx管脚边沿检测带消抖电路。
[6]	<b>CAPDBEN</b>	定时器外部捕获输入管脚消抖使能位 0 = TMx _EXT (x= 0~3)管脚或ACMP输出消抖禁用 1 = TMx _EXT (x= 0~3)管脚或ACMP输出消抖使能 <b>注意:</b> 如果该位使能, TMx _EXT管脚或ACMP输出边沿检测带消抖电路。
[5]	<b>CAPIEN</b>	定时器外部捕获中断使能位 0 = TMx _EXT (x= 0~3) 管脚检测中断禁止. 1 = TMx _EXT (x= 0~3)管脚检测中断使能 <b>注意:</b> CAPIEN 用于使能定时器外部中断。如果 CAPIEN 使能, 当 CAPIF (TIMERx_EINTSTS[0])标志设为1时, 定时器会产生一个外部捕获中断信号。 例如, 当 CAPIEN = 1, CAPEN = 1, 且 CAPEdge = 00, TMx _EXT 管脚上的由1变0变化, 将导致CAPIF 被置位, 然后中断信号产生并发送到NVIC并告知CPU.
[4]	<b>CAPFUNCS</b>	捕获功能选择 0 =外部捕获模式使能. 1 =外部复位模式使能. <b>注意1:</b> 当 CAPFUNCS为 0, TMx _EXT (x= 0~3) 管脚上的变化用于保存 24-位定时器计数值(CNT) 到 CAPDAT. <b>注意2:</b> 当 CAPFUNCS为 1, TMx _EXT (x= 0~3) 管脚上的变化用于复位复位24-位定时器计数器值(CNT) 到 CAPDAT, 然后CNT值立即复位。
[3]	<b>CAPEN</b>	定时器外部捕获管脚使能 该位使能TMx _EXT管脚捕获输入功能 0 = TMx _EXT (x= 0~3)管脚禁止。 1 = TMx _EXT (x= 0~3)管脚使能。
[2:1]	<b>Reserved</b>	保留.
[0]	<b>CNTPHASE</b>	定时器外部计数管脚相位检测选择 该位表示TMx (x= 0~3)管脚相位检测。 0 =管脚的下降沿将被统计。 1 =管脚的上升沿将被统计

**TIMERx\_EINTSTS 定时器外部中断状态寄存器**

寄存器	偏移量	R/W	描述	复位值
<b>TIMER0_EINTSTS</b>	TMR01_BA+0x18	R/W	Timer0外部中断状态寄存器	0x0000_0000
<b>TIMER1_EINTSTS</b>	TMR01_BA+0x118	R/W	Timer1外部中断状态寄存器	0x0000_0000
<b>TIMER2_EINTSTS</b>	TMR23_BA+0x18	R/W	Timer2外部中断状态寄存器	0x0000_0000
<b>TIMER3_EINTSTS</b>	TMR23_BA+0x118	R/W	Timer3外部中断状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved							<b>CAPIF</b>

位	描述	
[31:1]	<b>Reserved</b>	保留.
[0]	<b>CAPIF</b>	<p><b>定时器外部捕获中断标志位</b>          该位表示定时器外部捕获中断标志状态。.          0 = TMx_EXT (x= 0~3)管脚中断没有发生。          1 = TMx_EXT (x= 0~3)管脚中断发生。</p> <p><b>注意1:</b> 该位写 1 清零。</p> <p><b>注意2:</b> 当 CAPEN (TIMERx_EXTCTL[3]) 位被置1, CAPFUNCS (TIMERx_EXTCTL[4]) 位为 0, TMx_EXT (x= 0~3) 管脚的变化与CAPEdge (TIMERx_EXTCTL[2:1]) 设定匹配, 该位将被硬件置1。</p> <p><b>注意3:</b> CPU 清 CAPIF 状态前会检测到一个新的捕获事件.如果以上条件发生, 定时器将保持 TIMERx_CAP 的值不变并丢掉新的捕获值。</p>

**TIMERx\_TRGCTL 定时器 触发控制寄存器**

寄存器	偏移地址	R/W	描述	复位值
TIMER0_TRGCTL	TMR01_BA+0x1C	R/W	Timer0 触发控制寄存器	0x0000_0000
TIMER1_TRGCTL	TMR01_BA+0x11C	R/W	Timer1触发控制寄存器	0x0000_0000
TIMER2_TRGCTL	TMR23_BA+0x1C	R/W	Timer2触发控制寄存器	0x0000_0000
TIMER3_TRGCTL	TMR23_BA+0x11C	R/W	Timer3触发控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	3	4	3	2	1	0
Reserved			TRGPDMA	TRGDAC	TRGEADC	TRGPWM	TRGSSEL

位	描述	
[31:5]	Reserved	保留.
[4]	TRGPDMA	<b>触发 PDMA 使能位</b> 如果该位置1，每一个定时器超时事件或捕获事件都可以触发PDMA传输 0 = 定时器中断触发 PDMA 禁止. 1 = 定时器中断触发 PDMA 使能. <b>注意:</b> 如果 TRGSSEL (TIMERx_TRGCTL[0]) = 0, 超时中断信号会触发PDMA传输 如果 TRGSSEL (TIMERx_TRGCTL[0]) = 1, 捕获中断信号 PDMA 传输.
[3]	TRGDAC	<b>触发 DAC 使能位</b> 如果该位置1，每一个定时器超时事件或捕获事件都可以触发DAC 0 = 定时器中断触发DAC禁止. 1 = 定时器中断触发DAC使能. <b>注意:</b> 如果 TRGSSEL (TIMERx_TRGCTL[0]) = 0, 超时中断信号会触发DAC 如果 TRGSSEL (TIMERx_TRGCTL[0]) = 1, 捕获中断信号DAC
[2]	TRGEADC	<b>触发 ADC 使能位</b> 如果该位置1，每一个定时器超时事件或捕获事件都可以触发ADC转换 0 = 定时器中断触发EADC禁止. 1 = 定时器中断触发EADC使能. <b>注意:</b> 如果 TRGSSEL (TIMERx_TRGCTL[0]) = 0, 超时中断信号会触发EADC转换

		如果 TRGSSEL (TIMERx_TRGCTL[0]) = 1, 捕获中断信号ADC转换
[1]	TRGPWM	<b>触发PWM使能位</b> 如果该位置1, 每一个定时器超时事件或捕获事件都可以作为EPWM计数器时钟源 0 = 定时器中断触发EPWM禁止. 1 = 定时器中断触发EPWM使能. <b>注意:</b> 如果 TRGSSEL (TIMERx_TRGCTL[0]) = 0, 超时中断信号作为EPWM计数器时钟源 如果 TRGSSEL (TIMERx_TRGCTL[0]) = 1, 捕获中断信号作为PWM计数器时钟源
[0]	TRGSSEL	<b>触发源选择位</b> 该位用于选择内部触发源来自定时器超时中断信号或捕获中断信号 0 = 超时中断信号用于内部触发PWM, PDMA, 和 ADC. 1 = 捕获中断信号用于内部触发 PWM, PDMA, 和 ADC.

**TIMERx ALTCTL 定时器选择控制寄存器**

寄存器	偏移地址	R/W	描述	复位值
TIMER0_ALT_CTL	TMR01_BA+0x20	R/W	Timer0 选择控制寄存器	0x0000_0000
TIMER1_ALT_CTL	TMR01_BA+0x120	R/W	Timer1选择控制寄存器	0x0000_0000
TIMER2_ALT_CTL	TMR23_BA+0x20	R/W	Timer2选择控制寄存器	0x0000_0000
TIMER3_ALT_CTL	TMR23_BA+0x120	R/W	Timer3选择控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved							FUNCSEL

位	描述	
[31:1]	Reserved	保留.
[0]	FUNCSEL	<p>功能选择 0 = 定时器控制器用于定时器功能 1 = 定时器控制器用于PWM功能.</p> <p><b>注意:</b> 当定时器用作 PWM，定时器控制器时钟源会被强制自动选为PCLKx。</p>

**TIMERx\_PWMCTL 定时器 PWM 控制寄存器**

寄存器	偏移地址	R/W	描述	复位值
<b>TIMER0_PWM_CTL</b>	TMR01_BA+0x40	R/W	Timer0 PWM 控制寄存器	0x0000_0000
<b>TIMER1_PWM_CTL</b>	TMR01_BA+0x140	R/W	Timer1 PWM控制寄存器	0x0000_0000
<b>TIMER2_PWM_CTL</b>	TMR23_BA+0x40	R/W	Timer2 PWM控制寄存器	0x0000_0000
<b>TIMER3_PWM_CTL</b>	TMR23_BA+0x140	R/W	Timer3 PWM控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
<b>DBGTRIOFF</b>	<b>DBGHALT</b>	Reserved					
23	22	21	20	19	18	17	16
Reserved							<b>OUTMODE</b>
15	14	13	12	11	10	9	8
Reserved						<b>IMMLDEN</b>	<b>CTRLD</b>
7	6	5	4	3	2	1	0
Reserved				<b>CNTMODE</b>	<b>CNTTYPE</b>		<b>CNTEN</b>

位	描述
[31]	<b>DBGTRIOFF</b> 仿真器 (ICE) 调试模式响应禁止位（写保护位） 0 = 仿真器调试模式影响PWM输出。 当ICE调试模式响应时，PWM输出管脚被强制为三态模式。 1 = 仿真器调试模式响应禁用。 无论ICE调试模式响应与否，PWM输出管脚保持输出。 <b>注意：</b> 该位为写保护位。详情请参考SYS_REGLCTL 寄存器.
[30]	<b>DBGHALT</b> <b>ICE调试模式计数器暂停 (写保护)</b> 如果调试模式计数器暂停使能，PWM计数器会保持当前值直到退出ICE调试模式。 0 = ICE调试模式计数器暂停禁止。 1 = ICE 调试模式计数器暂停使能 <b>注意：</b> 该位为写保护位。详情请参考SYS_REGLCTL 寄存器.
[29:17]	<b>Reserved</b> 保留.
[16]	<b>OUTMODE</b> <b>PWM 输出模式</b> 该位控制对应PWM通道输出模式 0 = PWM 独立模式 1 = PWM 互补模式.
[15:10]	<b>Reserved</b> 保留.
[9]	<b>IMMLDEN</b> 立即加载使能位

		<p>0 = 当当前PWM周期完成，无论CTRLD是否使能，PERIOD加载到PBUF。如果CTRLD禁止，当当前PWM周期完成，CMP会加载到CMPBUF。如果CTRLD使能，在上下数计数模式，CMP会在当前周期的中心点加载到CMPBUF。</p> <p>1 = 当用户更新PERIOD/CMP.，PERIOD/CMP 会立即加载到 PBUF/CMPBUF</p> <p><b>注意:</b> 如果 IMMLDEN 使能, CTRLD 无效.</p>
[8]	<b>CTRLD</b>	<p><b>中心重载</b></p> <p>在上下数计数模式，当当前PWM周期完成，PERIOD会加载到PBUF， CMP会在当前周期的中心点加载到CMPBUF。</p>
[7:4]	<b>Reserved</b>	保留.
[3]	<b>CNTMODE</b>	<p><b>PWM 计数器模式</b></p> <p>0 = 自动重载模式.</p> <p>1 = 单周期模式</p>
[2:1]	<b>CNTTYPE</b>	<p><b>PWM 计数模式</b></p> <p>00 = 上计数模式.</p> <p>01 = 下计数模式</p> <p>10 = 上下计数模式</p> <p>11 = 保留.</p>
[0]	<b>CNTEN</b>	<p><b>PWM 计数器使能位</b></p> <p>0 = PWM 计数器和时钟预分频停止运行.</p> <p>1 = PWM 计数器和时钟预分频开始运行</p>

**TIMERx PWMCLKSRC 定时器 PWM 计数器时钟源寄存器**

寄存器	偏移地址	R/W	描述	复位值
TIMER0_PWM_CLKSRC	TMR01_BA+0x44	R/W	Timer0 PWM 计数器时钟源寄存器	0x0000_0000
TIMER1_PWM_CLKSRC	TMR01_BA+0x144	R/W	Timer1 PWM计数器时钟源寄存器	0x0000_0000
TIMER2_PWM_CLKSRC	TMR23_BA+0x44	R/W	Timer2 PWM计数器时钟源寄存器	0x0000_0000
TIMER3_PWM_CLKSRC	TMR23_BA+0x144	R/W	Timer3 PWM计数器时钟源寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved				CLKSRC			

位	描述	
[31:3]	Reserved	保留.
[2:0]	CLKSRC	<p><b>PWM 计数器时钟源选择</b></p> <p>PWM计数器时钟源可以选择来自TMRx_CLK或内部定时器超时或捕获事件</p> <p>000 = TMRx_CLK.</p> <p>001 = 内部 TIMER0 超时或捕获事件.</p> <p>010 = 内部 TIMER1超时或捕获事件.</p> <p>011 = 内部TIMER2超时或捕获事件.</p> <p>100 = 内部TIMER3超时或捕获事件.</p> <p>其他 = 保留.</p>

**TIMERx PWMCLKPSC 定时器 PWM 计数器时钟预分频寄存器**

寄存器	偏移地址	R/W	描述	复位值
<b>TIMER0_PWM_CLKPSC</b>	TMR01_BA+0x48	R/W	Timer0 PWM 计数器时钟预分频寄存器	0x0000_0000
<b>TIMER1_PWM_CLKPSC</b>	TMR01_BA+0x148	R/W	Timer1 PWM计数器时钟预分频寄存器	0x0000_0000
<b>TIMER2_PWM_CLKPSC</b>	TMR23_BA+0x48	R/W	Timer2 PWM计数器时钟预分频寄存器	0x0000_0000
<b>TIMER3_PWM_CLKPSC</b>	TMR23_BA+0x148	R/W	Timer3 PWM计数器时钟预分频寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved				CLKPSC			
7	6	5	4	3	2	1	0
CLKPSC							

位	描述	
[31:12]	<b>Reserved</b>	保留.
[11:0]	<b>CLKPSC</b>	<b>PWM 计数器时钟预分频</b> PWM计数器的实际时钟由计数器时钟预分频决定，时钟被 $(\text{CLKPSC} + 1)$ 分频。如果 CLKPSC 是 0，PWM计数器时钟不分频

**TIMERx PWM CNTCLR 定时器 PWM 清计数器寄存器**

寄存器	偏移地址	R/W	描述	复位值
<b>TIMER0_PWM_CNTCLR</b>	TMR01_BA+0x4C	R/W	Timer0 PWM 清计数器寄存器	0x0000_0000
<b>TIMER1_PWM_CNTCLR</b>	TMR01_BA+0x14C	R/W	Timer1 PWM清计数器寄存器	0x0000_0000
<b>TIMER2_PWM_CNTCLR</b>	TMR23_BA+0x4C	R/W	Timer2 PWM清计数器寄存器	0x0000_0000
<b>TIMER3_PWM_CNTCLR</b>	TMR23_BA+0x14C	R/W	Timer3 PWM清计数器寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved							CNTCLR

位	描述	
[31:1]	Reserved	保留.
[0]	CNTCLR	<b>清 PWM 计数器控制位</b> 该位由硬件自动清除 0 = 无效。 1 = 在上数和上下数计数模式，清 16-bit PWM 计数器为 0x10000。在下数计数模式复位计数器值为PERIOD。

**TIMERx\_PWMPERIOD 定时器 PWM 周期寄存器**

寄存器	偏移地址	R/W	描述	复位值
<b>TIMER0_PWM_PERIOD</b>	TMR01_BA+0x50	R/W	Timer0 PWM 周期寄存器	0x0000_0000
<b>TIMER1_PWM_PERIOD</b>	TMR01_BA+0x150	R/W	Timer1 PWM周期寄存器	0x0000_0000
<b>TIMER2_PWM_PERIOD</b>	TMR23_BA+0x50	R/W	Timer2 PWM周期寄存器	0x0000_0000
<b>TIMER3_PWM_PERIOD</b>	TMR23_BA+0x150	R/W	Timer3 PWM周期寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
PERIOD							
7	6	5	4	3	2	1	0
PERIOD							

位	描述	
[31:16]	Reserved	保留.
[15:0]	PERIOD	<p><b>PWM 周期寄存器</b></p> <p>在上计数模式, PWM计数器从0计数到PERIOD, 又从0重新开始。</p> <p>在下计数模式, PWM计数器从 PERIOD计数到0, 又从PERIOD重新开始。</p> <p>在上下计数模式, PWM计数器从0计数到PERIOD, 然后再递减到0, 如此反复。</p> <p>在上计数和下计数模式</p> <p>PWM 周期时间 = (PERIOD + 1) * (CLKPSC + 1) * TMRx_PWMCLK.</p> <p>在上下计数模式:</p> <p>PWM 周期时间 = 2 * PERIOD * (CLKPSC+ 1) * TMRx_PWMCLK.</p> <p><b>注意:</b> 在各种计数模式中, 用户应该关注DIRF (TIMERx_PWMCNT[16])位来监视当前计数器的方向。</p>

TIMERx PWMCMPPDAT 定时器 PWM 比较寄存器

寄存器	偏移地址	R/W	描述	复位值
<b>TIMER0_PWM_CMPDAT</b>	TMR01_BA+0x54	R/W	Timer0 PWM 比较寄存器	0x0000_0000
<b>TIMER1_PWM_CMPDAT</b>	TMR01_BA+0x154	R/W	Timer1 PWM比较寄存器	0x0000_0000
<b>TIMER2_PWM_CMPDAT</b>	TMR23_BA+0x54	R/W	Timer2 PWM比较寄存器	0x0000_0000
<b>TIMER3_PWM_CMPDAT</b>	TMR23_BA+0x154	R/W	Timer3 PWM比较寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
CMP							
7	6	5	4	3	2	1	0
CMP							

位	描述	
[31:16]	<b>Reserved</b>	保留.
[15:0]	<b>CMP</b>	<b>PWM 比较寄存器</b> PWM CMP用于与PWM CNT比较产生PWM 输出波形、中断事件和触发ADC开始转换。

**TIMERx PWMDTCTL 定时器 PWM 死区控制寄存器**

寄存器	偏移地址	R/W	描述	复位值
<b>TIMER0_PWM_DTCCTL</b>	TMR01_BA+0x58	R/W	Timer0 PWM 死区控制寄存器	0x0000_0000
<b>TIMER1_PWM_DTCCTL</b>	TMR01_BA+0x158	R/W	Timer1 PWM死区控制寄存器	0x0000_0000
<b>TIMER2_PWM_DTCCTL</b>	TMR23_BA+0x58	R/W	Timer2 PWM死区控制寄存器	0x0000_0000
<b>TIMER3_PWM_DTCCTL</b>	TMR23_BA+0x158	R/W	Timer3 PWM死区控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							<b>DTCKSEL</b>
23	22	21	20	19	18	17	16
Reserved							<b>DTEN</b>
15	14	13	12	11	10	9	8
Reserved				<b>DTCNT</b>			
7	6	5	4	3	2	1	0
<b>DTCNT</b>							

位	描述	
[31:25]	<b>Reserved</b>	保留.
[24]	<b>DTCKSEL</b>	<b>死区时钟选择 (写保护)</b> 0 = 死区时钟源来自TMRx_PWMCLK， 不带计数器时钟分频 1 = 死区时钟源来自TMRx_PWMCLK， 带计数器时钟分频 <b>注意:</b> 该位为写保护位。详情请参考SYS_REGLCTL 寄存器.
[23:17]	<b>Reserved</b>	保留.
[16]	<b>DTEN</b>	<b>对 PWMx_CH0 和 PWMx_CH1使能死区插入 (写保护)</b> 仅当PWM互补模式使能，死区插入功能才有效。如果死区插入无效，PWMx_CH0 和 PWMx_CH1是不带任何延时的互补输出。 0 = 死区插入禁止. 1 = 死区插入使能. <b>注意:</b> 该位为写保护位。详情请参考SYS_REGLCTL 寄存器.
[15:12]	<b>Reserved</b>	保留.
[11:0]	<b>DTCNT</b>	<b>死区计数器 (写保护)</b> 死区时间可以通过下面两个公式计算 死区时间= (DTCNT[11:0] + 1) * TMRx_PWMCLK (如果 DTCKSEL 是 0) . 死区时间= (DTCNT[11:0] + 1) * TMRx_PWMCLK * (CLKPSC + 1) (如果 DTCKSEL 是 1.) <b>注意:</b> 该位为写保护位。详情请参考SYS_REGLCTL 寄存器.

**TIMERx PWM CNT 定时器 PWM 计数器寄存器**

寄存器	偏移地址	R/W	描述	复位值
<b>TIMER0_PWM_CNT</b>	TMR01_BA+0x5C	R	Timer0 PWM 计数器寄存器	0x0000_0000
<b>TIMER1_PWM_CNT</b>	TMR01_BA+0x15C	R	Timer1 PWM计数器寄存器	0x0000_0000
<b>TIMER2_PWM_CNT</b>	TMR23_BA+0x5C	R	Timer2 PWM计数器寄存器	0x0000_0000
<b>TIMER3_PWM_CNT</b>	TMR23_BA+0x15C	R	Timer3 PWM计数器寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
CNT							
7	6	5	4	3	2	1	0
CNT							

位	描述	
[31:17]	<b>Reserved</b>	保留.
[16]	<b>DIRF</b>	<b>PWM 计数器方向指示标志 (只读)</b> 0 = 计数器在下计数方向 1 = 计数器在上计数方向
[15:0]	<b>CNT</b>	<b>PWM 计数器值寄存器 (只读)</b> 用户可以读 CNT以知道当前计数器值是多少

**TIMERx\_PWMMSKEN 定时器 PWM 输出屏蔽使能寄存器**

寄存器	偏移地址	R/W	描述	复位值
<b>TIMER0_PWM_MSKEN</b>	TMR01_BA+0x60	R/W	Timer0 PWM 输出屏蔽使能寄存器	0x0000_0000
<b>TIMER1_PWM_MSKEN</b>	TMR01_BA+0x160	R/W	Timer1 PWM输出屏蔽使能寄存器	0x0000_0000
<b>TIMER2_PWM_MSKEN</b>	TMR23_BA+0x60	R/W	Timer2 PWM输出屏蔽使能寄存器	0x0000_0000
<b>TIMER3_PWM_MSKEN</b>	TMR23_BA+0x160	R/W	Timer3 PWM输出屏蔽使能寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved						<b>MSKEN1</b>	<b>MSKEN0</b>

位	描述	
[31:2]	<b>Reserved</b>	保留.
[1]	<b>MSKEN1</b>	<p><b>PWMx_CH1 输出屏蔽使能位</b>  当该位使能， PWMx_CH1 输出信号被屏蔽， PWMx_CH1 输出 MSKDAT1 (TIMER_PWMMSK[1]) 数据。.  0 = PWMx_CH1输出信号不屏蔽  1 = PWMx_CH1 输出信号屏蔽且输出MSKDAT1 数据.</p>
[0]	<b>MSKEN0</b>	<p><b>PWMx_CH0 输出屏蔽使能位</b>  当该位使能， PWMx_CH0 输出信号被屏蔽， PWMx_CH0 输出 MSKDAT0 (TIMER_PWMMSK[0]) 数据。.  0 = PWMx_CH0输出信号不屏蔽  1 = PWMx_CH0 输出信号屏蔽且输出MSKDAT0 数据.</p>

**TIMERx PWMMSK 定时器 PWM 输出屏蔽数据控制寄存器**

寄存器	偏移地址	R/W	描述	复位值
<b>TIMER0_PWM_MSK</b>	TMR01_BA+0x64	R/W	Timer0 PWM Output Mask Data Control Register	0x0000_0000
<b>TIMER1_PWM_MSK</b>	TMR01_BA+0x164	R/W	Timer1 PWM Output Mask Data Control Register	0x0000_0000
<b>TIMER2_PWM_MSK</b>	TMR23_BA+0x64	R/W	Timer2 PWM Output Mask Data Control Register	0x0000_0000
<b>TIMER3_PWM_MSK</b>	TMR23_BA+0x164	R/W	Timer3 PWM Output Mask Data Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved						<b>MSKDAT1</b>	<b>MSKDAT0</b>

位	描述	
[31:2]	<b>Reserved</b>	保留.
[1]	<b>MSKDAT1</b>	<b>PWMx_CH1 输出屏蔽数据控制位</b> 当PWMx_CH1输出屏蔽功能使能(MSKEN1 = 1), 该位用于控制PWMx_CH1管脚的输出状态 0 = PWMx_CH1输出逻辑低 1 = PWMx_CH1输出逻辑高
[0]	<b>MSKDAT0</b>	<b>PWMx_CH0输出屏蔽数据控制位</b> 当PWMx_CH0输出屏蔽功能使能(MSKEN0 = 1), 该位用于控制PWMx_CH0管脚的输出状态 0 = PWMx_CH0输出逻辑低 1 = PWMx_CH0输出逻辑高

**TIMERx PWMBNF 定时器 PWM 刹车管脚噪音滤波寄存器**

寄存器	偏移地址	R/W	描述	复位值
<b>TIMER0_PWM_BNF</b>	TMR01_BA+0x68	R/W	Timer0 PWM 刹车管脚噪音滤波寄存器	0x0000_0000
<b>TIMER1_PWM_BNF</b>	TMR01_BA+0x168	R/W	Timer1 PWM刹车管脚噪音滤波寄存器	0x0000_0000
<b>TIMER2_PWM_BNF</b>	TMR23_BA+0x68	R/W	Timer2 PWM刹车管脚噪音滤波寄存器	0x0000_0000
<b>TIMER3_PWM_BNF</b>	TMR23_BA+0x168	R/W	Timer3 PWM刹车管脚噪音滤波寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved						BKPINSRC	
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
BRKPINV	BRKFCNT			BRKNFSEL			BRKNFEN

位	描述	
[31:18]	<b>Reserved</b>	保留.
[17:16]	<b>BKPINSRC</b>	<p><b>刹车管脚源选择</b></p> <p>00 =刹车管脚源来自 PWM0_BRAKE0 管脚.            01 =刹车管脚源来自 PWM0_BRAKE1 管脚..            10 =刹车管脚源来自 PWM1_BRAKE0 管脚..            11 =刹车管脚源来自 PWM1_BRAKE1 管脚..</p>
[15:8]	<b>Reserved</b>	保留.
[7]	<b>BRKPINV</b>	<p><b>刹车管脚侦测控制位</b></p> <p>0 =在边沿侦测模式， PWMx_BRAKEy管脚状态从低到高跳变，或在电平侦测模式，管脚状态是高，则刹车管脚事件会被侦测到。</p> <p>1 =在边沿侦测模式， PWMx_BRAKEy管脚状态从高到低跳变，或在电平侦测模式，管脚状态是低，则刹车管脚事件会被侦测到。.</p>
[6:4]	<b>BRKFCNT</b>	<p><b>刹车管脚噪音滤波计数</b></p> <p>该域用于控制噪音滤波采样时间            一次噪音滤波采样时间 = (BRKDBCS的周期时间) * BRKFCNT.</p>
[3:1]	<b>BRKNFSEL</b>	<p><b>刹车管脚噪音滤波时钟选择</b></p> <p>000 = 噪音滤波时钟是 PCLKx.            001 =噪音滤波时钟是PCLKx/2.            010 =噪音滤波时钟是PCLKx/4.</p>

		011 =噪音滤波时钟是PCLKx/8. 100 =噪音滤波时钟是PCLKx/16. 101 =噪音滤波时钟是PCLKx/32. 110 =噪音滤波时钟是PCLKx/64. 111 =噪音滤波时钟是PCLKx/128.
[0]	<b>BRKNFEN</b>	刹车管脚噪音滤波使能位 0 = PWMx_BRAKEy 管脚噪音滤波侦测禁止. 1 = PWMx_BRAKEy 管脚噪音滤波侦测使能

**TIMERx PWMFAILBRK 定时器PWM 系统失败刹车控制寄存器**

寄存器	偏移地址	R/W	描述	复位值
TIMER0_PWMFAILBRK	TMR01_BA+0x6C	R/W	Timer0 PWM 系统失败刹车控制寄存器	0x0000_0000
TIMER1_PWMFAILBRK	TMR01_BA+0x16C	R/W	Timer1 PWM系统失败刹车控制寄存器	0x0000_0000
TIMER2_PWMFAILBRK	TMR23_BA+0x6C	R/W	Timer2 PWM系统失败刹车控制寄存器	0x0000_0000
TIMER3_PWMFAILBRK	TMR23_BA+0x16C	R/W	Timer3 PWM系统失败刹车控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	2	4	3	2	1	0
Reserved				CORBRKEN	RAMBRKEN	BODBRKEN	CSSBRKEN

位	描述	
[31:4]	Reserved	保留.
[3]	CORBRKEN	内核锁死侦测触发PWM刹车功能使能位 0 = 禁止刹车功能由内核锁死事件触发 1 = 使能刹车功能由内核锁死事件触发
[2]	RAMBRKEN	SRAM 校验错误侦测触发PWM刹车功能使能位 0 = 禁止刹车功能由SRAM校验错误侦测触发 1 = 使能刹车功能由SRAM校验错误侦测触发
[1]	BODBRKEN	欠压侦测触发PWM刹车功能使能位 0 = 禁止刹车功能由BOD事件触发 1 = 使能刹车功能由BOD事件触发
[0]	CSSBRKEN	时钟安全系统侦测侦测触发PWM刹车功能使能位 0 = 禁止刹车功能由时钟失败侦测触发 1 = 使能刹车功能由时钟失败侦测触发

**TIMERx PWM\_BRKCTL 定时器 PWM 刹车控制寄存器**

寄存器	偏移地址	R/W	描述	复位值
TIMER0_PWM_BRKCTL	TMR01_BA+0x70	R/W	Timer0 PWM 刹车控制寄存器	0x0000_0000
TIMER1_PWM_BRKCTL	TMR01_BA+0x170	R/W	Timer1 PWM 刹车控制寄存器	0x0000_0000
TIMER2_PWM_BRKCTL	TMR23_BA+0x70	R/W	Timer2 PWM 刹车控制寄存器	0x0000_0000
TIMER3_PWM_BRKCTL	TMR23_BA+0x170	R/W	Timer3 PWM 刹车控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved				BRKAODD		BRKAEVEN	
15	14	13	12	11	10	9	8
SYSLBEN	Reserved		BRKPLEN	Reserved		CPO1LBEN	CPO0LBEN
7	6	5	4	3	2	1	0
SYSEBEN	Reserved		BRKPEEN	Reserved		CPO1EBEN	CPO0EBEN

位	描述	
[31:20]	Reserved	保留.
[19:18]	BRKAODD	<b>PWMx_CH1的PWM刹车动作选择 (写保护)</b> 00 = PWMx_BRAKEy 刹车事件不会影响 PWMx_CH1 输出。 01 = 当 PWMx_BRAKEy 刹车事件发生, PWMx_CH1 输出三态。 10 = 当 PWMx_BRAKEy 刹车事件发生, PWMx_CH1 输出低。 11 = 当 PWMx_BRAKEy 刹车事件发生, PWMx_CH1 输出高。 <b>注意:</b> 该位为写保护位。详情请参考SYS_REGLCTL 寄存器..
[17:16]	BRKAEVEN	<b>PWMx_CH0的PWM刹车动作选择 (写保护)</b> 00 = PWMx_BRAKEy 刹车事件不会影响 PWMx_CH0 输出。 01 = 当 PWMx_BRAKEy 刹车事件发生, PWMx_CH0 输出三态。 10 = 当 PWMx_BRAKEy 刹车事件发生, PWMx_CH0 输出低。 11 = 当 PWMx_BRAKEy 刹车事件发生, PWMx_CH0 输出高。 <b>注意:</b> 该位为写保护位。详情请参考SYS_REGLCTL 寄存器..
[15]	SYSLBEN	<b>使能系统失败作为电平侦测刹车源 (写保护)</b> 0 = 系统失败条件作为电平侦测刹车源禁止 1 = 系统失败条件作为电平侦测刹车源使能。 <b>注意:</b> 该位为写保护位。详情请参考SYS_REGLCTL 寄存器.
[14:13]	Reserved	保留.

[12]	<b>BRKPLEN</b>	使能 TM_BRAKE <sub>x</sub> 管脚作为电平侦测刹车源 (写保护) 0 = PWM <sub>x</sub> _BRAKE <sub>y</sub> 管脚事件作为电平侦测刹车源禁止 1 = PWM <sub>x</sub> _BRAKE <sub>y</sub> 管脚事件作为电平侦测刹车源使能. <b>注意:</b> 该位为写保护位。详情请参考SYS_REGLCTL 寄存器..
[11:10]	<b>Reserved</b>	保留.
[9]	<b>CPO1LBEN</b>	使能内部 ACMP1_O 数字输出作为电平侦测刹车源(写保护) 0 = 内部 ACMP1_O 信号作为电平侦测刹车源禁止 . 1 = 内部 ACMP1_O 信号 作为电平侦测刹车源使能. <b>注意1:</b> 作为刹车事件仅内部 ACMP1_O 信号从低到高会被侦测到 <b>注意2:</b> 该位为写保护位。详情请参考SYS_REGLCTL 寄存器.
[8]	<b>CPO0LBEN</b>	使能内部 ACMP0_O 数字输出作为电平侦测刹车源(写保护) 0 = 内部 ACMP0_O 信号作为电平侦测刹车源禁止 . 1 = 内部 ACMP0_O 信号 作为电平侦测刹车源使能. <b>注意1:</b> 作为刹车事件仅内部 ACMP0_O 信号从低到高会被侦测到 <b>注意2:</b> 该位为写保护位。详情请参考SYS_REGLCTL 寄存器.
[7]	<b>SYSEBEN</b>	使能系统失败作为边沿侦测刹车源(写保护) 0 = 系统失败条件作为边沿侦测刹车源禁止 1 = 系统失败条件作为边沿侦测刹车源使能 <b>注意:</b> 该位为写保护位。详情请参考SYS_REGLCTL 寄存器..
[6:5]	<b>Reserved</b>	保留.
[4]	<b>BRKPEEN</b>	使能 TM_BRAKE <sub>x</sub> 管脚作为边沿侦测刹车源 (写保护) 0 = PWM <sub>x</sub> _BRAKE <sub>y</sub> 管脚事件作为边沿侦测刹车源禁止 1 = PWM <sub>x</sub> _BRAKE <sub>y</sub> 管脚事件作为边沿侦测刹车源使能 <b>注意:</b> 该位为写保护位。详情请参考SYS_REGLCTL 寄存器.
[3:2]	<b>Reserved</b>	保留.
[1]	<b>CPO1EBEN</b>	使能内部 ACMP1_O 数字输出作为边沿侦测刹车源(写保护) 0 = 内部 ACMP1_O 信号作为边沿侦测刹车源禁止 . 1 = 内部 ACMP1_O 信号 作为边沿侦测刹车源使能. <b>注意1:</b> 作为刹车事件仅内部 ACMP1_O 信号从低到高会被侦测到 <b>注意2:</b> 该位为写保护位。详情请参考SYS_REGLCTL 寄存器...
[0]	<b>CPO0EBEN</b>	使能内部 ACMP0_O 数字输出作为边沿侦测刹车源(写保护) 0 = 内部 ACMP0_O 信号作为边沿侦测刹车源禁止 . 1 = 内部 ACMP0_O 信号 作为边沿侦测刹车源使能. <b>注意1:</b> 作为刹车事件仅内部 ACMP0_O 信号从低到高会被侦测到 <b>注意2:</b> 该位为写保护位。详情请参考SYS_REGLCTL 寄存器...

**TIMERx PWM POLCTL 定时器 PWM 管脚输出极性控制寄存器**

寄存器	偏移地址	R/W	描述	复位值
<b>TIMER0_PWM POLCTL</b>	TMR01_BA+0x74	R/W	Timer0 PWM 管脚输出极性控制寄存器	0x0000_0000
<b>TIMER1_PWM POLCTL</b>	TMR01_BA+0x174	R/W	Timer1 PWM管脚输出极性控制寄存器	0x0000_0000
<b>TIMER2_PWM POLCTL</b>	TMR23_BA+0x74	R/W	Timer2 PWM管脚输出极性控制寄存器	0x0000_0000
<b>TIMER3_PWM POLCTL</b>	TMR23_BA+0x174	R/W	Timer3 PWM管脚输出极性控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved						PINV1	PINV0

位	描述	
[31:2]	Reserved	保留.
[1]	PINV1	<b>PWMx_CH1 输出管脚极性控制位</b> 该位用于控制PWMx_CH1输出的极性状态 0 = PWMx_CH1输出管脚极性反转禁止 1 = PWMx_CH1输出管脚极性反转使能.
[0]	PINV0	<b>PWMx_CH0 输出管脚极性控制位</b> 该位用于控制PWMx_CH0输出的极性状态 0 = PWMx_CH0输出管脚极性反转禁止 1 = PWMx_CH0输出管脚极性反转使能.

**TIMERx PWMPOEN 定时器 PWM 管脚输出使能寄存器**

寄存器	偏移地址	R/W	描述	复位值
<b>TIMER0_PWM POEN</b>	TMR01_BA+0x78	R/W	Timer0 PWM 管脚输出使能寄存器	0x0000_0000
<b>TIMER1_PWM POEN</b>	TMR01_BA+0x178	R/W	Timer1 PWM管脚输出使能寄存器	0x0000_0000
<b>TIMER2_PWM POEN</b>	TMR23_BA+0x78	R/W	Timer2 PWM管脚输出使能寄存器	0x0000_0000
<b>TIMER3_PWM POEN</b>	TMR23_BA+0x178	R/W	Timer3 PWM管脚输出使能寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved						POEN1	POENO

位	描述	
[31:2]	Reserved	保留.
[1]	POEN1	<b>PWMx_CH1 输出管脚使能位</b> 0 = PWMx_CH1 管脚在三态模式. 1 = PWMx_CH1 管脚在输出模式.
[0]	POENO	<b>PWMx_CH0 输出管脚使能位</b> 0 = PWMx_CH0 管脚在三态模式. 1 = PWMx_CH0 管脚在输出模式.

**TIMERx\_PWM\_SWBRK 定时器 PWM 软件触发刹车控制寄存器**

寄存器	偏移地址	R/W	描述	复位值
<b>TIMER0_PWM_SWBRK</b>	TMR01_BA+0x7C	W	Timer0 PWM 软件触发刹车控制寄存器	0x0000_0000
<b>TIMER1_PWM_SWBRK</b>	TMR01_BA+0x17C	W	Timer1 PWM软件触发刹车控制寄存器	0x0000_0000
<b>TIMER2_PWM_SWBRK</b>	TMR23_BA+0x7C	W	Timer2 PWM软件触发刹车控制寄存器	0x0000_0000
<b>TIMER3_PWM_SWBRK</b>	TMR23_BA+0x17C	W	Timer3 PWM软件触发刹车控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							BRKLTRG
7	6	5	4	3	2	1	0
Reserved							BRKETRG

位	描述	
[31:9]	<b>Reserved</b>	保留.
[8]	<b>BRKLTRG</b>	<p>软件触发电平侦测刹车源 (只写) (写保护)</p> <p>写 1 到该位触发PWM电平侦测刹车源，然后TIMERx_PWMINTSTS1 寄存器中BRKLIF0 和 BRKLIF1 会自动置1。</p> <p><b>注意:</b> 该位为写保护位。详情请参考SYS_REGLCTL 寄存器.</p>
[7:1]	<b>Reserved</b>	保留.
[0]	<b>BRKETRG</b>	<p>软件触发边沿侦测刹车源 (只写) (写保护)</p> <p>写 1 到该位触发PWM边沿侦测刹车源，然后TIMERx_PWMINTSTS1 寄存器中BRKEIF0和BRKEIF1会自动置1。</p> <p><b>注意:</b> 该位为写保护位。详情请参考SYS_REGLCTL 寄存器.</p>

**TIMERx PWMINTENO 定时器 PWM 中断使能寄存器 0**

寄存器	偏移地址	R/W	描述	复位值
<b>TIMER0_PWMINTENO</b>	TMR01_BA+0x80	R/W	Timer0 PWM 中断使能寄存器 0	0x0000_0000
<b>TIMER1_PWMINTENO</b>	TMR01_BA+0x180	R/W	Timer1 PWM中断使能寄存器 0	0x0000_0000
<b>TIMER2_PWMINTENO</b>	TMR23_BA+0x80	R/W	Timer2 PWM中断使能寄存器 0	0x0000_0000
<b>TIMER3_PWMINTENO</b>	TMR23_BA+0x180	R/W	Timer3 PWM中断使能寄存器 0	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved				CMPDIEN	CMPUIEN	PIEN	ZIEN

位	描述	
[31:4]	<b>Reserved</b>	保留.
[3]	<b>CMPDIEN</b>	<b>PWM下数比较中断使能位</b> 0 = 下数比较中断禁止. 1 = 下数比较中断使能
[2]	<b>CMPUIEN</b>	<b>PWM上数比较中断使能位</b> 0 = 上数比较中断禁止. 1 = 上数比较中断使能
[1]	<b>PIEN</b>	<b>PWM 周期点中断使能位</b> 0 = 周期点中断禁止 1 = 周期点中断使能 <b>注意:</b> 在上下计数模式中，周期点就是当前PWM周期的中心点的意思。
[0]	<b>ZIEN</b>	<b>PWM 零点中断使能</b> 0 = 零点中断禁止. 1 = 零点中断使能

TIMERx PWMINTEN1 定时器 PWM 中断使能寄存器 1

寄存器	偏移地址	R/W	描述	复位值
TIMER0_PWM_INTEN1	TMR01_BA+0x84	R/W	Timer0 PWM 中断使能寄存器 1	0x0000_0000
TIMER1_PWM_INTEN1	TMR01_BA+0x184	R/W	Timer1 PWM中断使能寄存器 1	0x0000_0000
TIMER2_PWM_INTEN1	TMR23_BA+0x84	R/W	Timer2 PWM中断使能寄存器 1	0x0000_0000
TIMER3_PWM_INTEN1	TMR23_BA+0x184	R/W	Timer3 PWM中断使能寄存器 1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							BRKLIEN
7	6	5	4	3	2	1	0
Reserved							BRKEIEN

位	描述	
[31:9]	Reserved	保留.
[8]	BRKLIEN	<b>PWM 电平侦测刹车中断使能 (写保护)</b> 0 = PWM 电平侦测刹车中断禁止. 1 = PWM电平侦测刹车中断使能. <b>注意:</b> 该位为写保护位。详情请参考SYS_REGLCTL 寄存器.
[7:1]	Reserved	保留.
[0]	BRKEIEN	<b>PWM 边沿侦测刹车中断使能 (写保护)</b> 0 = PWM 边沿侦测刹车中断禁止. 1 = PWM边沿侦测刹车中断使能. <b>注意:</b> 该位为写保护位。详情请参考SYS_REGLCTL 寄存器.

**Timer PWM Interrupt Status Register 0 (TIMERx\_PWMINTSTS0)**

寄存器	偏移地址	R/W	描述	复位值
TIMER0_PWM_INTSTS0	TMR01_BA+0x88	R/W	Timer0 PWM 中断状态寄存器 0	0x0000_0000
TIMER1_PWM_INTSTS0	TMR01_BA+0x188	R/W	Timer1 PWM中断状态寄存器 0	0x0000_0000
TIMER2_PWM_INTSTS0	TMR23_BA+0x88	R/W	Timer2 PWM中断状态寄存器 0	0x0000_0000
TIMER3_PWM_INTSTS0	TMR23_BA+0x188	R/W	Timer3 PWM中断状态寄存器 0	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved				CMPDIF	CMPUIF	PIF	ZIF

位	描述
[31:4]	<b>Reserved</b> 保留.
[3]	<b>CMPDIF</b> <b>PWM下数比较中断标志</b> 当TIMERx_PWM计数器在下计数方向且计数到CMP比较值，该位由硬件置位。 <b>注意1:</b> 如果 CMP 等于 PERIOD, 在下计数模式没有 CMPDIF 标志。. <b>注意2:</b> 该位写1清0.
[2]	<b>CMPUIF</b> <b>PWM上数比较中断标志</b> 当TIMERx_PWM计数器在上计数方向且计数到CMP比较值，该位由硬件置位。 <b>注意1:</b> 如果 CMP 等于 PERIOD, 在上计数模式和上下计数模式没有 CMPDIF 标志。. <b>注意2:</b> 该位写1清0.
[1]	<b>PIF</b> <b>PWM 周期点中断标志</b> 当TIMERx_PWM计数器计数到PERIOD, 该位由硬件置位。 <b>注意1:</b> ,在上下计数模式，PIF标志是当前PWM周期中心点标志的意思。. <b>注意2:</b> 该位写1清0.
[0]	<b>ZIF</b> <b>PWM 零点中断标志</b> 当TIMERx_PWM计数器计数到0, 该位由硬件置位。 <b>注意:</b> 该位写1清0.

**TIMERx PWMINTSTS1 定时器 PWM 中断状态寄存器 1**

寄存器	偏移地址	R/W	描述	复位值
TIMER0_PWMINTSTS1	TMR01_BA+0x8C	R/W	Timer0 PWM 中断状态寄存器 1	0x0000_0000
TIMER1_PWMINTSTS1	TMR01_BA+0x18C	R/W	Timer1 PWM中断状态寄存器 1	0x0000_0000
TIMER2_PWMINTSTS1	TMR23_BA+0x8C	R/W	Timer2 PWM中断状态寄存器 1	0x0000_0000
TIMER3_PWMINTSTS1	TMR23_BA+0x18C	R/W	Timer3 PWM中断状态寄存器 1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved						BRKLSTS1	BRKLSTS0
23	22	21	20	19	18	17	16
Reserved						BRKESTS1	BRKESTS0
15	14	13	12	11	10	9	8
Reserved						BRKLIF1	BRKLIF0
7	6	5	4	3	2	1	0
Reserved						BRKEIF1	BRKEIF0

位	描述	
[31:26]	Reserved	保留.
[25]	BRKLSTS1	<p><b>PWMx_CH1的电平侦测刹车状态 (只读)</b>            0 = PWMx_CH1 电平侦测刹车状态释放.            1 = PWMx_CH1 在电平侦测刹车状态  <b>注意:</b> 如果TIMERx_PWM 电平侦测刹车源释放, 当当前PWM周期完成, PWMx_CH0 和 PWMx_CH1 都会释放刹车状态, 且PWMx_CH0 和 PWMx_CH1 从下一个完整PWM周期开始重新开始输出波形。</p>
[24]	BRKLSTS0	<p><b>PWMx_CH0的电平侦测刹车状态 (只读)</b>            0 = PWMx_CH0电平侦测刹车状态释放.            1 = PWMx_CH0 在电平侦测刹车状态  <b>注意:</b> 如果TIMERx_PWM 电平侦测刹车源释放, 当当前PWM周期完成, PWMx_CH0 和 PWMx_CH0 都会释放刹车状态, 且PWMx_CH0 和 PWMx_CH1 从下一个完整PWM周期开始重新开始输出波形。</p>
[23:18]	Reserved	保留.
[17]	BRKESTS1	<p><b>PWMx_CH1的边沿侦测刹车状态 (只读)</b>            0 = PWMx_CH1 边沿侦测刹车状态释放.            1 = PWMx_CH1 在边沿侦测刹车状态  <b>注意:</b> 用户可以设置 BRKEIF1 为1来清除BRKEIF1 标志, 当当前PWM周期完成, PWMx_CH1 会释放刹车状态, 且PWMx_CH1 从下一个完整PWM周期开始重新开始输出波形。.</p>

[16]	<b>BRKESTS0</b>	<b>PWMx_CH0的边沿侦测刹车状态(只读)</b> 0 = PWMx_CH0 边沿侦测刹车状态释放. 1 = PWMx_CH0 在边沿侦测刹车状态 <b>注意:</b> 用户可以设置 BRKEIF0 为1来清除BRKEIF0标志，当当前PWM周期完成，PWMx_CH0 会释放刹车状态，且PWMx_CH0 从下一个完整PWM周期开始重新开始输出波形。.
[15:10]	<b>Reserved</b>	保留.
[9]	<b>BRKLIF1</b>	<b>PWMx_CH1 的电平侦测刹车中断标志(写保护)</b> 0 = PWMx_CH1 电平侦测刹车事件没有发生 1 = PWMx_CH1 发生了电平侦测刹车事件. <b>注意1:</b> 该位写1清0. <b>注意2:</b> 该位为写保护位。详情请参考SYS_REGLCTL 寄存器.
[8]	<b>BRKLIF0</b>	<b>PWMx_CH0 的电平侦测刹车中断标志(写保护)</b> 0 = PWMx_CH0 电平侦测刹车事件没有发生 1 = PWMx_CH0 发生了电平侦测刹车事件. <b>注意1:</b> 该位写1清0. <b>注意2:</b> 该位为写保护位。详情请参考SYS_REGLCTL 寄存器.
[7:2]	<b>Reserved</b>	保留.
[1]	<b>BRKEIF1</b>	<b>PWMx_CH1 的边沿侦测刹车中断标志(写保护)</b> 0 = PWMx_CH1 边沿侦测刹车事件没有发生 1 = PWMx_CH1 发生了边沿侦测刹车事件. <b>注意1:</b> 该位写1清0. <b>注意2:</b> 该位为写保护位。详情请参考SYS_REGLCTL 寄存器.
[0]	<b>BRKEIF0</b>	<b>PWMx_CH0 的边沿侦测刹车中断标志(写保护)</b> 0 = PWMx_CH0 边沿侦测刹车事件没有发生 1 = PWMx_CH0 发生了边沿侦测刹车事件. <b>注意1:</b> 该位写1清0. <b>注意2:</b> 该位为写保护位。详情请参考SYS_REGLCTL 寄存器.

**TIMERx PWMADCTS 定时器 PWM ADC 触发控制寄存器**

寄存器	偏移地址	R/W	描述	复位值
<b>TIMER0_PWMADCTS</b>	TMR01_BA+0x90	R/W	Timer0 PWM ADC 触发源选择寄存器	0x0000_0000
<b>TIMER1_PWMADCTS</b>	TMR01_BA+0x190	R/W	Timer1 PWM ADC触发源选择寄存器	0x0000_0000
<b>TIMER2_PWMADCTS</b>	TMR23_BA+0x90	R/W	Timer2 PWM ADC触发源选择寄存器	0x0000_0000
<b>TIMER3_PWMADCTS</b>	TMR23_BA+0x190	R/W	Timer3 PWM ADC触发源选择寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
TRGEN	Reserved				TRGSEL		

位	描述	
[31:8]	Reserved	保留.
[7]	TRGEN	<b>PWM 计数器事件触发ADC转换使能位</b> 0 = PWM 计数器事件触发ADC 转换禁止 1 = PWM计数器事件触发ADC 转换使能
[6:3]	Reserved	保留.
[2:0]	TRGSEL	<b>PWM 计数器事件源选择来触发ADC转换</b> 000 = 触发 EADC 转换在零点 (ZIF). 001 =触发EADC 转换在周期点 (PIF). 010 =触发EADC 转换在零点或周期点 (ZIF or PIF). 011 =触发EADC 转换在上数比较点 (CMPUIF). 100 =触发EADC 转换在下数比较点 (CMPPDF). 其他 = 保留.

**TIMERx PWMSCTL 定时器 PWM 同步控制寄存器**

寄存器	偏移地址	R/W	描述	复位值
TIMER0_PWMS CTL	TMR01_BA+0x94	R/W	Timer0 PWM 同步控制寄存器	0x0000_0000
TIMER1_PWMS CTL	TMR01_BA+0x19 4	R/W	Timer1 PWM同步控制寄存器	0x0000_0000
TIMER2_PWMS CTL	TMR23_BA+0x94	R/W	Timer2 PWM同步控制寄存器	0x0000_0000
TIMER3_PWMS CTL	TMR23_BA+0x19 4	R/W	Timer3 PWM同步控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							SYNCSRC
7	6	5	4	3	2	1	0
Reserved						SYNCMODE	

位	描述	
[31:9]	Reserved	保留.
[8]	SYNCSRC	<p><b>PWM 同步计数器启动/清除动作触发源选择</b></p> <p>0 = 计数器由TIMER0_PWMSTRG STRGEN触发同步启动/清除 1 = 计数器由TIMER2_PWMSTRG STRGEN触发同步启动/清除</p> <p><b>注意1:</b> 如果 TIMER0/1/2/3 PWM 计数器同步源来自 TIMER0，那么 TIMER0_PWMSCTL[8], TIMER1_PWMSCTL[8], TIMER2_PWMSCTL[8] 和 TIMER3_PWMSCTL[8] 应该要为 0。</p> <p><b>注意2:</b> 如果 TIMER0/1/ PWM 计数器同步源来自 TIMER0, 那么 TIMER0_PWMSCTL[8] 和 TIMER1_PWMSCTL[8] 应该置 0。如果 TIMER2/3/ PWM 计数器同步源来自 TIMER2, TIME2_PWMSCTL[8] 和 TIMER3_PWMSCTL[8] 应该置1。</p>
[7:2]	Reserved	保留.
[1:0]	SYNCMODE	<p><b>PWM 同步模式使能选择</b></p> <p>00 = PWM 同步功能禁止. 01 = PWM计数器同步启动功能使能 10 = 保留. 11 = PWM 计数器同步清除功能使能</p>

**TIMERx\_PWMSTRG 定时器 PWM 同步触发寄存器**

寄存器	偏移地址	R/W	描述	复位值
TIMER0_PWMSTRG	TMR01_BA+0x98	W	Timer0 PWM 同步触发寄存器	0x0000_0000
TIMER2_PWMSTRG	TMR23_BA+0x98	W	Timer2 PWM同步触发寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved							STRGEN

位	描述	
[31:1]	Reserved	保留.
[0]	STRGEN	<p><b>PWM 计数器同步触发使能位 (只写)</b></p> <p>PMW 计数器同步功能用于对选择的PWM通道（包括TIMER0/1/2/3 PWM, TIMER0/1 PWM 和 TIMER2/3 PWM）计数器根据TIMERx_PWMCTL 的设定在相同的时刻启动或清除计数器。</p> <p><b>注意:</b> 该位仅对 TIMER0 和 TIMER2有效</p>

## TIMERx PWMSTATUS 定时器 PWM 状态寄存器

寄存器	偏移地址	R/W	描述	复位值
TIMER0_PWM_STATUS	TMR01_BA+0x9C	R/W	Timer0 PWM 状态寄存器	0x0000_0000
TIMER1_PWM_STATUS	TMR01_BA+0x19C	R/W	Timer1 PWM状态寄存器	0x0000_0000
TIMER2_PWM_STATUS	TMR23_BA+0x9C	R/W	Timer2 PWM状态寄存器	0x0000_0000
TIMER3_PWM_STATUS	TMR23_BA+0x19C	R/W	Timer3 PWM状态寄存器	0x0000_0000

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

位	描述	
[31:17]	Reserved	保留.
[16]	EADCTRGF	<p>触发 ADC 开始转换标志            0 = 未发生EADC 计数器事件触发ADC开始转换            1 = 发生了EADC计数器事件触发ADC开始转换.  <b>注意:</b> 该位写1清0.</p>
[15:1]	Reserved	保留.
[0]	CNTMAXF	<p>PWM 计数器等于 0xFFFF 标志            0 = 表示 PWM 计数器值没有到达它的最大值0xFFFF.            1 = 表示 PWM 计数器值到达它的最大值  <b>注意:</b> 该位写1清0.</p>

**TIMERx PWMPBUF 定时器 PWM 周期值缓存寄存器**

寄存器	偏移地址	R/W	描述	复位值
<b>TIMER0_PWM_PBUF</b>	TMR01_BA+0xA0	R	Timer0 PWM 周期值缓存寄存器	0x0000_0000
<b>TIMER1_PWM_PBUF</b>	TMR01_BA+0x1A0	R	Timer1 PWM周期值缓存寄存器	0x0000_0000
<b>TIMER2_PWM_PBUF</b>	TMR23_BA+0xA0	R	Timer2 PWM周期值缓存寄存器	0x0000_0000
<b>TIMER3_PWM_PBUF</b>	TMR23_BA+0x1A0	R	Timer3 PWM周期值缓存寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
PBUF							
7	6	5	4	3	2	1	0
PBUF							

位	描述	
[31:16]	Reserved	保留.
[15:0]	PBUF	PWM 周期值缓存寄存器 (只读) 用于使PERIOD生效

**TIMERx PWMCMPCBUF 定时器 PWM 比较值缓存寄存器**

寄存器	偏移地址	R/W	描述	复位值
<b>TIMER0_PWM_CMPBUF</b>	TMR01_BA+0xA4	R	Timer0 PWM 比较值缓存寄存器	0x0000_0000
<b>TIMER1_PWM_CMPBUF</b>	TMR01_BA+0x1A4	R	Timer1 PWM比较值缓存寄存器	0x0000_0000
<b>TIMER2_PWM_CMPBUF</b>	TMR23_BA+0xA4	R	Timer2 PWM比较值缓存寄存器	0x0000_0000
<b>TIMER3_PWM_CMPBUF</b>	TMR23_BA+0x1A4	R	Timer3 PWM比较值缓存寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
CMPBUF							
7	6	5	4	3	2	1	0
CMPBUF							

位	描述	
[31:16]	Reserved	保留.
[15:0]	CMPBUF	PWM 周期值缓存寄存器 (只读) 用于使CMP生效

## 6.8 WDT 看门狗定时器

### 6.8.1 概述

设计看门狗定时器的目的是，当系统运行到一个未知状态时，通过它来使系统复位。这种做法可以预防系统进入到无限期的死循环。此外，该看门狗定时器支持系统从空闲/掉电模式唤醒。

### 6.8.2 特性

- 18位的向上看门狗定时器可满足用户溢出时间间隔要求
- 溢出时间间隔( $2^4 \sim 2^{18}$ )个WDT\_CLK时钟周期可选，如WDT\_CLK = 10 kHz，那么溢出时间间隔是1.6 ms ~ 26.214 s
- 系统复位保持时间( $1 / \text{WDT\_CLK} * 63$ )
- 支持看门狗定时器复位延时周期，包括1026、130、18或3个WDT\_CLK的复位延时时间
- 通过设置CONFIG0中CWDTE[2:0]位，支持芯片上电或复位条件下看门狗强制打开
- 如果时钟源选择内部低速10k时钟或LXT时钟，支持看门狗定时器溢出唤醒。

### 6.8.3 框图

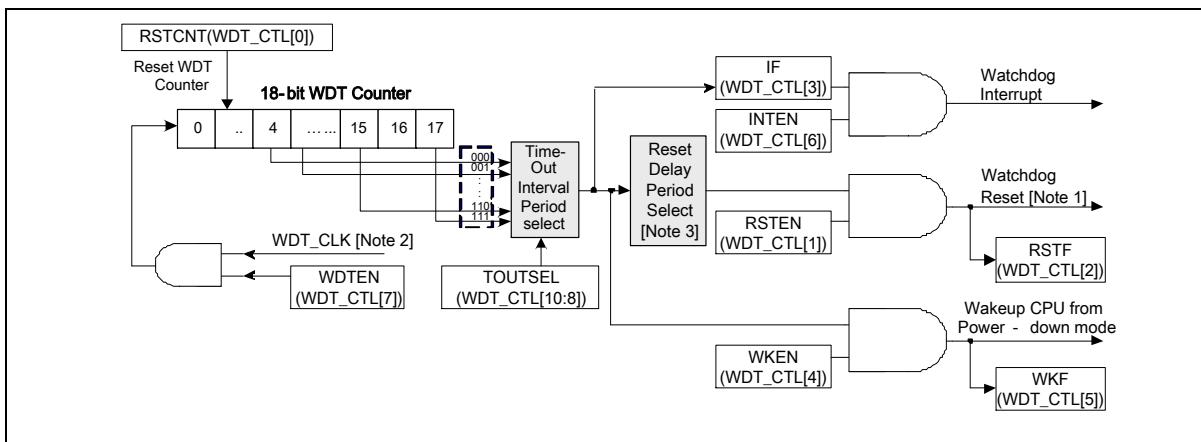


图 6.8-1 看门狗定时器框图

**注意1:** 看门狗定时器复位CPU将持续63个WDT\_CLK

**注意2:** 如果看门狗定时器时钟源选择内部 10kHz 时钟源，CPU可以被看门狗定时溢出中断信号唤醒。

**注意3:** 看门狗定时器复位延时周期可以选择3/18/130/1026个WDT\_CLK。

### 6.8.4 基本配置

- 时钟源配置
  - 通过WDTSEL (CLK\_CLKSEL1[1:0])选择 WDT 外设时钟源
  - 通过 WDTCKEN (CLK\_APBCLK0[0])使能WDT外设时钟
  - 通过CWDTE[2:0] (CWDTE[2] 是 Config0[31], CWDTE[1:0] 是 Config0[4:3])配置在芯片上电或复位后强制使能 WDT 控制器

WDT 时钟控制如下。

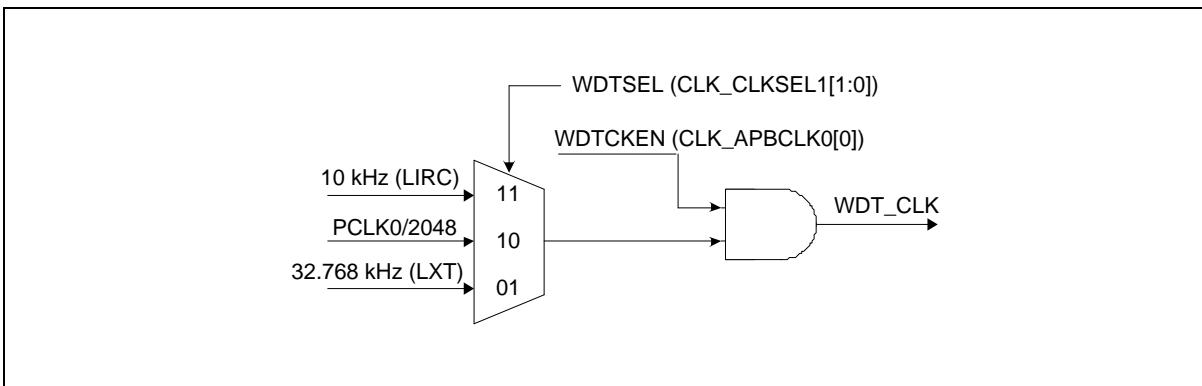


图 6.8-2 看门狗定时器时钟控制

## 6.8.5 功能描述

看门狗定时器(WDT)包含一个18位的可设置溢出时间间隔的向上计数器。表 6.8-1 显示了WDT溢出时间间隔选择，图 6.8-3 看门狗定时器溢出时间间隔和复位周期。

### 6.8.5.1 WDT 定时溢出中断

对WDTEN (WDTCR[7])置位可以使能WDT功能，然后WDT计数器向上计数。SYNC (WDT\_CTL[30])表示使能/禁止WDTEN功能是否完成。通过设置TOUTSEL (WDTCR[10:8]), 可以选择8个不同的溢出时间间隔。当WDT计数器计到TOUTSEL (WDTCR[10:8])设定值，WDT产生溢出中断，然后IF (WDT\_CTL[3])标志立即被置1。如果INTEN (WDT\_CTL[6])使能，产生WDT超时中断。

### 6.8.5.2 WDT 复位延时周期和复位系统

如果(WDT\_CTL[3])标志被置位后，还会有一个固定的延时周期 $T_{RSTD}$ ，用户应在 $T_{RSTD}$ 延时计完之前对RSTCNT (WDT\_CTL[0])或WDT\_RSTCNT置1来复位18位的WDT向上计数器的值，以防止产生WDT时间溢出复位信号。此外，用户还应该设置RSTDSEL (WDT\_ALTCTL [1:0])选择复位延时周期用于清看门狗定时器，如 $T_{RSTD}$ 时间计满以后，WDT向上计数器的值仍没有被清除，若RSTEN (WDT\_CTL[1])位为1，则RSTF (WDT\_CTL[2])标志会被置1，然后芯片立即复位。 $T_{RST}$ 复位周期将保持至少63个WDT时钟周期，然后芯片从复位向量地址(0x0000\_0000)重新开始执行程序。芯片由WDT定时溢出复位后，RSTF (WDT\_CTL[2])标志将会保持1。用户可以通过检查该标志来确认系统是否因WDT定时溢出复位。

TOUTSEL	超时间隔周期 $T_{TIS}$	复位延时周期 $T_{RSTD}$
000	$2^4 * T_{WDT}$	$(3/18/130/1026) * T_{WDT}$
001	$2^6 * T_{WDT}$	$(3/18/130/1026) * T_{WDT}$
010	$2^8 * T_{WDT}$	$(3/18/130/1026) * T_{WDT}$
011	$2^{10} * T_{WDT}$	$(3/18/130/1026) * T_{WDT}$
100	$2^{12} * T_{WDT}$	$(3/18/130/1026) * T_{WDT}$
101	$2^{14} * T_{WDT}$	$(3/18/130/1026) * T_{WDT}$
110	$2^{16} * T_{WDT}$	$(3/18/130/1026) * T_{WDT}$
111	$2^{18} * T_{WDT}$	$(3/18/130/1026) * T_{WDT}$

表 6.8-1 看门狗定时器定时溢出间隔周期选择

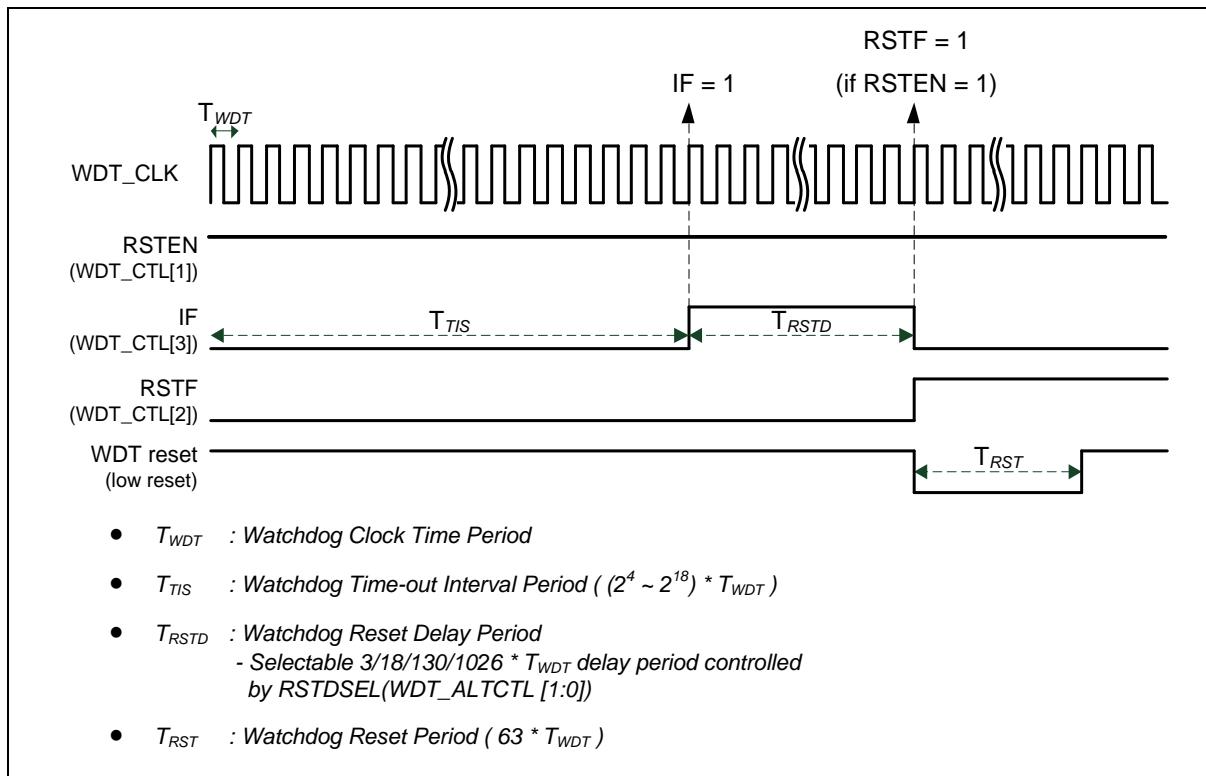


图 6.8-3 看门狗定时器定时溢出间隔和复位周期时序图

### 6.8.5.3 WDT 唤醒

如果看门狗时钟源选择为内部10k或外部低速时钟，如果WKEN (WDT\_CTL[4])位被使能，看门狗定时溢出中断产生后，系统能从Power-down模式下被唤醒。注意，用户需在系统进入power down模式前，设置LXTEN (CLK\_PWRCTL [1]) 或LIRCN (CLK\_PWRCTL [3])作为时钟源，因为当系统进入power down模式后外设时钟将被禁止。同时，WKF (WDT\_CTL[5])标志会被自动置1。用户可以通过查询WKF (WDT\_CTL[5])标志知道系统是否是被看门狗定时溢出中断唤醒。

#### 6.8.5.4 WDT ICE 调试

当ICE连接到MCU，WDT计数器是否还在计数由ICEDEBUG(WDT\_CTL[31])决定。ICEDEBUG默认值是0，当CPU被ICE停住，WDT停止计数。如果ICEDEBUG置1，无论CPU是否被ICE停住，WDT将保持计数。

### 6.8.6 寄存器映射

R: 只读, W: 只写, R/W: 可读写

寄存器	偏移地址	R/W	描述	复位值
<b>WDT 基地址:</b>				
<b>WDT_BA = 0x4004_0000</b>				
WDT_CTL	WDT_BA+0x00	R/W	WDT 控制寄存器	0x0000_0700
WDT_ALTCTL	WDT_BA+0x04	R/W	WDT 选择控制寄存器	0x0000_0000
WDT_RSTCNT	WDT_BA+0x08	W	WDT 复位计数器寄存器	0x0000_0000

## 6.8.7 寄存器描述

WDT\_CTL WDT 控制寄存器

寄存器	偏移地址	R/W	描述	复位值
WDT_CTL	WDT_BA+0x00	R/W	WDT 控制寄存器	0x0000_0700

31	30	29	28	27	26	25	24
ICEDEBUG	SYNC	Reserved					
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved					TOUTSEL		
7	6	5	4	3	2	1	0
WDTEN	INTEN	WKF	WKEN	IF	RSTF	RSTEN	RSTCNT

位	描述
[31]	<b>ICEDEBUG</b> ICE 调试模式下看门狗计数控制位(写保护) 0 = ICE 调试模式影响看门狗计数, 当ICE让CPU运行停止, 看门狗计数器也停止。 1 = ICE 调试模式不影响看门狗计数。不论CPU受ICE停止还是运行, 看门狗继续计数 <b>注意:</b> 该位为写保护位. 详情请参考SYS_REGLCTL 寄存器
[30]	<b>SYNC</b> <b>WDT 使能控制 SYNC 标志指示 (只读)</b> 如果用户使能/禁止WDTEN (WDT_CTL[7]), 该标志可以指示使能/禁止WDTEN操作是否完成。 0 = 设置 WDTEN 位完成。 1 = 设置 WDTEN位正在同步, 还未生效。 <b>注意:</b> 执行使能/禁止 WDTEN 位需要2 * WDT_CLK 个时钟周期来生效
[29:11]	<b>Reserved</b> 保留.
[10:8]	<b>TOUTSEL</b> <b>看门狗定时器溢出时间间隔选择 (写保护)</b> 以下三个位用于选择看门狗定时溢出周期 000 = $2^4 * WDT\_CLK$ . 001 = $2^6 * WDT\_CLK$ . 010 = $2^8 * WDT\_CLK$ . 011 = $2^{10} * WDT\_CLK$ . 100 = $2^{12} * WDT\_CLK$ . 101 = $2^{14} * WDT\_CLK$ . 110 = $2^{16} * WDT\_CLK$ . 111 = $2^{18} * WDT\_CLK$ . <b>注意:</b> 该位为写保护位. 详情请参考SYS_REGLCTL 寄存器.
[7]	<b>WDTEN</b> <b>看门狗定时器使能控制位 (写保护)</b> 0 =看门狗定时器禁止. (该操作会复位看门狗计数器的值.)

		1 = 看门狗定时器使能。  <b>注意1:</b> 该位为写保护位。详情请参考SYS_REGLCTL 寄存器。 <b>注意2:</b> 如果CWDTEN[2:0] (Config0[31] 和 Config0[4:3]的组合) 位的值非111, 该位会被强制置1, 且用户不能把它改为0.
[6]	INTEN	<b>看门狗定时器定时溢出中断使能控制(写保护)</b> 如果该位使能, 看门狗定时溢出后会产生中断信号, 并告知CPU 0 = 看门狗定时溢出中断禁止. 1 = 看门狗定时溢出中断使能. <b>注意:</b> 该位为写保护位。详情请参考SYS_REGLCTL 寄存器
[5]	WKF	<b>看门狗定时器溢出唤醒标志(写保护)</b> 该位表明看门狗定时器中断唤醒标志的状态 0 = 看门狗定时器没有导致芯片唤醒 1 = 芯片从Idle或Power-down 模式被唤醒 <b>注意1:</b> 该位为写保护位。详情请参考SYS_REGLCTL 寄存器 <b>注意2:</b> 该位为写1清零.
[4]	WKEN	<b>看门狗定时器定时溢出唤醒功能控制位 (写保护)</b> 如果此位被置1, 当IF (WDT_CTL[3])被置1且INTEN (WDT_CTL[6])被使能, 看门狗定时溢出中断信号将会触发唤醒MCU. 0 = 唤醒触发事件禁止, 即便看门狗定时器定时溢出中断发生 1 = 唤醒触发事件使能, 如果看门狗定时溢出中断产生, 将唤醒MCU <b>注意1:</b> 该位为写保护位。详情请参考SYS_REGLCTL 寄存器 <b>注意2:</b> MCU能被看门狗定时器定时溢出中断信号唤醒的条件是看门狗定时器时钟源必须为内部10K时钟或外部低速时钟
[3]	IF	<b>看门狗定时器定时中断标志</b> 当看门狗定时器计数器的值达到溢出时间间隔, 该位将被置1 0 = 没有发生看门狗定时溢出中断 1 = 有看门狗定时溢出中断发生. <b>注意:</b> 该位写1清零
[2]	RSTF	<b>看门狗定时器复位标志</b> 该位用来指示系统是否因看门狗定时器定时溢出发生复位 0 = 没有发生看门狗定时溢出复位 1 = 发生了看门狗定时溢出复位. <b>注意:</b> 该位写1清零.
[1]	RSTEN	<b>看门狗定时器溢出复位使能位 (写保护)</b> 如果看门狗计数器的值达到设定值, 且指定的WDT复位延时时间已经计完, 计数器值还没有被清零, 如果该位被设置, 则会使能看门狗定时溢出复位 0 = 看门狗定时溢出复位功能禁止. 1 = 看门狗定时溢出复位功能使能 <b>注意:</b> 该位为写保护位。详情请参考SYS_REGLCTL 寄存器
[0]	RSTCNT	<b>清看门狗计数器 (写保护)</b> 0 = 不影响. 1 = 18位看门狗定时器计数值复位清零. <b>注意1:</b> 该位为写保护位。详情请参考SYS_REGLCTL 寄存器

	<b>注意2:</b> 该位硬件自动清零.
--	-----------------------

WDT\_ALTCTL WDT 选择控制寄存器

寄存器	偏移地址	R/W	描述	复位值
T_ALTCTL	WDT_BA+0x04	R/W	WDT 选择控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved						RSTDSEL	

位	描述	
[31:2]	<b>Reserved</b>	保留.
[1:0]	<b>RSTDSEL</b>	<p><b>看门狗定时器复位延时选择 (写保护)</b></p> <p>当看门狗定时器发生溢出，用户有一个复位延时时间来将看门狗计数器清零，以防止看门狗溢出复位。通过置位RSTCNT (WDT_CTL[0])清看门狗计数器。对不同的看门狗溢出时间，用户也可以设置RSTDSEL值选择适当的看门狗复位延时时间。</p> <p>00 = 看门狗定时器复位延时时间是1026 * WDT_CLK.      01 = 看门狗定时器复位延时时间是130 * WDT_CLK.      10 = 看门狗定时器复位延时时间是18 * WDT_CLK.      11 = 看门狗定时器复位延时时间是3 * WDT_CLK.</p> <p><b>注意1:</b> 该位为写保护位. 详情请参考SYS_REGLCTL 寄存器  <b>注意2:</b> 如果看门狗定时器超时复位发生，该寄存器的值将被清零</p>

WDT\_RSTCNT WDT 复位计数器寄存器

寄存器	偏移地址	R/W	描述	复位值
WDT_RSTCNT	WDT_BA+0x08	W	WDT 复位计数器寄存器	0x0000_0000

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

位	描述	
[31:0]	RSTCNT	<p><b>WDT 复位计数器寄存器</b>            写 0x00005AA5 到该域会复位内部WDT18位上数计数器值为0。  <b>注意:</b> 执行 RSTCNT 复位计数器需要2 * WDT_CLK 周期生效。  <b>注意:</b> RSTCNT (WDT_CTL[0]) 位是写保护位, RSTCNT (WDT_RSTCNT[31:0]) 不是写保护位</p>

## 6.9 WWDT 窗口看门狗定时器

### 6.9.1 概述

窗口看门狗定时器(WWDT)用于在一个窗口时间内执行系统复位，以防止程序在不可预知条件下跑到一个不可控的状态。

### 6.9.2 特性

- 6位向下计数值(CNTDAT, WWDT\_CNT[5:0])和6位比较值(CMPDAT, WWDT\_CTL[21:16])，使得窗口周期更加灵活
- 支持4位值(PSCSEL, WWDT\_CTL[11:8])选择看门狗预分频值，预分频计数器最大可达11位
- WWDT计数器在空闲或掉电模式下暂停

### 6.9.3 框图

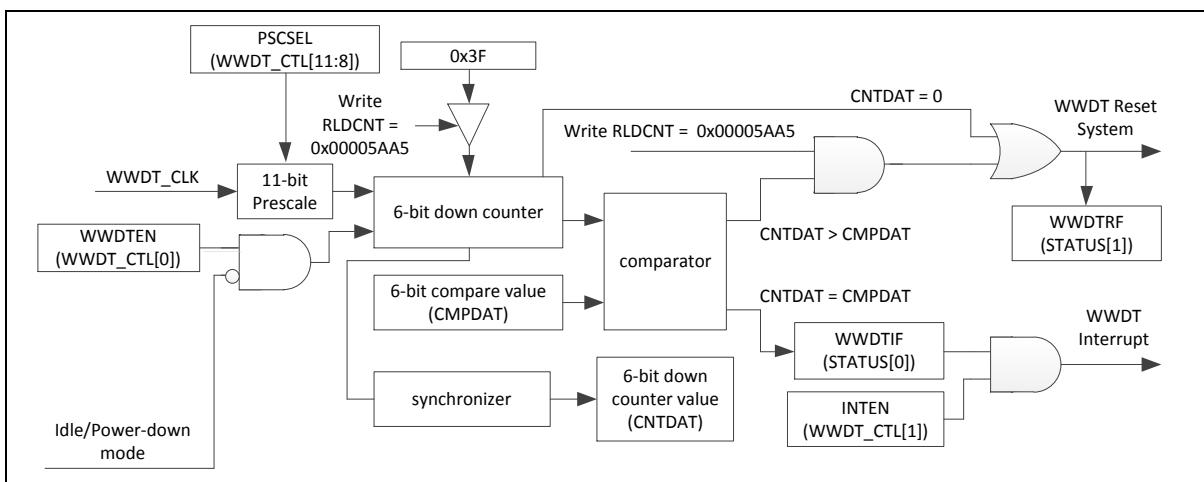


图 6.9-1 WWDT 框图

### 6.9.4 基本配置

- 时钟源配置
  - 通过WWDTSEL (CLK\_CLKSEL1[31:30])选择WWDT外设时钟源
  - 通过WDTCKEN (CLK\_APBCLK0[0])使能WWDT外设时钟

WWDT时钟控制如下图所示：

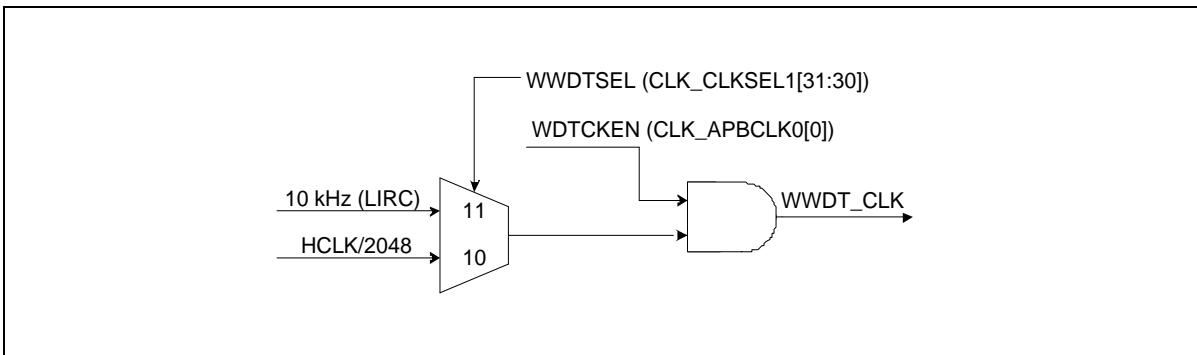


图 6.9-2 WWDT 时钟控制

### 6.9.5 功能描述

窗口看门狗定时器(WWDT)是一个 6位向下计数器，该计数器带一个可选择预分频值，不同的预分频值对应不同的看门狗定时溢出时间。6位窗口看门狗定时器的时钟源可以是系统时钟经2048 (HCLK/2048) 分频的时钟，也可以是内部10k低速RC振荡时钟源，看门狗的时钟源带一个可选择的11位预分频值，该值可通过PSCSEL (WWDT\_CTL[11:8])位来设置选择，对应预分频值如下表

PSCSEL	预分频值	最大定时溢出间隔 (WWDT_CLK=10 KHz)
0000	1	$1 * 64 * T_{WWDT}$
0001	2	$2 * 64 * T_{WWDT}$
0010	4	$4 * 64 * T_{WWDT}$
0011	8	$8 * 64 * T_{WWDT}$
0100	16	$16 * 64 * T_{WWDT}$
0101	32	$32 * 64 * T_{WWDT}$
0110	64	$64 * 64 * T_{WWDT}$
0111	128	$128 * 64 * T_{WWDT}$
1000	192	$192 * 64 * T_{WWDT}$
1001	256	$256 * 64 * T_{WWDT}$
1010	384	$384 * 64 * T_{WWDT}$
1011	512	$512 * 64 * T_{WWDT}$
1100	768	$768 * 64 * T_{WWDT}$
1101	1024	$1024 * 64 * T_{WWDT}$
1110	1536	$1536 * 64 * T_{WWDT}$
1111	2048	$2048 * 64 * T_{WWDT}$

表 6.9-1 WWDT 预分频值选择

#### 6.9.5.1 WWDT 计数

当WWDTEN (WWDT\_CTL[0])位被使能，窗口看门狗向下计数器将会从0x3F向下递减计数到0. 为了防止程序在非用户指定位置关闭窗口看门狗定时器，窗口看门狗定时器控制寄存器WWDT\_CTL在芯片上电或复位后仅可写一次。当WWDTEN (WWDT\_CTL[0])位被软件使能以后，用户不能禁止窗口看门狗定时

器 (WWDTEN), 修改计数器预分频周期(PSCSEL), 或修改窗口比较值 (CMPDAT), 除非芯片复位, 除非芯片复位。

为了避免在CPU时钟禁止时系统复位, 当CPU进入空闲或掉电模式, WWDT计数器会停止计数。在CPU进入正常模式后, WWDT计数器会开始下计数。

#### 6.9.5.2 WWDT 比较匹配中断

窗口看门狗定时器向下计数过程中, 当窗口看门狗定时器计数值(CNTDAT) 等于比较值CMPDAT, WWDTIF (WWDT\_STATUS[0])会被置1, WWDTIF 可以被软件清零; 如果INTEN (WWDT\_CTL[1])位被使能, 当WWDTIF 位被硬件置1, 就会产生窗口看门狗比较匹配中断。

#### 6.9.5.3 WWDT 复位系统

图 6.9-3 表示WWDT三种复位和重载动作情况.

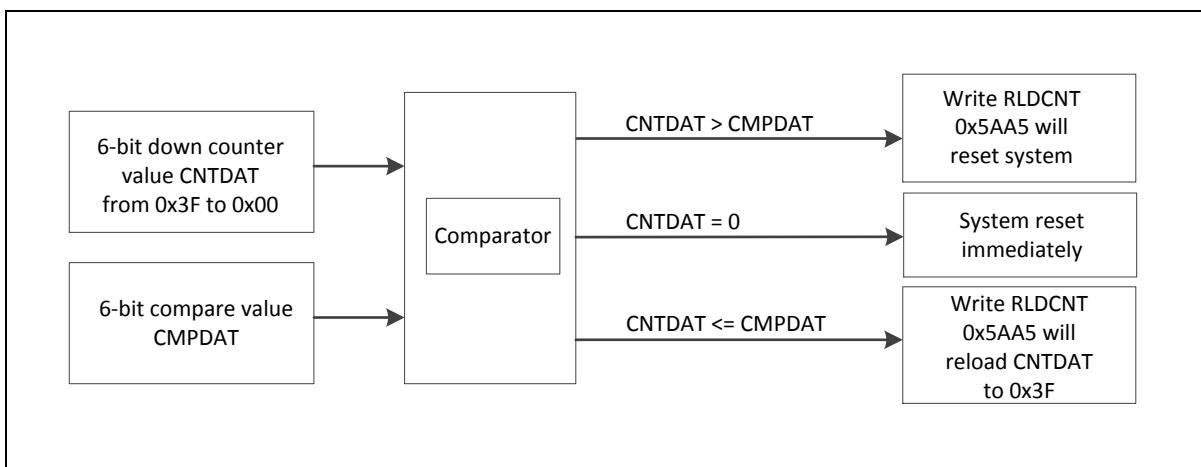


图 6.9-3 WWDT 复位和重载动作

如果当前 CNTDAT (WWDT\_CNT[5:0]) 大于 CMPDAT (WWDT\_CTL[21:16]) 且用户写0x00005AA5 到 WWDT\_RLDCNT 寄存器, WWDT 立即产生复位系统信号, 芯片复位。当CNTDAT > CMPDAT , WWDT重载计数器波形如下图。

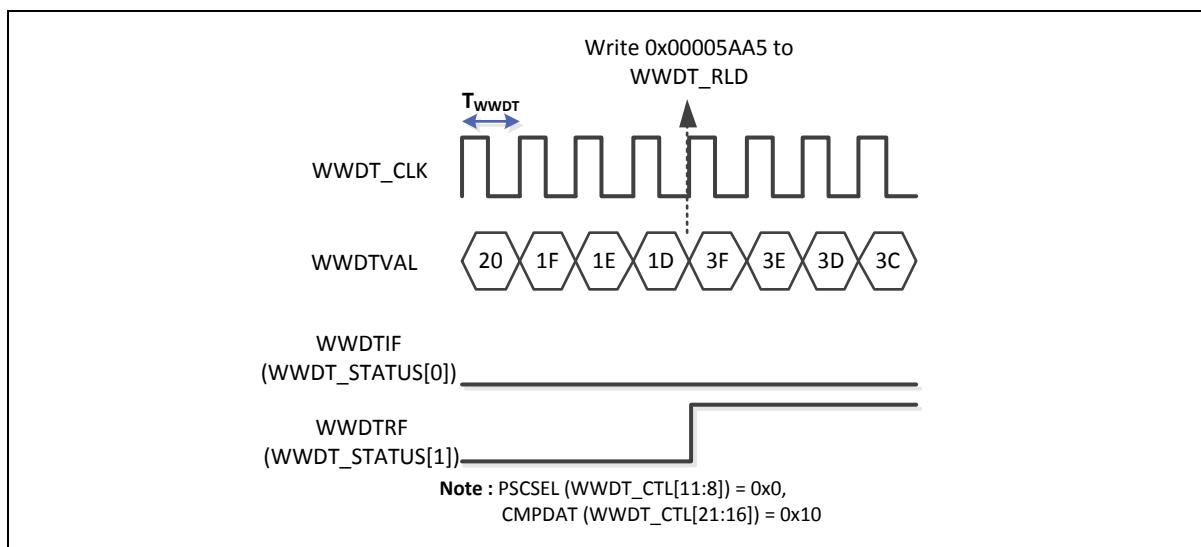


图 6.9-4 当 CNTDAT > CMPDAT WWDT 重载计数器

当 WWDTIF (WWDT\_STATUS[0]) 置位，用户必须通过写 0x00005AA5 到 WWDT\_RLDCNT 重载 WWDT 计数器值为 0x3F，以防止 WWDT 计数器值到 0 产生系统复位信号使系统复位。图 6.9-5 表示当 CNTDAT < CMPDAT WWDT 重载计数器波形。图 6.9-6 表示如果用户在计数器数到 0 之前没有写 0x00005AA5 到 WWDT\_RLD WWDT 产生复位系统的波形。

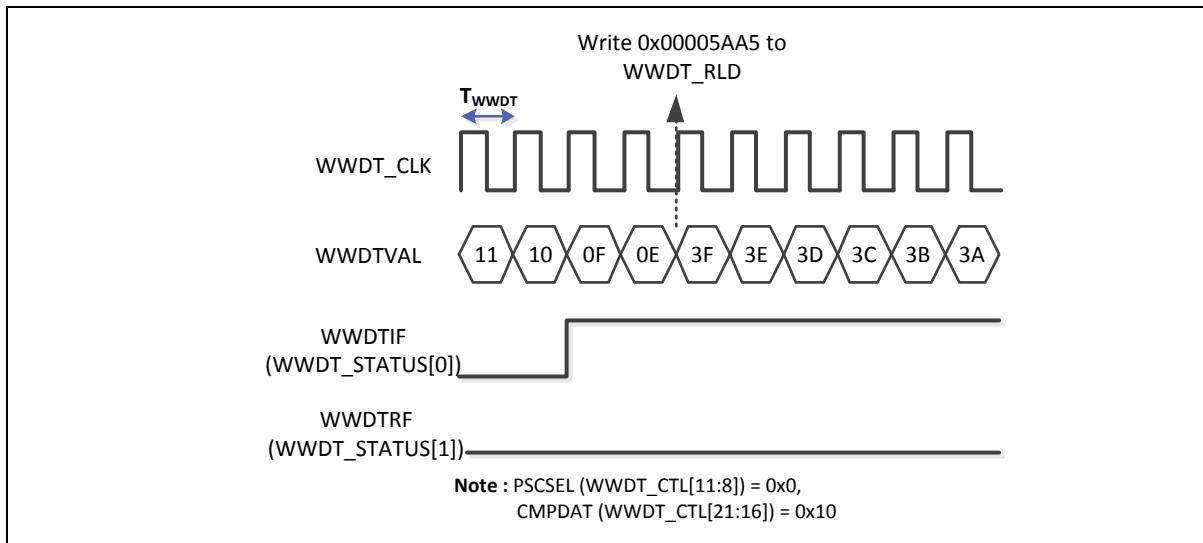


图 6.9-5 当 WWDT\_CNT < WINCMPWWDT 重载计数器

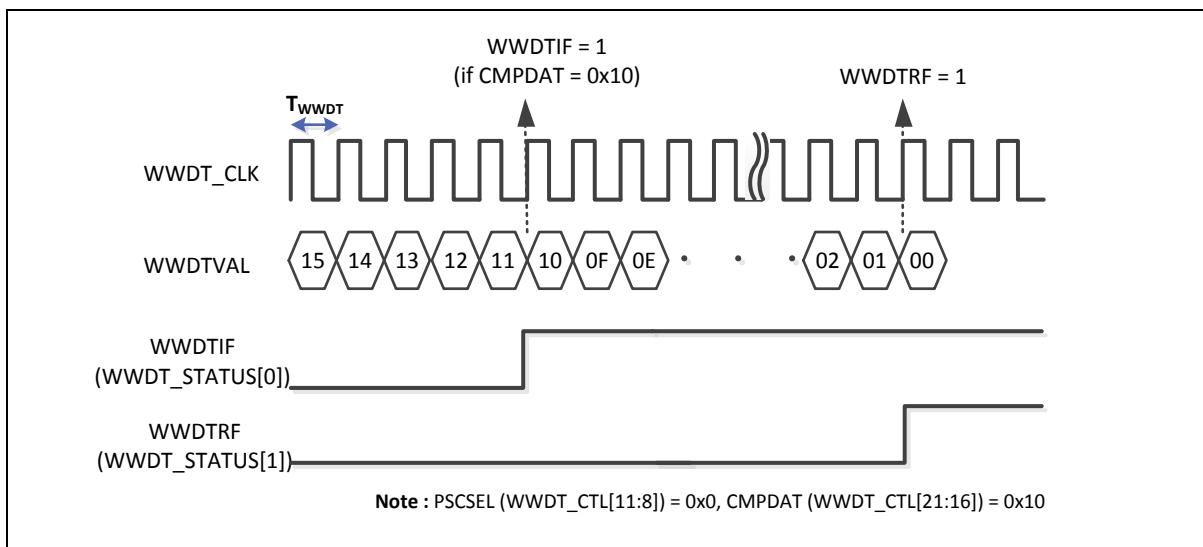


图 6.9-6 WWDT 中断和复位信号

#### 6.9.5.4 WWDT 窗口设置限制

当用户对 WWDT\_RLDCNT 寄存器写 0x00005AA5 来重载 WWDT 的值到 0x3F 的时候，大概需要 3 个窗口看门狗定时器时钟来同步重载命令到实际执行重载操作。这意味着如果用户设置 PSCSEL (WWDT\_CTL[11:8]) 的值为 0000，计数器预分频值应该设置为 1，CMPDAT (WWDT\_CTL[21:16]) 寄存器的值必须大于 2；否则，当 WWDTIF 标志产生后，写 WWDT\_RLDCNT 来重载窗口看门狗定时器的值为 0x3F 操作无效，WWDTIF (WWDT\_STATUS[0]) 将会产生，窗口看门狗定时器系统复位将发生。WWDT CMPDAT 设置限制如下表。

如果 CMPDATA 设置为 0x3F 或 0x0，则中断不会发生。因为 WWDT 计数到 0x0 发生复位，所以当

CMPDATA 为 0x0 时中断不会发生。

PSCSEL	预分频值	有效 CMPDAT 值
0000	1	0x3 ~ 0x3E
0001	2	0x2 ~ 0x3E
其它	其它	0x1 ~ 0x3E

表 6.9-2 CMPDAT 设置限制

#### 6.9.5.5 WWDT ICE 调试

当 ICE 连接到 MCU，WWDT 计数器是否计数由ICEDEBUG (WWDT\_CTL[31])决定。ICEDEBUG默认值是0，当 CPU 被 ICE停住，WWDT 计数器停止计数。如果ICEDEBUG 置 1，WWDT 计数器会保持计数，无论CPU 被 ICE停住与否。

### 6.9.6 寄存器映射

R: 只读, W: 只写, R/W: 可读写

寄存器	偏移量	R/W	描述	复位值
<b>WWDT 基地址:</b>				
<b>WWDT_BA = 0x4004_0100</b>				
WWDT_RLDCNT	WWDT_BA+0x00	W	窗口看门狗定时器重载计数器寄存器	0x0000_0000
WWDT_CTL	WWDT_BA+0x04	R/W	窗口看门狗定时器控制寄存器	0x003F_0800
WWDT_STATUS	WWDT_BA+0x08	R/W	窗口看门狗定时器状态寄存器	0x0000_0000
WWDT_CNT	WWDT_BA+0x0C	R	窗口看门狗定时器计数器值寄存器	0x0000_003F

### 6.9.7 寄存器描述

#### WWDT\_RLDCNT 窗口看门狗定时器重载计数器寄存器

寄存器	偏移量	R/W	描述	复位值
WWDT_RLDCNT	WWDT_BA+0x00	W	窗口看门狗定时器重载计数器寄存器	0x0000_0000

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

位	描述	
[31:0]	RLDCNT	<p><b>窗口看门狗定时器重载计数器寄存器</b>          对此寄存器写0x00005AA5 将会重载窗口看门狗定时器计数器的值到0x3F.</p> <p><b>注意:</b>用户只能当窗口看门狗计数器的值在0到CMPDAT (WWDT_CTL[21:16])之间才可以写WWDT_RLDCNT寄存器来重载窗口看门狗计数器。如果窗口看门狗计数器的值大于CMPDAT 时写WWDT_RLDCNT寄存器，窗口看门狗定时器复位信号会立即产生。</p>

**WWDT\_CTL WWDT 控制寄存器**

寄存器	偏移量	R/W	描述	复位值
WWDT_CTL	WWDT_BA+0x04	R/W	WWDT 控制寄存器	0x003F_0800

注意:此寄存器仅可以当芯片上电或复位后写一次。

31	30	29	28	27	26	25	24	
ICEDEBUG	Reserved							
23	22	21	20	19	18	17	16	
Reserved		CMPDAT						
15	14	13	12	11	10	9	8	
Reserved				PSCSEL				
7	6	5	4	3	2	1	0	
Reserved						INTEN	WWDTEN	

位	描述
[31]	<b>ICEDEBUG</b> ICE 调试模式下窗口看门狗计数控制位 0 =ICE 调试模式下影响窗口看门狗定时器计数, 当CPU被ICE暂停后, 窗口看门狗向下计数器计数值将保持不变 1 = ICE 调试模式下不影响窗口看门狗定时器计数, 不管CPU是否被ICE暂停, 窗口看门狗定时器向下计数器将会保持继续计数。
[30:22]	<b>Reserved</b> 保留.
[21:16]	<b>CMPDAT</b> 窗口看门狗定时器窗口比较寄存器 设置此寄存器来调整有效重载窗口 <b>注意:</b> 当前窗口看门狗定时器计数器的值在0到CMPDAT之间时, 用户才可以写WWDT_RLDCNT来重载窗口看门狗定时器计数器的值。如果当前窗口看门狗定时器计数器的值大于CMPDAT时用户写WWDT_RLDCNT寄存器, 窗口看门狗定时器复位信号会立即产生。
[15:12]	<b>Reserved</b> 保留.
[11:8]	<b>PSCSEL</b> 窗口看门狗定时器计数器预分频周期选择 0000 =预分频为1; 最大定时溢出周期是1 * 64 * WWDT_CLK. 0001 =预分频为2; 最大定时溢出周期是2 * 64 * WWDT_CLK. 0010 =预分频为4; 最大定时溢出周期是4 * 64 * WWDT_CLK. 0011 =预分频为8; 最大定时溢出周期是8 * 64 * WWDT_CLK. 0100 =预分频为16; 最大定时溢出周期是16 * 64 * WWDT_CLK. 0101 =预分频为32; 最大定时溢出周期是32 * 64 * WWDT_CLK. 0110 =预分频为64; 最大定时溢出周期是64 * 64 * WWDT_CLK. 0111 =预分频为128; 最大定时溢出周期是128 * 64 * WWDT_CLK. 1000 =预分频为192; 最大定时溢出周期是192 * 64 * WWDT_CLK. 1001 =预分频为256; 最大定时溢出周期是256 * 64 * WWDT_CLK. 1010 =预分频为384; 最大定时溢出周期是384 * 64 * WWDT_CLK.

		1011 = 预分频为512; 最大定时溢出周期是 $512 * 64 * \text{WWDT\_CLK}$ . 1100 = 预分频为768; 最大定时溢出周期是 $768 * 64 * \text{WWDT\_CLK}$ . 1101 = 预分频为1024; 最大定时溢出周期是 $1024 * 64 * \text{WWDT\_CLK}$ . 1110 = 预分频为1536; 最大定时溢出周期是 $1536 * 64 * \text{WWDT\_CLK}$ . 1111 = 预分频为2048; 最大定时溢出周期是 $2048 * 64 * \text{WWDT\_CLK}$ .
[7:2]	<b>Reserved</b>	保留.
[1]	<b>INTEN</b>	<b>窗口看门狗定时器中断使能位</b> 如果该位被使能, 如果有窗口看门狗定时器计数器比较匹配中断信号产生, 就会通知CPU 0 = 窗口看门狗定时器计数器比较匹配中断禁止 1 = 窗口看门狗定时器计数器比较匹配中断使能
[0]	<b>WWDTEN</b>	<b>窗口看门狗定时器使能控制位</b> 设置该位来使能窗口看门狗定时器计数器计数 0 = 窗口看门狗定时器计数器停止. 1 = 窗口看门狗定时器计数器计数开始

WWDT\_STATUS 窗口看门狗定时器状态寄存器

寄存器	偏移量	R/W	描述	复位值
WWDT_STAT US	WWDT_BA+0x08	R/W	窗口看门狗定时器状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved						WWDTRF	WWDTIF

位	描述	
[31:2]	Reserved	保留.
[1]	WWDTRF	<p>窗口看门狗定时器溢出复位标志 该位指示系统是否已被窗口看门狗定时器溢出复位 0 = 窗口看门狗定时器没有发生复位 1 = 窗口看门狗定时器发生了复位 注意：该位是写1清零</p>
[0]	WWDTIF	<p>窗口看门狗定时器比较匹配中断标志 该位指示当WWDT 的计数值与CMPDAT (WWDT_CTL[21:16])匹配时，窗口看门狗定时器的比较匹配中断状态标志 0 = 无影响 1 = 窗口看门狗定时器计数器的值与CMPDAT寄存器的值匹配 注意：该位是写1清零</p>

WWDT\_CNT 窗口看门狗定时器计数器值寄存器

寄存器	偏移量	R/W	描述	复位值
WWDT_CNT	WWDT_BA+0x0C	R	窗口看门狗定时器计数器值寄存器	0x0000_003F

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved		CNTDAT					

位	描述	
[31:6]	Reserved	保留.
[5:0]	CNTDAT	窗口看门狗定时器计数器值 CNTDAT将会被持续更新，以监视6位向下计数器的值

## 6.10 RTC 实时时钟

### 6.10.1 概述

实时时钟 (RTC) 控制器用于记录实时时间及日历等信息。RTC控制器支持可配置的时间节拍和闹钟定时中断。时间及日历等信息的表示格式为BCD 码。可对外接晶振的频率精度进行数字频率补偿。

### 6.10.2 特性

- 支持时间计数（秒，分，时）和日历计数（日，月，年），用户可以通过访问寄存器RTC\_TIME和RTC\_CAL查看时间及日历。
- 可设定闹钟时间（秒，分，时）和日历（日，月，年），参看寄存器RTC\_TALM 和 RTC\_CALM。
- 可设定闹钟时间（秒，分，时）和日历（日，月，年）的掩码使能功能,参看RTC\_TAMSK 和 RTC\_CAMSK寄存器。
- 可选择12-小时或 24-小时制式,参看RTC\_CLKFMT 寄存器。
- 支持闰年自动识别,参看RTC\_LEAPYEAR寄存器。
- 支持周内日期计数,参看RTC\_WEEKDAY寄存器。
- 支持RTC时钟源频率补偿功能, 参看寄存器RTC\_FREQADJ 。
- 所有时间、日期的数据格式为 BCD 码。
- 支持周期RTC时间节拍中断，提供 8个周期选项供选择，分别为：1/128, 1/64, 1/32, 1/16, 1/8, 1/4, 1/2 及 1 秒。
- 支持 RTC 定时节拍和闹钟定时中断。
- 支持1 Hz时钟输出。
- 支持RTC中断从空闲模式或掉电模式下唤醒芯片。
- 通过RTC\_DSTCTL寄存器，支持夏令时软件控制。
- 支持3对动态循环tamper 管脚 或 6 个独立 tamper 管脚。
- 提供80字节的备用寄存器用于存储用户信息，并提供tamper检测脚用于清除备用寄存器中内容。

## 6.10.3 框图

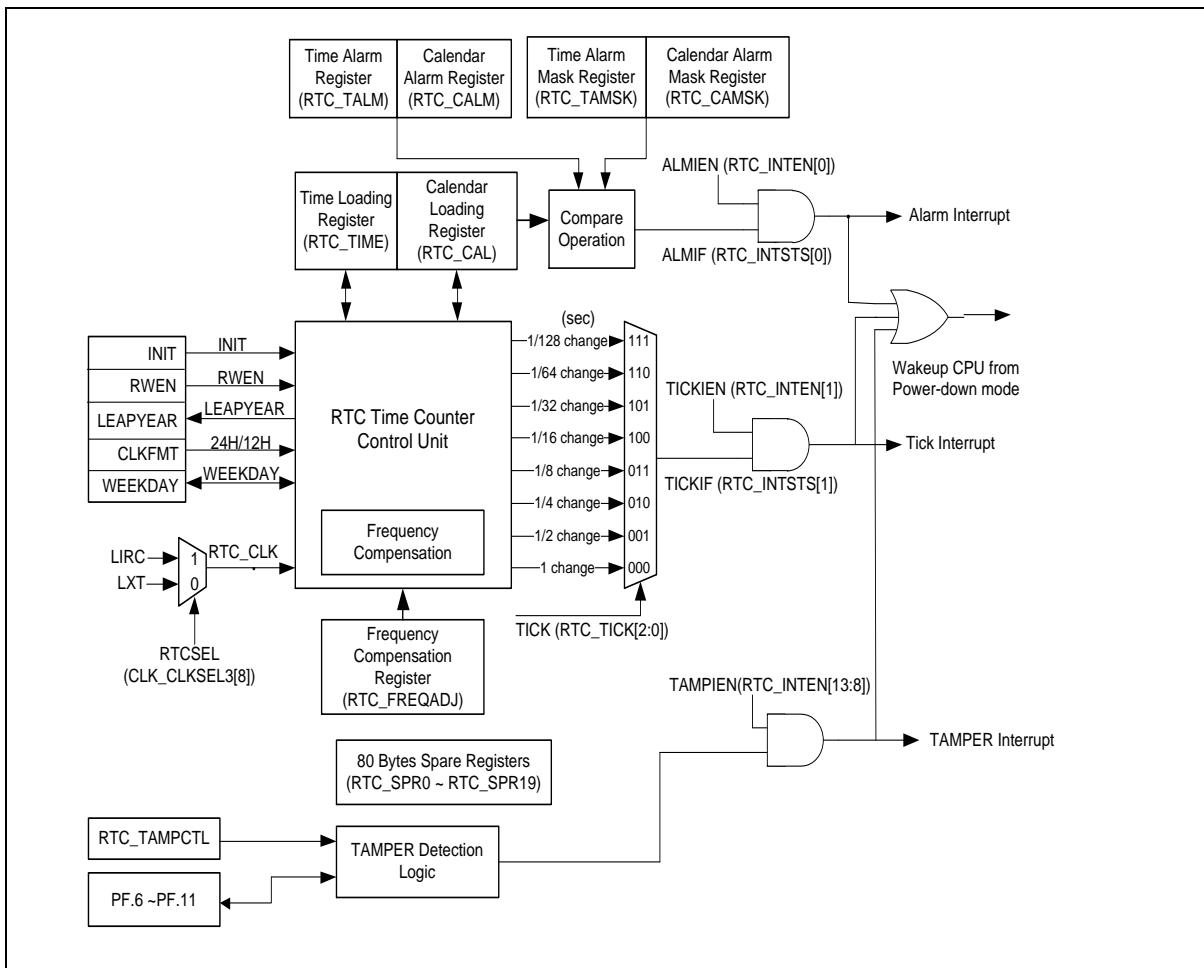


图 6.10-1 RTC 框图

## 6.10.4 基本配置

- 时钟源配置
  - RTC控制器时钟源通过RTCCKEN (APBCLK0[1])使能， RTC时间计数器时钟源通过 CLK\_CLKSEL3[8] 选择LXT 或 LIRC。
- 管脚配置

组	管脚名称	GPIO	MFP
X32	X32_OUT	PF.4	MFP10
	X32_IN	PF.5	MFP10
TAMPER0	TAMPER0	PF.6	MFP10
TAMPER1	TAMPER1	PF.7	MFP10
TAMPER2	TAMPER2	PF.8	MFP10

TAMPER3	TAMPER3	PF.9	MFP10
TAMPER4	TAMPER4	PF.10	MFP10
TAMPER5	TAMPER5	PF.11	MFP10

### 6.10.5 功能描述

#### 6.10.5.1 RTC 初始化

当 RTC 模块上电后，RTC 处于 reset 状态。用户需要写入数据 (0x a5eb1357) 到 RTC 初始化寄存器 INIT RTC\_INIT[31:0] 中，让 RTC 离开 reset 状态。一旦 INIT RTC\_INIT[31:0] 被写入 0xa5eb1357，RTC 将会永远处于 Active 状态。用户可以读 INIT[0](RTC\_INIT[0]) 来判断 RTC 是处于 Active 状态或是 reset 状态。

#### 6.10.5.2 RTC 读/写使能

如果 RWENF (RTC\_RWEN[16]) 位置 1，意味着 RTC 寄存器是可读写的。当在 1120 个 PCLK 周期内执行写 RTC 寄存器命令超过 6 次，RTCBUSY (RTC\_RWEN[24]) 标志会被置 1，RWENF (RTC\_RWEN[16]) 会被清 0。当 RWENF 是 1 和 0 时，RTC 寄存器访问属性如下表：

寄存器	INIR = 0	RWENF = 1	RWENF = 0 或 RTCBUSY=1
RTC_INIT	available	R/W	R/W
RTC_RWEN	available	R/W	R/W
RTC_FREQADJ	available	R/W	R
RTC_TIME	Not available	R/W	R
RTC_CAL	Not available	R/W	R
RTC_CLKFMT	Not available	R/W	R
RTC_WEEKDAY	Not available	R/W	R
RTC_TALM	Not available	R/W	R
RTC_CALM	Not available	R/W	R
RTC_LEAPYEAR	Not available	R	R
RTC_INTEN	available	R/W	R/W
RTC_INTSTS	available	R/W	R/W
RTC_TICK	Not available	R/W	R
RTC_TAMSK	Not available	R/W	R
RTC_CAMSK	Not available	R/W	R
RTC_SPRCTL	available	R/W	Not available
RTC_SPRx	available	R/W	Not available
RTC_LXTCTL	available	R/W	R/W
RTC_GPIOCTL0	available	R/W	R/W
RTC_GPIOCTL1	available	R/W	R/W
RTC_DSTCTL	Not available	R/W	R
RTC_TAMPCTL	Not available	R/W	R
RTC_TAMPSEED	Not available	R/W	R

表 6.10-1 RTC 读/写 使能

### 6.10.5.3 频率补偿

RTC\_FREQADJ 寄存器允许用户对输入时钟做数字补偿，请参考下面例子和公式写32k晶振的实际频率到RTC\_FREQADJ寄存器。下面是频率高于或低于32768 Hz补偿的例子。

#### 例 1：

频率计数器测量: 32773.65 Hz (> 32768 Hz)

$$\text{FREQADJ} = (32768 * 0x200000) / 32773.65 = 0x1FFE96$$

#### 例 2：

频率计数器测量: 32763.25 Hz (< 32768 Hz)

$$\text{FREQADJ} = (32768 * 0x200000) / 32763.25 = 0x200130$$

**注意:** 当补偿没有执行， RTC\_FREQADJ 寄存器是值默认值(0x0020\_0000)。用户可以通过时钟输出功能使用频率计数器来测量RTC时钟源，同时用户可以使用时钟输出功能检查RTC的时钟补偿结果。

### 6.10.5.4 时间和日历计数器

RTC\_TIME 和 RTC\_CAL 用于加载实时时间和日期。 RTC\_TALM 和 RTC\_CALM 用于设置闹钟时间和日期。

#### 6.10.5.5 12/24 小时制选择

设置 24HEN (RTC\_CLKFMT[0]) 选择 12/24 小时时标制式。当 RTC 运行在 12 小时时标制模式时，RTC\_TIME[21] (TENHR[1:0]) 中的高位代表 AM/PM 提示符）。（如果 RTC\_TIME[21]=1，表示目前为 PM 时间，RTC\_TIME[21]=0，表示目前为 AM 时间）。下图为 12/24 小时制选择 RTC\_TIME 映射表。

注：小时信息写入到RTC_TIME[21:16],信息格式为BCD格式			
24-小时制 (24HEN = 1)		12-小时制 (PM 时间 + 0x20) (24HEN = 0) (PM 时间 + 0x20)	
0x00 (AM12)	0x12 (PM12)	0x12 (AM12)	0x32 (PM12)
0x01 (AM01)	0x13 (PM01)	0x01 (AM01)	0x21 (PM01)
0x02 (AM02)	0x14 (PM02)	0x02 (AM02)	0x22 (PM02)
0x03 (AM03)	0x15 (PM03)	0x03 (AM03)	0x23 (PM03)
0x04 (AM04)	0x16 (PM04)	0x04 (AM04)	0x24 (PM04)
0x05 (AM05)	0x17 (PM05)	0x05 (AM05)	0x25 (PM05)
0x06 (AM06)	0x18 (PM06)	0x06 (AM06)	0x26 (PM06)
0x07 (AM07)	0x19 (PM07)	0x07 (AM07)	0x27 (PM07)
0x08 (AM08)	0x20 (PM08)	0x08 (AM08)	0x28 (PM08)
0x09 (AM09)	0x21 (PM09)	0x09 (AM09)	0x29 (PM09)
0x10 (AM10)	0x22 (PM10)	0x10 (AM10)	0x30 (PM10)
0x11 (AM11)	0x23 (PM11)	0x11 (AM11)	0x31 (PM11)

表 6.10-2 12/24 小时制选择

### 6.10.5.6 星期计数器

RTC控制器提供一周日期寄存器WEEKDAY (RTC\_WEEKDAY [2: 0]), 其值由0至6, 用于表示周日至周六。

### 6.10.5.7 时间周期节拍中断

通过设置TICK (RTC\_TICK [2:0]) 来选择周期中断, 周期中断有8个选项 1/128, 1/64, 1/32, 1/16, 1/8, 1/4, 1/2 以及 1 秒。当TICKIEN (RTC\_INTEN[1])被置 1, 周期中断使能后, MCU根据RTC\_TICK[2:0]寄存器内设定的值周期性的发生中断。

### 6.10.5.8 闹钟中断

当RTC\_TIME 和RTC\_CAL中的值等于RTC\_TALM和RTC\_CALM中的设定值, 如果此时闹钟中断已设为使能 (ALMIEN (RTC\_INTEN[0])=1), 则闹钟中断标志ALMIF (RTC\_INTSTS[0])被置位同时发出闹钟中断请求。

RTC控制器提供时间屏蔽寄存器 (RTC\_TAMSK) 和日历闹钟屏蔽寄存器 (RTC\_CAMSK) , 用于屏蔽指定数字并产生周期中断而无需改动每个闹钟中断服务程序中的闹钟匹配条件 (匹配条件根据RTC\_TALM 和 RTC\_CAL中配置决定) 。

### 6.10.5.9 夏令时

RTC 控制器也提供 RTC\_DSTCTL 寄存器来存储夏令时应用的控制设置。用户可以读DSBAK(RTC\_DSTCTL[2]) 值来检查当前RTC时间日期计数器是否运行在夏令时模式还是正常模式。

### 6.10.5.10 1 Hz 时钟输出

RTC控制器提供1Hz 时钟输出到 CLK0 功能管脚。用户可以设置 CLK1HZEN (CLK\_CLKOCTL[6]) 为 1 且使能 RTC, 1Hz 时钟会输出到 CLK0 功能管脚。

### 6.10.5.11 应用指南

1. RTC\_TALM, RTC\_CALM, RTC\_TIME 和 RTC\_CAL寄存器中数据格式为BCD 格式。
2. 用户必须保证载入值为有效合理值。例如, RTC\_CAL = 201a (年), 13 (月), 00 (日), 或 RTC\_CAL 与RTC\_WEEKDAY 不匹配, 等等。
3. 在RTC\_CAL 和 RTC\_CALM 中, 仅 2 位 BCD 码用于指示“年”。目前仅支持20xy年份, 不支持19xy 或 21xy年。
4. 12小时制时间设定范例  
如果RTC时间为PM12:59:30, RTC\_TIME设定为:
5. HOUR:  
RTC\_TIME[21:16]: 0x32 (0x12+0x20), 可表示为: TENHR (RTC\_TIME[21:20]) is 0x3, HR (RTC\_TIME[19:16]) is 0x2.
6. MIN:  
RTC\_TIME[14:8]: 0x59 可表示为: TENMIN (RTC\_TIME[14:12]) is 0x5, MIN (RTC\_TIME[11:8]) is 0x9.
7. SEC:  
RTC\_TIME[6:0]: 0x30 , 可表示为: TENSEC (RTC\_TIME[6:4]) is 0x3, SEC (RTC\_TIME[3:0]) is 0x0.
8. 表 6.10-3 表示在内核上电和电池初次上电后的寄存器值。

寄存器	复位状态
RTC_INIT	0

RTC_RWEN	0
RTC_CAL	15/8/8 (年/月/日)
RTC_TIME	00:00:00 (时 : 分 : 秒)
RTC_CALM	00/00/00 (年/月/日)
RTC_TALM	00:00:00 (时 : 分 : 秒)
RTC_CLKFMT	1 (24小时模式)
RTC_WEEKDAY	6 (星期六)
RTC_INTEN	0
RTC_INTSTS	0
RTC_LEAPYEAR	0
RTC_TICK	0
RTC_DSTCTL	0

表 6.10-3 上电后寄存器值

9. 下表列出寄存器属于内核电源域，其他属于电池电源域。

寄存器	电源域
RTC_RWEN	内核电源域
RTC_INTEN	内核电源域
RTC_INTSTS	内核电源域
Others	电池电压域

表 6.10-4 寄存器电源域

#### 6.10.5.12 备用寄存器和盗取(Tamper)事件检测

RTC模块提供了80字节的备用寄存器用于存储用户的重要信息，这些寄存器位于RTC时钟域，在向20个中任意一个备用寄存器(RTC\_SPR0 ~ RTC\_SPR19)写值前，用户必须使能寄存器SPRRWEN(RTC\_SPRCTL[2])。

当侦测到信号变化满足RTC\_TAMPCTL中定义的条件，tamper中断标志会TAMPxIF(RTC\_INTSTS[13:8])置位，同时备用寄存器(RTC\_SPR0 ~ RTC\_SPR19)中的80个字节数据将被硬件自动清除，以防保密数据被盗取。当前RTC时间和日期会加载到RTC\_TAMPTIME 和 RTC\_TAMPCAL寄存器，这些值仅可以被自动清除或当所有的TAMPxIF都被清0时再次更新。如果TAMPxIF置1，且tamper侦测中断使能，会产生中断到NVIC。

RTC支持3对动态循环tamper管脚或6个独立的tamper管脚。DYNRATE (RTC\_TAMPCTL[7:5])决定一个参考位的持续时间，如图 6.10-2所示。在表 6.10-5中，设置DYNSSRC (RTC\_TAMPCTL[3:2])可以选择侦测信号是硬件随机值产生器产生的新值或之前的随机值或用户定义的种子值(RTC\_TAMPSEED[31:0])，用于动态循环tamper管脚。设置DYN1ISS (RTC\_TAMPCTL[0])为1可以选择对1侦测源来自tamper 0，设置DYN12SS (RTC\_TAMPCTL[1]) 为1可以选择对2侦测源也来自tamper 0。TAMPxLV (RTC\_TAMPCTL[4x+9], x=0, 1, ..., 5)取决于用于独立tamper侦测的TAMPERx管脚的电平属性。Tamper控制效果如表 6.10-6 和 表 6.10-7所示。

DYNRATE	Duration	RTC_CLK	TAMPERx_output	Ref_B[0]	Ref_B[1]	Ref_B[2]
000	1/32 sec			X		
001	1/16 sec				X	
010	1/8 sec					X
011	1/4 sec					
100	1/2 sec					
101	1 sec					
110	2 sec					
111	4 sec					
$x=0, 2 \text{ and } 4.$						

图 6.10-2 动态率定义

DYNPRxEN	DYNSRC[1:0]	侦测信号描述
0	x	TAMPERn 是静态 tamper 侦测 ( $n=0 \sim 5$ )
1	00	当目前动态样本使用完后，产生新的随机值作为参考样本
1	01	当目前的动态样本使用完后，不会产生另一组新的随机数值，供动态样本使用，而是重复使用之前的动态样本
1	10	当目前动态样本使用完后，产生新的随机值作为参考样本
1	11	当动态样本使用完后，重复使用用户输入到种子值作为参考样本

表 6.10-5 动态样本源选择

Tamper 配置			Tamper 控制位效果						
DYNPROEN	TAMPER0EN	TAMPER1EN	TAMP0LV	TAMP1LV	TAMP0DBE N	TAMP1DBE N	TAMPER 0	TAMPER 1	
0	0	0	-	-	-	-	-	-	-
0	0	1	-	V	-	V	-	-	Static
0	1	0	V	-	V	-	Static	X	
0	1	1	V	V	V	V	Static	Static	
1	x	0	-	-	-	-	Dynamic Out	-	
1	x	1	-	-	-	-	Dynamic Out	Dynamic In	

表 6.10-6 Tamper 控制位效果位对 0 对的影响

Tamper 配置				Tamper 控制位效果						
DYNPRxE N	DYNxISS	TAMPERn EN	TAMPEm REN	TAMPnLV	TAMPmLV	TAMPnDBE N	TAMPmDB EN	TAMPER n	TAMPER m	
0	x	0	0	-	-	-	-	-	-	-
0	x	0	1	-	V	-	V	-	Static	
0	x	1	0	V	-	V	-	Static	-	
0	x	1	1	V	V	V	V	Static	Static	

1	0	x	0	-	-	-	-	Dynamic Out	-	
1	0	x	1	-	-	-	-	Dynamic Out	Dynamic In	
1	1	0	1	-	-	-	-	-	Dynamic Input form TAMPER 0	
1	1	1	1	V	V	-	-	Static	Dynamic Input form TAMPER 0	
x= 1 and 2		n= 2 and 4, m= 3 and 5								

表 6.10-7 Tamper 控制位效果位对 1 和 2 对的影响

**静态 Tamper 编程顺序范例.**

1. 清 TAMPxIF (RTC\_INTSTS[x+8], x=0, 1, ..., 5)
2. 设置 TAMPxLV (RTC\_TAMPCTL[4x+9], x=0, 1, ..., 5) 和 TAMPxDBEN (RTC\_TAMPCTL[4x+10], x=0, 1, ..., 5).
3. 使能 TAMPxEN (RTC\_TAMPCTL[4x+8], x=0, 1, ..., 5).

**动态 Tamper 编程顺序范例.**

1. 清 TAMPxIF (RTC\_INTSTS[x+8], x=0, 1, ..., 5)
2. 填 SEED (RTC\_TAMPSEED[31:0]).
3. 设置 DYNSRC (RTC\_TAMPCTL[3:2]), DYNxISS (RTC\_TAMPCTL[x], x=0, 1) and DYNRATE (RTC\_TAMPCTL[7:5]).
4. 使能 DYNPRxEN (RTC\_TAMPCTL[8x+16], x=0, 1, 2).
5. 使能 TAMPxEN (RTC\_TAMPCTL[4x+8], x=0, 1, ..., 5).
6. 设置 SEEDRLD (RTC\_TAMPCTL[4]).

**6.10.5.13 GPIO 备用功能**

如果PF.4/X32O 和 PF.5/X32I脚没有用于连接低速32K 晶振，这两根管脚可当作GPIO使用。CTLSEL0 (RTC\_GPIOCTL0[3])用于选择PF.4/X32O是被RTC模块还是GPIO模块控制，PF.5/X32I 管脚由 CTLSEL1 in RTC\_GPIOCTL0[11]控制。下图表示备用I/O控制图。

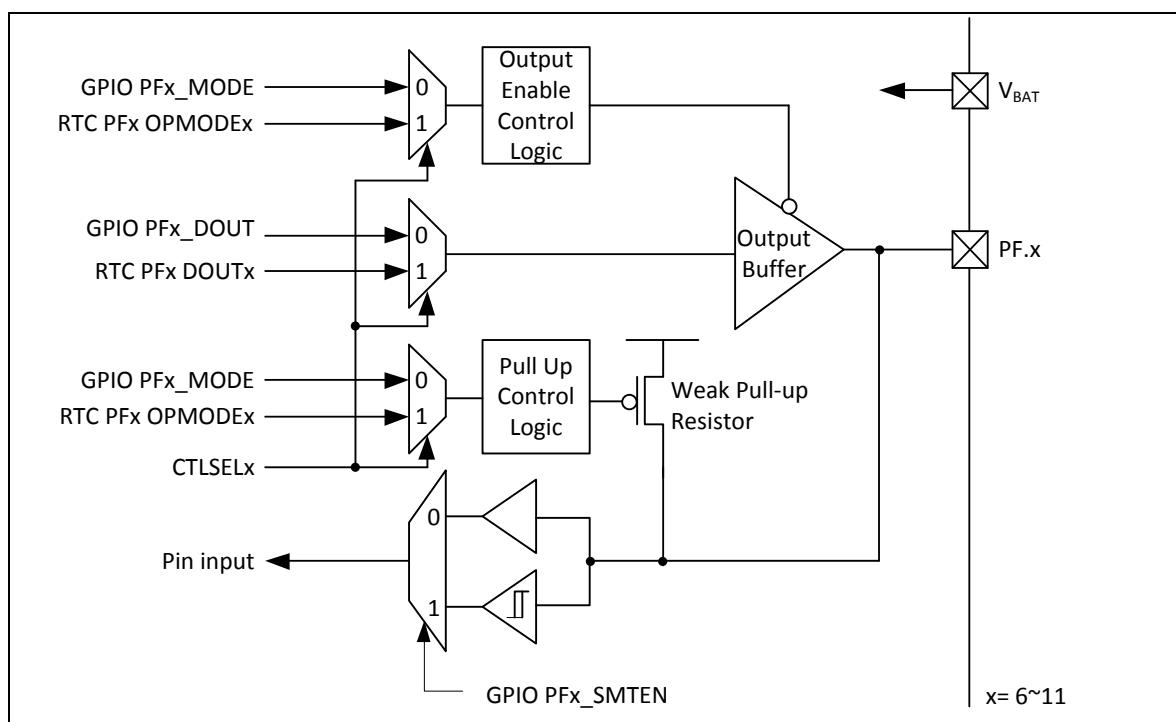


图 6.10-3 备用 I/O 控制图

### 6.10.6 寄存器映射

R: 只读, W: 只写, R/W: 可读写

寄存器	偏移地址	R/W	描述	复位值
<b>RTC 基地址:</b>				
<b>RTC_BA = 0x4004_1000</b>				
RTC_INIT	RTC_BA+0x00	R/W	RTC初始化寄存器	0x0000_0000
RTC_RWEN	RTC_BA+0x04	R/W	RTC访问使能寄存器	0x0000_0000
RTC_FREQADJ	RTC_BA+0x08	R/W	RTC频率补偿寄存器	0x0020_0000
RTC_TIME	RTC_BA+0x0C	R/W	时间载入寄存器	0x0000_0000
RTC_CAL	RTC_BA+0x10	R/W	日历载入寄存器	0x0015_0808
RTC_CLKFMT	RTC_BA+0x14	R/W	时间制式(时标)选择寄存器	0x0000_0001
RTC_WEEKDAY	RTC_BA+0x18	R/W	星期寄存器	0x0000_0006
RTC_TALM	RTC_BA+0x1C	R/W	时间闹钟寄存器	0x0000_0000
RTC_CALM	RTC_BA+0x20	R/W	日历闹钟寄存器	0x0000_0000
RTC_LEAPYEAR	RTC_BA+0x24	R	闰年指示寄存器	0x0000_0000
RTC_INTEN	RTC_BA+0x28	R/W	RTC中断使能寄存器	0x0000_0000
RTC_INTSTS	RTC_BA+0x2C	R/W	RTC中断指示寄存器	0x0000_0000
RTC_TICK	RTC_BA+0x30	R/W	RTC时钟节拍寄存器	0x0000_0000
RTC_TAMSK	RTC_BA+0x34	R/W	RTC时间闹钟屏蔽寄存器	0x0000_0000
RTC_CAMSK	RTC_BA+0x38	R/W	RTC日历闹钟屏蔽寄存器	0x0000_0000
RTC_SPRCTL	RTC_BA+0x3C	R/W	RTC备用功能控制寄存器	0x0000_0000
RTC_SPR0	RTC_BA+0x40	R/W	RTC备用寄存器0	0x0000_0000
RTC_SPR1	RTC_BA+0x44	R/W	RTC备用寄存器1	0x0000_0000
RTC_SPR2	RTC_BA+0x48	R/W	RTC 备用寄存器2	0x0000_0000
RTC_SPR3	RTC_BA+0x4C	R/W	RTC 备用寄存器3	0x0000_0000
RTC_SPR4	RTC_BA+0x50	R/W	RTC 备用寄存器4	0x0000_0000
RTC_SPR5	RTC_BA+0x54	R/W	RTC 备用寄存器5	0x0000_0000
RTC_SPR6	RTC_BA+0x58	R/W	RTC 备用寄存器6	0x0000_0000
RTC_SPR7	RTC_BA+0x5C	R/W	RTC 备用寄存器7	0x0000_0000
RTC_SPR8	RTC_BA+0x60	R/W	RTC备用寄存器8	0x0000_0000

<b>RTC_SPR9</b>	RTC_BA+0x64	R/W	RTC 备用寄存器 9	0x0000_0000
<b>RTC_SPR10</b>	RTC_BA+0x68	R/W	RTC 备用寄存器 10	0x0000_0000
<b>RTC_SPR11</b>	RTC_BA+0x6C	R/W	RTC 备用寄存器 11	0x0000_0000
<b>RTC_SPR12</b>	RTC_BA+0x70	R/W	RTC 备用寄存器 12	0x0000_0000
<b>RTC_SPR13</b>	RTC_BA+0x74	R/W	RTC 备用寄存器 13	0x0000_0000
<b>RTC_SPR14</b>	RTC_BA+0x78	R/W	RTC 备用寄存器 14	0x0000_0000
<b>RTC_SPR15</b>	RTC_BA+0x7C	R/W	RTC 备用寄存器 15	0x0000_0000
<b>RTC_SPR16</b>	RTC_BA+0x80	R/W	RTC 备用寄存器 16	0x0000_0000
<b>RTC_SPR17</b>	RTC_BA+0x84	R/W	RTC 备用寄存器 17	0x0000_0000
<b>RTC_SPR18</b>	RTC_BA+0x88	R/W	RTC 备用寄存器 18	0x0000_0000
<b>RTC_SPR19</b>	RTC_BA+0x8C	R/W	RTC 备用寄存器 19	0x0000_0000
<b>RTC_LXTCTL</b>	RTC_BA+0x100	R/W	RTC 32.768 kHz 晶振控制寄存器	0x0000_000E
<b>RTC_GPIOCTL_0</b>	RTC_BA+0x104	R/W	RTC GPIO 控制寄存器 0	0x0000_0000
<b>RTC_GPIOCTL_1</b>	RTC_BA+0x108	R/W	RTC GPIO 控制寄存器 1	0x0000_0000
<b>RTC_DSTCTL</b>	RTC_BA+0x110	R/W	RTC 夏令时控制寄存器	0x0000_0000
<b>RTC_TAMPCTL</b>	RTC_BA+0x120	R/W	RTC Tamper 管脚控制寄存器	0x0000_0000
<b>RTC_TAMPSEED</b>	RTC_BA+0x128	R/W	RTC Tamper 动态种子寄存器	0x0000_0000
<b>RTC_TAMPTIME</b>	RTC_BA+0x130	R	RTC Tamper 时间寄存器	0x0000_0000
<b>RTC_TAMPCA_L</b>	RTC_BA+0x134	R	RTC Tamper 日历寄存器	0x0000_0000

## 6.10.7 寄存器描述

RTC\_INIT RTC 初始化寄存器

寄存器	偏移地址	R/W	描述	复位值
RTC_INIT	RTC_BA+0x00	R/W	RTC 初始化寄存器	0x0000_0000

31	30	29	28	27	26	25	24
INIT							
23	22	21	20	19	18	17	16
INIT							
15	14	13	12	11	10	9	8
INIT							
7	6	5	4	3	2	1	0
INIT							INIT/ACTIVE

位	描述	
[31:1]	<b>INIT[31:1]</b>	<b>RTC 初始化(只写)</b> 当 RTC 模块上电后，RTC 处于复位状态。用户需要写入数据 (0xa5eb1357) 到 INIT 使得 RTC 离开复位状态。一旦 INIT 被写入 0xa5eb1357，RTC 将会一直处于非复位状态。 INIT寄存器的这些位只能被写，如果读的话，总是为0
[0]	<b>INIT[0]/ACTIVE</b>	<b>RTC 激活状态 (只读)</b> 0 = RTC 在复位状态 1 = RTC 在正常激活状态

**RTC RWEN RTC 访问使能寄存器**

寄存器	偏移地址	R/W	描述	复位值
RTC_RWEN	RTC_BA+0x04	R/W	RTC 访问使能寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							RTCBUSY
23	22	21	20	19	18	17	16
Reserved							RWENF
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved							

位	描述	
[31:25]	<b>Reserved</b>	保留.
[24]	<b>RTCBUSY</b>	<b>RTC写忙标志</b> 该位表示 RTC寄存器是否可写 0: RTC 寄存器可写. 1: RTC 寄存器不可写, RTC 在忙状态. <b>注意:</b> 当在1120个 PCLK周期内执行写RTC寄存器命令超过6次, RTCBUSY 标志会被置位。
[23:17]	<b>Reserved</b>	保留.
[16]	<b>RWENF</b>	<b>RTC 寄存器访问使能标志 (只读)</b> 0 = 禁止RTC寄存器读/写访问 1 = 使能RTC寄存器读/写访问 <b>注意:</b> 当 RTCBUSY 是 1 期间, RWENF 会被屏蔽为 0 。第一次打开 RTCCKEN (CLK_APBCLK[1]) 也会。
[15:0]	<b>Reserved</b>	保留.

## RTC\_FREQADJ RTC 频率补偿寄存器

寄存器	偏移地址	R/W	描述	复位值
RTC_FREQADJ	RTC_BA+0x08	R/W	RTC 频率补偿寄存器	0x0020_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
FREQADJ							
7	6	5	4	3	2	1	0
FREQADJ							

位	描述	
[31:22]	Reserved	保留.
[21:0]	FREQADJ	<p>频率补偿寄存器</p> <p>用户必须获取用于 RTC 应用的LXT 的实际频率。</p> <p><math>FCR = 0x200000 * (32768 / LXT \text{ 频率})</math>.</p> <p><b>注意:</b> 该公式仅当RTC时钟源来自LXT ( RTCSEL (CLK_CLKSEL3[8]) 为 0) 时适用。</p>

## RTC TIME RTC时间载入寄存器

寄存器	偏移地址	R/W	描述	复位值
RTC_TIME	RTC_BA+0x0C	R/W	RTC 时间载入寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved		TENHR			HR		
15	14	13	12	11	10	9	8
Reserved	TENMIN			MIN			
7	6	5	4	3	2	1	0
Reserved	TENSEC			SEC			

位	描述	
[31:22]	Reserved	保留.
[21:20]	TENHR	小时的十位数(0~2) 当 RTC采用12-小时制时， RTC_TIME[21]( TENHR[1:0]的高位)表示 AM/PM 提示符.(如果 IRTC_TIME[21]=1,表示为PM时间.)
[19:16]	HR	小时的个位数(0~9)
[15]	Reserved	保留.
[14:12]	TENMIN	分钟的十位数(0~5)
[11:8]	MIN	分钟的个位数(0~9)
[7]	Reserved	保留.
[6:4]	TENSEC	秒钟的十位数(0~5)
[3:0]	SEC	秒钟的个位数(0~9)

注:

1. RTC\_TIME 为 BCD 计数方式， RTC 不会对载入值的合理性进行检测。
2. 括号内列出载入值的合理范围。.
3. 当写RTC\_FREQADJ , RTC\_TIME, RTC\_CAL, RTC\_WEEKDAY时， FREQADJ'的计数器会复位以开始补偿。RTC时间会重新开始。

## RTC CAL RTC 日历载入寄存器

寄存器	偏移地址	R/W	描述	复位值
RTC_CAL	RTC_BA+0x10	R/W	RTC 日历载入寄存器	0x0015_0808

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
TENYEAR				YEAR			
15	14	13	12	11	10	9	8
Reserved			TENMON	MON			
7	6	5	4	3	2	1	0
Reserved		TENDAY		DAY			

位	描述	
[31:24]	Reserved	保留
[23:20]	TENYEAR	年日历的十位数(0~9)
[19:16]	YEAR	年日历的个位数(0~9)
[15:13]	Reserved	保留
[12]	TENMON	月日历的十位数(0~1)
[11:8]	MON	月日历的个位数(0~9)
[7:6]	Reserved	保留
[5:4]	TENDAY	天日历的十位数(0~3)
[3:0]	DAY	天日历的个位数(0~9)

注意:

1. RTC\_CAL为BCD计数格式，RTC不会检测载入值的合理性。
2. 括号内列出载入值的合理范围。

RTC\_CLKFMT RTC 时间制式选择寄存器

寄存器	偏移地址	R/W	描述	复位值
RTC_CLKFMT	RTC_BA+0x14	R/W	RTC时间制式选择寄存器	0x0000_0001

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved							24HEN

位	描述	
[31:1]	Reserved	保留.
[0]	24HEN	<p><b>24-小时 / 12-小时 时间制式选择</b></p> <p>用于表示RTC_TIME 和 RTC_TALM的时间为 24-小时 或 12-小时制式</p> <p>0 =选择 12-小时制带AM 和 PM指示 1 =选择 24-小时制</p>

## RTC WEEKDAY RTC 星期寄存器

寄存器	偏移地址	R/W	描述	复位值
RTC_WEEKDAY	RTC_BA+0x18	R/W	RTC 星期寄存器	0x0000_0006

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved					WEEKDAY		

位	描述	
[31:3]	Reserved	保留.
[2:0]	WEEKDAY	<p>星期寄存器</p> <p>000 = 星期日      001 = 星期一      010 = 星期二      011 = 星期三      100 = 星期四      101 = 星期五.      110 = 星期六      111 = 保留</p>

**RTC\_TALM RTC时间闹钟寄存器**

寄存器	偏移地址	R/W	描述	复位值
RTC_TALM	RTC_BA+0x1C	R/W	RTC时间闹钟寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved		TENHR			HR		
15	14	13	12	11	10	9	8
Reserved	TENMIN			MIN			
7	6	5	4	3	2	1	0
Reserved	TENSEC			SEC			

位	描述	
[31:24]	Reserved	保留
[21:20]	TENHR	闹钟中小时时间的十位数(0~2) 当 RTC采用12-小时制时, RTC_TIME[21]( TENHR[1:0]的高位)表示 AM/PM 提示符.(如果 RTC_TIME[21]=1,表示为PM时间.)
[19:16]	HR	闹钟中小时时间的个位数(0~9)
[15]	Reserved	保留
[14:12]	TENMIN	闹钟中分钟时间的十位数(0~5)
[11:8]	MIN	闹钟中分钟时间的个位数(0~9)
[7]	Reserved	保留.
[6:4]	TENSEC	闹钟中秒钟时间的十位数(0~5)
[3:0]	SEC	闹钟中秒钟时间的个位数(0~9)

注意:

1. RTC\_TALM 为 BCD 计数方式, RTC 不会对载入值的合理性进行检测。
2. 括号内列出载入值的合理范围。
3. 在 RTC 访问使能位RWENF (RTC\_RWEN[16])激活后, 该寄存器的值可以被读回。

**RTC CALM RTC日历闹钟寄存器**

寄存器	偏移地址	R/W	描述	复位值
RTC_CALM	RTC_BA+0x20	R/W	RTC 日历闹钟寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
TENYEAR				YEAR			
15	14	13	12	11	10	9	8
Reserved			TENMON	MON			
7	6	5	4	3	2	1	0
Reserved		TENDAY		DAY			

位	描述	
[31:24]	Reserved	保留
[23:20]	TENYEAR	闹钟中年日历的十位数(0~9)
[19:16]	YEAR	闹钟中年日历的个位数(0~9)
[15:13]	Reserved	保留.
[12]	TENMON	闹钟中月日历的十位数 (0~1)
[11:8]	MON	闹钟中月日历的个位数(0~9)
[7:6]	Reserved	保留.
[5:4]	TENDAY	闹钟中天日历的十位数(0~3)
[3:0]	DAY	闹钟中天日历的个位数(0~9)

注意:

1. RTC\_CALM 为 BCD 计数方式, RTC 不会对载入值的合理性进行检测。
2. 括号内列出载入值的合理范围。
3. 在 RTC 访问使能位RWENF (RTC\_RWEN[16])激活后, 该寄存器的值可以被读回。

## RTC LEAPYEAR RTC闰年指示寄存器

寄存器	偏移地址	R/W	描述	复位值
RTC_LEAPYEAR	RTC_BA+0x24	R	RTC 闰年指示寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved							LEAPYEAR

位	描述	
[31:1]	Reserved	保留.
[0]	LEAPYEAR	闰年指示寄存器（只读） 0 = 表示该年非闰年 1 = 表示该年为闰年

## RTC INTEN RTC 中断使能寄存器

寄存器	偏移地址	R/W	描述	复位值
RTC_INTEN	RTC_BA+0x28	R/W	RTC 中断使能寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved		TAMP5IEN	TAMP4IEN	TAMP3IEN	TAMP2IEN	TAMP1IEN	TAMPOIEN
7	6	5	4	3	2	1	0
Reserved						TICKIEN	ALMIEN

位	描述
[31:14]	<b>Reserved</b> 保留.
[13]	<b>TAMP5IEN</b> Tamper 5 或 对 2 中断使能位 设置 TAMP5IEN 为 1 也可以使能tamper 5 中断事件唤醒芯片功能 0 = Tamper 5 或对 2 中断禁止. 1 = Tamper 5或对 2 中断使能.
[12]	<b>TAMP4IEN</b> Tamper 4中断使能位 设置 TAMP4IEN 为 1 也可以使能tamper 4 中断事件唤醒芯片功能 0 = Tamper 4 中断禁止. 1 = Tamper 4 中断使能.
[11]	<b>TAMP3IEN</b> Tamper 3 或 对 1 中断使能位 设置 TAMP3IEN 为 1 也可以使能tamper 3 中断事件唤醒芯片功能 0 = Tamper 3 或对 1 中断禁止. 1 = Tamper 3或对 1 中断使能.
[10]	<b>TAMP2IEN</b> Tamper 2中断使能位 设置 TAMP2IEN 为 1 也可以使能tamper 2 中断事件唤醒芯片功能 0 = Tamper 2 中断禁止. 1 = Tamper 2 中断使能.
[9]	<b>TAMP1IEN</b> Tamper 1 或 对 0中断使能位 设置 TAMP1IEN 为 1 也可以使能tamper 1 中断事件唤醒芯片功能 0 = Tamper 1 或对 0 中断禁止. 1 = Tamper 3或对 1 中断使能.
[8]	<b>TAMPOIEN</b> Tamper 0中断使能位 设置 TAMPOIEN 为 1 也可以使能tamper 0 中断事件唤醒芯片功能 0 = Tamper 0 中断禁止.

		1 = Tamper 0 中断使能
[7:2]	<b>Reserved</b>	保留.
[1]	<b>TICKIEN</b>	<b>时钟节拍中断使能位</b> 设置 TICKIEN 为 1 也可以使能芯片节拍中断唤醒功能 0 = 禁止 RTC 时钟节拍中断 1 = 使能 RTC 时钟节拍中断
[0]	<b>ALMIEN</b>	<b>闹钟中断使能位</b> 设置 ALMIEN 为 1 也可以使能芯片闹钟中断唤醒功能 0 = 禁止 RTC 闹钟中断 1 = 使能 RTC 闹钟中断

## RTC\_INTSTS RTC 中断状态寄存器

寄存器	偏移地址	R/W	描述	复位值
RTC_INTSTS	RTC_BA+0x2C	R/W	RTC 中断状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved		TAMP5IF	TAMP4IF	TAMP3IF	TAMP2IF	TAMP1IF	TAMPOIF
7	6	5	4	3	2	1	0
Reserved						TICKIF	ALMIF

位	描述	
[31:14]	Reserved	保留.
[13]	TAMP5IF	<p><b>Tamper 5 或对 2 中断标志</b>  当 TAMP5_PIN 借测到电平不等于 TAMP5LV (RTC_TAMPCTL[29]), 或在 DYNPR2EN (RTC_TAMPCTL[31]) 激活期间 TAMP4_PIN 和 TAMP5_PIN 未连接, 或在 DYNPR2EN (RTC_TAMPCTL[31]) 和 DYN2ISS (RTC_TAMPCTL[1]) 激活期间 TAMP0_PIN 和 TAMP5_PIN 未连接, 该位置1。  0 = 没有 Tamper 5 或对 2 中断标志产生.  1 = Tamper 5 或对 2 中断标志产生.  <b>注意1:</b> 该位写1清0.  <b>注意2:</b> 清所有的 TAPMxIF 会自动清除 RTC_TAMPTIME 和 RTC_TAMPCAL</p>
[12]	TAMP4IF	<p><b>Tamper 4 中断标志</b>  当 TAMP4_PIN 借测到电平不等于 TAMP4LV (RTC_TAMPCTL[25]), 该位置1。  0 = 没有 Tamper 4 中断标志产生.  1 = Tamper 4 中断标志产生.  <b>注意1:</b> 该位写1清0.  <b>注意2:</b> 清所有的 TAPMxIF 会自动清除 RTC_TAMPTIME 和 RTC_TAMPCAL.</p>
[11]	TAMP3IF	<p><b>Tamper 3 或对 1 中断标志</b>  当 TAMP3_PIN 借测到电平不等于 TAMP3LV (RTC_TAMPCTL[21]), 或在 DYNPR1EN (RTC_TAMPCTL[23]) 激活期间 TAMP2_PIN 和 TAMP3_PIN 未连接, 或在 DYNPR1EN (RTC_TAMPCTL[23]) 和 DYN1ISS (RTC_TAMPCTL[0]) 激活期间 TAMP0_PIN 和 TAMP3_PIN 未连接, 该位置1。  0 = 没有 Tamper 3 或对 1 中断标志产生.  1 = Tamper 3 或对 1 中断标志产生.  <b>注意1:</b> 该位写1清0.  <b>注意2:</b> 清所有的 TAPMxIF 会自动清除 RTC_TAMPTIME 和 RTC_TAMPCAL</p>
[10]	TAMP2IF	<b>Tamper 2 中断标志</b>

		<p>当TAMP2_PIN 侦测到电平不等于TAMP2LV (RTC_TAMPCTL[17]), 该位置1。</p> <p>0 = 没有 Tamper 2 中断标志产生。 1 = Tamper 2 中断标志产生。</p> <p><b>注意1:</b> 该位写1清0。</p> <p><b>注意2:</b> 清所有的 TAPMxIF 会自动清除 RTC_TAMPTIME 和 RTC_TAMPCAL..</p>
[9]	<b>TAMP1IF</b>	<p><b>Tamper 1 或对0 中断标志</b></p> <p>当TAMP1_PIN 侦测到电平不等于TAMP1LV (RTC_TAMPCTL[13]), 或在DYNPROEN (RTC_TAMPCTL[15])激活期间TAMP0_PIN 和 TAMP1_PIN未连接, 该位置1。</p> <p>0 = 没有 Tamper 3 或对 1 中断标志产生。 1 = Tamper 3 或对 1 中断标志产生。</p> <p><b>注意1:</b> 该位写1清0。</p> <p><b>注意2:</b> 清所有的 TAPMxIF 会自动清除 RTC_TAMPTIME 和 RTC_TAMPCAL..</p>
[8]	<b>TAMP0IF</b>	<p><b>Tamper 0 中断标志</b></p> <p>当TAMP0_PIN 侦测到电平不等于TAMP0LV (RTC_TAMPCTL[9]), 该位置1。</p> <p>0 = 没有 Tamper 0中断标志产生。 1 = Tamper 0 中断标志产生。</p> <p><b>注意1:</b> 该位写1清0。</p> <p><b>注意2:</b> 清所有的 TAPMxIF 会自动清除 RTC_TAMPTIME 和 RTC_TAMPCAL..</p>
[7:2]	<b>Reserved</b>	保留.
[1]	<b>TICKIF</b>	<p><b>RTC时钟节拍中断标志</b></p> <p>0 = 未发生节拍情况 1 = 发生节拍情况</p> <p><b>注:</b> 该位写1清0</p>
[0]	<b>ALMIF</b>	<p><b>RTC 闹钟中断标志</b></p> <p>0 = 不匹配闹钟条件 1 = 匹配闹钟条件</p> <p><b>注:</b> 该位写1清0</p>

**RTC TICK RTC时钟节拍寄存器**

寄存器	偏移地址	R/W	描述	复位值
RTC_TICK	RTC_BA+0x30	R/W	RTC时钟节拍寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved					TICK		

位	描述	
[31:3]	Reserved	保留.
[2:0]	TICK	<p><b>时钟节拍寄存器</b></p> <p>该位用来设定产生时钟节拍中断的周期.</p> <p>000 = 时钟节拍 为1秒      001 = 时钟节拍 为1/2 秒      010 = 时钟节拍 为1/4 秒      011 = 时钟节拍 为1/8 秒      100 = 时钟节拍 为1/16 秒      101 = 时钟节拍 为1/32 秒      110 = 时钟节拍 为1/64 秒.      111 = 时钟节拍 为1/128 秒</p> <p><b>注意:</b> 在 RTC 访问使能位RWENF (RTC_RWEN[16])激活后, 该寄存器的值可以被读回。</p>

注: 在 RTC 访问使能位RWENF (RTC\_RWEN[16])激活后, 该寄存器的值可以被读回。

## RTC\_TAMSK RTC 时间闹钟屏蔽寄存器

寄存器	偏移地址	R/W	描述	复位值
RTC_TAMSK	RTC_BA+0x34	R/W	RTC 时间闹钟屏蔽寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved		MTENHR	MHR	MTENMIN	MMIN	MTENSEC	MSEC

位	描述	
[31:6]	<b>Reserved</b>	保留.
[5]	<b>MTENHR</b>	屏蔽闹钟小时时间的十位数(0~2)
[4]	<b>MHR</b>	屏蔽闹钟小时时间的个位数(0~9)
[3]	<b>MTENMIN</b>	屏蔽闹钟分钟的十位数(0~5)
[2]	<b>MMIN</b>	屏蔽闹钟分钟的个位数(0~9)
[1]	<b>MTENSEC</b>	屏蔽闹钟秒钟的十位数(0~5)
[0]	<b>MSEC</b>	屏蔽闹钟秒钟的个位数(0~9)

注:

1. RTC\_TAMSK 为 BCD 计数方式, RTC 不会对载入值的合理性进行检测。.
2. 括号内列出载入值的合理范围。
3. MTENHR/MHR 基于24小时制。.

## RTC CAMSK RTC 日历闹钟屏蔽寄存器

寄存器	偏移地址	R/W	描述	复位值
RTC_CAMSK	RTC_BA+0x38	R/W	RTC日历闹钟屏蔽寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved		MTENYEAR	MYEAR	MTENMON	MMON	MTENDAY	MDAY

位	描述	
	Reserved	保留
[5]	MTENYEAR	屏蔽闹钟年日历的十位数(0~9)
[4]	MYEAR	屏蔽闹钟年日历的个位数(0~9)
[3]	MTENMON	屏蔽闹钟月日历的十位数(0~1)
[2]	MMON	屏蔽闹钟月日历的个位数(0~9)
[1]	MTENDAY	屏蔽闹钟天日历的个位数(0~3)
[0]	MDAY	屏蔽闹钟天日历的十位数(0~9)

注:

1. RTC\_CALM为BCD计数方式，RTC不会对载入值的合理性进行检测。
2. 括号内列出载入值的合理范围。

**RTC SPRCTL RTC备用功能控制寄存器**

寄存器	偏移地址	R/W	描述	复位值
RTC_SPRCTL	RTC_BA+0x3C	R/W	RTC备用功能控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved		SPRCSTS	Reserved		SPRRWEN	Reserved	

位	描述	
[31:6]	Reserved	保留.
[5]	SPRCSTS	<p><b>SPR 清除标志</b>            该位表明如果侦测到指定盗取事件后, RTC_SPR0 ~ RTC_SPR19中内容是否已被清除.            0 = 特定寄存器内容没有被清除            1 = 特定寄存器内容已被清除            写1可清除该位  <b>注:</b> RTC_INTSTS[13:8] 不等于0, 该位保持1.</p>
[4:3]	Reserved	保留.
[2]	SPRRWEN	<p><b>SPR 寄存器使能位</b>            0 = 备用寄存器禁用            1 = 备用寄存器使能  <b>注意:</b> 当备用寄存器禁用后, RTC_SPR0 ~ RTC_SPR19将不可访问</p>
[1:0]	Reserved	保留.

**RTC\_SPRx RTC 备用寄存器**

寄存器	偏移地址	R/W	描述	复位值
RTC_SPR0	RTC_BA+0x40	R/W	RTC备用寄存器0	0x0000_0000
RTC_SPR1	RTC_BA+0x44	R/W	RTC 备用寄存器1	0x0000_0000
RTC_SPR2	RTC_BA+0x48	R/W	RTC 备用寄存器2	0x0000_0000
RTC_SPR3	RTC_BA+0x4C	R/W	RTC 备用寄存器3	0x0000_0000
RTC_SPR4	RTC_BA+0x50	R/W	RTC 备用寄存器4	0x0000_0000
RTC_SPR5	RTC_BA+0x54	R/W	RTC 备用寄存器5	0x0000_0000
RTC_SPR6	RTC_BA+0x58	R/W	RTC 备用寄存器6	0x0000_0000
RTC_SPR7	RTC_BA+0x5C	R/W	RTC 备用寄存器7	0x0000_0000
RTC_SPR8	RTC_BA+0x60	R/W	RTC 备用寄存器8	0x0000_0000
RTC_SPR9	RTC_BA+0x64	R/W	RTC 备用寄存器9	0x0000_0000
RTC_SPR10	RTC_BA+0x68	R/W	RTC 备用寄存器10	0x0000_0000
RTC_SPR11	RTC_BA+0x6C	R/W	RTC 备用寄存器11	0x0000_0000
RTC_SPR12	RTC_BA+0x70	R/W	RTC 备用寄存器12	0x0000_0000
RTC_SPR13	RTC_BA+0x74	R/W	RTC 备用寄存器13	0x0000_0000
RTC_SPR14	RTC_BA+0x78	R/W	RTC 备用寄存器14	0x0000_0000
RTC_SPR15	RTC_BA+0x7C	R/W	RTC 备用寄存器15	0x0000_0000
RTC_SPR16	RTC_BA+0x80	R/W	RTC 备用寄存器16	0x0000_0000
RTC_SPR17	RTC_BA+0x84	R/W	RTC 备用寄存器17	0x0000_0000
RTC_SPR18	RTC_BA+0x88	R/W	RTC 备用寄存器18	0x0000_0000
RTC_SPR19	RTC_BA+0x8C	R/W	RTC 备用寄存器19	0x0000_0000

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

位	描述	
[31:0]	<b>SPARE</b>	<p>备用寄存器 这个区域用于存储和备份用户的信息。 一旦tamper引脚事件被侦测到，这部分区域的内容将会被硬件自动清除。在向备用寄存器存储备份信息之前，用户应该确认使能位REWNF (RTC_RWEN[16])处于使能状态</p>

RTC\_LXTCTL RTC 32K晶振控制寄存器

寄存器	偏移地址	R/W	描述	复位值
RTC_LXTCTL	RTC_BA+0x100	R/W	RTC 32.768 kHz晶振控制寄存器	0x0000_000E

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved					GAIN		Reserved

位	描述	
[31:3]	Reserved	保留.
[2:1]	GAIN	<p><b>晶振增益选择</b></p> <p>根据晶振外部负载和工作温度范围,用户可以选择晶振增益。 增益值越大对应越强的驱动能力和更高的功耗.</p> <p>000 = L0 模式.</p> <p>001 = L1 模式..</p> <p>010 = L2 模式.</p> <p>011 = L3 模式.</p>
[0]	Reserved	保留.

**RTC GPIOCTL0 RTC GPIO控制寄存器0**

寄存器	偏移地址	R/W	描述				复位值
RTC_GPIOCTL0	RTC_BA+0x104	R/W	RTC GPIO 控制寄存器0				0x0000_0000

31	30	29	28	27	26	25	24
Reserved		PUSEL3		CTLSEL3	DOUT3	OPMODE3	
23	22	21	20	19	18	17	16
Reserved		PUSEL2		CTLSEL2	DOUT2	OPMODE2	
15	14	13	12	11	10	9	8
Reserved		PUSEL1		CTLSEL1	DOUT1	OPMODE1	
7	6	5	4	3	2	1	0
Reserved		PUSEL0		CTLSEL0	DOUT0	OPMODE0	

位	描述
[31:30]	Reserved 保留.
[29:28]	PUSEL3 <b>IO 上拉和下拉使能</b> 决定 PF.7 I/O 上拉或下拉 00 = PF.7 上拉和下拉禁止. 01 = PF.7 上拉使能. 10 = PF.7 下拉使能. <b>注:</b> 基本上，上拉控制和下拉控制有如下限制 仅当OPMODE3设置成三态或开漏模式时，单独上拉/下拉控制才有效。
[27]	CTLSEL3 <b>IO 管脚状态备用选择</b> 当 TAMP1EN 禁止，PF.7 管脚 (TAMPER1 管脚) 可以用作 GPIO 功能。用户可以编程 CTLSEL3 来决定 PF.7 I/O 功能是由系统电源域 GPIO 模块控制还是由 V <sub>BAT</sub> 电源域 RTC_GPIOCTL0 控制寄存器控制。 0 = PF.7 管脚 I/O 功能由 GPIO 模块控制。 当系统电源关闭，硬件自动变CTLSEL3 = 1。 1 = PF.7 管脚 I/O 功能由V <sub>BAT</sub> 电源域控制 CTLSEL3置1后，PF.7 管脚功能和I/O 状态由 OPMODE3[1:0] 和DOUT3控制。 <b>注:</b> 当系统电源关闭且INIT[0](RTC_INIT[0])是1，CTLSEL3会被硬件自动置1。
[26]	DOUT3 <b>IO 输出数据</b> 0 = PF.7 输出低. 1 = PF.7 输出高.
[25:24]	OPMODE3 <b>IO 操作模式</b> 00 = PF.7是仅输入模式 01 = PF.7是推挽输出模式

		10 = PF.7 是开漏模式。 11 = PF.7 是准双向模式
[23:22]	<b>Reserved</b>	保留.
[21:20]	<b>PUSEL2</b>	<p><b>IO 上拉和下拉使能</b>            决定 PF.6 I/O 上拉或下拉            00 = PF.6 上拉和下拉禁止.            01 = PF.6 上拉使能.            10 = PF.6 下拉使能.</p> <p><b>注:</b>            基本上，上拉控制和下拉控制有如下限制            仅当OPMODE2设置成三态或开漏模式时，单独上拉/下拉控制才有效。</p>
[19]	<b>CTLSEL2</b>	<p><b>IO 管脚状态备用选择</b>            当TAMP0EN禁止， PF.6 管脚 (TAMPER0管脚) 可以用作 GPIO 功能。用户可以编程CTLSEL2来决定 PF.6 I/O 功能是由系统电源域GPIO模块控制还是由V<sub>BAT</sub> 电源域RTC_GPIOCTL0 控制寄存器控制。</p> <p>0 = PF.6 管脚 I/O 功能由 GPIO 模块控制.            当系统电源关闭，硬件自动变CTLSEL2 = 1.            1 = PF.6 管脚 I/O 功能由V<sub>BAT</sub> 电源域控制            CTLSEL2置1后，PF.6 管脚功能和I/O 状态由OPMODE2 [1:0] 和DOUT2控制。  <b>注：</b>当系统电源关闭且INIT[0](RTC_INIT[0])是1，CTLSEL2会被硬件自动置1。</p>
[18]	<b>DOUT2</b>	<p><b>IO 输出数据</b>            0 = PF.6 输出低.            1 = PF.6 输出高.</p>
[17:16]	<b>OPMODE2</b>	<p><b>IO 操作模式</b>            00 = PF.6是仅输入模式            01 = PF.6是推挽输出模式            10 = PF.6 是开漏模式.            11 = PF.6 是准双向模式</p>
[15:14]	<b>Reserved</b>	保留.
[13:12]	<b>PUSEL1</b>	<p><b>IO 上拉和下拉使能</b>            决定 PF.5 I/O 上拉或下拉            00 = PF.5 上拉和下拉禁止.            01 = PF.5 上拉使能.            10 = PF.5 下拉使能.</p> <p><b>注:</b>            基本上，上拉控制和下拉控制有如下限制            仅当OPMODE1设置成三态或开漏模式时，单独上拉/下拉控制才有效。</p>
[11]	<b>CTLSEL1</b>	<p><b>IO 管脚状态备用选择</b>            当32 kHz晶振禁止， PF.5 管脚 (X32KI管脚) 可以用作 GPIO 功能。用户可以编程CTLSEL1来决定 PF.5 I/O 功能是由系统电源域GPIO模块控制还是由V<sub>BAT</sub> 电源域RTC_GPIOCTL0 控制寄存器控制。</p> <p>0 = PF.5 管脚 I/O 功能由 GPIO 模块控制.            当系统电源关闭，硬件自动变CTLSEL1 = 1.</p>

		<b>1 = PF.5 管脚 I/O 功能由V<sub>BAT</sub> 电源域控制</b> CTLSEL1置1后，PF.5 管脚功能和I/O 状态由OPMODE1 [1:0] 和DOUT1控制。 <b>注：</b> 当系统电源关闭且INIT[0](RTC_INIT[0])是1，CTLSEL1会被硬件自动置1。
[10]	<b>DOUT1</b>	<b>IO 输出数据</b> 0 = PF.5 输出低. 1 = PF.5 输出高.
[9:8]	<b>OPMODE1</b>	<b>IO 操作模式</b> 00 = PF.5是仅输入模式 01 = PF.5是推挽输出模式 10 = PF.5 是开漏模式. 11 = PF.5 是准双向模式
[7:6]	<b>Reserved</b>	保留.
[5:4]	<b>PUSEL0</b>	<b>IO 上拉和下拉使能</b> 决定 PF.4 I/O 上拉或下拉 00 = PF.4 上拉和下拉禁止. 01 = PF.4 上拉使能. 10 = PF.4 下拉使能. <b>注：</b> 基本上，上拉控制和下拉控制有如下限制 仅当OPMODE0设置成三态或开漏模式时，单独上拉/下拉控制才有效。
[3]	<b>CTLSEL0</b>	<b>IO 管脚状态备用选择</b> 当32 kHz晶振禁止，PF.4 管脚 (X32KO管脚) 可以用作 GPIO 功能。用户可以编程CTLSEL0来决定 PF.4 I/O 功能是由系统电源域GPIO模块控制还是由V <sub>BAT</sub> 电源域RTC_GPIOCTL0 控制寄存器控制。 0 = PF.4 管脚 I/O 功能由 GPIO 模块控制. 当系统电源关闭，硬件自动变CTLSEL0 = 1. 1 = PF.4 管脚 I/O 功能由V <sub>BAT</sub> 电源域控制 CTLSEL0置1后，PF.4 管脚功能和I/O 状态由OPMODE0 [1:0] 和DOUT0控制。 <b>注：</b> 当系统电源关闭且INIT[0](RTC_INIT[0])是1，CTLSEL0会被硬件自动置1。
[2]	<b>DOUT0</b>	<b>IO 输出数据</b> 0 = PF.4 输出低. 1 = PF.4 输出高.
[1:0]	<b>OPMODE0</b>	<b>IO 操作模式</b> 00 = PF.4是仅输入模式，不带上拉电阻 01 = PF.4是推挽输出模式 10 = PF.4是开漏模式. 11 = PF.4 是准双向模式带内部上拉

## RTC GPIOCTL1 RTC GPIO 控制寄存器1

寄存器	偏移地址	R/W	描述	复位值
RTC_GPIOCTL1	RTC_BA+0x108	R/W	RTC GPIO 控制寄存器1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved		PUSEL7		CTLSEL7	DOUT7	OPMODE7	
23	22	21	20	19	18	17	16
Reserved		PUSEL6		CTLSEL6	DOUT6	OPMODE6	
15	14	13	12	11	10	9	8
Reserved		PUSEL5		CTLSEL5	DOUT5	OPMODE5	
7	6	5	4	3	2	1	0
Reserved		PUSEL4		CTLSEL4	DOUT4	OPMODE4	

位	描述	
[31:30]	Reserved	保留.
[29:28]	PUSEL7	<p><b>IO 上拉和下拉使能</b>            决定 PF.11 I/O 上拉或下拉            00 = PF.11 上拉和下拉禁止.            01 = PF.11 上拉使能.            10 = PF.11 下拉使能.  <b>注:</b>            基本上，上拉控制和下拉控制有如下限制            仅当OPMODE7设置成三态或开漏模式时，单独上拉/下拉控制才有效。</p>
[27]	CTLSEL7	<p><b>IO 管脚状态备用选择</b>            当TAMP5EN禁止， PF.11 管脚 (TAMPER5管脚) 可以用作 GPIO 功能。用户可以编程 CTLSEL7 来决定 PF.11 I/O 功能是由系统电源域 GPIO 模块控制还是由V<sub>BAT</sub> 电源域 RTC_GPIOCTL1 控制寄存器控制。            0 = PF.11 管脚 I/O 功能由 GPIO 模块控制.            当系统电源关闭，硬件自动变CTLSEL7 = 1.            1 = PF.11 管脚 I/O 功能由V<sub>BAT</sub> 电源域控制            CTLSEL7置1后， PF.11 管脚功能和I/O 状态由OPMODE7 [1:0] 和DOUT7控制。  <b>注:</b> 当系统电源关闭且INIT[0](RTC_INIT[0])是1， CTLSEL7会被硬件自动置1。</p>
[26]	DOUT7	<p><b>IO 输出数据</b>            0 = PF.11 输出低.            1 = PF.11 输出高.</p>
[25:24]	OPMODE7	<p><b>IO 操作模式</b>            00 = PF.11是仅输入模式            01 = PF.11是推挽输出模式            10 = PF.11 是开漏模式.</p>

		11 = PF.11 是准双向模式
[23:22]	Reserved	保留.
[21:20]	PUSEL6	<p><b>IO 上拉和下拉使能</b>            决定 PF.10 I/O 上拉或下拉            00 = PF.10 上拉和下拉禁止.            01 = PF.10 上拉使能.            10 = PF.10 下拉使能.  <b>注:</b>            基本上，上拉控制和下拉控制有如下限制            仅当OPMODE6设置成三态或开漏模式时，单独上拉/下拉控制才有效。</p>
[19]	CTLSEL6	<p><b>IO 管脚状态备用选择</b>            当TAMP4EN禁止， PF.10 管脚 (TAMPER4管脚) 可以用作 GPIO 功能。用户可以编程 CTLSEL6 来决定 PF.10 I/O 功能是由系统电源域 GPIO 模块控制还是由 V<sub>BAT</sub> 电源域 RTC_GPIOCTL1 控制寄存器控制。            0 = PF.10 管脚 I/O 功能由 GPIO 模块控制.            当系统电源关闭，硬件自动变CTLSEL6 = 1.            1 = PF.10 管脚 I/O 功能由 V<sub>BAT</sub> 电源域控制            CTLSEL6置1后，PF.10 管脚功能和I/O 状态由OPMODE6 [1:0] 和DOUT6控制。  <b>注:</b> 当系统电源关闭且INIT[0](RTC_INIT[0])是1，CTLSEL6会被硬件自动置1。</p>
[18]	DOUT6	<p><b>IO 输出数据</b>            0 = PF.10 输出低.            1 = PF.10 输出高.</p>
[17:16]	OPMODE6	<p><b>IO 操作模式</b>            00 = PF.10是仅输入模式            01 = PF.10是推挽输出模式            10 = PF.10 是开漏模式.            11 = PF.10 是准双向模式带</p>
[15:14]	Reserved	保留.
[13:12]	PUSEL5	<p><b>IO 上拉和下拉使能</b>            决定 PF.9 I/O 上拉或下拉            00 = PF.9 上拉和下拉禁止.            01 = PF.9 上拉使能.            10 = PF.9 下拉使能.  <b>注:</b>            基本上，上拉控制和下拉控制有如下限制            仅当OPMODE5设置成三态或开漏模式时，单独上拉/下拉控制才有效。</p>
[11]	CTLSEL5	<p><b>IO 管脚状态备用选择</b>            当TAMP3EN禁止， PF.9 管脚 (TAMPER3管脚) 可以用作 GPIO 功能。用户可以编程 CTLSEL5 来决定 PF.9 I/O 功能是由系统电源域 GPIO 模块控制还是由 V<sub>BAT</sub> 电源域 RTC_GPIOCTL1 控制寄存器控制。            0 = PF.9 管脚 I/O 功能由 GPIO 模块控制.            当系统电源关闭，硬件自动变CTLSEL5 = 1.            1 = PF.9 管脚 I/O 功能由 V<sub>BAT</sub> 电源域控制</p>

		CTLSEL5置1后，PF.11管脚功能和I/O状态由OPMODE5[1:0]和DOUT5控制。 注：当系统电源关闭且INIT[0](RTC_INIT[0])是1，CTLSEL5会被硬件自动置1。
[10]	DOUT5	<b>IO 输出数据</b> 0 = PF.9 输出低。 1 = PF.9 输出高。
[9:8]	OPMODE5	<b>IO 操作模式</b> 00 = PF.9 是仅输入模式 01 = PF.9 是推挽输出模式 10 = PF.9 是开漏模式。 11 = PF.9 是准双向模式
[7:6]	Reserved	保留。
[5:4]	PUSEL4	<b>IO 上拉和下拉使能</b> 决定 PF.8 I/O 上拉或下拉 00 = PF.8 上拉和下拉禁止。 01 = PF.8 上拉下拉使能。 10 = PF.8 下拉使能。 <b>注：</b> 基本上，上拉控制和下拉控制有如下限制 仅当OPMODE4设置成三态或开漏模式时，单独上拉/下拉控制才有效。
[3]	CTLSEL4	<b>IO 管脚状态备用选择</b> 当TAMP2EN禁止，PF.8管脚(TAMPER2管脚)可以用作GPIO功能。用户可以编程CTLSEL4来决定PF.8 I/O功能是由系统电源域GPIO模块控制还是由V <sub>BAT</sub> 电源域RTC_GPIOCTL1控制寄存器控制。 0 = PF.8管脚I/O功能由GPIO模块控制。 当系统电源关闭，硬件自动变CTLSEL4 = 1。 1 = PF.8管脚I/O功能由V <sub>BAT</sub> 电源域控制 CTLSEL4置1后，PF.8管脚功能和I/O状态由OPMODE4[1:0]和DOUT4控制。 <b>注：</b> 当系统电源关闭且INIT[0](RTC_INIT[0])是1，CTLSEL4会被硬件自动置1。
[2]	DOUT4	<b>IO 输出数据</b> 0 = PF.8 输出低。 1 = PF.8 输出高。
[1:0]	OPMODE4	<b>IO 操作模式</b> 00 = PF.8 是仅输入模式 01 = PF.8 是推挽输出模式 10 = PF.8 是开漏模式。 11 = PF.8 是准双向模式

RTC DSTCTL RTC 夏令时控制寄存器

寄存器	偏移地址	R/W	描述	复位值
RTC_DSTCTL	RTC_BA+0x110	R/W	RTC 夏令时控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved					DSBAK	SUBHR	ADDHR

位	描述	
[31:3]	<b>Reserved</b>	保留.
[2]	<b>DSBAK</b>	夏令时返回 0= 正常模式. 1= 夏令时模式.
[1]	<b>SUBHR</b>	减 1 小时 0 = 无影响. 1 = 表示RTC小时数字已经被减掉一个小时用于冬季时间的变化
[0]	<b>ADDHR</b>	加 1 小时 0 = 无影响. 1 = 表示RTC小时数字已经被加上一个小时用于夏季时间的变化

RTC\_TAMPCTL RTC Tamper 管脚控制寄存器

寄存器	偏移地址	R/W	描述	复位值
RTC_TAMPC TL	RTC_BA+0x120	R/W	RTC Tamper 管脚控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
DYNPR2EN	TAMP5DBEN	TAMP5LV	TAMP5EN	Reserved	TAMP4DBEN	TAMP4LV	TAMP4EN
23	22	21	20	19	18	17	16
DYNPR1EN	TAMP3DBEN	TAMP3LV	TAMP3EN	Reserved	TAMP2DBEN	TAMP2LV	TAMP2EN
15	14	13	12	11	10	9	8
DYNPR0EN	TAMP1DBEN	TAMP1LV	TAMP1EN	Reserved	TAMP0DBEN	TAMP0LV	TAMP0EN
7	6	5	4	3	2	1	0
DYNRATE			SEEDRLD	DYNsrc		DYN2ISS	DYN1ISS

位	描述
[31]	<b>DYNPR2EN</b> 动态对 2 使能位 0 = 静态侦测. 1 = 动态侦测.
[30]	<b>TAMP5DBEN</b> <b>Tamper 5 消抖使能位</b> 0 = Tamper 5 消抖禁止. 1 = Tamper 5 消抖使能.
[29]	<b>TAMP5LV</b> <b>Tamper 5 电平</b> 该位取决于用于静态tampter 侦测的tampter 管脚的电平属性 0 = 侦测电平是低. 1 = 侦测电平是高.
[28]	<b>TAMP5EN</b> <b>Tamper 5 侦测使能位</b> 0 = Tamper 5 侦测禁止. 1 = Tamper 5 侦测使能. <b>注意1:</b> 参考是 RTC时钟, Tamper 侦测需要同步2 ~ 3 个RTC时钟.
[27]	<b>Reserved</b> 保留 .
[26]	<b>TAMP4DBEN</b> <b>Tamper 4 消抖使能位</b> 0 = Tamper 4 消抖禁止. 1 = Tamper 4 消抖使能.
[25]	<b>TAMP4LV</b> <b>Tamper 4 电平</b> 该位取决于用于静态tampter 侦测的tampter 管脚的电平属性 0 = 侦测电平是低. 1 = 侦测电平是高.
[24]	<b>TAMP4EN</b> <b>Tamper 4 侦测使能位</b>

		0 = Tamper 4 值测禁止. 1 = Tamper 4 值测使能. <b>注意1:</b> 参考是 RTC时钟, Tamper 值测需要同步2 ~ 3 个RTC时钟.
[23]	DYNPR1EN	<b>动态对 1使能位</b> 0 = 静态值测. 1 = 动态值测.
[22]	TAMP3DBEN	<b>Tamper 3 消抖使能位</b> 0 = Tamper 3 消抖禁止. 1 = Tamper 3 消抖使能.
[21]	TAMP3LV	<b>Tamper 3 电平</b> 该位取决于用于静态tampter 值测的tampter 管脚的电平属性 0 = 值测电平是低. 1 = 值测电平是高.
[20]	TAMP3EN	<b>Tamper 3 值测使能位</b> 0 = Tamper 3 值测禁止. 1 = Tamper 3 值测使能. <b>注意1:</b> 参考是 RTC时钟, Tamper 值测需要同步2 ~ 3 个RTC时钟.
[19]	Reserved	保留.
[18]	TAMP2DBEN	<b>Tamper 2 消抖使能位</b> 0 = Tamper 2 消抖禁止. 1 = Tamper 2 消抖使能.
[17]	TAMP2LV	<b>Tamper 2 电平</b> 该位取决于用于静态tampter 值测的tampter 管脚的电平属性 0 = 值测电平是低. 1 = 值测电平是高.
[16]	TAMP2EN	<b>Tamper 2 值测使能位</b> 0 = Tamper 2 值测禁止. 1 = Tamper 2 值测使能. <b>注意1:</b> 参考是 RTC时钟, Tamper 值测需要同步2 ~ 3 个RTC时钟.
[15]	DYNPROEN	<b>动态对 0使能位</b> 0 = 静态值测. 1 = 动态值测.
[14]	TAMP1DBEN	<b>Tamper 1 消抖使能位</b> 0 = Tamper 1 消抖禁止. 1 = Tamper 1 消抖使能.
[13]	TAMP1LV	<b>Tamper 1 电平</b> 该位取决于用于静态tampter 值测的tampter 管脚的电平属性 0 = 值测电平是低. 1 = 值测电平是高.
[12]	TAMP1EN	<b>Tamper 1 值测使能位</b> 0 = Tamper 1 值测禁止.

		1 = Tamper 1 侦测使能。 <b>注意1:</b> 参考是 RTC时钟, Tamper 侦测需要同步2 ~ 3 个RTC时钟.
[11]	Reserved	保留.
[10]	TAMP0DBEN	<b>Tamper 0 消抖使能位</b> 0 = Tamper 0 消抖禁止. 1 = Tamper 0 消抖使能.
[9]	TAMPOLV	<b>Tamper 0 电平</b> 该位取决于用于静态tampter 侦测的tampter 管脚的电平属性 0 = 侦测电平是低. 1 = 侦测电平是高.
[8]	TAMP0EN	<b>Tamper 0 侦测使能位</b> 0 = Tamper 0 侦测禁止. 1 = Tamper 0 侦测使能. <b>注意1:</b> 参考是 RTC时钟, Tamper 侦测需要同步2 ~ 3 个RTC时钟.
[7:5]	DYNRATE	<b>动态变化率</b> 此项用于选择动态tamper输出变化率 000 = $2^{10} * \text{RTC\_CLK}$ . 001 = $2^{11} * \text{RTC\_CLK}$ . 010 = $2^{12} * \text{RTC\_CLK}$ . 011 = $2^{13} * \text{RTC\_CLK}$ . 100 = $2^{14} * \text{RTC\_CLK}$ . 101 = $2^{15} * \text{RTC\_CLK}$ . 110 = $2^{16} * \text{RTC\_CLK}$ . 111 = $2^{17} * \text{RTC\_CLK}$ . <b>注意:</b> 在修改该域之后, 设置SEEDRLD (RTC_TAMPCTL[4]) 可以立即重载变化率
[4]	SEEDRLD	<b>为PRNG引擎重载新的种子</b> 设置该位, tamper 配置会被重载 0 = 基于当前种子产生密匙. 1 = 重载新的种子. <b>注意:</b> 设置该位之前, tamper配置应该设置完成.
[3:2]	DYNNSRC	<b>动态参考 样本</b> 在动态对模式, 当.当前样本使用完后, 该域决定了新的参考样本 00 或 10 =当参考样本使用完毕, 新的参考样本通过随机数产生器产生。 01 =当参考样本使用完毕, 新的参考样本重复之前的随机值。 11 =当参考 样本使用完后, 新的参考样本重覆使用用户输入到seed的值 <b>注意:</b> 在修改该位域之后, SEEDRLD (RTC_TAMPCTL[4]) 应该置位.
[1]	DYN2ISS	<b>动态对 2 输入源选择</b> 该位决定在动态模式 Tamper 5 输入是来自Tamper 4 或 Tamper 0 0 = Tamper 输入来自 Tamper 4. 1 = Tamper 输入来自Tamper 0. <b>注意:</b> 仅 当DYNPR2EN (RTC_TAMPCTL[24]) 和 DYNPROEN (RTC_TAMPCTL[15]) 置位, 该位才有效。
[0]	DYN1ISS	<b>动态对 1 输入源选择</b>

		<p>该位决定在动态模式 Tamper 3 输入是来自 Tamper 2 或 Tamper 0</p> <p>0 = Tamper 输入来自 Tamper 2.</p> <p>1 = Tamper 输入来自 Tamper 0.</p> <p><b>注意:</b>仅 当 DYNPR1EN (RTC_TAMPCTL[16]) 和 DYNPR0EN (RTC_TAMPCTL[15]) 置位，该位才有效。</p>
--	--	--

RTC\_TAMPSEED RTC Tamper 动态种子寄存器

寄存器	偏移地址	R/W	描述	复位值
RTC_TAMPS EED	RTC_BA+0x128	R/W	RTC Tamper 动态种子寄存器	0x0000_0000

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

位	描述	
[31:0]	SEED	种子值

RTC\_TAMPTIME RTC Tamper 时间寄存器

寄存器	偏移地址	R/W	描述	复位值
RTC_TAMPTIME	RTC_BA+0x130	R	RTC Tamper 时间寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved		TENHR			HR		
15	14	13	12	11	10	9	8
Reserved	TENMIN			MIN			
7	6	5	4	3	2	1	0
Reserved	TENSEC			SEC			

位	描述	
[31:24]	Reserved	保留.
[21:20]	TENHR	TAMPER 时间小时的十位 (0~2) 注意: 仅支持24小时制
[19:16]	HR	TAMPER 时间小时的个位 (0~9)
[15]	Reserved	保留.
[14:12]	TENMIN	TAMPER 时间分钟的十位 (0~5)
[11:8]	MIN	TAMPER 时间分钟的个位 (0~9)
[7]	Reserved	保留.
[6:4]	TENSEC	TAMPER 时间秒的十位 (0~5)
[3:0]	SEC	TAMPER 时间秒的个位 (0~9)

注意:

1. RTC\_TALM 是BCD格式数字计数器。
2. 数值的合理范围在括号中列出了
3. 该域不能更新直到所有TAMPxFIF 被清除

## RTC Tamper 日期寄存器

寄存器	偏移地址	R/W	描述	复位值
RTC_TAMPC AL	RTC_BA+0x134	R	RTC Tamper 日期寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
TENYEAR				YEAR			
15	14	13	12	11	10	9	8
Reserved			TENMON	MON			
7	6	5	4	3	2	1	0
Reserved		TENDAY		DAY			

位	描述	
[31:24]	Reserved	保留.
[23:20]	TENYEAR	TAMPER 日期年的十位 (0~9)
[19:16]	YEAR	TAMPER日期年的个位(0~9)
[15:13]	Reserved	保留.
[12]	TENMON	TAMPER 日期月的十位 (0~1)
[11:8]	MON	TAMPER 日期月的个位 (0~9)
[7:6]	Reserved	保留.
[5:4]	TENDAY	TAMPER 日期日的十位 (0~3)
[3:0]	DAY	TAMPER日期日的个位(0~9)

注意:

1. RTC\_CTAMP是BCD格式数字计数器.
2. 数值的合理范围在括号中列出了
3. 该域不能更新直到所有TAMPxFIF 被清除

## 6.11 EPWM发生器和捕获定时器

### 6.11.1 概述

该芯片提供了两路EPWM发生器EPWM0 和 EPWM1。每路EPWM支持6通道EPWM输出或输入捕获。有一个12位的预分频器把时钟源分频后输入给16位的计数器，另外还有一个16位的比较器。EPWM计数器支持向上，向下，上下计数方式。EPWM用比较器和计数器的比较来产生事件，这些事件用来产生EPWM脉冲，中断，EADC/DAC转换触发信号。

EPWM发生器支持两种标准EPWM输出模式：独立模式和互补模式，它们的架构不同。标准输出模式又有两种输出功能：组功能和同步功能。组功能可以在独立模式和互补模式下使能。同步功能只有在互补模式下才可以被使能。互补模式，有两个比较器产生各种带12位死区时间的EPWM脉宽，另外还有一个自由触发比较器来产生给EADC的触发信号。EPWM输出控制单元，它支持极性输出，独立管脚屏蔽和刹车功能。

EPWM也支持输入捕获功能，当输入通道有向上跳变、向下跳变、或者两者都有的跳变时，锁存EPWM计数器的值到相应的寄存器中。捕获功能也支持通过PDMA把捕获到的数据搬到内存。

### 6.11.2 特性

#### 6.11.2.1 EPWM 功能特性

- 支持最大时钟频率到PLL 的最大频率
- 支持两个EPWM模块，每个模块提供6个输出通道
- 支持独立模式的EPWM输出/输入捕获
- 支持3组互补通道的互补模式
  - 12位解析度的死区插入
  - 相控制的同步功能
  - 每个周期两个比较值
- 支持12位从1到4096的预分频
- 支持16位解析度的EPWM计数器
  - 向上，向下和上下计数操作类型
- 支持one-shot或自动装载计数器工作模式
- 支持组功能
- 支持同步功能
- 每个EPWM管脚支持屏蔽功能和三态使能
- 支持刹车功能
  - 刹车源来自管脚、模拟比较器和系统安全事件（时钟故障、SRAM奇偶校验错误、欠压监测和CPU锁死）。
  - 刹车源管脚噪声滤波器
  - Leading edge blanking (LEB) 功能用于刹车源来自模拟比较器情况
  - 通过边沿检测刹车源来控制刹车状态直到刹车中断清除
  - 刹车条件解除后通过电平检测刹车源来控制自动恢复功能

- 支持下列事件中断:
  - EPWM计数器值为0、周期值或比较值
  - 发生刹车条件
- 支持下列事件触发EADC/DAC:
  - EPWM计数器值为0、周期值或比较值
  - EPWM计数器匹配自由触发比较器比较值(仅EADC)

#### 6.11.2.2 捕获功能特性

- 支持12个16位解析度的输入捕获通道
- 支持上升/下降沿捕获条件
- 支持输入上升/下降沿捕获中断
- 支持计数器重载选项的上升/下降沿捕获
- 支持EPWM的所有通道PDMA数据搬移功能

#### 6.11.3 框图

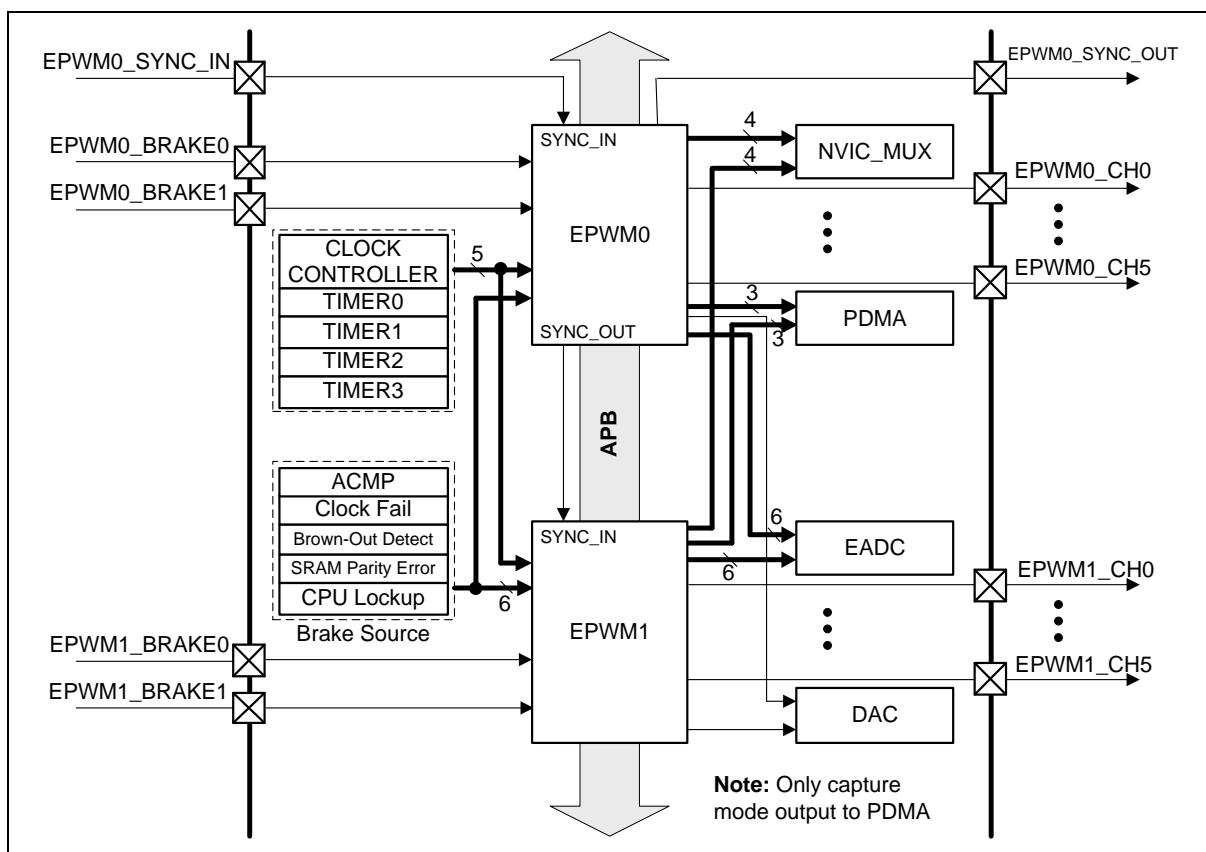


图 6.11-1 EPWM 发生器简要方块图

EPWM系统时钟可以设为等于或两倍的PCLK频率，图 6.11-2，寄存器设置细节请参考表 6.11-1。每个EPWM发生器有三个输入时钟源，每个时钟源可以选择来自EPWM时钟或者4组定时器触发EPWM输出，如图 6.11-3，通过ECLKSRC0 (EPWM\_CLKSRC[2:0]) 设置 EPWM\_CLK0，ECLKSRC2 (EPWM\_CLKSRC[10:8]) 设置 EPWM\_CLK2 和 ECLKSRC4 (EPWM\_CLKSRC[18:16]) 设置

EPWM\_CLK4。

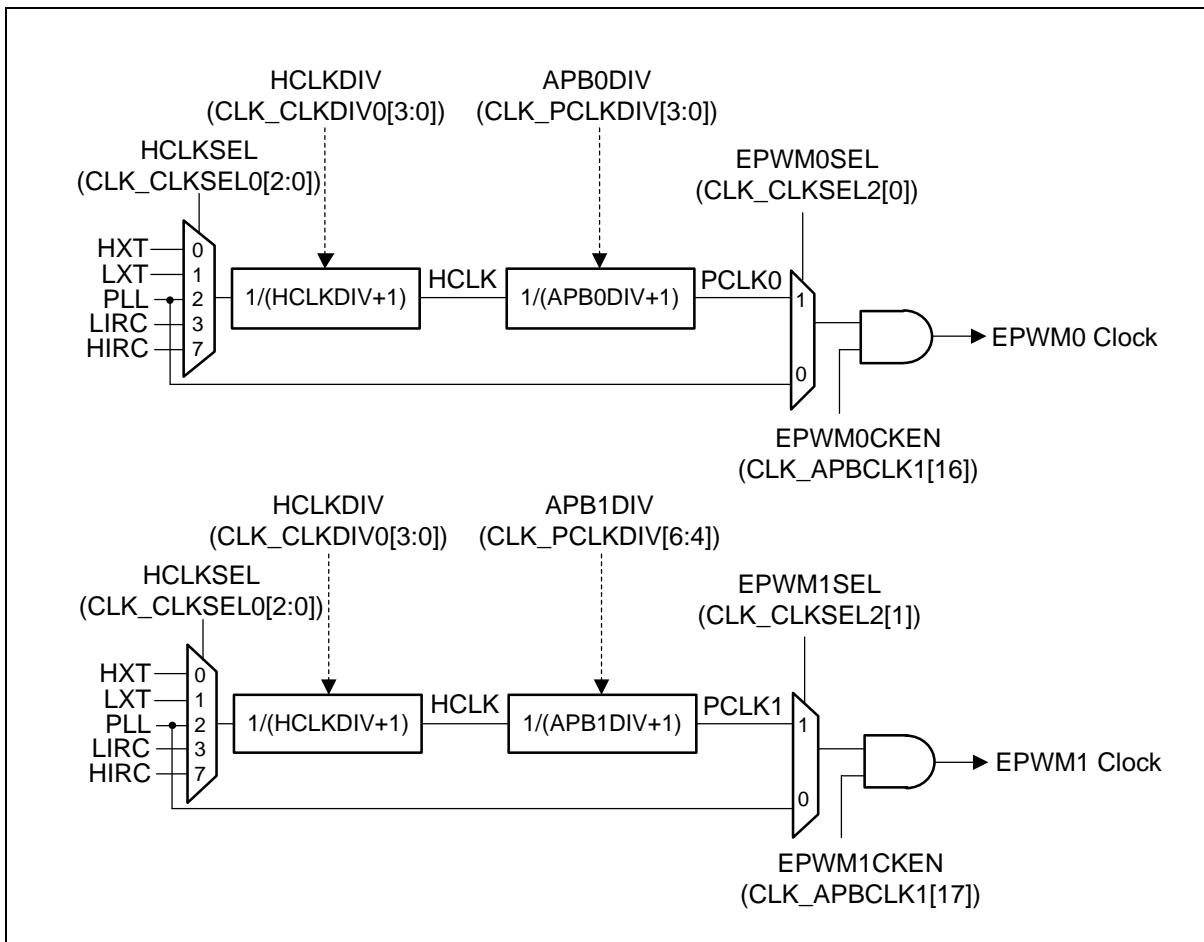


图 6.11-2 EPWM 时钟源控制

频率比 PCLK:EPWM 时钟	HCLK	PCLK	EPWM 时钟	HCLKSEL CLK_CLKSEL0[2:0]	HCLKDIV CLK_CLKDIV0[3: 0]	APBnDIV (CLK_CLKDIVn [2+4n:4n]), N 表示 0 或 1	EPWMnSEL (CLK_CLKSEL2[N]) ,N 表示 0 或 1
1:1	HCLK	PCLK	PCLK	不在乎	不在乎	不在乎	1
1:2	PLL	PLL/2	PLL	2	0	1	0
1:2	PLL/2	PLL/2	PLL	2	1	0	0

表 6.11-1 EPWM 时钟源控制寄存器设置表

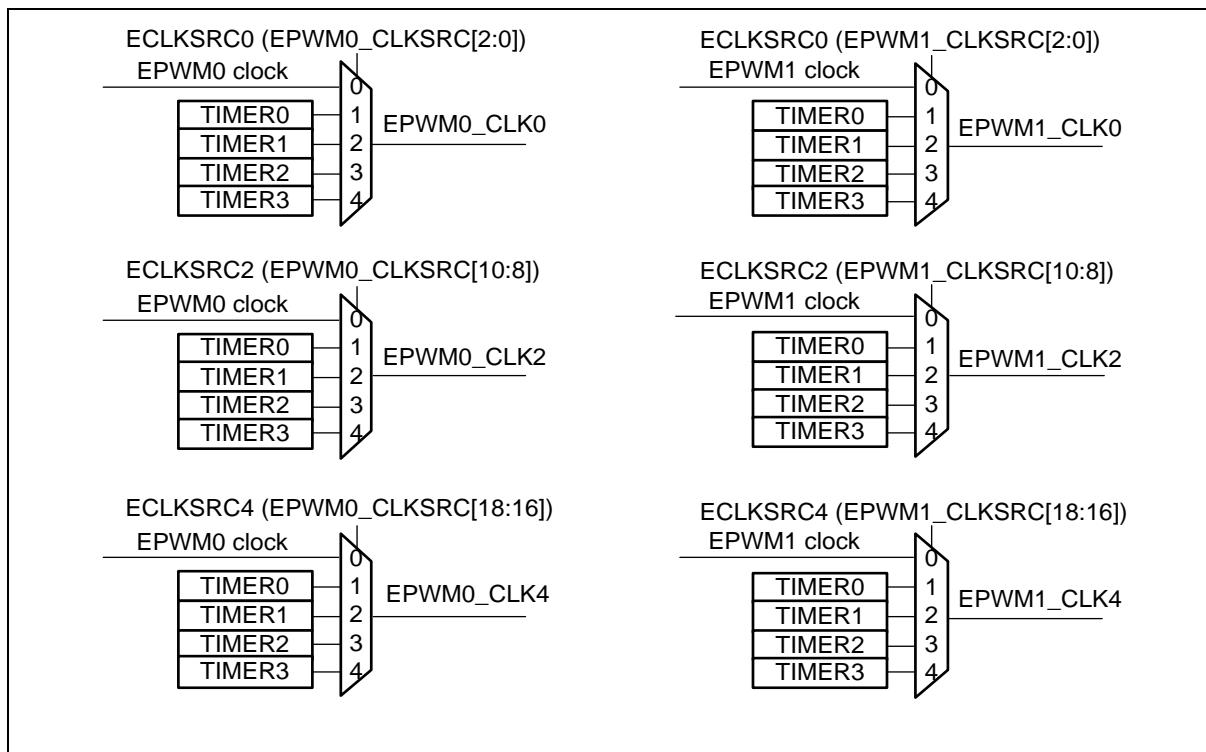


图 6.11-3 EPWM 时钟源控制

图 6.11-4 和 图 6.11-5 表示 EPWM 独立模式和互补模式的架构。不管独立模式还是互补模式，一对通道组 (EPWM\_CH0 和 EPWM\_CH1, EPWM\_CH2 和 EPWM\_CH3, EPWM\_CH4 和 EPWM\_CH5) 计数器来自相同的时钟源和预分频。当计数器的值等于 0，PERIOD(EPWM\_PERIODn[15:0]) 或比较器值，将产生事件。这些事件通过相应的发生器来产生 EPWM 脉冲、中断信号、EADC/DAC 的转换触发信号。输出控制是用来改变 EPWM 脉冲输出状态的。输出控制的刹车功能也能产生中断事件。在互补模式，同步功能有效，偶数通道用奇数通道比较器产生事件，自由触发比较器事件只用于产生触发 EADC 信号。

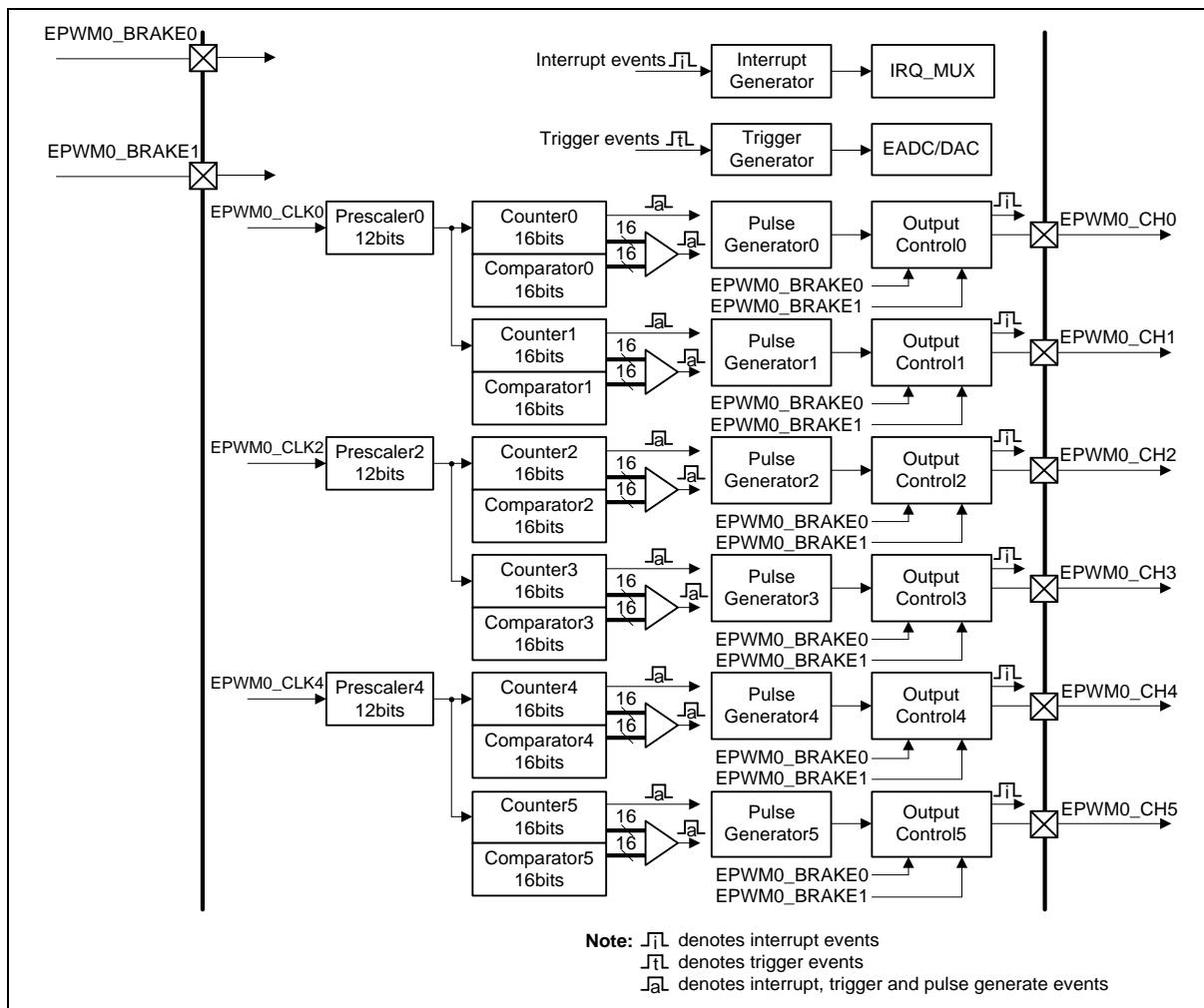


图 6.11-4 EPWM 独立模式架构图

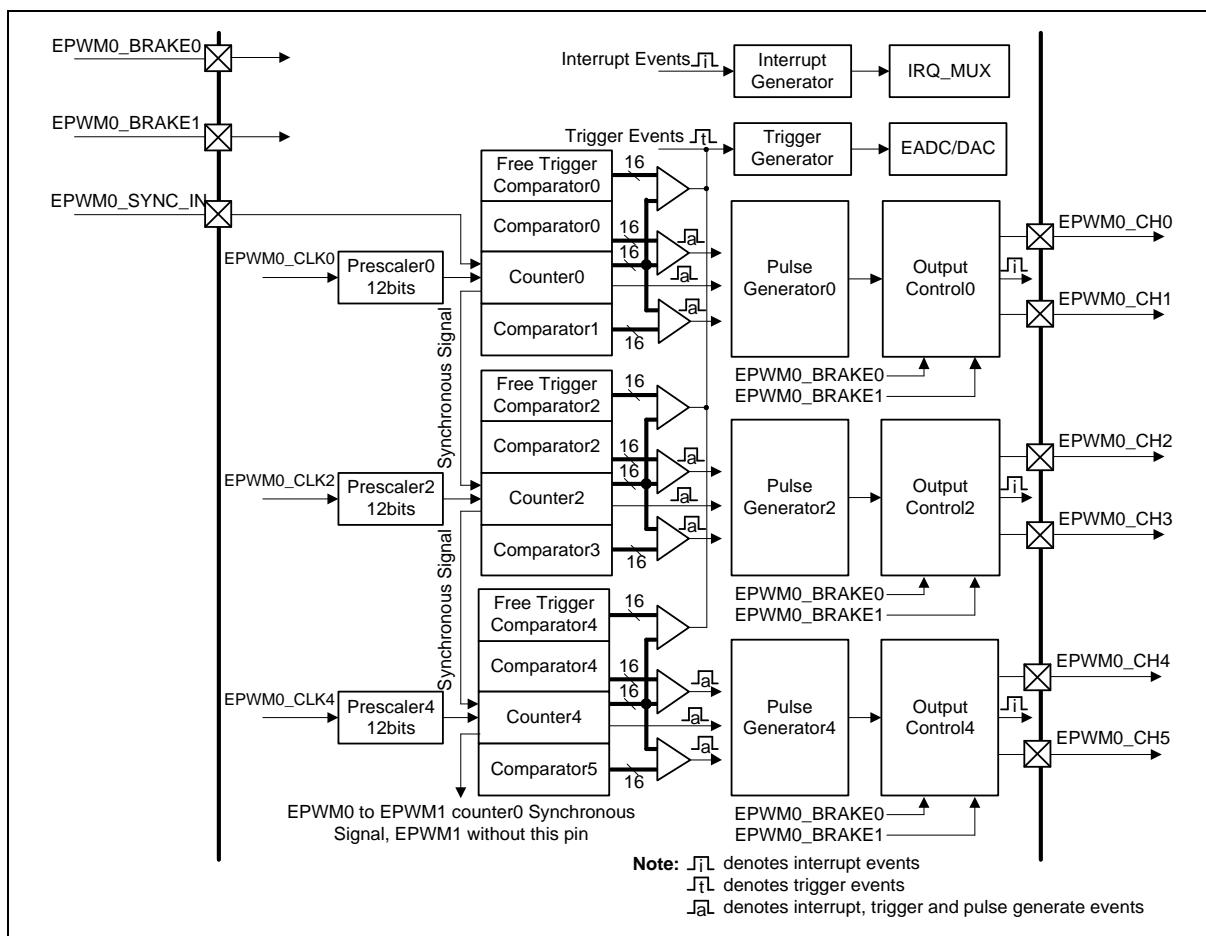


图 6.11-5 EPWM 互补模式架构图

#### 6.11.4 基本配置

##### 6.11.4.1 EPWM0 基本配置

- 时钟源配置
  - 通过EPWM0SEL (CLK\_CLKSEL2[0])选择EPWM外设时钟源
  - 通过EPWM0CKEN CLK\_APBCLK1[16]使能 EPWM0 外设时钟
- 复位配置
  - 通过EPWM0RST SYS\_IPRST2[16]复位 EPWM0
- 管脚配置

组	管脚名称	GPIO	MFP
EPWM0	EPWM0_BRAKE0	PE.8	MFP11
		PB.1	MFP13
	EPWM0_BRAKE1	PE.9	MFP11
		PB.0	MFP13
	EPWM0_CH0	PE.8	MFP10

	PB.5, PG.8	MFP11
	PE.7	MFP12
	PA.5	MFP13
EPWM0_CH1	PE.9	MFP10
	PB.4, PG.7	MFP11
	PE.6	MFP12
	PA.4	MFP13
EPWM0_CH2	PE.10	MFP10
	PB.3, PG.6	MFP11
	PE.5	MFP12
	PA.3	MFP13
EPWM0_CH3	PE.11	MFP10
	PB.2, PG.5	MFP11
	PE.4	MFP12
	PA.2	MFP13
EPWM0_CH4	PE.12	MFP10
	PB.1, PD.14	MFP11
	PE.3	MFP12
	PA.1	MFP13
EPWM0_CH5	PE.13	MFP10
	PB.0, PH.11	MFP11
	PE.2	MFP12
	PA.0	MFP13
EPWM0_SYNC_IN	PC.14	MFP11
	PA.15	MFP12
EPWM0_SYNC_OUT	PF.5	MFP9
	PA.11	MFP10

#### 6.11.4.2 EPWM1 基本配置

- 时钟源配置
  - 通过 EPWM1SEL (CLK\_CLKSEL2[1])选择EPWM1外设时钟源
  - 通过 EPWM1CKEN (CLK\_APBCLK1[17]).使能EPWM1外设时钟
- 复位配置
  - 通过 EPWM1RST SYS\_IPRST2[17]复位EPWM1
- 管脚配置

组	管脚名称	GPIO	MFP
EPWM1	EPWM1_BRAKE0	PB.7, PE.10	MFP11
	EPWM1_BRAKE1	PB.6, PE.11	MFP11
	EPWM1_CH0	PB.15, PE.13	MFP11
		PC.5, PC.12	MFP12
	EPWM1_CH1	PB.14, PC.8	MFP11
		PC.4, PC.11	MFP12
	EPWM1_CH2	PB.13, PC.7	MFP11
		PC.3, PC.10	MFP12
	EPWM1_CH3	PB.12, PC.6	MFP11
		PC.2, PC.9	MFP12
	EPWM1_CH4	PA.7	MFP11
		PB.1, PB.7, PC.1	MFP12
	EPWM1_CH5	PA.6	MFP11
		PB.0, PB.6, PC.0	MFP12

## 6.11.5 功能描述

### 6.11.5.1 EPWM 预分频器

EPWM预分频器是用于时钟源除频，预分频器计数CLKPSC +1次，EPWM计数器只加一次。通过CLKPSC (EPWM\_CLKPSCn[11:0], n = 0, 2, 4)来设置预分频双缓存。**錯誤！找不到參照來源。**是EPWM通道0 预分频波形的例子。预分频计数器会在下一个预分频计数器下数开始时重载CLKPSC。

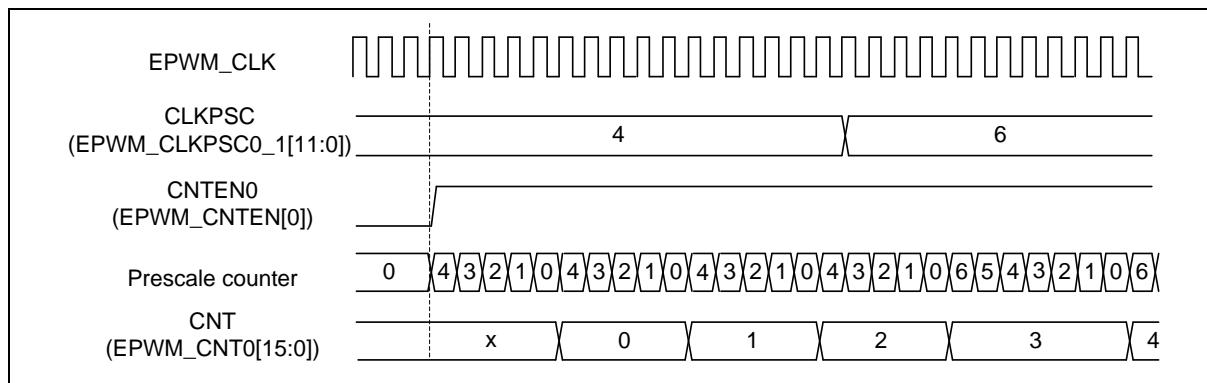


图 6.11-6 EPWM\_CH0 上计数模式的预分频器波形

### 6.11.5.2 EPWM 计数器

EPWM支持3种计数方式操作：向上计数，向下计数，上下计数方式。

对于EPWM通道0，CNT(EPWM\_CNT0[15:0])可以通过置位CNTCLR0 (EPWM\_CNTCLR[0])来清0，当预分频计数器数到0，CNT会被清除，且CNTCLR被硬件自动置0。

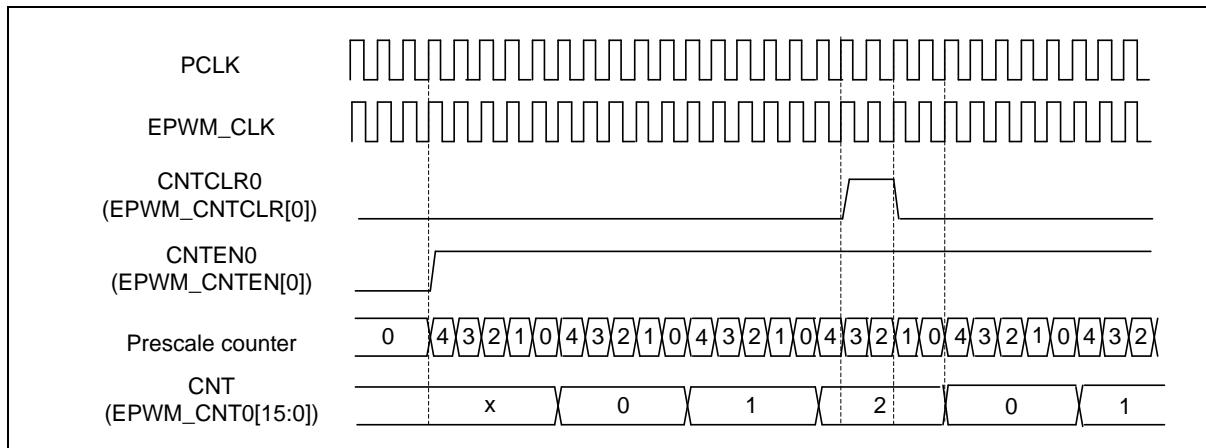


图 6.11-7 当设置清计数器 EPWMx 计数器的波形

#### 6.11.5.3 上计数模式

在向上计数操作中, CNTTYPEEn (EPWM\_CTL1[2n+1:2n], n = 0,1..5) 是 0x0。16位EPWM计数器是一个向上计数器并从0开始向上计数到PERIOD (EPWM\_PERIODn[15:0], 其中n表示通道数)来完成一个EPWM周期。当前计数值可以从CNT (EPWM\_CNTn[15:0])读出。当计数器计数到0且预分频计数到0将产生零点事件, 当计数到PERIOD且预分频计数到0将产生周期点事件。如下图显示一个向上计数器例子, EPWM周期时间=  $(\text{PERIOD}+1) * (\text{CLKPSC}+1) * \text{EPWMx\_CLK}$ .

$$\text{EPWM 周期}= (\text{PERIOD}+1) * (\text{CLKPSC}+1) * \text{EPWMx\_CLK}.$$

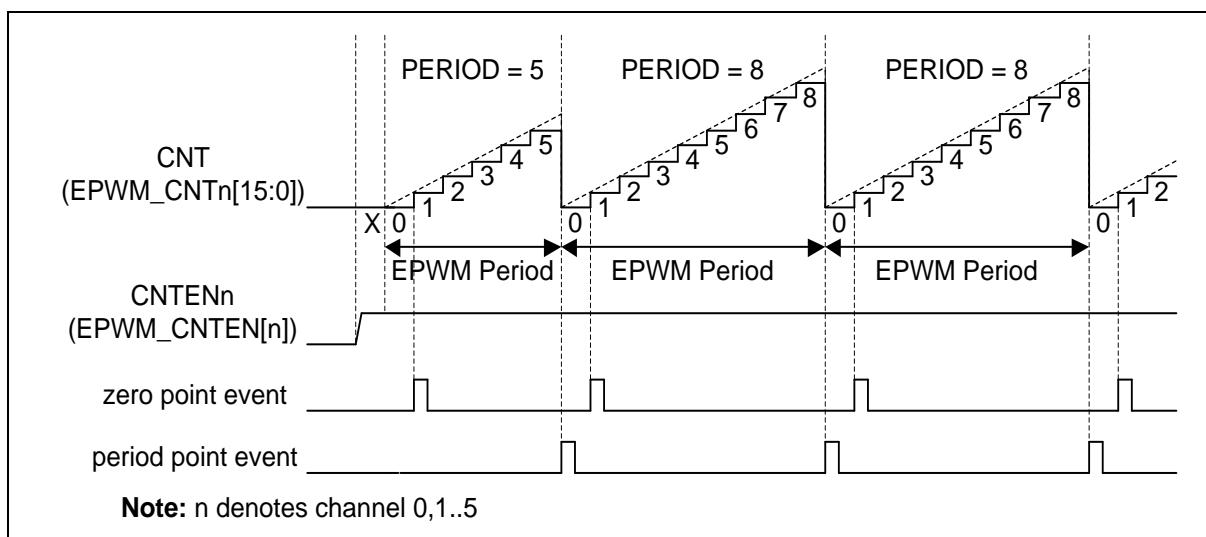


图 6.11-8 EPWM 上计数器模式

#### 6.11.5.4 下计数模式

在向下计数方式中, CNTTYPEEn (EPWM\_CTL1[2n+1:2n], n = 0,1..5) 是 0x1, 16位EPWM计数器是一个向下计数器并从PERIOD开始向下计数到0来完成一个EPWM周期。当前计数值可以从CNT (EPWM\_CNTn[15:0])读出。当计数器计数到0且预分频计数到0将产生零点事件, 当计数到PERIOD且预分频计数到0将产生周期事件, 如下图显示一个向下计数器例子:

EPWM 周期 = (PERIOD+1) \* (CLKPSC+1) \* EPWMx\_CLK.

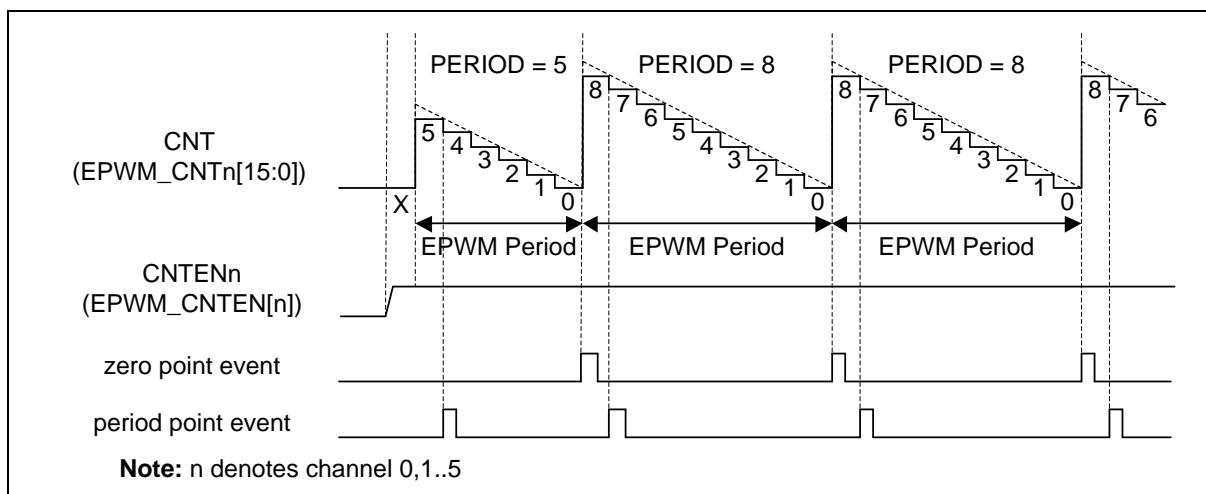


图 6.11-9 EPWM 下计数模式

#### 6.11.5.5 上下计数模式

在上下计数操作中, CNTTYPE<sub>n</sub> (EPWM\_CTL1[2n+1:2n], n = 0,1..5) 是 0x2, 16位EPWM计数器是一个上下计数器, 并开始向上计数从0到PERIOD然后又向下计数到0完成一个EPWM周期。当前计数值可以从CNT读出。当计数器计数到0且预分频计数到0将产生零点事件, 当计数到PERIOD将产生中心点事件, 如下图显示一个上下计数器例子:

EPWM 周期=(2\*PERIOD) \* (CLKPSC+1) \* EPWMx\_CLK.

DIRF (EPWM\_CNTn[16])是计数器方向标志, 高是向上计数, 低是向下计数。

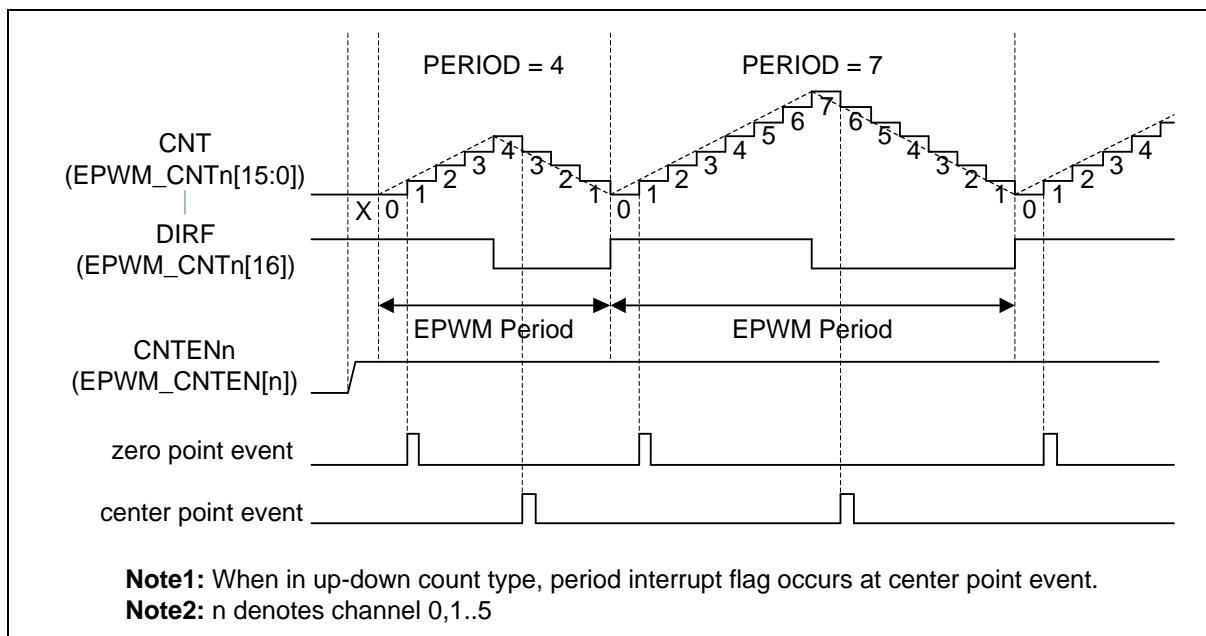


图 6.11-10 EPWM 上下计数器模式

## 6.11.5.6 EPWM 比较器

EPWM模块有两个比较器寄存器，一个是CMPDAT<sub>n</sub>(n = 0,1..5)，另一个是FTCMCPDAT<sub>n,m</sub>(n = 0,2,4, m = 1,3,5)。CMPDAT<sub>n</sub>是EPWM通道n的基本比较器寄存器。独立模式下每个通道只有一个CMPDAT<sub>n</sub>。CMPDAT<sub>n</sub>值一直与相应通道的计数器值作比较。互补模式下，奇数通道的计数器值无效，相应通道的比较器值一直与互补偶数通道的计数器值作比较。例如，通道0和通道1为互补通道，在互补模式，通道1的比较器持续与通道0计数器相比较，而不与通道1比较。当计数器值等于比较寄存器值，EPWM产生一个事件，并用事件产生一个EPWM脉冲，中断，或者触发EADC/DAC。在上下计数方式中，一个EPWM周期将产生两个事件，如下图所示。CMPU是上数比较点事件，CMRD是下数比较点事件。

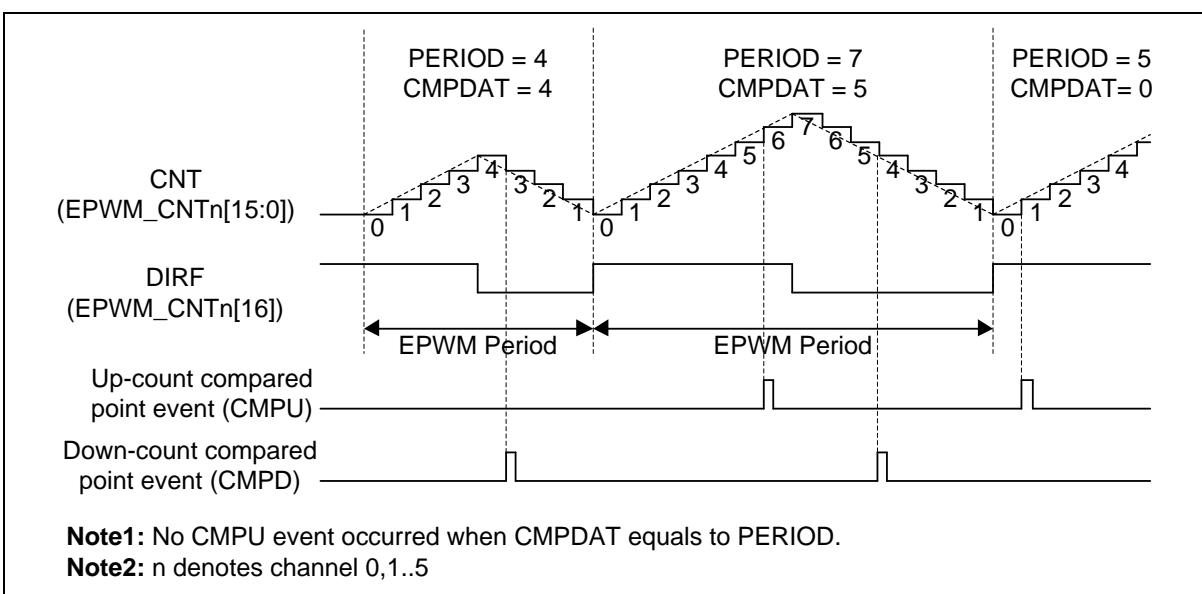


图 6.11-11 EPWM 在上下计数方式中的比较点事件

FTCMCPDAT是一个自由触发比较器寄存器。每组互补通道仅支持一个FTCMCPDAT寄存器。FTCMCPDAT<sub>n,m</sub>(n = 0,2,4, m = 1,3,5)的值持续与互补通道的偶通道相比较。当CNT等于FTCMCPDAT，下计数时FTCMD<sub>n</sub> (EPWM\_FTCI[10:8], n=0,2,4)被置1和上计数时FTCMU<sub>n</sub> (EPWM\_FTCI[2:0], n=0,2,4)被置1。另外EPWM产生一个事件，该事件仅可以触发EADC。

## 6.11.5.7 EPWM 双缓存

双缓存是用两个缓存器来把软件写和硬件操作时序分开。载入值到缓存有四种装载模式：周期装载模式，立即装载模式，窗口装载模式和中心装载模式。软件设置好寄存器后，硬件将按照装载模式时序将寄存器值装载到缓存寄存器中。硬件的操作是基于缓存寄存器的值。这样可以避免软硬件不同步时的操作问题。

EPWM 提供 PBUF (EPWM\_PBUFn[15:0]) 作为使 PERIOD 生效的缓存寄存器， CMPBUF (EPWM\_CMPBUFn[15:0]) 作为使 CMPDAT 生效的缓存寄存器， FTCMPBUF (EPWM\_FTCCMPBUFn\_m[15:0]) 作为使 FTCMPDAT 生效的缓存寄存器， CPSCBUF (EPWM\_CPSBUFN\_m[15:0]) 作为是 CLKPSC 生效的缓存寄存器。双缓存的概念是用在装载模式，如下面描述。例如图 6.11-12所示，在周期装载模式，通过软件写PERIOD、 CMPDAT和FTCMCPDAT，在下个周期时EPWM 将载入新值到他们的缓存 PBUF(EPWM\_PBUFn[15:0]), CMPBUF (EPWM\_CMPBUFn[15:0]) 和 FTCMPBUF (EPWM\_FTCCMPBUFn[15:0])而不影响当前计数操作。FTCMCU代表向上计数自由触发比较点事件，FTCMRD代表向下计数自由触发比较事件。

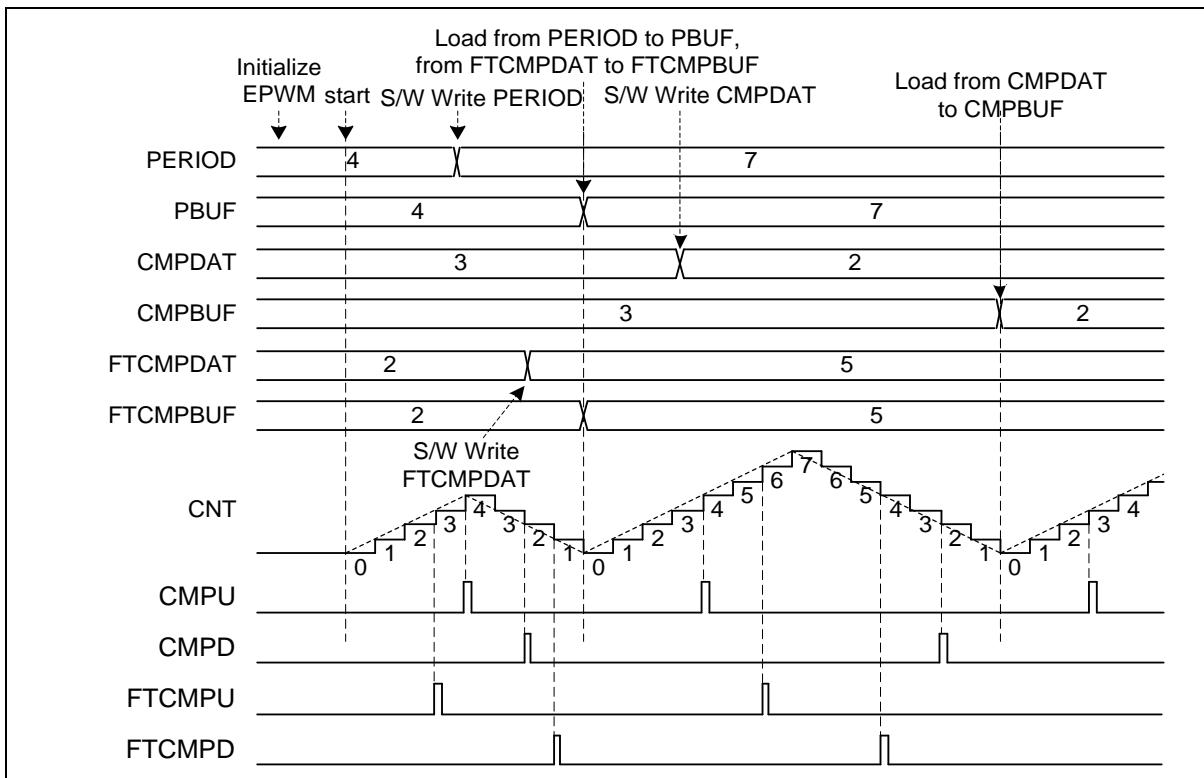


图 6.11-12 EPWM 双缓存说明

#### 6.11.5.8 周期装载模式

当立即装载模式，窗口装载模式和中心装载模式禁止，EPWM\_CTL0的IMMLDENn位，WINLDENn位和CTRLDn位为0。EPWM工作在周期装载模式。在周期装载模式，当每一个周期完成，CLKPSC(EPWM\_CLKPSCn\_m[11:0]),PERIOD(EPWM\_PERIODn[15:0])和CMP(EPWM\_CMPDATn[15:0])会全部加载到他们的激活寄存器CPSCBUF, PBUF 和 CMPBUF。例如：在向上计数操作EPWM计数器从零计数到PERIOD后或在向下计数操作从PERIOD计数到零，或在上下计数操作向上计数从零到PERIOD然后向下计数到零。

图 6.11-13表示向上计数操作时的周期装载时序，PERIOD DATA0表示PERIOD初始数据，PERIOD DATA1表示通过软件等所更新的第一个PERIOD 数据。CMPDAT同样遵照这规则。以下是图 6.11-13的步骤描述。用户可以通过观察EPWM周期和CMPU事件知道PERIOD 和 CMPDAT的更新条件。

1. point1处，软件写CMPDAT DATA1到CMPDAT
2. point 2处，EPWM周期结束，硬件装载CMPDAT DATA1 到 CMPBUF
3. point 3处，软件写PERIOD DATA1到 PERIOD
4. point 4处，EPWM周期结束，硬件装载CMPDAT DATA1 到 PBUF
5. point 5处，软件写PERIOD DATA2 到 PERIOD
6. point 6处，EPWM周期结束，硬件装载PERIOD DATA2到 PBUF

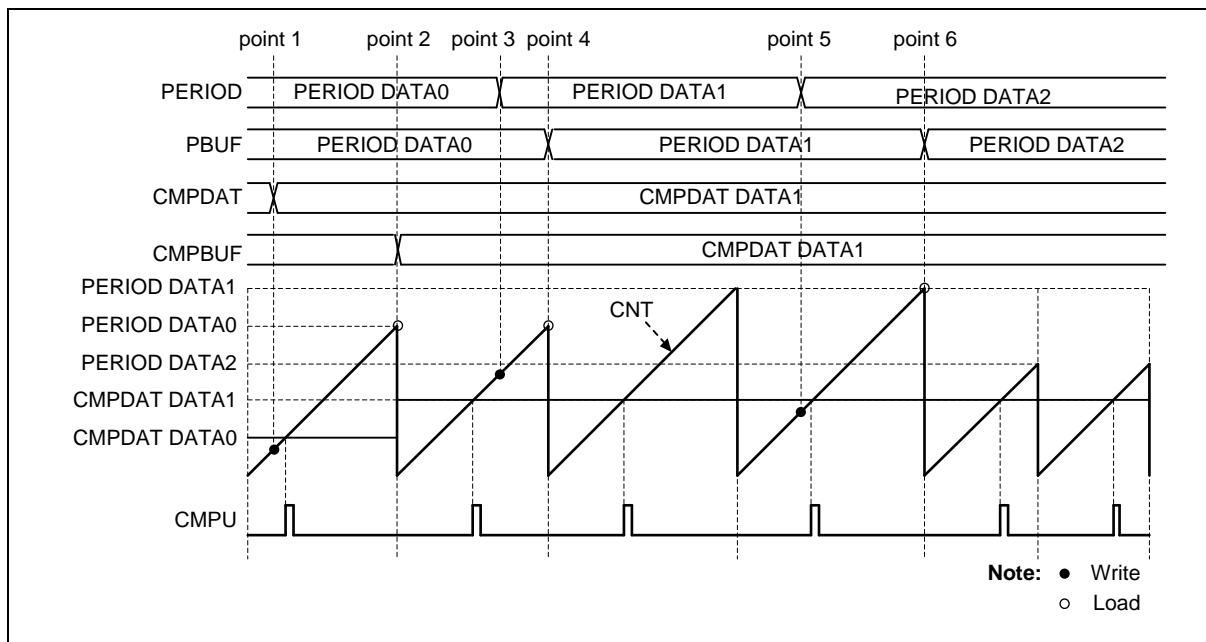


图 6.11-13 向上计数方式的周期装载模式

#### 6.11.5.9 立即装载模式

如果IMMLDENn (EPWM\_CTL0[21:16])位被置1时，EPWM工作在立即装载模式。在立即装载模式下，当软件更新CLKPSC(EPWM\_CLKPSCn\_m[11:0]), PERIOD(EPWM\_PERIODn[15:0])或CMP(EPWM\_CMPDATn[15:0])后，软件立即将CLKPSC、PERIOD和CMPDAT值装载到缓存CPSCBUF (EPWM\_CPSBUFn\_m[15:0]), PBUF (EPWM\_PBUFn[15:0])和CMPBUF (EPWM\_CMPBUFn[15:0])。如果PERIOD更新值小于当前计数值，计数器会计数到0xFFFF，当计数器计数到0xFFFF且预分频计数到0，标志CNTMAXF(EPWMx\_STATUS[5:0])会置位，然后计数器循环计数。立即装载模式有最高优先级，如果IMMLDENn被设置，通道n的其他装载模式将失效。图 6.11-14 所示例子的步骤顺序如下描述

1. 软件写CMPDAT DATA1，硬件在point1处将立即CMPDAT DATA1装载到CMPBUF
2. point 2处，软件写入PERIOD DATA1值大于当前值，计数器将继续计数到PERIOD DATA1来完成装载。
3. point 3处，软件写入PERIOD DATA2值小于当前值，计数器将继续计数到最大值0xFFFF并循环从0开始到PERIOD DATA2来完成这个周期装载。

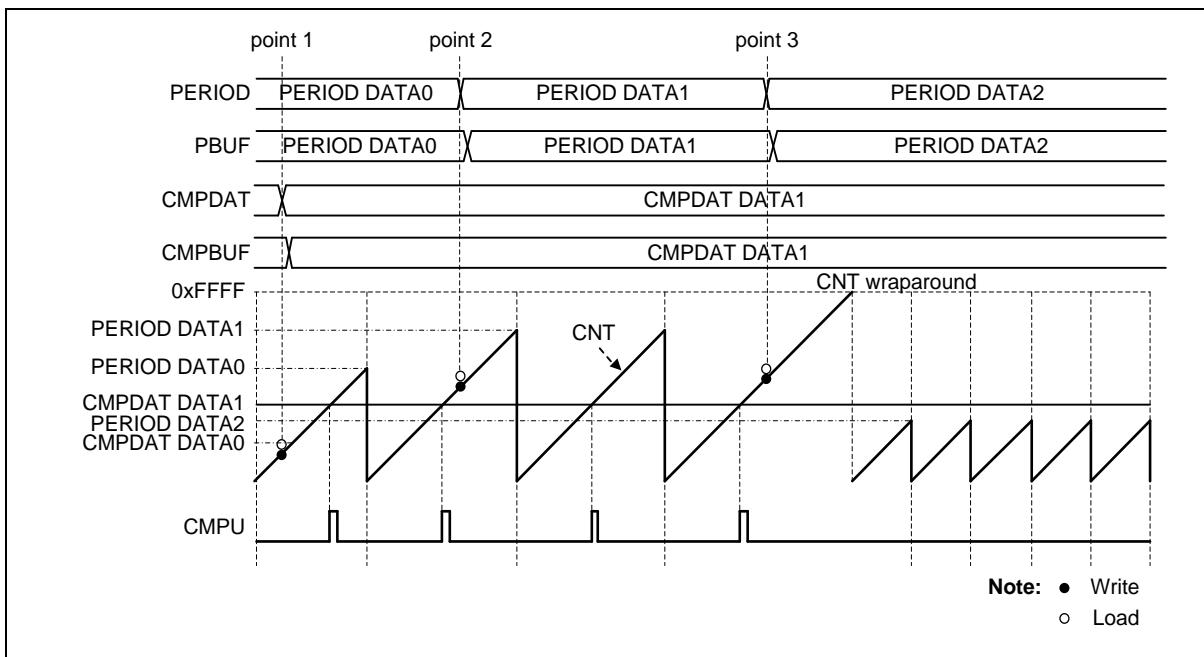


图 6.11-14 向上计数方式的立即装载模式

#### 6.11.5.10 窗口装载模式

如果 WINLDEN<sub>n</sub> (EPWM\_CTL0[13:8]) 位被置为 1, EPWM 工作在窗口装载模式。在窗口装载模式 CLKPSC(EPWM\_CLKPSCn\_m[11:0]), PERIOD(EPWM\_PERIODn[15:0]) 和 CMP(EPWM\_CMPDATn[15:0]) 会在每个周期完成装载到他们的激活寄存器 CPSCBUF, PBUF 和 CMPBUF, 但是所有装载仅当窗口打开时有效。每个通道n的装载窗口打开方式都是通过设置相应的 LOADn (EPWM\_LOAD[5:0])位为1, 然后硬件会在EPWM周期终止时关闭该窗口。图 6.11-15所示例子的步骤顺序如下描述

1. point 1处软件写CMPDAT DATA1 , 在这个周期中装载窗口不会被打开, 所以CMPDAT的值不会被装载到CMPBUF。
2. point2处软件写装载来打开装载窗口。
3. point 3处软件写PERIOD DATA1。
4. point 4处, 装载窗口被打开, 硬件将装载 CLKPSC DATA1 及 PERIOD DATA1 和 CMPDAT DATA1 到它们的缓冲区然后再EPWM周期终止时关闭装载窗口。
5. point 5处软件写PERIOD DATA2。
6. point 6处硬件在EPWM周期终止时装载CLKPSC DATA2 及 PERIOD DATA2到PBUF
7. point 7处软件写PERIOD DATA3
8. point8处软件写装载来打开装载窗口。
9. 硬件装载 CLKPSC DATA3 及 PERIOD DATA3 到PBUF , 然后在point 9处关闭装载窗口。

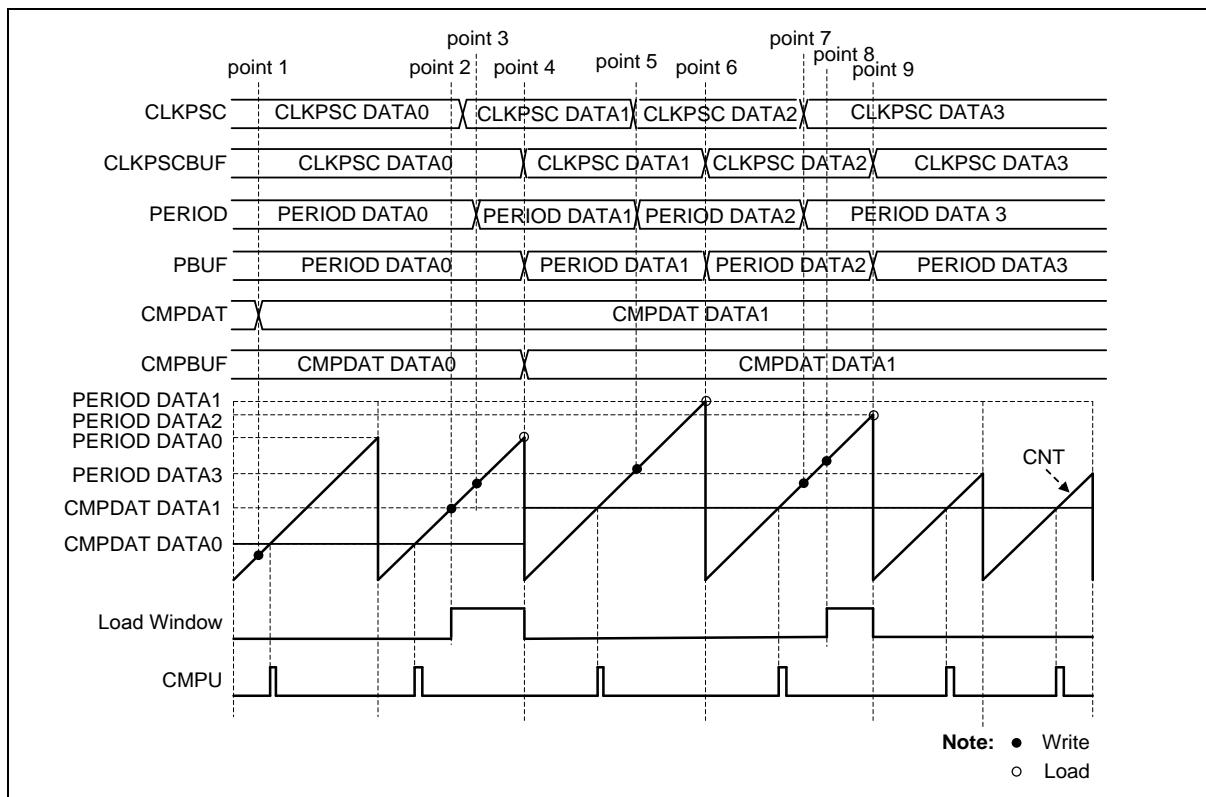


图 6.11-15 向上计数模式窗口装载

#### 6.11.5.11 中心装载模式

当CTRLDn (EPWM\_CTL0[5:0])的EPWM通道n相应位被置1，且EPWM计数器设置为上下计数模式，CNTTYPEn (EPWM\_CTL1[2n+1:2n], n = 0,1..5) 是 0x2，EPWM工作在中心装载模式。在中心装载模式，在每个周期的中点CMP(EPWM\_CMPDATn[15:0])值将装载到CMPBUF寄存器，也就是说当当前周期完成，计数器计数到PERIOD点，CLKPSC(EPWM\_CLKPSCn\_m[11:0]) 和 PERIOD(EPWM\_PERIODn[15:0])会装载到激活寄存器CPSCBUF 和 PBUF。周期装载模式可以配合窗口装载模式一起工作，CMP(EPWM\_CMPDATn[15:0])会在每一个周期的中心装载到激活寄存器CMPBUF，但是它仅在装载窗口期间有效。图 6.11-16所示例子的步骤顺序如下描述：

1. point 1处，软件写CMPDAT DATA1。
2. point 2处，EPWM周期中心点硬件装载CMPDAT DATA1到CMPBUF
3. point 3处，软件写PERIOD DATA1。
4. point 4处，EPWM周期终点硬件装载PERIOD DATA1到PBUF
5. point 5处，软件写CMPDAT DATA2
6. point 6处，在EPWM周期中心点硬件装载CMPDAT DATA2到CMPBUF
7. point 7处，软件写PERIOD DATA2
8. point 8处，在EPWM周期终点硬件装载PERIOD DATA2到PBUF

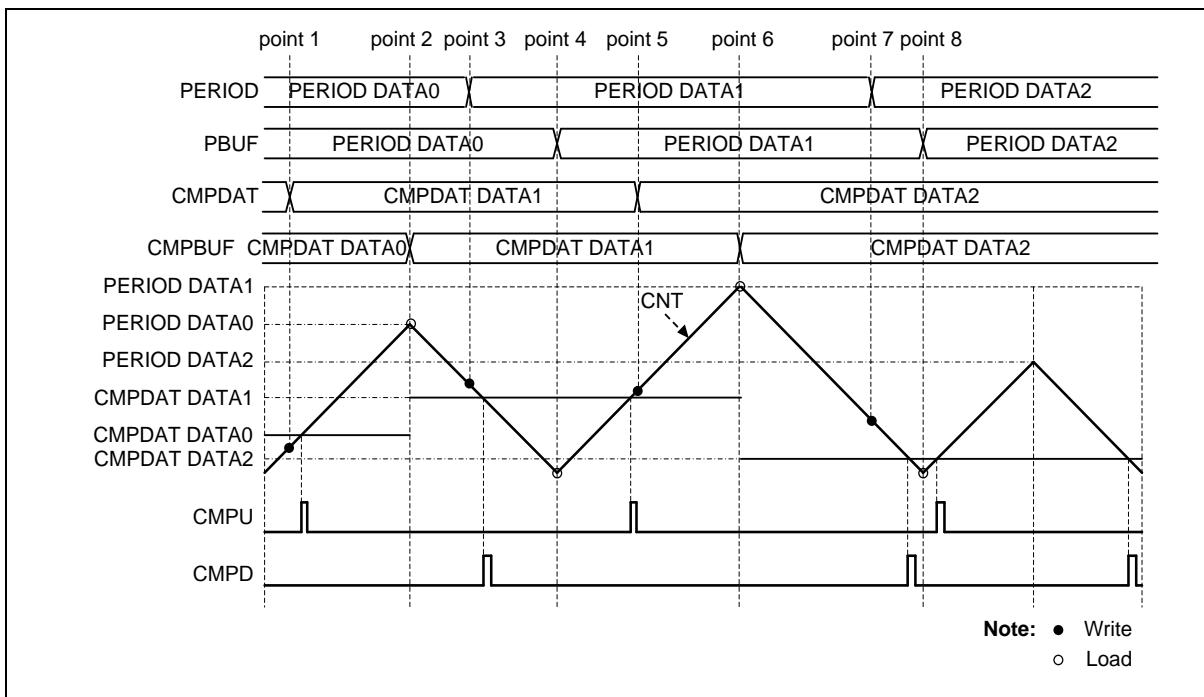


图 6.11-16 上下计数方式的中心装载模式

#### 6.11.5.12 EPWM 计数器工作模式

EPWM 计数器支持两种工作模式：单次（One-shot）模式和自动装载模式。如果 CNTMODEn (EPWM\_CTL1[21:16]) 位被置 1，EPWM 计数器将工作在 One-shot 模式，如果被设为 0 则工作在自动装载模式。

One-shot 模式，CMPDAT 和 PERIOD 应该先被写，然后设置 CNTENn (EPWM\_CNTEN[5:0]) 通道 n 相应位置 1 来使能 EPWM 预分频和计数开始运行。EPWM 计数器计完一个周期后，计数器的值将保持为 0。

在下一个单次周期，软件需要重新写一个新的 CMPDAT 比较值来重新启动。如果 one-shot 计数器保持计数，写 CMPDAT 将使得下一个单次接着下一个单次。此外，如果继续 one-shot 操作，对 CMPDAT 写两次，单次周期值将使用最后一次写入值作为 CMPDAT 比较值，也仅产生一个 one-shot 脉冲周期。图 6.11-17 是一个例子，操作步骤如下：

1. point 1 处软件写 PERIOD DATA1，硬件会立即装载 PERIOD DATA1 到 PBUF。
2. point 2 处软件写 CMPDAT DATA1（等于 CMPDAT DATA0），硬件会立即装载 CMPDAT DATA1 到 CMPBUF，该事件也会触发 one-shot。
3. point 3 处软件写 CMPDAT DATA2 会重新触发下一个 one-shot (持续 one-shot)。
4. point 4 处软件写 CMPDAT DATA3 来覆盖 CMPDAT DATA2，然后再触发下一个 one-shot 周期。
5. point 5 处周期装载 CMPDAT DATA3 到 CMPBUF。
6. point 6 处，如果在上一个周期中，没有新的 CMPDAT 比较值被写入，那么计数器的值将保持为 0。

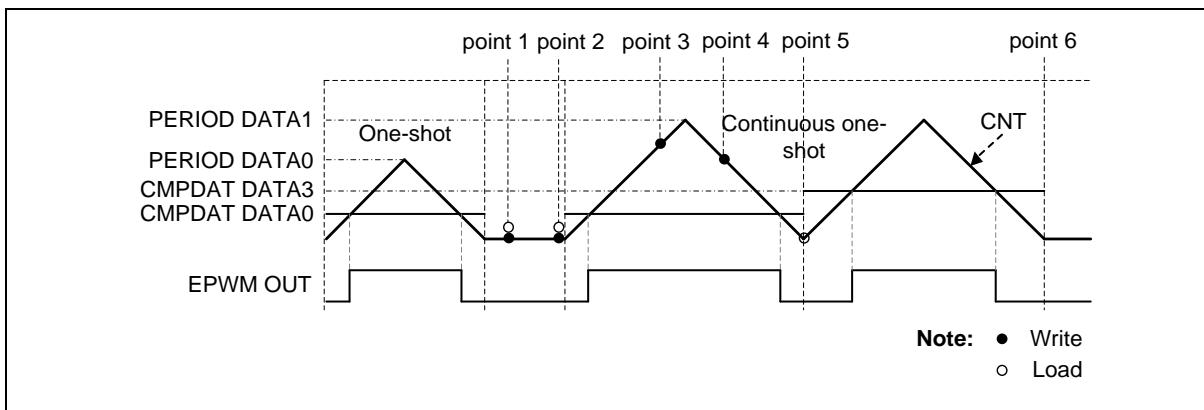


图 6.11-17 EPWM One-shot 模式输出波形

在自动装载模式，应该先写CMPDAT 和 PERIOD，然后 CNTEFn通道n位被置1来使能EPWM预分频器并开始计数。根据不同的装载模式，CLKPSC(EPWM\_CLKPSCn\_m[11:0]), PERIOD(EPWM\_PERIODn[15:0]) 和 CMP(EPWM\_CMPDATn[15:0])的值将被自动装载到它们的缓冲区。如果周期PERIOD被设为0，EPWM计数器将被设为0。

#### 6.11.5.13 EPWM脉冲发生器

EPWM脉冲发生器是用计数器和比较器事件来产生EPWM脉冲。这些事件包括：向上和向下计数方式的零点或周期点，上下计数方式的中心点，以及三种方式中的计数器等于比较值。作为上下计数方式，计数器有两个等于比较器的点，一个在向上计数，一个在向下计数。此外，互补模式有两个比较器值与计数器比较，因此在上下计数方式中有四个相等值，向上或向下计数方式各有两个。

通过设定EPWM\_WGCTL0 和 EPWM\_WGCTL1，每个事件点可以决定EPWM波形：不变(X)，为低(L)，为高(H)或切换(T)。用这些点可以很容易产生一个不规则的EPWM脉冲或可变波形，如图6.11-18。图中，EPWM是互补模式，两个比较器n和m来产生EPWM脉冲。n表示偶数通道0, 2, 4; m表示奇数通道1,3,5。n和m通道是互补组。互补模式用两个通道((CH0 和 CH1, CH2 和 CH3, CH4 和 CH5) 作为一组EPWM输出一个互补波形。CMPU表示当向上计数时CNT等于CMPDAT.CMPD表示当向下计数时CNT等于CMPDAT。

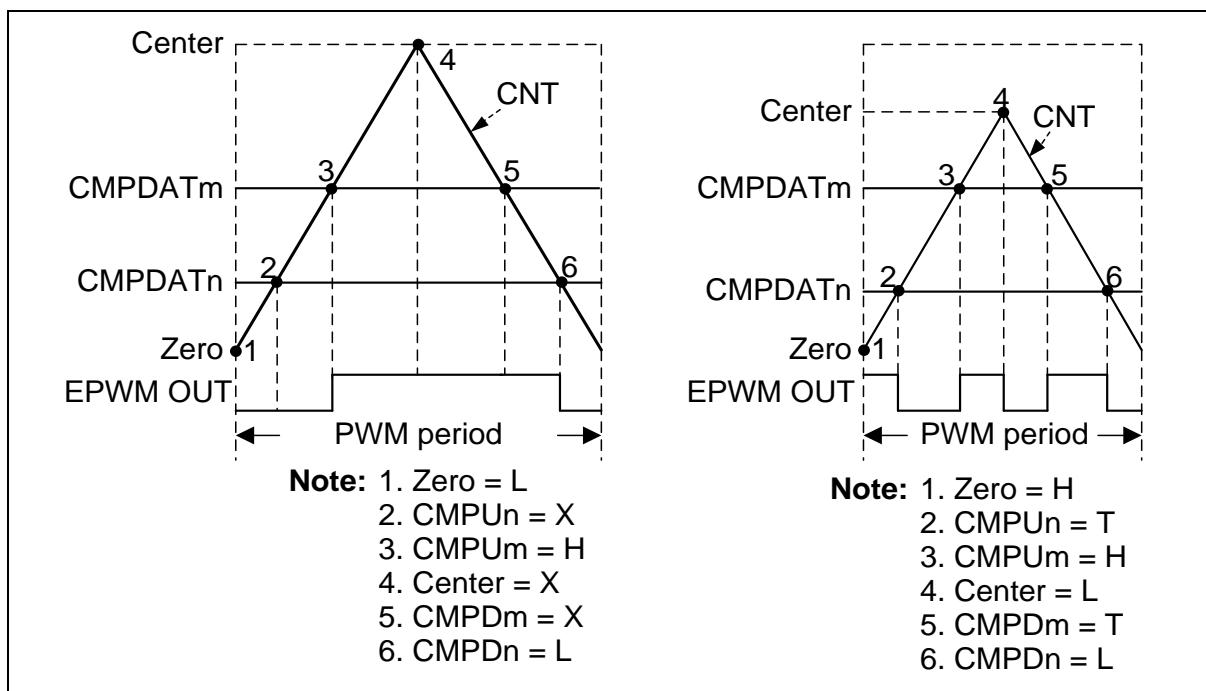


图 6.11-18 EPWM 脉冲产生

有时事件会被同时产生，因此，不同计数方式的事件优先级列表如下：向上计数方式（表 6.11-2），向下计数方式（表 6.11-3），上下计数方式（表 6.11-4）。通过事件优先级，使用者可以很容易产生0%到100%占空比的脉冲，图 6.11-19所示。

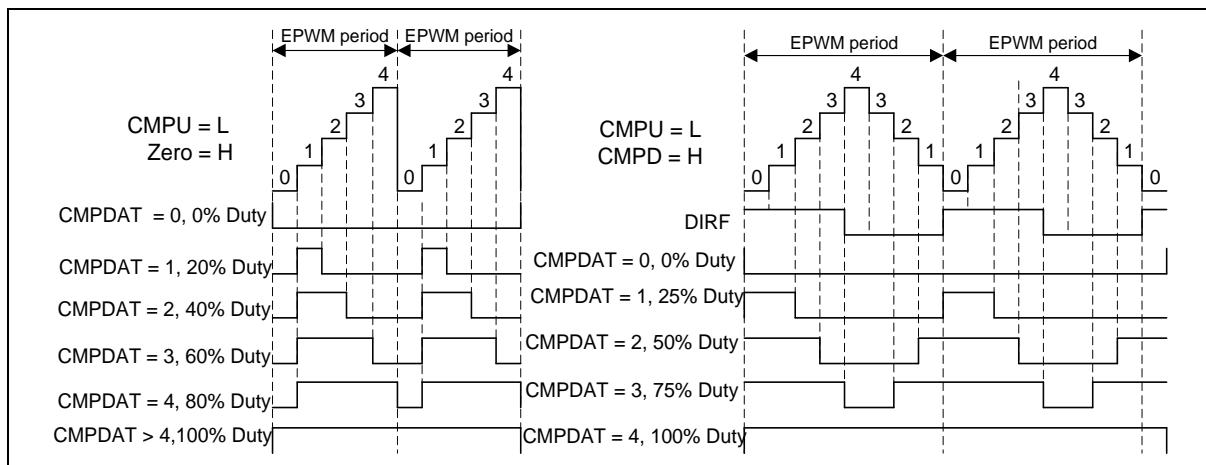


图 6.11-19 EPWM 0% 到 100% 占空比脉冲产生

优先级	向上事件
1 (最高)	周期点事件 (CNT = PERIOD)
2	奇数通道的向上比较事件 (CNT = CMPUm)
3	偶数通道的向上比较事件(CNT = CMPUn)

4 (最低)	零点事件 (CNT = 0)
--------	----------------

表 6.11-2 向上计数器 EPWM 脉冲发生事件优先级

优先级	向下事件
1 (最高)	零点事件 (CNT = 0)
2	奇数通道的向下比较事件(CNT = CMPDm )
3	偶数通道的向下比较事件(CNT = CMPDn )
4 (最低)	周期点事件 (CNT = PERIOD)

表 6.11-3 向下计数器 EPWM 脉冲发生优先级

优先级	向上事件	向下事件
1 (最高)	奇数通道的向上比较事件(CNT = CMPUm)	奇数通道的向下比较事件(CNT = CMPlm)
2	偶数通道的向上比较事件(CNT = CMPUn)	偶数通道的向下比较事件(CNT = CMPln)
3	零点事件(CNT = 0)	周期点(中点)事件 (CNT = PERIOD)
4	奇数通道的向下比较事件(CNT = CMPlm)	奇数通道的向上比较事件(CNT = CMPUm)
5 (最低)	偶数通道的向下比较事件(CNT = CMPln)	偶数通道的向上比较事件(CNT = CMPUn)

表 6.11-4 上下计数器 EPWM 脉冲产生事件优先级

#### 6.11.5.14 EPWM输出模式

EPWM支持两个输出模式：独立模式，可以应用于直流电机系统；带死区插入互补模式，可以由于交流感应电机和永磁同步电机。

#### 6.11.5.15 独立模式

默认EPWM在独立模式，当通道n相应位EPWMMODEn (EPWM\_CTL1[26:24])置0 独立模式被使能。独立模式中6个EPWM通道：EPWM\_CH0, EPWM\_CH1, EPWM\_CH2, EPWM\_CH3, EPWM\_CH4 和 EPWM\_CH5 都运行在各自的周期占空比，如下图所示：

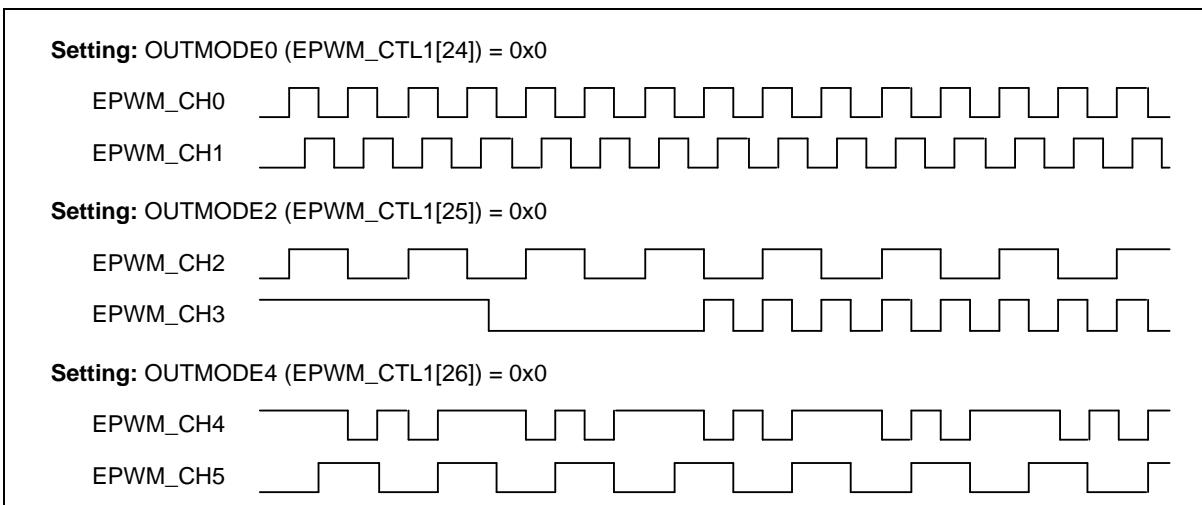


图 6.11-20 EPWM 独立模式波形

#### 6.11.5.16 互补模式

当互补通道EPWMMODEn (EPWM\_CTL1[26:24])相应位被置1，互补模式被使能。互补模式有3个EPWM发生器，模块中总共3组EPWM输出管脚。互补模式中内部奇数EPWM信号必须与相应偶数通道的EPWM信号互补。EPWM\_CH1将互补于EPWM\_CH0，EPWM\_CH3将互补于EPWM\_CH2，EPWM\_CH5将互补于EPWM\_CH4，如下图所示

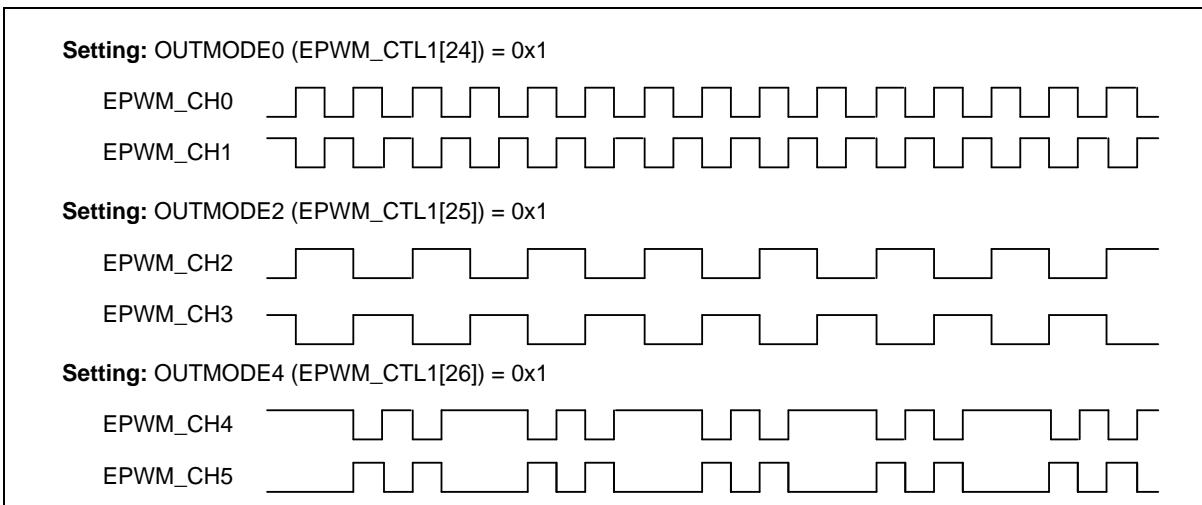


图 6.11-21 EPWM 互补模式波形

#### 6.11.5.17 EPWM输出功能

根据输出模式，有两种输出功能：组功能和高级输出控制同步功能。组功能，会强制EPWM\_CH2、EPWM\_CH4与EPWM\_CH0发生器同步，EPWM\_CH3、EPWM\_CH5与EPWM\_CH1同步，主要用于在DC和BLDC马达应用的更新占空比控制。此外，同步功能使得EPWM0和EPWM1任何一个通道在相位控制上，用户可以控制相值和方向。

#### 6.11.5.18 组功能

不管是在独立模式还是互补模式，当GROUPEN (EPWM\_CTL0[24])位被置1，组功能会被使能。该控制允许所有EPWM偶数通道通过EPWM\_PERIOD0和EPWM\_CMPDAT0寄存器输出可控，所有奇数

EPWM通道通过EPWM\_PERIOD1和EPWM\_CMPDAT1 寄存器输出可控。也就是，用户仅只需设置EPWM\_CH0 使得 EPWM\_CH0, EPWM\_CH2 和 EPWM\_CH4输出相同的脉冲，设置EPWM\_CH1 使得 EPWM\_CH1, EPWM\_CH3 和 EPWM\_CH5 输出相同的脉冲。如下图所示。当运行在组功能，独立模式下 EPWMMODE0, EPWMMODE2 和 EPWMMODE4必须被设置为0，或在互补模式下置为1。

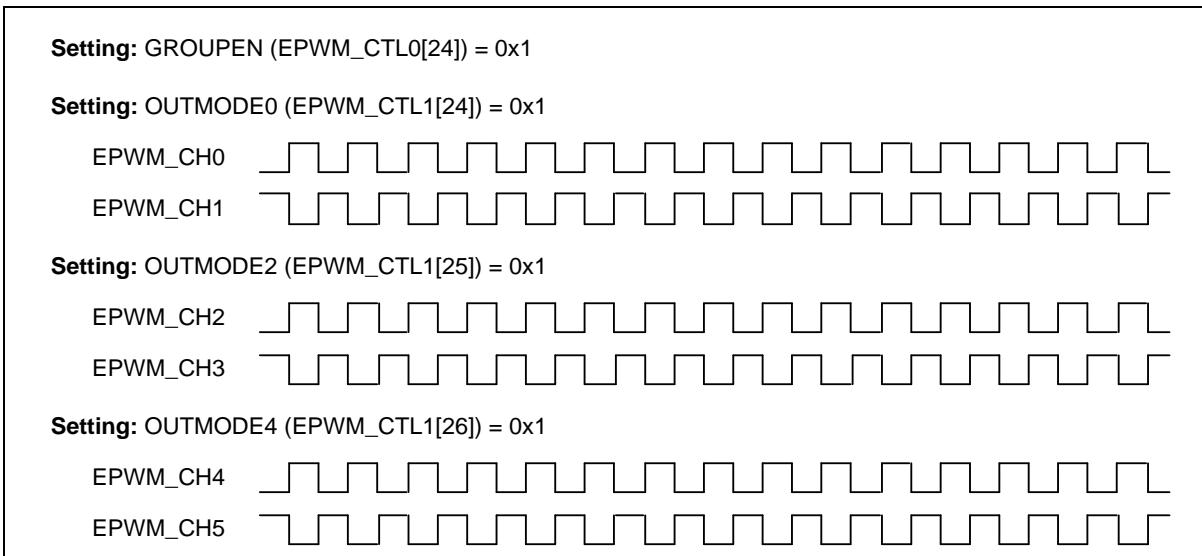


图 6.11-22 EPWM 组功能波形

#### 6.11.5.19 同步功能

同步功能仅当在互补模式下才可被使能。图 6.11-24是计数器同步功能框图。每个EPWM组计数器有一个SYNC\_IN 和一个SYNC\_OUT信号。第一个EPWM0组计数器SYNC\_IN来自于EPWM0\_SYNC\_IN管脚，其它的来自于上一个EPWM组计数器的SYNC\_OUT信号。来自EPWM0\_SYNC\_IN管脚的输入信号会被3位的滤波器滤波，如图 6.11-23。此外通过设置SINPINV (EPWM\_SYNC[23])可以实现对输入信号的极性控制。噪音滤波采样时钟可以通过设置SFLTCNT (EPWM\_SYNC[22:20])来选择以适应不同的噪音特性。用户可以定义多少个采样时钟周期来识别一个有效的SYNC\_IN信号边沿。配置SNFLTEN (EPWM\_SYNC[16])会使能噪音滤波功能，默认是禁止的。

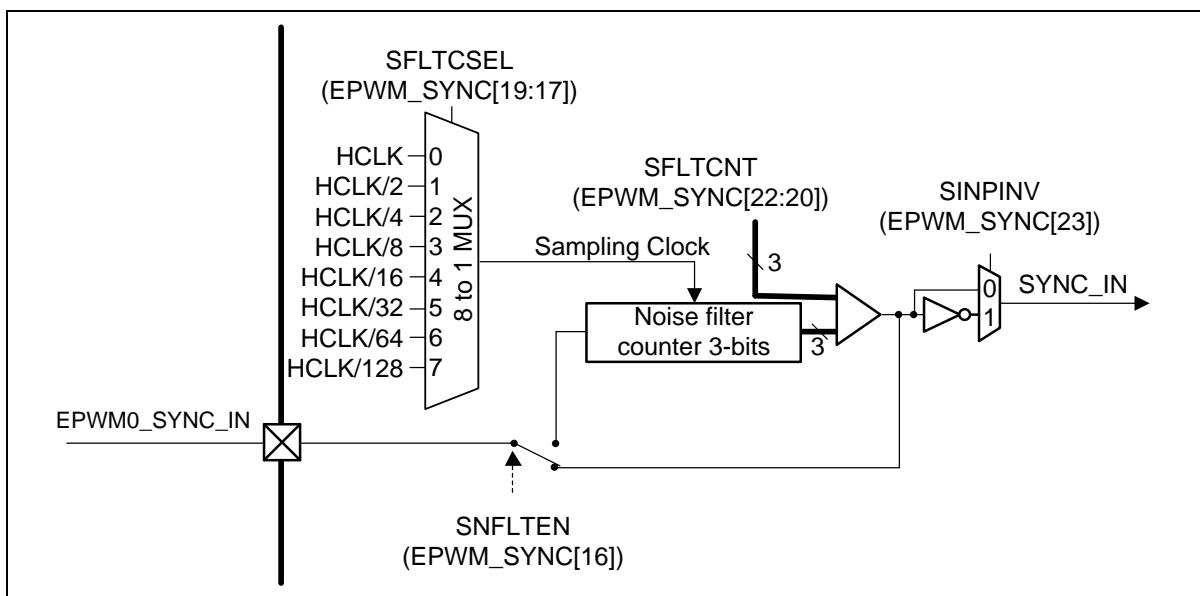


图 6.11-23 EPWM SYNC\_IN 噪音滤波框图

用户可以使用 **SINSRCn** (EPWM\_SYNC[13:8]) 来选择同步源。当 **SINSRCn** 位置 0，当 **EPWM0\_SYNC\_IN** 管脚是高或设置 **SWSYNCn** (EPWM\_SWSYNC[2:0]) 为 1，可以产生 **SYNC\_IN** 信号给下一个计数器同步。同步源也可以被选择为 **CNT = 0** 或 **CNT = EPWM\_CMPDATm** (如果是上下计数器类型，在一个EPWM周期中将同步两次)来触发一个同步事件或禁止 **SYNC\_OUT** 信号。

当 **PHSENn** (EPWM\_SYNC[2:0]) 被使能，且同步源有事件发生，计数器将从 **PHS** (EPWM\_PHS[15:0]) 寄存器装载数据。该方法将在同时同步计数器到不同相位。在上-下计数模式，同步后用户可以设置 **PHSDIRn** (EPWM\_SYNC[26:24]) 来控制计数方向。尽管同步功能可以在相位中同步通道，但是 EPWM 刚使能时不能使用。要在同一时刻启动这些计数器，用户则必须设置 EPWM 同步启动控制寄存器 (EPWM\_SSCTL[5:0]) 来使能这些一起同步的通道计数器，然后设置 EPWM 同步启动触发寄存器 **CNTSEN** (EPWM\_SSTRG[0])。

在应用当中，请不要同时使用组和同步功能，因为此时同步功能将无效。

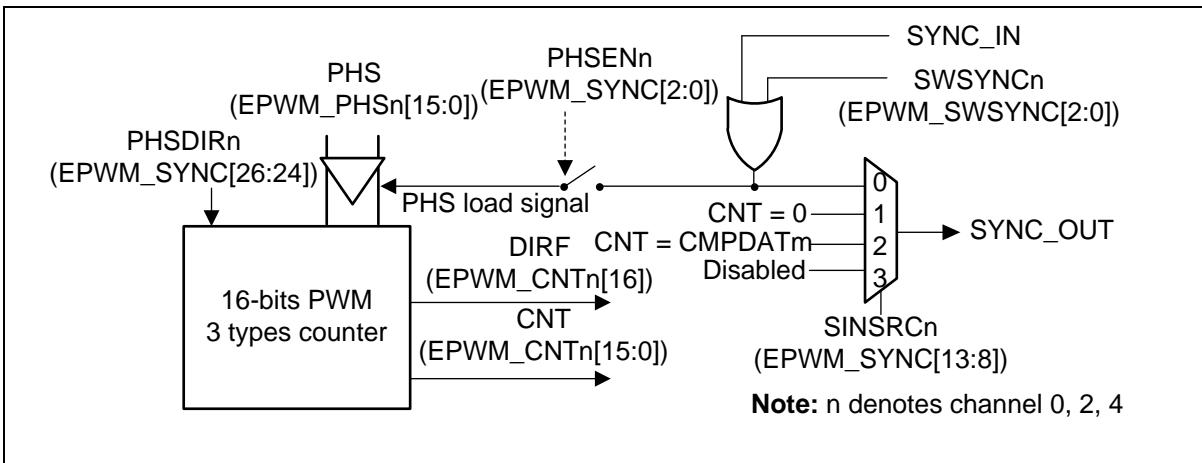


图 6.11-24 EPWM 计数器同步功能框图

图 6.11-25 是上-下计数模式中同步功能的例子。例中，同步源来自外部 **EPWM\_SYNC\_IN**。刚开始，**EPWM\_CH0**, **EPWM\_CH2** 和 **EPWM\_CH4** 处于相同的相位。然后在 A 点，**EPWM\_SYNC** 输入信号成

为一个同步事件，导致所有计数器相位切换并且计数方向改变。如果在同步事件到来前修改计数器，用户需要在PHS (EPWM\_PHSn[15:0]) 中设置相应的相位值以及在PHSDIRn (EPWM\_SYNC[26:24])中设置计数方向。在这种情况下，可以在同步以后通过设置相应通道n计数器的计数方向，实现移动1/3的相位，如图 6.11-25左边所示。

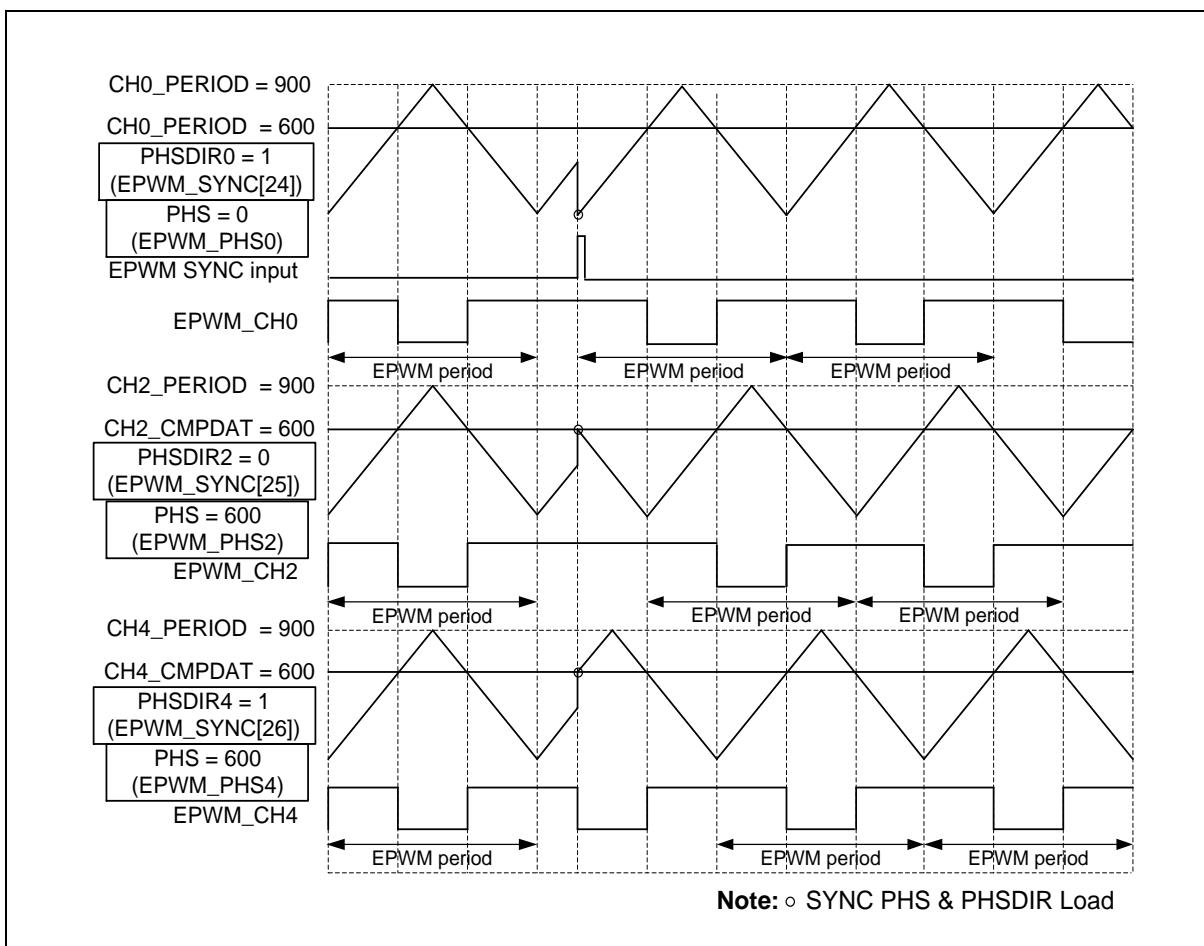


图 6.11-25 EPWM 同步源来自 SYNC\_IN 信号的同步功能

#### 6.11.5.20 EPWM输出控制

EPWM脉冲产生后，有4到6步来控制EPWM通道输出。独立模式中，屏蔽，刹车，管脚极性和输出使能4步如图 6.11-26所示。互补模式中，在以上4步前增加2步：互补通道和死区插入，如图 6.11-27所示。

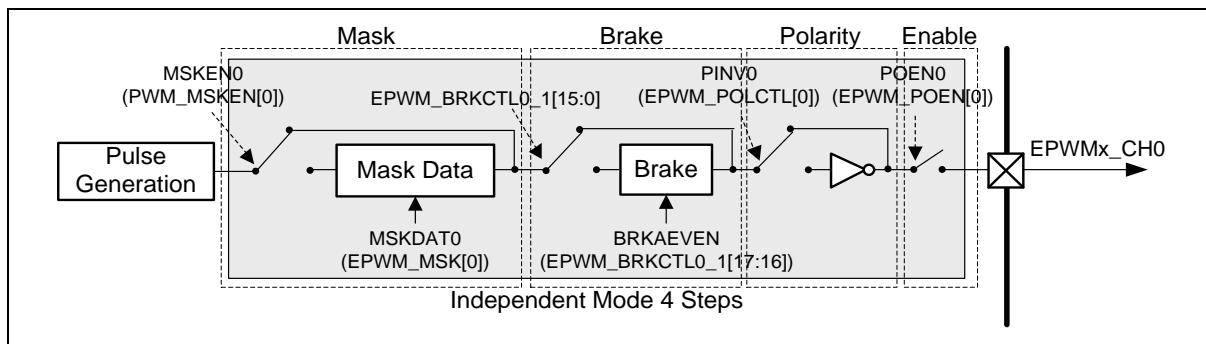


图 6.11-26 EPWMx\_CH0 独立模式下的输出控制

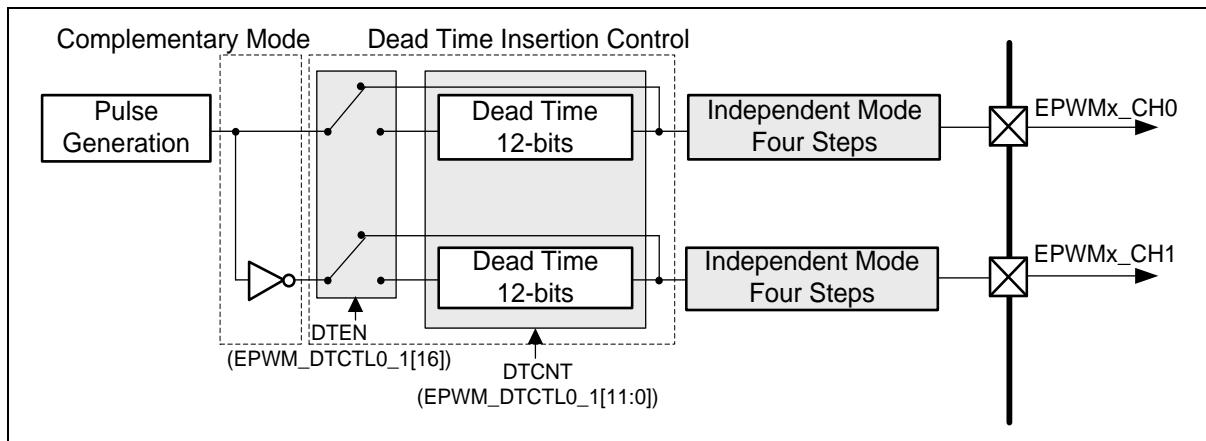


图 6.11-27 EPWMx\_CH0 和 EPWMx\_CH1 在互补模式下的输出控制

### 6.11.5.21 死区插入

互补应用中，互补通道也许用来驱动外部设备例如开关电源，在互补输出通道间死区发生器插入一个低电平时段叫“死区时间”来安全驱动设备，防止系统或设备烧坏。因此死区控制是正确操作互补系统的重要机制。通过设置通道n DTEN (EPWM\_DTCTLn[16])的相应位来使能死区功能，设置DTCNT (EPWM\_DTCTLn[11:0])控制死区时间周期，死区时间可以通过以下公式计算：

$$\text{死区时间} = (\text{DTCNT} (\text{EPWM}_\text{DTCTLn}[11:0])+1) * \text{EPWMx_CLK 周期}$$

死区插入时钟源可以通过设置DTCKSEL (EPWM\_DTCTLn\_m[24])为1来选择来自预分频器输出。默认时钟源来自EPWM\_CLK，就是预分频器的输入，死区时间可以通过下面公式计算：

$$\text{死区时间} = (\text{DTCNT} (\text{EPWM}_\text{DTCTLn}[11:0])+1) * \\ (\text{CLKPSC} (\text{EPWM_CLKPSCn}[11:0])+1) * \text{EPWMx_CLK 周期}$$

请注意EPWM\_DTCTLn\_m 是写保护寄存器。

图 6.11-28 表示一对EPWM信号的死区插入

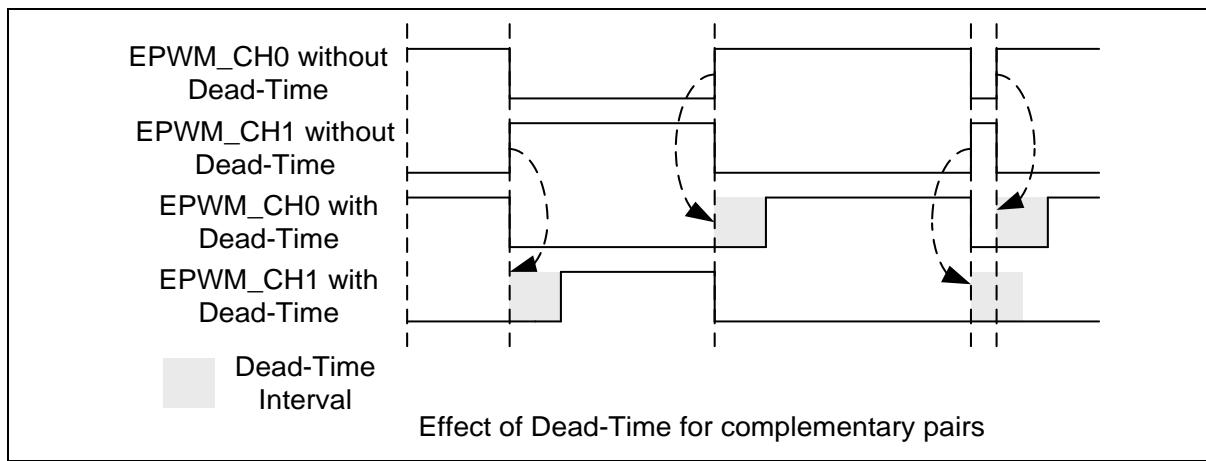


图 6.11-28 死区插入

#### 6.11.5.22 EPWM屏蔽输出功能

通过设置EPWM屏蔽使能控制寄存器(EPWM\_MSKEN)和EPWM屏蔽数据寄存器(EPWM\_MSK)可以对每个EPWM的输出手动覆盖。通过这些设置，EPWM通道的输出可以被强制设定为特定的逻辑状态，而与比较单元无关。EPWM屏蔽位用于控制各类电子整流电机(ECM)，如BLDC电机，非常有用。EPWM\_MSKEN寄存器有六个位，MSKENn (EPWM\_MSKEN[5:0])。如果MASKENn置高，通道n输出将被覆盖。EPWM\_MSK寄存器有六个位，MSKDATn (EPWM\_MSK[5:0])。当通道被覆盖，MSKDATn位值决定EPWM通道n的输出状态值。图 6.11-29显示如何将EPWM屏蔽控制器用于覆盖特性的例子。

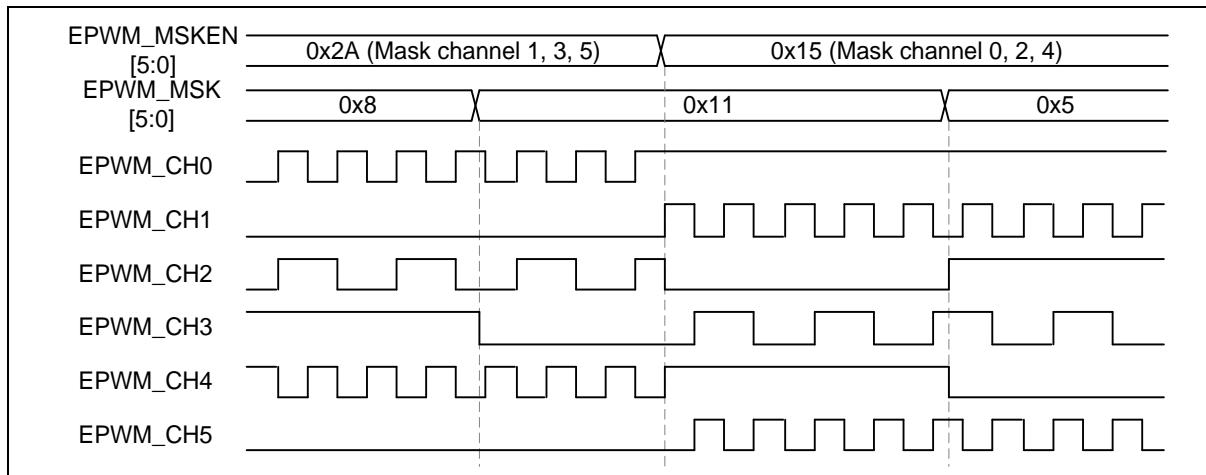


图 6.11-29 屏蔽控制波形图解

#### 6.11.5.23 EPWM刹车

每个EPWM模块有两个外部输入刹车控制信号。用户可以通过BNF寄存器的BKxSRC位( $x=0,1, y=0,1$ )选择激活刹车的管脚源来自EPWM $x$ \_BRAKE $y$ 管脚。外部信号会被一个3位的滤波器滤波。用户可以通过BNF寄存器的BRKxNFEN位使能噪音滤波器功能。噪音滤波器的采样时钟可以通过BNF寄存器的BRKxNFSEL位选择以适应不同的噪音特性。通过设置BRKxFCNT，用户可以定义滤波器需要多少次采样时钟周期来确认有效刹车信号。

此外，外部信号可以通过设置BRKxPINV( $x$ 表示输入管脚0或1)被翻转，来实现刹车控制型号的极性设置。设置BRKxPINV位为0，当EPWM $x$ \_BRAKE $y$ ( $x=0,1, y=0,1$ )管脚状态从低到高时，刹车事件会发

生。设置BRKxPINV位为1，当EPWMx\_BRAKEy管脚状态从高到低时，刹车事件会发生。

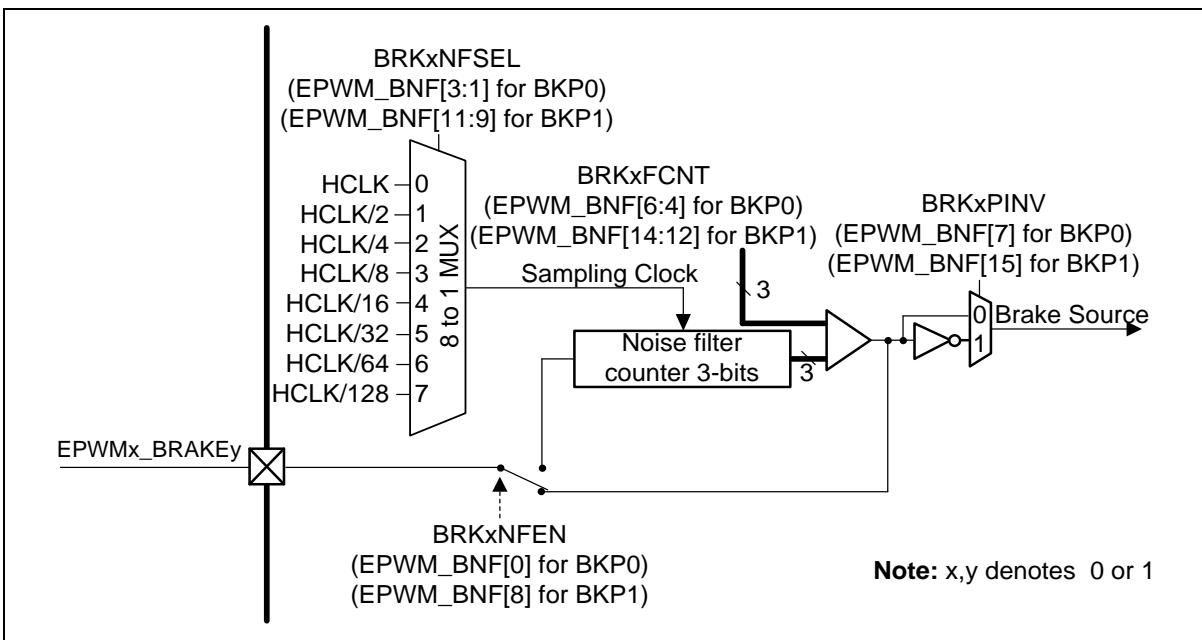


图 6.11-30 刹车噪音滤波器方块图

对于互补模式，一旦刹车事件发生，通常需要设置一个安全输出状态到互补输出组。

每组互补通道共享一个刹车功能，图 6.11-31。当刹车事件发生时，用户可以设置偶数通道的BRKAEVEN (EPWM\_BRKCTL0\_1[17:16])和奇数通道的BRKAODD (EPWM\_BRKCTL0\_1[19:18])，确保通道输出安全状态。有两个刹车检测：边沿检测和电平检测。当边沿检测发现刹车信号并且BRKEIENn\_m (EPWM\_INTEN1[2:0])被使能，刹车功能产生BRK\_INT中断。这个中断需要用软件清除，且在中断清除后BRKESTS<sub>n</sub> (EPWM\_INTSTS1[21:16])刹车状态将保持直到下一个EPWM周期开始。通过电平检测，刹车功能也可以用另一种方式操作。一旦电平检测发现刹车信号并且BRKLIEEn\_m (EPWM\_INTEN1[10:8])被使能，刹车功能将产生BRK\_INT中断，但是当刹车信号回到高电平，当前EPWM周期结束后，EPWM输出将变为正常，BRKLSTS<sub>n</sub> (EPWM\_INTSTS1[29:24])位被自动清除。

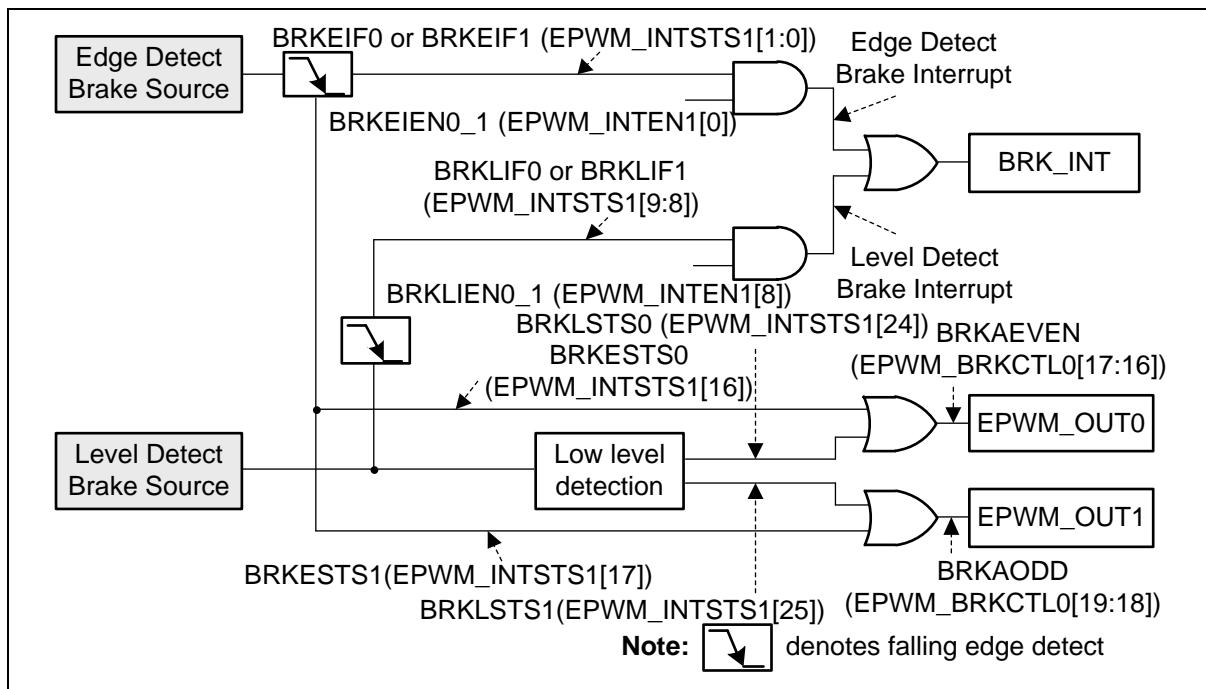
图 6.11-31 EPWM<sub>x</sub>\_CH0 和 EPWM<sub>x</sub>\_CH1 通道组 刹车方块图

图 6.11-32 表示 EPWM<sub>x</sub>\_CH0 和 EPWM<sub>x</sub>\_CH1 通道组的边沿检测波形。这种情况下，边沿检测到两次刹车事件。当刹车事件发生 BRKEIF0 和 BRKEIF1 都被置 1 并且 BRKESTS0 和 BRKESTS1 也被置 1 表明 EPWM<sub>x</sub>\_CH0 和 EPWM<sub>x</sub>\_CH1 处于刹车状态。当第一个事件发生，软件需写 1 清 BRKEIF0。然后在下一个 EPWM 周期开始时硬件清除 BRKESTS0。此时即使刹车事件依旧存在，EPWM<sub>x</sub>\_CH0 仍然输出正常波形。第二个事件也触发相同标志，但此时软件需写 1 去清 BRKEIF1，此后在下个 EPWM 周期开始 EPWM<sub>x</sub>\_CH1 正常输出。

相对于边沿检测例子，图 6.11-33 表示 EPWM<sub>x</sub>\_CH0 和 EPWM<sub>x</sub>\_CH1 通道组的电平检测波形。这种情况下，BRKLIF0 和 BRKLIF1 只可以表示刹车事件已发生。在下一个 EPWM 周期开始时不管 BRKLIF0 和 BRKLIF1 是什么状态，BRKLSTS0 和 BRKLSTS1 的刹车状态都会被自动恢复。

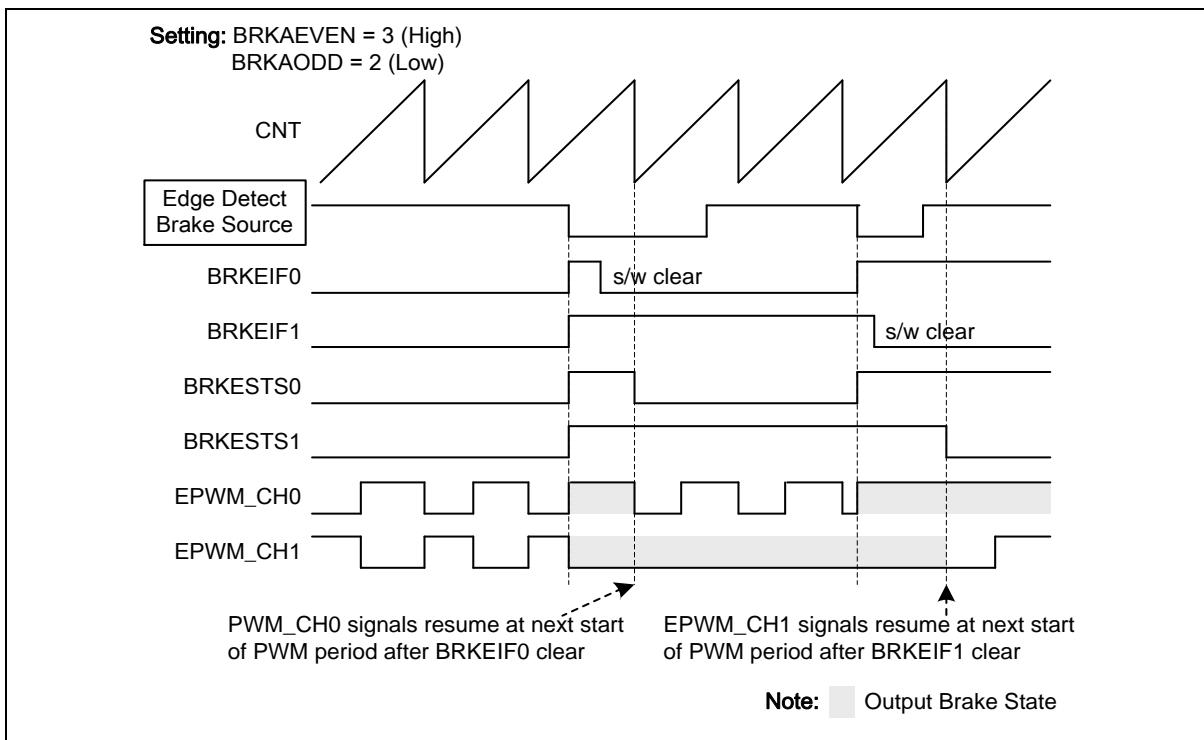


图 6.11-32 EPWMx\_CH0 和 EPWMx\_CH1 通道组的边沿检测波形

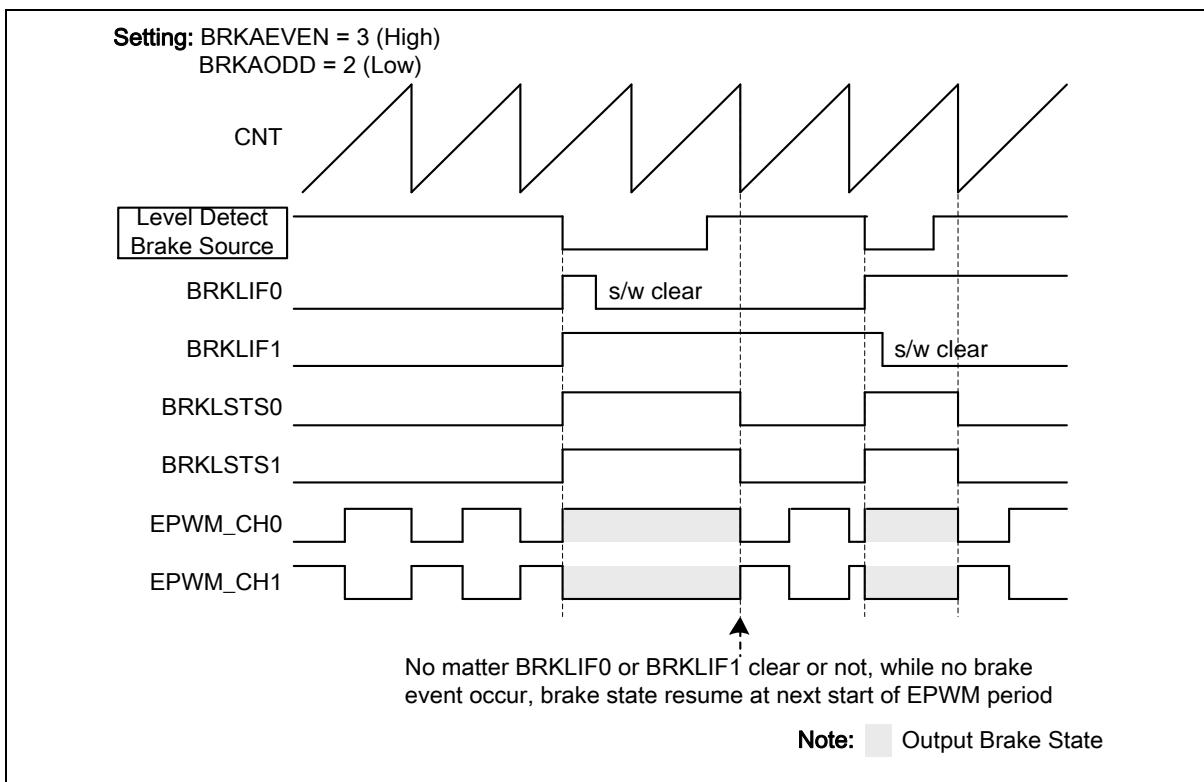


图 6.11-33 EPWMx\_CH0 和 EPWMx\_CH1 通道组的电平检测波形

两种检测器检测七个同样的刹车源：两个从外部输入的信号，两个来自模拟比较器(ACMP)，一个来自

EADC 结果监视器(EADCRM)，一个来自系统故障和一个来自软件触发，如图 6.11-34。仅当内部 ACMP0\_O 或 ACMP1\_O 信号从低到高，ACMP 刹车源才会被侦测到。

在以上描述的几个刹车源中，来自系统故障刹车源可以被指定为几种不同系统故障条件，这些故障条件，包括时钟故障，欠压侦测，SRAM 奇偶校验错误和内核锁死。图 6.11-35 显示通过设置相应使能位，系统故障条件可以作为一个系统故障刹车源发送给 EPWM 刹车。

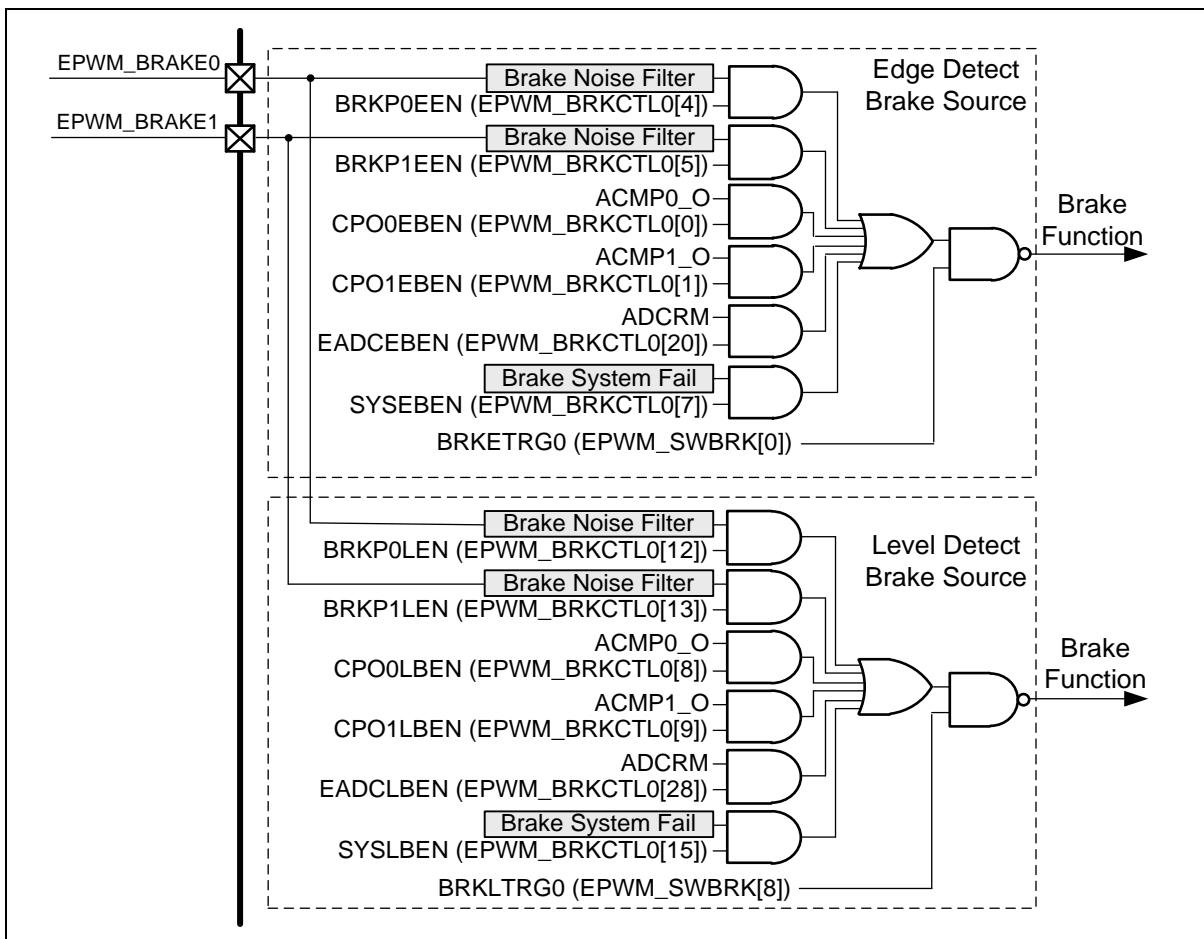


图 6.11-34 刹车源框图

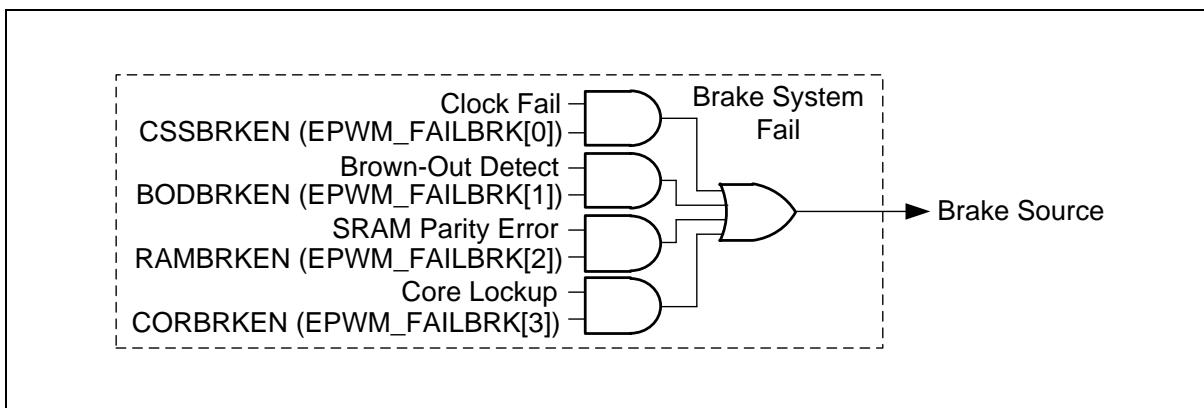


图 6.11-35 系统故障刹车框图

### 6.11.5.24 LEB 功能

EPWM Leading edge blanking (LEB)的功能是用在 EPWM 的 刹车源 是 ACMP\_O时，当 EPWM 通道输出管脚 与 ACMP 输入管脚相邻时，EPWM 输出的跳变 可能会产生干扰 造成 ACMP 误判而触发 EPWM brake。因此，EPWM LEB 提供使用者可以设定一段时间 (blanking Time)，在这段时间内忽略 ACMP 送过来的触发信号。设置 LEBEN (EPWM\_LEBCTL[0]) 使能该功能，LEB 源来自 EPWM\_CH0, EPWM\_CH2 和 EPWM\_CH4, SRCENn (EPWM\_LEBCTL[10:8]) 作为输入源使能。LEB功能blanking Time由LEBCNT (EPWM\_LEBCNT[8:0])决定。当LEB侦测到触发边沿，blanking Time会从LEBCNT+1 计数到 0，计数器时钟基于ECLK。如果新的触发事件发生，blanking 计数器会复位到 LEBCNT且再次向下计数 。LEB 触发边沿可以是上升沿，下降沿或上升沿和下降沿都可以，通过设置TRGTYPE (EPWM\_LEBCTL[17:16])来决定。图 6.11-36 所示由EPWM\_CH0 和 EPWM\_CH4引起的LEB。

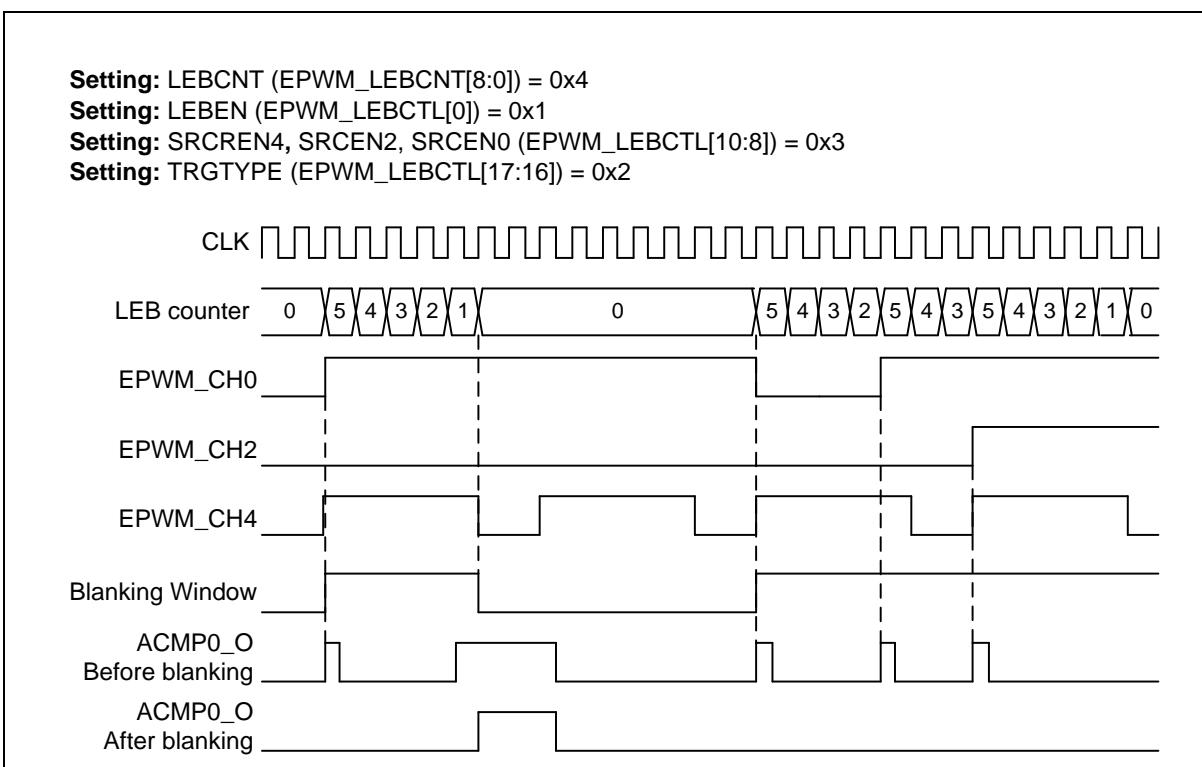


图 6.11-36 EPWM LEB 功能波形

### 6.11.5.25 极性控制

每个EPWM端口从EPWM\_CH0 到 EPWM\_CH5都有一个独立极性控制模块来配置EPWM输出的极性状态。默认EPWM输出是高，意味着EPWM 关闭状态时为低，打开时为高。对每个独立的EPWM通道，EPWM的极性可以通过EPWM负极控制寄存器(EPWM\_POLCTL)来设置。图 6.11-37所示，EPWM不同极性设置开始前的初始状态。

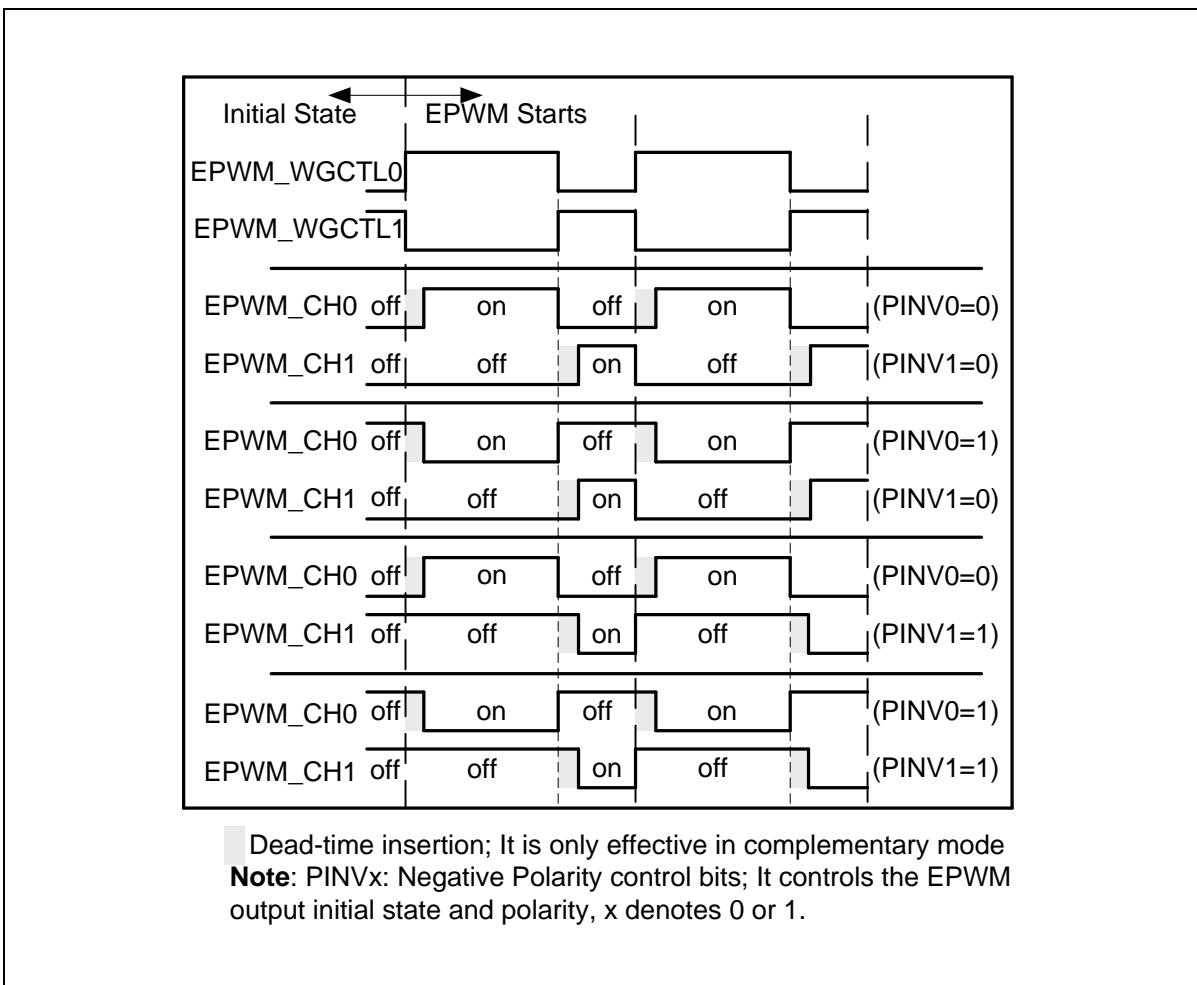


图 6.11-37 带上升沿死区插入初始状态和极性控制

#### 6.11.5.26 EPWM 中断发生器

每个EPWM有三个独立中断，如图 6.11-39.。

第一个 EPWM 中断(EPWM\_INT)来自 EPWM 互补组事件。计数器可以产生零点中断标志 ZIFn (EPWM\_INTSTS0[5:0]) 和周期点中断标志 PIFn (EPWM\_INTSTS0[13:8])。当EPWM通道n 计数器值等于存在EPWM\_CMPDATn比较器值的时候，根据计数方向将触发不同的中断标志。如发生在向上计数则向上中断标志 CMPUIFn (EPWM\_INTSTS0[21:16]) 被置 1，如果相反方向则向下计数中断标志 CMPDIFn (EPWM\_INTSTS0[29:24]) 被置 1。如果相应的中断使能位置 1，事件触发将产生中断信号。

EPWM\_INT可以使用EPWM\_IFAn (n=0~5)寄存器来累加已经触发的中断标志次数。通过设置IFAEN (EPWM\_IFAn[31], n=0~5)中的一个为 1 来使能累加器。EPWM\_INT会切换中断源，从每个事件触发中断切换为每次到达累加次数触发中断。

通过设置IFASEL (EPWM\_IFAn[29:28], n=0~5)，用户可以选择4个中断源之一来累加每个通道的中断标志次数，且中断标志的次数会和IFACNT (EPWM\_IFAn[15:0], n=0~5)比较。如果用户使能IFAIEn (EPWM\_AINTEN[n], n=0~5)位，当中断累加器等于 IFACNT 会置位 IFAIFn (EPWM\_AINTSTS[n], n=0~5)作为EPWM\_INT信号。每个通道的累加器中断也可以作为PDMA的请求源。图 6.11-38 是通道0 使用EPWM\_IFA0寄存器来每发生IFCNT0+1次中断事件输出一次EPWM\_INT的例子。

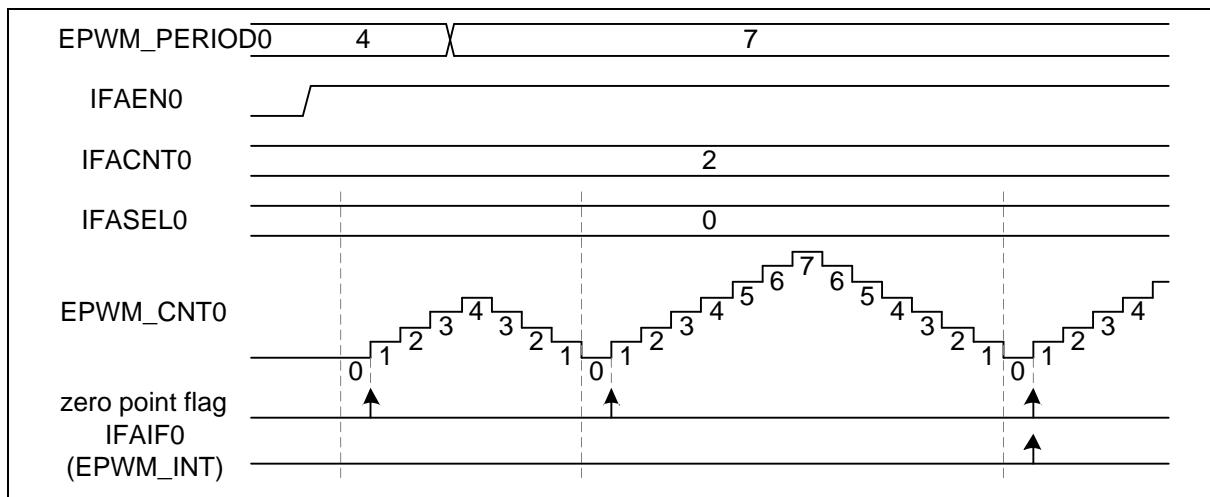
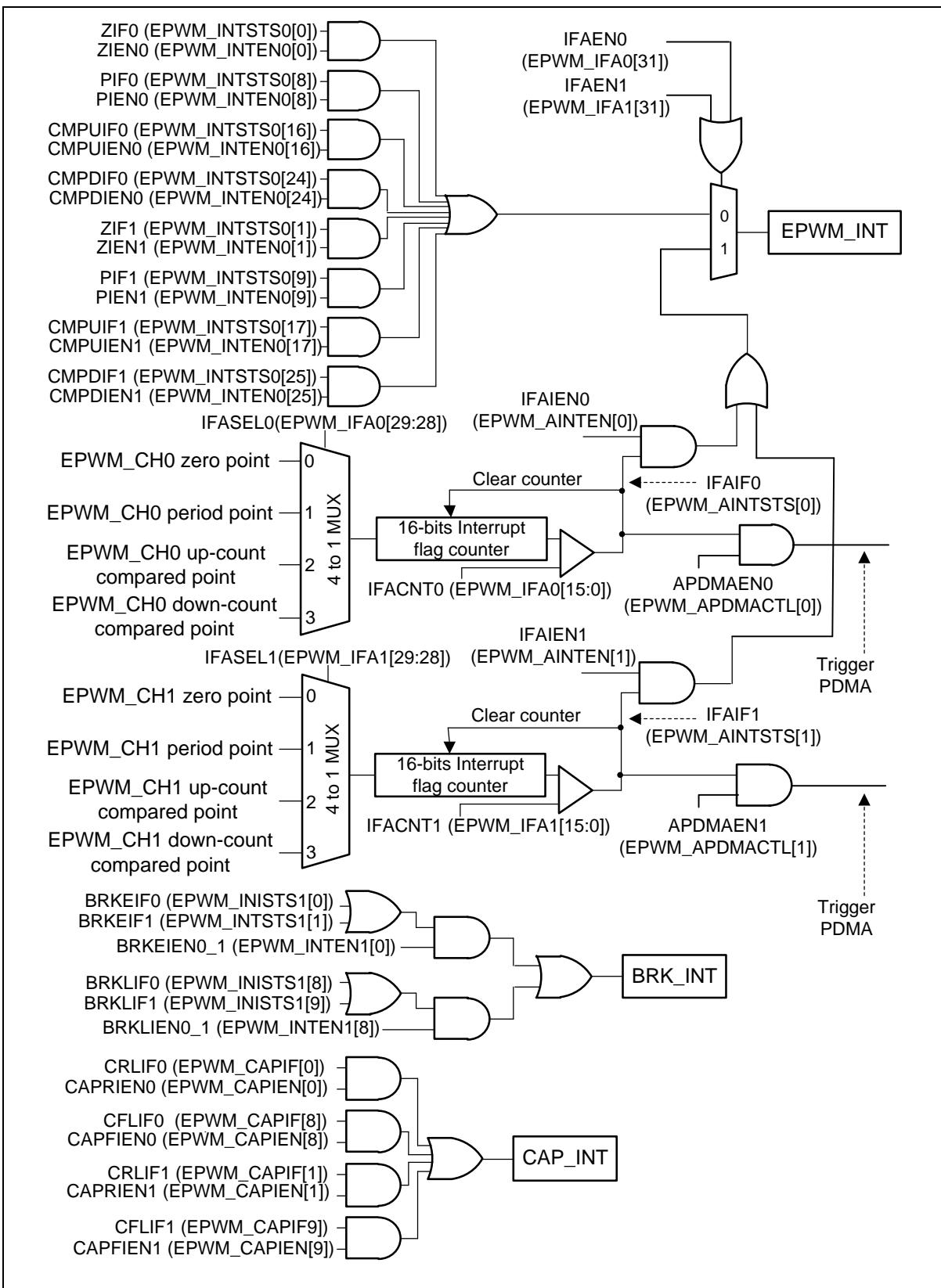


图 6.11-38 EPWMx\_CH0 累加中断波形

第二个中断是捕获中断(CAP\_INT)，在NVIC中与EPWM\_INT共用中断入口。当CAPRLIFn (EPWM\_CAPIF[5:0])被触发，并且捕获上升沿中断使能位CAPRIENn (EPWM\_CAPIEN[5:0])置1，可以产生CAP\_INT中断。或者在下降沿条件，当捕获下降沿中断使能位CAPFIENn (EPWM\_CAPIEN[13:8])置1，CAPFLIFn (EPWM\_CAPIF[13:8])标志可以被触发。

最后是刹车中断(BRK\_INT)，在EPWM刹车章节有BRK\_INT详细描述。

图 6.11-39表示EPWM中断架构。

图 6.11-39 EPWM<sub>x</sub>\_CH0 和 EPWM<sub>x</sub>\_CH1 组中断架构图

### 6.11.5.27 EPWM 触发 EADC/DAC 产生器

EPWM 可以作为一个 EADC 转换触发源。每一个 EPWM 通道组都共享同样的触发源。设置 EPWM\_EADCTS0 和 EPWM\_EADCTS1 寄存器的 TRGSELn 来选择触发源，TRGSELn 有 TRGSEL0, TRGSEL1...TRGSEL5，它们分别位于 EPWM\_EADCTS0[3:0], EPWM\_EADCTS0[11:8], EPWM\_EADCTS0[19:16], EPWM\_EADCTS0[27:24], EPWM\_EADCTS1[3:0] 和 EPWM\_EADCTS1[11:8]。设置相应的 EPWM\_EADCTS0 和 EPWM\_EADCTS1 寄存器的 TRGENn 来使能触发输出到 EADC，TRGENn 是 TRGEN0, TRGEN1, ..., TRGEN5，他们分别位于 EPWM\_EADCTS0[7], EPWM\_EADCTS0[15], EPWM\_EADCTS0[23], EPWM\_EADCTS0[31], EPWM\_EADCTS1[7] 和 EPWM\_EADCTS1[15]。数字 n(n = 0, 1, .., 5) 表示 EPWM 通道数。

一对通道可以有 16 个 EPWM 事件触发源可选择，如图 6.11-40。图 6.11-41 是在上下计数模式触发 EADC 的时序图。

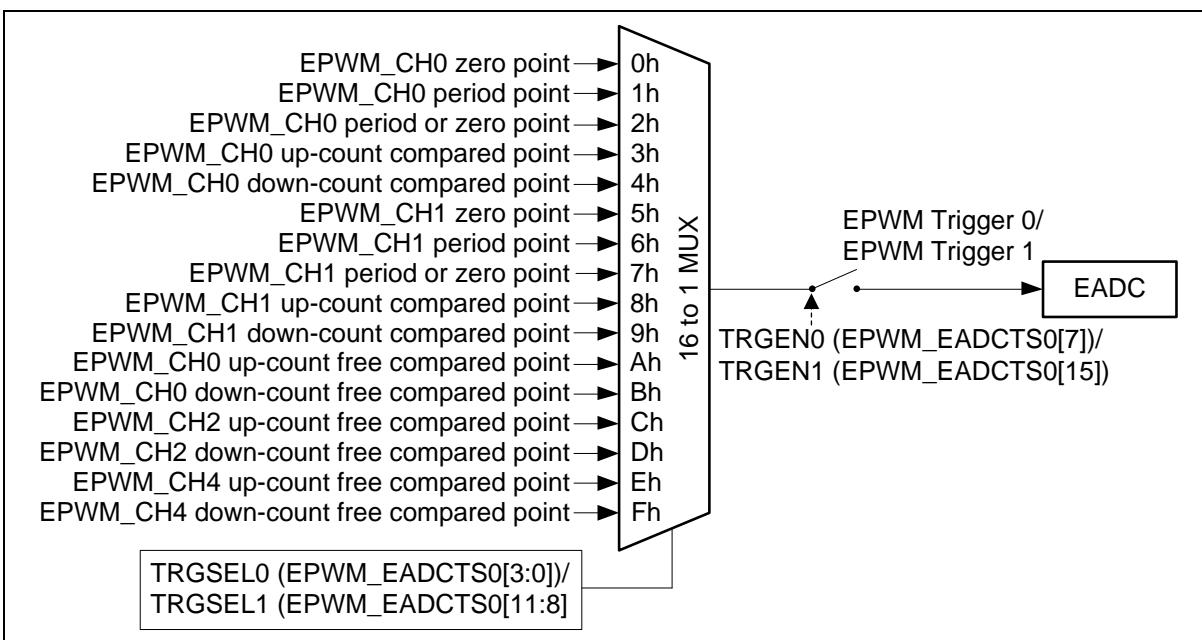


图 6.11-40 EPWMx\_CH0 和 EPWMx\_CH1 组触发 EADC 框图

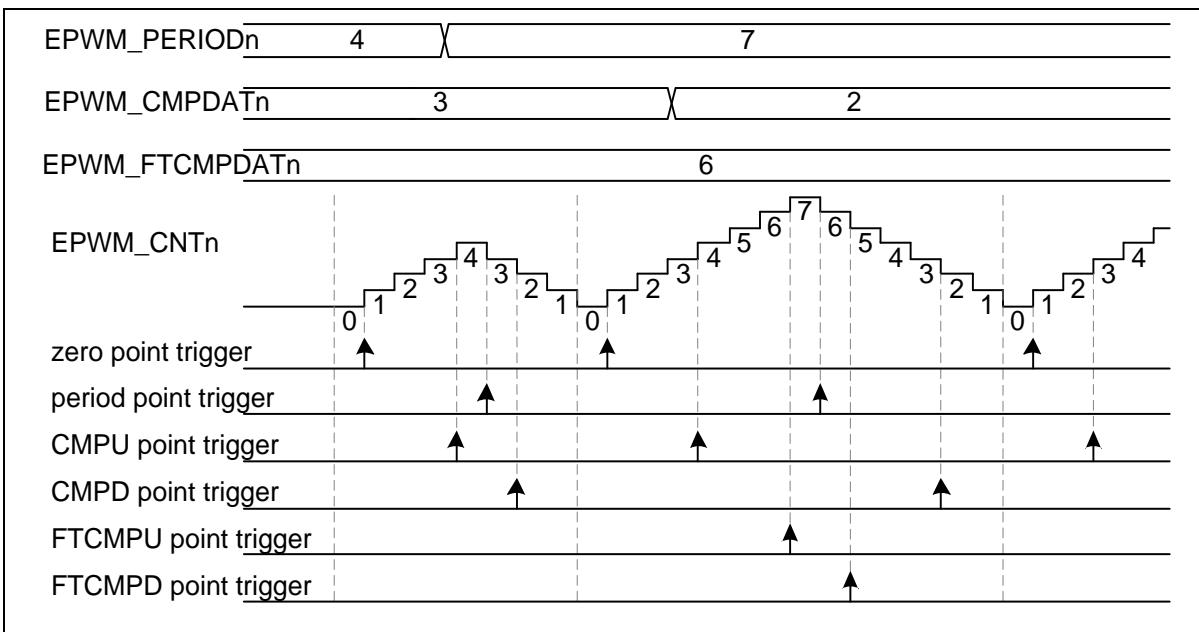


图 6.11-41 上下计数方式 EPWM 触发 EADC 的时序波形

EPWM也可以用来触发DAC转换。每个EPWM对的通道(CH0和CH1, CH2和CH3, CH4和CH5)产生一个触发信号。使用EPWM触发DAC使能寄存器(EPWM\_DACTRGEN)可以决定哪一个点来触发DAC。EPWM触发DAC的时序类似于触发EADC。然而，DAC触发功能不包括来自于与FTCMPDAT比较的触发事件。也就是，FTCMPDATU和FTCMPDATD没有触发点。如EADC触发所示。

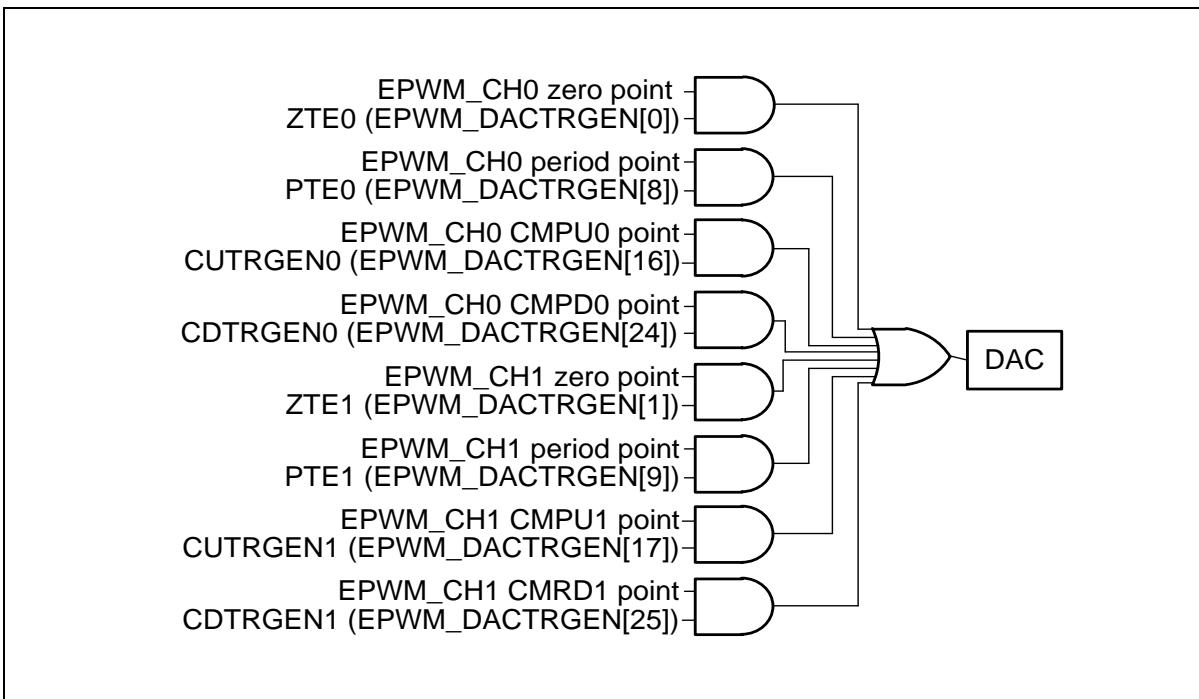


图 6.11-42 EPWM\_CH0 和 EPWM\_CH1 组触发 DAC 框图

### 6.11.5.28 捕获操作

捕获输入通道和EPWM输出通道共用管脚和计数器。计数器可以是上或下计数方式。如果输入通道有上升沿或下降沿跳变时，捕获功能将EPWM计数器值分别锁存到RCAPDATn (EPWM\_RCAPDATn[15:0])或FCAPDATn (EPWM\_FCAPDATn[15:0])寄存器。如果上升沿或下降沿锁存发生并且相应通道n的上升沿或下降沿中断使能位被设置，捕获功能也将产生一个中断CAP\_INT (使用 EPWM\_INT向量)，在这里CAPRIENn (EPWM\_CAPIEN[5:0])设置上升沿中断使能，CAPFIENn (EPWM\_CAPIEN[13:8])设置下降沿中断使能。当上升沿或下降沿锁存发生，相应的EPWM计数器是否将重载EPWM\_PERIODn值，这取决于设置RCRLDENn 或 FCRLDENn (RCRLDENn 和 FCRLDENn分别位于 EPWM\_CAPCTL[21:16]和 EPWM\_CAPCTL[29:24])。注：相应的GPIO管脚必须通过使能CAPINEn (EPWM\_CAPINEN[5:0])相应捕获通道n来配置为捕获功能。图 6.11-43是通道0的捕获方块图。

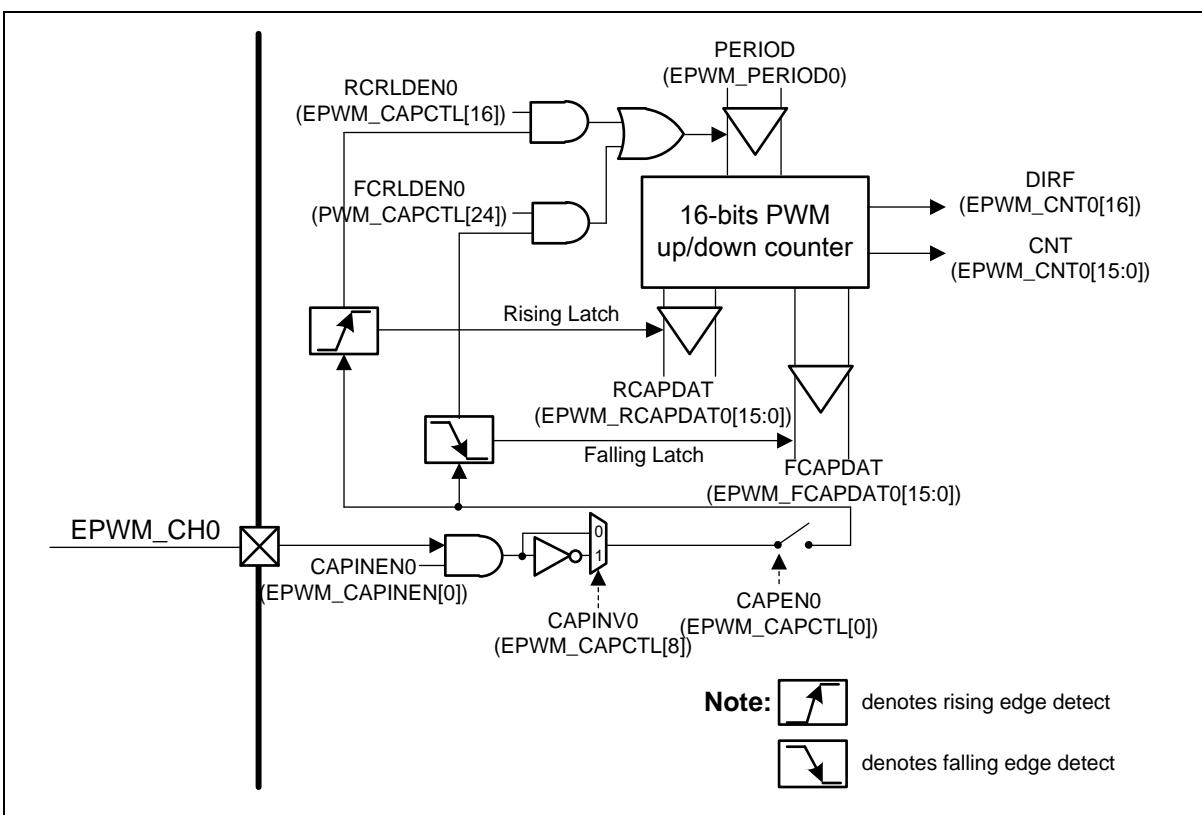


图 6.11-43 EPWM\_CH0 捕获框图

图 6.11-44为捕获功能时序。在这个例子中，捕获计数器被设置成EPWM下计数器类型，而且周期PERIOD被设置成8，所以这个计数器计数的方向是从上到下的，从8到0。当输入捕获管脚检测到一个下降沿时，捕获功能把计数器数值锁存到EPWM\_FCAPDATn中。当输入捕获管脚检测到一个上升沿，它锁存计数器数值到EPWM\_RCAPDATn中。在这个时序框图中，当一开始检测到下降沿时，捕获器重新加载计数器的数值，数值为周期值（PERIOD），原因是使能了FCRLDENn。但是在第二次下降沿时，计数器不会重新加载，这是因为关闭了FCRLDENn。在这个例子中，计数器在捕获到上升沿时也被重新加载了，原因是RCRLDENn也被使能了。

另外，这种情况如果是设为向上计数方式，计数器将重载零并向上计数到PERIOD值。

图 6.11-44也表示中断和中断标志产生的时序例子。当上升沿在通道n被检测到，相应位CRLIFn (EPWM\_CAPIF[5:0])将被硬件设置。同样，通道n检测到下降沿 相应位CFLIFn (EPWM\_CAPIF[13:8])被硬件设置。CRLIFn和CFLIFn可以通过软件写1清除。如果CRLIFn被设置并且CAPRIENn被使能，捕

获功能将产生一个中断。如果CFLIFn被设置并且CAPFIENn被使能，中断也会产生。

在本图没有描述的一个情况是：当CRLIF已经设定了，如果上升锁存再次发生了，运行状态寄存器CRLIFOVn (EPWM\_CAPSTS[5:0])将通过硬件被置起，来指示CAPRIF超载。同理，当下降锁存再次发生，对于中断标志CFLIF和超载状态CFLIFOVn (BEPWM\_CAPSTS[13:8])，相同的硬件操作也会发生。

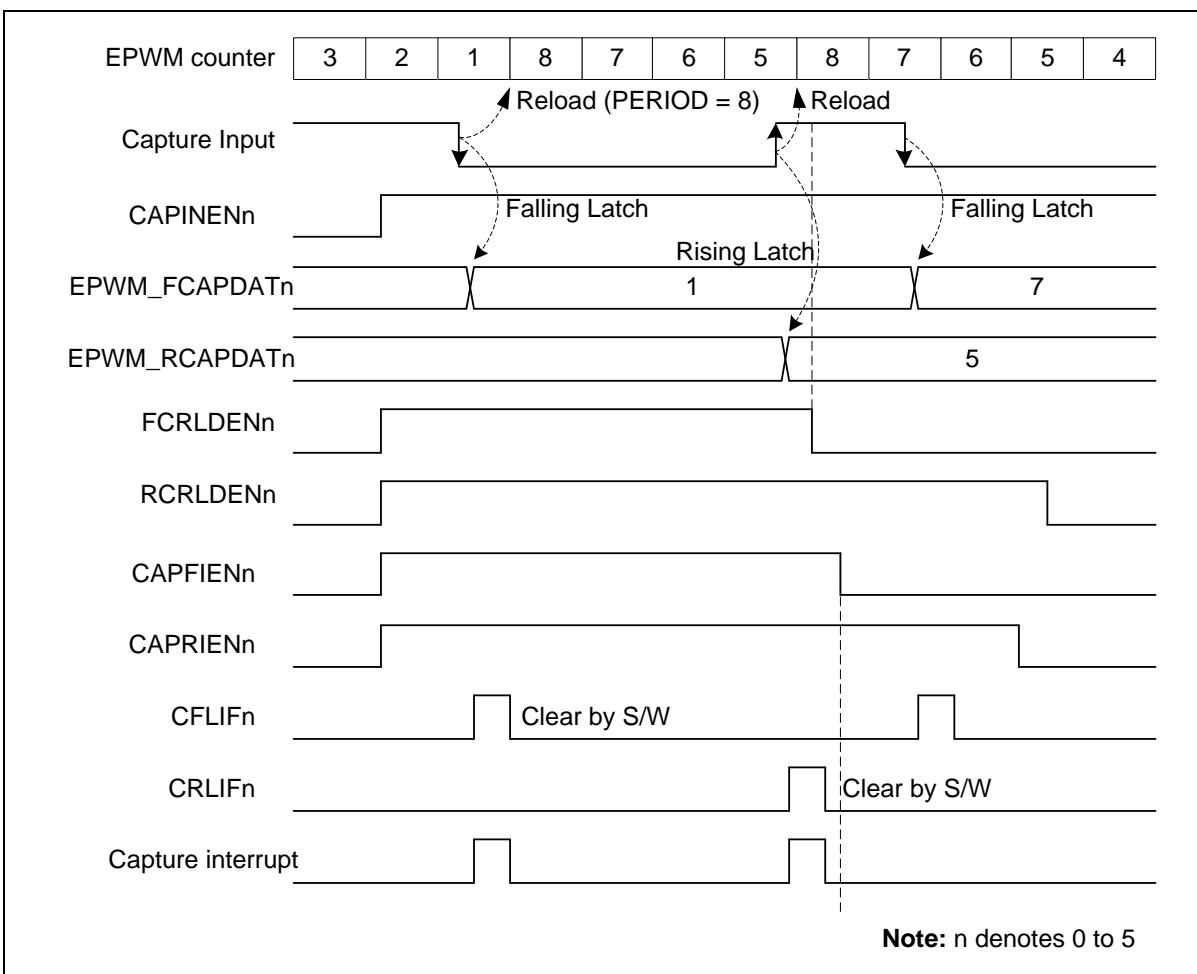


图 6.11-44 捕获操作波形图

捕获脉冲宽度计算，如下公式：

对于负脉冲的情况，通道低电平脉冲宽度为 $(EPWM\_PERIODn + 1 - EPWM\_RCAPDATn)$ 个EPWM计数器时间。这里一个WPM计数器时间是 $(CLKPSC+1) * EPWMx\_CLK$ 。在图 6.11-44 的情况，低脉冲的宽度是 $8+1-5 = 4$ 个EPWM计数器时间。

对于正脉冲的情况，通道高电平脉冲的宽度为 $(EPWM\_PERIODn + 1 - EPWM\_FCAPDATn)$ 个EPWM计数器时间。这里一个WPM计数器时间是 $(CLKPSC+1) * EPWMx\_CLK$ 。在图 6.11-44 的情况，高电平的宽度是 $8+1-7 = 2$ 个EPWM计数器时间。

#### 6.11.5.29 捕获 PDMA 功能

当EPWM工作在捕获模式，EPWM 模块支持PDMA 数据搬移功能。当相应的PDMA 使能位 CHENn\_m

(CHEN0\_1 在 EPWM\_PDMACTL[0], CHEN2\_3 在 EPWM\_PDMACTL[8] , CHEN4\_5 在 EPWM\_PDMACTL[16], n和m 代表互补通道)被使能, 当捕获事件发生后, 捕获模块将会产生一个到 PDMA 控制器的请求。PDMA模块读取了捕获模块CAPBUF (EPWM\_PDMACAPn\_m[15:0], n, m代表互补通道) 寄存器的值, 并搬移寄存器数据到内存后, PDMA 控制器将会给捕获模块产生一个应答。通过设置 CAPMODn\_m (CAPMOD0\_1 在 EPWM\_PDMACTL[2:1], CAPMOD2\_3 在 EPWM\_PDMACTL[10:9] , CAPMOD4\_5 在 EPWM\_PDMACTL[18:17]), PDMA可以搬移上升沿或下降沿或两个边沿捕获到的数据到内存。当使用PDMA 来搬移上升沿、下降沿的数据时, 不要忘记设置 CAPORDn\_m (CAPORD0\_1 在 EPWM\_PDMACTL[3], CAPORD2\_3 在 EPWM\_PDMACTL[11] , CAPORD4\_5 在 EPWM\_PDMACTL[19])寄存器来设定搬移数据的顺序(下降沿捕获数据在先还是上升沿捕获数据在先). 互补通道共用一个PDMA 通道。因此, 需要设置 CHSELn\_m (CHSEL0\_1 (EPWM\_PDMACTL[4]), CHSEL2\_3 (EPWM\_PDMACTL[12]) , CHSEL4\_5 (EPWM\_PDMACTL[20]))来确定是选择通道n还是通道m作为PDMA 通道。

图 6.11-45 是捕获PDMA的波形。在该例子当中, CHSEL0\_1 (EPWM\_PDMACTL[4]) 被设置为0。因此PDMA将会选择通道0 作为捕获数据搬移通道。CAPMOD0\_1 (EPWM\_PDMACTL[2:1]) 被设为3, 表明上升沿和下降沿捕获到的数据都会被搬移到内存。CAPORD0\_1 (EPWM\_PDMACTL[3]) = 1, 先搬移上升沿捕获到的数据, 然后再搬移下降沿数据。如图 6.11-45所示, CRLIFO 和 CFLIFO 信号的最后一部分有些重叠。EPWM\_RCAPDATA0 的值11 将被装载到EPWM\_PDMACAP0\_1来等待搬移, 但是EPWM\_FCAPDATA0 的值不会被搬移。EPWM\_PDMACAP0\_1 保存了将通过PDMA搬到内存的数据。图中HWDATA表示通过PDMA搬移的数据。

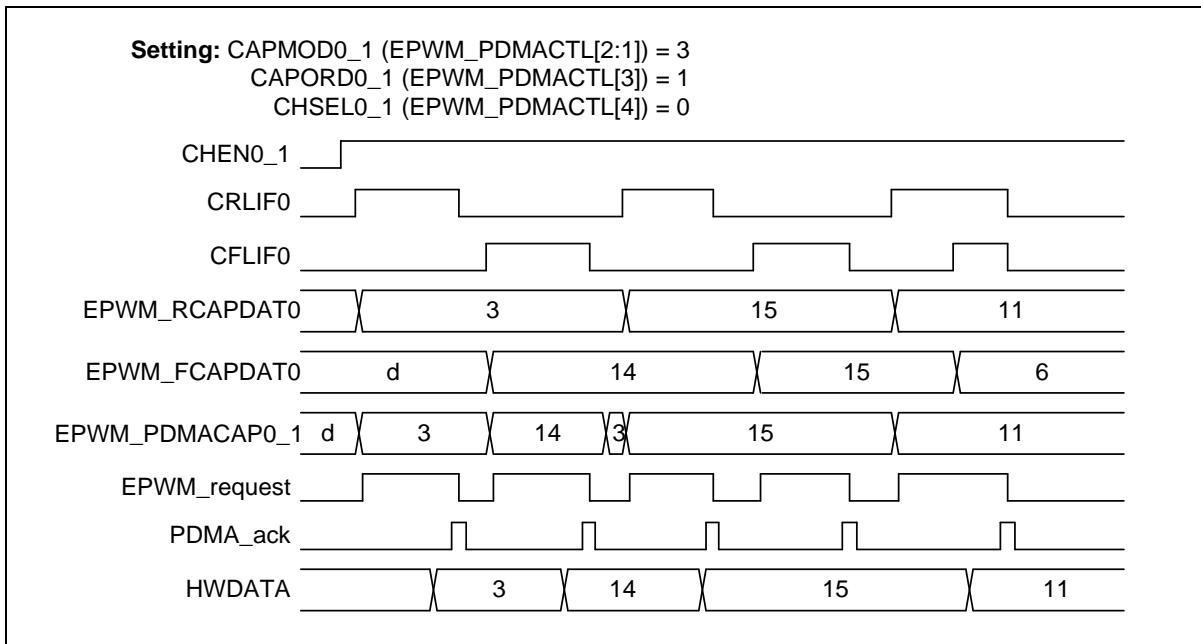


图 6.11-45 通道 0 捕获 PDMA 操作波形

### 6.11.5.30 累加器 PDMA 功能

EPWM 模块支持当累加器中断发生用PDMA传输功能。图 6.11-46 表示累加器 PDMA 功能架构。当对应的 PDMA 使能位 APDMAENn (EPWM\_APDMACTL[n], n=0~5) 置位, 累加器中断发生的时候也就是 IFAIFn (EPWM\_AINTSTS[n], n=0~5) 置 1, 累加器模块会发送一个请求到PDMA控制器。PDMA控制器在读内存数据且发送数据到指定寄存器(EPWM\_PERIOD, 等.)后会产生一个应答给累加器。所以用户可以使用这个功能来改变累加器中断频率。

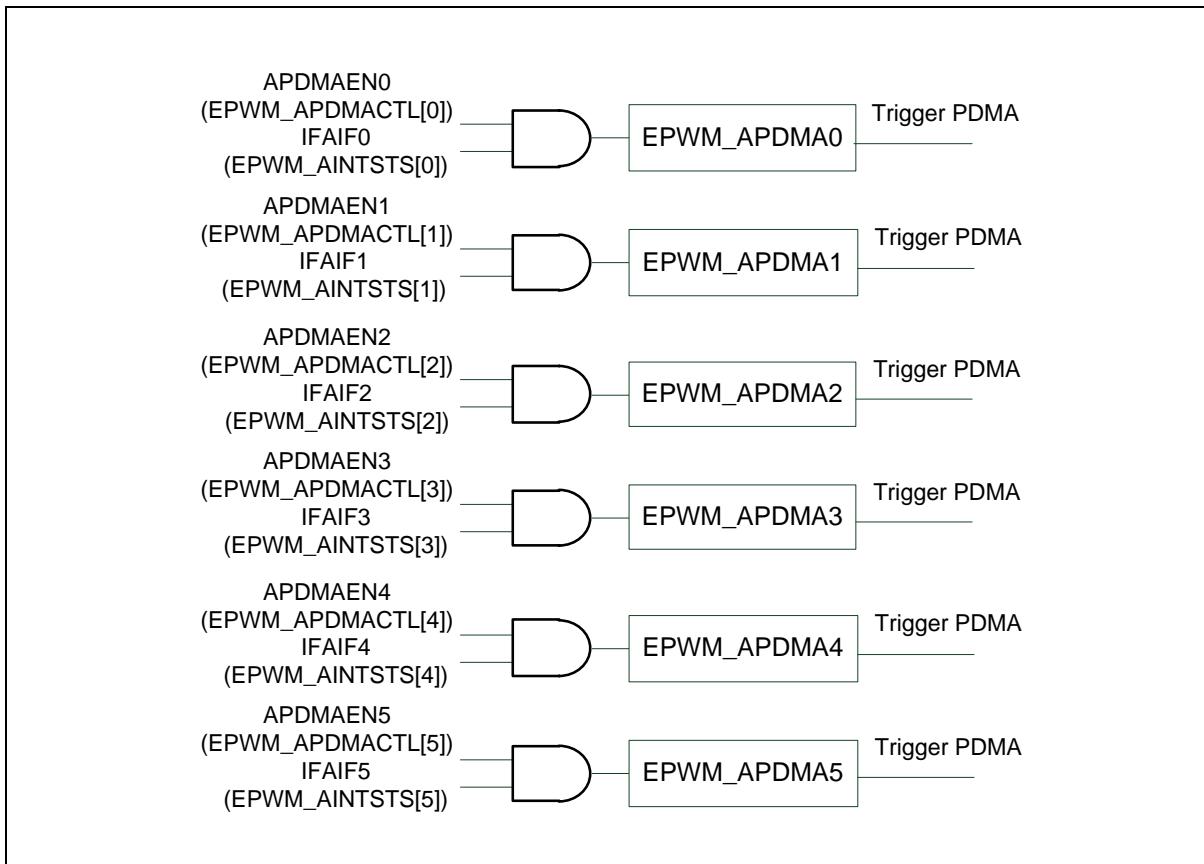


图 6.11-46 累加器 PDMA 功能架构

### 6.11.6 寄存器映射

R: 只读, W: 只写, R/W: 可读写

寄存器	偏移量	R/W	描述	复位值
<b>EPWM 基地址:</b>				
EPWM0_BA = 0x4005_8000				
EPWM1_BA = 0x4005_9000				
x=0,1				
EPWM_CTL0 x=0,1	EPWMx_BA+0x00	R/W	EPWM 控制寄存器 0	0x0000_0000
EPWM_CTL1 x=0,1	EPWMx_BA+0x04	R/W	EPWM 控制寄存器 1	0x0000_0000
EPWM_SYNC x=0,1	EPWMx_BA+0x08	R/W	EPWM同步寄存器	0x0000_0000
EPWM_SWSYNC x=0,1	EPWMx_BA+0x0C	R/W	EPWM 软件控制同步寄存器	0x0000_0000
EPWM_CLKSRC x=0,1	EPWMx_BA+0x10	R/W	EPWM 时钟源寄存器	0x0000_0000
EPWM_CLKPSC0_1 x=0,1	EPWMx_BA+0x14	R/W	EPWM时钟预分频寄存器0/1	0x0000_0000
EPWM_CLKPSC2_3 x=0,1	EPWMx_BA+0x18	R/W	EPWM时钟预分频寄存器2/3	0x0000_0000
EPWM_CLKPSC4_5 x=0,1	EPWMx_BA+0x1C	R/W	EPWM时钟预分频寄存器4/5	0x0000_0000
EPWM_CNTEN x=0,1	EPWMx_BA+0x20	R/W	EPWM 计数使能寄存器	0x0000_0000
EPWM_CNTCLR x=0,1	EPWMx_BA+0x24	R/W	EPWM清计数器寄存器	0x0000_0000
EPWM_LOAD x=0,1	EPWMx_BA+0x28	R/W	EPWM装载寄存器	0x0000_0000
EPWM_PERIOD0 x=0,1	EPWMx_BA+0x30	R/W	EPWM周期寄存器0	0x0000_0000
EPWM_PERIOD1 x=0,1	EPWMx_BA+0x34	R/W	EPWM周期寄存器1	0x0000_0000
EPWM_PERIOD2 x=0,1	EPWMx_BA+0x38	R/W	EPWM周期寄存器2	0x0000_0000
EPWM_PERIOD3 x=0,1	EPWMx_BA+0x3C	R/W	EPWM周期寄存器3	0x0000_0000
EPWM_PERIOD4 x=0,1	EPWMx_BA+0x40	R/W	EPWM周期寄存器4	0x0000_0000
EPWM_PERIOD5 x=0,1	EPWMx_BA+0x44	R/W	EPWM周期寄存器5	0x0000_0000

<b>EPWM_CMPDAT0 x=0,1</b>	EPWMx_BA+0x50	R/W	EPWM比较寄存器0	0x0000_0000
<b>EPWM_CMPDAT1 x=0,1</b>	EPWMx_BA+0x54	R/W	EPWM比较寄存器1	0x0000_0000
<b>EPWM_CMPDAT2 x=0,1</b>	EPWMx_BA+0x58	R/W	EPWM比较寄存器2	0x0000_0000
<b>EPWM_CMPDAT3 x=0,1</b>	EPWMx_BA+0x5C	R/W	EPWM比较寄存器3	0x0000_0000
<b>EPWM_CMPDAT4 x=0,1</b>	EPWMx_BA+0x60	R/W	EPWM比较寄存器4	0x0000_0000
<b>EPWM_CMPDAT5 x=0,1</b>	EPWMx_BA+0x64	R/W	EPWM比较寄存器5	0x0000_0000
<b>EPWM_DTCTL0_1 x=0,1</b>	EPWMx_BA+0x70	R/W	EPWM死区控制寄存器0/1	0x0000_0000
<b>EPWM_DTCTL2_3 x=0,1</b>	EPWMx_BA+0x74	R/W	EPWM死区控制寄存器2/3	0x0000_0000
<b>EPWM_DTCTL4_5 x=0,1</b>	EPWMx_BA+0x78	R/W	EPWM死区控制寄存器4/5	0x0000_0000
<b>EPWM_PHS0_1 x=0,1</b>	EPWMx_BA+0x80	R/W	EPWM计数器相位寄存器0/1	0x0000_0000
<b>EPWM_PHS2_3 x=0,1</b>	EPWMx_BA+0x84	R/W	EPWM计数器相位寄存器2/3	0x0000_0000
<b>EPWM_PHS4_5 x=0,1</b>	EPWMx_BA+0x88	R/W	EPWM计数器相位寄存器4/5	0x0000_0000
<b>EPWM_CNT0 x=0,1</b>	EPWMx_BA+0x90	R	EPWM计数寄存器0	0x0000_0000
<b>EPWM_CNT1 x=0,1</b>	EPWMx_BA+0x94	R	EPWM计数寄存器1	0x0000_0000
<b>EPWM_CNT2 x=0,1</b>	EPWMx_BA+0x98	R	EPWM计数寄存器2	0x0000_0000
<b>EPWM_CNT3 x=0,1</b>	EPWMx_BA+0x9C	R	EPWM计数寄存器3	0x0000_0000
<b>EPWM_CNT4 x=0,1</b>	EPWMx_BA+0xA0	R	EPWM计数寄存器4	0x0000_0000
<b>EPWM_CNT5 x=0,1</b>	EPWMx_BA+0xA4	R	EPWM计数寄存器5	0x0000_0000
<b>EPWM_WGCTL0 x=0,1</b>	EPWMx_BA+0xB0	R/W	EPWM发生寄存器0	0x0000_0000
<b>EPWM_WGCTL1 x=0,1</b>	EPWMx_BA+0xB4	R/W	EPWM发生寄存器1	0x0000_0000
<b>EPWM_MSKEN</b>	EPWMx_BA+0xB8	R/W	EPWM屏蔽使能寄存器	0x0000_0000

x=0,1				
<b>EPWM_MSK</b> x=0,1	EPWMx_BA+0xBC	R/W	EPWM屏蔽数据寄存器	0x0000_0000
<b>EPWM_BNF</b> x=0,1	EPWMx_BA+0xC0	R/W	EPWM 刹车噪音滤波寄存器	0x0000_0000
<b>EPWM_FAILBRK</b> x=0,1	EPWMx_BA+0xC4	R/W	EPWM 系统故障刹车控制寄存器	0x0000_0000
<b>EPWM_BRKCTL0_1</b> x=0,1	EPWMx_BA+0xC8	R/W	EPWM 刹车边沿检测控制寄存器0/1	0x0000_0000
<b>EPWM_BRKCTL2_3</b> x=0,1	EPWMx_BA+0xCC	R/W	EPWM 刹车边沿检测控制寄存器2/3	0x0000_0000
<b>EPWM_BRKCTL4_5</b> x=0,1	EPWMx_BA+0xD0	R/W	EPWM 刹车边沿检测控制寄存器4/5	0x0000_0000
<b>EPWM_POLCTL</b> x=0,1	EPWMx_BA+0xD4	R/W	EPWM管脚极性反转寄存器	0x0000_0000
<b>EPWM_POEN</b> x=0,1	EPWMx_BA+0xD8	R/W	EPWM输出使能寄存器	0x0000_0000
<b>EPWM_SWBRK</b> x=0,1	EPWMx_BA+0xDC	W	EPWM软件刹车控制寄存器	0x0000_0000
<b>EPWM_INTENO</b> x=0,1	EPWMx_BA+0xE0	R/W	EPWM 中断使能寄存器0	0x0000_0000
<b>EPWM_INTEN1</b> x=0,1	EPWMx_BA+0xE4	R/W	EPWM中断使能寄存器1	0x0000_0000
<b>EPWM_INTSTS0</b> x=0,1	EPWMx_BA+0xE8	R/W	EPWM中断标志寄存器0	0x0000_0000
<b>EPWM_INTSTS1</b> x=0,1	EPWMx_BA+0xEC	R/W	EPWM 中断标志寄存器1	0x0000_0000
<b>EPWM_DACTRGEN</b> x=0,1	EPWMx_BA+0xF4	R/W	EPWM 触发 DAC使能寄存器	0x0000_0000
<b>EPWM_EADCTS0</b> x=0,1	EPWMx_BA+0xF8	R/W	EPWM触发EADC源选择寄存器 0	0x0000_0000
<b>EPWM_EADCTS1</b> x=0,1	EPWMx_BA+0xFC	R/W	EPWM触发EADC源选择寄存器 1	0x0000_0000
<b>EPWM_FT CMPDAT0_1</b> x=0,1	EPWMx_BA+0x100	R/W	EPWM 自由触发比较寄存器0/1	0x0000_0000
<b>EPWM_FT CMPDAT2_3</b> x=0,1	EPWMx_BA+0x104	R/W	EPWM 自由触发比较寄存器2/3	0x0000_0000
<b>EPWM_FT CMPDAT4_5</b> x=0,1	EPWMx_BA+0x108	R/W	EPWM 自由触发比较寄存器4/5	0x0000_0000

<b>EPWM_SSCTL x=0,1</b>	EPWMx_BA+0x110	R/W	EPWM 同步开始控制寄存器	0x0000_0000
<b>EPWM_SSTRG x=0,1</b>	EPWMx_BA+0x114	W	EPWM同步开始触发寄存器	0x0000_0000
<b>EPWM_LEBCTL x=0,1</b>	EPWMx_BA+0x118	R/W	EPWM Leading Edge Blanking 控制寄存器	0x0000_0000
<b>EPWM_LEBCNT x=0,1</b>	EPWMx_BA+0x11C	R/W	EPWM Leading Edge Blanking 计数器寄存器	0x0000_0000
<b>EPWM_STATUS x=0,1</b>	EPWMx_BA+0x120	R/W	EPWM 状态寄存器	0x0000_0000
<b>EPWM_IFA0 x=0,1</b>	EPWMx_BA+0x130	R/W	EPWM 中断标志累加器寄存器 0	0x0000_0000
<b>EPWM_IFA1 x=0,1</b>	EPWMx_BA+0x134	R/W	EPWM中断标志累加器寄存器1	0x0000_0000
<b>EPWM_IFA2 x=0,1</b>	EPWMx_BA+0x138	R/W	EPWM中断标志累加器寄存器2	0x0000_0000
<b>EPWM_IFA3 x=0,1</b>	EPWMx_BA+0x13C	R/W	EPWM中断标志累加器寄存器3	0x0000_0000
<b>EPWM_IFA4 x=0,1</b>	EPWMx_BA+0x140	R/W	EPWM中断标志累加器寄存器4	0x0000_0000
<b>EPWM_IFA5 x=0,1</b>	EPWMx_BA+0x144	R/W	EPWM中断标志累加器寄存器5	0x0000_0000
<b>EPWM_AINTSTS x=0,1</b>	EPWMx_BA+0x150	R/W	EPWM累加器中断标志寄存器	0x0000_0000
<b>EPWM_AINTEN x=0,1</b>	EPWMx_BA+0x154	R/W	EPWM累加器中断使能寄存器	0x0000_0000
<b>EPWM_APDMACTL x=0,1</b>	EPWMx_BA+0x158	R/W	EPWM 累加器 PDMA 控制寄存器	0x0000_0000
<b>EPWM_CAPINEN x=0,1</b>	EPWMx_BA+0x200	R/W	EPWM 捕获输入使能寄存器	0x0000_0000
<b>EPWM_CAPCTL x=0,1</b>	EPWMx_BA+0x204	R/W	EPWM捕获控制寄存器	0x0000_0000
<b>EPWM_CAPSTS x=0,1</b>	EPWMx_BA+0x208	R	EPWM 捕获状态寄存器	0x0000_0000
<b>EPWM_RCAPDATA0 x=0,1</b>	EPWMx_BA+0x20C	R	EPWM 上升沿捕获数据寄存器 0	0x0000_0000
<b>EPWM_FCAPDATA0 x=0,1</b>	EPWMx_BA+0x210	R	EPWM 下降沿捕获数据寄存器 0	0x0000_0000
<b>EPWM_RCAPDATA1 x=0,1</b>	EPWMx_BA+0x214	R	EPWM上升沿捕获数据寄存器1	0x0000_0000
<b>EPWM_FCAPDATA1</b>	EPWMx_BA+0x218	R	EPWM下降沿捕获数据寄存器1	0x0000_0000

x=0,1				
EPWM_RCAPDAT2 x=0,1	EPWMx_BA+0x21C	R	EPWM 上升沿捕获数据寄存器 2	0x0000_0000
EPWM_FCAPDAT2 x=0,1	EPWMx_BA+0x220	R	EPWM 下降沿捕获数据寄存器2	0x0000_0000
EPWM_RCAPDAT3 x=0,1	EPWMx_BA+0x224	R	EPWM 上升沿捕获数据寄存器3	0x0000_0000
EPWM_FCAPDAT3 x=0,1	EPWMx_BA+0x228	R	EPWM 下降沿捕获数据寄存器3	0x0000_0000
EPWM_RCAPDAT4 x=0,1	EPWMx_BA+0x22C	R	EPWM 上升沿捕获数据寄存器4	0x0000_0000
EPWM_FCAPDAT4 x=0,1	EPWMx_BA+0x230	R	EPWM 下降沿捕获数据寄存器4	0x0000_0000
EPWM_RCAPDAT5 x=0,1	EPWMx_BA+0x234	R	EPWM上升沿捕获数据寄存器 5	0x0000_0000
EPWM_FCAPDAT5 x=0,1	EPWMx_BA+0x238	R	EPWM 下降沿捕获数据寄存器 5	0x0000_0000
EPWM_PDMACTL x=0,1	EPWMx_BA+0x23C	R/W	EPWM PDMA 控制寄存器	0x0000_0000
EPWM_PDMACAP0_1 x=0,1	EPWMx_BA+0x240	R	EPWM捕获通道01 PDMA 寄存器	0x0000_0000
EPWM_PDMACAP2_3 x=0,1	EPWMx_BA+0x244	R	EPWM捕获通道23 PDMA 寄存器	0x0000_0000
EPWM_PDMACAP4_5 x=0,1	EPWMx_BA+0x248	R	EPWM捕获通道45 PDMA 寄存器	0x0000_0000
EPWM_CAPIEN x=0,1	EPWMx_BA+0x250	R/W	EPWM捕获中断使能寄存器	0x0000_0000
EPWM_CAPIF x=0,1	EPWMx_BA+0x254	R/W	EPWM捕获中断标志寄存器	0x0000_0000
EPWM_PBUF0 x=0,1	EPWMx_BA+0x304	R	EPWM PERIOD0缓存	0x0000_0000
EPWM_PBUF1 x=0,1	EPWMx_BA+0x308	R	EPWM PERIOD1缓存	0x0000_0000
EPWM_PBUF2 x=0,1	EPWMx_BA+0x30C	R	EPWM PERIOD2缓存	0x0000_0000
EPWM_PBUF3 x=0,1	EPWMx_BA+0x310	R	EPWM PERIOD3缓存	0x0000_0000
EPWM_PBUF4 x=0,1	EPWMx_BA+0x314	R	EPWM PERIOD4缓存	0x0000_0000
EPWM_PBUF5 x=0,1	EPWMx_BA+0x318	R	EPWM PERIOD5缓存	0x0000_0000

<b>EPWM_CMPBUF0 x=0,1</b>	EPWMx_BA+0x31C	R	EPWM CMPDAT0缓存	0x0000_0000
<b>EPWM_CMPBUF1 x=0,1</b>	EPWMx_BA+0x320	R	EPWM CMPDAT1缓存	0x0000_0000
<b>EPWM_CMPBUF2 x=0,1</b>	EPWMx_BA+0x324	R	EPWM CMPDAT2缓存	0x0000_0000
<b>EPWM_CMPBUF3 x=0,1</b>	EPWMx_BA+0x328	R	EPWM CMPDAT3缓存	0x0000_0000
<b>EPWM_CMPBUF4 x=0,1</b>	EPWMx_BA+0x32C	R	EPWM CMPDAT4缓存	0x0000_0000
<b>EPWM_CMPBUF5 x=0,1</b>	EPWMx_BA+0x330	R	EPWM CMPDAT5缓存	0x0000_0000
<b>EPWM_CPSBUF0_1 x=0,1</b>	EPWMx_BA+0x334	R	EPWM CLKPSC0_1缓存	0x0000_0000
<b>EPWM_CPSBUF2_3 x=0,1</b>	EPWMx_BA+0x338	R	EPWM CLKPSC2_3缓存	0x0000_0000
<b>EPWM_CPSBUF4_5 x=0,1</b>	EPWMx_BA+0x33C	R	EPWM CLKPSC4_5缓存	0x0000_0000
<b>EPWM_FTCBUF0_1 x=0,1</b>	EPWMx_BA+0x340	R	EPWM FTCMPDAT0_1缓存	0x0000_0000
<b>EPWM_FTCBUF2_3 x=0,1</b>	EPWMx_BA+0x344	R	EPWM FTCMPDAT2_3缓存	0x0000_0000
<b>EPWM_FTCBUF4_5 x=0,1</b>	EPWMx_BA+0x348	R	EPWM FTCMPDAT4_5缓存	0x0000_0000
<b>EPWM_FTCI x=0,1</b>	EPWMx_BA+0x34C	R/W	EPWM FTCMPDAT 指示寄存器	0x0000_0000

### 6.11.7 寄存器描述

#### EPWM\_CTL0 EPWM 控制寄存器 0

寄存器	偏移量	R/W	描述	复位值
EPWM_CTL0	EPWMx_BA+0x00	R/W	EPWM 控制寄存器 0	0x0000_0000

31	30	29	28	27	26	25	24
DBGTRIOFF	DBGHALT	Reserved				GROUPEN	
23	22	21	20	19	18	17	16
Reserved		IMMLDEN5	IMMLDEN4	IMMLDEN3	IMMLDEN2	IMMLDEN1	IMMLDEN0
15	14	13	12	11	10	9	8
Reserved		WINLDEN5	WINLDEN4	WINLDEN3	WINLDEN2	WINLDEN1	WINLDEN0
7	6	5	4	3	2	1	0
Reserved		CTRLD5	CTRLD4	CTRLD3	CTRLD2	CTRLD1	CTRLD0

位	描述
[31]	<b>DBGTRIOFF</b>  ICE 调试模式禁止（写保护） 0= ICE 调试模式影响EPWM输出 ICE 调试模式期间EPWM管脚将强制作三态模式。 1= 禁止ICE 调试模式影响EPWM输出。 EPWM管脚将保持输出不管是否在ICE 调试模式 注意：该寄存器写保护，详情请参考SYS_REGLCTL 寄存器
[30]	<b>DBGHALT</b>  ICE 调试模式计数暂停（写保护） 如果 计数暂停被使能， EPWM所有计数器将保持当前值直到退出ICE调试模式。 0 =禁止ICE 调试模式计数暂停 1 = 使能ICE 调试模式计数暂停 注意：该寄存器写保护，详情请参考SYS_REGLCTL 寄存器
[29:26]	<b>Reserved</b> 保留.
[24]	<b>GROUPEN</b>  组功能使能位 0 = 每个EPWM通道输出波形相互独立 1 = EPWM_CH2 、 EPWM_CH4 与 EPWM_CH0 输出相同波形， EPWM_CH3 、 EPWM_CH5和EPWM_CH1输出相同波形
[23:22]	<b>Reserved</b> 保留.
[16+n] n=0,1..5	<b>IMMLDENn</b>  立即装载使能位 每位n控制相应EPWM通道n 0 = 每个周期结束时PERIOD将载入到PBUF。通过设定CTRLD位， CMPDAT将在每个周期的终点或者中心点载入到CMPBUF。 1= 当软件更新 PERIOD/CMPDAT， PERIOD/CMPDAT 将立即分别载入到 PBUF 和 CMPBUF。 注：如果IMMLDENn使能， WINLDENn 和 CTRLDn无效

[15:14]	<b>Reserved</b>	保留.
[8+n] n=0,1..5	<b>WINLDENn</b>	<b>窗口载入使能</b> 每位n控制相应EPWM通道n 0 = 在每个周期终点, PERIOD 将重载到 PBUF。通过设置CTRLD位, 每个周期终点或中心点CMPDAT将重载到CMPBUF。 1 = 在每个周期终点, PERIOD将重载到PBUF 。当有效重载窗口被设置以后, 在每个周期终点, CMPDAT将载入到 CMPBUF。有效重载窗口可通过软件写1到EPWM_LOAD 寄存器来设置, 装载成功后硬件自动清零。
[7:6]	<b>Reserved</b>	保留.
[n] n=0,1..5	<b>CTRLDn</b>	<b>中心重载</b> 每位n控制相应EPWM通道n 在上-下计数方式, 在每个周期终点PERIOD将载入到PBUF。. 每个周期中心点CMPDAT将载入到CMPBUF。

EPWM\_CTL1 EPWM 控制寄存器 1

寄存器	偏移量	R/W	描述	复位值
EPWM_CTL1	EPWMx_BA+0x04	R/W	EPWM 控制寄存器 1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved					OUTMODE4	OUTMODE2	OUTMODE0
23	22	21	20	19	18	17	16
Reserved		CNTMODE5	CNTMODE4	CNTMODE3	CNTMODE2	CNTMODE1	CNTMODE0
15	14	13	12	11	10	9	8
Reserved				CNTTYPE5	CNTTYPE4		
7	6	5	4	3	2	1	0
CNTTYPE3		CNTTYPE2			CNTTYPE1	CNTTYPE0	

位	描述	
[31:27]	Reserved	保留.
[24+n/2] n=0,2,4	OUTMODEn	<b>EPWM 输出模式</b> 每位n控制相应EPWM通道n 0=EPWM独立模式 1=EPWM互补模式 注：当操作在成组功能，这些位必须都设置为相同模式。
[23:22]	Reserved	保留.
[16+n] n=0,1..5	CNTMODEn	<b>EPWM 计数器模式</b> 每位n控制相应EPWM通道n. 0 = 自动重载模式. 1 = 单周期 (One-shot) 模式.
[15:12]	Reserved	保留.
[2n+1:2n] n=0,1..5	CNTTYPEn	<b>EPWM计数器方式类型</b> 每位n控制相应EPWM通道n. 00 = 向上计数类型(支持捕获模式). 01 = 向下计数模式 (支持捕获模式). 10 = 上下计数模式. 11 = 保留.

**EPWM SYNC EPWM同步寄存器**

寄存器	偏移量	R/W	描述	复位值
EPWM_SYNC	EPWMx_BA+0x08	R/W	EPWM 同步寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved				PHSDIR4	PHSDIR2	PHSDIR0	
23	22	21	20	19	18	17	16
SINPINV	SFLTCNT			SFLTCSEL			SNFLTEN
15	14	13	12	11	10	9	8
Reserved		SINSRC4		SINSRC2		SINSRC0	
7	6	5	4	3	2	1	0
Reserved					PHSEN4	PHSEN2	PHSEN0

位	描述	
[31:27]	Reserved	保留.
[24+n/2] n=0,2,4	PHSDIRn	<b>EPWM 相位方向控制</b> 每位n控制相应EPWM通道n. 0 = 控制 EPWM 计数器同步后递减计数. 1 =控制 EPWM 计数器同步后递增计数.
[23]	SINPINV	<b>同步输入管脚反向</b> 0 = 同步管脚的状态被传到负边沿侦测器 1 =同步管脚的反向状态被传到负边沿侦测器
[22:20]	SFLTCNT	<b>同步边沿侦测滤波计数</b> 该位控制边沿侦测的计数数目
[19:17]	SFLTCSEL	<b>同步边沿侦测滤波器时钟选择</b> 000 =滤波器时钟= HCLK. 001 =滤波器时钟= HCLK/2. 010 =滤波器时钟= HCLK/4. 011 =滤波器时钟= HCLK/8. 100 =滤波器时钟= HCLK/16. 101 =滤波器时钟= HCLK/32. 110 =滤波器时钟= HCLK/64. 111 =滤波器时钟= HCLK/128.
[16]	SNFLTEN	<b>EPWM0同步输入噪声滤波使能</b> 0 = EPWM0同步输入口噪声滤波禁止.. 1 = EPWM0同步输入口噪声滤波使能
[15:14]	Reserved	保留.
[9+n:8+n]	SINSRCn	<b>EPWM0同步输入源选择</b>

n=0,2,4		00 = 同步源来自同步输入或软件同步. 01 = 计数器等于0. 10 = 计数器等于 EPWM_CMPDATm, m 为 1, 3, 5. 11 = 无同步输出.
[7:3]	<b>Reserved</b>	保留.
[n/2] n=0,2,4	<b>PHSENn</b>	同步相位使能 0 = EPWM计数值禁止装载 PHS 值. 1 = EPWM 计数值使能装载 PHS 值.

EPWM\_SWSYNC EPWM软件控制同步寄存器

寄存器	偏移量	R/W	描述	复位值
EPWM_SWSYNC	EPWMx_BA+0x0C	R/W	EPWM 软件控制同步寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved					SWSYNC4	SWSYNC2	SWSYNC0

位	描述	
[31:3]	Reserved	保留.
[n/2] n=0,2,4	SWSYNCn	软件控制同步功能 每位n控制相应EPWM通道n 当 SINSRCn (EPWM_SYNC[13:8]) 设为 0, 同步输出源来自同步输入或此位

EPWM\_CLKSRC EPWM 时钟源寄存器

寄存器	偏移量	R/W	描述	复位值
EPWM_CLKSRC	EPWMx_BA+0x10	R/W	EPWM 时钟源寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved					ECLKSRC4		
15	14	13	12	11	10	9	8
Reserved					ECLKSRC2		
7	6	5	4	3	2	1	0
Reserved					ECLKSRC0		

位	描述	
[31:19]	<b>Reserved</b>	保留.
[18:16]	<b>ECLKSRC4</b>	<b>EPWM_CH45外部时钟源选择</b> 000 = EPWMx_CLK, x 表示0 或 1。 001 = TIMERO溢出。 010 = TIMER1溢出。 011 = TIMER2溢出。. 100 = TIMER3溢出。 其他 = 保留.
[15:11]	<b>Reserved</b>	保留.
[10:8]	<b>ECLKSRC2</b>	<b>EPWM_CH23外部时钟源选择</b> 000 = EPWMx_CLK, x表示0 或 1。 001 = TIMERO溢出。 010 = TIMER1溢出。 011 = TIMER2溢出。 100 = TIMER3溢出。. 其他 =保留.
[7:3]	<b>Reserved</b>	保留.
[2:0]	<b>ECLKSRC0</b>	<b>EPWM_CH01外部时钟源选择</b> 000 = EPWMx_CLK, x表示0 或 1。 001 = TIMERO溢出。. 010 = TIMER1溢出。 011 = TIMER2溢出。 100 = TIMER3溢出。 其他 =保留..

EPWM\_CLKPSC0 1, 2 3, 4 5 EPWM 时钟预分频寄存器0 1, 2 3, 4 5

寄存器	偏移量	R/W	描述	复位值
EPWM_CLKPSC0_1	EPWMx_BA+0x14	R/W	EPWM 时钟预分频寄存器0/1	0x0000_0000
EPWM_CLKPSC2_3	EPWMx_BA+0x18	R/W	EPWM 时钟预分频寄存器2/3	0x0000_0000
EPWM_CLKPSC4_5	EPWMx_BA+0x1C	R/W	EPWM 时钟预分频寄存器4/5	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved				CLKPSC			
7	6	5	4	3	2	1	0
CLKPSC							

位	描述	
[31:12]	Reserved	保留.
[11:0]	CLKPSC	<b>EPWM</b> 计数器时钟预分频 EPWM计数器时钟由时钟预分频器决定。每个EPWM组共享一个EPWM寄存器时钟预分频器。EPWM计数器时钟源被(CLKPSC+1)除频。

EPWM\_CNTEN EPWM计数使能寄存器

寄存器	偏移量	R/W	描述	复位值
EPWM_CNTEN	EPWMx_BA+0x20	R/W	EPWM 计数使能寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved		CNTEN5	CNTEN4	CNTEN3	CNTEN2	CNTEN1	CNTEN0

位	描述	
[31:6]	Reserved	保留.
[n] n=0,1..5	CNTENn	<b>EPWM计数使能位</b> 每位n控制相应EPWM通道n 0 = EPWM计数器和时钟分频器停止工作 1 = EPWM计数器和时钟分频器开始工作

EPWM\_CNTCLR EPWM 清计数器寄存器

寄存器	偏移量	R/W	描述	复位值
EPWM_CNTCLR	EPWMx_BA+0x24	R/W	EPWM 清计数器寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved		CNTCLR5	CNTCLR4	CNTCLR3	CNTCLR2	CNTCLR1	CNTCLR0

位	描述	
[31:6]	<b>Reserved</b>	保留.
[n] n=0,1..5	<b>CNTCLRn</b>	<p><b>清 EPWM计数器控制位</b></p> <p>此位由硬件自动清零。每位n控制相应EPWM通道n。</p> <p>0 = 无影响。 1 =清16位EPWM计数器到0000H</p>

EPWM LOAD EPWM装载寄存器

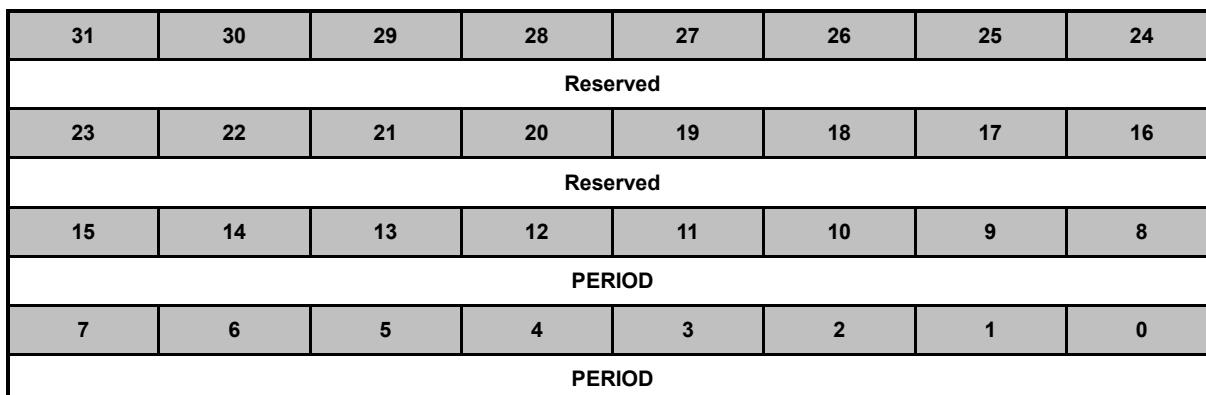
寄存器	偏移量	R/W	描述	复位值
EPWM_LOAD	EPWMx_BA+0x28	R/W	EPWM 装载寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved		LOAD5	LOAD4	LOAD3	LOAD2	LOAD1	LOAD0

位	描述	
[31:6]	Reserved	保留.
[n] n=0,1..5	LOADn	<p><b>重载EPWM 比较值寄存器 (CMPDAT) 控制位</b></p> <p>该位为软件置位，当当前EPWM周期结束时硬件清零。每位n控制相应EPWM通道n。</p> <p>写操作:</p> <p>0 = 无影响 1 = 设置窗口装载模式的装载窗口.</p> <p>读操作</p> <p>0 = 没有装载窗口设置 1 = 装载窗口被设置.</p> <p><b>注意:</b>该位只在窗口装载模式 WINLDENn(EPWM_CTL0[13:8]) = 1下使用</p>

EPWM\_PERIOD0~5 EPWM周期寄存器0~5

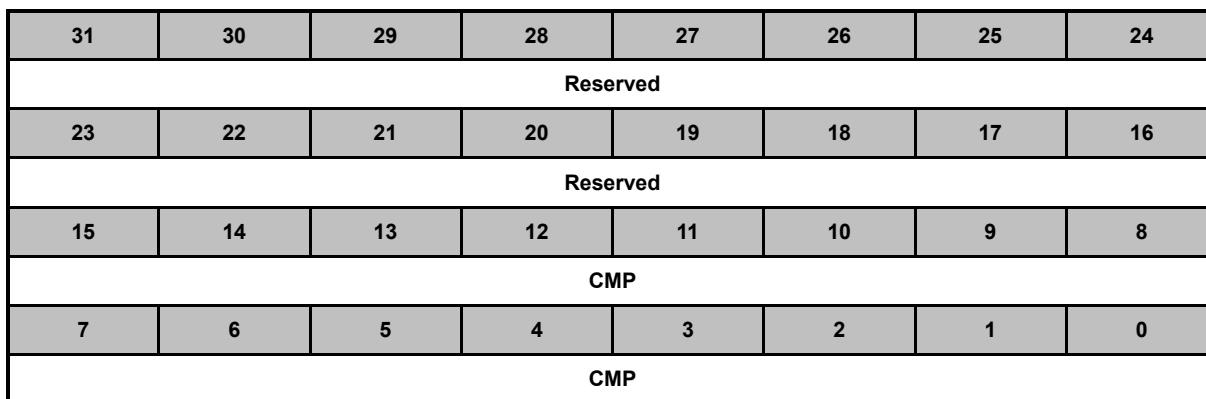
寄存器	偏移量	R/W	描述	复位值
EPWM_PERI_0D0	EPWMx_BA+0x30	R/W	EPWM 周期寄存器0	0x0000_0000
EPWM_PERI_0D1	EPWMx_BA+0x34	R/W	EPWM 周期寄存器1	0x0000_0000
EPWM_PERI_0D2	EPWMx_BA+0x38	R/W	EPWM 周期寄存器2	0x0000_0000
EPWM_PERI_0D3	EPWMx_BA+0x3C	R/W	EPWM 周期寄存器 3	0x0000_0000
EPWM_PERI_0D4	EPWMx_BA+0x40	R/W	EPWM 周期寄存器 4	0x0000_0000
EPWM_PERI_0D5	EPWMx_BA+0x44	R/W	EPWM 周期寄存器5	0x0000_0000



位	描述	
[31:16]	Reserved	保留.
[15:0]	PERIOD	<p><b>EPWM 周期寄存器</b></p> <p>向上计数模式，此模式下，EPWM计数器从0计数到PERIOD，并从0重新开始计数。</p> <p>向下计数模式：此模式下，EPWM计数器从PERIOD计数到0，并从PERIOD重新开始计数。</p> <p>EPWM周期时间=(PERIOD+1) * EPWM_CLK 周期.</p> <p>上下计数模式：这个模式EPWM计数器从0计数到PERIOD然后递减到0，并重复。</p> <p>EPWM周期时间=2 * PERIOD * EPWM_CLK周期.</p>

EPWM\_CMPDAT0~5 EPWM比较寄存器0~5

寄存器	偏移量	R/W	描述	复位值
EPWM_CMPDAT0	EPWMx_BA+0x50	R/W	EPWM比较寄存器0	0x0000_0000
EPWM_CMPDAT1	EPWMx_BA+0x54	R/W	EPWM比较寄存器1	0x0000_0000
EPWM_CMPDAT2	EPWMx_BA+0x58	R/W	EPWM比较寄存器2	0x0000_0000
EPWM_CMPDAT3	EPWMx_BA+0x5C	R/W	EPWM比较寄存器3	0x0000_0000
EPWM_CMPDAT4	EPWMx_BA+0x60	R/W	EPWM比较寄存器4	0x0000_0000
EPWM_CMPDAT5	EPWMx_BA+0x64	R/W	EPWM比较寄存器5	0x0000_0000



位	描述	
[31:16]	Reserved	保留.
[15:0]	CMP	<p><b>EPWM 比较寄存器</b></p> <p>CMP用于与CNTR比较来产生EPWM波形，中断和触发EADC/DAC。</p> <p>独立模式，CMPDAT0~5作为6个独立EPWM_CH0~5比较点。</p> <p>互补模式， CMPDAT0, 2, 4作为第一比较点而CMPDAT1, 3, 5作为第二比较点对应于相应的三个互补组EPWM_CH0 和 EPWM_CH1, EPWM_CH2 和 EPWM_CH3, EPWM_CH4 和 EPWM_CH5。</p>

EPWM\_DTCTL0\_1,2,3,4,5 EPWM死区控制寄存器0\_1,2,3,4,5

寄存器	偏移量	R/W	描述	复位值
EPWM_DTCTL0_1	EPWMx_BA+0x70	R/W	EPWM死区控制寄存器0/1	0x0000_0000
EPWM_DTCTL2_3	EPWMx_BA+0x74	R/W	EPWM死区控制寄存器2/3	0x0000_0000
EPWM_DTCTL4_5	EPWMx_BA+0x78	R/W	EPWM死区控制寄存器4/5	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							DTCKSEL
23	22	21	20	19	18	17	16
Reserved							DTEN
15	14	13	12	11	10	9	8
Reserved				DTCNT			
7	6	5	4	3	2	1	0
DTCNT							

位	描述	
[31:25]	Reserved	保留.
[24]	DTCKSEL	<p>死区时钟源选择 (写保护)            0 = 死区时钟源来自 EPWM_CLK.            1 = 死区时钟源来自预分频器输出.  <b>注意:</b> 该位为写保护位. 详情请参考SYS_REGLCTL 寄存器</p>
[23:17]	Reserved	保留.
[16]	DTEN	<p>使能 EPWM 组 (EPWM_CH0, EPWM_CH1) (EPWM_CH2, EPWM_CH3) (EPWM_CH4, EPWM_CH5) 死区插入(写保护)            死区插入只有当该组互补EPWM使能才激活。如果死区插入未激活，该互补组输出管脚没有任何延时。            0 = 禁止管脚组死区插入。            1 = 使能管脚组死区插入.  <b>注意:</b> 该位为写保护位. 详情请参考SYS_REGLCTL 寄存器</p>
[15:12]	Reserved	保留.
[11:0]	DTCNT	<p>死区计数器 (写保护)            死区时间可以根据以下公式计算:            死区时间=(DTCNT[11:0]+1) * EPWM_CLK 周期            注: 该寄存器写保护, 参考REGWRPROT寄存器  <b>注意:</b> 该位为写保护位. 详情请参考SYS_REGLCTL 寄存器</p>

EPWM\_PHS0\_1, 2, 3, 4, 5 EPWM计数器相位寄存器0\_1, 2, 3, 4, 5

寄存器	偏移量	R/W	描述	复位值
EPWM_PHS0_1	EPWMx_BA+0x80	R/W	EPWM 计数器相位寄存器0/1	0x0000_0000
EPWM_PHS2_3	EPWMx_BA+0x84	R/W	EPWM 计数器相位寄存器2/3	0x0000_0000
EPWM_PHS4_5	EPWMx_BA+0x88	R/W	EPWM 计数器相位寄存器4/5	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
PHS							
7	6	5	4	3	2	1	0
PHS							

位	描述	
[31:16]	Reserved	保留.
[15:0]	PHS	EPWM 同步开始相位 PHS决定EPWM 同步开始的相位值. 该位用于同步功能.

EPWM\_CNT0~5 EPWM 计数寄存器0~5

寄存器	偏移量	R/W	描述	复位值
EPWM_CNT0	EPWMx_BA+0x90	R	EPWM计数寄存器0	0x0000_0000
EPWM_CNT1	EPWMx_BA+0x94	R	EPWM计数寄存器1	0x0000_0000
EPWM_CNT2	EPWMx_BA+0x98	R	EPWM计数寄存器2	0x0000_0000
EPWM_CNT3	EPWMx_BA+0x9C	R	EPWM计数寄存器3	0x0000_0000
EPWM_CNT4	EPWMx_BA+0xA0	R	EPWM计数寄存器4	0x0000_0000
EPWM_CNT5	EPWMx_BA+0xA4	R	EPWM计数寄存器5	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
CNT							
7	6	5	4	3	2	1	0
CNT							

位	描述	
[31:17]	Reserved	保留.
[16]	DIRF	<b>EPWM方向标志(只读)</b> 0 =向下计数 1 =向上计数
[15:0]	CNT	<b>EPWM数据寄存器 (只读)</b> 用户可以读CNT以知道16位周期计数器当前值。

EPWM WGCTL0 EPWM产生寄存器 0

寄存器	偏移量	R/W	描述	复位值
EPWM_WGCTL0	EPWMx_BA+0xB0	R/W	EPWM 产生寄存器 0	0x0000_0000

31	30	29	28	27	26	25	24
Reserved				PRDPCTL5		PRDPCTL4	
23	22	21	20	19	18	17	16
PRDPCTL3		PRDPCTL2		PRDPCTL1		PRDPCTL0	
15	14	13	12	11	10	9	8
Reserved				ZPCTL5		ZPCTL4	
7	6	5	4	3	2	1	0
ZPCTL3		ZPCTL2		ZPCTL1		ZPCTL0	

位	描述	
[31:28]	<b>Reserved</b>	保留.
[17+2n:16+2n] n=0,1..5	<b>PRDPCTL<sub>n</sub></b>	<p><b>EPWM 周期（中心）点控制</b></p> <p>00 =不变 01 = EPWM周期（中心）点输出低 10 = EPWM周期（中心）点输出高 11 = EPWM周期（中心）点输出翻转</p> <p>当EPWM计数器计数到(PERIODn+1)，可以控制EPWM输出电平</p> <p>注意：当EPWM寄存器工作在上下计数方式，该位是中心点控制。</p>
[15:12]	<b>Reserved</b>	保留.
[1+2n:2n] n=0,1..5	<b>ZPCTL<sub>n</sub></b>	<p><b>EPWM零点控制</b></p> <p>00 =不变 01 = EPWM零点输出低 10 = EPWM零点输出高 11 = EPWM零点输出翻转</p> <p>当EPWM计数器计数到零点可以控制EPWM输出电平.</p>

EPWM\_WGCTL1 EPWM 产生寄存器 1

寄存器	偏移量	R/W	描述	复位值
EPWM_WGCTL1	EPWMx_BA+0xB4	R/W	EPWM 产生寄存器 1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved				CMPDCTL5		CMPDCTL4	
23	22	21	20	19	18	17	16
CMPDCTL3		CMPDCTL2		CMPDCTL1		CMPDCTL0	
15	14	13	12	11	10	9	8
Reserved				CMPUCTL5		CMPUCTL4	
7	6	5	4	3	2	1	0
CMPUCTL3		CMPUCTL2		CMPUCTL1		CMPUCTL0	

位	描述	
[31:28]	<b>Reserved</b>	保留.
[17+2n:16+2n] n=0..5	<b>CMPDCTL<sub>n</sub></b>	<p><b>EPWM向下比较点控制</b></p> <p>00 =不变 01 = EPWM向下比较点输出低 10 = EPWM向下比较点输出高 11 =向下比较点输出翻转</p> <p>当寄存器向下计数到CMPDAT, 可控制EPWM输出电平</p> <p>注: 在互补模式, CMPDCTL 1,3,5作为通道0, 2, 4.的另一个CMPDCTL。</p>
[15:12]	<b>Reserved</b>	保留.
[1+2n:2n] n=0..5	<b>CMPUCTL<sub>n</sub></b>	<p><b>EPWM向上比较点控制</b></p> <p>每位n控制相应EPWM通道n</p> <p>00 =不变 01 = EPWM 向上比较点输出低 10 = EPWM 向上比较点输出高 11 = EPWM向上比较点输出翻转</p> <p>当寄存器向上计数到CMPDAT, EPWM可控制输出电平</p> <p>注: 在互补模式, CMPUCTL11,3,5作为通道0, 2, 4.的另一个CMPUCTL。</p>

EPWM MSKEN EPWM屏蔽使能寄存器

寄存器	偏移量	R/W	描述	复位值
EPWM_MSKE_N	EPWMx_BA+0xB8	R/W	EPWM 屏蔽使能寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved		MSKEN5	MSKEN4	MSKEN3	MSKEN2	MSKEN1	MSKEN0

位	描述	
[31:6]	<b>Reserved</b>	保留.
[n] n=0,1..5	<b>MSKENn</b>	<p><b>EPWM 屏蔽使能寄存器</b>            每位n控制相应EPWM通道n</p> <p>当该位使能， EPWM输出信号将被屏蔽。 相应的EPWM通道n将输出MSKDATn (EPWM_MSK[5:0])数据</p> <p>0 = 输出信号不屏蔽            1 = EPWM输出信号被屏蔽并输出MSKDATn数据.</p>

EPWM MSK EPWM 屏蔽数据寄存器

寄存器	偏移量	R/W	描述	复位值
EPWM_MSK	EPWMx_BA+0xBC	R/W	EPWM 屏蔽数据寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved		MSKDAT5	MSKDAT4	MSKDAT3	MSKDAT2	MSKDAT1	MSKDAT0

位	描述	
[31:6]	Reserved	保留.
[n] n=0,1..5	MSKDATn	<b>EPWM 屏蔽数据位</b> 如果相应屏蔽功能使能，该数据位控制EPWMn输出管脚状态。每位n控制相应EPWM通道n 0 = 输出逻辑低到EPWM通道n 1 = 输出逻辑高到EPWM通道n

EPWM\_BNF EPWM 刹车噪音滤波寄存器

寄存器	偏移量	R/W	描述	复位值
EPWM_BNF	EPWMx_BA+0xC0	R/W	EPWM 刹车噪音滤波寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							BK1SRC
23	22	21	20	19	18	17	16
Reserved							BK0SRC
15	14	13	12	11	10	9	8
BRK1PINV	BRK1FCNT			BRK1NFSEL			BRK1NFEN
7	6	5	4	3	2	1	0
BRK0PINV	BRK0FCNT			BRK0NFSEL			BRK0NFEN

位	描述	
[31:25]	Reserved	保留.
[24]	BK1SRC	<p>刹车 1 管脚源选择</p> <p>对于 EPWM0 的设置:</p> <p>0 = 刹车 1 管脚源来自 EPWM0_BRAKE1. 1 = 刹车 1 管脚源来自 EPWM1_BRAKE1.</p> <p>对于 EPWM1 的设置:</p> <p>0 = 刹车 1 管脚源来自 EPWM1_BRAKE1. 1 = 刹车 1 管脚源来自 EPWM0_BRAKE1.</p>
[23:17]	Reserved	保留.
[16]	BK0SRC	<p>刹车 0 管脚源选择</p> <p>对于 EPWM0 的设置:</p> <p>0 = 刹车 0 管脚源来自 EPWM0_BRAKE0. 1 = 刹车 0 管脚源来自 EPWM1_BRAKE0.</p> <p>对于 EPWM1 的设置:</p> <p>0 = 刹车 0 管脚源来自 EPWM1_BRAKE0. 1 = 刹车 0 管脚源来自 EPWM0_BRAKE0.</p>
[15]	BRK1PINV	<p>刹车1管脚翻转</p> <p>0 = EPWMx_BRAKE1管脚状态被传到负边沿检测器。 1 = EPWMx_BRAKE1管脚反向状态被传到负边沿检测器。</p>
[14:12]	BRK1FCNT	刹车1边沿检测滤波器计数 该位控制刹车1滤波计数器从0到BRK1FCNT计数
[11:9]	BRK1NFSEL	刹车1边沿检测滤波器时钟选择 000 = 滤波器时钟 = HCLK. 001 = 滤波器时钟 HCLK/2.

		010 =滤波器时钟= HCLK/4. 011 =滤波器时钟= HCLK/8. 100 =滤波器时钟= HCLK/16. 101 =滤波器时钟= HCLK/32. 110 =滤波器时钟= HCLK/64. 111 =滤波器时钟= HCLK/128.
[8]	<b>BRK1NFEN</b>	刹车1噪音滤波器使能 0 = 禁止EPWM刹车1噪音滤波器 1 = 使能EPWM刹车1噪音滤波器.
[7]	<b>BRK0PINV</b>	刹车0管脚翻转 0 = EPWMx_BRAKE0管脚状态被传到负边沿检测器。 1 = EPWMx_BRAKE0管脚反向状态被传到负边沿检测器。
[6:4]	<b>BRK0FCNT</b>	刹车0边沿检测滤波器计数 寄存器位控制刹车0滤波计数器从0到BRK0FCNT计数
[3:1]	<b>BRK0NFSEL</b>	刹车0边沿检测滤波器时钟选择 000 =滤波器时钟= HCLK. 001 =滤波器时钟= HCLK/2. 010 =滤波器时钟= HCLK/4. 011 =滤波器时钟= HCLK/8. 100 =滤波器时钟= HCLK/16. 101 =滤波器时钟= HCLK/32. 110 =滤波器时钟= HCLK/64. 111 =滤波器时钟= HCLK/128.
[0]	<b>BRK0NFEN</b>	EPWM刹车0噪音滤波器使能 0 = 禁止EPWM刹车0噪音滤波器 1 = 使能EPWM刹车0噪音滤波器

EPWM FAILBRK EPWM系统故障刹车控制寄存器

寄存器	偏移量	R/W	描述	复位值
EPWM_FAILBRK	EPWMx_BA+0xC4	R/W	EPWM 系统故障刹车控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved				CORBRKEN	RAMBRKEN	BODBRKEN	CSSBRKEN

位	描述	
[31:4]	Reserved	保留.
[3]	CORBRKEN	内核锁死检测触发EPWM刹车功能0使能 0 = 禁止通过内核锁死检测触发刹车功能 1 = 使能通过内核锁死检测触发刹车功能
[2]	RAMBRKEN	SRAM 奇偶校验错误检查触发EPWM 刹车功能0 使能 0 = SRAM奇偶校验错误检查触发刹车功能禁止. 1 = SRAM奇偶校验错误检查触发刹车功能使能.
[1]	BODBRKEN	欠压检测触发EPWM刹车功能0使能 0 = 禁止BOD触发刹车功能 1 = 使能BOD触发刹车功能
[0]	CSSBRKEN	时钟安全系统检测触发EPWM刹车功能0使能 0 = 禁止通过CSS检测触发刹车功能 1 = 使能通过CSS检测触发刹车功能

EPWM\_BRKCTL0\_1, 2, 3, 4, 5 EPWM刹车边沿检测控制寄存器0\_1, 2, 3, 4, 5

寄存器	偏移量	R/W	描述	复位值
EPWM_BRKCTL_0_1	EPWMx_BA+0xC8	R/W	EPWM刹车边沿检测控制寄存器0/1	0x0000_0000
EPWM_BRKCTL_2_3	EPWMx_BA+0xCC	R/W	EPWM刹车边沿检测控制寄存器2/3	0x0000_0000
EPWM_BRKCTL_4_5	EPWMx_BA+0xD0	R/W	EPWM刹车边沿检测控制寄存器4/5	0x0000_0000

31	30	29	28	27	26	25	24
Reserved			EADCLBEN	Reserved			
23	22	21	20	19	18	17	16
Reserved			EADCEBEN	BRKAODD		BRKAEVEN	
15	14	13	12	11	10	9	8
SYSLBEN	Reserved	BRKP1LEN	BRKP0LEN	Reserved		CPO1LBEN	CPO0LBEN
7	6	5	4	3	2	1	0
SYSEBEN	Reserved	BRKP1EEN	BRKP0EEN	Reserved		CPO1EBEN	CPO0EBEN

位	描述
[31:29]	<b>Reserved</b> 保留.
[28]	<b>EADCLBEN</b> 使能 EADC 结果监视器 (EADCRM) 作为电平侦测刹车源 (写保护) 0 = EADCRM 作为电平侦测刹车源禁止 1 = EADCRM 作为电平侦测刹车源使能. 注：该寄存器写保护，参考SYS_REGLCTL寄存器
[27:21]	<b>Reserved</b> 保留.
[20]	<b>EADCEBEN</b> 使能 EADC 结果监视器 (EADCRM) 作为边沿侦测刹车源 (写保护) 0 = EADCRM 作为边沿侦测刹车源禁止 1 = EADCRM 作为边沿侦测刹车源使能. 注：该寄存器写保护，参考SYS_REGLCTL寄存器
[19:18]	<b>BRKAODD</b> EPWM奇数通道刹车行为选择 (写保护) 00 = EPWMx的刹车事件不影响奇数通道输出。 01 = 当EPWMx的刹车事件发生，EPWM奇数通道输出三态。 10 = 当EPWMx的刹车事件发生，EPWM奇数通道输出低电平。 11 = 当EPWMx的刹车事件发生，EPWM奇数通道输出高电平。 注：该寄存器写保护，参考SYS_REGLCTL寄存器.
[17:16]	<b>BRKAEVEN</b> EPWM偶数通道刹车行为选择 (写保护) 00 = EPWMx的刹车事件不影响偶数通道输出。 01 = 当EPWMx的刹车事件发生，EPWM偶数通道输出三态。 10 = 当EPWMx的刹车事件发生，EPWM偶数通道输出低电平。

		11 =当EPWMx的刹车事件发生， EPWM偶数通道输出高电平。 注：该寄存器写保护，参考SYS_REGLCTL寄存器。
[15]	<b>SYSLBEN</b>	使能系统故障作为电平检测刹车源（写保护） 0 =禁止系统故障条件作为电平检测刹车源 1 =使能系统故障条件作为电平检测刹车源 注：该寄存器写保护，参考SYS_REGLCTL寄存器
[14]	<b>Reserved</b>	保留.
[13]	<b>BRKP1LEN</b>	使能管脚BKP1作为电平检测刹车源（写保护） 0 =.禁止EPWMx_BRAKE1管脚作为电平检测刹车源 1 =使能EPWMx_BRAKE1管脚作为电平检测刹车源 注：该寄存器写保护，参考SYS_REGLCTL寄存器
[12]	<b>BRKPOLEN</b>	使能管脚BKP0作为电平检测刹车源（写保护） 0 = 禁止EPWMx_BRAKE0管脚作为电平检测刹车源 1 = 使能EPWMx_BRAKE0管脚作为电平检测刹车源 注意：该寄存器写保护，参考SYS_REGLCTL寄存器
[11:10]	<b>Reserved</b>	保留.
[9]	<b>CPO1LBEN</b>	使能 ACMP1_O 数字输出作为电平检测刹车源（写保护） 0 = ACMP1_O 作为电平检测刹车源禁止 1 = ACMP1_O 作为电平检测刹车源使能 注意：该寄存器写保护，参考SYS_REGLCTL寄存器
[8]	<b>CPO0LBEN</b>	使能ACMP0_O数字输出作为电平检测刹车源（写保护） 0 = ACMP0_O作为电平检测刹车源禁止. 1 = ACMP0_O作为电平检测刹车源使能 注意：该寄存器写保护，参考SYS_REGLCTL寄存器.
[7]	<b>SYSEBEN</b>	使能系统故障作为边沿检测刹车源（写保护） 0 =系统故障条件作为沿检测刹车源禁止. 1 = 系统故障条件作为沿检测刹车源使能. 注意：该寄存器写保护，参考SYS_REGLCTL寄存器
[6]	<b>Reserved</b>	保留.
[5]	<b>BRKP1EEN</b>	使能EPWMx_BRAKE1 引脚作为边沿检测刹车源（写保护） 0 = EPWMx_BRAKE1管脚作为边沿检测刹车源禁止. 1 = EPWMx_BRAKE1管脚作为边沿检测刹车源使能. 注意：该寄存器写保护，参考SYS_REGLCTL寄存器
[4]	<b>BRKPOEEN</b>	使能EPWMx_BRAKE0引脚作为边沿检测刹车源（写保护） 0 = EPWMx_BRAKE0管脚作为边沿检测刹车源禁止. 1 = EPWMx_BRAKE0管脚作为边沿检测刹车源使能. 注意：该寄存器写保护，参考SYS_REGLCTL寄存器
[3:2]	<b>Reserved</b>	保留.
[1]	<b>CPO1EBEN</b>	使能ACMP1_O 数字输出作为边沿检测刹车源（写保护） 0 = ACMP1_O 作为边沿检测刹车源禁止

		1 = ACMP1_O 作为边沿检测刹车源使能. 注意：该寄存器写保护，参考SYS_REGLCTL寄存器
[0]	<b>CPO0EBEN</b>	使能ACMP0_O数字输出作为边沿检测刹车源（写保护 0 = ACMP0_O 作为边沿检测刹车源禁止. 1 = ACMP0_O 作为边沿检测刹车源使能. 注意：该寄存器写保护，参考SYS_REGLCTL寄存器

EPWM POLCTL EPWM 管脚极性反转控制

寄存器	偏移量	R/W	描述	复位值
EPWM_POLCTL	EPWMx_BA+0xD4	R/W	EPWM 管脚极性反转寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved		PINV5	PINV4	PINV3	PINV2	PINV1	PINV0

位	描述	
[31:6]	Reserved	保留.
[n] n=0,1..5	PINVn	<b>EPWM管脚极性反转控制</b> 该寄存器控制EPWM输出的极性状态。每位n控制相应EPWM通道n 0 =禁止EPWM输出极性反转 1 =使能EPWM输出极性反转

EPWM POEN EPWM输出使能寄存器

寄存器	偏移量	R/W	描述	复位值
EPWM_POEN	EPWMx_BA+0xD8	R/W	EPWM 输出使能寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved		POEN5	POEN4	POEN3	POEN2	POEN1	POEN0

位	描述	
[31:6]	Reserved	保留.
[n] n=0,1..5	POENn	<b>EPWM 管脚输出使能</b> 每位n控制相应EPWM通道n. 0 = EPWM 管脚在三态模式 1 = EPWM 管脚在输出模式.

EPWM\_SWBRK EPWM软件刹车控制寄存器

寄存器	偏移量	R/W	描述	复位值
EPWM_SWBRK	EPWMx_BA+0xDC	W	EPWM 软件刹车控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved					BRKLTRG4	BRKLTRG2	BRKLTRG0
7	6	5	4	3	2	1	0
Reserved					BRKETRG4	BRKETRG2	BRKETRG0

位	描述	
[31:11]	Reserved	保留.
[8+n/2] n=0,2,4	BRKLTRGn	EPWM电平刹车软件触发（只写）（写保护） 写1到该位将触发电平刹车，并将EPWM_INTSTS1寄存器的BRKLIFn置1. 注：该寄存器写保护，参考SYS_REGLCTL寄存器
[7:3]	Reserved	保留.
[n/2] n=0,2,4	BRKETRGn	EPWM边沿刹车软件触发（只写）（写保护） 写1到该位将触发边沿刹车，并将EPWM_INTSTS1寄存器的BRKEIFn置1. 注：该寄存器写保护，参考SYS_REGLCTL寄存器

**EPWM\_INTEN0 EPWM 中断使能寄存器 0**

寄存器	偏移量	R/W	描述	复位值
EPWM_INTEN0	EPWMx_BA+0xE0	R/W	EPWM 中断使能寄存器 0	0x0000_0000

31	30	29	28	27	26	25	24
		Reserved	CMPDIEN5	CMPDIEN4	CMPDIEN3	CMPDIEN2	CMPDIEN1
23	22		21	20	19	18	17
		Reserved	CMPUIEN5	CMPUIEN4	CMPUIEN3	CMPUIEN2	CMPUIEN1
15	14		13	12	11	10	9
		Reserved	PIEN5	PIEN4	PIEN3	PIEN2	PIEN1
7	6		5	4	3	2	1
		Reserved	ZIEN5	ZIEN4	ZIEN3	ZIEN2	ZIEN1
							ZIEN0

位	描述	
[31:30]	Reserved	保留.
[24+n] n=0,1..5	CMPDIENn	<b>EPWM比较向下计数中断使能</b> 0 =禁止比较向下计数中断 1 =使能比较向下计数中断 注：在互补模式，CMPDIEN 1,3,5没有用，作为通道 0, 2, 4.的另一个CMPDIEN
[23:22]	Reserved	保留.
[16+n] n=0,1..5	CMPUIENn	<b>EPWM比较向上计数中断使能</b> 0 =禁止比较向上计数中断 1 =使能比较向上计数中断 注：在互补模式，CMPUIEN 1,3,5没有用，作为通道 0, 2, 4.的另一个CMPUIEN
[15:14]	Reserved	保留.
[8+n] n=0,1..5	PIENn	<b>EPWM 周期点中断使能</b> 每位n控制相应EPWM通道n 0 =禁止周期点中断 1 =使能周期点中断 注意1：上下计数方式周期点指的是中心点。 注意2：在互补模式，奇数通道读数总是0
[7:6]	Reserved	保留.
[n] n=0,1..5	ZIENn	<b>EPWM 0点中断使能</b> 0 = 0点中断使能. 1 = 0点中断使能. 注意：在互补模式奇数通道读数总是0.

EPWM\_INTEN1 EPWM中断使能寄存器1

寄存器	偏移量	R/W	描述	复位值
EPWM_INTEN1	EPWMx_BA+0xE4	R/W	EPWM 中断使能寄存器 1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved					BRKLIEN4_5	BRKLIEN2_3	BRKLIEN0_1
7	6	5	4	3	2	1	0
Reserved					BRKEIEN4_5	BRKEIEN2_3	BRKEIEN0_1

位	描述	
[31:11]	<b>Reserved</b>	保留.
[10]	<b>BRKLIEN4_5</b>	<b>EPWM通道4/5电平检测刹车中断使能 (写保护)</b> 0 =禁止通道4/5电平检测刹车中断 1 =使能通道4/5电平检测刹车中断 注意：该寄存器写保护，参考SYS_REGLCTL寄存器
[9]	<b>BRKLIEN2_3</b>	<b>EPWM通道2/3电平检测刹车中断使能 (写保护)</b> 0 =禁止通道2/3电平检测刹车中断 1 =使能通道2/3电平检测刹车中断 注意：该寄存器写保护，参考SYS_REGLCTL寄存器
[8]	<b>BRKLIEN0_1</b>	<b>EPWM通道0/1电平检测刹车中断使能 (写保护)</b> 0 =禁止通道0/1电平检测刹车中断 1 =使能通道0/1电平检测刹车中断 注：该寄存器写保护，参考SYS_REGLCTL寄存器
[7:3]	<b>Reserved</b>	保留.
[2]	<b>BRKEIEN4_5</b>	<b>EPWM通道4/5边沿检测刹车中断使能 (写保护)</b> 0 =禁止通道4/5边沿检测刹车中断 1 =使能通道4/5边沿检测刹车中断 注：该寄存器写保护，参考SYS_REGLCTL寄存器
[1]	<b>BRKEIEN2_3</b>	<b>EPWM通道2/3边沿检测刹车中断使能 (写保护)</b> 0 =禁止通道2/3边沿检测刹车中断 1 =使能通道2/3边沿检测刹车中断 注意：该寄存器写保护，参考SYS_REGLCTL寄存器
[0]	<b>BRKEIEN0_1</b>	<b>EPWM通道0/1边沿检测刹车中断使能 (写保护)</b>

		0 = 禁止通道0/1边沿检测刹车中断
		1 = 使能通道0/1边沿检测刹车中断
		注意：该寄存器写保护，参考SYS_REGLCTL寄存器

EPWM\_INTSTS0 EPWM中断标志寄存器0

寄存器	偏移量	R/W	描述	复位值
EPWM_INTSTS0	EPWMx_BA+0xE8	R/W	EPWM 中断标志寄存器 0	0x0000_0000

31	30	29	28	27	26	25	24
	Reserved	CMPDIF5	CMPDIF4	CMPDIF3	CMPDIF2	CMPDIF1	CMPDIF0
23	22	21	20	19	18	17	16
	Reserved	CMPUIF5	CMPUIF4	CMPUIF3	CMPUIF2	CMPUIF1	CMPUIF0
15	14	13	12	11	10	9	8
	Reserved	PIF5	PIF4	PIF3	PIF2	PIF1	PIF0
7	6	5	4	3	2	1	0
	Reserved	ZIF5	ZIF4	ZIF3	ZIF2	ZIF1	ZIF0

位	描述	
[31:30]	Reserved	保留.
[24+n] n=0,1..5	CMPDIFn	<b>EPWM比较向下计数中断标志</b> 当EPWM计数器向下计数到EPWM_CMPDATn，标志被硬件置1。软件写1到该位将清标志。 注意：在互补模式，CMPDIF 1,3,5作为通道 0, 2, 4.的另一个CMPDIF
[23:22]	Reserved	保留.
[16+n] n=0,1..5	CMPUIFn	<b>EPWM比较向上计数中断标志</b> 当EPWM计数器向上计数到EPWM_CMPDATn，标志被硬件置1。软件写1到该位将清标志。 注意：在互补模式，CMPDIF 1,3,5作为通道 0, 2, 4.的另一个CMPUIF
[15:14]	Reserved	保留.
[8+n] n=0,1..5	PIFn	<b>EPWM 周期点中断标志</b> 当EPWM计数值达到EPWM_PERIODn，该位被硬件置1，软件写1到该位将清标志。
[7:6]	Reserved	保留.
[n] n=0,1..5	ZIFn	<b>EPWM 零点中断标志</b> 当EPWM计数器减到0该位被硬件置1，硬件对该位置1，软件写1到该位将清标志

EPWM\_INTSTS1 EPWM中断标志寄存器1

寄存器	偏移量	R/W	描述	复位值
EPWM_INTSTS1	EPWMx_BA+0xEC	R/W	EPWM 中断标志寄存器 1	0x0000_0000

31	30	29	28	27	26	25	24
		Reserved	BRKLSTS5	BRKLSTS4	BRKLSTS3	BRKLSTS2	BRKLSTS1
23	22	21	20	19	18	17	16
		Reserved	BRKESTS5	BRKESTS4	BRKESTS3	BRKESTS2	BRKESTS1
15	14	13	12	11	10	9	8
		Reserved	BRKLIF5	BRKLIF4	BRKLIF3	BRKLIF2	BRKLIF1
7	6	5	4	3	2	1	0
		Reserved	BRKEIF5	BRKEIF4	BRKEIF3	BRKEIF2	BRKEIF1

位	描述	
[31:30]	Reserved	保留.
[29]	BRKLSTS5	<b>EPWM通道5电平检测刹车状态（只读）</b> 0 = EPWM通道5电平检测刹车状态被释放。 1 = 当EPWM通道5电平检测到任何一个已被使能的刹车源有个下降沿时，该位被置1表示EPWM通道5处在刹车状态。 注意：该位只读并被硬件自动清除。当已使能刹车源恢复到高电平，直到当前周期完成后EPWM输出才会释放刹车状态。EPWM波形将在下个完整的EPWM周期开始输出。
[28]	BRKLSTS4	<b>EPWM通道4电平检测刹车状态（只读）</b> 0 = EPWM通道4电平检测刹车状态被释放。 1 = 当EPWM通道4电平检测到任何一个已被使能的刹车源有个下降沿时，该位被置1表示EPWM通道4处在刹车状态。 注意：该位只读并被硬件自动清除。当已使能刹车源恢复到高电平，直到当前周期完成后EPWM输出才会释放刹车状态。EPWM波形将在下个完整的EPWM周期开始输出。
[27]	BRKLSTS3	<b>EPWM通道3电平检测刹车状态（只读）</b> 0 = EPWM通道3电平检测刹车状态被释放。 1 = 当EPWM通道3电平检测到任何一个已被使能的刹车源有个下降沿时，该位被置1表示EPWM通道3处在刹车状态。 注意：该位只读并被硬件自动清除。当已使能刹车源恢复到高电平，直到当前周期完成后EPWM输出才会释放刹车状态。EPWM波形将在下个完整的EPWM周期开始输出。
[26]	BRKLSTS2	<b>EPWM通道2电平检测刹车状态（只读）</b> 0 = EPWM通道2电平检测刹车状态被释放。 1 = 当EPWM通道2电平检测到任何一个已被使能的刹车源有个下降沿时，该位被置1表示EPWM通道2处在刹车状态。 注意：该位只读并被硬件自动清除。当已使能刹车源恢复到高电平，直到当前周期完成后EPWM输出才会释放刹车状态。EPWM波形将在下个完整的EPWM周期开始输出。
[25]	BRKLSTS1	<b>EPWM通道1电平检测刹车状态（只读）</b>

		0 =EPWM通道1电平检测刹车状态被释放。 1 =当EPWM通道1电平检测到任何一个已被使能的刹车源有个下降沿时，该位被置1表示EPWM通道1处在刹车状态。 注意：该位只读并被硬件自动清除。当已使能刹车源恢复到高电平，直到当前周期完成后EPWM输出才会释放刹车状态。EPWM波形将在下个完整的EPWM周期开始输出。
[24]	<b>BRKLSTS0</b>	<b>EPWM通道0电平检测刹车状态（只读）</b> 0 = EPWM通道0电平检测刹车状态被释放。 1 =当EPWM通道0电平检测到任何一个已被使能的刹车源有个下降沿时，该位被置1表示EPWM通道0处在刹车状态。 注意：该位只读并被硬件自动清除。当已使能刹车源恢复到高电平，直到当前周期完成后EPWM输出才会释放刹车状态。EPWM波形将在下个完整的EPWM周期开始输出。
[23:22]	<b>Reserved</b>	保留.
[21]	<b>BRKESTS5</b>	<b>EPWM通道5边沿检测刹车状态（只读）</b> 0 = EPWM通道5边沿检测刹车状态释放 1 =当EPWM通道5边沿刹车检测到任何被使能的刹车源有一个下降沿时，该位被置1表示通道5处在刹车状态，该位写1清除。
[20]	<b>BRKESTS4</b>	<b>EPWM通道4边沿检测刹车状态（只读）</b> 0 =EPWM通道4边沿检测刹车状态释放 1 =. 当EPWM通道4边沿刹车检测到任何被使能的刹车源有一个下降沿时，该位被置1表示通道4处在刹车状态，该位写1清除。
[19]	<b>BRKESTS3</b>	<b>EPWM通道3边沿检测刹车状态（只读）</b> 0 = EPWM通道3边沿检测刹车状态释放 1 =当EPWM通道3边沿刹车检测到任何被使能的刹车源有一个下降沿时，该位被置1表示通道3处在刹车状态，该位写1清除。
[18]	<b>BRKESTS2</b>	<b>EPWM通道2边沿检测刹车状态（只读）</b> 0 =EPWM通道2边沿检测刹车状态释放 1 =当EPWM通道2边沿刹车检测到任何被使能的刹车源有一个下降沿时，该位被置1表示通道2处在刹车状态，该位写1清除。
[17]	<b>BRKESTS1</b>	<b>EPWM通道1边沿检测刹车状态（只读）</b> 0 = EPWM通道1边沿检测刹车状态释放 1 =当EPWM通道1边沿刹车检测到任何被使能的刹车源有一个下降沿时，该位被置1表示通道1处在刹车状态，该位写1清除。
[16]	<b>BRKESTS0</b>	<b>EPWM通道0边沿检测刹车状态（只读）</b> 0 = EPWM通道0边沿检测刹车状态释放 1 =当EPWM通道0边沿刹车检测到任何被使能的刹车源有一个下降沿时，该位被置1表示通道0处在刹车状态，该位写1清除。
[15:14]	<b>Reserved</b>	保留.
[13]	<b>BRKLIF5</b>	<b>EPWM通道5电平检测刹车中断标志（写保护）</b> 0 =EPWM通道5电平检测刹车事件没发生 1 =当EPWM通道5电平检测中断事件发生，该位被置1，该位写1清除。 注：该寄存器写保护，参考SYS_REGLCTL寄存器
[12]	<b>BRKLIF4</b>	<b>EPWM通道4电平检测刹车中断标志（写保护）</b> 0 = WM通道4电平检测刹车事件没发生 1 =当EPWM通道4电平检测中断事件发生，该位被置1，该位写1清除。

		注：该寄存器写保护，参考SYS_REGLCTL寄存器
[11]	<b>BRKLIF3</b>	<b>EPWM通道3电平检测刹车中断标志（写保护）</b> 0 = EPWM通道3电平检测刹车事件没发生 1 = 当EPWM通道3电平检测中断事件发生，该位被置1，该位写1清除。 注：该寄存器写保护，参考SYS_REGLCTL寄存器
[10]	<b>BRKLIF2</b>	<b>EPWM通道2电平检测刹车中断标志（写保护）</b> 0 = EPWM通道2电平检测刹车事件没发生 1 = 当EPWM通道2电平检测中断事件发生，该位被置1，该位写1清除。 注：该寄存器写保护，参考SYS_REGLCTL寄存器
[9]	<b>BRKLIF1</b>	<b>EPWM通道1电平检测刹车中断标志（写保护）</b> 0 = EPWM通道1电平检测刹车事件没发生 1 = 当EPWM通道1电平检测中断事件发生，该位被置1，该位写1清除。 注：该寄存器写保护，参考SYS_REGLCTL寄存器
[8]	<b>BRKLIF0</b>	<b>EPWM通道0电平检测刹车中断标志（写保护）</b> 0 = EPWM通道0电平检测刹车事件没发生 1 = 当EPWM通道0电平检测中断事件发生，该位被置1，该位写1清除。 注：该寄存器写保护，参考SYS_REGLCTL寄存器
[7:6]	<b>Reserved</b>	保留.
[5]	<b>BRKEIF5</b>	<b>EPWM通道5边沿检测刹车中断标志（写保护）</b> 0 = EPWM通道5边沿检测刹车事件没发生 1 = 当EPWM通道5边沿检测中断事件发生，该位被置1，该位写1清除。 注：该寄存器写保护，参考SYS_REGLCTL寄存器
[4]	<b>BRKEIF4</b>	<b>EPWM通道4边沿检测刹车中断标志（写保护）</b> 0 = 通道4边沿检测刹车事件没发生 1 = 当EPWM通道4边沿检测中断事件发生，该位被置1，该位写1清除。 注：该寄存器写保护，参考SYS_REGLCTL寄存器
[3]	<b>BRKEIF3</b>	<b>EPWM通道3边沿检测刹车中断标志（写保护）</b> 0 = EPWM通道3边沿检测刹车事件没发生 1 = 当EPWM通道3边沿检测中断事件发生，该位被置1，该位写1清除。 注：该寄存器写保护，参考SYS_REGLCTL寄存器
[2]	<b>BRKEIF2</b>	<b>EPWM通道2边沿检测刹车中断标志（写保护）</b> 0 = EPWM通道2边沿检测刹车事件没发生 1 = 当EPWM通道2边沿检测中断事件发生，该位被置1，该位写1清除。 注：该寄存器写保护，参考SYS_REGLCTL寄存器
[1]	<b>BRKEIF1</b>	<b>EPWM通道1边沿检测刹车中断标志（写保护）</b> 0 = 通道1边沿检测刹车事件没发生 1 = 当EPWM通道1边沿检测中断事件发生，该位被置1，该位写1清除。 注：该寄存器写保护，参考SYS_REGLCTL寄存器
[0]	<b>BRKEIF0</b>	<b>EPWM通道0边沿检测刹车中断标志（写保护）</b> 0 = EPWM通道0边沿检测刹车事件没发生 1 = 当EPWM通道0边沿检测中断事件发生，该位被置1，该位写1清除。

[注: 该寄存器写保护, 参考SYS\_REGLCTL寄存器]

EPWM\_DACTRGEN EPWM触发 DAC使能寄存器

寄存器	偏移量	R/W	描述	复位值
EPWM_DACTRGEN	EPWMx_BA+0xF4	R/W	EPWM 触发 DAC 使能寄存器	0x0000_0000

31	30	29	28	27	26	25	24
		Reserved	CDTRGEN5	CDTRGEN4	CDTRGEN3	CDTRGEN2	CDTRGEN1
23	22	21	20	19	18	17	16
		Reserved	CUTRGEN5	CUTRGEN4	CUTRGEN3	CUTRGEN2	CUTRGEN1
15	14	13	12	11	10	9	8
		Reserved	PTE5	PTE4	PTE3	PTE2	PTE1
7	6	5	4	3	2	1	0
		Reserved	ZTE5	ZTE4	ZTE3	ZTE2	ZTE1
							ZTE0

位	描述	
[31:30]	Reserved	保留.
[24+n] n=0,1..5	CDTRGEN	<p><b>EPWM 比较器向下计数点触发DAC 使能</b>            如果该位置1, 当EPWM计数器向下计数到CMPDAT , EPWM 可以触发DAC启动。            0 = EPWM比较器向下计数点触发DAC功能禁止.            1 = EPWM 比较器向下计数点触发 DAC 功能使能.</p> <p><b>注意1:</b> 当EPWM 计数器工作在向上计数模式., 该位应该保持 0  <b>注意2:</b> 在互补模式下, CDTRGE1, 3, 5作为通道 0, 2, 4.的另一个CDTRGE</p>
[23:22]	Reserved	保留.
[16+n] n=0,1..5	CUTRGEN	<p><b>EPWM 比较器向上计数点触发 DAC使能</b>            如果该位置1, 当EPWM计数器向上计数到CMPDAT , EPWM 可以触发DAC启动。            0 = EPWM比较器向上计数点触发DAC功能禁止.            1 = EPWM比较器向上计数点触发DAC功能使能</p> <p><b>注意1:</b> 当EPWM 计数器工作在向下计数模式., 该位应该保持 0  <b>注意2:</b> 在互补模式下, CDTRGE1, 3, 5作为通道 0, 2, 4.的另一个CUTRGE</p>
[15:14]	Reserved	保留.
[8+n] n=0,1..5	PTEn	<p><b>EPWM周期点触发DAC使能</b>            如果该位设置为1, 当EPWM向上计数器计到(PERIODn+1)EPWM可以触发DAC开始转换。            0 = EPWM 周期点触发 DAC 功能禁止.            1 = EPWM周期点触发 DAC 功能使能</p>
[7:6]	Reserved	保留.
[n] n=0,1..5	ZTEn	<p><b>EPWM 0点触发DAC 使能</b>            如果该位置1, 当EPWM向下计数器计到0, EPWM 可以触发EADC/DAC/DMA 开始动作。            0 = EPWM 0点触发DAC功能禁止.</p>

		1 = EPWM 0点触发DAC功能使能.
--	--	-----------------------

**EPWM\_EADCTS0 EPWM触发EADC源选择寄存器0**

寄存器	偏移量	R/W	描述	复位值
EPWM_EADC_TS0	EPWMx_BA+0xF8	R/W	EPWM 触发 EADC 源选择寄存器 0	0x0000_0000

31	30	29	28	27	26	25	24
TRGEN3	Reserved			TRGSEL3			
23	22	21	20	19	18	17	16
TRGEN2	Reserved			TRGSEL2			
15	14	13	12	11	10	9	8
TRGEN1	Reserved			TRGSEL1			
7	6	5	4	3	2	1	0
TRGEN0	Reserved			TRGSEL0			

位	描述	
[31]	TRGEN3	EPWM_CH3 触发 EADC 使能位
[30:28]	Reserved	保留.
[27:24]	TRGSEL3	<p><b>EPWM_CH3 触发 EADC源选择</b></p> <p>0000 = EPWM_CH2 0点.      0001 = EPWM_CH2 周期点.      0010 = EPWM_CH2 0点或周期点      0011 = EPWM_CH2 向上计数到CMPDAT 值点.      0100 = EPWM_CH2向下计数到CMPDAT 值点      0101 = EPWM_CH3 0点      0110 = EPWM_CH3周期点.      0111 = EPWM_CH3 0点或周期点.      1000 = EPWM_CH3向上计数到CMPDAT 值点.      1001 = EPWM_CH3向下计数到CMPDAT 值点      1010 = EPWM_CH0向上计数到自由CMPDAT 值点      1011 = EPWM_CH0向下计数到自由CMPDAT 值点.      1100 = EPWM_CH2向上计数到自由CMPDAT 值点.      1101 = EPWM_CH2向下计数到自由CMPDAT 值点      1110 = EPWM_CH4向上计数到自由CMPDAT 值点.      1111 = EPWM_CH4向下计数到自由CMPDAT 值点</p>
[23]	TRGEN2	EPWM_CH2 触发 EADC 使能位
[22:20]	Reserved	保留.
[19:16]	TRGSEL2	<p><b>EPWM_CH2触发 EADC源选择</b></p> <p>0000 = EPWM_CH2 0点.      0001 = EPWM_CH2 周期点.</p>

		0010 = EPWM_CH2 0点或周期点 0011 = EPWM_CH2向上计数到CMPDAT 值点 0100 = EPWM_CH2向下计数到CMPDAT 值点. 0101 = EPWM_CH3 0点. 0110 = EPWM_CH3 周期点 0111 = EPWM_CH3 0点或周期点. 1000 = EPWM_CH3向上计数到CMPDAT 值点. 1001 = EPWM_CH3向下计数到CMPDAT 值点 1010 = EPWM_CH0向上计数到自由CMPDAT 值点. 1011 = EPWM_CH0向下计数到自由CMPDAT 值点 1100 = EPWM_CH2向上计数到自由CMPDAT 值点. 1101 = EPWM_CH2向下计数到自由CMPDAT 值点. 1110 = EPWM_CH4向上计数到自由CMPDAT 值点 1111 = EPWM_CH4向下计数到自由CMPDAT 值点
[15]	<b>TRGEN1</b>	EPWM_CH1 触发 EADC 使能位
[14:12]	<b>Reserved</b>	保留.
[11:8]	<b>TRGSEL1</b>	<b>EPWM_CH1 触发 EADC 源选择</b> 0000 = EPWM_CH0 0点 0001 = EPWM_CH0周期点. 0010 = EPWM_CH0 0点或周期点 0011 = EPWM_CH0向上计数到CMPDAT 值点. 0100 = EPWM_CH0向下计数到CMPDAT 值点. 0101 = EPWM_CH1 0点. 0110 = EPWM_CH1周期点 0111 = EPWM_CH1 0点或周期点 1000 = EPWM_CH1向上计数到CMPDAT 值点 1001 = EPWM_CH1向下计数到CMPDAT 值点 1010 = EPWM_CH0向上计数到自由CMPDAT 值点. 1011 = EPWM_CH0向下计数到自由CMPDAT 值点 1100 = EPWM_CH2向上计数到自由CMPDAT 值点 1101 = EPWM_CH2向下计数到自由CMPDAT 值点 1110 = EPWM_CH4向上计数到自由CMPDAT 值点. 1111 = EPWM_CH4向下计数到自由CMPDAT 值点.
[7]	<b>TRGEN0</b>	EPWM_CH0 触发 EADC 使能位
[6:4]	<b>Reserved</b>	保留.
[3:0]	<b>TRGSEL0</b>	<b>EPWM_CH0 触发 EADC 源选择</b> 0000 = EPWM_CH0 0点 0001 = EPWM_CH0周期点. 0010 = EPWM_CH0 0点或周期点 0011 = EPWM_CH0向上计数到CMPDAT 值点. 0100 = EPWM_CH0向下计数到CMPDAT 值点. 0101 = EPWM_CH1 0点. 0110 = EPWM_CH1周期点

		<p>0111 = EPWM_CH1 0点或周期点 1000 = EPWM_CH1向上计数到CMPDAT 值点 1001 = EPWM_CH1向下计数到CMPDAT 值点 1010 = EPWM_CH0向上计数到自由CMPDAT 值点. 1011 = EPWM_CH0向下计数到自由CMPDAT 值点 1100 = EPWM_CH2向上计数到自由CMPDAT 值点 1101 = EPWM_CH2向下计数到自由CMPDAT 值点 1110 = EPWM_CH4向上计数到自由CMPDAT 值点. 1111 = EPWM_CH4向下计数到自由CMPDAT 值点.</p>
--	--	--

EPWM\_EADCTS1 EPWM 触发EADC源选择寄存器1

寄存器	偏移量	R/W	描述	复位值
EPWM_EADC_TS1	EPWMx_BA+0xFC	R/W	EPWM 触发 EADC 源选择寄存器 1	0x0000_0000

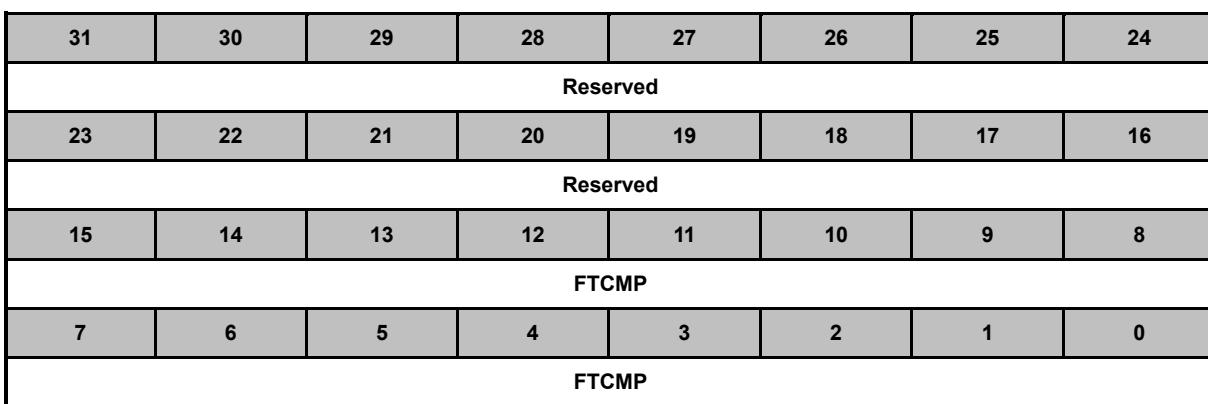
31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
TRGEN5	Reserved			TRGSEL5			
7	6	5	4	3	2	1	0
TRGEN4	Reserved			TRGSEL4			

位	描述
[31:16]	<b>Reserved</b> 保留.
[15]	<b>TRGEN5</b> EPWM_CH5 触发EADC使能位
[14:12]	<b>Reserved</b> 保留
[11:8]	<b>TRGSEL5</b> EPWM_CH5 触发EADC源选择 0000 = EPWM_CH4 0点. 0001 = EPWM_CH4 周期点. 0010 = EPWM_CH4 0点或周期点. 0011 = EPWM_CH4 向上计数到CMPDAT 值点. 0100 = EPWM_CH4 向下计数到CMPDAT 值点. 0101 = EPWM_CH5 0点 0110 = EPWM_CH5 周期点. 0111 = EPWM_CH5 0点或周期点. 1000 = EPWM_CH5 向上计数到CMPDAT 值点. 1001 = EPWM_CH5 向下计数到CMPDAT 值点 1010 = EPWM_CH0 向上计数到自由CMPDAT 值点. 1011 = EPWM_CH0 向下计数到自由CMPDAT 值点 1100 = EPWM_CH2 向上计数到自由CMPDAT 值点 1101 = EPWM_CH2 向下计数到自由CMPDAT 值点 1110 = EPWM_CH4 向上计数到自由CMPDAT 值点. 1111 = EPWM_CH4 向下计数到自由CMPDAT 值点.
[7]	<b>TRGEN4</b> EPWM_CH4 触发 EADC 使能位
[6:4]	<b>Reserved</b> 保留.
[3:0]	<b>TRGSEL4</b> EPWM_CH4 触发 EADC 源选择 0000 = EPWM_CH4 0点.

		<p>0001 = EPWM_CH4周期点 0010 = EPWM_CH4 0点或周期点 0011 = EPWM_CH4向上计数到CMPDAT 值点. 0100 = EPWM_CH4向下计数到CMPDAT 值点. 0101 = EPWM_CH5 0点 0110 = EPWM_CH5周期点 0111 = EPWM_CH5 0点或周期点 1000 = EPWM_CH5向上计数到CMPDAT 值点 1001 = EPWM_CH5向下计数到CMPDAT 值点. 1010 = EPWM_CH0向上计数到自由CMPDAT 值点. 1011 = EPWM_CH0向下计数到自由CMPDAT 值点 1100 = EPWM_CH2向上计数到自由CMPDAT 值点 1101 = EPWM_CH2向下计数到自由CMPDAT 值点 1110 = EPWM_CH4向上计数到自由CMPDAT 值点. 1111 = EPWM_CH4向下计数到自由CMPDAT 值点.</p>
--	--	---

EPWM\_FTCMPDAT0\_1, 2, 3, 4, 5 EPWM自由触发比较寄存器0\_1, 2, 3, 4, 5

寄存器	偏移量	R/W	描述	复位值
EPWM_FTCM_PDATA0_1	EPWMx_BA+0x100	R/W	EPWM自由触发比较寄存器0/1	0x0000_0000
EPWM_FTCM_PDATA2_3	EPWMx_BA+0x104	R/W	EPWM自由触发比较寄存器2/3	0x0000_0000
EPWM_FTCM_PDATA4_5	EPWMx_BA+0x108	R/W	EPWM自由触发比较寄存器4/5	0x0000_0000



位	描述	
[31:16]	Reserved	保留.
[15:0]	FTCMP	<b>EPWM自由触发比较寄存器</b> FTCMP 用于与偶数通道的CNTR 比较来触发EADC. FTCMPDAT0, 2, 4 对应互补组是 EPWM_CH0 和 EPWM_CH1, 、 EPWM_CH2 和 EPWM_CH3 、 EPWM_CH4 和 EPWM_CH5.

EPWM\_SSCTL EPWM同步开始控制寄存器

寄存器	偏移量	R/W	描述	复位值
EPWM_SSCTL	EPWMx_BA+0x110	R/W	EPWM 同步开始控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved						SSRC	
7	6	5	4	3	2	1	0
Reserved		SSEN5	SSEN4	SSEN3	SSEN2	SSEN1	SSEN0

位	描述	
[31:10]	Reserved	保留.
[9:8]	SSRC	<b>EPWM 同步启动源选择</b> 00 = 同步启动源来自 EPWM0. 01 = 同步启动源来自EPWM1. 10 =同步启动源来自BEPWM0. 11 =同步启动源来自BEPWM1.
[7:6]	Reserved	保留.
[n] n=0,1..5	SSENn	<b>EPWM 同步开始 功能使能</b> 当同步开始功能使能， EPWM计数器使能寄存器(EPWM_CNTEN)可以通过对写EPWM同步开始触发位(CNTSEN)来使能。 0 = EPWM 同步开始功能禁止. 1 = EPWM 同步开始功能使能.

**EPWM\_SSTRG EPWM同步开始触发寄存器**

寄存器	偏移量	R/W	描述	复位值
EPWM_SSTRG	EPWMx_BA+0x114	W	EPWM 同步开始触发寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved							CNTSEN

位	描述	
[31:1]	Reserved	保留.
[0]	CNTSEN	<p><b>EPWM 计数器同步开始使能 (只写)</b></p> <p>PMW 计数器同步使能功能用于选择EPWM通道 (包括 EPWM0_CHx 和 EPWM1_CHx) 开始同时计数</p> <p>如果相关EPWM通道计数器同步开始功能被使能，该位置1也会置位计数器使能位(CNTENn, n 代表通道0 到 5)。</p>

EPWM\_LEBCTL EPWM Leading Edge Blanking 控制寄存器

寄存器	偏移量	R/W	描述	复位值
EPWM_LEBCTL	EPWMx_BA+0x118	R/W	EPWM Leading Edge Blanking 控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved						TRGTYPE	
15	14	13	12	11	10	9	8
Reserved						SRCEN4	SRCEN2
7	6	5	4	3	2	1	0
Reserved						LEBEN	

位	描述	
[31:18]	Reserved	保留.
[17:16]	TRGTYPE	<b>EPWM Leading Edge Blanking 触发模式</b> 0 = 当侦测到 leading edge blanking 源有上升沿, blanking 计数器开始计数. 1 = 当侦测到 leading edge blanking 源有下降沿, blanking 计数器开始计数 2 = 当侦测到 leading edge blanking 源有下降沿或上升沿, blanking 计数器开始计数 3 = 保留.
[15:11]	Reserved	保留.
[10]	SRCEN4	<b>EPWM Leading Edge Blanking 源来自 EPWM_CH4 Enable 位</b> 0 = EPWM Leading Edge Blanking 源来自 EPWM_CH4 禁止. 1 = EPWM Leading Edge Blanking 源来自 EPWM_CH4 使能.
[9]	SRCEN2	<b>EPWM Leading Edge Blanking 源来自 EPWM_CH2 Enable 位</b> 0 = EPWM Leading Edge Blanking 源来自 EPWM_CH2 禁止. 1 = EPWM Leading Edge Blanking 源来自 EPWM_CH2 使能.
[8]	SRCEN0	<b>EPWM Leading Edge Blanking 源来自 EPWM_CH0 Enable 位</b> 0 = EPWM Leading Edge Blanking 源来自 EPWM_CH0 禁止. 1 = EPWM Leading Edge Blanking 源来自 EPWM_CH0 使能.
[7:1]	Reserved	保留.
[0]	LEBEN	<b>EPWM Leading Edge Blanking 使能位</b> 0 = EPWM Leading Edge Blanking 禁止. 1 = EPWM Leading Edge Blanking 使能.

EPWM LEBCNT EPWM Leading Edge Blanking 计数器寄存器

寄存器	偏移量	R/W	描述	复位值
EPWM_LEBCNT	EPWMx_BA+0x11C	R/W	EPWM Leading Edge Blanking 计数器寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							LEBCNT
7	6	5	4	3	2	1	0
LEBCNT							

位	描述	
[31:9]	Reserved	保留.
[8:0]	LEBCNT	<b>EPWM Leading Edge Blanking 计数器</b> 该计数器值决定了 leading edge blanking 窗口大小: Blanking 窗口大小 = LEBCNT+1, LEB 计数器时钟基于 ECLK.

**EPWM STATUS EPWM状态寄存器**

寄存器	偏移量	R/W	描述	复位值
EPWM_STATUS	EPWMx_BA+0x120	R/W	EPWM 状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							DACTRGF
23	22	21	20	19	18	17	16
Reserved		EADCTRGF5	EADCTRGF4	EADCTRGF3	EADCTRGF2	EADCTRGF1	EADCTRGF0
15	14	13	12	11	10	9	8
Reserved					SYNCINF4	SYNCINF2	SYNCINFO
7	6	5	4	3	2	1	0
Reserved		CNTMAXF5	CNTMAXF4	CNTMAXF3	CNTMAXF2	CNTMAXF1	CNTMAXF0

位	描述	
[31:25]	Reserved	保留.
[24]	DACTRGF	<b>DAC开始转换标志</b> 0 = 表示没有 DAC开始转换触发事件发生 1 =表示有 DAC开始转换触发事件发生。该位可以软件写1清零.
[23:22]	Reserved	保留.
[16+n] n=0,1..5	EADCTRGFn	<b>EADC 开始转换标志</b> 0 =表示没有EADC开始转换触发事件发生. 1 =表示有EADC开始转换触发事件发生。该位可以软件写1清零
[15:11]	Reserved	保留.
[8+n/2] n=0,2,4	SYNCINFn	<b>输入同步锁存标志</b> 0 =表示没有 SYNC_IN事件发生. 1 =表示有SYNC_IN 事件发生。该位可以软件写1清零
[7:6]	Reserved	保留.
[n] n=0,1..5	CNTMAXFn	<b>时基计数器等于0xFFFF 锁存标志</b> 0 = 表示时基计数器 从未达到最大值 0xFFFF. 1 = 表示时基计数器达到其最大值。该位可以软件写1清零

EPWM\_IFAn EPWM 中断标志累加器寄存器

寄存器	偏移量	R/W	描述	复位值
EPWM_IFA0	EPWMx_BA+0x130	R/W	EPWM 中断标志累加器寄存器 0	0x0000_0000
EPWM_IFA1	EPWMx_BA+0x134	R/W	EPWM中断标志累加器寄存器1	0x0000_0000
EPWM_IFA2	EPWMx_BA+0x138	R/W	EPWM中断标志累加器寄存器2	0x0000_0000
EPWM_IFA3	EPWMx_BA+0x13C	R/W	EPWM中断标志累加器寄存器3	0x0000_0000
EPWM_IFA4	EPWMx_BA+0x140	R/W	EPWM中断标志累加器寄存器4	0x0000_0000
EPWM_IFA5	EPWMx_BA+0x144	R/W	EPWM中断标志累加器寄存器5	0x0000_0000

31	30	29	28	27	26	25	24
IFAEN	Reserved	IFASEL		Reserved			
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
IFACNT							
7	6	5	4	3	2	1	0
IFACNT							

位	描述
[31]	IFAEN EPWM_CHn 中断标志累加器使能位 0 = EPWM_CHn 中断标志累加器禁止 1 = EPWM_CHn 中断标志累加器使能.
[30]	Reserved 保留.
[29:28]	IFASEL EPWM_CHn 中断标志累加器源选择 000 = 通道n CNT 等于0. 001 = 通道n CNT 等于PERIOD. 010 = 通道n CNT 等于CMPU. 011 = 通道n CNT 等于CMPD.
[27:16]	Reserved 保留.
[15:0]	IFACNT EPWM_CHn 中断标志计数器 该寄存器定义多少次EPWM_CHn周期发生来置位IFAIFn以请求EPWM周期中断。 每IFACNT[15:0] 个EPWM周期， EPWM标志会置位。

EPWM\_AINTSTS EPWM 累加器中断标志寄存器

寄存器	偏移量	R/W	描述	复位值
EPWM_AINTSTS	EPWMx_BA+0x150	R/W	EPWM 累加器中断标志寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved		IFAI5	IFAI4	IFAI3	IFAI2	IFAI1	IFAI0

位	描述	
[31:6]	<b>Reserved</b>	保留.
[n] n=0,1..5	<b>IFAIFn</b>	<b>EPWM_CHn 中断标志累加器中断标志</b> 当条件和EPWM_IFAn寄存器中IFASEL匹配，该标志被硬件置位。

EPWM\_AINTEN EPWM 累加器中断使能寄存器

寄存器	偏移量	R/W	描述	复位值
EPWM_AINTE_N	EPWMx_BA+0x154	R/W	EPWM 累加器中断使能寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved		IFAIEN5	IFAIEN4	IFAIEN3	IFAIEN2	IFAIEN1	IFAIEN0

位	描述	
[31:6]	Reserved	保留.
[n] n=0,1..5	IFAIENn	EPWM_CHn 中断标志累加器中断使能位 0 = 中断标志累加器中断禁止. 1 = 中断标志累加器中断使能

EPWM\_APDMACTL EPWM 累加器 PDMA 控制寄存器

寄存器	偏移量	R/W	描述	复位值
EPWM_APDMACTL	EPWMx_BA+0x158	R/W	EPWM 累加器 PDMA 控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved		APDMAEN5	APDMAEN4	APDMAEN3	APDMAEN2	APDMAEN1	APDMAEN0

位	描述	
[31:6]	Reserved	保留.
[n] n=0,1..5	APDMAENn	通道 N 累加器 PDMA 使能位 0 = 通道 n PDMA 功能禁止. 1 = 通道 n PDMA 功能使能，用于通道n触发PDMA传输内存数据到寄存器。

**EPWM\_CAPINEN EPWM捕获输入使能寄存器**

寄存器	偏移量	R/W	描述	复位值
EPWM_CAPI NEN	EPWMx_BA+0x200	R/W	EPWM 捕获输入使能寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved		CAPINEN5	CAPINEN4	CAPINEN3	CAPINEN2	CAPINEN1	CAPINEN0

位	描述	
[31:6]	Reserved	保留.
[n] n=0,1..5	CAPINENn	捕获输入使能 0 =禁止EPWM输入捕获, EPWM通道捕获功能的输入总是被认为是0 1 =使能EPWM输入捕获, EPWM通道捕获功能的输入来自相关的复用管脚。

EPWM\_CAPCTL EPWM 捕获控制寄存器

寄存器	偏移量	R/W	描述	复位值
EPWM_CAPCTL	EPWMx_BA+0x204	R/W	EPWM 捕获控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
		Reserved	FCRLDEN5	FCRLDEN4	FCRLDEN3	FCRLDEN2	FCRLDEN1
23	22	21	20	19	18	17	16
		Reserved	RCRLDEN5	RCRLDEN4	RCRLDEN3	RCRLDEN2	RCRLDEN1
15	14	13	12	11	10	9	8
		Reserved	CAPINV5	CAPINV4	CAPINV3	CAPINV2	CAPINV1
7	6	5	4	3	2	1	0
		Reserved	CAPEN5	CAPEN4	CAPEN3	CAPEN2	CAPEN1
							CAPEN0

位	描述	
[31:30]	Reserved	保留.
[24+n] n=0,1..5	FCRLDENn	下降沿捕获重载使能 0 = 禁止下降沿捕获重载计数器 1 = 使能下降沿捕获重载计数器
[23:22]	Reserved	保留.
[16+n] n=0,1..5	RCRLDENn	上升沿捕获重载使能 0 = 禁止上升沿捕获重载计数器 1 = 使能上升沿捕获重载计数器
[15:14]	Reserved	保留.
[8+n] n=0,1..5	CAPINVn	捕获反向使能 0 = 禁止捕获源反向 1 = 使能捕获源反向，将来自GPIO的信号反向
[7:6]	Reserved	保留.
[n] n=0,1..5	CAPENn	捕获功能使能 0 = 禁止捕获功能。RCAPDAT/FCAPDAT不会更新。 1 = 使能捕获功能。当检测到输入信号的上升/下降沿时锁存EPWM计数器值并保存到RCAPDAT（上升沿锁存）和FCAPDAT（下降沿锁存）。

EPWM\_CAPSTS EPWM 捕获状态寄存器

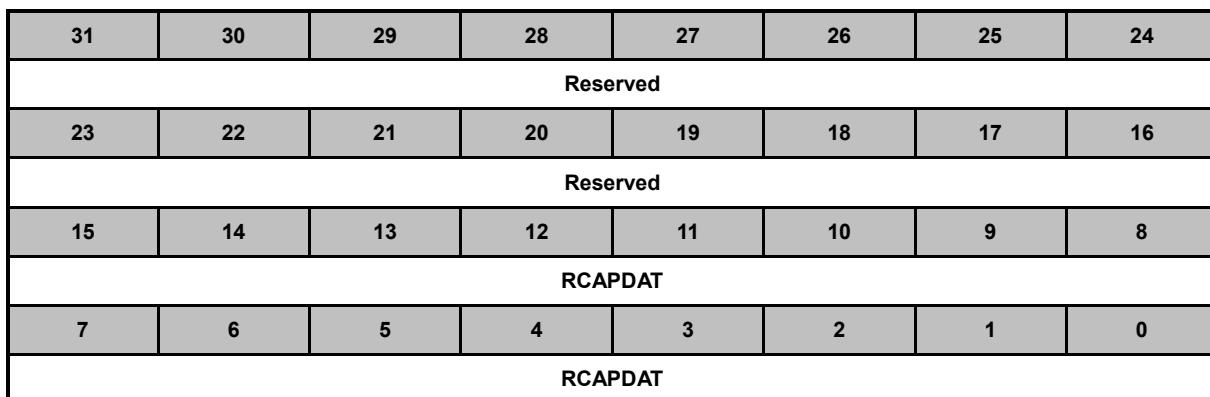
寄存器	偏移量	R/W	描述	复位值
EPWM_CAPSTS	EPWMx_BA+0x208	R	EPWM 捕获状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved		CFLIFOV5	CFLIFOV4	CFLIFOV3	CFLIFOV2	CFLIFOV1	CFLIFOV0
7	6	5	4	3	2	1	0
Reserved		CRLIFOV5	CRLIFOV4	CRLIFOV3	CRLIFOV2	CRLIFOV1	CRLIFOV0

位	描述	
[31:14]	<b>Reserved</b>	保留.
[8+n] n=0,1..5	<b>CFLIFOVn</b>	下降沿捕获锁存中断标志溢出状态（只读） 如果相应的CFLIF是1，且下降沿锁存事件发生，此标志为1。 注意：当用户清相应的CFLIF，该位将自动清零。
[7:6]	<b>Reserved</b>	保留.
[n] n=0,1..5	<b>CRLIFOVn</b>	上升沿捕获锁存中断标志溢出状态（只读） 如果相应的CRLIF是1，且上升沿捕获事件发生，此标志为1。 注：当用户清相应的CRLIF，该位将自动清零。

EPWM\_RCAPDAT 0~5 EPWM上升沿捕获数据寄存器0~5

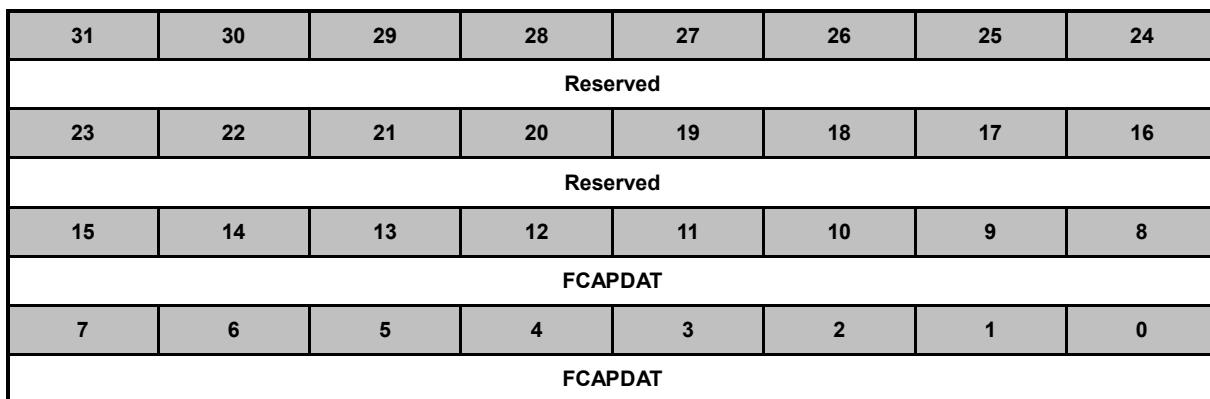
寄存器	偏移量	R/W	描述	复位值
EPWM_RCAPDAT0	EPWMx_BA+0x20C	R	EPWM 上升沿捕获数据寄存器0	0x0000_0000
EPWM_RCAPDAT1	EPWMx_BA+0x214	R	EPWM 上升沿捕获数据寄存器1	0x0000_0000
EPWM_RCAPDAT2	EPWMx_BA+0x21C	R	EPWM 上升沿捕获数据寄存器2	0x0000_0000
EPWM_RCAPDAT3	EPWMx_BA+0x224	R	EPWM 上升沿捕获数据寄存器3	0x0000_0000
EPWM_RCAPDAT4	EPWMx_BA+0x22C	R	EPWM 上升沿捕获数据寄存器4	0x0000_0000
EPWM_RCAPDAT5	EPWMx_BA+0x234	R	EPWM 上升沿捕获数据寄存器5	0x0000_0000



位	描述	
[31:16]	Reserved	保留.
[15:0]	RCAPDAT	EPWM上升沿捕获数据寄存器（只读） 当上升沿捕获条件发生，EPWM计数器值将被保存到该寄存器。

**EPWM\_FCAPDAT 0~5 EPWM 下降沿捕获数据寄存器0~5**

寄存器	偏移量	R/W	描述	复位值
EPWM_FCAPDAT0	EPWMx_BA+0x210	R	EPWM 下降沿捕获寄存器0	0x0000_0000
EPWM_FCAPDAT1	EPWMx_BA+0x218	R	EPWM 下降沿捕获寄存器1	0x0000_0000
EPWM_FCAPDAT2	EPWMx_BA+0x220	R	EPWM 下降沿捕获寄存器2	0x0000_0000
EPWM_FCAPDAT3	EPWMx_BA+0x228	R	EPWM 下降沿捕获寄存器3	0x0000_0000
EPWM_FCAPDAT4	EPWMx_BA+0x230	R	EPWM 下降沿捕获寄存器4	0x0000_0000
EPWM_FCAPDAT5	EPWMx_BA+0x238	R	EPWM 下降沿捕获寄存器5	0x0000_0000



位	描述		
[31:16]	Reserved	保留.	
[15:0]	FCAPDAT	EPWM下降沿捕获寄存器(只读) 当下降沿捕获条件发生，EPWM计数器值将被保存到该寄存器。	

EPWM\_PDMACTL EPWM PDMA控制寄存器

寄存器	偏移量	R/W	描述	复位值
EPWM_PDMA_CTL	EPWMx_BA+0x23C	R/W	EPWM PDMA 控制寄存器	0x0000_0000

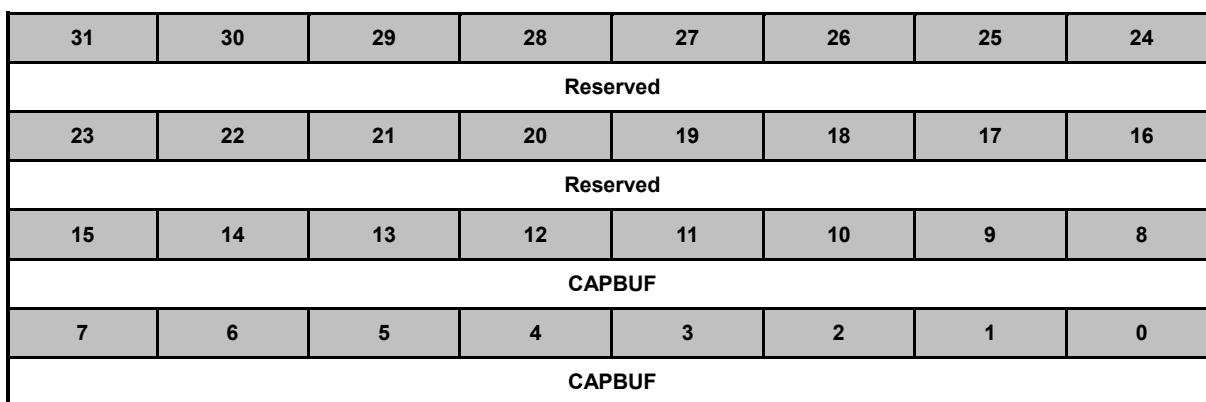
31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved			CHSEL4_5	CAPORD4_5	CAPMOD4_5		CHEN4_5
15	14	13	12	11	10	9	8
Reserved			CHSEL2_3	CAPORD2_3	CAPMOD2_3		CHEN2_3
7	6	5	4	3	2	1	0
Reserved			CHSEL0_1	CAPORD0_1	CAPMOD0_1		CHENO_1

位	描述	
[31:21]	Reserved	保留.
[20]	CHSEL4_5	选择 通道 4/5 用于PDMA传输 0 = 通道4. 1 = 通道5.
[19]	CAPORD4_5	捕获通道 4/5 上升沿/下降沿 顺序 设置该位用于确定，当CAPMOD4_5 =11时，是EPWM_RCAPDATA4/5 还是EPWM_FCAPDATA4/5作为第一次捕获的数据通过PDMA开始搬移到内存。 0 = EPWM_FCAPDATA4/5是先捕获并搬移到内存的数据 1 = EPWM_RCAPDATA4/5是先捕获并搬移到内存的数据
[18:17]	CAPMOD4_5	选择 EPWM_RCAPDATA4/5 或 EPWM_FCAPDATA4/5 来做PDMA 数据搬移 00 = 保留. 01 = EPWM_RCAPDATA4/5. 10 = EPWM_FCAPDATA4/5. 11 = EPWM_RCAPDATA4/5 和 EPWM_FCAPDATA4/5.
[16]	CHEN4_5	通道4/5 PDMA 使能 0 = 通道 4/5 PDMA功能禁止 1 =通道4/5 PDMA 功能使能，用于通道4/5 捕获数据并搬移到内存.
[15:13]	Reserved	保留.
[12]	CHSEL2_3	选择 通道 2/3用于PDMA传输 0 =通道2. 1 =通道3.
[11]	CAPORD2_3	捕获通道 2/3 上升沿/下降沿 顺序 设置该位用于确定，当CAPMOD2_3 =11时，是EPWM_RCAPDATA2/3 还是EPWM_FCAPDATA2/3作为第一次捕获的数据通过PDMA开始搬移到内存。

		0 = EPWM_FCAPDAT2/3是先捕获并搬移到内存的数据 1 = EPWM_RCAPDAT2/3是先捕获并搬移到内存的数据
[10:9]	CAPMOD2_3	选择 EPWM_RCAPDAT2/3 或 EPWM_FCAODAT2/3来做PDMA 数据搬移 00 = 保留. 01 = EPWM_RCAPDAT2/3. 10 = EPWM_FCAPDAT2/3. 11 = EPWM_RCAPDAT2/3 和 EPWM_FCAPDAT2/3.
[8]	CHEN2_3	通道2/3 PDMA 使能 0 = 通道 2/3 PDMA功能禁止 1 = 通道2/3功能使能，用于通道2/3捕获数据并搬移到内存
[7:5]	Reserved	保留.
[4]	CHSEL0_1	选择 通道0/1来做PDMA 数据搬移 0 = 通道0. 1 = 通道1.
[3]	CAPORD0_1	捕获通道0/1上升沿/下降沿 顺序 设置该位用于确定，当CAPMOD0_1 =11时，是EPWM_RCAPDATA0/1还是EPWM_FCAPDATA0/1作为第一次捕获的数据通过PDMA开始搬移到内存。 0 = EPWM_FCAPDATA0/1是先捕获并搬移到内存的数据 1 = EPWM_RCAPDATA0/1是先捕获并搬移到内存的数据
[2:1]	CAPMOD0_1	选择 EPWM_RCAPDATA0/1 或 EPWM_FCAPDATA0/1来做PDMA 数据搬移 00 = 保留. 01 = EPWM_RCAPDATA0/1. 10 = EPWM_FCAPDATA0/1. 11 = EPWM_RCAPDATA0/1 和 EPWM_FCAPDATA0/1.
[0]	CHENO_1	通道0/1 PDMA 使能 0 = 通道0/1 PDMA功能禁止. 1 = 通道0/1 PDMA功能使能，用于通道0/1捕获数据并搬移到memory

**EPWM\_PDMACAP 0\_1, 2\_3, 4\_5 EPWM 捕获通道0\_1, 2\_3, 4\_5 PDMA 寄存器**

寄存器	偏移量	R/W	描述	复位值
EPWM_PDMA_CAP0_1	EPWMx_BA+0x240	R	EPWM 捕获通道01 PDMA 寄存器	0x0000_0000
EPWM_PDMA_CAP2_3	EPWMx_BA+0x244	R	EPWM 捕获通道23 PDMA寄存器	0x0000_0000
EPWM_PDMA_CAP4_5	EPWMx_BA+0x248	R	EPWM 捕获通道45 PDMA寄存器	0x0000_0000



位	描述	
[31:16]	Reserved	保留.
[15:0]	CAPBUF	<b>EPWM 捕获 PDMA 寄存器 (只读)</b> 该寄存器用于作为一个 缓存，通过PDMA.来传输EPWM 捕获到的上升沿或下降沿数据到内存

EPWM CAPIEN EPWM捕获中断使能寄存器

寄存器	偏移量	R/W	描述	复位值
EPWM_CAPIEN	EPWMx_BA+0x250	R/W	EPWM 捕获中断使能寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved		CAPFIEN5	CAPFIEN4	CAPFIEN3	CAPFIEN2	CAPFIEN1	CAPFIENO
7	6	5	4	3	2	1	0
Reserved		CAPRIEN5	CAPRIEN4	CAPRIEN3	CAPRIEN2	CAPRIEN1	CAPRIENO

位	描述	
[31:14]	Reserved	保留.
[8+n] n=0,1..5	CAPFIENn	<b>EPWM 下降沿捕获锁存中断使能</b> 0 = 禁止下降沿锁存中断 1 = 使能下降沿锁存中断
[7:6]	Reserved	保留.
[n] n=0,1..5	CAPRIENn	<b>EPWM上升沿捕获锁存中断使能</b> 0 = 禁止上升沿锁存中断 1 = 使能上升沿锁存中断

EPWM\_CAPIF EPWM捕获中断标志寄存器

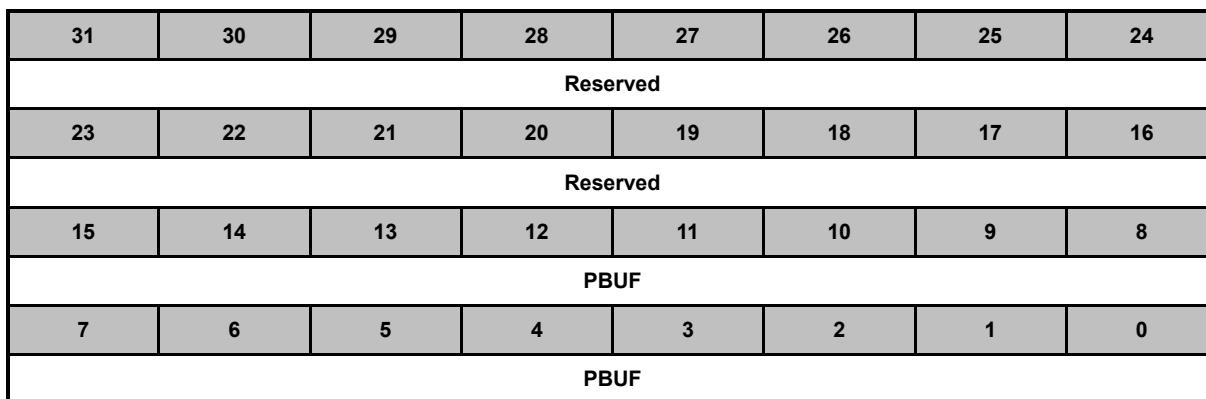
寄存器	偏移量	R/W	描述	复位值
EPWM_CAPIF	EPWMx_BA+0x254	R/W	EPWM 捕获中断标志寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved		CFLIF5	CFLIF4	CFLIF3	CFLIF2	CFLIF1	CFLIF0
7	6	5	4	3	2	1	0
Reserved		CRLIF5	CRLIF4	CRLIF3	CRLIF2	CRLIF1	CRLIF0

位	描述	
[31:14]	Reserved	保留.
[8+n] n=0,1..5	CFLIFn	<p><b>EPWM下降沿捕获锁存中断标志</b>            该位写1清0。            0 =没有下降沿捕获锁存条件发生。            1 =下降沿捕获锁存条件发生，该标志被置高。</p> <p><b>注意：</b>当带PDMA运行捕获功能， PDMA 搬移完数据后， CAPIF对应通道的CFLIF位将被硬件清除。</p>
[7:6]	Reserved	保留.
[n] n=0,1..5	CRLIFn	<p><b>EPWM上升沿捕获锁存中断标志</b>            该位写1清0。            0 =没有上升沿捕获锁存条件发生。            1 =上升沿捕获锁存条件发生，该标志被置高。</p> <p><b>注意：</b>当带PDMA运行捕获功能， PDMA 搬移完数据后， CAPIF对应通道的CFLIF位将被硬件清除。</p>

EPWM\_PBUF0~5 EPWM 周期寄存器缓存 0~5

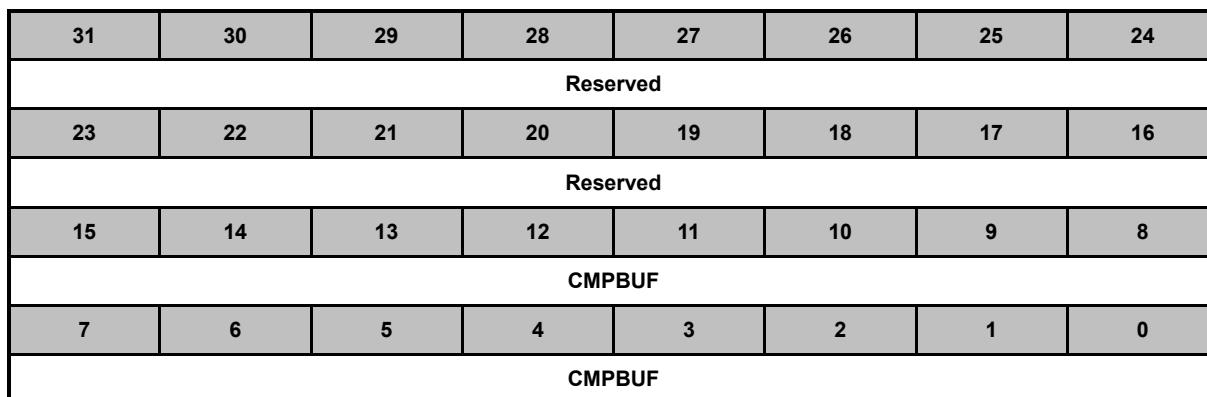
寄存器	偏移量	R/W	描述	复位值
EPWM_PBUF0	EPWMx_BA+0x304	R	EPWM PERIOD0 缓存	0x0000_0000
EPWM_PBUF1	EPWMx_BA+0x308	R	EPWM PERIOD1缓存	0x0000_0000
EPWM_PBUF2	EPWMx_BA+0x30C	R	EPWM PERIOD2缓存	0x0000_0000
EPWM_PBUF3	EPWMx_BA+0x310	R	EPWM PERIOD3缓存	0x0000_0000
EPWM_PBUF4	EPWMx_BA+0x314	R	EPWM PERIOD4缓存	0x0000_0000
EPWM_PBUF5	EPWMx_BA+0x318	R	EPWM PERIOD5缓存	0x0000_0000



位	描述	
[31:16]	Reserved	保留.
[15:0]	PBUF	EPWM周期寄存器缓存（只读） 用作有效周期寄存器.

EPWM\_CMPBUF0~5 EPWM 比较寄存器缓存0~5

寄存器	偏移量	R/W	描述	复位值
EPWM_CMPBUF0	EPWMx_BA+0x31C	R	EPWM CMPDAT0 缓存	0x0000_0000
EPWM_CMPBUF1	EPWMx_BA+0x320	R	EPWM CMPDAT1 缓存	0x0000_0000
EPWM_CMPBUF2	EPWMx_BA+0x324	R	EPWM CMPDAT2 缓存	0x0000_0000
EPWM_CMPBUF3	EPWMx_BA+0x328	R	EPWM CMPDAT3 缓存	0x0000_0000
EPWM_CMPBUF4	EPWMx_BA+0x32C	R	EPWM CMPDAT4 缓存	0x0000_0000
EPWM_CMPBUF5	EPWMx_BA+0x330	R	EPWM CMPDAT5 缓存	0x0000_0000



位	描述	
[31:16]	Reserved	保留.
[15:0]	CMPBUF	EPWM比较寄存器缓存（只读） 用作有效CMP寄存器

EPWM CPSCBUF0\_1, 2, 3, 4, 5 EPWM CLKPSC 缓存 0, 1, 2, 3, 4, 5

寄存器	偏移量	R/W	描述	复位值
EPWM_CpscBuf0_1	EPWMx_BA+0x334	R	EPWM CLKPSC0_1 缓存	0x0000_0000
EPWM_CpscBuf2_3	EPWMx_BA+0x338	R	EPWM CLKPSC2_3 缓存	0x0000_0000
EPWM_CpscBuf4_5	EPWMx_BA+0x33C	R	EPWM CLKPSC4_5 缓存	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved				CPSCBUF			
7	6	5	4	3	2	1	0
CPSCBUF							

位	描述	
[31:12]	Reserved	保留.
[11:0]	CPSCBUF	EPWM 计数器时钟预分频缓存 用作 EPWM 计数器时钟预分频有效寄存器

EPWM\_FTCBUF0\_1,2,3,4,5 EPWM\_FTCMPDAT 缓存

寄存器	偏移量	R/W	描述	复位值
EPWM_FTCB UF0_1	EPWMx_BA+0x340	R	EPWM FTCMPDAT0_1 缓存	0x0000_0000
EPWM_FTCB UF2_3	EPWMx_BA+0x344	R	EPWM FTCMPDAT2_3 缓存	0x0000_0000
EPWM_FTCB UF4_5	EPWMx_BA+0x348	R	EPWM FTCMPDAT4_5 缓存	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
FTCMPBUF							
7	6	5	4	3	2	1	0
FTCMPBUF							

位	描述	
[31:16]	Reserved	保留.
[15:0]	FTCMPBUF	EPWM FTCMPDAT 缓存 (只读) 用作有效FTCMPDAT 寄存器.

EPWM\_FTCI EPWM FTCMPDAT 指示寄存器

寄存器	偏移量	R/W	描述	复位值
EPWM_FTCI	EPWMx_BA+0x34C	R/W	EPWM FTCMPDAT 指示寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved					FTCMD4	FTCMD2	FTCMD0
7	6	5	4	3	2	1	0
Reserved					FTCMU4	FTCMU2	FTCMU0

位	描述	
[31:11]	Reserved	保留.
[8+n/2] n=0,2,4	FTCMDn	<b>EPWM FTCMPDAT 向下指示</b> 当 EPWM 计数器下数到 EPWM_FTCMPDATn 时该指示位被硬件置位。软件可以写 1 来清零。
[7:3]	Reserved	保留.
[n/2] n=0,2,4	FTCMUn	<b>EPWM FTCMPDAT 向上指示</b> 当 EPWM 计数器上数到 EPWM_FTCMPDATn 时该指示位被硬件置位。软件可以写 1 来清零。

## 6.12 BPWM 基本 PWM 发生器和捕获定时器

### 6.12.1 概述

如图 6.12-1 所示，M480 有 2 组 BPWM 发生器—BPWM0 和 BPWM1。

每组 BPWM 都支持 6 通道的 BPWM 输出和捕捉输入。12 位的预分频器，16 位比较器，6 路通道共享一个 16 位的 BPWM 计数器。BPWM 计数器支持向上、向下和上下的计数方式。BPWM 通过比较比较器和计数器的值来产生事件。这些事件可以用来产生 BPWM 脉冲、中断和触发 EADC 开始转换的信号。BPWM 的输出控制单元支持极性输出、独立的管脚屏蔽和三态输出使能。

BPWM 发生器还支持输入捕捉功能，在输入通道有上升沿、下降沿或双边沿信号发生时锁存 BPWM 计数器的值到对应的寄存器。

### 6.12.2 特性

#### 6.12.2.1 BPWM 功能特性

- 时钟频率最高可达到最大的 PLL 频率
- 2 组 BPWM 模块，每组都提供 6 路输出通道
- BPWM 输出/捕捉通道支持独立模式
- 从 1 到 4096 的 12 位预分频器
- 16 位精度的 BPWM 计数器，每个模块都有 1 个 BPWM 计数器
  - 向上、向下及上下/的计数器操作模式
- 每个 BPWM 管脚都支持屏蔽功能和三态使能
- 以下事件可以触发中断：
  - BPWM 计数器到 0，或周期值，或比较器的值
- 以下事件可以触发 EADC：
  - BPWM 计数器到 0，或周期值，或比较器的值

#### 6.12.2.2 捕捉功能特性

- 多达 12 路 16 位精度的输入捕捉通道
- 支持向上或向下的捕捉条件
- 支持向上或向下的捕捉中断
- 支持向上或向下的捕捉触发计数器重

## 6.12.3 框图

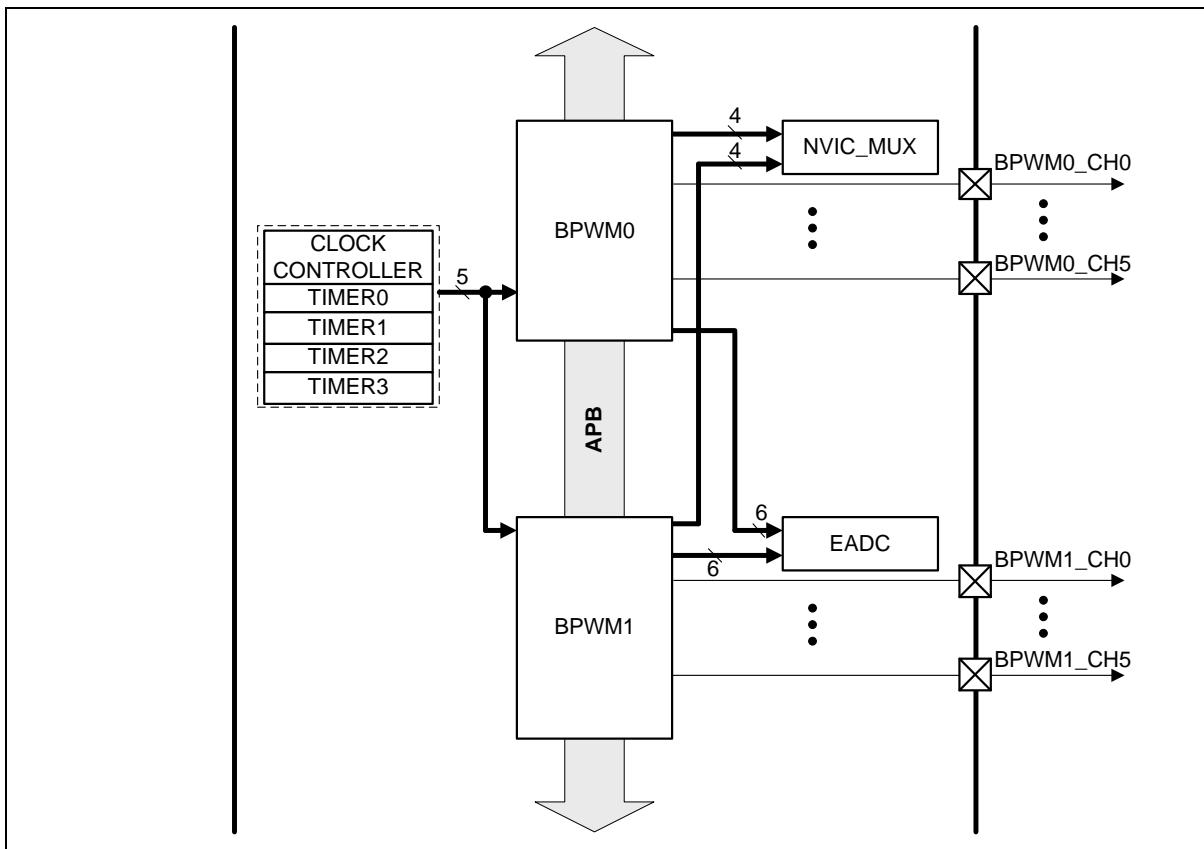


图 6.12-1 BPWM 发生器框图

如图 6.12-2 所示，每个 BPWM 发生器只有一个时钟源输入，通过 BPWM\_CLK0 的 ECLKSRC0 (BPWM\_CLKSRC[2:0]) 寄存器决定由 BPWM 时钟或者定时器触发 BPWM 输出。通常，BPWM0 一定要设置成 BPWM0SEL (CLK\_CLKSEL2[8]) 为 1 选则 PCLK0，BPWM1 一定要设置成 BPWM1SEL (CLK\_CLKSEL2[9]) 为 1 选则 PCLK1。

如图 6.12-3 和表 6.12-1 所示，当运行在最大 PLL 频率时，BPWM0 一定要设置成 BPWM0SEL (CLK\_CLKSEL2[8]) 为 0，BPWM1 一定要设置成 BPWM1SEL (CLK\_CLKSEL2[9]) 为 0。

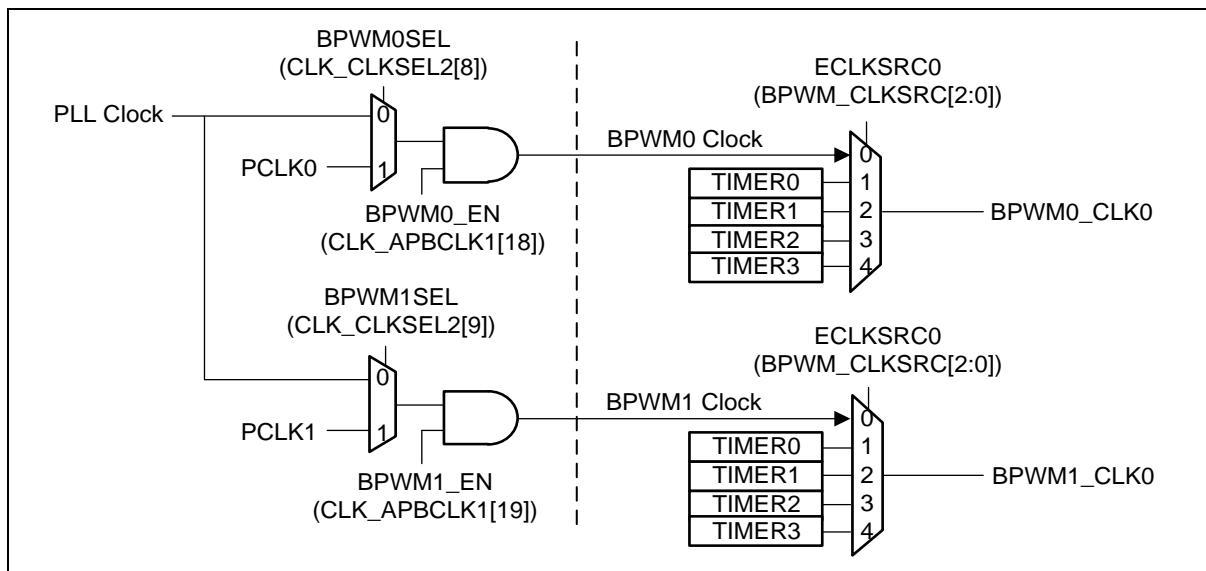


图 6.12-2 BPWM 时钟源控制

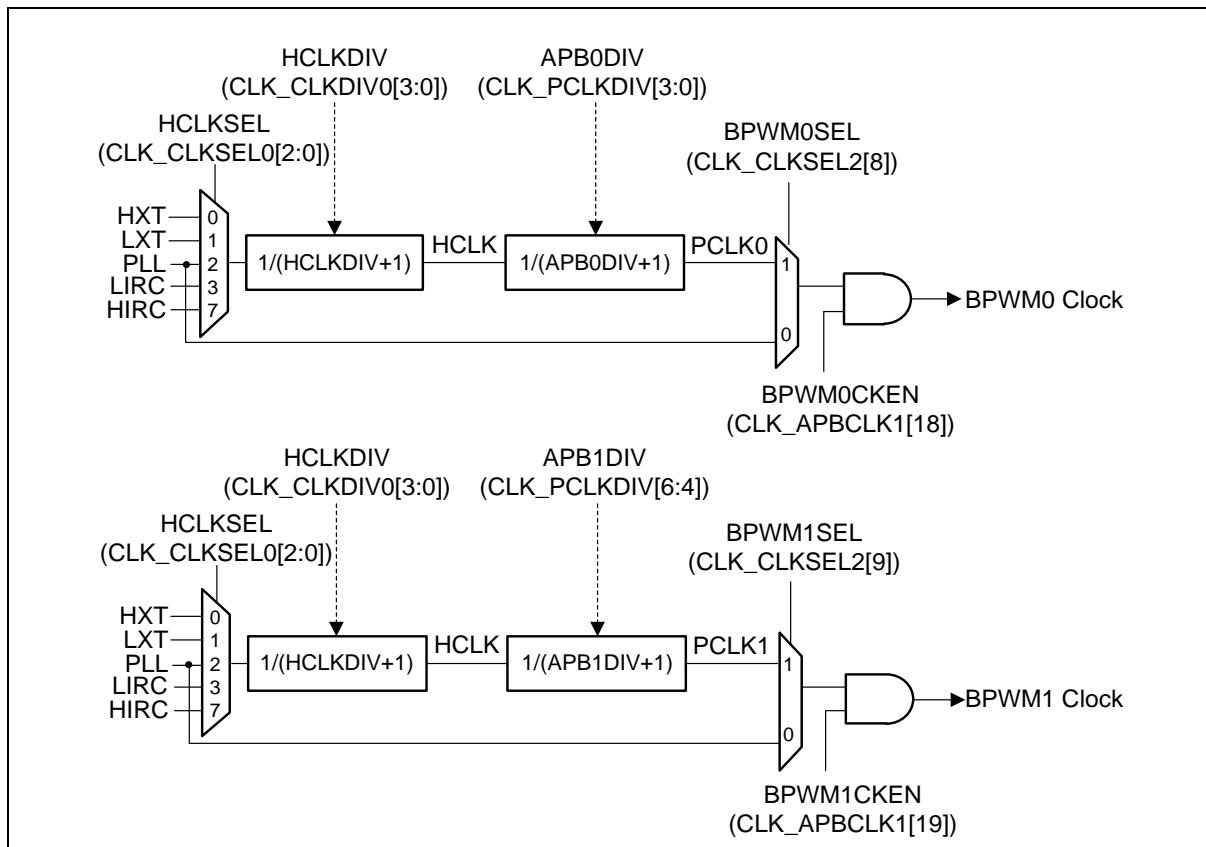


图 6.12-3 BPWM 时钟源控制

PCLK:BPWM 时钟 频率比	HCLK	PCLK	BPWM 时钟	HCLKSEL CLK_CLKSEL0[2:0]	HCLKDIV CLK_CLKDIV0[3:0]	APBnDIV (CLK_CLKDIVn [2+4n:4n]), N 为 0 或 1	BPWMnSEL (CLK_CLKSEL2[ N+8]), N 为 0 或 1
1:1	HCLK	PCLK	PCLK	无影响	无影响	无影响	1
1:2	PLL	PLL/2	PLL	2	0	1	0
1:2	PLL/2	PLL/2	PLL	2	1	0	0

表 6.12-1 BPWM 时钟源控制寄存器设置

如图 6.12-4 所示是 BPWM 独立模式的架构。所有的 6 路通道共享一个计数器。当计数器计数到 0, PERIOD (BPWM\_PERIOD[15:0]) 或者到比较器的值, 将会产生事件。事件被传到对应的发生器来产生 BPWM 脉冲, 中断信号或者触发 EADC 开始转换的信号。输出控制则用来改变 BPWM 脉冲的输出状态。

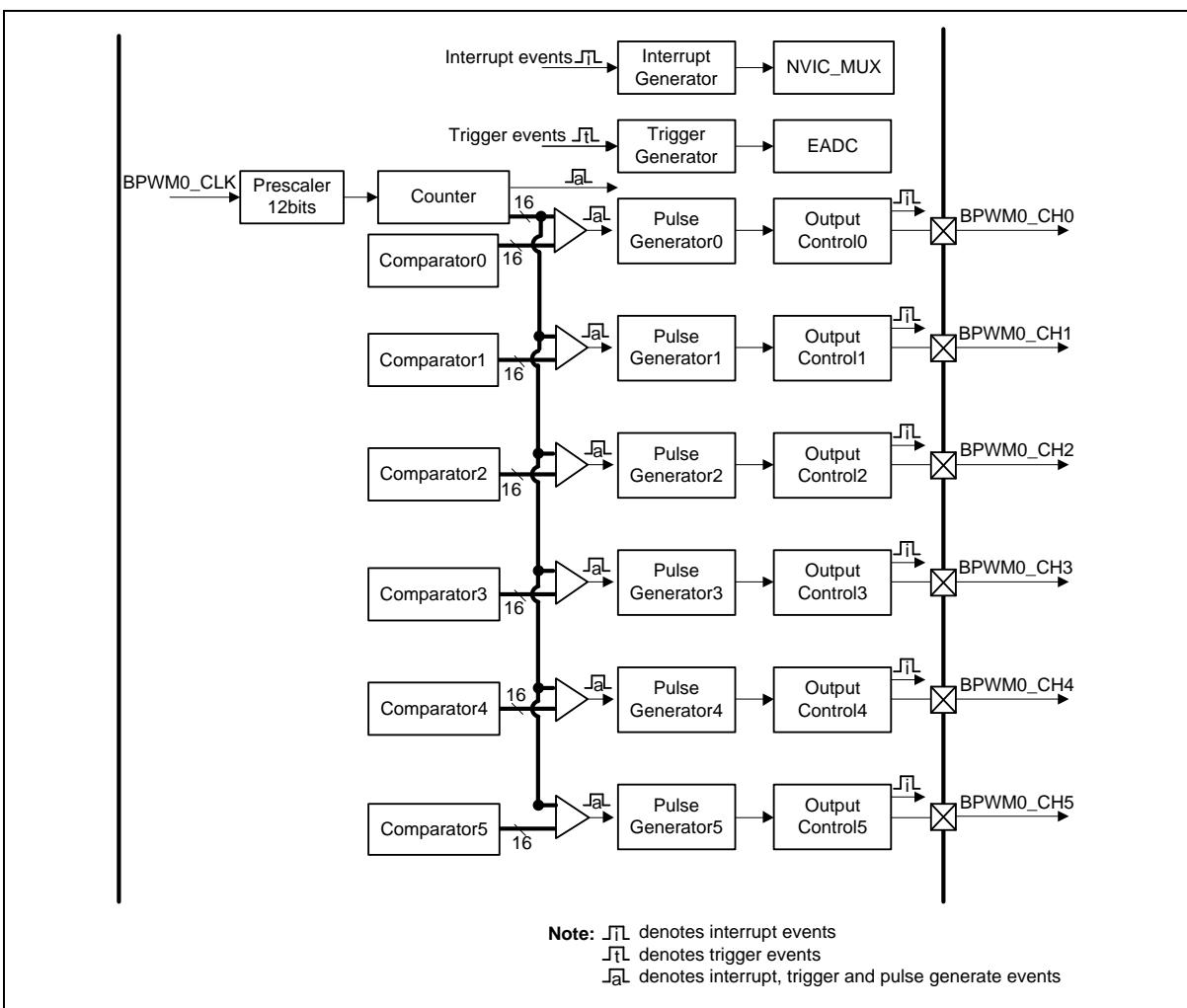


图 6.12-4 BPWM 独立模式框架图

### 6.12.4 基本配置

#### 6.12.4.1 BPWM0 基本配置

- 时钟源配置
  - 在寄存器 BPWM0SEL (CLK\_CLKSEL2[8]) 选择 BPWM0 的时钟源.
  - 在寄存器 BPWM0CKEN (CLK\_APBCLK1[18]) 使能 BPWM0 外设时钟.
- 复位配置
  - 在寄存器 BPWM0RST (SYS\_IPRST2[18]) 复位 BPWM0 控制器
- 管脚配置

Group	Pin Name	GPIO	MFP
BPWM0	BPWM0_CH0	PA.11	MFP9
		PA.0, PG.14	MFP12
		PE.2	MFP13
	BPWM0_CH1	PA.10	MFP9
		PA.1, PG.13	MFP12
		PE.3	MFP13
	BPWM0_CH2	PA.9	MFP9
		PA.2, PG.12	MFP12
		PE.4	MFP13
	BPWM0_CH3	PA.8	MFP9
		PA.3, PG.11	MFP12
		PE.5	MFP13
	BPWM0_CH4	PF.5	MFP8
		PC.13	MFP9
		PA.4, PG.10	MFP12
		PE.6	MFP13
	BPWM0_CH5	PF.4	MFP8
		PD.12	MFP9
		PA.5, PG.9	MFP12
		PE.7	MFP13

#### 6.12.4.2 BPWM1 基本配置

- 时钟源配置
  - 在寄存器 BPWM1SEL (CLK\_CLKSEL2[9]) 选择 BPWM1 的时钟源.
  - 在寄存器 BPWM1CKEN (CLK\_APBCLK1[19]) 使能 BPWM1 的外设时钟.
- 复位配置
  - 在寄存器 BPWM1RST (SYS\_IPRST2[19]) 复位 BPWM1 控制器.

- 管脚配置

Group	Pin Name	GPIO	MFP
BPWM1	BPWM1_CH0	PB.11	MFP10
		PF.3	MFP11
		PC.7, PF.0	MFP12
	BPWM1_CH1	PB.10	MFP10
		PF.2	MFP11
		PC.6, PF.1	MFP12
	BPWM1_CH2	PB.9	MFP10
		PA.12	MFP11
		PA.7	MFP12
	BPWM1_CH3	PB.8	MFP10
		PA.13	MFP11
		PA.6	MFP12
	BPWM1_CH4	PB.7	MFP10
		PA.14	MFP11
		PC.8	MFP12
	BPWM1_CH5	PB.6	MFP10
		PA.15	MFP11
		PE.13	MFP12

### 6.12.5 功能描述

#### 6.12.5.1 BPWM 预分频器

BPWM 预分频器用来分频时钟源，预分频器计数 CLKPSC+1 次，而 BPWM 计数器则只计数 1 次。预分频的值是由 CLKPSC (BPWM\_CLKPSC[11:0]) 设置。如图 6.12-5 是 BPWM 通道 0 CLKPSC 波形示例图。预分频计数器会在下一个预分频向下计数的开始时重载 CLKPSC 的值。

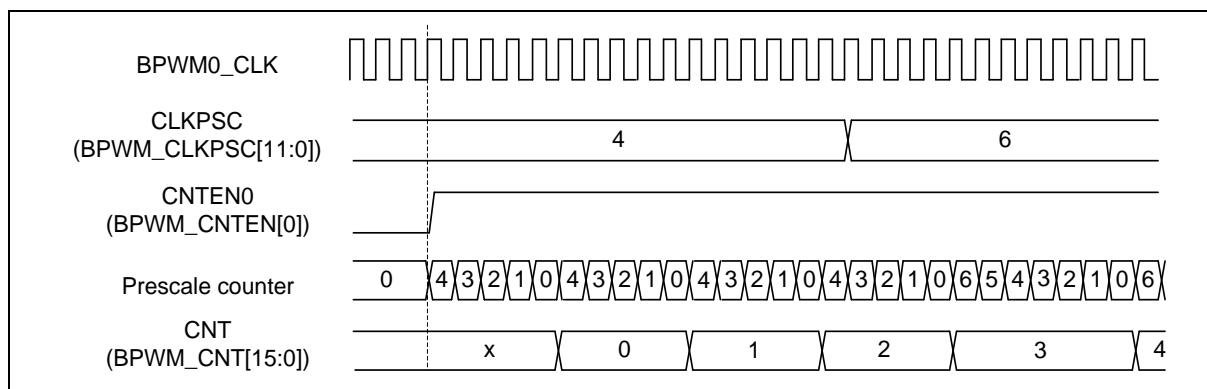


图 6.12-5 BPWM\_CH0 CLKPSC 波形

### 6.12.5.2 BPWM 计数器

BPWM 的计数器支持 3 种计数操作方式：向上，向下和上下计数。

对于 BPWM 通道 0，可以设置当前分频计数器向下计数到 0 时，CNTCLR0 (BPWM\_CNTCLR[0]) 会清空 CNT (BPWM\_CNT[15:0]) 到 0x00

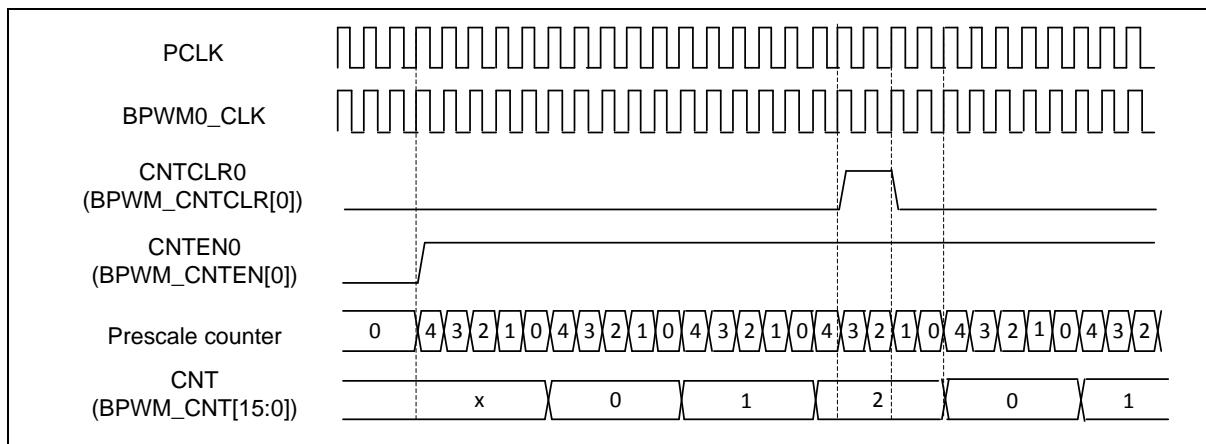


图 6.12-6 BPWM 计数器清空波形

### 6.12.5.3 向上计数方式

在向上计数操作中，16 位的 BPWM 计数器从 0 开始向上计数到 PERIOD (BPWM\_PERIOD) 完成一个 BPWM 周期。从 CNT (BPWM\_CNT[15:0]) 寄存器可以读到当前的计数值。当计数器计数到 0 时，BPWM 发生器会产生零位事件；计数器计数到 PERIOD 时则会产生周期位置事件。如图 6.12-7 所示是向上计数方式下的周期时间示例，BPWM 的周期时间 = (PERIOD+1) \* (CLKPSC+1) \* BPWMx\_CLK.

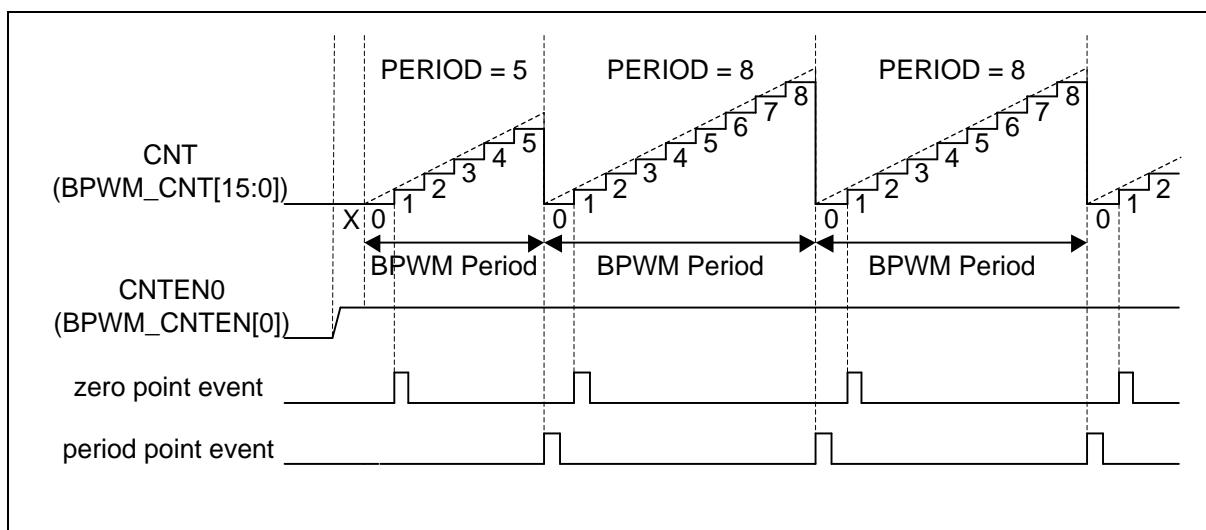


图 6.12-7 BPWM Up 向上计数方式

### 6.12.5.4 向下计数方式

在向下计数操作中，16 位的 BPWM 计数器从 PERIOD 开始向下计数到 0 完成一个 BPWM 周期。从 CNT (BPWM\_CNT[15:0]) 寄存器可以读到当前的计数值。当计数器计数到 0 时，BPWM 发生器

会产生零位事件；计数器计数到 PERIOD 时则会产生中位事件。如图 6.12-8 所示是向下计数方式下的周期时间示例，BPWM 的周期时间 = (PERIOD+1) \* (CLKPSC+1) \* BPWMx\_CLK.

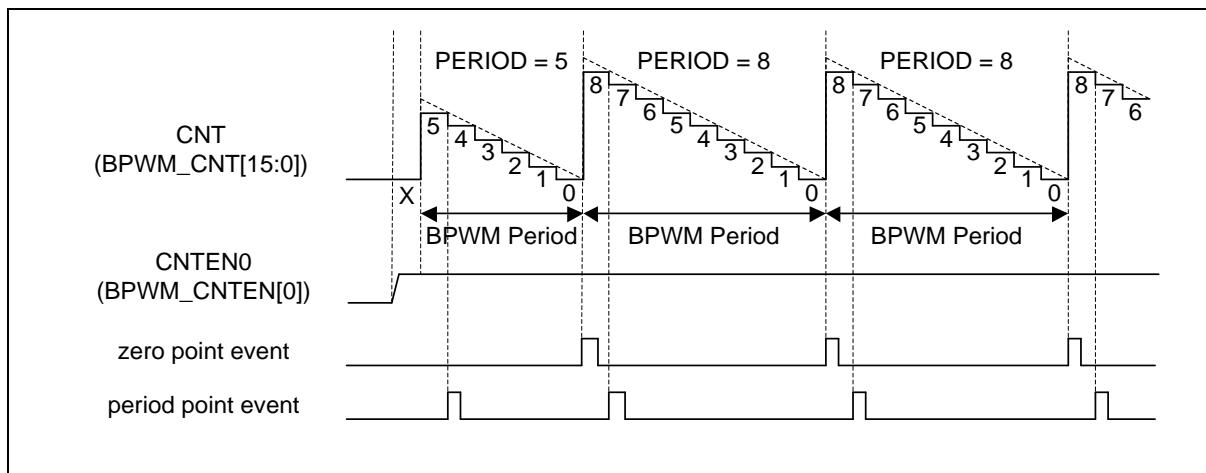


图 6.12-8 BPWM 向下计数方式

#### 6.12.5.5 上下计数方式

在上下计数操作中，16 位的 BPWM 计数器先从 0 开始向上计数到 PERIOD，再从 PERIOD 开始向下计数到 0 完成一个 BPWM 周期。从 CNT (BPWM\_CNT[15:0]) 寄存器可以读到当前的计数值。当计数器计数到 0 时，BPWM 发生器会产生零位事件；计数器计数到 PERIOD 时则会产生中位事件。如图 6.12-9 所示是上下计数方式下的周期时间示例，BPWM 的周期时间 = (2\*PERIOD) \* (CLKPSC+1) \* BPWMx\_CLK. 寄存器 DIRF (BPWM\_CNT[16]) 是计数方向的标志位，向上计数时为 1，向下计数时为 0。

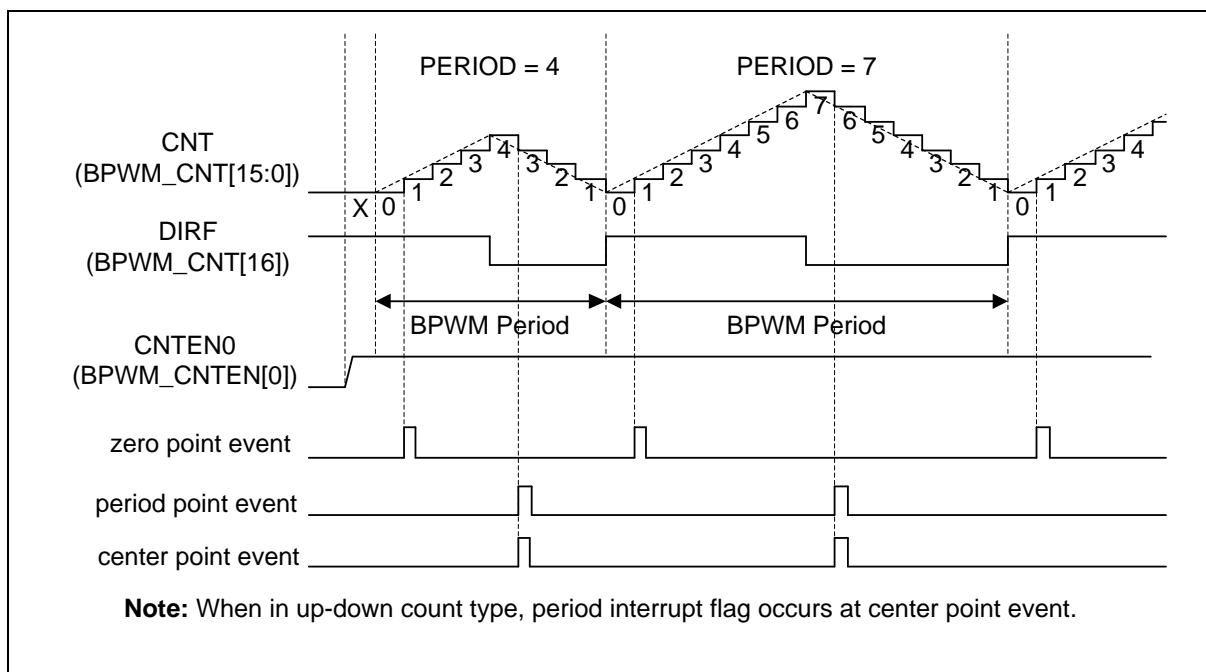


图 6.12-9 BPWM 上下计数方式

#### 6.12.5.6 BPWM 比较器

CMPDAT (BPWM\_CMPDATn[15:0]) 是 BPWM 通道 n 的基本比较器寄存器，每路通道都有且仅有

一个 CMPDAT。CMPDAT 的值不断与计数器的值做比较。当计数器计数到比较器寄存器的值时，BPWM 会产生一个事件并用以之产生 BPWM 脉冲，中断或者 EADC 触发信号。如图 6.12-10 所示，在上下计数方式里会有 2 个事件产生。

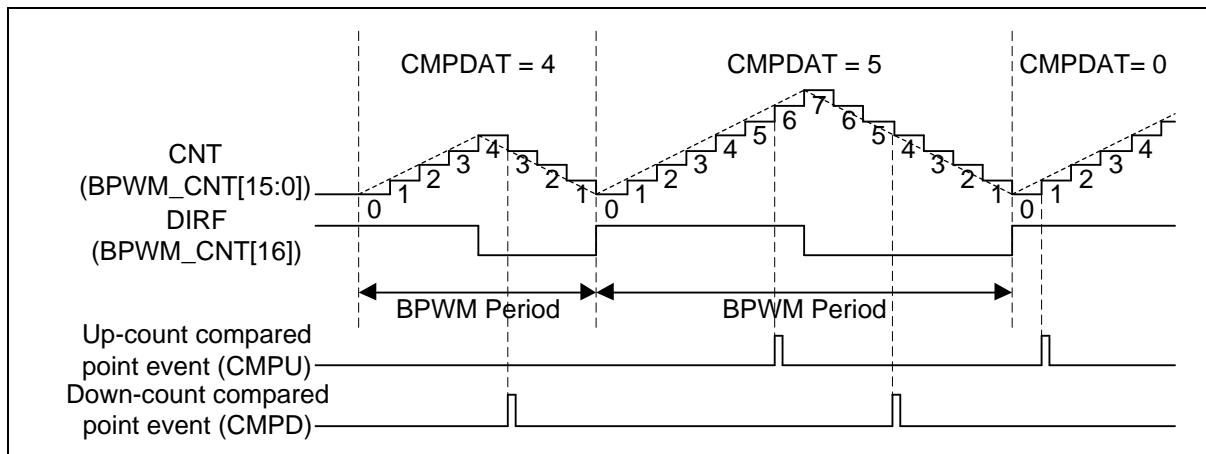


图 6.12-10 上下计数方式时 BPWM CMPDAT 事件

#### 6.12.5.7 周期载入模式

周期载入模式是默认的载入模式。在所有载入模式里它是优先级最低的。当周期结束时，PERIOD 和 CMPDAT 都会被载入到对应的缓存。例如，在向上计数操作里 BPWM 从 0 向上计数到 PERIOD 后，或者向下计数操作里从 PERIOD 向下计数到 0 后，亦或上下计数操作里从 0 计数到 PERIOD 再向下计数到 0 后，都会进行重载。

如图 6.12-11 所示是向上计数方式的重载时机，图示里 PERIOD DATA0 为 PERIOD 的初始值，PERIOD DATA1 为 PERIOD 第一次由软件更新的值，以此类推，CMPDAT 也按这种方式标注。图 6.12-11 的具体步骤顺序参见下面的描述。用户可以查看 BPWM 的周期和 CMU 事件来了解 PERIOD 和 CMPDAT 的更新条件。

1. 在位置 1，软件写 CMPDAT DATA1 到 CMPDAT
2. 在位置 2，PWM 周期结束时，硬件载入 CMPDAT DATA1 到 CMPBUF
3. 在位置 3，软件写 PERIOD DATA1 到 PERIOD
4. 在位置 4，PWM 周期结束时，硬件载入 PERIOD DATA1 到 PBUF
5. 在位置 5，软件写 PERIOD DATA2 到 PERIOD
6. 在位置 5，PWM 周期结束时，硬件载入 PERIOD DATA2 到 PBUF.

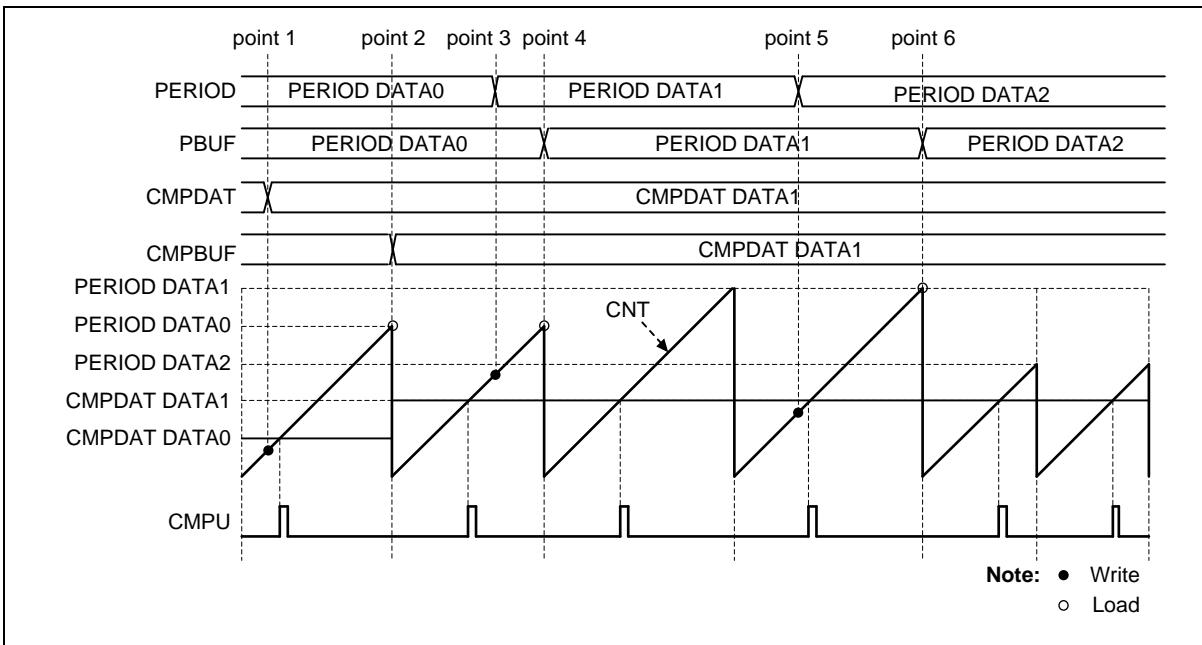


图 6.12-11 向上计数方式下的周期载入模式

#### 6.12.5.8 立即载入模式

如果 BPWM 通道 n 在寄存器 IMMLDENn (BPWM\_CTL0[21:16]) 里对应的位置 1 时，当软件更新 PERIOD 或者 CMPDAT 时，硬件会立刻载入 PERIOD 和 CMPDAT 到缓存。如果更新的 PERIOD 值小于当前的计数器值，计数器会一直计数到 0xFFFF，当计数器计数到 0xFFFF 且预分频值计数到 0，寄存器 CNTMAX0 (BPWM\_STATUS[0]) 置位，接着计数器回环计数。立即载入模式拥有最高的优先级。如果 IMMLDENn 被置位，其他针对通道 n 的模式设置都会无效。下面是对图 6.12-12示例的文字说明：

1. 在位置 1 软件写 CMPDAT DATA1，硬件立即重载 CMPDAT DATA1 到 CMPBUF
2. 在位置 2 软件写入一个比当前计数器大的值 PERIOD DATA1，计数器会一直计数到 PERIOD DATA1 的值以完成一个载入周期。
3. 在位置 3 软件写入一个比当前计数器小的值 PERIOD DATA2，计数器会一直计数到最大值 0xFFFF，再从 0 开始回环计数到 PERIOD DATA2 以完成周期载入。

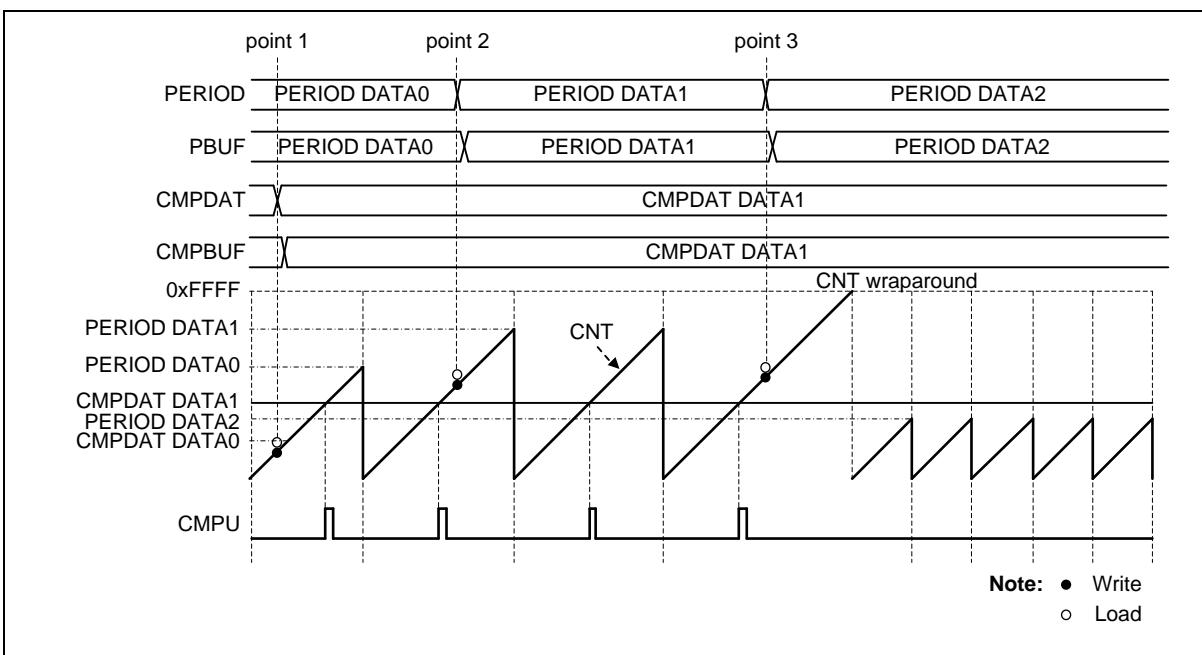


图 6.12-12 向上计数方式下的立即载入模式

#### 6.12.5.9 中心载入模式

如果 BPWM 通道 n 在寄存器 `CTRLDn` (`BPWM_CTL0[5:0]`) 里对应的位置 1 且处于上下计数方式时, `CMPDAT` 的值会在周期中间被载入到 `CMPBUFn`, 即在计数器计数到 `PERIOD` 时载入。`PERIOD` 的载入时间同周期载入模式相同。下面是对图 6.12-13示例的文字说明:

1. 在位置 1 软件写 `CMPDAT DATA1`
2. 在位置 2 PWM 周期中间位置, 硬件载入 `CMPDAT DATA1` 到 `CMPBUFn`
3. 在位置 3 软件写 `PERIOD DATA1`
4. 在位置 4 PWM 周期末位置, 硬件载入 `PERIOD DATA1` 到 `PBUF`
5. 在位置 5 软件写 `CMPDAT DATA2`
6. 在位置 6 PWM 周期中间位置, 硬件载入 `CMPDAT DATA2` 到 `CMPBUFn`
7. 在位置 7 软件写 `PERIOD DATA2`
8. 在位置 8 PWM 周期末位置, 硬件写 `PERIOD DATA2` 到 `PBUF`

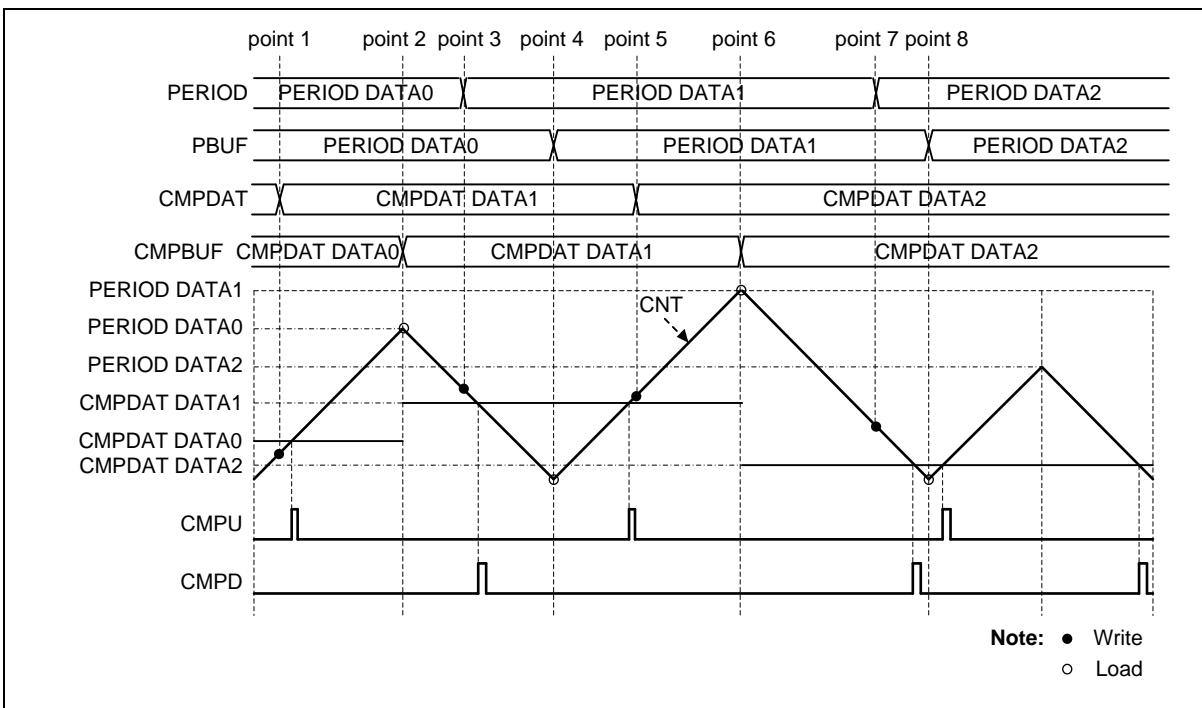


图 6.12-13 上下计数方式下的中间载入模式

#### 6.12.5.10 BPWM 脉冲发生器

BPWM 脉冲发生器使用计数器和比较器事件来产生 BPWM 脉冲。对应的事件有：零位、向上计数器方式和向下计数器方式的周期位置、上下计数方式的中心位置以及三种模式下计数器计数到比较器值的位置。对于上下计数方式，有两个计数器计数到比较器值的位置，一是向上计数的过程，二是向下计数的过程。

通过 BPWM\_WGCTL0 和 BPWM\_WGCTL1 寄存器可以设定每个位置点 BPWM 波形：无动作 (X)、拉低 (L)、拉高 (H) 或者触发 (T)。如图 6.12-14，通过这些位置点可以轻易的产生不对称的 BPWM 脉冲及多样的波形。图中，比较器 n 用于产生 BPWM 脉冲，n 指通道 0 ~ 5；CMPU 指向上计数时 CNT 计数到 CMPDAT，CMRD 指向下计数时 CNT 计数到 CMRDAT。

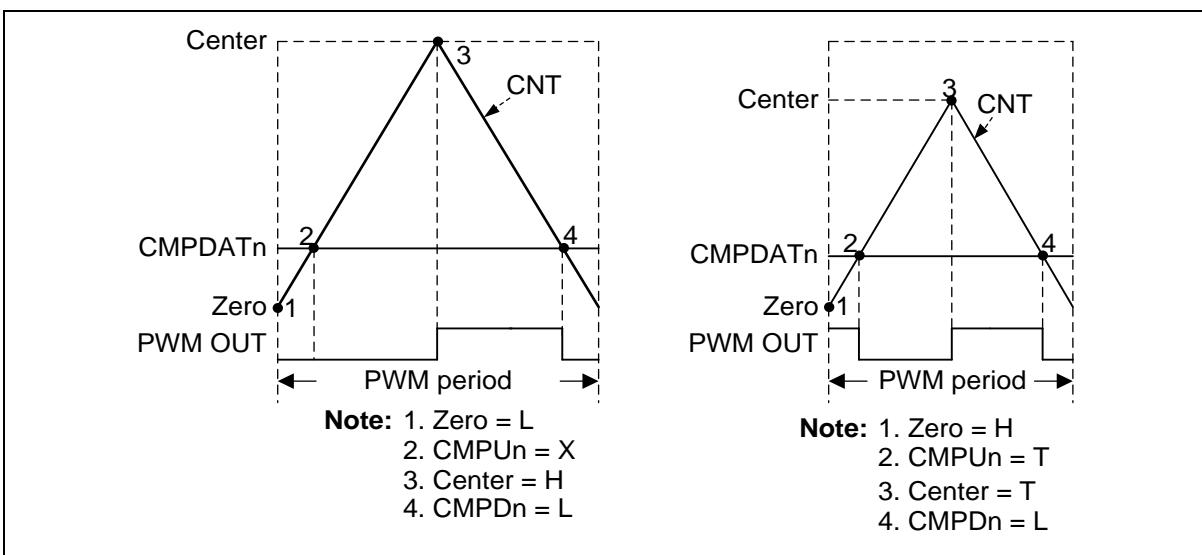


图 6.12-14 BPWM 脉冲发生 (左: 不对称脉冲, 右: 多样脉冲)

多个事件有时候会发生在同一时刻。这时候，不同的计数方式的事件的优先级如下所示：表 6.12-2是向上计数方式的优先级、表 6.12-3是向下计数方式的优先级而表 6.12-4是上下计数方式的优先级。如图 6.12-15 所示，通过事件优先级，用户可以轻易产生从 0% 到 100% 占空比的脉冲。

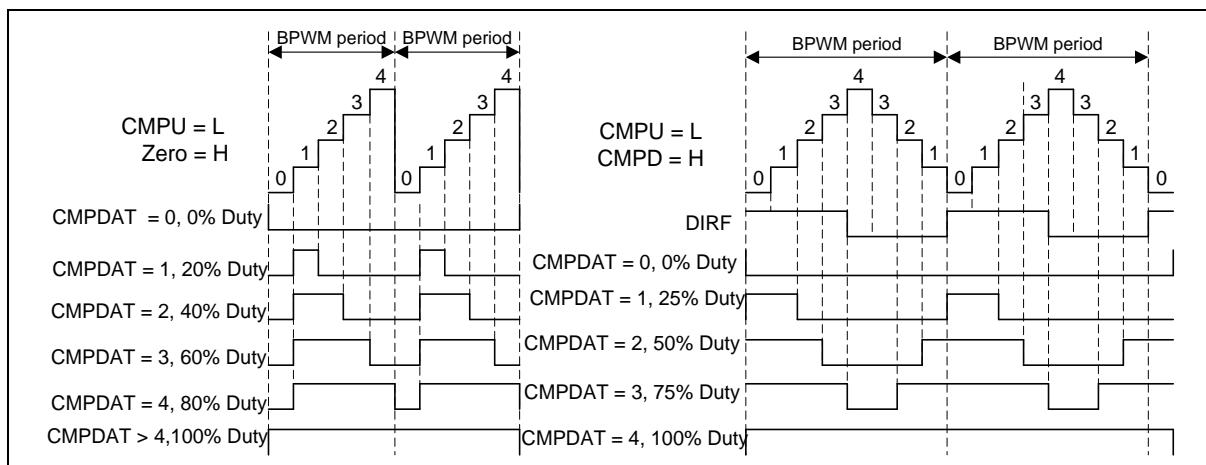


图 6.12-15 BPWM 0% 到 100% 脉冲产生 (左: 上计数器模式, 右: 上下计数器模式)

优先级	向上计数事件
1 (最高)	周期事件 (CNT = PERIOD)
2	比较事件 (CNT = CMPUn)
3 (最低)	零位事件 (CNT = 0)

表 6.12-2 向上计数方式下的 BPWM 脉冲发生事件优先级

优先级	向下计数方式
1 (最高)	零位事件 (CNT = 0)
2	比较事件 (CNT = CMPDn)
3 (最低)	周期事件 (CNT = PERIOD)

表 6.12-3 向下计数方式的 BPWM 脉冲发生事件优先级

优先级	向上计数	向下计数
1 (最高)	向上比较事件 (CNT = CMPUn)	向下计数事件 (CNT = CMPDn)
2	零位事件 (CNT = 0)	周期中间事件 (CNT = PERIOD)
3 (最低)	向下比较事件 (CNT = CMPDn)	向上比较事件 (CNT = CMPUn)

表 6.12-4 上下计数方式的 BPWM 脉冲发生事件优先级

### 6.12.5.11 同步功能

同时使能 BPWM 和 PWM 计数器需要：设定 BPWM 同步开始控制寄存器 (BPWM\_SSCTL[0]) 以使能

希望开始同步计数的计数器通道，设定 寄存器 SSRC (BPWM\_SSCTL[9:8]) 来选择同步开始源，然后再设定 BPWM 同步开始触发寄存器 CNTSEN (BPWM\_SSTRG[0])。

#### 6.12.5.12 BPWM 输出控制

BPWM 脉冲发生后，可以分三步来控制 BPWM 通道输出。这三个步骤如图 6.12-16 分别是：屏蔽，管脚极性和输出使能。

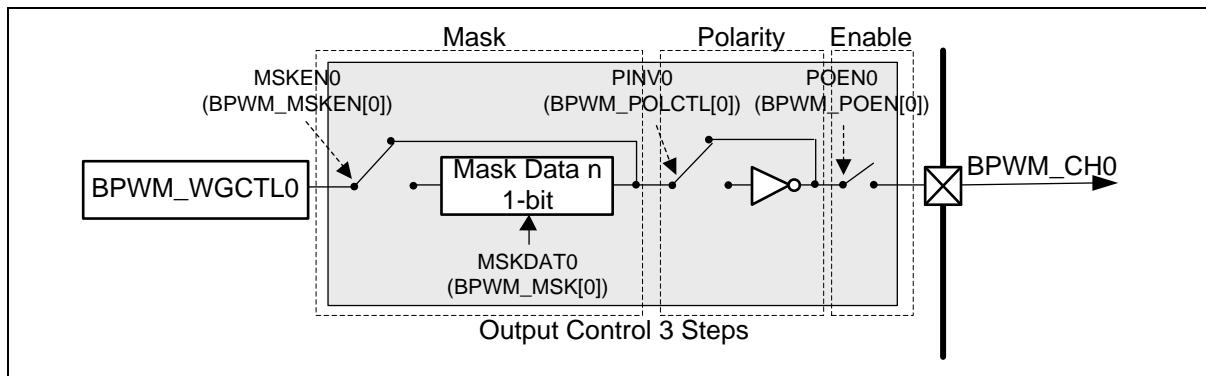


图 6.12-16 BPWM\_CH0 输出控制的 3 步

#### 6.12.5.13 BPWM 屏蔽输出功能

每个 BPWM 的输出通道都可以通过人为地修改 BPWM 屏蔽使能寄存器 (BPWM\_MSKEN) 和 BPWM 屏蔽数据寄存器 (BPWM\_MSK) 对应的位，来输出独立于占空比周期比较单元的特定逻辑状态。在驱动类似 BLDC 电机的电力整流电机时，BPWM 的屏蔽位会非常有用。BPWM\_MSKEN 寄存器包含 6 位：MSKENN (BPWM\_MSKEN[5:0]) 控制哪路 BPWM 通道输出会被修改，这 6 位是高有效。BPWM\_MSK 寄存器包含 6 位，MSKDATn (BPWM\_MSK[5:0]) 控制被 MSKDAT 对应位屏蔽的 BPWM 通道输出状态。如图 6.12-17 所示是通过 BPWM 屏蔽功能修改输出的例子。

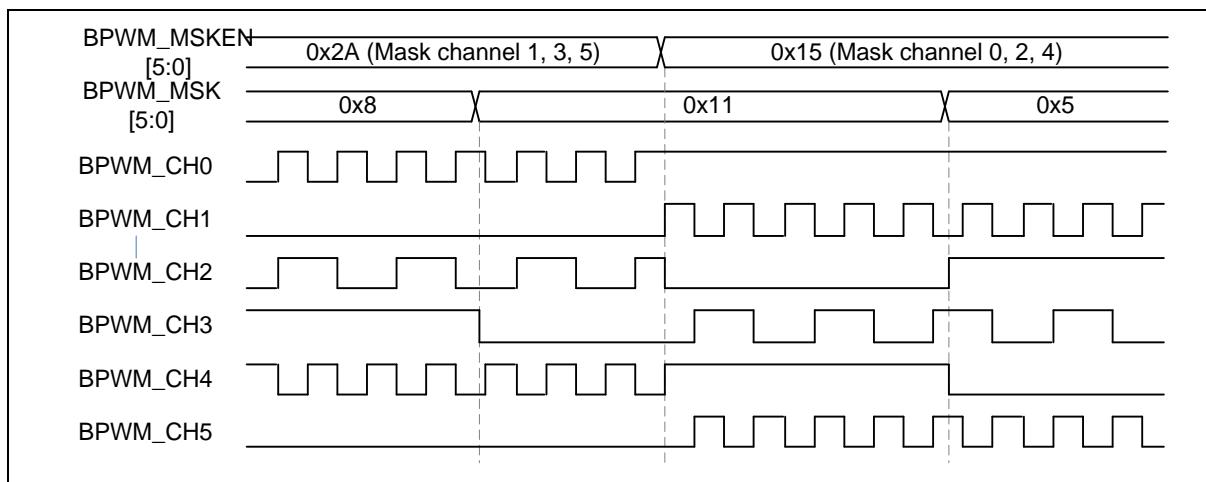


图 6.12-17 屏蔽功能波形说明

#### 6.12.5.14 极性控制

从 BPWM\_CH0 到 BPWM\_CH5 每个 BPWM 端口都有独立的极性控制模块来配置 BPWM 输出的有效状态的极性。BPWM 默认输出高态有效。也就是说，BPWM 关闭状态为低，打开状态为高。每个独立的 BPWM 通道都可以通过配置 BPWM 阴极极性控制寄存器 (BPWM\_POLCTL) 来自由配置有效状态。如图 6.12-18 是应用不同极性配置前 BPWM 的初始状态。

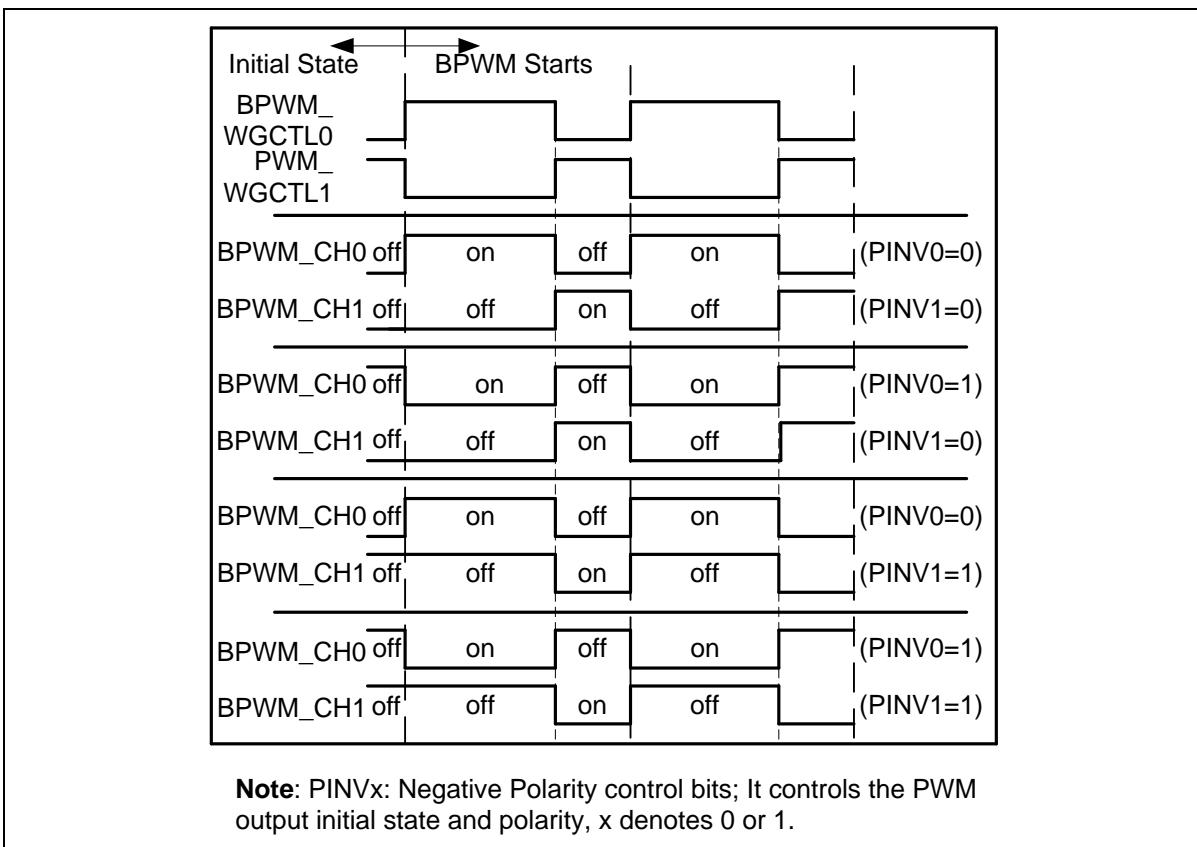


图 6.12-18 初始状态和极性控制

#### 6.12.5.15 BPWM 中断发生器

如图 6.12-19, 每个 BPWM 都有 2 个独立的中断。

BPWM 中断 (BPWM\_INT) 来自于对应的 BPWM 事件。计数器可以产生零位中断标志 ZID0 (BPWM\_INTSTS0[0]) 和周期位置中断标志 PIF0 (BPWM\_INTSTS0[8])。当 BPWM 通道 n 的计数值和 BPWM\_CMPDATn 里的比较值相同时, 会根据计数的方向触发不同的中断标志。如果发生在向上计数时, 向上中断标志 CMPUIFn (BPWM\_INSTS0[21:16]) 会被置位; 如果发生在向下计数时, 向下中断标志 CMPDIFn (BPWM\_INSTS0[29:24]) 会被置位。此时如果对应的中断使能位置位, 触发事件会产生相应的中断信号。

另一个中断是捕捉中断 (CAP\_INT)。在 NVIC 中和 BPWM 中断共用 BPWM\_INT 向量。CAPRIFn (BPWM\_CAPIF[5:0]) 触发及上升沿捕捉中断使能位 CAPRIENn (BPWM\_CAPIEN[5:0]) 置 1 都会触发 CAP\_INT; 或者, 下降沿条件下, 下降沿中断使能位 CAPFIENn (BPWM\_CAPIEN[13:8]) 置 1 时会触发 CAPFIFn (BPWM\_CAPIF[13:8])。

如图 6.12-19所示是 BPWM 中断的架构。

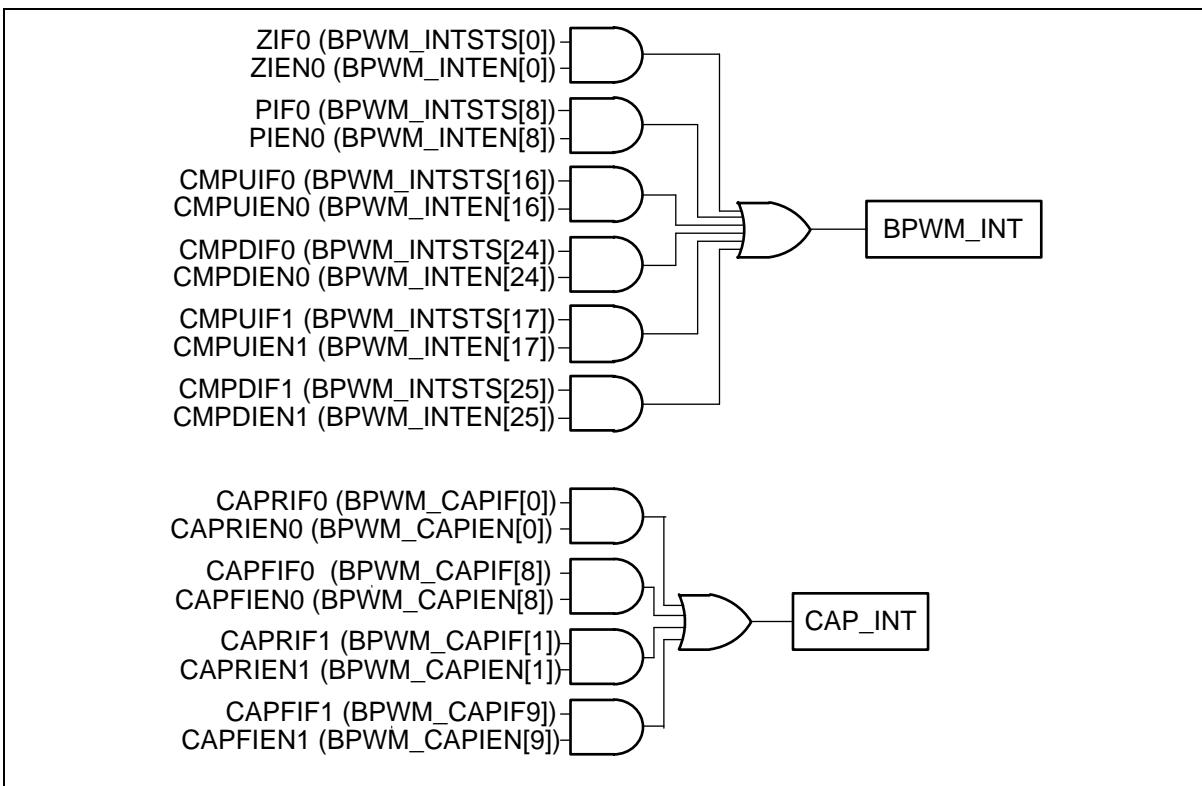


图 6.12-19 BPWM\_CH0 和 BPWM\_CH1 的中断架构

#### 6.12.5.16 BPWM 触发 EADC 发生器

BPWM 可以作为 EADC 转换的触发源。每个 BPWM 通道对共享一个触发源。设置 TRGSEL $n$  选择触发源，TRGSEL $n$  指 TRGSEL0、TRGSEL1 ... 和 TRGESL5，分别对应到 BPWM\_EADCTS0[3:0]、BPWM\_EADCTS0[11:8]、BPWM\_EADCTS0[19:16]、BPWM\_EADCTS0[27:24]、BPWM\_EADCTS1[3:0] 和 BPWM\_EADTS1[11:8]。设置 TRGEN $n$  使能触发输出到 EADC，TRGEN $n$  指 TRGEN0、TRGEN1 ... 和 TRGEN5，分别对应到 BPWM\_EADCTS0[7]、BPWM\_EADCTS0[15]、BPWM\_EADCTS0[23]、BPWM\_EADCTS0[31]、BPWM\_EADCTS1[7] 和 BPWM\_EADTS1[15]。其中， $n$  ( $n = 0, 1, \dots, 5$ ) 对应 BPWM 通道数。

一对 BPWM 通道有 7 个事件可以选作触发源。如图 6.12-20 以 BPWM\_CH0 和 BPWM\_CH1 为例。设定 PERIOD 和 CMPDAT 可以配置在不同时间点 BPWM 触发 EADC 开始转换。图 6.12-22 上下计数方式下触发 EADC 的时序波形。

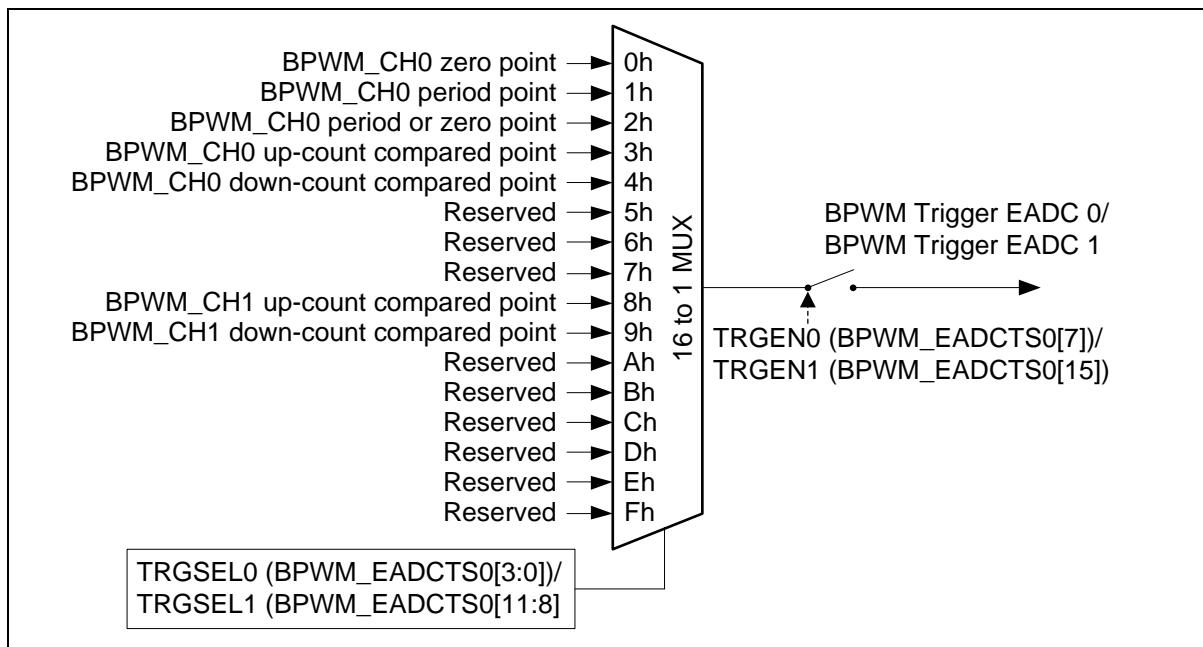


图 6.12-20 BPWM\_CH0 和 BPWM\_CH1 触发 EADC 源的框架图

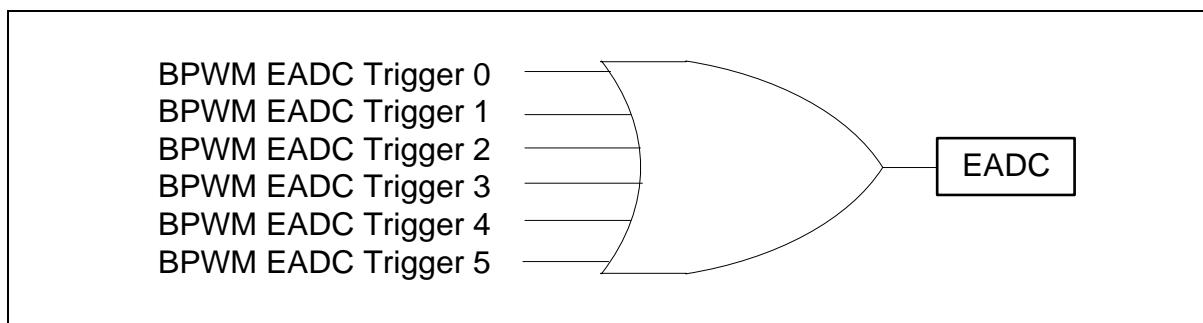


图 6.12-21 BPWM CH0~CH5 触发 EADC 的框架图

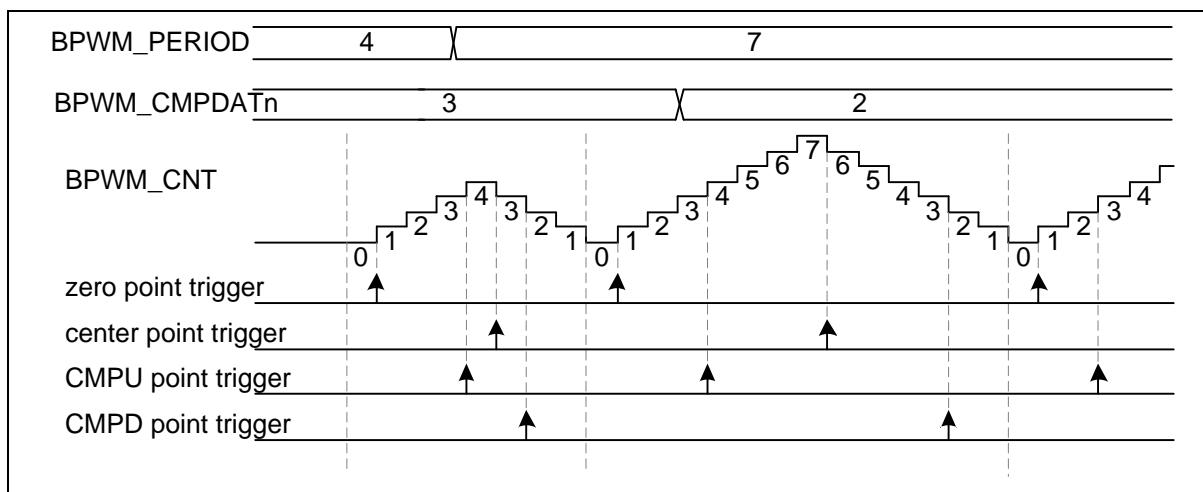


图 6.12-22 上下计数方式下 BPWM 触发 EADC 的时序波形图

### 6.12.5.17 捕捉操作

输入捕捉通道和 BPWM 输出通道共享相同的管脚和计数器。这里，计数器可以工作在向上或者向下计数方式。捕捉功能会占有 BPWM 计数器，在输入通道有上升沿时锁存 BPWM 计数器到寄存器 RCAPDATn (BPWM\_RCAPDATn[15:0])，在输入通道有下降沿时锁存 BPWM 计数器到寄存器 FCAPDATn (BPWM\_FCAPDATn[15:0])。捕捉功能在对应的上升沿或者下降沿使能时，上升沿或者下降沿发生时会产生 CAP\_INT (使用 BPWM\_INT 向量) 中断，其中 CAPRIENn (BPWM\_CAPIEN[5:0]) 是上升沿的使能位，CAPFIENn (BPWM\_CAPIEN[13:8]) 是下降沿的使能位。当上升沿锁存发生时，对应的 BPWM 计数器会重载 BPWM\_PERIOD 的值，这部分由 RCRLDENn 或者 FCRLDENn 决定 (RCRLDENn 和 FCRLDENn 分别在寄存器 BPWM\_CAPCTL[21:16] 和 BPWM\_CAPCTL[29:24] 内)。值得注意的是，需要使能捕捉通道 n 对应的 CAPINENn (BPWM\_CAPINEN[5:0]) 寄存器位来配置管脚的捕捉功能。图 6.12-23 是通道 0 的捕捉框图。

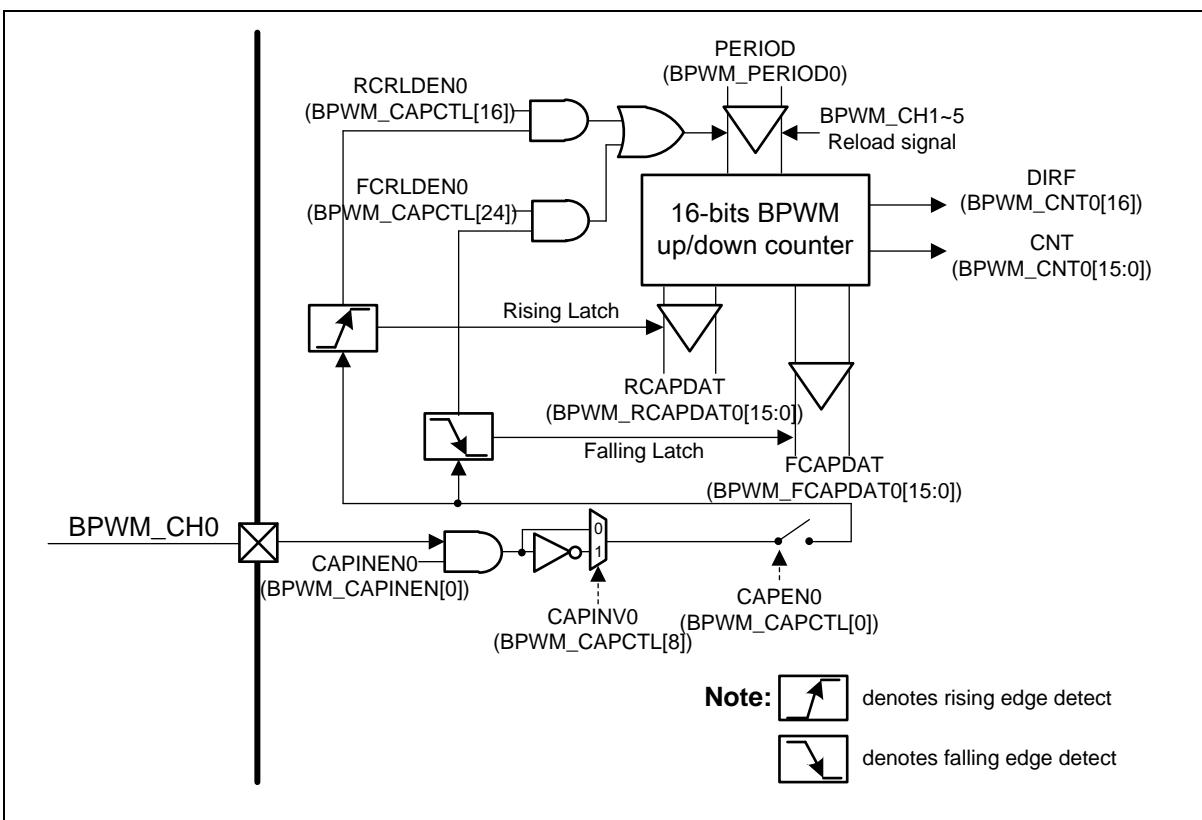


图 6.12-23 BPWM\_CH0 捕捉框图

图 6.12-24 是捕捉功能的时序示例。此时，捕捉计数器处于 BPWM 向下计数方式，PERIOS 为 8，计数器会从 8 到 0 向下计数。当检测到捕捉管脚的下降沿时，捕捉功能会锁存计数器的值到 BPWM\_RCAPDATn。在时序图中，第一次检测到下降沿时，由于 FCRLDENn 被使能，捕捉功能会从 PERIOD 的配置值重载计数器的值。但在第二次时，由于禁止了 FCRLDENn，下降沿就不会引起计数器重载了。在这个例子里，由于使能了 RCRLDENn，计数器的值在上升沿时也会被重载。

另外，对于向上计数方式，计数器的值会被重载为 0 再向上计数到 PERIOD 的值。需要特别注意的是，所有通道共享一个计数器，所以计数器重载的时间点也是有所有通道的重载信号一起控制的。

图 6.12-24 还展示了中断和中断标志发生的时序。当通道 n 检测到了上升沿时，对应的 CAPRIFn (BPWM\_CAPIF[5:0]) 位由硬件置位。类似的，当通道 n 检测到了下降沿时，对应的 CAPFIFn (BPWM\_CAPIF[13:8]) 位由硬件置位。CAPRIFn (BPWM\_CAPIF[5:0]) 和 CAPFIFn

(BPWM\_CAPIF[13:8]) 由软件写 1 清 0。如果 CAPRIFn (BPWM\_CAPIF[5:0]) 且 CAPRIENn 使能，捕捉功能会产生一个中断。如果 CAPFIFn (BPWM\_CAPIF[13:8]) 置位且 CAPFIENn 使能，同样产生一个中断。

图中没有列出的一种情况是：如果在 CAPRIFn (BPWM\_CAPIF[5:0]) 刚置位时，上升沿锁存又发生了，过载状态寄存器 CRIFOVn (BPWM\_CAPSTS[5:0]) 会由硬件置 1 来表示 CAPRIFn (BPWM\_CAPIF[5:0]) 过载了。类似的，如果下降沿锁存再次发生了，同样的情况会发生在过载状态寄存器 CRIFOVn (BPWM\_CAPSTS[13:8]) 和 CAPFIFn (BPWM\_CAPIF[13:8]) 寄存器。

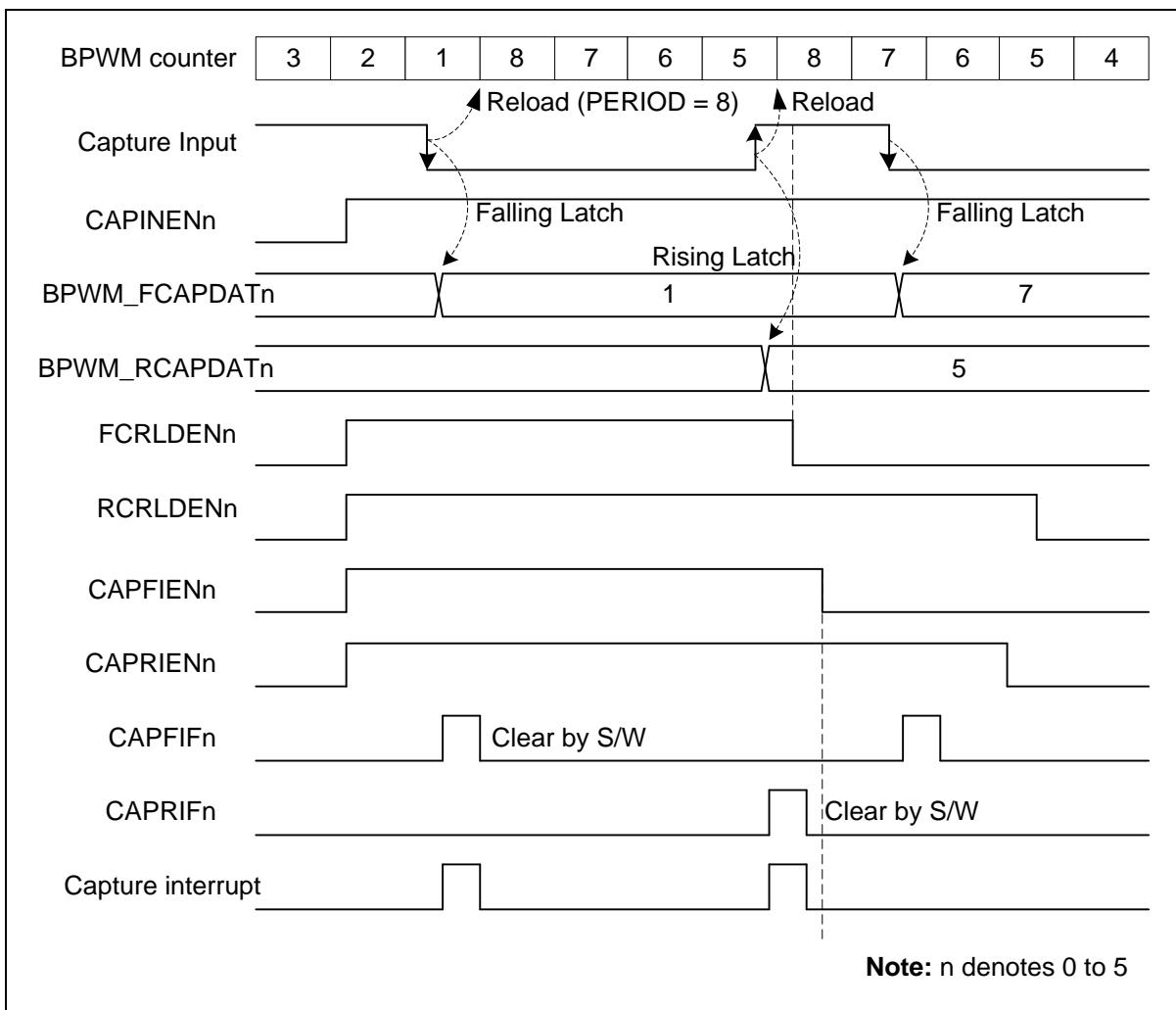


图 6.12-24 捕捉操作波形图

捕捉脉冲的宽度可以通过下面的公式进行计算：

对于负脉冲，通道上的低脉冲宽度为： $(BPWM\_PERIOD + 1 - BPWM\_RCAPDATn)$  个 BPWM 计数器时间，其中 BPWM 计数器时间为  $(CLKPSC+1) * BPWMx\_CLK$  时钟时间。如图 6.12-24，低脉冲宽度为  $8+1-5 = 4$  个 BPWM 计数器时间。

对于正脉冲，通道上的高脉冲宽度为： $(BPWM\_PERIOD + 1 - BPWM\_FCAPDATn)$  个 BPWM 计数器时间，其中 BPWM 计数器时间为  $(CLKPSC+1) * BPWMx\_CLK$  时钟时间。如图 6.12-24，低脉冲宽度为  $8+1-7 = 2$  个 BPWM 计数器时间。

### 6.12.6 寄存器映射

R: 只读, W: 只写, R/W: 读/写

寄存器	偏移量	R/W	描述	复位值
<b>BPWM 基地址:</b>				
<b>BPWM0_BA = 0x4005_A000</b>				
<b>BPWM1_BA = 0x4005_B000</b>				
<b>BPWM_CTL0</b> <i>x=0, 1</i>	BPWMx_BA+0x00	R/W	BPWM 控制寄存器 0	0x0000_0000
<b>BPWM_CTL1</b> <i>x=0, 1</i>	BPWMx_BA+0x04	R/W	BPWM 控制寄存器 1	0x0000_0000
<b>BPWM_CLKSRC</b> <i>x=0, 1</i>	BPWMx_BA+0x10	R/W	BPWM 时钟源寄存器	0x0000_0000
<b>BPWM_CLKPSC</b> <i>x=0, 1</i>	BPWMx_BA+0x14	R/W	BPWM 时钟预分频寄存器	0x0000_0000
<b>BPWM_CNTEN</b> <i>x=0, 1</i>	BPWMx_BA+0x20	R/W	BPWM 计数器使能寄存器	0x0000_0000
<b>BPWM_CNTCLR</b> <i>x=0, 1</i>	BPWMx_BA+0x24	R/W	BPWM 清除计数器寄存器	0x0000_0000
<b>BPWM_PERIOD</b> <i>x=0, 1</i>	BPWMx_BA+0x30	R/W	BPWM 周期寄存器	0x0000_0000
<b>BPWM_CMPDAT0</b> <i>x=0, 1</i>	BPWMx_BA+0x50	R/W	BPWM 比较值寄存器 0	0x0000_0000
<b>BPWM_CMPDAT1</b> <i>x=0, 1</i>	BPWMx_BA+0x54	R/W	BPWM 比较值寄存器 1	0x0000_0000
<b>BPWM_CMPDAT2</b> <i>x=0, 1</i>	BPWMx_BA+0x58	R/W	BPWM 比较值寄存器 2	0x0000_0000
<b>BPWM_CMPDAT3</b> <i>x=0, 1</i>	BPWMx_BA+0x5C	R/W	BPWM 比较值寄存器 3	0x0000_0000
<b>BPWM_CMPDAT4</b> <i>x=0, 1</i>	BPWMx_BA+0x60	R/W	BPWM 比较值寄存器 4	0x0000_0000
<b>BPWM_CMPDAT5</b> <i>x=0, 1</i>	BPWMx_BA+0x64	R/W	BPWM 比较值寄存器 5	0x0000_0000
<b>BPWM_CNT</b> <i>x=0, 1</i>	BPWMx_BA+0x90	R	BPWM 计数器寄存器	0x0000_0000
<b>BPWM_WGCTL0</b> <i>x=0, 1</i>	BPWMx_BA+0xB0	R/W	BPWM 发生器寄存器 0	0x0000_0000
<b>BPWM_WGCTL1</b> <i>x=0, 1</i>	BPWMx_BA+0xB4	R/W	BPWM 发生器寄存器 1	0x0000_0000
<b>BPWM_MSKEN</b> <i>x=0, 1</i>	BPWMx_BA+0xB8	R/W	BPWM 屏蔽使能寄存器	0x0000_0000

<b>BPWM_MSK</b> x=0, 1	BPWMx_BA+0xBC	R/W	BPWM 屏蔽数据寄存器	0x0000_0000
<b>BPWM_POLCTL</b> x=0, 1	BPWMx_BA+0xD4	R/W	BPWM 管脚极性反转寄存器	0x0000_0000
<b>BPWM_POEN</b> x=0, 1	BPWMx_BA+0xD8	R/W	BPWM 输出使能寄存器	0x0000_0000
<b>BPWM_INTEN</b> x=0, 1	BPWMx_BA+0xE0	R/W	BPWM 中断使能寄存器	0x0000_0000
<b>BPWM_INTSTS</b> x=0, 1	BPWMx_BA+0xE8	R/W	BPWM 中断标志寄存器	0x0000_0000
<b>BPWM_EADCTS0</b> x=0, 1	BPWMx_BA+0xF8	R/W	BPWM 触发 EADC 源选择寄存器 0	0x0000_0000
<b>BPWM_EADCTS1</b> x=0, 1	BPWMx_BA+0xFC	R/W	BPWM 触发 EADC 源选择寄存器 1	0x0000_0000
<b>BPWM_SSCTL</b> x=0, 1	BPWMx_BA+0x110	R/W	BPWM 同步开始控制寄存器	0x0000_0000
<b>BPWM_SSTRG</b> x=0, 1	BPWMx_BA+0x114	W	BPWM 同步开始触发寄存器	0x0000_0000
<b>BPWM_STATUS</b> x=0, 1	BPWMx_BA+0x120	R/W	BPWM 状态寄存器	0x0000_0000
<b>BPWM_CAPINEN</b> x=0, 1	BPWMx_BA+0x200	R/W	BPWM 输入捕捉使能寄存器	0x0000_0000
<b>BPWM_CAPCTL</b> x=0, 1	BPWMx_BA+0x204	R/W	BPWM 捕捉控制寄存器	0x0000_0000
<b>BPWM_CAPSTS</b> x=0, 1	BPWMx_BA+0x208	R	BPWM 捕捉状态寄存器	0x0000_0000
<b>BPWM_RCAPDATA0</b> x=0, 1	BPWMx_BA+0x20C	R	BPWM 上升沿捕捉数据寄存器 0	0x0000_0000
<b>BPWM_FCAPDATA0</b> x=0, 1	BPWMx_BA+0x210	R	BPWM 下降沿捕捉数据寄存器 0	0x0000_0000
<b>BPWM_RCAPDATA1</b> x=0, 1	BPWMx_BA+0x214	R	BPWM 上升沿捕捉数据寄存器 1	0x0000_0000
<b>BPWM_FCAPDATA1</b> x=0, 1	BPWMx_BA+0x218	R	BPWM 下降沿捕捉数据寄存器 1	0x0000_0000
<b>BPWM_RCAPDATA2</b> x=0, 1	BPWMx_BA+0x21C	R	BPWM 上升沿捕捉数据寄存器 2	0x0000_0000
<b>BPWM_FCAPDATA2</b> x=0, 1	BPWMx_BA+0x220	R	BPWM 下降沿捕捉数据寄存器 2	0x0000_0000
<b>BPWM_RCAPDATA3</b> x=0, 1	BPWMx_BA+0x224	R	BPWM 上升沿捕捉数据寄存器 3	0x0000_0000
<b>BPWM_FCAPDATA3</b>	BPWMx_BA+0x228	R	BPWM 下降沿捕捉数据寄存器 3	0x0000_0000

x=0, 1				
<b>BPWM_RCAPDAT4</b> x=0, 1	BPWMx_BA+0x22C	R	BPWM 上升沿捕捉数据寄存器 4	0x0000_0000
<b>BPWM_FCAPDAT4</b> x=0, 1	BPWMx_BA+0x230	R	BPWM 下降沿捕捉数据寄存器 4	0x0000_0000
<b>BPWM_RCAPDAT5</b> x=0, 1	BPWMx_BA+0x234	R	BPWM 上升沿捕捉数据寄存器 5	0x0000_0000
<b>BPWM_FCAPDAT5</b> x=0, 1	BPWMx_BA+0x238	R	BPWM 下降沿捕捉数据寄存器 5	0x0000_0000
<b>BPWM_CAPIEN</b> x=0, 1	BPWMx_BA+0x250	R/W	BPWM 捕捉中断使能寄存器	0x0000_0000
<b>BPWM_CAPIF</b> x=0, 1	BPWMx_BA+0x254	R/W	BPWM 捕捉中断标志寄存器	0x0000_0000
<b>BPWM_PBUF</b> x=0, 1	BPWMx_BA+0x304	R	BPWM 周期缓存	0x0000_0000
<b>BPWM_CMPBUF0</b> x=0, 1	BPWMx_BA+0x31C	R	BPWM CMPDAT 0 缓存	0x0000_0000
<b>BPWM_CMPBUF1</b> x=0, 1	BPWMx_BA+0x320	R	BPWM CMPDAT 1 缓存	0x0000_0000
<b>BPWM_CMPBUF2</b> x=0, 1	BPWMx_BA+0x324	R	BPWM CMPDAT 2 缓存	0x0000_0000
<b>BPWM_CMPBUF3</b> x=0, 1	BPWMx_BA+0x328	R	BPWM CMPDAT 3 缓存	0x0000_0000
<b>BPWM_CMPBUF4</b> x=0, 1	BPWMx_BA+0x32C	R	BPWM CMPDAT 4 缓存	0x0000_0000
<b>BPWM_CMPBUF5</b> x=0, 1	BPWMx_BA+0x330	R	BPWM CMPDAT 5 缓存	0x0000_0000

### 6.12.7 寄存器描述

#### BPWM\_CTL0 控制寄存器 0

寄存器	偏移量	R/W	描述	复位值
BPWM_CTL0	BPWMx_BA+0x00	R/W	BPWM 控制寄存器 0	0x0000_0000

31	30	29	28	27	26	25	24
DBGTRIOFF	DBGHALT	Reserved					
23	22	21	20	19	18	17	16
Reserved		IMMLDEN5	IMMLDEN4	IMMLDEN3	IMMLDEN2	IMMLDEN1	IMMLDEN0
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved		CTRLD5	CTRLD4	CTRLD3	CTRLD2	CTRLD1	CTRLD0

位	描述
[31]	<b>DBGTRIOFF</b>  ICE 调试模式确认禁止(写保护) 0 = ICE 调试模式确认会影响 BPWM 输出。此时，BPWM 管脚会强制设为三态模式 1 = ICE 调试模式确认禁止。无论 ICE 调试模式是否确认，BPWM 管脚都会保持输出 <b>注:</b> 该寄存器写保护，具体参考该寄存器写保护，具体请参考 SYS_REGLCTL 寄存器
[30]	<b>DBGHALT</b>  ICE 调试模式计数器停止(写保护) 如果计数器停止使能，BPWM 所有计数器都会保持当前值直到退出 ICE 调试模式 0 = 禁止 ICE 调试模式计数器关闭 1 = 使能 ICE 调试模式计数器关闭 <b>注:</b> 该寄存器写保护，具体请参考 SYS_REGLCTL 寄存器
[29:22]	<b>Reserved</b> 保留
[16+n] n=0,1..5	<b>IMMLDENn</b>  立即重载使能位 每一位 n 控制着对应的 BPWM 通道 n 0 = 每个周期结束位置 PERIOD 重载到 PBUF。设定 CTRLD 位，CMPDAT 会在中间位置或者结束位置重载到 CMPBUF。 1 = 软件更新 PERIOD/CMPDAT 时会立即重载到 PBUF 和 CMPBUF。 <b>注:</b> 如果 IMMLDENn 使能，WINLDENn 和 CTRLDn 无效
[15:6]	<b>Reserved</b> 保留
[n] n=0,1..5	<b>CTRLDn</b>  中心重载 每一位 n 控制着对应的 BPWM 通道 n 上下计数方式时，PERIOD 会在每个周期结束位置重载到 PBUF。CMPDAT 则会在周期中间位置重载到 CMPBUF。

**BPWM\_CTL1 控制寄存器 1**

寄存器	偏移量	R/W	描述	复位值
BPWM_CTL1	BPWMx_BA+0x04	R/W	BPWM 控制寄存器 1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved						<b>CNTTYPE0</b>	

位	描述	
[31:2]	<b>Reserved</b>	Reserved.
[1:0]	<b>CNTTYPE0</b>	<p><b>BPWM 计数器规则类型 0</b></p> <p>每一位 n 控制着对应的 BPWM 通道 n</p> <p>00 = 向上计数方式 (支持捕捉模式)</p> <p>01 = 向下计数方式 (支持捕捉模式)</p> <p>10 = 上下计数方式</p> <p>11 = 保留</p>

**BPWM\_CLKSRC 时钟源寄存器**

寄存器	偏移量	R/W	描述	复位值
BPWM_CLKSRC	BPWMx_BA+0x10	R/W	BPWM 时钟源寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved					<b>ECLKSRC0</b>		

位	描述	
[31:3]	<b>Reserved</b>	Reserved.
[2:0]	<b>ECLKSRC0</b>	<p><b>BPWM_CH01 外部时钟源选择</b></p> <p>000 = BPWMx_CLK, x 指 0 或 1.      001 = TIMER0 溢出.      010 = TIMER1 溢出.      011 = TIMER2 溢出.      100 = TIMER3 溢出.      Others = 保留</p>

**BPWM\_CLKPSC 时钟分频寄存器**

寄存器	偏移量	R/W	描述	复位值
BPWM_CLKPSC	BPWMx_BA+0x14	R/W	BPWM 时钟分频寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved				CLKPSC			
7	6	5	4	3	2	1	0
CLKPSC							

位	描述	
[31:12]	Reserved	保留
[11:0]	CLKPSC	<b>BPWM 计数器时钟分频</b> BPWM 计数器的时钟会被分频器除频。每个 BPWM 对共享一个 BPWM 计数器时钟分频器。BPWM 计数器会被除 (CLKPSC+1)。

**BPWM\_CNTEN 计数器使能寄存器**

寄存器	偏移量	R/W	描述	复位值
BPWM_CNTEN	BPWMx_BA+0x20	R/W	BPWM 计数器使能寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved							CNTEN0

位	描述	
[31:1]	Reserved	保留
[0]	CNTEN0	<b>BPWM 计数器 0 使能位</b> 0 = BPWM 计数器和时钟分频器停止运行. 1 = BPWM 计数器和时钟分频器开始运行.

BPWM\_CNTCLR 清除计数器寄存器

寄存器	偏移量	R/W	描述	复位值
BPWM_CNTCLR	BPWMx_BA+0x24	R/W	BPWM 清除计数器寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved							CNTCLR0

位	描述	
[31:1]	Reserved	Reserved.
[0]	CNTCLR0	<p>清除 BPWM 计数器控制位 0 硬件自动清除 0 = 无效 1 = 清除 16 位的 BPWM 计数器到 0000H</p>

**BPWM\_PERIOD 周期寄存器**

寄存器	偏移量	R/W	描述	复位值
BPWM_PERIOD	BPWMx_BA+0x30	R/W	BPWM 周期寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
PERIOD							
7	6	5	4	3	2	1	0
PERIOD							

位	描述	
[31:16]	<b>Reserved</b>	保留
[15:0]	<b>PERIOD</b>	<p><b>BPWM 周期 寄存器</b></p> <p>向上计数模式：该模式下，BPWM 计数器从 0 计数到 PERIOD，再重新从 0 计数</p> <p>向下计数模式：该模式下，BPWM 计数器从 PERIOD 计数到 0，再重新从 PERIOD 计数以上，BPWM 周期时间 = (PERIOD+1) * BPWM_CLK 周期</p> <p>上下计数模式：该模式下，BPWM 计数器从 0 计数到 PERIOD，再向下递减到 0，依次循环</p> <p>BPWM 周期时间= 2 * PERIOD * BPWM_CLK 周期</p>

**BPWM\_CMPDAT0~5 比较值寄存器 0~5**

寄存器	偏移量	R/W	描述	复位值
BPWM_CMPDAT0	BPWMx_BA+0x50	R/W	BPWM 比较值寄存器 0	0x0000_0000
BPWM_CMPDAT1	BPWMx_BA+0x54	R/W	BPWM 比较值寄存器 1	0x0000_0000
BPWM_CMPDAT2	BPWMx_BA+0x58	R/W	BPWM 比较值寄存器 2	0x0000_0000
BPWM_CMPDAT3	BPWMx_BA+0x5C	R/W	BPWM 比较值寄存器 3	0x0000_0000
BPWM_CMPDAT4	BPWMx_BA+0x60	R/W	BPWM 比较值寄存器 4	0x0000_0000
BPWM_CMPDAT5	BPWMx_BA+0x64	R/W	BPWM 比较值寄存器 5	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
CMPDAT							
7	6	5	4	3	2	1	0
CMPDAT							

位	描述	
[31:16]	Reserved	保留
[15:0]	CMPDAT	<b>BPWM 比较值寄存器</b> CMPDAT 用来和 CNTR 比较从而产生 BPWM 波形、中断和触发 EADC 信号 在独立模式下，CMPDAT0~5 代表 6 路 BPWM_CH0~5 的比较点

**BPWM\_CNT** 计数器寄存器

寄存器	偏移量	R/W	描述	复位值
BPWM_CNT	BPWMx_BA+0x90	R	BPWM 计数器寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
CNT							
7	6	5	4	3	2	1	0
CNT							

位	描述	
[31:17]	<b>Reserved</b>	保留
[16]	<b>DIRF</b>	<b>BPWM 方向指示标志 (只读)</b> 0 = 计数器向下计数。 1 = 计数器向上计数。
[15:0]	<b>CNT</b>	<b>BPWM 数据寄存器 (只读)</b> 用户可以从 CNTR 寄存器读到当前 16 位周期计数器的值

**BPWM\_WGCTL0** 发生器寄存器 0

寄存器	偏移量	R/W	描述	复位值
BPWM_WGCTL0	BPWMx_BA+0xB0	R/W	BPWM 发生器寄存器 0	0x0000_0000

31	30	29	28	27	26	25	24
Reserved				PRDPCTL5		PRDPCTL4	
23	22	21	20	19	18	17	16
PRDPCTL3		PRDPCTL2		PRDPCTL1		PRDPCTL0	
15	14	13	12	11	10	9	8
Reserved				ZPCTL5		ZPCTL4	
7	6	5	4	3	2	1	0
ZPCTL3		ZPCTL2		ZPCTL1		ZPCTL0	

位	描述	
[31:28]	<b>Reserved</b>	保留
[16+2n+1:16+2n] n=0,1..5	<b>PRDPCTL<sub>n</sub></b>	<b>BPWM 周期 (中间) 位置控制</b> 每一位 n 控制着对应的 BPWM 通道 n 00 = 无动作. 01 = BPWM 周期 (中间) 位置输出低 10 = BPWM 周期 (中间) 位置输出高 11 = BPWM 周期 (中间) 位置输出触发 BPWM 计数器计数到 (PERIOD+1) 时, BPWM 可以控制输出电平 <b>注:</b> 该位是 BPWM 计数器上下计数方式时的中间位置控制
[15:12]	<b>Reserved</b>	保留
[2n+1:2n] n=0,1..5	<b>ZPCTL<sub>n</sub></b>	<b>BPWM 零位控制</b> 每一位 n 控制着对应的 BPWM 通道 n 00 = 无动作. 01 = BPWM 零位输出低 10 = BPWM 零位输出高 11 = BPWM 零位输出触发 BPWM 可以在 BPWM 计数器计数到 0 时控制输出电平

**BPWM\_WGCTL1** 发生器寄存器 1

寄存器	偏移量	R/W	描述	复位值
BPWM_WGCTL1	BPWMx_BA+0xB4	R/W	BPWM 发生器寄存器 1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved				CMPDCTL5		CMPDCTL4	
23	22	21	20	19	18	17	16
CMPDCTL3		CMPDCTL2		CMPDCTL1		CMPDCTL0	
15	14	13	12	11	10	9	8
Reserved				CMPUCTL5		CMPUCTL4	
7	6	5	4	3	2	1	0
CMPUCTL3		CMPUCTL2		CMPUCTL1		CMPUCTL0	

位	描述	
[31:28]	Reserved	保留
[16+2n+1:16+2n] n=0,1..5	CMPDCTLn	<b>BPWM 向下比较位置控制</b> 每一位 n 控制着对应的 BPWM 通道 n 00 = 无动作. 01 = BPWM 向下比较位置输出低 10 = BPWM 向下比较位置输出高 11 = BPWM 向下比较位置输出触发 BPWM 在向下计数到 CMPDAT 时控制输出电平
[15:12]	Reserved	保留
[2n+1:2n] n=0,1..5	CMPUCTLn	<b>BPWM 向上比较位置控制</b> 每一位 n 控制着对应的 BPWM 通道 n 00 = 无动作. 01 = BPWM 向上比较位置输出低 10 = BPWM 向上比较位置输出高 11 = BPWM 向上比较位置输出触发 BPWM 在向上计数到 CMPDAT 时控制输出电平

**BPWM\_MSKEN 屏蔽使能寄存器**

寄存器	偏移量	R/W	描述	复位值
BPWM_MSKEN	BPWMx_BA+0xB8	R/W	BPWM 屏蔽使能寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved		MSKEN5	MSKEN4	MSKEN3	MSKEN2	MSKEN1	MSKEN0

位	描述	
[31:6]	Reserved	保留
[n] n=0,1..5	MSKENn	<p><b>BPWM 屏蔽使能 位</b></p> <p>每一位 n 控制着对应的 BPWM 通道 n</p> <p>该位使能时 BPWM 输出信号被屏蔽。对应的 BPWM 通道 n 会输出 MSKDATn (BPWM_MSK[5:0]) 的值</p> <p>0 = BPWM 输出信号未被屏蔽</p> <p>1 = BPWM 输出信号被屏蔽，输出 MSKDATn 的值</p>

**BPWM\_MSK 屏蔽数据寄存器**

寄存器	偏移量	R/W	描述	复位值
BPWM_MSK	BPWMx_BA+0xBC	R/W	BPWM 屏蔽数据寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved		MSKDAT5	MSKDAT4	MSKDAT3	MSKDAT2	MSKDAT1	MSKDAT0

位	描述	
[31:6]	Reserved	保留
[n] n=0,1..5	MSKDATn	<p><b>BPWM 屏蔽数据位</b></p> <p>如果对应的管脚屏蔽功能使能，该数据位控制 BPWM 管脚输出状态</p> <p>每一位 n 控制着对应的 BPWM 通道 n</p> <p>0 = BPWMn 输出逻辑电平低</p> <p>1 = BPWMn 输出逻辑电平高</p>

**BPWM\_POLCTL 管脚极性反转寄存器**

寄存器	偏移量	R/W	描述	复位值
BPWM_POLCTL	BPWMx_BA+0xD4	R/W	BPWM 管脚极性反转寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved		PINV5	PINV4	PINV3	PINV2	PINV1	PINV0

位	描述	
[31:6]	Reserved	保留
[n] n=0,1..5	PINVn	<p><b>BPWM 管脚极性反转控制</b></p> <p>该寄存器控制 BPWM 输出极性的状态</p> <p>每一位 n 控制着对应的 BPWM 通道 n</p> <p>0 = 禁止BPWM 输出极性反转</p> <p>1 = 使能BPWM 输出极性反转</p>

**BPWM\_POEN** 输出使能寄存器

寄存器	偏移量	R/W	描述	复位值
BPWM_POEN	BPWMx_BA+0xD8	R/W	BPWM 输出使能寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved		POEN5	POEN4	POEN3	POEN2	POEN1	POEN0

位	描述	
[31:6]	Reserved	保留
[n] n=0,1..5	POENn	<b>BPWM 管脚输出使能位</b> 每一位 n 控制着对应的 BPWM 通道 n 0 = BPWM 管脚处于三态模式 1 = BPWM 管脚处于输出模式

**BPWM\_INTEN 中断使能寄存器**

寄存器	偏移量	R/W	描述	复位值
BPWM_INTEN	BPWMx_BA+0xE0	R/W	BPWM 中断使能寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved		CMPDIEN5	CMPDIEN4	CMPDIEN3	CMPDIEN2	CMPDIEN1	CMPDIEN0
23	22	21	20	19	18	17	16
Reserved		CMPUIEN5	CMPUIEN4	CMPUIEN3	CMPUIEN2	CMPUIEN1	CMPUIEN0
15	14	13	12	11	10	9	8
Reserved							PIENO
7	6	5	4	3	2	1	0
Reserved							ZIENO

位	描述	
[31:30]	<b>Reserved</b>	保留
[24+n] n=0,1..5	<b>CMPDIENn</b>	<b>BPWM 向下计数中断使能 位</b> 每一位 n 控制着对应的 BPWM 通道 n 0 = 禁止向下计数中断 1 = 使能向下计数中断
[23:22]	<b>Reserved</b>	保留
[16+n] n=0,1..5	<b>CMPUIENn</b>	<b>BPWM 向上计数中断使能 位</b> 每一位 n 控制着对应的 BPWM 通道 n 0 = 禁止向上计数中断 1 = 使能向上计数中断
[15:9]	<b>Reserved</b>	保留
[8]	<b>PIENO</b>	<b>BPWM 周期位置中断 0 使能位</b> 0 = 禁止周期位置中断 1 = 使能周期位置中断 <b>注:</b> 上下计数方式里周期位置指中间位置
[7:1]	<b>Reserved</b>	保留
[0]	<b>ZIENO</b>	<b>BPWM 零位中断 0 使能位</b> 0 = 禁止零位中断 1 = 使能零位中断

**BPWM\_INTSTS 中断标志寄存器**

寄存器	偏移量	R/W	描述				复位值
BPWM_INTSTS	BPWMx_BA+0xE8	R/W	BPWM 中断标志寄存器				0x0000_0000

31	30	29	28	27	26	25	24
Reserved		CMPDIF5	CMPDIF4	CMPDIF3	CMPDIF2	CMPDIF1	CMPDIF0
23	22	21	20	19	18	17	16
Reserved		CMPUIF5	CMPUIF4	CMPUIF3	CMPUIF2	CMPUIF1	CMPUIF0
15	14	13	12	11	10	9	8
Reserved							PIF0
7	6	5	4	3	2	1	0
Reserved							ZIF0

位	描述	
[31:30]	<b>Reserved</b>	保留
[24+n] n=0,1..5	<b>CMPDIFn</b>	<b>BPWM 向下比较计数中断标志位</b> 每一位 n 控制着对应的 BPWM 通道 n BPWM 向下计数到 BPWM_CMPDATn 时，硬件置位该标志。软件可写 1 清零该位。 <b>注:</b> 如果 CMPDAT 等于 PERIOD，该标志在向下计数方式时无效
[23:22]	<b>Reserved</b>	保留
[16+n] n=0,1..5	<b>CMPUIFn</b>	<b>BPWM 向上比较计数中断标志</b> 每一位 n 控制着对应的 BPWM 通道 n BPWM 向上计数到 BPWM_CMPDATn 时，硬件置位该标志。软件可写 1 清 0 该位。 <b>注:</b> 如果 CMPDAT 等于 PERIOD，该标志在向上计数方式时无效。
[15:9]	<b>Reserved</b>	保留
[8]	<b>PIF0</b>	<b>BPWM 周期位置中断标志 0</b> BPWM_CH0 向上计数到 BPWM_PERIOD0 时，硬件置位该位。软件可写 1 清 0 该位。
[7:1]	<b>Reserved</b>	保留
[0]	<b>ZIF0</b>	<b>BPWM 零位中断标志 0</b> BPWM_CH0 向上计数到 0 时，硬件置位该位。软件可写 1 清 0 该位。

**BPWM\_EADCTS0 触发 EADC 源选择寄存器 0**

寄存器	偏移量	R/W	描述	复位值
BPWM_EADCTS0	BPWMx_BA+0xF8	R/W	BPWM 触发 EADC 源选择寄存器 0	0x0000_0000

31	30	29	28	27	26	25	24
TRGEN3	Reserved				TRGSEL3		
23	22	21	20	19	18	17	16
TRGEN2	Reserved				TRGSEL2		
15	14	13	12	11	10	9	8
TRGEN1	Reserved				TRGSEL1		
7	6	5	4	3	2	1	0
TRGEN0	Reserved				TRGSEL0		

位	描述
[31]	TRGEN3      BPWM_CH3 触发 EADC 使能位
[30:28]	Reserved      保留
[27:24]	TRGSEL3 <b>BPWM_CH3 触发 EADC 源选择</b> 0000 = BPWM_CH2 零位. 0001 = BPWM_CH2 周期位置. 0010 = BPWM_CH2 零或周期位置. 0011 = BPWM_CH2 向上计数的 CMPDAT 位置. 0100 = BPWM_CH2 向下计数的 CMPDAT 位置. 0101 = 保留 0110 = 保留 0111 = 保留 1000 = BPWM_CH3 向上计数的 CMPDAT 位置. 1001 = BPWM_CH3 向下计数的 CMPDAT 位置. 其他保留
[23]	TRGEN2      BPWM_CH2 触发 EADC 使能位
[22:20]	Reserved      保留
[19:16]	TRGSEL2 <b>BPWM_CH2 触发 EADC 源选择</b> 0000 = BPWM_CH2 零位. 0001 = BPWM_CH2 周期位置. 0010 = BPWM_CH2 零位或周期位置. 0011 = BPWM_CH2 向上计数的 CMPDAT 位置. 0100 = BPWM_CH2 向下计数的 CMPDAT 位置. 0101 = 保留 0110 = 保留

		0111 = 保留 1000 = BPWM_CH3 向上计数的 CMPDAT 位置. 1001 = BPWM_CH3 向下计数的 CMPDAT 位置. 其他保留
[15]	<b>TRGEN1</b>	BPWM_CH1 触发 EADC 使能位
[14:12]	<b>Reserved</b>	保留
[11:8]	<b>TRGSEL1</b>	<b>BPWM_CH1 触发 EADC 源选择</b> 0000 = BPWM_CH0 零位. 0001 = BPWM_CH0 周期位置. 0010 = BPWM_CH0 零位或周期位置. 0011 = BPWM_CH0 向上计数的 CMPDAT 位置. 0100 = BPWM_CH0 向下计数的 CMPDAT 位置. 0101 = 保留 0110 = 保留 0111 = 保留 1000 = BPWM_CH1 向上计数的 CMPDAT 位置. 1001 = BPWM_CH1 向下计数的 CMPDAT 位置. 其他保留
[7]	<b>TRGEN0</b>	BPWM_CH0 触发 EADC 使能位
[6:4]	<b>Reserved</b>	保留
[3:0]	<b>TRGSEL0</b>	<b>BPWM_CH0 触发 EADC 源选择</b> 0011 = BPWM_CH0 向上计数的 CMPDAT 位置. 0100 = BPWM_CH0 向下计数的 CMPDAT 位置. 0101 = 保留 0110 = 保留 0111 = 保留 1000 = BPWM_CH1 向上计数的 CMPDAT 位置. 1001 = BPWM_CH1 向下计数的 CMPDAT 位置. 其他保留

**BPWM\_EADCTS1 触发 EADC 源选择寄存器 1**

寄存器	偏移量	R/W	描述	复位值
BPWM_EADCTS1	BPWMx_BA+0xFC	R/W	BPWM 触发 EADC 源选择寄存器 1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
TRGEN5	Reserved			TRGSEL5			
7	6	5	4	3	2	1	0
TRGEN4	Reserved			TRGSEL4			

位	描述
[31:16]	<b>Reserved</b> 保留
[15]	<b>TRGEN5</b> BPWM_CH5 触发 EADC 使能位
[14:12]	<b>Reserved</b> 保留
[11:8]	<b>TRGSEL5</b> BPWM_CH5 触发 EADC 源选择 0000 = BPWM_CH4 零位. 0001 = BPWM_CH4 周期位置. 0010 = BPWM_CH4 零位或周期位置. 0011 = BPWM_CH4 向上计数的 CMPDAT 位置. 0100 = BPWM_CH4 向下计数的 CMPDAT 位置. 0101 = 保留 0110 = 保留 0111 = 保留 1000 = BPWM_CH5 向上计数的 CMPDAT 位置. 1001 = BPWM_CH5 向下计数的 CMPDAT 位置. 其他保留
[7]	<b>TRGEN4</b> BPWM_CH4 触发 EADC 使能位
[6:4]	<b>Reserved</b> 保留
[3:0]	<b>TRGSEL4</b> BPWM_CH4 触发 EADC 源选择 0000 = BPWM_CH4 零位. 0001 = BPWM_CH4 周期位置. 0010 = BPWM_CH4 零位或周期位置. 0011 = BPWM_CH4 向上计数的 CMPDAT 位置. 0100 = BPWM_CH4 向下计数的 CMPDAT 位置.

		0101 = 保留 0110 = 保留 0111 = 保留 1000 = BPWM_CH5 向上计数的 CMPDAT 位置. 1001 = BPWM_CH5 向下计数的 CMPDAT 位置. 其他保留
--	--	---

**BPWM\_SSCTL** 同步开始寄存器

寄存器	偏移量	R/W	描述	复位值
BPWM_SSCTL	BPWMx_BA+0x110	R/W	BPWM 同步开始寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved						SSRC	
7	6	5	4	3	2	1	0
Reserved							

位	描述	
[31:10]	<b>Reserved</b>	保留
[9:8]	<b>SSRC</b>	<b>BPWM 同步开始源选择</b> 00 = 同步开始源来自 PWM0. 01 = 同步开始源来自 PWM1. 10 = 同步开始源来自 BPWM0. 11 = 同步开始源来自 BPWM1.
[7:1]	<b>Reserved</b>	保留
[0]	<b>SSEN0</b>	<b>BPWM 同步开始功能 0 使能位</b> 同步开始功能使能后，写 BPWM 同步开始使能位 (CNTSEN) 可以使能 BPWM_CHO 计数器使能位 (CNTENO) 0 = 禁止BPWM 同步开始功能 1 = 使能BPWM 同步开始功能

**BPWM\_SSTRG 同步开始触发寄存器**

寄存器	偏移量	R/W	描述	复位值
BPWM_SSTRG	BPWMx_BA+0x114	W	BPWM 同步开始触发寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved							CNTSEN

位	描述	
[31:1]	Reserved	保留
[0]	CNTSEN	<b>BPWM 计数器同步开始使能位 (只写)</b> BPWM 计数器同步使能功能可以使 PWM 和 BPWM 通道同时开始计数 如果相关的 BPWM 通道计数器同步开始功能也使能了，向该位写 1 还会使能计数器使能位

**BPWM\_STATUS 状态寄存器**

寄存器	偏移量	R/W	描述	复位值
BPWM_STATUS	BPWMx_BA+0x120	R/W	BPWM 状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved	EADCTRG5	EADCTRG4	EADCTRG3	EADCTRG2	EADCTRG1	EADCTRG0	
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved							CNTMAX0

位	描述	
[31:22]	<b>Reserved</b>	保留
[16+n] n=0,1..5	<b>EADCTRGn</b>	<b>EADC 开始转换状态</b> 每一位 n 控制着对应的 BPWM 通道 n 0 = 没有 EADC 开始转换触发事件发生 1 = 有 EADC 开始转换触发事件发生，软件写 1 清 0 该位
[15:1]	<b>Reserved</b>	保留
[0]	<b>CNTMAX0</b>	<b>时间基准计数器 0 为 0xFFFF 锁存状态</b> 0 = 时间基准计数器从未达到最大值 0xFFFF 1 = 时间基准计数器达到最大值，软件写 1 清 0 该位

**BPWM\_CAPINEN 输入捕捉使能寄存器**

寄存器	偏移量	R/W	描述	复位值
BPWM_CAPINEN	BPWMx_BA+0x200	R/W	BPWM 输入捕捉使能寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved		CAPINEN5	CAPINEN4	CAPINEN3	CAPINEN2	CAPINEN1	CAPINEN0

位	描述	
[31:6]	Reserved	保留
[n] n=0,1..5	CAPINENn	<p><b>输入捕捉使能 位</b>            每一位 n 控制着对应的 BPWM 通道 n            0 = 禁止BPWM 通道输入捕捉通路。BPWM 通道捕捉功能的输入将被视作 0            1 = 使能BPWM 通道输入捕捉通路。BPWM 通道捕捉的输入由多功能引脚的状态决定         </p>

**BPWM\_CAPCTL 捕捉控制寄存器**

寄存器	偏移量	R/W	描述	复位值
BPWM_CAPCTL	BPWMx_BA+0x204	R/W	BPWM Capture Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved		FCRLDEN5	FCRLDEN4	FCRLDEN3	FCRLDEN2	FCRLDEN1	FCRLDEN0
23	22	21	20	19	18	17	16
Reserved		RCRLDEN5	RCRLDEN4	RCRLDEN3	RCRLDEN2	RCRLDEN1	RCRLDEN0
15	14	13	12	11	10	9	8
Reserved		CAPINV5	CAPINV4	CAPINV3	CAPINV2	CAPINV1	CAPINV0
7	6	5	4	3	2	1	0
Reserved		CAPEN5	CAPEN4	CAPEN3	CAPEN2	CAPEN1	CAPEN0

位	描述	
[31:30]	<b>Reserved</b>	保留
[24+n] n=0,1..5	<b>FCRLDENn</b>	下降沿捕捉重载使能位 每一位 n 控制着对应的 BPWM 通道 n 0 = 禁止下降沿捕捉重载计数器 1 = 使能下降沿捕捉重载计数器
[23:22]	<b>Reserved</b>	保留
[16+n] n=0,1..5	<b>RCRLDENn</b>	上升沿捕捉重载使能位 每一位 n 控制着对应的 BPWM 通道 n 0 = 禁止上升沿捕捉重载计数器 1 = 使能上升沿捕捉重载计数器
[15:14]	<b>Reserved</b>	保留
[8+n] n=0,1..5	<b>CAPINVn</b>	捕捉反相使能位 每一位 n 控制着对应的 BPWM 通道 n 0 = 禁止捕捉源反相 1 = 使能捕捉源反相。从 GPIO 反转输入信号。
[7:6]	<b>Reserved</b>	保留
[n] n=0,1..5	<b>CAPENn</b>	捕捉功能使能位 每一位 n 控制着对应的 BPWM 通道 n 0 = 禁止捕捉功能。RCAPDAT/FCAPDAT 不会被更新。 1 = 使能捕捉功能。

**BPWM\_CAPSTS 捕捉状态寄存器**

寄存器	偏移量	R/W	描述	复位值
BPWM_CAPSTS	BPWMx_BA+0x208	R	BPWM 捕捉状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved		CFIFOV5	CFIFOV4	CFIFOV3	CFIFOV2	CFIFOV1	CFIFOV0
7	6	5	4	3	2	1	0
Reserved		CRIFOV5	CRIFOV4	CRIFOV3	CRIFOV2	CRIFOV1	CRIFOV0

位	描述	
[31:14]	Reserved	保留
[8+n] n=0,1..5	CFIFOVn	下降沿捕捉中断标志溢出状态 (只读) 每一位 n 控制着对应的 BPWM 通道 n 对应的 CAPIF 为 1 时, 如果发生了下降沿锁存, 该位会置位。 注: 用户清除对应的 CAPIF 时, 该位会自动清零
[7:6]	Reserved	保留
[n] n=0,1..5	CRIFOVn	上升沿捕捉中断标志溢出状态 (只读) 每一位 n 控制着对应的 BPWM 通道 n 对应的 CAPRIF 为 1 时, 如果发生了上升沿锁存, 该位会置位。 注: 用户清除对应的 CAPRIF 时, 该位会自动清零

**BPWM\_RCAPDAT 0~5 上升沿捕捉数据寄存器 0~5**

寄存器	偏移量	R/W	描述	复位值
BPWM_RCAPDAT0	BPWMx_BA+0x20C	R	BPWM 上升沿捕捉数据寄存器 0	0x0000_0000
BPWM_RCAPDAT1	BPWMx_BA+0x214	R	BPWM 上升沿捕捉数据寄存器 1	0x0000_0000
BPWM_RCAPDAT2	BPWMx_BA+0x21C	R	BPWM 上升沿捕捉数据寄存器 2	0x0000_0000
BPWM_RCAPDAT3	BPWMx_BA+0x224	R	BPWM 上升沿捕捉数据寄存器 3	0x0000_0000
BPWM_RCAPDAT4	BPWMx_BA+0x22C	R	BPWM 上升沿捕捉数据寄存器 4	0x0000_0000
BPWM_RCAPDAT5	BPWMx_BA+0x234	R	BPWM 上升沿捕捉数据寄存器 5	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
RCAPDAT							
7	6	5	4	3	2	1	0
RCAPDAT							

位	描述	
[31:16]	Reserved	保留
[15:0]	RCAPDAT	BPWM 上升沿捕捉数据 (只读) 上升沿捕捉发生时，BPWM 计数器值会被保存到该寄存器

**BPWM\_FCAPDAT 0~5 下降沿捕捉数据寄存器 0~5**

寄存器	偏移量	R/W	描述	复位值
BPWM_FCAPDAT0	BPWMx_BA+0x210	R	BPWM 下降沿捕捉数据寄存器 0	0x0000_0000
BPWM_FCAPDAT1	BPWMx_BA+0x218	R	BPWM 下降沿捕捉数据寄存器 1	0x0000_0000
BPWM_FCAPDAT2	BPWMx_BA+0x220	R	BPWM 下降沿捕捉数据寄存器 2	0x0000_0000
BPWM_FCAPDAT3	BPWMx_BA+0x228	R	BPWM 下降沿捕捉数据寄存器 3	0x0000_0000
BPWM_FCAPDAT4	BPWMx_BA+0x230	R	BPWM 下降沿捕捉数据寄存器 4	0x0000_0000
BPWM_FCAPDAT5	BPWMx_BA+0x238	R	BPWM 下降沿捕捉数据寄存器 5	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
FCAPDAT							
7	6	5	4	3	2	1	0
FCAPDAT							

位	描述	
[31:16]	Reserved	保留
[15:0]	FCAPDAT	BPWM 下降沿捕捉数据 (只读) 下降沿捕捉发生时，BPWM 计数器值会被保存到该寄存器

**BPWM\_CAPIEN** 捕捉中断使能寄存器

寄存器	偏移量	R/W	描述	复位值
BPWM_CAPIEN	BPWMx_BA+0x250	R/W	BPWM 捕捉中断使能寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved		CAPFIEN5	CAPFIEN4	CAPFIEN3	CAPFIEN2	CAPFIEN1	CAPFIEN0
7	6	5	4	3	2	1	0
Reserved		CAPRIEN5	CAPRIEN4	CAPRIEN3	CAPRIEN2	CAPRIEN1	CAPRIEN0

位	描述	
[31:14]	Reserved	保留
[13:8]	CAPFIENn	<b>BPWM 捕捉下降沿锁存中断使能位</b> 每一位 n 控制着对应的 BPWM 通道 n 0 = 禁止 捕捉下降沿锁存中断 1 = 使能捕捉下降沿锁存中断
[7:6]	Reserved	保留
[5:0]	CAPRIENn	<b>BPWM 捕捉上升沿锁存中断使能位</b> 每一位 n 控制着对应的 BPWM 通道 n 0 = 禁止捕捉上升沿锁存中断 1 = 使能捕捉上升沿锁存中断

**BPWM\_CAPIF****捕捉中断标志寄存器**

寄存器	偏移量	R/W	描述	复位值
BPWM_CAPIF	BPWMx_BA+0x254	R/W	BPWM 捕捉中断标志寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved		CAPFIF5	CAPFIF4	CAPFIF3	CAPFIF2	CAPFIF1	CAPFIF0
7	6	5	4	3	2	1	0
Reserved		CAPRIF5	CAPRIF4	CAPRIF3	CAPRIF2	CAPRIF1	CAPRIF0

位	描述	
[31:14]	<b>Reserved</b>	保留
[8+n] n=0,1..5	<b>CAPFIFn</b>	<b>BPWM 捕捉下降沿锁存中断标志</b> 该位写 1 清 0. 每一位 n 控制着对应的 BPWM 通道 n 0 = 无捕捉下降沿锁存情况发生 1 = 捕捉下降沿锁存发生, 该位置 1
[7:6]	<b>Reserved</b>	保留
[n] n=0,1..5	<b>CAPRIFn</b>	<b>BPWM 捕捉上升沿锁存中断标志</b> 该位写 1 清 0. 每一位 n 控制着对应的 BPWM 通道 n 0 = 无捕捉上升沿锁存情况发生 1 = 捕捉上升沿锁存发生, 该位置 1

**BPWM\_PBUF 周期缓存**

寄存器	偏移量	R/W	描述	复位值
BPWM_PBUF	BPWMx_BA+0x304	R	BPWM 周期缓存	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
PBUF							
7	6	5	4	3	2	1	0
PBUF							

位	描述	
[31:16]	Reserved	保留
[15:0]	PBUF	BPWM 周期缓存 (只读) 同 PERIOD 有效寄存器一样使用

**BPWM\_CMPBUF0~5 比较值寄存器 缓存 0~5**

寄存器	偏移量	R/W	描述	复位值
BPWM_CMPBUF0	BPWMx_BA+0x31C	R	BPWM CMPDAT 0 缓存	0x0000_0000
BPWM_CMPBUF1	BPWMx_BA+0x320	R	BPWM CMPDAT 1 缓存	0x0000_0000
BPWM_CMPBUF2	BPWMx_BA+0x324	R	BPWM CMPDAT 2 缓存	0x0000_0000
BPWM_CMPBUF3	BPWMx_BA+0x328	R	BPWM CMPDAT 3 缓存	0x0000_0000
BPWM_CMPBUF4	BPWMx_BA+0x32C	R	BPWM CMPDAT 4 缓存	0x0000_0000
BPWM_CMPBUF5	BPWMx_BA+0x330	R	BPWM CMPDAT 5 缓存	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
CMPBUF							
7	6	5	4	3	2	1	0
CMPBUF							

位	描述	
[31:16]	Reserved	保留
[15:0]	CMPBUF	BPWM 比较器缓存 (只读) 同 CMP 有效寄存器一样使用

## 6.13 QEI 正交编码接口

### 6.13.1 概述

M480 有两组 QEI 控制器。正交编码接口 (QEI) 解读转速和动作传感器的信息。它可以用在任何使用正交编码器作为反馈的应用里。

### 6.13.2 特性

#### 6.13.2.1 正交编码接口 (QEI) 特性

- 多达 2 个 QEI 控制器, QEIO 及 QEI1
- 2 个 QEI 相位输入, QEA 及 QEB; 1 个索引输入
- 1 个 32 位的上/下正交编码脉冲计数器 (QEI\_CNT)
- 1 个 32 位的软件锁存正交编码脉冲计数器保存寄存器 (QEI\_CNTHOLD)
- 1 个 32 位的正交编码脉冲计数器索引锁存寄存器 (QEI\_CNTLATCH)
- 1 个 32 位的正交编码脉冲计数器比较寄存器 (QEI\_CNTCMP), 并提供最大计数预设值寄存器 (QEI\_CNTMAX)
- 1 个 QEI 控制寄存器 (QEI\_CTL) 和 1 个 QEI 状态寄存器 (QEI\_STATUS)
- 4 种 正交编码脉冲计数器操作模式
- 支持 x4 自由计数模式
- 支持 x2 自由计数模式
- 支持 x4 比较计数模式
- 支持 x2 比较计数模式
- 编码脉冲宽度测量模式
- 不做消噪时, QEA/QEB/IDX 的输入频率必须低于 PCLK/4
- 有做消噪时, QEA/QEB/IDX 的输入频率必须低于 噪声滤波器的 Clk/8

### 6.13.3 框图

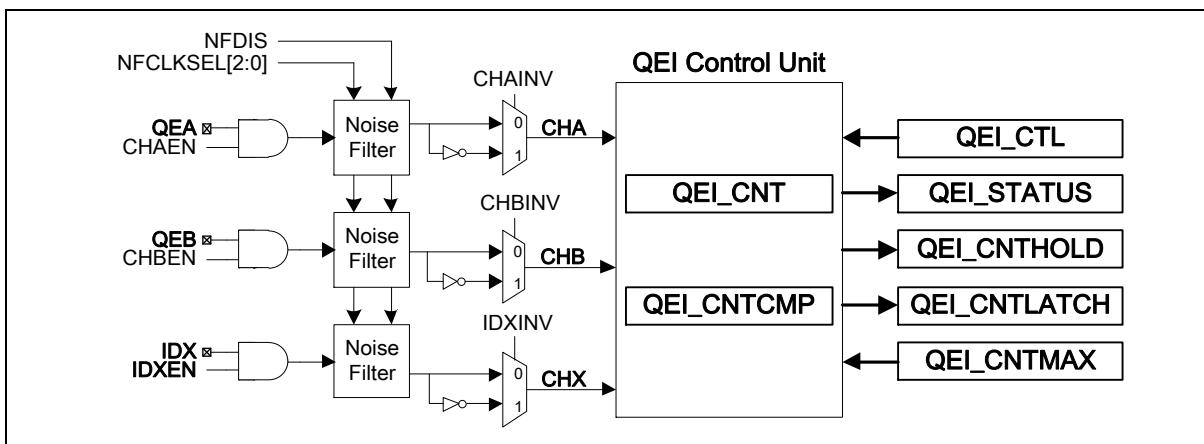


图 6.13-1 QEI 框图

QEI 控制器 QEA 和 QEB 的输入来自于正交编码源的输出，如增量式光电轴角编码器。需要两个相位相差 90 度的通道 A 和 B。正交编码器一般会提供一个索引信号（管脚 IDX），可以指示一个绝对位置。每个 QEI 控制单元前都有一个噪声滤波器和极性控制。

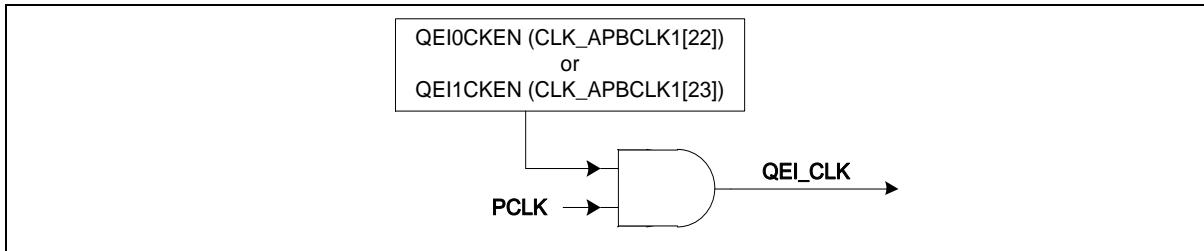


图 6.13-2 QEI 时钟控制

#### 6.13.4 基本配置

##### 6.13.4.1 QEI0 基本配置

- 时钟源配置
  - 在 QEI0CKEN (CLK\_APBCLK1[22]) 里使能 QEI0 外设时钟
- 复位配置
  - 在 QEI0RST (SYS\_IPRST2[22]) 复位 QEI0 控制器
- 管脚配置

Group	Pin Name	GPIO	MFP
QEI0	QEI0_A	PD.11	MFP10
		PE.3	MFP11
		PA.4	MFP14
	QEI0_B	PD.10	MFP10
		PE.2	MFP11
		PA.3	MFP14
	QEI0_INDEX	PD.12	MFP10
		PE.4	MFP11
		PA.5	MFP14

##### 6.13.4.2 QEI1 基本配置

- 时钟源选择
  - 在 QEI1CKEN (CLK\_APBCLK1[23]) 使能 QEI1 外设时钟
- 复位配置
  - 在 QEI1RST (SYS\_IPRST2[23]) 复位 QEI1 控制器
- 管脚配置

Group	Pin Name	GPIO	MFP
QEI1	QEI1_A	PA.9	MFP10

	PE.6	MFP11
	PA.13	MFP12
QEI1_B	PA.8	MFP10
	PE.5	MFP11
	PA.14	MFP12
QEI1_INDEX	PA.10	MFP10
	PE.7	MFP11
	PA.12	MFP12

### 6.13.5 功能描述

QEI 控制器通过检测过滤后的信号 CHA、CHB 以及 CHX 间的相位提前/滞后情况来确定方向标志位 DIRF (QEI\_STATUS[8]) 的值和脉冲计数器的值。比较计数模式下，通过判断脉冲计数器和最大计数值来判断是否重载脉冲计数器。自由计数模式下，脉冲计数值 CNT (QEI\_CNT[31:0]) 会一直计数到 0xFFFF\_FFFF。而比较计数模式下，脉冲计数器会计数到 CNTMAX (QEI\_CNTMAX[31:0])，接着脉冲计数器会复位成 0，开始下一个计数循环。

#### 6.13.5.1 噪声滤波器

每个 QEI 输入管脚都配备了一个噪声滤波器，可以过滤掉多余的噪声。通过 NFDIS (QEI\_CTL[3]) 寄存器位可以关闭 QEA、QEB 和 IDX 的噪声滤波器。如果使能了噪声滤波器，捕捉逻辑会在连续 4 个采样值相同时才传递信号。图 7.13\_5 是一个数字滤波器的实现，输入捕捉要求的脉冲间隔为 4 个 QEI\_CLK。任何小于 4 个 QEI\_CLK 的脉冲信号都会被忽略，噪声滤波器所需的时序参见图 7.13\_5。如图 7.13\_3，CHA, CHB 和 CHX 分别是 QEA, QEB 和 IDX 经过噪声滤波器和极性控制后的输出。如果关闭了噪声滤波器，输入信号 QEA, QEB 和 IDX 会直接传到内部的 CHA, CHB 和 CHX，没有任何延时。

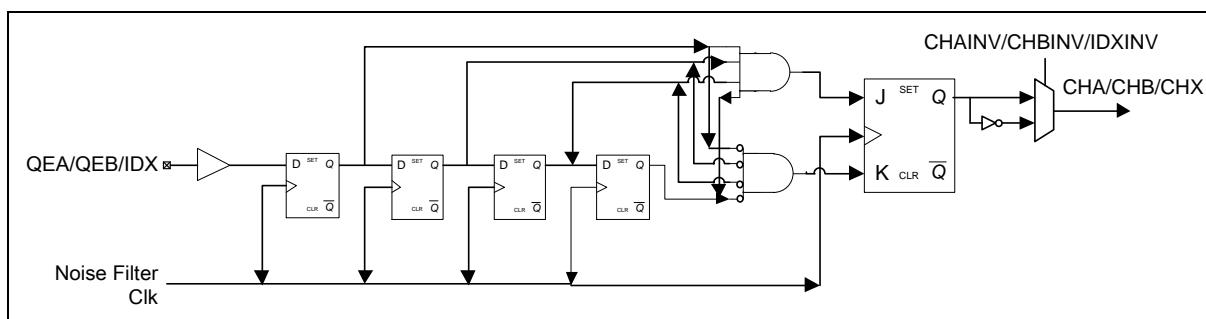


图 6.13-3 噪声滤波器

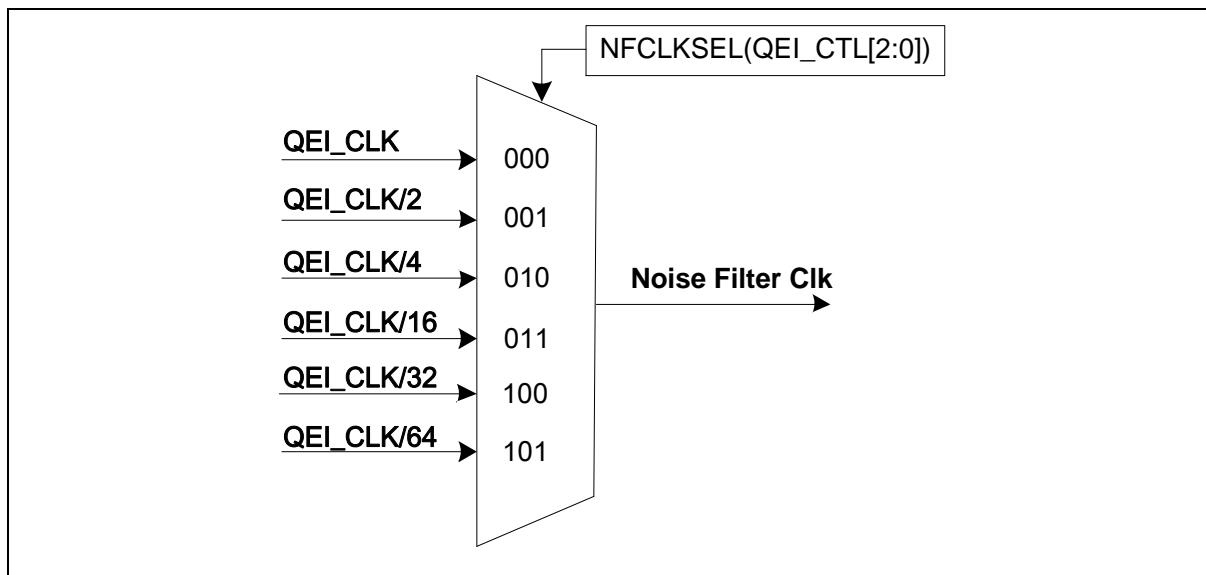


图 6.13-4 噪声滤波器采样时钟源选择

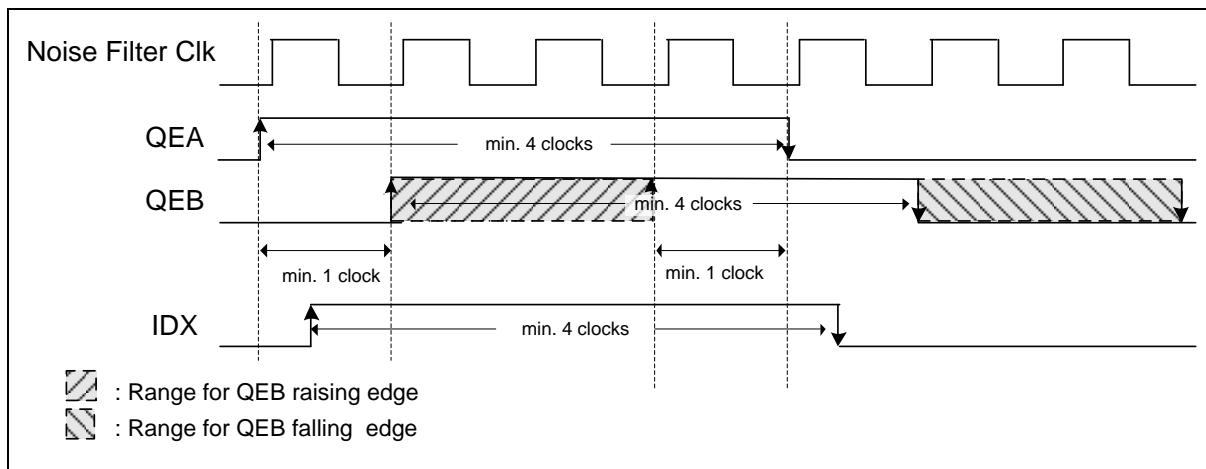


图 6.13-5 QEA/QEB/IDX 通过噪声滤波器所需时序

### 6.13.5.2 正交编码接口操作模式

共有 4 种正交编码脉冲计数操作模式

模式 0 : x4 自由计数模式

模式 1 : x2 自由计数模式

模式 2 : x4 比较计数模式

模式 3 : x2 比较计数模式

### 6.13.5.3 自由计数模式

正交编码器脉冲计数器 CNT (QEI\_CNT[31:0]) 按方向标志位 DIRF (QEI\_STATUS[8]) 向上或向下计数。参考图 6.13-6 和图 6.13-7，当计数器溢出或下溢时，会置位 OVUNF (QEI\_STATUS[2])。

### 6.13.5.4 比较计数模式

向上计数还是向下计数取决于方向标志位 DIRF (QEI\_STATUS[8])。向上计数时，OVUNF

(QEI\_STATUS[2]) 会在 CNT (QEI\_CNT[31:0]) 溢出从 CNTMAX (QEI\_CNTMAX[31:0]) 到 0 时置位，对于 x2 计数模式，这发生在下一个 CHA 边沿，而对于 x4 计数模式，这发生 CHA/CHB 边沿。向下计数时，OVUNF (QEI\_STATUS[2]) 会在 CNT (QEI\_CNT[31:0]) 下溢从 0 返回 CNTMAX (QEI\_CNTMAX[31:0]) 时置位，对于 x2 计数模式，这发生在下一个 CHA 边沿，而对于 x4 计数模式，这发生 CHA/CHB 边沿。这些模式可以提供给客户转子的位置。如果正交编码器每秒输出 1024 个脉冲到 CHA，用户在 x4 模式可以配置 QEI\_MAXCNT 和 CNTCMP (QEI\_CNTCMP[31:0]) 为 4095，在 x2 模式则配置为 2047，并在比较计数开始前复位 CNT (QEI\_CNT[31:0])。每当 CNT (QEI\_CNT[31:0]) 从 CNTCMP (QEI\_CNTCMP[31:0]) 溢出，CNTCMP (QEI\_CNTCMP[31:0]) 会被设成和 CNTMAX (QEI\_CNTMAX[31:0]) 一样的值，也就是说转子会在下一个 CHA/CHB 边沿前转完一圈。具体参考图 6.13-6 和图 6.13-7。

#### 6.13.5.5 X4/X2 计数模式

在 x4 计数模式，依据 CHA 和 CHB 的相位关系在每个 CHA 和 CHB 边沿都计数。

QEI x4 计数模式相对于 QEI x2 模式在每个 QEA/QEB 的边缘都计数，频繁的脉冲计数意味着可以提供更高的转子定位精度。设置 QEI 计数模式选择位 MODE (QEI\_CTL[9:8]) 为 00b 或 01b 即为 x4 模式。该模式下，QEI 逻辑检测每个 QEA 和 QEB 的输入边沿。

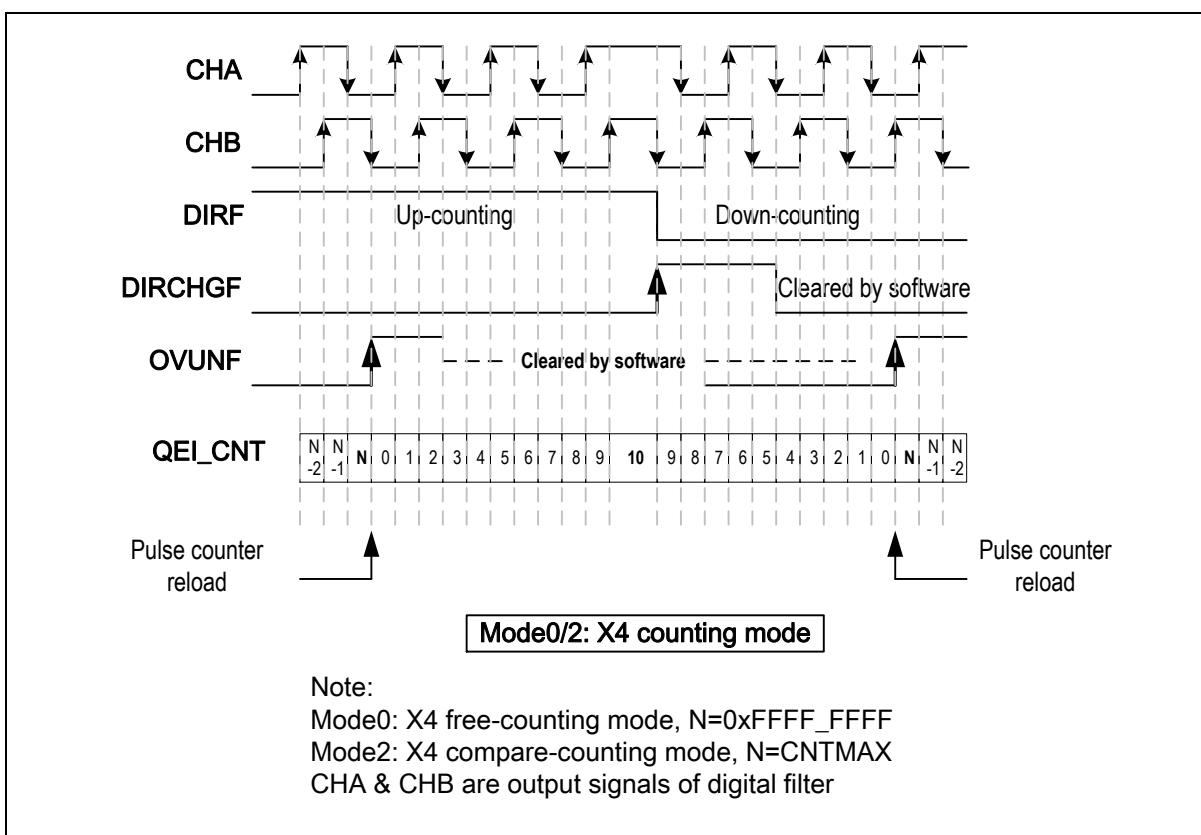


图 6.13-6 X4 计数模式

#### X2 计数模式, 根据CHA和CHB的相位, 在CHA的边沿增或减计数一次

在 x2 计数模式，依据 CHA 和 CHB 的相位关系在每个 CHA 边沿进行计数。设置 QEI 计数模式选择位 MODE (QEI\_CTL[9:8]) 为 01b 或 11b 即为 x2 模式。该模式下，QEI 逻辑只检测每个 QEA 的输入边沿。QEA 上的每个上升沿和下降沿信号都会触发脉冲计数。

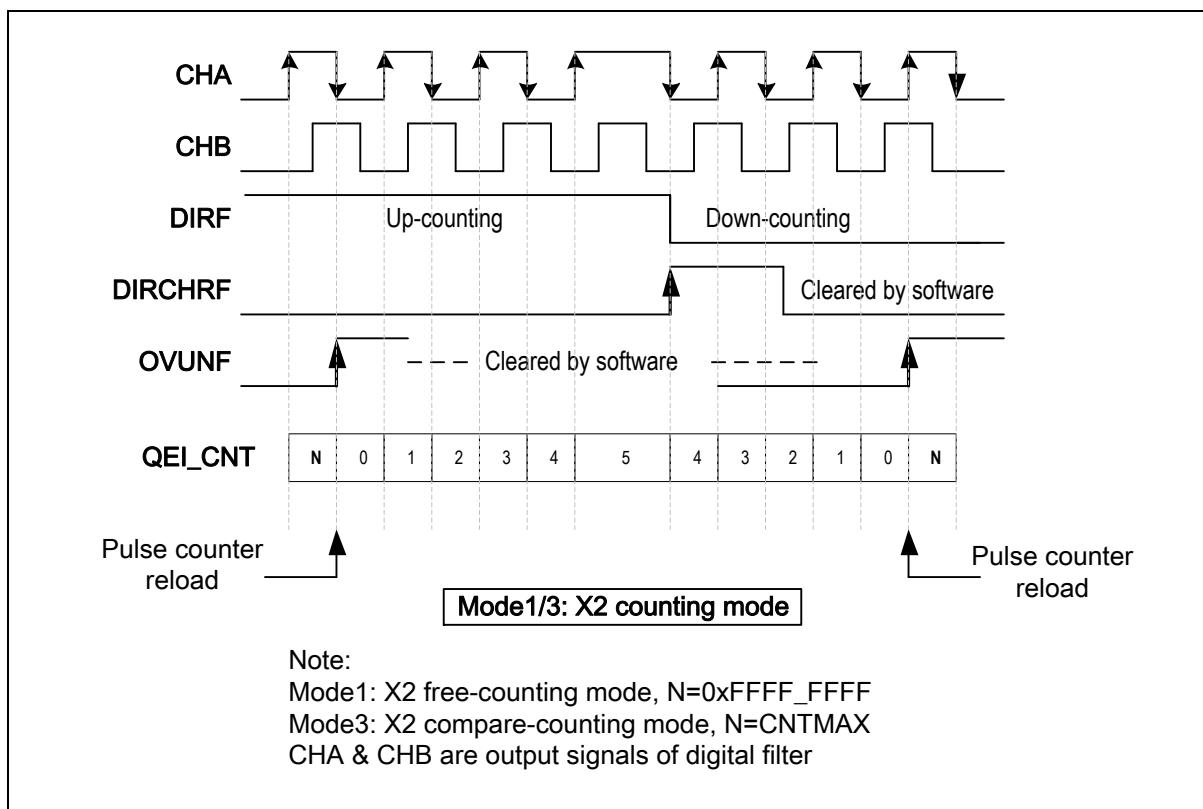


图 6.13-7 X2 计数模式

#### 6.13.5.6 计数方向

CHA 超前 CHB，脉冲信号计数加 1。CHA 落后 CHB，脉冲信号计数减 1。QEI 控制器会产生一个信号来置位 DIRF (QEI\_STATUS[8])，并确定当前计数方向。CHA 超前 CHB 时，DIRF (QEI\_STATUS[8]) 置 1，位置计数器在每个有效的边沿递增计数。CHA 落后 CHB 时，DIRF (QEI\_STATUS[8]) 清 0，位置计数器在每个有效的边沿递减计数。具体参考表 6.13-1。

当前检测到的信号	上一次检测到的信号				DIR (计数方向)	
	上升沿		下降沿			
	CHA	CHB	CHA	CHB		
CHA rising				✓	1 (加)	
		✓			0 (减)	
			✓		触发 (方向变化)	
CHA falling				✓	0 (减)	
		✓			1 (加)	
	✓				触发 (方向变化)	
CHB rising	✓				1 (加)	
			✓		0 (减)	
				✓	触发 (方向变化)	

CHB falling			✓		1 (加)
	✓				0 (减)
		✓			触发 (方向变化)

表 6.13-1 计数方向

### 6.13.5.7 向上计数

向上计数时 DIRF (QEI\_STATUS[8]) 置 1。软件需要先清空 OVUNF (QEI\_STATUS[2]) 标志。对于自由计数模式，CNT (QEI\_CNT[31:0]) 计数到 0xFFFF\_FFFF，在下一个计数方向的边沿置 OVUNF (QEI\_STATUS[2]) 高，再复位 CNT (QEI\_CNT[31:0]) 到 0。对于比较计数模式，CNT (QEI\_CNT[31:0]) 计数到 CNTMAX (QEI\_CNTMAX[31:0])，在下一个计数方向的边沿置 OVUNF (QEI\_STATUS[2]) 高，再复位 CNT (QEI\_CNT[31:0]) 到 0。改变计数方向会触发向下计数，CNT (QEI\_CNT[31:0]) 的计数值开始递减。X2 模式下，只在 CHA 边沿置位 OVUNF (QEI\_STATUS[2])。而 X4 模式下，CHA 和 CHB 边沿都可以置位 OVUNF (QEI\_STATUS[2])。

### 6.13.5.8 向下计数

改变计数方向会触发向下计数，此时 DIRF (QEI\_STATUS[8]) 会清 0 且 DIRCHGF (QEI\_STATUS[3]) 会置为 1，CNT (QEI\_CNT[31:0]) 则会开始向下计数。对于自由计数模式，向下计数到 0 时，脉冲计数器会重载为 0xFFFF\_FFFF，OVUNF (QEI\_STATUS[2]) 会在下一个边沿置 1。对于比较计数模式，向下计数到 0 时，脉冲计数器会重载为 CNTMAX (QEI\_CNTMAX[31:0])，OVUNF (QEI\_STATUS[2]) 会在下一个边沿置 1。X2 模式下，只在 CHA 边沿置位 OVUNF (QEI\_STATUS[2])。而 X4 模式下，CHA 和 CHB 边沿都可以置位 OVUNF (QEI\_STATUS[2])。

### 6.13.5.9 比较功能

QEI 控制器通过比较 CNT (QEI\_CNT[31:0]) 和比较寄存器 CNTCMP (QEI\_CNTCMP[31:0]) 实现动态计数。CNT (QEI\_CNT[31:0]) 向上/下计数到 CNTCMP (QEI\_CNTCMP[31:0])，CMPPF (QEI\_STATUS[1]) 置位。写 1 到 CMP\_EN (QEI\_CTL[28]) 使能比较功能，写 0 则禁止该功能。

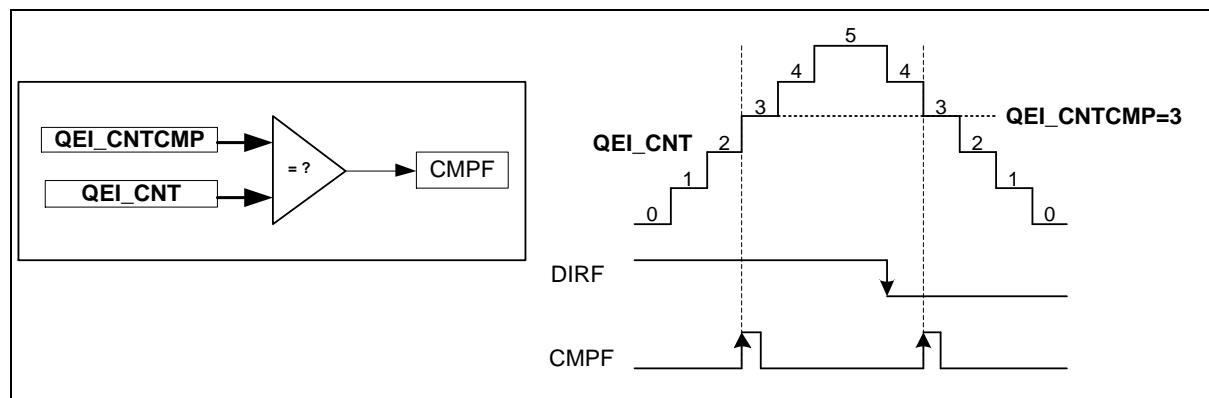


图 6.13-8 比较操作

### 6.13.5.10 通过 IDX 管脚重载计数器

CHX 信号 (来自滤波器和极性控制输出管脚 IDX) 可以触发 CNT (QEI\_CNT[31:0]) 复位为零或是重载为 CNTMAX (QEI\_CNTMAX[31:0])。计数器向上计数时，置位 IDX 重载位 IDXRLD\_EN (QEI\_CTL[27])，CHX 上的上升沿会导致 QEI 控制器复位 CNT (QEI\_CNT[31:0]) 到 0。计数器向下计数时，CHX 上的上升沿会导致 QEI 控制器重载 CNT (QEI\_CNT[31:0]) 为 CNTMAX (QEI\_CNTMAX[31:0])。具体参考图 6.13-9。

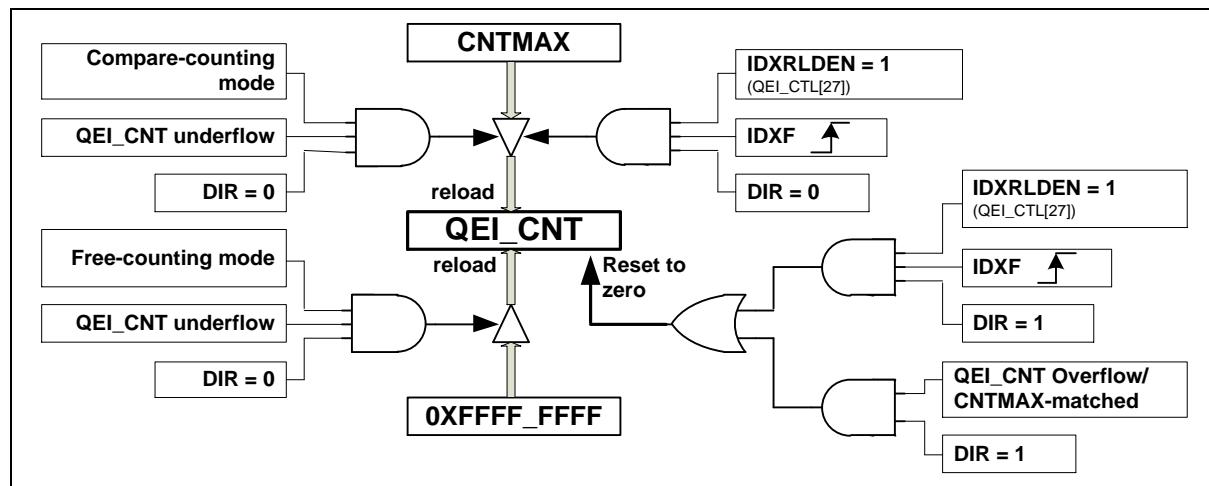


图 6.13-9 QEI\_CNT 重载/复位控制

#### 6.13.5.11 捕捉 QEI 计数器

如果 HOLDcnt (QEI\_CTL[24]) 置位，CNT (QEI\_CNT[31:0]) 的值将会被捕捉到 QEI 计数器保持寄存器 CNTHold (QEI\_CNTHold[31:0])，直到下一个 HOLDcnt (QEI\_CTL[24]) 触发信号。HOLDcnt 位可以由软件写 1 或者用定时器上升沿中断标志位 TIF (TIMERx\_INTSTS[0]) 置位。

**注:** CNTHold (QEI\_CNTHold[31:0]) 捕捉到 QEI 计数器的值后，HOLDcnt 由硬件自动清 0。

如果 IDXLATEN (QEI\_CTL[25]) 置位，CNT (QEI\_CNT[31:0]) 的内容会在每个 CHX 上升沿信号被锁存到 QEI 计数器索引锁存寄存器 CNTLATCH (QEI\_CNTLATCH[31:0])

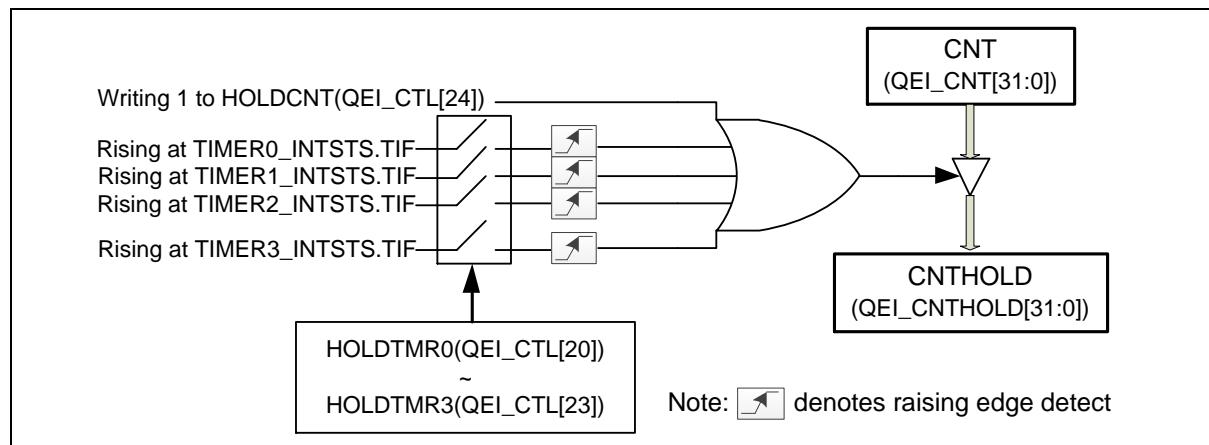


图 6.13-10 捕捉 QEI 计数器的触发控制

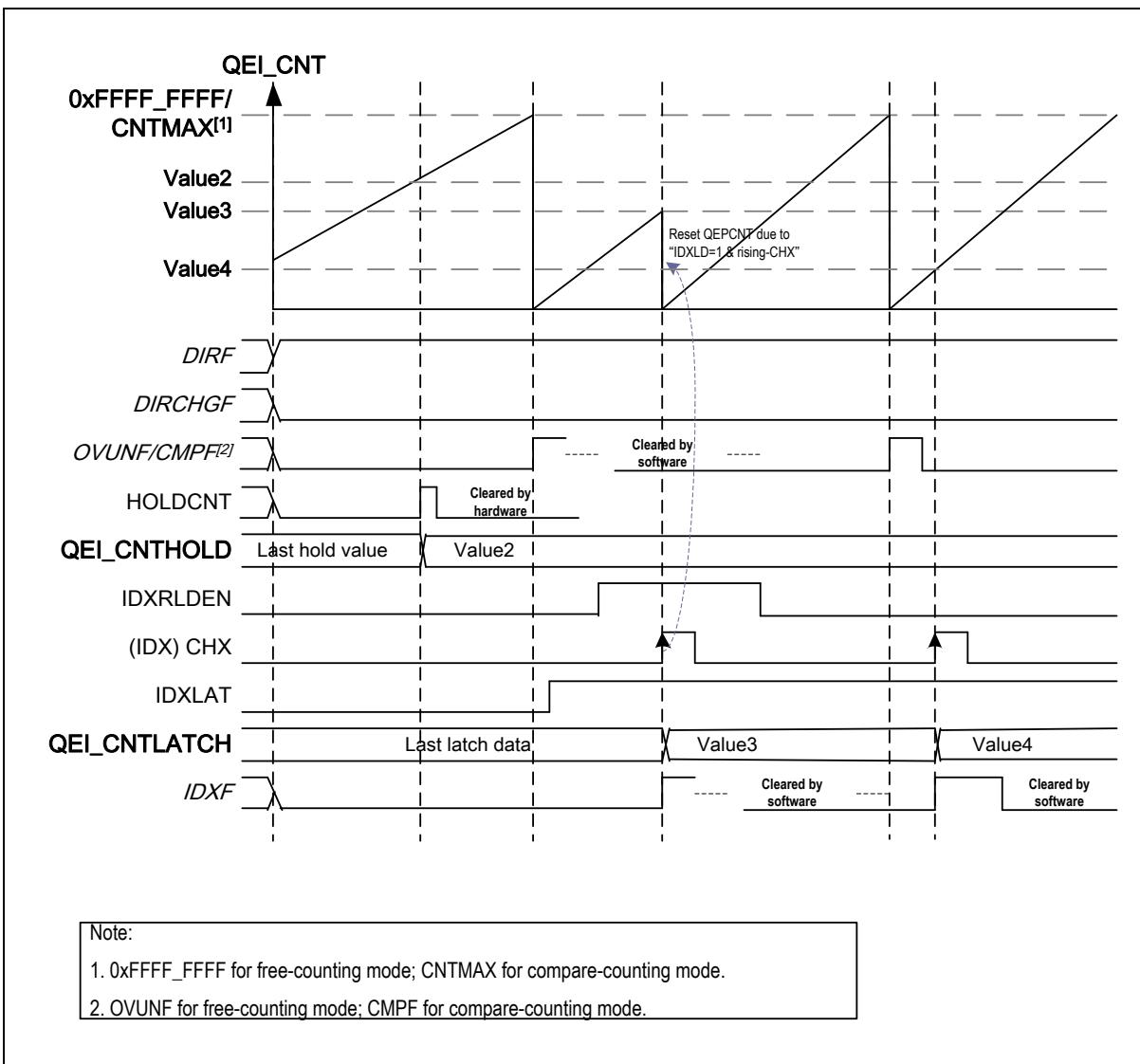


图 6.13-11 捕捉和锁存 QEI 计数器

#### 6.13.5.12 QEI 中断结构

QEI 控制器有 4 个中断触发源，每个都有对应的中断标志和使能控制位。当 QEI 计数器向上计数 CNT (QEI\_CNT[31:0]) 溢出，或者向下计数 CNT (QEI\_CNT[31:0]) 下溢时，溢出/下溢标志 OVUNF (QEI\_STATUS[2]) 会由硬件置位，如果此时 OVUNIEN (QEI\_CTL[16]) 为高，就会触发 QEI 中断请求。当 QEI 控制器检测到了编码器换向时，会触发方向标志 DIRF (QEI\_STATUS[8])，硬件会自动置位换向标志 DIRCHGF (QEI\_STATUS[3])，如果此时 DIRIEN (QEI\_CTL[17]) 位高，就会触发 QEI 中断请求。当 QEI 计数器计数到 QEI 计数器比较寄存器 CNTCMP (QEI\_CNTCMP[31:0])，硬件会自动置位 CMPF (QEI\_STATUS[1])，如果此时 CMPIEN (QEI\_CTL[18]) 位高，就会触发 QEI 中断请求。当 QEI 控制器检测到了 CHX (管脚 IDX 经过滤波器和极性控制的输出) 上的上升沿信号，硬件会自动置位 IDXF (QEI\_STATUS[1])，如果此时 IDXIEN (QEI\_CTL[19]) 位高，就会触发 QEI 中断请求。需要注意，OVUNF (QEI\_STATUS[2]), DIRCHGF (QEI\_STATUS[3]), CMPF (QEI\_STATUS[1]) 和 IDXF (QEI\_STATUS[0]) 这四个标志由硬件自动置位，必须由软件清 0。如图 6.13-12 是 QEI 控制器的中断架构。

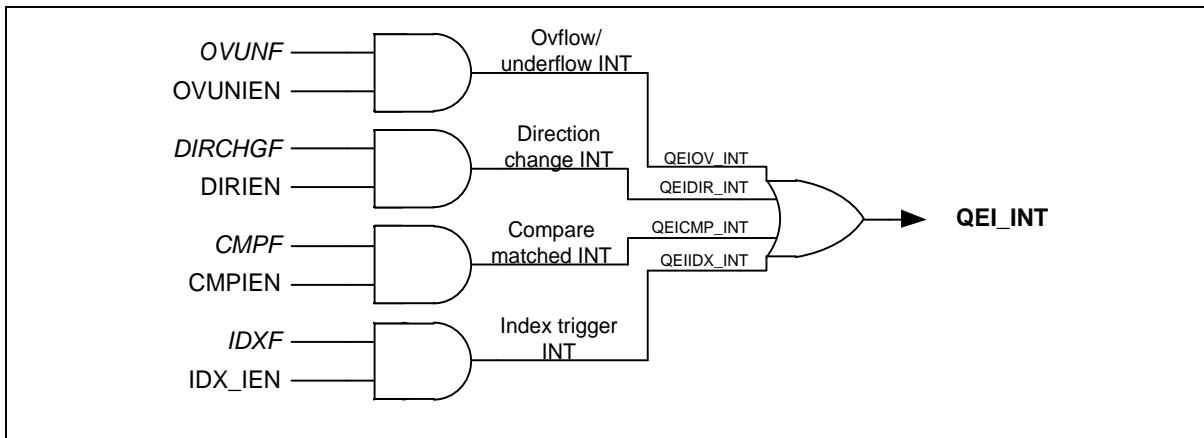


图 6.13-12 QEI 中断架构图

### 6.13.6 寄存器映射

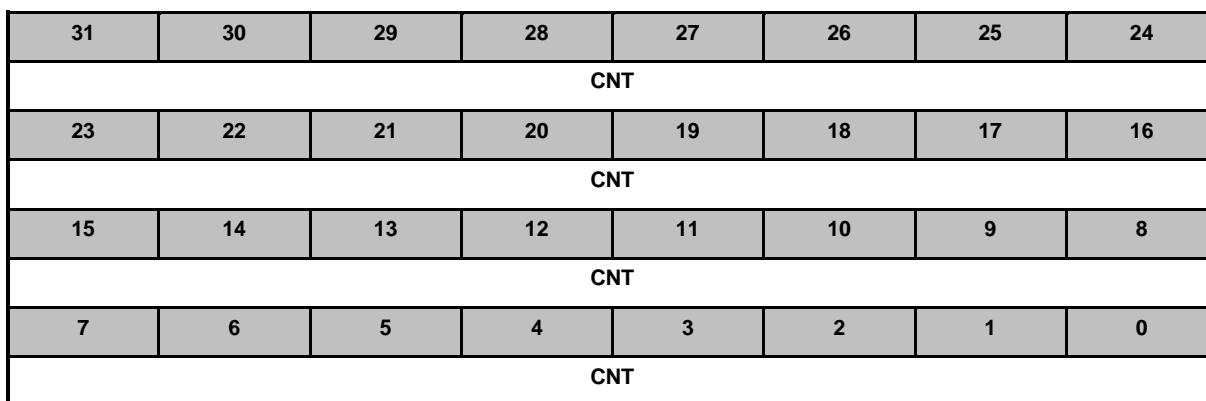
R: 只读, W: 只写, R/W: 读/写

寄存器	偏移量	R/W	描述	复位值
<b>QEI 基地址:</b>				
<b>QEI0_BA = 0x400B_0000</b>				
<b>QEI1_BA = 0x400B_1000</b>				
QEI_CNT <i>x=0, 1</i>	QEIx_BA+0x00	R/W	QEI 计数值寄存器	0x0000_0000
QEI_CNTHOLD <i>x=0, 1</i>	QEIx_BA+0x04	R/W	QEI 计数器保持寄存器	0x0000_0000
QEI_CNTLATCH <i>x=0,1</i>	QEIx_BA+0x08	R/W	QEI 计数器索引锁存寄存器	0x0000_0000
QEI_CNTCMP <i>x=0, 1</i>	QEIx_BA+0x0C	R/W	QEI 计数器比较寄存器	0x0000_0000
QEI_CNTMAX <i>x=0, 1</i>	QEIx_BA+0x14	R/W	QEI 预设最大计数值寄存器	0x0000_0000
QEI_CTL <i>x=0, 1</i>	QEIx_BA+0x18	R/W	QEI 控制器控制寄存器	0x0000_0000
QEI_STATUS <i>x=0, 1</i>	QEIx_BA+0x2C	R/W	QEI 控制器状态寄存器	0x0000_0000

## 6.13.7 寄存器描述

QEI\_CNT QEI 计数值寄存器

寄存器	偏移量	R/W	描述	复位值
QEI_CNT	QEIx_BA+0x00	R/W	QEI Counter Register	0x0000_0000



位	描述
[31:0]	<b>CNT</b> <b>QEI 计数值</b> 32 位的向上/下计数器。当检测到有效的脉冲相位, DIRF (QEI_STATUS[8]) 为 1, 则计数值递增; DIRF (QEI_STATUS[8]) 为 0, 则计数值递减。该寄存器通过积分的方式来计算当前的编码器位置。 下列事件会导致脉冲计数器恢复初始化: 1. QEIEN (QEI_CTL[29]) = 0 时, 软件可写 2. QEIEN (QEI_CTL[29]) = 1 且比较计数模式下, 计数值与比较值相等时 (上计数时, 值回 0) 3. QEIEN=1 且 IDXRLDEN (QEI_CTL[27])=1 时 IDX 信号发生变化

QEI\_CNTHOLD QEI 计数器保持寄存器

寄存器	偏移量	R/W	描述	复位值
QEI_CNTHOLD	QEIx_BA+0x04	R/W	QEI Counter Hold Register	0x0000_0000

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

位	描述	
[31:0]	<b>CNTHOLD</b>	QEI 计数器保持值 HOLDCNT (QEI_CTL[24]) 由低到高时, CNT (QEI_CNT[31:0]) 的值会被复制到 CNTHOLD (QEI_CNTHOLD[31:0]) 寄存器。

QEI\_CNTLATCH QEI 计数器索引锁存寄存器

寄存器	偏移量	R/W	描述	复位值
QEI_CNTLATCH	QEIx_BA+0x08	R/W	QEI Counter Index Latch Register	0x0000_0000

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

位	描述	
[31:0]	CNTLATCH	QEI 计数器索引锁存 当 IDXF (QEI_STATUS[0]) 置位, CNT (QEI_CNT[31:0]) 的值会被复制到 CNTLATCH (QEI_CNTLATCH[31:0]) 寄存器。

QEI\_CNTCMP QEI 计数器比较寄存器

寄存器	偏移量	R/W	描述	复位值
QEI_CNTCMP	QEIx_BA+0x0C	R/W	QEI Counter Compare Register	0x0000_0000

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

位	描述	
[31:0]	CNTCMP	QEI 计数器比较值 如果 QEI 控制器处于比较计数模式 CMPEN (QEI_CTL[28]) =1, 当 CNT (QEI_CNT[31:0]) 的值计数到 CNTCMP (QEI_CNTCMP[31:0]), CMPF 会被置位。该寄存器软件可写。

QEI\_CNTMAX QEI 预设最大计数值寄存器

寄存器	偏移量	R/W	描述	复位值
QEI_CNTMAX	QEIx_BA+0x14	R/W	QEI Pre-set Maximum Count Register	0x0000_0000

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

位	描述	
[31:0]	CNTMAX	QEI 预设最大计数值 该寄存器由用户设定，是比较计数模式下的最大值。

QEI\_CTL QEI 控制器控制寄存器

寄存器	偏移量	R/W	描述	复位值
QEI_CTL	QEIx_BA+0x18	R/W	QEI Controller Control Register	0x0000_0000

31	30	29	28	27	26	25	24
	Reserved	QEIEN	CMPEN	IDXRLDEN	Reserved	IDXLATEN	HOLDCNT
23	22	21	20	19	18	17	16
HOLDTMR3	HOLDTMR2	HOLDTMR1	HOLDTMR0	IDXIEN	CMPIEN	DIRIEN	OVUNIEN
15	14	13	12	11	10	9	8
Reserved	IDXINV	CHBINV	CHAINV	Reserved		MODE	
7	6	5	4	3	2	1	0
Reserved	IDXEN	CHBEN	CHAEN	NFDIS		NFCLKSEL	

位	描述	
[31:30]	Reserved	保留
[29]	QEIEN	<b>QEI 控制器使能位</b> 0 = 禁止QEI 控制器功能 1 = 使能QEI 控制器功能
[28]	CMPEN	<b>比较功能使能位</b> QEI 比较功能用于动态比较 QEI_CNT 和 CNTCMP ( QEI_CNTCMP[31:0] )，如果 CNT (QEI_CNT[31:0]) 计数到 CNTCMP ( QEI_CNTCMP[31:0] )，该位 CMPF 置 1 0 = 禁止比较功能。 1 = 使能比较功能。
[27]	IDXRLDEN	<b>索引触发 QEI_CNT 重载使能位</b> 向上计数时 (DIRF (QEI_STATUS[8]) = 1)，若该位为高，检测到 CHX 的上升沿时会复位 CNT (QEI_CNT[31:0]) 为 0。向下计数时 (DIRF (QEI_STATUS[8]) = 0)，若该位为高，检测到 CHX 的上升沿时会重载 CNT (QEI_CNT[31:0]) 为 CNTMAX (QEI_CNTMAX[31:0])。 0 = 禁止重载功能。 1 = 使能索引信号初始化 QEI_CNT 功能。
[26]	Reserved	保留
[25]	IDXLATEN	<b>索引锁存 QEI_CNT 使能位</b> 该位置位时，CHX 信号的每个上升沿都会锁存 CNT (QEI_CNT[31:0]) 的值到 CNTLATCH (QEI_CNTLATCH[31:0])。 0 = 禁止索引信号锁存 QEI 计数器值功能 1 = 使能索引信号锁存 QEI 计数器值功能

[24]	<b>HOLDCNT</b>	<b>保持 QEI_CNT 控制</b> 当该位由低到高时，CNT (QEI_CNT[31:0]) 会被复制到 CNTHOLD (QEI_CNTHOLD[31:0])。写 1 到该位或是 Timer0~Timer3 中断标志 TIF (TIMERx_INTSTS[0]) 会置位该位。 0 = 无操作。 1 = QEI_CNT 值会被捕捉和存储到 CNTHOLD (QEI_CNTHOLD[31:0]). 注: QEI_CNTHOLD 保持 QEI_CNT 后，该位自动清 0
[23]	<b>HOLDTMR3</b>	<b>Timer3 保持功能使能位</b> 0 = TIF (TIMER3_INTSTS[0]) 不影响 HOLDCNT。 1 = TIF (TIMER3_INTSTS[0]) 置 1 时，会把 HOLDCNT 置为1。
[22]	<b>HOLDTMR2</b>	<b>Time 2 保持功能使能位</b> 0 = TIF (TIMER2_INTSTS[0]) 不影响 HOLDCNT. 1 = TIF (TIMER2_INTSTS[0]) 置 1 时，会把 HOLDCNT 置为1。
[21]	<b>HOLDTMR1</b>	<b>Timer1 保持功能使能位</b> 0 = TIF (TIMER1_INTSTS[0]) 不影响 HOLDCNT. 1 = TIF (TIMER1_INTSTS[0]) 置 1 时，会把 HOLDCNT 置为1。
[20]	<b>HOLDTMR0</b>	<b>Timer0 保持功能使能位</b> 0 = TIF (TIMER0_INTSTS[0]) 不影响 HOLDCNT. 1 = TIF (TIMER0_INTSTS[0]) 置 1 时，会把 HOLDCNT 置为1。
[19]	<b>IDXIEN</b>	<b>IDXF 触发 QEI 中断使能位</b> 0 = 关 IDX 中断 1 = 开 IDX 中断
[18]	<b>CMPFIEN</b>	<b>CMPF 触发 QEI 中断使能</b> 0 = 关 CMPF 中断 1 = 开 CMPF 中断.
[17]	<b>DIRIEN</b>	<b>DIRCHGF 触发 QEI 中断使能</b> 0 = 关 DIRCHGF 中断 1 = 开 DIRCHGF 中断.
[16]	<b>OVUNIEN</b>	<b>OVUNF 触发 QEI 中断使能</b> 0 = 关 OVUNF 中断 1 = 开 OVUNF 中断
[15]	<b>Reserved</b>	保留
[14]	<b>IDXINV</b>	<b>IDX 反相</b> 0 = IDX 不反相 1 = IDX 反相
[13]	<b>CHBINV</b>	<b>QEB 反相</b> 0 = 不反相 1 = QEB 反相
[12]	<b>CHAINV</b>	<b>QEA 反相</b> 0 = 不反相 1 = QEA 反相

[11:10]	<b>Reserved</b>	保留
[9:8]	<b>MODE</b>	<b>QEI</b> 计数模式选择 00 = X4 自由计数模式 01 = X2 自由计数模式 10 = X4 比较计数模式 11 = X2 比较计数模式
[7]	<b>Reserved</b>	保留
[6]	<b>IDXEN</b>	<b>IDX</b> 输入使能 0 = 关 IDX 输入 1 = 开通 IDX 输入
[5]	<b>CHBEN</b>	<b>QEB</b> 输入使能 0 = 关 QEB 输入 1 = 开通 QEB 输入
[4]	<b>CHAEN</b>	<b>QEA</b> 输入使能 0 = 关 QEA 输入 1 = 开通 QEA 输入
[3]	<b>NFDIS</b>	<b>QEI</b> 噪声滤波器禁止位 0 = 使能噪声滤波功能 1 = 禁噪声滤波.
[2:0]	<b>NFCLKSEL</b>	噪声滤波器时钟分频选择 000 = QEI_CLK. 001 = QEI_CLK/2. 010 = QEI_CLK/4. 011 = QEI_CLK/16. 100 = QEI_CLK/32. 101 = QEI_CLK/64.

**QEI STATUS QEI 控制器状态寄存器**

寄存器	偏移量	R/W	描述	复位值
QEI_STATUS	QEIx_BA+0x2C	R/W	QEI Controller Status Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved				DIRCHGF	OVUNF	CMPF	IDXF

位	描述	
[31:9]	<b>Reserved</b>	保留
[8]	<b>DIRF</b>	<b>QEI 计数方向指示位</b> 0 = 向下计数 1 = 向上计数 <b>注:</b> 硬件按 CHA 和 CHB 的超前或滞后, 决定此位的值。
[7:4]	<b>Reserved</b>	保留
[3]	<b>DIRCHGF</b>	<b>方向改变标志</b> 计数方向改变时此位由硬件自动置 1。软件写 1 可清 0。 0 = 计数方向未改变。 1 = 计数方向已改变。 <b>注:</b> 该位写 1 清 0.
[2]	<b>OVUNF</b>	<b>QEI 溢出/下溢标志</b> 自由计数模式计数值 QEI_CNT 从 0xFFFF_FFFF 变 0 时, 比较计数模式从 QEI_CNTMAX 变 0 时; 或反之, 从 0 变为 0xFFFF_FFFF 或 QEI_CNTMAX 时, 此位置 1。 0 = QEI 计数器未溢出/下溢。 1 = QEI 计数器已溢出/下溢。 <b>注:</b> 该位写 1 清 0.
[1]	<b>CMPF</b>	<b>比较匹配标志</b> 如果 QEI 比较功能使能, QEI 计数器向上/下计数到 CNTCMP (QEI_CNTCMP[31:0]), 该位由硬件自动置位。 0 = QEI 未计数到 CNTCMP (QEI_CNTCMP[31:0]). 1 = QEI 计数到 CNTCMP (QEI_CNTCMP[31:0]). <b>注:</b> 该位写 1 清 0.

[0]	IDXF	<b>IDX 检测标志</b> CHX 信号上检测到上升沿，该标志置位 0 = CHX 信号上检未测到上升沿。 1 = CHX 信号上检测到上升沿。 注：该位写 1 清 0.
-----	------	---

## 6.14 ECAP 增强型输入捕捉定时器

### 6.14.1 概述

M480 系列提供了多达两组的增强型输入捕捉定时器/计数器，以用于检测输入通道边沿信号的改变。每个单元含有三路输入通道。定时器/计数器有向上计数、重载及比较匹配的功能。

### 6.14.2 特性

- 多达两组输入捕捉定时器/计数器单元，CAP0 和 CAP1.
- 每个单元都有 3 个输入通道
- 每个单元都有独立的中断向量.
- 每个输入通道都有对应的捕捉计数器保存寄存器.
- 24 位的输入捕捉向上计数定时器/计数器
- 在输入端末前附有噪声过滤
- 三种边沿检测：
  - 上升沿检测
  - 下降沿检测
  - 双边沿检测
- 支持用被捕获的事情复位/重载捕捉计数器
- 支持比较匹配功能

### 6.14.3 框图

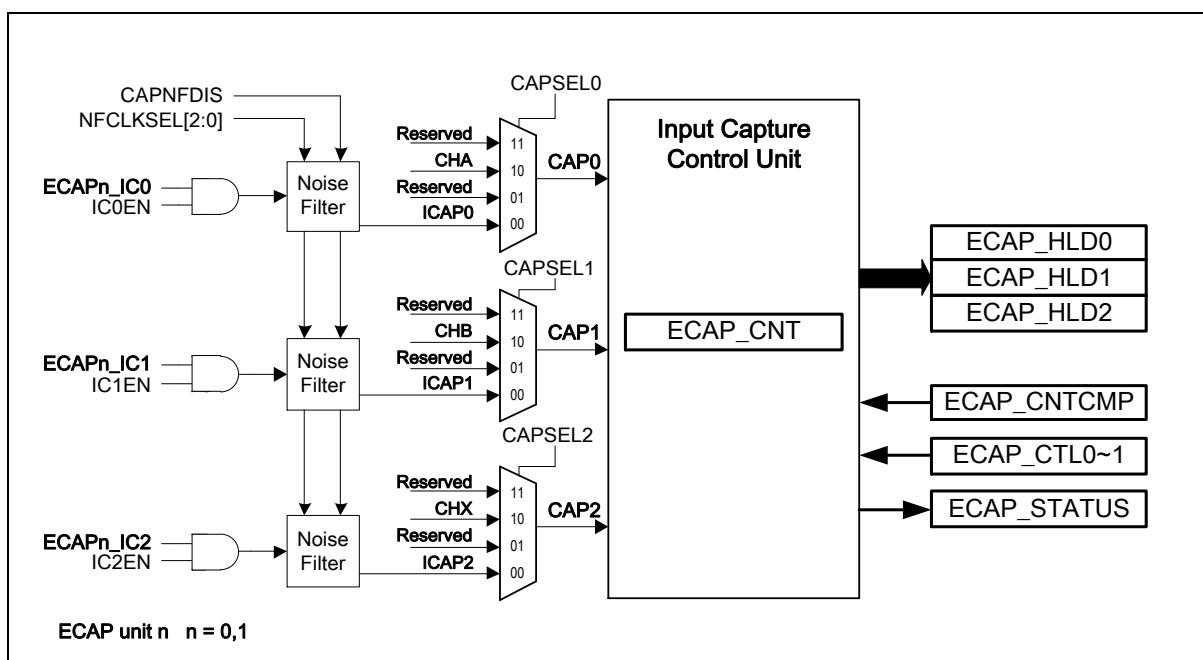


图 6.14-1 输入捕捉定时器/计数器架构

#### 6.14.4 基本配置

##### 6.14.4.1 ECAP0 基本配置

- 时钟源配置
  - 在寄存器 ECAP0CKEN (CLK\_APBCLK1[26]) 使能 ECAP0 外设时钟
- 复位配置
  - 在寄存器 ECAP0RST (SYS\_IPRST2[26]) 复位 ECAP0
- 管脚配置

Group	Pin Name	GPIO	MFP
ECAP0	ECAP0_IC0	PA.10	MFP11
		PE.8	MFP12
	ECAP0_IC1	PA.9	MFP11
		PE.9	MFP12
	ECAP0_IC2	PA.8	MFP11
		PE.10	MFP12

##### 6.14.4.2 ECAP1 基本配置

- 时钟源配置
  - 在寄存器 ECAP1CKEN (CLK\_APBCLK1[27]) 使能 ECAP1 外设时钟
- 复位配置
  - 在寄存器 ECAP0RST (SYS\_IPRST2[27]) 复位 ECAP1
- 管脚配置

Group	Pin Name	GPIO	MFP
ECAP1	ECAP1_IC0	PC.10	MFP11
		PE.13	MFP13
	ECAP1_IC1	PC.11	MFP11
		PE.12	MFP13
	ECAP1_IC2	PC.12	MFP11
		PE.11	MFP13

### 6.14.5 功能描述

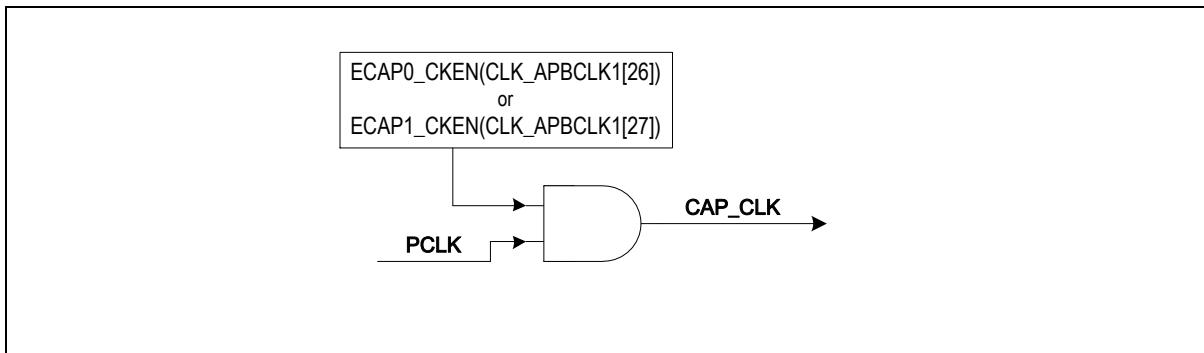


图 6.14-2 输入捕捉定时器/计数器时钟源控制

图 6.14-1 是输入捕捉的架构图。每个输入捕捉定时器/计数器单元支持三个输入信号源可编程的输入通道。管脚 ECAP\_IC0 到 ECAP\_IC2 可以通过噪声滤波器或者直接 (CAPNFDIS = 1) 作为输入捕捉单元的信号源，此外时，QEI 控制器的输入信号 (CHA, CHB, CHX) 也可以根据软件设置 ECAP\_CTL0 (CAPSEL0~ CAPSEL 2) 从内部通路连接到捕捉输入。

#### 6.14.5.1 输入噪声滤波器

输入噪声滤波器的结构类似 QEI 的输入滤波器。可以参考 QEI 的相关介绍。ECAP 有 6 个采样率选择，PCLK 运行在 90 MHz 时，ECAP 支持从 33 ns (NFCLKSEL = 000) 到 2.1 us (NFCLKSEL = 101) 的持续时间选择。表 6.14-1 列出了 NFCLKSEL、最大噪声持续时间和最小能被采样的信号持续时间的相关设置。

最大可以被滤掉的噪声持续时间不超过 3 个采样时钟周期，最小可以被采样的信号持续时间需超过 4 个采样时钟周期。

NFCLKSEL	Maximum Duration Of Noise	Minimum Duration Of Signal
000	33 ns	44 ns
001	67 ns	89 ns
010	133 ns	178 ns
011	533 ns	711 ns
100	1067 ns	1422 ns
101	2133 ns	2844 ns
PCLK= 90 MHz		

表 6.14-1 一个典型的滤波设置

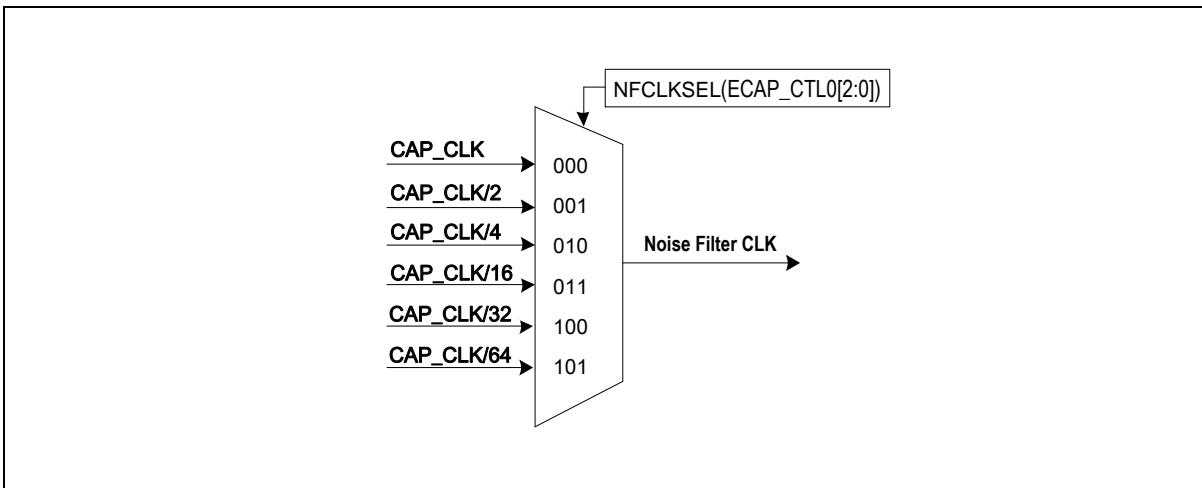


图 6.14-3 噪声滤波器采样时钟选择

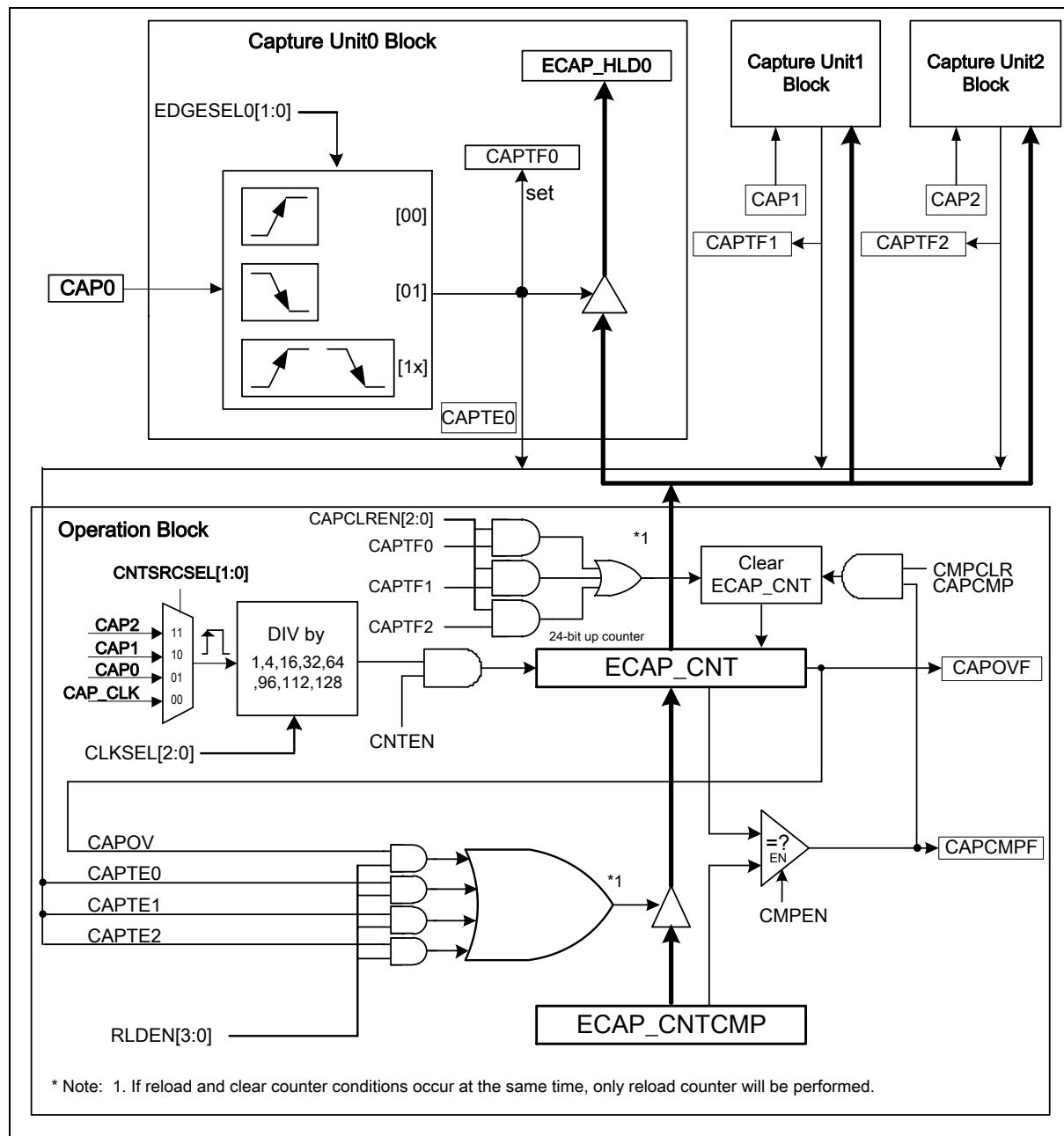


图 6.14-4 输入捕捉定时器/计数器功能框图

#### 6.14.5.2 输入捕捉定时器/计数器操作

输入捕捉定时器/计数器单元包含 2 个主要的功能模块：捕捉模块和操作模块。3 路捕捉通道共有 3 个捕捉单元。

捕捉单元功能检测脉冲宽度和方波周期。输入通道 0~2 在捕捉模块有他们自己的检测器，但是在操作模块里共享一个捕捉定时器/计数器，ECAP\_CNT。边沿触发选项由 EDGESEL (ECAP\_CTL1[5,4], [3,2], [1,0]) 寄存器配置，可以设成上升沿、下降沿和双边沿触发。每个捕捉单元包含一个使能控制位，IC0EN ~ IC2EN (ECAP\_CTL0[6:4])，以使能/禁止每个输入通道和状态位 CAP0 ~ CAP2 (ECAP\_STATUS[10:8])，从而让软件监控当前各通道的状态。

输入捕捉支持重载模式和比较模式。两个模式里，捕捉计数器 (ECAP\_CNT) 都可以作为 24 位的向上计

数器，其时钟分频可通过 CLKSEL[2:0] 设为除 1,4,16,32,64,96,112 和 128，时钟源则可以通过 CNTSRCSEL[1:0] 设置为系统时钟源、CAP\_CLK 或者输入通道 CAP0 ~ CAP2。而在重载模式里，ECAP\_CNTCMP 作为重载寄存器，比较模式里 ECAP\_CNTCMP 则作为比较寄存器。输入捕捉定时器/计数器使能位 CAPEN 在输入捕捉定时器/计数器功能使用时一定要使能。更多操作细节参见后文。

### 捕获功能

每当检测到有效的边沿改变信号时，就会触发捕捉事件 (CAPTE0~2)，工作中的 24 位捕捉计数器的值 ECAP\_CNT 会被捕捉并传送到被触发通道的捕捉暂存寄存器 ECAP\_HOLD0~2 中。触发动作还会使 CAPTFx (ECAP\_STATUS[2:0]) 标志位置位，并产生一个中断信号 (当 CAPIENx (ECAP\_CTL0[18:16]) 使能时)。触发标志由硬件置位，软件清 0。软件可以去读寄存器 ECAP\_STATUS 来读取标志的状态，再写 1 到 ECAP\_STATUS 对应位去清 0 标志。

设置 CAPxCLREN (ECAP\_CTL1[22:20]) 位后，在发生捕捉事件后会硬件会自动复位捕捉计数器 ECAP\_CNT。

### 比较模式

置 CMPPEN (ECAP\_CTL0[28]) 位为 1 将会使能比较模式。此时，ECAP\_CNTCMP 将作为比较寄存器。随着 ECAP\_CNT 递增，一旦达到了 ECAP\_CNTCMP 的值，CMPPF (ECAP\_STATUS[4]) 将被置位，同时若使能了捕捉比较中断位 CMPIEN (ECAP\_CTL0[21]) 将会产生一个中断请求。

设置 CMPCLR (ECAP\_CTL0[25]) 位会让硬件在比较-匹配成功后自动复位捕捉计数器。

### 重载模式

输入捕捉定时器/计数器还可以配置成重载模式。置 RLDEN[3:0] (ECAP\_CTL1[11:8]) 位为 1，也就是 OVRLDEN, CAPxRLDEN 为 1 将会使能重载源。

该模式下，ECAP\_CNTCMP 作为重载寄存器。ECAP\_CNT 溢出后，如果已置位 OVRLDEN，将产生重载信号使得 ECAP\_CNTCMP 寄存器的值重载到 ECAP\_CNT 寄存器里；另外 CAPxRLDEN, x=0~2 可以配置 CAPTE2 ~ CAPTE0 为重载源。

还需要注意的是如果 CAPxCLREN 和 CAPxRLDEN 都置位了，当触发事件 CAPTE<sub>x</sub> 到达，只有 RELAOD 功能会被执行。

#### 6.14.5.3 输入捕捉定时器/计数器中断架构

图 6.14-5 是输入捕捉定时器/计数器中断架构示例。总共有 5 个中断源 (OVF\_INT, CMP\_INT, CAPTF0\_INT~CAPTF2\_INT)，共同处于一个输入捕捉单元，每个都有一个中断标志 (CAPOVF, CAPCMPPF, CAPTF0~CAPTF2) 来触发中断 (ECAP\_INT)。可以用使能控制位 (OVIEN, CMPIEN, CAPIEN0~CAPIEN2) 来使能/禁止这些标志。

需要注意的是所有的中断标志都由硬件自动置位，必须由软件写 1 到 (ECAP\_STATUS[5:4],[2:0]) 清 0 对应标志。

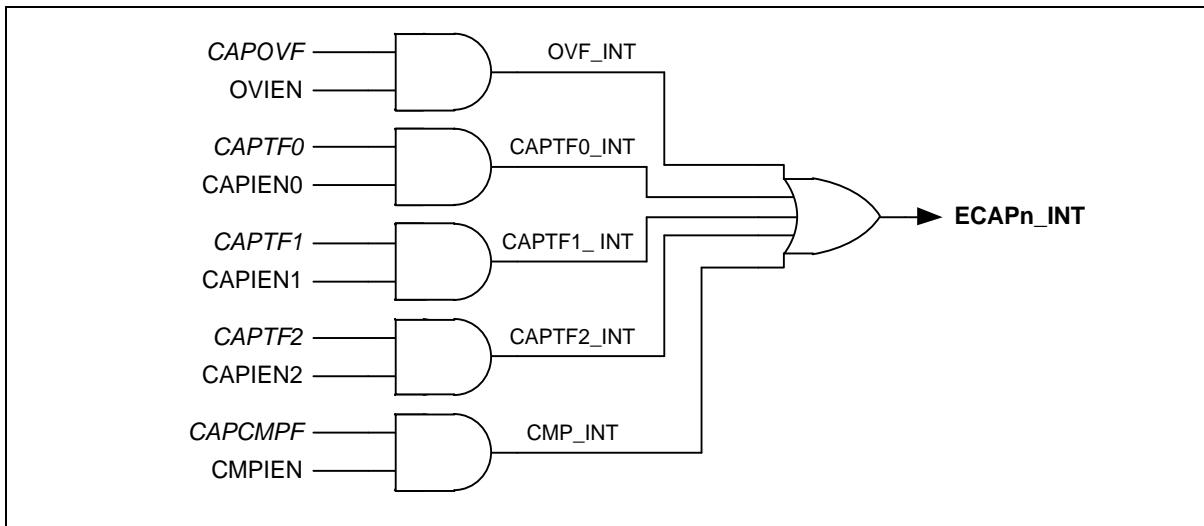


图 6.14-5 输入捕捉定时器/计数器中断架构图

### 6.14.6 寄存器映射

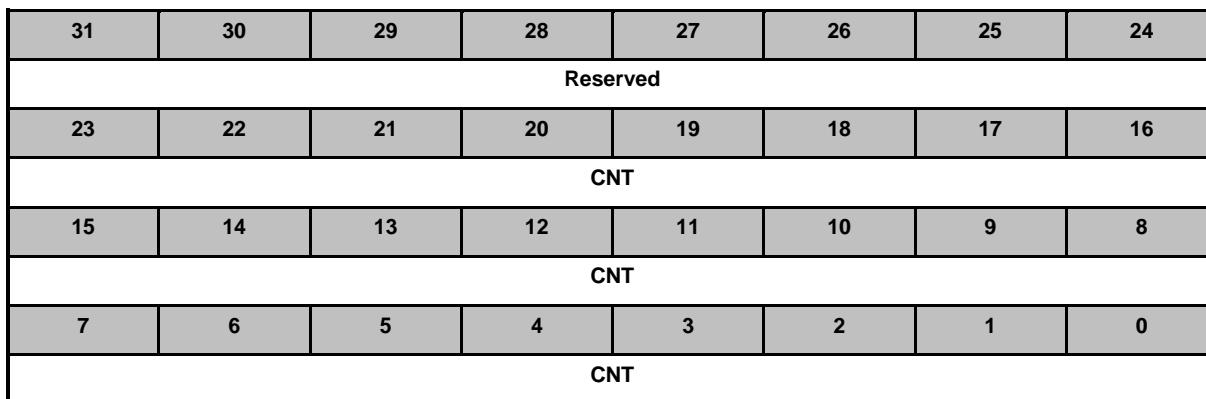
R: 只读, W: 只写, R/W: 读/写

寄存器	偏移量	R/W	描述	复位值
<b>ECAP 基地址:</b> <b>ECAPn_BA = 0x400B_4000 + (0x0000_1000 * n)</b> n=0, 1				
<b>ECAP_CNT</b>	ECAPn_BA+0x00	R/W	输入捕捉计数器 (24 位向上计数)	0x0000_0000
<b>ECAP_HLD0</b>	ECAPn_BA+0x04	R/W	输入捕捉保持寄存器 0	0x0000_0000
<b>ECAP_HLD1</b>	ECAPn_BA+0x08	R/W	输入捕捉保持寄存器 1	0x0000_0000
<b>ECAP_HLD2</b>	ECAPn_BA+0x0C	R/W	输入捕捉保持寄存器 2	0x0000_0000
<b>ECAP_CNTCMP</b>	ECAPn_BA+0x10	R/W	输入捕捉比较寄存器	0x0000_0000
<b>ECAP_CTL0</b>	ECAPn_BA+0x14	R/W	输入捕捉控制寄存器 0	0x0000_0000
<b>ECAP_CTL1</b>	ECAPn_BA+0x18	R/W	输入捕捉控制寄存器 1	0x0000_0000
<b>ECAP_STATUS</b>	ECAPn_BA+0x1C	R/W	输入捕捉状态寄存器	0x0000_0000
<b>ECAP_VERSION</b>	ECAPn_BA+0xFFC	R	ECAP 版本控制寄存器	0x0202_0000

### 6.14.7 寄存器描述

**ECAP\_CNT 捕获计数器**

寄存器	偏移量	R/W	描述	复位值
ECAP_CNT	ECAPn_BA+0x00	R/W	输入捕捉计数器 (24 位向上计数)	0x0000_0000



位	描述	
[31:24]	Reserved	保留
[23:0]	CNT	输入捕捉定时器/计数器 输入捕捉定时器/计数器是一个 24 位的向上计数器。其时钟源来自时钟分频器的输出。

**ECAP\_HLD0~2 捕获计数器保持寄存器**

寄存器	偏移量	R/W	描述	复位值
ECAP_HLD0	ECAPn_BA+0x04	R/W	输入捕捉保持寄存器 0	0x0000_0000
ECAP_HLD1	ECAPn_BA+0x08	R/W	输入捕捉保持寄存器 1	0x0000_0000
ECAP_HLD2	ECAPn_BA+0x0C	R/W	输入捕捉保持寄存器 2	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
HOLD							
15	14	13	12	11	10	9	8
HOLD							
7	6	5	4	3	2	1	0
HOLD							

位	描述	
[31:24]	Reserved	保留
[23:0]	HOLD	<b>输入捕捉计数器 Hold 寄存器</b> 当一个使能的输入捕捉通道检测到了以有效的边沿信号改变, ECAP_CNT 的值将会被锁定在对应的保持寄存器。每个输入通道都有对应的的保持寄存器, 序号 0~2 命名为 ECAP_HOLDx 来表示输入来源于 IC0~IC2 的哪一个

**ECAP\_CNTCMP 捕获计数器 比较寄存器**

寄存器	偏移量	R/W	描述	复位值
ECAP_CNTCMP	ECAPn_BA+0x10	R/W	输入捕捉比较寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
CNTCMP							
15	14	13	12	11	10	9	8
CNTCMP							
7	6	5	4	3	2	1	0
CNTCMP							

位	描述	
[31:24]	Reserved	保留
[23:0]	CNTCMP	<p><b>输入捕捉计数器 比较 寄存器</b></p> <p>如果使能了比较功能 (CMPEN= 1), ECAP_CNTCMP 被用作与捕捉计数器 (ECAP_CNT) 相比较</p> <p>如果使能了重载功能 (RLDEN[n] = 1, n=0~3), 溢出事件和捕捉事件会触发硬件自动重载 ECAP_CNTCMP 到 ECAP_CNT</p>

**ECAP\_CTL0 捕获控制寄存器寄存器 0**

寄存器	偏移量	R/W	描述	复位值
<b>ECAP_CTL0</b>	ECAPn_BA+0x14	R/W	输入捕捉控制寄存器 0	0x0000_0000

31	30	29	28	27	26	25	24
Reserved		CAPEN	CMPEN	Reserved		CMPCLREN	CNTEN
23	22	21	20	19	18	17	16
Reserved		CMPIEN	OVIEN	Reserved	CAPIEN2	CAPIEN1	CAPIEN0
15	14	13	12	11	10	9	8
Reserved		CAPSEL2		CAPSEL1		CAPSEL0	
7	6	5	4	3	2	1	0
Reserved	IC2EN	IC1EN	IC0EN	CAPNFDIS		NFCLKSEL	

位	描述
[31:30]	<b>Reserved</b> 保留
[29]	<b>CAPEN</b> 输入捕捉定时器/计数器使能位 0 = 关闭输入捕捉功能. 1 = 使能输入捕捉功能.
[28]	<b>CMPEN</b> 比较功能使能位 输入捕捉定时器/计数器的比较功能会动态比较 ECAP_CNT 的值和比较寄存器 ECAP_CNTCMP 的值, 如果 ECAP_CNT 的值达到了 ECAP_CNTCMP, CAPCMF 标志位会被置位. 0 = 关闭比较功能. 1 = 使能比较功能.
[25]	<b>CMPCLREN</b> 比较-匹配控制清除输入捕捉计数器 如果该位为 1, 比较-匹配事件 (CAPCMF = 1) 发生, 捕捉计数器 (ECAP_CNT) 会被清 0 0 = 比较-匹配事件 (CAPCMF) 清除捕捉计数器 (ECAP_CNT) 禁止. 1 = 比较-匹配事件 (CAPCMF) 清除捕捉计数器 (ECAP_CNT) 使能.
[24]	<b>CNTEN</b> 输入捕捉计数器开始 配置该位为 1, 捕捉计数器 (ECAP_CNT) 与输入捕捉时钟 (CAP_CLK) 相同步地开始向上计数. 0 = ECAP_CNT 停止计数. 1 = ECAP_CNT 开始计数.
[23:22]	<b>Reserved</b> 保留
[21]	<b>CMPIEN</b> CAPCMF 触发输入捕捉中断使能位 0 = 禁止 CAPCMF 触发输入捕捉中断. 1 = 使能 CAPCMF 触发输入捕捉中断.

位	描述	
[20]	<b>OVIEN</b>	<b>CAPOVF 触发输入捕捉中断使能位</b> 0 = 禁止 CAPOVF 触发输入捕捉中断. 1 = 使能 CAPOVF 触发输入捕捉中断.
[19]	<b>Reserved</b>	保留
[18]	<b>CAPIEN2</b>	<b>输入捕捉通道 2 中断使能位</b> 0 = 禁止 CAPTF2 标志触发输入捕捉中断. 1 = 使能 CAPTF2 标志触发输入捕捉中断.
[17]	<b>CAPIEN1</b>	<b>输入捕捉通道 1 中断使能位</b> 0 = 禁止 CAPTF1 标志触发输入捕捉中断. 1 = 使能 CAPTF1 标志触发输入捕捉中断.
[16]	<b>CAPIEN0</b>	<b>输入捕捉通道 0 中断使能位</b> 0 = 禁止 CAPTF0 标志触发输入捕捉中断. 1 = 使能 CAPTF0 标志触发输入捕捉中断.
[15:14]	<b>Reserved</b>	保留
[13:12]	<b>CAPSEL2</b>	<b>CAP2 输入源选择</b> 00 = CAP2 输入来自于端口管脚 ICAP2. 01 = 保留. 10 = CAP2 输入来自于 QEI 控制单元 n 的信号 CHx. 11 = 保留. <b>注:</b> 输入比较单元 n 与 QEIn 相对应, 这里 n=0~1.
[11:10]	<b>CAPSEL1</b>	<b>CAP1 输入源选择</b> 00 = CAP1 输入来自于端口管脚 ICAP1. 01 = 保留. 10 = CAP1 输入来自于 QEI 控制单元 n 的信号 CHB. 11 = 保留. <b>注:</b> 输入比较单元 n 与 QEIn 相对应, 这里 n=0~1.
[9:8]	<b>CAPSEL0</b>	<b>CAP0 输入源选择</b> 00 = CAP0 输入来自于端口管脚 ICAP0. 01 = 保留. 10 = CAP0 输入来自于 QEI 控制单元 n 的信号 CHA. 11 = 保留. <b>注:</b> 输入比较单元 n 与 QEIn 相对应, 这里 n=0~1.
[7]	<b>Reserved</b>	保留
[6]	<b>IC2EN</b>	<b>端口管脚 IC2 输入到输入捕捉单元使能位</b> 0 = 禁止 IC2 输入到输入捕捉单元. 1 = 使能 IC2 输入到输入捕捉单元.
[5]	<b>IC1EN</b>	<b>端口管脚 IC1 输入到输入捕捉单元使能位</b> 0 = 禁止 IC1 输入到输入捕捉单元. 1 = 使能 IC1 输入到输入捕捉单元.

位	描述	
[4]	<b>IC0EN</b>	端口管脚 IC0 输入到输入捕捉单元使能位 0 = 禁止 IC0 输入到输入捕捉单元. 1 = 使能 IC0 输入到输入捕捉单元.
[3]	<b>CAPNFDIS</b>	输入捕捉噪声过滤禁止位 0 = 使能输入捕捉噪声过滤. 1 = 禁止输入捕捉噪声过滤.
[2:0]	<b>NFCLKSEL</b>	噪声过滤时钟预分频选择 选择噪声过滤时钟采样频率 000 = CAP_CLK. 001 = CAP_CLK/2. 010 = CAP_CLK/4. 011 = CAP_CLK/16. 100 = CAP_CLK/32. 101 = CAP_CLK/64.

**ECAP\_CTL1 捕获定时器/计数器控制寄存器 1**

寄存器	偏移量	R/W	描述	复位值
ECAP_CTL1	ECAPn_BA+0x18	R/W	输入捕捉控制寄存器 1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved	CAP2CLREN	CAP1CLREN	CAP0CLREN	Reserved		CNTSRCSEL	
15	14	13	12	11	10	9	8
Reserved	CLKSEL			OVRLDEN	CAP2RLDEN	CAP1RLDEN	CAP0RLDEN
7	6	5	4	3	2	1	0
Reserved		EDGESEL2		EDGESEL1		EDGESEL0	

位	描述
[31:18]	Reserved 保留
[22]	<b>CAP2CLREN</b> 捕捉事件 2 清除捕捉计数器 0 = 禁止事件 CAPTE2 清除捕捉计数器 (ECAP_CNT) 0 = 使能事件 CAPTE2 清除捕捉计数器 (ECAP_CNT)
[21]	<b>CAP1CLREN</b> 捕捉事件 1 清除捕捉计数器 0 = 禁止事件 CAPTE1 清除捕捉计数器 (ECAP_CNT) 0 = 使能事件 CAPTE1 清除捕捉计数器 (ECAP_CNT)
[20]	<b>CAP0CLREN</b> 捕捉事件 0 清除捕捉计数器 0 = 禁止事件 CAPTE0 清除捕捉计数器 (ECAP_CNT) 0 = 使能事件 CAPTE0 清除捕捉计数器 (ECAP_CNT)
[17:16]	<b>CNTSRCSEL</b> 捕捉定时器/计数器时钟源选择 选择捕捉定时器/计数器时钟源 00 = CAP_CLK (默认). 01 = CAP0. 10 = CAP1. 11 = CAP2.
[15]	Reserved 保留

位	描述
[14:12]	<b>CLKSEL</b>  捕捉定时器时钟分频选择 捕捉定时器时钟的前置分频器可以由 CLKSEL[2:0] 配置 8 种不同的选项： 000 = CAP_CLK/1. 001 = CAP_CLK/4. 010 = CAP_CLK/16. 011 = CAP_CLK/32. 100 = CAP_CLK/64. 101 = CAP_CLK/96. 110 = CAP_CLK/112. 111 = CAP_CLK/128.
[11]	<b>Reserved</b> 保留
[11]	<b>OVRLDEN</b>  使能溢出事件触发捕捉计数器重载功能 0 = 禁止 CAPOV 触发重载. 1 = 使能 CAPOV 触发重载.
[10]	<b>CAP2RLDEN</b>  使能 CAPTE2 事件触发捕捉计数器重载功能 0 = 禁止 CAPTE2 事件触发重载. 1 = 使能 CAPTE2 事件 触发重载.
[9]	<b>CAP1RLDEN</b>  使能 CAPTE1 事件触发捕捉计数器重载功能 0 = 禁止 CAPTE1 事件触发重载. 1 = 使能 CAPTE1 事件 触发重载.
[8]	<b>CAP0RLDEN</b>  使能 CAPTE0 事件触发捕捉计数器重载功能 0 = 禁止 CAPTE0 事件触发重载. 1 = 使能 CAPTE0 事件 触发重载.
[7:6]	<b>Reserved</b> 保留
[5:4]	<b>EDGESEL2</b>  通道 2 捕捉边沿选择 输入捕捉可以检测下降沿变化、上升沿变化及双边沿变化 00 = 检测下降沿. 01 = 检测上升沿. 1x = 检测双边沿
[3:2]	<b>EDGESEL1</b>  通道 1 捕捉边沿选择 输入捕捉可以检测下降沿变化、上升沿变化及双边沿变化 00 = 检测下降沿. 01 = 检测上升沿. 1x = 检测双边沿.

位	描述	
[1:0]	<b>EDGESEL0</b>	<p><b>通道 0 捕捉边沿选择</b></p> <p>输入捕捉可以检测下降沿变化、上升沿变化及双边沿变化</p> <p>00 = 检测下降沿.</p> <p>01 = 检测上升沿.</p> <p>1x = 检测双边沿.</p>

**ECAP STATUS 捕获定时器/计数器状态寄存器**

寄存器	偏移量	R/W	描述	复位值
<b>ECAP_STATUS</b>	ECAPn_BA+0x1C	R/W	输入捕捉状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved					CAP2	CAP1	CAP0
7	6	5	4	3	2	1	0
Reserved		CAPOVF	CAPCMPPF	Reserved	CAPTF2	CAPTF1	CAPTF0

位	描述	
[31:6]	<b>Reserved</b>	保留
[10]	<b>CAP2</b>	输入通道 2 的值 CAP2 (只读) 输入通道 2 CAP2 的值 (该位只读, 写无效)
[9]	<b>CAP1</b>	输入通道 1 的值 CAP1 (只读) 输入通道 1 CAP1 的值 (该位只读, 写无效)
[8]	<b>CAP0</b>	输入通道 0 的值 CAP0 (只读) 输入通道 0 CAP0 的值 (该位只读, 写无效)
[5]	<b>CAPOVF</b>	输入捕捉计数器溢出标志 计数器 (ECAP_CNT) 从 0x00FF_FFFF 溢出到 0, 该标志由硬件置位 0 = 上次清 0 后没有溢出事件发生 1 = 上次清 0 后有溢出事件发生 注: 该位写 1 清 0.
[4]	<b>CAPCMPPF</b>	输入捕捉比较-匹配标志 如果使能了输入捕捉比较功能, 当捕捉计数器 (ECAP_CNT) 向上计数到 ECAP_CNTCMP 的值, 硬件置位该位. 0 = ECAP_CNT 不等于 ECAP_CNTCMP 的值. 1 = ECAP_CNT 等于 ECAP_CNTCMP 的值. 注意: 该位写 1 清 0.
[3]	<b>Reserved</b>	保留

位	描述	
[2]	<b>CAPTF2</b>	<b>输入捕捉通道 2 捕捉标志</b> 当输入捕捉通道 2 检测到 CAP2 输入上的有效边沿时, CAPTF2 将会被置. 0 = 上次清 0 后 CAP2 输入上未检测到有效边沿. 1 = 上次清 0 后 CAP2 输入上检测到了有效边沿. <b>注意:</b> 该位写 1 清 0.
[1]	<b>CAPTF1</b>	<b>输入捕捉通道 1 捕捉标志</b> 当输入捕捉通道 1 检测到 CAP1 输入上的有效边沿时, CAPTF1 将会被置. 0 = 上次清 0 后 CAP1 输入上未检测到有效边沿. 1 = 上次清 0 后 CAP1 输入上检测到了有效边沿. <b>注意:</b> 该位写 1 清 0.
[0]	<b>CAPTF0</b>	<b>输入捕捉通道 0 捕捉标志</b> 当输入捕捉通道 0 检测到 CAP0 输入上的有效边沿时, CAPTF0 将会被置. 0 = 上次清 0 后 CAP0 输入上未检测到有效边沿. 1 = 上次清 0 后 CAP0 输入上检测到了有效边沿. <b>注意:</b> 该位写 1 清 0.

ECAP\_VERSION ECAP 版本控制寄存器

寄存器	偏移量	R/W	描述	复位值
ECAP_VERSION	ECAPn_BA+0xFFC	R	ECAP 版本控制寄存器	0x0202_0000

31	30	29	28	27	26	25	24
MAJOR							
23	22	21	20	19	18	17	16
SUB							
15	14	13	12	11	10	9	8
MINOR							
7	6	5	4	3	2	1	0
MINOR							

位	描述	
[31:24]	<b>MAJOR</b>	<b>ECAP RTL 设计主版本号</b> 和产品线相关的主版本号 0x02: 当前主版本号
[23:16]	<b>SUB</b>	<b>ECAP RTL 设计副版本号</b> 和关键特性相关的副版本号 0x02: 当前副版本号 – 和 CLR 一样 RLD 资源可以分开控制
[15:0]	<b>MINOR</b>	<b>ECAP RTL 设计小版本号</b> 依据 ECO 版本控制而得到的小版本号 0x0000: 当前设计小版本号

## 6.15 UART 接口控制器

### 6.15.1 概述

M480 系列提供 6 路通用异步收发器 (UART)。UART 控制器作为标准速度的 UART 还支持流控功能。UART 控制器的接收过程是把外设的串行数据转为并行数据，发送过程是把 CPU 的并行数据转成串行数据发送出去。每个 UART 通道支持 10 种类型的中断。UART 控制器还支持 IrDA SIR,RS-485 和波特率自动测量功能。

### 6.15.2 特性

- 全双工，异步通讯口
- 独立的接收/发送 16/16 字节 FIFO
- 支持硬件自动流控制
- 接收缓存触发等级的数据长度可设
- 每个通道波特率可单独设置
- 支持 nCTS，输入数据，接受 FIFO 数据达到阈值及 RS-485 地址匹配 (AAD 模式) 唤醒功能
- 支持 8 位接收缓存定时溢出检测功能
- 通过设置寄存器 DLY (UA\_TOR [15:8])，可配置两个数据之间 (从上一个 stop 位到下一个 start 位) 的传送时间间隔
- 支持自动波特率检测和波特率补偿功能
  - UART\_CLK 选择 LXT 时支持 9600
- 支持间隔错误，帧错误，校验错误和收/发缓冲区溢出检测等功能
- 可编程串行接口特性
  - 数据位长度可设为 5~8 位
  - 可编程校验，包括奇、偶、无校验，或固定校验位生成和检测
  - 可设置停止位长度为 1 位，1.5 位或 2 位
- 支持 IrDA SIR 功能模式
  - 标准模式下支持 3/16 位宽功能
- 支持 LIN 功能模式 (仅 UART0/UART1 支持)
  - 支持 LIN 主/从模式
  - 传输中支持 分隔域生成功能可设
  - 支持接收器 分隔域检测功能
- 支持 RS-485 模式
  - 支持 RS-485 9-位模式
  - 支持软/硬件控制 nRTS 管脚，用于控制 RS-485 传送方向
- 支持 PDMA 传输功能

UART 特征	UART0/ UART1	UART2/ UART4/ UART5	UART3/ UART5	SC_UART	USCI-UART
FIFO 字节数	16	16	4		TX: 1 RX: 2
自动流控 (CTS/RTS)	√	√	-		√
IrDA	√	√	-		-
LIN	√	-	-		-
RS-485 功能模式	√	√	-		√
nCTS 唤醒	√	√	-		√
输入数据唤醒	√	√	-		√
接收数据达到 FIFO 阈值唤醒	√	√	-		-
RS-485 地址匹配 (AAD) 唤醒	√	√	-		-
自动波特率检测	√	√	-		√
停止位长度	1, 1.5, 2 bit	1, 1.5, 2 bit	1, 2 bit		1, 2 bit
字长	5, 6, 7, 8 bits	5, 6, 7, 8 bits	5, 6, 7, 8 bits		6~13 bits
奇偶校验位	√	√	√		√
插入位	√	√	-		-
<b>Note:</b> √= Supported					

表 6.15-1 NuMicro™ M480 系列 UART 特性

### 6.15.3 框图

UART 时钟控制和模块框图分别如图 6.15-1 和图 6.15-2 所示。

注: UARTx\_CLK 的时钟频率不能超过HCLK的30倍

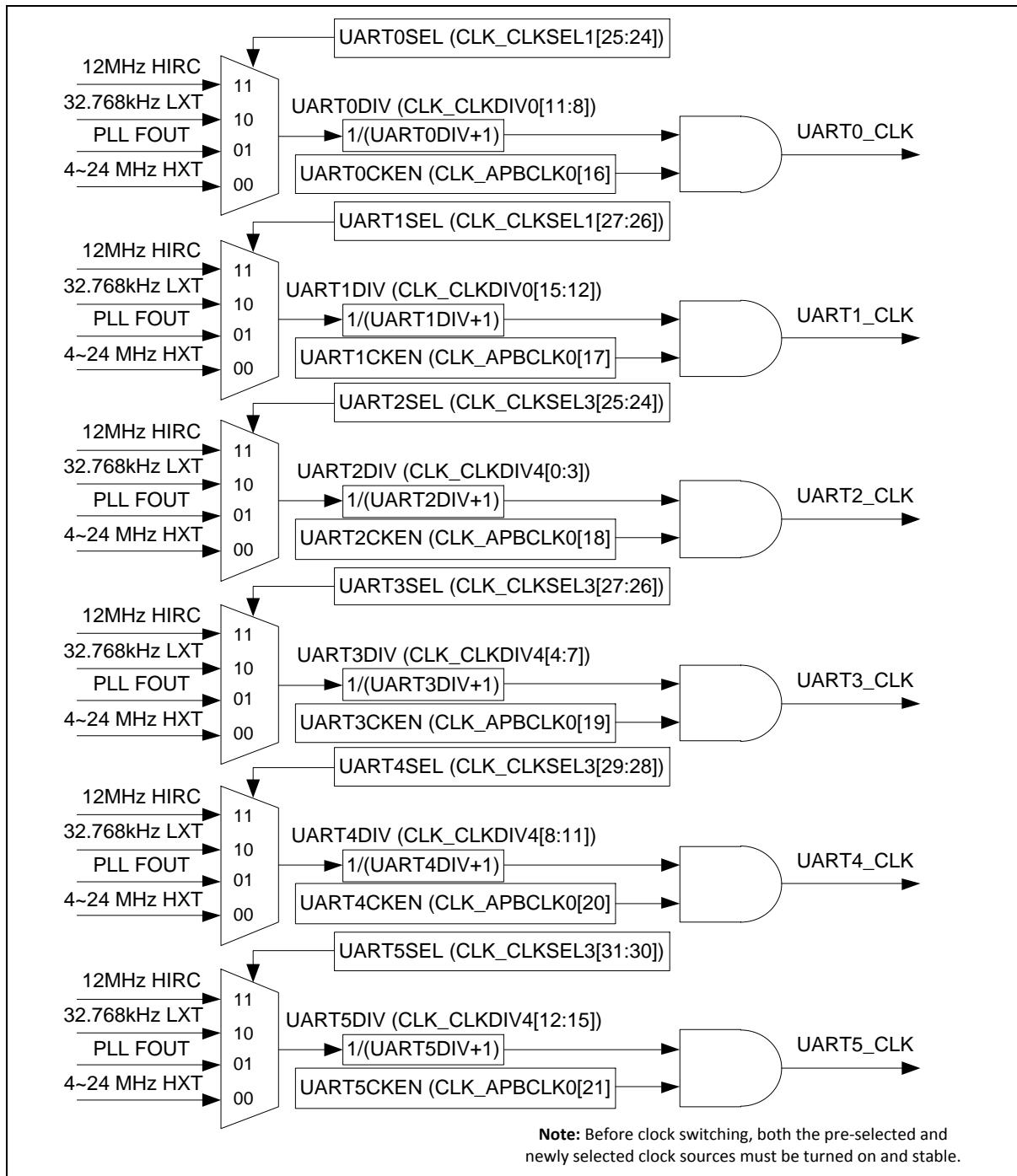


图 6.15-1 UART 时钟控制框图

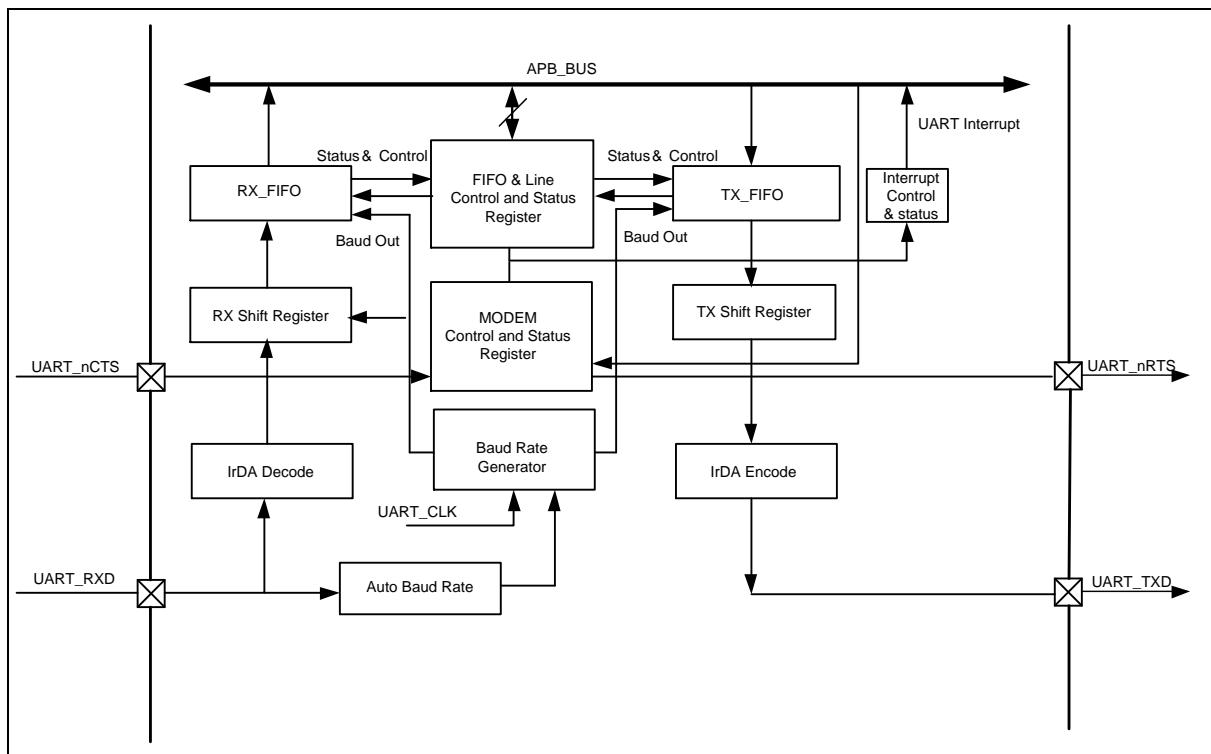


图 6.15-2 UART 结构图

每个模块功能详述如下：

#### **TX\_FIFO**

发送口带有一个 16 字节的 FIFO 缓冲区以减少 CPU 中断的频率

#### **RX\_FIFO**

接收口带有一个 16 字节的 FIFO 缓冲区（另加 3 个出错位， BIF (UART\_FIFOSTS[6]), FEF (UART\_FIFOSTS[5]), PEF (UART\_FIFOSTS[4])) 以减少 CPU 中断的频率

#### **TX 移位寄存器**

该模块用于控制把并行数据串行输出

#### **RX 移位寄存器**

该模块用于控制把串行数据并行输入

#### **Modem 控制和状态寄存器**

该寄存器用于控制到 Modem 或数传机 (或类似于 Modem 的外设) 的接口

#### **波特率发生器**

通过把输入的外部时钟除频得到期望的波特率。详情请参考波特率公式

**IrDA 编码**

该模块为 IrDA 编码控制模块

**IrDA 解码**

该模块为 IrDA 解码控制模块

**FIFO & 线控控制和状态寄存器**

该寄存器组包括 FIFO 控制寄存器 (UART\_FIFO)， FIFO 状态寄存器 (UART\_FIFOSTS) 和 线控寄存器 (UART\_LINE)。定时溢出寄存器 (UART\_TOUT) 配置时间溢出中断条件。

**波特率自动检测**

该模块用于自动检测波特率

**中断控制和状态寄存器**

一共有 10 种类型的中断，包括接收数据可用中断 (RDAINT)，发送保持寄存器空中断 (THERINT)，发送完成中断 (TXENDINT)，接收线状态中断 (校验错误或帧错误或 break 错误) (RLSINT)，MODEM 状态中断 (MODEMINT)，接收超时中断 (RXTOINT)，缓存错误中断 (BUFERRINT)，和 LIN 总线中断 (LININT)，唤醒中断 (WKINT) 和自动波特率中断 (ABRINT)。中断使能寄存器 (UART\_INTEN) 用于使能或禁止相应中断，中断状态寄存器 (UART\_INTSTS) 显示哪些中断正在发生。

中断	描述
RDAINT	接收数据可用中断
THERINT	发送保持寄存器空中断
TXENDINT	发送完成中断
RLSINT	接收线状态中断 (校验错误或帧错误或 break 错误)
MODEMINT	MODEM 状态中断
RXTOINT	接收缓存超时中断
BUFERRINT	缓存错误中断.
LININT	LIN 总线中断
WKINT	唤醒中断
ABRINT	自动波特率中断

表 6.15-2 UART 中断

**6.15.4 基本配置**

UART0 基本配置如下：

- 时钟源配置
  - 在寄存器 UART0SEL (CLK\_CLKSEL1[25:24]) 选择 UART0 外设时钟源
  - 在寄存器 UART0DIV (CLK\_CLKDIV0[11:8]) 选择 UART0 外设时钟分频数
  - 在寄存器 UART0CKEN (CLK\_APBCLK1[16]) 使能 UART0 外设时钟

- 用寄存器 UART0RST (SYS\_IPRST1[16]) 复位 UART0 控制器
- 管脚配置

组	管脚名	GPIO	MFP
UART0	UART0_RXD	PA.15, PC.11, PF.2	MFP3
		PB.8	MFP5
		PB.12	MFP6
		PA.0, PA.6	MFP7
		PH.11	MFP8
	UART0_TXD	PD.2	MFP9
		PA.14, PC.12, PF.3	MFP3
		PB.9	MFP5
		PB.13	MFP6
		PA.1, PA.7	MFP7
	UART0_nCTS	PH.10	MFP8
		PD.3	MFP9
		PB.11	MFP5
	UART0_nRTS	PB.15	MFP6
		PA.5, PC.7	MFP7
		PB.10	MFP5
	UART0_nRTS	PB.14	MFP6
		PA.4, PC.6	MFP7

UART1 基本配置如下：

- 时钟源选择
  - 在寄存器 UART1SEL (CLK\_CLKSEL1[27:26]) 配置 UART1 外设时钟源
  - 在寄存器 UART1DIV (CLK\_CLKDIV0[15:12]) 配置 UART1 外设时钟分频数
  - 在寄存器 UART1CKEN (CLK\_APBCLK1[17]) 使能 UART1 外设时钟
- 用寄存器 UART1RST (SYS\_IPRST1[17]) 复位 UART1 控制器
- 管脚配置

组	管脚名	GPIO	MFP
UART1	UART1_RXD	PF.1	MFP2
		PD.6, PD.10	MFP3
		PB.2, PB.6	MFP6
		PA.8	MFP7
		PA.2, PC.8, PG.1	MFP8
		PH.9	MFP10

UART1_TXD	PF.0	MFP2
	PD.7, PD.11	MFP3
	PB.3, PB.7	MFP6
	PA.9	MFP7
	PA.3, PE.13, PG.0	MFP8
	PH.8	MFP10
	PB.9	MFP6
UART1_nCTS	PA.1, PE.11	MFP8
	PB.8	MFP6
UART1_nRTS	PA.0, PE.12	MFP8

UART2 管脚配置如下：

- 时钟源选择
  - 在寄存器 UART2SEL (CLK\_CLKSEL3[25:24]) 配置 UART2 外设时钟源
  - 在寄存器 UART2DIV (CLK\_CLKDIV4[3:0]) 选择 UART2 外设时钟分频数
  - 在寄存器 UART2CKEN (CLK\_APBCLK1[18]) 使能外设时钟
- 在寄存器 UART2RST (SYS\_IPRST1[18]) 复位 UART2 控制器
- 管脚配置n

组	管脚名	GPIO	MFP
UART2	UART2_RXD	PF.5	MFP2
		PE.15	MFP3
		PG.0	MFP6
		PB.0, PD.12, PE.9	MFP7
		PC.0, PC.4	MFP8
	UART2_TXD	PF.4	MFP2
		PE.14	MFP3
		PG.1	MFP6
		PB.1, PC.13, PE.8	MFP7
		PC.1, PC.5	MFP8
	UART2_nCTS	PD.9, PF.5	MFP4
		PC.2	MFP8
	UART2_nRTS	PD.8, PF.4	MFP4
		PC.3	MFP8

TUART3 管脚配置如下：

- 时钟源选择
  - 在寄存器 UART3SEL (CLK\_CLKSEL3[27:26]) 配置 UART3 外设时钟源

- 在寄存器 UART3DIV (CLK\_CLKDIV4[7:4]) 配置 UART3 外设时钟分频数
- 在寄存器 UART3CKEN (CLK\_APBCLK1[19]) 使能外设时钟
- 在寄存器 UART3RST (SYS\_IPRST1[19]) 复位 UART3 控制器
- 管脚配置

组	管脚名	GPIO	MFP
UART3	UART3_RXD	PD.0	MFP5
		PB.14, PC.9, PE.0, PE.11	MFP7
		PC.2	MFP11
	UART3_TXD	PD.1	MFP5
		PB.15, PC.10, PE.1, PE.10	MFP7
		PC.3	MFP11
	UART3_nCTS	PD.2	MFP5
		PB.12, PH.9	MFP7
	UART3_nRTS	PD.3	MFP5
		PB.13, PH.8	MFP7

UART4 管脚配置如下：

- 时钟源选择
  - 在寄存器 UART4SEL (CLK\_CLKSEL3[29:28]) 选择 UART4 外设时钟源
  - 在寄存器 UART4DIV (CLK\_CLKDIV4[11:8]) 配置 UART4 外设时钟源分频数
  - 在寄存器 UART4CKEN (CLK\_APBCLK1[20]) 使能 UART4 外设时钟
- 在寄存器 UART4RST (SYS\_IPRST1[20]) 复位 UART4 控制器
- 管脚配置

组	管脚名	GPIO	MFP
UART4	UART4_RXD	PA.13	MFP3
		PC.6, PH.3	MFP5
		PB.10, PF.6	MFP6
		PA.2, PH.11	MFP7
		PC.4	MFP11
	UART4_TXD	PA.12	MFP3
		PC.7, PH.2	MFP5
		PB.11, PF.7	MFP6
		PA.3, PH.10	MFP7
	UART4_nCTS	PC.5	MFP11
		PC.8	MFP5

		PE.1	MFP9
	UART4_nRTS	PE.13	MFP5
		PE.0	MFP9

UART5 管脚配置如下：

- 时钟源选择
  - 在寄存器 UART5SEL (CLK\_CLKSEL3[31:30]) 选择 UART5 外设时钟源
  - 寄存器 UART5DIV (CLK\_CLKDIV4[15:12]) 选择 UART5 外设时钟分频数
  - 寄存器 UART5CKEN (CLK\_APBCLK1[21]) 使能 UART5 外设时钟
- 在寄存器 UART5RST (SYS\_IPRST1[21]) 复位 UART5 控制器
- 管脚配置

组	管脚名	GPIO	MFP
UART5	UART5_RXD	PH.1	MFP4
		PB.4	MFP7
		PA.4, PE.6	MFP8
	UART5_TXD	PH.0	MFP4
		PB.5	MFP7
		PA.5, PE.7	MFP8
	UART5_nCTS	PH.3	MFP4
		PB.2	MFP7
	UART5_nRTS	PH.2	MFP4
		PB.3	MFP7

表 6.15-3 是 UART 接口控制器管脚描述：

管脚	类型	描述
UARTx_TXD	Output	UARTx 发送
UARTx_RXD	Input	UARTx 接收
UARTx_nCTS	Input	UARTx modem 清零发送
UARTx_nRTS	Output	UARTx modem 请求发送

表 6.15-3 UART 接口控制器

### 6.15.5 功能描述

UART 控制器支持四种功能模式，包括 UART, IrDA, LIN 和 RS-485 模式。用户可以通过对 UART\_FUNCSEL 设置选择功能。四种功能模式将在下面的章节中描述。.

#### 6.15.5.1 UART 控制器波特率发生器

UART 控制器包含一个可编程波特率发生器，其通过分频器对输入时钟源分频而得到收发数据所需的串行时钟。下表列出了各种条件下的 UART 波特率计算公式和参数的设置。通过设置相应的参

数和寄存器可以得到零误差的波特率。IrDA 功能模式，波特率发生器必须是模式 0。寄存器表 UART\_BAUD 一栏中将更详细地介绍寄存器内容。共有三种设定模式：设定 UART\_BAUD[29:28]=00 为模式 0，设定 UART\_BAUD[29:28]=10 为模式 1，设定 UART\_BAUD[29:28]=11 为模式 2。

模式	BAUDM1	BAUDM0	波特率公式
模式 0	0	0	UART_CLK / [16 * (BRD+2)].
模式 1	1	0	UART_CLK / [(EDIVM1+1) * (BRD+2)], EDIVM1 要 $\geq$ 8.
模式 2	1	1	UART_CLK / (BRD+2) 如果 UART_CLK $\leq$ 3*HCLK, BRD 要 $\geq$ 9. 如果 UART_CLK > 3*HCLK, BRD 要 $\geq$ 3*N - 1. N 大于等于 UART_CLK / HCLK 的最小整数 例如, 如果 3*HCLK < UART_CLK $\leq$ 4*HCLK, BRD 要 $\geq$ 11. 如果 4*HCLK < UART_CLK $\leq$ 5*HCLK, BRD 要 $\geq$ 14. (UART_CLK 选择为 LXT, BRD 大于等于 1)

表 6.15-4 UART 控制器波特率公式表

UART 外设时钟 = 12 MHz			
波特率	模式 0	模式 1	模式 2
921600	不支持	不推荐	BRD=11
460800	不推荐	BRD=0, EDIVM1 =13	BRD=24
230400	不推荐	BRD =2, EDIVM1 =13	BRD =50
115200	不推荐	BRD =6, EDIVM1 =13	BRD =102
57600	BRD =11	BRD =14, EDIVM1 =13	BRD =206
38400	BRD =18	BRD =22, EDIVM1 =13	BRD =311
19200	BRD =37	BRD =123, EDIVM1 =5	BRD =623
9600	BRD =76	BRD =123, EDIVM1 =10	BRD =1248
4800	BRD =154	BRD =248, EDIVM1 =10	BRD =2498

表 6.15-5 UART 控制器波特率参数设置示例

UART 外设时钟 = 12 MHz			
波特率	UART_BAUD 值		
	模式 0	模式 0	Mode 2
921600	不支持	不推荐	0x3000_000B
460800	不推荐	0x2D00_0000	0x3000_0018
230400	不推荐	0x2D00_0002	0x3000_0032
115200	不推荐	0x2D00_0006	0x3000_0066

57600	0x0000_000B	0x2D00_000E	0x3000_00CE
38400	0x0000_0012	0x2D00_0016	0x3000_0137
19200	0x0000_0025	0x2500_007B	0x3000_026F
9600	0x0000_004C	0x2A00_007B	0x3000_04E0
4800	0x0000_009A	0x2A00_00F8	0x3000_09C2

表 6.15-6 UART 控制器波特率寄存器设置示例

### 6.15.5.2 UART 控制器波特率补偿

UART 控制器支持波特率补偿功能。这提高了每一位的精度。由于每一位都由 BRCOMDEC 位 (UART\_BRCOMP[31]) 来定义正负补偿，所以补偿的精度为 UART 模块时钟的一半。如果 BRCOMDEC bit (UART\_BRCOMP[31]) 为 0，每一位都正向补偿，被补偿的位会超过一个模块时钟长度。如果 BRCOMDEC bit (UART\_BRCOMP[31]) 为 1，每一位都负向补偿，被补偿的位会少于一个模块时钟长度。

BRCOMP[8:0] (UART\_BRCOMP[8:0]) 总共有 9 位，可以由用户自由定义对应的位是否被补偿。BRCOMP[7:0] 用来配置 UART\_DAT[7:0] 的补偿，而 BRCOMP[8] 用来定义校验位。

示例：

(1). UART 外设时钟为 32.768K，波特率为 9600

UART 外设时钟为 32.768K，波特率为 9600 → 即每位 3.413 个外设时钟

如果分频值设为 1 (每位 3 个外设时钟)，则每位的误差为 -0.413 个外设时钟，BRCOMPDEC = 0

Bit	名称	总误差	BRCOMP 补偿	最终误差
0	Start	-0.413	x	-0.413
1	UART_DAT[0]	-0.826(-0.413-0.413)	1	0.174
2	UART_DAT[1]	-0.239(0.174-0.413)	0	-0.239
3	UART_DAT[2]	-0.652(-0.239-0.413)	1	0.348
4	UART_DAT[3]	-0.065(0.348-0.413)	0	-0.065
5	UART_DAT[4]	-0.478(-0.065-0.413)	0	-0.478
6	UART_DAT[5]	-0.891(-0.478-0.413)	1	0.109
7	UART_DAT[6]	-0.304(0.109-0.413)	0	-0.304
8	UART_DAT[7]	-0.717(-0.304-0.413)	1	0.283
9	Parity	-0.130(0.283-0.413)	0	-0.13

表 6.15-7 波特率补偿示例 1

所以 BRCOMP (UART\_BRCOMP[8:0]) 可以设为 9'b010100101 = 0xa5.

(2). UART 外设时钟为 32.768K，波特率为 4800

UART 外设时钟为 32.768K，波特率为 4800 → 每位 6.827 个外设时钟

如果分频值设为 5 (每位 7 个外设时钟), 则每位的误差为 0.173 个外设时钟, **BRCOMPDEC = 1**

Bit	名称	总误差	BRCOMP 补偿	最终误差
0	Start	0.173	x	0.173
1	UART_DAT[0]	0.346(0.173+0.173)	0	0.346
2	UART_DAT[1]	0.519(0.346+0.173)	1	-0.481
3	UART_DAT[2]	-0.308(-0.481+0.173)	0	-0.308
4	UART_DAT[3]	-0.135(-0.308+0.173)	0	-0.135
5	UART_DAT[4]	-0.038(-0.135+0.173)	0	0.038
6	UART_DAT[5]	0.211(0.038+0.173)	0	0.211
7	UART_DAT[6]	0.384(0.211+0.173)	0	0.384
8	UART_DAT[7]	0.557(0.384+0.173)	1	-0.443
9	Parity	-0.270(-0.443+0.173)	0	-0.270

表 6.15-8 波特率补偿示例 2

所以 BRCOMP (UART\_BRCOMP[8:0]) 可以设为 9'b010000010 = 0x82.

#### 6.15.5.3 UART 控制器自动波特率功能

自动波特率功能可以自动测量从 UART RX 管脚输入数据的波特率。当自动波特率测量结束, 测量结果将会放置在 BRD (UA\_BAUD[15:0]) 中。AUDM1 (UART\_BAUD[29]) 和 BAUDM0 (UART\_BAUD[28]) 都被自动设置为 1。在自动波特率检测帧中, UART RX 数据从起始位到第一个上升沿的时间由 2 ABRDBITS 位时间设定。

从起始位到第一个上升沿之间的 2 ABRDBITS 位时间是通过设定 ABRDBITS (UART\_ALTCTL[20:19]) 计算得出。通过设置 ABRDEN (UART\_ALTCTL[18]) 可以使能自动波特率检测功能。在初始阶段, UART RX 保持为 1。一旦检测到下降沿, 即为接收到起始位。自动波特率计数器被重启并开始计数。当检测到第一个上升沿时, 自动波特率计数器将停止计数。然后, 自动波特率计数器数值除以 ABRDBITS (UART\_ALTCTL[20:19]) 的结果将会自动载入到 BRD (UART\_BAUD[15:0]) 中, 而且 ABRDEN (UART\_ALTCTL[18]) 会被清零。如图 7.15 3 所示, 一旦自动波特率测量结束, ABRDIF (UART\_FIFOSTS[1]) 会被置位。当自动波特率计数器溢出, ABRTOIF (UART\_FIFOSTS[2]) 将会被置位。ABRDIF (UART\_FIFOSTS[1]) 及 ABRDTOIF (UART\_FIFOSTS[2]) 会产生自动波特率标志位 ABRIF (UART\_ALTCTL[17])。如果使能了 ABRIEN (UART\_INTEN[18]), ABRIF (UART\_ALTCTL[17]) 将会引发自动波特率中断 ABRINT (UART\_INTSTS[31]) 产生。

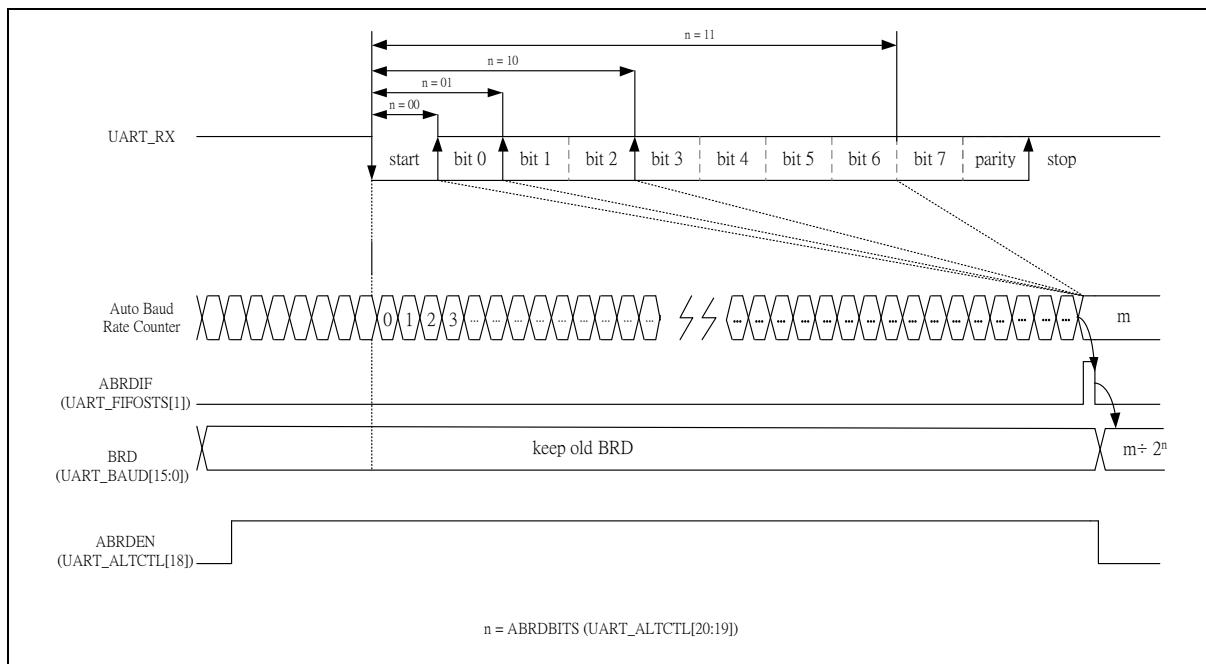


图 6.15-3 自动波特率检测

#### 编程顺序示例:

1. 设置 ABRDBITS (UART\_ALTCTL[20:19]) 用于决定 UART RX 数据从起始到第一个上升沿时间用的 2 ABRDBITS 位时间
2. 置位 ABRIEN (UART\_INTEN[18]) 用于使能自动波特率检测功能中断
3. 置位 ABRDEN (UART\_ALTCTL[18]) 用于使能自动波特率检测功能
4. 当 ABRDIF (UART\_FIFOSTS[1]) 为 1, 表示自动波特率测量结束
5. 执行 UART 发送和接收的动作
6. 当 ABRDTOIF (UART\_FIFOSTS[2]) 为 1, 表示自动波特率计数器溢出
7. 返回步骤 3

#### 6.15.5.4 UART 控制器发送延时时间

UART 控制器可以通过设置 DLY (UART\_TOUT [15:8]) 来控制传输过程两个数据帧之间即上一个数据帧的停止位和下一个数据帧的起始位之间的间隔。单位是波特。延时操作如图 7.15 4. 所示

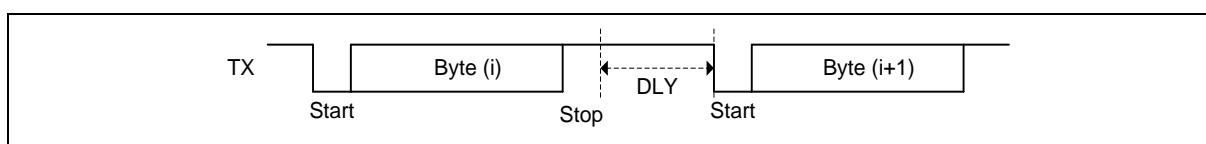


图 7.15-4 发送延时操作

#### 6.15.5.5 UART 控制器 FIFO 控制和状态

UART 控制器内置一个 16 字节的发送 FIFO (TX\_FIFO) 和一个 16 字节的接收 FIFO (RX\_FIFO), 通讯中, 使用这些收发 FIFO 可以减少对 CPU 的中断次数。CPU 在操作过程中任何时间都可以读取 UART 的状态, 包括 3 个错误状态 (校验错误, 帧错误, 间隔错误) 如果接收的数据出现校验错误、帧或间隔错误。FIFO 控制和状态也支持所有的功能模式, 包括 UART, IrDA, LIN 和 RS-485 模式。

### 6.15.5.6 UART 控制器唤醒功能

UART 控制器支持系统唤醒功能。唤醒功能包括 nCTS 唤醒和接收数据唤醒，接收 FIFO 数据达到阈值唤醒，RS-485 地址匹配（AAD 模式）唤醒和接收数据 FIFO 阀值超时唤醒。CTSWKF(UART\_WKSTS[0]), DATWKF (UART\_WKSTS[1]), RFRTWKF (UART\_WKSTS[2]), RS485WKF(UART\_WKSTS[3]) 或 TOUTWKF (UART\_WKSTS[4]) 可以引发产生唤醒中断标志 WKIF(UART\_INTSTS[6])。如果 WKIEN (UART\_INTEN[6]) 使能，唤醒中断标志 WKIF(UART\_INTSTS[6]) 引发产生唤醒中断 WKINT (UART\_INTSTS[14])。

#### nCTS 管脚唤醒：

当系统处于掉电模式且 WKCTSEN (UART\_WKCTL[0]) 置位，nCTS 管脚可以唤醒系统。如果 WKCTSEN (UART\_WKCTL[0]) 使能，nCTS 管脚上的跳变会产生 nCTS 唤醒标志 CTSWF (UART\_WKSTS[0])。如图 7.15 5 和图 7.15 6 是唤醒示意图：.

#### nCTS 唤醒示例 1 (nCTS 发送由低到高)

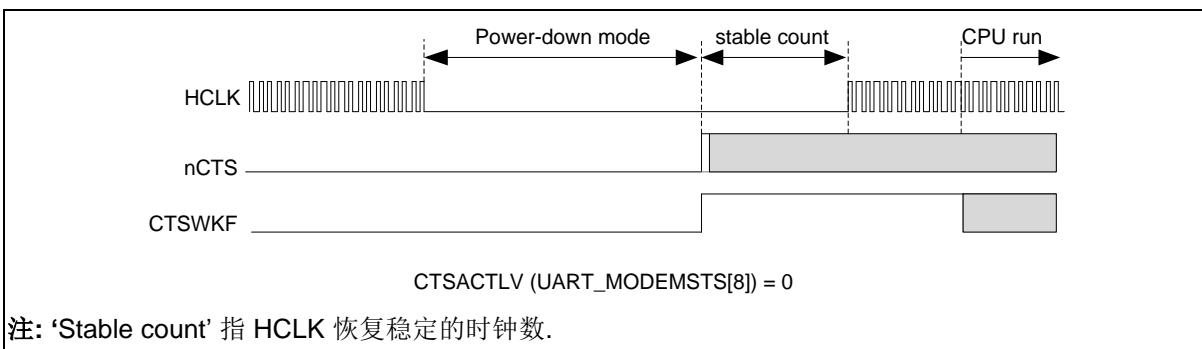


图 6.15-5 UART nCTS 唤醒示例 1

#### nCTS 唤醒示例 2 (nCTS 发送由高到低)

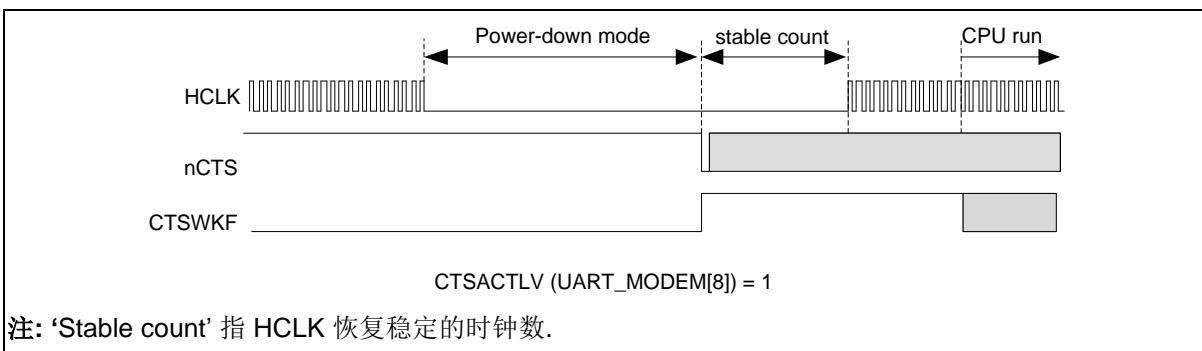


图 6.15-6 UART nCTS 唤醒示例 2

#### 输入数据唤醒：

当系统处于掉电模式且 WKDATEN (UART\_WKCTL [1]) 置位，输入数据 (UART\_RXD) 管脚上的跳变可以唤醒系统。为了能在系统唤醒后正确接收数据，需要设置 STCOMP (UART\_DWKCOMP[15:0]) 位域。STCOMP 里的各位代表系统从掉电模式唤醒后，可以接收第 1 个位 (开始位) 需要的时钟数。

输入数据唤醒系统后，输入的数据会被存储到 FIFO 内。如果使能了 WKDATEN (UART\_WKCTL[1])，输入数据 (UART\_RXD) 管脚上的跳变会触发输入数据唤醒标志 DATWKF (UART\_WKSTS[1])。图 7.15 7 是输入数据唤醒的示例。

**注 1:** UART 控制器的时钟源需要选为 HIRC 且开始位的补偿时间需为 15.68us。也就是说，STCOMP

(UART\_DWKCOMP[15:0]) 可以设为 347.

**注 2:** BRD (UART\_BAUD[15:0]) 的值需要大于 STCOMP (UART\_DWKCOMP[15:0]).

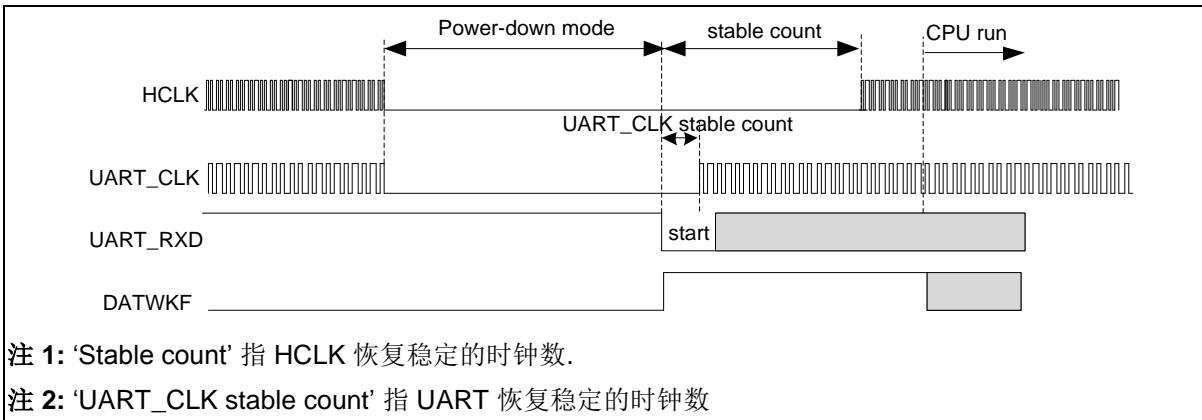


图 6.15-7 UART 数据唤醒

#### 接收数据 FIFO 达到阈值唤醒:

设定 WKRFRTEN (UART\_WKCTL[2]) 可以使能接收数据 FIFO 达到阈值唤醒功能。在掉电模式，当 RX FIFO 里接收到的数据达到了阈值设定 RFITL (UART\_FIFO[7:4])，就会唤醒系统。如果 WKRFRTEN (UART\_WKCTL[2]) 使能，RXFIFO 接收到了 RFITL (UART\_FIFO[7:4]) 设定个数的值后会触发接收数据 FIFO 达到阈值标志 RFRTWKF (UART\_WKSTS[2])。如图 7.15 8 是唤醒示例：

**注:** 为了接收数据，掉电模式下 UART 时钟源需要设为 LXT

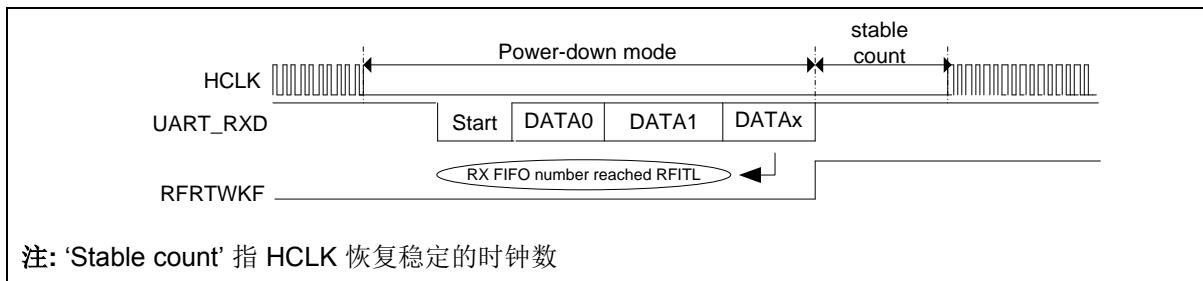
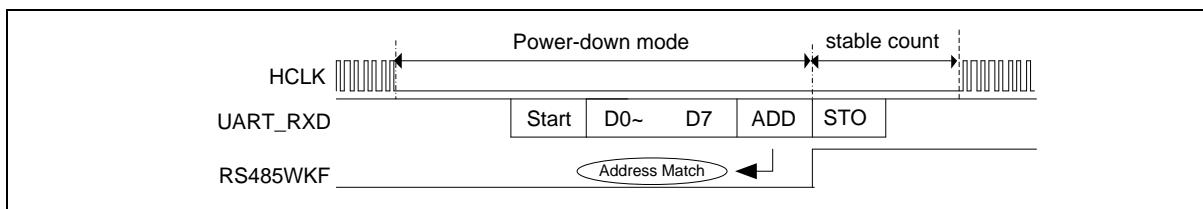


图 6.15-8 UART 接收数据 FIFO 达到阈值唤醒

#### RS-485 地址匹配 (AAD 模式) 唤醒:

设定 WKRFRTEN (UART\_WKCTL[2]) 和 WKRS485EN (UART\_WKCTL[3]) 使能 RS-485 地址匹配唤醒功能。该功能需要在 ADDRDEN (UART\_ALTCTL[15]) 置 1，处于 RS-485 自动地址检测 (AAD) 模式下使用。在掉电模式下，检测到的地址位与 ADDRMV (UART\_ALTCTL[31:24]) 相匹配或者 RXFIFO 里接收到的数据达到了阈值设定 RFITL (UART\_FIFO[7:4]) 都会唤醒系统。如果使能 WKRS485EN (UART\_WKCTL[3])，检测到的地址位与 ADDRMV (UART\_ALTCTL[31:24]) 相匹配时会触发 RS485 地址匹配 (AAD 模式) 标志位 RS485WKF (UART\_WKSTS[3])。如图 7.15 9 是唤醒示例

**注:** 为了接收数据，掉电模式下 UART 时钟源需要设为 LXT



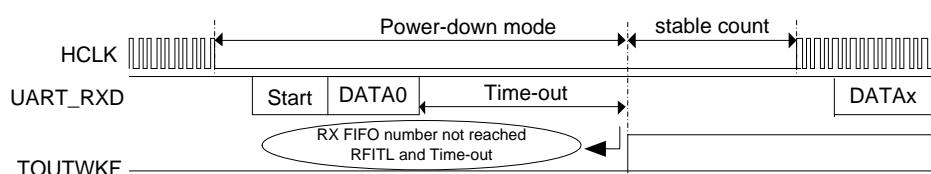
注: 'Stable count' 指 HCLK 恢复稳定的时钟数

图 6.15-9 UART RS-485 AAD 模式地址匹配唤醒

#### 数据接收 FIFO 超时唤醒:

设置 WKRFRTEN (UART\_WKCTL[2]) 和 WKTOUTEN (UART\_WKCTL[4]) 使能数据接收 FIFO 超时唤醒功能。置位 TOCNTEN (UART\_INTEN[11]) 使能接收缓存超时计数器。掉电模式下, RXFIFO 里接收到的数据阈值设定 RFITL (UART\_FIFO[7:4]) 时, 超时计数器计数到 TOIC (UART\_TOUT[7:0]) 会唤醒系统。如果 WKTOUTEN (UART\_WKCTL[4]) 使能, 超时计数器计数到 TOIC (UART\_TOUT[7:0]) 的值会触发数据接收 FIFO 超时唤醒标志位 TOUTWKF (UART\_WKSTS[4])。如图 7.15 1 是唤醒例

注: 为了接收数据, 掉电模式下 UART 时钟源需要设为 LXT



注: 'Stable count' 指 HCLK 恢复稳定的时钟数

图 6.15-10 UART 数据接收 FIFO 超时唤醒

#### 6.15.5.7 UART 控制器中断和状态

每个 UART 控制器都有 10 种类型中断:

- 接收数据可用中断 (RDAINT)
- 发送保持寄存器空中断 (THERINT)
- 发送完成中断 (TXENDIF)
- Line 接收状态中断 (RLSINT)
  - 间隔中断标志 (BIF)
  - 帧错误标志 (FEF)
  - 校验错误标志 (PEF)
  - RS-485 地址位检测标志 (ADDRDETF)
- MODEM 状态中断 (MODEMINT)
  - 检测 nCTS 状态改变标志 (CTSDETF)
- 接收缓存超时中断 (RXTOINT)
- 缓存错误中断 (BUFERRINT)
  - TX 溢出错误中断标志 (TXOVIF)
  - RX 溢出错误中断标志 (RXOVIF)
- LIN 总线中断 (LININT)
  - LIN 间隔检测 (BRKDETF)
  - 位错误检测状态标志 (BITEF)

- LIN 从机 ID 校验错误标志 (SLVIDPEF)
- LIN 从机报头错误标志 (SLVHEF)
- LIN 从机报头检测标志 (SLVHDETF)
- 唤醒中断 (WKINT)
  - nCTS 唤醒标志 (CTSWKF)
  - 输入数据唤醒标志 (DATWKF)
  - 接收数据 FIFO 达到阈值唤醒标志 (RFRTWKF)
  - RS-485 地址匹配 (AAD 模式) 唤醒标志 (RS485WKF)
  - 数据接收 FIFO 超时唤醒标志 (TOUTWKF)
- 自动波特率中断 (ABRINT)
  - 自动波特率检测中断标志 (ABRDIF)
  - 自动波特率检测超时中断标志 (ABRDTOIF)

表 7.15 9 所示是各中断源和标志位。当中断使能位使能，中断标志位触发，就会发生中断。用户需要在中断发生后清除中断标志。

中断源	中断指示	中断使能位	中断标志位	标志触发条件	清除标志方法
接收数据可用中断	RDAINT	RDAIEN	<b>RDAIF</b>	N/A	读 UART_DAT
发送保持寄存器空中断	THERINT	THREIEN	<b>THREIF</b>	N/A	写 UART_DAT
发送完成中断	TXENDINT	TXENDIEN	<b>TXENDIF</b>	N/A	写 UART_DAT
Line 接收状态中断	RLSINT	RLSIEN	<b>RLSIF</b>	<b>RLSIF = BIF</b>	写 1 到 BIF
				<b>RLSIF = FEF</b>	写 1 到 FEF
				<b>RLSIF = PEF</b>	写 1 到 PEF
				<b>RLSIF = ADDRDETF</b>	写 1 到 ADDRDETF
MODEM 状态中断	MODEMINT	MODEMIEN	<b>MODEMIF</b>	<b>MODEMIF</b> CTSDETF	写 1 到 CTSDETF
接收缓存超时中断	RXTOINT	RXTOIEN	<b>RXTOIF</b>	N/A	读 UART_DAT
缓存错误中断	BUFERRINT	BUFERRIEN	<b>BUFERRIF</b>	<b>BUFERRIF</b> TXOVIF	写 1 到 TXOVIF
				<b>BUFERRIF</b> RXOVIF	写 1 到 RXOVIF
LIN 总线中断	LININT	LINIEN	<b>LINIF</b>	<b>LINIF</b> = BRKDETF	写 1 到 LINIF 和 写 1 到 BRKDETF
				<b>LINIF</b> = BITEF	写 1 到 LINIF 和 写 1 到 BITEF

				<b>LINIF = SLVIDPEF</b>	写 1 到 LINIF 和 写 1 到 SLVIDPEF
				<b>LINIF = SLVHEF</b>	写 1 到 LINIF 和 写 1 到 SLVHEF
				<b>LINIF = SLVHDETF</b>	写 1 到 LINIF 和 写 1 到 SLVHDETF
唤醒中断	WKINT	WKIEN	WKIF	<b>WKIF = CTSWKF</b>	写 1 到 CTSWKF
				<b>WKIF = DATWKF</b>	写 1 到 DATWKF
				<b>WKIF = RFRTWKF</b>	写 1 到 RFRTWKF
				<b>WKIF = RS485WKF</b>	写 1 到 RS485WKF
				<b>WKIF = TOUTWKF</b>	写 1 到 TOUTWKF
自动波特率中断	ABRINT	ABRIEN	ABRIF	<b>ABRIF = ABRDIF</b>	写 1 到 ABRDIF
				<b>ABRIF = ABRDTOIF</b>	写 1 到 ABRDTOIF

表 6.15-9 UART 控制器中断源和标志位列表

#### 6.15.5.8 UART 功能模式

UART 控制器提供了 UART 功能 (用户须设置 FUNCSEL (UART\_FUNCSEL [1:0]) 设置为 '00' 来使能 UART 功能模式)。UART 波特率最高速度是 1 Mbps.

UART 为全双工异步通讯接口。收发各包含一个 16 字节的 FIFO 缓冲区。用户可以设置接收时的 FIFO 触发阈值以及定时溢出检测时间。发送数据帧间 (即从上一帧停止位到下一帧起始位) 时间间隔通过 DLY (UA\_TOR [15:8]) 位可设。UART 支持硬件自动流控功能，且 nRTS 流控触发电平可设。如果 RX FIFO 中数据字节数大于或等于 RTSTRGLV (UART\_FIFO[19:16])，nRTS 将被释放。

#### UART 线控功能

UART 控制器通过设置 UART\_LINE 寄存器支持串行接口全部特性。通过对 UART\_LINE 寄存器设置数据位和停止位长度以及校验位。下表列出了 UART 数据位和停止位长度的设置以及 UART 校验位的设置。

NSB (UART_LINE[2])	WLS (UART_LINE[1:0])	数据长度 (Bit)	停止位长度 (Bit)
0	00	5	1
0	01	6	1
0	10	7	1
0	11	8	1
1	00	5	1.5

1	01	6	2
1	10	7	2
1	11	8	2

表 6.15-10 UART 线控的数据位和停止位长度设置

校验类型	SPE (UART_LINE[5])	EPE (UART_LINE[4])	PSS (UART_LINE[7])	PBE (UART_LINE[3])	描述
No Parity	x	x	x	0	无奇偶校验位输出
Parity source from UART_DAT	x	x	1	1	软件产生并验证校验位.
Odd Parity	0	0	0	1	奇校验位的计算方法是把数据流中的所有的 1 相加, 使得包括校验位在内,1 的总数为奇数个
Even Parity	0	1	0	1	偶校验位的计算方法是把数据流中的所有的 1 相加, 使得包括校验位在内,1 的总数为偶数个
Forced Mask Parity	1	0	0	1	奇偶校验位总是逻辑 1 不管数据位中 1 的个数是奇数还是偶数, 奇偶校验位永远都是逻辑 1
Forced Space Parity	1	1	0	1	奇偶校验位总是逻辑 0 不管数据位中 1 的个数是奇数还是偶数, 奇偶校验位永远都是逻辑 0

表 6.15-11 UART 线控校验位设置

### UART 自动流控功能

UART 支持自动流控功能, 该功能用到两根信号线 nCTS (清零发送) 和 nRTS (请求发送) 来控制 UART 与外部设备 (如 Modem) 间的数据传输。当自动流控使能后, 只有等到 UART 对外部设备发出有效的 nRTS 信号后才允许接收数据, 否则不接收。当 RX FIFO 接收到数据的数量等于 RTSTRGLV (UART\_FIFO [19:16]) 位的值后, nRTS 信号会被取消。当 UART 检测到外部设备给 nCTS 信号脚有效信号后, UART 开始发送数据, 否则 UART 不会发送数据。流控控制的框图参见图 7.15.11。.

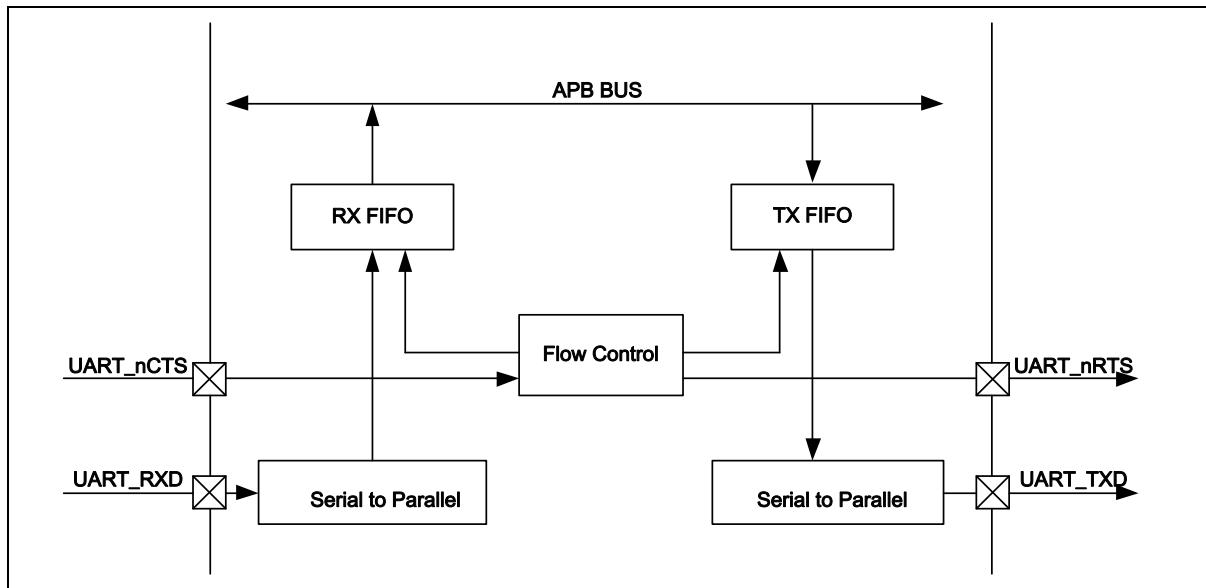


图 6.15-11 自动流控框图

图 7.15-12 展示了 UART nCTS 自动流控功能。用户必须要先设置 ATOCTSEN (UART\_INTEN[13]) 以使能 nCTS 自动流控功能。CTSACTLV (UART\_MODEMSTS [8]) 位可以设置 CTS 脚输入的有效状态。当 nCTS 脚上任何电平变化将导致 CTSDETF (UART\_MODEMSTS[0]) 位被置 1，然后 TX FIFO 将自动将数据发送到 TX 脚，并传送出去。

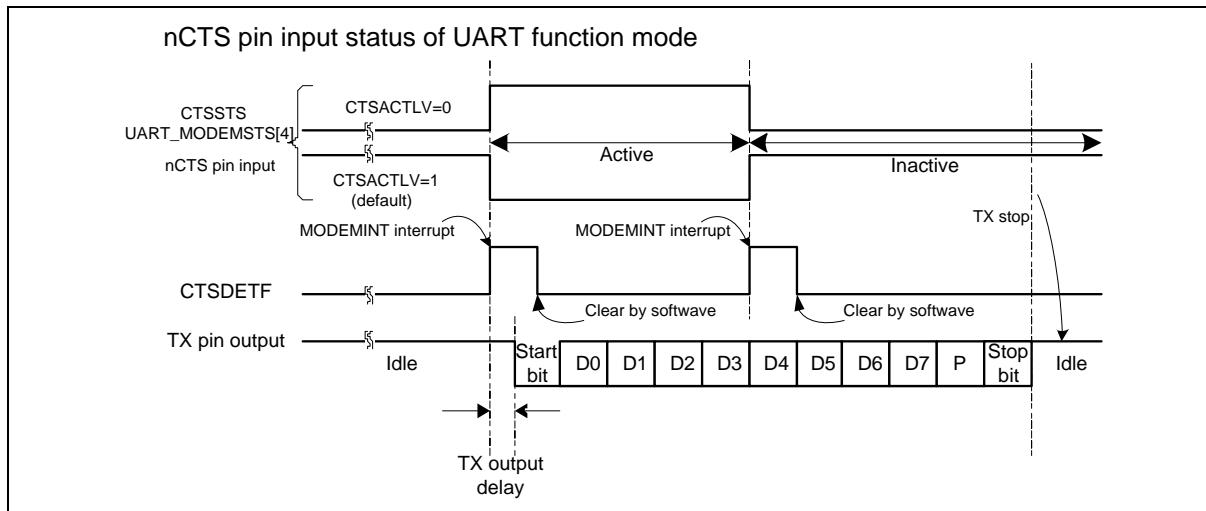


图 7.15-12 UART nCTS 自动流控使能

如下图 7.15-13 所示，UART nRTS 自动流控模式 (ATORTSEN (UART\_INTEN[12])=1) 中，nRTS 的触发阈值由 UART FIFO 控制寄存器的 RTSTRGLV (UART\_FIFO[19:16]) 控制。

设置 RTSACTLV (UART\_MODEM[9]) 可以控制 nRTS 管脚正常输出或反向，信号来自内部的nRTS。用户可以读 RTSSTS (UART\_MODEM[13]) 来获取 nRTS 脚输出电压的真实逻辑状态。

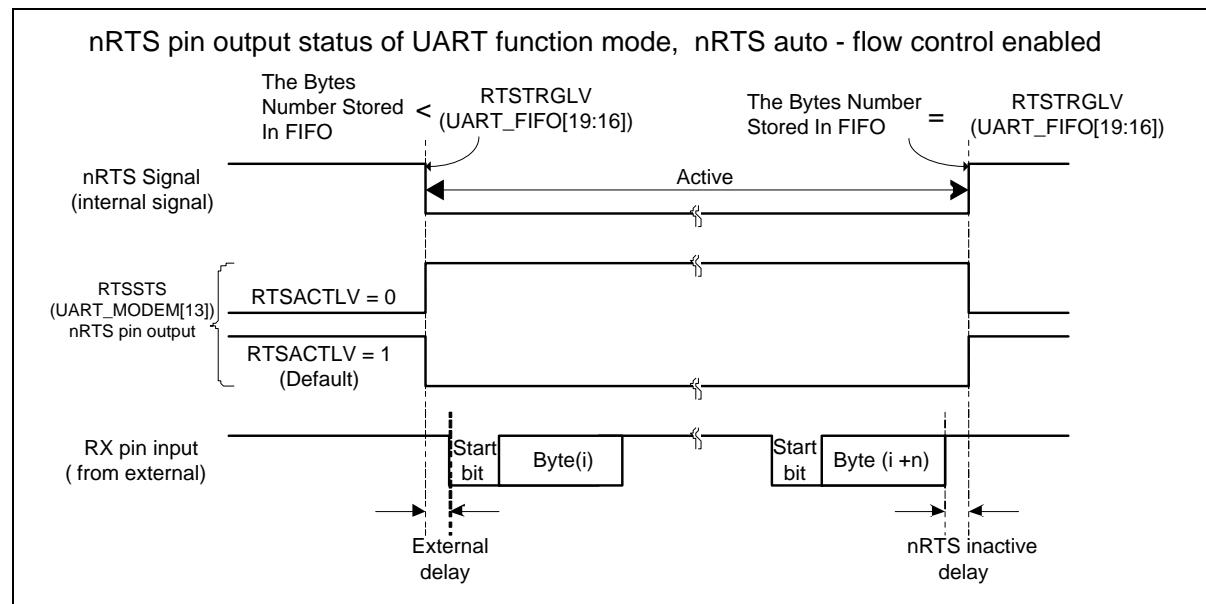


图 6.15-13 UART nRTS 自动流控功能使能

如图 7.15-14，在软件模式下 (ATORTSEN (UART\_INTEN[12])=0)，软件改动 RTS (UART\_MODEM[1]) 控制位来实现 nRTS 流控。

设置 RTSACTLV (UART\_MODEM[9]) 可以控制 nRTS 管脚正常输出或反向，信号来自 RTS(UART\_MODEM[1])。用户可以读 RTSSTS (UART\_MODEM[13]) 位来获取 nRTS 管脚真实输出电平状态。.

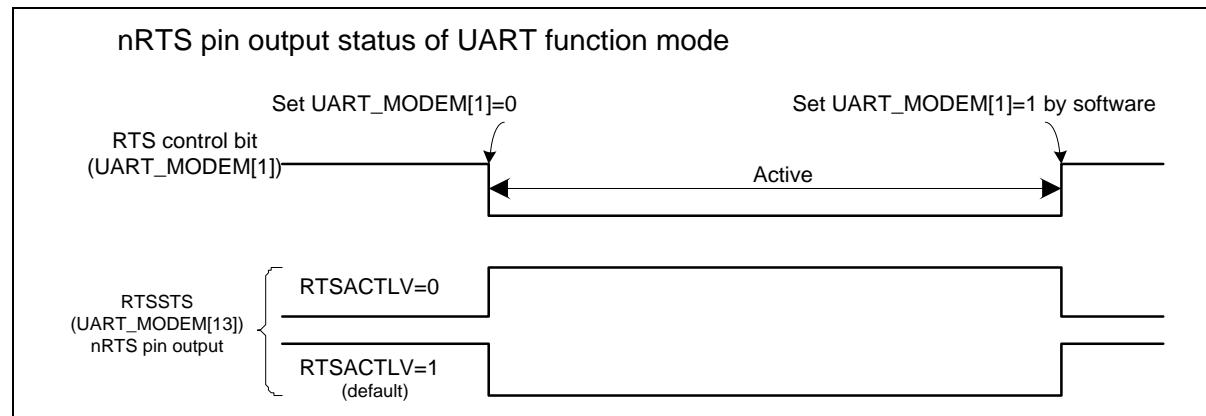


图 6.15-14 UART nRTS 软件控制的自动流控

### 6.15.5.9 IrDA 功能模式

UART 控制器也提供串行 IrDA (SIR, 串行红外) 功能 (用户必须设置 UART\_FUNCSEL [1:0] 为'10' 来使能 IrDA 功能)。SIR 规范定义了一个短距离红外异步串行传输模式，它包括一个起始位，8 个数据位，一个停止位。最大速率 115.2kbps。IrDA SIR 模块包含一个 IrDA SIR 协议编/解码器。IrDA SIR 协议是半双工工作模式，所以它不能同时收发数据。IrDA SIR 物理层规定了发送与接收数据的时间上至少 10ms 的时间间隔，该延迟特性需通过软件来完成。

IrDA 模式下，BAUDM1 (UART\_BAUD [29]) 需为 0

波特率 = Clock / (16 \* (BRD +2))，这里 BRD (UART\_BAUD[15:0]) 是 UA\_BAUD 寄存器中定义的波特率分频器。

注: IrDA 主机和从机间波特率不能相差超过 $\pm 5\%$ .

图 7.15 15 展示了 IrDA 控制模块框图。

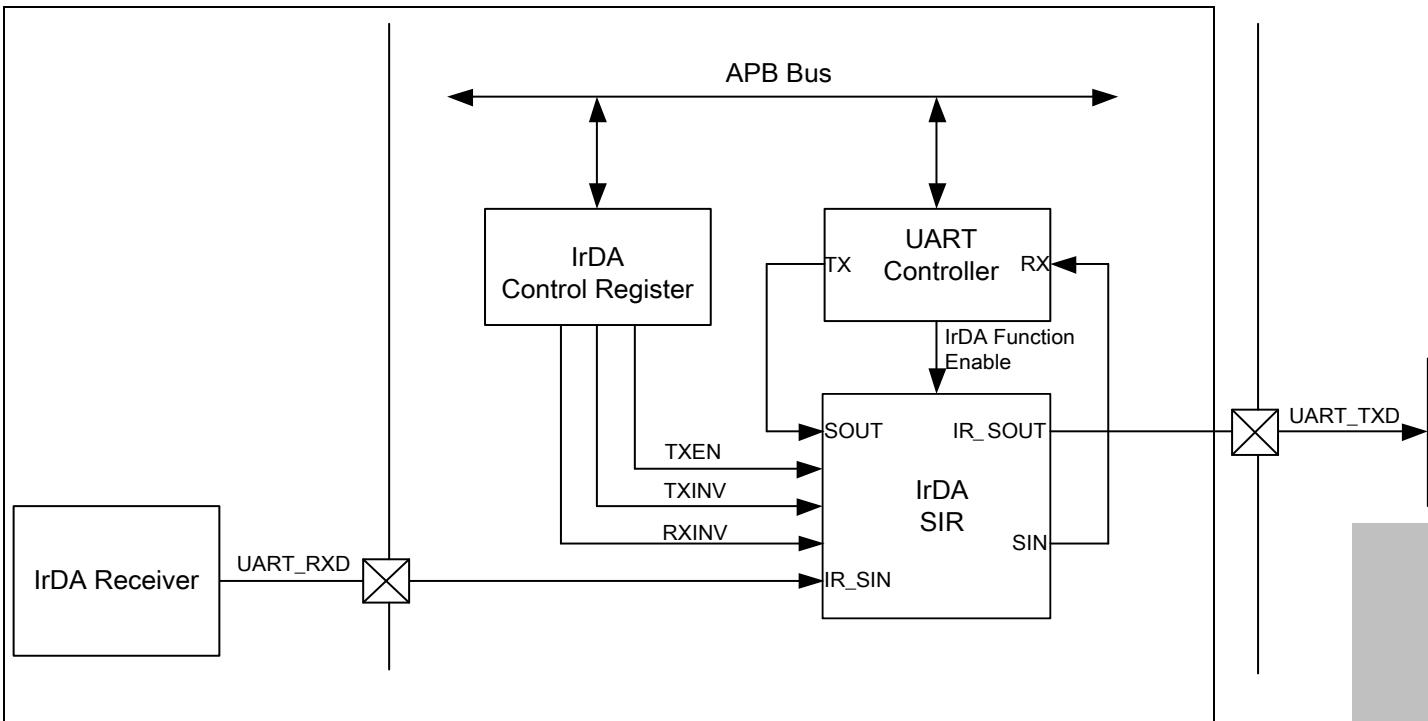


图 6.15-15 IrDA 控制框图

#### IrDA SIR 发送编码

IrDA SIR 传送编码调制采用 Non-Return-to Zero (NRZ) 编码，数据流编码后从 UART 接口输出。IrDA SIR 物理层指定使用归零反向调制编码 (Return-to-Zero, 反转 (RZI))，用一个红外光脉冲代表逻辑 0，被调整的脉冲输出到外部输出驱动器和红外线发光二极管。

在正常模式下，传输脉冲的宽度为 3/16 波特率周期。

#### IrDA SIR 接收解码

IrDA SIR 接收解码 IrDA SIR 接收解码器对输入管脚的 (Return-to-Zero, Inverted (RZI)) 串行位流进行解调，并输出 NRZ 串行位流到 UART 接收数据输入端。

当解码器输入端为低时，表明接收到一个起始位。在空闲状态里，解码器输入端通常为高。正常操作时，RXINV (UART\_IRDA[6]) 设置为 1，TXINV (UART\_IRDA[5]) 设置为 0。

#### IrDA SIR 操作

IrDA SIR 编码/解码提供 UART 数据流和半双工串行 SIR 之间的转换。图 7.15 16 是 IrDA 编码/解码波形图：.

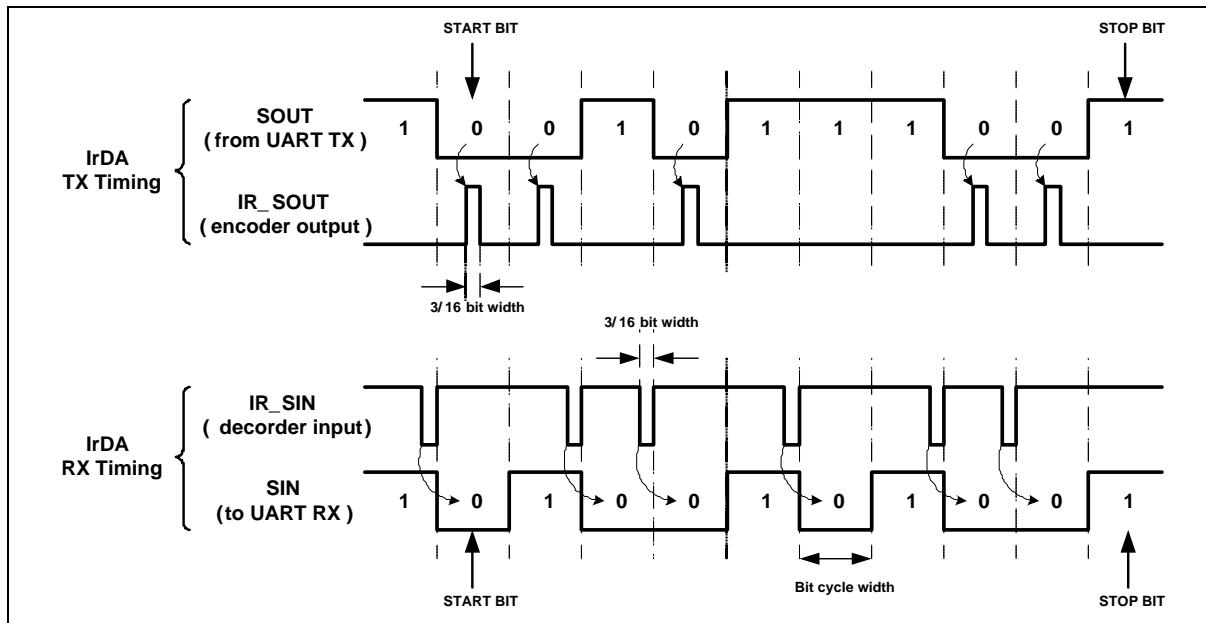


图 6.15-16 IrDA TX/RX 时序图

#### 6.15.5.10 LIN (本地互联网络) 模式

UART 控制器支持 LIN 功能，通过设置 FUNCSEL (UART\_FUNCSEL[1:0]) 为 '01' 可以将 UART 设定为 LIN 模式。在主机模式，UART 控制器支持 LIN 间隔/分隔的产生以及间隔/分隔的侦测，在 LIN 从机模式下，支持报头侦测和自动重新同步。

#### LIN 帧结构

根据 LIN 协议，所有的传输信息被打包为帧。一个帧由一个报头（主机任务提供）和一个紧跟其后的应答（从机任务提供）组成。报头（主机任务提供）由一个间隔域和一个同步域再跟一个帧识别码（帧 ID）组成。帧 ID 仅作为定义帧的用途。从机任务负责回应相关的帧 ID。应答由一个数据域和一个校验域组成。图 7.15-17 是 LIN 帧的结构。

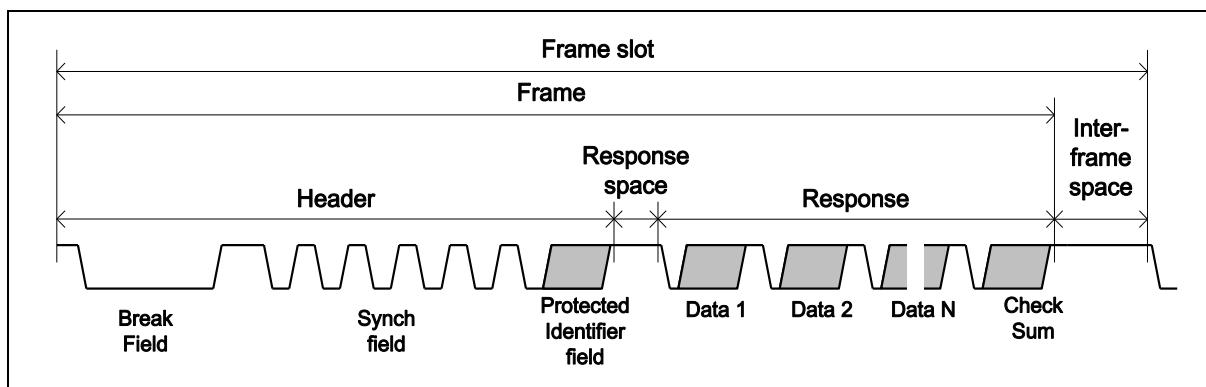


图 7.15-17 LIN 帧结构

#### LIN 字节结构

在 LIN 模式，根据 LIN 的标准，每个字节由值为 0 (显性) 的 START 位开始，接着是 8 位数据位，没有校验位，LSB 优先，由一个值为 1 (隐性) 的 STOP 位结束。字节的结构如下图

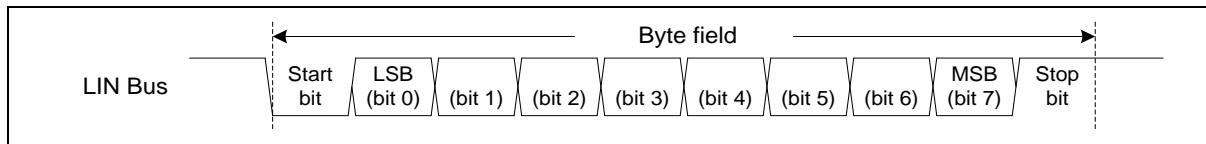


图 6.15-18 LIN 字节结构

### LIN 主机模式

UART 控制器支持 LIN 主机模式，使能并初始化 LIN 主机模式需要如下步骤：

1. 设置 UART\_BAUD 寄存器设定波特率。
2. 设置 WLS (UART\_LINE[1:0])='11'配置数据长度为 8 位， PBE (UART\_LINE[3])=0 禁止奇偶校验， NSB (UART\_LINE[2])=0 配置 1 位 stop 位
3. 设置 FUNCSEL (UART\_FUNCSEL[1:0]) 位为“01”选择 LIN 功能模式。

一个完整的报头由一个间隔域和同步域再跟一个帧标识符 (帧 ID) 组成。UART控制器可以选择三种报头发送模式，通过设置 HSEL (UART\_LINCTL[23:22]) 可以选择“间隔域”或“间隔域和同步域”或“间隔域，同步域和 帧 ID 域”作为发送的报头。如果选择的报头是“间隔域”，发送一个完整的报头到总线，软件必须依次填同步数据 (0x55) 和帧 ID 数据到 UART\_DAT 寄存器。如果选择的报头是“间隔域和同步域”，软件必须填帧 ID 数据到 UART\_DAT 寄存器，按顺序来发送一个完整的报头到总线。如果选择的报头是“间隔域，同步域和 帧 ID 域”，硬件会自动控制报头发送顺序，但是软件必须填帧 ID 数据到 PID (UART\_LINCTL [31:24])。当选择的报头模式是“间隔域，同步域和 帧 ID 域”时，帧 ID 校验位可以由软件或硬件来计算，这取决于 IDPEN (UART\_LINCTL[9]) 位是否置位。

HSEL	间隔域	同步域	ID 域
0	硬件产生	软件产生	软件产生
1	硬件产生	硬件产生	软件产生
2	硬件产生	硬件产生	硬件产生 (先由软件填 ID 到 PID (UART_LINCTL[31:24]))

表 6.15-12 LIN 主机模式报头选择

当 UART 工作于 LIN 数据传输模式时， LIN 总线传输状态可以由硬件或软件监控。通过设置 BITERREN (UART\_LINCTL [12]) 为 “1”使能硬件监控。如果在 LIN 发送状态输入管脚 (UART\_RX) 状态不同于输出管脚 (UART\_TX) 状态，硬件会产生一个中断到 CPU。软件也能通过读回 UART\_DAT 寄存器数据监视 LIN 总线传输状态。以下为编程顺序示例。

在主机模式下，没有软件错误监控的步骤：

1. 填保密标识符 ID 到 PID (UART\_LINCTL[31:24]).
2. 通过设置 HSEL (UART\_LINCTL [23:22])=10，选择硬件传输的报头域包括“间隔域 + 同步域+保密标识符 ID 域”。
3. 通过设置 SENDH (UART\_LINCTL[8]) 为 1 选择硬件传输报头。
4. 等待 SENDH (UART\_LINCTL[8]) 被硬件清零。
5. 等待 TXEMPTYF (UART\_FIFOSTS[28]) 被硬件置位。

**注 1:** 间隔域 的缺省值是 12 个显性位 (间隔域) 和 1 个隐性位 (break/sync 分隔符). 软件可以通过

设定 BRKFL (UART\_LINCTL [19:16]) 和 BSL (UART\_LINCTL[21:20]) 来改变间隔域的长度和 break/sync 分隔符的长度.

**注 2:** 间隔域/同步域分隔符缺省长度是 1 比特时间，字节间隔缺省也是 1 个比特时间。软件可以通过设定 BSL (UART\_LINCTL[21:20]) 和 DLY (UART\_TOUT[15:8]) 来改变它们的间隔.

**注 3:** 如果报头包含“间隔域，同步域和帧 ID 域”，在触发硬件发送报头之前（设定 SENDH (UART\_LINCTL[8])，软件必须填帧 ID 到 PID (UART\_LINCTL[31:24])。根据 IDPEN (UART\_LINCTL[9]) 的设定，帧 ID 校验可以由硬件或者软件产生。如果校验由软件产生 IDPEN (UART\_LINCTL[9]) 为 0, 软件必须填 8 位数据 (包括 2 个校验位) 到帧 ID 域；如果校验位由硬件产生 (IDPEN (UART\_LINCTL[9])= 1)，软件填 ID0~ID5 就好，硬件负责计算 P0 和 P1。

在主机模式下，有软件错误监控的步骤：

1. 设置 UART\_LINCTL [23:22]= 00，选择硬件传输的报头域，其内容只包括“间隔域”
2. 设置 BRKDETEN (UART\_LINCTL[10]) 使能间隔域侦测功能
3. 设置 SENDH (UART\_LINCTL[8]) 请求硬件传输间隔+ 间隔/同步 分隔符.
4. 等待 BRKDETF (UART\_LINSTS[8]) 标志被硬件置为“1”.
5. 写 0x55 到 UART\_DAT 寄存器发送 同步域.
6. 等待 RDAIF (UART\_INTSTS[0]) 标志被硬件置为“1”，然后读回 UART\_DAT 寄存器的值.
7. 写保护 ID 到 UART\_DAT 寄存器，请求帧 ID 域
8. 等待 RDAIF (UART\_INTSTS[0]) 标志被硬件置为“1”，然后读回 UART\_DAT 寄存器的值.

### LIN 暂停和分隔检测

当软件通过设定 BRKDETEN (UART\_LINCTL[10]) 使能 间隔域侦测功能时， 间隔域侦测功能被激活。 间隔域侦测电路和 UART 接收电路是完全独立的。

当间隔域侦测功能使能时，电路侦测 UART\_RX 引脚的起始信号。如果 UART LIN 控制器侦测到显性连续大于 11 个比特并跟随 1 个隐性比特（分隔符），在间隔域结束时 BRKDETF (UART\_LINSTS[8]) 标志将被置位，如果 LINIEN (UART\_INTEN[8])=1, LININT (UART\_INTSTS[15]) 中断将发生，break 侦测和 break 标志如图 7.15 19 所示。

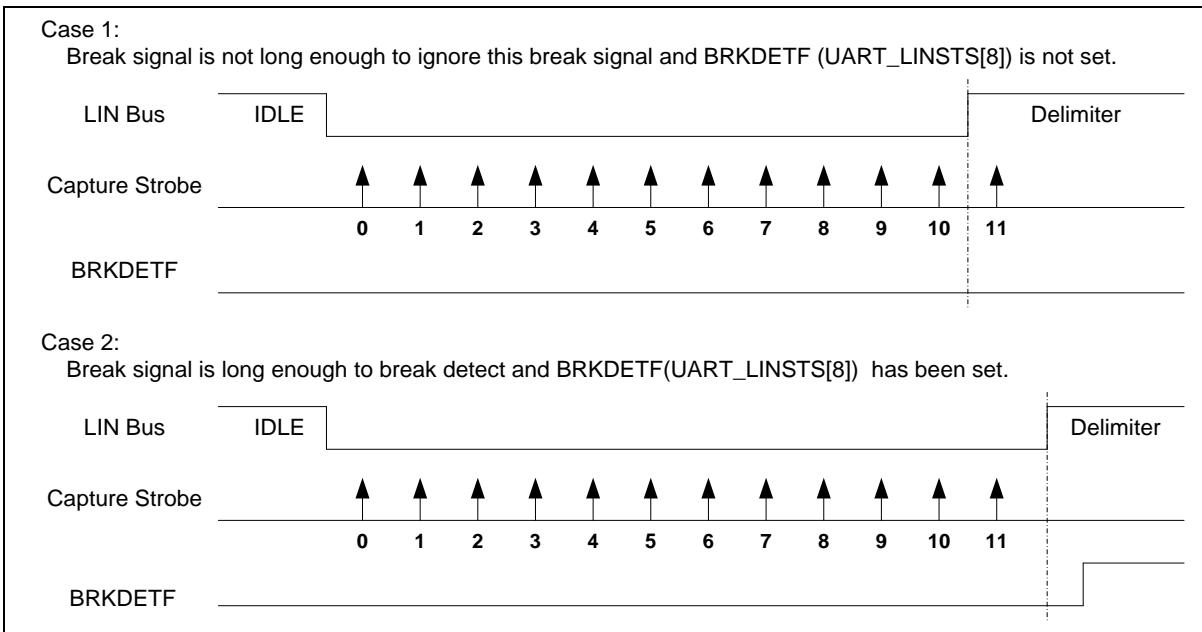


图 6.15-19 LINE 模式暂停检测

### LIN 帧 ID 和校验格式

LIN 模式下 LIN 的帧 ID 值如下，帧 ID 校验位可以通过软件或硬件产生，产生方式取决于 IDPEN (UART\_LINCTL[9]) 位的设置。

如果校验位通过硬件产生，用户需要填写 ID0~ID5 (UART\_LINCTL [29:24] )，硬件将会计算 P0 (UART\_LINCTL[30]) 和 P1 (UART\_LINCTL[31])，否则用户必须填写帧 ID 和校验位

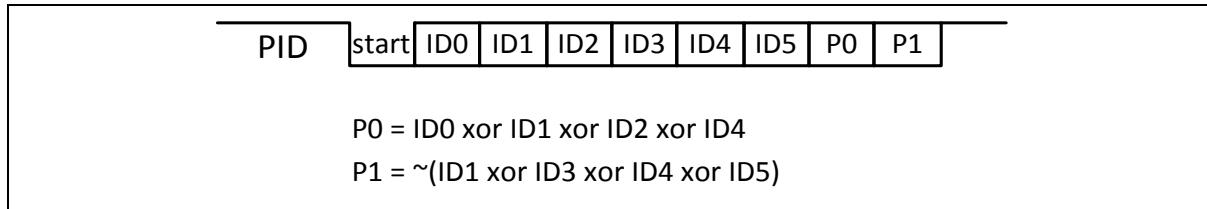


图 6.15-20 LIN 帧 ID 和校验格式

### LIN 从机模式

UART 控制器支持 LIN 从模式。使能和初始化 LIN 从模式的必要步骤如下：

1. 设置 UART\_BAUD 寄存器设定波特率。
2. 设定 WLS (UART\_LINE[1:0])=11 设定数据长度为 8 比特，清 PBE (UART\_LINE[3]) 禁止校验，清 NSB (UART\_LINE[2]) 设定停止位为 1 比特
3. 设定 FUNCSEL (UART\_FUNCSEL[1:0]) 为“01”，选择 LIN 功能。
4. 设定 SLVEN (UART\_LINCTL[0]) 为 1，使能 LIN 从机模式。

### LIN 报头接收

根据 LIN 协议，从节点必须等待从主节点接收一个有效的报头，然后应用程序可以采取以下动作 (根据主报头帧 ID 的值)。

- 接收响应

- 发送响应.
- 忽略响应，等待下一个报头.

LIN 从模式下，通过设定 SLVHDEN (UART\_LINCTL[1])，使能从机报头检测功能来侦测完整的报头（接收“间隔域”，“sync 域”和“frame ID 域”）。接收到一个 LIN 报头后，SLVHDETF (UART\_LINSTS[0]) 标志将被置位。如果 LINIEN (UART\_INTEN[8]) =1，中断将发生。通过设定 IDPEN (UART\_LINCTL[9])，使能帧 ID 校验检查功能。如果收到的帧 ID 校验位不正确时 (break 和 sync 域正确)，SLVIDPEF (UART\_LINSTS[2]) 标志将被置 1。如果 LINIEN (UART\_INTEN[8]) =1，中断将发生，SLVHDETF (UART\_LINSTS[0]) 会被置 1。通过设定 MUTE (UART\_LINCTL[4])=1，用户也可以将 LIN 设为 mute 模式。该模式只允许检测报头 (break + sync + frame ID)，不允许接收任何其它字符。为了避免比特率偏差，控制器支持自动重同步功能，避免时钟误差错误，通过设定 SLVAREN (UART\_LINCTL[2]) 使能该特性。

### LIN 发送响应

LIN 从机模式可以发送响应和接收响应。当从机节点是响应的发送者时，通过将数据填到 UART\_DAT 寄存器发送响应；如果从机节点是响应的接收者时，从接节点从 LIN 总线接收数据。

### LIN 报头超时错误

LIN 从机控制器包含一个报头超时计数器。如果整个报头没有在 57 比特的最大时间限制内收到，报头错误标志 SLVHEF (UART\_LINSTS [1]) 将被置位。超时计数器在每个 break 侦测沿使能，在下列条件下将停止。

- LIN 帧 ID 域已经收到了.
- 报头错误标志被置起.
- 写 1 到 SLVSYNCF (UART\_LINSTS[3])，重新侦测新的报头.

### Mute 模式 及 LIN 从 Mute 模式退出的条件

Mute 模式下，LIN 从机节点需满足一定的条件时才能接收其它数据。此模式只允许侦测报头，禁止接收任何其它数据。通过设定 MUTE (UART\_LINCTL[4] 使能 Mute 模式，设置 HSEL (UART\_LINCTL[23:22]) 来退出 Mute 模式。

注：建议校验字节发送之后，将 LIN 从机节点设定为 Mute 模式

LIN 从机控制器从 Mute 模式退出描述如下：如果 HSEL (UART\_LINCTL[23:22]) 设为“间隔域”，当 LIN 从机控制器检测到一个有效的 LIN break + 分隔符时，控制器将使能接收器（退出 Mute 模式），随后的数据（sync 数据，帧 ID，响应数据）将被收到 RX-FIFO 中。

如果 HSEL (UART\_LINCTL[23:22]) 设为“间隔域 和 sync 域”，当 LIN 从机控制器检测到一个有效的 LIN break + 分隔符，后面跟一个有效的同步域并且没有帧错误时，控制器将使能接收器（退出 Mute 模式），随后的数据（帧 ID，响应数据）将被收到 RX-FIFO 中。如果 HSEL (UART\_LINCTL[23:22]) 设为“间隔域，sync 域和 ID 域”，当 LIN 从机控制器检测到一个有效的 LIN break + 分隔符和有效的同步域，并且没有帧错误后面跟一个有效的帧 ID 域并且没有帧错误，并且收到的帧 ID 和 PID (UART\_LINCTL[31:24]) 中的值匹配时，控制器将使能接收器（退出 Mute 模

式) 随后的数据(响应数据)将被收到 RX-FIFO 中。

### 非自动重新同步从机模式 (NAR)

用户可以关闭自动重新同步功能来固定通信波特率。当工作在非自动重新同步模式时，软件需要一些初始化过程，初始化过程如下所示：

1. 设定 **UART\_BAUD** 寄存器设定波特率.
2. 设定 **FUNCSEL** (**UART\_FUNCSEL[1:0]**) 为“01”选择 LIN 功能.
3. 设定 **SLVAREN** (**UART\_LINCTL[2]**)= 0 关闭自动重新同步功能.
4. 设定 **SLVEN** (**UART\_LINCTL[0]**) 为 1 使能 LIN 从机模式.

### 自动重新同步从机模式 (AR)

自动重新同步模式下，每次收到同步域时，控制器将调整比特率发生器。软件需要一些初始化过程，初始化过程如下所示：

1. 设定 **UART\_BAUD** 寄存器设定波特率.
2. 设定 **UART\_FUNCSEL** (**UART\_FUNCSEL[1:0]**) 为‘01’选择 LIN 功能模式.
3. 设定 **SLVAREN** (**UART\_LINCTL[2]**) 为 ‘1’使能自动重新同步功能.
4. 设定 **SLVEN** (**UART\_LINCTL[0]**) 为‘1’，使能 LIN 从机模式.

当自动重新同步功能使能后，每个 LIN 间隔域后面，用 LIN 的工作时钟持续采样 5 个下降沿的时间，测量的结果储存在内部一个 13-bit 寄存器中，在第 5 个下降沿结束，**UART\_BAUD** 寄存器的值将被自动更新。如果在 5 个下降沿之前，测量计数器 (13-bit) 溢出，报头错误标志 **SLVHEF** (**UART\_LINSTS [1]**) 将被置位

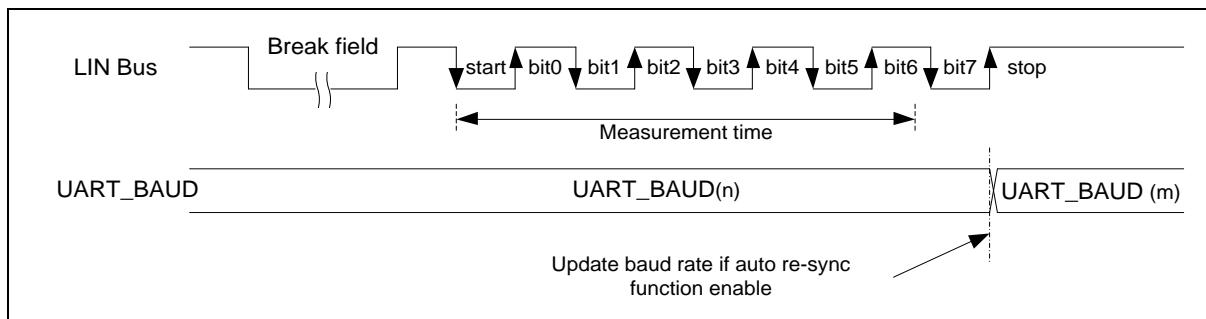


图 6.15-21 LIN 同步域测量

自动重新同步模式 (AR) 下，软件必须通过设定 **UART\_BAUD** 寄存器设定希望的波特率，硬件将保存到内部的 **TEMP\_REG** 寄存器，每次 LIN 间隔域之后，5 个下降沿被采样，测量结果保存到内部 13-bit 寄存器 (**BAUD\_LIN**)，这个结果将自动更新到 **UART\_BAUD** 寄存器。

为了保证传输波特率，每个新的间隔域收到之前，波特率发生器必须重新加载初始值。初始值在初始化的时候由应用程序设定 (**TEMP\_REG**)。用户可以设定 **SLVDUEN** (**UART\_LINCTL [3]**) 来使能自动加载初始波特率功能。如果 **SLVDUEN** (**UART\_LINCTL [3]**)= 1，当前帧结束，收到下一个字符之前，硬件将自动加载初始值到 **UART\_BAUD** 寄存器，**UA\_BAUD** 被更新之后，**SLVDUEN** (**UART\_LINCTL [3]**) 将被自动清 0。LIN 波特率的更新方法如下图所示

**注 1:** 收到检验字符之前，推荐设定 **SLVDUEN** 位为 1。

**注 2:** 值检测到报头错误时, 用户需要对 SLVSYNCF (UART\_LINSTS[3]) 写 1 来重新检测报头。当写 1 到该位时, 硬件将重载初始波特率 (TEMP\_REG) 并重新搜索新的报头。

**注 3:** 自动重新同步模式下, 波特率必须设定为模式 2 (BAUDM1 UART\_BAUD [29]) 和 BAUDM0 (UART\_BAUD[28]) 必须为 1)

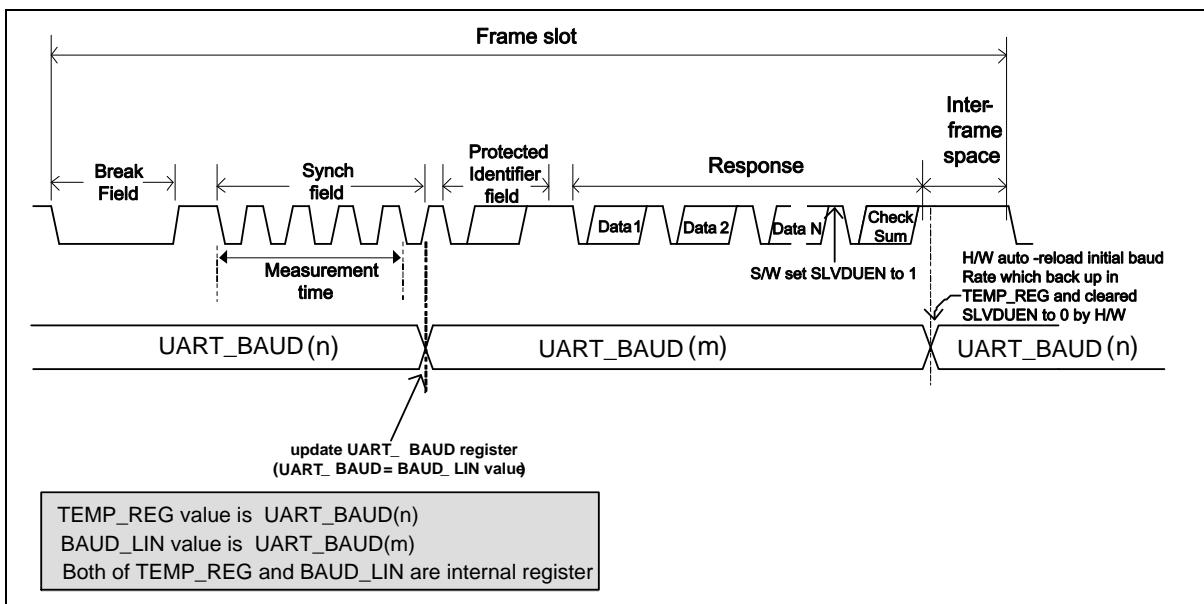


图 6.15-22 AR 模式下, SLVEUEN=1 时, UART\_BAUD 更新顺序

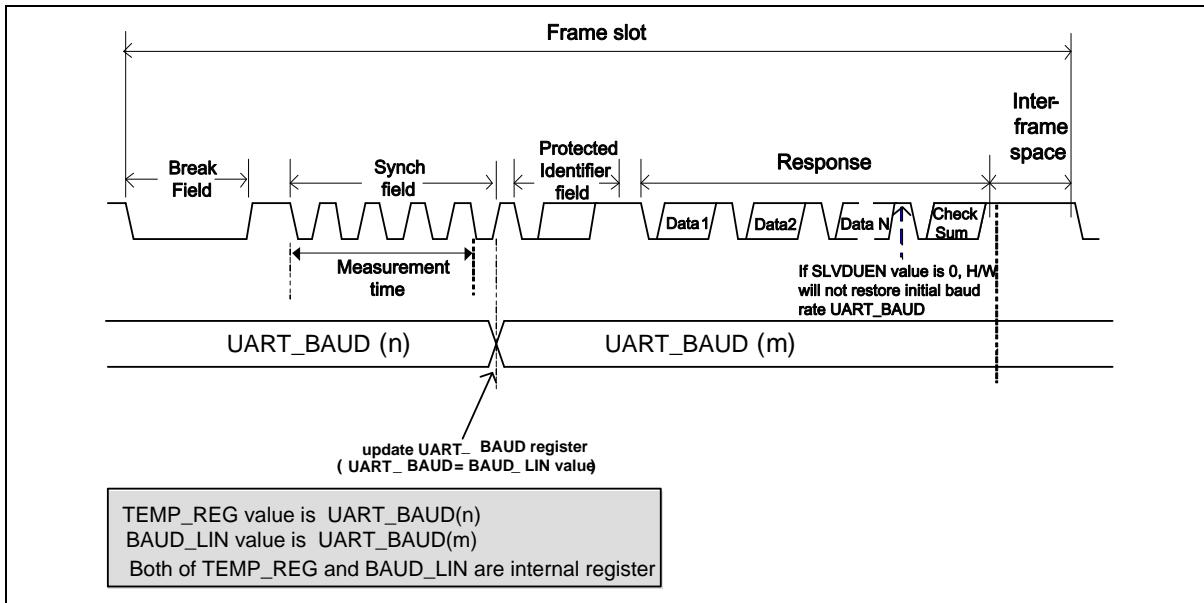


图 6.15-23 AR 模式下, SLVEUEN=0 时, UART\_BAUD 更新顺序 0

### 同步域误差错误

自动重新同步模式下, 控制器将检测同步域的误差错误。误差错误检测比较当前波特率和接收到的同步域的波特率。两个检测被同步执行。

检查 1: 根据同步域的第一个下降沿和最后一个下降沿的测量值。

- 如果误差大于 14.84%, 报头错误标志 SLVHEF (UART\_LINSTS[1]) 将被置位

- 如果误差小于 14.06%, 报头错误标志 SLVHEF (UART\_LINSTS[1]) 不会被置位.
- 如果误差在 14.84% 和 14.06%之间, 报头错误标志 SLVHEF (UART\_LINSTS[1]) 可能被置位也可能没有被置位 (取决于数据失相).

检查 2: 根据同步域的每一个下降沿的测量值.

- 如果误差大于 18.75%,报头错误标志 SLVHEF (UART\_LINSTS[1]) 将被置位.
- 如果误差小于 15.62%,报头错误标志 SLVHEF (UART\_LINSTS[1]) 不会被置位.
- 如果误差在 18.75% 和 15.62%之间, 报头错误标志 SLVHEF (UART\_LINSTS[1])) 可能被置位也可能没有被置位

**注:** 误差检测基于当前波特率时钟。因而，为了保证误差检测的正确性，通过设定 SLVDUEN (UART\_LINCTL[3]), 新的间隔域收到之前，波特率必须重新加载为初始值 (每个校验和收到之前，推荐设定 SLVDUEN (UART\_LINCTL[3]) 为 1)

### LIN 报头错误检测

LIN 从机模式下, 当用户通过设定 SLVHDEN (UART\_LINCTL[1]) 使能报头检测功能时, 硬件将处理报头检测流程。如果报头有错误, LIN 报头错误标志 SLVHEF (UART\_LINSTS[1]) 将被置位, 如果 LINIEN (UART\_INTEN[8])=1, 中断将发生。报头错误被检测到时, 用户必须写 1 到 SLVSYNCF (UART\_LINSTS[3]) 来复位检测电路以重新检测新的报头

如果下列一个条件发生, LIN 报头错误标志 SLVHEF (UART\_LINSTS[1]) 将被置位

- 间隔域分隔符太短 (小于 0.5 比特时间).
- 同步域或者帧 ID 域帧错误.
- 同步域数据不是 0x55 (非自动重新同步模式).
- 同步域误差错误 (自动重新同步模式).
- 同步域测量超时 (自动重新同步模式).

- LIN 报头接收超时.

### 6.15.5.11 RS-485 功能模式

UART 控制器另一个可选择的功能是 RS-485 功能 (用户必须设置 UART\_FUNCSEL [1:0] = '11'来使能 RS-485 功能), 方向控制则由异步串口的 nRTS 脚来控制。RS-485 收发器的驱动控制是通过 nRTS 控制信号来驱动的。RS-485 模式下的 RX 和 TX 大多数特性与 UART 相同

RS-485 模式, 控制器可以配置成 RS-485 可寻址的从机模式, RS-485 主机发送可通过设置检验位 (第 9 位) 为 1 标识地址特性。对于数据字符, 检验位设置为 0。设置寄存器 UART\_LINE 控制第 9 位 (PBE, EPE 和 SPE 置位时, 第 9 位发送 0; PBE 和 SPE 置位, EPE 清零时, 第 9 位发送 1)。

该控制器支持三种操作模式: RS-485 普通多点操作模式 (NMM), RS-485 自动地址识别模式 (AAD) 和 RS-485 自动方向控制模式 (AUD), 可通过 UART\_ALTCTL 寄存器的设置选择其中一种工作模式, 通过设置 DLY (UART\_TOUT [15:8]) 可以设置上一个停止位与下一个开始位之间的延迟时间。.

#### NMM RS-485 普通多点操作模式

RS-485 普通多点操作模式 (RS485NMM (UART\_ALTCTL[8]) = 1)，在检测到地址字节之前的数据会被忽略。如果检测到地址字节 (第 9 位为 1)，会产生一个中断到 CPU，软件可以通过设置 RXOFF (UART\_FIFO[8])，决定是否使能或禁用接收器接收数据。如果使能接收器，就会接收所有字节数据并存储到 RX-FIFO。如果设置 RXOFF (UART\_FIFO [8]) 禁用接收器，会忽略所有接收到的字节数据，直到检测到下一个地址字节。当检测到地址字节后，控制器将清除 RXOFF (UART\_FIFO [8]) 且地址字节数据将存储到 RX-FIFO

#### AAD RS-485 自动地址识别工作模式

RS-485 自动地址识别模式 (RS485AAD (UART\_ALTCTL[9]) = 1)，接收器在检测到地址字节 (第 9 位为 1) 并且地址字节数据与 ADDRMV (UART\_ALTCTL[31:24]) 的值相匹配之前将忽略所有数据。地址字节数据将存储在 RX-FIFO。所有接收字节数据将被接收并存储于 RX-FIFO 直到一个地址字节与 ADDRMV (UART\_ALTCTL[31:24]) 的值不匹配

#### AUD RS-485 自动方向控制功能

RS-485 控制器的另一个功能是 RS-485 自动方向控制功能 (RS485AUD (UART\_ALTCTL[10]) = 1)。RS-485 通过 nRTS 驱动控制异步串口的控制信号，使能 RS-485 驱动器。nRTS 连接到 RS-485 驱动器，设置 nRTS 线为高 (逻辑 1) 使能 RS-485 驱动器。设置 nRTS 为低 (逻辑 0)，使驱动器进入 tri-state 状态。用户通过设置寄存器 UART\_MODEM 中的 RTSACTLV 改变 nRTS 驱动电平

图 7.15 24 展示了 AUD 模式下 RS-485 nRTS 驱动电平。nRTS 脚在 TX 数据发送阶段自动驱动收发器。

设置 RTSACTLV (UART\_MODEM[9]) 可以控制 nRTS 脚的输出电平。用户可以通过读 RTSSTS (UART\_MODEM[13]) 来获取 nRTS 脚上实际的输出逻辑电平。

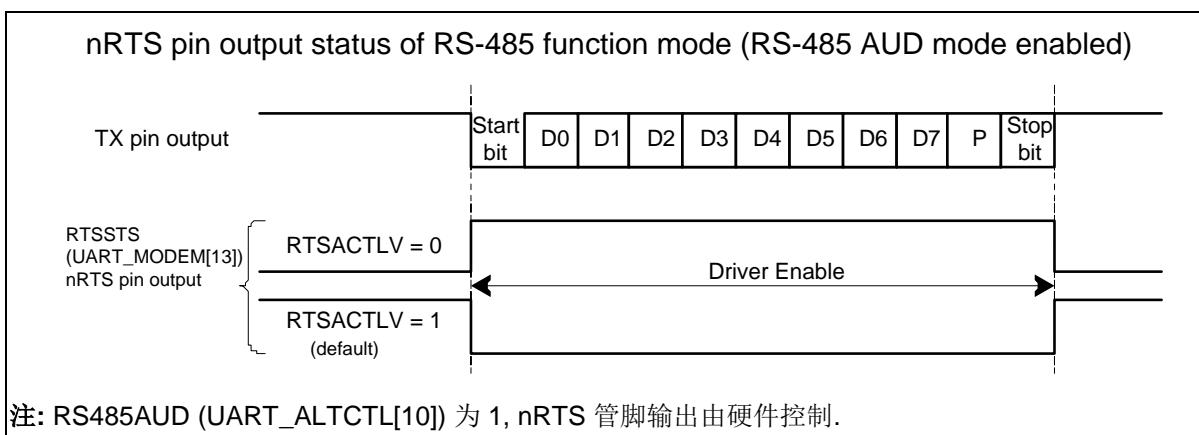


图 6.15-24 RS-485 nRTS 自动方向控制模式下 nRTS 管脚驱动电平

图 7.15 25 展示了通过软件控制 (RS485AUD (UART\_ALTCTL[10])=0) 模式下，RS-485 nRTS 脚的驱动电平。nRTS 驱动电平通过设置 RTS (UART\_MODEM[1]) 来控制。

设置 RTSACTLV (UART\_MODEM[9]) 可以控制 nRTS 脚的输出与 RTS (UART\_MODEM[1]) 设置值是否相反。用户可以读 RTSSTS (UART\_MODEM[13]) 来获取 nRTS 脚上实际的逻辑电平。RS-485 的帧结构参见图 7.15 26。

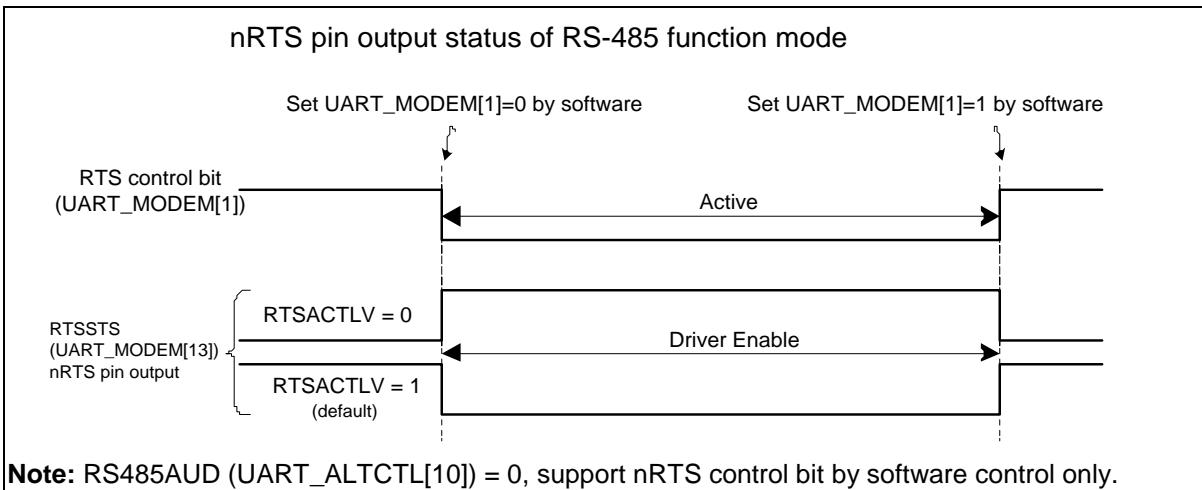


图 6.15-25 RS-485 nRTS 软件控制时的驱动电平

编程流程示例：

1. 设置 UART\_FUNCSEL 中的 FUN\_SEL 选择 RS-485 功能.
2. 设置 RXOFF (UART\_FIFO[8])，使能或禁用 RS-485 接收器.
3. 设置 RS485NMM (UART\_ALTCTL[8]) 或 RS485AAD (UART\_ALTCTL[9]) 模式.
4. 如果选择 RS485AAD (UART\_ALTCTL[9]) 模式，ADDRMV (UART\_ALTCTL[31:24]) 需设置成自动地址匹配值.
1. 设置 RS485\_AUD (UA\_ALT\_CSR[10]) 来决定是否为自动方向控制).

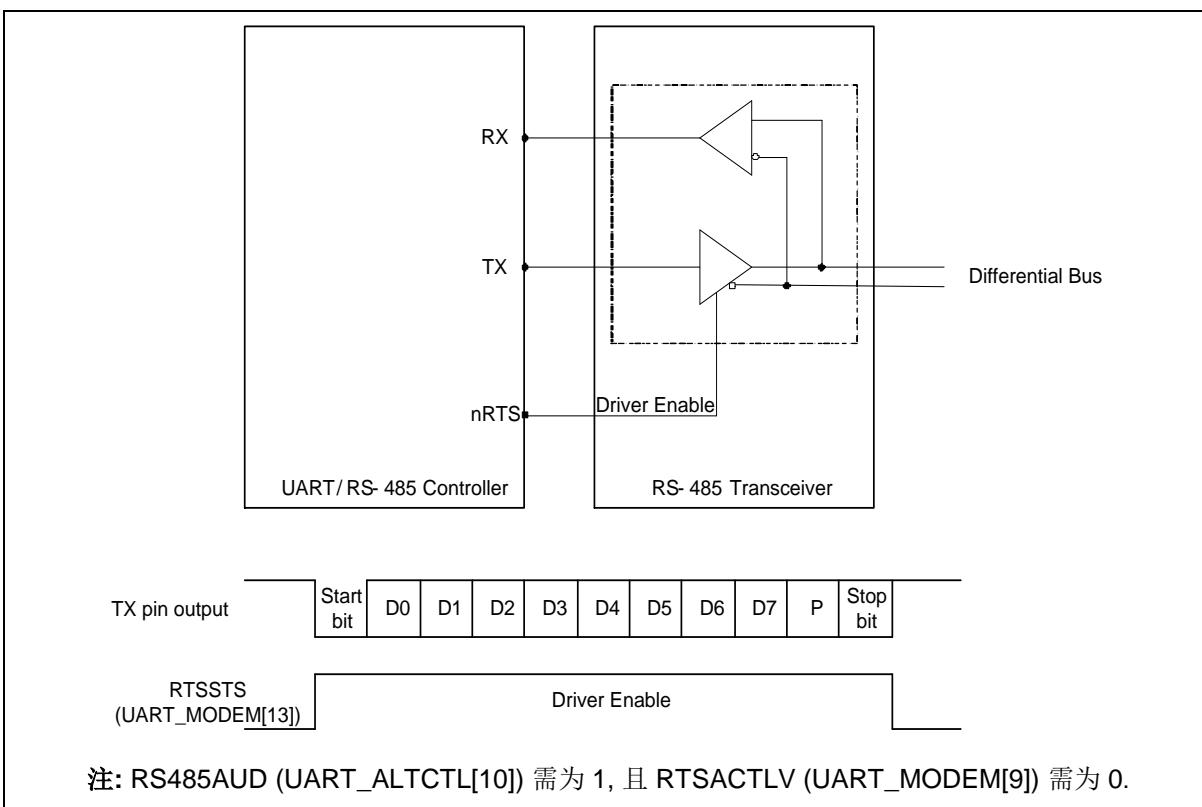


图 6.15-26 RS-485 帧结构

#### 6.15.5.12 PDMA 传输功能

UART 控制器支持 PDMA 传输功能。

设定 PDMA 相关参数并设定 UART\_DAT 为 PDMA 目标地址后，当 TXPDMAEN (UART\_INTEN[14]) 为 1，UART 控制器会请求 PDMA 控制器自动开始 PDMA 传输。

设定 PDMA 相关参数并设定 UART\_DAT 为 PDMA 为源地址后，当 RXPDMAEN (UART\_INTEN[15]) 为 1，UART 控制器会请求 PDMA 控制器在 RX FIFO 缓存有数据时，自动开始 PDMA 传输。

**注：**如果 配置了 STOPn (PDMA\_STOP[n]) 寄存器停止 UART RXPDMA 任务，且 UART 还未完成接收，UART 控制器会完成传输并存储当前的接收数据到接收缓存。读寄存器 RXEMPTY (UART\_FIFOSTS[14]) 可以知道当前接收缓存有无有效数据。

### 6.15.6 寄存器映射

R: 只读, W: 只写, R/W: 读/写

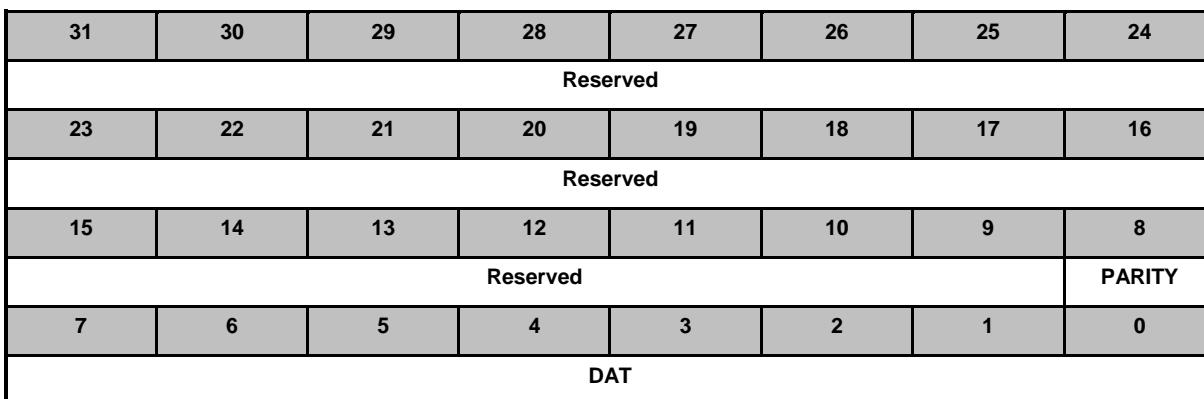
寄存器	偏移量	R/W	描述	复位值
<b>UART 基址:</b>				
<b>UARTx_BA = 0x4007_0000 + (0x1000 * x)</b>				
<b>x=0,1,2,3,4,5</b>				
<b>UART_DAT</b> <b>x=0,1,2,3,4,5</b>	UARTx_BA+0x00	R/W	UART 接收/发送缓存寄存器	未定义
<b>UART_INTEN</b> <b>x=0,1,2,3,4,5</b>	UARTx_BA+0x04	R/W	UART 中断使能寄存器	0x0000_0000
<b>UART_FIFO</b> <b>x=0,1,2,3,4,5</b>	UARTx_BA+0x08	R/W	UART FIFO 控制寄存器	0x0000_0101
<b>UART_LINE</b> <b>x=0,1,2,3,4,5</b>	UARTx_BA+0x0C	R/W	UART 线控寄存器	0x0000_0000
<b>UART_MODEM</b> <b>x=0,1,2,3,4,5</b>	UARTx_BA+0x10	R/W	UART Modem 控制寄存器	0x0000_0200
<b>UART_MODEM_STS</b> <b>x=0,1,2,3,4,5</b>	UARTx_BA+0x14	R/W	UART Modem 状态寄存器	0x0000_0110
<b>UART_FIFOSTS</b> <b>x=0,1,2,3,4,5</b>	UARTx_BA+0x18	R/W	UART FIFO 状态寄存器	0xB040_4000
<b>UART_INTSTS</b> <b>x=0,1,2,3,4,5</b>	UARTx_BA+0x1C	R/W	UART 中断状态寄存器	0x0040_0002
<b>UART_TOUT</b> <b>x=0,1,2,3,4,5</b>	UARTx_BA+0x20	R/W	UART 超时寄存器	0x0000_0000
<b>UART_BAUD</b> <b>x=0,1,2,3,4,5</b>	UARTx_BA+0x24	R/W	UART 波特率分频器	0x0F00_0000
<b>UART_IRDA</b> <b>x=0,1,2,3,4,5</b>	UARTx_BA+0x28	R/W	UART IrDA 控制寄存器	0x0000_0040
<b>UART_ALTCTL</b> <b>x=0,1,2,3,4,5</b>	UARTx_BA+0x2C	R/W	UART 选择控制/状态寄存器	0x0000_000C
<b>UART_FUNCS_EL</b> <b>x=0,1,2,3,4,5</b>	UARTx_BA+0x30	R/W	UART 功能选择寄存器	0x0000_0000
<b>UART_LINCTL</b> <b>x=0,1</b>	UARTx_BA+0x34	R/W	UART LIN 控制寄存器	0x000C_0000
<b>UART_LINSTS</b> <b>x=0,1</b>	UARTx_BA+0x38	R/W	UART LIN 状态寄存器	0x0000_0000

<b>UART_BRCOMP</b> <b>x=0,1,2,3,4,5</b>	UARTx_BA+0x3C	R/W	UART 波特率补偿寄存器	0x0000_0000
<b>UART_WKCTL</b> <b>x=0,1,2,3,4,5</b>	UARTx_BA+0x40	R/W	UART 唤醒控制寄存器	0x0000_0000
<b>UART_WKSTS</b> <b>x=0,1,2,3,4,5</b>	UARTx_BA+0x44	R/W	UART 唤醒状态寄存器	0x0000_0000
<b>UART_DWKCOM</b> <b>x=0,1,2,3,4,5</b>	UARTx_BA+0x48	R/W	UART 输入数据唤醒补偿寄存器	0x0000_0000

### 6.15.7 寄存器描述

#### UART\_DAT    UART 接收/发送缓存寄存器

寄存器	偏移量	R/W	描述	复位值
<b>UART_DAT</b> <b>x=0,1,2,3,4,5</b>	UARTx_BA+0x00	R/W	UART 接收/发送缓存寄存器	Undefined



位	描述	
[31:9]	<b>Reserved</b>	保留
[8]	<b>PARITY</b>	<p>校验位接收/发送缓存 写操作: 写该位，校验位会被存入发送 FIFO。 如果 PBE (UART_LINE[3]) 和 PSS (UART_LINE[7]) 置位，UART 控制器会通过 UART_TXD 发送 DAT (UART_DAT[7:0]) 及该位 如果 PBE (UART_LINE[3]) 和 PSS (UART_LINE[7]) 使能，可以读这位获得校验位 注: 仅 PBE (UART_LINE[3]) 和 PSS (UART_LINE[7]) 置位时，该位有效。</p>
[7:0]	<b>DAT</b>	<p>数据接收/发送缓存 写操作: 写数据到该寄存器，数据将会保存到发送 FIFO. UART 控制器将会通过 UART_TXD 把存放在 FIFO 中最前面的数据发送出去 读操作: 读该寄存器，UART 将返回从接收 FIFO 中接收到的 8 位数据 (LSB 优先)</p>

UART\_INTEN UART 中断使能寄存器

寄存器	偏移量	R/W	描述	复位值
UART_INTEN x=0,1,2,3,4,5	UARTx_BA+0x04	R/W	UART 中断使能寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved	TXENDIEN	Reserved			ABRIEN	Reserved	
15	14	13	12	11	10	9	8
RXPDMAEN	TXPDMAEN	ATOCTSEN	ATORTSEN	TOCNTEN	Reserved		LINIEN
7	6	5	4	3	2	1	0
Reserved	WKIEN	BUFERRIEN	RXTOIEN	MODEMIEN	RLSIEN	THREIEN	RDAIEN

位	描述	
[31:23]	Reserved	保留
[22]	TXENDIEN	<b>发送完成中断使能位</b> 使能 TXENDIEN (UART_INTEN[22]), 若 TXENDIF (UART_INTSTS[22]) 置位, (TX FIFO (UART_DAT) 为空且最后一字节的停止位已发送, 则会产生发送完成中断 0 = 禁止发送完成中断 1 = 使能发送完成中断
[21:19]	Reserved	保留
[18]	ABRIEN	<b>自动波特率中断使能位</b> 0 = 禁止自动波特率中断 1 = 使能自动波特率中断
[17:16]	Reserved	保留
[15]	RXPDMAEN	<b>RX PDMA 使能位</b> 该位可以使能或禁止 RX PDMA 功能. 0 = 禁止 RX PDMA. 1 = 使能 RX PDMA. 注:如果 RLSIEN (UART_INTEN[2]) 为使能 并且 HWRLSINT (UART_INTSTS[26])=1, RLS (接收线状态) 中断将产生。如果 RLS 中断是因为分隔错误标志 BIF (UART_FIFOSTS[6]), 帧错误标志 FEF (UART_FIFO[5]) 或是奇偶校验错误标志 PEF (UART_FIFOSTS[4]) 产生, UART PDMA 接收请求将被停止。写“1”到错误标志 BIF、FEF 和 PEF, 分别清除分隔错误标志 BIF、帧错误标志 FEF 和奇偶校验错误标志 PEF 后, UART PDMA 接收请求将继续进行
[14]	TXPDMAEN	<b>TX PDMA 使能位</b> 该位可以使能或禁止 TX PDMA 功能. 0 = 禁止 TX PDMA.

		1 = 使能 TX PDMA.  注: 如果 RLSIEN (UART_INTEN[2]) 为使能 并且 HWRLSINT (UART_INTSTS[26])=1, RLS (接收线状态) 中断将产生。如果 RLS 中断是因为 break 错误标志 BIF (UART_FIFOSTS[6]), 帧错误标志 FEF (UART_FIFO[5]) 或是奇偶校验错误标志 PEF (UART_FIFOSTS[4]) 产生, UART PDMA 接收请求将被停止。写“1”到错误标志 BIF、FEF 和 PEF, 分别清除 break 错误标志 BIF、帧错误标志 FEF 和奇偶校验错误标志 PEF 后, UART PDMA 接收请求将继续进行。
[13]	ATOCTSEN	<b>nCTS 自动流控使能位</b> 0 = 禁止 nCTS 自动流控. 1 = 使能 nCTS 自动流控.  注: 当 nCTS 自动流控使能后, 当 nCTS 输入有效, UART 会发送数据到外部设备 (UART 将不会发送数据到外部设备直到 nCTS 有效).
[12]	ATORTSEN	<b>nRTS 自动流控使能位</b> 0 = 禁止 nRTS 自动流控. 1 = 使能 nRTS 自动流控.  注: 当 nRTS 自动流控使能后, 如果 RX FIFO 中字节的数量等于 RTSTRGLV (UART_FIFO[19:16]), UART 会自动禁止 nRTS 信号
[11]	TOCNTEN	<b>接收缓存超时计数器使能位</b> 0 = 禁止接收缓存超时计数器 1 = 使能接收缓存超时计数器
[10:9]	Reserved	保留
[8]	LINIEN	<b>LIN 总线中断使能位</b> 0 = 禁止 LIN 总线中断. 1 = 使能 LIN 总线中断.  注: 该位 LIN 模式下有效.
[7]	Reserved	保留
[6]	WKIEN	<b>唤醒中断使能位</b> 0 = 禁止 唤醒中断 1 = 使能 唤醒中断.
[5]	BUFERRIEN	<b>缓存错误中断使能位</b> 0 = 禁止缓存错误中断. 1 = 使能缓存错误中断.
[4]	RXTOIEN	<b>RX 超时中断使能位</b> 0 = 禁止 RX 超时中断. 1 = 使能 RX 超时中断.
[3]	MODEMIEN	<b>Modem 状态中断使能位</b> 0 = 禁止 Modem 状态中断. 1 = 使能 Modem 状态中断.
[2]	RLSIEN	<b>接收 Line 状态中断使能位</b> 0 = 禁止接收 Line 状态中断. 1 = 使能接收 Line 状态中断.
[1]	THREIEN	<b>发送保持寄存器空中断使能位</b> 0 = 禁止发送保持寄存器空中断.

		1 = 使能发送保持寄存器空中断.
[0]	RDAIEN	<b>接收数据可用中断使能位</b> 0 = 禁止接收数据可用中断. 1 = 使能接收数据可用中断.

## UART\_FIFO UART FIFO 控制寄存器

寄存器	偏移量	R/W	描述	复位值
UART_FIFO x=0,1,2,3,4,5	UARTx_BA+0x08	R/W	UART FIFO 控制寄存器	0x0000_0101

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved				RTSTRGLV			
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
RFITL				Reserved	TXRST	RXRST	Reserved

位	描述	
[31:20]	Reserved	保留
[19:16]	RTSTRGLV	<b>nRTS 自动流控触发阈值</b> 0000 = nRTS 触发阈值为 1 字节. 0001 = nRTS 触发阈值为 4 字节. 0010 = nRTS 触发阈值为 8 字节. 0011 = nRTS 触发阈值为 14 字节. Others = 保留. <b>注:</b> 该区域用于自动 nRTS 流控制
[15:9]	Reserved	保留
[8]	RXOFF	<b>接收禁止位</b> 是否禁用接收 (置 1 禁用接收) 0 = 使能接收 1 = 禁用接收 <b>注:</b> 该位用于 RS-485 普通多点模式,需要在 RS485NMM (UART_ALTCTL [8]) 被设置之前设置
[7:4]	RFITL	<b>RX FIFO 中断触发阈值</b> 当 FIFO 接收字节数等于 RFITL 后, RDAIF (UART_INTSTS[0]) 将被置位 (如果 RDAIEN (UART_INTEN [0]) 使能, 将产生中断)。 0000 = RX FIFO 触发中断阈值为 1 字节 0001 = RX FIFO 触发中断阈值为 4 字节 0010 = RX FIFO 触发中断阈值为 8 字节 0011 = RX FIFO 触发中断阈值为 14 字节 其它 = 保留
[3]	Reserved	保留

[2]	<b>TXRST</b>	<b>TX 域软件复位</b> 当 TXRST (UART_FIFO[2]) 置位，发送 FIFO 和 TX 内部状态机中的所有数据将被清除。 0 = 无效 1 = 复位 TX 内部状态机和指针. <b>注 1:</b> 最少 3 个 UART 外设时钟后，该位自动清 0 . <b>注 2:</b> 设置该位前，需要先等待 TXEMPTYF (UART_FIFOSTS[28]) 被置位
[1]	<b>RXRST</b>	<b>RX 域软件复位</b> 当 RXRST (UART_FIFO[1]) 置位，发送 FIFO 和 RX 内部状态机中的所有数据将被清除。 0 = 无效 1 = 复位 RX 内部状态机和指针. <b>注 1:</b> 最少 3 个 UART 外设时钟后，该位自动清 0 <b>注 2:</b> 设置该位前，需要先等待 RXEMPTYF (UART_FIFOSTS[29]) 被置位
[0]	<b>Reserved</b>	保留

## UART LINE UART 线控寄存器

寄存器	偏移量	R/W	描述	复位值
UART_LINE x=0,1,2,3,4,5	UARTx_BA+0x0C	R/W	UART 线控寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved						RXDINV	TXDINV
7	6	5	4	3	2	1	0
PSS	BCB	SPE	EPE	PBE	NSB	WLS	

位	描述	
[31:10]	Reserved	保留
[9]	RXDINV	<p><b>RX 数据反转</b>            0 = 禁止接收数据信号反转.            1 = 使能接收数据信号反转.</p> <p><b>注 1:</b> 设置该位前，需要先置位 TXRXDIS (UART_FUNCSEL[3])，再等 TXRXACT (UART_FIFOSTS[31]) 被清除。完成配置后，清除 TXRXDIS (UART_FUNCSEL[3]) 以触发 UART 控制器</p> <p><b>注 2:</b> 该位仅在 FUNCSEL (UART_FUNCSEL[1:0]) 配置为 UART, LIN 或 RS485 模式下有效</p>
[8]	TXDINV	<p><b>TX 数据反转</b>            0 = 禁止发送数据信号反转.            1 = 使能发送数据信号反转.</p> <p><b>注 1:</b> 设置该位前，需要先置位 TXRXDIS (UART_FUNCSEL[3])，再等 TXRXACT (UART_FIFOSTS[31]) 被清除。完成配置后，清除 TXRXDIS (UART_FUNCSEL[3]) 以触发 UART 控制器</p> <p><b>注 2:</b> 该位仅在 FUNCSEL (UART_FUNCSEL[1:0]) 配置为 UART, LIN 或 RS485 模式下有效</p>
[7]	PSS	<p><b>校验位源选择</b>            校验位可以由软件或是自动产生并校验</p> <p>0 = 校验位由设置EPE (UART_LINE[4]) 和 SPE (UART_LINE[5])并自动检查.            1 = 校验位由软件产生并校验.</p> <p><b>注 1:</b> 仅 PBE (UART_LINE[3]) 置位时，该位有效.</p> <p><b>注 2:</b> 如果 PSS 为 0，校验位自动发送并检查。如果 PSS 为 1，发送出去的校验位可由 PARITY (UART_DAT[8]) 设置，读 PARITY (UART_DAT[8]) 可得校验位</p>
[6]	BCB	<b>Break 控制位</b> 0 = 禁止Break 控制

		<p><b>1 = 使能Break 控制</b></p> <p><b>注:</b> 当该位被置逻辑 1, 串行数据输出 (TX) 将强制到 Spacing 状态 (logic 0)。该位仅作用于 TX, 对传输逻辑不起作用</p>
[5]	<b>SPE</b>	<p><b>Stick 校验使能位</b></p> <p>0 = 禁止Stick 校验.</p> <p>1 = 使能Stick 校验.</p> <p><b>注:</b> 如果 PBE (UART_LINE[3]) 和 EPE (UART_LINE[4]) 为逻辑 1, 校验位发送和检验值为逻辑 0。如果 PBE (UART_LINE[3]) 是1 , EPE (UART_LINE[4]) 是 0, 则校验位发送和检验值为 1。</p>
[4]	<b>EPE</b>	<p><b>偶校验使能位</b></p> <p>0 = 逻辑 1 的奇数数目在每个字节中被发送和检验</p> <p>1 = 逻辑 1 的偶数数目在每个字节中被发送和检验</p> <p><b>注:</b> 该位只在 PBE (UART_LINE[3]) 置位时有效。</p>
[3]	<b>PBE</b>	<p><b>校验使能位</b></p> <p>0 = 禁止生成校验位</p> <p>1 = 使能生成校验位</p> <p><b>注:</b> 每一个发送字符中都产生校验位, 对每一个传进来的数据进行校验位检测</p>
[2]	<b>NSB</b>	<p><b>停止位数目</b></p> <p>0= 当发送数据时, 产生 1 个“STOP bit”</p> <p>1= 当发送数据, 选择 5-位 字长度时, 产生 1.5 “STOP bit” 当选择 6-, 7- 和 8-位字长度时, 产生 2 个“STOP bit”.</p>
[1:0]	<b>WLS</b>	<p><b>字长选择</b></p> <p>该域选择 UART 的字长.</p> <p>00 = 5 位.</p> <p>01 = 6 位.</p> <p>10 = 7 位.</p> <p>11 = 8 位.</p>

## UART MODEM    UART Modem 控制寄存器

寄存器	偏移量	R/W	描述	复位值
UART_MODEM x=0,1,2,3,4,5	UARTx_BA+0x10	R/W	UART Modem 控制寄存器	0x0000_0200

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved		RTSSTS	Reserved			RTSACTLV	Reserved
7	6	5	4	3	2	1	0
Reserved						RTS	Reserved

位	描述
[31:14]	<b>Reserved</b> 保留
[13]	<b>RTSSTS</b> <b>nRTS 管脚状态 (只读)</b> 该位的值对应于 nRTS 管脚的输出电平 0 = nRTS 管脚为低电平逻辑状态 1 = nRTS 管脚为高电平逻辑状态.
[12:10]	<b>Reserved</b> 保留
[9]	<b>RTSACTLV</b> <b>nRTS 脚的有效电平</b> 该位定义了 nRTS 输出管脚的有效电平 0 = nRTS 管脚输出为高电平有效. 1 = nRTS 管脚输出为低电平有效.(默认) <b>注 1:</b> UART 的功能模式请参考图 6.15-13 和 图 6.15-14. <b>注 2:</b> RS-485 功能模式请参考图 6.15-24 和 图 6.15-25. <b>注 3:</b> 设置该位前, 需要先置位 TXRXDIS (UART_FUNCSEL[3]), 再等 TXRXACT (UART_FIFOSTS[31]) 被清除。完成配置后, 清除 TXRXDIS (UART_FUNCSEL[3]) 以触发 UART 控制器
[8:2]	<b>Reserved</b> 保留
[1]	<b>RTS</b> <b>nRTS (请求发送) 信号控制</b> 该位直接控制内部 nRTS 脚信号是否有效, 然后使用 RTSACTLV 位的配置驱动 nRTS 脚输出 0 = nRTS 信号有效. 1 = nRTS 信号无效. <b>注 1:</b> UART 功能模式下, 当 nRTS 自动流控被使能后, nRTS 信号控制位无效 <b>注 2:</b> RS-485 模式下, 当 RS-485 自动方向模式 (AUD) 被使能后, nRTS 信号控制位无效
[0]	<b>Reserved</b> 保留



## UART MODEMSTS UART Modem 状态寄存器

寄存器	偏移量	R/W	描述	复位值
UART_MODEMSTS x=0,1,2,3,4,5	UARTx_BA+0x14	R/W	UART Modem 状态寄存器	0x0000_0110

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							CTSACTLV
7	6	5	4	3	2	1	0
Reserved			CTSSTS	Reserved			CTSDETF

位	描述	
[31:9]	Reserved	保留
[8]	CTSACTLV	<p><b>nCTS 管脚有效电平</b>            该位定义了 nCTS 输入管脚脚的有效电平            0 = nCTS 输入脚高电平有效            1 = nCTS 输入脚低电平有效.(默认)  <b>注:</b> 设置该位前, 需要先置位 TXRXDIS (UART_FUNCSEL[3]), 再等 TXRXACT (UART_FIFOSTS[31]) 被清除。完成配置后, 清除 TXRXDIS (UART_FUNCSEL[3]) 以触发 UART 控制器</p>
[7:5]	Reserved	保留
[4]	CTSSTS	<p><b>nCTS 管脚状态 (只读)</b>            该位对应于 nCTS 管脚输入逻辑状态            0 = nCTS 管脚输入状态为低电平            1 = nCTS 管脚输入状态为高电平  <b>注:</b> 当 UART 控制器外设时钟被使能, 且 nCTS 多功能管脚被使能, 该位才有效。</p>
[3:1]	Reserved	保留
[0]	CTSDETF	<p><b>检测到 nCTS 状态改变标志</b>            如果 nCTS 输入脚上有电平变化, 该位将被置位, 如果 MODEMIEN (UART_INTEN [3]) 位被置位, 将会产生 Modem 中断。            0 = nCTS 输入管脚状态无变化.            1 = nCTS 输入管脚状态已变化.  <b>注:</b> 该位写 1 清 0 .</p>

## UART\_FIFOSTS UART FIFO 状态寄存器

寄存器	偏移量	R/W	描述	复位值
UART_FIFOSTS x=0,1,2,3,4,5	UARTx_BA+0x18	R/W	UART FIFO 状态寄存器	0xB040_4000

31	30	29	28	27	26	25	24
TXRXACT	Reserved	RXIDLE	TXEMPTYF	Reserved			TXOVIF
23	22	21	20	19	18	17	16
TXFULL	TXEMPTY	TXPTR					
15	14	13	12	11	10	9	8
RXFULL	RXEMPTY	RXPTR					
7	6	5	4	3	2	1	0
Reserved	BIF	FEF	PEF	ADDRDETF	ABRDTOIF	ABRDIF	RXOVIF

位	描述
[31]	<b>TXRXACT</b>  <b>TX 和 RX 有效状态 (只读)</b> 该位反映 TX 和 RX 是否有效 0 = TX 和 RX 无效. 1 = TX 和 RX 有效.(默认)  注: 当 TXRXDIS (UART_FUNCSEL[3]) 置位, TX 和 RX 都空闲时, 该位清 0。UART 控制器此时无法发送或接收数据。其他情况, 该位均为 1.
[30]	<b>Reserved</b>
[29]	<b>RXIDLE</b>  <b>RX 空闲状态 (只读)</b> RX 空闲时, 硬件自动置位该位. 0 = RX 忙. 1 = RX 空闲 (默认)
[28]	<b>TXEMPTYF</b>  <b>发送空标志 (只读)</b> 当 TX FIFO (UART_DAT) 为空, 并且最后一个字节的 STOP 位也已经被发送完毕, 该位被硬件置 1. 0 = TX FIFO 不为空或最后一个字节的 STOP 位还没有发送. 1 = TX FIFO 为空而且最后一个字节的 STOP 位已经发送  注: 当 TX FIFO 不为空或最后一个字节的 STOP 位还没有被发送完毕, 那么该位将被自动清零
[27:25]	<b>Reserved</b>
[24]	<b>TXOVIF</b>  <b>TX 溢出错误中断标志 (只读)</b> 如果 TX FIFO (UART_DAT) 满, 如果再向 UART_DAT 写入数据, 将会导致此位被置 1.. 0 = TX FIFO 未溢出 1 = TX FIFO 已溢出..  注: 该位写 1 清 0.
[23]	<b>TXFULL</b>  <b>发送 FIFO 满 (只读)</b> 此位用于指示 TX FIFO 是否已满.. 0 = TX FIFO 未满

		1 = TX FIFO 已满。 注: 当 TX FIFO Buffer 中的字节数量等于 16, 此位将被置 1. 否则被硬件清零.
[22]	<b>TXEMPTY</b>	<b>发送 FIFO 空 (只读)</b> 此位指示 TX FIFO 是否为空。 0 = TX FIFO 不为空. 1 = TX FIFO 为空. 注: 当 TX FIFO 中的最后一个字节被发送到发送移位寄存器, 硬件将把此位置 1. 当写入数据到 UART_DAT 中, (TX FIFO 不为空), 此位被清零
[21:16]	<b>TXPTR</b>	<b>TX FIFO 指针 (只读)</b> 此位指示 TX FIFO 缓冲区指针位置。当 CPU 写一个字节到 UART_DAT 寄存器, TXPTR 将累加 1. 当 TX FIFO 发送一个字节到发送移位寄存器中, TXPTR 指针将减 1. TXPTR 显示的最大值是 15。当 TX FIFO 缓冲区所填充数据数量达到 16, TXFULL 将被置 1, TXPTR 显示为0。如果 TX FIFO 中发送一个字节到发送移位寄存器, TXFULL 位将被清零, TXPTR 显示 15。
[15]	<b>RXFULL</b>	<b>接收 FIFO 满 (只读)</b> 该位指示 RX FIFO 是否已满。 0 = RX FIFO 未满 1 = RX FIFO 已满 注: 当 RX FIFO 缓冲区中的数据数量等于 16 后, 此位将被置 1, 否则被硬件清零。
[14]	<b>RXEMPTY</b>	<b>接收 FIFO 空 (只读)</b> 此位指示 RX FIFO 是否为空。 0 = RX FIFO 不为空. 1 = RX FIFO 为空. 注: 当 RX FIFO 中最后一个字节被 CPU 读取后, 硬件将对此位置 1, UART 接收到新数据后此位将被清零。
[13:8]	<b>RXPTR</b>	<b>RX FIFO 指针 (只读)</b> 此位指示 RX FIFO 缓冲区指针。当 UART 从外部设备接收到一个字节, RXPTR 将累加 1. 当 RX FIFO 的数据被 CPU 读取一个字节, RXPTR 将递减 1. RXPTR 显示的最大值是 15。当 RX FIFO 的数据达到 16, RXFULL 位将被置 1, RXPTR 显示 0, RX FIFO 当中的数据被 CPU 读取一个后, RXFULL 将被清零, RXPTR 显示 15。
[7]	<b>Reserved</b>	保留
[6]	<b>BIF</b>	<b>Break 中断标志位 (只读)</b> 每当接收到数据输入 (RX) 维持在“空状态”(电平 0) 的时间长于一个全字的传输时间 (即开始位 + 数据位 + 校验位 + 停止位 的总时间), 该位置 1。当 CPU 向该位写 1, 该位重置。 0 = 没有 Break 中断产生. 1 = 有 Break 中断产生. 注: 该位写 1 清 0 .
[5]	<b>FEF</b>	<b>帧错误标志 (只读)</b> 每当接收到的字符没有有效的“停止位”(及跟在最后的数据位后或奇偶校验位后的数据为 0) 该位置 1。 0 = 没有帧错误产生 1 = 有帧错误产生. 注: 该位写 1 清 0 .
[4]	<b>PEF</b>	奇偶校验错误标志 (只读)

		每当接收到的字符没有有效的奇偶校验位，该位置 1。当 CPU 向该位写 1，该位重置。 0 = 没有奇偶校验错误产生 1 = 有奇偶校验错误产生 <b>注:</b> 该位写 1 清 0 .
[3]	<b>ADDRDETF</b>	<b>RS-485 地址位检测标志 (只读)</b> 0 = 接收到的数据没有地址位标记 (bit 9 ='0'). 1 = 接收到的数据有地址位标记 (bit 9 ='1'). <b>注 1:</b> 此位用于 RS-485 功能模式，且 ADDRDEN (UART_ALTCTL[15]) 位被置 1 使能地址检测模式 <b>注 2:</b> 该位写 1 清 0 .
[2]	<b>ABRDTOIF</b>	<b>自动波特率检测超时中断 (只读)</b> 0 = 自动波特率计数器没有溢出. 1 = 自动波特率计数器溢出. <b>注:</b> 该位写 1 清 0 .
[1]	<b>ABRDIF</b>	<b>自动波特率检测中断</b> 当自动波特率检测完成时该位将被置为“1”. 0 = 自动波特率检测还没有完成. 1 = 自动波特率检测已经完成 <b>注:</b> 该位写 1 清 0 .
[0]	<b>RXOVIF</b>	<b>RX 溢出错误标志 (只读)</b> 当 RX FIFO 溢出时被置 1 如果接收到的数据数量大于 RX_FIFO (UART_DAT) 16 字节，该位将被置位. 0 = RX FIFO 未溢出. 1 = RX FIFO 溢出 <b>注:</b> 该位写 1 清 0 .

## UART\_INTSTS UART 中断状态寄存器

寄存器	偏移量	R/W	描述	复位值
UART_INTSTS x=0,1,2,3,4,5	UARTx_BA+0x1C	R/W	UART 中断状态寄存器	0x0040_0002

31	30	29	28	27	26	25	24
ABRINT	TXENDINT	HWBUFEINT	HWTOINT	HWMODINT	HWRLSINT	Reserved	
23	22	21	20	19	18	17	16
Reserved	TXENDIF	HWBUFEIF	HWTOIF	HWMODIF	HWRLSIF	Reserved	
15	14	13	12	11	10	9	8
LININT	WKINT	BUFERRINT	RXTOINT	MODEMINT	RLSINT	THREINT	RDAINT
7	6	5	4	3	2	1	0
LINIF	WKIF	BUFERRIF	RXTOIF	MODEMIF	RLSIF	THREIF	RDAIF

位	描述
[31]	<b>ABRINT</b>  自动波特率中断指示 (只读) ABRIEN (UART_INTEN[18]) 和 ABRIF (UART_ALTCTL[17]) 都为 1, 该位置 1. 0 = 无自动波特率中断产生. 1 = 有自动波特率中断产生.
[30]	<b>TXENDINT</b>  发送空中断指示器 (只读) TXENDIEN (UART_INTEN[22]) 和 TXENDIF (UART_INTSTS[22]) 都为 1, 该位置 1. 0 = 无发送空中断发生. 1 = 有发送空中断发生.
[29]	<b>HWBUFEINT</b>  PDMA 模式缓存错误中断 (只读) BUFERRIEN (UART_INTEN[5]) 和 HWBUFEIF (UART_INTSTS[21]) 都为 1, 该位置 1. 0 = PDMA 模式下, 无缓存错误中断发生. 1 = PDMA 模式下, 有缓存错误中断发生.
[28]	<b>HWTOINT</b>  PDMA 模式 RX 超时中断指示 (只读) RXTOIEN (UART_INTEN[4]) 和 HWTOIF (UART_INTSTS[20]) 都为 1, 该位被置 1. 0 = PDMA 模式时, 无 RX 超时中断产生 1 = PDMA 模式时, 有 RX 超时中断产生
[27]	<b>HWMODINT</b>  PDMA 模式 MODEM 状态中断指示 (只读) MODEMIEN (UART_INTEN[3]) 和 HWMODIF (UART_INTSTS[19]) 都为 1, 该位置 1. 0 = PDMA 模式下, 无 MODEM 状态中断发生. 1 = PDMA 模式下, 有 MODEM 状态中断发生.
[26]	<b>HWRLSINT</b>  PDMA 模式 RLS 中断指示 (只读) RLSIEN (UART_INTEN[2]) 和 HWRLSIF (UART_INTSTS[18]) 都为 1, 该位置 1. 0 = PDMA 模式下, 无 RLS 中断发生.

		1 = PDMA 模式下, 有 RLS 中断发生.
[25:23]	Reserved	保留
[22]	TXENDIF	<p><b>发送空中断标志</b>            该位在 TX FIFO (UART_DAT) 空且最后一字节的停止位已发送出去 (TXEMPTYF (UART_FIFOSTS[28]) 置位) 时置位。如果使能了 TXENDIEN (UART_INTEN[22]), 发送空中断触发            0 = 未触发发送空中断标志.            1 = 已触发发送空中断标志            注:该位在 TX FIFO 为非空或最后一字节的发送还未完成时由硬件自动清空</p>
[21]	HWBUFEIF	<p><b>PDMA 模式缓存错误中断标志 (只读)</b>            当 TX 或 RX FIFO 溢出 (即 (TXOVIF (UART_FIFOSTS [24]) 或 RXOVIF (UART_FIFOSTS[0])) 被置 1), 该位被置 1.当 BERRIF (UART_INTSTS[5]) 被置位, 传送有可能出错. 如果 BFERRIEN (UART_INTEN [5]) 为使能, 缓存错误中断将产生            0= PDMA 模式下,无缓存错误中断标志产生            1= PDMA 模式下,有缓存错误中断标志产生            注:当 TXOVIF (UART_FIFOSTS[24])) 和 RXOVIF (UART_FIFOSTS[0]) 都被清零时,该位也会被清零.</p>
[20]	HWTOIF	<p><b>PDMA 模式 RX 超时中断标志 (只读)</b>            当 RX FIFO 不为空而且持续没有变化,当超时计数器达到 TOIC (UART_TOUT[7:0]) 时,该位被置 1.如果 RXTOIEN (UART_INTEN [4]) 为使能,超时溢出中断将产生            0= PDMA 模式下,无超时溢出中断标志产生            1= PDMA 模式下,有超时溢出中断标志产生            注:该位只读,用户可以通过读 UART_DAT (RX 为活动状态) 将该位清零.</p>
[19]	HWMODIF	<p><b>PDMA 模式 MODEM 中断标志 (只读)</b>            当 nCTS 管脚状态发生变化, (CTSDETF (UART_CTSDETF[0] =1)) 时,该位被置 1.如果 MODEMIEN (UART_INTEN [3]) 位使能, Modem 中断将产生            0= PDMA 模式下,无 Modem 中断标志产生            1= PDMA 模式下,有 Modem 中断标志产生            注: 该位只读,当写 1 到 CTSDETF (UART_CTSDETF [0]) 时, UART_CTSDETF (US_MSRI[0]) 将被清零,该位也将被清零.</p>
[18]	HWRLSIF	<p><b>PDMA 模式 RLS 中断标志 (只读)</b>            当 RX 接收数据出现校验错误,帧错误或 Break 错误,即至少 BIF (UART_FIFOSTS[6]), FEF (UART_FIFOSTS[5]) 和 PEF (UART_FIFOSTS[4]) 中的一位为 1,该位被置 1. 如果 RLSIEN (UART_INTEN [2])) 为使能, RLS 中断将产生            0= PDMA 模式时,无 RLS 中断标志产生            1= PDMA 模式时,有 RLS 中断标志产生            注 1: 在 RS-485 模式,该域包括“地址检测字节 (bit9 = '1')”            注 2: 在 UART 模式,该位只读,当 BIF (UART_FIFOSTS[6]), FEF (UART_FIFOSTS[5]) 和 PEF (UART_FIFOSTS[4]) 都被清零时,该位也将被清零.            注 3: 在 RS-485 模式,该位只读,当 BIF (UART_FIFOSTS[6]), FEF (UART_FIFOSTS[5]), PEF (UART_FIFOSTS[4]) 和 ADDRDETF (UART_FIFOSTS[3]) 都被清零时,该位也将被清零.</p>
[17:16]	Reserved	保留
[15]	LININT	<p><b>LIN 总线中断指示 (只读)</b>            LINIEN (UART_INTEN[8]) 和 LINIF (UART_INTSTS[7]) 都为 1, 该位置 1.            0 = 无 LIN 总线中断发生.</p>

		1 = 有 MODEM 状态中断发生.
[14]	<b>WKINT</b>	<b>UART 唤醒中断指示 (只读)</b> WKIEN (UART_INTEN[6]) 和 WKIF (UART_INTSTS[6]) 都为 1, 该位置 1. 0 = 无 UART 唤醒中断发生. 1 = 有 UART 唤醒中断发生.
[13]	<b>BUFERRINT</b>	<b>缓存错误中断指示 (只读)</b> BUFERRIEN (UART_INTEN[5]) 和 BUFERRIF (UART_INTSTS[5]) 都为 1, 该位置 1. 0 = 无缓存错误中断发生. 1 = 有缓存错误中断发生.
[12]	<b>RXTOINT</b>	<b>RX 超时中断指示 (只读)</b> RXTOIEN (UART_INTEN[4]) 和 RXTOIF (UART_INTSTS[4]) 都为 1, 该位置 1. 0 = 无 RX 超时中断发生. 1 = 有 RX 超时中断发生.
[11]	<b>MODEMINT</b>	<b>MODEM 状态中断 (只读)</b> MODEMIEN (UART_INTEN[3]) 和 MODEMIF (UART_INTSTS[3]) 都为 1, 该位置 1. 0 = 无 MODEM 状态中断发生. 1 = 有 MODEM 状态中断发生.
[10]	<b>RLSINT</b>	<b>RLS 中断指示 (只读)</b> RLSIEN (UART_INTEN[2]) 和 RLSIF (UART_INTSTS[2]) 都为 1, 该位置 1. 0 = 无 RLS 中断发生. 1 = 有 RLS 中断发生.
[9]	<b>THREINT</b>	<b>发送保持寄存器空中断指示 (只读)</b> THREIEN (UART_INTEN[1]) 和 THREIF (UART_INTSTS[1]) 都为 1, 该位置 1. 0 = 无发送保持寄存器空中断发生. 1 = 有发送保持寄存器空中断发生.
[8]	<b>RDAINT</b>	<b>RDA 中断指示 (只读)</b> RDAIEN (UART_INTEN[0]) 和 RDIF (UART_INTSTS[0]) 都为 1, 该位置 1. 0 = 无 RDA 中断发生. 1 = 有 RDA 中断发生.
[7]	<b>LINIF</b>	<b>LIN 总线中断标志 (只读)</b> 当 LIN 从机报头侦测 (SLVHDETF (UART_LINSTS[0] =1)), LIN break 侦测 (BRKDETF (UART_LINSTS[8]=1)), 位错误侦测 (BITEF (UART_LINSTS[9]=1), LIN 从机 ID 校验错误 (SLVIDPEF (UART_LINSTS[2] = 1) 或 LIN 从机报头错误侦测 (SLVHEF (UART_LINSTS[1]) = 1), 该位置 1。如果 LIN_IEN (UA_IER [8]) 被使能, LIN 中断产生。 0 = SLVHDETF, BRKDETF, BITEF, SLVIDPEF 和 SLVHEF 中没有任何一个标志产生 1 = SLVHDETF, BRKDETF, BITEF, SLVIDPEF 和 SLVHEF 中至少有一个标志产生 注:该位只读。当 SLVHDETF (UART_LINSTS[0]), BRKDETF (UART_LINSTS[8]), BITEF (UART_LINSTS[9]), SLVIDPEF (UART_LINSTS[2]) 和 SLVHEF (UART_LINSTS[1]) 都被清 0 时, 该位清 0。软件写 1 也可以清 0 该位。
[6]	<b>WKIF</b>	<b>UART 唤醒中断标志 (只读)</b> TOUTWKF (UART_WKSTS[4]), RS485WKF (UART_WKSTS[3]), RFRTWKF (UART_WKSTS[2]), DATWKF (UART_WKSTS[1]) 或 CTSWKF (UART_WKSTS[0] ) 为 1, 该位置 1.

		0 = 无 UART 唤醒中断标志。 1 = 有 UART 唤醒中断标志。 注: 写 1 清 0 TOUTWKF, RS485WKF, RFRTWKF, DATWKF 和 CTSWKF 后, 该位清 0.
[5]	<b>BUFERRIF</b>	<b>缓存错误中断标志 (只读)</b> TX FIFO 或 RX FIFO 溢出 (TXOVIF (UART_FIFOSTS[24]) 或 RXOVIF (UART_FIFOSTS[0])) 置位, 该位置位。传输错误时, BUFERRIIF (UART_INTSTS[5]) 置位. 如果使能了 BUFERRIEN (UART_INTEN [5]), 会产生缓存错误中断 0 = 无缓存错误中断产生. 1 = 有缓存错误中断产生. 注: 写 1 到 RXOVIF (UART_FIFOSTS[0]) 和 TXOVIF (UART_FIFOSTS[24]) 清空 RXOVIF (UART_FIFOSTS[0]) 和 TXOVIF (UART_FIFOSTS[24]) 后, 该位清 0
[4]	<b>RXTOIF</b>	<b>RX 超时中断标志 (只读)</b> 如果 RX FIFO 为非空且无其他事件时, 当超时计数器计数到 TOIC (UART_TOUT[7:0]) 时, 该位置位。此时如果使能了 RXTOIEN (UART_INTEN [4]), 会产生 RX 超时中断。 0 = 无 RX 超时中断标志产生. 1 = 有 RX 超时中断标志产生.. 注: 该位只读, 读 UART_DAT (RX 有效时) 清 0 该位.
[3]	<b>MODEMIF</b>	<b>MODEM 中断标志 (只读)</b> 如果 nCTS 管脚电平发生改变 (CTSDETF (UART_MODEMSTS[0]) = 1), 该位置位。此时如果使能了 MODEMIEN (UART_INTEN [3]), 会产生 Modem 中断 0 = 无 Modem 中断标注产生. 1 = 有 Modem 中断标注产生. 注: 该位只读, 写 1 到 CTSDETF (UART_MODEMSTS[0]) 清 0 CTSDETF 后, 该位清 0
[2]	<b>RLSIF</b>	<b>RLS 中断标志 (只读)</b> 当 RX 接收数据出现校验错误, 帧错误或 Break 错误, 即至少 BIF (UART_FIFOSTS[6]), FEF (UART_FIFOSTS[5]) 和 PEF (UART_FIFOSTS[4]) 中的一位为 1, 该位被置 1. 如果 RLSIEN (UART_INTEN [2])) 为使能, RLS 中断将产生 0= 无 RLS 中断标志产生 1= 有 RLS 中断标志产生 注 1: 在 RS-485 模式, 当“接收检测任一接收到的地址字节字符 (bit9 = '1') 位”时, 该位也会置 1, 同时, ADDRDETF (UART_FIFOSTS[3]) 位也被置 1 注 2: 该位只读, 当 BIF (UART_FIFOSTS[6]), FEF (UART_FIFOSTS[5]) 和 PEF (UART_FIFOSTS[4]) 都被清零时, 该位也将被清零. 注 3: 在 RS-485 模式, 该位只读, 当 BIF (UART_FIFOSTS[6]), FEF (UART_FIFOSTS[5]), PEF (UART_FIFOSTS[4]) 和 ADDRDETF (UART_FIFOSTS[3]) 都被清零时, 该位也将被清零.
[1]	<b>THREIF</b>	<b>发送保持寄存器空中断标志 (只读)</b> 当 TX FIFO 中最后数据传输到发送移位寄存器时, 该位置位。如果 THREIEN (UART_INTEN[1]) 使能, THRE 中断将产生。 0 = 没有 THRE 中断标志产生 1 = 有 THRE 中断标志产生. 注: 该位只读, 当写数据到 UART_DAT (TX FIFO 非空) 时, 该位将被清除
[0]	<b>RDAIF</b>	<b>接收数据可用中断标志 (只读)</b> 当 RX FIFO 中的数据数量等于 RFITL 时, RDAIF (UART_INTSTS[0]) 将被置位。如果 RDAIEN (UART_INTEN [0]) 使能, RDA 中断将产生。 0 = 没有 RDA 中断标志产生.

		1 =有 RDA 中断标志产生.
--	--	------------------

**注:** 该位只读, 当 RX FIFO 的未读数据低于阈值 (RFITL (UART\_FIFO[7:4])) 时, 该位将被清除

**UART\_TOUT**    **UART 超时寄存器**

寄存器	偏移量	R/W	描述	复位值
<b>UART_TOUT</b> <b>x=0,1,2,3,4,5</b>	UARTx_BA+0x20	R/W	UART 超时寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
DLY							
7	6	5	4	3	2	1	0
TOIC							

位	描述	
[31:16]	<b>Reserved</b>	保留
[15:8]	<b>DLY</b>	<b>TX 延迟时间值</b> 该域用于设置上一次停止位和下一次开始位之间的传输延迟时间
[7:0]	<b>TOIC</b>	<b>超时中断比较器</b> 使能 TOCNTEN (UART_INTEN[11]) 超时计数功能后，每当 RX FIFO 接收到一个新的数据字，定时溢出计数器都会重置并开始计数 (计数时钟 = 波特率)。一旦超时计数器的内容等于超时中断比较器 (TOIC (UART_TOUT[7:0])), 如果此时 RXTOIEN (UART_INTEN [4]) 使能，则接收超时中断 (RXTOINT (UART_INTSTS[12])) 产生。接收到新的数据或 RX FIFO 为空将把RXTOIF (UART_INTSTS[4])清零。为了避免接收超时中断在接收到一个字符就立即产生，TOIC 的值必须设置在 40 到 255 之间。例如，如果 TOIC 为 40，当 UART 传输设置为 1 位停止位且无奇偶校验位时，在 4 个字符时间长度后还没收到数据，超时中断将产生。

**UART BAUD** UART 波特率分频寄存器

寄存器	偏移量	R/W	描述	复位值
UART_BAUD x=0,1,2,3,4,5	UARTx_BA+0x24	R/W	UART 波特率分频寄存器	0x0F00_0000

31	30	29	28	27	26	25	24
Reserved		BAUDM1	BAUDM0	EDIVM1			
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
BRD							
7	6	5	4	3	2	1	0
BRD							

位	描述	
[31:30]	Reserved	保留
[29]	BAUDM1	<b>波特率模式选择位 1</b> 该位为波特率模式选择位 1。UART 提供三种波特率计算方式。该位和 BAUDM0 (UART_BAUD[28]) 组合选择波特率计算方式。详细描述在表 7.15 4 中有介绍。 注：IrDA 模式时必须选择模式 0
[28]	BAUDM0	<b>波特率模式选择位 0</b> 该位为波特率模式选择位 0。UART 提供三种波特率计算方式。该位和 BAUDM1 (UART_BAUD[29]) 组合选择波特率计算方式。详细描述在表 7.15 4 中有介绍。
[27:24]	EDIVM1	<b>波特率模式 1 的扩展分频</b> 该域用于波特率计算模式 1，对于波特率计算模式 0 和模式 2 无效。详细描述在表 7.15 4 中有介绍
[23:16]	Reserved	保留
[15:0]	BRD	<b>波特率分频</b> 该域用于波特率分频。详细描述在表 7.15 4 中有介绍

**UART\_IRDA    UART IrDA 控制寄存器**

寄存器	偏移量	R/W	描述	复位值
UART_IRDA x=0,1,2,3,4,5	UARTx_BA+0x28	R/W	UART IrDA 控制寄存器	0x0000_0040

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved	RXINV	TXINV	Reserved			TXEN	Reserved

位	描述	
[31:7]	Reserved	保留
[6]	RXINV	<p><b>IrDA 接收信号反转</b>            0 = 不反转接收信号。            1 = 反转接收信号 (默认)</p> <p><b>注 1:</b> 设置该位前，需要置位 TXRXDIS (UART_FUNCSEL[3]) 再等 TXRXACT (UART_FIFOSTS[31]) 清除。配置完成后，清除 TXRXDIS (UART_FUNCSEL[3]) 以激活 UART 控制器</p> <p><b>注 2:</b> 该位在 FUNCSEL (UART_FUNCSEL[1:0]) 选择为 IrDATA 功能时有效</p>
[5]	TXINV	<p><b>IrDA 发送信号反转</b>            0 = 不反转发送信号。            1 = 反转发送信号 (默认)</p> <p><b>注 1:</b> 设置该位前，需要置位 TXRXDIS (UART_FUNCSEL[3]) 再等 TXRXACT (UART_FIFOSTS[31]) 清除。配置完成后，清除 TXRXDIS (UART_FUNCSEL[3]) 以激活 UART 控制器</p> <p><b>注 2:</b> 该位在 FUNCSEL (UART_FUNCSEL[1:0]) 选择为 IrDATA 功能时有效</p>
[4:2]	Reserved	保留
[1]	TXEN	<p><b>IrDA 接收/发送选择使能位</b>            0 = IrDA 禁止发送和使能接收(默认)            1 = IrDA 使能发送和禁止接收。</p>
[0]	Reserved	保留

**UART\_ALTCTL UART 选择控制/状态 寄存器**

寄存器	偏移量	R/W	描述	复位值
UART_ALTCTL x=0,1,2,3,4,5	UARTx_BA+0x2C	R/W	UART 选择控制/状态寄存器	0x0000_000C

31	30	29	28	27	26	25	24
ADDRMV							
23	22	21	20	19	18	17	16
Reserved			ABRDBITS		ABRDEN	ABRIF	Reserved
15	14	13	12	11	10	9	8
ADDRDEN	Reserved				RS485AUD	RS485AAD	RS485NMM
7	6	5	4	3	2	1	0
LINTXEN	LINRXEN	Reserved		BRKFL			

位	描述
[31:24]	<b>ADDRMV</b>  地址匹配值 该位包括 RS-485 地址匹配值。 <b>注：</b> 该域用于 RS-485 自动地址检测模式
[23:21]	<b>Reserved</b> 保留
[20:19]	<b>ABRDBITS</b>  自动波特率检测位长 00 = 1-位时长，从起始位到第一个 上升沿。 输入数据格式应该为 0x01。 01 = 2-位时长，从起始位到第一个 上升沿。 输入数据格式应该为 0x02。 10 = 4-位时长，从起始位到第一个 上升沿。 输入数据格式应该为 0x08。 11 = 8-位时长，从起始位到第一个 上升沿。 输入数据格式应该为 0x80。 <b>注意：</b> 计算的位数包括起始位。
[18]	<b>ABRDEN</b>  自动波特率检测使能位 0 = 禁止自动波特率检测功能。 1 = 使能自动波特率检测功能  自动检测结束后，该位将被自动清零。
[17]	<b>ABRIF</b>  自动波特率中断标志 (只读) 当自动波特率检测结束，或者自动波特率计数器发生溢出，如果 ABRIEN (UART_INTEN [18]) 为 1，那么自动波特率中断将会产生。 <b>注：</b> 该位为只读位，但可以通过写“1”到 ABRDTOIF (UART_FIFOSTS[2]) 和 ABRDIF (UART_FIFOSTS[1]) 将该位清零。
[16]	<b>Reserved</b> 保留
[15]	<b>ADDRDEN</b>  RS-485 地址检测使能位 该位用于使能 RS-485 地址检测模式。 0 = 禁止地址检测模式。 1 = 使能地址检测模式。

		注：该位适用于各种 RS-485 操作模式
[14:11]	<b>Reserved</b>	保留
[10]	<b>RS485AUD</b>	<p><b>RS-485 自动方向模式 (AUD)</b></p> <p>0 =RS-485 自动方向操作模式 (AUD) 禁止 1 =RS-485 自动方向操作模式 (AUD) 使能</p> <p>注：仅在 RS-485_AAD 或 RS-485_NMM 操作模式有效</p>
[9]	<b>RS485AAD</b>	<p><b>RS-485 自动地址检测操作模式 (AAD)</b></p> <p>0 =禁止RS-485 自动地址检测模式 (AAD) 1 =使能RS-485 自动地址检测模式 (AAD)</p> <p>注：在 RS-485_NMM 操作模式下无效</p>
[8]	<b>RS485NMM</b>	<p><b>RS-485 普通多点操作模式 (NMM)</b></p> <p>0 =禁止RS-485 普通多点操作模式 (NMM) 1 =使能RS-485 普通多点操作模式 (NMM)</p> <p>注：在 RS-485_AAD 操作模式下无效.</p>
[7]	<b>LINTXEN</b>	<p><b>LIN TX Break 模式使能</b></p> <p>0 = 禁止LIN TX Break 模式 1 = 使能LIN TX Break 模式</p> <p>注：当 TX 间隔域传输操作完成后，该位将被自动清除.</p>
[6]	<b>LINRXEN</b>	<p><b>LIN RX 使能</b></p> <p>0 = 禁止LIN RX 模式 1 = 使能LIN RX 模式</p>
[5:4]	<b>Reserved</b>	保留
[3:0]	<b>BRKFL</b>	<p><b>UART LIN 间隔域长度</b></p> <p>该域表示一个 4-位 LIN TX 间隔域数量</p> <p><b>注 1：</b>该间隔域长度为 BRKFL + 1</p> <p><b>注 2：</b>根据 LIN 规范，复位值是 0xC (间隔域长度= 13).</p>

## UART FUNCSEL UART 功能选择寄存器

寄存器	偏移量	R/W	描述	复位值
UART_FUNCSEL x=0,1,2,3,4,5	UARTx_BA+0x30	R/W	UART 功能选择寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved				TXRXDIS	Reserved	FUNCSEL	

位	描述	
[31:4]	Reserved	保留
[3]	TXRXDIS	<p><b>TX 和 RX 禁止位</b>  置位该位以禁止 TX 和 RX.  0 = TX 和 RX 使能.  1 = TX 和 RX 禁止.</p> <p><b>注:</b> 该位置位后, TX 和 RX 不会立刻关闭。TX 和 RX 在完成当前任务后再关闭。关闭 TX 和 RX 后, TXRXACT (UART_FIFOSTS[31]) 清 0</p>
[2]	Reserved	保留
[1:0]	FUNCSEL	<p><b>功能选择</b>  00 = UART 功能.  01 = LIN 功能.  10 = IrDA 功能.  11 = RS-485 功能.</p>

## UART\_LINCTL UART LIN 控制寄存器

寄存器	偏移量	R/W	描述	复位值
UART_LINCTL x=0,1	UARTx_BA+0x34	R/W	UART LIN 控制寄存器	0x000C_0000

31	30	29	28	27	26	25	24
PID							
23	22	21	20	19	18	17	16
HSEL		BSL		BRKFL			
15	14	13	12	11	10	9	8
Reserved			BITERREN	LINRXOFF	BRKDETEN	IDPEN	SENDH
7	6	5	4	3	2	1	0
Reserved			MUTE	SLVDUEN	SLVAREN	SLVHDEN	SLVEN

位	描述
[31:24]	<p><b>PID</b></p> <p><b>LIN PID 位</b> 在 LIN 功能模式下，该域包含 LIN 帧 ID 值。帧 ID 的校验可以由软件或硬件产生，这取决于 IDPEN (UART_LINCTL[9]) 的设置。 如果校验由硬件产生，用户填 ID0~ID5, (LIN_PID[24:29]，硬件会计算 P0 (PID[30]) 和 P1 (PID[31])，否则用户必须填帧 ID 和校验位 <b>注 1:</b> 用户可以填任何 8-bit 值到该域，位 24 表示 ID0 (LSB 优先) <b>注 2:</b> 该域可以用在 LIN 主机模式或从机模式</p>
[23:22]	<p><b>HSEL</b></p> <p><b>LIN 帧报头选择</b> 00 = LIN 帧报头包括“间隔域” 01 = LIN 帧报头包括“间隔域”和“同步域” 10 = LIN 帧报头 包括“间隔域”，“同步域”域“帧 ID 域”。 11 = 保留 <b>注:</b> 该位可以用于 LIN 主机发送帧头域 (SENDH (UART_LINCTL[8])) = 1) 或用于提示 LIN 从机从 mute 模式退出条件 (LIN_MUTE_EN (UA_LIN_CTL[4]) = 1)。</p>
[21:20]	<p><b>BSL</b></p> <p><b>LIN 间隔域/同步域分隔符长度</b> 00 = LIN 间隔域/同步域分隔符长度为 1 bit 时间. 10 = LIN 间隔域/同步域分隔符长度为 2 bit 时间. 10 = LIN 间隔域/同步域分隔符长度为 3 bit 时间. 11 = LIN 间隔域/同步域分隔符长度为 4 bit 时间. <b>注:</b> 该位用于 LIN 主机发送帧报头域</p>
[19:16]	<p><b>BRKFL</b></p> <p><b>LIN 间隔域长度</b> 该域表示一个 4-bit LIN TX 间隔域数量 <b>注 1:</b> 这个寄存器是 BRKFL (UART_ALTCTL[3:0]) 的映射寄存器，用户可以通过设置 BRKFL (UART_ALTCTL[3:0]) 或 BRKFL (UART_LINCTL[19:16]) 对间隔域的长度进行读/写。</p>

		<b>注 2:</b> 该间隔域长度为 BRKFL + 1 <b>注 3:</b> 根据 LIN 规范, 复位值是 12 (间隔域长度= 13).
[15:13]	<b>Reserved</b>	保留
[12]	<b>BITERREN</b>	<b>位错误侦测使能</b> 0 = 位错误侦测功能禁止. 1 = 位错误侦测使能. <b>注:</b> 在 LIN 功能模式下, 当位错误发生时, BITEF (UART_LINSTS[9]) 标志会被置位。如果 LINIEN (UART_INTEN[8]) = 1, 会产生一个中断
[11]	<b>LINRXOFF</b>	<b>LIN 接收禁止位</b> 如果接收使能 (RXOFF (UART_LINCTL[11]) = 0), 所有接收到的数据字节会被接收并存储在 RX-FIFO, 如果接收禁止 (RXOFF (UART_LINCTL[11] = 1), 接收到的所有数据会被忽略 0 = LIN 接收使能. 1 = LIN 接收禁止. <b>注:</b> 该位仅工作在 LIN 功能模式下有效 (FUNCSEL (UART_FUNCSEL[1:0]) = 01)
[10]	<b>BRKDETEN</b>	<b>LIN 间隔域检测使能位</b> 当检测到显性位长度连续超过 11 位, 后跟一个分隔符, 在间隔域结束时 BRKDETF (UART_LINSTS[8]) 标志被置位。如果 LINIEN (UART_INTEN [8])=1, 将会产生中断。 0 = LIN 间隔域检测禁止 1 = LIN 间隔域检测使能.
[9]	<b>IDPEN</b>	<b>LIN ID 校验使能位</b> 0 = LIN 帧 ID 校验禁止. 1 = LIN 帧 ID 校验使能. <b>注 1:</b> 该位可以用于 LIN 主机发送帧头域 (SENDH (UART_LINCTL[8])) = 1 且 HSEL (UART_LINCTL[23:22]) = 10) 或用来使能 LIN 从机对接收到的帧 ID 进行奇偶校验检查。 <b>注 2:</b> 该位仅在发送器报头是在 HSEL (UART_LINCTL[23:22]) = 10 时有用。
[8]	<b>SENDH</b>	<b>LIN TX 发送帧报头使能位</b> LIN TX 帧报头可以是“间隔域”或“间隔和同步域”或“间隔, 同步和帧 ID 域”, 这取决于 HSEL (UART_LINCTL[23:22]) 的设置 0 = LIN TX 帧报头发送禁止. 1 = LIN TX 帧报头发送使能. <b>注 1:</b> 这些寄存器是 SENDH (UART_ALTCTL [7]) 的影子寄存器; 用户可以通过 SENDH (UART_ALTCTL [7]) 或 SENDH (UART_LINCTL [8]) 进行读写设置 <b>注 2:</b> 当发送器帧头域 (可能是“间隔”或“间隔 + 同步”或“间隔 + 同步 + 帧 ID”)通过 HSEL (UART_LINCTL[23:22]) 选择) 传输操作完成, 该位会自动清 0.
[7:5]	<b>Reserved</b>	保留
[4]	<b>MUTE</b>	<b>LIN Mute 模式使能位</b> 0 = LIN Mute 模式禁止 1 = LIN Mute 模式使能. <b>注意:</b> 退出 Mute 模式条件, 该域的控制和相互影响在章节 6.15.5.9 (LIN 从机模式) 中有详细解释
[3]	<b>SLVDUEN</b>	<b>LIN 从机分频器更新方式使能位</b> 0 = UA_BAUD 被软件更新 (如果同时没有自动重同步发生) 1 = UA_BAUD 在下次接收到字符的时候更新。用户必须在接收校验之前设置该位.

		<p><b>注 1:</b> 该位仅在 LIN 从模式有效 (SLVEN (UART_LINCTL[0]) = 1)</p> <p><b>注 2:</b> 该位用于 LIN 从机自动重同步模式。(对于非自动重同步模式, 该位需维持清零)</p> <p><b>注 3:</b> 该域的控制和相互作用在章节 6.15.5.9 中解释 (带自动重同步的从机模式)</p>
[2]	<b>SLVAREN</b>	<p><b>LIN 从机自动重同步模式使能位</b></p> <p>0 = LIN 从机自动重同步模式禁止</p> <p>1 = LIN 从机自动重同步模式使能.</p> <p><b>注意 1:</b> 该位仅在 LIN 从机模式 (SLVEN (UART_LINCTL[0]) = 1) 有效.</p> <p><b>注意 2:</b> 当工作在自动重同步模式, 波特率的设置必须是模式 2 (BAUD_M1 (UA_BAUD [29]) 和 BAUD_M0 (UA_BAUD [28]) 必须是 1).</p> <p><b>注意 3:</b> 该域的控制和相互作用在章节 6.15.5.9 (带自动重同步的从机模式) 中解释</p>
[1]	<b>SLVHDEN</b>	<p><b>LIN 从机报头侦测使能位</b></p> <p>0 = LIN 从机报头检测禁止.</p> <p>1 = LIN 从机报头检测使能.</p> <p><b>注 1:</b> 该位仅在 LIN 从机模式 (SLVEN (UART_LINCTL[0]) = 1) 有效.</p> <p><b>注 2:</b> 在 LIN 功能模式, 当侦测报头域 (间隔域 + 同步域 + 帧 ID), SLVHDETF (UART_LINSTS [0]) 标志会被置位, 如果 LINIEN (UART_INTEN[8]) = 1, 会产生一个中断</p>
[0]	<b>SLVEN</b>	<p><b>LIN 从模式使能位</b></p> <p>0 = LIN 从模式禁止</p> <p>1 = LIN 从模式使能</p>

## UART\_LINSTS UART LIN 状态寄存器

寄存器	偏移量	R/W	描述	复位值
UART_LINSTS x=0,1	UARTx_BA+0x38	R/W	UART LIN 状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved						BITEF	BRKDETF
7	6	5	4	3	2	1	0
Reserved				SLVSYNCF	SLVIDPEF	SLVHEF	SLVHDETF

位	描述
[31:10]	<b>Reserved</b> 保留
[9]	<b>BITEF</b> 位错误侦测状态标志(只读). TX 传输状态, 硬件会监视总线状态, 如果输入管脚 (UART_RXD) 状态不等于输出管脚 ((UART_TXD) 状态, BITEF (UART_LINSTS[9]) 位会被置位。 当发生位错误, 如果 LINIEN (UART_INTEN[8]) = 1, 会产生一个中断 0 = 未检测到位错误 1 = 检测到位错误 <b>注 1:</b> 向该位写 1 清 0 <b>注 2:</b> 该位仅当位错误侦测功能使能时有效 (BITERREN (UART_LINCTL [12]) = 1)
[8]	<b>BRKDETF</b> LIN 间隔域 检测标志(只读) 当侦测到一个间隔域, 该位由硬件置位。可通过软件写 1 将该位清零。 0 = LIN 间隔域没有被检测到 1 = LIN 间隔域被检测到. <b>注 1:</b> 向该位写 1 清 0. <b>注 2:</b> 该位仅当 LIN 间隔域侦测功能使能时有效 (BRKDETEN (UART_LINCTL[10]) =1)
[7:4]	<b>Reserved</b> 保留
[3]	<b>SLVSYNCF</b> LIN 从机同步域. (只读) 该位表示在自动重同步模式, LIN 同步域正在被分析。当接收器报头侦测到一些错误, 用户必须通过写 1 到该位复位内部电路来重新搜索新的帧报头 0 = 当前字符不在 LIN 同步状态. 1 = 当前字符在 LIN 同步状态. <b>注 1:</b> 该位仅在 LIN 从机模式有效 (SLVEN(UART_LINCTL[0]) = 1) <b>注 2:</b> 向该位写 1 清 0 <b>注 3:</b> 当向该位写 1, 硬件会重载初始波特率并重新搜索新的帧报头

[2]	<b>SLVIDPEF</b>	<p><b>LIN 从机 ID 校验错误标志 (只读)</b></p> <p>当接收帧 ID 校验错误，该位由硬件置 1 0 = 无效. 1 = 收到的帧 ID 校验错误.</p> <p><b>注 1:</b>向该位写 1 清 0.</p> <p><b>注 2:</b>该位仅在 LIN 从机模式 (SLVEN (UART_LINCTL [0])= 1)，并使能 LIN 帧 ID 校验功能 IDPEN (UART_LINCTL [9]) 时有效</p>
[1]	<b>SLVHEF</b>	<p><b>LIN 从机报头错误标志 (只读)</b></p> <p>在 LIN 从机模式，当侦测到一个 LIN 报头错误时，该位由硬件置 1，向该位写 1 清 0。报头错误包括“break 间隔符太短 (小于 0.5 位的时间)”，“在同步域或在识别域中帧错误，”“在非自动重同步模式同步域数据不是 0x55”，“自动重同步模式同步域偏离错误”，“自动重同步模式同步域测量超时”和“LIN 报头接收超时”</p> <p>0 = LIN 未检测到报头错误 1 = LIN 检测到报头错误.</p> <p><b>注 1:</b>向该位写 1 清 0</p> <p><b>注 2:</b> 该位仅在 LIN 从机模式 (SLVEN (UART_LINCTL [0]) = 1)，并且使能 LIN 从机报头侦测功能 (SLVHDEN (UART_LINCTL [1])) 时有效。</p>
[0]	<b>SLVHDETF</b>	<p><b>LIN 从机报头侦测标志 (只读)</b></p> <p>在 LIN 从机模式，当侦测到一个 LIN 报头，该位由硬件置位，通过软件写 1 清 0 0 = LIN 报头没有被侦测到. 1 = LIN 报头被侦测到 (分隔+ 同步+ 帧 ID)</p> <p><b>注 1:</b>向该位写 1 清 0</p> <p><b>注 2:</b> 该位仅在 LIN 从机模式 (SLVEN (UART_LINCTL [0]) = 1)，并且使能 LIN 从机报头侦测功能 (SLVHDEN (UART_LINCTL [1])) 时有效。</p> <p><b>注 3:</b> 当使能 ID 校验 IDPEN (UART_LINCTL [9])，如果硬件侦测到完整的报头 (“分隔+ 同步+ 帧 ID”)，SLVHDETF 会被置位，无论帧 ID 是否正确</p>

## UART\_BRCOMP UART 波特率补偿寄存器

寄存器	偏移量	R/W	描述	复位值
UART_BRCOMP $x=0,1,2,3,4,5$	UARTx_BA+0x3C	R/W	UART 波特率补偿寄存器	0x0000_0000

31	30	29	28	27	26	25	24
BRCOMPDEC	Reserved						
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							BRCOMP
7	6	5	4	3	2	1	0
BRCOMP							

位	描述	
[31]	<b>BRCOMPDEC</b>	波特率补偿方向 0 = 对应位正向 (加 1 个模块时钟) 补偿 1 = 对应位负向 (减 1 个模块时钟) 补偿.
[30:9]	<b>Reserved</b>	保留
[8:0]	<b>BRCOMP</b>	波特率补偿模式 这 9 位用以定义相关的位是否被补偿 BRCOMP[7:0] 用来定义 UART_DAT[7:0] 的补偿, BRCOM[8] 则是校验位

## UART\_WKCTL UART 唤醒控制寄存器

寄存器	偏移量	R/W	描述	复位值
UART_WKCTL x=0,1,2,3,4,5	UARTx_BA+0x40	R/W	UART 唤醒控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved			WKtoutEN	WKRS485EN	WKRFRTEN	WKDATEN	WKCTSEN

位	描述
[31:5]	<b>Reserved</b> 保留
[4]	<b>WKtoutEN</b> 数据接收 FIFO 达到阈值超时唤醒系统功能使能位 0 = 数据接收 FIFO 达到阈值超时唤醒系统功能禁止 1 = 数据接收 FIFO 达到阈值超时唤醒系统功能使能, 数据接收 FIFO 达到阈值超时唤醒系统功能可以将处于掉电模式的系统唤醒 <b>注:</b> WKRFRTEN (UART_WKCTL[2]) 置 1 时, 建议使能该位.
[3]	<b>WKRS485EN</b> RS-485 地址匹配 (AAD 模式) 唤醒使能位 0 = RS-485 地址匹配 (AAD 模式) 唤醒系统功能禁止. 1 = RS-485 地址匹配 (AAD 模式) 唤醒系统功能使能, RS-485 地址匹配 (AAD 模式) 唤醒系统功能可以将处于掉电模式的系统唤醒 <b>注:</b> 该位在 RS-485 自动地址检测 (AAD 模式), ADDRDEN (UART_ALTCTL[15]) 置 1 时使用
[2]	<b>WKRFRTEN</b> 数据接收 FIFO 达到阈值唤醒使能位 0 = 数据接收 FIFO 达到阈值唤醒系统功能禁止. 1 = 数据接收 FIFO 达到阈值唤醒系统功能使能, 数据接收 FIFO 达到阈值唤醒系统功能可以将处于掉电模式的系统唤醒
[1]	<b>WKDATEN</b> 输入数据唤醒使能位 0 = 输入数据唤醒系统功能禁止. 1 = 输入数据唤醒系统功能使能, 输入数据唤醒系统功能可以将处于掉电模式的系统唤醒
[0]	<b>WKCTSEN</b> nCTS 唤醒使能位 0 = nCTS 唤醒系统功能禁止 1 = nCTS 唤醒系统功能使能, 唤醒系统功能可以将处于掉电模式的系统唤醒

**UART\_WKSTS**    UART 唤醒状态寄存器

寄存器	偏移量	R/W	描述	复位值
<b>UART_WKSTS</b> x=0,1,2,3,4,5	UARTx_BA+0x44	R/W	UART 唤醒状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved			TOUTWKF	RS485WKF	RFRTWKF	DATWKF	CTSWKF

位	描述
[31:5]	<b>Reserved</b> 保留
[4]	<b>TOUTWKF</b> 接收数据 FIFO 阈值超时唤醒标志 掉电的芯片被接收数据 FIFO 阈值超时唤醒时，该位置位 0 = 芯片处于掉电模式。 1 = 掉电的芯片被接收数据 FIFO 阈值超时唤醒。 <b>注 1:</b> WKROUTEN (UART_WKCTL[4]) 使能，接收数据 FIFO 阈值超时唤醒导致该位置 1。 <b>注 2:</b> 该位写 1 清 0。
[3]	<b>RS485WKF</b> RS-485 地址匹配 (AAD 模式) 唤醒标志 掉电的芯片被 RS-485 地址匹配 (AAD 模式) 唤醒时，该位置位 0 = 芯片处于掉电模式。 1 = 掉电的芯片被 RS-485 地址匹配 (AAD 模式) 唤醒。 <b>注 1:</b> WKRS485EN (UART_WKCTL[3]) 使能，RS-485 地址匹配 (AAD 模式) 唤醒导致该位置 1。 <b>注 2:</b> 该位写 1 清 0。
[2]	<b>RFRTWKF</b> 接收数据 FIFO 达到阈值唤醒标志 掉电的芯片被接收数据 FIFO 达到阈值唤醒时，该位置位 0 = 芯片处于掉电模式。 1 = 掉电的芯片被接收数据 FIFO 达到阈值唤醒。 <b>注 1:</b> WKRFRTEN (UART_WKCTL[2]) 使能，接收数据 FIFO 达到阈值唤醒导致该位置 1。 <b>注 2:</b> 该位写 1 清 0。
[1]	<b>DATWKF</b> 输入数据唤醒标志 掉电的芯片被输入数据唤醒时，该位置位。 0 = 芯片处于掉电模式。 1 = 掉电的芯片被输入数据唤醒。

		<p><b>注 1:</b> WKDATEN (UART_WKCTL[1]) 使能, 输入数据唤醒导致该位置 1 <b>注 2:</b> 该位写 1 清 0.</p>
[0]	<b>CTSWKF</b>	<p><b>nCTS 唤醒标志</b> 掉电的芯片被nCTS唤醒时, 该位置 1 . 0 = 芯片处于掉电模式. 1 = 掉电的芯片被 nCTS 唤醒. <b>注 1:</b> WKCTSEN (UART_WKCTL[0]) 使能, nCTS 唤醒导致该位置 1 <b>注 2:</b> 该位写 1 清 0.</p>

**UART\_DWKCOMP UART 输入数据唤醒补偿寄存器**

寄存器	偏移量	R/W	描述	复位值
UART_DWKCOMP x=0,1,2,3,4,5	UARTx_BA+0x48	R/W	UART 输入数据唤醒补偿寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
STCOMP							
7	6	5	4	3	2	1	0
STCOMP							

位	描述	
[31:16]	Reserved	保留
[15:0]	STCOMP	<p><b>开始位补偿值</b>            这些位代表在芯片从掉电模式唤醒到 UART 控制器可以接收第一个开始位需要的 UART_CLK 数目</p> <p><b>注:</b> 仅在 WKDATE (UART_WKCTL[1]) 置位时有效.</p>

## 6.16 EMAC 以太网控制器

### 6.16.1 概述

M480 为网络应用提供了一组以太网 MAC 控制器 (EMAC)。

以太网控制器包括：用于识别以太网MAC地址的内置 CAM 功能的 IEEE 802.3/ 以太网协议引擎、发送缓存、接收缓存、收发状态机控制器、符合 IEEE 1588 的时间戳、魔术包解析引擎和状态控制器

支持 MII 和 RMII (简化的 MII) 接口连接外部 PHY.

### 6.16.2 特性

- 符合 IEEE Std. 802.3 CSMA/CD 协议
- 时间戳符合 IEEE Std. 1588 – 2002 协议
- 支持10 Mbps 或100 Mbps全双工或半双工
- 支持RMII 接口
- MII 管理功能可控制外接以太网物理层芯片
- 流控支持暂停和远程暂停功能
- 支特长帧 (大于 1518 字节) 和短帧 (小于 64 字节) 接收
- 支持 16 个 CAM条目 功能识别 MAC 地址
- 支持魔术包唤醒处于掉电状态的系统
- 收和发各 256 字节缓存
- 支持 DMA 功能

## 6.16.3 框图

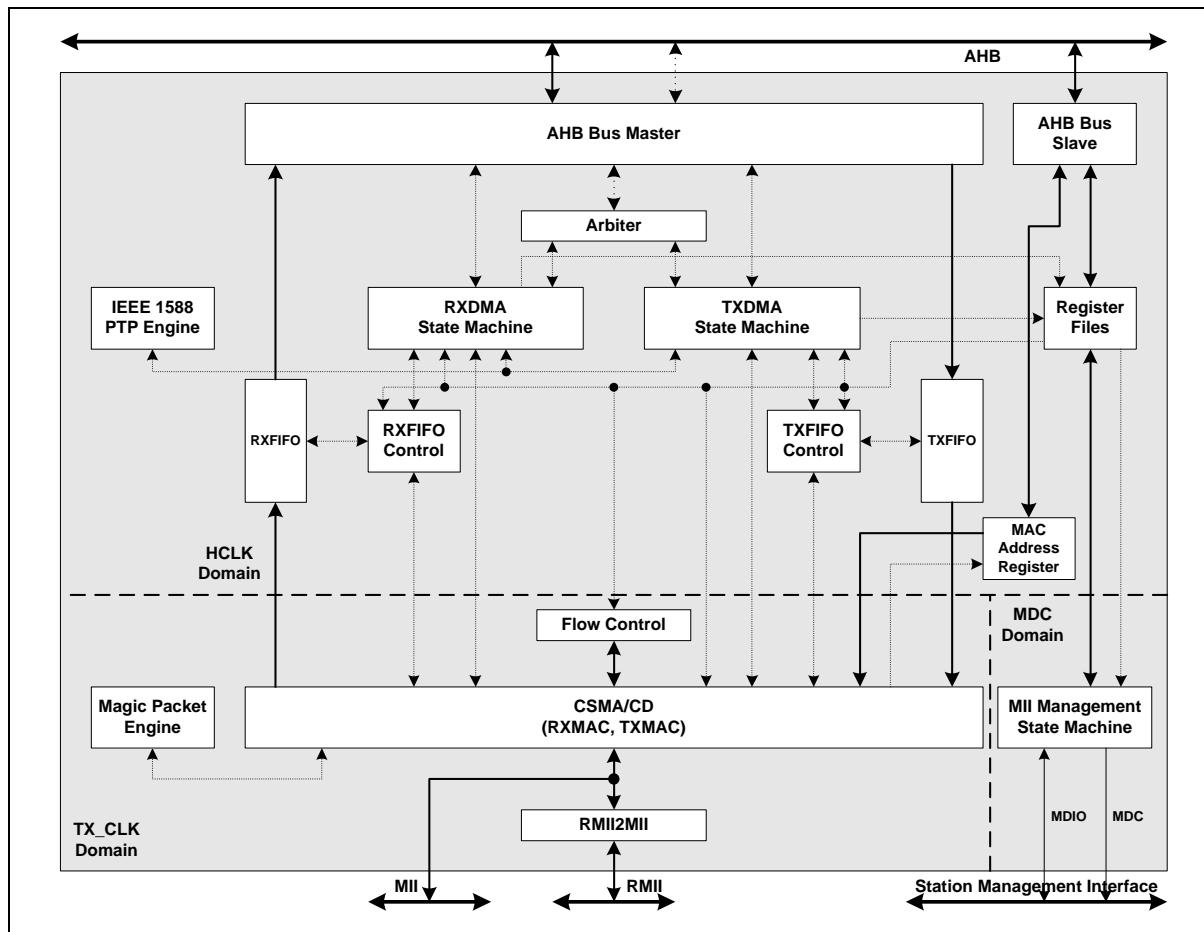


图 6.16-1 EMAC 框图

## 6.16.4 基本配置

## 6.16.4.1 EMAC 基本配置

- 时钟源配置
  - 在寄存器 EMACDIV (CLK\_CLKDIV3[21:16]) 选择 MDC 的时钟分频数
  - 在寄存器 EMACCKEN (CLK\_AHBCLOCK[5]) 使能 EMAC 时钟
- 复位配置
  - 在寄存器 EMACRST (SYS\_IPRST0[5]) 复位 EMAC 控制器
- 管脚配置n

组	管脚名	GPIO	MFP
EMAC	EMAC_RMII_MDC	PB.11, PE.8	MFP3
	EMAC_RMII_MDIO	PB.10, PE.9	MFP3
	EMAC_RMII_RXD0	PC.7 PB.4	MFP3 MFP4

	EMAC_RMII_RXD1	PC.6	MFP3
		PB.3	MFP4
	EMAC_RMII_CRSDV	PA.7	MFP3
		PB.2	MFP4
	EMAC_RMII_RXERR	PA.6	MFP3
		PB.1	MFP4
	EMAC_RMII_TXD0	PB.9, PE.10	MFP3
	EMAC_RMII_TXD1	PB.8, PE.11	MFP3
	EMAC_RMII_TXEN	PB.7, PE.12	MFP3
	EMAC_PPS	PB.6, PE.13	MFP3
	EMAC_RMII_REFCLK	PC.8	MFP3
		PB.5	MFP4

## 6.16.5 功能描述

### 6.16.5.1 仲裁器

在 EMAC 上，有 RXREQ 和 TXREQ 两种总线请求，由仲裁器在这两种请求间进行仲裁决定哪一个请求 AHB 总线，表 7.16 1 是仲裁结果示意：

RXREQ	TXREQ	授权
0	0	2 个都不授权
0	1	授权 TXDMA.
1	0	授权 RXDMA
1	1	若 TXFIFO 有效字节数少于 RXFIFO 空余字节数, 授权 TXDMA.
1	1	若 RXFIFO 空余字节数少于等于 TXFIFO 有效空间, 授权 RXDMA

表 6.16-1 仲裁器仲裁结果

### 6.16.5.2 TXDMA 状态机

TXDMA 状态机从内存读取数据通过 AHB 传输到 256 字节的缓存。首先，TXDMA 会请求发送 MAC 把数据发送出去。传输时，TXDMA 先获取发送描述符。按发送描述符里定义的缓存地址，TXDMA 从系统获取帧数据并存储在内部 256 字节的发送 FIFO 里。之后，发送 MAC 会从发送 FIFO 读取帧数据并发送出去。完成传输后，TXDMA 更新当前帧的发送状态，再把描述符回写到系统内存，以示发送结束。

### 6.16.5.3 RXDMA 状态机

RXDMA 状态机从内部的 256 字节接收 FIFO 读取数据通过 AHB 总线传输到系统内存。接收过程中，RXDMA 先获取接收描述符。RXDMA 从接收描述符里定义的缓存地址读出输入帧要在系统中存储的地址。当 MAC 指示收到数据后，RXDMA 从 256 字节的接收缓存读取数据并存入系统内存。传输结束后，RXDMA 更新当前帧的接收状态，再把描述符回写到系统内存，以示数据已存入指定内存。

### 6.16.5.4 流控

这里指的是 EMAC 工作在全双工模式下的流控功能。流控功能在 IEEE 802.3 第 31 章中有定义。在

IEEE 802.3 Std 里定义的流控帧仅指同一时间下的暂停帧。控制帧的收发可以通过控制寄存器配置。

控制寄存器 EMAC\_CTL 的位 ACP 置 1，就开始接收控制帧。若收到暂停帧，当前发送帧发送结束后，就暂停发送。

发送控制帧的流程是：先把控制帧的 MAC 目的地地址写入寄存器 {EMAC\_CAM13M, EMAC\_CAM13L}, 源 MAC 地址写入寄存器 {EMAC\_CAM14M, EMAC\_CAM14L}，控制帧的长度，操作码和操作写入寄存器 {EMAC\_CAM15MSB, EMAC\_CAM15LSB}，然后置 1 位 SDPZ (EMAC\_CTL[16]) 就会发送。发送结束后，位 SDPZ (EMAC\_CTL[16]) 自动清 0.

#### 6.16.5.5 MII 管理状态机

MII 管理功能符合 IEEE 802.3 标准。通过此接口，软件可以读取外部 PHY 芯片的控制和状态寄存器。MII 管理功能提供 2 个可编程的寄存器 EMAC\_MIIMDAT (MAC MII 管理数据寄存器) 和 EMAC\_MIIMCTL (MAC MII 管理数据控制和地址寄存器)。置 1 位 BUSY (EMAC\_MIIMCTL[17]) 将触发 MII 管理状态机。管理周期结束后，BUSY 位自动清 0。

#### 6.16.5.6 MAC

EMAC 功能完全符合 IEEE802.3 规范。如图 7.16 2 是帧结构和传输操作的示例。

数据帧从发送 DMA 到 MII 总线之前，先经 MAC 包装：加上前导码，分界符 SFD，帧校验序列，数据若不够 64 字节后面填充 0，最后再加上 CRC 码。包装后的帧格式如下所示：

110101010 --- 10101010	10101011	d0	d1	d2	--	dn	Padding	CRC31	CRC30	---	CRC0
------------------------	----------	----	----	----	----	----	---------	-------	-------	-----	------

图 6.16-2 以太网帧格式

如前面所提到的格式，前导码是一组连续的 7 个“10101010”，SFD 是 1 个字节的“10101011”。如果数据帧不足 64 位会在填充数据里补 0。发送描述符里的 P 位用来控制数据帧不足 64 位时是否需要填充 0。如果禁止了填充功能，则不会把填充数据附在数据帧之后。CRC0...CRC32 是 32 位的 CRC 校验序列。CRC 码符合 IEEE802.3 的标准。如果发送描述符里的 CRC 禁止功能置高，则不会在数据位后附 32 位的 CRC 序列。

MAC 还实现了其他 IEEE802.3 定义的功能如，帧间距功能，冲突检测，冲突执行，冲突回退和重发送。冲突回退时间与时间片有关，512 位时间。从当前发送尝试到下一次发送尝试之间的时间片延时符合 IEEE802.3 规定的正态分布随机算法。MAC 按 IEEE802.3 标准实现了接收相关的功能：地址识别，帧完整性检验，帧解析和防冲突检测。

#### 6.16.5.7 IEEE 1588 标准的时间戳引擎

EMAC 时间戳符合 IEEE 标准 1588。时间戳引擎里，64 位计数器产生参考时间，详见寄存器 EMAC\_TSSEC 和 ETSLSR。

发送帧，若 TSEN (EMAC\_TSCTL[0]) 和寄存器 TXDES0 的位 TTSEN 都为 1，帧发送结束后，64 位的时间值将写入 TXDES1 和 TXDES2。

接收帧，若 TSEN (EMAC\_TSCTL[0]) 为 1，接收结束后，64 位的时间值将写入 RXDES1 和 RXDES3。

图 7.16 3 描述了 64 位计数器如何产生参考时间。

2 个 32 位计数器 EMAC\_TSSEC 和 EMAC\_TSSUBSEC 组成一个 64 位计数器，对 HCLK 进行计数产生时间戳。两种计数方式由 TSMODE (EMAC\_TSCTL[3]) 控制，EMAC\_TSSUBSEC 每次增加 EMAC\_TSINC 值。TSMODE (EMAC\_TSCTL[3]) =0 时，TSLSR 每个时钟都增加，TSMODE (EMAC\_TSCTL[3]) =1 时，TSLSR 仅在累加器溢出时增加。

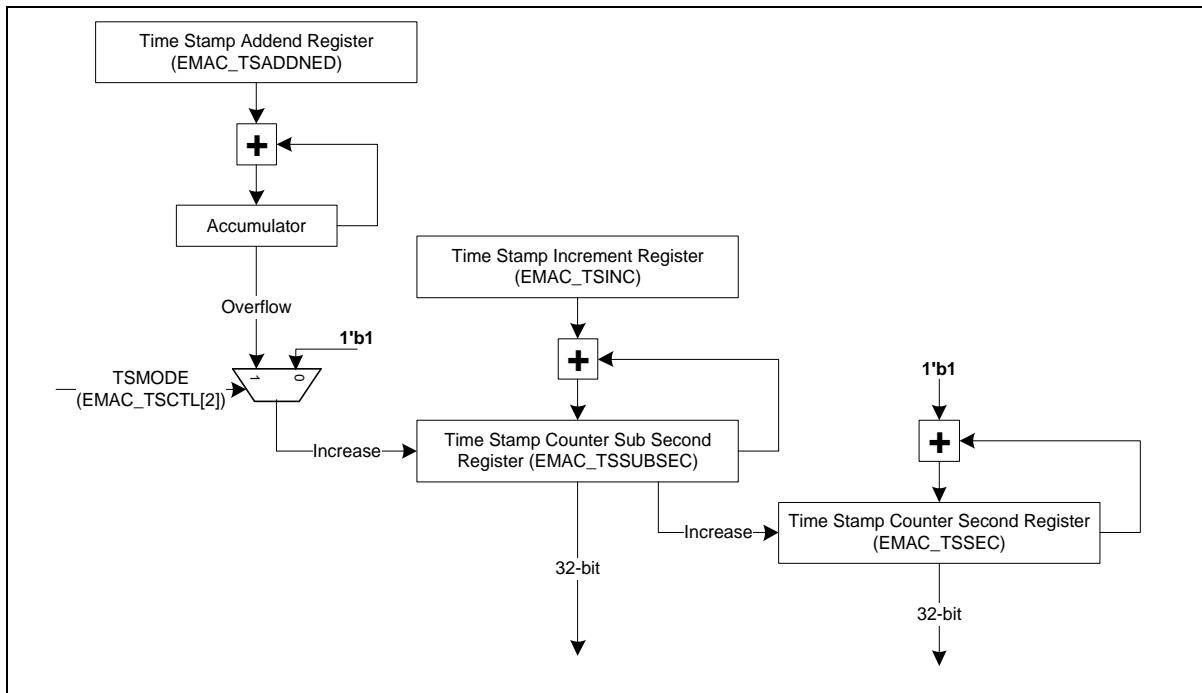


图 6.16-3 64 位参考时间计数器

#### 6.16.5.8 魔术包解析引擎

EMAC 支持魔术包解析。魔术包是一种广播包。它是紧跟在重复 16 次的 48 位 MAC 地址 (EMAC\_CAM0M 和 EMAC\_CAM0L) 之后的 6 个字节的 0xFF。

WOLEN (EMAC\_CTL[6]) 控制着魔术包引擎的开关。WOLEN (EMAC\_CTL[6]) =1 时，收到魔术包，位 WOLIF (EMAC\_INTSTS[15]) 置 1，同时唤醒芯片。若寄存器 MIEN 的位 WOLIEN =1，则会申请中断。

#### 6.16.5.9 DMA 描述符数据结构

描述符是用来保存每帧的控制、状态和数据信息的链表结构。CPU 和 EMAC 通过描述符交换帧传输的信息。

描述符有两种：RXDMA 描述符用与帧接收和 TXDMA 描述符用与帧发送。每个描述符有 4 个字，包含每帧的控制，状态和数据信息，他们的数据结构参见 7.16.5.10 和 7.16.5.11 章节。

#### 6.16.5.10 RXDMA 描述符数据结构

RXDMA 描述符包含 4 个 32 位字。如图 7.16-4 是 RXDMA 描述符的数据结构。

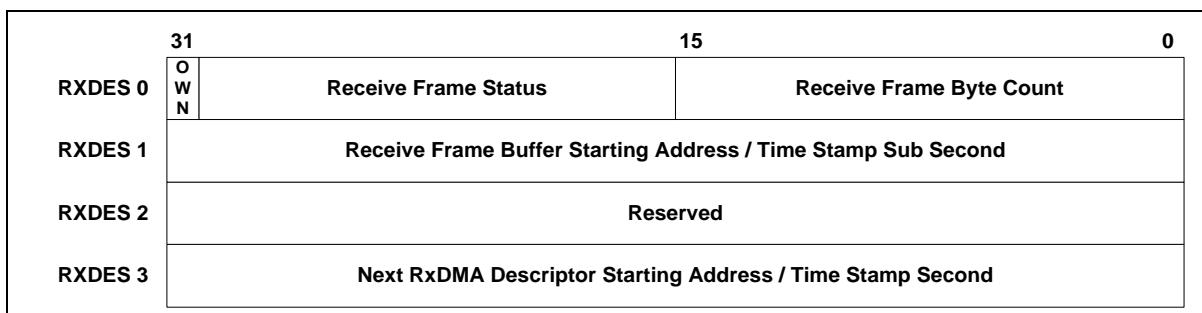


图 6.16-4 RXDMA 描述符数据结构

**RXDES 0: RXDMA 描述符字 0**

RXDMA 描述符字 0 包含一个描述符所有者标识，接收帧状态和接收帧字节数。详细描述如下：

31	30	29	28	27	26	25	24
Owner	Reserved						
23	22	21	20	19	18	17	16
RTSAS	RPIF	ALIEIF	RXGDIF	LPIF	Reserved	CRCEIF	RXIF
15	14	13	12	11	10	9	8
RBC							
7	6	5	4	3	2	1	0
RBC							

位	字段	描述
[31]	Owner	<b>所有者</b> 该寄存器位表示 RX 描述符的归属，是 CPU 还是 EMAC。只有描述符的所有者才能修改 RX 描述符，其他只能读描述符。 O = 1'b1 表示 EMAC RXDMA 是 RX 描述符的所有者，帧接收的 RX 描述符可用。帧接收完成后，EMAC RXDMA 修改该位的归属为 1'b0 O = 1'b0 表示 CPU 是 RX 描述符的所有者。CPU 帧处理完成后，修改该位的归属为 1'b1,，EMAC RXDMA 获得 RX 描述符的所有权 0 = 所有者为 CPU. 1 = 所有者为 EMAC.
[30:24]	Reserved	保留
[23]	RTSAS	<b>RX 时间戳状态</b> 该位表示输入帧成功获得时间戳。当该位置位时，输入帧接收完成后，RX 描述符字 1 和 RX 描述符字 3 保存时间戳的值。 0 = RX 描述符字 1 和 RX 描述符字 3 未保存时间戳 1 = RX 描述符字 1 和 RX 描述符字 3 已保存时间戳
[22]	RPIF	<b>短包</b> RPIF 表示 RX 描述符指向的数据缓存地址里的帧是个短帧 (帧长度小于 64 字节) 0 = 非短帧. 1 = 是短帧.
[21]	ALIEIF	<b>对齐错误</b> ALIEIF 表示 RX 描述符指向的数据缓存地址里的帧长度不是整字节的 0 = 帧长度是整字节 1 = 帧长度非整字节
[20]	RXGDIF	<b>帧接收完成</b> RXGDIF 表示 帧接收已完成且存入了 RX 描述符指向的数据缓存地址里 0 = 帧接收未完成 1 = 帧接收已完成.

[19]	<b>LPIF</b>	<b>长包中断标识</b> LPIF 表示 RX 描述符指向的数据缓存地址里的帧是长帧 (帧长度大于 1518) 0 = 非长帧. 1 = 是长帧.
[18]	<b>Reserved</b>	保留
[17]	<b>CRCEIF</b>	<b>CRC 错误</b> CRCEIF 表示 RX 描述符指向的数据缓存地址里的帧 CRC 错误 0 = CRC 正确. 1 = CRC 错误.
[16]	<b>RXIF</b>	<b>接收中断</b> RXIF 表示 RX 描述符指向的数据缓存地址里的帧 产生了一个中断 0 = 无中断. 1 = 有中断.
[15:0]	<b>RBC</b>	<b>接收字节计数</b> RBC 表示 RX 描述符指向的数据缓存地址里的帧的数目。包含 4 字节的 CRC。若使能了 STRIPCRC (EMAC_CTL[5]), 计算时 CRC 字节将不包含在内。

**RXDES 1: RXDMA 描述符字 1**

RXDMA 1 包含了帧接收缓存的起始地址或者时间戳的低 32 位。详情如下：

31	30	29	28	27	26	25	24
RXBSA/TSSUBSEC							
23	22	21	20	19	18	17	16
RXBSA/TSSUBSEC							
15	14	13	12	11	10	9	8
RXBSA/TSSUBSEC							
7	6	5	4	3	2	1	0
RXBSA/TSSUBSEC							

位	字段	描述
[31:0]	<b>RXBSA</b>	接收缓存的起始地址 RXBSA 接收缓存的起始地址。

表 6.16-2 RXDMA 描述符字 1 (TSEN (EMAC\_TSCTL[0]) 为 0)

位	字段	描述
[31:0]	<b>TSSUBSEC</b>	时间戳亚秒 使能 TSEN (EMAC_TSCTL[0]), 以太网 MAC 控制器写 RX 描述符到系统内存时, 会存储时间戳低 32 位 (ETSLSR) 到该字段

表 6.16-3 RXDMA 描述符字 1 (TSEN (EMAC\_TSCTL[0]) 为 1)

**RXDES 2: RXDMA 描述符字 2**

RXDMA 描述符字 2 暂时保留

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

位	字段	描述
[31:0]	Reserved	保留

表 6.16-4 RXDMA 描述符字 2

**RXDES 3: RXDMA 描述符字 3**

RXDMA 描述符字 3 包含下一个 RXDMA 描述符的起始地址或时间戳的高 32 位。详情如下

31	30	29	28	27	26	25	24
NRXDSA/TSSEC							
23	22	21	20	19	18	17	16
NRXDSA/TSSEC							
15	14	13	12	11	10	9	8
NRXDSA/TSSEC							
7	6	5	4	3	2	1	0
NRXDSA/TSSEC							

位	字段	描述
[31:0]	<b>NRXDSA</b>	下一个 RX 描述符的起始地址 NRXDSA 为下一个 RX 描述符的起始地址当以太网 MAC 控制器获取下一个 RX 描述符时，它会忽略 NRXDSA[1:0]

表 6.16-5 RXDMA 描述符字 3 (TSEN (EMAC\_TSCTL[0]) 为 0)

位	字段	描述
[31:0]	<b>TSSEC</b>	时间戳秒 使能 TSEN (EMAC_TSCTL[0]) 以太网 MAC 控制器写 RX 描述符到系统内存时，会存储时间戳高 32 位 (ETSMSR) 到该字段

表 6.16-6 RXDMA 描述符字 3 (TSEN (EMAC\_TSCTL[0]) 为 1)

## 6.16.5.11 TxDMA 描述符数据结构

TxDMA 描述符包含 4 个 32 位字。TxDMA 描述符数据结构如图 6.16-5 所示。

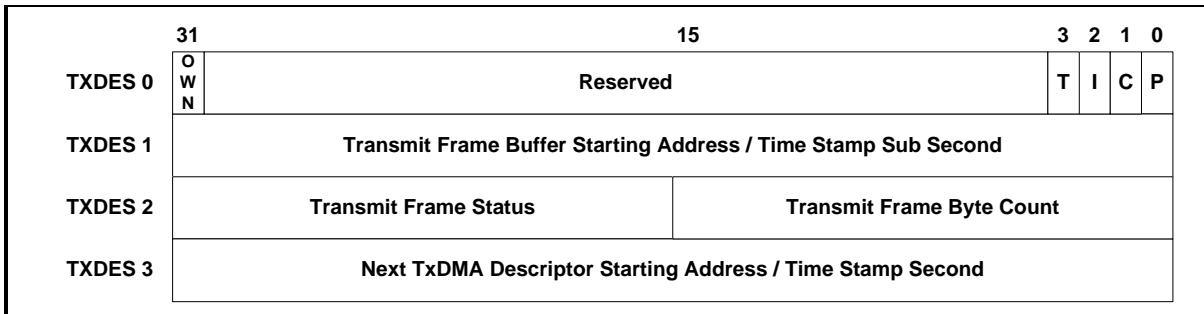


图 6.16-5 TxDMA 描述符数据结构

**TXDES 0: TXDMA 描述符字 0**

TXDMA 描述符字 0 包含一个描述符所有者标识。另外，它还包含了发送帧填充控制位，CRC 扩充，中断使能和时间戳控制。详情如下。

31	30	29	28	27	26	25	24
Owner	Reserved						
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved				TTSEN	INTEN	CRCAPP	PADEN

位	字段	描述
[31]	Owner	<b>所有者</b> 该寄存器位表示 TX 描述符的归属，是 CPU 还是 EMAC。只有描述符的所有者才能修改 TX 描述符，其他只能读描述符。 O = 1'b1 表示 EMAC TXDMA 是 TX 描述符的所有者，帧发送的 TX 描述符可用。帧发送完成后，EMAC RXDMA 修改该位的归属为 1'b0，CPU 获得 TX 描述符所有权 O = 1'b0 表示 CPU 是 TX 描述符的所有者。CPU 准备好新帧等待发送，修改该位的归属为 1'b1，EMAC RXDMA 获得 TX 描述符的所有权 0 = 所有者为 CPU. 1 = 所有者为 EMAC.
[30:4]	Reserved	保留
[3]	TTSEN	<b>TX 时间戳使能位</b> 该位置 1 且 IEEE1588 PTP 功能使能，内置时间戳电路会在发送完成帧的 SFD 后给该帧打上时间戳 0 = IEEE 1588 时间戳功能禁止. 1 = IEEE 1588 时间戳功能使能.
[2]	INTEN	<b>发送中断使能位</b> 帧传送完成后，INTEN 控制着是否触发中断。如果使能了 INTEN，帧传送完成后 EMAC 会触发中断，未使能则不会触发。 0 = 帧发送中断屏蔽. 1 = 帧发送中断使能
[1]	CRCAPP	<b>CRC 扩充</b> CRCAPP 控制着是否在帧发送时附加 CRC，如果 CRCAPP 使能，4 字节的 CRC 校验和会被附加到发送帧的结尾。 0 = 扩充 4-字节 CRC 禁止. 1 = 扩充 4-字节 CRC 使能.

[0]	<b>PADEN</b>	<b>填充使能位</b> PADEN 控制是否在帧长度小于 60 字节时附加填充位。如果 PADEN 使能，EMAC 会自动进行填充。 0 = 禁止附加填充位。 1 = 使能附加填充位。
-----	--------------	--

表 6.16-7 TxDMA 描述符字 0

**TXDES 1: TXDMA 描述符字 1**

TXDMA 描述符字 1 包括，发送帧缓存起始地址或时间戳低 32 位。详情如下：

31	30	29	28	27	26	25	24
TXBSA/TSSUBSEC							
23	22	21	20	19	18	17	16
TXBSA/TSSUBSEC							
15	14	13	12	11	10	9	8
TXBSA/TSSUBSEC							
7	6	5	4	3	2	1	0
TXBSA/TSSUBSEC							

位	字段	描述
[31:2]	TXBSA	<b>Transmit Buffer Starting Address</b> The TXBSA is the starting address of buffer where transmit packet data stored.

表 6.16-8 TXDMA 描述符字 1 (TSEN (EMAC\_TSCTL[0]) is 0)

位	字段	描述
[31:0]	TSSUBSEC	<b>时间戳亚秒</b> 使能 TSEN (EMAC_TSCTL[0]), 且 TX 描述符字 0 的 TTSEN 使能, 以太网 MAC 控制器写 TX 描述符到系统内存时, 会存储时间戳低 32 位 (EMAC_TSSUBSEC) 到该字段

表 6.16-9 TXDMA 描述符字 1 (TSEN (EMAC\_TSCTL[0]) is 1)

**TXDES 2: TXDMA 描述符字 2**

TXDMA 描述符字 2 包含发送帧状态和发送帧字节数，详情如下：

31	30	29	28	27	26	25	24
COLCNT				TTSAS	SQE	TXPAUSED	TXHALT
23	22	21	20	19	18	17	16
LCIF	TXABTIF	NCSIF	EXDEFIF	TXCPIF	Reserved	DEF	TXIF
15	14	13	12	11	10	9	8
TBC							
7	6	5	4	3	2	1	0
TBC							

位	字段	描述
[31:28]	<b>COLCNT</b>	<b>冲突数</b> COLCNT 表示在包传递过程中发生的连续冲突数。如果传输时连续冲突数达 16 个，COLCNT 归 0x0，TXABTIF 位置位
[27]	<b>TTSAS</b>	<b>TX 时间戳状态</b> 该位表示帧成功获得时间戳。该位置高时，TX 描述符字 1 和 TX 描述符字 3 在帧的 SFD 发送后保存时间戳的值 0 = TX 描述符字 1 和 TX 描述符字 3 未保存时间戳的值 1 = TX 描述符字 1 和 TX 描述符字 3 已保存时间戳的值
[26]	<b>SQE</b>	<b>SQE 错误</b> SQE 表示在 10 Mbps 半双工模式下包传送结尾发现 SQE 错误。EMAC 工作在 10Mbps 半双工模式下时，SQECHKEN (EMAC_CTL[17]) 使能，则会进行 SQE 错误检测。 0 = 包发送结尾未发现 SQE 错误。 1 = 包发送结尾发现 SQE 错误。
[25]	<b>TXPAUSED</b>	<b>发送暂停</b> 如果 EMAC 接收到暂停帧，或软件置位 SDPZ (EMAC_MCM[16]) 且 EMAC 发送了一个暂停帧出去，TXPAUSED 表示下个正常的包发送过程会暂停。 0 = 下一个正常的包发送过程正常 1 = 下一个正常的包发送过程暂停
[24]	<b>TXHALT</b>	<b>发送停止</b> TXON (EMAC_CTL[8]) 位由软件关闭，下一个正常的包发送过程停止 0 = 下一个正常的包发送过程正常 1 = 下一个正常的包发送过程停止
[23]	<b>LCIF</b>	<b>迟到的冲突</b> LCIF 表示在 64 字节的冲突窗口外发生了冲突。即在 64 字节长度的帧发送出去后仍然有冲突存在。迟到的冲突检测仅在 EMAC 半双工模式下有效。 0 = 在 64 字节的冲突窗口外未发生冲突 1 = 在 64 字节的冲突窗口外发生了冲突

[22]	<b>TXABTIF</b>	<b>发送取消</b> TXABTIF 表示在发送过程中由于连续出现了 16 次冲突而取消了包发送。发送取消功能仅在 EMAC 半双工模式下有效。 0 = 在发送过程中没有连续出现 16 次冲突 1 = 在发送过程中出现了连续 16 次冲突
[21]	<b>NCSIF</b>	<b>无载波监听</b> NCSIF 表示 MII I/F 信号 CRS 在包传送开始无效。 NCSIF 仅在 EMAC 半双工模式下有效 0 = MII I/F 信号 CRS 在包传送开始无效 1 = CRS 信号正常.
[20]	<b>EXDEFIF</b>	<b>延期超时</b> EXDEFIF 表示在 100Mbps 时，发送时帧等待发送超过了 0.32768ms.或者在 10 Mbps 时，等待超过 3.2768ms.延期超时检测仅在 NODEF (EMAC_CTL[9]) 位关闭，EMAC 工作在半双工模式下有效。 0 = 未发生延期超时 1 = 在 100Mbps 时，发送时帧等待发送超过了 0.32768ms.或者在 10 Mbps 时，等待超过 3.2768ms.
[19]	<b>TXCPIF</b>	<b>发送完成</b> TXCPIF 表示发送正确完成 0 = 发送未完成. 1 = 发送已完成
[18]	<b>Reserved</b>	保留
[17]	<b>DEF</b>	<b>发送延期</b> DEF 表示发送延期一次。 DEF 检测仅在半双工模式下有效 0 = 包发送未延期 1 = 包发送延期
[16]	<b>TXIF</b>	<b>发送中断</b> TXIF 表示包发送会触发中断 0 = 包发送未触发中断 1 = 包发送会触发中断
[15:0]	<b>TBC</b>	<b>发送字节数</b> TBC 表示 TX 描述符指向的数据缓存地址里的帧的数目

表 6.16-10 TXDMA 描述符字 2

**TXDES 3: TXDMA 描述符字 3**

TXDMA 描述符字 3 包含下一个 TXDMA 描述符的起始地址或时间戳的高 32 位。详情如下。

31	30	29	28	27	26	25	24
NTXDSA/TSSEC							
23	22	21	20	19	18	17	16
NTXDSA/TSSEC							
15	14	13	12	11	10	9	8
NTXDSA/TSSEC							
7	6	5	4	3	2	1	0
NTXDSA/TSSEC							

位	字段	描述
[31:0]	NTXDSA	<b>下一个 TX 描述符的起始地址</b> NTXDSA 为下一个 TX 描述符的起始地址当以太网 MAC 控制器获取下一个 TX 描述符时, 它会忽略 NTXDSA [1:0]

表 6.16-11 TXDMA 描述符字 3 (TSEN (EMAC\_TSCTL[0]) 为 0)

位	字段	描述
[31:0]	TSSEC	<b>时间戳秒</b> 使能 TSEN (EMAC_TSCTL[0]), 且 TX 描述符字 0 的 TTSEN 使能, 以太网 MAC 控制器写 TX 描述符到系统内存时, 会存储时间戳高 32 位 (EMAC_TSSEC) 到该字段

表 6.16-12 TXDMA 描述符字 3 (TSEN (EMAC\_TSCTL[0]) 为 1)

### 6.16.6 寄存器映射

R: 只读, W: 只写, R/W: 读/写

寄存器	偏移量	R/W	描述	复位值
<b>EMAC 基地址:</b>				
<b>EMAC_BA = 0x4000_B000</b>				
<b>EMAC_CAMCTL</b>	EMAC_BA+0x000	R/W	CAM 比较控制寄存器	0x0000_0000
<b>ECAM_CAMEN</b>	EMAC_BA+0x004	R/W	CAM 使能寄存器	0x0000_0000
<b>EMAC_CAM0M</b>	EMAC_BA+0x008	R/W	CAM0 高位字寄存器	0x0000_0000
<b>EMAC_CAM0L</b>	EMAC_BA+0x00C	R/W	CAM0 低位字寄存器	0x0000_0000
<b>EMAC_CAM1M</b>	EMAC_BA+0x010	R/W	CAM1 高位字寄存器	0x0000_0000
<b>EMAC_CAM1L</b>	EMAC_BA+0x014	R/W	CAM1 低位字寄存器	0x0000_0000
<b>EMAC_CAM2M</b>	EMAC_BA+0x018	R/W	CAM2 高位字寄存器	0x0000_0000
<b>EMAC_CAM2L</b>	EMAC_BA+0x01C	R/W	CAM2 低位字寄存器	0x0000_0000
<b>EMAC_CAM3M</b>	EMAC_BA+0x020	R/W	CAM3 高位字寄存器	0x0000_0000
<b>EMAC_CAM3L</b>	EMAC_BA+0x024	R/W	CAM3 低位字寄存器	0x0000_0000
<b>EMAC_CAM4M</b>	EMAC_BA+0x028	R/W	CAM4 高位字寄存器	0x0000_0000
<b>EMAC_CAM4L</b>	EMAC_BA+0x02C	R/W	CAM4 低位字寄存器	0x0000_0000
<b>EMAC_CAM5M</b>	EMAC_BA+0x030	R/W	CAM5 高位字寄存器	0x0000_0000
<b>EMAC_CAM5L</b>	EMAC_BA+0x034	R/W	CAM5 低位字寄存器	0x0000_0000
<b>EMAC_CAM6M</b>	EMAC_BA+0x038	R/W	CAM6 高位字寄存器	0x0000_0000
<b>EMAC_CAM6L</b>	EMAC_BA+0x03C	R/W	CAM6 低位字寄存器	0x0000_0000
<b>EMAC_CAM7M</b>	EMAC_BA+0x040	R/W	CAM7 高位字寄存器	0x0000_0000
<b>EMAC_CAM7L</b>	EMAC_BA+0x044	R/W	CAM7 低位字寄存器	0x0000_0000
<b>EMAC_CAM8M</b>	EMAC_BA+0x048	R/W	CAM8 高位字寄存器	0x0000_0000
<b>EMAC_CAM8L</b>	EMAC_BA+0x04C	R/W	CAM8 低位字寄存器	0x0000_0000
<b>EMAC_CAM9M</b>	EMAC_BA+0x050	R/W	CAM9 高位字寄存器	0x0000_0000
<b>EMAC_CAM9L</b>	EMAC_BA+0x054	R/W	CAM9 低位字寄存器	0x0000_0000
<b>EMAC_CAM10M</b>	EMAC_BA+0x058	R/W	CAM10 高位字寄存器	0x0000_0000
<b>EMAC_CAM10L</b>	EMAC_BA+0x05C	R/W	CAM10 低位字寄存器	0x0000_0000
<b>EMAC_CAM11M</b>	EMAC_BA+0x060	R/W	CAM11 高位字寄存器	0x0000_0000
<b>EMAC_CAM11L</b>	EMAC_BA+0x064	R/W	CAM11 低位字寄存器	0x0000_0000

<b>EMAC_CAM12M</b>	EMAC_BA+0x068	R/W	CAM12 高位字寄存器	0x0000_0000
<b>EMAC_CAM12L</b>	EMAC_BA+0x06C	R/W	CAM12 低位字寄存器	0x0000_0000
<b>EMAC_CAM13M</b>	EMAC_BA+0x070	R/W	CAM13 高位字寄存器	0x0000_0000
<b>EMAC_CAM13L</b>	EMAC_BA+0x074	R/W	CAM13 低位字寄存器	0x0000_0000
<b>EMAC_CAM14M</b>	EMAC_BA+0x078	R/W	CAM14 高位字寄存器	0x0000_0000
<b>EMAC_CAM14L</b>	EMAC_BA+0x07C	R/W	CAM14 低位字寄存器	0x0000_0000
<b>EMAC_CAM15MSB</b>	EMAC_BA+0x080	R/W	CAM15 高位字寄存器	0x0000_0000
<b>EMAC_CAM15LSB</b>	EMAC_BA+0x084	R/W	CAM15 低位字寄存器	0x0000_0000
<b>EMAC_TXDSA</b>	EMAC_BA+0x088	R/W	发送描述符链表起始寄存器	0xFFFF_FFFC
<b>EMAC_RXDSA</b>	EMAC_BA+0x08C	R/W	接收描述符链表起始寄存器	0xFFFF_FFFC
<b>EMAC_CTL</b>	EMAC_BA+0x090	R/W	MAC 控制寄存器	0x0040_0000
<b>EMAC_MIIMDAT</b>	EMAC_BA+0x094	R/W	MII 管理数据寄存器	0x0000_0000
<b>EMAC_MIIMCTL</b>	EMAC_BA+0x098	R/W	MII 管理控制和地址寄存器	0x0090_0000
<b>EMAC_FIFOCTL</b>	EMAC_BA+0x09C	R/W	FIFO 阈值控制寄存器	0x0000_0101
<b>EMAC_TXST</b>	EMAC_BA+0x0A0	W	请求开始发送寄存器	Undefined
<b>EMAC_RXST</b>	EMAC_BA+0x0A4	W	请求开始接收寄存器	Undefined
<b>EMAC_MRFL</b>	EMAC_BA+0x0A8	R/W	最大接收帧控制寄存器	0x0000_0800
<b>EMAC_INTEN</b>	EMAC_BA+0x0AC	R/W	MAC 中断使能寄存器	0x0000_0000
<b>EMAC_INTSTS</b>	EMAC_BA+0x0B0	R/W	MAC 中断状态寄存器	0x0000_0000
<b>EMAC_GENSTS</b>	EMAC_BA+0x0B4	R/W	MAC 通用状态寄存器	0x0000_0000
<b>EMAC_MPCNT</b>	EMAC_BA+0x0B8	R/W	丢包计数寄存器	0x0000_7FFF
<b>EMAC_RPCNT</b>	EMAC_BA+0x0BC	R	MAC 接收到的暂停帧计数寄存器	0x0000_0000
<b>EMAC_FRSTS</b>	EMAC_BA+0x0C8	R/W	DMA 接收帧状态寄存器	0x0000_0000
<b>EMAC_CTXDSA</b>	EMAC_BA+0x0CC	R	当前发送描述符起始地址寄存器	0x0000_0000
<b>EMAC_CTXBSA</b>	EMAC_BA+0x0D0	R	当前发送缓存起始地址寄存器	0x0000_0000
<b>EMAC_CRXDSA</b>	EMAC_BA+0x0D4	R	当前接收描述符起始地址寄存器	0x0000_0000
<b>EMAC_CRXBSA</b>	EMAC_BA+0x0D8	R	当前接收缓存起始地址寄存器	0x0000_0000
<b>EMAC_TSCTL</b>	EMAC_BA+0x100	R/W	时间戳控制寄存器	0x0000_0000
<b>EMAC_TSSEC</b>	EMAC_BA+0x110	R	时间戳秒计数器寄存器	0x0000_0000
<b>EMAC_TSSUBSEC</b>	EMAC_BA+0x114	R	时间戳亚秒计数器寄存器	0x0000_0000

<b>EMAC_TSINC</b>	EMAC_BA+0x118	R/W	时间戳增量寄存器	0x0000_0000
<b>EMAC_TSADDEND</b>	EMAC_BA+0x11C	R/W	时间戳加数寄存器	0x0000_0000
<b>EMAC_UPDSEC</b>	EMAC_BA+0x120	R/W	时间戳更新秒寄存器	0x0000_0000
<b>EMAC_UPDSUBSEC</b>	EMAC_BA+0x124	R/W	时间戳更新亚秒寄存器	0x0000_0000
<b>EMAC_ALMSEC</b>	EMAC_BA+0x128	R/W	时间戳报警秒寄存器	0x0000_0000
<b>EMAC_ALMSUBSEC</b>	EMAC_BA+0x12C	R/W	时间戳报警亚秒寄存器	0x0000_0000

### 6.16.7 寄存器描述

#### EMAC\_CAMCTL CAM 命令寄存器

EMAC 支持 CAM 功能以识别目标 MAC 地址。EMAC\_CAMCTL 控制 CAM 比较功能，单播，组播和广播包接收。

寄存器	偏移量	R/W	描述	复位值
EMAC_CAMCTL	EMAC_BA+0x000	R/W	CAM 比较控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved			CMPEN	COMPEN	ABP	AMP	AUP

位	描述	
[31:5]	Reserved	保留
[4]	CMPEN	<b>CAM 比较使能位</b> CMPEN 控制着是否使能 CAM 比较功能 (识别目标 MAC 地址)。如果软件想要接收特定目标 MAC 地址的封包，向 CAM 12-0 填入相应 MAC 地址，接着使能 CAM 条目，置 CMPEN 为 1 0 = CAM 比较功能 (识别目标 MAC 地址) 禁止 1 = CAM 比较功能 (识别目标 MAC 地址) 使能
[3]	COMPEN	<b>CAM 余集比较使能位</b> COMPEN 是 CAM 比较结果的余集。如果同时使能了 CMPEN 和 COMPEN，定义在 CAM 条目的特定目标 MAC 地址的封包将会被丢弃。其余来自不符合定义的 MAC 地址的封包将会被接收。 0 = CAM 余集比较禁止 1 = CAM 余集比较使能
[2]	ABP	<b>接收广播地址</b> ABP 控制着是否接收广播包。如果使能了 ABP，EMAC 会接收所有目标 MAC 地址为广播地址的输入包。 0 = EMAC 接收 CAM 比较结果指定地址的封包。 1 = EMAC 接收所有广播包。
[1]	AMP	<b>接收组播包</b> AMP 控制着是否接收组播包。如果使能了 AMP，EMAC 会接收所有目标 MAC 地址为组播地址的输入包。 0 = EMAC 接收 CAM 比较结果指定地址的封包。 1 = EMAC 接收所有组播包。

[0]	AUP	接收单包
		AUP 控制着是否接收单播包。如果使能了 AUP，EMAC 会接收所有目标 MAC 地址为单播地址的输入包。
		0 = EMAC 接收 CAM 比较结果指定地址的封包。 1 = EMAC 接收所有单播包。

**CAMCMR 设置和比较结果**

表 7.16 14 为不同 CAMCMR 配置下的地址识别结果。结果列显示了哪些输入封包类型可以通过 CAM 配置的可识别地址。C, U, M 和 B 分别代表

C: CAM 条目定义了输入封包的目标地址.

U: 输入封包为单播包

M: 输入封包为组播包

B: 输入封包为广播包

CMPPEN	CCAM	AUP	AMP	ABP	结果			
0	0	0	0	0	无封包			
0	0	0	0	1	B			
0	0	0	1	0	M			
0	0	0	1	1	M	B		
0	0	1	0	0	C	U		
0	0	1	0	1	C	U	B	
0	0	1	1	0	C	U	M	
0	0	1	1	1	C	U	M	B
0	1	0	0	0	C	U	M	B
0	1	0	0	1	C	U	M	B
0	1	0	1	1	C	U	M	B
0	1	1	0	0	C	U	M	B
0	1	1	0	1	C	U	M	B
0	1	1	1	0	C	U	M	B
1	0	0	0	0	C			
1	0	0	0	1	C	B		
1	0	0	1	0	C	M		
1	0	1	1	1	C	M	B	
1	0	1	0	0	C	U		
1	0	1	0	1	C	U	B	
1	0	1	1	0	C	U	M	

1	0	1	1	1	C	U	M	B
1	1	0	0	0	U	M	B	
1	1	0	0	1	U	M	B	
1	1	0	1	0	U	M	B	
1	1	0	1	1	U	M	B	
1	1	1	0	0	C	U	M	B
1	1	1	0	1	C	U	M	B
1	1	1	1	0	C	U	M	B
1	1	1	1	1	C	U	M	B

表 6.16-13 不同 CAMCMR 设置及可接收的包类型

**ECAM\_CAMEN CAM 使能寄存器**

ECAM\_CAMEN 控制着各 CAM 条目的有效性。每个 CAM 条目在参与目标 MAC 地址识别前需要提前使能。

寄存器	偏移量	R/W	描述	复位值
<b>ECAM_CAMEN</b>	EMAC_BA+0x004	R/W	CAM 使能寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
CAM15EN	CAM14EN	CAM13EN	CAM12EN	CAM11EN	CAM10EN	CAM9EN	CAM8EN
7	6	5	4	3	2	1	0
CAM7EN	CAM6EN	CAM5EN	CAM4EN	CAM3EN	CAM2EN	CAM1EN	CAM0EN

位	描述	
[31:16]	Reserved	保留
[x]	CAMxEN	<b>CAM 条目 X 使能位</b> ECAM_CAMEN 控制着各 CAM 条目 X 的有效性。 CAM 条目 13, 14 和 15 用于暂停帧的发送。软件发送暂停帧之前需要先使能这些 CAM 条目。 0 = CAM 条目 x 禁止。 1 = CAM 条目 x 使能。

**EMAC\_CAMxMSB, x = 0, 1, 2..14 CAM 条目寄存器**

EMAC 总共有 16 个 CAM 条目。在这 16 个 CAM 条目中, 有 13 个 (0~12 条目) 用于存储包识别的目标 MAC 地址。其他 3 个 (13~15 条目) 用于暂停帧的传输。每个条目包含 6 个字节, 所以需要两个寄存器。

寄存器对 {EMAC\_CAMxMSB, ECAMxL} 是用于存储包识别的目标 MAC 地址的 CAM 条目。使用时必须使能 CAMxEN (ECAM\_CAMEN[x]) 对应位,  $x = 0 \sim 12$ .

寄存器对 {EMAC\_CAM13M, EMAC\_CAM13L}, {EMAC\_CAM14M, EMAC\_CAM14L} 和 {EMAC\_CAM15MSB, EMAC\_CAM15LSB} 用于流控。

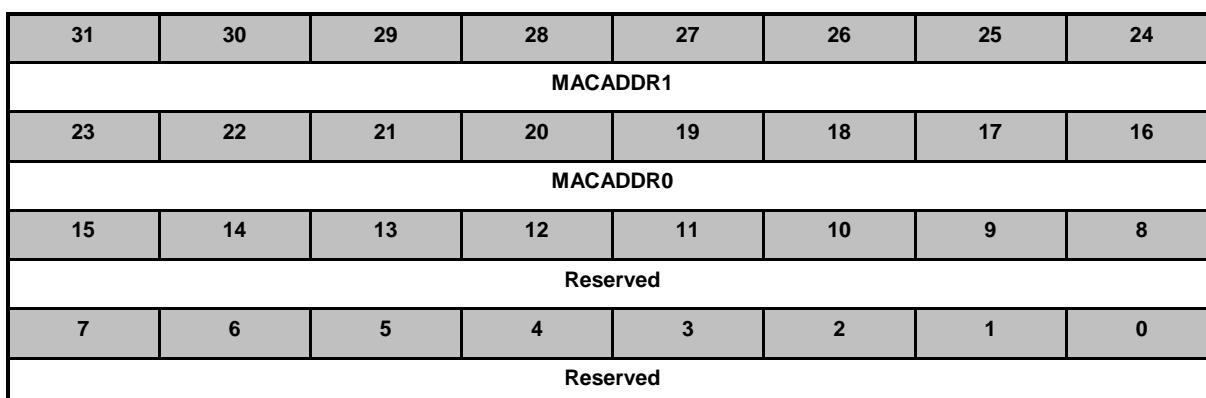
寄存器	偏移量	R/W	描述	复位值
<b>EMAC_CAM0M</b>	EMAC_BA+0x008	R/W	CAM0 高位字寄存器	0x0000_0000
<b>EMAC_CAM1M</b>	EMAC_BA+0x010	R/W	CAM1 高位字寄存器	0x0000_0000
<b>EMAC_CAM2M</b>	EMAC_BA+0x018	R/W	CAM2 高位字寄存器	0x0000_0000
<b>EMAC_CAM3M</b>	EMAC_BA+0x020	R/W	CAM3 高位字寄存器	0x0000_0000
<b>EMAC_CAM4M</b>	EMAC_BA+0x028	R/W	CAM4 高位字寄存器	0x0000_0000
<b>EMAC_CAM5M</b>	EMAC_BA+0x030	R/W	CAM5 高位字寄存器	0x0000_0000
<b>EMAC_CAM6M</b>	EMAC_BA+0x038	R/W	CAM6 高位字寄存器	0x0000_0000
<b>EMAC_CAM7M</b>	EMAC_BA+0x040	R/W	CAM7 高位字寄存器	0x0000_0000
<b>EMAC_CAM8M</b>	EMAC_BA+0x048	R/W	CAM8 高位字寄存器	0x0000_0000
<b>EMAC_CAM9M</b>	EMAC_BA+0x050	R/W	CAM9 高位字寄存器	0x0000_0000
<b>EMAC_CAM10M</b>	EMAC_BA+0x058	R/W	CAM10 高位字寄存器	0x0000_0000
<b>EMAC_CAM11M</b>	EMAC_BA+0x060	R/W	CAM11 高位字寄存器	0x0000_0000
<b>EMAC_CAM12M</b>	EMAC_BA+0x068	R/W	CAM12 高位字寄存器	0x0000_0000
<b>EMAC_CAM13M</b>	EMAC_BA+0x070	R/W	CAM13 高位字寄存器	0x0000_0000
<b>EMAC_CAM14M</b>	EMAC_BA+0x078	R/W	CAM14 高位字寄存器	0x0000_0000

31	30	29	28	27	26	25	24
MACADDR5							
23	22	21	20	19	18	17	16
MACADDR4							
15	14	13	12	11	10	9	8
MACADDR3							
7	6	5	4	3	2	1	0
MACADDR2							

位	描述	
[31:24]	<b>MACADDR5</b>	<b>MAC 地址字节 5</b> CAMxM 保存 MAC 地址的 47~16 位。x 为 0~14。寄存器对 {EMAC_CAMxM, EMAC_CAMxL} 代表一个 CAM 条目，保存整个 MAC 地址。 例如 CAM 条目 1 保存的 MAC 地址是 00-50-BA-33-BA-44，寄存器 EMAC_CAM1M 是 0x0050_BA33，寄存器 EMAC_CAM1L 是 0xBA44_0000.
[23:16]	<b>MACADDR4</b>	<b>MAC 地址字节 4</b>
[15:8]	<b>MACADDR3</b>	<b>MAC 地址字节 3</b>
[7:0]	<b>MACADDR2</b>	<b>MAC 地址字节 2</b>

## EMAC CAMxLSB; x = 0, 1, 2..14 CAM 条目寄存器

寄存器	偏移量	R/W	描述	复位值
EMAC_CAM0L	EMAC_BA+0x00C	R/W	CAM0 低位字寄存器	0x0000_0000
EMAC_CAM1L	EMAC_BA+0x014	R/W	CAM1 低位字寄存器	0x0000_0000
EMAC_CAM2L	EMAC_BA+0x01C	R/W	CAM2 低位字寄存器	0x0000_0000
EMAC_CAM3L	EMAC_BA+0x024	R/W	CAM3 低位字寄存器	0x0000_0000
EMAC_CAM4L	EMAC_BA+0x02C	R/W	CAM4 低位字寄存器	0x0000_0000
EMAC_CAM5L	EMAC_BA+0x034	R/W	CAM5 低位字寄存器	0x0000_0000
EMAC_CAM6L	EMAC_BA+0x03C	R/W	CAM6 低位字寄存器	0x0000_0000
EMAC_CAM7L	EMAC_BA+0x044	R/W	CAM7 低位字寄存器	0x0000_0000
EMAC_CAM8L	EMAC_BA+0x04C	R/W	CAM8 低位字寄存器	0x0000_0000
EMAC_CAM9L	EMAC_BA+0x054	R/W	CAM9 低位字寄存器	0x0000_0000
EMAC_CAM10L	EMAC_BA+0x05C	R/W	CAM10 低位字寄存器	0x0000_0000
EMAC_CAM11L	EMAC_BA+0x064	R/W	CAM11 低位字寄存器	0x0000_0000
EMAC_CAM12L	EMAC_BA+0x06C	R/W	CAM12 低位字寄存器	0x0000_0000
EMAC_CAM13L	EMAC_BA+0x074	R/W	CAM13 低位字寄存器	0x0000_0000
EMAC_CAM14L	EMAC_BA+0x07C	R/W	CAM14 低位字寄存器	0x0000_0000



位	描述	
[31:24]	MACADDR1	<b>MAC 地址字节 1</b> CAMxM 保存 MAC 地址的 15~0 位。x 为 0~14。寄存器对 {EMAC_CAMxM, EMAC_CAMxL} 代表一个 CAM 条目，保存整个 MAC 地址。 例如 CAM 条目 1 保存的 MAC 地址是 00-50-BA-33-BA-44，寄存器 EMAC_CAM1M 是 0x0050_BA33，寄存器 EMAC_CAM1L 是 0xBA44_0000。
[23:16]	MACADDR0	<b>MAC 地址字节 0</b>

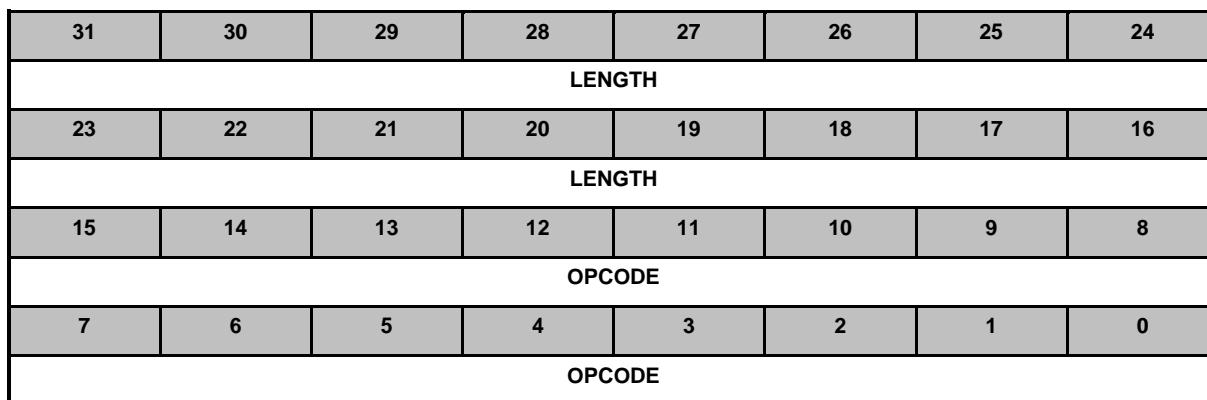
[15:0]

Reserved

保留

## EMAC CAM15MSB CAM 条目寄存器

寄存器	偏移量	R/W	描述	复位值
EMAC_CAM15MSB	EMAC_BA+0x080	R/W	CAM15 高位字寄存器	0x0000_0000



位	描述	
[31:16]	LENGTH	暂停帧的长度字段 在暂停帧里, 定义为 0x8808.
[15:0]	OPCODE	暂停帧的操作码字段 在暂停帧里, 定义为 0x0001.

**EMAC CAM15LSB CAM 条目寄存器**

寄存器	偏移量	R/W	描述	复位值
EMAC_CAM15LSB	EMAC_BA+0x084	R/W	CAM15 低位字寄存器	0x0000_0000

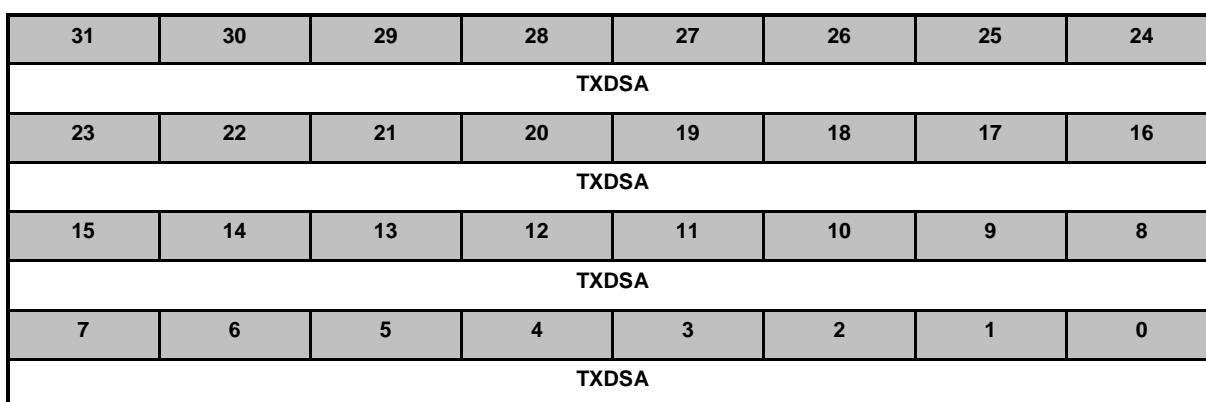
31	30	29	28	27	26	25	24
OPERAND							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved							

位	描述	
[31:24]	OPERAND	暂停参数 在暂停帧里，OPERAND 字段控制着目标地址以太网 MAC 控制器的暂停时间。 OPERAND 的单位是时间片，512 位时间
[23:0]	Reserved	保留

**EMAC\_TXDSA** 发送描述符链表起始地址寄存器

TX 描述符是 EMAC 中的一个链表数据结构。EMAC\_TXDSA 保存着该链表的起始地址，即 EMAC\_TXDSA 保存着第一个 TX 描述符的起始地址。软件必须在使能 TXON (EMAC\_CTL[8]) 前配置完 EMAC\_TXDSA。.

寄存器	偏移量	R/W	描述	复位值
EMAC_TXDSA	EMAC_BA+0x088	R/W	发送描述符链表起始寄存器	0xFFFF_FFFC



位	描述	
[31:0]	TXDSA	<b>发送描述符链表起始地址</b> TXDSA 存储着发送描述符链表起始地址。软件使能 TXON (EMAC_CTL[8]) 会将 TXDSA 的内容载入当前发送描述符起始地址寄存器 (EMAC_CTXDSA)。EMAC 不会再更新 TXDSA 的值。操作中，EMAC 会忽略 TXDSA[1:0] 的值。也就是说，TX 描述符一定要存储在内存地址的字边界。.

**EMAC\_RXDSA 接收描述符链表起始地址寄存器**

RX 描述符是 EMAC 中的一个链表数据结构。EMAC\_RXDSA 保存着该链表的起始地址，即 EMAC\_RXDSA 保存着第一个 RX 描述符的起始地址。软件必须在使能 RXON (EMAC\_CTL[0]) 前配置完 EMAC\_RXDSA。.

寄存器	偏移量	R/W	描述	复位值
EMAC_RXDSA	EMAC_BA+0x08C	R/W	接收描述符链表起始寄存器	0xFFFF_FFFC

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

位	描述	
[31:0]	RXDSA	<b>接收描述符链表起始地址</b> RXDSA 存储着发送描述符链表起始地址。软件使能 RXON (EMAC_CTL[0]) 会将 RXDSA 的内容载入当前接收描述符起始地址寄存器 (EMAC_RXDSA)。EMAC 不会再更新 RXDSA 的值。操作中，EMAC 会忽略 RXDSA[1:0] 的值。也就是说，RX 描述符一定要存储在内存地址的字边界。

EMAC\_CTL**MAC 控制寄存器**

EMAC\_CTL 提供了 EMAC 的控制信息。其中部分指令在帧的发送和接收中都有作用，如半/全双工选择位 FUDUP (EMAC\_CTL[18]), 100/10Mbps 速度选择位 OPMODE (EMAC\_CTL[20])。有些指令则分别控制帧的发送和接收如 TXON (EMAC\_CTL[8]) 和 RXON (EMAC\_CTL[0])

寄存器	偏移量	R/W	描述	复位值
EMAC_CTL	EMAC_BA+0x090	R/W	MAC 控制寄存器	0x0040_0000

31	30	29	28	27	26	25	24
Reserved							RST
23	22	21	20	19	18	17	16
Reserved	RMIIEN	Reserved	OPMODE	RMIIRXCTL	FUDUP	SQECHKEN	SDPZ
15	14	13	12	11	10	9	8
Reserved							NODEF
7	6	5	4	3	2	1	0
Reserved	WOLEN	STRIPCRC	AEP	ACP	ARP	ALP	RXON

位	描述	
[31:25]	Reserved	保留
[24]	RST	<b>软件复位</b> RST 实现了软件复位的功能，可以使 EMAC 恢复默认状态。RST 是一个自动清 0 的位。也就是说，在软件复位完成后，RST 会自动清 0。使能 RST 可以复位所有的控制和状态寄存器，除了 RMIIEN (EMAC_CTL[22]) 和 OPMODE (EMAC_CTL[20])。 软件复位完成后，EMAC 需要重新初始化 0 = 软件复位完成 1 = 软件复位使能
[23]	Reserved	保留
[22]	RMIIEN	<b>RMII 模式使能位</b> 此位控制通过 MII 还是 RMII 连接外部物理层芯片，复位 (EMAC_CTL[24]) 不影响 RMIIEN 的值 0 = RMII 模式禁止 1 = RMII 模式使能。 注：该字段必须为 1。
[21]	Reserved	保留
[20]	OPMODE	<b>工作模式选择</b> OPMODE 定义 EMAC 是处于 10Mbps 还是 100Mbps 模式。RST (EMAC_CTL[24]) 复位不影响 OPMODE 值 0 = EMAC 工作做在 10Mbps 模式。 1 = EMAC 工作做在 100Mbps 模式。

[19]	<b>RMIIRXCTL</b>	<b>RMII RX 控制</b> RMIIRXCTL 控制这 RMII 模式下的数据接收。RMIIEN (EMAC_CTL[ [22)) 置位时，该位必须使能 0 = RMII RX 控制禁止. 1 = RMII RX 控制使能.
[18]	<b>FUDUP</b>	<b>全双工模式选择</b> FUDUP 控制 EMAC 工作在半双工还是全双工模式下 0 = EMAC 工作在半双工模式下 1 = EMAC 工作在全双工模式下
[17]	<b>SQECHKEN</b>	<b>SQE 检验使能位</b> SQECHKEN 控制着 SQE 检验的使能。SQE 检测仅在 10Mbps 的半双工模式下有效。也就是说 SQECKEN 并不影响 EMAC 的工作模式。 0 = SQE 检验在 10Mbps 的双工模式下禁止 1 = SQE 检验在 10Mbps 的双工模式下使能
[16]	<b>SDPZ</b>	<b>发送暂停帧</b> SDPZ 控制这暂停帧的发送 软件发送暂停帧前，需要确保 CAM 条目 13, 14, 15 已配置完成且对应的 CAMEEN 寄存器使能位已置位。接着，置 SDPZ 为 1 使能暂停帧发送。 SDPZ 是自清零的位。也就是说，在暂停帧发送完成后，SDPZ 会自动清 0。 建议仅在全双工模式下使能 SNDPAUSE 0 = 暂停帧发送完成 1 = 暂停帧发送使能
[15:10]	<b>Reserved</b>	保留
[9]	<b>NODEF</b>	<b>关闭延期计数</b> NODEF 控制着是否关闭延期计数功能。如果 NODEF 置位，延期计数功能关闭。延期计数功能仅在 EMAC 半双工模式下有效。 0 = 使能延期计数 1 = 关闭延期计数
[8]	<b>TXON</b>	<b>帧发送启动</b> TXON 控制着 EMAC 正常包的发送。TXON 置位时，EMAC 开始发送封包的进程，包括获取 TX 描述符，包发送和 TX 描述符的修改。 使能 TXON 前需要完成 EMAC 初始化流程。否则，EMAC 将会不工作。 如果 EMAC 发送过程中关闭了 TXON，EMAC 会在当前帧发送完成后停止封包发送进程 0 = 包发送停止 1 = 包发送启动
[7]	<b>Reserved</b>	保留
[6]	<b>WOLEN</b>	<b>LAN 口唤醒使能位</b> WOKEN 置 1 使能，以太网控 MAC 制器会在检测到魔术包输入时从掉电模式唤醒系统 如果掉电模式下，以太网控制器检测到魔术包输入，以太网 MAC 控制器会产生一个唤醒事件来唤醒系统 0 = 魔术包唤醒功能禁止 1 = 魔术包唤醒功能使能

[5]	<b>STRIPCRC</b>	<b>去除 CRC</b> STRIPCRC 标示计算输入封包长度是否包含 4 字节的 CRC 校验和。如果 STRIPCRC 置 1，计算输入包长度时将不会计算 4 字节的 CRC 校验和。 0 = 计算输入包时包含 4 字节的 CRC 校验和 1 = 计算输入包时不包含 4 字节的 CRC 校验和
[4]	<b>AEP</b>	<b>接收 CRC 出错包</b> AEP 控制 EMAC 接收或丢弃 CRC 出错的包。如果 AEP 为 1，CRC 出错的包将被 EMAC 视为正常的封包进行接收 0 = EMAC 控制器丢弃 CRC 出错的包 1 = EMAC 控制器接收 CRC 出错的包
[3]	<b>ACP</b>	<b>接收控制包</b> ACP 控制着控制帧的接收。如果 ACP 为 1，EMAC 会接收控制帧。否则，控制帧会被丢弃。建议仅在 EMAC 全双工模式下使能 ACP 0 = EMAC 控制器丢弃控制帧 1 = EMAC 控制器接收控制帧
[2]	<b>ARP</b>	<b>接收短包</b> ARP 控制是否接收长度小于 64 字节的短包。如果 ARP 为 1，EMAC 接收短包，否则会丢弃短包。 0 = EMAC 控制器丢弃短包 1 = EMAC 控制器接收短包
[1]	<b>ALP</b>	<b>接收长包</b> ALP 控制是否接收长度大于 1518 字节的长包。如果 ALP 为 1，EMAC 接收长包，否则会丢弃长包。 0 = EMAC 控制器丢弃长包 1 = EMAC 控制器接收长包
[0]	<b>RXON</b>	<b>帧接收启动</b> RXON 控制着 EMAC 正常包的接收。RXON 置位时，EMAC 开始接收封包的进程，包括获取 RX 描述符，包接收和 RX 描述符的修改。 使能 RXON 前需要完成 EMAC 初始化流程。否则，EMAC 将会不工作。 如果 EMAC 接收过程中关闭了 RXON，EMAC 会在当前帧接收完成后停止封包接收进程 0 = 包接收停止 1 = 包接收启动

**EMAC\_MIIMDAT MII 管理数据寄存器**

EMAC 通过 MII 管理功能控制和读取外部 PHY 的寄存器状态。EMAC\_MIIMDAT 寄存器用来存储写入命令将要写入外部 PHY 寄存器的数据及读取命令从外部 PHY 寄存器读得的数据。

寄存器	偏移量	R/W	描述	复位值
EMAC_MIIMDAT	EMAC_BA+0x094	R/W	MII 管理数据寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
DATA							
7	6	5	4	3	2	1	0
DATA							

位	描述	
[31:16]	Reserved	保留
[15:0]	DATA	16 位的 DATA 为，写入命令将要写入外部 PHY 寄存器的数据或读取命令从外部 PHY 寄存器读得的数据

**EMAC\_MIIMCTL MII 管理控制和地址寄存器**

EMAC 通过 MII 管理功能控制和读取外部 PHY 的寄存器状态。EMAC\_MIIMCTL 寄存器用来存储 MII 管理的命令信息，如寄存器地址，外部 PHY 地址，MDC 时钟率，读/写命令等。

寄存器	偏移量	R/W	描述	复位值
EMAC_MIIMCTL	EMAC_BA+0x098	R/W	MII 管理控制和地址寄存器	0x0090_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved				MDCON	PREAMSP	BUSY	WRITE
15	14	13	12	11	10	9	8
Reserved			PHYADDR				
7	6	5	4	3	2	1	0
Reserved			PHYREG				

位	描述	
[31:20]	Reserved	保留
[19]	MDCON	<b>MDC 时钟使能</b> MDC 控制着 MDC 时钟的发生。MDC 置 1，MDC 时钟使能 0 = MDC 时钟关闭 1 = MDC 时钟使能
[18]	PREAMSP	<b>前导码禁止</b> PREAMSP 控制着 MII 管理帧的前导码产生。如果 PREAMSP 为高，MII 管理帧产生将会跳过前导码 0 = MII 管理帧产生将会有正常的前导码 1 = MII 管理帧产生将会跳过前导码
[17]	BUSY	<b>忙</b> 该位控制着 MII 管理帧发生的使能。软件操作外部 PHY 寄存器前，需要先置 该位为 1，再由 EMAC 产生 MII 管理帧到外部 PHY。该位自动清 0。也就是说，该位在 MII 管理命令完成后会自动清 0。 0 = MII 管理命令完成 1 = MII 管理命令使能
[16]	WRITE	<b>写命令</b> 该命令指示 MII 管理命令是读还是写 0 = MII 管理命令是读命令 1 = MII 管理命令是写命令
[15:13]	Reserved	保留

[12:8]	<b>PHYADDR</b>	<b>PHY 地址</b> PHYADDR 保存了 MII 管理命令的目标 PHY 地址
[7:5]	<b>Reserved</b>	保留
[4:0]	<b>PHYREG</b>	<b>PHY 寄存器 地址</b> PHYREG 保存了 MII 管理命令的目标 PHY 寄存器地址

**MII 管理功能帧格式**

IEEE Std. 802.3 的 22.2.4 章节定义了 MII 管理功能的相关规范。MII 管理功能是指对 PHY 的控制和状态读取。如下是 MII 管理帧的格式：

	管理帧字段							
	PRE	ST	OP	PHYAD	REGAD	TA	DATA	IDLE
READ	1...1	01	10	AAAAAA	RRRRR	Z0	DDDDDDDDDDDDDDDDDD	Z
WRITE	1...1	01	01	AAAAAA	RRRRR	10	DDDDDDDDDDDDDDDDDD	Z

表 6.16-14 MII 管理帧格式

**MII 管理功能配置时序**

读	写
<ol style="list-style-type: none"> <li>1. 配置合适的 EMAC_MDCCR.</li> <li>2. 配置 PHYADDR 和 PHYREG.</li> <li>3. 写 0 到 WRITE 寄存器</li> <li>4. 写 1 到 BUSY (EMAC_MIIMCTL[17]) , 发送一个 MII 管理帧</li> <li>5. 等 BUSY (EMAC_MIIMCTL[17]) 位变为 0 .</li> <li>6. 从 EMAC_MIIMDAT 寄存器读数据</li> <li>7. 完成</li> </ol>	<ol style="list-style-type: none"> <li>1. 写数据到 EMAC_MIIMDAT 寄存器</li> <li>2. 配置合适的 EMAC_MDCCR.</li> <li>3. 配置 PHYADDR 和 PHYREG.</li> <li>4. 写 1 到 WRITE 寄存器</li> <li>5. 写 1 到 BUSY (EMAC_MIIMCTL[17]) , 发送一个 MII 管理帧</li> <li>6. 等 BUSY (EMAC_MIIMCTL[17]) 位变为 0 .</li> <li>7. 完成</li> </ol>

表 6.16-15 MII 管理功能配置时序

**EMAC\_FIFOCCTL FIFO 阈值控制寄存器**

EMAC\_FIFOCCTL 定义了内部 FIFO 的高域和低域，包括 TXFIFO 和 RXFIFO。内部的 FIFO 域和 EMAC 请求发生及帧发送开始有关。EMAC\_FIFOCCTL 还定义了访问系统内存时的最大 AHB 总线时钟数。

寄存器	偏移量	R/W	描述	复位值
EMAC_FIFOCCTL	EMAC_BA+0x09C	R/W	FIFO 阈值控制寄存器	0x0000_0101

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved		BURSTLEN			Reserved		
15	14	13	12	11	10	9	8
Reserved						TXFIFOTH	
7	6	5	4	3	2	1	0
Reserved						RXFIFOTH	

位	描述	
[31:22]	Reserved	保留
[21:20]	BURSTLEN	<b>DMA 最大长度</b> 定义了 EMAC 访问系统内存时需要的最大 AHB 总线时钟数 00 = 4 字。 01 = 8 字。 10 = 16 字。 11 = 16 字。
[19:10]	Reserved	保留
[9:8]	TXFIFOTH	<b>TXFIFO 门限</b> TXFIFOTH 控制着系统内存到 TXFIFO 的 TXDMA 请求。TXFIFOTH 不仅定义了 TXFIFO 的下限值还定义了其上限值。一般来说，上限值是下限值的 2 倍。在包传送过程中，如果 TXFIFO 达到了上限值，TXDMA 会停止请求把数据从系统内存传到 TXFIFO。如果，TXFIFO 里的帧数据小于其下限值，TXDMA 会把系统内存的帧数据传入 TXFIFO。 TXFIFOTH 还定义了何时 TXMAC 开始把帧发送到外部网络。TXFIFO 中的数据第一次达到上限时，TXMAC 开始发送，若一帧数据个数少于上限，此帧所有数据全写入 TXFIFO 后也会启动发送。 00 = 未定义。 01 = TXFIFO 下限 64B 上限 128B。 10 = TXFIFO 下限 80B 上限 160B。 11 = TXFIFO 下限 96B 上限 192B。
[7:2]	Reserved	保留

[1:0]	RXFIFOTH	<p><b>RXFIFO 门限</b></p> <p>RXFIFOTH 控制着 RXFIFO 到系统内存的 RXDMA 请求。RXFIFOTH 不仅定义了 RXFIFO 的下限值还定义了其上限值。一般来说，上限值是下限值的 2 倍。在包接收过程中，如果 RXFIFO 达到了上限值，RXDMA 会把数据从 RXFIFO 传到系统内存。如果，RXFIFO 里的帧数据小于其下限值，RXDMA 会停止把帧数据传入系统内存。</p> <p>00 = 按突发传输的配置，若突发传输长 8 个字，此处上限亦 8 个字 01 = RXFIFO 上限 64B 下限 32B. 10 = RXFIFO 上限 128B 下限 64B. 11 = RXFIFO 上限 192B 下限 96B.</p>
-------	----------	--

**EMAC\_TXST 请求开始发送寄存器**

软件产生一个写命令到 EMAC\_TXST 寄存器使 TXDMA 脱离停止状态开始帧传送

寄存器	偏移量	R/W	描述	复位值
EMAC_TXST	EMAC_BA+0x0A0	W	Transmit Start Demand Register	Undefined

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

位	描述	
[31:0]	TXST	<b>发送开始请求</b> TXON (EMAC_CTL[8]) 使能后, 若 TX 描述符仍不可用, TXDMA 的 FSM 机制进入中止状态, TX 描述符准备好以后, 必须写个命令到存器 EMAC_TXST, 让 TXDMA 继续发送。 此寄存器只写。读, 未定义。 仅当 TXDMA 处于关闭状态时, 写该寄存器才有效

**EMAC\_RXST 请求开始接收寄存器**

软件产生一个写命令到 EMAC\_RXST 寄存器使 RXDMA 脱离停止状态开始帧接收

寄存器	偏移量	R/W	描述	复位值
EMAC_RXST	EMAC_BA+0x0A4	W	请求开始接收寄存器	Undefined

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

位	描述	
[31:0]	RXST	<b>接收开始请求</b> 使能 RXON (EMAC_CTL[0]) 以后，若 RX 描述符仍不可用，RXDMA 机制将进入中止状态。RX 描述符准备好以后，写一个命令到此寄存器，RXDMA 将继续接收。 此寄存器只写。读，未定义。 仅当 RXDMA 处于关闭状态时，写该寄存器才有效

**EMAC\_MRFL 最大接收帧控制寄存器**

EMAC\_MRFL 定义了内存可以存储的最大接收帧长度。建议仅在软件希望接收超过 1518 字节的帧时才使用该寄存器

寄存器	偏移量	R/W	描述	复位值
EMAC_MRFL	EMAC_BA+0x0A8	R/W	最大接收帧控制寄存器	0x0000_0800

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
MRFL							
7	6	5	4	3	2	1	0
MRFL							

位	描述	
[31:16]	Reserved	保留
[15:0]	MRFL	<b>最大接收帧长度</b> MRFL 定义了最大接收帧长度. 接收帧长度超过此定义时, 位 MFLEIF (EMAC_INTSTS[8]) 置 1, 若 MFLEIEN (EMAC_INTEN[8]) =1, 将触发 RX 中断. 建议仅在软件希望接收超过 1518 字节的帧时才使用该寄存器

**EMAC\_INTEN MAC 中断使能及存储区**

EMAC\_INTEN 控制着 EMAC 的中断使能。从 EMAC 到 CPU 总共有两个中断，帧接收中断 RXIF 和帧发送中断 TXIF。

寄存器	偏移量	R/W	描述	复位值
EMAC_INTEN	EMAC_BA+0x0AC	R/W	MAC 中断使能寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved			TSALMIEN	Reserved			TXBEIEN
23	22	21	20	19	18	17	16
TDUIEN	LCIEN	TXABTIEN	NCSIEN	EXDEFIEN	TXCPIEN	TXUDIEN	TXIEN
15	14	13	12	11	10	9	8
WOLIEN	CFRIEN	Reserved		RXBEIEN	RDUIEN	DENIEN	MFLEIEN
7	6	5	4	3	2	1	0
MPCOVIEN	RPIEN	ALIEIEN	RXGDIEN	LPIEN	RXOVIEN	CRCEIEN	RXIEN

位	描述
[31:29]	Reserved 保留
[28]	TSALMIEN 时间戳报警中断使能位 TSALMIEN 控制着 TSALMIF (EMAC_INTSTS[28]) 中断的发生。如果 TSALMIF (EMAC_INTSTS[28]) 置位，且 TSALMIEN 和 TXIEN (EMAC_INTEN[16]) 都使能了，EMAC 将触发 TX 中断到 CPU。如果 TSALMIEN 或 TXIEN (EMAC_INTEN[16]) 关闭了，即使 TXTSALMIF (EMAC_INTSTS[28]) 置位也不会触发 TX 中断。 0 = 禁止 TXTSALMIF (EMAC_INTSTS[28]) 触发 TX 中断。 1 = 使能 TXTSALMIF (EMAC_INTSTS[28]) 触发 TX 中断。
[27:25]	Reserved 保留
[24]	TXBEIEN 发送总线错误中断使能位 TXBEIEN 控制着 TXBEIF (EMAC_INTSTS[24]) 中断的发生。如果 TXBEIF (EMAC_INTSTS[24]) 置位，且 TXBEIEN 和 TXIEN (EMAC_INTEN[16]) 都使能了，EMAC 将触发 TX 中断到 CPU。如果 TXBEIEN 或 TXIEN (EMAC_INTEN[16]) 关闭了，即使 TXBEIF (EMAC_INTSTS[24]) 置位也不会触发 TX 中断。 0 = 禁止 TXBEIF (EMAC_INTSTS[24]) 触发 TX 中断。 1 = 使能 TXBEIF (EMAC_INTSTS[24]) 触发 TX 中断。
[23]	TDUIEN 发送描述符不可用中断使能位 TDUIEN 控制着 TDUIF (EMAC_INTSTS[23]) 中断的发生。如果 TDUIF 置位，且 TDUIEN 和 TXIEN (EMAC_INTEN[16]) 都使能了，EMAC 将触发 TX 中断到 CPU。如果 TDUIEN 或 TXIEN (EMAC_INTEN[16]) 关闭了，即使 TDUIF (EMAC_INTSTS[23]) 置位也不会触发 TX 中断。 0 = 禁止 TDUIF (EMAC_INTSTS[23]) 触发 TX 中断。 1 = 使能 TDUIF (EMAC_INTSTS[23]) 触发 TX 中断。

[22]	<b>LCIEN</b>	<b>迟到冲突中断使能位</b> LCIEN 控制着 LCIF (EMAC_INTSTS[22]) 中断的发生。如果 LCIF (EMAC_INTSTS[22]) 置位，且 LCIEN 和 TXIEN (EMAC_INTEN[16]) 都使能了，EMAC 将触发 TX 中断到 CPU。如果 TSALMIEN 或 TXIEN (EMAC_INTEN[16]) 关闭了，即使 LCIEN (EMAC_INTSTS[22]) 置位也不会触发 TX 中断。 0 = 禁止 LCIEN (EMAC_INTSTS[22]) 触发 TX 中断。 1 = 使能 LCIEN (EMAC_INTSTS[22]) 触发 TX 中断。
[21]	<b>TXABTIEN</b>	<b>发送停止中断使能位</b> TXABTIEN 控制着 TXABTIF (EMAC_INTSTS[21]) 中断的发生。如果 TXABTIF (EMAC_INTSTS[21]) 置位，且 TXABTIEN 和 TXIEN (EMAC_INTEN[16]) 都使能了，EMAC 将触发 TX 中断到 CPU。如果 TXABTIEN 或 TXIEN (EMAC_INTEN[16]) 关闭了，即使 TXABTIF (EMAC_INTSTS[21]) 置位也不会触发 TX 中断。 0 = 禁止 TXABTIEN (EMAC_INTSTS[21]) 触发 TX 中断。 1 = 使能 TXABTIEN (EMAC_INTSTS[21]) 触发 TX 中断。
[20]	<b>NCSIEN</b>	<b>无载波中断使能位</b> NCSIEN 控制着 NCSIF (EMAC_INTSTS[20]) 中断的发生。如果 NCSIF (EMAC_INTSTS[20]) 置位，且 NCSIEN 和 TXIEN (EMAC_INTEN[16]) 都使能了，EMAC 将触发 TX 中断到 CPU。如果 NCSIEN 或 TXIEN (EMAC_INTEN[16]) 关闭了，即使 NCSIF (EMAC_INTSTS[20]) 置位也不会触发 TX 中断。 0 = 禁止 NCSIEN (EMAC_INTSTS[20]) 触发 TX 中断。 1 = 使能 NCSIEN (EMAC_INTSTS[20]) 触发 TX 中断。
[19]	<b>EXDEFIEN</b>	<b>延期超时中断使能位</b> EXDEFIEN 控制着 EXDEFIF (EMAC_INTSTS[19]) 中断的发生。如果 EXDEFIF (EMAC_INTSTS[19]) 置位，且 EXDEFIEN 和 TXIEN (EMAC_INTEN[16]) 都使能了，EMAC 将触发 TX 中断到 CPU。如果 EXDEFIEN 或 TXIEN (EMAC_INTEN[16]) 关闭了，即使 EXDEFIF (EMAC_INTSTS[19]) 置位也不会触发 TX 中断。 0 = 禁止 EXDEFIEN (EMAC_INTSTS[19]) 触发 TX 中断。 1 = 使能 EXDEFIEN (EMAC_INTSTS[19]) 触发 TX 中断。
[18]	<b>TXCPIEN</b>	<b>发送完成中断使能位</b> TXCPIEN 控制着 TXCPIF (EMAC_INTSTS[18]) 中断的发生。如果 TXCPIF (EMAC_INTSTS[18]) 置位，且 TXCPIEN 和 TXIEN (EMAC_INTEN[16]) 都使能了，EMAC 将触发 TX 中断到 CPU。如果 TXCPIEN 或 TXIEN (EMAC_INTEN[16]) 关闭了，即使 TXCPIF (EMAC_INTSTS[18]) 置位也不会触发 TX 中断。 0 = 禁止 TXCPIEN (EMAC_INTSTS[18]) 触发 TX 中断。 1 = 使能 TXCPIEN (EMAC_INTSTS[18]) 触发 TX 中断。
[17]	<b>TXUDIEN</b>	<b>发送 FIFO 下溢使能位</b> TXUDIEN 控制着 TXUDIF (EMAC_INTSTS[17]) 中断的发生。如果 TXUDIF (EMAC_INTSTS[17]) 置位，且 TXUDIEN 和 TXIEN (EMAC_INTEN[16]) 都使能了，EMAC 将触发 TX 中断到 CPU。如果 TXUDIEN 或 TXIEN (EMAC_INTEN[16]) 关闭了，即使 TXUDIF (EMAC_INTSTS[17]) 置位也不会触发 TX 中断。 0 = 禁止 TXUDIEN (EMAC_INTSTS[17]) 触发 TX 中断。 1 = 使能 TXUDIEN (EMAC_INTSTS[17]) 触发 TX 中断。

[16]	<b>TXIEN</b>	<b>发送中断使能位</b> TXIEN 控制着 TX 中断的发生 如果使能了 TXIEN 且 TXIF (EMAC_INTSTS[16]) 置位，EMAC 将产生 TX 中断到 CPU。如果 TXIEN 未使能，即使 EMAC_INTSTS[24:17] 中某一位置且对应的 EMAC_INTEN 已使能，都不会产生 TX 中断。也就是说，软件如果想要使能 TX 中断，必须要先使能该位。想要关闭 TX 中断时，禁止该位即可。 0 = TXIF (EMAC_INTSTS[16]) 被屏蔽，禁止 TX 中断的发生 1 = TXIF (EMAC_INTSTS[16]) 未屏蔽，允许 TX 中断的发生
[15]	<b>WOLIEN</b>	<b>LAN 口唤醒中断使能位</b> WOLIEN 控制着 WOLIF (EMAC_INTSTS[15]) 中断的发生。如果 WOLIF (EMAC_INTSTS[15]) 置位，且 WOLIEN 和 RXIEN (EMAC_INTEN[0]) 都使能了，EMAC 将触发 RX 中断到 CPU。如果 WOLIEN 或 RXIEN (EMAC_INTEN[0]) 关闭了，即使 WOLIF (EMAC_INTSTS[15]) 置位也不会触发 RX 中断。 0 = 禁止 WOLIF (EMAC_INTSTS[15]) 触发 RX 中断。 1 = 使能 WOLIF (EMAC_INTSTS[15]) 触发 RX 中断。
[14]	<b>CFRIEN</b>	<b>控制帧接收中断使能位</b> CFRIEN 控制着 CFRIF (EMAC_INTSTS[14]) 中断的发生。如果 CFRIF (EMAC_INTSTS[14]) 置位，且 CFRIEN 和 RXIEN (EMAC_INTEN[0]) 都使能了，EMAC 将触发 RX 中断到 CPU。如果 CFRIEN 或 RXIEN (EMAC_INTEN[0]) 关闭了，即使 CFRIF (EMAC_INTSTS[14]) 置位也不会触发 RX 中断。 0 = 禁止 CFRIF (EMAC_INTSTS[14]) 触发 RX 中断。 1 = 使能 CFRIF (EMAC_INTSTS[14]) 触发 RX 中断。
[13:12]	<b>Reserved</b>	保留
[11]	<b>RXBEIEN</b>	<b>接收总线错误中断使能位</b> RXBEIEN 控制着 RXBEIF (EMAC_INTSTS[11]) 中断的发生。如果 RXBEIF (EMAC_INTSTS[11]) 置位，且 RXBEIEN 和 RXIEN (EMAC_INTEN[0]) 都使能了，EMAC 将触发 RX 中断到 CPU。如果 RXBEIEN 或 RXIEN (EMAC_INTEN[0]) 关闭了，即使 RXBEIF (EMAC_INTSTS[11]) 置位也不会触发 RX 中断。 0 = 禁止 RXBEIF (EMAC_INTSTS[11]) 触发 RX 中断。 1 = 使能 RXBEIF (EMAC_INTSTS[11]) 触发 RX 中断。
[10]	<b>RDUIEN</b>	<b>接收描述符不可用中断使能位</b> RDUIEN 控制着 RDUIF (EMAC_INTSTS[10]) 中断的发生。如果 RDUIF (EMAC_INTSTS[10]) 置位，且 RDUIEN 和 RXIEN (EMAC_INTEN[0]) 都使能了，EMAC 将触发 RX 中断到 CPU。如果 RDUIEN 或 RXIEN (EMAC_INTEN[0]) 关闭了，即使 RDUIF (EMAC_MIOSTA[10]) 置位也不会触发 RX 中断。 0 = 禁止 RDUIF (EMAC_MIOSTA[10]) 触发 RX 中断。 1 = 使能 RDUIF (EMAC_MIOSTA[10]) 触发 RX 中断。
[9]	<b>DENIEN</b>	<b>DMA 早期通知中断使能位</b> DENIEN 控制着 DENIF (EMAC_INTSTS[9]) 中断的发生。如果 DENIF (EMAC_INTSTS[9]) 置位，且 DENIEN 和 RXIEN (EMAC_INTEN[0]) 都使能了，EMAC 将触发 RX 中断到 CPU。如果 DENIEN 或 RXIEN (EMAC_INTEN[0]) 关闭了，即使 DENIF (EMAC_INTSTS[9]) 置位也不会触发 RX 中断。 0 = 禁止 DENIF (EMAC_INTSTS[9]) 触发 RX 中断。 1 = 使能 DENIF (EMAC_INTSTS[9]) 触发 RX 中断。

[8]	<b>MFLEIEN</b>	<b>超出最大帧长度中断使能位</b> MFLEIEN 控制着 MFLEIF (EMAC_INTSTS[8]) 中断的发生。如果 MFLEIF (EMAC_INTSTS[8]) 置位，且 MFLEIEN 和 RXIEN (EMAC_INTEN[0]) 都使能了，EMAC 将触发 RX 中断到 CPU。如果 MFLEIEN 或 RXIEN (EMAC_INTEN[0]) 关闭了，即使 MFLEIF (EMAC_INTSTS[8]) 置位也不会触发 RX 中断。 0 = 禁止 MFLEIF (EMAC_INTSTS[8]) 触发 RX 中断。 1 = 使能 MFLEIF (EMAC_INTSTS[8]) 触发 RX 中断。
[7]	<b>MPCOVIEN</b>	<b>丢包计数器溢出中断使能位</b> MPCOVIEN 控制着 MPCOVIF (EMAC_INTSTS[7]) 中断的发生。如果 MPCOVIF (EMAC_INTSTS[7]) 置位，且 MPCOVIEN 和 RXIEN (EMAC_INTEN[0]) 都使能了，EMAC 将触发 RX 中断到 CPU。如果 MPCOVIEN 或 RXIEN (EMAC_INTEN[0]) 关闭了，即使 MPCOVIF (EMAC_INTSTS[7]) 置位也不会触发 RX 中断。 0 = 禁止 MPCOVIF (EMAC_INTSTS[7]) 触发 RX 中断。 1 = 使能 MPCOVIF (EMAC_INTSTS[7]) 触发 RX 中断。
[6]	<b>RPIEN</b>	<b>短包中断使能位</b> RPIEN 控制着 RPIF (EMAC_INTSTS[6]) 中断的发生。RPIF (EMAC_INTSTS[6]) 置位，且 RPIEN 和 RXIEN (EMAC_INTEN[0]) 都使能了，EMAC 将触发 RX 中断到 CPU。如果 RPIEN 或 RXIEN (EMAC_INTEN[0]) 关闭了，即使 RPIF (EMAC_INTSTS[6]) 置位也不会触发 RX 中断。 0 = 禁止 RPIF (EMAC_INTSTS[6]) 触发 RX 中断。 1 = 使能 RPIF (EMAC_INTSTS[6]) 触发 RX 中断。
[5]	<b>ALIEIEN</b>	<b>对齐错误中断使能位</b> ALIEIEN 控制着 ALIEIF (EMAC_INTSTS[5]) 中断的发生。如果 ALIEIF (EMAC_INTSTS[5]) 置位，且 ALIEIEN 和 RXIEN (EMAC_INTEN[0]) 都使能了，EMAC 将触发 RX 中断到 CPU。如果 ALIEIEN 或 RXIEN (EMAC_INTEN[0]) 关闭了，即使 ALIEIF (EMAC_INTSTS[5]) 置位也不会触发 RX 中断。 0 = 禁止 ALIEIF (EMAC_INTSTS[5]) 触发 RX 中断。 1 = 使能 ALIEIF (EMAC_INTSTS[5]) 触发 RX 中断。
[4]	<b>RXGDIEN</b>	<b>接收正常中断使能位</b> RXGDIEN 控制着 RXGdif (EMAC_INTSTS[4]) 中断的发生。如果 RXGdif (EMAC_INTSTS[4]) 置位，且 RXGDIEN 和 RXIEN (EMAC_INTEN[0]) 都使能了，EMAC 将触发 RX 中断到 CPU。如果 RXGDIEN 或 RXIEN (EMAC_INTEN[0]) 关闭了，即使 RXGdif (EMAC_INTSTS[4]) 置位也不会触发 RX 中断。 0 = 禁止 RXGdif (EMAC_INTSTS[4]) 触发 RX 中断。 1 = 使能 RXGdif (EMAC_INTSTS[4]) 触发 RX 中断。
[3]	<b>LPIEN</b>	<b>长包中断使能位</b> LPIEN 控制着 LPIF (EMAC_INTSTS[3]) 中断的发生。如果 LPIF (EMAC_INTSTS[3]) 置位，且 LPIEN 和 RXIEN (EMAC_INTEN[0]) 都使能了，EMAC 将触发 RX 中断到 CPU。如果 LPIEN 或 RXIEN (EMAC_INTEN[0]) 关闭了，即使 LPIF (EMAC_INTSTS[3]) 置位也不会触发 RX 中断。 0 = 禁止 LPIF (EMAC_INTSTS[3]) 触发 RX 中断。 1 = 使能 LPIF (EMAC_INTSTS[3]) 触发 RX 中断。

[2]	<b>RXOVIEN</b>	<p><b>接收 FIFO 溢出中断使能位</b></p> <p>RXOVIF 控制着 RXOVIF (EMAC_INTSTS[2]) 中断的发生。如果 RXOVIF (EMAC_INTSTS[2]) 置位，且 RXOVIEN 和 RXIEN (EMAC_INTEN[0]) 都使能了，EMAC 将触发 RX 中断到 CPU。如果 RXOVIEN 或 RXIEN (EMAC_INTEN[0]) 关闭了，即使 RXOVIF (EMAC_INTSTS[2]) 置位也不会触发 RX 中断。</p> <p>0 = 禁止 RXOVIF (EMAC_INTSTS[2]) 触发 RX 中断。 1 = 使能 RXOVIF (EMAC_INTSTS[2]) 触发 RX 中断。</p>
[1]	<b>CRCEIEN</b>	<p><b>CRC 错误中断使能位</b></p> <p>CRCEIF 控制着 CRCEIF (EMAC_INTSTS[1]) 中断的发生。如果 CRCEIF (EMAC_INTSTS[1]) 置位，且 CRCEIEN 和 RXIEN (EMAC_INTEN[0]) 都使能了，EMAC 将触发 RX 中断到 CPU。如果 CRCEIEN 或 RXIEN (EMAC_INTEN[0]) 关闭了，即使 CRCEIF (EMAC_INTSTS[1]) 置位也不会触发 RX 中断。</p> <p>0 = 禁止 CRCEIF (EMAC_INTSTS[1]) 触发 RX 中断。 1 = 使能 CRCEIF (EMAC_INTSTS[1]) 触发 RX 中断。</p>
[0]	<b>RXIEN</b>	<p><b>接收中断使能位</b></p> <p>RXIEN 控制着 RX 中断的发生</p> <p>如果使能了 RXIEN 且 RXIF (EMAC_INTSTS[0]) 置位，EMAC 将产生 RX 中断到 CPU。如果 RXIEN 未使能，即使 EMAC_INTSTS[15:1] 中某一位置且对应的 EMAC_INTEN 已使能，都不会产生 RX 中断。也就是说，软件如果想要使能 RX 中断，必须要先使能该位。想要关闭 RX 中断时，禁止该位即可。</p> <p>0 = RXIF (EMAC_INTSTS[0]) 被屏蔽，禁止 RX 中断的发生 1 = RXIF (EMAC_INTSTS[0]) 未屏蔽，允许 RX 中断的发生</p>

**EMAC\_INTSTS MAC 中断状态寄存器**

EMAC\_INTSTS 保存 EMAC 的状态，如帧发送/接收状态和内部 FIFO 状态。EMAC\_INTSTS 的状态将会触发 TX 或 RX 中断。EMAC\_INTSTS 是写1清 0 的寄存器，写 1 到对应位即可清 0 对应状态和中断。

寄存器	偏移量	R/W	描述	复位值
EMAC_INTSTS	EMAC_BA+0x0B0	R/W	MAC 中断状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved			TSALMIF	Reserved			TXBEIF
23	22	21	20	19	18	17	16
TDUIF	LCIF	TXABTIF	NCSIF	EXDEFIF	TXCPIF	TXUDIF	TXIF
15	14	13	12	11	10	9	8
WOLIF	CFRIF	Reserved		RXBEIF	RDUIF	DENIF	MFLEIF
7	6	5	4	3	2	1	0
MPCOVIF	RPIF	ALIEIF	RXGDIF	LPIF	RXOVIF	CRCEIF	RXIF

位	描述	
[31:29]	Reserved	保留
[28]	TSALMIF	<b>时间戳报警中断</b> TSALMIF 置位表示 EMAC_TSSEC 寄存器的值等与 EMAC_ALMSEC 寄存器的值，EMAC_TSSUBSEC 寄存器的值等与 EMAC_TSMLSR 寄存器的值 如果 TSALMIF 置位，且 TSALMIEN (EMAC_INTEN[28]) 使能，TXIF 将为高 写 1 到该位，清除 TSALMIF 的状态。 0 = EMAC_TSSEC 寄存器的值不等与 EMAC_ALMSEC 寄存器的值，或 EMAC_TSSUBSEC 寄存器的值不等与 EMAC_TSMLSR 寄存器的值 1 = EMAC_TSSEC 寄存器的值等与 EMAC_ALMSEC 寄存器的值，且 EMAC_TSSUBSEC 寄存器的值等与 EMAC_TSMLSR 寄存器的值
[27:25]	Reserved	保留
[24]	TXBEIF	<b>发送总线错误中断</b> TXBEIF 置位表示在包发送的过程中，EMAC 通过 TXDMA 读取系统内存的时候，内存控制器返回了错误的响应。TXBEIF 状态为高时，建议复位 EMAC。 如果 TXBEIEN (EMAC_INTEN[24]) 使能，TXIF 将为高 写 1 到该位，清除 TXBEIF 的状态。 0 = 无错误响应 1 = 接收到错误的响应

[23]	<b>TDUIF</b>	<b>发送描述符不可用中断</b> TDUIF 置位表示包传输过程需要的 TX 描述符不可用，TXDMA 处于关闭状态。一旦 TXDMA 进入了关闭状态，软件必须在获得了可用 TX 描述符后写一条命令到 TSDR 寄存器使 TXDMA 离开关闭状态。 如果 TDUIF 置位，且 TDUIEN (EMAC_INTEN[23]) 使能，TXIF 将为高 写 1 到该位，清除 TDUIF 的状态。 0 = TX 描述符可用 1 = TX 描述符不可用
[22]	<b>LCIF</b>	<b>迟到冲突中断</b> LCIF 置位表示在 64 字节的冲突窗口外发生了新的冲突。即，在 64 字节的帧发送出去后产生了冲突。迟到冲突检测仅在 EMAC 半双工模式下有效。 如果 LCIF 置位，且 LCIEN (EMAC_INTEN[22]) 使能，TXIF 将为高 写 1 到该位，清除 LCIF 的状态。 0 = 在 64 字节的冲突窗口外无新的冲突 1 = 在 64 字节的冲突窗口外发生了新的冲突
[21]	<b>TXABTIF</b>	<b>发送停止中断</b> TXABTIF 置位表示在传输过程中发生了连续的 16 个冲突，接着传输过程会因此停止。该功能仅在 EMAC 半双工模式下有效 如果 TXABTIEN (EMAC_INTEN[21]) 使能，TXIF 将为高 写 1 到该位，清除 TXABTIF 的状态。 0 = 在传输过程中没有发生连续的 16 个冲突 1 = 在传输过程中发生了连续的 16 个冲突
[20]	<b>NCSIF</b>	<b>无载波中断</b> NCSIF 置高表示 MII I/F 信号 CRS 在包发送的开始不正常。该功能仅在 EMAC 半双工模式下有效 如果 NCSIEN (EMAC_INTEN[20]) 使能，TXIF 将为高 写 1 到该位，清除 NCSIF 的状态。 0 = CRS 信号正常 1 = CRS 信号在包发送的开始不正常
[19]	<b>EXDEFIF</b>	<b>延期超时中断</b> EXDEFIF 置位表示在 100Mbps 时，发送时帧等待发送超过了 0.32768ms. 或者在 10 Mbps 时，等待超过 3.2768ms. 延期超时检测仅在 MCMDR 寄存器位 NODEF (EMAC_CTL[9]) 关闭，EMAC 工作在半双工模式下有效。 如果 EXDEFIEN (EMAC_INTEN[19]) 使能，TXIF 将为高 写 1 到该位，清除 EXDEFIF 的状态。 0 = 未发生延期超时 1 = 在 100Mbps 时，发送时帧等待发送超过了 0.32768ms. 或者在 10 Mbps 时，等待超过 3.2768ms.
[18]	<b>TXCPIF</b>	<b>发送完成中断</b> TXCPIF 置位表示发送正确完成 如果 TXCPIEN (EMAC_INTEN[18]) 使能，TXIF 将为高 写 1 到该位，清除 TXCPIF 的状态。 0 = 发送未完成. 1 = 发送已完成

[17]	<b>TXUDIF</b>	<b>发送 FIFO 下溢中断</b> TXUDIF 置位表示发送过程中发生了发送 FIFO 下溢。当 TXFIFO 下溢发生，EMAC 会自动重新发送包而不需要软件干预。如果 TXFIFO 频繁发生，建议修改 TXFIFO 的阈值控制寄存器 FFTCR 的 TXFIFOTH 位到更高的级别 如果 TXUDIF 置位，且 TXUDIEN (EMAC_INTEN[17]) 使能，TXIF 将为高写 1 到该位，清除 TXUDIF 的状态。 0 = 传输过程中未发生 TXFIFO 下溢事件 1 = 传输过程中发生了 TXFIFO 下溢事件
[16]	<b>TXIF</b>	<b>发送中断</b> TXIF 代表 TX 中断的状态 如果 TXIF 置位且对应的使能位 TXIEN (EMAC_INTEN[16]) 也同时置位，表示 EMAC 产生了一个 TX 中断到 CPU。如果 TXIF 置位但 TXIEN (EMAC_INTEN[16]) 没有，则不会有中断发生 TXIF 的值是 EMAC_INTSTS[28:17] 和 EMAC_INTEN[28:17] 的值逻辑“与”后的各个位做逻辑“或”的结果。即，如果 EMAC_INTSTS[28:17] 某一位为高且 EMAC_INTEN[28:17] 对应的使能位也使能了，TXIF 就会置位。由于 TXIF 是逻辑“或”的结果，清 EMAC_INTSTS[28:17] 会清 0 TXIF。 0 = EMAC_INTSTS[28:17] 所有位都未置位，EMAC_INTEN[28:17] 所有位都未使能 1 = EMAC_INTSTS[28:17] 至少有一位置位且对应的 EMAC_INTEN[28:17] 位使能了
[15]	<b>WOLIF</b>	<b>LAN 口唤醒中断</b> WOLIF 置位表示 EMAC 收到了魔术封包。该位仅在系统处于掉电模式下且 WOLEN 置位时有效 如果 WOLIF 置位，且 WOLIEN (EMAC_INTEN[15]) 使能，RXIF 将为高写 1 到该位，清除 WOLIF 的状态。 0 = EMAC 未收到魔术封包 1 = EMAC 收到了魔术封包
[14]	<b>CFRIF</b>	<b>控制帧接收中断</b> CFRIF 置位表示 EMAC 收到了一个流控帧。该位仅在全双工模式下有效 如果 CFRIF 置位，且 CFRIEN (EMAC_INTEN[14]) 使能，RXIF 将为高写 1 到该位，清除 CFRIF 的状态。 0 = EMAC 未收到流控帧 1 = EMAC 收到了流控帧
[13:12]	<b>Reserved</b>	保留
[11]	<b>RXBEIF</b>	<b>接收总线错误中断</b> RXBEIF 置位表示在包发送的过程中，EMAC 通过 RXDMA 读取系统内存的时候，内存控制器返回了错误的响应。RXBEIF 状态为高时，建议复位 EMAC。 如果 RXBEIF 置位，且 RXBEIEN (EMAC_INTEN[11]) 使能，RXIF 将为高写 1 到该位，清除 RXBEIF 的状态。 0 = 无错误响应 1 = 接收到错误的响应

[10]	<b>RDUIF</b>	<p><b>接收描述符不可用中断</b></p> <p>RDUIF 置位表示包接收过程需要的 RX 描述符不可用，RXDMA 处于关闭状态。一旦 RXDMA 进入了关闭状态，软件必须在获得了可用 RX 描述符后写一条命令到 RSDR 寄存器使 RXDMA 离开关闭状态。</p> <p>如果 RDUIF 置位，且 RDUIEN (EMAC_INTEN[10]) 使能，RXIF 将为高写 1 到该位，清除 RDUIF 的状态。</p> <p>0 = RX 描述符可用 1 = RX 描述符不可用</p>
[9]	<b>DENIF</b>	<p><b>DMA 早期通知中断</b></p> <p>DENIF 置位表示 EMAC 收到了输入包的长度字段</p> <p>如果 DENIF 置位，且 DENIENI (EMAC_INTEN[9]) 使能，RXIF 将为高写 1 到该位，清除 DENIF 的状态。</p> <p>0 = EMAC 未收到输入包的长度字段 1 = EMAC 收到了输入包的长度字段</p>
[8]	<b>MFLEIF</b>	<p><b>超出最大帧长度中断</b></p> <p>MFLEIF 置位表示输入包的长度超过了 DMARFC 寄存器里配置的长度，该包会被丢弃</p> <p>如果 MFLEIF 置位，且 MFLEIEN (EMAC_INTEN[8]) 使能，RXIF 将为高写 1 到该位，清除 MFLEIF 的状态。</p> <p>0 = 输入包的长度未超过 DMARFC 寄存器里配置的长度 1 = 输入包的长度超过了 DMARFC 寄存器里配置的长度</p>
[7]	<b>MPCOVIF</b>	<p><b>丢包计数器溢出中断</b></p> <p>MPCOVIF 置位表示丢包计数器 MPCNT 溢出。</p> <p>如果 MPCOVIEN (EMAC_INTEN[7]) 使能，RXIF 将为高写 1 到该位，清除 MPCOVIF 的状态。</p> <p>0 = 丢包计数器 MPCNT 未溢出。 1 = 丢包计数器 MPCNT 溢出</p>
[6]	<b>RPIF</b>	<p><b>短包中断</b></p> <p>RPIF 置位表示会丢弃长度少于 64 字节的输入包。如果 ARP (EMAC_CTL[2]) 置位，短包会被视作正常包，RPIF 亦不会置位</p> <p>如果 RPIF 置位，且 RPIVEN (EMAC_INTEN[6]) 使能，RXIF 将为高写 1 到该位，清除 RPIF 的状态。</p> <p>0 = 输入包为短包，或软件希望接收短包。 1 = 丢弃为短包的输入包</p>
[5]	<b>ALIEIF</b>	<p><b>对齐错误中断</b></p> <p>ALIEIF 置位表示输入帧的长度不是整字节的</p> <p>如果 ALIEIEN (EMAC_INTEN[5]) 使能，RXIF 将为高写 1 到该位，清除 ALIEIF 的状态。</p> <p>0 = 输入帧的长度是整字节的 1 = 输入帧的长度不是整字节的</p>

[4]	<b>RXGDIF</b>	<p><b>接收正常中断</b></p> <p>RXGDIF 置位表示帧正常接收</p> <p>如果 RXGDIF 置位, 且 RXGDIEN (EMAC_MIEN[4]) 使能, RXIF 将为高写 1 到该位, 清除 RXGDIF 的状态.</p> <p>0 = 帧接收不正常 1 = 帧正常接收</p>
[3]	<b>LPIF</b>	<p><b>长包中断</b></p> <p>LPIF 置位表示丢弃输入包的长度超过 1518 个字节的长包。如果 ALP (EMAC_CTL[1]) 置位, 长包将被视作正常包, LPIF 亦不会置位。</p> <p>如果 LPIF 置位, 且 LPIEN (EMAC_INTEN[3]) 使能, RXIF 将为高写 1 到该位, 清除 LPIF 的状态.</p> <p>0 = 输入包为长包, 或软件希望接收长包. 1 = 丢弃为长包的输入包</p>
[2]	<b>RXOVIF</b>	<p><b>接收 FIFO 溢出中断</b></p> <p>RXOVIF 置位表示 RXFIFO 在包的接收过程中发生了溢出。当 RXFIFO 溢出发生, EMAC 会丢弃当前接收到的包。如果 RXFIFO 频繁溢出, 建议修改 RXFIFO 的阈值控制寄存器 FFTCR 的 RXFIFOTH 位到更高的级别</p> <p>如果 RXOVIF 置位, 且 RXOVIEN (EMAC_INTEN[2]) 使能, RXIF 将为高写 1 到该位, 清除 RXOVIF 的状态.</p> <p>0 = RXFIFO 在包的接收过程中未发生溢出 1 = RXFIFO 在包的接收过程中发生了溢出</p>
[1]	<b>CRCEIF</b>	<p><b>CRC 错误中断</b></p> <p>CRCEIF 置位表示输入包有 CRC 错误, 封包将会被丢弃。如果 AEP (EMAC_CTL[4]) 置位, CRC 错误的包将被视作正常包, CRCEIF 亦不会置位。</p> <p>如果 CRCEIF 置位, 且 CRCEIEN (EMAC_INTEN[1]) 使能, RXIF 将为高写 1 到该位, 清除 CRCEIF 的状态.</p> <p>0 = 输入包有 CRC 正常 1 = 输入包有 CRC 错误</p>
[0]	<b>RXIF</b>	<p><b>接收中断</b></p> <p>RXIF 代表 RX 中断的状态</p> <p>如果 RXIF 置位且对应的使能位 RXIEN (EMAC_INTEN[0]) 也同时置位, 表示 EMAC 产生了一个 RX 中断到 CPU。如果 RXIF 置位但 RXIEN (EMAC_INTEN[0]) 没有, 则不会有中断发生</p> <p>RXIF 的值是 EMAC_INTSTS[15:1] 和 EMAC_INTEN[15:1] 的值逻辑“与”后的各个位做逻辑“或”的结果。即, 如果 EMAC_INTSTS[15:1] 某一位为高且 EMAC_INTEN[15:1] 对应的使能位也使能了, RXIF 就会置位。由于 RXIF 是逻辑“或”的结果, 清 EMAC_INTSTS[15:1] 会清 0 RXIF。</p> <p>0 = EMAC_INTSTS[15:1] 所有位都未置位, EMAC_INTEN[15:1] 所有位都未使能 1 = EMAC_INTSTS[15:1] 至少有一位置位且对应的 EMAC_INTEN[15:1] 位使能了</p>

**EMAC\_GENSTS MAC 通用状态寄存器**

EMAC\_GENSTS 存储着 EMAC 的状态。但是存储在 EMAC\_GENSTS 的状态并不会触发中断。EMAC\_GENSTS 寄存器写清 0, 写 1 到对应位会清除其状态

寄存器	偏移量	R/W	描述	复位值
EMAC_GENSTS	EMAC_BA+0x0B4	R/W	MAC Ge 通用状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved			RPSTS	TXHALT	SQE	TXPAUSED	DEF
7	6	5	4	3	2	1	0
COLCNT				Reserved	RXFFULL	RXHALT	CFR

位	描述	
[31:13]	Reserved	保留
[12]	RPSTS	<b>远程暂停状态</b> RPSTS 表示远程暂停计数器开始向下计数。 在 EMAC 控制器发送暂停帧成功后，远程暂停计数器开始向下计数。当该位置位，预示着 EMAC 控制器在向下计数完成前都不会在进行包传输。 0 = 远程暂停计数器计数完成 1 = 远程暂停计数器向下计数中
[11]	TXHALT	<b>发送停止</b> TXHALT 置位表示由于软件禁止了 TXON (EMAC_CTL[8]) 位，下一个正常包的发送过程将被停止。 0 = 下一个正常包的发送过程正常进行 1 = 下一个正常包的发送过程将被停止
[10]	SQE	<b>信号质量错误 (SQE)</b> SQE 置位表示在 10 Mbps 半双工模式下包传送结尾发现 SQE 错误。EMAC 工作在 10Mbps 半双工模式下时，SQECHKEN (EMAC_CTL[17]) 使能，则会进行 SQE 错误检测。 0 = 包发送结尾未发现 SQE 错误。 1 = 包发送结尾发现 SQE 错误。
[9]	TXPAUSED	<b>发送暂停</b> TXPAUSED 置位表示 EMAC 接收到暂停帧下个正常的包发送过程会被暂停。 0 = 下一个正常的包发送正常。 1 = 下一个正常的包发送被暂停。

[8]	<b>DEF</b>	<b>发送延期</b> DEF 置位表示发送延期一次。DEF 检测仅在半双工模式下有效 0 = 包发送未延期 1 = 包发送延期
[7:4]	<b>COLCNT</b>	<b>冲突数</b> COLCNT 表示在包传递过程中发生的连续冲突数。如果传输时连续冲突数达 16 个，COLCNT 归 0x0, TXABTIF 位置位。
[3]	<b>Reserved</b>	保留
[2]	<b>RXFFULL</b>	<b>RXFIFO 满</b> RXFFULL 表示 4 个 64 字节的包存储在 RXFIFO 中导致其满了，之后的输入包将会被丢弃 0 = The RXFIFO 未满 1 = The RXFIFO 满了，之后的输入包将会被丢弃
[1]	<b>RXHALT</b>	<b>接收停止</b> RXHALT 置位表示由于软件禁止了寄存器 MCMDR 的 RXON 位，下一个正常的包接收过程会被停止 0 = 下一个正常的包接收过程正常进行 1 = 下一个正常的包接收过程会被停止
[0]	<b>CFR</b>	<b>控制帧接收</b> CFRIF 置位表示 EMAC 接收到了一个流控帧。该位仅在全双工模式下有效 0 = EMAC 未接收到流控帧 1 = EMAC 接收到了流控帧

**EMAC MPCNT 丢包计数寄存器**

EMAC\_MPCNT 保存由于各种接收错误而产生的丢包的个数。EMAC\_MPCNT 是一个读清除的寄存器。另外，软件可以写初始值到 EMAC\_MPCNT，丢包计数器会从设置的值开始计数。如果丢包计数器溢出，MPCOVIF (EMAC\_INTSTS[7]) 会被置位

寄存器	偏移量	R/W	描述	复位值
EMAC_MPCNT	EMAC_BA+0x0B8	R/W	丢包计数寄存器	0x0000_7FFF

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
MPCNT							
7	6	5	4	3	2	1	0
MPCNT							

位	描述	
[31:16]	Reserved	保留
[15:0]	MPCNT	<b>丢包计数值</b> MPCNT 保存由于各种接收错误而产生的丢包的个数。以下错误都会使计数值增加： 1.输入包使 RXFIFO 溢出 2.由于 RXON 关闭而使输入包被丢弃 3.输入包发生 CRC 错误

**EMAC\_RPCNT MAC 接收到的暂停帧计数寄存器**

EMAC 支持暂停帧的接收与识别。如果 EMAC 接收到了一个暂停帧，暂停帧的 OPERAND 字段被解析存储到 EMAC\_RPCNT 寄存器。EMAC\_RPCNT 寄存器在 EMAC 暂停过程中一直保持不变。EMAC\_RPCNT 只读，写操作对该寄存器无效。

寄存器	偏移量	R/W	描述	复位值
EMAC_RPCNT	EMAC_BA+0x0BC	R	MAC 接收到的暂停帧计数寄存器	0x0000_0000

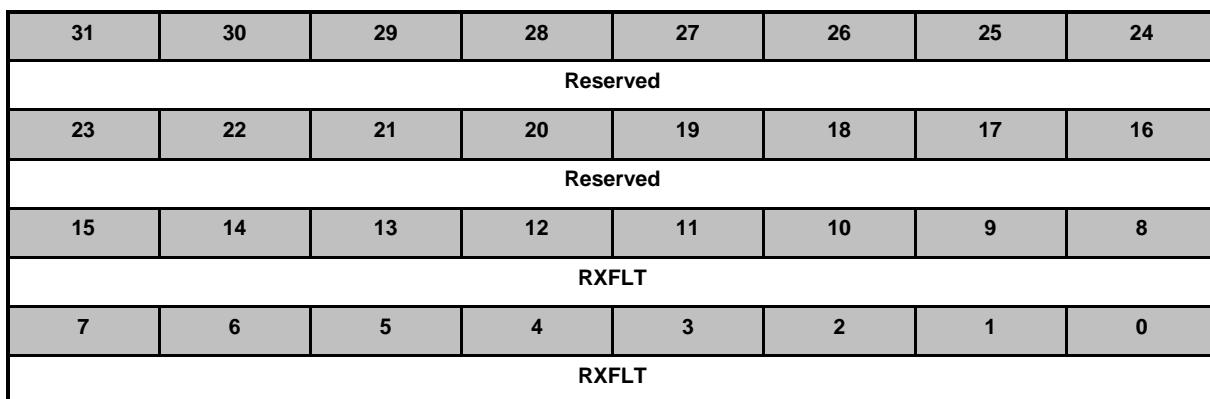
31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
RPCNT							
7	6	5	4	3	2	1	0
RPCNT							

位	描述	
[31:16]	Reserved	保留
[15:0]	RPCNT	MAC 暂停计数 RPCNT 保存暂停帧的 OPERAND 字段。即，EMAC 需要暂停多少个时间片 (512 位时间)

**EMAC\_FRSTS DMA 接收帧状态寄存器**

EMAC\_FRSTS 用来存储每个输入帧的长度字段

寄存器	偏移量	R/W	描述	复位值
EMAC_FRSTS	EMAC_BA+0x0C8	R/W	DMA 接收帧状态寄存器	0x0000_0000



位	描述	
[31:16]	Reserved	保留
[15:0]	RXFLT	<b>接收帧长度</b> RXFLT 用来存储每个输入帧的长度字段。如果使能了 DENIEN (EMAC_INTEN[9]) 且已收到输入帧的长度字段, DENIF (EMAC_INTSTS[9]) 会被置位, 并触发中断。RXFLT 会存储该长度字段。

**EMAC CTXDSA 当前发送描述符起始地址寄存器**

寄存器	偏移量	R/W	描述	复位值
EMAC_CTXDSA	EMAC_BA+0x0CC	R	当前发送描述符起始地址寄存器	0x0000_0000

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

位	描述	
[31:0]	CTXDSA	当前发送描述符起始地址 CTXDSA 保存 TXDMA 当前处理的 TX 描述符的起始地址。CTXDSQA 只读，写该寄存器无效

**EMAC CTXBSA 当前发送缓存起始地址寄存器**

寄存器	偏移量	R/W	描述	复位值
EMAC_CTXBSA	EMAC_BA+0x0D0	R	当前发送缓存起始地址寄存器	0x0000_0000

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

位	描述	
[31:0]	CTXBSA	当前发送缓存起始地址 CTXDSA 保存当前 TXDMA 发送缓存起始地址。CTXBSA 只读，写该寄存器无效

**EMAC CRXDSA 当前接收描述符起始地址寄存器**

寄存器	偏移量	R/W	描述	复位值
EMAC_CRXDSA	EMAC_BA+0x0D4	R	当前接收描述符起始地址寄存器	0x0000_0000

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

位	描述	
[31:0]	CRXDSA	当前接收描述符起始地址 CRXDSA 存储当前 RXDMA 接收描述符起始地址。CRXDSA 只读，写该寄存器无效

**EMAC CRXBSA 当前接收缓存起始地址寄存器**

寄存器	偏移量	R/W	描述	复位值
EMAC_CRXBSA	EMAC_BA+0x0D8	R	当前接收缓存起始地址寄存器	0x0000_0000

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

位	描述	
[31:0]	CRXBSA	当前接收缓存起始地址 CRXBSA 保存 RXDMA 当前接收缓存起始地址。CRXBSA 只读，写该寄存器无效。

**EMAC\_TSCTL 时间戳控制寄存器**

寄存器	偏移量	R/W	描述	复位值
EMAC_TSCTL	EMAC_BA+0x100	R/W	时间戳控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved		TSALMEN	Reserved	TSUPDATE	TSMODE	TSIEN	TSEN

位	描述	
[31:6]	Reserved	保留
[5]	TSALMEN	<p><b>时间戳报警使能位</b>  置位该位使能 EMAC 控制器在 EMAC_TSSEC 等于 EMAC_ALMSEC 和 EMAC_TSSUBSEC 等于 EMAC_ALMSUBSEC 时置位 TSALMIF (EMAC_INTSTS[28])。</p> <p>0 = EMAC_TSSEC 等于 EMAC_ALMSEC 和 EMAC_TSSUBSEC 等于 EMAC_ALMSUBSEC 时禁止报警。</p> <p>1 = EMAC_TSSEC 等于 EMAC_ALMSEC 和 EMAC_TSSUBSEC 等于 EMAC_ALMSUBSEC 时使能报警。</p>
[4]	Reserved	保留
[3]	TSUPDATE	<p><b>时间戳计数器时间更新使能位</b>  置位该位使能 EMAC 控制器增加寄存器 EMAC_UPDSEC 和 EMAC_UPDSUBSEC 的值到 PTP 时间戳计数器</p> <p>0 = 无效</p> <p>1 = EMAC_UPDSEC 更新到 EMAC_TSSEC , EMAC_UPDSUBSEC 更新到 EMAC_TSSUBSEC.</p>
[2]	TSMODE	<p><b>时间戳计数器精确更新使能位</b>  该位决定时间戳更新的模式</p> <p>0 = 时间戳计数器粗略更新模式</p> <p>1 = 时间戳计数器精确更新模式</p>
[1]	TSIEN	<p><b>时间戳计数器初始化使能位</b>  置位该位使能 EMAC 控制器加载寄存器 EMAC_UPDSEC 和 EMAC_UPDSUBSEC 的值到 PTP 时间戳计数器</p> <p>0 = 时间戳计数器初始化完成</p> <p>1 = 使能时间戳计数器初始化</p>

[0]	TSEN	<b>时间戳功能使能位</b> 该位控制是否使能符合 IEEE 1588 PTP 规范的时间戳功能 0 = 关闭符合 IEEE 1588 PTP 规范的时间戳功能 1 = 使能符合 IEEE 1588 PTP 规范的时间戳功能
-----	------	--

**EMAC\_TSSEC 时间戳秒计数器寄存器**

寄存器	偏移量	R/W	描述	复位值
EMAC_TSSEC	EMAC_BA+0x110	R	时间戳秒计数器寄存器	0x0000_0000

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

位	描述	
[31:0]	<b>SEC</b>	时间戳计数器秒寄存器 该寄存器为 64 位的参考时间计数器的 [63:32] 位。当 TSEN (EMAC_TSCTL[0]) 置位时，这 32 位是时间戳的秒部分。

## EMAC\_TSSUBSEC 时间戳亚秒计数器寄存器

寄存器	偏移量	R/W	描述	复位值
EMAC_TSSUBSEC	EMAC_BA+0x114	R	时间戳亚秒计数器寄存器	0x0000_0000

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

位	描述	
[31:0]	SUBSEC	时间戳计数器亚秒寄存器 该寄存器为 64 位的参考时间计数器的 [31:0] 位。当 TSEN (EMAC_TSCTL[0]) 置位时，这 32 位是时间戳的亚秒部分。

**EMAC\_TSINC** 时间戳增量寄存器

寄存器	偏移量	R/W	描述	复位值
EMAC_TSINC	EMAC_BA+0x118	R/W	时间戳增量寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
CNTINC							

位	描述	
[31:8]	Reserved	保留
[7:0]	CNTINC	<b>时间戳计数器增量</b> 时间戳计数器的增量 如果 TSEN (EMAC_TSCTL[0]) 置位, EMAC 增加 EMAC_TSSUBSEC 的值时, 每次增加这里的 8 位值

**EMAC\_TSADDEND 时间戳加数寄存器**

寄存器	偏移量	R/W	描述	复位值
EMAC_TSADDEND	EMAC_BA+0x11C	R/W	时间戳加数寄存器	0x0000_0000

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

位	描述	
[31:0]	ADDEND	<b>时间戳加数寄存器</b> 该寄存器 32 位的累加器的值，该累加器用来触发 EMAC_TSSUBSEC 的增加。 如果 TSEN (EMAC_TSCTL[0]) 和 TSMODE (EMAC_TSCTL[2]) 都为 1，EMAC 累加器在每个 HCLK 都增加这个值。一旦累加器溢出，将会触发 EMAC_TSSUBSEC 增加，增加的值为保存在 EMAC_TSINC 的 8 位值

**EMAC\_UPDSEC 时间戳更新秒寄存器**

寄存器	偏移量	R/W	描述	复位值
EMAC_UPDSEC	EMAC_BA+0x120	R/W	时间戳更新秒寄存器	0x0000_0000

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

位	描述	
[31:0]	<b>SEC</b>	时间戳更新秒寄存器 当 TSIEN (EMAC_TSCTL[1]) 为高， EMAC 直接加载该寄存器 32 位值到 EMAC_TSSEC。如果 TSUPDATE (EMAC_TSCTL[3]) 为高， EMAC 则把该寄存器的 32 位值累加到 EMAC_TSSEC 上

EMAC\_UPDSUBSEC 时间戳更新亚秒寄存器

寄存器	偏移量	R/W	描述	复位值
EMAC_UPDSUBSEC	EMAC_BA+0x124	R/W	时间戳更新亚秒寄存器	0x0000_0000

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

位	描述	
[31:0]	SUBSEC	<b>时间戳更新亚秒</b> 当 TSIEN (EMAC_TSCTL[1]) 为高, EMAC 直接加载该寄存器 32 位值到 EMAC_TSSUBSEC。如果 TSUPDATE (EMAC_TSCTL[3]) 为高, EMAC 则把该寄存器的 32 位值累加到 EMAC_TSSUBSEC 上

**EMAC\_ALMSEC 时间戳报警秒寄存器**

寄存器	偏移量	R/W	描述	复位值
EMAC_ALMSEC	EMAC_BA+0x128	R/W	时间戳报警秒寄存器	0x0000_0000

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

位	描述	
[31:0]	SEC	<b>时间戳报警秒寄存器</b> 时间戳计数器报警值的秒部分 该值仅在 ALMEN (EMAC_TSCTL[5]) 为高时有效。如果 ALMEN (EMAC_TSCTL[5]) 为高， EMAC_TSSEC 等于 EMAC_ALMSEC 和 EMAC_TSSUBSEC 等于 EMAC_ALMSUBSEC 时，EMAC 控制器会置位 TSALMIF (EMAC_INTSTS[28])

**EMAC\_ALMSUBSEC 时间戳报警亚秒寄存器**

寄存器	偏移量	R/W	描述	复位值
EMAC_ALMSUBSEC	EMAC_BA+0x12C	R/W	时间戳报警亚秒寄存器	0x0000_0000

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

位	描述
[31:0]	<b>SUBSEC</b> 时间戳报警亚秒寄存器 时间戳计数器报警值的亚秒部分 该值仅在 ALMEN (EMAC_TSCTL[5]) 为高时有效。如果 ALMEN (EMAC_TSCTL[5]) 为高， EMAC_TSSEC 等于 EMAC_ALMSEC 和 EMAC_TSSUBSEC 等于 EMAC_ALMSUBSEC 时，EMAC 控制器会置位 TSALMIF (EMAC_INTSTS[28])

## 6.17 智能卡主机接口

### 6.17.1 概述

智能卡接口控制器 (SC 控制器) 基于 ISO/IEC 7816-3 标准并完全兼容 PC/SC 规格。它还提供卡插入/移除的状态检测的功能。

### 6.17.2 特征

- 兼容 ISO-7816-3 T=0,T=1
- 兼容 EMV2000
- 一个 ISO-7816-3 端口
- 接收和发送各 4 字节 FIFO
- 可编程的发送时钟频率
- 可编程的接收器缓存触发水平
- 可编程的块保护时间选择 (11 ETU ~ 267 ETU)
- 一个 24-位和两个 8-位计数器用于请求应答 (Answer to Request (ATR)) 和等待时间处理
- 支持自动反向功能
- 支持传送器和接收器错误重试和错误数目限制功能
- 支持硬件激活序列处理，可配置上电到时钟产生的间隔
- 支持硬件热复位序列处理
- 支持硬件释放序列处理
- 支持当检测到卡移除时，硬件自动释放序列
- 支持 UART 模式
  - 全双工异步通信
  - 独立的接收和发送各 4 字节 FIFO
  - 支持可编程的波特率发生器
  - 支持可编程的接收器缓存触发水平
  - 通过设置 EGT (SC\_EGT[7:0])，可编程的发送数据延时时间 (最后一个停止位从 TX-FIFO 离开到释放的时间)。
  - 可编程的偶，奇或者无校验位的生成和检测
  - 可编程的停止位，1 位或 2 位的停止位生成

### 6.17.3 框图

SC 的时钟控制和框图如下图所示。SC 控制器包含 2 个完全异步的时钟区域，PCLK 和 引擎时钟。需要注意的是，PCLK 需要大于等于引擎时钟频率。

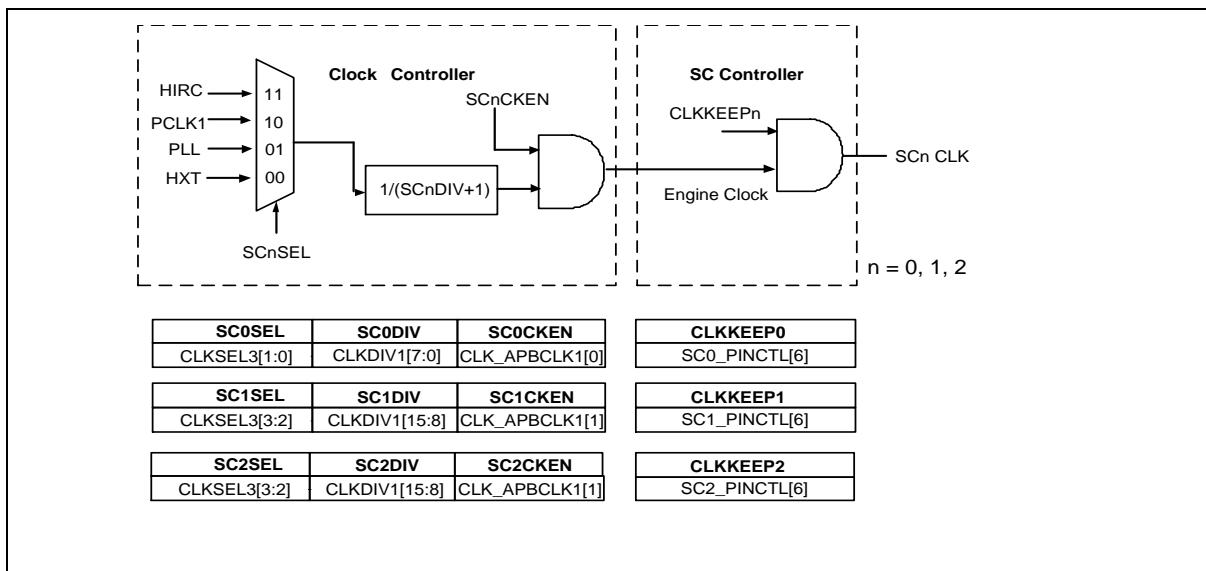


图 6.17-1 SC 时钟控制框图 (时钟控制器含 7 位的预分频计数器)

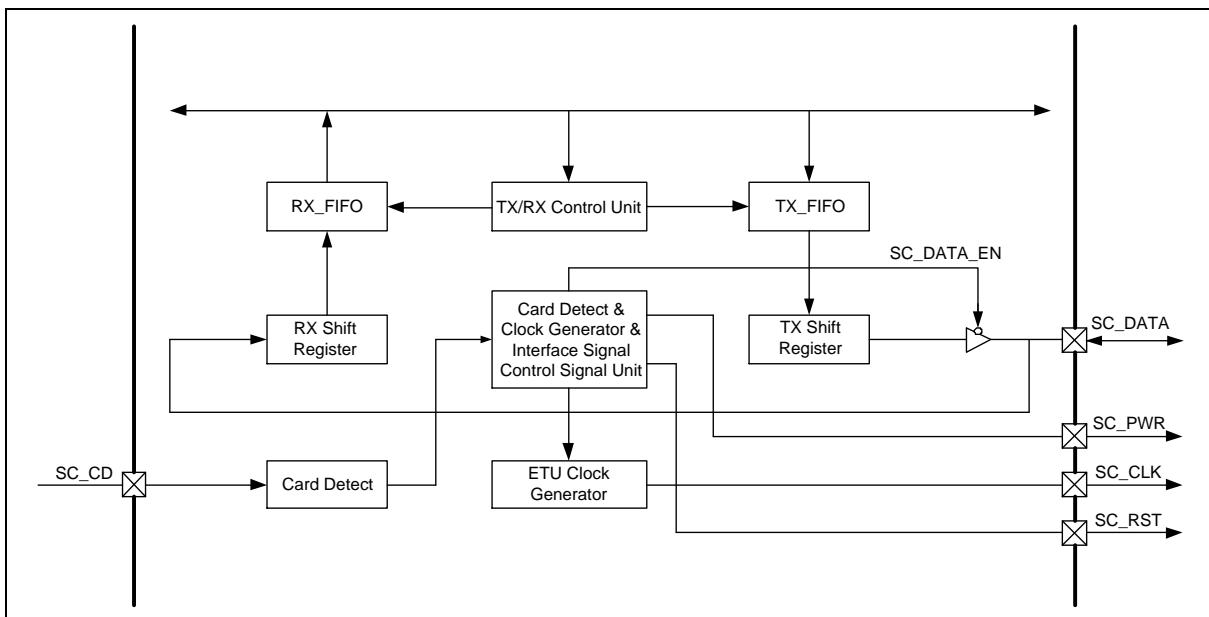


图 6.17-2 SC 控制器框图

#### 6.17.4 基本配置

SC 主控制器管脚描述如下所示：

管脚	类型	描述
SCn_DATA	双向的	SC 主控制器数据
SCn_CD	输入	SC 主控制器卡检测
SCn_PWR	输出	SC 主控制器卡电源开关
SCn_CLK	输出	SC 主控制器时钟
SCn_RST	输出	SC 主控制器复位

表 6.17-1 SC 主控制器管脚描述

UART 模式管脚描述如下所示：

管脚	类型	描述
SCn_DATA	输入	UART 接收数据
SCn_CLK	输出	UART 发送数据

表 6.17-2 UART 管脚描述

#### 6.17.4.1 SC0 基本配置

- 时钟源配置

- 在寄存器 SC0SEL (CLK\_CLKSEL3[1:0]) 配置 SC0 的外设时钟源
- 在寄存器 SC0DIV (CLK\_CLKDIV1[7:0]) 选择 SC0 外设时钟分频数
- 在寄存器 SC0CKEN (CLK\_APBCLK1[0]) 使能 SC0 的外设时钟

- 复位配置

- 在寄存器 SC0RST (SYS\_IPRST2[0]) 复位 SC0 控制器

- 管脚配置

组	管脚名	GPIO	MFP
SC0	SC0_CLK	PB.5	MFP9
		PF.6	MFP3
		PA.0	MFP6
		PE.2	MFP6
	SC0_DAT	PB.4	MFP9
		PF.7	MFP3
		PA.1	MFP6
		PE.3	MFP6
	SC0_PWR	PB.2	MFP9
		PF.9	MFP3
		PA.3	MFP6

		PE.5	MFP6
SC0_RST		PB.3	MFP9
		PF.8	MFP3
		PA.2	MFP6
		PE.4	MFP6
SC0_nCD		PC.12	MFP9
		PF.10	MFP3
		PA.4	MFP6
		PE.6	MFP6

#### 6.17.4.2 SC2 基本配置

- 时钟源配置
  - 在寄存器 SC2SEL (CLK\_CLKSEL3[5:4]) 配置 SC2 的外设时钟源
  - 在寄存器 SC2DIV (CLK\_CLKDIV1[23:16]) 选择 SC2 外设时钟分频数
  - 在寄存器 SC2CKEN (CLK\_APBCLK1[2]) 使能 SC2 的外设时钟
- 复位配置
  - 在寄存器 SC2RST (SYS\_IPRST2[2]) 复位 SC2 控制器

- 管脚配置

组	管脚名	GPIO	MFP
SC1	SC1_CLK	PC.0	MFP5
		PD.4	MFP8
		PG.8	MFP4
		PB.12	MFP3
	SC1_DAT	PC.1	MFP5
		PD.5	MFP8
		PG.7	MFP4
		PB.13	MFP3
	SC1_PWR	PC.3	MFP5
		PD.7	MFP8
		PG.5	MFP4
		PB.15	MFP3
	SC1_RST	PC.2	MFP5
		PD.6	MFP8
		PG.6	MFP4
		PB.14	MFP3
	SC1_nCD	PC.4	MFP5

		PD.3	MFP8
		PD.14	MFP4
		PC.14	MFP3

#### 6.17.4.3 SC2 基本配置

- 时钟源配置

- 在寄存器 SC2SEL (CLK\_CLKSEL3[5:4]) 配置 SC2 的外设时钟源
- 在寄存器 SC2DIV (CLK\_CLKDIV1[23:16]) 选择 SC2 外设时钟分频数
- 在寄存器 SC2CKEN (CLK\_APBCLK1[2]) 使能 SC2 的外设时钟

- 复位配置

- 在寄存器 SC2RST (SYS\_IPRST2[2]) 复位 SC2 控制器

- 管脚配置

组	管脚名	GPIO	MFP
SC2	SC2_CLK	PA.8	MFP3
		PA.6	MFP6
		PD.0	MFP7
		PA.15	MFP7
		PE.0	MFP4
	SC2_DAT	PA.9	MFP3
		PA.7	MFP6
		PD.1	MFP7
		PA.14	MFP7
		PE.1	MFP4
	SC2_PWR	PA.11	MFP3
		PC.7	MFP6
		PD.3	MFP7
		PA.12	MFP7
		PH.8	MFP4
	SC2_RST	PA.10	MFP3
		PC.6	MFP6
		PD.2	MFP7
		PA.13	MFP7
		PH.9	MFP4
	SC2_nCD	PC.13	MFP3
		PA.5	MFP6
		PD.13	MFP7

		PH.10	MFP4
--	--	-------	------

### 6.17.5 功能描述

智能卡接口扮演的就是一个半双工异步通信端口的角色，它的数据格式如下图所示的 10 个连续位组成

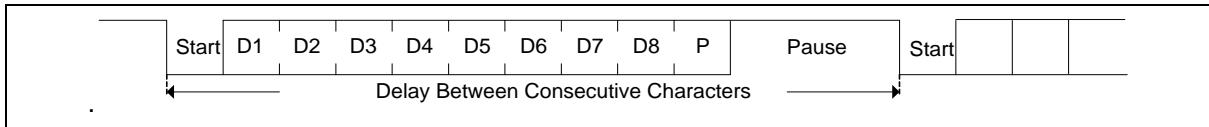


图 6.17-3 SC 数据格式

智能卡接口控制器支持硬件激活，热复位和释放序列。后面会对硬件激活，热复位和释放序列作详细说明

#### 6.17.5.1 智能卡管脚配置

智能卡接口的寄存器状态可以通过读取以下寄存器获得：

1. SCn\_RST 是一个输出管脚。读 RSTSTS (SCn\_PINCTL[18]) 获得对应状态。写 0 或 1 到 RSTEN (SCn\_PINCTL[1]) 可以拉低或拉高该管脚。
2. SCn\_PWR 是一个输出管脚。读 PWRSTS (SCn\_PINCTL[17]) 获得对应状态。写 0 或 1 到 PWREN (SCn\_PINCTL[0]) 可以拉低或拉高该管脚。PWRINV (SCn\_PINCTL[11]) 寄存器控制该管脚的反转功能。用户在智能卡使能前需要先设置 PWRINV (SCn\_PINCTL[11]) 寄存器)。
3. SCn\_DATA 是一个双向管脚。智能卡接收数据时，DATASTS (SCn\_PINCTL[16]) 存储对应的管脚状态。写 0 或 1 到 SCDATA (SCn\_PINCTL[9]) 可以拉低或拉高该管脚
4. SCn\_CLK 是一个输出管脚。它输出智能卡时钟 SCn CLK。写 0 或 1 到 CLKKEEP (SCn\_PINCTL[6]) 寄存器可以关闭或使能该管脚。CSTOPLV (SCn\_PINCTL[5]) 则决定了该管脚在时钟关闭输出时处于低电平还是高电平。
5. SCn\_CD (智能卡检测管脚) 表示当前卡片的插入状态。读 CDPINSTS (SC\_CDPINSTS[13])。寄存器获得当前 SCn\_CD 管脚的状态。SCn\_CD 相关的功能设置寄存器有 CDLV (SC\_CTL[26]), CDDBSEL (SC\_CTL[25:24]), CDIF (SC\_INTSTS[7]), CDIEN (SC\_INTEN[7])..
6. CDLV (SC\_CTL[26]) 决定了什么样的电平变化会被视作卡片插入。CDDBSEL (SC\_CTL[25:24]) 为消抖周期数。当卡片状态寄存器 CINSERT (SCn\_STATUS[12]) 或 CREMOVE (SCn\_STATUS[11]) 发生变化时，CDIF 会被置位。如果使能了 CDIEN，CDIF 置 1 时，SC 会产生中断到 CPU。建议在使能 SC 之前先配置该管脚。

#### 6.17.5.2 激活，热复位和释放序列

##### 激活

如图 6.17-4 是激活序列：

1. 写 0 到 RSTEN (SCn\_PINCTL[1])，设置 SCn\_RST 为低电平。然后，等待 SYNC (SCn\_PINCTL[31]) 清 0。.
2. 写 1 到 PWREN (SCn\_PINCTL[0]) 设置 SCn\_PWR 为高电平。写 1 到 SCDATA (SCn\_PINCTL[9]) 设置 SCn\_DATA 为高电平接收模式。等待 SYNC (SCn\_PINCTL[31]) 清 0.
3. 写 1 到 CLKKEEP (SCn\_PINCTL[6]) 使能 SCn\_CLK。等待 SYNC (SCn\_PINCTL[31])
4. 写 1 到 RSTEN (SCn\_PINCTL[1]) 释放 SCn\_RST 为高电平。等待 SYNC (SCn\_PINCTL[31]) 清 0

激活序列可以有两种方法控制，其过程如下所示：

- 软件控制时序

软件可以通过设置  $SC_n\_PINCTL$  和  $SC_n\_TMRCTLx$  ( $x = 0, 1, 2$ ) 来处理激活序列。通过设置  $SC_n\_PINCTL$ ，可以分别设置  $SC_n\_PWR, SC_n\_CLK, SC_n\_RST$  和  $SC_n\_DATA$  的管脚状态。编程方法在激活序列中有描述。激活序列时序可以通过设置  $SC_n\_TMRCTLx$  ( $x = 0, 1, 2$ ) 来控制。软件控制时序为用户在激活序列过程中提供了灵活的时序设置。

- 硬件时序控制：

软件设置 ACTEN ( $SC_n\_ALTCTL[3]$ ) 为 1，接口将通过硬件执行激活序列。 $SC_n\_PWR$  上电到  $SC_n\_CLK$  开始时间 (T1) 和  $SC_n\_CLK$  开始到  $SC_n\_RST$  复位的时间 (T2) 可以通过设置寄存器 INITSEL ( $SC\_ALTCTL[9:8]$ ) 来选择。 $SC_n\_PWR$  上电到开始时间  $SC_n\_CLK$  的长度可以通过寄存器 T1EXT ( $SC_n\_ACTCTL[4:0]$ ) 设置。硬件控制时序为用户激活序列提供了一种简单的设置。在硬件激活的过程中，RX 接收是关闭的无法接收任何数据。

如下是由硬件产生的激活控制序列：

1. 通过设置 INITSEL ( $SC_n\_ALTCTL[9:8]$ ) 来设置激活时序
2. 通过设置 TMRSEL ( $SC_n\_CTL[14:13]$ ) 为 '11'，来选择 Timer0。
3. 设置操作模式 OPMODE ( $SC_n\_TMRCTL0[27:24]$ ) 为 '0011'，通过设置 CNT ( $SC_n\_TMRCTL0[23:0]$ ) 来提供应答请求 (ATR) 的值。
4. 当硬件释放  $SC_n\_RST$  到高时，硬件将产生一个中断 INITIF ( $SC_n\_INTSTS[8]$ ) 到 CPU，同时 INITIEN ( $SC_n\_INTEN[8]$ ) 置为 1。
- 5 如果 Timer0 递减计数到 0 (从  $SC_n\_RST$  释放开始)，而且在此之前卡没有回应 ATR，硬件将产生 TMR0IF ( $SC_n\_INTSTS[3]$ ) 中断到 CPU。.

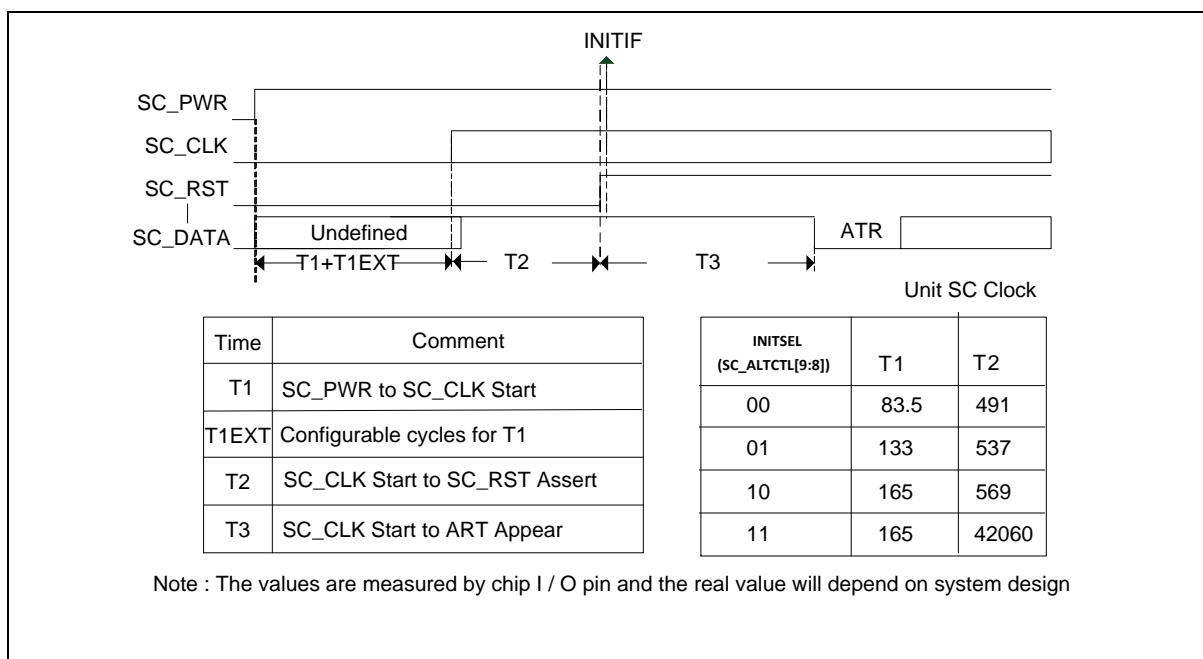


图 6.17-4 SC 激活序列

### 热复位

如图 6.17-5 是热复位示意：

1. 通过设置 RSTEN (SCn\_PINCTL[1]) 为 '0'，设置 SCn\_RST 为低。等待 SYNC (SCn\_PINCTL[31]) 清 0
2. 通过设置 SCDATA (SCn\_PINCTL[9]) 为 '1'，设置 SCn\_DAT 为高。等待 SYNC (SCn\_PINCTL[31]) 清 0
3. 通过设置 RSTEN (SCn\_PINCTL[1]) 为 '1'，设置 SCn\_RST 为高。等待 SYNC (SCn\_PINCTL[31]) 清 0

热复位序列可以有两种方法来控制，其过程如下所示。

- 软件时序控制：

软件通过设置 SCn\_PINCTL 和 SCn\_TMRCTLx (x=0,1,2) 来处理热复位序列。可以通过编程 SCn\_PINCTL 来设置 SCn\_RST 和 SCn\_DATA 的引脚状态。也可以通过设置 SCn\_TMRCTLx (x=0,1,2) 来控制热复位序列的时序。软件时序控制为用户热复位序列提供了灵活的时序设置。

- 硬件时序控制：

软件通过设置 WARSTEN (SCn\_ALTCTL[4]) 为 1，接口将通过硬件执行热复位序列。可以通过编程 INITSEL (SCn\_ALTCTL[9:8]) 来选择 SCn\_RST 到 SCn\_DATA 接收模式的时间 (T4) 和 SCn\_DATA 接收模式到 SCn\_RST 复位的时间 (T5)。硬件时序控制为用户热复位序列提供了一种简单的设置。

如下是由硬件产生的热复位控制序列：

1. 通过设置 INITSEL (SCn\_ALTCTL[9:8]) 来设定热复位的时序。
2. 通过设置 TMRSEL (SCn\_CTL[14:13]) 为 '11'，来选择 Timer0
3. 设置操作模式 OPMODE (SCn\_TMRCTL0[27:24]) 为 '0011'，和通过设置 CNT (SCn\_TMRCTL0[23:0]) 值来提供请求应答到请求的值
4. 通过设置 CNTEN0 (SCn\_ALTCTL[5]) 和 WARSTEN (SCn\_ALTCTL[4]) 来启动计数
5. 当硬件释放 SCn\_RST 为高时，硬件将同时产生 INITIF (SCn\_INTSTS[8]) 中断给 CPU (寄存器 INITIEN (SCn\_INTEN[8] = '1'))
6. 如果 Timer0 递减计数到 0 (从 SCn\_RST 开始)，而且在那之前卡没有回应 ATR，硬件将产生 TMR0IF (SCn\_INTSTS[3]) 中断给 CPU。

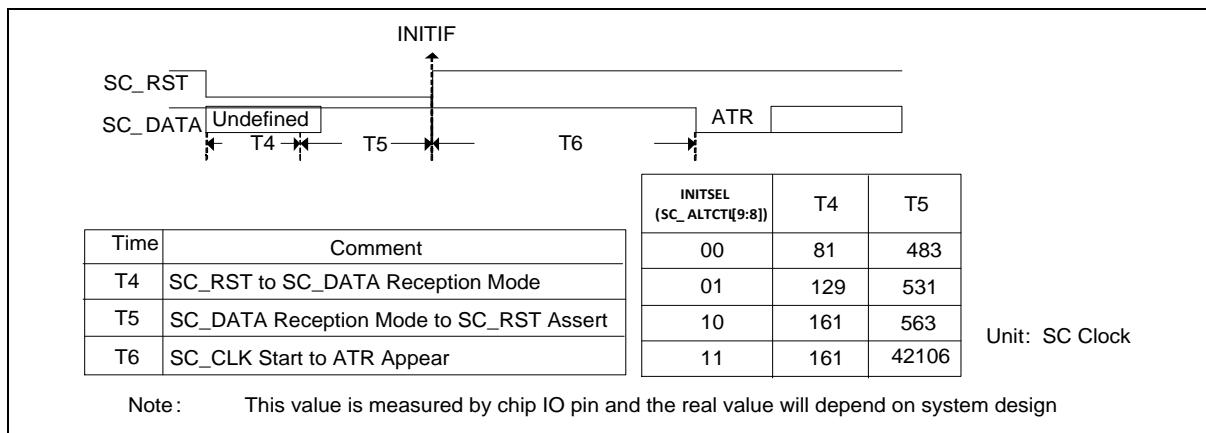


图 6.17-5 SC 热复位序列

释放

如图 6.17-6 是释放序列的示意

1. 通过设置 RSTEN (SCn\_PINCTL[1]) 为 '0'，设置 SCn\_RST 为低。等待 SYNC (SCn\_PINCTL[31]) 清 0
2. 通过设置 SCn\_CLK (SCn\_PINCTL[6]) 为 '0'，关闭 SCn\_CLK。等待 SYNC (SCn\_PINCTL[31]) 清 0
3. 通过设置 SCn\_DATA (SCn\_PINCTL[9]) 为 '0'，设置 SCn\_DATA 为低。等待 SYNC (SCn\_PINCTL[31]) 清 0
4. 通过设置 PWREN (SCn\_PINCTL[0]) 为 '0'，释放 SCn\_PWR。等待 SYNC (SCn\_PINCTL[31]) 清 0

释放序列可以有两种方法来控制，其过程如下所示：

- 软件时序控制：

软件通过设置 SCn\_PINCTL 和 SCn\_TMRCTL0 来处理释放序列。可以通过编程 SCn\_PINCTL 来设置 SCn\_PWR, SCn\_CLK, SCn\_RST 和 SCn\_DATA 的引脚状态。还可以通过设置 SCn\_TMRCTL0 来控制释放序列的时序。软件时序控制为用户释放序列提供了灵活的时序设置。

- 硬件时序控制：

软件通过设置 DACTEN (SCn\_ALTCTL[2]) 为 1，接口将通过硬件执行释放序列。可以通过配置 INITSEL (SCn\_ALTCTL[9:8]) 来选择释放触发到 SCn\_RST 变低的时间 (T7)，SCn\_RST 变低到 SCn\_CLK 停止的时间 (T8)，和 SCn\_CLK 停止到 SCn\_PWR 停止的时间 (T9)。硬件时序控制为用户释放序列提供了一种简单的设置。

如果 INITIF (SCn\_INTSTS[8]) 为 1，当硬件释放 SCn\_PWR 为低时，SC 控制器会产生一个中断 INITIF (SCn\_INTSTS[8]) 到 CPU

当设置了卡移除检测 (ADAC\_CDEN (SCn\_ALTCTL[11])=1) 时，SC 控制器也支持自动释放序列。

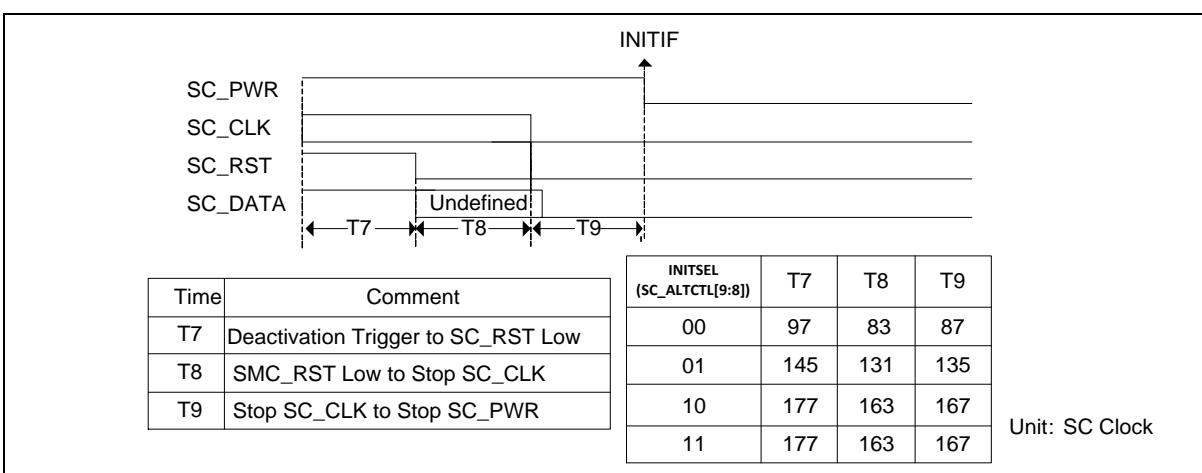


图 6.17-6 SC 释放序列

### 6.17.5.3 基本操作流程

智能卡的基本操作流程可以参考 ISO 7816-3 & EMV 协议里的规定。

如下是编程流程示意图：

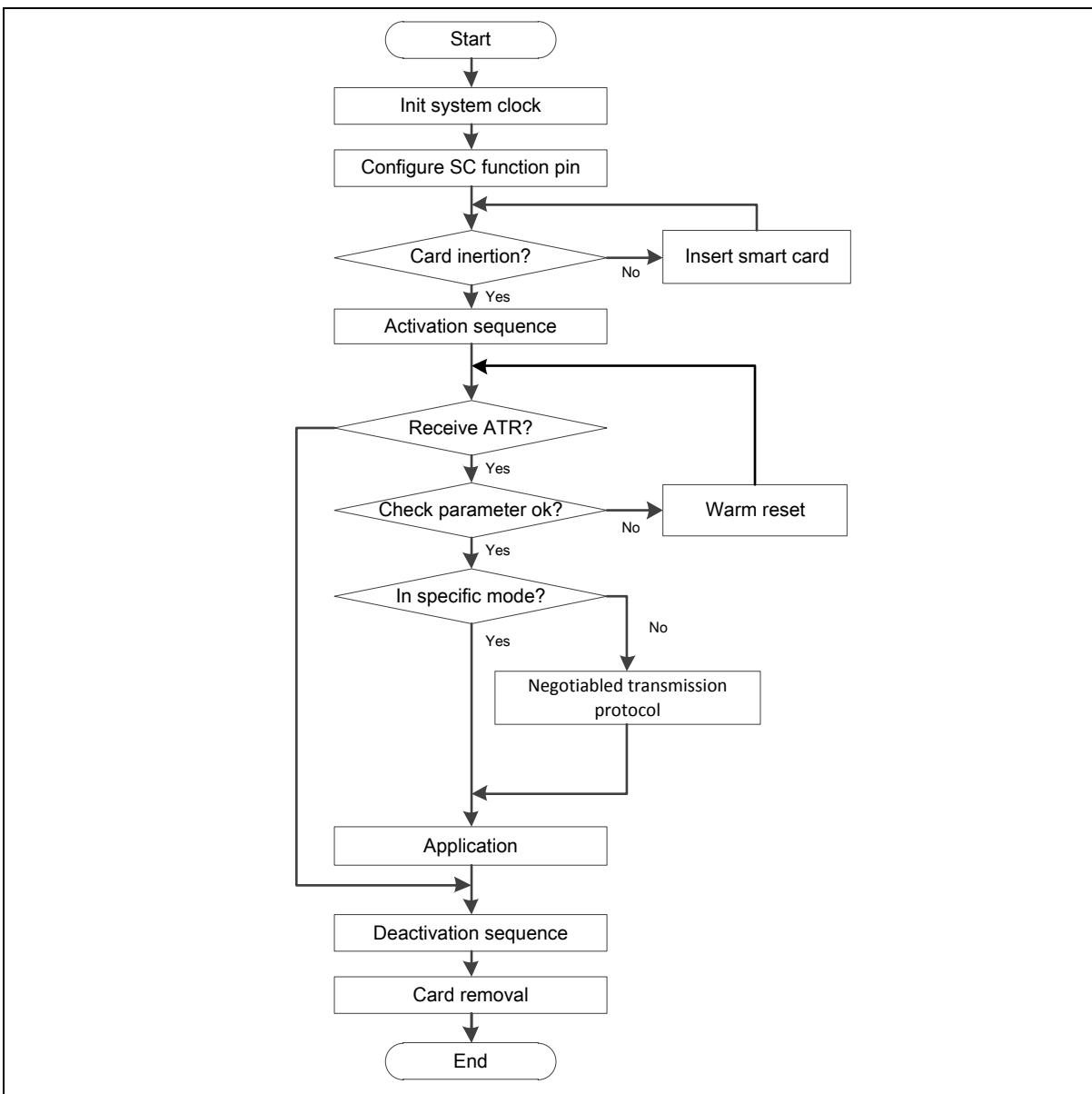


图 6.17-7 基本操作流程

#### 6.17.5.4 初始化 TS 字符

依据 ISO 7816-3 协议，初始化字符 TS 有两种可能的模式，如图 7.17-8 所示。如果 TS 模式是 1100\_0000，则是反向约定。当按反向约定进行解码后，则传送的字节等于 0x3F。如果 TS 模式是 1101\_1100，则是直接约定。当按直接约定解码后，则传送的字节等于 0x3B。用户可以设置 AUTOSEN (SCn\_CTL[3]) 让硬件来决定操作的约定。用户也可以设置 CONSEL (SCn\_CTL[5:4]) 为'00'或者'11'，在收到 ATR 的 TS 后才去改变操作的约定。

如果软件通过设置 AUTOSEN (SCn\_CTL[3]) 使能自动约定功能，则设置步骤必须在 ATR 状态之前完成，而且第一个数据必须是 0x3B 或 0x3F。在硬件收到第一个数据并放到 SCn\_DAT 中之后，硬件将决定约定模式并自动改变 CONSEL (SCn\_CTL[5:4])。如果第一个数据既不是 0x3B 也不是 0x3F, ACERRIF (SCn\_INTSTS[10] 自动约定错误中断状态标志) 置位，硬件将产生一个 INT\_ACON\_ERR 中断给 CPU (如果 ACERRIEN (SCn\_INTEN [10]) = '1')。

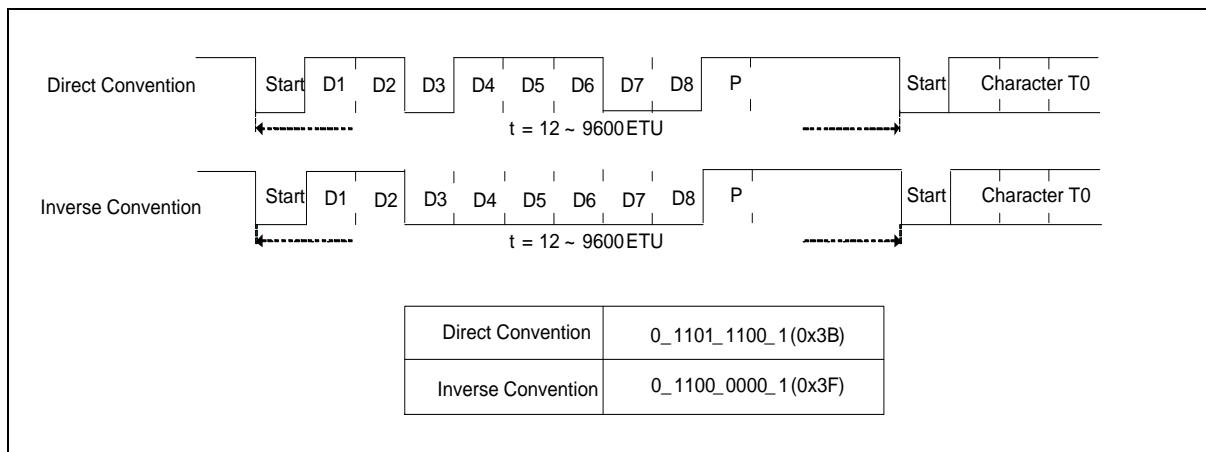


图 6.17-8 初始字符 TS

#### 6.17.5.5 发送数据流程和数据缓存状态

设置初始化序列后，SC 开始按 ISO 7816-3 规定的格式进行数据传输。配置 ETURDIV (SC\_ETUCTL[11:0]) 为 273 使 ETU 符合 ISO 7816-3 的规定。写数据到 SC\_DAT，SC 会发送 8 位的数据到智能卡。读 SC\_DAT，SC 会返回从智能卡接收到的 8 位数据。

SC\_STATUS 是 数据缓存状态寄存器。TXPOINT (SC\_STATUS[26:24]) 和 RXPOINT (SC\_STATUS[18:16]) 表示发送缓存和接收缓存里的数据量。TXEMPTY (SC\_STATUS[9]), TXFULL (SC\_STATUS[10]), TXOV (SC\_STATUS[8]), RXEMPTY (SC\_STATUS[1]), RXFULL (SC\_STATUS[2]), RXOV (SC\_STATUS[0]) 则表示发送/接收缓存的状态是满，空还是溢出。置位 TBEIEN (SCn\_INTEN[1]) 标志，SC 会在发送缓存空时产生中断到 CPU。中断产生后，写 1 到 TBEIEN (SCn\_INTEN[1]) 可以清除中断。SC 可以通过配置这些寄存器来决定是否传输数据。

置位 TXOFF (SC\_CTL[2]) 和 RXOFF (SC\_CTL[1]) 分别可以关闭 TX 和 RX。TXACT (SC\_STATUS[31]) 和 RXACT (SC\_STATUS[23]) 代表 TX 发送/RX 接收的状态是激活还是关闭。

#### 6.17.5.6 接收缓存超时

RX 缓存接收到一个新的值后超时计数器复位，开始向下计数。如果使能了 RXTOIEN (SCn\_INTEN[9])，计数器计数到 1 时还没有接收到新的值，或者 CPU 并没有通过 SCn\_DAT 读取数据，接收超时计数标志 RXTOIF (SCn\_INTSTS[9]) 会被置位，硬件会产生一个中断到 CPU

#### 6.17.5.7 错误信号和字符重复

依据 ISO7816-3 协议 T=0 模式的描述，如图 7.17 9 所示，如果接收器接收到一个错误的校验位，接收器将会拉低 SCn\_DATA 1.5 个位周期去通知发送器校验错误。然后发送器将重传该字符。SC 接口控制器支持接收器硬件错误检测功能和发送器硬件重传功能。

软件可以通过设置 TXRTYEN (SCn\_CTL[23]) 来使能重传功能。软件也可以在 TXRTY (SCn\_CTL[22:20]) 中定义重传的次数限制。如果重传次数在 1 到 TXRTY 之间，硬件自动置位 TXRERR (SCn\_STATUS[29])。重传次数可以达 TXRTY +1，当重传次数等于 TXRTY +1 时，硬件将会置 TXOVERR (SCn\_STATUS[30]) 标志，此时如果使能了 TERRIEN (SCn\_INTEN [2])，SC 控制器将产生一个传输错误中断给 CPU，TERRIF (SC\_INTSTS[2]) 标志会被置位。

用户可以配置 RXRTYEN (SCn\_CTL[19]) 使能重新接收功能。软件也可以在 RXRTY (SCn\_CTL[18:16]) 定义接收重传次数的限制。如果重收次数在 1 到 RXRTY 之间，硬件自动置位 RXRERR (SCn\_STATUS[21])。接收重传次数可达 RXTRY+1，如果接收错误的次数等于 RXTRY+1，接收器将接收该错误的数据到缓存并且硬件将会置位 RXOVERR (SCn\_STATUS[22])，如果 TERRIEN (SCn\_INTEN[2]) 使能，SC 控制将产生一个传输错误中断给 CPU，TERRIF (SC\_INTSTS[2]) 标志会被置位。

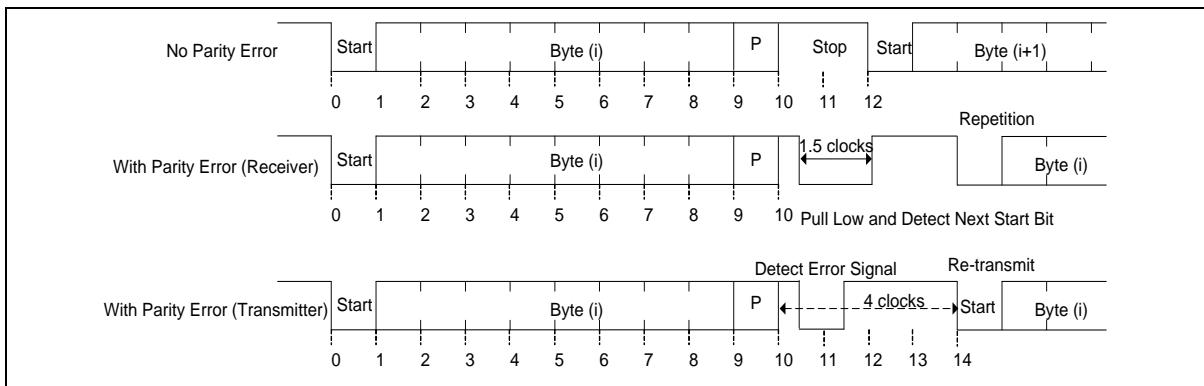


图 6.17-9 SC 错误信号

#### 6.17.5.8 内部定时器操作模式

智能卡接口包括一个 24 位超时计数器和两个 8 位超时计数器。这些计数器帮助控制器处理不同的实时间隔。每个计数器在使能位 (CNTECx 在 SCn\_ALTCTL[7:5], x = 0, 1, 2 中) 被置位或者检测到一个 START 位就开始进行计数。

下面是编程流程:

4. 配置 TMRSEL (SCn\_CTL[14:13]) 为 11 使能计数器
5. 选择操作模式 OPMODE (SCn\_TMRCTLx[27:24], x = 0, 1, 2)
6. 配置 Timer0, Timer1 和 Timer2 的计数器值寄存器 CNT0 (SCn\_TMRCTL0[23:0]), CNT1 (SCn\_TMRCTL1[7:0]) 和 CNT2 (SCn\_TMRCTL2[7:0] 寄存器).
7. 置位 CNTE0 (SCn\_ALTCTL [5]), CNTE1 (SCn\_ALTCTL [6]) 或 CNTE2 (SCn\_ALTCTL [7]) 使能定时器。ACTSTS0 (SCn\_ALTCTL[13]), ACTSTS1 (SCn\_ALTCTL[14]) 和 ACTSTS2 (SCn\_ALTCTL[15]) 指示对应计数器的使能状态
8. 满足计数条件后, 定时器开始计数
9. TMR0IEN (SCn\_INTEN[3]), TMR1IEN (SCn\_INTEN[4]), TMR2IEN (SCn\_INTEN[5]) 使能后, 当内部定时器计数器达到中断条件时, SC 会产生中断到 CPU

SCn\_TMRCTL0, SCn\_TMRCTL1 和 SCn\_TMRCTL2 定时器操作模式参见表 6.17-3.

**注 1:** 只有 Timer0 (SCn\_TMRCTL0 寄存器) 支持模式 0011.

**注 2:** START 位只能在 Tx 或 Rx 空闲时或完成最后一次传输后才能被检测到.

OPMODEM (SCn_TMRCTLx[27:24]) , X = 0, 1, 2)	操作模式描述	
0000	当 CNTECx (SCn_ALTCTL[7:5]) 使能时, 向下计数器开始计数, 当计数器超时时, 停止计数。超时值是 CNT (SCn_TMRCTL0[23:0], SCn_TMRCTL1[7:0], SCn_TMRCTL2[7:0])+1 的值。	
开始	当 CNTECx (SCn_ALTCTL[7:5]) 使能时开始计数。	
结束		
0001	当第一个 START 位 (接收或者发送) 检测到时向下计数器开始计数, 当计数器超时时停止计数。从	

	检测到 START 位到发送数据到 TX 或者从 RX 接收数据会花费 2 个 ETU 的时间。超时值是 CNT (SCn_TMRCTL0[23:0], SCn_TMRCTL1[7:0], SCn_TMRCTL2[7:0])+1 的值。	
	开始	在 CNTENx (SCn_ALTCTL[7:5]) 设置为 1 后，当检测到第一个 START 位 (接收或者发送) 时，计时器开始计数
	结束	当向下计数器的值计数到 0 时，硬件将 TMR0IF, TMR1IF, TMR2IF (SCn_INTSTS[5:3]) 置位并且自动将 CNTENx (SCn_ALTCTL[7:5]) 清零。
		当第一个 START 位 (接收) 检测到时，向下计数器开始计数，当计数器超时时，停止计数。从检测到 START 位到 RX 接收数据会花费 2 个 ETU 的时间。超时值是 CNT (SCn_TMRCTL0[23:0], SCn_TMRCTL1[7:0], SCn_TMRCTL2[7:0])+1 的值
0010	开始	在 CNTENx (SCn_ALTCTL[7:5]) 设置为 1 后，当检测到第一个 START 位 (接收) 时，计时器开始计数
0010	结束	当向下计数器的值等于 0 时，硬件将 TMR0IF, TMR1IF, TMR2IF (SCn_INTSTS[5:3]) 置位并且自动将 CNTENx (SCn_ALTCTL[7:5]) 清零。
		向下计数器只用于硬件激活，热复位序列下 ATR 时序的测量。 时序开始于 SCn_RST 释放，结束于 ATR 应答收到或超时。 如果在 ATR 应答收到之前，计数器计数到 0，此时如果 TMR0IEN (SCn_INTEN[3]) 已使能，则硬件会置位 TMR0IF (SCn_INTSTS[3])，并产生一个中断给 CPU。超时值为 (SCn_TMRCTL0[23:0])+1。
0011	开始	在 CNTENO (SCn_ALTCTL[5]) 设为 1 后，当 SCn_RST 释放时，计数器开始计数。仅用于硬件激活和热复位模式。
0011	结束	在 ATR 应答收到之前，如果向下计数器的值计数到 0，硬件将自动将 TMR0IF 置位并清 CNTENO (SCn_ALTCTL[5])。 当收到 ATR 时，向下计数器计数还未计数到 0，硬件将自动清除 CNTENO (SCn_ALTCTL[5])。
		开始 在 CNTENx (SCn_ALTCTL[7:5]) 设置为 1 后，计时器开始计数
0100	重计数 & 重载	ACTSTSx (SCn_ALTCTL[15:13]) 为 1 时，用户可以在任何时间更改寄存器 CNT (SCn_TMRCTL0[23:0], SCn_TMRCTL0[7:0], SCn_TMRCTL0[7:0]) 的值。在计数器计数到 0 之前，上次填入 CNT (SCn_TMRCTL0[23:0], SCn_TMRCTL1[7:0], SCn_TMRCTL2[7:0]) 的值会被加载到计数器。 但是仅当计数器计数到 0 时，计数器才会重载 CNT (SCn_TMRCTL0[23:0], SCn_TMRCTL1[7:0], SCn_TMRCTL2[7:0]) 并开始计数。
0100	中断	计数器计数到 0 时，硬件会置位 TMR0IF, TMR1IF, TMR2IF (SCn_INTSTS[5:3])，此时如果使能了 TMRxIEN (SCn_INTEN[5:3])，SC 会产生一个中断到 CPU。超时计数值为 CNT (SCn_TMRCTL0[23:0], SCn_TMRCTL1[7:0], SCn_TMRCTL2[7:0]) +1.
0100	结束	用户清除 CNTENx (SCn_ALTCTL[7:5]) 寄存器，向下计数器停止计数
		开始 CNTENx (SCn_ALTCTL[7:5]) 置 1 后，当第一个 START 位 (接收或者发送) 检测到时向下计时器开始计数。从检测到 START 位到发送数据到 TX 或者从 RX 接收数据会花费 2 个 ETU 的时间。
0101	重载	ACTSTSx (SCn_ALTCTL[15:13]) 为 1 时，用户可以在任何时间更改寄存器 CNT (SCn_TMRCTL0[23:0], SCn_TMRCTL0[7:0], SCn_TMRCTL0[7:0]) 的值。在计数器计数到 0 之前，上次填入 CNT (SCn_TMRCTL0[23:0], SCn_TMRCTL1[7:0], SCn_TMRCTL2[7:0]) 的值会被加载到计数器。 但是仅当计数器计数到 0 时，计数器才会重载 CNT (SCn_TMRCTL0[23:0], SCn_TMRCTL1[7:0], SCn_TMRCTL2[7:0])。
0101	重计数	向下计数器重载了 CNT 的值后，在检测到下一个 START 位时定时器重新开始计数
0101	中断	计数器计数到 0 时，硬件会置位 TMR0IF, TMR1IF, TMR2IF (SCn_INTSTS[5:3])，此时如

		如果使能了 TMRxIEN (SCn_INTEN[5:3]) , SC 会产生一个中断到 CPU。超时计数值为 CNT (SCn_TMRCTL0[23:0], SCn_TMRCTL1[7:0], SCn_TMRCTL2[7:0]) +1
	结束	用户清除 CNTENx (SCn_ALTCTL[7:5]) 寄存器, 向下计数器停止计数
0110	开始	CNTENx (SCn_ALTCTL[7:5]) 置 1 后, 当第一个 START 位 (接收或者发送) 检测到时向下计时器开始计数。从检测到 START 位到发送数据到 TX 或者从 RX 接收数据会花费 2 个 ETU 的时间。
	重载	ACTSTSx (SCn_ALTCTL[15:13]) 为 1 时, 用户可以在任何时间更改寄存器 CNT (SCn_TMRCTL0[23:0], SCn_TMRCTL0[7:0], SCn_TMRCTL0[7:0]) 的值。在计数器计数到 0 之前, 上次填入 CNT (SCn_TMRCTL0[23:0], SCn_TMRCTL1[7:0], SCn_TMRCTL2[7:0]) 的值会被加载到计数器。 但是仅当计数器计数到 0 时, 计数器才会重载 CNT (SCn_TMRCTL0[23:0], SCn_TMRCTL1[7:0], SCn_TMRCTL2[7:0])。
	重计数	向下计数器重载了 CNT 的值后, 在检测到下一个 START 位时定时器重新开始计数
	中断	计数器计数到 0 时, 硬件会置位 TMR0IF, TMR1IF, TMR2IF (SCn_INTSTS[5:3]), 此时如果使能了 TMRxIEN (SCn_INTEN[5:3]) , SC 会产生一个中断到 CPU。超时计数值为 CNT (SCn_TMRCTL0[23:0], SCn_TMRCTL1[7:0], SCn_TMRCTL2[7:0]) +1
	结束	用户清除 CNTENx (SCn_ALTCTL[7:5]) 寄存器, 向下计数器停止计数
0111	开始	CNTENx (SCn_ALTCTL[7:5]) 置 1 后, 当第一个 START 位 (接收或者发送) 检测到时向下计时器开始计数。从检测到 START 位到发送数据到 TX 或者从 RX 接收数据会花费 2 个 ETU 的时间。
	重载 & 重计数	仅当检测到下一个 START 位, 计数器会载入新的计数值 CNT (SCn_TMRCTL0[23:0], SCn_TMRCTL1[7:0], SCn_TMRCTL2[7:0]) 并重新开始计数
	中断	计数器计数到 0 时, 硬件会置位 TMR0IF, TMR1IF, TMR2IF (SCn_INTSTS[5:3]), 此时如果使能了 TMRxIEN (SCn_INTEN[5:3]) , SC 会产生一个中断到 CPU。超时计数值为 CNT (SCn_TMRCTL0[23:0], SCn_TMRCTL1[7:0], SCn_TMRCTL2[7:0]) +1.
	结束	用户清除 CNTENx (SCn_ALTCTL[7:5]) 寄存器, 向下计数器停止计数
1111	开始	在 CNTENx (SCn_ALTCTL[7:5]) 设置为 1 后, 计时器开始计数。超时后停止计数。
	重载 & 重计数	仅当检测到下一个 START 位, 计数器会载入新的计数值 CNT (SCn_TMRCTL0[23:0], SCn_TMRCTL1[7:0], SCn_TMRCTL2[7:0]) 并重新开始计数
	中断	计数器计数到 0 时, 硬件会置位 TMR0IF, TMR1IF, TMR2IF (SCn_INTSTS[5:3]), 此时如果使能了 TMRxIEN (SCn_INTEN[5:3]) , SC 会产生一个中断到 CPU。超时计数值为 CNT (SCn_TMRCTL0[23:0], SCn_TMRCTL1[7:0], SCn_TMRCTL2[7:0]) +1.
	结束	用户清除 CNTENx (SCn_ALTCTL[7:5]) 寄存器, 向下计数器停止计数

表 6.17-3 Timer0/Timer1/Timer2 操作模式

## 6.17.5.9 块保护时间和扩展保护时间

块保护时间是在不同传输方向之间的两个连续字符的第一位边缘之间的最短位长度。该域表示用于块保护时间的计数器。依据 ISO7816-3, 在 T=0 的模式下, 软件必须填充该域为 15 (实际的块保护时间为 16.5), 在 T=1 的模式下, 软件必须填充该域位 21 (实际的块保护时间为 22.5)。

发送时, 智能卡先发送数据到智能卡主控制器。在一段时间 (长度超过 BGT (SCn\_CTL[12:8])) 后, 智能卡主控制器开始发送数据。

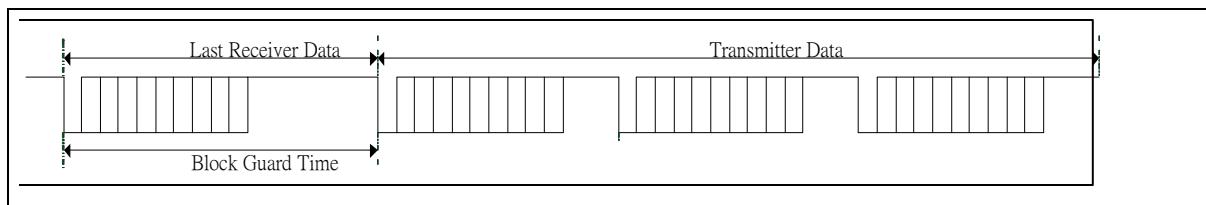


图 6.17-10 发送时块保护时间操作

接收时，智能卡主控制器先发送数据给智能卡。如果智能卡发送数据给智能卡主控制器的时间少于 BGT (SCn\_CTL[12:8])，当 RXBGTE (SCn\_ALTCTL[12]) 和 BGTIEN (SCn\_INTEN[6]) 都使能时，块保护时间中断 BGTFIF (SCn\_INTSTS[6]) 发生。

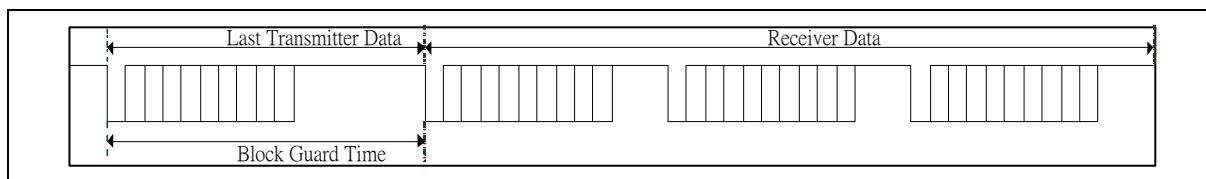


图 6.17-11 接收时的块保护时间操作

扩展保护时间为 EGT (SCn\_EGT[7:0])，它只影响由智能卡接口发送的数据，具体格式如图 7.17 12 所示：.

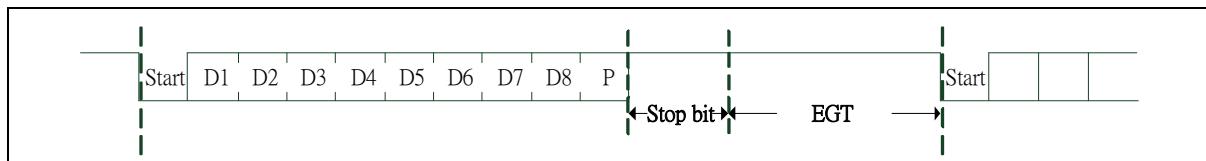


图 6.17-12 扩展保护时间操作

#### 6.17.5.10 UART 模式

当 UARTEN (SCn\_UARTCTL[0]) 被置位，智能卡控制器也可以作为一个基本的 UART 功能来使用。下面是一个 UART 模式的编程例子。

##### 编程示例：

1. 置位 UARTEN (SCn\_UARTCTL[0]) 进入 UART 模式。.
2. 配置 RXRST (SCn\_ALTCTL[1]) 和 RXRST (SCn\_ALTCTL[1]) 进行软件复位，以确保所有的状态机返回空闲状态。
3. 填充 0 到 CONSEL (SCn\_CTL[5:4]) 和 AUTOCEN (SCn\_CTL[3])。UART 模式下，这些位必须为 0
4. 配置 ETURDIV (SCn\_ETUCTL[11:0]) 选择串口波特率。例如，如果智能卡模块的时钟为 12MHZ，而且目标波特率为 115200，则 ETURDIV 应设填为 (12000000/115200-1)
5. 选择数据格式，包括数据长度 (配置寄存器 WLS (SCn\_UARTCTL [5:4]))，校验位格式 (配置寄存器 OPE (SCn\_UARTCTL[7]) 和 PBOFF (SCn\_UARTCTL[6])) 和停止位长度 (配置寄存器 NSB (SCn\_CTL[15] 或 EGT (SCn\_EGT[7:0])))。
6. 通过设定 RXTRGLV (SCn\_CTL[7:6]) 域来选择接收缓存触发级别，并通过设定 RFTM

(SCn\_RXTO[8:0]) 域来选择接收缓存超时间隔。

7. 写 SCn\_DAT (SCn\_DAT[7:0]) (TX) 或者读 SCn\_DAT (SCn\_DAT[7:0]) (RX) 执行 UART 功能。

### 6.17.6 寄存器映射

R: 只读, W: 只写, R/W: 读/写

寄存器	偏移量	R/W	描述	复位值
<b>SC 基址:</b> <b>SCn_BA = 0x4009_0000 + (0x1000 * n)</b> n=0,1,2				
<b>SC_DAT</b>	SCn_BA+0x00	R/W	SC 接收/发送缓存保持寄存器	0xFFFF_XXXX
<b>SC_CTL</b>	SCn_BA+0x04	R/W	SC 控制寄存器	0x0000_0000
<b>SC_ALTCTL</b>	SCn_BA+0x08	R/W	SC 交替控制寄存器	0x0000_0000
<b>SC_EGT</b>	SCn_BA+0x0C	R/W	SC 扩展保护寄存器	0x0000_0000
<b>SC_RXTOUT</b>	SCn_BA+0x10	R/W	SC 接收缓存超时寄存器	0x0000_0000
<b>SC_ETUCTL</b>	SCn_BA+0x14	R/W	SC ETU 控制寄存器	0x0000_0173
<b>SC_INTEN</b>	SCn_BA+0x18	R/W	SC 中断使能控制寄存器	0x0000_0000
<b>SC_INTSTS</b>	SCn_BA+0x1C	R/W	SC 中断状态寄存器	0x0000_0002
<b>SC_STATUS</b>	SCn_BA+0x20	R/W	SC 传输状态寄存器	0x0000_X202
<b>SC_PINCTL</b>	SCn_BA+0x24	R/W	SC 管脚控制状态寄存器	0x0000_0000
<b>SC_TMRCTL0</b>	SCn_BA+0x28	R/W	SC 内部定时器 0 寄存器	0x0000_0000
<b>SC_TMRCTL1</b>	SCn_BA+0x2C	R/W	SC 内部定时器 1 寄存器	0x0000_0000
<b>SC_TMRCTL2</b>	SCn_BA+0x30	R/W	SC 内部定时器 2 寄存器	0x0000_0000
<b>SC_UARTCTL</b>	SCn_BA+0x34	R/W	SC UART 模式控制寄存器	0x0000_0000
<b>SC_ACTCTL</b>	SCn_BA+0x4C	R/W	SC 激活控制寄存器	0x0000_0000

## 6.17.7 寄存器描述

**SC\_DAT** SC 接收/发送缓存保持寄存器

寄存器	偏移量	R/W	描述	复位值
<b>SC_DAT</b>	SCn_BA+0x00	R/W	SC 接收/发送缓存保持寄存器	0XXXXX_XXXX

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
DAT							

位	描述	
[31:8]	<b>Reserved</b>	保留
[7:0]	<b>DAT</b>	<p>接收/发送保持缓存器</p> <p>写操作:</p> <p>通过写数据到该寄存器, SC 将发送出一个 8 位数据</p> <p><b>注:</b> 如果 SCEN (SCn_CTL[0]) 没有使能, 不能写该寄存器.</p> <p>读操作:</p> <p>通过读该寄存器, SC 将返回接收到的一个 8-位数据</p>

**SC\_CTL SC 控制寄存器**

寄存器	偏移量	R/W	描述	复位值
SC_CTL	SCn_BA+0x04	R/W	SC 控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved	SYNC	Reserved			CDLV	CDBSEL	
23	22	21	20	19	18	17	16
TXRTYEN	TXRTY			RXRTYEN	RXRTY		
15	14	13	12	11	10	9	8
NSB	TMRSEL		BGT				
7	6	5	4	3	2	1	0
RXTRGLV		CONSEL		AUTOCEN	TXOFF	RXOFF	SCEN

位	描述	
[31]	Reserved	保留
[30]	SYNC	<b>SYNC 标志指示 (只读)</b> 用于同步，软件在写新的值到 RXTRY 和 TXTRY 之前必须检查该位 0 = 同步已经完成，用户可以写入新数据给 RXTRY 和 TXTRY 1 = 最后值在同步中。
[29:27]	Reserved	保留
[26]	CDLV	<b>智能卡检测电平选择</b> 0 = 当硬件检测到卡检测脚 (SCn_CD) 电平从高变为低时，为检测到卡 1 = 当硬件检测到卡检测脚 (SCn_CD) 电平从低变为高时，为检测到卡。 <b>注：</b> 软件必须在智能卡引擎使能前选定卡检测电平
[25:24]	CDBSEL	<b>智能卡检测消抖选择</b> 该域为智能卡检测消抖选择 00 = 每 384 (128*3) 个 SC 外部时钟去抖采样卡插入一次，每 128 个 SC 外部时钟去抖采样卡移除一次 其他配置保留
[23]	TXRTYEN	<b>TX 错误重传使能位</b> 当校验错误发生时，该位使能发送器重传 0 = 禁用 TX 错误重传功能。 1 = 使能 TX 错误重传功能
[22:20]	TXRTY	<b>TX 错误重传计数</b> 该域表明当校验错误发送后，发送器允许尝试的最多重传次数。 <b>注 1：</b> 实际的重传次数是 TXTRY+1，所以 8 次是最多的重传次数。 <b>注 2：</b> 当 TXRTYEN 使能时，该域不能被修改。修改该域的流程是先禁用 TXRTYEN，然后填充新的重传值。

[19]	<b>RXRTYEN</b>	<b>RX 错误重收使能位</b> 当校验位错误发生时，该位使能接收器重收功能 0 = 禁止 RX 错误重收功能 1 = 使能 RX 错误重收功能 <b>注：</b> 在使能该位之前，软件必须先填充 RXRTY
[18:16]	<b>RXRTY</b>	<b>RX 错误重收次数</b> 该域表示当校验错误发生时，接收器允许重收的最多次数。 <b>注 1：</b> 实际的重接收次数是 RXRTY+1，所以 8 次是最多的重收次数。 <b>注 2：</b> 当 RXRTYEN 使能时，该域不能被修改。修改该域的流程是先禁用 RXRTYEN，然后填充新的重收值。
[15]	<b>NSB</b>	<b>停止位长度</b> 该域表示停止位的长度。 0 = 停止位的长度是 2 ETU. 1 = 停止位的长度是 1 ETU. <b>注 1：</b> 默认停止位的长度是 2。SMC 和 UART 采用 NSB 来设置停止位的长度。 <b>注 2：</b> UART 模式下，配置 NSB 为 0 时，RX 可以接收 1 位或 2 位停止位的数据。
[14:13]	<b>TMRSEL</b>	<b>定时器通道选择</b> 00 = 禁用所有内部定时器 11 = 使能内部 24 位定时器和两个 8 位定时器。软件可以通过设置寄存器 SCn_TMRCTL0 [23:0]，寄存器 SCn_TMRCTL1 [7:0] 和寄存器 SCn_TMRCTL2 [7:0] 来配置定时器 其他配置保留
[12:8]	<b>BGT</b>	<b>块保护时间 (BGT)</b> 块保护时间是在不同传输方向之间的两个连续字符的第一位边缘之间的最短位长度。该域为用于块保护时间的计数器。依据 ISO7816-3，在 T=0 的模式下，软件必须填充该域为 15 (实际的块保护时间=16.5)，在 T=1 的模式下，软件必须填充该域为 21 (实际的块保护时间=22.5) <b>注：</b> 实际块保护时间是 BGT + 1.
[7:6]	<b>RXTRGLV</b>	<b>Rx 缓存触发水平</b> 当接收到的字节数等于 RXTRGLV 时，RDAIF 会被置位。此时如果使能了 RDAIEN (SCn_INTEN[0])，将会产生中断到 CPU 00 = Rx 缓存触发水平为 01 字节. 01 = Rx 缓存触发水平为 02 字节. 10 = Rx 缓存触发水平为 03 字节. 11 = 保留
[5:4]	<b>CONSEL</b>	<b>约定选择</b> 00 = 直接约定. 01 = 保留 10 = 保留 11 = 反向约定 <b>注：</b> 如果寄存器 AUTOCEN (SCn_CTL[3]) 使能，该域将被忽略
[3]	<b>AUTOCEN</b>	<b>自动约定使能位</b> 该位用来使能自动约定功能 0 = 禁止自动约定功能 1 = 使能自动约定功能.

		<p>如果软件使能自动约定功能，设置步骤必须在应答转复位状态之前完成，而且第一个数据必须是 0x3B 或者 0x3F。在硬件收到第一个数据并保存到缓存后，硬件将决定何种约定并自动改变寄存器 CONSEL (SCn_CTL[5:4]) 的值。如果收到的第一个字节是 0x3B，TS 为直接约定，CONSEL (SCn_CTL[5:4]) 会被自动设为 00。其他情况，TS 是反向约定 CONSEL (SCn_CTL[5:4]) 会被设为 11</p> <p>如果第一个数据既不是 0x3B 也不是 0x3F，且使能了 ACERRIEN (SCn_INTEN[10])，则硬件将产生中断到 CPU</p>
[2]	<b>TXOFF</b>	<p><b>禁止 TX 发送控制位</b></p> <p>该位用来禁止 TX 发送功能</p> <p>0 = 使能发送</p> <p>1 = 禁止发送</p>
[1]	<b>RXOFF</b>	<p><b>禁止 RX 接收控制位</b></p> <p>该位用来禁止 RX 接收功能</p> <p>0 = 使能接收</p> <p>1 = 禁止发送</p> <p><b>注 1:</b>如果 AUTOSEN (SCn_CTL[3]) 使能，该域被忽略.</p>
[0]	<b>SCEN</b>	<p><b>SC 控制器使能位</b></p> <p>该位为 0 时，写 1 到该位使能 SC 控制器</p> <p>0 = 强制 SC 进入空闲状态</p> <p>1 = SC 控制器使能，所有功能正常</p> <p><b>注 1:</b> 在写其他 SC 寄存器前需要确保 SCEN 为 1，否则智能卡将不能正常工作</p>

**SC\_ALTCTL SC 交替控制寄存器**

寄存器	偏移量	R/W	描述	复位值
SC_ALTCTL	SCn_BA+0x08	R/W	SC 交替控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
SYNC	Reserved						
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
ACTSTS2	ACTSTS1	ACTSTS0	RXBGTEN	ADACEN	Reserved	INITSEL	
7	6	5	4	3	2	1	0
CNTEN2	CNTEN1	CNTEN0	WARSTEN	ACTEN	DACTEN	RXRST	TXRST

位	描述	
[31]	SYNC	<b>SYNC 标志指示 (只读)</b> 用于同步，软件在写新的值到 SCn_ALTCTL 之前必须检查该位 0 = 同步已经完成，用户可以写入新数据给 SCn_ALTCTL 1 = 最后值在同步中。
[30:16]	Reserved	保留
[15]	ACTSTS2	<b>内部 Timer2 激活状态 (只读)</b> 该位表示内部定时器 Timer2 计数器的状态 0 = Timer2 未激活。 1 = Timer2 激活。 <small>注: Timer2 激活并不意味着 CNT (SCn_TMRCTL2[7:0]) 处于计数状态</small>
[14]	ACTSTS1	<b>内部 Timer1 激活状态 (只读)</b> 该位表示内部定时器 Timer1 计数器的状态 0 = Timer1 未激活 1 = Timer1 激活 <small>注: Timer1 激活并不意味着 CNT (SCn_TMRCTL1[7:0]) 处于计数状态</small>
[13]	ACTSTS0	<b>内部 Timer0 激活状态 (只读)</b> 该位表示内部定时器 Timer0 计数器的状态 0 = Timer0 未激活 1 = Timer0 激活 <small>注: Timer0 激活并不意味着 CNT (SCn_TMRCTL0[23:0]) 处于计数状态</small>
[12]	RXBGTEN	接收器块保护时间功能使能位 0 = 禁止接收器块保护时间功能 1 = 使能接收器块保护时间功能。
[11]	ADACEN	智能卡移除时自动释放

		该位用来使能智能卡移除时自动释放功能 0 = 禁止自动释放 1 = 使能自动释放 <b>注:</b> 当智能卡移除时, 如果该位置位, 硬件会停止所有操作并释放序列。自动序列释放后, 硬件会置位 INITIF (SCn_INTSTS[8])
[10]	<b>Reserved</b>	保留
[9:8]	<b>INITSEL</b>	<b>初始化时间选择</b> 该域表示硬件激活、热复位或者释放的初始化时间。初始化时间的单位是 SC 模块时钟 激活: 参考图 6.17-54 SC 激活序列。 热复位: 参考图 6.17-5 热复位序列。 释放: 参考图 6.17-56 释放序列。 <b>注:</b> 当释放和热复位处于 Timer0 操作模式 0011 时, 释放最多要 128 个 SC 模块时钟周期
[7]	<b>CNTEN2</b>	<b>内部定时器 2 开始使能位</b> 该位使能定时器 2 开始计数。软件可以写 0 来停止计数器, 置 1 来重载和计数。 0 = 停止计数 1 = 开始计数。 <b>注 1:</b> 当寄存器 TMRSEL (SCn_CTL[14:13]) = 11 时, 该域用于内部 8 位定时器。当寄存器 TMRSEL (SCn_CTL[14:13]) 不为 11 时, 不要填充 CNTEN2。 <b>注 2:</b> 如果操作模式不是自动重载模式 (SCn_TMRCTL2[26] = 0), 则该位会由硬件自动清零。 <b>注 3:</b> 如果寄存器 SCEN (SCn_CTL[0]) 未使能, 不能对该位进行编程。
[6]	<b>CNTEN1</b>	<b>内部定时器 1 开始使能位</b> 该位使能定时器 1 开始计数。软件可以写 0 来停止计数器, 置 1 来重载和计数。 0 = 停止计数 1 = 开始计数。 <b>注 1:</b> 当寄存器 TMRSEL (SCn_CTL[14:13]) = 10 或 11 时, 该域用于内部 8 位定时器。当寄存器 TMRSEL (SCn_CTL[14:13]) 不为 11 时, 不要填充 CNTEN1。 <b>注 2:</b> 如果操作模式不是自动重载模式 (SCn_TMRCTL1[26] = 0), 则该位将被硬件自动清除。 <b>注 3:</b> 如果寄存器 SCEN (SCn_CTL[0]) 未使能, 不能对该位进行编程。
[5]	<b>CNTEN0</b>	<b>内部定时器 0 开始使能位</b> 该位使能定时器 0 开始计数。软件可以写 0 来停止计数器, 置 1 来重载和计数。 0 = 停止计数 1 = 开始计数。 <b>注 1:</b> 当寄存器 TMRSEL (SCn_CTL[14:13]) = 11 时, 该域用于内部 24 位定时器。 <b>注 2:</b> 如果操作模式不是自动重载模式 (SCn_TMRCTL0[26] = 0), 则该位将被硬件自动清除。 <b>注 3:</b> 如果寄存器 SCEN (SCn_CTL[0]) 未使能, 不能对该位进行编程。
[4]	<b>WARSTEN</b>	<b>热复位序列发生器使能位</b> 该位使能 SC 控制器通过热复位序列初始化卡 0 = 无效。 1 = 使能热复位序列发生器。 <b>注 1:</b> 当热复位序列完成后, 该位将被自动清除, 寄存器 INITIF (SCn_INTSTS[8]) 将被设置为 1。 <b>注 2:</b> 该域将被 TXRST (SCn_ALTCTL[0]) 和 RXRST (SCn_ALTCTL[1]) 清除, 所以不要同

		时填充 WARSTEN、TXRST 和 RXRST。  <b>注 3:</b> 如果寄存器 SCEN (SCn_CTL[0]) 未使能, 该域不能被编程。 <b>注 4:</b> 热复位过程中, RX 会被自动关闭, 无法接收数据。热复位序列完成后, RXOFF (SCn_CTL[1]) 在热复位完成之前会一直保持不变
[3]	ACTEN	激活序列发生器使能位 该位使能 SC 控制器通过激活序列初始化智能卡 0 = 无效 1 = 使能激活序列发生器 <b>注 1:</b> 当激活序列完成后, 该位将被自动清除, 寄存器 INITIF (SCn_INTSTS[8]) 将被设置为 1。 <b>注 2:</b> 该域将被寄存器 TXRST (SCn_ALTCTL[0]) 和 RXRST (SCn_ALTCTL[1]) 清除, 所以不要同时填充 ACTEN、TXRST 和 RXRST <b>注 3:</b> 如果寄存器 SCEN (SCn_CTL[0]) 未使能, 该域不能被编程。 <b>注 4:</b> 激活序列过程中, RX 会被自动关闭, 无法接收数据。硬件激活完成后, RXOFF (SCn_CTL[1]) 在热复位完成之前会一直保持不变
[2]	DACTEN	释放序列发生器使能位 该位使能 SC 控制器通过释放序列初始化智能卡 0 = 无效 1 = 使能释放序列发生器 <b>注 1:</b> 当释放序列完成后, 该位将被自动清除, 寄存器 INITIF (SCn_INTSTS[8]) 将被设置为 1。 <b>注 2:</b> 该域将被寄存器 TXRST (SCn_ALTCTL[0]) 和 RXRST (SCn_ALTCTL[1]) 清除, 所以不要同时填充 DACTEN、TXRST 和 RXRST <b>注 3:</b> 如果寄存器 SCEN (SCn_CTL[0]) 未使能, 该域不能被编程。
[1]	RXRST	<b>RX 软件复位</b> 当 RXRST 被置位, 接收缓存中的所有数据和 RX 内部状态机将被清除。 0 = 无效 1 = 复位 RX 内部状态机和指针. <b>注:</b> 在复位完成后, 该位将被自动清除
[0]	TXRST	<b>TX 软件复位</b> 当 TXRST 被置位, 发送缓存中的所有数据和 TX 内部状态机将被清除。 0 = 无效 1 = 复位 TX 内部状态机和指针. <b>注:</b> 在复位完成后, 该位将被自动清除

**SC\_EGT SC 扩展保护寄存器**

寄存器	偏移量	R/W	描述	复位值
SC_EGT	SCn_BA+0x0C	R/W	SC 扩展保护寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
EGT							

位	描述	
[31:8]	Reserved	保留
[7:0]	EGT	<b>扩展保护时间</b> 该域表示扩展保护时间的值 <b>注:</b> 该计数器的单位为 ETU

**SC\_RXTOOUT SC 接收缓存超时寄存器**

寄存器	偏移量	R/W	描述	复位值
SC_RXTOOUT	SCn_BA+0x10	R/W	SC 接收缓存超时寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
RFTM							

位	描述	
[31:9]	<b>Reserved</b>	保留
[8:0]	<b>RFTM</b>	<p><b>SC 接收 FIFO 超时寄存器</b></p> <p>当 RX 接收到一个新的值时，向下的超时计数器复位并开始计数。如果计数器计数到 1 还没有新的接收值或 CPU 没去 SCn_DAT 读取数据，超时标志 RXTOIF (SCn_INTSTS[9]) 会被置位，如果 RXTOIEN (SCn_INTEN[9]) 使能，硬件会产生中断到 CPU</p> <p><b>注 1:</b> 计数单位基于 ETU，超时间隔为 RFTM + 0.5</p> <p><b>注 2:</b> 填充该寄存器所有位为 0 关闭该功能</p>

**SC ETUCTL SC ETU 控制寄存器**

寄存器	偏移量	R/W	描述	复位值
SC_ETUCTL	SCn_BA+0x14	R/W	SC ETU 控制寄存器	0x0000_0173

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved				ETURDIV			
7	6	5	4	3	2	1	0
ETURDIV							

位	描述	
[31:12]	<b>Reserved</b>	保留
[11:0]	<b>ETURDIV</b>	<p><b>ETU 除频</b>            该域用于 ETU 时钟频率除频            真实的 ETU 值为 ETURDIV + 1.  <b>注:</b> 用户配置该寄存器时需要确保大于 0x04</p>

**SC\_INTEN SC 中断使能控制寄存器**

寄存器	偏移量	R/W	描述	复位值
SC_INTEN	SCn_BA+0x18	R/W	SC 中断使能控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved					ACERRIEN	RXTOIEN	INITIEN
7	6	5	4	3	2	1	0
CDIEN	BGTIEN	TMR2IEN	TMR1IEN	TMR0IEN	TERRIEN	TBEIEN	RDAIEN

位	描述	
[31:11]	Reserved	保留
[10]	ACERRIEN	<b>自动约定错误中断使能位</b> 该域用于自动约定错误中断使能。 0 = 禁止自动约定错误中断。 1 = 使能自动约定错误中断。
[9]	RXTOIEN	<b>接收器缓存超时中断使能位</b> 该域用于接收器缓存超时中断使能 0 = 禁止接收缓存超时中断。 1 = 使能接收缓存超时中断。
[8]	INITIEN	<b>初始化结束中断使能位</b> 该域用于激活 (ACTEN (SCn_ALTCTL[3] = 1), 释放 ((DACTEN SCn_ALTCTL[2]) = 1) 和热复位 (WARSTEN (SCn_ALTCTL [4]))) 序列中断使能 0 = 禁止初始化结束中断。 1 = 使能初始化结束中断。
[7]	CDIEN	<b>卡检测中断使能位</b> 该域用于智能卡检测中断使能，智能卡检测状态的寄存器是 CINSERT (SCn_STATUS[13])。 0 = 禁止卡检测中断。 1 = 使能卡检测中断。
[6]	BGTIEN	<b>块保护时间中断使能位</b> 该域用于块保护时间中断使能 0 = 禁止块保护时间中断 1 = 使能块保护时间中断。 注：该位仅在块保护时间接收方向有效

[5]	<b>TMR2IEN</b>	<b>Timer2 中断使能位</b> 该域用于 Timer2 中断使能 0 = 禁止 Timer2 中断. 1 = 使能 Timer2 中断
[4]	<b>TMR1IEN</b>	<b>Timer1 中断使能位</b> 该域用于 Timer1 中断使能 0 = 禁止 Timer1 中断. 1 = 使能 Timer1 中断
[3]	<b>TMR0IEN</b>	<b>Timer0 中断使能位</b> 该域用于 Timer0 中断使能 0 = 禁止 Timer0 中断. 1 = 使能 Timer0 中断
[2]	<b>TERRIEN</b>	<b>传输错误中断使能位</b> 该域用于传输错误中断使能。传输错误状态保存在 SCn_STATUS 寄存器，包括接收器断开错误 BEF (SCn_STATUS[6])，帧错误 FEF (SCn_STATUS[5])，校验错误 PEF (SCn_STATUS[4])，接收缓存溢出错误 RXOV (SCn_STATUS[0])，传输缓存溢出错误 TXOV (SCn_STATUS[8])，接收器重收超过限制错误 RXOVERR (SCn_STATUS[22]) 和发送器重传超过限制错误 TXOVERR (SCn_STATUS[30])。 0 = 禁止传输错误中断. 1 = 使能传输错误中断
[1]	<b>TBEIEN</b>	<b>发送缓存空中断使能位</b> 该域用于发送缓存空中断使能 0 = 禁止发送缓存空中断. 1 = 使能发送缓存空中断
[0]	<b>RDAIEN</b>	<b>接收数据到达中断使能位</b> 该域用于接收数据达到触发水平寄存器 RXTRGLV (SCn_CTL[7:6]) 的中断使能 0 = 禁止接收数据达到触发水平中断. 1 = 使能接收数据达到触发水平中断.

**SC\_INTSTS SC 中断状态寄存器**

寄存器	偏移量	R/W	描述	复位值
<b>SC_INTSTS</b>	SCn_BA+0x1C	R/W	SC 中断状态寄存器	0x0000_0002

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved					ACERRIF	RXTOIF	INITIF
7	6	5	4	3	2	1	0
<b>CDIF</b>	<b>BGTIF</b>	<b>TMR2IF</b>	<b>TMR1IF</b>	<b>TMR0IF</b>	<b>TERRIF</b>	<b>TBEIF</b>	<b>RDAIF</b>

位	描述
[31:11]	<b>Reserved</b> 保留
[10]	<b>ACERRIF</b> 自动约定错误中断状态标志 该位指示自动约定序列错误 0 = ATR 状态收到的 TS 为 0x3B 或 0x3F. 1 = ATR 状态收到的 TS 既不是 0x3B 也不是 0x3F. <b>注:</b> 该位写 1 清 0
[9]	<b>RXTOIF</b> 接收缓存超时中断状态标志 (只读) 该域用来指示接收缓存超时中断状态标志 0 = 接收缓存超时中断未发生 1 = 接收缓存超时中断已发生 <b>注:</b> 该位只读, 用户需要通过读 SCn_DAT 寄存器清空所有接收缓存来清除它
[8]	<b>INITIF</b> 初始化结束中断状态标志 该位为激活 (ACTEN (SCn_ALTCTL[3])), 释放 (DACTEN (SCn_ALTCTL[2])) 和 热复位 (WARSTEN (SCn_ALTCTL[4])) 序列中断状态标志 0 = 初始化序列未完成 1 = 初始化序列已完成 <b>注:</b> 该位写 1 清 0
[7]	<b>CDIF</b> 卡检测中断状态标志 (只读) 该位为智能卡检测中断状态标志。智能卡检测状态相关寄存器有 CINSERT (SCn_STATUS[12]) 和 CREMOVE (SCn_STATUS[11]) 0 = 智能卡检测事件未发生 1 = 智能卡检测事件发生 <b>注:</b> 该位只读, 用户需要清除 CINSERT 或 CREMOVE 来清除该位
[6]	<b>BGTIF</b> 块保护时间中断状态标志

		<p>该域用于接收方向上的块保护时间中断状态标志 0 = 块保护时间中断未发生 1 = 块保护时间中断发生 <b>注 1:</b> 该位仅在 RXBGTE (SCn_ALTCTL[12]) 使能时有效 <b>注 2:</b> 该位写 1 清 0</p>
[5]	<b>TMR2IF</b>	<p><b>Timer2 中断状态标志</b> 该域为 Timer2 中断状态标志 0 = Timer2 中断未发生 1 = Timer2 中断已发生 <b>注:</b> 该位写 1 清 0</p>
[4]	<b>TMR1IF</b>	<p><b>Timer1 中断状态标志</b> 该域为 Timer1 中断状态标志 0 = Timer1 中断未发生 1 = Timer1 中断已发生 <b>注:</b> 该位写 1 清 0</p>
[3]	<b>TMROIF</b>	<p><b>Timer0 中断状态标志</b> 该域为 Timer0 中断状态标志 0 = Timer0 中断未发生 1 = Timer0 中断已发生 <b>注:</b> 该位写 1 清 0</p>
[2]	<b>TERRIF</b>	<p><b>传输错误中断状态标志</b> 该域为传输错误中断状态标志。传输错误状态保存在 SCn_STATUS 寄存器，包括接收器断开错误 BEF (SCn_STATUS[6]), 帧错误 FEF (SCn_STATUS[5]), 校验错误 PEF (SCn_STATUS[4]), 接收缓存溢出错误 RXOV (SCn_STATUS[0]), 传输缓存溢出错误 TXOV (SCn_STATUS[8]), 接收器重收超过限制错误 RXOVERR (SCn_STATUS[22]) 和发送器重传超过限制错误 TXOVERR (SCn_STATUS[30])。 0 = 传输错误中断未发生 1 = 传输错误中断已发生 <b>注 1:</b> 该位为 BEF, FEF, PEF, RXOV, TXOV, RXOVERR 或 TXOVERR 的标志 <b>注 2:</b> 该位写 1 清 0</p>
[1]	<b>TBEIF</b>	<p><b>发送缓存空中断状态标志 (只读)</b> 该域为发送缓存空中断状态标志 0 = 发送缓存非空 1 = 发送缓存空 <b>注:</b> 该位只读. 如果用户想要清除该未，用户写值到 DAT (SCn_DAT[7:0]), 该位会自动清除</p>
[0]	<b>RDAIF</b>	<p><b>接收数据到达中断状态标志 (只读)</b> 该域为 接收数据到达触发水平 RXTRGLV (SCn_CTL[7:6]) 中断状态标志 0 = 接收缓存的数量小于 RXTRGLV 的设置. 1 = 接收缓存的数量达到 RXTRGLV 的设置. <b>注:</b> 该位只读. 如果用户从 SCn_DAT 读取数据且接收缓存数据量小于 RXTRGLV，该位自动清零</p>

**SC\_STATUS SC 传输状态寄存器**

寄存器	偏移量	R/W	描述	复位值
SC_STATUS	SCn_BA+0x20	R/W	SC 传输状态寄存器	0x0000_X202

31	30	29	28	27	26	25	24
TXACT	TXOVERR	TXRERR	Reserved		TXPOINT		
23	22	21	20	19	18	17	16
RXACT	RXOVERR	RXRERR	Reserved		RXPOINT		
15	14	13	12	11	10	9	8
Reserved		CDPINSTS	CINSERT	CREMOVE	TXFULL	TXEMPTY	TXOV
7	6	5	4	3	2	1	0
Reserved	BEF	FEF	PEF	Reserved	RXFULL	RXEMPTY	RXOV

位	描述
[31]	<b>TXACT</b>  发送激活状态标志 (只读) 该位表示 Tx 发送的状态 0 = 当 TX 传输完成或者最后一个字节传输已经完成，则该位自动清除 1 = 当 TX 传输处于激活状态而且最后字节的 STOP 位已经被发送，硬件将该位置位。 注：该位只读。
[30]	<b>TXOVERR</b>  发送重传错误超标 该位用来表示发送重传次数超过限定的重传次数 0 = 发送重传次数少于 TXRTY (SCn_CTL[22:20]) + 1. 1 = 发送重传次数等于或超过 TXRTY (SCn_CTL[22:20]) + 1. 注：该位写 1 清 0
[29]	<b>TXRERR</b>  发送重传错误 该位表示出现错误需要重传，由硬件自动置位 0 = 无 Tx 重传 1 = Tx 发生错误需要重传 注 1：该位写 1 清 0 注 2：该位仅仅是个标志，不能触发中断
[28:27]	<b>Reserved</b> 保留
[26:24]	<b>TXPOINT</b>  发送缓存指针状态 (只读) 该域指示 TX 缓存指针状态标志。当 CPU 写数据到 SCn_DAT，TXPOINT 增加 1。当 TX 缓存的一个字节被传输到发送器移位寄存器时，TXPOINT 减少 1。
[23]	<b>RXACT</b>  接收激活状态标志 (只读) 该位表示 Rx 接收的状态 0 = 当 RX 完成接收后，该位自动清除 1 = 当 RX 传输处于激活状态，硬件将该位置位。

		注: 该位只读.
[22]	<b>RXOVERR</b>	<p><b>接收重收错误超标</b>            该位用来表示接收重收次数超过限定的重收次数            0 = 接收重收次数少于 RXRTY (SCn_CTL[18:16]) + 1.            1 = 接收重收次数等于或超过 RXRTY (SCn_CTL[18:16]) + 1.  <b>注 1:</b> 该位写 1 清 0  <b>注 2:</b> 如果通过 RXRTYEN (SCn_CTL[19]) 使能了接收器重收功能, 硬件将不会置位该位</p>
[21]	<b>RXRERR</b>	<p><b>接收器重收错误</b>            该位表示出现错误需要重收, 由硬件自动置位            0 = 无 Rx 重收            1 = Rx 发生错误需要重收  <b>注 1:</b> 该位写 1 清 0  <b>注 2:</b> 该位仅仅是个标志, 不能触发中断  <b>注 3:</b> 如果通过 RXRTYEN (SCn_CTL[19]) 使能了接收器重收功能, 硬件将不会置位该位</p>
[20:19]	<b>Reserved</b>	保留
[18:16]	<b>RXPOINT</b>	<p><b>接收器缓存指针状态标志 (只读)</b>            该域指示 RX 缓存指针状态标志。当 SC 从外部设备收到一个字节, 寄存器 RXPOINT 加 1。当 RX 缓存被 CPU 读取了一个字节, 寄存器 RXPOINT 减 1。</p>
[15:14]	<b>Reserved</b>	保留
[13]	<b>CDPINSTS</b>	<p><b>SCn_CD 管脚上的卡检测状态 (只读)</b>            该位为 SCn_CD 管脚状态的标志            0 = SCn_CD 管脚状态为低            1 = SCn_CD 管脚状态为高</p>
[12]	<b>CINSERT</b>	<p><b>SCn_CD 管脚上的卡片插入状态</b>            该位表示 SCn_CD 管脚上是否有智能卡插入            0 = 无效            1 = 智能卡插入  <b>注 1:</b> 该位写 1 清 0.  <b>注 2:</b> SCEN (SCn_CTL[0]) 置位后, 智能卡插入检测功能开始工作</p>
[11]	<b>CREMOVE</b>	<p><b>SCn_CD 管脚上的卡片移除状态</b>            该位表示 SCn_CD 管脚上是否有智能卡移除            0 = 无效            1 = 智能卡移除  <b>注 1:</b> 该位写 1 清 0.  <b>注 2:</b> SCEN (SCn_CTL[0]) 置位后, 智能卡移除检测功能开始工作</p>
[10]	<b>TXFULL</b>	<p><b>发送缓存满状态标志 (只读)</b>            该位表示 Tx 缓存是否满            0 = Tx 缓存数小于 4            1 = Tx 缓存数等于 4</p>
[9]	<b>TXEMPTY</b>	<p><b>发送缓存空状态标志 (只读)</b>            该位表示发送缓存是否为空            0 = Tx 缓存不为空</p>

		<p>1 = Tx 发送缓存为空, 即 Tx 缓存里的最后一个字节都被送到了发送移位寄存器里了 注: 写数据到 DAT (SCn_DAT[7:0]) 清 0 该位</p>
[8]	<b>TXOV</b>	<p>发送溢出错误中断状态标志 Tx 缓存溢出时该位置位 0 = Tx 缓存未溢出 1 = Tx 缓存已满且 DAT (SCn_DAT[7:0]) 上有新的值写入, Tx 缓存溢出 注: 该位写 1 清 0</p>
[7]	<b>Reserved</b>	保留
[6]	<b>BEF</b>	<p>接收器断开错误状态标志 每当接收到的输入数据 (RX) 保持在“空状态”(逻辑 0) 状态的时间长过一个完整的字节传输时间 (也就是“开始位”+ 数据位 + 校验位 + 停止位的总时间), 该位被设置为逻辑 1。 0 = 接收器断开错误未发生. 1 = 接收器断开错误已发生. 注 1: 该位写 1 清 0 注 2: 如果通过 RXRTYEN (SCn_CTL[19]) 使能了接收器重收功能, 硬件将不会置位该位</p>
[5]	<b>FEF</b>	<p>接收器帧错误状态标志 每当接收的字符没有一个有效的“停止位”(也就是紧跟在最后一位数据位或校验位后的停止位被检测为逻辑 0), 该位被置为逻辑 1。 0 = 接收器帧错误未发生. 1 = 接收器帧错误已发生. 注 1: 该位写 1 清 0 注 2: 如果通过 RXRTYEN (SCn_CTL[19]) 使能了接收器重收功能, 硬件将不会置位该位</p>
[4]	<b>PEF</b>	<p>接收器校验错误状态标志 接收到的数据位校验位错误时该位置 1 0 = 接收器校验错误未发生 1 = 接收器校验错误已发生 注 1: 该位写 1 清 0 注 2: 如果通过 RXRTYEN (SCn_CTL[19]) 使能了接收器重收功能, 硬件将不会置位该位</p>
[3]	<b>Reserved</b>	保留
[2]	<b>RXFULL</b>	<p>接收器缓存满状态标志 (只读) 该位表示 Rx 缓存是否满 0 = Rx 缓存数小于 4 1 = Rx 缓存数等于 4</p>
[1]	<b>RXEMPTY</b>	<p>接收器缓存空状态标志 (只读) 该位表示 Rx 缓存是否为空 0 = Rx 缓存不为空 1 = Rx 缓存为空, 即 CPU 已经读取了 DAT (SCn_DAT[7:0]) 里的最后一个字节</p>
[0]	<b>RXOV</b>	<p>接收器缓存溢出错误状态标志 接收器缓存溢出时该位置位 0 = Rx 缓存未溢出. 1 = 接收到的字节数超过了 Rx 的缓存 (4 字节), Rx 缓存溢出 注: 该位写 1 清 0</p>



**SC\_PINCTL SC 管脚控制状态寄存器**

寄存器	偏移量	R/W	描述	复位值
SC_PINCTL	SCn_BA+0x24	R/W	SC 管脚控制状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved	SYNC	Reserved					
23	22	21	20	19	18	17	16
Reserved					RSTSTS	PWRSTS	DATASTS
15	14	13	12	11	10	9	8
Reserved				PWRINV	Reserved	SCDATA	Reserved
7	6	5	4	3	2	1	0
Reserved	CLKKEEP	Reserved				RSTEN	PWREN

位	描述	
[31]	Reserved	保留
[30]	SYNC	<b>SYNC 标志指示 (只读)</b> 用于同步，软件在写新的值到 SCn_PINCTL 之前必须检查该位 0 = 同步已经完成，用户可以写入新数据给 SCn_PINCTL 1 = 最后值在同步中。
[29:19]	Reserved	保留
[18]	RSTSTS	<b>SCn_RST 管脚状态 (只读)</b> 该位为 SCn_RST 的管脚状态 0 = SCn_RST 管脚为低电平。 1 = SCn_RST 管脚为高电平。
[17]	PWRSTS	<b>SCn_PWR 管脚状态 (只读)</b> 该位为 SCn_PWR 的管脚状态 0 = SCn_PWR 管脚为低电平。 1 = SCn_PWR 管脚为高电平。
[16]	DATASTS	<b>SCn_DATA 管脚状态 (只读)</b> 该位为 SCn_DATA 的管脚状态 0 = The SCn_DATA 管脚为低电平。 1 = The SCn_DATA 管脚为高电平。
[15:12]	Reserved	保留
[11]	PWRINV	<b>SC_POW 管脚反向</b> 该位用于 SC_PWR 管脚反向 通过设置寄存器 PWRINV (SC_PINCTL[11]) 和 PWREN (SC_PINCTL[0]), SC_POW 管脚的设置

		<p>有 4 种组合。PWRINV (SC_PINCTL[11]) 和 PWREN (SC_PINCTL[0]) 的组合为 SC_PWR 管脚提供高或低电压选择。</p> <p>00 = SC_PWR 为 0.      01 = SC_PWR 为 1.      10 = SC_PWR 为 1.      11 = SC_PWR 为 0.</p> <p><b>注:</b> 在通过寄存器 SC_CTL[0] 使能智能卡之前，软件必须设置好寄存器 PWRINV (SC_PINCTL[11])。</p>
[10]	<b>Reserved</b>	保留
[9]	<b>SCDATA</b>	<p><b>SCn_DATA 管脚信号</b>      该位为 SCn_DATA 管脚的信号状态，用户也可以修改该位来驱动 SCn_DATA 上的电平      0 = 驱动 SCn_DATA 管脚为低      1 = 驱动 SCn_DATA 管脚为高      读该寄存器获得 SCn_DATA 的信号状态。      0 = SCn_DATA 为低      1 = SCn_DATA 为高  <b>注:</b> 当 SC 处于激活，暖复位或者释放模式时，该位将被自动改变，所以 SC 在这些模式下，不要设置该位。</p>
[8:7]	<b>Reserved</b>	保留
[6]	<b>CLKKEEP</b>	<p><b>SC 使能使能位</b>      0 = 禁止 SC 时钟发生器      1 = SC 时钟发生器正常运行  <b>注:</b> 当 SC 处于激活，暖复位或者释放模式时，该位将会自动改变。所以工作在这些模式时，不要填充该位。</p>
[5:2]	<b>Reserved</b>	保留
[1]	<b>RSTEN</b>	<p><b>SC_RST 管脚信号</b>      该位是 SC_RST 的管脚状态，但是用户可以通过设置 RSTEN (SCn_PINCTL[1]) 驱动 SC_RST 管脚为高或低      写该域来驱动 SC_RST 管脚      0 = 驱动 SC_RST 管脚为低.      1 = 驱动 SC_RST 管脚为高.      读该域来获取 SC_RST 管脚状态。      0 = SC_RST 管脚状态为低.      1 = SC_RST 管脚状态为高.  <b>注:</b> 当 SC 处于激活，暖复位或释放模式时，该位会自动改变。所以当工作在这些模式时，不要填充该位。</p>
[0]	<b>PWREN</b>	<p><b>SC_PWREN 管脚信号</b>      软件可以通过设置寄存器 PWRINV (SCn_PINCTL[11]) 和 PWREN (SCn_PINCTL[0]) 来决定 SC_PWR 管脚为高电平或者低电平      写该域以驱动 SC_PWR 管脚      编程 SC_PWR 管脚电平可参考寄存器 PWRINV (SC_PINCTL[11]) 的描述      读该域来获取 SC_PWR 管脚状态      0 = SC_PWR 管脚状态为低      1 = SC_PWR 管脚状态为高.  <b>注:</b> 当 SC 处于激活，暖复位或释放模式时，该位会自动改变。所以当操作在这些模式时，不要填充</p>

		该位。
--	--	-----

**SC\_TMRCTL0 SC 定时器 0 控制寄存器**

寄存器	偏移量	R/W	描述	复位值
SC_TMRCTL0	SCn_BA+0x28	R/W	SC 内部定时器 0 寄存器	0x0000_0000

31	30	29	28	27	26	25	24
SYNC	Reserved			OPMODE			
23	22	21	20	19	18	17	16
CNT							
15	14	13	12	11	10	9	8
CNT							
7	6	5	4	3	2	1	0
CNT							

位	描述	
[31]	<b>SYNC</b>	<b>SYNC 标志指示 (只读)</b> 用于同步，软件在写新的值到 SCn_TMRCTL0 之前必须检查该位 0 = 同步已经完成，用户可以写入新数据给 SCn_TMRCTL0 1 = 最后值在同步中。
[30:28]	<b>Reserved</b>	保留
[27:24]	<b>OPMODE</b>	<b>Timer0 工作模式选择</b> 该域用于选择内部 24 位定时器 Timer0 的工作模式 Timer0 的配置方法请参考表 7.17 3
[23:0]	<b>CNT</b>	<b>Timer0 计数值</b> 该域为 Timer0 计数值 <b>注:</b> Timer0 的计数单位基于 ETU.

**SC\_TMRCTL1 SC 定时器 1 控制寄存器**

寄存器	偏移量	R/W	描述	复位值
SC_TMRCTL1	SCn_BA+0x2C	R/W	SC 内部定时器 1 寄存器	0x0000_0000

31	30	29	28	27	26	25	24
SYNC	Reserved			OPMODE			
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
CNT							

位	描述	
[31]	<b>SYNC</b>	<b>SYNC 标志指示 (只读)</b> 用于同步，软件在写新的值到 SCn_TMRCTL1 之前必须检查该位 0 = 同步已经完成，用户可以写入新数据给 SCn_TMRCTL1 1 = 最后值在同步中。
[30:28]	<b>Reserved</b>	保留
[27:24]	<b>OPMODE</b>	<b>Timer 1 工作模式选择</b> 该域用于选择内部 8 位定时器 Timer1 的工作模式 Timer1 的配置方法请参考表 7.17 3
[23:8]	<b>Reserved</b>	保留
[7:0]	<b>CNT</b>	<b>Timer 1 计数值</b> 该域为 Timer1 计数值 注: Timer1 的计数单位基于 ETU.

SC\_TMRCTL2 SC 定时器 2 控制寄存器

寄存器	偏移量	R/W	描述	复位值
SC_TMRCTL2	SCn_BA+0x30	R/W	SC 内部定时器 2 寄存器	0x0000_0000

31	30	29	28	27	26	25	24
SYNC	Reserved			OPMODE			
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
CNT							

位	描述	
[31]	<b>SYNC</b>	<b>SYNC 标志指示 (只读)</b> 用于同步，软件在写新的值到 SCn_TMRCTL2 之前必须检查该位 0 = 同步已经完成，用户可以写入新数据给 SCn_TMRCTL2 1 = 最后值在同步中.
[30:28]	<b>Reserved</b>	保留
[27:24]	<b>OPMODE</b>	<b>Timer 2 工作模式选择</b> 该域用于选择内部 8 位定时器 Timer2 的工作模式 Timer2 的配置方法请参考表 7.17 3
[23:8]	<b>Reserved</b>	保留
[7:0]	<b>CNT</b>	<b>Timer 2 计数值</b> 该域为 Timer2 计数值 <b>注:</b> Timer2 的计数单位基于 ETU.

**SC\_UARTCTL SC UART 模式 控制寄存器**

寄存器	偏移量	R/W	描述	复位值
SC_UARTCTL	SCn_BA+0x34	R/W	SC UART 模式控制寄存器	0x0000_0000

位	描述	
[31:8]	<b>Reserved</b>	保留
[7]	<b>OPE</b>	<p><b>奇检验使能位</b>            该位用于奇/偶校验的选择            0 = 奇数个逻辑 1 被发送, 或者在接收模式下检查数据字和校验位            1 = 偶数个逻辑 1 被发送, 或者在接收模式下检查数据字或校验位  <b>注意:</b> 该位仅在 PBOFF 位为 0 时起作用</p>
[6]	<b>PBOFF</b>	<p><b>校验位禁止位</b>            该位用于关闭校验功能            0 = 校验位产生或者在串行数据的“最后一个数据位”和“停止位”之间被检查.            1 = 校验位不产生 (发送数据) 或者在传输过程中不被检查 (接收数据)  <b>注意:</b> 当工作在智能卡模式时, 该域必须设置为 0 (默认设置带校验位)</p>
[5:4]	<b>WLS</b>	<p><b>数据长度</b>            该位用来选择 UART 数据长度            00 = 字符数据长度为 8 位.            01 = 字符数据长度为 7 位.            10 = 字符数据长度为 6 位.            11 = 字符数据长度为 5 位.  <b>注意:</b> 当工作在智能卡模式时, WLS 必须设置为 00。</p>
[3:1]	<b>Reserved</b>	保留
[0]	<b>UARTEN</b>	<p><b>UART 模式 使能位</b>            该位用来使能 UART 模式            0 = 智能卡模式            1 = 串口模式  <b>注 1:</b> 当工作于串口模式, 用户必须设置 CONSEL (SC_CTL[5:4]) = 00 和 AUTOCEN (SC_CTL[3]) = 0。  <b>注 2:</b> 当工作在智能卡模式时, 用户必须设置寄存器 UARTEN (SC_UARTCTL [0]) = 00。  <b>注 3:</b> 当串口功能被使能时, 硬件将产生一个复位信号来复位内部缓存和内部状态机。</p>

**SC\_ACTCTL SC 激活控制寄存器**

寄存器	偏移量	R/W	描述	复位值
SC_ACTCTL	SCn_BA+0x4C	R/W	SC 激活控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved			T1EXT				

位	描述	
[31:5]	<b>Reserved</b>	保留
[4:0]	<b>T1EXT</b>	<p><b>T1 硬件激活的扩展时间</b>          该域用来配置硬件激活的扩展时间周期          该周期的放大因数是 2048          扩展周期 = (寄存器值 * 2048) 个周期.          参考 SC 激活序列举例：          SCLK = 4MHz, 每个周期为 0.25us.,          设该寄存器为 20          扩展周期 = 20 * 2048 * 0.25us = 10.24 ms.  <b>注:</b> 设该寄存器为 0 也符合 ISO/IEC 7816-3 协议</p>

## 6.18 I<sup>2</sup>S 控制器

### 6.18.1 概述

I<sup>2</sup>S 控制器提供符合 IIS 协议可外接语音 CODEC 接口。输入输出有独立的 16 字节深度的 FIFO，可以支持 8/16/24/32 位的数据宽度。PDMA 控制器可以在 FIFO 和内存间搬运数据。

### 6.18.2 特征

- 支持主机模式和从机模式
- 每路音频通道都支持 8, 16, 24 和 32 位的数据宽度
- 支持单声道和双声道的音频数据
- 支持 I<sup>2</sup>S 协议：飞利浦标准, MSB 对齐和 LSB 对齐的数据格式
- 支持 PCM 协议：PCM 标准, MSB 对齐和 LSB 对齐的数据格式
- PCM 协议支持在一个声音采样里 TDM 多通道传输，通道数可设为 2,4,6 或 8
- 提供 2 个 16 字节深度的数据缓存，分别为发送缓存和接收缓存
- 当缓存达到设定的阈值时可以产生中断请求到 CPU
- 支持 2 路 PDMA 请求，一路用于发送，另一路用于接收

### 6.18.3 框图

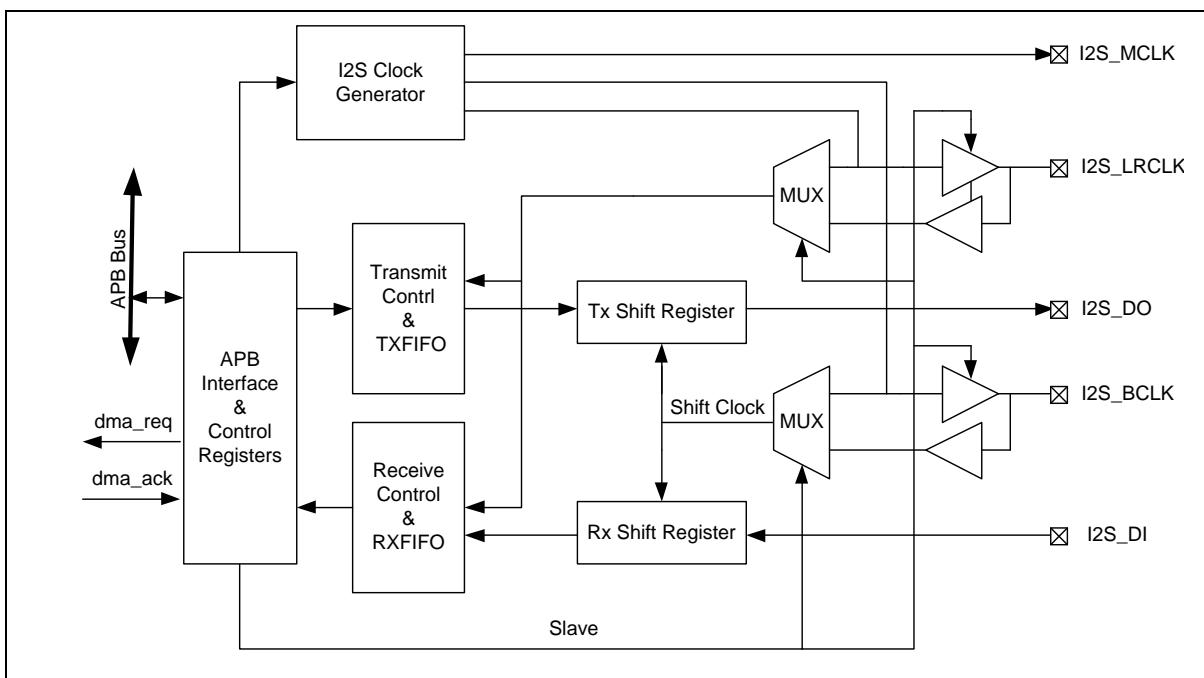


图 6.18-1 I<sup>2</sup>S 控制器框架

### 6.18.4 基本配置

#### 6.18.4.1 I<sup>2</sup>S 基本配置

- 时钟源配置

- 在寄存器 I2SSEL (CLK\_CLKSEL3[17:16]) 配置 I<sup>2</sup>S 的外设时钟源
- 在寄存器 I2SCKEN (CLK\_APBCLK0[29]) 使能 I<sup>2</sup>S 的外设时钟
- 复位配置
  - 在寄存器 I2SRST (SYS\_IPRST1[29]) 复位 I<sup>2</sup>S 控制器
- 管脚配置

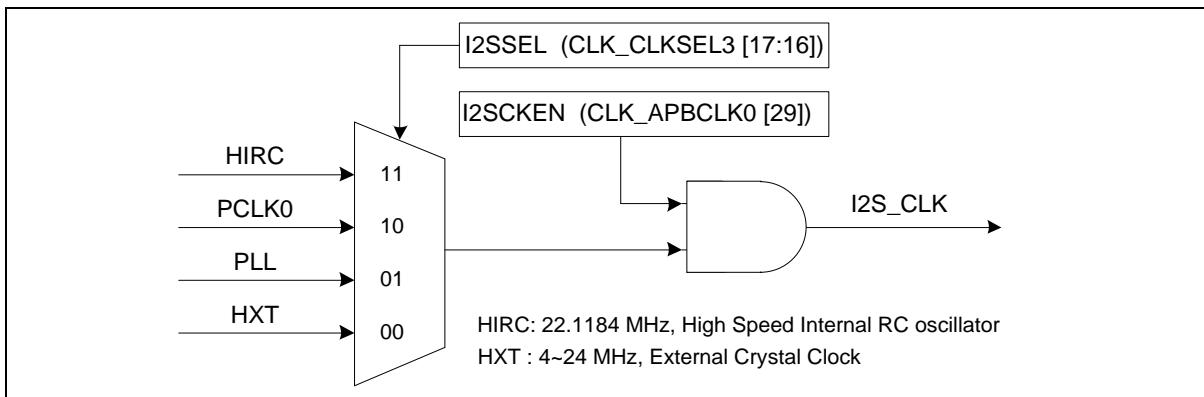
组	管脚名	GPIO	MFP
I2S0	I2S0_BCLK	PA.12	MFP2
		PE.8, PF.10	MFP4
		PE.1	MFP5
		PC.4	MFP6
		PB.5	MFP10
	I2S0_DI	PA.14	MFP2
		PE.10, PF.8	MFP4
		PH.8	MFP5
		PC.2	MFP6
		PB.3	MFP10
	I2S0_DO	PA.15	MFP2
		PE.11, PF.7	MFP4
		PH.9	MFP5
		PC.1	MFP6
		PB.2	MFP10
	I2S0_LRCK	PE.12, PF.6	MFP4
		PH.10	MFP5
		PC.0	MFP6
		PB.1	MFP10
	I2S0_MCLK	PA.13	MFP2
		PE.9, PF.9	MFP4
		PE.0	MFP5
		PC.3	MFP6
		PB.4	MFP10

表 6.18-2 I<sup>2</sup>C 控制器配置

### 6.18.5 功能描述

#### 6.18.5.1 I<sup>2</sup>S 时钟

在寄存器 I2SSEL (CLK\_CLKSEL3[17:16]) 里选择 I<sup>2</sup>S 的 4 个时钟源。I<sup>2</sup>S 的时钟频率不能大于系统时钟的频率。

图 6.18-3 I<sup>2</sup>S 时钟控制框图

#### 6.18.5.2 主机/从机接口

通过配置 SLAVE (I<sup>2</sup>S\_CTL0[8]) 寄存器可以切换 I<sup>2</sup>S 的主机或从机模式以连接其他的 I<sup>2</sup>S 从机或主机。串行总线的时钟 I<sup>2</sup>S\_BCLK 由主机产生且不管有无数据传输都会一直存在。字选择信号 I<sup>2</sup>S\_LRCLK 由主机产生，代表着新的数据字的开始和目标音频的通道。I<sup>2</sup>S\_LRCLK 和数据都在 I<sup>2</sup>S\_BCLK 下降沿同步改变。

在某些应用里，尤其是 音频-ADC 和 音频-DAC，主时钟信号 I<sup>2</sup>S\_MCLK 和 I<sup>2</sup>S\_BCLK 有一个固定的相位差。I<sup>2</sup>S\_MCLK 在寄存器 MCLKEN (I<sup>2</sup>S\_CTL0[15]) 里配置。主机模式下，主机输出 I<sup>2</sup>S\_MCLK, I<sup>2</sup>S\_BCLK, I<sup>2</sup>S\_LRCLK 到从机。从机模式下，I<sup>2</sup>S\_MCLK 输出到主机，I<sup>2</sup>S\_BCLK 或 I<sup>2</sup>S\_LRCLK 则由主机输入。

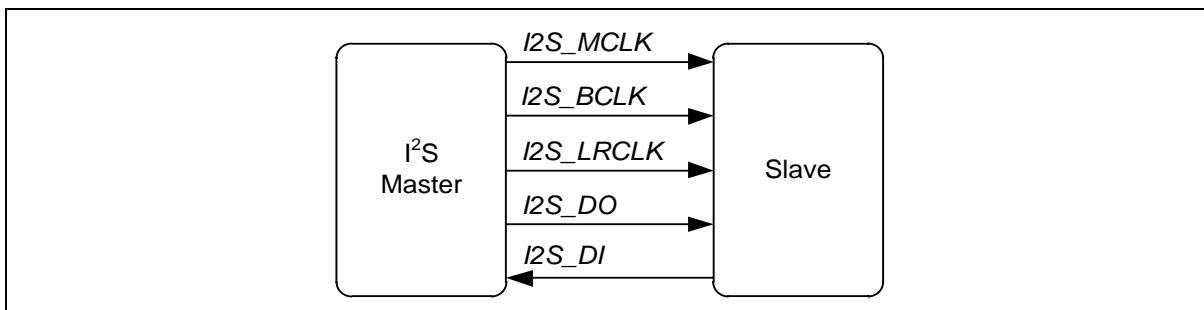


图 6.18-3 主机模式接口框图

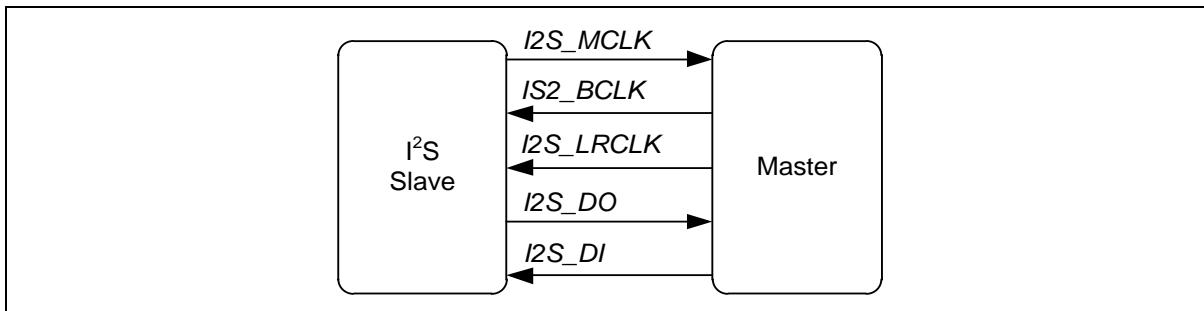


图 6.18-4 从机模式接口框图

### 6.18.5.3 I<sup>2</sup>S 操作

I<sup>2</sup>S 控制器支持飞利浦标准, MSB 对齐和 LSB 对齐的数据格式。I2S\_LRCLK 信号指示当前传输的音频通道。音频通道的位宽度寄存器为 CHWIDTH (I2S\_CTL0[29:28]), 数据字的位宽度寄存器为 DATWIDTH (I2S\_CTL0[5:4])。如果 CHWIDTH (I2S\_CTL0[29:28]) 小于 DATWIDTH (I2S\_CTL0[5:4]), 硬件会将通道的位宽度更成和数据位宽度一致。如果, CHWIDTH (I2S\_CTL0[29:28]) 大于 DATWIDTH (I2S\_CTL0[5:4]), 硬件则会填充零位到每个音频通道。

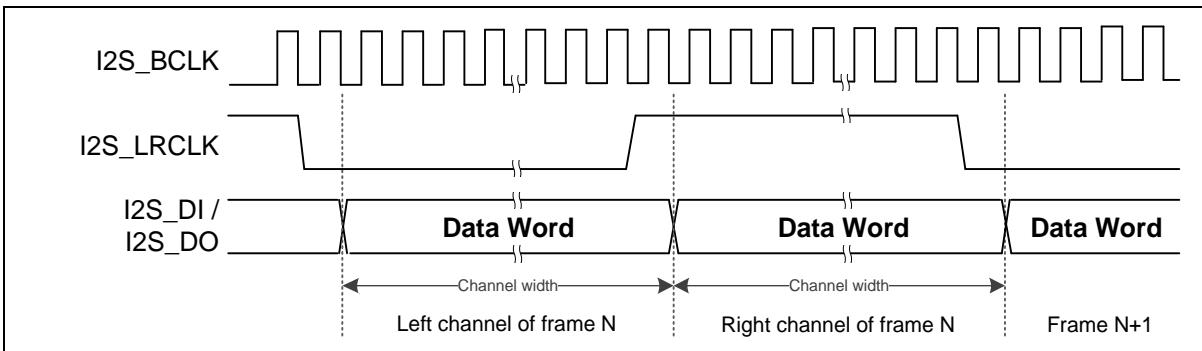


图 6.18-5 I<sup>2</sup>S 通道宽度和数据宽度 (CHWIDTH≤DATWIDTH)

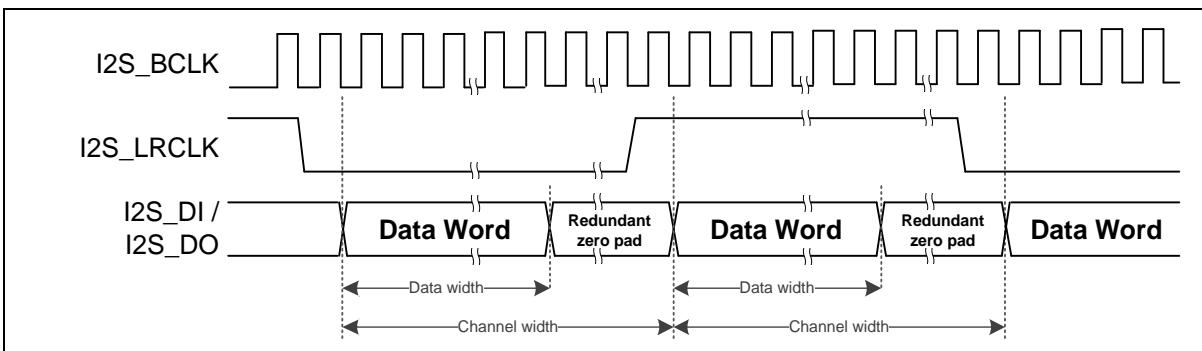
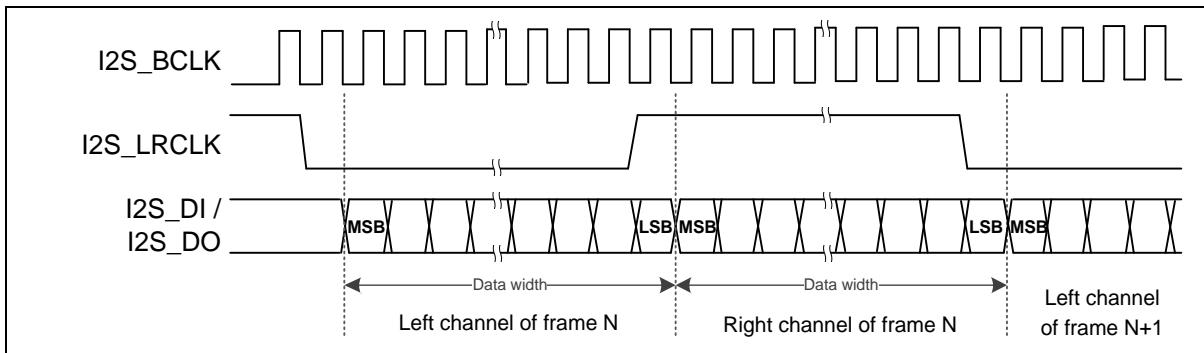


图 6.18-6 I<sup>2</sup>S 通道宽度和数据宽度 (CHWIDTH > DATWIDTH)

数据传输的序列一直是从 MSB 到 LSB。如图, 按 I<sup>2</sup>S 协议, 发送的数据在 I2S\_BCLK 的上升沿被读取, 在下降沿发送。在 I<sup>2</sup>S 的数据格式里, MSB 在 I2S\_LRCLK 变化后的下一个 I2S\_BCLK 周期的下降沿发送并锁存。在 MSB 对齐的数据格式里, I2S\_LRCLK 在每个音频通道的第一个数据位 (MSB) 发送时发生变化。在 LSB 对齐的数据格式里, I2S\_LRCLK 在每个音频通道的最后一个 I2S\_BCLK 周期发送并锁存。通过寄存器 FORMAT (I2S\_CTL0[26:24]) 可以切换 I<sup>2</sup>S 协议的 MSB 对齐和 LSB 对齐的数据格式。

图 6.18-7 I<sup>2</sup>S 数据格式时序图 (FORMAT = 0x0 ; CHWIDTH≤DATWIDTH)

如下图，数据的传输总是高位在前。数据在I2S\_BCLK 的上沿读入，在其下沿发出。I2S格式，第一位在I2S\_LRCLK 跳变后的下一个I2S\_BCLK时钟的下沿发出。I2S高位对齐格式，I2S\_LRCLK 跳变的同时发出最高位数据。I2S低位对齐格式，在最后一个时钟发出最低位。格式选择由 FORMAT (I2S\_CTL0[26:24])配置。

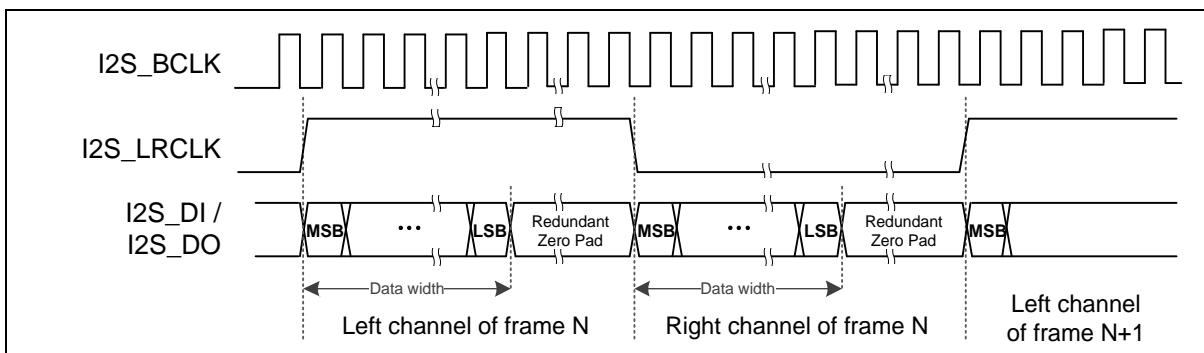


图 6.18-8 MSB 对齐数据格式 (FORMAT = 0x1 ; CHWIDTH &gt; DATWIDTH)

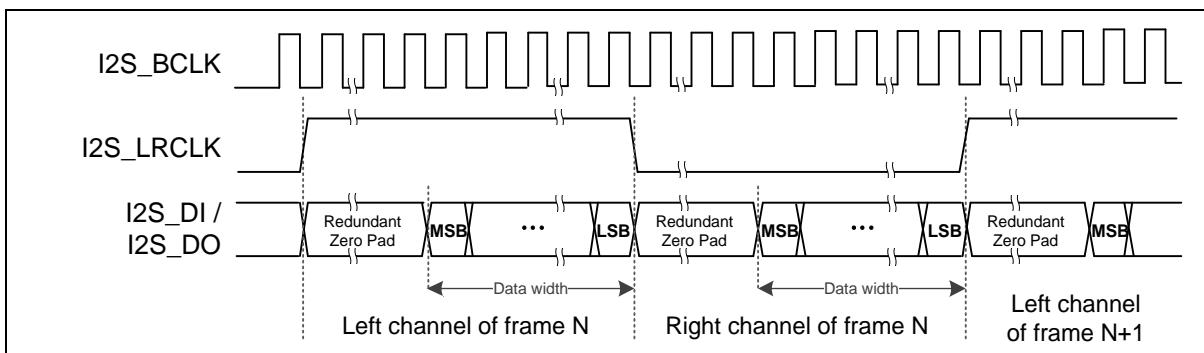


图 6.18-9 LSB 对齐数据格式 (FORMAT = 0x2 ; CHWIDTH &gt; DATWIDTH)

I<sup>2</sup>S 控制器还支持 PCM 格式的音频传输，对应的寄存器使能位为 FORMAT (I2S\_CTL0[26:24])。在 PCM 协议里，I2S\_LRCLK 只是以一个上升沿的脉冲来作为音频段（或音频帧）的开始。因此，I2S\_LRCLK 在 PCM 协议里也被称为“帧开始”或“帧同步”信号。在主机模式下，用户可以用 PCMSYNC (I2S\_CTL0[27]) 来选择 2 种常用的“帧开始”脉冲宽度，一个是采用和通道宽度一致的脉冲宽度，另一个是采用一个 I2S\_BCLK 周期的脉冲宽度。

和 I<sup>2</sup>S 模式一样，PCM 模式下，用户可以用 DATWIDTH (I2S\_CTL0[5:4]) 和 CHWIDTH

(I2S\_CTL0[29:28]) 选择不同的数据宽度和通道宽度。另外，通过 FORMAT (I2S\_CTL0[26:24]) 用户可以自由切换，PCM 标准，MSB 对齐和 LSB 对齐的数据格式三种数据格式。

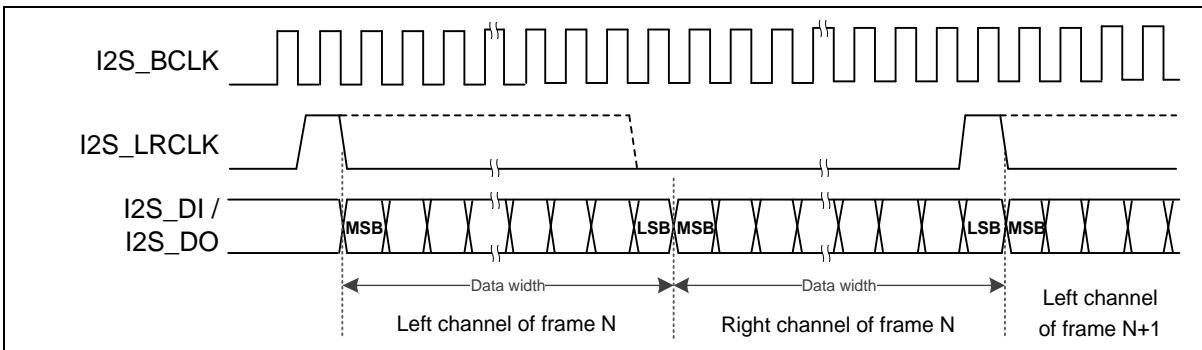


图 6.18-10 标准 PCM 音频时序图 (FORMAT = 0x4 ; CHWIDTH≤DATWIDTH)

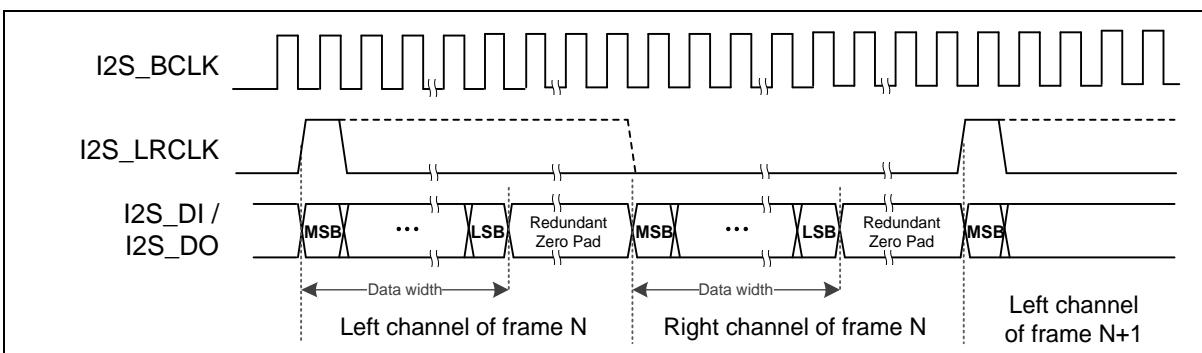


图 6.18-11 MSB 对齐的 PCM 数据格式 (FORMAT = 0x5 ; CHWIDTH > DATWIDTH)

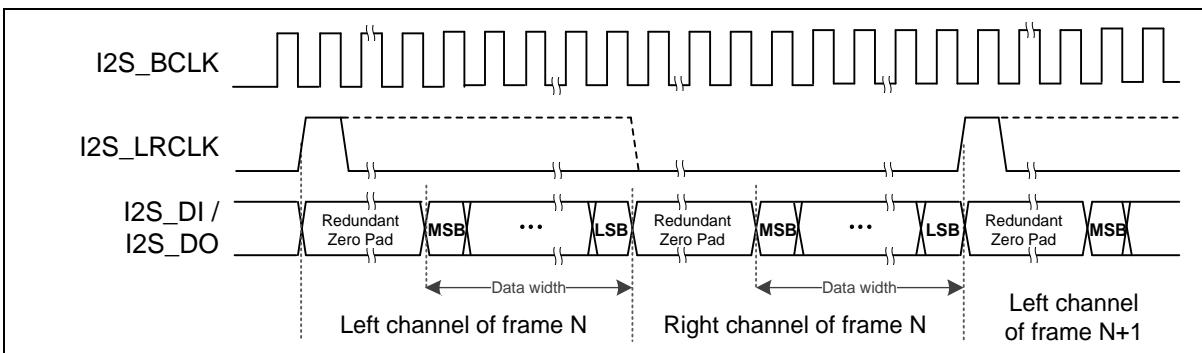


图 6.18-12 LSB 对齐的 PCM 数据格式 (FORMAT = 0x6 ; CHWIDTH > DATWIDTH)

#### 6.18.5.4 TDM 多通道传输

I<sup>2</sup>S 控制器处于 PCM 模式下时还支持 TDM 传输。分时多路复用 (TDM) 允许用户在一个数据线上上传输多个通道的音频数据。TDM 接口和 2 通道的 PCM 接口基本类似，不同的是它可以通过定义不同的 I2S\_LRCLK 周期实现在一个采样帧里传输多个音频通道。TDM 接口可以通过 TDMCHNUM (I2S\_CTL0[31:30]) 寄存器选择不同的通道数，典型值4个，6个或8个。

类似之前的 I<sup>2</sup>S 和 PCM 模式，每个通道块包含通过零位形成一个完整通道块的音频数据。用户可以用 DATWIDTH (I2S\_CTL0[5:4]) 和 CHWIDTH (I2S\_CTL0[29:28]) 选择不同的数据宽度和通道宽度。需要注意的是，使能了 TDM 的 PCM 模式支持 16 位，24 位和 32 位宽度的音频数据，不支持 8 位宽度。所以，如果 DATWIDTH (I2S\_CTL0[5:4]) 被设成了 0x0，硬件将会按 16 位宽度进行数据传输。帧开始信号的脉冲宽度则由 PCMSYNC (I2S\_CTL0[27]) 选择。

如图 6.18-13 是用 32 位通道块传输 6 通道 24 位音频数据的 TDM 传输示例。一般来说，对于 2 个通道的音频接口，我们称第一个为左声道，另一个为右声道（或者称为，通道 0 和通道 1）。在 TDM 多路传输里，我们称之为通道 0 和通道 1。

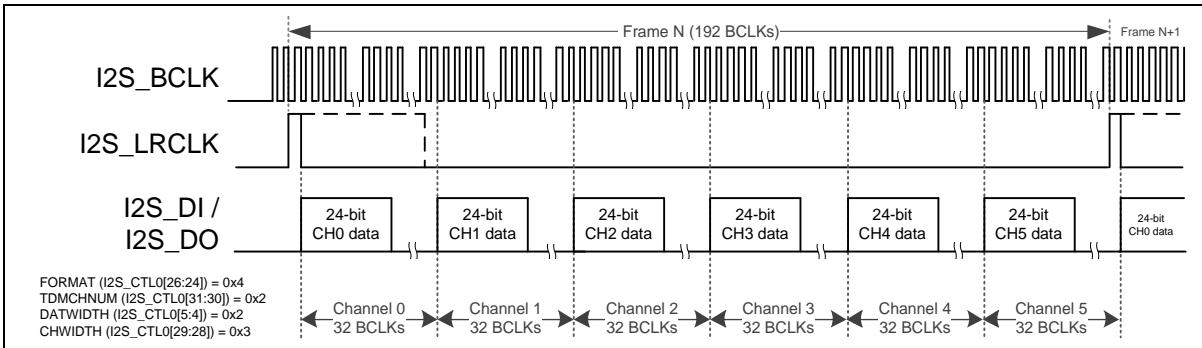


图 6.18-13 用 32 位通道块传输 6 通道 24 位音频数据的 TDM 传输 (PCM 标准模式; FORMAT=0x4)

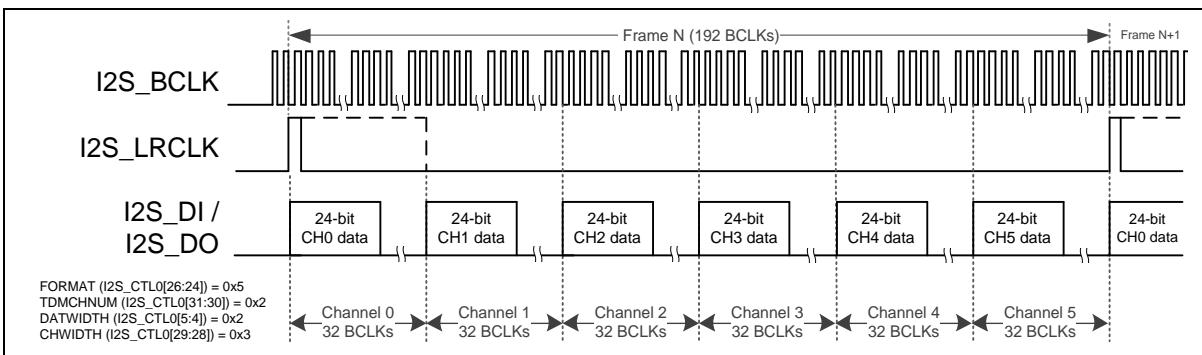


图 6.18-14 用 32 位通道块传输 6 通道 24 位音频数据的 TDM 传输 (PCM MSB 对齐; FORMAT=0x5)

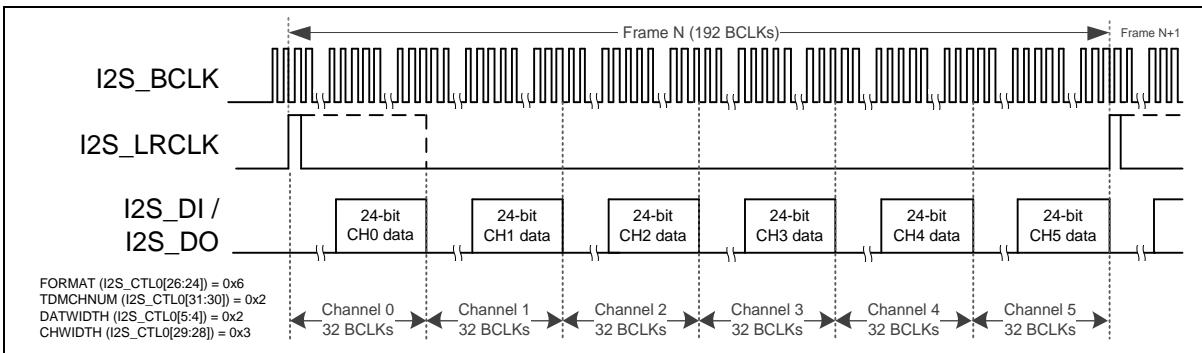


图 6.18-15 用 32 位通道块传输 6 通道 24 位音频数据的 TDM 传输 (PCM LSB 对齐; FORMAT=0x6)

### 6.18.5.5 过零检测

I2S 控制器播放音频时，通过 CPU 或 PDMA 把内存的数据发送出去。在播放过程中如果用户去切换增益时，很容易产生难听的爆破音。过零检测就是指在这个时候几乎关闭播放而避免产生噪音。过零检测中断可以用来指示增益的变化，从而避免声音产生很大的变化。

如果在寄存器 CH0ZCEN 到 CH7ZCEN (I2S\_CTL1[0] 到 I2S\_CTL1[7]) 使能了对应通道的过零检测功能，硬件去检测对应通道的下一个传输的数据是否是零或者数据 MSB 是否发生了变化。使能零检测功能后，硬件在检测到对应通道上的零值或 MSB 变化时，对应通道的状态位 CH0ZCIF 到 CH7ZCIF (I2S\_STATUS1[0] 到 I2S\_STATUS1[7]) 会被置位，且该通道会被自动静音（所有数据位都设为 0）直

到对应的状态位被软件清 0。

因此，如果用户想要避免爆破音，可以在修改音频增益前使能过零检测中断 CH0ZCIEN 到 CH7ZCIEN (I2S\_IEN[16:23])，然后在过零点去改变声音增益。.

#### 6.18.5.6 PDMA 模式

I<sup>2</sup>S 可以使能 PDMA 功能进行数据传输。如果使能了 TXPDMAEN (I2S\_CTL0[20]) 进行 PDMA 发送，I<sup>2</sup>S 控制器会产生一个请求信号，如果 TX FIFO 未满，就会通过 PDMA 接口自动获取内存里要发送的音频数据。如果使能了 RXPDMAEN (I2S\_CTL0[19]) 进行 PDMA 接收，I<sup>2</sup>S 控制器会产生一个请求信号，如果 RX FIFO 非空，就会通过 PDMA 接口自动把收到的音频数据搬到内存。因此，使用 PDMA 功能会减省 CPU 的时间用于其他功能。

#### 6.18.5.7 I<sup>2</sup>S 中断源

发送时，I<sup>2</sup>S 控制器支持独立音频通道的过零检测中断，发送 FIFO 阈值中断，发送 FIFO 溢出中断和发送 FIFO 下溢中断。接收时，则支持接收 FIFO 阈值中断，接收 FIFO 溢出中断和接收 FIFO 下溢中断。I<sup>2</sup>S 中断发生后，用户可以检查 I2STXINT (I2S\_STATUS0[2]) 和 I2SRXINT (I2S\_STATUS0[1]) 标志来识别中断源。

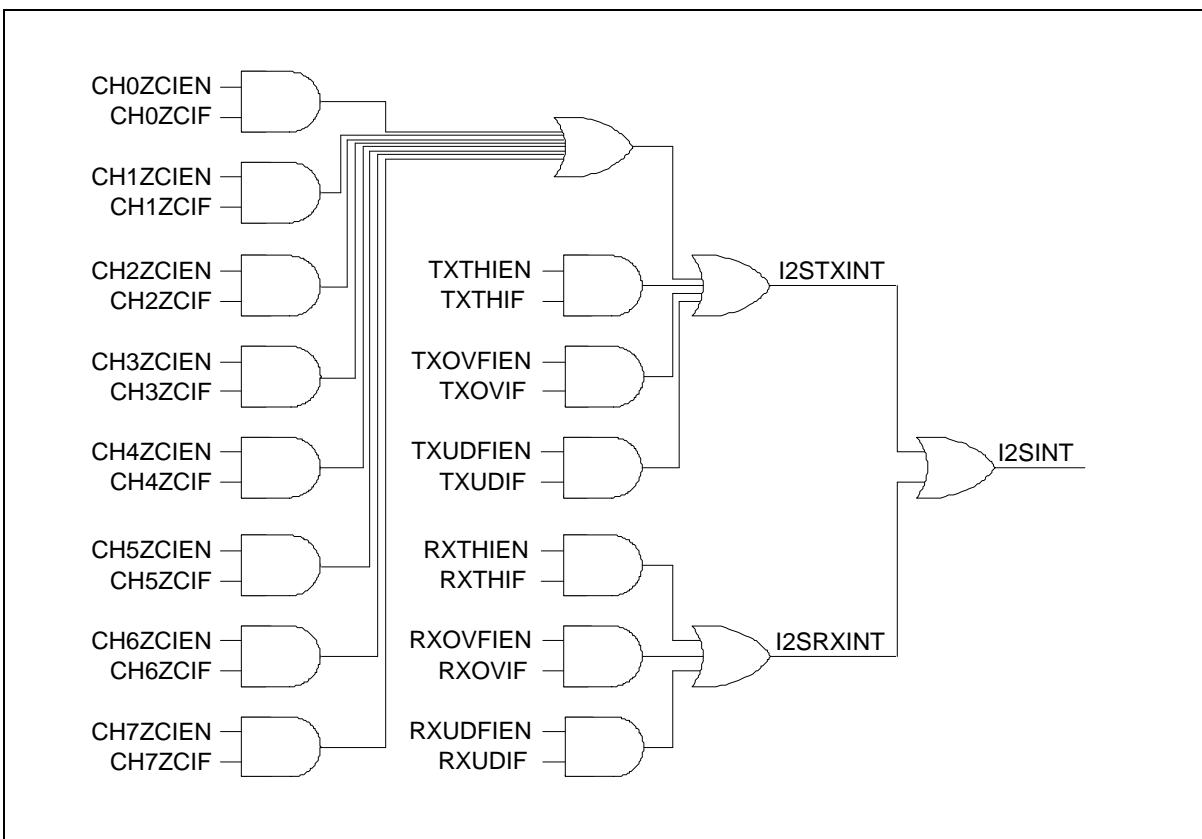


图 6.18-16 I<sup>2</sup>S 中断

#### 6.18.5.8 FIFO 操作

在 2 路通道的 I<sup>2</sup>S 或 PCM 模式里，一个通道块的音频数据的位宽可以是 8, 16, 24, 或者 32 位。如图 6.18-17 是不同设置下音频数据的内存管理。

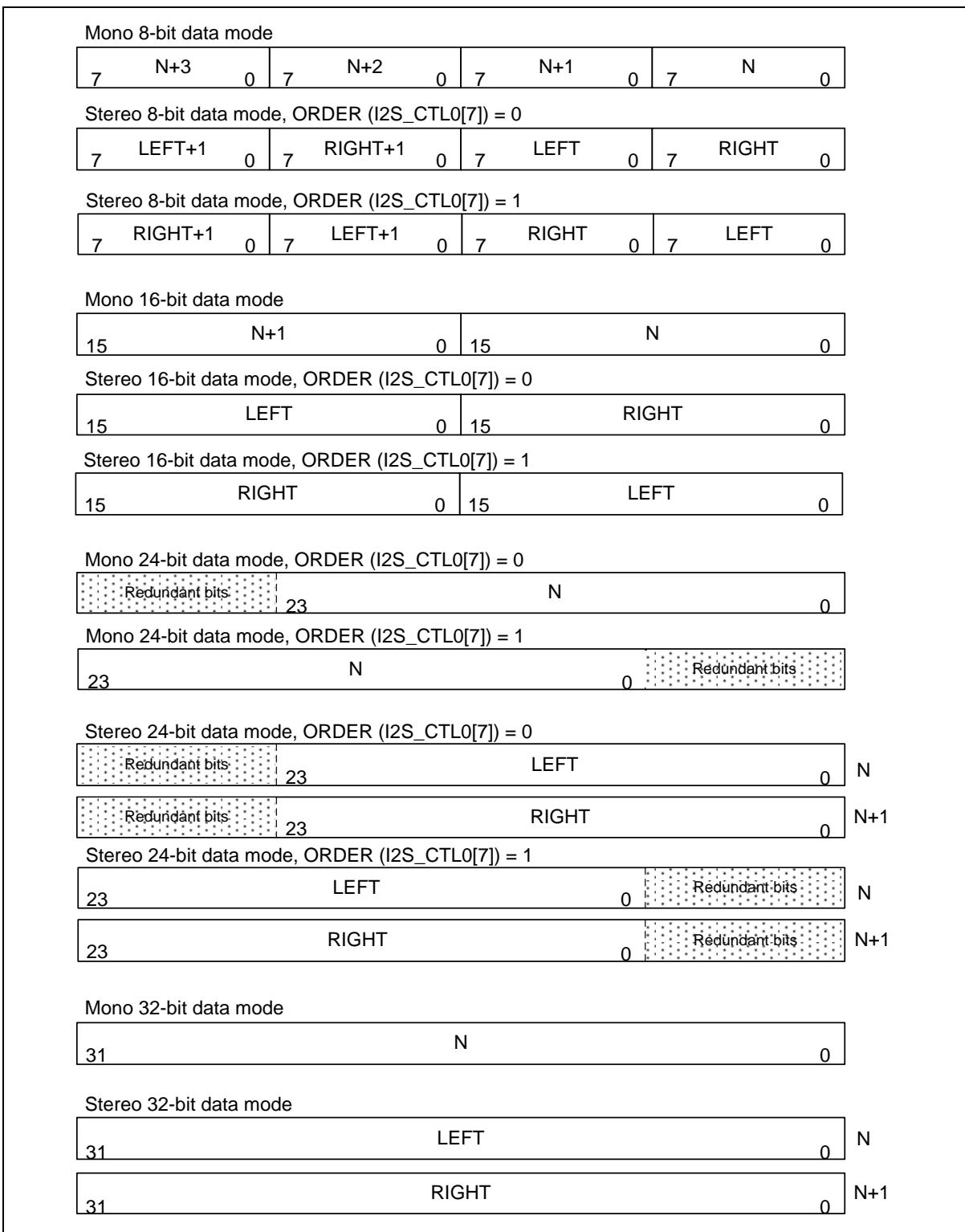


图 6.18-17 不同设置下 2 通道音频数据的 FIFO 示意

在 4 通道使能 TDM 的 PCM 模式里，一个通道块的音频数据的位宽可以是 16, 24, 或者 32 位。如图 6.18-18 是不同设置下音频数据的内存管理。

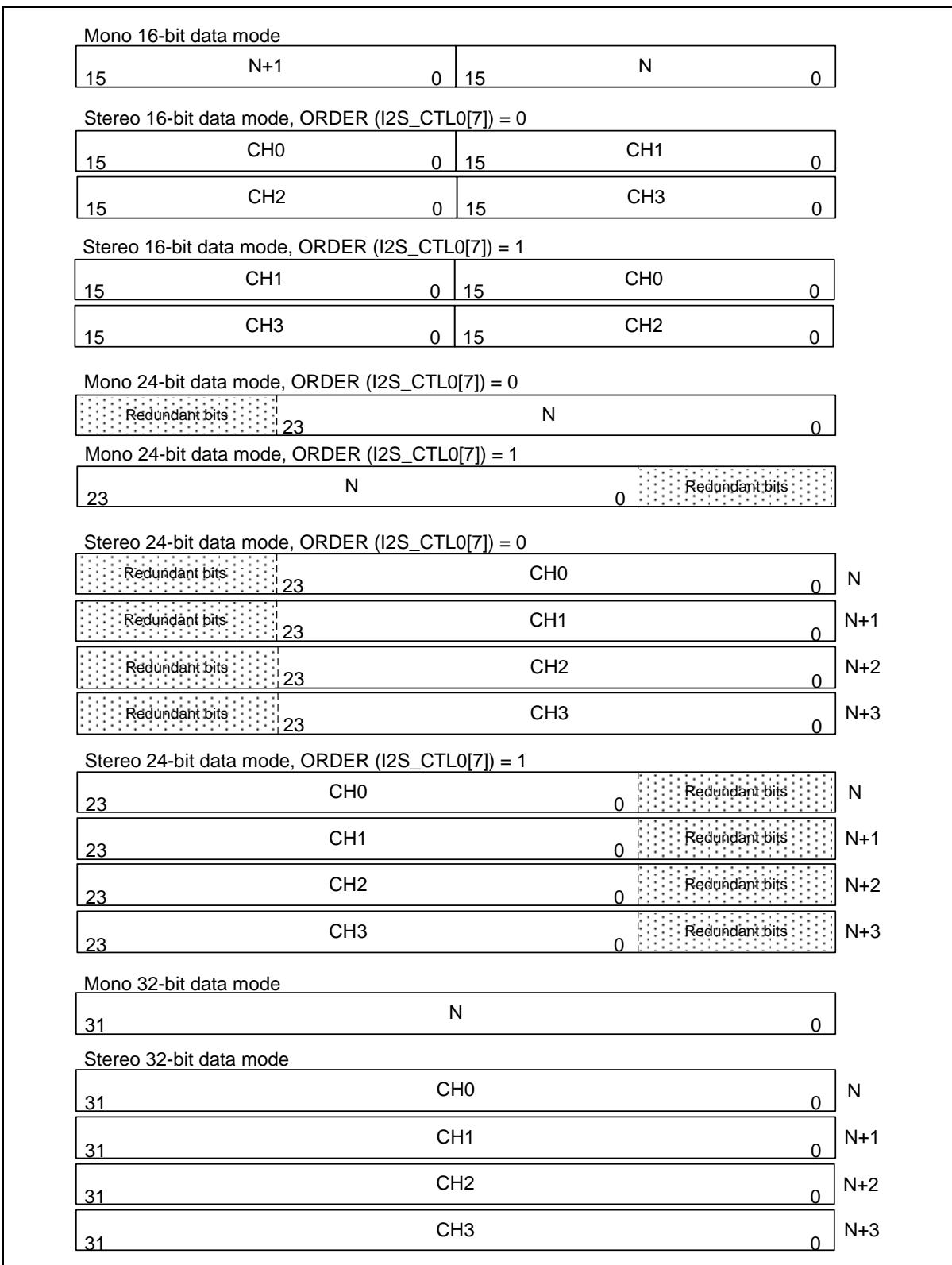


图 6.18-18 不同设置下 4 通道音频数据的 FIFO 示意

在 6 通道使能 TDM 的 PCM 模式里，一个通道块的音频数据的位宽可以是 16, 24, 或者 32 位。  
图 6.18-19 是不同设置下音频数据的内存管理。

16 位音频数据传输时，可以用 ORDER (I2S\_CTL0[7]) 交换存储在发送和接收 FIFO 里的奇数位通道和偶数位通道。在 24 位音频数据传输时，ORDER (I2S\_CTL0[7]) 可以用来选择存储在 32 位 FIFO 里的音频数据是左对齐还是右对齐。

8 通道使能 TDM 的 PCM 模式和 6 通道的情况很类似，这里就不再赘述。

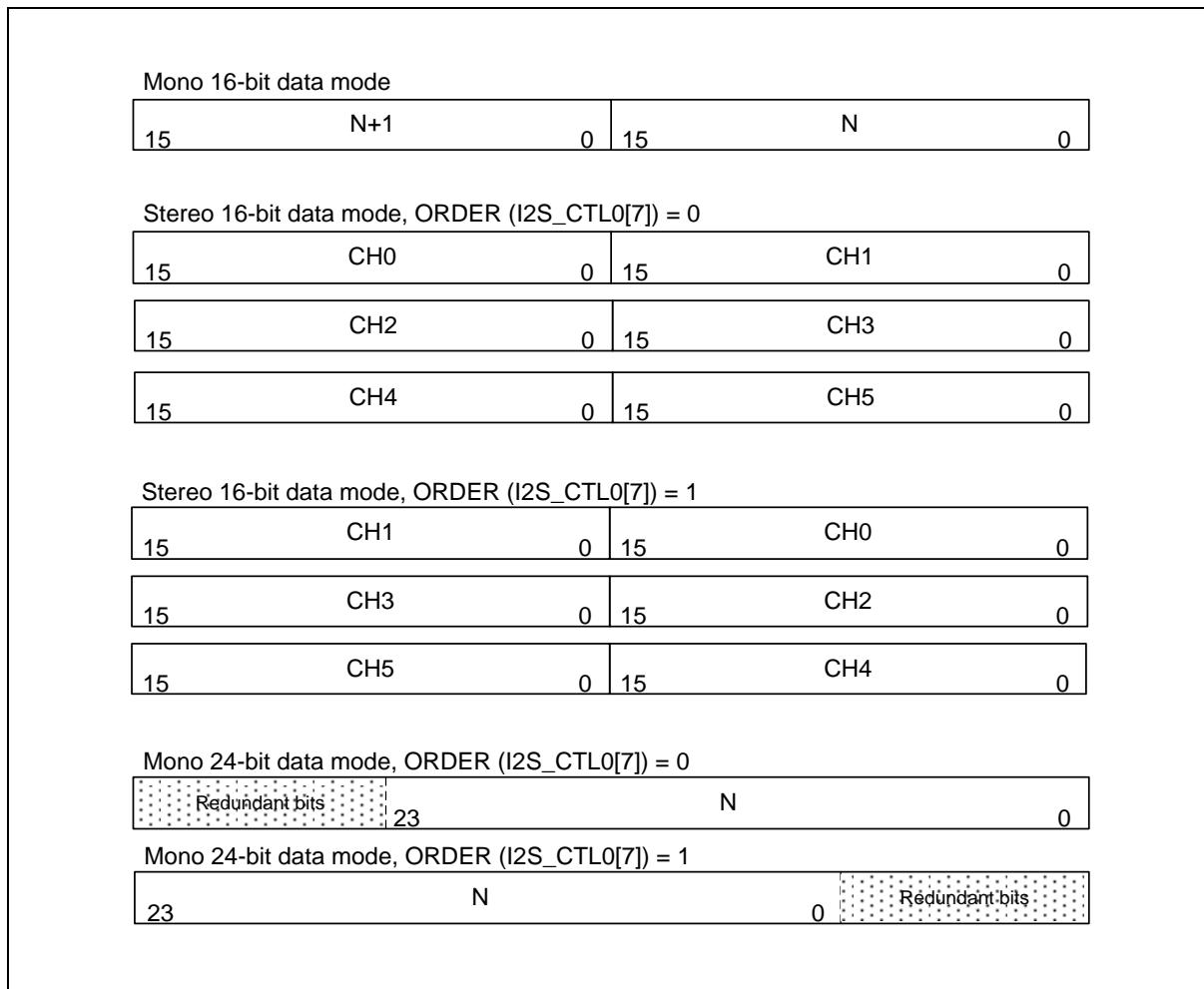


图 6.18-19 不同设置下 6 通道音频数据的 FIFO 示意 (1)

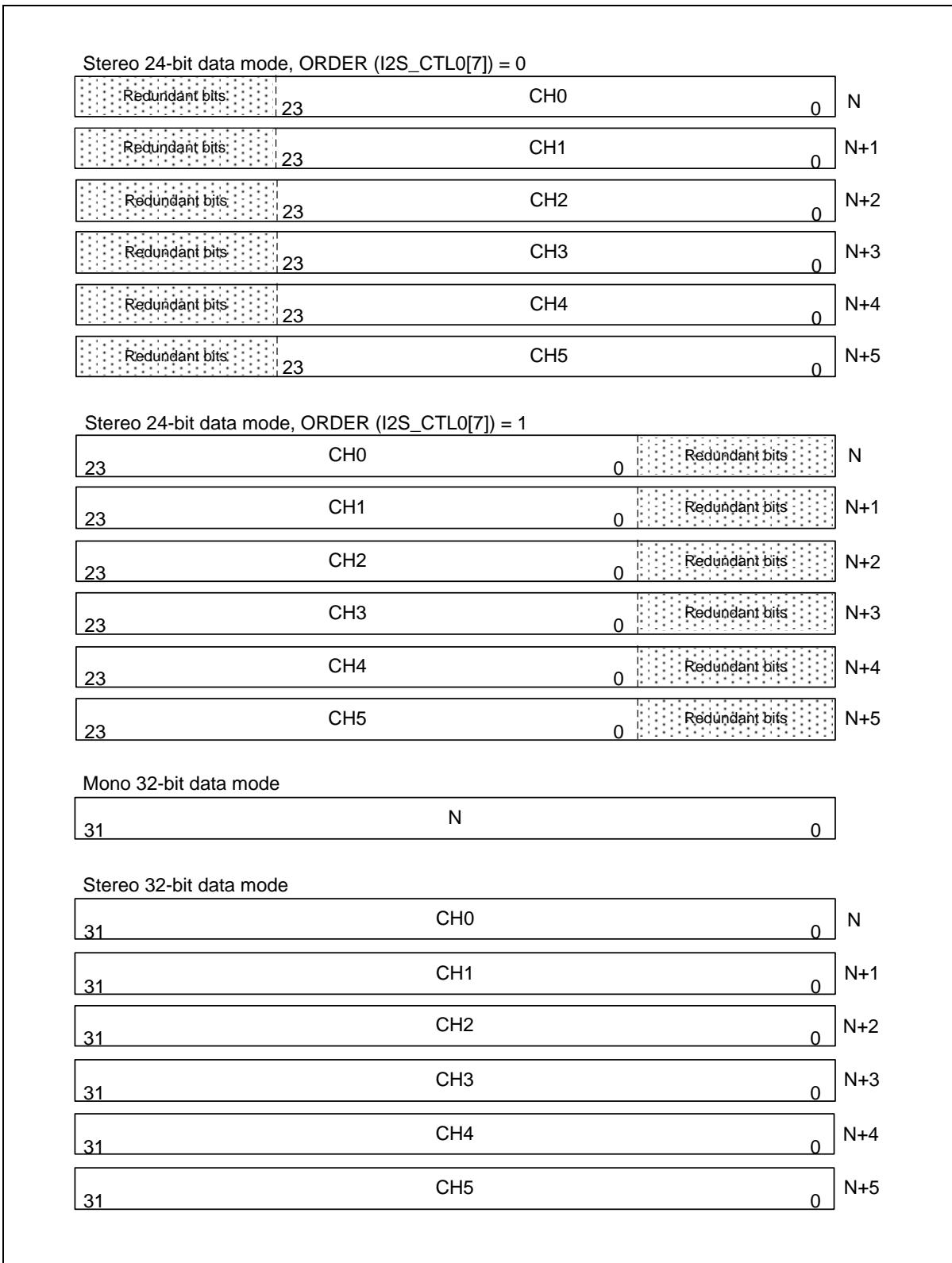


图 6.18-20 不同设置下 6 通道音频数据的 FIFO 示意 (2)

### 6.18.6 寄存器映射

R: 只读, W: 只写, R/W: 读/写

寄存器	偏移量	R/W	描述	复位值
<b>I2S 基址</b>				
<b>I2S_BA = 0x4004_8000</b>				
<b>I2S_CTL0</b>	I2S_BA+0x00	R/W	I <sup>2</sup> S 控制寄存器 0	0x0000_0100
<b>I2S_CTL1</b>	I2S_BA+0x20	R/W	I <sup>2</sup> S 控制寄存器 1	0x0000_0000
<b>I2S_CLKDIV</b>	I2S_BA+0x04	R/W	I <sup>2</sup> S 时钟分频器寄存器	0x0000_0000
<b>I2S_IEN</b>	I2S_BA+0x08	R/W	I <sup>2</sup> S 中断使能寄存器	0x0000_0000
<b>I2S_STATUS0</b>	I2S_BA+0x0C	R/W	I <sup>2</sup> S 状态寄存器 0	0x0014_1000
<b>I2S_STATUS1</b>	I2S_BA+0x24	R/W	I <sup>2</sup> S 状态寄存器 1	0x0000_0000
<b>I2S_TXFIFO</b>	I2S_BA+0x10	W	I <sup>2</sup> S 发送 FIFO 寄存器	0x0000_0000
<b>I2S_RXFIFO</b>	I2S_BA+0x14	R	I <sup>2</sup> S 接收 FIFO 寄存器	0x0000_0000

## 6.18.7 寄存器描述

I<sup>2</sup>S\_CTL0 I<sup>2</sup>S 控制寄存器 0

寄存器	偏移量	R/W	描述				复位值
I <sup>2</sup> S_CTL0	I <sup>2</sup> S_BA+0x00	R/W	I <sup>2</sup> S 控制寄存器 0				0x0000_0100

31	30	29	28	27	26	25	24
TDMCHNUM		CHWIDTH		PCMSYNC	FORMAT		
23	22	21	20	19	18	17	16
RXLCH	Reserved	RXPDMAEN	TXPDMAEN	RXFBCLR	TXFBCLR	Reserved	
15	14	13	12	11	10	9	8
MCLKEN	Reserved						SLAVE
7	6	5	4	3	2	1	0
ORDER	MONO	DATWIDTH		MUTE	RXEN	TXEN	I <sup>2</sup> SEN

位	描述	
[31:30]	TDMCHNUM	<b>TDM 通道数</b> 该位用来定义 PCM 模式 (FORMAT[2] = 1) 下，一个音频帧里的 TDM 通道数目 00 = 音频帧里有 2 个通道。 01 = 音频帧里有 4 个通道。 10 = 音频帧里有 6 个通道。 11 = 音频帧里有 8 个通道。
[29:28]	CHWIDTH	<b>通道宽度</b> 该位用来定义音频通道的宽度。如果 CHWIDTH (I <sup>2</sup> S_CTL0[29:28]) 小于 DATWIDTH (I <sup>2</sup> S_CTL0[5:4])，实际传输时，硬件会将通道的位宽度更成和数据位宽度一致 00 = 音频通道的位宽度为 8 位 01 = 音频通道的位宽度为 16 位 10 = 音频通道的位宽度为 24 位 11 = 音频通道的位宽度为 32 位
[27]	PCMSYNC	<b>PCM 同步脉冲长度选择</b> 该位表示 PCM 协议里同步帧信号的高脉冲长度。 0 = 一个 BCLK 周期 1 = 一个通道周期 <b>注:</b> 该位仅在主机模式下有效

[26:24]	<b>FORMAT</b>	<b>数据格式选择</b> 000 = I <sup>2</sup> S 标准格式 001 = I <sup>2</sup> S MSB 对齐 010 = I <sup>2</sup> S LSB 对齐 011 = 保留 100 = PCM 标准格式 101 = PCM MSB 对齐 110 = PCM LSB 对齐 111 = 保留
[23]	<b>RXLCH</b>	<b>左声道接收使能位</b> 选择了单声道格式 (MONO = 1) 的情况下, 如果 RXLCH 为 0, I <sup>2</sup> S 会接收通道 1 的数据, 如果 RXLCH 为 1, 则接收通道 0 的数据。 0 = 单声道模式下接收通道 1 的数据 1 = 单声道模式下接收通道 0 的数据
[22]	<b>Reserved</b>	保留
[21]	<b>RXPDMAEN</b>	<b>PDMA 接收使能</b> 0 = 禁止 PDMA 接收功能 1 = 使能 PDMA 接收功能
[20]	<b>TXPDMAEN</b>	<b>PDMA 发送使能</b> 0 = 禁止 PDMA 发送功能 1 = 使能 PDMA 发送功能
[19]	<b>RXFBCLR</b>	<b>清接收 FIFO</b> 0 = 无效 1 = 清 Rx FIFO <b>注 1:</b> 写 1 到该位清接收 FIFO, 内部指针会复位到 FIFO 起点, RXCNT (I2S_STATUS1[20:16]) 复位为 0 且接收 FIFO 会被清空。 <b>注 2:</b> 该位由硬件自动清 0, 读该位返回 0.
[18]	<b>TXFBCLR</b>	<b>清发送 FIFO</b> 0 = 无效 1 = 清 Tx FIFO. <b>注 1:</b> 写 1 到该位清发送 FIFO, 内部指针会复位到 FIFO 起点, TXCNT (I2S_STATUS1[12:8]) 复位为 0 且发送 FIFO 会被清空, 但发送 FIFO 里的数据不会改变 <b>注 2:</b> 该位由硬件自动清 0, 读该位返回 0.
[17:16]	<b>Reserved</b>	保留
[15]	<b>MCLKEN</b>	<b>主机时钟使能控制</b> 使能 MCLKEN, I <sup>2</sup> S 控制器会在 I2S_MCLK 管脚上为外部音频设备产生主机时钟 0 = 关闭主机时钟 1 = 使能主机时钟
[14:9]	<b>Reserved</b>	保留

[8]	<b>SLAVE</b>	<p><b>从机模式使能控制位</b>            0 = 主机模式            1 = 从机模式</p> <p><b>注:</b> I<sup>2</sup>S 可以工作在主机模式或从机模式。主机模式下, I<sup>2</sup>S_BCLK 和 I<sup>2</sup>S_LRCLK 管脚处于输出模式, 主机输出 I<sup>2</sup>S_BCLK 和 I<sup>2</sup>S_LRCLK 到音频编解码芯片。从机模式下, I<sup>2</sup>S_BCLK 和 I<sup>2</sup>S_LRCLK 管脚处于输入模式, I<sup>2</sup>S_BCLK 和 I<sup>2</sup>S_LRCLK 则由音频编解码芯片输入。</p>
[7]	<b>ORDER</b>	<p><b>FIFO 内双声道数据顺序</b>            对于 8/16 位数据宽度, 该位用来决定是奇数通道还是偶数通道存储在高字节。对于 24 位的数据宽度, 该位则用于选择音频数据存储在 32 位数据 FIFO 时是左对齐还是右对齐。            0 = 8/16 位数据宽度下, 偶数通道存储在高字节            24 位的 LSB 音频数据存储在 32 位数据 FIFO 时是右对齐            1 = 8/16 位数据宽度下, 偶数通道存储在低字节.            24 位的 MSB 音频数据存储在 32 位数据 FIFO 时是左对齐.</p>
[6]	<b>MONO</b>	<p><b>单声道数据控制</b>            0 = 双声道格式的数据            1 = 单声道格式的数据</p> <p><b>注:</b> 录音时, 若使能了单声道格式, RXLCH (I<sup>2</sup>S_CTL0[23]) 代表哪路声音数据会被录下来</p>
[5:4]	<b>DATWIDTH</b>	<p><b>数据宽度</b>            该域表示每个音频通道的数据宽度            00 = 数据宽度为 8 位            01 = 数据宽度为 16 位            10 = 数据宽度为 24 位            11 = 数据宽度为 32 位</p>
[3]	<b>MUTE</b>	<p><b>发送静音功能控制</b>            0 = 正常发送            1 = 发送 0 值到音频通道</p>
[2]	<b>RXEN</b>	<p><b>接收使能控制</b>            0 = 禁止接收            1 = 使能接收</p>
[1]	<b>TXEN</b>	<p><b>发送使能控制</b>            0 = 禁止发送            1 = 使能发送</p>
[0]	<b>I<sup>2</sup>SEN</b>	<p><b>I<sup>2</sup>S 控制器使能控制</b>            0 = I<sup>2</sup>S 控制器禁止            1 = I<sup>2</sup>S 控制器使能</p>

**I<sup>2</sup>S\_CTL1 I<sup>2</sup>S 控制寄存器 1**

寄存器	偏移量	R/W	描述	复位值
I <sup>2</sup> S_CTL1	I <sup>2</sup> S_BA+0x20	R/W	I <sup>2</sup> S 控制寄存器 1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved						PB16ORD	PBWIDTH
23	22	21	20	19	18	17	16
Reserved				RXTH			
15	14	13	12	11	10	9	8
Reserved				TXTH			
7	6	5	4	3	2	1	0
CH7ZCEN	CH6ZCEN	CH5ZCEN	CH4ZCEN	CH3ZCEN	CH2ZCEN	CH1ZCEN	CH0ZCEN

位	描述	
[31:26]	Reserved	Reserved.
[25]	PB16ORD	保留
[24]	PBWIDTH	<p><b>16 位外设总线R/W FIFO 顺序</b>            PBWIDTH 为 1 时，数据 FIFO 由 2 条外设总线进行存取。该位用来选择外设总线对 32 位 FIFO 进行操作时的顺序。            0 = 先存取低 16 位            1 = 先存取高 16 位  <b>注:</b> 仅当 PBWIDTH = 1 时可用</p>
[23:20]	Reserved	<p><b>外设总线数据宽度选择</b>            该位用来选择外设总线 APB 的数据宽度。使能 PDMA 功能时，必须置该位为 1 进入 16 位的传输模式。            0 = 32 位数据宽度            1 = 16 位数据宽度  <b>注 1:</b> 如果 PBWIDTH=1，32 位数据总线的低 16 位可用。  <b>注 2:</b> 如果 PBWIDTH=1，2 条 FIFO 写操作完成后，发送 FIFO 的深度加一。  <b>注 3:</b> 如果 PBWIDTH=1，2 条 FIFO 读操作完成后，接收 FIFO 的深度减一。</p>
[19:16]	RXTH	保留

[15:12]	<b>Reserved</b>	<b>接收 FIFO 阈值水平</b> 0000 = 接收 FIFO 阈值为 1 数据字 0001 = 接收 FIFO 阈值为 2 数据字 0010 = 接收 FIFO 阈值为 3 数据字 ... 1110 = 接收 FIFO 阈值为 15 数据字 1111 = 接收 FIFO 阈值为 16 数据字 <b>注:</b> 当接收到的数据数超过阈值水平时, RXTHIF (I2S_STATUS0[10]) 会被置位
[11:8]	<b>TXTH</b>	保留
[7]	<b>CH7ZCEN</b>	<b>发送 FIFO 阈值水平</b> 0000 = 发送 FIFO 阈值为 0 数据字 0001 = 发送 FIFO 阈值为 1 数据字 0010 = 发送 FIFO 阈值为 2 数据字 ... 1110 = 发送 FIFO 阈值为 14 数据字 1111 = 发送 FIFO 阈值为 15 数据字 <b>注:</b> 当发送 FIFO 里剩余的数据数小于等于发送阈值水平时, TXTHIF (I2S_STATUS0[18]) 会被置位
[6]	<b>CH6ZCEN</b>	<b>通道 7 过零检测使能控制</b> 0 = 通道 7 过零检测关闭 1 = 通道 7 过零检测使能 <b>注 1:</b> 该位仅在多路通道 PCM 模式, TDMCHNUM (I2S_CTL0[31:30]) = 0x1, 0x2, 0x3 时可用 <b>注 2:</b> 如果置位该位, 当通道 7 数据信号位改变或下一个数据所有位都为 0, CH7ZCIF (I2S_STATUS1[7]) 会被置位 <b>注 3:</b> CH7ZCIF 标志被置位时, 通道 7 会被静音
[5]	<b>CH5ZCEN</b>	<b>通道 6 过零检测使能控制</b> 0 = 通道 6 过零检测关闭 1 = 通道 6 过零检测使能 <b>注 1:</b> 该位仅在多路通道 PCM 模式, TDMCHNUM (I2S_CTL0[31:30]) = 0x1, 0x2, 0x3 时可用 <b>注 2:</b> 如果置位该位, 当通道 6 数据信号位改变或下一个数据所有位都为 0, CH6ZCIF (I2S_STATUS1[6]) 会被置位 <b>注 3:</b> CH6ZCIF 标志被置位时, 通道 6 会被静音
[4]	<b>CH4ZCEN</b>	<b>通道 5 过零检测使能控制</b> 0 = 通道 5 过零检测关闭 1 = 通道 5 过零检测使能 <b>注 1:</b> 该位仅在多路通道 PCM 模式, TDMCHNUM (I2S_CTL0[31:30]) = 0x1, 0x2, 0x3 时可用 <b>注 2:</b> 如果置位该位, 当通道 5 数据信号位改变或下一个数据所有位都为 0, CH5ZCIF (I2S_STATUS1[5]) 会被置位 <b>注 3:</b> CH5ZCIF 标志被置位时, 通道 5 会被静音

[3]	<b>CH3ZCEN</b>	<p><b>通道 4 过零检测使能控制</b>            0 = 通道 4 过零检测关闭            1 = 通道 4 过零检测使能  <b>注 1:</b> 该位仅在多路通道 PCM 模式, TDMCHNUM (I2S_CTL0[31:30]) = 0x1, 0x2, 0x3 时可用  <b>注 2:</b> 如果置位该位, 当通道 4 数据信号位改变或下一个数据所有位都为 0, CH4ZCIF (I2S_STATUS1[4]) 会被置位  <b>注 3:</b> CH4ZCIF 标志被置位时, 通道 4 会被静音</p>
[2]	<b>CH2ZCEN</b>	<p><b>通道 3 过零检测使能控制</b>            0 = 通道 3 过零检测关闭            1 = 通道 3 过零检测使能  <b>注 1:</b> 该位仅在多路通道 PCM 模式, TDMCHNUM (I2S_CTL0[31:30]) = 0x1, 0x2, 0x3 时可用  <b>注 2:</b> 如果置位该位, 当通道 3 数据信号位改变或下一个数据所有位都为 0, CH3ZCIF (I2S_STATUS1[3]) 会被置位  <b>注 3:</b> CH3ZCIF 标志被置位时, 通道 3 会被静音</p>
[1]	<b>CH1ZCEN</b>	<p><b>通道 2 过零检测使能控制</b>            0 = 通道 2 过零检测关闭            1 = 通道 2 过零检测使能  <b>注 1:</b> 该位仅在多路通道 PCM 模式, TDMCHNUM (I2S_CTL0[31:30]) = 0x1, 0x2, 0x3 时可用  <b>注 2:</b> 如果置位该位, 当通道 2 数据信号位改变或下一个数据所有位都为 0, CH2ZCIF (I2S_STATUS1[2]) 会被置位  <b>注 3:</b> CH2ZCIF 标志被置位时, 通道 2 会被静音</p>
[0]	<b>CH0ZCEN</b>	<p><b>通道 1 过零检测使能控制</b>            0 = 通道 1 过零检测关闭            1 = 通道 1 过零检测使能  <b>注 1:</b> 通道 1 在 I2S (FORMAT[2]=0) 模式下或 2 通道的 PCM 模式下也指右声道  <b>注 2:</b> 如果置位该位, 当通道 1 数据信号位改变或下一个数据所有位都为 0, CH1ZCIF (I2S_STATUS1[1]) 会被置位  <b>注 3:</b> CH1ZCIF 标志被置位时, 通道 1 会被静音</p>

**I2S\_CLKDIV I2S 时钟分频器寄存器**

寄存器	偏移量	R/W	描述	复位值
I2S_CLKDIV	I2S_BA+0x04	R/W	I <sup>2</sup> S 时钟分频器寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
BCLKDIV							
7	6	5	4	3	2	1	0
Reserved		MCLKDIV					

位	描述	
[31:17]	<b>Reserved</b>	保留
[16:8]	<b>BCLKDIV</b>	<b>位时钟除频</b> 主机模式下 I <sup>2</sup> S 控制器产生位时钟。软件可以配置该寄存器产生特定频率的采样时钟。 $F_{BCLK} = F_{I2SCLK} / (2 * (BCLKDIV + 1))$ . 注: F_BCLK 为 BCLK 的频率, F_I2SCLK 为 I2S_CLK 的频率
[7:6]	<b>Reserved</b>	保留
[5:0]	<b>MCLKDIV</b>	<b>主时钟除频</b> 如果外部晶振频率为 $(2 \times MCLKDIV) * 256fs$ , 则软件可以配置该寄存器产生 256fs 的时钟频率到音频编解码器芯片。如果 MCLKDIV 为 0, MCLK 和外部输入时钟一致 例如, 采样率为 24KHz, 芯片外部时钟为 12.288 MHz, 置 MCLKDIV 为 1 $F_{MCLK} = F_{I2SCLK} / (2 \times (MCLKDIV))$ ( $MCLKDIV \geq 1$ ). $F_{MCLK} = F_{I2SCLK}$ ( $MCLKDIV = 0$ ). 注: F_MCLK 为 MCLK 的频率, F_I2SCLK 为 I2S_CLK 的频率

**I<sup>2</sup>S\_IEN I<sup>2</sup>S 中断使能寄存器**

寄存器	偏移量	R/W	描述	复位值
I <sup>2</sup> S_IEN	I <sup>2</sup> S_BA+0x08	R/W	I <sup>2</sup> S 中断使能寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
CH7ZCIEN	CH6ZCIEN	CH5ZCIEN	CH4ZCIEN	CH3ZCIEN	CH2ZCIEN	CH1ZCIEN	CH0ZCIEN
15	14	13	12	11	10	9	8
Reserved					TXTHIEN	TXOVFIEN	TXUDFIEN
7	6	5	4	3	2	1	0
Reserved					RXTHIEN	RXOVFIEN	RXUDFIEN

位	描述
[31:24]	Reserved 保留
[23]	CH7ZCIEN 通道 7 过零检测中断使能控制 0 = 关闭中断 1 = 使能中断 <b>注 1:</b> 该位置位时, 如果通道 7 检测到过零, 则产生中断 <b>注 2:</b> 该位仅在多路通道 PCM 模式, TDMCHNUM (I <sup>2</sup> S_CTL0[31:30]) = 0x1, 0x2, 0x3 时可用
[22]	CH6ZCIEN 通道 6 过零检测中断使能控制 0 = 关闭中断 1 = 使能中断 <b>注 1:</b> 该位置位时, 如果通道 6 检测到过零, 则产生中断 <b>注 2:</b> 该位仅在多路通道 PCM 模式, TDMCHNUM (I <sup>2</sup> S_CTL0[31:30]) = 0x1, 0x2, 0x3 时可用
[21]	CH5ZCIEN 通道 5 过零检测中断使能控制 0 = 关闭中断 1 = 使能中断 <b>注 1:</b> 该位置位时, 如果通道 5 检测到过零, 则产生中断 <b>注 2:</b> 该位仅在多路通道 PCM 模式, TDMCHNUM (I <sup>2</sup> S_CTL0[31:30]) = 0x1, 0x2, 0x3 时可用
[20]	CH4ZCIEN 通道 4 过零检测中断使能控制 0 = 关闭中断 1 = 使能中断 <b>注 1:</b> 该位置位时, 如果通道 4 检测到过零, 则产生中断 <b>注 2:</b> 该位仅在多路通道 PCM 模式, TDMCHNUM (I <sup>2</sup> S_CTL0[31:30]) = 0x1, 0x2, 0x3 时可用

[19]	<b>CH3ZCIEN</b>	通道 3 过零检测中断使能控制 0 = 关闭中断 1 = 使能中断 <b>注 1:</b> 该位置位时, 如果通道 3 检测到过零, 则产生中断 <b>注 2:</b> 该位仅在多路通道 PCM 模式, TDMCHNUM (I2S_CTL0[31:30]) = 0x1, 0x2, 0x3 时可用
[18]	<b>CH2ZCIEN</b>	通道 2 过零检测中断使能控制 0 = 关闭中断 1 = 使能中断 <b>注 1:</b> 该位置位时, 如果通道 2 检测到过零, 则产生中断 <b>注 2:</b> 该位仅在多路通道 PCM 模式, TDMCHNUM (I2S_CTL0[31:30]) = 0x1, 0x2, 0x3 时可用
[17]	<b>CH1ZCIEN</b>	通道 1 过零检测中断使能控制 0 = 关闭中断 1 = 使能中断 <b>注 1:</b> 该位置位时, 如果通道 1 检测到过零, 则产生中断 <b>注 2:</b> 通道 1 在 I2S (FORMAT[2]=0) 模式下或 2 通道的 PCM 模式下也指右声道
[16]	<b>CH0ZCIEN</b>	通道 0 过零检测中断使能控制 0 = 关闭中断 1 = 使能中断 <b>注 1:</b> 该位置位时, 如果通道 0 检测到过零, 则产生中断 <b>注 2:</b> 通道 0 在 I2S (FORMAT[2]=0) 模式下或 2 通道的 PCM 模式下也指左声道
[15:11]	<b>Reserved</b>	保留
[10]	<b>TXTHIEN</b>	发送 FIFO 阈值水平中断使能控制 0 = 关闭中断 1 = 使能中断 <b>注:</b> 该位置位时, 如果发送 FIFO 里的数据少于 TXTH (I2S_CTL1[11:8]) 则发生中断
[9]	<b>TXOVFIEN</b>	发送 FIFO 溢出中断使能控制 0 = 关闭中断 1 = 使能中断 <b>注:</b> 该位置位时, 如果 TXOVIF (I2S_STATUS0[17]) 也置位, 则发生中断中断
[8]	<b>TXUDFIEN</b>	发送 FIFO 下溢中断使能控制 0 = 关闭中断 1 = 使能中断 <b>注:</b> 该位置位时, 如果 TXUDIF (I2S_STATUS0[16]) 也置位, 则发生中断中断
[7:3]	<b>Reserved</b>	保留
[2]	<b>RXTHIEN</b>	接收 FIFO 阈值水平中断使能控制 0 = 关闭中断 1 = 使能中断 <b>注:</b> 当接收 FIFO 的数据大于等于 RXTH (I2S_CTL1[19:16]) 且 RXTHIF (I2S_STATUS0[10]) 为 1, 如果使能了 RXTHIEN, 则会发生中断

[1]	<b>RXOVFIEN</b>	接收 FIFO 溢出中断使能控制 0 = 关闭中断 1 = 使能中断 <b>注:</b> 该位置位时, 如果 RXOVIF (I2S_STATUS0[9]) 也置位, 则发生中断中断
[0]	<b>RXUDFIEN</b>	接收 FIFO 下溢中断使能控制 0 = 关闭中断 1 = 使能中断 <b>注:</b> 接收 FIFO 空时, 软件读接收 FIFO 就会导致 RXUDIF (I2S_STATUS0[8]) 置位

**I2S\_STATUS0 I2S 状态寄存器 0**

寄存器	偏移量	R/W	描述	复位值
I2S_STATUS0	I2S_BA+0x0C	R/W	I <sup>2</sup> S 状态寄存器 0	0x0014_1000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved	TXBUSY	TXEMPTY	TXFULL	TXTHIF	TXOVIF	TXUDIF	
15	14	13	12	11	10	9	8
Reserved			RXEMPTY	RXFULL	RXTHIF	RXOVIF	RXUDIF
7	6	5	4	3	2	1	0
Reserved		DATACH			I2STXINT	I2SRXINT	I2SINT

位	描述	
[31:22]	Reserved	保留
[21]	TXBUSY	<b>发送忙 (只读)</b> 0 = 发送移位缓存空 1 = 发送移位缓存忙 <b>注:</b> 当发送 FIFO 和移位缓存里的数据都被移出时, 该位清 0。当有数据载入移位缓存时, 该位置 1。
[20]	TXEMPTY	<b>发送 FIFO 空 (只读)</b> 该为指示发送 FIFO 是否为空 0 = 非空 1 = 空
[19]	TXFULL	<b>发送 FIFO 满 (只读)</b> 该位指示发送 FIFO 是否满 0 = 未满 1 = 满。
[18]	TXTHIF	<b>发送 FIFO 域值中断标志 (只读)</b> 0 = FIFO 里的数据超过了阈值水平 1 = FIFO 里的数据低于阈值水平 <b>注:</b> 当 FIFO 里的的数据小于等于 TXTH (I2S_CTL1[11:8]) 的阈值设定, TXTHIF 该置 1。 软件写 TXFIFO 寄存器后, 当 TXCNT (I2S_STATUS1[12:8]) 超过 TXTH (I2S_CTL1[11:8]) 时, 该位才会改变

[17]	<b>TXOVIF</b>	<b>发送 FIFO 溢出中断标志</b> 0 = 未溢出 1 = 溢出。 <b>注 1:</b> 发送 FIFO 满时还向 FIFO 写数据会导致溢出 <b>注 2:</b> 写 1 清 0 该位。
[16]	<b>TXUDIF</b>	<b>发送 FIFO 下溢中断标志</b> 0 = 未下溢 1 = 下溢。 <b>注 1:</b> 当移位逻辑电路从发送 FIFO 读数据, 发送 FIFO 里的数据量小于一帧音频的数据量时, 该位置位 <b>注 2:</b> 写 1 清 0 该位。
[15:13]	<b>Reserved</b>	保留
[12]	<b>RXEMPTY</b>	<b>接收 FIFO 空 (只读)</b> 0 = 非空。 1 = 空。 <b>注:</b> 该位用来指示接收 FIFO 是否为空
[11]	<b>RXFULL</b>	<b>接收 FIFO 满 (只读)</b> 0 = 未满 1 = 满。 <b>注:</b> 该位用来指示接收 FIFO 是否已满
[10]	<b>RXTHIF</b>	<b>接收 FIFO 阈值中断标志 (只读)</b> 0 = 接收 FIFO 里的数据未超过阈值水平 1 = 接收 FIFO 里的数据超过了阈值水平 <b>注:</b> 当接收 FIFO 里的数据未超过了 RXTH (I2S_CTL1[19:16]) 设定的阈值水平, 该位置 1。软件读 RXFIFO 寄存器后, 当 RXCNT (I2S_STATUS1[20:16]) 小于 RXTH (I2S_CTL1[19:16]) 时, 该位才会改变
[9]	<b>RXOVIF</b>	<b>接收 FIFO 溢出中断标志</b> 0 = 未溢出 1 = 溢出 <b>注 1:</b> 接收 FIFO 已满, 硬件仍向接收 FIFO 收数据会导致该位置 1, 此时第一个缓存里的数据会被覆盖 <b>注 2:</b> 写 1 清 0 该位。
[8]	<b>RXUDIF</b>	<b>接收 FIFO 下溢中断标志</b> 0 = 未下溢。 1 = 下溢 <b>注 1:</b> 当接收 FIFO 为空时, 软件还去读接收 FIFO 会导致该位置 1, 这种情况就是下溢 <b>注 2:</b> 写 1 清 0 该位。
[7:6]	<b>Reserved</b>	保留

[5:3]	<b>DATACh</b>	<b>传输数据通道 (只读)</b> 该位表示当前是哪一个通道在传输数据 000 = 通道 0 (即 2 通道的 I <sup>2</sup> S/PCM 模式下的左声道). 001 = 通道 1 (即 2 通道的 I <sup>2</sup> S/PCM 模式下的右声道). 010 = 通道 2 (用于使能了 TDM 的 PCM 模式下的 4 通道传输) 011 = 通道 3 (用于使能了 TDM 的 PCM 模式下的 4 通道传输) 100 = 通道 4 (用于使能了 TDM 的 PCM 模式下的 6 通道传输) 101 = 通道 5 (用于使能了 TDM 的 PCM 模式下的 6 通道传输) 110 = 通道 6 (用于使能了 TDM 的 PCM 模式下的 8 通道传输) 111 = 通道 7 (用于使能了 TDM 的 PCM 模式下的 8 通道传输)
[2]	<b>I<sup>2</sup>STXINT</b>	<b>I<sup>2</sup>S 发送中断 (只读)</b> 0 = 无发送中断 1 = 有发送中断
[1]	<b>I<sup>2</sup>SRXINT</b>	<b>I<sup>2</sup>S 接收中断 (只读)</b> 0 = 无接收中断 1 = 有接收中断
[0]	<b>I<sup>2</sup>SINT</b>	<b>I<sup>2</sup>S 中断标志 (只读)</b> 0 = 无 I <sup>2</sup> S 中断 1 = 有 I <sup>2</sup> S 中断 <b>注:</b> 该位为 I <sup>2</sup> STXINT “或” I <sup>2</sup> SRXINT 的结构.

I<sup>2</sup>S\_STATUS1 I<sup>2</sup>S 状态寄存器 1

寄存器	偏移量	R/W	描述	复位值
I <sup>2</sup> S_STATUS1	I <sup>2</sup> S_BA+0x24	R/W	I <sup>2</sup> S 状态寄存器 1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved			RXCNT				
15	14	13	12	11	10	9	8
Reserved			TXCNT				
7	6	5	4	3	2	1	0
CH7ZCIF	CH6ZCIF	CH5ZCIF	CH4ZCIF	CH3ZCIF	CH2ZCIF	CH1ZCIF	CH0ZCIF

位	描述	
[31:21]	Reserved	保留
[20:16]	RXCNT	<p><b>接收 FIFO 水平 (只读)</b></p> <p>该位表示接收 FIFO 里的可用数据个数</p> <p>00000 = 无数据.</p> <p>00001 = 接收 FIFO 里有 1 个字数据.</p> <p>00010 = 接收 FIFO 里有 2 个字数据.</p> <p>....</p> <p>01110 = 接收 FIFO 里有 14 个字数据.</p> <p>01111 = 接收 FIFO 里有 15 个字数据.</p> <p>10000 = 接收 FIFO 里有 16 个字数据.</p> <p>其他保留</p>
[15:13]	Reserved	保留
[12:8]	TXCNT	<p><b>发送 FIFO 水平 (只读)</b></p> <p>该位表示发送 FIFO 里的可用数据个数</p> <p>00000 = 无数据.</p> <p>00001 = 发送 FIFO 里有 1 个字数据</p> <p>00010 = 发送 FIFO 里有 2 个字数据.</p> <p>....</p> <p>01110 = 发送 FIFO 里有 14 个字数据.</p> <p>01111 = 发送 FIFO 里有 15 个字数据.</p> <p>10000 = 发送 FIFO 里有 16 个字数据.</p> <p>其他保留</p>

[7]	<b>CH7ZCIF</b>	<p><b>通道 7 过零中断标志</b>          该位表示通道 7 下一笔数据符号位改变或所有位均为 0          0 = 通道 7 未检测到过零          1 = 通道 7 检测到过零.  <b>注 1:</b> 写 1 清 0 该位.  <b>注 2:</b> 该位仅在多路通道 PCM 模式, TDMCHNUM (I2S_CTL0[31:30]) = 0x1, 0x2, 0x3 时可用</p>
[6]	<b>CH6ZCIF</b>	<p><b>通道 6 过零中断标志</b>          该位表示通道 6 下一笔数据符号位改变或所有位均为 0          0 = 通道 6 未检测到过零          1 = 通道 6 检测到过零.  <b>注 1:</b> 写 1 清 0 该位.  <b>注 2:</b> 该位仅在多路通道 PCM 模式, TDMCHNUM (I2S_CTL0[31:30]) = 0x1, 0x2, 0x3 时可用</p>
[5]	<b>CH5ZCIF</b>	<p><b>通道 5 过零中断标志</b>          该位表示通道 5 下一笔数据符号位改变或所有位均为 0          0 = 通道 5 未检测到过零          1 = 通道 5 检测到过零.  <b>注 1:</b> 写 1 清 0 该位.  <b>注 2:</b> 该位仅在多路通道 PCM 模式, TDMCHNUM (I2S_CTL0[31:30]) = 0x1, 0x2, 0x3 时可用</p>
[4]	<b>CH4ZCIF</b>	<p><b>通道 4 过零中断标志</b>          该位表示通道 4 下一笔数据符号位改变或所有位均为 0          0 = 通道 4 未检测到过零          1 = 通道 4 检测到过零.  <b>注 1:</b> 写 1 清 0 该位.  <b>注 2:</b> 该位仅在多路通道 PCM 模式, TDMCHNUM (I2S_CTL0[31:30]) = 0x1, 0x2, 0x3 时可用</p>
[3]	<b>CH3ZCIF</b>	<p><b>通道 3 过零中断标志</b>          该位表示通道 3 下一笔数据符号位改变或所有位均为 0          0 = 通道 3 未检测到过零          1 = 通道 3 检测到过零.  <b>注 1:</b> 写 1 清 0 该位.  <b>注 2:</b> 该位仅在多路通道 PCM 模式, TDMCHNUM (I2S_CTL0[31:30]) = 0x1, 0x2, 0x3 时可用</p>
[2]	<b>CH2ZCIF</b>	<p><b>通道 2 过零中断标志</b>          该位表示通道 2 下一笔数据符号位改变或所有位均为 0          0 = 通道 2 未检测到过零          1 = 通道 2 检测到过零.  <b>注 1:</b> 写 1 清 0 该位.  <b>注 2:</b> 该位仅在多路通道 PCM 模式, TDMCHNUM (I2S_CTL0[31:30]) = 0x1, 0x2, 0x3 时可用</p>

[1]	<b>CH1ZCIF</b>	<b>通道 1 过零中断标志</b> 该位表示通道 1 下一笔数据符号位改变或所有位均为 0 0 = 通道 1 未检测到过零 1 = 通道 1 检测到过零. <b>注 1:</b> 写 1 清 0 该位. <b>注 2:</b> 通道 1 在 I2S (FORMAT[2]=0) 模式下或 2 通道的 PCM 模式下也指右声道
[0]	<b>CH0ZCIF</b>	<b>通道 0 过零中断标志</b> 该位表示通道 0 下一笔数据符号位改变或所有位均为 0 0 = 通道 0 未检测到过零 1 = 通道 0 检测到过零. <b>注 1:</b> 写 1 清 0 该位. <b>注 2:</b> 通道 0 在 I2S (FORMAT[2]=0) 模式下或 2 通道的 PCM 模式下也指左声道

**I2S\_TXFIFO I<sup>2</sup>S 发送 FIFO 寄存器**

寄存器	偏移量	R/W	描述	复位值
I2S_TXFIFO	I2S_BA+0x10	W	I <sup>2</sup> S 发送 FIFO 寄存器	0x0000_0000

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

位	描述	
[31:0]	TXFIFO	<b>发送 FIFO 位</b> I <sup>2</sup> S 有 16 个字 (16*32 位) 数据缓存用于数据发送。写数据到该寄存器以准备发送。其剩余的数据字数可以从 TXCNT (I2S_STATUS1[12:8]) 里读取

**I2S\_RXFIFO I<sup>2</sup>S 接收 FIFO 寄存器**

寄存器	偏移量	R/W	描述	复位值
I2S_RXFIFO	I2S_BA+0x14	R	I <sup>2</sup> S 接收 FIFO 寄存器	0x0000_0000

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

位	描述	
[31:0]	RXFIFO	<b>接收 FIFO 位</b> I <sup>2</sup> S 有 16 个字 (16*32 位) 数据缓存用于数据接收。从该寄存器读取 FIFO 里的数据。其剩余的数据字数可以从 RXCNT (I2S_STATUS1[20:16]) 里读取

## 6.19 SPI 接口

### 6.19.1 概述

SPI接口是全双工同步串行数据通讯接口，可做为主机或从机，用4线双向通讯。M480包含4组SPI控制器。

SPI0, SPI1, SPI2 和 SPI3支持 I<sup>2</sup>S 模式。每个都支持 PDMA 传输功能。

### 6.19.2 特征

- SPI 模式

- 4 组 SPI 控制器
- 支持主机模式和从机模式
- 主/从模式时钟最高 100 MHz, (VDD = 2.7~3.6V)
- 传输位长可为 8 ~ 32
- 收发独立的 4 级 FIFO 缓存
- 高低位在前可配置
- 支持字节重排功能
- 支持字节或字暂停模式
- 支持 PDMA 传输
- 支持一数据通道半双工传输
- 支持只接收模式

- I<sup>2</sup>S 模式 (for SPI0~SPI3)

- 支持主机/从模式
- 可处理 8, 16, 24 和 32 位数据宽度
- 每组控制器提供 2 个 4 级 FIFO 缓存，一个用于发送，另一个用于接收
- 支持单声道和双声道格式
- 支持 PCM A, PCM B, I<sup>2</sup>S 和 MSB 对齐的数据格式
- 支持 2 个 PDMA 请求，一个用于发送，另一个用于接收

	QSPI0	SPI0 / SPI1 / SPI2 / SPI3
双/四 I/O 模式	V	X
2 位传输模式	V	X
FIFO 深度	8 级	SPI 模式 8~16 位长: 8 级 其它: 4 级
从机超时功能	V	X
从机三线模式	V	X
I <sup>2</sup> S 模式	X	V

表 6.19-1 SPI 特性对比

### 6.19.3 框图

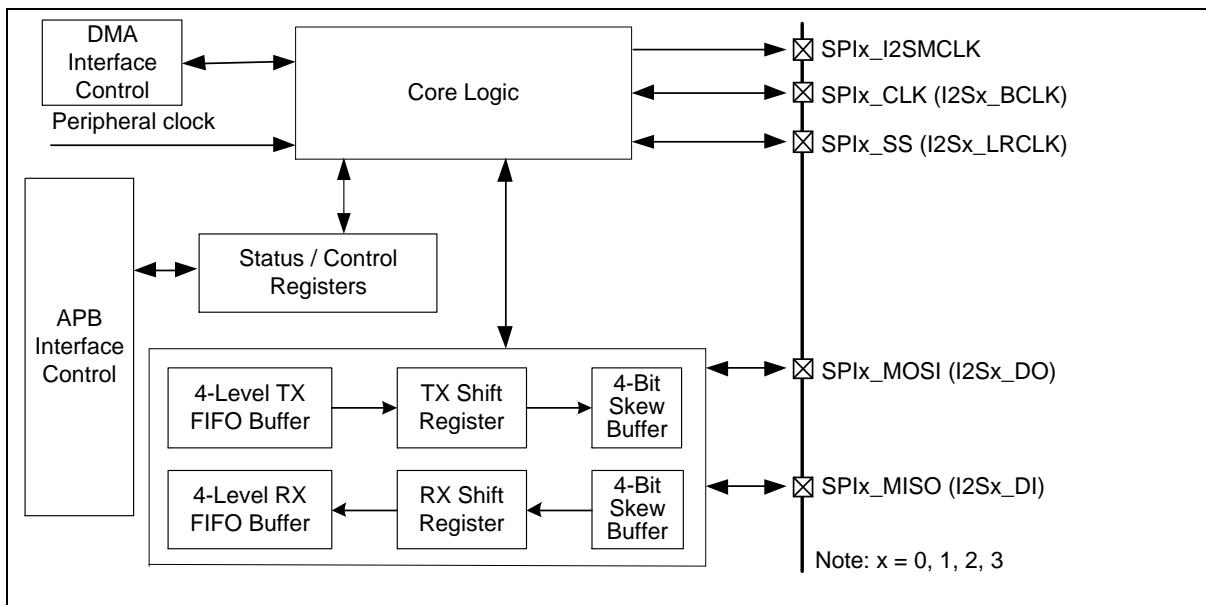


图 6.19-1 SPI 框架图 (SPI0/1/2/3)

#### **TX FIFO 缓存:**

发送 FIFO 4 级深度，32 位宽。数据写入寄存器 SPIx\_TX 就写入发送 FIFO。SPI0~SPI3 的 SPI 模式，如果数据长度为 8~16 位，则发送 FIFO 可以设为 8 级。

#### **RX FIFO 缓存:**

接收 FIFO 4 级深度，32 位宽。接收控制会把数据存到该缓存。软件可通过读 SPIx\_RX 得到该 FIFO 数据。SPI0~SPI3 处于 SPI 模式时，如果数据长度为 8~16 位，发送 FIFO 可以设为 8 级。

#### **TX 移位寄存器:**

发送移位寄存器是一个 32 位的寄存器。发送数据从 TX FIFO 缓存加载到这里，并一位一位的移到斜移缓存。

#### **RX 移位寄存器:**

接收移位寄存器是一个 32 位的寄存器。接收数据从斜移缓存一位一位的移到这里，当一个事务完成时再载入到 RX FIFO。

#### **斜移缓存:**

斜移缓存是一个 4 级 1 位的缓存。总共有 2 个斜移缓存分别在发送和接收端。在接收端，斜移缓存用于把数据从 SPI 总线移位到 Rx 移位寄存器。在发送端，斜移缓存用于把数据从 Tx 移位寄存器移位到 SPI 总线。

### 6.19.4 基本配置

#### 6.19.4.1 SPI0 基本配置

- 时钟配置

寄存器 SPI0SEL (CLK\_CLKSEL2[5:4]) 配置 SPI0 的时钟

寄存器 SPI0CKEN (CLK\_APBCLK0[13]) 使能 SPI0 的时钟

- 复位配置

寄存器 SPI0RST (SYS\_IPRST1[13]) 复位 SPI0 控制器

- 管脚配置

组	管脚名	GPIO	MFP
SPI0	SPI0_CLK	PA.2, PB.14, PD.2	MFP4
		PF.8	MFP5
	SPI0_I2SMCLK	PA.4, PC.14, PD.13	MFP4
		PF.10	MFP5
		PB.0	MFP8
		PB.11	MFP9
	SPI0_MISO	PA.1, PB.13, PD.1	MFP4
		PF.7	MFP5
	SPI0_MOSI	PA.0, PB.12, PD.0	MFP4
		PF.6	MFP5
	SPI0_SS	PA.3, PB.15, PD.3	MFP4
		PF.9	MFP5

#### 6.19.4.2 SPI1 基本配置

- 时钟配置
  - 寄存器 SPI1SEL (CLK\_CLKSEL2[7:6]) 配置 SPI1 的时钟
  - 寄存器 SPI1CKEN (CLK\_APBCLK0[14]) 使能 SPI1 的时钟
- 复位配置
  - 寄存器 SPI1RST (SYS\_IPRST1[14]) 复位 SPI1 控制器
- 管脚配置

组	管脚名	GPIO	MFP
SPI1	SPI1_CLK	PH.6	MFP3
		PA.7	MFP4
		PB.3, PD.5	MFP5
		PH.8	MFP6
		PC.1	MFP7
	SPI1_I2SMCLK	PH.3	MFP3
		PA.5	MFP4
		PB.1, PD.13	MFP5
		PH.10	MFP6

		PC.4	MFP7
SPI1_MISO		PH.4	MFP3
		PC.7	MFP4
		PB.5, PD.7	MFP5
		PE.1	MFP6
		PC.3	MFP7
SPI1_MOSI		PH.5	MFP3
		PC.6	MFP4
		PB.4, PD.6	MFP5
		PE.0	MFP6
		PC.2	MFP7
SPI1_SS		PH.7	MFP3
		PA.6	MFP4
		PB.2, PD.4	MFP5
		PH.9	MFP6
		PC.0	MFP7

#### 6.19.4.3 SPI2 基本配置

- 时钟配置
  - 寄存器 SPI2SEL (CLK\_CLKSEL2[11:10]) 配置 SPI2 的时钟
  - 寄存器 SPI2CKEN (CLK\_APBCLK0[15]) 使能 SPI2 的时钟
- 复位配置
  - 寄存器 SPI2RST (SYS\_IPRST1[15]) 复位 SPI2 控制器
- 管脚配置

组	管脚名	GPIO	MFP
SPI2	SPI2_CLK	PG.3	MFP3
		PA.10	MFP4
		PA.13, PE.8	MFP5
	SPI2_I2SMCLK	PG.1	MFP3
		PC.13	MFP4
		PE.12	MFP5
	SPI2_MISO	PG.4	MFP3
		PA.9	MFP4
		PA.14, PE.9	MFP5
	SPI2_MOSI	PF.11	MFP3
		PA.8	MFP4

		PA.15, PE.10	MFP5
	SPI2_SS	PG.2	MFP3
		PA.11	MFP4
		PA.12, PE.11	MFP5

## 6.19.4.4 SPI3 基本配置

- 时钟配置
  - 寄存器 SPI3SEL (CLK\_CLKSEL2[13:12]) 配置 SPI3 的时钟源
  - 寄存器 SPI3CKEN (CLK\_APBCLK1[6]) 使能 SPI3 的时钟
- 复位配置
  - 在寄存器 SPI3RST (SYS\_IPRST2[6]) 复位 SPI3 控制器
- 管脚配置

组	管脚名	GPIO	MFP
SPI3	SPI3_CLK	PG.3	MFP3
		PA.10	MFP4
		PA.13, PE.8	MFP5
	SPI3_I2SMCLK	PG.1	MFP3
		PC.13	MFP4
		PE.12	MFP5
	SPI3_MISO	PG.4	MFP3
		PA.9	MFP4
		PA.14, PE.9	MFP5
	SPI3_MOSI	PF.11	MFP3
		PA.8	MFP4
		PA.15, PE.10	MFP5
	SPI3_SS	PG.2	MFP3
		PA.11	MFP4
		PA.12, PE.11	MFP5

SPI/I2S (SPI0~SPI3) 接口管脚描述如下:

管脚	SPI 模式	I <sup>2</sup> S 模式
SPIx_SS	SPI 从机片选	I <sup>2</sup> S 左右通道时钟 (I2Sx_LRCLK)
SPIx_CLK	SPI 时钟	I <sup>2</sup> S 位时钟 (I2Sx_BCLK)
SPIx_MISO	SPI 主机输入或从机输出	I <sup>2</sup> S 数据输入 (I2Sx_DI)
SPIx_MOSI	SPI 主机输出或从机输入	I <sup>2</sup> S 数据输出 (I2Sx_DO)
SPIx_I2SMCLK	不可用	I <sup>2</sup> S 主时钟输出

表 6.19-2 SPI/I<sup>2</sup>S 接口管脚描述 (SPI0~SPI3)

### 6.19.5 功能描述

#### 6.19.5.1 术语

##### SPI 外设时钟和 SPI 总线时钟

SPI 时钟决定于时钟分频器 (SPIx\_CLKDIV) 和时钟源的设置，时钟源可以设置为 HXT, HIRC, PLL 时钟或 PCLK。寄存器 CLK\_CLKSEL2 的 SPIxSEL 位选择 SPI 的时钟源。寄存器 DIVIDER (SPI\_CLKDIV[8:0]) 的值决定时钟的分频值。

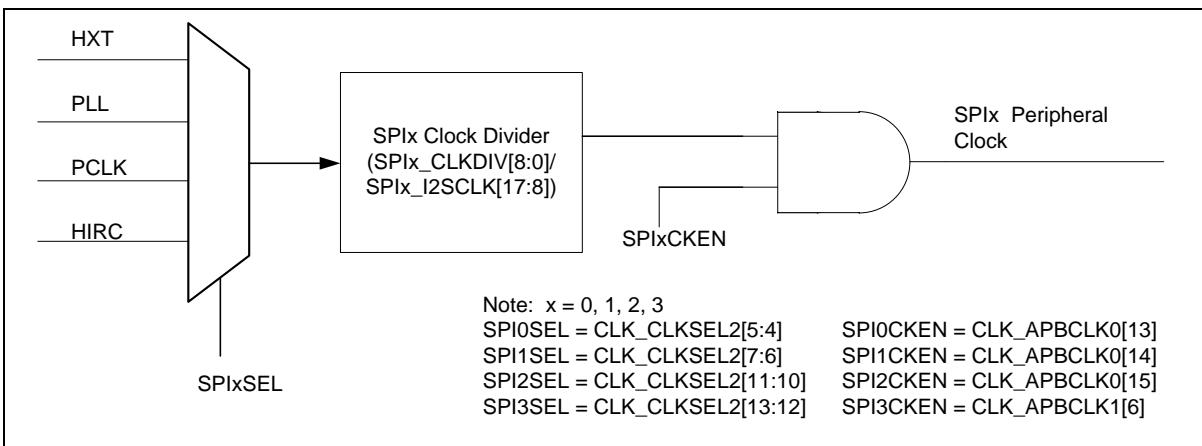


图 6.19-2 SPI 外设时钟

SPI 主机模式下，SPI 总线的时钟频率等于外设的时钟速率。从机模式下，SPI 总线时钟由主机设备提供。无论主机模式还是从机模式，SPI 时钟不能快于系统时钟。如果时钟源不是系统时钟，SPI 时钟不管是主机模式还是从机模式都必须慢于系统时钟频率。

I2S 模式下，外设时钟速率等于 I2S 的位时钟速率，I2S 的位时钟速率由寄存器 SPIx\_I2SCLK 配置。

##### 主从模式

可以通过寄存器 SLAVE (SPIx\_CTL[18]) 设成主机模式或从机模式。SPI 传送过程中，通过 HALFDPX (SPIx\_CTL[14]) 可以切换全双工模式或半双工模式。示意图如下。

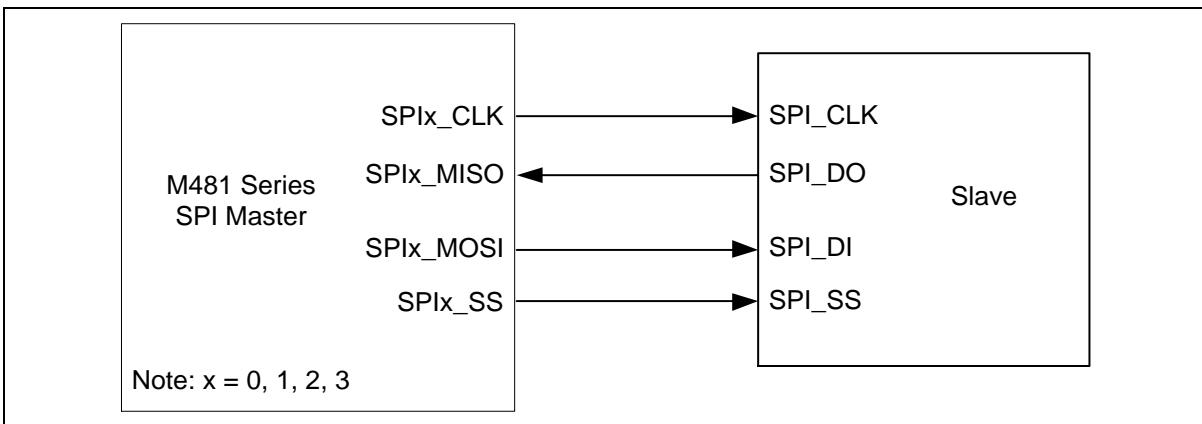


图 6.19-3 SPI 全双工主机模式

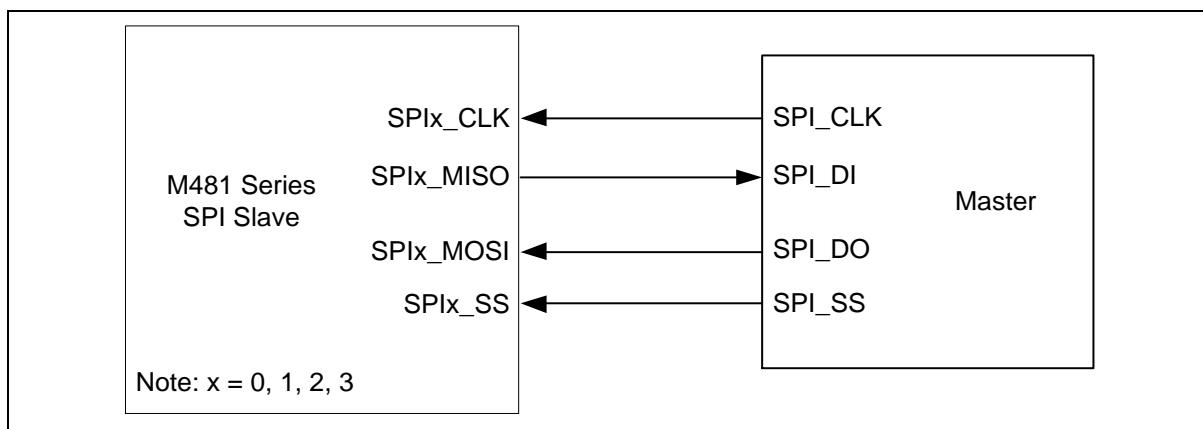


图 6.19-4 SPI 全双工从机模式

### 从机选择

主机模式下，SPI 通过从机片选 SPIx\_SS（输出）使能从机设备。从机模式，其它主机通过 SPIx\_SS（输入）使能 SPI 控制器。片选有效边沿到第一个 SPI 时钟输入应多于 3 个 SPI 外设时钟周期。

从机信号可通过寄存器 SSACTPOL (SPIx\_SSCTL[2]) 设定为低有效或高有效。从机模式下，从机片选信号的无效时间必须大于等于 3 个外设时钟周期。

### 时钟极性

CLKPOL (SPIx\_CTL[3]) 定义 SPI 时钟空闲时的电平。如果 CLKPOL = 1，SPI 时钟空闲时输出高电平，如果 CLKPOL = 0，则 SPI 时钟空闲时输出低电平。

TXNEG (SPIx\_CTL[2]) 定义数据是在 SPI 时钟的下沿发送还是上沿发送。RXNEG (SPIx\_CTL[1]) 定义数据在 SPI 时钟的下沿接收还是上沿接收。

**注意:** TXNEG 和 RXNEG 的设置是互斥的，收发不能都配置为上沿或都是下沿。

### 接发位长

位长由 DWIDTH (SPIx\_CTL[12:8]) 来配置，最多可为 32 位。

当 SPI 完成一次 DWIDTH (SPI\_CTL[12:8]) 定义的位长的收/发时，传输中断标志将被置 1..

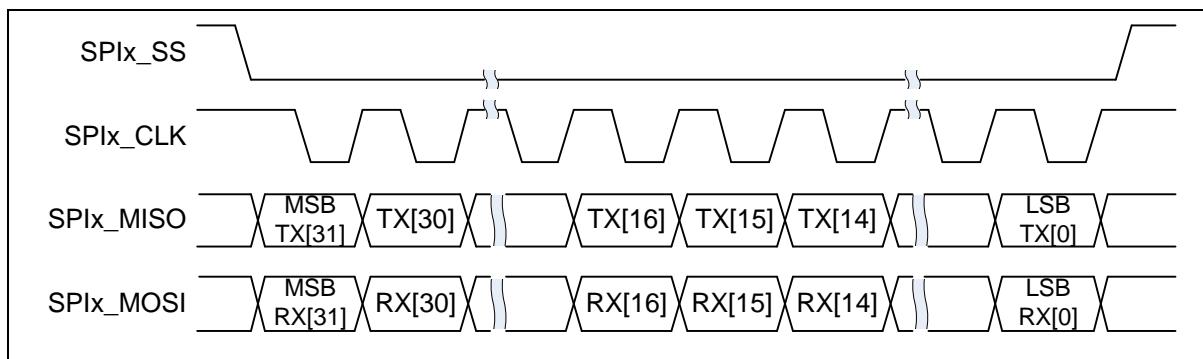


图 6.19-5 32 位的收发

### 高低位在前

LSB (SPIx\_CTL[13]) 定义传输顺序。LSB (SPIx\_CTL[13])=1 时，第 0 位会先传输。若 (SPIx\_CTL[13]) = 0，则先传高位。

### 休眠间隔

主机模式下，SUSPITV (SPIx\_CTL[7:4]) 可配置在两次连续传输之间，插入 0.5~15.5 个 SPI 时钟周期的空闲：空闲时间的计算是从前一次传输的最后一个时钟沿，到下一次传输的第一个时钟沿。SUSPITV 的默认值是 0x3 (3.5 个 SPI 时钟周期)。

#### 6.19.5.2 自动从机选择

主机模式下，如果 AUTOSS (SPIx\_SSCTL[3]) 置位，从机选择信号根据 SS (SPIx\_SSCTL[0]) 自动输出到 SPIx\_SS 管脚上。当数据写入 FIFO 数据传输启动，从机选择信号自动设置为有效状态。当 SPI 总线空闲时，从机选择信号将自动变为无效状态。SPI 总线不空闲时——比如 TX FIFO，TX 移位寄存器或 TX 斜移缓存不空——在 SUSPITV (SPIx\_CTL[7:4]) 的值大于或等于 3 的条件下，从机选择信号在两次传输之间将被置为无效状态。

主机模式下，如果 SUSPITV 的值小于 3 且 AUTOSS 置 1 时，两次连续传输之间，从机选择信号将保持有效。

如果 AUTOSS 被清零，从机选择输出信号的有效电平由 SSACTPOL (SPIx\_SSCTL[2]) 来定义。

从机片选信号有效边沿，到第一个 SPI 时钟边沿的间隔为 1 个 SPI 总线时钟周期。SPI 总线的最后一个时钟，到从机选择信号无效边沿的间隔为 1.5 个 SPI 总线时钟周期。

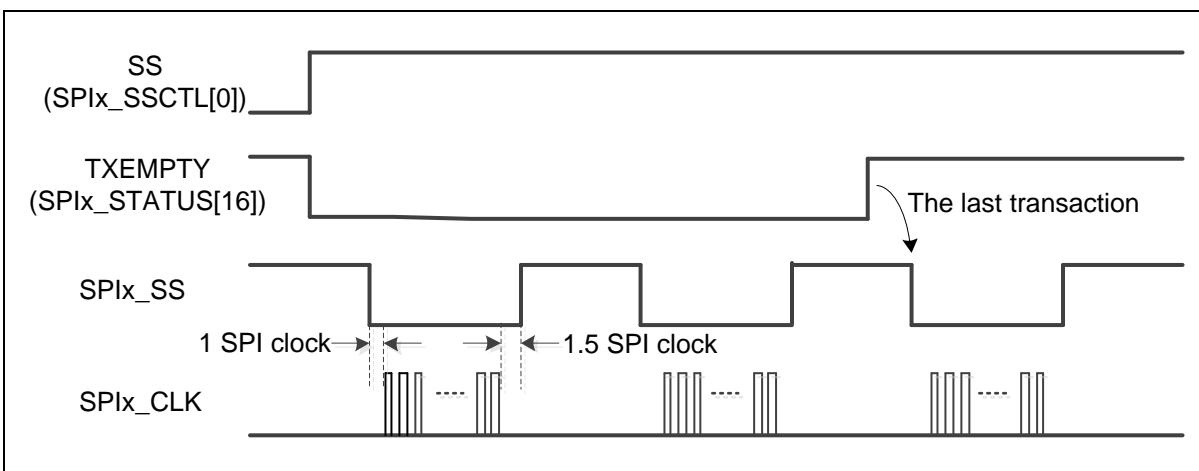


图 6.19-6 自动从机选择 (SSACTPOL = 0, SUSPITV > 0x2)

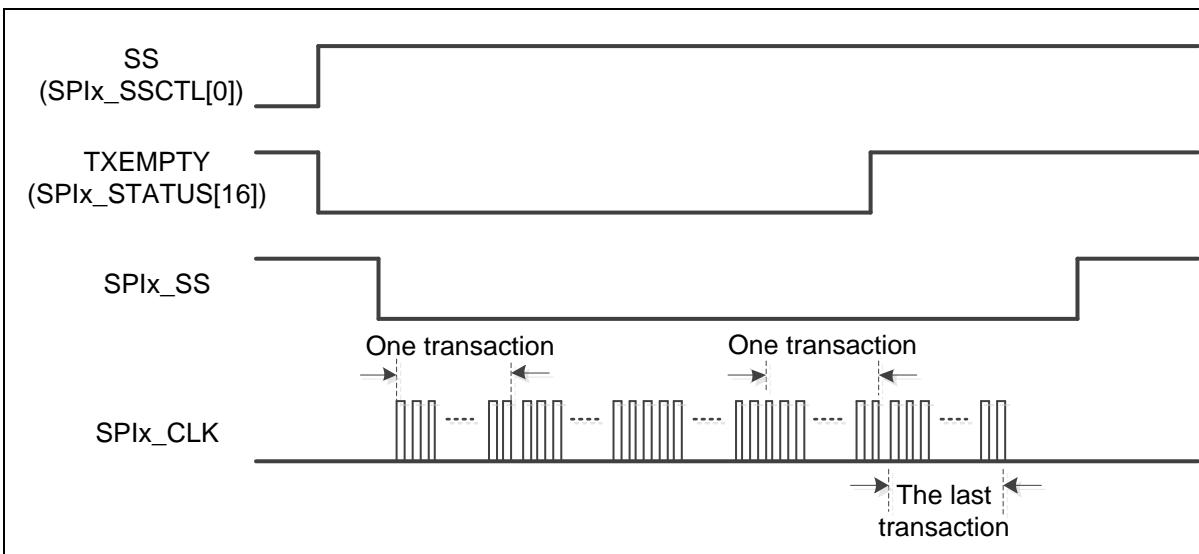


图 6.19-7 自动从机选择 (SSACTPOL = 0, SUSPITV &lt; 0x3)

### 6.19.5.3 字节重排和挂起功能

当传输设置为 MSB 优先 (LSB = 0) 且使能了 REORDER (SPIx\_CTL[19]), 在 32 位传输模式 (DWIDTH = 0) 下, 存储在 TX 缓存与 RX 缓存的数据将按 [BYTE0, BYTE1, BYTE2, BYTE3] 的顺序重新排列。数据发送/接收的顺序为 BYTE0, BYTE1, BYTE2, 和 BYTE3。如果 DWIDTH 设为 24 位传输模式, 存储在 TX 缓存与 RX 缓存的数据将按 [未知字节, BYTE0, BYTE1, BYTE2] 的顺序重新排列。如果 DWIDTH 设为 16 位传输模式, 存储在 TX 缓存与 RX 缓存的数据将按 [X, X, BYTE0, BYTE1] 的顺序重新排列。SPI 控制器将按照 BYTE0, BYTE1, BYTE2 的顺序发送/接收数据, 每个字节 MSB 优先发送/接收。16 位传输模式的规则与上面相同。字节重排序功能只适用于 DWIDTH 为 16, 24, 和 32 位时。

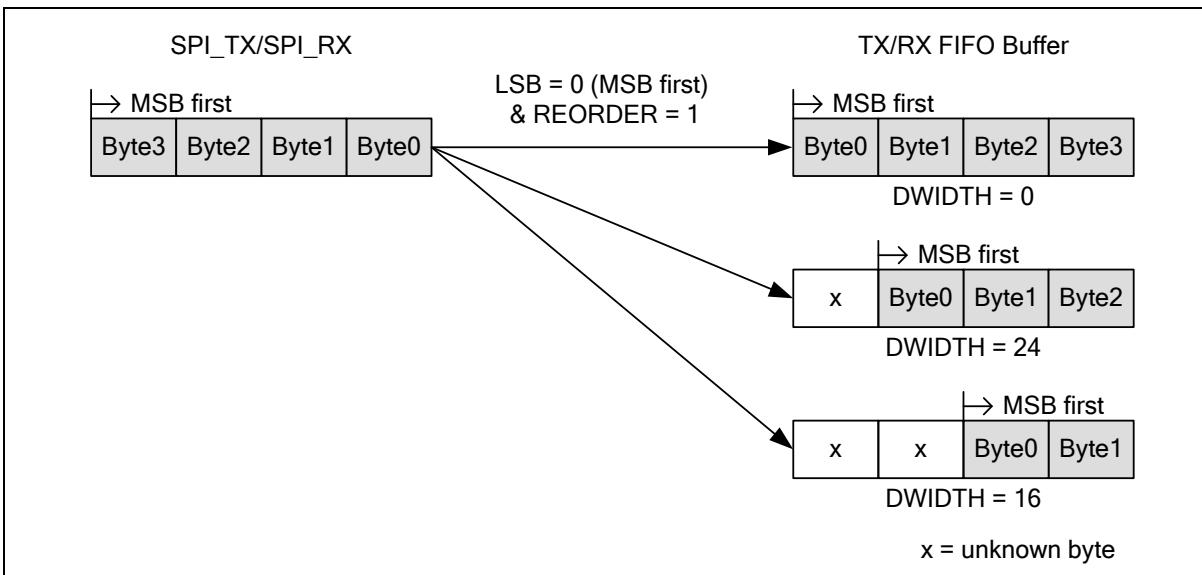


图 6.19-8 字节重排功能

主机模式下, 如果 REORDER (SPIx\_CTL[19]) 为 1, 两次连续传输字节之间将插入 0.5 ~ 15.5 个 SPI 时钟周期的挂起间隔。挂起间隔时间由 SUSPITV (SPIx\_CTL[7:4]) 设置。

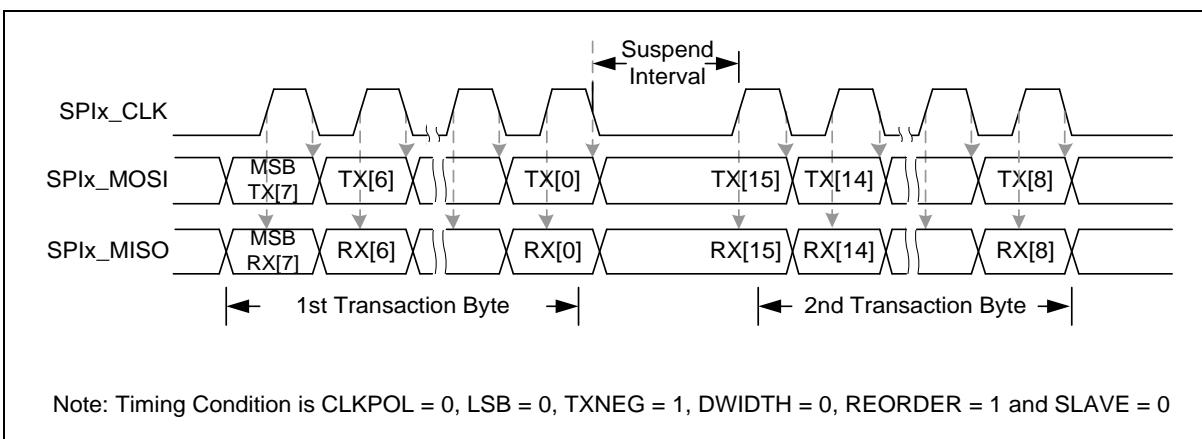


图 6.19-9 字节挂起波形图

### 6.19.5.4 半双工通讯

设置 HALFDPX (SPIx\_CTL[14]) 位可以使 SPI 工作在半双工模式。半双工模式下, 仅有一条数据线进行发送或接收, 方向由 DATDIR (SPIx\_CTL[20]) 配置。半双工模式下, 没有用到的 SPIx\_MISO 管脚可以配置成 GPIO 供其他功能使用。使能或关闭 HALFDPX (SPIx\_CTL[14]) 控制位将会同时产生

TXFBCLR (SPIx\_FIFOCTL[9]) 和 RXFBCLR (SPIx\_FIFOCTL[8])。

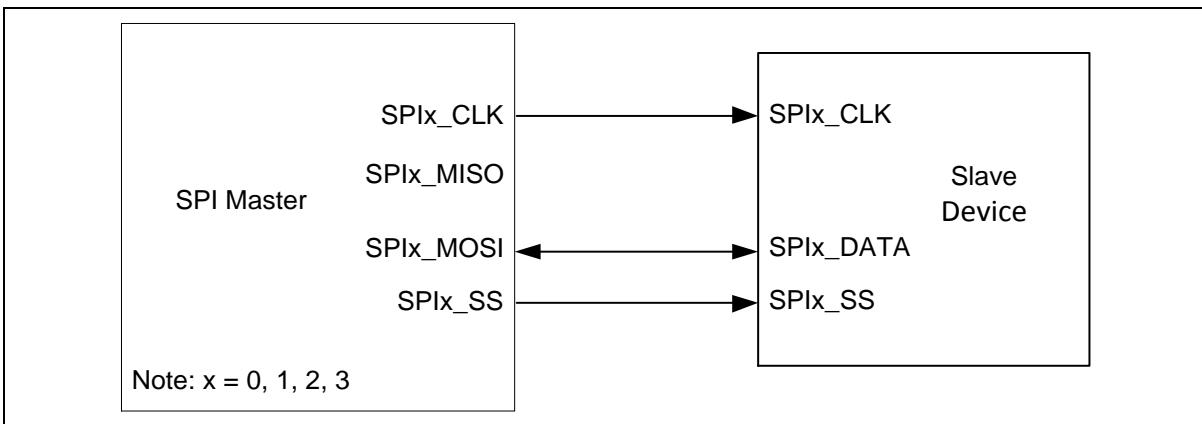


图 6.19-10 SPI 半双工主机模式

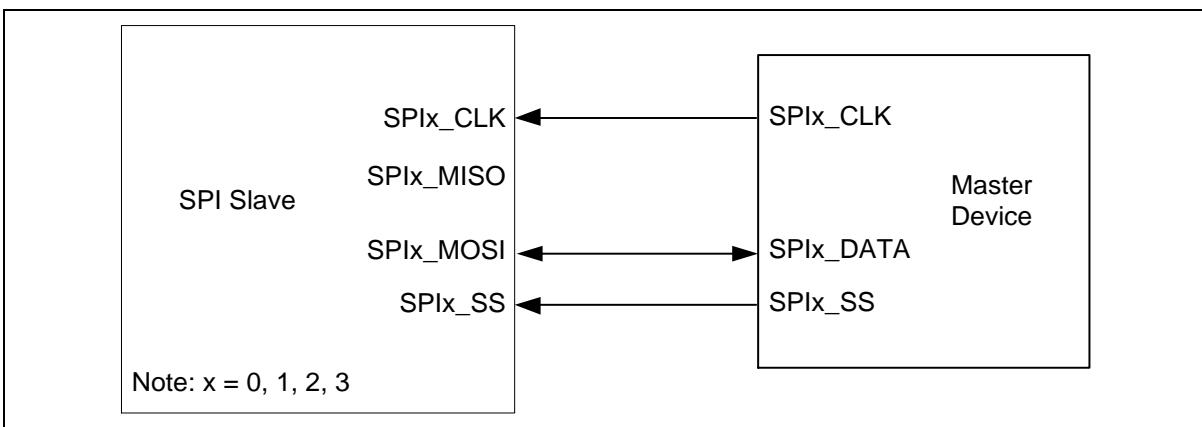


图 6.19-11 SPI 半双工从机模式

#### 6.19.5.5 只接收模式

对于 SPI 主机，可配置 RXONLY (SPIx\_CTL[15]) 让其工作在只接收模式，SPI 主机一直不断地产生 SPI 总线时钟。该模式下，如果使能了 AUTOSS (SPIx\_SSCTL[3])，SPI 主机将一直保持从机片选信号有效。

在只接收模式下，没有用到的 SPIx\_MOSI 管脚可以配置成 GPIO 供其他功能使用。由于 SPI 总线时钟一直存在，BUSY (SPIx\_STATUS[0]) 将会被置位。进入该模式会同时产生 TXFBCLR (SPIx\_FIFOCTL[9]) 和 RXFBCLR (SPIx\_FIFOCTL[8])。使能该模式后，SPI 总线时钟会在 6 个外设时钟周期内发送出去。该模式下，写入发送 FIFO 的数据也会被载入发送移位寄存器发送出去。

#### 6.19.5.6 PDMA 传输功能

SPI 控制器支持 PDMA 传输功能。

当 TXPDMAEN (SPIx\_PDMACTL[0]) 置 1 时，控制器会请求 PDMA 控制发送过程。

当 RXPDMAEN (SPIx\_PDMACTL[1]) 置 1 时，将启动 PDMA 接收过程。当接收 FIFO 有数据时，控制器会请求 PDMA 接收数据。

注：SPI 只支持单一 PDMA 请求（读/写），不支持 PDMA 突发请求。

### 6.19.5.7 FIFO缓存操作

SPI 控制器配备了 4 个 32 位宽的发送和接收 FIFO 缓存。存放在发送 FIFO 缓存的数据会通过发送控制逻辑进行读取和发送。如果发送 FIFO 缓存满了, TXFULL (SPIx\_STATUS[17]) 会被置 1。当 SPI 传输逻辑单元抽出发送 FIFO 缓存的最后一个数据, 发送 FIFO 缓存就为空了, TXEMPTY (SPIx\_STATUS[16]) 位被置 1。注意 TXEMPTY (SPIx\_STATUS[16]) 标志在最后一笔数据传输还在进行时就被置 1 了。在主机模式, 当 FIFO 缓存写入数据或者 SPI 总线上有任何事务, BUSY (SPIx\_STATUS[0]) 位被设置为 1。(如: 从机片选信号激活和 SPI 控制器在从机模式正在接收数据)。当传送缓存为空且当前事务已经完成后, 该位将设置为 0。因此, 软件可以检查 BUSY (SPIx\_STATUS[0]) 位的状态以确认 SPI 是否已经空闲。

接收控制逻辑存储 SPI 接收到的数据到接收 FIFO 缓存。有 FIFO 相关的状态位, 像 RXEMPTY (SPIx\_STATUS[8]) 和 RXFULL (SPIx\_STATUS[9]), 来表明当前 FIFO 缓存的状态。

发送和接收的阀值可以通过设置 TXTH (SPIx\_FIFOCTL[30:28]) 和 RXTH (SPIx\_FIFOCTL[26:24]) 来设定。当存储在发送 FIFO 缓存的有效数据数小于或等于 TXTH (SPIx\_FIFOCTL[30:28]) 的设定时, TXTHIF (SPIx\_STATUS[18]) 位会被置 1。当存储在接收 FIFO 缓存的有效数据数大于 RXTH (SPIx\_FIFOCTL[26:24]) 的设定, RXTHIF (SPIx\_STATUS[10]) 位会被置 1。.

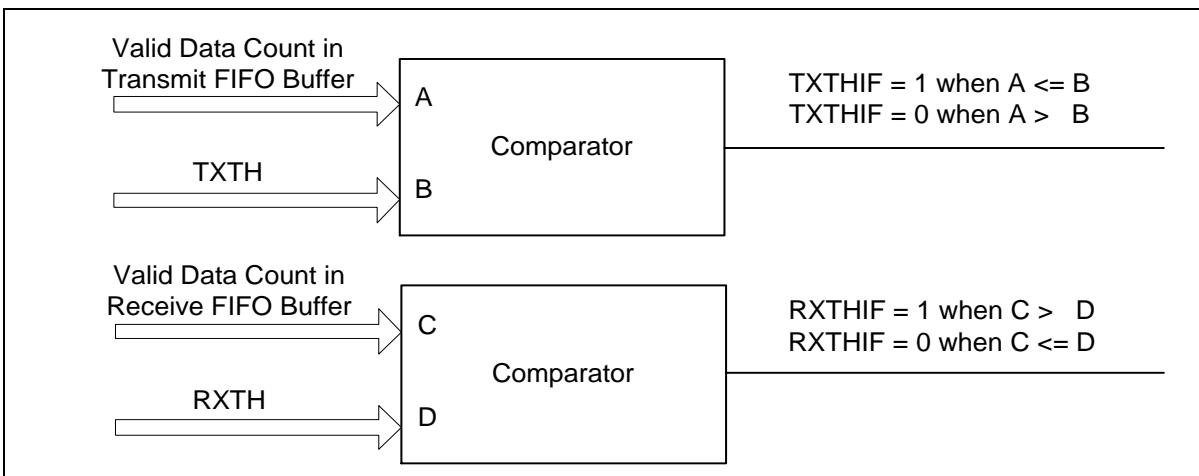


图 6.19-12 FIFO 阀值比较器

在主机模式, 当第一个数据写入 SPIx\_TX 寄存器时, TXEMPTY (SPIx\_STATUS[16]) 标志将会被清除为 0。在 1 个 APB 时钟周期和 6 个外设时钟周期后, 发送将开始。用户可以立即写下一个数据到 SPIx\_TX 寄存器。SPI 控制器将会在两个连续的事务之间插入一个休眠间隔, 休眠间隔的长度由 SUSPITV (SPIx\_CTL[7:4]) 的设定值决定。如果 SUSPITV (SPIx\_CTL[7:4]) 等于 0, SPI 控制器可以执行连续发送。只要 TXFULL (SPIx\_STATUS[17]) 为 0, 用户就可以向 SPIx\_TX 寄存器写入新的数据。

如下图的例 1 所示, 该图指明了 TXEMPTY (SPIx\_STATUS[16]) 的更新条件以及 FIFO 缓存, 移位寄存器和斜移缓存之间的关系。当 Data 0 写入 FIFO 缓存时, TXEMPTY (SPIx\_STATUS[16]) 位被设置为 0。Data 0 将由内核逻辑加载到移位寄存器, 此时, TXEMPTY (SPIx\_STATUS[16]) 将设置为 1。移位寄存器中 Data 0 将逐位移入斜移缓存传输直到传输完成。

在例 2 中, 该图为当 FIFO 缓存中有 8 个数据时, 更新 TXFULL (SPIx\_STATUS[17]) 的条件。当 TXFULL = 1 时, 下一数据 Data 9 不能被写入 FIFO 缓存。

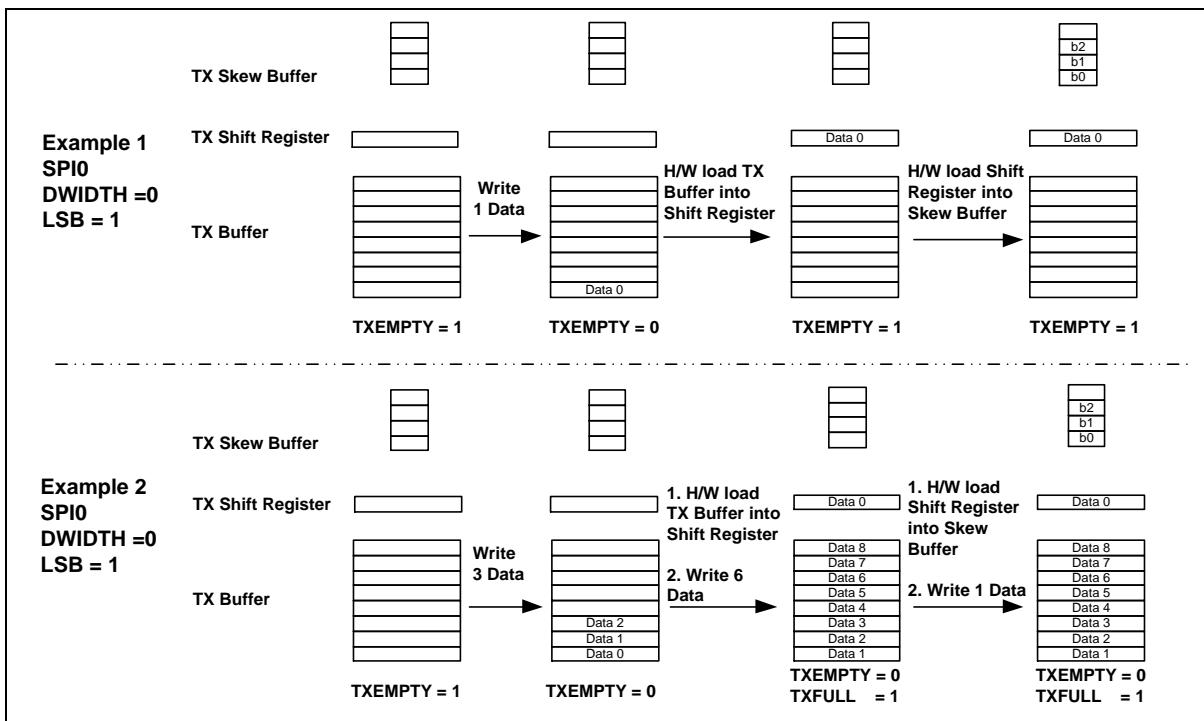


图 6.19-13 发送 FIFO 缓存示例

如果要发送的数据更新及时，接下来的事务将会被自动触发。如果在所有数据传输完成之后，**SPIx\_TX** 寄存器没有被更新，则传输停止。

在主机模式接收操作中，串行数据从 **SPIx\_MISO** 管脚接收并被存储在接收 FIFO 缓存。

接收数据 (Data 0's b0, b1, ...b31) 通过串行时钟 (**SPIx\_CLK**) 先存到斜移缓存，然后再逐位移到移位寄存器。当接收数据位达到 DWIDTH (**SPIx\_CTL[12:8]**) 的值时，内部逻辑将移位寄存器中的数据加载到 FIFO 缓存。当接收 FIFO 缓存有未读数据时 (参见示例 1)，**RXEMPTY** (**SPIx\_STATUS[8]**) 将清 0。只要 **RXEMPTY** (**SPIx\_STATUS[8]**) 为 0，用户就可以通过 **SPIx\_RX** 寄存器来读取接收的数据。如果接收 FIFO 缓存包含 8 个未读数据，**RXFULL** (**SPIx\_STATUS[9]**) 将设置为 1 (参见示例 2)。

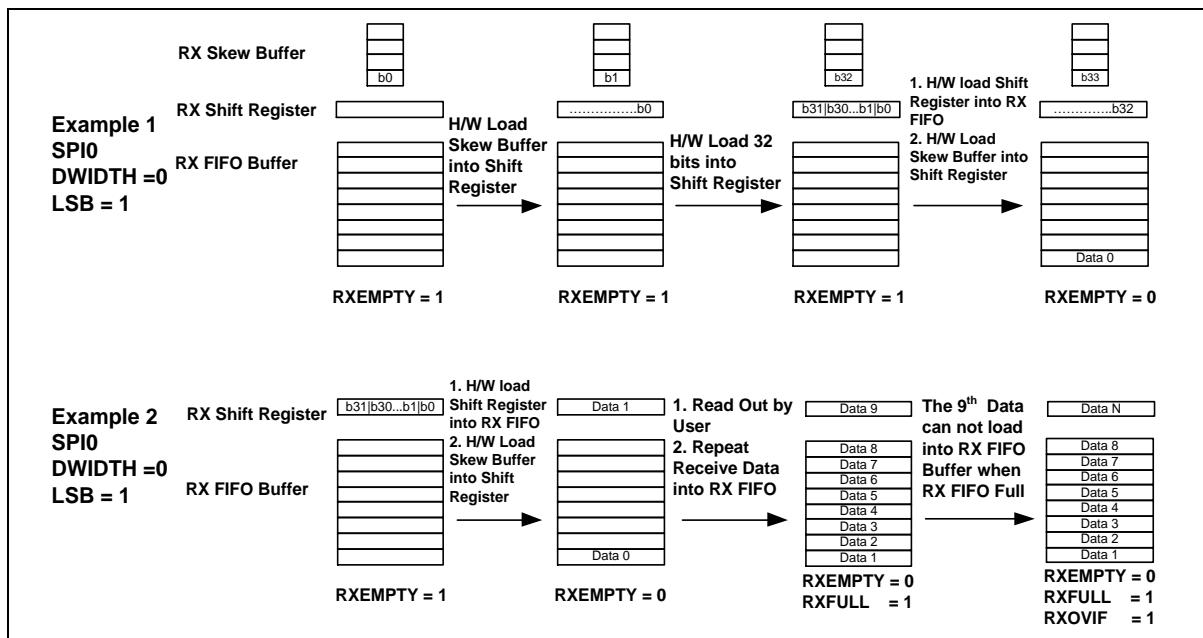


图 6.19-14 接收 FIFO 缓存示例

在从机模式中，当软件写数据到 SPIx\_TX 寄存器时，数据将会被加载到发送 FIFO 缓存，同时 TXEMPTY (SPIx\_STATUS[16]) 标志也将被设置为 0。当从设备从主机接收到时钟信号时，发送操作将会开始。只要 TXFULL (SPIx\_STATUS[17]) 标志为 0，用户就可以写数据到 SPIx\_TX 寄存器。所有数据都被 SPI 发送逻辑单元发送出去后，而且软件没有再更新 SPIx\_TX 寄存器，TXEMPTY (SPIx\_STATUS[16]) 标志将会被设置为 1。

当从机片选信号有效时，如果没有任何数据写入 SPIx\_TX 寄存器，发送下溢标志 TXUFIF (SPIx\_STATUS[19]) 将被设置为 1。输出的数据在本次传输中将会通过设置 TXUFPOL (SPIx\_FIFOCTL[6]) 保留，直到从机片选信号为无效状态。当传送下溢事件发生时，从机溢出运行标志 SLVURIF (SPIx\_STATUS[7]) 将置为 1。同时 SPIx\_SS 进入无效状态。

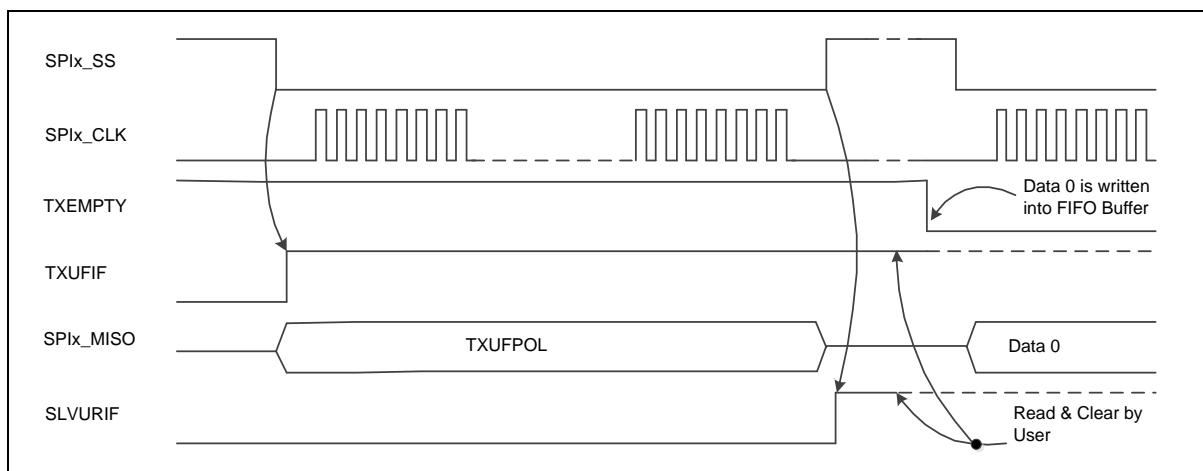


图 6.19-15 TX 下溢事件和从机溢出事件

在从机模式接收操作中，串行数据从 SPIx\_MOSI 管脚接收并存储到 SPIx\_RX 寄存器。接收机制类似于主机模式的接收操作。如果接收 FIFO 缓存包含 4 个未读数据，RXFULL (SPIx\_STATUS[9]) 将被置为 1，RXOVIF (SPIx\_STATUS[11]) 也会被置为 1。如果在 SPIx\_MOSI 管脚上有更多串行数据要接收，接下来的数据将被丢掉（参看接收 FIFO 缓存的示例图）。当从机片选线进入无效状态时，如果接收

到位数数据与 DWIDTH (SPIx\_CTL[12:8]) 设定的不一致, SLVBEIF (SPIx\_STATUS[6]) 将设置为 1。.

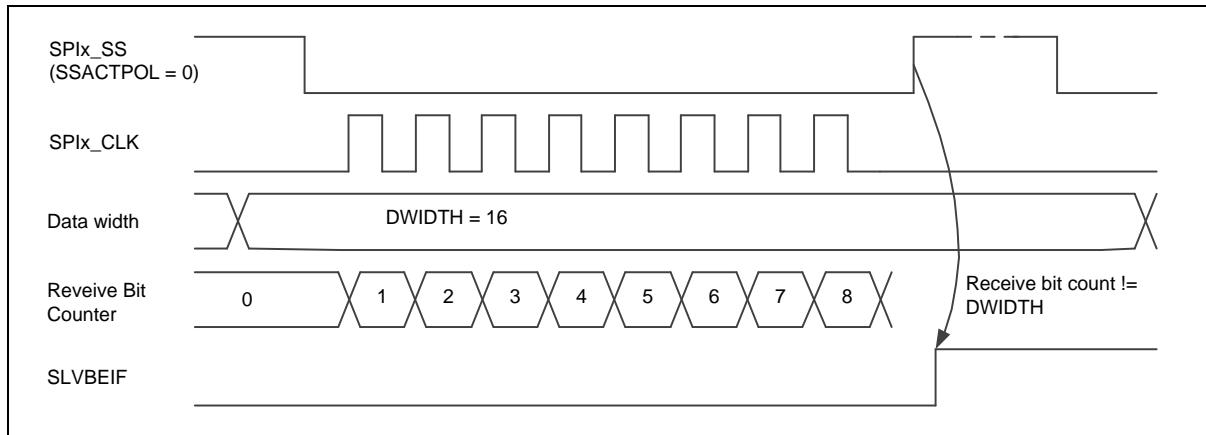


图 6.19-16 从机模式位计数错误

控制器内置了一个接收超时功能模块。当接收 FIFO 非空, 而且在主机模式时接收 FIFO 超过 64 个 SPI 外设时钟周期或者从机模式时超过 576 个 SPI 外设时钟周期没有读操作, 则接收超时发生, RXTOIF (SPI\_STATUS[12]) 置 1。当接收 FIFO 有用户读取时, 超时状态将自动清除。

#### 6.19.5.8 中断

- SPI 单元传输中断

当 SPI 控制器完成一个单元传输, 单元传输中断标志 UNITIF (SPIx\_STATUS[1]) 将会被设为 1。如果单元传输中断使能位 UNITIEN (SPI\_CTL[17]) 被置位, 则单元传输中断事件将给 CPU 产生中断请求。单位传输中断标志位只能写 1 清零。.

- SPI 从机片选有效/无效中断

从机模式下, 如果 SPIEN (SPIx\_CTL[0]) 和 SLAVE (SPIx\_CTL[18]) 置 1, 从机片选信号进入有效/无效状态, 从机片选有效/无效中断标志 SSACTIF (SPIx\_STATUS[2]) 和 SSINAIF (SPIx\_STATUS[3]) 会被置位。此时如果 SSINAIEN (SPIx\_SSCTL[13]) 或 SSACTIEN (SPIx\_SSCTL[12]) 为 1, SPI 控制器会产生中断。

- 从机位计数错误中断

在从机模式, 当从机片选信号线进入无效状态时, 如果发送或接收到位数据个数与 DWIDTH (SPIx\_CTL[12:8]) 设置的不一致, SLVBEIF (SPIx\_STATUS[6]) 将被设置为 1。未传输完成的数据将在 TX 和 RX 移位寄存器中弃除。如果 SLVBEIEN (SPIx\_SSCTL[8]) 位为 1, SPI 控制器将发生一个中断。

**注:**如果从机片选信号激活, 但是没有任何串行时钟输入, 当从机片选信号进入无效状态时, SLVBEIF (SPIx\_STATUS[6]) 也将设置为 1。

- TX 下溢中断

在 SPI 从机模式, 如果没有任何数据写入 SPIx\_TX 寄存器, 当从机选择信号激活时, TXUFIF (SPIx\_STATUS[19]) 将设置为 1。如果 TXUFIEN (SPIx\_FIFOCTL[7]) 为 1, SPI 控制器将发生发送下溢中断。

**注:** SPI 从机模式发生下溢事件后, 有两种方法可以使其恢复空闲状态并进入下个传输: (1) 置 TXRST 为 1 (2) 从机片选信号切为无效状态。

- 从机 TX 下溢运行中断

当 SPIx\_SS 进入无效状态时, 如果有 TX 下溢事件发生, SLVURIF (SPIx\_STATUS[7]) 将被设

置为 1。如果 **SLVURIEN** (**SPIx\_SSCTL[9]**) 为 1, SPI 控制器将发生发送溢出运行中断。

注意：在从机三线模式，从机片选信号被认为是一直有效的，用户需要去查看 **TXUFIF** (**SPI\_STATUS[19]**) 位来确定是否有发生 TX 下溢事件。

- 接收溢出中断

在从机模式，如果接收 FIFO 缓存已有 4 个未读数据，**RXFULL** (**SPIx\_STATUS[9]**) 和 **RXOVIF** (**SPIx\_STATUS[11]**) 标志将会被设置为 1。如果从 SPI 总线上接收到更多串行数据，多余的数据将会丢失。如果 **RXOVIEN** (**SPIx\_FIFOCTL[5]**) 为 1, SPI 控制器将发生接收溢出中断。

- 接收 FIFO 超时中断

如果在 FIFO 里有一个接收到的数据，在主机模式下用户超过 64 个 SPI 外设时钟周期没有去读取，或者从机模式下超过 576 个 SPI 外设时钟周期没有去读取，如果接收超时中断使能位 **RXTOIEN** (**SPIx\_FIFOCTL[4]**) 有设置为 1，则会向系统发出一个 RX 超时中断。

- 发送 FIFO 中断

在 FIFO 模式，如果发送 FIFO 缓存的有效数据少于或等于 **TXTH** (**SPIx\_FIFOCTL[30:28]**) 的设定值，发送 FIFO 中断标志 **TXTHIF** (**SPIx\_STATUS[18]**) 会被置 1。如果发送 FIFO 中断位 **TXTHIEN** (**SPIx\_FIFOCTL[3]**) 为 1，则 SPI 控制器会向系统产生一个发送 FIFO 中断。

- 接收 FIFO 中断

在 FIFO 模式，如果接收 FIFO 缓存的有效数据大于 **RXTH** (**SPIx\_FIFOCTL[26:24]**) 的设定值，接收 FIFO 中断标志 **RXTHIF** (**SPIx\_STATUS[10]**) 会被设置为 1。如果接收 FIFO 中断位 **RXTHIEN** (**SPIx\_FIFOCTL[2]**) 有设置为 1，SPI 控制器将会向系统产生一个接收 FIFO 中断。.

#### 6.19.5.9 I2S 模式

SPI0~SPI3 控制器支持 I2S 模式，支持 PCM 模式 A, PCM 模式 B 和 MSB 对齐以及 I2S 数据格式。音频通道的位宽由 **WDWIDTH** (**SPIx\_I2SCTL[5:4]**) 设定。传输顺序一般为 MSB (最高位优先)。数据在时钟线的上升沿时读取下降沿时传送。

在 I2S 数据格式中，最高位优先传送，在音频通道的第二个时钟发送。**I2S\_LRCLK** 信号指示正在传输的音频通道。.

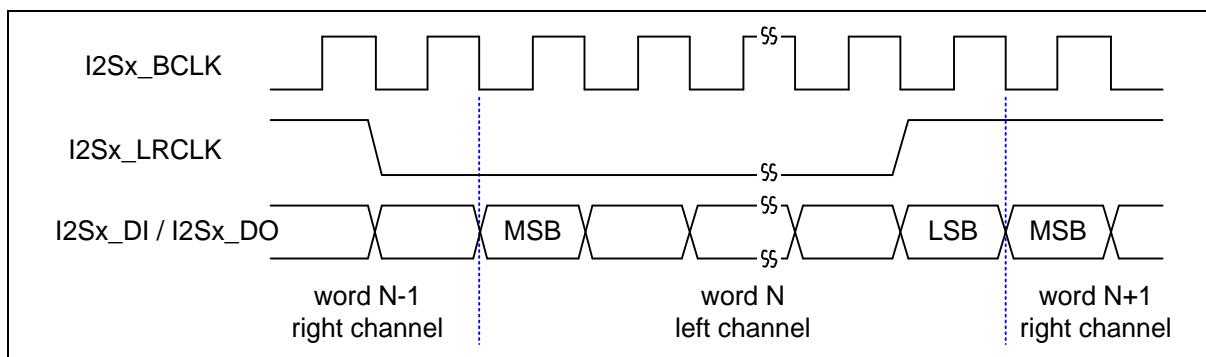


图 6.19-17 I2S 数据格式时序

在 MSB 对齐数据格式，最高位优先传送，在音频通道的第一个时钟发送。.

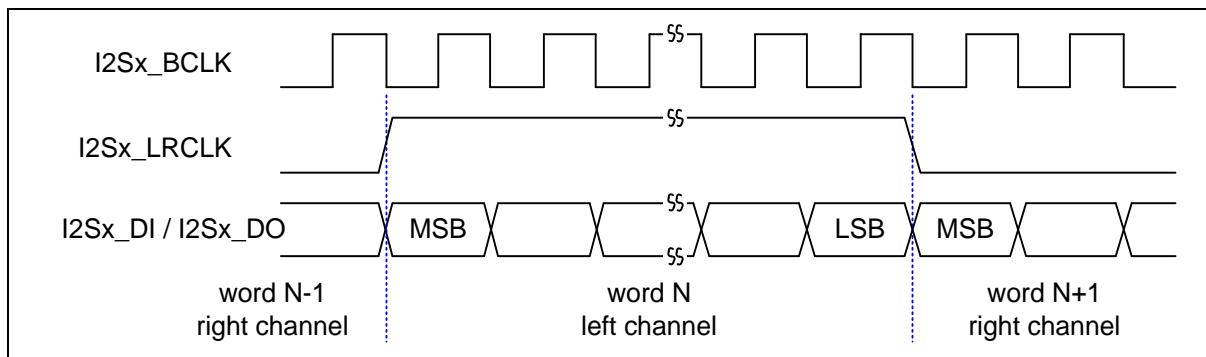


图 6.19-18 MSB 对齐数据格式时序图

I2Sx\_LRCLK 也支持 PCM 模式 A 和 PCM 模式 B。PCM 模式下 I2Sx\_LRCLK 用来指示音频帧的开始。

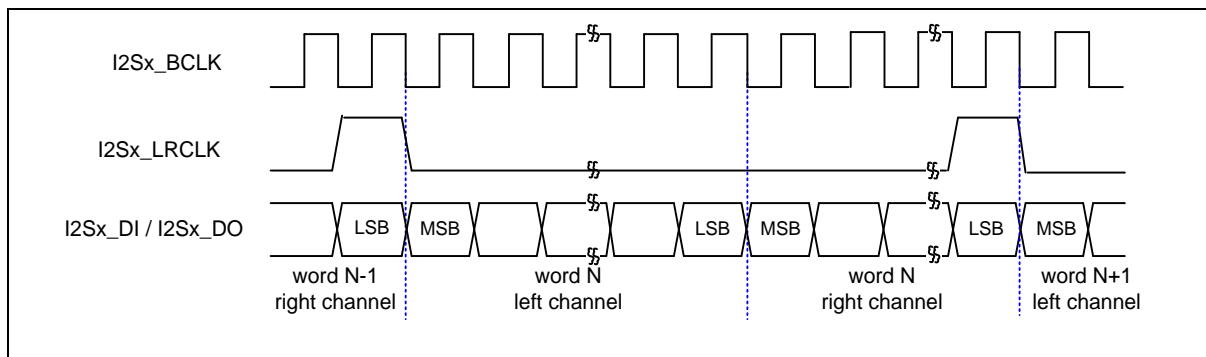


图 6.19-19 PCM 模式 A 时序图

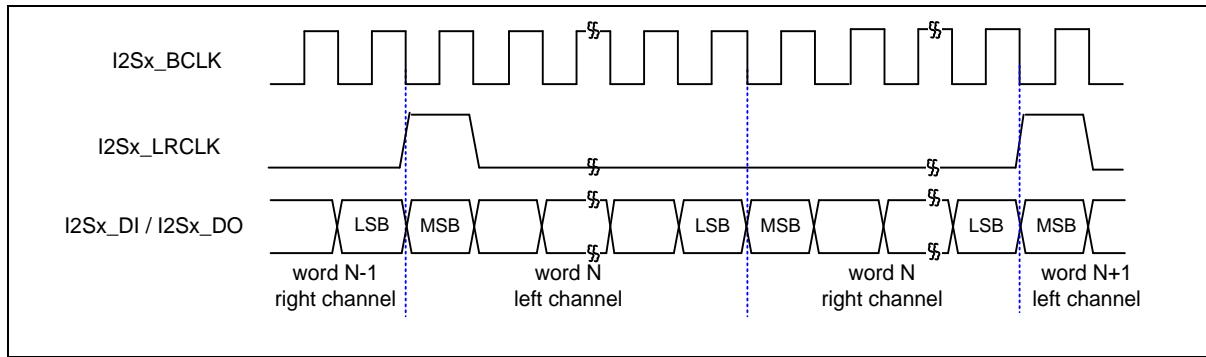


图 6.19-20 PCM 模式 B 时序图

## 6.19.5.10 I2S 模式 FIFO 操作

Mono 8-bit data mode

7	N+3	0	7	N+2	0	7	N+1	0	7	N	0
---	-----	---	---	-----	---	---	-----	---	---	---	---

Stereo 8-bit data mode, ORDER (SPIx\_I2SCTL[7]) = 0

7	LEFT+1	0	7	RIGHT+1	0	7	LEFT	0	7	RIGHT	0
---	--------	---	---	---------	---	---	------	---	---	-------	---

Stereo 8-bit data mode, ORDER (SPIx\_I2SCTL[7]) = 1

7	RIGHT+1	0	7	LEFT+1	0	7	RIGHT	0	7	LEFT	0
---	---------	---	---	--------	---	---	-------	---	---	------	---

Mono 16-bit data mode

15	N+1	0	15	N	0
----	-----	---	----	---	---

Stereo 16-bit data mode, ORDER (SPIx\_I2SCTL[7]) = 0

15	LEFT	0	15	RIGHT	0
----	------	---	----	-------	---

Stereo 16-bit data mode, ORDER (SPIx\_I2SCTL[7]) = 1

15	RIGHT	0	15	LEFT	0
----	-------	---	----	------	---

Mono 24-bit data mode

23	N	0
----	---	---

Stereo 24-bit data mode

23	LEFT	0	N
----	------	---	---

23	RIGHT	0	N+1
----	-------	---	-----

Mono 32-bit data mode

31	N	0
----	---	---

Stereo 32-bit data mode

31	LEFT	0	N
----	------	---	---

31	RIGHT	0	N+1
----	-------	---	-----

图 6.19-21 不同 I2S 模式下的 FIFO 内容

### 6.19.6 时序图

从机选择信号的有效状态可以由 SSACTPOL (SPIx\_SSCTL[2]) 配置。串行时钟的空闲状态可以通过 CLKPOL (SPIx\_CTL[3]) 配置为高电平或低电平。传输字段长度在 DWIDTH (SPIx\_CTL[12:8]) 中定义，发送/接收数据是以 MSB 还是 LSB 优先由 LSB 位 (SPIx\_CTL[13]) 定义。用户也可以通过 TXNEG/RXNEG (SPIx\_CTL[2:1]) 来选择发送/接收数据时串行时钟的边沿。四种 SPI 主机/从机操作时序图以及相关设置如下图。

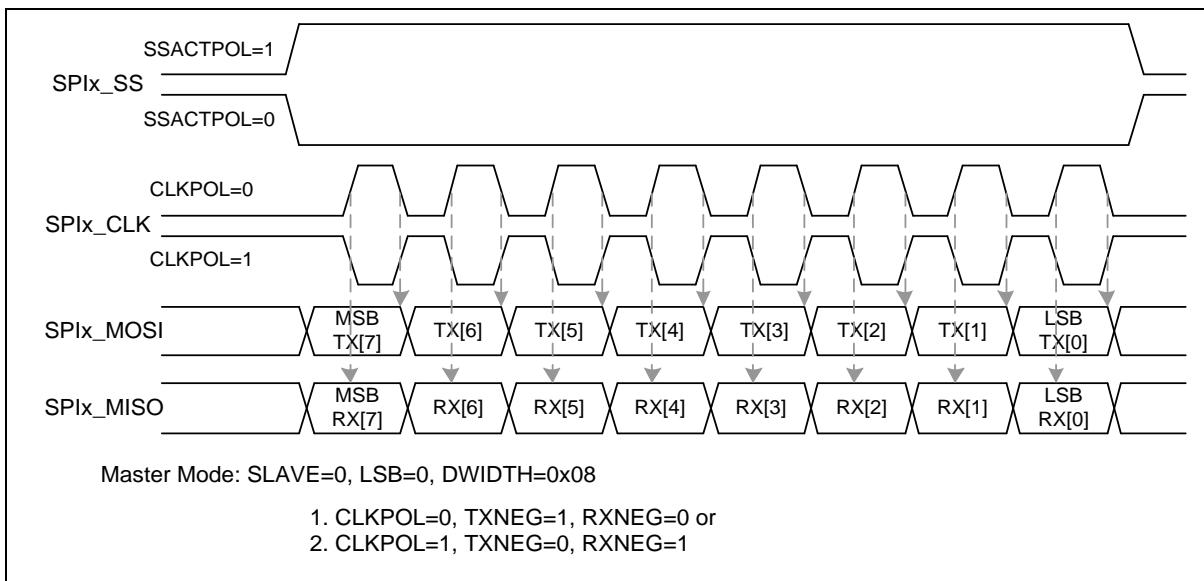


图 6.19-22 SPI 主机模式时序

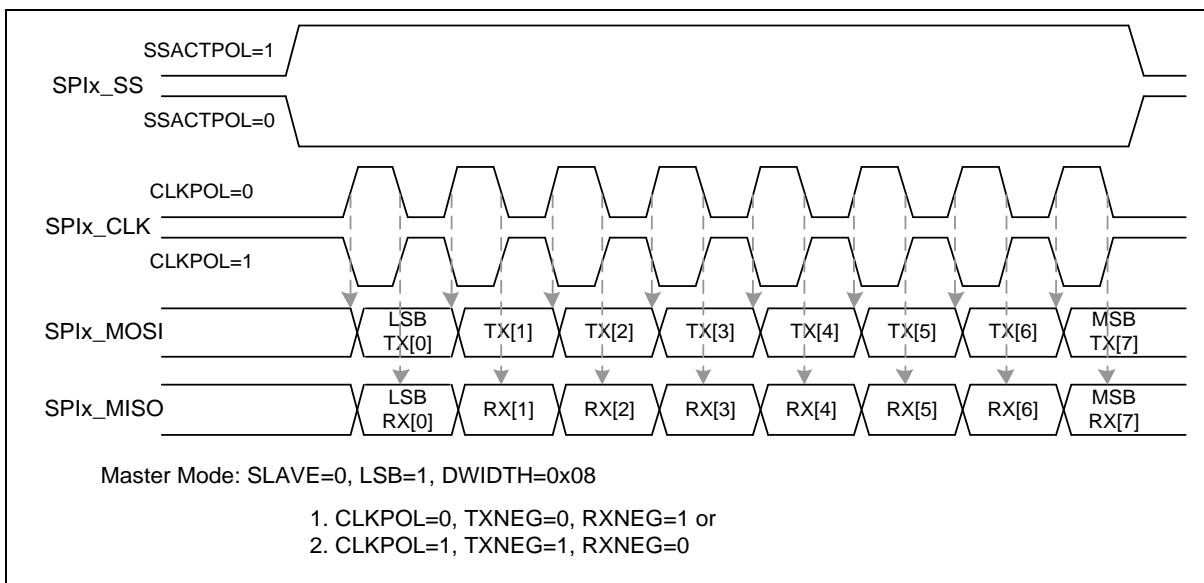


图 6.19-23 SPI 主机模式时序 (SPIx\_CLK 交替相位)

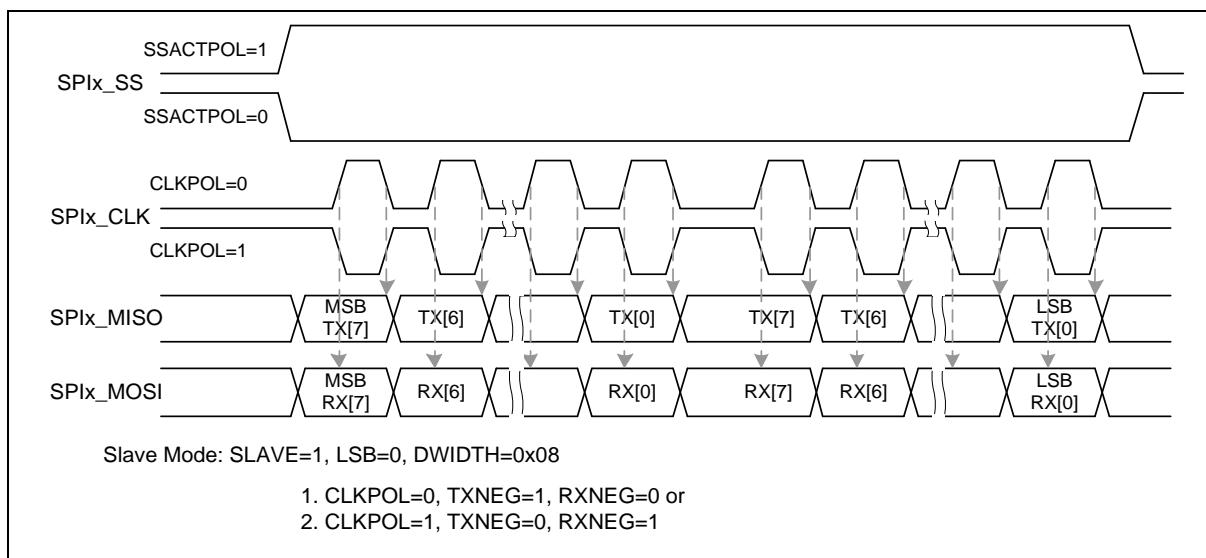
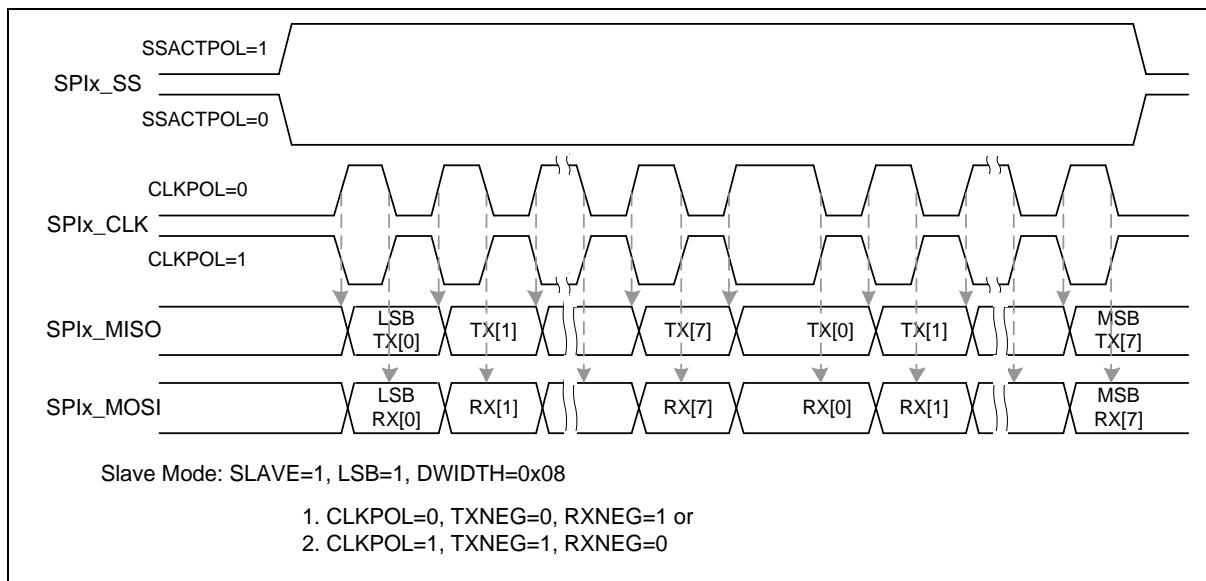


图 6.19-24 SPI 从机模式时序

图 6.19-25 SPI 从机模式时序 (SPI<sub>x</sub>\_CLK 交替相位)

### 6.19.7 编程示例

#### 例 1:

SPI 配置为全双工主机时序如下：

- SPI 时钟上沿锁存数据
- SPI 时钟下沿发送数据
- MSB 优先
- SPI 时钟空闲时处于低电平
- 每次只收发一个字节

- 片选信号低电平有效。

操作流程如下：

1. 在寄存器 DIVIDER (SPIx\_CLKDIV [8:0]) 配置 SPI 输出时钟频率
2. 写寄存器 SPIx\_SSCTL 配置 SPI 主机相关模式：
  - 1) 清 0 AUTOSS (SPIx\_SSCTL[3]) 关闭自动从机选择功能。
  - 2) 清 0 SSACTPOL (SPIx\_SSCTL[2]) 设置片选信号低电平有效。
  - 3) 置 SS (SPIx\_SSCTL[0]) 为 1 使能从机片选信号，激活片外从机设备。
3. 写寄存器 SPIx\_CTL 控制 SPI 主机行为：
  - 1) 设 SLAVE (SPIx\_CTL[18]) 为 0 配置 SPI 为主机设备。
  - 2) 清 0 CLKPOL (SPIx\_CTL[3]) 强制 SPI 空闲状态为低电平。
  - 3) 置 TXNEG (SPIx\_CTL[2]) 为 1 配置 SPI 总线时钟下沿传输数据。
  - 4) 清 0 RXNEG (SPIx\_CTL[1]) 配置 SPI 总线时钟上沿锁存数据。
  - 5) 设置传输长度 DWIDTH 为 8 位 (SPIx\_CTL[12:8] = 0x08)。
  - 6) 清 0 LSB (SPIx\_CTL[13]) 配置 MSB 传输优先。
4. 置 SPIEN (SPIx\_CTL[0]) 位 1 使能数据传输。
5. SPI 主机想要写一个字节的数据到片外设备时，将数据写到 SPIx\_TX 寄存器即可。
6. 等待 SPI 中断发生 (如果 UNITIEN (SPIx\_CTL[17]) 为 1)，或者轮询单元传输中断标志 UNITIF (SPIx\_STATUS[1])。
7. 从 SPIx\_RX 读接收到的数据。
8. 重复步骤 5 继续其他数据的传输，或者配置 SS (SPIx\_SSCTL[0]) 为 0 让片选信号无效以关闭片外从机设备。.

## 例 2：

SPI 为全双工从机。片外主机按下面的规则通讯：

- SPI 时钟上沿锁存数据
- SPI 时钟下沿发送数据
- LSB 优先
- SPI 时钟空闲时处于高电平
- 每次只收发一个字节
- 片选信号高电平有效。

操作流程如下：

1. 写寄存器 SPIx\_SSCTL 配置为从机模式。
2. 置 SSACTPOL (SPIx\_SSCTL[2]) 为 1 配置片选信号高电平有效。.
3. 写 SPIx\_CTL 配置从机行为
  - 1) 设 SLAVE (SPIx\_CTL[18]) 为 1 配置 SPI 控制球为从机设备。
  - 2) 置 CLKPOL (SPIx\_CTL[3]) 为 1 选择 SPI 空闲状态为高电平。

- 3) 置 TXNEG (SPIx\_CTL[2]) 为 1 配置 SPI 总线时钟下降沿传输数据。
- 4) 清 0 RXNEG (SPIx\_CTL[1]) 配置 SPI 总线时钟上升沿锁存数据。
- 5) 设置传输长度 DWIDTH 为 8 位 (SPIx\_CTL[12:8] = 0x08)。
4. 置 LSB (SPIx\_CTL[13]) 为 1 配置 LSB 传输优先
5. 置 SPIEN (SPIx\_CTL[0]) 为 1。等待片外主机设备的片选信号和 SPI 时钟输入来启动数据传输
6. 如果 SPI 从机想要发送一个字节的数据到片外主机，写数据到 SPIx\_TX 寄存器即可。
7. 如果 SPI 从机想要从片外主机接收一个字节的数据，且用户不关心什么数据被传输，则软件不需要去更新 SPIx\_RX 寄存器
8. 等待 SPI 中等待 SPI 中断发生 (如果 UNITIEN (SPIx\_CTL[17]) 为 1)，或者轮询单元传输中断标志 UNITIF (SPIx\_STATUS[1]) .
9. 从 SPIx\_RX 寄存器读接收到的数据。.
10. 回到 7 继续下个字节发送或停止发送

### 6.19.8 寄存器映射

R: 只读, W: 只写, R/W: 读/写

寄存器	偏移量	R/W	描述	复位值
<b>SPI 基址:</b>				
<b>SPIx_BA = 0x4006_1000 + (0x0000_1000 * x)</b>				
x=0, 1, 2, 3				
<b>SPIx_CTL</b>	SPIx_BA+0x00	R/W	SPI 控制寄存器	0x0000_0034
<b>SPIx_CLKDIV</b>	SPIx_BA+0x04	R/W	SPI 时钟分频寄存器	0x0000_0000
<b>SPIx_SSCTL</b>	SPIx_BA+0x08	R/W	SPI 从机选择控制寄存器	0x0000_0000
<b>SPIx_PDMACTL</b>	SPIx_BA+0x0C	R/W	SPI PDMA 控制寄存器	0x0000_0000
<b>SPIx_FIFOCTL</b>	SPIx_BA+0x10	R/W	SPI FIFO 控制寄存器	0x2200_0000
<b>SPIx_STATUS</b>	SPIx_BA+0x14	R/W	SPI 状态寄存器	0x0005_0110
<b>SPIx_TX</b>	SPIx_BA+0x20	W	SPI 数据发送寄存器	0x0000_0000
<b>SPIx_RX</b>	SPIx_BA+0x30	R	SPI 数据接收寄存器	0x0000_0000
<b>SPIx_I2SCTL</b>	SPIx_BA+0x60	R/W	I2S 控制寄存器	0x0000_0000
<b>SPIx_I2SCLK</b>	SPIx_BA+0x64	R/W	I2S 时钟分频控制寄存器	0x0000_0000
<b>SPIx_I2SSSTS</b>	SPIx_BA+0x68	R/W	I2S 状态寄存器	0x0005_0100

## 6.19.9 寄存器描述

**SPIx\_CTL** SPI 控制寄存器

寄存器	偏移量	R/W	描述	复位值
SPIx_CTL	SPIx_BA+0x00	R/W	SPI 控制寄存器	0x0000_0034

Note: Not supported in I<sup>2</sup>S mode.

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved			DATDIR	REORDER	SLAVE	UNITIEN	Reserved
15	14	13	12	11	10	9	8
RXONLY	HALFDPX	LSB	DWIDTH				
7	6	5	4	3	2	1	0
SUSPITV			CLKPOL	TXNEG	RXNEG	SPIEN	

位	描述	
[31:21]	Reserved	保留
[20]	DATDIR	<b>数据方向控制</b> 该位用来选则半双工模式和双/四 I/O 模式下数据输入/输出方向 0 = SPI 数据输入 1 = SPI 数据输出
[19]	REORDER	<b>字节重排功能使能位</b> 0 = 字节重排功能禁止 1 = 字节重排功能使能. 每两个字节之间会被插入一个休眠间隔, 休眠间隔的时间取决于 SUSPITV 的设置 注: 仅在 DWIDTH 为 16, 24 和 32 位时, 该功能有效。
[18]	SLAVE	<b>从机模式控制</b> 0 = 主机模式. 1 = 从机模式
[17]	UNITIEN	<b>单元传输中断使能位</b> 0 = SPI 单元传输中断禁止 1 = SPI 单元传输中断使能
[16]	Reserved	保留
[15]	RXONLY	<b>只接收模式使能位 (仅主机)</b> 该只用于主机模式。在只接收模式, SPI 主机总线时钟会一直保持以接收数据, 忙标志会一直置位。 0 = 只接收模式关闭 1 = 只接收模式使能
[14]	HALFDPX	<b>SPI 半双工传输使能位</b>

		该位用来切换 SPI 传输全双工模式和半双工模式。半双工模式下 DATDIR (SPIx_CTL[20]) 用来选择数据传输方向。 0 = SPI 工作在全双工模式 1 = SPI 工作在半双工模式
[13]	<b>LSB</b>	<b>LSB 优先发送</b> 0 = MSB, 具体发送/接收寄存器的哪一位首先被发送/接收, 取决于 DWIDTH 的设定值。 1 = LSB, SPI TX 寄存器的位 0, 首先被发送到 SPI 数据输出管脚, 从 SPI 数据输入管脚上接收到的第一个数据位将会被放置到 RX 寄存器 (SPI_RX 的位 0) LSB 的位置.
[12:8]	<b>DWIDTH</b>	<b>数据宽度</b> 该位用来标示每次传输时会有多少位被发送/接收。最低为 8 位, 最高为 32 位。 DWIDTH = 0x08 .... 8 位. DWIDTH = 0x09 .... 9 位. ..... DWIDTH = 0x1F .... 31 位. DWIDTH = 0x00 .... 32 位. <b>注:</b> 对于 SPI0~SPI3 SPI 模式, 该位决定了 TX/RX FIFO 的深度。因此, 修改该位时, 硬件会自动清空 SPI0~SPI3 的 TX/RX FIFO。
[7:4]	<b>SUSPITV</b>	<b>休眠间隔 (仅主机)</b> 该四位用来配置在一次数据传输过程中连续两个发送/接收事务之间的休眠间隔。休眠间隔是从当前事务字的最后一个时钟边沿到接下来的事务的第一个边沿时钟。默认值是 0x3. 休眠间隔的周期可以根据下面公式获得: $(SUSPITV[3:0] + 0.5) * \text{SPICLK 时钟周期}$ 例: SUSPITV = 0x0 .... 0.5 SPICLK 时钟周期 SUSPITV = 0x1 .... 1.5 SPICLK 时钟周期 ..... SUSPITV = 0xE .... 14.5 SPICLK 时钟周期 SUSPITV = 0xF .... 15.5 SPICLK 时钟周期
[3]	<b>CLKPOL</b>	<b>时钟极性</b> 0 = SPI 总线空闲时默认低 1 = SPI 总线空闲时默认高
[2]	<b>TXNEG</b>	<b>下降沿发送</b> 0 = 在 SPI 总线时钟的上升沿发送数据 1 = 在 SPI 总线时钟的下降沿发送数据
[1]	<b>RXNEG</b>	<b>下降沿接收</b> 0 = 在 SPI 总线时钟的上升沿锁存数据 1 = 在 SPI 总线时钟的下降沿锁存数据
[0]	<b>SPIEN</b>	<b>SPI 传输控制使能位</b> 在主机模式下, FIFO 缓存有数据时, 该位设置为 1 后, 开始传输。在从机模式下, 该位置为 1 时, 该设备已准备好接收数据。 0 = 禁止控制传输 1 = 使能控制传输

**SPIx\_CLKDIV SPI 时钟分频寄存器**

寄存器	偏移量	R/W	描述	复位值
SPIx_CLKDIV	SPIx_BA+0x04	R/W	SPI 时钟分频寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
DIVIDER							

位	描述	
[31:9]	Reserved	保留
[8:0]	DIVIDER	<p><b>时钟除频</b></p> <p>该域的值是产生外设时钟，<math>f_{spi\_eclk}</math> 和 SPI 主机的总线时钟的除频器的值。频率计算公式如下：</p> $f_{spi\_eclk} = \frac{f_{spi\_clock\_src}}{(DIVIDER + 1)}$ <p>其中，  <math>f_{spi\_clock\_src}</math> 为外设时钟源，定义于时钟控制寄存器 CLK_CLKSEL2 中  注：I<sup>2</sup>S 模式不支持</p>

**注意:** 谨慎配置 DIVIDER，SPI 时钟频率不能大于系统时钟频率

SPI<sub>x</sub> SSCTL SPI 从机选择控制寄存器

寄存器	偏移量	R/W	描述	复位值
SPI <sub>x</sub> SSCTL	SPI <sub>x</sub> _BA+0x08	R/W	SPI 从机选择控制寄存器	0x0000_0000

注: I<sup>2</sup>S 模式不支持.

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved		SSINAIEN	SSACTIEN	Reserved		SLVURIEN	SLVBEIEN
7	6	5	4	3	2	1	0
Reserved				AUTOSS	SSACTPOL	Reserved	SS

位	描述	
[31:14]	Reserved	保留
[13]	SSINAIEN	从机片选无效中断使能位 0 = 从机片选无效中断禁止 1 = 从机片选无效中断使能
[12]	SSACTIEN	从机片选有效中断使能位 0 = 从机片选有效中断禁止 1 = 从机片选有效中断使能
[11:10]	Reserved	保留
[9]	SLVURIEN	从机模式 TX 下溢中断使能位 0 = 从机模式 TX 下溢中断禁止 1 = 从机模式 TX 下溢中断使能
[8]	SLVBEIEN	从机模式位计数错误中断使能位 0 = 从机模式位计数错误中断禁止 1 = 从机模式位计数错误中断使能
[7:4]	Reserved	保留
[3]	AUTOSS	自动从机片选功能使能位 (仅主机) 0 = 自动从机片选功能禁止, 从机片选信号由 SS (SPI <sub>x</sub> _SSCTL[0]) 控制 1 = 自动从机片选功能使能
[2]	SSACTPOL	从机片选有效极性 该位定义从机片选信号 (SPI <sub>x</sub> _SS) 有效的极性 0 = 片选信号 SPI <sub>x</sub> _SS 低电平有效 1 = 片选信号 SPI <sub>x</sub> _SS 高电平有效
[1]	Reserved	保留

[0]	<b>SS</b>	从机片选控制位 (仅主机) 如果 AUTOSS 为 0, 0 = 设置 SPIx_SS 为无效状态 1 = 设置 SPIx_SS 为有效状态 如果 AUTOSS 置 1,
-----	-----------	---

SPIx\_PDMACTL SPI PDMA 控制寄存器

寄存器	偏移量	R/W	描述	复位值
SPIx_PDMACTL	SPIx_BA+0x0C	R/W	SPI PDMA 控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved					PDMARST	RXPDMAEN	TXPDMAEN

位	描述	
[31:3]	Reserved	保留
[2]	PDMARST	<b>PDMA 复位</b> 0 = 无效 1 = 复位 SPI 控制器的 PDMA 控制逻辑。该位自动清 0。
[1]	RXPDMAEN	<b>PDMA 接收使能位</b> 0 = PDMA 接收禁止 1 = PDMA 接收使能
[0]	TXPDMAEN	<b>PDMA 发送使能位</b> 0 = PDMA 发送禁止 1 = PDMA 发送使能 <b>注:</b> 在 SPI 主机模式支持全双工传输，如果发送和接收 PDMA 功能都使能，接收 PDMA 功能不能在发送 PDMA 功能之前使能。用户可以先使能发送 PDMA 功能或者同时使能两个功能。

**SPIx\_FIFOCTL SPI FIFO 控制寄存器**

寄存器	偏移量	R/W	描述	复位值
SPIx_FIFOCTL	SPIx_BA+0x10	R/W	SPI FIFO 控制寄存器	0x2200_0000

31	30	29	28	27	26	25	24
Reserved	TXTH				Reserved	RXTH	
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved						TXFBCLR	RXFBCLR
7	6	5	4	3	2	1	0
TXUFLEN	TXUFPOL	RXOVIEN	RXTOIEN	TXTHIEN	RXTHIEN	TXRST	RXRST

位	描述	
[31]	Reserved	保留
[30:28]	TXTH	<b>发送 FIFO 阈值</b> 如果发送 FIFO 缓存的有效数据个数小于或者等于 TXTH 的设置, TXTHIF 将会被设置为 1, 否则 TXTHIF 将会被设置为 0。对于 SPI0~SPI3, 该域的 MSB 只在 SPI 数据长度为 8~16 时有效。
[27]	Reserved	保留
[26:24]	RXTH	<b>接收 FIFO 域值</b> 如果接收 FIFO 缓存的有效数据个数大于 RXTH 的设置, RXTHIF 将会被设置为 1, 否则 RXTHIF 将会被设置为 0。对于 SPI0~SPI3, 该域的 MSB 只在 SPI 数据长度为 8~16 时有效。
[23:10]	Reserved	保留
[9]	TXFBCLR	<b>清发送 FIFO 缓存</b> 0 = 无效 1 = 清除发送 FIFO 指针, TXFULL 位将清除为 0 而且 TXEMPTY 位也将设置为 1。该位置 1 后, 在一个系统时钟后, 由硬件清 0。 <b>注:</b> TX 移位寄存器不会清空
[8]	RXFBCLR	<b>清接收 FIFO 缓存</b> 0 = 无效 1 = 清除接收 FIFO 指针, RXFULL 位将清除为 0 而且 RXEMPTY 位将设置为 1。该位设置为 1 后, 在一个系统时钟后, 由硬件清 0。 <b>注:</b> RX 移位寄存器不会清空
[7]	TXUFLEN	<b>TX 下溢中断使能位</b> 从机模式下 TX 下溢中断事件发生时, TXUFIF (SPIx_STATUS[19]) 会被置 1。该位用来使能 TX 下溢中断。 0 = 从机 TX 下溢中断禁止

		1 = 从机 TX 下溢中断使能
[6]	<b>TXUFFPOL</b>	<p><b>TX 下溢数据极性</b></p> <p>0 = 在从机模式，如果有发送下溢事件发生，SPI 数据输出保持为 0。      1 = 在从机模式，如果有发送下溢事件发生，SPI 数据输出保持为 1。</p> <p><b>注：</b></p> <ol style="list-style-type: none"> <li>从机片选信号有效时如果没有任何数据发送则会发生下溢事件。</li> <li>I<sup>2</sup>S 模式下该位需要设为 0。</li> <li>TX 下溢事件发生时，SPI<sub>x</sub>_MISO 管脚状态将由该位决定而不管 TX FIFO 是否为空。存储在 TX FIFO 的数据会在下一个传输帧通过 SPI<sub>x</sub>_MISO 发送出去。</li> </ol>
[5]	<b>RXOVEN</b>	<p><b>接收 FIFO 溢出中断使能位</b></p> <p>0 = 接收 FIFO 溢出中断禁止      1 = 接收 FIFO 溢出中断使能</p>
[4]	<b>RXTOIEN</b>	<p><b>从机接收超时中断使能位</b></p> <p>0 = 从机接收超时中断禁止      1 = 从机接收超时中断使能</p>
[3]	<b>TXTHIEN</b>	<p><b>发送 FIFO 阈值中断使能位</b></p> <p>0 = TX FIFO 阈值中断禁止      1 = TX FIFO 阈值中断使能</p>
[2]	<b>RXTHIEN</b>	<p><b>接收 FIFO 阈值中断使能位</b></p> <p>0 = RX FIFO 阈值中断禁止      1 = RX FIFO 阈值中断使能</p>
[1]	<b>TXRST</b>	<p><b>发送复位</b></p> <p>0 = 无效      1 = 复位发送 FIFO 指针和发送电路。TXFULL 位将被清 0，TXEMPTY 位将被置 1。该位置 1 后，大约经过 3 个系统时钟周期和 2 个外设时钟周期，硬件会将该位清除为 0。如果用户想确认复位是否已经完成，可以读取 TXXRST (SPI_STATUS[23]) 来确认。</p> <p><b>注：</b>如果从机模式下发生 TX 下溢事件发生，该位也可以用来使 SPI 返回空闲状态。</p>
[0]	<b>RXRST</b>	<p><b>接收复位</b></p> <p>0 = 无效      1 = 复位接收 FIFO 指针和接收电路。RXFULL 位将被清除为 0，RXEMPTY 位将被设置为 1。该位置 1 后，大约经过 3 个系统时钟周期和 2 个外设时钟周期，硬件将该位清除为 0。如果用户想确认复位是否已经完成，可以读 TXXRST (SPI_STATUS[23]) 来确认。</p>

**SPIx\_STATUS SPI 状态寄存器**

寄存器	偏移量	R/W	描述	复位值
SPIx_STATUS	SPIx_BA+0x14	R/W	SPI 状态寄存器	0x0005_0110

注: I²S 模式不支持

31	30	29	28	27	26	25	24
TXCNT				RXCNT			
23	22	21	20	19	18	17	16
TXRXRST	Reserved			TXUFIF	TXTHIF	TXFULL	TXEMPTY
15	14	13	12	11	10	9	8
SPIENSTS	Reserved		RXTOIF	RXOVIF	RXTHIF	RXFULL	RXEMPTY
7	6	5	4	3	2	1	0
SLVURIF	SLVBEIF	SLVTOIF	SSLINE	SSINAIF	SSACTIF	UNITIF	BUSY

位	描述	
[31:28]	TXCNT	发送 FIFO 数据个数(只读) 该位域表明发送 FIFO 缓存的有效数据个数。
[27:24]	RXCNT	接收 FIFO 数据个数(只读) 该域表明接收 FIFO 缓存有效数据个数..
[23]	TXRXRST	TX 或 RX 复位状态(只读) 0 = TXRST 或 RXRST 的复位功能已经完成 1 = TXRST 或 RXRST 正在复位 <b>注意:</b> TXRST 和 RXRST 的复位操作都需要 3 个系统时钟周期 + 2 个外设时钟周期。用户可以检查该状态位来检测复位功能是否完成。
[22:20]	Reserved	保留
[19]	TXUFIF	TX 下溢中断标志 当发送下溢事件发生时，该位被设置为 1，数据输出管脚的状态取决于 TXUFPOL 的设置。 0 = 无影响 1 = 当从机片选信号有效时，发送 FIFO 和发送移位寄存器中没有数据。 <b>注 1:</b> 该位写 1 清 0。 <b>注 2:</b> 当从机片选信号有效时，如果复位从机的发送电路，在 3 个系统时钟周期和 2 个外设时钟周期后，此时复位操作已经完成，该位将被设置为 1。
[18]	TXTHIF	发送 FIFO 阈值中断标志(只读) 0 = 发送 FIFO 缓存中的有效数据个数大于 TXTH 设置的值 1 = 发送 FIFO 缓存中的有效数据个数少于或等于 TXTH 设置的值
[17]	TXFULL	发送 FIFO 缓存满标志(只读) 0 = 发送 FIFO 缓存未满 1 = 发送 FIFO 缓存已满
[16]	TXEMPTY	发送 FIFO 缓存空标志(只读)

		0 = 发送 FIFO 缓存不空 1 = 发送 FIFO 缓存已空
[15]	<b>SPIENSTS</b>	<b>SPI 使能状态位 (只读)</b> 0 = 禁止 SPI 控制器. 1 = 使能 SPI 控制器. <b>注意:</b> SPI 外设时钟与系统时钟不同步。为了确保 SPI 控制器已经被禁止, 该位指示了 SPI 控制器的真实状态。
[14:13]	<b>Reserved</b>	保留
[12]	<b>RXTOIF</b>	<b>接收超时中断标志</b> 0 = 没有接收 FIFO 超时事件 1 = 接收 FIFO 缓存非空且主机模式下, 超过 64 个 SPI 时钟周期或从机模式下超过 576 个 SPI 引擎时钟周期, 接收 FIFO 缓存上没有读操作。当接收 FIFO 缓存被软件读取, 超时状态会自动清 0。 <b>注:</b> 该位写 1 清 0
[11]	<b>RXOVIF</b>	<b>接收 FIFO 溢出中断标志</b> 当接收 FIFO 缓存已经满了, 接下来的数据将被丢弃, 同时该位被设置为 1. 0 = 接收 FIFO 没有溢出 1 = 接收 FIFO 溢出 <b>注:</b> 该位写 1 清 0
[10]	<b>RXTHIF</b>	<b>接收 FIFO 阀值中断标志 (只读)</b> 0 = 接收 FIFO 缓存的有效数据个数少于或等于 RXTH 设置的值 1 = 接收 FIFO 缓存的有效数据个数大于 RXTH 设置的值
[9]	<b>RXFULL</b>	<b>接收 FIFO 缓存满标志 (只读)</b> 0 = 接收 FIFO 缓存未满 1 = 接收 FIFO 缓存已满
[8]	<b>RXEMPTY</b>	<b>接收 FIFO 缓存空标志 (只读)</b> 0 = 接收 FIFO 缓存不空 1 = 接收 FIFO 缓存为空
[7]	<b>SLVURIF</b>	<b>从机模式发送下溢中断标志</b> 在从机模式, 如果发送下溢事件发生, 而且从机片选线进入无效状态, 该中断标志将被设置为 1。 0 = 未发生从机发送下溢事件 1 = 发生从机发送下溢事件 <b>注:</b> 该位写 1 清 0
[6]	<b>SLVBEIF</b>	<b>从机模式位计数错误中断标志</b> 在从机模式, 当从机片选信号线进入无效状态时, 如果位计数与 DWIDTH 不一致, 该中断标志将被设置为 1。 0 = 未发生从机模式位计数错误事件 1 = 发生从机模式位计数错误事件 <b>注:</b> 如果从机片选激活, 但是没有总线时钟输入, 当从机片选进入无效状态时, SLVBEIF 也将被设置为 1. 该位写 1 清 0。
[5]	<b>SLVTOIF</b>	<b>从机超时中断标志 (仅 SPI0 支持)</b> 当从机片选信号激活和 SLVTOCNT 的值不为 0 时, 在检测到总线时钟时, SPI 控制器逻辑的从机超时计数器开始计数。在事务完成之前, 如果超时计数器的值大于或等于

		SLVTOCNT (SPI_SSCTL[31:16]) 里的值。将发生从机超时中断事件。 0 = 从机超时未发生 1 = 发生从机超时 <b>注:</b> 该位写 1 清 0
[4]	<b>SSLIN</b>	从机片选线总线状态 (只读) 0 = 从机片选线状态为 0 1 = 从机片选线状态为 1 <b>注:</b> 该位只在从机模式有效。如果 SSACTPOL (SPIx_SSCTL[2]) 设置为 0，而且 SSLIN 的值为 1，SPI 从机片选处于无效状态。
[3]	<b>SSINAIF</b>	从机片选无效中断标志 0 = 从机片选无效中断被清除或没有发生 1 = 发生了从机片选无效中断 <b>注:</b> 该位只在从机模式有效，该位写 1 清 0。
[2]	<b>SSACTIF</b>	从机片选激活中断标志 0 = 从机片选激活中断被清除或未发生 1 = 发生了从机片选激活中断 <b>注:</b> 该位只在从机模式有效，该位写 1 清 0。
[1]	<b>UNITIF</b>	单元传输中断标志 0 = 该位清除后，没有事务完成 1 = SPI 控制器已完成一个单元传输 <b>注:</b> 该位写 1 清 0。
[0]	<b>BUSY</b>	忙状态 (只读) 0 = SPI 控制器处于空闲状态 1 = SPI 控制器处于忙状态。 如下是忙条件： a. SPIx_CTL[0] = 1 且 TXEMPTY = 0。 b. 对于 SPI 主机模式，SPIx_CTL[0] = 1 且 TXEMPTY = 1 但是当前传输尚未完成。 c. 对于 SPI 主机模式，SPIx_CTL[0] = 1 和 RXONLY = 1。 d. 对于 SPI 从机模式，SPIx_CTL[0] = 1，从机选择有效时仍有串口时钟输入。 e. 对于 SPI 从机模式，SPIx_CTL[0] = 1，从机选择无效时发送缓存或发送移位寄存器仍非空。

SPIx\_TX

## SPI 数据发送寄存器

寄存器	偏移量	R/W	描述	复位值
SPIx_TX	SPIx_BA+0x20	W	SPI 数据发送寄存器	0x0000_0000

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

位	描述
[31:0]	<p><b>数据发送寄存器</b></p> <p>数据发送寄存器会把要发送的数据传入 4 级的发送 FIFO 缓存。该寄存器的有效位数在 SPI 模式下取决于 DWIDTH (SPIx_CTL[12:8])，在 I2S 模式下取决于 WDWIDHT (SPIx_I2SCTL[5:4])。</p> <p>SPI 模式下，如果 DWIDTH 为 0x08，TX[7:0] 位会被发送。如果 DWIDTH 为 0x00，SPI 控制器将发送 32 位。</p> <p>I2S 模式下，如果 WDWIDHT (SPIx_I2SCTL[5:4]) 为 0x2，数据宽度为 24 位，对应到该寄存器是 TX[23:0]。如果 WDWIDHT 为 0x0, 0x1, 或 0x3，该寄存器所有位均有效，具体数据排列由 I2S 模式 FIFO 操作决定。</p> <p><b>注：</b>主机模式下，SPI 控制器写该寄存器后，1 个 APB 时钟加 6 个外设时钟后才会开始 SPI 总线时钟的传输。</p>

**SPIx\_RX SPI 数据接收寄存器**

寄存器	偏移量	R/W	描述	复位值
SPIx_RX	SPIx_BA+0x30	R	SPI 数据接收寄存器	0x0000_0000

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

位	描述	
[31:0]	RX	<p><b>数据接收寄存器</b></p> <p>SPI 控制器有 4 级 FIFO 缓存。从 SPI 数据输入管脚接收到的数据会被保存到该寄存器。如果 RXEMPTY (SPIx_STATUS[8] 或 SPIx_I2SSTS[8]) 不为 1，软件可以通过读该寄存器获取 FIFO 缓存的数据。</p> <p><small>该寄存器只读</small></p>

**SPIx\_I2SCTL I2S 控制寄存器**

寄存器	偏移量	R/W	描述	复位值
SPIx_I2SCTL	SPIx_BA+0x60	R/W	I2S 控制寄存器	0x0000_0000

注: 该寄存器不支持 SPI 模式

31	30	29	28	27	26	25	24
Reserved		FORMAT			Reserved		LZCIEN
23	22	21	20	19	18	17	16
RXLCH	Reserved					LZCEN	RZCEN
15	14	13	12	11	10	9	8
MCLKEN	Reserved						SLAVE
7	6	5	4	3	2	1	0
ORDER	MONO	WDWIDTH			MUTE	RXEN	TXEN
		I2SEN					

位	描述	
[31:30]	Reserved	保留
[29:28]	FORMAT	<p>数据格式选择            00 = I<sup>2</sup>S 数据格式            01 = MSB 对齐数据格式            10 = PCM 模式 A.            11 = PCM 模式 B.</p>
[27:26]	Reserved	保留
[25]	LZCIEN	<p>左声道过零检测中断使能位            如果该位置 1, 左声道检测到过零事件时发生中断            0 = 关闭中断            1 = 使能中断</p>
[24]	RZCIEN	<p>右声道过零检测中断使能位            如果该位置 1, 右声道检测到过零事件时发生中断            0 = 关闭中断            1 = 使能中断</p>
[23]	RXLCH	<p>左声道接收使能位            I<sup>2</sup>S 控制器处于单声道模式 (MONO = 1) 时, 如果 RXLCH 为 0, I<sup>2</sup>S 控制器接收右声道的数据; 如果 RXLCH 为 1, I<sup>2</sup>S 控制器接收左声道的数据。            0 = 单声道模式下, 接收右声道的数据            1 = 单声道模式下, 接收左声道的数据</p>
[22:18]	Reserved	保留

[17]	<b>LZCEN</b>	<b>左声道过零检测使能位</b> 如果该位为 1，当左声道数据信号位改变或下一笔数据位全为 0， SPIx_I2SSTS 寄存器的 LZCIF 会被置 1。该功能仅在发送操作下有效。 0 = 左声道过零检测关闭 1 = 左声道过零检测使能
[16]	<b>RZCEN</b>	<b>右声道过零检测使能位</b> 如果该位为 1，当右声道数据信号位改变或下一笔数据位全为 0， SPIx_I2SSTS 寄存器的 RZCIF 会被置 1。该功能仅在发送操作下有效。 0 = 右声道过零检测关闭 1 = 右声道过零检测使能
[15]	<b>MCLKEN</b>	<b>主时钟使能位</b> 如果 MCLKEN 置 1，I2S 控制块会在 SPIx_I2SMCLK 管脚产生主时钟以供外部音频设备使用 0 = 主时钟关闭 1 = 主时钟使能
[14:9]	<b>Reserved</b>	保留
[8]	<b>SLAVE</b>	<b>从机模式</b> I2S 可以工作在主机模式或从机模式。主机模式时，I2Sx_BCLK 和 I2Sx_LRCLK 管脚都为输出模式，而且可以通过该芯片为音频解码芯片发送位时钟。从机模式时，I2Sx_BCLK 和 I2Sx_LRCLK 管脚都为输入模式，而且 I2Sx_BCLK 和 I2Sx_LRCLK 的信号都来自外部的音频解码芯片。 0 = 主机模式 1 = 从机模式
[7]	<b>ORDER</b>	<b>FIFO 内双通道数据顺序</b> 0 = 左声道位于高字节 1 = 左声道位于低字节
[6]	<b>MONO</b>	<b>单通道数据</b> 0 = 双通道的数据格式 1 = 单通道的数据格式
[5:4]	<b>WDWIDTH</b>	<b>字宽度</b> 00 = 数据宽度为 8 位 01 = 数据宽度为 16 位 10 = 数据宽度为 24 位 11 = 数据宽度为 32 位
[3]	<b>MUTE</b>	<b>发送静音使能位</b> 0 = 发送数据来自缓存 1 = 发送数据为 0
[2]	<b>RXEN</b>	<b>接收使能位</b> 0 = 数据接收关闭 1 = 数据接收使能
[1]	<b>TXEN</b>	<b>发送使能位</b> 0 = 数据发送关闭 1 = 数据发送使能

[0]	I2SEN	<p><b>I<sup>2</sup>S 控制器使能位</b></p> <p>0 = I<sup>2</sup>S 模式禁止</p> <p>1 = I<sup>2</sup>S 模式使能</p> <p>注:</p> <ol style="list-style-type: none"><li>1. 主机模式下使能该位后, I2Sx_BCLK 将开始输出</li><li>2. 用户在更改寄存器 SPIx_I2SCTL, SPIx_I2SCLK, 和 SPIx_FIFOCTL 前, 需要清 0 I2SEN (SPIx_I2SCTL[0]) 并确认 I2SENSTS (SPIx_I2SSTS[15]) 为 0。</li></ol>
-----	-------	---

**SPIx\_I2SCLK I2S 时钟分频控制寄存器**

寄存器	偏移量	R/W	描述	复位值
SPIx_I2SCLK	SPIx_BA+0x64	R/W	I2S 时钟分频控制寄存器	0x0000_0000

Note: Not supported in SPI mode.

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
BCLKDIV							
7	6	5	4	3	2	1	0
Reserved	MCLKDIV						

位	描述	
[31:18]	<b>Reserved</b>	保留
[17:8]	<b>BCLKDIV</b>	<p><b>位时钟除频</b> 主机模式下位时钟由 I2S 控制器产生。位时钟和 <math>f_{BCLK}</math> 有以下关系:</p> $f_{BCLK} = \frac{f_{i2s\_clock\_src}}{2 \times (\text{BCLKDIV} + 1)}$ <p>其中</p> <p><math>f_{i2s\_clock\_src}</math> 为 I2S 外设时钟源, 由寄存器 CLK_CLKSEL2 定义。</p> <p>从机模式下, 该域用来定义外设时钟的频率, 公式为:</p> $f_{i2s\_clock\_src} \div \left( \frac{\text{BCLKDIV}}{2} + 1 \right).$ <p>I2S 从机模式外设时钟频率必须等于或大于 6 倍输入位时钟的频率。</p>
[7]	<b>Reserved</b>	保留
[6:0]	<b>MCLKDIV</b>	<p><b>主时钟除频</b> MCLKEN 为 1 时, I2S 控制器会产生时钟到外部音频设备。主时钟, <math>f_{MCLK}</math> 由下列公式决定:</p> $\text{MCLKDIV} >= 1, \quad f_{MCLK} = \frac{f_{i2s\_clock\_src}}{2 \times \text{MCLKDIV}}$ $\text{MCLKDIV} = 0, \quad f_{MCLK} = f_{i2s\_clock\_src}$ <p>其中,</p> <p><math>f_{i2s\_clock\_src}</math> 为 I2S 外设时钟频率, 由 CLK_CLKSEL2 决定。通常来说, 主时钟的频率为采样时钟频率的 256 倍。</p>

**注意:** 谨慎配置 **BCLKDIV**，频率不能超过系统时钟

SPIx\_I2SSTS I2S 状态寄存器

寄存器	偏移量	R/W	描述	复位值
SPIx_I2SSTS	SPIx_BA+0x68	R/W	I2S 状态寄存器	0x0005_0100

注: 不支持 SPI 模式

31	30	29	28	27	26	25	24
Reserved	TXCNT				Reserved	RXCNT	
23	22	21	20	19	18	17	16
TXRXRST	Reserved	LZCIF	RZCIF	TXUFIF	TXTHIF	TXFULL	TXEMPTY
15	14	13	12	11	10	9	8
I2SENSTS	Reserved		RXTOIF	RXOVIF	RXTHIF	RXFULL	RXEMPTY
7	6	5	4	3	2	1	0
Reserved			RIGHT	Reserved			

位	描述	
[31]	Reserved	保留
[30:28]	TXCNT	发送 FIFO 数据计数 (只读) 该位表示发送 FIFO 缓存的有效数据个数
[27]	Reserved	保留
[26:24]	RXCNT	接收 FIFO 数据计数 (只读) 该位表示接收 FIFO 缓存的有效数据个数
[23]	TXRXRST	TX 或 RX 复位状态 (只读) 0 = TXRST 或 RXRST 复位完成 1 = TXRST 或 RXRST 复位中 注: TXRST 或 RXRST 复位需要花费 3 个系统时钟周期 + 2 个外设时钟周期。用户可以检查该位来判断复位是否完成。
[22]	Reserved	保留
[21]	LZCIF	左通道过零中断标志位 0 = 左通道未发生过零事件 1 = 左通道已发生过零事件
[20]	RZCIF	右通道过零中断标志位 0 = 右通道未发生过零事件 1 = 右通道已发生过零事件
[19]	TXUFIF	发送 FIFO 下溢中断标志 如果在发送 FIFO 缓存为空, FIFO 缓存里也没有数据写入, 此时如果有总线时钟输入, 该位置 1 注: 该位写 1 清 0
[18]	TXTHIF	发送 FIFO 阈值中断标志 (只读)

		0 = 传送 FIFO 缓存中的有效数据个数大于 TXTH 设置的值 1 = 传送 FIFO 缓存中的有效数据个数小于或等于 TXTH 设置的值 <b>注:</b> 如果 TXTHIEN = 1 且 TXTHIF = 1, SPI/I <sup>2</sup> S 控制器会产生一个 SPI 中断请求
[17]	TXFULL	<b>发送 FIFO 缓存满标志位 (只读)</b> 0 = 发送 FIFO 缓存未满 1 = 发送 FIFO 缓存已满
[16]	TXEMPTY	<b>发送 FIFO 缓存空标志位 (只读)</b> 0 = 发送 FIFO 缓存非空 1 = 发送 FIFO 缓存已空
[15]	I2SENSTS	<b>I<sup>2</sup>S 使能状态 (只读)</b> 0 = SPI/I <sup>2</sup> S 控制逻辑关闭 1 = SPI/I <sup>2</sup> S 控制逻辑使能 <b>注:</b> SPI/I <sup>2</sup> S 外设时钟与系统时钟不同步。为了确保 SPI/I <sup>2</sup> S 控制器已经被禁止, 该位指示了 SPI/I <sup>2</sup> S 控制器的真实状态。
[14:13]	Reserved	保留
[12]	RXTOIF	<b>接收超时中断标志</b> 0 = 无接收 FIFO 超时中断事件 1 = 接收 FIFO 缓存非空且主机模式下超过 64 个 SPI 时钟周期或从机模式下超过 576 个 SPI 引擎时钟周期, 接收 FIFO 缓存上没有读操作。当接收 FIFO 缓存被软件读取, 超时状态会自动清 0。 <b>注:</b> 该位写 1 清 0
[11]	RXOVIF	<b>接收 FIFO 下溢中断标志</b> 如果接收 FIFO 缓存已满, 之后接收的数据会被丢弃且该位置 1。 <b>注:</b> 该位写 1 清 0
[10]	RXTHIF	<b>接收 FIFO 阈值中断标志 (只读)</b> 0 = 接收 FIFO 缓存的有效数据个数少于或等于 RXTH 设置的值 1 = 接收 FIFO 缓存的有效数据个数大于 RXTH 设置的值 <b>注:</b> 如果 RXTHIEN = 1 且 RXTHIF = 1, SPI/I <sup>2</sup> S 控制器会产生一个 SPI 中断请求
[9]	RXFULL	<b>接收 FIFO 缓存满标志 (只读)</b> 0 = 接收 FIFO 缓存未满 1 = 接收 FIFO 缓存已满
[8]	RXEMPTY	<b>接收 FIFO 缓存空标志 (只读)</b> 0 = 接收 FIFO 缓存非空 1 = 接收 FIFO 缓存已空
[7:5]	Reserved	保留
[4]	RIGHT	<b>右声道 (只读)</b> 该位指示当前传输的数据是哪一个声道 0 = 左声道 1 = 右声道
[3:0]	Reserved	保留

## 6.20 QSPI 接口

### 6.20.1 概述

QSPI接口是全双工同步串行数据通讯接口，可做为主机或从机，用 4 线双向通讯。M480 包含 1 组 QSPI 控制器。

QSPI支持全双工的 2 位传输模式，也支持双 I/O 和四 I/O 传输模式。QSPI支持 PDMA传输功能。

### 6.20.2 特征

- QSPI 模式

- 支持主机模式和从机模式
- 主/从模式时钟最高 100 MHz，(VDD = 2.7~3.6V)
- QSPI 支持 2 位传输模式
- QSPI 支持双 I/O 和四 I/O 传输模式
- 传输位长可为 8 ~ 32
- 收发独立的 8 级 FIFO 缓存
- 高低位在前可配置
- 支持字节重排功能
- 支持字节或字暂停模式
- 支持 PDMA 传输
- QSPI 支持三线模式，无从机片选的双向接口
- 支持一数据通道半双工传输
- 支持只接收模式

### 6.20.3 框图

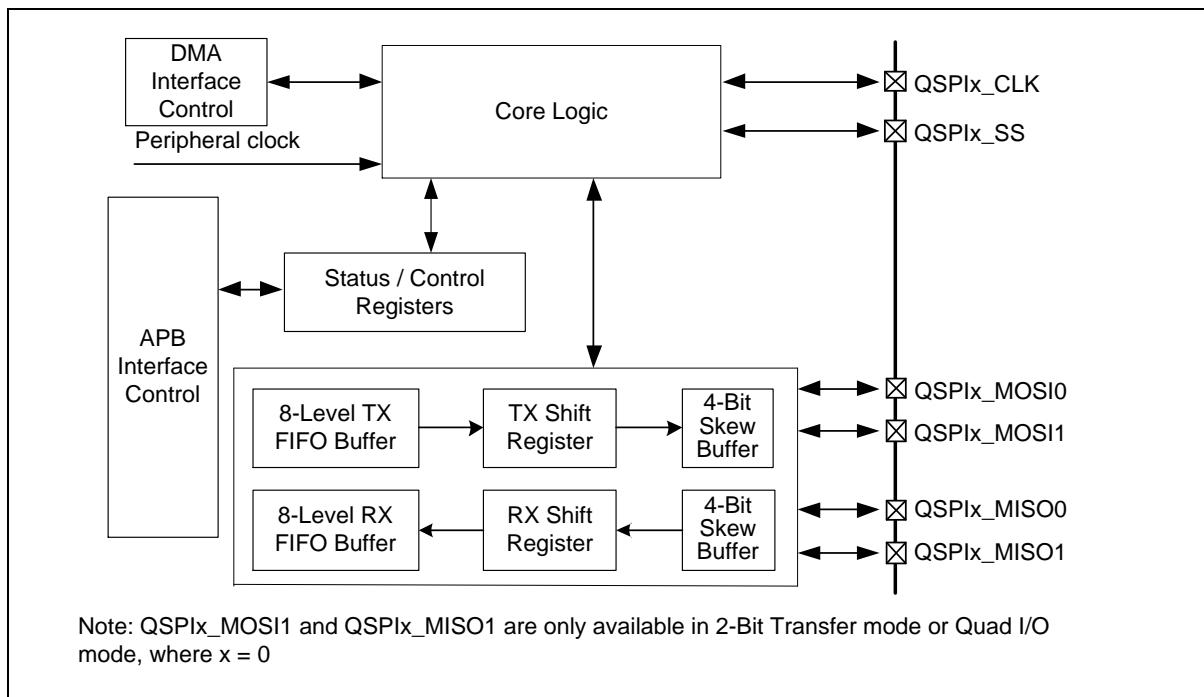


图 6.20-1 QSPI 框架图

**TX FIFO 缓存:**

发送 FIFO 8 级深度，32 位宽。数据写入寄存器 QSPIx\_TX 就写入发送 QSPI 模式，如果数据长度为 8~16 位，则发送 FIFO 可以设为 8 级。

**RX FIFO 缓存:**

接收 FIFO 8 级深度，32 位宽。接收控制会把数据存到该缓存。软件可通过读 QSPIx\_RX 得到该 FIFO 数据。QSPI 模式时，如果数据长度为 8~16 位，发送 FIFO 可以设为 8 级。

**TX 移位寄存器:**

发送移位寄存器是一个 32 位的寄存器。发送数据从 TX FIFO 缓存加载到这里，并一位一位的移到斜移缓存。

**RX 移位寄存器:**

接收移位寄存器是一个 32 位的寄存器。接收数据从斜移缓存一位一位的移到这里，当一个事务完成时再载入到 RX FIFO。

**斜移缓存:**

斜移缓存是一个 4 级 1 位的缓存。总共有 2 个斜移缓存分别在发送和接收端。在接收端，斜移缓存用于把数据从 SPI 总线移位到 Rx 移位寄存器。在发送端，斜移缓存用于把数据从 Rx 移位寄存器移位到 SPI 总线。

**6.20.4 基本配置****6.20.4.1 QSPI0 基本配置**

- 时钟配置

寄存器 QSPI0SEL (CLK\_CLKSEL2[3:2]) 配置 QSPI0 的时钟

寄存器 QSPI0CKEN (CLK\_APBCLK0[12]) 使能 QSPI0 的时钟

- 复位配置

寄存器 QSPI0RST (SYS\_IPRST1[12]) 复位 QSPI0 控制器

- 管脚配置

组	管脚名	GPIO	MFP
QSPI0	QSPI0_CLK	PA.2, PH.8	MFP3
		PC.2	MFP4
		PF.2	MFP5
		PC.14	MFP6
	QSPI0_MISO0	PA.1, PE.1	MFP3
		PC.1	MFP4
	QSPI0_MISO1	PA.5, PH.10	MFP3
		PC.5	MFP4
	QSPI0_MOSI0	PA.0, PE.0	MFP3
		PC.0	MFP4
	QSPI0_MOSI1	PA.4, PH.11	MFP3
		PC.4	MFP4
	QSPI0_SS	PA.3, PH.9	MFP3
		PC.3	MFP4

## 6.20.5 功能描述

### 6.20.5.1 术语

#### QSPI 外设时钟和 QSPI 总线时钟

QSPI 时钟决定于时钟分频器 (SPIx\_CLKDIV) 和时钟源的设置，时钟源可以设置为 HXT, HIRC, PLL 时钟或 PCLK.。寄存器 CLK\_CLKSEL2 的 QSPIxSEL 位选择 QSPI 的时钟源。寄存器 DIVIDER (QSPI\_CLKDIV[8:0]) 的值决定时钟的分频值。

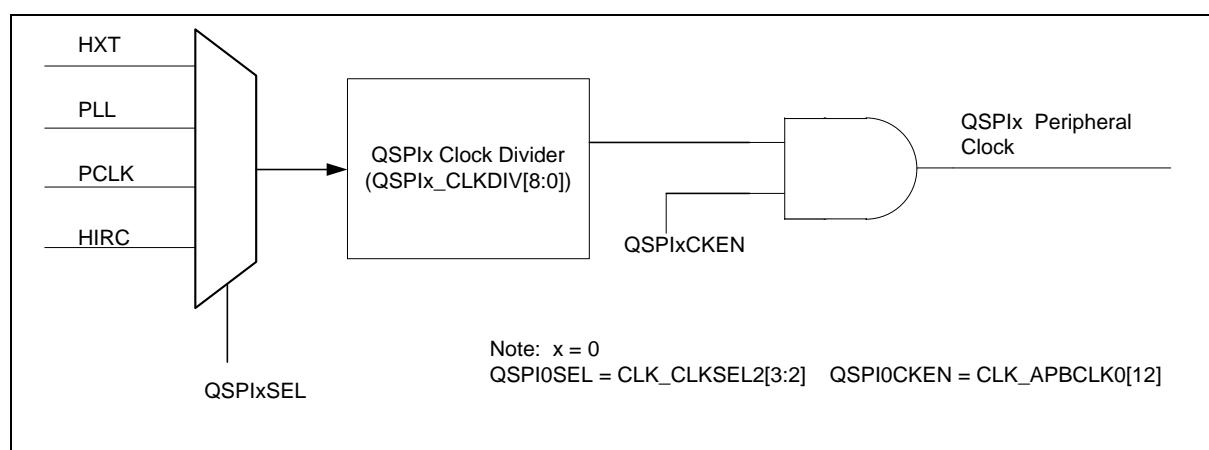


图 6.20-2 QSPI 外设时钟

QSPI 主机模式下，QSPI 总线的时钟频率等于外设的时钟速率。从机模式下，QSPI 总线时钟由主机设备提供。无论主机模式还是从机模式，QSPI 时钟不能快于系统时钟。如果时钟源不是系统时钟，QSPI 时钟不管是主机模式还是从机模式都必须慢于系统时钟频率。

### 主从模式

可以通过寄存器 SLAVE (QSPIx\_CTL[18]) 设成主机模式或从机模式。QSPI 传送过程中，通过 HALFDPX (QSPIx\_CTL[14]) 可以切换全双工模式或半双工模式。示意图如下。

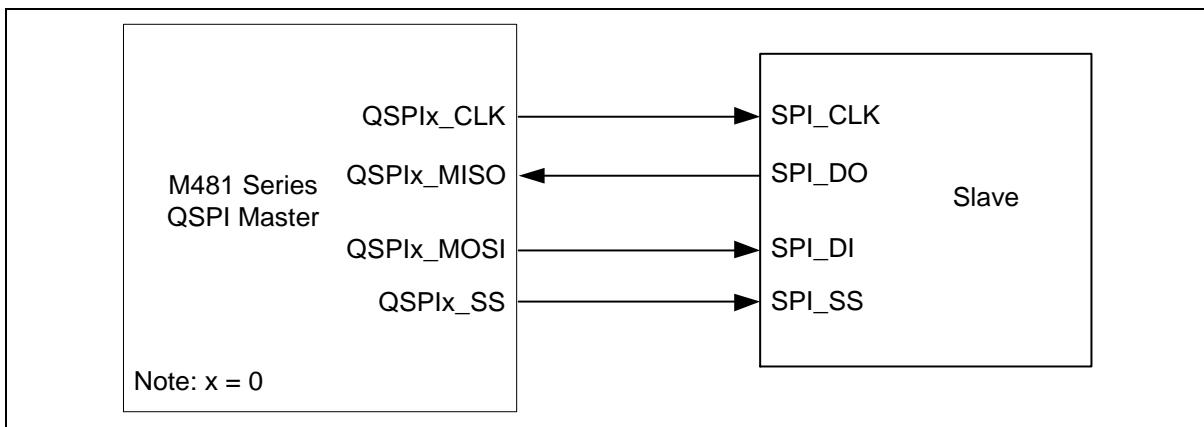


图 6.20-3 QSPI 全双工主机模式

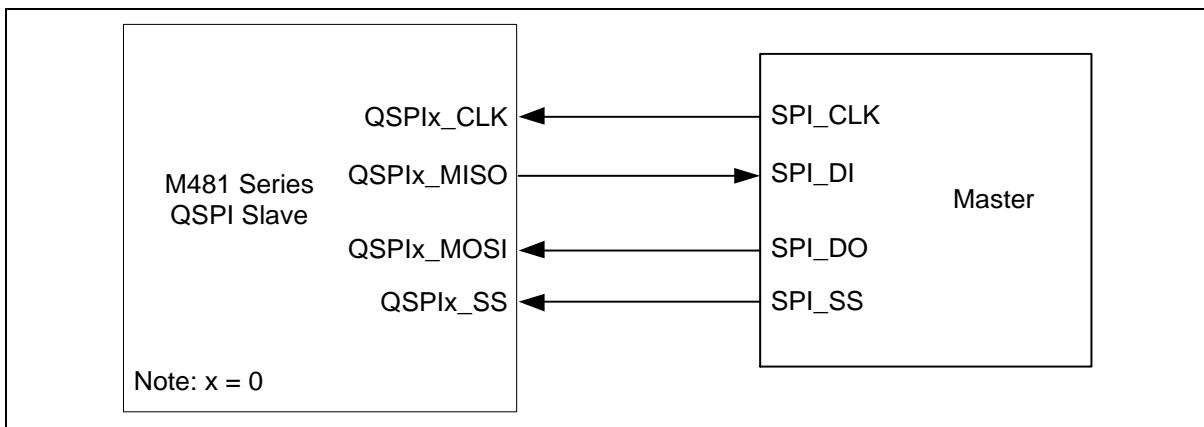


图 6.20-4 QSPI 全双工从机模式

### 从机选择

主机模式下，QSPI 通过从机片选 QSPIx\_SS (输出) 使能从机设备。从机模式，其它主机通过 QSPIx\_SS (输入) 使能 QSPI 控制器。片选有效边沿到第一个 QSPI 时钟输入应多于 3 个 SPI 外设时钟周期。

从机信号可通过寄存器 SSACTPOL (QSPIx\_SSCTL[2]) 设定为低有效或高有效。从机模式下，从机片选信号的无效时间必须大于等于 3 个外设时钟周期。

### 时钟极性

CLKPOL (QSPIx\_CTL[3]) 定义 QSPI 时钟空闲时的电平。如果 CLKPOL = 1，QSPI 时钟空闲时输出高电平，如果 CLKPOL = 0，则 QSPI 时钟空闲时输出低电平。

**TXNEG (QSPIx\_CTL[2])** 定义数据是在 QSPI 时钟的下沿发送还是上沿发送。**RXNEG (QSPIx\_CTL[1])** 定义数据在 QSPI 时钟的下沿接收还是上沿接收。

**注意:** TXNEG 和 RXNEG 的设置是互斥的，收发不能都配置为上沿或都是下沿。

### 接发位长

位长由 **DWIDTH (QSPIx\_CTL[12:8])** 来配置，最多可为 32 位。

当 QSPI 完成一次 **DWIDTH (QSPIx\_CTL[12:8])** 定义的位长的收/发时，传输中断标志将被置 1.

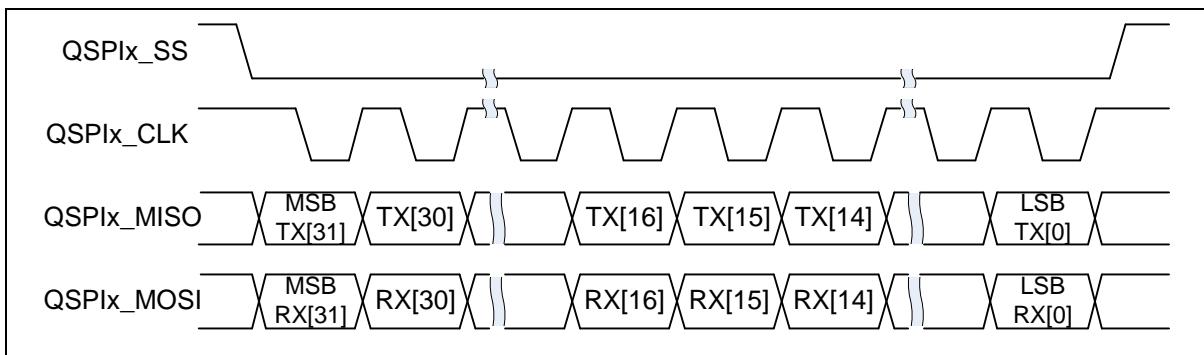


图 6.20-5 32 位的收发

### 高低位在前

**LSB (QSPIx\_CTL[13])** 定义传输顺序。**LSB (QSPIx\_CTL[13])=1** 时，第 0 位会先传输。若 **(QSPIx\_CTL[13]) = 0**，则先传高位。

### 休眠间隔

主机模式下，**SUSPITV (QSPIx\_CTL[7:4])** 可配置在两次连续传输之间，插入 0.5~15.5 个 QSPI 时钟周期的空闲：空闲时间的计算是从前一次传输的最后一个时钟沿，到下一次传输的第一个时钟沿。**SUSPITV** 的默认值是 0x3 (3.5 个 QSPI 时钟周期)。.

#### 6.20.5.2 自动从机选择

主机模式下，如果 **AUTOSS (QSPIx\_SSCTL[3])** 置位，从机选择信号根据 **SS (QSPIx\_SSCTL[0])** 自动输出到 **QSPIx\_SS** 管脚上。当数据写入 FIFO 数据传输启动，从机选择信号自动设置为有效状态。当 QSPI 总线空闲时，从机选择信号将自动变为无效状态。QSPI 总线不空闲时——比如 TX FIFO，TX 移位寄存器或 TX 斜移缓存不空——在 **SUSPITV (QSPIx\_CTL[7:4])** 的值大于或等于 3 的条件下，从机选择信号在两次传输之间将被置为无效状态。

主机模式下，如果 **SUSPITV** 的值小于 3 且 **AUTOSS** 置 1 时，两次连续传输之间，从机选择信号将保持有效。

如果 **AUTOSS** 被清零，从机选择输出信号的有效电平由 **SSACTPOL (QSPIx\_SSCTL[2])** 来定义。

从机片选信号有效边沿，到第一个 QSPI 时钟边沿的间隔为 1 个 QSPI 总线时钟周期。QSPI 总线的最后一个时钟，到从机选择信号无效边沿的间隔为 1.5 个 QSPI 总线时钟周期。

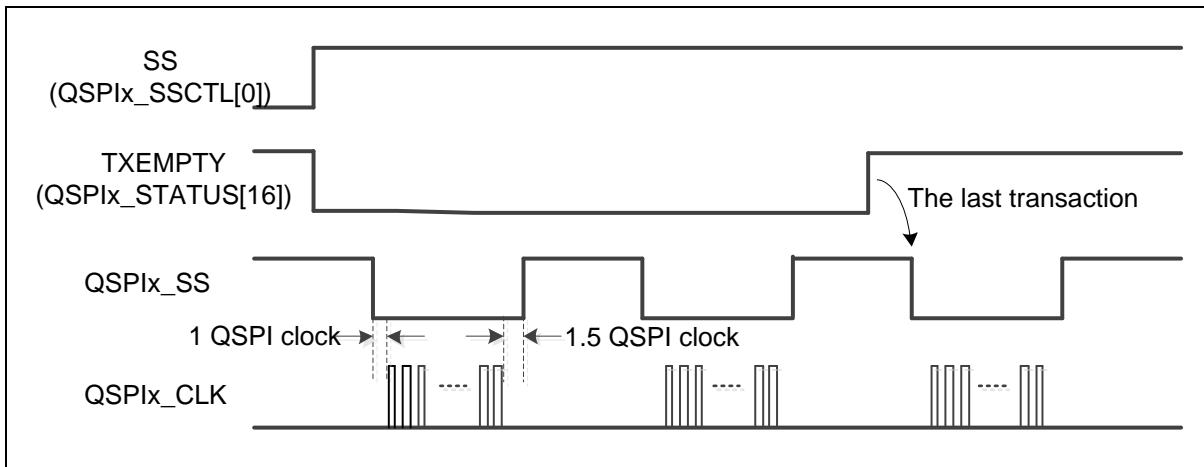


图 6.20-6 自动从机选择 (SSACTPOL = 0, SUSPITV &gt; 0x2)

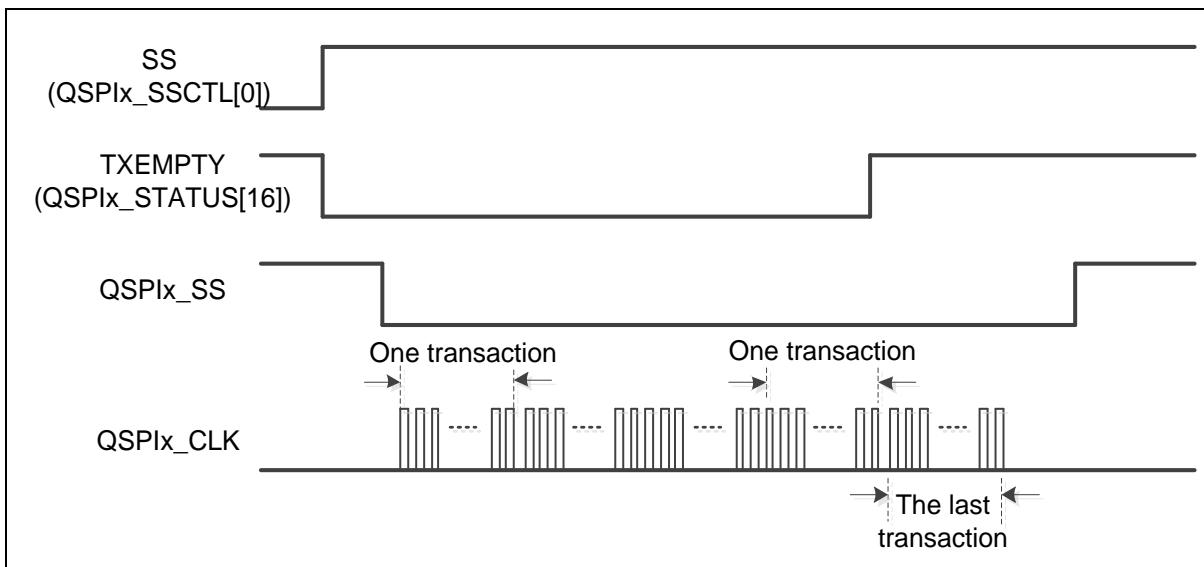


图 6.20-7 自动从机选择 (SSACTPOL = 0, SUSPITV &lt; 0x3)

### 6.20.5.3 字节重排和挂起功能

当传输设置为 MSB 优先 (LSB = 0) 且使能了 REORDER (QSPIx\_CTL[19])，在 32 位传输模式 (DWIDTH = 0) 下，存储在 TX 缓存与 RX 缓存的数据将按 [BYTE0, BYTE1, BYTE2, BYTE3] 的顺序重新排列。数据发送/接收的顺序为 BYTE0, BYTE1, BYTE2, 和 BYTE3。如果 DWIDTH 设为 24 位传输模式，存储在 TX 缓存与 RX 缓存的数据将按 [未知字节, BYTE0, BYTE1, BYTE2] 的顺序重新排列。QSPI 控制器将按照 BYTE0, BYTE1, BYTE2 的顺序发送/接收数据，每个字节 MSB 优先发送/接收。16 位传输模式的规则与上面相同。字节重排序功能只适用于 DWIDTH 为 16, 24, 和 32 位时。

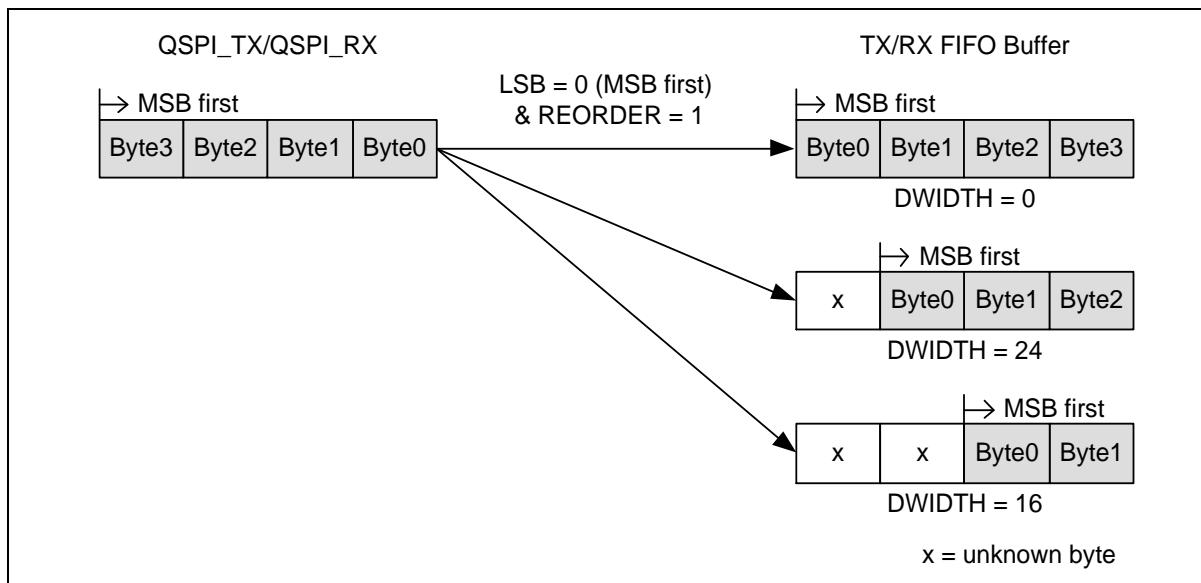


图 6.20-8 字节重排功能

主机模式下，如果 REORDER (QSPIx\_CTL[19]) 为 1，两次连续传输字节之间将插入 0.5 ~ 15.5 个 QSPI 时钟周期的挂起间隔。挂起间隔时间由 SUSPITV (QSPIx\_CTL[7:4]) 设置。

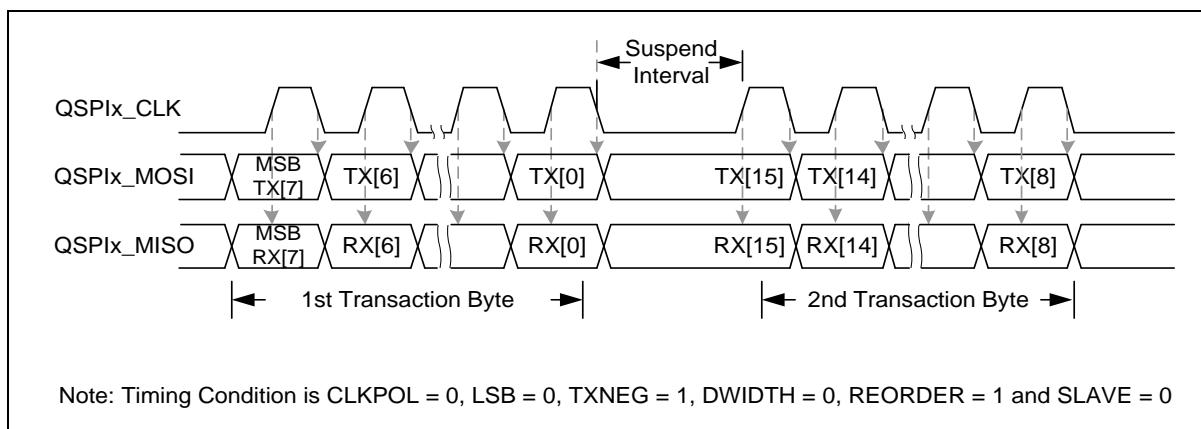


图 6.20-9 字节挂起波形图

#### 6.20.5.4 半双工通讯

设置HALFDPX (QSPIx\_CTL[14]) 位可以使能 QSPI 工作在半双工模式。半双工模式下，仅有一条数据线进行发送或接收，方向由DATADIR (QSPIx\_CTL[20]) 配置。半双工模式下，没有用到的 QSPIx\_MISO 管脚可以配置成 GPIO 供其他功能使用。使能或关闭 HALFDPX (QSPIx\_CTL[14]) 控制位将会同时产生 TXFBCLR (QSPIx\_FIFOCTL[9]) 和 RXFBCLR (QSPIx\_FIFOCTL[8])。

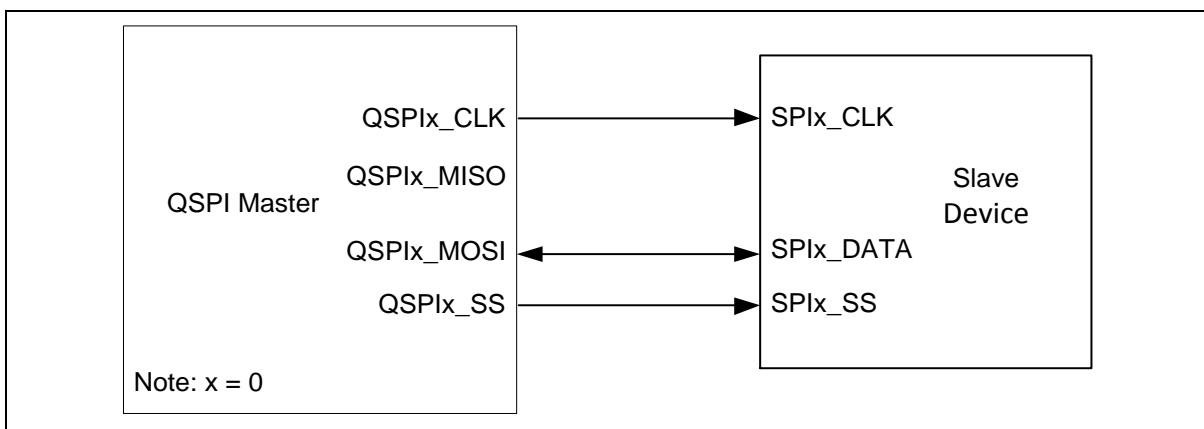


图 6.20-10 QSPI 半双工主机模式

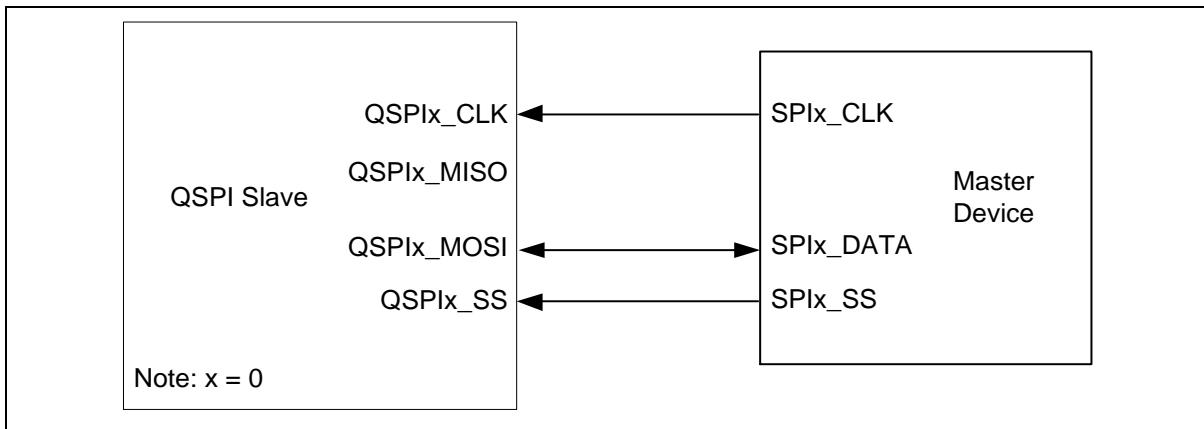


图 6.20-11 QSPI 半双工从机模式

### 6.20.5.5 只接收模式

对于 QSPI 主机，可配置 RXONLY (QSPIx\_CTL[15]) 让其工作在只接收模式，QSPI 主机一直不断地产生 QSPI 总线时钟。该模式下，如果使能了 AUTOSS (QSPIx\_SSCTL[3])，QSPI 主机将一直保持从机片选信号有效。

在只接收模式下，没有用到的 QSPIx\_MOSI 管脚可以配置成 GPIO 供其他功能使用。由于 QSPI 总线时钟一直存在，BUSY (QSPIx\_STATUS[0]) 将会被置位。该模式下会同时产生 TXFBCLR (QSPIx\_FIFOCTL[9]) 和 RXFBCLR (QSPIx\_FIFOCTL[8])。使能该模式后，QSPI 总线时钟会在 6 个外设时钟周期内发送出去。该模式下，写入发送 FIFO 的数据也会被载入发送移位寄存器发送出去。

### 6.20.5.6 从机三线模式

SLV3WIRE (QSPIx\_SSCTL[4]) 置 1 就使能了从机 3 线模式，仅需三个管脚：QSPIx\_CLK，QSPIx\_MISO，和 QSPIx\_MOSI，QSPI 控制器在无从机选择信号时正常工作，QSPIx\_SS 脚可配置成 GPIO。SLV3WIRE (QSPIx\_SSCTL[4]) 仅在从机模式有效。在与 QSPI 主机通讯时，当 SLV3WIRE (QSPIx\_SSCTL[4]) 设为 1 时，在 SPIEN (QSPIx\_CTL[0]) 置 1 后就开始准备发送和接收数据。

### 6.20.5.7 PDMA 传输功能

QSPI 控制器支持 PDMA 传输功能。

当 TXPDMAEN (QSPIx\_PDMACTL[0]) 置 1 时，控制器会请求 PDMA 控制发送过程。

当 RXPDMAEN (QSPIx\_PDMACTL[1]) 置 1 时，将启动 PDMA 接收过程。当接收 FIFO 有数据时，控

制器会请求 PDMA 接收数据。

注: QSPI 只支持单一 PDMA 请求(读/写), 不支持 PDMA 突发请求.

#### 6.20.5.8 2 位传输模式

TWOBIT (QSPIx\_CTL[16]) 为1时, 使能2位传输模式, QSPI 进行全双工数据传输, 可同时收发两位数据。

主机模式下, 寄存器 QSPIx\_TX 的偶数位 (TX Data (n)) 通过 QSPIx\_MOSI0 管脚发送, 奇数位 (TX Data (n+1)) 通过 QSPIx\_MOSI1 管脚发送。同时, 从 QSPIx\_MOSIO 管脚接收到的偶数位数据将优先于从 QSPIx\_MOSI1 管脚接收到的奇数位数据写入接收 FIFO。

从机模式下, 寄存器 QSPIx\_TX 中的偶数位和奇数位将分别从 QSPIx\_MISO0 管脚和 QSPIx\_MISO1 管脚发送。同时, 从 QSPIx\_MISO0 管脚接收到的偶数位和从 QSPIx\_MISO1 管脚接收到的奇数位将分别写入 QSPI\_RX 寄存器。FIFO 缓存的数据顺序与主机模式是一样的。

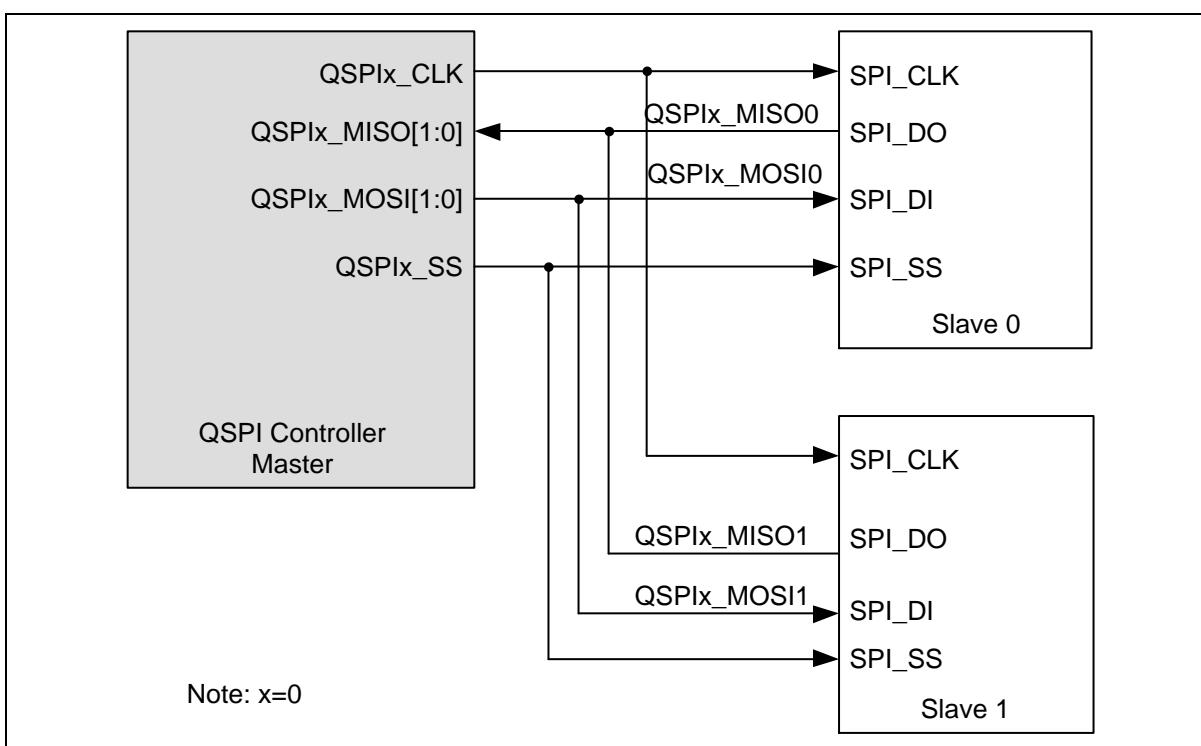


图 6.20-12 2 位传输模式

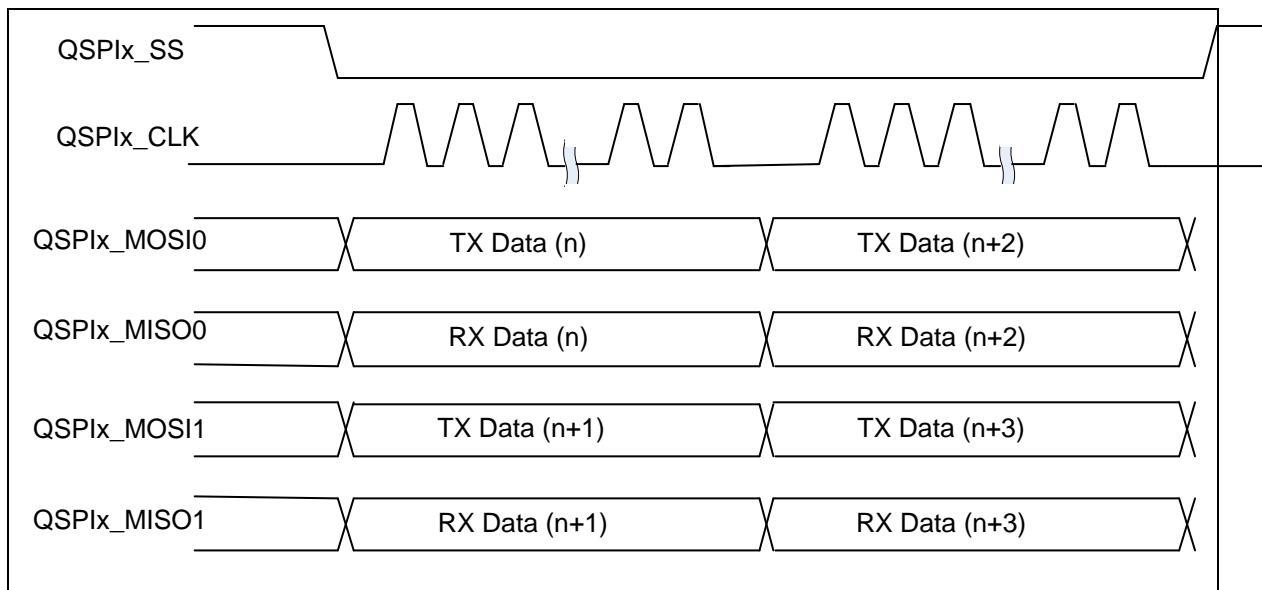


图 6.20-13 2 位传输模式的时序图 (主机模式)

#### 6.20.5.9 双 I/O 模式

DUALIOEN ((QSPIx\_CTL[21]) 置 1, QSPI 配置为双 I/O 传输。DATDIR (QSPIx\_CTL[20]) 用来定义传输数据的方向。DATDIR = 1 时, 为双路发送, 反之为双路接收。该功能支持 8, 16, 24, 和 32 位长度。从机三线模式或字节重排功能时, 不支持双 I/O 模式。

针对双 I/O 模式, 如果 DUALIOEN (QSPIx\_CTL[21]) 和 DATDIR (QSPIx\_CTL[20]) 都设置成 1, 则 QSPIx\_MISO0 输出偶数位, QSPIx\_MISO0 输出奇数位。如果 DUALIOEN (QSPIx\_CTL[21]) 置 1, DATDIR (QSPIx\_CTL[20]) 置 0, QSPIx\_MISO0 和 QSPIx\_MOSI0 都为输入端口。

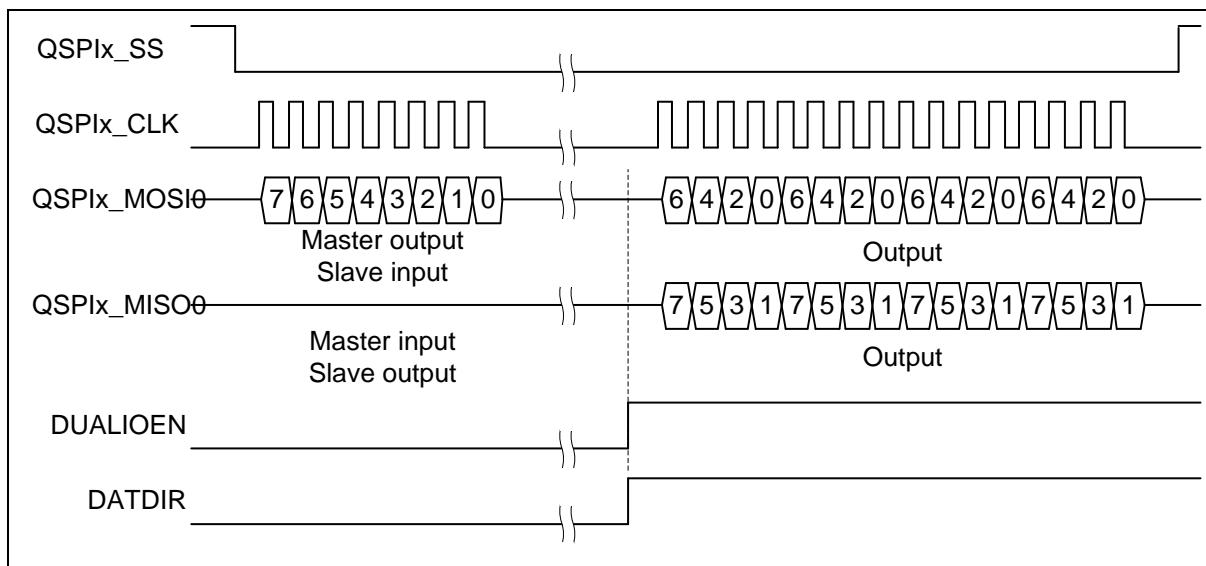


图 6.20-14 双输出模式的时序

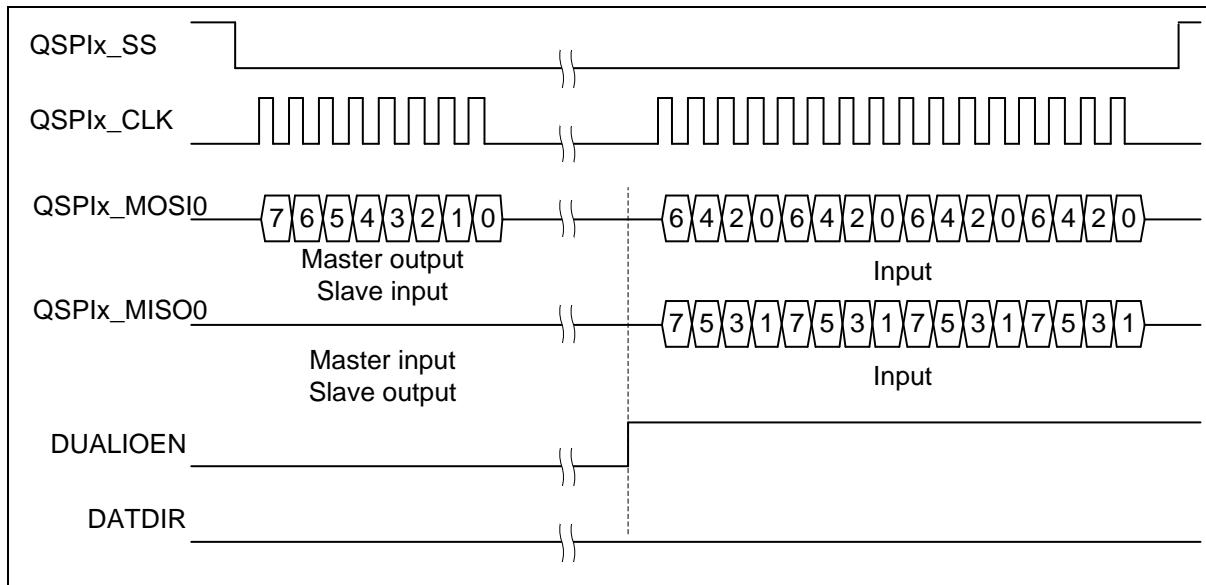


图 6.20-15 双输入模式的位时序

#### 6.20.5.10 四 I/O 模式

设置 QUADIOEN (QSPIx\_CTL[22]) = 1，为四 I/O 传输模式。DATDIR (QSPIx\_CTL[20]) 定义传输方向。当 DATDIR(QSPIx\_CTL[20])= 1，为发送，反之为接收。该功能支持 8, 16, 24, 和 32 位长度。

从机三线模式或字节重排功能时，不支持 4 I/O 模式。DUALIOEN (QSPIx\_CTL[21]) 和 QUADIOEN (QSPIx\_CTL[22]) 不能同时设置为 1。

对于四 I/O 模式，如果 QUADIOEN (QSPIx\_CTL[22]) 和 DATDIR (QSPIx\_CTL[20]) 都设成 1，则 QSPIx\_MOSI0 和 QSPIx\_MOSI1 输出偶数位，QSPIx\_MOSI0 和 QSPIx\_MOSI1 输出奇数位。如果 QUADIOEN (QSPIx\_CTL[22]) 置 1，DATDIR (QSPIx\_CTL[20]) 置 0，QSPIx\_MISO0, QSPIx\_MISO1, QSPIx\_MOSI0 和 QSPIx\_MOSI1 都是输入。

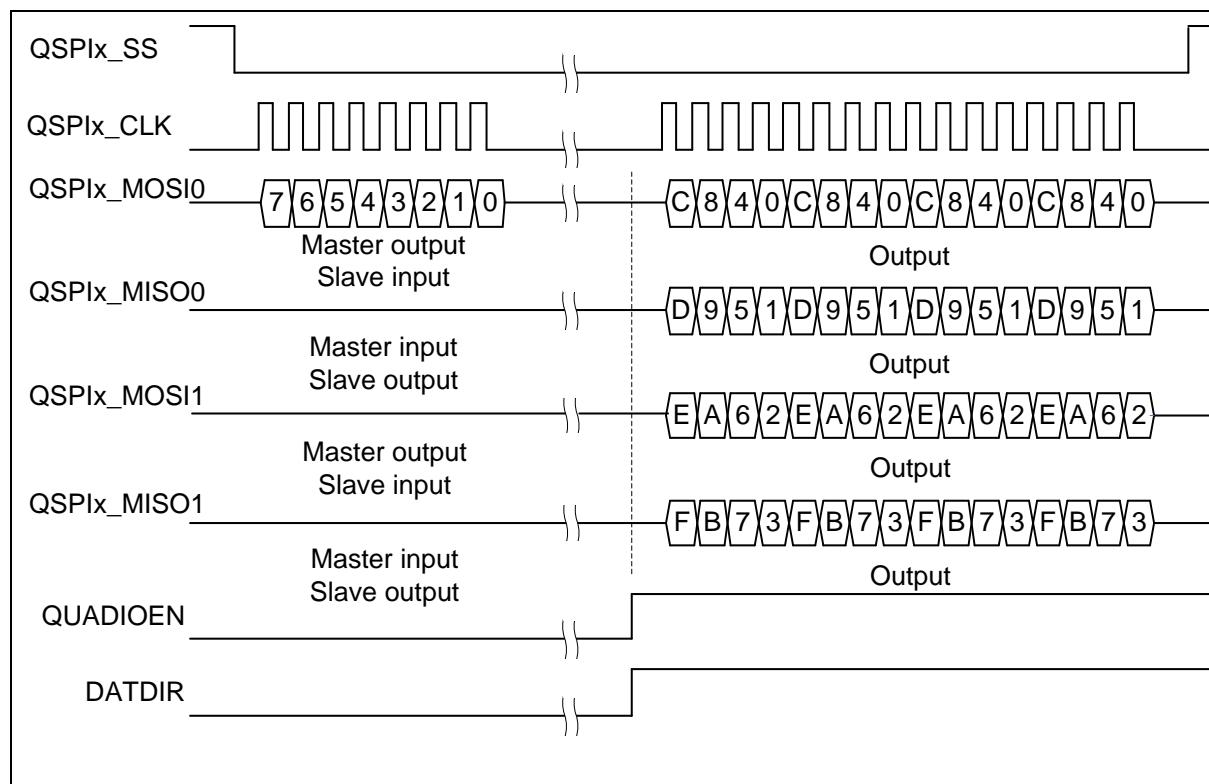


图 6.20-16 四 I/O 输出模式位顺序

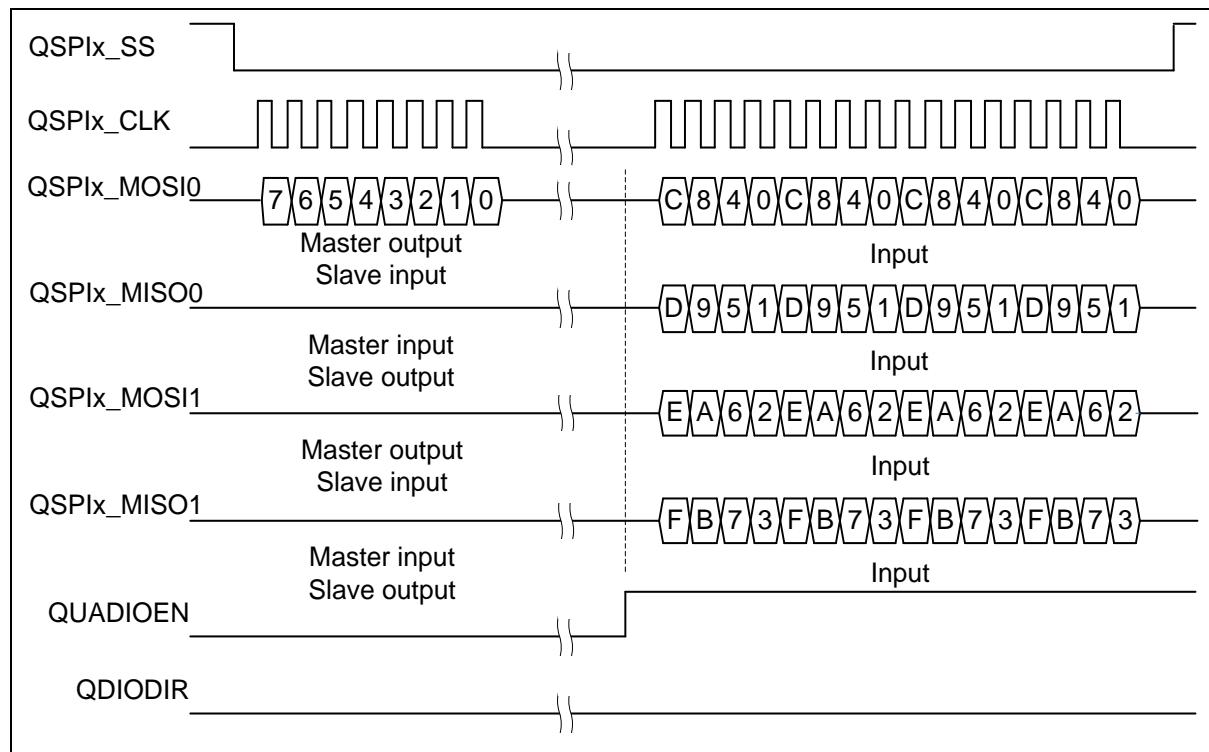


图 6.20-17 四 I/O 输入模式位顺序

### 6.20.5.11 FIFO缓存操作

QSPI 控制器配备了 4 个 32 位宽的发送和接收 FIFO 缓存。存放在发送 FIFO 缓存的数据会通过发送控制逻辑进行读取和发送。如果发送 FIFO 缓存满了，TXFULL (QSPIx\_STATUS[17]) 会被置 1。当 QSPI 传输逻辑单元抽出发送 FIFO 缓存的最后一个数据，发送 FIFO 缓存就为空了，TXEMPTY (QSPIx\_STATUS[16]) 位被置 1。注意 TXEMPTY (QSPIx\_STATUS[16]) 标志在最后一笔数据传输还在进行时就被置 1 了。在主机模式，当 FIFO 缓存写入数据或者 QSPI 总线上有任何事务，BUSY (QSPIx\_STATUS[0]) 位被设置为 1。(如：从机片选信号激活和 QSPI 控制器在从机模式正在接收数据)。当传送缓存为空且当前事务已经完成后，该位将设置为 0。因此，软件可以检查 BUSY (QSPIx\_STATUS[0]) 位的状态以确认 QSPI 是否已经空闲。

接收控制逻辑存储 QSPI 接收到的数据到接收 FIFO 缓存。有 FIFO 相关的状态位，像 RXEMPTY (QSPIx\_STATUS[8]) 和 RXFULL (QSPIx\_STATUS[9])，来表明当前 FIFO 缓存的状态。

发送和接收的阈值可以通过设置 TXTH (QSPIx\_FIFOCTL[30:28]) 和 RXTH (QSPIx\_FIFOCTL[26:24]) 来设定。当存储在发送 FIFO 缓存的有效数据数小于或等于 TXTH (QSPIx\_FIFOCTL[30:28]) 的设定时，TXTHIF (QSPIx\_STATUS[18]) 位会被置 1。当存储在接收 FIFO 缓存的有效数据数大于 RXTH (QSPIx\_FIFOCTL[26:24]) 的设定，RXTHIF (QSPIx\_STATUS[10]) 位会被置 1。.

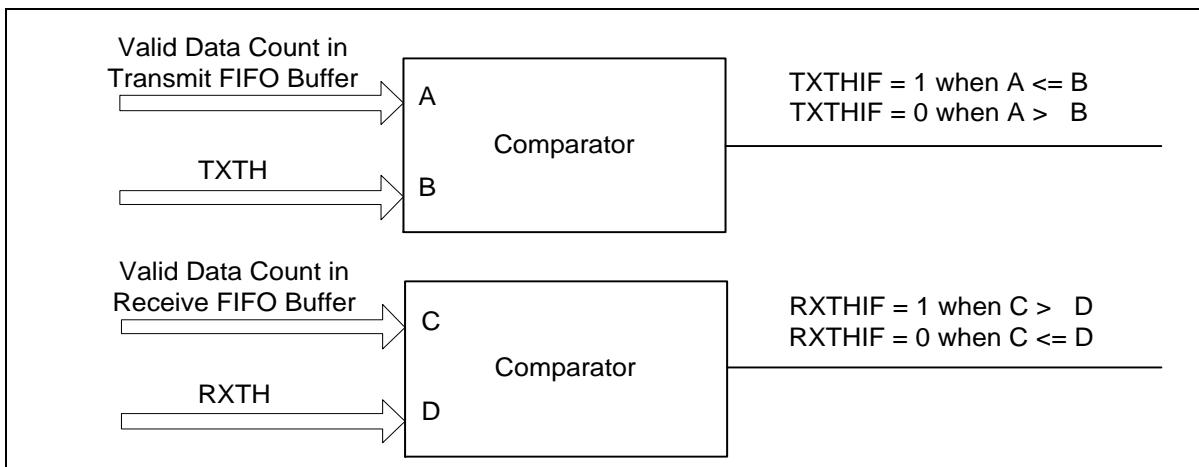


图 6.20-18 FIFO 阈值比较器

在主机模式，当第一个数据写入 QSPIx\_TX 寄存器时，TXEMPTY (QSPIx\_STATUS[16]) 标志将会被清除为 0。在 1 个 APB 时钟周期和 6 个外设时钟周期后，发送将开始。用户可以立即写下一个数据到 QSPIx\_TX 寄存器。QSPI 控制器将会在两个连续的事务之间插入一个休眠间隔，休眠间隔的长度由 SUSPITV (QSPIx\_CTL[7:4]) 的设定值决定。如果 SUSPITV (QSPIx\_CTL[7:4]) 等于 0，QSPI 控制器可以执行连续发送。只要 TXFULL (QSPIx\_STATUS[17]) 为 0，用户就可以向 QSPIx\_TX 寄存器写入新的数据。

如下图的例 1 所示，该图指明了 TXEMPTY (QSPIx\_STATUS[16]) 的更新条件以及 FIFO 缓存，移位寄存器和斜移缓存之间的关系。当 Data 0 写入 FIFO 缓存时，TXEMPTY (QSPIx\_STATUS[16]) 位被设置为 0。Data 0 将由内核逻辑加载到移位寄存器，此时，TXEMPTY (QSPIx\_STATUS[16]) 将设置为 1。移位寄存器中 Data 0 将逐位移入斜移缓存传输直到传输完成。

在例 2 中，该图为当 FIFO 缓存中有 8 个数据时，更新 TXFULL (QSPIx\_STATUS[17]) 的条件。当 TXFULL = 1 时，下一数据 Data 9 不能被写入 FIFO 缓存。

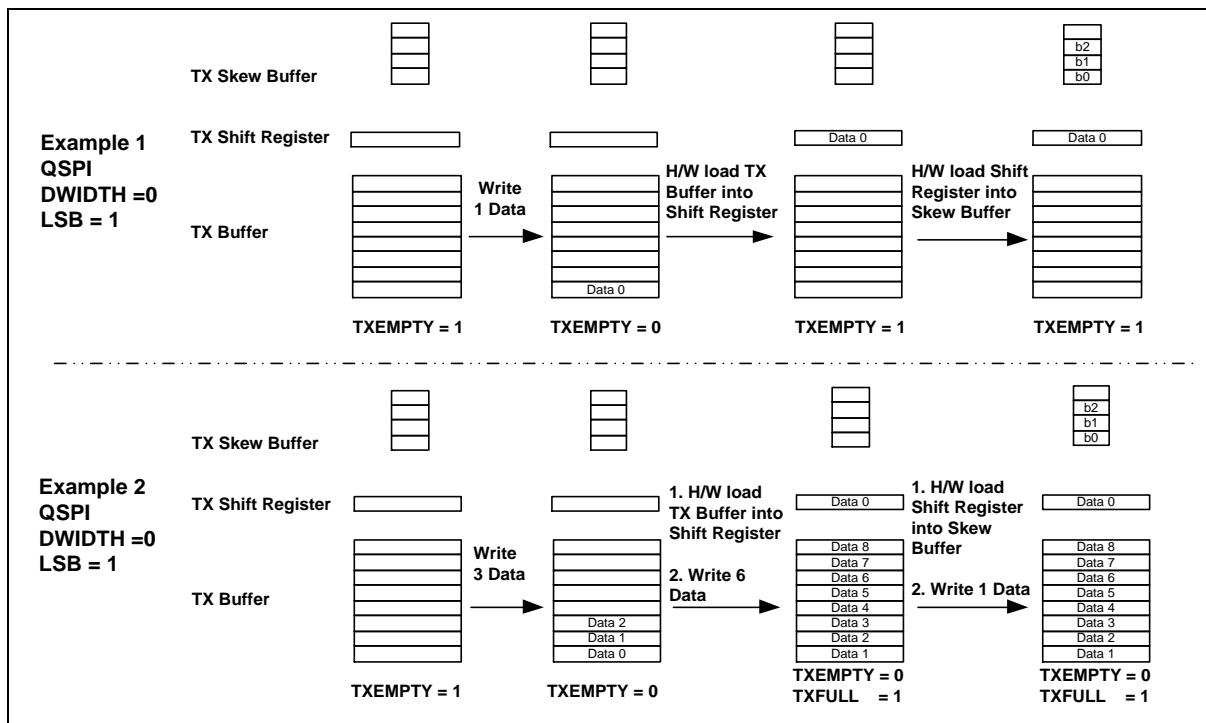


图 6.20-19 发送 FIFO 缓存示例

如果要发送的数据更新及时，接下来的事务将会被自动触发。如果在所有数据传输完成之后，QSPIx\_TX 寄存器没有被更新，则传输停止。

在主机模式接收操作中，串行数据从 QSPIx\_MISO 管脚接收并被存储在接收 FIFO 缓存。

接收数据 (Data 0's b0, b1, ...b31) 通过串行时钟 (QSPIx\_CLK) 先存到斜移缓存，然后再逐位移到移位寄存器。当接收数据位达到 DWIDTH (QSPIx\_CTL[12:8]) 的值时，内部逻辑将移位寄存器中的数据加载到 FIFO 缓存。当接收 FIFO 缓存有未读数据时 (参见示例 1)，RXEMPTY (QSPIx\_STATUS[8]) 将清 0。只要 RXEMPTY (QSPIx\_STATUS[8]) 为 0，用户就可以通过 QSPIx\_RX 寄存器来读取接收的数据。如果接收 FIFO 缓存包含 8 个未读数据，RXFULL (QSPIx\_STATUS[9]) 将设置为 1 (参见示例 2)。

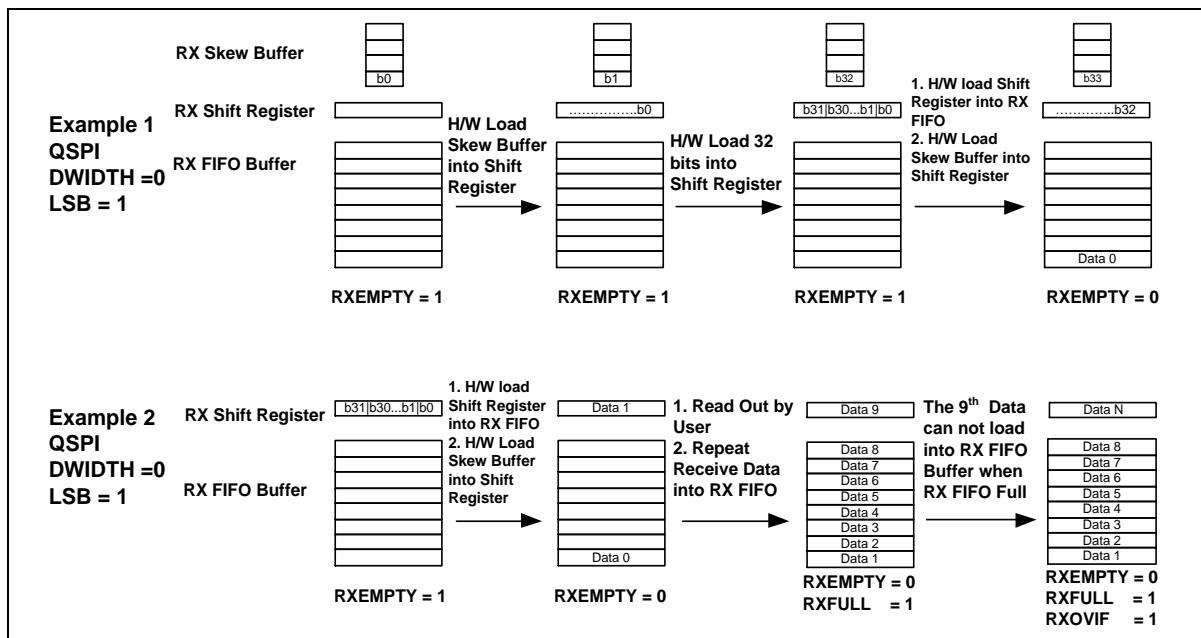


图 6.20-20 接收 FIFO 缓存示例

在从机模式中，当软件写数据到 QSPIx\_TX 寄存器时，数据将会被加载到发送 FIFO 缓存，同时 TXEMPTY (QSPIx\_STATUS[16]) 标志也将被设置为 0。当从设备从主机接收到时钟信号时，发送操作将会开始。只要 TXFULL (QSPIx\_STATUS[17]) 标志为 0，用户就可以写数据到 QSPIx\_TX 寄存器。所有数据都被 QSPI 发送逻辑单元发送出去后，而且软件没有再更新 QSPIx\_TX 寄存器 TXEMPTY (QSPIx\_STATUS[16]) 标志将会被设置为 1。

当从机片选信号有效时，如果没有任何数据写入 QSPIx\_TX 寄存器，发送下溢标志 TXUFIF (QSPIx\_STATUS[19]) 将被设置为 1。输出的数据在本次传输中将会通过设置 TXUFPOL (QSPIx\_FIFOCTL[6]) 保留，直到从机片选信号为无效状态。当传送下溢事件发生时，从机溢出运行标志 SLVURIF (QSPIx\_STATUS[7]) 将置为 1。同时 QSPIx\_SS 进入无效状态。

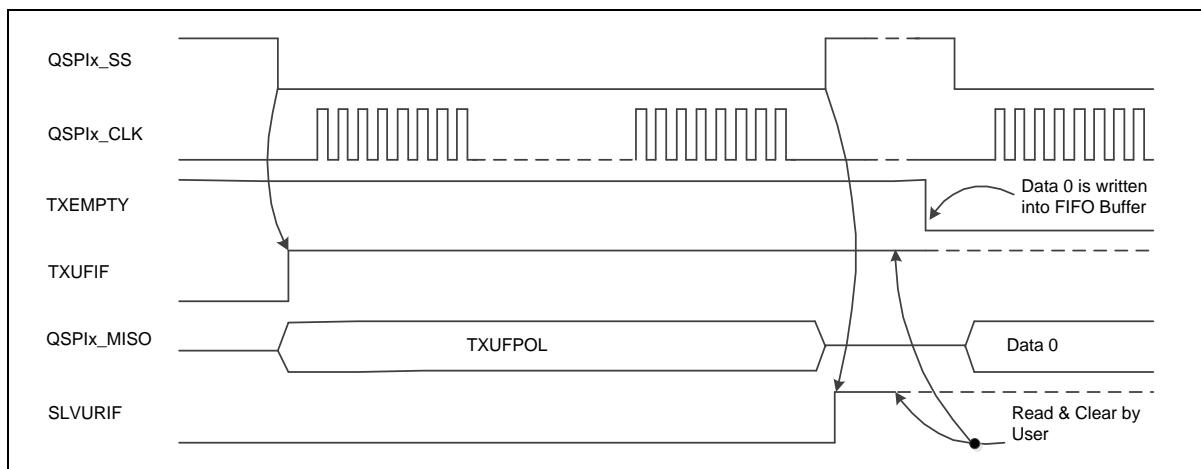


图 6.20-21 TX 下溢事件和从机溢出事件

在 2 位传输模式，在 2 个数据已经写入 TX FIFO 后，传送数据将被加载到移位寄存器。该模式同时使用两个移位寄存器和两个 4 级的斜移缓存。2 位传送模式的详细时序，请参考 2 位传输模式的相关介绍。

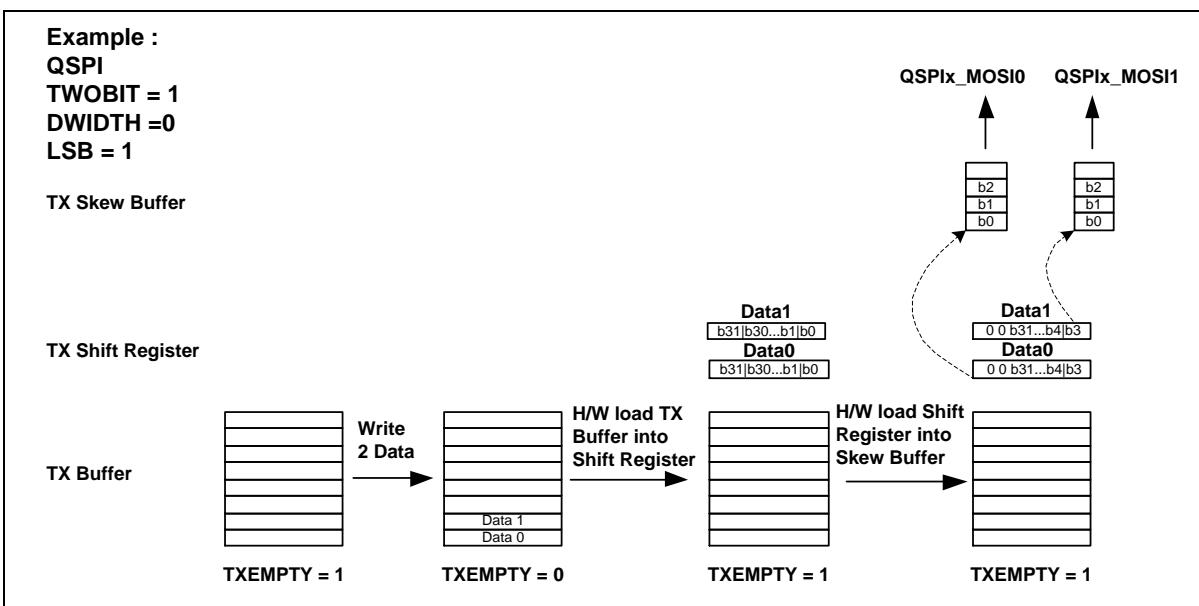


图 6.20-22 2 位传输模式 FIFO 缓存示例

在从机三线模式，如果在 QSPI 总线时钟出现前的 3 个外设时钟周期内，把数据写入发送 FIFO 缓存，则开始的 2 位数据是不可预知的（会保持上次传输最后一位的电平）。其它位则被 TXUFPOL (QSPIx\_FIFOCTL[6]) 保存。写入的数据将在下一次传输时发送。

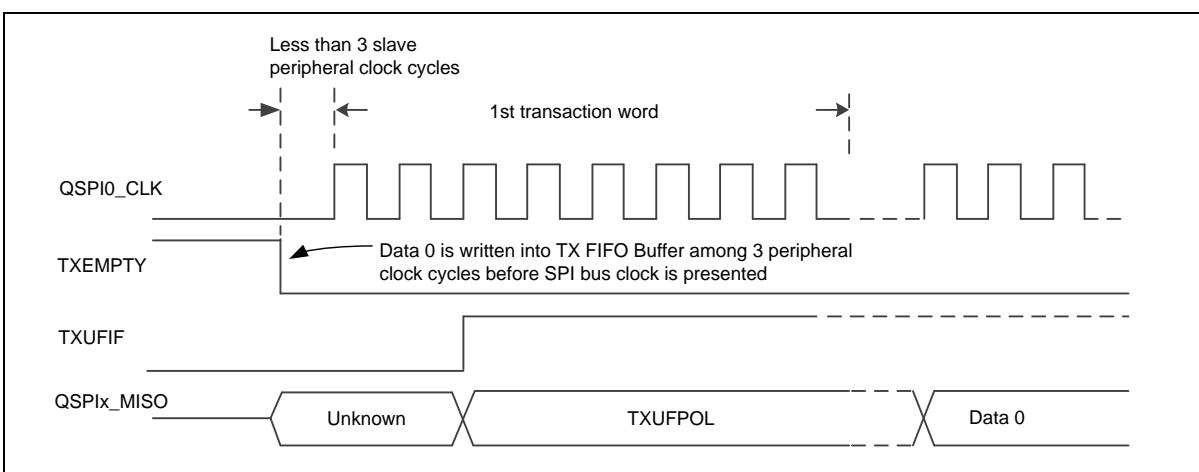


图 6.20-23 TX 下溢事件 (QSPI0 从机三线模式使能)

在从机模式接收操作中，串行数据从 QSPIx\_MOSI 管脚接收并存储到 QSPIx\_RX 寄存器。接收机制类似于主机模式的接收操作。如果接收 FIFO 缓存包含 8 个未读数据，RXFULL (QSPIx\_STATUS[9]) 将被置为 1，RXOVIF (QSPIx\_STATUS[11]) 也会被置为 1。如果在 QSPIx\_MOSI 管脚上有更多串行数据要接收，接下来的数据将被丢掉（参看接收 FIFO 缓存的示例图）。当从机片选线进入无效状态时，如果接收到位数数据与 DWIDTH (QSPIx\_CTL[12:8]) 设定的不一致，SLVBEIF (QSPIx\_STATUS[6]) 将设置为 1。.

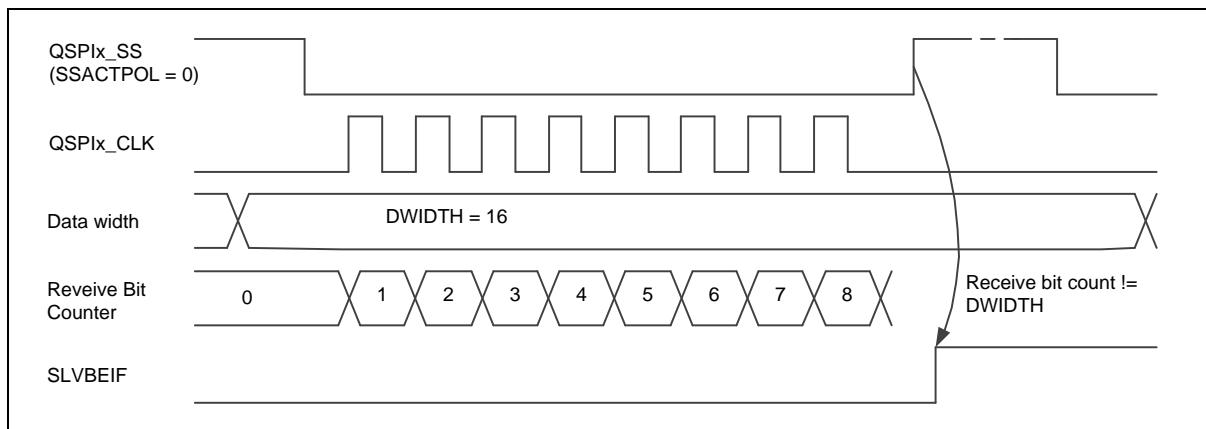


图 6.20-24 从机模式位计数错误

当从机选择信号有效且 **SLVTOCNT** (**QSPIx\_SSCTL[31:16]**) 的值不为 0 时，在串行时钟输入后，**QSPI** 控制器逻辑的从机超时计数器开始计数。在一次事务完成后或者 **SLVTOCNT** 设置为 0 后，该计数器清除。如果超时计数器的值在该事务完成前大于或等于 **SLVTOCNT** 的值，从机超时事件发生，**SLVTOIF** (**QSPIx\_STATUS[5]**) 将设置为 1。

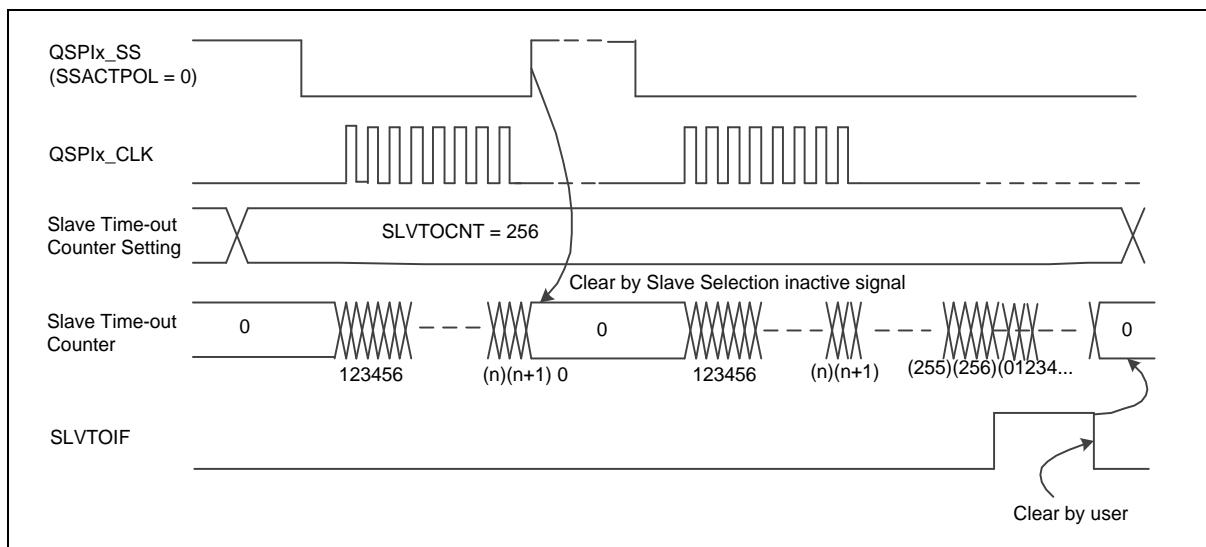


图 6.20-25 从机超时事件

控制器内置了一个接收超时功能模块。当接收 FIFO 非空，而且在主机模式时接收 FIFO 超过 64 个 **QSPI** 外设时钟周期或者从机模式时超过 576 个 **QSPI** 外设时钟周期没有读操作，则接收超时发生，**RXTOIF** (**QSPIx\_STATUS[12]**) 置 1。当接收 FIFO 有用户读取时，超时状态将自动清除。

### 6.20.5.12 中断

- **QSPI单元传输中断**

当 **QSPI** 控制器完成一个单元传输，单元传输中断标志 **UNITIF** (**QSPIx\_STATUS[1]**) 将会被设为 1。如果单元传输中断使能位 **UNITIEN** (**QSPIx\_CTL[17]**) 被置位，则单元传输中断事件将给 CPU 产生中断请求。单位传输中断标志位只能写 1 清零。.

- **QSPI从机片选有效/无效中断**

从机模式下，如果 **SPIEN** (**QSPIx\_CTL[0]**) 和 **SLAVE** (**QSPIx\_CTL[18]**) 置 1，从机片选信号进入有效/无效状态，从机片选有效/无效中断标志 **SSACTIF** (**QSPIx\_STATUS[2]**) 和 **SSINAIF** (**QSPIx\_STATUS[3]**) 会被置位。此时如果 **SSINAIVEN** (**QSPIx\_SSCTL[13]**) 或 **SSACTIVEN**

(QSPIx\_SSCTL[12]) 为 1, QSPI 控制器会产生中断。

- 从机超时中断

QSPI 在从机模式时, 用户可以通过从机超时功能识别有串行时钟输入但是超过了 SLVTOCNT (QSPIx\_SSCTL[31:16]) 中定义的从机外设时钟周期个数, 传输事务还没有完成。

当从机片选信号有效时, 而且 SLVTOCNT (QSPIx\_SSCTL[31:16]) 的值不为 0, 在串行时钟输入后, QSPI 控制器逻辑的从机超时计数器开始计数。在一次事务完成后或者 SLVTOCNT (QSPIx\_SSCTL[31:16]) 被置为 0, 该计数器将被清除。如果在一次事务传输完前, 超时计数器的值大于或等于 SLVTOCNT (QSPIx\_SSCTL[31:16]) 的值, 则从机超时事件发生, 同时 SLVTOIF (QSPIx\_STATUS[5]) 将设置为 1。如果 SLVTOIEN (QSPIx\_SSCTL[5]) 为 1, SPI 控制器将发生一个中断。

- 从机位计数错误中断

在从机模式, 当从机片选信号线进入无效状态时, 如果发送或接收到位数据个数与 DWIDTH (QSPIx\_CTL[12:8]) 设置的不一致, SLVBEIF (QSPIx\_STATUS[6]) 将被设置为 1。未传输完成的数据将在 TX 和 RX 移位寄存器中弃除。如果 SLVBEIEN (QSPIx\_SSCTL[8]) 位为 1, QSPI 控制器将发生一个中断。

注: 如果从机片选信号激活, 但是没有任何串行时钟输入, 当从机片选信号进入无效状态时, SLVBEIF (QSPIx\_STATUS[6]) 也将设置为 1。

- TX 下溢中断

在 QSPI 从机模式, 如果没有任何数据写入 QSPIx\_TX 寄存器, 当从机选择信号激活时, TXUFIIF (QSPIx\_STATUS[19]) 将设置为 1。如果 TXUFIEN (QSPIx\_FIFOCTL[7]) 为 1, QSPI 控制器将发生发送下溢中断。

注: QSPI 从机模式发生下溢事件后, 有两种方法可以使其恢复空闲状态并进入下个传输: (1) 置 TXRST 为 1 (2) 从机片选信号切为无效状态。

- 从机 TX 下溢运行中断

当 QSPIx\_SS 进入无效状态时, 如果有 TX 下溢事件发生, SLVURIF (QSPIx\_STATUS[7]) 将被设置为 1。如果 SLVURIEN (QSPIx\_SSCTL[9]) 为 1, QSPI 控制器将发生发送溢出运行中断。

注意: 在从机三线模式, 从机片选信号被认为是一直有效的, 用户需要去查看 TXUFIIF (QSPIx\_STATUS[19]) 位来确定是否有发生 TX 下溢事件。

- 接收溢出中断

在从机模式, 如果接收 FIFO 缓存已有 8 个未读数据, RXFULL (QSPIx\_STATUS[9]) 和 RXOVIF (QSPIx\_STATUS[11]) 标志将会被设置为 1。如果从 QSPI 总线上接收到更多串行数据, 多余的数据将会丢失。如果 RXOVIEN (QSPIx\_FIFOCTL[5]) 为 1, QSPI 控制器将发生接收溢出中断。

- 接收 FIFO 超时中断

如果在 FIFO 里有一个接收到的数据, 在主机模式下用户超过 64 个 QSPI 外设时钟周期没有去读取, 或者从机模式下超过 576 个 QSPI 外设时钟周期没有去读取, 如果接收超时中断使能位 RXTOIEN (QSPIx\_FIFOCTL[4]) 有设置为 1, 则会向系统发出一个 RX 超时中断。

- 发送 FIFO 中断

在 FIFO 模式, 如果发送 FIFO 缓存的有效数据少于或等于 TXTH (QSPIx\_FIFOCTL[30:28]) 的设定值, 发送 FIFO 中断标志 TXTHIF (QSPIx\_STATUS[18]) 会被置 1。如果发送 FIFO 中断位 TXTHIEN (QSPIx\_FIFOCTL[3]) 为 1, 则 QSPI 控制器会向系统产生一个发送 FIFO 中断。

- 接收 FIFO 中断

在 FIFO 模式，如果接收 FIFO 缓存的有效数据大于 RXTH (QSPIx\_FIFOCTL[26:24]) 的设定值，接收 FIFO 中断标志 RXTHIF (QSPIx\_STATUS[10]) 会被设置为 1。如果接收 FIFO 中断位 RXTHIEN (QSPIx\_FIFOCTL[2]) 有设置为 1，QSPI 控制器将会向系统产生一个接收 FIFO 中断。

#### 6.20.6 时序图

从机选择信号的有效状态可以由 SSACTPOL (QSPIx\_SSCTL[2]) 配置。串行时钟的空闲状态可以通过 CLKPOL (QSPIx\_CTL[3]) 配置为高电平或低电平。传输字段长度在 DWIDTH (QSPIx\_CTL[12:8]) 中定义，发送/接收数据是以 MSB 还是 LSB 优先由 LSB 位 (QSPIx\_CTL[13]) 定义。用户也可以通过 TXNEG/RXNEG (QSPIx\_CTL[2:1]) 来选择发送/接收数据时串行时钟的边沿。四种 QSPI 主机/从机操作时序图以及相关设置如下图。

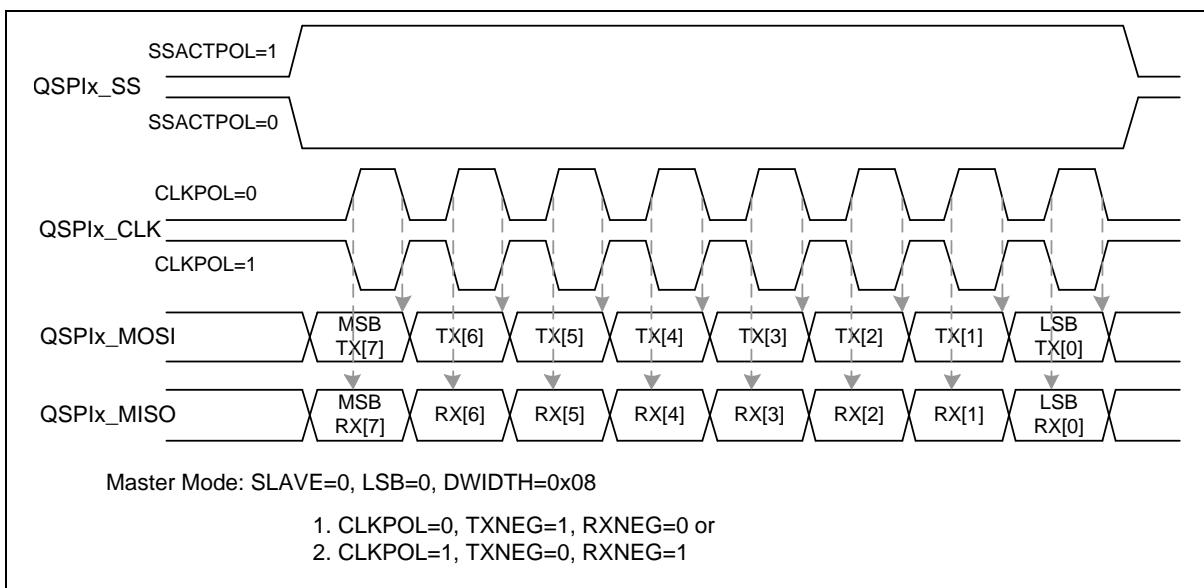


图 6.20-26 QSPI 主机模式时序

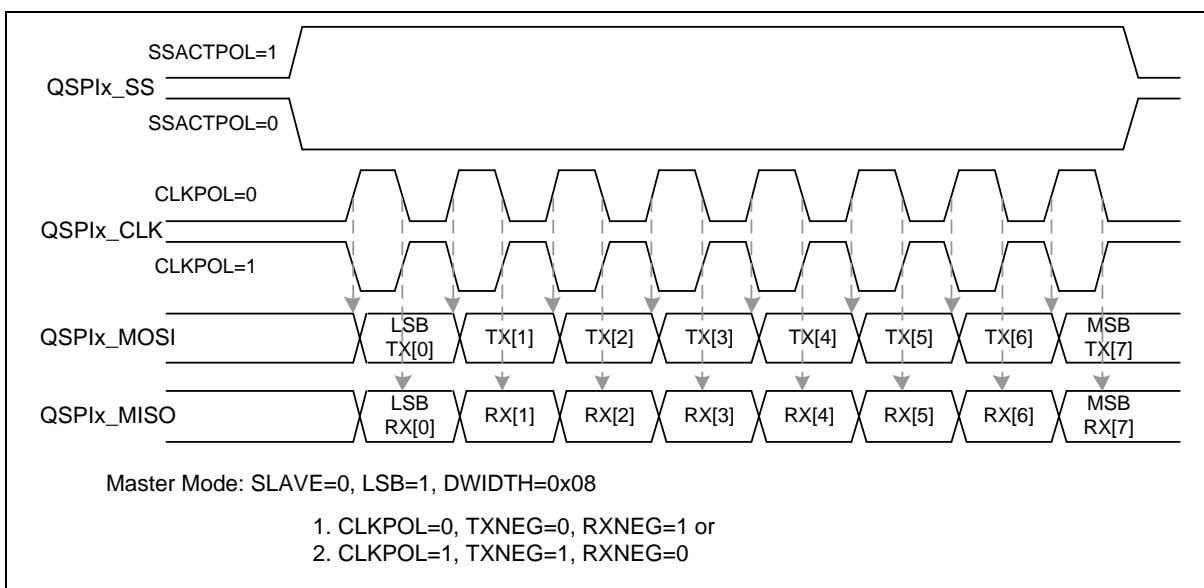


图 6.20-27 QSPI 主机模式时序 (QSPIx\_CLK 交替相位)

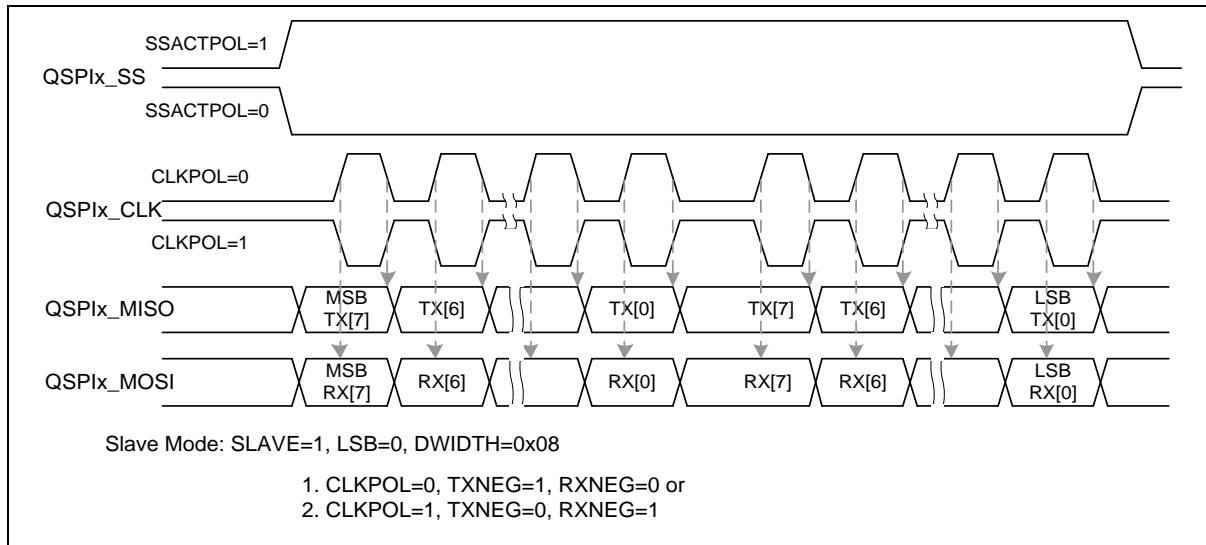


图 6.20-28 QSPI 从机模式时序

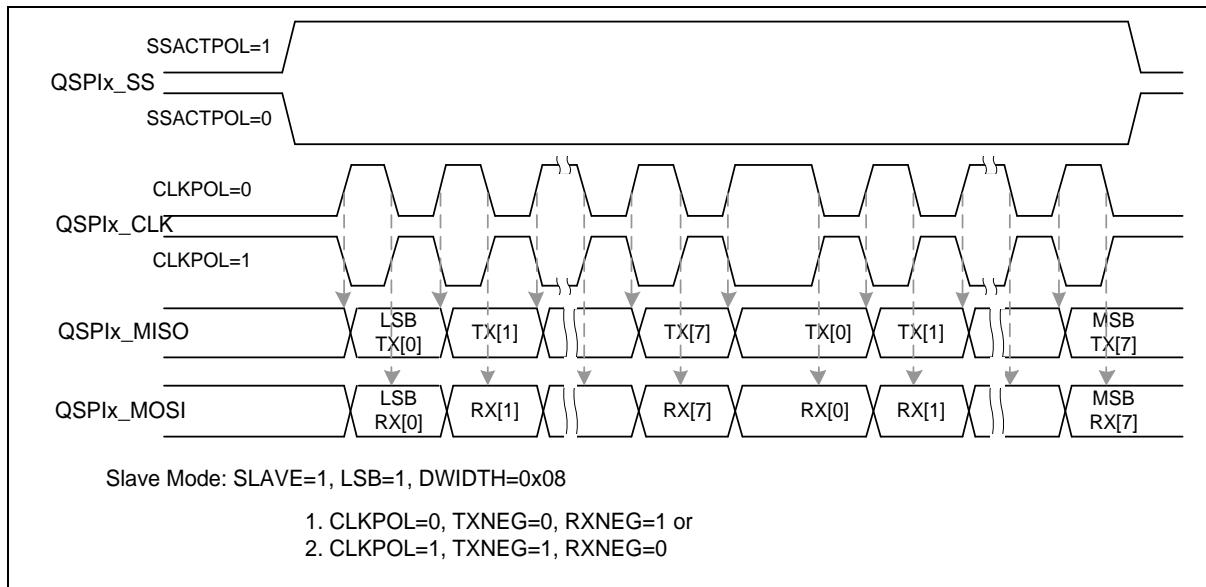


图 6.20-29 QSPI 从机模式时序 (QSPIx\_CLK 交替相位)

### 6.20.7 编程示例

#### 例 1:

QSPI 配置为全双工主机时序如下：

- QSPI 时钟上沿锁存数据
- QSPI 时钟下沿发送数据
- MSB 优先

- QSPI 时钟空闲时处于低电平
- 每次只收发一个字节
- 片选信号低电平有效。

操作流程如下：

1. 在寄存器 DIVIDER (SPIx\_CLKDIV [8:0]) 配置 SPI 输出时钟频率
2. 写寄存器 QSPIx\_SSCTL 配置 QSPI 主机相关模式：
  - 1) 清 0 AUTOSS (QSPIx\_SSCTL[3]) 关闭自动从机选择功能。
  - 2) 清 0 SSACTPOL (QSPIx\_SSCTL[2]) 设置片选信号低电平有效。
  - 3) 置 SS (QSPIx\_SSCTL[0]) 为 1 使能从机片选信号，激活片外从机设备。
3. 写寄存器 QSPIx\_CTL 控制 QSPI 主机行为：
  - 1) 设 SLAVE (QSPIx\_CTL[18]) 为 0 配置 QSPI 为主机设备。
  - 2) 清 0 CLKPOL (QSPIx\_CTL[3]) 强制 QSPI 空闲状态为低电平。
  - 3) 置 TXNEG (QSPIx\_CTL[2]) 为 1 配置 QSPI 总线时钟下沿传输数据。
  - 4) 清 0 RXNEG (QSPIx\_CTL[1]) 配置 QSPI 总线时钟上沿锁存数据。
  - 5) 设置传输长度 DWIDTH 为 8 位 (QSPIx\_CTL[12:8] = 0x08)。
  - 6) 清 0 LSB (QSPIx\_CTL[13]) 配置 MSB 传输优先。
4. 置 SPIEN (QSPIx\_CTL[0]) 位 1 使能数据传输。
5. QSPI 主机想要写一个字节的数据到片外设备时，将数据写到 QSPIx\_TX 寄存器即可。
6. 等待 QSPI 中断发生 (如果 UNITIEN (QSPIx\_CTL[17]) 为 1)，或者轮询单元传输中断标志 UNITIF (QSPIx\_STATUS[1])。
7. 从 QSPIx\_RX 读接收到的数据。
8. 重复步骤 5 继续其他数据的传输，或者配置 SS (QSPIx\_SSCTL[0]) 为 0 让片选信号无效以关闭片外从机设备。.

## 例 2:

QSPI 为全双工从机。片外主机按下面的规则通讯：

- QSPI 时钟上沿锁存数据
- QSPI 时钟下沿发送数据
- LSB 优先
- QSPI 时钟空闲时处于高电平
- 每次只收发一个字节
- 片选信号高电平有效。

操作流程如下：

1. 写寄存器 QSPIx\_SSCTL 配置为从机模式.
2. 置 SSACTPOL (QSPIx\_SSCTL[2]) 为 1 配置片选信号高电平有效。.

3. 写 QSPIx\_CTL 配置从机行为
  - 1) 设 SLAVE (QSPIx\_CTL[18]) 为 1 配置 QSPI 控制单元为从机设备。
  - 2) 置 CLKPOL (QSPIx\_CTL[3]) 为 1 选择 QSPI 空闲状态为高电平。
  - 3) 置 TXNEG (QSPIx\_CTL[2]) 为 1 配置 QSPI 总线时钟下降沿传输数据。
  - 4) 清 0 RXNEG (QSPIx\_CTL[1]) 配置 QSPI 总线时钟上升沿锁存数据。
  - 5) 设置传输长度 DWIDTH 为 8 位 (QSPIx\_CTL[12:8] = 0x08)。
4. 置 LSB (QSPIx\_CTL[13]) 为 1 配置 LSB 传输优先
5. 置 SPIEN (QSPIx\_CTL[0]) 为 1。等待片外主机设备的片选信号和 QSPI 时钟输入来启动数据传输
6. 如果 QSPI 从机想要发送一个字节的数据到片外主机，写数据到 QSPIx\_TX 寄存器即可。
7. 如果 QSPI 从机想要从片外主机接收一个字节的数据，且用户不关心什么数据被传输，则软件不需要去更新 QSPIx\_RX 寄存器
8. 等待 QSPI 中等待 QSPI 中断发生 (如果 UNITIEN (QSPIx\_CTL[17]) 为 1)，或者轮询单元传输中断标志 UNITIF (QSPIx\_STATUS[1])。
9. 从 QSPIx\_RX 寄存器读接收到的数据。
10. 回到 7 继续下个字节发送或停止发送

### 6.20.8 寄存器映射

R: 只读, W: 只写, R/W: 读/写

寄存器	偏移量	R/W	描述	复位值
<b>QSPI基址:</b> <b>QSPIx_BA = 0x4006_0000 + (0x0000_1000 * x)</b> x=0				
<b>QSPIx_CTL</b>	QSPIx_BA+0x00	R/W	QSPI控制寄存器	0x0000_0034
<b>QSPIx_CLKDIV</b>	QSPIx_BA+0x04	R/W	QSPI时钟分频寄存器	0x0000_0000
<b>QSPIx_SSCTL</b>	QSPIx_BA+0x08	R/W	QSPI从机选择控制寄存器	0x0000_0000
<b>QSPIx_PDMACTL</b>	QSPIx_BA+0x0C	R/W	QSPI PDMA控制寄存器	0x0000_0000
<b>QSPIx_FIFOCTL</b>	QSPIx_BA+0x10	R/W	QSPI FIFO控制寄存器	0x4400_0000
<b>QSPIx_STATUS</b>	QSPIx_BA+0x14	R/W	QSPI状态寄存器	0x0005_0110
<b>QSPIx_TX</b>	QSPIx_BA+0x20	W	QSPI数据发送寄存器	0x0000_0000
<b>QSPIx_RX</b>	QSPIx_BA+0x30	R	QSPI 数据接收寄存器	0x0000_0000

## 6.20.9 寄存器描述

QSPIx\_CTL      QSPI 控制寄存器

寄存器	偏移量	R/W	描述	复位值
QSPIx_CTL	SPIx_BA+0x00	R/W	QSPI 控制寄存器	0x0000_0034

Note: Not supported in I<sup>2</sup>S mode.

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved	QUADIOEN	DUALIOEN	DATDIR	REORDER	SLAVE	UNITIEN	TWOBIT
15	14	13	12	11	10	9	8
RXONLY	HALFDPX	LSB	DWIDTH				
7	6	5	4	3	2	1	0
SUSPITV				CLKPOL	TXNEG	RXNEG	SPIEN

位	描述
[31:21]	<b>Reserved</b> 保留
[22]	<b>QUADIOEN</b> 四 I/O 模式 使能位 0 = 四 I/O 模式关闭 1 = 四 I/O 模式使能
[21]	<b>DUALIOEN</b> 双 I/O 模式 使能位 0 = 双 I/O 模式关闭 1 = 双 I/O 模式使能
[20]	<b>DATDIR</b> 数据方向控制 该位用来选则半双工模式和双/四 I/O 模式下数据输入/输出方向 0 = QSPI 数据输入 1 = QSPI 数据输出
[19]	<b>REORDER</b> 字节重排功能使能位 0 = 字节重排功能禁止 1 = 字节重排功能使能. 每两个字节之间会被插入一个休眠间隔, 休眠间隔的时间取决于 SUSPITV 的设置 注: 仅在 DWIDTH 为 16, 24 和 32 位时, 该功能有效。
[18]	<b>SLAVE</b> 从机模式控制 0 = 主机模式. 1 = 从机模式
[17]	<b>UNITIEN</b> 单元传输中断使能位 0 = QSPI 单元传输中断禁止 1 = QSPI 单元传输中断使能
[16]	<b>TWOBIT</b> 2 位传输模式使能位

		<p>0 = 2 位传输模式关闭 1 = 2 位传输模式使能 <b>注:</b> 当 2-位传输模式使能时, 第一个串行传输位数据来自第一个 FIFO 缓存数据, 第二个串行传输位数据来自第二个 FIFO 缓存数据。同样, 第一个接收位数据存储到第一个 FIFO 缓存, 第二个接收位数据存放到第二个 FIFO 缓存。</p>
[15]	<b>RXONLY</b>	<p><b>只接收模式使能位 (仅主机)</b> 该只用于主机模式。在只接收模式, QSPI 主机总线时钟会一直保持以接收数据, 忙标志会一直置位。 0 = 只接收模式关闭 1 = 只接收模式使能</p>
[14]	<b>HALFDPX</b>	<p><b>QSPI 半双工传输使能位</b> 该位用来切换 QSPI 传输全双工模式和半双工模式。半双工模式下 DATDIR (QSPIx_CTL[20]) 用来选择数据传输方向。 0 = QSPI 工作在全双工模式 1 = QSPI 工作在半双工模式</p>
[13]	<b>LSB</b>	<p><b>LSB 优先发送</b> 0 = MSB, 具体发送/接收寄存器的哪一位首先被发送/接收, 取决于 DWIDTH 的设定值。 1 = LSB, SPI TX 寄存器的位 0, 首先被发送到 SPI 数据输出管脚, 从 QSPI 数据输入管脚上接收到的第一个数据位将会被放置到 RX 寄存器 (QSPI_RX 的位 0 ) LSB 的位置.</p>
[12:8]	<b>DWIDTH</b>	<p><b>数据宽度</b> 该位用来标示每次传输时会有多少位被发送/接收。最低为 8 位, 最高为 32 位。 DWIDHT = 0x08 .... 8 位. DWIDHT = 0x09 .... 9 位. ..... DWIDHT = 0x1F .... 31 位. DWIDHT = 0x00 .... 32 位.</p>
[7:4]	<b>SUSPITV</b>	<p><b>休眠间隔 (仅主机)</b> 该四位用来配置在一次数据传输过程中连续两个发送/接收事务之间的休眠间隔。休眠间隔是从当前事务字的最后一个时钟边沿到接下来的事务的第一个边沿时钟。默认值是 0x3. 休眠间隔的周期可以根据下面公式获得:  <math display="block">(\text{SUSPITV}[3:0] + 0.5) * \text{SPICLK} \text{ 时钟周期}</math>         例:  <math display="block">\text{SUSPITV} = 0x0 \dots 0.5 \text{ SPICLK 时钟周期}</math> <math display="block">\text{SUSPITV} = 0x1 \dots 1.5 \text{ SPICLK 时钟周期}</math> <math display="block">\dots</math> <math display="block">\text{SUSPITV} = 0xE \dots 14.5 \text{ SPICLK 时钟周期}</math> <math display="block">\text{SUSPITV} = 0xF \dots 15.5 \text{ SPICLK 时钟周期}</math> </p>
[3]	<b>CLKPOL</b>	<p><b>时钟极性</b> 0 = QSPI 总线空闲时默认低 1 = QSPI 总线空闲时默认高</p>
[2]	<b>TXNEG</b>	<p><b>下降沿发送</b> 0 = 在 QSPI 总线时钟的上升沿发送数据 1 = 在 QSPI 总线时钟的下降沿发送数据</p>
[1]	<b>RXNEG</b>	<p><b>下降沿接收</b></p>

		0 = 在 QSPI 总线时钟的上升沿锁存数据 1 = 在 QSPI 总线时钟的下降沿锁存数据
[0]	<b>SPIEN</b>	<b>QSPI 传输控制使能位</b> 在主机模式下， FIFO 缓存有数据时，该位设置为 1 后，开始传输。在从机模式下，该位置为 1 时，该设备已准备好接收数据。 0 = 禁止控制传输 1 = 使能控制传输 <b>注:</b> 在更改 SPIx_CTL, SPIx_CLKDIV, SPIx_SSCTL 和 SPIx_FIFOCTL 寄存器的配置前，用户需清 0 SPIEN (SPIx_CTL[0])，并且确定 SPIENSTS (SPI_STATUS[15]) 的值为 0。

QSPIx\_CLKDIV QSPI 时钟分频寄存器

寄存器	偏移量	R/W	描述	复位值
QSPIx_CLKDIV	SPIx_BA+0x04	R/W	QSPI 时钟分频寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							DIVIDER
7	6	5	4	3	2	1	0
DIVIDER							

位	描述	
[31:9]	Reserved	保留
[8:0]	DIVIDER	<p><b>时钟除频</b></p> <p>该域的值是产生外设时钟, <math>f_{spi\_eclk}</math> 和 QSPI 主机的总线时钟的除频器的值。频率计算公式如下:</p> $f_{spi\_eclk} = \frac{f_{spi\_clock\_src}}{(DIVIDER + 1)}$ <p>其中,</p> $f_{spi\_clock\_src}$ 为外设时钟源, 定义于时钟控制寄存器 CLK_CLKSEL2 中 <b>注:</b> I <sup>2</sup> S 模式不支持

注意: 谨慎配置 DIVIDER , QSPI 时钟频率不能大于系统时钟频率

## QSPIx\_SSCTL QSPI 从机选择控制寄存器

寄存器	偏移量	R/W	描述	复位值
QSPIx_SSCTL	SPIx_BA+0x08	R/W	SPI 从机选择控制寄存器	0x0000_0000

注: I<sup>2</sup>S 模式不支持.

31	30	29	28	27	26	25	24
SLVTOCNT							
23	22	21	20	19	18	17	16
SLVTOCNT							
15	14	13	12	11	10	9	8
<b>Reserved</b>		<b>SSINAIEN</b>	<b>SSACTIEN</b>	<b>Reserved</b>		<b>SLVURIEN</b>	<b>SLVBEIEN</b>
7	6	5	4	3	2	1	0
<b>Reserved</b>	<b>SLVTORST</b>	<b>SLVTOIEN</b>	<b>SLV3WIRE</b>	<b>AUTOSS</b>	<b>SSACTPOL</b>	<b>Reserved</b>	<b>SS</b>

位	描述	
[31:16]	<b>SLVTOCNT</b>	从机模式超时周期 在从机模式，当在从机片选信号有效期间有总线时钟输入时，这些位表示超时周期。超时计数器的时钟源是从机外设时钟。如果该位值为 0，表示禁止从机模式超时功能。
[15:14]	<b>Reserved</b>	保留
[13]	<b>SSINAIEN</b>	从机片选无效中断使能位 0 = 从机片选无效中断禁止 1 = 从机片选无效中断使能
[12]	<b>SSACTIEN</b>	从机片选有效中断使能位 0 = 从机片选有效中断禁止 1 = 从机片选有效中断使能
[11:10]	<b>Reserved</b>	保留
[9]	<b>SLVURIEN</b>	从机模式 TX 下溢中断使能位 0 = 从机模式 TX 下溢中断禁止 1 = 从机模式 TX 下溢中断使能
[8]	<b>SLVBEIEN</b>	从机模式位计数错误中断使能位 0 = 从机模式位计数错误中断禁止 1 = 从机模式位计数错误中断使能
[7]	<b>Reserved</b>	保留
[6]	<b>SLVTORST</b>	从机模式超时复位控制 0 = 从机模式超时事件发生时，TX 和 RX 控制电路不会复位 1 = 从机模式超时事件发生时，TX 和 RX 控制电路由硬件复位
[5]	<b>SLVTOIEN</b>	从机模式超时中断使能位 0 = 从机模式超时中断禁止

		1 = 从机模式超时中断使能
[4]	<b>SLV3WIRE</b>	<p><b>从机三线模式使能位</b>            只有QSPI支持从机三线模式。从机三线模式中，QSPI控制器可以工作在三线接口，QSPIx_CLK, QSPIx_MISO和SPIx_MOSI。            0 = 四线双向接口            1 = 三线双向接口</p>
[3]	<b>AUTOSS</b>	<p><b>自动从机片选功能使能位 (仅主机)</b>            0 = 自动从机片选功能禁止，从机片选信号由SS (QSPIx_SSCTL[0]) 控制            1 = 自动从机片选功能使能</p>
[2]	<b>SSACTPOL</b>	<p><b>从机片选有效极性</b>            该位定义从机片选信号 (QSPIx_SS) 有效的极性            0 = 片选信号 QSPIx_SS 低电平有效            1 = 片选信号 QSPIx_SS 高电平有效</p>
[1]	<b>Reserved</b>	保留
[0]	<b>SS</b>	<p><b>从机片选控制位 (仅主机)</b>            如果 AUTOSS 为 0,            0 = 设置 SPIx_SS 为无效状态            1 = 设置 SPIx_SS 为有效状态            如果 AUTOSS 置 1,            0 = 保持 SPIx_SS 为无效状态            1 = 进行数据传输时，SPIx_SS 会自动切换到有效状态；无数据传输时则会进入无效状态。            SPIx_SS 的有效状态定义于 SSACTPOL (SPIx_SSCTL[2])。</p>

QSPIx\_PDMACTL QSPI PDMA 控制寄存器

寄存器	偏移量	R/W	描述	复位值
QSPIx_PDMACTL	SPIx_BA+0x0C	R/W	QSPI PDMA 控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved					PDMARST	RXPDMAEN	TXPDMAEN

位	描述	
[31:3]	Reserved	保留
[2]	PDMARST	<b>PDMA 复位</b> 0 = 无效 1 = 复位 QSPI 控制器的 PDMA 控制逻辑。该位自动清 0。
[1]	RXPDMAEN	<b>PDMA 接收使能位</b> 0 = PDMA 接收禁止 1 = PDMA 接收使能
[0]	TXPDMAEN	<b>PDMA 发送使能位</b> 0 = PDMA 发送禁止 1 = PDMA 发送使能 <b>注:</b> 在 QSPI 主机模式支持全双工传输，如果发送和接收 PDMA 功能都使能，接收 PDMA 功能不能在发送 PDMA 功能之前使能。用户可以先使能发送 PDMA 功能或者同时使能两个功能。

QSPIx FIFOCTL QSPI FIFO 控制寄存器

寄存器	偏移量	R/W	描述	复位值
QSPIx_FIFOCTL	QSPIx_BA+0x10	R/W	QSPI FIFO 控制寄存器	0x2200_0000

31	30	29	28	27	26	25	24
Reserved	TXTH				Reserved	RXTH	
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved						TXFBCLR	RXFBCLR
7	6	5	4	3	2	1	0
TXUFIEN	TXUFPOL	RXOVIEN	RXTOIEN	TXTHIEN	RXTHIEN	TXRST	RXRST

位	描述	
[31:30]	Reserved	保留
[30:28]	TXTH	<b>发送 FIFO 阈值</b> 如果发送 FIFO 缓存的有效数据个数小于或者等于 TXTH 的设置，TXTHIF 将会被设置为 1，否则 TXTHIF 将会被设置为 0。
[27:26]	Reserved	保留
[26:24]	RXTH	<b>接收 FIFO 域值</b> 如果接收 FIFO 缓存的有效数据个数大于 RXTH 的设置，RXTHIF 将会被设置为 1，否则 RXTHIF 将会被设置为 0。
[23:10]	Reserved	保留
[9]	TXFBCLR	<b>清发送 FIFO 缓存</b> 0 = 无效 1 = 清除发送 FIFO 指针，TXFULL 位将清除为 0 而且 TXEMPTY 位也将设置为 1。该位置 1 后，在一个系统时钟后，由硬件清 0。 <b>注:</b> TX 移位寄存器不会清空
[8]	RXFBCLR	<b>清接收 FIFO 缓存</b> 0 = 无效 1 = 清除接收 FIFO 指针，RXFULL 位将清除为 0 而且 RXEMPTY 位将设置为 1。该位设置为 1 后，在一个系统时钟后，由硬件清 0。 <b>注:</b> RX 移位寄存器不会清空
[7]	TXUFIEN	<b>TX 下溢中断使能位</b> 从机模式下 TX 下溢中断事件发生时，TXUFI (QSPIx_STATUS[19]) 会被置 1。该位用来使能 TX 下溢中断。 0 = 从机 TX 下溢中断禁止 1 = 从机 TX 下溢中断使能

[6]	<b>TXUFFPOL</b>	<b>TX 下溢数据极性</b> 0 = 在从机模式，如果有发送下溢事件发生，QSPI 数据输出保持为 0。 1 = 在从机模式，如果有发送下溢事件发生，QSPI 数据输出保持为 1。 <b>注：</b> 1. 从机片选信号有效时如果没有任何数据发送则会发生下溢事件。 2. TX 下溢事件发生时，QSPIx_MISO 管脚状态将由该位决定而不管 TX FIFO 是否为空。存储在 TX FIFO 的数据会在下一个传输帧通过 QSPIx_MISO 发送出去。
[5]	<b>RXOVIEN</b>	<b>接收 FIFO 溢出中断使能位</b> 0 = 接收 FIFO 溢出中断禁止 1 = 接收 FIFO 溢出中断使能
[4]	<b>RXTOIEN</b>	<b>从机接收超时中断使能位</b> 0 = 从机接收超时中断禁止 1 = 从机接收超时中断使能
[3]	<b>TXTHIEN</b>	<b>发送 FIFO 阈值中断使能位</b> 0 = TX FIFO 阈值中断禁止 1 = TX FIFO 阈值中断使能
[2]	<b>RXTHIEN</b>	<b>接收 FIFO 阈值中断使能位</b> 0 = RX FIFO 阈值中断禁止 1 = RX FIFO 阈值中断使能
[1]	<b>TXRST</b>	<b>发送复位</b> 0 = 无效 1 = 复位发送 FIFO 指针和发送电路。TXFULL 位将被清 0，TXEMPTY 位将被置 1。该位置 1 后，大约经过 3 个系统时钟周期和 2 个外设时钟周期，硬件会将该位清除为 0。如果用户想确认复位是否已经完成，可以读取 TXXRST (QSPIx_STATUS[23]) 来确认。 <b>注：</b> 如果从机模式下发生 TX 下溢事件发生，该位也可以用来使 QSPI 返回空闲状态。
[0]	<b>RXRST</b>	<b>接收复位</b> 0 = 无效 1 = 复位接收 FIFO 指针和接收电路。RXFULL 位将被清除为 0，RXEMPTY 位将被设置为 1。该位置 1 后，大约经过 3 个系统时钟周期和 2 个外设时钟周期，硬件将该位清除为 0。如果用户想确认复位是否已经完成，可以读 TXXRST (SPI_STATUS[23]) 来确认。

**QSPIx STATUS    QSPI 状态寄存器**

寄存器	偏移量	R/W	描述	复位值
QSPIx_STATUS	QSPIx_BA+0x14	R/W	QSPI 状态寄存器	0x0005_0110

注: I<sup>2</sup>S 模式不支持

31	30	29	28	27	26	25	24
TXCNT				RXCNT			
23	22	21	20	19	18	17	16
TXRXRST	Reserved			TXUFIF	TXTHIF	TXFULL	TXEMPTY
15	14	13	12	11	10	9	8
SPIENSTS	Reserved		RXTOIF	RXOVIF	RXTHIF	RXFULL	RXEMPTY
7	6	5	4	3	2	1	0
SLVURIF	SLVBEIF	SLVTOIF	SSLINE	SSINAIF	SSACTIF	UNITIF	BUSY

位	描述
[31:28]	<b>TXCNT</b> <b>发送 FIFO 数据个数 (只读)</b> 该域表明发送 FIFO 缓存的有效数据个数。
[27:24]	<b>RXCNT</b> <b>接收 FIFO 数据个数 (只读)</b> 该域表明接收 FIFO 缓存有效数据个数..
[23]	<b>TXRXRST</b> <b>TX 或 RX 复位状态 (只读)</b> 0 = TXRST 或 RXRST 的复位功能已经完成 1 = TXRST 或 RXRST 正在复位 <b>注意:</b> TXRST 和 RXRST 的复位操作都需要 3 个系统时钟周期 + 2 个外设时钟周期。用户可以检查该状态位来检测复位功能是否完成。
[22:20]	<b>Reserved</b> 保留
[19]	<b>TXUFIF</b> <b>TX 下溢中断标志</b> 当发送下溢事件发生时, 该位被设置为 1, 数据输出管脚的状态取决于 TXUFPOL 的设置。 0 = 无影响 1 = 当从机片选信号有效时, 发送 FIFO 和发送移位寄存器中没有数据。 <b>注 1:</b> 该位写 1 清 0。 <b>注 2:</b> 当从机片选信号有效时, 如果复位从机的发送电路, 在 3 个系统时钟周期和 2 个外设时钟周期后, 此时复位操作已经完成, 该位将被设置为 1。
[18]	<b>TXTHIF</b> <b>发送 FIFO 阈值中断标志 (只读)</b> 0 = 发送 FIFO 缓存中的有效数据个数大于 TXTH 设置的值 1 = 发送 FIFO 缓存中的有效数据个数少于或等于 TXTH 设置的值
[17]	<b>TXFULL</b> <b>发送 FIFO 缓存满标志 (只读)</b> 0 = 发送 FIFO 缓存未满 1 = 发送 FIFO 缓存已满

[16]	<b>TXEMPTY</b>	<b>发送 FIFO 缓存空标志 (只读)</b> 0 = 发送 FIFO 缓存不空 1 = 发送 FIFO 缓存已空
[15]	<b>SPIENSTS</b>	<b>QSPI 使能状态位 (只读)</b> 0 = 禁止 QSPI 控制器. 1 = 使能 QSPI 控制器. <b>注意:</b> QSPI 外设时钟与系统时钟不同步。为了确保 QSPI 控制器已经被禁止, 该位指示了 QSPI 控制器的真实状态。
[14:13]	<b>Reserved</b>	保留
[12]	<b>RXTOIF</b>	<b>接收超时中断标志</b> 0 = 没有接收 FIFO 超时事件 1 = 接收 FIFO 缓存非空且主机模式下, 超过 64 个 QSPI 时钟周期或从机模式下超过 576 个 QSPI 引擎时钟周期, 接收 FIFO 缓存上没有读操作。当接收 FIFO 缓存被软件读取, 超时状态会自动清 0。 <b>注:</b> 该位写 1 清 0
[11]	<b>RXOVIF</b>	<b>接收 FIFO 溢出中断标志</b> 当接收 FIFO 缓存已经满了, 接下来的数据将被丢弃, 同时该位被设置为 1. 0 = 接收 FIFO 没有溢出 1 = 接收 FIFO 溢出 <b>注:</b> 该位写 1 清 0
[10]	<b>RXTHIF</b>	<b>接收 FIFO 阀值中断标志 (只读)</b> 0 = 接收 FIFO 缓存的有效数据个数少于或等于 RXTH 设置的值 1 = 接收 FIFO 缓存的有效数据个数大于 RXTH 设置的值
[9]	<b>RXFULL</b>	<b>接收 FIFO 缓存满标志 (只读)</b> 0 = 接收 FIFO 缓存未满 1 = 接收 FIFO 缓存已满
[8]	<b>RXEMPTY</b>	<b>接收 FIFO 缓存空标志 (只读)</b> 0 = 接收 FIFO 缓存不空 1 = 接收 FIFO 缓存为空
[7]	<b>SLVURIF</b>	<b>从机模式发送下溢中断标志</b> 在从机模式, 如果发送下溢事件发生, 而且从机片选线进入无效状态, 该中断标志将被设置为 1。 0 = 未发生从机发送下溢事件 1 = 发生从机发送下溢事件 <b>注:</b> 该位写 1 清 0
[6]	<b>SLVBEIF</b>	<b>从机模式位计数错误中断标志</b> 在从机模式, 当从机片选信号线进入无效状态时, 如果位计数与 DWIDTH 不一致, 该中断标志将被设置为 1。 0 = 未发生从机模式位计数错误事件 1 = 发生从机模式位计数错误事件 <b>注:</b> 如果从机片选激活, 但是没有总线时钟输入, 当从机片选进入无效状态时, SLVBEIF 也将被设置为 1. 该位写 1 清 0。
[5]	<b>SLVTOIF</b>	<b>从机超时中断标志</b>

		当从机片选信号激活和 SLVTOCNT 的值不为 0 时，在检测到总线时钟时，QSPI 控制器逻辑的从机超时计数器开始计数。在事务完成之前，如果超时计数器的值大于或等于 SLVTOCNT (QSPIx_SSCTL[31:16]) 里的值。将发生从机超时中断事件。 0 = 从机超时未发生 1 = 发生从机超时 <b>注:</b> 该位写 1 清 0
[4]	<b>SSLINE</b>	从机片选线总线状态 (只读) 0 = 从机片选线状态为 0 1 = 从机片选线状态为 1 <b>注:</b> 该位只在从机模式有效。如果SSACTPOL (QSPIx_SSCTL[2]) 设置为 0，而且 SSLINE 的值为 1，QSPI 从机片选处于无效状态。
[3]	<b>SSINAIF</b>	从机片选无效中断标志 0 = 从机片选无效中断被清除或没有发生 1 = 发生了从机片选无效中断 <b>注:</b> 该位只在从机模式有效，该位写 1 清 0。
[2]	<b>SSACTIF</b>	从机片选激活中断标志 0 = 从机片选激活中断被清除或未发生 1 = 发生了从机片选激活中断 <b>注:</b> 该位只在从机模式有效，该位写 1 清 0。
[1]	<b>UNITIF</b>	单元传输中断标志 0 = 该位清除后，没有事务完成 1 = SPI 控制器已完成一个单元传输 <b>注:</b> 该位写 1 清 0。
[0]	<b>BUSY</b>	忙状态 (只读) 0 = QSPI 控制器处于空闲状态 1 = QSPI 控制器处于忙状态。 如下是忙条件： a. QSPIx_CTL[0] = 1 且 TXEMPTY = 0。 b. 对于 QSPI 主机模式，QSPIx_CTL[0] = 1 且 TXEMPTY = 1 但是当前传输尚未完成。 c. 对于 QSPI 主机模式，QSPIx_CTL[0] = 1 和 RXONLY = 1。 d. 对于 SPI 从机模式，SPIx_CTL[0] = 1，从机选择有效时仍有串口时钟输入。 e. 对于 SPI 从机模式，SPIx_CTL[0] = 1，从机选择无效时发送缓存或发送移位寄存器仍非空。

QSPIx\_TXQSPI 数据发送寄存器

寄存器	偏移量	R/W	描述	复位值
QSPIx_TX	QSPIx_BA+0x20	W	QSPI 数据发送寄存器	0x0000_0000

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

位	描述	
[31:0]	TX	<p><b>数据发送寄存器</b></p> <p>数据发送寄存器会把要发送的数据传入 4 级的发送 FIFO 缓存。该寄存器的有效位数在 QSPI 模式下取决于 DWIDTH (QSPIx_CTL[12:8])，在 I<sup>2</sup>S 模式下取决于 WDWIDHT (QSPIx_I2SCTL[5:4])。</p> <p>QSPI 模式下，如果 DWIDTH 为 0x08，TX[7:0] 位会被发送。如果 DWIDTH 为 0x00，QSPI 控制器将发送 32 位。</p> <p><b>注：</b>主机模式下，QSPI 控制器写该寄存器后，1 个 APB 时钟加 6 个外设时钟后才会开始 QSPI 总线时钟的传输。</p>

QSPIx\_RX      QSPI 数据接收寄存器

寄存器	偏移量	R/W	描述	复位值
QSPIx_RX	QSPIx_BA+0x30	R	QSPI 数据接收寄存器	0x0000_0000

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

位	描述	
[31:0]	RX	<p><b>数据接收寄存器</b></p> <p>SPI 控制器有 8 级 FIFO 缓存。从 QSPI 数据输入管脚接收到的数据会被保存到该寄存器。如果 RXEMPTY (QSPIx_STATUS[8] 不为 1, 软件可以通过读该寄存器获取 FIFO 缓存的数据。</p> <p>该寄存器只读。</p>

## 6.21 SPIM 同步串行接口控制器 (SPI 主机模式)

### 6.21.1 概述

SPI同步串行接口控制器在接收外围设备数据时执行串到并的转换，在收到MCU数据后执行并到串的转换。该控制器可以驱动一个外部设备（外部的 SPI Flash），扮演 SPI 主机的角色。当数据传输完成，它可以产生中断信号，并且可以通过写1来清除中断标志。从机片选的有效电平根据外设可以选择低或高电平有效。写一个分频值到SPIM\_CTL1寄存器设置输出到外围设备的串行输出时钟的频率。

SPI flash控制器普通I/O模式包含4个32位的发送/接收缓存，可以提供1到4个突发操作模式。每次传输的位数可以是8, 16, 24, 或 32位。一次传输中最大可以连续发送/接收四次数据。

通过DMA写模式，用户可以从SRAM传输数据到外部SPI flash器件。在DMA读模式下，可以将外部SPI flash器件数据传输到SRAM。在直接内存映射模式（DMM模式）下，在没有MCU设置相关SPI flash命令情况下，该SPI flash控制器会将AHB总线命令转换成SPI flash操作。因此用户可以将外部SPI flash当作一个ROM模块。

在关闭直接内存映射缓存的情况下，在控制器执行一个直接内存映射访问之后会预取4个字flash中的数据。在直接内存映射缓存开启模式下，它会使用32KB的缓存去减少访问外部SPI flash器件的次数，这样一来提高了SPI flash访问性能。在不增加串行时钟频率的情况下为了提高读SPI flash操作，这个SPI flash控制器支持DTR / DDR(双倍传输速率/双倍数据速率)读取命令指令，该指令支持标准/双/四数据SPI 模式。第一字节的命令指令类似于其他所有的SPI命令被设备锁存在上升沿时钟。一旦DTR / DDR指令被设备接受，地址输入和数据输出将锁定在串行时钟的上升和下降的边沿。

在内核耦合存储器模式（CCM模式）下，缓存功能被硬件自动禁用，MCU可以把32KB的缓存当作一般的SRAM用。对于数据保护，在DMA 读/写模式和直接内存映射模式下，SPI flash 控制器支持密码的加密和解密电路来保护那些被用户放在外部SPI flash的数据。

### 6.21.2 特性

- 最大支持32MB的SPI flash
- 支持SPI主机模式
- 支持直接内存映射模式和普通I/O模式
- 普通I/O模式下支持8/16/24/32位传输
- 普通I/O模式下支持突发模式操作，一次传输中最大可以连续发送/接收四次数据
- 支持DMA下读/写
- 支持标准的1位、2位和4位I/O传输模式
- 支持DTR / DDR(双倍传输速率/双倍数据速率)传输模式
- 支持32KB缓存
- 在缓存禁用情况下支持32KB内核耦合存储器模式（CCM）
- 支持密码的加/解密
- 对于外部SPI flash器件有一根从机/设备选择线

## 6.21.3 框图

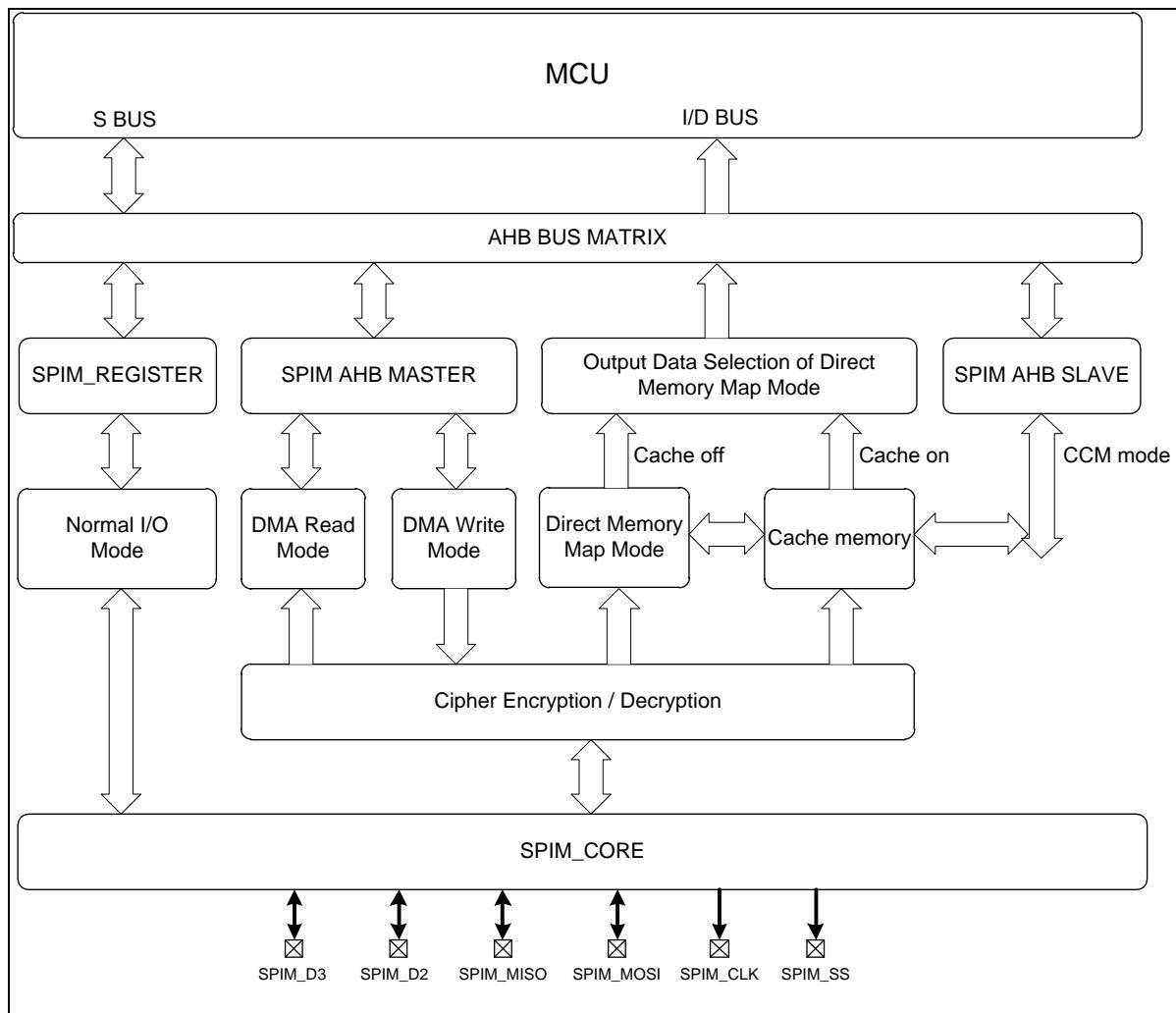


图 6.21-1 SPIM 方框图

#### 6.21.4 基本配置

- 时钟源配置
  - 使能SPIM外设时钟控制位是 SPIMCKEN (CLK\_AHBCLK[14])
- 复位配置
  - 复位SPIM控制器控制位是 SPIMRST (SYS\_IPRST0[14])中
- 引脚配置

组	引脚名称	GPIO	MFP
SPIM	SPIM_CLK	PA.2	MFP2
		PC.2	MFP3
		PE.4, PG.12	MFP4
	SPIM_D2	PA.5	MFP2
		PC.5	MFP3
		PE.7, PG.9	MFP4
	SPIM_D3	PA.4	MFP2
		PC.4	MFP3
		PE.6, PG.10	MFP4
	SPIM_MISO	PA.1	MFP2
		PC.1	MFP3
		PE.3, PG.13	MFP4
	SPIM_MOSI	PA.0	MFP2
		PC.0	MFP3
		PE.2, PG.14	MFP4
	SPIM_SS	PA.3	MFP2
		PC.3	MFP3
		PE.5, PG.11	MFP4

### 6.21.5 功能描述

#### 6.21.5.1 SPIM时序图

SPI传输时序图如下所示：

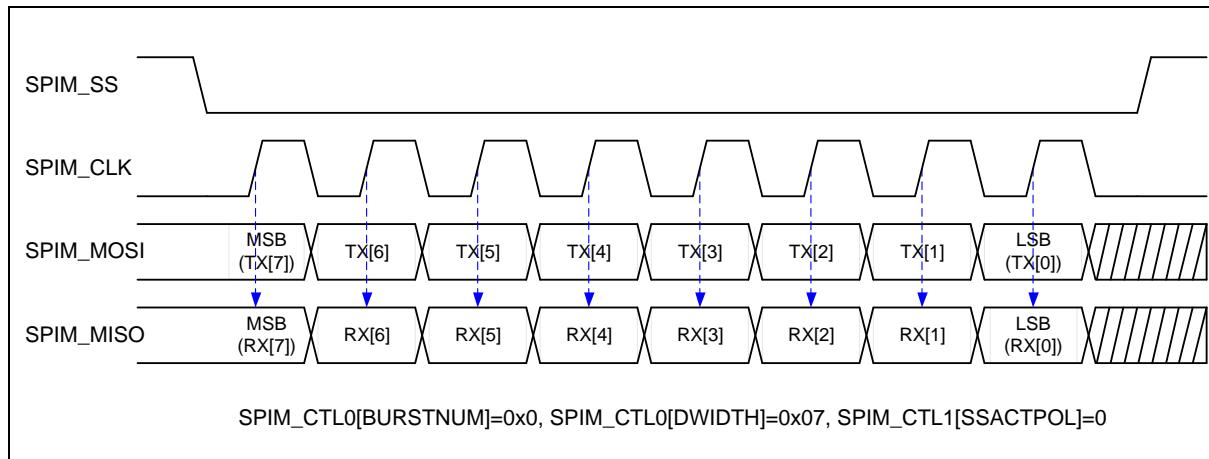


图 6.21-2 SPIM 时序图

#### 6.21.5.2 SPIM 不带DMA的编程范例(普通 I/O 模式)

依据如下的要求访问设备：

- 数据高位先传
- 每次发送/接收只有一个字节
- 片选信号低有效
- 你应该做如下基本设置（具体步骤请参考设备的规格）：
  1. 写一个分频值到 DIVIDER (SPIM\_CTL1[31:16]) 以确定串行时钟的频率。
  2. 设置 SSACTPOL (SPIM\_CTL1[5]) 为 0 来激活你想要访问的设备。
  3. 当发送 (写) 数据到设备：
    - QDIODIR (SPIM\_CTL0[15]) = 1
    - BURSTNUM (SPIM\_CTL0[14:13]) = 0x0
    - DWIDTH (SPIM\_CTL0[12:8]) = 0x07
    - 写你要发送的数据到 SPIM\_TX0[7:0]
  4. 当从设备接收 (读) 数据：
    - QDIODIR (SPIM\_CTL0[15]) = 0
    - BURSTNUM (SPIM\_CTL0[14:13]) = 0x0
    - DWIDTH (SPIM\_CTL0[12:8]) = 0x07
  5. 设置 SPIMEN (SPIM\_CTL1[0]) 为 1，开始传输。
  6. 等待中断或轮询 SPIMEN (SPIM\_CTL1[0]) 直到它变为 0
  7. 当接收 (读) 来自设备的数据：
    - 从 SPIM\_RX0[7:0] 寄存器读出接收到的数据

### 6.21.5.3 SPIM 带DMA的编程范例

如果用户想用 DMA 的功能访问设备，三个额外的寄存器需要被配置，包括 **SPIM\_DMACNT**, **SPIM\_SRAMADDR** 和 **SPIM\_FADDR**。DMA 功能可以被用来支持加载启动代码（从外设读数据到系统内存）或从系统内存读数据保存到外设。用户必须定义源地址、长度和目的地址，然后硬件会自动搬运想要的数据长度到指定的目的地址。用户在 **SRAM** 和 **SPI flash** 之间使用 DMA 读/写操作之前，强烈建议向外部 **SPI flash** 发送禁用连续读模式和突发包模式的命令。

### 6.21.5.4 从 SPI flash 搬运数据到系统内存(DMA 读模式)

步骤 1:通过使用普通I/O模式检查设备ID确认设备已连接

步骤 2:

- 在 **SPIM\_SRAMADDR** 寄存器设置目标内存地址
- 在 **SPIM\_DMACNT** 寄存器设置代码的长度
- 在 **SPIM\_FADDR** 寄存器设置 SPI Flash 的起始地址
- 设置 **OPMODE** (**SPIM\_CTL0[23:22]**) 为 DMA 读模式
- 设置 **CMDCODE** (**SPIM\_CTL0[31:24]**) 为 0x03 命令码
- 设置 **SPIMEN** (**SPIM\_CTL1[0]**) 启动，然后 SPIM 会从 **SPIM\_FADDR** 到 **SPIM\_SRAMADDR** 搬运 **SPIM\_DMACNT** 中设置的传输长度的代码块.
- 等待中断或轮询 **SPIMEN** (**SPIM\_CTL1[0]**) 位直到它变为 0

如果使用的 **SPI Flash** 支持其他相关的读命令，用户也可以使用下面的读命令代码

- 快速读 (0Bh), 设置 **CMDCODE** (**SPIM\_CTL0[31:24]**) 为 0x0B.
- 快速读双线输出 (3Bh), 设置 **CMDCODE** (**SPIM\_CTL0[31:24]**) 为 0x3B.
- 快速读双线I/O (BBh), 设置 **CMDCODE** (**SPIM\_CTL0[31:24]**) 为 0xBB.
- 快速读四线I/O (EBh), 设置 **CMDCODE** (**SPIM\_CTL0[31:24]**) 为 0xEB.
- 字读双线I/O (E7h), 设置 **CMDCODE** (**SPIM\_CTL0[31:24]**) 为 0xE7.
- DTR/DDR 快速读(0Dh), 设置 **CMDCODE** (**SPIM\_CTL0[31:24]**) 为 0x0D.
- DTR/DDR 快速读双线 I/O (BDh), 设置 **CMDCODE** (**SPIM\_CTL0[31:24]**) 为 0xBD.
- DTR/DDR 快速读四线 I/O (EDh), 设置 **CMDCODE** (**SPIM\_CTL0[31:24]**) 为 0xED.

### 6.21.5.5 从系统内存搬运数据到SPI flash (DMA 写模式)

步骤 1:写 SPI Flash 之前先擦除 SPI Flash (使用普通I/O模式)

步骤 2:

- 发送写使能命令到 SPI Flash (使用普通I/O 模式).
- 在 **SPIM\_SRAMADDR** 寄存器里设置源内存地址
- 在 **SPIM\_DMACNT** 寄存器里设置传输数据个数

- 在SPIM\_FADDR寄存器里设置 SPI Flash起始地址
- 设置 OPMODE (SPIM\_CTL0[23:22])为 DMA 写模式
- 设置 CMDCODE (SPIM\_CTL0[31:24])为 0x02 命令代码
- 设置 SPIMEN (SPIM\_CTL1[0])启动
- 等待中断或轮询 SPIMEN (SPIM\_CTL1[0])直到它变为 0.

如果使用的SPI Flash 支持其他相关的写命令，用户也可以使用下面命令代码（用户需要参考SPI Flash的规格来选择合适的四线写模式命令）

- 四线写, 设置CMDCODE (SPIM\_CTL0[31:24]) 为0x32用于 类型\_1编程命令 (参考图 6.21-3).
- 四线写, 设置CMDCODE (SPIM\_CTL0[31:24]) 为0x38用于 类型\_2编程命令 (参考图 6.21-4).
- 四线写, 设置CMDCODE (SPIM\_CTL0[31:24]) 为0x40用于 类型\_3编程命令 (参考图 6.21-5).

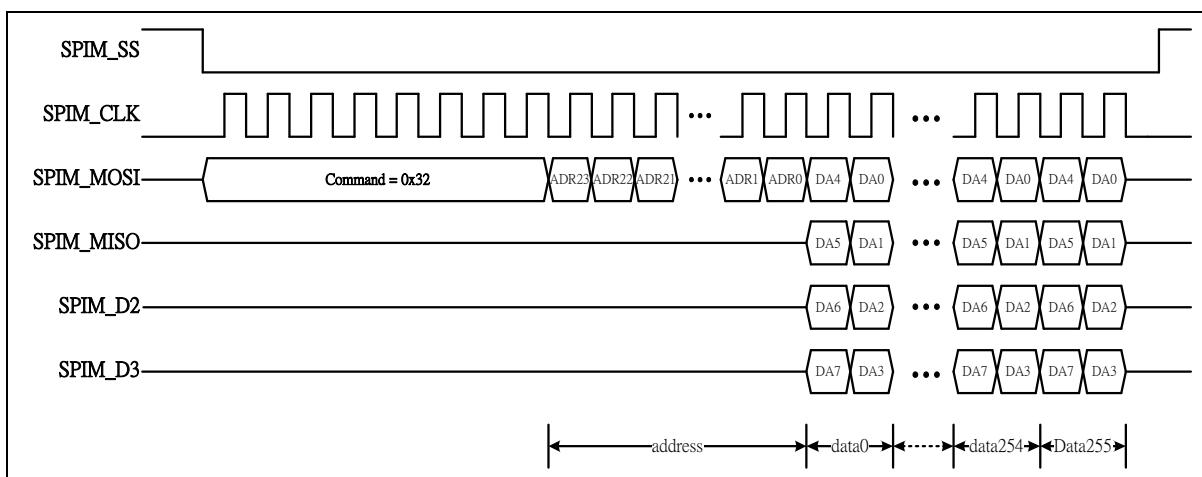


图 6.21-3 类型\_1 四线模式写的编程命令模式

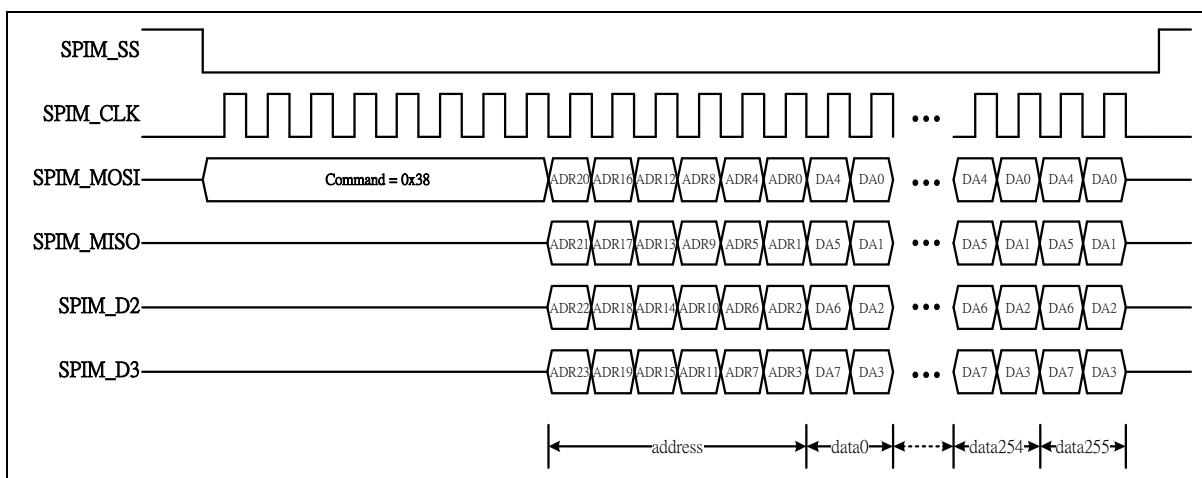


图 6.21-4 类型\_2 四线模式写的编程命令模式

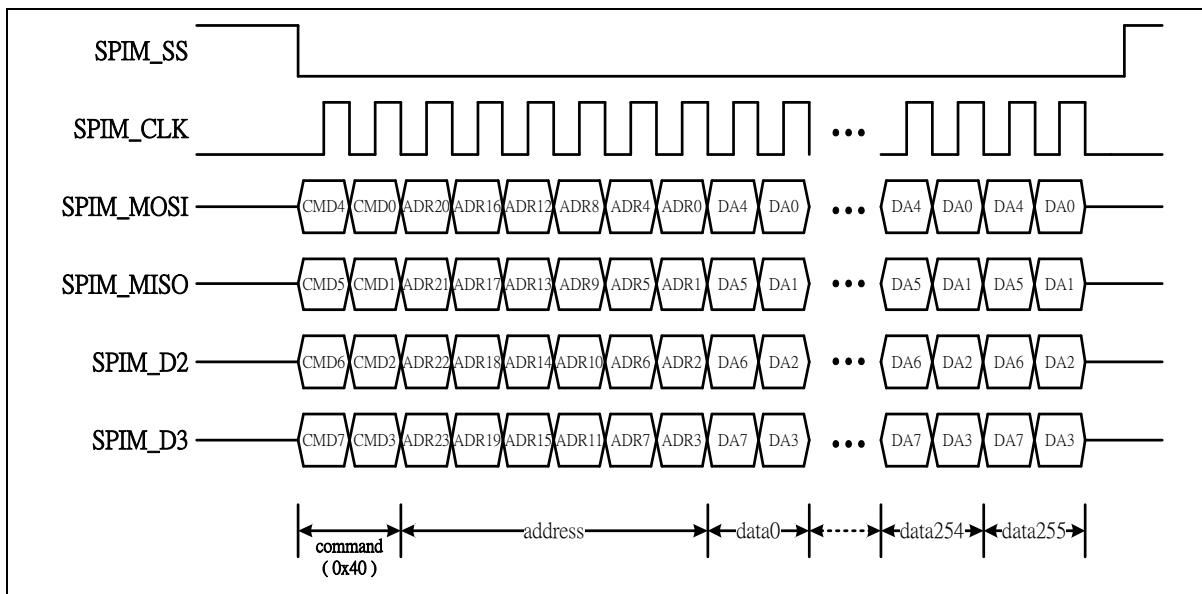


图 6.21-5 类型\_3 四线模式写的编程命令模式

### 6.21.5.6 直接内存映射模式

在直接内存映射模式下，SPI Flash可以被当作一个ROM模块。控制器会通过AHB访问SPI Flash，不需要MCU设置相关的SPI Flash命令。MCU唯一需要做的是设置CMDCODE(SPI\_M\_CTL0[31:24])为0x03, 0x0B, 0x3B, 0xBB, 0xEB, 0xE7, 0x0D, 0xBD,或0xED.命令代码，然后用户就可以像一个ROM模块一样访问SPI Flash。

在直接内存映射缓存关闭模式下，一个直接内存映射访问后，会预先取四个字的Flash数据。在直接内存映射模式，带有缓存，并且内核耦合存储器模式（CCM）关闭的情况下，将会使用缓存内存来减少对外部SPI flash器件的访问次数。因此对SPI flash访问可以改善。如果用户想要在直接内存映射访问之后修改SPI硬件寄存器，需要等至少250个外设时钟周期(SPI总线周期)。

在直接内存映射模式(DMM模式)下，双/四线读命令码“0xEB, 0xE7, 0xBB”和DTR/DDR 读命令码“0x0D, 0xBD, 0xED”支持连续读模式(或性能增强模式)这样可以减少8个命令时序时钟，在SPI flash器件片选拉低后读地址数据被允许进入。(用户需要参考SPI Flash的规格书了解支持的信息)。

在图 6.21-6中展示了在连续读模式禁用情况下地址模式是3字节的时候四线I/O模式SPI 数据传输。

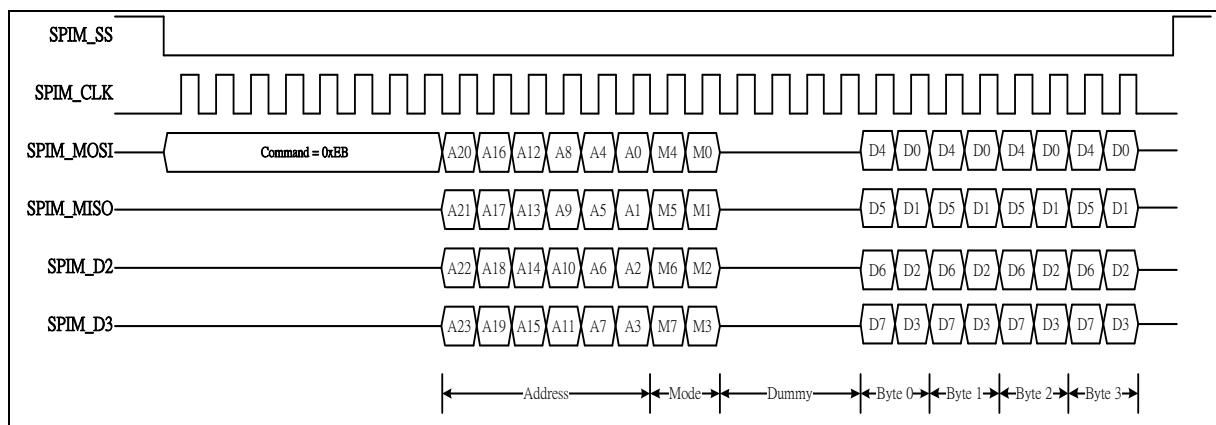


图 6.21-6 连续读模式被禁用下命令为 0xEB 的快速读四线 I/O

在图 6.21-7 中展示了在连续读模式使能情况下地址模式是3字节的时候四线 I/O 模式 SPI 数据传输。

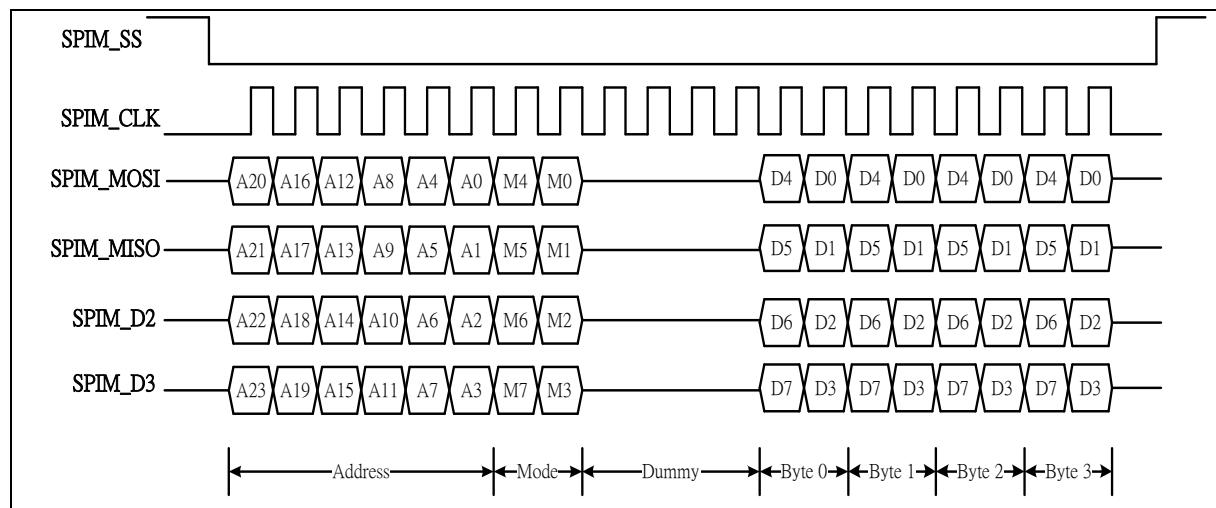


图 6.21-7 连续读模式使能下命令为 0xEB 的快速读四线

以下列出了在直接内存映射模式下 SPI flash 厂商连续读模式所能支持的读命令 0xBB, 0xEB, 0xE7, 0x0D, 0xBD, 和 0xED:

- 支持读命令 0xBB 的 SPI flash 厂商: 仅 Winbond
- 支持读命令 0xEB 的 SPI flash 厂商: Winbond, Gigadevice, Spansion, EON, 和 MXIC
- 支持读命令 0xE7 的 SPI flash 厂商: 仅 Winbond
- 支持 DTR/DDR 读命令 0x0D 的 SPI flash 厂商: 请检查所使用的 SPI flash 规格书
- 支持 DTR/DDR 双线 I/O 读命令 0xBD 的 SPI flash 厂商: 请检查所使用的 SPI flash 规格书
- 支持 DTR/DDR 四线 I/O 读命令 0xED 的 SPI flash 厂商: 请参考所使用的 SPI flash 规格书

在 DMM 模式连续读模式开启情况下, 用户在使用命令码 0xEB, 0xE7, 0x0D, 0xBD, 和 0xED 去读完 SPI flash 中最后所需的数据后, 用户需要使用普通 I/O 模式发送 0xFF(3 字节地址模式模式下 8 个 SPI 时钟)或 0x3FF(4 字节地址模式 10 个 SPI 时钟)到外部 SPI flash 中, 用 SPI 串口接口引脚 sdo\_0,

这样来禁用外部 SPI flash 连续读模式。了解连续读模式（或性能增强模式）的信息请参考所使用 SPI flash 规格书。

在 DMM 模式连续读模式开启情况下，用户在使用命令码“0xBB”去读完 SPI flash 中最后所需的数据后，用户需要使用普通 I/O 模式发送 0xFFFF(3 字节地址模式模式下 16 个 SPI 时钟)或 0xFFFF (4 字节地址模式 20 个 SPI 时钟)到外部 SPI flash 中，用 SPI 串口接口引脚 sdo\_0。了解连续读模式（或性能增强模式）的信息请参考所使用 SPI flash 规格书。

用户可以使用下面普通 I/O 模式的设置步骤来发送连续读模式复位到外部 SPI flash。想了解连续读模式（或性能增强模式）的信息请参考所使用 SPI flash 规格书。

- 在直接内存映射模式下对于命令码**0xEB, 0xE7, 0x0D, 0xBD, 0xED**连续读模式复位（禁用）的步骤：
  - 设置 OPMODE(SPIM\_CTL0[23:22])为0x0 (普通 I/O 模式)
  - 设置BITMODE(SPIM\_CTL0[21:20]) 为0x2 (四线模式, 使用4个串行SPI信号发送数据到SPI flash)
  - 设置QDIODIR(SPIM\_CTL0[15]) 为0x1 (写模式)
  - 设置BURSTNUM(SPIM\_CTL0[14:13]) 为0x0 (在一次传输中只有一次发送/接收数据交换)
  - 设置DWIDTH(SPIM\_CTL0[12:8]) 为0x1F (发送 32位数据到SPI flash)
  - 设置SPIM\_TX0 寄存器为0xFFFFFFFF
  - 置位SPIMEN(SPIM\_CTL1[0]) 启动普通 I/O 操作
  - 在普通I/O模式启动后，轮询 SPIMEN(SPIM\_CTL1[0]) 检查运行状态 (1 = 忙, 0 = 完成)
  - 设置SS(SPIM\_CTL1[4]) 让 SPIM\_SS 处在非激活电平
- 在直接内存映射模式下对于命令码**0xBB**连续读模式复位的步骤：
  - 设置OPMODE(SPIM\_CTL0[23:22]) 为 0x0 (普通 I/O 模式)
  - 设置BITMODE(SPIM\_CTL0[21:20]) 为 0x1 (双线模式, 使用2个串行SPI信号发送数据到SPI flash)
  - 设置QDIODIR(SPIM\_CTL0[15]) 为 0x1 (写模式)
  - 设置BURSTNUM(SPIM\_CTL0[14:13]) 为 0x0 (在一次传输中只有一次发送/接收数据交换)
  - 设置DWIDTH(SPIM\_CTL0[12:8]) 为 0x1F (发送 32位数据到SPI flash)
  - 设置SPIM\_TX0 寄存器为0xFFFFFFFF
  - 置位SPIMEN(SPIM\_CTL1[0]) 启动普通 I/O 操作
  - 在普通I/O模式启动后，轮询 SPIMEN(SPIM\_CTL1[0]) 检查运行状态 (1 = 忙, 0 = 完成)
  - 设置SS(SPIM\_CTL1[4]) 让 SPIM\_SS 处在非激活电平
- 在直接内存映射模式下，四线读命令“0xEB” 和 “0xE7”支持突发包模式的缓存应用和连续读模式（或性能增强模式）。对于缓存应用，突发包模式可以快速地被用来填充缓存线(在该SPI flash控制器下，我们使用缓存数据线大小是16个字节)。对于直接内存映射模式和缓存使能条件下的连续读模式(或性能增强模式)，如果缓存没有命中，突发包模式可以快速的让MCU获得SPI flash数据请

求。在突发包模式和连续读模式要从使能状态切换到禁用状态时，首先需要禁止外部SPI Flash的连续读模式，然后禁止外部SPI Flash的突发包模式。

在用户使用 DMM 模式下读命令码“0xEB 或 0xE7”和突发包模式访问外部 flash 之前，请使用 SPI flash 控制器的普通 I/O 模式通过 SPI 的 sdo\_0 ~ sdo\_3 串行接口管脚发送“0x1101\_1101 和 0x2000\_0000”(外部 SPI flash 突发包使能命令需要 16 个 SPI 时钟，请参考使用的 SPI Flash 的规格书了解突发包模式的配置)到外部 SPI flash。在用户使用 DMM 模式下读命令码“0xEB 和 0xE7”配合突发包模式读完外部 flash 最后需要的数据后，请使用 SPI flash 控制器的普通 I/O 模式通过 SPI 的 sdo\_0 ~ sdo\_3 串行接口管脚发送“0x1101\_1101 和 0x0000\_0000”(外部 SPI flash 突发包禁用命令需要 16 个 SPI 时钟，请参考使用的 SPI Flash 的规格书了解突发包模式的配置)到外部 SPI flash。

用户可以使用下面 SPI flash 控制器普通 I/O 模式的设置步骤来发送突发包模式使能或突发包模式禁用的 SPI 命令码到外部 SPI flash。

- 直接内存映射模式下使能突发包模式的步骤:

- 设置OPMODE(SPIM\_CTL0[23:22]) 为 0x0 (普通I/O)
- 设置BITMODE(SPIM\_CTL0[21:20]) 为 0x2 (四线模式, 使用4个串行SPI信号发送数据到SPI flash)
- 设置QDIODIR(SPIM\_CTL0[15]) 为0x1 (写模式)
- 设置BURSTNUM(SPIM\_CTL0[14:13]) 为0x1 (一次传输中有两次发送/接收数据交换)
- 设置DWIDTH(SPIM\_CTL0[12:8]) 为0x1F (发送 32位数据到SPI flash)
- 设置SPIM\_TX0 寄存器为 0x20000000, SPIM\_TX1 寄存器为0x11011101
- 设置SPIMEN(SPIM\_CTL1[0]) 启动普通 I/O 操作
- 在普通I/O模式启动后, 轮询 SPIMEN(SPIM\_CTL1[0]) 检查运行状态 (1 = 忙, 0 = 完成)
- 设置SS(SPIM\_CTL1[4]) 让 SPIM\_SS 处在非激活电平

- 直接内存映射模式下禁用突发包模式的步骤:

- 设置OPMODE(SPIM\_CTL0[23:22]) 为0x0 (普通I/O)
- 设置BITMODE(SPIM\_CTL0[21:20]) 为0x2 (四线模式, 使用4个串行SPI信号发送数据到SPI flash)
- 设置QDIODIR(SPIM\_CTL0[15]) 为0x1 (写模式)
- 设置BURSTNUM(SPIM\_CTL0[14:13]) 为0x1 (一次传输中有两次发送/接收数据交换)
- 设置DWIDTH(SPIM\_CTL0[12:8]) 为0x1F (发送 32位数据到SPI flash)
- 设置SPIM\_TX0寄存器为0x00000000, SPIM\_TX1寄存器为0x11011101
- 设置SPIMEN(SPIM\_CTL1[0]) 启动普通 I/O 操作
- 在普通I/O模式启动后, 轮询 SPIMEN(SPIM\_CTL1[0]) 检查运行状态 (1 = 忙, 0 = 完成)
- 设置SS(SPIM\_CTL1[4]) 让 SPIM\_SS 处在非激活电平

为了在不增加串行时钟频率下提高读操作的吞吐量, 该SPI flash控制器支持DTR/DDR (双倍传输速率/双倍数据速率)读命令码(i.e. 0x0D, 0xBD, 0xED), 该命令码在直接内存映射模式和DMA读模式下支持标准/双线/四线SPI模式。一个字节的命令码依然是8个上升沿串行时钟被锁存到设备。一旦一条DTR/DDR指令码被设备接受, SPI flash读地址、模式位、空闲周期 (*dummy cycle*) 和输出数据, 这些从SPI flash器件发送出数据会被在上升沿和下降沿锁存。下面为我们展示了DTR/DDR 读命令 0x0D, 0xBD, 和 0xED的时序图。

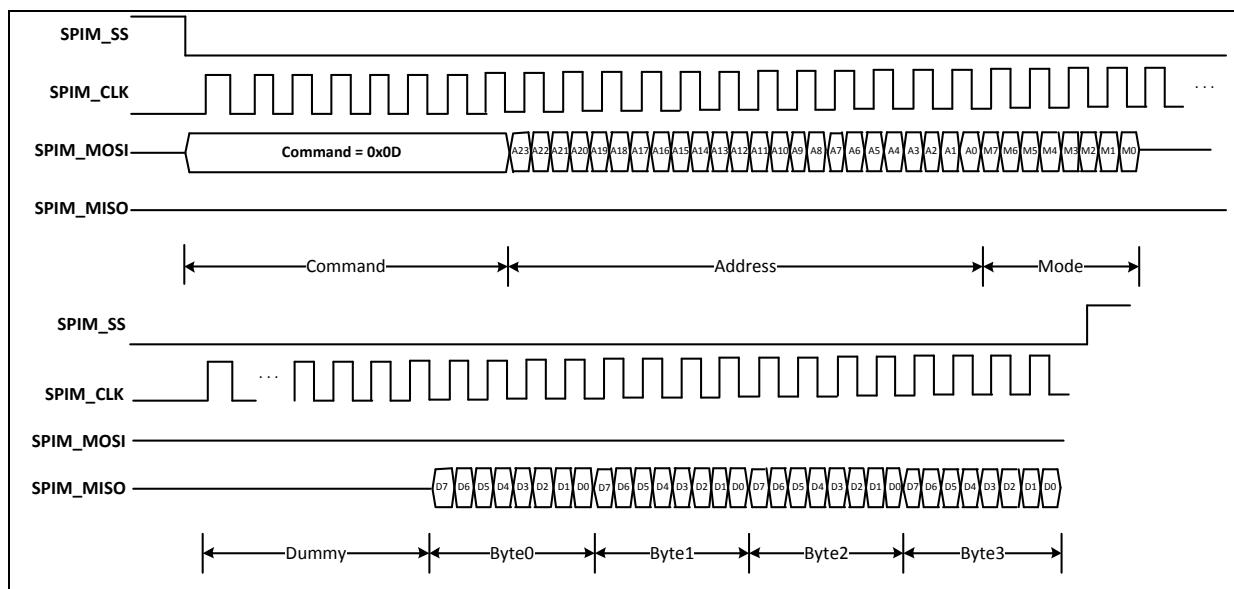


图 6.21-8 DTR/DDR 快速读命令 0x0D

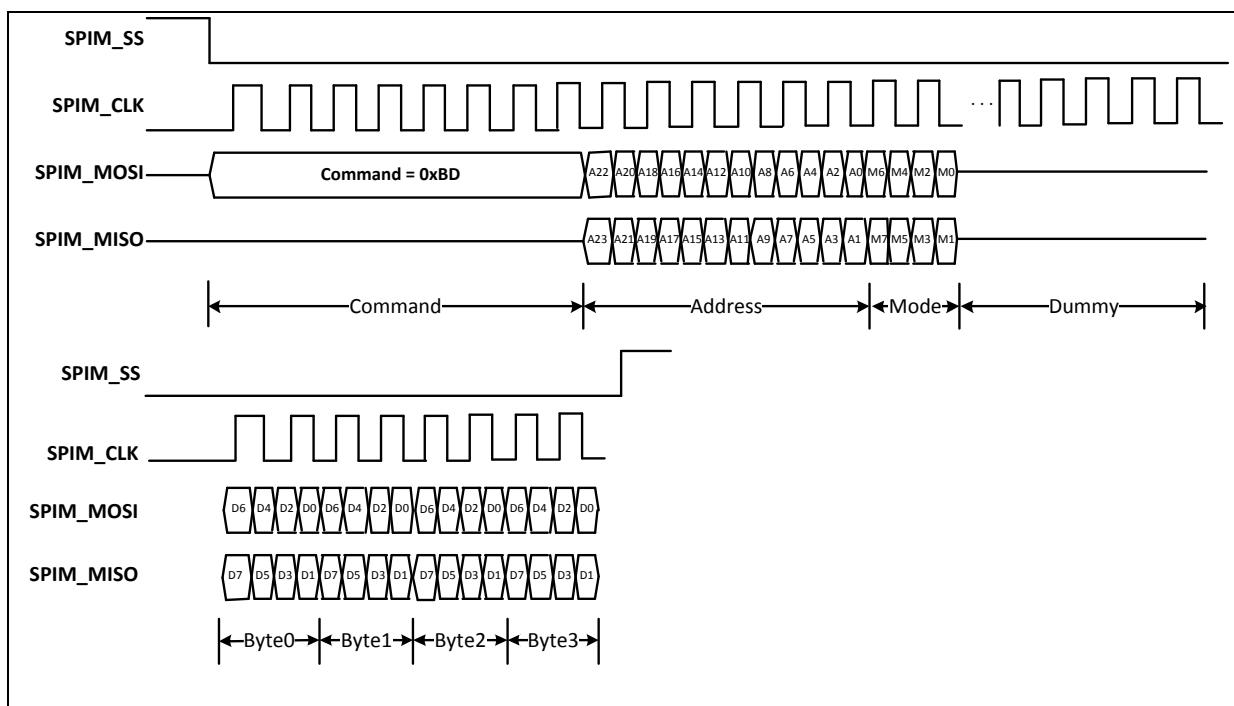


图 6.21-9 DTR/DDR 快速读双线 I/O 命令 0xBD

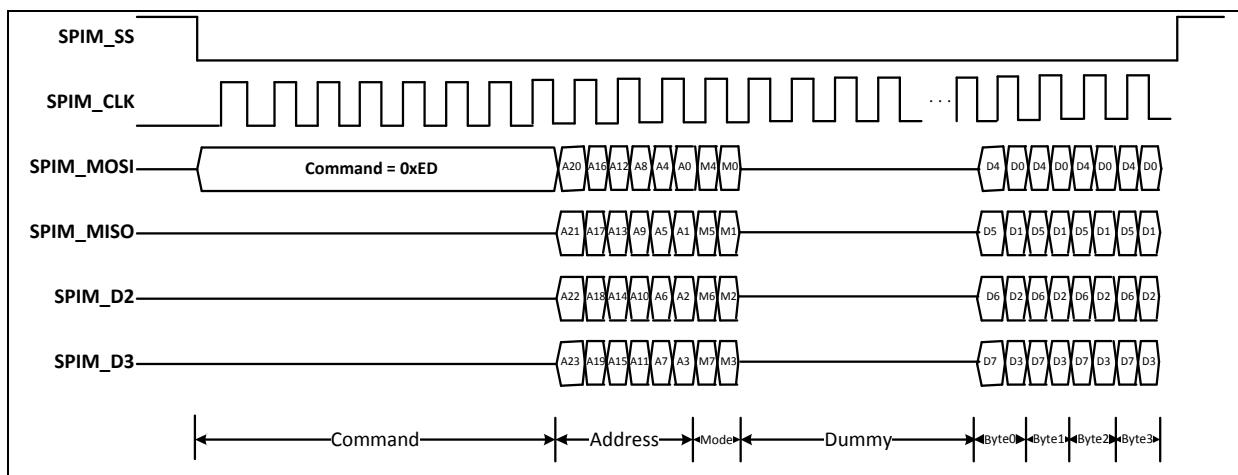


图 6.21-10 DTR/DDR 快速读四线 I/O 命令 0xED

空闲周期（dummy cycle）数取决于SPI flash 厂商、操作频率和命令类型，请参考所使用的SPI flash说明书了解准确的空闲周期（dummy cycle）数。

#### 6.21.5.7 内核耦合存储器模式（CCM模式）

当用户设置CCMEN(SPIM\_CTL1[2])为1时，缓存存储器操作模式会转变成内核耦合存储器模式（CCM模式）。在CCM模式下，SPI flash控制器的缓存功能会被硬件自动禁用，MCU访问内核耦合存储器就像是SRAM一样。当用户设置CCMEN(SPIM\_CTL1[2])为0时MCU访问CCM地址范围，SPI flash控制器将会产生AHB错误给MCU。

#### 6.21.5.8 密码加密和解密

SPI flash控制器支持密码加密和解密功能，该功能可以用来保护用户放在外部SPI flash的数据。当用户使用DMA写模式并且密码功能使能时，密码电路会加密从SRAM传输到外部SPI flash的数据。当用户使用DMA读模式并且密码功能使能时，密码电路会解密从外部SPI flash传输到SRAM的数据。当用户使用直接内存映射模式并且密码功能使能时，密码电路会解密MCU从外部SPI flash读到的数据。在普通I/O模式，发送到外部SPI flash或接收外部SPI flash的数据都不会用到密码电路进行加密和解密。

## 6.21.6 SPI flash控制器在AMBA系统中映射地址

地址空间	标记	控制器
<b>SPIM控制器映射地址, SPI flash内存空间, 内核耦合存储空间</b>		
0x4000_7000 – 0x4000_7FFF	SPIM_BA	SPIM 控制寄存器
(1) 当芯片 SRAM 大小为 96KB : 0x1001_0000 – 0x1001_7FFF 0x2001_0000 – 0x2001_7FFF		仅对MCU的SRAM内存空间(32KB)和SPIM缓存共享(当CCM模式使能)
(2) 当芯片 SRAM 大小为128KB : 0x1001_8000 – 0x1001_FFFF 0x2001_8000 – 0x2001_FFFF	SRAM2_BA	注：芯片可以配置成三种大小的SRAM(160KB/128KB/96KB). 在CCM模式下请检查SRAM的配置来确认MCU可以访问的地址范围。
(3) 当芯片 SRAM 大小为160KB : 0x1002_0000 – 0x1002_7FFF 0x2002_0000 – 0x2002_7FFF		
0x0800_0000 – 0x09FF_FFFF	SPIM_FLASH_BA	对于直接内存映射模式(DMM模式), SPIM内存空间(32MB)可以访问地址范围

表 6.21-1 SPI flash 控制器在 AMBA 系统中映射地址

## 6.21.7 寄存器映射

R: 只读, W: 只写, R/W: 可读可写, C: 只能写0

寄存器	偏移地址	R/W	描述	复位值
<b>SPIM 基址:</b>				
<b>SPIM_BA = 0x4000_7000</b>				
<b>SPIM_CTL0</b>	SPIM_BA+0x00	R/W	控制和状态寄存器 0	0x00C0_0002
<b>SPIM_CTL1</b>	SPIM_BA+0x04	R/W	控制寄存器 1	0x0000_0010
<b>SPIM_RXCLKDLY</b>	SPIM_BA+0x0C	R/W	RX 时钟延时控制寄存器	0x0000_010F
<b>SPIM_RX0</b>	SPIM_BA+0x10	R	数据接收寄存器0	0x0000_0000
<b>SPIM_RX1</b>	SPIM_BA+0x14	R	数据接收寄存器1	0x0000_0000
<b>SPIM_RX2</b>	SPIM_BA+0x18	R	数据接收寄存器2	0x0000_0000
<b>SPIM_RX3</b>	SPIM_BA+0x1C	R	数据接收寄存器3	0x0000_0000
<b>SPIM_TX0</b>	SPIM_BA+0x20	R/W	数据发送寄存器 0	0x0000_0000
<b>SPIM_TX1</b>	SPIM_BA+0x24	R/W	数据发送寄存器1	0x0000_0000
<b>SPIM_TX2</b>	SPIM_BA+0x28	R/W	数据发送寄存器2	0x0000_0000
<b>SPIM_TX3</b>	SPIM_BA+0x2C	R/W	数据发送寄存器3	0x0000_0000
<b>SPIM_SRAMADDR</b>	SPIM_BA+0x30	R/W	SRAM 内存地址寄存器	0x0000_0000
<b>SPIM_DMACNT</b>	SPIM_BA+0x34	R/W	DMA 传输字节计数寄存器	0x0000_0000
<b>SPIM_FADDR</b>	SPIM_BA+0x38	R/W	SPI Flash 地址寄存器	0x0000_0000
<b>SPIM_KEY1</b>	SPIM_BA+0x3C	W	密钥1 寄存器	0x0000_0000
<b>SPIM_KEY2</b>	SPIM_BA+0x40	W	密钥2 寄存器	0x0000_0000
<b>SPIM_DMMCTL</b>	SPIM_BA+0x44	R/W	直接内存映射模式控制寄存器	0x0008_0000
<b>SPIM_CTL2</b>	SPIM_BA+0x48	R/W	控制寄存器 2	0x0804_0000

注: 如果软件想写该外设的任何寄存器, SPIM\_CTL1 寄存器的 SPIMEN 位应该要为低状态

### 6.21.8 寄存器描述

#### SPIM\_CTL0 控制和状态寄存器0

寄存器	偏移地址	R/W	描述	复位值
SPIM_CTL0	SPIM_BA+0x00	R/W	控制和状态寄存器0	0x00C0_0002

31	30	29	28	27	26	25	24
CMDCODE							
23	22	21	20	19	18	17	16
OPMODE		BITMODE		SUSPITV			
15	14	13	12	11	10	9	8
QDIODIR	BURSTNUM		DWIDTH				
7	6	5	4	3	2	1	0
IF	IEN	B4ADDREN	Reserved		BALEN	Reserved	CIPHOFF

位	描述
[31:24]	<p><b>CMDCODE</b></p> <p>页编程命令码 (注 4)</p> <ul style="list-style-type: none"> <li>(1) 0x02 = 页编程 (用于 DMA 写模式).</li> <li>(2) 0x32 = 用类型_1编程流程的四线页编程(用于 DMA 写模式). (注3)</li> <li>(3) 0x38 = 用类型_2编程流程的四线页编程(用于 DMA 写模式). (注3)</li> <li>0x40 = 用类型_3 编程流程的四线页编程(用于 DMA 写模式). (注 3)</li> </ul> <p>其它 = 保留.</p> <p>读命令码 :</p> <ul style="list-style-type: none"> <li>(1) 0x03 = 标准读(用于 DMA 读/DMM 模式).</li> <li>(2) 0x0B = 快速读 (用于 DMA 读/DMM 模式).</li> </ul> <p>快速读命令码“0x0B”类似于标准读命令码“0x03”， 除此之外它还可以工作在最高可能的频率 (注 2)</p> <ul style="list-style-type: none"> <li>(3) 0x3B = 快速读双线输出 (用于 DMA 读/DMM 模式).</li> <li>(4) 0xBB = 快速读双线 I/O (用于 DMA 读/DMM 模式).</li> </ul> <p>快速读双线 I/O 命令码“0xBB”类似于快速读双线输出命令码“0x3B”， 但是它还具有在每个时钟下可以输入 2 位的地址位的能力 (注 2)</p> <ul style="list-style-type: none"> <li>(5) 0xEB = 快速四线读 (用于 DMA 读/DMM 模式).</li> <li>(6) 0xE7 = 字四线读 (用于 DMA 读/DMM 模式).</li> </ul> <p>字四线读命令码“0xE7”类似于快速四线读命令码“0xEB”， 除此之外它的最低地址位必须为 0 并且空闲周期数要比快速四线读少 (注 2)</p> <ul style="list-style-type: none"> <li>(7) 0xD = DTR/DDR 快速读 (用于 DMA 读/DMM 模式).</li> <li>(8) 0xBD = DTR/DDR 双线读 (用于 DMA 读/DMM 模式).</li> <li>(9) 0xED = DTR/DDR 四线读 (用于 DMA 读/DMM 模式).</li> </ul> <p>其它命令码保留</p> <p>DTR/DDR 读命令 0xD,0xBD,0xED”通过在 SPI flash 时钟(SPI_MCLK)的上升沿和下降沿传输地址和数据来提高吞吐量。它类似于“0x0B, 0xBB, 0xEB”这些命令码，但是它可以在 SPI flash 输出时钟的上升沿和下降沿都能够传输地址和数据。 (注 2)</p>

		<p><b>注 1:</b>在使用四线页编程/四线读命令之前，需要先通过普通 I/O 模式使能 SPI flash 的四线模式。</p> <p><b>注 2:</b>参考 SPI flash 规格书所支持的命令码</p> <p><b>注 3:</b>对于页编程命令码的 类型_1, 类型_2, 和类型_3, 请参考 图 6.21-3, 图 6.21-4, 和 图 6.21-5.</p> <p><b>注 4:</b>在使用 SPI flash 控制器的 DMA 写模式去编程外部 SPI flash 之前, 请禁用“连续读模式”和“突发包模式”。在使用 SPI flash 控制器 DMA 写模式去编程外部 SPI flash 的内容之后, 请设置 CDINVAL(SPIM_CTL1[3])为 1 (设置所有缓存数据无效)</p>
[23:22]	<b>OPMODE</b>	<p><b>SPI 功能操作模式</b></p> <p>0x0 = 普通 I/O 模式. <b>(注 1) (注 3)</b></p> <p>0x1 = DMA 写模式. <b>(注 2) (注 3)</b></p> <p>0x2 = DMA 读模式. <b>(注 3)</b></p> <p>0x3 = 直接内存映射模式(DMM 模式) (默认). <b>(注 4)</b></p> <p><b>注 1 :</b>在使用 SPI flash 控制器普通 I/O 模式去编程外部 SPI flash 内容之后, 请设置 CDINVAL(SPIM_CTL1[3])为 1 (设置所有缓存数据无效)</p> <p><b>注 2 :</b>在 DMA 模式下, 每次操作硬件会发送一页编程编程命令。用户必须要小心处理好跨页的情况。在使用 SPI flash 控制器 DMA 写模式去编程外部 SPI flash 的内容之后, 请设置 CDINVAL(SPIM_CTL1[3])为 1 (设置所有缓存数据无效)</p> <p><b>注 3 :</b>对于 32MB 的外部 SPI flash, 在使用普通 I/O 模式、DMA 写模式和 DMA 读模式去写/读该外部 SPI flash 时, 访问地址范围是 0x00000000 to 0x01FFFFFF。请用户核对好使用的 SPI flash 器件的容量, 了解该外部 SPI flash 的访问地址范围。</p> <p><b>注 4 :</b>对于 32MB 的外部 SPI flash, 在使用直接内存映射模式 (DMM 模式) 去读该外部 SPI flash 时, 访问地址范围是 0x08000000 to 0x09FFFFFF。请用户核对好使用的 SPI flash 器件的容量, 了解该外部 SPI flash 的访问地址范围。</p>
[21:20]	<b>BITMODE</b>	<p><b>SPI 接口位模式</b></p> <p>0x0 = 标准模式.</p> <p>0x1 = 双线模式.</p> <p>0x2 = 四线模式.</p> <p>0x3 = 保留.</p> <p><b>注:</b> 仅用于普通 I/O 模式</p>
[19:16]	<b>SUSPITV</b>	<p><b>暂停间隔</b></p> <p>这四个位域提供一次传输中两个连续发送/接收事务之间的挂起时间间隔的配置。默认值是 0x00。当 BURSTNUM = 00, 设置该域没有影响。想要的时间间隔可以根据下面的等式获得 (从当前时钟的最后一个下降沿到下一个时钟的上升沿)</p> <ul style="list-style-type: none"> <li>• (SUSPITV+2)* AHB 时钟周期</li> <li>• 0x0 = 2 AHB时钟周期</li> <li>• 0x1 = 3 AHB时钟周期</li> <li>• .....</li> <li>• 0xE = 16 AHB时钟周期</li> <li>• 0xF = 17 AHB时钟周期</li> <li>• <b>注:</b> 仅用于普通 I/O模式</li> </ul>
[15]	<b>QDIODIR</b>	<p><b>四线/双线模式的 SPI 接口方向选择</b></p> <p>0 =接口信号为输入</p> <p>1 =接口信号为输出</p> <p><b>注:</b> 仅用于普通 I/O 模式</p>

[14:13]	<b>BURSTNUM</b>	<b>发送/接收 突发个数</b> 该域指定在一次传输中多少个发送/接收事务应该被连续执行 0x0 = 在一次传输中, 仅执行一个发送/接收事务 0x1 = 在一次传输中, 执行两个连续的发送/接收事务. 0x2 = 在一次传输中, 执行三个连续的发送/接收事务. 0x3 = 在一次传输中, 执行四个连续的发送/接收事务. <b>注:</b> 仅用于普通 I/O 模式
[12:8]	<b>DWIDTH</b>	<b>发送/接收位长度</b> 该域指定一次发送/接收事务中多少位被发送/接收 0x7 = 8 位. 0xF = 16 为. 0x17 = 24 为. 0x1F = 32 为. 其他 = 不正确的传输结果 <b>注 1:</b> 仅用于普通 I/O 模式 <b>注 2:</b> 仅 8-, 16-, 24-, 和 32-位是允许的, 其他位长度会导致不正确的传输
[7]	<b>IF</b>	<b>中断标志</b> <b>(1) 写操作 :</b> 0 = 没有影响 1 = 写 1 清 0 <b>(2) 读操作 :</b> 0 = 传输还未完成 1 = 传输已完成
[6]	<b>IEN</b>	<b>中断使能控制</b> 0 = SPIM 中断禁用 1 = SPIM 中断使能
[5]	<b>B4ADDREN</b>	<b>4-字节地址模式使能控制</b> 0 = 禁用 4-字节地址模式, 和使能 3-字节地址模式 1 = 使能 4-字节地址 <b>注:</b> 用于 DMA 写/DMA 读/DMM 模式.
[4:3]	<b>Reserved</b>	保留
[2]	<b>BALEN</b>	<b>在密码使能和禁用控制之间平衡 AHB 控制时间</b> 当密码使能, AHB 控制信号根据编码解码的计算会延时一段时间, 所以如果设置 BALEN 为 1 , 会使密码使能和密码禁止时的 AHB 信号处理时间相等 <b>注:</b> 仅当密码禁用时有用
[1]	<b>Reserved</b>	保留
[0]	<b>CIPHOFF</b>	<b>密码禁用控制</b> 0 = 密码功能使能 1 = 密码功能禁用 <b>注 1:</b> 如果 KEY1(SPIM_KEY1[31:0]) 或 KEY2(SPIM_KEY2[31:0]) 什么都没有 (KEY1 为 0x0000_0000 或 KEY2 是 0x0000_0000) , 密码功能会被自动禁用。 <b>注 2:</b> 当 CIPHOFF(SPIM_CTL0[0]) 为 0, KEY1(SPIM_KEY1[31:0]) 和 KEY2(SPIM_KEY2[31:0]) 都不为 0x0000_0000(i.e. KEY1 ≠ 0x0000_0000 和 KEY2 ≠ 0x0000_0000) , 使能密码加密和解密 <b>注 3:</b> 当使能密码加密和解密时, 请设置 DESELTIM(SPIM_DMMCTL[20:16]) >= 0x10。当禁用密码加密和解密时, 请设置 DESELTIM(SPIM_DMMCTL[20:16]) >= 0x8。

**SPI\_M\_CTL1 控制寄存器 1**

寄存器	偏移地址	R/W	描述			复位值
SPI_M_CTL1	SPI_M_BA+0x04	R/W	控制寄存器 1			0x0000_0010

31	30	29	28	27	26	25	24
DIVIDER							
23	22	21	20	19	18	17	16
DIVIDER							
15	14	13	12	11	10	9	8
Reserved				IDLETIME			
7	6	5	4	3	2	1	0
Reserved		SSACTPOL	SS	CDINVAL	CCMEN	CACHEOFF	SPI MEN

位	描述
[31:16]	<b>DIVIDER</b> <b>时钟分频寄存器</b> 该域的值是 AHB 时钟(HCLK)的频率分频值，用来产生 SPI_M_CLK 引脚上输出的串行时钟，想要的频率可以根据下面的公式获得 $f_{SPI_M\_CLK} = \frac{f_{HCLK}}{(DIVIDER)*2}$ <p><b>注 1:</b> 当设置 DIVIDER 为 0， SPI_M_CLK 的频率会等于 HCLK 的频率  <b>注 2:</b> SCLK 为串行 SPI 输出时钟  <b>注 3:</b> 请参考所使用 SPI flash 器件的规格书决定 SPI flash 时钟频率  <b>注 4:</b> 对于 DTR/DDR 读命令 “0x0D, 0xBD, 0xED”， DIVIDER 设置值仅为 1,2,4,8,16,32,...,2^n, 且 n = 0,1,2,3,4, ....</p>
[15:12]	<b>Reserved</b> 保留
[11:8]	<b>IDLETIME</b> <b>空闲时间间隔</b> 在 DMM 模式, IDLECNT 用来控制两次 SPI Flash 访问之间的最小的空闲时间。 最小时间 = (IDLETIME + 1) * AHB 时钟周期时间. <p><b>注 1:</b> 仅使用于 DMM 模式.  <b>注 2 :</b> AHB 时钟周期时间= 1/AHB 时钟频率</p>
[7:6]	<b>Reserved</b> 保留
[5]	<b>SSACTPOL</b> <b>从机选择激活电平</b> 该位定义了设备/从机选择信号的激活电平(SPI_M_SS)，表 7.20-2 有列举 0 = SPI_M_SS 从机选择信号低有效. 1 = SPI_M_SS 从机选择信号高有效.
[4]	<b>SS</b> <b>从机选择激活使能控制</b> 0 = SPI_M_SS 处在激活电平 1 = SPI_M_SS 处在非激活电平(默认). <p><b>注:</b> 在给定的时间，这个接口仅驱动一个设备/从机。所以在开始任何读或写传输之前，已选择的设备的从机选择必须设置成它的有效电平。SSACTPOL(SPI_M_CTL1[5]) 和 SS 的功能</p>

		说明见表 7.20-2
[3]	CDINVAL	<p><b>缓存数据无效使能控制</b></p> <p><b>(1) 写操作:</b> 0 = 无影响 1 = 设置所有缓存数据无效。该位被硬件自动清除。</p> <p><b>(2) 读操作 :</b>无影响</p> <p><b>注:</b> 当 SPI flash 内存进行页擦除或整个 flash 擦除时, 请设置 CDINVAL 为 1. 在使用 SPI flash 控制器的普通 I/O 模式或 DMA 写模式去编程或擦除外部 SPI flash 的内容之后, 请设置 CDINVAL 为 1.</p>
[2]	CCMEN	<p><b>CCM 模式使能控制</b></p> <p>0 = 禁用 CCM 模式. (默认值) 1 = 使能 CCM 模式</p> <p><b>注 1:</b>当 CCM 模式使能时, 缓存功能会被硬件自动禁用。当 CCM 模式被禁用时, 缓存功能被使能或是禁用是用户自己设置。</p> <p><b>注 2:</b>当 CCM 模式禁用时, 用户通过总线主控访问内核耦合存储器。该情况下, SPI flash 控制器将会通过 HRESP 总线发送错误响应信号到总线主控。</p> <p><b>注 3:</b>当 CCM 模式使能, 用户需要设置 CCMEN 为 1 并且需要读该寄存器显示当前硬件的状态。当读出 CCMEN 值为 1 时, MCU 就可以启动从 CCM 存储空间读数据或是写入数据到 CCM 存储空间。</p>
[1]	CACHEOFF	<p><b>缓存存储功能禁用控制</b></p> <p>0 = 使能缓存存储功能. (默认值) 1 = 禁用缓存存储功能.</p> <p><b>注:</b> 当 CCM 模式使能时, 缓存功能会被硬件自动禁用。当 CCM 模式被禁用时, 缓存功能被使能或是禁用是用户自己设置。</p>
[0]	SPIMEN	<p><b>执行 和 忙状态</b></p> <p><b>(1) 写操作 :</b> 0 = 无影响 1 = 启动传输, 该位在传输期间保持为 1, 传输完成后被自动清除</p> <p><b>(2) 读操作 :</b> 0 = 传输已完成 1 = 传输还未完成</p> <p><b>注:</b> 在写 1 到 SPIMEN 位之前, 所有寄存器应该设置好。在传输过程中, 不应该写该外设的任何寄存器。</p>

SSACTPOL	SS	功能描述
0	0	SPIM_SS 低电平有效, 并且设置SPIM_SS为 0
0	1	SPIM_SS低电平有效, 并且设置SPIM_SS 为 1
1	0	SPIM_SS 高电平有效, 并且设置SPIM_SS 为1
1	1	SPIM_SS高电平有效, 并且设置SPIM_SS 为 0

表 6.21-2 从机片选电平和激活使能的功能描述

**SPIM\_RXCLKDLY RX时钟延时控制寄存器**

寄存器	偏移地址	R/W	描述	复位值
SPIM_RXCLKDLY	SPIM_BA+0x0C	R/W	RX时钟延时控制寄存器	0x0000_010F

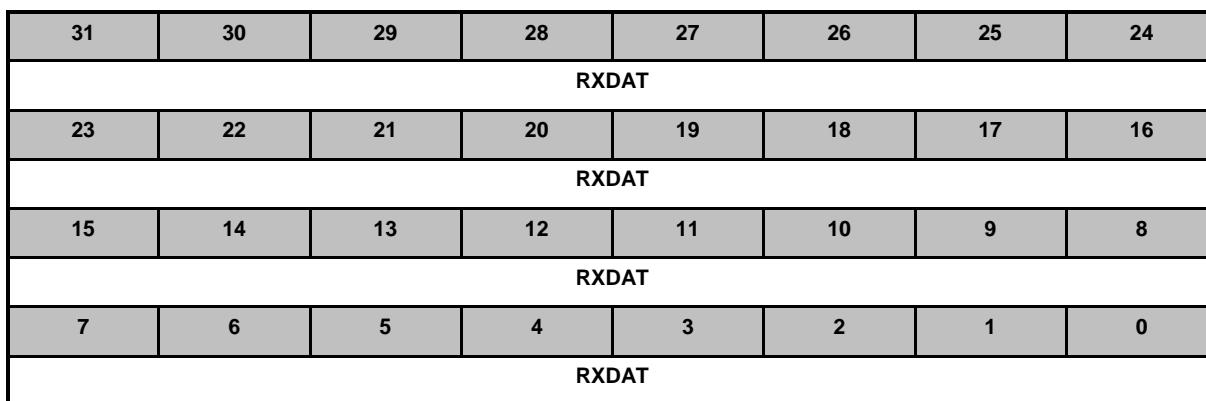
31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved			RDEdge	Reserved	RDDLYSEL		
15	14	13	12	11	10	9	8
PHDELSEL							
7	6	5	4	3	2	1	0
DWDELSEL							

位	描述	
[31:21]	<b>Reserved</b>	保留
[20]	<b>RDEdge</b>	<p>接收数据时的采样时钟边沿选择 针对于普通 I/O 模式、DMA 读模式、DMA 写模式和直接内存映射模式 0 : 使用 SPI 输入时钟上升沿去采样接收到数据 (默认值) 1 : 使用 SPI 输入时钟下降沿去采样接收到数据</p>
[19]	<b>Reserved</b>	保留
[18:16]	<b>RDDLYSEL</b>	<p>接收数据时的采样时钟延时选择 针对于普通 I/O 模式、DMA 读模式、DMA 写模式和直接内存映射模式 决定插入延时周期的数目。用来调整接收数据的采样时钟来锁存正确的数据。 0x0 : 没有延时. (默认值) 0x1 : 延时 1 个 SPI flash 时钟 0x2 : 延时 2 个 SPI flash 时钟 0x3 : 延时 3 个 SPI flash 时钟 ... 0x7 : 延时 7 个 SPI flash 时钟 <b>注 :</b>我们可以使用外部 SPI flash 器件的制造商 ID 或设备 ID 来决定 RDDLYSEL 正确的设置,请参考下面的范例。 例如, 外部 SPI flash 器件的制造商 ID 和设备 ID 分别是 0xEF 和 0x1234。首先我们设置 RDDLYSEL 为 0, 通过普通 I/O 模式使用读制造商 ID/设备 ID 命令去读制造商 ID (本例程的制造商 ID 是 0xEF (1110_1111))。 如果从外部 SPI flash 读出的制造商 ID 是 0xF7 (1111_0111), 这表明制造商 ID 被右移了一位并且大多数制造商 ID 的最高有效位(MSB)都是 1。依据从外部 SPI flash 读出的制造商 ID, 我们需要设置 RDDLYSEL 为 1 来接收正确的 SPI flash 数据。</p>
[15:8]	<b>PHDELSEL</b>	<p>DMA 写模式和 DMA 读模式下的 SPI Flash 的相位延时 这一位设定当 SPI 控制器向 SPI Flash 发送数据时, 指令数据, 地址数据和空指令周期的相位差。</p>

		DMA 写模式和 DMA 读模式下的相位延时 = (PHDESEL + 1) * AHB 时钟周期时间 (注).  注: AHB 时钟周期时间 = 1/AHB 时钟频率
[7:0]	<b>DWDESEL</b>	<b>DMA 写模式下的 SPI flash 取消选择时间间隔(默认值 = 15)</b> 仅针对 <b>DMA 写模式</b> 该寄存器设置 SPI flash 控制在 DMA 写模式下的取消选择时间间隔（也就是 SPIM_SS 非激活的时间间隔）(注 1) DMA 写模式的取消选择时间间隔= (DWDESEL + 1) * AHB 时钟周期时间 (注 2) <b>注 1:</b> 请用户依据使用的外部 SPI flash 器件去设置该寄存器的值。通常，当 SPI flash 执行编程操作时 SPI flash 的取消选择时间间隔大于 50ns <b>注 2:</b> AHB 时钟周期时间= 1/AHB 时钟频率

**SPIM\_RX0~3 数据接收寄存器0~3**

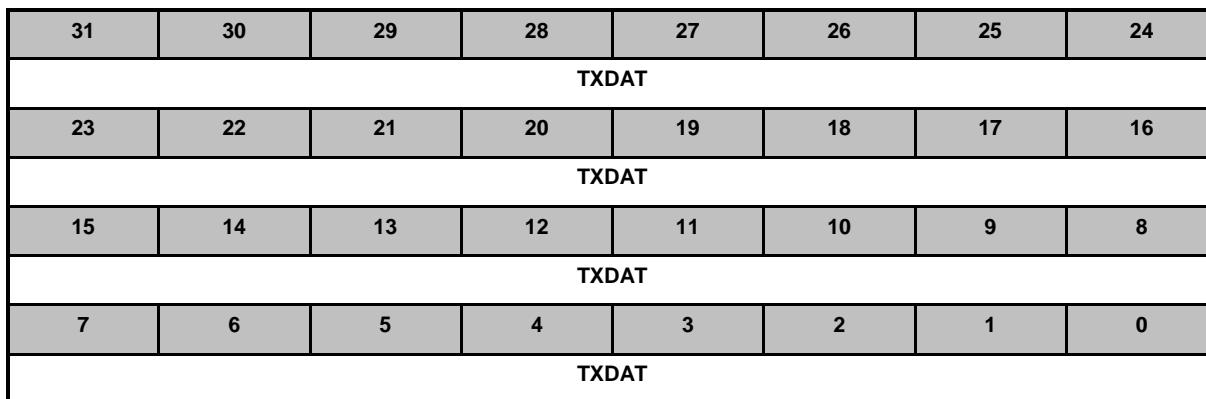
寄存器	偏移地址	R/W	描述	复位值
SPIM_RX0	SPIM_BA+0x10	R	数据接收寄存器0	0x0000_0000
SPIM_RX1	SPIM_BA+0x14	R	数据接收寄存器1	0x0000_0000
SPIM_RX2	SPIM_BA+0x18	R	数据接收寄存器2	0x0000_0000
SPIM_RX3	SPIM_BA+0x1C	R	数据接收寄存器3	0x0000_0000



位	描述
[31:0]	<b>RXDAT</b> <b>数据接收寄存器</b> 数据接收寄存器保留最近一次已执行传输的接收到的数据。 有效 RXDAT 寄存器的个数在 SPIM_CTL0[BURSTNUM] 中指定，如果 BURSTNUM > 0，接收到的数据先保存在最高有效 RXDAT 寄存器中。 有效位的个数在 SPIM_CTL0[DWIDHT] 中指定，如果 DWIDTH 是 16, 24, 或 32，接收到的数据先保存在 RXDAT 寄存器的最低有效字节。 在一个字节中，接收到的数据先保存在 RXDAT 寄存器的最高有效位。 <b>例 1:</b> 如果 SPIM_CTL0[BURSTNUM] = 0x3 和 SPIM_CTL1[DWIDHT] = 0x17，接收到的数据会按顺序保存在 SPIM_RX3[23:0], SPIM_RX2[23:0], SPIM_RX1[23:0], SPIM_RX0[23:0]。 <b>例 2:</b> 如果 SPIM_CTL0[BURSTNUM] = 0x0 和 SPIM_CTL0[DWIDHT] = 0x17，接收到的数据会按顺序保存在 SPIM_RX0[7:0], SPIM_RX0[15:8], SPIM_RX0[23:16]。 <b>例 3:</b> 如果 SPIM_CTL0[BURSTNUM] = 0x0 和 SPIM_CTL0[DWIDHT] = 0x07，接收到的数据会按顺序保存在 SPIM_RX0[7], SPIM_RX0[6], ..., SPIM_RX0[0]。

**SPIM\_TX0~3 数据发送寄存器0~3**

寄存器	偏移地址	R/W	描述	复位值
<b>SPIM_TX0</b>	SPIM_BA+0x20	R/W	数据发送寄存器0	0x0000_0000
<b>SPIM_TX1</b>	SPIM_BA+0x24	R/W	数据发送寄存器1	0x0000_0000
<b>SPIM_TX2</b>	SPIM_BA+0x28	R/W	数据发送寄存器2	0x0000_0000
<b>SPIM_TX3</b>	SPIM_BA+0x2C	R/W	数据发送寄存器3	0x0000_0000



位	描述
[31:0]	<b>TXDAT</b> <b>数据发送寄存器</b> 数 据 发 送 寄 存 器 保 存 下 一 次 传 输 要 发 送 的 数 据 有 效 TXDAT 寄 存 器 的 个 数 在 SPIM_CTL0[BURSTNUM] 中 指 定，如 果 BURSTNUM > 0，数 据 会 先 发 送 最 高 有 效 TXDAT 寄 存 器 。 有 效 位 的 个 数 在 SPIM_CTL0[DWIDTH] 中 指 定。如 果 DWIDTH 是 16, 24, 或 32，会 先 发 送 TXDAT 寄 存 器 的 最 低 有 效 字 节 在 一 个 字 节 中，先 发 送 TX 寄 存 器 的 最 高 有 效 位 <b>例 1：</b> 如 果 SPIM_CTL0[BURSTNUM] = 0x3 和 SPIM_CTL1[DWIDTH] = 0x17，下 次 传 输 数 据 会 按 SPIM_TX3[23:0], SPIM_TX2[23:0], SPIM_TX1[23:0], SPIM_TX0[23:0] 顺 序 发 送。 <b>例 2：</b> 如 果 SPIM_CTL0[BURSTNUM] = 0x0 和 SPIM_CTL0[DWIDTH] = 0x17，下 次 传 输 数 据 会 按 SPIM_TX0[7:0], SPIM_TX0[15:8], SPIM_TX0[23:16] 顺 序 发 送。 <b>例 3：</b> 如 果 SPIM_CTL0[BURSTNUM] = 0x0 和 SPIM_CTL0[DWIDTH] = 0x07，下 次 传 输 数 �据 会 按 SPIM_TX0[7], SPIM_TX0[6], ..., SPIM_TX0[0] 顺 序 发 送。

**SPIM\_SRAMADDR SRAM内存地址寄存器**

寄存器	偏移地址	R/W	描述	复位值
SPIM_SRAMADDR	SPIM_BA+0x30	R/W	SRAM内存地址寄存器	0x0000_0000

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

位	描述	
[31:0]	ADDR	<p><b>SRAM内存地址</b></p> <p>对于 DMA 读模式，该寄存器是 DMA 传输的目的地址      对于 DMA 写模式，该寄存器是 DMA 传输的源地址</p> <p>注：该地址必须是字对齐。</p>

**SPI\_MMACNT DMA传输字节计数寄存器**

寄存器	偏移地址	R/W	描述	复位值
SPI_MMACNT	SPI_MBA+0x34	R/W	DMA传输字节计数寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
DMACNT							
15	14	13	12	11	10	9	8
DMACNT							
7	6	5	4	3	2	1	0
DMACNT							

位	描述	
[31:24]	<b>Reserved</b>	保留
[23:0]	<b>DMACNT</b>	<p><b>DMA传输字节计数寄存器</b>          该位表示DMA传输的长度  <b>注1:</b>计数单位是字节  <b>注2:</b>数值必须是4的倍数  <b>注3:</b>请核对所使用的SPI flash的规格书了解页编程最大字节长度       </p>

**SPIM\_FADDR SPI Flash地址寄存器**

寄存器	偏移地址	R/W	描述	复位值
SPIM_FADDR	SPIM_BA+0x38	R/W	SPI Flash地址寄存器	0x0000_0000

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

位	描述	
[31:0]	ADDR	<p><b>SPI Flash地址寄存器</b></p> <p>对于 DMA 读模式，该寄存器是 DMA 传输的源地址。 对于 DMA 写模式，该寄存器是 DMA 传输的目的地址。</p> <p><b>注1：</b>该地址必须是字对齐。</p> <p><b>注2：</b>对于32MB的外部SPI flash，当用户使用DMA写和DMA读模式去写/读外部SPI flash数据时，该SPI flash地址寄存器“ADDR”的值是从0x00000000 到 0x1FFFFFF。请检查所使用的SPI flash器件的大小了解外部SPI flash的访问地址范围。</p>

SPI\_M KEY1 密钥1 寄存器

寄存器	偏移地址	R/W	描述	复位值
SPI_M KEY1	SPI_M_BA+0x3C	W	密钥1 寄存器	0x0000_0000

31	30	29	28	27	26	25	24
KEY1							
23	22	21	20	19	18	17	16
KEY1							
15	14	13	12	11	10	9	8
KEY1							
7	6	5	4	3	2	1	0
KEY1							

位	描述	
[31:0]	KEY1	<p><b>密钥1 寄存器</b>          该域是密码功能KEY1的数据</p> <p><b>注 1:</b> 如果 KEY1(SPI_M_KEY1[31:0]) 或 KEY2(SPI_M_KEY2[31:0])什么都没有 (KEY1 为 0x0000_0000 或 KEY2 是 0x0000_0000)，密码功能会被自动禁用。</p> <p><b>注 2:</b> 当 CIPHOFF(SPI_M_CTL0[0])为 0，KEY1(SPI_M_KEY1[31:0]) 和 KEY2(SPI_M_KEY2[31:0])都不为 0x0000_0000(也就是 KEY1≠ 0x0000_0000 和 KEY2≠0x0000_0000)，使能密码加密和解密</p>

SPI\_M KEY2 密钥2寄存器

寄存器	偏移地址	R/W	描述	复位值
SPI_M KEY2	SPI_M_BA+0x40	W	密钥2寄存器	0x0000_0000

31	30	29	28	27	26	25	24
KEY2							
23	22	21	20	19	18	17	16
KEY2							
15	14	13	12	11	10	9	8
KEY2							
7	6	5	4	3	2	1	0
KEY2							

位	描述	
[31:0]	KEY2	<p><b>密钥2 寄存器</b>          该域是密码功能KEY2的数据  <b>注 1:</b> 如果 KEY1(SPI_M_KEY1[31:0]) 或 KEY2(SPI_M_KEY2[31:0])什么都没有（KEY1 为 0x0000_0000 或 KEY2 是 0x0000_0000），密码功能会被自动禁用。  <b>注2:</b> 当CIPHOFF(SPI_M_CTL0[0])为0，KEY1(SPI_M_KEY1[31:0]) 和 KEY2(SPI_M_KEY2[31:0])都不为0x0000_0000(也就是KEY1≠ 0x0000_0000 和 KEY2≠0x0000_0000)，使能密码加密和解密       </p>

SPI\_M\_DMMCTL 直接内存映射模式控制寄存器

寄存器	偏移地址	R/W	描述				复位值
SPI_M_DMMCTL	SPI_M_BA+0x44	R/W	直接内存映射模式控制寄存器				0x0008_0000

31	30	29	28	27	26	25	24
ACTSCLKT				Reserved	UACTSCLK	CREN	BWEN
23	22	21	20	19	18	17	16
Reserved			DESELTIM				
15	14	13	12	11	10	9	8
CRMDAT							
7	6	5	4	3	2	1	0
Reserved							

位	描述
[31:28]	<b>ACTSCLKT</b> <b>SPI Flash 激活 SCLK 时间</b> 仅针对直接内存映射模式、DMA 写模式和 DMA 读模式 该寄存器设置 SPI_M SS 激活边沿和第一个串行 SPI 输出时钟边沿之间的时间间隔，见图 6.21-8。 <b>(1) ACTSCLKT = 0 (功能禁用) :</b> 时间间隔 = 1 AHB 时钟周期时间。 <b>(2) ACTSCLKT ≠ 0 (功能使能) :</b> 时间间隔 = (ACTSCLKT + 3) * AHB 时钟周期时间 <b>注 1 :</b> AHB 时钟周期时间.= 1/AHB 时钟频率 <b>注 2 :</b> SCLK 是 SPI 输出时钟 <b>注 3 :</b> 请参考所使用的 SPI flash 规格书去设置该寄存器的值，不同 SPI flash 厂商设置值可能不同
[27]	<b>Reserved</b> 保留
[26]	<b>UACTSCLK</b> <b>用户设置 SPI Flash 激活 SCLK 时间</b> 仅针对直接内存映射模式、DMA 写模式和 DMA 读模式 0 = 依据 DIVIDER(SPI_M_CTL1[31:16]), 由硬件自动设 ACTSCLKT(SPI_M_DMMCTL[31:28]) (默认值) 1 = 用户手动设置 ACTSCLKT(SPI_M_DMMCTL[31:28]) 当用户想手动设置 ACTSCLKT(SPI_M_DMMCTL[31:28]) 时, 请设置 UACTSCLK 为 1.
[25]	<b>CREN</b> <b>连续读模式使能控制</b> 仅针对直接内存映射模式下读命令码为 0xBB, 0xEB, 0xE7, 0x0D, 0xBD, 0xED(注 2) 0 = 禁用连续读模式 (默认) 1 = 使能连续读模式 对于 SPI flash 读操作, 快速四线读 I/O(0xEB)命令、四线 I/O 字读(0xE7 在 winbond SPI flash)、快速读双线 I/O(0xBB)、DTR/DDR 快速读 (0x0D)、DTR/DDR 快速读双线 I/O (0xBD) 和 DTR/DDR 快速读四线 I/O(0xED)可以在输入地址数据之后通过设置连续读模式 (8 位) 来减少命令的开销。 <b>注:</b> 当用户使用连续读模式功能并且设置 USETEN (SPI_M_CTL2[16]) 为 1 时,

		CRMDAT(SPIM_DMMCTL[15:8])必须依据使用的 SPI flash 规格设置。当用户使用连续读模式功能并且设置 USETEN (SPIM_CTL2[16])为 0 时，CRMDAT(SPIM_DMMCTL[15:8])被默认设置为 WINBOND SPI flash 的值
[24]	<b>BWEN</b>	<p><b>16 字节突发包模式使能控制寄存器( 默认值 = 0 )</b></p> <p>仅针对 WINBOND SPI flash, 直接内存映射模式, 缓存使能和读命令码“0xEB 和 0xE7”</p> <p>0 = 禁用突发包模式 (默认)</p> <p>1 = 使能突发包模式</p> <p>在直接内存映射模式下, 四线读命令“0xEB”和“0xE7”支持突发包模式的缓存应用和连续读模式(或性能增强模式)。对于缓存应用, 突发包模式可以快速地被用来填充缓存线(在该 SPI flash 控制器下, 我们使用缓存数据线大小是 16 个字节)。对于直接内存映射模式和缓存使能条件下的连续读模式(或性能增强模式), 缓存数据丢失, 突发包模式可以快速的让 MCU 获得 SPI flash 数据请求。</p>
[23:21]	<b>Reserved</b>	保留
[20:16]	<b>DESELTIM</b>	<p><b>SPI Flash 取消选择时间</b></p> <p>仅针对直接内存映射模式</p> <p>设置 SPI flash 取消时间的最小时宽 (就是说 SPIM_SS 最小取消选择时间), 见图 6.21-8.</p> <p><b>(1)缓存功能禁用 :</b> SPIM_SS 取消选择时间的最长时间宽度= <math>(DESELTIM + 1) * AHB</math> 时钟周期</p> <p><b>(2)缓存功能使能 :</b> SPIM_SS 取消选择时间的最长时间宽度= <math>(DESELTIM + 4) * AHB</math> 时钟周期</p> <p><b>注 1 :</b> AHB 时钟周期 = 1/AHB 时钟频率</p> <p><b>注 2 :</b> 当使能密码加密和解密时, 请设置 DESELTIM(SPIM_DMMCTL[20:16]) &gt;= 0x10。当禁用密码加密和解密时, 请设置 DESELTIM(SPIM_DMMCTL[20:16]) &gt;= 0x8。</p> <p><b>注 3 :</b> 请参考所使用的 SPI flash 规格书去设置该寄存器的值, 不同 SPI flash 厂商设置值可能不同</p>
[15:8]	<b>CRMDAT</b>	<p><b>连续读模式 (或性能增强模式) 的模式位数据 (默认值=0)</b></p> <p>仅针对直接内存映射模式</p> <p>针对连续读模式 (性能增强模式) 的模式位数据设置</p> <p>当我们设置该模式位(<b>注 1</b>)和设置 CREN(SPIM_DMMCTL[25])时, 可以减少 8 个命令相位时钟, 在 SPIM_SS 被激活后可以立刻读取地址</p> <p><b>注 1 :</b> 请参考所使用的 SPI flash 规格书去设置该寄存器的值, 不同 SPI flash 厂商设置值可能不同</p> <p><b>注 2 :</b> CRMDAT 需要配合 CREN(SPIM_DMMCTL[25])使用</p>
[7:0]	<b>Reserved</b>	保留

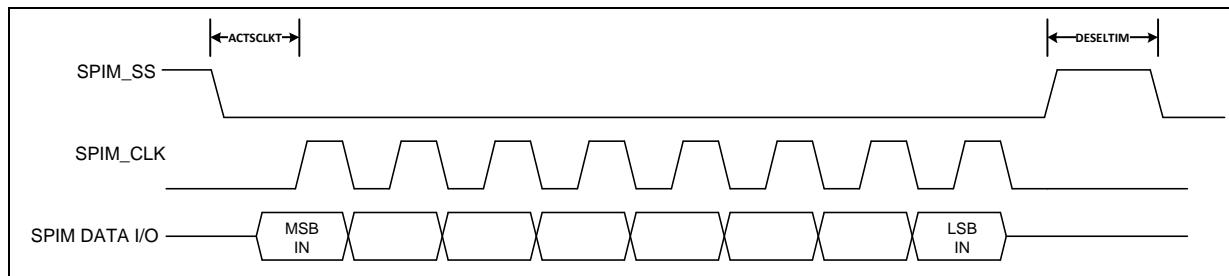


图 6.21-11 SPIM\_SS 激活建立时间与 SPIM\_CLK 的关联和 SPIM\_SS 取消选择时间

SPI\_M\_CTL2 控制寄存器 2

寄存器	偏移地址	R/W	描述	复位值
SPI_M_CTL2	SPI_M_BA+0x48	R/W	控制寄存器 2	0x0804_0000

31	30	29	28	27	26	25	24
Reserved			DCNUM				
23	22	21	20	19	18	17	16
Reserved			DTRMPOFF	Reserved			USETEN
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved							

位	描述	
[31:29]	Reserved	保留
[28:24]	DCNUM	<p>空闲周期 (Dummy Cycle) 数          仅针对直接内存映射模式和 DMA 读模式(注 1)          设置空闲周期数</p> <p>(1)对于非 DTR/非 DDR 命令码 0x03, 0x0B, 0x3B, 0xBB, 0xEB, 和 0xE7：          当读命令码不需要任何空闲周期时（就是空闲周期数=0），用户必须设置 DCNUM 为 0x0。          对于命令码 0xBB，如果模式周期数（或性能增强周期数）和空闲周期数同时不为 0，用户必须依据使用的 SPI flash 规格设置 DCNUM 为“模式周期数 + 空闲周期数”</p> <p>对于命令码 0xBB，如果只有空闲周期数（也就是空闲周期≠0x0 和模式周期数=0x0（或性能增强周期数=0x0）），用户必须依据使用的 SPI flash 规格设置 DCNUM 为“空闲周期数”</p> <p>对于命令码 0x0B, 0x3B, 0xEB, 和 0xE7，用户只需依据使用的 SPI flash 规格设置 DCNUM 为“空闲周期数”</p> <p>(2)对于 DTR/DDR 命令码 0x0D, 0xBD, 和 0xED：          用户只需依据使用的 SPI flash 规格设置 DCNUM 为“空闲周期数”和 DTRMPOFF(SPI_M_CTL2[20])</p> <p>注 1：空闲周期数取决于 SPI 输出的时钟频率、SPI flash 厂商和读命令类型。请参考所使用的 SPI flash 规格书去设置该寄存器的值。</p>
[23:21]	Reserved	保留
[20]	DTRMPOFF	<p>对于 DTR/DDR 命令码为 0x0D, 0xBD, 和 0xED 的模式相位关闭          仅针对直接内存映射模式和 DMA 读模式(注 1)          0 = 在 DTR/DDR 读命令码为 0x0D, 0xBD, 和 0xED 时，模式周期数（或性能增强周期数）不等于 0x0          1 = 在 DTR/DDR 读命令码为 0x0D, 0xBD, 和 0xED 时，模式周期数（或性能增强周期数）等于 0x0</p> <p>注 1：对于 DTR/DDR 命令码为 0x0D, 0xBD, 和 0xED，请参考所使用的 SPI flash 规格书去了解模式周期数（或性能增加周期数）</p>
[19:17]	Reserved	保留

[16]	<b>USETEN</b>	<p>用户设置值使能控制 仅针对于直接内存映射模式和读命令为 0x03,0x0B,0x3B,0xBB,0xEB,0xE7 的 DMA 读模式 0 = SPI flash 控制器的硬件电路会使用一下 DCNUM(SPIM_CTL2[28:24]) 和 CRMDAT(SPIM_DMMCTL[15:8])的默认值去自动配置 SPI flash 操作 空闲周期数 (DCNUM)： 读命令为 0x03 的空闲周期数: 0x0 读命令为 0x0B 的空闲周期数: 0x8 读命令为 0x3B 的空闲周期数: 0x8 读命令为 0xBB 的空闲周期数: 0x0 读命令为 0xEB 的空闲周期数: 0x4 读命令为 0xE7 的空闲周期数: 0x2 连续读模式的模式位数据(CRMDAT) : 0x20 1 = 如果 DCNUM(SPIM_CTL2[28:24]) 和 CRMDAT(SPIM_DMMCTL[15:8])没有设置为以上的默认值，用户必须设置 USETEN 为 1、DCNUM(SPIM_CTL2[28:24]) 和 CRMDAT(SPIM_DMMCTL[15:8])手动配置 SPI flash 操作 对于 DTR/DDR 命令码为 0x0D, 0xBD, 和 0xED, 请设置 USETEN 为 1</p>
[15:0]	<b>Reserved</b>	保留

## 6.22 I<sup>2</sup>C 接口

### 6.22.1 概述

I<sup>2</sup>C 为双线，双向串行总线，通过简单有效的连线方式实现器件间的数据交换。I<sup>2</sup>C 标准是多主机总线，包括冲突检测和仲裁，以防止在两个或多个主机同时尝试控制总线时发生数据损坏。有两组 I<sup>2</sup>C 控制器，都支持掉电唤醒功能。

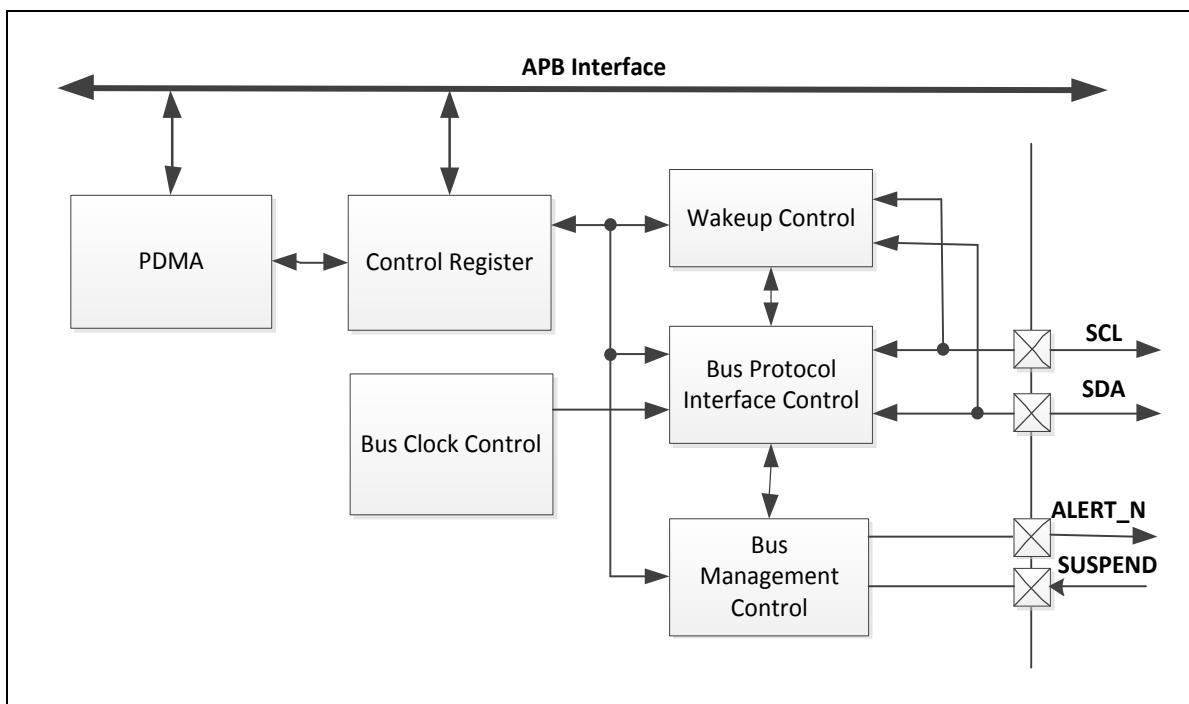
### 6.22.2 特性

I<sup>2</sup>C 通过 SDA 及 SCL 两条线与连接在总线上的器件传输信息，总线的主要特征有：

- 支持最多三组 I<sup>2</sup>C 接口
- 支持主机/从机 模式
- 主从机之间双向数据传输
- 总线支持多主机 (无中心主机)
- 支持10位模式
- 支持高速3.4Mbps模式
- 支持标准模式 (100 kbps), 快速模式 (400 kbps) 和 快速加模式(1 Mbps)
- 多主机间同时传输数据仲裁，避免总线上串行数据损坏
- 总线采用串行同步时钟，可实现设备之间以不同的速率传输
- 串行同步时钟被用作握手机制去挂起和恢复串行传输
- 内建14位溢出定时器，当I<sup>2</sup>C总线中止且定时器溢出时，产生I<sup>2</sup>C中断
- 可配置不同时钟以适用于可变速率控制
- 支持7位地址模式和10位地址模式
- 支持多地址识别（4组从机地址带mask 选项）
- 支持掉电唤醒功能
- 支持带有一个缓冲能力的PDMA
- 支持建立/保持时间可编程
- 支持总线管理(SM/PM兼容)功能

### 6.22.3 框图

I<sup>2</sup>C 控制器的框图如下图 6.22-1.

图 6.22-1 I<sup>2</sup>C 控制器框图

#### 6.22.4 基本配置

##### 6.22.4.1 I2C0 基本配置

- 时钟源配置
  - 使能I2C0时钟控制位是 I2C0CKEN (CLK\_APBCLK0 [8])
- 复位配置
  - 复位I2C0控制器的控制位是 I2C0RST(SYS\_IPRST1 [8])
- 管脚配置

组	管脚名称	GPIO	MFP
I2C0	I2C0_SCL	PC.12, PD.7, PE.13, PF.3, PG.0	MFP4
		PB.5, PH.2	MFP6
		PA.5, PC.1	MFP9
	I2C0_SDA	PC.8, PC.11, PD.6, PF.2, PG.1	MFP4
		PB.4, PH.3	MFP6
		PA.4, PC.0	MFP9
	I2C0_SMBAL	PG.2	MFP4
		PC.3	MFP9
	I2C0_SMBSUS	PG.3	MFP4
		PC.2	MFP9

		PA.15, PD.0	MFP6
		PA.10	MFP7
		PB.12	MFP8
		PA.0, PH.9	MFP9

#### 6.22.4.2 I2C1基本配置

- 时钟源配置
  - 使能I2C1时钟的控制位是 I2C1CKEN (CLK\_APBCLK0 [9])
- 复位配置
  - 复位I2C1控制器的控制位是 I2C1RST(SYS\_IPRST1 [9])
- 管脚配置

组	管脚名称	GPIO	MFP
I2C1	I2C1_SCL	PF.0	MFP3
		PA.12, PD.5	MFP4
		PG.2	MFP5
		PB.11	MFP7
		PA.7, PE.1	MFP8
		PA.3, PB.1, PC.5	MFP9
I2C1	I2C1_SDA	PF.1	MFP3
		PA.13, PD.4	MFP4
		PG.3	MFP5
		PB.10	MFP7
		PA.6, PE.0	MFP8
		PA.2, PB.0, PC.4	MFP9
I2C1	I2C1_SMBAL	PG.0	MFP5
		PB.9	MFP7
		PC.7, PH.8	MFP8
	I2C1_SMBSUS	PG.1	MFP5
		PB.8	MFP7
		PC.6, PH.9	MFP8

#### 6.22.4.3 I2C2基本配置

- 时钟源配置
  - 使能I2C2时钟的控制位是 I2C2CKEN (CLK\_APBCLK0 [10])

- 复位配置
  - 复位I2C2控制器的控制位是 I2C2RST(SYS\_IPRST1[10])
- 管脚配置

组	管脚名称	GPIO	MFP
I2C2	I2C2_SCL	PD.9	MFP3
		PA.14, PD.1	MFP6
		PA.11	MFP7
		PB.13	MFP8
		PA.1, PH.8	MFP9
	I2C2_SDA	PD.8	MFP3
		PA.15, PD.0	MFP6
		PA.10	MFP7
		PB.12	MFP8
		PA.0, PH.9	MFP9
	I2C2_SMBAL	PB.15	MFP8
	I2C2_SMBSUS	PB.14	MFP8

### 6.22.5 功能描述

在I<sup>2</sup>C总线上，数据通过时钟线SCL和数据线SDA在主从机间逐一字节同步传送。每个字节数据长度是8位。一个SCL时钟脉冲传输一个数据位，数据由最高位MSB开始传输，每个字节传输后跟随一个应答位，每个位在SCL为高时采样；因此，SDA线可能在SCL为低时改变，但在SCL为高时必须保持稳定。当SCL为高时，SDA线上的跳变视为一个命令(START或STOP)。更多关于I<sup>2</sup>C总线时序的细节请参考下图 6.22-2。

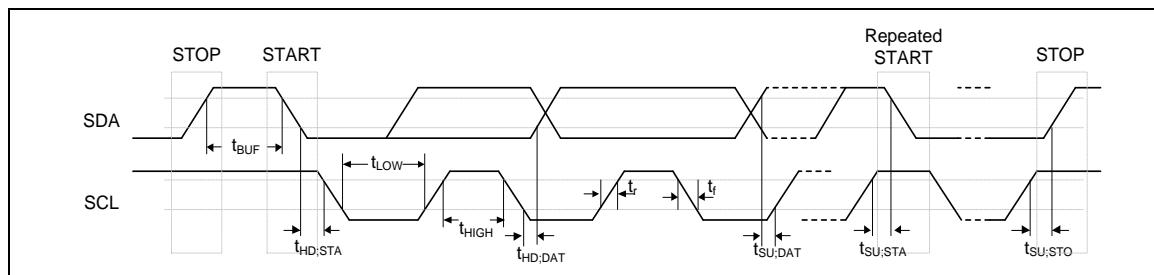


图 6.22-2 I<sup>2</sup>C 总线时序

片上I<sup>2</sup>C外设提供了一个符合I<sup>2</sup>C总线规范的串行接口。I<sup>2</sup>C端口自动处理字节传输。通过设置寄存器I2C\_CTL0的I2CEN为'1'可使能该端口。I<sup>2</sup>C硬件接口通过数据线SDA 和时钟线SCL两个管脚连到I<sup>2</sup>C总线。当I/O管脚作为I<sup>2</sup>C端口使用时，用户必须事先设定I/O管脚为I<sup>2</sup>C功能。

注：因为这两个管脚在I<sup>2</sup>C状态下为开漏脚，SDA 和 SCL两个管脚需要上拉电阻。

#### 6.22.4.4 I<sup>2</sup>C协议

标准I<sup>2</sup>C 协议如下图 6.22-3，通常标准通讯有以下4部分：

- (START) 或者重复起始信号(Repeated START)
- 从机地址传输和R/W 位传输
- 数据传输
- 停止信号(STOP)

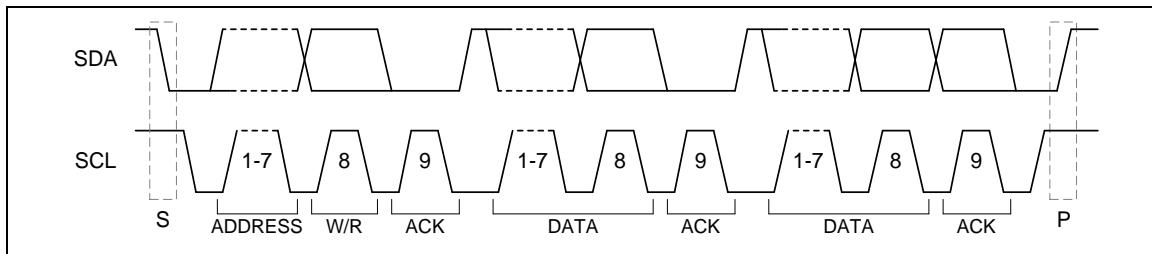


图 6.22-3 I<sup>2</sup>C 协议

- 起始或重复起始信号

当总线处于释放/空闲状态下，说明没有主机设备占用总线（SDA 与SCL线同时为高），主机可以通过发送起始(START)信号来发起传输过程。起始信号，通常表示为”S”位，当SCL 线为高时，SDA 线上信号由高至低变化，就被定义为起始信号。起始信号表示一个新的数据传输的开始。

主机发送完地址字节（地址和读/写位）后，可以发送任何数量的数据并带一个停止信号。也可以用另一个起始信号替代停止信号，随后是地址(包含读写位)和更多的数据。这个起始信号叫做重复起始。这个定义可以用来发送任意个起始信号，它的目的是在不释放总线情况下，对一个或多个设备能读/写操作，而不让操作被打断。用这个方法主机跟其他从机或同一个从机在不同方向传输（例如，写设备到读设备）不用释放总线。

- 停止信号

主机可以通过产生一个停止信号来终止数据通信。停止信号，通常表示为P位，当SCL 线为高时，SDA线上信号由低至高变化，就被定义为停止信号。

下图 6.22-4 为起始，重复起始和停止信号的波形。

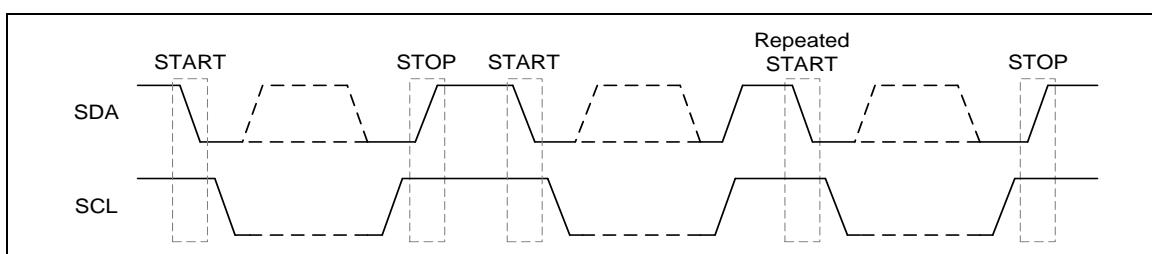


图 6.22-4 起始和停止状况

- 从地址传输

在一个起始（或重复起始）信号后，主机发送一个从机地址来确认通信的目标设备。起始地址有一

个或两个字节组成（针对7位或10位地址情况）。地址字节过后，从机会对传输的地址作应答反应。

因此，从机地址可以被编程并且与接收到的地址比较。一旦匹配，从机会回一个应答( $SDA = 0$ ).如果与目标不匹配则无应答( $SDA = 1$ ).除了编程地址匹配之外，如果从机有能力处理类似的请求，那么另外的地址值也必须有应答回应。

### ● 数据传输

当从机设备地址和R/W位被成功识别到，就可以根据R/W位所决定的方向按一字节一字节方式进行数据传输。每个字节传输完后，紧接着的第9个SCL时钟周期会有一个应答信号位。如果从机上产生无应答信号(NACK)，主机可以产生停止信号来中止数据传输或者产生重复起始信号开始新一轮数据传输。

当主机作为接收设备时，发生无应答信号(NACK)，则从机将释放SDA线，让主机产生停止信号或重复起始信号。图 6.22-5 和 图 6.22-6 展示了传输和应答位的波形。

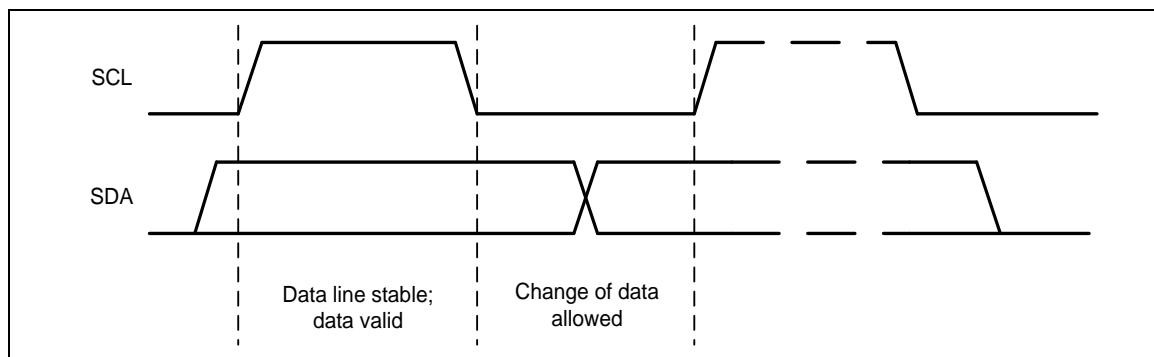


图 6.22-5 I<sup>2</sup>C 总线上的位传输

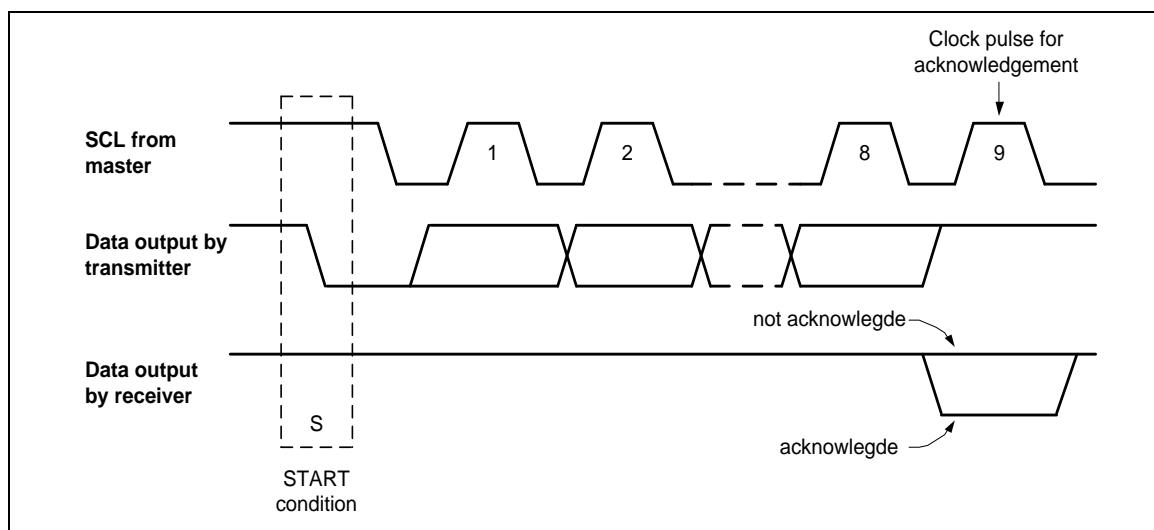


图 6.22-6 I<sup>2</sup>C 总线上的应答信号

- I<sup>2</sup>C 总线上的数据传输

图 6.22-7 表示7位地址情况下主机向从机传输数据。主机发出一个7位地址和1位写指示,表示主机想要传送数据给从机。从机回应答给主机之后, 主机继续传输数据。

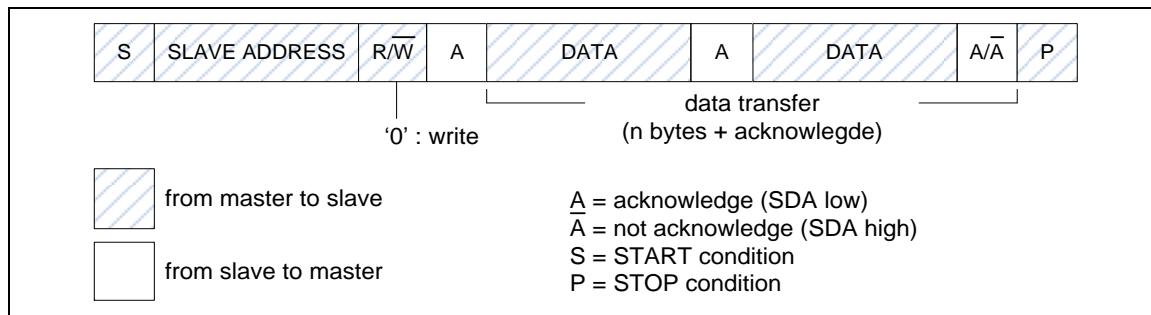


图 6.22-7 7 位地址情况下主机向从机传输数据

图 6.22-8 表示7位地址情况下主机向从机读取数据。主机发7位地址寻址和1位读指示, 表示主机要向从机读取数据, 从机返回应答给主机后, 就开始给主机传输数据。

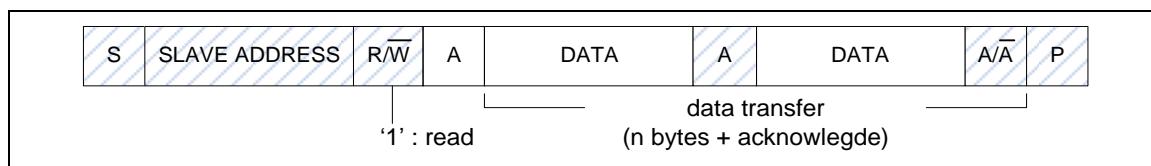


图 6.22-8 7 位地址情况下主机向从机读取数据

图 6.22-9表示10位地址情况下主机向从机传输数据。主机发送10位地址。首字节是有10位地址指示(5'b11110)和2位地址再加上写指示, 第二字节是剩下的8位地址。在第二字节之后, 主机继续传输数据。注意7位和10位地址设备可以工作在同一总线上。

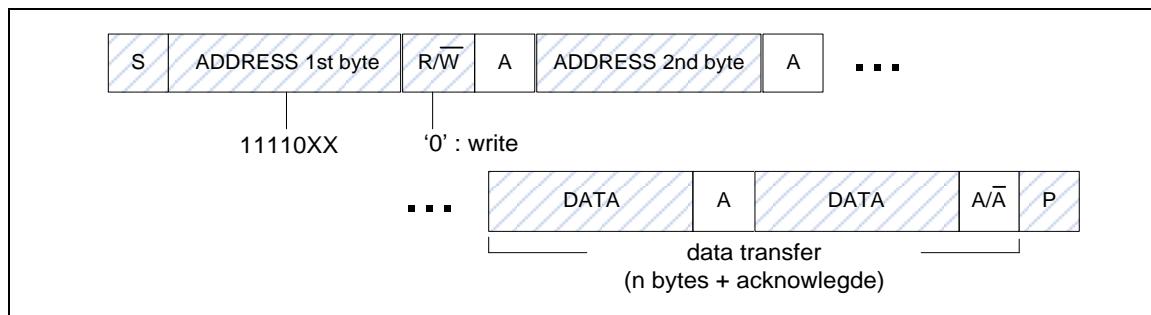


图 6.22-9 10 位地址情况下主机向从机传输数据

图 6.22-10表示10位地址情况下主机向从机读取数据。首先主机发送10位地址给从机, 在主机传输带有读指示的第一个字节之后。在带有读指示的第一个字节之后, 从机就开始给主机传输数据。

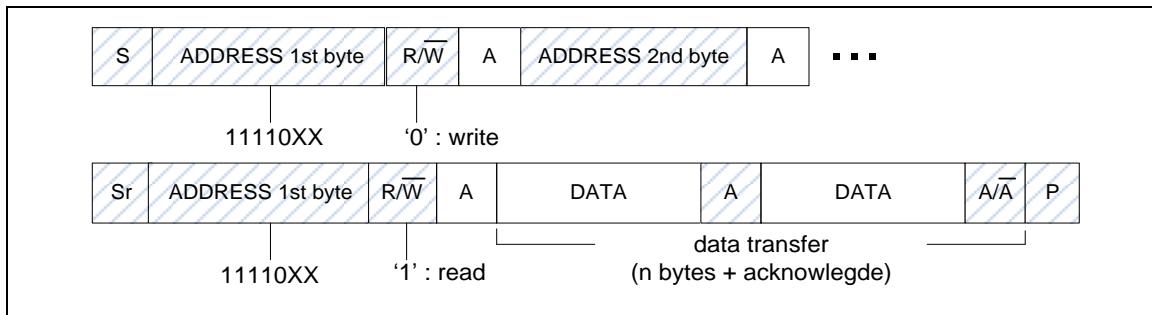


图 6.22-10 10 位地址情况下主机向从机读取数据

#### 6.22.4.5 工作模式

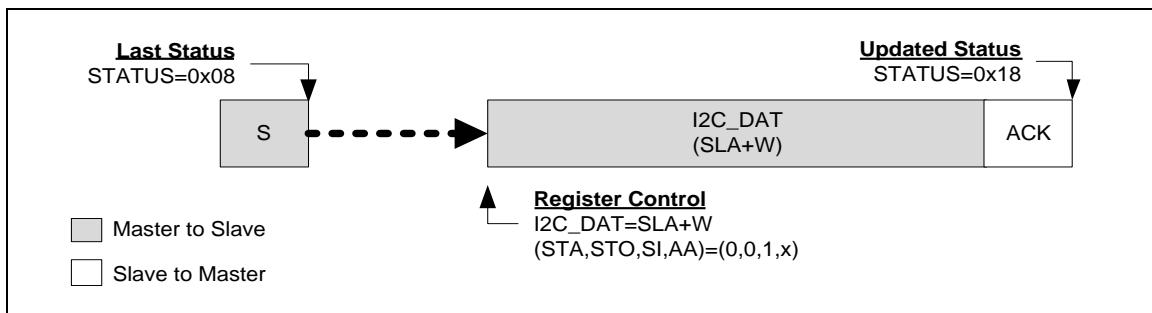
片上I<sup>2</sup>C端口支持三种工作模式：主机模式，从机模式和广播呼叫模式。

应用中，I<sup>2</sup>C 端口可以作为主机和从机。在从机模式，I<sup>2</sup>C端口硬件会查找自身从机地址和广播呼叫地址，如果这两个地址的任一个被检测到，并且从机想要从主机接收或向主机发送数据(通过设置AA位)，应答脉冲将会在第9个时钟发出，此时，如果中断使能，则主机和从机设备上都会发生一次中断请求。在微控制器想要成为总线主机时，在进入主机模式之前，硬件需等待到总线空闲，以保证合理的从机动作不会被打断。在主机模式下，如果总线仲裁丢失，I<sup>2</sup>C端口立即切换到从机模式，并可以在同一次串行传输过程中检测自身

为控制I<sup>2</sup>C总线的各种模式传输，用户需要按照寄存器I2C\_STATUS的当前状态码来设置I2C\_CTL, I2C\_DAT寄存器。换句话说，每个I<sup>2</sup>C总线动作都要检查I2C\_STATUS寄存器的当前状态，然后再设置I2C\_CTL, I2C\_DAT寄存器执行总线动作。最后，通过I2CSTATUS检查响应状态。

在寄存器I2C\_CTL [3]的SI标志清除后，寄存器I2C\_CTL的这些位STA, STO 和 AA 用来控制I2C硬件的下一个状态。当完成一个新的动作，I2C\_STATUS的状态代码将被更新，I2C\_CTL 寄存器的SI标志将被设置。如果I<sup>2</sup>C中断控制位INTEN (I2C\_CTL [7])被设置，新状态代码对应的动作或者软件将在中断服务程序中被执行。

图 6.22-11显示当前I<sup>2</sup>C状态码是0x08，然后通过设置I2C\_DATA=SLA+W和(STA,STO,SI,AA) = (0,0,1,x)发送地址到I<sup>2</sup>C总线。如果总线上有从机匹配相应的地址且返回ACK，I2C\_STATUS状态码更新为0x18。

图 6.22-11 通过当前状态控制I<sup>2</sup>C 总线

- 主机模式

图 6.22-12 和 图 6.22-13是I<sup>2</sup>C主机模式所有可能的协议展示。用户需要遵循恰当的流程来实现I<sup>2</sup>C

协议。

换句话说，用户可以发送一个起始信号到总线，I<sup>2</sup>C总线将设置成主机传输(MT)模式(图 6.22-12)，或主机接收(MR)模式(图 6.22-13)。起始信号设置成功后新的状态码将是0x08。起始信号后，用户可以发送从机地址，读/写位，数据和重复起始，停止来执行I<sup>2</sup>C协议。

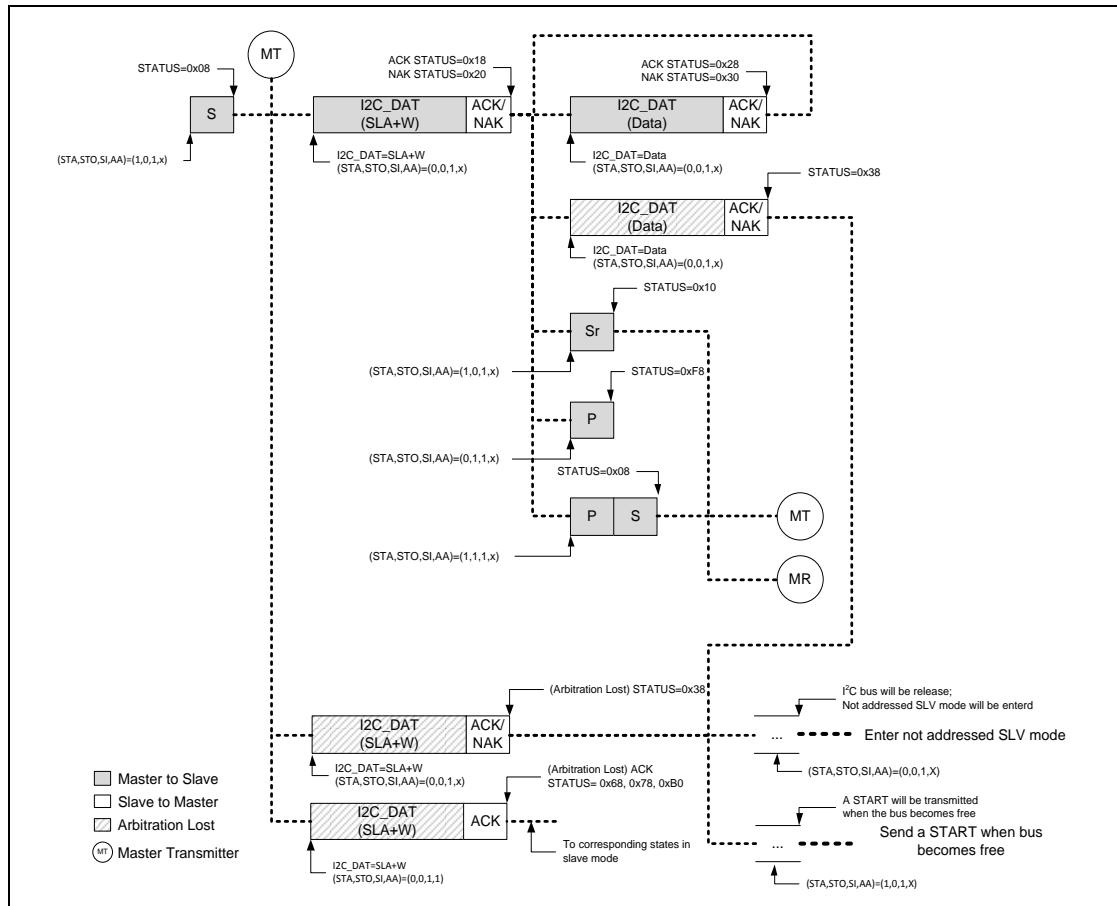


图 6.22-12 主机发送模式控制流程

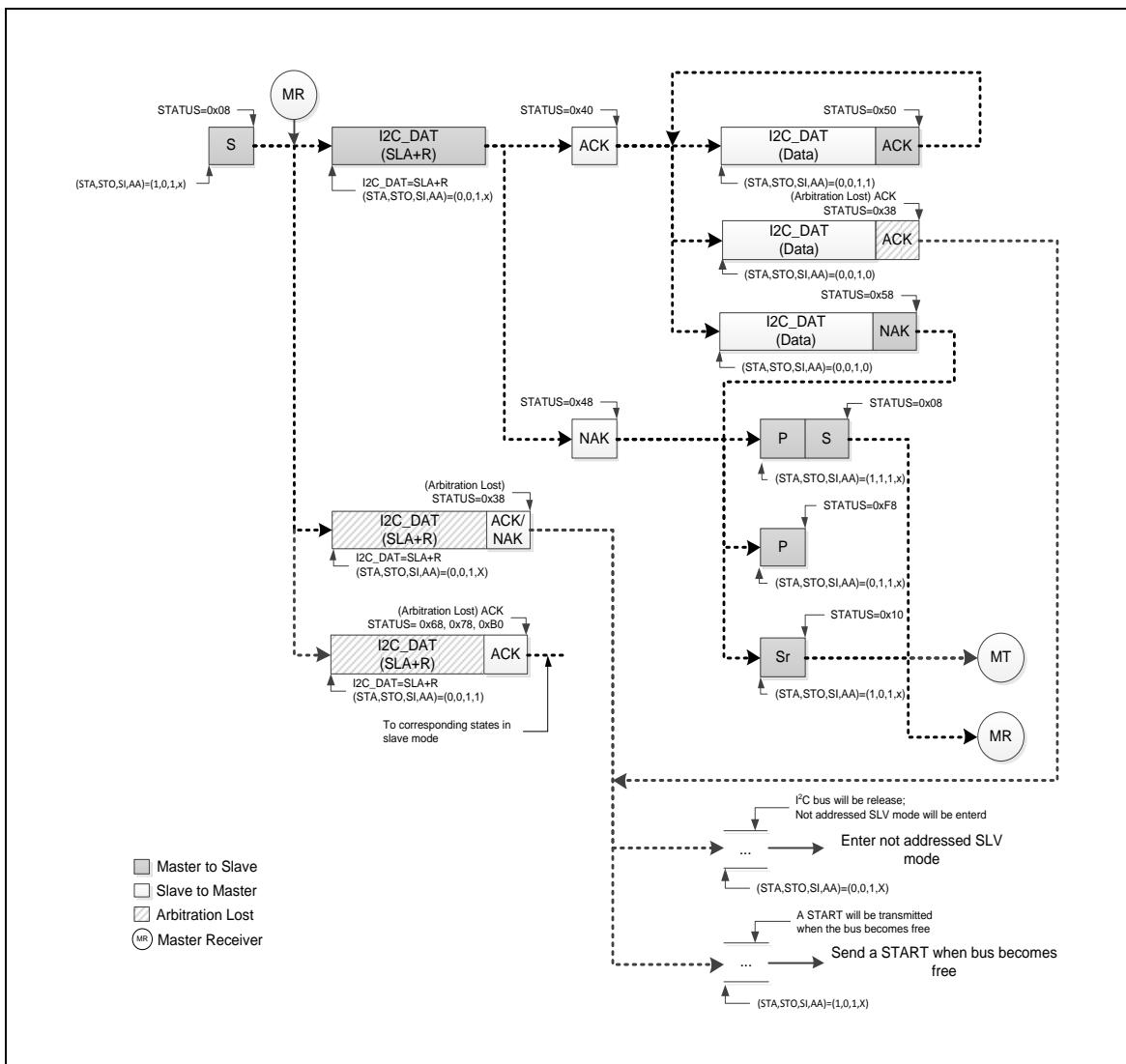


图 6.22-13 主机接收模式控制流程

如果I<sup>2</sup>C在主机模式并且仲裁丢失，状态码将置为0x38。在状态为0x38时，当总线空闲时，用户可以设置(STA, STO, SI, AA) = (1, 0, 1, X)发送起始信号来重新开始主机操作。否则，用户可以设置(STA, STO, SI, AA) = (0, 0, 1, X)来释放总线，并进入无地址从机模式。

### ● 从机模式

复位后默认情况下，I<sup>2</sup>C不会被寻址，并且不会识别I<sup>2</sup>C总线上的地址。用户可以通过I<sup>2</sup>C\_ADDRN(n=0~3)设置从机地址和设置(STA, STO, SI, AA) = (0, 0, 1, 1)来让I<sup>2</sup>C识别主机发送的地址。图 6.22-14所示为I<sup>2</sup>C从机模式的所有可能的流程。用户需要遵循（图 6.22-14）的流程来实现他们的I<sup>2</sup>C协议。

如果在主机模式总线仲裁丢失，I<sup>2</sup>C端口立即切换到从机模式，并且在同一串行传输中识别自有的从机地址。如果在仲裁丢失后识别到地址是SLA+W（主机想写数据到从机），状态码是0x68。如

果在仲裁丢失后识别到地址是SLA+R（主机向从机读数据），状态码是0xB0。

注：I<sup>2</sup>C通信期间，在从机模式下，当对SI标志写‘1’清除时，SCL时钟将被释放。

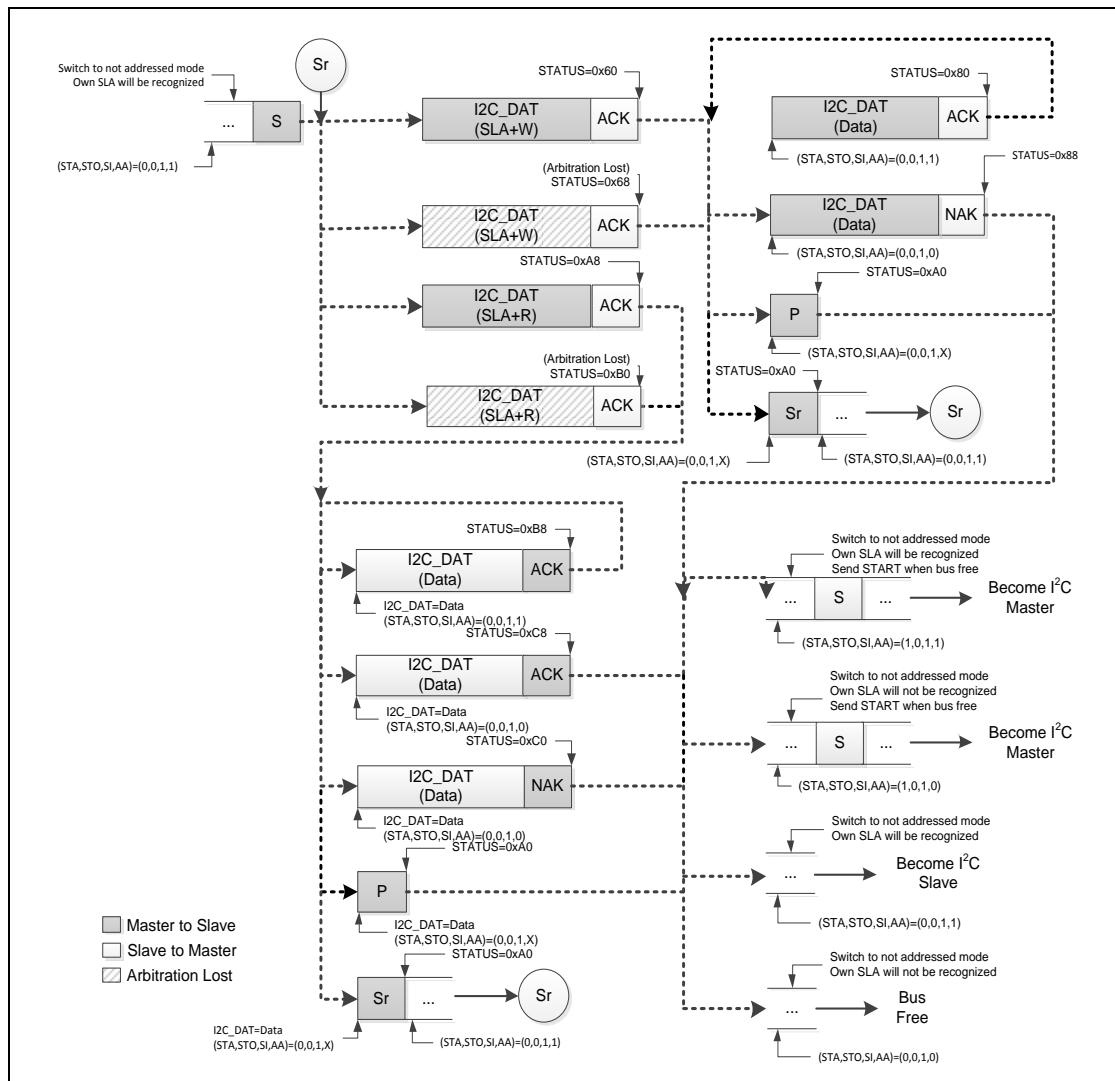


图 6.22-14 从机模式控制流程

如果I<sup>2</sup>C在可寻址从机模式接收数据时，收到停止或重复起始信号，状态码是0xA0。当状态码是0xA0时，用户可以遵循如上图状态码是0x88的操作。

如果I<sup>2</sup>C在可寻址从机模式传输数据时却收到停止或重复起始信号，状态码是0xA0。当状态码是0xA0时，用户可以遵循如上图状态码是0xC8的操作。

注：从机获得0x88, 0xC8, 0xC0 和0xA0状态后，从机可以切换到无地址模式，自身SLA不会被辨识。如果进入这种状态，从机不再接收主机任何信号或地址。在这种状态，I<sup>2</sup>C需要进入空闲模式。

- 广播呼叫模式 (GC)

如果 GC 位(I2C\_ADDRn [0]) 被设, I<sup>2</sup>C 端口硬件将响应广播呼叫地址(0x00). 用户可以通过清 GC 位来禁止广播呼叫功能。当 I<sup>2</sup>C 在从机模式时且 GC 位被设置, 可以接收主机地址(0x00)的广播呼叫, 将遵循广播模式状态。

当地址匹配时 GC 模式可以唤醒。注意默认地址是 0x00, 但是用户必须设置一个非 0x00 的地址。

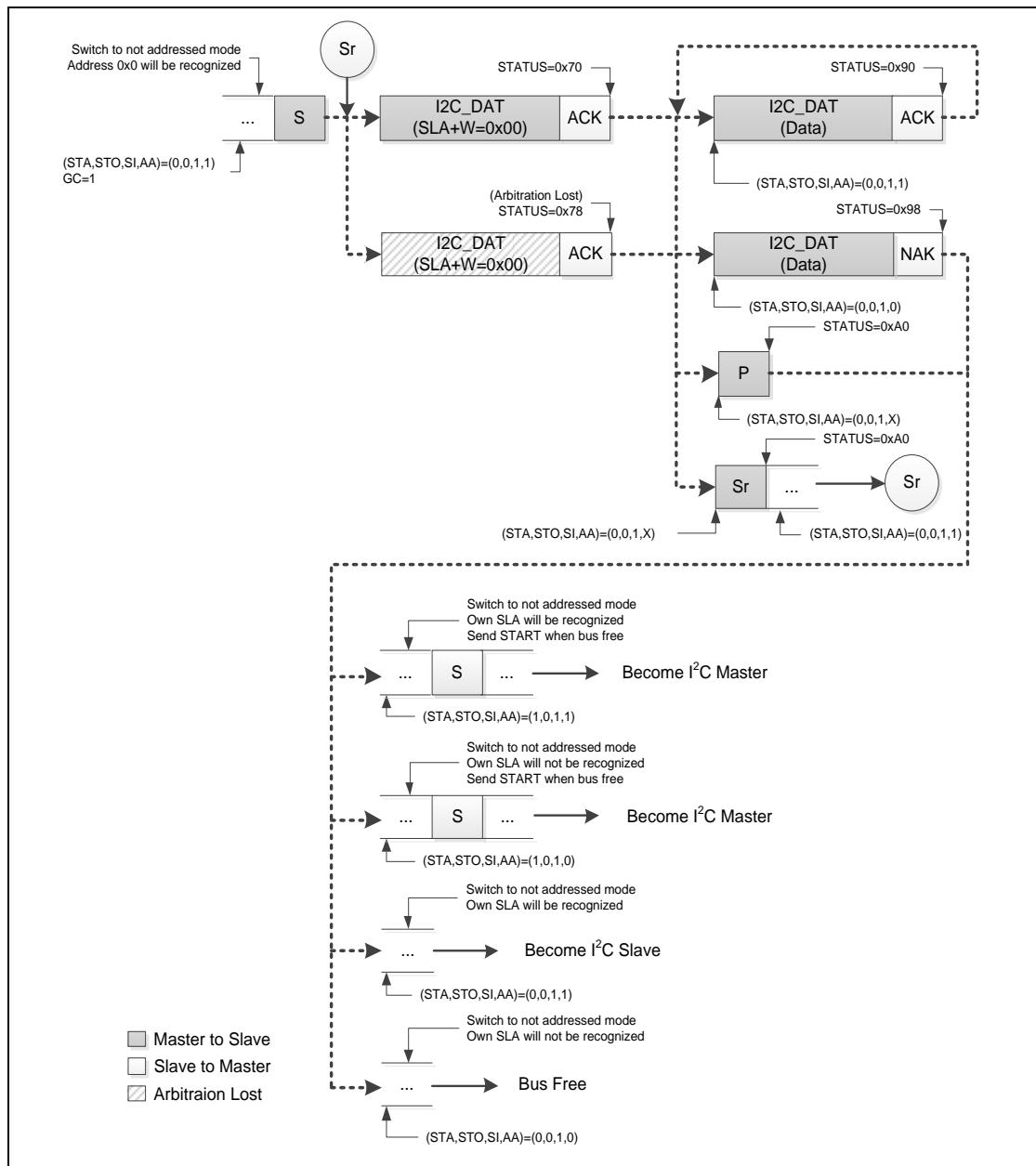


图 6.22-15 GC 模式

如果 I<sup>2</sup>C 在广播呼叫模式接收数据时, 却收到停止或重复起始信号, 状态码是 0xA0。当状态码是 0xA0 时, 用户可以遵循如上图状态码是 0x98 的流程处理。

**注：**从机获得0x98 和0xA0 状态后，从机会切换到无地址模式并且自身SLA将不被辨识。如果进入这种状态，从机不再接收主机的任何信号或地址。 $I^2C$ 需要进入空闲模式。

- 多主机模式

在一些应用中，一个 $I^2C$ 总线上有多个主机同时访问从机，并有可能同时在传送数据。 $I^2C$ 是支持多主机模式，并包含有冲突检测和仲裁，防止数据损坏。

如果两个主机同时发命令，通过仲裁来决定哪个优先并继续发命令。仲裁是在SCL为高时在SDA上执行的。每一个主机都会检测总线上的SDA信号是否符合它产生的SDA信号。如果检测到总线上SDA为低，但应该为高，则这个主机将失去仲裁。设备在仲裁丢失后会产生SCL脉冲直到本字节结束，然后释放总线进入从机模式。仲裁可以一直进行到所有数据被传输完。这样意味着多主机系统中主机必须监控总线和做相应的处理

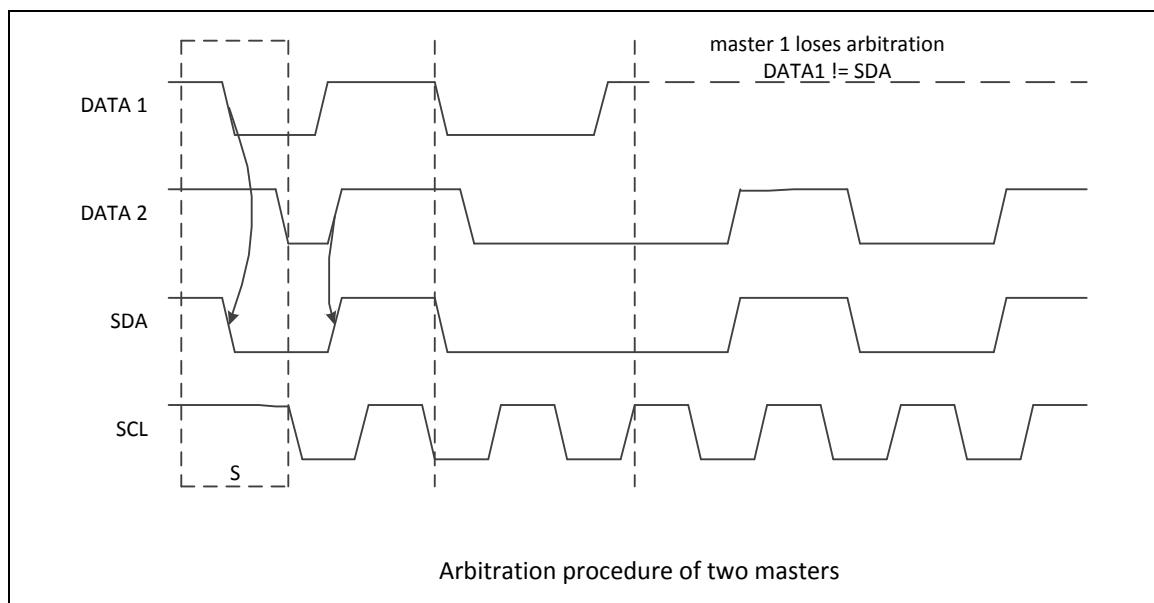


图 6.22-16 仲裁丢失

- 当I2C\_STATUS = 0x38代表接收到一个仲裁丢失。仲裁丢失事件可能发生在发送起始位，数据位或者停止位期间。用户可以在总线空闲时设置(STA, STO, SI, AA) = (1, 0, 1, X)来再次发送起始信号或者设置(STA, STO, SI, AA) = (0, 0, 1, X)发送停止信号来返回到无地址从机模式。用户可以通过ONBUSY (I2C\_STATUS1 [8])侦测总线被释放
- 当I2C\_STATUS = 0x00，接收到一个“总线错误”，为了从错误总线恢复到I<sup>2</sup>C总线，STO应被设置且SI应将被清0，然后对STO清0来释放总线。
  - 设置(STA, STO, SI, AA) = (0, 1, 1, X)停止当前传输
  - 设置(STA, STO, SI, AA) = (0, 0, 1, X)释放总线。
- 总线管理 (SMBus /PMBus兼容)

该部分相关内容只适合支持总线管理特性的I<sup>2</sup>C接口。

## 简介

总线管理是个I<sup>2</sup>C接口，通过总线管理各种设备可以相互通信以及与系统的其它部分通信。它是基

于I<sup>2</sup>C的操作规范。总线管理为系统和电源管理相关任务提供控制总线。

该外设与SMBUS规范2.0版本(<http://smbus.org/specs/>)以及PMBUS规范1.2版本(<http://pmbus.org/>)相兼容。

系统管理总线规范涉及到3个类型的设备。

- 从设备，可以接收数据或响应命令.
- 主设备，可以产生命令，产生时钟和结束传输.
- 特定主设备的主机，可以给系统CPU提供主接口，主机必须是主-从设备而且必须支持SMBus主机识别协议。系统里只允许有一个主机。

总线管理设备是基于I<sup>2</sup>C规范2.1版本

#### 设备识别 – 从机地址

任何存在总线管理的从机设备都有一个唯一的地址称为从机地址。下面的地址为保留而且禁止使用或者赋予任何总线管理设备。(详细请参看SMBus规范)

从设备地址位 位7-1	读/写为 位 0	说明
0000 000	0	广播呼叫地址
0000 000	1	起始字节
0000 001	X	CBUS 地址
0000 010	X	不同总线格式保留地址
0000 011	X	保留将来使用
0000 1XX	X	保留将来使用
0101 000	X	保留访问总线主机
0110 111	X	保留访问总线默认地址
1111 0XX	X	10位从机寻址
1111 1XX	X	保留将来使用
0001 000	X	SMBus 主机
0001 100	X	SMBus警报相应地址
1100 001	X	SMBus设备默认地址

表 6.22-1 SMBus 保留地址

#### 总线协议

任何设备都可能有11种命令协议。设备可以使用11种协议中某一或者所有来通信。这些协议为**Quick Command, Send Byte, Receive Byte, Write Byte, Write Word, Read Byte, Read Word, Process Call, Block Read, Block Write** 和 **Block Write-Block Read Process Call**.这些协议必须由软件来执行。(协议的更详细内容请参看SMBus规范2.0版本)。

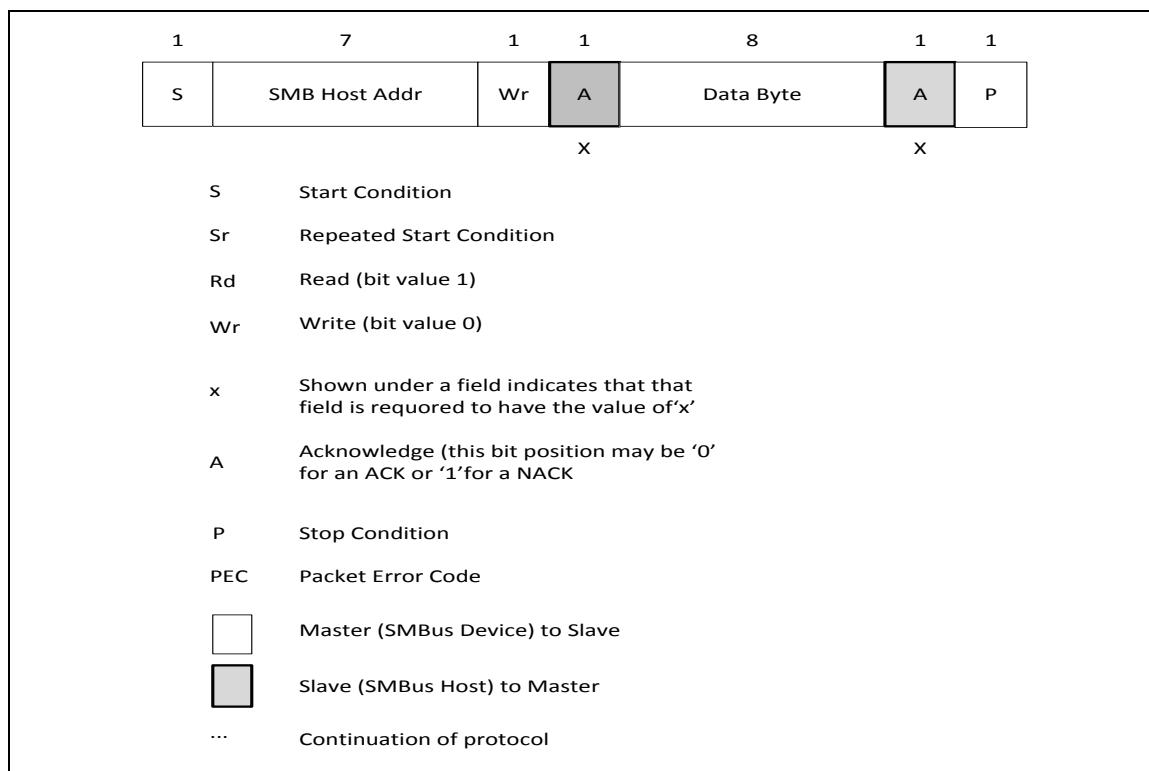


图 6.22-17 总线管理主要元素包协议图

### 地址解析协议(ARP)

从机地址冲突，总线管理可以通过动态给从设备赋予唯一的新地址解决。为了提供地址识别机制来区分各个设备，每个设备必须分配一个唯一设备标识符(UDID)，由软件分配128位数字。

该设备支持地址解析协议(ARP)。通过设置BUSEN (I2C\_BUSCTL[7]), BMDEN (I2C\_BUSCTL[2])和 ALERTEN (I2C\_BUSCTL[4])来使能总线管理设备默认地址(0b1100 001)。ARP命令必须由用户来执行，支持ARP的从机模式也可以执行仲裁。

### 接收命令和数据应答控制

总线管理接受器必须能够NACK每个收到的命令或数据。为了应许从机模式下的ACK控制，必须通过使能ACKMEN位(I2C\_BUSCTL[0])来设置成从设备字节控制模式。

### 主机通知协议

为防止未知设备未知信息的格式进入总线管理主机控制器，仅支持使用写字协议更改的协议。标准的写字协议通过替换命令代码与改变报警设备的地址来修改。

该外设支持主机通知协议，通过设置BUSEN (I2C\_BUSCTL[7]), BMHEN (I2C\_BUSCTL[3]) 和 ALERTEN (I2C\_BUSCTL[4])实现。在该情况下，主机将会确认总线管理主机地址(0b0001000)。该协议工作时设备作为主机，而主机作为从机。

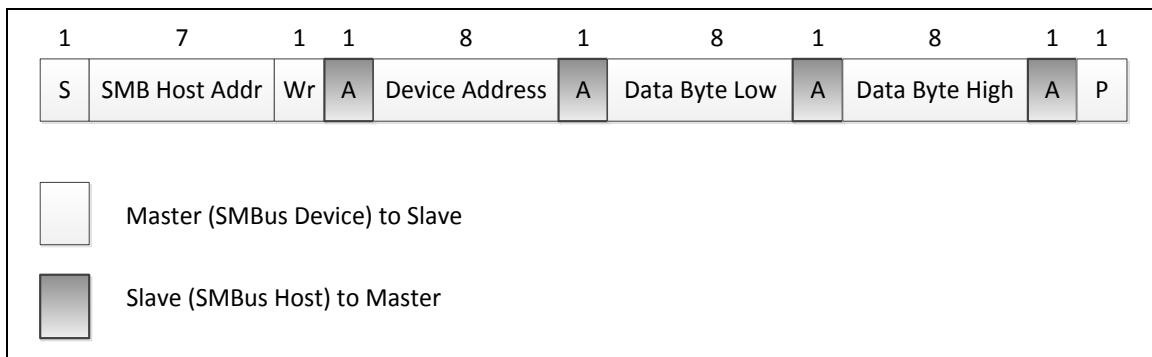


图 6.22-18 7-位可寻址的设备与主机通信

### 总线管理警报

总线管理支持可选警报信号。如果需要，从设备可以通过总线管理警报管脚(GPA[14]/GPE[10])向主机发信号。通过警报响应地址(0b0001 100)，主机可处理中断并可以同时访问所有总线管理警报管脚上的设备。只有拉低总线管理警报管脚的设备才能应答警报响应地址。

当配置成为从设备(BMHEN=0)时，总线管理警报管脚可以通过设置寄存器(I2C\_BUSCTL[4]) ALERTEN位拉低电平，与此同时警报响应地址(ARA)也使能了。

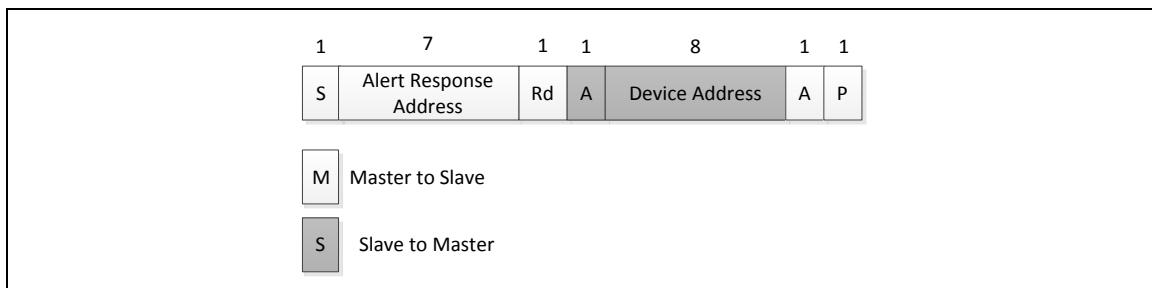


图 6.22-19 7-位可寻址的设备对警报响应地址的响应

配置成为主机(BMHEN=1)时，当总线管理警报管脚的一个下降沿检测到且ALERTEN=1时，警报标志(I2C\_BUSSTS[3])置位。当ALERTEN=0时，即使外部总线管理警报管脚为低电平，警报线还是认为高。如果总线管理警报管脚不需要，ALERTEN = 0时，总线管理警报管脚可用作标准的GPIO口。

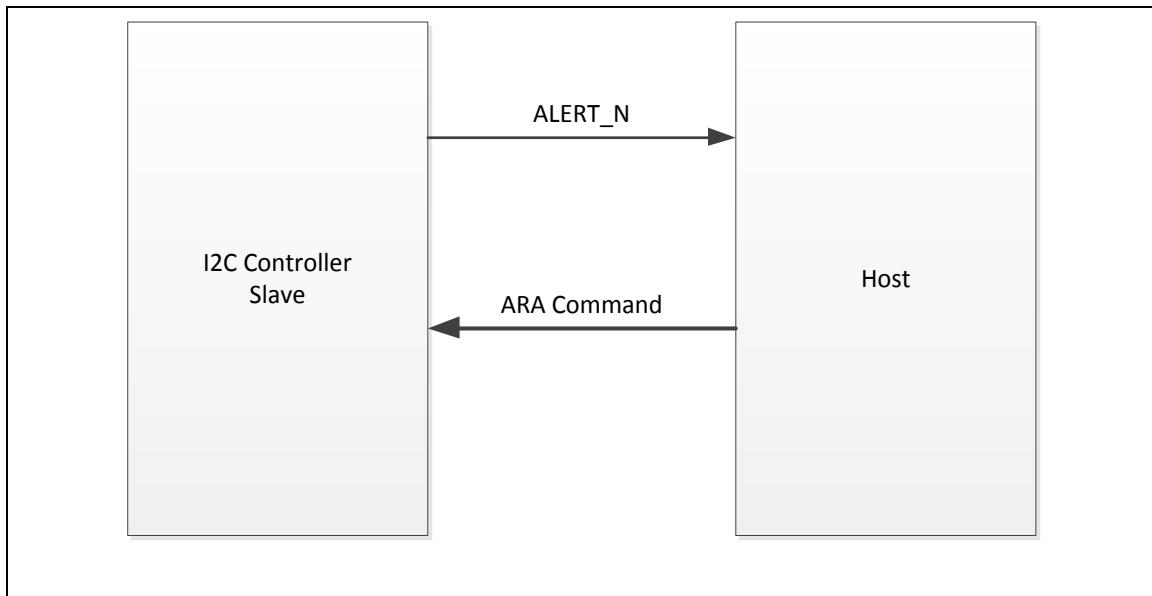


图 6.22-20 总线管理警报功能

### 包错误检查

包错误检查机制有在SMBus规范里介绍，是用来改进通信的可靠性和健壮性。包错误检查通过每次信息传输的最后增加一个错误包编码(PEC)来实现。PEC是通过使用 $C(x) = x^8 + x^2 + x + 1$  CRC-8多项式来对所有信息字节(包括地址和读/写位)来计算。

当PECEN 位(I2C\_BUSCTL[1])使能时，外设内嵌一个硬件PEC计算器，当收到字节数据与硬件计算的PEC值不匹配时，应许自动发送一个无应答信号。PEC计算的值也可以通过I2C\_PKTCRC读回。

### 超时

外设内嵌了硬件计数器为了与SMBus规范2.0版本里的3个超时相兼容。

#### 总线管理超时：

1. 总线忙时 SCLK 低电平超时条件

$$\begin{aligned} T_{\text{Time-out}} &= (\text{BUSTO}(\text{I2C_BUSTOUT}[7:0]) + 1) \times 16 \times 1024 \text{ (14-bit)} \times T_{\text{PCLK}} \text{ (如果 TOCDIV4 = 0).} \\ &= (\text{BUSTO}(\text{I2C_BUSTOUT}[7:0]) + 1) \times 16 \times 1024 \text{ (14-bit)} \times 4 \times T_{\text{PCLK}} \text{ (如果 TOCDIV4 = 1)} \end{aligned}$$

2. 总线空闲条件(SCLK 和 SDA 为高)

$$T_{\text{Time-out}} = (\text{BUSTO}(\text{I2C_BUSTOUT}[7:0]) + 1) \times 4 \times T_{\text{PCLK}}.$$

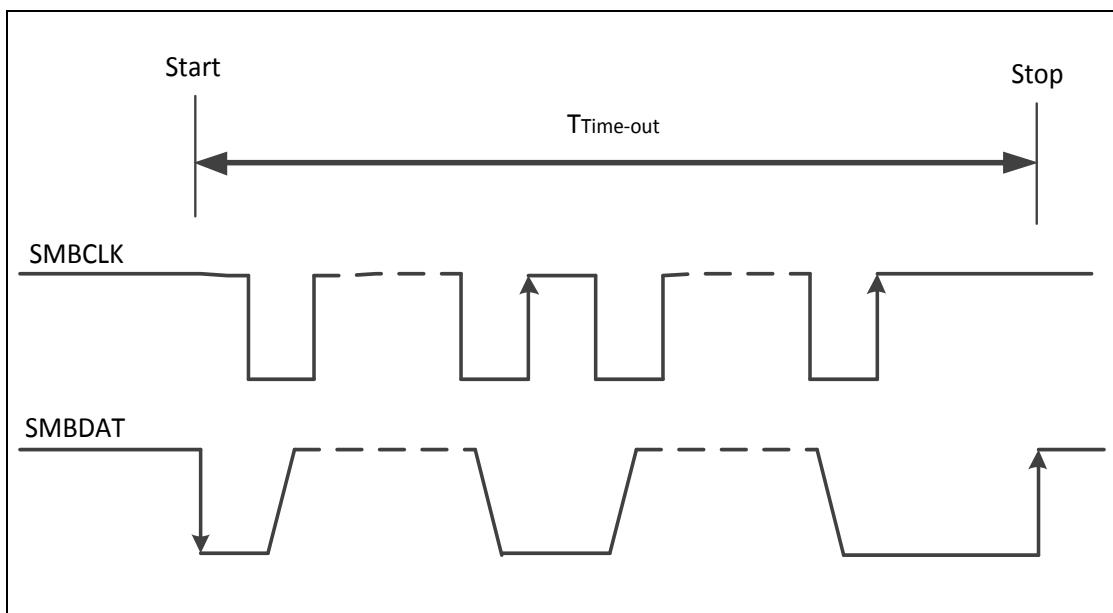


图 6.22-21 系统管理总线超时时序

**总线时钟低电平超时:**

在主机模式, 值测主机低电平延时时间( $T_{LOW:MEXT}$ )

在从机模式, 值测从机低电平延时时间( $T_{LOW:MEXT}$ )

$$T_{LOW:EXT} = (CLKTO (I2C_CLKTOUT[7:0])+1) \times 16 \times 1024 \text{ (14-bit)} \times T_{PCLK} \text{ (如果 TOCDIV4=0).}$$

$$= (CLKTO (I2C_CLKTOUT[7:0])+1) \times 16 \times 1024 \text{ (14-bit)} \times 4 \times T_{PCLK} \text{ (如果 TOCDIV4=1)}$$

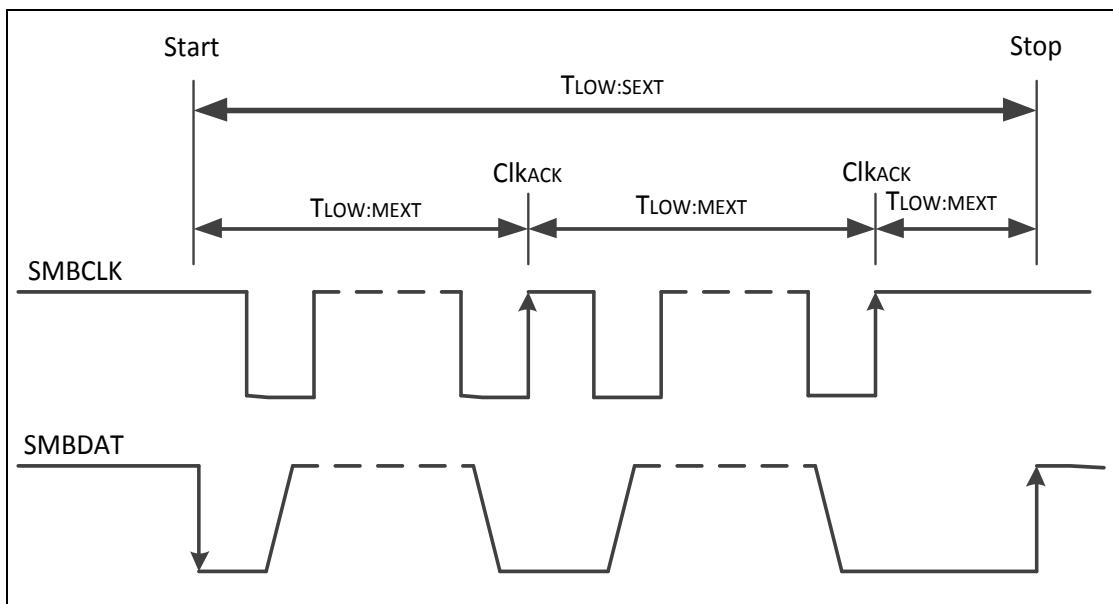


图 6.22-22 总线时钟低电平超时时序

**总线空闲侦测**

如果值测到时钟和数据线上信号高电平时间已经  $T_{IDLE}$  大于  $T_{HIGH,MAX}$  时, 主机可以认为总线已经空

闲。

时序参数包括了一个主机已被动态加入到总线，可能没有检测到SMBCLK和SMBDAT线上状态的变化条件。在这种情况下，主机必须等待足够长的时间，以确保传输不在进行中。外设支持硬件空闲检测。

#### 6.22.4.6 PDMA 传输功能

$I^2C$ 控制器支持PDMA传输功能。当TXPDMAEN (I2C\_CTL1 [0])被置1， $I^2C$ 控制器会产生请求到PDMA控制器使其自动地启动DMA发送进程

当RXPDMAEN (I2C\_CTL1 [1])被置为1时， $I^2C$ 控制器会启动PDMA接收进程。当有数据写入到接收BUFFER时， $I^2C$ 控制器会自动地产生请求到PDMA控制器。

当 $I^2C$ 进入到PDMA模式，大多数状态中断被屏蔽了。不让中断发生除了总线错误或NACK或STOP中断(0x20, 0x30, 0x38, 0x48, 0x58, 0x00, 0xA0, 0xC0, 0x88 和 0x98)

仅在 $I^2C$ 控制器处在主机TX模式下设置PDMASTR (I2C\_CTL1 [8])。如果PDMASTR被清0，在PDMA传输完成和缓存为空之后 $I^2C$ 会自动地发送STOP。如果PDMASTR被置1，在PDMA传输完成和缓存为空之后SI会被置1并且 $I^2C$ 总线会被硬件钳位。

#### 6.22.4.7 可编程的建立和保持时间

为了保证一个正确数据的建立和保持时间，则时序必须可以配置。通过编程HTCTL (I2C\_TMCTL[24:16])来配置保持时间，通过编程STCTL (I2C\_TMCTL[8:0])来配置建立时间

延时时序参考外设时钟(PCLK).当设备拉住主机时钟时，建立和保持时间的配置值不会被影响。

用户需要集中精力在建立和保持时间配置的极限上，时序设置必须遵从 $I^2C$ 协议。一旦建立时间配置大于设计极限，那就意味着如果设置的建立时间让SCL输出小于三个PCLK，由于SCL采样为三次那么 $I^2C$ 控制器就不能正常工作。一旦保持时间配置大于 $I^2C$ 时钟的极限， $I^2C$ 将会发送总线错误。在设置时序前建议用户结合传输速率和协议计算好合理的时序。表 6.22-2 显示了  $I^2C$  传输速率与 PCLK 之间的关系，表格呈现的数是一个时钟间隙包含多少 PCLK。建立和保持时间的配置哪怕在我们设计中编程一些极限值，但是用户必须要遵从 $I^2C$ 的标准协议。

PCLK	$I^2C$ 传输速率	100k	200k	400k	800k	1200k
12MHz	120	60	30	15	10	
24MHz	240	120	60	30	20	
48MHz	480	240	120	60	40	
72MHz	720	360	180	90	60	

表 6.22-2  $I^2C$  传输速率与PCLK之间的关系

对于建立时间错误调整范例，我们假设一个SCL周期包含5个PCLK并且设置STCTL (I2C\_TMCTL[8:0])为3那样对于建立时间设置拉伸三个PCLK。建立时间最大设置值： $ST_{limit} = (I2C_CLKDIV[7:0]+1) \times 2 - 6$ .

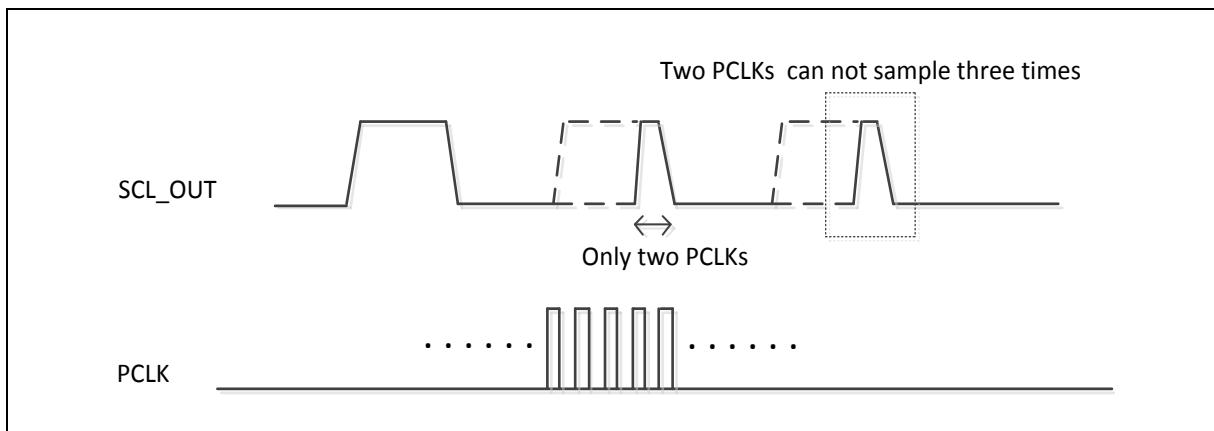


图 6.22-23 建立时间错误调整

对于保持时间错误调整范例，我们使用I<sup>2</sup>C传输速率=1200K和PCLK=72MHz，SCL的高电平/低电平空隙=60PCLK。当我设置HTCTL (I2C\_TMCTL[24:16])为61和STCTL (I2C\_TMCTL[8:0])为0时，然后SDA输出延时将会越过SCL为高的空隙，以至产生总线错误。保持时间最大值的是设定： $HT_{limit} = (I2C\_CLKDIV[7:0]+1) \times 2 - 9$ 。

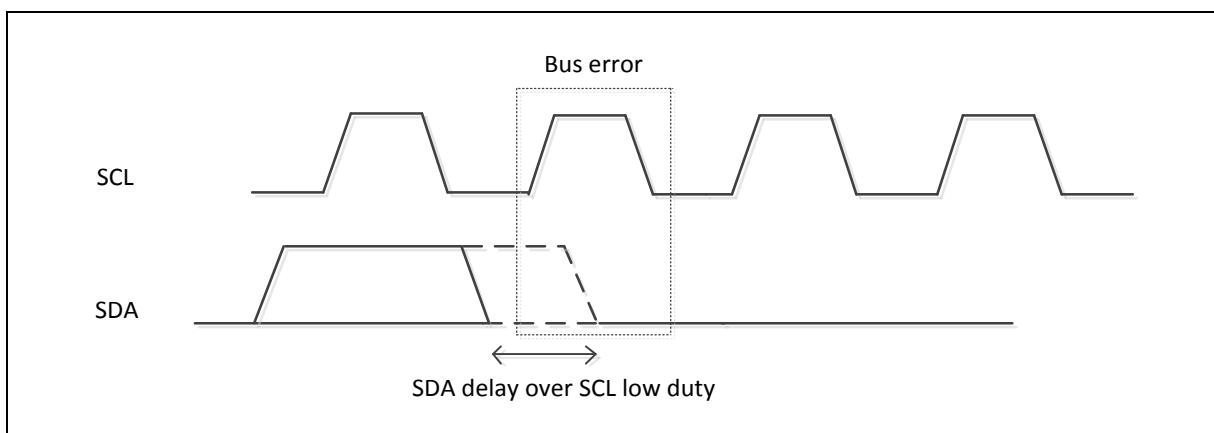


图 6.22-24 保持时间错误调整

#### 6.22.4.8 I<sup>2</sup>C 协议寄存器

通过下列15个特殊功能寄存器来控制I<sup>2</sup>C端口：I2C\_CTL（控制寄存器）I2C\_STATUS（状态寄存器），I2C\_DAT（数据寄存器），I2C\_ADDRn（地址寄存器，n=0~3），I2C\_ADDRMSKn（地址掩码寄存器，n=0~3），I2C\_CLKDIV（时钟速率寄存器），I2C\_TOCTL（超时控制寄存器），I2C\_WKCTL（唤醒控制寄存器）和I2C\_WKSTS（唤醒状态寄存器）

#### 地址寄存器 (I<sup>2</sup>C\_ADDR)

I<sup>2</sup>C端口内建4个从机地址寄存器I2C\_ADDRn (n=0~3)。当I2C作为主机时，这四个寄存器的内容不相关。在从机模式下，位字段ADDR(I2C\_ADDRn [7:1])必须装载芯片自己的从机地址。当I2C\_ADDRn地址与接收到的从机地址符合时I<sup>2</sup>C硬件会作出反应。

I<sup>2</sup>C 端口支持“广播呼叫”功能。当GC位(I2C\_ADDRn [0])被设置，I<sup>2</sup>C端口硬件将应答广播呼叫地址

(00H)。清GC位可禁用“广播呼叫”功能。

当GC位被置且I<sup>2</sup>C处于从机模式时，主机发出广播呼叫地址到I<sup>2</sup>C总线后，从机可以通过地址00H接收广播呼叫地址，然后它将跟随GC模式的状态。

### 从机地址掩码寄存器 (I<sup>2</sup>C\_ADDRMSK)

I<sup>2</sup>C总线控制器带有四个地址掩码寄存器I<sup>2</sup>C\_ADDRMSKn (n=0~3) 支持多地址识别。当地址掩码寄存器被置1，意味着收到的相应地址位是“无关的”。如果置0则收到相应地址位应跟地址寄存器完全一样。

### 数据寄存器 (I<sup>2</sup>C\_DAT)

该寄存器包含一个准备发送或刚接收到的一个字节的串行数据。当不在字节移位处理过程时，CPU可以直接读写访问I<sup>2</sup>C\_DAT[7:0]。当I<sup>2</sup>C处于一个确定的状态并且串行中断标志(SI)被置1，I<sup>2</sup>C\_DAT[7:0]中的数据保持稳定。当数据被移出，总线上的数据同时被移入。I<sup>2</sup>C\_DAT [7:0]的内容总是总线上传递的最后一个字节。

应答位由I<sup>2</sup>C硬件控制，不能被CPU访问。串行数据在SCL线上串行时钟脉冲的上升沿移入I<sup>2</sup>C\_DAT [7:0]。当一个字节被移入I<sup>2</sup>C\_DAT [7:0]，则I<sup>2</sup>C\_DAT [7:0]中的串行数据是可用的，应答位(ACK 或NACK)在第9个时钟脉冲由控制逻辑返回。为了在发送数据时监控总线的状态，当发送I<sup>2</sup>C\_DAT [7:0]到总线时，总线数据将同时被移入I<sup>2</sup>C\_DAT[7:0]。在发送数据过程中，串行数据在SCL时钟脉冲的下降沿从I<sup>2</sup>C\_DAT[7:0]移出，在SCL时钟脉冲的上升沿数据移入I<sup>2</sup>C\_DAT [7:0]。

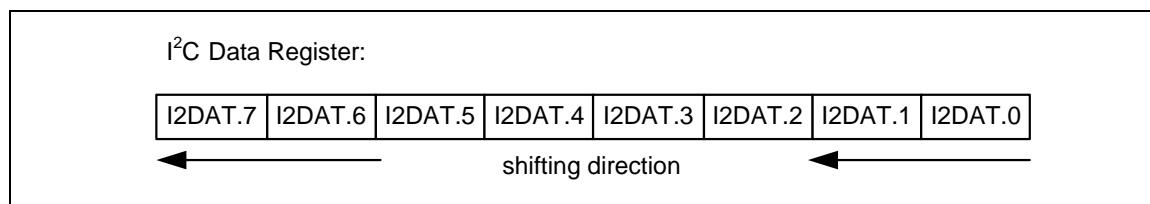


图 6.22-25 I<sup>2</sup>C 数据移位方向

### 控制寄存器0 (I<sup>2</sup>C\_CTL0)

CPU可以直接读或写寄存器I<sup>2</sup>C\_CTL，当I<sup>2</sup>C端口通过设置I2CEN (I<sup>2</sup>C\_CTL0 [6])为高使能，内部状态由I<sup>2</sup>C\_CTL和I<sup>2</sup>C逻辑硬件控制。

有两个位会受硬件影响：当I<sup>2</sup>C硬件产生中断时SI被置位，当总线上产生停止信号时，STO位被清零，当I2CEN =0时STO也会被清零。

一旦有新的状态码产生并存储在I<sup>2</sup>C\_STATUS，I<sup>2</sup>C中断标志位SI(I<sup>2</sup>C\_CTL0 [3])将被自动置位。若此时使能中断INTEN (I<sup>2</sup>C\_CTL0 [7])位被设置，I<sup>2</sup>C中断将产生。I<sup>2</sup>C\_STATUS[7:0]用来存储内部状态码，SI被软件清除前内容一直保持。

**状态寄存器 (I<sup>2</sup>C\_STATUS)**

I2C\_STATUS [7:0] 是一个8位只读寄存器。I2C\_STATUS [7:0]共有26个可能的状态码。所有状态码由下面表所列出。当I2C\_STATUS [7:0] 的值为0xF8时，没有串行中断请求。所有其他的I2C\_STATUS [7:0]的值对应I<sup>2</sup>C 的状态。当进入其中任一状态时，就会产生状态中断请求(SI=1)。在SI 被硬件置位或SI 被软件复位后一个机器周期，有效状态码出现在I2C\_STATUS [7:0] 中。

此外，状态0x00表示总线错误，这会出现在起始或停止条件处在不正确的I<sup>2</sup>C格式帧。总线错误会发生在地址字节、一个数据字节或一个应答位的串行传输。从错误总线恢复时STO应被置位，S应被清除，来进入无地址从机模式。然后STO清除释放总线并等待下一个传输。出现总线错误操作期间I<sup>2</sup>C总线不能识别停止条件。

主机模式		从机模式	
状态	描述	状态	描述
0x08 <sup>[1]</sup>	起始	0xA0	从机发送重新起始或停止
0x10 <sup>[1]</sup>	主机重新起始	0xA8 <sup>[1]</sup>	从机发送地址ACK
0x18 <sup>[1]</sup>	主机发送地址ACK	0xB8 <sup>[1]</sup>	从机发送数据ACK
0x20	主机发送地址NACK	0xC0	从机发送数据NACK
0x28 <sup>[1]</sup>	主机发送数据ACK	0xC8 <sup>[1]</sup>	从机发送最后数据ACK
0x30	主机发送数据NACK	0x60 <sup>[1]</sup>	从机接收地址 ACK
0x38	主机仲裁丢失	0x68 <sup>[1]</sup>	从机接收仲裁丢失
0x40 <sup>[1]</sup>	主机接收地址ACK	0x80 <sup>[1]</sup>	从机接收数据 ACK
0x48	主机接收地址NACK	0x88	从机接收数据NACK
0x50 <sup>[1]</sup>	主机接收数据ACK	0x70 <sup>[1]</sup>	广播模式地址ACK
0x58	主机接收数据NACK	0x78 <sup>[1]</sup>	广播模式仲裁丢失
0x00	总线错误	0x90 <sup>[1]</sup>	广播模式数据 ACK
		0x98	广播模式数据 NACK
		0xB0 <sup>[1]</sup>	从机发送仲裁丢失
0xF0	如果BMDEN =1且ACKMEN位使能，I2C_STATUS的值在从机接收条件时将固定为0xF0.		
0xF8	总线释放 <b>注意：</b> 主/从模式的“0xF8”状态，都不会产生中断 <b>注 [1]:</b> 在 PDMA 模式下没有中断		

表 6.22-3 I<sup>2</sup>C 状态码描述**时钟波特率位(I<sup>2</sup>C\_CLKDIV)**

当I<sup>2</sup>C 在主机模式下，I<sup>2</sup>C数据的波特率由CLK(I2C\_CLKDIV [7:0] )寄存器设定。在从机模式下不需要。在从机模式下，I<sup>2</sup>C 将自动与主机I<sup>2</sup>C设备时钟频率同步。在从机模式下，系统时钟频率应该大于I<sup>2</sup>C总线最大时钟的20倍。

$I^2C$  数据波特率的设定： $I^2C$ 数据波特率= (系统时钟) / (4x ((I2C\_CLKDIV [7:0] +1))。如果系统时钟 = 16 MHz, I2C\_CLKDIV [7:0] = 40 (0x28), 那么 $I^2C$ 数据波特率= 16 MHz/ (4x (40 +1)) = 97.5 Kbits/sec。

### 超时控制寄存器 ( $I^2C\_TOCTL$ )

系统提供一个14 位超时的计数器来处理当 $I^2C$ 总线锁死时的情况。当超时计数器计数功能使能后，计数器开始计数直至溢出(TOIF=1) 并向CPU 产生 $I^2C$ 中断或者清除TOCEN为0停止计数。当超时计数器使能，对SI 标志置高会使计数器复位，再对SI 清零会重新开始计数。如果 $I^2C$ 总线锁死，会使I2C\_STATUS 及SI 标志不再更新，该14-位超时计数器会发生溢出从而产生 $I^2C$ 中断通知CPU。参考下图14-位超时计数器。用户可以写1 清TOIF为0。

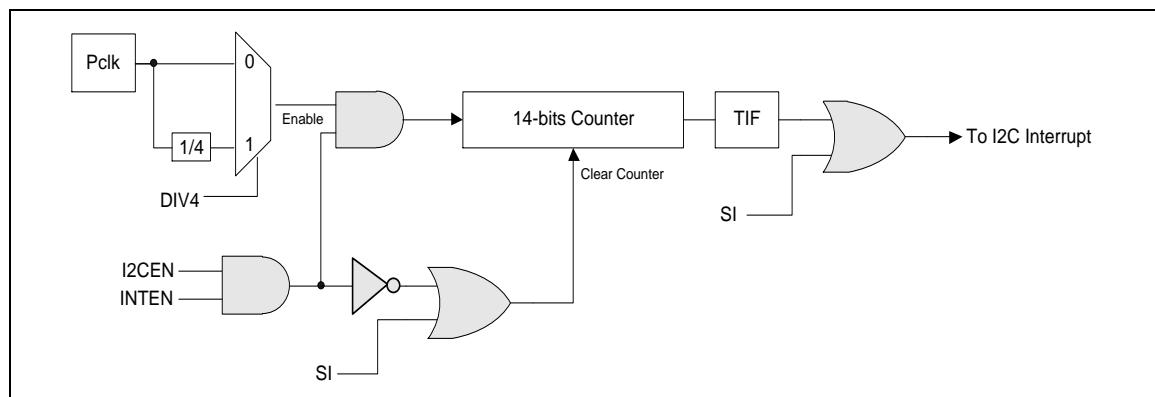


图 6.22-26  $I^2C$  超时计数模块框图

### 唤醒控制寄存器 ( $I^2C\_WKCTL$ )

当芯片进入掉电模式并且WKEN ( $I^2C\_WKCTL [0]$ )被设置为1时，其它 $I^2C$ 主机通过寻址我们 $I^2C$ 设备唤醒我们芯片，在进入睡眠前必须配置相关设置。当地址与设备地址匹配时并且ACK阶段过后，控制器会拉低SCL，然后控制器启动。如果NHDBUSEN ( $I^2C\_WKCTL [7]$ )被设置，控制器将不会拉低SCL。请注意当控制器不能拉低SCL时，发送或接收数据会立即执行。如果发送或是接收数据后SI没有被清除，用户必须复位控制器并且重新执行原先的操作。

### 唤醒状态寄存器 ( $I^2C\_WKSTS$ )

当系统被其他的 $I^2C$ 主机设备唤醒时，WKIF被置位表示该事件发生。用户需要写“1”来清除此位。

当芯片被设备地址寄存器(I2C\_ADDRn)其中之一地址匹配唤醒时，用户需要检查WKAKDONE ( $I^2C\_WKSTS [1]$ )是否被置1来确认地址字节已完成。WKAKDONE位表明在掉电时候ACK阶段已完成。当地址与设备地址匹配时并且ACK阶段过后，控制器会拉低SCL，直到WKAKDONE被用户清除。如果SCL是低频速率并且系统被地址匹配帧唤醒，用户需要检查WKAKDONE来确认该帧已完成传输然后去做唤醒后的动作。请注意用户不能通过清除WKAKDONE为0来释放WKIF。

在控制器发送地址前，WRSTSWK ( $I^2C\_WKSTS [2]$ )位用来记录读/写命令。在系统被地址匹配帧唤醒后，用户读该位的状态来准备下次传输数据(WRSTSWK = 0)或是等待到来的数据(WRSTSWK = 1)及时地被存储。请注意当写1到WKAKDONE ( $I^2C\_WKSTS [1]$ )时，WRSTSWK

(I2C\_WKSTS [2])被清除。

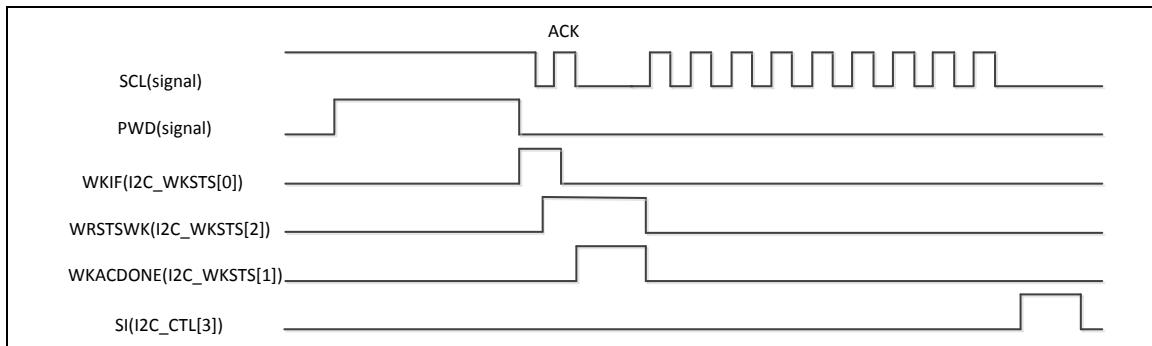


图 6.22-27 I<sup>2</sup>C 唤醒的相关信号波形

### I<sup>2</sup>C 控制寄存器 1 (I<sup>2</sup>C\_CTL1)

如果设置ADDR10EN (I2C\_CTL1 [9])使能10位地址模式，I<sup>2</sup>C将会执行10位模式

对于PDMA功能，设置TXPDMAEN (I2C\_CTL1 [0]) 和 RXPDMAEN (I2C\_CTL1 [1])来启动。通过设置PDMARST (I2C\_CTL1 [2])复位PDMA控制器

### I<sup>2</sup>C 状态寄存器 1 (I<sup>2</sup>C\_STATUS1)

I<sup>2</sup>C 控制器支持四组从机地址标志寄存器，ADMAT0, ADMAT1, ADMAT2 和 ADMAT3 (I2C\_STATUS1[3:0]).每个寄存器代表该地址被使用并且被置1通知软件。

### I<sup>2</sup>C 时序配置控制寄存器 (I<sup>2</sup>C\_TMCTL)

为了配置建立/保证时间，要根据实际需求设置 HTCTL (I2C\_TMCTL[24:16]) 和 STCTL (I2C\_TMCTL[8:0])。

### 总线管理控制器 (I<sup>2</sup>C\_BUSCTL)

SM总线管理控制事件在该寄存器定义。它包括手动控制应答 (ACKMEN (I2C\_BUSCTL[0])), 包错误检测使能(PECEN (I2C\_BUSCTL[1])), 设备默认地址使能 (BMDEN(I2C\_BUSCTL[2])) 或者主机功能使能(BMHEN (I2C\_BUSCTL[3])). 警报和挂起功能可以由 ALERTEN (I2C\_BUSCTL[4]), SCTLOSTS (I2C\_BUSCTL[5])) 和SCTLOEN (I2C\_BUSCTL[6])来设置。

可以通过控制PECTXEN 位 (I2C\_BUSCTL[8])来传送或者接收计算PEC(如果PECEN置位)的值。

该寄存器有个特别的位ACKM9SI (I2C\_BUSCTL[11])。当ACKMEN使能时，在第8个时钟输入时，有SI中断产生，用户此时可以读取数据和状态寄存器。当SI中断清除时第8个时钟总线被释放，此时，如果ACKM9SI位设置为1，在第9个时钟周期时会有另一个SI中断事件发生，这样就知道传输帧完成后的总线状态。

对于 PEC 控制流程则是设置 PECDIEN (I2C\_BUSCTL[13]), BCDIEN (I2C\_BUSCTL[12]) 或

PECCLR (I2C\_BUSCTL[10])

### I<sup>2</sup>C 总线管理定时器控制寄存器 (I<sup>2</sup>C\_BUSTCTL)

设置 TORSTEN (I2C\_BUSTCTL[4]), CLKTOIEN (I2C\_BUSTCTL[3]), BUSTOIEN (I2C\_BUSTCTL[2]), CLKTOEN (I2C\_BUSTCTL[1]) 和 BUSTOEN (I2C\_BUSTCTL[0]) 来配置总线超时或时钟低定时器超时

### I<sup>2</sup>C 总线管理状态寄存器 (I<sup>2</sup>C\_BUSSTS)

对于 PEC 控制流程，需要监控 PECDONE (I2C\_BUSSTS[7]), BCDONE (I2C\_BUSSTS[1]) 或 PECERR (I2C\_BUSSTS[2])

监控SCTLDIN (I2C\_BUSSTS[4]) 来 观察SUSCON 输入状态

### I<sup>2</sup>C 字节数寄存器 (I2C\_PKTSIZE)

当 PECEN 位 (I2C\_BUSCTL[1]) 使能时，I<sup>2</sup>C 控制器将计算总线上数据的 PEC 值。PLDSIZE (I2C\_PKTSIZE[8:0]) 用来定义总线上数据需要传输的个数。当计数值达到 PLDSIZE 时，且 PECTXEN (I2C\_BUSCTL[8]) 有设置，最后的 PEC 值将会发送或者自动接收。

### I<sup>2</sup>C PEC 值寄存器 (I<sup>2</sup>C\_PKTCRC)

该寄存器为 I<sup>2</sup>C 总线上传输数据计算出来的 PECCRC (I2C\_PKTCRC[7:0]) 值。详细信息由在 SM 总线的 PEC 部分说明。

### I<sup>2</sup>C 总线管理定时器和 I<sup>2</sup>C 时钟低定时器寄存器(I<sup>2</sup>C\_BUSTOUT/ I<sup>2</sup>C\_CLKTOUT)

这两寄存器的说明在 SM 总线超时部分由详细的描述。

### EEPROM随机读取范例

通过下面的步骤来配置 I<sup>2</sup>C0 相关寄存器，来使用 I<sup>2</sup>C 从 EEPROM 读取数据。

1. 在 SYS\_GPA\_MFPL 或 SYS\_GPD\_MFPL 或 SYS\_GPE\_MFPH 或 SYS\_GPA\_MFPH 寄存器中将 I<sup>2</sup>C0 多功能管脚设置成 SCL 和 SDA 管脚。
2. 通过设置寄存器 CLK\_APBCLK0 的 I2C0CKEN 为 1 使能 I<sup>2</sup>C0 APB 时钟。
3. 通过设置 I2C0RST = 1 来复位 I<sup>2</sup>C0 控制器，然后通过 I2C0RST= 0 将 I<sup>2</sup>C 控制器设置成普通操作。I2C0RST 位在 SYS\_IPRST1 寄存器中。
4. 通过设置寄存器 I2C\_CTL 中的 I2CEN 为 1 使能 I<sup>2</sup>C0 控制器。
5. 通过在 I2C\_CLKDIV 寄存器写 I<sup>2</sup>C0 时钟分频值。

6. 在“NVIC\_ISER2”寄存器中设置SETENA=0x00000040来设置I<sup>2</sup>C0 IRQ。
  7. 设置寄存器I2C\_CTL的INTEN为1使能I<sup>2</sup>C0中断
  8. 设置I<sup>2</sup>C0地址寄存器“I2C\_ADDR0~I2C\_ADDR3”。

随机读操作是访问EEPROM的其中一种方法。这个方法是允许主机访问EEPROM的任何一个地址。下图显示对EEPROM随机读取操作。

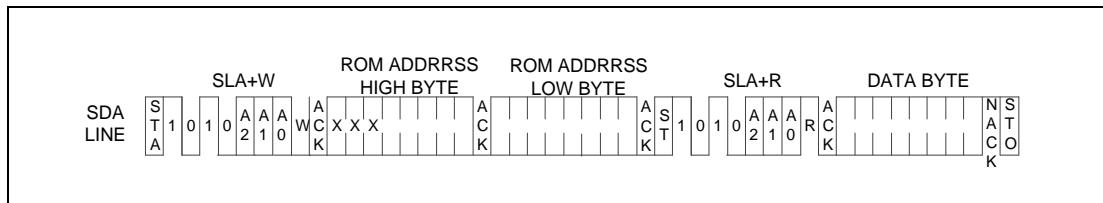


图 6.22-28 EEPROM 随机读取

图 6.22-29 显示怎样使用 I<sup>2</sup>C 控制器执行 EEPROM 随机读取操作协议。

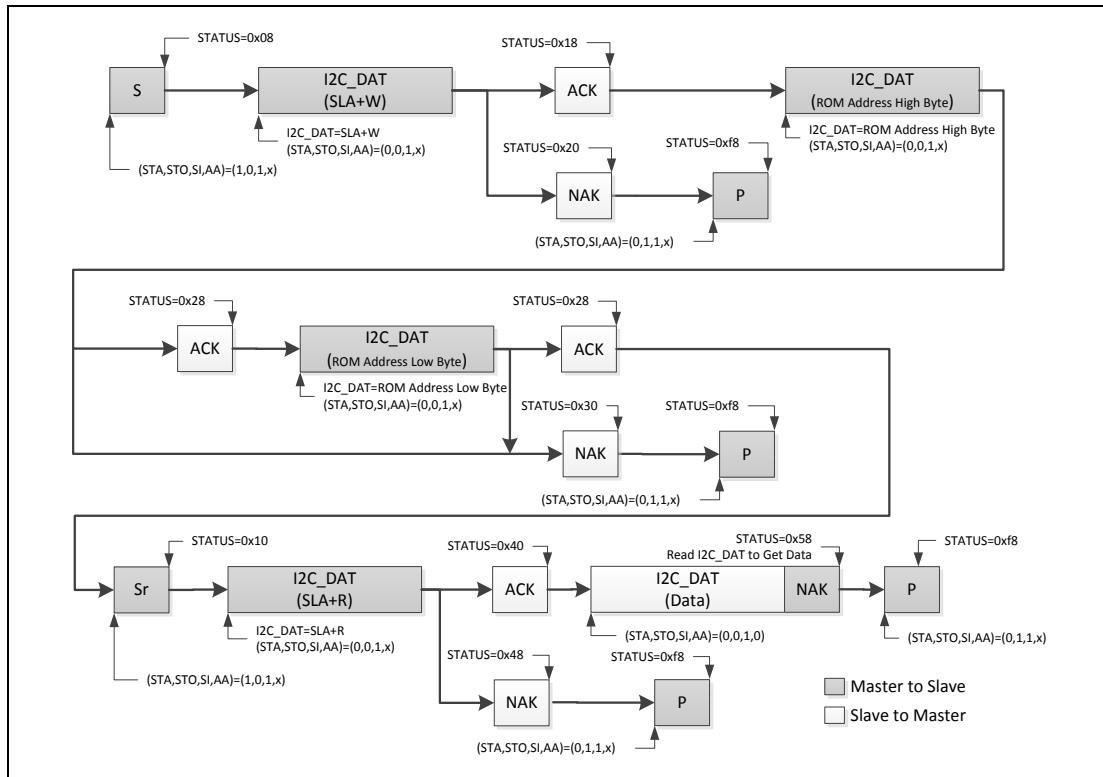


图 6.22-29 随机读取协议

I<sup>2</sup>C控制器作为主机发送起始信号到总线，然后发送SLA+W (从机地址 + 写 位)到EERPOM，跟着由两个字节数据地址来设置EEPROM被读的地址。最后，重复起始信号跟着SLA+R被发送来向EEPROM 读取数据。



### 6.22.6 寄存器映射

R: 只读, W: 只写, R/W: 读/写

寄存器	偏移地址	R/W	描述	复位值
<b>I<sup>2</sup>C 基地址:</b>				
<b>I2Cn_BA = 0x4008_0000 + (0x1000 * n)</b>				
<b>n= 0,1,2</b>				
<b>I2C_CTL0</b>	I2Cn_BA+0x00	R/W	I <sup>2</sup> C控制寄存器0	0x0000_0000
<b>I2C_ADDR0</b>	I2Cn_BA+0x04	R/W	I <sup>2</sup> C从机地址寄存器	0x0000_0000
<b>I2C_DAT</b>	I2Cn_BA+0x08	R/W	I <sup>2</sup> C数据寄存器	0x0000_0000
<b>I2C_STATUS</b>	I2Cn_BA+0x0C	R	I <sup>2</sup> C状态寄存器0	0x0000_00F8
<b>I2C_CLKDIV</b>	I2Cn_BA+0x10	R/W	I <sup>2</sup> C时钟分频寄存器	0x0000_0000
<b>I2C_TOCTL</b>	I2Cn_BA+0x14	R/W	I <sup>2</sup> C超时控制寄存器	0x0000_0000
<b>I2C_ADDR1</b>	I2Cn_BA+0x18	R/W	I <sup>2</sup> C 从机地址寄存器1	0x0000_0000
<b>I2C_ADDR2</b>	I2Cn_BA+0x1C	R/W	I <sup>2</sup> C 从机地址寄存器2	0x0000_0000
<b>I2C_ADDR3</b>	I2Cn_BA+0x20	R/W	I <sup>2</sup> C 从机地址寄存器3	0x0000_0000
<b>I2C_ADDRMSK0</b>	I2Cn_BA+0x24	R/W	I <sup>2</sup> C 从机地址掩码寄存器0	0x0000_0000
<b>I2C_ADDRMSK1</b>	I2Cn_BA+0x28	R/W	I <sup>2</sup> C 从机地址掩码寄存器1	0x0000_0000
<b>I2C_ADDRMSK2</b>	I2Cn_BA+0x2C	R/W	I <sup>2</sup> C 从机地址掩码寄存器2	0x0000_0000
<b>I2C_ADDRMSK3</b>	I2Cn_BA+0x30	R/W	I <sup>2</sup> C 从机地址掩码寄存器3	0x0000_0000
<b>I2C_WKCTL</b>	I2Cn_BA+0x3C	R/W	I <sup>2</sup> C 唤醒控制寄存器	0x0000_0000
<b>I2C_WKSTS</b>	I2Cn_BA+0x40	R/W	I <sup>2</sup> C 唤醒状态寄存器	0x0000_0000
<b>I2C_CTL1</b>	I2Cn_BA+0x44	R/W	I <sup>2</sup> C 控制寄存器1	0x0000_0000
<b>I2C_STATUS1</b>	I2Cn_BA+0x48	R/W	I <sup>2</sup> C 状态寄存器1	0x0000_0000
<b>I2C_TMCTL</b>	I2Cn_BA+0x4C	R/W	I <sup>2</sup> C 时序配置控制寄存器	0x0000_0000
<b>I2C_BUSCTL</b>	I2Cn_BA+0x50	R/W	I <sup>2</sup> C 总线管理控制寄存器	0x0000_0000
<b>I2C_BUSTCTL</b>	I2Cn_BA+0x54	R/W	I <sup>2</sup> C 总线管理定时器控制寄存器	0x0000_0000
<b>I2C_BUSSTS</b>	I2Cn_BA+0x58	R/W	I <sup>2</sup> C 总线管理状态寄存器	0x0000_0000
<b>I2C_PKTSIZE</b>	I2Cn_BA+0x5C	R/W	I <sup>2</sup> C 包错误检查字节数寄存器	0x0000_0000
<b>I2C_PKTCRC</b>	I2Cn_BA+0x60	R	I <sup>2</sup> C 包错误检查字节值寄存器	0x0000_0000
<b>I2C_BUSTOUT</b>	I2Cn_BA+0x64	R/W	I <sup>2</sup> C 总线管理定时器寄存器	0x0000_0005
<b>I2C_CLKTOUT</b>	I2Cn_BA+0x68	R/W	I <sup>2</sup> C 总线管理时钟低定时器寄存器	0x0000_0005

### 6.22.7 寄存器描述

**I2C\_CTL0 I<sup>2</sup>C控制寄存器0**

寄存器	偏移地址	R/W	描述	复位值
I2C_CTL0	I2Cn_BA+0x00	R/W	I <sup>2</sup> C控制寄存器0	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
INTEN	I2CEN	STA	STO	SI	AA	Reserved	

位	描述	
[31:8]	Reserved	保留
[7]	INTEN	<b>使能中断</b> 0 = 禁用 I <sup>2</sup> C 中断 1 = 使能 I <sup>2</sup> C 中断
[6]	I2CEN	<b>I<sup>2</sup>C 控制器使能位</b> 设置使能 I <sup>2</sup> C 串行功能控制器。当I2CEN =1, I <sup>2</sup> C 串行功能使能。SDA和SCL对应的多功能管脚功能必须先设置为 I <sup>2</sup> C 功能。 0 = I2C控制器禁用 1 = I2C控制器使能
[5]	STA	<b>I<sup>2</sup>C 起始控制位</b> 设置 STA 为 1, 进入主机模式, 如果总线处于空闲状态, I <sup>2</sup> C 硬件会送出 START 或 重复 START 条件。
[4]	STO	<b>I<sup>2</sup>C 停止控制位</b> 在主机模式, 设置 STO 来传送一个 STOP 条件到总线, 然后 I <sup>2</sup> C 硬件将会检查总线状况, 如果检测到一个 STOP 状况, 这个标志会被硬件自动清除。
[3]	SI	<b>I<sup>2</sup>C 中断标志</b> 当一个新的 I <sup>2</sup> C 状态出现在寄存器 I2C_STATUS 时, SI 标志由硬件置位, 并且如果INTEN (I2C_CTL [7])位被置位, 则产生 I <sup>2</sup> C 中断请求。SI 必须由软件通过向该位写 '1' 清零。 如果在从机读模式且ACKMEN置位, SI标志在第8个时钟周期置位, 此时用户确认应答位, 在第9个时钟周期时, 用户可以从数据缓冲器中读取数据。
[2]	AA	<b>应答控制位</b> 当 AA=1 先于地址或数据接收, 在SCL线上的应答时钟脉冲期间将返回一个应答 (SDA上为低电平), 有两种情况: 1.) 从机正在应答主机发送的地址。2.) 接收设备正在应答发送设备发送的数据。当 AA = 0 先于地址或数据接收, 则在SCL线上的应答时钟脉冲期间将返回一个非应答 (SDA上为高电平)。
[1:0]	Reserved	保留



**I2C DAT I<sup>2</sup>C数据寄存器**

寄存器	偏移地址	R/W	描述	复位值
I2C_DAT	I2Cn_BA+0x08	R/W	I <sup>2</sup> C数据寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
DAT							

位	描述	
[31:8]	Reserved	保留
[7:0]	DAT	I <sup>2</sup> C 数据位 该区域 8 位存放 I <sup>2</sup> C 串行端口8位传输/接收数据

**I2C STATUS I<sup>2</sup>C状态寄存器**

寄存器	偏移地址	R/W	描述	复位值
I2C_STATUS	I2Cn_BA+0x0C	R	I <sup>2</sup> C状态寄存器0	0x0000_00F8

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
STATUS							

位	描述	
[31:8]	Reserved	保留
[7:0]	STATUS	<p><b>I<sup>2</sup>C 状态位</b></p> <p>低3位有效位一直为0。高5位有效位包括状态码。有28种可能的状态码。当I2C_STATUS的内容为0xF8时，没有中断产生。所有其他的 I2C_STATUS的值对应 I<sup>2</sup>C 的状态。当进入其中任一状态时，就会产生状态中断请求 (SI=1)。在 SI 被硬件置位或 SI 被软件复位后一个机器周期，有效状态码出现在 I2C_STATUS中。</p> <p>此外，00H 状态表示总线错误。总线错误发生在 START 或 STOP 条件出现在帧结构不正确的位置。不正确的位比如是在串行传输地址字节、数据字节或应答位期间。</p>

**I2C CLKDIV I<sup>2</sup>C时钟分频寄存器**

寄存器	偏移地址	R/W	描述	复位值
I2C_CLKDIV	I2Cn_BA+0x10	R/W	I <sup>2</sup> C时钟分频寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved						DIVIDER	
7	6	5	4	3	2	1	0
DIVIDER							

位	描述	
[31:10]	Reserved	保留
[9:0]	DIVIDER	<b>I<sup>2</sup>C 时钟分频位</b> I <sup>2</sup> C 数据波特率 = (系统时钟) / (4x (I2C_CLKDIV +1)). <b>注意:</b> I2C_CLKDIV的最小值是 4.

**I<sup>2</sup>C TOCTL I<sup>2</sup>C 超时控制寄存器**

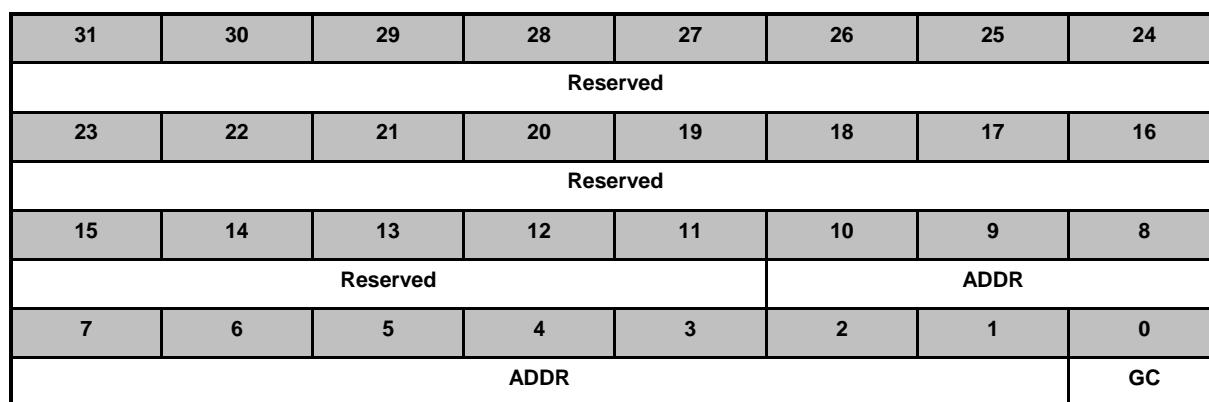
寄存器	偏移地址	R/W	描述	复位值
I <sup>2</sup> C_TOCTL	I <sup>2</sup> Cn_BA+0x14	R/W	I <sup>2</sup> C超时控制器寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved					TOCEN	TOCDIV4	TOIF

位	描述	
[31:3]	<b>Reserved</b>	保留
[2]	<b>TOCEN</b>	<p>超时计数器使能位 0 = 超时计数器禁用 1 = 超时计数器使能 当使能该位，则14-位溢出定时器将会在 SI 清零后开始计数。设置 SI位为高将会复位计数器，在 SI位清零之后，计数器会重新开始计数。</p>
[1]	<b>TOCDIV4</b>	<p>溢出定时器输入时钟除以 4 0 = 溢出定时器输入时钟禁用 1 = 溢出定时器输入时钟使能 该位使能后，超时时间扩大 4 倍。</p>
[0]	<b>TOIF</b>	<p>超时溢出标志 当超时发生时，该位由 硬件 置位，如果此时 I<sup>2</sup>C 的中断使能位INTEN置为1，则可引发CPU 的中断。 注意：写 1 清除该位。</p>

**ADDRx I<sup>2</sup>C从机地址寄存器**

寄存器	偏移地址	R/W	描述	复位值
I2C_ADDR0	I2Cn_BA+0x04	R/W	I <sup>2</sup> C 从机地址寄存器0	0x0000_0000
I2C_ADDR1	I2Cn_BA+0x18	R/W	I <sup>2</sup> C 从机地址寄存器1	0x0000_0000
I2C_ADDR2	I2Cn_BA+0x1C	R/W	I <sup>2</sup> C 从机地址寄存器2	0x0000_0000
I2C_ADDR3	I2Cn_BA+0x20	R/W	I <sup>2</sup> C 从机地址寄存器3	0x0000_0000



位	描述	
[31:11]	Reserved	保留
[10:1]	ADDR	<b>I<sup>2</sup>C 地址</b> 主机模式下，该寄存器内容没有意义。从机模式下，如果地址符合，I <sup>2</sup> C 硬件将会自动应答。 <b>注:</b> 没用到地址应该设置为0
[0]	GC	<b>广播呼叫功能</b> 0 = 广播呼叫功能禁用 1 = 广播呼叫功能使能 <b>注:</b> 当ADDR10EN (I2C_CTL1 [9]) 置1时，不要设置该位。

**ADDRMSKx I<sup>2</sup>C从机地址掩码寄存器**

寄存器	偏移地址	R/W	描述	复位值
I <sup>2</sup> C_ADDRMSK0	I2Cn_BA+0x24	R/W	I <sup>2</sup> C 从机地址掩码寄存器0	0x0000_0000
I <sup>2</sup> C_ADDRMSK1	I2Cn_BA+0x28	R/W	I <sup>2</sup> C 从机地址掩码寄存器1	0x0000_0000
I <sup>2</sup> C_ADDRMSK2	I2Cn_BA+0x2C	R/W	I <sup>2</sup> C 从机地址掩码寄存器2	0x0000_0000
I <sup>2</sup> C_ADDRMSK3	I2Cn_BA+0x30	R/W	I <sup>2</sup> C 从机地址掩码寄存器3	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved					ADDRMSK		
7	6	5	4	3	2	1	0
ADDRMSK							Reserved

位	描述	
[31:11]	Reserved	保留
[10:1]	ADDRMSK	<p><b>I<sup>2</sup>C 地址掩码位</b>            0 = 掩码禁用（接收到的相应地址必须完全符合地址寄存器）            1 = 掩码使能（接收到的相应地址位不予辨识）</p> <p>I<sup>2</sup>C 总线控制器有4个地址掩码寄存器，支持多地址识别。当地址掩码寄存器的某位被置‘1’，表示接收到的地址的相应位可忽略。如果该位被置‘0’，则表示接收到的地址的相应位必须和地址寄存器中的完全一致。</p> <p><b>注:</b>唤醒功能不能使用地址掩码</p>
[0]	Reserved	保留

**I<sup>2</sup>C WKCTL I<sup>2</sup>C唤醒控制寄存器**

寄存器	偏移地址	R/W	描述	复位值
I <sup>2</sup> C_WKCTL	I <sup>2</sup> Cn_BA+0x3C	R/W	I <sup>2</sup> C唤醒控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
NHDBUSEN	Reserved						WKEN

位	描述	
[31:8]	Reserved	保留
[7]	NHDBUSEN	<p>I<sup>2</sup>C总线保持禁用位 0 = 唤醒后I<sup>2</sup>C总线保持 1 = 唤醒后禁用I<sup>2</sup>C总线保持 <b>注:</b> 当WKIF事件没有被清除时I<sup>2</sup>C控制器会产生响应, 这样会导致数据发送或接收出错。如果数据发送或接收时WKIF事件没有清除, 用户必须复位I<sup>2</sup>C控制器并且重新执行原先的操作。</p>
[6:1]	Reserved	保留
[0]	WKEN	<p>I<sup>2</sup>C 唤醒功能使能位 0 = I<sup>2</sup>C 唤醒功能禁用 1 = I<sup>2</sup>C 唤醒功能使能</p>

**I2C WKSTS I<sup>2</sup>C唤醒状态寄存器**

寄存器	偏移地址	R/W	描述	复位值
I2C_WKSTS	I2Cn_BA+0x40	R/W	I <sup>2</sup> C唤醒状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved					WRSTSWK	WKAKDONE	WKIF

位	描述	
[31:3]	<b>Reserved</b>	保留
[2]	<b>WRSTSWK</b>	<b>地址唤醒帧中读/写 状态位</b> 0 =在地址匹配唤醒帧中写命令被记录 1 =在地址匹配唤醒帧中读命令被记录 <b>注:</b> 当软件写1到WKAKDONE时该位被清除。
[1]	<b>WKAKDONE</b>	<b>唤醒地址帧应答位完成</b> 0 =地址匹配帧ACK阶段没有完成 1 =地址匹配帧ACK阶段在掉电时已完成 <b>注:</b> 该位不能释放WKIF。该位软件写1清0。
[0]	<b>WKIF</b>	<b>I<sup>2</sup>C 唤醒标志</b> 当芯片从掉电模式下由I <sup>2</sup> C唤醒时， 该位设置为1， ,该位软件写1清0。

**I<sup>2</sup>C CTL1 I<sup>2</sup>C 控制寄存器 1**

寄存器	偏移地址	R/W	描述	复位值
I <sup>2</sup> C_CTL1	I <sup>2</sup> Cn_BA+0x44	R/W	I <sup>2</sup> C 控制 1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved						ADDR10EN	PDMASTR
7	6	5	4	3	2	1	0
Reserved						PDMARST	RXPDMAEN
Reserved						TXPDMAEN	

位	描述	
[31:8]	<b>Reserved</b>	保留
[9]	<b>ADDR10EN</b>	<b>10位地址功能使能</b> 0 =禁用10位地址匹配功能 1 =使能10位地址匹配功能
[8]	<b>PDMASTR</b>	<b>PDMA 锯位</b> 0 =在PDMA传输完成后 I <sup>2</sup> C 自动发送STOP (仅主机 TX) 1 =在PDMA传输完成后如果SI没有被清除, I <sup>2</sup> C SCL总线会被硬件拉住(仅主机 TX)
[7:3]	<b>Reserved</b>	保留
[2]	<b>PDMARST</b>	<b>PDMA 复位</b> 0 = 无影响 1 = I <sup>2</sup> C产生复位请求到PDMA
[1]	<b>RXPDMAEN</b>	<b>PDMA 接收通道有效</b> 0 =禁用PDMA接收功能 1 =使能PDMA接收功能
[0]	<b>TXPDMAEN</b>	<b>PDMA发送通道有效</b> 0 =禁用PDMA发送功能 1 =使能PDMA发送功能

**I<sup>2</sup>C STATUS1 I<sup>2</sup>C 状态寄存器 1**

寄存器	偏移地址	R/W	描述	复位值
I <sup>2</sup> C_STATUS1	I <sup>2</sup> Cn_BA+0x48	R/W	I <sup>2</sup> C 状态寄存器 1	0x0000_0000

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

位	描述	
[31:8]	<b>Reserved</b>	保留
[8]	<b>ONBUSY</b>	<b>总线正忙</b> 表明在总线上在处理一个通信。当起始信号被侦测到时该位被硬件置位。当停止信号被侦测到时该位被硬件清除 0 = 总线处在空闲状态(SCLK 和 SDA 都为高). 1 = 总线忙 <b>注:</b> 该位只读
[7:0]	<b>Reserved</b>	保留

**I<sup>2</sup>C TMCTL I<sup>2</sup>C 时序配置控制寄存器**

寄存器	偏移地址	R/W	描述	复位值
I <sup>2</sup> C_TMCTL	I <sup>2</sup> Cn_BA+0x4C	R/W	I <sup>2</sup> C时序配置控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							HTCTL
23	22	21	20	19	18	17	16
HTCTL							
15	14	13	12	11	10	9	8
Reserved							STCTL
7	6	5	4	3	2	1	0
STCTL							

位	描述	
[31:25]	<b>Reserved</b>	保留
[24:16]	<b>HTCTL</b>	<b>保持时间配置控制寄存器</b> 该域是发送模式下用来在SCL下降沿和SDA上升沿之间产生延时时序。 延时保持时间外设时钟数= HTCTL x PCLK.
[15:9]	<b>Reserved</b>	保留
[8:0]	<b>STCTL</b>	<b>建立时间配置控制寄存器</b> 该域是发送模式下用来在SDA下降沿和SCL上升沿之间产生延时时序。 延时建立时间外设时钟数= STCTL x PCLK. <b>注:</b> 建立时间设置不能让SCL输出少于三个PCLKs

I<sup>2</sup>C BUSCTL I<sup>2</sup>C总线管理控制寄存器

寄存器	偏移地址	R/W	描述	复位值
I <sup>2</sup> C_BUSCTL	I <sup>2</sup> Cn_BA+0x50	R/W	I <sup>2</sup> C总线管理控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved		PECDIEN	BCDIEN	ACKM9SI	PECCLR	TIDLE	PECTXEN
7	6	5	4	3	2	1	0
BUSEN	SCTLOEN	SCTLOSTS	ALERTEN	BMHEN	BMDEN	PECEN	ACKMEN

位	描述	
[31:14]	Reserved	保留
[13]	PECDIEN	包错误检测字节传输完成中断使能位 0 = 禁用PEC传输完成中断 1 = 使能PEC传输完成中断 注:该位在 PECEN =1时使用
[12]	BCDIEN	包错误检测字节计数完成中断使能位 0 = 禁用计数完成中断 1 = 使能计数完成中断 注:该位在 PECEN =1时使用
[11]	ACKM9SI	手工应答使能额外SI中断位 0 = 当BUSEN =1 和 ACKMEN =1时, 在第9个时钟周期没有SI中断 1 = 当BUSEN =1 和 ACKMEN =1时, 在第9个时钟周期有SI中断
[10]	PECCLR	在重新起始时清除PEC 当PECEN设置为1时, PEC的计算开始。当检测到STA或STO位时清除。当发生REPEAT START 时, PECCR位用来使能清除PEC计算。 0 = 发生“Repeat Start”功能时禁止清除PEC计算 1 = 发生“Repeat Start”功能时使能清除PEC计算
[9]	TIDLE	空闲状态时间检查 BUSTOUT用来计算有效总线上时钟低的时间和空闲总线上空闲的周期。该位用来定义使能哪个条件。 0 = BUSTOUT用来计算有效总线上时钟低的时间。 1 = BUSTOUT用来计算空闲总线上空闲的周期。 注意: BUSY (I <sup>2</sup> C_BUSSTS[0])位表示当前总线的状态。
[8]	PECTXEN	包错误检查字节发送/接收使能位

		该位由软件设置, 当PEC传送或者一个停止条件或者收到匹配的地址由硬件清除。 0 = 没有 PEC传送. 1 = 请求PEC 发送/接收. <b>注意:</b> 当ACKMEN =0时, 在从机模式该位无影响.
[7]	<b>BUSEN</b>	<b>总线使能位</b> 0 = 禁止系统管理功能. 1 = 使能系统管理功能. <b>注意:</b> 当该位使能,内部14-位计数器用来计算时钟低条件的事件时间。
[6]	<b>SCTLOEN</b>	<b>挂起或控制管脚输出使能位</b> 0 = SUSCON管脚输入 1 = 输出使能SUSCON管脚有效
[5]	<b>SCTLOSTS</b>	<b>挂起/控制数据输出状态</b> 0 = SUSCON 管脚输出为低. 1 = SUSCON 管脚输出为高.
[4]	<b>ALERTEN</b>	<b>总线管脚警报使能位</b> 从模式(BMHEN =0). 0 = 释放BM_ALERT管脚为高而且禁止警报响应地址头: 如果BMDEN 和 ACKMEN都使能, NACK后接着就是0001100x。 1 = 驱动BM_ALERT管脚为低而且使能警报响应地址头: 如果BMDEN 和 ACKMEN都使能, ACK后接着就是0001100x。 主模式(BMHEN =1). 0 = BM_ALERT 管脚不支持 1 = BM_ALERT 管脚支持.
[3]	<b>BMHEN</b>	<b>总线管理主机使能位</b> 0 = 禁止主机功能 1 = 使能主机功能
[2]	<b>BMDEN</b>	<b>总线管理设备默认地址使能位</b> 0 = 禁止设备默认地址。当检测地址0'b1100001x 且BMDEN 和 ACKMEN都使能, 设备响应NACK。 1 = 使能设备默认地址。当检测地址0'b1100001x 且BMDEN 和 ACKMEN都使能, 设备响应ACK。
[1]	<b>PECEN</b>	<b>包错误检查计算使能位</b> 0 = 禁止包错误检查计算 1 = 使能包错误检查计算 <b>注:</b> 当I <sup>2</sup> C进入掉电模式, 在唤醒之后如果需要PEC计算该位应该使能
[0]	<b>ACKMEN</b>	<b>手动应答控制使能位</b> 为了从机接收包括命令和数据应许ACK控制, 从机字节控制模式必须通过设置ACKMEN位使能。 0 = 禁止从机字节控制. 1 = 使能从机字节控制。用户通过接收的数据在第9位可以响应ACK或NACK。当字节被接收时, 第8个和第9个SCLK脉冲之间的SCLK低信号拉长。 <b>注意:</b> 如果 BMDEN =1 且 该位使能, 在从接收模式下I2C_STATUS 的信息将固定在0xF0 。

**I2C\_BUSTCTL I<sup>2</sup>C总线管理定时器控制寄存器**

寄存器	偏移地址	R/W	描述	复位值
I2C_BUSTCTL	I2Cn_BA+0x54	R/W	I <sup>2</sup> C总线管理定时器控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved			TORSTEN	CLKTOIEN	BUSTOIEN	CLKTOEN	BUSTOEN

位	描述	
[31:5]	Reserved	保留
[4]	TORSTEN	<p>超时复位使能位 0 = 禁用I<sup>2</sup>C状态机复位 1 = 使能I<sup>2</sup>C状态机复位(时钟和数据线将被放为高)</p>
[3]	CLKTOIEN	<p>延长时钟超时中断使能位 0 = 禁止长时间中断. 1 = 使能长时间中断</p>
[2]	BUSTOIEN	<p>超时中断使能位 BUSY =1. 0 = 禁止SCLK 低超时中断. 1 = 使能SCLK 低超时中断. BUSY =0. 0 = 禁止总线空闲超时中断 1 = 使能总线空闲超时中断</p>
[1]	CLKTOEN	<p>累积时钟低超时使能位 0 = 禁止累积时钟低电平超时检测. 1 = 使能累积时钟低电平超时检测. 主机, 计算从START到ACK周期 从机, 计算从START 到STOP周期</p>
[0]	BUSTOEN	<p>总线超时使能位 0 = 禁止总线时钟低电平超时检测 1 = 使能总线时钟低电平超时检测(总线时钟为低电平大于T时间 (BIDLE=0) 或者 高电平大于T时间 (BIDLE =1).</p>

I<sup>2</sup>C BUSSTS I<sup>2</sup>C 总线管理状态寄存器

寄存器	偏移地址	R/W	描述	复位值
I <sup>2</sup> C_BUSSTS	I <sup>2</sup> Cn_BA+0x58	R/W	I <sup>2</sup> C总线管理状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
PECDONE	CLKTO	BUSTO	SCTLDIN	ALERT	PECERR	BCDONE	BUSY

位	描述	
[31:6]	<b>Reserved</b>	保留
[7]	<b>PECDONE</b>	<b>PEC 字节发送/接收完成</b> 0 = PECEN置位时表明PEC发送/接收还未完成 1 = PECEN置位时表明PEC发送/接收已完成 <b>注:</b> 该位软件写1清0.
[6]	<b>CLKTO</b>	<b>时钟累积低电平超时状态</b> 0 = 未发生 时钟累积低电平超时 1 = 有发生 时钟累积低电平超时 <b>注:</b> 该位软件写1清0.
[5]	<b>BUSTO</b>	<b>总线超时状态</b> 0 = 未发生超时或未发生外部时钟超时 1 = 有发生超时或有发生外部时钟超时 在总线忙时该位指示时钟低电平总数发生超时事件，别的方式就指示总线空闲超时事件发生。 <b>注:</b> 该位软件写1清0.
[4]	<b>SCTLDIN</b>	<b>总线挂起或控制信号输入状态</b> 0 = SUSCON管脚输入状态为0 1 = SUSCON管脚输入状态为1
[3]	<b>ALERT</b>	<b>SMBus 警报状态</b> 从模式 (BMHEN =0). 0 = 指示SMASSERT管脚状态为低 1 = 指示SMASSERT管脚状态为高 主模式 (BMHEN =1). 0 = 无SMBALERT事件.

		<p>1 = 当BMHEN = 1(SMBus主机配置)和ALERTEN=1时，指示在SMASSERT管脚上有检测到SMBALERT事件(下降沿)发生。</p> <p><b>注意:</b> 1. SMASSERT管脚是开漏管脚,使用时需要上拉电阻。2. 该位软件写1清0.</p>
[2]	<b>PECERR</b>	<p><b>接收PEC错误状态位</b></p> <p>0 = 指示PEC值等于接收到PEC数据包。</p> <p>1 = 指示PEC值与接收到PEC数据包不匹配。</p> <p><b>注:</b> 该位软件写1清0</p>
[1]	<b>BCDONE</b>	<p><b>字节数传送/接收完成状态位</b></p> <p>0 = 当PECEN置位时，传送/ 接收还没有完成。</p> <p>1 = 当PECEN置位时，传送/ 接收已经完成。</p> <p><b>注:</b> 该位软件写1清0.</p>
[0]	<b>BUSY</b>	<p><b>总线忙状态位</b></p> <p>指示总线上通信的进程。当检测到起始条件时该位由硬件置位。当检测到停止条件时由硬件清除。</p> <p>0 = 总线处于空闲状态(SCLK 和 SDA 都为高).</p> <p>1 = 总线处于忙状态.</p>

**I<sup>2</sup>C\_PKTSIZE I<sup>2</sup>C 字节数寄存器**

寄存器	偏移地址	R/W	描述	复位值
I <sup>2</sup> C_PKTSIZE	I <sup>2</sup> Cn_BA+0x5C	R/W	I <sup>2</sup> C字节数寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
PLDSIZE							

位	描述	
[31:9]	<b>Reserved</b>	保留
[8:0]	<b>PLDSIZE</b>	<p><b>传输字节个数</b> 当PECEN置位时，指示一次事务中传送或接收数据的字节个数。最大传送或接受字节个数为256个字节。</p> <p><b>注：</b>字节计数包括地址、命令码和数据帧</p>

**I<sup>2</sup>C PKTCRC I<sup>2</sup>C PEC 值寄存器**

寄存器	偏移地址	R/W	描述	复位值
I <sup>2</sup> C_PKTCRC	I <sup>2</sup> Cn_BA+0x60	R	I <sup>2</sup> C包错误检查字节数值寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
PECCRC							

位	描述	
[31:8]	Reserved	保留
[7:0]	PECCRC	<b>包错误检查字节值</b> 该字节指示在传送或接收字节数据后，包错误检查内容通过使用 $C(x) = X8 + X2 + X + 1$ 计数。该位域只读。

**I<sup>2</sup>C 总线管理定时器寄存器 (I2C\_BUSTOUT)**

寄存器	偏移地址	R/W	描述	复位值
I2C_BUSTOUT	I2Cn_BA+0x64	R/W	I <sup>2</sup> C总线管理定时器寄存器	0x0000_0005

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
BUSTO							

位	描述	
[31:8]	<b>Reserved</b>	保留
[7:0]	<b>BUSTO</b>	<b>总线管理超时时间值</b> 该位域指示总线空闲或时钟低电平超时时间值 <b>注:</b> 如果用户想修改BUSTOUT的值, 先使能BUSEN(I2C_BUSCTL[7]), 然后把TORSTEN(I2C_BUSTCTL[4])位必须设置为1, 接着再清零。

**I2C CLKTOUT I<sup>2</sup>C 时钟低电平定时器寄存器**

寄存器	偏移地址	R/W	描述	复位值
I2C_CLKTOUT	I2Cn_BA+0x68	R/W	I <sup>2</sup> C总线管理时钟低电平定时器寄存器	0x0000_0005

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
CLKTO							

位	描述	
[31:8]	Reserved	保留
[7:0]	CLKTO	<p>总线时钟低电平时间 该位域用来配置累积时钟延长时间 <b>注:</b> 如果用户想要修改CLKLOUT的值, 先使能BUSEN, 然后把TORSTEN需设置为1, 接着再清零。</p>

## 6.23 USCI – 通用串行接口

### 6.23.1 概述

通用串行控制接口(USCI)是一种灵活的接口模块，它集中了几种串行通信协议。用户可以配置该控制器作为UART、SPI或是I<sup>2</sup>C功能协议。

### 6.23.2 特性

控制器可以依据应用需求进行单独的配置。主要支持以下几种协议：

- UART
- SPI
- I<sup>2</sup>C

### 6.23.3 框图

USCI的基本配置如下：

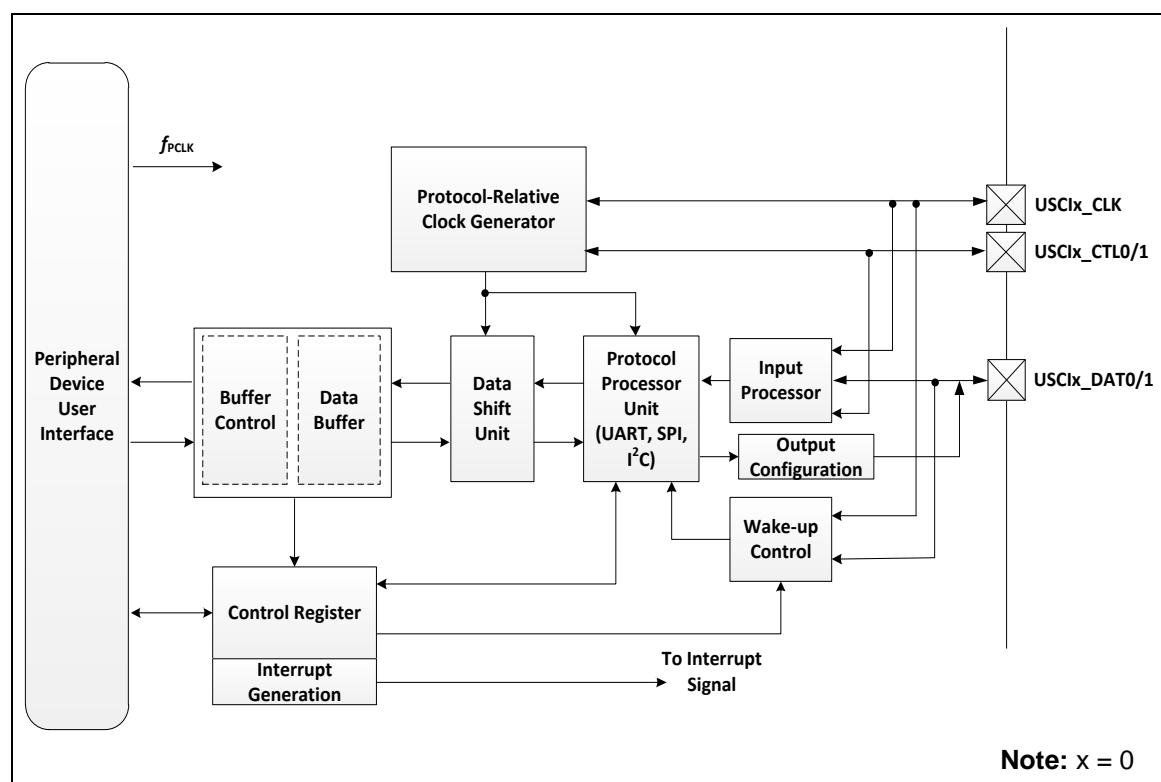


图 6.23-1 框图

#### 6.23.4 功能描述

USCI控制器的结构如图 6.23-1 输入信号被输入处理器处理。数据缓冲和数据移位单元支持数据传输。每一个具体协议功能被协议处理单元处理。时序和具体协议的时间事件控制信号被协议相关时钟发生器处理。所有具体协议事件被中断产生单元处理。具体协议唤醒功能被唤醒控制器单元处理。

USCI具有三种协议分别是: UART, SPI, 和 I<sup>2</sup>C.可以通过FUNMODE (USCI\_CTL [2:0])选择。在改变协议前FUNMODE必须设置为0.

##### 6.23.4.1 IO 处理

## 输入信号

所有输入阶段提供类似特性集合。可以用于所有协议。

表 6.23-1 为每种选择的协议列出了相关的输入信号。每种输入信号由输入处理器处理加工。比如信号反相选择控制或数字输入滤波

选择协议		UART	SPI	I <sup>2</sup> C
串行总线时钟输入	USCIx_CLK	-	SPI_CLK	SCL
控制输入	USCIx_CTL0	nCTS	SPI_SS	-
	USCIx_CTL1	-	-	-
数据输入	USCIx_DAT0	RX	SPI_MOSI_0	SDA
	USCIx_DAT1	-	SPI_MISO_0	-

表 6.23-1 不同协议的输入信号

具体协议的描述见相关的协议章节。

### 一般输入结构

数据输入结构和控制信号包含反相器、数字滤波器和边沿检测（仅数据信号）。

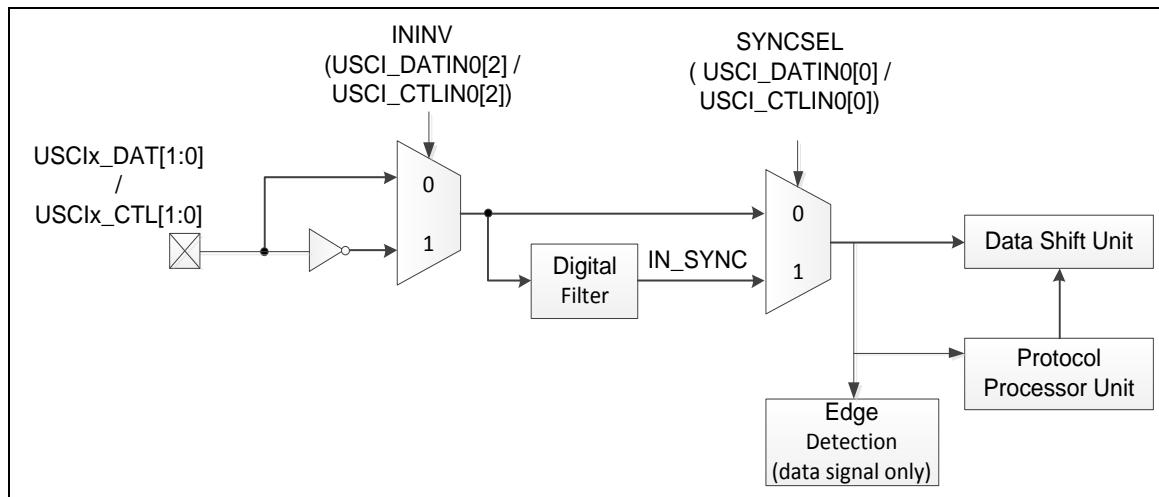


图 6.23-2 输入调节USCIx\_DAT[1:0] 和 USCIx\_CTL[1:0]

USCIx\_CLK输入结构类似于USCIx\_CTL[1:0]输入结构，除了不支持反相功能

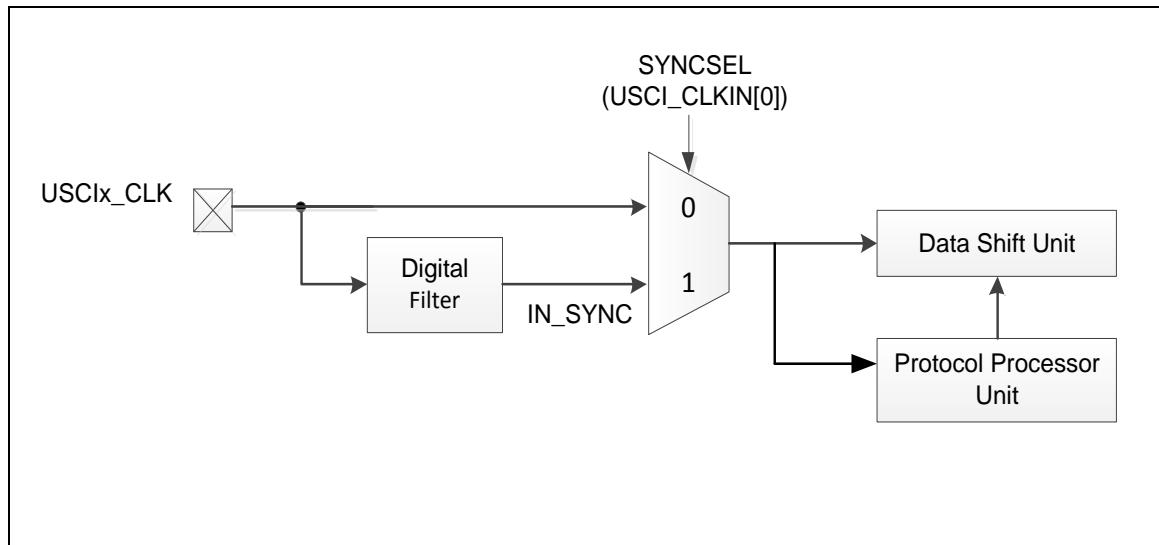


图 6.23-3 输入调节 USCIx\_CLK

控制、时钟和数据输入结构的配置分别在USCI\_CTLIN0, USCI\_CLKIN 和 USCI\_DATIN0寄存器中。EDGEDET (USCI\_DATINO[4:3]) 被用来选择边沿检测条件。在UART模式EDGEDET (USCI\_DATINO[4:3])必须设置为2'b10。可编程的边沿检测表明预期的事件被激活的触发信号触发了。

ININV (USCI\_DATINO[2] / USCI\_CTLINO[2])允许选择的输入信号被极性反转来适应输入信号到数据移位单元的内部极性和协议状态机

如果SYNCSEL (USCI\_DATINO[0] / USCI\_CTLINO[0] / USCI\_CLKIN[0])被设置为0，由于同步和过滤，输入信号的路径不会包含任何延时。如果信号中有噪声，有可能到同步信号（信号IN\_SYNC

同步到 $f_{PCLK}$ ) SYNCSEL = 1时同步输入信号延时需要考虑进去。同步导致2-3倍的 $f_{PCLK}$ 周期延时。

输出信号

表 6.23-2 为每种选择的协议列出了相关的输出信号。实际使用的输出数目依据所选的协议以及依据协议的意义来分类。

选择协议		UART	SPI	I <sup>2</sup> C
串行总线时钟输出	USCIx_CLK	-	SPI_CLK	SCL
控制输出	USCIx_CTL0	-	SPI_SS	-
	USCIx_CTL1	nRTS	-	-
数据输出	USCIx_DAT0	-	SPI_MOSI_0	SDA
	USCIx_DAT1	TX	SPI_MISO_0	-

表 6.23-2 不同协议输出信号

具体协议的描述见相关的协议章节。

#### 6.23.4.2 数据缓冲

USCI控制器的数据处理是建立在数据移位单元(DSU)和缓冲结构基础上。数据移位和缓存寄存器都是16位的宽度。数据移位单元输入包含移位数据、串行总线时钟和移位控制。传送输出USCIx\_DAT0管脚或USCIx\_DAT1管脚取决于协议的选择。

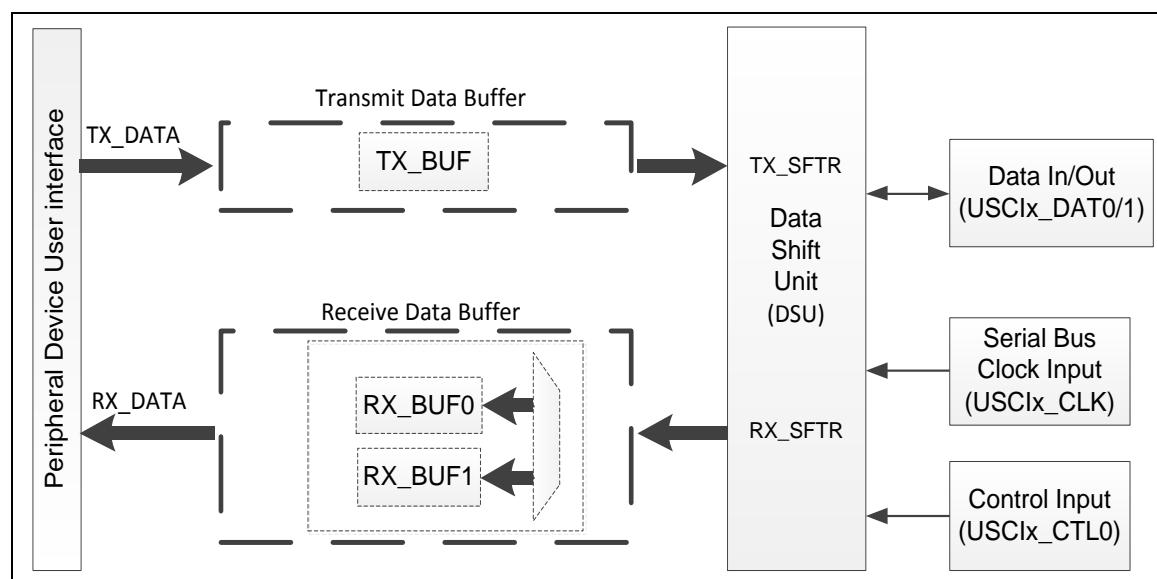


图 6.23-4 数据缓冲框图

数据处理操作包含：

- 外设用户接口(APB)被用作处理数据、中断、状态和控制信息。
- 发送器包含发送移位寄存器(TX\_SFTR)和发送数据缓冲(TX\_BUF). TXFULL / TXEMPTY (USCI\_BUFSTS[9:8]) 和 TXENDIF (USCI\_PROTSTS[2])被用来指示发送的状态。
- 接收器包含接收移位寄存器(RX\_SFTR)和双缓冲接收结构(RX\_BUF0, RX\_BUF1).。在双缓冲结构，用户不用关心接收顺序，如果用户没有及时读取USCI\_RXDAT寄存器的数据，接收到两次数据会被保持住。

### 数据访问结构

数据访问结构包括对接收数据读访问和对传送数据写访问。接收到的数据被保存在接收缓冲RX\_BUFO 和 RX\_BUF1中。用户不用关心接收顺序。通过读USCI\_RXDAT寄存器来访问接收缓冲区。先接收到数据被先读出，然后在USCI\_RXDAT可以查看下一个接收到的数据，可以再一次被读出。

通过写传送寄存器USCI\_TXDAT来把发送数据加载到TX\_BUF

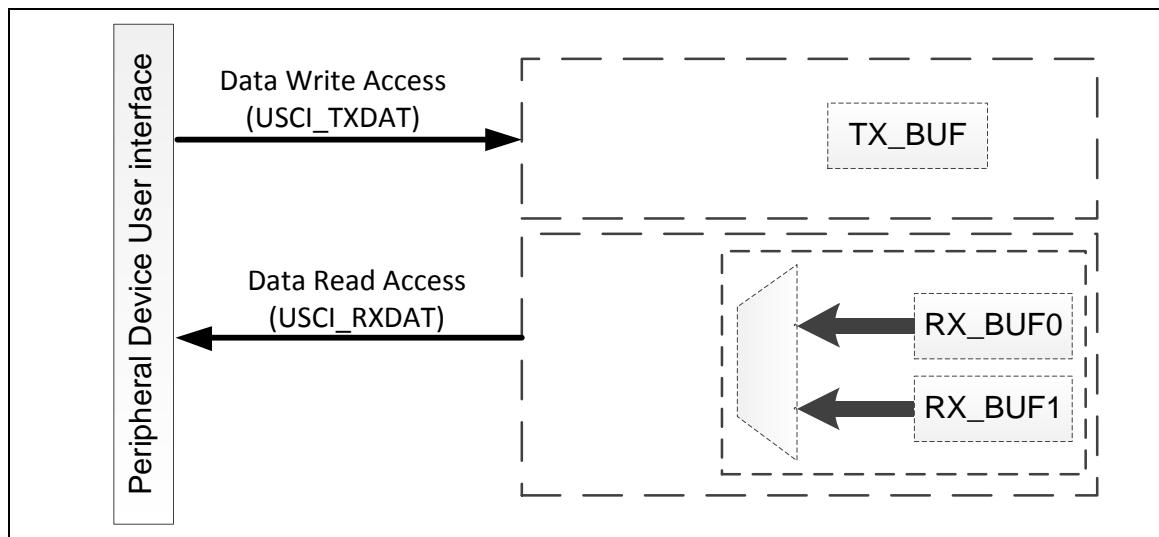


图 6.23-5 数据访问结构

### 传送数据路径

传送数据路径建立在16位宽度的传送移位寄存器(TX\_SFTR)和传送缓冲TX\_BUF。数据传输参数比如数据字长是通常是通过线控制寄存器USCI\_LINECTL来控制发送和接收的。

### 传送缓冲

传送移位寄存器不能被用户直接访问。在数值存到发送缓冲TX\_BUF时会被自动的更新。如果当前传送数据完成则新数据将会被传送。

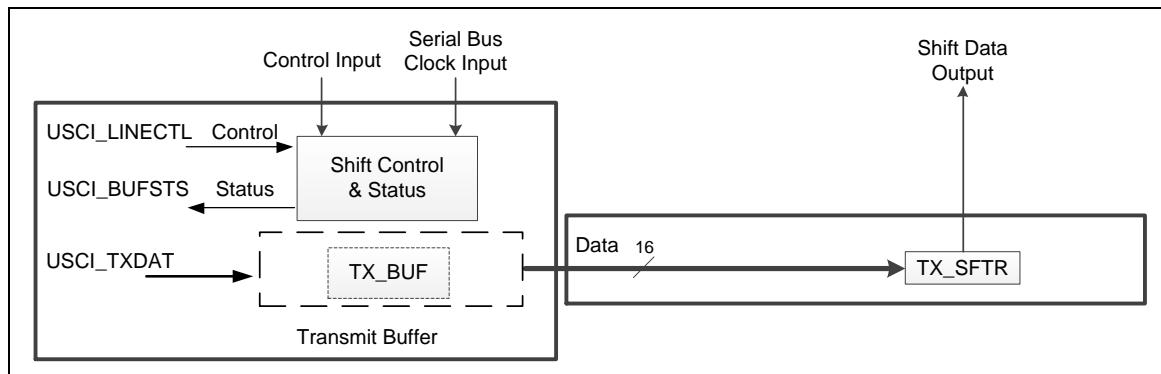


图 6.23-6 传送数据路径

### 传送数据的有效性

**TXEMPTY (USCI\_BUFSTS[8])** 的状态表明在传送缓冲(TX\_BUF)中传送数据是有效的，还是没有准备好。 TXSTIF (USCI\_PROTSTS[1]) 在每次数据的启动过程时指示。

- 如果 USCI 控制器是主机，数据传输只能将有效数据载入到传送缓冲(TX\_BUF)来启动。在这种情况下，传送移位寄存器会加载传送缓冲的内容。

**注:** 数据传输启动由主机决定

- 如果 USCI 控制器是从机，由主机请求数据传输，从机的启动必须独立于传送缓冲(TX\_BUF)的状态。如果一次数据的传输由主机发起，传送移位寄存器需依据具体协议控制信号加载有效的数据。

**注:** 从机不能自己启动但是必须有回应

- 从传送缓冲到数据移位单元加载数据时序取决于协议的配置。

**UART:** 在正常操作下，如果 TXEMPTY = 0，在缓冲中数据字被传送。在自动流控下，当 TXEMPTY = 0 和 USCIx\_CTL0 在有效状态时在缓冲中数据字的传送可以启动。

**SPI:** 在主机模式下，当 TXEMPTY (USCI\_BUFSTS[8]) 为 0 时数据传送将会启动。在从机模式下，当从机选择信号处在激活状态和 USCIx\_CLK 引脚出现时钟信号时，数据传送将会启动。

**I<sup>2</sup>C:** 如果 TXEMPTY = 0，在缓冲中数据字被传送。

- 如果 TXEMPTY (USCI\_BUFSTS [8]) = 0 在缓冲中传输数据会被传送。当一次有效的新的传送启动时，传送缓冲的内容(TX\_BUF)不能被新的数据重新写入。如果 TX\_BUF 的内容必须改变，在更新数据前用户必须设置 TXRST (USCI\_BUFCTL [16]) 为 1 去清除 TX\_BUF 的内容。而且当传送缓冲(TX\_BUF)被新数据更新时，TXEMPTY (USCI\_BUFSTS [8]) 会被自动的清除。当一次传送正在进行时，TX\_BUF 可以加载新的数据。用户必须在新的传送前更新 TX\_BUF

### 接收数据路径

接收数据路径建立在16位宽度的接收移位寄存器(RX\_SFTR)和传送缓冲RX\_BUF0 和 RX\_BUF1。数据传输参数比如数据字长或移位方向通常是通过线控制寄存器USCI\_LINECTL来控制发送和接收的。寄存器USCI\_BUFSTS监控USCI\_RXDAT数据的有效性。

## 接收缓冲

接收移位寄存器不能被用户直接访问。但是如果一个完整数据字被接收或是帧已完整会被自动地载入到接收缓冲中。缓冲的接收数据字可以通过寄存器USCI\_RXDAT被读出。

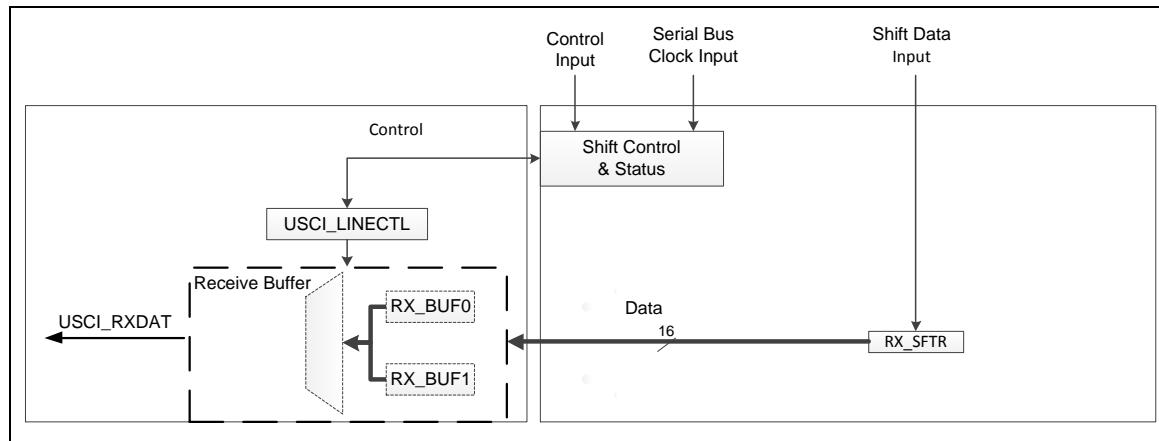


图 6.23-7 接收数据路径

### 6.23.4.3 端口方向控制

SPI协议是半双工配置，数据端口是双向的。端口方向控制通过专用的硬件接口来控制引脚方向。

通过PORTDIR (USCI\_TXDAT[16])来控制选中引脚的方向。当用户写USCI\_TXDAT寄存器时，传送数据和它的端口方向同时被操作。

### 6.23.4.4 协议控制和状态

相关协议的控制和状态信息在协议控制寄存器USCI\_PROTCTL和协议状态寄存器USCI\_PROTSTS。这些寄存器之间共享可用的协议。因此，这些寄存器每个位的意义在不同协议间会有所不同。详细信息参考每个协议的相关寄存器。

### 6.23.4.5 相关协议的时钟发生器

USCI控制器包含相关协议时钟发生器，由寄存器USCI\_BRGEN控制。当USCI\_BRGEN寄存器被写入时会产生复位。相关协议时钟发生器结构如下所示。

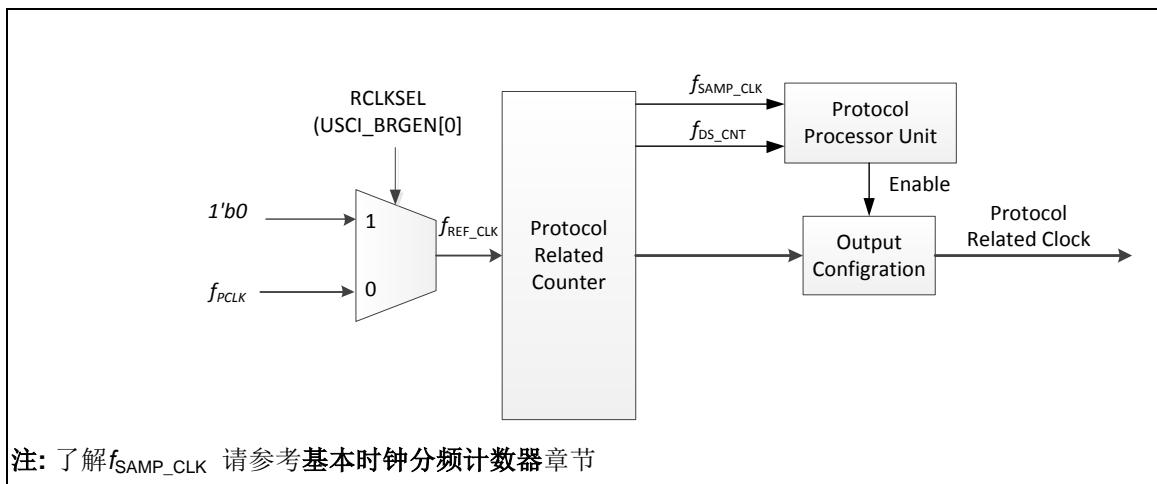


图 6.23-8 相关协议时钟发生器

协议相关的计数器包含基本时钟分频计数器和时序测量计数器。图是建立在分频阶段，为不同协议提供所需的频率。主要包含：

- 基本时钟分频计数器提供协议的相关时钟信号和其它相关协议信号( $f_{SAMP\_CLK}$  和  $f_{DS\_CLK}$ ).
- 时序测量计数器提供时间间隔测量，比如，UART协议的波特率侦测。
- 相关协议时钟发生器输出信号在管脚上可获得（比如SPI的USCIx\_CLK）

### 基本时钟分频计数器

基本时钟分频计数器是来自对  $f_{REF\_CLK2}$ ,  $f_{REF\_CLK}$ ,  $f_{DIV\_CLK}$ ,  $f_{SCLK}$  和  $f_{SAMP\_CLK}$  的分频。这些分频器的频率由 PTCLKSEL (USCI\_BRGEN [1]), CLKDIV (USCI\_BRGEN [25:16]), SPCLKSEL (USCI\_BRGEN [3:2]) 控制。

基本时钟分频计数器用来生成相关协议时序信号

$$f_{DIV\_CLK} = f_{REF\_CLK} \times \frac{1}{CLKDIV + 1} \text{ 如果 } PTCLKSEL = 0$$

$$f_{DIV\_CLK} = f_{REF\_CLK} \times \frac{1}{(CLKDIV + 1) \times 2} \text{ 如果 } PTCLKSEL = 1$$

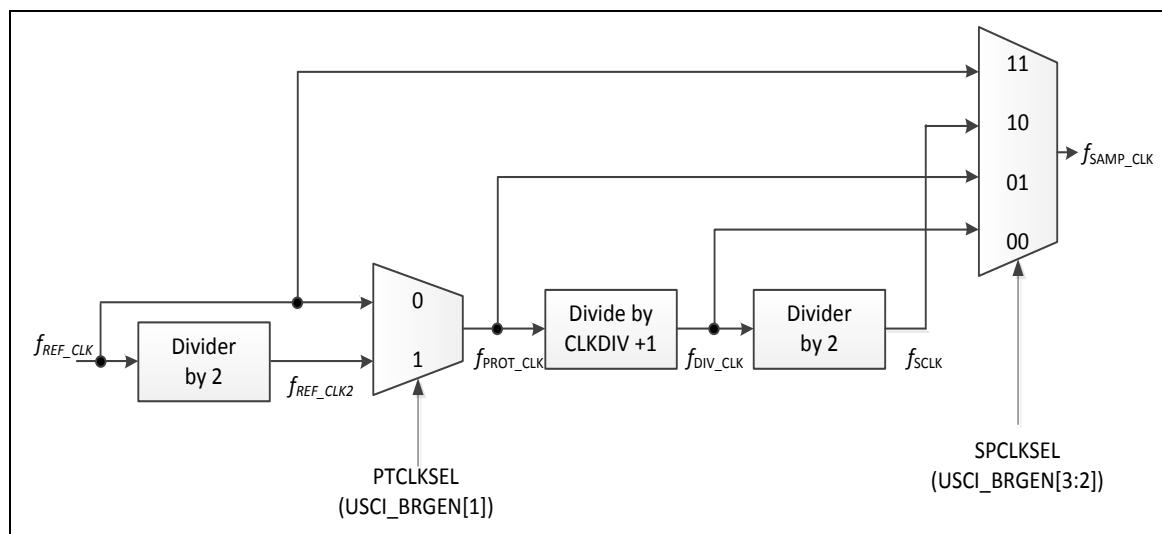


图 6.23-9 基本时钟分频计数器

### 时序测量计数器

时序测量计数器是用来对时间间隔的测量，通过TMCNTEN (USCI\_BRGEN [4]) = 1使能。当TMCNTSRC (USCI\_BRGEN [5])被置1，定时器时钟来自 $f_{DIV\_CLK}$ ，否则来自 $f_{PROT\_CLK}$ 。因此在串行数据接收和发送的同时定时器可以执行时序测量。定时器计数的相关协议信号来自 $f_{PROT\_CLK}$ 或 $f_{DIV\_CLK}$ 。当它达到用户指定的值时会停止计数。

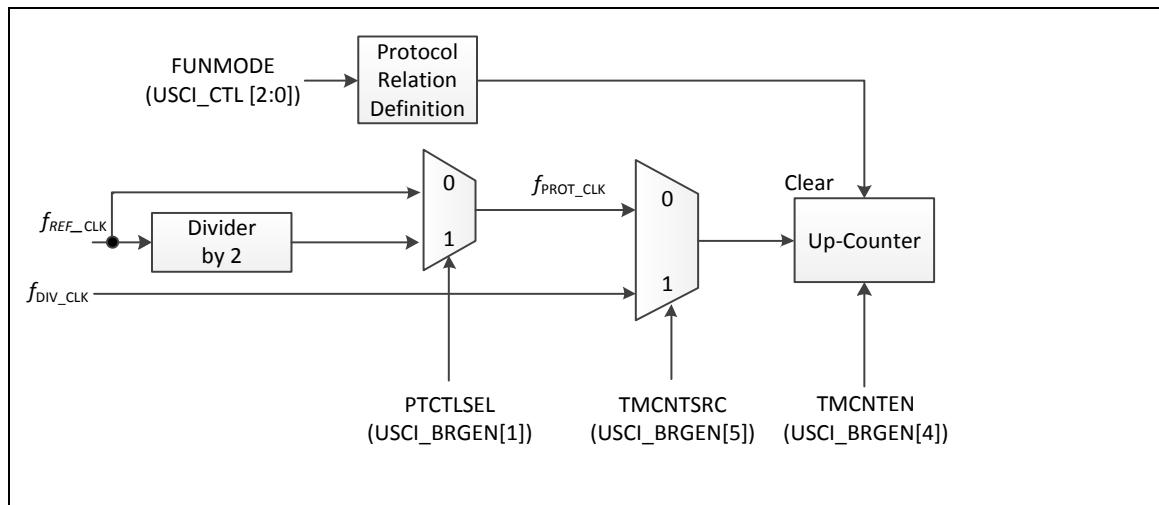


图 6.23-10 时序测量计数器框图

时序测量计数器是用来执行超时功能或自动波特率机制。该功能取决于所选用的协议如下所说。

- **UART:**时序测量计数器被用作自动波特率侦测
- **SPI:** 时序测量计数器被用作从机超时周期计数
- **I<sup>2</sup>C:**时序测量计数器被用作超时时钟周期

## 采样时间计数器

采样时间计数器与决定协议时序的协议相关计数器是有关联的。比如移位控制信号或位时序是建立在输入频率 $f_{SAMP\_CLK}$ 基础上的。采样时间计数器位具体的协议产生时间间隔。采样频率 $f_{PDS\_CNT}$ 周期是通过选择输入频率 $f_{SAMP\_CLK}$ 和可编程与分频值(PDSCNT (USCI\_BRGEN [9:8]))给定的。这就意味着采样时间取决于所选的协议。详细协议信息请参考相关的章节。

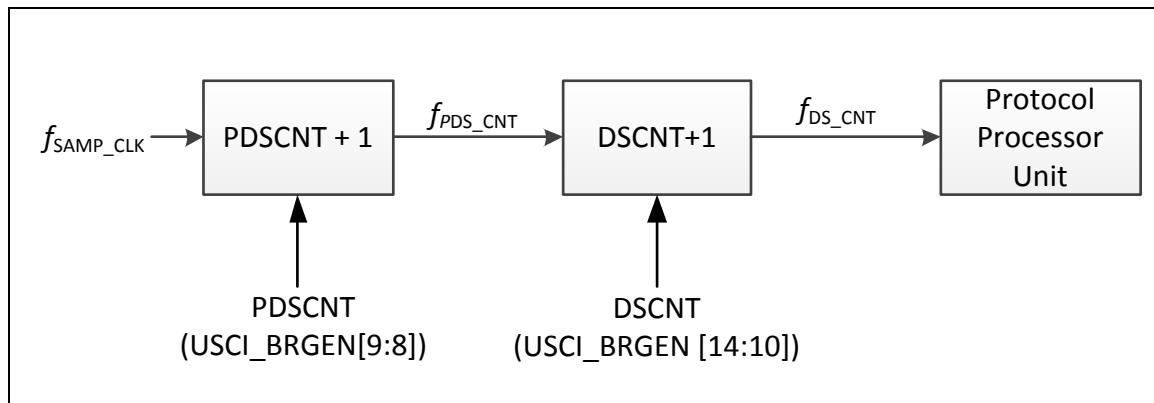


图 6.23-11 采样时间计数器

### 6.23.4.6 数据传输事件和中断

数据传输事件是建立在一个数据字的发送和接收基础上的。相关的标志在寄存器USCI\_PROTSTS中。所有事件可以为中断产生被单独的使能。如果FUNMODE (USCI\_CTL [2:0])被置0, USCI会被禁用。当FUNMODE (USCI\_CTL [2:0])被设置为某个协议端口时，内部状态会被所选的协议的逻辑硬件所控制。

- 传送起始中断事件来表明一个数据字传送已经启动：

当数据被载入到发送移位寄存器时一个传送起始中断发生。由TXSTIF (USCI\_PROTSTS [1])标志指示，如果中断使能将会产生传送起始中断。

- 传送结束中断事件表明一个数据字传送已经完成：

当在移位寄存器的发送数据已经完成时传送结束中断事件发生。由TXENDIF (USCI\_PROTSTS [2])标志指示，如果中断使能将会产生传送结束中断。当移位控制设置（字长度、移位方向等等）时该事件同时可以指示当前数据字传输被内部“冻结”。在UART和I<sup>2</sup>C模式下，依据TXEMPTY (USCI\_BUFSTS [8])和协议相关内部信号传送结束中断事件来确定传送数据的有效性

- 接收起始事件表明一个数据字接收已经启动：

当接收时钟边沿在新数据字的第一个位移动被检测并且接收使能的时候，接收起始事件发生。由RXSTIF (USCI\_PROTSTS [3])标志指示，如果中断使能将会产生接收起始中断。

- 接收事件表明一个数据字接收已经完成：

当一个新接收的字在接收缓冲中可用时，接收事件发生。由RXENDIF (USCI\_PROTSTS [4])标志指示，如果中断使能将会产生接收中断

- 数据丢失事件表明最新接收到的数据丢失：

如果在寄存器USCI\_RXDAT (旧的数据来自 RX\_BUF0 或 RX\_BUF1)的可用的数据没有被读出那么接收缓冲会溢出，新来的数据会丢失掉，该事件发生。由RXOVIF (USCI\_BUFSTS[3]) 标志指示，如果中断使能将会产生协议中断

事件产生和中断结构如下图 6.23-12所示

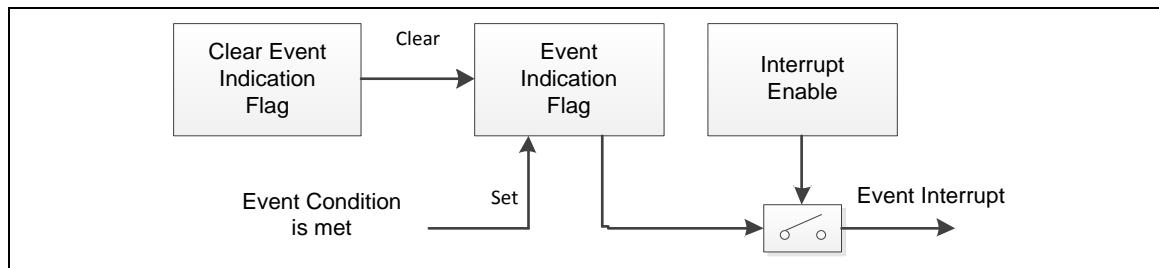


图 6.23-12事件和中断结构

每个中断使能是通过设置 USCI\_INTEN [4:1]. 中的 RXENDIEN, RXSTIEN, TXENDIEN, 和 TXSTIEN。这些事件包括接收结束中断事件、接收起始中断事件、传送结束中断事件和传送起始中断事件。对于具体的中断在每个协议中断使能寄存器中指定。

如果一个定义条件符合，一个事件被侦测到和事件指示标志自动被设置。该标志保持置位直到被软件清除。如果对应中断使能，在事件被侦测到时会产生中断。

寄存器、位和位域表明传输事件和控制产生下表 6.23-3所示的USCI中断。

事件	指示标志	指示清除	中断使能
传送结束中断事件	TXSTIF (USCI_PROTSTS [1])	由软件写1到 USCI_PROTSTS对应的中断位清除	TXSTIEN (USCI_INTEN [1])
传送结束中断事件	TXENDIF (USCI_PROTSTS [2])		TXENDIEN (USCI_INTEN [2])
接收起始中断事件	RXSTIF (USCI_PROTSTS [3])		RXSTIEN (USCI_INTEN [3])
接收结束中断事件	RXENDIF (USCI_PROTSTS [4])		RXENDIEN (USCI_INTEN [4])

表 6.23-3 数据传输事件和中断处理

#### 6.23.4.7 具体协议事件和中断

这些事件与在对应协议的章节描述的具体协议动作有关。相关的标志位在寄存器USCI\_PROTSTS 中。对于共同协议的中断，所有事件可以单独的使能。

事件	指示标志	指示清除	中断使能
UART模式下的协议	USCI_PROTSTS [17:16] 和 USCI_PROTSTS [11:5]		USCI_PROTIEN[2:1]
SPI模式下的协议	USCI_PROTSTS [9:8], USCI_PROTSTS [6:5]	由软件写1到 USCI_PROTSTS对应的中断位清除	USCI_PROTIEN [3:0]
I <sup>2</sup> C 模式下的协议	USCI_PROTSTS [13:8], USCI_PROTSTS [5]		USCI_PROTIEN [6:0]

表 6.23-4 具体协议事件和中断处理

#### 6.23.4.8 唤醒

相关协议唤醒功能信息在唤醒控制寄存器(USCI\_WKCTL)和唤醒状态寄存器(USCI\_WKSTS)中。在可用的协议之间可以共享。因此，这些寄存器的位域代表的意义在不同协议之间会有所不同。

#### 6.23.4.9 PDMA

USCI支持PDMA传输。当PDMAEN (USCI\_PDMACTL [3])置1时，PDMA功能使能。

当TXPDMAEN (USCI\_PDMACTL [1])置1时，控制器会产生请求到PDMA控制器来自动地启动PDMA

当RXPDMAEN (USCI\_PDMACTL [2]) 置1时，控制器会启动PDMA接收进程。当数据接收到FIFO缓冲时，USCI会自动地产生请求到PDMA控制器

在UART功能下，如果有错误事件（包括帧错误、奇偶校验错误和break侦测），RXPDMAEN需求会被清除并保持住。

## 6.24 USCI – UART 模式

### 6.24.1 概述

异步串行通道UART能处理异步数据帧的接收和发送。接收过程是把外设的串行数据转为并行数据，发送过程是把CPU的并行数据转成串行数据发送出去。接收和发送帧是独立的，他们可以在不同时刻启动。

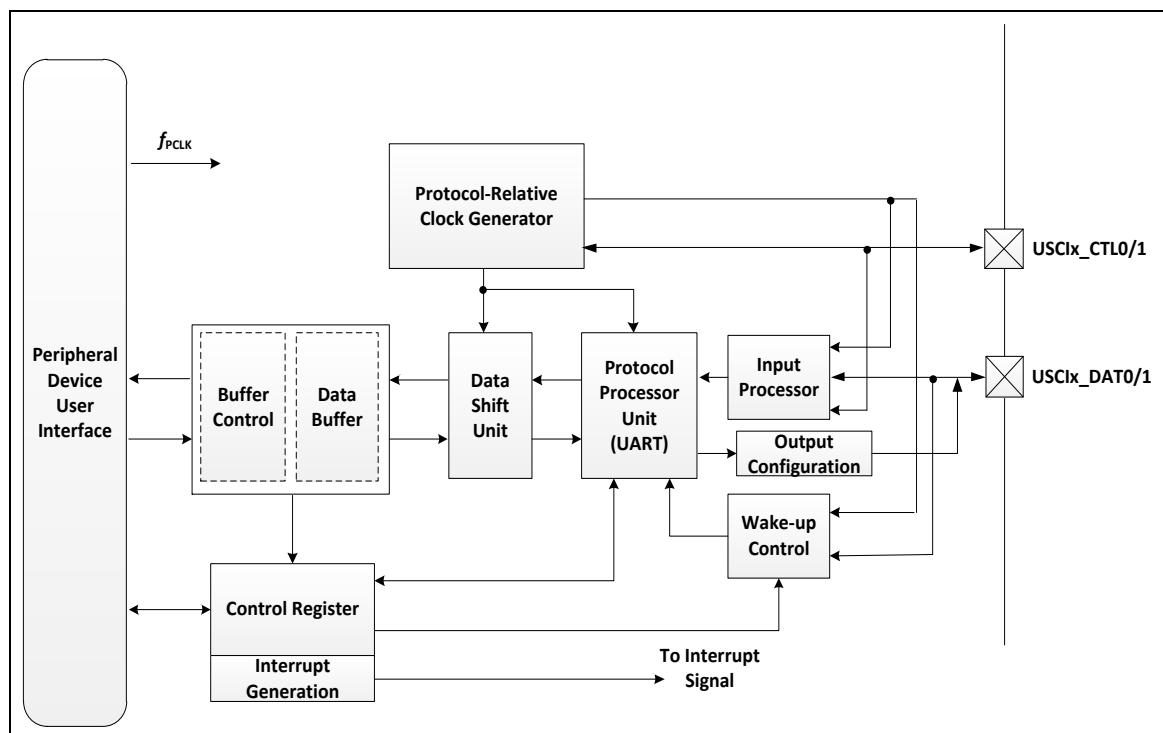
UART控制器也提供流控功能，有两个条件可以唤醒系统。

### 6.24.2 特性

- 有一个发送缓冲和两个接收缓冲来存放数据
- 支持硬件流控功能
- 支持可编程的波特率发生器
- 支持9位数据传输 (支持 9位 RS-485)
- 内建波特率发生捕获事件为波特率侦测提供了可行性
- 支持PDMA功能
- 支持唤醒功能 (仅数据和nCTS 唤醒)

### 6.24.3 框图

USCI的基本配置如下：



Note: x = 0, 1

图 6.24-1 USCI-UART模式框图

#### 6.24.4 基本配置

USCI0\_UART基本配置如下：

- 时钟源配置
  - 使能USCI0时钟的控制位是 CLK\_APBCLK1[8]中的USCI0CKEN位
- 复位USCI0的控制位是 SYS\_IPRST2[8]中的USCI0RST位
- 使能USCI0\_UART功能的控制位是 USCI0\_CTL[2:0] 寄存器, USCI0\_CTL[2:0]=3'b010
- 引脚配置

组	引脚名称	GPIO	MFP
USCI0	USCI0_CLK	PD.0	MFP3
		PB.12	MFP5
		PA.11	MFP6
		PE.2	MFP7
	USCI0_CTL0	PD.4	MFP3
		PC.14	MFP5
		PC.13	MFP6
		PE.6	MFP7
	USCI0_CTL1	PD.3	MFP3
		PB.15	MFP5
		PA.8	MFP6
		PE.5	MFP7
	USCI0_DAT0	PD.1	MFP3
		PB.13	MFP5
		PA.10	MFP6
		PE.3	MFP7
	USCI0_DAT1	PD.2	MFP3
		PB.14	MFP5
		PA.9	MFP6
		PE.4	MFP7

USCI1\_UART基本配置如下：

- 时钟源配置

- 使能USCI1时钟的控制位是 CLK\_APBCLK1[9]中的USCI1CKEN位
- 复位USCI1的控制位是SYS\_IPRST2[9]中的USCI1RST位
- 使能USCI1\_UART功能的控制位在 USCI1\_CTL[2:0] 寄存器, USCI1\_CTL[2:0]=3'b010
- 引脚配置

组	引脚名称	GPIO	MFP
USCI1	USCI1_CLK	PB.8	MFP4
		PD.7, PE.12	MFP6
		PB.1	MFP8
	USCI1_CTL0	PB.10	MFP4
		PD.3, PE.9	MFP6
		PB.5	MFP8
	USCI1_CTL1	PB.9	MFP4
		PD.4, PE.8	MFP6
		PB.4	MFP8
	USCI1_DAT0	PB.7	MFP4
		PD.5, PE.10	MFP6
		PB.2	MFP8
	USCI1_DAT1	PB.6	MFP4
		PD.6, PE.11	MFP6
		PB.3	MFP8

### 6.24.5 功能描述

#### 6.24.5.1 USCI 共同功能描述

具体信息参考 6.23.4。

#### 6.24.5.2 信号描述

UART连接的特点是在发送和接收之间使用单根连接线。接收输入信号(RXD)由输入USCIx\_DAT0处理，发送输出(TXD) 信号由USCIx\_DAT1处理。

对于全双工通信，独立的通信线在每个传输方向是必须的。图 6.24-2展示的是UART模块A和UART模块B之间用点对点全双工通信的连接。

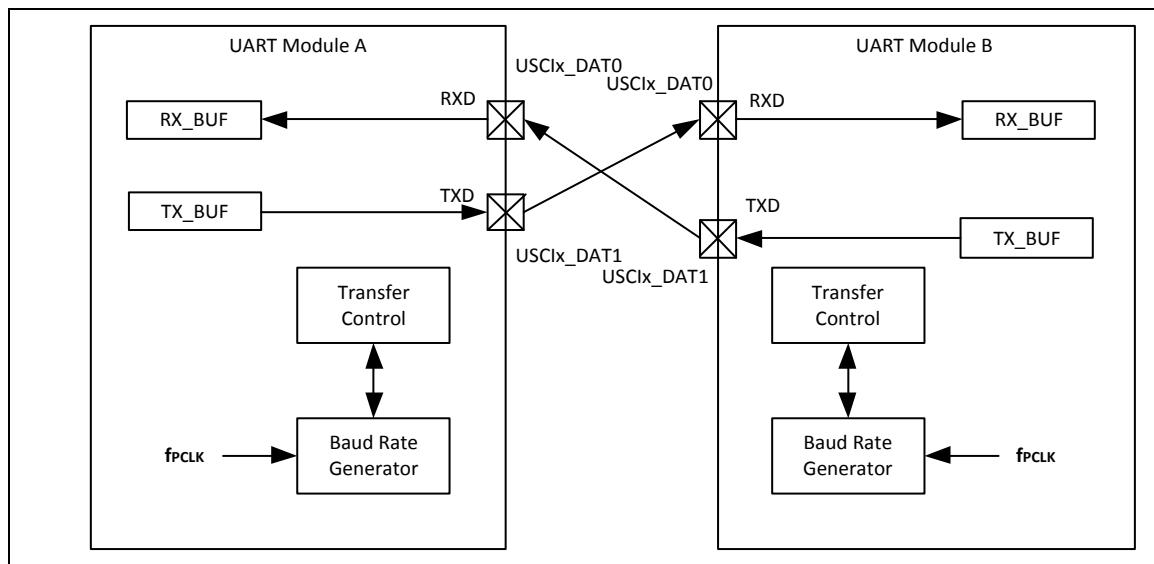


图 6.24-2 全双工通信的 UART 信号连接

#### 6.22.4.8.1 输入信号

对于UART协议，表 6.24-1中列出了输入信号的数目。每种输入信号由输入处理器处理加工。比如信号反相选择控制或数字输入滤波。他们可以依据在协议中的意义进行分类，见表 6.24-1

选择协议		UART
控制输入	USCIx_CTL0	nCTS
	USCIx_CTL1	X
数据输入	USCIx_DAT0	RX
	USCIx_DAT1	X

表 6.24-1 UART 协议输入信号

### 6.22.4.8.2 输出信号

对于UART协议，每个相关协议输出信号是可用的。使用输出的数目实际是取决于所选的协议。他们可以依据在协议中的意义进行分类。

选择协议		UART
控制输出	USCIx_CTL0	X
	USCIx_CTL1	nRTS
数据输出	USCIx_DAT0	X
	USCIx_DAT1	TX

表 6.24-2 UART 协议的输出信号

### 6.24.5.3 帧格式

标准UART帧如下图 6.24-3所示，包含：

- 信号为1的空闲时间
- 信号为0的一个起始帧位(SOF)
- 6~13 位数据
- 一个校验位 (P) ， 可选择奇校验或是偶校验。还可以选择需要该位。
- 信号为1的一到两位的停止位

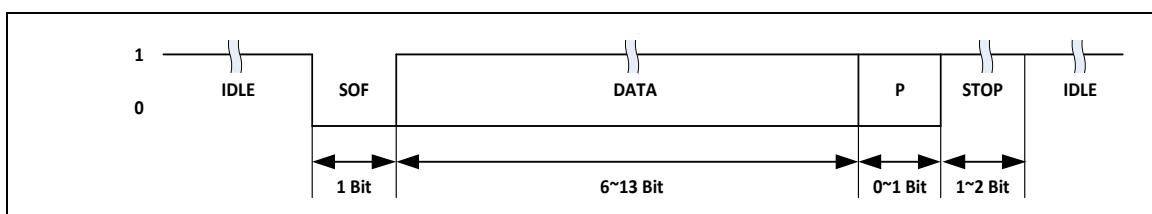


图 6.24-3 UART 标准帧格式

协议的特定位(SOF, P, STOP)由UART协议状态机自动地处理，不会通过接收和发送缓冲出现在数据流中。

#### 6.22.4.8.3 起始位

接收器输入信号USCIx\_DAT0检测是以一个下降沿开始的。当一个下降沿发生时同时接收器处于空闲时或是在最后一个停止位采样点之后，SOF位被侦测到。为了增强抗噪声能力，SOF位时序起始是在第一次下降沿被侦测到。如果采样位SOF的值为1，那么先前下降沿被当作是噪声并且接收器会再次进入空闲。

#### 6.22.4.8.4 数据域

数据域的长度（数据位数目）可以通过DWIDTH (USCI\_LINECTL[11:8])编程。取值范围在6到13数据位。

注:在UART协议中，为了先传输LSB需设置LSB (USCI\_LINECTL[0])为1。

#### 6.22.4.8.5 校验位

UART可以在发送的时候生成校验位，在接收的时候检查校验位。通过PARITYEN (USCI\_PROTCTL[1]) 和 EVENPARITY (USCI\_PROTCTL[2])来选择校验的类型（无校验、奇校验或偶校验）。如果禁用校验功能，UART帧不会包含校验位。为了保证一致性，所有通信的设备必须配置成一样的校验类型。

在数据域的最后数据位之后，如果校验功能使能，发送器会自动地计算校验位并发出。接收器解释该位作为接收校验并且和内部计算的相比较。校验检测和帧检测（停止位）在协议状态寄存器(USCI\_PROTSTS)中被监测。该寄存器包含监测相关协议状态和相关协议错误指示(FRMERR, PARITYERR)这些位。

#### 6.22.4.8.6 停止位

UART帧是以1到2位信号为1的停止位结束的（电平同空闲时电平）。停止位数目由STOPB (USCI\_PROTCTL[0])位编程。在停止位过后一个新的起始位可以被直接的传输。

#### 6.22.4.8.7 传输状态指示

RXBUSY (USCI\_PROTSTS[10])指示接收状态。

通过RXBUSY位监测接收状态。在起始帧位开始到最后的停止位结束期间RXBUSY被置位。

#### 6.24.5.4 操作模式

为了操作UART协议，下面遇到的问题必须考虑到：

- 选择UART模式：

通过设置FUNMODE (USCI\_CTL[2:0]) 为 010B 选择UART协议，通过设置PROTEN (USCI\_PROTCTL [31])为1使能UART协议。注意在改变协议前必须设置FUNMODE为0，强烈建议在使能UART协议前配置好UART的所有参数。

- 引脚连接：

在UART协议中USCIx\_DAT0引脚被用作接收数据输入信号(RX)。输入信号的属性在USCI\_DATIN0中配置。起始位侦测建议设置EDGEDET (USCI\_DATIN0[4:3])为10B。

在UART协议中USCIx\_DAT1引脚被用作发送数据输出信号(TX)。输出信号的属性在USCI\_LINECTL中配置。

在UART协议中USCIx\_CTL0引脚被用作UART清除发送信号(nCTS)。输入控制信号的属性在USCI\_CTLIN0中配置。

在UART协议中USCIx\_CTL1引脚被用作UART请求发送信号(nRTS)。输出控制信号的属性在USCI\_LINECTL中配置。

- 位时序配置：

预期波特率设置必须被选择包括波特率发生器和位时序在内。

- 帧格式配置：

字长度、停止位数目和校验模式必须依据应用需求来设定，这些都是通过USCI\_LINECTL 和 USCI\_PROTCTL寄存器来编程。如果应用需要，数据输入和输出信号可以反转。通过设置LSB (USCI\_LINECTL[0])为1来决定先传送LSB

#### 6.24.5.5 位时序

在UART模式，每一帧位被划分成数据采样时间是为了在sub-bit范围内提供间隔来调整应用需求的采样点。数据采样时间每一位数目在DSCNT (USCI\_BRGEN[14:10])中定义并且数据采样时间长度由PDSCNT (USCI\_BRGEN[9:8])给定。

在图 6.24-4给出的范例中，一位时间是由16个数据采样时间组成 (DSCNT(USCI\_BRGEN[14:10]) = 15)。每一位时间不建议编程少于等于4个数据采样时间。

每位数值的采样点位置固定在1/2采样时间处。这样以来采样的平均值作为采样的值是可行的。

位时序建立(数据采样时间数目)对于发送器和接收器是一样的，因为他们的硬件电路是一样的。

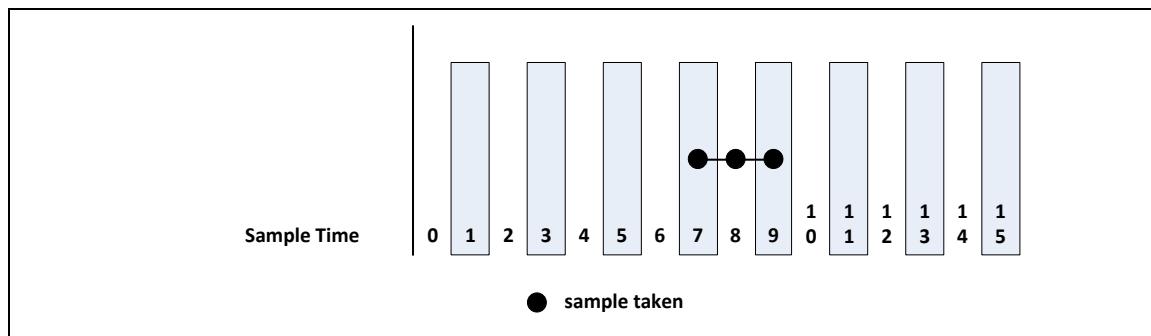


图 6.24-4 UART 位时序 (数据采样时间)

#### 6.24.5.6 波特率发生

在UART模式下波特率取决于数据采样时间每位时间的数目和他们的时序。波特率设定是在发送器和接收器空闲的时候改变。在RCLKSEL, SPCLKSEL, PDSCNT, 和 DSCNT中设置波特率：

- RCLKSEL (USCI\_BRGEN [0])

定义输入频率fREF\_CLK

- SPCLKSEL (USCI\_BRGEN[3:2])

定义采样时钟fSAM\_CLK的时钟源

- PDSCNT (USCI\_BRGEN [9:8])

定义数据采样时间的长度(由 fREF\_CLK 经 1, 2, 3, 或 4分频)

- DSCNT (USCI\_BRGEN [14:10])

定义采样时间每位时间的数目

标准设定是由RCLKSEL = 0 (fREF\_CLK = f<sub>PCLK</sub>), PTCLKSEL = 0 (f<sub>PROT\_CLK</sub> = f<sub>REF\_CLK</sub>) 和 SPCLKSEL = 2'b00 (f<sub>SAMP\_CLK</sub> = f<sub>DIV\_CLK</sub>)决定。在这些条件下，波特率值如下：

$$f_{\text{UART}} = f_{\text{REF\_CLK}} \times \frac{1}{\text{CLKDIV} + 1} \times \frac{1}{\text{PDSCNT} + 1} \times \frac{1}{\text{DSCNT} + 1}$$

为了产生更慢的频率，设置PTCLKSEL = 1 (f<sub>PROT\_CLK</sub> = f<sub>REF\_CLK</sub>2)选择附加除2阶段，如下：

$$f_{\text{UART}} = \frac{f_{\text{REF\_CLK}}}{2} \times \frac{1}{\text{CLKDIV} + 1} \times \frac{1}{\text{PDSCNT} + 1} \times \frac{1}{\text{DSCNT} + 1}$$

如果SPCLKSEL = 2'b10 (f<sub>SAMP\_CLK</sub> = f<sub>SCLK</sub>)，和 RCLKSEL = 0 (f<sub>REF\_CLK</sub> = f<sub>PCLK</sub>)，PTCLKSEL = 0 (f<sub>PROT\_CLK</sub> = f<sub>REF\_CLK</sub>)，波特率如下：

$$f_{\text{UART}} = f_{\text{REF\_CLK}} \times \frac{1}{\text{CLKDIV} + 1} \times \frac{1}{2} \times \frac{1}{\text{PDSCNT} + 1} \times \frac{1}{\text{DSCNT} + 1}$$

设置波特率参数之后UART波特率有一定的误差宽容度。表 6.26-1 I2C 传输速率与PCLK之间的关系

下面列出了相关误差率对于使用计算相关波特率的设定的范例

HCLK 源	PCLK 源	期望波特率	CLKDIV (USCI_BRGEN[25:16])	DSCNT (USCI_BRGEN[14:10])	PDSCNT	实际波特率	误差百分率
12MHz	HCLK	115200	0xC	0x7	0x0	115384	0.16%
12MHz	HCLK	9600	0x7C	0x9	0x0	9600	0%
12MHz	HCLK	2400	0x1F3	0x9	0x0	2400	0%

表 6.24-1 波特率关系

注: {SPCLKSEL, PTCLKSEL, RCLKSEL = 2'b0,1'b0,1'b0}

#### 6.24.5.7 自动波特率侦测

UART控制器支持自动波特率侦测功能。它被用来鉴别来自接收信号(USCIx\_DAT0)的输入波特率，然后在满足侦测波特率功能完成后修改波特率时钟分频器CLKDIV (USCI\_BRGEN[25:16])。依据时序测量计数器章节，时序测量计数器被用作输入信号(USCIx\_DAT0)的时间间隔测量，在每次侦测信号的下降沿，实际定时器值被捕捉到BRDETITV (USCI\_PROTCTL [24:16])中。

当ABREN (PROTOCOL[6])位被使能，必须使用0x55数据格式来进行自动波特率侦测。输入信号下降沿启动波特率计数器并且在下个下降沿时将会加载时序测量计数器值到BRDETITV (USCI\_PROTCTL [24:16])中。建议使用fDIV\_CLK (TMCNTSRC (USCI\_BRGENC[5]) =1)作为计数器源。

在自动波特率功能完成后(输入信号第4个下降沿时刻)，CLKDIV (USCI\_BRGEN[25:16])会被BRDETITV (USCI\_PROTCTL [25:16])修改。如果用户想要正确地接收下次连续的帧，最好是设置CLKDIV (USCI\_BRGEN[25:16]) 和 DSCNT (USCI\_BRGEN[14:10])为一样的值，该值应该在0xF与0x5之间，因为DSCNT被用作定义每位采样计数器并且PDSCNT (USCI\_BRGEN[9:8])是0.

在自动波特率侦测期间，在输入信号每次下降沿之后ABRDETIF (USCI\_PROTSTS[9]) 和 BRDETITV (USCI\_PROTCTL [24:16])会被更新并且自动波特率模式0x55在帧传输完后不会被接收到接收器缓冲里。在第四个输入信号下降沿被侦测到时ABREN位会被硬件清除。因此用户可以读ABREN的状态了解自动波特率功能是否完成。

如果CLKDIV 和 DSCNT没有在自动波特率功能计算中设置为同样的值，用户应该在自动波特率功能完成之后通过BRDETITV 和 CLKDIV的值计算出合理的平均波特率。

如果输入信号的波特率非常慢并且时序测量计数器的位时间不能计算出输入位时间的正确周期，有一个ABERRSTS (USCI\_PROTSTS[11])位来指示自动波特率侦测的错误信息。此刻，用户需要修改CLKDIV的值和要求主机再次发送0x55。

依据时序测量计数器的有效范围，对于BRDETITV最大自动波特率侦测是0x1FE。UART自动波特率控制如图 6.24-5所示。

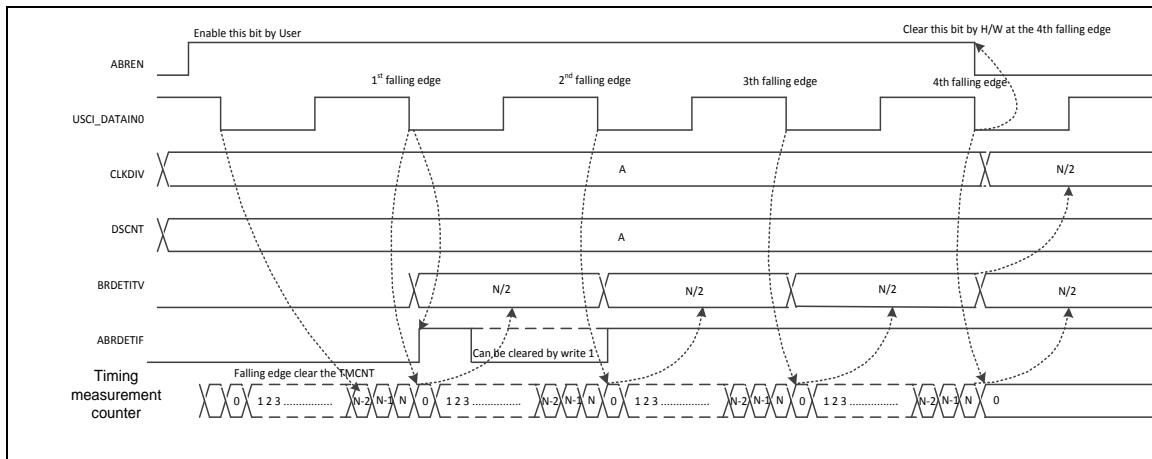


图 6.24-5 UART 自动波特率控制

#### 6.24.5.8 自动流控

UART 支持硬件自动流控，为接收器缓冲提供 nRTS 流控是由 RXFULL (UART\_BUFSTS[1]) 指示的。当缓冲满的时候 (RXFULL = 1)，nRTS 失效。

UART 为发送器也提供了 nCTS 流控。当 nCTS 有效时，nCTS 被用作控制数据发送。

#### 6.24.5.9 RS-485 功能

UART 控制器可以扮演 RS-485 主机发送器的角色通过设置校验位（第 9 位）为 1 来鉴别地址帧。对于数据帧，校验位设置为 0。当 STICKEN (USCI\_PROTCTL[26]) 被设置时，软件可以使用每一数据的 15 位来控制校验位（PARITYEN (USCI\_PROTCTL[1]) 被设置）。例如，如果 STICKEN 被置 1，并且数据序列是 0x8015, 0x8033, 0x0055, 0x0033 和 0x80AA 数据 0x15, 0x33, 0x55, 0x33 和 0xAA 发送校验将是 1, 1, 0, 0 和 1。

UART 控制器可以扮演 RS-485 地址可寻从机角色，当 PARITYEN (USCI\_PROTCTL[1]), EVENPARITY (USCI\_PROTCTL[2]) 和 STICKEN (USCI\_PROTCTL[26]) 被设置时，PARITYERR (USCI\_PROTSTS[5]) 相关协议错误被当作地址位侦测。如果 PARITYERR 被设置，意味着地址位在接收总线上的地址位被侦测到，否则数据被接收到缓冲中。

#### 6.24.5.10 唤醒功能

USCI 控制器在 UART 模式下支持唤醒系统功能。唤醒源包括输入数据和 nCTS 引脚。每一个源的描述如下：

##### (a) 输入数据唤醒

当系统处在掉电模式时并且 WKEN (USCI\_WKCTL [0]) 和 DATWKEN (USCI\_PROTCTL[9]) 都被设置，输入数据引脚触发唤醒系统。在系统唤醒后为了接收输入的数据，WAKECNT (USCI\_PROTCTL[14:11]) 需要被设置。WAKECNT (USCI\_PROTCTL[14:11]) 指明当设备被从掉电模式下唤醒时从机获得第一个位及起始位需要多少个时钟周期（该时钟是 f<sub>PDS\_CNT</sub>）。输入数据唤醒如图 6.24-6 所示。

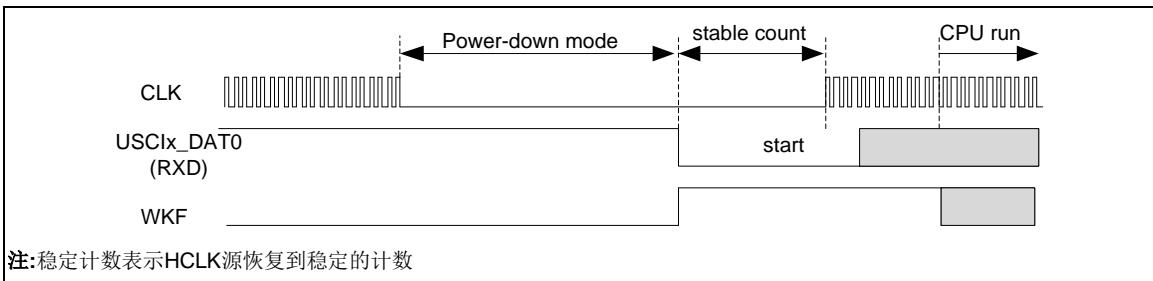


图 6.24-6 输入数据唤醒

## (b) nCTS 引脚唤醒

当系统处在掉电模式下时并且WKEN (USCI\_WKCTL [0]) 和 CTSWKEN (USCI\_PROTCTL[10])被设置, nCTS引脚的变化可以唤醒系统。nCTS唤醒如图 6.24-7和图 6.24-8所示。

## 状态 1(nCTS 从低变高):

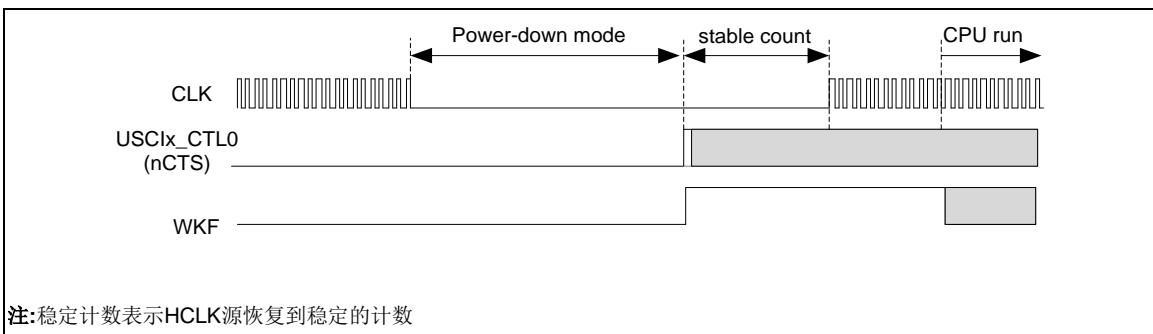


图 6.24-7 nCTS 唤醒状态 1

## 状态2 (nCTS 从高变低):

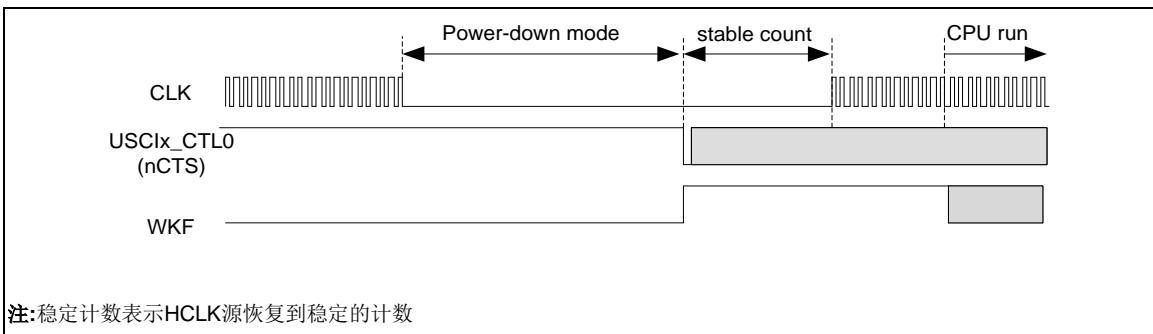


图 6.24-8 nCTS 唤醒状态 2

## 6.24.5.11 中断事件

UART为协议事件和数据传输事件提供了中断。描述如下:

#### 6.22.4.8.8 协议中断事件

在UART模式下，可以产生下面相关协议事件并且可以产生协议中断。

请注意USCI\_PROTSTS位不会被硬件自动地清除，必须通过软件清除，这样是为了监控新到来的事件。

- 接收器线状态：

两个相关协议错误FRMERR (USCI\_PROTSTS[6]) 和PARITYERR (USCI\_PROTSTS[5])标志在相应的接收器缓冲状态寄存器中被置位指示每次接收到的数据

在UART模式下，校验位检查结果是通过相关协议错误指示的（0=接收校验位等于计算校验值），帧检查是通过相关协议错误指示的（0=接收停止位等于格式值‘1’）。

当接收数据为0、接收校验和停止位都为0时break错误标志BREAK (UART\_PROTSTS[7])被置位。

在BREAK, FRMERR, PARITYERR (USCI\_PROTSTS[7:5])中有校验错误、帧错误或break数据侦测的中断指示。

- 自动波特率侦测：

自动波特率中断ABRDETIF (USCI\_PROTSTS [9])对于自动波特率捕获功能用来指示时序测量计数器已得到2位时间

在接收信号的第一次下降沿是自动波特率侦测功能被使能。在下次下降沿被侦测之后自动波特率侦测功能进行测量。当帧传输完成时结束。在传输完成后，被除两次的时序测量计数器的值等于采样时间每位的数目。用户可以读出BRDETITV (USCI\_PROTCTL[24:16])的值和写入波特率发生器寄存器CLKDIV (USCI\_BRGEN[25:16])。

#### 6.22.4.8.9 数据传输中断处理

数据传输中断指示相关UART帧处理事件。

- 发送起始中断:

在数据起始位之后TXSTIF (USCI\_PROTSTS [1])被置位，在缓冲模式下，当新数据可以被写入USCI\_TXDAT时这个时候是最早时刻点。

- 发送结束:

该中断表明发送器已经完全的完成所有在缓冲中的数据传输。TXENDIF (USCI\_PROTSTS [2])在最后停止位结束后被置位。

- 接收起始中断:

在采样到起始位的时刻RXSTIF (USCI\_PROTSTS [3])被置位。

- 接收帧完成:

该中断指示接收器已完全的完成一帧接收。在接收到最后位结束之后RXENDIF (USCI\_PROTSTS [4])被置位。

#### 6.24.5.12 编程范例

下面步骤是被用作配置UART协议设置和数据传输

1. 设置 FUNMODE (UART\_CTL[2:0]) 为 0x2 来选择 UART 协议
2. 写波特率寄存器 UUART\_BRGEN 来选择期望的波特率
  - 设置 SPCLKSEL (UART\_BRGEN[3:2]), PTCLKSEL (UART\_BRGEN[1]) 和 RCLKSEL (UART\_BRGEN[0]) 来选择时钟源
  - 配置 CLKDIV (UART\_BRGEN[25:16]), DSCNT (UART\_BRGEN[14:10]) 和 PDSCNT (UART\_BRGEN[9:8]) 来确定波特率分频
3. 写线控制寄存器 UUART\_LINECTL 和协议控制寄存器 UUART\_PROTCTL 来配置传输数据格式和 UART 协议设置
  - 在 DWIDTH (UART\_LINECTL[11:8]) 中编程数据域长度
  - 通过设置 EVENPARITY (UART\_PROTCTL[2]) 和 PARITYEN (UART\_PROTCTL[1]) 来使能校验位和确定校验类型
  - 通过设置 STOPB (UART\_PROTCTL[0]) 配置停止位长度
  - UART 协议下，使能 LSB (UART\_LINECTL[0]) 来选择先传输 LSB
  - 设置 EDGEDET (UART\_DATIN0[4:3]) 为“10”来选择下降沿作为接收起始位侦测

4. 设置 PROTEN(UUART\_PROTCTL[31]) 为 1 来使能 UART 协议
5. 发送和接收数据
  - 写发送数据寄存器 UUART\_TXDAT 来发送数据
  - 等待 TXSTIF(UUART\_PROTSTS[1]) 直到被置位，然后用户可以写入下次发送数据到 UUART\_TXDAT
  - 当 TXENDIF(UUART\_PROTSTS[2]) 被置位时，发送缓冲为空并且最后数据的停止位已经被发送。
  - 如果 RXENDIF(UUART\_PROTSTS[4])被置位时，数据帧已经完成。用户可以通过读接收数据寄存器 UUART\_RXDAT 可以获得该数据。

## 6.24.6 寄存器映射

R: 只读, W: 只写, R/W: 读/写

寄存器	偏移地址	R/W	描述	复位值
<b>USCI 基地址:</b>				
<b>USCI<sub>n</sub>_BA = 0x400D_0000 + (0x1000 * n)</b>				
<b>N= 0, 1</b>				
<b>USCI_CTL</b>	USCI <sub>n</sub> _BA+0x00	R/W	USCI控制寄存器	0x0000_0000
<b>USCI_INTEN</b>	USCI <sub>n</sub> _BA+0x04	R/W	USCI中断使能寄存器	0x0000_0000
<b>USCI_BRGEN</b>	USCI <sub>n</sub> _BA+0x08	R/W	USCI波特率发生器寄存器	0x0000_3C00
<b>USCI_DATINO</b>	USCI <sub>n</sub> _BA+0x10	R/W	USCI 0输入数据信号配置寄存器0	0x0000_0000
<b>USCI_CTLINO</b>	USCI <sub>n</sub> _BA+0x20	R/W	USCI输入控制信号配置寄存器0	0x0000_0000
<b>USCI_CLKIN</b>	USCI <sub>n</sub> _BA+0x28	R/W	USCI输入时钟信号配置寄存器	0x0000_0000
<b>USCI_LINECTL</b>	USCI <sub>n</sub> _BA+0x2C	R/W	USCI线控制寄存器	0x0000_0000
<b>USCI_TXDAT</b>	USCI <sub>n</sub> _BA+0x30	W	USCI发送数据寄存器	0x0000_0000
<b>USCI_RXDAT</b>	USCI <sub>n</sub> _BA+0x34	R	USCI接收数据寄存器	0x0000_0000
<b>USCI_BUFCTL</b>	USCI <sub>n</sub> _BA+0x38	R/W	USCI发送/接收缓冲控制寄存器	0x0000_0000
<b>USCI_BUFSTS</b>	USCI <sub>n</sub> _BA+0x3C	R	USCI发送/接收缓冲状态寄存器	0x0000_0101
<b>USCI_PDMACTL</b>	USCI <sub>n</sub> _BA+0x40	R/W	USCI PDMA控制寄存器	0x0000_0000
<b>USCI_WKCTL</b>	USCI <sub>n</sub> _BA+0x54	R/W	USCI唤醒控制寄存器	0x0000_0000
<b>USCI_WKSTS</b>	USCI <sub>n</sub> _BA+0x58	R/W	USCI唤醒状态寄存器	0x0000_0000
<b>USCI_PROTCTL</b>	USCI <sub>n</sub> _BA+0x5C	R/W	USCI协议控制寄存器	0x0000_0000
<b>USCI_PROTIEN</b>	USCI <sub>n</sub> _BA+0x60	R/W	USCI协议中断使能寄存器	0x0000_0000
<b>USCI_PROTSTS</b>	USCI <sub>n</sub> _BA+0x64	R/W	USCI协议状态寄存器	0x0000_0000

### 6.24.7 寄存器描述

#### USCI\_CTL USCI 控制寄存器

寄存器	偏移地址	R/W	描述	复位值
USCI_CTL	USCIIn_BA+0x00	R/W	USCI 控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved					FUNMODE		

位	描述	
[31:3]	Reserved	保留
[2:0]	FUNMODE	<p><b>功能模式</b></p> <p>该域是选择USCI控制器的协议。待选的协议是不可用的。当两个协议之间切换时，在选择新协议前USCI必须先禁用。在用户写000到FUNMODE同时USCI会被复位。</p> <p>000 = 禁用 USCI，所有协议相关的状态机器被设置为空闲状态。</p> <p>001 = 选择SPI 协议</p> <p>010 = 选择 UART 协议</p> <p>100 = 选择I<sup>2</sup>C 协议</p> <p><b>注:</b>其它值保留</p>

USCI\_INTEN USCI 中断使能寄存器

寄存器	偏移地址	R/W	描述	复位值
USCI_INTEN	USCIIn_BA+0x04	R/W	USCI 中断使能寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved			RXENDIEN	RXSTIEN	TXENDIEN	TXSTIEN	Reserved

位	描述	
[31:5]	<b>Reserved</b>	保留
[4]	<b>RXENDIEN</b>	<p><b>接收结束中断使能位</b>            该位是使能发生接收结束事件时候产生中断            0 = 禁用接收结束中断            1 = 使能接收结束中断</p>
[3]	<b>RXSTIEN</b>	<p><b>接收起始中断使能位</b>            该位是使能发生接收起始事件时候产生中断            0 = 禁用接收起始中断            1 = 使能接收起始中断</p>
[2]	<b>TXENDIEN</b>	<p><b>发送结束中断使能位</b>            该位是使能发生发送结束事件时候产生中断            0 = 禁用发送结束中断            1 = 使能发送结束中断</p>
[1]	<b>TXSTIEN</b>	<p><b>发送起始中断使能位</b>            该位是使能发生发送起始事件时候产生中断            0 = 禁用发送起始中断            1 = 使能发送起始中断</p>
[0]	<b>Reserved</b>	保留

## USCI\_BRGEN USCI 波特率发生器寄存器

寄存器	偏移地址	R/W	描述	复位值
USCI_BRGEN	USCIIn_BA+0x08	R/W	USCI 波特率发生器寄存器	0x0000_3C00

31	30	29	28	27	26	25	24
Reserved						CLKDIV	
23	22	21	20	19	18	17	16
CLKDIV							
15	14	13	12	11	10	9	8
Reserved	DSCNT					PDSCNT	
7	6	5	4	3	2	1	0
Reserved		TMCNTSRC	TMCNTEN	SPCLKSEL		PTCLKSEL	RCLKSEL

位	描述	
[31:26]	Reserved	保留
[25:16]	CLKDIV	<p><b>时钟分频器</b>            该域是定义协议时钟频率 <math>f_{PROT\_CLK}</math> 和时钟分频器频率 <math>f_{DIV\_CLK}</math> (<math>f_{DIV\_CLK} = f_{PROT\_CLK} / (CLKDIV+1)</math>)之间的比率  <b>注:</b> 在UART功能下, 当自动波特率功能(ABREN(USCI_PROTCTL[6]))使能时, 会在输入数据0x55的第四个下降沿时被硬件更新。修改值是在位5和位6之间的时间平均值。用户可以使用修改的CLKDIV和新的BRDETTIV (USCI_PROTCTL[24:16])来计算精确的波特率</p>
[15]	Reserved	保留
[14:10]	DSCNT	<p><b>采样计数器的分母</b>            该域定义采样时钟 <math>f_{SAMP\_CLK}</math> 的分频比。            分频 <math>f_{DS\_CNT} = f_{PDS\_CNT} / (DSCNT+1)</math>.  <b>注:</b> 在UART模式下, DSCNT最大值是0xF, 建议设置4以上来保证接收数据的采样时正确的值。</p>
[9:8]	PDSCNT	<p><b>采样计数器预分频器</b>            该域定义的是对来自采样时钟 <math>f_{SAMP\_CLK}</math> 的分频比            分频 <math>f_{PDS\_CNT} = f_{SAMP\_CLK} / (PDSCNT+1)</math>.</p>
[7:6]	Reserved	保留
[5]	TMCNTSRC	<p><b>时序测量计数器时钟源选择</b>            0 = 来自 <math>f_{PROT\_CLK}</math>.            1 = 来自 <math>f_{DIV\_CLK}</math>.</p>
[4]	TMCNTEN	<p><b>时序测量计数器使能位</b>            该位是使能10位时序测量计数器            0 = 禁用时序测量计数器            1 = 使能时序测量计数器</p>

[3:2]	<b>SPCLKSEL</b>	<b>采样时钟源选择</b> 该域用于对协议采样时钟( $f_{SAMP\_CLK}$ )的时钟源选择 00 = $f_{SAMP\_CLK} = f_{DIV\_CLK}$ . 01 = $f_{SAMP\_CLK} = f_{PROT\_CLK}$ . 10 = $f_{SAMP\_CLK} = f_{SCLK}$ . 11 = $f_{SAMP\_CLK} = f_{REF\_CLK}$ .
[1]	<b>PTCLKSEL</b>	<b>协议协议时钟源选择</b> 该位用于选择协议时钟( $f_{PROT\_CLK}$ )的源信号 0 = 参考时钟 $f_{REF\_CLK}$ . 1 = $f_{REF\_CLK2}$ (频率是 $f_{REF\_CLK}$ 一半).
[0]	<b>RCLKSEL</b>	<b>参考时钟源选择</b> 该位是选择参考时钟( $f_{REF\_CLK}$ )源信号 0 = 外设设备时钟 $f_{PCLK}$ . 1 = 保留

**USCI\_DATINO USCI输入数据信号配置寄存器0**

寄存器	偏移地址	R/W	描述	复位值
USCI_DATINO	USCIIn_BA+0x10	R/W	USCI输入数据信号配置寄存器0	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved			EDGEDET		ININV	Reserved	SYNCSEL

位	描述	
[31:5]	<b>Reserved</b>	保留
[4:3]	<b>EDGEDET</b>	<p><b>输入信号边沿侦测模式</b>          该域是选择输入信号数据信号边沿激活触发事件          00 = 禁用激活触发事件          01 = 上升沿激活输入数据信号触发事件          10 = 下降沿激活输入数据信号触发事件          11 = 双边沿激活输入数据信号触发事件  <b>注:</b>在UART功能模式，建议该域设置为10。</p>
[2]	<b>ININV</b>	<p><b>输入信号反相选择</b>          该域是使能对输入异步信号的反相          0 = 对异步输入信号不反相          1 = 对异步输入信号反相</p>
[1]	<b>Reserved</b>	保留
[0]	<b>SYNCSEL</b>	<p><b>输入信号同步选择</b>          该位是选择异步输入（带反相的）信号或是同步（带过滤的）信号被用作数据移位单元的输入          0 = 异步信号作为信号被用作数据移位单元的输入          1 = 同步信号作为信号被用作数据移位单元的输入</p>

**USCI\_CTLIn0 USCI输入控制信号配置寄存器0**

寄存器	偏移地址	R/W	描述	复位值
USCI_CTLIn0	USCIIn_BA+0x20	R/W	USCI输入控制信号配置寄存器0	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved					ININV	Reserved	SYNCSEL

位	描述	
[31:3]	<b>Reserved</b>	保留
[2]	<b>ININV</b>	<b>输入信号反相选择</b> 该域是定义使能对输入异步信号输入反相 0 = 对异步输入信号不反相 1 = 对异步输入信号反相
[1]	<b>Reserved</b>	保留
[0]	<b>SYNCSEL</b>	<b>输入同步信号选择</b> 该位是选择异步输入（带反相的）信号或是同步（带过滤的）信号被用作数据移位单元的输入 0 = 异步信号作为信号被用作数据移位单元的输入 1 = 同步信号作为信号被用作数据移位单元的输入

**USCI\_CLKIN USCI输入时钟信号配置寄存器**

寄存器	偏移地址	R/W	描述	复位值
USCI_CLKIN	USCIIn_BA+0x28	R/W	USCI 输入时钟信号配置寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved							SYNCSEL

位	描述	
[31:1]	Reserved	保留
[0]	SYNCSEL	<p><b>输入同步信号选择</b></p> <p>该位是选择异步输入（带反相的）信号或是同步（带过滤的）信号被用作数据移位单元的输入</p> <p>0 = 异步信号作为信号被用作数据移位单元的输入</p> <p>1 = 同步信号作为信号被用作数据移位单元的输入</p>

## USCI\_LINECTL USCI线控制寄存器

寄存器	偏移地址	R/W	描述	复位值
USCI_LINECTL	USCIIn_BA+0x2C	R/W	USCI 线控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved				DWIDTH			
7	6	5	4	3	2	1	0
CTLOINV	Reserved	DATOINV	Reserved				LSB

位	描述	
[31:12]	Reserved	保留
[11:8]	DWIDTH	<p><b>传输的字长</b></p> <p>该域定义发送和接收的数据字长。在数据缓冲中数据总是右对齐。USCI支持4到16位的字长</p> <p>0x0:在数据字中包含16位，位于[15:0]这些位中</p> <p>0x1: 保留</p> <p>0x2: 保留</p> <p>0x3: 保留</p> <p>0x4:在数据字中包含4位，位于[3:0]这些位中</p> <p>0x5:在数据字中包含5位，位于[4:0]这些位中</p> <p>...</p> <p>0xF:在数据字中包含15位，位于[14:0]这些位中</p> <p><b>注:</b> 在UART模式下，长度可配置为6~13位</p>
[7]	CTLOINV	<p><b>控制信号输出反相选择</b></p> <p>该位定义了内部控制信号和输出控制信号之间的关系</p> <p>0 = 无影响</p> <p>1 = 在输出前控制信号被反相</p> <p><b>注:</b> 在UART模式下，控制信号就是nRTS信号</p>
[6]	Reserved	保留
[5]	DATOINV	<p><b>数据输出反相选择</b></p> <p>该位定义了内部移位数据值和USCIx DAT1引脚输出数据信号之间的关系</p> <p>0 = USCIx_DAT1的值等于数据移位寄存器</p> <p>1 = USCIx_DAT1的值为数据移位寄存器的反相</p>
[4:1]	Reserved	保留

[0]	LSB	LSB优先传输选择 0 =取决于DWIDTH设置的接收/发送数据缓冲的MSB先发送/接收 1 =数据缓冲的位0及LSB被先发送/接收
-----	-----	--

USCI\_TXDAT USCI 发送数据寄存器

寄存器	偏移地址	R/W	描述	复位值
USCI_TXDAT	USCIIn_BA+0x30	W	USCI 发送数据寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
TXDAT							
7	6	5	4	3	2	1	0
TXDAT							

位	描述	
[31:16]	Reserved	保留
[15:0]	TXDAT	发送数据 软件可以写16位发送数据到该域用来发送

USCI\_RXDAT USCI 接收数据寄存器

寄存器	偏移地址	R/W	描述	复位值
USCI_RXDAT	USCIIn_BA+0x34	R	USCI 接收数据寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
RXDAT							
7	6	5	4	3	2	1	0
RXDAT							

位	描述	
[31:16]	Reserved	保留
[15:0]	RXDAT	<p><b>接收的数据</b>            该域值监控存储在接收数据缓冲的接收到的数据            注: RXDAT[15:13] 表明BREAK, FRMERR 和 PARITYERR (USCI_PROTSTS[7:5])相同帧状态</p>

USCI\_BUFCTL USCI 发送/接收缓冲控制寄存器

寄存器	偏移地址	R/W	描述	复位值
USCI_BUFCTL	USCIIn_BA+0x38	R/W	USCI 发送/接收缓冲控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved						RXRST	TXRST
15	14	13	12	11	10	9	8
RXCLR	RXOVIEN	Reserved					
7	6	5	4	3	2	1	0
TXCLR	Reserved						

位	描述	
[31:18]	<b>Reserved</b>	保留
[17]	<b>RXRST</b>	<p><b>接收复位</b> 0 = 无影响 1 = 复位接收相关计数器、状态机、接收移位寄存器的内容和数据缓冲。</p> <p><b>注1:</b> 1个PCLK周期后自动被清除 <b>注2:</b> 在该位被置1前，建议检查RXBUSY (USCI_PROTSTS[10])</p>
[16]	<b>TXRST</b>	<p><b>发送复位</b> 0 = 无影响 1 = 复位发送相关计数器、状态机、发送移位寄存器的内容和数据缓冲。</p> <p><b>注:</b> 1个PCLK周期后自动被清除</p>
[15]	<b>RXCLR</b>	<p><b>清接收缓冲</b> 0 = 无影响 1 = 接收缓冲被清除（填充被清除并且输出指针被设置为输入指针的值）仅当缓冲没有参与数据通信时被使用。</p> <p><b>注:</b> 1个PCLK周期后自动被清除</p>
[14]	<b>RXOVIEN</b>	<p><b>接收缓冲溢出错误中断使能控制</b> 0 = 禁用接收溢出中断 1 = 使能接收溢出中断</p>
[13:8]	<b>Reserved</b>	保留
[7]	<b>TXCLR</b>	<p><b>清发送缓冲</b> 0 = 无影响 1 = 发送缓冲被清除（填充被清除并且输出指针被设置为输入指针的值）仅当缓冲没有参与数据通信时被使用。</p> <p><b>注:</b> 1个PCLK周期后自动被清除</p>

[6:0]	Reserved	保留
-------	----------	----

USCI\_BUFSTS USCI 发送/接收缓冲状态寄存器

寄存器	偏移地址	R/W	描述	复位值
USCI_BUFSTS	USCIIn_BA+0x3C	R	USCI 发送/接收缓冲状态寄存器	0x0000_0101

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved						TXFULL	TXEMPTY
7	6	5	4	3	2	1	0
Reserved				RXOVIF	Reserved	RXFULL	RXEMPTY

位	描述	
[31:10]	<b>Reserved</b>	保留
[9]	<b>TXFULL</b>	<b>发送缓冲满指示</b> 0 =发送缓冲还没满 1 =发送缓冲满
[8]	<b>TXEMPTY</b>	<b>发送缓冲空指示</b> 0 =发送缓冲非空 1 =发送缓冲为空
[7:4]	<b>Reserved</b>	保留
[3]	<b>RXOVIF</b>	<b>接收缓冲溢出错误中断状态</b> 该位表示接收缓冲溢出错误事件被侦测到。如果RXOVIEN (USCI_BUFCTL[14])使能，会激活相应的中断请求。软件写1清除该位。 0 = 接收缓冲溢出错误事件没有被侦测到 1 = 接收缓冲溢出错误事件被侦测到
[2]	<b>Reserved</b>	保留
[1]	<b>RXFULL</b>	<b>接收缓冲满指示</b> 0 =接收缓冲还没满 1 =接收缓冲满
[0]	<b>RXEMPTY</b>	<b>接收缓冲空指示</b> 0 =接收缓冲非空 1 =接收缓冲为空

USCI\_PDMACTL USCI PDMA控制寄存器

寄存器	偏移地址	R/W	描述	复位值
USCI_PDMACTL	USCIIn_BA+0x40	R/W	USCI PDMA控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved				PDMAEN	RXPDMAEN	TXPDMAEN	PDMARST

位	描述	
[31:4]	Reserved	保留
[3]	PDMAEN	<b>PDMA模式使能位</b> 0 = 禁用PDMA功能 1 = 使能PDMA功能
[2]	RXPDMAEN	<b>PDMA接收通道有效</b> 0 = 禁用PDMA接收功能 1 = 使能PDMA接收功能
[1]	TXPDMAEN	<b>PDMA发送通道有效</b> 0 = 禁用PDMA发送功能 1 = 使能PDMA发送功能
[0]	PDMARST	<b>PDMA 复位</b> 0 = 无影响 1 = 复位USCI的PDMA控制逻辑。该位会自动地被清0

USCI\_WKCTL USCI 唤醒控制寄存器

寄存器	偏移地址	R/W	描述	复位值
USCI_WKCTL	USCIIn_BA+0x54	R/W	USCI 唤醒控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved					PDBOPT	Reserved	WKEN

位	描述	
[31:3]	<b>Reserved</b>	保留
[2]	<b>PDBOPT</b>	<b>掉电模式选择</b> 0 = 如果用户试图在协议正在传输的时候执行WFI来进入掉电模式，MCU会停止传输并且立即进入掉电模式 1 = 如果用户试图在协议正在传输的时候执行WFI来进入掉电模式，传输会继续进行并且MCU立即进入掉电模式
[1]	<b>Reserved</b>	保留
[0]	<b>WKEN</b>	<b>唤醒使能位</b> 0 = 禁用唤醒功能 1 = 使能唤醒功能

**USCI WKSTS USCI 唤醒状态寄存器**

寄存器	偏移地址	R/W	描述	复位值
USCI_WKSTS	USCIIn_BA+0x58	R/W	USCI 唤醒状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved							WKF

位	描述	
[31:1]	Reserved	保留
[0]	WKF	唤醒标志 当芯片被从掉电模式唤醒，该位置1.软件写1清除该位。

**USCI PROTCTL USCI 协议控制寄存器– UART**

寄存器	偏移地址	R/W	描述	复位值
USCI_PROTCTL	USCIIn_BA+0x5C	R/W	USCI 协议控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
PROTEN	Reserved	BCEN	Reserved	Reserved	STICKEN	Reserved	BRDESTITV
23	22	21	20	19	18	17	16
BRDESTITV							
15	14	13	12	11	10	9	8
Reserved	WAKECNT				CTSWKEN	DATWKEN	Reserved
7	6	5	4	3	2	1	0
Reserved	ABREN	RTSAUDIREN	CTSAUTOEN	RTSAUTOEN	EVENPARITY	PARITYEN	STOPB

位	描述
[31]	<b>PROTEN</b> UART协议使能位 0 = 禁用UART 协议 1 = 使能UART 协议
[30]	<b>Reserved</b> 保留
[29]	<b>BCEN</b> 发送 Break 控制使能位 0 =禁用发送Break控制 1 =使能发送Break控制 <b>注:</b> 当该位被置1，串行数据输出（TX）被强制为逻辑0状态。该位仅在TX线上有效，对发送逻辑单元没有影响
[27]	<b>Reserved</b> 保留
[26]	<b>STICKEN</b> Stick 校验使能位 0 =禁用Stick校验 1 =使能Stick校验 <b>注:</b> 详细信息请参考RS-485章节
[25]	<b>Reserved</b> 保留
[24:16]	<b>BRDESTITV</b> 波特率侦测间隔 该位指明从机在一个位时间内计算波特率需要多少时钟周期（该时钟由TMCNTSRC (USCI_BRGEN [5])决定）。总线的顺序应该是1和0这样交替（比如，输入数据格式应该是0x55）。用户可以读该值去了解当前总线输入波特率，不管ABRDETIF (USCI_PROTSTS[9])何时置位。 <b>注:</b> 软件写0到BRDESTITV.清除为0。
[15]	<b>Reserved</b> 保留
[14:11]	<b>WAKECNT</b> 唤醒计数器 该位指明当设备被从掉电模式下唤醒时从机获得第一个位及起始位需要多少个时钟周期（该

		时钟是 $f_{PDS\_CNT}$ ) 。
[10]	<b>CTSWKEN</b>	<b>nCTS唤醒模式使能位</b> 0 = 禁用nCTS 唤醒模式 1 =使能 nCTS唤醒模式
[9]	<b>DATWKEN</b>	<b>数据唤醒模式使能位</b> 0 = 禁用数据唤醒模式 1 =使能数据唤醒模式
[6]	<b>ABREN</b>	<b>自动波特率侦测使能位</b> 0 =禁用自动波特率侦测 1 =使能自动波特率侦测 <b>注:</b> 当自动波特率侦测操作结束时硬件清除该位。如果ARBIEN (USCI_PROTIEN [1])使能, 将产生相关中断ABRDETIF (USCI_PROTST[9])
[5]	<b>RTSAUDIREN</b>	<b>nRTS自动方向使能位</b> 当nRTS自动方向使能, 如果TX缓冲为空, UART自动激活nRTS信号 0 =禁用nRTS自动方向控制 1 =使能nRTS自动方向控制 <b>注 1:</b> 该位是用在RS485下nRTS自动方向控制 <b>注 2:</b> 该位仅在RTSAUTOEN没有置位情况下有作用
[4]	<b>CTSAUTOEN</b>	<b>nCTS自动流控使能位</b> 当nCTS自动流控使能时, 当nCTS输入激活时UART会发送数据到外部设备 (如nCTS输入没有激活UART不会发送数据到设备) 0 =禁用 nCTS 自动流控 1 = 使能nCTS 自动流控
[3]	<b>RTSAUTOEN</b>	<b>nRTS 自动流控使能位</b> 当nRTS自动流控使能时, 如果接收缓冲没有满(RXFULL (USCI_BUFSTS[1] =1), UART会解除nRTS信号。 0 =禁用 nRTS 自动流控 1 = 使能nRTS 自动流控 <b>注 :</b> 该位仅在RTSAUDIREN没有置位情况下有作用
[2]	<b>EVENPARITY</b>	<b>偶校验使能位</b> 0 =发送和检测每一个字中逻辑1是奇数。 1 =发送和检测每一个字中逻辑1是偶数。 <b>注:</b> 仅当PARITYEN置位时该位有作用
[1]	<b>PARITYEN</b>	<b>校验使能位</b> 在UART帧中, 该位定义校验位使能 0 =禁用校验位 1 =使能校验位
[0]	<b>STOPB</b>	<b>停止位</b> 在UART帧中, 该位定义停止位数目 0 = 1位停止位 1 = 2位停止位

USCI\_PROTIEN USCI 协议中断使能寄存器– UART

寄存器	偏移地址	R/W	描述	复位值
<b>USCI_PROTIEN</b>	USCIIn_BA+0x60	R/W	USCI 协议中断使能寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved					RLSIEN	ABRIEN	Reserved

位	描述	
[31:3]	<b>Reserved</b>	保留
[2]	<b>RLSIEN</b>	<p>接收线状态中断使能 0 =禁用接收线状态中断 1 =使能接收线状态中断 注: USCI_PROTSTS[7:5] 反映当前接收线状态中断事件</p>
[1]	<b>ABRIEN</b>	<p>自动波特率中断使能位 0 =禁用自动波特率中断 1 =使能自动波特率中断</p>
[0]	<b>Reserved</b>	保留

USCI PROTSTS USCI 协议状态寄存器– UART

寄存器	偏移地址	R/W	描述	复位值
USCI_PROTSTS	USCIIn_BA+0x64	R/W	USCI 协议状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved						CTSLV	CTSSYNCLV
15	14	13	12	11	10	9	8
Reserved				ABERRSTS	RXBUSY	ABRDETIF	Reserved
7	6	5	4	3	2	1	0
BREAK	FRMERR	PARITYERR	RXENDIF	RXSTIF	TXENDIF	TXSTIF	Reserved

位	描述
[31:18]	<b>Reserved</b> 保留
[17]	<b>CTSLV</b> <b>nCTS 引脚状态 (只读)</b> 该位被用作监控nCTS引脚输入的当前状态 0 = nCTS引脚为低电平 1 = nCTS引脚为高电平
[16]	<b>CTSSYNCLV</b> <b>nCTS 同步电平状态 (只读)</b> 该位被用作反映内部同步nCTS信号当前状态 0 = 内部同步 nCTS 为低 1 = 内部同步 nCTS 为高
[15:12]	<b>Reserved</b> 保留
[11]	<b>ABERRSTS</b> <b>自动波特率错误状态</b> 当自动波特率侦测计数器溢出时该位被置位。当自动波特率计数器溢出时，用户需要修改CLKDIV (USCI_BRGEN[25:16])值并且使能ABREN (USCI_PROTCTL[6])来再次侦测正确的波特率 0 = 自动波特率计数器没有溢出 1 = 自动波特率计数器溢出 <b>注 1:</b> 该位在ABRDETIF相同时间被置位。 <b>注 2:</b> 写1到ABRDETIF 或 ABERRSTS，清除该位
[10]	<b>RXBUSY</b> <b>RX 总线状态标志 (只读)</b> 该位反映接收器忙状态 0 = 接收器 处于空闲状态. 1 =接收器 处于忙状态.
[9]	<b>ABRDETIF</b> <b>自动波特率中断标志</b> 当自动波特率侦测在输入数据下降沿中间完成时，该位被置位。如果the ABRIEN

		(USCI_PROTCTL[6])置位，会产生自动波特率中断。当输入数据为0x55时，该位会被设置4次并且在输入总线下次下降沿之前被清除。 0 =自动波特率功能没有完成 1 =一位自动波特率功能已完成 注:该位可以写1清除
[8]	<b>Reserved</b>	保留
[7]	<b>BREAK</b>	<b>Break 标志</b> 每当接收数据输入（RX）保持在逻辑0状态，该状态持续时间超过了一整个字的传输（也就是：起始位+数据位+停止位 的时间），该位被置1. 0 =没有Break产生 1 =接收总线上出现Break状态 注:写1到BREAK, FRMERR 和 PARITYERR中该位被清除
[6]	<b>FRMERR</b>	<b>帧错误标志</b> 每当接收字符没有有效的停止位时，该位被置1.（也就是，仅更随在最后数据位或是校验位之后的停止位被侦测到是逻辑0） 0 =没有帧错误产生 1 =有帧错误发生 注:写1到BREAK, FRMERR 和 PARITYERR中该位被清除
[5]	<b>PARITYERR</b>	<b>校验错误标志</b> 每当接收字符没有有效的校验位时，该位被置1. 0 =没有校验错误 1 = 有校验错误发生 注:写1到BREAK, FRMERR 和 PARITYERR中该位被清除
[4]	<b>RXENDIF</b>	<b>接收结束中断标志</b> 0 =没有接收结束中断状态发生 1 = 有接收结束中断状态发生 注:写1清除该位
[3]	<b>RXSTIF</b>	<b>接收起始中断标志</b> 0 =没有接收起始中断状态发生 1 = 有接收起始中断状态发生 注:写1清除该位
[2]	<b>TXENDIF</b>	<b>发送结束中断标志</b> 0 =没有发送结束中断状态发生 1 = 有发送结束中断状态发生 注:写1清除该位
[1]	<b>TXSTIF</b>	<b>发送起始中断标志</b> 0 =没有发送起始中断状态发生 1 = 有发送起始中断状态发生 注1:写1清除该位 注 2:当没有数据在发送缓冲时用户可以用此标志位来加载下次发送数据
[0]	<b>Reserved</b>	保留

## 6.25 USCI - SPI 模式

### 6.25.1 概述

USCI控制器的SPI协议适用于同步串行数据通信并且支持全双工传输。支持主机和从机操作4线接口模式。USCI控制器的SPI模式在接收外设数据时执行的时串行到并行的转换，在发送数据到外设时执行的是并行到串行的转换。设置FUNMODE (USCI\_CTL[2:0]) = 0x1，选择SPI模式。

通过设置SLAVE (USCI\_PROTCTL[0])来选择SPI协议运行在主机或是从机模式。下图展示了主机和从机模式的应用。

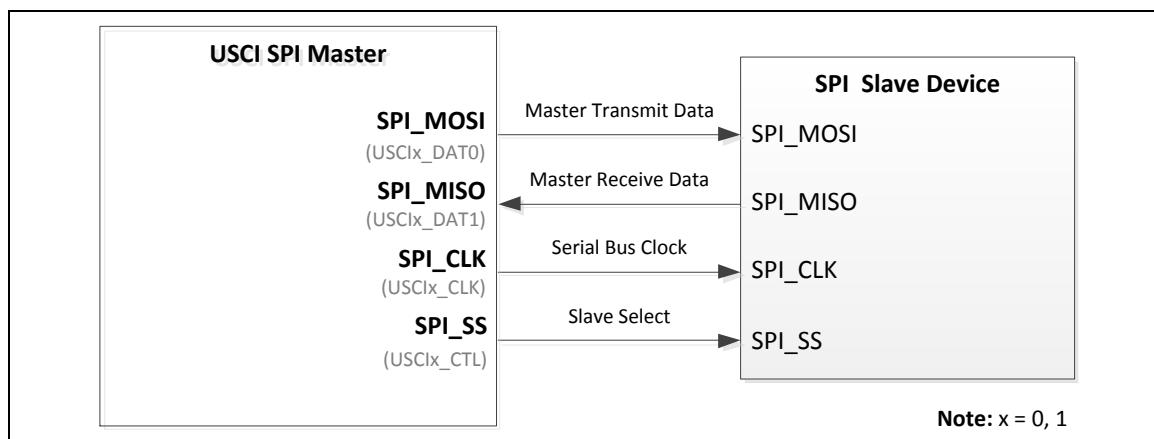


图 6.25-1 SPI 主机模式应用框图

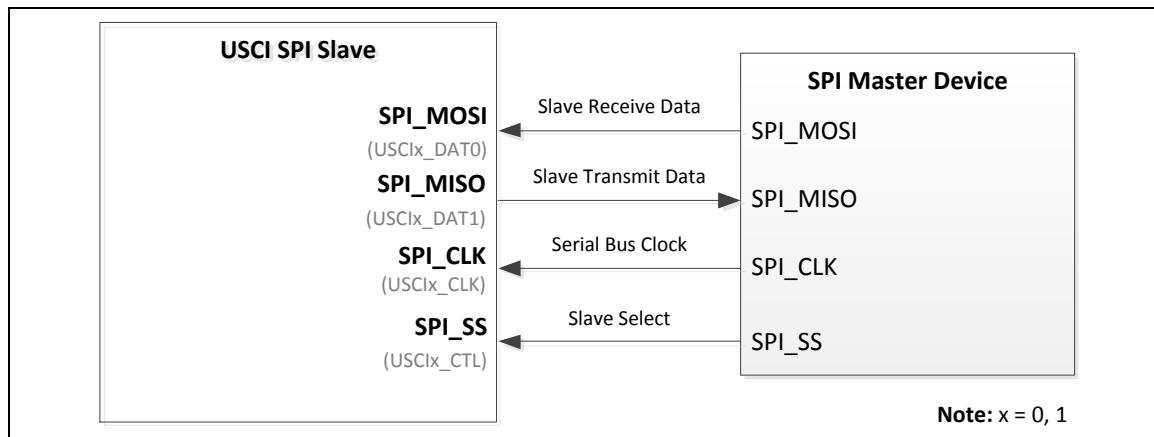


图 6.25-2 SPI 从机模式应用框图

### 6.25.2 特性

- 支持主机或从机模式操作（最大频率—主机==  $f_{PCLK}/2$ , 从机<  $f_{PCLK}/5$ ）
- 传输字的长度在4到16位可编程
- 支持一组发送缓冲和两组接收缓冲
- 支持MSB优先或是LSB优先传输

- 支持字传输暂停功能
- 支持PDMA传输
- 支持3-线，没有从机选择信号
- 从机模式下支持片由选信号唤醒功能
- 支持一个数据通道的半双工传输

### 6.25.3 框图

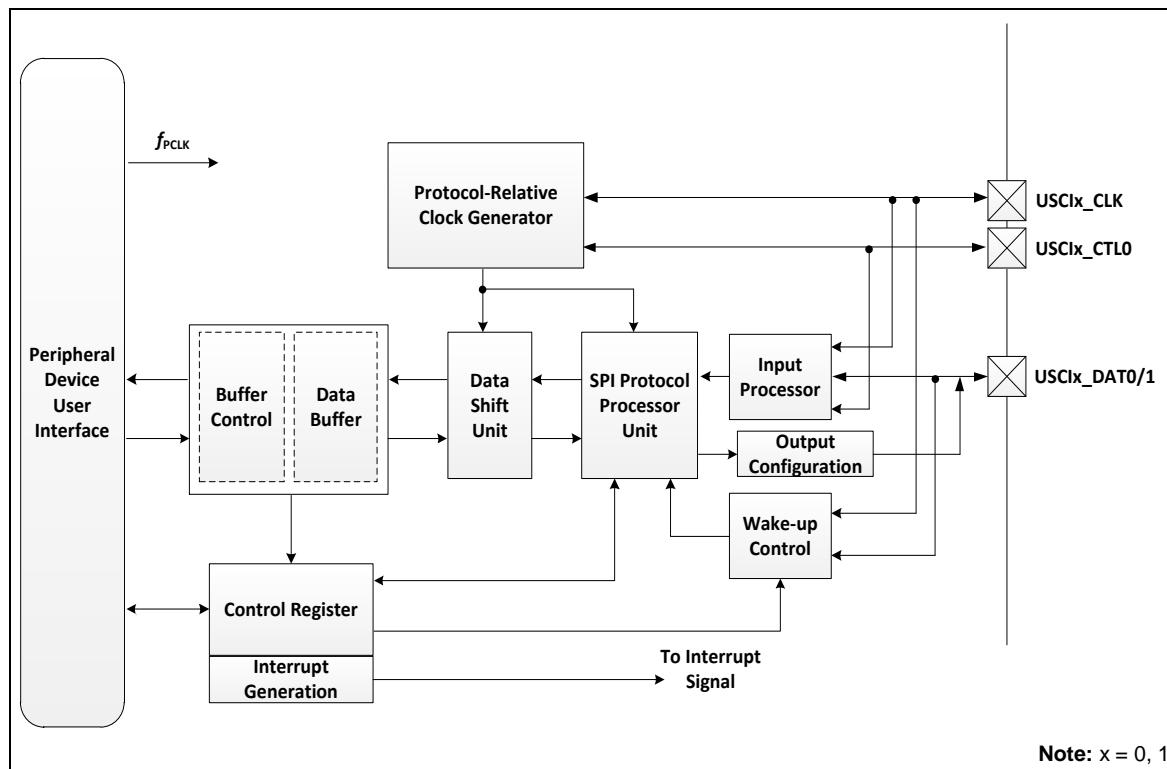


图 6.25-3 USCI SPI 模式框图

### 6.25.4 基本配置

#### 6.25.4.1 USCI0 SPI 基本配置

- 时钟源配置
  - 使能USCI0时钟的控制位是 CLK\_APBCLK1[8]中的USCI0CKEN位
  - 设置USCI\_CTL[2:0]=3'b001, 使能USCI0\_SPI功能
- 复位配置
  - 复位USCI0的控制位是SYS\_IPRST2[8]中的USCI0RST位
- 引脚配置

组	引脚名	GPIO	MFP
---	-----	------	-----

USCI0	USCI0_CLK	PD.0	MFP3
		PB.12	MFP5
		PA.11	MFP6
		PE.2	MFP7
	USCI0_CTL0	PD.4	MFP3
		PC.14	MFP5
		PC.13	MFP6
		PE.6	MFP7
	USCI0_DAT0	PD.1	MFP3
		PB.13	MFP5
		PA.10	MFP6
		PE.3	MFP7
	USCI0_DAT1	PD.2	MFP3
		PB.14	MFP5
		PA.9	MFP6
		PE.4	MFP7

#### 6.25.4.2 USCI1 SPI 基本配置

- 时钟源配置
  - 使能USCI1时钟的控制位是CLK\_APBCLK1[9]中的USCI1CKEN位
  - 设置USCI\_CTL[2:0]=3'b001，使能USCI1\_SPI功能
- 复位配置
  - 复位USCI1的控制位是SYS\_IPRST2[9]中的USCI1RST位
- 引脚配置

组	引脚名	GPIO	MFP
USCI1	USCI1_CLK	PB.8	MFP4
		PD.7, PE.12	MFP6
		PB.1	MFP8
	USCI1_CTL0	PB.10	MFP4
		PD.3, PE.9	MFP6
		PB.5	MFP8
	USCI1_DAT0	PB.7	MFP4
		PD.5, PE.10	MFP6
		PB.2	MFP8
	USCI1_DAT1	PB.6	MFP4

		PD.6, PE.11	MFP6
		PB.3	MFP8

### 6.25.5 功能描述

#### 6.25.5.1 USCI共同功能描述

具体信息参考 6.23.4。

#### 6.25.5.2 信号描述

工作在主机模式下的设备控制一次数据传输的开始和结束，并且SPI总线时钟和片选信号也是有主机设备产生。片选信号反映了一次数据的起始和结束，主机可以使用它使能对从机的发送或是接收操作。SPI通信信号如下

SPI 模式	接收数据	发送数据	串行总线时钟	从机选择
全双工 SPI 主机	SPI_MISO (USCIx_DAT1)	SPI_MOSI (USCIx_DAT0)	SPI_CLK (USCIx_CLK)	SPI_SS (USCIx_CTL0)
全双工 SPI 从机	SPI_MOSI (USCIx_DAT0)	SPI_MISO (USCIx_DAT1)	SPI_CLK (USCIx_CLK)	SPI_SS (USCIx_CTL0)
半双工 SPI 主机/从机	SPI_MOSI (USCIx_DAT0)	SPI_MOSI (USCIx_DAT0)	SPI_CLK (USCIx_CLK)	SPI_SS (USCIx_CTL0)

表 6.25-1 SPI 通信信号

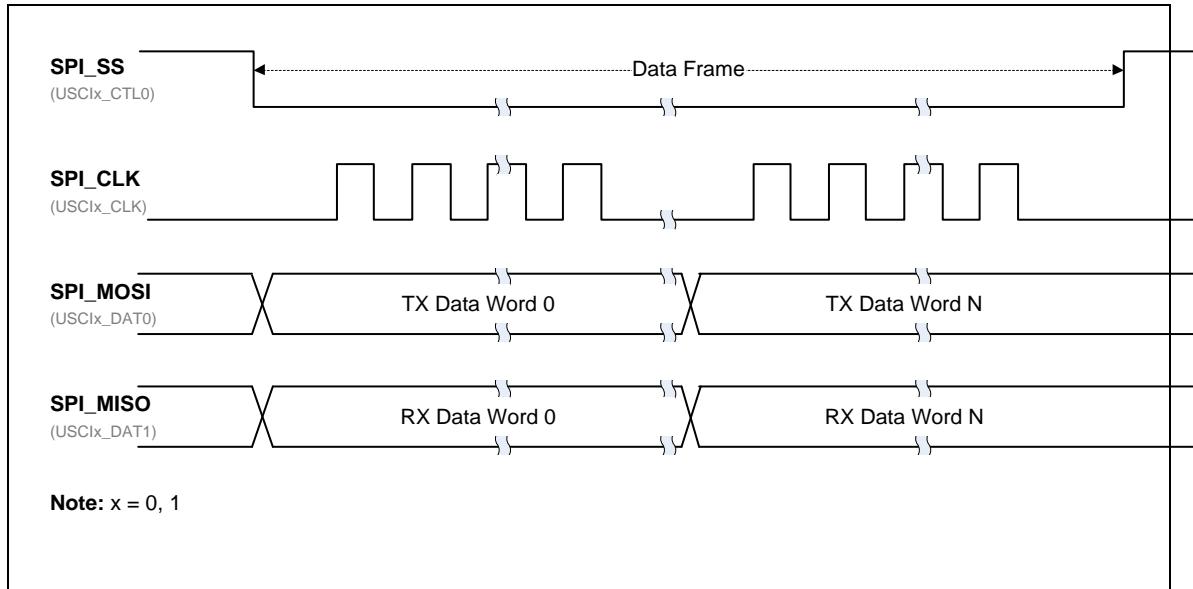


图 6.25-44-写 全双工 SPI 通信信号 (主机模式)

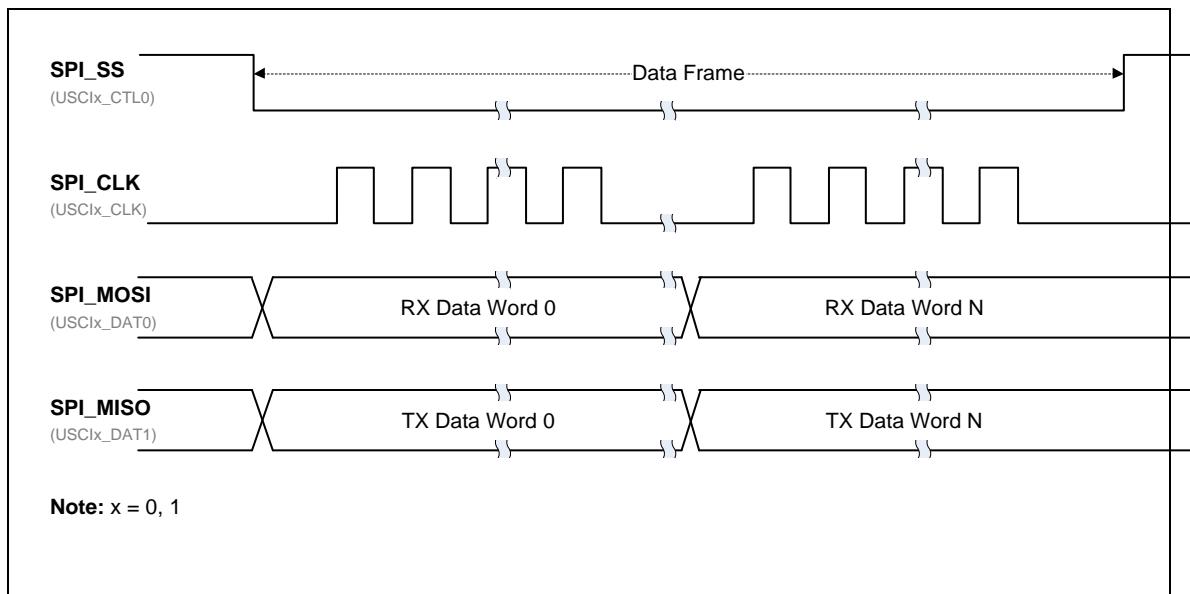


图 6.25-54-写 全双工 SPI 通信信号 (从机模式)

### 6.25.5.3 串行总线时钟配置

USCI控制器需要外设时钟来驱动USCI的逻辑单元来执行数据传输。外设时钟频率等于PCLK频率。

在主机模式下，SPI总线时钟频率是由相关协议时钟发生器决定的。SPI总线时钟就是SPI时钟。SPI时钟频率是 $f_{SAMP\_CLK}$ 的一半， $f_{SAMP\_CLK}$ 是由SPCLKSEL (USCI\_BRGEN[3:2])决定的。相关协议时钟发生器的详细内容请参考6.23.4。

在从机模式下，SPI总线时钟是有片外主机设备提供的。SPI从机外设时钟频率 $PCLK$ 必须快于SPI主机设备连接的串行总线时钟速率的5倍（也就是串行总线时钟速率 $< 1/5$ 外设时钟 $f_{PCLK}$ ）。

在SPI协议下，SCLKMODE (USCI\_PROTCTL[7:6]) 既定义了串行总线时钟的空闲状态也被用作定义发送与接收数据的串行时钟边沿。SCLKMODE配置对主机和从机是一样的。串行总线时钟的四种模式配置如下

SCLKMODE [1:0]	SPI 时钟空闲状态	发送时序	接收时序
0x0	低	下降沿	上升沿
0x1	低	上升沿	下降沿
0x2	高	上升沿	下降沿
0x3	高	下降沿	上升沿

表 6.25-2 串行总线时钟配置

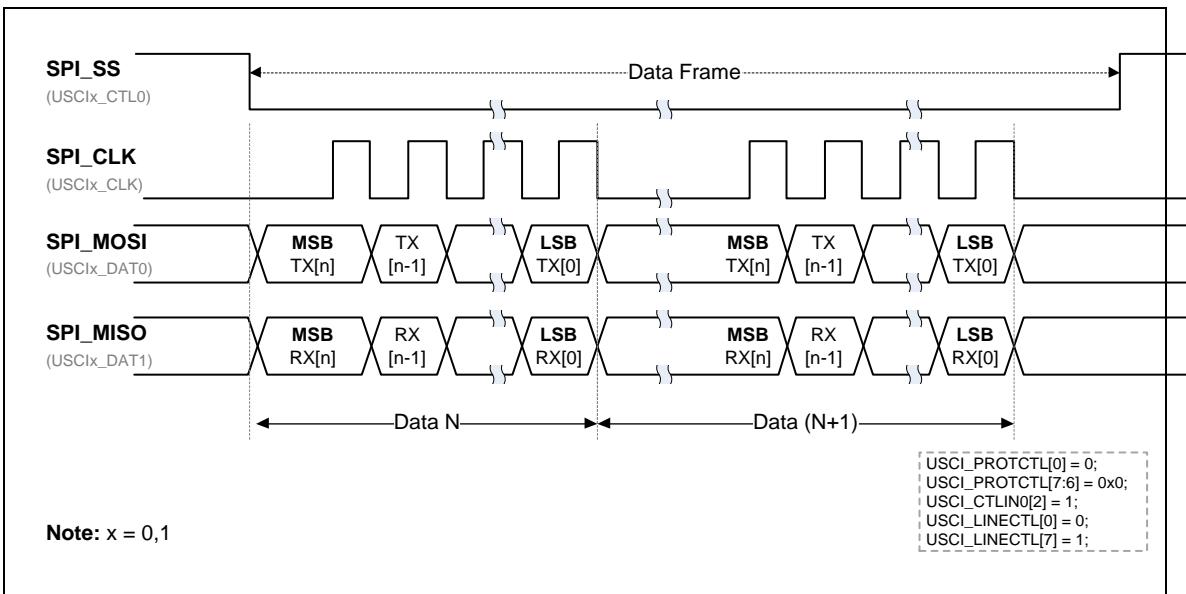


图 6.25-6 不同时钟模式下的SPI通信 (SCLKMODE=0x0)

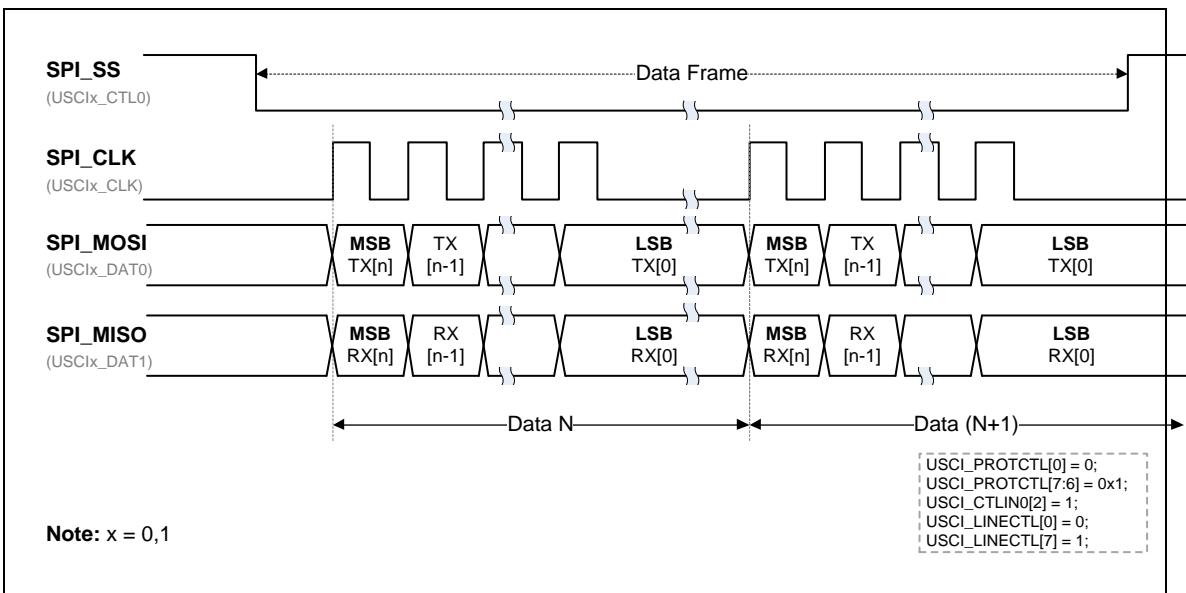


图 6.25-7 不同时钟模式下的SPI通信(SCLKMODE=0x1)

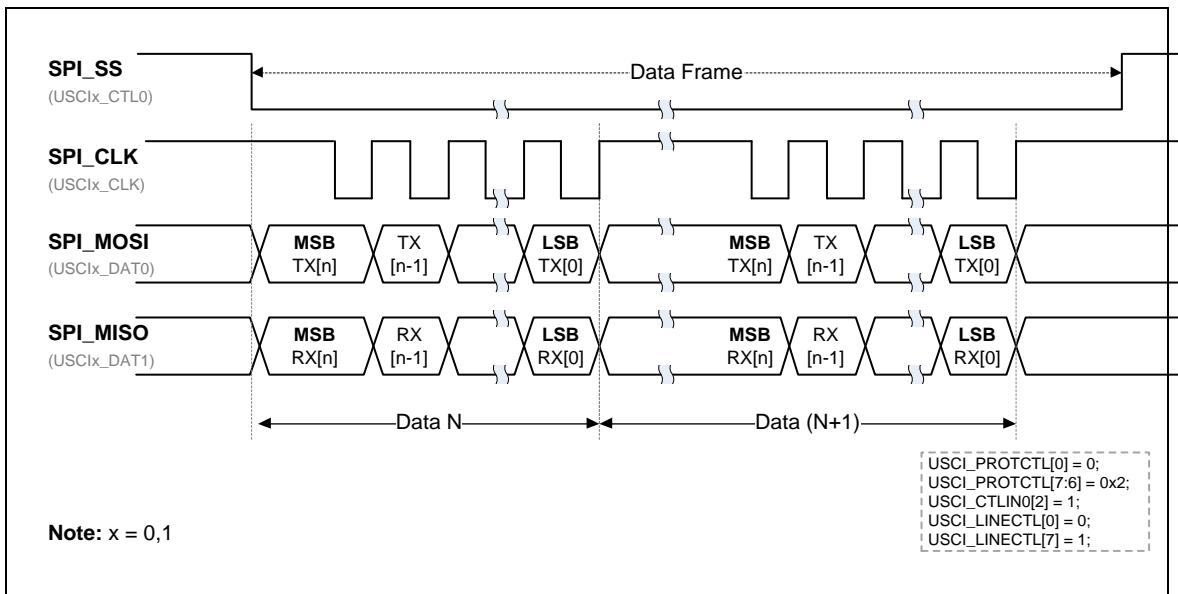


图 6.25-8 不同时钟模式下的SPI通信(SCLKMODE=0x2)

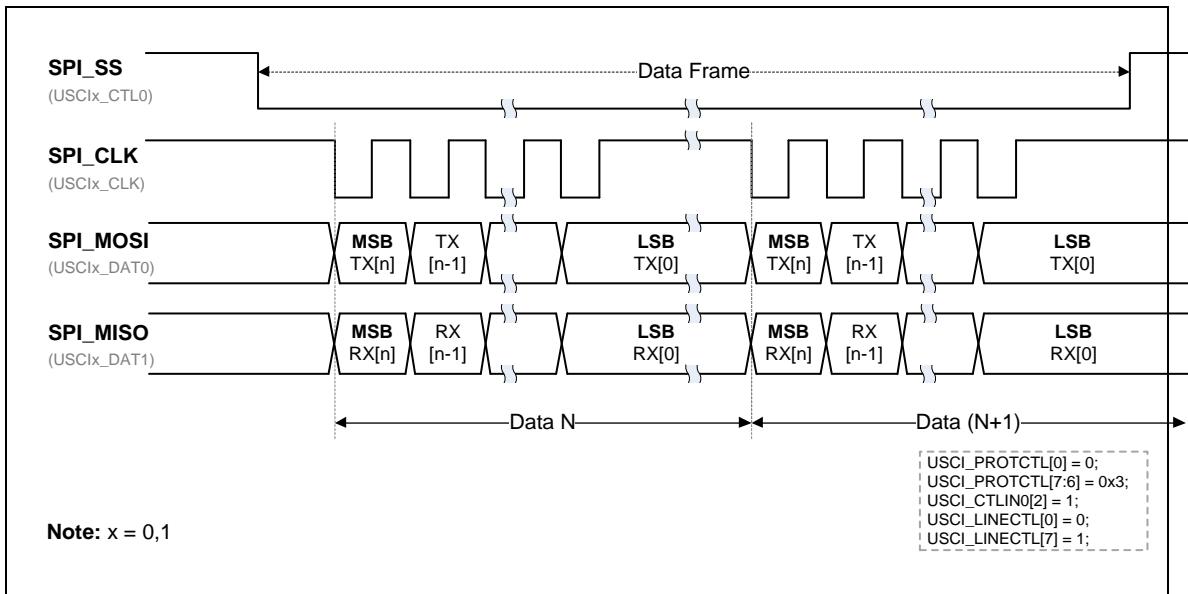


图 6.25-9 不同时钟模式下的SPI通信(SCLKMODE=0x3)

#### 6.25.5.4 从机选择信号

SPI协议的从机片选信号默认是高电平激活。在SPI主机模式下，USCI控制器可以通过从机选择引脚SPI\_SS (USCIx\_CTL0)来驱动控制外部SPI 从机设备。在SPI从机模式下，接收片选信号可以通过ININV (USCI\_CTLIN0[2])反相。

如果外部SPI主机设备的片选信号是低电平激活，那么从机设备的ININV (USCI\_CTLIN0[2])的设置应该设置为1来反相输入控制信号。如果USCI工作在SPI 主机模式下，那些需要片选激活信号是低

电平的外部SPI从机设备来说，主机输出从机选择反相CTLOINV (USCI\_LINECTL[7])也要置1。

从机选择激活边沿与第一个SPI时钟输入边沿之间会超过2个USCI外设时钟周期。

为了正确地检测到帧结束，在连续两帧之间SPI从机输入片选信号必须保持非激活状态时间至少在2个USCI外设时钟周期

#### 6.25.5.5 发送和接收数据

在USCI控制器SPI协议下，接收/发送数据位长度可以在DWIDTH (USCI\_LINECTL[11:8])中定义，在SPI通信中发送和接收数据长度最大可以配置为16位。

LSB位 (USCI\_LINECTL[0]) 定义传输数据位的顺序。如果LSB位被置1，那么传输的数据顺序是LSB优先。如果LSB位被清0，那么传输的数据顺序是MSB优先。

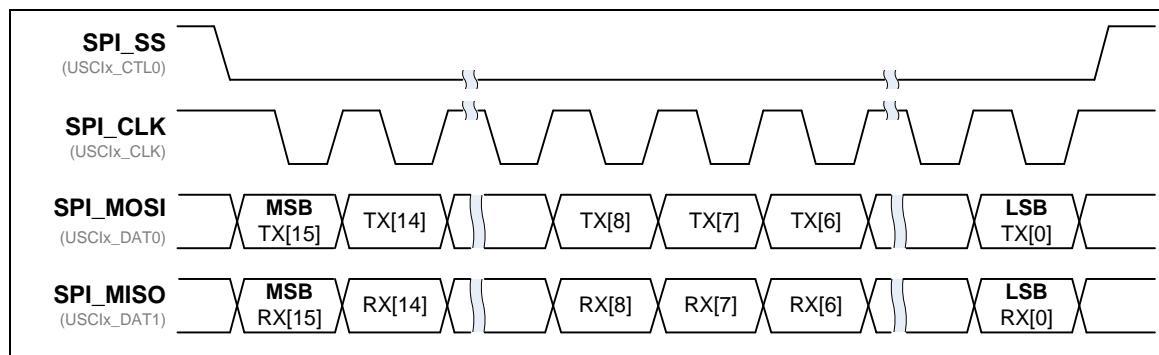


图 6.25-10 MSB优先格式下16位数据长度的一次字传输

#### 6.25.5.6 字暂停

主机模式下两个连续字传输之间可以由SUSPITV (USCI\_PROTCTL[11:8])提供一个0.5 ~ 15.5 SPI时钟周期可编程的暂停间隔的功能。暂停间隔指的是之前传输的最后一个时钟边沿到下次传输的第一个时钟边沿之间的间隔。SUSPITV (USCI\_PROTCTL[11:8])默认值是0x3(3.5 SPI时钟周期)。

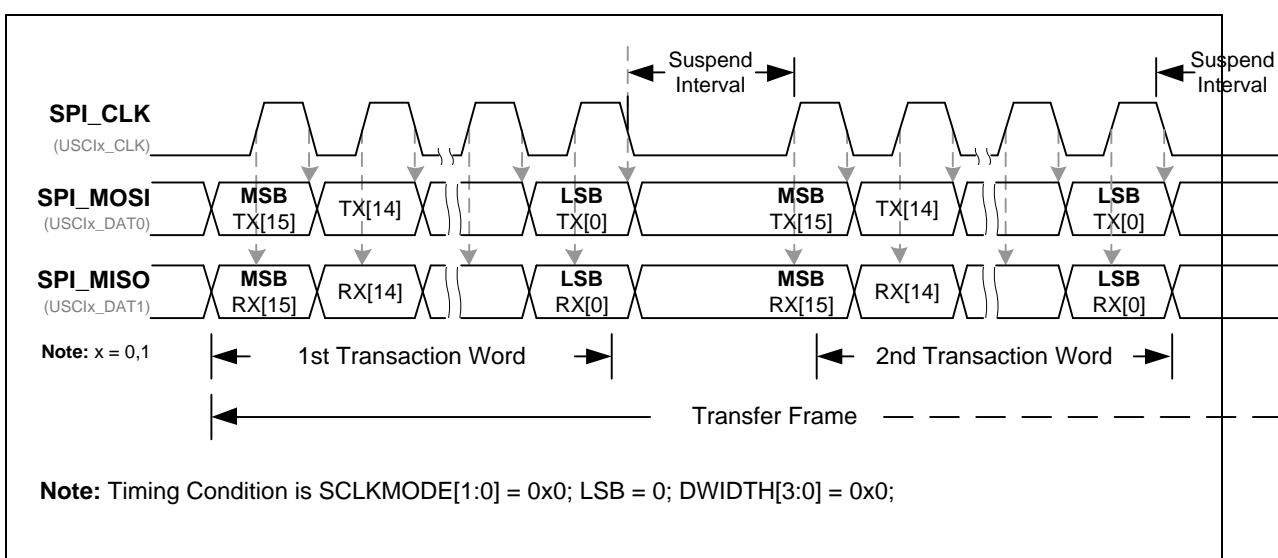


图 6.25-11 两组传输间的字暂停

### 6.25.5.7 自动片选功能

AUTOSS (USCI<sub>x</sub>\_PROTCTL[3]) 被用作 SPI 主机模式来使能自动片选功能。如果 AUTOSS (USCI<sub>x</sub>\_PROTCTL[3]) 被置位，片选信号会自动地产生并且 SS (USCI<sub>x</sub>\_PROTCTL[2]) 设置值不会影响输出从机选择 USCI<sub>x</sub>\_CTL0 线。也就意味着当向发送缓冲写入数据启动 SPI 数据传输的时候，片选信号会变成有效状态。在所有传输结束或是如果 SUSPITV (USCI<sub>x</sub>\_PROTCTL[11:8]) 大于等于 3 时一次字传输完成之后，片选信号会失效。

如果 AUTOSS 位 (USCI<sub>x</sub>\_PROTCTL[3]) 被清除，通过设置/清除 SS (USCI<sub>x</sub>\_PROTCTL[2]) 来使片选信号 USCI<sub>x</sub>\_CTL0 引脚有效/无效。内部片选信号激活电平是高，CTLOINV (USCI<sub>x</sub>\_LINECTL[7]) 可以被用作反相片选信号。

在 SPI 主机模式，如果 SUSPITV (USCI<sub>x</sub>\_PROTCTL[11:8]) 值少于 3 并且 AUTOSS (USCI<sub>x</sub>\_PROTCTL[3]) 被置 1，在连续两周字传输之间片选信号保持有效状态。

在 SPI 从机模式，在两组连续传输之间为了识别片选信号的失效状态，接收片选信号非激活的周期必须大于 2 个外设时钟周期。

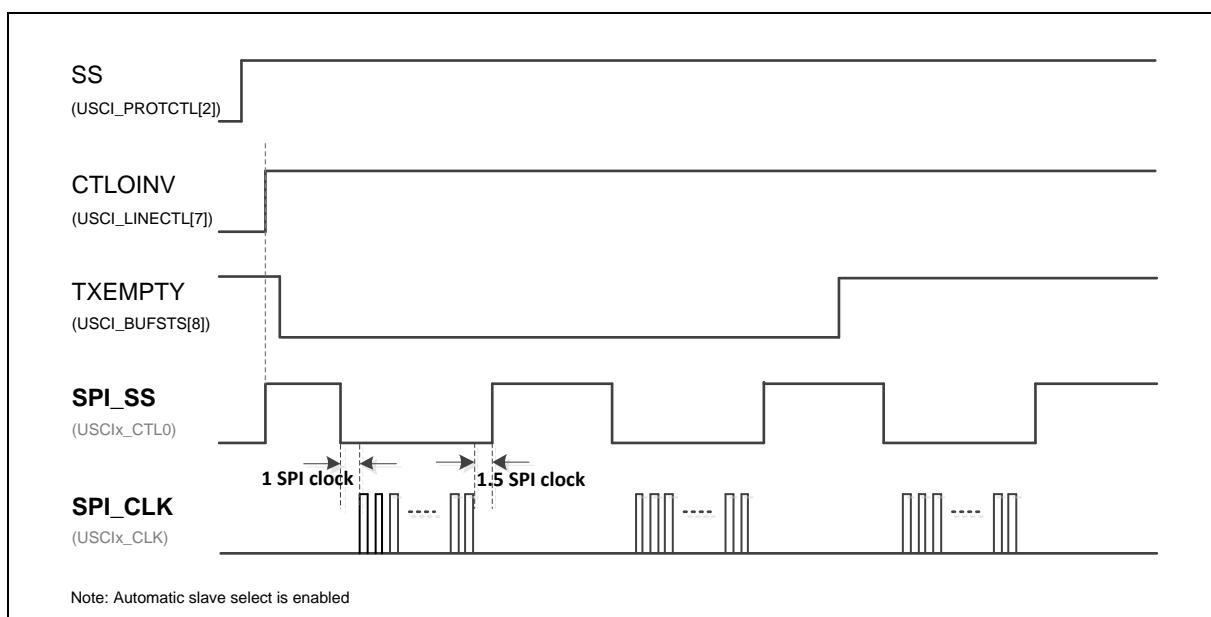


图 6.25-12 自动从机选择 (SUSPITV  $\geq$  0x3)

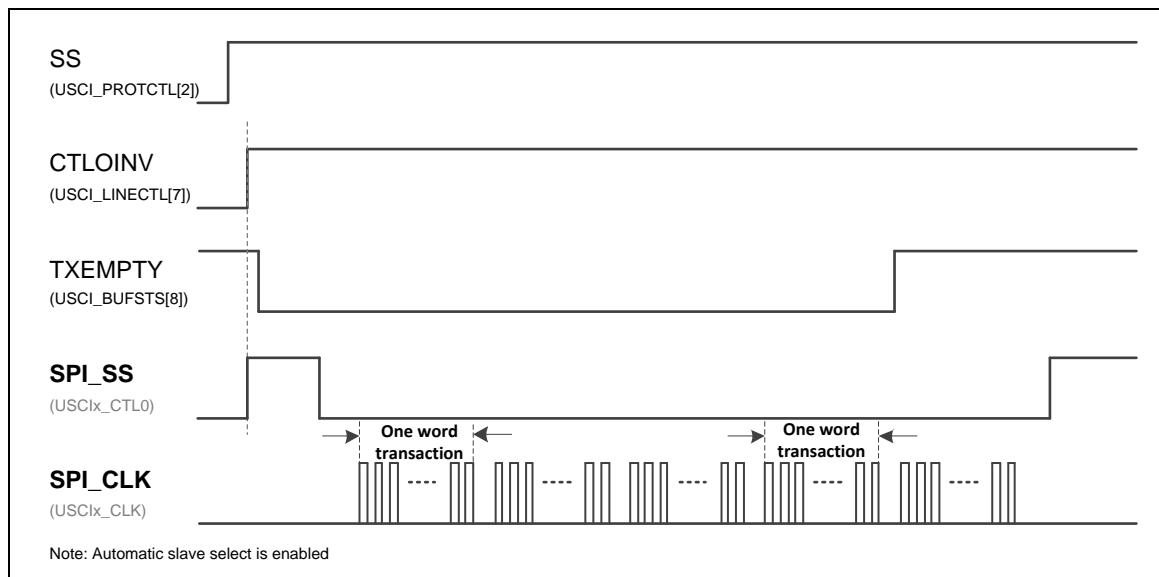


图 6.25-13 自动从机选择(SUSPITV &lt; 0x3)

#### 6.25.5.8 从机3-线模式

当通过软件设置SLV3WIRE (USCI\_PROTCTL[1])使能从机3-线模式的时候，USCI SPI通信在从机模式下可以在没有片选信号下工作。仅三个引脚 SPI\_CLK (USCIx\_CLK)，SPI\_MOSI (USCIx\_DAT0)，和 SPI\_MISO ( USCIx\_DAT1) 被用作与 SPI 主机通信。当 SLV3WIRE (USCI\_PROTCTL[1])被置1时，在通过设置FUNMODE(USCI\_CTL [2:0])为0x1使能SPI协议之后 SPI从机就好发送/接收数。

#### 6.25.5.9 数据传输模式

USCI控制器支持全双工SPI传输和一个数据通道半双工SPI传输

- 全双工SPI传输

在全双工SPI传输下，有两根数据引脚。一根用来发送数据，另外一根用来接收数据。因此数据发送和接收可以同时进行。

SCLKMODE (USCI\_PROTCTL[7:6])定义数据移位输出信号在USCIx\_DAT0引脚上的转变时序。转变可能发生在相应的SPI总线时钟的边沿或是片选信号激活边沿。一个数据字的最后数据位的电平会一直保持在USCIx\_DAT0引脚上直到下次串行总线时钟的边沿和下次数据开始。

- 一个数据通道半双工SPI传输

在一个数据通道半双工SPI传输下，只有一根数据引脚用于数据传输。因此数据发送和接收不能同时进行。数据移位方向由PORTDIR (USCI\_TXDAT[16])决定。更多信息参考寄存器描述。

- 一个数据通道半双工SPI传输类似于全双工SPI协议。所有传输数据时序等同全双工SPI传输。

下图是一个输出数据通道和一个输入数据通道于外部设备进行半双工传输的框图。

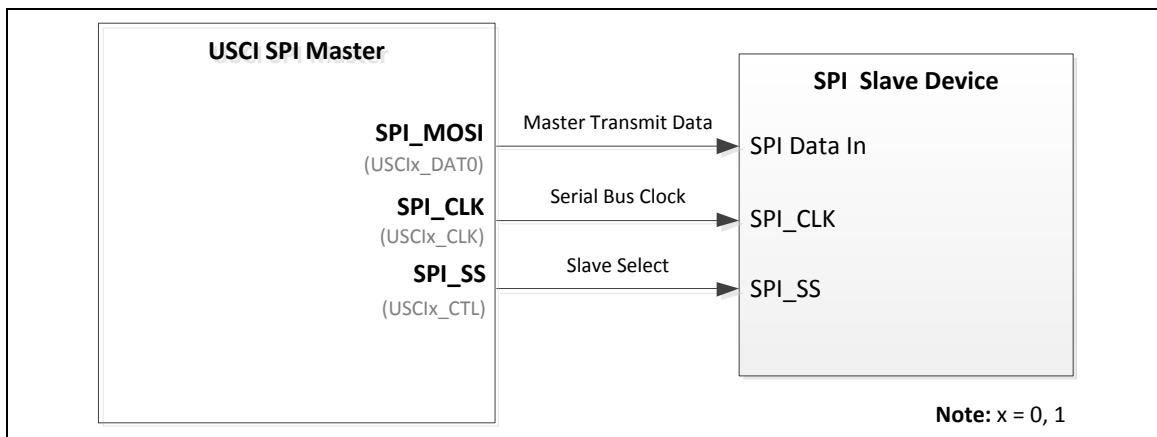


图 6.25-14 一个输出数据通道半双工通信 (SPI 主机模式)

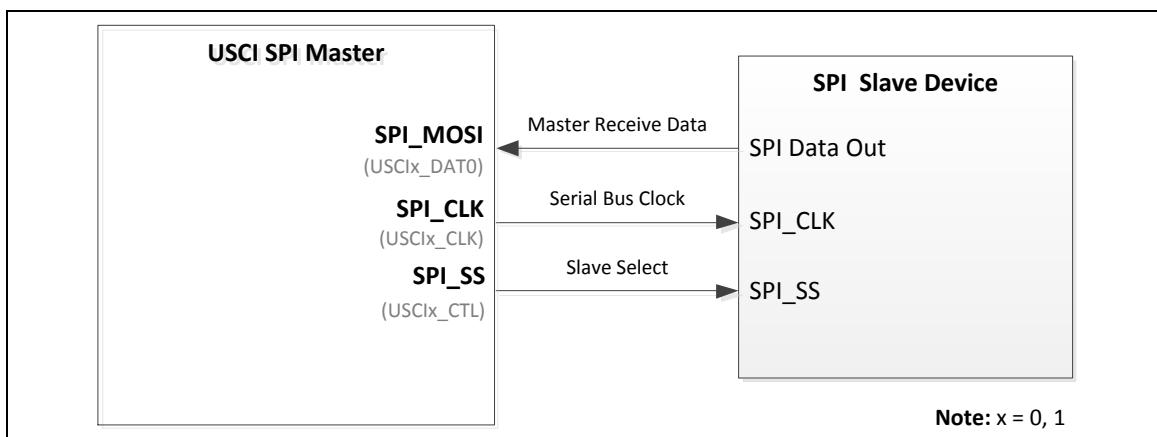


图 6.25-15 一个输出数据通道半双工通信(SPI 主机模式)

一个数据通道半双工传输模式是通过设置 TSMSEL[2:0] (USCI\_PROTCTL[14:12]) 和 PORTDIR (USCI\_TXDAT[16]) 来配置的。当 TSMSEL (USCI\_PROTCTL[14:12]) 设置为 0x4 时，一个数据通道半双工传输模式被选择。PORTDIR (USCI\_TXDAT[16]) 被用作定义相应传输数据的方向。当 PORTDIR 位被设置为 0 时，USCI 控制器发送相应数据到外部 SPI 设备。当 PORTDIR 位被设置为 1 时，USCI 控制器从外部 SPI 设备读相应的数据。

比如，在一个数据通道半双工传输模式下，如果 PORTDIR=0，USCI SPI 用 USCIx\_DAT0 引脚发送数据；如果 PORTDIR=1，USCI SPI 用 USCIx\_DAT0 引脚接收数据。

#### 6.25.5.10 中断

### 数据传输中断

- 发送起始中断

一次数据发送的第一个数据位的起始之后，中断事件TXSTIF (USCI\_PROTSTS[1])置位。该位可以写1清除。

- 发送结束中断

存储在发送缓冲的最后发送数据的最后数据位结束之后，中断事件TXENDIF (USCI\_PROTSTS[2])置位。该位可以写1清除。

- 接收起始中断

一次接收数据的第一个数据位的起始之后，中断事件RXSTIF (USCI\_PROTSTS[3])置位。该位可以写1清除。

- 接收结束中断

接收数据的最后数据位结束之后，中断事件RXENDIF (USCI\_PROTSTS[4])置位。该位可以写1清除。

### 相关协议中断

- SPI从机选择中断

在SPI从机模式下，当SLAVE (USCI\_PROTCTL [0])被置1并且从机检测到片选信号激活或非激活的时候，将会有从机选择激活 SSACTIF (USCI\_PROTSTS[9]) 和非激活 SSINAIF (USCI\_PROTSTS[8]) 中断标志被置1。如果SSINAIE (USCI\_PROTIEN[0]) 或 SSACTIEN (USCI\_PROTIEN[1])置1，SPI控制器会产生一个中断。因为SPI功能的内部从机片选信号是高电平激活，所以ININV (USCI\_CTLINO[2])可以被用作反相来自低电平激活设备的片选信号。

- 从机超时中断

在SPI从机模式下，在一次字传输期间将会有从机超时功能被用于了解有没串行时钟输入。从机超时功能使用时序测量计数器来进行从机超时周期计算，该超时周期由SLVTOCNT (USCI\_PROTCTL[25:16])定义。TMCNTSRC (USCI\_BRGEN[5])被用于时序测量计数器时钟频率的选择。

在SPI从机模式下当时序测量计数器由TMCNTEN (USCI\_BRGEN[4])使能并且SLVTOCNT (USCI\_PROTCTL[25:16])不为0的情况下，在每次接收字数据的第一个输入串行时钟之后，时序测量计数器会开始计数。计数器会在接收到随后的输入串行时钟时被复位，然后继续计数。最后，在当前字传输完成之后，时序测量计数器会被清除并且停止。在一个字传输完成之前，如果超时计数器的值大于等于SLVTOCNT (USCI\_PROTCTL[25:16])的值，从机超时中断事件会发生并且SLVTOIF (USCI\_PROTSTS[5])会被置1.

### 相关缓冲中断

如果USCI控制器中有发送/接收缓冲，相关缓冲中断有效。

- 接收缓冲上溢出中断

如果接收缓冲上溢出事件发生，RXOVIF (USCI\_BUFSTS[3])会被置1。可以写1清除。

- 发送缓冲下溢出中断

如果发送缓冲下溢出事件发生，TXUDRIF (USCI\_BUFSTS[11])会被置1。可以写1清除。

### 6.25.5.11 时序图

USCI SPI协议的从机片选信号默认值是高电平激活，可以设置CTLOINV (USCI\_LINECTL[7])反相。

串行时钟的空闲状态和被用于发送/接收数据的串行总线时钟边沿可以通过设置SCLKMODE (USCI\_PROTCTL[7:6])来配置。传输数据字的位宽度由DWIDTH (USCI\_LINECTL[11:8])决定，数据位传输顺序由LSB (USCI\_LINECTL[0])决定。主机/从机操作和相关设置如下面四组SPI时序图所示。

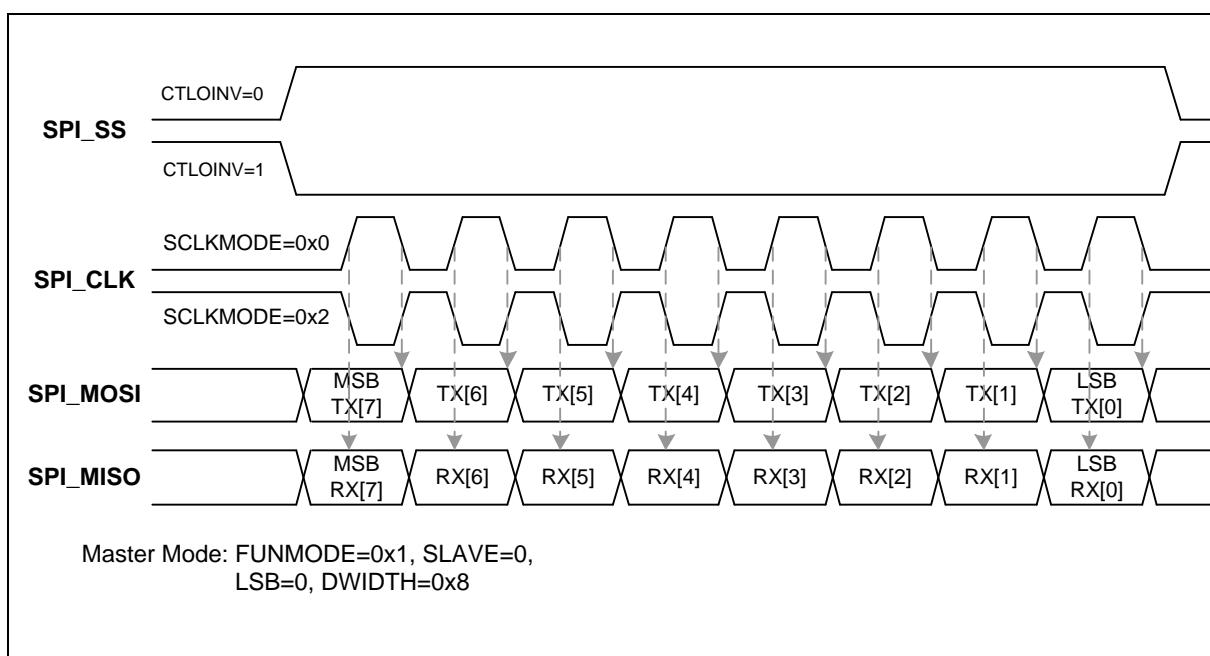


图 6.25-16 主机模式下的 SPI 时序

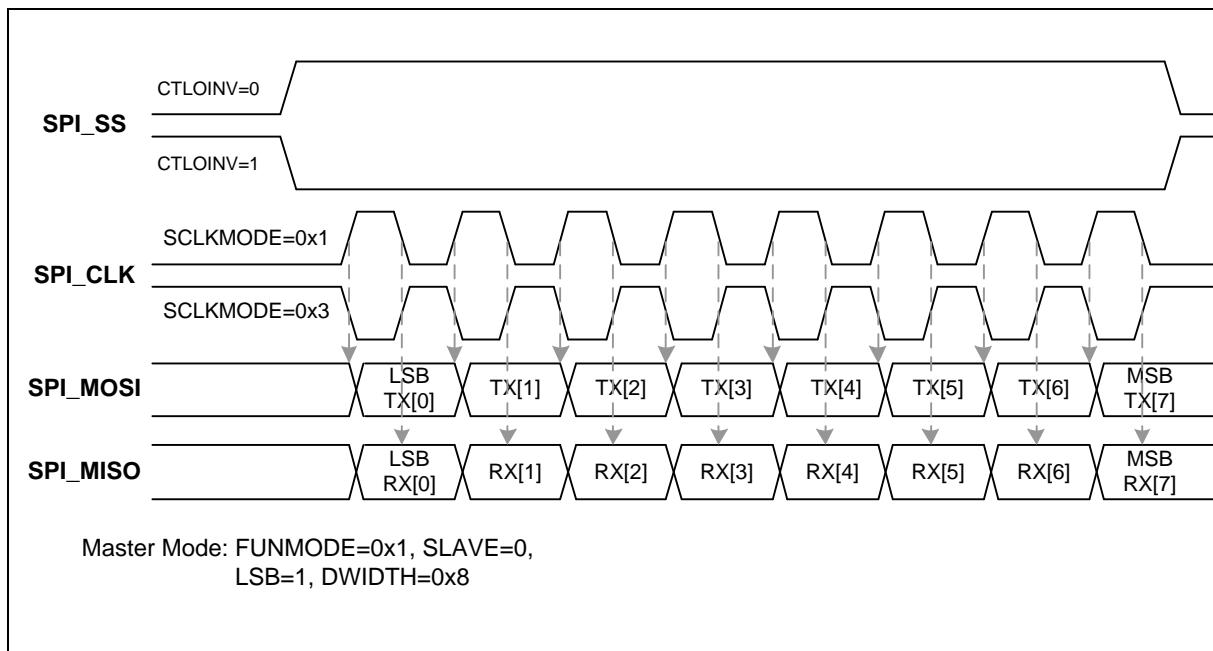


图 6.25-17 主机模式下的 SPI 时序(串行总线时钟的反相位)

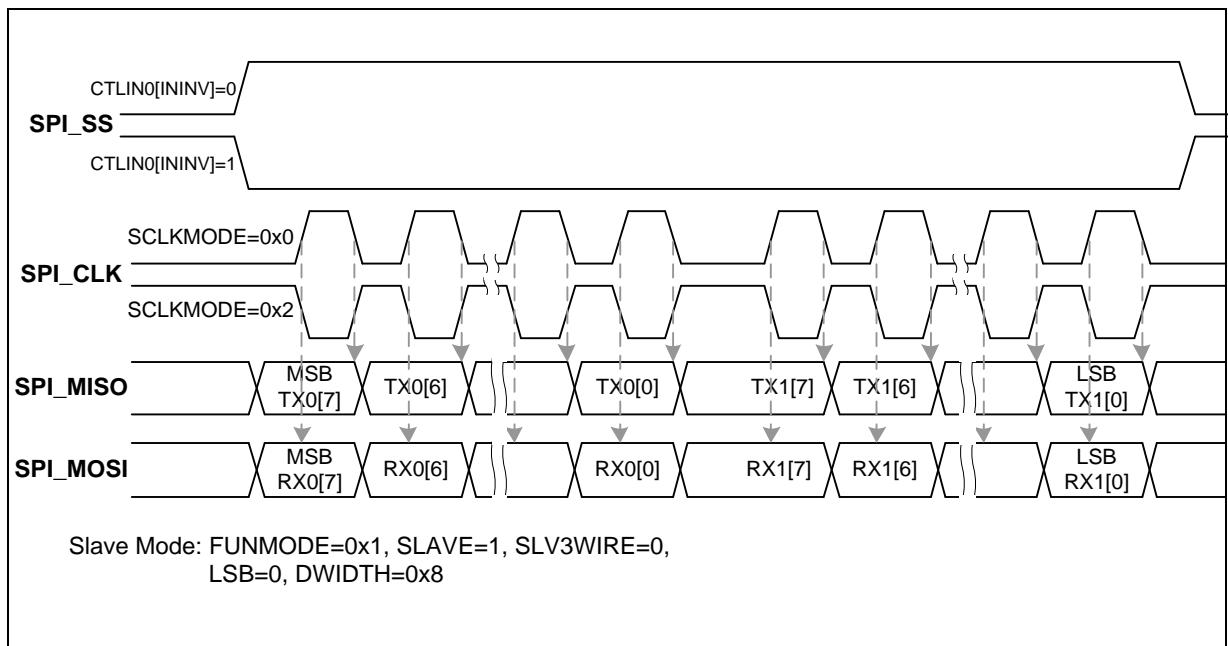


图 6.25-18 从机模式下的 SPI 时序

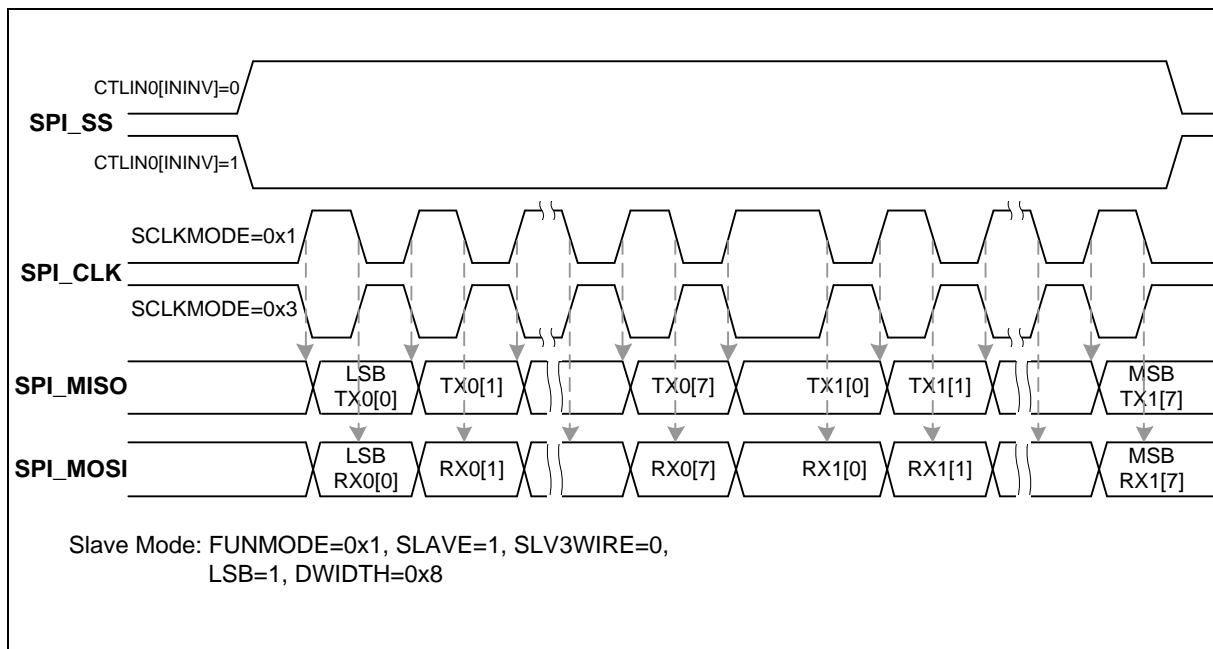


图 6.25-19 从机模式下的 SPI 时序(串行总线时钟的反相位)

### 6.25.5.12 编程流程

该章节描述了USCI SPI数据传输的编程流程

主机模式

1. 设置CLK\_APBCLK1寄存器，使能USCI外设时钟
2. 通过设置相关多功能控制寄存器配置用配置用户指定的引脚作为USCI功能引脚
3. 设置FUNMODE (USCI\_CTL[2:0])为1来选择SPI模式
4. 设置USCI\_BRGEN寄存器确定SPI总线时钟频率
5. 依据用户应用需求，配置设置如下
  - CTLOINV (USCI\_LINECTL[7]):如果从机片选信号激活时低电平，设置该位为1，否则设置为0
  - DWIDTH (USCI\_LINECTL[11:8]): 数据宽度设置
  - LSB (USCI\_LINECTL[0]): LSB优先还是MSB 优先
  - TSMSEL (USCI\_PROTCTL[14:12]):全双工SPI传输或一个通道半双工SPI传输
  - SCLKMODE (USCI\_PROTCTL[7:6]):确定时钟时序
  - AUTOSS (USCI\_PROTCTL[3]):是否使能自动从机片选功能
  - SLAVE (USCI\_PROTCTL[0]):主机模式设置为0

6. 设置PROTEN (USCI\_PROTCTL[31])为1来使能SPI协议

7. 如果禁用(AUTOSS=0)自动从机片选功能，在数据传输前设置SS

(USCI\_PROTCTL[2])为1; 设置SS为0来取消从机选择信号

8. 写USCI\_TXDAT寄存器来触发SPI传输。在半双工SPI传输中，数据引脚方向通过设置PORTDIR (USCI\_TXDAT[16])来确定
9. 只要RXEMPTY (USCI\_BUFSTS[0])为0，用户就可以通过读USCI\_RXDAT寄存器来获取接收的数据。

从机模式

1. 设置CLK\_APBCLK1寄存器，使能USCI外设时钟
2. 通过设置相关多功能控制寄存器配置用配置用户指定的引脚作为USCI功能引脚
3. 设置FUNMODE (USCI\_CTL[2:0])为1来选择SPI模式
4. 依据用户应用需求，配置设置如下
  - ININV (USCI\_CTLIN0[2]):如果从机片选信号激活时低电平，设置该位为1，否则设置为0
  - DWIDTH (USCI\_LINECTL[11:8]): 数据宽度设置
  - LSB (USCI\_LINECTL[0]): LSB优先还是MSB 优先
  - TSMSEL (USCI\_PROTCTL[14:12]):全双工SPI传输或一个通道半双工SPI传输
  - SCLKMODE (USCI\_PROTCTL[7:6]): 确定时钟时序
  - SLAVE (USCI\_PROTCTL[0]): 从机模式设置为1
5. 设置PROTEN (USCI\_PROTCTL[31])为1来使能SPI协议
6. 写USCI\_TXDAT寄存器来触发SPI传输。在半双工SPI传输中，数据引脚方向通过设置PORTDIR (USCI\_TXDAT[16])来确定
7. 只要RXEMPTY (USCI\_BUFSTS[0])为0，用户就可以通过读USCI\_RXDAT寄存器来获取接收的数据。只要TXFULL (USCI\_BUFSTS[9])为0，下次发送的数据就可以写入到USCI\_TXDAT寄存器中。

#### 6.25.5.13 唤醒功能

在SPI模式下USCI控制器支持唤醒系统功能。SPI协议中唤醒源是输入片选信号的转变。

## 6.25.6 寄存器映射

R: 只读, W: 只写, R/W: 读/写

寄存器	偏移地址	R/W	描述	复位值
<b>USCI 基地址:</b>				
<b>USCI<sub>n</sub>_BA = 0x400D_0000 + (0x1000 * n)</b>				
<b>N= 0, 1</b>				
<b>USCI_CTL</b>	USCI <sub>n</sub> _BA+0x00	R/W	USCI控制寄存器	0x0000_0000
<b>USCI_INTEN</b>	USCI <sub>n</sub> _BA+0x04	R/W	USCI中断使能寄存器	0x0000_0000
<b>USCI_BRGEN</b>	USCI <sub>n</sub> _BA+0x08	R/W	USCI波特率发生器寄存器	0x0000_3C00
<b>USCI_DATINO</b>	USCI <sub>n</sub> _BA+0x10	R/W	USCI 0输入数据信号配置寄存器0	0x0000_0000
<b>USCI_CTLINO</b>	USCI <sub>n</sub> _BA+0x20	R/W	USCI输入控制信号配置寄存器0	0x0000_0000
<b>USCI_CLKIN</b>	USCI <sub>n</sub> _BA+0x28	R/W	USCI输入时钟信号配置寄存器	0x0000_0000
<b>USCI_LINECTL</b>	USCI <sub>n</sub> _BA+0x2C	R/W	USCI线控制寄存器	0x0000_0000
<b>USCI_TXDAT</b>	USCI <sub>n</sub> _BA+0x30	W	USCI发送数据寄存器	0x0000_0000
<b>USCI_RXDAT</b>	USCI <sub>n</sub> _BA+0x34	R	USCI接收数据寄存器	0x0000_0000
<b>USCI_BUFCTL</b>	USCI <sub>n</sub> _BA+0x38	R/W	USCI发送/接收缓冲控制寄存器	0x0000_0000
<b>USCI_BUFSTS</b>	USCI <sub>n</sub> _BA+0x3C	R	USCI发送/接收缓冲状态寄存器	0x0000_0101
<b>USCI_PDMACTL</b>	USCI <sub>n</sub> _BA+0x40	R/W	USCI PDMA控制寄存器	0x0000_0000
<b>USCI_WKCTL</b>	USCI <sub>n</sub> _BA+0x54	R/W	USCI唤醒控制寄存器	0x0000_0000
<b>USCI_WKSTS</b>	USCI <sub>n</sub> _BA+0x58	R/W	USCI唤醒状态寄存器	0x0000_0000
<b>USCI_PROTCTL</b>	USCI <sub>n</sub> _BA+0x5C	R/W	USCI协议控制寄存器	0x0000_0300
<b>USCI_PROTIEN</b>	USCI <sub>n</sub> _BA+0x60	R/W	USCI协议中断使能寄存器	0x0000_0000
<b>USCI_PROTSTS</b>	USCI <sub>n</sub> _BA+0x64	R/W	USCI协议状态寄存器	0x0000_0000

### 6.25.7 寄存器描述

#### USCI\_CTL USCI 控制寄存器

寄存器	偏移地址	R/W	描述	复位值
USCI_CTL	USCIIn_BA+0x00	R/W	USCI 控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved					FUNMODE		

位	描述	
[31:3]	Reserved	保留
[2:0]	FUNMODE	<p><b>功能模式</b></p> <p>该域是选择USCI控制器的协议。待选的协议是不可用的。当两个协议之间切换时，在选择新协议前USCI必须先禁用。在用户写000到FUNMODE同时USCI会被复位。</p> <p>000 = 禁用 USCI，所有协议相关的状态机器被设置为空闲状态。</p> <p>001 = 选择SPI 协议</p> <p>010 = 选择 UART 协议</p> <p>100 = 选择I<sup>2</sup>C 协议</p> <p><b>注:</b>其它值保留</p>

USCI\_INTEN USCI 中断使能寄存器

寄存器	偏移地址	R/W	描述	复位值
USCI_INTEN	USCIIn_BA+0x04	R/W	USCI 中断使能寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved			RXENDIEN	RXSTIEN	TXENDIEN	TXSTIEN	Reserved

位	描述	
[31:5]	<b>Reserved</b>	保留
[4]	<b>RXENDIEN</b>	<p><b>接收结束中断使能位</b>            该位是使能发生接收结束事件时候产生中断            0 = 禁用接收结束中断            1 = 使能接收结束中断</p>
[3]	<b>RXSTIEN</b>	<p><b>接收起始中断使能位</b>            该位是使能发生接收起始事件时候产生中断            0 = 禁用接收起始中断            1 = 使能接收起始中断</p>
[2]	<b>TXENDIEN</b>	<p><b>发送结束中断使能位</b>            该位是使能发生发送结束事件时候产生中断            0 = 禁用发送结束中断            1 = 使能发送结束中断</p>
[1]	<b>TXSTIEN</b>	<p><b>发送起始中断使能位</b>            该位是使能发生发送起始事件时候产生中断            0 = 禁用发送起始中断            1 = 使能发送起始中断</p>
[0]	<b>Reserved</b>	保留

**USCI\_BRGEN USCI 波特率发生器寄存器**

寄存器	偏移地址	R/W	描述	复位值
USCI_BRGEN	USCIIn_BA+0x08	R/W	USCI 波特率发生器寄存器	0x0000_3C00

31	30	29	28	27	26	25	24
Reserved						CLKDIV	
23	22	21	20	19	18	17	16
CLKDIV							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved		TMCNTSRC	TMCNTEN	SPCLKSEL		PTCLKSEL	RCLKSEL

位	描述	
[31:26]	<b>Reserved</b>	保留
[25:16]	<b>CLKDIV</b>	<p><b>时钟分频器</b>          该域是定义协议时钟频率 <math>f_{PROT\_CLK}</math> 和时钟分频器频率 <math>f_{DIV\_CLK}</math> (<math>f_{DIV\_CLK} = f_{PROT\_CLK} / (CLKDIV + 1)</math>)之间的比率  <b>注:</b> 在UART功能下, 当自动波特率功能(ABREN(USCI_PROTCTL[6]))使能时, 会在输入数据0x55的第四个下降沿时被硬件更新。修改值是在位5和位6之间的时间平均值。用户可以使用修改的CLKDIV和新的BRDETTIV (USCI_PROTCTL[24:16])来计算精确的波特率</p>
[15:6]	<b>Reserved</b>	保留
[5]	<b>TMCNTSRC</b>	<p><b>时序测量计数器时钟源选择</b>          0 = 来自 <math>f_{PROT\_CLK}</math>.          1 = 来自 <math>f_{DIV\_CLK}</math>.</p>
[4]	<b>TMCNTEN</b>	<p><b>时序测量计数器使能位</b>          该位是使能10位时序测量计数器          0 = 禁用时序测量计数器          1 = 使能时序测量计数器</p>
[3:2]	<b>SPCLKSEL</b>	<p><b>采样时钟源选择</b>          该域用于对协议采样时钟(<math>f_{SAMP\_CLK}</math>)的时钟源选择          00 = <math>f_{DIV\_CLK}</math>.          01 = <math>f_{PROT\_CLK}</math>.          10 = <math>f_{SCLK}</math>.          11 = <math>f_{REF\_CLK}</math>.</p>
[1]	<b>PTCLKSEL</b>	<p><b>协议协议时钟源选择</b>          该位用于选择协议时钟(<math>f_{PROT\_CLK}</math>)的源信号          0 = 参考时钟 <math>f_{REF\_CLK}</math>.          1 = <math>f_{REF\_CLK2}</math> (频率是 <math>f_{REF\_CLK}</math> 的一半)</p>

[0]	RCLKSEL	<b>参考时钟源选择</b> 该位是选择参考时钟( $f_{REF\_CLK}$ )源信号 0 = 外设设备时钟 $f_{PCLK}$ . 1 = 保留
-----	---------	---

**USCI\_DATINO USCI输入数据信号配置寄存器0**

寄存器	偏移地址	R/W	描述	复位值
USCI_DATINO	USCIIn_BA+0x10	R/W	USCI输入数据信号配置寄存器0	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved					ININV	Reserved	SYNCSEL

位	描述	
[31:3]	Reserved	保留
[2]	ININV	<p><b>输入信号反相选择</b>            该域是使能对输入异步信号的反相            0 = 对异步输入信号不反相            1 = 对异步输入信号反相  <b>注:</b> 在 SPI 协议下, 我们建议该位设置为0.</p>
[1]	Reserved	保留
[0]	SYNCSEL	<p><b>输入信号同步选择</b>            该位是选择异步输入(带反相的)信号或是同步(带过滤的)信号被用作数据移位单元的输入            0 = 异步信号作为信号被用作数据移位单元的输入            1 = 同步信号作为信号被用作数据移位单元的输入  <b>注:</b> 在 SPI 协议下, 我们建议该位设置为0.</p>

**USCI\_CTLIN0 USCI输入控制信号配置寄存器0**

寄存器	偏移地址	R/W	描述	复位值
USCI_CTLIN0	USCIIn_BA+0x20	R/W	USCI输入控制信号配置寄存器0	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved					ININV	Reserved	SYNCSEL

位	描述	
[31:3]	<b>Reserved</b>	保留
[2]	<b>ININV</b>	<b>输入信号反相选择</b> 该域是定义使能对输入异步信号输入反相 0 = 对异步输入信号不反相 1 = 对异步输入信号反相
[1]	<b>Reserved</b>	保留
[0]	<b>SYNCSEL</b>	<b>输入同步信号选择</b> 该位是选择异步输入（带反相的）信号或是同步（带过滤的）信号被用作数据移位单元的输入 0 = 异步信号作为信号被用作数据移位单元的输入 1 = 同步信号作为信号被用作数据移位单元的输入 <b>注:</b> 在 SPI 协议下, 我们建议该位设置为0.

**USCI\_CLKIN USCI输入时钟信号配置寄存器**

寄存器	偏移地址	R/W	描述	复位值
USCI_CLKIN	USCIIn_BA+0x28	R/W	USCI 输入时钟信号配置寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved							SYNCSEL

位	描述	
[31:1]	Reserved	保留
[0]	SYNCSEL	<p><b>输入同步信号选择</b></p> <p>该位是选择异步输入（带反相的）信号或是同步（带过滤的）信号被用作数据移位单元的输入</p> <p>0 = 异步信号作为信号被用作数据移位单元的输入</p> <p>1 = 同步信号作为信号被用作数据移位单元的输入</p> <p>注：在 SPI 协议下，我们建议该位设置为 0。</p>

## USCI\_LINECTL USCI线控制寄存器

寄存器	偏移地址	R/W	描述	复位值
USCI_LINECTL	USCIIn_BA+0x2C	R/W	USCI 线控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved				DWIDTH			
7	6	5	4	3	2	1	0
CTLOINV	Reserved	DATOINV	Reserved				LSB

位	描述	
[31:12]	Reserved	保留
[11:8]	DWIDTH	<p>传输的字长</p> <p>该域定义发送和接收的数据字长。在数据缓冲中数据总是右对齐。USCI支持4到16位的字长</p> <p>0x0:在数据字中包含16位，位于[15:0]这些位中</p> <p>0x1: 保留</p> <p>0x2: 保留</p> <p>0x3: 保留</p> <p>0x4:在数据字中包含4位，位于[3:0]这些位中</p> <p>0x5:在数据字中包含5位，位于[4:0]这些位中</p> <p>...</p> <p>0xF:在数据字中包含15位，位于[14:0]这些位中</p>
[7]	CTLOINV	<p>控制信号输出反相选择</p> <p>该位定义了内部控制信号和输出控制信号之间的关系</p> <p>0 = 无影响</p> <p>1 =在输出前控制信号被反相</p> <p>注:在不同协议中控制信号有不同的定义。SPI协议中，控制信号是从机片选信号。</p>
[6]	Reserved	保留
[5]	DATOINV	<p>数据输出反相选择</p> <p>该位定义了内部移位数据值和USCIx_DAT0/1引脚输出数据信号之间的关系</p> <p>0 = 数据输出电不反相</p> <p>1 =数据输出电反相</p>
[4:1]	Reserved	保留

[0]	LSB	LSB优先传输选择 0 =取决于DWIDTH设置的接收/发送数据缓冲的MSB被先发送/接收 1 =数据缓冲的位0及LSB被先发送/接收
-----	-----	---

USCI\_TXDAT USCI 发送数据寄存器

寄存器	偏移地址	R/W	描述	复位值
USCI_TXDAT	USCIIn_BA+0x30	W	USCI 发送数据寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
TXDAT							
7	6	5	4	3	2	1	0
TXDAT							

位	描述	
[31:17]	<b>Reserved</b>	保留
[16]	<b>PORTDIR</b>	<p><b>端口方向控制</b>            该位域仅在USCI工作在SPI的半双工传输下有效。它被用来定义数据端口引脚的方向。当软件写USCI_TXDAT寄存器时，发送数据和端口方向同时被处理。            0 =数据引脚被配置成输出模式            1 =数据引脚被配置成输入模式</p>
[15:0]	<b>TXDAT</b>	<p><b>发送数据</b>            软件可以写16位发送数据到该域用来发送为了避免发送数据写溢出，用户必须在写发送数据到该域之前检查TXEMPTY (USCI_BUFSTS[8])状态。</p>

USCI\_RXDAT USCI 接收数据寄存器

寄存器	偏移地址	R/W	描述	复位值
USCI_RXDAT	USCIIn_BA+0x34	R	USCI 接收数据寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
RXDAT							
7	6	5	4	3	2	1	0
RXDAT							

位	描述	
[31:16]	Reserved	保留
[15:0]	RXDAT	接收的数据 该域值监控存储在接收数据缓冲的接收到的数据

USCI\_BUFCTL USCI 发送/接收缓冲控制寄存器

寄存器	偏移地址	R/W	描述	复位值
USCI_BUFCTL	USCIIn_BA+0x38	R/W	USCI 发送/接收缓冲控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved						RXRST	TXRST
15	14	13	12	11	10	9	8
RXCLR	RXOVIEN	Reserved					
7	6	5	4	3	2	1	0
TXCLR	TXUDRIEN	Reserved					

位	描述
[31:18]	<b>Reserved</b> 保留
[17]	<b>RXRST</b> 接收复位 0 = 无影响 1 = 复位接收相关计数器、状态机、接收移位寄存器的内容和数据缓冲。 <b>注:</b> 1个PCLK周期后自动被清除
[16]	<b>TXRST</b> 发送复位 0 = 无影响 1 = 复位发送相关计数器、状态机、发送移位寄存器的内容和数据缓冲。 <b>注1:</b> 1个PCLK周期后自动被清除 <b>注2:</b> 如果 USCI_BUFCTL[5]=0, 写1到该位将会拉低输出数据引脚
[15]	<b>RXCLR</b> 清接收缓冲 0 = 无影响 1 = 接收缓冲被清除。仅当缓冲没有参与数据通信时被使用。 <b>注:</b> 1个PCLK周期后自动被清除
[14]	<b>RXOVIEN</b> 接收缓冲溢出中断使能控制 0 = 禁用接收上溢出中断 1 = 使能接收上溢出中断
[13:8]	<b>Reserved</b> 保留
[7]	<b>TXCLR</b> 清发送缓冲 0 = 无影响 1 = 发送缓冲被清除。仅当缓冲没有参与数据通信时被使用。 <b>注:</b> 1个PCLK周期后自动被清除

[6]	<b>TXUDRIEN</b>	从机发送下溢出中断使能位 0 =禁用发送下溢出中断 1 =使能发送下溢出中断
[5:0]	<b>Reserved</b>	保留

USCI\_BUFSTS USCI 发送/接收缓冲状态寄存器

寄存器	偏移地址	R/W	描述	复位值
USCI_BUFSTS	USCIIn_BA+0x3C	R	USCI 发送/接收缓冲状态寄存器	0x0000_0101

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved				TXUDRIF	Reserved	TXFULL	TXEMPTY
7	6	5	4	3	2	1	0
Reserved				RXOVIF	Reserved	RXFULL	RXEMPTY

位	描述
[31:12]	<b>Reserved</b> 保留
[11]	<b>TXUDRIF</b> <b>发送缓冲下溢出中断状态</b> 该位表示发送缓冲下溢出事件被侦测到。如果TXUDRIEN (USCI_BUFCTL[6])使能，会激活相应的中断请求。软件写1清除该位。 0 = 发送缓冲下溢出事件未被侦测到 1 = 发送缓冲下溢出事件被侦测到
[10]	<b>Reserved</b> 保留
[9]	<b>TXFULL</b> <b>发送缓冲满指示</b> 0 = 发送缓冲还没满 1 = 发送缓冲满
[8]	<b>TXEMPTY</b> <b>发送缓冲空指示</b> 0 = 发送缓冲非空 1 = 发送缓冲为空，可进行下次数据发送
[7:4]	<b>Reserved</b> 保留
[3]	<b>RXOVIF</b> <b>接收缓冲上溢出中断状态</b> 该位表示接收缓冲上溢出事件被侦测到。如果RXOVIEN (USCI_BUFCTL[14])使能，会激活相应的中断请求。软件写1清除该位。 0 = 接收缓冲上溢出事件未被侦测到 1 = 接收缓冲上溢出事件被侦测到
[2]	<b>Reserved</b> 保留
[1]	<b>RXFULL</b> <b>接收缓冲满指示</b> 0 = 接收缓冲还没满 1 = 接收缓冲满

[0]	<b>RXEMPTY</b>	接收缓冲空指示 0 =接收缓冲非空 1 =接收缓冲为空
-----	----------------	-----------------------------------

USCI\_PDMACTL USCI PDMA控制寄存器

寄存器	偏移地址	R/W	描述	复位值
USCI_PDMACTL	USCIIn_BA+0x40	R/W	USCI PDMA控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved				PDMAEN	RXPDMAEN	TXPDMAEN	PDMARST

位	描述	
[31:4]	Reserved	保留
[3]	PDMAEN	<b>PDMA模式使能位</b> 0 = 禁用PDMA功能 1 = 使能PDMA功能 注: I²C 不支持 PDMA 功能
[2]	RXPDMAEN	<b>PDMA接收通道有效</b> 0 = 禁用PDMA接收功能 1 = 使能PDMA接收功能
[1]	TXPDMAEN	<b>PDMA发送通道有效</b> 0 = 禁用PDMA发送功能 1 = 使能PDMA发送功能
[0]	PDMARST	<b>PDMA 复位</b> 0 = 无影响 1 = 复位USCI的PDMA控制逻辑。该位会自动地被清0

USCI\_WKCTL USCI 唤醒控制寄存器

寄存器	偏移地址	R/W	描述	复位值
USCI_WKCTL	USCIIn_BA+0x54	R/W	USCI 唤醒控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved					PDBOPT	WKADDREN	WKEN

位	描述	
[31:3]	<b>Reserved</b>	保留
[2]	<b>PDBOPT</b>	<b>掉电阻塞选择</b> 0 = 如果用户试图在协议正在传输的时候执行WFI来进入掉电模式，MCU会停止传输并且立即进入掉电模式 1 = 如果用户试图在协议正在传输的时候执行WFI来进入掉电模式，传输会继续进行并且MCU立即进入掉电模式
[1]	<b>WKADDREN</b>	<b>唤醒地址匹配使能位</b> 0 = 数据唤醒芯片 1 = 地址匹配唤醒芯片
[0]	<b>WKEN</b>	<b>唤醒使能位</b> 0 = 禁用唤醒功能 1 = 使能唤醒功能

**USCI\_WKSTS USCI 唤醒状态寄存器**

寄存器	偏移地址	R/W	描述	复位值
USCI_WKSTS	USCIIn_BA+0x58	R/W	USCI 唤醒状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved							WKF

位	描述	
[31:1]	Reserved	保留
[0]	WKF	唤醒标志 当芯片被从掉电模式唤醒，该位置1.软件写1清除该位。

USCI PROTCTL USCI 协议控制寄存器-SPI

寄存器	偏移地址	R/W	描述	复位值
USCI_PROTCTL	USCIIn_BA+0x5C	R/W	USCI 协议控制寄存器	0x0000_0300

31	30	29	28	27	26	25	24
PROTEN	Reserved		TXUDRPOL	Reserved		SLVTOCNT	
23	22	21	20	19	18	17	16
SLVTOCNT							
15	14	13	12	11	10	9	8
Reserved	TSMSEL			SUSPITV			
7	6	5	4	3	2	1	0
SCLKMODE		Reserved		AUTOSS	SS	SLV3WIRE	SLAVE

位	描述	
[31]	PROTEN	<b>SPI协议使能位</b> 0 = 禁用SPI协议 1 = 使能SPI协议
[30:29]	Reserved	保留
[28]	TXUDRPOL	<b>发送下溢出数据极性(仅从机模式)</b> 该位定义在没有数据可发送时发送数据电平 0 = 如果TX下溢出事件发生, 输出数据电平为0. 1 = 如果TX下溢出事件发生, 输出数据电平为1.
[27:26]	Reserved	保留
[25:16]	SLVTOCNT	<b>从机模式超时周期(仅从机模式)</b> 在从机模式, 该域被用作从机超时周期。该域表示在输入SCLK的两个边沿之间有多少时钟周期 (时钟由TMCNTSRC, USCI_BRGEN[5]选择) 产生从机超时事件。写0x0到该位域禁用从机超时功能。 范例: SLVTOCNT为 0x0A 和 TMCNTSRC (USCI_BRGEN[5])为1 1, 表示如果SPI总线时钟引脚超过 (10+1) 个f <sub>DIV_CLK</sub> 周期没有改变, 将会有超时事件发生。
[15]	Reserved	保留
[14:12]	TSMSEL	<b>传输数据模式选择</b> 该位域定义发送和接收数据是如何移进和移出的。 TSMSEL = 000b: 全双工 SPI. TSMSEL = 100b: 半双工 SPI. 其它值保留 <b>注:</b> 改变该位域的值将会自动地产生TXRST 和 RXRST来清除TX/RX数据缓冲。
[11:8]	SUSPITV	<b>暂停间隔(仅主模式)</b> 该四位用来配置在一次数据传输过程中连续两个发送/接收事务之间的暂停间隔。暂停间隔是从当前事务字的最后一个时钟边沿到接下来的事务的第一个边沿时钟。 默认值是 0x3. 暂

		<p>停间隔的周期可以根据下面公式获得:</p> $(\text{SUSPITV [3:0]} + 0.5) * \text{SPI\_CLK} \text{时钟周期}$ <p>例:</p> <p>SUSPITV = 0x0 ... 0.5 SPI_CLK 时钟周期</p> <p>SUSPITV = 0x1 ... 1.5 SPI_CLK 时钟周期</p> <p>.....</p> <p>SUSPITV = 0xE ... 14.5 SPI_CLK 时钟周期</p> <p>SUSPITV = 0xF ... 15.5 SPI_CLK 时钟周期</p>
[7:6]	<b>SCLKMODE</b>	<p><b>串行总线时钟模式</b></p> <p>该位域定义SCLK空闲状态、数据传输和接收边沿</p> <p>MODE0 =空闲状态时是低电平。下降沿时发送数据，上升沿时接收数据。</p> <p>MODE1 =空闲状态时是低电平。上升沿时发送数据，下降沿时接收数据。</p> <p>MODE2 =空闲状态时是高电平。上升沿时发送数据，下降沿时接收数据。</p> <p>MODE3 =空闲状态时是高电平。下降沿时发送数据，上升沿时接收数据。</p>
[5:4]	<b>Reserved</b>	保留
[3]	<b>AUTOSS</b>	<p><b>自动从机片选功能使能(仅主模式)</b></p> <p>0 =通过设置SS (USCI_PROTCTL[2])值来控制从机片选信号。</p> <p>1 = 从机片选信号自动地产生。当发送/接收启动时从机片选信号会被SPI控制激活，在每次发送/接收结束之后片选信号取消</p>
[2]	<b>SS</b>	<p><b>从机选择控制(仅主模式)</b></p> <p>如果AUTOSS位被清除，设置该位为1将激活从机片选信号，设置设置该位为0将取消从机片选信号。</p> <p>如果AUTOSS功能使能(AUTOSS = 1),该位的设置值不会影响从机片选信号的当前的状态。</p> <p><b>注:</b> SPI 协议下，内部从机片选信号激活状态时为高电平。</p>
[1]	<b>SLV3WIRE</b>	<p><b>从机3-线模式选择(仅从机模式)</b></p> <p>从机模式下，SPI协议在3-线接口（及没有片选信号）下工作。</p> <p>0 = 4-线接口</p> <p>1 = 3-线接口</p>
[0]	<b>SLAVE</b>	<p><b>从机模式选择</b></p> <p>0 = 主机模式</p> <p>1 = 从机模式</p>

USCI\_PROTIEN USCI 协议中断使能寄存器-SPI

寄存器	偏移地址	R/W	描述	复位值
USCI_PROTIEN	USCIIn_BA+0x60	R/W	USCI 协议中断使能寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved				SLVBEIEN	SLVTOIEN	SSACTIEN	SSINAIEN

位	描述	
[31:5]	<b>Reserved</b>	保留
[3]	<b>SLVBEIEN</b>	<p><b>从机模式位计数错误中断使能控制</b>            如果在从机模式下数据传输由从机超时或是从机片选取消事件终止，这样以来发送/接收的数据位计数就会与DWIDTH (USCI_LINECTL[11:8])设置值不匹配。位计数错误事件发生。            0 =禁用从机模式位计数错误中断            1 =使能从机模式位计数错误中断</p>
[2]	<b>SLVTOIEN</b>	<p><b>从机超时中断使能控制</b>            SPI协议下，使能该中断以防从机超时事件            0 =禁用从机超时中断            1 =使能从机超时中断</p>
[1]	<b>SSACTIEN</b>	<p><b>从机片选激活中断使能控制</b>            如果从机片选信号变为有效状态，该位是使能/禁用产生从机片选激活中断            0 =禁用从机片选激活中断            1 =使能从机片选激活中断</p>
[0]	<b>SSINAIEN</b>	<p><b>从机片选取消中断使能控制</b>            如果从机片选信号变为无效状态，该位是使能/禁用产生从机片选取消中断            0 =禁用从机片选取消中断            1 =使能从机片选取消中断</p>

USCI\_PROTSTS USCI 协议状态寄存器-SPI

寄存器	偏移地址	R/W	描述	复位值
<b>USCI_PROTSTS</b>	USCIIn_BA+0x64	R/W	USCI 协议状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved						SLVUDR	BUSY
15	14	13	12	11	10	9	8
Reserved						SSACTIF	SSINAIF
7	6	5	4	3	2	1	0
Reserved	SLVBEIF	SLVTOIF	RXENDIF	RXSTIF	TXENDIF	TXSTIF	Reserved

位	描述
[31:19]	<b>Reserved</b> 保留
[18]	<b>SLVUDR</b> 从机模式发送下溢出状态（只读） 在从机模式下，如果没有可用的发送数据在缓冲中同时输入串行总线时钟使发送数据继续移出，这时该位就会被置1。该位置位与字传输当前移出数据是否已切换到TXUDRPOL (USCI_PROTCTL[28])没有关系。 0 =没有发生从机发送下溢出事件 1 =发生从机发送下溢出事件
[17]	<b>BUSY</b> 忙状态（只读） 0 = SPI 处在空闲状态 1 = SPI 处在忙状态 下面列出了总线忙的状况： a. USCI_PROTCTL[31] = 1 和 TXEMPTY = 0. b. 对于SPI主机模式， TXEMPTY = 1但是当前传输还没完成 c. 对于SPI从机模式， USCI_PROTCTL[31] = 1 并且在从机片选激活时有串行时钟输入到SPI内核 逻辑单元 d. 对于SPI从机模式， USCI_PROTCTL[31] = 1 并且即使在从机片选没有激活状况下发送缓冲或是发送移位寄存器为非空
[16]	<b>SSLINe</b> 从机选择线总线状态（只读） 仅在从模式下有效。该位是用来监控总线上输入从机片选信号当前的状态。 0 =从机选择线状态为0 1 =从机选择线状态为1
[15:10]	<b>Reserved</b> 保留
[9]	<b>SSACTIF</b> 从机选择激活中断标志(仅从机模式) 该位表示内部从机片选信号变为激活状态。软件写1清除该位。 0 =从机片选信号没有改变成激活状态

		1 =从机片选信号改变成激活状态  注: 内部从机片选信号激活状态时为高电平。
[8]	<b>SSINAIF</b>	从机选择取消中断标志(仅从机模式) 该位表示内部从机片选信号变为非激活状态。软件写1清除该位。 0 =从机片选信号没有改变成非激活状态 1 =从机片选信号改变成非激活状态  注: 内部从机片选信号激活状态时为高电平。
[7]	<b>Reserved</b>	保留
[6]	<b>SLVBEIF</b>	从机位计数错误中断标志(仅从机模式) 0 =没有发生从机位计数错误事件 1 =发生从机位计数错误事件  注:写1清除该位
[5]	<b>SLVTOIF</b>	从机超时中断标志(仅从机模式) 0 =没有发生从机超时事件 1 =发生从机超时事件  注:写1清除该位
[4]	<b>RXENDIF</b>	接收结束中断标志 0 =没有接收结束中断状态发生 1 =有接收结束中断状态发生  注:写1清除该位
[3]	<b>RXSTIF</b>	接收起始中断标志 0 =没有接收起始中断状态发生 1 =有接收起始中断状态发生  注:写1清除该位
[2]	<b>TXENDIF</b>	发送结束中断标志 0 =没有发送结束中断状态发生 1 =有发送结束中断状态发生  注:写1清除该位
[1]	<b>TXSTIF</b>	发送起始中断标志 0 =没有发送起始中断状态发生 1 =有发送起始中断状态发生  注:写1清除该位
[0]	<b>Reserved</b>	保留

## 6.26 USCI - I<sup>2</sup>C 模式

### 6.26.1 概述

在I<sup>2</sup>C总线上，数据通过时钟线SCL和数据线SDA在主从机间逐一字节同步传送。每个字节数据长度是8位。一个SCL时钟脉冲传输一个数据位，数据由最高位MSB开始传输，每个字节传输后跟随一个应答位，每个位在SCL为高时采样；因此，SDA线可能在SCL为低时改变，但在SCL为高时必须保持稳定。当SCL为高时，SDA线上的跳变视为一个命令(START或STOP)。更多关于I<sup>2</sup>C总线时序的细节请参考下图 6.26-1。

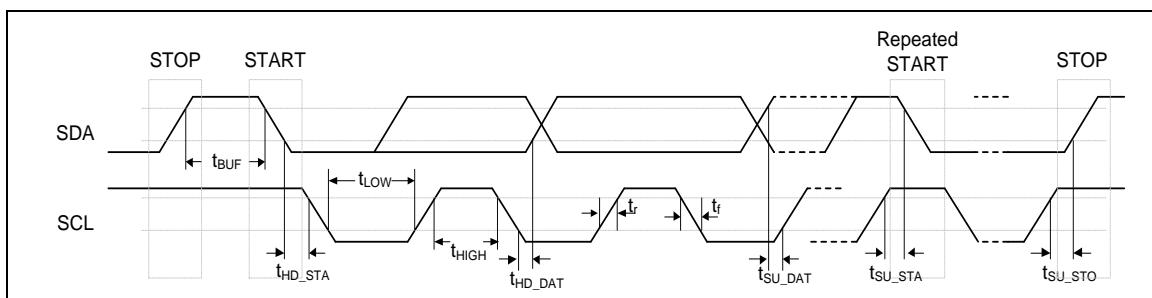


图 6.26-1 I<sup>2</sup>C 总线时序

片上I<sup>2</sup>C外设提供了一个符合I<sup>2</sup>C总线规范的串行接口。I<sup>2</sup>C端口自动处理字节传输。通过设置寄存FUNMODE (USCI\_CTL [2:0]) = 100B选择I<sup>2</sup>C模式。USCI硬件接口通过数据线SDA 和时钟线SCL两个管脚连到I<sup>2</sup>C总线。当I/O管脚作为I<sup>2</sup>C端口使用时，用户必须事先设定I/O管脚为I<sup>2</sup>C功能。

**注:** SDA 和 SCL两个管脚需要上拉电阻，因为这两个管脚在USCI被选作I<sup>2</sup>C模式下为开漏脚。

### 6.26.2 特性

- 支持主机/从机模式
- 支持7位地址模式和10位地址模式
- 支持标准模式 (100 kbps), 或 快速模式 (400 kbps)
- 支持多主机总线
- 支持一个发送缓冲和两个接收缓冲
- 支持10位总线超时
- 支持总线监控模式
- 支持掉电状况下由数据或是地址匹配唤醒
- 支持建立/保持时间可编程
- 支持多地址识别 (2组从机地址带mask选项)

### 6.26.3 框图

USCI的基本配置如下：

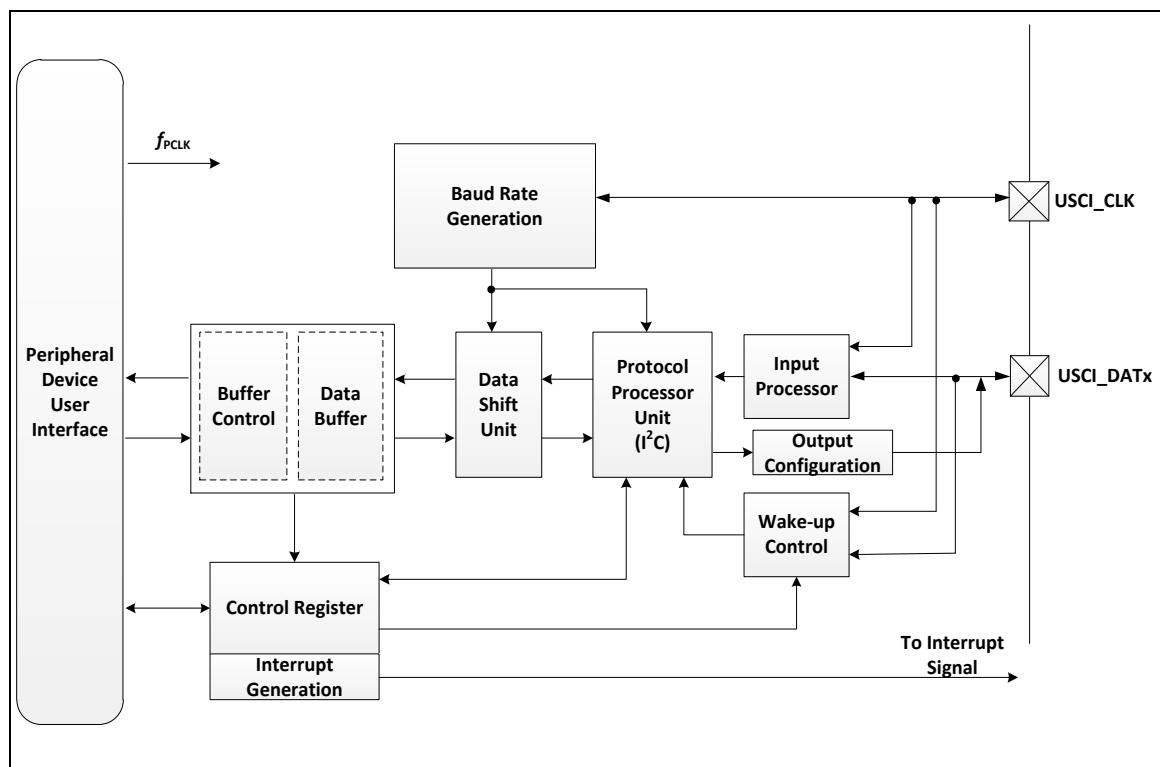


图 6.26-2 USCI I2C模式框图

#### 6.26.4 基本配置

##### 6.26.4.1 USCI0 I2C基本配置

- 时钟源配置
  - 使能USCI0时钟的控制位是CLK\_APBCLK1[8]中的USCI0CKEN位
  - 设置USCI\_CTL[2:0]=3'b100, 使能USCI0\_I2C功能
- 复位配置
  - 复位USCI0的控制位是SYS\_IPRST2[8]中的USCI0RST位
- 引脚配置

组	引脚名	GPIO	MFP
USCI0	USCI0_CLK	PD.0	MFP3
		PB.12	MFP5
		PA.11	MFP6
		PE.2	MFP7
	USCI0_DAT0	PD.1	MFP3
		PB.13	MFP5
		PA.10	MFP6

		PE.3	MFP7
--	--	------	------

#### 6.26.4.2 USCI1 I2C基本配置

- 时钟源配置
  - 使能USCI1时钟的控制位是CLK\_APBCLK1[9]中的USCI1CKEN位
  - 设置USCI\_CTL[2:0]=3'b100，使能USCI1\_I2C功能
- 复位配置
  - 复位USCI1的控制位是SYS\_IPRST2[9]中的USCI1RST位
- 引脚配置

组	引脚名	GPIO	MFP
USCI1	USCI1_CLK	PB.8	MFP4
		PD.7, PE.12	MFP6
		PB.1	MFP8
	USCI1_DAT0	PB.7	MFP4
		PD.5, PE.10	MFP6
		PB.2	MFP8

#### 6.26.5 功能描述

标准I<sup>2</sup>C 协议如下图 6.26-3，通常标准通讯有以下4部分：

- (START) 或者重复起始信号(Repeated START)
- 从机地址传输和R/W 位传输
- 数据传输
- 停止信号(STOP)

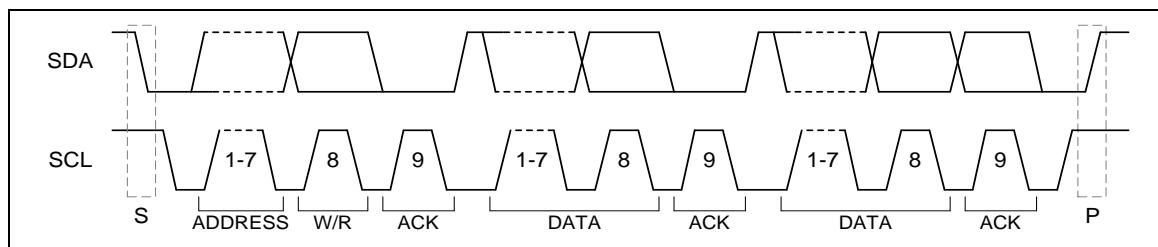


图 6.26-3 I<sup>2</sup>C 协议

##### 6.26.5.1 起始或重复起始信号

当总线处于释放/空闲状态下，说明没有主机设备占用总线（SDA 与SCL线同时为高），主机可以通过发送起始(START)信号来发起传输过程。起始信号，通常表示为"S"位，当SCL 线为高时，

SDA 线上信号由高至低变化，就被定义为起始信号。起始信号表示一个新的数据传输的开始。

主机发送完地址字节（地址和读/写位）后，可以发送任何数量的数据并带一个停止信号。也可以用另一个起始信号替代停止信号，随后是地址(包含读写位)和更多的数据。这个起始信号叫做重复起始。这个定义可以用来发送任意个起始信号，它的目的是在不释放总线情况下，对一个或多个设备能读/写操作，而不让操作被打断。用这个方法主机跟其他从机或同一个从机在不同方向传输（例如，写设备到读设备）不用释放总线。

#### 6.26.5.2 停止信号

主机可以通过产生一个停止信号来终止数据通信。停止信号，通常表示为P位，当SCL 线为高时，SDA线上信号由低至高变化，就被定义为停止信号。

下图 6.26-4 为起始，重复起始和停止信号的波形。

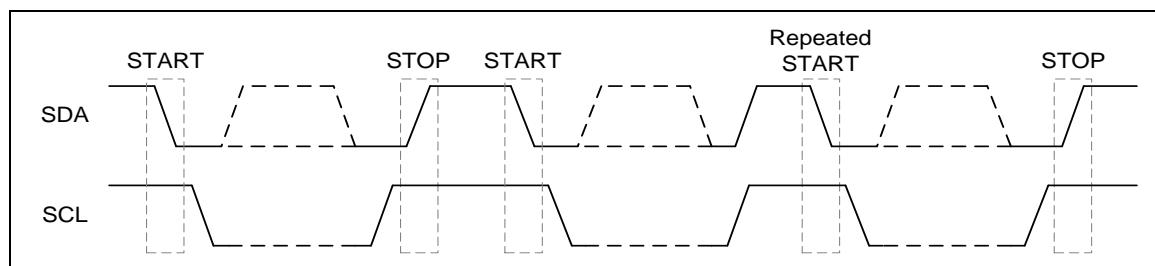


图 6.26-4 起始和停止状况

#### 6.26.5.3 从地址传输

在一个起始（或重复起始）信号后，主机发送一个从机地址来确认通信的目标设备。起始地址有一个或两个字节组成（针对7位或10位地址情况）。地址字节过后，从机会对传输的地址作应答反应。

因此，从机地址可以被编程并且与接收到的地址比较。一旦匹配，从机会回一个应答(SDA = 0).如果与目标不匹配则无应答(SDA = 1).除了编程地址匹配之外，如果从机有能力处理类似的请求，那么另外的地址值也必须有应答回应。地址为00H是广播地址也能应答。

为了可选择应答不同的地址值，控制机制是按如下实现的：

- 如果GCFUNC 位 (USCI\_PROTCTL [0])被置位，I<sup>2</sup>C端口硬件就会响应广播地址 (00H)。清除GC位禁用广播功能
- I<sup>2</sup>C端口装备了一个设备地址寄存器USCI\_DEVADDRn ( $n = 0\sim 1$ )。在7位地址模式下，首先接收到的7位地址会与编程的从机地址(USCI\_DEVADDRn [6:0])相比较。如果匹配，从机会回应答。
- 对于10位地址模式，当ADDR10EN (USCI\_PROTCTL [4])被置位时，如果从机地址是1111 0XXB，XX位用于和USCI\_DEVADDR [9:8]比较检测到地址匹配时将会产生应答。从机等待第二个地址字节然后与USCI\_DEVADDR [7:0]比较，依据收到的10位地址来决定是否产生应答。用户必须处理好保留的的地址（详细描述请参考I<sup>2</sup>C规格书）。仅支持地址1111 0XXB。在这

些条件下，当地址匹配时，SLASEL (USCI\_PROTSTS [14]) 会被置位。SLASEL (USCI\_PROTSTS [14]) 会被一个（重复）起始或停止条件自动的清除。

- I<sup>2</sup>C端口有两组地址掩码寄存器I2C\_ADDRMSKn (n = 0~1)，这样就能支持多地址识别。当地址掩码寄存器中位设置为1的时候，表示接收相应的地址为不需要关心，如果设置为0，表示接收相应地址位必须和地址寄存器是一样的。

#### 6.26.5.4 数据传输

当从机设备地址和R/W位被成功识别到，就可以根据R/W位所决定的方向按一字节一字节方式进行数据传输。每个字节传输完后，紧接着的第9个SCL时钟周期会有一个应答信号位。如果从机上产生无应答信号(NACK)，主机可以产生停止信号来中止数据传输或者产生重复起始信号开始新一轮数据传输。

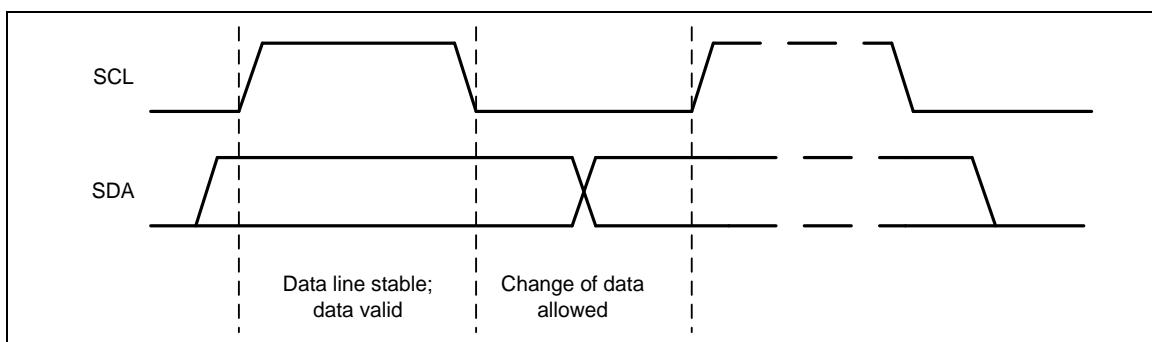


图 6.26-5 I<sup>2</sup>C 总线上的位传输

当主机作为接收设备时，发生无应答信号(NACK)，则从机将释放SDA 线，让主机产生停止信号或重复起始信号。

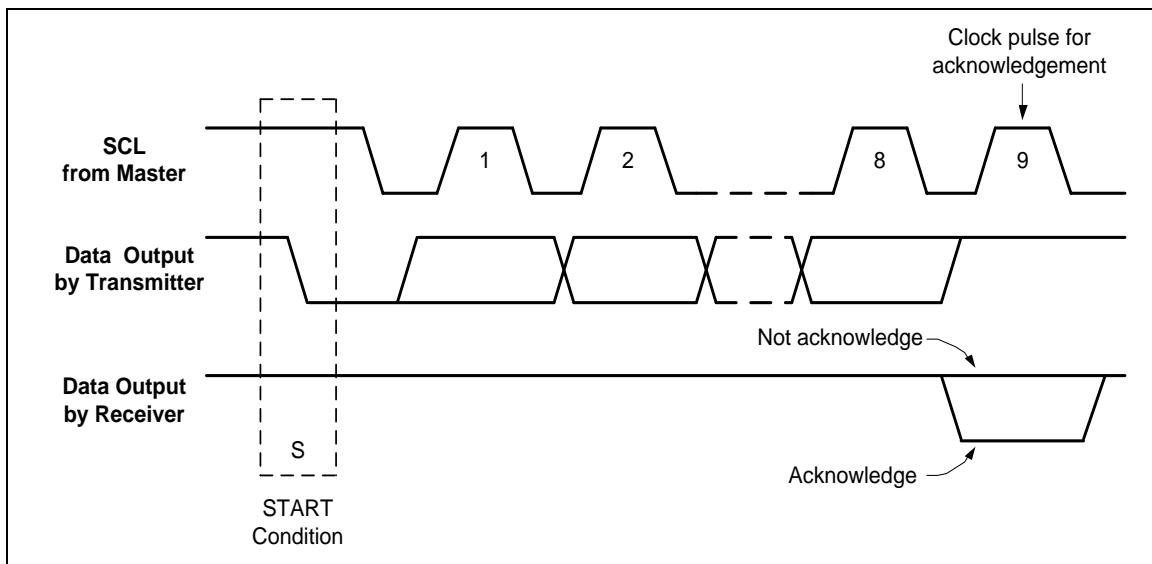


图 6.26-6 总线上的应答信号

#### 6.26.5.5 时钟波特率位

当I<sup>2</sup>C在主机模式下时，I<sup>2</sup>C的波特率由USCI\_BRGEN寄存器决定，在从机模式下不用管。从机模式

下, I<sup>2</sup>C会自动识别来自主机的时钟的任何频率。在RCLKSEL, SPCLKSEL, PDSCNT和DSCNT中定义波特率:

- RCLKSEL (USCI\_BRGEN [0])

    定义输入频率fREF\_CLK

- SPCLKSEL (USCI\_BRGEN[3:2])

    定义采样时钟fSAM\_CLK的时钟源

- PDSCNT (USCI\_BRGEN [9:8])

    定义数据采样时间的长度(由 fREF\_CLK 经 1, 2, 3, 或 4分频)

- DSCNT (USCI\_BRGEN [14:10])

    定义采样时间每位时间的数目

标准设定是由RCLKSEL = 0 (fREF\_CLK = f<sub>PCLK</sub>), PTCLKSEL = 0 (f<sub>PROT\_CLK</sub> = f<sub>REF\_CLK</sub>) 和SPCLKSEL = 2'b00 (fSAMP\_CLK = f<sub>DIV\_CLK</sub>)决定。在这些条件下, 波特率值如下:

$$f_{I2C} = f_{REF\_CLK} \times \frac{1}{CLKDIV + 1} \times \frac{1}{PDSCNT + 1} \times \frac{1}{DSCNT + 1}$$

为了产生更慢的频率, 设置PTCLKSEL = 1 (f<sub>PROT\_CLK</sub> = f<sub>REF\_CLK</sub>2)选择附加除2阶段, 如下:

$$f_{I2C} = \frac{f_{REF\_CLK}}{2} \times \frac{1}{CLKDIV + 1} \times \frac{1}{PDSCNT + 1} \times \frac{1}{DSCNT + 1}$$

如果SPCLKSEL = 2'b10 (fSAMP\_CLK = f<sub>SCLK</sub>), 和 RCLKSEL = 0 (fREF\_CLK = f<sub>PCLK</sub>), PTCLKSEL = 0 (f<sub>PROT\_CLK</sub> = f<sub>REF\_CLK</sub>), 波特率如下:

$$f_{I2C} = f_{REF\_CLK} \times \frac{1}{CLKDIV + 1} \times \frac{1}{2} \times \frac{1}{PDSCNT + 1} \times \frac{1}{DSCNT + 1}$$

#### 6.26.5.6 字节错位

如果设备被选用为主机/从机发送模式, 在需要发送数据的时候发送缓冲TXDAT没有有效的数据被传输, 设备就会在ACK位前拉低SCL到0。在软件写1到PTRG (USCI\_PROTCTL [5])中后等待周期结束。

#### 6.26.5.7 多主机仲裁

在一些应用中, 一个I<sup>2</sup>C总线上有多个主机同时访问从机, 并有可能同时在传送数据。I<sup>2</sup>C是支持多主机模式, 并包含有冲突检测和仲裁, 防止数据损坏。

如果两个主机同时发命令, 通过仲裁来决定哪个优先并继续发命令。仲裁是在SCL为高时在SDA上执行的。每一个主机都会检测总线上的SDA信号是否符合它产生的SDA信号。如果检测到总线上SDA为低, 但应该为高, 则这个主机将失去仲裁。设备在仲裁丢失后会产生SCL脉冲直到本字节结束, 然后释放总线进入从机模式。仲裁可以一直进行到所有数据被传输完。这样意味着多主机系统中主机必须监控总线和做相应的处理

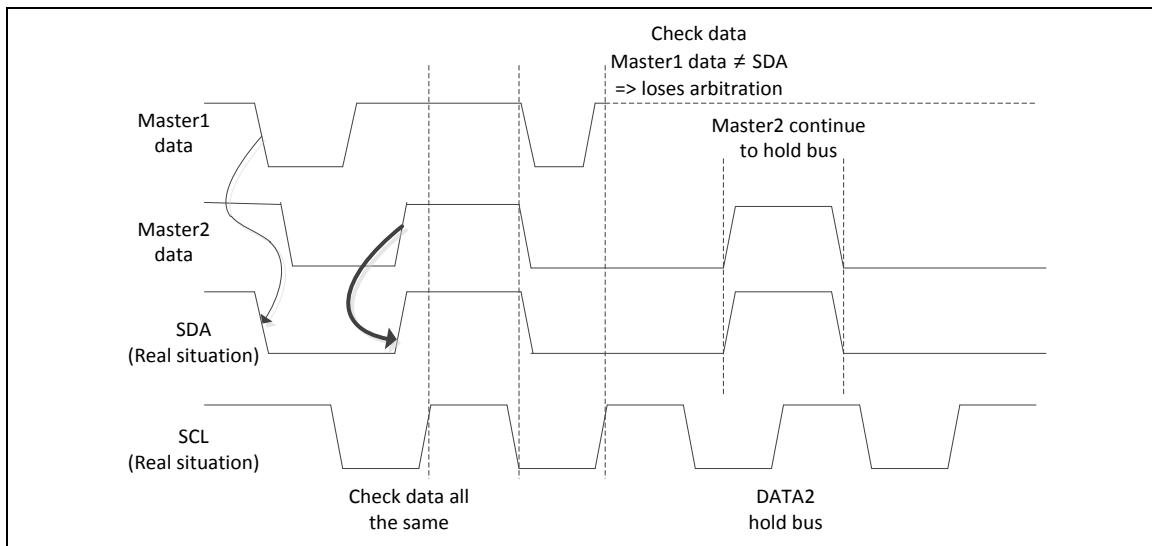


图 6.26-7 仲裁丢失

在地址和数据发送期间，主机发送器会对每个数据位在SCL的上升沿进行检查，如果发送的数据等于在SDA线上读到的数据，主机则继续保持总线，如果不等（发送值=1但是读的值是0），主机失去发送权利。该事件可由ARBLOIF (USCI\_PROTSTS [11])中断标志指示，如果使能ARBLOIEN (USCI\_PROTIEN [4])则会产生协议中断。

当发送仲裁丢失，软件必须再次初始化帧，为新的一次主机发送准备首地址字节的起始条件。仲裁也会发送在ACK位。如果主机仲裁丢失并且匹配地址时，那么主机会转变成从机角色。

#### 6.26.5.8 非应答和错误条件

在一个非应答(NACKIF (USCI\_PROTSTS [10]))或一个错误(ERRIF(USCI\_PROTSTS [12]))状况下，不会有进一步的发送产生。在依据上次事件配置传输（写TXDAT）之前，软件不能使传输缓冲失效并且禁用传输。

#### 6.26.5.9 I2C协议中断事件

下面协议相关的事件是在I<sup>2</sup>C模式下产生的，也可以产生协议中断。

请注意USCI\_PROTSTS中的位不会被硬件自动地清除，必须用软件清除这是为了监控新事件的到来。

- 在帧的正确的位置处接收到起始条件(STARIF (USCI\_PROTSTS [8]))
- 在帧的正确的位置处停止条件被传输(STORIF (USCI\_PROTSTS [9]))
- 主机仲裁丢失(ARBLOIF (USCI\_PROTSTS [11]))
- 从机读请求(SLAREAD (USCI\_PROTSTS [15]))
- 应答接收(ACKIF (USCI\_PROTSTS [13]))
- 非应答接收(NACKIF (USCI\_PROTSTS [10]))
- 起始条件没有在帧期望的位置(ERRIF (USCI\_PROTSTS [12]))

- 停止条件没有在帧期望的位置(ERRFIF (USCI\_PROTSTS [12]))

#### 6.26.5.10 操作 I<sup>2</sup>C

为了操作I<sup>2</sup>C协议，必须考虑到下面的问题：

##### 选择 I<sup>2</sup>C模式：

在运行并且FUNMODE (USCI\_CTL [2:0]) = 000B期间，建议I<sup>2</sup>C的配置参数不要改变。在FUNMODE (USCI\_CTL [2:0]) = 000B时I<sup>2</sup>C控制流程必须完成这样是为了避免意外的输入信号边沿，然后设置FUNMODE (USCI\_CTL [2:0]) = 100B使能I<sup>2</sup>C模式。

- 步骤 1. 设置 FUNMODE (USCI\_CTL [2:0]) = 000B
- 步骤 2. 设置 FUNMODE (USCI\_CTL [2:0]) = 100B

##### 引脚连接：

引脚被用作SDA和SCL时必须由USCI控制器配置成开漏模式来支持I<sup>2</sup>C总线的线与结构

注：使能备用输出端口功能的步骤只能是在I<sup>2</sup>C使能之后，这是为了避免意想不到的脉冲输出

##### 位时序配置：

在标准模式(100 kBit/s)下模块的最小频率是2MHz,然而在快速模式(400 kBit/s)最小频率是10MHz。此外，被用于消除脉冲的数字滤波阶段达到50ns,滤波器的频率需有20MHz。如果另外一个I<sup>2</sup>C参与拉低SCL，在SCL最大 $1/f_{PROT\_CLK}$ 为高的阶段可能会有一个不确定性。请注意SCL最大频率是SAMP\_CLK/2并且必须设置SPCLKSEL (USCI\_BRGEN [3:2])为0选择 $f_{SAMP\_CLK} = f_{DIV\_CLK}$ 。

##### 数据格式配置：

数据格式必须配置为8数据位(DWIDTH (USCI\_LINECTL [11:8]) = 8)并且MSB优先(LSB (USCI\_LINECTL [0]) = 0).最后USCI\_LINECTL必须设置为0x800.

##### 控制流程：

片上I<sup>2</sup>C端口支持三种模式主机模式、从机模式和广播模式。

应用中，I<sup>2</sup>C 端口可以作为主机和从机。在从机模式，I<sup>2</sup>C端口硬件会查找自身从机地址和广播呼叫地址，如果这两个地址的任一个被检测到，并且从机想要从主机接收或向主机发送数据(通过设置AA位)，应答脉冲将会在第9个时钟发出，此时，如果中断使能，则主机和从机设备上都会发生一次中断请求。在微控制器想要成为总线主机时，在进入主机模式之前，硬件需等待到总线空闲，以保证合理的从机动作不会被打断。在主机模式下，如果总线仲裁丢失，I<sup>2</sup>C端口立即切换到从机模式，并可以在同一次串行传输过程中检测自身

为控制I<sup>2</sup>C总线的各种模式传输，用户需要按照寄存器USCI\_PROTSTS的当前状态码来设置USCI\_PROTCTL, USCI\_PROTIEN, TXDAT寄存器。换句话说，每个I<sup>2</sup>C总线动作都要检查USCI\_PROTSTS寄存器的当前状态，然后再设置USCI\_PROTCTL, USCI\_PROTIEN, TXDAT寄存器执行总线动作。最后，通过USCI\_PROTSTS检查响应状态。

在中断标志清除后，寄存器USCI\_PROTCTL的这些位STA, STO 和 AA 用来控制I<sup>2</sup>C硬件的下一个状态。当完成一个新的动作，USCI\_PROTSTS的状态将被更新。如果I<sup>2</sup>C中断控制位USCI\_PROTIEN被设置，新状态对应的动作或者软件将在中断服务程序中被执行。

图 6.26-8显示当前I<sup>2</sup>C STARIF (USCI\_PROTSTS [8])被硬件置1，然后设置TXDAT = SLA+W (从机地址 + 写位), (PTRG, STA, STO, AA) = (1, 0, 0, x)来发送地址到I<sup>2</sup>C总线，写1到STARIF (USCI\_PROTSTS [8])清标志。如果在总线上一个从机地址匹配并且应答ACK，ACKIF (USCI\_PROTSTS [13])会被更新。

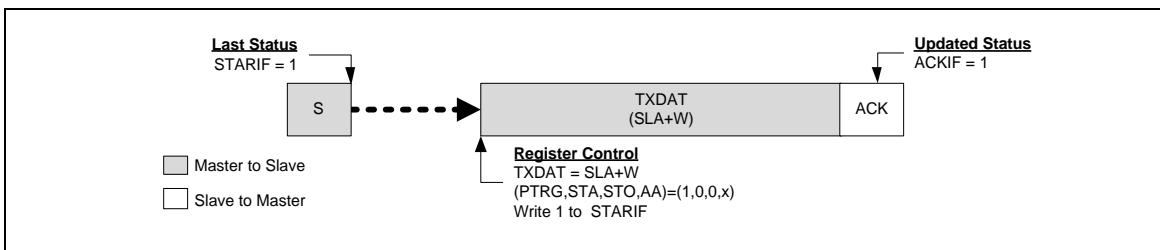


图 6.26-8 依据当I<sup>2</sup>C 状态控制I<sup>2</sup>C 总线

**I<sup>2</sup>C 总线上的数据传输**

图 6.26-9 表示7位地址情况下主机向从机传输数据。主机发出一个7位地址和1位写指示,表示主机想要传送数据给从机。从机回应答给主机之后, 主机继续传输数据。

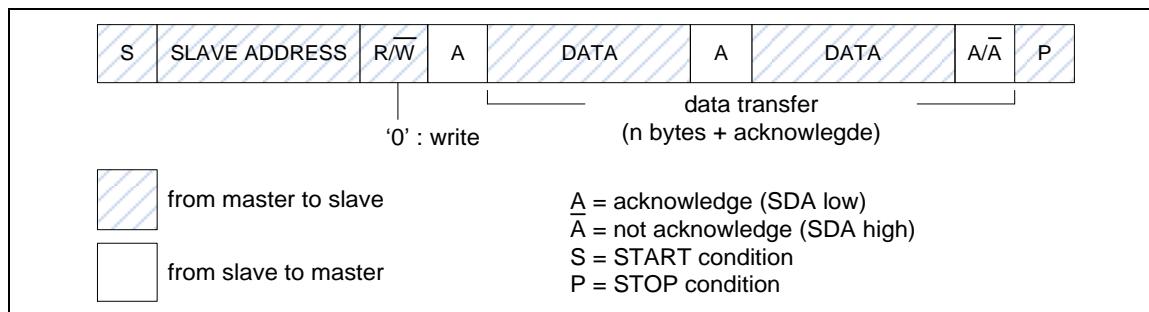


图 6.26-9 7 位地址情况下主机向从机传输数据

图 6.26-10 表示7位地址情况下主机向从机读取数据。主机发7-位地址寻址和1-位读指示, 表示主机要向从机读取数据, 从机返回应答给主机后, 就开始给主机传输数据。

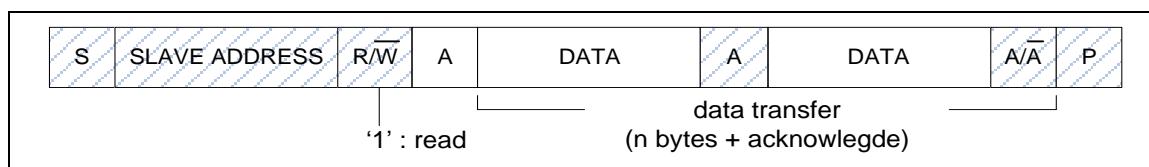


图 6.26-10 7 位地址情况下主机向从机读取数据

图 6.26-11 表示10位地址情况下主机向从机传输数据。主机发送10位地址。首字节是有10位地址指示(5'b11110)和2位地址再加上写指示, 第二字节是剩下的8位地址。在第二字节之后, 主机继续传输数据。注意7位和10位地址设备可以工作在同一总线上。

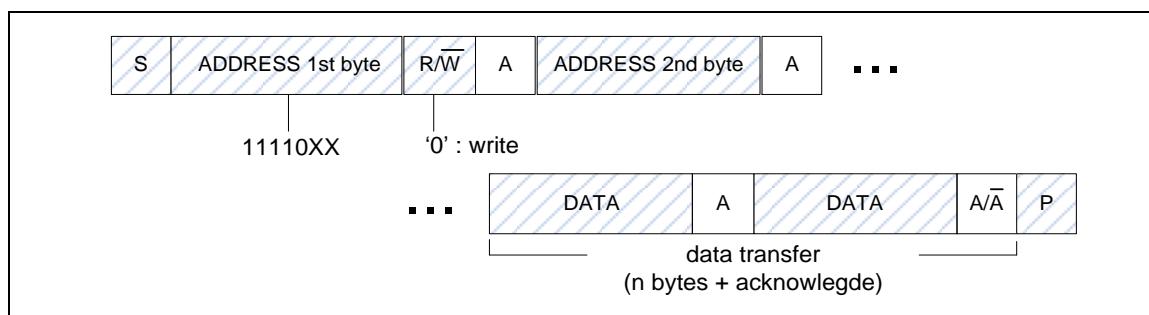


图 6.26-11 10位地址情况下主机向从机传输数据

图 6.26-12 表示10位地址情况下主机向从机读取数据。首先主机发送10位地址给从机, 在主机传输带有读指示的第一个字节之后。在带有读指示的第一个字节之后, 从机就开始给主机传输数据。

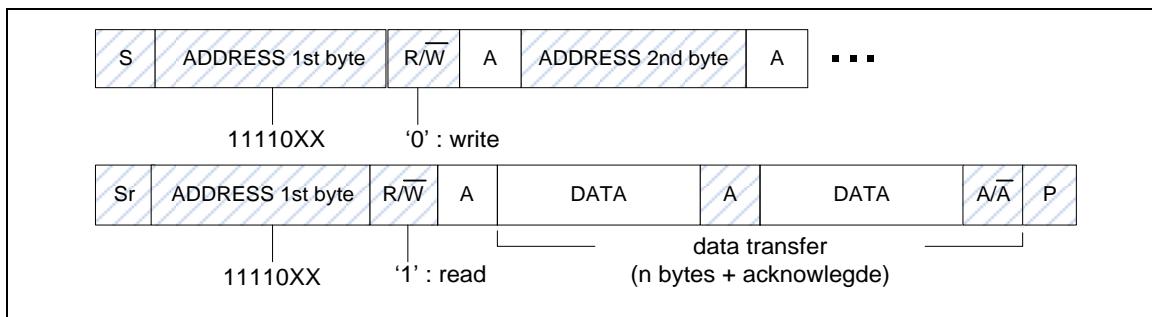


图 6.26-12 10位地址情况下主机向从机读取数据

## 主机模式

图 6.26-13 和图 6.26-14 是 I<sup>2</sup>C 主机模式所有可能的协议展示。用户需要遵循恰当的流程来实现 I<sup>2</sup>C 协议。

换句话说，用户可以发送一个起始信号到总线，I<sup>2</sup>C总线将设置成主机传输(MT)模式(图 6.26-13)，或主机接收(MR)模式(图 6.26-14)。起始信号设置成功后新的状态被设置 STARIF(USCI\_PROTSTS [8])。起始信号后，用户可以发送从机地址，读/写位，数据和重复起始，停止来执行I<sup>2</sup>C协议。

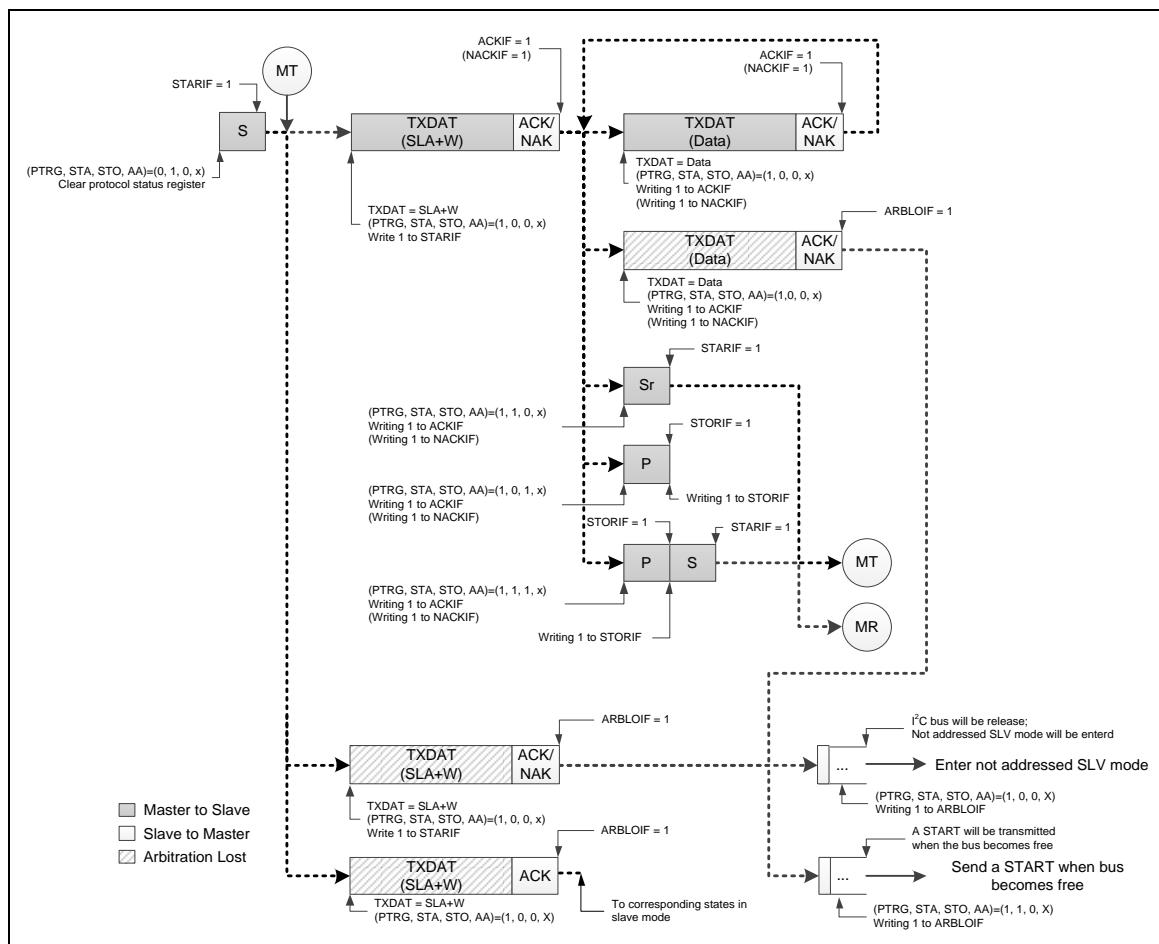


图 6.26-13 7位地址主机发送模式控制流程

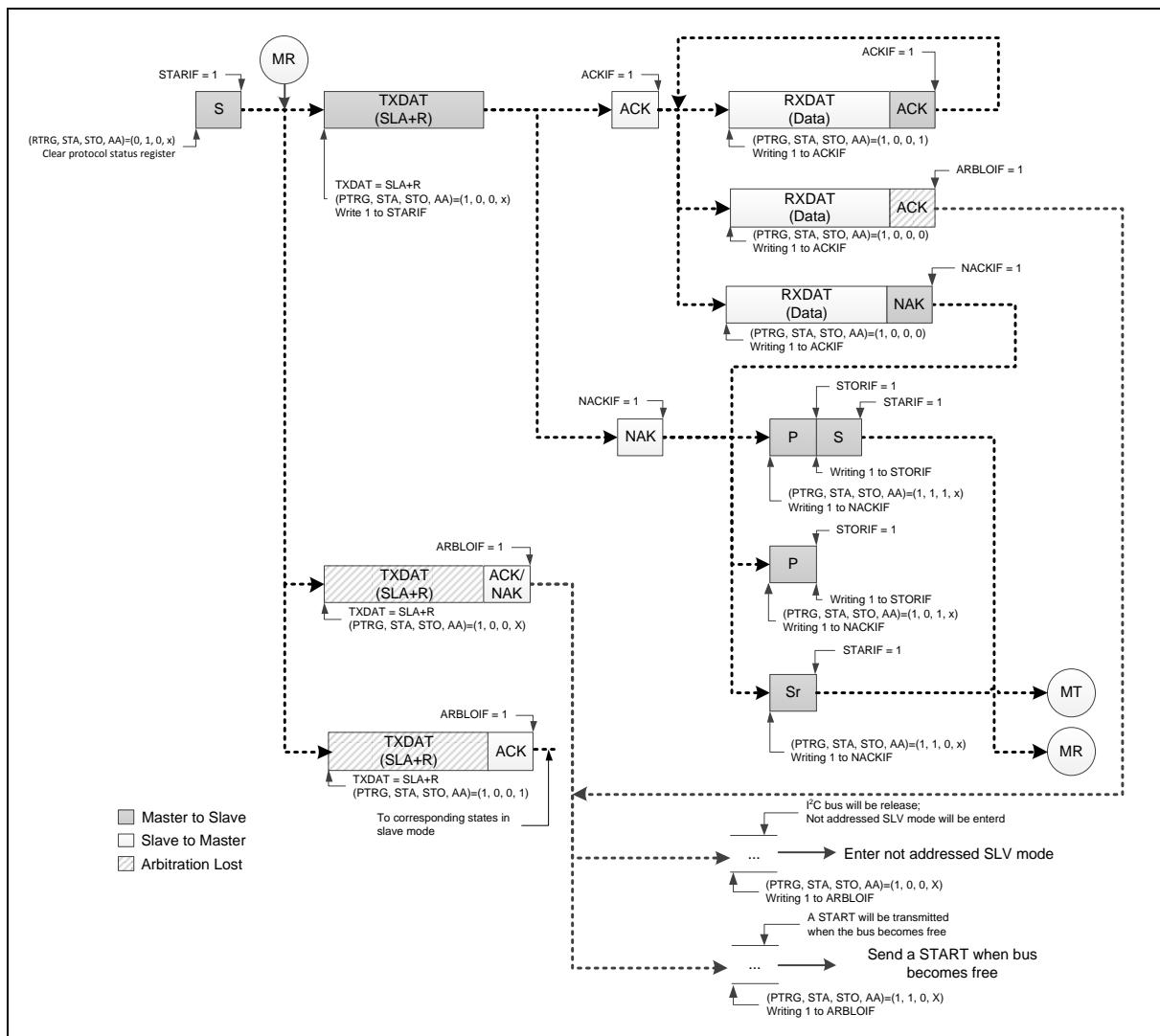


图 6.26-14 7 位地址主机接收模式控制流程

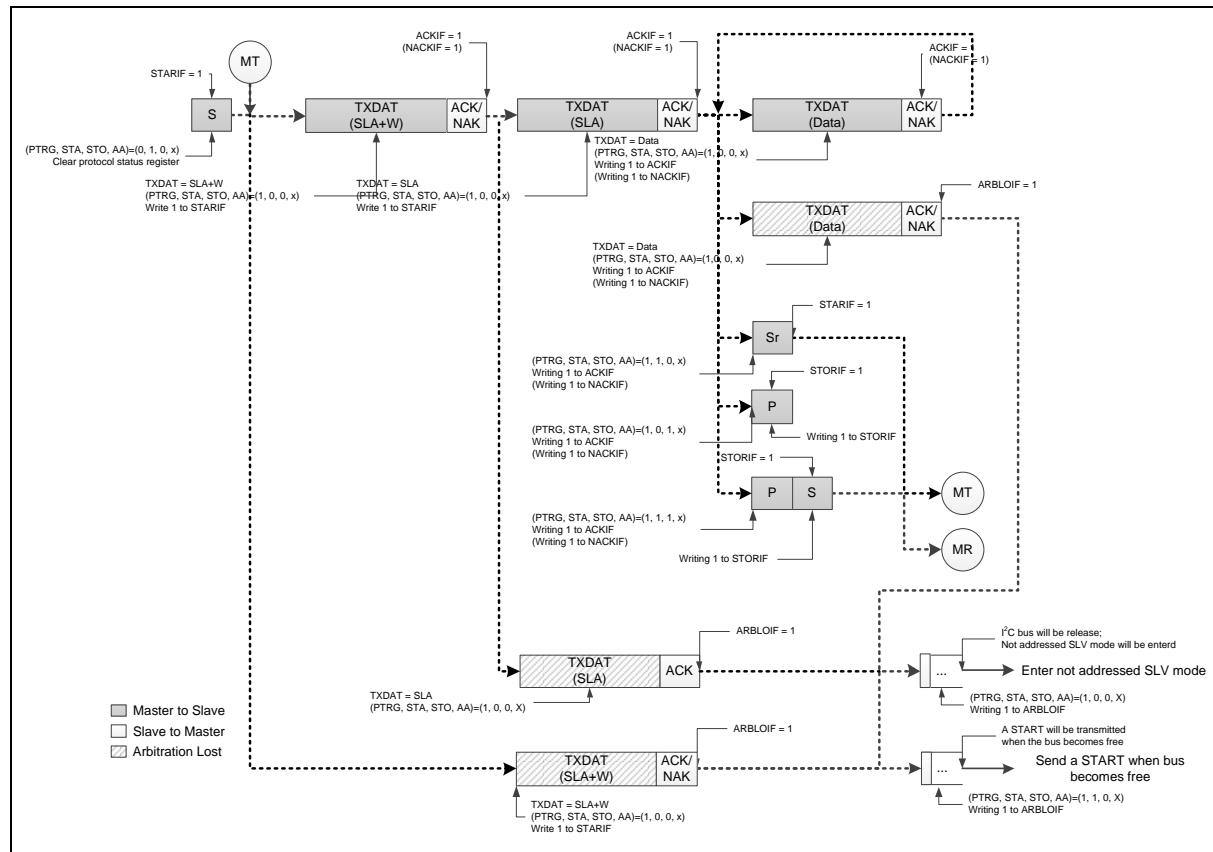


图 6.26-15 10位地址主机发送模式控制流程

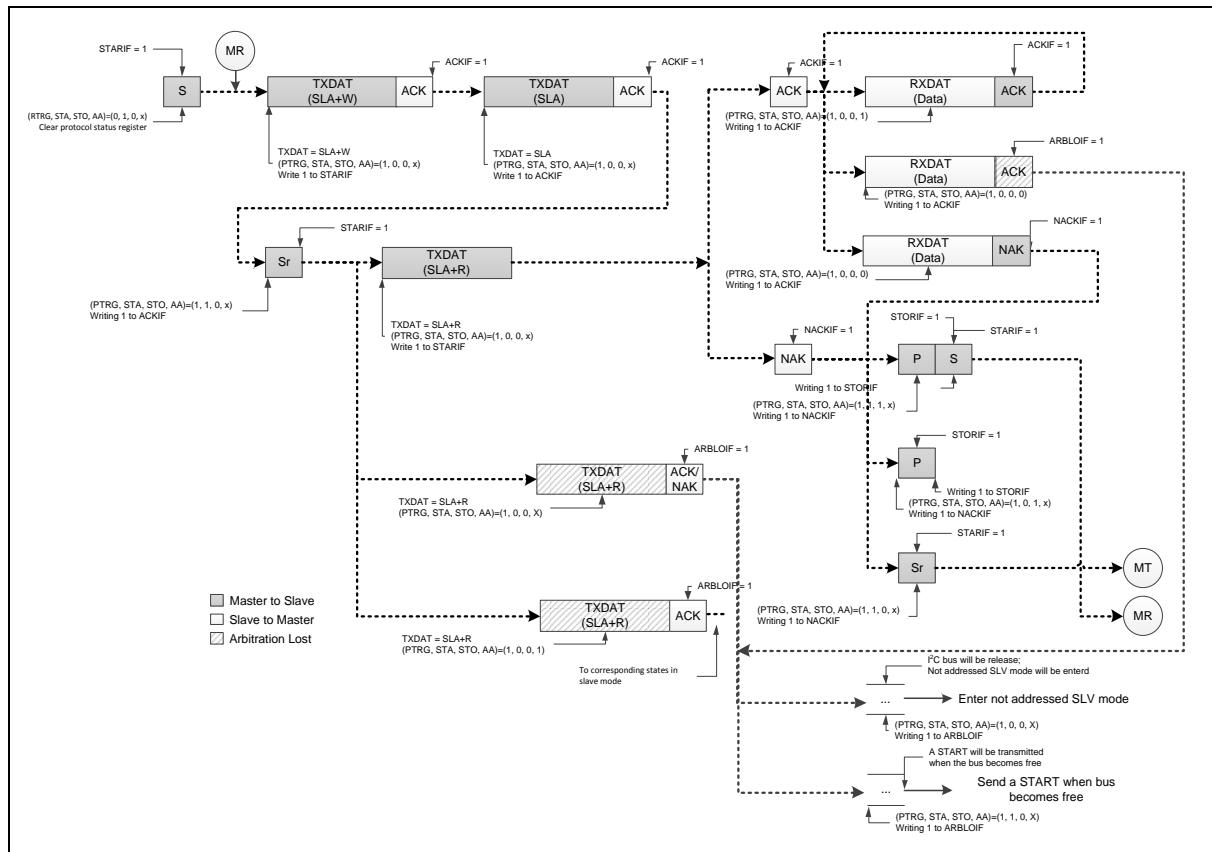


图 6.26-16 10位地址主机接收模式控制流程

如果I<sup>2</sup>C在主机模式并且仲裁丢失，ARBLOIF (USCI\_PROTSTS [11])位被置位。当总线空闲时，用户可以写1到ARBLOIF (USCI\_PROTSTS [11])并且设置(PTRG, STA, STO, AA) = (1, 1, 0, X)发送起始信号来重新开始主机操作。否则，用户可以写1到ARBLOIF (USCI\_PROTSTS [11])并且设置(PTRG, STA, STO, AA) = (1, 0, 0, X)来释放总线，并进入无地址从机模式。

## 从机模式

复位后默认情况下，I<sup>2</sup>C 不会被寻址，并且不会识别 I<sup>2</sup>C 总线上的地址。用户可以通过 USCI\_DEVADDRn 设置从机地址和设置(PTRG, STA, STO, AA) = (1, 0, 0, 1) 来让 I<sup>2</sup>C 识别主机发送的地址。图 6.26-17 所示为 I<sup>2</sup>C 从机模式的所有可能的流程。用户需要遵循（图 6.26-17）的流程来实现他们的 I<sup>2</sup>C 协议。

如果在主机模式总线仲裁丢失，I<sup>2</sup>C 端口立即切换到从机模式，并且在同一串行传输中识别自有的从机地址。如果在仲裁丢失后识别到地址是 SLA+W（主机想写数据到从机或是 SLA+R（主机向从机读数据），ARBLOIF 会被置 1。

**注：**I<sup>2</sup>C 通信期间，在从机模式下，当对 PTRG (USCI\_PROTCTL [5]) 标志写‘1’时，SCL 时钟将被释放。

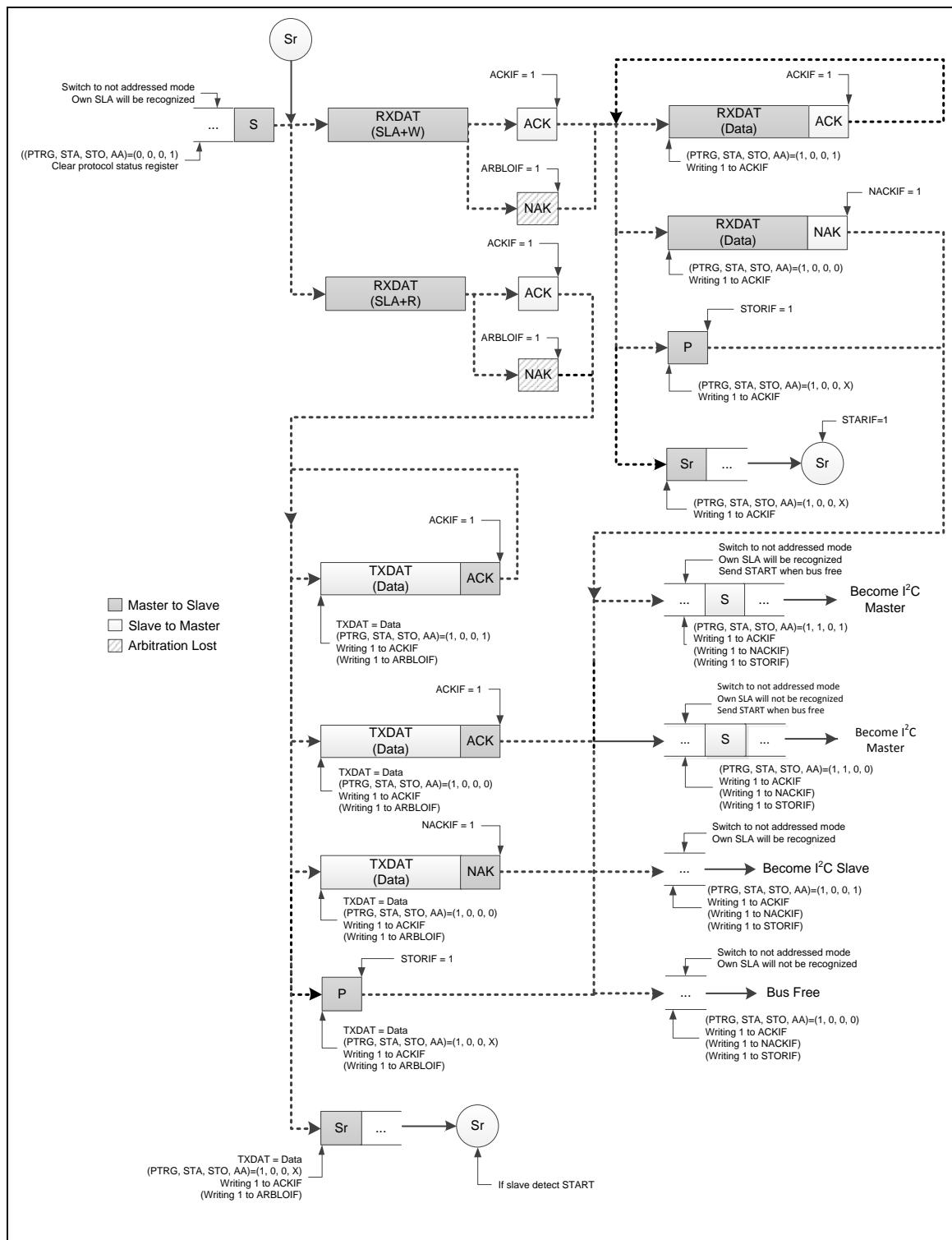


图 6.26-17 7 位地址从机模式控制流程

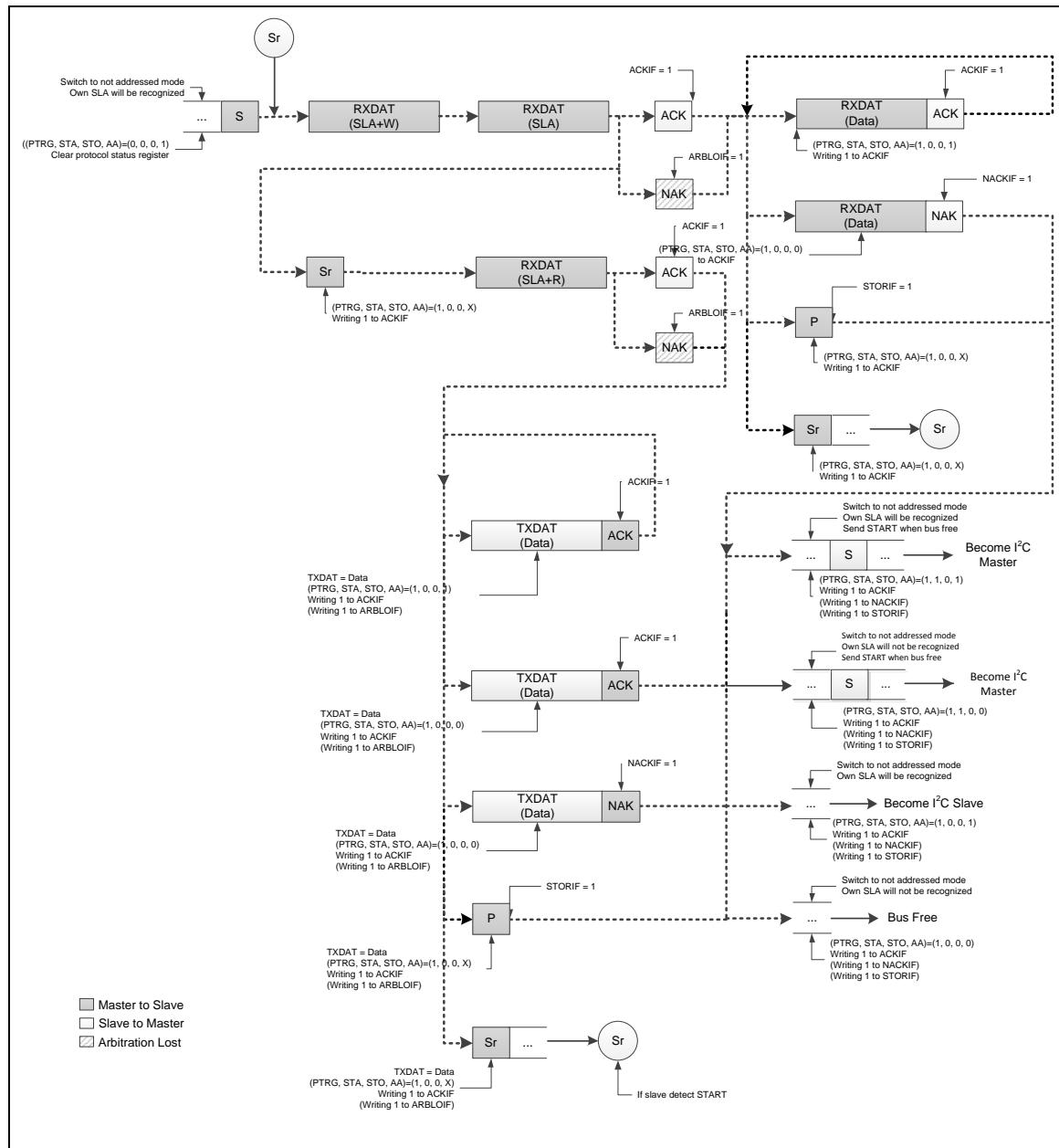


图 6.26-18 10 位地址从机模式控制流程

如果 I<sup>2</sup>C 在可寻址从机模式发送和接收数据时，收到停止或重复起始信号，STORIF (USCI\_PROTSTS [9]) 或 STARIF (USCI\_PROTSTS [8]) 会被置位。当 STARIF (USCI\_PROTSTS [8]) 被设置时，用户可以遵循如上图 NACKIF (USCI\_PROTSTS [10]) 的操作。

**注：**从机获得NACKIF (USCI\_PROTSTS [10]) 和 起始/停止 符号 包括 STARIF (USCI\_PROTSTS [8]) 和 STORIF (USCI\_PROTSTS [9]) 中断状态标志后,从机可以切换到无地址模式,自身SLA不会被辨识。如果进入这种状态,从机不再接收主机任何信号或地址。在这种状态, I<sup>2</sup>C需要设置FUNMODE (USCI\_CTL [2:0]) = 000B离开这个状态。

### 广播呼叫模式 (GC)

如果GCFUNC 位 (USCI\_PROTCTL [0])被设置, I<sup>2</sup>C端口硬件将响应广播呼叫地址(0x00). 用户可以通过清GC位来禁止广播呼叫功能。当I<sup>2</sup>C在从机模式时且GC 位被设置, 可以接收主机地址(0x00)的广播呼叫, 将遵循广播模式状态。

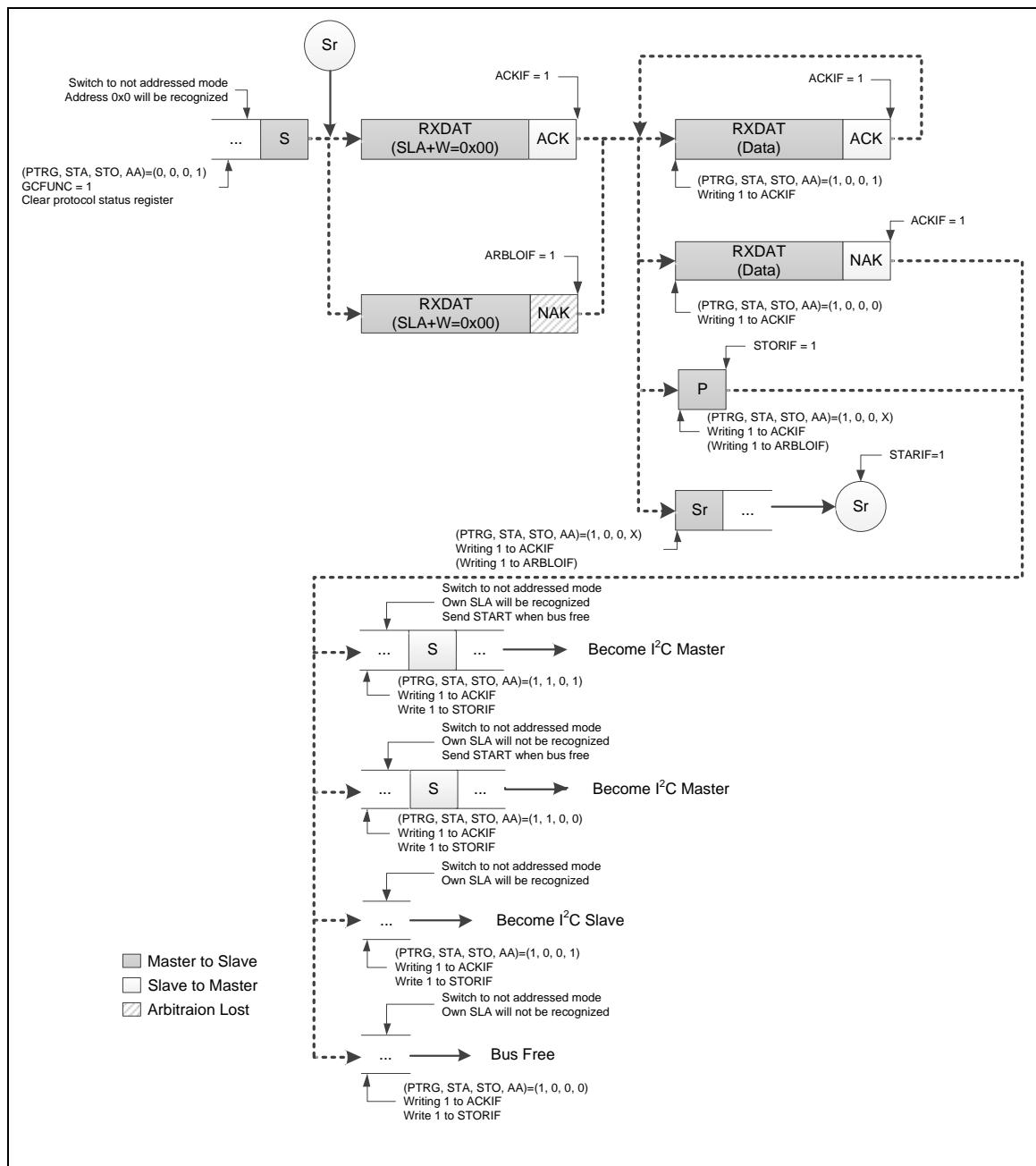


图 6.26-19 7 位地址的 GC 模式

如果I<sup>2</sup>C在广播呼叫模式接收数据时收到停止或重复起始信号， STORIF (USCI\_PROTSTS [9]) 或 STARIF (USCI\_PROTSTS [8])会被置位。当得知STORIF (USCI\_PROTSTS [9]) 或 STARIF (USCI\_PROTSTS [8])被置位时，用户可以遵循如上图NACKIF (USCI\_PROTSTS [10])的流程处理。

**注：**从机获得NACKIF (USCI\_PROTSTS [10]) 和 起始/停止 符号 包括 STARIF (USCI\_PROTSTS [8]) 和 STORIF (USCI\_PROTSTS [9])中断状态标志后，从机可以切换到无地址模式，自身SLA不会被辨识。如果进入这种状态，从机不再接收主机任何信号或地址。在这种状态， I<sup>2</sup>C需要设置FUNMODE (USCI\_CTL [2:0]) = 000B离开这个状态。

## 协议功能描述

- 监视器模式

当I<sup>2</sup>C进入监视器模式，在每次帧接收后设备总是返回NACK给主机即使是地址匹配。此外，设备会存储接收到的任何数据包括地址、命令码和数据。

### 监视器模式下的中断

当MONEN (USCI\_PROTCTL [9])被设置，所有中断处理与正常一样。注意当有初始化START时会发生第一个中断，这个与I<sup>2</sup>C从机不一样，但是其它中断是一样的。

地址匹配侦测之后，中断会在每次数据如同从机模式控制流程一样被接收之后或是再模块信任的每个字节作为从机读传输被传输之后产生。在第二个情况中，数据寄存器实际上会包含总线上一些其它被主机寻址的从机发送的数据。如果用户想监控其它设备，用户可以设置地址掩码和监视器。

如果监视器没有时间响应中断，当SCLOUTEN (USCI\_PROTCTL [8])被置1时SCL信号会被拉低。当SCLOUTEN (USCI\_PROTCTL [8])被置1时，用户必须设置PTRG (USCI\_PROTCTL [5])释放总线。如果SCLOUTEN (USCI\_PROTCTL [8])没有被置1，用户就不需要置PTRG (USCI\_PROTCTL [5])为1。

当设备地址匹配时，但是设备会响应NACK，该地址会被接收到缓冲中并且会产生NACK中断。

接下来的所有中断，处理器可能需要读数据寄存器来看总线上实际传输的是什么。

### 监视器模式下仲裁丢失

在监视器模式下，I<sup>2</sup>C模块不会响应由总线主机产生的ACK信息请求。总线上的其它从机会去响应。软件应该知道在模块处在监视器模式下并且不能检测到的任何仲裁丢失情况。

- 可编程的建立和保持时间

为了保证一个正确数据的建立和保持时间，则时序必须可以配置。通过编程HTCTL (USCI\_TMCTL[24:16])来配置保持时间，通过编程STCTL (USCI\_TMCTL[8:0])来配置建立时间

延时时序参考外设时钟(PCLK).当设备拉住主机时钟时，建立和保持时间的配置值不会被影响。

用户需要集中精力在建立和保持时间配置的极限上，时序设置必须遵从I<sup>2</sup>C协议。一旦建立时间配置大于设计极限，那就意味着如果设置的建立时间让SCL输出小于三个PCLK，由于SCL采样为三次那么I<sup>2</sup>C控制器就不能正常工作。一旦保持时间配置大于I<sup>2</sup>C时钟的极限，I<sup>2</sup>C将会发送总线错误。在设置时序前建议用户结合传输速率和协议计算好合理的时序。表 6.26-1 显示了 I<sup>2</sup>C 传输速率与 PCLK 之间的关系，表格呈现的数是一个时钟间隙包含多少 PCLK。建立和保持时间的配置哪怕在我们设计中编程一些极限值，但是用户必须要遵从I<sup>2</sup>C的标准协议。

PCLK	I <sup>2</sup> C传输速率	100k	200k	400k	800k	1200k
12MHz	120	60	30	15	10	
24MHz	240	120	60	30	20	
48MHz	480	240	120	60	40	

72MHz	720	360	180	90	60
-------	-----	-----	-----	----	----

表 6.26-1 I<sup>2</sup>C 传输速率与 PCLK 之间的关系

对于建立时间错误调整范例，我们假设一个 SCL 周期包含 5 个 PCLK 并且设置 STCTL (USCI\_TMCTL[8:0]) 为 3 那样对于建立时间设置拉伸三个 PCLK。建立时间最大设置值： $ST_{limit} = (USCI_BRGEN[25:16]+1) - 6$ .

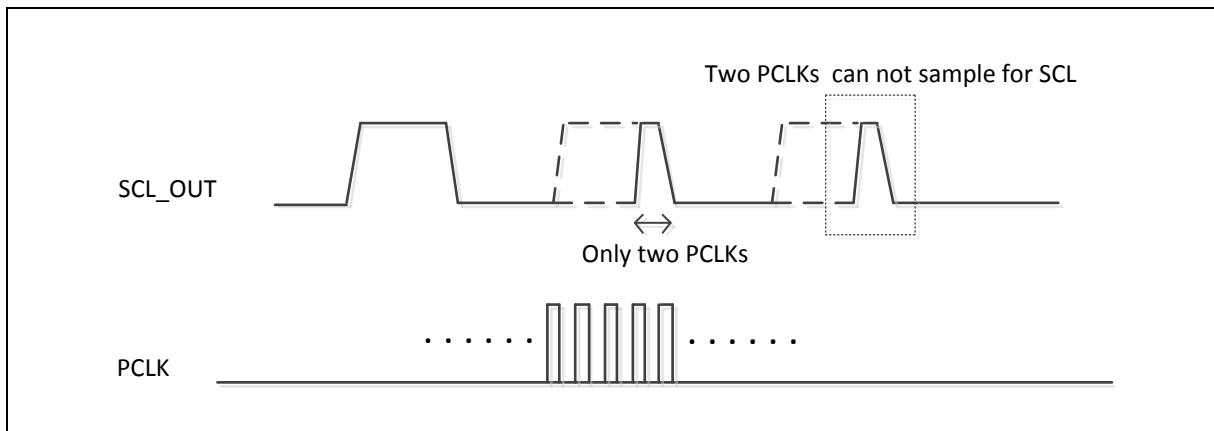


图 6.26-2 建立时间错误调整

对于保持时间错误调整范例，我们使用 I<sup>2</sup>C 传输速率=1200K 和 PCLK=72MHz，SCL 的高电平/低电平空隙=60PCLK。当我设置 HTCTL (USCI\_TMCTL[24:16]) 为 63 和 STCTL (USCI\_TMCTL[8:0]) 为 0 时，然后 SDA 输出延时将会越过 SCL 为高的空隙，以至产生总线错误。保持时间最大值的是设定： $HT_{limit} = (USCI_BRGEN[25:16]+1) - 9$ .

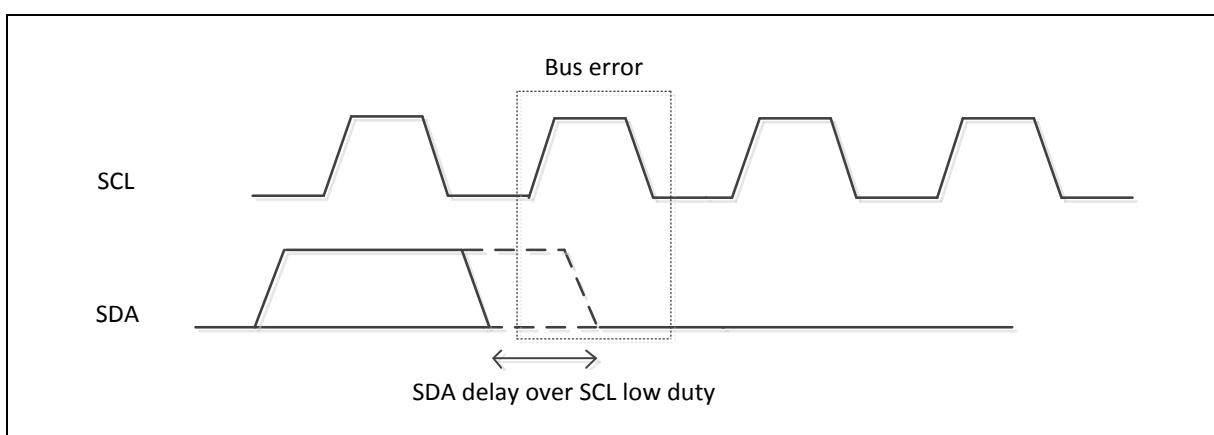


图 6.26-20 保持时间错误调整

## I<sup>2</sup>C 超时功能

系统提供一个10位超时的计数器TOCNT (USCI\_PROTCTL [25:16])来处理当I<sup>2</sup>C总线锁死时的情况。当超时计数器计数功能使能后，计数器开始上计数直至等于TOCNT (USCI\_PROTCTL [25:16])并向CPU产生I<sup>2</sup>C中断或者清除TOIEN (USCI\_PROTIEN [0])为0停止计数或是设置所有的I<sup>2</sup>C中断信号(ACKIF, ERRIF, ARBLOIF, NACKIF, STORIF, STARIF)。用户需要写1清TOIF(USCI\_PROTSTS[5])为0。当超时计数器使能，对TOIF写1会使计数器复位，TOIF被清除后重新开始计数。对于超时计数器TOCNT (USCI\_PROTCTL [25:16])、]。  $T_{TOCNT} = (TOCNT \times f_{SAMP\_CLK}) + 1$  x32 (5-bit) x T<sub>PCLK</sub>请参考图 6.26-21。请注意时间计数器时钟源TMCNTSRC (USCI\_BRGEN [5])必须设置为0。

如果I<sup>2</sup>C总线锁死，会使I<sup>2</sup>C\_STATUS 及SI 标志不再更新，该14-位超时计数器会发生溢出从而产生I<sup>2</sup>C中断通知CPU。参考下图14-位超时计数器。用户可以写1清TOIF为0。

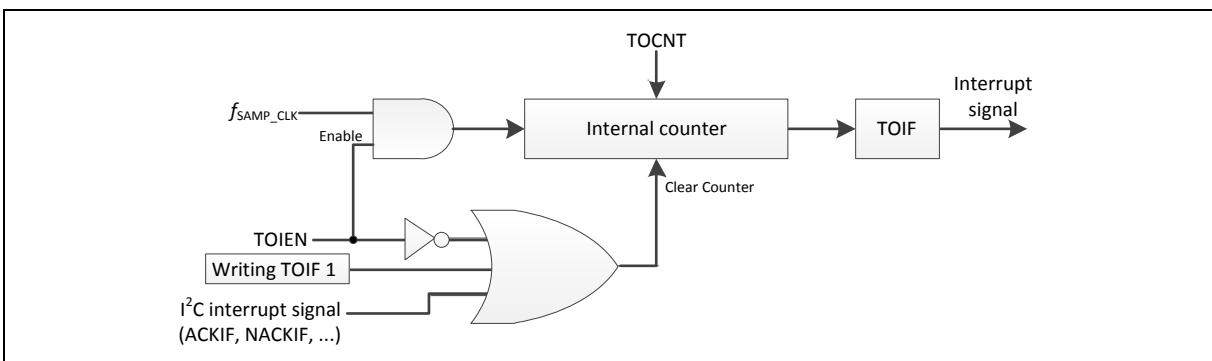


图 6.26-21 I<sup>2</sup>C 超时计数模块框图

## 唤醒功能

当芯片进入掉电模式并且WKEN (WKCTL[0])被设置为1时，其它I<sup>2</sup>C主机通过寻址我们I<sup>2</sup>C设备唤醒我们芯片，在进入睡眠前必须配置相关设置。地址匹配帧的ACK位阶段是在掉电情况下完成的。当地址与设备地址匹配时并且ACK阶段过后，控制器会拉低SCL。直到用户清除标志位SCL释放。如果SCL的频率是低速并且系统从地址匹配帧中唤醒，用户需要检查该位来确认该帧传输已经完成然后去处理唤醒的事务。因此，当芯片被设备地址寄存器(USCI\_DEVADDRn)其中之一地址匹配唤醒时，用户需要检查WKAKDONE (USCI\_PROTSTS [16])是否被置1来确认地址字节已完成。WKAKDONE位表明在掉电时候ACK阶段已完成。当地址与设备地址匹配时并且ACK阶段过后，控制器会拉低SCL，直到WKAKDONE被用户清除。如果SCL是低频速率并且系统被地址匹配帧唤醒，用户需要检查WKAKDONE来确认该帧已完成传输然后去做唤醒后的事务。请注意用户必须在清除WKAKDONE为0之后清除WKUPIF。

WRSTSWK (USCI\_PROTSTS [17])位用来记录在地址匹配唤醒帧中的读/写命令。在系统被地址匹配帧唤醒后，用户读该位的状态来准备下次传输数据(WRSTSWK = 0)或是等待到来的数据(WRSTSWK = 1)及时地被存储。

当系统被其他的I<sup>2</sup>C主机设备唤醒时，WKF(USCI\_WKSTS[0])被置位表示该事件发生。用户需要写“1”来清除此位。

### EEPROM随机读取范例

通过下面的步骤来配置USCI0\_I<sup>2</sup>C相关寄存器，来使用I<sup>2</sup>C从EEPROM读取数据。

1. 在SYS\_GP1\_MFPL或SYS\_GP1MFPH或SYS\_GP2\_MFPL或SYS\_GP4\_MFPL寄存器中将USCI0\_I2C多功能管脚设置成SCL和SDA管脚。
2. 通过设置寄存器CLK\_APBCLK的USCI0CKEN为1使能USCI0 APB时钟。
3. 通过设置USCI0RST=1来复位I<sup>2</sup>C控制器，然后通过设置USCI0RST=0将USC控制器设置成普通操作。USCI0RST位在SYS\_IPRST2[8]寄存器中。
4. 通过设置寄存器USCI\_CTL中的FUNMODE =100使能USCI0\_I<sup>2</sup>C控制器。
5. 通过在USCI\_BRGEN寄存器写USCI0\_I<sup>2</sup>C时钟分频值。
6. 在“NVIC\_ISR”寄存器中设置SETENA =0x00400000来设置USCI\_IRQ。
7. 设置寄存器USCI\_PROTIEN的ACKIEN,ERRIEN,ARBLOIEN,NACKIEN,STORIEN,STARIEN,和TOIEN使能I<sup>2</sup>C中断
8. 设置USCI地址寄存器“USCI\_ADDR0 ~ USCI\_ADDR1”。

随机读操作是访问EEPROM的其中一种方法。这个方法是允许主机访问EEPROM的任何一个地址。下图显示对EEPROM随机读取操作。

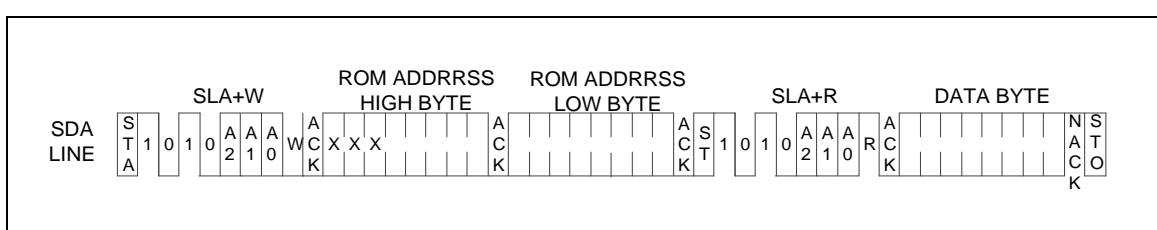


图 6.26-22 EEPROM 随机读取

图 6.26-23 显示怎样使用I<sup>2</sup>C控制器执行EEPROM随机读取操作协议

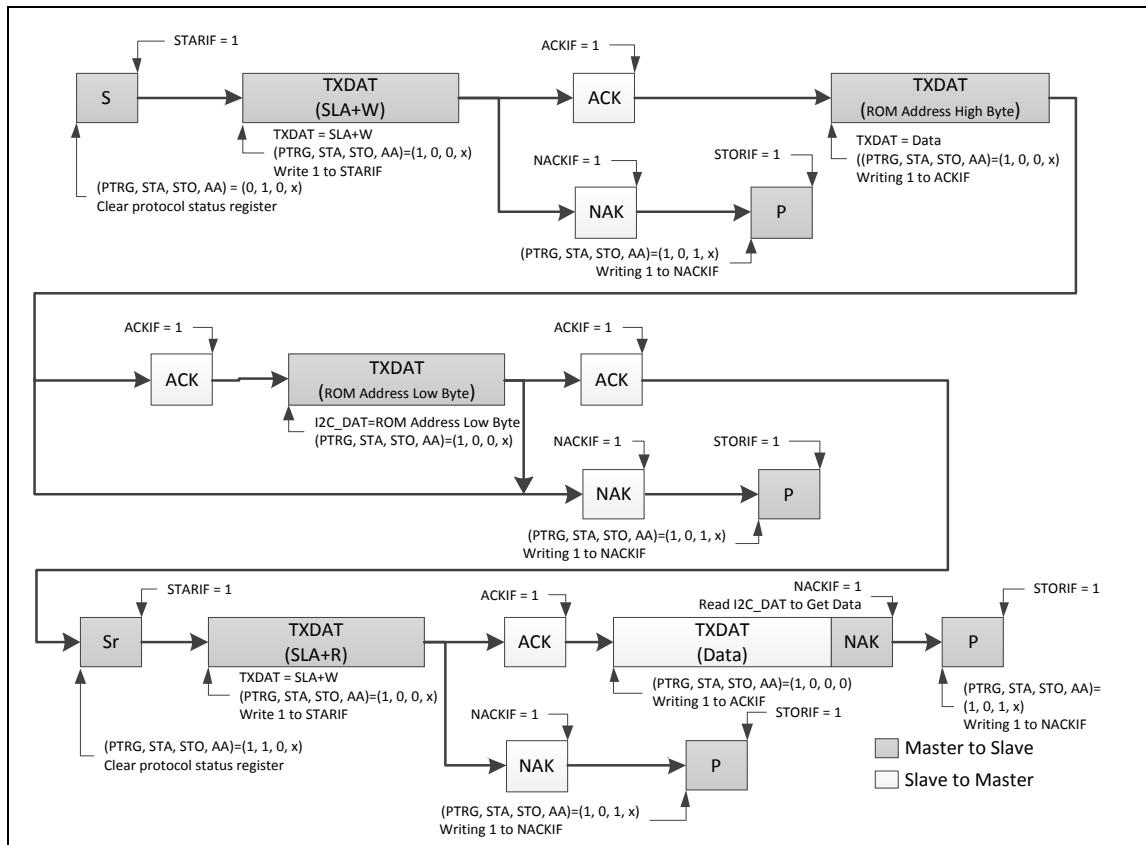


图 6.26-23 EEPROM 随机读取协议

I<sup>2</sup>C控制器作为主机发送起始信号到总线，然后发送SLA+W（从机地址 + 写位）到EEPROM，跟着由两个字节数据地址来设置EEPROM被读的地址。最后，重复起始信号跟着SLA+R被发送来向EEPROM读取数据。

### 6.26.6 寄存器映射

R: 只读, W: 只写, R/W: 读/写

寄存器	偏移地址	R/W	描述	复位值
<b>USCI 基址:</b>				
<b>USCI<sub>n</sub>_BA = 0x400D_0000 + (0x1000 * n)</b>				
<b>N= 0, 1</b>				
<b>USCI_CTL</b>	USCI <sub>n</sub> _BA+0x00	R/W	USCI 控制寄存器	0x0000_0000
<b>USCI_BRGEN</b>	USCI <sub>n</sub> _BA+0x08	R/W	USCI 波特率发生器寄存器	0x0000_3C00
<b>USCI_LINECTL</b>	USCI <sub>n</sub> _BA+0x2C	R/W	USCI 线控制寄存器	0x0000_0000
<b>USCI_TXDAT</b>	USCI <sub>n</sub> _BA+0x30	W	USCI 发送数据寄存器	0x0000_0000
<b>USCI_RXDAT</b>	USCI <sub>n</sub> _BA+0x34	R	USCI 接收数据寄存器	0x0000_0000
<b>USCI_DEVADDR0</b>	USCI <sub>n</sub> _BA+0x44	R/W	USCI 设备地址寄存器 0	0x0000_0000
<b>USCI_DEVADDR1</b>	USCI <sub>n</sub> _BA+0x48	R/W	USCI 设备地址寄存器1	0x0000_0000
<b>USCI_ADDRMSK0</b>	USCI <sub>n</sub> _BA+0x4C	R/W	USCI 设备地址掩码寄存器 0	0x0000_0000
<b>USCI_ADDRMSK1</b>	USCI <sub>n</sub> _BA+0x50	R/W	USCI 设备地址掩码寄存器1	0x0000_0000
<b>USCI_WKCTL</b>	USCI <sub>n</sub> _BA+0x54	R/W	USCI 唤醒控制寄存器	0x0000_0000
<b>USCI_WKSTS</b>	USCI <sub>n</sub> _BA+0x58	R/W	USCI 唤醒状态寄存器	0x0000_0000
<b>USCI_PROTCTL</b>	USCI <sub>n</sub> _BA+0x5C	R/W	USCI 协议控制寄存器	0x0000_0000
<b>USCI_PROTIEN</b>	USCI <sub>n</sub> _BA+0x60	R/W	USCI 协议中断使能寄存器	0x0000_0000
<b>USCI_PROTSTS</b>	USCI <sub>n</sub> _BA+0x64	R/W	USCI 协议状态寄存器	0x0000_0000
<b>USCI_TMCTL</b>	USCI <sub>n</sub> _BA+0x8C	R/W	I <sup>2</sup> C时序配置控制寄存器	0x0000_0000

### 6.26.7 寄存器描述

#### USCI\_CTL USCI 控制寄存器

寄存器	偏移地址	R/W	描述	复位值
USCI_CTL	USCIIn_BA+0x00	R/W	USCI控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved					FUNMODE		

位	描述	
[31:3]	Reserved	保留
[2:0]	FUNMODE	<p><b>功能模式</b></p> <p>该域是选择USCI控制器的协议。待选的协议是不可用的。当两个协议之间切换时，在选择新协议前USCI必须先禁用。在用户写000到FUNMODE同时USCI会被复位。</p> <p>000 = 禁用 USCI，所有协议相关的状态机器被设置为空闲状态。</p> <p>001 = 选择SPI 协议</p> <p>010 = 选择 UART 协议</p> <p>100 = 选择I<sup>2</sup>C 协议</p> <p><b>注:</b>其它值保留</p>

## USCI\_BRGEN USCI 波特率发生器寄存器

寄存器	偏移地址	R/W	描述	复位值
USCI_BRGEN	USCIIn_BA+0x08	R/W	USCI波特率发生器寄存器	0x0000_3C00

31	30	29	28	27	26	25	24
Reserved						CLKDIV	
23	22	21	20	19	18	17	16
CLKDIV							
15	14	13	12	11	10	9	8
Reserved	DSCNT					PDSCNT	
7	6	5	4	3	2	1	0
Reserved		TMCNTSRC	TMCNTEN	SPCLKSEL		PTCLKSEL	RCLKSEL

位	描述	
[31:26]	Reserved	保留
[25:16]	CLKDIV	<p><b>时钟分频器</b>            该域是定义协议时钟频率 <math>f_{PROT\_CLK}</math> 和时钟分频器频率 <math>f_{DIV\_CLK}</math> (<math>f_{DIV\_CLK} = f_{PROT\_CLK} / (CLKDIV+1)</math>)之间的比率  <b>注:</b> 在UART功能下, 当自动波特率功能(ABREN(USCI_PROTCTL[6]))使能时, 会在输入数据0x55的第四个下降沿时被硬件更新。修改值是在位5和位6之间的时间平均值。用户可以使用修改的CLKDIV和新的BRDETTIV (USCI_PROTCTL[24:16])来计算精确的波特率</p>
[15]	Reserved	保留
[14:10]	DSCNT	<p><b>采样计数器的分母</b>            该域定义采样时钟 <math>f_{SAMP\_CLK}</math> 的分频比。            分频 <math>f_{DS\_CNT} = f_{PDS\_CNT} / (DSCNT+1)</math>.  <b>注:</b> 在UART模式下, DSCNT最大值是0xF, 建议设置4以上来保证接收数据的采样时正确的值。</p>
[9:8]	PDSCNT	<p><b>采样计数器预分频器</b>            该域定义的是对来自采样时钟 <math>f_{SAMP\_CLK}</math> 的分频比            分频 <math>f_{PDS\_CNT} = f_{SAMP\_CLK} / (PDSCNT+1)</math>.</p>
[7:6]	Reserved	保留
[5]	TMCNTSRC	<p><b>时序测量计数器时钟源选择</b>            0 = 来自 <math>f_{PROT\_CLK}</math>.            1 = 来自 <math>f_{DIV\_CLK}</math>.</p>
[4]	TMCNTEN	<p><b>时序测量计数器使能位</b>            该位是使能10位时序测量计数器            0 = 禁用时序测量计数器            1 = 使能时序测量计数器</p>

[3:2]	<b>SPCLKSEL</b>	<b>采样时钟源选择</b> 该域用于对协议采样时钟( $f_{SAMP\_CLK}$ )的时钟源选择 00 = $f_{SAMP\_CLK} = f_{DIV\_CLK}$ . 01 = $f_{SAMP\_CLK} = f_{PROT\_CLK}$ . 10 = $f_{SAMP\_CLK} = f_{SCLK}$ . 11 = $f_{SAMP\_CLK} = f_{REF\_CLK}$ .
[1]	<b>PTCLKSEL</b>	<b>协议时钟源选择</b> 该位用于选择协议时钟( $f_{PROT\_CLK}$ )的源信号 0 = 参考时钟 $f_{REF\_CLK}$ . 1 = $f_{REF\_CLK2}$ (频率是 $f_{REF\_CLK}$ 一半)
[0]	<b>RCLKSEL</b>	<b>参考时钟源选择</b> 该位是选择参考时钟( $f_{REF\_CLK}$ )源信号 0 = 外设设备时钟 $f_{PCLK}$ . 1 = 保留

## USCI\_LINECTL USCI线控制寄存器

寄存器	偏移地址	R/W	描述	复位值
USCI_LINECTL	USCIIn_BA+0x2C	R/W	USCI线控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved				DWIDTH			
7	6	5	4	3	2	1	0
Reserved							LSB

位	描述	
[31:12]	<b>Reserved</b>	保留
[11:8]	<b>DWIDTH</b>	<p><b>传输的字长</b></p> <p>该域定义发送和接收的数据字长。在数据缓冲中数据总是右对齐。USCI支持4到16位的字长</p> <p>0x0:在数据字中包含16位，位于[15:0]这些位中</p> <p>0x1: 保留</p> <p>0x2: 保留</p> <p>0x3: 保留</p> <p>0x4:在数据字中包含4位，位于[3:0]这些位中</p> <p>0x5:在数据字中包含5位，位于[4:0]这些位中</p> <p>...</p> <p>0xF:在数据字中包含15位，位于[14:0]这些位中</p> <p><b>注:</b> 在UART模式下，长度可配置为6~13位。在I<sup>2</sup>C协议下长度固定为8位。</p>
[7:1]	<b>Reserved</b>	保留
[0]	<b>LSB</b>	<p><b>LSB优先传输选择</b></p> <p>0 =取决于DWIDTH设置的接收/发送数据缓冲的MSB先发送/接收</p> <p>1 =数据缓冲的位0及LSB被先发送/接收</p>

USCI\_TXDAT USCI 发送数据寄存器

寄存器	偏移地址	R/W	描述	复位值
USCI_TXDAT	USCIIn_BA+0x30	W	USCI发送数据寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
TXDAT							
7	6	5	4	3	2	1	0
TXDAT							

位	描述	
[31:16]	Reserved	保留
[15:0]	TXDAT	发送数据 软件可以写16位发送数据到该域用来发送

USCI\_RXDAT USCI 接收数据寄存器

寄存器	偏移地址	R/W	描述	复位值
USCI_RXDAT	USCIIn_BA+0x34	R	USCI接收数据寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
RXDAT							
7	6	5	4	3	2	1	0
RXDAT							

位	描述	
[31:16]	Reserved	保留
[15:0]	RXDAT	<p><b>接收的数据</b>            该域值监控存储在接收数据缓冲的接收到的数据</p> <p><b>注 1:</b>在I<sup>2</sup>C协议下， RXDAT[12:8]表示不同的传输条件，该条件定义在I<sup>2</sup>C中。</p> <p><b>注 2:</b> UART 协议下， RXDAT[15:13] 表明 BREAK, FRMERR 和 PARITYERR (USCI_PROTSTS[7:5])相同帧状态</p>

USCI\_DEVADDR USCI 设备地址寄存器

寄存器	偏移地址	R/W	描述	复位值
USCI_DEVADDR0	USCIIn_BA+0x44	R/W	USCI设备地址寄存器0	0x0000_0000
USCI_DEVADDR1	USCIIn_BA+0x48	R/W	USCI设备地址寄存器1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved						DEVADDR	
7	6	5	4	3	2	1	0
DEVADDR							

位	描述	
[31:10]	Reserved	保留
[9:0]	DEVADDR	<p><b>设备地址</b></p> <p>在 I<sup>2</sup>C 协议下，该位域包含可编程的从机地址。如果第一个接收到地址字节是 1111 0AAAX<sub>B</sub>，AA位会与 DEVADDR[9:8]比较检查地址地址匹配，X是读/写位。第二个地址字节会和 DEVADDR[7:0]比较。</p> <p><b>注 1:</b> 当 I<sup>2</sup>C 工作在7位地址模式时， DEVADDR [9:7]必须设置为3'b000</p> <p><b>注 2:</b>当软件设置10'h000时，地址不能被使用</p>

USCI\_ADDRMSK USCI 设备地址掩码寄存器- 仅针对I<sup>2</sup>C

寄存器	偏移地址	R/W	描述	复位值
USCI_ADDRMSK0	USCIIn_BA+0x4C	R/W	USCI设备地址掩码寄存器0	0x0000_0000
USCI_ADDRMSK1	USCIIn_BA+0x50	R/W	USCI设备地址掩码寄存器1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved						ADDRMSK	
7	6	5	4	3	2	1	0
ADDRMSK							

位	描述	
[31:10]	Reserved	保留
[9:0]	ADDRMSK	<p><b>USCI设备地址掩码</b></p> <p>0 = 掩码禁用 (接收的相关地址位必须完全与地址寄存器一致)</p> <p>1 = 使能掩码 (不用关心接收相关地址未)</p> <p>2组地址掩码寄存器使得USCI支持多地址识别。当在地址掩码寄存器中的位被置1，意味着不用关心接收的相关地址位。如果设置为0，意味着接收的相关地址位必须和地址寄存器一致。</p> <p><b>注:</b>唤醒功能不能使用地址掩码</p>

USCI\_WKCTL USCI 唤醒控制寄存器

寄存器	偏移地址	R/W	描述	复位值
USCI_WKCTL	USCIIn_BA+0x54	R/W	USCI唤醒控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved						WKADDREN	WKEN

位	描述	
[31:2]	<b>Reserved</b>	保留
[1]	<b>WKADDREN</b>	唤醒地址匹配使能位 0 =芯片依据数据变化唤醒 1 =芯片依据地址匹配唤醒
[0]	<b>WKEN</b>	唤醒使能位 0 =禁用唤醒功能 1 =使能唤醒功能

**USCI\_WKSTS USCI 唤醒状态寄存器**

寄存器	偏移地址	R/W	描述	复位值
USCI_WKSTS	USCIIn_BA+0x58	R/W	USCI唤醒状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved							WKF

位	描述	
[31:1]	Reserved	保留
[0]	WKF	唤醒标志 当芯片被从掉电模式唤醒，该位置1.软件写1清除该位。

**USCI PROTCTL USCI协议控制寄存器 – I<sup>2</sup>C**

寄存器	偏移地址	R/W	描述	复位值
USCI_PROTCTL	USCIIn_BA+0x5C	R/W	USCI 协议控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
PROTEN	Reserved						TOCNT
23	22	21	20	19	18	17	16
TOCNT							
15	14	13	12	11	10	9	8
Reserved						MONEN	SCLOUTEN
7	6	5	4	3	2	1	0
Reserved		PTRG	ADDR10EN	STA	STO	AA	GCFUNC

位	描述
[31]	<b>PROTEN</b> I <sup>2</sup> C 协议使能位 0 =禁用 I <sup>2</sup> C 协议 1 = 使能I <sup>2</sup> C 协议
[30:26]	<b>Reserved</b> 保留
[25:16]	<b>TOCNT</b> <b>超时时钟周期</b> 当每个中断被清除的时候，该位域表示需要多少时钟周期（时钟源由TMCNTSRC (USCI_BRGEN [5])选择）。当TOCNT大于0时超时使能。 <b>注:</b> 在I <sup>2</sup> C 模式下， TMCNTSRC (USCI_BRGEN [5]) 必须设置为0
[15:10]	<b>Reserved</b> 保留
[9]	<b>MONEN</b> <b>监视器模式使能位</b> 该位使能监视器模式。在监视器模式下， SDA会以高阻态输入。这样就可以阻止I <sup>2</sup> C模块输出任何数据（包括ACK）到I <sup>2</sup> C数据总线上。 0 =禁用监视器模式 1 =使能监视器模式 <b>注:</b> 取决于SCLOUTEN位的状态， SCL输出被强制为高，阻止模块控制I <sup>2</sup> C时钟线
[8]	<b>SCLOUTEN</b> <b>SCL输出使能位</b> 该位使能监视器SCL被拉低。监视器会拉低SCL直到有时间响应I <sup>2</sup> C中断 0 =由于开漏机制SCL输出强制为高， 1 = I <sup>2</sup> C模块扮演从机角色就像是正常操作， I <sup>2</sup> C保持时钟线为低直到有时间去清除I <sup>2</sup> C中断。
[7:6]	<b>Reserved</b> 保留

[5]	<b>PTRG</b>	<b>I<sup>2</sup>C 协议触发 (只写)</b> 当新状态出现在USCI_PROTSTS寄存器中时，如果相关中断使能位被设置，将会产生中断请求。在相关中断位被置1之后必须软件写1到该位，然后继续执行I <sup>2</sup> C协议功能直到STOP激活或是PROTEN被禁用。 0 = I <sup>2</sup> C's 钳位禁用，并且I <sup>2</sup> C协议功能继续执行 1 = I <sup>2</sup> C's 钳位激活
[4]	<b>ADDR10EN</b>	<b>位地址功能使能位</b> 0 = 禁用10位地址匹配功能 1 = 使能10位地址匹配功能
[3]	<b>STA</b>	<b>I<sup>2</sup>C 起始控制位</b> 设置 STA 为 1，进入主机模式，如果总线处于空闲状态，I <sup>2</sup> C 硬件会送出 START 或 重复 START 条件到总线
[2]	<b>STO</b>	<b>I<sup>2</sup>C 停止控制位</b> 在主机模式，设置 STO 来传送一个 STOP 条件到总线，然后 I <sup>2</sup> C 硬件将会检查总线状况，如果检测到一个 STOP 状况，这个标志会被硬件自动清除。在从机模式下，当总线错误 (USCI_PROTSTS.ERRIF = 1) 的时候，设置STO复位I <sup>2</sup> C硬件来进入“无地址”从机模式
[1]	<b>AA</b>	<b>应答控制位</b> 当 AA=1 先于地址或数据接收，在SCL线上的应答时钟脉冲期间将返回一个应答(SDA上为低电平)，有两种情况：1.) 从机正在应答主机发送的地址。2.) 接收设备正在应答发送设备发送的数据。当 AA = 0 先于地址或数据接收，则在SCL线上的应答时钟脉冲期间将返回一个非应答（SDA上为高电平）。
[0]	<b>GCFUNC</b>	<b>广播呼叫功能</b> 0 = 广播呼叫功能禁用 1 = 广播呼叫功能使能 注意：当设置了ADDR10EN (USCI_PROTCTL [4]) 时，不要设置该位。

USCI PROTEN USCI 协议中断使能寄存器- I<sup>2</sup>C

寄存器	偏移地址	R/W	描述	复位值
USCI_PROTEN	USCIIn_BA+0x60	R/W	USCI协议中断使能寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved	ACKIEN	ERRIEN	ARBLOIEN	NACKIEN	STORIEN	STARIENT	TOIEN

位	描述	
[31:7]	Reserved	保留
[6]	ACKIEN	<b>应答中断使能控制</b> 如果应答被主机检测到，该位是使能产生应答中断 0 =禁用应答中断 1 =使能应答中断
[5]	ERRIEN	<b>错误中断使能控制</b> 如果一个I <sup>2</sup> C错误条件被检测到(由 ERR (USCI_PROTSTS [16])指示)，该位是使能产生错误中断 0 =禁用错误中断 1 =使能错误中断
[4]	ARBLOIEN	<b>仲裁丢失中断使能控制</b> 如果仲裁丢失事件被检测到，该位是使能产生仲裁丢失中断 0 =禁用仲裁丢失中断 1 =使能仲裁丢失中断
[3]	NACKIEN	<b>非应答中断使能控制</b> 如果非应答被主机检测到，该位是使能产生非应答中断 0 =禁用非应答中断 1 =使能非应答中断
[2]	STORIEN	<b>停止条件接收中断使能控制</b> 如果停止条件被检测到，该位是使能产生停止条件中断 0 =禁用停止条件中断 1 =使能停止条件中断
[1]	STARIENT	起始条件接收中断使能控制

		如果起始条件被检测到，该位是使能产生起始条件中断 0 =禁用起始条件中断 1 =使能起始条件中断
[0]	<b>TOIEN</b>	<b>超时中断使能控制</b> 在I <sup>2</sup> C协议下，该位是使能产生超时事件中断 0 =禁用超时中断 1 =使能超时中断

USCI\_PROTSTS USCI 协议状态寄存器 – I<sup>2</sup>C

寄存器	偏移地址	R/W	描述	复位值
<b>USCI_PROTSTS</b>	USCIIn_BA+0x64	R/W	USCI协议状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved				ERRARBLO	BUSHANG	WRSTSWK	WKAKDONE
15	14	13	12	11	10	9	8
<b>SLAREAD</b>	<b>SLASEL</b>	<b>ACKIF</b>	<b>ERRIF</b>	<b>ARBLOIF</b>	<b>NACKIF</b>	<b>STORIF</b>	<b>STARIF</b>
7	6	5	4	3	2	1	0
<b>Reserved</b>	<b>ONBUSY</b>	<b>TOIF</b>	Reserved				

位	描述
[31:20]	<b>Reserved</b> 保留
[19]	<b>ERRARBLO</b> 仲裁丢失 错误 该位表示由于不能被输入处理器过滤的噪声引起总线仲裁丢失。当ERRARBLO被置位时I <sup>2</sup> C会发送起始条件。因此从机模式下无需关心。 0 = 总线传输状态正常 1 = 总线传输仲裁丢失错误 <b>注:</b> 该位没有中断信号，并且当START条件出现时会被硬件自动清除
[18]	<b>BUSHANG</b> 总线挂起 该位指示总线挂起状态，当SCL保持高时4位计数器会计数并且参考源是f <sub>SAMP_CLK</sub> .挂起计数器会计数到溢出并且在SDA位低时该位被置位。计数器由SCL下降沿信号复位。 0 = 总线传输正常 1 = 总线传输挂起 <b>注:</b> 该位没有中断信号，并且当START条件出现时会被硬件自动清除
[17]	<b>WRSTSWK</b> 地址帧中的读/写状态位 0 = 写命令被记录到地址匹配唤醒帧中 1 = 读命令被记录到地址匹配唤醒帧中
[16]	<b>WKAKDONE</b> 唤醒地址帧应答位完成 0 = 地址匹配帧ACK位阶段没有完成 1 = 掉电状况下地址匹配帧ACK位阶段已完成 <b>注:</b> 当WKUPIF被置位时该位不会释放
[15]	<b>SLAREAD</b> 从机读请求状态 该位表示从机读请求被检测到 0 = 没有检测到从机R/W位是1 1 = 检测到从机R/W位是1

		<b>注:</b> 该位没有中断信号, 它会被硬件自动清除
[14]	<b>SLASEL</b>	<p><b>从机选择状态</b>          该位指示设备被选作从机          0 =设备没有被选作从机          1 =设备被选作从机  <b>注:</b>该位没有中断信号, 它会被硬件自动清除</p>
[13]	<b>ACKIF</b>	<p><b>应答接收中断标志</b>          主机模式下该位表示应答位已被接收。如果USCI_PROTCTL.ACKIEN = 1会产生中断          0 =应答没有被接收          1 =应答被接收          软件写1清除</p>
[12]	<b>ERRIF</b>	<p><b>错误中断标志</b>          当START或STOP条件出现在信息帧的不合理位置时该位用于指示发生总线错误。不合理位置比如在地址字节、数据字节或应答位这些串行传输。如果USCI_PROTCTL.ERRIEN = 1会产生中断          0 = I<sup>2</sup>C错误没有被检测到          1 = I<sup>2</sup>C 错误被检测到          软件写1清除  <b>注:</b>从机模式该位被置位时, 用户必须写1到STO寄存器是设备进入“无地址”从机模式</p>
[11]	<b>ARBLOIF</b>	<p><b>仲裁丢失中断标志</b>          该位表示发送仲裁丢失。如果USCI_PROTCTL.ARBLIOIEN = 1会产生中断          0 =没有发生仲裁丢失          1 =仲裁丢失          软件写1清除</p>
[10]	<b>NACKIF</b>	<p><b>非应答接收中断标志</b>          主机模式下该位表示非应答位已被接收。如果USCI_PROTCTL.NACKIEN = 1会产生中断          0 =非应答没有被接收          1 =非应答被接收          软件写1清除</p>
[9]	<b>STORIF</b>	<p><b>停止条件接收中断标志</b>          该位表示停止条件在I<sup>2</sup>C总线上已被检测到。如果USCI_PROTCTL.STORIEN = 1会产生中断          0 =停止条件没有被检测到          1 =停止条件被检测到          软件写1清除  <b>注:</b>当显示从机RX模式下被置位</p>
[8]	<b>STARIF</b>	<p><b>起始条件接收中断标志</b>          该位表示主机模式下起始条件或是重复起始条件在I<sup>2</sup>C总线上已被检测到。然而在从机模式下表示重复起始条件被检测到。          如果USCI_PROTCTL.STARIEN = 1会产生中断          0 =起始条件还没有被检测到          1 =停止条件已被检测到          软件写1清除</p>

[7]	<b>Reserved</b>	保留
[6]	<b>ONBUSY</b>	<b>总线忙</b> 表明总线上正在进行数据传输。当START条件被检测到时被硬件置位。当STOP被检测到时被硬件清除 0 = 总线空闲 (SCLK 和 SDA 都为高). 1 = 总线忙
[5]	<b>TOIF</b>	<b>超时中断标志</b> 0 = 没有发生超时中断 1 = 没有发生超时中断 <b>注:</b> 软件写1清除
[0]	<b>Reserved</b>	保留

USCI\_TMCTL USCI 时序配置控制寄存器

寄存器	偏移地址	R/W	描述	复位值
USCI_TMCTL	USCIIn_BA+0x8C	R/W	I <sup>2</sup> C 时序配置控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							HTCTL
23	22	21	20	19	18	17	16
HTCTL							
15	14	13	12	11	10	9	8
Reserved							STCTL
7	6	5	4	3	2	1	0
STCTL							

位	描述	
[31:25]	<b>Reserved</b>	保留
[24:16]	<b>HTCTL</b>	<b>保持时间配置控制寄存器</b> 该域是发送模式下用来在SCL下降沿和SDA边沿之间产生延时时序。 延时保持时间外设时钟数= HTCTL x f <sub>PCLK</sub> .
[15:9]	<b>Reserved</b>	保留
[8:0]	<b>STCTL</b>	<b>建立时间配置控制寄存器</b> 该域是发送模式下用来在SDA边沿和SCL上升沿之间产生延时时序。 延时建立时间外设时钟数= STCTL x f <sub>PCLK</sub> .

## 6.27 CAN 接口

### 6.27.1 概述

C\_CAN由CAN内核，报文RAM，报文处理器，控制寄存器和模块接口组成。CAN内核按照CAN协议2.0版本A部分和B部分规范执行通信。位速率最高可达 1Mbit/s。为与物理层相连，还需另外外接硬件收发器。

在CAN网络中，各个报文对象是可以独立配置的。报文对象和用于在接收时进行报文过滤的标识符掩码都存储在报文RAM中。所有与报文处理相关的功能都在报文处理器中执行。这些功能包括接收过滤、CAN内核与报文RAM之间的报文传输、处理传送请求以及模块中断的产生。

C-CAN的寄存器组可以通过模块接口被软件直接访问。这些寄存器用来控制/配置CAN内核和报文处理器，以及访问报文RAM。

### 6.27.2 特性

- 支持CAN协议 2.0版本 A和 B部分
- 位速率最高可达1 Mbit/s
- 32个报文对象
- 每个报文对象都有自己的标示符掩码
- 可编程FIFO模式（链接报文对象）
- 中断可屏蔽
- 禁用时间触发CAN应用下的自动重传模式
- 支持用于自检测的可编程环回模式
- 连接到AMBA APB总线上的16-位模块接口
- 支持唤醒功能

### 6.27.3 框图

C\_CAN与AMBA APB总线相连，如图 6.27-1所示

- CAN 内核
  - 包括CAN协议控制器和用于报文串/并数据转换的RX/TX移位寄存器
- 报文 RAM
  - 用于保存报文对象和标识符掩码
- 寄存器组
  - 包含所有控制和配置C\_CAN的寄存器
- 报文处理器
  - 用于控制CAN内核的RX/TX移位寄存器和报文RAM间的数据传输的状态机，以及按照控制和配置寄存器中设定产生中断
- 模块接口
  - C\_CAN与来自CPU的AMBA APB16-位总线相接

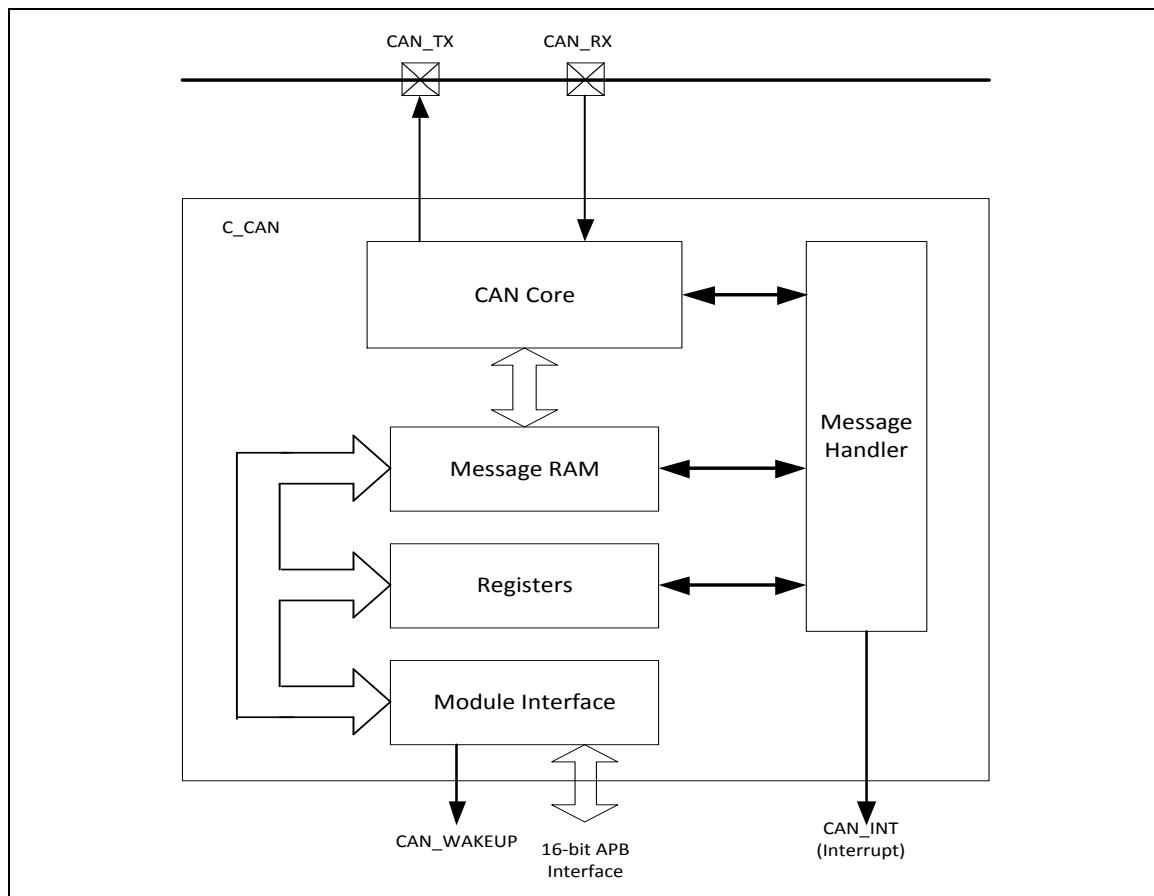


图 6.27-1 CAN 外设框图

#### 6.27.4 基本配置

##### 6.27.4.1 CAN0 基本配置

- 时钟源配置
  - 使能 CAN时钟的控制位是 (CAN\_EN (APBCLK0[24]))
- 复位配置
  - 复位 CAN控制器的控制位是 (CAN\_RST (IPRSTC2[24])).
- 引脚配置

组	引脚名	GPIO	MFP
CAN0	CAN0_RXD	PD.10, PE.15	MFP4
		PA.13	MFP6
		PB.10	MFP8
		PA.4, PC.4	MFP10
	CAN0_TXD	PD.11, PE.14	MFP4
		PA.12	MFP6

	PB.11	MFP8
	PA.5, PC.5	MFP10

#### 6.27.4.2 CAN1 基本配置

- 时钟源配置
  - 使能 CAN 时钟的控制位是 (CAN\_EN (APBCLK0[25])).
- 复位配置
  - 复位 CAN 控制器的控制位是 (CAN\_RST (IPRSTC2[25])).
- 引脚配置

组	引脚名	GPIO	MFP
CAN1	CAN1_RXD	PB.6, PD.12	MFP5
		PG.1	MFP7
		PC.9, PE.6	MFP9
		PC.2	MFP10
	CAN1_TXD	PB.7, PC.13	MFP5
		PG.0	MFP7
		PC.10, PE.7	MFP9
		PC.3	MFP10

#### 6.27.5 功能描述

##### 6.27.5.1 软件初始化

软件初始化可以通过软件或硬件复位，或者通过进入 Bus\_off 状态置位 CAN 控制寄存器的 Init 位 (CAN\_CON[0]) 来实现。

当 Init 位被置位后，所有 CAN 总线上的报文传输都会停止，CAN\_TX 输出管脚上的状态为隐性（高电平）。错误管理逻辑 (EML) 计数器将保持不变。设置 Init 位不会改变任何配置寄存器。

为了初始化 CAN 控制器，软件必须设置位定时寄存器和每个报文对象。如果不需要用到某个报文对象，那么必须清除相应的 MsgVal 位 (CAN\_IFn\_ARB2[15])，否则，整个报文对象都需要进行初始化。

当 CAN 控制寄存器的 Init 和 CCE 位 (CAN\_CON[6]) 都被置位时，用于配置位定时的位定时寄存器和波特率预分频扩展寄存器的访问才会被使能。

复位 Init 位（只能通过软件方式）完成软件初始化。接着位流处理器 (BSP)（参看 6.27.7.15：配置位定时）在等待 11 个连续隐性位（总线空闲）的位序列后先将自身与 CAN 总线上数据实现同步，然后再参与总线活动和开始报文传输。

报文对象的初始化是独立于 Init，可以在忙碌时完成。但在 BSP 开始传输报文之前，所有报文对象的标识符必须先配置成指定值或设成无效。

在正常工作期间改变一个报文对象的配置时，软件必须先复位相应的 MsgVal 位，然后再改变配置。当配置完成后，需再次设置 MsgVal 位。

### 6.27.5.2 CAN 报文传输

一旦C\_CAN被初始化以及Init bit (CAN\_CON[0])位被重置为0, C\_CAN内核就会将自身与CAN总线同步，并开始报文传输。

如果接收到的报文通过了报文处理器的接收过滤，将会被存储在相应的报文对象中。保存在报文对象中的报文包括所有的仲裁位，DLC(CAN\_IFn\_MCON[3:0])和8个字节数据(CAN\_IFn\_DAT\_A1/2; CAN\_IFn\_DAT\_B1/2)。如果使用了标识符掩码，那些被掩码为“无关”位的仲裁位会在报文对象中被覆盖。

软件可以通过接口寄存器在任何时间去读或写每条报文。如果同时访问，报文处理器将保证数据的一致性。

发送的报文由应用软件进行更新。如果一个报文永久性的存在某个报文对象中（仲裁位和控制位在配置时被设定），那么仅有数据字节会被更新，在置位TxRqst位(CAN\_IFn\_MCON[8])和NewDat(CAN\_IFn\_MCON[15])位会后就会开始传送。如果几条传输报文被指派给同一个报文对象(当报文对象的数量不够用时)，那么在发送这些报文前必须对整个报文对象进行配置。

任何数量的报文对象都可以在同一时间请求传送。报文对象会按照它们的内部优先级进行依次传送。报文可以随时更新或设置为无效，甚至在发送请求还在等待阶段也可进行。如果一条报文在其未发送前被更新，则旧的数据将会被丢弃。

根据报文对象的配置，在接收到匹配标识符的遥控帧后，会自动产生报文发送请求

### 6.27.5.3 禁用自动重传

依照CAN协议规范（见ISO11898, 6.3.3恢复管理），C\_CAN 为那些在传送过程中丢失仲裁或被错误干扰的帧，提供自动重传机制。在传送完全成功之前，帧传送服务是不能证实给用户。这也就意味着，默认情况下，自动重传是使能的。在C\_CAN工作在时间触发CAN (TTCAN, 见ISO11898-1) 环境时，可以禁用自动重传功能。

通过设置CAN控制器中的禁用自动重传 (DAR(CAN\_CON[5])) 位为1来禁用自动重传模式。在这种操作模式下，用户必须考虑报文缓存控制寄存器中 TxRqst(CAN\_IFn\_MCON[8]) 和 NewDat(CAN\_IFn\_MCON[15])位不同设置时的状况：

- 当传送开始时，各个报文缓存的TxRqst位被清除，而NewDat位一直置位
- 当传送完全成功时，NewDat位被清除。
- 当传送失败（丢失仲裁或发生错误），NewDat位保持置位
- 为了重新开始传送，软件必须再将TxRqst位置位

## 6.27.6 测试模式

设定CAN控制寄存器的Test(CAN\_CON[7])位进入测试模式。在测试模式下，测试寄存器的Tx1 (CAN\_TEST[6]), Tx0 (CAN\_TEST[5]), LBack (CAN\_TEST[4]), Silent (CAN\_TEST[3]) 和 Basic (CAN\_TEST[2])位可写。Rx位监视CAN\_RX管脚的状态，因此该位只读。当Test位被清除时，所有的测试寄存器功能被禁用

### 6.27.6.1 静默模式

通过设置测试寄存器中的Silent位(CAN\_TEST[3])为1，CAN内核可被设为Silent模式。在Silent模式中，

C\_CAN能够接收有效的数据帧和有效的遥控帧，但是它在CAN总线上仅发送隐性位，而且它不能启动发送。如果CAN内核被要求发送一个显性位（ACK位，错误帧），那么该显性位将在内部自动改道以便CAN内核检测到该显性位，而CAN总线仍然保持在隐性状态。因为可以在传送显性位时不会影响到CAN总线，所以Silent模式可以用来分析CAN总线的通讯状况。下图为静默模式下，CAN\_TX和CAN\_RX与CAN内核的连接信号

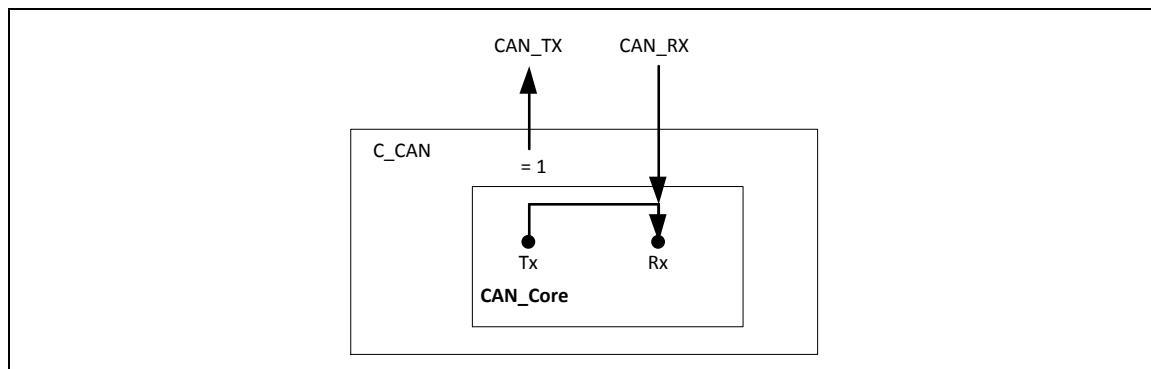


图 6.27-2 CAN内核静默模式

#### 6.27.6.2 回环模式

通过设定测试寄存器的LBack位(CAN\_TEST[4])为1，CAN内核可被设为回环模式。在回环模式下，CAN内核把自己发送的报文作为接收到的报文对待，并将它们保存在接收缓存中（如果它们通过了接收过滤）。图 6.27-3为回环模式下，CAN\_TX和CAN\_RX与CAN内核的连接信号。

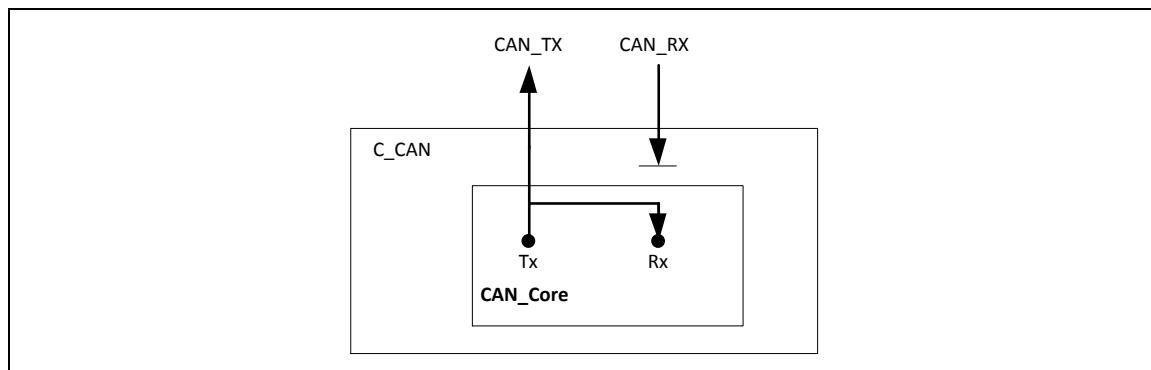


图 6.27-3 CAN 内核回环模式

该模式用于自检。为了不受外部的干扰，在回环模式下，CAN内核忽略应答错误（数据/远程帧应答槽内采样到隐性位）。在该模式下，CAN内核的Tx输出通过内部直接反馈到它的Rx输入上。CAN\_RX输入管脚的真实值则被CAN内核忽略。发送的报文信号仍可通过 CAN\_TX管脚进行监测。

#### 6.27.6.3 回环模式和静默模式的组合

通过设定LBack(CAN\_TEST[4])和Silent(CAN\_TEST[3])位同时为1，还可以将回环模式和静默模式结合一起使用。该模式可用于“热自测（Hot Selftest）”，即C\_CAN可以在不影响已与CAN\_TX 和CAN\_RX 管脚相连的CAN系统的情况下进行测试。在该模式下，CAN\_RX管脚与CAN内核断开，CAN\_TX管脚保持隐性电平。图 6.27-4为回环模式和静默模式整合下，CAN\_TX 和CAN\_RX与CAN内核的连接信号

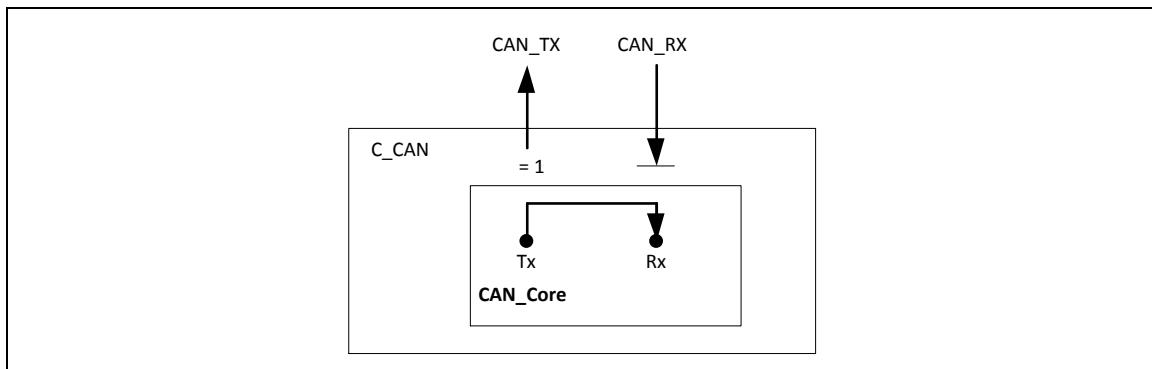


图 6.27-4 CAN内核回环模式与静默模式组合

#### 6.27.6.4 基本模式

通过设定测试寄存器的Basic位(CAN\_TEST[2])为1可以设定CAN内核工作在Basic模式。该模式下C\_CAN工作时没有报文RAM。

**IF1** 寄存器组用作发送缓存。**IF1** 寄存器组中内容通过设定**IF1**命令请求寄存器中的Busy位(CAN\_IFn\_CREQ[15])为1来请求发送。当Busy位置位时，**IF1** 寄存器组锁定。Busy位指示传送正在挂起。

一旦CAN总线空闲，**IF1** 寄存器组会被载入到CAN内核的移位寄存器，然后传送开始。当传输完成时，Busy位复位，并释放锁定的**IF1**寄存器组。

当**IF1**寄存器组锁定时，挂起的发送可以被随时中止，方法为复位**IF1**的命令请求寄存器的Busy位。如果软件已经复位了Busy位，那么为防止仲裁丢失或错误出现的重传是被禁用的。

**IF2**寄存器组用作接收缓存。收到报文后，移位寄存器的内容不经过任何接收过滤就存储到**IF2**寄存器组。

此外，在报文传输的过程中，移位寄存器的真实内容能够被监视。每次写**IF2** 命令请求寄存器中的Busy位为1，读报文对象就会被初始化，移位寄存器的内容则被保存在**IF2**寄存器组。

在基本模式下，所有与报文对象相关的控制和状态位以及**IFn**掩码寄存器的控制位都不能进行赋值。命令请求寄存器的报文编号也不能赋值。**IF2** 报文控制寄存器的 NewDat(CAN\_IFn\_MCON[15]) 和 MsgLst(CAN\_IFn\_MCON[14])位保留其功能。DLC3-0指示接收到的DLC(CAN\_IFn\_MCON[[3:0]])，其他控制位读取值皆为‘0’

#### 6.27.6.5 软件控制 CAN\_TX 管脚

CAN发送管脚CAN\_TX上有四种输出功能可用。除了缺省功能（串行数据输出），CAN发送管脚还能驱动输出CAN采样点信号，用于监视CAN内核的位定时，而且CAN发送管脚还能持续驱动输出显性或隐性电平。后两种功能，结合可读的CAN接收管脚CAN\_RX，能够用于检测CAN总线的物理层。

CAN\_TX管脚的输出模式通过设定CAN测试寄存器的Tx1 (CAN\_TEST[6])和Tx0(CAN\_TEST[5])位进行选择。

CAN\_TX管脚参与了三种测试模式所有的CAN协议功能。当CAN进行报文传输或者选择任一测试模式（回环模式，静默模式,或基本模式）时，CAN\_TX 必须使用它的默认功能

#### 6.27.7 CAN 通信

### 6.27.7.1 管理报文对象

报文RAM中的报文对象配置（除CAN控制寄存器的MsgVal, NewDat, IntPnd, 和TxRqst位外）不会受芯片复位影响。在应用软件清除Init位之前，所有的报文对象必须被应用软件初始化或是设置为“无效”（**MsgVal=0**），而且位时序必须在应用软件清Init位(**CAN\_CON[0]**)之前被配置好。

通过配置两个接口寄存器中一个接口寄存器的屏蔽、仲裁、控制和数据域为期望值后，报文对象的配置完成。通过写相应的IFn命令请求寄存器，IFn报文缓存寄存器组会被载入到报文RAM中指定地址的报文对象中。

CAN控制寄存器的Init位被清除后，CAN内核的CAN协议控制状态机和报文处理器状态机控制C\_CAN的内部数据流。收到的报文通过接收过滤后被保存在报文RAM中。挂起发送请求的报文被载入到CAN内核的移位寄存器，并通过CAN总线进行发送。

应用软件通过IFn接口寄存器组读取收到的报文以及更新发送的报文。依照配置，应用软件会被特定的CAN报文和CAN错误事件打断。

### 6.27.7.2 报文处理器状态机

报文处理器控制CAN内核的Rx/Tx移位寄存器、报文RAM和IFn寄存器组之间的数据传输。

报文处理器 FSM控制如下功能：

- 从IFn寄存器传送数据到报文RAM
- 从报文RAM传送数据到IFn寄存器
- 从移位寄存器传送数据到报文RAM
- 从报文RAM传送数据到移位寄存器
- 从移位寄存器传送数据到接收过滤单元
- 扫描报文RAM寻找匹配报文对象
- 处理TxRqst标志位
- 处理中断

### 6.27.7.3 报文RAM的数据传输

当应用软件对IFn寄存器组和报文RAM间的数据传输初始化后，报文处理器设置各自命令请求寄存器的Busy位（**CAN\_IFn\_CREQ[15]**）为‘1’。在传输完成后，Busy位会被再次清除（见图 6.27-5）。

命令掩码寄存器可以指定是一个完整的报文对象还是报文对象的部分内容将被传输。因为报文RAM的结构，写操作不能仅仅针对报文对象的单独某位/字节，而是必须写入完整的报文对象到报文RAM。因此，从IFn寄存器到报文RAM的数据传输需要一个读-修改-写的周期。首先，将报文对象中不变的部分从报文RAM读出，然后将报文缓存寄存器的完整报文内容写入到报文对象。

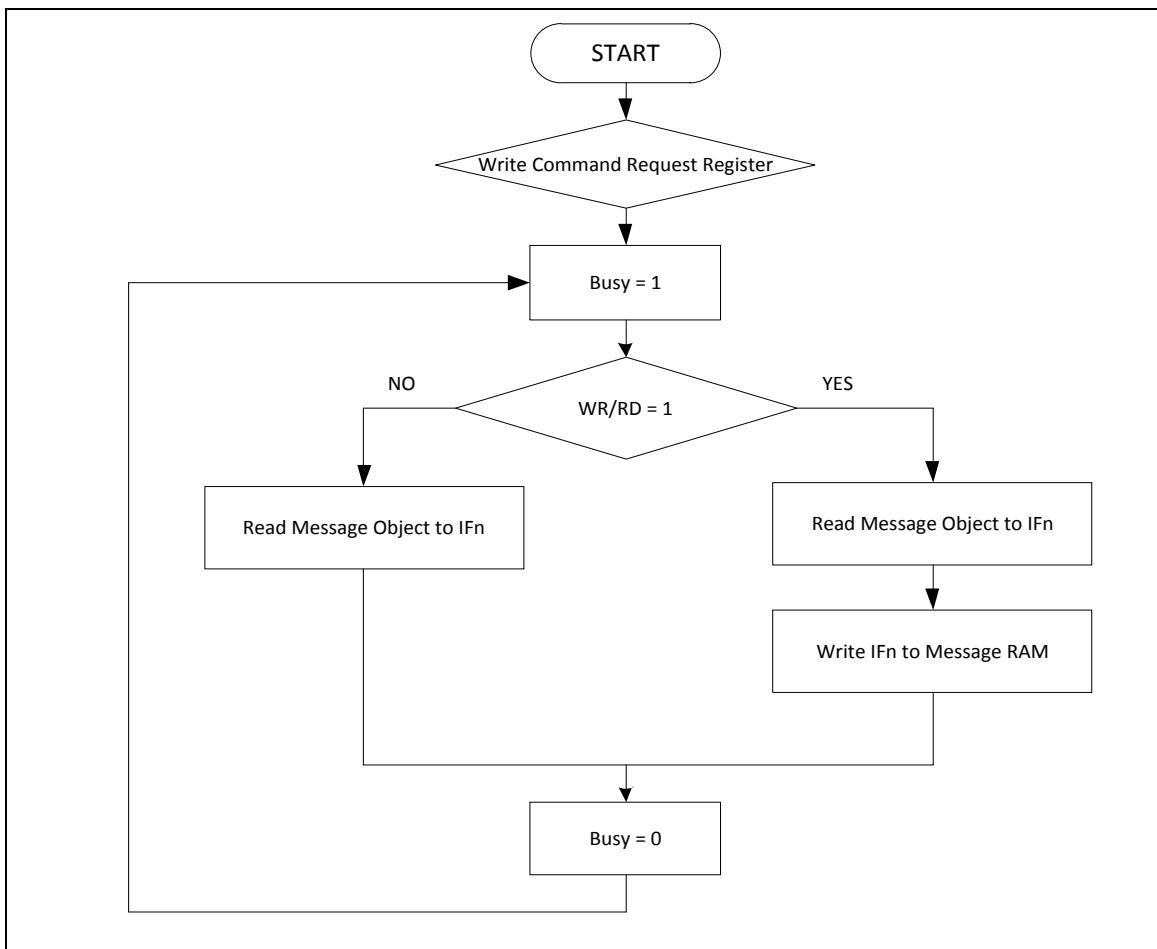


图 6.27-5 IFn寄存器和报文RAM间的数据传输

在写入一个报文对象部分内容后，没被命令掩码寄存器选中的报文缓存寄存器将设置选中报文对象的实际内容。

在读取一个报文对象部分内容后，没被命令掩码寄存器选中的报文缓存寄存器将保持不变

#### 6.27.7.4 报文传输

如果CAN内核单元的移位寄存器准备载入，但在IFn寄存器组和报文RAM之间没有数据传输，内核将重新评估MsgVal 位(CAN\_IFn\_ARB2[15]) 和TxRqst位(CAN\_TXREQ1/2)。具有最高优先级发送请求的有效报文对象将被报文处理器载入到移位寄存器，开始发送。报文对象的NewDat (CAN\_IFn\_MCON[15]) 位被复位。

成功传送后或者自传送开始后没有新的数据写入报文对象(NewDat = '0')，报文控制寄存器TxRqst 位(CAN\_IFn\_MCON[8])将被复位。如果报文控制寄存器TxIE位(CAN\_IFn\_MCON[11])被置位，中断标识符寄存器的IntPnd 位(CAN\_IFn\_MCON[13])在传送成功后将被置位。如果C\_CAN仲裁失败或者在传送过程中有错误发生，报文将会在CAN总线空闲的时候进行重传。同时，如果有更高优先级的报文传送请求，则报文将会按照报文的优先级顺序进行传送。

#### 6.27.7.5 收到报文的接收过滤

当一个输入报文的仲裁和控制域(Identifier + IDE + RTR + DLC)完全被移位到CAN内核的Rx/Tx移位寄存器时，报文处理器FSM开始扫描报文RAM，寻找匹配的有效报文对象。

扫描报文RAM寻找匹配的报文对象，需要将CAN内核移位寄存器的仲裁位载入到接收过滤单元。步骤为：先载入报文对象1的仲裁和掩码域（包括MsgVal (CAN\_IFn\_ARB2[15]), UMask (CAN\_IFn\_MCON[12]), NewDat (CAN\_IFn\_MCON[15]), 和EoB (CAN\_IFn\_MCON[7]) 到接收过滤单元，再和移位寄存器的仲裁位进行比较。该步骤对接下来的每一个报文对象重复进行，直到匹配的报文对象出现或者到达报文RAM的末端。

如果匹配发生，扫描停止，报文处理器FSM将根据接收帧的类型（数据帧或远程帧）进行处理。

### 接收数据帧

报文处理器FSM将CAN内核移位寄存器中报文保存到报文RAM中的各个报文对象。不仅数据字节，所有仲裁位和数据长度码都被保存到相应的报文对象中。这是为了将数据字节与标识符继续保持关联性

置位NewDat 位(CAN\_IFn\_MCON[15])用于指示新数据（还没有被软件看到的）已经收到了。当报文对象被读取后，应用软件必须复位NewDat 位(CAN\_IFn\_MCON[15])。如果在接收的时候，NewDat 位已经被置位，MsgLst (CAN\_IFn\_MCON[14])被置位用来指示之前的数据（假定还没有被软件看到）已丢失。如果RxIE位 (CAN\_IFn\_MCON[10])被置位，IntPnd 位(CAN\_IFn\_MCON[13])也置位将使中断寄存器指向该报文对象

当请求的数据帧刚刚收到时，报文对象的TxRqst(CAN\_IFn\_MCON[8])位会被复位用来阻止传送远程帧

### 接收远程帧

当收到一个远程帧时，必须考虑匹配的报文对象的三种配置

1) Dir (CAN\_IFn\_ARB2[13]) = '1' (方向 = 发送), RmtEn (CAN\_IFn\_MCON[9]) = '1', UMask (CAN\_IFn\_MCON[12]) = '1'或'0'

当收到匹配的远程帧时，报文对象的TxRqst位被置位。报文对象的其他部分保持不变。

2) Dir = '1' (方向=发送), RmtEn = '0', UMask ='0'

当收到匹配的远程帧时，报文对象的TxRqst位保持不变，远程帧被忽略。

3) Dir = '1' (方向=发送), RmtEn = '0', UMask ='1'

当收到匹配的远程帧时，报文对象的TxRqst位被复位。移位寄存器中的仲裁和控制域(Identifier + IDE + RTR + DLC)被保存到报文RAM中的报文对象，报文对象的NewDat (CAN\_IFn\_MCON[15])位被置位。报文对象的数据域保持不变。收到的远程帧处理方式与数据帧类似。

#### 6.27.7.6 收/发优先级

报文对象接收/发送的优先级与报文号相关。报文对象1拥有最高优先级，报文对象32拥有最低优先级。如果有一个以上的发送请求挂起时，它们将根据相应报文对象的优先级来进行服务

### 6.27.7.7 配置传送对象

表 6.27-1说明了怎样初始化一个传送对象

Ms	Arb	Data	Mask	EoB	Dir	NewDat	MsgLst	RxE	TxE	IntPnd	RmtEn	TxRqst
1	appl.	appl.	appl.	1	1	0	0	0	appl.	0	appl.	0

表 6.27-1 初始化传送对象

注: appl. = 应用软件.

仲裁寄存器(ID28-0 (CAN\_IFn\_ARB1/2) 和Xtd 位(CAN\_IFn\_ARB2[14]))由应用软件设置。它们定义标识符和发出的报文类型。如果11-位标识符(标准帧)被使用, 它将被编程到ID28 - ID18。ID17 - ID0可以被忽略。

如果TxE(CAN\_IFn\_MCON[11])位被置位, 在报文对象成功发送之后, IntPnd(CAN\_IFn\_MCON[13])位将被置位。

如果RmtEn(CAN\_IFn\_MCON[9])位被置位, 收到匹配的远程帧将置TxRqst(CAN\_IFn\_MCON[8])位, 而且该远程帧会自动被一个数据帧应答。

数据寄存器的值(DLC3-0 (CAN\_IFn\_MCON[3:0]), Data0-7)由应用软件提供。在数据有效之前, TxRqst 和RmtEn可能不会被置位。

(UMask (CAN\_IFn\_MCON[12]) = '1')时,掩码寄存器(Msk28-0, UMask, MXtd, 和 MDir 位)可以用于允许有相似标识符的远程帧组设置TxRqst位。但Dir (CAN\_IFn\_ARB2[13])位不能被屏蔽

### 6.27.7.8 更新一个传送对象

不必复位MsgVal (CAN\_IFn\_ARB2[15])和TxRqst(CAN\_IFn\_MCON[8]), 软件就可以通过IFn接口寄存器随时更新发送对象的数据字节。

即使只更新部分数据字节, 在内容传输给报文对象之前, IFn数据A寄存器或B寄存器中的所有4个字节数据都必须有效。应用软件必须写所有4个字节到IFn数据寄存器或者在软件写新的数据字节前报文对象已被传输到IFn数据寄存器。

仅当(8个)数据字节更新时, 先写0x0087到命令掩码寄存器, 然后再写报文对象的编号到命令请求寄存器, 同时更新数据字节及设置TxRqst

当数据更新时, 为防止在传输的最后阶段TxRqst复位, 必须同时置位NewDat (CAN\_IFn\_MCON[15])和TxRqst。

当NewDat和TxRqst一起被置位时, NewDat将会在新的传送开始时立即被复位

### 6.27.7.9 配置一个接收对象

表 6.27-2说明如何初始化一个接收对象

MsgVal	Arb	Data	Mask	EoB	Dir	NewDat	MsgLst	RxE	TxE	IntPnd	RmtEn	TxRqst
1	appl.	appl.	appl.	1	0	0	0	appl.	0	0	0	0

表 6.27-2 初始化接收对象

仲裁寄存器的值(ID28-0 (CAN\_IFn\_ARB1/2) 和 Xtd 位 (CAN\_IFn\_ARB2[14]))由应用软件设置。它们定义标识符和接收报文的类型。如果11-位标识符（“标准帧”）被使用，它将会编程到ID28 - ID18。ID17 - ID0可以被忽略。当带有11-位标识符的数据帧被收到时，ID17 - ID0将被置0。

如果 RxE(CAN\_IFn\_MCON[10])位置位，当接收到数据帧并被保存到报文对象时，IntPnd (CAN\_IFn\_MCON[13])位将被置位。

数据长度码 ((DLC3-0 (CAN\_IFn\_MCON[3:0]))由应用软件设置。当报文处理器保存一个数据帧到报文对象时，它将会保存收到的数据长度码和8数据字节。如果数据长度码少于8，则报文对象余下的字节将被未指定的值覆盖。

(UMask (CAN\_IFn\_MCON[12]) = '1')时，掩码寄存器(Msk28-0, UMask, MXtd, 和MDir)可以被用于允许有相似标识符的数据帧组被接收。在典型的应用中，Dir (CAN\_IFn\_ARB2[13])位不能被屏蔽。

#### 6.27.7.10 处理接收报文

应用软件可以通过IFn接口寄存器随时读一条收到报文。报文处理器的状态机将保证数据的一致性。

通常的，软件先写0x007F到命令掩码寄存器，然后写报文对象编号到命令请求寄存器。这就会将收到的报文整个从报文RAM传输到报文缓存寄存器。并且，在报文RAM(不是报文缓存)中NewDat (CAN\_IFn\_MCON[15])和IntPnd (CAN\_IFn\_MCON[13])位将被清除。

如果报文对象使用掩码进行接收过滤，则仲裁位表示哪条匹配的报文已经收到了。NewDat的真实值（当前值）表示自上次该报文对象被读取后，是否收到新的报文。MsgLst (CAN\_IFn\_MCON[14])的真实值表示自上次该报文对象被读取后，收到的报文是否超过1条。MsgLst不会被自动复位。

利用远程帧，软件可以请求另一个CAN节点为接收报文对象提供新的数据。设定接收报文对象的TxRqst(CAN\_IFn\_MCON[8])位将引发带有接收报文对象的标识符的远程帧进行传送。该远程帧触发另一个CAN节点开始发送与之匹配的数据帧。如果在发送远程帧之前收到匹配的数据帧，TxRqst位将被自动复位。

#### 6.27.7.11 配置报文对象FIFO缓存

除EoB(CAN\_IFn\_MCON[7])位外，FIFO缓存的接收报文对象群的配置和单个接收报文对象的配置是一样的。参看章节6.5.7.9：配置接收对象

为在FIFO缓存中关联2个或更多报文对象，这些报文对象的标识符和掩码（如使用）必须设置为要匹配的值。因为报文对象的默认优先级，最低编号的报文对象将会是FIFO缓存的第一个报文对象。FIFO缓存中（除了最后的报文对象外）所有报文对象的EoB位都必须设置为0。FIFO缓存中的最后一个报文对象的EoB位需设置为1，表示模块的结束

#### 6.27.7.12 用报文对象FIFO缓存接收报文

收到的报文如果匹配FIFO缓存中的标识符会被保存到该FIFO缓存中的一个报文对象中，报文存储的顺序为从最低编号的报文对象开始。

当一条报文保存到报文对象FIFO缓存中时，该报文对象的NewDat(CAN\_IFn\_MCON[15])位被置位。当EoB(CAN\_IFn\_MCON[7])为0时通过置位NewDat，报文对象将被锁定用于之后的报文处理器的写访问，直到应用软件将NewDat位写回0。

报文群将一直保存在FIFO缓存中直到FIFO缓存的最后一个报文对象。如果之前没有一个报文对象通过写NewDat位为0释放，则FIFO缓存之后收到的报文都将被写入到该FIFO缓存的最后一个报文对象中，当然之前的报文会被覆盖。

#### 6.27.7.13 从报文对象FIFO中读数据

当应用软件通过写报文对象编号到IFn命令请求寄存器将报文对象的内容传输到IFn报文缓存寄存器时，相应的命令掩码寄存器设置为：(TxRqst/NewDat (CAN\_IFn\_CMASK[2]) = '1' 和 ClrlntPnd (CAN\_IFn\_CMASK[3]) = '1')时，NewDat (CAN\_IFn\_MCON[15]) 和 IntPnd (CAN\_IFn\_MCON[13])位复位为零。报文控制寄存器以上各个寄存器位一直反映复位前的状态。

为保证FIFO缓存的正确使用，应用软件须从FIFO缓存中最低报文编号的报文对象读起。

下图 6.27-6 表示应用软件如何处理一组和FIFO缓存相连的报文对象

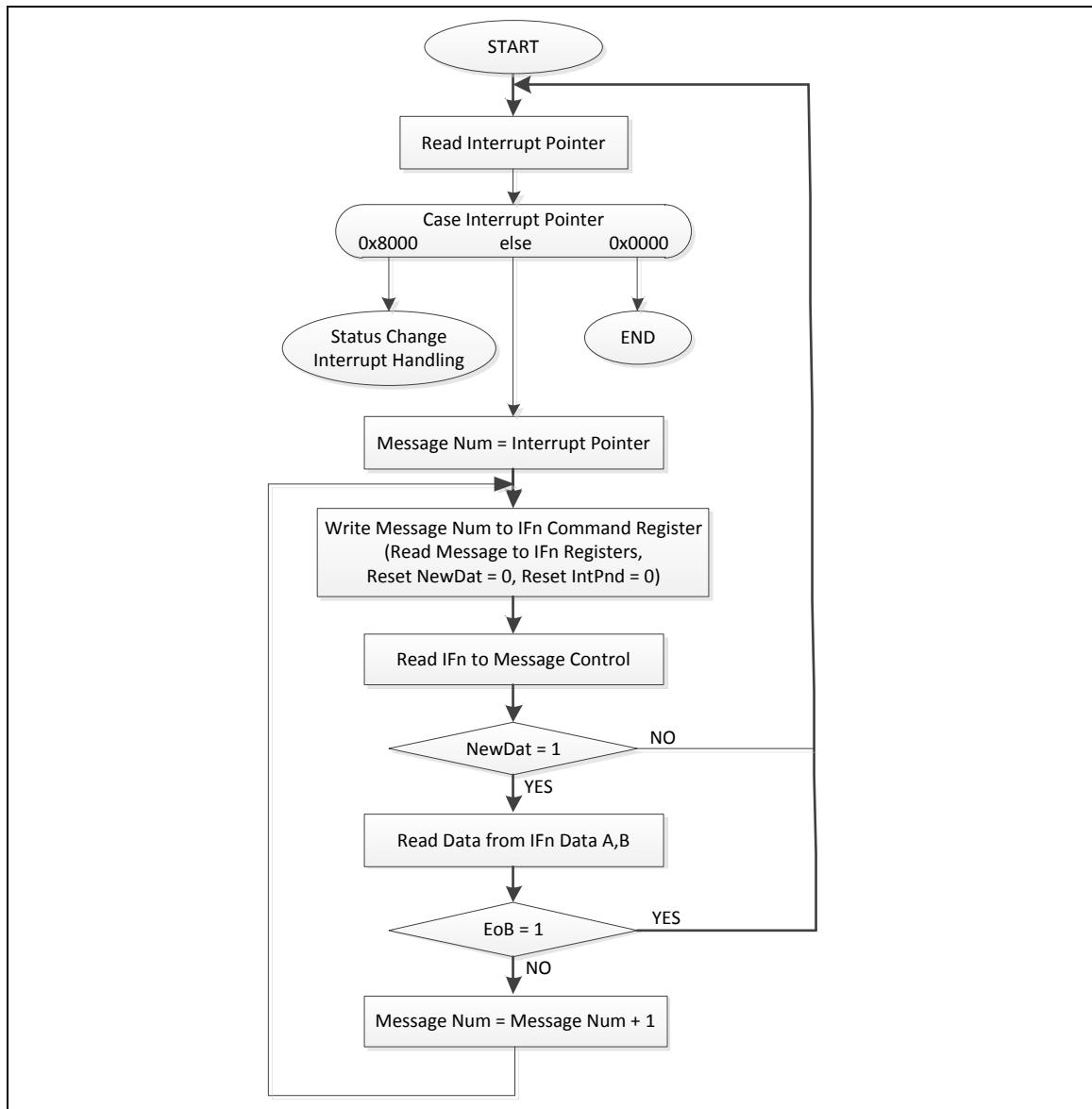


图 6.27-6 应用软件处理 FIFO 缓存

#### 6.27.7.14 中断处理

如果几个中断同时挂起，则CAN的中断寄存器将指向具有最高优先级的挂起中断，而无需理会他们的时序。一个中断会保持挂起状态直到应用软件清除它。

状态中断拥有最高优先级。在报文中断中，报文对象的中断优先级随着报文编号的增加而降低

报文中断通过清除报文对象的IntPnd(CAN\_IFn\_MCON[13])位进行清除。状态中断通过读状态寄存器清除。

中断寄存器的中断标识符IntId指示引发中断的报文对象编号。当没有中断挂起时，寄存器将保持为0。如果中断寄存器的值不为0，则有中断挂起。如果IE(CAN\_CON[1])被置位，则CAN\_INT中断信号被激活。中断保持激活直到中断寄存器归零（中断源被复位）或直到IE被复位

CAN\_IIDR的值为0x8000指示有中断正被挂起，原因是CAN内核更新了（不是必须改变）状态寄存器（错误中断或状态中断）。该中断具有最高优先级，应用软件可以更新（复位）状态位RxOk

(CAN\_STATUS[4]), TxOk (CAN\_STATUS[3]) 和 LEC (CAN\_STATUS[2:0]), 但是软件对状态寄存器的写访问不会产生或复位中断。

其他的值表示中断源是其中一个报文对象。IntId指向挂起中断中的具有最高中断优先级的报文中断。

应用软件决定改变状态寄存器是否可能导致产生中断 (EIE (CAN\_MCON[3]) 和 SIE (CAN\_MCON[2]))，以及当中断寄存器的值不等于零时中断是否需要激活 (CAN控制寄存器的IE位)。即使IE被复位，中断寄存器仍将会更新。

应用软件跟踪报文中断源有2种可能方式。一种是通过查看中断寄存器的IntId位，第二种是查询中断挂起寄存器。

中断处理服务惯例为：读触发中断源的报文并在读报文的同时复位报文对象的IntPnd (ClrlntPnd(CAN\_IFn\_CMASK[3])) 位。当IntPnd被清除，中断寄存器将指向下一个带有挂起中断的报文对象。

#### 6.27.7.15 配置位定时

CAN位定时配置的小错误不会导致通讯立即失败，但是会显著降低CAN网络的整体性能。

很多情况下，CAN位同步可以修补CAN位定时的错误配置，而这种错误有可能仅仅是偶发性发生一个错误帧。但是，在仲裁时，当两个或更多CAN节点同时试着发送一帧，那么一个错位的采样点可能会导致其中一个传送器变成被动错误状态。

如果要分析此类分散性错误，需要对CAN节点内CAN位同步以及CAN总线上CAN节点间相互作用有详细的了解。

#### 6.27.7.16 位时间和位速率

CAN支持的位速率低至1 kb/s，高至1000 kb/s。CAN网络中的每个成员都有自己的时钟发生器，通常是由石英振荡器。位时间的定时参数（例如：位速率的倒数）可以单独配置给每个CAN节点，这样尽管CAN节点的振荡器周期( $f_{osc}$ )可能不同，仍可创建一个公共的位速率。

这些振荡器的频率不是绝对的稳定，温度或电压的变化或者元件性能恶化会导致一些小的变化。只要这种变化保持在指定的振荡器容差范围 ( $d_f$ ) 内，CAN节点就能够通过重同步对不同的位速率进行补偿。

依据CAN规范，位时间分为四段：同步段，传播时间段，相位缓冲段1和相位缓冲段2（见图6.27-7）。每段由一个特定的、可编程的时间片组成（见表 6.27-3）。时间片的长度( $t_q$ )，是位时间的基本时间单元，由CAN控制器的APB时钟 $f_{APB}$ 和位定时寄存器 (CAN\_BTR) 的BRP位定义：  

$$(CAN_BTIM[5:0]) : t_q = BRP / f_{APB}$$

同步段Sync\_Seg，是位时间的一部分，是CAN总线电平跳变边沿发生的地方。Sync\_Seg之外发生跳变边沿和Sync\_Seg之间的距离称为边沿的相位误差。传播时间段Prop\_Seg用于补偿CAN网络内的物理延时时间。相位缓冲段Phase\_Seg1和Phase\_Seg2围绕着采样点。（重）同步跳转宽度 (SJW) 定义了重同步将采样点在相位缓冲段定义的范围内可能移动的距离，以此用于补偿边沿相位误差。

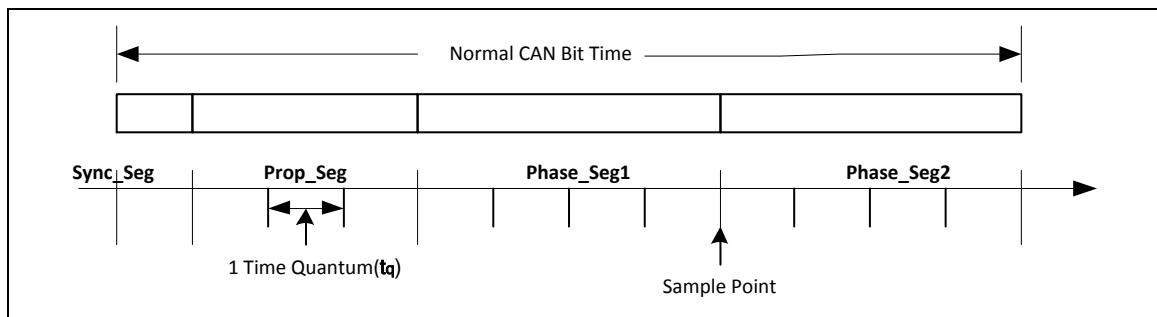


图 6.27-7 位时间

参数	范围	备注
BRP	[1..32]	定义时间片的长度 $t_q$
Sync_Seg	$1 t_q$	固定长度, CAN总线输入到APB的同步
Prop_Seg	$[1..8] t_q$	补偿物理延时时间
Phase_Seg1	$[1..8] t_q$	同步功能会将此值暂时延长
Phase_Seg2	$[1..8] t_q$	同步功能会将此值暂时缩短
SJW	$[1..4] t_q$	不能比任一个相位缓存段长
该表格描述了CAN协议要求的最小可编程范围		

表 6.27-3 CAN位时间参数

一个给定的位速率可以有不同的位时间配置，但是对于CAN网络的功能来说，物理延时时间和振荡器容错范围必须考虑。

#### 6.27.7.17 传播时间段

该部分位时间用于补偿网络的物理延时时间。这些延时是由总线上的信号传播时间和CAN节点的内部延时时间组成。

CAN总线上，CAN节点间会发生相位出错的问题，这是由两个节点间的传播时间造成的。CAN协议非破坏性的逐位仲裁和CAN 接收器发出的显性响应位，要求发送位流的CAN节点必须同时能够接收其他CAN 节点发送的显性位。图 6.27-8表示连两个CAN节点间的相位移动和传播时间

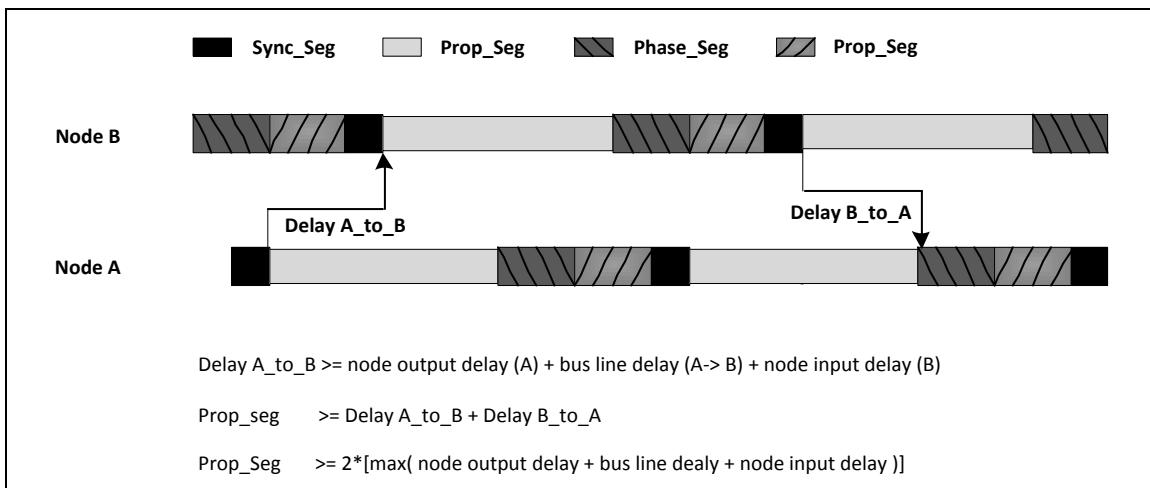


图 6.27-8 传播时间段

该示例中，节点A和B都是发送器，执行CAN总线仲裁。节点A早于节点B发送它的起始帧位，时差小于1个位时间，节点B利用接收到的从隐性到显性的边沿与A节点同步。因为B节点是在边沿发出延时(A\_to\_B)后才收到，所以B的位时间段移位到与A同步。B节点发送的是一个高优先级的标识符，所以在B传送一个显性位，而A发送一个隐性位时，B将在这个标识位仲裁中胜出。节点B发送的显性位将延时(B\_to\_A)后到达在A。

由于振荡器容差功能，节点A采样点的实际位置可以在节点A相位缓存段标称的任何位置。所以节点B的位传送必须在Phase\_Seg1开始之前到达节点A。该条件决定Prop\_Seg的长度。

如果节点B传送的隐性到显性边沿在Phase\_Seg1开始后才到达节点A，那么可能会出现节点A采样结果显性位变为隐性位的情况，这将导致一个位错误和错误标志并破坏当前帧。

这种错误只在两个节点都要求CAN总线进行仲裁，而两个节点的振荡器的偏差方向相反，且处于长总线两端的情况下才会发生。这是一个位定时配置错误(Prop\_Seg 定义时间太短)导致偶发性总线错误的例子。

一些CAN硬件提供可选择的3次采样模式，但是C\_CAN没有。在该模式下，CAN总线的输入信号通过一个低通滤波器，使用3个样本并根据大数逻辑来决定有效位值。这会导致额外的1个输入延时 $t_q$ ，因而需要更长的Prop\_Seg。

#### 6.27.7.18 相位缓冲段及同步

相位缓冲段(Phase\_Seg1 和 Phase\_Seg2)和同步跳转宽度(SJW)用于振荡器误差补偿。通过同步，可以延长或缩短相位缓冲段

同步发生在从隐性到显性的边沿，目的是用来控制边沿和采样点之间的距离。

边沿检测是通过采样每个时间片的总线电平，并与前一个采样电平进行比较实现的。同步仅发生在总线电平从隐性到显性的变化时。

如果边沿发生在Sync\_Seg里面，那么它是处于同步状态，否则边沿和Sync\_Seg结束处之间的距离即为边沿相位误差，时间片为时间误差单位。如果边沿在Sync\_Seg之前发生，相位误差为负，否则为正。

存在两种同步类型：硬同步和重同步。

硬同步在帧一开始时就会进行，而在帧内进行的是重同步

- 硬同步

硬同步之后，位时间将在Sync\_Seg的结束处重新开始，而不用管边沿相位误差。因此硬同步强制导致硬同步的边沿处于同步段内，并重新开始位计时

- 位重同步

位重同步导致位时间的缩短或延长，以致采样点位置随边沿移动

当引发重同步的边沿相位误差是正的，Phase\_Seg1被延长。如果相位误差的值少于SJW，Phase\_Seg1延长误差值个时间片，否则延长SJW时间片

当造成重同步的边沿相位误差是负的，Phase\_Seg2缩短。如果相位误差的值少于SJW，Phase\_Seg2缩短误差值个时间片，否则缩短SJW个时间片

当边沿相位误差的值小于或者等于SJW的设定值，硬同步和重同步作用相同。如果相位误差大于SJW，则重同步无法完全补偿相位误差，仍有误差（相位误差 - SJW）存在

在两个采样点间只做一次同步，在边沿和采样点间最小的距离内完成，给总线电平稳定和过滤出小于(Prop\_Seg + Phase\_Seg1)的干扰脉冲的时间

除了噪声脉冲，多数同步是由仲裁导致的。所有节点会“硬”同步在“领先”收发器（最先发送数据的收发器）的传送边沿，但是由于传播延时，这些节点不能达到理想的同步。“领头”发送器不一定获得仲裁，因此各接收器必须将自身同步到随后的“夺得领先”发送器(不同于之前的“领头”发生器)。同样的情况也发生在响应域，发送器和一些接收器必须要与在传送显性响应位时“夺得领先”的接收器取得同步。

当发送器和接收振荡器时钟周期的差异在同步时间内（最多10位）累加时，这些累加的差异不能比SJW长，这就限定了振荡器误差的范围

图 6.27-9为相位缓冲段是如何用于相位误差补偿的示例，其内有三张有两个连续的位定时的图。最上面一张表示同步在“late”边沿，最下面一张表示同步在“early”边沿，中间一张为无同步的参照图

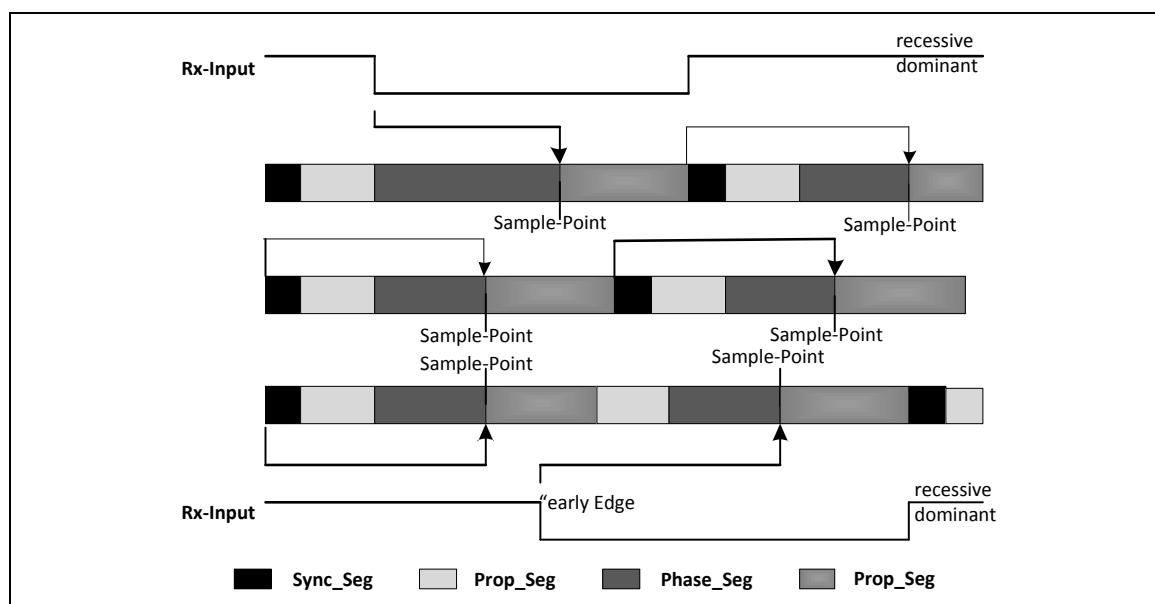


图 6.27-9同步在“late”及“early”边沿

在第一个示例中隐性到显性的跳变边沿发生在Prop\_Seg尾端。该边沿称之为“late”，因为它发生在Sync\_Seg之后。为了应对“late”边沿，Phase\_Seg1被延长，以使边沿到采样点的距离与没有边沿误差发生情况下Sync\_Seg到采样点的距离保持一致。“late”边沿的相位误差小于SJW，所以能被完全补偿。

在第二个示例中为隐性到显性的跳变边沿发生在Phase\_Seg2中。该边沿称之为“early”，因为它发生在Sync\_Seg之前。为了应对“early”边沿的情况，Phase\_Seg2被缩短，Sync\_Seg被忽略，以使边沿到采样点的距离与没有边沿误差发生的情况下Sync\_Seg到采样点的距离保持一致。和上例相同，该“early”边沿的相位误差小于SJW，所以能被完全补偿。

相位缓冲段被延长或缩短只是暂时的，在下一个位时间，它们又将恢复成程序设置的值。

在以上例子中，是从CAN硬件状态机的观点去看位时序，包括位开始的位置和采样点结束的位置。当同步“early”边沿时，硬体状态机会忽略Sync\_Seg，因为发生跳变的位置是在Phase\_Seg2内不能再重新定义为Sync\_Seg了。

图 6.27-11所示为同步如何过滤短显性噪声脉冲。这些例子中噪声都在Prop\_Seg尾端开始，长度为(Prop\_Seg+Phase\_Seg1)

在第一个例子中，同步跳转宽度大于或等于噪声脉冲的相位误差。因此采样点移动到干扰脉冲尾部后，采样到一个隐性总线电平

在第二个例子中，SJW小于相位误差，所以采样点移动不够，显性噪声脉冲被当作总线电平采样

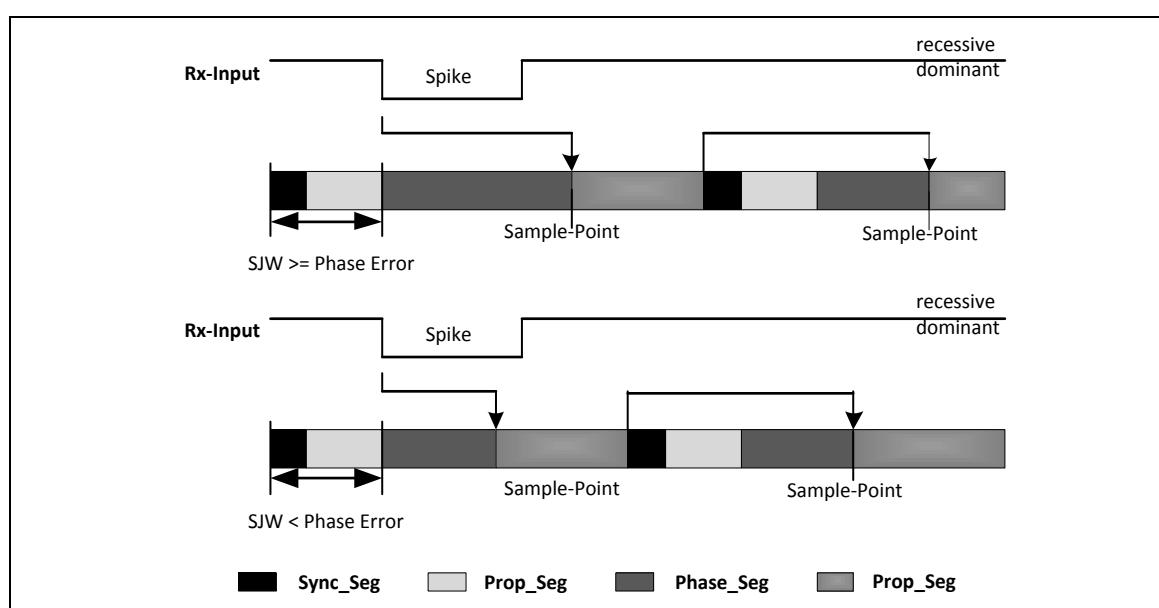


图 6.27-10 过滤显性干扰短脉冲

#### 6.27.7.19 振荡器容差范围

CAN协议从版本1.1发展到版本1.2时，扩展了振荡器偏差范围。（版本1.0未在芯片应用中实现过）信号显性到隐性边沿进行同步的方式被放弃，仅有从隐性到显性的边沿被用于同步。协议更新到版本2.0（A和B），振荡器偏差范围没有更新

振荡器频率 $f_{osc}$ 基于标称频率 $f_{nom}$ 的偏差范围 $d_f$ 的公式为

$$(1 - d_f) \cdot f_{nom} \leq f_{osc} \leq (1 + d_f) \cdot f_{nom}$$

该值取决于Phase\_Seg1, Phase\_Seg2 的比例，SJW和位时间。最大偏差值 $d_f$ 由两个条件决定（两个都

必须满足) :

$$\text{min}(\text{Phase\_Seg1}, \text{Phase\_Seg2}) \\ \text{I: } d_f \leq \frac{2 * (13 * \text{bit\_time} - \text{Phase\_Seg2})}{\dots}$$

SJW

$$\text{II: } d_f \leq \frac{20 * \text{bit\_time}}{\dots}$$

**注:** 这些条件基于APB时钟 =  $f_{osc}$ .

需要考虑SJW可能不会大于较小相位缓存段部分, 传播时间段则会限制可被用于相位缓存段的位时间

当Prop\_Seg = 1, Phase\_Seg1 = Phase\_Seg2 = SJW = 4时, 允许最大可能的振荡器偏差为1.58%。本组合中传播时间段仅占位时间10%不适用于短位时间, 可被用于在总线40m长度上位速率达到125k Bit/s (位时间=8us) 的情况

#### 6.27.7.20 配置CAN协议控制器

在大多数CAN控制器, 也包括C\_CAN, 位定时配置设置在两个寄存器字节中。Prop\_Seg, Phase\_Seg1 (为TSEG1 (CAN\_BTIME[11:8])) 和 Phase\_Seg2 (TSEG2 (CAN\_BTIME[14:12])) 组合在一个字节中。SJW和BRP组合在另一个字节中

在这些定时寄存器中, 程序必须为TSEG1, TSEG2, SJW, 和 BRP这四个成员赋值, 赋的值比它们的实际功能值小1。因此程序赋值范围为[0..n-1], 而不是[1..n]。举例: SJW (功能范围为[1..4]) 仅用两个位就可以表示

因此位时间的长度 (程序设定值) 为[TSEG1 + TSEG2 + 3]  $t_q$  或 (功能值) 为[Sync\_Seg + Prop\_Seg + Phase\_Seg1 + Phase\_Seg2]  $t_q$

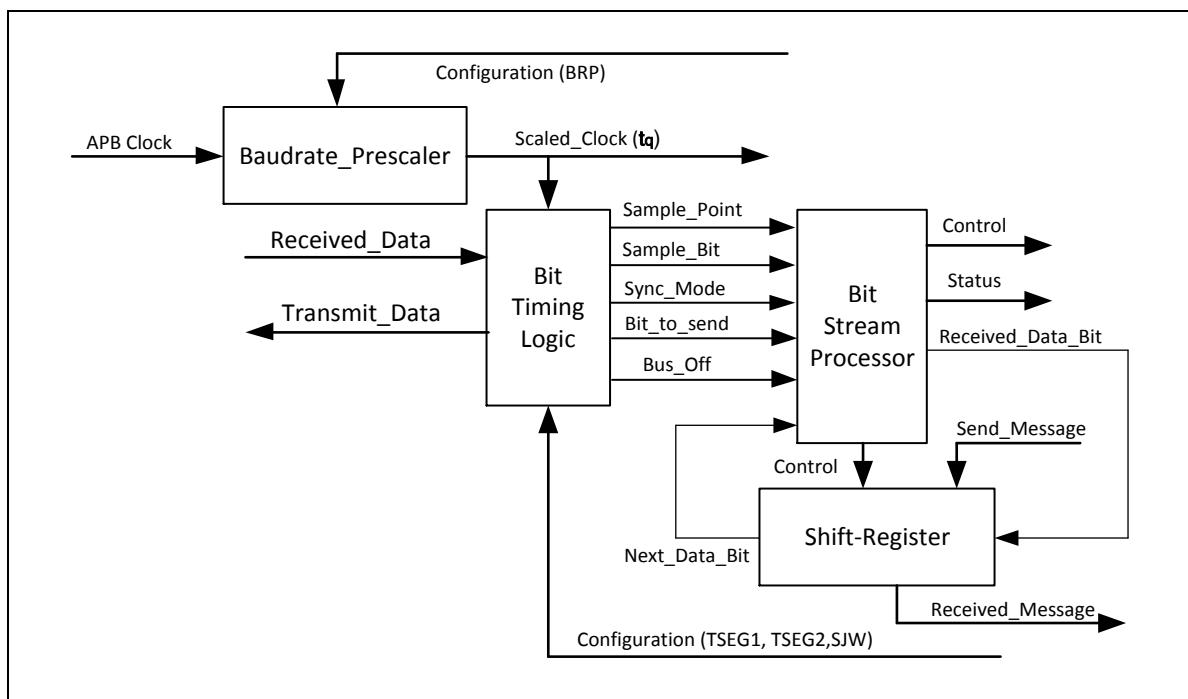


图 6.27-11 CAN内核CAN协议控制器架构

根据CAN协议配置位定时寄存器中的数据。波特率分频器（用BRP配置）定义了时间片的长度，位时间的基本时间单元；位定时逻辑（由TSEG1, TSEG2, 和 SJW设置）定义了位时间中的时间片数目。

位时间的处理，采样点位置的计算和偶发性同步都是由位定时逻辑BTL(Bit Timing Logic) 控制，BTL每时间片就要计算一次。CAN协议控制器的其他部分，位流处理器BSP(Bit Stream Processor)每个位时间在采样点计算一次

移位寄存器以串行方式发送报文，以并行方式接收报文。该寄存器的载入和移位由BSP控制

BSP把报文变成帧，或反之。它产生和丢弃包固定格式位，插入和提取填充位，计算和检查CRC码，执行错误管理，以及决定使用哪种同步类型。它在采样点进行计算并采样总线输入位，计算在采样点之后发送的下一位（如数据位，CRC位，填充位，错误标记或空闲）所需的时间被称为信息处理时间（IPT）

IPT和应用有关，但是不会长于 $2 t_q$ ；对C\_CAN来说IPT为 $0 t_q$ 。它的长度是Phase\_Seg2程序设定长度的下限。在同步的情况下，Phase\_Seg2可能被缩短到一个小于IPT的值，该值不会影响总线定时

#### 6.27.7.21 计算位定时参数

通常，位定时配置的计算从一个期望的位速率或位时间开始。位时间（1/位 速率）的结果必须是APB时钟周期的整数倍。

位时间可能包括4到25个时间片，时间片的长度 $t_q$ 由波特率分频器定义  $t_q = (\text{Baud Rate Prescaler})/f_{\text{apb\_clk}}$ 。通过下面的步骤，几种组合都可达到期望的位时间

首先，位时间中需要定义的部分是Prop\_Seg。它的长度取决于延时时间（用APB时钟测量）。对于可扩展的CAN 总线系统，还要定义最大的总线长度和最大的节点延时。Prop\_Seg的时间也要转化成时间片（向上取最接近的 $t_q$ 的整数倍的值）

Sync\_Seg是 $1 t_q$ （固定）长度，留下（位时间- Prop\_Seg - 1） $t_q$ 给两个相位缓存段。如果剩下的 $t_q$ 个

数是偶数，相位缓存段有相同长度， $\text{Phase\_Seg2} = \text{Phase\_Seg1}$ ，如果是奇数情况，则 $\text{Phase\_Seg2} = \text{Phase\_Seg1} + 1$ 。

$\text{Phase\_Seg2}$ 最小的标称值也必须要考虑一下。 $\text{Phase\_Seg2}$ 不能比CAN控制器的IPT短，而IPT的范围是 $[0..2] t_q$ 。取决于控制器的实际硬件

同步跳转宽度的长度要设为它的最大值，即4和 $\text{Phase\_Seg1}$ 中的较小值。

由振荡器偏差造成的最终配置结果由章节“振荡器偏差范围”中的方程式计算

如果有多种配置可选，那么应该选允许最高振荡器偏差范围的配置。

带有多个不同系统时钟的CAN节点要求不同的配置来达到相同的位速率。CAN网络中的传播时间计算是基于最长延时时间的节点，整个网络只做一次传播时间计算。

CAN系统振荡器偏差范围由最低偏差容忍的节点决定。

根据计算结果，可能会要求减少总线长度或位速率，或者提高振荡器频率的稳定性，目的是寻找满足协议的CAN位定时配置。配置结果写入位定时寄存器：

( $\text{Phase\_Seg2-1}$ ) & ( $\text{Phase\_Seg1+Prop\_Seg-1}$ ) & ( $\text{SynchronisationJumpWidth-1}$ ) & ( $\text{Prescaler-1}$ )

高波特率位定时示例：

该例中，APB\_CLK的时钟为10MHz，BRP (CAN\_BTIME[5:0])为0，位速率为1 Mbit/s

$t_q$	100	ns	$= t_{APB\_CLK}$
总线驱动延时	50	ns	
接收器电路延时	30	ns	
总线延时 (40m)	220	ns	
$t_{Prop}$	600	ns	$= 6 \cdot t_q$
$t_{SJW}$	100	ns	$= 1 \cdot t_q$
$t_{TSeg1}$	700	ns	$= t_{Prop} + t_{SJW}$
$t_{TSeg2}$	200	ns	$= \text{信息处理时间} + 1 \cdot t_q$
$t_{Sync-Seg}$	100	ns	$= 1 \cdot t_q$
位时间	1000	ns	$= t_{Sync-Seg} + t_{TSeg1} + t_{TSeg2}$
APB_CLK容差	0.39	%	$= \frac{\text{Min}(PB1, PB2)}{2 \times 13 \times (\text{bit time} - PB2)}$
			$0.1 \text{ s}$
			$= \frac{0.1 \text{ s}}{2 \times 13 \times 1 \text{ s} - 0.2 \text{ s}}$

这个例子里，位时钟参数 (2-1)3&(7-1)4&(1-1)2&(1-1)6，位时钟寄存器设定为0x1600.

#### Note:

1. PB1/2: 表示相位缓冲段 1/2
2. 下标  $(2-1)_3$  表示 位定时寄存器相关为的数值

低波特率位定时示例：

该例中，APB\_CLK的时钟为2MHz，BRP (CAN\_BTME[5:0])为1，位速率为1 100 Kbit/s.

$$t_q \quad 1 \text{ } \mu\text{s} = 2 \cdot t_{\text{APB\_CLK}}$$

总线驱动延时	200	ns
接收器电路延时	80	ns
总线延时 (40m)	220	ns
$t_{\text{Prop}}$	1	$\mu\text{s} = 1 \cdot t_q$
$t_{\text{SJW}}$	4	$\mu\text{s} = 4 \cdot t_q$
$t_{\text{TSeq1}}$	5	$\mu\text{s} = t_{\text{Prop}} + t_{\text{SJW}}$
$t_{\text{TSeq2}}$	4	$\mu\text{s} = \text{信息处理时间} + 3 \cdot t_q$
$t_{\text{Sync-Seg}}$	1	$\mu\text{s} = 1 \cdot t_q$
位时间	10	$\mu\text{s} = t_{\text{Sync-Seg}} + t_{\text{TSeq1}} + t_{\text{TSeq2}}$
APB_CLK容差 1.58%		$= \frac{\text{Min}(PB1, PB2)}{2 \times 13 \times (\text{bit time} - PB2)}$

4us

$$= \frac{4\mu\text{s}}{2 \times (13 \times (10\mu\text{s} - 4\mu\text{s}))}$$

这个例子里，位时钟参数是(4-1)<sub>3</sub> & (5-1)<sub>4</sub> & (4-1)<sub>2</sub> & (2-1)<sub>6</sub>, 位时钟寄存器的值是=0x34C1.

### 6.27.7.22 CAN接口复位状态

在硬件复位后，C\_CAN寄存器中值为“CAN寄存器映射寄存器描述”中的复位值

此外，复位后为Bus-off态，CAN\_TX的输出被设置为隐性 (HIGH)。CAN控制寄存器的值为0x0001 (Init='1') 使能软件初始化。直到应用软件设置Init (CAN\_CON[0])位为'0' C\_CAN才会影响CAN总线。保存在报文RAM中的数据不受硬件复位影响。在上电后，报文RAM中的内容是不确定的

### 6.27.8 寄存器映射

**R:** 只读, **W:** 只写, **R/W:** 可读写

寄存器	偏移量	R/W	描述	复位值
<b>CAN 基地址:</b>				
<b>CAN_BA = 0x400A_0000</b>				
CAN_CON	CAN_BA+0x00	R/W	控制寄存器	0x0000_0001
CAN_STATUS	CAN_BA+0x04	R/W	状态寄存器	0x0000_0000
CAN_ERR	CAN_BA+0x08	R	错误计数寄存器	0x0000_0000
CAN_BTIME	CAN_BA+0x0C	R/W	位定时寄存器	0x0000_2301
CAN_IIDR	CAN_BA+0x10	R	中断标识符寄存器	0x0000_0000
CAN_TEST	CAN_BA+0x14	R/W	测试寄存器 (寄存器表注1)	0x0000_0080
CAN_BRPE	CAN_BA+0x18	R/W	波特率预分频扩展寄存器	0x0000_0000
CAN_IFn_CREQ n = 1,2	CAN_BA+0x20 (0x60 *(n-1))	+ R/W	IFn (寄存器表注2) 命令请求寄存器	0x0000_0001
CAN_IFn_CMASK n=1,2	CAN_BA+0x24 (0x60 *(n-1))	+ R/W	IFn 命令掩码寄存器	0x0000_0000
CAN_IFn_MASK1 n=1,2	CAN_BA+0x28 (0x60 *(n-1))	+ R/W	IFn 掩码 1 寄存器	0x0000_FFFF
CAN_IFn_MASK2 n=1,2	CAN_BA+0x2C (0x60 *(n-1))	+ R/W	IFn 掩码 2 寄存器	0x0000_FFFF
CAN_IFn_ARB1 n=1,2	CAN_BA+0x30 (0x60 *(n-1))	+ R/W	IFn 仲裁 1 寄存器	0x0000_0000
CAN_IFn_ARB2 n=1,2	CAN_BA+0x34 (0x60 *(n-1))	+ R/W	IFn 仲裁 2 寄存器	0x0000_0000
CAN_IFn_MCON n=1,2	CAN_BA+0x38 (0x60 *(n-1))	+ R/W	IFn 报文控制寄存器	0x0000_0000
CAN_IFn_DAT_A1 n=1,2	CAN_BA+0x3C (0x60 *(n-1))	+ R/W	IFn 数据 A1 寄存器 (寄存器表注3)	0x0000_0000
CAN_IFn_DAT_A2 n=1,2	CAN_BA+0x40 (0x60 *(n-1))	+ R/W	IFn 数据 A2 寄存器 (寄存器表注3)	0x0000_0000
CAN_IFn_DAT_B1 n=1,2	CAN_BA+0x44 (0x60 *(n-1))	+ R/W	IFn 数据 B1 寄存器 (寄存器表注 3)	0x0000_0000
CAN_IFn_DAT_B2 n=1,2	CAN_BA+0x48 (0x60 *(n-1))	+ R/W	IFn 数据 B2 寄存器 (寄存器表注3)	0x0000_0000
CAN_TXREQ1	CAN_BA+0x100	R	发送请求寄存器 1	0x0000_0000
CAN_TXREQ2	CAN_BA+0x104	R	发送请求寄存器 2	0x0000_0000
CAN_NDAT1	CAN_BA+0x120	R	新数据寄存器 1	0x0000_0000

CAN_NDAT2	CAN_BA+0x124	R	新数据寄存器 2	0x0000_0000
CAN_IPND1	CAN_BA+0x140	R	中断挂起寄存器 1	0x0000_0000
CAN_IPND2	CAN_BA+0x144	R	中断挂起寄存器2	0x0000_0000
CAN_MVLD1	CAN_BA+0x160	R	报文有效寄存器 1	0x0000_0000
CAN_MVLD2	CAN_BA+0x164	R	报文有效寄存器2	0x0000_0000
CAN_WU_EN	CAN_BA+0x168	R/W	唤醒使能控制寄存器	0x0000_0000
CAN_WU_STATUS	CAN_BA+0x16C	R/W	唤醒状态寄存器	0x0000_0000

- 注: 1. 0x00 & 0br0000000, 其中r表示CAN\_RX的当前值  
2. IFn: 两组报文接口寄存器— IF1 和 IF2有相同的功能  
3. An/Bn: 两组数据寄存器A1, A2 和 B1, B2  
4. CANx\_BA,CAN寄存器基地址,x=0或x=1

CAN寄存器表中各个位的功能

地址偏移	寄存器名	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
00h	CAN_CON									Test	CCE	DAR	Res	EIE	SIE	IE	Init
04h	CAN_STATUS									BOff	EWarn	EPass	RxOk	TxOk			LEC
08h	CAN_ERR	RP															TEC7-0
0Ch	CAN_BTIME	Res			TSeg2			TSeg1		SJW							BRP
10h	CAN_IIDR																IntId7-0
14h	CAN_TEST									Rx	Tx1	Tx0	LBack	Silent	Basic		Reserved
18h	CAN_BRPE																BRPE
20h	CAN_IF1_CRE_Q	Busy															Message Number
24h	CAN_IF1_CMASK									WR/RD	Mask	Arb	Control	ClIntPnd	TxRqst/	Data A	Data B
28h	CAN_IF1_MAS_K1										Msk15-0						
2Ch	CAN_IF1_MAS_K2	MXtd	MDir	Res													Msk28-16
30h	CAN_IF1_ARB1											ID15-0					
34h	CAN_IF1_ARB2	MsgVal	Xtd	Dir									ID28-16				

地址偏移	寄存器名	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
38h	CAN_IF1_MCO_N	NewDat	MsgLst	IntPnd	UMask	TxIE	RxIE	RmtEn	TxRqst	EoB	Reserved	Reserved	Reserved	Reserved	Reserved	DLC3-0	Reserved
3Ch	CAN_IF1_DAT_A1				Data(1)						Data(0)						
40h	CAN_IF1_DAT_A2				Data(3)						Data(2)						
44h	CAN_IF1_DAT_B1				Data(5)						Data(4)						
48h	CAN_IF1_DAT_B2				Data(7)						Data(6)						
80h	CAN_IF2_CREQ	Busy														Message Number	
84h	CAN_IF2_CMASK											WR/RD	Mask	Arb	Control	ClrlntPnd	TxRqst/
88h	CAN_IF2_MASK_1										Msk15-0					Data A	Data B
8Ch	CAN_IF2_MASK_2	MXtd	MDir	Res.							Msk28-16						
90h	CAN_IF2_ARB1										ID15-0						
94h	CAN_IF2_ARB2	MsgVal	Xtd	Dir							ID28-16						
98h	CAN_IF2_MCO_N	NewDat	MsgLst	IntPnd	UMask	TxIE	RxIE	RmtEn	TxRqst	EoB	Reserved	Reserved	Reserved	Reserved	Reserved	DLC3-0	Reserved
9Ch	CAN_IF2_DAT_A1				Data(1)						Data(0)						

地址偏移	寄存器名	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
------	------	----	----	----	----	----	----	---	---	---	---	---	---	---	---	---	---

A0h	CAN_IF2_DAT_A2	Data(3)	Data(2)
A4h	CAN_IF2_DAT_B1	Data(5)	Data(4)
A8h	CAN_IF2_DAT_B2	Data(7)	Data(6)
100h	CAN_TXREQ1	TxRqst16-1	
104h	CAN_TXREQ2	TxRqst32-17	
120h	CAN_NDAT1	NewDat16-1	
124h	CAN_NDAT2	NewDat32-17	
140h	CAN_IPND1	IntPnd16-1	
144h	CAN_IPND2	IntPnd32-17	
160h	CAN_MVLD1	MsgVal16-1	
164h	CAN_MVLD2	MsgVal32-17	
168h	CAN_WU_EN	Reserved	WAKUP_EN
16Ch	CAN_WU_STAT_US	Reserved	WAKUP_STS
170h	CAN_RAM_CEN	Reserved	RAM_CEN
Others	Reserved	Reserved	

表 6.27-4 CAN 寄存器中每个位的功能

注: 读保留位都为‘0’, 除了IFn Mask 2寄存器, 读IFn Mask 2中保留位的值为‘1’

Res. = 保留

### 6.27.9 寄存器描述

C\_CAN 占用256字节的地址空间。这些寄存器按照16-位寄存器安排。

两组接口寄存器(IF1 和 IF2)控制软件对报文RAM的访问，它们是为报文RAM传输提供的缓存，避免软件访问和报文接收/发送之间发生冲突。

## CAN CON CAN 控制寄存器

寄存器	偏移量	R/W	描述	复位值
CAN_CON	CAN_BA+0x00	R/W	控制寄存器	0x0000_0001

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Test	CCE	DAR	Reserved	EIE	SIE	IE	Init

位	描述	
[31:8]	<b>Reserved</b>	保留
[7]	<b>Test</b>	测试模式使能 0=正常操作模式 1=测试模式
[6]	<b>CCE</b>	配置改变使能 0=禁止写访问到 位定时寄存器 1=允许写访问到 位定时寄存器(CAN_BTIME)(当Init(CAN_CON[0]) = 1)
[5]	<b>DAR</b>	禁止自动重传 0=使能被干扰的报文自动重传 1=禁止自动重传
[4]	<b>Reserved</b>	保留
[3]	<b>EIE</b>	错误中断使能 0=禁止 - 错误状态不会产生中断 1=使能 - 改变状态寄存器的Boff (CAN_STATUS[7]) 或 EWarn (CAN_STATUS[6])位将产生中断
[2]	<b>SIE</b>	状态改变中断使能 0=禁止 - 状态改变不会产生中断 1=使能 - 当一条报文传输成功或一个CAN总线错误被检测到时将产生中断
[1]	<b>IE</b>	模块中断使能 0=禁止 1=使能
[0]	<b>Init</b>	Init初始化 0=正常操作

		1=初始化开始
--	--	---------

**注:** 不能通过设置或复位Init(CAN\_CON[0])位来缩短bus-off修复时序(参见CAN规范REV.2.0)。如果设备进入bus-off状态, 它将按照自己的步调设置Init位, 停止所有总线的活动。一旦Init被CPU清除, 设备将在恢复正常操作之前等待129个总线空闲 (129\*11个连续的隐性位)。在bus-off修复时序的最后, 错误管理计数器将被复位。

在复位Init之后的等待过程中, 每次检测到11个连续的隐性位, 一个Bit0Error码就会写入状态寄存器, 使能CPU立即去检查CAN总线是否被显性或连续的干扰困住, 从而监控busoff修复时序的进程。

## CAN STATUS CAN 状态寄存器

寄存器	偏移量	R/W	描述	复位值
CAN_STATUS	CAN_BA+0x04	R/W	状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
BOff	EWarn	EPass	RxOK	TxOK	LEC		

位	描述	
[31:8]	Reserved	保留
[7]	BOff	<p><b>Bus-Off状态 (只读)</b>            0=CAN模块没有处于Bus-Off状态            1=CAN模块处于Bus-Off状态</p>
[6]	EWarn	<p><b>错误警告状态 (只读)</b>            0=所有错误计数器的数值都在96次错误警告以下            1= EML中至少有一个错误计数器达到了96次错误警告</p>
[5]	EPass	<p><b>被动错误 (只读)</b>            0=Can 内核处于主动错误状态            1=CAN内核处于CAN规范定义的被动错误状态</p>
[4]	RxOK	<p><b>成功接收一条报文</b>            0=自上次该位被CPU复位后,没有报文被成功接收到。CAN内核不能复位该位            1=自上次该位被CPU复位后,一条报文被成功接收到。(独立于接收过滤的结果)</p>
[3]	TxOK	<p><b>成功发送一条报文</b>            0=自上次该位被CPU复位后,没有报文被成功发送。CAN内核不能复位该位            1=自上次该位被CPU复位后,一条报文被成功发送。</p>
[2:0]	LEC	<p><b>上一次的错误码 (最近一次出现在CAN总线上的错误的类型)</b>            LEC中的数值代表最近一次出现在CAN总线上错误的类型。当报文被成功传送, 并且没有任何错误, LEC中的数值将被清零。'7'为没有用到的代码, CPU 写入该数值到LEC用于检查是否有更新。下表详细描述错误代码的意义。</p>

错误码	意义
0	无错误
1	无错误
2	填充错误：连续超过5个相同的位出现在接收报文中，这是不允许的。
3	格式错误：收到的帧中固定格式的部分出现了错误格式。
4	应答错误：CAN内核传送的报文没有被另一个节点应答。
5	Bit1错误：在发送一条报文的过程中（除仲裁域之外），设备想发送一个隐形位（逻辑值为'1'的位），但是监测到总线的值为显性。.
6	Bit0错误：在发送一条报文的过程中（或应答位，或主动错误标志，或超载标志），设备想发送一个显性位（逻辑值为'0'的位），但是监测到总线的值为隐性。在bus-off恢复的过程中，每检测到一个连续的11位隐性位，该状态设置一次。使能CPU监控bus-off恢复序列的进程。（提示总线没有被显性或连续的干扰困住）
7	CRC错误：收到的报文CRC检查不正确，发过来的报文中接收的CRC和计算收到数据得到的CRC不一致。

表 6.27-5 错误码

### 状态中断

状态中断由 BOff (CAN\_STATUS[7]) 和 EWarn(CAN\_STATUS[6]) (错误中断) 位或 RxOK (CAN\_STATUS[4]), TxOK (CAN\_STATUS[3]) 和 LEC (CAN\_STATUS[2:0]) (状态改变中断) 位产生 (假设CAN寄存器中相应的使能位被置位了)。改变EPass(CAN\_STATUS[5])位或写入数据到RxOK, TxOK或LEC的不会产生状态中断。

如果该中断挂起，读状态寄存器将清除中断寄存器中的中断状态值 (8000h)

CAN\_ERR CAN 错误计数寄存器

寄存器	偏移量	R/W	描述	复位值
CAN_ERR	CAN_BA+0x08	R	错误计数寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
RP	REC						
7	6	5	4	3	2	1	0
TEC							

位	描述	
[31:16]	<b>Reserved</b>	保留
[15]	<b>RP</b>	<b>接收错误认可</b> 0=接收错误计数器在被动错误标准以下 1=接收错误计数器已经达到CAN规范定义的被动错误标准
[14:8]	<b>REC</b>	<b>接收错误计数器</b> 接收错误计数器的当前状态。值到0到127之间。
[7:0]	<b>TEC</b>	<b>发送错误计数器</b> 发送错误计数器的当前状态。值到0到255之间。

**CAN\_BTIME 位定时寄存器**

寄存器	偏移量	R/W	描述	复位值
CAN_BTIME	CAN_BA+0x0C	R/W	位定时寄存器	0x0000_2301

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved	TSeg2			TSeg1			
7	6	5	4	3	2	1	0
SJW		BRP					

位	描述	
[31:15]	<b>Reserved</b>	保留
[14:12]	<b>TSeg2</b>	采样点后的时间段 0x0-0x7: TSeg2有效值为[0 … 7]。该位硬件实际解释值比程序设定值多1
[11:8]	<b>TSeg1</b>	采样点前减Sync_Seg的时间段 0x1-0x0f: TSeg1有效值为[1 … 15]. 该位硬件实际解释值比程序设定值多1
[7:6]	<b>SJW</b>	(重)同步跳转宽度 0x0-0x3:有效值为[0 … 3]. 该位硬件实际解释值比程序设定值多1
[5:0]	<b>BRP</b>	波特率预分频器 0x01-0x3f:该值用于振荡器频率除频产生位时间片。位时间是时间片的倍数累积。波特率分频器的有效值为[0 … 63]. 该位硬件实际解释值比程序设定值多1

**注:** 复位值为0x2301, 对应的模块时钟APB\_CLK为8 MHz, C\_CAN的位速率为500kBit/s。这些寄存器只在CCE(CAN\_CON[6])位和CAN (CAN\_CON[0])控制寄存器中Init位被置位后才可写。

## CAN\_IIDR 中断标识寄存器

寄存器	偏移量	R/W	描述	复位值
CAN_IIDR	CAN_BA+0x10	R	中断标识符寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
IntId							
7	6	5	4	3	2	1	0
IntId							

位	描述
[15:0]	<b>IntId</b> <b>中断标识符（提示中断源）</b> 如果几个中断同时挂起，CAN 中断寄存器将忽略中断挂起时序而指向拥有最高优先级的挂起中断。中断将保持挂起直到应用软件清除它。如果IntId不为0x0000且IE(CAN_IFn_MCON[1])置位，则到IEC的IRQ中断信号会被激活。中断保持激活状态直到IntId恢复为0x0000（引起中断的原因已复位）或直到IE复位。 状态中断拥有最高优先级。在报文中断中，报文对象的优先级随着报文编号的增加而降低。 一个报文中断通过清除该报文对象的IntPnd(CAN_IFn_MCON[13])位清除。状态中断通过状态寄存器清除

IntId值	意义
0x0000	没有中断挂起。
0x0001-0x0020	造成中断的报文对象编号
0x0021-0x7FFF	未使用
0x8000	状态中断
0x8001-0xFFFF	未使用

表 6.27-6 中断源

## CAN TEST 测试寄存器

寄存器	偏移量	R/W	描述	复位值
CAN_TEST	CAN_BA+0x14	R/W	测试寄存器 (寄存器表备注1)	0x0000_0080

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Rx	Tx		LBack	Silent	Basic	Reserved	

位	描述	
[31:8]	Reserved	保留
[7]	Rx	监测CAN_RX管脚的当前实际值 (只读) *注 0 = CAN总线为显性(CAN_RX = '1'). 1 = CAN总线为隐性(CAN_RX = '0').
[6:5]	Tx	Tx[1:0]: 控制CAN_TX 管脚 00 = 复位值, CAN_TX由CAN内核控制 01 = 采样点监测CAN_TX管脚 10 = CAN_TX管脚驱动一个显性值('0') 11 = CAN_TX管脚驱动一个 隐性值('1')
[4]	LBack	回环模式 0=回环模式禁止 1=回环模式使能
[3]	Silent	静默 模式 0 = 正常操作 1 = 模块工作在静默模式
[2]	Basic	基本模式 0 = 基本模式禁止 1 = IF1 寄存器用作TX Buffer, IF2寄存器用作RX Buffer.
[1:0]	Reserved	保留

复位值: 0000 0000 R000 0000 b (R:RX管脚的当前值)

注: 通过设置CAN控制寄存器的Test位来使能测试寄存器的写访问。不同的测试功能可以组合使用，但是Tx[1:0] (CAN\_TEST[6:5]) “00”会干扰报文传输。

CAN\_BRPE 波特率预分频扩展寄存器

寄存器	偏移量	R/W	描述	复位值
CAN_BRPE	CAN_BA+0x18	R/W	波特率预分频扩展寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved				BRPE			

位	描述	
[31:4]	Reserved	保留
[3:0]	BRPE	<b>BRPE: 波特率预分频扩展</b> 0x00-0x0F: 通过编程BRPE，波特率分频可以被扩展到1023。该位硬件实际解释值比程序计算BRPE (MSBs) 和 BTIME (LSBs) 的值要多1

**报文接口寄存器组：**

模块共有2组接口寄存器，用于控制CPU访问报文RAM。接口寄存器通过缓存传输避免CPU访问报文RAM时与CAN报文发送和接收发生冲突。一个完整的报文对象或部分报文对象在报文RAM和IFn报文缓存寄存器间只需一次传输就可以完成。

除基本测试模式外，2组接口寄存器的功能是相同的。它们可以这样用：一组寄存器用于传送数据到报文RAM，另一组用于接收来自报文RAM的数据，它们可以互相中断。下表（IF1和IF2 报文接口寄存器组）提供了2组接口寄存器的概述。

每组接口寄存器由报文缓存寄存器构成，分别由各自的命令寄存器控制。命令掩码寄存器设定数据传输的方向和报文对象的哪个部分将要被传输。命令请求寄存器用于选择报文RAM中的一个报文对象作为传输的目标或来源，并开始执行命令掩码寄存器中指定的命令。

地址	IF1 寄存器组	地址	IF2 寄存器组
CAN_BA+0x20	IF1 命令请求	CAN_BA+0x80	IF2 命令请求
CAN_BA+0x24	IF1 命令掩码	CAN_BA+0x84	IF2 命令掩码
CAN_BA+0x28	IF1 掩码 1	CAN_BA+0x88	IF2 掩码1
CAN_BA+0x2C	IF1 掩码 2	CAN_BA+0x8C	IF2 掩码 2
CAN_BA+0x30	IF1 仲裁 1	CAN_BA+0x90	IF2 仲裁1
CAN_BA+0x34	IF1 仲裁2	CAN_BA+0x94	IF2 仲裁2
CAN_BA+0x38	IF1 报文控制	CAN_BA+0x98	IF2 报文控制
CAN_BA+0x3C	IF1 数据 A 1	CAN_BA+0x9C	IF2 数据 A 1
CAN_BA+0x40	IF1 数据 A 2	CAN_BA+0xA0	IF2 数据 A 2
CAN_BA+0x44	IF1 数据 B 1	CAN_BA+0xA4	IF2 数据 B 1
CAN_BA+0x48	IF1 数据 B 2	CAN_BA+0xA8	IF2 数据 B 2

表 6.27-7 IF1 及 IF2 报文接口寄存器

CAN\_IFn\_CREQ\_IFn 命令请求寄存器

寄存器	偏移量	R/W	描述	复位值
CAN_IFn_CREQ	CAN_BA+0x20 (0x60*(n-1))	+R/W	IFn (寄存器表备注2) 命令请求寄存器	0x0000_0001

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Busy	Reserved						
7	6	5	4	3	2	1	0
Reserved		Message Number					

位	描述	
[31:16]	Reserved	保留
[15]	Busy	忙标志 0 = 读 / 写动作已经完成 1 = 写到IFn命令请求寄存器的动作正在进行。该位只能由软件读取。
[14:6]	Reserved	保留
[5:0]	Message Number	报文编号 0x01-0x20: 有效的报文号, 报文RAM中被选择用于数据传输的报文对象 0x00: 非有效的报文号, 视同0x20 0x21-0x3f: 非有效的报文号, 视同0x01-0x1F

一旦应用软件写报文编号到命令请求寄存器, 报文传输就开始了。伴随着这个写操作, Busy位将自动被置位来通知CPU传输正在进行中。在等待3到6个APB\_CLK 周期后, 接口寄存器和报文RAM之间的传输完成了。Busy位被清除。

**注意:** 当写入到命令请求寄存器中的报文编号为非有效值时, 报文号将被解释转换为一个有效的值, 对应报文对象也将会被传输

CAN\_IFn\_CMASK\_IFn 命令掩码寄存器

IFn命令掩码寄存器指定传输方向和选择哪个IFn报文缓存寄存器作为数据传输的源或目标。

寄存器	偏移量	R/W	描述	复位值
CAN_IFn_CMASK	CAN_BA+0x24 (0x60 *(n-1))	+R/W	IFn 命令掩码寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
WR/RD	Mask	Arb	Control	ClrlntPnd	TxRqst/NewDat	DAT_A	DAT_B

位	描述
[31:8]	<b>Reserved</b> 保留
[7]	<b>WR/RD</b> 写/读 模式 0= 读：将命令请求寄存器中指定地址的报文对象传输到选中的报文缓存寄存器中 1 = 写：将选中的报文缓存寄存器的数据传送到命令请求寄存器中指定地址的报文对象中
[6]	<b>Mask</b> 访问掩码位 写操作 0=掩码位不变 1=传输标识符 Mask + MDir + MXtd 到报文对象 读操作： 0=掩码位不变 1=传输标识符 Mask + MDir + MXtd 到IFn报文缓存寄存器
[5]	<b>Arb</b> 访问仲裁位 写操作： 0=仲裁位不变 1= 传 输 Identifier + Dir (CAN_IFn_ARB2[13]) + Xtd (CAN_IFn_ARB2[14]) + MsgVal (CAN_IFn_ARB2[15])到报文对象 读操作： 0=仲裁位不变 1= 传 输 Identifier + Dir (CAN_IFn_ARB2[13]) + Xtd (CAN_IFn_ARB2[14]) + MsgVal (CAN_IFn_ARB2[15])到IFn报文缓存寄存器
[4]	<b>Control</b> 访问控制位 写操作： 0=控制位不变

		<p>1=传输控制位到报文对象 读操作： 0=控制位不变 1=传输控制位到IFn报文缓存寄存器</p>
[3]	<b>ClrIntPnd</b>	<p><b>清除中断挂起位：</b> 写操作： 当写报文对象时，该位忽略。 读操作： 0=IntPnd(CAN_IFn_MCON[13])位保持不变 1=清除报文对象中的IntPnd位</p>
[2]	<b>TxRqst/NewDat</b>	<p><b>写操作时访问传送请求位</b> 0=TxRqst位不变 1=设置TxRqst位 <b>注意：</b>如果传送是通过设置IFn命令掩码寄存器中的TxRqst/NewDat位请求进行的，则IFn报文控制寄存器中的TxRqst位将被忽略。 读操作时访问New Data位 0&gt;NewDat位保持不变 1=清除报文对象中的NewDat位 <b>注意：</b>可通过复位控制位IntPnd和NewDat实现对一个报文对象的读取。传送到IFn报文控制寄存器的这些值总是反映其状态（在复位这些位之前）。</p>
[1]	<b>DAT_A</b>	<p><b>访问数据字节[3: 0]</b> 写： 0=数据[3:0]未变 1=写传输数据字节[3:0]到报文对象 读： 0= 数据字节[3:0]未变 1=传送数据字节[3:0]到IFn报文缓存寄存器</p>
[0]	<b>DAT_B</b>	<p><b>访问数据字节[7: 4]</b> 写： 0=数据[7:4]未变 1=写传输数据字节[7:4]到报文对象 读： 0= 数据字节[7:4]未变 1=传送数据字节[7:4]到IFn报文缓存寄存器</p>

CAN\_IFn\_MASK1 IFn 掩码 1 寄存器

寄存器	偏移量	R/W	描述	复位值
CAN_IFn_MASK1	CAN_BA+0x28 (0x60 *(n-1))	+R/W	IFn 掩码 1 寄存器	0x0000_FFFF

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Msk							
7	6	5	4	3	2	1	0
Msk							

位	描述	
[31:16]	<b>Reserved</b>	保留
[15:0]	<b>Msk</b>	<p><b>标识符掩码15-0</b></p> <p>0=对应的报文对象标识符位不用于接收过滤 1=对应的报文对象标识符位用于接收过滤</p>

CAN\_IFn\_MASK2 IFn 掩码 2 寄存器

寄存器	偏移量	R/W	描述	复位值
CAN_IFn_MASK2	CAN_BA+0x2C (0x60 *(n-1))	+R/W	IFn 掩码 2 寄存器	0x0000_FFFF

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
MXtd	MDir	Reserved	Msk				
7	6	5	4	3	2	1	0
Msk							

位	描述	
[31:16]	<b>Reserved</b>	保留
[15]	<b>MXtd</b>	<p><b>掩码扩展标识符</b>            0=扩展标识符 (IDE) 对于接收过滤没有影响            1=扩展标识符 (IDE) 用于接收过滤</p> <p><b>注意:</b> 报文对象采用11位(标准)标识符时, 接收到的数据帧的标识符将会写入到ID28 - ID18 (CAN_IFn_ARB2[12:2])。所有这些位以及掩码位 Msk28 - Msk18 (CAN_IFn_MASK2[12:2]), 都要列入到接收过滤的考虑中。</p>
[14]	<b>MDir</b>	<p><b>掩码报文方向</b>            1=报文方向(Dir (CAN_IFn_ARB2[13]))位用于接收过滤            0=报文方向位对于接收过滤没有影响</p>
[13]	<b>Reserved</b>	保留
[12:0]	<b>Msk</b>	<p><b>标识符掩码28-16</b>            0=报文对象标识符的相应位不用于接收过滤的匹配操作            1=相应的标识符用于接收过滤</p>

CAN\_IFn\_ARB1\_IFn\_仲裁 1 寄存器

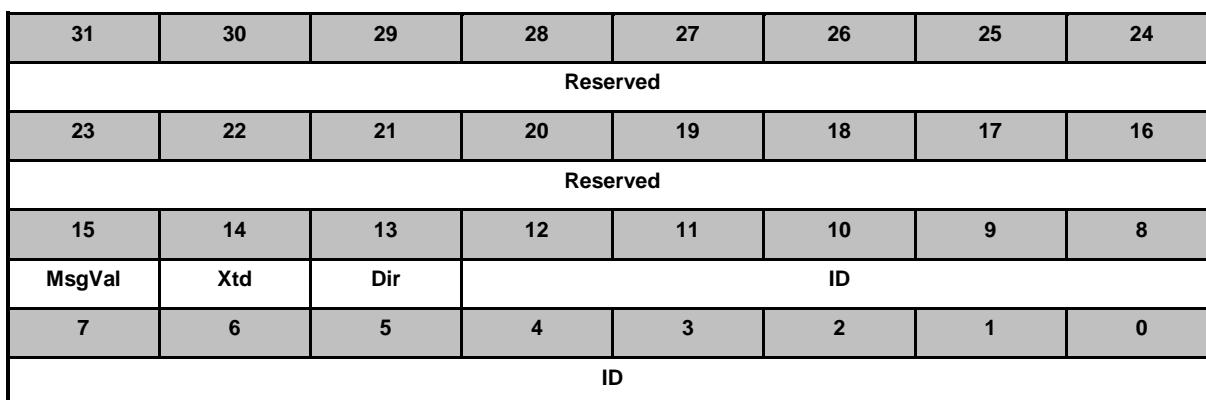
寄存器	偏移量	R/W	描述	复位值
CAN_IFn_ARB1	CAN_BA+0x30 (0x60 *(n-1))	+R/W	IFn 仲裁 1 寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
ID							
7	6	5	4	3	2	1	0
ID							

位	描述	
[31:16]	<b>Reserved</b>	保留
[15:0]	<b>ID</b>	<p><b>报文标识符15-0</b></p> <p>ID28 - ID0, 29-位标识符（“扩展帧”）</p> <p>ID28 - ID18, 11-位标识符（“标准帧”）</p>

## CAN\_IFn\_ARB2\_IFn\_仲裁2寄存器

寄存器	偏移量	R/W	描述	复位值
CAN_IFn_ARB2	CAN_BA+0x34 (0x60 *(n-1))	+R/W	IFn_仲裁2寄存器	0x0000_0000



位	描述	
[31:16]	<b>Reserved</b>	保留
[15]	<b>MsgVal</b>	<p><b>报文有效</b>            0=报文对象无效，报文处理器将忽略此报文对象            1=报文对象有效，报文对象已配置，而且报文处理器将处理此报文对象  <b>注意：</b>在初始化过程中，复位CAN控制寄存器的Init位之前，应用软件必须复位所有未使用的报文对象的MsgVal位。如果报文对象需要修改ID28-0，控制位Xtd，Dir，或数据长度码DLC23-0，那么在ID28-0，控制位Xtd，Dir，或数据长度码DLC23-0修改之前需将此位复位。</p>
[14]	<b>Xtd</b>	<p><b>扩展标识符：</b>            0 = 11-位（标准）标识符用于报文对象            1 = 29-位（扩展）标识符用于报文对象</p>
[13]	<b>Dir</b>	<p><b>报文方向：</b>            1=方向为发送            TxRqst 端，各自的报文对象作为数据帧被传输。在收到带有匹配标识符的远程帧时，该报文对象的TxRqst位被置位。（如果RmtEn=1）            0=方向为接收            TxRqst端，带有该报文对象标识符的远程帧被传输。在收到带有匹配标识符的数据帧时，报文保存在该报文对象内</p>
[12:0]	<b>ID</b>	<p><b>报文标识符28-16</b>            ID28 - ID0, 29位标识符（“扩展帧”）            ID28 - ID18, 11位标识符（“标准帧”）</p>

CAN\_IFn\_MCON\_IFn报文控制寄存器

寄存器	偏移量	R/W	描述	复位值
CAN_IFn_MCON	CAN_BA+0x38 (0x60 *(n-1))	+R/W	IFn 报文控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
NewDat	MsgLst	IntPnd	UMask	TxE	RxE	RmtEn	TxRqst
7	6	5	4	3	2	1	0
EoB	Reserved			DLC			

位	描述	
[31:16]	Reserved	保留
[15]	NewDat	<b>新数据</b> 0=自上次该位被应用软件清除后，该报文对象的数据部分还没有由报文处理器写入新数据。 1=报文处理器或应用软件已经写了新数据到该报文对象的数据部分。
[14]	MsgLst	<b>报文丢失（只对报文对象方向 =接收 有效）</b> 0=自上次该位被CPU复位后无报文丢失 1=当报文处理器保存一个新的报文到报文对象，NewDat虽然被置位，但CPU已经丢失了一条报文。
[13]	IntPnd	<b>中断挂起</b> 0=该报文对象不是中断源 1=该报文对象是中断源。如果没有更高优先级的中断源存在时，中断寄存器中的中断标识符将指向该报文对象。
[12]	UMask	<b>使用验收掩码</b> 0=忽略掩码 1=使用掩码(Msk28-0, MXtd, and MDir)用于接收过滤 <b>注意：</b> 如果UMask位被置1，则在报文对象初始化的过程中，MsgVal(CAN_IFn_ARB2[15])位被置为1之前，报文对象的掩码位必须先被设定。
[11]	TxE	<b>发送中断使能</b> 0= IntPnd (CAN_IFn_MCON[13])在成功传送一帧后将保持不变 1= IntPnd (CAN_IFn_MCON[13])在成功传送一帧后将被置位
[10]	RxE	<b>接收中断使能</b> 0= IntPnd (CAN_IFn_MCON[13])在成功接收一帧后将保持不变 1= IntPnd (CAN_IFn_MCON[13])在成功接收一帧后将被置位

[9]	<b>RmtEn</b>	<b>远程使能</b> 0=在收到一远程帧后，TxRqst保持不变 1=在收到一远程帧后，TxRqst被置位
[8]	<b>TxRqst</b>	<b>传送请求</b> 0=该报文对象不在等待传送 1=该报文对象已请求传送但还没有完成
[7]	<b>EoB</b>	<b>缓存结束</b> 0=报文对象属于某FIFO缓存，但不是该FIFO缓存的最后一个报文对象。 1=单报文对象或FIFO缓存的最后一个报文对象。 <b>注意：</b> 该位用于关联两个或多个 报文对象（最多32 个）组成一个FIFO 缓存。对于单个报文对象（不属于任何FIFO 缓存），该位必须始终设置为 1。
[6:4]	<b>Reserved</b>	保留.
[3:0]	<b>DLC</b>	<b>数据长度码</b> 0-8: 数据帧有0-8个数据字节 9-15: 数据帧有8个数据字节 <b>注意：</b> 报文对象的数据长度码必须和其他节点中所有带有相同标识符的报文对象的数据长度码相同。当报文处理器保存一个数据帧时，它将把收到报文提供的数值写入DLC。 Data 0: CAN数据帧的1st数据字节 Data 1: CAN数据帧的2nd数据字节 Data 2: CAN数据帧的3rd数据字节 Data 3: CAN数据帧的4th数据字节 Data 4: CAN数据帧的5th数据字节 Data 5: CAN数据帧的6th数据字节 Data 6: CAN数据帧的7th数据字节 Data 7: CAN数据帧的8th数据字节 <b>注意：</b> Data 0字节是接收过程中，移入CAN内核的移位寄存器的第一个数据字节，Data 7字节是最后一个。当报文处理器保存一个数据帧，他将会写所有8个数据字节到报文对象中。如果数据长度码小于8，则报文对象剩下的字节将被非指定值覆盖。

CAN\_IFn\_DAT\_A1\_IFn 数据 A1 寄存器

寄存器	偏移量	R/W	描述	复位值
CAN_IFn_DAT_A1	CAN_BA+0x3C (0x60 *(n-1))	+R/W	IFn 数据 A1 寄存器 (寄存器表备注3)	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Data(1)							
7	6	5	4	3	2	1	0
Data(0)							

位	描述	
[31:16]	<b>Reserved</b>	保留
[15:8]	<b>Data(1)</b>	数据字节1 CAN 数据帧的第2个数据字节
[7:0]	<b>Data(0)</b>	数据字节0 CAN 数据帧的第1个数据字节

CAN\_IFn\_DAT\_A2\_IFn 数据 A2 寄存器

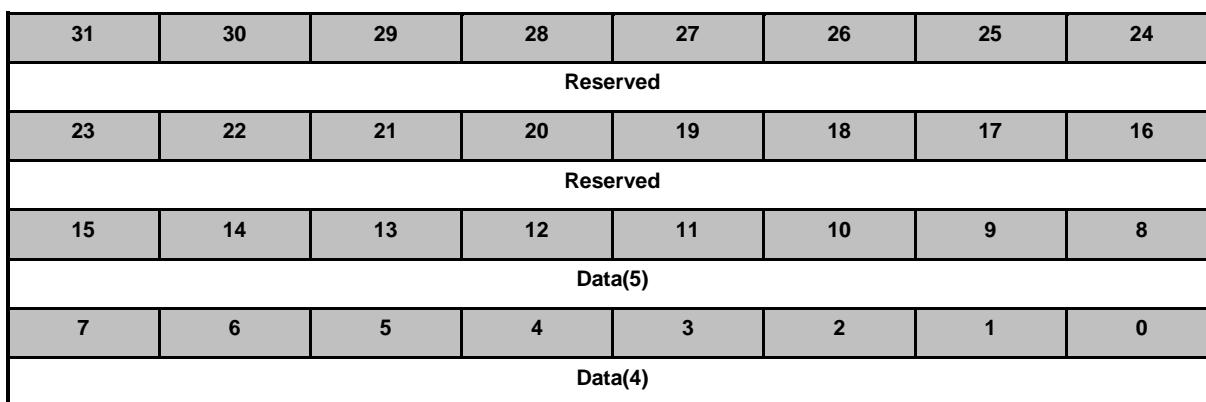
寄存器	偏移量	R/W	描述	复位值
CAN_IFn_DAT_A2	CAN_BA+0x40 (0x60 *(n-1))	+R/W	IFn 数据 A2 寄存器 (寄存器表备注3)	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Data(3)							
7	6	5	4	3	2	1	0
Data(2)							

位	描述	
[31:16]	Reserved	保留
[15:8]	Data(3)	数据字节3 CAN 数据帧的第4个数据字节
[7:0]	Data(2)	数据字节2 CAN 数据帧的第3个数据字节

CAN\_IFn\_DAT\_B1\_IFn 数据 B1 寄存器

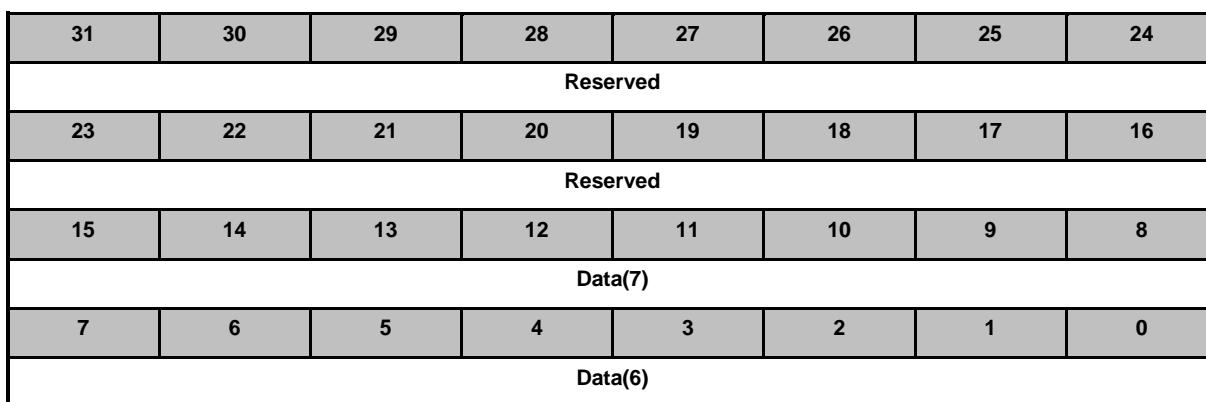
寄存器	偏移量	R/W	描述	复位值
CAN_IFn_DAT_B1	CAN_BA+0x44 (0x60 *(n-1))	+R/W	IFn 数据 B1 寄存器 (寄存器表备注3)	0x0000_0000



位	描述	
[31:16]	Reserved	保留
[15:8]	Data(5)	数据字节5 CAN 数据帧的第6个数据字节
[7:0]	Data(4)	数据字节4 CAN 数据帧的第5个数据字节

**CAN\_IFn\_DAT\_B2 IFn 数据 B2 寄存器**

寄存器	偏移量	R/W	描述	复位值
CAN_IFn_DAT_B2	CAN_BA+0x48 (0x60 *(n-1))	+R/W	IFn数据B2 寄存器(寄存器表备注 3)	0x0000_0000



位	描述	
[31:16]	Reserved	保留
[15:8]	Data(7)	数据字节7 CAN 数据帧的第8个数据字节
[7:0]	Data(6)	数据字节6 CAN 数据帧的第7个数据字节

在一个CAN数据帧中，数据[0]是第一个，数据[7]是传送或接收的最后一个字节数据。在CAN的串行位流，每个字节的MSB将被先传送。

## 报文内存中的报文对象

在报文RAM中有32个报文对象。为了避免应用软件在访问报文RAM时与CAN报文接收发送发生冲突，CPU不能直接访问报文对象，所有访问读取都要通过IFn接口寄存器。下表详细列出了一个报文对象的结构

报文对象												
UMask	Msk [28:0]	MXtd	MDir	EoB	NewDat		MsgLst	RxlE	TxlE	IntPnd	RmtEn	TxRqst
MsgVal	ID [28:0]	Xtd	Dir	DLC [3:0]	Data(0)	Data(1)	Data(2)	Data(3)	Data(4)	Data(5)	Data(6)	Data(7)

表 6.27-8 报文对象在内存中的结构

仲裁寄存器ID28-0(CAN\_IFn\_ARB1/2), Xtd (CAN\_IFn\_ARB2[14])和Dir(CAN\_IFn\_ARB2[13])用于定义标识符，传出报文的传送类型以及传入报文的接收过滤（接收过滤还要用到屏蔽寄存器Msk28-0(CAN\_IFn\_MASK1/2), MXtd (CAN\_IFn\_MASK2[15]), 和 MDir (CAN\_IFn\_MASK2[14])）。接收到的报文存储在对应标识符匹配，方向=接收（数据帧）或方向=发送（远程帧）的有效报文对象中。扩展帧只能保存在Xtd=1的报文对象中，标准帧保存在Xtd=0的报文对象中。如果一条接收到的报文（数据帧或远程帧）和1个以上的有效报文对象匹配，那么它将被保存在报文编号最低的报文对象中

## 报文处理寄存器

所有的报文处理寄存器都是只读。报文处理寄存器内容（每个报文对象的TxRqst(CAN\_IFn\_MCON[8]), NewDat (CAN\_IFn\_MCON[15]), IntPnd(CAN\_IFn\_MCON[13]), 和 MsgVal (CAN\_IFn\_ARB2[15]), 以及中断标志）为报文处理器FSM提供的状态信息。

**CAN\_TXREQ1 发送请求寄存器 1**

这些寄存器保存32个报文对象的TxRqst位。通过读取TxRqst位，软件可以检查哪个报文对象的传送请求正在挂起。在收到一帧或者成功传送一帧后，应用软件可以通过IFn报文接口寄存器或者报文处理器，可以置位/复位指定报文对象的TxRqst位

寄存器	偏移量	R/W	描述	复位值
CAN_TXREQ1	CAN_BA+0x100	R	发送请求寄存器1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
TxRqst16-9							
7	6	5	4	3	2	1	0
TxRqst8-1							

位	描述	
[31:16]	Reserved	保留
[15:0]	TxRqst16-1	发送请求位16-1（所有报文对象） 0=该报文对象没有等待传送 1=该报文对象已有传送请求但还未完成。 该位只读

CAN\_TXREQ2 发送请求寄存器 2

寄存器	偏移量	R/W	描述	复位值
CAN_TXREQ2	CAN_BA+0x104	R	发送请求寄存器2	0x0000_0000

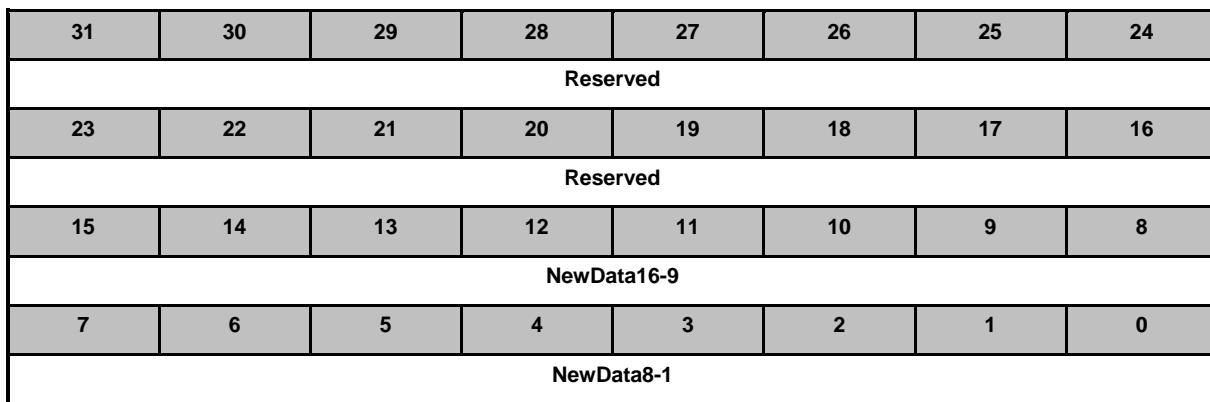
31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
TxRqst32-25							
7	6	5	4	3	2	1	0
TxRqst24-17							

位	描述	
[31:16]	Reserved	保留
[15:0]	TxRqst32-17	<p>发送请求位32-17（所有报文对象）</p> <p>0=该报文对象没有等待传送 1=该报文对象已有传送请求但还未完成。 该位只读</p>

**CAN\_NDAT1 新数据寄存器 1**

这些寄存器保存32个报文对象的NewDat位。通过读取NewDat位，软件可以检查哪个报文对象的数据部分被更新了。在收到一帧或者成功传送一帧后，应用软件可以通过IFn报文接口寄存器或者报文处理器，置位/复位指定报文对象的NewDat位

寄存器	偏移量	R/W	描述	复位值
CAN_NDAT1	CAN_BA+0x120	R	新数据寄存器 1	0x0000_0000



位	描述	
[31:16]	Reserved	保留
[15:0]	NewData16-1	新数据位 16-1 (所有 报文对象) 0=自上次该标志被软件清除后，报文处理器没有写入新数据到该报文对象的数据部分 1=报文处理器或应用软件已经写入了新数据到该报文对象的数据部分

**CAN\_NDAT2 新数据寄存器 2**

寄存器	偏移量	R/W	描述	复位值
CAN_NDAT2	CAN_BA+0x124	R	新数据寄存器2	0x0000_0000

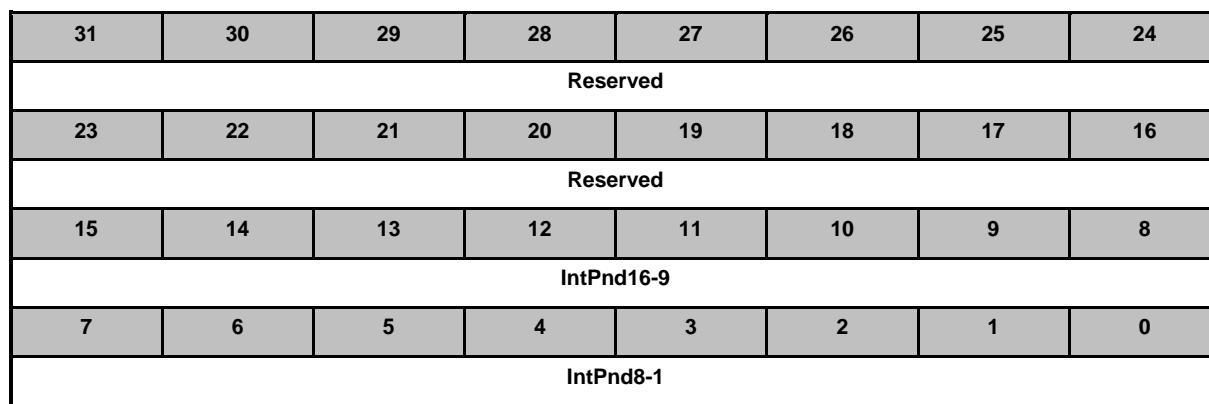
31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
NewData32-25							
7	6	5	4	3	2	1	0
NewData24-17							

位	描述	
[31:16]	Reserved	保留
[15:0]	NewData32-17	<p><b>新数据位 32-17 (所有 报文对象)</b></p> <p>0=自上次该标志被软件清除后，报文处理器没有写入新数据到该报文对象的数据部分 1=报文处理器或应用软件已经写入了新数据到该报文对象的数据部分</p>

**CAN\_IPND1 中断挂起寄存器1**

这些寄存器保存32个报文对象的IntPnd位。通过读取IntPnd位，软件可以检查哪个报文对象的中断挂起。在收到一帧或者成功传送一帧后，应用软件可以通过IFn报文接口寄存器或者报文处理器，置位/复位指定报文对象的IntPnd位。这也也将会影响中断寄存器的IntId的值。

寄存器	偏移量	R/W	描述	复位值
CAN_IPND1	CAN_BA+0x140	R	中断挂起寄存器1	0x0000_0000



位	描述	
[31:16]	Reserved	保留
[15:0]	IntPnd16-1	中断挂起位16-1(所有报文对象) 0=表示该报文对象不是中断源 1=表示该报文对象是中断源

CAN\_IPND2 中断挂起寄存器2

寄存器	偏移量	R/W	描述	复位值
CAN_IPND2	CAN_BA+0x144	R	中断挂起寄存器 2	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
IntPnd32-25							
7	6	5	4	3	2	1	0
IntPnd24-17							

位	描述	
[31:16]	Reserved	保留
[15:0]	IntPnd32-17	<p>中断挂起位32-17(所有报文对象) 0=表示该报文对象不是中断源 1=表示该报文对象是中断源</p>

**CAN\_MVLD1 报文有效寄存器1**

这些寄存器保存32个报文对象的MsgVal位。通过读取MsgVal位，应用软件可以检查哪个报文对象是有效的。应用软件（通过IFn报文接口寄存器）可以来置位/复位指定报文对象的MsgVal位

寄存器	偏移量	R/W	描述	复位值
CAN_MVLD1	CAN_BA+0x160	R	报文有效寄存器1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
MsgVal16- 9							
7	6	5	4	3	2	1	0
MsgVal8-1							

位	描述	
[31:16]	Reserved	配置
[15:0]	MsgVal16-1	<p>报文有效位16-1（所有报文对象）（只读）</p> <p>0= 该报文对象无效, 被报文处理器忽略</p> <p>1= 该报文对象有效, 可以被报文处理器考虑</p> <p>例如: .CAN_MVLD1[0]表示No.1对象是否有效, 如果CAN_MVLD1[0]被置位, 则报文对象No.1有效。</p>

**CAN\_MVLD2 报文有效寄存器2**

寄存器	偏移量	R/W	描述	复位值
CAN_MVLD2	CAN_BA+0x164	R	报文有效寄存器2	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
MsgVal32-25							
7	6	5	4	3	2	1	0
MsgVal24-17							

位	描述	
[31:16]	<b>Reserved</b>	保留
[15:0]	<b>MsgVal32-17</b>	<p>报文有效位32-17（所有报文对象）（只读）</p> <p>0= 该报文对象无效，被报文处理器忽略 1= 该报文对象有效，可以被报文处理器考虑</p> <p>例如：.CAN_MVLD2[15]表示No.32对象是否有效，如果CAN_MVLD2[15]被置位，则报文对象No.32为有效</p>

CAN\_WU\_EN 唤醒使能寄存器

寄存器	偏移量	R/W	描述	复位值
CAN_WU_EN	CAN_BA+0x168	R/W	唤醒使能寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved							WAKUP_EN

位	描述	
[31:1]	Reserved	保留
[0]	WAKUP_EN	<p>唤醒使能位 0 = 禁止唤醒功能 1 = 使能唤醒功能 注: 当CAN_Rx管脚上有下降沿信号时, 用户可唤醒系统</p>

CAN\_WU\_STATUS 唤醒状态寄存器

寄存器	偏移量	R/W	描述	复位值
CAN_WU_STATUS	CAN_BA+0x16C	R/W	唤醒状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved							WAKUP_STS

位	描述	
[31:1]	Reserved	保留
[0]	WAKUP_STS	<p>唤醒状态 0= 没有唤醒事件发生 1= 有唤醒事件发生了 注: 该位可写'0'清除</p>

## 6.28 SD 卡主机接口

### 6.28.1 概述

安全数字主机控制器(SD 主机)包含 DMAC 单元和 SD 单元。DMAC 单元提供SD共享缓存（128个字节）与系统内存间的交换数据DMA（直接内存访问）功能。SD 单元控制SD/SDHC的接口。SDHOST 控制器支持SD/SDHC 并且有 DMAC 提供系统内存与卡之间快速的数据传输。

### 6.28.2 特性

- 连接到 AMBA AHB 主/从 接口， 提供数据传输和寄存器读写。
- 支持单通道 DMA
- 支持硬件 分散-收集功能
- 使用一个128 字节共享缓存用做系统和卡之间的数据交换
- 单一时钟域AHB总线时钟 (HCLK)的同步设计DMA
- 寄存器读写与数据传输的DMAC 接口
- 支持 SD/SDHC 卡.
- 两个时钟域完全异步设计， HCLK和引擎时钟， 注意HCLK的频率需要比外设时钟快

### 6.28.3 框图

SDHC的框图和焊盘分配如下面所示：

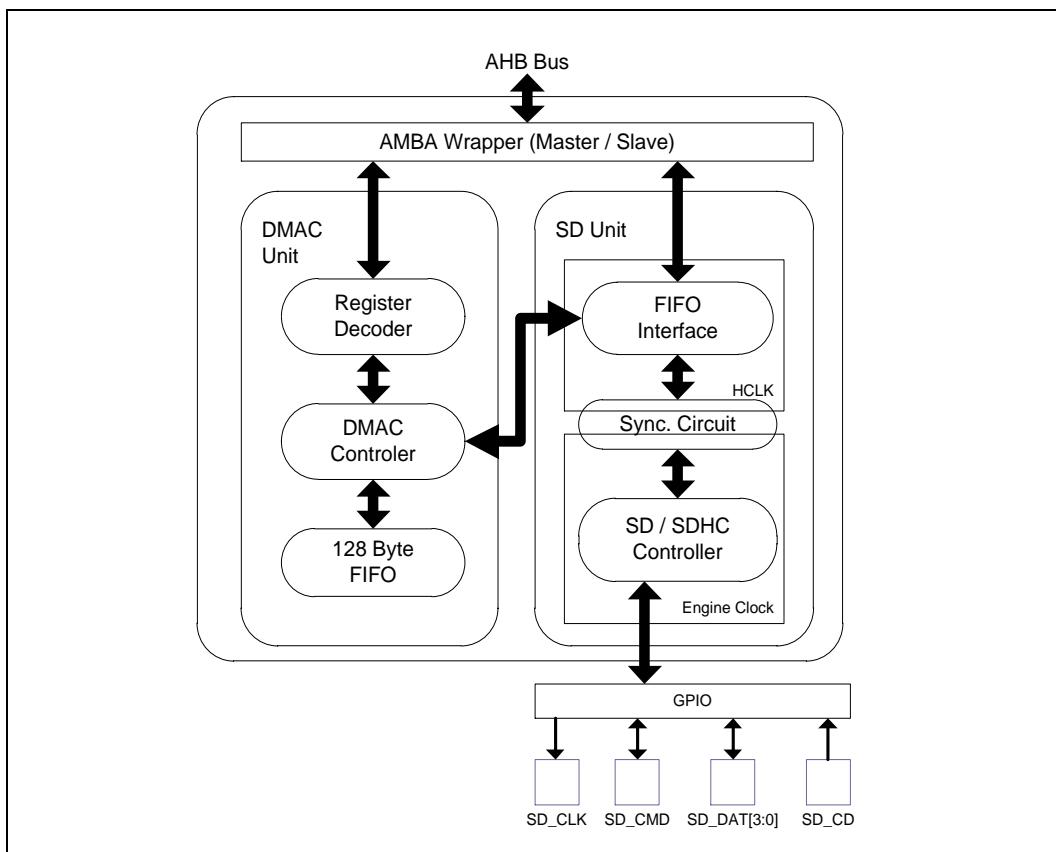


图 6.28-1 SD 主机控制器框图

#### 6.28.4 基本配置

##### 6.28.4.1 SD0 基本配置

- 时钟源配置
  - 选择SD0时钟源的控制位是 SDH0SEL (CLK\_CLKSEL0[21:20])
  - 选择SD0时钟分频的控制位是 SDH0DIV (CLK\_CLKDIV0[31:24])
  - 使能SD0时钟的控制位是 SDH0CKEN (CLK\_AHBCLK[6])
- 复位配置
  - 复位SD0控制器的控制位是 SDH0RST (SYS\_IPRST0[6])
- 引脚配置

组	引脚名	GPIO	MFP
SD0	SD0_CLK	PB.1, PE.6	MFP3
	SD0_CMD	PB.0, PE.7	MFP3
	SD0_DAT0	PB.2, PE.2	MFP3
	SD0_DAT1	PB.3, PE.3	MFP3
	SD0_DAT2	PB.4, PE.4	MFP3
	SD0_DAT3	PB.5, PE.5	MFP3
	SD0_nCD	PD.13	MFP3
		PB.12	MFP9

表 6.28-1 SD0 引脚配置

##### 6.28.4.2 SD1 基本配置

- 时钟源配置
  - 选择SD1时钟源的控制位是 SDH1SEL (CLK\_CLKSEL0[23:22])
  - 选择SD1时钟分频的控制位是 SDH1DIV (CLK\_CLKDIV3[31:24])
  - 使能SD1时钟的控制位是 SDH1CKEN (CLK\_AHBCLK[17])
- 复位配置
  - 复位SD1控制器的控制位是 SDH1RST (SYS\_IPRST0[17])
- 引脚配置

组	引脚名	GPIO	MFP
SD1	SD1_CLK	PG.14	MFP3
		PA.4	MFP5
		PB.6	MFP7
	SD1_CMD	PG.13	MFP3
		PA.5	MFP5

	PB.7	MFP7
SD1_DAT0	PG.12	MFP3
	PA.0, PA.8	MFP5
SD1_DAT1	PG.11	MFP3
	PA.1, PA.9	MFP5
SD1_DAT2	PG.10	MFP3
	PA.2, PA.10	MFP5
SD1_DAT3	PG.9	MFP3
	PA.3, PA.11	MFP5
SD1_nCD	PG.15	MFP3
	PA.6, PE.14	MFP5

表 6.28-2 SD1 引脚配置

### 6.28.5 功能描述

SD 主机提供访问 SD/SDHC 卡的接口，该接口提供 1 位和 4 位数据总线模式。

SD 控制器使用一个独立的时钟源，SDCLK，作为引擎时钟。SDCLK 可以与系统时钟 HCLK 完全异步。程序可以任意改变 SD 时钟，注意 HCLK 需要快于 SDCLK。

#### 6.28.5.1 基本操作

SD 控制器可以产生所有类型的 48 位的指令到 SD 卡，并且从 SD 卡获取所有类型的响应。收到响应后，响应的内容存入 SDRSP0 和 SDRSP1。SD 控制器会计算 CRC7 确认响应的正确性。如果 CRC7 错误，CDRIF (SDH\_INTSTS[1]) 会置 1 且 CRC7 (SDH\_INTSTS[2]) 会清 0。对于响应 R1b，程序需要注意在收到响应后 SD 卡会在数据线 DAT0 产生忙信号；程序可以通过时钟轮询检查这个状态直到它变高。对于响应 R3，CRC7 无效；但是 SD 控制器仍然会计算 CRC7 并得到错误的结果，程序要忽略这个错误并清除 CDRIF 标志 (SDH\_INTSTS[1])。

SD 控制器包含 2 个状态机，指令/响应部分和数据部分。对于指令/响应部分，触发位是 SDH\_CTL 寄存器的 COEN, RIEN, R2\_EN, CLK74OE 和 CLK8\_OE 位。如果程序使能所有这些位，处理优先级会是 CLK74OE > COEN > RIEN/R2\_EN > CLK8\_OE，注意 RIEN 和 R2\_EN 不能同时触发。对于数据部分，有 DIEN 和 DOEN 可以选择。程序只能同时触发其中之一。如果 DIEN 被触发，SD 控制器会立即从 DAT0 数据线等待开始位，然后从 SD 卡得到特定数量的数据。收到数据后，SD 控制器会检查 CRC16 的正确性；如果有错误，CDRIF (SDH\_INTSTS[1]) 会置 1 且 CRC16 (SDH\_INTSTS[3]) 会清 0。如果 DOEN 被触发，SD 控制器会等待响应结束，然后发送特定数据到 SD 卡。数据发送之后 SD 控制器会从 SD 卡得到 CRC 状态并确认正确性；CRC 需要是 ‘010’，否则 CDRIF (SDH\_INTSTS[1]) 会置 1 且 CRCSTAT (SDH\_INTSTS[6:4]) 会是接收到的数值。如果 IR2\_EN 被触发，SD 控制器会从 SD 卡接收响应 R2 (136 位)，CRC7 和结束位会丢弃。接收到的数据会放入 DMA 缓存，从地址偏移 0x0 开始。

#### 6.28.5.2 多数据块传输

SD 控制器提供多数据块传输功能（改变 BLKLEN 来改变数据块长度）。程序可以用这个功能来加速数据传输。如果 CRC7, CRC16 或 CRC 状态错误，SD 控制器会结束传输且置位 CDRIF (SDH\_INTSTS[1])，程序在这种状况发生时复位引擎。

在SD引擎里有一个硬件响应超时机制。程序可以设置一个24位超时值TOUT, SD控制器会根据这个值来决定什么时候超时。

#### 6.28.5.3 DMA控制器

SD 主机 DMA 控制器在SD主机控制器和系统内存 (SRAM) 和共享缓存 (128个字节) 之间提供DMA (直接内存访问) 功能。SD主机间的DMA请求的仲裁由DMA总线管理器控制。程序只需要简单的填入开始地址和使能DMA, 然后让DMA来自动控制数据传输。

DMA有128个字节的共享缓存, 可以为SD主机提供多模块的数据传输。程序可以在SD主机不忙时直接访问缓存。

#### 6.28.5.4 编程流程

这是一个简单的没有使能分散采集的DMA的编程例子。

1. 设置 DMAEN (SDH\_DMACTL[0])使能 DMA.
2. 在 SD 主机的 SDH\_DMASA 填入相应的开始地址。
3. 触发 SD 主机开始 DMA 传输
4. 等待传输完成.

这是一个使能了分散采集的DMA编程示例:

1. 设置 DMAEN (SDH\_DMACTL[0]) 使能 DMA (SDH\_DMACTL[0]) 且设置 SGEN (SDH\_DMACTL[3])使能分散采集功能。
2. 在 SD 主机的 SDH\_DMASA 填写物理地址描述符 (PAD) 表的开始地址。
3. 当 SDH\_DMASA 的第0位为1, 不会按顺序读取 PAD 列表, 如果为0, 会从 PAD 按序载入描述符。在这种功能下第一次在寄存器里写第0位是无效的, 这一位要在 PAD 列表中设置。
4. 触发 SD 主机开始 DMA 传输
5. 等待传输完成

PAD 表格式如图 6.28-2所示。注意所有PAD的总共字节计数必须等于填写到SD主机的字节计数。EOF 在最后的描述符中必须设置为1。

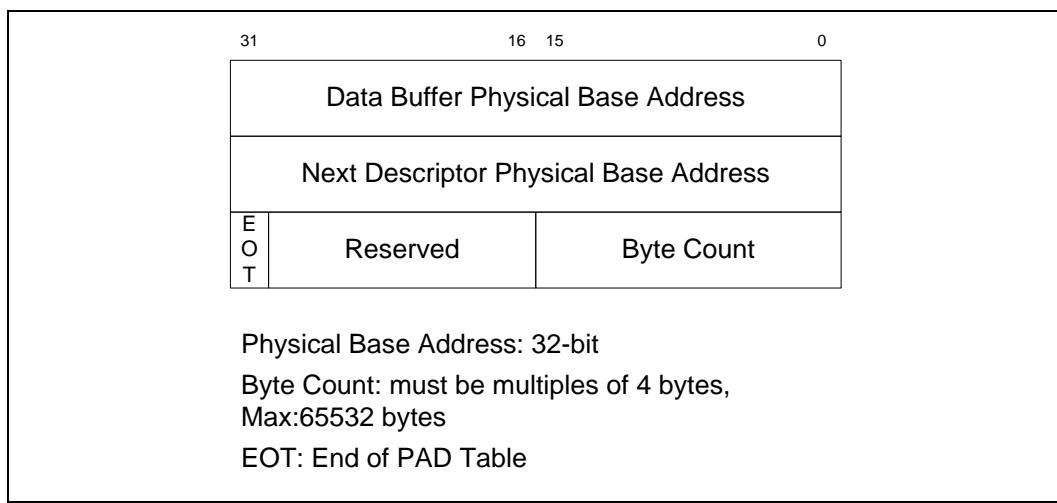


图 6.28-2 PAD (物理地址描述符) 表格式

## 6.28.6 寄存器映射

R: 只读, W: 只写, R/W: 可读写

寄存器	偏移量	R/W	描述	复位值
<b>SDH 基地址:</b>				
<b>SDH0_BA = 0x4000_D000</b>				
<b>SDH1_BA = 0x4000_E000</b>				
<b>SDH_FB_n</b> $x=0,1$ $n=0,1..31$	SDHx_BA+0x000 + 0x4 * n	R/W	共享缓存 (FIFO)	0x0000_0000
<b>SDH_DMACTL</b> $x=0,1$	SDHx_BA+0x400	R/W	DMA 控制和状态寄存器	0x0000_0000
<b>SDH_DMASA</b> $x=0,1$	SDHx_BA+0x408	R/W	DMA 传输开始地址寄存器	0x0000_0000
<b>SDH_DMABCNT</b> $x=0,1$	SDHx_BA+0x40C	R	DMA 传输字节数寄存器	0x0000_0000
<b>SDH_DMAINTEN</b> $x=0,1$	SDHx_BA+0x410	R/W	DMA 中断使能控制寄存器	0x0000_0001
<b>SDH_DMAINTSTS</b> $x=0,1$	SDHx_BA+0x414	R/W	DMA 中断状态寄存器	0x0000_0000
<b>SDH_GCTL</b> $x=0,1$	SDHx_BA+0x800	R/W	全局控制和状态寄存器	0x0000_0000
<b>SDH_GINTEN</b> $x=0,1$	SDHx_BA+0x804	R/W	全局中断控制寄存器	0x0000_0001
<b>SDH_GINTSTS</b> $x=0,1$	SDHx_BA+0x808	R/W	全局中断状态寄存器	0x0000_0000
<b>SDH_CTL</b>	SDHx_BA+0x820	R/W	SD 控制和状态寄存器	0x0101_0000
<b>SDH_CMDARG</b> $x=0,1$	SDHx_BA+0x824	R/W	SD 指令参数寄存器	0x0000_0000
<b>SDH_INTEN</b> $x=0,1$	SDHx_BA+0x828	R/W	SD 中断控制寄存器	0x0000_0A00
<b>SDH_INTSTS</b> $x=0,1$	SDHx_BA+0x82C	R/W	SD 中断状态寄存器	0x000X_008C
<b>SDH_RESP0</b> $x=0,1$	SDHx_BA+0x830	R	SD 接收响应标志寄存器 0	0x0000_0000
<b>SDH_RESP1</b> $x=0,1$	SDHx_BA+0x834	R	SD 接收响应标志寄存器 1	0x0000_0000
<b>SDH_BLEN</b> $x=0,1$	SDHx_BA+0x838	R/W	SD 数据块长度寄存器	0x0000_01FF
<b>SDH_TOUT</b>	SDHx_BA+0x83C	R/W	SD 响应或数据输入超时寄存器	0x0000_0000

x=0,1				
-------	--	--	--	--

### 6.28.7 寄存器描述

#### SDH\_DMACTL DMA 控制和状态寄存器

寄存器	偏移量	R/W	描述	复位值
SDH_DMACTL	SDHx_BA+0x400	R/W	DMA控制和状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved						DMABUSY	Reserved
7	6	5	4	3	2	1	0
Reserved				SGEN	Reserved	DMARST	DMAEN

位	描述	
[31:10]	<b>Reserved</b>	保留
[9]	<b>DMABUSY</b>	<b>DMA 传输正在进行</b> 这一位指示SD主机被允许且正在进行DMA传输 0 = DMA传输不在进行 1 = DMA传输正在进行
[8:4]	<b>Reserved</b>	保留
[3]	<b>SGEN</b>	<b>分散采集功能使能位</b> 0 = 禁用分散采集功能 (DMA 会把DMASAR 里的起始地址作为一个单一内存块的开始指针) 1 = 使能分散采集功能 (DMA 会把 DMASAR 里的起始地址作为物理地址描述符 (PAD) 列表的起始地址, PAD的格式会在之后描述)
[2]	<b>Reserved</b>	保留
[1]	<b>DMARST</b>	<b>软件引擎复位</b> 0 = 没效果 1 = 复位内部状态机和指针。控制寄存器的值不会被清除。这一位会在几个时钟周期后自动清0。 <b>注意:</b> 程序复位DMA相关寄存器
[0]	<b>DMAEN</b>	<b>DMA 引擎使能位</b> 0 = DMA 禁止. 1 = DMA 使能. 如果这一位为0, DMA会忽略所有SD主机的请求, 强制总线管理器进入空闲状态。

	注意: 如果发送目标异常, DMAEN 会清0.
--	--------------------------

**SDH\_DMASA DMA 传输开始地址寄存器**

寄存器	偏移量	R/W	描述	复位值
SDH_DMASA	SDHx_BA+0x408	R/W	DMA 传输开始地址寄存器	0x0000_0000

31	30	29	28	27	26	25	24
DMASA							
23	22	21	20	19	18	17	16
DMASA							
15	14	13	12	11	10	9	8
DMASA							
7	6	5	4	3	2	1	0
DMASA							ORDER

位	描述	
[31:1]	DMASA	<b>DMA 传输开始地址</b> 这个区域是PADO的起始地址指示32位的系统内存SRAM的起始地址，用作DMA发送或接收数据。如果 DMA 不在正常工作模式，这个区域指向物理地址描述符（PAD）列表的起始地址。 <b>注意:</b> SRAM的起始地址必须字对齐，例如0x0000_0000, 0x0000_0004...
[0]	ORDER	<b>决定PAD列表是否按顺序读取</b> 0 = PAD 列表按顺序读取 1 = PAD 列表不按顺序读取 <b>注意:</b> 分散采集模式下当SGEN = 1时bit0有效

**SDH\_DMABCNT DMA 传输字节数寄存器**

寄存器	偏移量	R/W	描述	复位值
SDH_DMABCNT	SDHx_BA+0x40C	R	DMA 传输字节数寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved						BCNT	
23	22	21	20	19	18	17	16
BCNT							
15	14	13	12	11	10	9	8
BCNT							
7	6	5	4	3	2	1	0
BCNT							

位	描述	
[31:26]	Reserved	保留
[25:0]	BCNT	<b>DMA 传输字节数(只读)</b> 这个区域显示DMA传输剩下的字节数。这个值只在DMA忙碌时有效，其他情况，这个值是0。

**SDH DMAINTEN DMA 中断使能控制寄存器**

寄存器	偏移量	R/W	描述	复位值
SDH_DMAINTEN	SDHx_BA+0x410	R/W	DMA 中断使能控制寄存器	0x0000_0001

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved						WEOTIEN	ABORTIEN

位	描述	
[31:2]	Reserved	保留
[1]	WEOTIEN	<p>遇到错误EOT中断使能位 0 = 当遇到错误EOT时不产生中断 1 = 当遇到错误EOT时产生中断</p>
[0]	ABORTIEN	<p>DMA读/写目标异常中断使能位 0 = DMA传输时目标异常不产生中断 1 = DMA传输时目标异常产生中断</p>

**SDH DMAINTSTS SDH DMA 中断状态寄存器**

寄存器	偏移量	R/W	描述	复位值
SDH_DMAINTSTS	SDHx_BA+0x414	R/W	SDH DMA中断状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved						WEOTIF	ABORTIF

位	描述	
[31:2]	<b>Reserved</b>	保留
[1]	<b>WEOTIF</b>	<p><b>遇到错误EOT中断标志（只读）</b>  当使能了分散采集DMA功能，在DMA传输完成前，描述符EOT位为1（PAD所有数据个数比SD 主机设定的数目少），这一位会置1。  0 = DMA传输完成前没有遇到EOT位。  1 = DMA传输完成前遇到了EOT位。  <b>注意：</b>此位只读，写1清0</p>
[0]	<b>ABORTIF</b>	<p><b>DMA读/写目标异常中断标志（只读）</b>  0 = 没有接收到总线错误响应  1 = 接收到总线错误响应  <b>注意1：</b>此位只读，写1清0  <b>注意2：</b>当DMA总线控制器收到错误响应，意味着发生了目标异常。DMA 会停止传输响应这个事件，总线进入空闲态。当发生了目标异常或WEOTIF为1，程序需要复位DMA 和 SD 主机，然后重新开始数据传输</p>

## SDH GCTL 全局控制和状态寄存器

寄存器	偏移量	R/W	描述	复位值
SDH_GCTL	SDHx_BA+0x800	R/W	全局控制和状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved						SDEN	GCTRLRST

位	描述	
[31:2]	<b>Reserved</b>	保留
[1]	<b>SDEN</b>	<b>SD功能使能位</b> 0 = SD 功能禁止 1 = SD 功能使能
[0]	<b>GCTRLRST</b>	<b>程序引擎复位</b> 0 = 没有影响 1 = 复位 SD 主机。 控制寄存器不会被清除。此位在Reset完成后自动清0

**SDH\_GINTEN 全局中断控制寄存器**

寄存器	偏移量	R/W	描述	复位值
SDH_GINTEN	SDHx_BA+0x804	R/W	全局中断控制寄存器	0x0000_0001

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved							DTAIEN

位	描述	
[31:1]	Reserved	保留
[0]	DTAIEN	<b>DMA 读/写目标异常中断使能位</b> 0 = 禁止DMA 读/写目标异常中断 1 = 使能DMA 读/写目标异常中断

**SDH\_GINTSTS 全局中断状态寄存器**

寄存器	偏移量	R/W	描述	复位值
SDH_GINTSTS	SDHx_BA+0x808	R/W	全局中断状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved							DTAIF

位	描述	
[31:1]	Reserved	保留
[0]	DTAIF	<p><b>DMA 读/写目标异常中断标志(只读)</b></p> <p>这一位指示DMA在读/写操作时接收到从内部AHB总线上收到错误响应。当发生了目标异常，请复位所有的引擎。</p> <p>0 = 没有收到总线错误响应 1 = 接收到总线错误响应</p> <p><b>注意:</b> 此位只读，写1清0</p>

## SDH CTL SD 控制和状态寄存器

寄存器	偏移量	R/W	描述	复位值
SDH_CTL	SDHx_BA+0x820	R/W	SD 控制和状态寄存器	0x0101_0000

31	30	29	28	27	26	25	24
Reserved				SDNWR			
23	22	21	20	19	18	17	16
BLKCNT							
15	14	13	12	11	10	9	8
DBW	CTLRST	CMDCODE					
7	6	5	4	3	2	1	0
CLKKEEP	CLK8OEN	CLK74OEN	R2EN	DOEN	DIEN	RIEN	COEN

位	描述	
[31: 28]	<b>Reserved</b>	保留
[27:24]	<b>SDNWR</b>	<b>数据块写操作写信号参数</b> 这个值是数据块写操作时NWR参数，以SD时钟为单位，实际值是SDNWR+1.
[23:16]	<b>BLKCNT</b>	<b>需要发送或接收的数据块个数</b> 这个区域包含输入输出的数据块个数。用于READ_MULTIPLE_BLOCK 和 WRITE_MULTIPLE_BLOCK 指令，程序可以使用这个功能加速数据传输提升性能。不要在这里填0x0。 <b>注意：</b> 对于READ_MULTIPLE_BLOCK 和 WRITE_MULTIPLE_BLOCK 指令，实际长度是BLKCNT * (BLKLEN +1).
[15]	<b>DBW</b>	<b>SD 数据总线宽度 (1线/4线选择)</b> 0 = 数据总线宽度是1位 1 = 数据总线宽度是4位
[14]	<b>CTLRST</b>	<b>软件引擎复位</b> 0 = 没影响 1 = 复位内部状态机和计数器。控制寄存器的值不会清除(但是 RIEN, DIEN, DOEN 和 R2_EN 位会清0). 这一位会在几个时钟后自动清0.
[13:8]	<b>CMDCODE</b>	<b>SD 指令码</b> T这个寄存器存放 SD 指令码 (0x00 – 0x3F).
[7]	<b>CLKKEEP</b>	<b>SD 端口时钟使能位</b> 0 = SD 主机自动决定什么时候输出时钟，什么时候停止输出 1 = SD 时钟总是输出

[6]	<b>CLK8OEN</b>	<b>产生8个时钟周期输出使能位</b> 0 = 没效果 (请通过 DMARST (SDH_CTL [0]) 清这一位) 1 = 使能 SD 主机输出8个时钟周期 <b>注意:</b> 操作结束后, 此位自动清0, 不要写0到这个寄存器, 否则控制器会工作不正常
[5]	<b>CLK74OEN</b>	<b>初始化 74 个时钟周期输出使能位</b> 0 = 没效果 (请通过 DMARST (SDH_CTL [0]) 清这一位) 1 = 使能, SD 主机会输出74个时钟周期到SD卡。 <b>注意:</b> 操作结束后, 此位自动清0, 不要写0到这个寄存器, 否则控制器会工作不正常
[4]	<b>R2EN</b>	<b>响应 R2 输入使能位</b> 0 = 没效果 (请通过 DMARST (SDH_CTL [0]) 清这一位) 1 = 使能, SD 主机会等待从SD卡接收R2响应, 并且存储响应数据到DMC的Flash缓存 (不包含CRC7) <b>注意:</b> 操作结束后, 此位自动清0, 不要写0到这个寄存器, 否则控制器会工作不正常
[3]	<b>DOEN</b>	<b>数据输出使能位</b> 0 = 没效果 (请通过 DMARST (SDH_CTL [0]) 清这一位) 1 = 使能, SD 主机会传输数据块和CRC16值到SD卡 <b>注意:</b> 操作结束后, 此位自动清0, 不要写0到这个寄存器, 否则控制器会工作不正常
[2]	<b>DIEN</b>	<b>数据输入使能位</b> 0 = 没效果 (请通过 DMARST (SDH_CTL [0]) 清这一位) 1 = 使能, SD 主机会等待从SD卡接收CRC16值 <b>注意:</b> 操作结束后, 此位自动清0, 不要写0到这个寄存器, 否则控制器会工作不正常
[1]	<b>RIEN</b>	<b>响应输入使能位</b> 0 = 没效果 (请通过 DMARST (SDH_CTL [0]) 清这一位) 1 = 使能, SD 主机等待从SD卡接收响应 <b>注意:</b> 操作结束后, 此位自动清0, 不要写0到这个寄存器, 否则控制器会工作不正常
[0]	<b>COEN</b>	<b>指令输出使能位</b> 0 = 没效果 (请通过 DMARST (SDH_CTL [0]) 清这一位) 1 = 使能, SD 主机会输出指令到SD卡 <b>注意:</b> 操作结束后, 此位自动清0, 不要写0到这个寄存器, 否则控制器会工作不正常

**SDH\_CMDARG SD 指令参数寄存器**

寄存器	偏移量	R/W	描述	复位值
SDH_CMDARG	SDHx_BA+0x824	R/W	SD指令参数寄存器	0x0000_0000

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

位	描述	
[31:0]	ARGUMENT	SD指令参数 这个寄存器包含从主控制器到SD卡的一个32位SD指令参数值，在触发COEN (SDH_CTL [0])前，程序需要填这个区域。

## SDH INTEN SD 中断控制寄存器

寄存器	偏移量	R/W	描述	复位值
SDH_INTEN	SDHx_BA+0x828	R/W	SD中断控制寄存器	0x0000_0A00

31	30	29	28	27	26	25	24
Reserved	CDSRC	Reserved					
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved	WKIEN	DITOIEN	RTOIEN	Reserved		Reserved	CDIEN
7	6	5	4	3	2	1	0
Reserved						CRCIEN	BLKDIEN

位	描述	
[31]	Reserved	保留
[30]	CDSRC	<b>SD0卡片检测源选择</b> 0 = 从 SD 卡的 DAT3 管脚. 主机需要时钟从DAT3读取数据。为了保证DAT3能产生时钟, 请确定 CLKKEEP(SDH_CTL[7]) 是1 1 = 从GPIO 管脚.
[29:15]	Reserved	保留
[14]	WKIEN	<b>产生唤醒信号使能位</b> 使能或禁止当前SD卡通过DAT1管脚产生的中断（唤醒）信号 0 = 禁止. 1 = 使能.
[13]	DITOIEN	<b>数据输入超时中断使能位</b> 使能或禁止SD控制器输入超时时产生的中断, 超时时间由TOUT 寄存器设置 0 = 禁止. 1 = 使能.
[12]	RTOIEN	<b>响应超时中断使能位</b> 使能或禁止当接收响应或R2超时时SD控制器产生的中断, 超时时间由TOUT 寄存器设置 0 = 禁止. 1 = 使能.
[11:10]	Reserved	保留
[9]	Reserved	保留.

[8]	<b>CDIEN</b>	<b>SD 卡片检测中断使能位</b> 使能或禁止当SD卡片插入或拔出时产生中断 0 = 禁止. 1 = 使能.
[7:2]	<b>Reserved</b>	保留
[1]	<b>CRCIEN</b>	<b>CRC7, CRC16 和CRC状态错误中断使能位</b> 0 = SD 主机当CRC7, CRC16 和 CRC状态错误时不产生中断. 1 = SD 主机当CRC7, CRC16 和 CRC状态错误时产生中断
[0]	<b>BLKDIEN</b>	<b>块传输完成中断使能位</b> 0 = SD 主机当数据输入输出传输完成后不产生中断 1 = SD 主机当数据输入输出传输完成后产生中断.

## SDH INTSTS SD 中断状态寄存器

寄存器	偏移量	R/W	描述	复位值
SDH_INTSTS	SDHx_BA+0x82C	R/W	SD中断状态寄存器	0x000X_008C

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved					DAT1STS	Reserved	CDSTS
15	14	13	12	11	10	9	8
Reserved		DITOIF	RTOIF	Reserved			CDIF
7	6	5	4	3	2	1	0
DAT0STS	CRCSTS			CRC16	CRC7	CRCIF	BLKDIF

位	描述	
[31:19]	Reserved	保留
[18]	DAT1STS	<b>SD端口的DAT1 管脚状态 (只读)</b> 这一位显示SD端口的DAT1 管脚状态
[17]	Reserved	保留
[16]	CDSTS	<b>SD卡片检测状态(只读)</b> 这一位只是SD卡片检测管脚的状态，用来卡片检测。当SD有卡片插入或拔出，程序需要检查这一位来确认卡片是否真的插入或拔出了。 如果 CDSRC (SDH_INTEN[30]) = 0, 选择 DAT3作为卡片检测： 0 = 卡片拔出。 1 = 卡片插入。 如果CDSRC(SDH_INTEN[30]) = 1,选择 GPIO 作为卡片检测： 0 = 卡片拔出。 1 = 卡片插入。
[15:14]	Reserved	保留
[13]	DITOIF	<b>数据输入超时中断标志(只读)</b> 这一位指示SD主机接收数据时发生了超时(等待开始位). 0 = 没有超时. 1 = 数据输入超时. <b>注意:</b> 此位只读，写1清0
[12]	RTOIF	<b>响应超时中断标志(只读)</b> 这一位指示SD主机接收到R2时发生了超时(等待开始位). 0 = 没有超时. 1 = 响应超时. <b>注意:</b> 此位只读，写1清0

[11:9]	<b>Reserved</b>	保留
[8]	<b>CDIF</b>	<p><b>SD卡片检测中断状态标志(只读)</b>            这一位指示SD卡片插入或拔出。只有当 CDIEN(SDH_INTEN[8]) 设为1, 这一位才有效.            0 = 没有卡片插入或拔出            1 = SD有卡片插入或拔出  <b>注意:</b> 此位只读, 写1清0</p>
[7]	<b>DAT0STS</b>	<p><b>目前选择的SD端口的DAT0 管脚状态(只读)</b>            这一位指示目前选择的SD端口的 DAT0 管脚状态</p>
[6:4]	<b>CRCSTS</b>	<p><b>数据输出传输CRC状态值(只读)</b>            SD 主机会记录输出数据的CRC状态值。程序可以用这个值了解数据传输时发生了哪种错误            010 =正CRC 状态.            101 = 负 CRC状态            111 =发生了SD 卡编程错误</p>
[3]	<b>CRC16</b>	<p><b>输入数据传输CRC16 状态值(只读)</b>            SD 主机在数据输入时确认CRC16 正确性.            0 = 错误            1 = 正确</p>
[2]	<b>CRC7</b>	<p><b>CRC7 检测状态(只读)</b>            SD主机在每笔响应输入时会确认CRC7正确性。如果响应不包含 CRC7 信息 (如. R3), 那么程序需要关闭 CRCIEN (SDH_INTEN[1]) 然后忽略这一位。            0 = 错误            1 =正确</p>
[1]	<b>CRCIF</b>	<p><b>CRC7, CRC16 和 CRC 状态错误中断标志(只读)</b>            这一位指示SD 主机在响应输入, 数据输入或数据输出 (CRC状态错误) 传输时发生了CRC错误。如果发生了CRC错误, 程序需要复位SD引擎。有一些响应(例如 R3) 并没有CRC7域, SD 主机仍然会计算CRC7, 得到CRC错误并置位这个标志。在这种情况下程序需要忽略CRC错误并手动清除此位。            0 = 没有 CRC 错误发生            1 =发生了CRC错误  <b>注意:</b> 此位只读, 写1清0</p>
[0]	<b>BLKDIF</b>	<p><b>块传输完成中断标志(只读)</b>            这一位指示SD主机完成了所有数据输入输出的传输。如果在块传输中产生了CRC16错误或不正确的CRC状态, 传输会中断此位也会置1.            0 = 没有完成            1 = 完成了  <b>注意:</b> 此位只读, 写1清0</p>

**SDH\_RESP0 SD 接收响应标志寄存器 0**

寄存器	偏移量	R/W	描述	复位值
SDH_RESP0	SDHx_BA+0x830	R	SD接收响应标志寄存器 0	0x0000_0000

31	30	29	28	27	26	25	24
RESPTK0							
23	22	21	20	19	18	17	16
RESPTK0							
15	14	13	12	11	10	9	8
RESPTK0							
7	6	5	4	3	2	1	0
RESPTK0							

位	描述	
[31:0]	RESPTK0	<b>SD 接收响应标志寄存器 0</b> SD 主机控制器会当RIEN (SDH_CTL[1])为1时接收响应标志从SD卡得到回复。这个区域包含响应标志的响应位47-16位

**SDH RESP1 SD 接收响应标志寄存器 1**

寄存器	偏移量	R/W	描述	复位值
SDH_RESP1	SDHx_BA+0x834	R	SD接收响应标志寄存器 1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
RESPTK1							

位	描述	
[7:0]	RESPTK1	<b>SD接收响应标志 1</b> SD 主机控制器当RIEN (SDH_CTL[1])为1时会接收响应标志得到SD卡的回复。这个寄存器包含相应标识的15-8 位

**SDH\_BLEN SD 数据块长度寄存器**

寄存器	偏移量	R/W	描述	复位值
SDH_BLEN	SDHx_BA+0x838	R/W	SD数据块长度寄存器	0x0000_01FF

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved					BLKLEN		
7	6	5	4	3	2	1	0
BLKLEN							

位	描述	
[10:0]	BLKLEN	<b>SD 数据块长度字节数</b> 一个11位的值设定SD传输数据块的字节数。实际的字节数是BLKLEN+1. <b>注意:</b> 默认的SD数据块长度是512字节

**SDH TOUT SD 响应/数据输入超时寄存器**

寄存器	偏移量	R/W	描述	复位值
SDH_TOUT	SDHx_BA+0x83C	R/W	SD响应/数据输入超时寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
TOUT							
15	14	13	12	11	10	9	8
TOUT							
7	6	5	4	3	2	1	0
TOUT							

位	描述	
[23:0]	TOUT	<p><b>SD 响应/数据输入超时值</b>            一个24位的值设定响应/数据输入超时计数值。SD 主机控制器会等待输入数据或相应的开始位直到计数达到这个值。时钟周期依照SD引擎时钟频率。不要在这里写入过小的值，那样你会因为超时而收不到数据或响应。</p> <p><b>注意:</b> 填0x0会禁止硬件超时功能</p>

## 6.29 EBI 外部总线接口

### 6.29.1 概述

芯片的外部总线接口(EBI)，支持地址与数据复用模式。有三个片选，可连接三个不同时序的外部设备。

### 6.29.2 特性

- 支持三路存储bank
- 为每个bank提供了有极性控制的片选引脚
- 每个bank访问空间达到1MB，实际上外部可寻址空间取决于芯片封装引脚
- 支持8位/16位数据宽度
- 16位数据宽度模式下支持字节写
- 支持地址/数据复用模式
- 每个存储bank支持时序参数独立调整
- 支持LCD i80接口模式
- 支持PDMA模式
- 支持基于HCLK所产生的不同频率的总线基本时钟(MCLK)
- 支持可配置的空闲周期以用于不同访问条件：空闲写命令完成(W2X)，空闲连续读(R2R)
- 支持地址总线个数据总线分离模式

## 6.29.3 框图

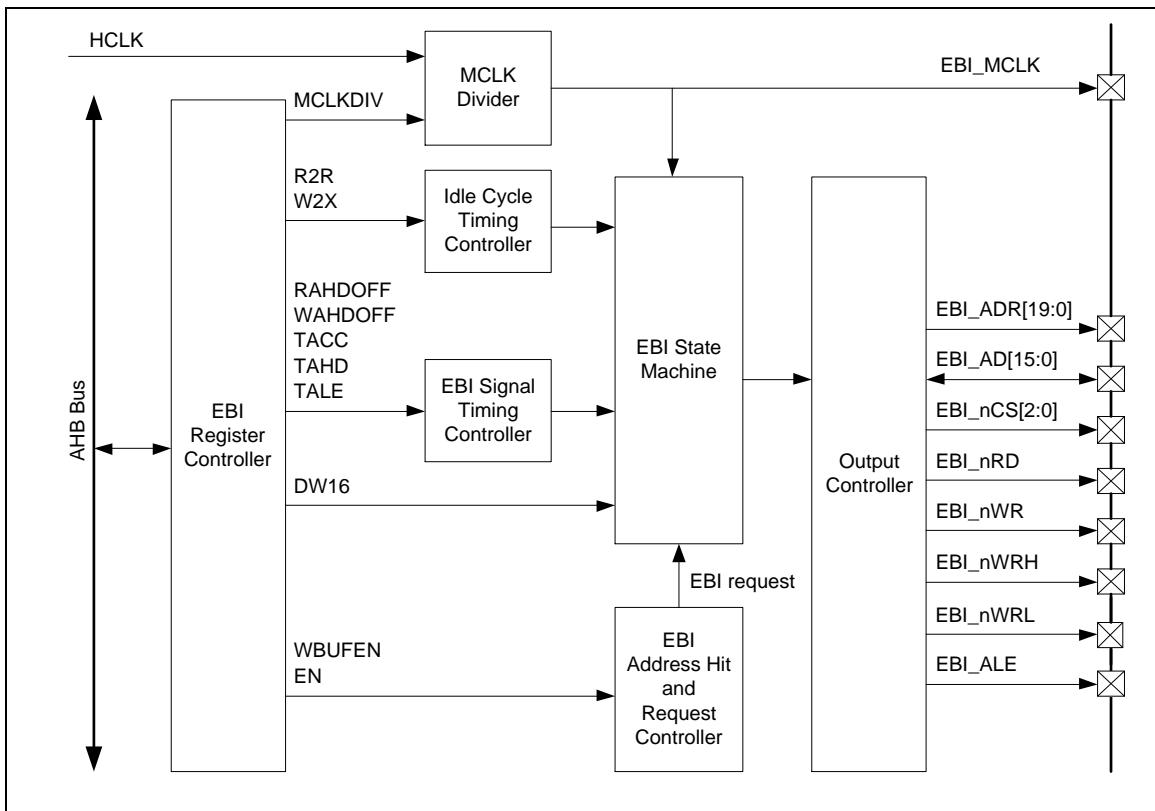


图 6.29-1 EBI 框图

## 6.29.4 基本配置

- 时钟源配置
  - 通过设置EBICKEN (CLK\_AHBCLK[3])使能EBI控制器时钟
- 复位配置
  - 通过设置QSPI0\_RST (SYS\_IPRST0[3])复位EBI控制器
- 管脚配置

组	管脚名称	GPIO	MFP
EBI	EBI_AD0	PC.0, PG.9	MFP2
	EBI_AD1	PC.1, PG.10	MFP2
	EBI_AD2	PC.2, PG.11	MFP2
	EBI_AD3	PC.3, PG.12	MFP2
	EBI_AD4	PC.4, PG.13	MFP2
	EBI_AD5	PC.5, PG.14	MFP2
	EBI_AD6	PA.6, PD.8	MFP2
	EBI_AD7	PA.7, PD.9	MFP2

EBI_AD8	PC.6, PE.14	MFP2
EBI_AD9	PC.7, PE.15	MFP2
EBI_AD10	PD.3, PD.13, PE.1	MFP2
EBI_AD11	PC.14, PD.2, PE.0	MFP2
EBI_AD12	PB.15, PD.1, PH.8	MFP2
EBI_AD13	PB.14, PD.0, PH.9	MFP2
EBI_AD14	PB.13, PH.10	MFP2
EBI_AD15	PB.12, PH.11	MFP2
EBI_ADR0	PB.5, PH.7	MFP2
EBI_ADR1	PB.4, PH.6	MFP2
EBI_ADR2	PB.3, PH.5	MFP2
EBI_ADR3	PB.2, PH.4	MFP2
EBI_ADR4	PC.12, PH.3	MFP2
EBI_ADR5	PC.11, PH.2	MFP2
EBI_ADR6	PC.10, PH.1	MFP2
EBI_ADR7	PC.9, PH.0	MFP2
EBI_ADR8	PB.1, PG.0	MFP2
EBI_ADR9	PB.0, PG.1	MFP2
EBI_ADR10	PC.13, PE.8	MFP2
EBI_ADR11	PE.9, PG.2	MFP2
EBI_ADR12	PE.10, PG.3	MFP2
EBI_ADR13	PE.11, PG.4	MFP2
EBI_ADR14	PE.12, PF.11	MFP2
EBI_ADR15	PE.13, PF.10	MFP2
EBI_ADR16	PB.11, PC.8, PF.9	MFP2
EBI_ADR17	PB.10, PF.8	MFP2
EBI_ADR18	PB.9, PF.7	MFP2
EBI_ADR19	PB.8, PF.6	MFP2
EBI_ALE	PA.8, PE.2	MFP2
EBI_MCLK	PA.9, PE.3	MFP2
EBI_nCS0	PD.12, PD.14, PF.3	MFP2
	PF.6	MFP7
	PB.7	MFP8
EBI_nCS1	PD.11, PF.2, PG.5	MFP2
	PB.6	MFP8

	EBI_nCS2	PD.10, PG.6	MFP2
	EBI_nRD	PA.11, PE.5	MFP2
	EBI_nWR	PA.10, PE.4	MFP2
	EBI_nWRH	PB.6, PG.8	MFP2
	EBI_nWRL	PB.7, PG.7	MFP2

### 6.29.5 功能描述

#### 6.29.5.1 EBI区域和地址

EBI 映射地址分布在0x6000\_0000 ~ 0x602F\_FFFF，内存空间为3M。系统请求在EBI 的内存空间内时，相应的EBI芯片选择信号有效。

片选信号	地址映射
EBI_nCS0	0x6000_0000 ~ 0x600F_FFFF
EBI_nCS1	0x6010_0000 ~ 0x601F_FFFF
EBI_nCS2	0x6020_0000 ~ 0x602F_FFFF

表 6.29-1 EBI 地址映射

为了映射整个EBI 内存空间， 8-位设备需20-位地址宽度， 16-位设备需19-位地址宽度。

小于20-位地址的设备， EBI 将映射设备到镜像空间。例如，一个18-位 EBI地址设备，EBI ( Bank0/EBI\_nCS0) 将外设同时映射到 0x6000\_0000 ~0x6003\_FFFF, 0x6004\_0000 ~ 0x6007\_FFFF, 0x6008\_0000 ~0x600B\_FFFF 和 0x600C\_0000 ~0x600F\_FFFF.

#### 6.29.5.2 EBI数据宽度连接-地址与数据总线复用模式

EBI 支持地址总线和数据总线复用。对于地址总线和数据总线分开的外部设备，与设备的连接需要额外的逻辑单元锁存地址。这样，管脚EBI\_ALE需要连接到锁存器上锁存地址值。管脚EBI\_AD为锁存器的输入，锁存器的输出连接到外部设备的地址总线上。

对于16-位设备， EBI\_AD[15:0]是地址与数据复用线， EBI\_ADR [18:16]仅为地址，直接与16-位设备连接。当EBI 数据宽度设置为16位， EBI\_ADR[19]无作用。对于8-位设备，仅EBI\_AD [7:0]地址与数据复用， EBI\_AD[15:8] 和 EBI\_ADR[19:16] 为地址，直接与8-位设备连接。图 6.29-2描述 16-位 EBI 数据宽度与16-位设备的连接，图 6.29-3描述8-位 EBI 数据宽度与8-位设备的连接

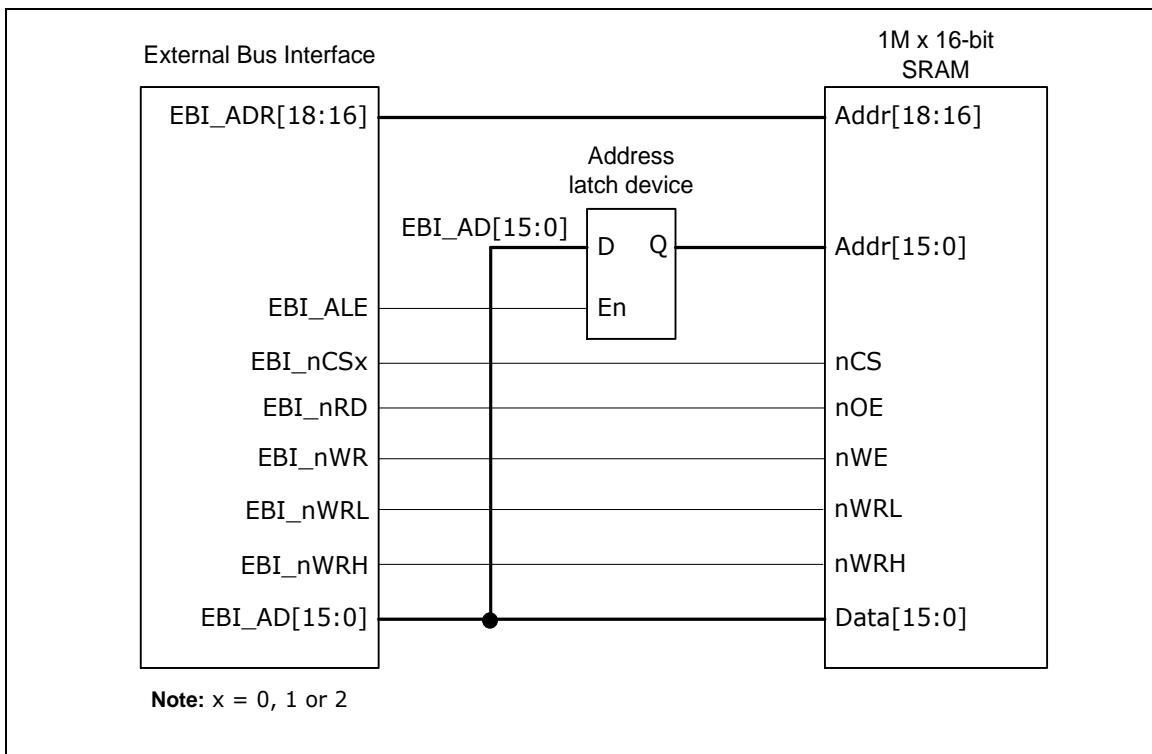


图 6.29-2 16-位 EBI 数据宽度与16-位设备的连接

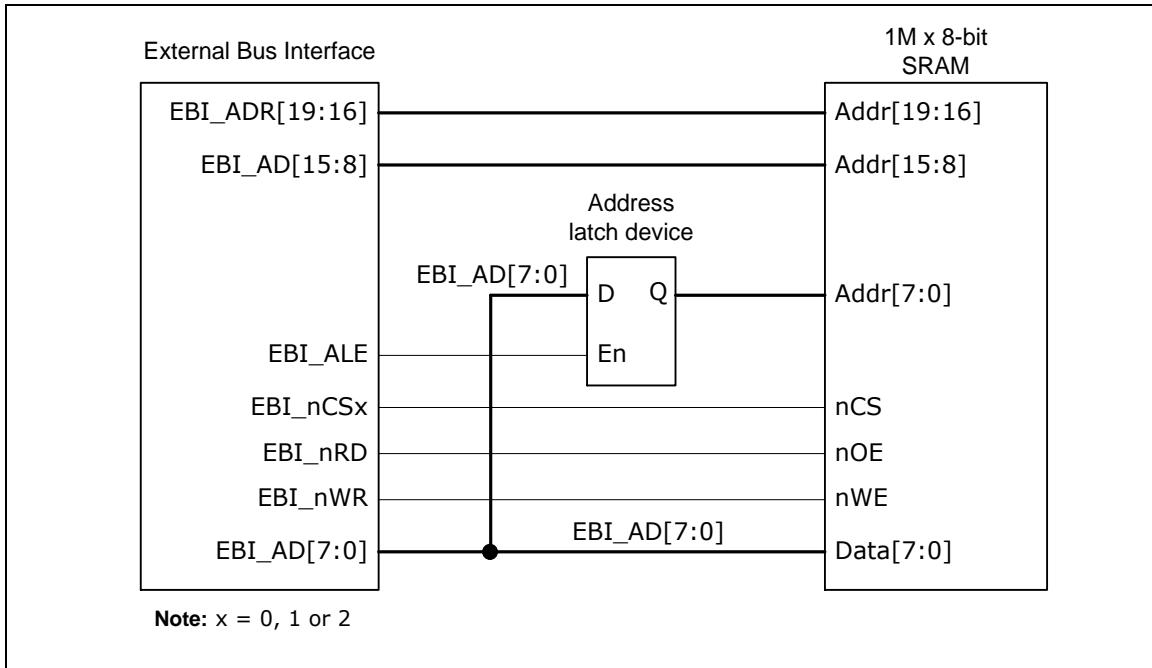


图 6.29-3 8-位 EBI 数据宽度与8-位设备的连接

当系统访问数据宽度大于EBI 的数据宽度时， EBI 将访问一次以上。例如，如果系统通过EBI设备请求32-位数据，当EBI为8-位数据宽度时， EBI将4次访问总线。

## 6.29.5.3 EBI数据宽度连接-地址与数据总线分离模式

EBI 支持地址和数据总线分离模式。用户设置 ADSEOPEN (EBI\_CTLx[3]) 使能该模式：EBI\_AD数据总线直接连接到设备数据总线，EBI\_ADR直接连到设备地址总线。

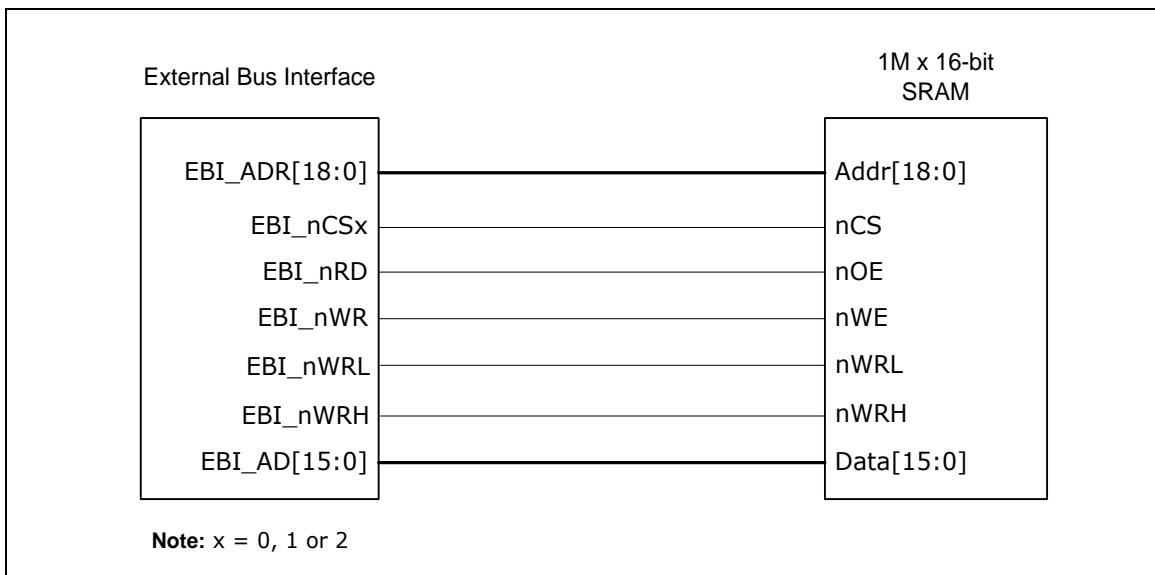


图 6.29-4 分离模式下16-位 EBI 数据宽度与16-位设备的连接

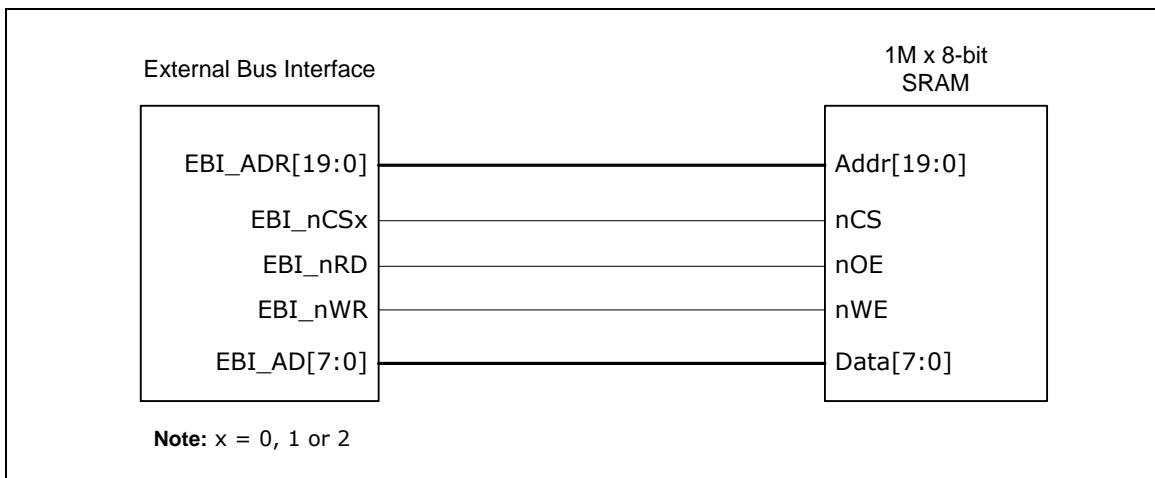


图 6.29-5 分离模式下8-位 EBI 数据宽度与8-位设备的连接

## 6.29.5.4 EBI操作控制

**MCLK 控制**

EBI工作时，所有EBI信号与EBI\_MCLK同步。以低工作频率连接到外部设备时，EBI\_MCLK可以通过寄存器MCLKDIV (EBI\_CTLx[10:8])最多分频到HCLK/32。如果EBI\_MCLK设置为HCLK/1，则EBI信号与EBI\_MCLK的正边沿同步，否则与EBI\_MCLK的负边沿同步。

**操作和访问时序控制**

开始访问时，片选(EBI\_nCS0、EBI\_nCS1和EBI\_nCS2)置低并等待一个EBI\_MCLK时间 (tASU)以使地址稳定，然后 EBI\_ALE置高并保持一段时间(tALE)用于地址锁存。地址锁存后，EBI\_ALE置低并等待一个EBI\_MCLK时间 (tLHD)，再插入一个EBI\_MCLK时间 (tA2D)用于总线转换(地址到数据)。读时EBI\_nRD置低，写时EBI\_nWR置低。在保持访问时间(tACC)后EBI\_nRD或EBI\_nWR置高完成读或写。之后，EBI信号在数据保持时间(tAHD)之后片选置高，一个总线周期结束。

EBI 控制器提供灵活的时序控制。tASU, tLHD 和tA2D固定为1个EBI\_MCLK时间。 tAHD通过寄存器设定TAHD (EBI\_TCTLx[10:8])可以在1~8个EBI\_MCLK周期内调节， tACC 通过寄存器TACC(EBI\_TCTLx[7:3]) 可以在1~32个EBI\_MCLK 周期内调节， tALE 通过寄存器TALE(EBI\_CTL0[18:16] 可以在1~8个EBI\_MCLK周期内调节。一些外设可以支持数据访问零保持时间，EBI控制器可以略过tAHD，通过寄存器设定WAHDOFF (EBI\_TCTLx[23]) 和 RAHD OFF (EBI\_TCTLx[22]) 提高访问速度。

对每个芯片的片选，除了tALE 可以由EBI\_CTL0控制，其他的寄存器设定，EBI会提供单独的寄存器用于时序控制。

参数	值	单位	描述
tASU	1	MCLK	地址锁存建立时间
tALE	1 ~ 8	MCLK	ALE 高时间. 由 TALE (EBI_CTL0[18:16])控制。
tLHD	1	MCLK	地址锁存保持时间。
tA2D	1	MCLK	地址到数据的延时 (总线转换时间).
tACC	1 ~ 32	MCLK	数据访问时间. 由 TACC (EBI_TCTLx[7:3])控制。
tAHD	1 ~ 8	MCLK	数据访问保持时间. 由 TAHD (EBI_TCTLx[10:8])控制。
IDLE	0 ~ 15	MCLK	空闲周期. 由 R2R (EBI_TCTLx[27:24])和W2X (EBI_TCTLx[15:12])控制。

表 6.29-2 时序控制参数

图 6.29-6 是一个设置 16-位数据宽度的例子。在该例中，EBI\_AD 总线用作地址[15:0]和数据[15:0]。当 EBI\_ALE 置高，EBI\_AD 为地址输出。在地址锁存后，EBI\_ALE 置低并且在读取访问操作时，EBI\_AD 总线转换成高阻以等待设备输出数据，或用于写数据输出。

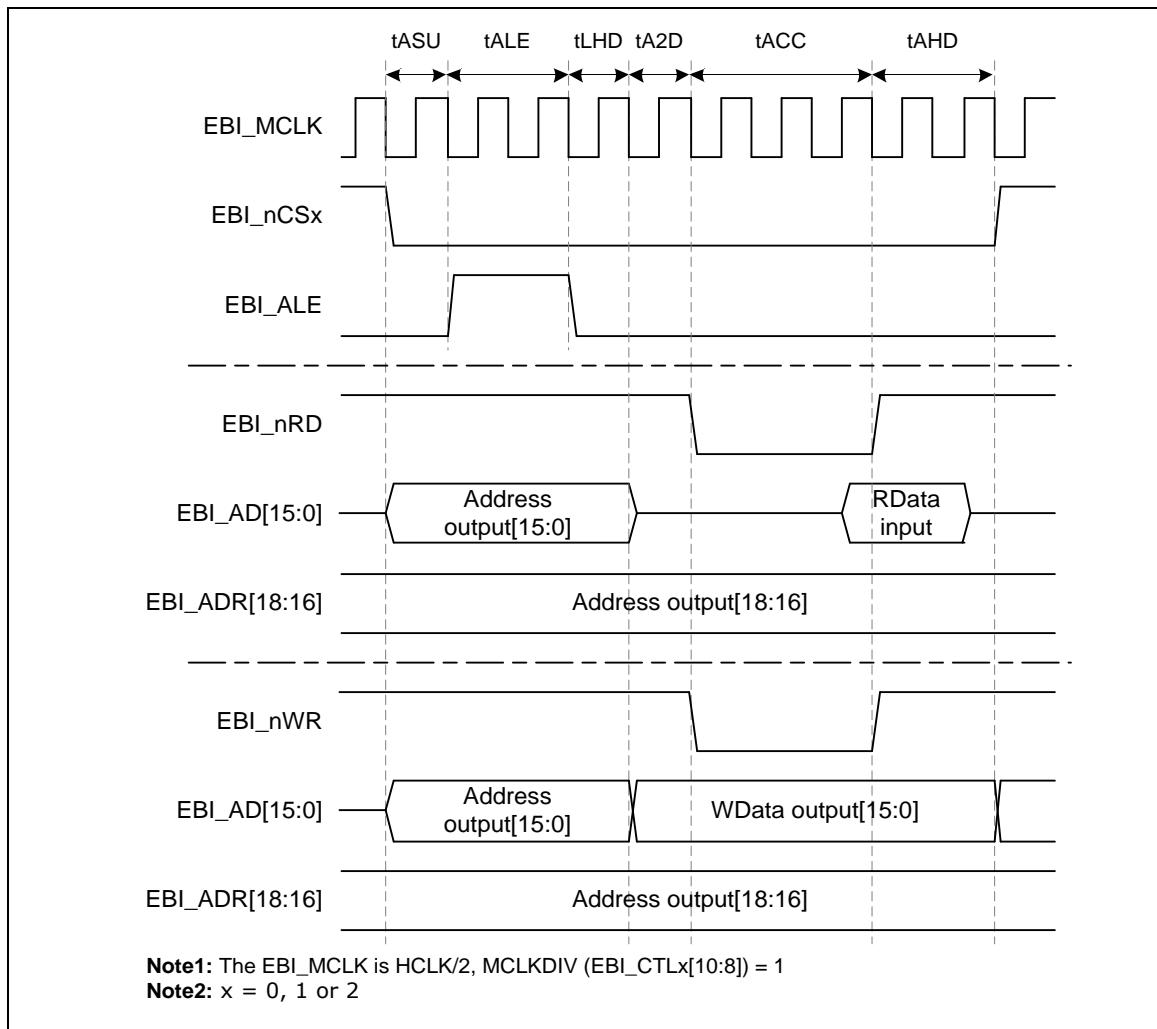


图 6.29-6 16-位数据宽度的时序控制波形

图 6.29-7 是一个设置 8 位数据宽度的例子。8 位和 16 位数据宽度的不同之处在于 EBI\_AD[15:8]。在 8 位数据宽度的设置中，EBI\_AD[15:8] 总为地址 [15:8] 输出，因此外部锁存只需 8 位宽度。

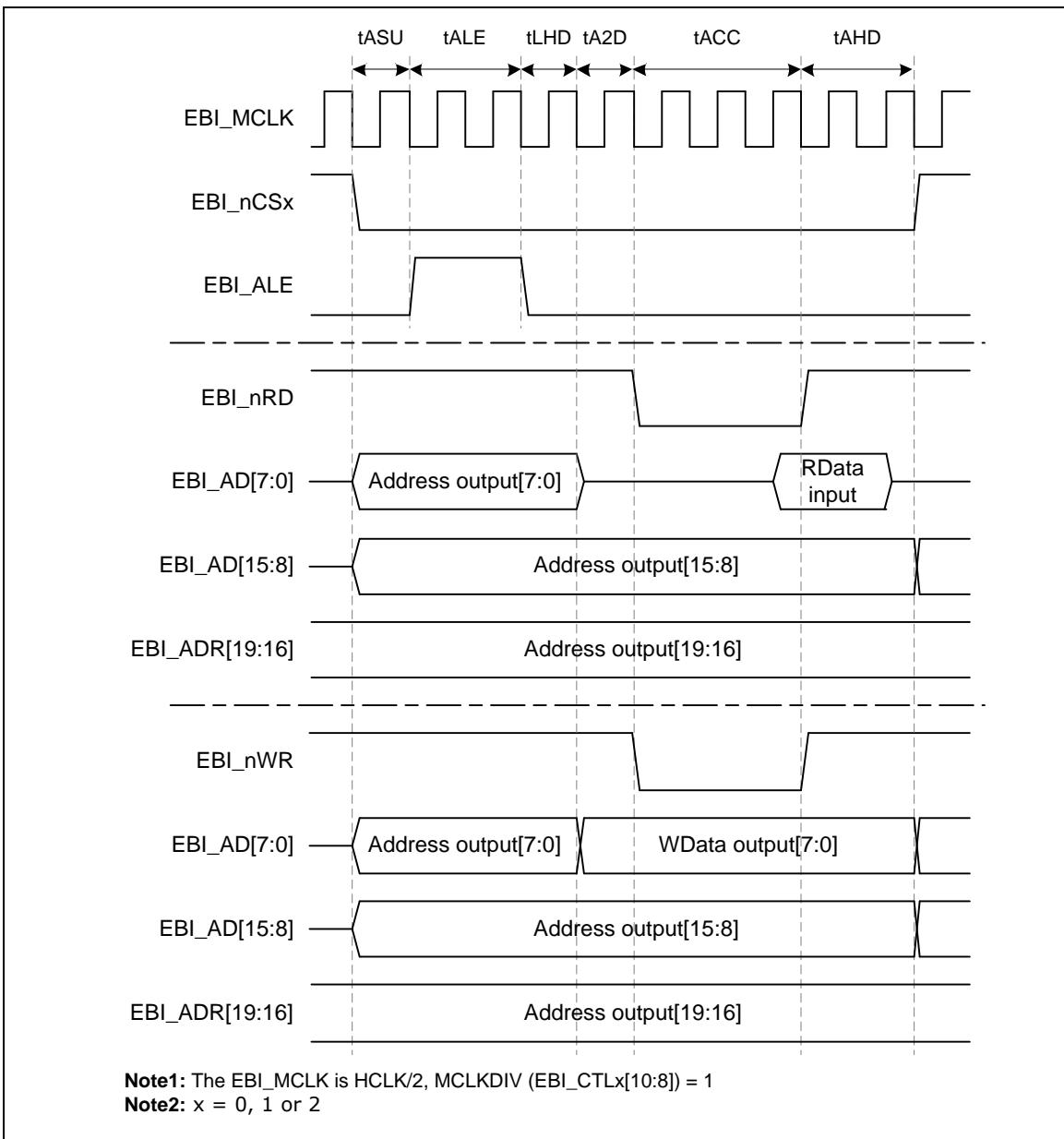


图 6.29-7 8-位数据宽度时序控制波形

### 字节访问

当连接 16 位设备时 EBI 支持字节访问。16 位数据总线上用引脚 EBI\_nWRH 和 EBI\_nWRL 指示高字节使能和低字节使能。图 6.29-8 展示用 EBI\_nWRH 指示在 EBI\_AD[15:8] 上 8 位数据宽度的写操作

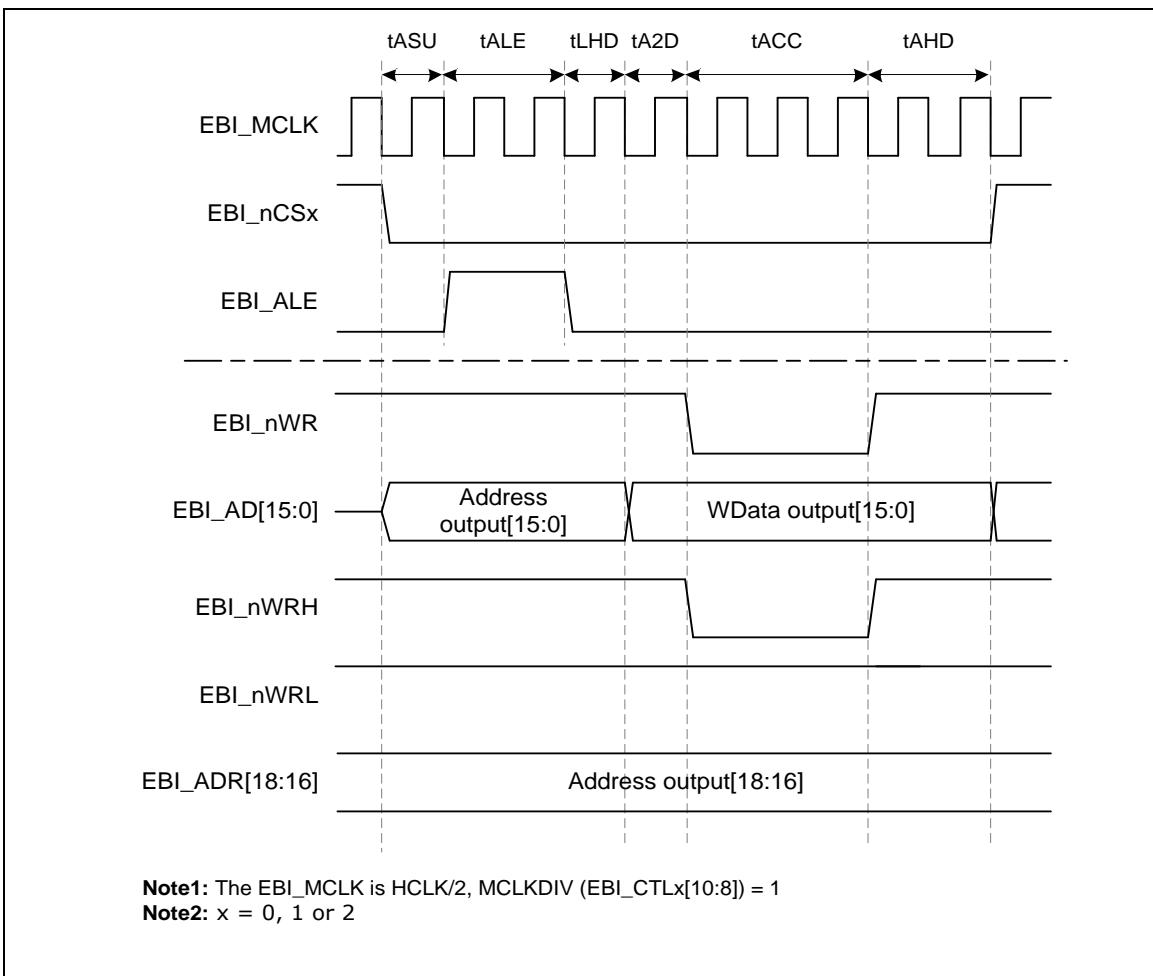


图 6.29-8 16位数据宽度字节写的时序控制波形

### 插入空闲周期

当EBI连续访问时，如果器件访问时间较慢，可能会有总线冲突。EBI控制器支持额外空闲周期以解决该问题。在空闲周期，所有EBI的控制信号无效。如下空闲周期图

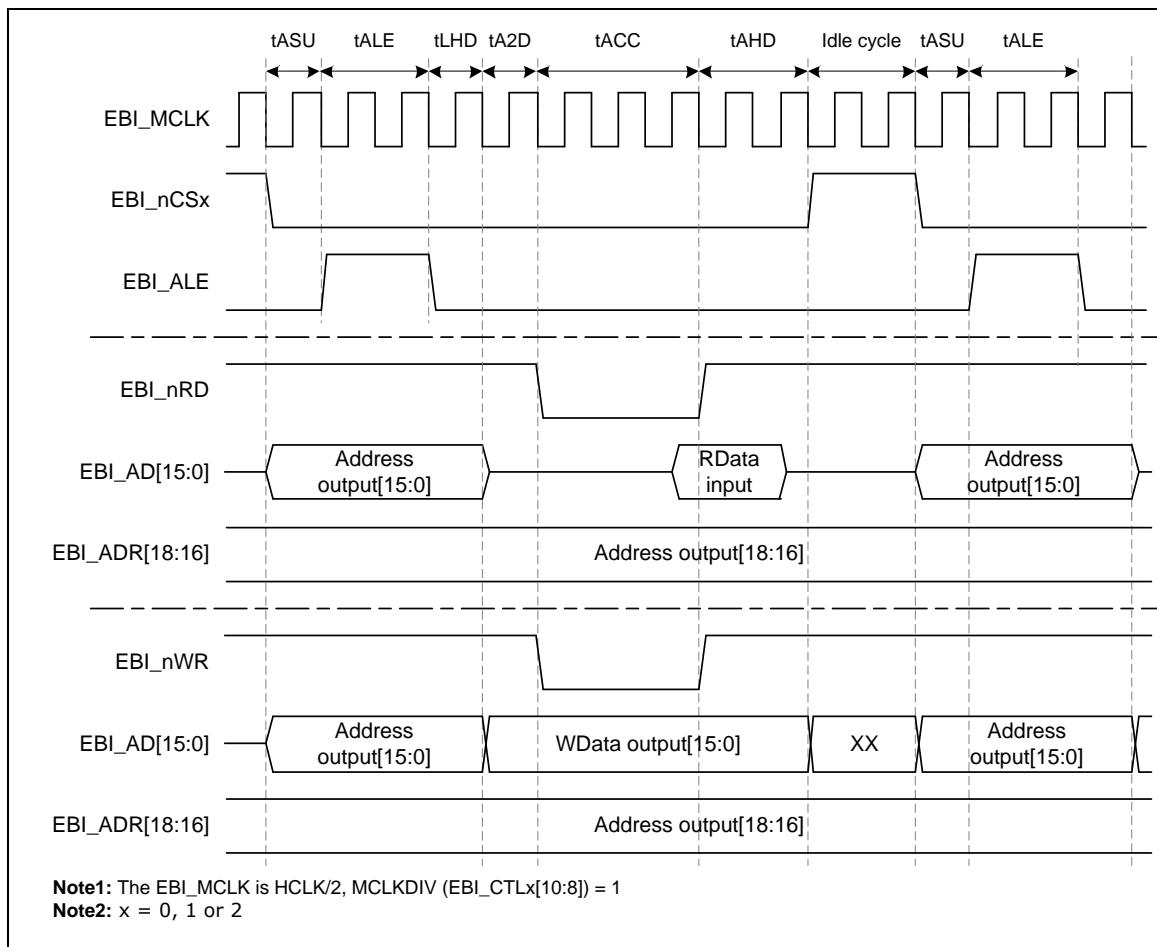


图 6.29-9 插入空闲周期的时序控制

以下两个条件，EBI可插入空闲周期：

1. 写访问之后
2. 读访问之后与下一个读访问之前(R2R 空闲周期)

通过设定寄存器W2X (EBI\_TCTLx[15:12]), 和R2R (EBI\_TCTLx[27:24]), 空闲周期可设定在0~15 EBI\_MCLK。

### 片选极性控制

CSPOLINV (EBI\_CTLx[2]) 置0，片选(EBI\_nCSx)低电平有效。这意味着在EBI\_nCSx低电平时访问外部设备。CSPOLINV (EBI\_CTLx[2]) 置1，片选(EBI\_nCSx)高电平有效。这意味着当在EBI\_nCSx高电平时访问外部设备。

### 写缓冲

当用户通过EBI总线写一个数据到外设时，EBI控制器将立刻开始写动作，CPU是保持状态，直到EBI写动作结束。用户可以使能写缓存功能来提高CPU和EBI的访问效率。当EBI写缓存功能使能后，在EBI向外设写数据期间，CPU还可以持续执行其他指令。例外的情况，当EBI执行写动作时，如果CPU通过EBI执行另一个数据访问，CPU会是保持状态。

用户可以通过设置 WBUFEN (EBI\_CTL0[24])使能写缓存.

### 地址数据分离模式

当EBI设置为分离模式，tALE, tLHD, tA2D周期会被忽略。EBI\_AD 和 EBI\_ADR分别各自用于数据和地址总线。

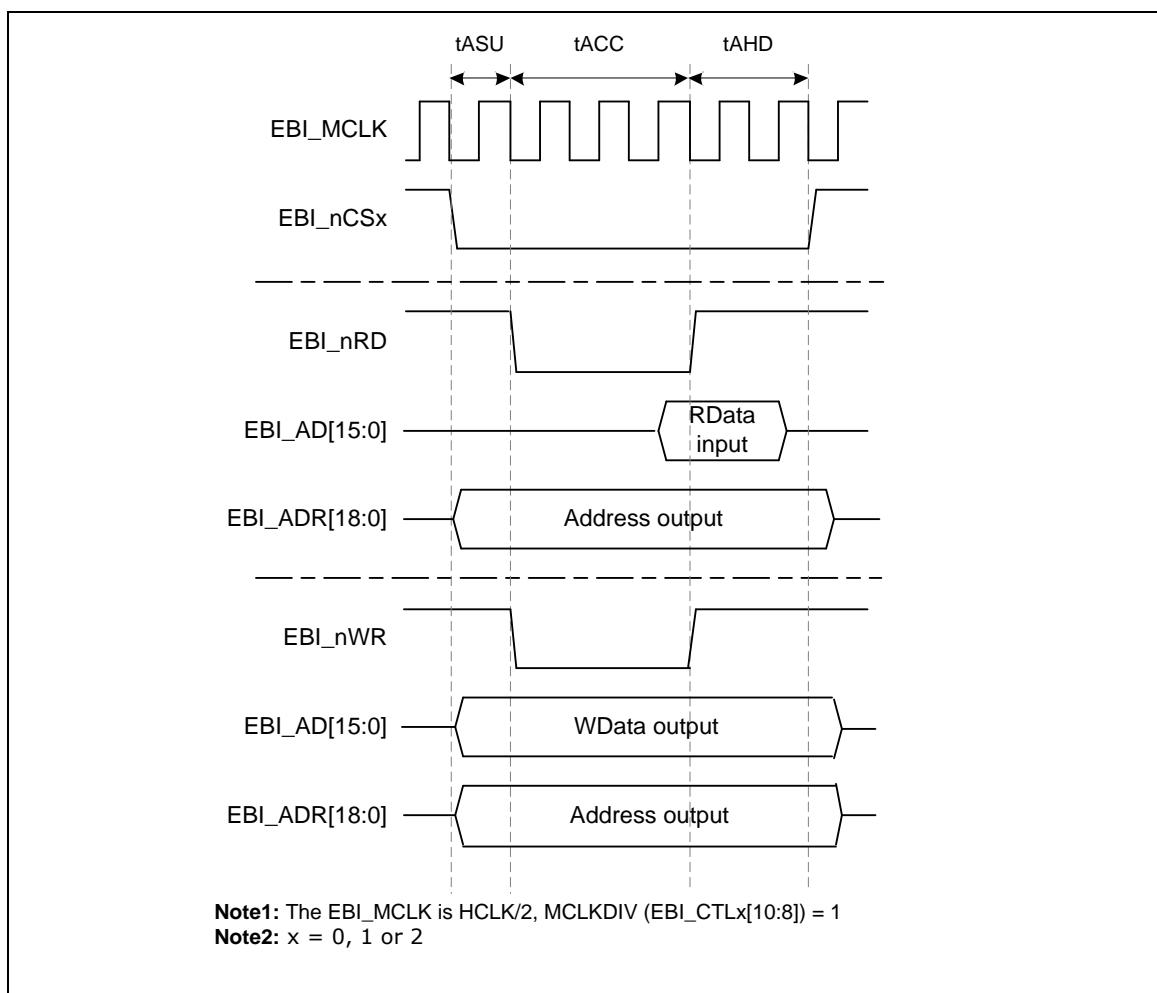


图 6.29-10 16位数据宽度在分离模式下的时序控制波形

### 连续数据访问模式

对于需要更快的数据访问且不需要地址控制的接口，EBI支持连续数据访问。每个bank，用户可以通过设置CACCESS (EBI\_CTLX[4])使能该模式。当EBI设置为连续数据访问模式，tASU, tALE, tLHD周期会被忽略，且EBI可以在一个读或写命令内连续地访问数据。在每个访问命令之间会有无效的周期时间。时序如下图。

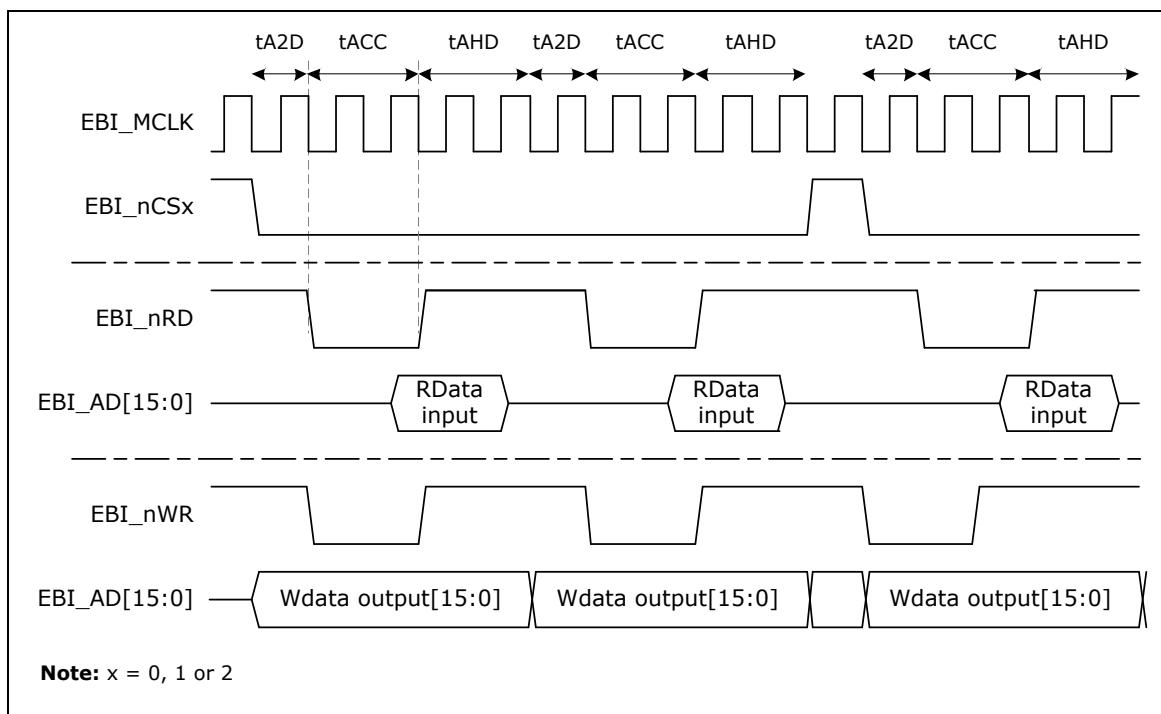


图 6.29-11 连续数据访问模式的时序控制波形

### 6.29.6 寄存器映射

R: 只读, W: 只写, R/W: 可读写

寄存器	偏移地址	R/W	描述	复位值
<b>EBI 基地址:</b>				
<b>EBI_BA = 0x4001_0000</b>				
<b>EBI_CTL0</b>	EBI_BA+0x00	R/W	外部总线接口 Bank0 控制寄存器	0x0000_0000
<b>EBI_TCTL0</b>	EBI_BA+0x04	R/W	外部总线接口 Bank0 时序控制寄存器	0x0000_0000
<b>EBI_CTL1</b>	EBI_BA+0x10	R/W	外部总线接口 Bank1 控制寄存器	0x0000_0000
<b>EBI_TCTL1</b>	EBI_BA+0x14	R/W	外部总线接口 Bank1 时序控制寄存器	0x0000_0000
<b>EBI_CTL2</b>	EBI_BA+0x20	R/W	外部总线接口 Bank2 控制寄存器	0x0000_0000
<b>EBI_TCTL2</b>	EBI_BA+0x24	R/W	外部总线接口 Bank2 时序控制寄存器	0x0000_0000

### 6.29.7 寄存器描述

#### EBI\_CTLx 外部总线接口控制寄存器

寄存器	偏移地址	R/W	描述	复位值
EBI_CTL0	EBI_BA+0x00	R/W	外部总线接口Bank0控制寄存器	0x0000_0000
EBI_CTL1	EBI_BA+0x10	R/W	外部总线接口Bank1控制寄存器	0x0000_0000
EBI_CTL2	EBI_BA+0x20	R/W	外部总线接口Bank2控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24	
								WBUFEN
23	22	21	20	19	18	17	16	
								TALE
15	14	13	12	11	10	9	8	
								MCLKDIV
7	6	5	4	3	2	1	0	
			CACCESS	ADSEOPEN	CSPOLINV	DW16	EN	

位	描述
[31:25]	保留.
[24]	<b>WBUFEN</b> <b>EBI 写缓存使能位</b> 0 = 禁用 EBI 写缓存 1 = 使能 EBI 写缓存 <b>注意:</b> 该位仅在 EBI_CTL0 寄存器中有效
[23:19]	保留.
[18:16]	<b>TALE</b> <b>ALE的扩展时间</b> 控制 EBI_ALE 高电平时间(tALE) $tALE = (TALE + 1) * EBI_MCLK$ . <b>注意:</b> 该位仅在 EBI_CTL0 寄存器中有效
[15:11]	保留.
[10:8]	<b>MCLKDIV</b> <b>外部输出时钟分频器</b> MCLK 频率如下: 000 = HCLK/1. 001 = HCLK/2. 010 = HCLK/4. 011 = HCLK/8. 100 = HCLK/16. 101 = HCLK/32. 110 = HCLK/64. 111 = HCLK/128.

[7:5]		保留.
[4]	<b>CACCESS</b>	<b>连续数据访问模式</b> 当连续访问模式使能，为了连续数据传输需要，tASU, tALE 和 tLHD周期会被忽略。 0 = 连续数据访问模式禁止. 1 = 连续数据访问模式使能.
[3]	<b>ADSEOPEN</b>	<b>EBI地址/数据总线分离模式使能位</b> 0 = EBI地址/数据总线复用模式 1 = EBI地址/数据总线分离模式
[2]	<b>CSPOLINV</b>	<b>片选管脚极性反转</b> 0 = 片选管脚 (EBI_nCS) 低电平有效 1 = 片选管脚 (EBI_nCS) 高电平有效
[1]	<b>DW16</b>	<b>EBI 数据宽度16-位选择</b> 0 = EBI 数据宽度是 8-位 1 = EBI 数据宽度是 16-位
[0]	<b>EN</b>	<b>EBI 使能位</b> 该位是EBI的功能使能位 0 = 禁用EBI 功能 1 = 使能EBI 功能

## EBI\_TCTLx 外部总线接口时序控制寄存器

寄存器	偏移地址	R/W	描述				复位值
EBI_TCTL0	EBI_BA+0x04	R/W	外接总线接口Bank0时序控制寄存器				0x0000_0000
EBI_TCTL1	EBI_BA+0x14	R/W	外接总线接口Bank1时序控制寄存器				0x0000_0000
EBI_TCTL2	EBI_BA+0x24	R/W	外接总线接口Bank2时序控制寄存器				0x0000_0000

31	30	29	28	27	26	25	24
				R2R			
23	22	21	20	19	18	17	16
WAHDOFF	RAHDOFF						
15	14	13	12	11	10	9	8
W2X				Reversed	TAHD		
7	6	5	4	3	2	1	0
TACC				Reserved			

位	描述
[31:30]	保留.
[27:24]	R2R  读-读之间的空闲状态周期  读-读(R2R)空闲状态周期= (R2R * EBI_MCLK).  当读动作完成, 下一个动作也是读时, 插入读-读(R2R)空闲状态周期后, EBI_nCS回空闲状态
[23]	WAHDOFF  EBI写时, 访问保持时间禁用  0 = 在EBI写期间, 使能数据访问保持时间(tAHD) 1 = 在EBI写期间, 禁用数据访问保持时间(tAHD).
[22]	RAHDOFF  EBI读时, 访问保持时间禁用  0 = 在EBI读期间, 使能数据访问保持时间(tAHD) 1 = 在EBI读期间, 禁用数据访问保持时间(tAHD).
[21:16]	保留.
[15:12]	W2X  写之后的空闲状态周期  W2X 空闲状态周期 = (W2X * EBI_MCLK).  当写动作完成, 插入W2X空闲状态周期后, EBI_nCS回空闲状态
[11]	保留.
[10:8]	TAHD  EBI 数据访问保持时间  TAHD 定义数据访问保持时间(tAHD). tAHD = (TAHD +1) * EBI_MCLK.
[7:3]	TACC  EBI 数据访问时间  TACC 定义数据访问时间 (tACC).

		tACC = (TACC +1) * EBI_MCLK.
[2:0]		保留.

## 6.30 USBD USB 1.1设备控制器

### 6.30.1 特性

- 兼容USB 2.0全速规范
- 一个4种中断事件（包括唤醒、插拔、USB、总线）的中断向量
- 支持控制、批量、中断、同步四种传输
- 总线闲置3ms以上切换到总线挂起功能
- 提供可配置为控制/批量/中断/同步四种传输模式的12个通讯端点，以及一个最大1K字节的数据缓冲区
- 提供远程唤醒功能

### 6.30.2 框图

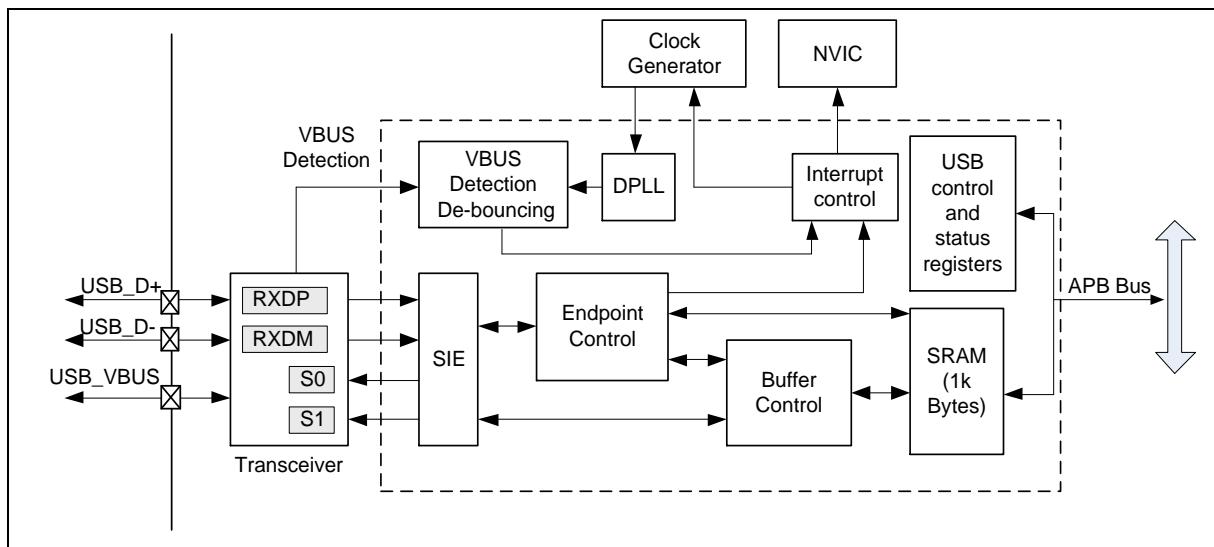


图 6.30-1 USB 框图

### 6.30.3 基本配置

USB 的帧的功能由USBROLE (SYS\_USBPHY[1:0])来设置。内部USB 3.3V LDO可通过使能LDO33EN (SYS\_USBPHY[8])打开。这两个配置都是写保护位。

USBD设备的时钟由PLL时钟分频所得。用户必须在USB设备控制器使能前，先配置PLL时钟。设置USBDCKEN (CLK\_APBCLK0[27])位可使能USBD时钟，设置USBDIV (CLK\_CLKDIV0[7:4])对USB时钟预分频，产生合适的时钟频率。

#### 6.30.3.1 USB 设备 1.1 基本配置

- 时钟源配置
  - 在CLK\_PLLCTL中设置PLL控制器

- 在USBDIV(CLK\_CLKDIV0[7:4])中选择USBD外设时钟分频数目
- 通过设置USBDCKEN (CLK\_APBCLK0[27])使能USBD外设时钟
- 复位配置
  - 通过设置USBDRST (SYS\_IPRST1[27])复位USBD控制器
- 管脚配置

组	管脚名称	GPIO	MFP
USB	USB_D+	PA.14	MFP14
	USB_D-	PA.13	MFP14
	USB_OTG_ID	PA.15	MFP14
	USB_VBUS	PA.12	MFP14
	USB_VBUS_EN	PB.6, PB.15	MFP14
	USB_VBUS_ST	PB.7, PC.14, PD.4	MFP14

#### 6.30.4 功能描述

##### 6.30.4.1 串行接口引擎 (SIE)

SIE是设备控制器的前端，用于处理USB协议。SIE的主要工作是向上发送信号到事务处理层，包括：

- 包识别，事务排序
- SOF,EOP, RESET, RESUME 信号的检测与产生
- 时钟/数据分离
- NRZI数据编解码和位填充
- CRC校验的生成和检测（仅对Token和data）
- 包ID (PID)产生和检测/解码
- 串-并/并-串转换

##### 6.30.4.2 端点控制

控制器中总共有12个端点。每个端点都可以配置成控制，批量，中断或同步传输模式。这四种模式所对应的传输过程都是通过这个模块来完成。该控制器也用于管理数据流同步，端点状态控制，当前端点起始地址，当前事务状态，以及每个端点的数据缓冲区状态。

##### 6.30.4.3 数字锁相环 (DPLL)

USB数据的传输比特率是12MHz，DPLL使用来自时钟控制器的48MHz时钟源来锁定RXDP和RXDM上的输入数据。12M的比特率时钟也是由DPLL转换而来。

#### 6.30.4.4 VBUS去抖检测

USB设备有可能经常在主机上被插拔。当USB设备从主机上拔下后，为了监测到它的状态，设备控制器提供了一个硬件去抖USB VBUS检测中断，来避免USB插拔时的抖动问题。USB设备插拔操作10ms后会产生一个VBUS检测中断。用户可以通过读USBD\_VBUSDET寄存器的内容知道USB设备的插拔状态。VBUSDET标志反应了USB BUS 没有消抖情况下的当前状态。如果VBUSDET为1，表示USB线被插入。如果用户轮询该标志来检测USB的状态，需要通过软件来做消抖。

#### 6.30.4.5 中断控制

USB控制器带有一个USB中断向量，该中断包含了四个中断事件（唤醒事件，插拔事件，USB事件，BUS事件）。其中唤醒中断事件(NEVWK)发生在当系统从Power-down低功耗模式中唤醒时，（低功耗模式在Power-down控制寄存器CLK\_PWRCTL做了定义）。插拔中断事件(VBUSDET)用于USB设备插拔检测。USB中断事件用于告知MCU产生了一些USB请求，如IN ACK, OUT ACK等。BUS中断事件用来告知MCU产生了总线事件，如挂起，恢复等。当需要用到这些中断时，必须在USB设备控制器的中断使能控制寄存器(USBD\_INTEN)中打开相应位。

当系统在Power- down模式下唤醒后，如果20ms内没有其它USB中断事件发生，则发生唤醒中断。系统进入Power-down后，如果USB唤醒功能使能，USB\_VBUS,USB\_D+ 和 USB\_D-上的任何变化可以唤醒MCU。如果这个变化不是有意而为的，那么只有唤醒中断会发生。若 USB 唤醒超过 20 ms 后，如果没有其他 USB 中断事件发生，则唤醒中断将发生。图 6.30-2为唤醒中断的控制流程。

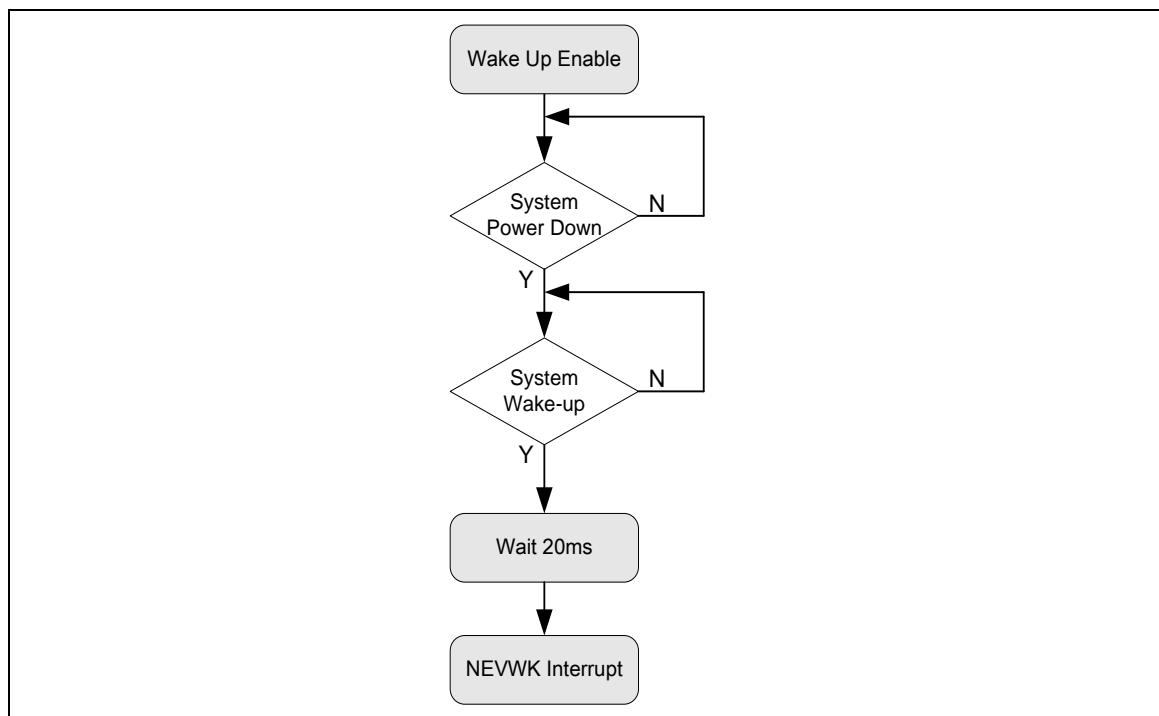


图 6.30-2 唤醒中断操作流程

USB中断事件用于告知MCU USB总线上所发生的事件，MCU可以读取EPSTS位(USBD\_EPSTS0 和 USBD\_EPSTS1)和EPEVT11~0 (USBD\_INTSTS[27:16])的内容，来知道USB当前的状态以便进行下一步操作。

和USB中断事件一样，BUS中断事件是用于告知MCU一些BUS事件。比如USB复位，挂起，超时

溢出以及总线恢复等。用户可以读USBD\_ATTR寄存器内容，从而知道USB的总线的状态。

#### 6.30.4.6 省电

在一些特殊情况下，比如挂起，用户可以写0到USBD\_ATTR[4]位来手动禁止PHY来省电。

#### 6.30.4.7 缓存控制

USB控制器中总共有1K字节的SRAM，12个端点可以共享这些缓存。在USB模块功能使能前，应先在缓冲段寄存器配置每个端点的有效起始地址。”缓冲区控制”模块就是用于控制每个端点的有效起始地址和它所分配的SRAM缓冲区间大小(在USBD\_MXPLDx寄存器定义)。

图 6.30-3描述了，根据USBD\_BUFSEGx 和 USBD\_MXPLDx寄存器中所定义的，各个端点缓冲区起始地址和大小。如果USBD\_BUFSEG0被设置为0x08h, USBD\_MXPLD0被设为0x40h,那么端点0所分配的缓冲区的大小就是从USBD\_BA+0x108h开始，到USBD\_BA+0x148h结束。(注意：USBD的SRAM起始地址是从USBD\_BA+0x100h开始)。

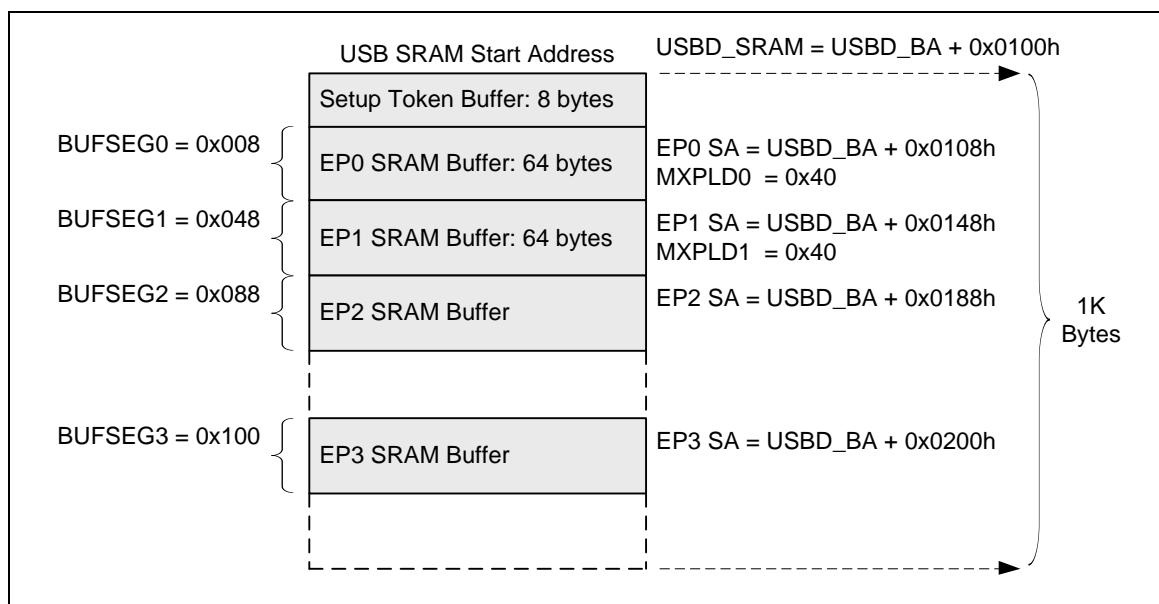


图 6.30-3端点SRAM 结构

#### 6.30.4.8 与USB外设通信处理

用户可以使用中断或轮询USBD\_INTSTS寄存器来监测USB的数据通信。当通信发生时，USBD\_INTSTS寄存器被硬件置位，并向CPU产生一个中断请求(如果相关中断打开)，或者也可以不使用中断方式，用轮询USBD\_INTSTS寄存器的相应位的方法来获取事件信息。以下是使用中断方式的控制流程。

当USB主机向设备控制器请求数据时，用户需要预先把相关数据放到指定的端点缓存。填充完数据后，用户需要写实际数据长度到USBD\_MXPLDx这个寄存器当中。一旦这个寄存器数据被写入，内部信号“In\_Rdy”信号会被设置，当接收到主机发来的IN token信号后，缓冲区数据会被立即传出去。需要注意的是在指定数据发送完成后，“In\_Rdy”信号会由硬件自动清除。

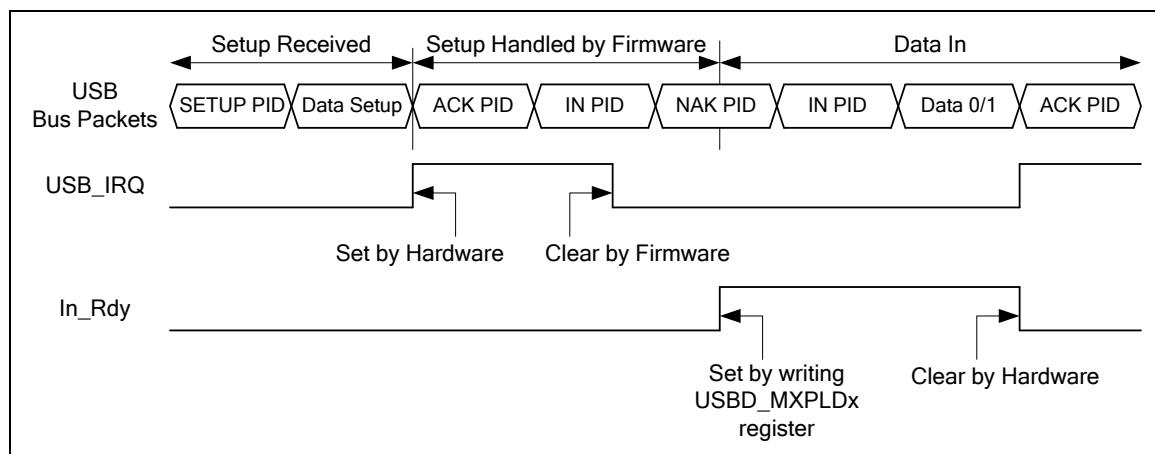


图 6.30-4 Setup 事务后接着是 Data In 事务

相应的，当USB主机想要发送数据到从设备控制器的输出端点时，硬件会把数据填充到指定的端点缓冲区。通信完成后，硬件会在端点对应的USBD\_MXPLDx寄存器中自动记录数据长度，并清除“Out\_Rdy”信号。这将会避免硬件在用户没有取走当前数据时又接收到下一笔数据。一旦用户处理了这次通信时，由软件写入特定的寄存器USBD\_MXPLDx，以再次产生“Out\_Rdy”信号来接收下一次通信。

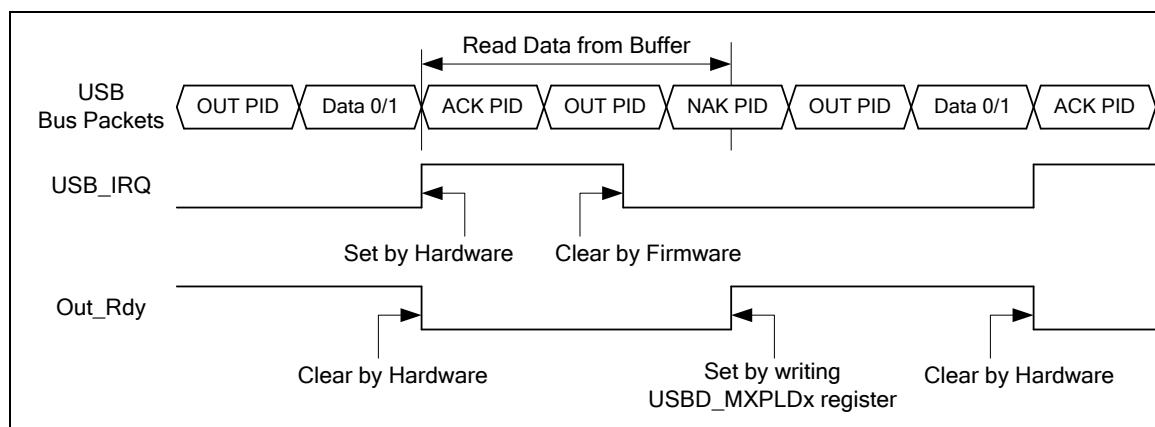


图 6.30-5 数据输出

## 6.30.4.9 链路电源管理(LPM)

电源管理就是类似于挂起/恢复功能，但是电源状态过渡延时只有10us（取代了USB2.0挂起/恢复大于20ms的延时）

新快速机制用于总线上根端口从使能状态（L0）到新休眠状态（L1）的转换。L0和L1状态详见表7.29-1，寄存器 USBD\_ATTR & USBD\_LPMATTR可以让用户了解LPM机制的当前电源状态

LPM 状态	描述
L0(运行)	在这个状态，端口被使能可以进行信号的传输。L0下的端口不是在激活状态就是在空闲状态。所谓激活就是正在发送或是接收数据。这种状态下开始帧包（SOF）由主机产生，其速率与客户设备一致。
L1(休眠)	L1类似于下面的L2使用，但是还是有一点差别。进入L1需要向hub或是主机发出请求。一个LPM事务下发到设备。仅当在设备相应ACK握手后请求事务才能发生。通过远程唤醒、恢复信号、复位信号或是断开连接离开L1状态。连接的设备如L2一样但是L1不会从VBUS消耗多少功耗。当在L1时，主机或是设备可以初始化复位信号。尽管信号恢复如同L2，信号周期和L1到L0过渡延时更短。
L2(挂起)	该状态就是USB2.0挂起状态。向hub或是主机端口发送命令进入L2。设备发现挂起条件是观察到3ms非活动状态。结果状态不是低速就是全速空闲。L2会从VBUS消耗很多功耗。通过远程唤醒、恢复信号、复位信号或是断开连接离开L2状态。
L3(关闭)	这种状态下，端口不能进行数据传输。相当于断电，断开连接和禁用。

表 6.30-1 USB 链路电源管理 (Lx) 状态

状态转化过程请参考图7.19-6，更多的USB链路电源管理（LPM）信息请参考USB2.0链路电源管理ECN

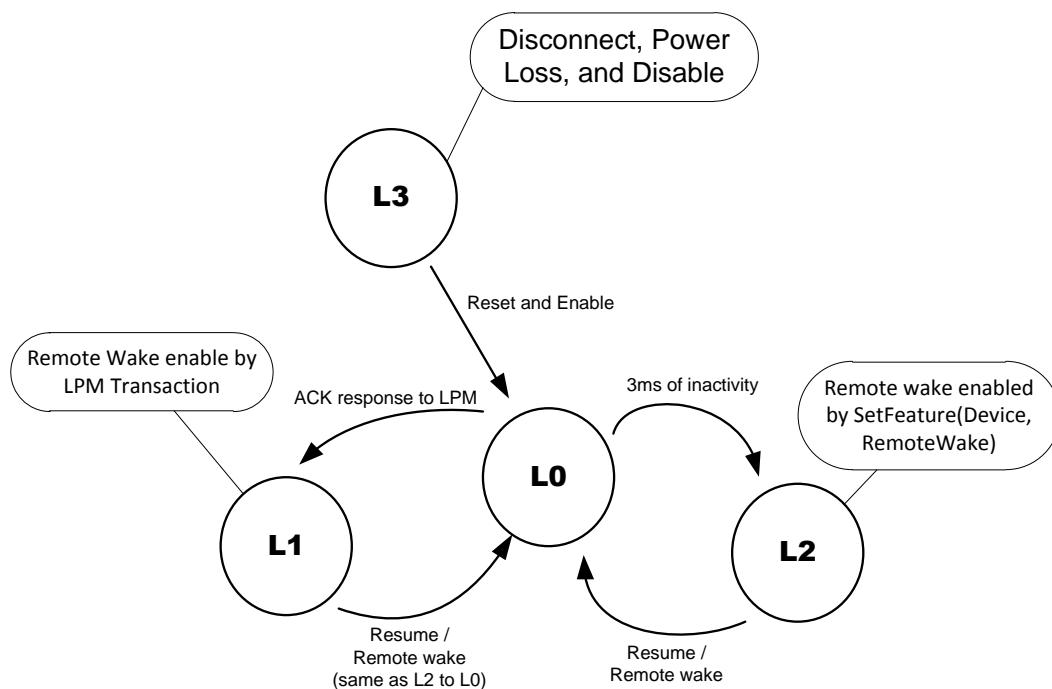


图 6.30-6 LPM 状态转换图

## 6.30.5 寄存器映射

R: 只读, W: 只写, R/W: 读/写

寄存器	偏移地址	R/W	描述	复位值
<b>USBD 基地址:</b>				
<b>USBD_BA = 0x400C_0000</b>				
<b>USBD_INTEN</b>	USBD_BA+0x000	R/W	USB 设备中断使能寄存器	0x0000_0000
<b>USBD_INTSTS</b>	USBD_BA+0x004	R/W	USB 设备中断事件状态寄存器	0x0000_0000
<b>USBD_FADDR</b>	USBD_BA+0x008	R/W	USB 设备功能地址寄存器	0x0000_0000
<b>USBD_EPSTS</b>	USBD_BA+0x00C	R	USB 设备端点状态寄存器	0x0000_0000
<b>USBD_ATTR</b>	USBD_BA+0x010	R/W	USB 设备总线状态和属性寄存器	0x0000_0040
<b>USBD_VBUSDET</b>	USBD_BA+0x014	R	USB 设备 VBUS 检测寄存器	0x0000_0000
<b>USBD_STBUFSEG</b>	USBD_BA+0x018	R/W	SETUP 令牌 包缓存段寄存器	0x0000_0000
<b>USBD_EPSTS0</b>	USBD_BA+0x020	R	USB 设备端点状态寄存器0	0x0000_0000
<b>USBD_EPSTS1</b>	USBD_BA+0x024	R	USB 设备端点状态寄存器1	0x0000_0000
<b>USBD_LPMATTR</b>	USBD_BA+0x088	R	USB LPM 属性寄存器	0x0000_0000
<b>USBD_FN</b>	USBD_BA+0x08C	R	USB 帧号寄存器	0x0000_0XXX
<b>USBD_SE0</b>	USBD_BA+0x090	R/W	USB 设备驱动 SE0 控制寄存器	0x0000_0001
<b>USBD_BUFSEG0</b>	USBD_BA+0x500	R/W	端点0缓存段寄存器	0x0000_0000
<b>USBD_MXPLD0</b>	USBD_BA+0x504	R/W	端点0 最大载荷寄存器	0x0000_0000
<b>USBD_CFG0</b>	USBD_BA+0x508	R/W	端点0 配置寄存器	0x0000_0000
<b>USBD_CFGP0</b>	USBD_BA+0x50C	R/W	端点0设置Stall和清除In/Out 准备控制寄存器	0x0000_0000
<b>USBD_BUFSEG1</b>	USBD_BA+0x510	R/W	端点1缓存段寄存器	0x0000_0000
<b>USBD_MXPLD1</b>	USBD_BA+0x514	R/W	端点1 最大载荷寄存器	0x0000_0000
<b>USBD_CFG1</b>	USBD_BA+0x518	R/W	端点1 配置寄存器	0x0000_0000
<b>USBD_CFGP1</b>	USBD_BA+0x51C	R/W	端点1设置Stall和清除In/Out 准备控制寄存器	0x0000_0000
<b>USBD_BUFSEG2</b>	USBD_BA+0x520	R/W	端点2缓存段寄存器	0x0000_0000
<b>USBD_MXPLD2</b>	USBD_BA+0x524	R/W	端点2 最大载荷寄存器	0x0000_0000
<b>USBD_CFG2</b>	USBD_BA+0x528	R/W	端点2 配置寄存器	0x0000_0000
<b>USBD_CFGP2</b>	USBD_BA+0x52C	R/W	端点2设置Stall和清除In/Out 准备控制寄存器	0x0000_0000
<b>USBD_BUFSEG3</b>	USBD_BA+0x530	R/W	端点3缓存段寄存器	0x0000_0000
<b>USBD_MXPLD3</b>	USBD_BA+0x534	R/W	端点3 最大载荷寄存器	0x0000_0000
<b>USBD_CFG3</b>	USBD_BA+0x538	R/W	端点3 配置寄存器	0x0000_0000
<b>USBD_CFGP3</b>	USBD_BA+0x53C	R/W	端点3设置Stall和清除In/Out 准备控制寄存器	0x0000_0000

<b>USBD_BUFSEG4</b>	USBD_BA+0x540	R/W	端点4缓存段寄存器	0x0000_0000
<b>USBD_MXPLD4</b>	USBD_BA+0x544	R/W	端点4 最大载荷寄存器	0x0000_0000
<b>USBD_CFG4</b>	USBD_BA+0x548	R/W	端点4 配置寄存器	0x0000_0000
<b>USBD_CFGP4</b>	USBD_BA+0x54C	R/W	端点4设置Stall和清除In/Out 准备控制寄存器	0x0000_0000
<b>USBD_BUFSEG5</b>	USBD_BA+0x550	R/W	端点5缓存段寄存器	0x0000_0000
<b>USBD_MXPLD5</b>	USBD_BA+0x554	R/W	端点5 最大载荷寄存器	0x0000_0000
<b>USBD_CFG5</b>	USBD_BA+0x558	R/W	端点5 配置寄存器	0x0000_0000
<b>USBD_CFGP5</b>	USBD_BA+0x55C	R/W	端点5设置Stall和清除In/Out 准备控制寄存器	0x0000_0000
<b>USBD_BUFSEG6</b>	USBD_BA+0x560	R/W	端点6缓存段寄存器	0x0000_0000
<b>USBD_MXPLD6</b>	USBD_BA+0x564	R/W	端点6最大载荷寄存器	0x0000_0000
<b>USBD_CFG6</b>	USBD_BA+0x568	R/W	端点6配置寄存器	0x0000_0000
<b>USBD_CFGP6</b>	USBD_BA+0x56C	R/W	端点6设置Stall和清除In/Out 准备控制寄存器	0x0000_0000
<b>USBD_BUFSEG7</b>	USBD_BA+0x570	R/W	端点7缓存段寄存器	0x0000_0000
<b>USBD_MXPLD7</b>	USBD_BA+0x574	R/W	端点7最大载荷寄存器	0x0000_0000
<b>USBD_CFG7</b>	USBD_BA+0x578	R/W	端点7配置寄存器	0x0000_0000
<b>USBD_CFGP7</b>	USBD_BA+0x57C	R/W	端点7设置Stall和清除In/Out 准备控制寄存器	0x0000_0000
<b>USBD_BUFSEG8</b>	USBD_BA+0x580	R/W	端点8缓存段寄存器	0x0000_0000
<b>USBD_MXPLD8</b>	USBD_BA+0x584	R/W	端点8最大载荷寄存器	0x0000_0000
<b>USBD_CFG8</b>	USBD_BA+0x588	R/W	端点8配置寄存器	0x0000_0000
<b>USBD_CFGP8</b>	USBD_BA+0x58C	R/W	端点8设置Stall和清除In/Out 准备控制寄存器	0x0000_0000
<b>USBD_BUFSEG9</b>	USBD_BA+0x590	R/W	端点9缓存段寄存器	0x0000_0000
<b>USBD_MXPLD9</b>	USBD_BA+0x594	R/W	端点9最大载荷寄存器	0x0000_0000
<b>USBD_CFG9</b>	USBD_BA+0x598	R/W	端点9配置寄存器	0x0000_0000
<b>USBD_CFGP9</b>	USBD_BA+0x59C	R/W	端点9设置Stall和清除In/Out 准备控制寄存器	0x0000_0000
<b>USBD_BUFSEG10</b>	USBD_BA+0x5A0	R/W	端点10缓存段寄存器	0x0000_0000
<b>USBD_MXPLD10</b>	USBD_BA+0x5A4	R/W	端点10最大载荷寄存器	0x0000_0000
<b>USBD_CFG10</b>	USBD_BA+0x5A8	R/W	端点10配置寄存器	0x0000_0000
<b>USBD_CFGP10</b>	USBD_BA+0x5AC	R/W	端点10设置Stall和清除In/Out 准备控制寄存器	0x0000_0000
<b>USBD_BUFSEG11</b>	USBD_BA+0x5B0	R/W	端点11缓存段寄存器	0x0000_0000
<b>USBD_MXPLD11</b>	USBD_BA+0x5B4	R/W	端点11最大载荷寄存器	0x0000_0000
<b>USBD_CFG11</b>	USBD_BA+0x5B8	R/W	端点11配置寄存器	0x0000_0000

USBD_CFGP11	USBD_BA+0x5BC	R/W	端点11设置Stall和清除In/Out准备控制寄存器	0x0000_0000
-------------	---------------	-----	-----------------------------	-------------

内存类别	地址	大小	描述
<b>USBD_BA = 0x400C_0000</b>			
USBD_SRAM	USBD_BA+0x100 ~ USBD_BA+0x4FF	1024 Bytes	SRAM用于整个端点缓冲区。详细内容请参考段7.29.5.7, 端点SRAM结构和描述

### 6.30.6 寄存器描述

#### USBD\_INTEN USB中断使能寄存器

寄存器	偏移地址	R/W	描述	复位值
USBD_INTEN	USBD_BA+0x000	R/W	USB设备中断使能寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
INNAKEN	Reserved						WKEN
7	6	5	4	3	2	1	0
Reserved			SOFIEN	NEVWKIEN	VBDETIEN	USBIEN	BUSIEN

位	描述	
[31:16]	Reserved	保留
[15]	INNAKEN	<p>收到IN Token时，激活的 NAK功能及状态</p> <p>0 = 当从设备收到 IN token 应答 NAK 时，IN NAK 状态不会被更新到端点状态寄存器 USBD_EPSTS0 和 USBD_EPSTS1，所以不会产生USB中断事件。</p> <p>1 = 当从设备收到 IN token 时应答 NAK，IN NAK 的状态被更新到端点状态寄存器 USBD_EPSTS0 和 USBD_EPSTS1，并发生 USB 中断事件。</p>
[14:9]	Reserved	保留
[8]	WKEN	<p>唤醒功能使能位</p> <p>0 = USB 唤醒功能禁止</p> <p>1 = USB 唤醒功能使能</p>
[7:5]	Reserved	保留
[4]	SOFIEN	<p>开始帧中断使能位</p> <p>0 = SOF 中断禁止</p> <p>1 = SOF 中断使能</p>
[3]	NEVWKIEN	<p>USB 无事件唤醒中断使能位</p> <p>0 = 无事件唤醒中断禁止</p> <p>1 = 无事件唤醒中断使能</p>
[2]	VBDETIEN	<p>VBUS检测中断使能位</p> <p>0 = VBUS检测中断禁止</p> <p>1 = VBUS检测中断使能</p>
[1]	USBIEN	<p>USB事件中断使能位</p> <p>0 = USB事件中断禁止</p>

		1 = USB事件中断使能
[0]	<b>BUSIEN</b>	<b>Bus 事件中断使能位</b> 0 = BUS 事件中断禁止 1 = BUS 事件中断使能

USBD\_INTSTS USB中断事件状态寄存器

寄存器	偏移地址	R/W	描述	复位值
USBD_INTSTS	USBD_BA+0x004	R/W	USB设备中断事件状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
SETUP	Reserved			EPEVT11	EPEVT10	EPEVT9	EPEVT8
23	22	21	20	19	18	17	16
EPEVT7	EPEVT6	EPEVT5	EPEVT4	EPEVT3	EPEVT2	EPEVT1	EPEVT0
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved			SOFIF	NEVWKIF	VBDETIF	USBIF	BUSIF

位	描述	
[31]	SETUP	<b>Setup 事件状态</b> 0 = 没有SETUP事件 1 = 发生SETUP事件, 写1到 USBD_INTSTS[31]位清零.
[30:28]	Reserved	保留
[27]	EPEVT11	<b>端点 11的USB 事件状态</b> 0 = 端点11没有事件发生 1 = 端点11有USB 事件发生, 检查USBD_EPSTS1[ 15:12]位可以知道发生了哪种USB事件, 可以对USBD_INTSTS[27]位或USBD_INTSTS[1]位写1来清零.
[26]	EPEVT10	<b>端点 10的USB 事件状态</b> 0 = 端点10没有事件发生 1 = 端点10有USB 事件发生, 检查USBD_EPSTS1[11 :8]位可以知道发生了哪种USB事件, 可以对USBD_INTSTS[26]位或USBD_INTSTS[1]位写1来清零.
[25]	EPEVT9	<b>端点 9的USB 事件状态</b> 0 = 端点9没有事件发生 1 = 端点9有USB 事件发生, 检查USBD_EPSTS1[7 :4]位可以知道发生了哪种USB事件, 可以对USBD_INTSTS[25]位或USBD_INTSTS[1]位写1来清零.
[24]	EPEVT8	<b>端点 8的USB 事件状态</b> 0 = 端点8没有事件发生 1 = 端点8有USB 事件发生, 检查USBD_EPSTS1[3 :0]位可以知道发生了哪种USB事件, 可以对USBD_INTSTS[24]位或USBD_INTSTS[1]位写1来清零.
[23]	EPEVT7	<b>端点 7的USB 事件状态</b> 0 = 端点7没有事件发生 1 = 端点7有USB 事件发生, 检查USBD_EPSTS0[31:28]位可以知道发生了哪种USB事件, 可以对USBD_INTSTS[23]位或USBD_INTSTS[1]位写1来清零.

[22]	<b>EPEVT6</b>	<b>端点 6的USB 事件状态</b> 0 =端点6没有事件发生. 1 =端点6有USB 事件发生, 检查USBD_EPSTS0[27:24]位可以知道发生了哪种USB事件, 可以对 USBD_INTSTS[22] 位或 USBD_INTSTS[1]位写1来清零
[21]	<b>EPEVT5</b>	<b>端点 5的USB 事件状态</b> 0 =端点5没有事件发生 1 =端点5有USB 事件发生, 检查USBD_EPSTS0[23:20]位可以知道发生了哪种USB事件, 可以对 USBD_INTSTS[21] 位或 USBD_INTSTS[1]位写1来清零
[20]	<b>EPEVT4</b>	<b>端点4的USB 事件状态</b> 0 =端点4没有事件发生 1 =端点4有USB 事件发生, 检查USBD_EPSTS0[19:16]位可以知道发生了哪种USB事件, 可以对USBD_INTSTS[20] 位或 USBD_INTSTS[1]位写1来清零
[19]	<b>EPEVT3</b>	<b>端点 3的USB 事件状态</b> 0 =端点3没有事件发生 1 =端点3有USB 事件发生, 检查USBD_EPSTS0[15:12]位可以知道发生了哪种USB事件,, 可以对USBD_INTSTS[19] 位或USBD_INTSTS[1]位写1来清零
[18]	<b>EPEVT2</b>	<b>端点2的USB 事件状态</b> 0 =端点2没有事件发生 1 =端点2有USB 事件发生, 检查USBD_EPSTS0[11:8]位可以知道发生了哪种USB事件, 可以对USBD_INTSTS[18] 位或USBD_INTSTS[1]位写1来清零
[17]	<b>EPEVT1</b>	<b>端点 1的USB 事件状态</b> 0 =端点1没有事件发生 1 =端点1有USB 事件发生, 检查USBD_EPSTS0[7:4]位可以知道发生了哪种USB事件, 可以对USBD_INTSTS[17] 位或USBD_INTSTS[1]位写1来清零
[16]	<b>EPEVT0</b>	<b>端点 0的USB 事件状态</b> 0 =端点0没有事件发生. 1 =端点0有USB 事件发生, 检查USBD_EPSTS0[3:0]位可以知道发生了哪种USB事件, 可以对USBD_INTSTS[16] 位或USBD_INTSTS[1]位写1来清零
[15:5]	<b>Reserved</b>	保留
[4]	<b>SOFIF</b>	<b>开始帧中断状态</b> 0 = 没有SOF 事件发生 1 =有SOF事件发生, 对USBD_INTSTS[4]位写1来清零
[3]	<b>NEVWKIF</b>	<b>无事件唤醒中断状态</b> 0 = 没有唤醒事件发生 1 = 有无唤醒事件发生, 对USBD_INTSTS[3]位写1来清零
[2]	<b>VBDETIF</b>	<b>VBUS检测中断状态</b> 0 = 没有USB设备连接/分离事件发生 1 = USB总线上有连接/分离事件发生, 对USBD_INTSTS[2]位写1清零
[1]	<b>USBIF</b>	<b>USB 中断事件状态</b> USB 事件包括在总线上的SETUP Token, IN Token, OUT ACK, ISO IN, 或 ISO OUT 事件 0 = 没有USB事件发生 1 = 有 USB 事件发生, 检查 EPSTS0~5[2:0] 位可以知道发生了哪种USB 事件。对 USBD_INTSTS[1] 位或 对EPSTS0~11 和SETUP (USBD_INTSTS[31])位写1清零

[0]	<b>BUSIF</b>	<b>BUS中断事件状态</b> BUS中断事件说明总线上发生了挂起或恢复功能. 0 = 没有BUS中断事件发生 1 = 发生了Bus中断事件; 检查 USBD_ATTR[3:0] 位可以知道发生了哪种BUS中断事件, 对 USBD_INTSTS[0]位写1清零
-----	--------------	--

**USBD\_FADDR USB 设备功能地址寄存器**

7位有效数据用于设置USB总线上设备地址

寄存器	偏移地址	R/W	描述	复位值
USBD_FADDR	USBD_BA+0x008	R/W	USB 设备功能地址寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved	FADDR						

位	描述	
[31:7]	Reserved	保留
[6:0]	FADDR	USB 设备功能地址

USBD\_EPSTS USB端点状态寄存器

寄存器	偏移地址	R/W	描述	复位值
USBD_EPSTS	USBD_BA+0x00C	R	USB设备端点状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
OV	Reserved						

位	描述	
[31:8]	Reserved	保留
[7]	OV	<b>Overrun</b> 用于指示所接收数据长度是否超过最大负荷范围 0 = 未超出。 1 = 主机发送数据超出MXPLD 寄存器设置范围，或Setup 数据超过8 个字节
[6:0]	Reserved	保留

**USBD\_ATTR USB总线状态和属性寄存器**

寄存器	偏移地址	R/W	描述	复位值
USBD_ATTR	USBD_BA+0x010	R/W	USB设备总线状态和属性寄存器	0x0000_0040

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved		L1RESUME	L1SUSPEND	LPMACK	BYTEM	Reserved	DPPUEN
7	6	5	4	3	2	1	0
USBEN	Reserved	RWAKEUP	PHYEN	TOUT	RESUME	SUSPEND	USBRST

位	描述	
[31:14]	Reserved	保留
[13]	L1RESUME	<b>LPM L1 恢复</b> 0 = 总线没有LPM L1状态恢复 1 = LPM L1状态从挂起恢复 <b>注:</b> 该位只读
[12]	L1SUSPEND	<b>LPM L1 挂起</b> 0 = 总线没有LPM L1状态挂起 1 = 该位在LPM命令进入L1状态被成功接收和应答的时候由硬件置位。 <b>注:</b> 该位只读
[11]	LPMACK	<b>LPM Token 应答使能位</b> NYET/ACK仅在一个成功的LPM事务后被返回也就是在EXT token 和LPM token和有效的bLinkState = 0001 (L1)被接收时没有出错, 否则ERROR 和 STALL分别会自动地被返回 0= 有效的 LPM Token 不会被应答 1= 有效的 LPM Token 会被应答
[10]	BYTEM	<b>CPU 存取 USB SRAM 大小模式选择</b> 0 = 字模式: CPU到 USB SRAM 的数据传输必须以字为单位 1 = 字节模式: CPU到 USB SRAM 的数据传输必须以字节为单位
[9]	Reserved	保留
[8]	DPPUEN	<b>USB_DP管脚上拉电阻使能位</b> 0 = USB_D+ 脚上的上拉电阻禁止. 1 = USB_D+ 脚上的上拉电阻打开.
[7]	USBEN	<b>USB 控制器使能位</b> 0 = USB 控制器禁止. 1 = USB 控制器使能

[6]	<b>Reserved</b>	保留
[5]	<b>RWAKEUP</b>	<b>远程唤醒</b> 0 = 把USB总线从 K 状态释放 1 = 强迫USB 总线到 K (USB_D+ low, USB_D- high) 状态, 用于远程唤醒
[4]	<b>PHYEN</b>	<b>PHY 收发器使能位</b> 0 = PHY 收发器功能禁止. 1 = PHY收发器功能使能.
[3]	<b>TOUT</b>	<b>时间溢出状态</b> 0 = 没有时间溢出情况发生 1 = 超过18个位长度计数器时间, 总线上仍没有数据回复. <b>注意:</b> 该位只读
[2]	<b>RESUME</b>	<b>总线恢复状态</b> 0 =总线没有恢复. 1 = 总线从挂起状态恢复. <b>注意:</b> 该位只读.
[1]	<b>SUSPEND</b>	<b>挂起状态</b> 0 =总线没有挂起事件发生. 1 = 总线闲置状态超过3ms, 有可能是从设备被拔下或主机进入了睡眠模式 <b>注意:</b> 该位只读.
[0]	<b>USBRST</b>	<b>USB 总线复位状态</b> 0 =总线没有复位 1 =总线有复位, 此时SE0 (single-ended 0)已超过 2.5us. <b>注意:</b> 该位只读.

**USBD\_VBUSDET USB 设备VBUS检测寄存器**

寄存器	偏移地址	R/W	描述	复位值
USBD_VBUSDET	USBD_BA+0x014	R	USB 设备 VBUS 检测寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved							VBUSDET

位	描述	
[31:1]	Reserved	保留
[0]	VBUSDET	设备VBUS检测状态 0 = 控制器没有连接到USB主机 1 = 控制器连接到到USB主机

**USBD\_STBUFSEG USB SETUP令牌缓存段寄存器**

寄存器	偏移地址	R/W	描述	复位值
USBD_STBUFSEG	USBD_BA+0x018	R/W	SETUP令牌包缓存段寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
STBUFSEG					Reserved		

位	描述	
[31:9]	Reserved	保留
[8:3]	STBUFSEG	<p><b>Setup 令牌包缓存段设置位</b></p> <p>用于指示 SETUP 令牌包在USB设备SRAM中的起始地址，它的有效起始地址是 USBD_SRAM 地址 + {STBUFSEG[8:3], 3'b000}</p> <p>此处USBD_SRAM 地址 = USBD_BA+0x100h.</p> <p><b>注意:</b>该内容只适用于SETUP令牌包.</p>
[2:0]	Reserved	保留

**USBD\_EPSTS0 USB端点状态寄存器0**

寄存器	偏移地址	R/W	描述	复位值
USBD_EPSTS0	USBD_BA+0x020	R	USB 设备端点状态寄存器0	0x0000_0000

31	30	29	28	27	26	25	24
EPSTS7				EPSTS6			
23	22	21	20	19	18	17	16
EPSTS5				EPSTS4			
15	14	13	12	11	10	9	8
EPSTS3				EPSTS2			
7	6	5	4	3	2	1	0
EPSTS1				EPSTS0			

位	描述
[31:28]	<b>EPSTS7</b>  端点7状态 这些位用于指示该端点的当前状态 0000 = In ACK. 0001 = In NAK. 0010 = Out Packet Data0 ACK. 0011 = Setup ACK. 0110 = Out Packet Data1 ACK. 0111 = 同步传输结束.
[27:24]	<b>EPSTS6</b>  端点6状态 这些位用于指示该端点的当前状态 0000 = In ACK. 0001 = In NAK. 0010 = Out Packet Data0 ACK. 0011 = Setup ACK. 0110 = Out Packet Data1 ACK. 0111 = 同步传输结束.
[23:20]	<b>EPSTS5</b>  端点5状态 这些位用于指示该端点的当前状态 0000 = In ACK. 0001 = In NAK. 0010 = Out Packet Data0 ACK. 0011 = Setup ACK. 0110 = Out Packet Data1 ACK. 0111 = 同步传输结束.

[19:16]	EPSTS4	<b>端点4状态</b> 这些位用于指示该端点的当前状态 0000 = In ACK. 0001 = In NAK. 0010 = Out Packet Data0 ACK. 0011 = Setup ACK. 0110 = Out Packet Data1 ACK. 0111 =同步传输结束.
[15:12]	EPSTS3	<b>端点3状态</b> 这些位用于指示该端点的当前状态 0000 = In ACK. 0001 = In NAK. 0010 = Out Packet Data0 ACK. 0011 = Setup ACK. 0110 = Out Packet Data1 ACK. 0111 =同步传输结束.
[11:8]	EPSTS2	<b>端点2状态</b> 这些位用于指示该端点的当前状态 0000 = In ACK. 0001 = In NAK. 0010 = Out Packet Data0 ACK. 0011 = Setup ACK. 0110 = Out Packet Data1 ACK. 0111 =同步传输结束.
[7:4]	EPSTS1	<b>端点1状态</b> 这些位用于指示该端点的当前状态 0000 = In ACK. 0001 = In NAK. 0010 = Out Packet Data0 ACK. 0011 = Setup ACK. 0110 = Out Packet Data1 ACK. 0111 =同步传输结束.
[3:0]	EPSTS0	<b>端点0状态</b> 这些位用于指示该端点的当前状态 0000 = In ACK. 0001 = In NAK. 0010 = Out Packet Data0 ACK. 0011 = Setup ACK. 0110 = Out Packet Data1 ACK. 0111 =同步传输结束.

USBD\_EPSTS1 USB端点状态寄存器1

寄存器	偏移地址	R/W	描述	复位值
USBD_EPSTS1	USBD_BA+0x024	R	USB 设备端点状态寄存器1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
EPSTS11				EPSTS10			
7	6	5	4	3	2	1	0
EPSTS9				EPSTS8			

位	描述	
[31:16]	Reserved	保留
[15:12]	EPSTS11	<p><b>端点11 状态</b>            这些位用于指示该端点的当前状态            0000 = In ACK.            0001 = In NAK.            0010 = Out Packet Data0 ACK.            0011 = Setup ACK.            0110 = Out Packet Data1 ACK.            0111 = 同步传输结束.</p>
[11:8]	EPSTS10	<p><b>端点10 状态</b>            这些位用于指示该端点的当前状态            0000 = In ACK.            0001 = In NAK.            0010 = Out Packet Data0 ACK.            0011 = Setup ACK.            0110 = Out Packet Data1 ACK.            0111 = 同步传输结束.</p>
[7:4]	EPSTS9	<p><b>端点9状态</b>            这些位用于指示该端点的当前状态            0000 = In ACK.            0001 = In NAK.            0010 = Out Packet Data0 ACK.            0011 = Setup ACK.            0110 = Out Packet Data1 ACK.            0111 = 同步传输结束.</p>

[3:0]	<b>EPSTS8</b>	<b>端点8状态</b> 这些位用于指示该端点的当前状态 0000 = In ACK. 0001 = In NAK. 0010 = Out Packet Data0 ACK. 0011 = Setup ACK. 0110 = Out Packet Data1 ACK. 0111 =同步传输结束.
-------	---------------	---

**USBD\_LPMATTR USB LPM属性寄存器**

寄存器	偏移地址	R/W	描述	复位值
USBD_LPMATTR	USBD_BA+0x088	R	USB LPM属性寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
LPMBESL				LPMLINKSTS			

位	描述	
[31:9]	Reserved	保留
[8]	LPMRWAKUP	<b>LPM 远程唤醒</b> 该位包含接收的最后ACK LPM Token的远程唤醒值
[7:4]	LPMBESL	<b>LPM最大力度服务延时</b> 该位包含接收的最后ACK LPM Token的BESL值
[3:0]	LPMLINKSTS	<b>LPM 链路状态</b> 该位包含接收的最后ACK LPM Token的链路状态

USBD\_FN USB 帧号寄存器

寄存器	偏移地址	R/W	描述	复位值
USBD_FN	USBD_BA+0x08C	R	USB 帧号寄存器	0x0000_0XXX

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved					FN		
7	6	5	4	3	2	1	0
FN							

位	描述	
[31:11]	Reserved	保留.
[10:0]	FN	<b>帧编号</b> 该域包含了最近一次收到的SOF包的11位的帧号

**USBD\_SE0 USB驱动SE0寄存器**

寄存器	偏移地址	R/W	描述	复位值
USBD_SE0	USBD_BA+0x090	R/W	USB 驱动SE0控制寄存器	0x0000_0001

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved							<b>SE0</b>

位	描述	
[31:1]	Reserved	保留.
[0]	SE0	在USB总线上驱动单端0控制位 当USB_D+ 和 USB_D-都拉低时, 为单端0 (SE0)。 0 = 正常操作 1 = 强制USB PHY 收发器驱动SE0.

**USB BUFSEGx USB缓存段寄存器**

寄存器	偏移地址	R/W	描述	复位值
<b>USBD_BUFSEG0</b>	USBD_BA+0x500	R/W	端点0 缓存段寄存器	0x0000_0000
<b>USBD_BUFSEG1</b>	USBD_BA+0x510	R/W	端点1 缓存段寄存器	0x0000_0000
<b>USBD_BUFSEG2</b>	USBD_BA+0x520	R/W	端点2 缓存段寄存器	0x0000_0000
<b>USBD_BUFSEG3</b>	USBD_BA+0x530	R/W	端点3 缓存段寄存器	0x0000_0000
<b>USBD_BUFSEG4</b>	USBD_BA+0x540	R/W	端点4 缓存段寄存器	0x0000_0000
<b>USBD_BUFSEG5</b>	USBD_BA+0x550	R/W	端点5 缓存段寄存器	0x0000_0000
<b>USBD_BUFSEG6</b>	USBD_BA+0x560	R/W	端点6 缓存段寄存器	0x0000_0000
<b>USBD_BUFSEG7</b>	USBD_BA+0x570	R/W	端点7 缓存段寄存器	0x0000_0000
<b>USBD_BUFSEG8</b>	USBD_BA+0x580	R/W	端点8 缓存段寄存器	0x0000_0000
<b>USBD_BUFSEG9</b>	USBD_BA+0x590	R/W	端点9 缓存段寄存器	0x0000_0000
<b>USBD_BUFSEG10</b>	USBD_BA+0x5A0	R/W	端点10 缓存段寄存器	0x0000_0000
<b>USBD_BUFSEG11</b>	USBD_BA+0x5B0	R/W	端点11 缓存段寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							BUFSEG
7	6	5	4	3	2	1	0
BUFSEG					Reserved		

位	描述	
[31:9]	Reserved	保留.
[8:3]	BUFSEG	<p><b>端点缓存段设置位</b>            用于指示每个端点在USB SRAM 的偏移地址，端点的有效起始地址是            USBD_SRAM 地址 + { BUFSEG[8:3], 3'b000 }            此处USBD_SRAM 地址 = USBD_BA+0x100h.            请参考段6.30.4.7，查阅端点SRAM的结构和相关描述</p>
[2:0]	Reserved	保留.

**USB MXPLDx USB最大负荷设置寄存器**

寄存器	偏移地址	R/W	描述	复位值
<b>USBD_MXPLD0</b>	USBD_BA+0x504	R/W	端点0 最大负荷设置寄存器	0x0000_0000
<b>USBD_MXPLD1</b>	USBD_BA+0x514	R/W	端点1 最大负荷设置寄存器	0x0000_0000
<b>USBD_MXPLD2</b>	USBD_BA+0x524	R/W	端点2 最大负荷设置寄存器	0x0000_0000
<b>USBD_MXPLD3</b>	USBD_BA+0x534	R/W	端点3 最大负荷设置寄存器	0x0000_0000
<b>USBD_MXPLD4</b>	USBD_BA+0x544	R/W	端点4 最大负荷设置寄存器	0x0000_0000
<b>USBD_MXPLD5</b>	USBD_BA+0x554	R/W	端点5 最大负荷设置寄存器	0x0000_0000
<b>USBD_MXPLD6</b>	USBD_BA+0x564	R/W	端点6 最大负荷设置寄存器	0x0000_0000
<b>USBD_MXPLD7</b>	USBD_BA+0x574	R/W	端点7 最大负荷设置寄存器	0x0000_0000
<b>USBD_MXPLD8</b>	USBD_BA+0x584	R/W	端点8 最大负荷设置寄存器	0x0000_0000
<b>USBD_MXPLD9</b>	USBD_BA+0x594	R/W	端点9 最大负荷设置寄存器	0x0000_0000
<b>USBD_MXPLD10</b>	USBD_BA+0x5A4	R/W	端点10 最大负荷设置寄存器	0x0000_0000
<b>USBD_MXPLD11</b>	USBD_BA+0x5B4	R/W	端点11 最大负荷设置寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
MXPLD							

位	描述	
[31:9]	Reserved	保留.
[8:0]	MXPLD	<p><b>最大有效负荷位</b>            定义了发送到主机(IN token)的实际数据长度, 或 从主机接收到数据(OUT token)的实际长度. 也用于指示IN token时做好了发送数据准备, 或OUT token时做好了接收数据准备.</p> <p>(1) 当CPU写入值到该寄存器后,            对 IN token, MXPLD寄存器的值用于指示要发送的数据长度, 且已做好发送准备.            对OUT token, 设备已经做好了从主机接收数据的准备。MXPLD的值就表示从主机所接收数据的最大长度。            (2) 当CPU读该寄存器时,</p>

		对IN token, MXPLD 的值表示要发送到主机的数据长度 对OUT token, MXPLD 的值表示从主机接收到的实际数据长度 <b>注意:</b> 一旦 MXPLD 的值被写入, 收到IN/OUT token后, 数据包可以立即收发
--	--	--

**USB CFGx USB配置寄存器**

寄存器	偏移地址	R/W	描述	复位值
<b>USBD_CFG0</b>	USBD_BA+0x508	R/W	端点0 配置寄存器	0x0000_0000
<b>USBD_CFG1</b>	USBD_BA+0x518	R/W	端点1 配置寄存器	0x0000_0000
<b>USBD_CFG2</b>	USBD_BA+0x528	R/W	端点2 配置寄存器	0x0000_0000
<b>USBD_CFG3</b>	USBD_BA+0x538	R/W	端点3 配置寄存器	0x0000_0000
<b>USBD_CFG4</b>	USBD_BA+0x548	R/W	端点4 配置寄存器	0x0000_0000
<b>USBD_CFG5</b>	USBD_BA+0x558	R/W	端点5 配置寄存器	0x0000_0000
<b>USBD_CFG6</b>	USBD_BA+0x568	R/W	端点6 配置寄存器	0x0000_0000
<b>USBD_CFG7</b>	USBD_BA+0x578	R/W	端点7 配置寄存器	0x0000_0000
<b>USBD_CFG8</b>	USBD_BA+0x588	R/W	端点8 配置寄存器	0x0000_0000
<b>USBD_CFG9</b>	USBD_BA+0x598	R/W	端点9 配置寄存器	0x0000_0000
<b>USBD_CFG10</b>	USBD_BA+0x5A8	R/W	端点10 配置寄存器	0x0000_0000
<b>USBD_CFG11</b>	USBD_BA+0x5B8	R/W	端点11 配置寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved						CSTALL	Reserved
7	6	5	4	3	2	1	0
DSQSYNC	STATE		ISOCH	EPNUM			

位	描述	
[31:10]	Reserved	保留.
[9]	CSTALL	清STALL 响应位 0 = 在setup阶段禁止设备清除STALL 1 = 在setup阶段应许设备清除STALL
[8]	Reserved	保留.
[7]	DSQSYNC	数据时序同步位 0 = DATA0 PID. 1 = DATA1 PID. <b>注意:</b> 该位用于指定在接下来的IN token 传输过程是DATA0还是DATA1 PID. 在IN token

		过程，硬件会基于该位硬件自动触发选择
[6:5]	<b>STATE</b>	<b>端点状态</b> 00 =端点被禁止 01 =输出端点 10 =输入端点 11 =未定义.
[4]	<b>ISOCH</b>	<b>同步端点设置</b> 该位用于设置该端点为同步端点。无握手信号 0 = 非同步端点. 1 = 同步端点
[3:0]	<b>EPNUM</b>	<b>端点号</b> 用于定义当前端点的端点号

USB CFGPx USB扩展配置寄存器

寄存器	偏移地址	R/W	描述	复位值
<b>USBD_CFGP0</b>	USBD_BA+0x50C	R/W	端点0设置Stall 和清In/Out 准备控制寄存器	0x0000_0000
<b>USBD_CFGP1</b>	USBD_BA+0x51C	R/W	端点1设置Stall 和清In/Out 准备控制寄存器	0x0000_0000
<b>USBD_CFGP2</b>	USBD_BA+0x52C	R/W	端点2设置Stall 和清In/Out 准备控制寄存器	0x0000_0000
<b>USBD_CFGP3</b>	USBD_BA+0x53C	R/W	端点3设置Stall 和清In/Out 准备控制寄存器	0x0000_0000
<b>USBD_CFGP4</b>	USBD_BA+0x54C	R/W	端点4设置Stall 和清In/Out 准备控制寄存器	0x0000_0000
<b>USBD_CFGP5</b>	USBD_BA+0x55C	R/W	端点5设置Stall 和清In/Out 准备控制寄存器	0x0000_0000
<b>USBD_CFGP6</b>	USBD_BA+0x56C	R/W	端点6设置Stall 和清In/Out 准备控制寄存器	0x0000_0000
<b>USBD_CFGP7</b>	USBD_BA+0x57C	R/W	端点7设置Stall 和清In/Out 准备控制寄存器	0x0000_0000
<b>USBD_CFGP8</b>	USBD_BA+0x58C	R/W	端点8设置Stall 和清In/Out 准备控制寄存器	0x0000_0000
<b>USBD_CFGP9</b>	USBD_BA+0x59C	R/W	端点9设置Stall 和清In/Out 准备控制寄存器	0x0000_0000
<b>USBD_CFGP10</b>	USBD_BA+0x5AC	R/W	端点10设置Stall 和清In/Out 准备控制寄存器	0x0000_0000
<b>USBD_CFGP11</b>	USBD_BA+0x5BC	R/W	端点11设置Stall 和清In/Out 准备控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved						SSTALL	CLRRDY

位	描述	
[31:2]	Reserved	保留.
[1]	SSTALL	<b>STALL设置位</b> 0 = 禁止设备响应STALL. 1 = 设置设备自动响应STALL
[0]	CLRRDY	<b>清除准备位</b> 当USBD_MXPLDx寄存器被设置后，表示该端点准备好可以发送或接收数据。如果用户想在传输开始前关闭传输，需要设置该位为1来进行关闭，该位会自动清零。 对IN token，写‘1’清除 IN token 时发送数据到 USB 的准备信号 对OUT token，写‘1’清除 OUT token 时从 USB 接收数据的准备信号

		该位只能写 1，读数据返回值总是 0。
--	--	---------------------

## 6.31 HSUSBD 高速 USB2.0 设备接口

### 6.31.1 概述

USB 设备控制器与 AHB 总线和 UTMI 总线都有接口。USB 控制器同时包含 AHB 主接口和 AHB 从接口。CPU 通过 AHB 从接口编写 USB 控制器寄存器。在收发传输中，USB 设备控制器需要通过 AHB 主接口来从内存读写数据。USB 设备控制器符合 USB2.0 协议规范，包含 12 个可配置的端点以及一个控制端点。这些端点可以配置成批量模式、中断模式或等时模式。USB 设备控制器内建 DMA 以降低 CPU 负担。

### 6.31.2 特性

- 符合 USB2.0 协议规范
- 支持 12 个可配置的端点以及一个控制端点
- 在收发方向，每个端点均可配置成等时传输，批量传输或中断传输
- 输入端点有 3 种工作模式——自动确认模式，手动确认模式和自由模式。
- 支持 DMA 操作
- 4092 个字节的可配置 RAM 作为端点缓存
- 端点最大支持 1024 个字节大小的数据包

### 6.31.3 框图

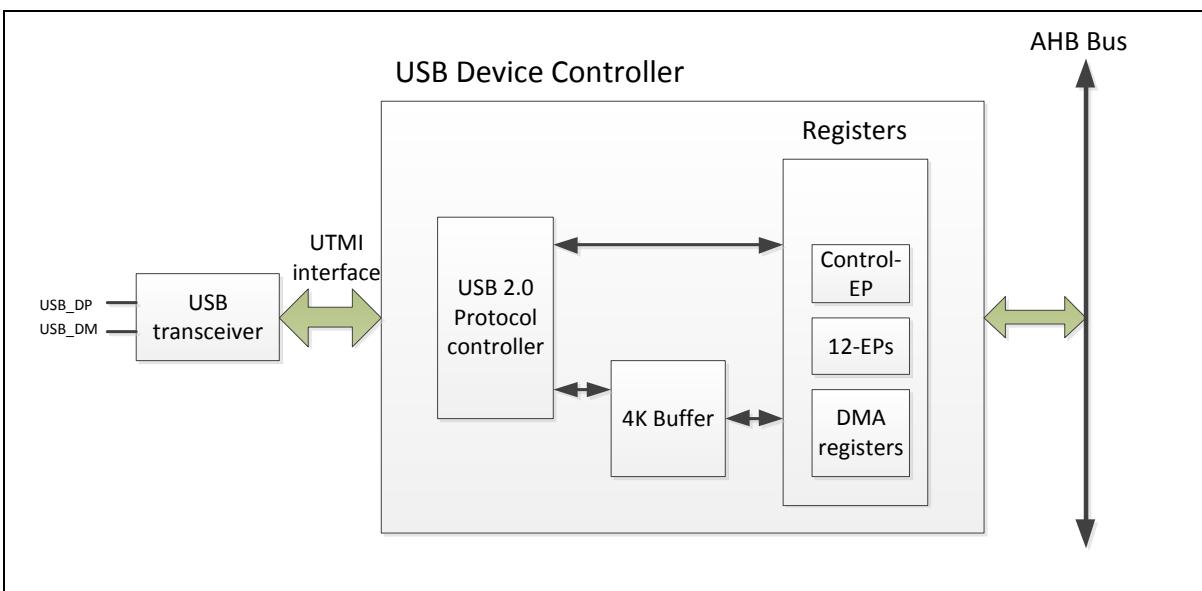


图 6.31-1 USB 设备控制器框图

### 6.31.4 基本配置

- 时钟源控制
  - 高速 USB 设备控制器时钟使能位是：HSUSBDCKEN (CLK\_AHBCLOCK[10])
- 复位控制
  - 高速 USB 设备控制器复位控制位是：HSUSBDRST (SYS\_IPRST0[10])

### 6.31.5 功能描述

#### 6.31.5.1 (IN 数据) 发送模式的不同操作模式

当接收到一个in-token，数据就可以写入缓存做发送准备。CPU对发送数据的确认有3种方式：

- 自动确认模式
- 手动确认模式
- 自由模式

#### 6.31.5.2 自动确认模式

如果端点配置为自动确认模式，端点只会在数据个数等于寄存器EPMPS配置的个数时响应in-token。如果CPU需要在最后一笔传输中发送一个短包，需要对位SHORTTXEN (HSUSBD\_EPxRSPCTL[6]) 置1，缓存里剩下的数据就会在收到下一个in-token后发送给主机。

这种模式CPU的介入较少，主要由USB设备控制器完成。这种模式一般在发送给主机的数据量大于最大包时使用。

SHORTTXEN	缓存里的有效数据	数据发送/NAK发送
0	< 最大包大小	NAK发送
0	>=最大包大小	发送最大包数据
1	< 最大包大小	发送所有的有效数据
1	>=最大包大小	发送最大包数据

#### 6.31.5.3 手动确认模式

如果端点配置为手动确认模式，每次都在CPU确认缓存里的数据个数——向寄存器EPxTxCNT写入数据个数——后才会响应in-token。向寄存器EPxTxCNT写入数据个数后，下次收到 in-token 就会发送数据给主机。

这种模式每次传输都需要 CPU 的参与。对每次传输的数据个数都要由CPU确认的应用场合很有用。

EPxTxCNT 写入	缓存里数据有效个数	数据发送/NAK发送
没有	-	NAK
写入	EPxTxCNT	发送 EPxTxCNT长度的数据

#### 6.31.5.4 自由模式

自由模式是最简单的操作模式，没有确认的过程。CPU向缓存写入数据后，如果接收到主机发送的in-token，缓存里的数据会自动确认然后发送给主机。如果缓存里的数据个数多于最大包时，控制器自动发送最大包的数据给主机。

这种模式需要 CPU最少的参与，适合传输速率比包大小重要的等时传输模式的发送。.

缓存里的有效数据	数据发送
< 最大包大小	发送所有的有效数据
>= 最大包大小	最大包发送

#### 6.31.5.5 分散-收集功能

SGEN (HSUSBD\_DMACTL[6]) =1使能 DMA 分散收集功能，然后设置 HSUSBD\_DMACNT 为8个字节，内存地址和长度会在描述符里重排序，格式如表 6.31-1所示。

	格式		
	[31]	[30]	[29:0]
字0	内存地址[31:0]		
字1	EOT	RD	保留      计数 [19:0]

表 6.31-2 分散-收集描述符格式.

**EOT:** 传输结束.当此位为高，意味着是最后一个描述符。

**RD:** “1” 意味着从内存读取数据到缓存. “0” 意味着从缓存读取数据到内存

## 6.31.6 寄存器映射

R: 只读, W: 只写, R/W: 可读可写

寄存器	偏移量	R/W	描述	复位值
<b>HSUSBD 基地址:</b>				
<b>HSUSBD_BA = 0x4001_9000</b>				
HSUSBD_GINTSTS	HSUSBD_BA+0x000	R	总体中断状态寄存器	0x0000_0000
HSUSBD_GINTEN	HSUSBD_BA+0x008	R/W	总体中断使能寄存器	0x0000_0001
HSUSBD_BUSINTSTS	HSUSBD_BA+0x010	R/W	USB总线中断状态寄存器	0x0000_0000
HSUSBD_BUSINTEN	HSUSBD_BA+0x014	R/W	USB 总线中断使能寄存器	0x0000_0040
HSUSBD_OPER	HSUSBD_BA+0x018	R/W	USB操作寄存器	0x0000_0002
HSUSBD_FRAMECNT	HSUSBD_BA+0x01C	R	USB 帧计数寄存器	0x0000_0000
HSUSBD_FADDR	HSUSBD_BA+0x020	R/W	USB 设备地址寄存器	0x0000_0000
HSUSBD_TEST	HSUSBD_BA+0x024	R/W	USB 测试模式寄存器	0x0000_0000
HSUSBD_CEPDAT	HSUSBD_BA+0x028	R/W	控制端点数据缓存	0x0000_0000
HSUSBD_CEPCTL	HSUSBD_BA+0x02C	R/W	控制端点控制寄存器	0x0000_0000
HSUSBD_CEPINTEN	HSUSBD_BA+0x030	R/W	控制端点中断使能	0x0000_0000
HSUSBD_CEPINTSTS	HSUSBD_BA+0x034	R/W	控制端点中断状态	0x0000_1800
HSUSBD_CEPTXCNT	HSUSBD_BA+0x038	R/W	控制端点输入传输数据个数	0x0000_0000
HSUSBD_CEPRXCNT	HSUSBD_BA+0x03C	R	控制端点输出传输数据个数	0x0000_0000
HSUSBD_CEPDATCNT	HSUSBD_BA+0x040	R	控制端点数据个数	0x0000_0000
HSUSBD_SETUP1_0	HSUSBD_BA+0x044	R	Setup1 & Setup0字节	0x0000_0000
HSUSBD_SETUP3_2	HSUSBD_BA+0x048	R	Setup3 & Setup2 字节	0x0000_0000
HSUSBD_SETUP5_4	HSUSBD_BA+0x04C	R	Setup5 & Setup4 字节	0x0000_0000
HSUSBD_SETUP7_6	HSUSBD_BA+0x050	R	Setup7 & Setup6 字节	0x0000_0000
HSUSBD_CEPBUFSTART	HSUSBD_BA+0x054	R/W	控制端点内存起始地址寄存器	0x0000_0000
HSUSBD_CEPBUFEND	HSUSBD_BA+0x058	R/W	控制端点内存结束地址寄存器	0x0000_0000
HSUSBD_DMACTL	HSUSBD_BA+0x05C	R/W	DMA控制状态寄存器	0x0000_0000
HSUSBD_DMACNT	HSUSBD_BA+0x060	R/W	DMA 计数寄存器	0x0000_0000
HSUSBD_EPADAT	HSUSBD_BA+0x064	R/W	端点 A 数据寄存器	0x0000_0000
HSUSBD_EPAINTSTS	HSUSBD_BA+0x068	R/W	端点 A 中断状态寄存器	0x0000_0003
HSUSBD_EPAINTEN	HSUSBD_BA+0x06C	R/W	端点 A 中断使能寄存器	0x0000_0000
HSUSBD_EPADATCNT	HSUSBD_BA+0x070	R	端点A 数据有效个数寄存器	0x0000_0000
HSUSBD_EPARSPCTL	HSUSBD_BA+0x074	R/W	端点 A 响应控制寄存器	0x0000_0000
HSUSBD_EPAMPS	HSUSBD_BA+0x078	R/W	端点A 最大包大小寄存器	0x0000_0000

<b>HSUSBD_EPATXCNT</b>	HSUSBD_BA+0x07C	R/W	端点 A 发送字节个数寄存器	0x0000_0000
<b>HSUSBD_EPACFG</b>	HSUSBD_BA+0x080	R/W	端点 A 配置寄存器	0x0000_0012
<b>HSUSBD_EPABUFSTART</b>	HSUSBD_BA+0x084	R/W	端点 A 内存起始地址寄存器	0x0000_0000
<b>HSUSBD_EPABUFEND</b>	HSUSBD_BA+0x088	R/W	端点 A 内存结束地址寄存器	0x0000_0000
<b>HSUSBD_EPBDAT</b>	HSUSBD_BA+0x08C	R/W	端点 B 数据寄存器	0x0000_0000
<b>HSUSBD_EPBINTSTS</b>	HSUSBD_BA+0x090	R/W	端点 B 中断状态寄存器	0x0000_0003
<b>HSUSBD_EPBINTEN</b>	HSUSBD_BA+0x094	R/W	端点 B 中断使能寄存器	0x0000_0000
<b>HSUSBD_EPBDAWCNT</b>	HSUSBD_BA+0x098	R	端点 B 数据有效个数寄存器	0x0000_0000
<b>HSUSBD_EPBRESPCTL</b>	HSUSBD_BA+0x09C	R/W	端点 B 响应控制寄存器	0x0000_0000
<b>HSUSBD_EPBMPMS</b>	HSUSBD_BA+0x0A0	R/W	端点 B 最大包大小寄存器	0x0000_0000
<b>HSUSBD_EPBTCNT</b>	HSUSBD_BA+0x0A4	R/W	端点 B 传输字节数寄存器	0x0000_0000
<b>HSUSBD_EPBCFG</b>	HSUSBD_BA+0x0A8	R/W	端点 B 配置寄存器	0x0000_0022
<b>HSUSBD_EPBBUFSTART</b>	HSUSBD_BA+0x0AC	R/W	端点 B 内存起始地址寄存器	0x0000_0000
<b>HSUSBD_EPBBUFEND</b>	HSUSBD_BA+0x0B0	R/W	端点 B 内存结束地址寄存器	0x0000_0000
<b>HSUSBD_EPCDAT</b>	HSUSBD_BA+0x0B4	R/W	端点 C 数据寄存器	0x0000_0000
<b>HSUSBD_EPCINTSTS</b>	HSUSBD_BA+0x0B8	R/W	端点 C 中断状态寄存器	0x0000_0003
<b>HSUSBD_EPCINTEN</b>	HSUSBD_BA+0x0BC	R/W	端点 C 中断使能寄存器	0x0000_0000
<b>HSUSBD_EPCDATCNT</b>	HSUSBD_BA+0x0C0	R	端点 C 数据有效字节数寄存器	0x0000_0000
<b>HSUSBD_EPCRSPCTL</b>	HSUSBD_BA+0x0C4	R/W	端点 C 响应控制寄存器	0x0000_0000
<b>HSUSBD_EPCMPS</b>	HSUSBD_BA+0x0C8	R/W	端点 C 最大包大小寄存器	0x0000_0000
<b>HSUSBD_EPCTXCNT</b>	HSUSBD_BA+0x0CC	R/W	端点 C 传输字节数寄存器	0x0000_0000
<b>HSUSBD_EPCCFG</b>	HSUSBD_BA+0x0D0	R/W	端点 C 配置寄存器	0x0000_0032
<b>HSUSBD_EPCBUFSTART</b>	HSUSBD_BA+0x0D4	R/W	端点 C 内存起始地址寄存器	0x0000_0000
<b>HSUSBD_EPCBUFEND</b>	HSUSBD_BA+0x0D8	R/W	端点 C 内存结束地址寄存器	0x0000_0000
<b>HSUSBD_EPDDAT</b>	HSUSBD_BA+0x0DC	R/W	端点 D 数据寄存器	0x0000_0000
<b>HSUSBD_EPDINTSTS</b>	HSUSBD_BA+0x0E0	R/W	端点 D 中断状态寄存器	0x0000_0003
<b>HSUSBD_EPDINTEN</b>	HSUSBD_BA+0x0E4	R/W	端点 D 中断使能寄存器	0x0000_0000
<b>HSUSBD_EPDATCNT</b>	HSUSBD_BA+0x0E8	R	端点 D 数据有效字节数寄存器	0x0000_0000
<b>HSUSBD_EPDRESPCTL</b>	HSUSBD_BA+0x0EC	R/W	端点 D 响应控制寄存器	0x0000_0000
<b>HSUSBD_EPDMPMS</b>	HSUSBD_BA+0x0F0	R/W	端点 D 最大包大小寄存器	0x0000_0000
<b>HSUSBD_EPDTCNT</b>	HSUSBD_BA+0x0F4	R/W	端点 D 传输计数器寄存器	0x0000_0000
<b>HSUSBD_EPDCFG</b>	HSUSBD_BA+0x0F8	R/W	端点 D 配置寄存器	0x0000_0042
<b>HSUSBD_EPDBUFSTART</b>	HSUSBD_BA+0x0FC	R/W	端点 D 内存起始地址寄存器	0x0000_0000

<b>HSUSBD_EPDBUFEND</b>	HSUSBD_BA+0x100	R/W	端点 D 内存结束地址寄存器	0x0000_0000
<b>HSUSBD_EPEDAT</b>	HSUSBD_BA+0x104	R/W	端点 D 数据寄存器	0x0000_0000
<b>HSUSBD_EPEINTSTS</b>	HSUSBD_BA+0x108	R/W	端点 E 数据寄存器	0x0000_0003
<b>HSUSBD_EPEINTEN</b>	HSUSBD_BA+0x10C	R/W	端点 E 中断状态寄存器	0x0000_0000
<b>HSUSBD_EPEDATCNT</b>	HSUSBD_BA+0x110	R	端点 E 中断使能寄存器	0x0000_0000
<b>HSUSBD_EPERSPCTL</b>	HSUSBD_BA+0x114	R/W	端点 E 数据有效字节数寄存器	0x0000_0000
<b>HSUSBD_EPEMPS</b>	HSUSBD_BA+0x118	R/W	端点 E 响应控制寄存器	0x0000_0000
<b>HSUSBD_EPETXCNT</b>	HSUSBD_BA+0x11C	R/W	端点 E 最大包大小寄存器	0x0000_0000
<b>HSUSBD_EPECFG</b>	HSUSBD_BA+0x120	R/W	端点 E 传输计数器寄存器	0x0000_0052
<b>HSUSBD_EPEBUFSTART</b>	HSUSBD_BA+0x124	R/W	端点 E 配置寄存器	0x0000_0000
<b>HSUSBD_EPEBUFEND</b>	HSUSBD_BA+0x128	R/W	端点 E 内存起始地址寄存器	0x0000_0000
<b>HSUSBD_EPFDAT</b>	HSUSBD_BA+0x12C	R/W	端点 F 数据寄存器	0x0000_0000
<b>HSUSBD_EPFINTSTS</b>	HSUSBD_BA+0x130	R/W	端点 F 中断状态寄存器	0x0000_0003
<b>HSUSBD_EPFINTEN</b>	HSUSBD_BA+0x134	R/W	端点 F 中断使能寄存器	0x0000_0000
<b>HSUSBD_EPFDATCNT</b>	HSUSBD_BA+0x138	R	端点 F 数据有效字节数寄存器	0x0000_0000
<b>HSUSBD_EPFRSPCTL</b>	HSUSBD_BA+0x13C	R/W	端点 F 响应控制寄存器	0x0000_0000
<b>HSUSBD_EPFMPS</b>	HSUSBD_BA+0x140	R/W	端点 F 最大包大小寄存器	0x0000_0000
<b>HSUSBD_EPFTXCNT</b>	HSUSBD_BA+0x144	R/W	端点 F 传输计数器寄存器	0x0000_0000
<b>HSUSBD_EPFCFG</b>	HSUSBD_BA+0x148	R/W	端点 F 配置寄存器	0x0000_0062
<b>HSUSBD_EPFBUFSTART</b>	HSUSBD_BA+0x14C	R/W	端点 F 内存起始地址寄存器	0x0000_0000
<b>HSUSBD_EPFBUFEND</b>	HSUSBD_BA+0x150	R/W	端点 F 内存结束地址寄存器	0x0000_0000
<b>HSUSBD_EPGDAT</b>	HSUSBD_BA+0x154	R/W	端点 G 数据寄存器	0x0000_0000
<b>HSUSBD_EPGINTSTS</b>	HSUSBD_BA+0x158	R/W	端点 G 中断状态寄存器	0x0000_0003
<b>HSUSBD_EPGINTEN</b>	HSUSBD_BA+0x15C	R/W	端点 G 中断使能寄存器	0x0000_0000
<b>HSUSBD_EPGDATCNT</b>	HSUSBD_BA+0x160	R	端点 G 数据有效字节数寄存器	0x0000_0000
<b>HSUSBD_EPGRSPCTL</b>	HSUSBD_BA+0x164	R/W	端点 G 响应控制寄存器	0x0000_0000
<b>HSUSBD_EPGMPS</b>	HSUSBD_BA+0x168	R/W	端点 G 最大包大小寄存器	0x0000_0000
<b>HSUSBD_EPGTXCNT</b>	HSUSBD_BA+0x16C	R/W	端点 G 传输计数器寄存器	0x0000_0000
<b>HSUSBD_EPGCFG</b>	HSUSBD_BA+0x170	R/W	端点 G 配置寄存器	0x0000_0072
<b>HSUSBD_EPGBUFSTART</b>	HSUSBD_BA+0x174	R/W	端点 G 内存起始地址寄存器	0x0000_0000
<b>HSUSBD_EPGBUFEND</b>	HSUSBD_BA+0x178	R/W	端点 G 内存结束地址寄存器	0x0000_0000
<b>HSUSBD_EPHDAT</b>	HSUSBD_BA+0x17C	R/W	端点 H 数据寄存器	0x0000_0000
<b>HSUSBD_EPHINTSTS</b>	HSUSBD_BA+0x180	R/W	端点 H 中断状态寄存器	0x0000_0003

<b>HSUSBD_EPHINTEN</b>	HSUSBD_BA+0x184	R/W	端点 H 中断使能寄存器	0x0000_0000
<b>HSUSBD_EPHDATCNT</b>	HSUSBD_BA+0x188	R	端点 H 数据有效字节数寄存器	0x0000_0000
<b>HSUSBD_EPHRSPCTL</b>	HSUSBD_BA+0x18C	R/W	端点 H 响应控制寄存器	0x0000_0000
<b>HSUSBD_EPHMPS</b>	HSUSBD_BA+0x190	R/W	端点 H 最大包大小寄存器	0x0000_0000
<b>HSUSBD_EPHTXCNT</b>	HSUSBD_BA+0x194	R/W	端点 H 传输计数器寄存器	0x0000_0000
<b>HSUSBD_EPHCFG</b>	HSUSBD_BA+0x198	R/W	端点 H 配置寄存器	0x0000_0082
<b>HSUSBD_EPHBUFSTART</b>	HSUSBD_BA+0x19C	R/W	端点 H 内存起始地址寄存器	0x0000_0000
<b>HSUSBD_EPHBUFEND</b>	HSUSBD_BA+0x1A0	R/W	端点 H 内存结束地址寄存器	0x0000_0000
<b>HSUSBD_EPIDAT</b>	HSUSBD_BA+0x1A4	R/W	端点 I 数据寄存器	0x0000_0000
<b>HSUSBD_EPIINTSTS</b>	HSUSBD_BA+0x1A8	R/W	端点 I 中断状态寄存器	0x0000_0003
<b>HSUSBD_EPIINTEN</b>	HSUSBD_BA+0x1AC	R/W	端点 I 中断使能寄存器	0x0000_0000
<b>HSUSBD_EPIDATCNT</b>	HSUSBD_BA+0x1B0	R	端点 I 数据有效字节数寄存器	0x0000_0000
<b>HSUSBD_EPIRSPCTL</b>	HSUSBD_BA+0x1B4	R/W	端点 I 响应控制寄存器	0x0000_0000
<b>HSUSBD_EPIMPS</b>	HSUSBD_BA+0x1B8	R/W	端点 I 最大包大小寄存器	0x0000_0000
<b>HSUSBD_EPITXCNT</b>	HSUSBD_BA+0x1BC	R/W	端点 I 传输计数器寄存器	0x0000_0000
<b>HSUSBD_EPICFG</b>	HSUSBD_BA+0x1C0	R/W	端点 I 配置寄存器	0x0000_0092
<b>HSUSBD_EPIBUFSTART</b>	HSUSBD_BA+0x1C4	R/W	端点 I 内存起始地址寄存器	0x0000_0000
<b>HSUSBD_EPIBUFEND</b>	HSUSBD_BA+0x1C8	R/W	端点 I 内存结束地址寄存器	0x0000_0000
<b>HSUSBD_EPJDAT</b>	HSUSBD_BA+0x1CC	R/W	端点 J 数据寄存器	0x0000_0000
<b>HSUSBD_EPJINTSTS</b>	HSUSBD_BA+0x1D0	R/W	端点 J 中断状态寄存器	0x0000_0003
<b>HSUSBD_EPJINTEN</b>	HSUSBD_BA+0x1D4	R/W	端点 J 中断使能寄存器	0x0000_0000
<b>HSUSBD_EPJDATCNT</b>	HSUSBD_BA+0x1D8	R	端点 J 数据有效字节数寄存器	0x0000_0000
<b>HSUSBD_EPJRSPECTL</b>	HSUSBD_BA+0x1DC	R/W	端点 J 响应控制寄存器	0x0000_0000
<b>HSUSBD_EPJMPMS</b>	HSUSBD_BA+0x1E0	R/W	端点 J 最大包大小寄存器	0x0000_0000
<b>HSUSBD_EPJTXCNT</b>	HSUSBD_BA+0x1E4	R/W	端点 J 传输计数器寄存器	0x0000_0000
<b>HSUSBD_EPJCFG</b>	HSUSBD_BA+0x1E8	R/W	端点 J 配置寄存器	0x0000_00A2
<b>HSUSBD_EPJBUFFSTART</b>	HSUSBD_BA+0x1EC	R/W	端点 J 内存起始地址寄存器	0x0000_0000
<b>HSUSBD_EPJBUFFEND</b>	HSUSBD_BA+0x1F0	R/W	端点 J 内存结束地址寄存器	0x0000_0000
<b>HSUSBD_EPKDAT</b>	HSUSBD_BA+0x1F4	R/W	端点 K 数据寄存器	0x0000_0000
<b>HSUSBD_EPKINTSTS</b>	HSUSBD_BA+0x1F8	R/W	端点 K 中断状态寄存器	0x0000_0003
<b>HSUSBD_EPKINTEN</b>	HSUSBD_BA+0x1FC	R/W	端点 K 中断使能寄存器	0x0000_0000
<b>HSUSBD_EPKDATCNT</b>	HSUSBD_BA+0x200	R	端点 K 数据有效字节数寄存器	0x0000_0000
<b>HSUSBD_EPKRSPCTL</b>	HSUSBD_BA+0x204	R/W	端点 K 响应控制寄存器	0x0000_0000

<b>HSUSBD_EPKMPS</b>	HSUSBD_BA+0x208	R/W	端点 K 最大包大小寄存器	0x0000_0000
<b>HSUSBD_EPKTXCNT</b>	HSUSBD_BA+0x20C	R/W	端点 K 传输计数器寄存器	0x0000_0000
<b>HSUSBD_EPKCFG</b>	HSUSBD_BA+0x210	R/W	端点 K 配置寄存器	0x0000_00B2
<b>HSUSBD_EPKBUFSTART</b>	HSUSBD_BA+0x214	R/W	端点 K 内存起始地址寄存器	0x0000_0000
<b>HSUSBD_EPKBUFEND</b>	HSUSBD_BA+0x218	R/W	端点 K 内存结束地址寄存器	0x0000_0000
<b>HSUSBD_EPLDAT</b>	HSUSBD_BA+0x21C	R/W	端点 L 数据寄存器	0x0000_0000
<b>HSUSBD_EPLINTSTS</b>	HSUSBD_BA+0x220	R/W	端点 L 中断状态寄存器	0x0000_0003
<b>HSUSBD_EPLINTEN</b>	HSUSBD_BA+0x224	R/W	端点 L 中断使能寄存器	0x0000_0000
<b>HSUSBD_EPLDATCNT</b>	HSUSBD_BA+0x228	R	端点 L 数据有效字节数寄存器	0x0000_0000
<b>HSUSBD_EPLRSPCTL</b>	HSUSBD_BA+0x22C	R/W	端点 L 响应控制寄存器	0x0000_0000
<b>HSUSBD_EPLMPS</b>	HSUSBD_BA+0x230	R/W	端点 L 最大包大小寄存器	0x0000_0000
<b>HSUSBD_EPLTXCNT</b>	HSUSBD_BA+0x234	R/W	端点 L 传输计数器寄存器	0x0000_0000
<b>HSUSBD_EPLCFG</b>	HSUSBD_BA+0x238	R/W	端点 L 配置寄存器	0x0000_00C2
<b>HSUSBD_EPLBUFSTART</b>	HSUSBD_BA+0x23C	R/W	端点 L 内存起始地址寄存器	0x0000_0000
<b>HSUSBD_EPLBUFEND</b>	HSUSBD_BA+0x240	R/W	端点 L 内存结束地址寄存器	0x0000_0000
<b>HSUSBD_DMAADDR</b>	HSUSBD_BA+0x700	R/W	AHB DMA 地址寄存器	0x0000_0000
<b>HSUSBD_PHYCTL</b>	HSUSBD_BA+0x704	R/W	USB PHY 控制寄存器	0x0000_0420

### 6.31.7 寄存器描述

#### HSUSBD\_GINTSTS 总体中断状态寄存器

寄存器	偏移量	R/W	描述	复位值
HSUSBD_GINTSTS	HSUSBD_BA+0x000	R	总体中断状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved		EPLIF	EPKIF	EPJIF	EPIIF	EPHIF	EPGIF
7	6	5	4	3	2	1	0
EPFIF	EPEIF	EPDIF	EPCIF	EPBIF	EPAIF	CEPIF	USBIF

位	描述	
[31:14]	Reserved	保留.
[13]	EPLIF	<p><b>端点 L 中断</b>            此位置1后, 需读端点 L 的中断状态寄存器以了解产生中断的原因            0 = 没发生中断.            1 = 发生了中断.</p>
[12]	EPKIF	<p><b>端点 K 中断</b>            当此位为1, 需要去读相应的端点 K 的中断状态寄存器以了解产生中断的原因            0 = 没发生中断.            1 = 发生了中断.</p>
[11]	EPJIF	<p><b>端点 J 中断</b>            当此位为1, 需要去读相应的端点 J 的中断状态寄存器以了解产生中断的原因            0 = 没发生中断.            1 = 发生了中断.</p>
[10]	EPIIF	<p><b>端点 I 中断</b>            当此位为1, 需要去读相应的端点 I 的中断状态寄存器以了解产生中断的原因            0 = 没发生中断.            1 = 发生了中断.</p>
[9]	EPHIF	<p><b>端点 H 中断</b>            当此位为1, 需要去读相应的端点 H 的中断状态寄存器以了解产生中断的原因            0 = 没发生中断.            1 = 发生了中断.</p>

[8]	<b>EPIF</b>	<b>端点 G 中断</b> 当此位为1，需要去读相应的端点 G 的中断状态寄存器以了解产生中断的原因 0 = 没发生中断. 1 = 发生了中断.
[7]	<b>EPFI</b>	<b>端点 F 中断</b> 当此位为1，需要去读相应的端点 F 的中断状态寄存器以了解产生中断的原因 0 = 没发生中断. 1 = 发生了中断.
[6]	<b>EPEIF</b>	<b>端点 E 中断</b> 当此位为1，需要去读相应的端点 E 的中断状态寄存器以了解产生中断的原因 0 = 没发生中断. 1 = 发生了中断.
[5]	<b>EPDIF</b>	<b>端点 D 中断</b> 当此位为1，需要去读相应的端点 D 的中断状态寄存器以了解产生中断的原因 0 = 没发生中断. 1 = 发生了中断.
[4]	<b>EPCIF</b>	<b>端点 C 中断</b> 当此位为1，需要去读相应的端点 C 的中断状态寄存器以了解产生中断的原因 0 = 没发生中断. 1 = 发生了中断.
[3]	<b>EPBIF</b>	<b>端点 B 中断</b> 当此位为1，需要去读相应的端点 B 的中断状态寄存器以了解产生中断的原因 0 = 没发生中断. 1 = 发生了中断.
[2]	<b>EPAIF</b>	<b>端点 A 中断</b> 当此位为1，需要去读相应的端点 A 的中断状态寄存器以了解产生中断的原因 0 = 没发生中断. 1 = 发生了中断.
[1]	<b>CEPIF</b>	<b>控制端点中断</b> 当此位为1，需要去读相应的控制端点的中断状态寄存器以了解产生中断的原因 0 = 没发生中断. 1 = 发生了中断.
[0]	<b>USBIF</b>	<b>USB 中断</b> 此位是USB特殊事件端点的中断表达，当此位置1，需要去读USB中断状态寄存器来了解产生中断的原因。 0 = 没发生中断. 1 = 发生了中断.

HSUSBD\_GINTEN 总体中断使能寄存器

寄存器	偏移量	R/W	描述	复位值
HSUSBD_GINTEN	HSUSBD_BA+0x008	R/W	总体中断使能寄存器	0x0000_0001

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved		EPLIEN	EPKIEN	EPJIEN	EPIIEN	EPHIEN	EPGIEN
7	6	5	4	3	2	1	0
EPFIEN	EPEIEN	EPDIEN	EPCIEN	EPBIEN	EPAIEN	CEPIEN	USBIEN

位	描述	
[31:14]	Reserved	保留.
[13]	EPLIEN	端点 L 中断使能位 0 = 禁止端点 L 中断. 1 = 使能端点 L 中断.
[12]	EPKIEN	端点 K 中断使能位 0 = 禁止端点 K 中断. 1 = 使能端点 K 中断.
[11]	EPJIEN	端点 J 中断使能位 0 = 禁止端点 J 中断. 1 = 使能端点 J 中断.
[10]	EPIIEN	端点 I 中断使能位 0 = 禁止端点 I 中断. 1 = 使能端点 I 中断.
[9]	EPHIEN	端点 H 中断使能位 0 = 禁止端点 H 中断. 1 = 使能端点 H 中断.
[8]	EPGIEN	端点 G 中断使能位 0 = 禁止端点 G 中断. 1 = 使能端点 G 中断.
[7]	EPFIEN	端点 F 中断使能位 0 = 禁止端点 F 中断. 1 = 使能端点 F 中断.

[6]	<b>EPEIEN</b>	端点 E 中断使能位 0 = 禁止端点 E 中断. 1 = 使能端点 E 中断.
[5]	<b>EPDIEN</b>	端点 D 中断使能位 0 = 禁止端点 D 中断. 1 = 使能端点 D 中断.
[4]	<b>EPCIEN</b>	端点 C 中断使能位 0 = 禁止端点 C 中断. 1 = 使能端点 C 中断.
[3]	<b>EPBIEN</b>	端点 B 中断使能位 0 = 禁止端点 B 中断. 1 = 使能端点 B 中断.
[2]	<b>EPAIEN</b>	端点 A 中断使能位 0 = 禁止端点 A 中断. 1 = 使能端点 A 中断.
[1]	<b>CEPIEN</b>	控制端点中断使能位 0 = 禁止控制端点中断. 1 = 使能控制端点中断.
[0]	<b>USBIEN</b>	USB 中断使能位 0 = 禁止中断. 1 = 使能中断.

**HSUSBD\_BUSINTSTS USB 总线中断状态寄存器**

寄存器	偏移量	R/W	描述	复位值
HSUSBD_BUSINTSTS	HSUSBD_BA+0x010	R/W	USB 总线中断状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							<b>VBUSDETIF</b>
7	6	5	4	3	2	1	0
Reserved	PHYCLKVLDIF	DMADONEIF	HISPDIF	SUSPENDIF	RESUMEIF	RSTIF	SOFIF

位	描述	
[31:9]	<b>Reserved</b>	保留.
[8]	<b>VBUSDETIF</b>	<b>VBUS 检测中断状态</b> 0 = 没有 VBUS 插入. 1 = VBUS已插入. <b>注意:</b> 此位写1清0
[7]	<b>Reserved</b>	保留.
[6]	<b>PHYCLKVLDIF</b>	<b>可用时钟中断</b> 0 = 可用时钟未就绪 1 = 传输器提供的可用时钟有效. <b>注意:</b> 此位写1清0.
[5]	<b>DMADONEIF</b>	<b>DMA 完成中断</b> 0 = 没有DMA传输完成. 1 = DMA传输完成. <b>注意:</b> 此位写1清0.
[4]	<b>HISPDIF</b>	<b>高速设置</b> 0 = 没有检测到有效的高速复位协议. 1 = 检测到有效的高速复位协议，并且设备被设置为高速. <b>注意:</b> 此位写1清0.
[3]	<b>SUSPENDIF</b>	<b>挂起请求</b> 此位默认是1，要在USB复位前写1清0，此位在检测到主机发送的 USB 挂起请求后置1。 0 = 没有检测到主机发送的USB挂起请求。 1=检测到主机发送的USB挂起请求。 <b>注意:</b> 此位写1清0.

[2]	<b>RESUMEIF</b>	<b>恢复状态</b> 0 = 没发生总线恢复 1 = 发生总线恢复. <b>注意:</b> 此位写1清0..
[1]	<b>RSTIF</b>	<b>复位状态</b> 当此位置1, 指示 USB 根端口的复位结束了. 0 = USB 根端口复位没有结束. 1 = USB根端口复位结束. <b>注意:</b> 此位写1清0..
[0]	<b>SOFIF</b>	<b>SOF接受控制</b> 0 = 没有接收到帧开始包 1 = 接收到帧开始包 <b>注意:</b> 此位写1清0

HSUSBD\_BUSINTEN USB 总线中断使能寄存器

寄存器	偏移量	R/W	描述	复位值
HSUSBD_BUSINTEN	HSUSBD_BA+0x014	R/W	USB总线中断使能寄存器	0x0000_0040

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							VBUSDETIEN
7	6	5	4	3	2	1	0
Reserved	PHYCLKVLDIE_N	DMADONEIEN	HISPDIE_N	SUSPENDIE_N	RESUMEIEN	RSTIEN	SOFIEN

位	描述
[31:9]	Reserved 保留.
[8]	<b>VBUSDETIEN</b> VBUS 检测中断使能位 0 = 禁止 VBUS 浮空检测中断. 1 = 使能 VBUS 浮空检测中断.
[7]	Reserved 保留.
[6]	<b>PHYCLKVLDIEN</b> 可用时钟中断使能位 0 = 禁止 可用时钟中断. 1 = 使能 可用时钟中断.
[5]	<b>DMADONEIEN</b> DMA 完成中断 使能 0 = 禁止 DMA 完成中断 1 = 使能 DMA 完成中断.
[4]	<b>HISPDIE_N</b> 高速设置中断使能 0 = 禁止 高速设置中断. 1 = 使能 高速设置中断.
[3]	<b>SUSPENDIE_N</b> 挂起请求中断使能 0 = 禁止 挂起中断. 1 = 使能 挂起中断
[2]	<b>RESUMEIEN</b> 恢复中断使能 0 = 禁止 恢复中断 1 = 使能 恢复中断.
[1]	<b>RSTIEN</b> 复位状态 0 = 禁止 USB 复位中断 1 = 使能 USB 复位中断.

[0]	<b>SOFIEN</b>	<b>SOF 中斷使能</b> 0 = 禁止 SOF 中斷. 1 = 使能 SOF 中斷.
-----	---------------	---

**HSUSBD\_OPER USB操作寄存器**

寄存器	偏移量	R/W	描述	复位值
HSUSBD_OPER	HSUSBD_BA+0x018	R/W	USB 操作寄存器	0x0000_0002

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved					CURSPD	HISPDEN	RESUMEEN

位	描述	
[31:3]	<b>Reserved</b>	保留.
[2]	<b>CURSPD</b>	<b>USB 目前速度</b> 0 = 设备工作在全速模式. 1 = USB设备工作在高速模式
[1]	<b>HISPDEN</b>	<b>USB 高速</b> 0 = USB 设备在复位协议中不理会线性调频序列，意味着即使连上了USB2.0 的主机也只允许USB 设备控制器工作在全速模式，. 1 = USB 设备控制器在复位时，发起线性调频序列.
[0]	<b>RESUMEEN</b>	<b>产生恢复序列</b> 0 = 没有向主机发起恢复序列 1 = 当使能设备远程唤醒，向主机发起恢复序列。此位自动清0。

**HSUSBD FRAMECNT USB 帧计数寄存器**

寄存器	偏移量	R/W	描述	复位值
HSUSBD_FRAMECNT	HSUSBD_BA+0x01C	R	USB帧计数寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved		FRAMECNT					
7	6	5	4	3	2	1	0
FRAMECNT					MFRAFECNT		

位	描述	
[31:14]	Reserved	保留
[13:3]	FRAMECNT	帧计数 这里包含最近的帧开始包的计数值
[2:0]	MFRAFECNT	微帧计数 这里包含帧计数域里的微帧数量

**HSUSBD\_FADDR USB 设备地址寄存器**

寄存器	偏移量	R/W	描述	复位值
HSUSBD_FADDR	HSUSBD_BA+0x020	R/W	USB 设备地址寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved	FADDR						

位	描述	
[31:7]	Reserved	保留.
[6:0]	FADDR	<b>USB 设备地址</b> 这里包含USB设备的配置地址，在检测到根端口复位后清0.

**HSUSBD TEST USB 测试模式寄存器**

寄存器	偏移量	R/W	描述	复位值
HSUSBD_TEST	HSUSBD_BA+0x024	R/W	USB 测试模式寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved					TESTMODE		

位	描述	
[31:3]	Reserved	保留
[2:0]	TESTMODE	<p>测试模式选择            000 = 正常操作.            001 = Test_J.            010 = Test_K.            011 = Test_SE0_NAK.            100 = Test_Packet.            101 = Test_Force_Enable.            110 = 保留.            111 = 保留.</p> <p><b>注意:</b> 在检测到根端口复位后清0</p>

**HSUSBD\_CEPDAT 控制端点数据缓存**

寄存器	偏移量	R/W	描述	复位值
HSUSBD_CEPDAT	HSUSBD_BA+0x028	R/W	控制端点数据缓存	0x0000_0000

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

位	描述
[31:0]	<b>DAT</b> 控制端点数据缓存 控制端点读写的数据缓存 <b>注意:</b> 只支持字或字节访问.

**HSUSBD\_CEPCTL 控制断点控制寄存器**

寄存器	偏移量	R/W	描述	复位值
HSUSBD_CEPCTL	HSUSBD_BA+0x02C	R/W	控制端点控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved				FLUSH	ZEROLEN	STALLEN	NAKCLR

位	描述	
[31:4]	<b>Reserved</b>	保留.
[3]	<b>FLUSH</b>	<b>CEP 清除 位</b> 0 = 无效。 1 = 清除包缓存及HSUSBD_CEPDATCNT寄存器，此位自动清0。
[2]	<b>ZEROLEN</b>	<b>0长度包</b> 这一位只在自动确认模式下有效。. 0 = 无效 1 = USB 设备收到IN token 后向主机发送0长度数据包。这一位在发送0长度的包之后自动清0，所以CPU不需要清除此位。
[1]	<b>STALLEN</b>	<b>STALL使能位</b> stall 位置1通常用来响应无效的或不支持的请求。此位置1必须同时把 NAKCLR位清0，因为NAKCLR位的优先级比STALL位高。在接收到下一个setup-token后此位自动清除，所以，CPU不需要再清除此位。 0 = 控制端点收到令牌，不回复 STALL。 1 = 控制端点收到任何令牌，都回复 STALL <b>注意:</b> 只有当CPU对HSUSBD_CEPCTL [1:0] 写入 2'b10 或2'b00, 此位才会更新
[0]	<b>NAKCLR</b>	<b>无响应控制</b> 0 = 此位由CPU写0清除，表示 CPU 已对 USB Device 配置好并已做好接收数据、响应 ACK 的准备 1 = USB 设备收到Setup token后置1，USB 设备对接下来的主机请求响应 NAK。本地 CPU可以用这个时间完成基于这个请求的相关工作之后清除这一位。

HSUSBD\_CEPINTEN 控制端点中断使能

寄存器	偏移量	R/W	描述	复位值
HSUSBD_CEPINTEN	HSUSBD_BA+0x030	R/W	控制端点中断使能	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved			BUFEMPTYIEN	BUFFULLIEN	STSDONEIE N	ERRIEN	STALLIEN
7	6	5	4	3	2	1	0
NAKIEN	RXPKIEN	TXPKIEN	PINGIEN	INTKIEN	OUTTKIEN	SETUPPKIEN	SETUPTKIEN

位	描述
[31:13]	<b>Reserved</b> 保留
[12]	<b>BUFEMPTYIEN</b> 缓存空中断 0 = 禁止控制端点缓存空中断. 1 = 使能控制端点缓存空中断.
[11]	<b>BUFFULLIEN</b> 缓存满中断 0 = 禁止控制端点缓存满中断 1 = 使能控制端点缓存满中断
[10]	<b>STSDONEIEN</b> 状态完成中断 0 = 禁止控制端点状态完成中断. 1 = 使能控制端点状态完成中断.
[9]	<b>ERRIEN</b> USB 错误中断 0 = 禁止控制端点 USB 错误中断. 1 = 使能控制端点 USB 错误中断.
[8]	<b>STALLIEN</b> STALL发送中断 0 = 禁止控制端点 STALL发送中断 1 = 使能控制端点 STALL 发送中断
[7]	<b>NAKIEN</b> NAK发送中断 0 = 禁止控制端点 NAK发送中断 1 = 使能控制端点 NAK 发送中断
[6]	<b>RXPKIEN</b> 数据包接收中断 0 = 禁止控制端点数据接收中断 1 = 使能控制端点数据接收中断

[5]	<b>TXPKIEN</b>	<b>数据包发送中断</b> 0 = 禁止控制端点数据包发送中断. 1 = 使能控制端点数据包发送中断
[4]	<b>PINGIEN</b>	<b>Ping Token 中断</b> 0 = 禁止控制端点ping token中断 1 = 使能控制端点ping token 中断.
[3]	<b>INTKIEN</b>	<b>In Token 中断</b> 0 = 禁止控制端点 IN token中断. 1 = 使能控制端点 IN token中断.
[2]	<b>OUTTKIEN</b>	<b>Out Token 中断</b> 0 =禁止控制端点 OUT token中断 1 =使能控制端点 OUT token中断.
[1]	<b>SETUPPKIEN</b>	<b>Setup包中断</b> 0 = 禁止控制端点 SETUP 包中断 1 = 使能控制端点 SETUP 包中断
[0]	<b>SETUPTKIEN</b>	<b>Setup Token 中断使能位</b> 0 = 禁止控制端点 SETUP token中断. 1 = 使能控制端点 SETUP token中断.

**HSUSBD\_CEPINTSTS 控制端点中断状态**

寄存器	偏移量	R/W	描述	复位值
HSUSBD_CEPINTSTS	HSUSBD_BA+0x034	R/W	控制端点中断状态	0x0000_1800

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved			BUFEMPTYIF	BUFFULLIF	STSDONEIF	ERRIF	STALLIF
7	6	5	4	3	2	1	0
NAKIF	RXPKIF	TXPKIF	PINGIF	INTKIF	OUTTKIF	SETUPPKIF	SETUPTKIF

位	描述
[31:13]	<b>Reserved</b> 保留.
[12]	<b>BUFEMPTYIF</b> 缓存空中断 0 = 控制端点缓存非空 1 = 控制端点缓存空 注意: 此位写1清0
[11]	<b>BUFFULLIF</b> 缓存满中断 0 = 控制端点缓存未满. 1 = 控制端点缓存满. 注意: 此位写1清0
[10]	<b>STSDONEIF</b> 状态完成中断 0 = 没有USB事务成功结束 1 = USB 事务的状态阶段成功完成. 注意: 此位写1清0
[9]	<b>ERRIF</b> USB 错误中断 0 = 事务中没有错误发生. 1 = 事务中发生了一个错误. 注意: 此位写1清0
[8]	<b>STALLIF</b> STALL 发送中断 0 = 没有发送stall-token 响应收发令牌 1 = 发送stall-token 响应收发令牌. 注意: 此位写1清0

[7]	<b>NAKIF</b>	<b>NAK 发送中断</b> 0 = 没有发送 NAK-token响应令牌 1 =发送 NAK-token响应令牌 <b>注意:</b> 此位写1清0.
[6]	<b>RXPKIF</b>	<b>数据包接受中断</b> 0 = 没有发生这样的状况: 在OUT-token 时从主机成功接收数据并发送 ACK给主机 1 = 在OUT-token 时从主机成功接收数据并发送 ACK给主机 <b>注意:</b> 此位写1清0
[5]	<b>TXPKIF</b>	<b>数据包发送中断</b> 0 = 没有发生这样的状况: 在IN-token 时向主机成功发送数据并接收到ACK-token 1 =在IN-token 时向主机成功发送数据并接收到ACK-token. <b>注意:</b> 此位写1清0
[4]	<b>PINGIF</b>	<b>Ping Token 中断</b> 0 = 控制端点没有从主机接收到ping token. 1 =控制端点从主机接收到ping token. <b>注意:</b> 此位写1清0
[3]	<b>INTKIF</b>	<b>in Token 中断</b> 0 = 控制端点没有从主机接收到 IN token. 1 = 控制端点从主机接收到 IN token. <b>注意:</b> 此位写1清0
[2]	<b>OUTTKIF</b>	<b>Out Token 中断</b> 0 = 控制端点没有从主机接收到 OUT token 1 =控制端点从主机接收到 OUT token <b>注意:</b> 此位写1清0.
[1]	<b>SETUPPKIF</b>	<b>Setup Packet 中断</b> 这一位必须在收到下一个setup packet 前清除（写1）。如果这一位没有清除，那么连续的 setup packets 会覆盖setup packet 缓存 0 =没有从主机接收到 Setup packet 1 = 从主机接收到 Setup packet. <b>注意:</b> 此位写1清0.
[0]	<b>SETUPTKIF</b>	<b>Setup Token中断</b> 0 = 没有接收到Setup token. 1 = 接收到Setup token <b>注意:</b> 此位写1清0.

**HSUSBD\_CEPTXCNT 控制端点输入传输数据个数**

寄存器	偏移量	R/W	描述	复位值
HSUSBD_CEPTXCNT	HSUSBD_BA+0x038	R/W	控制端点输入传输数据个数	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
TXCNT							

位	描述	
[31:8]	<b>Reserved</b>	保留
[7:0]	<b>TXCNT</b>	<b>IN 数据计数</b> 控制端点没有操作模式选择（但是类似手动模式）。本地CPU接受到in-token 向控制传输缓存写入数据，并向这个寄存器写入发送字节数。如果这里是0，会向主机发送一个0长度的包。如果写入的个数大于最大包的长度，只会发送最大包长度的数据

**HSUSBD\_CEPRXCNT 控制端点输出传输数据个数**

寄存器	偏移量	R/W	描述	复位值
HSUSBD_CEPRXCNT	HSUSBD_BA+0x03C	R	控制端点输出传输数据个数	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
RXCNT							

位	描述	
[31:8]	Reserved	保留.
[7:0]	RXCNT	OUT 数据个数 控制传输，在OUT传输中收到的数据个数

**HSUSBD\_CEPDATCNT 控制端点数据个数**

寄存器	偏移量	R/W	描述	复位值
HSUSBD_CEPDATCNT	HSUSBD_BA+0x040	R	控制端点数据个数	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
DATCNT							
7	6	5	4	3	2	1	0
DATCNT							

位	描述	
[31:16]	Reserved	保留.
[15:0]	DATCNT	控制端点数据个数 OUT时，控制端点缓存中收到的数据个数；IN时，控制端点尚未发出去的数据个数。

**HSUSBD SETUP1\_0 Setup1 & Setup0 字节**

寄存器	偏移量	R/W	描述	复位值
HSUSBD_SETUP1_0	HSUSBD_BA+0x044	R	Setup1 & Setup0 字节	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
SETUP1							
7	6	5	4	3	2	1	0
SETUP0							

位	描述	
[31:16]	Reserved	保留.
[15:8]	SETUP1	<p><b>Setup 字节 1[15:8]</b></p> <p>这个字节的内容是最近接收到的setup packet的字节1，对于标准的设备请求，下面是bRequest的对应信息。</p> <p>00000000 = 读取状态.      00000001 = 清除特征字.      00000010 = 保留      00000011 = 设置特征字.      00000100 = 保留.      00000101 = 设置地址.      00000110 = 读取描述符.      00000111 = 设置描述符.      00001000 = 读取附加.      00001001 = 设置配置.      00001010 = 读取接口.      00001011 = 设置接口.      00001100 = 同步帧.</p>

[7:0]	SETUP0	<p><b>Setup 字节 0[7:0]</b></p> <p>这个寄存器是最近接收到的 setup packet 的字节 0. 对于标准的设备请求，下面是 bmRequestType 的对应信息。</p> <p>Bit 7(方向):</p> <ul style="list-style-type: none"><li>0: 主机到设备</li><li>1: 设备到主机</li></ul> <p>Bit 6-5 (类型):</p> <ul style="list-style-type: none"><li>00: 标准</li><li>01: 类</li><li>10: 厂家</li><li>11: 保留</li></ul> <p>Bit 4-0 (接收者)</p> <ul style="list-style-type: none"><li>00000: 设备</li><li>00001: 接口</li><li>00010: 端点</li><li>00011: 其他</li><li>Others: 保留</li></ul>
-------	--------	--

HSUSBD SETUP3\_2 Setup3 & Setup2 字节

寄存器	偏移量	R/W	描述	复位值
HSUSBD_SETUP3_2	HSUSBD_BA+0x048	R	Setup3 & Setup2 字节	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
SETUP3							
7	6	5	4	3	2	1	0
SETUP2							

位	描述	
[31:16]	<b>Reserved</b>	保留
[15:8]	<b>SETUP3</b>	<b>Setup 字节 3 [15:8]</b> 这个寄存器是最近接收到的 setup packet 的字节3. 对于标准的设备请求，这里是 wValue域的高字节。
[7:0]	<b>SETUP2</b>	<b>Setup 字节 2 [7:0]</b> 这个寄存器是最近接收到的 setup packet 的字节2. 对于标准的设备请求，这里是 wValue域的低字节。

**HSUSBD SETUP5\_4 Setup5 & Setup4 字节**

寄存器	偏移量	R/W	描述	复位值
HSUSBD_SETUP5_4	HSUSBD_BA+0x04C	R	Setup5 & Setup4 字节	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
SETUP5							
7	6	5	4	3	2	1	0
SETUP4							

位	描述	
[31:16]	Reserved	保留
[15:8]	SETUP5	<b>Setup 字节 5[15:8]</b> 这个寄存器是最近接收到的 setup packet 的字节5。对于标准的设备请求，这里是 wIndex域的高字节..
[7:0]	SETUP4	<b>Setup 字节 4[7:0]</b> 这个寄存器是最近接收到的 setup packet 的字节4。对于标准的设备请求，这里是 wIndex域的低字节..

**HSUSBD SETUP7\_6 Setup7 & Setup6 字节**

寄存器	偏移量	R/W	描述	复位值
HSUSBD_SETUP7_6	HSUSBD_BA+0x050	R	Setup7 & Setup6 字节	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
SETUP7							
7	6	5	4	3	2	1	0
SETUP6							

位	描述	
[31:16]	Reserved	保留.
[15:8]	SETUP7	<b>Setup 字节 7[15:8]</b> 这个寄存器是最近接收到的 setup packet 的字节7. 对于标准的设备请求，这里是 wLength域的高字节..
[7:0]	SETUP6	<b>Setup 字节 6[7:0]</b> 这个寄存器是最近接收到的 setup packet 的字节6. 对于标准的设备请求，这里是 wLength域的低字节..

HSUSBD\_CEPBUFSTART 控制端点内存起始地址

寄存器	偏移量	R/W	描述	复位值
HSUSBD_CEPBUFSTART	HSUSBD_BA+0x054	R/W	控制端点内存起始地址	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved				SADDR			
7	6	5	4	3	2	1	0
SADDR							

位	描述	
[31:12]	Reserved	保留.
[11:0]	SADDR	控制端点起始地址 这里是控制端点内存的起始地址.

**HSUSBD\_CEPBUFEND 控制端点内存结束地址寄存器**

寄存器	偏移量	R/W	描述	复位值
HSUSBD_CEPBUFEND	HSUSBD_BA+0x058	R/W	控制端点内存结束地址寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved				EADDR			
7	6	5	4	3	2	1	0
EADDR							

位	描述	
[31:12]	Reserved	保留.
[11:0]	EADDR	控制端点结束地址 这里是控制端点内存的结束地址.

HSUSBD\_DMACTL DMA 控制状态寄存器

寄存器	偏移量	R/W	描述	复位值
HSUSBD_DMACTL	HSUSBD_BA+0x05C	R/W	DMA 控制状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
DMARST	SGEN	DMAEN	DMARD	EPNUM			

位	描述	
[31:9]	<b>Reserved</b>	保留.
[8]	<b>SVINEP</b>	<b>端点方向</b> 这里决定 DMA 服务于输入端点或输出端点 0: DMA服务输出端点 1: DMA 服务输入端点
[7]	<b>DMARST</b>	<b>复位 DMA 状态机</b> 0 : 无效 1 : 复位 DMA 状态机.
[6]	<b>SGEN</b>	<b>分散-收集 功能使能位</b> 0 : 禁止 分散-收集 功能. 1 : 使能 分散-收集 功能
[5]	<b>DMAEN</b>	<b>DMA 使能位</b> 0 : 禁止 DMA 功能. 1 : 使能 DMA 功能
[4]	<b>DMARD</b>	<b>DMA 操作</b> 0 : DMA 写操作 (从 USB 缓存读取). DMA 会在执行写操作之前根据EPNM的设定确认端点数据的有效个数HSUSBD_EPxDATCNT 1 : DMA 读操作 (写入 USB缓存).
[3:0]	<b>EPNUM</b>	<b>DMA端点地址位</b> 设定端点地址

**HSUSBD DMACNT DMA 计数寄存器**

寄存器	偏移量	R/W	描述	复位值
HSUSBD_DMACNT	HSUSBD_BA+0x060	R/W	DMA计数寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved				DMACNT			
15	14	13	12	11	10	9	8
DMACNT							
7	6	5	4	3	2	1	0
DMACNT							

位	描述	
[31:20]	Reserved	保留
[19:0]	DMACNT	DMA 传输计数 在这里写入需要DMA 传输的数据个数.

## HSUSBD\_EPADAT~ HSUSBD\_EPLDAT 端点 A~L 数据寄存器

寄存器	偏移量	R/W	描述	复位值
HSUSBD_EPADAT	HSUSBD_BA+0x064	R/W	端点 A 数据寄存器	0x0000_0000
HSUSBD_EPBDAT	HSUSBD_BA+0x08C	R/W	端点 B 数据寄存器	0x0000_0000
HSUSBD_EPCDAT	HSUSBD_BA+0x0B4	R/W	端点 C 数据寄存器	0x0000_0000
HSUSBD_EPDDAT	HSUSBD_BA+0x0DC	R/W	端点 D 数据寄存器	0x0000_0000
HSUSBD_EPEDAT	HSUSBD_BA+0x104	R/W	端点 E 数据寄存器	0x0000_0000
HSUSBD_EPFDAT	HSUSBD_BA+0x12C	R/W	端点 F 数据寄存器	0x0000_0000
HSUSBD_EPGDAT	HSUSBD_BA+0x154	R/W	端点 G 数据寄存器	0x0000_0000
HSUSBD_EPHDAT	HSUSBD_BA+0x17C	R/W	端点 H 数据寄存器	0x0000_0000
HSUSBD_EPIDAT	HSUSBD_BA+0x1A4	R/W	端点 I 数据寄存器	0x0000_0000
HSUSBD_EPJDAT	HSUSBD_BA+0x1CC	R/W	端点 J 数据寄存器	0x0000_0000
HSUSBD_EPKDAT	HSUSBD_BA+0x1F4	R/W	端点 K 数据寄存器	0x0000_0000
HSUSBD_EPLDAT	HSUSBD_BA+0x21C	R/W	端点 L 数据寄存器	0x0000_0000

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

位	描述	
[31:0]	EPDAT	端点 A~L 数据寄存器 端点 A~L 读写数据缓存 <b>注意:</b> 只支持字节或字访问。

## HSUSBD\_EPAINTSTS~ HSUSBD\_EPLINTSTS 端点 A~L 中断状态寄存器

寄存器	偏移量	R/W	描述	复位值
HSUSBD_EPAINTSTS	HSUSBD_BA+0x068	R/W	端点 A 中断状态寄存器	0x0000_0003
HSUSBD_EPBINTSTS	HSUSBD_BA+0x090	R/W	端点 B 中断状态寄存器	0x0000_0003
HSUSBD_EPCINTSTS	HSUSBD_BA+0x0B8	R/W	端点 C 中断状态寄存器	0x0000_0003
HSUSBD_EPDINTSTS	HSUSBD_BA+0x0E0	R/W	端点 D 中断状态寄存器	0x0000_0003
HSUSBD_EPEINTSTS	HSUSBD_BA+0x108	R/W	端点 E 中断状态寄存器	0x0000_0003
HSUSBD_EPFINTSTS	HSUSBD_BA+0x130	R/W	端点 F 中断状态寄存器	0x0000_0003
HSUSBD_EPGINTSTS	HSUSBD_BA+0x158	R/W	端点 G 中断状态寄存器	0x0000_0003
HSUSBD_EPHINTSTS	HSUSBD_BA+0x180	R/W	端点 H 中断状态寄存器	0x0000_0003
HSUSBD_EPIINTSTS	HSUSBD_BA+0x1A8	R/W	端点 I 中断状态寄存器	0x0000_0003
HSUSBD_EPJINTSTS	HSUSBD_BA+0x1D0	R/W	端点 J 中断状态寄存器	0x0000_0003
HSUSBD_EPKINTSTS	HSUSBD_BA+0x1F8	R/W	端点 K 中断状态寄存器	0x0000_0003
HSUSBD_EPLINTSTS	HSUSBD_BA+0x220	R/W	端点 L 中断状态寄存器	0x0000_0003

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved			SHORTRXIF	ERRIF	NYETIF	STALLIF	NAKIF
7	6	5	4	3	2	1	0
PINGIF	INTKIF	OUTTKIF	RXPKIF	TXPKIF	SHORTTXIF	BUFEMPTYIF	BUFFULLIF

位	描述	
[31:13]	Reserved	保留.
[12]	SHORTRXIF	<p>接收到 Bulk Out 短包            0 = 没有接收到 bulk out 短包.            1 = 接收到 bulk out 短包 (包含0长度的包).            注意: 此位写1清0</p>
[11]	ERRIF	<p>错误发送            0 = 传输中没有错误            1 = 传输中发生错误.            注意: 此位写1清0</p>

[10]	<b>NYETIF</b>	<b>NYET 发送</b> 0 = 内存中有足够的空间存放下一个数据包. 1 = 内存中没有足够的空间存放下一个数据包. <b>注意:</b> 此位写1清0.
[9]	<b>STALLIF</b>	<b>USB STALL 发送</b> 0 = 上一个数据包已被接受或提供, 没有响应STALL. 1 = 上一个数据包没有被接受或提供, 响应了STALL <b>注意:</b> 此位写1清0
[8]	<b>NAKIF</b>	<b>USB NAK 发送</b> 0 = 上一个IN-token 提供了数据 (未发NAK) 1 = 上一个IN-token未提供数据, 响应了 NAK. <b>注意:</b> 此位写1清0
[7]	<b>PINGIF</b>	<b>PING Token 中断</b> 0 = 从机没有接收到 数据 PING token. 1 = 从机接收到一个数据 PING token <b>注意:</b> 此位写1清0.
[6]	<b>INTKIF</b>	<b>数据 IN Token 中断</b> 0 = 从机没收到 IN token 1 = 从机收到一个 IN token <b>注意:</b> 此位写1清0.
[5]	<b>OUTTKIF</b>	<b>数据 OUT Token 中断</b> 0 = 从机没收到 OUT token 1 = 从机收到 OUT token。这一位在高速模式下也可以被 PING token 置位 <b>注意:</b> 此位写1清0
[4]	<b>RXPKIF</b>	<b>数据包接收中断</b> 0 = 端点没有收到数据包. 1 = 端点收到一个数据包. <b>注意:</b> 此位写1清0.
[3]	<b>TXPKIF</b>	<b>数据包发送中断</b> 0 = 端点没有向主机发送数据包. 1 = 端点向主机发送一个数据包. <b>注意:</b> 此位写1清0
[2]	<b>SHORTTXIF</b>	<b>短包发送中断</b> 0 = 上一个数据包不小于最大包 (EPMPS). 1 = 上一个数据包小于最大包 (EPMPS). <b>注意:</b> 此位写1清0.

[1]	<b>BUFEMPTYIF</b>	<p><b>缓存空</b> 对于输入端点. 0 = 端点缓存非空. 1 = 端点缓存空. 对于输出端点 0 = 缓存里可读数据不是0 1 = 缓存里可读数据是0, 或本地没有可用缓存（没有可读数据） <b>注意:</b> 此位只读.</p>
[0]	<b>BUFFULLIF</b>	<p><b>缓存满</b> 对于输入缓存, 目前选择的缓存满了, 或者本地没有空间可写, 对于输出缓存, FIFO已满, 整个数据包都可以读取 0 = 端点包缓存未满. 1 = 端点包缓存已满. <b>注意:</b> 此位只读</p>

## HSUSBD\_EPAINTEN~ HSUSBD\_EPLINTEN 端点 A~L 中断使能寄存器

寄存器	偏移量	R/W	描述	复位值
HSUSBD_EPAINTEN	HSUSBD_BA+0x06C	R/W	端点 A 中断使能寄存器	0x0000_0000
HSUSBD_EPBINTE N	HSUSBD_BA+0x094	R/W	端点 B 中断使能寄存器	0x0000_0000
HSUSBD_EPCINTE N	HSUSBD_BA+0x0BC	R/W	端点 C 中断使能寄存器	0x0000_0000
HSUSBD_EPDINTE N	HSUSBD_BA+0x0E4	R/W	端点 D 中断使能寄存器	0x0000_0000
HSUSBD_EPEINTE N	HSUSBD_BA+0x10C	R/W	端点 E 中断使能寄存器	0x0000_0000
HSUSBD_EPFINTE N	HSUSBD_BA+0x134	R/W	端点 F 中断使能寄存器	0x0000_0000
HSUSBD_EPGINTE N	HSUSBD_BA+0x15C	R/W	端点 G 中断使能寄存器	0x0000_0000
HSUSBD_EPHINTE N	HSUSBD_BA+0x184	R/W	端点 H 中断使能寄存器	0x0000_0000
HSUSBD_EPIINTE N	HSUSBD_BA+0x1AC	R/W	端点 I 中断使能寄存器	0x0000_0000
HSUSBD_EPJINTE N	HSUSBD_BA+0x1D4	R/W	端点 J 中断使能寄存器	0x0000_0000
HSUSBD_EPKINTE N	HSUSBD_BA+0x1FC	R/W	端点 K 中断使能寄存器	0x0000_0000
HSUSBD_EPLINTE N	HSUSBD_BA+0x224	R/W	端点 L 中断使能寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved			SHORTRXIEN	ERRIEN	NYETIEN	STALLIEN	NAKIEN
7	6	5	4	3	2	1	0
PINGIEN	INTKIEN	OUTTKIEN	RXPKIEN	TXPKIEN	SHORTTXIEN	BUFEMPTYIEN	BUFFULLIE N

位	描述						
[31:13]	Reserved	保留.					
[12]	SHORTRXIEN	Bulk Out 短包中断使能位 0 = 禁止 Bulk out 短包中断 1 = 使能 Bulk out 短包中断.					

[11]	<b>ERRIEN</b>	<b>错误中断使能位</b> 0 = 禁止错误事件中断. 1 = 使能错误事件中断.
[10]	<b>NYETIEN</b>	<b>NYET 中断使能位</b> 0 = 禁止 NYET 情况的中断. 1 = 使能 NYET 情况的中断.
[9]	<b>STALLIEN</b>	<b>STALL 发送中断使能位</b> 当此位置1, 端点向主机发送 stall 时产生中断. 0 = 禁止 STALL 中断. 1 = 使能 STALL 中断.
[8]	<b>NAKIEN</b>	<b>NAK 发送中断使能位</b> 当此位置1, 端点向主机发送 NAK 时产生中断. 0 = 禁止 NAK 中断 1 = 使能 NAK 中断.
[7]	<b>PINGIEN</b>	<b>PING Token中断使能位</b> 当此位置1, 端点收到 PING token时产生中断. 0 = 禁止PING token 中断. 1 = 使能PING token中断.
[6]	<b>INTKIEN</b>	<b>IN Token 中断使能位</b> 当此位置1, 端点收到 IN token 时产生中断. 0 = 禁止IN token中断 1 = 使能IN token中断.
[5]	<b>OUTTKIEN</b>	<b>OUT Token 中断使能位</b> 当此位置1, 端点收到 OUT token时产生中断. 0 = 禁止 OUT token 中断. 1 = 使能 OUT token 中断.
[4]	<b>RXPKIEN</b>	<b>数据包接收中断使能位</b> 当此位置1, 端点收到一个数据包时产生中断. 0 = 禁止数据包接收中断 1 = 使能数据包接收中断.
[3]	<b>TXPKIEN</b>	<b>数据包发送中断使能位</b> 当此位置1, 端点向主机发送一个数据包时产生中断. 0 = 禁止向主机发送数据包的中断 1 = 使能向主机发送数据包的中断.
[2]	<b>SHORTTXIEN</b>	<b>短包发送中断使能位</b> 当此位置1, 当这个端点向主机发送或接收到短包时产生中断 0 = 禁止短包中断. 1 = 使能短包中断.
[1]	<b>BUFEMPTYIEN</b>	<b>缓存空中断</b> 0 = 禁止缓存空中断. 1 = 使能缓存空中断.

[0]	<b>BUFFULLIEN</b>	缓存满中断 0 = 禁止缓存满中断 1 = 使能缓存满中断.
-----	-------------------	--------------------------------------

**HSUSBD\_EPADATCNT~ HSUSBD\_EPLDATCNT 端点A~L 数据有效个数寄存器**

寄存器	偏移量	R/W	描述	复位值
HSUSBD_EPADATCNT	HSUSBD_BA+0x070	R	端点 A 数据有效个数寄存器	0x0000_0000
HSUSBD_EPBDATCNT	HSUSBD_BA+0x098	R	端点 B 数据有效个数寄存器	0x0000_0000
HSUSBD_EPCDATCNT	HSUSBD_BA+0x0C0	R	端点 C 数据有效个数寄存器	0x0000_0000
HSUSBD_EPDDATCNT	HSUSBD_BA+0x0E8	R	端点 D 数据有效个数寄存器	0x0000_0000
HSUSBD_EPEDATCNT	HSUSBD_BA+0x110	R	端点 E 数据有效个数寄存器	0x0000_0000
HSUSBD_EPFDATCNT	HSUSBD_BA+0x138	R	端点 F 数据有效个数寄存器	0x0000_0000
HSUSBD_EPGDATCNT	HSUSBD_BA+0x160	R	端点 G 数据有效个数寄存器	0x0000_0000
HSUSBD_EPHDATCNT	HSUSBD_BA+0x188	R	端点 H 数据有效个数寄存器	0x0000_0000
HSUSBD_EPIDATCNT	HSUSBD_BA+0x1B0	R	端点 I 数据有效个数寄存器	0x0000_0000
HSUSBD_EPJDATCNT	HSUSBD_BA+0x1D8	R	端点 J 数据有效个数寄存器	0x0000_0000
HSUSBD_EPKDATCNT	HSUSBD_BA+0x200	R	端点 K 数据有效个数寄存器	0x0000_0000
HSUSBD_EPLDATCNT	HSUSBD_BA+0x228	R	端点 L 数据有效个数寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved	DMALOOP						
23	22	21	20	19	18	17	16
DMALOOP							
15	14	13	12	11	10	9	8
DATCNT							
7	6	5	4	3	2	1	0
DATCNT							

位	描述	
[31]	Reserved	保留.
[30:16]	DMALOOP	DMA 循环 这个字节是未完成 DMA 循环. 每个循环传输32个字节

[15:0]	<b>DATCNT</b>	<b>数据个数</b> 对于输入端点 (EPDIR(HSUSBD_EPxCFG[3] 为高.), 这里是输入端点缓存里的数据有效个数 对于输出端点 (EPDIR(HSUSBD_EPxCFG[3]为低.), 这里是主机传送来的数据有效个数.
--------	---------------	---

**HSUSBD\_EPARSPCTL~ HSUSBD\_EPLRSPCTL 端点 A~L 响应控制寄存器**

寄存器	偏移量	R/W	描述	复位值
HSUSBD_EPARSPCTL	HSUSBD_BA+0x074	R/W	端点 A 响应控制寄存器	0x0000_0000
HSUSBD_EPBRSPCTL	HSUSBD_BA+0x09C	R/W	端点 B 响应控制寄存器	0x0000_0000
HSUSBD_EPCRSPCTL	HSUSBD_BA+0x0C4	R/W	端点 C 响应控制寄存器	0x0000_0000
HSUSBD_EPD_RSPCTL	HSUSBD_BA+0x0EC	R/W	端点 D 响应控制寄存器	0x0000_0000
HSUSBD_EPERSPCTL	HSUSBD_BA+0x114	R/W	端点 E 响应控制寄存器	0x0000_0000
HSUSBD_EPFRSPCTL	HSUSBD_BA+0x13C	R/W	端点 F 响应控制寄存器	0x0000_0000
HSUSBD_EPGRSPCTL	HSUSBD_BA+0x164	R/W	端点 G 响应控制寄存器	0x0000_0000
HSUSBD_EPHRSPCTL	HSUSBD_BA+0x18C	R/W	端点 H 响应控制寄存器	0x0000_0000
HSUSBD_EPIRSPCTL	HSUSBD_BA+0x1B4	R/W	端点 I 响应控制寄存器	0x0000_0000
HSUSBD_EPJRSPCTL	HSUSBD_BA+0x1DC	R/W	端点 J 响应控制寄存器	0x0000_0000
HSUSBD_EPKRSPCTL	HSUSBD_BA+0x204	R/W	端点 K 响应控制寄存器	0x0000_0000
HSUSBD_EPLRSPCTL	HSUSBD_BA+0x22C	R/W	端点 L 响应控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
DISBUF	SHORTTXEN	ZEROLEN	HALT	TOGGLE	MODE	FLUSH	

位	描述	
[31:8]	Reserved	保留.

[7]	<b>DISBUF</b>	<b>缓存禁止位</b> 这一位用来接收不知道大小的OUT短包，接收包的大小依据HSUSBD_EPxDATCNT寄存器。 0 = 收到 Bulk-OUT 短包时收下数据. 1 = 收到 Bulk-OUT 短包时禁止缓存.
[6]	<b>SHORTTXEN</b>	<b>短包传输使能位</b> 此位只在自动确认模式下有效。如果此位置1，缓存里剩下的小于最大包大小的数据被确认作为最后一笔传输。当数据包发送后此位自动清0 0 = 缓存里数据个数小于最大包个数时，收到 IN token 不发出去(回NAK) 1 = 缓存里数据个数小于最大包个数时，收到 IN token 就发出去.
[5]	<b>ZEROLEN</b>	<b>0长度</b> 这一位用来发送0长度的包来响应 IN-token. 当此位置1，接收到IN-token 后会发送0长度的包。此位在发送完0长度的包之后自动清除 0 = 收到IN-token 不发送0长度的包 1 = 收到IN-token 发送0长度的包.
[4]	<b>HALT</b>	<b>端点挂起</b> 这一位用来收到主机令牌后发送 STALL握手 。当本地CPU检测到(ep_halt)特性后，需要写1清0.. 0 = 从主机收到令牌后不发送 STALL 1 = 从主机收到令牌后发送 STALL.
[3]	<b>TOGGLE</b>	<b>端点交替</b> 这一位用来清除端点数据交替位，读得到目前端点数据交替位 本地CPU在接收到主机的总线请求或清除特征字(ep_halt)请求时可以用这一位来建立端点交替。只有当交替位为1，这一位才可以写入 0 = 不清除端点交替位. 1 = 清除端点交替位
[2:1]	<b>MODE</b>	<b>模式控制</b> 这两位决定输入端点的操作模式. 00: 自动确认模式 01: 手动确认模式 10: 自由模式 11: 保留 这两位对输出端点无效，设为11会选择自动确认模式.
[0]	<b>FLUSH</b>	<b>清空缓存</b> 写1清除缓存以及相应的EP_AVAIL 寄存器。此位自动清0.这一位常在配置后写1. 0 = 包缓存不被清空. 1 = 清空包缓存.

## HSUSBD\_EPAMPS~ HSUSBD\_EPLMPS 端点 A~L 最大包长度寄存器

寄存器	偏移量	R/W	描述	复位值
HSUSBD_EPAMPS	HSUSBD_BA+0x078	R/W	端点 A 最大包长度寄存器	0x0000_0000
HSUSBD_EPBMPS	HSUSBD_BA+0x0A0	R/W	端点 B 最大包长度寄存器	0x0000_0000
HSUSBD_EPCMPS	HSUSBD_BA+0x0C8	R/W	端点 C 最大包长度寄存器	0x0000_0000
HSUSBD_EPDMPS	HSUSBD_BA+0x0F0	R/W	端点 D 最大包长度寄存器	0x0000_0000
HSUSBD_EPEMPS	HSUSBD_BA+0x118	R/W	端点 E 最大包长度寄存器	0x0000_0000
HSUSBD_EPFMPS	HSUSBD_BA+0x140	R/W	端点 F 最大包长度寄存器	0x0000_0000
HSUSBD_EPGMPS	HSUSBD_BA+0x168	R/W	端点 G 最大包长度寄存器	0x0000_0000
HSUSBD_EPHMPS	HSUSBD_BA+0x190	R/W	端点 H 最大包长度寄存器	0x0000_0000
HSUSBD_EPIMPS	HSUSBD_BA+0x1B8	R/W	端点 I 最大包长度寄存器	0x0000_0000
HSUSBD_EPJMPS	HSUSBD_BA+0x1E0	R/W	端点 J 最大包长度寄存器	0x0000_0000
HSUSBD_EPKMPS	HSUSBD_BA+0x208	R/W	端点 K 最大包长度寄存器	0x0000_0000
HSUSBD_EPLMPS	HSUSBD_BA+0x230	R/W	端点 L 最大包长度寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved					EPMPS		
7	6	5	4	3	2	1	0
EPMPS							

位	描述	
[31:11]	Reserved	保留.
[10:0]	EPMPS	端点最大包大小 这里决定端点的最大包大小

HSUSBD\_EPATXCNT~ HSUSBD\_EPLTXCNT 端点 A~L传输个数寄存器

寄存器	偏移量	R/W	描述	复位值
HSUSBD_EPATXCNT	HSUSBD_BA+0x07C	R/W	端点 A 传输个数寄存器	0x0000_0000
HSUSBD_EPBTXCNT	HSUSBD_BA+0x0A4	R/W	端点 B 传输个数寄存器	0x0000_0000
HSUSBD_EPCTXCNT	HSUSBD_BA+0x0CC	R/W	端点 C 传输个数寄存器	0x0000_0000
HSUSBD_EPDTXCNT	HSUSBD_BA+0x0F4	R/W	端点 D 传输个数寄存器	0x0000_0000
HSUSBD_EPETXCNT	HSUSBD_BA+0x11C	R/W	端点 E 传输个数寄存器	0x0000_0000
HSUSBD_EPFTXCNT	HSUSBD_BA+0x144	R/W	端点 F 传输个数寄存器	0x0000_0000
HSUSBD_EPGTXCNT	HSUSBD_BA+0x16C	R/W	端点 G 传输个数寄存器	0x0000_0000
HSUSBD_EPHTXCNT	HSUSBD_BA+0x194	R/W	端点 H 传输个数寄存器	0x0000_0000
HSUSBD_EPITXCNT	HSUSBD_BA+0x1B4	R/W	端点 I 传输个数寄存器	0x0000_0000
HSUSBD_EPJTXCNT	HSUSBD_BA+0x1E4	R/W	端点 J 传输个数寄存器	0x0000_0000
HSUSBD_EPKTXCNT	HSUSBD_BA+0x20C	R/W	端点 K 传输个数寄存器	0x0000_0000
HSUSBD_EPLTXCNT	HSUSBD_BA+0x234	R/W	端点 L 传输个数寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved				TXCNT			
7	6	5	4	3	2	1	0
TXCNT							

位	描述	
[31:11]	Reserved	保留.
[10:0]	TXCNT	端点传输个数 对于输入端点，手动确认模式这里决定发送给主机的数据个数 对输出端点无效.

HSUSBD\_EPACFG~ HSUSBD\_EPLCFG 端点 A~L 配置寄存器

寄存器	偏移量	R/W	描述	复位值
HSUSBD_EPA_CFG	HSUSBD_BA+0x080	R/W	端点 A 配置寄存器	0x0000_0012
HSUSBD_EPB_CFG	HSUSBD_BA+0x0A8	R/W	端点 B 配置寄存器	0x0000_0022
HSUSBD_EPC_CFG	HSUSBD_BA+0x0D0	R/W	端点 C 配置寄存器	0x0000_0032
HSUSBD_EPD_CFG	HSUSBD_BA+0x0F8	R/W	端点 D 配置寄存器	0x0000_0042
HSUSBD_EPE_CFG	HSUSBD_BA+0x120	R/W	端点 E 配置寄存器	0x0000_0052
HSUSBD_EPF_CFG	HSUSBD_BA+0x148	R/W	端点 F 配置寄存器	0x0000_0062
HSUSBD_EPG_CFG	HSUSBD_BA+0x170	R/W	端点 G 配置寄存器	0x0000_0072
HSUSBD_EPH_CFG	HSUSBD_BA+0x198	R/W	端点 H 配置寄存器	0x0000_0082
HSUSBD_EPIC_CFG	HSUSBD_BA+0x1C0	R/W	端点 I 配置寄存器	0x0000_0092
HSUSBD_EPJ_CFG	HSUSBD_BA+0x1E8	R/W	端点 J 配置寄存器	0x0000_00A2
HSUSBD_EPK_CFG	HSUSBD_BA+0x210	R/W	端点 K 配置寄存器	0x0000_00B2
HSUSBD_EPL_CFG	HSUSBD_BA+0x238	R/W	端点 L 配置寄存器	0x0000_00C2

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
EPNUM				EPDIR	EPTYPE		EPEN

位	描述	
[31:6]	Reserved	保留.
[7:4]	EPNUM	端点号 这里选择端点号，有效值是1到15。 <b>注意:</b> 不支持两个端点有同一个端点号.

[3]	<b>EPDIR</b>	<b>端点方向</b> 0 = 输出端点 (主机输出到设备). 1 = 输入端点 (主机从设备输入).
[2:1]	<b>EPTYPE</b>	<b>端点类型</b> 这里选择端点类型, 端点 0是控制端点. 00 = 保留. 01 = 批量传输. 10 = 中断传输.. 11 = 等时传输.
[0]	<b>EPEN</b>	<b>端点有效</b> 置1使能端点, 这一位对端点0无效, 端点0总是使能. 0 = 禁止端点. 1 = 使能端点.

HSUSBD\_EPABUFSTART~ HSUSBD\_EPLBUFSTART 端点 A~L 内存起始地址寄存器

寄存器	偏移量	R/W	描述	复位值
HSUSBD_EPABUFSTART	HSUSBD_BA+0x084	R/W	端点 A 内存起始地址寄存器	0x0000_0000
HSUSBD_EPBBUFSTART	HSUSBD_BA+0x0AC	R/W	端点 B 内存起始地址寄存器	0x0000_0000
HSUSBD_EPCBUFSTART	HSUSBD_BA+0x0D4	R/W	端点 C 内存起始地址寄存器	0x0000_0000
HSUSBD_EPDBUFSTART	HSUSBD_BA+0x0FC	R/W	端点 D 内存起始地址寄存器	0x0000_0000
HSUSBD_EPEBUFSTART	HSUSBD_BA+0x124	R/W	端点 E 内存起始地址寄存器	0x0000_0000
HSUSBD_EPFBUFSTART	HSUSBD_BA+0x14C	R/W	端点 F 内存起始地址寄存器	0x0000_0000
HSUSBD_EPGBUFSTART	HSUSBD_BA+0x174	R/W	端点 G 内存起始地址寄存器	0x0000_0000
HSUSBD_EPHBUFSTART	HSUSBD_BA+0x19C	R/W	端点 H 内存起始地址寄存器	0x0000_0000
HSUSBD_EPIBUFSTAR	HSUSBD_BA+0x1C4	R/W	端点 I 内存起始地址寄存器	0x0000_0000
HSUSBD_EPJBUFSTART	HSUSBD_BA+0x1EC	R/W	端点 J 内存起始地址寄存器	0x0000_0000
HSUSBD_EPKBUFSTART	HSUSBD_BA+0x214	R/W	端点 K 内存起始地址寄存器	0x0000_0000
HSUSBD_EPLBUFSTART	HSUSBD_BA+0x23C	R/W	端点 L 内存起始地址寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved				SADDR			
7	6	5	4	3	2	1	0
SADDR							

位	描述	
[31:12]	Reserved	保留.
[11:0]	SADDR	端点起始地址 这里是对应端点的内存起始地址.

## HSUSBD\_EPABUFEND~ HSUSBD\_EPLBUFEND 端点 A~L 内存结束地址寄存器

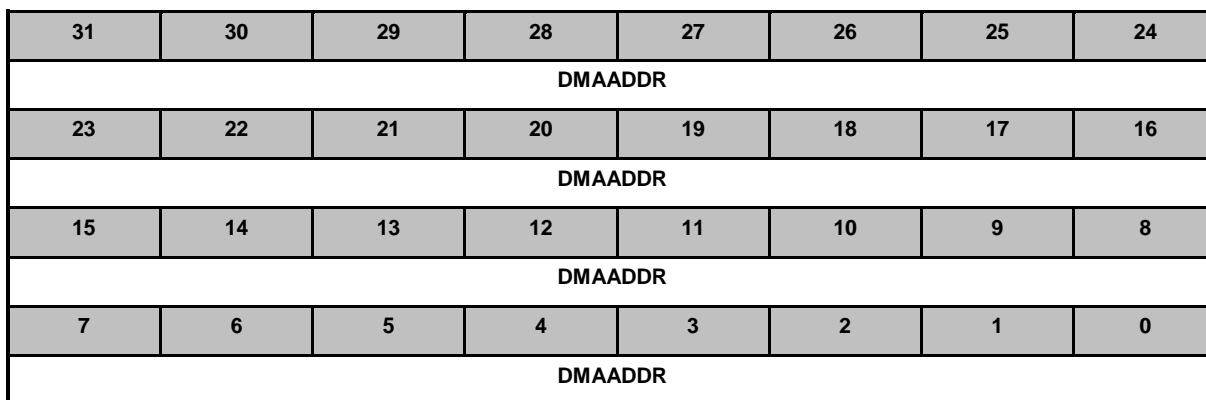
寄存器	偏移量	R/W	描述	复位值
HSUSBD_EPABUFEND	HSUSBD_BA+0x088	R/W	端点 A 内存结束地址寄存器	0x0000_0000
HSUSBD_EPBBUFEND	HSUSBD_BA+0x0B0	R/W	端点 B 内存结束地址寄存器	0x0000_0000
HSUSBD_EPCBUFEND	HSUSBD_BA+0x0D8	R/W	端点 C 内存结束地址寄存器	0x0000_0000
HSUSBD_EPDBUFEND	HSUSBD_BA+0x100	R/W	端点 D 内存结束地址寄存器	0x0000_0000
HSUSBD_EPEBUFEND	HSUSBD_BA+0x128	R/W	端点 E 内存结束地址寄存器	0x0000_0000
HSUSBD_EPFBUFEND	HSUSBD_BA+0x150	R/W	端点 F 内存结束地址寄存器	0x0000_0000
HSUSBD_EPGBUFEND	HSUSBD_BA+0x178	R/W	端点 G 内存结束地址寄存器	0x0000_0000
HSUSBD_EPHBUFEND	HSUSBD_BA+0x1A0	R/W	端点 H 内存结束地址寄存器	0x0000_0000
HSUSBD_EPIBUFEND	HSUSBD_BA+0x1C8	R/W	端点 I 内存结束地址寄存器	0x0000_0000
HSUSBD_EPJBUFEND	HSUSBD_BA+0x1F0	R/W	端点 J 内存结束地址寄存器	0x0000_0000
HSUSBD_EPKBUFEND	HSUSBD_BA+0x218	R/W	端点 K 内存结束地址寄存器	0x0000_0000
HSUSBD_EPLBUFEND	HSUSBD_BA+0x240	R/W	端点 L 内存结束地址寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved				EADDR			
7	6	5	4	3	2	1	0
EADDR							

位	描述	
[31:12]	Reserved	保留.
[11:0]	EADDR	端点结束地址 这里是对应端点的内存结束地址.

**HSUSBD DMAADDR AHB 地址寄存器**

寄存器	偏移量	R/W	描述	复位值
HSUSBD_DMAADDR	HSUSBD_BA+0x700	R/W	AHB DMA 地址寄存器	0x0000_0000



位	描述	
[31:0]	DMAADDR	DMAADDR 这里指定 DMA 的读写地址，地址必须32位对齐..

## USBD\_PHYCTL USB PHY控制寄存器

寄存器	偏移量	R/W	描述	复位值
HSUSBD_PHYCTL	HSUSBD_BA+0x704	R/W	USB PHY 控制寄存器	0x0000_0420

31	30	29	28	27	26	25	24
VBUSDET	Reserved						WKEN
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved						PHYEN	DPPUEN
7	6	5	4	3	2	1	0
Reserved							

位	描述	
[31]	<b>VBUSDET</b>	<b>VBUS 状态</b> 0 = 没有检测到VBUS. 1 = 检测到 VBUS
[30:25]	<b>Reserved</b>	保留.
[24]	<b>WKEN</b>	<b>唤醒使能位</b> 0 = 禁止唤醒功能. 1 = 使能唤醒功能.
[23:10]	<b>Reserved</b>	保留.
[9]	<b>PHYEN</b>	<b>PHY 挂起使能位</b> 0 = USB PHY 挂起. 1 = USB PHY 未挂起.
[8]	<b>DPPUEN</b>	<b>DP 上拉</b> 0 = 禁止 D+ 上拉电阻. 1 = 使能 D+ 上拉电阻.
[7:0]	<b>Reserved</b>	保留.

## 6.32 USBH USB2.0主机接口

### 6.32.1 概述

这颗芯片配有USB 2.0 HS/FS 主机控制器 (USBH) , 支持增强型主机控制器接口(EHCI) 以及开放主机控制器接口(OpenHCI, OHCI) 协议, 主机控制器寄存器级的操作以管理设备以及USB通讯。

USBH 内建一个根 Hub 以及一个 USB 端口, 实时在系统内存与USB总线间传输数据的 DMA, 端口电源控制以及过流检测功能。

USBH 负责检测USB 设备的插入和拔出, 管理数据传输, 收集状态以及使能 USB 总线, 提供电源控制以及检测插入设备的过流的情况。

### 6.32.2 特性

- 支持USB规范2.0版本
- 支持EHCI规范 1.0版本
- 支持OpenHCI规范 1.0版本
- 支持高速 (480Mbps), 全速(12Mbps) 以及低速 (1.5Mbps) USB 设备.
- 支持控制, 批量, 中断, 等时 以及拆分 传输.
- 内建根 Hub.
- 支持端口路径逻辑将全速, 低速设备接入OHCI 控制器.
- 支持 USB 主机端口与USB 设备端口共享 (OTG 功能).
- 支持端口电源控制以及端口过流检测
- 支持实时 DMA 数据传输.

## 6.32.3 框图

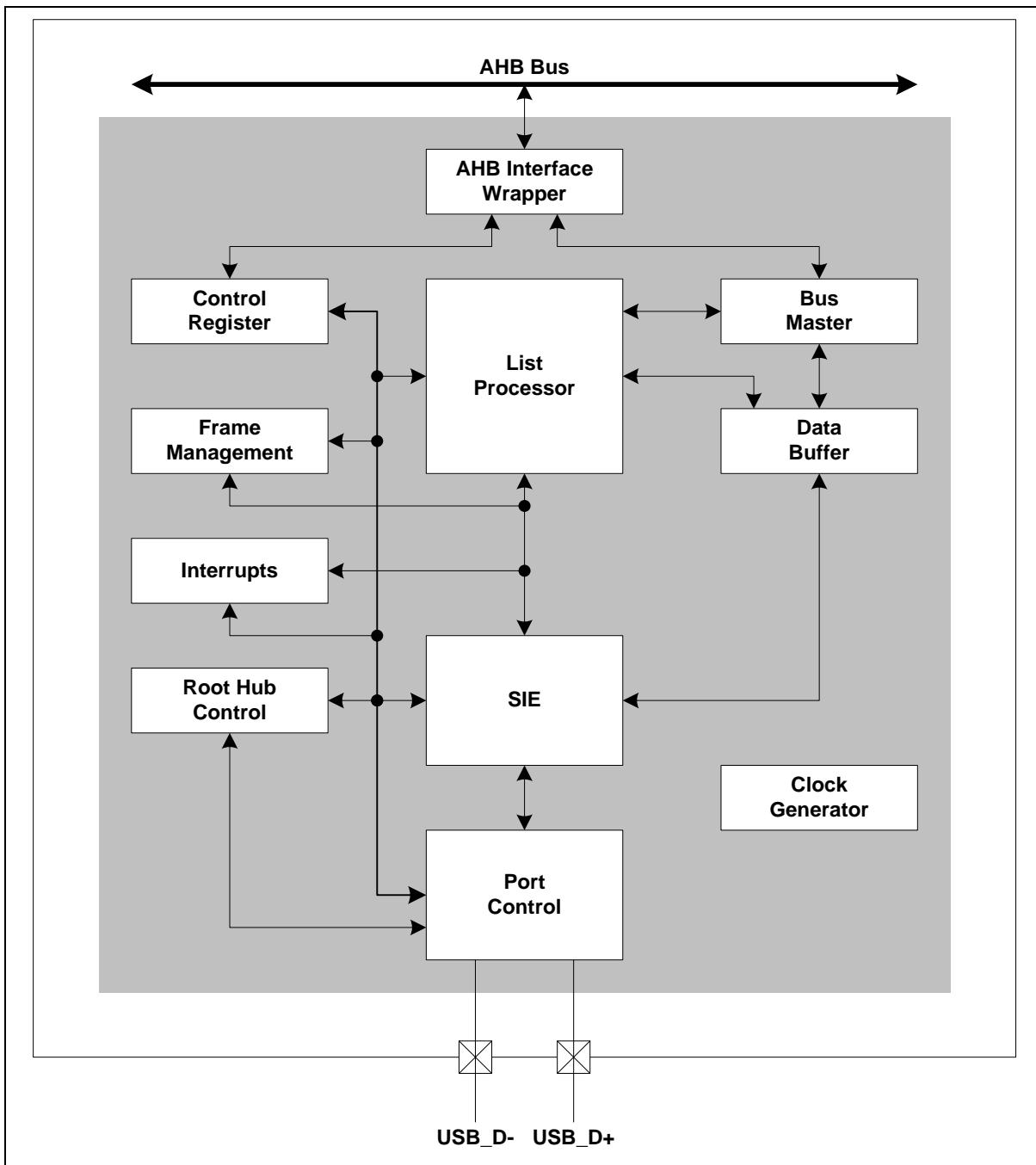


图 6.32-1 USB 2.0 FS 主机控制器框图

## 6.32.4 基本配置

## 6.32.4.1 USBH\_EHCI 控制器基本配置

- 时钟源配置
  - 配置USBSEL (CLK\_CLKSEL0[8])选择USBH时钟源

- 配置USBDIV (CLK\_CLKDIV0[7:4]).设置USBH除频
- 配置HSUSBHCKEN (CLK\_AHBCLK[16])使能 USBH时钟.
- 复位配置
  - 通过UHC20RST (SYS\_IPRST0[16]) 复位USBH 控制器.

#### 6.32.4.2 USBH OHCI 控制器基本配置

- 时钟源配置
  - 配置USBSEL (CLK\_CLKSEL0[8])选择USBH时钟源
  - 配置USBDIV (CLK\_CLKDIV0[7:4]).设置USBH除频
  - 配置HSUSBHCKEN (CLK\_AHBCLK[16])使能 USBH时钟.
- 复位配置
  - 通过UHC20RST (SYS\_IPRST0[16]) 复位USBH 控制器.
- 管脚配置

组	管脚名称	GPIO	MFP
USB	USB_D+	PA.14	MFP14
	USB_D-	PA.13	MFP14
	USB_OTG_ID	PA.15	MFP14
	USB_VBUS	PA.12	MFP14
	USB_VBUS_EN	PB.6, PB.15	MFP14
	USB_VBUS_ST	PB.7, PC.14, PD.4	MFP14

#### 6.32.5 功能描述

##### 6.32.5.1 EHCI 控制器

EHCI 与系统通过AHB接口连接， CPU 通过AHB读写寄存器。CPU 作为总线管理者发起传输，EHCI做出响应。例如，如果 CPU 要向HCl寄存器写数据，CPU提供地址和数据值。EHCI 通过该地址找到寄存器填入数据值。如果读寄存器，EHCI 通过该地址找到寄存器读到值并放在系统总线上。

同样的，当 EHCI 要执行数据传输，它就作为管理者发起数据传输，这时系统内存作为总线目标. EHCI, 作为管理者可以执行两种数据传输：从 EHCI 到系统内存,或从系统内存到 EHCI. 当 EHCI 想要将从 USB2.0 设备来的数据存入系统内存，它就通过内存接口信号，配置EHC1 写控制字 (write)，数据以及要写入内存的数据个数，内存控制器收到数据将其存入内存。如果数据从内存传到设备， EHCI 执行向系统总线的读操作。内存控制器通过内存接口信号提供数据， EHCI 接收到数据然后将其传向设备

##### 6.32.5.2 OHCI控制器

##### AHB 接口

OpenHCl 主机控制器与系统通过AHB 总线连接。需要主从两种总线操作。作主，主机控制器响应 AHB 总线上的时钟周期通过EDs 和 TDs 在内存和本地数据缓存间传输数据。做从，主机控制器监控AHB 总

线的时钟周期决定何时响应这些信号。通过**AHB** 总线从机接口非实时的访问主机控制器寄存器。

### 主机控制器

主机控制器包含 5 个功能模块，包括列表处理器，帧管理，中断执行，主控制器总线管理器以及数据缓存。

列表处理器管理主机控制器驱动的数据结构以及主机控制器其他协同事项。

帧管理模块负责管理**USB** 协议以及 **OpenHCI**协议要求的帧相关事务，包括：

- 管理 **OpenHCI** 帧特殊操作寄存器
- 管理最大数据包计数.
- 执行**SIE**要求的**USB** 传输帧格式
- 产生**SIE**要求的 **SOF token**

中断是主机控制器驱动**HC-initiated** 通讯的一种通讯手段。有一些事件会触发主机控制器中断，每个特殊事件置位**HcInterruptStatus**寄存器的特定位.

主控制器总线管理是数据通道的核心单元，它联系各种访问到**AHB**总线上..。主控制有两个总线管理员：列表处理器以及数据缓存引擎。

数据缓存作为总线管理者与**SIE**之间的数据接口，它包含64个字节的固定基址的双向异步**FIFO**，以及一个双字长**AHB** 保持寄存器

### USB 接口

**USB** 接口包含内建的根以及一个 **USB** 端口，端口 1和串行接口引擎 (**SIE**) 和**USB** 时钟发生器。 接口执行主控制器需求的总线传输以及**USB**的**hub**以及端口管理。

**SIE** 负责管理所有的**USB**事物，控制总线协议，包的生成与解包，数据的并串转换，**CRC**码，位填充，以及不归零码的编码。所有的**USB** 事物由列表处理器以及帧管理单元提出。

根**Hub** 是独立控制的端口集， **hub** 管理所有端口的功能的控制与状态。

### 6.32.6 寄存器映射

R: 只读, W: 只写, R/W: 可读可写

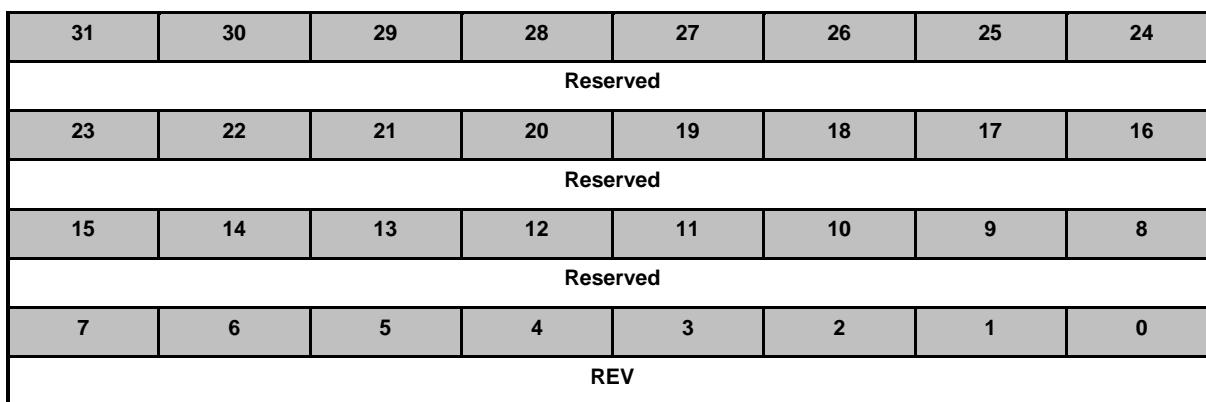
寄存器	偏移量	R/W	描述	复位值
<b>USBH 基地址:</b>				
<b>USBH_BA = 0x4000_9000</b>				
<b>HSUSBH_BA = 0x4001_A000</b>				
HcRevision	USBH_BA+0x000	R	主机控制器版本寄存器	0x0000_0110
HcControl	USBH_BA+0x004	R/W	主机控制器控制寄存器	0x0000_0000
HcCommandStatus	USBH_BA+0x008	R/W	主机控制器指令状态寄存器	0x0000_0000
HcInterruptStatus	USBH_BA+0x00C	R/W	主机控制器中断状态寄存器	0x0000_0000
HcInterruptEnable	USBH_BA+0x010	R/W	主机控制器中断使能寄存器	0x0000_0000
HcInterruptDisable	USBH_BA+0x014	R/W	主机控制器中断禁止寄存器	0x0000_0000
HcHCCA	USBH_BA+0x018	R/W	主机控制器通讯范围寄存器	0x0000_0000
HcPeriodCurrentED	USBH_BA+0x01C	R/W	主机控制器周期传输目前 ED 寄存器	0x0000_0000
HcControlHeadED	USBH_BA+0x020	R/W	主机控制器控制传输首个 ED 寄存器	0x0000_0000
HcControlCurrentED	USBH_BA+0x024	R/W	主机控制器控制传输目前 ED 寄存器	0x0000_0000
HcBulkHeadED	USBH_BA+0x028	R/W	主机控制器块传输首个 ED 寄存器	0x0000_0000
HcBulkCurrentED	USBH_BA+0x02C	R/W	主控制器块传输目前 ED 寄存器	0x0000_0000
HcDoneHead	USBH_BA+0x030	R/W	主控制器完成首个寄存器	0x0000_0000
HcFmInterval	USBH_BA+0x034	R/W	主机控制器帧间隔寄存器	0x0000_2EDF
HcFmRemaining	USBH_BA+0x038	R	主机控制器帧剩余寄存器	0x0000_0000
HcFmNumber	USBH_BA+0x03C	R	主机控制器帧号码寄存器	0x0000_0000
HcPeriodicStart	USBH_BA+0x040	R/W	主机控制器周期开始寄存器	0x0000_0000
HcLSThreshold	USBH_BA+0x044	R/W	主机控制器低速阈值寄存器	0x0000_0628
HcRhDescriptorA	USBH_BA+0x048	R/W	主机控制器根 Hub 描述符 A 寄存器	0x0000_0902
HcRhDescriptorB	USBH_BA+0x04C	R/W	主机控制器根 Hub 描述符 B 寄存器	0x0000_0000
HcRhStatus	USBH_BA+0x050	R/W	主机控制器根 Hub 状态 寄存器	0x0000_0000
HcRhPortStatus1	USBH_BA+0x058	R/W	主机控制器根 Hub 端口状态 寄存器1	0x0000_0000
HcPhyControl	USBH_BA+0x200	R/W	主机控制器 PHY 控制寄存器	0x0000_0000
HcMiscControl	USBH_BA+0x204	R/W	主机控制器杂项控制寄存器	0x0000_0000
EHCVNR	HSUSBH_BA+0x000	R	EHCI 版本号寄存器	0x0095_0020
EHCSPR	HSUSBH_BA+0x004	R	EHCI 结构参数寄存器	0x0000_0012

寄存器	偏移量	R/W	描述	复位值
<b>EHCCPR</b>	HSUSBH_BA+0x008	R	EHCI 性能参数寄存器	0x0000_0000
<b>UCMDR</b>	HSUSBH_BA+0x020	R/W	USB 指令寄存器	0x0008_0000
<b>USTSR</b>	HSUSBH_BA+0x024	R/W	USB 状态寄存器	0x0000_1000
<b>UIENR</b>	HSUSBH_BA+0x028	R/W	USB 中断使能寄存器	0x0000_0000
<b>UFINDR</b>	HSUSBH_BA+0x02C	R/W	USB 帧索引寄存器	0x0000_0000
<b>UPFLBAR</b>	HSUSBH_BA+0x034	R/W	USB 周期帧列表基地址寄存器	0x0000_0000
<b>UCALAR</b>	HSUSBH_BA+0x038	R/W	USB 目前异步列表地址寄存器	0x0000_0000
<b>UASSTR</b>	HSUSBH_BA+0x03C	R/W	USB 异步计划休眠定时器寄存器	0x0000_0BD6
<b>UCFGR</b>	HSUSBH_BA+0x060	R/W	USB 配置标志寄存器	0x0000_0000
<b>UPSCR0</b>	HSUSBH_BA+0x064	R/W	USB 端口 0 状态和控制寄存器	0x0000_2000
<b>UPSCR1</b>	HSUSBH_BA+0x068	R/W	USB 端口 1 状态和控制寄存器	0x0000_2000
<b>USBPCR0</b>	HSUSBH_BA+0x0C4	R/W	USB PHY 0 控制寄存器	0x0000_0060
<b>USBPCR1</b>	HSUSBH_BA+0x0C8	R/W	USB PHY 1 控制寄存器	0x0000_0020

### 6.32.7 寄存器描述

#### HcRevision 主机控制器版本寄存器

寄存器	偏移量	R/W	描述	复位值
HcRevision	USBH_BA+0x000	R	主机控制器版本寄存器	0x0000_0110



位	描述								
[31:8]	Reserved	保留.							
[7:0]	REV	版本号 标明硬件的 Open HCI 协议版本号. 主控制器支持 1.1 标准. (X.Y = XYh).							

HcControl 主控制器控制寄存器

寄存器	偏移量	R/W	描述	复位值
HcControl	USBH_BA+0x004	R/W	主控制器控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
HCFS		BLE	CLE	IE	PLE	CBSR	

位	描述	
[31:8]	Reserved	保留.
[7:6]	HCFS	<p><b>主控制器功能状态</b></p> <p>这里设定主机控制器状态. 控制器当检测到下游端口发来的恢复信号后会从USBSUSPEND状态变为USBRESUME状态。状态包括:</p> <p>00 = USB 挂起. 01 = USB 操作. 10 = USB 恢复. 11 = USB 复位.</p>
[5]	BLE	<p><b>批量列表使能位</b></p> <p>0 = 禁止在下一个SOF ((帧起始)处理批量列表 1 = 在下一帧处理批量列表.</p>
[4]	CLE	<p><b>控制列表使能位</b></p> <p>0 = 禁止在下一个SOF ((帧起始)处理控制列表 1 = 在下一帧处理控制列表.</p>
[3]	IE	<p><b>同步列表使能位</b></p> <p>ISOEn 和 PLE (HcControl[2]) 都为高的时候使能控制器处理同步列表。ISOEn 或 PLE (HcControl[2]) 有一个为低禁止主机控制器处理同步列表</p> <p>0 = 禁止在下一个SOF (Start-Of-Frame)处理同步列表 1 = 如果PLE (HcControl[2]) 也为高, 在下一帧处理同步列表.</p>
[2]	PLE	<p><b>周期列表使能位</b></p> <p>使能处理周期 (中断和同步) 列表。主控制器在处理周期传输帧前确认此位。</p> <p>0 = 禁止在下一个SOF (帧起始)处理中断和同步列表 1 = 在下一帧处理中断和同步列表..</p> <p><b>注意:</b> 要处理同步列表, PLE 和 IE (HcControl[3]) 都要为高.</p>

[1:0]	<b>CBSR</b>	<b>控制与批量服务比例</b> 控制端点与批量端点的服务比例。在处理非周期列表之前，控制器根据有多少非空的控制端点与批量端点的比例，决定是继续服务其他的控制端点，还是切换到批量端点。帧切换的时候内部计数器会保持。当复位的时候HCD 负责恢复这个值。 00 = 控制端点与批量端点服务比例是 1:1. 01 = 控制端点 与批量端点服务比例是 2:1. 10 =控制端点 与批量端点服务比例是 3:1. 11 =控制端点 与批量端点服务比例是 4:1.
-------	-------------	---

**HcCommandStatus 主控制器指令状态寄存器**

寄存器	偏移量	R/W	描述	复位值
HcCommandStatus	USBH_BA+0x008	R/W	主控制器指令状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved						SOC	
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved					BLF	CLF	HCR

位	描述	
[31:18]	Reserved	保留.
[17:16]	SOC	<b>流程重写计数</b> 每次流程重写错误时加1，初始化是00b 在11b回绕。即使SO (HcInterruptStatus[0])已经置1，当检测到流程重写错误时也会加1.
[15:3]	Reserved	保留.
[2]	BLF	<b>填入块列表</b> 为1指示批量列表里有活动的 TD。此位可以由软件或主机控制器来置1，在主控制器每次处理批量列表头之后清0. 0 = 在批量列表里没有发现活动的 TD或主机控制器处理到批量列表头. 1 = 块列表加入了活动的 TD.
[1]	CLF	<b>填入控制列表</b> 为1指示控制列表里有活动的 TD。此位可以由软件或主机控制器来置1，在主控制器每次处理到控制列表头之后清0. 0 = 在控制列表里没有发现活动的 TD或主机控制器处理到控制列表头. 1 = 控制列表加入了活动的 TD.
[0]	HCR	<b>主机控制器复位</b> 这里设置主机控制器软复位，完成复位操作后此位由主机控制器清除. 此位置1后不会复位根Hub，也不会向下游设备发送复位信号. 0 = 主机控制器不在软复位状态. 1 = 主机控制器在软复位状态.

**HcInterruptStatus 主机控制器中断状态寄存器**

寄存器	偏移量	R/W	描述	复位值
<b>HcInterruptStatus</b>	USBH_BA+0x00C	R/W	主机控制器中断状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved	RHSC	FNO	Reserved	RD	SF	WDH	SO

位	描述	
[31:7]	Reserved	保留.
[6]	RHSC	<b>根 Hub 状态改变</b> 这一位当HcRhStatus的内容或HcRhPortStatus1内容改变时置位 0 = HcRhStatus 与 HcRhPortStatus1寄存器的值没有改变. 1 = HcRhStatus 与 HcRhPortStatus1寄存器的值改变了.
[5]	FNO	<b>帧编号溢出</b> 这一位当帧编号的bit 15 由 1 变 0 或由 0 变 1 时置位. 0 = 帧编号的bit 15没变 1 = 帧编号的bit 15 由 1 变 0 或由 0 变 1.
[4]	Reserved	保留.
[3]	RD	<b>恢复 信号检测</b> 当从下游端口检测到 恢复信号时置位. 0 = 没有从下游端口检测到恢复 信号. 1 = 从下游端口检测到恢复 信号.
[2]	SF	<b>帧起始</b> 当帧管理功能单元产生帧起始事件时置位， 主机控制器同时产生一个 SOF token. 0 =.没有帧起始事件. 1 =.产生帧起始事件， 主机控制器同时产生一个 SOF token
[1]	WDH	<b>回写完成头</b> 当主控制器将 HcDoneHead 写入 HccaDoneHead 时置位， 之后在此位清除之前 HccaDoneHead 不会更新 0 =.主机控制器没有更新 HccaDoneHead. 1 =.主控制器将HcDoneHead 写入 HccaDoneHead.

[0]	SO	计划表超期 当列表处理器确定发生计划表超期时置位 0 = 计划表超期没有发生. 1 = 发生计划表超期.
-----	----	---

**HcInterruptEnable** 主机控制器中断使能寄存器

寄存器	偏移量	R/W	描述	复位值
<b>HcInterruptEnable</b>	USBH_BA+0x010	R/W	主机控制器中断使能寄存器	0x0000_0000

31	30	29	28	27	26	25	24
MIE	Reserved						
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved	RHSC	FNO	Reserved	RD	SF	WDH	SO

位	描述	
[31]	<b>MIE</b>	<p><b>管理者中断使能位</b>            这一位是整体中断使能，写1允许其他位使能后产生中断。            写操作：            0 = 没影响。            1 = 使能当 RHSC (HcInterruptStatus[6]), FNO (HcInterruptStatus[5]), RD (HcInterruptStatus[3]), SF (HcInterruptStatus[2]), WDH (HcInterruptStatus[1]) 或 SO (HcInterruptStatus[0]) 使能如果相应的 HcInterruptEnable 的对应位为高时产生中断。            读操作：            0 = 当 RHSC (HcInterruptStatus[6]), FNO (HcInterruptStatus[5]), RD (HcInterruptStatus[3]), SF (HcInterruptStatus[2]), WDH (HcInterruptStatus[1]) 或 SO (HcInterruptStatus[0]) 使能如果相应的 HcInterruptEnable 的对应位为高时不会产生中断..            1 = 当 RHSC (HcInterruptStatus[6]), FNO (HcInterruptStatus[5]), RD (HcInterruptStatus[3]), SF (HcInterruptStatus[2]), WDH (HcInterruptStatus[1]) 或 SO (HcInterruptStatus[0]) 使能如果相应的 HcInterruptEnable 的对应位为高时产生中断..</p>
[30:7]	<b>Reserved</b>	保留.
[6]	<b>RHSC</b>	<p><b>根 Hub 状态改变使能位</b>            写操作：            0 = 没影响。            1 = 使能 RHSC (HcInterruptStatus[6]) 引起的中断。            读操作：            0 = RHSC (HcInterruptStatus[6]) 引起的中断已禁止。            1 = RHSC (HcInterruptStatus[6]) 引起的中断已使能。</p>

[5]	<b>FNO</b>	帧编号溢出使能位 写操作: 0 = 没影响. 1 = 使能 FNO (HcInterruptStatus[5]) 引起的中断. 读操作: 0 = FNO (HcInterruptStatus[5]) 引起的中断已禁止. 1 = FNO (HcInterruptStatus[5]) 引起的中断已使能.
[4]	<b>Reserved</b>	保留.
[3]	<b>RD</b>	<b>Resume</b> 检测使能位 写操作: 0 = 没影响. 1 = 使能由 RD (HcInterruptStatus[3]) 引起的中断. 读操作: 0 = 由 RD (HcInterruptStatus[3]) 引起的中断已禁止. 1 = 由 RD (HcInterruptStatus[3]) 引起的中断已使能.
[2]	<b>SF</b>	帧起始使能位 写操作: 0 = 没影响. 1 = 使能由 SF (HcInterruptStatus[2]) 引起的中断. 读操作: 0 = 由 SF (HcInterruptStatus[2]) 引起的中断已禁止. 1 = 由 SF (HcInterruptStatus[2]) 引起的中断已使能
[1]	<b>WDH</b>	回写完成头使能位 写操作: 0 = 没影响. 1 = 使能由 WDH (HcInterruptStatus[1]) 引起的中断. 读操作: 0 = 由 WDH (HcInterruptStatus[1]) 引起的中断已禁止. 1 = 由 WDH (HcInterruptStatus[1]) 引起的中断已使能.
[0]	<b>SO</b>	计划表超期使能位 写操作: 0 = 没影响. 1 = 使能由 SO (HcInterruptStatus[0]) 引起的中断. 读操作: 0 = 由 SO (HcInterruptStatus[0]) 引起的中断已禁止. 1 = 由 SO (HcInterruptStatus[0]) 引起的中断已使能.

**HcInterruptDisable** 主机控制器中断禁止寄存器

寄存器	偏移量	R/W	描述	复位值
<b>HcInterruptDisable</b>	USBH_BA+0x014	R/W	主机控制器中断禁止寄存器	0x0000_0000

31	30	29	28	27	26	25	24
MIE	Reserved						
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved	RHSC	FNO	Reserved	RD	SF	WDH	SO

位	描述	
[31]	<b>MIE</b>	<p>管理者中断禁止位 总的中断禁止，写1禁用所有中断 写操作： 0 = 没影响。 1 = 禁止当 RHSC (HcInterruptStatus[6]), FNO (HcInterruptStatus[5]), RD (HcInterruptStatus[3]), SF (HcInterruptStatus[2]), WDH (HcInterruptStatus[1]) 或 SO (HcInterruptStatus[0]) 使能如果相应的 HcInterruptEnable的对应位为高时产生中断。 读操作： 0 = 当 RHSC (HcInterruptStatus[6]), FNO (HcInterruptStatus[5]), RD (HcInterruptStatus[3]), SF (HcInterruptStatus[2]), WDH (HcInterruptStatus[1]) 或 SO (HcInterruptStatus[0]) 使能如果相应的 HcInterruptEnable的对应位为高时不会产生中断.. 1 = 当 RHSC (HcInterruptStatus[6]), FNO (HcInterruptStatus[5]), RD (HcInterruptStatus[3]), SF (HcInterruptStatus[2]), WDH (HcInterruptStatus[1]) 或 SO (HcInterruptStatus[0]) 使能如果相应的 HcInterruptEnable的对应位为高时产生中断..</p>
[30:7]	<b>Reserved</b>	保留.
[6]	<b>RHSC</b>	<p>根 Hub 状态改变禁止位 写操作： 0 = 没影响。 1 = 禁止 RHSC (HcInterruptStatus[6]) 引起的中断. 读操作： 0 = RHSC (HcInterruptStatus[6]) 引起的中断已禁止。 1 = RHSC (HcInterruptStatus[6]) 引起的中断已使能</p>

[5]	<b>FNO</b>	帧编号溢出禁止位 写操作: 0 = 没影响. 1 = 禁止 FNO (HcInterruptStatus[5]) 引起的中断. 读操作: 0 = FNO (HcInterruptStatus[5]) 引起的中断已禁止. 1 = FNO (HcInterruptStatus[5]) 引起的中断已使能.
[4]	<b>Reserved</b>	保留.
[3]	<b>RD</b>	<b>Resume</b> 检测禁止位 写操作: 0 = 没影响. 1 = 禁止由 RD (HcInterruptStatus[3]) 引起的中断. 读操作: 0 = 由 RD (HcInterruptStatus[3]) 引起的中断已禁止. 1 = 由 RD (HcInterruptStatus[3]) 引起的中断已使能.
[2]	<b>SF</b>	帧起始禁止位 写操作: 0 = 没影响. 1 = 禁止由 SF (HcInterruptStatus[2]) 引起的中断. 读操作: 0 = 由 SF (HcInterruptStatus[2]) 引起的中断已禁止. 1 = 由 SF (HcInterruptStatus[2]) 引起的中断已使能
[1]	<b>WDH</b>	回写完成头禁止位 写操作: 0 = 没影响. 1 = 禁止由 WDH (HcInterruptStatus[1]) 引起的中断. 读操作: 0 = 由 WDH (HcInterruptStatus[1]) 引起的中断已禁止. 1 = 由 WDH (HcInterruptStatus[1]) 引起的中断已使能.
[0]	<b>SO</b>	计划表超期禁止位 写操作: 0 = 没影响. 1 = 禁止由 SO (HcInterruptStatus[0]) 引起的中断. 读操作: 0 = 由 SO (HcInterruptStatus[0]) 引起的中断已禁止. 1 = 由 SO (HcInterruptStatus[0]) 引起的中断已使能.

HcHCCA 主机控制器通讯范围寄存器

寄存器	偏移量	R/W	描述	复位值
HcHCCA	USBH_BA+0x018	R/W	主机控制器通讯范围寄存器	0x0000_0000

31	30	29	28	27	26	25	24
HCCA							
23	22	21	20	19	18	17	16
HCCA							
15	14	13	12	11	10	9	8
HCCA							
7	6	5	4	3	2	1	0
Reserved							

位	描述	
[31:8]	<b>HCCA</b>	主机控制器通讯范围 指向主机控制器通讯范围 (HCCA)的基地址.
[7:0]	<b>Reserved</b>	保留.

**HcPeriodCurrentED 主机控制器周期目前 ED 寄存器**

寄存器	偏移量	R/W	描述	复位值
HcPeriodCurrentED	USBH_BA+0x01C	R/W	主机控制器周期目前 ED 寄存器	0x0000_0000

31	30	29	28	27	26	25	24
PCED							
23	22	21	20	19	18	17	16
PCED							
15	14	13	12	11	10	9	8
PCED							
7	6	5	4	3	2	1	0
PCED				Reserved			

位	描述	
[31:4]	PCED	周期目前 ED 指向目前同步或中断端点描述符的物理地址.
[3:0]	Reserved	保留.

**HcControlHeadED 主机控制器控制头 ED 寄存器**

寄存器	偏移量	R/W	描述	复位值
HcControlHeadED	USBH_BA+0x020	R/W	主机控制器控制头 ED 寄存器	0x0000_0000

31	30	29	28	27	26	25	24
CHED							
23	22	21	20	19	18	17	16
CHED							
15	14	13	12	11	10	9	8
CHED							
7	6	5	4	3	2	1	0
CHED				Reserved			

位	描述	
[31:4]	CHED	控制头 ED 指向控制列表的第一个端点描述符的物理地址
[3:0]	Reserved	保留.

HcControlCurrentED 主机控制器控制目前 ED 寄存器

寄存器	偏移量	R/W	描述	复位值
HcControlCurrentED	USBH_BA+0x024	R/W	主机控制器控制目前 ED 寄存器	0x0000_0000

31	30	29	28	27	26	25	24
CCED							
23	22	21	20	19	18	17	16
CCED							
15	14	13	12	11	10	9	8
CCED							
7	6	5	4	3	2	1	0
CCED				Reserved			

位	描述	
[31:4]	CCED	控制目前头 ED 指向控制列表里目前端点描述符的物理地址
[3:0]	Reserved	保留.

**HcBulkHeadED 主机控制器块头 ED 寄存器**

寄存器	偏移量	R/W	描述	复位值
HcBulkHeadED	USBH_BA+0x028	R/W	主机控制器块头 ED 寄存器	0x0000_0000

31	30	29	28	27	26	25	24
BHED							
23	22	21	20	19	18	17	16
BHED							
15	14	13	12	11	10	9	8
BHED							
7	6	5	4	3	2	1	0
BHED				Reserved			

位	描述	
[31:4]	<b>BHED</b>	块头 ED 指向块列表第一个端点描述符的物理地址.
[3:0]	<b>Reserved</b>	保留.

**HcBulkCurrentED 主机控制器块目前 ED 寄存器**

寄存器	偏移量	R/W	描述	复位值
HcBulkCurrentED	USBH_BA+0x02C	R/W	主机控制器块目前 ED 寄存器	0x0000_0000

31	30	29	28	27	26	25	24
BCED							
23	22	21	20	19	18	17	16
BCED							
15	14	13	12	11	10	9	8
BCED							
7	6	5	4	3	2	1	0
BCED				Reserved			

位	描述	
[31:4]	BCED	块目前 ED 指向块列表目前端点描述符的物理地址
[3:0]	Reserved	保留.

**HcDoneHead 主机控制器完成头寄存器**

寄存器	偏移量	R/W	描述	复位值
HcDoneHead	USBH_BA+0x030	R/W	主机控制器完成头寄存器	0x0000_0000

31	30	29	28	27	26	25	24
DH							
23	22	21	20	19	18	17	16
DH							
15	14	13	12	11	10	9	8
DH							
7	6	5	4	3	2	1	0
DH				Reserved			

位	描述	
[31:4]	DH	完成头 指向刚刚传输完成加入完成队列的描述符的物理地址.
[3:0]	Reserved	保留.

**HcFmInterval** 主机控制器帧间隔寄存器

寄存器	偏移量	R/W	描述	复位值
HcFmInterval	USBH_BA+0x034	R/W	主机控制器帧间隔寄存器	0x0000_2EDF

31	30	29	28	27	26	25	24
FIT	<b>FSMPS</b>						
23	22	21	20	19	18	17	16
<b>FSMPS</b>							
15	14	13	12	11	10	9	8
<b>Reserved</b>		<b>FI</b>					
7	6	5	4	3	2	1	0
<b>FI</b>							

位	描述	
[31]	<b>FIT</b>	<b>帧间隔交替</b> 当载入新的值到FI (HcFmInterval[13:0]), 主机控制器驱动会反转此位 0 = 主机控制器没有载入新的值到 FI (HcFmInterval[13:0]). 1 = 主机控制器载入新的值到FI (HcFmInterval[13:0]).
[30:16]	<b>FSMPS</b>	<b>FS 最大数据包</b> 这里的值在每一帧的开始载入最大数据包计数器.
[15:14]	<b>Reserved</b>	保留.
[13:0]	<b>FI</b>	<b>帧间隔</b> 这里设定帧时间长度 (bit times - 1). 例如一个帧 12,000 bit times, 这里的值是11,999.

HcFmRemaining 主机控制器帧剩余寄存器

寄存器	偏移量	R/W	描述	复位值
HcFmRemaining	USBH_BA+0x038	R	主机控制器帧剩余寄存器	0x0000_0000

31	30	29	28	27	26	25	24
FRT	Reserved						
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved		FR					
7	6	5	4	3	2	1	0
FR							

位	描述	
[31]	<b>FRT</b>	<b>帧剩余反转</b> 这一位当FR (HcFmRemaining[13:0])到0之后从FIT (HcFmInterval[31]) 载入.
[30:14]	<b>Reserved</b>	保留.
[13:0]	<b>FR</b>	<b>帧剩余</b> 当主机控制器在 USBOPERATIONAL 状态, 这个14位递减的计数器在每个12MHz的时钟周期递减。当计数器的值减到0 (帧结束), 计数器载入帧间隔, 另外, 计数器也在主机控制器进入USBOPERATIONAL状态时载入

**HcFmNumber 主机控制器帧编号寄存器**

寄存器	偏移量	R/W	描述	复位值
HcFmNumber	USBH_BA+0x03C	R	主机控制器帧编号寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
FN							
7	6	5	4	3	2	1	0
FN							

位	描述	
[31:16]	Reserved	保留
[15:0]	FN	帧编号 16位上数计数器在每次重载FR (HcFmRemaining[13:0])时加1，溢出时由'FFFFh' 变为'0h.'

**HcPeriodicStart 主机控制器周期开始寄存器**

寄存器	偏移量	R/W	描述	复位值
HcPeriodicStart	USBH_BA+0x040	R/W	主机控制器周期开始寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved		PS					
7	6	5	4	3	2	1	0
PS							

位	描述	
[31:14]	Reserved	保留
[13:0]	PS	<b>周期开始</b> 这里的值被列表处理器用来决定周期列表处理需要在帧的哪里开始.

**HcLSThreshold 主控制器低速阈值寄存器**

寄存器	偏移量	R/W	描述	复位值
HcLSThreshold	USBH_BA+0x044	R/W	主控制器低速阈值寄存器	0x0000_0628

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved				LST			
7	6	5	4	3	2	1	0
LST							

位	描述	
[31:12]	Reserved	保留.
[11:0]	LST	低速阈值 在建立低速传输前比较这里与FR (HcFmRemaining[13:0]) 的值。 传输只会在 FR (HcFmRemaining[13:0]) >= 这里的值的时候开始. 这个值由主机控制器驱动考虑传输条件设定并预先初始化。.

**HcRhDescriptorA 主机控制器根 Hub 描述符 A 寄存器**

寄存器	偏移量	R/W	描述	复位值
HcRhDescriptorA	USBH_BA+0x048	R/W	主机控制器根 Hub 描述符 A 寄存器	0x0000_0902

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved			NOCP	OCPM	Reserved		PSM
7	6	5	4	3	2	1	0
NDP							

位	描述	
[31:13]	Reserved	保留.
[12]	NOCP	<b>没有过流保护</b> 这一位描述根Hub端口报告的过流状态 0 = 报告过流状态. 1 = 没有报告过流状态.
[11]	OCPM	<b>过流保护模式</b> 这一位描述根Hub端口报告的过流状态，这一位只在NOCP (HcRhDescriptorA[12]) 为0时有效。 0 = 整体过流. 1 = 单独过流.
[10:9]	Reserved	保留.
[8]	PSM	<b>电源切换模式</b> 这一位决定根Hub端口如何控制电源切换 0 = 整体切换 1 = 单独切换
[7:0]	NDP	<b>下游端口数量</b> USB 主机控制器支持两个下游端口，本系列芯片支持1个

**HcRhDescriptorB 主机控制器根 Hub 描述符 B 寄存器**

寄存器	偏移量	R/W	描述	复位值
HcRhDescriptorB	USBH_BA+0x04C	R/W	主机控制器根 Hub 描述符 B 寄存器	0x0000_0000

31	30	29	28	27	26	25	24
PPCM							
23	22	21	20	19	18	17	16
PPCM							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved							

位	描述	
[31:16]	PPCM	<p>端口电源控制屏蔽 这里仅在PowerSwitchingMode 为1 (单独端口切换)时有效。当为1, 端口只响应独立端口电源切换指令 (Set/ClearPortPower)。当为0, 端口只响应整体电源切换指令 (Set/ClearGlobalPower)。 0 = 端口电源由整体电源切换控制。 1 = 端口电源由单独电源切换控制 注意: PPCM[15:2] 以及 PPCM[0] 保留</p>
[15:0]	Reserved	保留.

**HcRhStatus 主机控制器根 Hub 状态寄存器**

寄存器	偏移量	R/W	描述	复位值
HcRhStatus	USBH_BA+0x050	R/W	主机控制器根Hub状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
CRWE	Reserved						
23	22	21	20	19	18	17	16
Reserved							OCIC
15	14	13	12	11	10	9	8
DRWE	Reserved						
7	6	5	4	3	2	1	0
Reserved							OCI
Reserved							LPS

位	描述	
[31]	CRWE	<b>清除远端唤醒使能位</b> 这一位用来清除 DRWE (HcRhStatus[15]). 这一位永远读到0. 写操作: 0 = 没影响. 1 = 清除 DRWE (HcRhStatus[15]). 
[31:18]	Reserved	保留.
[17]	OCIC	<b>过流指示改变</b> 这一位由硬件当OCI (HcRhStatus[1])改变时设置 此位写1清0. 0 = OCI (HcRhStatus[1]) 没有改变. 1 = OCI (HcRhStatus[1]) 改变了.
[16]	LPSC	<b>设置整体电源</b> 在整体电源模式 (PSM (HcRhDescriptorA[8]) = 0),这一位写1使能所有端口的电源 这一位总是读为0. 写操作: 0 = 没影响. 1 = 设置整体电源.

[15]	<b>DRWE</b>	<b>设备远程唤醒使能位</b> 这一位控制在远端唤醒事件时端口连接状态是否改变 写操作: 0 = 没影响. 1 = 使能在远端唤醒事件时端口连接状态改变. 读操作: 0 = 在远端唤醒事件时端口连接状态改变已禁止. 1 = 在远端唤醒事件时端口连接状态改变已使能.
[14:2]	<b>Reserved</b>	保留.
[1]	<b>OCI</b>	<b>过流指示</b> 这一位反映过流状态管脚的状态，仅在 NOCP (HcRhDesA[12]) 和 OCPM (HcRhDesA[11]) 为0时有效 0 = 没有过流状态. 1 = 过流状态.
[0]	<b>LPS</b>	<b>清除整体电源</b> 在整体电源模式 (PSM (HcRhDescriptorA[8]) = 0), 这一位写1清除所有端口的电源 这一位读总是为0. 写操作: 0 = 没影响. 1 = 清除整体电源

HcRhPrt [1] 主机控制器根 Hub 端口状态

寄存器	偏移量	R/W	描述	复位值
HcRhPortStatus1	USBH_BA+0x058	R/W	主机控制器根 Hub 端口状态 [1]	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved			PRSC	OCIC	PSSC	PESC	CSC
15	14	13	12	11	10	9	8
Reserved						LSDA	PPS
7	6	5	4	3	2	1	0
Reserved			PRS	POCI	PSS	PES	CCS

位	描述	
[31:21]	Reserved	保留.
[20]	PRSC	<p>端口复位状态改变 这一位指示端口复位信号已经完成. 此位写1清0. 0 = 端口复位没有完成. 1 = 端口复位已完成.</p>
[19]	OCIC	<p>端口过流指示改变 T这一位当 POCI (HcRhPortStatus1[3]) 改变时置1. 此位写1清0. 0 = POCI (HcRhPortStatus1[3]) 没有改变. 1 = POCI (HcRhPortStatus1[3]) 改变了.</p>
[18]	PSSC	<p>端口挂起状态改变 这一位指示端口的选择恢复流程完成 此位写1清0 0 = 端口恢复没有完成. 1 = 端口恢复已完成.</p>
[17]	PESC	<p>端口使能状态改变 这一位指示因为硬件事件端口被禁止了(PES (HcRhPortStatus1[1]) 为0). 此位写1清0. 0 = PES (HcRhPortStatus1[1]) 没有改变. 1 = PES (HcRhPortStatus1[1]) 已改变.</p>

[16]	<b>CSC</b>	<b>连接状态改变</b> 这一位指示检测到连接或断开事件(CCSC (HcRhPortStatus1[0]) 改变). 此位写1清0. 0 = 没有连接或断开事件 (CCSC (HcRhPortStatus1[0]) 没有改变). 1 = 硬件检测到连接或断开事件 (CCSC (HcRhPortStatus1[0]) 改变了).
[15:10]	<b>Reserved</b>	保留.
[9]	<b>LSDA</b>	<b>低速设备 (读) 或清除端点电源 (写)</b> 这一位决定插入设备的速度 (和 bus空闲)。 这一位仅在 CCS (HcRhPortStatus1[0]) 为1时有效. 这一位也用来清除端口电源 写操作: 0 = 没影响. 1 = 清除 PPS (HcRhPortStatus1[8]). 读操作: 0 = 全速设备. 1 = 低速设备.
[8]	<b>PPS</b>	<b>端口电源状态</b> 这一位反映端口电源状态，不管电源切换模式 写操作: 0 = 没影响. 1 = 端口电源使能. 读操作: 0 = 端口电源禁止. 1 = 端口电源使能.
[7:5]	<b>Reserved</b>	保留.
[4]	<b>PRS</b>	<b>端口复位状态</b> 这一位反映端口的复位状态. 写操作: 0 = 没影响. 1 = 设置端口复位. 读操作 0 = 端口复位信号无效. 1 = 端口复位信号有效.
[3]	<b>POCI</b>	<b>端口过流指示 (读) 或清除端口挂起 (写)</b> 这一位反映端口过流状态管脚的状态，这里仅当 NOCP (HcRhDescriptorA[12]) 为0且 OCPM (HcRhDescriptorA[11]) 为1时有效. 这一位也用来建立端口的选择结果流程 写操作: 0 = 没影响. 1 = 清除端口挂起. 读操作: 0 = 没有过流状态. 1 = 有过流状态.

[2]	PSS	端口挂起状态 此位指示端口已挂起 写操作: 0 = 没影响. 1 = 设置端口挂起. 都操作: 0 = 端口没有挂起. 1 = 端口选择挂起.
[1]	PES	端口使能状态 写操作: 0 = 没影响. 1 = 设置端口使能. 读操作: 0 = 端口禁止 1 = 端口使能.
[0]	CCS	目前连接状态 (读)或清除端口使能位(写) 写操作: 0 = 没影响. 1 = 清除端口使能. 读操作: 0 = 没有设备连接. 1 = 设备已连接.

**HcPhyControl 主机控制器 PHY 控制寄存器**

寄存器	偏移量	R/W	描述	复位值
HcPhyControl	USBH_BA+0x200	R/W	主机控制器 PHY 控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved				STBYEN	Reserved		
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved							

位	描述	
[31:28]	Reserved	保留.
[27]	STBYEN	<b>USB 收发器待命使能位</b> 这意味控制USB 收发器是否进入待命模式以降低电源功耗 0 = USB 收发器不会进入待命模式. 1 = USB 收发器在端口掉电状态（端口电源无效）会进入待命模式
[26:0]	Reserved	保留.

**HcMiscControl 主机控制器杂项控制寄存器**

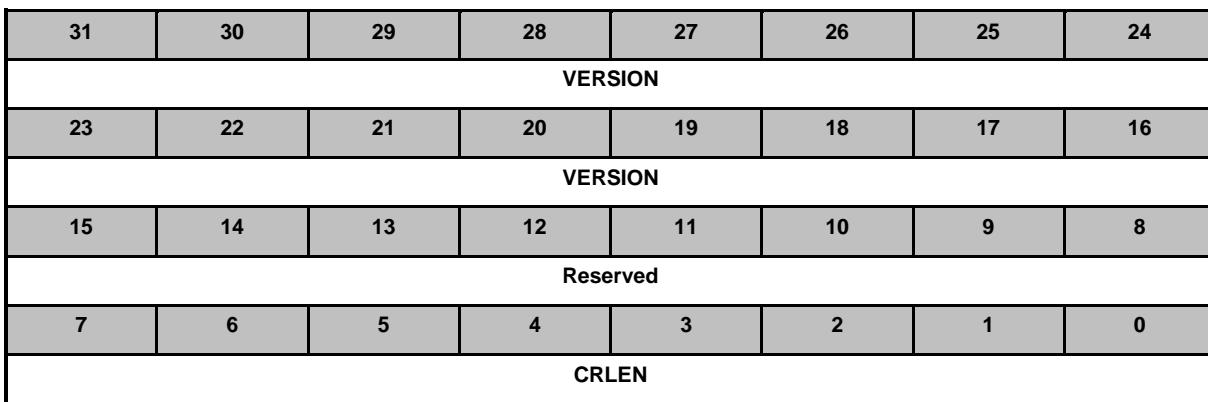
寄存器	偏移量	R/W	描述	复位值
HcMiscControl	USBH_BA+0x204	R/W	主机控制器杂项控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved				OCAL	Reserved	ABORT	Reserved

位	描述	
[31:17]	<b>Reserved</b>	保留.
[16]	<b>DPRT1</b>	<p><b>禁止端口 1</b>            这一位控制是否禁止端口1的USB 主机控制器与收发器之间的连接。如果连接被禁止，USB 主机控制器不会确认USB总线事件。            此位置1，不管USB主控制如何工作，端口1的收发器也会强制进入待命模式。            0 = 端口1的USB 主机控制器与收发器之间的连接使能。            1 = 端口1的USB 主机控制器与收发器之间的连接禁止，端口1的收发器也会强制进入待命模式。</p>
[15:4]	<b>Reserved</b>	保留.
[3]	<b>OCAL</b>	<p><b>过流低有效</b>            这里设定外部电源IC过流标识的极性.            0 = 过流标志高有效.            1 = 过流标志低有效.</p>
[2]	<b>Reserved</b>	保留.
[1]	<b>ABORT</b>	<p><b>AHB 总线错误响应</b>            这一位指示AHB 总线上收到一个错误响应            0 = 没有收到错误响应.            1 = 收到错误响应.</p>
[0]	<b>Reserved</b>	保留.

**EHCVNR EHCI 版本号寄存器**

寄存器	偏移量	R/W	描述	复位值
EHCVNR	HSUSBH_BA+0x00	R	EHCI 版本号寄存器	0x0095_0020



位	描述	
[31:16]	<b>VERSION</b>	主控制器接口版本号 这里两个字节组合的BCD码是主机控制器支持的 EHCI 版本号。高字节是主版本，低字节是小版本。.
[15:8]	<b>Reserved</b>	保留。
[7:0]	<b>CRLEN</b>	性能寄存器长度 这个寄存器的值作为一个偏移量加到寄存器地址得到操作寄存器空间的开始地址。

**EHCSPR EHCI 结构参数寄存器**

寄存器	偏移量	R/W	描述	复位值
EHCSPR	HSUSBH_BA+0x004	R	EHCI 结构参数寄存器	0x0000_0012

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
<b>N_CC</b>				<b>N_PCC</b>			
7	6	5	4	3	2	1	0
Reserved			PPC	<b>N_PORTS</b>			

位	描述	
[31:16]	<b>Reserved</b>	保留.
[15:12]	<b>N_CC</b>	<p><b>配套控制器个数</b>            这里指定 USB 2.0 主控制器的配套控制器的个数            这里为0指示没有配有配套的主机控制器。不支持端口所有权变更，主机控制器根端口只支持高速设备。            大于0的值表示有配套的 USB 1.1 主机控制器(s)。支持端口所有权变更。主控制器根端口支持高速，全速和低速设备。</p>
[11:8]	<b>N_PCC</b>	<p><b>每个配套处理器的端口数</b>            这里设定每个配套处理器支持的端口数，用来决定系统程序端口出线配置            例如，如果 N_PORTS 值是6 而 N_CC 的值是 2，那么 N_PCC 的值可以是3。约定是前3个 N_PCC 端口被设置连接配套处理器1，接下来的N_PCC 端口连接配套处理器2，在前一个例子中，N_PCC 可以是 4，前 4 个连接到配套处理器1，后2个连接到配套处理器2。            这里的值必须要与 N_PORTS 以及 N_CC一致。</p>
[7:5]	<b>Reserved</b>	保留.
[4]	<b>PPC</b>	<p><b>端口电源控制</b>            这里决定是否主控制器贯彻端口电源控制。这里是1表明端口有端口电源切换，是0表示端口没有端口电源切换。这里的值影响到每个端口状态和控制寄存器的功能.</p>
[3:0]	<b>N_PORTS</b>	<p><b>物理下游端口数量</b>            这是指定这个主机控制器实现的物理下行端口数。这里的值决定操作寄存器空间有多少个端口寄存器地址(见表 2-8). 有效值范围是 1H 到 FH.            这里填0未定义.</p>

## EHCCPR EHCI 性能参数寄存器

寄存器	偏移量	R/W	描述	复位值
EHCCPR	HSUSBH_BA+0x008	R	EHCI 性能参数寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
EECP							
7	6	5	4	3	2	1	0
IST				Reserved	ASPC	PFLF	AC64

位	描述	
[31:16]	<b>Reserved</b>	保留
[15:8]	<b>EECP</b>	<b>EHCI 扩展容量指针 (EECP)</b> 0 = 不支持扩展容量.
[7:4]	<b>IST</b>	<b>同步列表阈值</b> 软件依据这里的值, 以及主机控制器目前的执行位置, 可以可靠的更新同步列表 当 bit [7] 是0, 最低3位指示微帧数量, 主控制器可以在清除状态前持有一套同步数据结构 (一个或多个)。
[3]	<b>Reserved</b>	保留.
[2]	<b>ASPC</b>	<b>同步列表栈能力</b> 0 = EHCI 主机控制器不支持同步列表高速队列头栈特性.
[1]	<b>PFLF</b>	<b>可编程帧列表标志</b> 0 = EHCI 主机控制器里, 系统软件必须使用1024个元素的帧列表
[0]	<b>AC64</b>	<b>64位寻址能力</b> 0 = 数据结构使用 32位地址内存指针.

**UCMDR USB 指令寄存器**

寄存器	偏移量	R/W	描述	复位值
UCMDR	HSUSBH_BA+0x020	R/W	USB 指令寄存器	0x0008_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
ITC							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved	IAAD	ASEN	PSEN	FLSZ	HCRST	RUN	

位	描述	
[31:24]	Reserved	保留.
[23:16]	ITC	<p><b>中断阈值控制(读写)</b>            系统软件用这里来选择主机控制器执行中断的最大频率。有效的值在下表列出。如果软件写入一个未确定的值，结果不可知</p> <p>最大中断间隔：</p> <ul style="list-style-type: none"> <li>0x00 = 保留.</li> <li>0x01 = 1 微帧.</li> <li>0x02 = 2微帧.</li> <li>0x04 = 4微帧..</li> <li>0x08 = 8微帧. (默认, 相当于 1 ms).</li> <li>0x10 = 16微帧. (2 ms).</li> <li>0x20 = 32微帧. (4 ms).</li> <li>0x40 = 64微帧. (8 ms).</li> </ul> <p>这个寄存器填入其他值结果未定义.</p> <p>程序当HCHalted 为0时修改这一位结果不可知.</p>
[15:7]	Reserved	保留.
[6]	IAAD	<p><b>同步增强型门铃中断(读写)</b>            这一位被程序用作门铃去告诉主机控制器执行中断，下一次同步计划表递进。程序写1接响门铃。.</p> <p>当主机控制器已经驱逐所有相关的暂存的计划表状态，设定USBSTS 寄存器同步递进状态中断位为1，主机控制器会在下个中断时隙执行终端。</p> <p>设定USBSTS 寄存器同步递进状态中断位为1后主机控制器设置这位为0。</p> <p>程序不能在禁止同步计划表时写这里为1，那样做会导致不可预知的后果。</p>

[5]	<b>ASEN</b>	<b>同步计划表使能 (读写)</b> 这一位控制主机控制器是否跳过执行同步计划表: 0 = 不要执行同步计划表. 1 = 使用 ASYNCLISTADDR 寄存器执行同步计划表.
[4]	<b>PSEN</b>	<b>周期计划表使能 (读写)</b> 这一位控制主机控制器是否跳过执行周期计划表: 0 = 不执行周期计划表. 1 = 使用 PERIODICLISTBASE 寄存器执行周期计划表.
[3:2]	<b>FLSZ</b>	<b>帧列表大小 (读写或只读)</b> 这里仅在HCCPARAMS 寄存器的可编程帧列表标志为1时可读写。这里决定帧列表的大小。帧列表大小控制帧索引寄存器的哪些位用作目前的帧列表索引: 00 = 1024 个元素 (4096字节) 默认值. 01 = 512 元素 (2048 字节). 10 = 256 元素 (1024 字节) – 资源紧张的环境下. 11 = 保留.
[1]	<b>HCRST</b>	<b>主机控制器复位 (HCRESET) (读写)</b> 这个控制位是主机控制器的软件复位。这一位对于根Hub寄存器的效果与芯片硬件复位类似当软件向此位写1，主机控制器复位内部流水线，定时器，计数器，状态机等到它们的默认值。任何USB正在执行的流程会被终止，USB 复位不会去复位下游端口。 所有的操作寄存器，包括端口寄存器和端口状态机会回复初始化的值。端口所有权会更改为配套控制器。软件需要重新初始化主机控制器来回到操作状态。 这一位当复位流程结束后会由硬件自动清0，程序不要写0打断复位流程。 程序不能在USBSTS 寄存.
[0]	<b>RUN</b>	<b>运行/停止 (读写)</b> 写1，主机控制器执行计划表，主机控制器会在此位为1时一直执行。当此位为0，主机控制器完成目前的以及所有有效的流水线的USB传输后挂起。主机控制器在此位清0后需要在16个微帧的时间内挂起。当主机控制器完成流水线传输进入停止态后状态寄存器HC Halted位置1.程序不要在主机控制器在挂起状态（USBSTS寄存器的HCHalted 位为1）时向此位写1，否则将不可预知后果。 0 = 停止. 1 =运行.

**USTSR USB 状态寄存器**

寄存器	偏移量	R/W	描述	复位值
USTSR	HSUSBH_BA+0x024	R/W	USB 状态寄存器	0x0000_1000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
ASS	PSS	RECLA	HCHalted	Reserved			
7	6	5	4	3	2	1	0
Reserved		IAA	HSERR	FLR	PCD	UERRINT	USBINT

位	描述	
[31:16]	Reserved	保留.
[15]	ASS	<b>同步计划表状态 (只读)</b> 这一位报告目前同步计划表的状态，如果这一位是0，那么同步计划表禁止。如果这一位是1，那么同步计划表使能。.主机控制器并不需要在程序修改USBCMD寄存器的同步计划表使能位后立即禁止或使能。当这里和同步计划表使能位是相同的值，同步计划表是使能或禁止。
[14]	PSS	<b>周期计划表状态 (只读)</b> 这一位报告目前周期计划表的状态，如果这一位是0，那么周期计划表禁止。如果这一位是1，那么周期计划表使能。.主机控制器并不需要在程序修改USBCMD寄存器的周期计划表使能位后立即禁止或使能。当这里和周期计划表使能位是相同的值，周期计划表是使能或禁止。
[13]	RECLA	<b>开拓 (只读)</b> 这是只读的状态位，用来检测同步计划表空.
[12]	HCHalted	<b>HCHalted (只读)</b> 这一位当 运行/停止 位是1时为0.主机控制器当软件或硬件（例如内部错误）将运行/停止 位设为0后将此位置1
[11:6]	Reserved	保留.
[5]	IAA	<b>同步递进中断 (R/W)</b> 系统程序可以向USBCMD寄存器的同步递进门铃位写1强制主机控制器执行下一个同步计划表，这一位是中断源的状态位。.
[4]	HSERR	<b>主机系统错误 (R/W)</b> 主机控制器当主机系统访问主机控制模块发生严重错误时将此位置1

[3]	<b>FLR</b>	<b>帧列表轮转 (R/WC)</b> 主机控制器当帧列表索引从最大值轮转到0时将此位置1。轮转发生的值取决于帧列表大小。例如, 如果帧列表大小 (USBCMD 寄存器的帧列表大小域) 是 1024, the帧索引寄存器每次在 FRINDEX[13] 反转时回转. 同样的, 如果大小是512, 主机控制器每次在 FRINDEX[12] 反转时将此位置1.
[2]	<b>PCD</b>	<b>端口改变检测(R/WC)</b> 主机控制器当任何端口的端口所有者位设为0, 位传输从0变1, 或者强制端口恢复位从0变1导致挂起端口检测到J-K 跳变时将此位置1。这一位也在系统软件向端口的的端口所有者位写1让出所有权导致连接状态改变时设为1。 这一位运行在辅助电源正常时保持, 或者, 也接受EHCI 主机控制器设备D3 到 D0 转变 这一位又OR 载入所有的PORTSC改变位 (包括: 强制端口恢复, 过流改变, 使能禁止改变和连接状态改变).
[1]	<b>UERRINT</b>	<b>USB 错误中断 (USBERRINT) (R/WC)</b> 主机控制器当完成的USB事物结果是错误状况 (例如计数器下溢错误)时置1, 如果没有错误中断发生的 TD 的 IOC位为1, 这一位和USBINT 位都会置1.
[0]	<b>USBINT</b>	<b>USB 中断 (USBINT) (R/WC)</b> 主机控制器当完成USB事物后将此位置1, 隐退的传输描述符的IOC位会置1. 主机控制器也当检测到短包时将此位置1 (实际字节数少于期望字节数).

**UIENR USB 中断使能寄存器**

寄存器	偏移量	R/W	描述	复位值
UIENR	HSUSBH_BA+0x028	R/W	USB 中断使能寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved		IAAEN	HSERREN	FLREN	PCIEN	UERRIEN	USBIEN

位	描述
[31:6]	<b>Reserved</b>  同步递进中断使能抑制位 当此位置1，且USBSTS 寄存器的同步递进中断位是1，主机控制器会在下一个中断入口执行中断。中断在软件清除同步递进中断位后确认 0 = 禁止同步递进中断。 1 = 使能同步递进中断
[5]	<b>IAAEN</b>  主机控制器错误使能禁止位 当此位为1，并且USBSTS 寄存器的主机系统错误状态是1，主机控制器会执行一个中断。中断当软件清除主机错误中断位后确认 0 = 主机系统错误中断禁止。 1 = 主机系统错误中断使能
[4]	<b>HSERREN</b>  帧列表回转使能禁止位 当此位为1，且USBSTS寄存器的帧列表回转位为1，主机控制器会执行一个中断。中断当软件清除帧列表回转位后确认。 0 = 帧列表回转中断禁止。 1 = 帧列表回转中断使能。
[3]	<b>FLREN</b>  端口改变中断使能禁止位 当此位为1，且USBSTS寄存器的端口改变检测位是1，主机控制器会执行一个中断。中断当软件清除端口中断检测位后确认。 0 = 端口改变中断禁止。 1 = 端口改变中断使能。
[2]	<b>PCIEN</b>  同步递进中断使能抑制位 当此位置1，且USBSTS 寄存器的同步递进中断位是1，主机控制器会在下一个中断入口执行中断。中断在软件清除同步递进中断位后确认 0 = 禁止同步递进中断。 1 = 使能同步递进中断

[1]	<b>UERRIEN</b>	<b>USB 错误中断使能禁止位</b> 当此位为1, 且USBSTS寄存器的USBERRINT 位为1, 主机控制器会在下一个中断入口执行一个中断。中断当软件清除 USBERRINT 位后确认。 0 = USB 错误中断禁止. 1 = USB 错误中断使能.
[0]	<b>USBIEN</b>	<b>USB 中断使能禁止位</b> 当此位为1, 且USBSTS 的USBINT位为1, 主机控制器会在下一个中断入口执行一个中断, 中断在软件清除USBINT位后确认。 0 = USB 中断禁止. 1 = USB 中断使能.

UFINDR USB 帧索引寄存器

寄存器	偏移量	R/W	描述	复位值
UFINDR	HSUSBH_BA+0x02C	R/W	USB 帧索引寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved		FI					
7	6	5	4	3	2	1	0
FI							

位	描述																
[31:14]	Reserved	保留.															
[13:0]	FI	<p><b>帧索引</b>            这里的值在每个时间帧（如微帧）结束时加1， Bits [N:3] 用做帧列表目前索引。这意味着帧列表的每个位置在移到下一个索引前会访问8个时间周期（帧或微帧）。下表是根据USBCMD 帧列表大小的对应值。</p> <table> <thead> <tr> <th>FLSZ (UCMDR[3:2])</th> <th>Number Elements</th> <th>N</th> </tr> </thead> <tbody> <tr> <td>0x0</td> <td>1024</td> <td>12</td> </tr> <tr> <td>0x1</td> <td>512</td> <td>11</td> </tr> <tr> <td>0x2</td> <td>256</td> <td>10</td> </tr> <tr> <td>0x3</td> <td>保留</td> <td></td> </tr> </tbody> </table>	FLSZ (UCMDR[3:2])	Number Elements	N	0x0	1024	12	0x1	512	11	0x2	256	10	0x3	保留	
FLSZ (UCMDR[3:2])	Number Elements	N															
0x0	1024	12															
0x1	512	11															
0x2	256	10															
0x3	保留																

**UPFLBAR USB 周期帧列表基地址寄存器**

寄存器	偏移量	R/W	描述	复位值
UPFLBAR	HSUSBH_BA+0x034	R/W	USB 周期帧列表基地址寄存器	0x0000_0000

31	30	29	28	27	26	25	24
BADDR							
23	22	21	20	19	18	17	16
BADDR							
15	14	13	12	11	10	9	8
BADDR				Reserved			
7	6	5	4	3	2	1	0
Reserved							

位	描述	
[31:12]	BADDR	基地址 这些位对应内存地址信号 [31:12]
[11:0]	Reserved	保留.

UCALAR USB 目前同步列表地址寄存器

寄存器	偏移量	R/W	描述	复位值
UCALAR	HSUSBH_BA+0x038	R/W	USB 目前同步列表地址寄存器	0x0000_0000

31	30	29	28	27	26	25	24
LPL							
23	22	21	20	19	18	17	16
LPL							
15	14	13	12	11	10	9	8
LPL							
7	6	5	4	3	2	1	0
LPL			Reserved				

位	描述	
[31:5]	LPL	链接低指针 (LPL) 这些位对应与内存地址信号[31:5], 只被 Queue Head (QH)参考.
[4:0]	Reserved	保留.

**UASSTR USB 异步计划表休眠定时器寄存器**

寄存器	偏移量	R/W	描述	复位值
UASSTR	HSUSBH_BA+0x03C	R/W	USB 异步计划表休眠定时器寄存器	0x0000_0BD6

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved				ASSTMR			
7	6	5	4	3	2	1	0
ASSTMR							

位	描述	
[31:11]	<b>Reserved</b>	保留
[11:0]	<b>ASSTMR</b>	<p><b>异步计划表休眠定时器</b>          这里决定EHCI AsyncSchedSleep时间。          这里用来控制当异步计划表为空。主机控制器多久去系统内存搬运异步计划表          这个定时器的默认值是 12'hBD6. 因为这个定时器用 UTMI 时钟 (30MHz) 休眠时间差不多 100us.</p>

**UCFGR USB 配置标识寄存器**

寄存器	偏移量	R/W	描述	复位值
UCFGR	HSUSBH_BA+0x060	R/W	USB 配置标志寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved							CF

位	描述	
[31:1]	Reserved	保留
[0]	CF	<b>配置标识 (CF)</b> 主机程序设置此位作为配置主机定时器的最后行动。这一位控制默认端口出线控制逻辑。 0 = 每个端口出线逻辑默认贯彻标准控制器默认逻辑。 1 = P每个端口出线逻辑默认贯彻此控制器默认逻辑

## UPSCR USB 端口状态和控制寄存器

寄存器	偏移量	R/W	描述	复位值
UPSCR0	HSUSBH_BA+0x064	R/W	USB 端口 0 端口和控制寄存器	0x0000_2000
UPSCR1	HSUSBH_BA+0x068	R/W	USB 端口 1 端口和控制寄存器	0x0000_2000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved				PTC			
15	14	13	12	11	10	9	8
Reserved		PO	PP	LSTS		Reserved	PRST
7	6	5	4	3	2	1	0
SUSPEND	FPR	OCC	OCA	PEC	PE	CSC	CCS

位	描述	
[31:20]	Reserved	保留.
[19:16]	PTC	<p><b>端口测试控制 (读写)</b></p> <p>当这里是0，端口不会工作在测试模式。非零值表示工作特定的测试模式，测试模式编码如下(0x6 ~ 0xF 保留):</p> <p>测试模式位</p> <p>0x0 = 测试 模式关闭      0x1 = 测试J_STATE.      0x2 = 测试K_STATE.      0x3 = 测试SE0_NAK.      0x4 = 测试包.      0x5 = 测试FORCE_ENABLE.</p>
[15:14]	Reserved	保留.
[13]	PO	<p><b>端口所有者 (读写)</b></p> <p>当CONFIGFLAG 寄存器的控制位由0变1时，这一位无条件的变为0b。当控制位为1时，此位无条件变1..</p> <p>系统软件用这一位来释放端口所有权来选择主机控制器（如果插入的设备不是高速设备）当插入的不是高速设备时程序向这里写1，这里是1意味着配套控制器控制这个端口</p>
[12]	PP	<p><b>端口电源 (PP)</b></p> <p>主机控制器有电源控制切换，这一位反映目前的开关设定（0关闭，1打开），当端口没有有效电源（PP为0），端口无法工作，且不会报告插入，拔出等。.</p> <p>当供电端口检测到过流状况，且PPC 是1，主机控制器会设置 PP位由1变0（关闭端口电源）</p>

[11:10]	<b>LSTS</b>	<p><b>线路状态(R0)</b></p> <p>这里反映目前信号线 D+ (bit 11) 与 D- (bit 10) 信号线的逻辑电平。这两位用来在端口复位前检测低速设备以及使能序列。这里仅在使能位为0且目前连接状态位是1时有效。</p> <p>位编码如下：</p> <p>Bits[11:10] USB 状态翻译</p> <p>00 = SEO 不是低速设备, 执行 EHCI 复位.</p> <p>01 = K-state 低速设备, 释放端口所有权.</p> <p>10 = J-state 不是低速设备, 执行 EHCI 复位.</p> <p>11 = 未定义的不是低速设备, 执行 EHCI 复位.</p> <p>在端口电源是0, 这里的值没有定义</p>
[9]	<b>Reserved</b>	保留.
[8]	<b>PRST</b>	<p><b>端口复位(读写)</b></p> <p>当软件向这里写1 (之前是0), 就开始 USB2.0标准定义的复位序列。程序向这里写0中断复位序列。程序需要保持这一位为1足够的时间来确保 USB2.0协议要求的复位序列能够完成。. 注意: 当程序向这里写1, 也必须想端口使能位写0。</p> <p>注意当程序向这里写0, 到这一位变为0会有一个延时。这一位直到复位完成才会读为0。如果复位完成端口会是高速模式, 主机控制器会自动使能这个端口 (设置端口使能位为1). 这一位由1变0, 主机控制器终止复位且保留端口状态2mS。例如: 如果端口在复位时检测到插入设备是高速设备, 那么程序向这一位写0, 主机控制器保证端口在使能状态2 mS</p> <p>在程序尝试使用这一位之前, USBSTS寄存器的 HCHalted 位需要是0。主机控制器在HCHalted为1时保持端口复位。.</p> <p>当端口电源是0时, 此位是0.</p> <p>0 = 端口没有复位.</p> <p>1 = 端口复位.</p>
[7]	<b>SUSPEND</b>	<p><b>挂起(读写)</b></p> <p>这个寄存器的端口使能位和挂起位定义端口状态如下:</p> <p>00 = 端口禁止.</p> <p>01 = 端口禁止.</p> <p>10 = 端口使能.</p> <p>11 = 端口挂起.</p> <p>在挂起态, 下游传播数据在这个端口屏蔽, 除了端口复位。如果传输正在进行时此位被置1, 屏蔽将在传输结束时发生。在挂起态, 端口检测恢复信号。注意这一位的状态直到端口进入挂机态后才会改变, 如果设置此位时有USB事物正在进行, 那么就会有一个延时。</p> <p>向此位写0会被主机控制器忽略, 主机控制器会在下列条件将此位置0:</p> <p>程序设置强制端口恢复位为0 (从1)</p> <p>程序设置端口复位位为1 (从0) .</p> <p>如果设置此位为1, 但是端口没有使能 (端口使能位为0) , 结果未知。</p> <p>当端口电源为0, 此位为0.</p> <p>0 = 端口不在挂起态.</p> <p>1 = 端口在挂起态.</p>

[6]	<b>FPR</b>	<p><b>强制端口恢复 (读写)</b></p> <p>操作这一位要依据Suspend 位的值。例如，如果端口没有挂起（挂起位和使能位都是1），而程序修改此位为1，对总线的影响不可知。.</p> <p>程序设置此位为1驱动恢复信号。当端口在挂起态，主机控制器检测到J-to-K 跳变时设置此位为1。当是因为检测到J-to-K跳变设置此位是1，USBSTS寄存器的端口改变检测位也会也会置1。如果是程序设置此位为1，主机控制器不会设置端口改变检测位。.</p> <p>注意当EHCI 控制器拥有端口，回复序列依照 USB 2.0协议规定。在此位为1时端口持续产生恢复信号(全速 'K')。程序需要评估恢复信号的时间然后将此位置0。将此位从1写0是端口恢复到高速模式（强制总线下游端口进入高速空闲）这一位保持为1直到端口切换到高速空闲。在软件将此位写0后主机控制器需要在2mS内完成这些。</p> <p>当端口电源是0此位是0.</p> <p>0 = 端口没有检测到或驱动恢复 (Kstate) .</p> <p>1 = 端口检测到或驱动恢复.</p>
[5]	<b>OCC</b>	<p><b>过流改变 (R/WC)</b></p> <p>1 = 当改变为过流有效时此位为1，程序写1清除此位.</p>
[4]	<b>OCA</b>	<p><b>过流有效 (RO)</b></p> <p>此位在过流状况消除后自动由1变0.</p> <p>0 = 这个端口没有过流状况.</p> <p>1 = 端口有过流状况.</p>
[3]	<b>PEC</b>	<p><b>端口使能/禁止改变 (R/WC)</b></p> <p>对于根hub, 这一位仅在端口因为EOF2点上的相应条件被禁止时置1（见USB规范第11章端口错误定义），程序写1清除此位.</p> <p>如果端口电源是0此位是0.</p> <p>0 = 没有改变.</p> <p>1 = 端口使能禁止状态改变了.</p>
[2]	<b>PE</b>	<p><b>Port 使能/禁止 (R/W)</b></p> <p>端口仅在作为复位与使能的一部分有主机控制器使能。程序不能向这里写1使能端口。主机控制器仅在复位序列中决定插入的设备是高速设备时将此位置1。.</p> <p>端口可以由任意错误条件（未连接事件或其他错误条件）禁止或由程序禁止。注意这一位在端口状态真正改变后才会改变，可能在禁止使能端口时由于其他控制器或总线事件造成延时。</p> <p>当端口被禁止 (0b)，下游数据会在这个端口被屏蔽，除了复位.</p> <p>如果端口电源是0这里是0.</p> <p>0 = 端口禁止.</p> <p>1 = 端口使能.</p>
[1]	<b>CSC</b>	<p><b>连接状态改变 (R/W)</b></p> <p>指示端口目前的连接状态改变了。主机控制器会在任何端口设备连接状态改变时设置此位，即使系统软件没有清除一个存在的连接状态改变。例如，在系统软件清除改变状态前插入状态改变了两次，硬件会设置已经设置的位。程序对此位写1清0。.</p> <p>当端口电源是0这里是0.</p> <p>0 = 没有改变.</p> <p>1 = 目前连接状态改变了.</p>

[0]	CCS	<b>目前连接状态 (只读)</b> 这位反映前端口的状态，不一定直接引起连接状态改变位(Bit 1)为1. 如果端口电源是0此位是0. 0 = 没有设备连接 1 = 端口上有设备
-----	-----	--

**USBPCR0 USB PHY 0 控制寄存器**

寄存器	偏移量	R/W	描述	复位值
USBPCR0	HSUSBH_BA+0x0C4	R/W	USB PHY 0 控制寄存器	0x0000_0060

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved				CLKVALID	Reserved		SUSPEND
7	6	5	4	3	2	1	0
Reserved							

位	描述	
[31:12]	<b>Reserved</b>	保留.
[11]	<b>CLKVALID</b>	<p><b>UTMI 时钟有效</b>            这一位指示从USB 2.0 PHY来的 UTMI时钟有效. 程序在这个标识有效前不要去写控制寄存器.            0 = UTMI 时钟无效.            1 = UTMI 时钟有效.</p>
[10:9]	<b>Reserved</b>	保留.
[8]	<b>SUSPEND</b>	<p><b>挂起声明</b>            这一位控制 USB PHY 0.挂起模式            当 PHY 已挂起, PHY 的所有电路会掉电, 输出三态。            这一位默认是 1'b0 , 意味着 USB PHY 0默认是挂起。需要在配置 USB主机控制器之前设置此位为1'b1 让 USB PHY 0 离开挂起状态。            0 = USB PHY 0 已挂起.            1 = USB PHY 0 没有挂起</p>
[7:0]	<b>Reserved</b>	保留.

**USBPCR1 USB PHY 1 控制寄存器**

寄存器	偏移量	R/W	描述	复位值
USBPCR1	HSUSBH_BA+0x0C8	R/W	USB PHY 1 控制寄存器	0x0000_0020

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

位	描述	
[31:9]	<b>Reserved</b>	保留.
[8]	<b>SUSPEND</b>	<p><b>挂起声明</b></p> <p>这一位控制 USB PHY 1.挂起模式</p> <p>当 PHY 已挂起, PHY 的所有电路会掉电, 输出三态。</p> <p>这一位默认是 1'b0 , 意味着 USB PHY 1默认是挂起。需要在配置 USB主机控制器之前设置此位为1'b1 让 USB PHY 1 离开挂起状态.</p> <p>0 = USB PHY 1 已挂起. 1 = USB PHY 1 没有挂起</p>
[7:0]	<b>Reserved</b>	保留.

## 6.33 USB OTG

### 6.33.1 概述

OTG 控制器连接 USB PHY 和 USB 控制器，包含USB 1.1 主机控制器和 USB 2.0 全速设备控制器。OTG 控制器支持“On-The-Go和集成主机 USB 2.0协议版本2.0 补充规范”定义的HNP 和 SRP 协议

USB 框架，包括 USB 主机，USB 设备，和 OTG 控制器，可以配置为 仅主机，仅设备，依据ID或 USBROLE (SYS\_USBPHY[1:0])定义的OTG 设备模式。在仅主机模式下，USB 框架工作在USB主机。USB 框架都支持全速和低速传输。在仅从机模式，USB 工作在 USB 设备。USB 框架只支持全速传输。在依据 ID模式，USB 框架依据USB\_ID 管脚状态可以工作在 USB 主机和 USB 设备模式。在 OTG 设备模式，USB 框架的角色依据 OTG 标准的定义。USB 框架在OTG 设备作为外设时只支持全速传输。

### 6.33.2 特性

- 内建 USB PHY
- 可配置工作在：
  - 仅主机
  - 仅设备
  - 依据ID: USB框架的角色仅依据USB\_ID 管脚状态—作为 USB 主机 (USB\_ID 管脚是低) 或 USB 设备 (USB\_ID 管脚是高). 不支持 HNP 或 SRP 协议.
  - OTG设备:依据 USB\_ID 管脚状态作为 A-device (USB\_ID 管脚是低) 或 B-device (USB\_ID 管脚是高).支持 HNP 和 SRP 协议.

### 6.33.3 框图

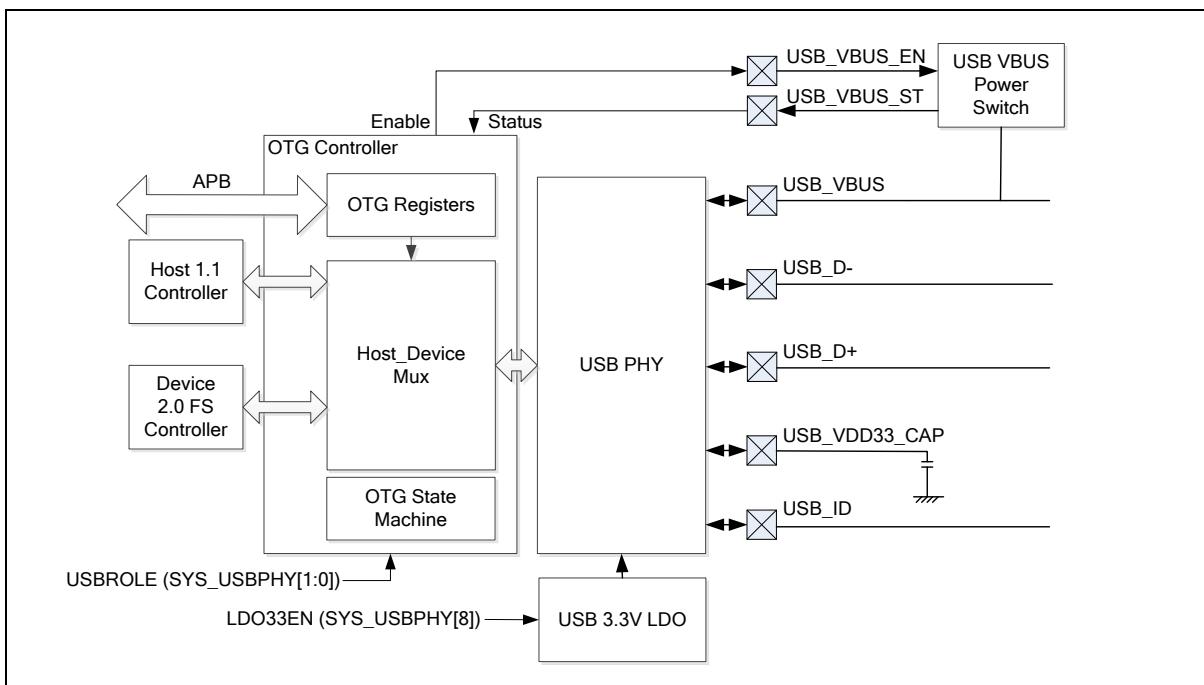


图 6.33-1 USB OTG 框图

### 6.33.4 基本配置

OTG 外设时钟由 OTGCKEN (CLK\_APBCLK0[26]) 使能。USB 框架的角色由 USBROLE (SYS\_USBPHY[1:0]) 设置。内部 USB 3.3V LDO 由 LDO33EN (SYS\_USBPHY[8]) 使能。这两个配置是写保护位，在写这些位之前，用户需要禁用寄存器保护功能。参考 SYS\_REGLCTL 寄存器的描述。USB\_VBUS\_EN 和 USB\_VBUS\_ST 管脚功能由 SYS\_GPA\_MFPL 或 SYS\_GPC\_MFPL 寄存器配置。

### 6.33.5 功能描述

USB 框架的角色依据 USBROLE (SYS\_USBPHY[1:0]) 的设定以及 USB\_ID 管脚状态决定。USBROLE 的配置优先于 USB\_ID 管脚的状态。用户可以配置 OTG 控制器到 USB 主机模式、USB 设备模式、依据 ID 模式和 OTG 设备模式。在 USB 主机模式，主机控制器直接与 USB PHY 互动。在 USB 设备模式，设备控制器直接与 USB PHY 互动。在这些情况下，OTG 控制器作为简单的多路复用器使用。在依据 ID 模式，USB\_ID 管脚的状态决定 USB 帧充当 USB 主机或 USB 设备。如果 USB\_ID 管脚在 FALSE 状态（低电平），USB 框架充当 USB 主机。如果 USB\_ID 管脚在 TRUE 状态（高电平），USB 帧充当 USB 设备。在 OTG 设备模式，OTG 控制器会执行 OTG HNP 和 SRP 协议，如果 USB\_ID 管脚是 FALSE 状态（低电平），OTG 控制器充当 OTG A-设备，如果 USB\_ID 管脚是 TRUE 状态（高电平），OTG 控制器充当 OTG B-设备。

#### 6.33.5.1 USB 框架的角色

##### USB 设备模式

当 USBROLE (SYS\_USBPHY[1:0]) 设为 0，USB 框架充当 USB 设备。USB 主机功能不可用。

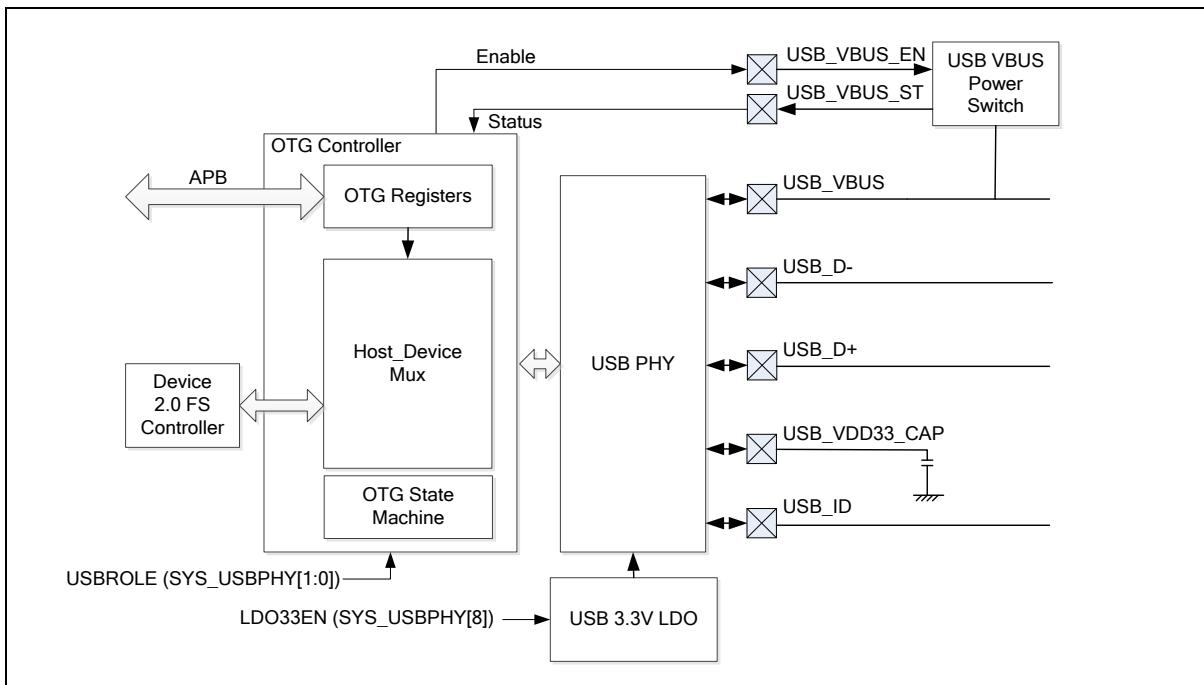


图 6.33-2 USB 设备模式

##### USB 主机模式

当 USBROLE (SYS\_USBPHY[1:0]) 设为 1，USB 框架充当 USB 主机。USB 设备功能不可用。

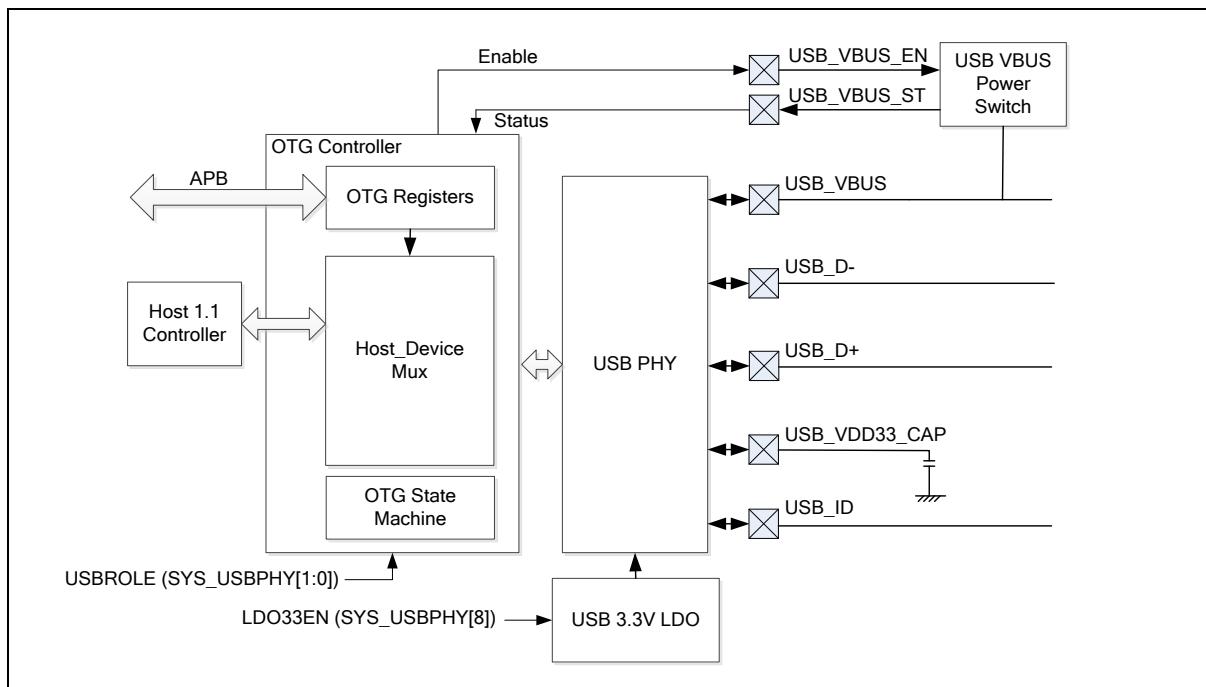


图 6.33-3 USB 主机模式

### 依据ID 模式

当 USBROLE (SYS\_USBPHY[1:0]) 设为 2, USB 框架的角色依据 USB\_ID 管脚的状态。ID 检测功能由 IDDETEN (OTG\_PHYCTL[1]) 使能。USB\_ID 管脚状态影响 IDSTS (OTG\_STATUS[1])。当 USB 框架充当 USB 主机 (USB\_ID 管脚低), 框图与 USB 主机模式相同。当 USB 框架充当 USB 设备 (USB\_ID 管脚是高), 框图与 USB 设备模式相同。

### OTG 设备模式

当 USBROLE (SYS\_USBPHY[1:0]) 等于 3, USB 框架的角色取决于 USB\_ID 管脚的状态。ID 检测功能由 IDDETEN (OTG\_PHYCTL[1]) 使能。USB\_ID 管脚状态影响 IDSTS (OTG\_STATUS[1])。当 USB\_ID 管脚是低电平, OTG 控制器充当 OTG A-device。当 USB\_ID 管脚是高电平, OTG 控制器充当 OTG B-device。请参考 OTG 标准得到详细的 A-device 和 B-device 的行为描述。

#### 6.33.5.2 会话请求协议(SRP)

当 USB 框架配置为 OTG 设备模式, OTG 控制器支持 SRP 来保护电源, 参考 OTG 标准得到 SRP 的详细描述。

#### A-device 会话请求协议

1. A-device 关闭 USB 总线电源保护电源, B-device 通过检查 VBUS 的状态识别到这一状况。
2. 当 B-device 想要连接到 A-device, B-device 在数据线产生脉冲请求 A-device 提供 USB 总线电源
3. A-device 通过检查 SRPDETIF (OTG\_INTSTS[13]) 认到 USB 总线电源请求
4. A-device 当 SRPDETIF (OTG\_INTSTS[13]) 由硬件置 1 时通过设置 BUSREG (OTG\_CTL[1]) 为 1 开始驱动 VBUS, 如果 VBUS 在特定时间内达到有效电平, 并且 B-device 已连接, A-device 会变为 USB 主机, HOSTIF (OTG\_INTSTS[7]) 会设为 1。如果 VBUS 在特定时间内不能达到有效电平, 这意味着电流事件发生, 那么 VBEIF (OTG\_IS[1]), 会设为 1。

### B-device会话请求协议

1. A-device 关闭 USB 总线电源保护电源. B-device 通过检查 VBUSVLD (OTG\_STATUS[5]) 确认这一事件.
2. B-device 可以通过设定 BUSREQ(OTG\_CTL[1]) 请求 A-device 提供 USB 总线电源.
3. B-device 会产生 OTG 定义的数据线脉冲
4. A-device 检测到数据线脉冲后会驱动 VBUS , B-device 可以通过确认 VBUSVLD (OTG\_STATUS[5]) 识别这一状况。如果 A-device 在特定时间内驱动 VBUS 到有效电平, B-device 变为 USB 外设, PDEVIF (OTG\_INTSTS[6]) 设为 1. 如果 A-device 没有在特定时间内驱动 VBUS 驱动到有效电平, SRP 错误标志, SRPFIF (OTG\_INTSTS[2]), 会置 1 且 B-device 会进入 OTG 协议定义的空闲态.

### 6.33.5.3 主机谈判协议(HNP)

当USB 框架配置为 OTG 设备模式, 主机功能可以在两个直连的OTG 设备间切换, 而不要改变接头连接, 参考OTG 标准得到HNP 的详细介绍.

#### A-device 主机谈判协议

1. OTG 协议定义的 A-Host 发送 SetFeature b\_hnp\_enable 指令使能 B-device HNP 功能. B-device 响应 ACK 表明 B-device 支持 HNP. 用户需要设定 HNPREQEN (OTG\_CTL[2]) 为 1 使能 HNP 协议.
2. A-Host 完成所有必须的操作之后 设置 BUSREQ (OTG\_CTL[1]) 为 0 进入挂起状态, 使 USB 总线进入 J-state (USB\_D+ 高且 USB\_D- 低)
3. A-Host 通过在特定的时间内检测 USB\_D+ 和 USB\_D- low 为低检测到 B-peripheral 断开连接, A-Host 变为 A-Peripheral。如果 A-Host 在特定时间内不能检测到 B-Peripheral 断开连接, A-Host 会回到空闲态.

#### B-device 主机谈判协议

1. 当 B-Peripheral 成功接收到 SetFeature b\_hnp\_enable 指令, user 通过设定 HNPREQEN (OTG\_CTL[2]) 为 1 使能 B-peripheral HNP 功能.
2. 在检测到 USB 总线在 J-state(USB\_D+ 高且 USB\_D- 低), 用户设置 BUSREG (OTG\_CTL[1]) 为 1. 接着 USB\_D+ 上拉电阻移除导致 USB 断开连接状态 (USB\_D+ 低且 USB\_D- 低).
3. 如果 B-device 在特定时间内检测到 A-device 已连接 (USB\_D+ 高), B-device 变为 B-Host. 如果 B-device 在特定时间内无法检测到 A-device 已连接 (USB\_D+ 高), HNP 错误标志, HNPFIF (OTG\_INTSTS[3]), 会设为 1.

### 6.33.6 寄存器映射

**R:** 只读, **W:** 只写, **R/W:** 可读可写

寄存器	偏移量	R/W	描述	复位值
<b>OTG 基地址:</b>				
<b>OTG_BA = 0x4004_D000</b>				
<b>OTG_CTL</b>	OTG_BA+0x00	R/W	OTG 控制寄存器	0x0000_0000
<b>OTG_PHYCTL</b>	OTG_BA+0x04	R/W	OTG PHY控制寄存器	0x0000_0000
<b>OTG_INTEN</b>	OTG_BA+0x08	R/W	OTG 中断使能寄存器	0x0000_0000
<b>OTG_INTSTS</b>	OTG_BA+0x0C	R/W	OTG 中断状态寄存器	0x0000_0000
<b>OTG_STATUS</b>	OTG_BA+0x10	R	OTG 状态寄存器	0x0000_0006

### 6.33.7 寄存器描述

#### OTG\_CTL OTG 控制寄存器

寄存器	偏移量	R/W	描述	复位值
OTG_CTL	OTG_BA+0x00	R/W	OTG控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved		WKEN	OTGEN	Reserved	HNPREQEN	BUSREQ	VBUSDROP

位	描述
[31:6]	<b>Reserved</b> 保留.
[5]	<b>WKEN</b> <b>OTG ID管脚唤醒使能位</b> 0 = OTG ID 管脚状态改变唤醒功能禁止. 1 = OTG ID 管脚状态改变唤醒功能使能
[4]	<b>OTGEN</b> <b>OTG 功能使能位</b> 用户需要当 USB框架设为 OTG设备时设置此位使能OTG功能. 当 USB框架没有设为 OTG设备时, 此位需要设为0. 0= OTG 功能禁止. 1 = OTG 功能使能.
[3]	<b>Reserved</b> 保留.
[2]	<b>HNPREQEN</b> <b>OTG HNP 请求使能位</b> 当USB框架作为 A-device, 设置此位当A-device 允许执行 HNP协议—A-device 改变角色从主机变为外设. 当OTG状态从a_suspend变为a_peripheral或回到a_idle状态时此位会清0 当 USB 框架作为 B-device, 设置此位在 OTG A-device 成功发送 SetFeature(b_hnp_enable) 指令到 OTG B-device开始角色转换后—B-device改变角色从外设到主机。这一位会在OTG 状态从b_peripheral 变为b_wait_acon 或回到 b_idle 状态后清0. 0 = HNP 请求禁止. 1 = HNP 请求使能 (A-device 改变角色从主机到设备, B-device 可以改变角色从外设到主机). <b>注意:</b> 参考 OTG 协议了解 a_suspend, a_peripheral, a_idle 和 b_idle状态.
[1]	<b>BUSREQ</b> <b>OTG 总线请求</b> 如果OTG A-device 通过USB总线传输数据, 设定此位驱动 VBUS位高检测 USB 设备连接. 如果用户不再想使用总线, 清除此位拉低 VBUS 来节省电源. 这一位当 A-device 变为 A_wait_vfall 状态后清除. 当 VBUSDROP (OTG_CTL[0]) 置1或IDSTS (OTG_STATUS[1]) 改变后清除. 如果OTG-B 设备想要请求VBus, 设定这一位运行SRP协议. 这一位当SRP错误 (在B-

位	描述
	device 在特定时间内执行OTG 协议定义的ARP， OTG A-设备不提供VBUS)后清除。此位也在 VBUSDROP (OTG_CTL[0]) 位为1且 IDSTS (OTG_STATUS[1]) 改变后清除。 0 = OTG A-device 没有发动VBUS 或OTG B-device 没有请求SRP 1 = OTG A-device 发动VBUS 或OTG B-device 请求SRP.
[0]	<b>VBUSDROP</b> 拉低 VBUS控制 如果OTG A-device上的用户应用想要保护电源，设置这一位拉低VBUS. 不管是A-device 或 B-device BUSREQ (OTG_CTL[1])都会清除。 0 = 没有拉低 VBUS. 1 = 拉低 VBUS.

**OTG PHYCTL OTG PHY 控制寄存器**

寄存器	偏移量	R/W	描述	复位值
OTG_PHYCTL	OTG_BA+0x04	R/W	OTG PHY 控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved		VBSTSPOL	VBNPOL	Reserved		IDDETEN	OTGPHYEN

位	描述	
[31:6]	Reserved	保留.
[5]	VBSTSPOL	<p><b>片外 USB VBUS 电源开关状态极性</b></p> <p>片外USB VBUS 电源开关信号的有效极性依据所选择的元件。USB_VBUS_ST 管脚用来监控外设 USB VBUS 电源开关的有效信号. 依据外部USB VBUS 电源开关的极性设置此位。</p> <p>0 = 片外USB VBUS 电源开关信号的有效极性是高. 1 = 片外USB VBUS 电源开关信号的有效极性是低.</p>
[4]	VBNPOL	<p><b>片外 USB VBUS 电源开关使能极性</b></p> <p>OTG 控制器需要在需要时会使能片外 USB VBUS 电源开关提供 VBUS 电源. USB_VBUS_EN 管脚用来控制片外USB VBUS 电源开关.</p> <p>使能外部USB VBUS 电源开关的极性 (高有效或低有效) 依据所选择的器件。设置这一位选择合适的极性.</p> <p>0 = 片外 USB VBUS 电源开关使能高有效. 1 = 片外 USB VBUS 电源开关使能低有效</p>
[3:2]	Reserved	保留.
[1]	IDDETEN	<p><b>ID 检测使能位</b></p> <p>0 = 禁止检测 ID 管脚状态 1 = 使能检测 ID 管脚状态.</p>
[0]	OTGPHYEN	<p><b>OTG PHY 使能</b></p> <p>当 USB 框架配置为OTG-device 或 ID-dependent, 用户需要在使用OTG 功能前设置此位。 如果设备没有配置为 OTG-device 或ID-dependent , 此位不重要</p> <p>0 = OTG PHY 禁止. 1 = OTG PHY 使能.</p>

**OTG\_INTEN OTG 中断使能寄存器**

寄存器	偏移量	R/W	描述	复位值
OTG_INTEN	OTG_BA+0x08	R/W	OTG中断使能寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved		SRPDETIEN	Reserved	SECHGIEN	VBCHGIEN	AVLDCHGIEN	BVLDCHGIEN
7	6	5	4	3	2	1	0
HOSTIEN	PDEVIEEN	IDCHGIEN	GOIDLEIEN	HNPFIEN	SRPFIEN	VBEIEN	ROLECHGIEN

位	描述
[31:14]	<b>Reserved</b> 保留.
[13]	<b>SRPDETIEN</b> SRP 检测中断使能位 0 = 中断禁止. 1 = 中断使能.
[12]	<b>Reserved</b> 保留.
[11]	<b>SECHGIEN</b> SESSEND 状态改变中断使能位 如果此位是 1 且 SESSEND (OTG_STATUS[2]) 状态改变, 中断会产生 0 = 中断禁止. 1 = 中断使能.
[10]	<b>VBCHGIEN</b> VBUSVLD 状态改变中断使能位 如果此位是 1 且 VBUSVLD (OTG_STATUS[5]) 状态改变, 中断会产生 0 = 中断禁止. 1 = 中断使能.
[9]	<b>AVLDCHGIEN</b> A-device 会话有效状态改变中断使能位 如果此位是 1 且 AVLD (OTG_STATUS[4]) 状态改变, 中断会产生 0 = 中断禁止. 1 = 中断使能.
[8]	<b>BVLDCHGIEN</b> B-device 会话有效状态改变中断使能位 如果此位是 1 且 BVLD (OTG_STATUS[3]) 状态改变, 中断会产生. 0 = 中断禁止. 1 = 中断使能.
[7]	<b>HOSTIEN</b> 充当主机中断使能位 如果此位是 1 且设备作为主机, 中断会产生.

位	描述
	0 = 设备变主机中断禁止. 1 = 设备变主机终端使能.
[6]	<b>PDEVIEEN</b> <b>充当外设中断使能位</b> 如果此位为1, 设备变为外设, 中断会产生 0 = 设备变为外设中断禁止. 1 = 设备变为外设中断使能.
[5]	<b>IDCHGIEN</b> <b>IDSTS 改变中断使能</b> 如果此位为 1 且 IDSTS (OTG_STATUS[1]) 状态改变, 中断会产生. 0 = 中断禁止. 1 = 中断使能.
[4]	<b>GOIDLEIEN</b> <b>OTG 设备变为 IDLE 状态中断使能位</b> 0 = 中断禁止. 1 = 中断使能. <b>注意:</b> 进入空闲态意味着进入a_idle 或 b_idle 状态. 请参考OTG标准A-device 状态图 和 B-device 状态图.
[3]	<b>HNPFIEN</b> <b>HNP 失败中断使能位</b> 0 = 中断禁止. 1 = 中断使能.
[2]	<b>SRPFIEN</b> <b>SRP 失败中断使能位</b> 0 = 中断禁止. 1 = 中断使能.
[1]	<b>VBEIEN</b> <b>VBUS 错误中断使能位</b> 0 = 中断禁止. 1 = 中断使能. <b>注意:</b> VBUS 错误意味着进入a_vbus_err状态. 请参考OTG标准A-device 状态图.
[0]	<b>ROLECHGIEN</b> <b>角色(主机或外设) 改变中断使能位</b> 0 = 中断禁止. 1 = 中断使能.

OTG\_INTSTS OTG 中断状态寄存器

寄存器	偏移量	R/W	描述	复位值
OTG_INTSTS	OTG_BA+0x0C	R/W	OTG 中断状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved		SRPDETIF	Reserved	SECHGIF	VBCHGIF	AVLDCHGIF	BVLDCHGIF
7	6	5	4	3	2	1	0
HOSTIF	PDEVIF	IDCHGIF	GOIDLEIF	HNPFIF	SRPFIF	VBEIF	ROLECHGIF

位	描述	
[31:14]	Reserved	保留.
[13]	SRPDETIF	<b>SRP 检测中断状态</b> 0 = SRP 没有检测到. 1 = SRP 检测到. <b>注意:</b> 写1清除这个状态.
[12]	Reserved	保留
[11]	SECHGIF	<b>SESEND 状态改变中断状态</b> 0 = SESSEND (OTG_STATUS[2]) 没有翻转. 1 = SESSEND (OTG_STATUS[2]) 改变. <b>注意:</b> 写1清这个状态.
[10]	VBCHGIF	<b>VBUSVLD 状态改变中断状态</b> 0 = VBUSVLD (OTG_STATUS[5]) 没有翻转. 1 = VBUSVLD (OTG_STATUS[5]) 改变. <b>注意:</b> 写1清这个状态.
[9]	AVLDCHGIF	<b>A-device 会话有效状态改变中断状态</b> 0 = AVLD (OTG_STATUS[4]) 没有翻转. 1 = AVLD (OTG_STATUS[4]) 改变. <b>注意:</b> 写1清这个状态.
[8]	BVLDCHGIF	<b>B-device 会话有效状态改变中断状态</b> 0 = BVLD (OTG_STATUS[3]) 没有翻转. 1 = BVLD (OTG_STATUS[3]) 改变. <b>注意:</b> 写1清这个状态.
[7]	HOSTIF	作为主机中断状态

位	描述
	0= 设备没有作为主机. 1 = 设备作为主机. <b>注意:</b> 写1清这个状态.
[6]	<b>PDEVIF</b> 充当外设中断状态 0= 设备没有充当外设. 1 = 这个设备充当外设. <b>注意:</b> 写1清这个标志.
[5]	<b>IDCHGIF</b> <b>ID 状态改变中断状态</b> 0 = IDSTS (OTG_STATUS[1]) 没有翻转. 1 = IDSTS (OTG_STATUS[1]) 改变. <b>注意:</b> 写1清这个标志.
[4]	<b>GOIDLEIF</b> <b>OTG 进入 IDLE 中断状态</b> 当OTG设备从非空闲变为空闲态此位置1, OTG设备可以是主机或者是外设. 0 = OTG 设备没有回到空闲态 (a_idle 或 b_idle). 1 = OTG 设备回到空闲态(a_idle 或 b_idle). <b>注意 1:</b> 进入空闲态就是进入a_idle 或 b_idle 状态. 请参考 OTG 规范. <b>注意 2:</b> 写1清这个标志.
[3]	<b>HNPFIF</b> <b>HNP 错误中断状态</b> 当 A-device 已授予 B-device 作为主机, 且 USB 总线在 SEO (USB_D+ 和 USB_D-都是低) 状态, 当 A-device 在期望时间内没有连接此位置1. 0 = A-device 在特定时间内连上了 B-device. 1 = A-device 在特定时间内没有连上B-device. <b>注意:</b> 写1清这个标志.
[2]	<b>SRPFIIF</b> <b>SRP 错误中断状态</b> 初始化 SRP 之后, OTG B-device 会等待 OTG A-device 至少在 OTG 规范定义的 TB_SRP_FAIL minimum 的时间内拉高VBUS, 当 OTG B-device 在这个时间内没有检测到 VBUS 拉高, 这个标志置1. 0 = OTG B-device 在这时间内检测到 VBUS 变高. 1 = OTG B-device 没有在这时间内检测到 VBUS 变高. <b>注意:</b> 写1清这个标志.
[1]	<b>VBEIF</b> <b>VBUS 错误中断状态</b> 当OTG A-device 开始驱动VBUS 为高之后, 在最大时间 100ms后VBUS 的电平仍无法达到最低阈值4.4V 时此位置1. 0 = OTG A-device 驱动 VBUS 在期望时间内达到阈值电压. 1 = OTG A-device 驱动 VBUS 在期望时间内没有达到阈值电压. <b>注意:</b> 写 1 清这个标志并恢复VBUS 错误状态.
[0]	<b>ROLECHGIF</b> <b>OTG 角色改变中断状态</b> 当USB_ID 管脚没有改变时, OTG 设备由主机变为外设或由外设变为主机时此位置1. 0 = OTG 设备角色没有改变. 1 = OTG设备角色改变了. <b>注意:</b> 写1清这个标志.

OTG\_STATUS OTG 功能状态寄存器

寄存器	偏移量	R/W	描述	复位值
OTG_STATUS	OTG_BA+0x10	R	OTG 状态寄存器	0x0000_0006

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
ASHOST	ASPERI	VBUSVLD	AVLD	BVLD	SESEND	IDSTS	OVERCUR

位	描述	
[31:8]	Reserved	保留.
[7]	ASHOST	作为 主机 状态 当 OTG 作为主机, 此位置1. 0: OTG 没有作为主机 1: OTG 作为主机
[6]	ASPERI	作为外设状态 当 OTG作为外设, 此位置1. 0: OTG 没有作为外设 1: OTG 作为外设
[5]	VBUSVLD	VBUS 有效状态 当 VBUS高于 4.7V,此位置 1. 0 = VBUS还未有效. 1 = VBUS有效.
[4]	AVLD	A-device 会话有效状态 0 = A-device 会话无效. 1 = A-device 会话有效.
[3]	BVLD	B-device 会话有效状态 0 = B-device 会话无效. 1 = B-device会话有效.
[2]	SESEND	会话结束状态 当 VBUS 电压低于 0.4V, t此位置 1. 会话结束意味这 VBUS上没有电压. 0 = 会话没有结束. 1 = 会话已结束.

位	描述	
[1]	<b>IDSTS</b>	<b>USB_ID Mini-b/Micro-plug 管脚状态</b> 0 = Mini-A/Micro-A 插入. 1 = Mini-B/Micro-B 插入.
[0]	<b>OVERCUR</b>	<b>过流状况</b> 当OTG A-device 开始驱动VBUS 为高之后，在最大时间 100ms后VBUS 的电平仍无法达到最低阈值4.4V时此位置1. 0 = OTG A-device 成功驱动VBUS. 1 = OTG A-device 在特定时间内无法拉高 VBUS

## 6.34 HSOTG 高速 USB OTG

### 6.34.1 概述

HSOTG 控制器连接USB PHY 和 USB 控制器，包含USB 2.0主机控制器和USB 2.0 高速设备控制器。OTG 控制器支持“On-The-Go and Embedded Host Supplement to the USB 2.0 Revision 1.3 Specification”定义的 HNP 和 SRP 协议。

USB 框架，包括 USB 主机，USB 设备，和 OTG 控制器，可以配置为 仅主机，仅设备，依据ID或 HSUSBROLE (SYS\_USBPHY[17:16])定义的OTG 设备模式。在仅主机模式下，USB 框架工作在USB 主机。USB 框架都支持高速，全速和低速传输。在仅从机模式，USB 工作在 USB 设备。USB 框架支持高速和全速传输。在依据 ID模式，USB 框架依据USB\_ID 管脚状态可以工作在 USB 主机和 USB 设备模式。在 OTG 设备模式，USB 框架的角色依据 OTG 标准的定义。USB 框架在OTG 设备作为外设时支持高速和全速传输。

### 6.34.2 特性

- 内建 USB PHY
- 可配置工作在：
  - 仅主机
  - 仅设备
  - 依据ID: USB 框架的角色仅依据USB\_ID 管脚的值—作为 USB 主机 (USB\_ID管脚为低) 或 USB 设备 (USB\_ID 管脚为高). 不支持 HNP 或 SRP 协议
  - OTG 设备: 依据 USB\_ID 管脚状态作为 A-device (USB\_ID 管脚为低) 或 B-device (USB\_ID 管脚为高). 支持 HNP 和 SRP 协议.

### 6.34.3 框图

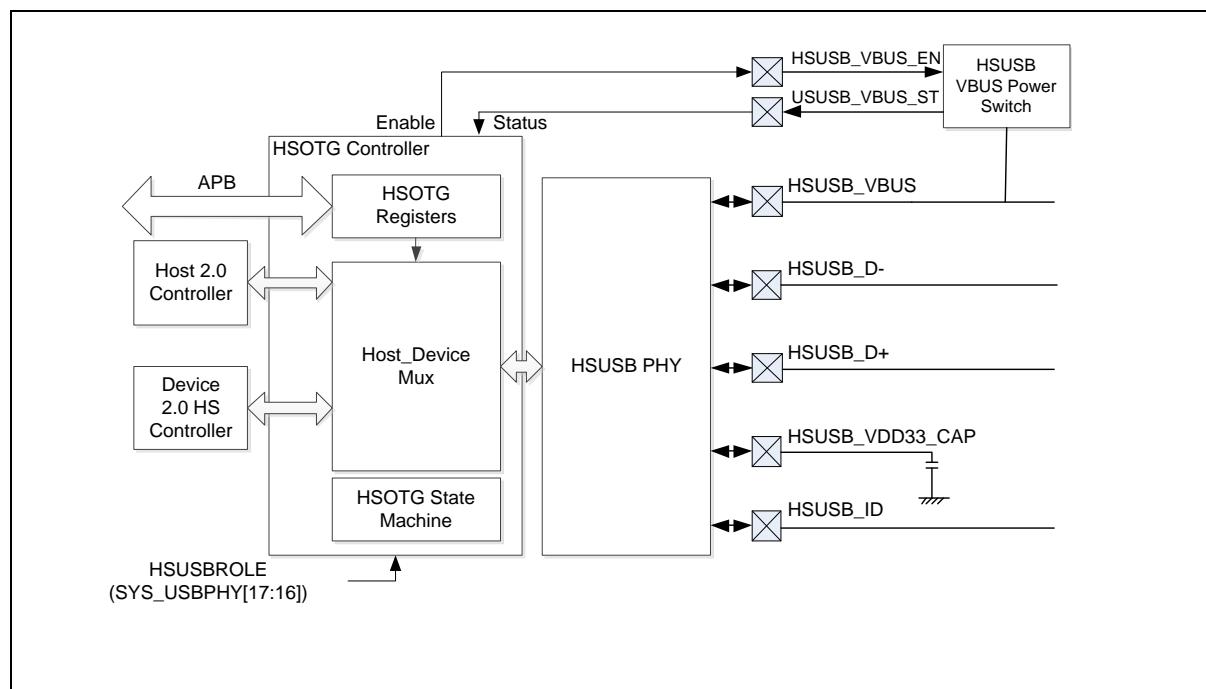


图 6.34-1 USB OTG 框图

#### 6.34.4 基本配置

- 时钟源配置
  - 在HSOTGCKEN(CLK\_AHBCLK[30])使能HSOTG 时钟
- 复位配置
  - 在HSOTGRST (SYS\_IPRST2[30])复位 HSOTG.

OTG 外设时钟可以由HSOTGCKEN (CLK\_APBCLK0[30])使能. USB 框架的结构由 HSUSBROLE (SYS\_USBPHY[17:16])来决定。这两个配置是写保护的，在写这些位之前，用户需要禁止寄存器保护功能。参考SYS\_REGLCTL 寄存器的描述。USB\_VBUS\_EN 和 USB\_VBUS\_ST 管脚功能由SYS\_GPA\_MFPL 或 SYS\_GPC\_MFPL 寄存器定义。

#### 6.34.5 功能描述

USB 框架的角色由HSUSBROLE (SYS\_USBPHY[17:16])和USB\_ID 管脚状态来决定。HSUSBROLE 的配置优先与USB\_ID 管脚的状态。用户客户配置 OTG 控制器作为 USB 主机模式, USB 设备模式, 依据ID 模式或OTG 设备模式。在 USB 主机模式, 主机控制器直接与USB PHY 互动。在 USB 设备模式, t 设备控制器直接与 USB PHY 互动. 在这些情况下, OTG 控制器作为简单的多路复用器使用。在依据ID 模式, USB\_ID 管脚的状态决定 USB 帧充当 USB 主机或 USB 设备。如果USB\_ID 管脚在 FALSE 状态 (低电平), USB 框架充当 USB 主机。如果 USB\_ID 管脚在 TRUE 状态 (高电平), USB 帧充当 USB 设备。在OTG 设备模式, OTG 控制器会执行OTG HNP 和 SRP 协议, 如果USB\_ID 管脚是 FALSE 状态 (低电平), OTG 控制器充当 OTG A-device, 如果 USB\_ID 管脚是 TRUE 状态 (高电平), OTG 控制器充当 OTG B-device.

## 6.34.5.1 USB框架角色

**USB 设备模式**

当 USBROLE (SYS\_USBPHY[1:0]) 为 0, USB 框架充当 USB 设备. USB 主机功能不可用.

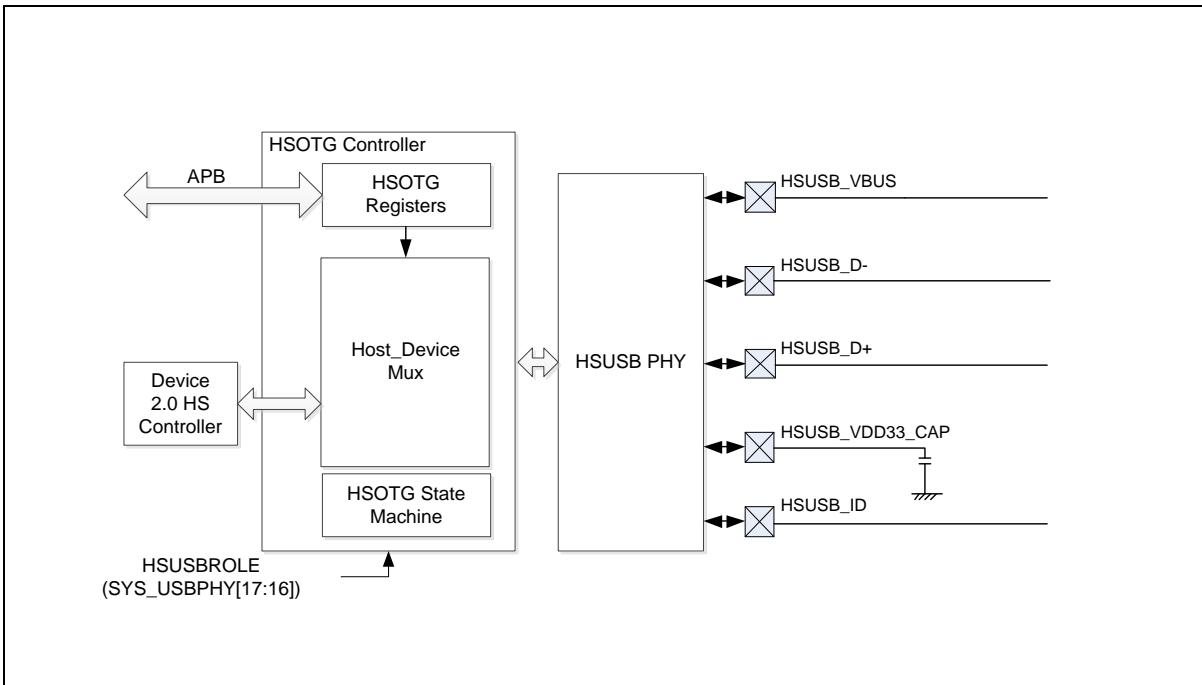


图 6.34-2 USB 设备模式

**USB 主机模式**

当 HSUSBROLE (SYS\_USBPHY[17:16]) 为 1, USB 框架充当 USB 主机. USB 设备功能不可用

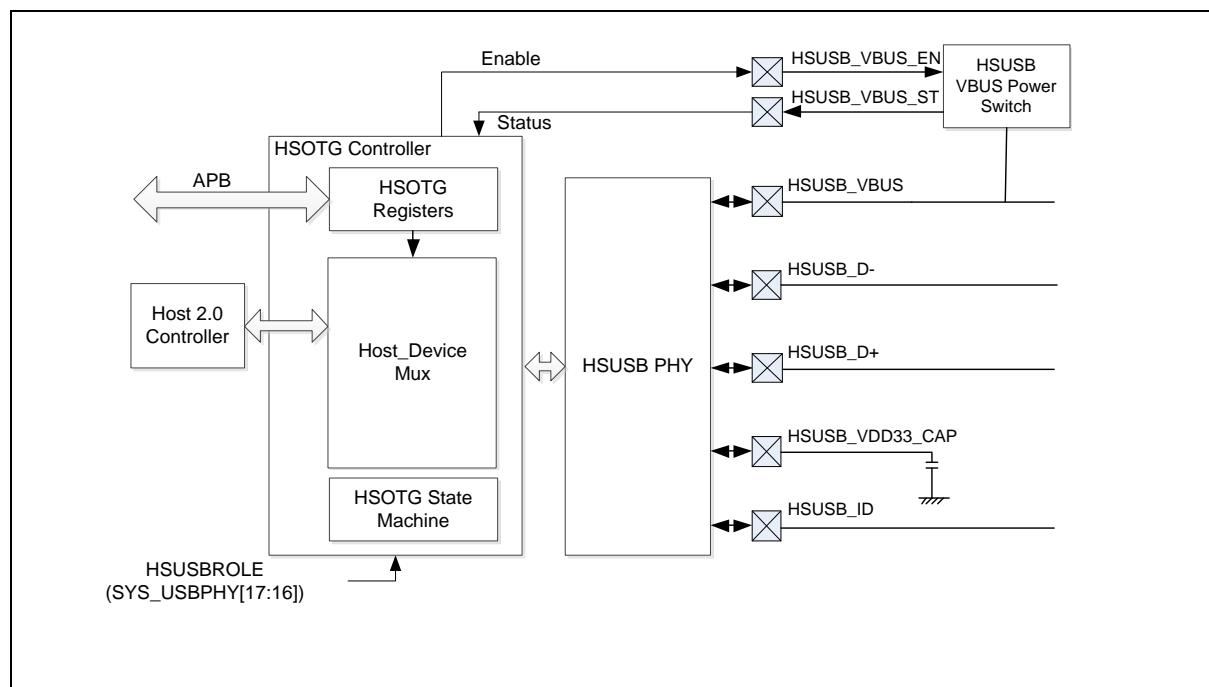


图 6.34-3 USB 主机模式

### 依据ID 模式

当 HSUSBRROLE (SYS\_USBPHY[17:16]) 设为 2, USB 框架的角色依据 USB\_ID 管脚的状态。ID 检测功能由 IDDETEN (HSOTG\_PHYCTL[1]) 使能。USB\_ID 管脚状态影响 IDSTS (HSOTG\_STATUS[1])。当 USB 框架充当 USB 主机 (USB\_ID 管脚低), 框图与 USB 主机模式相同。当 USB 框架充当 USB 设备 (USB\_ID 管脚是高), 框图与 USB 设备模式相同。

### OTG 设备模式

当 HSUSBRROLE (SYS\_USBPHY[17:16]) 等于 3, USB 框架的角色取决于 USB\_ID 管脚的状态。ID 检测功能由 IDDETEN (HSOTG\_PHYCTL[1]) 使能。USB\_ID 管脚状态影响 IDSTS (HSOTG\_STATUS[1])。当 USB\_ID 管脚是低电平, OTG 控制器充当 OTG A-device。当 USB\_ID 管脚是高电平, OTG 控制器充当 OTG B-device。请参考 OTG 标准得到详细的 A-device 和 B-device 的行为描述。

#### 6.34.5.2 会话请求协议(SRP)

当 USB 框架配置为 OTG 设备模式, OTG 控制器支持 SRP 来保护电源, 参考 OTG 标准得到 SRP 的详细描述。

##### A-Device 会话请求协议

1. A-device 关闭 USB 总线电源保护电源, B-device 通过检查 VBUS 的状态识别到这一状况。
2. 当 B-device 想要连接到 A-device, B-device 在数据线产生脉冲请求 A-device 提供 USB 总线电源
3. A-device 通过检查 SRPDETF (HSOTG\_INTSTS[13]) 认到 USB 总线电源请求
4. A-device 当 SRPDETF (HSOTG\_INTSTS[13]) 由硬件置 1 时通过设置 BUSREG (HSOTG\_CTL[1]) 为 1 开始驱动 VBUS, 如果 VBUS 在特定时间内达到有效电平, 并且 B-device 已连接, A-device 会变为 USB 主机, HOSTIF (HSOTG\_INTSTS[7]) 会设为 1. 如果 VBUS 在特定时间内不能达到有效电平, 这意味着电流事件发生, 那么 VBEIF (HSOTG\_IS[1]), 会设为 1.

**B-device 会话请求协议**

1. A-device 关闭 USB 总线电源保护电源. B-device 通过检查 VBUSVLD (HSOTG\_STATUS[5]) 确认这一事件.
2. B-device 可以通过设定 BUSREQ(HSOTG\_CTL[1]) 请求 A-device 提供 USB 总线电源.
3. B-device 会产生 OTG 定义的数据线脉冲
4. A-device 检测到数据线脉冲后会驱动 VBUS , B-device 可以通过确认 VBUSVLD (HSOTG\_STATUS[5]) 识别这一状况。如果 A-device 在特定时间内驱动 VBUS 到有效电平, B-device 变为 USB 外设, PDEVIF (HSOTG\_INTSTS[6]) 设为 1. 如果 A-device 没有在特定时间内驱动 VBUS 驱动到有效电平, SRP 错误标志, SRPFIF (HSOTG\_INTSTS[2]), 会置 1 且 B-device 会进入 OTG 协议定义的空闲态.

**6.34.5.3 主机谈判协议(HNP)**

当USB 框架配置为 OTG 设备模式, 主机功能可以在两个直连的OTG 设备间切换, 而不要改变接头连接, 参考OTG 标准得到HNP 的详细介绍.

**A-device 主机谈判协议**

1. OTG 协议定义的 A-Host 发送 SetFeature b\_hnp\_enable 指令使能 B-device HNP 功能. B-device 响应 ACK 表明 B-device 支持 HNP. 用户需要设定 HNPREQEN (HSOTG\_CTL[2]) 为 1 使能 HNP 协议.
2. A-Host 完成所有必须的操作之后 设置 BUSREQ (HSOTG\_CTL[1]) 为 0 进入挂起状态, 使 USB 总线进入 J-state (USB\_D+ 高且 USB\_D- 低)
3. A-Host 通过在特定的时间内检测 USB\_D+ 和 USB\_D- low 为低检测到 B-peripheral 断开连接, A-Host 变为 A-Peripheral. 如果 A-Host 在特定时间内不能检测到 B-Peripheral 断开连接, A-Host 会回到空闲态.

**B-device 主机谈判协议**

1. 当 B-Peripheral 成功接收到 SetFeature b\_hnp\_enable 指令, user 通过设定 HNPREQEN (HSOTG\_CTL[2]) 为 1 使能 B-peripheral HNP 功能.
2. 在检测到 USB 总线在 J-state(USB\_D+ 高且 USB\_D- 低), 用户设置 BUSREG (HSOTG\_CTL[1]) 为 1. 接着 USB\_D+ 上拉电阻移除导致 USB 断开连接状态 (USB\_D+ 低且 USB\_D- 低).
3. 如果 B-device 在特定时间内检测到 A-device 已连接 (USB\_D+ 高), B-device 变为 B-Host. 如果 B-device 在特定时间内无法检测到 A-device 已连接 (USB\_D+ 高), HNP 错误标志, HNPFIF (HSOTG\_INTSTS[3]), 会设为 1

### 6.34.6 寄存器映射

R: 只读, W: 只写, R/W: 可读可写

寄存器	偏移量	R/W	描述	复位值
<b>HSOTG 基地址:</b>				
<b>HSOTG_BA = 0x4004_F000</b>				
<b>HSOTG_CTL</b>	HSOTG_BA+0x00	R/W	HSOTG 控制寄存器	0x0000_0000
<b>HSOTG_PHYCTL</b>	HSOTG_BA+0x04	R/W	HSOTG PHY 控制寄存器	0x0000_0000
<b>HSOTG_INTEN</b>	HSOTG_BA+0x08	R/W	HSOTG 中断使能寄存器	0x0000_0000
<b>HSOTG_INTSTS</b>	HSOTG_BA+0x0C	R/W	HSOTG 中断状态寄存器	0x0000_0000
<b>HSOTG_STATUS</b>	HSOTG_BA+0x10	R	HSOTG 状态寄存器	0x0000_0006

### 6.34.7 寄存器描述

#### HSOTG\_CTL HSOTG 控制寄存器

寄存器	偏移量	R/W	描述	复位值
HSOTG_CTL	HSOTG_BA+0x00	R/W	HSOTG控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved		WKEN	OTGEN	Reserved	HNPREQEN	BUSREQ	VBUSDROP

位	描述
[31:6]	<b>Reserved</b> 保留.
[5]	<b>WKEN</b> <b>OTG ID 管脚唤醒使能位</b> 0 = OTG ID管脚状态改变唤醒功能禁止. 1 = OTG ID 管脚状态改变唤醒功能使能.
[4]	<b>OTGEN</b> <b>OTG 功能使能位</b> 当USB框架被设为OTG设备时，用户需要设置此位使能OTG功能。如果没有配置成OTG设备，这里需要是0。 0= OTG 功能禁止. 1 = OTG 功能使能.
[3]	<b>Reserved</b> 保留.
[2]	<b>HNPREQEN</b> <b>OTG HNP 请求使能位</b> 当USB框架作为 A-device, 设置此位当A-device 允许执行 HNP协议—A-device 改变角色从主机变为外设。当OTG状态从a_suspend变为a_peripheral或回到a_idle状态时此位会清0 当 USB框架作为 B-device, 设置此位在OTG A-device 成功发送SetFeature (b_hnp_enable) 指令到 OTG B-device开始角色转换后—B-device改变角色从外设到主机。这一位会在OTG 状态从b_peripheral 变为b_wait_acon 或回到 b_idle 状态后清0。 0 = HNP 请求禁止. 1 = HNP 请求使能 (A-device 改变角色从主机到设备， B-device 可以改变角色从外设到主机). <b>注意:</b> 参考 OTG 协议了解 a_suspend, a_peripheral, a_idle 和 b_idle状态.
[1]	<b>BUSREQ</b> <b>OTG 总线请求</b> 如果OTG A-device 通过USB总线传输数据, 设定此位驱动 VBUS位高检测 USB 设备连接. 如果用户不再想使用总线, 清除此位拉低 VBUS 来节省电源. 这一位当 A-device 变为 A_wait_vfall 状态后清除. 当 VBUSDROP (OTG_CTL[0]) 置1或IDSTS (OTG_STATUS[1]) 改变后清除. 如果OTG-B 设备想要请求VBUS, 设定这一位运行SRP协议. 这一位当SRP错误 (在B-

位	描述
	device 在特定时间内执行OTG 协议定义的ARP， OTG A-设备不提供VBUS)后清除。此位也在 VBUSDROP (OTG_CTL[0]) 位为1且 IDSTS (OTG_STATUS[1]) 改变后清除。 0 = OTG A-device 没有发动VBUS 或OTG B-device 没有请求SRP 1 = OTG A-device 发动VBUS 或OTG B-device 请求SRP.
[0]	<b>VBUSDROP</b> 拉低 VBUS控制 如果OTG A-device上的用户应用想要保护电源，设置这一位拉低VBUS. 不管是A-device 或 B-device BUSREQ (OTG_CTL[1])都会清除。 0 = 没有拉低 VBUS. 1 = 拉低 VBUS.

## HSOTG\_PHYCTL HSOTG PHY 控制寄存器

寄存器	偏移量	R/W	描述	复位值
HSOTG_PHYCTL	HSOTG_BA+0x04	R/W	HSOTG PHY 控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved		VBSTSPOL	VBNPOL	Reserved		IDDETEN	OTGPHYEN

位	描述	
[31:6]	Reserved	保留.
[5]	VBSTSPOL	<p><b>片外 USB VBUS 电源开关状态极性</b></p> <p>片外USB VBUS 电源开关信号的有效极性依据所选择的元件。USB_VBUS_ST 管脚用来监控外设 USB VBUS 电源开关的有效信号. 依据外部USB VBUS 电源开关的极性设置此位。</p> <p>0 = 片外USB VBUS 电源开关信号的有效极性是高. 1 = 片外USB VBUS 电源开关信号的有效极性是低.</p>
[4]	VBNPOL	<p><b>片外 USB VBUS 电源开关使能极性</b></p> <p>OTG 控制器需要在需要时会使能片外 USB VBUS 电源开关提供 VBUS 电源. USB_VBUS_EN 管脚用来控制片外 USB VBUS 电源开关.</p> <p>使能外部USB VBUS 电源开关的极性 (高有效或低有效) 依据所选择的器件。设置这一位选择合适的极性.</p> <p>0 = 片外 USB VBUS 电源开关使能高有效. 1 = 片外 USB VBUS 电源开关使能低有效</p>
[3:2]	Reserved	保留.
[1]	IDDETEN	<p><b>ID 检测使能位</b></p> <p>0 = 禁止检测 ID 管脚状态 1 = 使能检测 ID 管脚状态.</p>
[0]	OTGPHYEN	<p><b>OTG PHY 使能</b></p> <p>当 USB 框架配置为OTG-device 或 ID-dependent, 用户需要在使用OTG 功能前设置此位。 如果设备没有配置为 OTG-device 或ID-dependent , 此位不重要</p> <p>0 = OTG PHY 禁止. 1 = OTG PHY 使能.</p>

**HSOTG INTEN HSOTG 中断使能寄存器**

寄存器	偏移量	R/W	描述	复位值
HSOTG_INTE N	HSOTG_BA+0x08	R/W	HSOTG中断使能寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved		SRPDETIEN	Reserved	SECHGIEN	VBCHGIEN	AVLDCHGIEN	BVLDCHGIEN
7	6	5	4	3	2	1	0
HOSTIEN	PDEVIEEN	IDCHGIEN	GOIDLEIEN	HNPFIEN	SRPFIEN	VBEIEN	ROLECHGIEN

位	描述	
[31:14]	Reserved	保留.
[13]	SRPDETIEN	<b>SRP 检测中断使能位</b> 0 = 中断禁止. 1 = 中断使能.
[12]	Reserved	保留.
[11]	SECHGIEN	<b>SESEND 状态改变中断使能位</b> 如果此位是 1 且 SESSEND (OTG_STATUS[2]) 状态改变, 中断会产生 0 = 中断禁止. 1 = 中断使能.
[10]	VBCHGIEN	<b>VBUSVLD 状态改变中断使能位</b> 如果此位是 1 且 VBUSVLD (OTG_STATUS[5]) 状态改变, 中断会产生 0 = 中断禁止. 1 = 中断使能.
[9]	AVLDCHGIEN	<b>A-device 会话有效状态改变中断使能位</b> 如果此位是 1 且 AVLD (OTG_STATUS[4]) 状态改变, 中断会产生 0 = 中断禁止. 1 = 中断使能.
[8]	BVLDCHGIEN	<b>B-device 会话有效状态改变中断使能位</b> 如果此位是 1 且 BVLD (OTG_STATUS[3]) 状态改变, 中断会产生. 0 = 中断禁止. 1 = 中断使能.
[7]	HOSTIEN	<b>充当主机中断使能位</b> 如果此位是 1 且设备作为主机, 中断会产生.

位	描述
	0 = 设备变主机中断禁止. 1 = 设备变主机终端使能.
[6]	<b>PDEVIEEN</b> <b>充当外设中断使能位</b> 如果此位为1，设备变为外设，中断会产生 0 = 设备变为外设中断禁止. 1 = 设备变为外设中断使能.
[5]	<b>IDCHGIEN</b> <b>IDSTS 改变中断使能</b> 如果此位为 1 且 IDSTS (OTG_STATUS[1]) 状态改变，中断会产生. 0 = 中断禁止. 1 = 中断使能.
[4]	<b>GOIDLEIEN</b> <b>OTG 设备变为 IDLE 状态中断使能位</b> 0 = 中断禁止. 1 = 中断使能. <b>注意:</b> 进入空闲态意味着进入a_idle 或 b_idle 状态. 请参考OTG标准A-device 状态图 和 B-device 状态图.
[3]	<b>HNPFIEN</b> <b>HNP 失败中断使能位</b> 0 = 中断禁止. 1 = 中断使能.
[2]	<b>SRPFIEN</b> <b>SRP 失败中断使能位</b> 0 = 中断禁止. 1 = 中断使能.
[1]	<b>VBEIEN</b> <b>VBUS 错误中断使能位</b> 0 = 中断禁止. 1 = 中断使能. <b>注意:</b> VBUS 错误意味着进入a_vbus_err状态. 请参考OTG标准A-device 状态图.
[0]	<b>ROLECHGIEN</b> <b>角色 (主机或外设) 改变中断使能位</b> 0 = 中断禁止. 1 = 中断使能.

**HSOTG\_INTSTS HSOTG 中断状态寄存器**

寄存器	偏移量	R/W	描述	复位值
HSOTG_INTSTS	HSOTG_BA+0x0C	R/W	HSOTG 中断状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved		SRPDETIF	Reserved	SECHGIF	VBCHGIF	AVLDCHGIF	BVLDCHGIF
7	6	5	4	3	2	1	0
HOSTIF	PDEVIF	IDCHGIF	GOIDLEIF	HNPFIF	SRPFIF	VBEIF	ROLECHGIF

位	描述	
[31:14]	Reserved	保留.
[13]	SRPDETIF	<b>SRP 检测中断状态</b> 0 = SRP 没有检测到. 1 = SRP 检测到. <b>注意:</b> 写1清除这个状态.
[12]	Reserved	保留
[11]	SECHGIF	<b>SESEND 状态改变中断状态</b> 0 = SESSEND (OTG_STATUS[2]) 没有翻转. 1 = SESSEND (OTG_STATUS[2]) 改变. <b>注意:</b> 写1清这个状态.
[10]	VBCHGIF	<b>VBUSVLD 状态改变中断状态</b> 0 = VBUSVLD (OTG_STATUS[5]) 没有翻转. 1 = VBUSVLD (OTG_STATUS[5]) 改变. <b>注意:</b> 写1清这个状态.
[9]	AVLDCHGIF	<b>A-device 会话有效状态改变中断状态</b> 0 = AVLD (OTG_STATUS[4]) 没有翻转. 1 = AVLD (OTG_STATUS[4]) 改变. <b>注意:</b> 写1清这个状态.
[8]	BVLDCHGIF	<b>B-device 会话有效状态改变中断状态</b> 0 = BVLD (OTG_STATUS[3]) 没有翻转. 1 = BVLD (OTG_STATUS[3]) 改变. <b>注意:</b> 写1清这个状态.
[7]	HOSTIF	作为主机中断状态

位	描述
	<p>0= 设备没有作为主机. 1 = 设备作为主机. <b>注意:</b> 写1清这个状态.</p>
[6]	<p><b>PDEVIF</b></p> <p>充当外设中断状态 0= 设备没有充当外设. 1 = 这个设备充当外设. <b>注意:</b> 写1清这个标志.</p>
[5]	<p><b>IDCHGIF</b></p> <p><b>ID 状态改变中断状态</b> 0 = IDSTS (OTG_STATUS[1]) 没有翻转. 1 = IDSTS (OTG_STATUS[1]) 改变. <b>注意:</b> 写1清这个标志.</p>
[4]	<p><b>GOIDLEIF</b></p> <p><b>OTG 进入 IDLE 中断状态</b> 当OTG设备从非空闲变为空闲态此位置1, OTG设备可以是主机或者是外设. 0 = OTG 设备没有回到空闲态 (a_idle 或 b_idle). 1 = OTG 设备回到空闲态(a_idle 或 b_idle). <b>注意 1:</b> 进入空闲态就是进入a_idle 或 b_idle 状态. 请参考 OTG 规范. <b>注意 2:</b> 写1清这个标志.</p>
[3]	<p><b>HNPFIF</b></p> <p><b>HNP 错误中断状态</b> 当 A-device 已授予 B-device 作为主机, 且 USB 总线在 SEO (USB_D+ 和 USB_D-都是低) 状态, 当 A-device 在期望时间内没有连接此位置1. 0 = A-device 在特定时间内连上了 B-device. 1 = A-device 在特定时间内没有连上B-device. <b>注意:</b> 写1清这个标志.</p>
[2]	<p><b>SRPFIIF</b></p> <p><b>SRP 错误中断状态</b> 初始化 SRP 之后, OTG B-device 会等待 OTG A-device 至少在 OTG 规范定义的 TB_SRPF_FAIL minimum 的时间内拉高VBUS, 当 OTG B-device 在这个时间内没有检测到 VBUS 拉高, 这个标志置1. 0 = OTG B-device 在这时间内检测到 VBUS 变高. 1 = OTG B-device 没有在这时间内检测到 VBUS 变高. <b>注意:</b> 写1清这个标志.</p>
[1]	<p><b>VBEIF</b></p> <p><b>VBUS 错误中断状态</b> 当OTG A-device 开始驱动VBUS 为高之后, 在最大时间 100ms后VBUS 的电平仍无法达到最低阈值4.4V 时此位置1. 0 = OTG A-device 驱动 VBUS 在期望时间内达到阈值电压. 1 = OTG A-device 驱动 VBUS 在期望时间内没有达到阈值电压. <b>注意:</b> 写 1 清这个标志并恢复VBUS 错误状态.</p>
[0]	<p><b>ROLECHGIF</b></p> <p><b>OTG 角色改变中断状态</b> 当USB_ID 管脚没有改变时, OTG 设备由主机变为外设或由外设变为主机时此位置1. 0 = OTG 设备角色没有改变. 1 = OTG设备角色改变了. <b>注意:</b> 写1清这个标志.</p>

**HSOTG\_STATUS HSOTG 功能状态寄存器**

寄存器	偏移量	R/W	描述	复位值
HSOTG_STATUS	HSOTG_BA+0x10	R	HSOTG 状态寄存器	0x0000_0006

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
ASHOST	ASPERI	VBUSVLD	AVLD	BVLD	SESEND	IDSTS	OVERCUR

位	描述	
[31:8]	Reserved	保留
[7]	ASHOST	作为 主机 状态 当 OTG 作为主机, 此位置1. 0: OTG 没有作为主机 1: OTG 作为主机
[6]	ASPERI	作为外设状态 当 OTG作为外设, 此位置1. 0: OTG 没有作为外设 1: OTG 作为外设
[5]	VBUSVLD	<b>VBUS 有效状态</b> 当 VBUS高于 4.7V,此位置 1. 0 = VBUS还未有效. 1 = VBUS有效.
[4]	AVLD	<b>A-device 会话有效状态</b> 0 = A-device 会话无效. 1 = A-device 会话有效.
[3]	BVLD	<b>B-device 会话有效状态</b> 0 = B-device 会话无效. 1 = B-device会话有效.
[2]	SESEND	会话结束状态 当 VBUS 电压低于 0.4V, 此位置 1. 会话结束意味这 VBUS上没有电压. 0 = 会话没有结束. 1 = 会话已结束.

位	描述	
[1]	IDSTS	<b>USB_ID Mini-b/Micro-plug 管脚状态</b> 0 = Mini-A/Micro-A 插入. 1 = Mini-B/Micro-B 插入.
[0]	OVERCUR	<b>过流状况</b> 当OTG A-device 开始驱动VBUS 为高之后，在最大时间 100ms后VBUS 的电平仍无法达到最低阈值4.4V时此位置1. 0 = OTG A-device 成功驱动VBUS. 1 = OTG A-device 在特定时间内无法拉高 VBUS

## 6.35 CRC 控制器

### 6.35.1 概述

CRC循环冗余发生器使用4种常见的多项式CRC-CCITT, CRC-8, CRC-16, 和 CRC-32 执行 CRC 计算

### 6.35.2 特性

- 支持4种常见的多项式 CRC-CCITT, CRC-8, CRC-16, 和 CRC-32
  - CRC-CCITT:  $X^{16} + X^{12} + X^5 + 1$
  - CRC-8:  $X^8 + X^2 + X + 1$
  - CRC-16:  $X^{16} + X^{15} + X^2 + 1$
  - CRC-32:  $X^{32} + X^{26} + X^{23} + X^{22} + X^{16} + X^{12} + X^{11} + X^{10} + X^8 + X^7 + X^5 + X^4 + X^2 + X + 1$
- 可编程种子值
- 对于输入数据和CRC 校验和, 支持可编程的位顺序反转设定
- 对于输入数据和CRC 校验和, 支持可编程的补码设定
- 支持 8/16/32-bit 数据宽度
  - 8-bit 写模式: 1-AHB 时钟周期操作
  - 16-bit 写模式: 2-AHB 时钟周期操作
  - 32-bit 写模式: 4-AHB 时钟周期操作
- 支持使用 PDMA 写数据执行 CRC 操作

### 6.35.3 框图

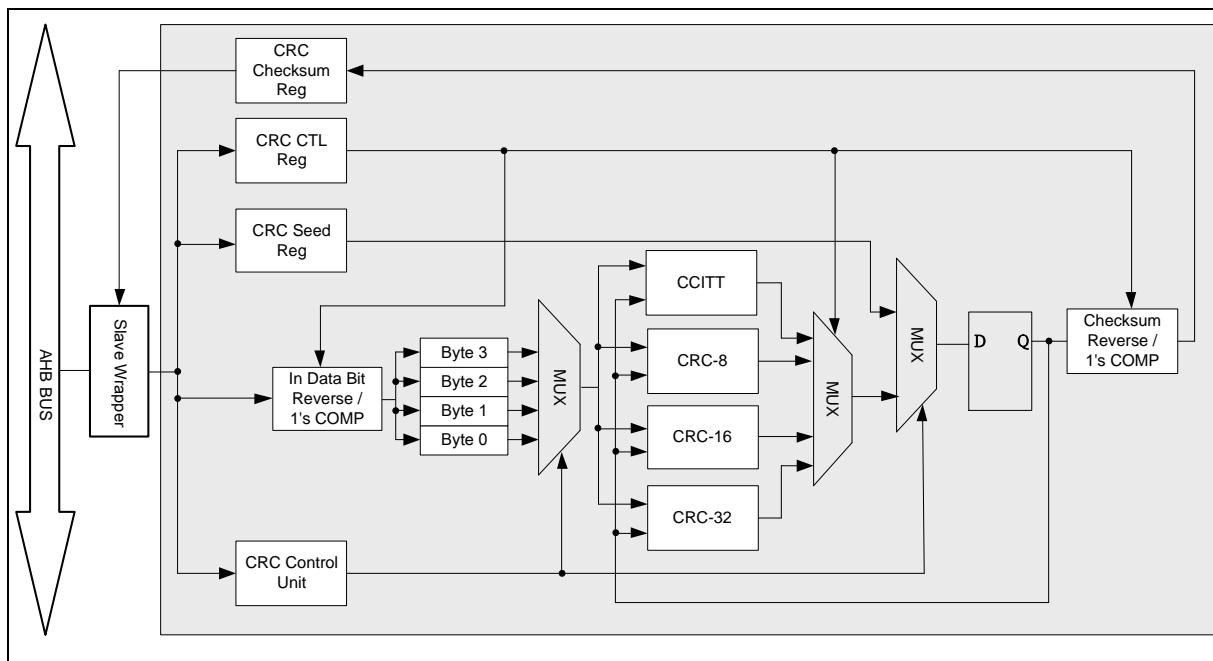


图 6.35-1 CRC 发生器框图

#### 6.35.4 基本配置

- 时钟源配置
  - 在CRCCKEN (CLK\_AHCLK[7])使能CRC 外设时钟
- 复位配置
  - 在CRCRST (SYS\_IPRST0[7])复位CRC 控制器.

#### 6.35.5 功能描述

CRC 计算可选择4中常见多项式：CRC-CCITT, CRC-8, CRC-16 和 CRC-32; 通过 CRCMODE[1:0] (CRC\_CTL[31:30])选择 CRC多项式

下面是编程流程示例：

1. 设置CRCEN (CRC\_CTL[0])使能CRC 发生器
2. 初始化计算设定.
  - 1) 设置CHKSFMT (CRC\_CTL[27])配置CRC 校验和补码.
  - 2) 设置 CHKSREV (CRC\_CTL[25])配置CRC校验和位顺序反转见图 6.35-2 校验和位顺序反转功能框图
  - 3)
  - 4) 配置DATFMT (CRC\_CTL[26])配置CRC 写数据补码
  - 5) 配置DATREV (CRC\_CTL[24])配置 CRC 写数据每个字节的位顺序反转, 功能框图见图 6.35-3.
3. 执行 CHKSINIT (CRC\_CTL[1]) 从CRC\_SEED寄存器中载入检验和初值.
4. 写数据到 CRC\_DAT 寄存器计算 CRC 校验和.

5. 读CRC\_CHECKSUM 寄存器得到CRC 校验和结果

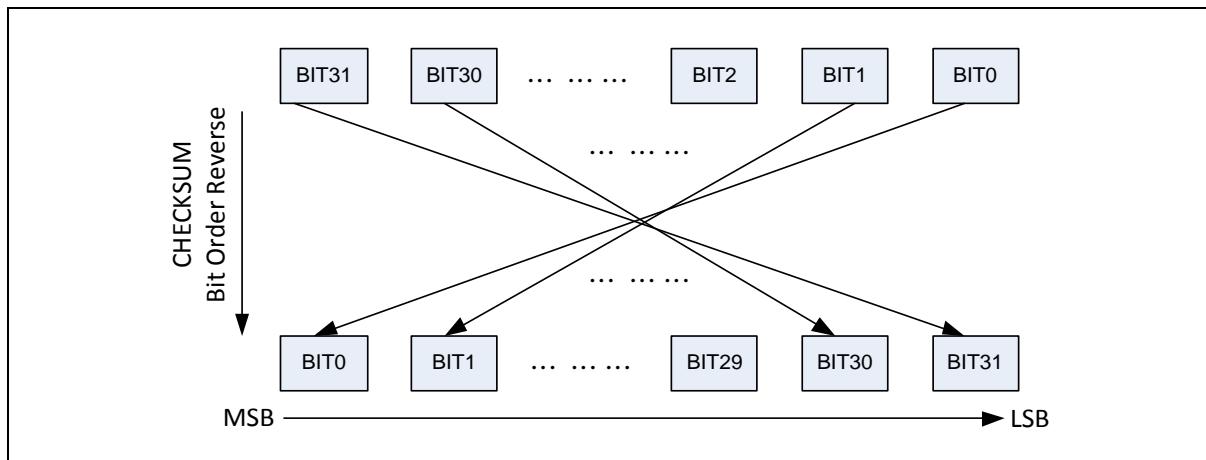


图 6.35-2 校验和位顺序反转功能框图

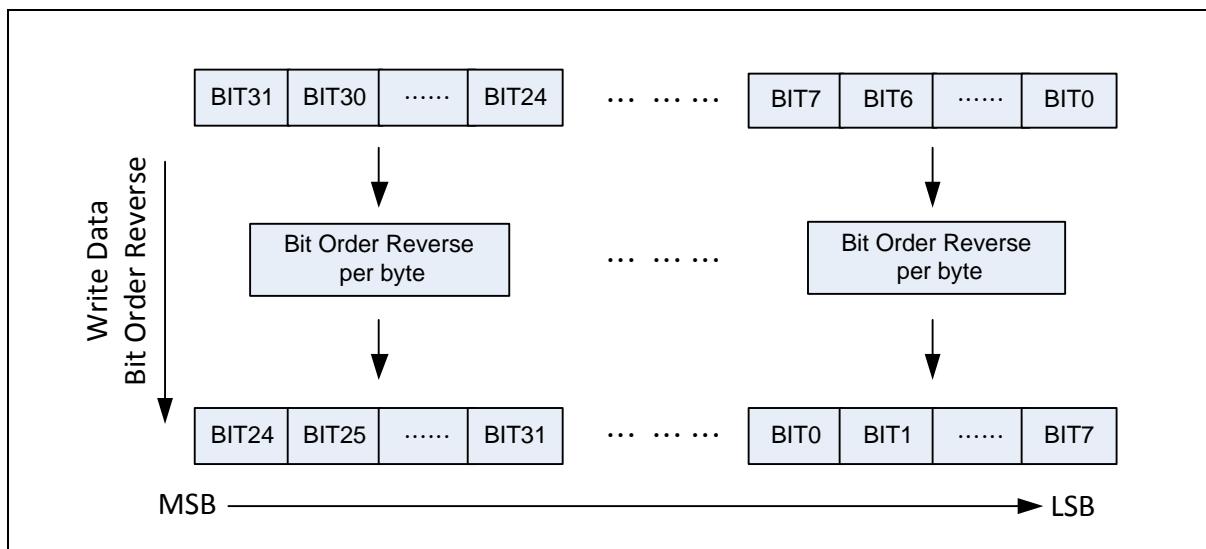


图 6.35-3 写数据位顺序反转功能框图

### 6.35.6 寄存器映射

R: 只读, W: 只写, R/W: 可读可写

寄存器	偏移量	R/W	描述	复位值
<b>CRC 基地址:</b> <b>CRC_BA = 0x4003_1000</b>				
<b>CRC_CTL</b>	CRC_BA+0x00	R/W	CRC 控制寄存器	0x2000_0000
<b>CRC_DAT</b>	CRC_BA+0x04	R/W	CRC 写数据寄存器	0x0000_0000
<b>CRC_SEED</b>	CRC_BA+0x08	R/W	CRC 种子寄存器	0xFFFF_FFFF
<b>CRC_CHECKSUM</b>	CRC_BA+0x0C	R	CRC 校验和寄存器	0xFFFF_FFFF

### 6.35.7 寄存器描述

#### CRC\_CTL CRC 控制寄存器

寄存器	偏移量	R/W	描述	复位值
CRC_CTL	CRC_BA+0x00	R/W	CRC控制寄存器	0x2000_0000

31	30	29	28	27	26	25	24
<b>CRCMODE</b>		<b>DATLEN</b>		<b>CHKSFMT</b>	<b>DATFMT</b>	<b>CHKSREV</b>	<b>DATREV</b>
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved						<b>CHKSINIT</b>	<b>CRCEN</b>

位	描述
[31:30]	<b>CRCMODE</b>  CRC 多项式模式 这里选择 CRC 多项式模式. 00 = CRC-CCITT多项式模式. 01 = CRC-8多项式模式. 10 = CRC-16多项式模式. 11 = CRC-32多项式模式.
[29:28]	<b>DATLEN</b>  CPU 写数据长度 这里设置写数据长度. 00 = 数据长度是 8-bit模式. 01 = 数据长度是 16-bit 模式. 1x = 数据长度是 32-bit 模式.  <b>注意:</b> 当写数据长度是8-bit 模式, CRC_DAT 寄存器的有效数据是DATA[7:0]位; 如果写数据长度是 16-bit 模式, CRC_DAT寄存器的有效数据是 DATA[15:0].
[27]	<b>CHKSFMT</b>  校验和补码 0 = 禁止 CRC校验和补码. 1 = 使能 CRC 校验和补码.
[26]	<b>DATFMT</b>  写数据补码 0 =禁止 CRC 写入数据做补码. 1 = 使能CRC 写入数据做补码.
[25]	<b>CHKSREV</b>  校验和位顺序反转. 0 = 禁止 CRC 校验和位顺序反转. 1 = 使能CRC 校验和位顺序反转.  <b>注意:</b> 如果校验和结果是0xDD7B0F2E, 位顺序反转的CRC 校验和是0x74F0DEBB.

[24]	<b>DATREV</b>	<b>写数据位顺序反转</b> 这一位用来使能写入CRC_DAT 寄存器的值位顺序反转. 0 = CRC 写入数据位顺序反转禁止. 1 = CRC 写入数据位顺序反转使能 (按字节看). <b>注意:</b> 如果写入数据是0xAABBCCDD,位顺序反转过的 CRC 写入数据是 0x55DD33BB.
[23:2]	<b>Reserved</b>	保留.
[1]	<b>CHKSINIT</b>	<b>校验和初始化</b> 0 = 没影响. 1 = 初始化校验和的值, 载入 CRC_SEED 寄存器的值到CRC_CHECKSUM 寄存器. <b>注意:</b> 这一位会自动清除
[0]	<b>CRCEN</b>	<b>CRC 操作使能位</b> 0 = 没影响 1 = CRC 操作使能.

**CRC\_DAT CRC 写数据寄存器**

寄存器	偏移量	R/W	描述	复位值
CRC_DAT	CRC_BA+0x04	R/W	CRC写数据寄存器	0x0000_0000

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

位	描述	
[31:0]	DATA	<b>CRC 写数据位</b> 可以CPU直接写入，或者使用PDMA功能写数据到这个寄存器执行CRC操作 <b>注意：</b> 当数据长度是8-bit 模式， CRC_DAT 寄存器的有效数据是DATA[7:0]位；如果写数据长度是 16-bit 模式，CRC_DAT寄存器的有效数据是 DATA[15:0].

**CRC\_SEED CRC 种子寄存器**

寄存器	偏移量	R/W	描述	复位值
CRC_SEED	CRC_BA+0x08	R/W	CRC 种子寄存器	0xFFFF_FFFF

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

位	描述	
[31:0]	SEED	<b>CRC 种子值</b> 这里是 CRC 种子值。 <b>注意:</b> 这里在执行 CHKSINIT (CRC_CTL[1]) 后会重载作为校验和的初始值 (CRC_CHECKSUM 寄存器)

**CRC\_CHECKSUM CRC 校验和寄存器**

寄存器	偏移量	R/W	描述	复位值
CRC_CHECKSUM	CRC_BA+0x0C	R	CRC校验和寄存器	0xFFFF_FFFF

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

位	描述	
[31:0]	CHECKSUM	CRC 校验和结果(只读) 这里是CRC校验和结果

## 6.36 CRYPTO 加解密算法加速器

### 6.36.1 概述

支持 AES, DES/TDES, SHA 和 HMAC 算法, 包含一个伪随机数生成器(PRNG), 可产生 64 位, 128 位, 192 位, 和 256 位随机数。

AES引擎支持ECB、CBC、CFB、OFB、CTR、CBC-CS1、CBC-CS2、和 CBC-CS3 模式。

DES/TDES引擎支持ECB、CBC、CFB，OFB和CTR模式。

SHA引擎完全兼容SHA -160、SHA-224、SHA-256和相应的 HMAC 算法

ECC 加速器在二元域和素数域中用多项式基实现完全符合椭圆曲线密码体制

### 6.36.2 特性

- PRNG
  - 支持 64位, 128 位 , 192 位, 和 256位随机数的产生
- AES
  - 支持 FIPS NIST 197
  - 支持 SP800-38A 和 addendum
  - 支持 128, 192, 和 256 位密钥
  - 支持 ECB, CBC, CFB, OFB, CTR, CBC-CS1, CBC-CS2, 和 CBC-CS3 模式
  - 支持密钥扩展
- DES
  - 支持 FIPS 46-3
  - 支持加密和解密
  - 支持 ECB, CBC, CFB, OFB, 和 CTR 模式
- TDES
  - 支持 FIPS NIST 800-67
  - 根据 X9.52 标准执行
  - 支持双密钥或三密钥模式
  - 支持加密和解密
  - 支持 ECB, CBC, CFB, OFB, 和 CTR 模式
- SHA
  - 支持 FIPS NIST 180, 180-2
  - 支持 SHA-160, SHA-224, SHA-256, SHA-384, 和 SHA-512
- HMAC
  - 支持 FIPS NIST 180, 180-2
  - 支持 HMAC-SHA-160, HMAC-SHA-224, HMAC-SHA-256, HMAC-SHA-384, 和 HMAC-SHA-512

- ECC

- 同时支持素数域  $GF(p)$  和二进制域  $GF(2^m)$
- 支持 NIST P-192, P-224, P-256, P-384, 和 P-521
- 支持 NIST B-163, B-233, B-283, B-409, 和 B-571
- 支持 NIST K-163, K-233, K-283, K-409, 和 K-571
- 在  $GF(p)$  和  $GF(2^m)$  支持点乘法, 加法和加倍运算 在  $GF(p)$  和  $GF(2^m)$
- 在  $GF(p)$  支持模划分, 乘法, 加法和减法运算
- 

### 6.36.3 框图

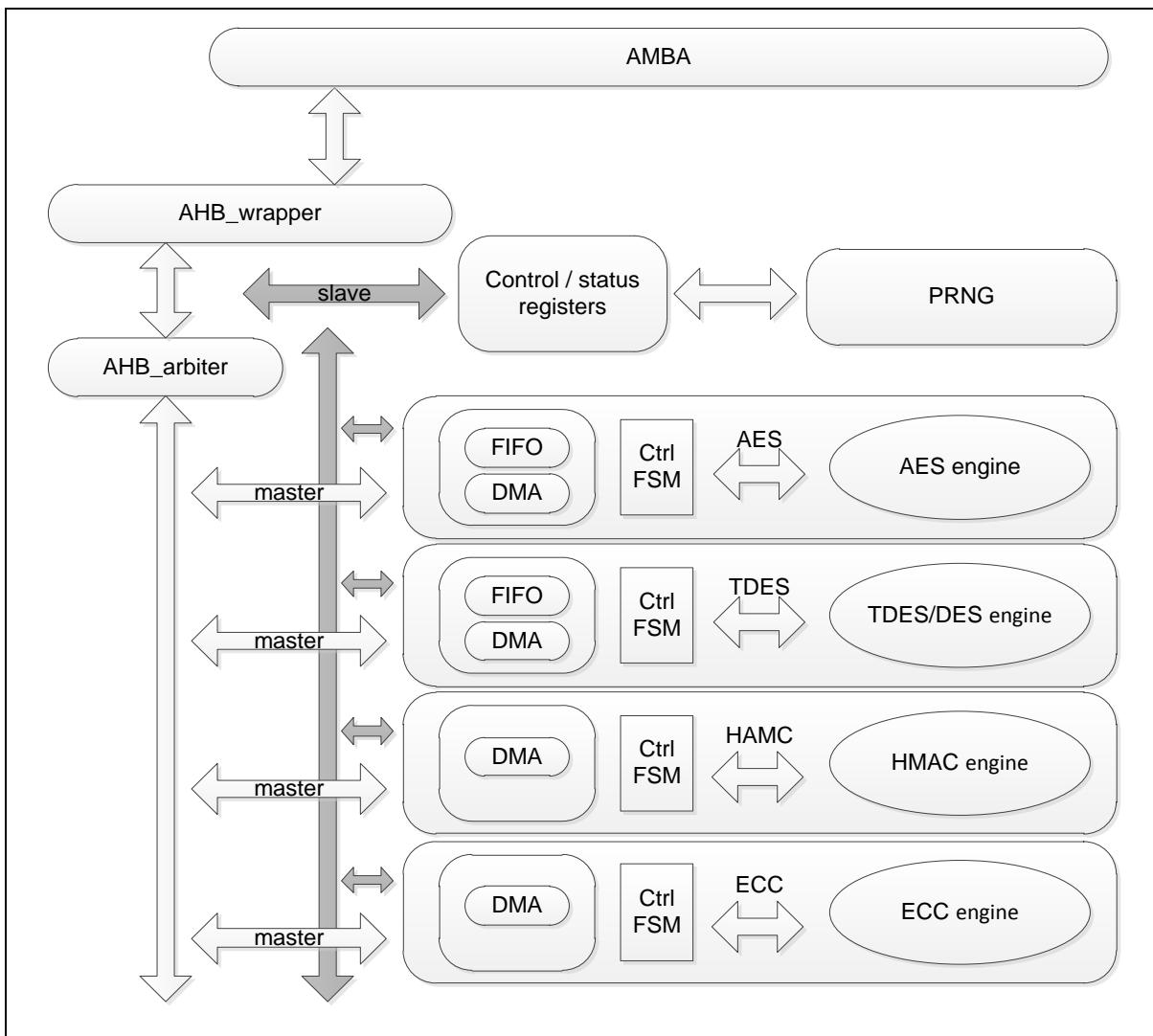


图 6.36-1 加解算法加速器框图

#### 6.36.4 基本配置

- 时钟源使能
  - CRYPTO时钟源使能位: CRYPTOCKEN (CLK\_AHCLK[12])
- 复位使能
  - 复位CRYPTO 控制在CRYPTORST (SYS\_IPRST0[12])

#### 6.36.5 功能描述

CRYPTO加解算法加速器包含一个安全的伪随机数生成器（PRNG）内核，支持AES、DES/TDES、SHA和ECC的算法。该引擎可用于不同的数据安全应用，例如需要加密保护和完整性的安全通信。

1. PRNG 内核依据支持 KEYSZ 的设定，支持 64 位, 128 位, 192 位, 以及 256 位随机数的生成。  
AES 加速器完全兼容实现 AES (高级加密标准) 加解密的算法。AES 加速器支持 ECB, CBC, CFB, OFB, CTR, CBC-CS1, CBC-CS2, 和 CBC-CS3 模式。AES 加速器提供 DMA 功能降低 CPU 负担，并且支持 3 种突发长度, 16 字, 8 字, 和 4 字。
2. DES/TDES 加速器完全兼容实现 DES 和三倍 DES 加解密算法。DES/TDES 加速器支持 ECB, CBC, CFB, OFB, 和 CTR 模式。DES/TDES 也支持 DMA 功能降低 CPU 负担。支持两种突发长度, 4 字和 8 字。5
3. SHA/HMAC 加速器完全兼容实现 SHA-160, SHA-224, SHA-256, SHA-384, SHA-512, 和相应的 HMAC 算法。SHA/HMAC 加速器也支持 DMA 功能降低 CPU 参与。支持 3 种, 16 字, 8 字和 4 字。
4. ECC 加速器完全兼容实现素数域 GF(p) 和二进制域 GF( $2^m$ ) 算法。素数域 GF(p) 支持 NIST P-192, P-224, P-256, P-384 和 P-521。二进制域 GF( $2^m$ ) 支持 NIST B-163, B-233, B-283, B-409, B-571 和 NIST K-163, K-233, K-283, K-409 和 K-571。

程序可以使能CRYPTO\_INTEN 控制数据流，检查CRYPTO\_INTSTS 监控加速器状态。当任意引擎发生操作错误或缓存错误，相应的错误标识会置1，如果中断使能会通知CPU。如果想知道错误的详细信息，程序可以确认每个引擎的状态标志寄存器。表 6.36-1列出所有引擎错误使能位，错误标志位和错误条件。

引擎	错误中断使能位	错误中断标志	错误条件
AES	AESEIEN (CRYPTO_INTEN[1])	AESEIF (CRYPTO_INTSTS[1])	INBUFERR/OUTBUFERR/BUSERR
TDES/DES	TDESEIEN (CRYPTO_INTEN[9])	TDEEIF (CRYPTO_INTSTS[9])	INBUFERR/OUTBUFERR/BUSERR
SHA/HMAC	HMACEIEN (CRYPTO_INTEN[25])	HMACEIF (CRYPTO_INTSTS[25])	DMAERR/BUSERR
ECC	ECCEIEN (CRYPTO_INTEN[23])	ECCEIF (CRYPTO_INTSTS[23])	BUSERR

表 6.36-1 每个引擎错误条件和错误标识

加解算法加速器支持下列特性提升性能

### DMA 模式

当CPU配置好了DMA 源地址寄存器, 目的地址寄存器, 和字节数寄存器, 对加速器搬运数据完全由DMA 逻辑完成. 这个模式可以降低 CPU 的负担. 加解算法加速器为AES引擎内建4个硬件DMA通道, DES/TDES引擎有4个硬件DMA 通道, SHA/HMAC 引擎有1个硬件DMA 通道.

引擎	DMA 使能位
AES	DMAEN (CRYPTO_AES_CTL[7])
TDES/DES	DMAEN (CRYPTO_TDES_CTL[7])
SHA/HMAC	DMAEN (CRYPTO_HMAC_CTL[7])
ECC	DMAEN (CRYPTO_ECC_CTL[7])

表 6.36-2 DMA 使能位列表

### DMA 串流模式

当 SRAM 内存不够使用, 或者是其他周边装置等待处理的情况下, 可以采用分次串流处理的方式来完成加解密操作。在 DMA 串流模式下, 不需要一次完成加密操作, 应用程序可以加解密一段数据, 更新 DMA 来源、目的地址、及数据量计数, 然后再接续加解密操作。过程中, AES硬件会记住当前状态, 应用程序可以加解密一段数据, 去服务其他周边装置, 然后再回来继续加解密操作, 直到全部数据处理完成。另一个好处是, 内存的需求也会大幅度地降低。

引擎	DMA 串流位
AES	DMACSCAD (CRYPTO_AES_CTL[6])
TDES/DES	DMACSCAD (CRYPTO_TDES_CTL[6])

表 6.36-3 DMA 串流位表

### 无 DMA 模式

输入数据量比较小的情况, DMA模式是不可取的。这种非DMA模式可以减少对引擎的处理时间, 因为没有DMA相关的寄存器需要配置, 且没有引入DMA逻辑的延迟。输入数据通过写数据输入寄存器传入到加密引擎。

### 通道扩展模式

在这种模式下, 在AES 或 DES/TDES模式, 多个虚拟通道用四个DMA通道的其中一个可行的。总的通道数可以超过四个DMA 通道的限制。来自反馈寄存器 (CRYPTO\_AES\_FDBCKx, CRYPTO\_TDES\_FDBCKH, 和 CRYPTO\_TDES\_FDBCKL) 的中间数据应该暂时储存在数据SRAM。并切换到另一个引擎操作的配置设置, 包括操作模式, 加密/解密, 密钥, 密钥大小, IV, 和其他参数。一旦切换回来, 来自反馈寄存器的中间数据应写入初始向量 (CRYPTO\_AESn\_IVx, CRYPTO\_TDESx\_IVH, 和 CRYPTO\_TDESx\_IVL) 用于引擎带初始配置设定继续操作。请注, 在ECB模式, 没有必要将中间数据从反馈寄存器移到IV。

引擎	通道选择位
AES	CHANNEL (CRYPTO_AES_CTL[25,24])

TDES/DES

CHANNEL (CRYPTO\_TDES\_CTL[25,24])

表 6.36-4 通道选择位表

## 6.36.5.2 PRNG (伪随机数生成器)

PRNG 框图见 图 6.36-2. 内核支持 64 位, 128 位, 192 位, 和 256 位随机数的生成, 由 KEYSZ(CRYPTO\_PRNG\_CTL[3:2] )配置。

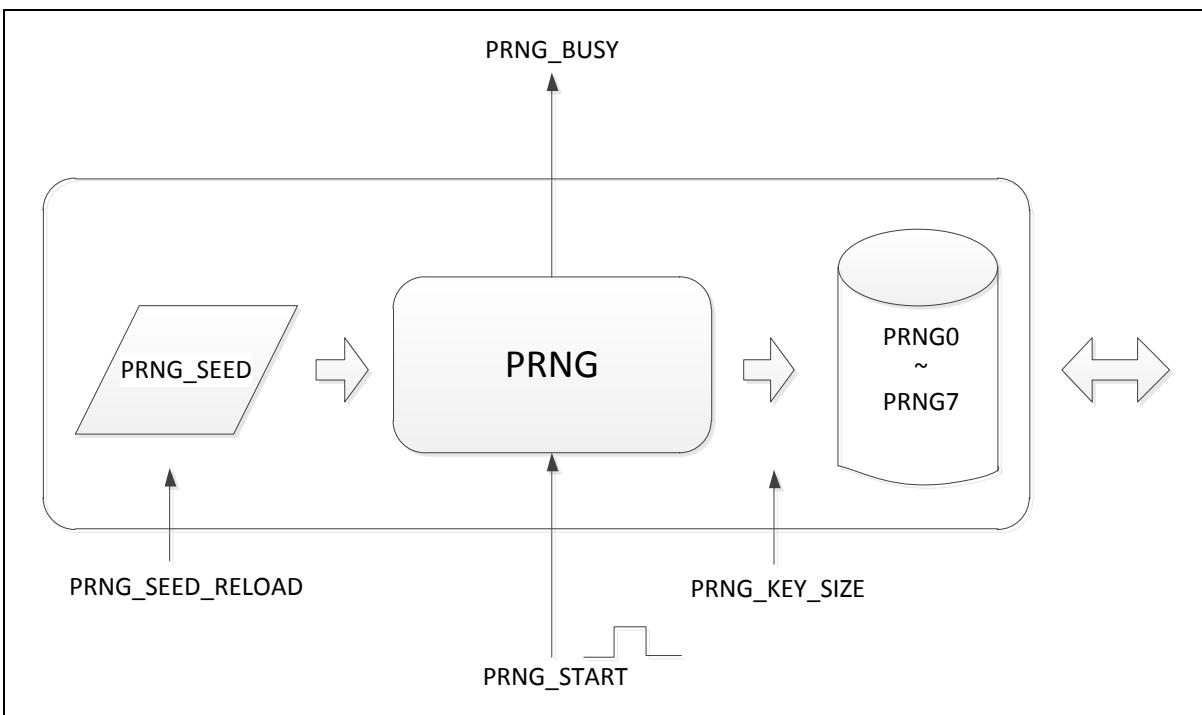


图 6.36-2 PRNG 功能框图

得到一个伪随机数的编程步骤如下描述

1. 检查 BUSY(CRYPTO\_PRNG\_CTL[8]) 直到它变 0.
2. 初始化 PRNG 参数. 配置 KEYSZ (CRYPTO\_PRNG\_CTL[3:2]), 写一个随机种子到 CRYPTO\_PRNG\_SEED. 注意CRYPTO\_PRNG\_SEED 需要初始化, 因为芯片上电并不会初始化这里.
3. 配置 PRNG 控制寄存器 CRYPTO\_PRNG\_CTL 中的密钥大小(KEYSZ), 种子重加载 (SEEDRLD), 和 PRNG 启动(START).
4. 软件检查 BUSY(CRYPTO\_PRNG\_CTL[8]) 直到变成 0, 或是等待 PRNGIF (CRYPTO\_INTSTS[16]) (必须使能 PRNGIEN (CRYPTO\_INTEN[16])). 然后软件从 CRYPTO\_PRNG\_KEY0 ~ CRYPTO\_PRNG\_KEY7中读出产生的随机数 (KEY)

## 6.36.5.3 AES (高级加密标准)

## 电子密码本模式

电子密码本 (ECB) 模式是一种保密模式, 特征是一个固定的密文块分配给每个明文块, 对于一个给定的密匙。它与码本中的码字分配类似。ECB 加密, 每一组明文块都直接独立地应用到了正向密码函数

$CIPH_k$ 。产生的序列输出块是密文。**ECB**解密，每一组密文块都直接独立地应用到了反向密码函数 $CIPH^{-1}_k$ 。产生的序列的输出块是明文。

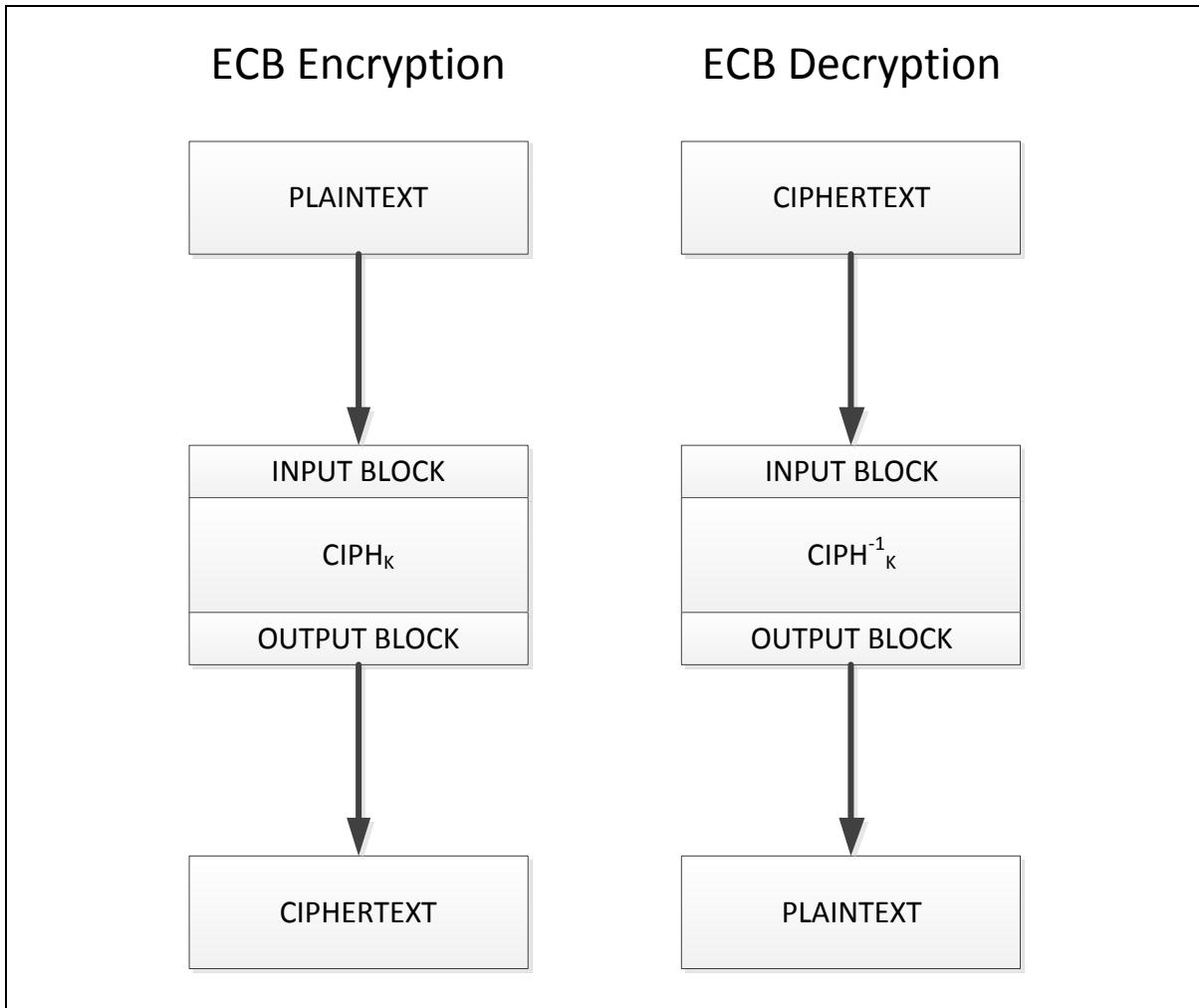


图 6.36-3 电子密码本模式

在**ECB**模式，任何给定的明文块总是在一个给定的密匙下被加码成密文块的。如果在特定的应用中这个属性是不可行的，则不应该使用**ECB**模式。

#### 密码块链模式

密码块链（**CBC**）模式是一种保密模式，它的加密处理特点结合明文块链和先前的密文块。**CBC**模式需要一个初始化向量（**IV**）来结合第一个明文块。**IV**不需要保密，但它必须是不可预测的。

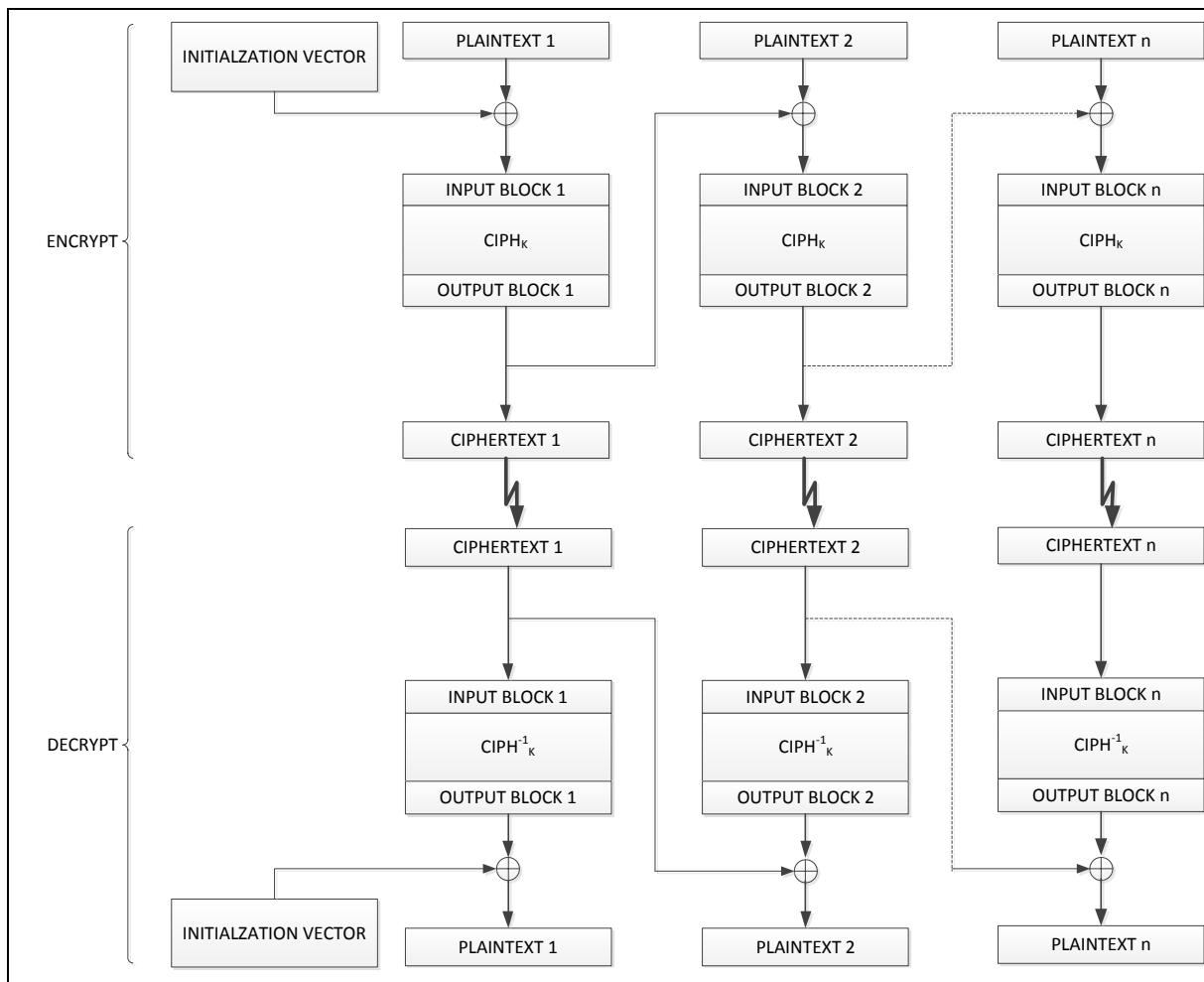


图 6.36-4 密码块链模式

### 密码反馈模式

密码反馈（CFB）模式是一种保密模式，特征是连续密文段反馈到了正向密码输入块来生成输出块，这是与明文进行异或运算产生密文，反之亦然。CFB模式需要一个IV作为初始输入块，IV不需要保密，但它必须是不可预测的。CFB模式下AES仅支持128bit段长度

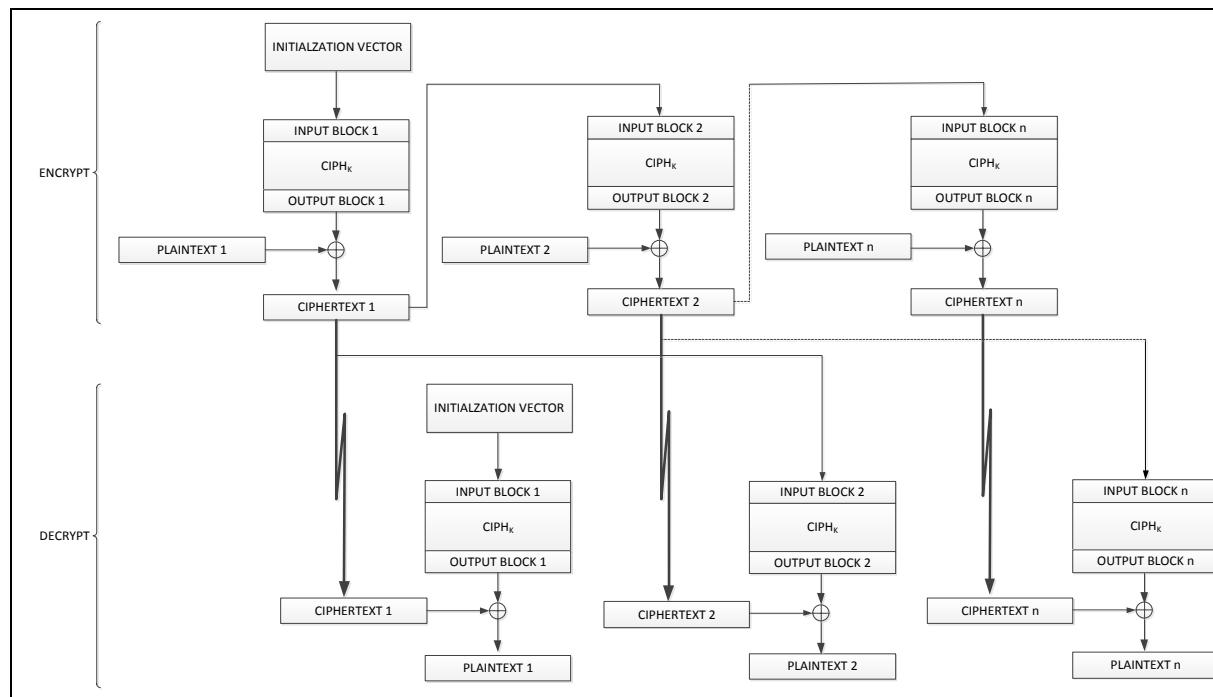


图 6.36-5 密码反馈模式

### 输出反馈模式

输出反馈（OFB）模式是一种保密模式，特征是正向密码迭代一个IV来生产一个输出块的序列。这是与明文进行异或运算产生密文，反之亦然。OFB模式要求IV是一个唯一数，即在一个给定的密匙下，对于该模式的每一次执行IV必须是唯一的。

在一个给定的密匙下，对于已加密的每一个消息，OFB模式需要一个唯一的IV。如果违反了这一要求，相同的IV用于多条消息的加密，那么这些消息的机密性可能会受到损害。在给定密钥下，对于一个消息的加密，如果到正向密码函数的任何输入块被指定作为另一个消息加密的IV，则保密性可能同样被损害。

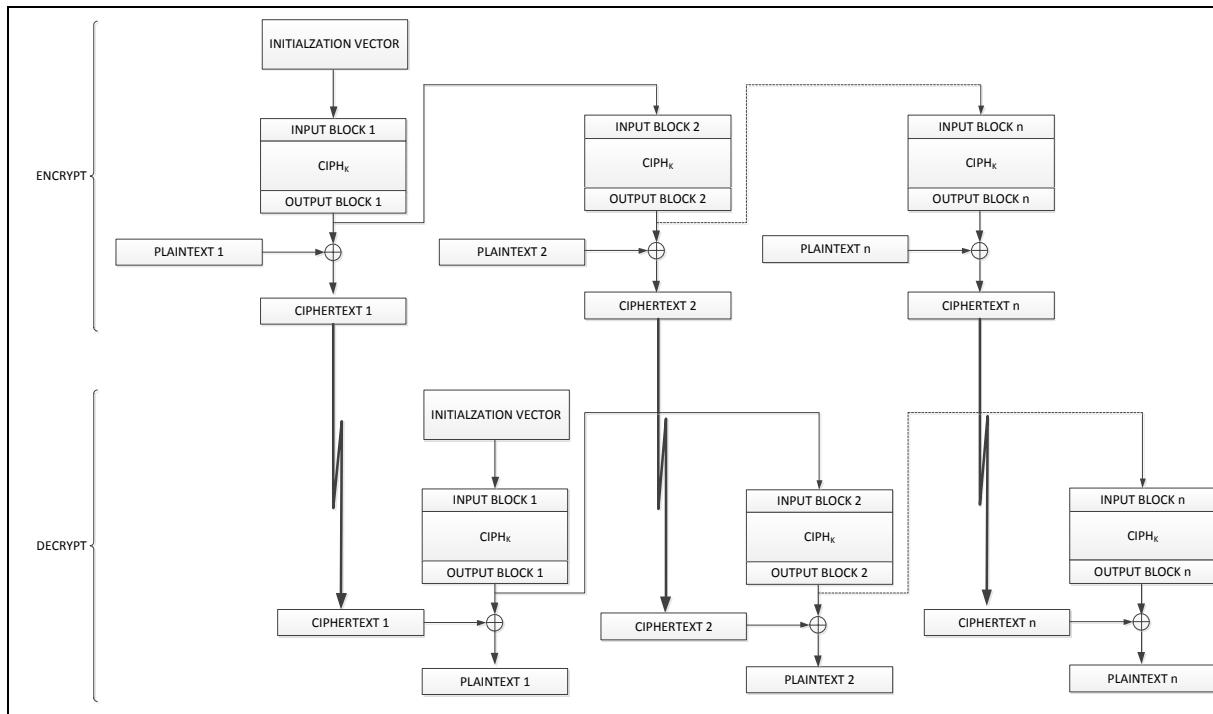


图 6.36-6 输出反馈模式

### 计数器模式

计数器（CTR）模式是一种保密模式，特点是正向密码应用到输入块组，叫做计数器，来产生一个输出块序列，这用于和明文异或运算来产生密文，反之亦然。计数器序列必须有这样的属性，序列中每一块不同于每一个其他块。此条件并不局限于单个消息，在给定的密钥下所有被加密的消息，所有的计数器必须是不同的。

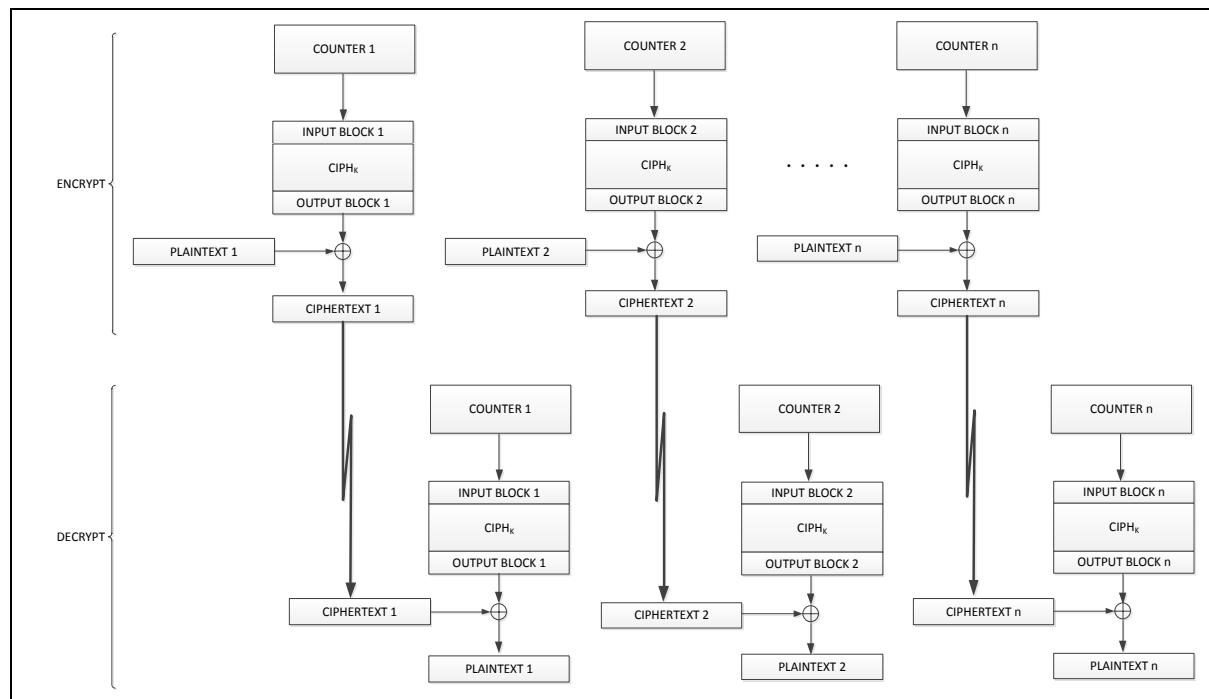


图 6.36-7 计数器模式

**CBC 密文窃取 1 模式(CBC-CS1)**

图 6.36-8 展示  $P_n^*$  为局部块的情况下 CBC-CS1-Encrypt 算法。加解密加速器会添加 0 到  $P_n^*$  组成完整的  $P_n$  块。

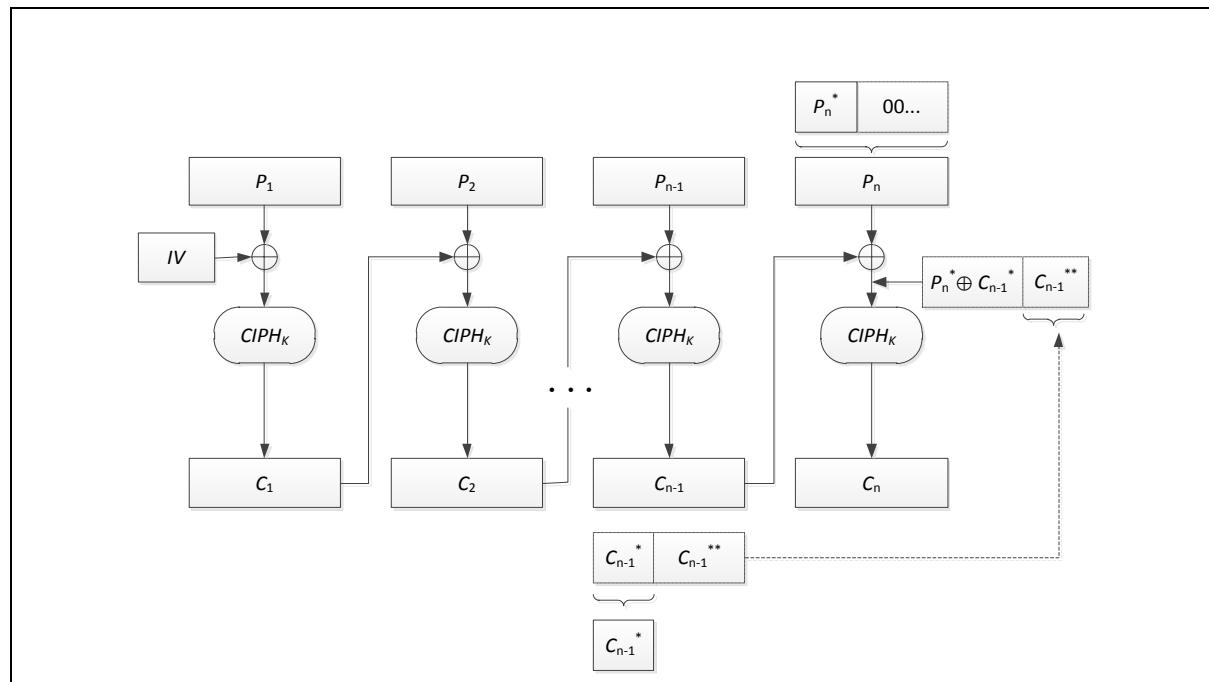


图 6.36-8 CBC-CS1 加密

图 6.36-9 列出  $C_{n-1}^*$  是局部块的情况下 CBC-CS1-Decrypt 算法

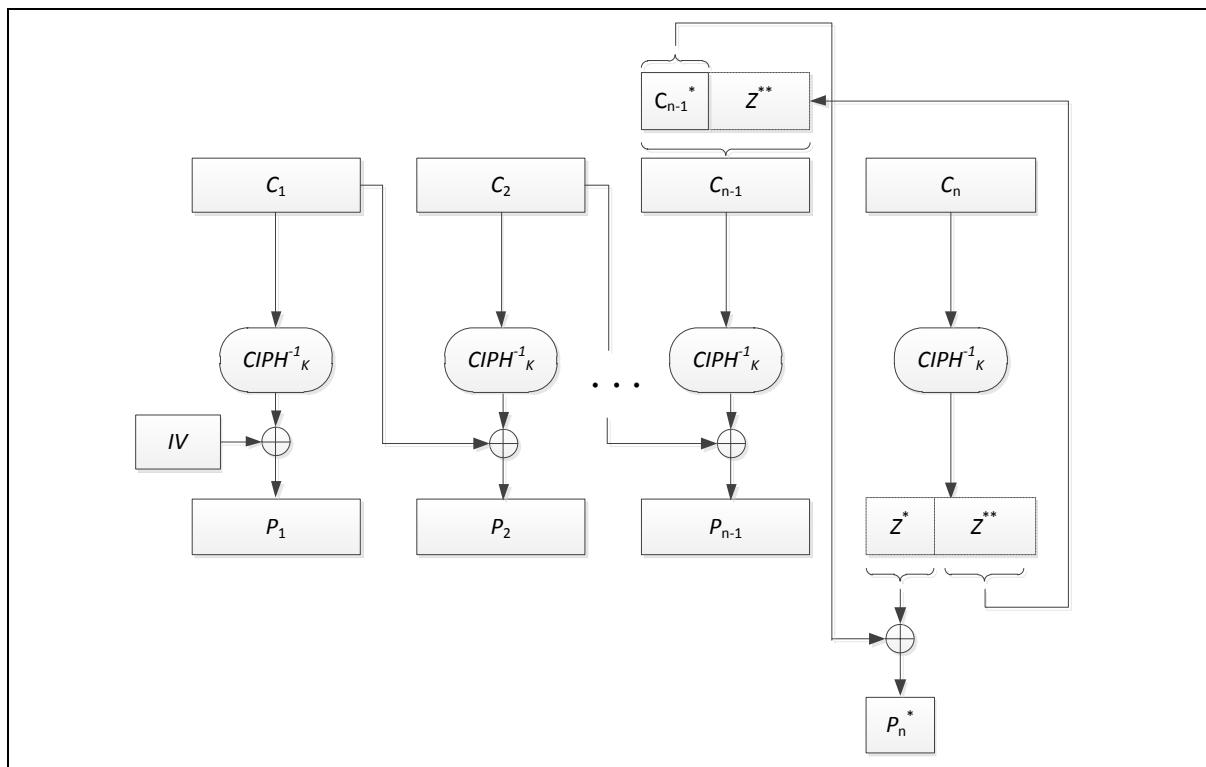


图 6.36-9 CBC-CS1 解密

**CBC 密文窃取 2 模式(CBC-CS2)**

当  $P_n^*$  是局部块, 那么 CBC-CS2-Encrypt 与 CBC-CS1-Encrypt 区别仅在于  $C_{n-1}^*$  和  $C_n$  的排序不同.

**CBC 密文窃取 3 模式(CBC-CS3)**

$C_{n-1}^*$  和  $C_n$  是无条件交换, 即使当  $C_{n-1}^*$  是一个完整块; 那么, CBC-CS不是严格的CBC模式的延伸. 在其他情况下,如果  $C_{n-1}^*$  是一个非空的局部块, CBC-CS3-Encrypt 和 CBC-CS2-Encrypt 相同.

如何编程AES相关的寄存器, 请参考下面编程步骤.

**AES DMA 模式编程流程**

1. 写 1 到 AESIEN(CRYPTO\_INTEN [0]) 使能AES 中断.
2. 从四个DMA 通道中选择一个通道.
3. 编程 AES 密匙到寄存器CRYPTO\_AESn\_KEY0 ~ CRYPTO\_AESn\_KEY7. (这里的n是选择的通道号)
4. 编程初始化向量到寄存器CRYPTO\_AESn\_IV0 ~ CRYPTO\_AESn\_IV3.
5. 编程 DMA 源地址到寄存器CRYPTO\_AESn\_SADDR.
6. 编程 DMA 目的地地址到寄存器CRYPTO\_AESn\_DADDR.
7. 编程 DMA 传输字节数到寄存器CRYPTO\_AESn\_CNT.
8. 配置 AES 控制寄存器CRYPTO\_AES\_CTL 来保护密钥, 选择通道, 加密/解密, 操作模式, DMA模式, 密钥长度和DMA输入/输出交换。
9. 写选择的DMA字节数个输入数据到DMA源地址

10. 写 1 到 START(CRYPTO\_AES\_CTL[0]) 开始 AES 加密/解密.
11. 等待AES 中断标志AESIF(CRYPTO\_INTSTS[0]) 置位.
12. 从DMA目的地址读取选择的DMA字节数个数据。
13. 如果使能了DMACSCAD (CRYPTO\_AES\_CTL[6]), 重复步骤9 到步骤12 直到所有的数据处理完.

#### AES 非-DMA 模式编程流程

1. 写 1 到AESIEN(CRYPTO\_INTEN[0]) 使能AES 中断.
2. 编程 AES 密匙到寄存器CRYPTO\_AESn\_KEY0 ~ CRYPTO\_AESn\_KEY7. (这里的n是选择的通道号)
3. 编程初始化向量到寄存器CRYPTO\_AESn\_IV0 ~ CRYPTO\_AESn\_IV3.
4. 配置 AES 控制寄存器CRYPTO\_AES\_CTL 来保护密钥, 选择通道, 加密/解密, 操作模式和密匙长度。
5. 写 1 到 START(CRYPTO\_AES\_CTL[0]) 来开始 AES 加密/解密.
6. 查询 INBUFFULL(CRYPTO\_AES\_STS[9]) 和 OUTBUFEMPTY(CRYPTO\_AES\_STS[16]). 如果 INBUFFULL(CRYPTO\_AES\_STS[9]) 是 0, 写 32 位输入数据到CRYPTO\_AES\_DATIN, 如果 OUTBUFEMPTY(CRYPTO\_AES\_STS[16]) 是 0, 从CRYPTO\_AES\_DATOUT读取 32 位数据。
7. 重复 步骤 6 直到 128 位数据 (16 字节) 写入AES引擎或从AES引擎读出。
8. 写 1 到 DMALAST(CRYPTO\_AES\_CTL[5]).
9. 如果当前操作是最后的操作, 那么最后需要往寄存器CRYPTO\_AES\_CNT中写入数据字节计数
10. 重复步骤 6 到步骤 9 直到所有的数据处理完.

#### 6.36.5.4 DES/TDES (数据加密标准 / 三重DES)

FIPS 46-3 指定两种保密算法, 数据加密标准 (DES) 和三重数据加密算法 (TDEA)。密码引擎支持FIPS 46-3, 加密和解密, 和 ECB, CBC, CFB, OFB 和 CTR 模式.

#### TDES DMA 模式编程流程

1. 写 1 到 TDESIEN(CRYPTO\_INTEN[8]) 来使能 TDES 中断
2. 检查 TDES 引擎是否在空闲状态 引擎是否在空闲状态 引擎是否在空闲状态 , 即 BUSY(CRYPTO\_TDES\_STS[0]) 要是 0..
3. 编程 TDES 密匙到寄存器 CRYPTO\_TDESn\_KEY1H, CRYPTO\_TDESn\_KEY1L, CRYPTO\_TDESn\_KEY2H, CRYPTO\_TDESn\_KEY2L, CRYPTO\_TDESn\_KEY3H, 和 CRYPTO\_TDESn\_KEY3L. (这里的 n是所选的通道号 是所选的通道号 是所选的通道号 )
4. 编程初始化向量到寄存器 CRYPTO\_TDESn\_IVH 和 CRYPTO\_TDESn\_IVL....
5. 编程 DMA 源地址到寄存器 CRYPTO\_TDESn\_SA.
6. 编程 DMA 目的地址到寄存器 CRYPTO\_TDESn\_DA.
7. 编程 DMA 传输字节数到寄存器CRYPTO\_TDESn\_CNT.

8. 配置 TDES 控制寄存器 CRYPTO\_TDES\_CTL 设置通道使能(CHANNEL), 加/解密(ENCRYPTO), 操作模式(OPMODE), DMA 模式(DMAEN), TDES 密钥(3KEYS), 和 DMA 输入输出交换(INSWAP/OUTSWAP).
9. 写选择的DMA字节数个输入数据到DMA源地址.
- 10.写 1 到 START(CRYPTO\_TDES\_CTL[0]) 来开始 TDES 加密 /解密.
- 11.等待 TDES 中断标志 TDESIF(CRYPTO\_INTSTS[8]) 置位.
- 12.从DMA目的地址读取选择的DMA字节数个数据.
- 13.重复步骤 9 步骤 12 直到所有的数据处理完.

#### TDES 非-DMA 模式编程流程

1. 写1 到TDESIEN(CRYPTO\_INTEN[8]) 来使能 TDES 中断.
2. 检查TDES 引擎是否在空闲状态, 即BUSY(CRYPTO\_TDES\_STS[0]) 要是0.
3. 编程TDES 密匙到寄存器CRYPTO\_TDESn\_KEY1H, CRYPTO\_TDESn\_KEY1L, CRYPTO\_TDESn\_KEY2H, CRYPTO\_TDESn\_KEY2L, CRYPTO\_TDESn\_KEY3H, 和 CRYPTO\_TDESn\_KEY3L. (这里的n是所选的通道号)
4. 编程初始化向量到寄存器CRYPTO\_TDESn\_IVH 和CRYPTO\_TDESn\_IVL.
5. 配置 TDES 控制寄存器 CRYPTO\_TDES\_CTL 来选择通道, 加密/解密, 操作模式和TDES 密匙。
6. 写 1 到 START(CRYPTO\_TDES\_CTL[0]) 来开始 TDES 加密/解密.
- 7.
8. 查询 INBUFFULL(CRYPTO\_TDES\_STS[9]) 和 OUTBUFEMPTY(CRYPTO\_TDES\_STS[16]). 如果 INBUFFULL(CRYPTO\_TDES\_STS[9]) 是 0, 写 32 位输入数据到 CRYPTO\_TDES\_DATIN, 如果OUTBUFEMPTY(CRYPTO\_TDES\_STS[16]) 是 0, 从 CRYPTO\_TDES\_DATOUT读取32位数据。
9. 重复步骤 7 直到 64 位数据 (8 字节) 写入 TDES 引擎和从TDES 引擎读出。
- 10.写 1 到 DMALAST(CRYPTO\_TDES\_CTL[5]).
- 11.如果当前操作是最后的操作, 那么最后需要往寄存器CRYPTO\_TDES\_CNT中写入数据字节计数
- 12.重复步骤 7 到步骤 10直到所有数据处理完.

#### 6.36.5.5 SHA (安全 Hash 算法)

用户可以参考下面步骤来理解如何编程SHA相关寄存器.

#### SHA DMA 模式编程流程

1. 写1到 HMACIEN(CRYPTO\_INTEN[24]) 使能SHA/HMAC 中断.
2. 配置 SHA/HMAC控制寄存器 CRYPTO\_HMAC\_CTL 来配置 SHA/HMAC 引擎输入输出数据交换 (INSWAP/OUTSWAP), DMA 模式(DMAEN), 和 SHA 工作模式(OPMODE). 清除 HMACEN(CRYPTO\_HMAC\_CTL[4]) 来选择SHA 模式.

3. 填写 DMA 源地址寄存器 CRYPTO\_HMAC\_SADDR.
4. 填写 DMA 字节数寄存器 CRYPTO\_HMAC\_DMACNT.
5. 向 DMA 源地址写入选定的DMA字节数的数据.
6. 写1到START(CRYPTO\_HMAC\_CTL[0]) 开始 SHA 加密.
7. 等待 SHA 终端标志 HMACIF(CRYPTO\_INTSTS[24]) 置位.
8. 读输出数据 (SHA160: CRYPTO\_HMAC\_DGST0 ~ CRYPTO\_HMAC\_DGST4, SHA224: CRYPTO\_HMAC\_DGST0 ~ CRYPTO\_HMAC\_DGST6, SHA256: CRYPTO\_HMAC\_DGST0 ~ CRYPTO\_HMAC\_DGST7, SHA384: CRYPTO\_HMAC\_DGST0 ~ CRYPTO\_HMAC\_DGST11, SHA512: CRYPTO\_HMAC\_DGST0 ~ CRYPTO\_HMAC\_DGST15).

注意: KEYCNT(CRYPTO\_HMAC\_KEYCNT[31:0]) 保持 SHA/HMAC 引擎操作的种子的字节数.

#### SHA 非-DMA 模式编程流程

1. 配置 SHA/HMAC 控制寄存器 CRYPTO\_HMAC\_CTL 配置 SHA/HMAC 引擎输入输出数据交换 (INSWAP/OUTSWAP), DMA 模式(DMAEN), 和 SHA 工作模式(OPMODE). 清除 HMACEN(CRYPTO\_HMAC\_CTL[4]) 来选择 SHA 模式.
2. 如果是最后的输入字, 设置 DMALAST(CRYPTO\_HMAC\_CTL[5]).
3. 写 1 到 START(CRYPTO\_HMAC\_CTL[0]) 开始 SHA 加密.
4. 等待 SHA 数据输入请求 DATINREQ(CRYPTO\_HMAC\_STS[16]) 置位.
5. 写一笔数据到 CRYPTO\_HMAC\_DATIN.
6. 重复步骤 2 到 5 直到所有的输入数据写入到 SHA 引擎.
7. 等待 BUSY (CRYPTO\_HMAC\_STS[0]) 清除.
8. 读取输出数据 (SHA160: CRYPTO\_HMAC\_DGST0 ~ CRYPTO\_HMAC\_DGST4, SHA224: CRYPTO\_HMAC\_DGST0 ~ CRYPTO\_HMAC\_DGST6, SHA256: CRYPTO\_HMAC\_DGST0 ~ CRYPTO\_HMAC\_DGST7, SHA384: CRYPTO\_HMAC\_DGST0 ~ CRYPTO\_HMAC\_DGST11, SHA512: CRYPTO\_HMAC\_DGST0 ~ CRYPTO\_HMAC\_DGST15).

注意: KEYCNT(CRYPTO\_HMAC\_KEYCNT[31:0]) 保持 SHA/HMAC 引擎操作的种子的字节数.

#### 6.36.5.6 HMAC (*Keyed-Hash*消息认证码)

**Keyed-Hash** 消息认证码是一个特殊的架构来用于计算涉及密码散列函数与秘密加密密钥结合的消息验证码的特定结构。任意加密散列函数，例如SHA-1，可能用于计算一个 HMAC；由此产生的MAC 相应的称为 **HMAC-SHA1**

用户可以参考下面的步骤理解如何编写 HMAC 相关寄存器。

#### HMAC DMA 模式编程流程

1. 写 1 到 HMACIEN(CRYPTO\_INTEN[24]) 使能 HMAC 中断.

2. 配置 SHA/HMAC 控制寄存器 CRYPTO\_HMAC\_CTL 设置 HMAC 引擎输入输出数据交换 (INSWAP/OUTSWAP), DMA 模式(DMAEN), 和 HMAC 操作模式(OPMODE). 设置 HMACEN(CRYPTO\_HMAC\_CTL[4]) 选择 HMAC 模式
3. 填写 DMA 源地址寄存器 CRYPTO\_HMAC\_SADDR.
4. 填写 DMA 字节数寄存器 CRYPTO\_HMAC\_DMACNT.
5. 向 DMA 源地址写入选择DMA字节数的数据.
6. 写 1 到 START(CRYPTO\_HMAC\_CTL[0]) 开始HMAC 加密.
7. 等待 HMAC 中断标识 HMACIF(CRYPTO\_INTSTS[24]) 置位
8. 读取输出数据 (HMAC-SHA160: CRYPTO\_HMAC\_DGST0 ~ CRYPTO\_HMAC\_DGST4, HMAC-SHA224: CRYPTO\_HMAC\_DGST0 ~ CRYPTO\_HMAC\_DGST6, HMAC-SHA256: CRYPTO\_HMAC\_DGST0 ~ CRYPTO\_HMAC\_DGST7, HMAC-SHA384: CRYPTO\_HMAC\_DGST0 ~ CRYPTO\_HMAC\_DGST11, HMAC-SHA512: CRYPTO\_HMAC\_DGST0 ~ CRYPTO\_HMAC\_DGST15).

注意: KEYCNT(CRYPTO\_HMAC\_KEYCNT[31:0]) 保持SHA/HMAC 引擎操作的种子的字节数.

#### HMAC 非-DMA 模式编程流程

1. 配置 SHA/HMAC 控制寄存器 CRYPTO\_HMAC\_CTL 设置 HMAC 引擎输入输出数据交换 (INSWAP/OUTSWAP), 和 HMAC 操作模式(OPMODE). 设置 HMACEN(CRYPTO\_HMAC\_CTL[4]) 选择 HMAC 模式.
2. 如果是最后的输入字, 设置DMALAST(CRYPTO\_HMAC\_CTL[5]).
3. 写 1 到 START(CRYPTO\_HMAC\_CTL[0]) 开始 HMAC 加密.
4. 等待 HMAC 数据输入请求 DATINREQ(CRYPTO\_HMAC\_STS[16]) 置位.
5. 写一笔数据到 CRYPTO\_HMAC\_DATIN.
6. 重复步骤 2 到 5 直到所有的输入数据写入SHA 引擎.
7. 等待 BUSY (CRYPTO\_HMAC\_STS[0]) 清除.
8. 读到输出数据 (HMAC-SHA160: CRYPTO\_HMAC\_DGST0 ~ CRYPTO\_HMAC\_DGST4, HMAC-SHA224: CRYPTO\_HMAC\_DGST0 ~ CRYPTO\_HMAC\_DGST6, HMAC-SHA256: CRYPTO\_HMAC\_DGST0 ~ CRYPTO\_HMAC\_DGST7, HMAC-SHA384: CRYPTO\_HMAC\_DGST0 ~ CRYPTO\_HMAC\_DGST11, HMAC-SHA512: CRYPTO\_HMAC\_DGST0 ~ CRYPTO\_HMAC\_DGST15).
- 9.
10. 注意: KEYCNT(CRYPTO\_HMAC\_KEYCNT[31:0]) 保持SHA/HMAC 引擎操作的种子的字节数.

#### 6.36.5.7 ECC (椭圆曲线密码体制)

椭圆曲线密码体制 (ECC) 是一种著名的公钥密码体制的方法。近年来，许多协议和应用都利用有限域上椭圆曲线的代数循环群特征构造密码系统. 椭圆曲线的所有点都遵循椭圆曲线的公式： $y^2 \equiv x^3 + Ax + B \pmod{N}$  在  $GF(p)$  和  $y^2 + x^3 \equiv x^3 + Ax^2 + B \pmod{N}$  在  $GF(2^m)$ . 图 7.35-10 显示 ECC 应用的主要层次结构图。经常出现的参数和相应的寄存器如表 6.36-5 所示。

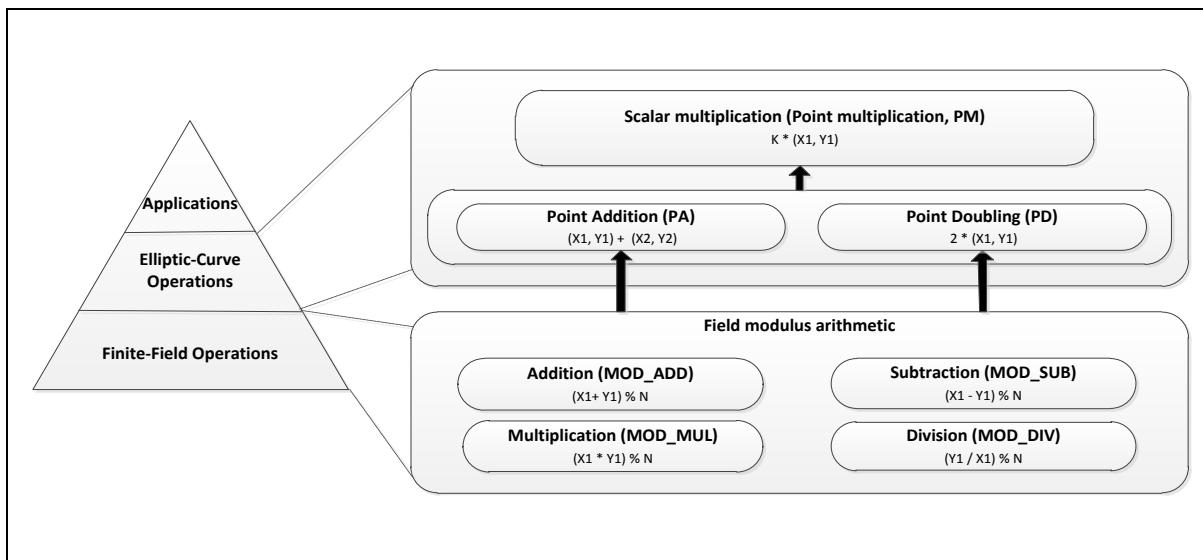


图 6.36-10 ECC 的主要层次结构图

参数	描述	相应寄存器
X1	点 1 的 X 坐标	CRYPTO_ECC_X1_00~ CRYPTO_ECC_X1_17
Y1	点 1 的 Y 坐标	CRYPTO_ECC_Y1_00~ CRYPTO_ECC_Y1_17
X2	点 2 的 X 坐标	CRYPTO_ECC_X2_00~ CRYPTO_ECC_X2_17
Y2	点 2 的 Y 坐标	CRYPTO_ECC_Y2_00~ CRYPTO_ECC_Y2_17
A	曲线参数 A	CRYPTO_ECC_A_00~ CRYPTO_ECC_A_17
B	曲线参数 B	CRYPTO_ECC_B_00~ CRYPTO_ECC_B_17
N	曲线参数 N	CRYPTO_ECC_N_00~ CRYPTO_ECC_N_17
M	曲线长度	CRYPTO_ECC_CTL[31:22]
K	标量常数	CRYPTO_ECC_K_00~ CRYPTO_ECC_K_17

表 6.36-5 ECC 参数和相应寄存器列表

标量乘法（点乘）是ECC应用中的核心运算。标量乘法的计算由点加法和点乘运算组成。此外，在点加法和倍运算公式中存在许多有限域模运算。为了加速 ECC 应用，我们提出了一个椭圆曲线加密加速器，可以处理不仅在  $GF(p)$  和  $GF(2^m)$  的三点操作也支持在  $GF(p)$  的四模操作。在开始ECC 加速器之前，用户必须提供所需的ECC操作输入数据包括点坐标 ( $X1, Y1, X2, Y2$ )，曲线参数 ( $A, B, N, M$ ) 和标量数据 ( $K$ ) 如 表 6.36-6 .标记“√”表示输入数据对这个操作是必须的。在我们的ECC加速器的输入数据和相应的寄存器的详细定义显示在下一节寄存器映射

在 ECC 加速器完成之后，所有的点操作会产生一个输出点，包含 CRYPTO\_ECC\_X1\_00 到 CRYPTO\_ECC\_X1\_17 寄存器的 X 坐标和 CRYPTO\_ECC\_Y1\_00 到 CRYPTO\_ECC\_Y1\_17 寄存器的 Y 坐标。在所有的模操作中，ECC 加速器只产生结果到寄存器 CRYPTO\_ECC\_X1\_00 搭配 CRYPTO\_ECC\_X1\_17.

操作	PM	PA	PD	MOD_DIV	MOD_MUL	MOD_ADD	MOD_SUB
ECCOP[1:0]	00	10	11	01	01	01	01
MODOP[1:0]	XX	XX	XX	00	01	10	11
X1	√	√	√	√	√	√	√
Y1	√	√	√	√	√	√	√
X2		√					
Y2		√					
A	√	√	√				
B	√		√				
N	√	√	√	√	√	√	√
M	√	√	√		√		
K	√						

表 6.36-6 各种操作所需的输入数据

用户可以参考以下步骤来了解如何编写ECC相关寄存器。

#### ECC DMA 模式编程流程

1. 写 1 到 ECCIEN(CRYPTO\_INTEN[22]) 使能 ECC 中断.
2. 填写 DMA 源地址寄存器 CRYPTO\_ECC\_SADDR.
3. 填写 DMA 目的地址寄存器 CRYPTO\_ECC\_DADDR.
4. 填写 DMA 字数寄存器 CRYPTO\_ECC\_WORDCNT.
5. 向开始寄存器地址填写所有的输入数据表 6.36-6 展示我们要更新到寄存器 CRYPTO\_ECC\_STARTREG 的内容.
6. 向 DMA 源地址寄存器写入设定 DMA 字数的数据
7. 配置 ECC 控制寄存器 CRYPTO\_ECC\_CTL 配置 ECC 加速器, 例如 ECC 加速器开始信号, DMA 模式使能信号, 域选择, 点操作模式, 模操作模式, 所有输入数据寄存器的控制信号和椭圆曲线的种子长度.
8. 等待 ECC 中断标志 ECCIF(CRYPTO\_INTSTS[22]) 置位.
9. 读取输出数据, 清除 ECC 中断标志 ECCIF.

#### ECC 非-DMA 模式编程流程

1. 依据表 6.36-6 写数据到相应的寄存器, 比如 CURVEA, CURVEB, CURVEN, SCALAR
2. 配置 ECC 控制寄存器 CRYPTO\_ECC\_CTL 配置 ECC 加速器, 例如 ECC 加速器开始信号, DMA 模式使能信号, 域选择, 点操作模式, 模操作模式, 所有输入数据寄存器的控制信号和椭圆曲线的密钥长度
3. 等待 BUSY (CRYPTO\_ECC\_STS[0]) 位清 0.
4. 读取输出数据, 清除 ECC 中断标志 ECCIF.

**ECC 加速器的一些注意事项如下**

1. ECC 加速器支持的密钥长度从 163 到 256 位.
2. 所有的输入输出数据都必须是正数 (如果输入数据是负数, 需要加上N).
3. GF( $2^m$ ) 不可约多项式必须采用HP里最小的, 请参考表 6.36-7

	<b>2,1</b>	<b>3,1</b>	<b>4,1</b>	<b>5,2</b>
6,1	7,1	8,4,3,1	9,1	10,3
11,2	12,3	13,4,3,1	14,5	15,1
16,5,3,1	17,3	18,3	19,5,2,1	20,3
21,2	22,1	23,5	24,4,3,1	25,3
26,4,3,1	27,5,2,1	28,1	29,2	30,1
31,3	32,7,3,2	33,10	34,7	35,2
36,9	37,6,4,1	38,6,5,1	39,4	40,5,4,3
41,3	42,7	43,6,4,3	44,5	45,4,3,1
46,1	47,5	48,5,3,2	49,9	50,4,3,2
51,6,3,1	52,3	53,6,2,1	54,9	55,7
56,7,4,2	57,4	58,19	59,7,4,2	60,1
61,5,2,1	62,29	63,1	64,4,3,1	65,18
66,3	67,5,2,1	68,9	69,6,5,2	70,5,3,1
71,6	72,10,9,3	73,25	74,35	75,6,3,1
76,21	77,6,5,2	78,6,5,3	79,9	80,9,4,2
81,4	82,8,3,1	83,7,4,2	84,5	85,8,2,1
86,21	87,13	88,7,6,2	89,38	90,27
91,8,5,1	92,21	93,2	94,21	95,11
96,10,9,6	97,6	98,11	99,6,3,1	100,15
101,7,6,1	102,29	103,9	104,4,3,1	105,4
106,15	107,9,7,4	108,17	109,5,4,2	110,33
111,10	112,5,4,3	113,9	114,5,3,2	115,8,7,5
116,4,2,1	117,5,2,1	118,33	119,8	120,4,3,1
121,18	122,6,2,1	123,2	124,19	125,7,6,5
126,21	127,1	128,7,2,1	129,5	130,3
131,8,3,2	132,17	133,9,8,2	134,57	135,11
136,5,3,2	137,21	138,8,7,1	139,8,5,3	140,15
141,10,4,1	142,21	143,5,3,2	144,7,4,2	145,52
146,71	147,14	148,27	149,10,9,7	150,53
151,3	152,6,3,2	153,1	154,15	155,62

156,9	157,6,5,2	158,8,6,5	159,31	160,5,3,2
161,18	162,27	163,7,6,3	164,10,8,7	165,9,8,3
166,37	167,6	168,15,3,2	169,34	170,11
171,6,5,2	172,1	173,8,5,2	174,13	175,6
176,11,3,2	177,8	178,31	179,4,2,1	180,3
181,7,6,1	182,81	183,56	184,9,8,7	185,24
186,11	187,7,6,5	188,6,5,2	189,6,5,2	190,8,7,6
191,9	192,7,2,1	193,15	194,87	195,8,3,2
196,3	197,9,4,2	198,9	199,34	200,5,3,2
201,14	202,55	203,8,7,1	204,27	205,9,5,2
206,10,9,5	207,43	208,9,3,1	209,6	210,7
211,11,10,8	212,105	213,6,5,2	214,73	215,23
216,7,3,1	217,45	218,11	219,8,4,1	220,7
221,8,6,2	222,5,4,2	223,33	224,9,8,3	225,32
226,10,7,3	227,10,9,4	228,113	229,10,4,1	230,8,7,6
231,26	232,9,4,2	233,74	234,31	235,9,6,1
236,5	237,7,4,1	238,73	239,36	240,8,5,3
241,70	242,95	243,8,5,1	244,111	245,6,4,1
246,11,2,1	247,82	248,15,14,10	249,35	250,103
251,7,4,2	252,15	253,46	254,7,2,1	255,52
256,10,5,2	257,12	258,71	259,10,6,2	260,15
261,7,6,4	262,9,8,4	263,93	264,9,6,2	265,42
266,47	267,8,6,3	268,25	269,7,6,1	270,53
271,58	272,9,3,2	273,23	274,67	275,11,10,9
276,63	277,12,6,3	278,5	279,5	280,9,5,2
281,93	282,35	283,12,7,5	284,53	285,10,7,5
286,69	287,71	288,11,10,1	289,21	290,5,3,2
291,12,11,5	292,37	293,11,6,1	294,33	295,48
296,7,3,2	297,5	298,11,8,4	299,11,6,4	300,5
301,9,5,2	302,41	303,1	304,11,2,1	305,102
306,7,3,1	307,8,4,2	308,15	309,10,6,4	310,93
311,7,5,3	312,9,7,4	313,79	314,15	315,10,9,1
316,63	317,7,4,2	318,45	319,36	320,4,3,1
321,31	322,67	323,10,3,1	324,51	325,10,5,2
326,10,3,1	327,34	328,8,3,1	329,50	330,99

331,10,6,2	332,89	333,2	334,5,2,1	335,10,7,2
336,7,4,1	337,55	338,4,3,1	339,16,10,7	340,45
341,10,8,6	342,125	343,75	344,7,2,1	345,22
346,63	347,11,10,3	348,103	349,6,5,2	350,53
351,34	352,13,11,6	353,69	354,99	355,6,5,1
356,10,9,7	357,11,10,2	358,57	359,68	360,5,3,2
361,7,4,1	362,63	363,8,5,3	364,9	365,9,6,5
366,29	367,21	368,7,3,2	369,91	370,139
371,8,3,2	372,111	373,8,7,2	374,8,6,5	375,16
376,8,7,5	377,41	378,43	379,10,8,5	380,47
381,5,2,1	382,81	383,90	384,12,3,2	385,6
386,83	387,8,7,1	388,159	389,10,9,5	390,9
391,28	392,13,10,6	393,7	394,135	395,11,6,5
396,25	397,12,7,6	398,7,6,2	399,26	400,5,3,2
401,152	402,171	403,9,8,5	404,65	405,13,8,2
406,141	407,71	408,5,3,2	409,87	410,10,4,3
411,12,10,3	412,147	413,10,7,6	414,13	415,102
416,9,5,2	417,107	418,199	419,15,5,4	420,7
421,5,4,2	422,149	423,25	424,9,7,2	425,12
426,63	427,11,6,5	428,105	429,10,8,7	430,14,6,1
431,120	432,13,4,3	433,33	434,12,11,5	435,12,9,5
436,165	437,6,2,1	438,65	439,49	440,4,3,1
441,7	442,7,5,2	443,10,6,1	444,81	445,7,6,4
446,105	447,73	448,11,6,4	449,134	450,47
451,16,10,1	452,6,5,4	453,15,6,4	454,8,6,1	455,38
456,18,9,6	457,16	458,203	459,13,5,2	460,19
461,7,6,1	462,73	463,93	464,19,18,13	465,31
466,14,11,6	467,11,6,1	468,27	469,9,5,2	470,9
471,1	472,11,3,2	473,200	474,191	475,9,8,4
476,9	477,16,15,7	478,121	479,104	480,15,9,6
481,138	482,9,6,5	483,9,6,4	484,105	485,17,16,6
486,81	487,94	488,4,3,1	489,83	490,219
491,11,6,3	492,7	493,10,5,3	494,17	495,76
496,16,5,2	497,78	498,155	499,11,6,5	500,27
501,5,4,2	502,8,5,4	503,3	504,15,14,6	505,156

506,23	507,13,6,3	508,9	509,8,7,3	510,69
511,10	512,8,5,2	513,26	514,67	515,14,7,4
516,21	517,12,10,2	518,33	519,79	520,15,11,2
521,32	522,39	523,13,6,2	524,167	525,6,4,1
526,97	527,47	528,11,6,2	529,42	530,10,7,3
531,10,5,4	532,1	533,4,3,2	534,161	535,8,6,2
536,7,5,3	537,94	538,195	539,10,5,4	540,9
541,13,10,4	542,8,6,1	543,16	544,8,3,1	545,122
546,8,2,1	547,13,7,4	548,10,5,3	549,16,4,3	550,193
551,135	552,19,16,9	553,39	554,10,8,7	555,10,9,4
556,153	557,7,6,5	558,73	559,34	560,11,9,6
561,71	562,11,4,2	563,14,7,3	564,163	565,11,6,1
566,153	567,28	568,15,7,6	569,77	570,67
571,10,5,2	572,12,8,1	573,10,6,4	574,13	575,146
576,13,4,3	577,25	578,23,22,16	579,12,9,7	580,237

表 6.36-7 轻量级二进制不可约多项式

4. 只有当 START 和 DMAEN (CRYPTO\_ECC\_CTL[0] 和 [7]) 同时为1, ECC DMA 模式才有效.
5. 当 ECC 引擎正在工作 (BUSY是 1 且 DMABUSY (CRYPTO\_ECC\_STS[1]) 是 0), 用户不能修改所有的输入数据寄存器 (CRYPTO\_ECC\_X1\_00 ~ CRYPTO\_ECC\_K\_17).
6. 如果用户想要停止 ECC 加速器, 请配置 STOP (CRYPTO\_ECC\_CTL[1]) 为 1. 注意: 为了避免下次操作的传输错误, BUSY 信号不会立即清除, 直到DMA操作完成
7. 在二进制域不支持模操作.
8. PF的模乘运算和除法运算的输入数据必须小于n。
9. K是私有密钥, 所以这个寄存器只写.

以下是密钥对产生和 ECDSA 应用的方法.

### 密钥对的产生

公用密钥产生方程 :  $Q = dG \pmod{N}$

1. 依据表 6.36-5写曲线参数 A, B, N, 和曲线长度 M 到相应的寄存器
2. 依据表 6.36-5写点G(x, y) 到 X1, Y1 寄存器
3. 依据表 6.36-5写私有密钥 d 到 K 寄存器
4. 设置 ECCOP(CRYPTO\_ECC\_CTL[10:9]) 为 00
5. 设置 FSEL(CRYPTO\_ECC\_CTL[8]) 依据使用的素数域或二进制域的曲线
6. 设置 START(CRYPTO\_ECC\_CTL[0]) 为 1
7. 等待 BUSY(CRYPTO\_ECC\_STS[0])清0

8. 从X1, Y1 寄存器读公共密钥 Q

### 椭圆曲线数字签名算法(ECDSA)

#### ECDSA 签名产生流程:

1. 计算  $e = \text{HASH}(m)$ , 这里 HASH 是密码哈希算法(即 SHA-1)
  - 1) 使用 SHA 计算 e
2. 从[1, n-1]选择随机整数k
  - 1) 注意n是数阶, 不是素数模或不可约多项式函数
3. 计算  $r = x1 \pmod{n}$ , 这里  $(x1, y1) = k * G$ . 如果  $r = 0$ , 调到步骤 2
  - 1) 依据表 6.36-5写入曲线参数 A, B, N 和曲线长度 M 到相应的寄存器
  - 2) 依据表 6.36-5将素数模或不可约多项式函数写到 N 寄存器
  - 3) 依据表 6.36-5写点 G(x, y) 到 X1, Y1 寄存器
  - 4) 依据表 6.36-5写随机整数 k 到 K 寄存器
  - 5) 设置 ECCOP(CRYPTO\_ECC\_CTL[10:9]) 为 00
  - 6) 设置 FSEL(CRYPTO\_ECC\_CTL[8]) 依据使用的素数域或二进制域的曲线
  - 7) 设置 START(CRYPTO\_ECC\_CTL[0]) 为 1
  - 8) 等待 BUSY(CRYPTO\_ECC\_STS[0]) 清0
  - 9) 依据表 6.36-5将曲线顺序和曲线长度写入 N ,M 寄存器
  - 10) 写 0x0 到 Y1 寄存器
  - 11) 设置 ECCOP(CRYPTO\_ECC\_CTL[10:9]) 为 01
  - 12) 设置 MOPOP(CRYPTO\_ECC\_CTL[12:11]) 为 10
  - 13) 设置 START(CRYPTO\_ECC\_CTL[0]) 为 1
  - 14) 等待 BUSY(CRYPTO\_ECC\_STS[0]) 清0
  - 15) 读 X1 寄存器得到 r
4. 计算  $s = k^{-1} \times (e + d \times r) \pmod{n}$ . 如果 s = 0, 调到步骤 2
  - 1) 依据表 6.36-5写曲线顺序到 N 寄存器
  - 2) 依据表 6.36-5写随机整数 k 到 X1 寄存器
  - 3) 设置 ECCOP(CRYPTO\_ECC\_CTL[10:9]) 为 01
  - 4) 设置 MOPOP(CRYPTO\_ECC\_CTL[12:11]) 为 00
  - 5) 设置 START(CRYPTO\_ECC\_CTL[0]) 到 1
  - 6) 等待 BUSY(CRYPTO\_ECC\_STS[0]) 清0
  - 7) 读 X1 寄存器得到  $k^{-1}$
  - 8) 依据表 6.36-5写曲线顺序和曲线长度到 N ,M 寄存器
  - 9) 写 r, d 到 X1, Y1 寄存器

- 10) 设置 ECCOP(CRYPTO\_ECC\_CTL[10:9]) 为 01
  - 11) 设置 MOPOP(CRYPTO\_ECC\_CTL[12:11]) 为 01
  - 12) 设置 START(CRYPTO\_ECC\_CTL[0]) 到 1
  - 13) 等待 BUSY(CRYPTO\_ECC\_STS[0]) 清0
  - 14) 依据表 6.36-5写曲线顺序到N 寄存器
  - 15) 写 e 到 Y1 寄存器
  - 16) 设置 ECCOP(CRYPTO\_ECC\_CTL[10:9]) 到 01
  - 17) 设置MOPOP(CRYPTO\_ECC\_CTL[12:11]) 到 10
  - 18) 设置 START(CRYPTO\_ECC\_CTL[0]) 为 1
  - 19) 等待BUSY(CRYPTO\_ECC\_STS[0]) 清0
  - 20) 依据表 6.36-5写曲线顺序和曲线长度到N ,M 寄存器
  - 21) 写  $k^{-1}$  到 Y1 寄存器
  - 22) 设置 ECCOP(CRYPTO\_ECC\_CTL[10:9]) 为 01
  - 23) 设置 MOPOP(CRYPTO\_ECC\_CTL[12:11]) 为 01
  - 24) 设置 START(CRYPTO\_ECC\_CTL[0]) 为 1
  - 25) 等待 BUSY(CRYPTO\_ECC\_STS[0]) 清0
  - 26) 读 X1 寄存器得到 s
5. 签名是 pair (r, s)

#### ECDSA 签名验证步骤:

1. 校验r 和 s 是 [1, n-1]范围内的整数. 如果不是, 签名无效
  - 1) 计算  $e = \text{HASH}(m)$ , 这里 HASH 签名生成中的哈希算法
2. 用 SHA 来计算 e
3. 计算  $w = s^{-1} \pmod{n}$ 
  - 1) 依据表 6.36-5写曲线顺序到N 寄存器
  - 2) 依据表 6.36-5写 s 到 X1 寄存器
  - 3) 依据表 6.36-5写 0x1 到 Y1寄存器
  - 4) 设置 ECCOP(CRYPTO\_ECC\_CTL[10:9]) 为 01
  - 5) 设置 MOPOP(CRYPTO\_ECC\_CTL[12:11]) 为 00
  - 6) 设置 FSEL(CRYPTO\_ECC\_CTL[8]) 依据使用的素数域或二进制域曲线a
  - 7) 设置 START(CRYPTO\_ECC\_CTL[0]) 为 1
  - 8) 等待 BUSY(CRYPTO\_ECC\_STS[0]) 清0
  - 9) 读 X1 寄存器得到 w
4. 计算  $u1 = e \times w \pmod{n}$  和  $u2 = r \times w \pmod{n}$ 
  - 1) 依据表 6.36-5写曲线顺序和曲线长度到N ,M 寄存器

- 2) 写 e, w 到 X1, Y1 寄存器
  - 3) 设置 ECCOP(CRYPTO\_ECC\_CTL[10:9]) 为 01
  - 4) 设置 MOPOP(CRYPTO\_ECC\_CTL[12:11]) 为 01
  - 5) 设置 START(CRYPTO\_ECC\_CTL[0]) 为 1
  - 6) 等待 BUSY(CRYPTO\_ECC\_STS[0]) 清0
  - 7) 读 X1 寄存器得到 u1
  - 8) 依据表 6.36-5 写曲线顺序和曲线长度到 N, M 寄存器
  - 9) 写 r, w 到 X1, Y1 寄存器
  - 10) 设置 ECCOP(CRYPTO\_ECC\_CTL[10:9]) 为 01
  - 11) 设置 MOPOP(CRYPTO\_ECC\_CTL[12:11]) 为 01
  - 12) 设置 START(CRYPTO\_ECC\_CTL[0]) 为 1
  - 13) 等待 BUSY(CRYPTO\_ECC\_STS[0]) 清0
  - 14) 读 X1 寄存器得到 u2
5. 计算  $X' (x1', y1') = u1 * G + u2 * Q$
- 1) 依据表 6.36-5 写曲线参数 A, B, N, 和曲线长度 M 到相应寄存器
  - 2) 写点 G(x, y) 到 X1, Y1 寄存器
  - 3) 写 u1 到 K 寄存器
  - 4) 设置 ECCOP(CRYPTO\_ECC\_CTL[10:9]) 为 00
  - 5) 设置 START(CRYPTO\_ECC\_CTL[0]) 为 1
  - 6) 等待 BUSY(CRYPTO\_ECC\_STS[0]) 清0
  - 7) 读 X1, Y1 寄存器得到 u1\*G
  - 8) 依据表 6.36-5 写曲线参数 A, B, N, 和曲线长度 M 到相应寄存器
  - 9) 写公共密钥 Q(x,y) 到 X1, Y1 寄存器
  - 10) 写 u2 到 K 寄存器
  - 11) 设置 ECCOP(CRYPTO\_ECC\_CTL[10:9]) 为 00
  - 12) 设置 START(CRYPTO\_ECC\_CTL[0]) 为 1
  - 13) 等待 BUSY(CRYPTO\_ECC\_STS[0]) 清0
  - 14) 依据表 6.36-5 写曲线参数 A, B, N, 和曲线长度 M 到相应寄存器
  - 15) 写结果数据 u1\*G 到 X2, Y2 寄存器
  - 16) 设置 ECCOP(CRYPTO\_ECC\_CTL[10:9]) 为 10
  - 17) 设置 START(CRYPTO\_ECC\_CTL[0]) 为 1
  - 18) 等待 BUSY(CRYPTO\_ECC\_STS[0]) 清0
  - 19) 读 X1, Y1 寄存器得到 X'(x1', y1')
  - 20) 依据表 6.36-5 写曲线顺序和曲线长度到 N, M 寄存器

- 21) 写 X1' 到 X1 寄存器
  - 22) 写 0x0 到 Y1 寄存器
  - 23) 设置 ECCOP(CRYPTO\_ECC\_CTL[10:9]) 为 01
  - 24) 设置 MOPOP(CRYPTO\_ECC\_CTL[12:11]) 为 10
  - 25) 设置 START(CRYPTO\_ECC\_CTL[0]) 为 1
  - 26) 等待 BUSY(CRYPTO\_ECC\_STS[0]) 清 0
  - 27) 读 X1 寄存器得到  $x_1' \pmod{n}$
6. 签名当  $x_1' = r$  时有效, 其他无效

### 椭圆曲线 Diffie-Hellman

共享密钥产生公式:  $Z$  是  $Q$  的  $x$ -坐标,  $Q = dG \pmod{N}$

1. 依据表 6.36-5 写曲线参数 A, B, N 和曲线长度 M 到相应的寄存器
2. 据表 6.36-5 写接收方公共密钥  $Q(x, y)$  到 X1, Y1 寄存器
3. 据表 6.36-5 写 (辅因子  $h *$  我的私钥  $d$ ) 到 K 寄存器
4.  $h=1$  在 P-192, P-224, P-256, P-384 和 P-521 中
5.  $h=2$  在 B-163, B-233, B-283, B-409, B-571 和 K-163 中
6.  $h=4$  在 K-233, K-283, K-409 和 K-571 中
7. 设置 ECCOP(CRYPTO\_ECC\_CTL[10:9]) 为 00
8. 设置 FSEL(CRYPTO\_ECC\_CTL[8]) 依据使用的素数域或二进制域曲线
9. 设置 START(CRYPTO\_ECC\_CTL[0]) 为 1
10. 等待 BUSY(CRYPTO\_ECC\_STS[0]) 清 0
11. 读 X1, Y1 寄存器得到公钥 Q

基于哈希密钥推导公式:  $\text{DerivedKeyingMaterial} = \text{KDF}(Z, \text{其它输入})$

步骤 1 For  $i = 1$  to  $\text{reps}$ , 其中  $\text{reps} = \text{ceil}((Z \text{ 长度}, \text{其它输入}) / \text{哈希长度})$

1. 依据表 6.36-5 写曲线参数 A, B, N 和曲线长度 M 到相应的寄存器
2. 据表 6.36-5 写接收方公共密钥  $Q(x, y)$  到 X1, Y1 寄存器

步骤 2 返回  $\text{DerivedKeyingMaterial} = \text{Hash}_1 \parallel \text{Hash}_2 \parallel \dots \parallel \text{Hash}_{\text{reps}}$

注:

1. 其它输入详情请参考 NIST SP 800-56A 中的第 46 页, 有关 Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography ([http://csrc.nist.gov/groups/ST/toolkit/documents/SP800-56Arev1\\_3-8-07.pdf](http://csrc.nist.gov/groups/ST/toolkit/documents/SP800-56Arev1_3-8-07.pdf))
2. 有关辅因子详情请参考 “RECOMMENDED ELLIPTIC CURVES FOR FEDERAL GOVERNMENT USE” (<http://csrc.nist.gov/groups/ST/toolkit/documents/dss/NISTReCur.pdf>)

### 6.36.6 寄存器映射

R: 只读, W: 只写, R/W: 可读可写

寄存器	偏移量	R/W	描述	复位值
<b>CRYPTO 基地址:</b>				
<b>CRYPTO_BA = 0x5008_0000</b>				
CRYPTO_INTEN	CRYPTO_BA+0x000	R/W	加密中断使能控制寄存器	0x0000_0000
CRYPTO_INTSTS	CRYPTO_BA+0x004	R/W	加密中断标志	0x0000_0000
CRYPTO_PRNG_CTL	CRYPTO_BA+0x008	R/W	PRNG 控制寄存器	0x0000_0000
CRYPTO_PRNG_SEED	CRYPTO_BA+0x00C	W	PRNG种子	Undefined
CRYPTO_PRNG_KEY0	CRYPTO_BA+0x010	R	PRNG 产生密钥0	Undefined
CRYPTO_PRNG_KEY1	CRYPTO_BA+0x014	R	PRNG产生密钥1	Undefined
CRYPTO_PRNG_KEY2	CRYPTO_BA+0x018	R	PRNG产生密钥2	Undefined
CRYPTO_PRNG_KEY3	CRYPTO_BA+0x01C	R	PRNG产生密钥3	Undefined
CRYPTO_PRNG_KEY4	CRYPTO_BA+0x020	R	PRNG产生密钥4	Undefined
CRYPTO_PRNG_KEY5	CRYPTO_BA+0x024	R	PRNG产生密钥5	Undefined
CRYPTO_PRNG_KEY6	CRYPTO_BA+0x028	R	PRNG产生密钥6	Undefined
CRYPTO_PRNG_KEY7	CRYPTO_BA+0x02C	R	PRNG产生密钥7	Undefined
CRYPTO_AES_FDBCK0	CRYPTO_BA+0x050	R	AES引擎加密操作后输出反馈数据	0x0000_0000
CRYPTO_AES_FDBCK1	CRYPTO_BA+0x054	R	AES引擎加密操作后输出反馈数据	0x0000_0000
CRYPTO_AES_FDBCK2	CRYPTO_BA+0x058	R	AES引擎加密操作后输出反馈数据	0x0000_0000
CRYPTO_AES_FDBCK3	CRYPTO_BA+0x05C	R	AES引擎加密操作后输出反馈数据	0x0000_0000
CRYPTO_TDES_FDBCK_H	CRYPTO_BA+0x060	R	TDES/DES 引擎加密操作后输出反馈数据高位	0x0000_0000
CRYPTO_TDES_FDBCK_L	CRYPTO_BA+0x064	R	TDES/DES 引擎加密操作后输出反馈数据低位	0x0000_0000
CRYPTO_AES_CTL	CRYPTO_BA+0x100	R/W	AES 控制寄存器	0x0000_0000
CRYPTO_AES_STS	CRYPTO_BA+0x104	R	AES 引擎标志	0x0001_0100
CRYPTO_AES_DATIN	CRYPTO_BA+0x108	R/W	AES 引擎数据输入端口寄存器	0x0000_0000
CRYPTO_AES_DATOUT	CRYPTO_BA+0x10C	R	AES 引擎数据输出端口寄存器	0x0000_0000
CRYPTO_AES0_KEY0	CRYPTO_BA+0x110	R/W	AES 通道0 密钥字0 寄存器	0x0000_0000
CRYPTO_AES0_KEY1	CRYPTO_BA+0x114	R/W	AES 通道0 密钥字1寄存器	0x0000_0000
CRYPTO_AES0_KEY2	CRYPTO_BA+0x118	R/W	AES 通道0 密钥字2 寄存器	0x0000_0000
CRYPTO_AES0_KEY3	CRYPTO_BA+0x11C	R/W	AES 通道0 密钥字3 寄存器	0x0000_0000
CRYPTO_AES0_KEY4	CRYPTO_BA+0x120	R/W	AES 通道0 密钥字4 寄存器	0x0000_0000
CRYPTO_AES0_KEY5	CRYPTO_BA+0x124	R/W	AES 通道0 密钥字5 寄存器	0x0000_0000

CRYPTO_AES0_KEY6	CRYPTO_BA+0x128	R/W	AES 通道0 密钥字6 寄存器	0x0000_0000
CRYPTO_AES0_KEY7	CRYPTO_BA+0x12C	R/W	AES 通道0 密钥字7 寄存器	0x0000_0000
CRYPTO_AES0_IV0	CRYPTO_BA+0x130	R/W	AES 通道0 初始化向量字0 寄存器	0x0000_0000
CRYPTO_AES0_IV1	CRYPTO_BA+0x134	R/W	AES 通道0 初始化向量字1 寄存器	0x0000_0000
CRYPTO_AES0_IV2	CRYPTO_BA+0x138	R/W	AES 通道0 初始化向量字2 寄存器	0x0000_0000
CRYPTO_AES0_IV3	CRYPTO_BA+0x13C	R/W	AES 通道0 初始化向量字3 寄存器	0x0000_0000
CRYPTO_AES0_SADDR	CRYPTO_BA+0x140	R/W	AES 通道0 DMA源地址寄存器	0x0000_0000
CRYPTO_AES0_DADDR	CRYPTO_BA+0x144	R/W	AES 通道0 DMA 目的地址寄存器	0x0000_0000
CRYPTO_AES0_CNT	CRYPTO_BA+0x148	R/W	AES 通道0字节数寄存器	0x0000_0000
CRYPTO_AES1_KEY0	CRYPTO_BA+0x14C	R/W	AES 通道1 密钥字0 寄存器	0x0000_0000
CRYPTO_AES1_KEY1	CRYPTO_BA+0x150	R/W	AES 通道1 密钥字1寄存器	0x0000_0000
CRYPTO_AES1_KEY2	CRYPTO_BA+0x154	R/W	AES 通道1 密钥字2 寄存器	0x0000_0000
CRYPTO_AES1_KEY3	CRYPTO_BA+0x158	R/W	AES 通道1 密钥字3 寄存器	0x0000_0000
CRYPTO_AES1_KEY4	CRYPTO_BA+0x15C	R/W	AES 通道1 密钥字4 寄存器	0x0000_0000
CRYPTO_AES1_KEY5	CRYPTO_BA+0x160	R/W	AES 通道1 密钥字5 寄存器	0x0000_0000
CRYPTO_AES1_KEY6	CRYPTO_BA+0x164	R/W	AES 通道1 密钥字6 寄存器	0x0000_0000
CRYPTO_AES1_KEY7	CRYPTO_BA+0x168	R/W	AES 通道1 密钥字7 寄存器	0x0000_0000
CRYPTO_AES1_IV0	CRYPTO_BA+0x16C	R/W	AES 通道1初始化向量字0 寄存器	0x0000_0000
CRYPTO_AES1_IV1	CRYPTO_BA+0x170	R/W	AES 通道1初始化向量字1 寄存器	0x0000_0000
CRYPTO_AES1_IV2	CRYPTO_BA+0x174	R/W	AES 通道1初始化向量字2 寄存器	0x0000_0000
CRYPTO_AES1_IV3	CRYPTO_BA+0x178	R/W	AES 通道1初始化向量字3 寄存器	0x0000_0000
CRYPTO_AES1_SADDR	CRYPTO_BA+0x17C	R/W	AES 通道1 DMA源地址寄存器	0x0000_0000
CRYPTO_AES1_DADDR	CRYPTO_BA+0x180	R/W	AES 通道1 DMA 目的地址寄存器	0x0000_0000
CRYPTO_AES1_CNT	CRYPTO_BA+0x184	R/W	AES 通道1字节数寄存器	0x0000_0000
CRYPTO_AES2_KEY0	CRYPTO_BA+0x188	R/W	AES 通道2 密钥字0 寄存器	0x0000_0000
CRYPTO_AES2_KEY1	CRYPTO_BA+0x18C	R/W	AES 通道2 密钥字1寄存器	0x0000_0000
CRYPTO_AES2_KEY2	CRYPTO_BA+0x190	R/W	AES 通道2 密钥字2 寄存器	0x0000_0000
CRYPTO_AES2_KEY3	CRYPTO_BA+0x194	R/W	AES 通道2 密钥字3 寄存器	0x0000_0000
CRYPTO_AES2_KEY4	CRYPTO_BA+0x198	R/W	AES 通道2 密钥字4 寄存器	0x0000_0000
CRYPTO_AES2_KEY5	CRYPTO_BA+0x19C	R/W	AES 通道2 密钥字5 寄存器	0x0000_0000
CRYPTO_AES2_KEY6	CRYPTO_BA+0x1A0	R/W	AES 通道2 密钥字6 寄存器	0x0000_0000
CRYPTO_AES2_KEY7	CRYPTO_BA+0x1A4	R/W	AES 通道2 密钥字7 寄存器	0x0000_0000
CRYPTO_AES2_IV0	CRYPTO_BA+0x1A8	R/W	AES 通道2初始化向量字0 寄存器	0x0000_0000

CRYPTO_AES2_IV1	CRYPTO_BA+0x1AC	R/W	AES 通道2初始化向量字1 寄存器	0x0000_0000
CRYPTO_AES2_IV2	CRYPTO_BA+0x1B0	R/W	AES 通道2初始化向量字2 寄存器	0x0000_0000
CRYPTO_AES2_IV3	CRYPTO_BA+0x1B4	R/W	AES 通道2初始化向量字3 寄存器	0x0000_0000
CRYPTO_AES2_SADDR	CRYPTO_BA+0x1B8	R/W	AES 通道2 DMA源地址寄存器	0x0000_0000
CRYPTO_AES2_DADDR	CRYPTO_BA+0x1BC	R/W	AES 通道2 DMA 目的地址寄存器	0x0000_0000
CRYPTO_AES2_CNT	CRYPTO_BA+0x1C0	R/W	AES 通道2字节数寄存器	0x0000_0000
CRYPTO_AES3_KEY0	CRYPTO_BA+0x1C4	R/W	AES 通道3 密钥字0 寄存器	0x0000_0000
CRYPTO_AES3_KEY1	CRYPTO_BA+0x1C8	R/W	AES 通道3 密钥字1寄存器	0x0000_0000
CRYPTO_AES3_KEY2	CRYPTO_BA+0x1CC	R/W	AES 通道3 密钥字2 寄存器	0x0000_0000
CRYPTO_AES3_KEY3	CRYPTO_BA+0x1D0	R/W	AES 通道3 密钥字3 寄存器	0x0000_0000
CRYPTO_AES3_KEY4	CRYPTO_BA+0x1D4	R/W	AES 通道3 密钥字4 寄存器	0x0000_0000
CRYPTO_AES3_KEY5	CRYPTO_BA+0x1D8	R/W	AES 通道3 密钥字5 寄存器	0x0000_0000
CRYPTO_AES3_KEY6	CRYPTO_BA+0x1DC	R/W	AES 通道3 密钥字6 寄存器	0x0000_0000
CRYPTO_AES3_KEY7	CRYPTO_BA+0x1E0	R/W	AES 通道3 密钥字7 寄存器	0x0000_0000
CRYPTO_AES3_IV0	CRYPTO_BA+0x1E4	R/W	AES 通道3初始化向量字0 寄存器	0x0000_0000
CRYPTO_AES3_IV1	CRYPTO_BA+0x1E8	R/W	AES 通道3初始化向量字1 寄存器	0x0000_0000
CRYPTO_AES3_IV2	CRYPTO_BA+0x1EC	R/W	AES 通道3初始化向量字2 寄存器	0x0000_0000
CRYPTO_AES3_IV3	CRYPTO_BA+0x1F0	R/W	AES 通道3初始化向量字3 寄存器	0x0000_0000
CRYPTO_AES3_SADDR	CRYPTO_BA+0x1F4	R/W	AES 通道3 DMA源地址寄存器	0x0000_0000
CRYPTO_AES3_DADDR	CRYPTO_BA+0x1F8	R/W	AES 通道3 DMA 目的地址寄存器	0x0000_0000
CRYPTO_AES3_CNT	CRYPTO_BA+0x1FC	R/W	AES 通道3字节数寄存器	0x0000_0000
CRYPTO_TDES_CTL	CRYPTO_BA+0x200	R/W	TDES/DES 控制寄存器	0x0000_0000
CRYPTO_TDES_STS	CRYPTO_BA+0x204	R	TDES/DES 引擎标志	0x0001_0100
CRYPTO_TDES0_KEY1_H	CRYPTO_BA+0x208	R/W	TDES/DES 通道0 密钥 1 高位寄存器	0x0000_0000
CRYPTO_TDES0_KEY1_L	CRYPTO_BA+0x20C	R/W	TDES/DES 通道0 密钥 1 低位寄存器	0x0000_0000
CRYPTO_TDES0_KEY2_H	CRYPTO_BA+0x210	R/W	TDES/DES 通道0 密钥 2 高位寄存器	0x0000_0000
CRYPTO_TDES0_KEY2_L	CRYPTO_BA+0x214	R/W	TDES/DES 通道0 密钥 2 低位寄存器	0x0000_0000
CRYPTO_TDES0_KEY3_H	CRYPTO_BA+0x218	R/W	TDES/DES 通道0 密钥 3高位寄存器	0x0000_0000
CRYPTO_TDES0_KEY3_L	CRYPTO_BA+0x21C	R/W	TDES/DES 通道0 密钥 3 低位寄存器	0x0000_0000
CRYPTO_TDES0_IVH	CRYPTO_BA+0x220	R/W	TDES/DES 通道0初始化向量高位寄存器	0x0000_0000

CRYPTO_TDES0_IVL	CRYPTO_BA+0x224	R/W	TDES/DES 通道0初始化向量低位寄存器	0x0000_0000
CRYPTO_TDES0_SADD_R	CRYPTO_BA+0x228	R/W	TDES/DES 通道0 DMA 源地址寄存器	0x0000_0000
CRYPTO_TDES0_DADD_R	CRYPTO_BA+0x22C	R/W	TDES/DES 通道0 DMA 目的地址寄存器	0x0000_0000
CRYPTO_TDES0_CNT	CRYPTO_BA+0x230	R/W	TDES/DES 通道0字节数寄存器	0x0000_0000
CRYPTO_TDES_DATIN	CRYPTO_BA+0x234	R/W	TDES/DES 引擎输入数据寄存器	0x0000_0000
CRYPTO_TDES_DATOUT	CRYPTO_BA+0x238	R	TDES/DES 引起输出数据寄存器	0x0000_0000
CRYPTO_TDES1_KEY1_H	CRYPTO_BA+0x248	R/W	TDES/DES 通道1密钥 1 高位寄存器	0x0000_0000
CRYPTO_TDES1_KEY1_L	CRYPTO_BA+0x24C	R/W	TDES/DES 通道1 密钥 1 低位寄存器	0x0000_0000
CRYPTO_TDES1_KEY2_H	CRYPTO_BA+0x250	R/W	TDES/DES 通道1 密钥 2 高位寄存器	0x0000_0000
CRYPTO_TDES1_KEY2_L	CRYPTO_BA+0x254	R/W	TDES/DES 通道1 密钥 2 低位寄存器	0x0000_0000
CRYPTO_TDES1_KEY3_H	CRYPTO_BA+0x258	R/W	TDES/DES 通道1 密钥 3高位寄存器	0x0000_0000
CRYPTO_TDES1_KEY3_L	CRYPTO_BA+0x25C	R/W	TDES/DES 通道1 密钥 3 低位寄存器	0x0000_0000
CRYPTO_TDES1_IVH	CRYPTO_BA+0x260	R/W	TDES/DES 通道1初始化向量高位寄存器	0x0000_0000
CRYPTO_TDES1_IVL	CRYPTO_BA+0x264	R/W	TDES/DES 通道1初始化向量低位寄存器	0x0000_0000
CRYPTO_TDES1_SADD_R	CRYPTO_BA+0x268	R/W	TDES/DES 通道1 DMA 源地址寄存器	0x0000_0000
CRYPTO_TDES1_DADD_R	CRYPTO_BA+0x26C	R/W	TDES/DES 通道1 DMA 目的地址寄存器	0x0000_0000
CRYPTO_TDES1_CNT	CRYPTO_BA+0x270	R/W	TDES/DES 通道1字节数寄存器	0x0000_0000
CRYPTO_TDES2_KEY1_H	CRYPTO_BA+0x288	R/W	TDES/DES 通道2密钥 1 高位寄存器	0x0000_0000
CRYPTO_TDES2_KEY1_L	CRYPTO_BA+0x28C	R/W	TDES/DES 通道2 密钥 1 低位寄存器	0x0000_0000
CRYPTO_TDES2_KEY2_H	CRYPTO_BA+0x290	R/W	TDES/DES 通道2 密钥 2 高位寄存器	0x0000_0000
CRYPTO_TDES2_KEY2_L	CRYPTO_BA+0x294	R/W	TDES/DES 通道2 密钥 2 低位寄存器	0x0000_0000
CRYPTO_TDES2_KEY3_H	CRYPTO_BA+0x298	R/W	TDES/DES 通道2 密钥 3高位寄存器	0x0000_0000
CRYPTO_TDES2_KEY3_L	CRYPTO_BA+0x29C	R/W	TDES/DES 通道2 密钥 3 低位寄存器	0x0000_0000
CRYPTO_TDES2_IVH	CRYPTO_BA+0x2A0	R/W	TDES/DES 通道2初始化向量高位寄存器	0x0000_0000
CRYPTO_TDES2_IVL	CRYPTO_BA+0x2A4	R/W	TDES/DES 通道2初始化向量低位寄存器	0x0000_0000
CRYPTO_TDES2_SADD_R	CRYPTO_BA+0x2A8	R/W	TDES/DES 通道2 DMA 源地址寄存器	0x0000_0000

<b>CRYPTO_TDES2_DADD_R</b>	CRYPTO_BA+0x2AC	R/W	TDES/DES 通道2 DMA 目的地址寄存器	0x0000_0000
<b>CRYPTO_TDES2_CNT</b>	CRYPTO_BA+0x2B0	R/W	TDES/DES 通道2字节数寄存器	0x0000_0000
<b>CRYPTO_TDES3_KEY1_H</b>	CRYPTO_BA+0x2C8	R/W	TDES/DES 通道3 密钥 1 高位寄存器	0x0000_0000
<b>CRYPTO_TDES3_KEY1_L</b>	CRYPTO_BA+0x2CC	R/W	TDES/DES 通道3 密钥 1 低位寄存器	0x0000_0000
<b>CRYPTO_TDES3_KEY2_H</b>	CRYPTO_BA+0x2D0	R/W	TDES/DES 通道3 密钥 2 高位寄存器	0x0000_0000
<b>CRYPTO_TDES3_KEY2_L</b>	CRYPTO_BA+0x2D4	R/W	TDES/DES 通道3 密钥 2 低位寄存器	0x0000_0000
<b>CRYPTO_TDES3_KEY3_H</b>	CRYPTO_BA+0x2D8	R/W	TDES/DES 通道3 密钥 3高位寄存器	0x0000_0000
<b>CRYPTO_TDES3_KEY3_L</b>	CRYPTO_BA+0x2DC	R/W	TDES/DES 通道3 密钥 3 低位寄存器	0x0000_0000
<b>CRYPTO_TDES3_IVH</b>	CRYPTO_BA+0x2E0	R/W	TDES/DES 通道3初始化向量高位寄存器	0x0000_0000
<b>CRYPTO_TDES3_IVL</b>	CRYPTO_BA+0x2E4	R/W	TDES/DES 通道3初始化向量低位寄存器	0x0000_0000
<b>CRYPTO_TDES3_SADD_R</b>	CRYPTO_BA+0x2E8	R/W	TDES/DES 通道3 DMA 源地址寄存器	0x0000_0000
<b>CRYPTO_TDES3_DADD_R</b>	CRYPTO_BA+0x2EC	R/W	TDES/DES 通道3 DMA 目的地址寄存器	0x0000_0000
<b>CRYPTO_TDES3_CNT</b>	CRYPTO_BA+0x2F0	R/W	TDES/DES 通道3字节数寄存器	0x0000_0000
<b>CRYPTO_HMAC_CTL</b>	CRYPTO_BA+0x300	R/W	SHA/HMAC 控制寄存器	0x0000_0000
<b>CRYPTO_HMAC_STS</b>	CRYPTO_BA+0x304	R	SHA/HMAC 状态标志	0x0000_0000
<b>CRYPTO_HMAC_DGST_0</b>	CRYPTO_BA+0x308	R	SHA/HMAC 信息摘要 0	0x0000_0000
<b>CRYPTO_HMAC_DGST_1</b>	CRYPTO_BA+0x30C	R	SHA/HMAC信息摘要1	0x0000_0000
<b>CRYPTO_HMAC_DGST_2</b>	CRYPTO_BA+0x310	R	SHA/HMAC信息摘要2	0x0000_0000
<b>CRYPTO_HMAC_DGST_3</b>	CRYPTO_BA+0x314	R	SHA/HMAC信息摘要3	0x0000_0000
<b>CRYPTO_HMAC_DGST_4</b>	CRYPTO_BA+0x318	R	SHA/HMAC信息摘要4	0x0000_0000
<b>CRYPTO_HMAC_DGST_5</b>	CRYPTO_BA+0x31C	R	SHA/HMAC信息摘要5	0x0000_0000
<b>CRYPTO_HMAC_DGST_6</b>	CRYPTO_BA+0x320	R	SHA/HMAC信息摘要6	0x0000_0000
<b>CRYPTO_HMAC_DGST_7</b>	CRYPTO_BA+0x324	R	SHA/HMAC信息摘要7	0x0000_0000
<b>CRYPTO_HMAC_DGST_8</b>	CRYPTO_BA+0x328	R	SHA/HMAC信息摘要8	0x0000_0000
<b>CRYPTO_HMAC_DGST_9</b>	CRYPTO_BA+0x32C	R	SHA/HMAC信息摘要9	0x0000_0000

CRYPTO_HMAC_DGST_10	CRYPTO_BA+0x330	R	SHA/HMAC信息摘要10	0x0000_0000
CRYPTO_HMAC_DGST_11	CRYPTO_BA+0x334	R	SHA/HMAC信息摘要11	0x0000_0000
CRYPTO_HMAC_DGST_12	CRYPTO_BA+0x338	R	SHA/HMAC信息摘要12	0x0000_0000
CRYPTO_HMAC_DGST_13	CRYPTO_BA+0x33C	R	SHA/HMAC信息摘要13	0x0000_0000
CRYPTO_HMAC_DGST_14	CRYPTO_BA+0x340	R	SHA/HMAC信息摘要14	0x0000_0000
CRYPTO_HMAC_DGST_15	CRYPTO_BA+0x344	R	SHA/HMAC信息摘要15	0x0000_0000
CRYPTO_HMAC_KEYCNT	CRYPTO_BA+0x348	R/W	SHA/HMAC 密钥字节数寄存器	0x0000_0000
CRYPTO_HMAC_SADDR	CRYPTO_BA+0x34C	R/W	SHA/HMAC DMA 源地址寄存器	0x0000_0000
CRYPTO_HMAC_DMACNT	CRYPTO_BA+0x350	R/W	SHA/HMAC 字节数寄存器	0x0000_0000
CRYPTO_HMAC_DATIN	CRYPTO_BA+0x354	R/W	SHA/HMAC引擎 非-DMA 模式数据输入端口寄存器	0x0000_0000
CRYPTO_ECC_CTL	CRYPTO_BA+0x800	R/W	ECC 控制寄存器	0x0000_0000
CRYPTO_ECC_STS	CRYPTO_BA+0x804	R	ECC 状态寄存器	0x0000_0000
CRYPTO_ECC_X1_00	CRYPTO_BA+0x808	R/W	ECC 第一点的 X坐标字0	0x0000_0000
CRYPTO_ECC_X1_01	CRYPTO_BA+0x80C	R/W	ECC 第一点的 X坐标字1	0x0000_0000
CRYPTO_ECC_X1_02	CRYPTO_BA+0x810	R/W	ECC 第一点的 X坐标字2	0x0000_0000
CRYPTO_ECC_X1_03	CRYPTO_BA+0x814	R/W	ECC 第一点的 X坐标字3	0x0000_0000
CRYPTO_ECC_X1_04	CRYPTO_BA+0x818	R/W	ECC 第一点的 X坐标字4	0x0000_0000
CRYPTO_ECC_X1_05	CRYPTO_BA+0x81C	R/W	ECC 第一点的 X坐标字5	0x0000_0000
CRYPTO_ECC_X1_06	CRYPTO_BA+0x820	R/W	ECC 第一点的 X坐标字6	0x0000_0000
CRYPTO_ECC_X1_07	CRYPTO_BA+0x824	R/W	ECC 第一点的 X坐标字7	0x0000_0000
CRYPTO_ECC_X1_08	CRYPTO_BA+0x828	R/W	ECC 第一点的 X坐标字8	0x0000_0000
CRYPTO_ECC_X1_09	CRYPTO_BA+0x82C	R/W	ECC 第一点的 X坐标字9	0x0000_0000
CRYPTO_ECC_X1_10	CRYPTO_BA+0x830	R/W	ECC 第一点的 X坐标字10	0x0000_0000
CRYPTO_ECC_X1_11	CRYPTO_BA+0x834	R/W	ECC 第一点的 X坐标字11	0x0000_0000
CRYPTO_ECC_X1_12	CRYPTO_BA+0x838	R/W	ECC 第一点的 X坐标字12	0x0000_0000
CRYPTO_ECC_X1_13	CRYPTO_BA+0x83C	R/W	ECC 第一点的 X坐标字13	0x0000_0000
CRYPTO_ECC_X1_14	CRYPTO_BA+0x840	R/W	ECC 第一点的 X坐标字14	0x0000_0000
CRYPTO_ECC_X1_15	CRYPTO_BA+0x844	R/W	ECC 第一点的 X坐标字15	0x0000_0000
CRYPTO_ECC_X1_16	CRYPTO_BA+0x848	R/W	ECC 第一点的 X坐标字16	0x0000_0000
CRYPTO_ECC_X1_17	CRYPTO_BA+0x84C	R/W	ECC 第一点的 X坐标字17	0x0000_0000

CRYPTO_ECC_Y1_00	CRYPTO_BA+0x850	R/W	ECC 第一点的 Y坐标字0	0x0000_0000
CRYPTO_ECC_Y1_01	CRYPTO_BA+0x854	R/W	ECC 第一点的 Y坐标字1	0x0000_0000
CRYPTO_ECC_Y1_02	CRYPTO_BA+0x858	R/W	ECC 第一点的 Y坐标字2	0x0000_0000
CRYPTO_ECC_Y1_03	CRYPTO_BA+0x85C	R/W	ECC 第一点的 Y坐标字3	0x0000_0000
CRYPTO_ECC_Y1_04	CRYPTO_BA+0x860	R/W	ECC 第一点的 Y坐标字4	0x0000_0000
CRYPTO_ECC_Y1_05	CRYPTO_BA+0x864	R/W	ECC 第一点的 Y坐标字5	0x0000_0000
CRYPTO_ECC_Y1_06	CRYPTO_BA+0x868	R/W	ECC 第一点的 Y坐标字6	0x0000_0000
CRYPTO_ECC_Y1_07	CRYPTO_BA+0x86C	R/W	ECC 第一点的 Y坐标字7	0x0000_0000
CRYPTO_ECC_Y1_08	CRYPTO_BA+0x870	R/W	ECC 第一点的 Y坐标字8	0x0000_0000
CRYPTO_ECC_Y1_09	CRYPTO_BA+0x874	R/W	ECC 第一点的 Y坐标字9	0x0000_0000
CRYPTO_ECC_Y1_10	CRYPTO_BA+0x878	R/W	ECC 第一点的 Y坐标字10	0x0000_0000
CRYPTO_ECC_Y1_11	CRYPTO_BA+0x87C	R/W	ECC 第一点的 Y坐标字11	0x0000_0000
CRYPTO_ECC_Y1_12	CRYPTO_BA+0x880	R/W	ECC 第一点的 Y坐标字12	0x0000_0000
CRYPTO_ECC_Y1_13	CRYPTO_BA+0x884	R/W	ECC 第一点的 Y坐标字13	0x0000_0000
CRYPTO_ECC_Y1_14	CRYPTO_BA+0x888	R/W	ECC 第一点的 Y坐标字14	0x0000_0000
CRYPTO_ECC_Y1_15	CRYPTO_BA+0x88C	R/W	ECC 第一点的 Y坐标字15	0x0000_0000
CRYPTO_ECC_Y1_16	CRYPTO_BA+0x890	R/W	ECC 第一点的 Y坐标字16	0x0000_0000
CRYPTO_ECC_Y1_17	CRYPTO_BA+0x894	R/W	ECC 第一点的 Y坐标字17	0x0000_0000
CRYPTO_ECC_X2_00	CRYPTO_BA+0x898	R/W	ECC 第二点的 X坐标字0	0x0000_0000
CRYPTO_ECC_X2_01	CRYPTO_BA+0x89C	R/W	ECC 第二点的 X坐标字1	0x0000_0000
CRYPTO_ECC_X2_02	CRYPTO_BA+0x8A0	R/W	ECC 第二点的 X坐标字2	0x0000_0000
CRYPTO_ECC_X2_03	CRYPTO_BA+0x8A4	R/W	ECC 第二点的 X坐标字3	0x0000_0000
CRYPTO_ECC_X2_04	CRYPTO_BA+0x8A8	R/W	ECC 第二点的 X坐标字4	0x0000_0000
CRYPTO_ECC_X2_05	CRYPTO_BA+0x8AC	R/W	ECC 第二点的 X坐标字5	0x0000_0000
CRYPTO_ECC_X2_06	CRYPTO_BA+0x8B0	R/W	ECC 第二点的 X坐标字6	0x0000_0000
CRYPTO_ECC_X2_07	CRYPTO_BA+0x8B4	R/W	ECC 第二点的 X坐标字7	0x0000_0000
CRYPTO_ECC_X2_08	CRYPTO_BA+0x8B8	R/W	ECC 第二点的 X坐标字8	0x0000_0000
CRYPTO_ECC_X2_09	CRYPTO_BA+0x8BC	R/W	ECC 第二点的 X坐标字9	0x0000_0000
CRYPTO_ECC_X2_10	CRYPTO_BA+0x8C0	R/W	ECC 第二点的 X坐标字10	0x0000_0000
CRYPTO_ECC_X2_11	CRYPTO_BA+0x8C4	R/W	ECC 第二点的 X坐标字11	0x0000_0000
CRYPTO_ECC_X2_12	CRYPTO_BA+0x8C8	R/W	ECC 第二点的 X坐标字12	0x0000_0000
CRYPTO_ECC_X2_13	CRYPTO_BA+0x8C C	R/W	ECC 第二点的 X坐标字13	0x0000_0000
CRYPTO_ECC_X2_14	CRYPTO_BA+0x8D0	R/W	ECC 第二点的 X坐标字14	0x0000_0000

CRYPTO_ECC_X2_15	CRYPTO_BA+0x8D4	R/W	ECC 第二点的 X坐标字15	0x0000_0000
CRYPTO_ECC_X2_16	CRYPTO_BA+0x8D8	R/W	ECC 第二点的 X坐标字16	0x0000_0000
CRYPTO_ECC_X2_17	CRYPTO_BA+0x8DC	R/W	ECC 第二点的 X坐标字17	0x0000_0000
CRYPTO_ECC_Y2_00	CRYPTO_BA+0x8E0	R/W	ECC 第二点的 Y坐标字0	0x0000_0000
CRYPTO_ECC_Y2_01	CRYPTO_BA+0x8E4	R/W	ECC 第二点的 Y坐标字1	0x0000_0000
CRYPTO_ECC_Y2_02	CRYPTO_BA+0x8E8	R/W	ECC 第二点的 Y坐标字2	0x0000_0000
CRYPTO_ECC_Y2_03	CRYPTO_BA+0x8EC	R/W	ECC 第二点的 Y坐标字3	0x0000_0000
CRYPTO_ECC_Y2_04	CRYPTO_BA+0x8F0	R/W	ECC 第二点的 Y坐标字4	0x0000_0000
CRYPTO_ECC_Y2_05	CRYPTO_BA+0x8F4	R/W	ECC 第二点的 Y坐标字5	0x0000_0000
CRYPTO_ECC_Y2_06	CRYPTO_BA+0x8F8	R/W	ECC 第二点的 Y坐标字6	0x0000_0000
CRYPTO_ECC_Y2_07	CRYPTO_BA+0x8FC	R/W	ECC 第二点的 Y坐标字7	0x0000_0000
CRYPTO_ECC_Y2_08	CRYPTO_BA+0x900	R/W	ECC 第二点的 Y坐标字8	0x0000_0000
CRYPTO_ECC_Y2_09	CRYPTO_BA+0x904	R/W	ECC 第二点的 Y坐标字9	0x0000_0000
CRYPTO_ECC_Y2_10	CRYPTO_BA+0x908	R/W	ECC 第二点的 Y坐标字10	0x0000_0000
CRYPTO_ECC_Y2_11	CRYPTO_BA+0x90C	R/W	ECC 第二点的 Y坐标字11	0x0000_0000
CRYPTO_ECC_Y2_12	CRYPTO_BA+0x910	R/W	ECC 第二点的 Y坐标字12	0x0000_0000
CRYPTO_ECC_Y2_13	CRYPTO_BA+0x914	R/W	ECC 第二点的 Y坐标字13	0x0000_0000
CRYPTO_ECC_Y2_14	CRYPTO_BA+0x918	R/W	ECC 第二点的 Y坐标字14	0x0000_0000
CRYPTO_ECC_Y2_15	CRYPTO_BA+0x91C	R/W	ECC 第二点的 Y坐标字15	0x0000_0000
CRYPTO_ECC_Y2_16	CRYPTO_BA+0x920	R/W	ECC 第二点的 Y坐标字16	0x0000_0000
CRYPTO_ECC_Y2_17	CRYPTO_BA+0x924	R/W	ECC 第二点的 Y坐标字17	0x0000_0000
CRYPTO_ECC_A_00	CRYPTO_BA+0x928	R/W	ECC 椭圆曲线参数CURVEA 字 0	0x0000_0000
CRYPTO_ECC_A_01	CRYPTO_BA+0x92C	R/W	ECC 椭圆曲线参数CURVEA 字 1	0x0000_0000
CRYPTO_ECC_A_02	CRYPTO_BA+0x930	R/W	ECC 椭圆曲线参数CURVEA 字 2	0x0000_0000
CRYPTO_ECC_A_03	CRYPTO_BA+0x934	R/W	ECC 椭圆曲线参数CURVEA 字 3	0x0000_0000
CRYPTO_ECC_A_04	CRYPTO_BA+0x938	R/W	ECC 椭圆曲线参数CURVEA 字 4	0x0000_0000
CRYPTO_ECC_A_05	CRYPTO_BA+0x93C	R/W	ECC 椭圆曲线参数CURVEA 字 5	0x0000_0000
CRYPTO_ECC_A_06	CRYPTO_BA+0x940	R/W	ECC 椭圆曲线参数CURVEA 字 6	0x0000_0000
CRYPTO_ECC_A_07	CRYPTO_BA+0x944	R/W	ECC 椭圆曲线参数CURVEA 字 7	0x0000_0000
CRYPTO_ECC_A_08	CRYPTO_BA+0x948	R/W	ECC 椭圆曲线参数CURVEA 字 8	0x0000_0000
CRYPTO_ECC_A_09	CRYPTO_BA+0x94C	R/W	ECC 椭圆曲线参数CURVEA 字9	0x0000_0000
CRYPTO_ECC_A_10	CRYPTO_BA+0x950	R/W	ECC 椭圆曲线参数CURVEA 字 10	0x0000_0000
CRYPTO_ECC_A_11	CRYPTO_BA+0x954	R/W	ECC 椭圆曲线参数CURVEA 字 11	0x0000_0000

CRYPTO_ECC_A_12	CRYPTO_BA+0x958	R/W	ECC 椭圆曲线参数CURVEA 字 12	0x0000_0000
CRYPTO_ECC_A_13	CRYPTO_BA+0x95C	R/W	ECC 椭圆曲线参数CURVEA 字 13	0x0000_0000
CRYPTO_ECC_A_14	CRYPTO_BA+0x960	R/W	ECC 椭圆曲线参数CURVEA 字 14	0x0000_0000
CRYPTO_ECC_A_15	CRYPTO_BA+0x964	R/W	ECC 椭圆曲线参数CURVEA 字 15	0x0000_0000
CRYPTO_ECC_A_16	CRYPTO_BA+0x968	R/W	ECC 椭圆曲线参数CURVEA 字 16	0x0000_0000
CRYPTO_ECC_A_17	CRYPTO_BA+0x96C	R/W	ECC 椭圆曲线参数CURVEA 字 17	0x0000_0000
CRYPTO_ECC_B_00	CRYPTO_BA+0x970	R/W	ECC 椭圆曲线参数CURVEB 字 0	0x0000_0000
CRYPTO_ECC_B_01	CRYPTO_BA+0x974	R/W	ECC 椭圆曲线参数CURVEB 字 1	0x0000_0000
CRYPTO_ECC_B_02	CRYPTO_BA+0x978	R/W	ECC 椭圆曲线参数CURVEB 字 2	0x0000_0000
CRYPTO_ECC_B_03	CRYPTO_BA+0x97C	R/W	ECC 椭圆曲线参数CURVEB 字 3	0x0000_0000
CRYPTO_ECC_B_04	CRYPTO_BA+0x980	R/W	ECC 椭圆曲线参数CURVEB 字 4	0x0000_0000
CRYPTO_ECC_B_05	CRYPTO_BA+0x984	R/W	ECC 椭圆曲线参数CURVEB 字 5	0x0000_0000
CRYPTO_ECC_B_06	CRYPTO_BA+0x988	R/W	ECC 椭圆曲线参数CURVEB 字 6	0x0000_0000
CRYPTO_ECC_B_07	CRYPTO_BA+0x98C	R/W	ECC 椭圆曲线参数CURVEB 字 7	0x0000_0000
CRYPTO_ECC_B_08	CRYPTO_BA+0x990	R/W	ECC 椭圆曲线参数CURVEB 字 8	0x0000_0000
CRYPTO_ECC_B_09	CRYPTO_BA+0x994	R/W	ECC 椭圆曲线参数CURVEB 字 9	0x0000_0000
CRYPTO_ECC_B_10	CRYPTO_BA+0x998	R/W	ECC 椭圆曲线参数CURVEB 字 10	0x0000_0000
CRYPTO_ECC_B_11	CRYPTO_BA+0x99C	R/W	ECC 椭圆曲线参数CURVEB 字 11	0x0000_0000
CRYPTO_ECC_B_12	CRYPTO_BA+0x9A0	R/W	ECC 椭圆曲线参数CURVEB 字 12	0x0000_0000
CRYPTO_ECC_B_13	CRYPTO_BA+0x9A4	R/W	ECC 椭圆曲线参数CURVEB 字 13	0x0000_0000
CRYPTO_ECC_B_14	CRYPTO_BA+0x9A8	R/W	ECC 椭圆曲线参数CURVEB 字 14	0x0000_0000
CRYPTO_ECC_B_15	CRYPTO_BA+0x9AC	R/W	ECC 椭圆曲线参数CURVEB 字 15	0x0000_0000
CRYPTO_ECC_B_16	CRYPTO_BA+0x9B0	R/W	ECC 椭圆曲线参数CURVEB 字 16	0x0000_0000
CRYPTO_ECC_B_17	CRYPTO_BA+0x9B4	R/W	ECC 椭圆曲线参数CURVEB 字 17	0x0000_0000
CRYPTO_ECC_N_00	CRYPTO_BA+0x9B8	R/W	ECC 椭圆曲线参数CURVEN 字 0	0x0000_0000
CRYPTO_ECC_N_01	CRYPTO_BA+0x9BC	R/W	ECC 椭圆曲线参数CURVEN 字 1	0x0000_0000
CRYPTO_ECC_N_02	CRYPTO_BA+0x9C0	R/W	ECC 椭圆曲线参数CURVEN 字 2	0x0000_0000
CRYPTO_ECC_N_03	CRYPTO_BA+0x9C4	R/W	ECC 椭圆曲线参数CURVEN 字 3	0x0000_0000
CRYPTO_ECC_N_04	CRYPTO_BA+0x9C8	R/W	ECC 椭圆曲线参数CURVEN 字 4	0x0000_0000
CRYPTO_ECC_N_05	CRYPTO_BA+0x9C C	R/W	ECC 椭圆曲线参数CURVEN 字 5	0x0000_0000
CRYPTO_ECC_N_06	CRYPTO_BA+0x9D0	R/W	ECC 椭圆曲线参数CURVEN 字 6	0x0000_0000
CRYPTO_ECC_N_07	CRYPTO_BA+0x9D4	R/W	ECC 椭圆曲线参数CURVEN 字 7	0x0000_0000
CRYPTO_ECC_N_08	CRYPTO_BA+0x9D8	R/W	ECC 椭圆曲线参数CURVEN 字 8	0x0000_0000

CRYPTO_ECC_N_09	CRYPTO_BA+0x9DC	R/W	ECC 椭圆曲线参数CURVEN 字9	0x0000_0000
CRYPTO_ECC_N_10	CRYPTO_BA+0x9E0	R/W	ECC 椭圆曲线参数CURVEN 字 10	0x0000_0000
CRYPTO_ECC_N_11	CRYPTO_BA+0x9E4	R/W	ECC 椭圆曲线参数CURVEN 字 11	0x0000_0000
CRYPTO_ECC_N_12	CRYPTO_BA+0x9E8	R/W	ECC 椭圆曲线参数CURVEN 字 12	0x0000_0000
CRYPTO_ECC_N_13	CRYPTO_BA+0x9EC	R/W	ECC 椭圆曲线参数CURVEN 字 13	0x0000_0000
CRYPTO_ECC_N_14	CRYPTO_BA+0x9F0	R/W	ECC 椭圆曲线参数CURVEN 字 14	0x0000_0000
CRYPTO_ECC_N_15	CRYPTO_BA+0x9F4	R/W	ECC 椭圆曲线参数CURVEN 字 15	0x0000_0000
CRYPTO_ECC_N_16	CRYPTO_BA+0x9F8	R/W	ECC 椭圆曲线参数CURVEN 字 16	0x0000_0000
CRYPTO_ECC_N_17	CRYPTO_BA+0x9FC	R/W	ECC 椭圆曲线参数CURVEN 字 17	0x0000_0000
CRYPTO_ECC_K_00	CRYPTO_BA+0xA00	W	ECC 点乘标量 SCALARK 字 0	0x0000_0000
CRYPTO_ECC_K_01	CRYPTO_BA+0xA04	W	ECC 点乘标量 SCALARK 字 1	0x0000_0000
CRYPTO_ECC_K_02	CRYPTO_BA+0xA08	W	ECC 点乘标量 SCALARK 字 2	0x0000_0000
CRYPTO_ECC_K_03	CRYPTO_BA+0xA0C	W	ECC 点乘标量 SCALARK 字 3	0x0000_0000
CRYPTO_ECC_K_04	CRYPTO_BA+0xA10	W	ECC 点乘标量 SCALARK 字 4	0x0000_0000
CRYPTO_ECC_K_05	CRYPTO_BA+0xA14	W	ECC 点乘标量 SCALARK 字 5	0x0000_0000
CRYPTO_ECC_K_06	CRYPTO_BA+0xA18	W	ECC 点乘标量 SCALARK 字 6	0x0000_0000
CRYPTO_ECC_K_07	CRYPTO_BA+0xA1C	W	ECC 点乘标量 SCALARK 字 7	0x0000_0000
CRYPTO_ECC_K_08	CRYPTO_BA+0xA20	W	ECC 点乘标量 SCALARK 字 8	0x0000_0000
CRYPTO_ECC_K_09	CRYPTO_BA+0xA24	W	ECC 点乘标量 SCALARK 字 9	0x0000_0000
CRYPTO_ECC_K_10	CRYPTO_BA+0xA28	W	ECC 点乘标量 SCALARK 字 10	0x0000_0000
CRYPTO_ECC_K_11	CRYPTO_BA+0xA2C	W	ECC 点乘标量 SCALARK 字 11	0x0000_0000
CRYPTO_ECC_K_12	CRYPTO_BA+0xA30	W	ECC 点乘标量 SCALARK 字 12	0x0000_0000
CRYPTO_ECC_K_13	CRYPTO_BA+0xA34	W	ECC 点乘标量 SCALARK 字 13	0x0000_0000
CRYPTO_ECC_K_14	CRYPTO_BA+0xA38	W	ECC 点乘标量 SCALARK 字 14	0x0000_0000
CRYPTO_ECC_K_15	CRYPTO_BA+0xA3C	W	ECC 点乘标量 SCALARK 字 15	0x0000_0000
CRYPTO_ECC_K_16	CRYPTO_BA+0xA40	W	ECC 点乘标量 SCALARK 字 16	0x0000_0000
CRYPTO_ECC_K_17	CRYPTO_BA+0xA44	W	ECC 点乘标量 SCALARK 字 17	0x0000_0000
CRYPTO_ECC_SADDR	CRYPTO_BA+0xA48	R/W	ECC DMA源地址寄存器	0x0000_0000
CRYPTO_ECC_DADDR	CRYPTO_BA+0xA4C	R/W	ECC DMA 目的地址寄存器	0x0000_0000
CRYPTO_ECC_STARTR EG	CRYPTO_BA+0xA50	R/W	ECC 更新起始地址寄存器	0x0000_0000
CRYPTO_ECC_WORDCNT	CRYPTO_BA+0xA54	R/W	ECC DMA 字数寄存器	0x0000_0000

### 6.36.7 寄存器描述

#### 6.36.7.1 加密寄存器

##### **CRYPTO\_INTEN** 加密中断使能控制寄存器

寄存器	偏移量	R/W	描述	复位值
CRYPTO_INTEN	CRYPTO_BA+0x000	R/W	加密中断使能控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved						HMACEIEN	HMACIEN
23	22	21	20	19	18	17	16
ECCEIEN	ECCIEN	Reserved					
15	14	13	12	11	10	9	8
Reserved						TDESEIEN	TDESIEN
7	6	5	4	3	2	1	0
Reserved						AESEIEN	AESIEN

位	描述
[31:26]	<b>Reserved</b> 保留.
[25]	<b>HMACEIEN</b> <b>SHA/HMAC 错误中断使能控制</b> 0 = SHA/HMAC 错误中断标志禁止. 1 = SHA/HMAC 错误中断标志使能
[24]	<b>HMACIEN</b> <b>SHA/HMAC 中断使能控制</b> 0 = SHA/HMAC中断禁止. 1 = SHA/HMAC中断使能. 在 DMA 模式,当SHA_DMA_CNT设定数量的数据存入SHA/HMAC引擎后会产生一个中断, 在 非-DMA模式, 当SHA/HMAC 引擎完成操作以后会触发一个中断.
[23]	<b>ECCEIEN</b> <b>ECC 错误中断使能控制</b> 0 = ECC 错误终端标志禁止 1 = ECC 错误中断标志使能
[22]	<b>ECCIEN</b> <b>ECC 中断使能控制</b> 0 = ECC中断禁止. 1 = ECC中断使能. 在 DMA 模式, 当ECC_DMA_CNT 设定数量的数据参入ECC 引擎后会触发一个中断 在非-DMA 模式, 当 ECC 引擎完成操作后会触发一个中断
[21:17]	<b>Reserved</b> 保留.
[16]	<b>PRNGIEN</b> <b>PRNG 中断使能位</b> 0 = PRNG 中断禁止. 1 = PRNG 中断使能.

[15:10]	<b>Reserved</b>	<b>TDES/DES 错误标志使能控制</b> 0 = TDES/DES 错误中断标志禁止. 1 = TDES/DES 错误中断标志使能
[9]	<b>TDESEIEN</b>	<b>TDES/DES 中断使能控制</b> 0 = TDES/DES 中断禁止. 1 = TDES/DES 中断使能. 在 DMA 模式, 当TDES_DMA_CNT 设定数量的数据存入 TDES 引擎后会触发一个中断 在 非-DMA 模式, 当 TDES 引擎完成操作后会触发一个中断.
[8]	<b>TDESIEN</b>	保留.
[7:2]	<b>Reserved</b>	<b>AES 错误标志使能控制</b> 0 = AES 错误中断标志禁止. 1 = AES 错误中断标志使能.
[1]	<b>AESEIEN</b>	<b>AES 中断使能控制</b> 0 = AES 中断禁止. 1 = AES 中断使能. 在 DMA模式, 当AES_DMA_CNT 设定数量的数据存入AES 引擎会触发一个中断 在 非-DMA 模式, 当AES 引擎完成操作以后会触发一个中断.
[0]	<b>AESIEN</b>	保留.

**CRYPTO INTSTS 加密中断标志寄存器**

寄存器	偏移量	R/W	描述	复位值
CRYPTO_INTSTS	CRYPTO_BA+0x004	R/W	加密中断标志寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved						HMACEIF	HMACIF
23	22	21	20	19	18	17	16
ECCEIF	ECCIF	Reserved					
15	14	13	12	11	10	9	8
Reserved						TDESEIF	TDESIF
7	6	5	4	3	2	1	0
Reserved						AESEIF	AESIF

位	描述
[31:26]	<b>Reserved</b> 保留
[25]	<b>HMACEIF</b> <b>SHA/HMAC 错误标志</b> 这个寄存器包括操作和设置错误，具体的标志见CRYPTO_HMAC_STS寄存器。 此位写1清0，写0无效。 0 = 没有 SHA/HMAC 错误。 1 = SHA/HMAC 错误中断。
[24]	<b>HMACIF</b> <b>SHA/HMAC 完成中断标志</b> 此位写1清0，写0无效。 0 = 没有 SHA/HMAC 中断 1 = SHA/HMAC 操作完成中断。
[23]	<b>ECCEIF</b> <b>ECC 错误标志</b> 这个寄存器包括操作和设置错误，具体的标志见CRYPTO_ECC_STS寄存器。 此位写1清0，写0无效 0 = 没有 ECC 错误。 1 = ECC 错误中断。
[22]	<b>ECCIF</b> <b>ECC 完成中断标志</b> 此位写1清0，写0无效。 0 = 没有 ECC 中断。 1 = ECC 操作完成中断。
[21:17]	<b>Reserved</b> 保留。
[16]	<b>PRNGIF</b> <b>PRNG完成中断标志</b> 此位写1清0，写0无效。 0 = 没有 PRNG 中断。 1 = PRNG 密钥产生完成中断。

[15:10]	<b>Reserved</b>	保留.
[9]	<b>TDESEIF</b>	<b>TDES/DES 错误标志</b> 这个寄存器包括操作和设置错误，具体的标志见CRYPTO_TDES_STS寄存器。 此位写1清0，写0无效 0 = 没有 TDES/DES 错误. 1 = TDES/DES 加解密错误中断.
[8]	<b>TDESIF</b>	<b>TDES/DES 完成中断标志</b> 此位写1清0，写0无效 0 = 没有 TDES/DES中断 1 = TDES/DES 加解密完成中断
[7:2]	<b>Reserved</b>	保留.
[1]	<b>AESEIF</b>	<b>AES 错误标志</b> 此位写1清0，写0无效. 0 = 没有 AES错误. 1 = AES 加解密错误中断.
[0]	<b>AESIF</b>	<b>AES 完成中断标志</b> 此位写1清0，写0无效. 0 =没有 AES中断 1= AES加解密完成中断.

## 6.36.7.2 PRNG 寄存器

CRYPTO\_PRNG\_CTL PRNG 控制寄存器

寄存器	偏移量	R/W	描述	复位值
CRYPTO_PRNG_CTL	CRYPTO_BA+0x008	R/W	PRNG控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved				KEYSZ		SEEDRLD	
Reserved				KEYSZ		SEEDRLD	

位	描述
[31:9]	<b>Reserved</b>
	保留.
[8]	<b>BUSY</b>
	<b>PRNG 忙 (只读)</b> 0 = PRNG 引擎空闲. 1 = 表明 PRNG 引擎正在产生 CRYPTO_PRNG_KEYx.
[7:4]	<b>Reserved</b>
	保留
[3:2]	<b>KEYSZ</b>
	<b>PRNG 产生密钥大小</b> 00 = 64位. 01 = 128位. 10 = 192位. 11 = 256位.
[1]	<b>SEEDRLD</b>
	<b>PRNG 引擎重载新的种子</b> 0 = 根据目前种子产生密钥. 1 = 重载新的种子
[0]	<b>START</b>
	<b>开始 PRNG引擎</b> 0 = 停止 PRNG 引擎. 1 = 产生新的密钥，并把密钥存入寄存器CRYPTO_PRNG_KEYx (会在新密钥产生时清除)

CRYPTO PRNG SEED PRNG 种子寄存器

寄存器	偏移量	R/W	描述	复位值
CRYPTO_PRNG_SEED	CRYPTO_BA+0x00C	W	PRNG 种子	Undefined

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

位	描述	
[31:0]	SEED	PRNG种子(只写) 这里存放 PRNG 引擎的种子

## CRYPTO\_PRNG\_KEYx PRNG 密钥 x 寄存器

寄存器	偏移量	R/W	描述	复位值
CRYPTO_PRNG_KEY0	CRYPTO_BA+0x010	R	PRNG 产生密钥 0	Undefined
CRYPTO_PRNG_KEY1	CRYPTO_BA+0x014	R	PRNG 产生密钥 1	Undefined
CRYPTO_PRNG_KEY2	CRYPTO_BA+0x018	R	PRNG 产生密钥 2	Undefined
CRYPTO_PRNG_KEY3	CRYPTO_BA+0x01C	R	PRNG 产生密钥 3	Undefined
CRYPTO_PRNG_KEY4	CRYPTO_BA+0x020	R	PRNG 产生密钥 4	Undefined
CRYPTO_PRNG_KEY5	CRYPTO_BA+0x024	R	PRNG 产生密钥 5	Undefined
CRYPTO_PRNG_KEY6	CRYPTO_BA+0x028	R	PRNG 产生密钥 6	Undefined
CRYPTO_PRNG_KEY7	CRYPTO_BA+0x02C	R	PRNG 产生密钥 7	Undefined

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

位	描述		
[31:0]	KEY	存储 PRNG 产生的密钥 (只读) 这里存储 PRNG 产生的密钥.	

## 6.36.7.3 AES 寄存器

AES 控制寄存器 (CRYPTO\_AES\_CTL)

寄存器	偏移量	R/W	描述	复位值
CRYPTO_AES_CTL	CRYPTO_BA+0x100	R/W	AES 控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
KEYPRT	KEYUNPRT					CHANNEL	
23	22	21	20	19	18	17	16
INSWAP	OUTSWAP	Reserved					ENCRYPTO
15	14	13	12	11	10	9	8
OPMODE							
7	6	5	4	3	2	1	0
DMAEN	DMACSCAD	DMALAST	Reserved	KEYSZ		STOP	START

位	描述
[31]	<b>KEYPRT</b> 保护密钥 读到一个标志反映 KEYPRT. 0 = 没影响. 1 = 保护AES密钥的内容不被读取。读 CRYPTO_AESn_KEYx 寄存器的值不是 CRYPTO_AESn_KEYx寄存器的内容.当此位置1, 可以被KEYUNPRT清除, 但密钥的内容也会被清除
[30:26]	<b>KEYUNPRT</b> 不保护密钥 写 0 到 CRYPTO_AES_CTL[31]且写 “10110” 到 CRYPTO_AES_CTL[30:26] 来解除保护AES密钥。 KEYUNPRT 可以读可以写, 当AES 引擎正在操作时写入, BUSY标志是 1, 写入无效.
[25:24]	<b>CHANNEL</b> AES 引擎工作通道 00 = 目前控制寄存器设定用于通道 0. 01 = 目前控制寄存器设定用于通道1. 10 = 目前控制寄存器设定用于通道2. 11 = 目前控制寄存器设定用于通道 3.
[23]	<b>INSWAP</b> AES引擎输入数据交换 0 = 保持原有顺序. 1 = CPU 存入加速器的数据顺序从{byte3, byte2, byte1, byte0} 变为 {byte0, byte1, byte2, byte3}.
[22]	<b>OUTSWAP</b> AES 引擎输出数据交换 0 = 保持原有顺序. 1 = 从加速器存入 CPU 的数据顺序从 {byte3, byte2, byte1, byte0} 变为 {byte0, byte1, byte2, byte3}.
[17]	Reserved 保留.

[16]	<b>ENCRYPTO</b>	<b>AES 加密/解密</b> 0 = AES 引擎执行解密操作. 1 = AES 引擎执行加密操作.
[15:8]	<b>OPMODE</b>	<b>AES 引擎操作模式</b> 0x00 = ECB (电子密码本模式) 0x01 = CBC (密码块链接模式). 0x02 = CFB (密码反馈模式). 0x03 = OFB (输出反馈模式). 0x04 = CTR (计数器模式). 0x10 = CBC-CS1 (CBC 密文窃取模式 1). 0x11 = CBC-CS2 (CBC密文窃取模式2). 0x12 = CBC-CS3 (CBC密文窃取模式3).
[7]	<b>DMAEN</b>	<b>AES引擎 DMA 使能控制</b> 0 = AES DMA引擎禁止 AES 引擎工作在 非-DMA模式, 从端口 CRYPTO_AES_DATIN获取数据. 1 = AES_DMA引擎使能. AES 引擎工作在DMA模式,从引擎写入读出数据由DMA逻辑完成.
[6]	<b>DMACSCAD</b>	<b>AES 引擎 DMA 级联模式</b> 0 = DMA 级联模式禁止 1 = 在 DMA 级联模式,程序可以一次级联操作中更新 DMA s源地址寄存器, 目的地址寄存器, 和字节数寄存器,不需要等加速器操作完成
[5]	<b>DMALAST</b>	<b>AES 最后一个数据块</b> 在 DMA 模式,这一位需要在在开始最后DMA级联循环前置1. 在 非-DMA 模式, 在ECB, CBC, CTR, OFB, 和 CFB 模式下存入最后一笔数据时置1, 在 CBC-CS1, CBC-CS2, 个 CBC-CS3 模式下存入最后一笔的前一笔数据时置1. 此位总是读为0, .当 START 触发需要重新写入
[3:2]	<b>KEYSZ</b>	<b>AES 密钥大小</b> 这里定义三种不同 AES操作密钥大小. 2'b00 = 128 位密钥. 2'b01 = 192位密钥. 2'b10 = 256位密钥. 2'b11 = 保留. 如果 AES 加速器正在操作且相应的标志BUSY为 1, 更新此位无效
[1]	<b>STOP</b>	<b>AES 引擎停止</b> 0 =没效果. 1 = 停止 AES 引擎. <b>注意:</b> 此位总是读为0.
[0]	<b>START</b>	<b>AES 引擎开始</b> 0 = 没效果. 1 = 开始t AES 引擎. BUSY 标志置1. <b>注意:</b> 此位总是读为0.

## CRYPTO AES STS AES 状态标志寄存器

寄存器	偏移量	R/W	描述	复位值
CRYPTO_AES_STS	CRYPTO_BA+0x104	R	AES引擎标志	0x0001_0100

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved			BUSERR	Reserved	OUTBUFERR	OUTBUFFULL	OUTBUFEMPTY
15	14	13	12	11	10	9	8
Reserved			CNTERR	Reserved	INBUFERR	INBUFFULL	INBUFEMPTY
7	6	5	4	3	2	1	0
Reserved							BUSY

位	描述
[31:21]	<b>Reserved</b> 保留.
[20]	<b>BUSERR</b> <b>AES DMA 访问总线错误标志</b> 0 = 没有错误. 1 = 总线错误会停止 DMA操作和 AES 引擎.
[19]	<b>Reserved</b> 保留
[18]	<b>OUTBUFERR</b> <b>AES输出缓存错误标志</b> 0 = 没有错误. 1 = 从 AES 引擎获取数据时发生错误.
[17]	<b>OUTBUFFULL</b> <b>AES 输出缓存满标志</b> 0 = AES 输出缓存未满. 1 = AES 输出缓存已满, 程序需要从 CRYPTO_AES_DATOUT读取数据, 不然 AES 引擎会因为输出缓存满而等待
[16]	<b>OUTBUFEMPTY</b> <b>AES输出缓存空</b> 0 = AES 输出缓存非空。输出缓存里仍有有效数据. 1 = AES 输出缓存已空。程序不能再从CRYPTO_AES_DATOUT读数据, 标志 OUTBUFERR 当输出缓存空时会置1.
[15:13]	<b>Reserved</b> 保留
[12]	<b>CNTERR</b> <b>CRYPTO_AESn_CNT 设置错误</b> 0 = CRYPTO_AESn_CNT 设置没有错误. 1 = CRYPTO_AESn_CNT在 ECB, CBC, CFB, OFB, 和 CTR 模式下 (如果 CRYPTO_AES_CTL[7]使能) 没有乘以16.
[11]	<b>Reserved</b> 保留.

[10]	<b>INBUFERR</b>	<b>AES 输入缓存错误标志</b> 0 = 没有错误. 1 = 写数据到 AES 引擎时发生错误.
[9]	<b>INBUFFULL</b>	<b>AES 输入缓存满标志</b> 0 = AES 输入缓存未满. 程序可以向 AES 引擎写入数据. 1 = AES 输入缓存已满. 程序不能向 AES 引擎写入数据. 标志 INBUFERR 会置 1.
[8]	<b>INBUFEMPTY</b>	<b>AES 输入缓存空</b> 0 = 输入缓存里有一些数据等待 AES 引擎执行. 1 = AES 输入缓存空. 程序需要向 AES 引擎写入数据. 另外, AES 引擎会挂起等待输入数据.
[7:1]	<b>Reserved</b>	保留.
[0]	<b>BUSY</b>	<b>AES 引擎忙</b> 0 = AES 引擎在空闲或完成态. 1 = AES 引擎正在执行.

## CRYPTO AES DATIN AES 数据输入端口寄存器

寄存器	偏移量	R/W	描述	复位值
CRYPTO_AES_DATIN	CRYPTO_BA+0x108	R/W	AES引擎数据输入端口寄存器	0x0000_0000

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

位	描述	
[31:0]	DATIN	AES 引擎输入端口 CPU 通过这个端口向 AES 引擎存入数据，确认CRYPTO_AES_STS的状态，当 INBUFFULL 是 0时存入数据.

**CRYPTO AES DATOUT AES 数据输出端口寄存器**

寄存器	偏移量	R/W	描述	复位值
CRYPTO_AES_DATOUT	CRYPTO_BA+0x10C	R	AES 引擎数据输出端口寄存器	0x0000_0000

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

位	描述	
[31:0]	DATOUT	AES 引擎输出端口 CPU 从这里得到 AES 引擎输出的数据，需确认 CRYPTO_AES_STS 寄存器，当 OUTBUFEMPTY 为 0 时读取数据。

CRYPTO AES0 KEYx, CRYPTO AES1 KEYx, CRYPTO AES2 KEYx, CRYPTO AES3 KEYx  
AES密钥字 x 寄存器

寄存器	偏移量	R/W	描述	复位值
CRYPTO_AES0_KEY_0	CRYPTO_BA+0x110	R/W	AES 通道0 密钥字0 寄存器	0x0000_0000
CRYPTO_AES0_KEY_1	CRYPTO_BA+0x114	R/W	AES 通道0 密钥字1 寄存器	0x0000_0000
CRYPTO_AES0_KEY_2	CRYPTO_BA+0x118	R/W	AES 通道0 密钥字2 寄存器	0x0000_0000
CRYPTO_AES0_KEY_3	CRYPTO_BA+0x11C	R/W	AES 通道0 密钥字3 寄存器	0x0000_0000
CRYPTO_AES0_KEY_4	CRYPTO_BA+0x120	R/W	AES 通道0 密钥字4 寄存器	0x0000_0000
CRYPTO_AES0_KEY_5	CRYPTO_BA+0x124	R/W	AES 通道0 密钥字5 寄存器	0x0000_0000
CRYPTO_AES0_KEY_6	CRYPTO_BA+0x128	R/W	AES 通道0 密钥字6 寄存器	0x0000_0000
CRYPTO_AES0_KEY_7	CRYPTO_BA+0x12C	R/W	AES 通道0 密钥字7 寄存器	0x0000_0000
CRYPTO_AES1_KEY_0	CRYPTO_BA+0x14C	R/W	AES 通道1 密钥字0 寄存器	0x0000_0000
CRYPTO_AES1_KEY_1	CRYPTO_BA+0x150	R/W	AES 通道1 密钥字1 寄存器	0x0000_0000
CRYPTO_AES1_KEY_2	CRYPTO_BA+0x154	R/W	AES 通道1 密钥字2 寄存器	0x0000_0000
CRYPTO_AES1_KEY_3	CRYPTO_BA+0x158	R/W	AES 通道1 密钥字3 寄存器	0x0000_0000
CRYPTO_AES1_KEY_4	CRYPTO_BA+0x15C	R/W	AES 通道1 密钥字4 寄存器	0x0000_0000
CRYPTO_AES1_KEY_5	CRYPTO_BA+0x160	R/W	AES 通道1 密钥字5 寄存器	0x0000_0000
CRYPTO_AES1_KEY_6	CRYPTO_BA+0x164	R/W	AES 通道1 密钥字6 寄存器	0x0000_0000
CRYPTO_AES1_KEY_7	CRYPTO_BA+0x168	R/W	AES 通道1 密钥字7 寄存器	0x0000_0000
CRYPTO_AES2_KEY_0	CRYPTO_BA+0x188	R/W	AES 通道2 密钥字0 寄存器	0x0000_0000
CRYPTO_AES2_KEY_1	CRYPTO_BA+0x18C	R/W	AES 通道2 密钥字1 寄存器	0x0000_0000
CRYPTO_AES2_KEY_2	CRYPTO_BA+0x190	R/W	AES 通道2 密钥字2 寄存器	0x0000_0000
CRYPTO_AES2_KEY_3	CRYPTO_BA+0x194	R/W	AES 通道2 密钥字3 寄存器	0x0000_0000
CRYPTO_AES2_KEY_4	CRYPTO_BA+0x198	R/W	AES 通道2 密钥字4 寄存器	0x0000_0000
CRYPTO_AES2_KEY_5	CRYPTO_BA+0x19C	R/W	AES 通道2 密钥字5 寄存器	0x0000_0000

CRYPTO_AES2_KEY_6	CRYPTO_BA+0x1A0	R/W	AES 通道2 密钥字6 寄存器	0x0000_0000
CRYPTO_AES2_KEY_7	CRYPTO_BA+0x1A4	R/W	AES 通道2 密钥字7 寄存器	0x0000_0000
CRYPTO_AES3_KEY_0	CRYPTO_BA+0x1C4	R/W	AES 通道3 密钥字0 寄存器	0x0000_0000
CRYPTO_AES3_KEY_1	CRYPTO_BA+0x1C8	R/W	AES 通道3 密钥字1寄存器	0x0000_0000
CRYPTO_AES3_KEY_2	CRYPTO_BA+0x1CC	R/W	AES 通道3 密钥字2 寄存器	0x0000_0000
CRYPTO_AES3_KEY_3	CRYPTO_BA+0x1D0	R/W	AES 通道3 密钥字3 寄存器	0x0000_0000
CRYPTO_AES3_KEY_4	CRYPTO_BA+0x1D4	R/W	AES 通道3 密钥字4 寄存器	0x0000_0000
CRYPTO_AES3_KEY_5	CRYPTO_BA+0x1D8	R/W	AES 通道3 密钥字5 寄存器	0x0000_0000
CRYPTO_AES3_KEY_6	CRYPTO_BA+0x1DC	R/W	AES 通道3 密钥字6 寄存器	0x0000_0000
CRYPTO_AES3_KEY_7	CRYPTO_BA+0x1E0	R/W	AES 通道3 密钥字7 寄存器	0x0000_0000

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

位	描述	
[31:0]	KEY	<p><b>CRYPTO_AESn_KEYx</b></p> <p>KEY 保存AES 操作的安全密钥。</p> <p>n = 0, 1..3.</p> <p>x = 0, 1..7.</p> <p>AES 加速器的安全密钥可以是128, 192, 或 256位, 需要4, 6, 8个32位寄存器来存放。</p> <p>{CRYPTO_AESn_KEY3, CRYPTO_AESn_KEY2, CRYPTO_AESn_KEY1, CRYPTO_AESn_KEY0}存放128位的AES操作安全密钥。{CRYPTO_AESn_KEY5, CRYPTO_AESn_KEY4, CRYPTO_AESn_KEY3, CRYPTO_AESn_KEY2, CRYPTO_AESn_KEY1, CRYPTO_AESn_KEY0} 存放 192位AES 操作安全密钥。</p> <p>{CRYPTO_AESn_KEY7, CRYPTO_AESn_KEY6, CRYPTO_AESn_KEY5, CRYPTO_AESn_KEY4, CRYPTO_AESn_KEY3, CRYPTO_AESn_KEY2, CRYPTO_AESn_KEY1, CRYPTO_AESn_KEY0} 存放 256位 AES操作安全密钥。</p>

CRYPTO AES0 IVx, CRYPTO AES1 IVx, CRYPTO AES2 IVx, CRYPTO AES3 IVx      AES 初  
化向量字 x 寄存器

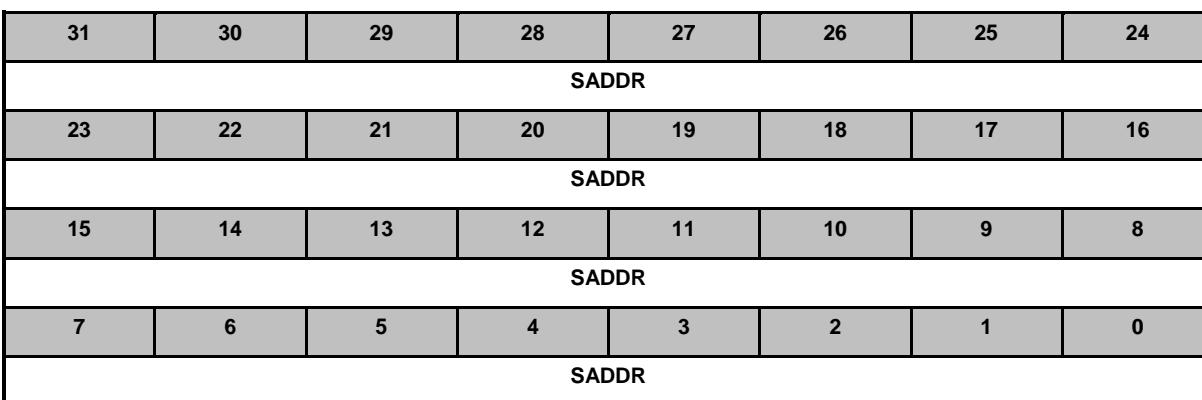
寄存器	偏移量	R/W	描述	复位值
CRYPTO_AES0_I_V0	CRYPTO_BA+0x130	R/W	AES 通道0初始化向量字0寄存器	0x0000_0000
CRYPTO_AES0_I_V1	CRYPTO_BA+0x134	R/W	AES 通道0初始化向量字1寄存器	0x0000_0000
CRYPTO_AES0_I_V2	CRYPTO_BA+0x138	R/W	AES 通道0初始化向量字2寄存器	0x0000_0000
CRYPTO_AES0_I_V3	CRYPTO_BA+0x13C	R/W	AES 通道0初始化向量字3寄存器	0x0000_0000
CRYPTO_AES1_I_V0	CRYPTO_BA+0x16C	R/W	AES 通道1初始化向量字0寄存器	0x0000_0000
CRYPTO_AES1_I_V1	CRYPTO_BA+0x170	R/W	AES 通道1初始化向量字1寄存器	0x0000_0000
CRYPTO_AES1_I_V2	CRYPTO_BA+0x174	R/W	AES 通道1初始化向量字2寄存器	0x0000_0000
CRYPTO_AES1_I_V3	CRYPTO_BA+0x178	R/W	AES 通道1初始化向量字3寄存器	0x0000_0000
CRYPTO_AES2_I_V0	CRYPTO_BA+0x1A8	R/W	AES 通道2初始化向量字0寄存器	0x0000_0000
CRYPTO_AES2_I_V1	CRYPTO_BA+0x1AC	R/W	AES 通道2初始化向量字1寄存器	0x0000_0000
CRYPTO_AES2_I_V2	CRYPTO_BA+0x1B0	R/W	AES 通道2初始化向量字2寄存器	0x0000_0000
CRYPTO_AES2_I_V3	CRYPTO_BA+0x1B4	R/W	AES 通道2初始化向量字3寄存器	0x0000_0000
CRYPTO_AES3_I_V0	CRYPTO_BA+0x1E4	R/W	AES 通道3初始化向量字0寄存器	0x0000_0000
CRYPTO_AES3_I_V1	CRYPTO_BA+0x1E8	R/W	AES 通道3初始化向量字1寄存器	0x0000_0000
CRYPTO_AES3_I_V2	CRYPTO_BA+0x1EC	R/W	AES 通道3初始化向量字2寄存器	0x0000_0000
CRYPTO_AES3_I_V3	CRYPTO_BA+0x1F0	R/W	AES 通道3初始化向量字3寄存器	0x0000_0000

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

位	描述
[31:0]	<b>IV</b> <b>AES 初始向量</b> $n = 0, 1..3.$ $x = 0, 1..3.$ 4个初始化向量 (CRYPTO_AESn_IV0, CRYPTO_AESn_IV1, CRYPTO_AESn_IV2, 和 CRYPTO_AESn_IV3) 是AES工作于CBC, CFB, 和 OFB模式需要的. 4个寄存器 (CRYPTO_AESn_IV0, CRYPTO_AESn_IV1, CRYPTO_AESn_IV2, 和 CRYPTO_AESn_IV3) 在 AES 工作在 CTR 模式时做为临时计数器.

CRYPTO\_AES0\_SADDR,      CRYPTO\_AES1\_SADDR,      CRYPTO\_AES2\_SADDR,  
CRYPTO AES3 SADDR    AES DMA 源地址寄存器

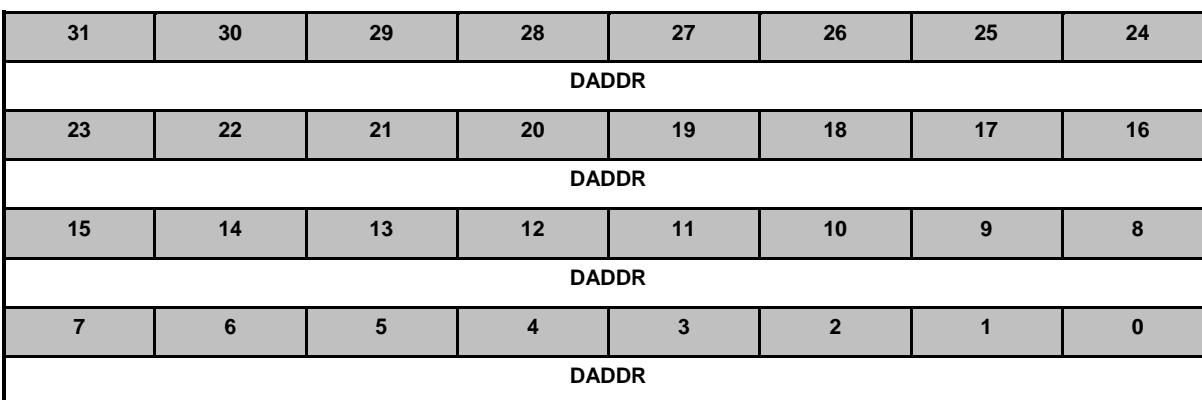
寄存器	偏移量	R/W	描述	复位值
CRYPTO_AES0_SADDR	CRYPTO_BA+0x140	R/W	AES DMA通道 0源地址寄存器	0x0000_0000
CRYPTO_AES1_SADDR	CRYPTO_BA+0x17C	R/W	AES DMA通道 1源地址寄存器	0x0000_0000
CRYPTO_AES2_SADDR	CRYPTO_BA+0x1B8	R/W	AES DMA通道 2源地址寄存器	0x0000_0000
CRYPTO_AES3_SADDR	CRYPTO_BA+0x1F4	R/W	AES DMA通道 3源地址寄存器	0x0000_0000



位	描述	
[31:0]	SADDR	<p><b>AES DMA 源地址</b></p> <p>AES 加速器支持 DMA 功能在SRAM内存和内建FIFO之间传输原始文本。 SADDR存放源文本存放的数据缓存的源地址。基于源地址, AES 加速器可以从SRAM内存读到原始文本(加密)/加密文本(解密)来执行AES 操作. 源地址必须是字对齐, 也就是说, SADDR位0和位1会被忽略。</p> <p>SADDR 可读可写, 当AES正在操作时写SADDR 不会影响目前的 AES操作, 但 SADDR 寄存器的值会在稍后更新。因此, 程序可以为下一次AES 操作准备源地址。</p> <p>在 DMA 模式, 程序可以在触发START 前更新下一笔CRYPTO_AESn_SADDR.</p> <p>CRYPTO_AESn_SADDR 和 CRYPTO_AESn_DADDR 的值可以相同。</p>

CRYPTO\_AES0\_DADDR,      CRYPTO\_AES1\_DADDR,      CRYPTO\_AES2\_DADDR,  
CRYPTO AES3 DADDR AES DMA 目的地址寄存器

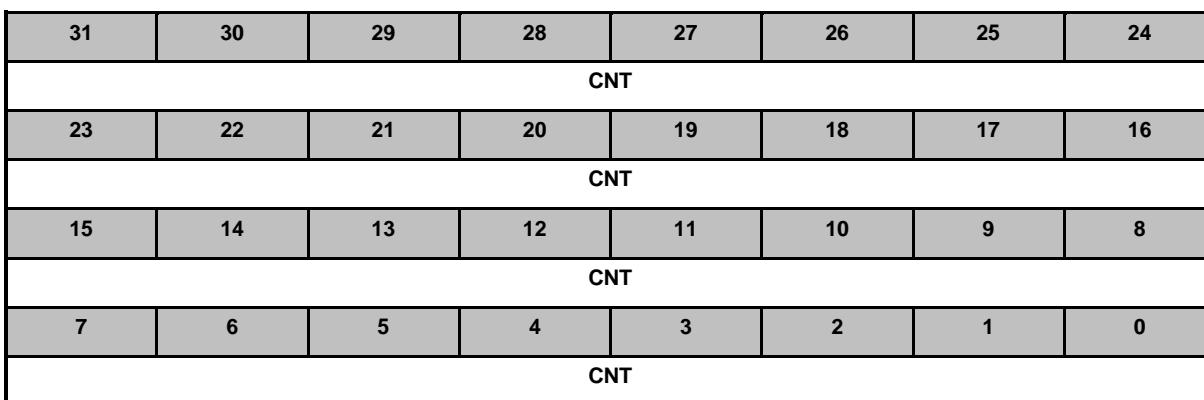
寄存器	偏移量	R/W	描述	复位值
CRYPTO_AES0_DAD DR	CRYPTO_BA+0x144	R/W	AES通道0 DMA目的地址寄存器	0x0000_0000
CRYPTO_AES1_DAD DR	CRYPTO_BA+0x180	R/W	AES通道1DMA目的地址寄存器	0x0000_0000
CRYPTO_AES2_DAD DR	CRYPTO_BA+0x1B C	R/W	AES通道2 DMA目的地址寄存器	0x0000_0000
CRYPTO_AES3_DAD DR	CRYPTO_BA+0x1F 8	R/W	AES通道3 DMA目的地址寄存器	0x0000_0000



位	描述
[31:0]	<b>DADDR</b> <b>AES DMA 目的地地址</b> AES 加速器支持DMA 功能来在SRAM内存和内建FIFO间传输加密文本。. DADDR存放引擎输出文本存放的数据缓存的目的地址 。基于目的地址， AES 加速器可以在AES 操作完成之后将加密文本(加密)/原始文本(解密)写回SRAM内存。目的地址的开始必须是字对齐， DADDR 的位0和位1会被忽略。 DADDR 可读可写。在AES正在操作时写入DADDR 不会影响目前的AES操作，但DADDR 会在稍后更新，因此程序可以为下次AES操作准备目的地址的值.. 在DMA 模式, 程序可以在触发START 前更新下一个CRYPTO_AESn_DADDR. CRYPTO_AESn_SADDR 和 CRYPTO_AESn_DADDR 的值可以相同。

**CRYPTO\_AES0\_CNT, CRYPTO\_AES1\_CNT, CRYPTO\_AES2\_CNT, CRYPTO\_AES3\_CNT AES字节数寄存器**

寄存器	偏移量	R/W	描述	复位值
CRYPTO_AES0_CNT	CRYPTO_BA+0x148	R/W	AES 通道0字节数寄存器	0x0000_0000
CRYPTO_AES1_CNT	CRYPTO_BA+0x184	R/W	AES 通道1字节数寄存器	0x0000_0000
CRYPTO_AES2_CNT	CRYPTO_BA+0x1C0	R/W	AES 通道2字节数寄存器	0x0000_0000
CRYPTO_AES3_CNT	CRYPTO_BA+0x1FC	R/W	AES 通道3字节数寄存器	0x0000_0000



位	描述	
[31:0]	CNT	<p><b>AES 字节数</b></p> <p>在 AES 引擎工作在 DMA 模式时，CRYPTO_AESn_CNT 存放源文本的字节数，CRYPTO_AESn_CNT 是32位，所以最大字节数是 4G。</p> <p>CRYPTO_AESn_CNT 可读可写。当 AES 加速器正在操作时写入CRYPTO_AESn_CNT 不会影响目前的AES操作。但是CRYPTO_AESn_CNT 会在稍后更新。因此，程序可以为下一次AES操作准备字节数。</p> <p>依照 CBC-CS1, CBC-CS2, 和 CBC-CS3 标准，操作数据的个数最少要一个块。少于一个块会导致不可预料的结果。</p> <p>在非-DMA ECB, CBC, CFB, OFB, 和 CTR 模式，CRYPTO_AESn_CNT 在存入最后一个数据块的数据前必须写入最后一个数据块的字节数。在非-DMA CBC-CS1, CBC-CS2, 和 CBC-CS3 模式，CRYPTO_AESn_CNT 在存入最后两个数据块的数据前必须写入最后两个数据块的字节数。</p>

**CRYPTO AES\_FDBCKx AES反馈 x 寄存器**

寄存器	偏移量	R/W	描述	复位值
CRYPTO_AES_FDBCK0	CRYPTO_BA+0x050	R	AES 引擎加密操作后输出反馈数据	0x0000_0000
CRYPTO_AES_FDBCK1	CRYPTO_BA+0x054	R	AES 引擎加密操作后输出反馈数据	0x0000_0000
CRYPTO_AES_FDBCK2	CRYPTO_BA+0x058	R	AES 引擎加密操作后输出反馈数据	0x0000_0000
CRYPTO_AES_FDBCK3	CRYPTO_BA+0x05C	R	AES 引擎加密操作后输出反馈数据	0x0000_0000

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

位	描述	
[31:0]	FDBCK	<p><b>AES反馈信息</b> 反馈值是 128 位大小。</p> <p>在 DMA 级联模式 AES 引擎使用 CRYPTO_AES_FDBCKx 的数据作为输入到 CRYPTO_AESn_IVx 的下一个块。</p> <p>AES 引擎输出下次快操作的IV的反馈信息。程序可以用这个反馈信息实现多于4个DMA 通道。程序可以暂存反馈值。当切换回来，向同一个通道的这个寄存器写入这个暂存值，下面的操作会依照当初的设定。</p>

## 6.36.7.4 TDES/DES 寄存器

CRYPTO\_TDES\_CTL TDES/DES 控制寄存器

寄存器	偏移量	R/W	描述	复位值
CRYPTO_TDES_CTL	CRYPTO_BA+0x200	R/W	TDES/DES 控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
KEYPRT	KEYUNPRT					CHANNEL	
23	22	21	20	19	18	17	16
INSWAP	OUTSWAP	BLKSWAP	Reserved				ENCRYPTO
15	14	13	12	11	10	9	8
Reserved					OPMODE		
7	6	5	4	3	2	1	0
DMAEN	DMACSCAD	DMALAST	Reserved	3KEYS	TMODE	STOP	START

位	描述
[31]	<b>KEYPRT</b> 保护密钥 读到 KEYPRT的标志。 0 = 没影响。 1 = 这一位保护TDES密钥不被读出。读 CRYPTO_TDESn_KEYxH/L 读到的值不是 CRYPTO_TDESn_KEYxH/的内容。当此位置1，可以通过访问 KEYUNPRT清0，密钥的内容也同时被清除。
[30:26]	<b>KEYUNPRT</b> 不保护密钥 写 0 到 CRYPTO_TDES_CTL [31] 写 “10110” 到 CRYPTO_TDES_CTL [30:26] 解保护 TDES 密钥。 KEYUNPRT 可读写，当TDES 引擎正在操作时写入， BUSY 标志是 1,不会改变 KEYUNPRT。
[25:24]	<b>CHANNEL</b> <b>TDES/DES 引擎工作通道</b> 00 = 目前控制寄存器设定为通道 0. 01 = 目前控制寄存器设定为通道 1. 10 = 目前控制寄存器设定为通道 2. 11 = 目前控制寄存器设定为通道 3.
[23]	<b>INSWAP</b> <b>TDES/DES 引擎输入数据交换</b> 0 = 保持原始顺序。 1 = CPU 存入加速器的数据顺序从{byte3, byte2, byte1, byte0} 变为 {byte0, byte1, byte2, byte3}.
[22]	<b>OUTSWAP</b> <b>TDES/DES 引擎输出数据交换</b> 0 = 保持原始顺序。 1 = CPU 输出数据到加速器的顺序从 {byte3, byte2, byte1, byte0} 变为 {byte0, byte1, byte2, byte3}.

[21]	<b>BLKSWAP</b>	<b>TDES/DES 引擎模块双字交换</b> 0 = 保持原始顺序. {WORD_H, WORD_L}. 1 = 当此位置1, TDES 改变高低位顺序 {WORD_L, WORD_H}.
[20:17]	<b>Reserved</b>	保留.
[16]	<b>ENCRYPTO</b>	<b>TDES/DES 加密/解密</b> 0 = TDES引擎执行解密操作. 1 = TDES 引擎执行加密操作.
[15:11]	<b>Reserved</b>	保留.
[10:8]	<b>OPMODE</b>	<b>TDES/DES 引擎工作模式</b> 0x00 = ECB (电子密码本模式). 0x01 = CBC (密码块链接模式). 0x02 = CFB (密码反馈模式). 0x03 = OFB (输出反馈模式). 0x04 = CTR (计数器模式). Others = CTR (计数器模式).
[7]	<b>DMAEN</b>	<b>TDES/DES 引擎 DMA 使能控制</b> 0 = TDES_DMA引擎禁止. TDES 引擎工作在非-DMA 模式, 从端口CRYPTO_TDES_DATIN读取数据. 1 = TDES_DMA 引擎使能. TDES 引擎工作在 DMA 模式, 数据搬运由 DMA 逻辑完成.
[6]	<b>DMACSCAD</b>	<b>TDES/DES 引擎 DMA 级联模式</b> 0 = DMA 级联功能禁止. 1 = 在 DMA 级联模式, 程序可以在没有完成加速器操作前更新DMA 源地址寄存器, 目的地址寄存器, 字节数寄存器
[5]	<b>DMALAST</b>	<b>TDES/DES 引擎开始最后一个块</b> 在 DMA 模式, 这一位必须在开始最后一笔DMA 级联循环前置1. 在 非-DMA 模式, 这一位在存入最后一个块前置1.
[4]	<b>Reserved</b>	保留.
[3]	<b>3KEYS</b>	<b>TDES/DES 密钥编号</b> 0 = TDES/DES引擎选择KEY1 和 KEY2. 1 = TDES/DES引擎三个密钥都使能.
[2]	<b>TMODE</b>	<b>TDES/DES 引擎操作模式</b> 0 = TDES/DES引擎设置 DES模式. 1 = TDES/DES引擎设置三重 DES 模式.
[1]	<b>STOP</b>	<b>TDES/DES 引擎停止</b> 0 = 没影响. 1 = 停止 TDES/DES 引擎. <b>注意:</b> 读此位总为0.

[0]	<b>START</b>	TDES/DES 引擎开始 0 = 没影响. 1 = 开始 TDES/DES 引擎.标志 BUSY 会置1. 注意: 读此位总为0.
-----	--------------	---

## CRYPTO\_TDES0\_STS TDES/DES 状态标志寄存器

寄存器	偏移量	R/W	描述	复位值
CRYPTO_TDES_STS	CRYPTO_BA+0x204	R	TDES/DES 引擎标志	0x0001_0100

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved			BUSERR	Reserved	OUTBUFERR	OUTBUFFULL	OUTBUFEMPTY
15	14	13	12	11	10	9	8
Reserved					INBUFERR	INBUFFULL	INBUFEMPTY
7	6	5	4	3	2	1	0
Reserved							BUSY

位	描述
[31:21]	<b>Reserved</b> 保留.
[20]	<b>BUSERR</b> <b>TDES/DES DMA 访问总线出错标志</b> 0 = 没有错误 1 = 总线错误会停止 DMA操作和 TDES/DES引擎.
[19]	<b>Reserved</b> 保留.
[18]	<b>OUTBUFERR</b> <b>TDES/DES输出缓存错误标志</b> 0 = 没有错误. 1 = 从TDES/DES 引擎读出测试结果时出错.
[17]	<b>OUTBUFFULL</b> <b>TDES/DES 输出缓存满标志</b> 0 = TDES/DES 输出缓存未满. 1 = TDES/DES 输出缓存满了,程序需要从 TDES_DATA_OUT读出数据。. DES/DES 引擎会挂起因为输出缓存满了.
[16]	<b>OUTBUFEMPTY</b> <b>TDES/DES 输出换成空标志</b> 0 = TDES/DES 输出缓存非空. 输出缓存里有一些有效数据. 1 = TDES/DES 输出缓存空, 程序不能从 TDES_DATA_OUT读取数据。 标志 OUTBUFERR会置 1, 因为输出缓存空了.
[15:11]	<b>Reserved</b> 保留.
[10]	<b>INBUFERR</b> <b>TDES/DES输入缓存错误标志</b> 0 = 没有错误. 1 = 数据存入 TDES/DES引擎,时发生错误
[9]	<b>INBUFFULL</b> <b>TDES/DES输入缓存满标志</b> 0 = TDES/DES输入缓存未满. 程序可以向 TDES/DES引擎填入数据. 1 = TDES输入缓存满了. 程序不能向 TDES/DES 引擎填入数据. 标志 INBUFERR 会置 1.

[8]	<b>INBUFEMPTY</b>	<b>TDES/DES 输入缓存空</b> 0 = 输入缓存中仍有一些数据等待 TDES/DES 引擎执行. 1 = TDES/DES 输入缓存空. 程序需要填入数据到TDES/DES引擎 TDES/DES 会挂起等待输入数据
[7:1]	<b>Reserved</b>	保留.
[0]	<b>BUSY</b>	<b>TDES/DES 引擎忙</b> 0 = TDES/DES 引擎在空闲或完成态. 1 = TDES/DES 引擎正在执行.

TDES KEY1H/L, TDES KEY2H/L, TDES KEY3H/L TDES/DES 密钥 1, 2, 3 高低字寄存器

寄存器	偏移量	R/W	描述	复位值
CRYPTO_TDES0_KEY1_H	CRYPTO_BA+0x208	R/W	TDES/DES 通道0 密钥 1 高位字	0x0000_0000
CRYPTO_TDES0_KEY1_L	CRYPTO_BA+0x20C	R/W	TDES/DES 通道0 密钥 1 低位字	0x0000_0000
CRYPTO_TDES0_KEY2_H	CRYPTO_BA+0x210	R/W	TDES/DES 通道0 密钥 2 高位字	0x0000_0000
CRYPTO_TDES0_KEY2_L	CRYPTO_BA+0x214	R/W	TDES/DES 通道0 密钥 2 低位字	0x0000_0000
CRYPTO_TDES0_KEY3_H	CRYPTO_BA+0x218	R/W	TDES/DES 通道0 密钥 3 高位字	0x0000_0000
CRYPTO_TDES0_KEY3_L	CRYPTO_BA+0x21C	R/W	TDES/DES 通道0 密钥 3 低位字	0x0000_0000
CRYPTO_TDES1_KEY1_H	CRYPTO_BA+0x248	R/W	TDES/DES 通道1 密钥 1 高位字	0x0000_0000
CRYPTO_TDES1_KEY1_L	CRYPTO_BA+0x24C	R/W	TDES/DES 通道1 密钥 1 低位字	0x0000_0000
CRYPTO_TDES1_KEY2_H	CRYPTO_BA+0x250	R/W	TDES/DES 通道1 密钥 2 高位字	0x0000_0000
CRYPTO_TDES1_KEY2_L	CRYPTO_BA+0x254	R/W	TDES/DES 通道1 密钥 2 低位字	0x0000_0000
CRYPTO_TDES1_KEY3_H	CRYPTO_BA+0x258	R/W	TDES/DES 通道1 密钥 3 高位字	0x0000_0000
CRYPTO_TDES1_KEY3_L	CRYPTO_BA+0x25C	R/W	TDES/DES 通道1 密钥 3 低位字	0x0000_0000
CRYPTO_TDES2_KEY1_H	CRYPTO_BA+0x288	R/W	TDES/DES 通道2 密钥 1 高位字	0x0000_0000
CRYPTO_TDES2_KEY1_L	CRYPTO_BA+0x28C	R/W	TDES/DES 通道2 密钥 1 低位字	0x0000_0000
CRYPTO_TDES2_KEY2_H	CRYPTO_BA+0x290	R/W	TDES/DES 通道2 密钥 2 高位字	0x0000_0000
CRYPTO_TDES2_KEY2_L	CRYPTO_BA+0x294	R/W	TDES/DES 通道2 密钥 2 低位字	0x0000_0000
CRYPTO_TDES2_KEY3_H	CRYPTO_BA+0x298	R/W	TDES/DES 通道2 密钥 3 高位字	0x0000_0000
CRYPTO_TDES2_KEY3_L	CRYPTO_BA+0x29C	R/W	TDES/DES 通道2 密钥 3 低位字	0x0000_0000
CRYPTO_TDES3_KEY1_H	CRYPTO_BA+0x2C8	R/W	TDES/DES 通道3 密钥 1 高位字	0x0000_0000
CRYPTO_TDES3_KEY1_L	CRYPTO_BA+0x2CC	R/W	TDES/DES 通道3 密钥 1 低位字	0x0000_0000
CRYPTO_TDES3_KEY2_H	CRYPTO_BA+0x2D0	R/W	TDES/DES 通道3 密钥 2 高位字	0x0000_0000
CRYPTO_TDES3_KEY2_L	CRYPTO_BA+0x2D4	R/W	TDES/DES 通道3 密钥 2 低位字	0x0000_0000

<b>CRYPTO_TDES3_KEY3_H</b>	CRYPTO_BA+0x2D8	R/W	TDES/DES 通道3 密钥 3 高位字	0x0000_0000
<b>CRYPTO_TDES3_KEY3_L</b>	CRYPTO_BA+0x2DC	R/W	TDES/DES 通道3 密钥 3 低位字	0x0000_0000

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

位	描述	
[31:0]	<b>KEY</b>	<p><b>TDES/DES 密钥 高/低 字</b>            TDES/DES 算法运算密钥寄存器            TDES/DES 加速器的安全密钥是64位. T需要两个 32-位寄存器存储安全密钥. 寄存器 CRYPTO_TDESn_KEYxH用来存放位 [63:32] , 寄存器 CRYPTO_TDESn_KEYxL用来存放位 [31:0].</p>

CRYPTO\_TDES0\_IVH/L,  
CRYPTO\_TDES3\_IVH/L
CRYPTO\_TDES1\_IVH/L,CRYPTO\_TDES2\_IVH/L,

TDES/DES IV 高低字寄存器

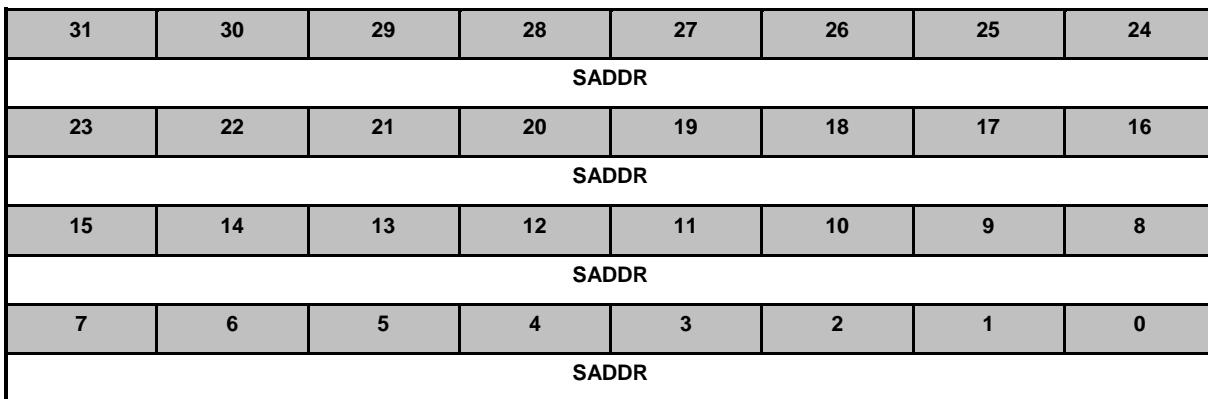
寄存器	偏移量	R/W	描述	复位值
CRYPTO_TDES0_IVH	CRYPTO_BA+0x220	R/W	TDES/DES 通道0 初始化向量高字寄存器	0x0000_0000
CRYPTO_TDES0_IVL	CRYPTO_BA+0x224	R/W	TDES/DES 通道0 初始化向量低字寄存器	0x0000_0000
CRYPTO_TDES1_IVH	CRYPTO_BA+0x260	R/W	TDES/DES 通道1 初始化向量高字寄存器	0x0000_0000
CRYPTO_TDES1_IVL	CRYPTO_BA+0x264	R/W	TDES/DES 通道1 初始化向量低字寄存器	0x0000_0000
CRYPTO_TDES2_IVH	CRYPTO_BA+0x2A0	R/W	TDES/DES 通道2 初始化向量高字寄存器	0x0000_0000
CRYPTO_TDES2_IVL	CRYPTO_BA+0x2A4	R/W	TDES/DES 通道2 初始化向量低字寄存器	0x0000_0000
CRYPTO_TDES3_IVH	CRYPTO_BA+0x2E0	R/W	TDES/DES 通道3 初始化向量高字寄存器	0x0000_0000
CRYPTO_TDES3_IVL	CRYPTO_BA+0x2E4	R/W	TDES/DES 通道3 初始化向量低字寄存器	0x0000_0000

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

位	描述	
[31:0]	IV	TDES/DES 初始化向量高/低字 初始化向量 (IV) 在 TDES/DES 引擎的 CBC, CFB, 和 OFB 模式时使用. IV 在 CTR 模式作为临时计数器使用.

CRYPTO\_TDES0\_SADDR, CRYPTO\_TDES1\_SADDR, CRYPTO\_TDES2\_SADDR,  
CRYPTO\_TDES3\_SADDR TDES/DES DMA 源地址寄存器

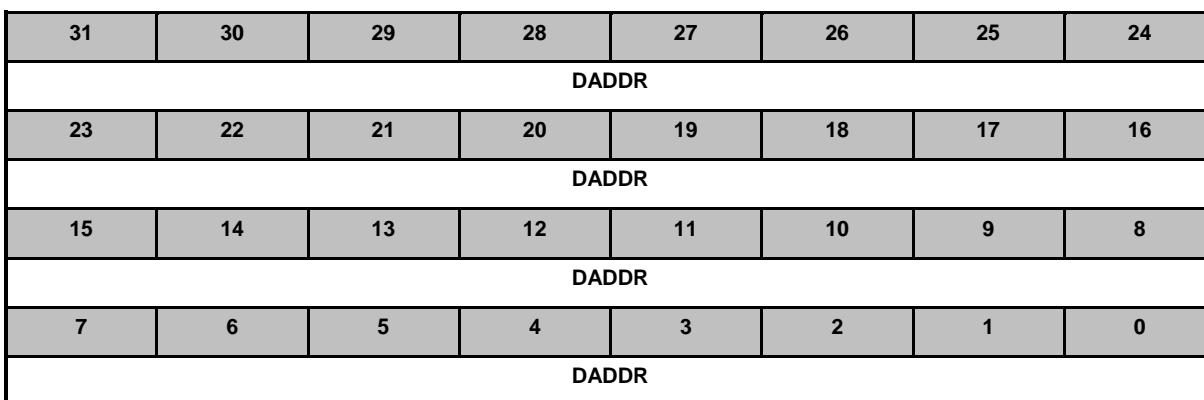
寄存器	偏移量	R/W	描述	复位值
CRYPTO_TDES0_SADDR	CRYPTO_BA+0x28	R/W	TDES/DES 通道0 DMA源地址寄存器	0x0000_0000
CRYPTO_TDES1_SADDR	CRYPTO_BA+0x268	R/W	TDES/DES 通道1 DMA源地址寄存器	0x0000_0000
CRYPTO_TDES2_SADDR	CRYPTO_BA+0x2A8	R/W	TDES/DES 通道2 DMA源地址寄存器	0x0000_0000
CRYPTO_TDES3_SADDR	CRYPTO_BA+0x2E8	R/W	TDES/DES 通道3 DMA源地址寄存器	0x0000_0000



位	描述
[31:0]	<b>SADDR</b> <b>TDES/DES DMA 源地址</b> TDES/DES 加速器支持 DMA 功能来在 SRAM 内存和内建 FIFO 间传输原始文本。CRYPTO_TDESn_SADDR 存放原始文本存放的数据缓存的源地址。基于这个源地址，TDES/DES 加速器可以从SRAM内存读到原始文本(加密)/加密文本(解密)并且执行 TDES/DES 操作. 源地址起始地址需要字对齐，就是说CRYPTO_TDESn_SADDR的位1和位0会被忽略。 CRYPTO_TDESn_SADDR 可读可写，在 TDES/DES 加速器正在操作时写 CRYPTO_TDESn_SADDR 不会影响目前的 TDES/DES 操作. 但是 CRYPTO_TDESn_SADDR 的值会在稍后更新，因此，程序可以为下次TDES/DES 操作准备 DMA 源地址。 在 DMA 模式，程序可以在触发START 之前更新下一笔 CRYPTO_TDESn_SADDR. CRYPTO_TDESn_SADDR 和 CRYPTO_TDESn_DADDR 值可以相同。

CRYPTO\_TDES0\_DADDR, CRYPTO\_TDES1\_DADDR, CRYPTO\_TDES2\_DADDR,  
CRYPTO\_TDES3\_DADDR TDES/DES DMA 目的地址寄存器

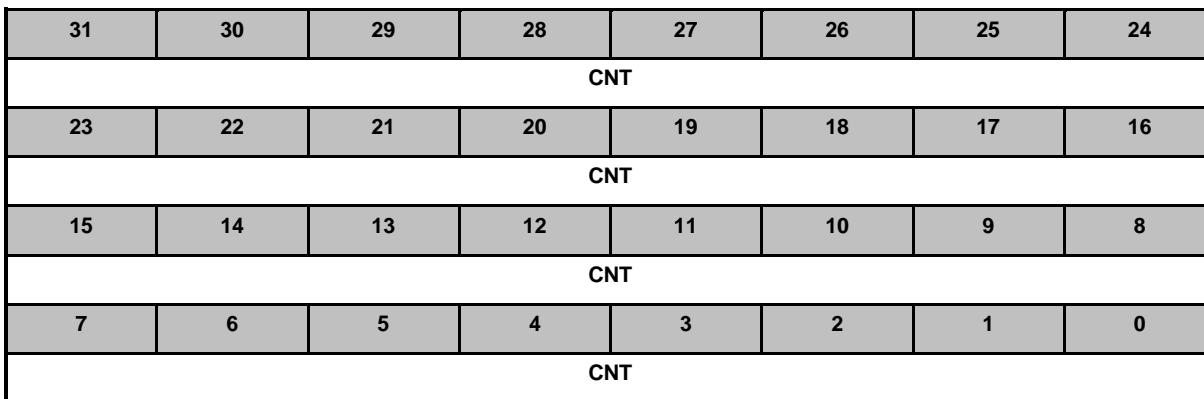
寄存器	偏移量	R/W	描述	复位值
CRYPTO_TDES0_DAD DR	CRYPTO_BA+0x22C	R/W	TDES/DES 通道0 DMA 目的地址寄存器	0x0000_0000
CRYPTO_TDES1_DAD DR	CRYPTO_BA+0x26C	R/W	TDES/DES 通道1 DMA 目的地址寄存器	0x0000_0000
CRYPTO_TDES2_DAD DR	CRYPTO_BA+0x2AC	R/W	TDES/DES 通道2 DMA 目的地址寄存器	0x0000_0000
CRYPTO_TDES3_DAD DR	CRYPTO_BA+0x2EC	R/W	TDES/DES 通道3 DMA 目的地址寄存器	0x0000_0000



位	描述	
[31:0]	DADDR	<p><b>TDES/DES DMA 目的地</b></p> <p>TDES/DES 加速器支持 DMA 功能在 SRAM 内存和内建 FIFO 间传输加密文本。CRYPTO_TDESn_DADDR 存放引擎输出文本存放的数据缓存的目的地址。基于这个目的地址，TDES/DES 加速器可以在 TDES/DES 操作完成之后把加密文本(加密)/原始文本(解密)写入 SRAM 内存。目的地址的开始位置必须是字对齐，就是说 CRYPTO_TDESn_DADDR 的位0和位1会被忽略。</p> <p>CRYPTO_TDESn_DADDR 可读可写。在 TDES/DES 操作时写入 CRYPTO_TDESn_DADDR 不会影响当前的 TDES/DES 操作，但 CRYPTO_TDESn_DADDR 的值会在稍后更新，因此，程序可以为下一笔 TDES/DES 操作准备目的地址。</p> <p>在 DMA 模式，程序可以在触发 START 前更新系一个 CRYPTO_TDESn_DA DDR CRYPTO_TDESn_SADDR 和 CRYPTO_TDESn_DA DDR 的值可以相同。</p>

**CRYPTO\_TDES0\_CNT, CRYPTO\_TDES1\_CNT, CRYPTO\_TDES2\_CNT, CRYPTO\_TDES3\_CNT  
TDES/DES 块计数寄存器**

寄存器	偏移量	R/W	描述	复位值
CRYPTO_TDES0_CNT	CRYPTO_BA+0x230	R/W	TDES/DES 通道0字节计数寄存器	0x0000_0000
CRYPTO_TDES1_CNT	CRYPTO_BA+0x270	R/W	TDES/DES 通道1字节计数寄存器	0x0000_0000
CRYPTO_TDES2_CNT	CRYPTO_BA+0x2B0	R/W	TDES/DES 通道2字节计数寄存器	0x0000_0000
CRYPTO_TDES3_CNT	CRYPTO_BA+0x2F0	R/W	TDES/DES 通道3字节计数寄存器	0x0000_0000



位	描述	
[31:0]	CNT	<p><b>TDES/DES 字节计数</b></p> <p>TDES/DES 工作在 DMA 模式，CRYPTO_TDESn_CNT 存放原始文本的字节数，CRYPTO_TDESn_CNT 是32位，所以最大字节数是 4G 字节。</p> <p>CRYPTO_TDESn_CNT 可读可写。当TDES/DES 正在工作时写CRYPTO_TDESn_CNT 不会影响目前的 TDES/DES 操作。但CRYPTO_TDESn_CNT 的值会在稍后更新，因此程序可以为下次TDES/DES操作准备好字节数。</p> <p>在 非-DMA ECB, CBC, CFB, OFB, 和 CTR 模式, CRYPTO_TDESn_CNT 在存储最后一个数据块之前设置最后一个数据块的字节数。</p>

**CRYPTO\_TDES\_DATIN TDES/DES 数据输入端口寄存器**

寄存器	偏移量	R/W	描述	复位值
CRYPTO_TDES_DATIN	CRYPTO_BA+0x234	R/W	TDES/DES 引擎输入数据字寄存器	0x0000_0000

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

位	描述	
[31:0]	DATIN	TDES/DES 引擎输入端口 CPU 通过这个端口向TDES/DES 引擎填写数据，确认CRYPTO_TDES_STS寄存器，在INBUFFULL 为 0时填写数据.

## CRYPTO\_TDES\_DATOUT TDES/DES 数据输出端口寄存器

寄存器	偏移量	R/W	描述	复位值
CRYPTO_TDES_DATOUT	CRYPTO_BA+0x238	R	TDES/DES引擎输出数据字寄存器	0x0000_0000

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

位	描述	
[31:0]	DATOUT	TDES/DES 引擎输出端口 CPU 通过这个端口从TDES/DES 引擎得到结果，确认CRYPTO_TDES_STS寄存器，在 OUTBUFEMPTY 为 0时读取数据.

## CRYPTO\_TDES\_FDBCKx TDES/DES 反馈 x 寄存器

寄存器	偏移量	R/W	描述	复位值
CRYPTO_TDES_FDBCKH	CRYPTO_BA+0x060	R	TDES/DES 引擎加密操作后输出反馈高位数据	0x0000_0000
CRYPTO_TDES_FDBCKL	CRYPTO_BA+0x064	R	TDES/DES 引擎加密操作后输出反馈低位数据	0x0000_0000

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

位	描述	
[31:0]	FDBCK	<p><b>TDES/DES 反馈</b> 反馈值是 64 位。</p> <p>在 DMA 级联模式下，TDES/DES 引擎用 {CRYPTO_TDES_FDBCKH, CRYPTO_TDES_FDBCKL} 的数据作为下一个数据块的 {CRYPTO_TDESn_IVH, CRYPTO_TDESn_IVL}。反馈寄存器在 CBC, CFB, 和 OFB 模式下使用。</p> <p>TDES/DES 引擎输出反馈信息作为下一个块操作的初始化向量。程序可以使用这个反馈信息实现多于4个DMA通道。程序可以暂存反馈值，当切换回来，将反馈值填入同一个通道的 CRYPTO_TDESn_IVH/L 寄存器，可以依照原来的设定继续执行。</p>

## 6.36.7.5 SHA/HMAC 寄存器

CRYPTO HMAC CTL SHA/HMAC 控制寄存器

寄存器	偏移量	R/W	描述	复位值
CRYPTO_HMAC_CTL	CRYPTO_BA+0x300	R/W	SHA/HMAC 控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
INSWAP	OUTSWAP	Reserved					
15	14	13	12	11	10	9	8
Reserved					OPMODE		
7	6	5	4	3	2	1	0
DMAEN	Reserved	DMALAST	HMACEN	Reserved	STOP	START	

位	描述
[31:24]	Reserved 保留.
[23]	INSWAP <b>SHA/HMAC 引擎输入数据交换</b> 0 = 保持原有顺序. 1 = CPU存入加速器的数据顺序从 {byte3, byte2, byte1, byte0} 变为 {byte0, byte1, byte2, byte3}.
[22]	OUTSWAP <b>SHA/HMAC 引擎输出数据交换</b> 0 = 保持原有顺序. 1 = CPU从加速器读取数据的顺序从 {byte3, byte2, byte1, byte0} 变为 {byte0, byte1, byte2, byte3}.
[21:11]	Reserved 保留.
[10:8]	OPMODE <b>SHA/HMAC 引擎操作模式</b> 0x0xx: SHA160 0x100: SHA256 0x101: SHA224 0x110: SHA512 0x111: SHA384 这些位可读写. 但当BUSY为1时写入没有效果.
[7]	DMAEN <b>SHA/HMAC 引擎 DMA 使能控制</b> 0 = SHA/HMAC DMA 引擎禁止. SHA/HMAC 引擎工作在非-DMA 模式, 从端口 CRYPTO_HMAC_DATIN得到数据. 1 = SHA/HMAC DMA 引擎使能. SHA/HMAC 引擎工作在 DMA 模式, 数据存取由 DMA 逻辑完成.
[6]	Reserved 保留.

[5]	<b>DMALAST</b>	<b>SHA/HMAC 最后块</b> 此位在最后一个数据块时置1
[4]	<b>HMACEN</b>	<b>HMAC_SHA 引擎操作模式</b> 0 = 执行 SHA 功能. 1 = 执行 HMAC 功能.
[3:2]	<b>Reserved</b>	保留.
[1]	<b>STOP</b>	<b>SHA/HMAC 引擎停止</b> 0 = 没影响. 1 = 停止 SHA/HMAC 引擎. 此位总是读为0
[0]	<b>START</b>	<b>SHA/HMAC 引擎开始</b> 0 = 没影响. 1 = 开始 SHA/HMAC 引擎. BUSY标志置1. 此位总是读为0.

## CRYPTO HMAC STS SHA/HMAC 状态寄存器

寄存器	偏移量	R/W	描述	复位值
CRYPTO_HMAC_STS	CRYPTO_BA+0x304	R	SHA/HMAC 状态标志	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved						DMABUSY	BUSY

位	描述
[31:16]	<b>Reserved</b>
[16]	<b>DATINREQ</b> SHA/HMAC 非-DMA模式数据输入请求 0 = 没影响. 1 = 请求SHA/HMAC 非-DMA 模式数据输入.
[15:9]	<b>Reserved</b>
[8]	<b>DMAERR</b> SHA/HMAC引擎 DMA 错误标志 0 = 表明SHA/HMAC 引擎正常访问. 1 = 表明 SHA/HMAC引擎访问错误
[7:2]	<b>Reserved</b>
[1]	<b>DMABUSY</b> SHA/HMAC 引擎 DMA 忙标志 0 = SHA/HMAC DMA 引擎在空闲或完成态. 1 = SHA/HMAC DMA 引擎忙.
[0]	<b>BUSY</b> SHA/HMAC 引擎忙 0 = SHA/HMAC 在空闲或完成态 1 = SHA/HMAC 引擎忙.

## CRYPTO HMAC DGSTx SHA/HMAC输出摘要字寄存器

寄存器	偏移量	R/W	描述	复位值
CRYPTO_HMAC_D_GST0	CRYPTO_BA+0x308	R	SHA/HMAC摘要信息0	0x0000_0000
CRYPTO_HMAC_D_GST1	CRYPTO_BA+0x30C	R	SHA/HMAC摘要信息1	0x0000_0000
CRYPTO_HMAC_D_GST2	CRYPTO_BA+0x310	R	SHA/HMAC摘要信息2	0x0000_0000
CRYPTO_HMAC_D_GST3	CRYPTO_BA+0x314	R	SHA/HMAC摘要信息3	0x0000_0000
CRYPTO_HMAC_D_GST4	CRYPTO_BA+0x318	R	SHA/HMAC摘要信息4	0x0000_0000
CRYPTO_HMAC_D_GST5	CRYPTO_BA+0x31C	R	SHA/HMAC摘要信息5	0x0000_0000
CRYPTO_HMAC_D_GST6	CRYPTO_BA+0x320	R	SHA/HMAC摘要信息6	0x0000_0000
CRYPTO_HMAC_D_GST7	CRYPTO_BA+0x324	R	SHA/HMAC摘要信息7	0x0000_0000
CRYPTO_HMAC_D_GST8	CRYPTO_BA+0x328	R	SHA/HMAC摘要信息8	0x0000_0000
CRYPTO_HMAC_D_GST9	CRYPTO_BA+0x32C	R	SHA/HMAC摘要信息9	0x0000_0000
CRYPTO_HMAC_D_GST10	CRYPTO_BA+0x330	R	SHA/HMAC摘要信息10	0x0000_0000
CRYPTO_HMAC_D_GST11	CRYPTO_BA+0x334	R	SHA/HMAC摘要信息11	0x0000_0000
CRYPTO_HMAC_D_GST12	CRYPTO_BA+0x338	R	SHA/HMAC摘要信息12	0x0000_0000
CRYPTO_HMAC_D_GST13	CRYPTO_BA+0x33C	R	SHA/HMAC摘要信息13	0x0000_0000
CRYPTO_HMAC_D_GST14	CRYPTO_BA+0x340	R	SHA/HMAC摘要信息14	0x0000_0000
CRYPTO_HMAC_D_GST15	CRYPTO_BA+0x344	R	SHA/HMAC摘要信息15	0x0000_0000

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

位	描述
[31:0]	<b>DGST</b> <b>SHA/HMAC 摘要信息输出寄存器</b> 对于 SHA-160, 信息摘要存于 CRYPTO_HMAC_DGST0 ~ CRYPTO_HMAC_DGST4. 对于 SHA-224, 信息摘要存于 CRYPTO_HMAC_DGST0 ~ CRYPTO_HMAC_DGST6. 对于 SHA-256, 信息摘要存于 CRYPTO_HMAC_DGST0 ~ CRYPTO_HMAC_DGST7. 对于 SHA-384, 信息摘要存于 CRYPTO_HMAC_DGST0 ~ CRYPTO_HMAC_DGST11. 对于 SHA-512, 信息摘要存于 CRYPTO_HMAC_DGST0 ~ CRYPTO_HMAC_DGST15.

## CRYPTO HMAC KEYCNT SHA/HMAC 密钥字节计数寄存器

寄存器	偏移量	R/W	描述	复位值
CRYPTO_HMAC_KEYC NT	CRYPTO_BA+0x3 48	R/W	SHA/HMAC密钥字节计数寄存器	0x0000_0000

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

位	描述	
[31:0]	KEYCNT	<p><b>SHA/HMAC 密钥字节计数</b></p> <p>CRYPTO_HMAC_KEYCNT 保存SHA/HMAC 引擎操作的密钥字节个数。这个寄存器是32位，最大的字节数是4G字节. 这里可读可写。</p> <p>在SHA/HMAC操作时写 CRYPTO_HMAC_KEYCNT 不会影响当前的 SHA/HMAC操作，但是 CRYPTO_SHA_KEYCNT的值会在稍后更新，所以程序可以为下次SHA/HMAC操作准备密钥个数</p>

CRYPTO HMAC\_SADDR SHA/HMAC DMA 源地址寄存器

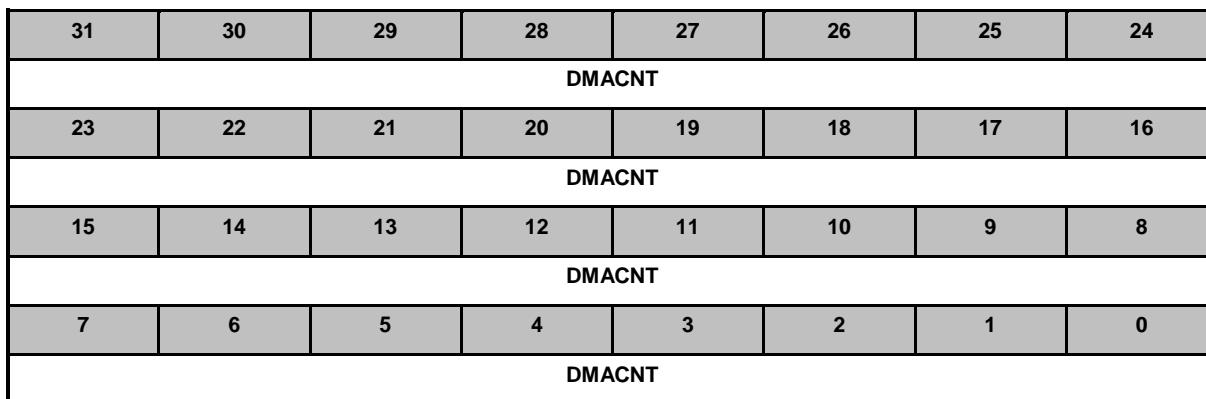
寄存器	偏移量	R/W	描述	复位值
CRYPTO_HMAC_SAD DR	CRYPTO_BA+0x34C	R/W	SHA/HMAC DMA 源地址寄存器	0x0000_0000

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

位	描述
[31:0]	<b>SADDR</b> <b>SHA/HMAC DMA源地址</b> SHA/HMAC 加速器支持 DMA 功能在 SRAM 内存和内建 FIFO 之间传输原始文本 CRYPTO_HMAC_SADDR 存放原始文本存放数据缓存的源地址。基于这个源地址，SHA/HMAC 加速器可以从SRAM中读到原始文本然后执行 SHA/HMAC 操作..。源地址的开始地址必须是字对齐的，就是说CRYPTO_HMAC_SADDR的位1和位0会被忽略 CRYPTO_HMAC_SADDR 可读可写。当 SHA/HMAC 加速器正在工作时写 CRYPTO_HMAC_SADDR 不会影响当前的 SHA/HMAC 操作，但CRYPTO_HMAC_SADDR 的值会在稍后更新。程序可以为下次SHA/HMAC操作准备 DMA 源地址。 在 DMA 模式，程序可以在触发START 更新下一笔CRYPTO_HMAC_SADDR. CRYPTO_HMAC_SADDR 和 CRYPTO_HMAC_DADDR 值可以相同。

CRYPTO HMAC DMACNT SHA/HMAC 字节计数寄存器

寄存器	偏移量	R/W	描述	复位值
CRYPTO_HMAC_DMACNT	CRYPTO_BA+0x350	R/W	SHA/HMAC 字节计数寄存器	0x0000_0000



位	描述
[31:0]	<b>DMACNT</b>  SHA/HMAC 操作字节数 CRYPTO_HMAC_DMACNT 存放 SHA/HMAC 引擎工作在 DMA 模式时原始文本的字节数。 CRYPTO_HMAC_DMACNT 是32位，最大字节数是4G 字节。 CRYPTO_HMAC_DMACNT 可读可写，在 SHA/HMAC 加速器正在操作时写 CRYPTO_HMAC_DMACNT 不会影响目前的 SHA/HMAC 操作。但 CRYPTO_HMAC_DMACNT 的值会在稍后更新，程序可以为下次 SHA/HMAC 操作准备数据字节数。 在非-DMA模式，CRYPTO_HMAC_DMACNT 必须在填写最后一个数据块之前设置

## CRYPTO HMAC DATIN SHA/HMAC 数据输入端口寄存器

寄存器	偏移量	R/W	描述	复位值
CRYPTO_HMAC_DATIN	CRYPTO_BA+0x354	R/W	SHA/HMAC 引擎 非-DMA 模式数据输入端口寄存器	0x0000_0000

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

位	描述	
[31:0]	DATIN	SHA/HMAC 引擎输入端口 CPU 通过这个端口填写数据到SHA/HMAC 引擎，确认CRYPTO_HMAC_STS的状态，在 DATINREQ 为 1时填写数据。

## 6.36.7.6 ECC 寄存器

CRYPTO\_ECC\_CTL ECC 控制寄存器

寄存器	偏移量	R/W	描述	复位值
CRYPTO_ECC_CTL	CRYPTO_BA+0x800	R/W	ECC 控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
CURVEM							
23	22	21	20	19	18	17	16
CURVEM	LDK	LDN	LDB	LDA	LDP2	LDP1	
15	14	13	12	11	10	9	8
Reserved			MODOP		ECCOP		FSEL
7	6	5	4	3	2	1	0
DMAEN	Reserved				STOP	START	

位	描述
[31:22]	<b>CURVEM</b>
[21]	<b>LDK</b> SCALARK 寄存器控制信号 0 = SCALARK 的寄存器不被DMA或用户修改 1 = SCALARK 的寄存器被DMA或用户修改.
[20]	<b>LDN</b> 椭圆曲线参数 <b>CURVEN</b> 寄存器控制信号 0 = CURVEN 的寄存器不被DMA或用户修改. 1 = CURVEN 的寄存器被DMA或用户修改
[19]	<b>LDB</b> 椭圆曲线参数 <b>CURVEB</b> 寄存器控制信号 0 = CURVEB 的寄存器不被DMA或用户修改. 1 = CURVEB 的寄存器被DMA或用户修改
[18]	<b>LDA</b> 椭圆曲线参数 <b>CURVEA</b> 寄存器控制信号 0 = CURVEA 的寄存器不被DMA或用户修改. 1 = CURVEA 的寄存器被DMA或用户修改
[17]	<b>LDP2</b> 第二点( <b>POINTX2</b> , <b>POINTY2</b> )的X, Y坐标的寄存器控制信号 0 = POINTX2 和 POINTY2的寄存器不被DMA或用户修改. 1 = POINTX2 和 POINTY2的寄存器被DMA或用户修改.
[16]	<b>LDP1</b> 第一点( <b>POINTX2</b> , <b>POINTY2</b> )的X, Y坐标的寄存器控制信号 0 = POINTX1 和 POINTY1的寄存器不被DMA或用户修改. 1 = POINTX1 和 POINTY1的寄存器被DMA或用户修改.
[15:13]	<b>Reserved</b> 保留.

[12:11]	<b>MODOP</b>	<b>PF的模操作</b> 00 = 除 :. $\text{POINTX1} = (\text{POINTY1} / \text{POINTX1}) \% \text{CURVEN}.$ 01 = 乘 :. $\text{POINTX1} = (\text{POINTX1} * \text{POINTY1}) \% \text{CURVEN}.$ 10 = 加 :. $\text{POINTX1} = (\text{POINTX1} + \text{POINTY1}) \% \text{CURVEN}.$ 11 = 减 :. $\text{POINTX1} = (\text{POINTX1} - \text{POINTY1}) \% \text{CURVEN}.$ MODOP只当 ECCOP = 01时有效.
[10:9]	<b>ECCOP</b>	<b>BF 和 PF 的点操作</b> 00 = 点乘 :. $(\text{POINTX1}, \text{POINTY1}) = \text{SCALARK} * (\text{POINTX1}, \text{POINTY1}).$ 01 = 模操作 : 由 MODOP (CRYPTO_ECC_CTL[12:11])选择. 10 = 点加 :. $(\text{POINTX1}, \text{POINTY1}) = (\text{POINTX1}, \text{POINTY1}) +.$ $(\text{POINTX2}, \text{POINTY2})$ 11 = 点翻倍 :. $(\text{POINTX1}, \text{POINTY1}) = 2 * (\text{POINTX1}, \text{POINTY1}).$ 除了上面3个输入数据, 点操作还需要椭圆曲线的其他参数(CURVEA, CURVEB, CURVEN 和 CURVEM) 入图 6.27-11.
[8]	<b>FSEL</b>	<b>域选择</b> 0 = 二进制域 ( $GF(2^m)$ ). 1 = 素数域 ( $GF(p)$ ).
[7]	<b>DMAEN</b>	<b>ECC 加速器 DMA 使能控制</b> 0 = ECC DMA 引擎禁止. 1 = ECC DMA 引擎使能 只有当 START 和 DMAEN 为 1, ECC DMA 引擎才会使能
[6:2]	<b>Reserved</b>	保留.
[1]	<b>STOP</b>	<b>ECC 加速器停止</b> 0 = 没影响. 1 = 中断 ECC 加速器并让它进入空闲态 读此位总为0. 在停止ECC 加速器之后记得清 ECC 中断标志.
[0]	<b>START</b>	<b>ECC加速器开始</b> 0 = 没影响. 1 = 开始 ECC 加速器, . BUSY 标志会置1 读此位总为0.. 当BUSY 标志是1.ECC 加速器会忽略这次START 信号 .

## CRYPTO ECC STS ECC 状态寄存器

寄存器	偏移量	R/W	描述	复位值
CRYPTO_ECC_STS	CRYPTO_BA+0x804	R	ECC 状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved						DMABUSY	BUSY

位	描述
[31:17]	<b>Reserved</b>
	保留
[16]	<b>BUSERR</b>
	<b>ECC DMA 访问总线错误标志</b> 0 = 没有错误。 1 = 总线错误会停止 DMA 操作和 ECC 加速器。
[15:2]	<b>Reserved</b>
	保留.
[1]	<b>DMABUSY</b>
	<b>ECC DMA 忙标志</b> 0 = ECC DMA在空闲或完成态 1 = ECC DMA忙
[0]	<b>BUSY</b>
	<b>ECC 加速器忙标志</b> 0 = ECC 加速器在空闲或完成态. 1 = ECC 加速器正在执行且保护所有寄存器. ECC 加速器完成之后记得清 ECC 中断标志

CRYPTO\_ECC\_X1\_ECC第一点 X坐标值寄存器

寄存器	偏移量	R/W	描述	复位值
CRYPTO_ECC_X1_00	CRYPTO_BA+0x808	R/W	ECC第一点 X坐标值字0	0x0000_0000
CRYPTO_ECC_X1_01	CRYPTO_BA+0x80C	R/W	ECC第一点 X坐标值字1	0x0000_0000
CRYPTO_ECC_X1_02	CRYPTO_BA+0x810	R/W	ECC第一点 X坐标值字2	0x0000_0000
CRYPTO_ECC_X1_03	CRYPTO_BA+0x814	R/W	ECC第一点 X坐标值字3	0x0000_0000
CRYPTO_ECC_X1_04	CRYPTO_BA+0x818	R/W	ECC第一点 X坐标值字4	0x0000_0000
CRYPTO_ECC_X1_05	CRYPTO_BA+0x81C	R/W	ECC第一点 X坐标值字5	0x0000_0000
CRYPTO_ECC_X1_06	CRYPTO_BA+0x820	R/W	ECC第一点 X坐标值字6	0x0000_0000
CRYPTO_ECC_X1_07	CRYPTO_BA+0x824	R/W	ECC第一点 X坐标值字7	0x0000_0000
CRYPTO_ECC_X1_08	CRYPTO_BA+0x828	R/W	ECC第一点 X坐标值字8	0x0000_0000
CRYPTO_ECC_X1_09	CRYPTO_BA+0x82C	R/W	ECC第一点 X坐标值字9	0x0000_0000
CRYPTO_ECC_X1_10	CRYPTO_BA+0x830	R/W	ECC第一点 X坐标值字10	0x0000_0000
CRYPTO_ECC_X1_11	CRYPTO_BA+0x834	R/W	ECC第一点 X坐标值字11	0x0000_0000
CRYPTO_ECC_X1_12	CRYPTO_BA+0x838	R/W	ECC第一点 X坐标值字12	0x0000_0000
CRYPTO_ECC_X1_13	CRYPTO_BA+0x83C	R/W	ECC第一点 X坐标值字13	0x0000_0000
CRYPTO_ECC_X1_14	CRYPTO_BA+0x840	R/W	ECC第一点 X坐标值字14	0x0000_0000
CRYPTO_ECC_X1_15	CRYPTO_BA+0x844	R/W	ECC第一点 X坐标值字15	0x0000_0000
CRYPTO_ECC_X1_16	CRYPTO_BA+0x848	R/W	ECC第一点 X坐标值字16	0x0000_0000
CRYPTO_ECC_X1_17	CRYPTO_BA+0x84C	R/W	ECC第一点 X坐标值字17	0x0000_0000

31	30	29	28	27	26	25	24
POINTX1							
23	22	21	20	19	18	17	16
POINTX1							
15	14	13	12	11	10	9	8

POINTX1							
7	6	5	4	3	2	1	0
POINTX1							

位	描述
[31:0]	<b>POINTX1</b>  <b>ECC第一点 X坐标值 (POINTX1)</b> 对于 B-163 或 K-163, POINTX1 存储于 CRYPTO_ECC_X1_00~CRYPTO_ECC_X1_05 对于 B-233 或 K-233, POINTX1 存储于 CRYPTO_ECC_X1_00~CRYPTO_ECC_X1_07 对于 B-283 或 K-283, POINTX1 存储于 CRYPTO_ECC_X1_00~CRYPTO_ECC_X1_08 对于 B-409 或 K-409, POINTX1 存储于 CRYPTO_ECC_X1_00~CRYPTO_ECC_X1_12 对于 B-571 或 K-571, POINTX1 存储于 CRYPTO_ECC_X1_00~CRYPTO_ECC_X1_17 对于 P-192, POINTX1 存储于 CRYPTO_ECC_X1_00~CRYPTO_ECC_X1_05 对于 P-224, POINTX1 存储于 CRYPTO_ECC_X1_00~CRYPTO_ECC_X1_06 对于 P-256, POINTX1 存储于 CRYPTO_ECC_X1_00~CRYPTO_ECC_X1_07 对于 P-384, POINTX1 存储于 CRYPTO_ECC_X1_00~CRYPTO_ECC_X1_11 对于 P-521, POINTX1 存储于 CRYPTO_ECC_X1_00~CRYPTO_ECC_X1_16

## CRYPTO ECC Y1 ECC第一点 Y坐标值寄存器

寄存器	偏移量	R/W	描述	复位值
CRYPTO_ECC_Y1_00	CRYPTO_BA+0x850	R/W	ECC第一点 Y坐标值字0	0x0000_0000
CRYPTO_ECC_Y1_01	CRYPTO_BA+0x854	R/W	ECC第一点 Y坐标值字1	0x0000_0000
CRYPTO_ECC_Y1_02	CRYPTO_BA+0x858	R/W	ECC第一点 Y坐标值字2	0x0000_0000
CRYPTO_ECC_Y1_03	CRYPTO_BA+0x85C	R/W	ECC第一点 Y坐标值字3	0x0000_0000
CRYPTO_ECC_Y1_04	CRYPTO_BA+0x860	R/W	ECC第一点 Y坐标值字4	0x0000_0000
CRYPTO_ECC_Y1_05	CRYPTO_BA+0x864	R/W	ECC第一点 Y坐标值字5	0x0000_0000
CRYPTO_ECC_Y1_06	CRYPTO_BA+0x868	R/W	ECC第一点 Y坐标值字6	0x0000_0000
CRYPTO_ECC_Y1_07	CRYPTO_BA+0x86C	R/W	ECC第一点 Y坐标值字7	0x0000_0000
CRYPTO_ECC_Y1_08	CRYPTO_BA+0x870	R/W	ECC第一点 Y坐标值字8	0x0000_0000
CRYPTO_ECC_Y1_09	CRYPTO_BA+0x874	R/W	ECC第一点 Y坐标值字9	0x0000_0000
CRYPTO_ECC_Y1_10	CRYPTO_BA+0x878	R/W	ECC第一点 Y坐标值字10	0x0000_0000
CRYPTO_ECC_Y1_11	CRYPTO_BA+0x87C	R/W	ECC第一点 Y坐标值字11	0x0000_0000
CRYPTO_ECC_Y1_12	CRYPTO_BA+0x880	R/W	ECC第一点 Y坐标值字12	0x0000_0000
CRYPTO_ECC_Y1_13	CRYPTO_BA+0x884	R/W	ECC第一点 Y坐标值字13	0x0000_0000
CRYPTO_ECC_Y1_14	CRYPTO_BA+0x888	R/W	ECC第一点 Y坐标值字14	0x0000_0000
CRYPTO_ECC_Y1_15	CRYPTO_BA+0x88C	R/W	ECC第一点 Y坐标值字15	0x0000_0000
CRYPTO_ECC_Y1_16	CRYPTO_BA+0x890	R/W	ECC第一点 Y坐标值字16	0x0000_0000
CRYPTO_ECC_Y1_17	CRYPTO_BA+0x894	R/W	ECC第一点 Y坐标值字17	0x0000_0000

31	30	29	28	27	26	25	24
POINTY1							
23	22	21	20	19	18	17	16
POINTY1							
15	14	13	12	11	10	9	8

POINTY1							
7	6	5	4	3	2	1	0
POINTY1							

位	描述
[31:0]	<b>POINTY1</b>  <b>ECC第一点 Y坐标值寄存器_ (POINTY1)</b> 对于 B-163 或 K-163, POINTY1 存储于 CRYPTO_ECC_Y1_00~CRYPTO_ECC_Y1_05 对于B-233 或 K-233, POINTY1 存储于 CRYPTO_ECC_Y1_00~CRYPTO_ECC_Y1_07 对于B-283 或 K-283, POINTY1 存储于 CRYPTO_ECC_Y1_00~CRYPTO_ECC_Y1_08 对于B-409 或 K-409, POINTY1 存储于 CRYPTO_ECC_Y1_00~CRYPTO_ECC_Y1_12 对于B-571 或 K-571, POINTY1 存储于 CRYPTO_ECC_Y1_00~CRYPTO_ECC_Y1_17 对于P-192, POINTY1 存储于 CRYPTO_ECC_Y1_00~CRYPTO_ECC_Y1_05 对于P-224, POINTY1 存储于 CRYPTO_ECC_Y1_00~CRYPTO_ECC_Y1_06 对于P-256, POINTY1 存储于 CRYPTO_ECC_Y1_00~CRYPTO_ECC_Y1_07 对于P-384, POINTY1 存储于 CRYPTO_ECC_Y1_00~CRYPTO_ECC_Y1_11 对于P-521, POINTY1 存储于 CRYPTO_ECC_Y1_00~CRYPTO_ECC_Y1_16

## CRYPTO ECC X2 ECC第二点 X坐标值寄存器

寄存器	偏移量	R/W	描述	复位值
CRYPTO_ECC_X2_00	CRYPTO_BA+0x898	R/W	ECC第二点 X坐标值字0	0x0000_0000
CRYPTO_ECC_X2_01	CRYPTO_BA+0x89C	R/W	ECC第二点 X坐标值字1	0x0000_0000
CRYPTO_ECC_X2_02	CRYPTO_BA+0x8A0	R/W	ECC第二点 X坐标值字2	0x0000_0000
CRYPTO_ECC_X2_03	CRYPTO_BA+0x8A4	R/W	ECC第二点 X坐标值字3	0x0000_0000
CRYPTO_ECC_X2_04	CRYPTO_BA+0x8A8	R/W	ECC第二点 X坐标值字4	0x0000_0000
CRYPTO_ECC_X2_05	CRYPTO_BA+0x8AC	R/W	ECC第二点 X坐标值字5	0x0000_0000
CRYPTO_ECC_X2_06	CRYPTO_BA+0x8B0	R/W	ECC第二点 X坐标值字6	0x0000_0000
CRYPTO_ECC_X2_07	CRYPTO_BA+0x8B4	R/W	ECC第二点 X坐标值字7	0x0000_0000
CRYPTO_ECC_X2_08	CRYPTO_BA+0x8B8	R/W	ECC第二点 X坐标值字8	0x0000_0000
CRYPTO_ECC_X2_09	CRYPTO_BA+0x8BC	R/W	ECC第二点 X坐标值字9	0x0000_0000
CRYPTO_ECC_X2_10	CRYPTO_BA+0x8C0	R/W	ECC第二点 X坐标值字10	0x0000_0000
CRYPTO_ECC_X2_11	CRYPTO_BA+0x8C4	R/W	ECC第二点 X坐标值字11	0x0000_0000
CRYPTO_ECC_X2_12	CRYPTO_BA+0x8C8	R/W	ECC第二点 X坐标值字12	0x0000_0000
CRYPTO_ECC_X2_13	CRYPTO_BA+0x8CC	R/W	ECC第二点 X坐标值字13	0x0000_0000
CRYPTO_ECC_X2_14	CRYPTO_BA+0x8D0	R/W	ECC第二点 X坐标值字14	0x0000_0000
CRYPTO_ECC_X2_15	CRYPTO_BA+0x8D4	R/W	ECC第二点 X坐标值字15	0x0000_0000
CRYPTO_ECC_X2_16	CRYPTO_BA+0x8D8	R/W	ECC第二点 X坐标值字16	0x0000_0000
CRYPTO_ECC_X2_17	CRYPTO_BA+0x8DC	R/W	ECC第二点 X坐标值字17	0x0000_0000

31	30	29	28	27	26	25	24
POINTX2							
23	22	21	20	19	18	17	16
POINTX2							
15	14	13	12	11	10	9	8

POINTX2							
7	6	5	4	3	2	1	0
POINTX2							

位	描述
[31:0]	<b>POINTX2</b>  <b>ECC第二点 X坐标值寄存器 (POINTX2)</b> 对于 B-163 或 K-163, POINTX2 存储于 CRYPTO_ECC_X2_00~CRYPTO_ECC_X2_05 对于B-233 或 K-233, POINTX2 存储于 CRYPTO_ECC_X2_00~CRYPTO_ECC_X2_07 对于B-283 或 K-283, POINTX2 存储于 CRYPTO_ECC_X2_00~CRYPTO_ECC_X2_08 对于B-409 或 K-409, POINTX2 存储于 CRYPTO_ECC_X2_00~CRYPTO_ECC_X2_12 对于B-571 或 K-571, POINTX2 存储于 CRYPTO_ECC_X2_00~CRYPTO_ECC_X2_17 对于P-192, POINTX2 存储于 CRYPTO_ECC_X2_00~CRYPTO_ECC_X2_05 对于P-224, POINTX2 存储于 CRYPTO_ECC_X2_00~CRYPTO_ECC_X2_06 对于P-256, POINTX2 存储于 CRYPTO_ECC_X2_00~CRYPTO_ECC_X2_07 对于P-384, POINTX2 存储于 CRYPTO_ECC_X2_00~CRYPTO_ECC_X2_11 对于P-521, POINTX2 存储于 CRYPTO_ECC_X2_00~CRYPTO_ECC_X2_16

## CRYPTO ECC Y2 ECC第二点 Y坐标值寄存器

寄存器	偏移量	R/W	描述	复位值
CRYPTO_ECC_Y2_00	CRYPTO_BA+0x8E0	R/W	ECC第二点 Y坐标值字0	0x0000_0000
CRYPTO_ECC_Y2_01	CRYPTO_BA+0x8E4	R/W	ECC第二点 Y坐标值字1	0x0000_0000
CRYPTO_ECC_Y2_02	CRYPTO_BA+0x8E8	R/W	ECC第二点 Y坐标值字2	0x0000_0000
CRYPTO_ECC_Y2_03	CRYPTO_BA+0x8EC	R/W	ECC第二点 Y坐标值字3	0x0000_0000
CRYPTO_ECC_Y2_04	CRYPTO_BA+0x8F0	R/W	ECC第二点 Y坐标值字4	0x0000_0000
CRYPTO_ECC_Y2_05	CRYPTO_BA+0x8F4	R/W	ECC第二点 Y坐标值字5	0x0000_0000
CRYPTO_ECC_Y2_06	CRYPTO_BA+0x8F8	R/W	ECC第二点 Y坐标值字6	0x0000_0000
CRYPTO_ECC_Y2_07	CRYPTO_BA+0x8FC	R/W	ECC第二点 Y坐标值字7	0x0000_0000
CRYPTO_ECC_Y2_08	CRYPTO_BA+0x900	R/W	ECC第二点 Y坐标值字8	0x0000_0000
CRYPTO_ECC_Y2_09	CRYPTO_BA+0x904	R/W	ECC第二点 Y坐标值字9	0x0000_0000
CRYPTO_ECC_Y2_10	CRYPTO_BA+0x908	R/W	ECC第二点 Y坐标值字10	0x0000_0000
CRYPTO_ECC_Y2_11	CRYPTO_BA+0x90C	R/W	ECC第二点 Y坐标值字11	0x0000_0000
CRYPTO_ECC_Y2_12	CRYPTO_BA+0x910	R/W	ECC第二点 Y坐标值字12	0x0000_0000
CRYPTO_ECC_Y2_13	CRYPTO_BA+0x914	R/W	ECC第二点 Y坐标值字13	0x0000_0000
CRYPTO_ECC_Y2_14	CRYPTO_BA+0x918	R/W	ECC第二点 Y坐标值字14	0x0000_0000
CRYPTO_ECC_Y2_15	CRYPTO_BA+0x91C	R/W	ECC第二点 Y坐标值字15	0x0000_0000
CRYPTO_ECC_Y2_16	CRYPTO_BA+0x920	R/W	ECC第二点 Y坐标值字16	0x0000_0000
CRYPTO_ECC_Y2_17	CRYPTO_BA+0x924	R/W	ECC第二点 Y坐标值字17	0x0000_0000

31	30	29	28	27	26	25	24
POINTY2							
23	22	21	20	19	18	17	16
POINTY2							
15	14	13	12	11	10	9	8

POINTY2							
7	6	5	4	3	2	1	0
POINTY2							

位	描述
[31:0]	<b>POINTY2</b>  <b>ECC第二点 Y坐标值寄存器(POINTY2)</b> 对于 B-163 或 K-163, POINTY2存储于 CRYPTO_ECC_Y2_00~CRYPTO_ECC_Y2_05 对于B-233 或 K-233, POINTY2存储于 CRYPTO_ECC_Y2_00~CRYPTO_ECC_Y2_07 对于B-283 或 K-283, POINTY2存储于 CRYPTO_ECC_Y2_00~CRYPTO_ECC_Y2_08 对于B-409 或 K-409, POINTY2存储于 CRYPTO_ECC_Y2_00~CRYPTO_ECC_Y2_12 对于B-571 或 K-571, POINTY2存储于 CRYPTO_ECC_Y2_00~CRYPTO_ECC_Y2_17 对于P-192, POINTY2存储于 CRYPTO_ECC_Y2_00~CRYPTO_ECC_Y2_05 对于P-224, POINTY2存储于 CRYPTO_ECC_Y2_00~CRYPTO_ECC_Y2_06 对于P-256, POINTY2存储于 CRYPTO_ECC_Y2_00~CRYPTO_ECC_Y2_07 对于P-384, POINTY2存储于 CRYPTO_ECC_Y2_00~CRYPTO_ECC_Y2_11 对于P-521, POINTY2存储于 CRYPTO_ECC_Y2_00~CRYPTO_ECC_Y2_16

## CRYPTO\_ECC\_A ECC 椭圆曲线参数 CURVEA 值寄存器

寄存器	偏移量	R/W	描述	复位值
CRYPTO_ECC_A_0	CRYPTO_BA+0x928	R/W	ECC 椭圆曲线参数 CURVEA 字0	0x0000_0000
CRYPTO_ECC_A_0	CRYPTO_BA+0x92C	R/W	ECC 椭圆曲线参数 CURVEA 字1	0x0000_0000
CRYPTO_ECC_A_0	CRYPTO_BA+0x930	R/W	ECC 椭圆曲线参数 CURVEA 字2	0x0000_0000
CRYPTO_ECC_A_0	CRYPTO_BA+0x934	R/W	ECC 椭圆曲线参数 CURVEA 字3	0x0000_0000
CRYPTO_ECC_A_0	CRYPTO_BA+0x938	R/W	ECC 椭圆曲线参数 CURVEA 字4	0x0000_0000
CRYPTO_ECC_A_0	CRYPTO_BA+0x93C	R/W	ECC 椭圆曲线参数 CURVEA 字5	0x0000_0000
CRYPTO_ECC_A_0	CRYPTO_BA+0x940	R/W	ECC 椭圆曲线参数 CURVEA 字6	0x0000_0000
CRYPTO_ECC_A_0	CRYPTO_BA+0x944	R/W	ECC 椭圆曲线参数 CURVEA 字7	0x0000_0000
CRYPTO_ECC_A_0	CRYPTO_BA+0x948	R/W	ECC 椭圆曲线参数 CURVEA 字8	0x0000_0000
CRYPTO_ECC_A_0	CRYPTO_BA+0x94C	R/W	ECC 椭圆曲线参数 CURVEA 字9	0x0000_0000
CRYPTO_ECC_A_1	CRYPTO_BA+0x950	R/W	ECC 椭圆曲线参数 CURVEA 字10	0x0000_0000
CRYPTO_ECC_A_1	CRYPTO_BA+0x954	R/W	ECC 椭圆曲线参数 CURVEA 字11	0x0000_0000
CRYPTO_ECC_A_1	CRYPTO_BA+0x958	R/W	ECC 椭圆曲线参数 CURVEA 字12	0x0000_0000
CRYPTO_ECC_A_1	CRYPTO_BA+0x95C	R/W	ECC 椭圆曲线参数 CURVEA 字13	0x0000_0000
CRYPTO_ECC_A_1	CRYPTO_BA+0x960	R/W	ECC 椭圆曲线参数 CURVEA 字14	0x0000_0000
CRYPTO_ECC_A_1	CRYPTO_BA+0x964	R/W	ECC 椭圆曲线参数 CURVEA 字15	0x0000_0000
CRYPTO_ECC_A_1	CRYPTO_BA+0x968	R/W	ECC 椭圆曲线参数 CURVEA 字16	0x0000_0000
CRYPTO_ECC_A_1	CRYPTO_BA+0x96C	R/W	ECC 椭圆曲线参数 CURVEA 字17	0x0000_0000

31	30	29	28	27	26	25	24
CURVEA							
23	22	21	20	19	18	17	16
CURVEA							
15	14	13	12	11	10	9	8

CURVEA							
7	6	5	4	3	2	1	0
CURVEA							

位	描述
[31:0]	<b>CURVEA</b>  <b>ECC 椭圆曲线参数 CURVEA 值 (CURVEA)</b> 椭圆曲线公式是 $y^2=x^3+CURVEA*x+CURVEB$ 在 $GF(p)$ 以及 $y^2+x*y=x^3+CURVEA*x^2+CURVEB$ 在 $GF(2^m)$ . 对于 B-163 或 K-163, CURVEA 存储于 CRYPTO_ECC_A_00~CRYPTO_ECC_A_05 对于 B-233 或 K-233, CURVEA 存储于 CRYPTO_ECC_A_00~CRYPTO_ECC_A_07 对于 B-283 或 K-283, CURVEA 存储于 CRYPTO_ECC_A_00~CRYPTO_ECC_A_08 对于 B-409 或 K-409, CURVEA 存储于 CRYPTO_ECC_A_00~CRYPTO_ECC_A_12 对于 B-571 或 K-571, CURVEA 存储于 CRYPTO_ECC_A_00~CRYPTO_ECC_A_17 对于 P-192, CURVEA 存储于 CRYPTO_ECC_A_00~CRYPTO_ECC_A_05 对于 P-224, CURVEA 存储于 CRYPTO_ECC_A_00~CRYPTO_ECC_A_06 对于 P-256, CURVEA 存储于 CRYPTO_ECC_A_00~CRYPTO_ECC_A_07 对于 P-384, CURVEA 存储于 CRYPTO_ECC_A_00~CRYPTO_ECC_A_11 对于 P-521, CURVEA 存储于 CRYPTO_ECC_A_00~CRYPTO_ECC_A_16

## CRYPTO ECC B ECC 椭圆曲线参数 CURVEB 值寄存器

寄存器	偏移量	R/W	描述	复位值
CRYPTO_ECC_B_0	CRYPTO_BA+0x970	R/W	ECC 椭圆曲线参数 CURVEB 字0	0x0000_0000
CRYPTO_ECC_B_0	CRYPTO_BA+0x974	R/W	ECC 椭圆曲线参数 CURVEB 字1	0x0000_0000
CRYPTO_ECC_B_0	CRYPTO_BA+0x978	R/W	ECC 椭圆曲线参数 CURVEB 字2	0x0000_0000
CRYPTO_ECC_B_0	CRYPTO_BA+0x97C	R/W	ECC 椭圆曲线参数 CURVEB 字3	0x0000_0000
CRYPTO_ECC_B_0	CRYPTO_BA+0x980	R/W	ECC 椭圆曲线参数 CURVEB 字4	0x0000_0000
CRYPTO_ECC_B_0	CRYPTO_BA+0x984	R/W	ECC 椭圆曲线参数 CURVEB 字5	0x0000_0000
CRYPTO_ECC_B_0	CRYPTO_BA+0x988	R/W	ECC 椭圆曲线参数 CURVEB 字6	0x0000_0000
CRYPTO_ECC_B_0	CRYPTO_BA+0x98C	R/W	ECC 椭圆曲线参数 CURVEB 字7	0x0000_0000
CRYPTO_ECC_B_0	CRYPTO_BA+0x990	R/W	ECC 椭圆曲线参数 CURVEB 字8	0x0000_0000
CRYPTO_ECC_B_0	CRYPTO_BA+0x994	R/W	ECC 椭圆曲线参数 CURVEB 字9	0x0000_0000
CRYPTO_ECC_B_1	CRYPTO_BA+0x998	R/W	ECC 椭圆曲线参数 CURVEB 字10	0x0000_0000
CRYPTO_ECC_B_1	CRYPTO_BA+0x99C	R/W	ECC 椭圆曲线参数 CURVEB 字11	0x0000_0000
CRYPTO_ECC_B_1	CRYPTO_BA+0x9A0	R/W	ECC 椭圆曲线参数 CURVEB 字12	0x0000_0000
CRYPTO_ECC_B_1	CRYPTO_BA+0x9A4	R/W	ECC 椭圆曲线参数 CURVEB 字13	0x0000_0000
CRYPTO_ECC_B_1	CRYPTO_BA+0x9A8	R/W	ECC 椭圆曲线参数 CURVEB 字14	0x0000_0000
CRYPTO_ECC_B_1	CRYPTO_BA+0x9AC	R/W	ECC 椭圆曲线参数 CURVEB 字15	0x0000_0000
CRYPTO_ECC_B_1	CRYPTO_BA+0x9B0	R/W	ECC 椭圆曲线参数 CURVEB 字16	0x0000_0000
CRYPTO_ECC_B_1	CRYPTO_BA+0x9B4	R/W	ECC 椭圆曲线参数 CURVEB 字17	0x0000_0000

31	30	29	28	27	26	25	24
CURVEB							
23	22	21	20	19	18	17	16
CURVEB							
15	14	13	12	11	10	9	8

CURVEB							
7	6	5	4	3	2	1	0
CURVEB							

位	描述
[31:0]	<b>CURVEB</b>  <b>ECC 椭圆曲线参数 CURVEB 值(CURVEB)</b> 椭圆曲线公式是 $y^2=x^3+CURVEA*x+CURVEB$ 在 $GF(p)$ 和 $y^2+x*y=x^3+CURVEA*x^2+CURVEB$ 在 $GF(2^m)$ . 对于 B-163 或 K-163, CURVEB存储于 CRYPTO_ECC_B_00~CRYPTO_ECC_B_05 对于 B-233 或 K-233, CURVEB存储于 CRYPTO_ECC_B_00~CRYPTO_ECC_B_07 对于 B-283 或 K-283, CURVEB存储于 CRYPTO_ECC_B_00~CRYPTO_ECC_B_08 对于 B-409 或 K-409, CURVEB存储于 CRYPTO_ECC_B_00~CRYPTO_ECC_B_12 对于 B-521 或 K-521, CURVEB存储于 CRYPTO_ECC_B_00~CRYPTO_ECC_B_17 对于 P-192, CURVEB存储于CRYPTO_ECC_B_00~CRYPTO_ECC_B_05 对于 P-224, CURVEB存储于CRYPTO_ECC_B_00~CRYPTO_ECC_B_06 对于 P-256, CURVEB存储于CRYPTO_ECC_B_00~CRYPTO_ECC_B_07 对于 P-384, CURVEB存储于CRYPTO_ECC_B_00~CRYPTO_ECC_B_11 对于 P-521, CURVEB存储于CRYPTO_ECC_B_00~CRYPTO_ECC_B_16

## CRYPTO\_ECC\_N\_ECC 椭圆曲线参数 CURVEN 值寄存器

寄存器	偏移量	R/W	描述	复位值
CRYPTO_ECC_N_0	CRYPTO_BA+0x9B8	R/W	ECC 椭圆曲线参数 CURVEN 字0	0x0000_0000
CRYPTO_ECC_N_0	CRYPTO_BA+0x9BC	R/W	ECC 椭圆曲线参数 CURVEN 字1	0x0000_0000
CRYPTO_ECC_N_0	CRYPTO_BA+0x9C0	R/W	ECC 椭圆曲线参数 CURVEN 字2	0x0000_0000
CRYPTO_ECC_N_0	CRYPTO_BA+0x9C4	R/W	ECC 椭圆曲线参数 CURVEN 字3	0x0000_0000
CRYPTO_ECC_N_0	CRYPTO_BA+0x9C8	R/W	ECC 椭圆曲线参数 CURVEN 字4	0x0000_0000
CRYPTO_ECC_N_0	CRYPTO_BA+0x9CC	R/W	ECC 椭圆曲线参数 CURVEN 字5	0x0000_0000
CRYPTO_ECC_N_0	CRYPTO_BA+0x9D0	R/W	ECC 椭圆曲线参数 CURVEN 字6	0x0000_0000
CRYPTO_ECC_N_0	CRYPTO_BA+0x9D4	R/W	ECC 椭圆曲线参数 CURVEN 字7	0x0000_0000
CRYPTO_ECC_N_0	CRYPTO_BA+0x9D8	R/W	ECC 椭圆曲线参数 CURVEN 字8	0x0000_0000
CRYPTO_ECC_N_0	CRYPTO_BA+0x9DC	R/W	ECC 椭圆曲线参数 CURVEN 字9	0x0000_0000
CRYPTO_ECC_N_1	CRYPTO_BA+0x9E0	R/W	ECC 椭圆曲线参数 CURVEN 字10	0x0000_0000
CRYPTO_ECC_N_1	CRYPTO_BA+0x9E4	R/W	ECC 椭圆曲线参数 CURVEN 字11	0x0000_0000
CRYPTO_ECC_N_1	CRYPTO_BA+0x9E8	R/W	ECC 椭圆曲线参数 CURVEN 字12	0x0000_0000
CRYPTO_ECC_N_1	CRYPTO_BA+0x9EC	R/W	ECC 椭圆曲线参数 CURVEN 字13	0x0000_0000
CRYPTO_ECC_N_1	CRYPTO_BA+0x9F0	R/W	ECC 椭圆曲线参数 CURVEN 字14	0x0000_0000
CRYPTO_ECC_N_1	CRYPTO_BA+0x9F4	R/W	ECC 椭圆曲线参数 CURVEN 字15	0x0000_0000
CRYPTO_ECC_N_1	CRYPTO_BA+0x9F8	R/W	ECC 椭圆曲线参数 CURVEN 字16	0x0000_0000
CRYPTO_ECC_N_1	CRYPTO_BA+0x9FC	R/W	ECC 椭圆曲线参数 CURVEN 字17	0x0000_0000

31	30	29	28	27	26	25	24
CURVEN							
23	22	21	20	19	18	17	16
CURVEN							
15	14	13	12	11	10	9	8

CURVEN							
7	6	5	4	3	2	1	0
CURVEN							

位	描述
[31:0]	<b>CURVEN</b>  <b>ECC 椭圆曲线参数 CURVEN 值(CURVEN)</b> 在GF( $p$ ), CURVEN 是素数多项式. 在GF( $2^m$ ), CURVEN 是不可约多项式. 对于 B-163 或 K-163, CURVEN 存储于 CRYPTO_ECC_N_00~CRYPTO_ECC_N_05 对于 B-233 或 K-233, CURVEN 存储于 CRYPTO_ECC_N_00~CRYPTO_ECC_N_07 对于 B-283 或 K-283, CURVEN 存储于 CRYPTO_ECC_N_00~CRYPTO_ECC_N_08 对于 B-409 或 K-409, CURVEN 存储于 CRYPTO_ECC_N_00~CRYPTO_ECC_N_12 对于 B-571 或 K-571, CURVEN 存储于 CRYPTO_ECC_N_00~CRYPTO_ECC_N_17 对于 P-192, CURVEN 存储于 CRYPTO_ECC_N_00~CRYPTO_ECC_N_05 对于 P-224, CURVEN 存储于 CRYPTO_ECC_N_00~CRYPTO_ECC_N_06 对于 P-256, CURVEN 存储于 CRYPTO_ECC_N_00~CRYPTO_ECC_N_07 对于 P-384, CURVEN 存储于 CRYPTO_ECC_N_00~CRYPTO_ECC_N_11 对于 P-521, CURVEN 存储于 CRYPTO_ECC_N_00~CRYPTO_ECC_N_16

## CRYPTO\_ECC\_K ECC 椭圆曲线标量 K 值寄存器

寄存器	偏移量	R/W	描述	复位值
CRYPTO_ECC_K_0	CRYPTO_BA+0xA00	W	ECC 点乘标量 SCALARK 字0	0x0000_0000
CRYPTO_ECC_K_0	CRYPTO_BA+0xA04	W	ECC 点乘标量 SCALARK 字1	0x0000_0000
CRYPTO_ECC_K_0	CRYPTO_BA+0xA08	W	ECC 点乘标量 SCALARK 字2	0x0000_0000
CRYPTO_ECC_K_0	CRYPTO_BA+0xA0C	W	ECC 点乘标量 SCALARK 字3	0x0000_0000
CRYPTO_ECC_K_0	CRYPTO_BA+0xA10	W	ECC 点乘标量 SCALARK 字4	0x0000_0000
CRYPTO_ECC_K_0	CRYPTO_BA+0xA14	W	ECC 点乘标量 SCALARK 字5	0x0000_0000
CRYPTO_ECC_K_0	CRYPTO_BA+0xA18	W	ECC 点乘标量 SCALARK 字6	0x0000_0000
CRYPTO_ECC_K_0	CRYPTO_BA+0xA1C	W	ECC 点乘标量 SCALARK 字7	0x0000_0000
CRYPTO_ECC_K_0	CRYPTO_BA+0xA20	W	ECC 点乘标量 SCALARK 字8	0x0000_0000
CRYPTO_ECC_K_0	CRYPTO_BA+0xA24	W	ECC 点乘标量 SCALARK 字9	0x0000_0000
CRYPTO_ECC_K_1	CRYPTO_BA+0xA28	W	ECC 点乘标量 SCALARK 字10	0x0000_0000
CRYPTO_ECC_K_1	CRYPTO_BA+0xA2C	W	ECC 点乘标量 SCALARK 字11	0x0000_0000
CRYPTO_ECC_K_1	CRYPTO_BA+0xA30	W	ECC 点乘标量 SCALARK 字12	0x0000_0000
CRYPTO_ECC_K_1	CRYPTO_BA+0xA34	W	ECC 点乘标量 SCALARK 字13	0x0000_0000
CRYPTO_ECC_K_1	CRYPTO_BA+0xA38	W	ECC 点乘标量 SCALARK 字14	0x0000_0000
CRYPTO_ECC_K_1	CRYPTO_BA+0xA3C	W	ECC 点乘标量 SCALARK 字15	0x0000_0000
CRYPTO_ECC_K_1	CRYPTO_BA+0xA40	W	ECC 点乘标量 SCALARK 字16	0x0000_0000
CRYPTO_ECC_K_1	CRYPTO_BA+0xA44	W	ECC 点乘标量 SCALARK 字17	0x0000_0000

31	30	29	28	27	26	25	24
SCALARK							
23	22	21	20	19	18	17	16
SCALARK							
15	14	13	12	11	10	9	8

SCALARK							
7	6	5	4	3	2	1	0
SCALARK							

位	描述
[31:0]	<b>SCALARK</b>  <b>ECC 椭圆曲线标量SCALARK 值(SCALARK)</b> 因为 SCALARK 一般存储私有密钥, ECC 加速器不允许读SCALARK寄存器. 对于 B-163 或 K-163, SCALARK存储于 CRYPTO_ECC_K_00~CRYPTO_ECC_K_05 对于 B-233 或 K-233, SCALARK存储于 CRYPTO_ECC_K_00~CRYPTO_ECC_K_07 对于 B-283 或 K-283, SCALARK存储于 CRYPTO_ECC_K_00~CRYPTO_ECC_K_08 对于 B-409 或 K-409, SCALARK存储于 CRYPTO_ECC_K_00~CRYPTO_ECC_K_12 对于 B-571 或 K-571, SCALARK存储于 CRYPTO_ECC_K_00~CRYPTO_ECC_K_17 对于 P-192, SCALARK存储于CRYPTO_ECC_K_00~CRYPTO_ECC_K_05 对于 P-224, SCALARK存储于CRYPTO_ECC_K_00~CRYPTO_ECC_K_06 对于 P-256, SCALARK存储于CRYPTO_ECC_K_00~CRYPTO_ECC_K_07 对于 P-384, SCALARK存储于CRYPTO_ECC_K_00~CRYPTO_ECC_K_11 对于 P-521, SCALARK存储于CRYPTO_ECC_K_00~CRYPTO_ECC_K_16

## CRYPTO ECC SADDR ECC DMA 源地址寄存器

寄存器	偏移量	R/W	描述	复位值
CRYPTO_ECC_SADDR DR	CRYPTO_BA+0xA48	R/W	ECC DMA 源地址寄存器	0x0000_0000

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

位	描述
[31:0]	<b>Reserved</b> <b>ECC DMA 源地址</b> ECC 加速器支持DMA功能在SRAM内存和ECC加速器之间传输数据和参数。SADDR 存放原始文本存放的数据缓存的源地址，基于源地址，ECC 加速器可以从SRAM内存读到数据和参数执行ECC 操作。起始地址的开始必须是字对齐的，SADDR 的位1和位0会被忽略。SADDR 可读可写。在DMA 模式，程序需要在触发START前更新CRYPTO_ECC_SADDR。

## CRYPTO ECC DADDR ECC DMA 目的地址寄存器

寄存器	偏移量	R/W	描述	复位值
CRYPTO_ECC_DAD DR	CRYPTO_BA+0xA 4C	R/W	ECC DMA 目的地址寄存器	0x0000_0000

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

位	描述
[31:0]	<b>DADDR</b> <b>ECC DMA 目的地址</b> ECC 加速器支持 DMA 功能在SRAM内存和ECC 加速器之间传输数据和参数。DADDR 存放ECC 引擎输出数据存放的数据缓存的地址，基于目的地址， ECC 加速器可以在ECC操作完成之后把结果数据写回SRAM内存。目的地址的其实位置必须是字对齐的，DADDR的位1和位0会被忽略。 DADDR 可读可写. 在 DMA 模式, 程序需要在触发START 前更新 CRYPTO_ECC_DADDR

CRYPTO ECC STARTREG ECC 更新寄存器起始地址

寄存器	偏移量	R/W	描述	复位值
CRYPTO_ECC_STA RTREG	CRYPTO_BA+0xA 50	R/W	ECC 更新寄存器起始地址	0x0000_0000

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

位	描述
[31:0]	<b>STARTREG</b> ECC 更新寄存器起始地址 DMA从更新寄存器的地址向ECC 引擎填写第一笔数据或参数. 当 ECC 引擎正在工作, ECC 加速器不允许用户修改STARTREG. 例如, 我们想要更新 CRYPTO_ECC POINTX1的输入数据, STARTREG 的值是 0x808.

## CRYPTO ECC WORDCNT ECC DMA 字计数

寄存器	偏移量	R/W	描述	复位值
CRYPTO_ECC_WO_RDCNT	CRYPTO_BA+0xA 54	R/W	ECC DMA 字计数	0x0000_0000

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

位	描述
[31:0]	<b>WORDCNT</b> <b>ECC DMA 字计数</b> CRYPTO_ECC_WORDCNT 存放ECC 加速器在各种DMA 模式下的操作需要的输入数据的源地址。尽管 CRYPTO_ECC_WORDCNT 是 32-位, ECC 加速器的最大字数是144个字. CRYPTO_ECC_WORDCNT 可读可写.

## 6.37 EADC 增强型 12位模数转换器

### 6.37.1 概述

M480 含一个12位逐次比较式的 ADC，有16个外部输入和3个内部输入。转换可由软件触发，EPWM0/1 触发，BPWM0/1 触发，timer0~3 触发，ADINT0, ADINT1 中断 EOC (转换结束) 触发 和 外部管脚(EADC0\_ST) 触发。

### 6.37.2 特性

- 输入电压范围: 0~  $V_{REF}$  (最大到 3.6V)
- 参考电压:  $V_{REF}$  或  $AV_{DD}$
- 12位分辨率, 10位精度保证
- 16 路单端输入, 或8 对差分输入
- 3 路内部输入分别是内部带隙电压 ( $V_{BG}$ ), 温度传感器 ( $V_{TEMP}$ ), 和电源电压( $V_{DD}$ )
- 4 个 ADC 中断 (ADINT0~3) , 都有独立的中断向量地址
- 时钟频率最大 72 MHz
- 转换率高达 5.14 MSPS
- 采样时间可配置.
- 可配置12-位, 10-位, 8-位, 6-位模式.
- 支持校准和载入较准字
- 内部参考电压 VREF: 1.6V, 2.0V, 2.5V, 和 3.0V.
- 支持3种省电模式:
  - 深度掉电模式
  - 掉电模式
  - 待机模式
- 多达 19 个采样模块
  - 每个采样模块可配置ADC 通道 EADC\_CH0~15 , 触发源和采样时间
  - ADC16, 17, 18通道固定为: 带隙电压, 温度传感器和电源电压( $V_{DD}$ )
  - 采样模块0~3双数据缓存
  - 转换结果保存在 19 个数据寄存器, 有效位和覆盖标志
- 一次 ADC 可以由下列条件触发:
  - 写 1 到 SWTRGn (EADC\_SWTRG[n] , n = 0~18)
  - 外部管脚 EADC0\_ST
  - Timer0~3 溢出脉冲
  - ADINT0 和 ADINT1 的EOC (转换结束) 中断脉冲
  - EPWM / BPWM

- 支持 PDMA 传输
- 有转换结果监控比较模式

### 6.37.3 框图

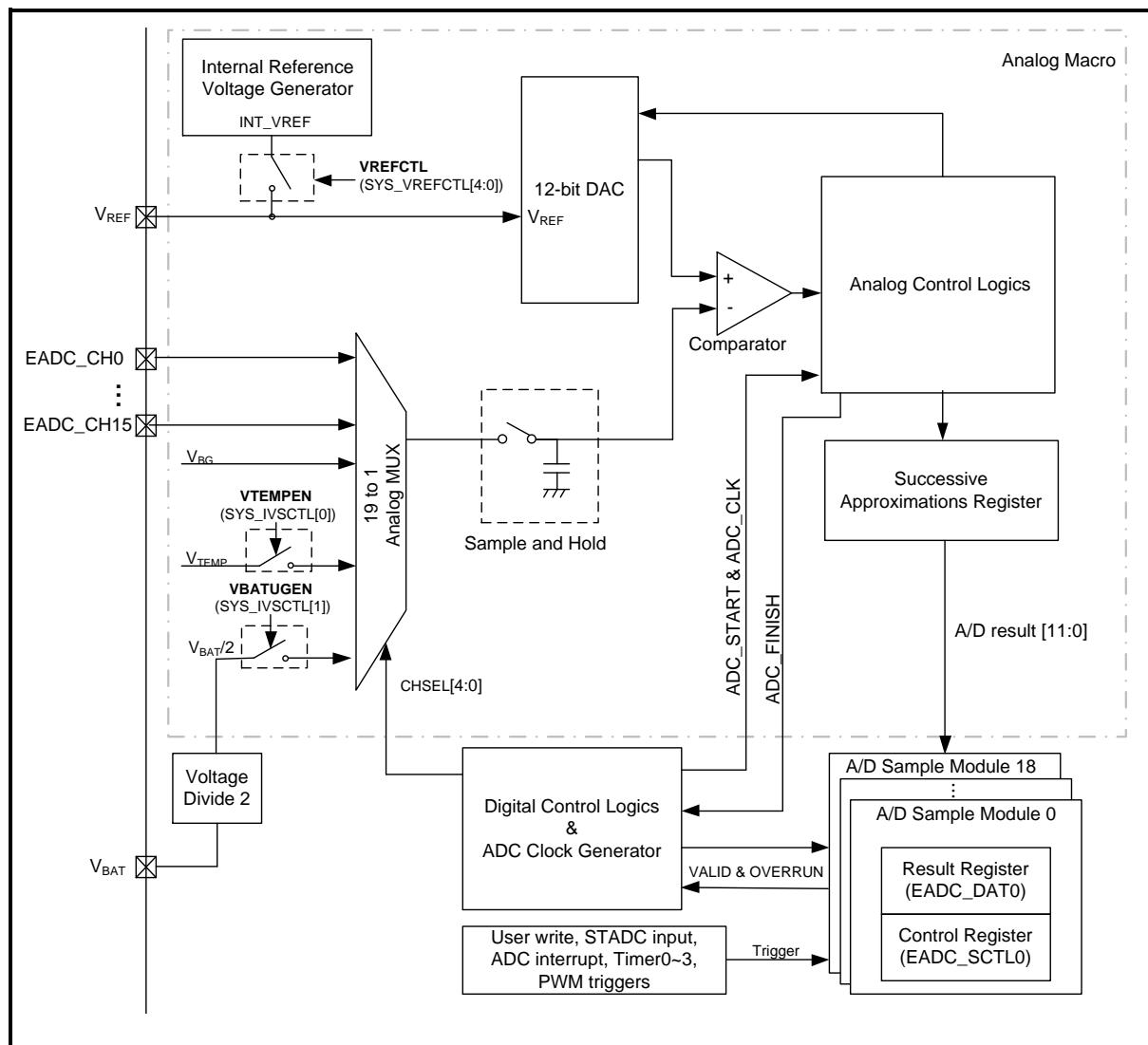


图 6.37-1 ADC 框图

### 6.37.4 基本配置

- 时钟源配置
  - 在 EADCDIV (CKL\_CLKDIV0[23:16]) 选择时钟除频
  - 在 EADCCKEN (CLK\_APBCLK0[28]) 使能 EADC 外设时钟.
- 复位配置
  - 复位 EADC 控制器在 ADCRST (EADC\_CTL [1]).

## ● 管脚配置

组	管脚名	GPIO	MFP
EADC0	EADC0_CH0	PB.0	MFP1
	EADC0_CH1	PB.1	MFP1
	EADC0_CH2	PB.2	MFP1
	EADC0_CH3	PB.3	MFP1
	EADC0_CH4	PB.4	MFP1
	EADC0_CH5	PB.5	MFP1
	EADC0_CH6	PB.6	MFP1
	EADC0_CH7	PB.7	MFP1
	EADC0_CH8	PB.8	MFP1
	EADC0_CH9	PB.9	MFP1
	EADC0_CH10	PB.10	MFP1
	EADC0_CH11	PB.11	MFP1
	EADC0_CH12	PB.12	MFP1
	EADC0_CH13	PB.13	MFP1
	EADC0_CH14	PB.14	MFP1
	EADC0_CH15	PB.15	MFP1
EADC0_ST	EADC0_ST	PF.5	MFP11
		PC.13, PD.12	MFP14
		PG.15	MFP15

### 6.37.5 功能描述

EADC 包含19个模拟开关，19个采样模块以及一个12位转换器。模块0~18, 每一个都可以配置触发源，转换通道。模块 0~15 可以配置到通道 EADC\_CH0~15, 以及不同的触发源。模块 0~3 和模块 4~15 如下图所示。

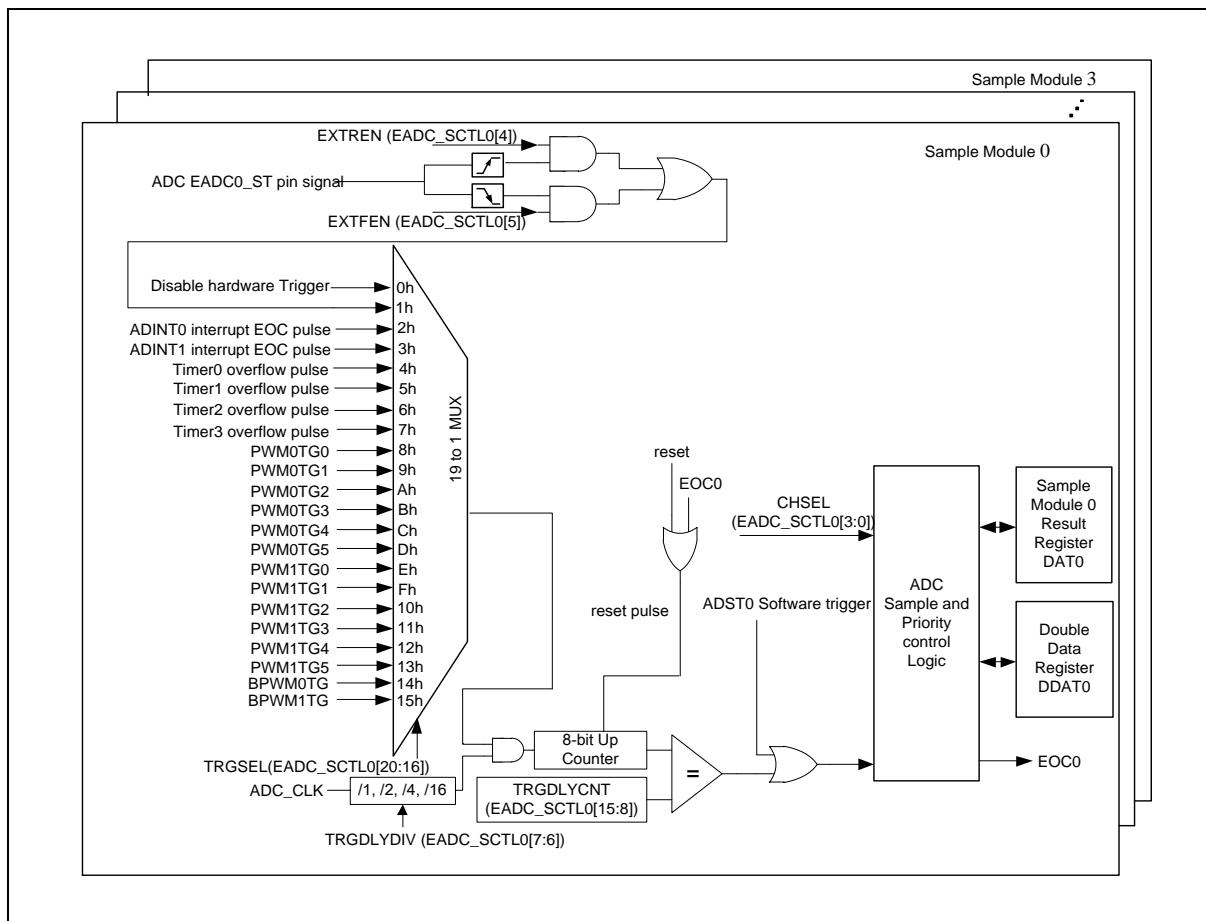


图 6.37-2 模块 0~3 框图

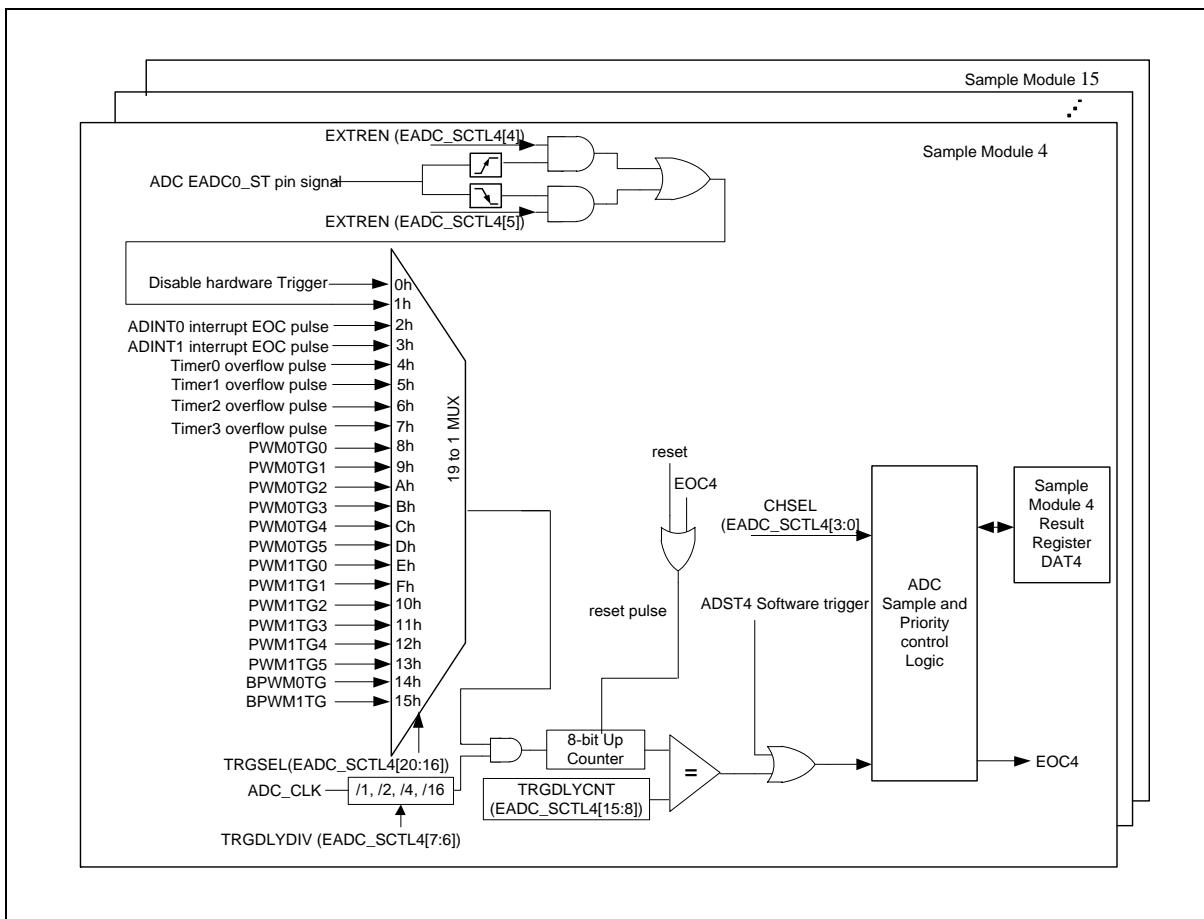


图 6.37-3 模块 4~15 框图

模块 16~18 转换内部通道 ( $V_{BG}$ ,  $V_{TEMP}$ ,  $V_{DD}$ )，可以通过写 SWTRGn (EADC\_SWTRG[n], n = 16~18) 触发。图 6.37-4 展示模块 16~18。

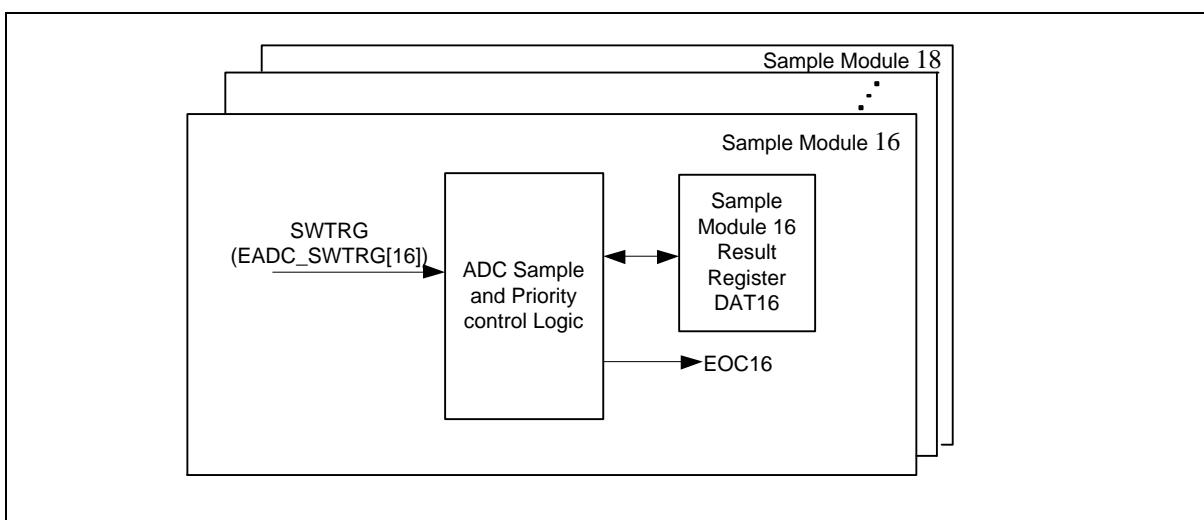


图 6.37-4 模块 16~18 框图

- 模块0~15的ADC 触发源如下:

- 写 1 到 SWTRGn (EADC\_SWTRG[n], n = 0~15)
- 外部管脚 EADC0\_ST
- Timer0~3 溢出脉冲
- ADINT0, ADINT1 ADC 的 EOC (转换结束) 中断
- EPWM/BPWM

当 ADINT0 或 ADINT1 指定的通道转换完成时，产生中断脉冲，可以触发另一个 ADC 转换，在连续转换模式中可以被用到。

#### 6.37.5.1 ADC时钟产生

EADC时钟频率最大可达 72 MHz，采样率最高可达 5.14 MSPS.

EADC 时钟控制如 图 6.37-5. EADC 时钟源来自 HCLK, ADC 时钟被一个8位计数器分频:

$$\text{EADC 时钟频率} = (\text{PCLK1}) / (\text{EADCDIV} (\text{CLK_CLKDIV0}[23:16]) + 1)$$

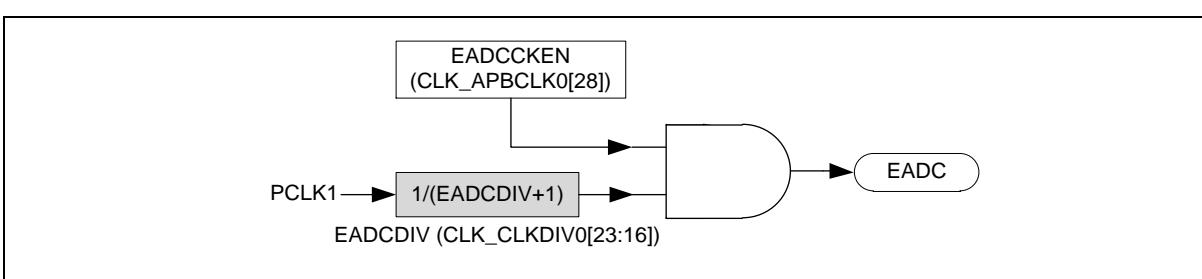


图 6.37-5 EADC 时钟控制

#### 6.37.5.2 ADC软件触发模式

在指定的通道上开始单端ADC 转换,操作如下:

1. ADC 转换由SWTRGn (EADC\_SWTRG[n], n=0~18) 置 1 触发或其他输入触发.
2. SWTRGn (n=0~18) 位在ADC 转换时保持为1，ADC 转换结束, SWTRGn (n=0~18) 位自动清 0，接着去执行其他通道的转换.
3. 当 ADC 转换已完成, 12位结果存于模块的数据寄存器 EADC\_DATn (n=0~18)
4. 转换完成后, ADIFn (EADC\_STATUS2[3:0], n=0~3) 置 1，若ADCIEFn (EADC\_CTL[5:2], n=0~3)为1， ADC 中断 (ADINTn, n=0~3) 产生

转换周期的时序图如 图 6.37-6.

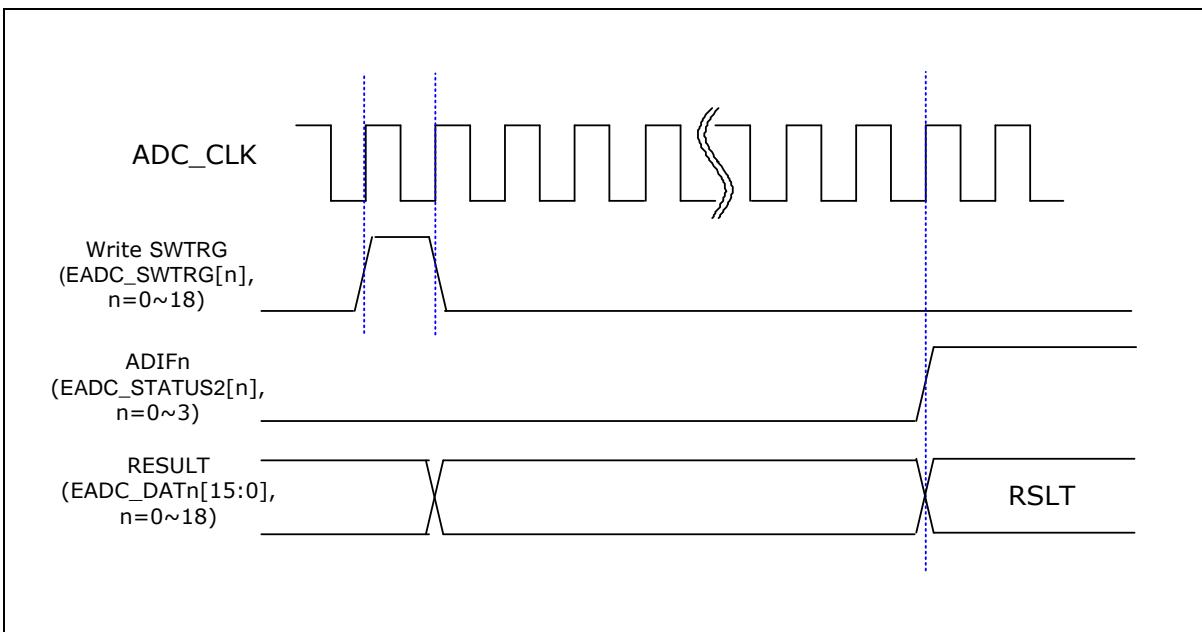


图 6.37-6 ADC 转换时序图, n=0~18

如果多于1个模块使能转换，高优先级的模块先转换，低优先级的等待。模块编号较低的有较高的优先级。模块0有最高的优先级，采样模块18优先级最低。

**注:**如果下次转换间隔超过100us，ADC会自动地进入空闲状态。使用前用户需要执行一次空转换，也就是当ADC处在空闲状态时第一次转换的结果是不准确的。

### 6.37.5.3 ADC 转换优先级

模块优先级见图 6.37-7。当多个模块同时启动转换，编号较低的先转换，其他模块进入等待队列，等待标志 STPF(EADC\_PENDSTS[n], n=0~18) 由硬件置1。当模块完成转换，STPF(EADC\_PENDSTS[n], n=0~18) 自动清0。如果队列里的采样模块被触发多次，覆盖标志 SPOVF(EADC\_OVSTS[n], n=0~18)由硬件置1。

例如，采样模块 0, 2, 3, 5 同时触发。采样模块 0 的通道会先转换。模块 2, 3, 5 会挂起，且 STPF(EADC\_PENDSTS[2], EADC\_PENDSTS [3], EADC\_PENDSTS [5]) 置 1。如果模块 5 同时触发2次，SPOVF(EADC\_OVSTS[5]) 置 1。

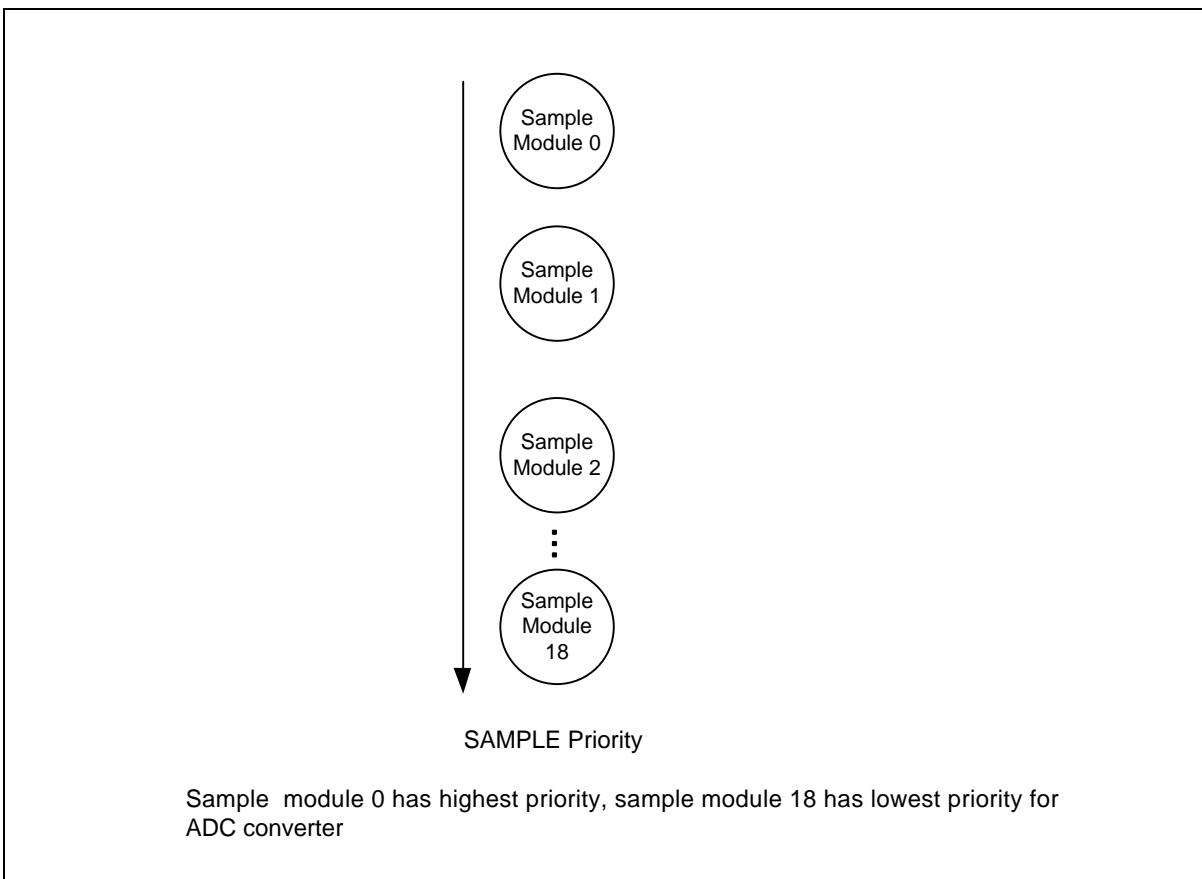


图 6.37-7 模块转换优先级仲裁框图

#### 6.37.5.4 转换周期和采样率

RESSEL (EADC\_CTL[7:6]) 可配置 ADC 结果的位数，对应不同的转换周期。如表 6.37-1。

解析度	最小转换周期
6位	8 ADC_CLK
8 位	10 ADC_CLK
10 位	12 ADC_CLK
12位	14 ADC_CLK

表 6.37-1 解析度与转换周期的关系

有高速通道和低速通道。EADC\_CH10~15 是高速通道，EADC\_CH0~9 是低速通道。高速通道的最大采样率是 5.14 MSPS，低速通道是 2.14 MSPS. 超过这个转换率会得到错误的结果。采样率由下面的公式计算：

$$\text{采样率} = (\text{EADC 时钟频率}) / (\text{转换周期})$$

#### 6.37.5.5 软件触发最大采样率

如果用户需要以最大采样率检测快速通道，转换需要在如下条件下：多采样模块，由软件触发，在上次转换后连续触发。连续转换的例子如下：

1. 使用模块 0~15 来执行连续的转换. 设置 CHSEL (EADC\_SCTL0~15[3:0]) 选择通道 (EADC\_CH10~ EADC\_CH15). 设置 EXTSMPT (EADC\_SCTL0~15[31:24]) 和 TRGDLYCNT (EADC\_SCTL0~15[15:8]) 为 0x00 选择最小采样时间.
2. 设置 SWTRG (EADC\_SWTRG[18:0]) 为 0xffff 来触发模块 0~15.
3. 等待 CURSPL (EADC\_STATUS3[4:0]) 变为 0xf 意味着模块 0~14 已经执行，而模块 15 正在执行. 设置 SWTRG (EADC\_SWTRG[18:0]) 为 0x7fff 来触发模块 0~14 下一轮转换.
4. 等待 CURSPL (EADC\_STATUS3[4:0]) 变为 0x1, 设置 SWTRG (EADC\_SWTRG[18:0]) 为 0x8000 触发模块 15.
5. 重复步骤 3~4 来连续转换.

#### 6.37.5.6 ADC 采样模块转换完成中断操作

ADC 有 4 个中断 ADINT0~3, 每个中断有自己的中断向量, 可以设置多个模块 EOC 脉冲 (转换完成脉冲) 作为中断源。图 6.37-8 展示中断控制逻辑。选 ADINT0 为例, 当 ADCIEN0 (EADC\_CTL[2]) = 1 且 SPLIE<sub>n</sub> (EADC\_INTSRC0[n]) = 1 ( $n=0\sim18$ ), 选择模块的 EOC (转换完成) 脉冲会设置标志 ADIF0 (EADC\_STATUS[0]) 为 1 然后中断 (ADINT0) 会产生。

中断脉冲 (ADINT0/1) 在指定模块 EOC 产生时产生, 可以作为采样模块转换的触发源。用户使用它实现 ADC 连续转换。

中断触发连续转换的例子如下：

1. 如果 ADC 采样模块 2 的 EOC2 作为 ADINT0 中断触发源, SPLIE2 (EADC\_INTSRC0[2]) = 1 且 ADINT0 作为模块 0, 1, 2 转换触发源。
2. 设置软件触发 SWTRG2 (EADC\_SWTRG[2]) 为 1 开始模块 2 转换, 转换完成后, 产生转换结束信号和 ADINT0 中断脉冲, ADINT0 中断脉冲会触发采样模块 0, 1, 2 开始 ADC 转换。
3. ADINT0 中断脉冲重复触发采样模块 0, 1, 2 连续 ADC 转换。
4. 当需要停在连续转换时, 清除 TRGSEL (EADC\_SCTL2[20:16]) 为 0, 禁止模块 2 的 ADINT0 中断脉冲触发转换

**注意：** 因为系统需要在中断脉冲后消耗 3 ADC\_CLK 触发下一个模块, 中断触发连续采样周期是 17 ADC\_CLK。

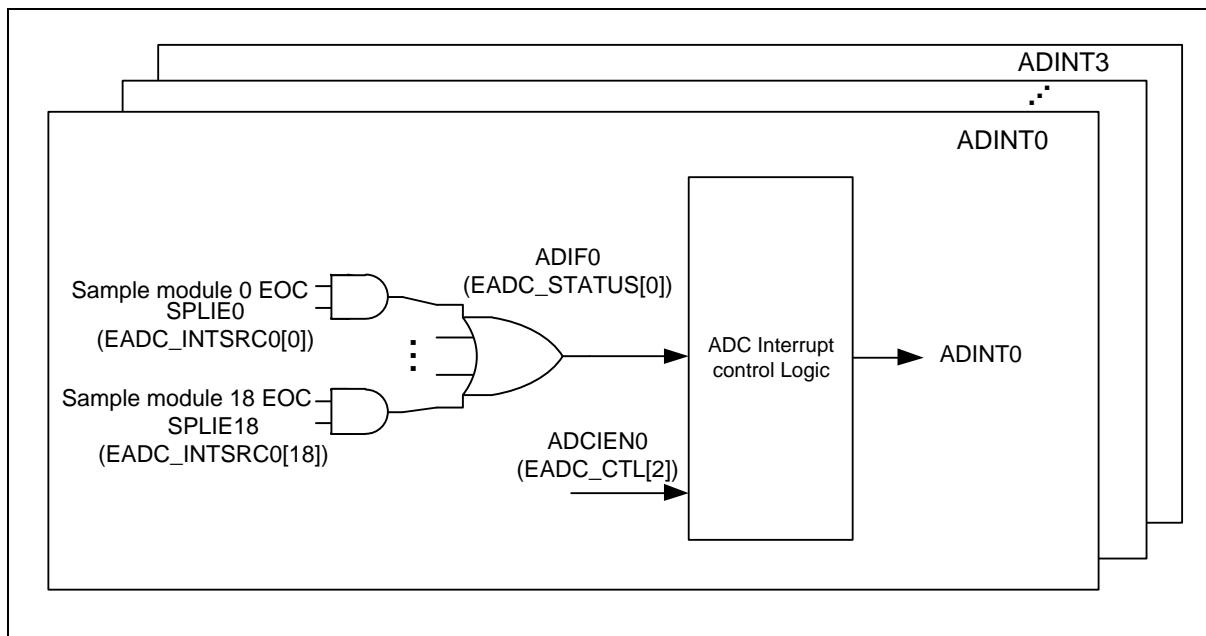


图 6.37-8 选定模块 ADC EOC 信号 ADINT0~3 中断

#### 6.37.5.7 ADC由定时器溢出触发或由外部管脚 EADC0\_ST 触发

有4个定时器和外部管脚 EADC0\_ST 可以配置为模块 0~15 的触发源.

#### 6.37.5.8 ADC 由 EPWM / BPWM 同步触发

除了软件触发, ADINT0/1 中断脉冲, 外部管脚 EADC0\_ST 和 Timer0~3 溢出脉冲都可以启动 ADC 转换, 新的特性允许 EPWM/BPWM 触发启动 ADC 转换. 要配置 EPWM/BPWM 触发类型: EPWM/BPWM 的上升沿, 下降沿或 EPWM/BPWM (仅中心对齐模式) 的中点触发ADC. 以及从触发到 ADC 开始的延时. 通过设置 TRGDLYCNT (EADC\_SCTL $n$ [15:8],  $n=0\sim15$ ) 和 TRGDLYDIV (EADC\_SCTL $n$ [7:6],  $n=8\sim15$ ) 配置延迟时间。延时可配置的EPWM/BPWM触发ADC转换时序图见 图 6.37-9

图 6.37-10 展示其他触发源配置延时时间

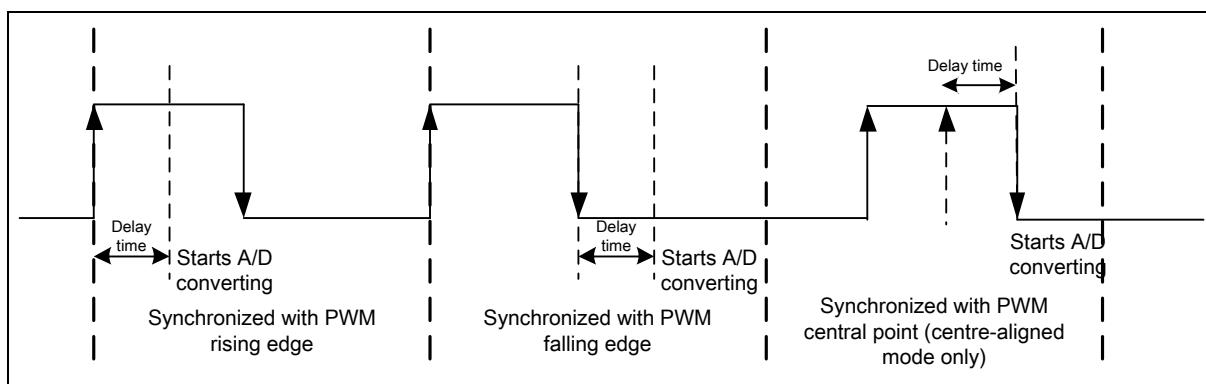


图 6.37-9 EPWM 触发 ADC 开始转换

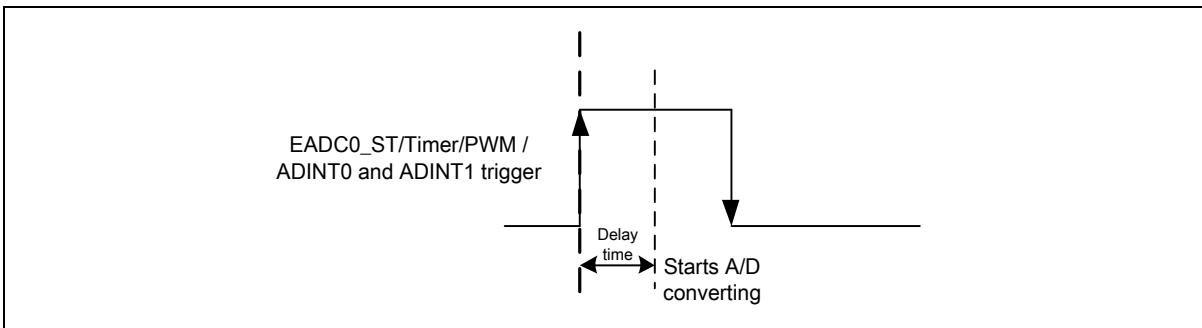


图 6.37-10 外部触发 ADC 开始转换

### 6.37.5.9 ADC 转换时间和外部触发

当 SWTRG $n$  (EADC\_SWTRG[ $n$ ],  $n=0\sim18$ ) 设为 1, 延时( $T_d$ )后采样输入然后开始转换。ADC 时钟由 PCLK 除以 (ADC(CLKDIV[23:16])+1) 得到, 从写 SWTRG $n$  到 ADC 开始采样输入的最大延时是2个 ADC 时钟周期. 延时时间如 图 6.37-11.

设置 TRGSEL (EADC\_SCTL $n$ [20:16],  $n=0\sim15$ ) 为 0x01 选择由 EADC0\_ST 引脚输入信号触发ADC。可以设置 EXTFEN (EADC\_SCTL $n$ [5],  $n=0\sim15$ ) 和 EXTREN (EADC\_SCTL $n$ [4],  $n=0\sim15$ ) 使能引脚 EADC0\_ST 触发条件是上升沿还是下降沿。如果选择上升沿触发, 低电平最少保持 2 PCLK 周期且接下来的高电平至少保持 3 PCLK 周期。如果选择下降沿触发, 高电平至少保持 2 PCLK 周期且接下来的下降沿至少保持 3 PCLK 周期. 短于设定的脉冲会被忽略. 外部管脚触发时序见 图 6.37-12.

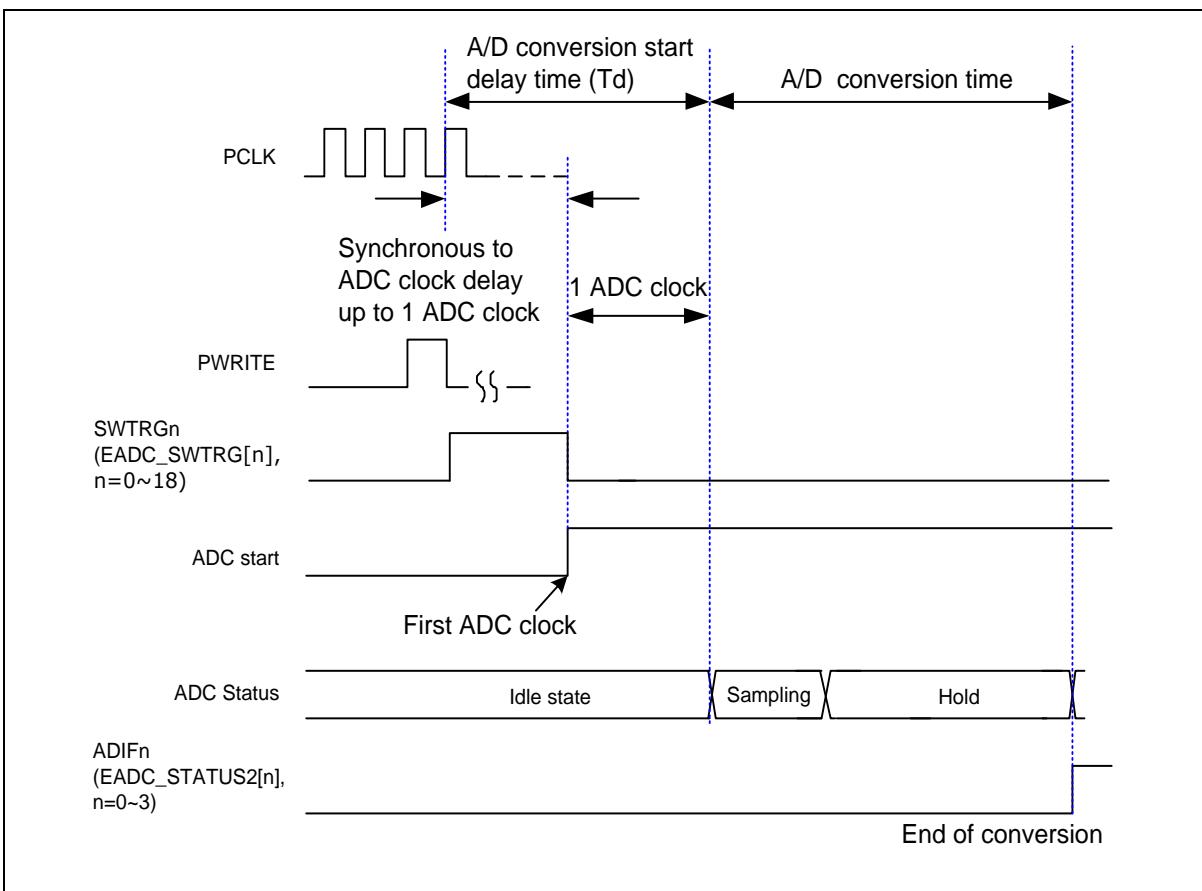


图 6.37-11 转换开始延时时间

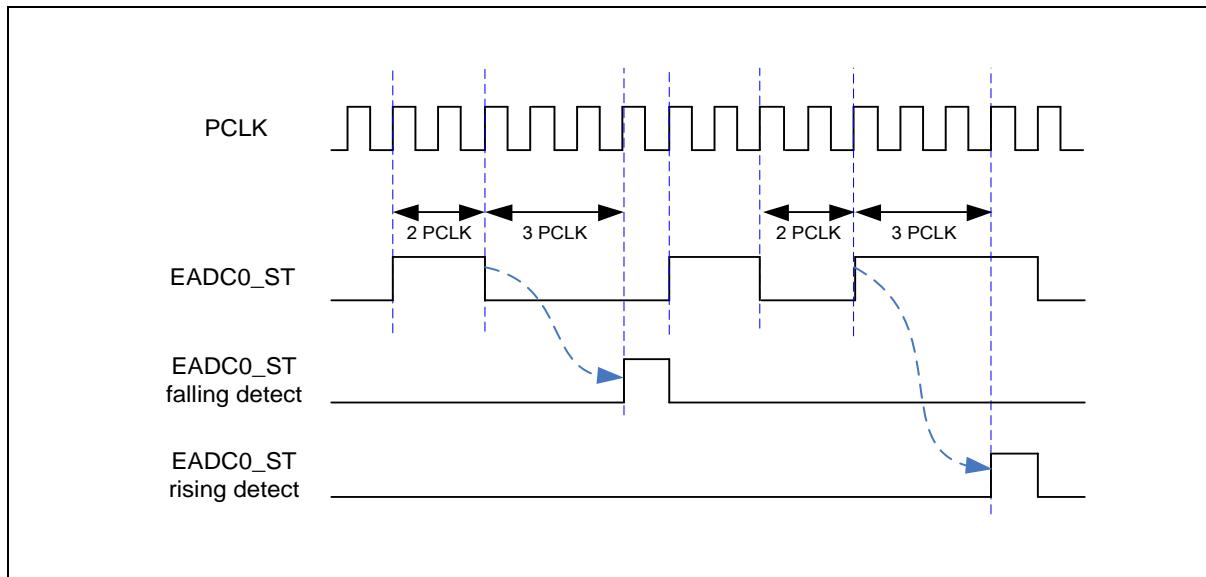


图 6.37-12 EADC0\_ST 防抖时序

#### 6.37.5.10 ADC 延长采样时间

当 ADC 采样率较高时, 如果模拟电路的输出阻抗很高, 导致充电时间过长, 模拟输入电压的采样时间可能不够。可以对每个采样通道写 EXTSMPT (EADC\_SCTL $n$ [31:24],  $n=0\sim15$ ) 延长采样时间。延长采样时间的范围是 0 ~255 ADC 时钟。延长采样时间如图 6.37-13.

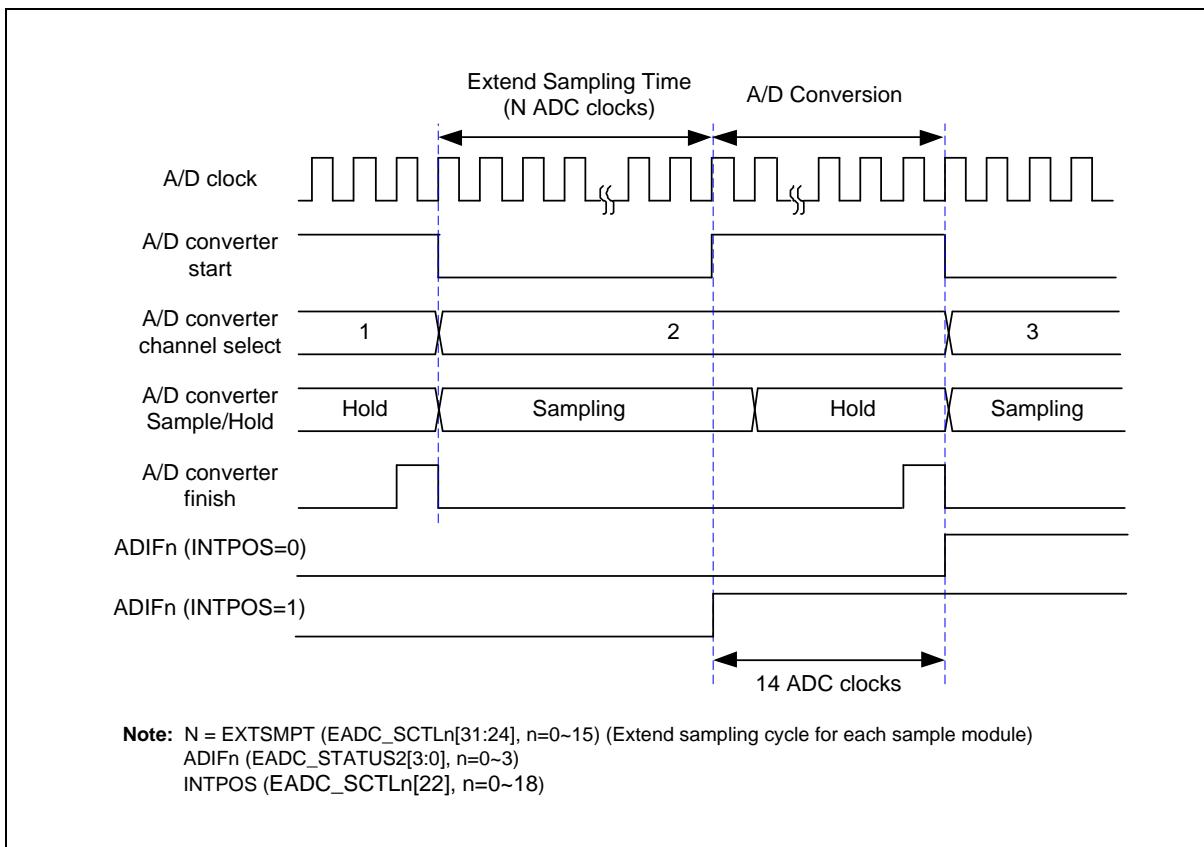


图 6.37-13 ADC 延长采样时间时序

#### 6.37.5.11 转换结果比较模式

有4个比较寄存器EADC\_CMP0 ~ EADC\_CMP3监控模块0~18中4个转换结果，如图 6.37-14. 用户设定CMPSPL (EADC\_CMPn[7:3], n =0~3)选择监控的模块， CMPCOND (EADC\_CMPn[2], n =0~3) 配置监控条件是结果小于，还是大于等于CMPDAT (EADC\_CMPn[27:16] , n =0~3)设定的值。当 CMPSPL 选择的模块转换完成，就自动比较一次。若比较结果满足比较条件，内部比较符合计数器加1。如果比较结果不满足条件 比较符合计数器重置为 0。当计数器的值大到 (CMPPMCNT (EADC\_CMPn[11:8])+1, n =0~3) 的设 定时， ADCMPFn (EADC\_STATUS2[7:4] , n =0~3) 位 置 1, 如果 ADCMPIE (EADC\_CMPn[1] , n =0~3)为1使能中断，那么 ADINT3 中断会产生。用户可以用这个功能监控外部模拟电压变化。详细的逻辑框图如图 6.37-14.

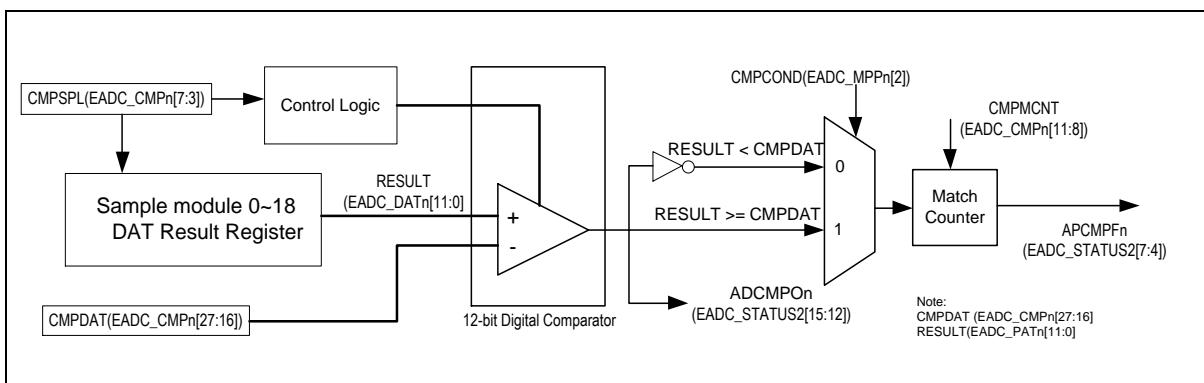


图 6.37-14 ADC 转换结果监控逻辑框图

配置 CMPWEN (EADC\_CMP0[15]/ EADC\_CMP2[15] ) 使能窗口比较模式：当 EADC\_CMP0 和 EADC\_CMP1 比较条件都满足时 ADCMPF0 (EADC\_STATUS2[4]) 置 1。当 EADC\_CMP2 和 EADC\_CMP3 比较条件都满足的时候 ADCMPF2 (EADC\_STATUS2[6]) 置 1。

#### 6.37.5.12 差分模式

DIFFEN (EADC\_CTL[8]) 配置差分模式，差分输入见表 6.37-2.

差分输入电压( $V_{diff}$ ) =  $V_{正} - V_{负}$ ,  $V_{正}$ 是模拟输入;  $V_{负}$ 是反向模拟输入

差分模拟输入通道对	ADC 模拟输入	
	$V_{plus}$	$V_{minus}$
0	EADC_CH0	EADC_CH1
1	EADC_CH2	EADC_CH3
2	EADC_CH4	EADC_CH5
3	EADC_CH6	EADC_CH7
4	EADC_CH8	EADC_CH9
5	EADC_CH10	EADC_CH11
6	EADC_CH12	EADC_CH13
7	EADC_CH14	EADC_CH15

表 6.37-2 EADC 差分模式通道选择

差分输入模式，要在 CHSEL (EADC\_SCTLn[3:0]) 中使能偶数编号通道。转换结果存放在对应使能的偶数编号通道寄存器中。若 DMOF (EADC\_CTL[9]) = 1 转换结果是二进制补码形式。

#### 6.37.5.13 双缓存模式

模块 0~3 支持双数据缓存，使能位在 DBMEN (EADC\_SCTLn[23], n=0~3)，此模式，第一次 ADC 转换结束，VALID (EADC\_DATn[17], n=0~3) 置 1，但 VALID (EADC\_DDATn[17], n=0~3) 保持 0。第二次 ADC 转换完成后 VALID (EADC\_DDATn[17], n=0~3) 置 1，用户可以在 EADC\_DATn 和 EADC\_DDATn 寄存器得到两次转换结果。

#### 6.37.5.14 PDMA 请求

使用 PDMA 传输 ADC 结果，要使能 PDMAEN (EADC\_CTL[11]) 且配置 PDMA 通道的源地址是 EADC\_CURDAT (EADC\_BA+0x4C)。任意 VALID (EADC\_DATn[17], n=0~18) 为 1，ADC 都会发 PDMA 请求。EADC\_CURDAT 寄存器是 EADC\_DAT 寄存器的影子寄存器。编号较低的模块有更高的优先级。PDMA 读 EADC\_CURDAT 寄存器之后，相应 EADC\_DAT 寄存器的 VAILD 位自动清除。

#### 6.37.5.15 中断源

在转换开始或转换完成产生 ADIFn (EADC\_STATUS2[3:0], n=0~3) 由 INTPOS (EADC\_SCTLn[22],

$n=0\sim15$ ) 决定。如果  $\text{ADCIEN}_n$  ( $\text{EADC\_CTL}[5:2]$ ,  $n=0\sim3$ ) 为 1, 转换完成产生中断请求  $\text{ADINT}_n$  ( $n=0\sim3$ )。中断控制器如图 6.37-15。

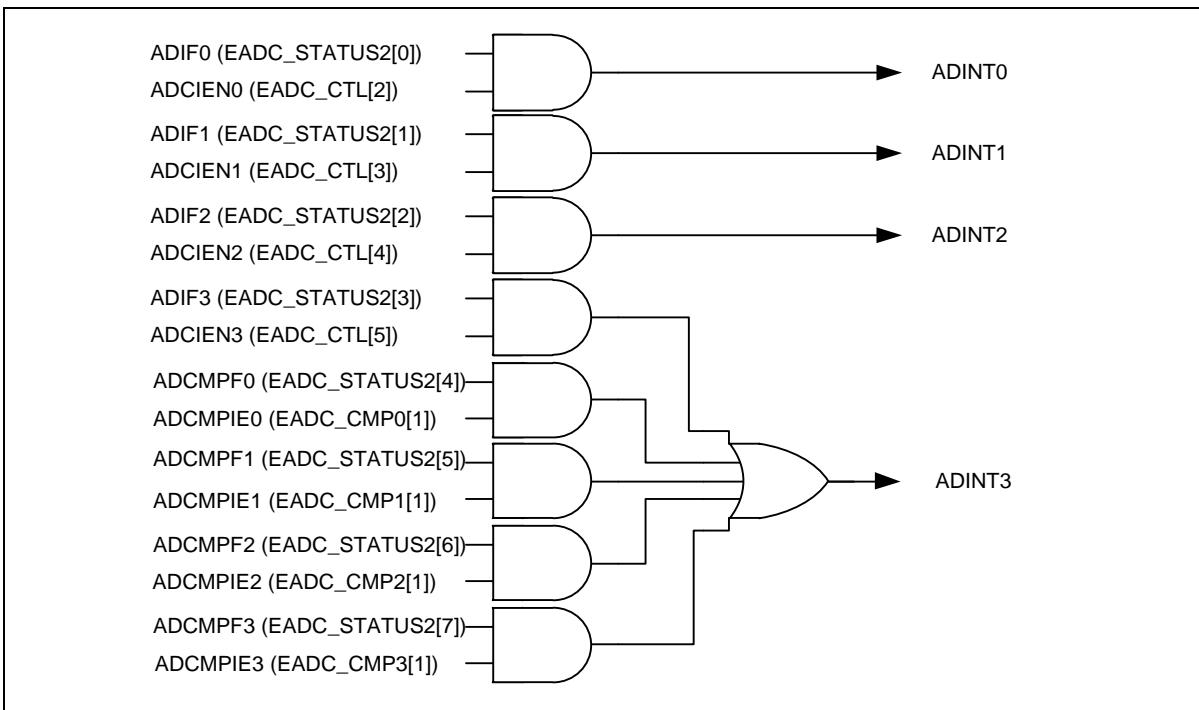


图 6.37-15 ADC 控制器中断

#### 6.37.5.16 电源管理和校准

有3种省电模式：深度掉电模式，掉电模式和待机模式。要在  $\text{ADCEN}$  ( $\text{EADC\_CTL}[0]$ )=0 时设定  $\text{PWDMOD}$  ( $\text{EADC_PWRM}[3:2]$ ) 选择模式。3种模式的不同如表 6.37-3。因为内部 LDO 会在深度掉电和掉电模时关闭，再次使能需要较长时间恢复。恢复时间由  $\text{LDOSUT}$  ( $\text{EADC_PWRM}[19:8]$ ) 设定，需要长于 20  $\mu\text{s}$ 。在待机模式，LDO 保持使能，不需要恢复时间。

电源供应	深度掉电	掉电	待机
内部 LDO	禁止	禁止	使能
内部电源开关	禁止	使能	使能

表 6.37-3 EADC 省电模式

当  $\text{ADCEN}$  ( $\text{EADC_CTL}[0]$ ) 为 1 设定 EADC 有效，便执行开始序列。开始序列完成后， $\text{PWUPRDY}$  ( $\text{EADC_PWRM}[0]$ ) 由硬件置 1 意味着准备好转换。在开始序列中， $\text{ADCEN}$  ( $\text{EADC_CTL}[0]$ ) 需要保持为 1 直到  $\text{PWUPRDY}$  ( $\text{EADC_PWRM}[0]$ ) 设为 1。在开始序列中修改  $\text{ADCEN}$  ( $\text{EADC_CTL}[0]$ ) 会导致 EADC 功能失败。

ADC 的转换结果可以通过校准来更加精确。用户可以设置  $\text{PWUCALEN}$  ( $\text{EADC_PWRM}[1]$ ) 为 1 在启动时执行校准。这一位需要与  $\text{CALSEL}$  ( $\text{EADC_CALCTL}[3]$ ) 配套设置， $\{\text{PWUCALEN}, \text{CALSEL}\}$  配置如表 6.37-4。启动校准的例子如图 6.37-16。

PWUCALEN	CALSEL	配置
0	0	无校准启动
0	1	无校准启动
1	0	启动时载入校准值
1	1	启动校准

表 6.37-4 EADC 校准启动

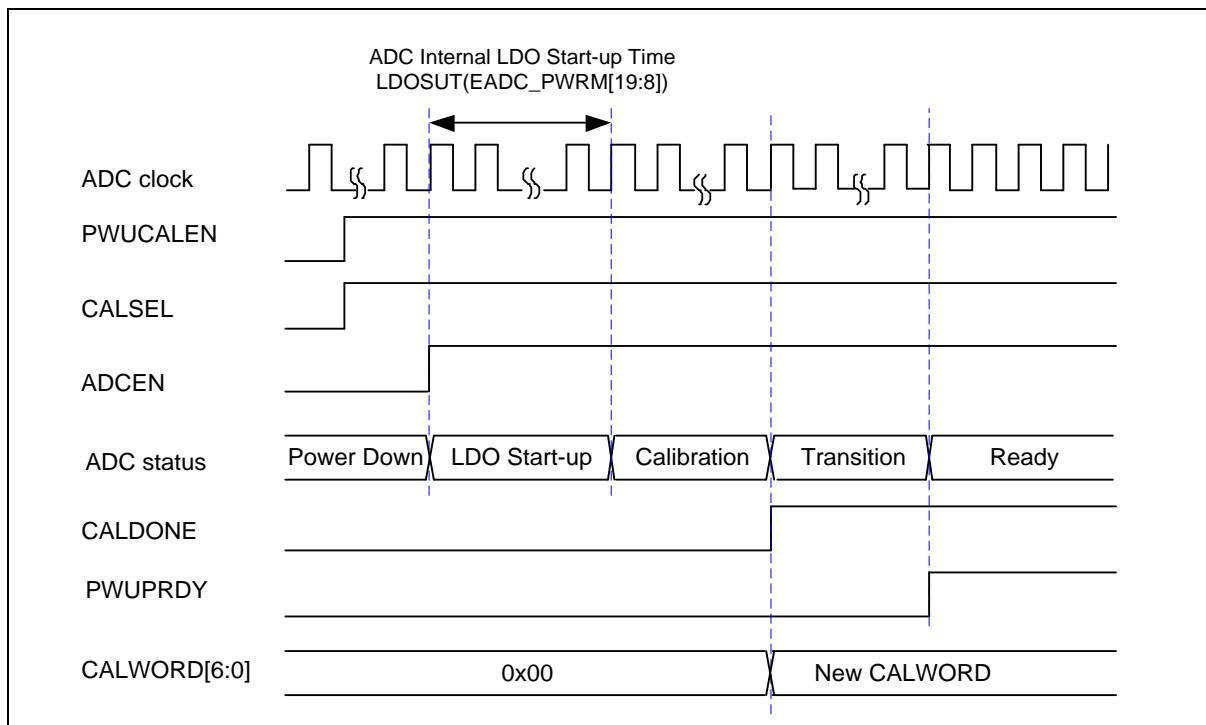


图 6.37-16 ADC 带校准的启动序列

为了得到更精密的结果，用户可以在数次转换之后再次校准。设置 CALSTART (EADC\_CALCTL[1]) 为 1 使能再次校准，这一位需要与 CALSEL (EADC\_CALCTL[3]) 同时操作。设置 CALSTART (EADC\_CALCTL[1]) 为 1 且 CALSEL (EADC\_CALCTL[3]) 为 1 会执行再次校准，然后更新 CALWORD (EADC\_CALDWRD[6:0])。设置 CALSTART (EADC\_CALCTL[1]) 为 1 且 CALSEL (EADC\_CALCTL[3]) 为 0 会载入用户定义的 CALWORD (EADC\_CALDWRD[6:0])。重新校准的流程如下所示：

1. 设置CALSEL (EADC\_CALCTL[3]) 为 1 或 0 选择校准功能
2. 设置 CALSTART (EADC\_CALCTL[1]) 为 1 来使能校准
3. CALDONE (EADC\_CALCTL[2]) 会在校准中由硬件设为 0
4. 等待 CALDONE (EADC\_CALCTL[2]) 由硬件设为 1。如果 CALSEL (EADC\_CALCTL[3]) 设为 1，新的校准值会更新到 CALWORD (EADC\_CALDWRD[6:0])。如果 CALSEL (EADC\_CALCTL[3]) 设为 0，设定的 CALWORD (EADC\_CALDWRD[6:0]) 会载入而不需要执行校准。

注：注意：在将 CALSTART（EADC\_CALCTL [1]）设置为1以再次开始校准之前，EADCDIV（CKL\_CLKDIV0 [23:16]）必须为0。

### 6.37.6 寄存器映射

R: 只读, W: 只写, R/W: 可读可写

寄存器	偏移量	R/W	描述	复位值
<b>EADC 基地址:</b>				
<b>EADC_BA = 0x4004_3000</b>				
<b>EADC_DAT0</b>	EADC_BA+0x00	R	ADC 采样模块 0 数据寄存器 0	0x0000_0000
<b>EADC_DAT1</b>	EADC_BA+0x04	R	ADC 采样模块 1 数据寄存器 1	0x0000_0000
<b>EADC_DAT2</b>	EADC_BA+0x08	R	ADC 采样模块 2 数据寄存器 2	0x0000_0000
<b>EADC_DAT3</b>	EADC_BA+0x0C	R	ADC 采样模块 3 数据寄存器 3	0x0000_0000
<b>EADC_DAT4</b>	EADC_BA+0x10	R	ADC 采样模块 4 数据寄存器 4	0x0000_0000
<b>EADC_DAT5</b>	EADC_BA+0x14	R	ADC 采样模块 5 数据寄存器 5	0x0000_0000
<b>EADC_DAT6</b>	EADC_BA+0x18	R	ADC 采样模块 6 数据寄存器 6	0x0000_0000
<b>EADC_DAT7</b>	EADC_BA+0x1C	R	ADC 采样模块 7 数据寄存器 7	0x0000_0000
<b>EADC_DAT8</b>	EADC_BA+0x20	R	ADC 采样模块 8 数据寄存器 8	0x0000_0000
<b>EADC_DAT9</b>	EADC_BA+0x24	R	ADC 采样模块 9 数据寄存器 9	0x0000_0000
<b>EADC_DAT10</b>	EADC_BA+0x28	R	ADC 采样模块 10 数据寄存器 10	0x0000_0000
<b>EADC_DAT11</b>	EADC_BA+0x2C	R	ADC 采样模块 11 数据寄存器 11	0x0000_0000
<b>EADC_DAT12</b>	EADC_BA+0x30	R	ADC 采样模块 12 数据寄存器 12	0x0000_0000
<b>EADC_DAT13</b>	EADC_BA+0x34	R	ADC 采样模块 13 数据寄存器 13	0x0000_0000
<b>EADC_DAT14</b>	EADC_BA+0x38	R	ADC 采样模块 14 数据寄存器 14	0x0000_0000
<b>EADC_DAT15</b>	EADC_BA+0x3C	R	ADC 采样模块 15 数据寄存器 15	0x0000_0000
<b>EADC_DAT16</b>	EADC_BA+0x40	R	ADC 采样模块 16 数据寄存器 16	0x0000_0000
<b>EADC_DAT17</b>	EADC_BA+0x44	R	ADC 采样模块 17 数据寄存器 17	0x0000_0000
<b>EADC_DAT18</b>	EADC_BA+0x48	R	ADC 采样模块 18 数据寄存器 18	0x0000_0000
<b>EADC_CURDAT</b>	EADC_BA+0x4C	R	ADC PDMA 目前传输数据寄存器	0x0000_0000
<b>EADC_CTL</b>	EADC_BA+0x50	R/W	ADC 控制寄存器	0x0000_00C0
<b>EADC_SWTRG</b>	EADC_BA+0x54	W	ADC 采样模块软件触发寄存器	0x0000_0000
<b>EADC_PENDSTS</b>	EADC_BA+0x58	R/W	ADC 开始转换等待标志寄存器	0x0000_0000
<b>EADC_OVSTS</b>	EADC_BA+0x5C	R/W	ADC 采样模块开始转换覆盖标志寄存器	0x0000_0000
<b>EADC_SCTL0</b>	EADC_BA+0x80	R/W	ADC 采样模块 0 控制寄存器	0x0000_0000

寄存器	偏移量	R/W	描述	复位值
<b>EADC 基地址:</b>				
<b>EADC_BA = 0x4004_3000</b>				
<b>EADC_SCTL1</b>	EADC_BA+0x84	R/W	ADC 采样模块 1 控制寄存器	0x0000_0000
<b>EADC_SCTL2</b>	EADC_BA+0x88	R/W	ADC 采样模块 2 控制寄存器	0x0000_0000
<b>EADC_SCTL3</b>	EADC_BA+0x8C	R/W	ADC 采样模块 3 控制寄存器	0x0000_0000
<b>EADC_SCTL4</b>	EADC_BA+0x90	R/W	ADC 采样模块 4 控制寄存器	0x0000_0000
<b>EADC_SCTL5</b>	EADC_BA+0x94	R/W	ADC 采样模块 5 控制寄存器	0x0000_0000
<b>EADC_SCTL6</b>	EADC_BA+0x98	R/W	ADC 采样模块 6 控制寄存器	0x0000_0000
<b>EADC_SCTL7</b>	EADC_BA+0x9C	R/W	ADC 采样模块 7 控制寄存器	0x0000_0000
<b>EADC_SCTL8</b>	EADC_BA+0xA0	R/W	ADC 采样模块 8 控制寄存器	0x0000_0000
<b>EADC_SCTL9</b>	EADC_BA+0xA4	R/W	ADC 采样模块 9 控制寄存器	0x0000_0000
<b>EADC_SCTL10</b>	EADC_BA+0xA8	R/W	ADC 采样模块 10 控制寄存器	0x0000_0000
<b>EADC_SCTL11</b>	EADC_BA+0xAC	R/W	ADC 采样模块 11 控制寄存器	0x0000_0000
<b>EADC_SCTL12</b>	EADC_BA+0xB0	R/W	ADC 采样模块 12 控制寄存器	0x0000_0000
<b>EADC_SCTL13</b>	EADC_BA+0xB4	R/W	ADC 采样模块 13 控制寄存器	0x0000_0000
<b>EADC_SCTL14</b>	EADC_BA+0xB8	R/W	ADC 采样模块 14 控制寄存器	0x0000_0000
<b>EADC_SCTL15</b>	EADC_BA+0xBC	R/W	ADC 采样模块 15 控制寄存器	0x0000_0000
<b>EADC_SCTL16</b>	EADC_BA+0xC0	R/W	ADC 采样模块 16 控制寄存器	0x0000_0000
<b>EADC_SCTL17</b>	EADC_BA+0xC4	R/W	ADC 采样模块 17 控制寄存器	0x0000_0000
<b>EADC_SCTL18</b>	EADC_BA+0xC8	R/W	ADC 采样模块 18 控制寄存器	0x0000_0000
<b>EADC_INTSRC0</b>	EADC_BA+0xD0	R/W	ADC 中断 0 源使能控制寄存器.	0x0000_0000
<b>EADC_INTSRC1</b>	EADC_BA+0xD4	R/W	ADC 中断 1 源使能控制寄存器.	0x0000_0000
<b>EADC_INTSRC2</b>	EADC_BA+0xD8	R/W	ADC 中断 2 源使能控制寄存器.	0x0000_0000
<b>EADC_INTSRC3</b>	EADC_BA+0xDC	R/W	ADC 中断 3 源使能控制寄存器.	0x0000_0000
<b>EADC_CMP0</b>	EADC_BA+0xE0	R/W	ADC 结果比较寄存器 0	0x0000_0000
<b>EADC_CMP1</b>	EADC_BA+0xE4	R/W	ADC 结果比较寄存器 1	0x0000_0000
<b>EADC_CMP2</b>	EADC_BA+0xE8	R/W	ADC 结果比较寄存器 2	0x0000_0000
<b>EADC_CMP3</b>	EADC_BA+0xEC	R/W	ADC 结果比较寄存器 3	0x0000_0000
<b>EADC_STATUS0</b>	EADC_BA+0xF0	R	ADC 状态寄存器 0	0x0000_0000
<b>EADC_STATUS1</b>	EADC_BA+0xF4	R	ADC 状态寄存器 1	0x0000_0000

寄存器	偏移量	R/W	描述	复位值
<b>EADC 基地址:</b> <b>EADC_BA = 0x4004_3000</b>				
<b>EADC_STATUS2</b>	EADC_BA+0xF8	R/W	ADC 状态寄存器 2	0x0000_0000
<b>EADC_STATUS3</b>	EADC_BA+0xFC	R	ADC 状态寄存器 3	0x0000_001F
<b>EADC_DDAT0</b>	EADC_BA+0x100	R	ADC 采样模块 0 双数据寄存器 0	0x0000_0000
<b>EADC_DDAT1</b>	EADC_BA+0x104	R	ADC 采样模块 1 双数据寄存器 1	0x0000_0000
<b>EADC_DDAT2</b>	EADC_BA+0x108	R	ADC 采样模块 2 双数据寄存器 2	0x0000_0000
<b>EADC_DDAT3</b>	EADC_BA+0x10C	R	ADC 采样模块 3 双数据寄存器 3	0x0000_0000
<b>EADC_PWRM</b>	EADC_BA+0x110	R/W	ADC 电源管理寄存器	0x0006_E012
<b>EADC_CALCTL</b>	EADC_BA+0x114	R/W	ADC 校准控制寄存器	0x0000_0008
<b>EADC_CALDWRD</b>	EADC_BA+0x118	R/W	ADC 校准载入值寄存器	0x0000_00XX

### 6.37.7 寄存器描述

#### EADC DAT0~18 ADC 数据寄存器

寄存器	偏移量	R/W	描述	复位值
EADC_DAT0	EADC_BA+0x00	R	ADC 采样模块 0 数据寄存器	0x0000_0000
EADC_DAT1	EADC_BA+0x04	R	ADC 采样模块 1 数据寄存器	0x0000_0000
EADC_DAT2	EADC_BA+0x08	R	ADC 采样模块 2 数据寄存器	0x0000_0000
EADC_DAT3	EADC_BA+0x0C	R	ADC 采样模块 3 数据寄存器	0x0000_0000
EADC_DAT4	EADC_BA+0x10	R	ADC 采样模块 4 数据寄存器	0x0000_0000
EADC_DAT5	EADC_BA+0x14	R	ADC 采样模块 5 数据寄存器	0x0000_0000
EADC_DAT6	EADC_BA+0x18	R	ADC 采样模块 6 数据寄存器	0x0000_0000
EADC_DAT7	EADC_BA+0x1C	R	ADC 采样模块 7 数据寄存器	0x0000_0000
EADC_DAT8	EADC_BA+0x20	R	ADC 采样模块 8 数据寄存器	0x0000_0000
EADC_DAT9	EADC_BA+0x24	R	ADC 采样模块 9 数据寄存器	0x0000_0000
EADC_DAT10	EADC_BA+0x28	R	ADC 采样模块 10 数据寄存器	0x0000_0000
EADC_DAT11	EADC_BA+0x2C	R	ADC 采样模块 11 数据寄存器	0x0000_0000
EADC_DAT12	EADC_BA+0x30	R	ADC 采样模块 12 数据寄存器	0x0000_0000
EADC_DAT13	EADC_BA+0x34	R	ADC 采样模块 13 数据寄存器	0x0000_0000
EADC_DAT14	EADC_BA+0x38	R	ADC 采样模块 14 数据寄存器	0x0000_0000
EADC_DAT15	EADC_BA+0x3C	R	ADC 采样模块 15 数据寄存器	0x0000_0000
EADC_DAT16	EADC_BA+0x40	R	ADC 采样模块 16 数据寄存器	0x0000_0000
EADC_DAT17	EADC_BA+0x44	R	ADC 采样模块 17 数据寄存器	0x0000_0000
EADC_DAT18	EADC_BA+0x48	R	ADC 采样模块 18 数据寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved						VALID	OV
15	14	13	12	11	10	9	8
RESULT							
7	6	5	4	3	2	1	0
RESULT							

位	描述	
[31:18]	<b>Reserved</b>	保留.
[17]	<b>VALID</b>	<b>有效标志</b> 转换完成后置 1， EADC_DAT被读取后硬件清0 0 = RESULT[11:0]数据无效. 1 = RESULT[11:0] 数据有效.
[16]	<b>OV</b>	<b>覆盖标志</b> RESULT[11:0] 里的转换结果在被读出前，又有新的转换结果存入， OV会设为 1. 0 = RESULT[11:0]的结果是最近的转换结果. 1 = RESULT[11:0] 被覆盖. <b>注意：</b> EADC_DAT寄存器被读取后此位被硬件清0.
[15:0]	<b>RESULT</b>	<b>ADC 转换结果</b> 这里是 12 位的转换结果. 当 DMOF (EADC_CTL[9])设为 0, 12-位 ADC 转换结果是 RESULT[11:0] 的无符号数， RESULT[15:12]会填0. 当 DMOF (EADC_CTL[9])设为 1, 12-位 ADC 转换结果是RESULT[11:0] 的有符号二进制补码， RESULT[15:12]填符号位.

**EADC\_CURDAT ADC PDMA 目前传输数据寄存器**

寄存器	偏移量	R/W	描述	复位值
EADC_CURDAT	EADC_BA+0x4C	R	ADC PDMA目前传输数据寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved						CURDAT	
15	14	13	12	11	10	9	8
CURDAT							
7	6	5	4	3	2	1	0
CURDAT							

位	描述	
[31:18]	Reserved	保留.
[17:0]	CURDAT	<b>ADC PDMA 目前传输数据寄存器 (只读)</b> 这个寄存器是 EADC_DATn (n=0~18) 的影子寄存器，为了支持 PDMA .

**EADC\_CTL ADC 控制寄存器**

寄存器	偏移量	R/W	描述	复位值
EADC_CTL	EADC_BA+0x50	R/W	ADC控制寄存器	0x0000_00C0

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved				PDMAEN	Reserved	DMOF	DIFFEN
7	6	5	4	3	2	1	0
RESSEL		ADCIEN3	ADCIEN2	ADCIEN1	ADCIEN0	ADCRST	ADCEN

位	描述	
[31:12]	Reserved	保留.
[11]	PDMAEN	<b>PDMA 传输使能位</b> 此位置1时，当ADC转换完成会产生PDMA数据传输请求。 0 = PDMA 数据传输禁止 1 = PDMA 数据传输使能. <b>注意：</b> 当此位为1，用户需要设置ADCIENN (EADC_CTL[5:2], n=0~3) = 0 禁止中断。
[10]	Reserved	保留.
[9]	DMOF	<b>差分模式输出格式</b> 0 = ADC 转换结果是无符号数的格式. 1 = ADC转换结果是二进制补码的格式
[8]	DIFFEN	<b>差分模式使能位</b> 0 = 单端模拟输入模式. 1 = 差分模拟输入模式.
[7:6]	RESSEL	<b>ADC 结果位数选择</b> 00 = 6位 ADC 结果存入 RESULT (EADC_DATn[5:0]). 01 = 8位 ADC 结果存入 RESULT (EADC_DATn[7:0]). 10 = 10位 ADC结果存入RESULT (EADC_DATn[9:0]). 11 = 12位 ADC 结果存入RESULT (EADC_DATn[11:0]).
[5]	ADCIEN3	<b>ADINT3中断使能位</b> 选定的通道转换完成后，完成标志ADIF3 (EADC_STATUS2[3]) 置1，如果ADCIEN3 为1会产生转换完成中断ADINT3 0 = ADINT3 中断禁止 1 = ADINT3中断使能.

位	描述	
[4]	<b>ADCIEN2</b>	<b>ADINT2中断使能位</b> 选定的通道转换完成后，完成标志ADIF2 (EADC_STATUS2[2]) 置1，如果ADCIEN2为1会产生转换完成中断ADINT2 0 = ADINT2 中断禁止 1 = ADINT2中断使能.
[3]	<b>ADCIEN1</b>	<b>ADINT1中断使能位</b> 选定的通道转换完成后，完成标志ADIF1 (EADC_STATUS2[1]) 置1，如果ADCIEN1为1会产生转换完成中断ADINT1 0 = ADINT1 中断禁止 1 = ADINT1中断使能
[2]	<b>ADCIEN0</b>	<b>ADINT0中断使能位</b> 选定的通道转换完成后，完成标志ADIF0 (EADC_STATUS2[0]) 置1，如果ADCIEN0 为1会产生转换完成中断ADINT0 0 = ADINT0 中断禁止 1 = ADINT0中断使能
[1]	<b>ADCRST</b>	<b>ADC 复位</b> 0 = 没影响. 1 = ADC 复位到初始态，但不会改变 ADC 寄存器的值. <b>注意:</b> ADCRST 在 ADC 复位时保持为1，当 ADC复位结束，ADCRST位会自动清 0.
[0]	<b>ADCEN</b>	<b>ADC使能位</b> 0 = 禁止 EADC. 1 =使能 EADC. <b>注意:</b> 开始转换前，这一位要设为 1. 此位清0 ADC 模拟电路断电节省功耗.

**EADC\_SWTRG ADC 软件启动寄存器**

寄存器	偏移量	R/W	描述	复位值
<b>EADC_SWTRG</b>	EADC_BA+0x54	W	ADC采样模块软件启动寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved					SWTRG		
15	14	13	12	11	10	9	8
SWTRG							
7	6	5	4	3	2	1	0
SWTRG							

位	描述	
[31:19]	<b>Reserved</b>	保留.
[18:0]	<b>SWTRG</b>	<p><b>采样模块 0~18 软件启动</b>            0 = 没影响.            1 =当该模块获得优先级时，启动 ADC 转换  <b>注意：</b>写这个寄存器启动 ADC转换后，EADC_PENDSTS 寄存器显示哪一个通道会被转换。            如果用户想要禁止采样模块转换，可以写EADC_PENDSTS 寄存器清除.</p>

**EADC\_PENDSTS ADC模块转换等待标志寄存器**

寄存器	偏移量	R/W	描述	复位值
<b>EADC_PENDSTS</b>	EADC_BA+0x58	R/W	ADC开始转换等待标志寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved					STPF		
15	14	13	12	11	10	9	8
STPF							
7	6	5	4	3	2	1	0
STPF							

位	描述	
[31:19]	<b>Reserved</b>	保留.
[18:0]	<b>STPF</b>	<p><b>ADC 模块 0~18 转换等待标志</b></p> <p><b>读:</b> 0 = 模块没有等待转换. 1 = 模块在等待转换.</p> <p><b>写:</b> 1 = 清等待标志且取消采样模块转换.</p> <p><b>注意:</b>在等待转换期间对应位为1，当对应的 ADC 转换完成自动清 0.</p>

**EADC\_OVSTS ADC 模块覆盖标志寄存器**

寄存器	偏移量	R/W	描述	复位值
EADC_OVSTS	EADC_BA+0x5C	R/W	ADC 模块开始转换覆盖标志寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved				SPOVF			
15	14	13	12	11	10	9	8
SPOVF							
7	6	5	4	3	2	1	0
SPOVF							

位	描述	
[31:19]	<b>Reserved</b>	保留.
[18:0]	<b>SPOVF</b>	<p><b>ADC 模块0~18 覆盖标志</b></p> <p>0 = 没有覆盖事件. 1 = 新的事件已产生而旧的事件还在等待</p> <p><b>注意:</b> 此位写1清0.</p>

**EADC\_SCTL0~3 ADC 模块 0~3 控制寄存器**

寄存器	偏移量	R/W	描述	复位值
<b>EADC_SCTL0</b>	EADC_BA+0x80	R/W	ADC 模块 0 控制寄存器	0x0000_0000
<b>EADC_SCTL1</b>	EADC_BA+0x84	R/W	ADC 模块 1 控制寄存器	0x0000_0000
<b>EADC_SCTL2</b>	EADC_BA+0x88	R/W	ADC 模块 2 控制寄存器	0x0000_0000
<b>EADC_SCTL3</b>	EADC_BA+0x8C	R/W	ADC 模块 3 控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
<b>EXTSMPT</b>							
23	22	21	20	19	18	17	16
<b>DBMEN</b>	<b>INTPOS</b>	<b>Reserved</b>	<b>TRGSEL</b>				
15	14	13	12	11	10	9	8
<b>TRGDLYCNT</b>							
7	6	5	4	3	2	1	0
<b>TRGDLYDIV</b>		<b>EXTFEN</b>	<b>EXTREN</b>	<b>CHSEL</b>			

位	描述	
[31:24]	<b>EXTSMPT</b>	<b>ADC 采样时间加长</b> 当 ADC 转换速率较高，并且外部模拟电路的输出阻抗较高时，采样时间短，可能采样保持电容还没充完电就开始转换，ADC结果会偏小。这种情况可以加长采样时间。 延时范围从 0~255 ADC时钟。
[23]	<b>DBMEN</b>	<b>双缓存模式使能位</b> 0 = 只有一个结果寄存器 (默认). 1 = 有2个结果寄存器.
[22]	<b>INTPOS</b>	<b>中断标志位置选择</b> 0 = 在ADC转换结束设置中断标志 ADIFn (EADC_STATUS2[n], n=0~3). 1 = 在ADC转换开始设置中断标志ADIFn (EADC_STATUS2[n], n=0~3)
[21]	<b>Reserved</b>	保留.

位	描述
[20:16]	<b>TRGSEL</b> <b>ADC 模块转换触发源选择</b> 0H = 禁止触发. 1H = EADC0_ST外部管脚触发. 2H = ADC ADINT0 中断 EOC (转换完成) 脉冲触发. 3H = ADC ADINT1 中断 EOC (转换完成) 脉冲触发. 4H = Timer0 溢出脉冲触发. 5H = Timer1 溢出脉冲触发. 6H = Timer2溢出脉冲触发. 7H = Timer3溢出脉冲触发. 8H = EPWM0TG0. 9H = EPWM0TG1. AH = EPWM0TG2. BH = EPWM0TG3. CH = EPWM0TG4. DH = EPWM0TG5. EH = EPWM1TG0. FH = EPWM1TG1. 10H = EPWM1TG2. 11H = EPWM1TG3. 12H = EPWM1TG4. 13H = EPWM1TG5. 14H = BPWM0TG. 15H = BPWM1TG. 其他 = 保留.
[15:8]	<b>TRGDLYCNT</b> <b>ADC 模块转换触发延迟时间</b> 触发延时 = TRGDLYCNT x ADC_CLK x n (n=1,2,4,16 由 TRGDLYDIV设定).
[7:6]	<b>TRGDLYDIV</b> <b>ADC 模块转换触发延时除频选择</b> 延时时钟频率: 00 = ADC_CLK/1. 01 = ADC_CLK/2. 10 = ADC_CLK/4. 11 = ADC_CLK/16.
[5]	<b>EXTFEN</b> <b>ADC 外部触发下降沿使能位</b> 0 =当 ADC 选择 EADC0_ST 作为触发源, 禁止下降沿触发. 1 =当 ADC 选择 EADC0_ST 作为触发源, 使能下降沿触发.
[4]	<b>EXTREN</b> <b>ADC 外部触发上升沿使能位</b> 0 =当 ADC 选择 EADC0_ST 作为触发源, 禁止上升沿触发. 1 =当 ADC 选择 EADC0_ST 作为触发源, 使能上升沿触发.

位	描述
[3:0]	<b>CHSEL</b>  ADC 模块通道选择 00H = EADC_CH0 (慢速通道). 01H = EADC_CH1 (慢速通道). 02H = EADC_CH2 (慢速通道). 03H = EADC_CH3 (慢速通道). 04H = EADC_CH4 (慢速通道). 05H = EADC_CH5 (慢速通道). 06H = EADC_CH6 (慢速通道). 07H = EADC_CH7 (慢速通道). 08H = EADC_CH8 (慢速通道). 09H = EADC_CH9 (慢速通道). 0AH = EADC_CH10 (快速通道). 0BH = EADC_CH11 (快速通道). 0CH = EADC_CH12 (快速通道). 0DH = EADC_CH13 (快速通道). 0EH = EADC_CH14 (快速通道). 0FH = EADC_CH15 (快速通道).

**EADC\_SCTL4~15 ADC 模块 4~15 控制寄存器**

寄存器	偏移量	R/W	描述	复位值
EADC_SCTL4	EADC_BA+0x90	R/W	ADC 采样模块 4 控制寄存器	0x0000_0000
EADC_SCTL5	EADC_BA+0x94	R/W	ADC 采样模块 5 控制寄存器	0x0000_0000
EADC_SCTL6	EADC_BA+0x98	R/W	ADC 采样模块 6 控制寄存器	0x0000_0000
EADC_SCTL7	EADC_BA+0x9C	R/W	ADC 采样模块 7 控制寄存器	0x0000_0000
EADC_SCTL8	EADC_BA+0xA0	R/W	ADC 采样模块 8 控制寄存器	0x0000_0000
EADC_SCTL9	EADC_BA+0xA4	R/W	ADC 采样模块 9 控制寄存器	0x0000_0000
EADC_SCTL10	EADC_BA+0xA8	R/W	ADC 采样模块 10 控制寄存器	0x0000_0000
EADC_SCTL11	EADC_BA+0xAC	R/W	ADC 采样模块 11 控制寄存器	0x0000_0000
EADC_SCTL12	EADC_BA+0xB0	R/W	ADC 采样模块 12 控制寄存器	0x0000_0000
EADC_SCTL13	EADC_BA+0xB4	R/W	ADC 采样模块 13 控制寄存器	0x0000_0000
EADC_SCTL14	EADC_BA+0xB8	R/W	ADC 采样模块 14 控制寄存器	0x0000_0000
EADC_SCTL15	EADC_BA+0xBC	R/W	ADC 采样模块 15 控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
EXTSMPT							
23	22	21	20	19	18	17	16
Reserved	INTPOS	Reserved	TRGSEL				
15	14	13	12	11	10	9	8
TRGDLYCNT							
7	6	5	4	3	2	1	0
TRGDLYDIV		EXTFEN	EXTREN	CHSEL			

位	描述	
[31:24]	EXTSMPT	<b>ADC 采样时间加长</b> 当 ADC 转换速率较高，并且外部模拟电路的输出阻抗较高时，采样时间短，可能采样保持电容还没充完电就开始转换，ADC结果会偏小。这种情况可以加长采样时间。 延时范围从 0~255 ADC时钟。
[23]	Reserved	保留。
[22]	INTPOS	<b>中断标志位置选择</b> 0 = 在ADC转换结束设置中断标志ADIFn (EADC_STATUS2[n], n=0~3). 1 =在ADC转换开始设置中断标志ADIFn (EADC_STATUS2[n], n=0~3)
[21]	Reserved	保留。

位	描述
[20:16]	<b>TRGSEL</b> <b>ADC 模块转换触发源选择</b> 0H = 禁止触发. 1H = EADC0_ST外部管脚触发. 2H = ADC ADINT0 中断 EOC (转换完成) 脉冲触发. 3H = ADC ADINT1 中断 EOC (转换完成) 脉冲触发. 4H = Timer0 溢出脉冲触发. 5H = Timer1 溢出脉冲触发. 6H = Timer2溢出脉冲触发. 7H = Timer3溢出脉冲触发. 8H = EPWM0TG0. 9H = EPWM0TG1. AH = EPWM0TG2. BH = EPWM0TG3. CH = EPWM0TG4. DH = EPWM0TG5. EH = EPWM1TG0. FH = EPWM1TG1. 10H = EPWM1TG2. 11H = EPWM1TG3. 12H = EPWM1TG4. 13H = EPWM1TG5. 14H = BPWM0TG. 15H = BPWM1TG. 其他 = 保留.
[15:8]	<b>TRGDLYCNT</b> <b>ADC 模块转换触发延迟时间</b> 触发延时 = TRGDLYCNT x ADC_CLK x n (n=1,2,4,16 由 TRGDLYDIV设定).
[7:6]	<b>TRGDLYDIV</b> <b>ADC 模块转换触发延时除频选择</b> 延时时钟频率: 00 = ADC_CLK/1. 01 = ADC_CLK/2. 10 = ADC_CLK/4. 11 = ADC_CLK/16.
[5]	<b>EXTFEN</b> <b>ADC 外部触发下降沿使能位</b> 0 =当 ADC 选择 EADC0_ST 作为触发源, 禁止下降沿触发. 1 =当 ADC 选择 EADC0_ST 作为触发源, 使能下降沿触发.
[4]	<b>EXTREN</b> <b>ADC 外部触发上升沿使能位</b> 0 =当 ADC 选择 EADC0_ST 作为触发源, 禁止上升沿触发. 1 =当 ADC 选择 EADC0_ST 作为触发源, 使能上升沿触发.

位	描述
[3:0]	<b>CHSEL</b>  ADC 模块通道选择 00H = EADC_CH0 (慢速通道). 01H = EADC_CH1 (慢速通道). 02H = EADC_CH2 (慢速通道). 03H = EADC_CH3 (慢速通道). 04H = EADC_CH4 (慢速通道). 05H = EADC_CH5 (慢速通道). 06H = EADC_CH6 (慢速通道). 07H = EADC_CH7 (慢速通道). 08H = EADC_CH8 (慢速通道). 09H = EADC_CH9 (慢速通道). 0AH = EADC_CH10 (快速通道). 0BH = EADC_CH11 (快速通道). 0CH = EADC_CH12 (快速通道). 0DH = EADC_CH13 (快速通道). 0EH = EADC_CH14 (快速通道). 0FH = EADC_CH15 (快速通道).

**EADC\_SCTL16~18 ADC 模块 16~18 控制寄存器**

寄存器	偏移量	R/W	描述	复位值
<b>EADC_SCTL16</b>	EADC_BA+0xC0	R/W	ADC 采样模块 16 控制寄存器	0x0000_0000
<b>EADC_SCTL17</b>	EADC_BA+0xC4	R/W	ADC 采样模块 17 控制寄存器	0x0000_0000
<b>EADC_SCTL18</b>	EADC_BA+0xC8	R/W	ADC 采样模块 18 控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
EXTSMPT							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved							

位	描述	
[31:24]	<b>EXTSMPT</b>	<b>ADC 采样时间延长</b> 当 ADC 转换速率较高时，采样时间短，可能采样保持电容还没充完电就开始转换，ADC结果会偏小。这种情况可以加长采样时间。 延时范围从 0~255 ADC 时钟。
[23:0]	<b>Reserved</b>	保留。

**EADC\_INTSRC0~3 ADC 中断源使能控制寄存器**

寄存器	偏移量	R/W	描述	复位值
<b>EADC_INTSRC0</b>	EADC_BA+0xD0	R/W	ADC 中断 0 源使能控制寄存器.	0x0000_0000
<b>EADC_INTSRC1</b>	EADC_BA+0xD4	R/W	ADC 中断 1 源使能控制寄存器.	0x0000_0000
<b>EADC_INTSRC2</b>	EADC_BA+0xD8	R/W	ADC 中断 2 源使能控制寄存器.	0x0000_0000
<b>EADC_INTSRC3</b>	EADC_BA+0xDC	R/W	ADC 中断 3 源使能控制寄存器.	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved					SPLIE18	SPLIE17	SPLIE16
15	14	13	12	11	10	9	8
SPLIE15	SPLIE14	SPLIE13	SPLIE12	SPLIE11	SPLIE10	SPLIE9	SPLIE8
7	6	5	4	3	2	1	0
SPLIE7	SPLIE6	SPLIE5	SPLIE4	SPLIE3	SPLIE2	SPLIE1	SPLIE0

位	描述
[18]	<b>SPLIE18</b> 模块 18 中断使能位 0 = 模块 18 中断禁止. 1 = 模块 18 中断使能.
[17]	<b>SPLIE17</b> 模块 17 中断使能位 0 = 模块 17 中断禁止. 1 = 模块 17 中断使能.
[16]	<b>SPLIE16</b> 模块 16 中断使能位 0 = 模块 16 中断禁止. 1 = 模块 16 中断使能.
[15]	<b>SPLIE15</b> 模块 15 中断使能位 0 = 模块 15 中断禁止. 1 = 模块 15 中断使能.
[14]	<b>SPLIE14</b> 模块 14 中断使能位 0 = 模块 14 中断禁止. 1 = 模块 14 中断使能.
[13]	<b>SPLIE13</b> 模块 13 中断使能位 0 = 模块 13 中断禁止. 1 = 模块 13 中断使能.

位	描述	
[12]	SPLIE12	模块 12 中断使能位 0 = 模块 12 中断禁止. 1 = 模块 12 中断使能.
[11]	SPLIE11	模块 11 中断使能位 0 = 模块 11 中断禁止. 1 = 模块 11 中断使能.
[10]	SPLIE10	模块 10 中断使能位 0 = 模块 10 中断禁止. 1 = 模块 10 中断使能.
[9]	SPLIE9	模块 9 中断使能位 0 = 模块 9 中断禁止. 1 = 模块 9 中断使能.
[8]	SPLIE8	模块 8 中断使能位 0 = 模块 8 中断禁止. 1 = 模块 8 中断使能.
[7]	SPLIE7	模块 7 中断使能位 0 = 模块 7 中断禁止. 1 = 模块 7 中断使能.
[6]	SPLIE6	模块 6 中断使能位 0 = 模块 6 中断禁止. 1 = 模块 6 中断使能.
[5]	SPLIE5	模块 5 中断使能位 0 = 模块 5 中断禁止. 1 = 模块 5 中断使能.
[4]	SPLIE4	模块 4 中断使能位 0 = 模块 4 中断禁止. 1 = 模块 4 中断使能.
[3]	SPLIE3	模块 3 中断使能位 0 = 模块 3 中断禁止. 1 = 模块 3 中断使能.
[2]	SPLIE2	模块 2 中断使能位 0 = 模块 2 中断禁止. 1 = 模块 2 中断使能.
[1]	SPLIE1	模块 1 中断使能位 0 = 模块 1 中断禁止. 1 = 模块 1 中断使能.
[0]	SPLIE0	模块 0 中断使能位 0 = 模块 0 中断禁止. 1 = 模块 0 中断使能.

EADC\_CMP0~3 ADC 结果比较寄存器 0/1/2/3

寄存器	偏移量	R/W	描述	复位值
EADC_CMP0	EADC_BA+0xE0	R/W	ADC 结果比较寄存器 0	0x0000_0000
EADC_CMP1	EADC_BA+0xE4	R/W	ADC 结果比较寄存器 1	0x0000_0000
EADC_CMP2	EADC_BA+0xE8	R/W	ADC 结果比较寄存器 2	0x0000_0000
EADC_CMP3	EADC_BA+0xEC	R/W	ADC 结果比较寄存器 3	0x0000_0000

31	30	29	28	27	26	25	24
Reserved				CMPDAT			
23	22	21	20	19	18	17	16
CMPDAT							
15	14	13	12	11	10	9	8
CMPWEN	Reserved			CMPPMCNT			
7	6	5	4	3	2	1	0
CMPSPL				CMPCOND	ADCMPIE	ADCMPPEN	

位	描述	
[31:28]	Reserved	保留
[27:16]	CMPDAT	<b>比较数据</b> 这里 12 位的值用来和选定采样模块的转换结果比较。用户可以监控外部模拟输入管脚电压的变化，而不用增加软件负担。
[15]	CMPWEN	<b>窗口模式比较使能位</b> 0 = ADCMPF0 (EADC_STATUS2[4]) 当 EADC_CMP0 比较条件满足时置1。ADCMPF2 (EADC_STATUS2[6]) 当 EADC_CMP2 比较条件满足时置1。 1 = ADCMPF0 (EADC_STATUS2[4]) 当 EADC_CMP0 和 EADC_CMP1 的比较条件都满足时置1。ADCMPF2 (EADC_STATUS2[6]) 当 EADC_CMP2 和 EADC_CMP3 比较条件都满足时置1。 <b>注意:</b> 这一位只在EADC_CMP0 和 EADC_CMP2 寄存器里存在。
[14:12]	Reserved	保留
[11:8]	CMPPMCNT	<b>比较满足计数</b> 当选定的 ADC 通道转换结果满足 CMPCOND (EADC_CMPn[2], n=0~3) 定义的比较条件，比较计数器加 1. 如果比较结果没有满足比较条件，比较计数器重置为0. 当计数器计数到 (CMPPMCNT +1) 时，ADCMPPFn (EADC_STATUS2[7:4], n=0~3) 会置1。

位	描述
[7:3]	<b>CMPSPL</b>  比较采样模块选择 00000 = 采样模块 0 转换结果 EADC_DAT0被选择用来比较. 00001 = 采样模块 1 转换结果 EADC_DAT1被选择用来比较. 00010 = 采样模块 2 转换结果 EADC_DAT2被选择用来比较. 00011 = 采样模块 3 转换结果 EADC_DAT3被选择用来比较. 00100 = 采样模块 4 转换结果 EADC_DAT4被选择用来比较. 00101 = 采样模块 5 转换结果 EADC_DAT5被选择用来比较. 00110 = 采样模块 6 转换结果 EADC_DAT6被选择用来比较. 00111 = 采样模块 7 转换结果 EADC_DAT7被选择用来比较. 01000 = 采样模块 8 转换结果 EADC_DAT8被选择用来比较. 01001 = 采样模块 9 转换结果 EADC_DAT9被选择用来比较. 01010 = 采样模块 10 转换结果 EADC_DAT10被选择用来比较. 01011 = 采样模块 11 转换结果 EADC_DAT11被选择用来比较. 01100 = 采样模块 12 转换结果 EADC_DAT12被选择用来比较. 01101 = 采样模块 13 转换结果 EADC_DAT13被选择用来比较. 01110 = 采样模块 14 转换结果 EADC_DAT14被选择用来比较. 01111 = 采样模块 15 转换结果 EADC_DAT15被选择用来比较. 10000 = 采样模块 16 转换结果 EADC_DAT16被选择用来比较. 10001 = 采样模块 17 转换结果 EADC_DAT17被选择用来比较. 10010 = 采样模块 18 转换结果 EADC_DAT18被选择用来比较.
[2]	<b>CMPCOND</b>  比较条件 0= 当 12位ADC转换结果小于12位 CMPDAT (EADC_CMPn [27:16]),比较计数器加1 1=当 12位ADC转换结果大于等于12位 CMPDAT (EADC_CMPn [27:16]),比较计数器加1. 注意: 当比较计数器等于 (CMPMCNT (EADC_CMPn[11:8], n=0~3) +1), CMPF位置1
[1]	<b>ADCMPIE</b>  ADC结果比较中断使能位 0 = 比较功能中断禁止. 1 = 比较功能中断使能. 如果比较功能中断使能且转换结果满足 CMPCOND (EADC_CMPn[2], n=0~3) 和 CMPMCNT (EADC_CMPn[11:8], n=0~3) 的条件 , ADCMPFn (EADC_STATUS2[7:4], n=0~3) 会置1,如果 ADCMPIE设为1, 比较中断请求会产生.
[0]	<b>ADCM PEN</b>  ADC 结果比较使能位 0 = 禁止比较. 1 = 使能比较. 设置此位为 1 , 当转换结果存入寄存器EADC_DAT, 使能CMPDAT (EADC_CMPn[27:16], n=0~3) 与选定通道的转换结果比较.

**EADC\_STATUS0 ADC 状态寄存器 0**

寄存器	偏移量	R/W	描述	复位值
EADC_STATUS0	EADC_BA+0xF0	R	ADC 状态寄存器 0	0x0000_0000

31	30	29	28	27	26	25	24
OV							
23	22	21	20	19	18	17	16
OV							
15	14	13	12	11	10	9	8
VALID							
7	6	5	4	3	2	1	0
VALID							

位	描述	
[31:16]	OV	<b>EADC_DAT0~15 覆盖标志</b> 这一位是ADC模块数据寄存器EADC_DATn. (n=0~15)中 OV位的镜像.
[15:0]	VALID	<b>EADC_DAT0~15 数据有效标志</b> 这一位是ADC模块数据寄存器EADC_DATn. (n=0~15)中 VALID位的镜像

**EADC\_STATUS1 ADC 状态寄存器1**

寄存器	偏移量	R/W	描述	复位值
EADC_STATUS1	EADC_BA+0xF4	R	ADC 状态寄存器 1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved					OV		
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved					VALID		

位	描述	
[31:19]	<b>Reserved</b>	保留.
[18:16]	<b>OV</b>	<b>EADC_DAT16~18 覆盖标志</b> 这一位是ADC模块数据寄存器EADC_DATn. (n=16~18)中 OV位的镜像
[15:3]	<b>Reserved</b>	保留.
[2:0]	<b>VALID</b>	<b>EADC_DAT16~18 数据有效标志</b> 这一位是ADC模块 数据寄存器EADC_DATn. (n=16~18)中VALID位的镜像

## EADC\_STATUS2 ADC 状态寄存器 2

寄存器	偏移量	R/W	描述	复位值
EADC_STATUS2	EADC_BA+0xF8	R/W	ADC 状态寄存器 2	0x0000_0000

31	30	29	28	27	26	25	24	
		Reserved			AOV	AVALID	STOVF	ADOVIF
23	22	21	20	19	18	17	16	
BUSY	Reserved		CHANNEL					
15	14	13	12	11	10	9	8	
ADCMPO3	ADCMPO2	ADCMPO1	ADCMPO0	ADOVIF3	ADOVIF2	ADOVIF1	ADOVIF0	
7	6	5	4	3	2	1	0	
ADCMF3	ADCMF2	ADCMF1	ADCMF0	ADIF3	ADIF2	ADIF1	ADIF0	

位	描述
[31:28]	<b>Reserved</b> 保留.
[27]	<b>AOV</b> 所有采样模块 ADC 结果数据寄存器覆盖标志确认 (只读) n=0~18. 0 = 没有采样模块 ADC 结果数据寄存器覆盖标志 OVn (EADC_DATn[16]) 设为 1. 1 = 有采样模块 ADC 结果数据寄存器覆盖标志 OVn (EADC_DATn[16]) 设为 1. <b>注意:</b> 任意 OVn 标志为1 此位保持为 1.
[26]	<b>AVALID</b> 所有采样模块 ADC 结果数据寄存器数据有效标志确认 (只读) n=0~18. 0 = 没有采样模块 ADC 结果数据寄存器数据有效标志 VALIDn (EADC_DATn[17]) 设为 1. 1 = 有采样模块 ADC 结果数据寄存器数据有效标志 VALIDn (EADC_DATn[17]) 设为 1. <b>注意:</b> 任意 VALIDn 标志为1 此位保持为 1..
[25]	<b>STOVF</b> 所有 ADC 采样模块开始转换覆盖标志确认(只读) n=0~18. 0 = 没有 ADC 采样模块开始转换覆盖标志 SPOVFn (EADC_OVSTS[n]) 设为 1. 1 = 有 ADC 采样模块开始转换覆盖标志 SPOVFn (EADC_OVSTS[n]) 设为 1. <b>注意:</b> 任意 SPOVFn 标志为1此位保持为1.
[24]	<b>ADOVIF</b> 所有 ADC 中断标志覆盖位确认 (只读) n=0~3. 0 = 没有ADINT 终端标识 ADOVIFn (EADC_STATUS2[11:8]) 被覆盖为 1. 1 = 有ADINT 终端标识 ADOVIFn (EADC_STATUS2[11:8]) 被覆盖为 1 <b>注意:</b> 当任意 ADOVIFn 标志为 1, 此位保持为1.
[23]	<b>BUSY</b> 忙/闲 (只读) 0 = EADC在空闲态. 1 = EADC在忙于转换

位	描述	
[22:21]	<b>Reserved</b>	保留.
[20:16]	<b>CHANNEL</b>	<p><b>目前转换通道 (只读)</b>            这里反映当BUSY=1, ADC 目前的转换通道.            此位只读.</p> <p>00H = EADC_CH0.            01H = EADC_CH1.            02H = EADC_CH2.            03H = EADC_CH3.            04H = EADC_CH4.            05H = EADC_CH5.            06H = EADC_CH6.            07H = EADC_CH7.            08H = EADC_CH8.            09H = EADC_CH9.            0AH = EADC_CH10.            0BH = EADC_CH11.            0CH = EADC_CH12.            0DH = EADC_CH13.            0EH = EADC_CH14.            0FH = EADC_CH15.            10H = <math>V_{BG}</math> (默认).            11H = <math>V_{TEMP}</math>.            12H = <math>V_{DD}/4</math>.</p>
[15]	<b>ADCMPO3</b>	<p><b>ADC 比较模块 3 输出状态 (只读)</b>            12 位比较模块3数据CMPDAT3 (EADC_CMP3[27:16]) 用来比较指定转换模块的转换结果.            可以用来监控模拟输入电压状态.            0 = EADC_DAT 的转换结果小于CMPDAT3的设置.            1 = EADC_DAT 的转换结果大于等于CMPDAT3的设置.</p>
[14]	<b>ADCMPO2</b>	<p><b>ADC 比较模块 2 输出状态 (只读)</b>            12 位比较模块2数据CMPDAT2 (EADC_CMP3[27:16]) 用来比较指定转换模块的转换结果.            可以用来监控模拟输入电压状态.            0 = EADC_DAT 的转换结果小于CMPDAT2的设置.            1 = EADC_DAT 的转换结果大于等于CMPDAT2的设置.</p>
[13]	<b>ADCMPO1</b>	<p><b>ADC 比较模块 1 输出状态 (只读)</b>            12 位比较模块1数据CMPDAT1 (EADC_CMP3[27:16]) 用来比较指定转换模块的转换结果.            可以用来监控模拟输入电压状态.            0 = EADC_DAT 的转换结果小于CMPDAT1的设置.            1 = EADC_DAT 的转换结果大于等于CMPDAT1的设置.</p>
[12]	<b>ADCMPO0</b>	<p><b>ADC 比较模块 0 输出状态 (只读)</b>            12 位比较模块0数据CMPDAT0 (EADC_CMP3[27:16]) 用来比较指定转换模块的转换结果.            可以用来监控模拟输入电压状态.            0 = EADC_DAT 的转换结果小于CMPDAT0的设置.            1 = EADC_DAT 的转换结果大于等于CMPDAT0的设置.</p>

位	描述	
[11]	ADOVIF3	<p><b>ADC ADINT3 中断标志覆盖</b>            0 = ADINT3 中断标志没有覆盖到 1.            1 = ADINT3中断标志覆盖到 1.  <b>注意:</b> 此位写1清0.</p>
[10]	ADOVIF2	<p><b>ADC ADINT2 中断标志覆盖</b>            0 = ADINT2 中断标志没有覆盖到 1.            1 = ADINT2中断标志覆盖到 1.  <b>注意:</b> 此位写1清0.</p>
[9]	ADOVIF1	<p><b>ADC ADINT1 中断标志覆盖</b>            0 = ADINT1 中断标志没有覆盖到 1.            1 = ADINT1中断标志覆盖到 1.  <b>注意:</b> 此位写1清0.</p>
[8]	ADOVIF0	<p><b>ADC ADINT0 中断标志覆盖</b>            0 = ADINT0 中断标志没有覆盖到 1.            1 = ADINT0中断标志覆盖到 1.  <b>注意:</b> 此位写1清0.</p>
[7]	ADCMPF3	<p><b>ADC 比较模块 3 标志</b>            当选定的比较模块ADC 转换结果满足 EADC_CMP3 的条件，那么此位置 1.            0 = EADC_DAT的转换结果没有 符合 EADC_CMP3寄存器的设定            1 = EADC_DAT的转换结果 符合 EADC_CMP3寄存器的设定.  <b>注意:</b> 此位写1清0.</p>
[6]	ADCMPF2	<p><b>ADC 比较模块 2 标志</b>            当选定的比较模块ADC 转换结果满足 EADC_CMP2 的条件，那么此位置 1.            0 = EADC_DAT的转换结果没有 符合 EADC_CMP2寄存器的设定            1 = EADC_DAT的转换结果 符合 EADC_CMP2寄存器的设定.  <b>注意:</b> 此位写1清0.</p>
[5]	ADCMPF1	<p><b>ADC 比较模块 1 标志</b>            当选定的比较模块ADC 转换结果满足 EADC_CMP1 的条件，那么此位置 1.            0 = EADC_DAT的转换结果没有 符合 EADC_CMP1寄存器的设定            1 = EADC_DAT的转换结果 符合 EADC_CMP1寄存器的设定.  <b>注意:</b> 此位写1清0.</p>
[4]	ADCMPF0	<p><b>ADC 比较模块 0 标志</b>            当选定的比较模块ADC 转换结果满足 EADC_CMP0 的条件，那么此位置 1.            0 = EADC_DAT的转换结果没有 符合 EADC_CMP0寄存器的设定            1 = EADC_DAT的转换结果 符合 EADC_CMP0寄存器的设定.  <b>注意:</b> 此位写1清0.</p>

位	描述	
[3]	<b>ADIF3</b>	<b>ADC ADINT3中断标志</b> 0 = 没收到 ADINT3 中断脉冲. 1 = 收到 ADINT3 中断脉冲. <b>注意1:</b> 此位写1清0 <b>注意2:</b> 此位反映选择的采样模块 ADC 是否转换完成
[2]	<b>ADIF2</b>	<b>ADC ADINT2中断标志</b> 0 = 没收到 ADINT2 中断脉冲. 1 = 收到 ADINT2 中断脉冲. <b>注意1:</b> 此位写1清0 <b>注意2:</b> 此位反映选择的采样模块 ADC 是否转换完成
[1]	<b>ADIF1</b>	<b>ADC ADINT1中断标志</b> 0 = 没收到 ADINT1 中断脉冲. 1 = 收到 ADINT1 中断脉冲. <b>注意1:</b> 此位写1清0 <b>注意2:</b> 此位反映选择的采样模块 ADC 是否转换完成
[0]	<b>ADIF0</b>	<b>ADC ADINT0中断标志</b> 0 = 没收到 ADINT0 中断脉冲. 1 = 收到 ADINT0 中断脉冲. <b>注意1:</b> 此位写1清0 <b>注意2:</b> 此位反映选择的采样模块 ADC 是否转换完成

**EADC\_STATUS3 ADC 状态寄存器 3**

寄存器	偏移量	R/W	描述	复位值
EADC_STATUS3	EADC_BA+0xFC	R	ADC 状态寄存器 3	0x0000_001F

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved			CURSPL				

位	描述	
[31:5]	Reserved	保留.
[4:0]	CURSPL	<b>ADC 目前采样模块 (只读)</b> 这个寄存器反映目前哪个模块执行ADC. 如果ADC空闲，这里读到 0x1F.

**EADC\_DDAT0~3 ADC 模块双数据寄存器**

寄存器	偏移量	R/W	描述	复位值
EADC_DDAT0	EADC_BA+0x100	R	ADC 采样模块0双数据寄存器	0x0000_0000
EADC_DDAT1	EADC_BA+0x104	R	ADC 采样模块1双数据寄存器	0x0000_0000
EADC_DDAT2	EADC_BA+0x108	R	ADC 采样模块2双数据寄存器	0x0000_0000
EADC_DDAT3	EADC_BA+0x10C	R	ADC 采样模块3双数据寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved						VALID	OV
15	14	13	12	11	10	9	8
RESULT							
7	6	5	4	3	2	1	0
RESULT							

位	描述	
[31:17]	Reserved	保留.
[17]	VALID	<b>有效标志</b> 0 = RESULT (EADC_DDATn[15:0])里的数据无效 1 = RESULT (EADC_DDATn[15:0])里的数据有效. 此位当相应的模块转换完成时置 1， 寄存器当EADC_DDATn. (n=0~3)被读取后由硬件清0.
[16]	OV	<b>覆盖标志</b> 0 = RESULT (EADC_DATn[15:0], n=0~3)里数据未被覆盖 1 = RESULT (EADC_DATn[15:0], n=0~3)里数据被覆盖. 如果RESULT[15:0]的数据在载入新值之前没有被读出， OV会置 1. 读EADC_DDAT 后硬件自动清0.
[15:0]	RESULT	<b>ADC 转换结果</b> 这里是 12位转换结果. 当 DMOF (EADC_CTL[9])为 0, 12-位 ADC 转换结果以无符号数存在RESULT [11:0] , RESULT [15:12]填0. \W当DMOF (EADC_CTL[9]) 为 1, 12位 ADC 转换结果以二进制补码存在 RESULT [11:0] RESULT [15:12]填符号位.

**EADC\_PWRM ADC 电源管理寄存器**

寄存器	偏移量	R/W	描述	复位值
EADC_PWRM	EADC_BA+0x110	R/W	ADC电源管理寄存器	0x0006_E012

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved				LDOSUT			
15	14	13	12	11	10	9	8
LDOSUT							
7	6	5	4	3	2	1	0
Reserved				PWDMOD	PWUCALEN	PWUPRDY	

位	描述
[31:20]	<b>Reserved</b> 保留.
[19:8]	<b>LDOSUT</b> <b>ADC 内部 LDO 启动时间</b> 设置这里控制LDO 启动时间. LDO 需要的最短的启动时间是20us. LDO 启动时间 = $(1/\text{ADC\_CLK}) \times \text{LDOSUT}$ .
[7:4]	<b>Reserved</b> 保留.
[3:2]	<b>PWDMOD</b> <b>ADC掉电模式</b> 这里设置ADC 系统掉电时的模式. 00 = ADC 深度掉电模式. 01 = ADC 掉电. 10 = ADC 等待. 11 = ADC 深度掉电模式. <b>注意:</b> 不同的 PWDMOD 有不同的上下电序列,为了避免错误的ADC序列, 需要在下电和上电的时候保持 PWMOD的值不变.
[1]	<b>PWUCALEN</b> <b>上电校验功能使能控制</b> 0 = 禁止上电校验功能. 1 = 使能上电校验功能. <b>注意:</b> 这一位与 CALSEL (EADC_CALCTL [3])一起起作用, {PWUCALEN, CALSEL } 描述: PWUCALEN 是 0 且 CALSEL 是 0: 不需要校准. PWUCALEN 是 0 且 CALSEL 是 1: 不需要校准. PWUCALEN 是 1 且 CALSEL 是 0: 启动载入校准值. PWUCALEN 是 1 且 CALSEL 是 1: 上电校准
[0]	<b>PWUPRDY</b> <b>ADC 启动序列完成(只读)</b> 0 = ADC没有准备好, 可能处在掉电状态或正在启动序列中. 1 = ADC准备好转换.

**EADC\_CALCTL ADC 校准控制寄存器**

寄存器	偏移量	R/W	描述	复位值
<b>EADC_CALCTL</b>	EADC_BA+0x114	R/W	ADC 校准控制寄存器	0x0000_0008

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved				CALSEL	CALDONE	CALSTART	Reserved

位	描述	
[31:4]	<b>Reserved</b>	保留.
[3]	<b>CALSEL</b>	<b>选择校准功能</b> 0 = 在校准功能有效时载入校准值. 1 = 在校准功能有效时执行校准.
[2]	<b>CALDONE</b>	<b>校准功能完成 (只读)</b> 0 = 正在校准. 1 = 校准完成.
[1]	<b>CALSTART</b>	<b>校准功能启动</b> 0 = 停止校准功能. 1 = 启动校准功能. <b>注意:</b> 此位由软件置1, 重校准完成后由硬件清0 注意: 将 CALSTART (EADC_CALCTL [1]) 设置为1再次开始校准之前, EADCDIV (CKL_CLKDIV0 [23:16]) 必须为0。
[0]	<b>Reserved</b>	保留.

**EADC\_CALDWORD ADC 校准载入字寄存器**

寄存器	偏移量	R/W	描述	复位值
EADC_CALDWRD	EADC_BA+0x118	R/W	ADC校准载入字寄存器	0x0000_00XX

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved	CALWORD						

位	描述	
[31:7]	Reserved	保留.
[6:0]	CALWORD	<p><b>校准字</b>            在载入校准前写入预校准的值            校准完成后读这个寄存器的值.</p> <p><b>注意:</b> 校准包括两个部分“校准”和“载入校准”; 如果配置为“校准”; 这个寄存器存放之前校准完成后的结果; 如果配置为“载入校准”; 在载入之前配置这个寄存器, 载入校准完成后, 载入的校准字会应用到 ADC; 载入校准功能中, 载入的值直到校准完成后才会等于原始的 CALWORD</p>

## 6.38 DAC 数模转换器

### 6.38.1 概述

DAC是一个12位数模转换器，可配置为12位或 8位输出模式，并且可以由PDMA 控制。集成了一个电压输出缓冲器，可以减小输出阻抗并且可以直接驱动外部负载无需再加外部运算放大器。

### 6.38.2 特性

- 输出电压范围: 0~AVDD.
- 支持 12位 或 8位输出模式.
- 轨到轨建立时间 8us.
- 支持 12-位 1 MSPS 电压类型 DAC.
- 参考电压可以选择内部参考电压 (INT\_VREF) 或 VREF管脚.
- DAC 最大更新率 1 MSPS.
- 支持电压输出缓存模式和旁路电压输出缓存模式.
- 支持软件和硬件触发, 包括 Timer0~3, EPWM0, EPWM1, 和外部触发管脚触发 DAC 转换.
- 支持 PDMA 模式.
- 两个DAC支持组模式，同步更新.

### 6.38.3 框图

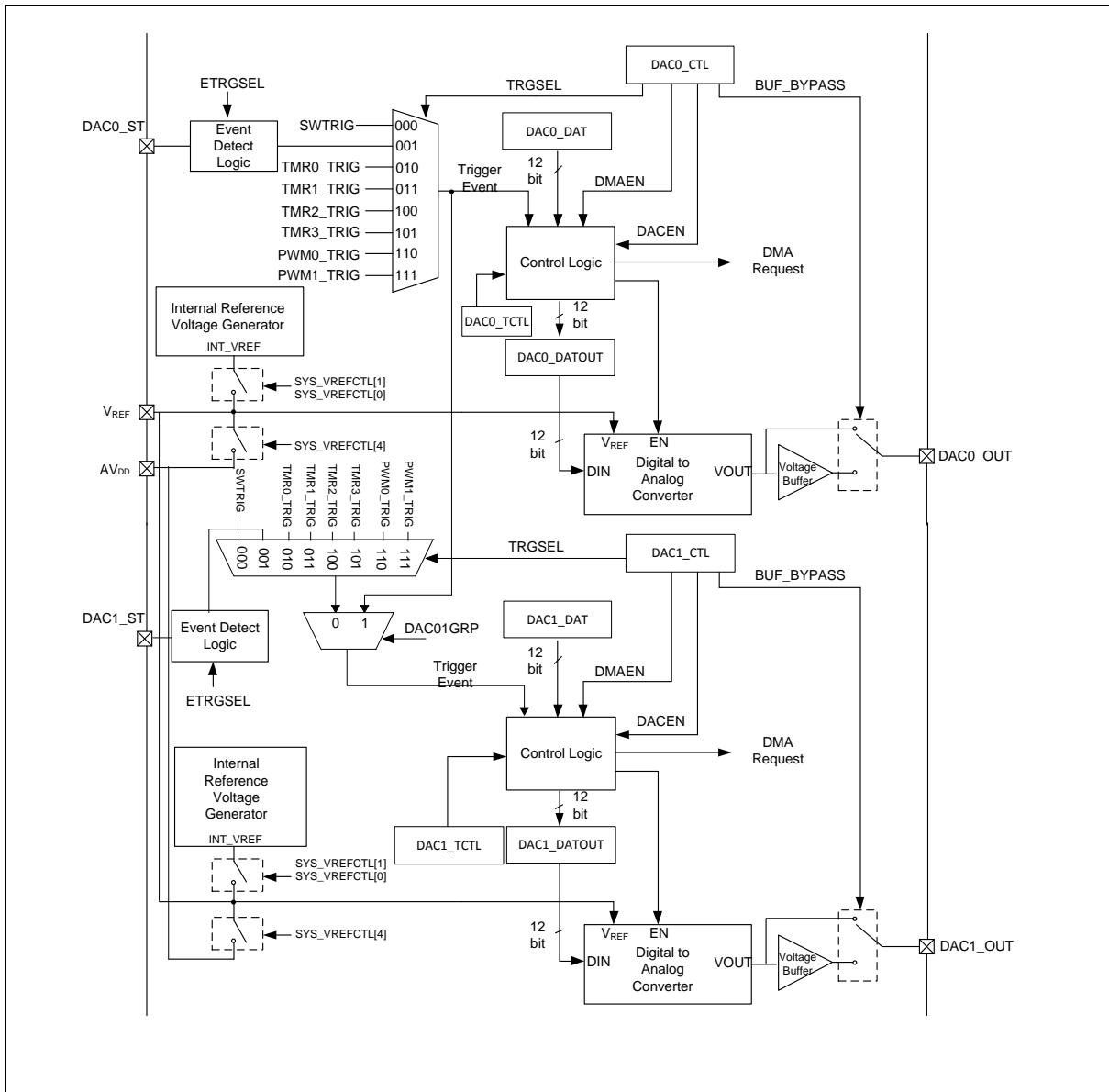


图 6.38-1 数模转换器框图

### 6.38.4 基本配置

#### 6.38.4.1 DAC0 基本配置

- 时钟源配置
  - 在DACKEN (CLK\_APBCLK1[12])使能 DAC0 外设时钟.
- 复位设置
  - 在DACKEN (SYS\_IPRST2[12])复位 DAC0 控制器.
- 管脚配置

组	管脚名称	GPIO	MFP
---	------	------	-----

DAC0	DAC0_OUT	PB.12	MFP1
	DAC0_ST	PA.10	MFP14
		PA.0	MFP15

#### 6.38.4.2 DAC1 基本配置

- 时钟源配置
  - 在DACCEN (CLK\_APBCLK1[12])使能 DAC1 外设时钟.
- 复位设置
  - 在DCCRST (SYS\_IPRST2[12])复位 DAC1 控制器.
- 管脚配置

组	管脚名称	GPIO	MFP
DAC1	DAC1_OUT	PB.13	MFP1
	DAC1_ST	PA.11	MFP14
		PA.1	MFP15

#### 6.38.5 功能描述

##### 6.38.5.1 DAC 输出

DAC 集成集成了一个电压输出缓冲器来降低输出阻抗，无需外加运放可直接驱动负载，输出缓存由 BYPASS (DACn\_CTL[8]) $n=0, 1$ 使能或禁用。DAC最大输出电压受到所选择的参考源限制。

##### 6.38.5.2 DAC 参考电压

DAC 的参考电压与 EADC 共用，由系统管理控制寄存器VREFCTL (SYS\_VREFCTL[4:0]) 配置。可配置为外部参考电压 ( $V_{REF}$ ) 或内部参考电压 (INT\_VREF)。

##### 6.38.5.3 DAC 数据格式

支持数据左对齐或右对齐模式。需要设置下列寄存器：

- 12-位左对齐：用户把数据存入 DACn\_DAT[15:4] 位. DACn\_DAT[31:16] 和 DACn\_DAT[3:0] 在 DAC 转换中被忽略.
- 12-位右对齐：用户把数据存入 DACn\_DAT[11:0] 位, DACn\_DAT[31:12]在 DAC 转换中被忽略

当 DAC 工作在 8-位 模式,对齐设定无效. 要设定 8-位模式, 设置 BWSEL(DACn\_CTL[15:14])为 01. 否则保持BWSEL 为 00.

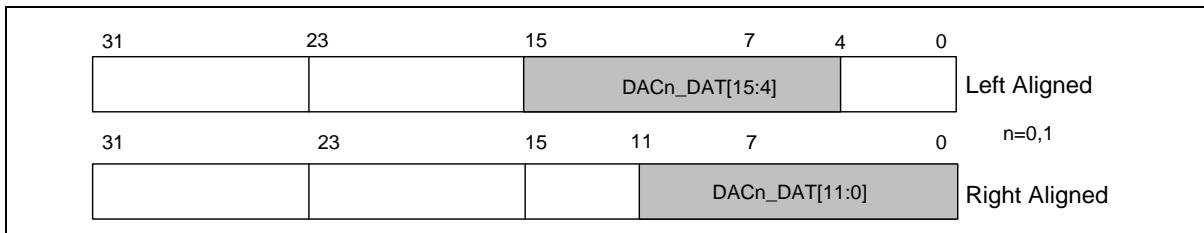


图 6.38-2 数据保持寄存器格式

#### 6.38.5.4 DAC 转换

图 6.38-3 展示软件启动 DAC 转换过程。写数据到寄存器 DACn\_DAT，数据会载入输出寄存器 DACn\_DATOUT，DAC 在一个PCLK (APB clock) 时钟后开始转换。图 6.38-4 展示硬件触发开始DAC (外部管脚 DACn\_ST, 定时器触发事件 或 EPWM 触发事件). DACn\_DAT 寄存器里写入的数据在一个 PCLK (APB clock)后，传输到输出缓存DACn\_DATOUT.

输出寄存器 DACn\_DATOUT 载入 DACn\_DAT 的内容,转换建立时间后变为有效。输入从最低 (0x000) 到最高 (0xFFFF)需要8us 。相邻的数值转换建立时间是1us。 DAC 提供一个 10-位计数器计算转换时间。连续操作时，需要往寄存器 SETTLET (DACn\_TCTL[9:0])配置 DAC 转换时间。这个值必须大于 DAC电器特性表列出的 DAC 转换建立时间。例如，当 DAC APB 时钟速度 80MHz 且 DAC 转换建立时间是8us，选择的 SETTLET 值必须大于 0x280。转换开始后，转换完成标志 FINISH (DACn\_STATUS[0]) 会被硬件清 0，当计数器计数到 SETTLET时置1

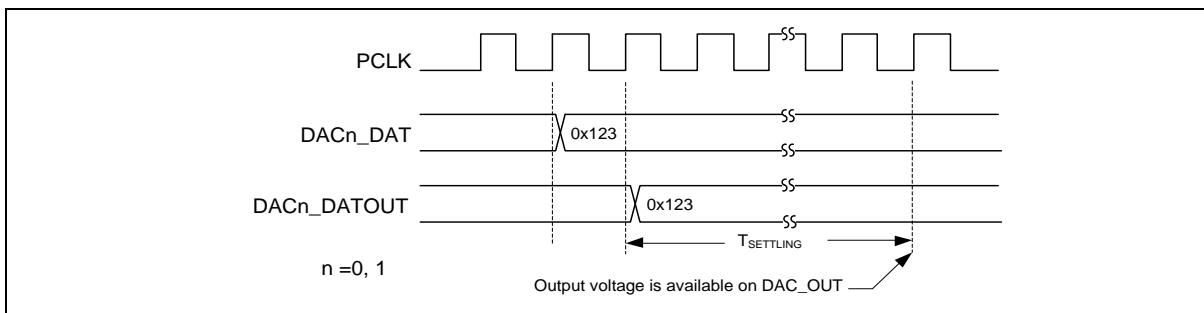


图 6.38-3 DAC 转换由软件写触发

#### 6.38.5.5 DAC输出电压

DAC输出电压范围是0到V<sub>REF</sub>，管脚输出的模拟电压由下列公式计算:

$$\text{DACOUT} = V_{\text{REF}} * \frac{\text{DAT}_n\text{OUT}[11:0]}{4096}, n=0,1$$

#### 6.38.5.6 DAC触发选择

DAC 转换可以由写DACn\_DAT触发，软件触发或硬件触发。TRGEN (DACn\_CTL[4]) = 0时，转换由写寄存器DACn\_DAT开始，TRGEN (DACn\_CTL[4])= 1时，转换由外部 DACn\_ST 管脚,定时器，或EPWM 触发。若选择软件触发，每次写 SWTRG (DACn\_SWTRG[0])为 1触发一次转换。当 DACn\_DATOUT 载入DACDAT 内容后SWTRG 由硬件清0。TRGSEL (DACn\_CTL[7:5]) 选择8种触发转换方式。

当 DAC 检测到触发事件输入, 之前写到 DACDAT 的数据传输到 DACn\_DATOUT[11:0]， DAC 转换在 1 个 PCLK (APB 时钟) 后开始。

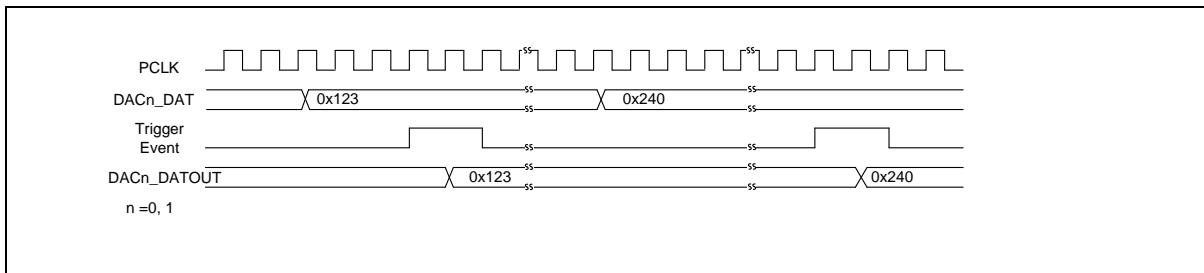


图 6.38-4 DAC 转换由硬件触发事件开始

#### 6.38.5.7 DAC组模式

GRPEN (DAC0\_CTL[16]) 可配置DAC0和DAC1为组模式，硬件保证两个 DAC 同时更新。组模式下 DAC1 的行为由 DAC0\_CTL 和 DAC0\_TCTL 控制，DAC1\_CTL 和 DAC1\_TCTL 不起作用。图 6.38-5 展示组模式与普通模式的比较。

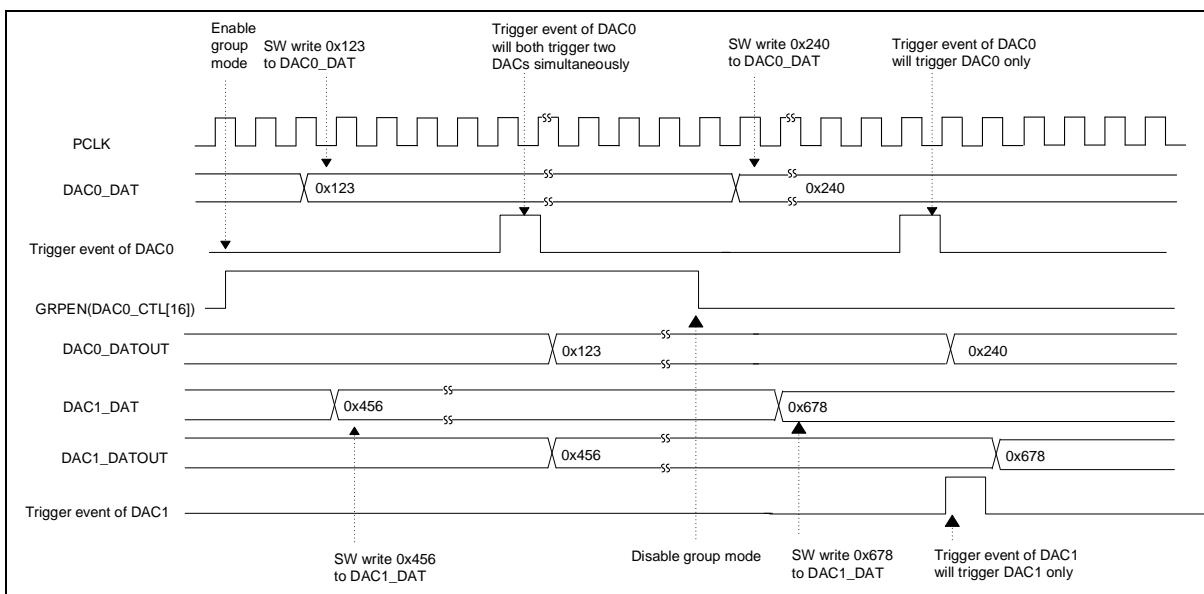


图 6.38-5 DAC0 和 DAC1 组模式和非组模式更新示例

#### 6.38.5.8 DMA操作

DMAEN (DACn\_CTL[2]) =1使能DMA功能，DAC DMA 请求在硬件触发时产生。DACn\_DAT 的内容传到 DACn\_DATOUT[11:0]，DAC 在1个PCLK (APB 时钟)后开始转换。写到DACn\_DAT的新数据在下次触发时转换。图 6.38-6 展示 DAC PDMA 下溢的情形，当第二个 DMA 请求在第一笔转换完成前到来，新的PDMA 请求不会执行，DMA 下溢标志 DMAUDR (DACn\_STATUS[1])置 1 报告错误情况。DMA 数据传输禁止且不触发新的 DMA 请求，DAC 继续转换最后的数据。如果相应的DMAURien (DACn\_CTL[3]) 使能就会产生中断。用户需要及时调整触发的频率或定时器、EPWM 等，然后重新开始 DAC 转换。

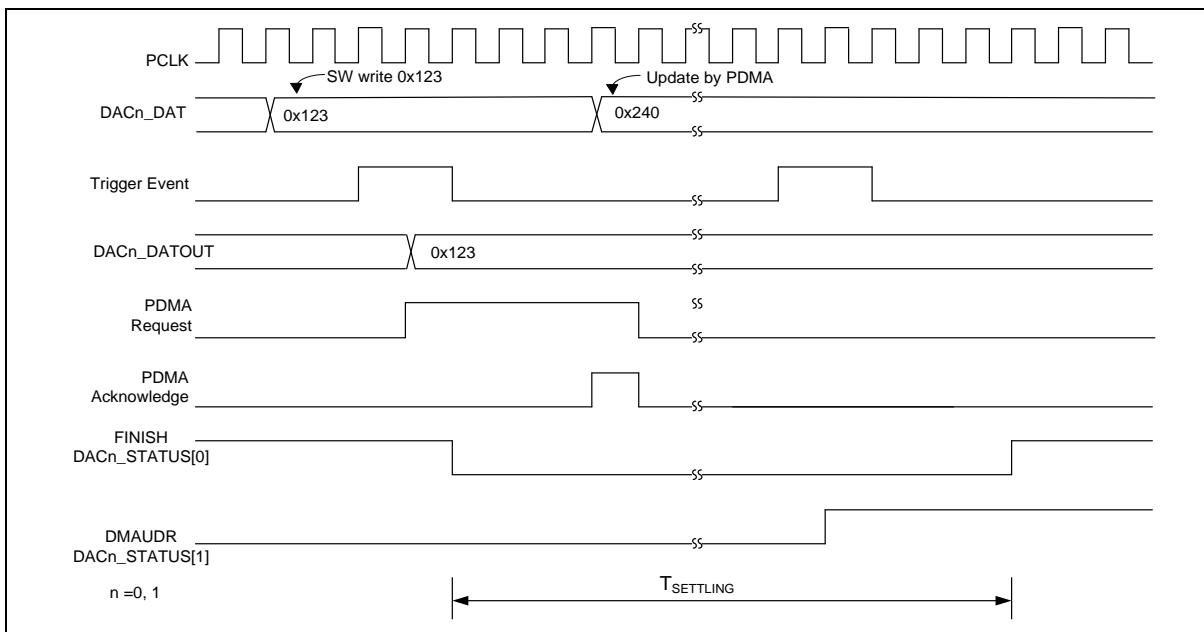


图 6.38-6 DAC PDMA 下溢条件示例

DMA 请求也可以由软件使能产生，用户设置 DMAEN (DACn\_CTL[2]) 为 1 以及 TRGEN (DACn\_CTL[4]) 为 0, DMA 请求周期产生，周期依据 SETTLET (DACn\_TCTL[9:0]) 的值. DAC输出周期性更新。当用户清 DMAEN (DACn\_CTL[2]) 为 0, DAC 控制器会停止产生新的 PDMA 传输请求。图 6.38-7 提供软件PDMA模式 DAC 连续转换的示例。

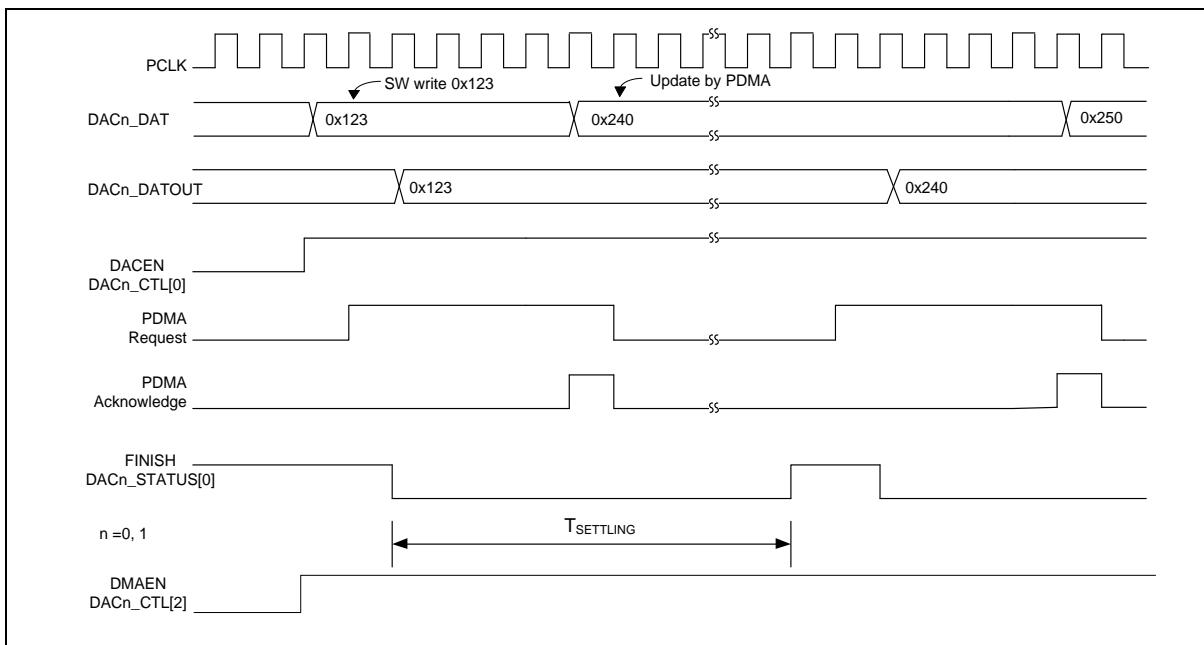


图 6.38-7 软件 PDMA 模式 DAC 连续转换

### 6.38.5.9 中断源

DAC 控制器有两个中断源，一个是DAC 数据转换完成中断，另一个是 DMA下溢中断，如图 6.38-8. DAC 转换完成, FINISH (DACn\_STATUS[0])位会置 1，如果DaciEN (DACn\_CTL[1])=1则产生中断。如果新的 DMA 触发事件在DAC 数据转换完成之前产生, DMA 下溢标志 DMAUDR (DACn\_STATUS[1])

置1，若 DMAURIEN (DACn\_CTL[3])=1使能了中断，会申请中断。

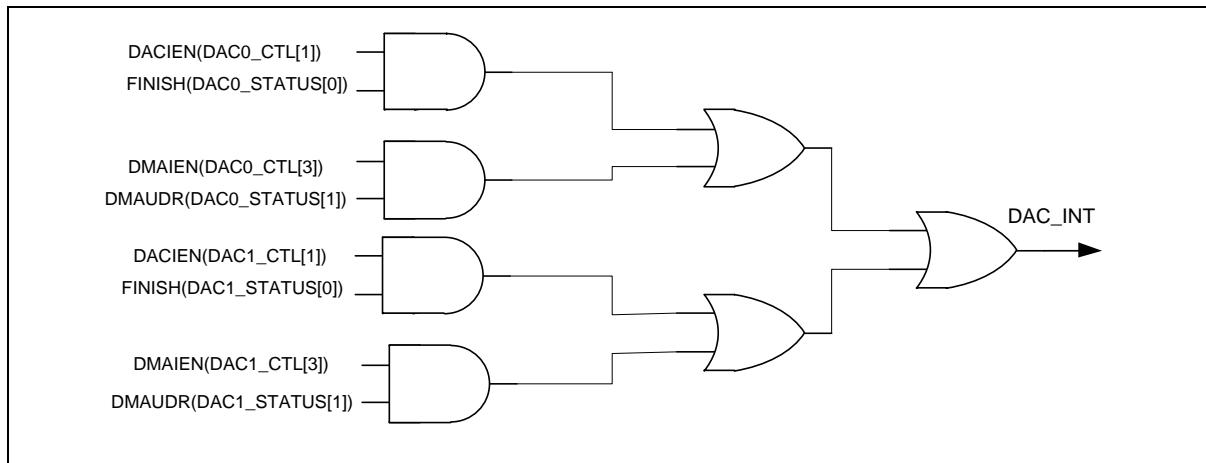


图 6.38-8 DAC 中断源

### 6.38.6 寄存器映射

R: 只读, W: 只写, R/W: 可读可写

寄存器	偏移量	R/W	描述	复位值
<b>DAC 基地址:</b>				
<b>DAC_BA = 0x4004_7000</b>				
<b>DAC0_CTL</b>	DAC_BA+0x00	R/W	DAC0 控制寄存器	0x0000_0000
<b>DAC0_SWTRG</b>	DAC_BA+0x04	R/W	DAC0 软件触发控制寄存器	0x0000_0000
<b>DAC0_DAT</b>	DAC_BA+0x08	R/W	DAC0 数据保持寄存器	0x0000_0000
<b>DAC0_DATOUT</b>	DAC_BA+0x0C	R	DAC0 数据输出寄存器	0x0000_0000
<b>DAC0_STATUS</b>	DAC_BA+0x10	R/W	DAC0 状态寄存器	0x0000_0000
<b>DAC0_TCTL</b>	DAC_BA+0x14	R/W	DAC0 时序控制寄存器	0x0000_0000
<b>DAC1_CTL</b>	DAC_BA+0x40	R/W	DAC1 控制寄存器	0x0000_0000
<b>DAC1_SWTRG</b>	DAC_BA+0x44	R/W	DAC1 软件触发控制寄存器	0x0000_0000
<b>DAC1_DAT</b>	DAC_BA+0x48	R/W	DAC1 数据保持寄存器	0x0000_0000
<b>DAC1_DATOUT</b>	DAC_BA+0x4C	R	DAC1 数据输出寄存器	0x0000_0000
<b>DAC1_STATUS</b>	DAC_BA+0x50	R/W	DAC1 状态寄存器	0x0000_0000
<b>DAC1_TCTL</b>	DAC_BA+0x54	R/W	DAC1 时序控制寄存器	0x0000_0000

### 6.38.7 寄存器描述

#### DAC0\_CTL DAC0 控制寄存器

寄存器	偏移量	R/W	描述	复位值
DAC0_CTL	DAC_BA+0x00	R/W	DAC0 控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
BWSEL		ETRGSEL		Reserved	LALIGN	Reserved	BYPASS
7	6	5	4	3	2	1	0
TRGSEL			TRGEN	DMAURIEN	DMAEN	DACIEN	DACEN

位	描述	
[31:17]	Reserved	保留.
[16]	GRPEN	<b>DAC 组模式使能位</b> 0 = DAC0 和 DAC1 组模式禁止. 1 = DAC0 和 DAC1 组模式使能.
[15:14]	BWSEL	<b>DAC 数据位宽选择</b> 00 = 数据是 12 位. 01 = 数据是 8 位. 其他 = 保留.
[13:12]	ETRGSEL	<b>外部管脚触发选择</b> 00 = 低电平触发. 01 = 高电平触发. 10 = 下降沿触发 11 = 上升沿触发
[11]	Reserved	保留.
[10]	LALIGN	<b>DAC数据左对齐使能控制</b> 0 = 右对齐. 1 = 左对齐
[9]	Reserved	保留.
[8]	BYPASS	<b>输出缓存选择</b> 0 = 输出电压缓存使能. 1 = 输出电压缓存禁止.
[7:5]	TRGSEL	<b>触发源选择</b>

		000 = 软件触发. 001 = 外部管脚 DAC0_ST触发 010 = 定时器 0 触发. 011 = 定时器1 触发. 100 =定时器 2 触发. 101 =定时器 3 触发. 110 = EPWM0 触发. 111 = EPWM1 触发.
[4]	<b>TRGEN</b>	<b>触发模式使能位</b> 0 = DAC 事件触发模式禁止. 1 = DAC 事件触发模式使能.
[3]	<b>DMAURIEN</b>	<b>DMA 下溢中断使能位</b> 0 = DMA 下溢中断禁止. 1 = DMA 下溢中断使能.
[2]	<b>DMAEN</b>	<b>DMA 模式使能位</b> 0 = DMA 模式禁止. 1 = DMA 模式使能.
[1]	<b>DACIEN</b>	<b>DAC 中断使能位</b> 0 = 中断禁止. 1 =中断使能.
[0]	<b>DACEN</b>	<b>DAC 使能位</b> 0 = DAC 禁止. 1 = DAC 使能.

**DAC0\_SWTRG DAC0 软件触发控制寄存器**

寄存器	偏移量	R/W	描述	复位值
DAC0_SWTRG	DAC_BA+0x04	R/W	DAC0 软件触发控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved							SWTRG

位	描述	
[31:1]	Reserved	保留
[0]	SWTRG	<p>软件触发 0 = 禁止软件触发. 1 = 使能软件触发.</p> <p>用户向这一位写1产生一个一次性脉冲，然后由硬件清0；读此位总为0</p>

**DAC0\_DAT DAC0 数据保持寄存器**

寄存器	偏移量	R/W	描述	复位值
DAC0_DAT	DAC_BA+0x08	R/W	DAC0数据保持寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
DACDAT							
7	6	5	4	3	2	1	0
DACDAT							

位	描述	
[31:16]	Reserved	保留.
[15:0]	DACDAT	<p><b>DAC 12-位保持数据</b></p> <p>写DAC 输出的12-位数据. 不使用的位 (左对齐DAC_DAT[3:0], 右对齐DAC_DAT[15:12]) 会被 DAC控制器硬件忽略.</p> <p>12 位左对齐:数据填到 DAC_DAT[15:4].</p> <p>12 位右对齐:数据填到 DAC_DAT[11:0].</p>

**DAC0 DATOUT DAC0数据输出寄存器**

寄存器	偏移量	R/W	描述	复位值
DAC0_DATOUT	DAC_BA+0x0C	R	DAC0 数据输出寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved				DATOUT			
7	6	5	4	3	2	1	0
DATOUT							

位	描述	
[31:12]	Reserved	保留
[11:0]	DATOUT	<p><b>DAC 12-位 输出数据</b></p> <p>这里是目前数据用来 DAC 输出转换。 从 DAC_DAT 寄存器搬来，用户不能直接改写</p>

## DAC0 STATUS DAC0 状态寄存器

寄存器	偏移量	R/W	描述	复位值
DAC0_STATUS	DAC_BA+0x10	R/W	DAC0 状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved						DMAUDR	FINISH

位	描述
[31:9]	<b>Reserved</b>
[8]	<b>BUSY</b> DAC 忙标志 (只读) 0 = DAC准备好下一次转换. 1 = DAC正忙于转换.
[7:2]	<b>Reserved</b>
[1]	<b>DMAUDR</b> DMA 下溢中断标志 0 = 没有 DMA 下溢错误状况发生. 1 = 有 DMA 下溢错误状况发生. 此位写1清0.
[0]	<b>FINISH</b> DAC 转换完成中断 0 = DAC正在转换. 1 = DAC转换完成. 转换计数器计到 SETTLET时此位置1. 当 DAC开始新的转换时清0. 用户写1可以清0.

**DAC0\_TCTL DAC0 时序控制寄存器**

寄存器	偏移量	R/W	描述	复位值
DAC0_TCTL	DAC_BA+0x14	R/W	DAC0 时序控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved						SETTLET	
7	6	5	4	3	2	1	0
SETTLET							

位	描述	
[31:10]	Reserved	保留.
[9:0]	SETTLET	<p><b>DAC输出建立时间</b>            填的值必须满足DAC 转换建立时间，单位是 PCLK (APB时钟)            例如, DAC 时钟是 80MHz 且 DAC 转换建立时间是 1 us, SETTLET值必须大于 0x50.  <math>SELTTLET = DAC\text{控制器时钟频率} \times \text{建立时间}</math></p>

**DAC1\_CTL DAC1 控制寄存器**

寄存器	偏移量	R/W	描述	复位值
DAC1_CTL	DAC_BA+0x40	R/W	DAC1 控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
BWSEL		ETRGSEL		Reserved	LALIGN	Reserved	BYPASS
7	6	5	4	3	2	1	0
TRGSEL			TRGEN	DMAURIEN	DMAEN	DACIEN	DACEN

位	描述	
[31:16]	Reserved	保留
[15:14]	BWSEL	<b>DAC 数据位宽选择</b> 00 = 数据是 12 位. 01 = 数据是 8 位. 其他 = 保留.
[13:12]	ETRGSEL	<b>外部管脚触发选择</b> 00 = 低电平触发. 01 = 高电平触发. 10 = 下降沿触发 11 = 上升沿触发
[11]	Reserved	保留.
[10]	LALIGN	<b>DAC数据左对齐使能控制</b> 0 = 右对齐. 1 = 左对齐
[9]	Reserved	保留.
[8]	BYPASS	<b>输出缓存选择</b> 0 = 输出电压缓存使能. 1 = 输出电压缓存禁止.
[7:5]	TRGSEL	<b>触发源选择</b> 000 = 软件触发. 001 = 外部管脚 DAC0_ST 触发 010 = 定时器 0 触发. 011 = 定时器 1 触发. 100 = 定时器 2 触发.

		101 = 定时器 3 触发. 110 = EPWM0 触发. 111 = EPWM1 触发.
[4]	<b>TRGEN</b>	<b>触发模式使能位</b> 0 = DAC 事件触发模式禁止. 1 = DAC 事件触发模式使能.
[3]	<b>DMAURIEN</b>	<b>DMA 下溢中断使能位</b> 0 = DMA 下溢中断禁止. 1 = DMA 下溢中断使能.
[2]	<b>DMAEN</b>	<b>DMA 模式使能位</b> 0 = DMA 模式禁止. 1 = DMA 模式使能.
[1]	<b>DACIEN</b>	<b>DAC 中断使能位</b> 0 = 中断禁止. 1 = 中断使能.
[0]	<b>DACEN</b>	<b>DAC 使能位</b> 0 = DAC 禁止. 1 = DAC 使能.

**DAC1\_SWTRG DAC1 软件触发控制寄存器**

寄存器	偏移量	R/W	描述	复位值
DAC1_SWTRG	DAC_BA+0x44	R/W	DAC1 软件触发控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved							SWTRG

位	描述	
[31:1]	Reserved	保留
[0]	SWTRG	<p>软件触发 0 = 禁止软件触发. 1 = 使能软件触发.</p> <p>用户向这一位写1产生一个一次性脉冲，然后由硬件清0；读此位总为0</p>

**DAC1\_DAT DAC1 数据保持寄存器**

寄存器	偏移量	R/W	描述	复位值
DAC1_DAT	DAC_BA+0x48	R/W	DAC1数据保持寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
DACDAT							
7	6	5	4	3	2	1	0
DACDAT							

位	描述	
[31:16]	Reserved	保留.
[15:0]	DACDAT	<p><b>DAC 12-位保持数据</b></p> <p>写DAC 输出的12-位数据. 不使用的位 (左对齐DAC_DAT[3:0], 右对齐DAC_DAT[15:12]) 会被 DAC控制器硬件忽略.</p> <p>12 位左对齐:数据填到 DAC_DAT[15:4].</p> <p>12 位右对齐:数据填到 DAC_DAT[11:0].</p>

**DAC1 DATOUT DAC1 数据输出寄存器**

寄存器	偏移量	R/W	描述	复位值
DAC1_DATOUT	DAC_BA+0x4C	R	DAC1 数据输出寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved				DATOUT			
7	6	5	4	3	2	1	0
DATOUT							

位	描述	
[31:12]	Reserved	保留
[11:0]	DATOUT	<b>DAC 12-位 输出数据</b> 这里是目前数据用来 DAC 输出转换。 从 DAC_DAT 寄存器搬来，用户不能直接改写

## DAC1 STATUS DAC1 状态寄存器

寄存器	偏移量	R/W	描述	复位值
DAC1_STATUS	DAC_BA+0x50	R/W	DAC1 状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved						DMAUDR	FINISH

位	描述
[31:9]	<b>Reserved</b>
[8]	<b>BUSY</b> <b>DAC 忙标志 (只读)</b> 0 = DAC准备好下一次转换. 1 = DAC正忙于转换.
[7:2]	<b>Reserved</b>
[1]	<b>DMAUDR</b> <b>DMA 下溢中断标志</b> 0 = 没有 DMA 下溢. 1 = 有 DMA 下溢错误状况发生. 此位写1清0.
[0]	<b>FINISH</b> <b>DAC 转换完成中断</b> 0 = DAC正在转换. 1 = DAC转换完成. 转换计数器计到 SETTLET时此位置1. 当 DAC开始新的转换时清0. 用户写1可以清0.

**DAC1\_TCTL DAC1 时序控制寄存器**

寄存器	偏移量	R/W	描述	复位值
DAC1_TCTL	DAC_BA+0x54	R/W	DAC1 时序控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved						SETTLET	
7	6	5	4	3	2	1	0
SETTLET							

位	描述	
[31:10]	Reserved	保留.
[9:0]	SETTLET	<p><b>DAC输出建立时间</b>            填的值必须满足DAC 转换建立时间，单位是 PCLK (APB时钟)            例如, DAC 时钟是 80MHz 且 DAC 转换建立时间是 1 us, SETTLET值必须大于 0x50.  <math>SELTTLET = DAC\text{控制器时钟频率} \times \text{建立时间}</math></p>

## 6.39 ACMP 模拟比较器控制器

### 6.39.1 概述

芯片提供两个比较器。当正输入大于负输入时比较器输出逻辑1。每个比较器在输出翻转时都可以配置产生中断。

### 6.39.2 特性

- 输入电压范围: 0 ~  $V_{DDA}$  ( $AV_{DD}$  管脚电压)
- 两组轨到轨模拟比较器
- 支持迟滞回差功能
  - 迟滞回差电压: 0mV, 10mV, 20mV 和 30mV
- 支持唤醒功能
- 支持传输延时和低功耗模式
- 可配置正输入和负输入引脚
- ACMP0 支持:
  - 可选4个 I/O 管脚做正输入源:
    - ◆ ACMP0\_P0, ACMP0\_P1, ACMP0\_P2, 或 ACMP0\_P3
  - 4 个负端输入源:
    - ◆ ACMP0\_N
    - ◆ 比较器参考电压(CRV)
    - ◆ 内部带隙电压(VBG)
    - ◆ DAC0 输出(DAC0\_OUT)
- ACMP1 支持:
  - 可选4个 I/O 管脚做正输入源:
    - ◆ ACMP1\_P0, ACMP1\_P1, ACMP1\_P2, 或 ACMP1\_P3
  - 4 个负端输入源:
    - ◆ ACMP1\_N
    - ◆ 比较器参考电压(CRV)
    - ◆ 内部带隙电压(VBG)
    - ◆ DAC0 输出(DAC0\_OUT)
- 共享一个 ACMP 中断向量
- 比较结果改变时产生中断 (中断条件可配置)
- 支持触发 EPWM 急停事件
- 支持窗口比较模式和窗口锁存模式

### 6.39.3 框图

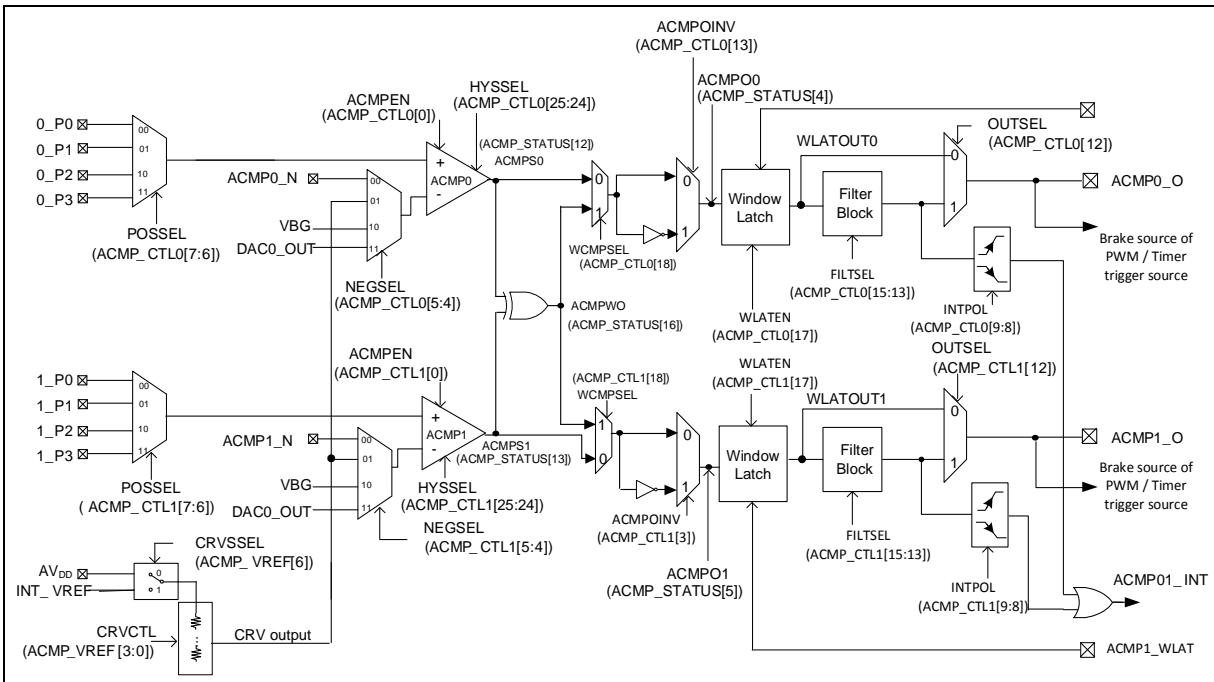


图 6.39-1 模拟比较器框图

### 6.39.4 基本配置

#### 6.39.4.1 ACMP0 基本配置

- 时钟源配置
  - 在ACMP01CKEN (CLK\_APBCLK0[7])使能 ACMP0外设时钟.
- 复位配置
  - 在ACMP01RST (SYS\_IPRST1[7]) 复位 ACMP0 控制器.
- 管脚配置

组	管脚名称	GPIO	MFP
ACMP0	ACMP0_N	PB.3	MFP1
	ACMP0_O	PC.1, PC.12	MFP14
		PB.7	MFP15
	ACMP0_P0	PA.11	MFP1
	ACMP0_P1	PB.2	MFP1
	ACMP0_P2	PB.12	MFP1
	ACMP0_P3	PB.13	MFP1
	ACMP0_WLAT	PA.7	MFP13

#### 6.39.4.2 ACMP1 基本配置

- 时钟源配置

- 在ACMP01CKEN (CLK\_APBCLK0[7]) 使能 ACMP1 外设时钟.
- 复位配置
- 在ACMP01RST (SYS\_IPRST1[7]) 复位 ACMP1 控制器.
- 管脚配置

组	管脚名称	GPIO	MFP
ACMP1	ACMP1_N	PB.5	MFP1
	ACMP1_O	PC.0, PC.11	MFP14
		PB.6	MFP15
	ACMP1_P0	PA.10	MFP1
	ACMP1_P1	PB.4	MFP1
	ACMP1_P2	PB.12	MFP1
	ACMP1_P3	PB.13	MFP1
	ACMP1_WLAT	PA.6	MFP13

## 6.39.5 功能描述

### 6.39.5.1 迟滞回差功能

模拟比较器输出支持迟滞回差功能，保证输出的稳定，参考 图 6.39-2. 如果比较器输出 0，直到正端电压比负端电压高于阈值电压时输出才会变为1。同样的，如果比较器输出为1，直到正端电压比负端电压低，且压差大于阈值电压时才会变0。

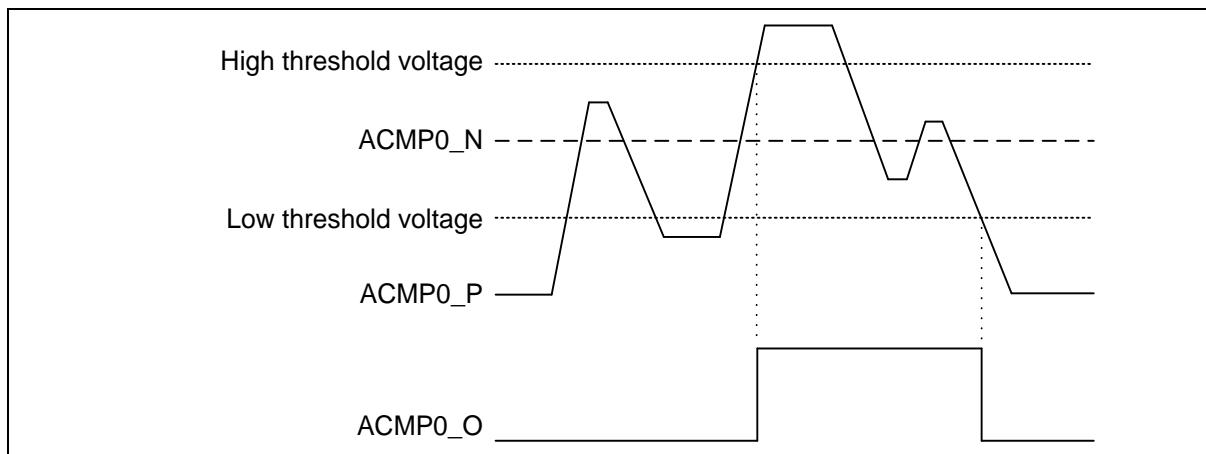


图 6.39-2 ACMP0 比较器迟滞回差功能

### 6.39.5.2 窗口锁存模式

图 6.39-3 未比较器工作在窗口锁存模式。设置WLATEN (ACMP\_CTL0/1[17]) 为 1使能窗口锁存模式。当使能窗口锁存功能时，ACMP0/1\_WLAT管脚用来控制输出WLATOUT0/1。当 ACMP0/1\_WLAT管脚是高, ACMP0/1 输出到WLATOUT0/1; 当 ACMP0/1\_WLAT管脚是低, WLATOUT0/1保持WLATOUT0/1上一次的状态。

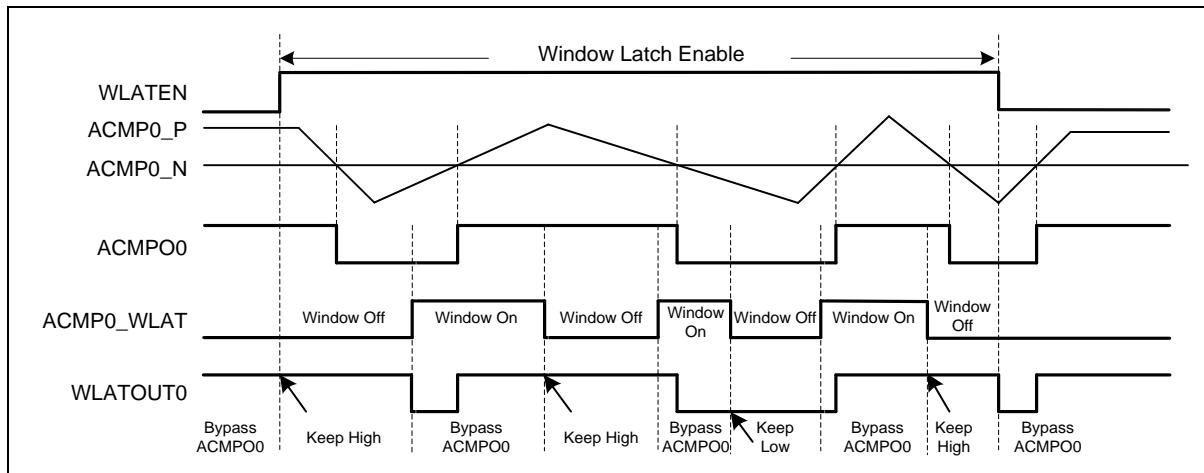


图 6.39-3 窗口锁存模式

### 6.39.5.3 滤波功能

为了避免比较器输出不稳定的状态，模拟比较器提供了滤波功能。

设定 FILTSEL (ACMP\_CTL0[15:13], ACMP\_CTL1[15:13]), 模拟比较器的输出会由连续的 PCLKs时钟采样。采样间隔越长，比较器的输出越稳定。但是比较器的敏感性也越低。

图 6.39-4 为ACMP0的FILTSEL = 3 (4 PCLK)时的例子。所有的结果在输出到ACMPO0前必须保持4个PCLK 时钟。如果比较结果短于4个PCLK,会被滤掉.

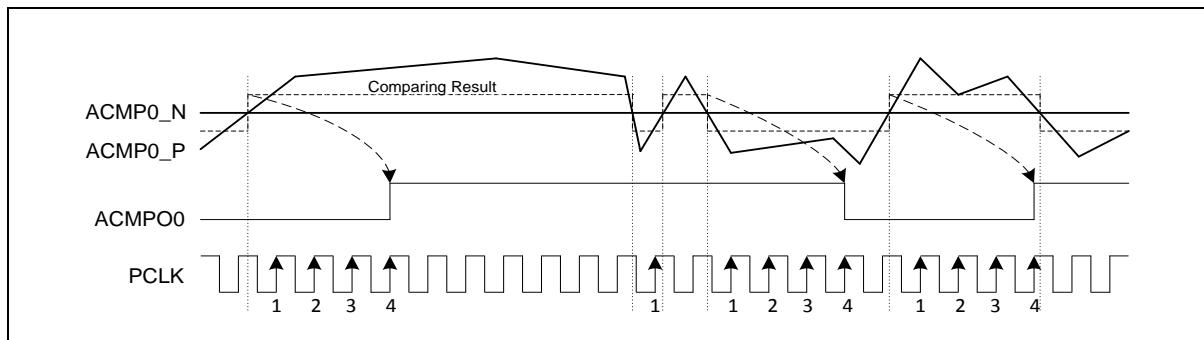


图 6.39-4 滤波功能示例

### 6.39.5.4 中断源

ACMP0 和 ACMP1 的输出值分别反映在 ACMPO0 (ACMP\_STATUS[4]) 和 ACMPO1 (ACMP\_STATUS[5]), 该值是被窗口锁存和滤波处理后的值，可以去触发中断。参考 图 6.39-5, 如果寄存器 ACMP\_CTL0/1 的 ACMPIE 位设为 1, 将使能中断。如果 ACMPO0/1 输出改变符合INTPOL (ACMP\_CTL0/1[9:8]) 配置的，中断标志 ACMPIF0 (ACMP\_STATUS[0]) 或 ACMPIF1 (ACMP\_STATUS[1])置 1. 中断标志写1清0。

如果ACMP中断使能并且ACMP唤醒功能使能，系统被ACMP唤醒后，WKIF (ACMP\_STATUS[8], ACMP\_STATUS[9]) 会置1并且产生中断申请。

图 6.39-5 展示ACMPIF 或 WKIF的中断，由 ACMPIE使能或禁止。

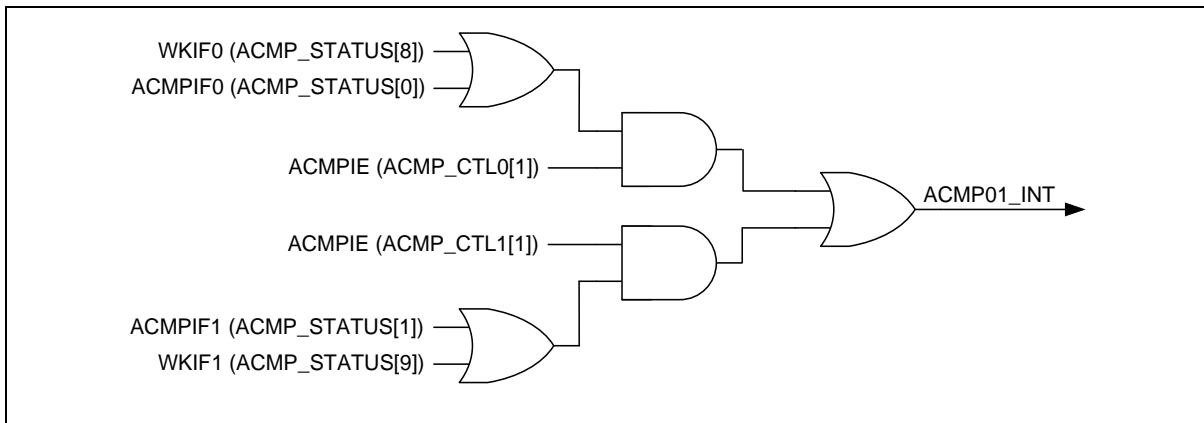


图 6.39-5 比较器控制器中断

#### 6.39.5.5 比较器参考电压(CRV)

CRV 模块包含电阻梯和模拟开关。设置 CRVCTL (ACMP\_VREF[3:0]) 选择 CRV 输出电压。配置 NEGSEL (ACMP\_CTL0[5:4], ACMP\_CTL1[5:4]) 可将此电压连接到比较器负输入端。图 6.39-6 展示模拟参考电压的框图。

NEGSEL (ACMP\_CTL0[5:4], ACMP\_CTL1[5:4]) 若没有选择 CRV 模块，电阻梯会被硬件关闭以降低功耗。电阻梯的参考电压可以通过 SYS\_VREFCTL 选择 AVDD 管脚或内部参考电压。

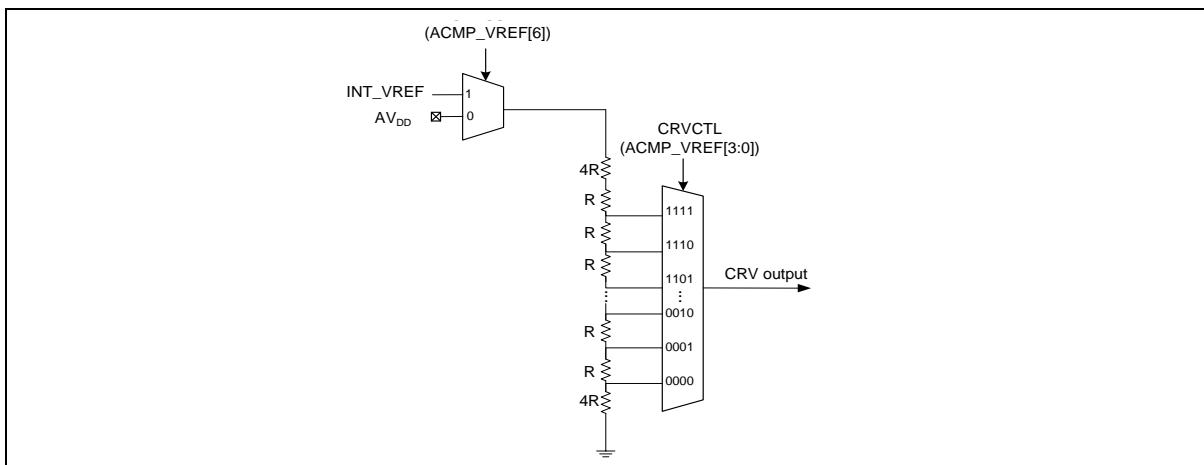


图 6.39-6 比较器参考电压框图

#### 6.39.5.6 窗口比较模式

设置 WCMPSL (ACMP\_CTL0/1[18]) 为 1 使能窗口比较器模式，可监控某个电压是否在特定范围内。用户可以连接特定的模拟电压源到两个比较器的正输入端或是负输入端。监测电压范围的上、下限取决于两个比较器的另外的输入端。如果一个比较器的输出为低，而另一个比较器的输出为高这就意味着一个比较器监测的是上限，另一个比较器监测的是下限。用户可以通过 ACMPWO (ACMP\_STATUS[16]) 直接监控模拟输入电压，如果 ACMPWO 是高，意味着输入电压在上限和下限的范围之内，叫做模拟电压在窗口里。

图 6.39-7 给出窗口比较模式的例子，该例程中窗口比较模式被选中，用户选择每个比较器四路正输入端中的一路，然后在芯片外部将该两组比较器的输入连接到一起。

如果 ACMP0S0 输出高且 ACMP1S1 输出低，意味着电压源在上限和下限之内，叫做电压源在窗口里，否则，电压源在窗口外。

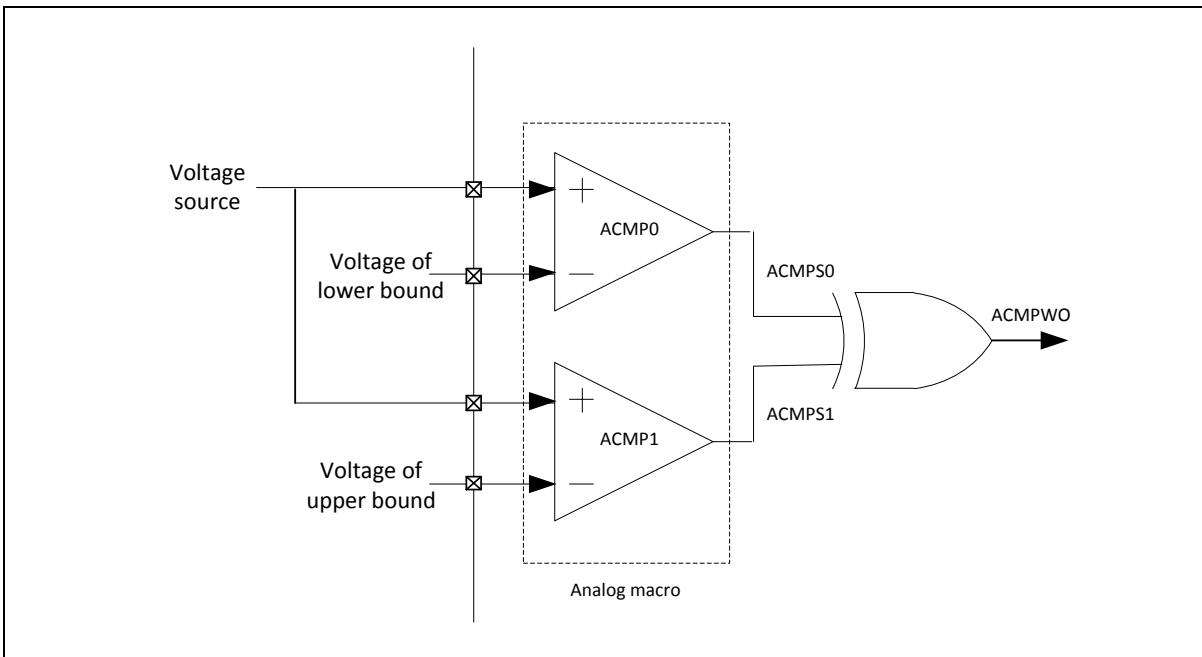


图 6.39-7 窗口比较模式示例

比较器窗口输出 (ACMPWO) 在 ACMP\_STATUS[16]，窗口比较逻辑的真值表如表 6.39-1.

ACMP0S0	ACMP1S1	ACMPWO
0	0	0
0	1	1
1	1	0
1	0	1

表 6.39-1 窗口比较逻辑真值表

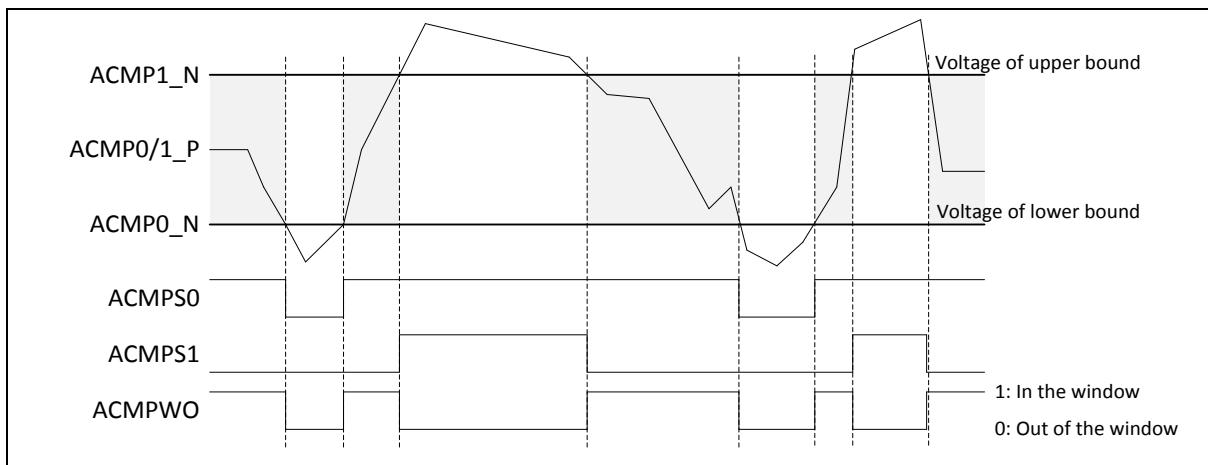


图 6.39-8 窗口比较模式示例

如图 6.39-8 所示，如果 ACMPWO 为 1，这就意味着正端输入电压在窗口内，否则就是在窗口外。因此 ACMPWO 可以被用作监控外部模拟管脚的电压转变。更进一步，ACMPWO 也可以用做窗口锁存，过滤功能和 ACMP 中断。

注意负端输入必须是不同的电压源，否则该功能就没有任何意义。

### 6.39.6 寄存器映射

R: 只读, W: 只写, R/W: 可读可写

寄存器	偏移量	R/W	描述	复位值
<b>ACMP 基地址:</b> <b>ACMP01_BA = 0x4004_5000</b>				
<b>ACMP_CTL0</b>	ACMP01_BA+0x00	R/W	模拟比较器 0 控制寄存器	0x0000_0000
<b>ACMP_CTL1</b>	ACMP01_BA+0x04	R/W	模拟比较器 1 控制寄存器	0x0000_0000
<b>ACMP_STATUS</b>	ACMP01_BA+0x08	R/W	模拟比较器状态寄存器	0x0000_0000
<b>ACMP_VREF</b>	ACMP01_BA+0x0C	R/W	模拟比较器参考电压控制寄存器	0x0000_0000

## 6.39.7 寄存器描述

ACMP\_CTL0 模拟比较器 0 控制寄存器

寄存器	偏移量	R/W	描述	复位值
ACMP_CTL0	ACMP01_BA+0x00	R/W	模拟比较器 0 控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved		MODESEL			Reserved		HYSSEL
23	22	21	20	19	18	17	16
Reserved					WCMPSEL	WLATEN	WKEN
15	14	13	12	11	10	9	8
FILTSEL			OUTSEL	Reserved		INTPOL	
7	6	5	4	3	2	1	0
POSSEL		NEGSEL		ACMPOINV	Reserved	ACMPIE	ACMPEN

位	描述
[31:30]	Reserved 保留.
[29:28]	MODESEL 传输延时模式选择 00 = 最大传输延时是 4.5uS, 操作电流是 1.2uA. 01 =最大传输延时是 2uS, 操作电流是 3uA. 10 =最大传输延时是 600nS, 操作电流是 10uA. 11 =最大传输延时是 200nS, 操作电流是 75uA.
[27:26]	Reserved 保留.
[25:24]	HYSSEL 迟滞回差电压选择 00 = 0mV. 01 = 10mV. 10 = 20mV. 11 = 30mV.
[23:19]	Reserved 保留
[18]	WCMPSEL 窗口比较模式选择 0 = 不选择窗口比较模式. 1 = 选择窗口比较模式.
[17]	WLATEN 窗口锁存模式使能 0 =窗口锁存模式禁止 1 = 窗口锁存模式使能
[16]	WKEN 掉电唤醒使能位 0 = 唤醒功能禁止. 1 = 唤醒功能使能
[15:13]	FILTSEL 比较器输出滤波采样时长选择

		000 = 滤波功能禁止. 001 = ACMP0输出经过 1 个连续的 PCLK 采样. 010 = ACMP0输出经过 2个连续的 PCLKs采样. 011 = ACMP0输出经过 4个连续的PCLKs采样. 100 = ACMP0输出经过 8个连续的PCLKs采样. 101 = ACMP0输出经过 16个连续的PCLKs采样. 110 = ACMP0输出经过 32个连续的PCLKs采样. 111 = ACMP0输出经过 64 个连续的PCLKs采样.
[12]	<b>OUTSEL</b>	<b>比较器滤波输出选择</b> 0 = 比较器 0 输出到 ACMP0_O管脚不经过滤波 1 = 比较器 0 输出到 ACMP0_O管脚经过滤波
[11:10]	<b>Reserved</b>	保留.
[9:8]	<b>INTPOL</b>	<b>中断条件极性选择</b> ACMPIFO 当下列输出条件被检测到时设为1 00 = 上升沿或下降沿. 01 = 上升沿. 10 = 下降沿. 11 = 保留.
[7:6]	<b>POSSEL</b>	<b>比较器正端输入选择</b> 00 = 从 ACMP0_P0输入. 01 = 从 ACMP0_P1输入. 10 = 从 ACMP0_P2输入. 11 = 从ACMP0_P3输入.
[5:4]	<b>NEGSEL</b>	<b>比较器负端输入选择</b> 00 = ACMP0_N 管脚. 01 = 内部比较器参考电压 (CRV). 10 = 带隙电压. 11 = DAC 输出.
[3]	<b>ACMPOINV</b>	<b>比较器输出反相</b> 0 = 禁止比较器 0 输出反相. 1 = 使能比较器 0 输出反相.
[2]	<b>Reserved</b>	保留.
[1]	<b>ACMPIE</b>	<b>比较器中断使能位</b> 0 = 比较器 0 中断禁止. 1 = 比较器 0 中断使能.如果 WKEN (ACMP_CTL0[16])设为 1, 中断唤醒功能也会使能.
[0]	<b>ACMPEN</b>	<b>比较器使能位</b> 0 = 比较器 0 禁止. 1 = 比较器 0 使能.

**ACMP\_CTL1 模拟比较器 1 控制寄存器**

寄存器	偏移量	R/W	描述	复位值
ACMP_CTL1	ACMP01_BA+0x04	R/W	模拟比较器 1 控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved		MODESEL			Reserved	HYSSEL	
23	22	21	20	19	18	17	16
Reserved					WCMPSEL	WLATEN	WKEN
15	14	13	12	11	10	9	8
FILTSEL			OUTSEL	Reserved		INTPOL	
7	6	5	4	3	2	1	0
POSSEL		NEGSEL		ACMPOINV	Reserved	ACMPIE	ACMPEN

位	描述	
[31:30]	Reserved	保留.
[29:28]	MODESEL	传输延时模式选择 00 = 最大传输延时是 4.5uS, 操作电流是 1.2uA. 01 =最大传输延时是 2uS, 操作电流是 3uA. 10 =最大传输延时是 600nS, 操作电流是 10uA. 11 =最大传输延时是 200nS, 操作电流是 75uA.
[27:26]	Reserved	保留.
[25:24]	HYSSEL	迟滞回差电压选择 00 = 0mV. 01 = 10mV. 10 = 20mV. 11 = 30mV.
[23:19]	Reserved	保留.
[18]	WCMPSEL	窗口比较模式选择 0 = 不选择窗口比较模式. 1 = 选择窗口比较模式.
[17]	WLATEN	窗口锁存模式使能 0 = 窗口锁存模式禁止 1 = 窗口锁存模式使能
[16]	WKEN	掉电唤醒使能位 0 = 唤醒功能禁止. 1 = 唤醒功能使能

位	描述
[15:13]	<b>FILTSEL</b> 比较器输出滤波采样时长选择 000 = 滤波功能禁止. 001 = ACMP1输出经过 1 个连续的 PCLK 采样. 010 = ACMP1输出经过 2个连续的PCLKs采样. 011 = ACMP1输出经过 4个连续的PCLKs采样. 100 = ACMP1输出经过 8个连续的PCLKs采样. 101 = ACMP1输出经过 16个连续的PCLKs采样. 110 = ACMP1输出经过 32个连续的PCLKs采样. 111 = ACMP1输出经过 64 c个连续的PCLKs采样.
[12]	<b>OUTSEL</b> 比较器输出选择 0 = 比较器 1 输出到 ACMP1_O管脚不经过滤波. 1 = 比较器 1 输出到 ACMP1_O管脚经过滤波.
[11:10]	<b>Reserved</b> 保留.
[9:8]	<b>INTPOL</b> 中断极性选择 ACMPIF1 当下列输出条件被检测到时设为1 00 = 上升沿或下降沿. 01 = 上升沿. 10 = 下降沿. 11 = 保留.
[7:6]	<b>POSSEL</b> 比较器正输入端选择 00 = 从 ACMP1_P0输入. 01 = 从 ACMP1_P1输入. 10 = 从 ACMP1_P2输入. 11 = 从ACMP1_P3输入.
[5:4]	<b>NEGSEL</b> 比较器负输入端选择 00 = ACMP1_N 管脚. 01 = 内部比较器参考电压 (CRV). 10 = 带隙电压. 11 = DAC 输出.
[3]	<b>ACMPOINV</b> 比较器输出反相 0 = 禁止比较器 1 输出反相. 1 = 使能比较器 1 输出反相.
[2]	<b>Reserved</b> 保留.
[1]	<b>ACMPIE</b> 比较器中断使能位 0 = 比较器 1 中断禁止. 1 = 比较器 1 中断使能.如果 WKEN (ACMP_CTL1[16])设为 1, 中断唤醒功能也会使能.
[0]	<b>ACMPEN</b> 比较器使能位 0 = 比较器 1 禁止. 1 = 比较器 1 使能.

ACMP\_STATUS 模拟比较器状态寄存器

寄存器	偏移量	R/W	描述	复位值
ACMP_STATUS	ACMP01_BA+0x08	R/W	模拟比较器状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved		ACMPS1	ACMPS0	Reserved		WKIF1	WKIF0
7	6	5	4	3	2	1	0
Reserved		ACMPO1	ACMPO0	Reserved		ACMPIF1	ACMPIF0

位	描述	
[31:17]	Reserved	保留
[16]	ACMPWO	<b>比较器窗口输出</b> 这一位展示窗口比较模式的输出状态 0 = 正向输入电压在窗口外 1 = 正向输入电压在窗口里
[15:14]	Reserved	保留.
[13]	ACMPS1	<b>比较器 1 状态</b> 允许程序同步于PCLK 读取., 当比较器 1 禁止 (ACMPEN (ACMP_CTL1[0])=0) 时清0
[12]	ACMPS0	<b>比较器 0 状态</b> 允许程序同步于PCLK 读取., 当比较器 0 禁止 (ACMPEN (ACMP_CTL0[0])=0) 时清0
[11:10]	Reserved	保留.
[9]	WKIF1	保留.
[8]	WKIF0	<b>比较器 1 掉电唤醒中断标志</b> 当ACMP1 唤醒中断事件发生时, 这一位置 1. 0 =没有掉电唤醒发生. 1 =发生掉电唤醒. <b>注意:</b> 此位写1清0.
[7:6]	Reserved	<b>比较器 0 掉电唤醒中断标志</b> 当ACMP0 唤醒中断事件发生时, 这一位置 1. 0 =没有掉电唤醒发生. 1 =发生掉电唤醒. <b>注意:</b> 此位写1清0.
[5]	ACMPO1	比较器 1 输出

位	描述
	允许程序同步于PCLK 读取., 当比较器 1 禁止 (ACMPEN (ACMP_CTL1[0])=0) 时清0
[4]	<b>ACMPO0</b> <b>比较器 0 输出</b> 允许程序同步于PCLK 读取., 当比较器 0 禁止 (ACMPEN (ACMP_CTL0[0])=0) 时清0
[3:2]	<b>Reserved</b> 保留.
[1]	<b>ACMPIF1</b> <b>比较器 1 中断标志</b> 当比较器1的输出检测到INTPOL (ACMP_CTL1[9:8])定义的条件成立, 此位由硬件置1, 当ACMPIE (ACMP_CTL1[1])为1会产生中断. <b>注意:</b> 此位写1清0..
[0]	<b>ACMPIF0</b> <b>比较器 0 中断标志</b> 当比较器0的输出检测到INTPOL (ACMP_CTL0[9:8])定义的条件成立, 此位由硬件置1, 当ACMPIE (ACMP_CTL0[1])为1会产生中断. <b>注意:</b> 此位写1清0..

ACMP\_VREF 模拟比较器参考电压控制寄存器

寄存器	偏移量	R/W	描述	复位值
ACMP_VREF	ACMP01_BA+0x0C	R/W	模拟比较器参考电压控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved	CRVSSEL	Reserved		CRVCTL			

位	描述	
[31:7]	Reserved	保留.
[6]	CRVSSEL	<b>CRV 电压源选择</b> 0 = V <sub>DDA</sub> 选择作为 CRV 电压源. 1 = SYS_VREFCTL 寄存器选择的参考电压作为 CRV 电压源.
[5:4]	Reserved	保留.
[3:0]	CRVCTL	<b>比较器参考电压设置</b> CRV = CRV 电压源 * (1/6+CRVCTL/24).

## 6.40 OPA 运算放大器

### 6.40.1 概述

芯片集成3个运算放大器。用户依据不同的应用独立的使能它们。为了测量需求有一个运算放大器输出和ADC的通道相连。运算放大器电路增益可编程。

### 6.40.2 特性

- 模拟电压输入范围: 0~ $V_{DD}$
- 3个运算放大器
- 施密特输出可做基本比较器功能
- 施密特输出可触发中断

## 6.40.3 框图

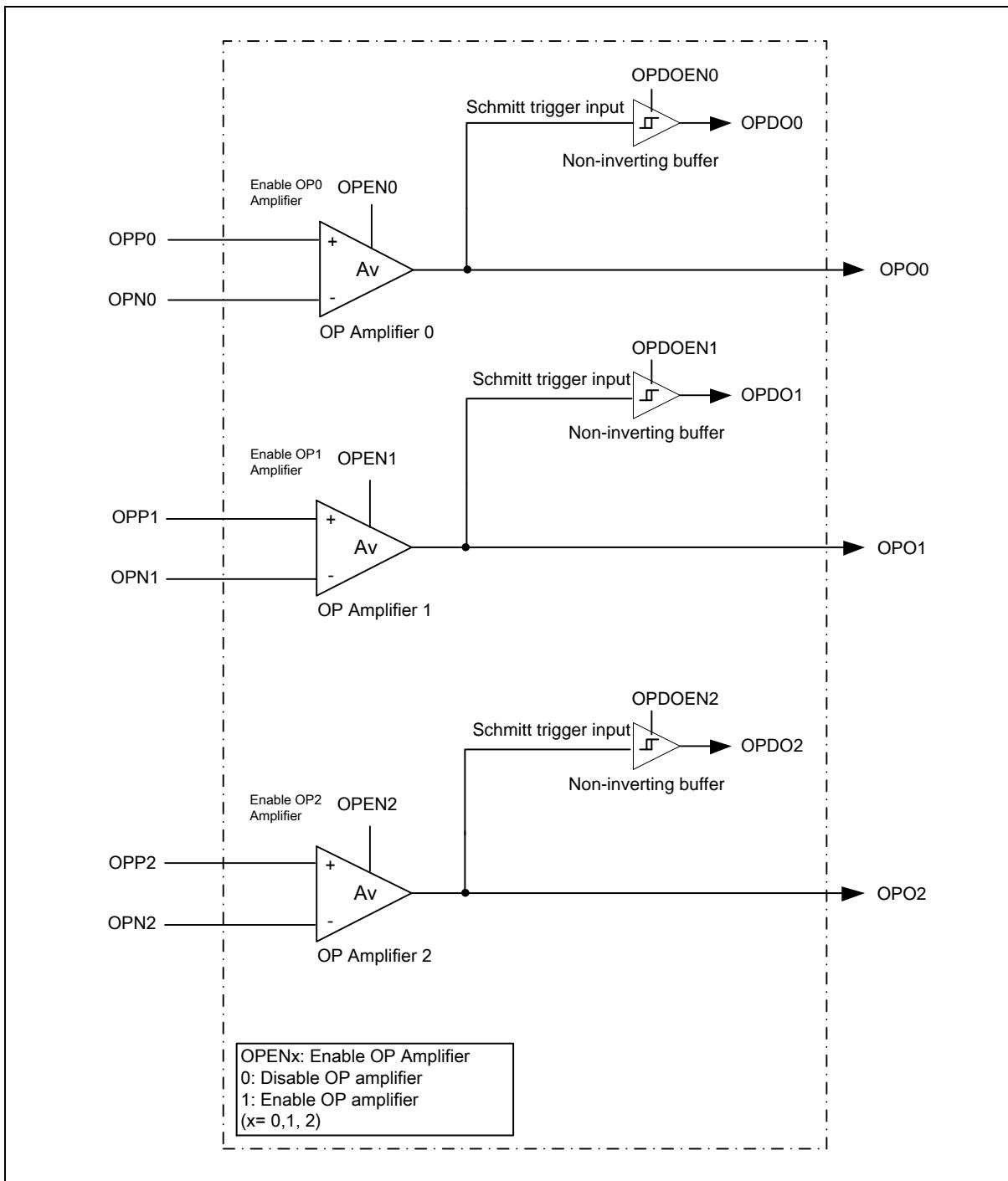


图 6.40-1 运算放大器框图

## 6.40.4 基本配置

## 6.40.4.1 OPA0 基本配置

- 时钟源配置
  - 使能 OPA0 外设时钟在OPACKEN (CLK\_APBCLK1[30])

- 复位配置
  - 复位OPA0 控制在OPARST (SYS\_IPRST2[30])
- 管脚配置

组	管脚名称	GPIO	MFP
OPA0	OPA0_N	PB.1	MFP1
	OPA0_O	PB.2	MFP1
	OPA0_P	PB.0	MFP1

#### 6.40.4.2 OPA1 基本配置

- 时钟源配置
  - 使能 OPA1 外设时钟在OPACKEN (CLK\_APBCLK1[30])
- 复位配置
  - 复位OPA1 控制在OPARST (SYS\_IPRST2[30])
- 管脚配置

组	管脚名称	GPIO	MFP
OPA1	OPA1_N	PA.9	MFP1
	OPA1_O	PA.10	MFP1
	OPA1_P	PA.8	MFP1

#### 6.40.4.3 OPA2 基本配置

- 时钟源配置
  - 使能 OPA2 外设时钟在OPACKEN (CLK\_APBCLK1[30]).
- 复位配置
  - 复位OPA2 控制在OPARST (SYS\_IPRST2[30])
- 管脚配置

组	管脚名称	GPIO	MFP
OPA2	OPA2_N	PD.11	MFP1
	OPA2_O	PD.12	MFP1
	OPA2_P	PD.10	MFP1

### 6.40.5 功能描述

#### 6.40.5.1 OP 放大器功能

运放通过 OPENx (OPA\_CTL[1:0]) 位使能，x=0, 1, 2 对应 OPA0, OPA1, OPA2。OPA 管脚功能参考图 6.40-1。为了测量OPA0的输出同时也内部连接到ADC通道。设置OPDOENx (OPA\_CTL[6:4])使能施密特触发。施密特输出使能并且OP输出改变时，数字输出OPDOx (OPA\_STATUS[2:0])置1。施密特输出禁止时，总是输出0。

#### 6.40.5.2 校准功能

OP 放大器的增益可调为 2, 4, 8倍... . 该电路有五个校准位，用于校准失调电压。校准可以通过设置 CALTRGx (OPA\_CALCTL[2:0])启动。用户可以根据应用设置 CALCLKx (OPA\_CALCTL[9:4]) 选择校准时钟， $x=0, 1, 2$ 。校准之后，输入失调电压可以降到  $\pm 1.6\text{mV}$ 以内。读 OPA\_CALST 寄存器可以监控校准状态，包含校准完成状态，NMOS，和 PMOS 校准结果状态。轨到轨共模输入范围是通过使用 NMOS 和 PMOS 差分对并联连接来实现的。

#### 6.40.5.3 中断源

当施密特输出中断 OPDOIEN0 (OPA\_CTL[8]), OPDOIEN1 (OPA\_CTL[9]), 和 OPDOIEN2 (OPA\_CTL[10])使能时，如果 OPA0, 1, 2 施密特输出发生改变，中断标志 OPDOIF0 (OPA\_STATUS[4]), OPDOIF1 (OPA\_STATUS[5]) 和 OPDOIF2 (OPA\_STATUS[6])会置1。这些标志写1清0，施密特输出可以作为比较器中断源。

#### 6.40.6 寄存器映射

R: 只读, W: 只写, R/W: 可读可写, C: 只能写0

寄存器	偏移量	R/W	描述	复位值
<b>OPA 基地址:</b>				
<b>OPA_BA = 0x4004_6000</b>				
<b>OPA_CTL</b>	OPA_BA+0x00	R/W	OP 放大器控制寄存器	0x0000_0000
<b>OPA_STATUS</b>	OPA_BA+0x04	R/W	OP 放大器状态寄存器	0x0000_0000
<b>OPA_CALCTL</b>	OPA_BA+0x08	R/W	OP 放大器校准控制寄存器	0x0000_0000
<b>OPA_CALST</b>	OPA_BA+0x0C	R	OP 放大器校准状态寄存器	0x0000_0000

## 6.40.7 寄存器描述

OPA\_CTL OPA 控制寄存器

寄存器	偏移量	R/W	描述	复位值
OPA_CTL	OPA_BA+0x00	R/W	OP放大器控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved					OPDOIEN2	OPDOIEN1	OPDOIENO
7	6	5	4	3	2	1	0
Reserved	OPDOIEN2	OPDOIEN1	OPDOIENO	Reserved	OPEN2	OPEN1	OPEN0

位	描述
[31:11]	<b>Reserved</b> 保留.
[10]	<b>OPDOIEN2</b> 运放 2 施密特输出中断使能位 0 = 禁止运放 2 施密特输出中断. 1 = 使能运放 2 施密特输出中断 当运放2施密特输出改变时，中断标志OPDOIF2置1，就是说 OPDOIEN2 为 1，会产生比较器中断请求
[9]	<b>OPDOIEN1</b> 运放 1 施密特输出中断使能位 0 = 禁止运放 1 施密特输出中断. 1 = 使能运放 1 施密特输出中断 当运放1施密特输出改变时，中断标志OPDOIF1置1，就是说 OPDOIEN1 为 1，会产生比较器中断请求
[8]	<b>OPDOIENO</b> 运放 0 施密特输出中断使能位 0 = 禁止运放 0 施密特输出中断. 1 = 使能运放 0 施密特输出中断 当运放0施密特输出改变时，中断标志OPDOIF0置1，就是说 OPDOIENO 为 1，会产生比较器中断请求
[7]	<b>Reserved</b> 保留.
[6]	<b>OPDOEN2</b> 运放2 施密特输出使能位 0 = 禁止OP2 施密特输出. 1 = 使能OP2 施密特输出.
[5]	<b>OPDOEN1</b> 运放1 施密特输出使能位 0 = 禁止OP1施密特输出. 1 = 使能OP1 施密特输出.

位	描述	
[4]	<b>OPDOEN0</b>	运放2 施密特输出使能位 0 = 禁止OP0 施密特输出. 1 = 使能OP0 施密特输出.
[3]	<b>Reserved</b>	保留.
[2]	<b>OPEN2</b>	放大器 2 使能位 0 = OP 放大器 2 禁止. 1 = OP 放大器 2 使能. 注意: OP 放大器 2 在OPEN2置1后需要等待 20μs输出稳定.
[1]	<b>OPEN1</b>	放大器 1 使能位 0 = OP 放大器 1 禁止. 1 = OP 放大器 1 使能. 注意: OP 放大器 1 在OPEN1置1后需要等待 20μs输出稳定.
[0]	<b>OPEN0</b>	放大器 0 使能位 0 = OP 放大器 0 禁止. 1 = OP 放大器 0 使能. 注意: OP 放大器 0 在OPEN0置1后需要等待 20μs输出稳定.

OPA\_STATUS OPA 状态寄存器

寄存器	偏移量	R/W	描述	复位值
OPA_STATUS	OPA_BA+0x04	R/W	OP 放大器状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved	OPDOIF2	OPDOIF1	OPDOIF0	Reserved	OPDO2	OPDO1	OPDO0

位	描述	
[31:7]	Reserved	保留
[6]	OPDOIF2	<b>OP2 施密特输出中断标志</b> 在OP2施密特输出改变时置1。此位写1清0.
[5]	OPDOIF1	<b>OP1 施密特输出中断标志</b> 在OP1施密特输出改变时置1。此位写1清0.
[4]	OPDOIF0	<b>OP0 施密特输出中断标志</b> 在OP0施密特输出改变时置1。此位写1清0.
[3]	Reserved	保留.
[2]	OPDO2	<b>OP2施密特输出</b> 施密特输出禁止时(OPDOEN2 = 0) 为 0.
[1]	OPDO1	<b>OP1施密特输出</b> 施密特输出禁止时(OPDOEN1 = 0) 为 0 .
[0]	OPDO0	<b>OP0施密特输出</b> 施密特输出禁止时(OPDOEN0 = 0) 为 0

OPA\_CALCTL OPA 校准控制寄存器

寄存器	偏移量	R/W	描述	复位值
OPA_CALCTL	OPA_BA+0x08	R/W	OP 放大器校准控制寄存器	0x0000_03F0

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved					CALRVS2	CALRVS1	CALRVS0
15	14	13	12	11	10	9	8
Reserved					CALCLK2		
7	6	5	4	3	2	1	0
CALCLK1		CALCLK0		Reserved	CALTRG2	CALTRG1	CALTRG0

位	描述	
[31:19]	Reserved	保留.
[18]	CALRVS2	<b>OPA2 校准参考电压选择</b> 0 = VREF 是 $\frac{1}{2}$ AVDD. 1 = VREF 从高 vcm 到低 vcm.
[17]	CALRVS1	<b>OPA1 校准参考电压选择</b> 0 = VREF 是 $\frac{1}{2}$ AVDD. 1 = VREF 从高 vcm 到低 vcm.
[16]	CALRVS0	<b>OPA0 校准参考电压选择</b> 0 = VREF 是 $\frac{1}{2}$ AVDD. 1 = VREF 从高 vcm 到低 vcm.
[15:10]	Reserved	保留.
[9:8]	CALCLK2	<b>OPA2 校准时钟速率选择</b> 00 = 1KHz 01 = 保留 10 = 保留 11 = 保留
[7:6]	CALCLK1	<b>OPA1 校准时钟速率选择</b> 00 = 1KHz 01 = 保留 10 = 保留 11 = 保留

位	描述	
[5:4]	<b>CALCLK0</b>	<b>OPA0 校准时钟速率选择</b> 00 = 1KHz 01 = 保留 10 = 保留 11 = 保留
[3]	<b>Reserved</b>	保留.
[2]	<b>CALTRG2</b>	<b>OP 放大器 2校准触发位</b> 0 = 停止校准 1 = 触发校准 <b>注1:</b> 在使能该位前，OPEN2应该被置位并且内部HIRC应该提前使能。 <b>注2:</b> 校准结束后硬件自动清除该位
[1]	<b>CALTRG1</b>	<b>OP 放大器 1校准触发位</b> 0 = 停止校准 1 = 触发校准 <b>注1:</b> 在使能该位前，OPEN1应该被置位并且内部HIRC应该提前使能。 <b>注2:</b> 校准结束后硬件自动清除该位
[0]	<b>CALTRG0</b>	<b>OP 放大器 0校准触发位</b> 0 = 停止校准 1 = 触发校准 <b>注1:</b> 在使能该位前，OPEN0应该被置位并且内部HIRC应该提前使能。 <b>注2:</b> 校准结束后硬件自动清除该位

OPA\_CALST OPA 校准状态寄存器

寄存器	偏移量	R/W	描述	复位值
OPA_CALST	OPA_BA+0x0C	R	OP 放大器校准状态寄存器	0x0000_0000

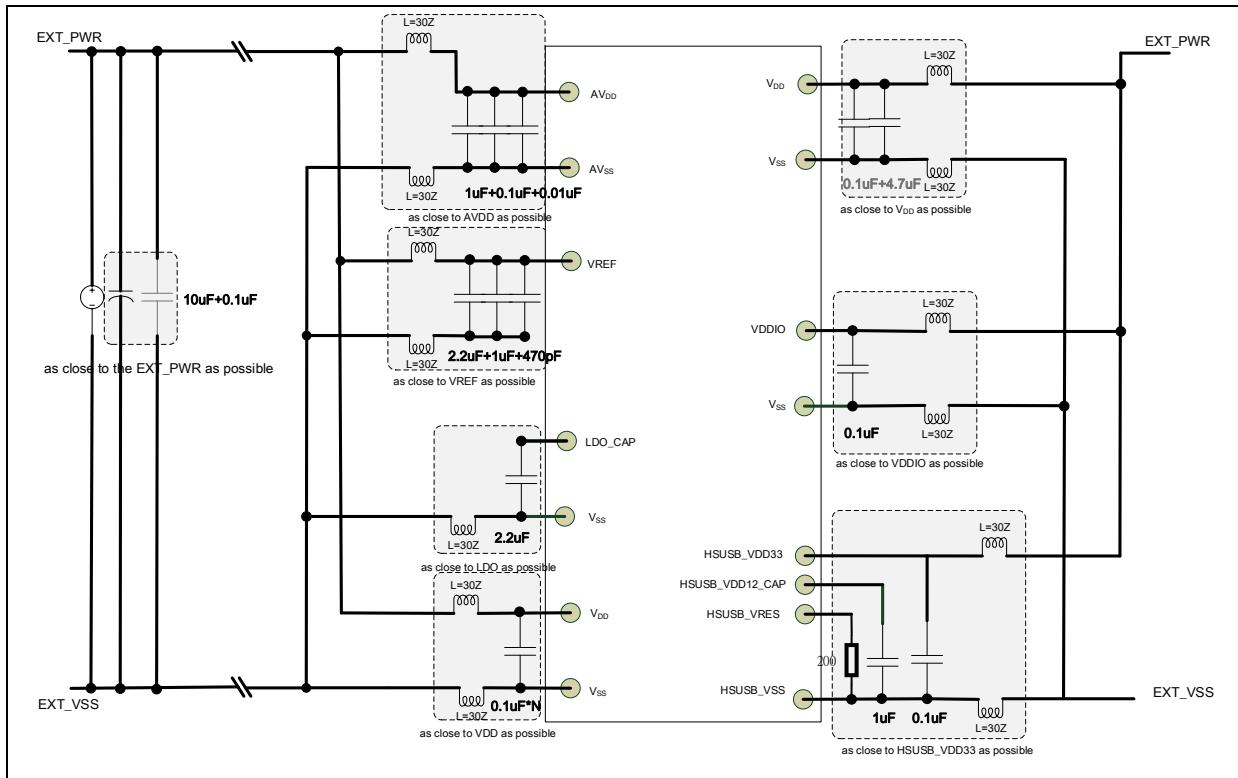
31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved					CALPS2	CALNS2	DONE2
7	6	5	4	3	2	1	0
Reserved	CALPS1	CALNS1	DONE1	Reserved	CALPS0	CALNS0	DONE0

位	描述
[31:11]	<b>Reserved</b> 保留.
[10]	<b>CALPS2</b> OP2 PMOS校准结果 0 = 通过. 1 = 失败.
[9]	<b>CALNS2</b> OP2 NMOS校准结果 0 = 通过. 1 = 失败.
[8]	<b>DONE2</b> OP2 校准完成状态 0 = 正在校准. 1 = 校准完成.
[7]	<b>Reserved</b> 保留.
[6]	<b>CALPS1</b> OP1 PMOS校准结果 0 = 通过. 1 = 失败.
[5]	<b>CALNS1</b> OP1 NMOS校准结果 0 = 通过. 1 = 失败.
[4]	<b>DONE1</b> OP1 校准完成状态 0 = 正在校准. 1 = 校准完成.
[3]	<b>Reserved</b> 保留.

位	描述	
[2]	<b>CALPS0</b>	<b>OP0 PMOS校准结果</b> 0 = 通过. 1 = 失败.
[1]	<b>CALNS0</b>	<b>OP0 NMOS校准结果</b> 0 = 通过. 1 = 失败.
[0]	<b>DONE0</b>	<b>OP0 校准完成状态</b> 0 = 正在校准. 1 = 校准完成.

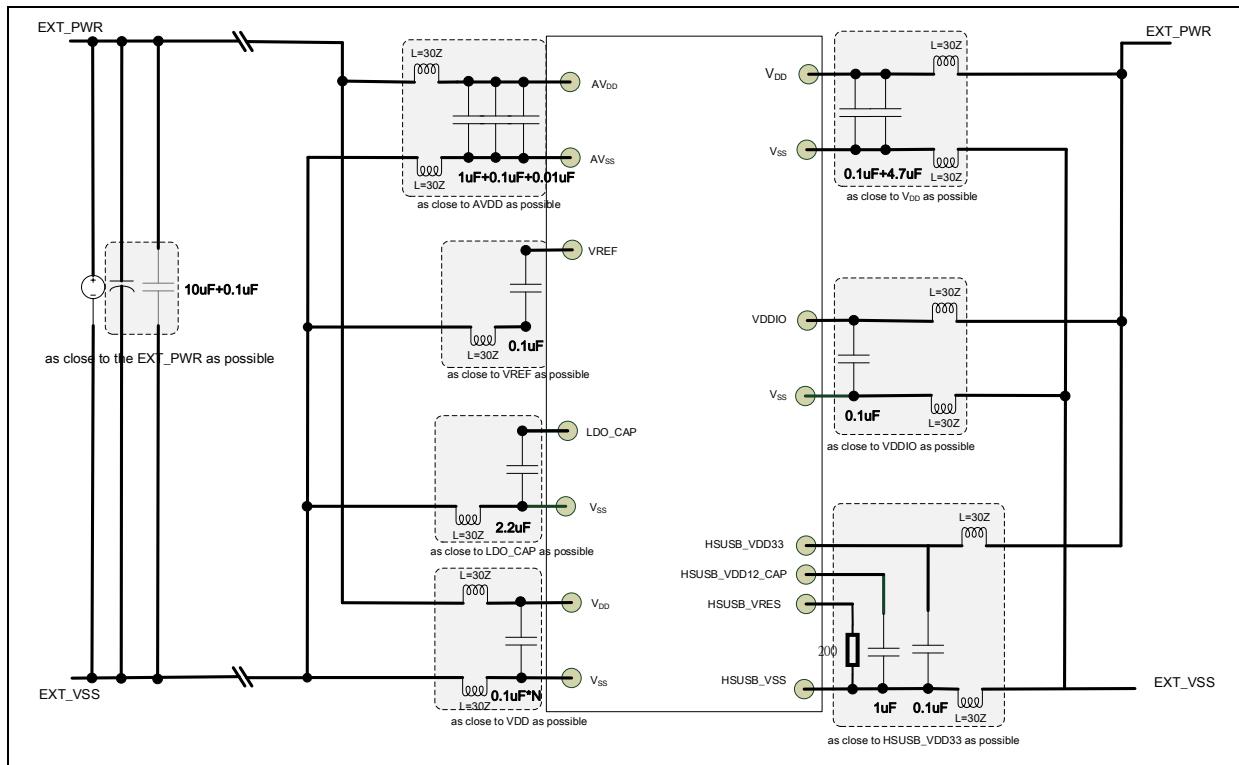
## 7 应用电路

### 7.1 外接Vref电源线路图

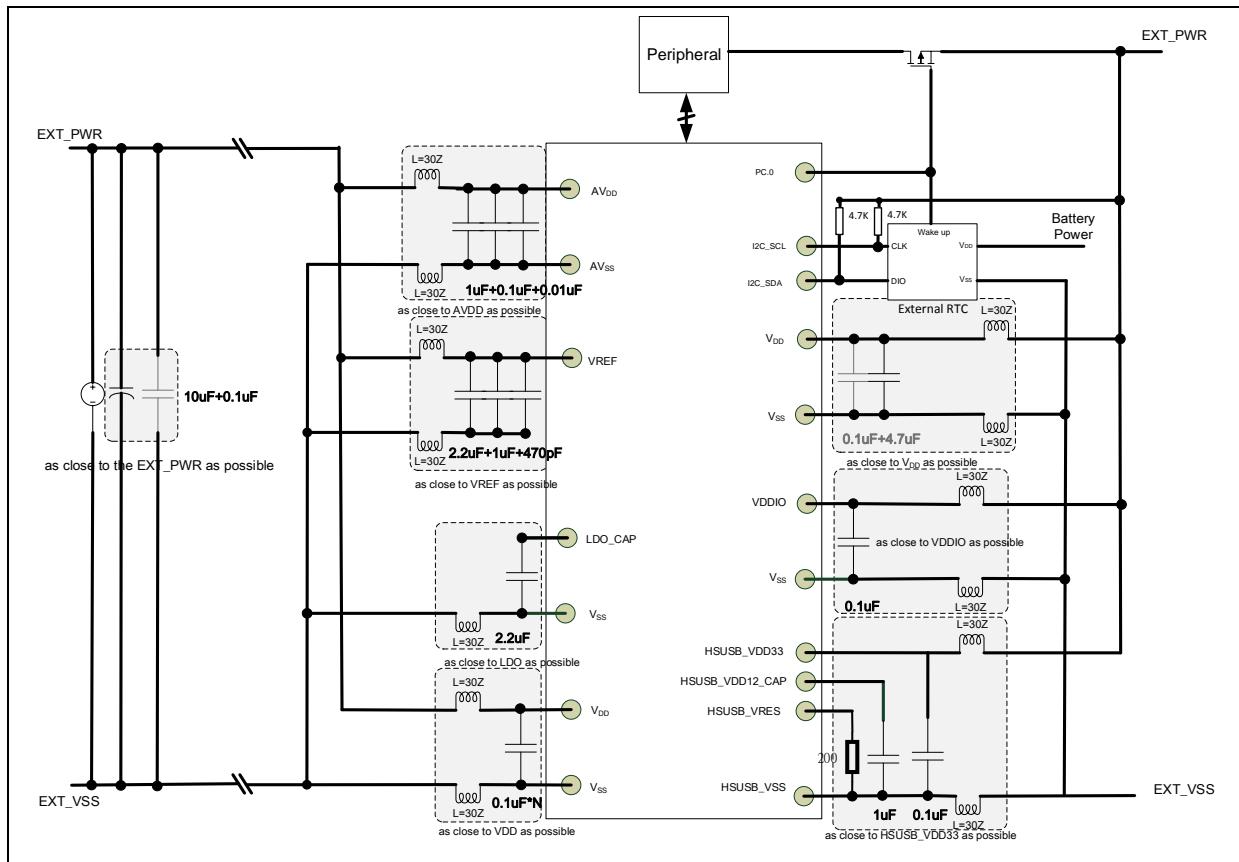


註: LDO\_CAP 腳位的電容值加總為 2.2uF

## 7.2 内置Vref电源线路图

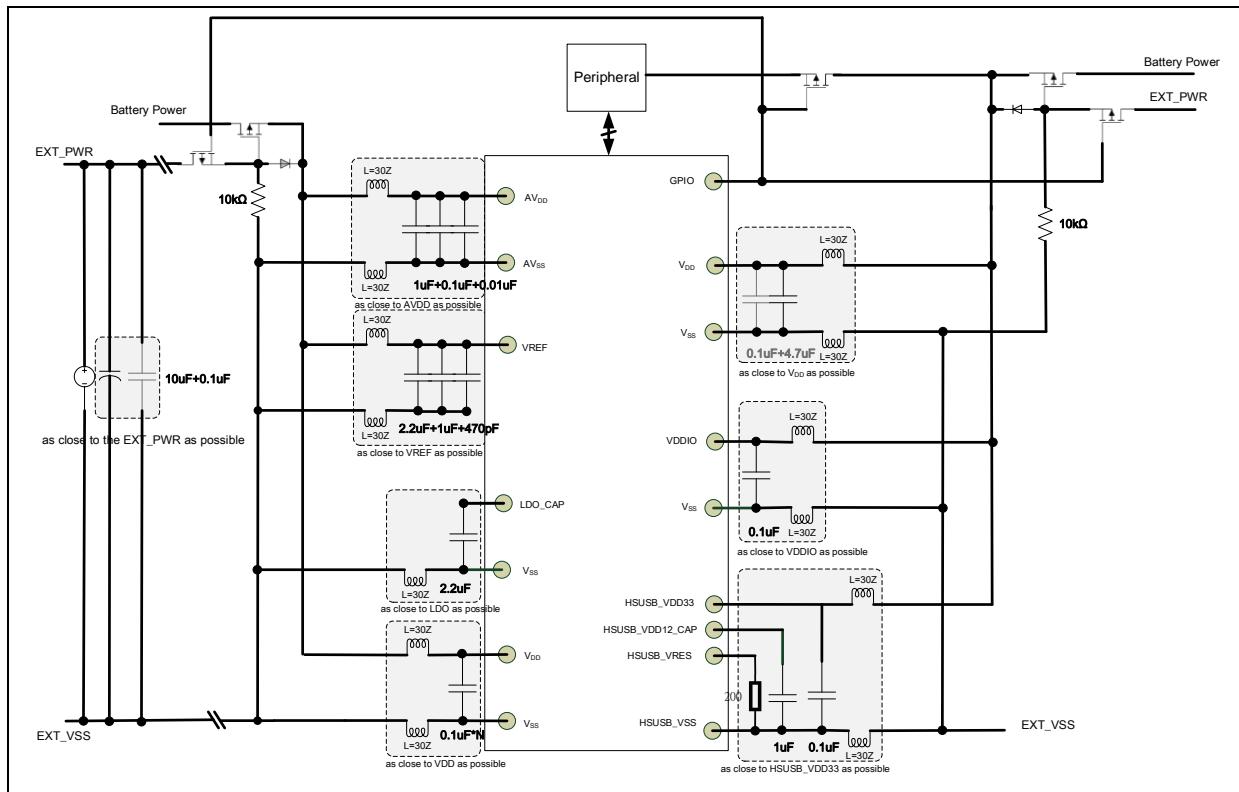


## 7.3 外接Vref及外接RTC线路图



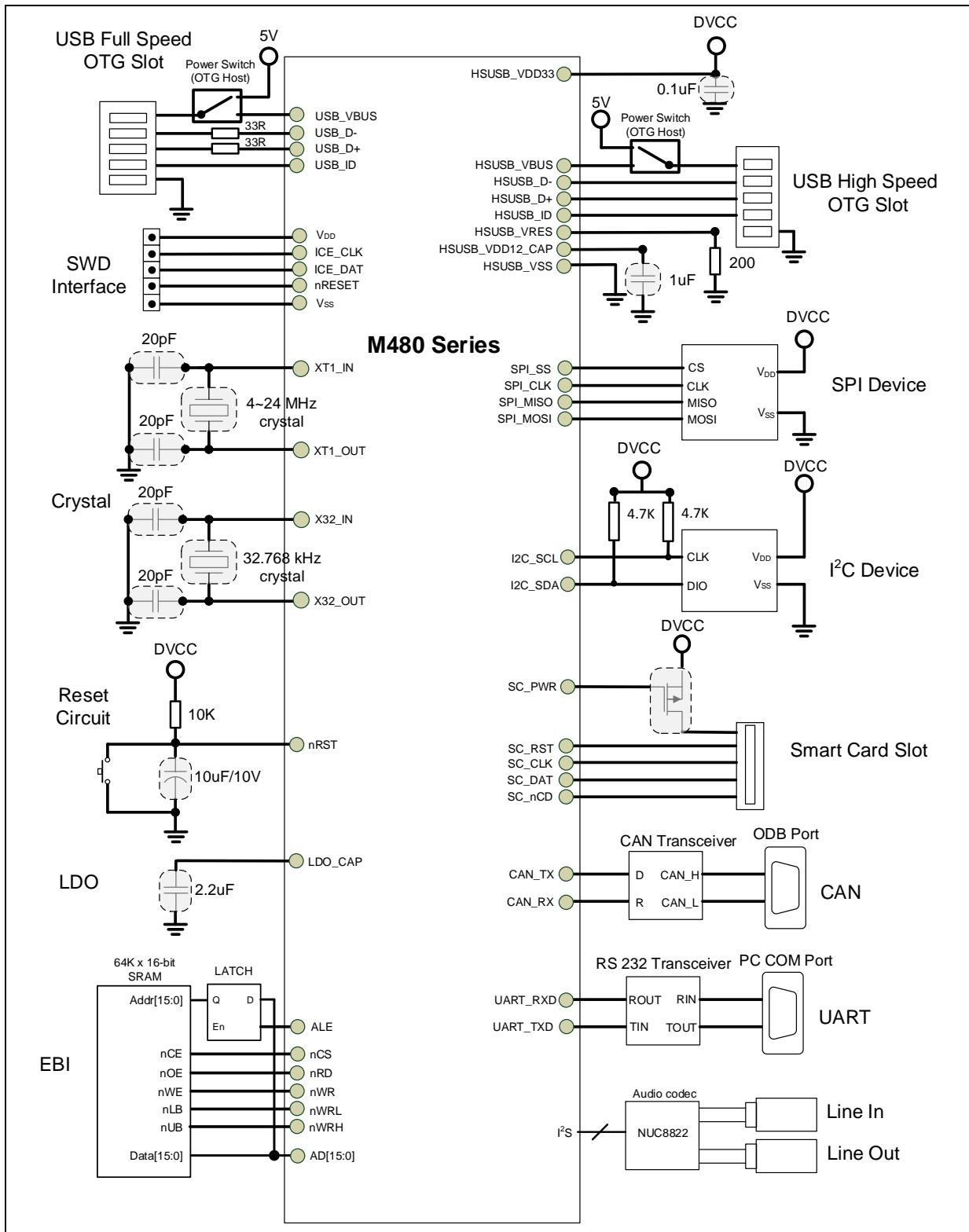
註: LDO\_CAP腳位的電容值加總為2.2uF

## 7.4 外接Vref及内置RTC线路图



註: LDO\_CAP 腳位的電容值加總為 2.2μF

## 7.5 外设应用线路图

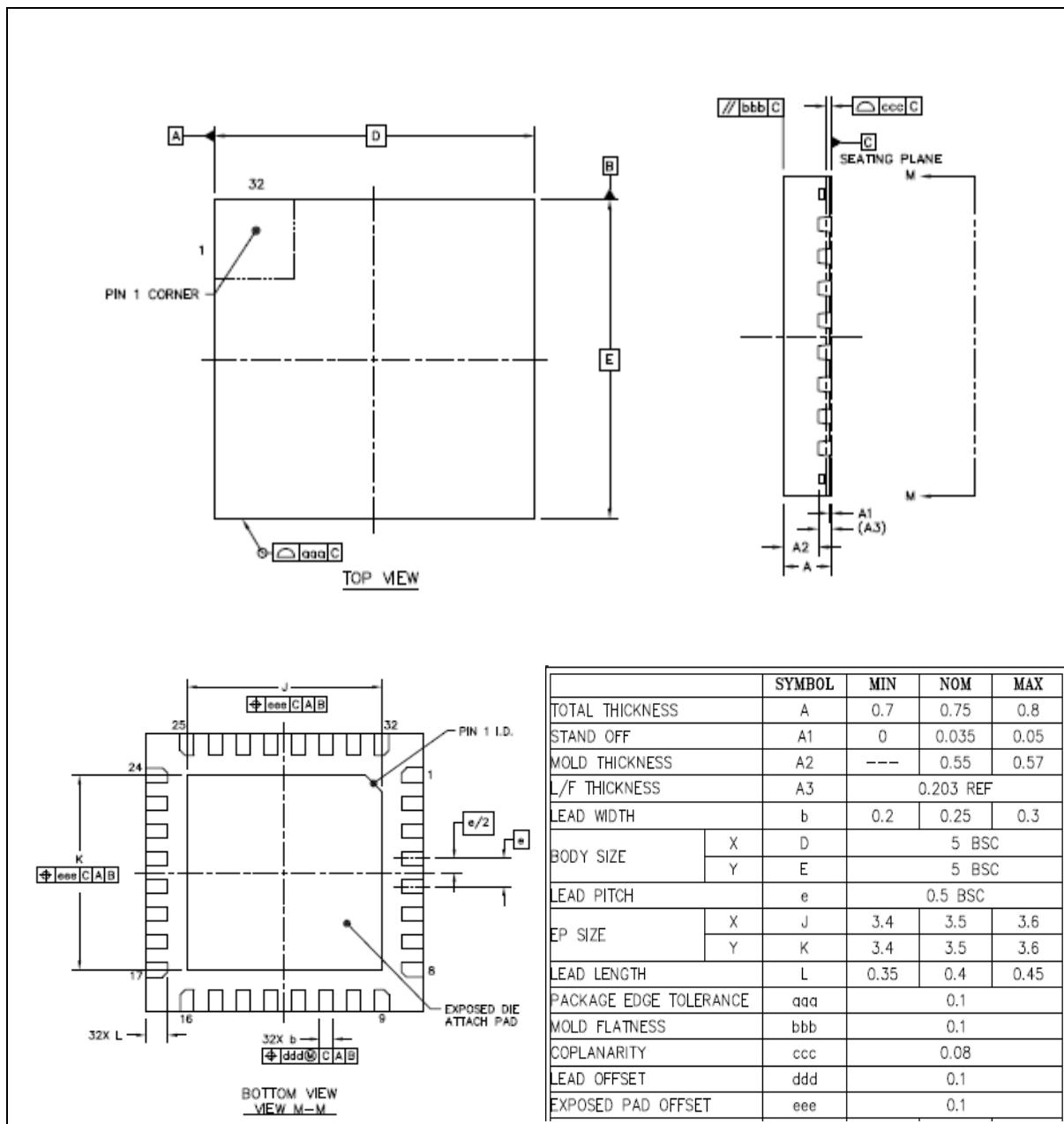


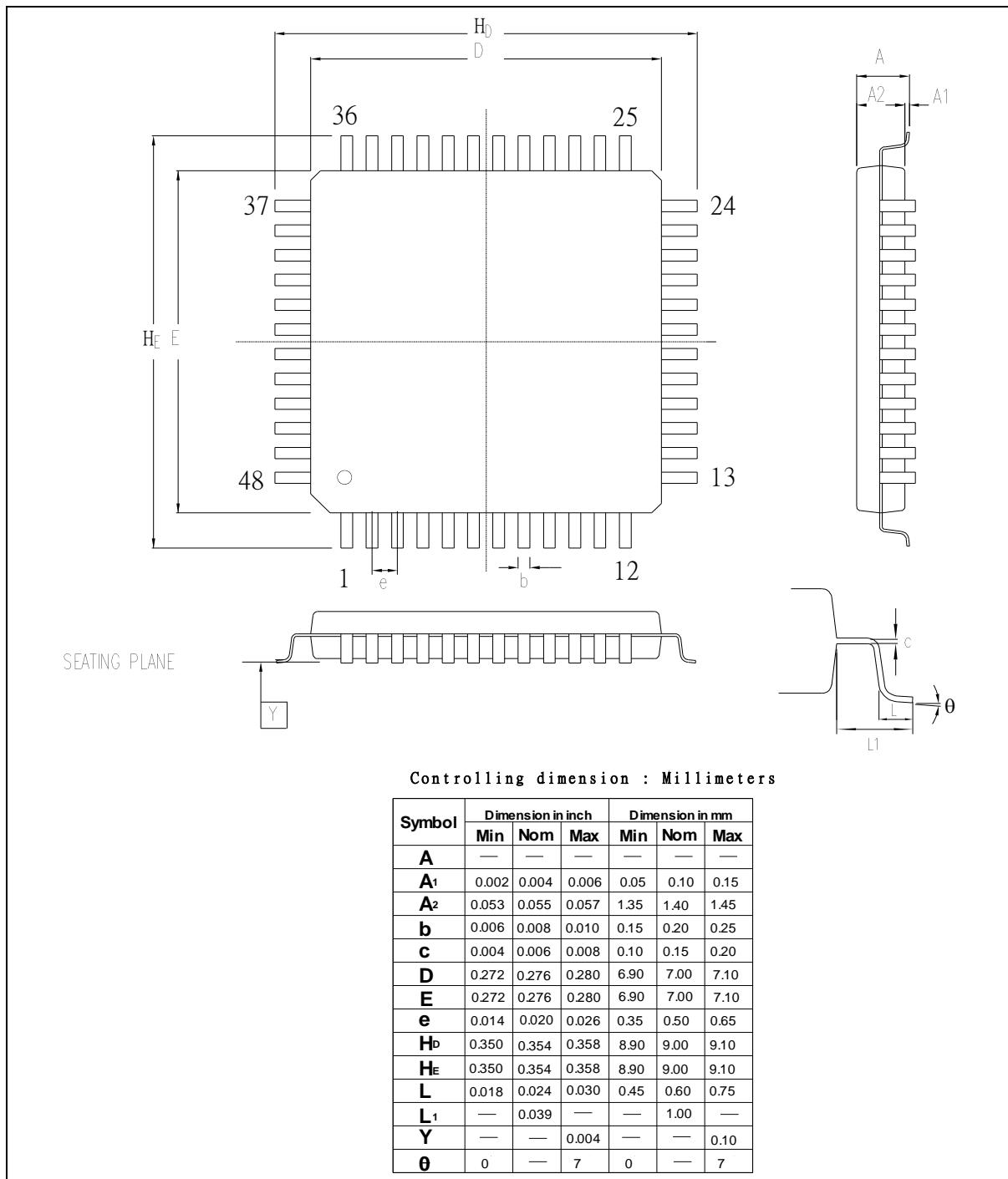
註: 使用USB或USB HS无OTG时, USB\_ID, HSUSB\_ID可浮接。

註: LDO\_CAP脚位的電容值加總為2.2uF

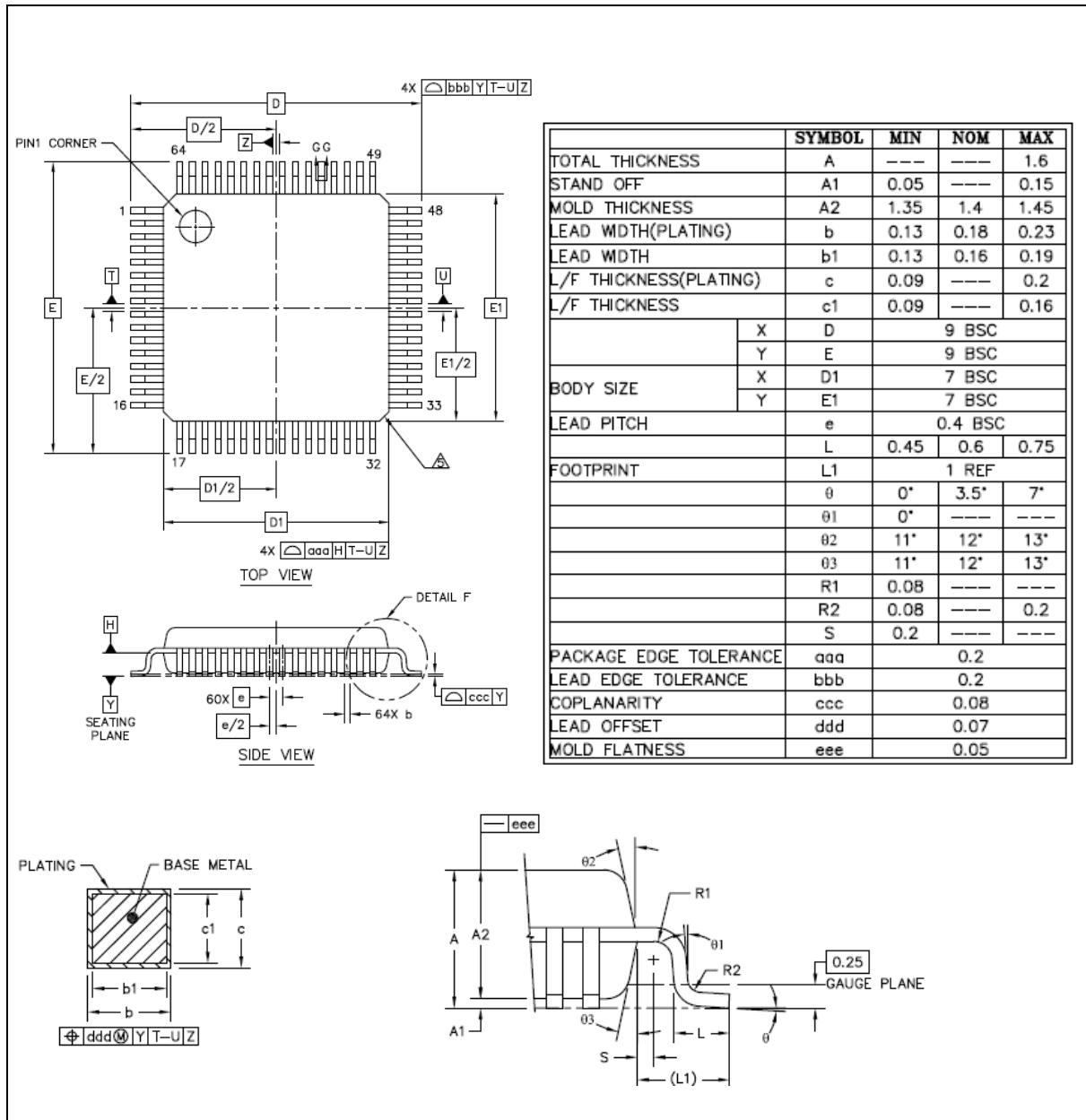
## 8 封装尺寸

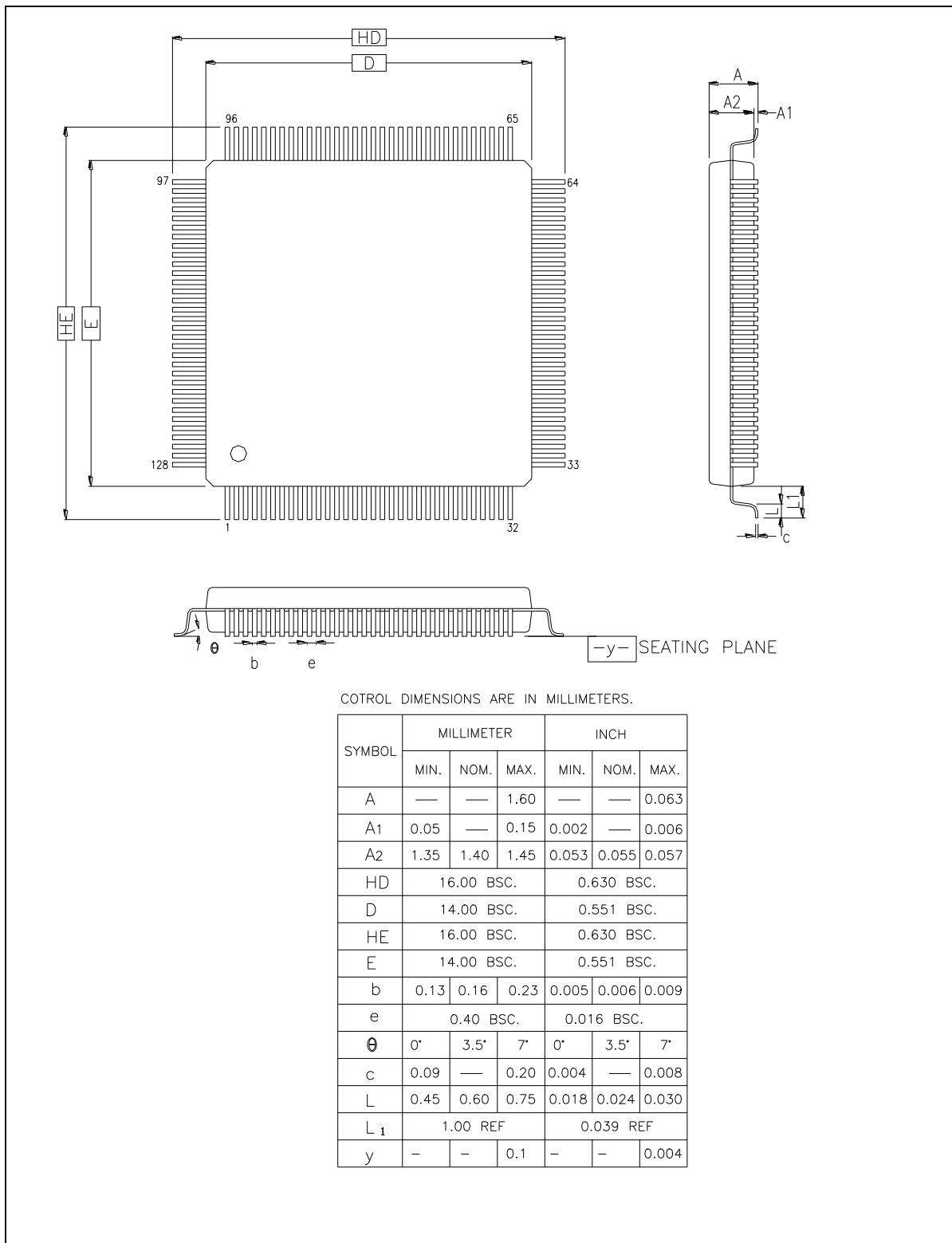
### 8.1 QFN 33L (5x5x0.8 mm<sup>3</sup> Pitch 0.5 mm)

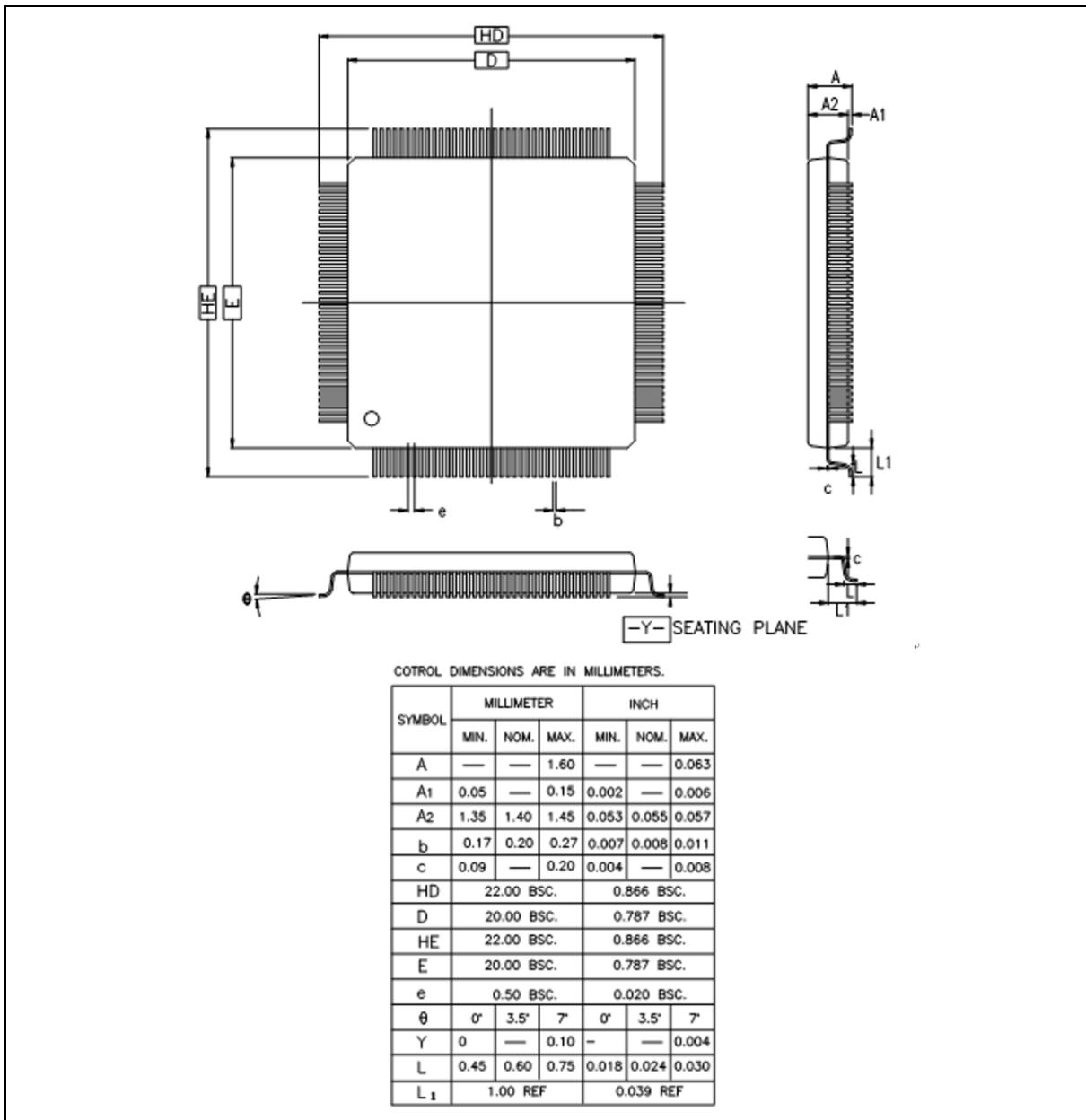


**8.2 LQFP 48L (7x7x1.4 mm<sup>3</sup> Footprint 2.0mm)**



8.3 LQFP 64L (7x7x1.4 mm<sup>3</sup> footprint 2.0 mm)

**8.4 LQFP 128L (14x14x1.4 mm<sup>3</sup> footprint 2.0 mm)**

8.5 LQFP 144L (20x20x1.4 mm<sup>3</sup> footprint 2.0 mm)

## 9 缩写

### 9.1 缩写

缩写	描述
ACMP	模拟比较器Controller
ADC	Analog-to-Digital Converter
AES	Advanced Encryption Standard
APB	Advanced Peripheral Bus
AHB	Advanced High-Performance Bus
BOD	Brown-out Detection
CAN	Controller Area Network
DAP	Debug Access Port
DES	Data Encryption Standard
EADC	Enhanced Analog-to-Digital Converter
EBI	External Bus Interface
EMAC	Ethernet MAC Controller
EPWM	Enhanced Pulse Width Modulation
FIFO	First In, First Out
FMC	Flash Memory Controller
FPU	Floating-point Unit
GPIO	General-Purpose Input/Output
HCLK	The Clock of Advanced High-Performance Bus
HIRC	12 MHz Internal High Speed RC Oscillator
HXT	4~24 MHz External High Speed Crystal Oscillator
IAP	In Application Programming
ICP	In Circuit Programming
ISP	In System Programming
LDO	Low Dropout Regulator
LIN	Local Interconnect Network
LIRC	10 kHz internal low speed RC oscillator (LIRC)
MPU	Memory Protection Unit
NVIC	Nested Vectored Interrupt Controller
PCLK	The Clock of Advanced Peripheral Bus
PDMA	Peripheral Direct Memory Access
PLL	Phase-Locked Loop

PWM	Pulse Width Modulation
QEI	Quadrature Encoder Interface
SD	Secure Digital
SPI	Serial Peripheral Interface
SPS	Samples per Second
TDES	Triple Data Encryption Standard
TK	Touch Key
TMR	Timer Controller
UART	Universal Asynchronous Receiver/Transmitter
UCID	Unique Customer ID
USB	Universal Serial Bus
WDT	Watchdog Timer
WWDT	Window Watchdog Timer

表 9.1-1 缩写表

## 10 版本历史

日期	版本	描述
2018.3	1.00	初始版本

### Important Notice

Nuvoton Products are neither intended nor warranted for usage in systems or equipment, any malfunction or failure of which may cause loss of human life, bodily injury or severe property damage. Such applications are deemed, "Insecure Usage".

Insecure usage includes, but is not limited to: equipment for surgical implementation, atomic energy control instruments, airplane or spaceship instruments, the control or operation of dynamic, brake or safety systems designed for vehicular use, traffic signal instruments, all types of safety devices, and other applications intended to support or sustain life.

All Insecure Usage shall be made at customer's risk, and in the event that third parties lay claims to Nuvoton as a result of customer's Insecure Usage, customer shall indemnify the damages and liabilities thus incurred by Nuvoton.

Please note that all data and specifications are subject to change without notice.  
All the trademarks of products and companies mentioned in this datasheet belong to their respective owners.