



# MC9S08DZ60

# MC9S08DZ48

# MC9S08DZ32

# MC9S08DZ16

产品介绍：详细信息

*HCS08*  
微控制器

MC9S08DZ60  
第三版  
10/2007

[freescale.com](http://freescale.com)





# MC9S08DZ60 系列产品的特性

## 8 位 HCS08 中央处理器 (CPU)

- 40-MHz HCS08 CPU (20-MHz 总线)
- HC08 指令集, 带附加的 BGND 指令
- 支持最多 32 个中断 / 复位源

## 片内存储器

- 整个工作电压和温度范围内可读取 / 编程 / 擦除的 Flash 存储器
  - MC9S08DZ60 = 60K
  - MC9S08DZ48 = 48K
  - MC9S08DZ32 = 32K
  - MC9S08DZ16 = 16K
- 最大2K的EEPROM在线可编程内存; 支持8字节单页或4字节双页擦除分区; 执行Flash程序的同时可进行编程和擦除操作; 支持擦除取消操作
- 最大 4K 的随机存取内存 (RAM)

## 省电模式

- 两种超低功耗停止模式
- 降低功耗的等待模式
- 超低功耗实时时钟中断, 在运行、等待和停止模式下均可操作

## 时钟源选项

- 振荡器 (XOSC) — 闭环控制的皮尔斯 (Pierce) 振荡器; 支持范围 31.25 kHz 至 38.4 kHz 或 1 MHz 至 16MHz 之间的晶体或陶瓷谐振器
- 多功能时钟生成器 (MCG) — PLL 和 FLL 模式 (在使用内部温度补偿时 FLL 能够达到 1.5% 内的偏差); 带微调功能的内部参考时钟源; 带可选择晶体振荡器或陶瓷谐振器的外部参考时钟源

## 系统保护

- 监视微控制器正常操作的看门狗 (COP) 复位, 支持选择专用的后备 1-kHz 内部时钟源或总线时钟运行
- 带复位和中断的低压检测电路; 可选择的电压阀值
- 支持非法指令代码复位
- 支持非法操作地址复位
- 支持 Flash 块保护
- 支持时钟信号丢失保护

## 开发支持

- 单线背景调试接口
- 片上及在线仿真 (ICE), 带总线实时捕获功能

## 外围设备

- ADC — 24 通道, 12 位分辨率, 2.5uS 转换时间, 自动比较功能, 1.7 mV/°C 温度传感器, 包含内部能隙参考源通道
- ACMPx — 两个模拟比较器, 支持比较器输出的上升、下降或任意边沿触发的中断; 可选择与内部参考电压源进行比较
- MSCAN — CAN 协议 - V2.0 A 和 B; 支持标准和扩展数据帧; 支持远程帧; 5 个带有 FIFO 存储机制的接收缓冲器; 灵活的接收识别符过滤器, 可编程如下: 2 x 32 位、 4 x 16 位或 8 x 8 位
- SCIx — 两个 SCI, 可支持 LIN 2.0 协议和 SAE J2602 协议; 全双工; 主节点支持 break 信号生成; 从节点支持 break 信号检测; 支持激活边沿唤醒
- SPI — 全双工或单线双向; 双重缓冲发射和接收; 主从模式选择; 支持高位优先或低位优先的移位
- IIC — 支持最高 100kbps 的总线波特率; 多主节点模式运行; 可编程的从地址; 通用呼叫地址; 逐字节数据传输驱动的中断
- TPMx — 一个 6 通道 (TPM1) 和一个 2 通道 (TPM2); 可支持输入捕捉, 输出比较, 或每个通道带缓冲的边沿对齐 PWM 输出
- RTC — (实时时钟计数器) 8 位模数计数器, 带基于二进制或十进制的预分频器; 实时时钟功能, 使用外部晶体和 RTC 来确保精确时基、时间、日历或任务调度功能; 内带低功耗振荡器 (1 kHz), 用于周期唤醒而不需要外部器件

## 输入 / 输出

- 53 个通用输入 / 输出 (I/O) 管脚和 1 个专用输入管脚
- 24 个中断管脚, 每个管脚带触发极性选择
- 所有输入管脚上带电压滞后和可配置的上下拉器件
- 所有输入管脚上可配置输出斜率和驱动强度

## 封装选项

- 64 管脚小尺寸四方扁平封装 (LQFP) — 10x10 mm
- 48 管脚小尺寸四方扁平封装 (LQFP) — 7x7 mm
- 32 管脚小尺寸四方扁平封装 (LQFP) — 7x7 mm



# 章节列表

章节号	标题	页码
第 1 章	器件概述 .....	19
第 2 章	管脚和连接.....	25
第 3 章	操作模式 .....	33
第 4 章	存储器 .....	39
第 5 章	复位、中断和系统总控制 .....	65
第 6 章	并行输入 / 输出控制 .....	81
第 7 章	中央处理器 (S08CPUV3) .....	107
第 8 章	多功能时钟发生器 (S08MCGV1) .....	127
第 9 章	模拟比较器 (S08ACMPV3).....	157
第 10 章	数模转换器 (S08ADC12V1).....	163
第 11 章	IIC 模块 (S08IICV2) .....	189
第 12 章	飞思卡尔控制器局域网 (S08MSCANV1) .....	207
第 13 章	串行外围器件接口 (S08SPIV3) .....	257
第 14 章	串行通信接口 (S08SCIV4).....	271
第 15 章	实时计数器 (S08RTCV1) .....	289
第 16 章	定时器脉冲宽度调节器 (S08TPMV3) .....	299
第 17 章	开发支持 .....	323
附录 A	电气特征 .....	342
附录 B	定时器脉宽调制器 (TPMV2).....	364
附录 C	订购信息和机械图 .....	378



# 目录

章节号	标题	页码
第 1 章 器件概述		
1.1	MC9S08DZ60 系列器件.....	19
1.2	MCU 结构图.....	20
1.3	系统时钟分配.....	23
第 2 章 管脚和连接		
2.1	器件管脚分配.....	25
2.2	推荐的系统连接.....	28
2.2.1	电源 .....	29
2.2.2	振荡器 .....	29
2.2.3	RESET (复位) .....	29
2.2.4	后台调试和模式选择 .....	30
2.2.5	ADC 参考管脚 ( $V_{REFH}$ , $V_{REFL}$ ) .....	30
2.2.6	通用 I/O 和外围设备端口 .....	30
第 3 章 操作模式		
3.1	简介.....	33
3.2	特性.....	33
3.3	运行模式.....	33
3.4	主动后台模式.....	33
3.5	等待模式.....	34
3.6	停止模式.....	34
3.6.1	Stop3 模式 .....	35
3.6.2	Stop2 模式 .....	36
3.6.3	停止模式中的片上外围模块 .....	36
第 4 章 存储器		
4.1	MC9S08DZ60 系列产品存储器映射.....	39
4.2	复位和中断向量分配.....	40
4.3	寄存器地址和位分配.....	41
4.4	RAM.....	49
4.5	Flash 和 EEPROM .....	49
4.5.1	特性 .....	49

章节号	标题	页码
4.5.2	编程和擦除时间 .....	50
4.5.3	编程和擦除命令的执行 .....	50
4.5.4	突发编程执行 .....	52
4.5.5	分区擦除终止 .....	53
4.5.6	访问错误 .....	55
4.5.7	块保护 .....	55
4.5.8	向量重定向 .....	56
4.5.9	安全性 .....	56
4.5.10	EEPROM 映射 .....	57
4.5.11	Flash 和 EEPROM 寄存器及控制位 .....	57

## 第 5 章 复位、中断和系统总控制

5.1	介绍 .....	65
5.2	特性 .....	65
5.3	MCU 复位 .....	65
5.4	计算机正常操作 (COP) 看门狗 .....	66
5.5	中断 .....	67
	5.5.1 中断堆栈帧 .....	67
	5.5.2 外部中断请求 (IRQ) 管脚 .....	68
	5.5.3 中断向量、源和本地掩码 .....	69
5.6	低电压检测 (LVD) 系统 .....	71
	5.6.1 加电复位操作 .....	71
	5.6.2 低压检测 (LVD) 复位操作 .....	71
	5.6.3 低压警告 (LVW) 中断操作 .....	71
5.7	MCLK 输出 .....	71
5.8	复位、中断及系统控制寄存器和控制位 .....	71
	5.8.1 中断管脚请求状态和控制寄存器 (IRQSC) .....	72
	5.8.2 系统复位状态寄存器 (SRS) .....	72
	5.8.3 系统后台调试强制复位寄存器 (SBDFR) .....	74
	5.8.4 系统选项寄存器 1 (SOPT1) .....	74
	5.8.5 系统选项寄存器 2 (SOPT2) .....	75
	5.8.6 系统器件识别寄存器 (SDIDH, SDIDL) .....	76
	5.8.7 系统电源管理状态和控制寄存器 1 (SPMSC1) .....	77
	5.8.8 系统电源管理状态和控制寄存器 2 (SPMSC2) .....	78

## 第 6 章 并行输入 / 输出控制

6.1	端口数据和数据方向 .....	81
6.2	上拉、斜率和驱动强度 .....	82
6.3	管脚中断 .....	83
	6.3.1 仅边沿敏感度 .....	83

章节号	标题	页码
6.3.2	边沿和电平敏感度 .....	83
6.3.3	上拉 / 下拉电阻器 .....	84
6.3.4	管脚中断初始化 .....	84
6.4	停止模式中的管脚行为 .....	84
6.5	并行 I/O 和管脚控制寄存器 .....	84
6.5.1	A 端口寄存器 .....	84
6.5.2	B 端口寄存器 .....	88
6.5.3	C 端口寄存器 .....	92
6.5.4	D 端口寄存器 .....	95
6.5.5	E 端口寄存器 .....	99
6.5.6	F 端口寄存器 .....	101
6.5.7	G 端口寄存器 .....	104

## 第 7 章 中央处理器 (S08CPUV3)

7.1	介绍 .....	107
	7.1.1 特性 .....	107
7.2	程序员模型和 CPU 寄存器 .....	108
	7.2.1 累加器 (A) .....	108
	7.2.2 索引寄存器 (H:X) .....	108
	7.2.3 堆栈指针 (SP) .....	109
	7.2.4 程序计数器 (PC) .....	109
	7.2.5 条件码寄存器 (CCR) .....	109
7.3	寻址模式 .....	111
	7.3.1 固有寻址模式 (INH) .....	111
	7.3.2 关联寻址模式 (REL) .....	111
	7.3.3 立即寻址模式 (IMM) .....	111
	7.3.4 直接寻址模式 (DIR) .....	111
	7.3.5 扩展寻址模式 (EXT) .....	111
	7.3.6 索引寻址模式 .....	111
7.4	特殊运算 .....	113
	7.4.1 复位顺序 .....	113
	7.4.2 中断时序 .....	113
	7.4.3 等待模式操作 .....	114
	7.4.4 停止模式操作 .....	114
	7.4.5 BGND 指令 .....	114
7.5	HCS08 指令集小结 .....	115

## 第 8 章 多功能时钟发生器 (S08MCGV1)

8.1	介绍 .....	127
	8.2.1 特性 .....	129

章节号	标题	页码
8.2.2	运行模式 .....	131
8.3	外部信号描述.....	131
8.4	寄存器定义.....	131
8.4.1	MCG 控制寄存器 1 (MCGC1) .....	131
8.4.2	MCG 控制寄存器 2 (MCGC2) .....	132
8.4.3	MCG 修正寄存器 (MCGTRM) .....	133
8.4.4	MCG 状态和控制寄存器 (MCGSC) .....	134
8.4.5	MCG Control Register 3 (MCGC3) .....	135
8.5	特性描述.....	136
8.5.1	运行模式 .....	136
8.5.2	模式切换 .....	140
8.5.3	总线分频器 .....	140
8.5.4	低功率位使用 .....	140
8.5.5	内部参考时钟 .....	141
8.5.6	外部参考时钟 .....	141
8.5.7	固定频率时钟 .....	141
8.6	初始化 / 应用报文 .....	141
8.6.1	MCG 模块初始化顺序 .....	142
8.6.2	MCG 模式切换 .....	143
8.6.3	校准内部参考时钟 (IRC) .....	154

## 第 9 章 模拟比较器 (S08ACMPV3)

9.1	介绍.....	157
9.1.1	ACMP 配置报文 .....	157
9.1.2	特性 .....	159
9.1.3	运行模式 .....	159
9.1.4	结构图 .....	160
9.2	外部信号描述.....	160
9.3	存储器映射 / 寄存器定义.....	161
9.3.1	ACMPx 状态和控制寄存器 (ACMPxSC) .....	161
9.4	功能描述.....	162

## 第 10 章 数模转换器 (S08ADC12V1)

10.1	介绍.....	163
10.1.1	模拟功率和接地信号名称 .....	163
10.1.2	信道分配 .....	163
10.1.3	替代时钟 .....	164
10.1.4	硬件触发 .....	165
10.1.5	温度传感器 .....	165
10.2.6	特性 .....	167

章节号	标题	页码
10.2.7 结构图 .....		167
10.3 外部信号描述 .....		169
10.3.1 模拟电源 (VDDAD) .....		169
10.3.2 模拟接地 (VSSAD) .....		169
10.3.3 参考电压高 (VREFH) .....		169
10.3.4 参考电压低 (VREFL) .....		169
10.3.5 模拟通道输入 (ADx) .....		169
10.4 寄存器定义 .....		170
10.4.1 状态和控制寄存器 1 (ADCSC1) .....		170
10.4.2 状态和控制寄存器 2 (ADCSC2) .....		171
10.4.3 数据结果高地址寄存器 (ADCRH) .....		172
10.4.4 比较值高地址寄存器 (ADCCVH) .....		172
10.4.5 比较值低地址寄存器 (ADCCVL) .....		173
10.4.6 .....		173
10.4.7 配置寄存器 (ADCCFG) .....		173
10.4.8 管脚控制寄存器 1 (APCTL1) .....		174
10.4.9 管脚控制寄存器 2 (APCTL2) .....		175
10.4.10 管脚控制寄存器 3 (APCTL3) .....		176
10.5 功能描述 .....		178
10.5.1 时钟选择和分频控制 .....		178
10.5.2 输入选择和管脚控制 .....		178
10.5.3 硬件触发 .....		178
10.5.4 转换控制 .....		179
10.5.5 自动比较功能 .....		181
10.5.6 MCU 等待模式运行 .....		181
10.5.7 MCU STOP3 模式运行 .....		181
10.5.8 MCU STOP1 和 STOP2 模式运行 .....		182
10.6 初始化报文 .....		183
10.6.1 ADC 模块初始化示例 .....		183
10.7 应用报文 .....		185
10.7.1 外部管脚和布线 .....		185
10.7.2 错误源 .....		186

## 第 11 章 IIC 模块 (S08IICV2)

11.1 介绍 .....		189
11.2.1 特性 .....		191
11.2.2 运行模式 .....		191
11.2.3 结构图 .....		192
11.3 外部信号描述 .....		192
11.3.1 SCL — 串行时钟线 .....		192
11.3.2 SDA — 串行数据线 .....		192

章节号	标题	页码
11.4	寄存器定义 . . . . .	193
11.4.1	IIC 地址寄存器 (IICA) . . . . .	193
11.4.2	11.3.2 IIC 分频器寄存器 (IICF) . . . . .	193
11.4.3	IIC 控制寄存器 (IICC1) . . . . .	196
11.4.4	IIC 状态寄存器 (IICS) . . . . .	196
11.4.5	IIC 数据 I/O 寄存器 (IICD) . . . . .	197
11.4.6	IIC 控制寄存器 2 (IICC2) . . . . .	198
11.5	功能描述 . . . . .	199
11.5.1	IIC 协议 . . . . .	199
11.5.2	10 位地址 . . . . .	202
11.5.3	通用呼叫地址 . . . . .	203
11.6	复位 . . . . .	203
11.7	中断 . . . . .	203
11.7.1	字节传输中断 . . . . .	203
11.7.2	地址检测中断 . . . . .	203
11.7.3	仲裁丢失中断 . . . . .	204
11.8	初始化 / 应用报文 . . . . .	205

## 第 12 章 飞思卡尔控制器局域网 (S08MSCANV1)

12.1	介绍 . . . . .	207
12.1.1	特性 . . . . .	209
12.1.2	运行模式 . . . . .	209
12.1.3	结构图 . . . . .	210
12.2	外部信号描述 . . . . .	210
12.2.1	RXCAN — CAN 接收器输入管脚 . . . . .	210
12.2.2	TXCAN — CAN T 发射器输出管脚 . . . . .	210
12.2.3	CAN 系统 . . . . .	210
12.3	寄存器定义 . . . . .	211
12.3.1	MSCAN 控制寄存器 0 (CANCTL0) . . . . .	211
12.3.2	控制寄存器 1 (CANCTL1) . . . . .	214
12.3.3	MSCAN 总线计时寄存器 0 (CANBTR0) . . . . .	215
12.3.4	MSCAN 总线计时寄存器 (CANBTR1) . . . . .	216
12.3.5	MSCAN 接收器中断使能寄存器 (CANRIER) . . . . .	219
12.3.6	MSCAN 发送器标志寄存器 (CANTFLG) . . . . .	220
12.3.7	MSCAN 发送器中断使能寄存器 (CANTIER) . . . . .	221
12.3.8	MSCAN Transmitter 发送器报文中止请求寄存器 (CANTARQ) . . . . .	222
12.3.9	MSCAN 发送器报文中止确认寄存器 (CANTAACK) . . . . .	223
12.3.10	MSCAN 发送缓冲器选择寄存器 (CANTBSEL) . . . . .	223
12.3.11	MSCAN 标识符验收控制寄存器 (CANIDAC) . . . . .	224
12.3.12	MSCAN 其他寄存器 (CANMISC) . . . . .	225
12.3.13	MSCAN 接收错误计数器 (CANRXERR) . . . . .	226

章节号	标题	页码
12.3.14MSCAN 发送错误计数器 (CANTXERR) .....	226	
12.3.15MSCAN 标识符接收寄存器 (CANIDAR0-7) .....	227	
12.3.16MSCAN 标识符掩码寄存器 (CANIDMR0-CANIDMR7) .....	228	
<b>12.4 报文存储模式.....</b>	<b>229</b>	
12.4.1 标识符寄存器 (IDR0-IDR3) .....	231	
12.4.2 标准标识符映射的 IDR0 – IDR3 .....	233	
12.4.3 数据段寄存器 (DSR0-7) .....	234	
12.4.4 数据长度寄存器 (DLR) .....	235	
12.4.5 发送缓冲器优先寄存器 (TBPR) .....	236	
12.4.6 时间标签寄存器 (TSRH – TSRL) .....	236	
<b>12.5 功能描述.....</b>	<b>237</b>	
12.5.1 概述 .....	237	
12.5.2 报文存储 .....	238	
12.5.3 标识符接收滤波器 .....	241	
12.5.4 运行模式 .....	247	
12.5.5 低功耗选项 .....	248	
12.5.6 复位初始化 .....	253	
12.5.7 中断 .....	253	
<b>12.6 初始化 / 应用信息 .....</b>	<b>255</b>	
12.6.1 MSCAN 初始化 .....	255	
12.6.2 总线脱离恢复 .....	255	

## 第 13 章 串行外围器件接口 (S08SPIV3)

<b>13.1 介绍.....</b>	<b>257</b>
13.1.1 特性 .....	259
13.1.2 结构图 .....	259
13.1.3 SPI 波特率生成 .....	261
<b>13.2 外部信号描述.....</b>	<b>262</b>
13.2.1 SPSCK — SPI 串行时钟 .....	262
13.2.2 MOSI — Master Data Out, Slave Data In .....	262
13.2.3 MISO — Master Data In, Slave Data Out .....	262
13.2.4 SS — 从选择 .....	262
<b>13.3 运行模式.....</b>	<b>262</b>
13.3.1 处于停止模式的 SPI .....	262
<b>13.4 寄存器定义.....</b>	<b>263</b>
13.4.1 SPI 控制寄存器 1 (SPIC1) .....	263
13.4.2 SPI 控制寄存器 2 (SPIC2) .....	264
13.4.3 SPI 波特率寄存器 (SPIBR) .....	265
13.4.4 SPI 状态寄存器 (SPIS) .....	266
13.4.5 SPI 数据寄存器 (SPID) .....	267
<b>13.5 功能描述.....</b>	<b>267</b>

章节号	标题	页码
13.5.1 SPI 时钟格式 .....		268
13.5.2 SPI 中断 .....		270
13.5.3 模式故障检测 .....		270

## 第 14 章 串行通信接口 (S08SCIV4)

14.1 介绍 .....		271
14.1.1 SCI2 配置信息 .....		271
14.1.2 特性 .....		273
14.1.3 运行模式 .....		273
14.1.4 结构图 .....		273
14.2 寄存器定义 .....		276
14.2.1 SCI 波特率寄存器 (SCIxBDH, SCIxBDL) .....		276
14.2.2 SCI 控制寄存器 1(SCIxC1) .....		277
14.2.3 SCI 控制寄存器 2 (SCIxC2) .....		278
14.2.4 SCI 状态寄存器 1 (SCIxS1) .....		279
14.2.5 SCI 状态寄存器 2 (SCIxS2) .....		280
14.2.6 SCI 控制寄存器 3 (SCIxC3) .....		282
14.2.7 SCI 数据寄存器 (SCIxD) .....		283
14.3 功能描述 .....		283
14.3.1 波特率生成 .....		283
14.3.2 发射器功能描述 .....		284
14.3.3 接收器功能描述 .....		285
14.3.4 中断和状态标记 .....		286
14.3.5 其他 SCI 功能 .....		287

## 第 15 章 实时计数器 (S08RTCV1)

15.1 简介 .....		289
15.1.1 RTC 时钟信号名称 .....		289
15.1.2 功能 .....		291
15.1.3 运行模式 .....		291
15.1.4 结构图 .....		292
15.2 外部信号描述 .....		292
15.3 寄存器定义 .....		292
15.3.1 RTC 状态和控制寄存器 (RTCSC) .....		293
15.3.2 RTC 计数器寄存器 (RTCCNT) .....		294
15.3.3 RTC 模数寄存器 (RTCMOD) .....		294
15.4 功能描述 .....		294
15.4.1 操作实例 .....		296
15.5 初始化 / 应用信息 .....		296

章节号	标题	页码
	<b>第 16 章 定时器脉冲宽度调节器 (S08TPMV3)</b>	
16.1 简介 . . . . .		299
16.1.1 功能 . . . . .		301
16.1.2 运行模式 . . . . .		301
16.1.3 结构图 . . . . .		302
16.2 信号描述 . . . . .		304
16.2.1 详细信号描述 . . . . .		304
16.3 寄存器定义 . . . . .		308
16.3.1 TPM 状态和控制寄存器 (TPMxSC) . . . . .		308
16.3.2 计数器的寄存器 (TPMxCNTH:TPMxCNTL) . . . . .		309
16.3.3 TPM 计数器模数寄存器 (TPMxMODH:TPMxMODL) . . . . .		310
16.3.4 TPM 通道 n 状态和控制寄存器 (TPMxCnSC) . . . . .		311
16.3.5 TPM 通道值寄存器 (TPMxCnVH:TPMxCnVL) . . . . .		312
16.4 功能描述 . . . . .		314
16.4.1 计数器 . . . . .		314
16.4.2 通道模式选择 . . . . .		316
16.5 复位概述 . . . . .		319
16.5.1 概况 . . . . .		319
16.5.2 复位操作介绍 . . . . .		319
16.6 中断 . . . . .		319
16.6.1 General . . . . .		319
16.6.2 中断操作描述 . . . . .		319
	<b>第 17 章 开发支持</b>	
17.1 介绍 . . . . .		323
17.1.1 强制激活背景调试 . . . . .		323
17.1.2 特性 . . . . .		324
17.2 背景调试控制器 (BDC) . . . . .		325
17.2.1 BKGD 管脚描述 . . . . .		325
17.2.2 通信详细介绍 . . . . .		326
17.2.3 BDC 命令 . . . . .		328
17.2.4 BDC 硬件断点 . . . . .		330
17.3 片上调试系统 (DBG) . . . . .		331
17.3.1 比较器 A 和 B . . . . .		331
17.3.2 总线捕获信息和 FIFO 操作 . . . . .		331
17.3.3 流变化信息 . . . . .		332
17.3.4 标记 vs. 强制断点和触发器 . . . . .		332
17.3.5 触发模式 . . . . .		333
17.3.6 硬件断点 . . . . .		334
17.4 寄存器定义 . . . . .		334

章节号	标题	页码
17.4.1	BDC 寄存器和控制位 .....	334
17.4.2	系统背景调试强制复位寄存器 (SBDFR) .....	336
17.4.3	DBG 寄存器和控制位 .....	336

## 附录 A 电气特征

A.1	简介 .....	342
A.2	参数分类 .....	342
A.3	绝对最大额定值 .....	342
A.4	热特性 .....	343
A.5	ESD 保护和抗闭锁方法 .....	344
A.6	DC 特性 .....	345
A.7	电源电流特性 .....	347
A.8	模拟比较器 (ACMP) 电气特性 .....	348
A.9	ADC 特性 .....	349
A.10	外部振荡器 (XOSC) 特性 .....	352
A.11	MCG 规范 .....	353
A.12	AC 特性 .....	354
	A.12.1 控制时序 .....	355
	A.12.2 定时器 /PWM .....	356
	A.12.3 MSCAN .....	357
	A.12.4 SPI .....	358
A.13	闪存和 EEPROM .....	361
A.14	EMC 性能 .....	362
	A.14.1 辐射放射性 .....	362

## 附录 B 定时器脉宽调制器 (TPMV2)

B.1	介绍 .....	364
B.2	特性 .....	364
B.3	结构图 .....	365
B.4	外部信号描述 .....	366
	B.4.1 外部 TPM 时钟源 .....	366
	B.4.2 TPMxCHn — TPMx 通道 n I/O 管脚 .....	366
B.5	寄存器定义 .....	366
	B.5.1 定时器状态和控制寄存器 (TPMxSC) .....	367
	B.5.2 定时器计数器寄存器 (TPMxCNTH:TPMxCNTL) .....	368
	B.5.3 定时器计数器模量寄存器 (TPMxMODH:TPMxMODL) .....	369
	B.5.4 定时器通道 n 状态和控制寄存器 (TPMxCnSC) .....	370
	B.5.5 TPM 通道值寄存器 (TPMxCnVH:TPMxCnVL) .....	371
B.6	功能介绍 .....	372
	B.6.1 计数器 .....	372

章节号	标题	页码
B.6.2	通道模式选择 .....	373
B.6.3	中央对齐 PWM 模式 .....	374
B.7	TPM 中断 .....	376
B.7.1	清除定时器中断标记 .....	376
B.7.2	定时器溢出中断描述 .....	376
B.7.3	通道事件中断描述 .....	376
B.7.4	PWM 占空比结束事件 .....	377

## 附录 C 订购信息和机械图

C.1	订购信息 .....	378
C.1.1	MC9S08DZ60 Series 设备 .....	378
C.2	机械图 .....	378



# 第 1 章

## 器件概述

MC9S08DZ60 系列器件主要用于需要融合 CAN (Controller Area Network) 网络和内嵌的 EEPROM 的应用中，它有助于帮助用户降低成本，增强产品的性能并提高产品的质量。

### 1.1 MC9S08DZ60 系列器件

本数据手册介绍了以下 MC9S08DZ60 系列微控制器，表 1-1 列举了它们的特性。

- MC9S08DZ60
- MC9S08DZ48
- MC9S08DZ32
- MC9S08DZ16

表 1-1. MC9S08DZ60 系列产品的特性（按 MCU 和管脚数量分）

特性	MC9S08DZ60			MC9S08DZ48			MC9S08DZ32			MC9S08DZ16	
Flash 大小 (字节)	60032			49152			33792			16896	
RAM 大小 (字节)	4096			3072			2048			1024	
EEPROM 大小 (字节)	2048			1536			1024			512	
管脚数量	64	48	32	64	48	32	64	48	32	48	32
ACMP1	是										
ACMP2	是	是 <sup>1</sup>	no	是	是 <sup>1</sup>	no	是	是 <sup>1</sup>	no	是 <sup>1</sup>	no
ADC 通道数	24	16	10	24	16	10	24	16	10	16	10
DBG	是										
IIC	是										
IRQ	是										
MCG	是										
MSCAN	是										
RTC	是										
SCI1	是										
SCI2	是										
SPI	是										
TPM1 通道数	6	6	4	6	6	4	6	6	4	6	4
TPM2 通道数	2										
XOSC	是										
COP Watchdog	是										

<sup>1</sup> ACMP2O 不可用。

## 1.2 MCU 结构图

图 1-1 为 MC9S08DZ60 系列产品的系统结构图。

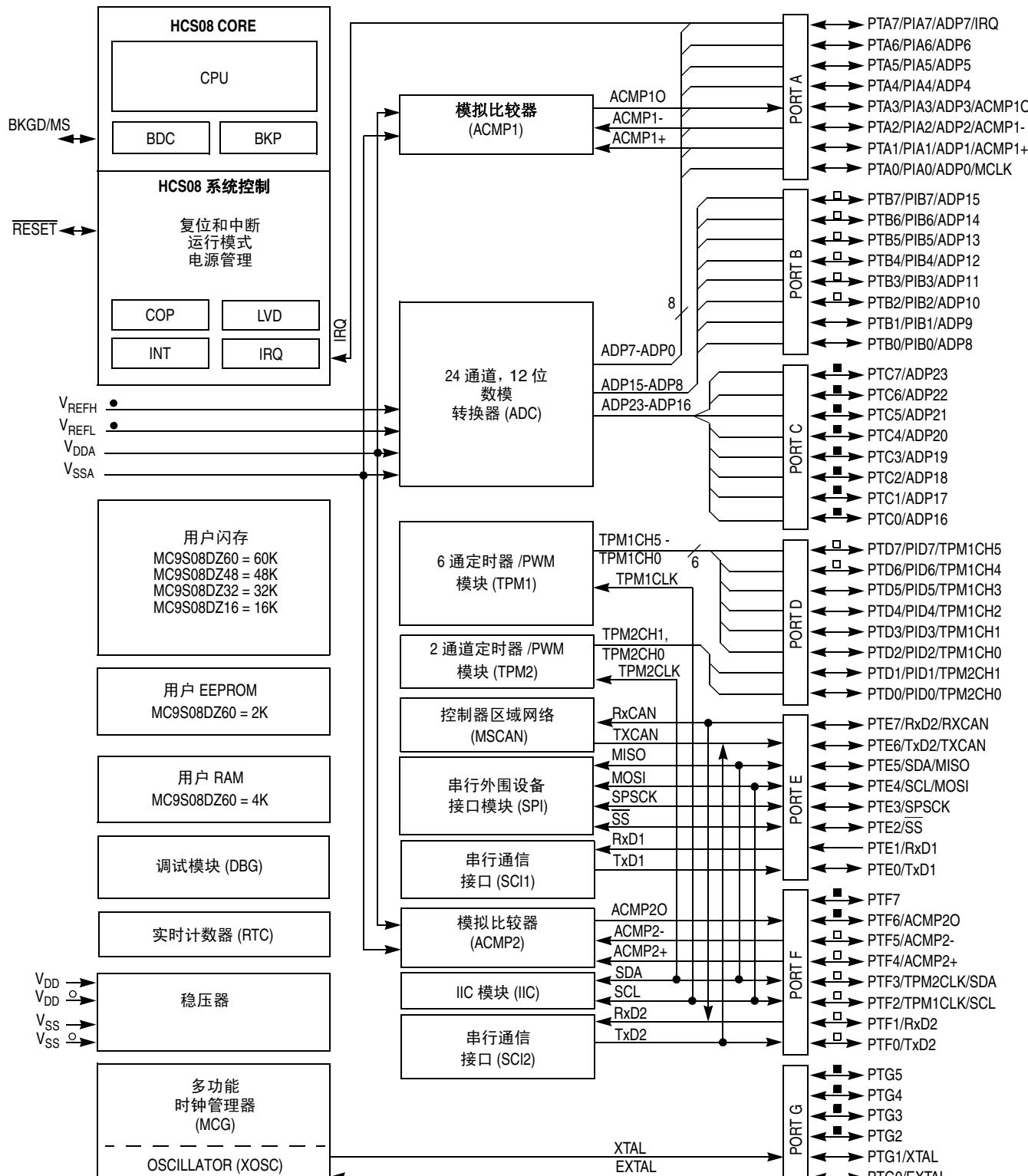


图 1-1. MC9S08DZ60 结构图

表 1-2 为芯片模块的功能版本。

表 1-2. 模块版本

模块	版本
中央处理器 (CPU)	3
多功能时钟生成器 (MCG)	1
模拟比较器 (ACMP)	3
模数转换器 (ADC)	1
IIC 总线 (IIC)	2
飞思卡尔的 CANN (MSCAN)	1
串行外围接口 (SPI)	3
串行通信接口 (SCI)	4
实时计数器 (RTC)	1
定时器脉宽调制器 (TPM)	3 <sup>1</sup>
调试模块 (DBG)	2

<sup>1</sup> 3M05C 和更早版本的掩码有 TPM 第 2 版。

## 1.3 系统时钟分配

图 1-2 为简化的时钟连接结构图。MCU 的某些模块的时钟输入可选。到各模块的时钟输入用于驱动该模块的功能。

下面列出了本 MCU 中使用的时钟：

- BUSCLK — 总线频率始终为 MCGOUT 的一半
- LPO — 独立的 1 kHz 时钟，可以作为 COP 和 RTC 模块的时钟源。
- MCGOUT — MCG 的主输出，为总线频率的两倍。
- MCGLCLK — 在 BUSCLK 被配置为以很低的频率运行的系统中，开发工具可以选择这一时钟源来加快 BDC 通信。
- MCGERCLK — 外部参考时钟，可用作 RTC 时钟源。它还可以用作 ADC 和 MSCAN 的备用时钟。
- MCGIRCLK — 内部参考时钟，可用作 RTC 时钟源。
- MCGFFCLK — 固定频率时钟，可用作 TPM1 和 TPM2 的时钟源。
- TPM1CLK — TPM1 的外部输入时钟源。
- TPM2CLK — TPM2 的外部输入时钟源。

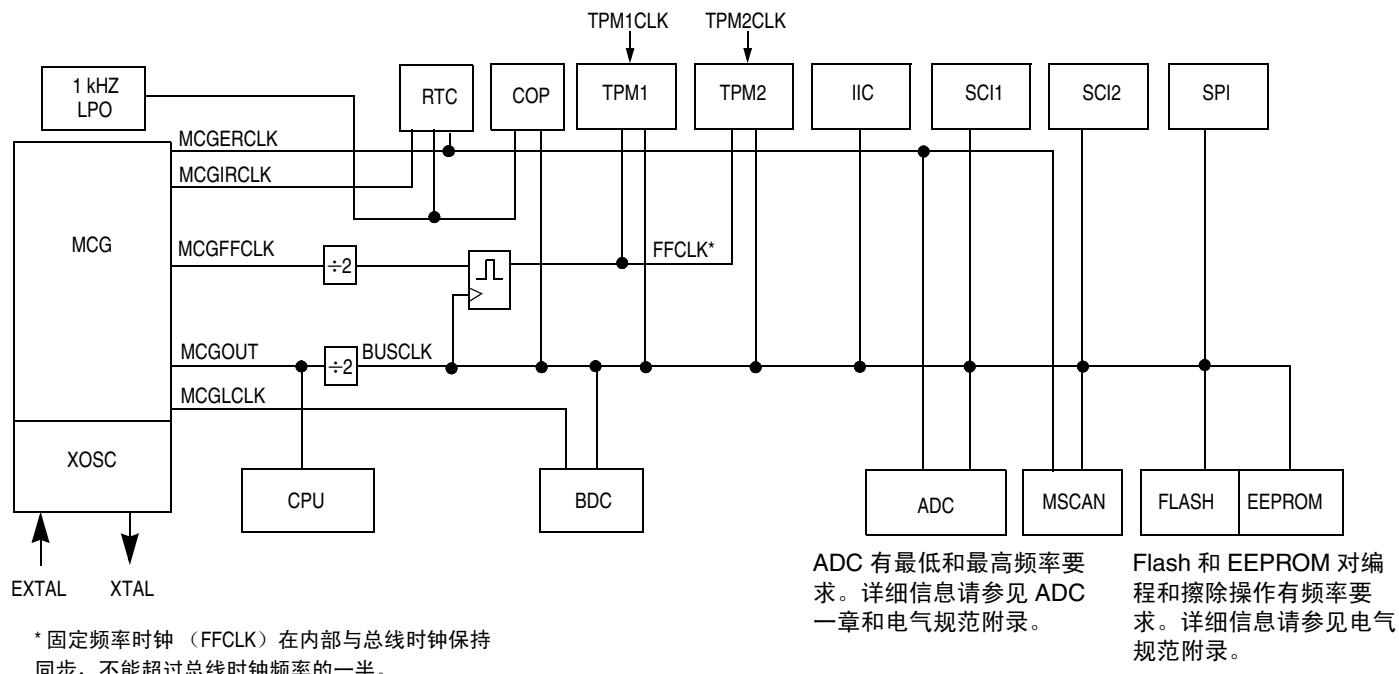


图 1-2. MC9S08DZ60 系统时钟分配图



## 第 2 章 管脚和连接

本章描述连接到各封装管脚的信号，内容包括管脚布局图、建议的系统连接并对信号进行了详细地描述。

### 2.1 器件管脚分配

本节介绍了 MC9S08DZ60 系列 MCU 各种封装的管脚分配状况。

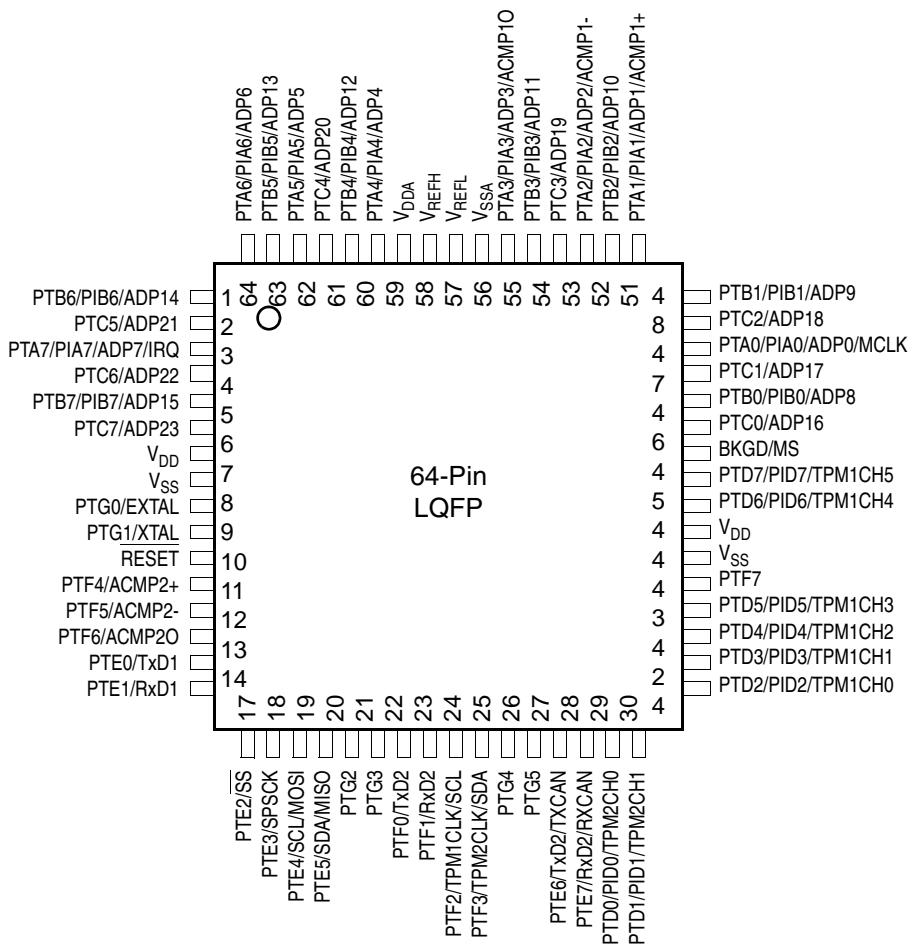
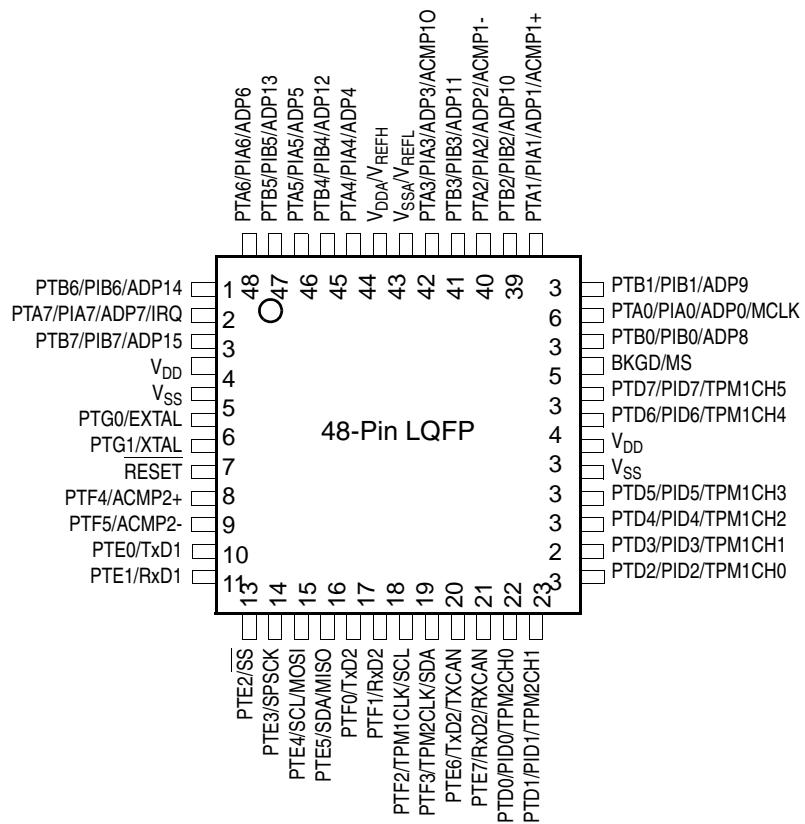
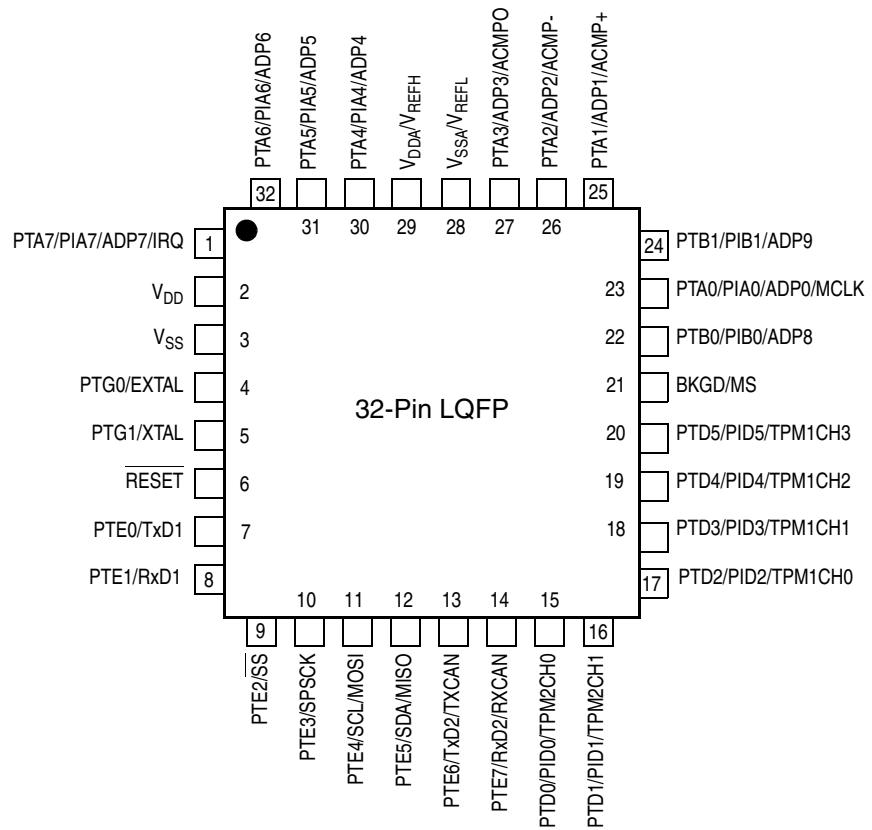


图 2-1. 64 管脚 LQFP



V<sub>REFH</sub> 和 V<sub>REFL</sub> 在内部分别连接到 V<sub>DDA</sub> 和 V<sub>SSA</sub>。

图 2-2. 48 管脚 LQFP



VREFH 和 VREFL 在内部分别连接到 VDDA 和 VSSA。

图 2-3. 32 管脚 LQFP

## 2.2 推荐的系统连接

图 2-4 为 MC9S08DZ60 系列产品在应用系统中的通用管脚连接。

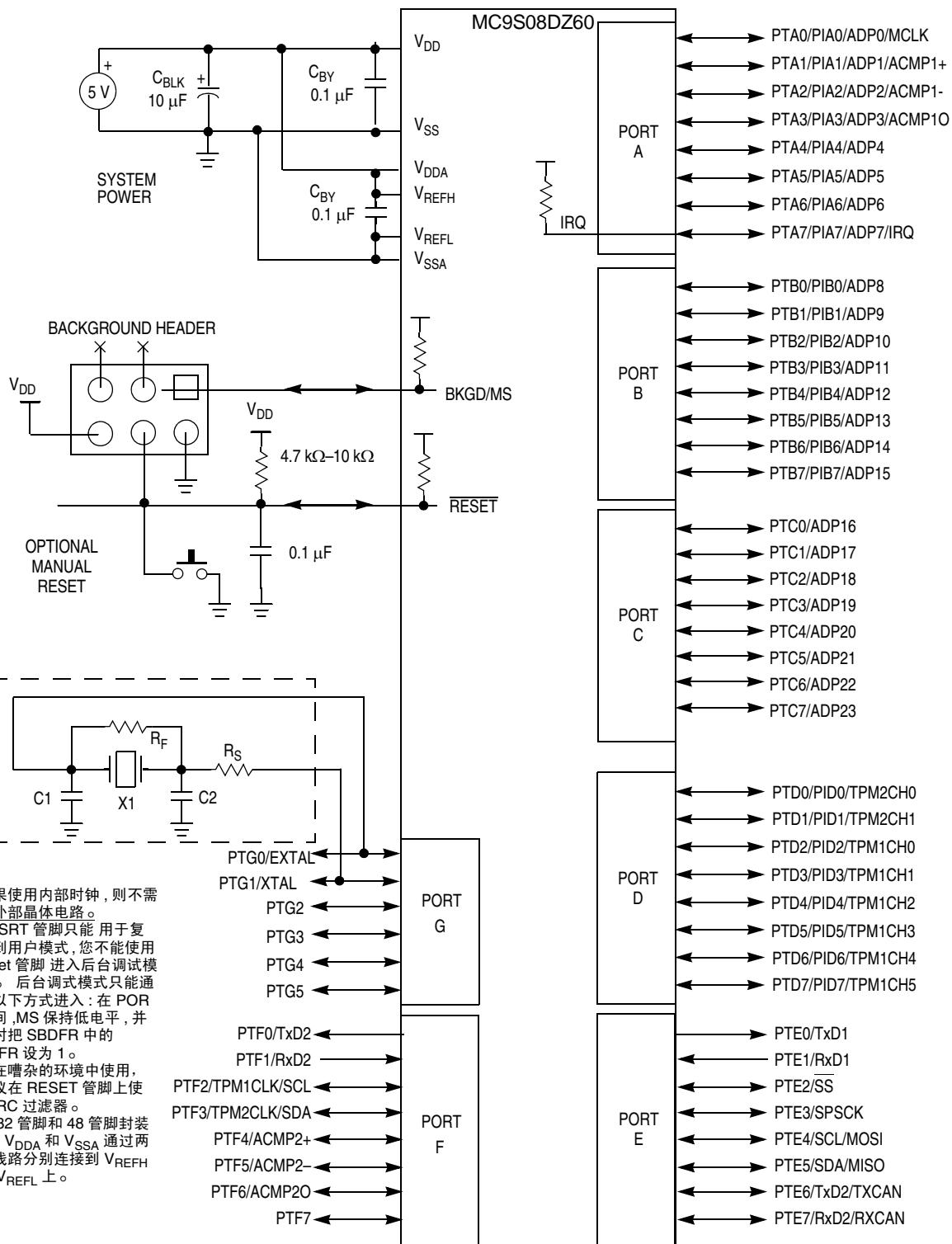


图 2-4. 基本系统连接图 (64 管脚封装)

## 2.2.1 电源

$V_{DD}$  和  $V_{SS}$  是 MCU 基本的电源管脚。该电源为所有 I/O 缓冲器电路和一个内部稳压器供电。内部稳压器为 CPU 及 MCU 的其他内部电路提供经过稳压的低电压电源。

通常，应用系统在电源管脚上需要安装两个独立的电容器。其中一个为大容量电解电容器（如  $10\mu F$  钽电容器）为整个系统提供大容量电荷存储。同时应在离 MCU 电源管脚尽可能近的地方安装一个  $0.1\mu F$  的陶瓷旁路电容器来抑制高频噪音。MC9S08DZ60 系列有两个  $V_{DD}$  管脚（32 管脚封装除外），每个管脚都必须有一个旁路电容器以实现最有效的噪音抑制。

$V_{DDA}$  和  $V_{SSA}$  是 MCU 的模拟电源管脚。该管脚引入的电源为 ADC 模块供电。我们应在离 MCU 电源管脚尽可能近的地方安装一个  $0.1\mu F$  陶瓷旁路电容器来抑制高频噪音。

## 2.2.2 振荡器

复位完成后，MCU 立即开始使用由 MCG（多功能时钟生成器）模块提供的内部时钟。关于 MCG 的更详尽信息，请参见第 8 章，“[多功能时钟发生器（S08MCGV1）](#)”。

本 MCU 中的振荡器（XOSC）为皮尔斯（Pierce）振荡器，可以支持晶体和陶瓷谐振器。除了晶体或陶瓷谐振器外，我们还可以将一个外部振荡器连接到 EXTAL 输入管脚上。

如图 2-4 所示， $R_S$ （如果使用了的话）和  $R_F$  必须采用低感电阻器，如碳膜电阻器。而不能采用感应系数过高的线绕和金属薄膜电阻器。 $C_1$  和  $C_2$  必须使用专为高频应用设计的高质量陶瓷电容器。

$R_F$  用来提供偏置路径用于在晶体启动过程中将 EXTAL 输入保持在线性范围内。它的值并不是在所有情况下都非常关键。一般系统采用  $1M \sim 10M$  之间的  $R_F$ 。过高的阻抗对湿度太敏感，而过低的阻抗会减少增益并（在一些极端情况下）导致无法正常启动。

$C_1$  和  $C_2$  一般采用  $5pF \sim 25pF$  的电容，并且必须满足匹配特定晶体或谐振器的要求。在选择  $C_1$  和  $C_2$  时必须考虑印刷电路板（PCB）的电容和 MCU 管脚的电容。晶体生产商一般都规定了一个负载电容—— $C_1$  和  $C_2$ （二者的尺寸通常是相同的）的系列组合。按照一次近似原则，我们应使用  $10 pF$  作为每个振荡器管脚（EXTAL 和 XTAL）的管脚和 PCB 总电容的估计值。

## 2.2.3 RESET（复位）

RESET 是一个专用管脚，带有内置的上拉器件。它有输入电压迟滞、大电流输出驱动器但没有输出斜率控制。由于存在内部加电复位电路和低压复位电路，因此在一般情况下不必使用外部复位电路。该管脚通常连接到标准的 6 脚后台调试接头，以保证开发系统可以直接复位 MCU 系统。如果需要，我们可以增加一个到地线的简单开关（拉低复位管脚以强制进行复位）来实现手动外部复位。

在任何情况下触发复位时（不管是外部信号还是内部系统），RESET 管脚都会下拉约 34 个总线周期。复位电路会解析复位原因并且在系统复位状态寄存器（SRS）中设置一个相应的位来记录这一原因。

## 2.2.4 后台调试和模式选择

**BKGD/MS** 管脚在进行复位的过程中作为模式选择功能管脚。在复位信号上升沿后，该管脚立即用作后台调试管脚而用于后台调试通信。当作为后台调试或模式选择功能管脚时，该管脚包括一个内部上拉器件、输入电压滞后、标准的输出驱动器而没有输出斜率控制功能。

如果没有任何设备连接到该管脚上，**MCU** 将在复位的上升沿进入正常操作模式。如果有一个调试系统连接到 6 脚标准后台调试头上，它将在复位的上升沿将 **BKGD** 保持在低位，从而强迫 **MCU** 进入活动后台调试模式。

**BKGD/MS** 管脚主要用于后台调试控制器（**BDC**）通信。这一通信过程中使用一种定制的协议，该协议在每个比特时间周期内使用目标 **MCU** 的 **BDC** 时钟的 16 个时钟周期。目标 **MCU** 的 **BDC** 时钟可以和总线时钟速率一样快，因此在任何情况下都不应将大电容器件连接到 **BKGD/MS** 管脚（这样会干扰后台调试串行通信）。

虽然 **BKGD/MS** 管脚是一种准开漏管脚，但后台调试通信协议可以提供瞬间、主动驱动的高速脉冲来确保快速上升沿。电缆上的小电容和内部上拉器件参数几乎不会影响 **BKGD/MS** 管脚上的上升和下降沿时间。

## 2.2.5 ADC 参考管脚 ( $V_{REFH}$ , $V_{REFL}$ )

$V_{REFH}$  和  $V_{REFL}$  管脚分别是 ADC 模块的电压参考高端和电压参考低端的输入管脚。

## 2.2.6 通用 I/O 和外围设备端口

MC9S08DZ60 系列 **MCU** 最多可提供 53 个通用 I/O 管脚和 1 个专用输入管脚。这些管脚和片上外围设备（定时器、串行 I/O、ADC、MSCAN 等）共享。

当一个端口管脚被配置为通用输出时，或者某外围设备使用该端口管脚作为输出时，通过软件可以在两个驱动强度中选择一种并同时启用或禁用斜率控制。当一个端口管脚被配置为通用输入或某外围设备使用该端口管脚作为输入时，软件可以选择一个上拉器件。复位完成后，所有这些管脚被立即配置为高阻抗通用输入（内部上拉器件被禁用）。

当一个片上外围系统控制管脚时，即使该外围模块通过控制该管脚的输出缓冲器的启用来控制管脚方向时，仍然由数据方向控制位决定从端口数据寄存器中读取的内容。如何控制这些管脚作为通用 I/O 管脚的详尽信息，请参见[第 6 章，“并行输入 / 输出控制”](#)。

### 注意

为了避免悬空输入管脚消耗额外的电流，应用程序中的复位初始化程序应该启用片内上拉器件或将未使用或未绑定的管脚的方向设置为输出以确保他们不会悬空。

表 2-1. 管脚可用性 (按封装管脚数)

管脚编号			<-- 最低 优先级 --> 最高			
64	48	32	端口管脚 / 中断		Alt 1	Alt 2
1	1	—	PTB6	PIB6	ADP14	
2	—	—	PTC5		ADP21	
3	2	1	PTA7	PIA7	ADP7	IRQ
4	—	—	PTC6		ADP22	
5	3	—	PTB7	PIB7	ADP15	
6	—	—	PTC7		ADP23	
7	4	2			$V_{DD}$	
8	5	3			$V_{SS}$	
9	6	4	PTG0		EXTAL	
10	7	5	PTG1		XTAL	
11	8	6			RESET	
12	9	—	PTF4		ACMP2+	
13	10	—	PTF5		ACMP2-	
14	—	—	PTF6		ACMP2O	
15	11	7	PTE0		TxD1	
16	12	8	PTE1		RxD1	
17	13	9	PTE2		SS	
18	14	10	PTE3		SPSCK	
19	15	11	PTE4		SCL <sup>3</sup>	MOSI
20	16	12	PTE5		SDA <sup>3</sup>	MISO
21	—	—	PTG2			
22	—	—	PTG3			
23	17	—	PTF0		TxD2 <sup>4</sup>	
24	18	—	PTF1		RxD2 <sup>4</sup>	
25	19	—	PTF2		TPM1CLK	SCL <sup>3</sup>
26	20	—	PTF3		TPM2CLK	SDA <sup>3</sup>
27	—	—	PTG4			
28	—	—	PTG5			
29	21	13	PTE6		TxD2 <sup>4</sup>	TXCAN
30	22	14	PTE7		RxD2 <sup>4</sup>	RxCAN
31	23	15	PTD0	PID0		TPM2CH0
32	24	16	PTD1	PID1		TPM2CH1

管脚编号			<-- 最低 优先级 --> 最高			
64	48	32	端口管脚 / 中断		Alt 1	Alt 2
33	25	17	PTD2	PID2		$V_{DD}$
34	26	18	PTD3	PID3		$V_{DD}$
35	27	19	PTD4	PID4		$V_{DD}$
36	28	20	PTD5	PID5		$V_{DD}$
37	—	—	PTF7			
38	29	—				$V_{SS}$
39	30	—				$V_{DD}$
40	31	—	PTD6	PID6		$V_{DD}$
41	32	—	PTD7	PID7		$V_{DD}$
42	33	21			BKGD	MS
43	—	—	PTC0		ADP16	
44	34	22	PTB0	PIB0	ADP8	
45	—	—	PTC1		ADP17	
46	35	23	PTA0	PIA0	ADP0	MCLK
47	—	—	PTC2		ADP18	
48	36	24	PTB1	PIB1	ADP9	
49	37	25	PTA1	PIA1	ADP1 <sup>1</sup>	ACMP1+ <sup>1</sup>
50	38	—	PTB2	PIB2	ADP10	
51	39	26	PTA2	PIA2	ADP2	ACMP1- <sup>1</sup>
52	—	—	PTC3		ADP19	
53	40	—	PTB3	PIB3	ADP11	
54	41	27	PTA3	PIA3	ADP3	ACMP1O
55	42	28				$V_{SSA}$
56						$V_{REFL}$
57	43	29				$V_{REFH}$
58						$V_{DDA}$
59	44	30	PTA4	PIA4	ADP4	
60	45	—	PTB4	PIB4	ADP12	
61	—	—	PTC4		ADP20	
62	46	31	PTA5	PIA5	ADP5	
63	47	—	PTB5	PIB5	ADP13	
64	48	32	PTA6	PIA6	ADP6	

- 如果这两个模拟模块都被启用，他们都将可以访问该管脚。
- 管脚不包含到  $V_{DD}$  的嵌位二极管，因此不应用在高于  $V_{DD}$ 。内部上拉器件被启用时在该管脚上测得的电压可能会低达  $V_{DD} - 0.7$  V。连接到该管脚上的内部门被拉到  $V_{DD}$  上。
- IIC 模块管脚可以通过 SOPT1 寄存器中的 IICPS 位进行重定位。缺省复位位置在 PTF2 和 PTF3 上。
- SCI2 模块管脚可以通过 SOPT1 寄存器中的 SCI2PS 位进行重定位。缺省复位位置在 PTF0 和 PTF1 上。



# 第 3 章

## 操作模式

### 3.1 简介

本章介绍 MC9S08DZ60 系列产品的操作模式，同时描述了如何进入各种模式、如何从各种模式中退出及各种模式下可提供的功能。

### 3.2 特性

- 主动后台模式：用于代码开发
- 等待模式：CPU 关闭以省电；系统时钟正常运行，内部稳压器正常工作
- 停止模式：系统时钟被关闭，内部稳压器处于待机状态
  - Stop3 — 所有内部电路都接通电源以实现快速恢复
  - Stop2 — 内部电路的部分电源被关闭；RAM 内容被保留

### 3.3 运行模式

这是 MC9S08DZ60 系列产品的正常操作模式。当 BKGD/MS 管脚位于复位的上升边最高位置时选择该模式。在这种模式下，CPU 执行内部存储器中的代码。代码在复位完成后运行，并且从内存 0xFFFFE - 0xFFFF 上获取其起始地址。

### 3.4 主动后台模式

主动后台模式功能通过 HCS08 内核中的后台调试控制器（BDC）进行管理。在软件开发过程中，BDC 与片上调试模块（DBG）一起用于分析 MCU 的运行情况。

进入主动后台模式的方式有以下五种：

- 当处于复位的上升沿时，BKGD/MS 脚置于低电平；
- 通过 BKGD/MS 脚收到 BACKGROUND 命令时；
- 执行 BGND 指令时；
- 遇到 BDC 断点时；
- 遇到 DBG 断点时。

进入主动后台模式后，CPU 保持挂起状态，等待串行后台命令而不执行来自用户应用程序的指令。

后台命令有两种类型：

- 非中断型命令，被定义为可在用户程序运行时发出。非中断型命令可在 MCU 处于运行模式时通过 BKGD/MS 管脚发出；非中断型命令也可以在 MCU 处于主动后台模式时执行。  
非中断型命令包括：
  - 内存访问命令
  - 带状态内存访问命令
  - BDC 寄存器访问命令
  - 后台命令
- 主动后台命令只能在 MCU 处于主动后台模式时执行。主动后台命令主要用于执行以下操作：
  - 读或写 CPU 寄存器
  - 在特定时间跟踪一个用户程序指令
  - 退出主动后台模式，返回到用户应用程序（GO）

主动后台模式用于在 MCU 第一次以运行模式运行前将 Bootloader 或用户应用程序写入到 Flash 程序存储器中。MC9S08DZ60 系列产品从飞思卡尔工厂运出时，除非特别说明，Flash 程序存储器在缺省被擦除，以确保在 Flash 首次被编程前不会有程序在运行模式下被执行。主动后台模式可用于擦除或重新编程先前已编程的 Flash。

关于主动后台模式的详尽信息，请参见 [Development Support](#) 一章。

## 3.5 等待模式

等待模式通过执行 WAIT 指令进入。在执行 WAIT 指令后，CPU 进入无时钟的低功耗状态。CPU 进入等待模式后，CCR 中的 I 位被清除，进而使能中断操作。发生中断请求后，CPU 退出等待模式并执行恢复处理，先开始执行堆栈中的中断业务程序。

MCU 处于等待模式时，后台调试命令的使用受限。MCU 处于等待模式时，只有后台命令和带状态内存访问命令可用。带状态内存访问命令虽然不允许内存访问，但它们会上报错误，指出 MCU 处于停止或等待模式。可以使用后台命令将 MCU 从等待模式中唤醒并进入主动后台模式。

## 3.6 停止模式

在 SOPT1 寄存器中设置了 STOPE 位时，执行 STOP 指令后会进入停止模式。在停止模式下，所有内部时钟都被暂停。我们可对 MCG 模块进行适当设置，使参考时钟保持运行。更详尽信息请参见 [第 8 章，“多功能时钟发生器（S08MCGV1）”](#)。

**表 3-1** 列出了影响停止模式选择的所有控制位及各种条件下选择的模式。被选择的模式在执行一个 STOP 指令后进入。

表 3-1. 停止模式选择

STOPE	ENBDM <sup>1</sup>	LVDE	LVDSE	PPDC	停止模式
0	x		x	x	停止模式被禁用；如果执行了 STOP 指令，则进行非法 Opcode 代码复位
1	1		x	x	Stop3 模式，BDM 被启用 <sup>2</sup>
1	0	两个位都必须为 1		x	Stop3 模式，电压调节器处于活动状态
1	0	其中一个位为 0		0	Stop3 模式
1	0	其中一个位为 0		1	Stop2 模式

<sup>1</sup> ENBDM 位于 BDCSCR（只能通过 BDC 命令访问）中。详细信息请参见第 17.4.1.1 部分“BDC 状态和控制寄存器（BDCSCR）”。

<sup>2</sup> 处于 Stop3 模式而且 BDM 被启用时，因为启用了内部时钟，SIDD 将接近 RIDD 水平。

### 3.6.1 Stop3 模式

Stop3 模式通过表 3-1 所述条件下执行 STOP 指令进入。所有此时的内部寄存器和逻辑、RAM 内容和 I/O 管脚状态都被维持。

从 Stop3 模式中退出的操作通过 RESET 管脚或异步中断脚实现。这些异步中断脚包括 IRQ、PIA0 ~ PIA7、PIB0 ~ PIB7 和 PID0 ~ PID7。从 Stop3 模式中退出的操作也可以通过低压检测（LVD）复位、低压警告（LWV）中断、ADC 转换完成中断、实时时钟（RTC）中断、MSCAN 唤醒中断或 SCI 接收器中断完成。

如果通过 RESET 脚的方式退出 Stop3 模式，MCU 将复位，操作将在获取复位向量后恢复。如果通过中断的方式退出，MCU 将获取相应的中断向量。

#### 3.6.1.1 在 Stop3 模式中启用 LVD

在电源电压下降到 LVD 电压以下时，LVD 系统可以生成一个中断或复位。在 CPU 执行 STOP 指令时，如果 LVD 在停止模式下被启用（SPMSC1 中的 LVDE 和 LVDSE 位均被设置），那么内部稳压器在停止模式下将继续保持活动状态。

要使 ADC 正常运行，LVD 必须在进入 Stop3 时处于启用状态。

#### 3.6.1.2 在 Stop3 模式中启用活动 BDM

如果 BDCSCR 中设置了 ENBDM，将启用从运行模式进入主动后台模式的操作。该寄存器在第 17 章，“开发支持”中有详细描述。如果 CPU 执行 STOP 指令时设置了 ENBDM，连接到后台调试逻辑的系统时钟将在 MCU 进入停止模式时继续保持活动状态。因此，这种情况下后台调试通信仍可进行。此外，内部稳压器不会进入低功耗待机状态，而保持正常工作。

大多数后台命令在停止模式不能使用。带状态内存访问命令不允许内存访问，但它们会上报错误，指出 MCU 处于停止或等待模式。可以使用后台命令将 MCU 从停止模式中唤醒并进入主动后台模式（如果 ENBDM 位已设置）。进入后台调试模式后，所有后台命令都可以使用。

### 3.6.2 Stop2 模式

Stop2 模式通过表 3-1 所示的情况下执行 STOP 指令进入。除 RAM 外，MCU 的大部分内部电路在 Stop2 模式下处于断电状态。在进入 Stop2 模式后，所有 I/O 管脚控制信号被锁定，以确保管脚可以在 Stop2 模式下保持原来的状态。

从 Stop2 模式中退出的操作通过输入有效 RESET 信号完成。只有在 3M05C 或更老的掩码集中，您可以通过输入 PTA7/ADP7/IRQ 中断信号来退出 Stop2。

#### 注意

只有在 3M05C 或更老的掩码集中，PTA7/ADP7/IRQ 是低电平唤醒，因此在执行 STOP 指令前必须配置为输入，以避免从 Stop2 中立即退出。如果 PTA7/ADP7/IRQ 被配置为高驱动输出，那么它可以禁止唤醒功能。为了在 Stop2 模式下最大限度地降低功耗，该管脚在被配置为输入时不应保持开路（启用内部上拉器件；或连接外部上拉 / 下拉器件；或将管脚设置为输出）。

此外，实时时钟计数器（RTC）也可以从 Stop2 模式下唤醒 MCU（如果已启用）。

MCU 从 Stop2 模式中唤醒后，启动过程和加电复位（POR）相同：

- 所有模块控制寄存器和状态寄存器被复位
- LVD 复位功能启用；如果  $V_{DD}$  低于 LVD 跳变点（由于 POR 选择的低跳变点），MCU 仍处于复位状态
- CPU 读取复位向量

并且，在从 Stop2 模式中唤醒后，SPMSC2 中的 PPDF 也会被设置用于将用户代码引导到 Stop2 恢复程序中。PPDF 仍保持有效且 I/O 管脚状态被锁定，直到 1 被写入到 SPMSC2 中的 PPDACK 中。

为了在进入 Stop2 之前保持被设置为通用 I/O 管脚的 I/O 状态，用户必须将 I/O 端口寄存器（保存在 RAM 中）的内容恢复到端口寄存器中，然后再写入到 PPDACK 位中。如果端口寄存器在写入到 PPDACK 中之前没有从 RAM 中恢复，那么在写入 PPDACK 时该管脚将切换到复位状态。

对于配置为外围 I/O 的管脚，用户在写入 PPDACK 位之前必须重新配置连接到该管脚的外围模块。如果外围模块在写入 PPDACK 之前没有启用，那么在 I/O 锁定被打开时，该管脚将由相关的端口控制寄存器控制。

### 3.6.3 停止模式中的片上外围模块

当 MCU 进入任何停止模式时，连接到内部外围模块的系统时钟被停止。即使在异常情况下（ENBDM = 1）（连接到后台调试逻辑的时钟继续运行），到外围系统的时钟也将被停止以降低功耗。有关停止模式下系统的详细信息，请参见 3.6.2，“Stop2 模式”和 3.6.1，“Stop3 模式”。

表 3-2. 停止模式

外围设备	模式	
	Stop2	Stop3
CPU	关闭	待机
RAM	待机	待机
Flash/EEPROM	关闭	待机
并行端口寄存器	关闭	待机
ACMP	关闭	关闭
ADC	关闭	可选择开启 <sup>1</sup>
IIC	关闭	待机
MCG	关闭	可选择开启 <sup>2</sup>
MSCAN	关闭	待机
RTC	可选择开启 <sup>3</sup>	可选择开启 <sup>3</sup>
SCI	关闭	待机
SPI	关闭	待机
TPM	关闭	待机
电压调节器	关闭	可选择开启 <sup>4</sup>
XOSC	关闭	可选择开启 <sup>5</sup>
I/O 管脚	状态被保持	状态被保持
BDM	关闭 <sup>6</sup>	可选择开启
LVD/LVW	关闭 <sup>7</sup>	可选择开启

<sup>1</sup> 要求启用异步 ADC 时钟和 LVD，否则为待机。

<sup>2</sup> MCGC1 中设置 IRCLKEN 和 IREFSTEN，否则为待机。

<sup>3</sup> 要求启用 RTC，否则为待机。

<sup>4</sup> 要求启用 LVD 或 BDC。

<sup>5</sup> MCGC2 中设置 ERCLKEN 和 EREFSTEN，否则为待机。在高频率范围（MCGC2 中设置 RANGE）还要求在 Stop3 中启用 LVD。

<sup>6</sup> 如果进入 Stop2 模式时设置了 ENBDM，MCU 实际上会进入 Stop3 模式。

<sup>7</sup> 如果进入 Stop2 模式时设置了 LVDSE，MCU 实际上会进入 Stop3 模式。



## 第 4 章 存储器

### 4.1 MC9S08DZ60 系列产品存储器映射

MC9S08DZ60 系列产品中的片上存储器包括 RAM、EEPROM、用于非易失性数据存储的 Flash 程序存储器、I/O 和控制 / 状态寄存器。这些寄存器可分为以下 3 类：

- 直接页面寄存器 (0x0000 ~ 0x007F)
- 高端页面 (High-page) 寄存器 (0x1800 ~ 0x18FF)
- 非易失性寄存器 (0xFFB0 ~ 0xFFFF)

MC9S08DZ60	MC9S08DZ48	MC9S08DZ32	MC9S08DZ16
0x0000 DIRECT PAGE REGISTERS 0x007F 128 BYTES	0x0000 DIRECT PAGE REGISTERS 0x007F 128 BYTES	0x0000 DIRECT PAGE REGISTERS 0x007F 128 BYTES	0x0000 DIRECT PAGE REGISTERS 0x007F 128 BYTES
0x0080 RAM 4096 BYTES	0x0080 RAM 3072 BYTES	0x0080 RAM 2048 BYTES	0x0080 RAM 1024 BYTES
0x107F	0x0C7F	0x087F	0x047F
0x1080 FLASH 896 BYTES	0x0C80 UNIMPLEMENTED 2176 BYTES	0x0880 UNIMPLEMENTED 3456 BYTES	0x0480 UNIMPLEMENTED 4736 BYTES
0x1400 EEPROM <sup>1</sup> 2 x 1024 BYTES	0x14FF 0x1500 EEPROM <sup>1</sup> 2 x 768 BYTES	0x15FF 0x1600 EEPROM <sup>1</sup> 2 x 512 BYTES	0x16FF 0x1700 EEPROM <sup>1</sup> 2 x 256 BYTES
0x1800 HIGH PAGE REGISTERS 256 BYTES	0x1800 HIGH PAGE REGISTERS 256 BYTES	0x1800 HIGH PAGE REGISTERS 256 BYTES	0x1800 HIGH PAGE REGISTERS 256 BYTES
0x18FF	0x18FF	0x18FF	0x18FF
0x1900	0x1900 UNIMPLEMENTED 9984 BYTES	0x1900 UNIMPLEMENTED 25,344 BYTES	0x1900 UNIMPLEMENTED 42,240 BYTES
0xFFFF	0xFFFF FLASH 49152 BYTES	0xFFFF FLASH 33792 BYTES	0xFFFF FLASH 16896 BYTES

<sup>1</sup> EEPROM 地址范围显示总 EEPROM 的一半。详尽信息请参见 4.5.10，“EEPROM 映射”。

图 4-1. MC9S08DZ60 存储器图

## 4.2 复位和中断向量分配

表 4-1 为复位和中断向量的地址分配情况。该表中使用的向量名称为飞思卡尔半导体提供的 MC9S08DZ60 系列通用文件中使用的标签。

表 4-1. 复位和中断向量表

地址（高 / 低）	向量	向量名称
0xFFC0:0xFFC1	ACMP2	Vacmp2
0xFFC2:0xFFC3	ACMP1	Vacmp1
0xFFC4:0xFFC5	MSCAN Transmit	Vcantx
0xFFC6:0xFFC7	MSCAN Receive	Vcanrx
0xFFC8:0xFFC9	MSCAN errors	Vcanerr
0xFFCA:0xFFCB	MSCAN wake up	Vcanwu
0xFFCC:0xFFCD	RTC	Vrtc
0xFFCE:0xFFCF	IIC	Viic
0xFFD0:0xFFD1	ADC Conversion	Vadc
0xFFD2:0xFFD3	Port A, Port B, Port D	Vport
0xFFD4:0xFFD5	SCI2 Transmit	Vsci2tx
0xFFD6:0xFFD7	SCI2 Receive	Vsci2rx
0xFFD8:0xFFD9	SCI2 Error	Vsci2err
0xFFDA:0xFFDB	SCI1 Transmit	Vsci1tx
0xFFDC:0xFFDD	SCI1 Receive	Vsci1rx
0xFFDE:0xFFDF	SCI1 Error	Vsci1err
0xFFE0:0xFFE1	SPI	Vspi
0xFFE2:0xFFE3	TPM2 Overflow	Vtpm2ovf
0xFFE4:0xFFE5	TPM2 Channel 1	Vtpm2ch1
0xFFE6:0xFFE7	TPM2 Channel 0	Vtpm2ch0
0xFFE8:0xFFE9	TPM1 Overflow	Vtpm1ovf
0xFFEA:0xFFEB	TPM1 Channel 5	Vtpm1ch5
0xFFEC:0xFFED	TPM1 Channel 4	Vtpm1ch4
0xFFEE:0xFFEF	TPM1 Channel 3	Vtpm1ch3
0xFFF0:0xFFF1	TPM1 Channel 2	Vtpm1ch2
0xFFF2:0xFFF3	TPM1 Channel 1	Vtpm1ch1
0xFFF4:0xFFF5	TPM1 Channel 0	Vtpm1ch0
0xFFF6:0xFFF7	MCG Loss of lock	Vlol
0FFF8:0FFF9	Low-Voltage Detect	Vlvd
0xFFFFA:0xFFFFB	IRQ	Virq
0xFFFFC:0xFFFFD	SWI	Vswi
0xFFFFE:0xFFFFF	Reset	Vreset

## 4.3 寄存器地址和位分配

MC9S08DZ60 系列产品中的寄存器可分为以下几组：

- 直接页面寄存器，位于存储器映象的前 128 个位置上。这些寄存器可以通过高效的直接寻址模式指令访问。
- 高端页面（High-page）寄存器，不经常使用，因此位于存储器映象中 0x1800 以上。在直接页面寄存器中为经常使用的寄存器和 RAM 留出了更多空间。
- 非易失性寄存器，由 Flash 中 0xFFB0 ~ 0xFFFF 之间 16 个位置组成的位置段组成。非易失性寄存器位置包括：
  - NPROT 和 NVOPT，在复位时上载到工作寄存器中。
  - 一个 8 字节后门对比密钥，可选择为用户分配有控制的安全内存访问权限。

由于非易失性寄存器的位置是在 Flash 中，所以必须像其他位置 Flash 一样擦除和编程。

直接页面寄存器可以通过高效的直接寻址模式指令访问。位操作指令可用于访问任何直接页面寄存器中的任何位。[表 4-2](#) 总结了所有用户可访问的直接页面寄存器和控制位。

[表 4-2](#) 所列的直接页面寄存器可以使用更高效的直接寻址模式（这种模式只需要地址的较低字节）。因此，第 1 栏中地址的较低字节用粗体显示。在 [表 4-3](#) 和 [表 4-5](#) 中，第 1 栏中的整个地址都用粗体显示。在 [表 4-2](#), [表 4-3](#), 和 [表 4-5](#) 中，第 2 栏中的寄存器名称用粗体显示以便与右侧的位名称区分。与所列出的位不相关的单元在阴影中显示。带有 0 的阴影单元表示这个未使用的位始终应为 0。带有破折号的阴影单元表示未使用的或预留的位，可以是 1 或 0。

表 4-2. 直接页面寄存器总结（第 1 页，共 3 页）

地址	寄存器名称	位 7	6	5	4	3	2	1	位 0
0x0000	PTAD	PTAD7	PTAD6	PTAD5	PTAD4	PTAD3	PTAD2	PTAD1	PTAD0
0x0001	PTADD	PTADD7	PTADD6	PTADD5	PTADD4	PTADD3	PTADD2	PTADD1	PTADD0
0x0002	PTBD	PTBD7	PTBD6	PTBD5	PTBD4	PTBD3	PTBD2	PTBD1	PTBD0
0x0003	PTBDD	PTBDD7	PTBDD6	PTBDD5	PTBDD4	PTBDD3	PTBDD2	PTBDD1	PTBDD0
0x0004	PTCD	PTCD7	PTCD6	PTCD5	PTCD4	PTCD3	PTCD2	PTCD1	PTCD0
0x0005	PTCDD	PTCDD7	PTCDD6	PTCDD5	PTCDD4	PTCDD3	PTCDD2	PTCDD1	PTCDD0
0x0006	PTDD	PTDD7	PTDD6	PTDD5	PTDD4	PTDD3	PTDD2	PTDD1	PTDD0
0x0007	PTDDD	PTDDD7	PTDDD6	PTDDD5	PTDDD4	PTDDD3	PTDDD2	PTDDD1	PTDDD0
0x0008	PTED	PTED7	PTED6	PTED5	PTED4	PTED3	PTED2	PTED1	PTED0
0x0009	PTEDD	PTEDD7	PTEDD6	PTEDD5	PTEDD4	PTEDD3	PTEDD2	PTEDD1	PTEDD0
0x000A	PTFD	PTFD7	PTFD6	PTFD5	PTFD4	PTFD3	PTFD2	PTFD1	PTFD0
0x000B	PTFDD	PTFDD7	PTFDD6	PTFDD5	PTFDD4	PTFDD3	PTFDD2	PTFDD1	PTFDD0
0x000C	PTGD	0	0	PTGD5	PTGD4	PTGD3	PTGD2	PTGD1	PTGD0
0x000D	PTGDD	0	0	PTGDD5	PTGDD4	PTGDD3	PTGDD2	PTGDD1	PTGDD0
0x000E	ACMP1SC	ACME	ACBGS	ACF	ACIE	ACO	ACOPE	ACMOD1	ACMOD0
0x000F	ACMP2SC	ACME	ACBGS	ACF	ACIE	ACO	ACOPE	ACMOD1	ACMOD0
0x0010	ADCSC1	COCO	AIEN	ADCO			ADCH		
0x0011	ADCSC2	ADACT	ADTRG	ACFE	ACFGT	0	0	—	—
0x0012	ADCRH	0	0	0	0	ADR11	ADR10	ADR9	ADR8
0x0013	ADCRL	ADR7	ADR6	ADR5	ADR4	ADR3	ADR2	ADR1	ADR0
0x0014	ADCCVH	0	0	0	0	ADCV11	ADCV10	ADCV9	ADCV8
0x0015	ADCCVL	ADCV7	ADCV6	ADCV5	ADCV4	ADCV3	ADCV2	ADCV1	ADCV0
0x0016	ADCCFG	ADLPC		ADIV	ADLSMP		MODE		ADICLK
0x0017	APCTL1	ADPC7	ADPC6	ADPC5	ADPC4	ADPC3	ADPC2	ADPC1	ADPC0
0x0018	APCTL2	ADPC15	ADPC14	ADPC13	ADPC12	ADPC11	ADPC10	ADPC9	ADPC8
0x0019	APCTL3	ADPC23	ADPC22	ADPC21	ADPC20	ADPC19	ADPC18	ADPC17	ADPC16
0x001A- 0x001B	预留	—	—	—	—	—	—	—	—
0x001C	IRQSC	0	IRQPDD	IRQEDG	IRQPE	IRQF	IRQACK	IRQIE	IRQMOD
0x001D- 0x001F	预留	—	—	—	—	—	—	—	—
0x0020	TPM1SC	TOF	TOIE	CPWMS	CLKSB	CLKSA	PS2	PS1	PS0
0x0021	TPM1CNTH	Bit 15	14	13	12	11	10	9	Bit 8
0x0022	TPM1CNTL	Bit 7	6	5	4	3	2	1	Bit 0
0x0023	TPM1MODH	Bit 15	14	13	12	11	10	9	Bit 8
0x0024	TPM1MODL	Bit 7	6	5	4	3	2	1	Bit 0
0x0025	TPM1C0SC	CH0F	CH0IE	MS0B	MS0A	ELS0B	ELS0A	0	0
0x0026	TPM1C0VH	Bit 15	14	13	12	11	10	9	Bit 8
0x0027	TPM1C0VL	Bit 7	6	5	4	3	2	1	Bit 0
0x0028	TPM1C1SC	CH1F	CH1IE	MS1B	MS1A	ELS1B	ELS1A	0	0

表 4-2. 直接页面寄存器总结（第 1 页，共 3 页）

地址	寄存器名称	位 7	6	5	4	3	2	1	位 0
0x0029	TPM1C1VH	Bit 15	14	13	12	11	10	9	Bit 8
0x002A	TPM1C1VL	Bit 7	6	5	4	3	2	1	Bit 0
0x002B	TPM1C2SC	CH2F	CH2IE	MS2B	MS2A	ELS2B	ELS2A	0	0
0x002C	TPM1C2VH	Bit 15	14	13	12	11	10	9	Bit 8
0x002D	TPM1C2VL	Bit 7	6	5	4	3	2	1	Bit 0
0x002E	TPM1C3SC	CH3F	CH3IE	MS3B	MS3A	ELS3B	ELS3A	0	0
0x002F	TPM1C3VH	Bit 15	14	13	12	11	10	9	Bit 8
0x0030	TPM1C3VL	Bit 7	6	5	4	3	2	1	Bit 0
0x0031	TPM1C4SC	CH4F	CH4IE	MS4B	MS4A	ELS4B	ELS4A	0	0
0x0032	TPM1C4VH	Bit 15	14	13	12	11	10	9	Bit 8
0x0033	TPM1C4VL	Bit 7	6	5	4	3	2	1	Bit 0
0x0034	TPM1C5SC	CH5F	CH5IE	MS5B	MS5A	ELS5B	ELS5A	0	0
0x0035	TPM1C5VH	Bit 15	14	13	12	11	10	9	Bit 8
0x0036	TPM1C5VL	Bit 7	6	5	4	3	2	1	Bit 0
0x0037	预留	—	—	—	—	—	—	—	—
0x0038	SCI1BDH	LBKDI	RXEDGIE	0	SBR12	SBR11	SBR10	SBR9	SBR8
0x0039	SCI1BDL	SBR7	SBR6	SBR5	SBR4	SBR3	SBR2	SBR1	SBR0
0x003A	SCI1C1	LOOPS	SCISWAI	RSRC	M	WAKE	ILT	PE	PT
0x003B	SCI1C2	TIE	TCIE	RIE	ILIE	TE	RE	RWU	SBK
0x003C	SCI1S1	TDRE	TC	RDRF	IDLE	OR	NF	FE	PF
0x003D	SCI1S2	LBKDIF	RXEDGIF	0	RXINV	RWUID	BRK13	LBKDE	RAF
0x003E	SCI1C3	R8	T8	TXDIR	TXINV	ORIE	NEIE	FEIE	PEIE
0x003F	SCI1D	Bit 7	6	5	4	3	2	1	Bit 0
0x0040	SCI2BDH	LBKDI	RXEDGIE	0	SBR12	SBR11	SBR10	SBR9	SBR8
0x0041	SCI2BDL	SBR7	SBR6	SBR5	SBR4	SBR3	SBR2	SBR1	SBR0
0x0042	SCI2C1	LOOPS	SCISWAI	RSRC	M	WAKE	ILT	PE	PT
0x0043	SCI2C2	TIE	TCIE	RIE	ILIE	TE	RE	RWU	SBK
0x0044	SCI2S1	TDRE	TC	RDRF	IDLE	OR	NF	FE	PF
0x0045	SCI2S2	LBKDIF	RXEDGIF	0	RXINV	RWUID	BRK13	LBKDE	RAF
0x0046	SCI2C3	R8	T8	TXDIR	TXINV	ORIE	NEIE	FEIE	PEIE
0x0047	SCI2D	Bit 7	6	5	4	3	2	1	Bit 0
0x0048	MCGC1	CLKS		RDIV			IREFS	IRCLKEN	IREFSTEN
0x0049	MCGC2	BDIV		RANGE	HGO	LP	EREFs	ERCLKEN	EREFSTEN
0x004A	MCGTRM	TRIM							
0x004B	MCGSC	LOLS	LOCK	PLLST	IREFST	CLKST		OSCINIT	FTRIM
0x004C	MCGC3	LOLIE	PLLS	CME	0	VDIV			
0x004D-	预留	—	—	—	—	—	—	—	—
0x004F		—	—	—	—	—	—	—	—
0x0050	SPIC1	SPIE	SPE	SPTIE	MSTR	CPOL	CPHA	SSOE	LSBFE
0x0051	SPIC2	0	0	0	MODFEN	BIDIROE	0	SPISWAI	SPCO

表 4-2. 直接页面寄存器总结（第 1 页，共 3 页）

地址	寄存器名称	位 7	6	5	4	3	2	1	位 0
0x0052	SPIBR	0	SPPR2	SPPR1	SPPR0	0	SPR2	SPR1	SPR0
0x0053	SPIS	SPRF	0	SPTEF	MODF	0	0	0	0
0x0054	预留	0	0	0	0	0	0	0	0
0x0055	SPID	Bit 7	6	5	4	3	2	1	Bit 0
0x0056-		—	—	—	—	—	—	—	—
0x0057	预留	—	—	—	—	—	—	—	—
0x0058	IICA	AD7	AD6	AD5	AD4	AD3	AD2	AD1	0
0x0059	IICF	MULT				ICR			
0x005A	IICC1	IICEN	IICIE	MST	TX	TXAK	RSTA	0	0
0x005B	IICS	TCF	IAAS	BUSY	ARBL	0	SRW	IICIF	RXAK
0x005C	IICD				DATA				
0x005D	IICC2	GCAEN	ADEXT	0	0	0	AD10	AD9	AD8
0x005E-		—	—	—	—	—	—	—	—
0x005F	预留	—	—	—	—	—	—	—	—
0x0060	TPM2SC	TOF	TOIE	CPWMS	CLKSB	CLKSA	PS2	PS1	PS0
0x0061	TPM2CNTH	Bit 15	14	13	12	11	10	9	Bit 8
0x0062	TPM2CNTL	Bit 7	6	5	4	3	2	1	Bit 0
0x0063	TPM2MODH	Bit 15	14	13	12	11	10	9	Bit 8
0x0064	TPM2MODL	Bit 7	6	5	4	3	2	1	Bit 0
0x0065	TPM2C0SC	CH0F	CH0IE	MS0B	MS0A	ELS0B	ELS0A	0	0
0x0066	TPM2C0VH	Bit 15	14	13	12	11	10	9	Bit 8
0x0067	TPM2C0VL	Bit 7	6	5	4	3	2	1	Bit 0
0x0068	TPM2C1SC	CH1F	CH1IE	MS1B	MS1A	ELS1B	ELS1A	0	0
0x0069	TPM2C1VH	Bit 15	14	13	12	11	10	9	Bit 8
0x006A	TPM2C1VL	Bit 7	6	5	4	3	2	1	Bit 0
0x006B	预留	—	—	—	—	—	—	—	—
0x006C	RTCSC	RTIF	RTCLKS	RTIE			RTCPS		
0x006D	RTCCNT				RTCCNT				
0x006E	RTCMOD				RTCMOD				
0x006F	预留	—	—	—	—	—	—	—	—
0x0070-		—	—	—	—	—	—	—	—
0x007F	预留	—	—	—	—	—	—	—	—

表 4-3 中列出的高端页面寄存器的访问频率比其它 I/O 和控制寄存器低很多，因此存放在可直接寻址的内存空间外，从 0x1800 开始。

表 4-3. 高端页面寄存器总结（第 1 页，共 3 页）

地址	寄存器名称	位 7	6	5	4	3	2	1	位 0
0x1800	SRS	POR	PIN	COP	ILOP	ILAD	LOCS	LVD	0
0x1801	SBDFR	0	0	0	0	0	0	0	BDFR
0x1802	SOPT1	COPT		STOPE	SCI2PS	IICPS	0	0	0
0x1803	SOPT2	COPCLKS	COPW	0	ADHTS	0	MCSEL		
0x1804 – 0x1805	预留	—	—	—	—	—	—	—	—
0x1806	SDIDH	—	—	—	—	ID11	ID10	ID9	ID8
0x1807	SDIDL	ID7	ID6	ID5	ID4	ID3	ID2	ID1	ID0
0x1808	预留	—	—	—	—	—	—	—	—
0x1809	SPMSC1	LVWF	LVWACK	LVWIE	LVDRE	LVDSE	LVDE	0	BGBE
0x180A	SPMSC2	0	0	LVDV	LVWV	PPDF	PPDACK	0	PPDC
0x180B – 0x180F	预留	—	—	—	—	—	—	—	—
0x1810	DBGCAH	Bit 15	14	13	12	11	10	9	Bit 8
0x1811	DBGCAL	Bit 7	6	5	4	3	2	1	Bit 0
0x1812	DBGCBH	Bit 15	14	13	12	11	10	9	Bit 8
0x1813	DBGCBL	Bit 7	6	5	4	3	2	1	Bit 0
0x1814	DBGFH	Bit 15	14	13	12	11	10	9	Bit 8
0x1815	DBGFL	Bit 7	6	5	4	3	2	1	Bit 0
0x1816	DBGC	DBGEN	ARM	TAG	BRKEN	RWA	RWAEN	RWB	RWBEN
0x1817	DBGT	TRGSEL	BEGIN	0	0	TRG3	TRG2	TRG1	TRG0
0x1818	DBGS	AF	BF	ARMF	0	CNT3	CNT2	CNT1	CNT0
0x1819 – 0x181F	预留	—	—	—	—	—	—	—	—
0x1820	FCDIV	DIVLD	PRDIV8	DIV					
0x1821	FOPT	KEYEN	FNORED	EPGMOD	0	0	0	SEC	
0x1822	预留	—	—	—	—	—	—	—	—
0x1823	FCNFG	0	EPGSEL	KEYACC	Reserved <sup>1</sup>	0	0	0	1
0x1824	FPROT	EPS		FPS					
0x1825	FSTAT	FCBEF	FCCF	FPVIOL	FACCERR	0	FBLANK	0	0
0x1826	FCMD	FCMD							
0x1827 – 0x183F	预留	—	—	—	—	—	—	—	—
0x1840	PTAPE	PTAPE7	PTAPE6	PTAPE5	PTAPE4	PTAPE3	PTAPE2	PTAPE1	PTAPE0
0x1841	PTASE	PTASE7	PTASE6	PTASE5	PTASE4	PTASE3	PTASE2	PTASE1	PTASE0
0x1842	PTADS	PTADS7	PTADS6	PTADS5	PTADS4	PTADS3	PTADS2	PTADS1	PTADS0
0x1843	预留	—	—	—	—	—	—	—	—
0x1844	PTASC	0	0	0	0	PTAIF	PTAACK	PTAIE	PTAMOD

表 4-3. 高端页面寄存器总结（第1页，共3页）

地址	寄存器名称	位 7	6	5	4	3	2	1	位 0
0x1845	PTAPS	PTAPS7	PTAPS6	PTAPS5	PTAPS4	PTAPS3	PTAPS2	PTAPS1	PTAPS0
0x1846	PTAES	PTAES7	PTAES6	PTAES5	PTAES4	PTAES3	PTAES2	PTAES1	PTAES0
0x1847	预留	—	—	—	—	—	—	—	—
0x1848	PTBPE	PTBPE7	PTBPE6	PTBPE5	PTBPE4	PTBPE3	PTBPE2	PTBPE1	PTBPE0
0x1849	PTBSE	PTBSE7	PTBSE6	PTBSE5	PTBSE4	PTBSE3	PTBSE2	PTBSE1	PTBSE0
0x184A	PTBDS	PTBDS7	PTBDS6	PTBDS5	PTBDS4	PTBDS3	PTBDS2	PTBDS1	PTBDS0
0x184B	预留	—	—	—	—	—	—	—	—
0x184C	PTBSC	0	0	0	0	PTBIF	PTBACK	PTBIE	PTBMOD
0x184D	PTBPS	PTBPS7	PTBPS6	PTBPS5	PTBPS4	PTBPS3	PTBPS2	PTBPS1	PTBPS0
0x184E	PTBES	PTBES7	PTBES6	PTBES5	PTBES4	PTBES3	PTBES2	PTBES1	PTBES0
0x184F	预留	—	—	—	—	—	—	—	—
0x1850	PTCPE	PTCPE7	PTCPE6	PTCPE5	PTCPE4	PTCPE3	PTCPE2	PTCPE1	PTCPE0
0x1851	PTCSE	PTCSE7	PTCSE6	PTCSE5	PTCSE4	PTCSE3	PTCSE2	PTCSE1	PTCSE0
0x1852	PTCDS	PTCDS7	PTCDS6	PTCDS5	PTCDS4	PTCDS3	PTCDS2	PTCDS1	PTCDS0
0x1853-0x1857	预留	—	—	—	—	—	—	—	—
0x1858	PTDPE	PTDPE7	PTDPE6	PTDPE5	PTDPE4	PTDPE3	PTDPE2	PTDPE1	PTDPE0
0x1859	PTDSE	PTDSE7	PTDSE6	PTDSE5	PTDSE4	PTDSE3	PTDSE2	PTDSE1	PTDSE0
0x185A	PTDDS	PTDDS7	PTDDS6	PTDDS5	PTDDS4	PTDDS3	PTDDS2	PTDDS1	PTDDS0
0x185B	预留	—	—	—	—	—	—	—	—
0x185C	PTDSC	0	0	0	0	PTDIF	PTDACK	PTDIE	PTDMOD
0x185D	PTDPS	PTDPS7	PTDPS6	PTDPS5	PTDPS4	PTDPS3	PTDPS2	PTDPS1	PTDPS0
0x185E	PTDES	PTDES7	PTDES6	PTDES5	PTDES4	PTDES3	PTDES2	PTDES1	PTDES0
0x185F	预留	—	—	—	—	—	—	—	—
0x1860	PTEPE	PTEPE7	PTEPE6	PTEPE5	PTEPE4	PTEPE3	PTEPE2	PTEPE1	PTEPE0
0x1861	PTESE	PTESE7	PTESE6	PTESE5	PTESE4	PTESE3	PTESE2	PTESE1	PTESE0
0x1862	PTEDS	PTEDS7	PTEDS6	PTEDS5	PTEDS4	PTEDS3	PTEDS2	PTEDS1	PTEDS0
0x1863-0x1867	预留	—	—	—	—	—	—	—	—
0x1868	PTFPE	PTFPE7	PTFPE6	PTFPE5	PTFPE4	PTFPE3	PTFPE2	PTFPE1	PTFPE0
0x1869	PTFSE	PTFSE7	PTFSE6	PTFSE5	PTFSE4	PTFSE3	PTFSE2	PTFSE1	PTFSE0
0x186A	PTFDS	PTFDS7	PTFDS6	PTFDS5	PTFDS4	PTFDS3	PTFDS2	PTFDS1	PTFDS0
0x186B-0x186F	预留	—	—	—	—	—	—	—	—
0x1870	PTGPE	0	0	PTGPE5	PTGPE4	PTGPE3	PTGPE2	PTGPE1	PTGPE0
0x1871	PTGSE	0	0	PTGSE5	PTGSE4	PTGSE3	PTGSE2	PTGSE1	PTGSE0
0x1872	PTGDS	0	0	PTGDS5	PTGDS4	PTGDS3	PTGDS2	PTGDS1	PTGDS0
0x1873-0x187F	预留	—	—	—	—	—	—	—	—
0x1880	CANCTL0	RXFRM	RXACT	CSWAI	SYNCH	TIME	WUPE	SLPRQ	INITRQ

表 4-3. 高端页面寄存器总结（第 1 页，共 3 页）

地址	寄存器名称	位 7	6	5	4	3	2	1	位 0
0x1881	CANCTL1	CANE	CLKSRC	LOOPB	LISTEN	BORM	WUPM	SLPAK	INITAK
0x1882	CANBTR0	SJW1	SJW0	BRP5	BRP4	BRP3	BRP2	BRP1	BRP0
0x1883	CANBTR1	SAMP	TSEG22	TSEG21	TSEG20	TSEG13	TSEG12	TSEG11	TSEG10
0x1884	CANRFLG	WUPIF	CSCIF	RSTAT1	RSTAT0	TSTAT1	TSTAT0	OVRIF	RXF
0x1885	CANRIER	WUPIE	CSCIE	RSTATE1	RSTATE0	TSTATE1	TSTATE0	OVRIE	RXFIE
0x1886	CANTFLG	0	0	0	0	0	TXE2	TXE1	TXE0
0x1887	CANTIER	0	0	0	0	0	TXEIE2	TXEIE1	TXEIE0
0x1888	CANTARQ	0	0	0	0	0	ABTRQ2	ABTRQ1	ABTRQ0
0x1889	CANTAAK	0	0	0	0	0	ABTAK2	ABTAK1	ABTAK0
0x188A	CANTBSEL	0	0	0	0	0	TX2	TX1	TX0
0x188B	CANIDAC	0	0	IDAM1	IDAM0	0	IDHIT2	IDHIT1	IDHIT0
0x188C	预留	0	0	0	0	0	0	0	0
0x188D	CANMISC	0	0	0	0	0	0	0	BOHOLD
0x188E	CANRXERR	RXERR7	RXERR6	RXERR5	RXERR4	RXERR3	RXERR2	RXERR1	RXERR0
0x188F	CANTXERR	TXERR7	TXERR6	TXERR5	TXERR4	TXERR3	TXERR2	TXERR1	TXERR0
0x1890 – 0x1893	CANIDAR0 – CANIDAR3	AC7	AC6	AC5	AC4	AC3	AC2	AC1	AC0
0x1894 – 0x1897	CANIDMR0 – CANIDMR3	AM7	AM6	AM5	AM4	AM3	AM2	AM1	AM0
0x1898 – 0x189B	CANIDAR4 – CANIDAR7	AC7	AC6	AC5	AC4	AC3	AC2	AC1	AC0
0x189C – 0x189F	CANIDMR4 – CANIDMR7	AM7	AM6	AM5	AM4	AM3	AM2	AM1	AM0
0x18BE	CANTTSRH	TSR15	TSR14	TSR13	TSR12	TSR11	TSR10	TSR9	TSR8
0x18BF	CANTTSRL	TSR7	TSR6	TSR5	TSR4	TSR3	TSR2	TSR1	TSR0
0x18C0 – 0x18FF	预留	—	—	—	—	—	—	—	—

<sup>1</sup> 该位被预留。用户必须在该位上写一个 1。否则可能会导致出现异常。

图 4-4 为接收及发送缓冲器（用于扩展的识别符映射）的结构。这些寄存器各有不同，具体取决于选择了标准映射还是扩展的映射。有关标准映射和扩展映射的更详尽信息请参见第 12 章，“飞思卡尔控制器局域网 (S08MSCANV1)”。

表 4-4. MSCAN 前台接收和发射缓冲器布局—显示的为扩展映射

0x18A0	CANRIDR0	ID28	ID27	ID26	ID25	ID24	ID23	ID22	ID21
0x18A1	CANRIDR1	ID20	ID19	ID18	SRR <sup>(1)</sup>	IDE <sup>(1)</sup>	ID17	ID16	ID15
0x18A2	CANRIDR2	ID14	ID13	ID12	ID11	ID10	ID9	ID8	ID7
0x18A3	CANRIDR3	ID6	ID5	ID4	ID3	ID2	ID1	ID0	RTR <sup>2</sup>
0x18A4 -	CANRDSR0 -	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0x18AB	CANRDSR7	—	—	—	—	DLC3	DLC2	DLC1	DLC0
0x18AC	CANRDLR	—	—	—	—	—	—	—	—
0x18AD	预留	—	—	—	—	—	—	—	—
0x18AE	CANRTSRH	TSR15	TSR14	TSR13	TSR12	TSR11	TSR10	TSR9	TSR8
0x18AF	CANRTSRL	TSR7	TSR6	TSR5	TSR4	TSR3	TSR2	TSR1	TSR0
0x18B0	CANTIDR0	ID10	ID9	ID8	ID7	ID6	ID5	ID4	ID3
0x18B1	CANTIDR1	ID2	ID1	ID0	RTR	IDE	—	—	—
0x18B2	CANTIDR2	—	—	—	—	—	—	—	—
0x18B3	CANTIDR3	—	—	—	—	—	—	—	—
0x18B4 -	CANTDSR0 -	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0x18BB	CANTDSR7	—	—	—	—	DLC3	DLC2	DLC1	DLC0
0x18BC	CANTDLR	PRI07	PRI06	PRI05	PRI04	PRI03	PRI02	PRI01	PRI00
0x18BD	CANTTBPB	—	—	—	—	—	—	—	—

<sup>1</sup> SRR 和 IDE 均为 1s。

<sup>2</sup> RTR 的位置在扩展识别符映射和标准识别符映射不同。

表 4-5 的非易失性 Flash 寄存器位于 Flash 中。这些寄存器包括 8 个字节的后门密钥 NVBACKKEY。该密钥可用于访问安全的内存资源。在复位过程中，Flash 中非易失性寄存器区域的 NVPROT 和 NVOPT 内容会被转移到高端页面寄存器中相应的 FPROT 和 FOPT 工作寄存器中，以控制安全性和块保护选项。

表 4-5. 非易失性寄存器总结

地址	寄存器名称	位 7	6	5	4	3	2	1	位 0
0xFFAE	预留用于存储 FTRIM	0	0	0	0	0	0	0	FTRIM
TRIM									
8 字节对比密钥									
—									
0xFFB8 -	预留	—	—	—	—	—	—	—	—
0xFFBC	—	—	—	—	—	—	—	—	—
0xFFBD	NVPROT	EPS		FPS					
0xFFBE	预留	—	—	—	—	—	—	—	—
0xFFBF	NVOPT	KEYEN	FNORED	EPGMOD	0	0	0	SEC	

如果密钥启用（**KEYEN**）位为 1，那么 8 字节对比密钥可用于暂时脱离内存安全的限制。这种密钥机制只能通过在安全内存中运行的用户代码访问。（安全密钥不能通过后台调试命令直接输入。）这个安全密钥可通过将 **KEYEN** 位设为 0 来完全禁用。如果这个安全密钥被禁用，那么脱离安全限制的唯一方式是整体擦除 Flash（通常通过后台调试接口）并确认 Flash 是空的。为了避免在下一次复位后返回到安全模式，应该将安全位（**SEC**）设置为非安全状态（1: 0）。

## 4.4 RAM

MC9S08DZ60 系列包括静态 RAM。RAM 中 0x0100 以下的位置可以使用更高效的直接寻址模式访问，而这一区域中的任何单一比特可以通过位操作指令（**BCLR**、**BSET**、**BRCLR** 和 **BRSET**）访问。首选的方式是在这一区域中查找 RAM 最频繁访问的程序变量。

在 MCU 处于低功耗等待、Stop2 或 Stop3 模式时，RAM 会保留数据。加电启动时，RAM 中的内容不会被初始化。如果电源电压没有降低到 RAM 保留 ( $V_{RAM}$ ) 的最低值以下，RAM 数据就不会受到复位的任何影响。

为了实现与 M68HC05 MCU 的兼容性，HCS08 会将栈指针复位为 0x00FF。在 MC9S08DZ60 系列中，最好的方法通常是将栈指针重新初始化到 RAM 顶部，以便使经常被访问的 RAM 变量和位可寻址程序变量处于直接寄存器。复位初始化程序（其中的 **RamLast** 等于飞思卡尔半导体等同文件中 RAM 的最高地址）中包含以下两个指令序列。

```
LDHX      #RamLast+1    ;point one past RAM
TXS          ;SP<-(H:X-1)
```

在启用了安全性的情况下，RAM 被认为是一种安全的内存资源，不能通过 BDM 或从非安全内存中执行代码来访问。若欲了解有关安全特性的更详尽描述，请参见 [4.5.9，“安全性”](#)。

## 4.5 Flash 和 EEPROM

MC9S08DZ60 系列器件包括 Flash 和 EEPROM 存储器。这些存储器主要用于保存程序和数据。在线编程使正在运行的程序和数据可以在应用产品的最终组装完成后分别上载到 Flash 和 EEPROM 中。我们可以通过单线后台调试接口对阵列进行编程。由于擦除和编程操作不需要特殊的电压，所以也可以通过其他软件控制的通信路径来实现应用编程。有关在线和应用内编程的更详尽描述，请参见“[HCS08 系列参考手册，第 1 卷](#)”（飞思卡尔半导体文件编号 HCS08RMv1）。

### 4.5.1 特性

Flash 和 EEPROM 存储器具有以下特性：

- 阵列大小请参见表 1-1
- Flash 分区大小：768 字节
- EEPROM 分区大小：可选 4 字节或 8 字节分区映射操作
- 单一电源程序和擦除
- 用于快速编程和擦除操作的命令接口
- 一般电压和温度下最多 100,000 个编程 / 擦除循环
- 灵活的块保护和向量重定向

- Flash、EEPROM 和 RAM 的安全特性
- 突发编程功能
- 扇区擦除终止

## 4.5.2 编程和擦除时间

在接受任何编程或擦除命令前，必须通过写 Flash 和 EEPROM 时钟分频寄存器 (FCDIV) 以将 Flash 和 EEPROM 模块的内部时钟设置为 150 kHz ~ 200 kHz 之间的频率 ( $f_{FCLK}$ ) (请参见 4.5.11.1，“Flash 和 EEPROM 时钟分频寄存器 (FCDIV)”)。这个寄存器只能写入一次，因此这一写入操作通常是在复位初始化过程中执行的。用户必须确保在写入 FCDIV 寄存器之前没有设置 FACCERR。命令处理器使用最终时钟 ( $1/f_{FCLK}$ ) 的一个周期来对编程和擦除脉冲定时。命令处理器利用这些定时脉冲的一个整数来完成编程或擦除命令。

表 4-6 显示了编程和擦除时间。总线时钟频率和 FCDIV 决定 FCLK 的频率 ( $f_{FCLK}$ )。一个 FCLK 周期为  $t_{FCLK} = 1/f_{FCLK}$ 。定时器显示为多个 FCLK 循环和一个绝对时间 ( $t_{FCLK} = 5\text{s}$ )。显示的编程和擦除时间包括命令状态机的开销及编程和擦除电压的启用及禁用的时间。

表 4-6. 编程和擦除时间

参数	FCLK 循环	FCLK = 200 kHz 时的时间
字节程序	9	45 ms
突发程序	4	20 ms <sup>1</sup>
分区擦除	4000	20 ms
整体擦除	20,000	100 ms
分区擦除终止	4	20 ms <sup>1</sup>

<sup>1</sup> 不包括开始 / 结束开销。

## 4.5.3 编程和擦除命令的执行

在复位和错误标记被清除后，FCDIV 寄存器在开始命令执行之前必须初始化。命令执行步骤如下：

1. 将一个数据值写入到 Flash 或 EEPROM 阵列中的一个地址中。该地址和写入的数据信息被锁定到 Flash 和 EEPROM 接口上。这一写入操作是任何命令序列中要求的第一步。对于擦除和空白检查命令，这些数据的值并不重要。对于分区擦除命令，地址可以是将要擦除的 Flash 或 EEPROM 分区中的任何地址。对于整体擦除和空白检查命令，地址可以是 Flash 或 EEPROM 内存中的任何地址。Flash 和 EEPROM 擦除互相独立。

### 注意

在对 Flash 或 EEPROM 中的特定字节进行编程前，该字节所在的分区必须通过整体或分区擦除操作擦除。如果对已经编程的字节中的位进行重新编程而不首先进行擦除，可能会造成 Flash 或 EEPROM 内存中保存数据的错误。

2. 将命令代码写入到 FCMD 中。6 个有效的命令分别是空白检查（blank check, 0x05）、字节编程（byte program, 0x20）、突发编程（burst program, 0x25）、分区擦除（sector erase, 0x40）、整体擦除（mass erase<sup>1</sup>, 0x41）和分区擦除终止（sector erase abort, 0x47）。命令代码被锁定到命令缓冲器中。
  3. 将一个 1 写入到 FSTAT 中的 FCBEF 位上，以清除 FCBEF 并发起命令（包括其地址和数据信息）。

在写内存阵列之后到写 1 用于清除 **FCBEF** 并发起完整命令之前的任何时候，可以通过向 **FCBEF** 中写入一个“0”，来手工终止部分命令顺序。以这种方式终止一个命令会设置 **FACCERR** 访问错误标记，而这个标记必须在开始一个新命令之前清除掉。

整个过程必须遵守严格监控的流程，否则命令将不会被接受。通过这种方式可以最大限度地降低无意中修改内存内容的可能性。命令完整标记（FCCF）用于指示一条命令是否完整。要启动命令，必须通过清除 FCBEF 来使命令序列完整。图 4-2 是执行除突发编程和分区擦除终止以外的所有命令的流程。

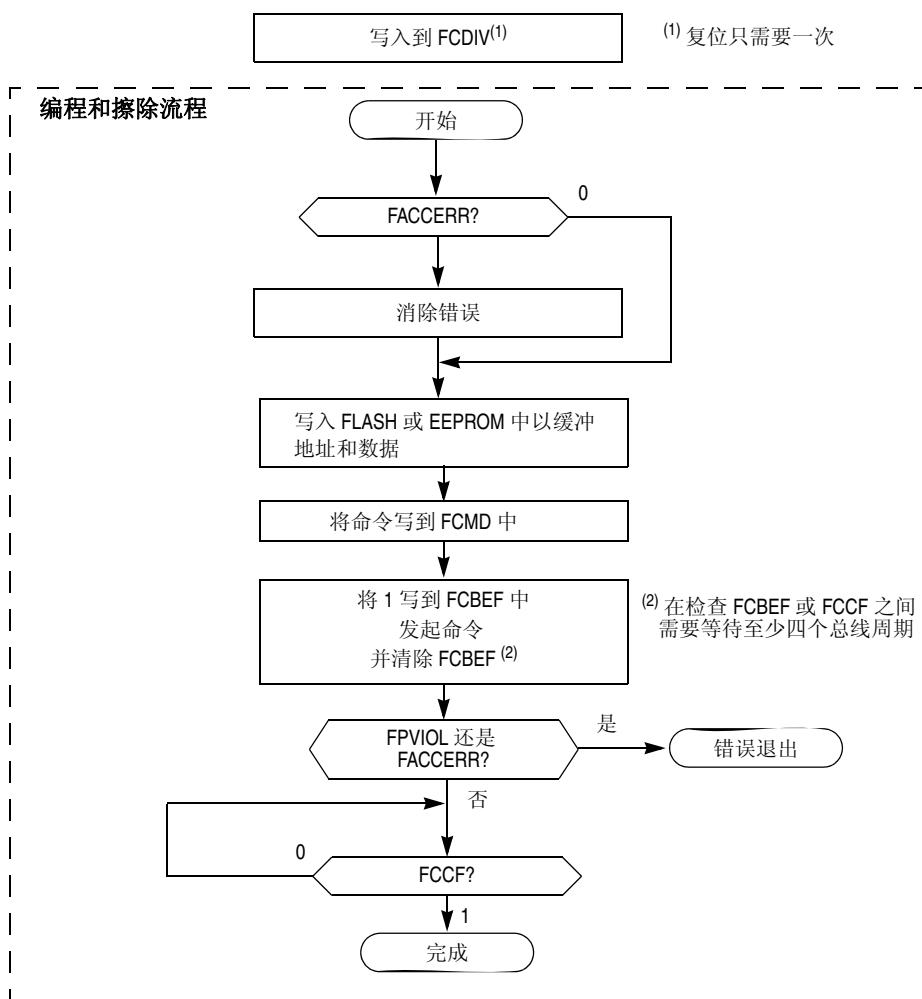


图 4-2. 编程和擦除流程图

1. 整体擦除只能在 Flash 块完全不受保护的情况下进行。

#### 4.5.4 突发编程执行

突发编程命令能在比标准编程命令更短的时间内对数据的连续字节进行编程。这是因为 Flash 阵列的高电压在编程操作之间不需要断开。通常情况下，在发出编程或擦除命令后，必须启用与 Flash 相关的一个内部电荷泵用于为阵列提供高电压。命令执行完成后，该电荷泵会被关闭。发出突发编程命令后，电荷泵在以下两种条件下会被开启而且在突发编程操作完成后将保持开启状态：

- 下一个突发编程命令序列在设置 FCCF 位之前已开始。
- 下一个顺序地址从所编程的当前字节所在的相同突发块中选择了一个字节。该 Flash 中的突发块包括 32 个字节。新的突发块在每个 32 字节地址的边界开始。

在突发模式下对一系列连续字节的第一个字节进行编程所需要的时间与标准模式下编程一个字节所需的时间相同。如果达到上述两个条件，后面的字节将在突发编程时间内编程。如果下一个顺序地址是新的一行的开始，那么该字节的编程时间将是标准时间而不是突发时间。这是因为到阵列的高电压必须断开后重新开启。如果在当前命令完成前，队列中没有任何新的突发命令，那么电荷泵将关闭，高电压将从阵列上断开。

图 4-3 为执行突发编程操作的流程。

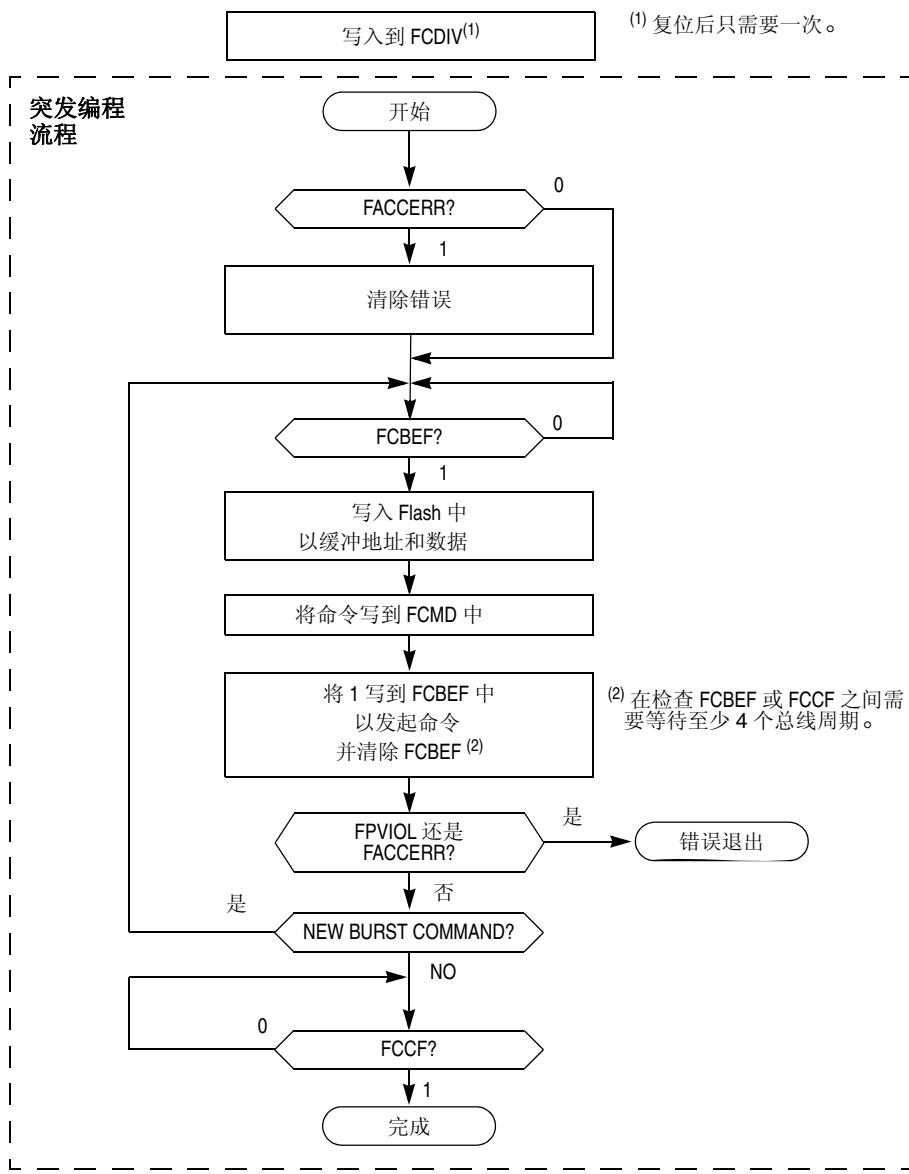


图 4-3. 突发编程流程图

#### 4.5.5 分区擦除终止

分区擦除终止操作用于终止正在进行的分区擦除操作，以便使其他分区可用于读取和编程操作而不需要等待分区擦除完成。

分区擦除终止命令写入顺序如下：

1. 写入任何 Flash 或 EEPROM 地址以开始分区擦除终止命令的命令写入顺序。写入的地址和数据将被忽略。
2. 向 FCMD 寄存器中写入分区擦除终止命令 0x47。
3. 将一个 1 写入到 FCBEF 中来发起分区擦除终止命令以清除 FSTAT 寄存器中的 FCBEF 标记。

如果正在进行的分区擦除操作由于分区擦除终止命令而提前终止，FACCERR 将在操作完成（由设置的 FCCF 标记显示）后马上设置。设置 FACCERR 标记的目的是告诉用户 Flash 分区 k 可能没有完全擦除，对这个分区内任何位置进行编程前需要发出一个新的分区擦除命令。

如果发出分区擦除终止命令时，分区擦除操作正常完成了，那么在该操作完成（由设置的 FCCF 标记显示）后将不会设置 FACCERR 标记。因此，如果在分区擦除终止命令完成后没有设置 FACCERR 标记，那么发出终止命令时正在擦除的分区将完全擦除。

图 4-4 为分区擦除终止操作的流程。

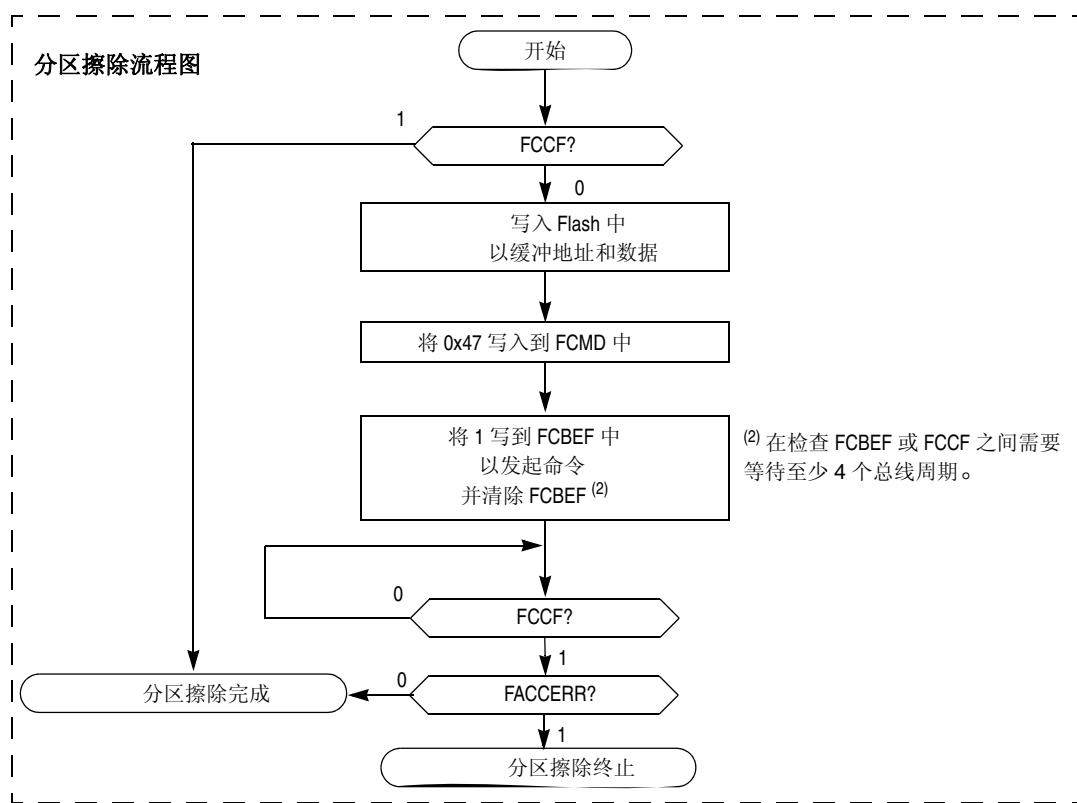


图 4-4. 分区擦除终止流程图

### 注意

FCBEF 标记在发出分区擦除终止命令后不会设置。如果在一个分区擦除终止操作执行过程中，开始一个新的命令写入顺序，那么 FSTAT 寄存器中将设置 FACCERR 标记。ACCERR 标记（如果被设置）被清除后，新的命令写入顺序可以开始。

### 注意

分区擦除终止命令应谨慎使用，因为终止分区擦除操作将计算为一个完整的编程或擦除周期。

## 4.5.6 访问错误

在违反命令执行协议时会出现访问错误。

下面所列的任何操作都会导致 FSTAT 中设置访问错误标记 (FACCERR)。在处理任何命令前，必须通过将一个 1 写入到 FSTAT 中的 FACCERR 来清除 FACCERR。

- 在通过写 FCDIV 寄存器来设置内部 Flash 和 EEPROM 时钟频率前写入 Flash 地址。
- 在没有设置 FCBEF 的情况下写入 Flash 地址。(命令缓冲器清空之前新的命令不能开始。)
- 在发出前一个命令之前第 2 次写入 Flash 地址。(每个命令只能向 Flash 中写入一次。)
- 在发出前一个命令之前第 2 次写入 FCMD。(每个命令只能向 FCMD 中写入一次。)
- 在写 Flash 地址后写 FCMD 以外的任何 Flash 控制寄存器。
- 向 FCMD 中写入 6 个允许的代码 (0x05, 0x20, 0x25, 0x40, 0x41, or 0x47) 以外的任何命令代码。
- 向 FCMD 中写命令之后写 FSTAT 以外的任何 Flash 控制寄存器 (以清除 FCBEF 并发出命令)。
- MCU 在某个编程或擦除命令执行过程中进入停止模式。(命令被终止。)
- MCU 受到安全保护时通过后台调试命令写入字节编程、突发编程、分区擦除或分区擦除终止命令代码 (0x20, 0x25, 0x40, or 0x47)。后台调试控制器只能在 MCU 受到安全保护时才能执行空白检查和整体擦除命令。)
- 将 0 写入到 FCBEF 中以取消部分命令。

## 4.5.7 块保护

块保护特性可以防止 Flash 或 EEPROM 的受保护区域发生编程或擦除修改。块保护通过 Flash and EEPROM 保护寄存器 (FPROT) 进行控制。EPS 位决定 EEPROM 的受保护区域而 FPS 位决定 Flash 的受保护区域。详细说明请参见 [4.5.11.4，“Flash 和 EEPROM 保护寄存器 \(FPROT and NVPROT\)”](#)。

在退出复位操作后，FPROT 会上载 NVPROT 位置（在 Flash 的非易失性寄存器块中）的内容。任何试图减少受保护区域大小的 FPROT 写入操作都会被忽略。由于 NVPROT 在 Flash 的最后一个分区中，所以如果任何数量的内存受到保护，NVPROT 自身也会受到保护，不会受到任何应用软件的影响（不管是有意的还是无意的）。若要擦除和重新编程受保护的 Flash，可通过后台调试命令写 FPROT。

块保护的一个用途是为 bootloader 程序保护而 Flash 的一个区域。这个 bootloader 程序可以调用 Flash 外的常用程序（可用于分区擦除 Flash 的其余部分并进行重新编程）。bootloader 可以受到很好的保护，即使在擦除和编程操作中 MCU 电源中断。

### 4.5.8 向量重定向

在 Flash 受到块保护的同时，复位和中断向量也将受到有效保护。向量重定向使用户可以修改中断向量信息而不必影响对 bootloader 和复位向量空间的保护。**向量重定向通过在地址 0xFFBF 到 0 上的 NVOPT 寄存器中编程 FNORED 位来启用。**要使重定向正常进行，至少一部分 Flash 必须通过对地址 0xFFBD 上的 NVPROT 寄存器进行编程来进行块保护。所有中断向量（内存位置为 0xFFC0–0xFFFF）都被重定向，虽然复位向量（0xFFFFE:0xFFFF）没有被重定向。

例如，如果 Flash 的 1536 个字节受到保护，那么受保护的地址区域为从 0xFA00 到 0xFFFF。中断向量（0xFFC0–0xFFFFD）被重定向到位置 0xF9C0–0xF9FD。如果启用了向量重定向并发生了中断，那么 0xF9E0:0xF9E1 上的值被用于向量而不是 0FFE0:0FFE1 上的值。这样，用户就可以利用新的程序代码对 Flash 的未保护部分进行重新编程，增加新的中断向量值，同时保留受保护的区域（这包括不发生任何变化的缺省向量）。

### 4.5.9 安全性

MC9S08DZ60 系列包含用于防止非法访问 Flash、EEPROM 和 RAM 存储器内容的电路。启用了安全性功能后，Flash、EEPROM 和 RAM 被看作是安全的资源。直接页面寄存器、高端页面寄存器和后台调试控制器被看作是不安全的资源。安全存储器中执行的程序可以正常访问 MCU 位置和资源。通过不安全存储器空间内运行的程序或通过后台调试接口访问安全存储器位置的任何尝试都将被阻止（写入操作被忽略，而读取操作则全部返回 0）。

安全性的启用和关闭由 FOPT 寄存器中的两个寄存器位 (SEC[1:0]) 的状态确定。在复位过程中，非易失性位置 NVOPT 的内容从 Flash 中拷贝到高端页面寄存器空间内的工作 FOPT 寄存器上。用户可以通过编程 NVOPT 位置来启用安全性。这一操作可以在对 Flash 进行编程的同时进行。1:0 状态会关闭安全性；而另外 3 种组合会启用安全性。请注意，擦除状态 (1:1) 会使 MCU 处于安全状态。在开发过程中，不管 Flash 什么时候被擦除，立即将 NVOPT 中的 SEC0 设置为 0 以便使 SEC = 1:0 是一种有效的方法。这将使 MCU 在后来的复位完成后继续处于不安全状态。

MCU 处于安全状态时不能启用片上调试模块。您可以为后台存储器访问命令使用独立的后台调试控制器，但 MCU 不能进入主动后台模式，除非您在复位的上升边将 BKGD 保持在较低位置。

用户可以选择通过一个 8 字节后门安全密钥来设置允许或不允许安全解锁机制。如果 NVOPT/FOPT 中的非易失性 KEYEN 位为 0，那么说明后门密钥被禁用，在没有完全擦除所有 Flash 位置的情况下不能启用安全性。如果 KEYEN 为 1，那么安全的用户程序可以通过以下方式来暂时关闭安全性：

1. 向 FCNFG 寄存器中的 KEYACC 写入 1。这将使 Flash 模块将后门对比密钥位置 (NVBACKKEY through NVBACKKEY+7) 的写入解释为将与密钥进行对比的值而不是 Flash 编程或擦除命令的第一步。
2. 将用户输入的密钥值写入到位置 NVBACKKEY 到 NVBACKKEY+7 上。这些写入操作必须按顺序进行，以 NVBACKKEY 值开始，以 NVBACKKEY+7 值结束。这些写入操作中必须使用 STHX，因为这些写入不能在相邻的总线循环上完成。用户软件一般通过通信接口（如串行 I/O）从 MCU 系统外部获取密钥代码。
3. 向 FCNFG 寄存器中的 KEYACC 写入 0。如果写入的 8 字节密钥与 Flash 位置上保存的密钥相匹配，那么 SEC 位被自动修改为 1:0，同时安全性将关闭，直到下一次复位。

安全密钥只能从安全存储器（RAM、EEPROM 或 Flash）中写入，因此在没有安全的用户程序协作的情况下不能通过后台命令输入。

后门对比密钥 (NVBACKKEY through NVBACKKEY+7) 保存在非易失性寄存器空间内的 Flash 位置上，因此用户可以准确地编程这些位置，就象编程任何其他 Flash 位置一样。非易失性寄存器与复位和中断向量在 Flash 的同一个 768 字节块中，因此对这一空间进行块保护同时也可以保护后门对比密钥。块保护不能从用户应用程序上修改，因此，如果向量空间受到块保护，后门安全密钥机制就不能永久性地修改块保护、安全设置或后门密钥。

通过以下步骤，您可以通过后台调试接口始终关闭安全性：

1. 通过写入 FPROT 来禁用任何块保护。FPROT 只能通过后台调试命令写入而不能通过应用软件写入。
2. 在必要时整体擦除 Flash。
3. 对 Flash 进行空白检查。如果 Flash 内完全擦除，那么在下一次复位前安全性一直处于关闭状态。为了避免在下一次复位后返回到安全模式，对 NVOPT 进行编程使 SEC = 1:0。

## 4.5.10 EEPROM 映射

只有一半 EEPROM 处于存储器映象中。FCNFG 寄存器中的 EPGSEL 位用于确定阵列的那一半可从前台访问，而另一半不能从后台访问。对于配置 8 字节 EEPROM 分区，有两种映射模式可供选择：4 字节模式和 8 字节模式。每种模式都通过 FOPT 寄存器中的 EPGMOD 位确定。

在 4 字节分区模式 (EPGMOD = 0) 中，每个 8 个字节分区被分成两部分，4 个字节在前台，4 个在后台，但都在相同的地址上。EPGSEL 位确定哪 4 个字节可以访问。在分区擦除过程中，整个 8 字节分区（前台的 4 个字节和后台的 4 个字节）都被擦除。

在 8 字节分区模式 (EPGMOD = 1) 中，每个 8 字节分区在一个页面上。EPGSEL 位确定哪些分区在后台。在分区擦除过程中，前台的整个 8 字节分区会被擦除。

## 4.5.11 Flash 和 EEPROM 寄存器及控制位

Flash 和 EEPROM 模块在高端页面寄存器空间内有 7 个 8 位寄存器，在 Flash 的非易失性寄存器空间内有 3 个位置。这些位置中的两个在复位时被拷贝到两个相应的高端页面控制寄存器中。Flash 中还有一个 8 字节对比密钥。对于 Flash 和 EEPROM 寄存器的绝对地址分配情况，请参见 [表 4-3](#) 和 [表 4-5](#)。这一部分用名称指代寄存器和控制位。通常飞思卡尔半导体提供变量定义或头文件用于将这些名称转换为相应的绝对地址。

### 4.5.11.1 Flash 和 EEPROM 时钟分频寄存器 (FCDIV)

该寄存器的第 7 位是一个只读标记。6:0 位可以在任何时候读取但只能写入一次。在开始任何擦除或编程操作之前，写入该寄存器以将非易失性内存系统的时钟频率设置在可接受的限度内。

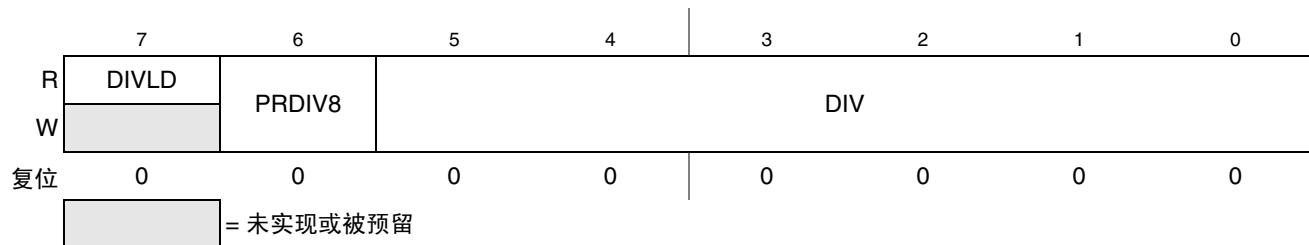


图 4-5. Flash 和 EEPROM 时钟分频寄存器 (FCDIV)

表 4-7. FCDIV 寄存器字段描述

字段	描述
7 DIVLD	<b>Divisor 加载状态标识</b> — 当置 1 时，这个只读状态标记指出 FCDIV 寄存器自从复位后已有写入。复位会清除该位而且第一次写入该寄存器的操作将导致该位被置 1，不管写入什么数据。 0 FCDIV 自复位后没有写入；Flash 和 EEPROM 的擦除和编程操作被禁止。 1 FCDIV 自复位后已写入；Flash 和 EEPROM 的擦除和编程操作已开启。
6 PRDIV8	<b>预分频（分频）Flash 和 EEPROM 时钟除以 8</b> （该位只设置一次。） 0 Flash 和 EEPROM 时钟分频器的时钟输入为总线速率时钟。 1 Flash 和 EEPROM 时钟分频器的时钟输入为总线速率时钟除以 8。
5:0 DIV	<b>Divisor for Flash 和 EEPROM 时钟分频器</b> — 这些位只写入一次。Flash 和 EEPROM 时钟分频器用 6 位 DIV 字段中的值除总线速率时钟（如果 PRDIV8 = 1，则用总线速率时钟除以 8）再加上 1。最后的内部 Flash 和 EEPROM 时钟的频率必须在 200 kHz 到 150 kHz 的范围内，这样才能使 Flash 和 EEPROM 正常运行。编程 / 擦除定时脉冲为这个内部 Flash 和 EEPROM 时钟的一个循环，这相当于 5 ms 到 6.7 ms。自动编程逻辑使用这些脉冲的整数来完成擦除或编程操作。请参见 等式 4-1 和 等式 4-2。

$$\text{if } \text{PRDIV8} = 0 - f_{\text{FCLK}} = f_{\text{Bus}} \div (\text{DIV} + 1) \quad \text{等式 4-1}$$

$$\text{if } \text{PRDIV8} = 1 - f_{\text{FCLK}} = f_{\text{Bus}} \div (8 \times (\text{DIV} + 1)) \quad \text{等式 4-2}$$

表 4-8 为给定总线频率上 PRDIV8 和 DIV 的相应值。

表 4-8. Flash 和 EEPROM 时钟分频器设置

$f_{\text{Bus}}$	PRDIV8 (二进制)	DIV (十进制)	$f_{\text{FCLK}}$	编程 / 擦除定时脉冲 (最少 5 $\mu\text{s}$ , 最多 6.7 $\mu\text{s}$ )
20 MHz	1	12	192.3 kHz	5.2 $\mu\text{s}$
10 MHz	0	49	200 kHz	5 $\mu\text{s}$
8 MHz	0	39	200 kHz	5 $\mu\text{s}$
4 MHz	0	19	200 kHz	5 $\mu\text{s}$
2 MHz	0	9	200 kHz	5 $\mu\text{s}$
1 MHz	0	4	200 kHz	5 $\mu\text{s}$
200 kHz	0	0	200 kHz	5 $\mu\text{s}$
150 kHz	0	0	150 kHz	6.7 $\mu\text{s}$

### 4.5.11.2 Flash 和 EEPROM 选项寄存器 (FOPT 和 NVOPT)

在复位过程中，非易失性位置 NVOPT 上的内容从 Flash 拷贝到 FOPT 中。若想修改这个寄存器中的值，可对 Flash 中的 NVOPT 位置进行擦除和重新编程，然后发出新的 MCU 复位命令。

	7	6	5	4	3	2	1	0
R	KEYEN	FNORED	EPGMOD	0	0	0	SEC	
W								
复位	F	F	F	0	0	0	F	F
= 未实现或被预留					F = 在复位期间从非易失性位置 NVOPT 上上载			

图 4-6. Flash 和 EEPROM 选项寄存器 (FOPT)

表 4-9. FOPT 寄存器字段描述

字段	描述
7 KEYEN	<b>后门密钥机制启动</b> — 该位设置为 0 时，后门密钥机制不能用于关闭安全性。后门密钥机制只能从用户（受保护）固件上访问。BDM 命令不能用于写入可能会解锁后门密钥的密钥对比值。若欲了解有关后门密钥机制的更详尽信息，请参见 4.5.9，“安全性”。 0 不允许后门密钥访问。 1 如果用户固件写入一个与非易失性后门密钥（按顺序为 NVBACKKEY 到 NVBACKKEY+7）相匹配的 8 字节值，安全性在下一次 MCU 复位前会暂时关闭。
6 FNORED	<b>向量重定向禁用</b> — 该位为 1 时向量重定向被禁用。 0 向量重定向启用。 1 向量重定向禁用。
5 EPGMOD	<b>EEPROM 分区模式</b> — 该位为 0 时，每个分区分为两个页面（4 字节模式）。该位为 1 时，每个分区在一个页面中（8 字节模式）。 0 每个 EEPROM 分区的一半在页面 0 中而另一半在页面 1 中。 1 每个分区在一个页面中。
1:0 SEC	<b>安全状态代码</b> — 这个 2 位字段决定 MCU 的安全状态，如表 4-10 所示。MCU 处于安全状态时，RAM、EEPROM 和 Flash 中的内容不能通过指令从不安全的源（包括后台调试接口）上访问。后门密钥被成功输入或对 Flash 进行了成功的空白检查后，SEC 将变为 1:0。若欲了解有关安全性的更详尽信息，请参见 4.5.9，“安全性”。

表 4-10. Security States<sup>1</sup>

SEC[1:0]	描述
0:0	安全
0:1	安全
1:0	不安全
1:1	安全

<sup>1</sup> 后门密钥被成功输入或成功地对 Flash 进行了空白检查后，SEC 将变为 1:0。

### 4.5.11.3 Flash 和 EEPROM 配置寄存器 (FCNFG)

	7	6	5	4		3	2	1	0
R	0	EPGSEL	KEYACC	Reserved <sup>1</sup>	0	0	0	1	
W									
复位	0	0	0	1	0	0	0	1	
	= 未实现或被预留								

图 4-7. Flash Configuration Register (FCNFG)

<sup>1</sup> 用户必须在该位上写入一个 1。否则可能会导致意外的行为。

表 4-11. FCNFG 寄存器字段描述

字段	描述
6 EPGSEL	<b>页面选择</b> — 该位选择存储器映象中的哪个 EEPROM 页面可以访问。 0 页面 0 在存储器映象的前台。页面 1 在后台，而且不能访问。 1 页面 1 在存储器映象的前台。页面 0 在后台，而且不能访问。
5 KEYACC	<b>启用访问密钥的写入</b> — 该位启用后门对比密钥的写入。若欲了解有关后门密钥机制的更详尽信息，请参见 4.5.9，“安全性”。 0 写入 0xFFB0–0xFFB7 被解释为 Flash 编程或擦除命令的开始。 1 写入 NVBACKKEY (0xFFB0–0xFFB7) 解释为对比密钥写入。

### 4.5.11.4 Flash 和 EEPROM 保护寄存器 (FPROT and NVPROT)

FPROT 寄存器定义不受编程和擦除操作的影响 Flash 和 EEPROM 分区。

在复位顺序中，FPROT 寄存器从非易失性位置 NVPROT 中上载。若想改变复位序列中将上载的保护，包含 NVPROT 的分区必须不受保护并被擦除，然后才可以对 NVPROT 进行重新编程。

FPROT 位任何时候都可以读取，但只能在受保护区域的范围增加时才可以写入。任何企图缩小受保护内存的 FPROT 写入操作都会被忽略。

尝试修改任何受保护区域的数据将导致保护违反错误，FSTAT 寄存器中将设置 FPVIOL 标记。如果有任何一个分区被保护，就不能进行整体擦除。

	7	6	5	4		3	2	1	0
R	EPS <sup>1</sup>				FPS <sup>1</sup>				
W									
复位									
	本寄存器在复位过程中从非易失性位置 NVPROT 中上载。								

<sup>1</sup> 后台命令可用于修改 FPROT 中的这些位的内容。

图 4-8. Flash 和 EEPROM 保护寄存器 (FPROT)

表 4-12. FPROT 寄存器字段描述

字段	描述
7:6 EPS	<b>EEPROM 保护选择位</b> — 这个 2 位字段决定不能被擦除或编程的受保护 EEPROM 位置。参见表 4-13.
5:0 FPS	<b>Flash 保护选择位</b> — 这个 6 位字段决定不能被擦除或编程的受保护 Flash 位置。参见表 4-14.

表 4-13. EEPROM 块保护

EPS	受保护的地址域	受保护的内存大小 (字节)	受保护的扇区数量
0x3	N/A	0	0
0x2	0x17F0 - 0x17FF	32	4
0x1	0x17E0 - 0x17FF	64	8
0x0	0x17C0-0x17FF	128	16

表 4-14. Flash 块保护

FPS	受保护的地址域	受保护的内存大小 (字节)	受保护的扇区数量
0x3F	N/A	0	0
0x3E	0xFA00-0xFFFF	1.5K	2
0x3D	0xF400-0xFFFF	3K	4
0x3C	0xEE00-0xFFFF	4.5K	6
0x3B	0xE800-0xFFFF	6K	8
...	...	...	...
0x37	0xD000-0xFFFF	12K	16
0x36	0xCA00-0xFFFF	13.5K	18
0x35	0xC400-0xFFFF	15K	20
0x34	0xBE00-0xFFFF	16.5K	22
...	...	...	...
0x2C	0x8E00-0xFFFF	28.5K	38
0x2B	0x8800-0xFFFF	30K	40
0x2A	0x8200-0xFFFF	31.5K	42
0x29	0x7C00-0xFFFF	33K	44
...	...	...	...
0x22	0x5200-0xFFFF	43.5K	58
0x21	0x4C00-0xFFFF	45K	60
0x20	0x4600-0xFFFF	46.5K	62
0x19	0x4000-0xFFFF	48K	64
...	...	...	...

表 4-14. Flash 块保护 (continued)

FPS	受保护的地址域	受保护的内存大小 (字节)	受保护的扇区数量
0x1B	0x2800–0xFFFF	54K	72
0x1A	0x2200–0xFFFF	55.5K	74
0x19	0x1C00–0xFFFF	57K	76
0x18–0x00	0x0000–0xFFFF	64K	86

#### 4.5.11.5 Flash 和 EEPROM 状态寄存器 (FSTAT)

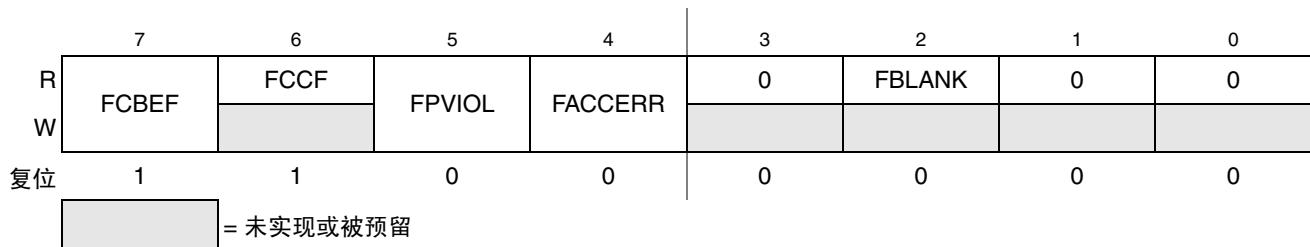


图 4-9. Flash 和 EEPROM 状态寄存器 (FSTAT)

表 4-15. FSTAT 寄存器字段描述

字段	描述
7 FCBEF	<b>命令缓冲器空标记</b> — FCBEF 位用于发出命令。它还用于标识命令缓冲器是空的，因此可以在执行突发编程时执行新的命令顺序。FCBEF 位通过在其中写入 1 或一个突发编程命令被发送到阵列中以进行编程时清除。只有突发编程命令可以被缓冲。 0 命令缓冲器满（没有准备好缓冲额外的命令）。 1 命令缓冲器中可写入新的突发编程命令。
6 FCCF	<b>命令完成标记</b> — FCCF 在命令缓冲器变空而且没有处理任何命令时自动设置。FCCF 在开始执行一个新命令时自动清除（通过将 1 写到 FCBEF 中以登记一个命令）。向 FCCF 写入内容没有任何意义或效果。 0 命令正在执行过程中。 1 所有命令都已完成。
5 FPVIOL	<b>保护规则违反标记</b> — FPVIOL 在发出试图擦除或编程受保护块中的一个位置的命令后自动置 1（错误的命令会被忽略）。FPVIOL 通过向 FPVIOL 中写入 1 来清除。 0 无保护规则违反。 1 有人尝试擦除或编程一个受保护的位置。

表 4-15. FSTAT 寄存器字段描述 (continued)

字段	描述
4 FACCERR	访问错误标记 — FACCERR 在以下情况下自动设置：正确的命令顺序没有严格遵守（错误的命令将被忽略），FCDIV 寄存器初始化之前尝试进行编程或擦除操作，或在命令正在执行时 MCU 进入停止模式。若欲了解会被认为是访问错误的具体操作的更详尽信息，请参见 4.5.6，“访问错误”。FACCERR 通过向 FACCERR 中写入一个 1 来清除。向 FACCERR 中写入 0 没有任何意义或效果。 0 没有访问错误。 1 发生了访问错误。
2 FBLANK	验证为全空（被擦除）标记 — FBLANK 在空白检查命令完成后自动设置为 1（如果整个 Flash 或 EEPROM 阵列被确认已擦除）。FBLANK 通过清除 FCBEF 以写入新的有效命令来清除。向 FBLANK 中写入没有任何意义或效果。 0 在空白检查命令执行完成而且 FCCF = 1 的情况下，FBLANK = 0 表示 Flash 或 EEPROM 阵列未被完全擦除。 1 在空白检查命令执行完成而且 FCCF = 1 的情况下，FBLANK = 1 表示 Flash 或 EEPROM 阵列已完全擦除（全部为 0xFFFF）。

#### 4.5.11.6 Flash 和 EEPROM 命令寄存器 (FCMD)

如表 4-16 所示，正常用户模式下只能识别 6 种命令代码。所有其他命令代码都是非法的，会产生访问错误。请参见 4.5.3，“编程和擦除命令的执行”来了解对 Flash 和 EEPROM 编程及擦除操作的详细描述。

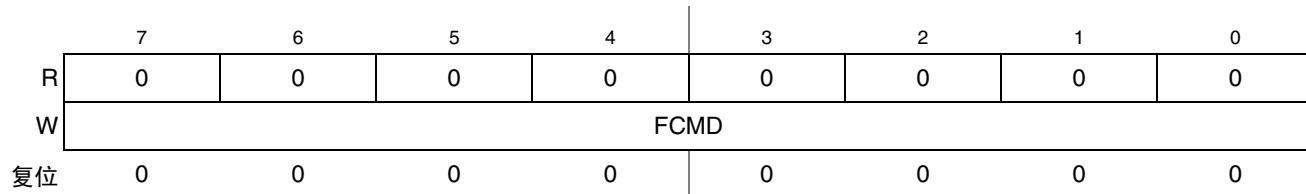


图 4-10. Flash 和 EEPROM 命令寄存器 (FCMD)

表 4-16. Flash 和 EEPROM 命令

命令	FCMD	等同文件标签
空白检查	0x05	mBlank
字节编程	0x20	mByteProg
突发编程	0x25	mBurstProg
分区擦除	0x40	mSectorErase
整体擦除	0x41	mMassErase
分区擦除终止	0x47	mEraseAbort

进行整体擦除操作后没有必要再执行空白检查命令。空白检查只是安全解锁机制的一部分。



# 第 5 章

## 复位、中断和系统总控制

### 5.1 介绍

本章详细介绍 MC9S08DZ60 系列的基本复位和中断机制，以及它们的各种源。外围模块的某些中断源在本数据手册的其他章节中进行了更详细地讨论。本节收集了所有复位和中断源的基本报文，以便参考。而有些复位和中断源，包括计算机正常操作（COP）的看门狗，并不是作为片上外围系统的一部分而有其独立的章节进行介绍。

### 5.2 特性

复位和中断功能包括：

- 多源复位，实现灵活的系统配置和可靠操作
- 复位状态寄存器 (SRS)，显示最新的复位源
- 各个模块的单独中断向量（减少轮询开销），参见表 5-1

### 5.3 MCU 复位

复位 MCU 提供了一条从已知初始条件启动处理的途径。复位期间，大部分控制和状态寄存器被迫使用初始值，并从复位向量 (0xFFFFE:0xFFFF) 上载程序计数器。片上外围模块被禁止，I/O 管脚初始配置为上拉器件被禁止的通用高阻抗输入。条件码寄存器 (CCR) 中的 I 位被设置用来拦截可隐藏中断，以便用户程序有机会对堆栈指针 (SP) 和系统控制设置进行初始化（如需了解中断 (I) 位的相关信息，请参见 CPU 章节）。SP 在复位时强制设为 0x00FF。

MC9S08DZ60 系列有 8 个用于复位的源：

- 加电复位 (POR)
- 外部管脚复位 (PIN)
- 计算机正常操作 (COP) 定时器
- 非法操作码检测 (ILOP)
- 非法地址检测 (ILAD)
- 低电压检测 (LVD)
- 时钟丢失 (LOC)
- 后台调试强制复位 (BDFR)

上述复位源（后台调试强制复位除外）在系统复位状态寄存器 (SRS) 中都有一个相关位。

## 5.4 计算机正常操作 (COP) 看门狗

当系统软件不能正常执行时，COP 看门狗将强制进行系统复位。为了防止从 COP 定时器（当 COP 定时器被使能时）发起系统复位，应用软件必须定期复位 COP 计数器。如果应用程序在超时前未能复位 COP 计数器，这时会生成一个系统复位，强迫系统回到已知起点。

每次复位后，COP 看门狗都会被激活（更多信息请参见 [5.8.4，“系统选项寄存器 1 \(SOPT1\)"](#)）。如果应用中没有使用 COP 看门狗，可以通过清除 SOPT1 的 COPT 位进行禁止。

COP 在设定的超时周期内，通过把 0x55 和 0xAA（按此顺序）写入 SRS 地址来复位 COP 计数器。写入不会对只读 SRS 中的数据造成影响。一旦写入顺序确定，COP 超时周期就会重新开始计算。如果程序在超时周期内未能完成该操作，MCU 将复位。此外，如果向 SRS 写入了非 0x55 或 0xAA 外的其他值，MCU 会被立即复位。

SOPT2 中的 COPCLKS 位（更多信息请参见 [5.8.5，“系统选项寄存器 2 \(SOPT2\)"](#)），设置供 COP 定时器使用的时钟源。时钟源可以是总线时钟或 1 kHz 内部时钟源。对任意一个时钟源来说，都有 3 个由 SOPT1 中的 COPT 控制的相关超时计数器。表 5-6 概括地介绍了 COPCLKS 和 COPT 位的控制功能。COP 看门狗默认设置为 1 kHz 时钟源的最长超时 ( $2^{10}$  周期)。

当选定了总线时钟源后，设置 SOPT2 寄存器中的 COPW 可以实现窗口化 COP。在该模式中，写入 SRS 寄存器来清除 COP 定时器必须发生在所选超时时段的后 25% 的时间内。提前写入会立即复位 MCU。当选择 1 kHz 时钟源时，窗口化 COP 操作不可用。

首次写入 SOPT1 和 SOPT2 寄存器后的任意一种系统复位后，COP 计数器被初始化。SOPT1 和 SOPT2 的后续写入不会对 COP 操作产生影响。即使应用程序使用 COPT、COPCLKS 和 COPW 位的默认复位设置，用户也必须在复位初始化过程中写入 write-once（一次写入）SOPT1 和 SOPT2 寄存器上，以便在该设置中锁定。如果应用程序丢失，使用这种方式可以防止意外修改。

用于服务（清除）COP 计数器的 SRS 写入操作不能放在中断服务程序（ISR）中，因为即使是在主应用程序不能正常执行时，仍然可以定期执行 ISR。

如果选择了总线时钟源，当 MCU 处于后台调试模式或者系统处于停止模式时，COP 计数器不会计数。当 MCU 退出后台调试模式或停止模式时，COP 计数器会重新开始计数。

如果选择 1 kHz 时钟源，那么一旦进入后台调试模式或停止模式时，COP 计数器就会被重新初始化为 0，并在退出后台调试模式或停止模式时会从 0 开始计数。

## 5.5 中断

在执行中断服务程序 (ISR) 前，当前 CPU 状态和寄存器被保存，而在执行中断服务程序 (ISR) 后，保存的 CPU 状态将被恢复。这样可以从中断前的位置重新开始处理。与软件中断 (SWI) 不同 (SWI 由程序指令触发)，中断是由诸如 IRQ 管脚边沿或定时器溢出样的硬件事件触发。调试模块也可以在特定环境下导致 SWI。

如果中断源内的事件发生，将会设置相关的只读状态标记。但不会响应 CPU，除非是由本地中断使能位置为 1 导致并且 CCR 中的 I 位为 0 来允许的中断。CCR 中的全球中断屏蔽 (I 位) 在复位后首次设置会阻止所有可屏蔽的中断源。在清除 I 位之前，用户程序初始化堆栈指针，执行其他系统设置，以便允许 CPU 响应中断。

当 CPU 接收到符合条件的中断请求时，它会在响应中断前先完成当前指令。中断顺序与 SWI 指令的逐周期顺序相同，这个顺序是：

- 在堆栈上保存 CPU 寄存器；
- 在 CCR 中设置 I 位，禁止中断；
- 为当前悬而未决的最高优先级中断获取中断向量；
- 用程序报文的前 3 个字节填写指令队列，程序报文从在中断向量位置上获取的地址开始；

当 CPU 响应中断时，会自动设置 I 位以避免出现中断 ISR 自身的另外一个中断（这也叫做中断嵌套）。在正常情况下，当 CCR 从 ISR 入口处堆栈的值进行恢复时，I 位就恢复为 0。在极个别情况下，I 位可以在 ISR 内部清除（在清除生成中断的状态标志后），所以无需等待第一个业务程序完成，就可以执行另一个中断。该操作仅供有丰富经验的程序员使用，因为它可能导致难以调试发现的程序错误。

中断服务程序以中断恢复 (RTI) 指令作为结束。RTI 指令从堆栈中读取先前保存的报文，将 CCR、A、X 和 PC 寄存器恢复为中断前的值。

### 注意

为了实现与 M68HC08 器件的兼容，H 寄存器不能自动保存和恢复。

建议在中断服务程序 (ISR) 开始时就将 H 推到堆栈上，并在 RTI (用来从 ISR 中恢复) 前立即恢复它。

当 I 位被清除时有多个挂起的中断，处理优先级最高的最先被处理（参见表 5-1）。

### 5.5.1 中断堆栈帧

图 5-1 为堆栈帧的内容和结构。在中断前，堆栈指针 (SP) 指向堆栈的下一个可用字节。CPU 寄存器的当前值保存在堆栈中，以程序计数器 (PCL) 的低阶字节开始，以 CCR 结束。在一次堆栈操作后，SP 指向堆栈的下一个可用位置，该堆栈是比保存 CCR 的地址小一的地址。被堆栈的 PC 值是主程序的指令地址，如果中断没有发生，那么将在下一次中断中实施主程序。

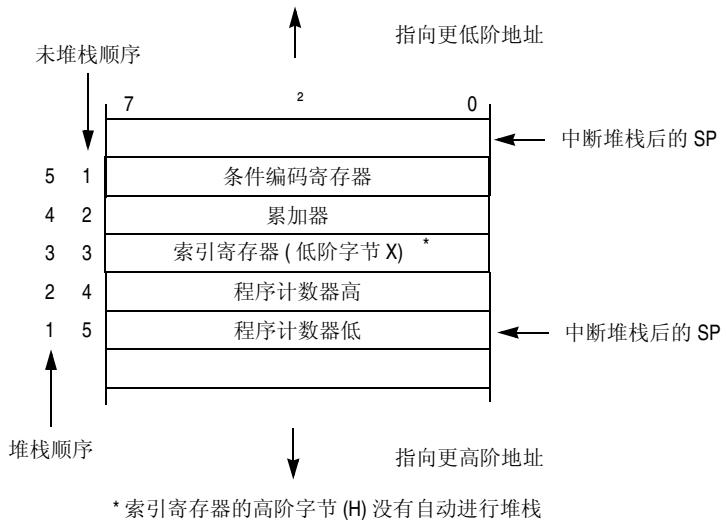


图 5-1. 中断堆栈帧

实施 RTI 指令时，这些值以相反顺序从堆栈中恢复。作为 RTI 顺序的一部分，CPU 通过读取程序报文的 3 个字节（从堆栈中恢复的 PC 地址开始）来填写指令。

在从 ISR 返回前，必须确认 / 清除与中断源对应的状态标记。通常，该标记在 ISR 开始时被清除，这样如果同一源生成另外一个中断，它就可以被登记，从而在当前 ISR 完成后为其提供服务。

## 5.5.2 外部中断请求 (IRQ) 管脚

外部中断由 IRQ 状态和控制寄存器 (IRQSC) 管理。当 IRQ 功能打开后，同步逻辑监控管脚是否发生边沿或边沿加电平的触发事件。当 MCU 处于停止模式且系统时钟关闭时，将使用单独的异步路径，这样 IRQ（如果被打开）就可以唤醒 MCU。

### 5.5.2.1 管脚配置选项

IRQSC 中的 IRQ 管脚激活 (IRQPE) 控制位必须为 1，这样 IRQ 管脚才能作为中断请求 (IRQ) 输入。作为 IRQ 输入，用户可以选择检测边沿或电平 (IRQEDG) 的极性，无论管脚只检测到边沿还是同时检测到边沿和电平 (IRQMOD)，也无论事件触发的中断还是只设置了软件可以轮询的 IRQF 标记。

IRQ 管脚激活后，默认使用内部上拉器件 (IRQPDD = 0)，器件究竟是上拉还是下拉取决于所选的极性。如果用户希望使用外部上拉或下拉，可以在 IRQPDD 中写入 1，以关闭内部器件。

当把 IRQ 管脚配置为 IRQ 输入时，可以使用 BIH 和 BIL 指令来检测 IRQ 管脚上的电平。

#### 注意

这个管脚不包括  $V_{DD}$  的钳位二极管，并且在高于  $V_{DD}$  时不应驱动。  
内部上拉 IRQ 管脚上测量到的电压可能只有  $V_{DD} - 0.7\text{ V}$ 。与该管脚相连的内门一直拉到  $V_{DD}$ 。

### 5.5.2.2 边沿和电平敏感度

IRQMOD 控制位重新配置检测逻辑，这样它就能检测边沿事件和管脚电平。在边沿和电平检测模式中，当检测到边沿时（IRQ 管脚从断言解除改为断言），IRQF 状态标记就被设置，但只要 IRQ 管脚处于断言级，就会连续设置该标记（并且不能清除）。

### 5.5.3 中断向量、源和本地掩码

表 5-1 总结了所有中断源。较高优先级的源位于表格下方。中断服务程序地址的高阶字节位于向量地址栏的第一个地址，中断服务程序地址的低阶字节位于下一个较高阶地址中。

当出现中断时，相关标记位被设置。如果相关的本地中断激活位是 1，中断请求会发送到 CPU。在 CPU 中，如果全球中断屏蔽（CCR 中的 I 位）是 0，CPU 将完成当前指令；堆栈 PCL、PCH、X、A 和 CCR CPU 寄存器；设置 I 位；然后为挂起的最高优先级中断获取中断向量。然后继续处理中断服务程序。

表 5-1. 向量摘要<sup>1</sup>

向量编号	地址 (高 / 低)	向量名称	模块	源	使能	描述
31	0xFFC0/0xFFC1	Vacmp2	ACMP2	ACF	ACIE	模拟比较器 2
30	0xFFC2/0xFFC3	Vacmp1	ACMP1	ACF	ACIE	模拟比较器 1
29	0xFFC4/0xFFC5	Vcantx	MSCAN	TXE[2:0]	TXEIE[2:0]	CAN 发送
28	0xFFC6/0xFFC7	Vcanrx	MSCAN	RXF	RXFIE	CAN 接收
27	0xFFC8/0xFFC9	Vcanerr	MSCAN	CSCIF, OVRIF	CSCIE, OVRIE	CAN 错误
26	0xFFCA/0xFFCB	Vcanwu	MSCAN	WUPIF	WUPIE	CAN 唤醒
25	0xFFCC/0xFFCD	Vrtc	RTC	RTIF	RTIE	实时中断
24	0xFFCE/0xFFCF	Viic	IIC	IICIS	IICIE	IIC 控制
23	0xFFD0/0xFFD1	Vadc	ADC	COCO	AIEN	ADC
22	0xFFD2/0xFFD3	Vport	端口 A,B,D	PTAIF, PTBIF, PTDIF	PTAIE, PTBIE, PTDIE	端口管脚
21	0xFFD4/0xFFD5	Vsci2tx	SCI2	TDRE, TC	TIE, TCIE	SCI2 发送
20	0xFFD6/0xFFD7	Vsci2rx	SCI2	IDLE, LBKDIF, RDRF, RXEDGIF	ILIE, LBKDIE, RIE, RXEDGIE	SCI2 接收
19	0xFFD8/0xFFD9	Vsci2err	SCI2	OR, NF, FE, PF	ORIE, NFIE, FEIE, PFIE	SCI2 错误
18	0xFFDA/0xFFDB	Vsci1tx	SCI1	TDRE, TC	TIE, TCIE	SCI1 发送
17	0xFFDC/0xFFDD	Vsci1rx	SCI1	IDLE, LBKDIF, RDRF, RXEDGIF	ILIE, LBKDIE, RIE, RXEDGIE	SCI1 接收
16	0xFFDE/0xFFDF	Vsci1err	SCI1	OR, NF, FE, PF	ORIE, NFIE, FEIE, PFIE	SCI1 错误
15	0xFFE0/0xFFE1	Vspi	SPI	SPIF, MODF, SPTEF	SPIE, SPIE, SPTIE	SPI
14	0xFFE2/0xFFE3	Vtpm2ovf	TPM2	TOF	TOIE	TPM2 溢出
13	0xFFE4/0xFFE5	Vtpm2ch1	TPM2	CH1F	CH1IE	TPM2 通道 1
12	0xFFE6/0xFFE7	Vtpm2ch0	TPM2	CH0F	CH0IE	TPM2 通道 0
11	0xFFE8/0xFFE9	Vtpm1ovf	TPM1	TOF	TOIE	TPM1 溢出
10	0xFFEA/0xFFEB	Vtpm1ch5	TPM1	CH5F	CH5IE	TPM1 通道 5
9	0xFFEC/0xFFED	Vtpm1ch4	TPM1	CH4F	CH4IE	TPM1 通道 4
8	0xFFEE/0xFFEF	Vtpm1ch3	TPM1	CH3F	CH3IE	TPM1 通道 3
7	0xFFFF0/0xFFFF1	Vtpm1ch2	TPM1	CH2F	CH2IE	TPM1 通道 2
6	0xFFFF2/0xFFFF3	Vtpm1ch1	TPM1	CH1F	CH1IE	TPM1 通道 1
5	0xFFFF4/0xFFFF5	Vtpm1ch0	TPM1	CH0F	CH0IE	TPM1 通道 0
4	0xFFFF6/0xFFFF7	Vlol	MCG	LOLS	LOLIE	锁定丢失
3	0xFFFF8/0xFFFF9	Vlvd	系统控制	LVWF	LVWIE	低压警告
2	0xFFFFA/0xFFFFB	Virq	IRQ	IRQF	IRQIE	IRQ pin
1	0xFFFFC/0xFFFFD	Vswi	内核	SWI 指令	—	软件中断
0	0xFFFFE/0xFFFFF	Vreset	系统控制	COP, LOC, LVD, RESET, ILOP, ILAD, POR, BDFR	COPE CME LVDRE — — — — —	看门狗定时器 时钟丢失 低压检测 外部管脚 非法 opcode 非法地址 加电复位 BDM- 强制复位

<sup>1</sup> 向量优先级采用从低（第一行）到高（最后一行）的顺序表示。例如，Vreset 向量优先级最高。

## 5.6 低电压检测 (LVD) 系统

MC9S08DZ60 系列包括一个防止低电压的系统，以便在电源电压不稳时保护存储器内容、控制 MCU 系统状态。该系统由加电复位 (POR) 电路和 LVD 电路组成，其中 LVD 电路带脱扣电压用于警告和检测。当 SPMSC1 中的 LVDE 设置为 1 时，LVD 电路使能。当进入停止模式时，LVD 禁止，除非 SPMSC1 中设置了 LVDSE。如果同时设置了 LVDSE 和 LVDE，那么 MCU 不能进入 stop2 (进入 stop3)，LVD 激活的 stop3 模式更耗电。

### 5.6.1 加电复位操作

当首次接通 MCU 的电源时，或当电源电压低于加电复位准备电压  $V_{POR}$  时，POR 电路会发起复位。随着电源电压升高，LVD 电路让 MCU 保持复位状态，直到电源高于低压检测低阈值  $V_{LVDL}$ 。POR 后，SRS 中 POR 位和 LVD 位同时被设置。

### 5.6.2 低压检测 (LVD) 复位操作

通过把 LVDRE 设置为 1，可以在检测到低压情况时配置 LVD 以发起复位。低压检测阈值由 LVDV 位决定。在 LVD 复位后，LVD 系统会让 MCU 保持复位状态，直到电源电压高于低压检测阈值。LVD 复位或 POR 后都会在 SRS 寄存器中设置 LVD 位。

### 5.6.3 低压警告 (LVW) 中断操作

LVD 系统有一个低压警告标记，告知用户电源电压正接近低压条件。当检测到低压警告并配置了中断操作 (LVWIE 设置为 1) 时，SPMSC1 中的 LVWF 会被设置，而且会出现 LVW 中断请求。

## 5.7 MCLK 输出

PTAO 管脚共用于 MCLK 时钟输出。如果 MCSEL 位都是 0，MCLK 时钟禁止。若 MCSEL 的任意一个位被设置，无论该管脚的端口数据方向控制位的状态如何，都会导致 PTAO 管脚输出内部 MCU 总线时钟分频后时钟。分频比率由 MCSEL 位决定。管脚的斜率和驱动强度分别由 PTASE0 和 PTADS0 控制。如果斜率控制功能被打开，最大时钟输出频率将会受到限制。如需了解不同情况下的最大频率，请参见电气技术规范。

## 5.8 复位、中断及系统控制寄存器和控制位

直接页面寄存器空间里的一个 8 位寄存器和高页寄存器空间的八个 8 位寄存器都与复位和中断系统有关。

如需了解各寄存器的绝对地址分配，请参见本产品说明第 4 章，“存储器”的表 4-2 和表 4-3。本节只根据它们的名称提及寄存器和控制位。飞思卡尔提供的等式或头文件用来把这些名称转换为相应的绝对地址。

SOPT1 和 SPMSC2 寄存器中的有些控制位与运行模式有关。此处对这些位进行简要描述，更详细的描述参见第 3 章，“操作模式”。

## 5.8.1 中断管脚请求状态和控制寄存器 (IRQSC)

该直接页面寄存器包括用来配置 IRQ 功能、报告状态和确认 IRQ 事件的状态和控制位。

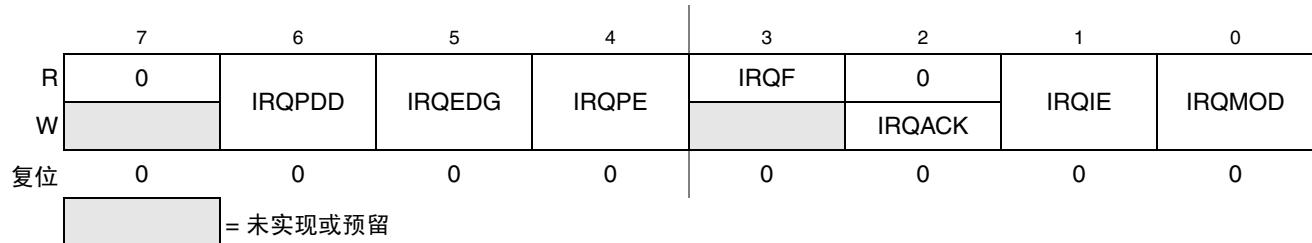


图 5-2. 中断请求状态和控制寄存器 (IRQSC)

表 5-2. IRQSC 寄存器字段描述

字段	描述
6 IRQPDD	<b>中断请求 (IRQ) 上拉器件禁止</b> — IRQ 管脚激活时 (IRQPE = 1)，这个读 / 写控制位用来禁止内部上拉 / 下拉器件，从而允许使用外部器件。 0 如果 IRQPE = 1, IRQ 上拉器件激活； 1 如果 IRQPE = 1, IRQ 上拉器件禁止。
5 IRQEDG	<b>中断请求 (IRQ) 边沿选择</b> — 这个读 / 写控制位选择导致 IRQF 设置的 IRQ 管脚上的边沿或电平的极性。IRQMOD 控制位决定 IRQ 管脚对边沿和电平都敏感还是只对边沿敏感。当 IRQ 管脚作为 IRQ 输入被使能且配置来检测上升边沿时，它就下拉。当 IRQ 管脚作为 IRQ 输入被使能且配置来检测下降边沿时，它就上拉。 0 IRQ 是下降边沿，或者对下降边沿 / 低电平敏感； 1 IRQ 是上升边沿，或者对上升边沿 / 高电平敏感。
4 IRQPE	<b>IRQ 管脚使能</b> — 这个读写控制位使能 IRQ 管脚。当设置了该位时，IRQ 管脚可以用作中断请求。 0 IRQ 管脚禁止； 1 IRQ 管脚使能。
3 IRQF	<b>IRQ 标志</b> — 该只读状态位显示中断请求事件发生的时间。 0 无 IRQ 请求； 1 检测到 IRQ 事件。
2 IRQACK	<b>IRQ 确认</b> — 这个只写位用来确认中断请求事件（写 1 来清除 IRQF）。写入 0 则没有任何意义或影响。读总是返回 0。如果选择了边沿和电平检测 (IRQMOD = 1)，则不能清除 IRQF，并且 IRQ 管脚位于断言级。
1 IRQIE	<b>IRQ 中断使能</b> — 这个读 / 写控制位决定 IRQ 事件是否生成中断请求。 0 当 IRQF = 1 时不产生中断请求（使用轮询）； 1 当 IRQF = 1 时产生中断请求。
0 IRQMOD	<b>IRQ 检测模式</b> — 这个读 / 写控制位选择只边沿检测还是边沿和电平检测。IRQEDG 控制位决定检测为中断请求事件的边沿和电平极性。如需了解更多信息，请参见 5.5.2.2，“边沿和电平敏感度”。 0 只下降边沿或上升边沿的 IRQ 事件； 1 下降边沿和低电平 IRQ 事件或上升边沿和高电平 IRQ 事件。

## 5.8.2 系统复位状态寄存器 (SRS)

该高页寄存器包括只读状态标记，以显示最近复位的源。将 1 写入 SBDFR 寄存器的 BDFR 时，调试主机强制完成复位，SRS 中不设置任何状态位。COP 使能时，在寄存器地址中写入任意值

(0x55 和 0xAA 除外) 都会导致 COP 复位。在该地址中写入 0x55-0xAA 顺序会清除 COP 看门狗定时器，但不会对寄存器内容造成影响。这些位的复位状态取决于导致 MCU 复位的原因。

	7	6	5	4	3	2	1	0
R	POR	PIN	COP	ILOP	ILAD	LOC	LVD	0
W	在 SRS 地址中写入 0x55 和 0xAA, 清除 COP 看门狗定时器							
POR:	1	0	0	0	0	0	1	0
LVD:	u	0	0	0	0	0	1	0
其他 复位:	0	注意 (1)	注意 (1)	注意 (1)	注意 (1)	0	0	0

<sup>1</sup> 复位时活动的任何复位源会造成设置相应的位；复位时不活动的源对应的位将被清除。

图 5-3. 系统复位状态 (SRS)

表 5-3. SRS 寄存器字段描述

字段	描述
7 POR	<b>加电复位</b> — 复位由电源开检测逻辑造成。由于此时内部电源电压在上升，所以还要设置低压复位 (LVD) 状态位以显示已进行的复位，这时内部电源低于 LVD 阈值。 0 非 POR 造成的复位。 1 POR POR 造成的复位。
6 PIN	<b>外部复位管脚</b> — 复位由外部复位管脚上的活动低电平造成。 0 非外部复位管脚造成的复位。 1 外部复位管脚造成的复位。
5 COP	<b>计算机正常操作 (COP)</b> 看门狗 — 复位由 COP 看门狗定时器超时导致。该复位源可以由 COPE = 0 拦截。 0 非 COP 超时造成的复位。 1 超时造成的复位。
4 ILOP	<b>非法操作码</b> — 复位由试图实施未实现或非法操作码导致。如果停止由 SOPT 寄存器里 STOPE = 0 禁止，STOP 指令被视为非法。如果主动后台模式由 BDCSC 寄存器里 ENBDM = 0 禁止，BGND 指令被视为非法。 0 不是由非法操作码造成的复位。 1 非法操作码造成的复位。
3 ILAD	<b>非法地址</b> — 复位由试图在未实现的存储器地址上存取数据或指令导致。 0 不是由非法地址造成的复位。 1 由非法地址造成的复位。
2 LOC	<b>时钟丢失</b> — 复位由外部时钟丢失导致。 0 不是由外部时钟丢失造成的复位。 1 由外部时钟丢失造成的复位。
1 LVD	<b>低压检测</b> — 如果设置了 LVDRE 位且电源降到 LVD 脱扣电压以下，就会发生 LVD 复位。该位也可以由 POR 设置。 0 不是由 LVD 脱扣或 POR 造成的复位。 1 由 LVD 脱扣或 POR 造成的复位。

### 5.8.3 系统后台调试强制复位寄存器 (SBDFR)

这个高页寄存器只包括一个只写控制位。串行后台命令，如 WRITE\_BYTE 必须用来写入 SBDFR。从用户程序写入寄存器的尝试被忽略。读总是返回 0x00。

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0
W								BDFR <sup>1</sup>
复位：	0	0	0	0	0	0	0	0

= 未实现或预留

<sup>1</sup> 只能通过串行后台调试命令，而非用户程序写入 BDFR。

图 5-4. 后台调试强制复位寄存器 (SBDFR)

表 5-4. SBDFR 寄存器字段描述

字段	描述
0 BDFR	后台调试强制复位 — 可以使用串行后台命令，如 WRITE_BYTE，使外部调试主机强制进行目标系统复位。在该位中写入 1 就能强制进行 MCU 复位。该位不能从用户程序中写入。

### 5.8.4 系统选项寄存器 1 (SOPT1)

该高页寄存器是 write-once 寄存器，因此只重视复位后的第一次写入。它可以在任何时候读取。任何后续 SOPT1 写入尝试（有意或无意）都将被忽略，以避免对这些敏感器件的意外修改。该寄存器应在用户复位初始化程序期间写入，以设置期望的控制，即便期望的设置与复位设置相同。

	7	6	5	4	3	2	1	0
R	COPT	STOPE	SCI2PS	IICPS	0	0	0	0
W								

复位： 1 1 0 0 0 0 0 0 0  
= 未实现或预留

图 5-5. 系统选项寄存器 1 (SOPT1)

表 5-5. SOPT1 寄存器字段描述

字段	描述
7:6 COPT[1:0]	COP 看门狗超时 — 这些单次写入有效的位选择 COP 的超时周期。STOP2 中的 COPT 和 COPCLKS 定义 COP 超时周期。参见表 5-6。
5 STOPE	停止模式使能 — 这个单次写入有效的位用来使能停止模式。如果停止模式禁止且用户程序试图实施 STOP 指令，则会强制进行非法操作码复位。 0 停止模式禁止。 1 停止模式使能。

表 5-5. SOPT1 寄存器字段描述

字段	描述
4 SCI2PS	<b>SCI2 管脚选择</b> — 这个单次写入有效的位选择 SCI2 模块的 RxD2 和 TxD2。 0 TxD2 在 PTF0 上, RxD2 在 PTF1 上。 1 TxD2 在 PTE6 上, RxD2 在 PTE7 上。
3 IICPS	<b>IIC 管脚选择</b> — 这个单次写入有效的位选择 IIC 模块的 SCL 和 SDA 管脚的位置。 0 SCL 在 PTF2 上, SDA 在 PTF3 上。 1 SCL 在 PTE4 上, SDA 在 PTE5 上。

表 5-6. COP 配置选项

控制位		时钟源	COP 窗口 <sup>1</sup> 打开 (COPW = 1)	COP 溢出计数
COPCLKS	COPT[1:0]			
无	0:0	无	无	COP 禁止
0	0:1	1 kHz	无	$2^5$ 周期 (32 ms <sup>2</sup> )
0	1:0	1 kHz	无	$2^8$ 周期 (256 ms <sup>1</sup> )
0	1:1	1 kHz	无	$2^{10}$ 周期 (1.024 s <sup>1</sup> )
1	0:1	总线	6144 周期	$2^{13}$ 周期
1	1:0	总线	49,152 周期	$2^{16}$ 周期
1	1:1	总线	196,608 周期	$2^{18}$ 周期

<sup>1</sup> 窗口化 COP 操作要求用户清除所选超时周期后 25% 时间内的 COP 定时器。本栏显示在窗口化 COP 模式中，在 COP 定时器复位前必须提供的时钟最小计数。

<sup>2</sup> 数值采用毫秒单位，并且  $t_{LPO} = 1 \text{ ms}$ 。该值的容限请参见 A.12.1，“控制时序”里的  $t_{LPO}$ 。

## 5.8.5 系统选项寄存器 2 (SOPT2)

这个高页寄存器包含在 MC9S08DZ60 系列器件上配置 MCU 特定功能的位。

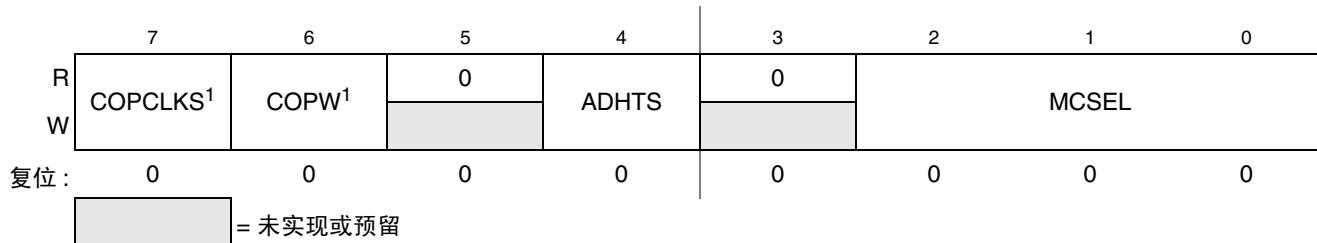


图 5-6. 系统选项寄存器 2 (SOPT2)

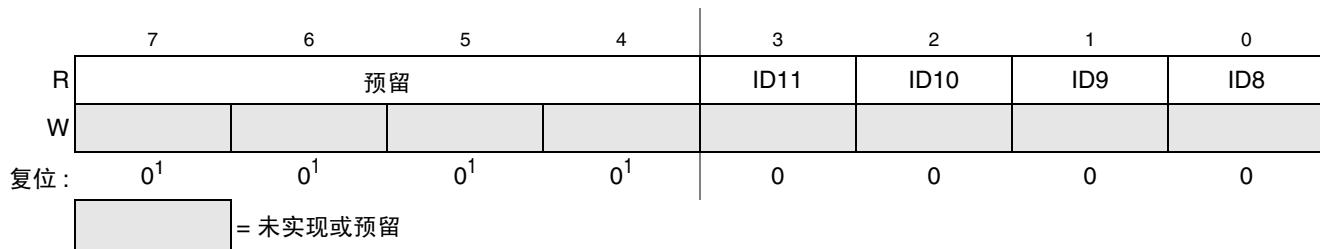
<sup>1</sup> 复位后该位只可以写入一次 (write-once)。其他写入被忽略。

表 5-7. SOPT2 寄存器字段描述

字段	描述
7 COPCLKS	<b>COP 看门狗时钟选择</b> — 这个单次写入有效的位选择 COP 看门狗时钟源。如需了解详细信息，请参见表 5-6。 0 内部 1-kHz 时钟是 COP 源。 1 总线时钟是 COP 源。
6 COPW	<b>COP Window</b> — 窗口 — 这个单次写入有效的位选择 COP 运行模式。设置时，向 SRS 寄存器的 0x55-0xAA 写入顺序必须出现在所选时段的后 25% 时间内，所选时段前 75% 时间内出现的任何 SRS 寄存器写入都将复位 MCU。 0 常规 COP 操作。 1 窗口式 COP 操作。
4 ADHTS	<b>ADC 硬件触发选择</b> — 这个位选择决定使能 ADC 硬件触发 (ADCSC2 寄存器设置了 ADCTRG) 时由哪个硬件触发初始化数模转换器的转换操作。 0 实时计数器 (RTC) 溢出。 1 外部中断请求 (IRQ) 管脚。
2:0 MCSEL	<b>MCLK 分频选择</b> — 这些位在 PTA0 管脚上 MCLK 输出使能，并根据下面的公式选择当 MCSEL 位不全部等于 0 时 MCLK 输出的分频比率。如果 MCSEL 位都等于 0，MCLK 输出禁止。 $MCLK \text{ 频率} = \text{总线时钟频率}^3 (2 * MCSEL)$

## 5.8.6 系统器件识别寄存器 (SDIDH, SDIDL)

这些高页只读寄存器也包括在内，这样主机开发系统就能够识别 HCS08 衍生和修定编号。这使得开发软件能够识别特定的存储器块、寄存器和控制位在目标 MCU 的什么位置。



<sup>1</sup> 硬编码到这些位的修订编号反应当前的芯片修订水平。

图 5-7. 系统器件识别寄存器 — 高 (SDIDH)

表 5-8. SDIDH 寄存器字段描述

字段	描述
3:0 ID[11:8]	<b>部件识别编号</b> — MC9S08DZ60 cμj-MCU 被硬编码为值 0x00E。也请参见表 5-9 里的 ID 位。

	7	6	5	4	3	2	1	0
R	ID7	ID6	ID5	ID4	ID3	ID2	ID1	ID0
W								
复位:	0	0	0	0	1	1	1	0
	[未实现或预留]							

图 5-8. 系统器件识别寄存器 — 低 (SDIDL)

表 5-9. SDIDL 寄存器字段描述

字段	描述
7:0 ID[7:0]	部件识别编号 — MC9S08DZ60 系列 MCU 被硬编码为值 0x00E。也请参见表 5-8 中的 ID 位。

### 5.8.7 系统电源管理状态和控制寄存器 1 (SPMSC1)

这个高页寄存器包含状态位和控制位，以支持低电检测功能，使能 ADC 和 ACMP 模块使用的带隙电压参考。该寄存器应在用户复位初始化程序期间写入，以设置期望的控制，即便期望的设置与复位设置相同。

	7	6	5	4	3	2	1	0
R	LVWF <sup>1</sup>	0	LVIWE	LVDRE <sup>2</sup>	LVDSE	LVDE <sup>2</sup>	0	BGBE
W		LVWACK					0	
复位:	0	0	0	1	1	1	0	0
	[未实现或预留]							

<sup>1</sup> 当出现 V<sub>Supply</sub> 转换低于跳变点或者复位后 V<sub>Supply</sub> 已经低于 V<sub>LVW</sub> 的情况时，要设置 LVWF。

<sup>2</sup> 复位后该位只能写入一次。其他写入被忽略。

图 5-9. 系统电源管理状态和控制寄存器 1 (SPMSC1)

表 5-10. SPMSC1 寄存器字段描述

字段	描述
7 LVWF	低压警告标志 — LVWF 位显示低压警告状态。 0 低压警告未出现。 1 低压警告已出现或出现过。
6 LVWACK	低压警告确认 — 如果 LVWF = 1，就会出现低压条件。为了确认该低压警告，将 1 写入 LVWACK，如果低电压警告不再出现，该操作会将 LVWF 自动清除至 0。
5 LVIWE	低压警告中断使能 — 该位支持 LVWF 硬件中断请求。 0 硬件中断禁止 (使用轮询)。 1 当 LVWF = 1，发出硬件中断请求。

表 5-10. SPMSC1 寄存器字段描述

字段	描述
4 LVDRE	<b>低压检测复位使能</b> — 这个 write-once 位支持 LVD 事件生成硬件复位（假设 LVDE = 1）。 0 LVD 事件不生成硬件复位。 1 当发生使能的低压检测事件时，强制实行 MCU 复位。
3 LVDSE	<b>低压检测停止使能</b> — 假设 LVDE = 1，这个读 / 写位决定当 MCU 处于停止模式时是否操作低压检测功能。 0 停止模式期间低压检测禁止。 1 停止模式期间低压检测使能。
2 LVDE	<b>低压检测使能</b> — 这个 write-once 位支持低压检测逻辑，并且限定该寄存器中的其他位的操作。 0 LVD 逻辑禁止。 1 LVD 逻辑使能。
0 BGBE	<b>带隙缓冲器使能</b> — 这个位支持内部缓冲器，提供带隙电压参考，可供任何一个内部通道上的 ADC 和 ACMP 模块使用。 0 带隙缓冲器禁止。 1 带隙缓冲器使能。

### 5.8.8 系统电源管理状态和控制寄存器 2 (SPMSC2)

该寄存器用来报告低压警告功能状态，配置 MCU 的停止模式行为。该寄存器应在用户复位初始化程序期间写入，以设置期望的控制，即便期望的设置与复位设置相同。

	7	6	5	4		3	2	1	0
R	0	0	LVDV <sup>1</sup>	LVWV	PPDF	0	0	PPDC <sup>2</sup>	
						PPDACK			
加电复位:	0	0	0	0	0	0	0	0	0
LVD 复位:	0	0	u	u	0	0	0	0	0
其他复位:	0	0	u	u	0	0	0	0	0
	= 未实现或预留				u = 未受复位影响				

<sup>1</sup> 加电复位后，这个位只能写入一次。其他写入被忽略。

<sup>2</sup> 复位后这个字节只能写入一次。其他写入被忽略。

图 5-10. 系统电源管理状态和控制寄存器 2 (SPMSC2)

表 5-11. SPMSC2 寄存器字段描述

字段	描述
5 LVDV	<b>低压检测电压选择</b> — 这个单次写入有效的位选择低压检测（LVD）跳变点设置。它还选择警告电压范围。参见表 5-12。
4 LVWV	<b>低压警告电压选择</b> — 这个位选择低压警告（LVW）跳变点电压。参见表 5-12。
3 PPDF	<b>局部断电标志</b> — 这个只读状态位显示 MCU 已经从 STOP2 模式中恢复。 0 MCU 还没有从 STOP2 模式中恢复。 1 MCU 已经从 STOP2 模式中恢复。

表 5-11. SPMSC2 寄存器字段描述

字段	描述
2 PPDACK	局部断电确认 — 向 PPDACK 写入 1 清除 PPDF 位。
0 PPDC	局部断电控制 — 这个单次写入有效的位控制着是选择 STOP2 还是选择 STOP3 模式。 0 Stop3 模式启动。 1 Stop2、局部断电、模式启动。

表 5-12. LVD 和 LVW 跳变点典型值<sup>1</sup>

LVDV:LVWV	LVW 跳变点	LVD 跳变点
0:0	$V_{LVW0} = 2.74 \text{ V}$	$V_{LVD0} = 2.56 \text{ V}$
0:1	$V_{LVW1} = 2.92 \text{ V}$	
1:0	$V_{LVW2} = 4.3 \text{ V}$	$V_{LVD1} = 4.0 \text{ V}$
1:1	$V_{LVW3} = 4.6 \text{ V}$	

<sup>1</sup> 最小最大值请参见附录电气特性。



# 第 6 章

## 并行输入 / 输出控制

本小节解释了与并行输入 / 输出和管脚控制相关的软件控制。MC9S08DZ60 系列有 7 个并行输入 / 输出端口，这 7 个端口总共包含 53 个输入 / 输出管脚和 1 个仅输入管脚。如需了解这些管脚的管脚分配和外部硬件注意事项的更多信息，请参见[第 2 章，“管脚和连接”](#)。

这些管脚中的很多都在片上外围设备中共用，如定时器系统、通信系统和管脚中断，如表 2-1 所示。外围设备模块的优先级把通用输入 / 输出功能的优先级高，因此当使能某个外围设备时，与该共用管脚相关的输入 / 输出功能被禁止。

复位后，共用外围设备功能被禁止，管脚被配置为输入 ( $\text{PTxDn} = 0$ )。每个管脚的管脚控制功能都配置如下：斜率控制使能 ( $\text{PTxSEn} = 1$ )、低驱动强度选定 ( $\text{PTxDSn} = 0$ )、内部上拉被禁止 ( $\text{PTxPEn} = 0$ )。

### 注意

不是所有封装都提供通用输入 / 输出管脚。为了避免从输入引脚浮接抽取过多电流，应用程序中的用户复位初始化程序必须要么使能片上拉器件，要么将未连接管脚的方向更改为输出，使管脚不会浮接。

## 6.1 端口数据和数据方向

通过端口数据寄存器执行并行输入 / 输出读取 / 写入。不管是输入还是输出方向，都由端口数据方向寄存器控制。[图 6-1](#) 中的块状示意图介绍了单个管脚的并行输入 / 输出端口功能。

数据方向控制位 ( $\text{PTxDn}$ ) 决定是否启动相关管脚使用的输出缓冲器，同时控制端口数据寄存器读取的源。相关管脚的输入缓冲器总是处于使能状态，除非管脚用作模拟功能或输出管脚。

当为管脚使能共用数字功能时，输出缓冲器由共用功能控制。但是，数据方向寄存器位将继续控制端口数据寄存器读取的源。

当为管脚使能共用模拟功能时，输出和输出缓冲器都被禁止。当该位为输入位 ( $\text{PTxDn} = 0$ )，输入缓冲器禁止时，任意端口数据位的读数均为 0。总体来说，每当数字功能和模拟功能共用一个管脚时，模拟功能都优先。因此数字和模拟功能同时使能时，管脚由模拟功能控制。

一个不错的编程习惯是在把端口管脚方向修改为输出前就写入端口数据寄存器，这确保不会用在端口数据寄存器内的旧数据值来临时驱动管脚。

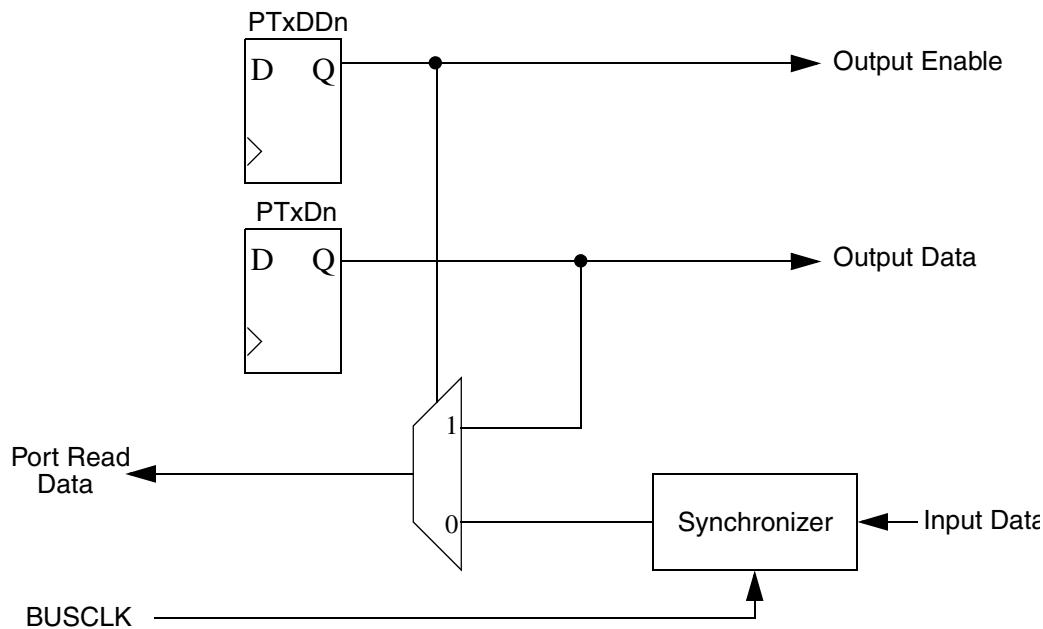


图 6-1. 并行输入 / 输出示意图

## 6.2 上拉、斜率和驱动强度

与并行输入 / 输出端口相关的是位于高页寄存器空间的一套寄存器，它们的操作独立于输入 / 输出寄存器。这些寄存器用来控制管脚的上拉、斜率和驱动强度。

在上拉寄存器（PTxPEn）中设置对应的位可以为所对应的端口管脚使能内部上拉器件。如果并行输入 / 输出控制逻辑或任何外围设备功能将管脚配置为输出，上拉器件就会被禁止，这与对应的上拉寄存器位的状态无关。如果管脚由模拟功能控制，上拉器件同样会被禁止。

在斜率控制寄存器（PTxSEn）中设置对应的位可以为所对应的端口管脚使能斜率控制。该功能启动时，转换控制限制输出的转换速度，减少 EMC 发射。斜率控制对配置为输入的管脚没有任何影响。

### 注意

工程样品和最终成品的斜率复位默认值可能不同。一定要将斜率控制初始化为规定的值，以保证正确的操作。

在驱动强度选择寄存器（PTxDSn）中设置对应的位可以选择一个具有高输出驱动强度的输出管脚。选定好高驱动后，管脚就能够送出和吸收更大的电流。即使可以将每个 I/O 管脚选择为高驱动，用户也必须保证不超出 MCU 的总送出和吸收电流限制。驱动强度选择旨在影响 I/O 管脚的 DC 行为，然而，AC 行为也会受到影响。高驱动允许管脚以与低驱动使能管脚在更小负载中一样的交换速度，驱动更大的负载。正因为如此，EMC 放射可能会由于使能管脚作为高驱动而受到影响。

## 6.3 管脚中断

A 端口、B 和 D 管脚可以配置为外部中断输入，或从停止或等待低功率模式中唤醒 MCU 的外部方式。

各端口中断逻辑的示意图如图 6-2 所示。

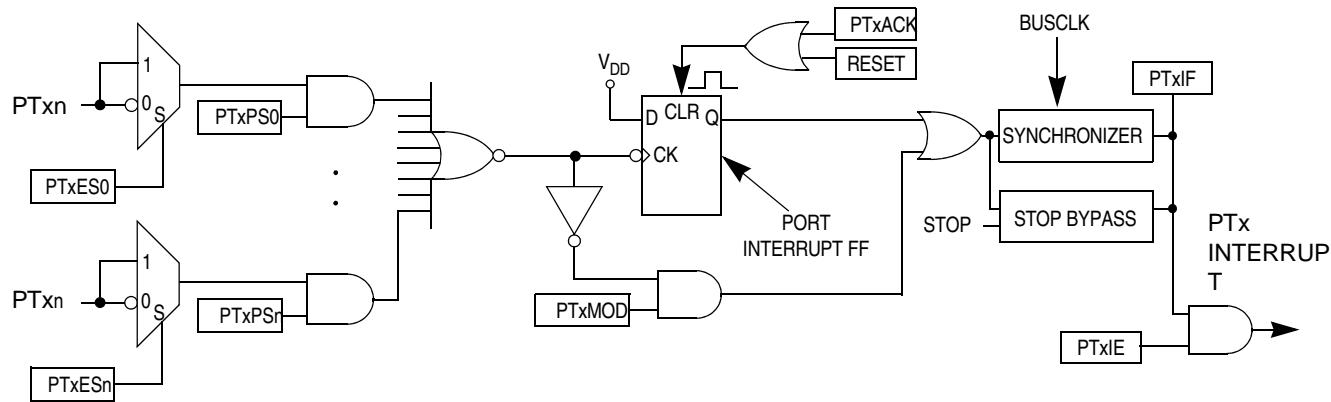


图 6-2. 端口中断示意图

在端口中断管脚选择寄存器（PTxPS）中写入 PTxPSn 位可以独立使能或禁止各端口管脚的中断功能。根据端口中断状态和控制寄存器（PTxSC）中的 PTxMOD 位，可以将各个端口配置为边沿敏感或边沿和电平敏感。边沿敏感可以通过软件编程设定为下降或上升，电平可以为低或高。边沿或者边沿和电平极性敏感度用端口中断边沿选择器（PTxES）中的 PTxESn 位选择。

同步逻辑用来检测边沿。在检测边沿前，已使能的端口输入必须位于无效状态逻辑电平。当端口输入信号在一个总线周期中被视为逻辑 1（无效状态电平），在下一个周期被视为逻辑 0（有效状态电平）时，就能检测到下降边沿。而当输入信号在一个总线周期中被视为逻辑 0，而在下一个周期被视为逻辑 1 时，就能检测到上升边沿。

### 6.3.1 仅边沿敏感度

使能端口管脚上的有效边沿将置位在 PTxSC 中的 PTxIF 位。如果 PTxSC 中置位了 TxIE，CPU 上会出现中断请求。将 1 写入 PTxSC 中的 PTxACK 位会清除 PTxIF。

### 6.3.2 边沿和电平敏感度

使能端口管脚上的有效边沿或电平将置位在 PTxSC 中的 PtxFIF 位。如果 PTxSC 中设置了 PTxIE，CPU 上会出现中断请求。将 1 写入 PTxSC 中的 PTxACK 会清除 PTxIF，假设所有使能端口输入都位于各自的无效状态电平。在试图通过向 PTxACK 写入 1 来进行清除 PTxIF 时，有效状态仍出现在使能的端口上，PTxIF 仍将保持置位状态。

### 6.3.3 上拉 / 下拉电阻器

使用相关 I/O 端口上拉使能寄存器，可以配置端口中断管脚来使用内部上拉 / 下拉电阻器。如果内部电阻器使能，则使用 PTxES 寄存器选择电阻器是上拉（PTxESn = 0）还是下拉（PTxESn = 1）。

### 6.3.4 管脚中断初始化

第一次使能中断管脚时，可能会得到一个错误的中断标记。要防止管脚中断初始化期间的错误中断请求，用户应遵循以下步骤：

1. 清除 PTxSC 里的 PTxACK，屏蔽中断；
2. 在 PTxES 中设置适当的 PTxESn 位，选择管脚极性；
3. 如果使用内部上拉 / 下拉器件，在 PTxPE 中配置相关的使能位；
4. 在 PTxPS 中设置适当的 PTxPSn 位，使能中断管脚；
5. 在 PTxSC 中写入 PTxACK，清除所有错误中断；
6. 在 PTxSC 中置位 PTxIE，使能中断；

## 6.4 停止模式中的管脚行为

执行 STOP 指令后的管脚行为取决于进入的停止模式。不同停止模式的管脚行为的解释如下：

- STOP2 模式是局部断电模式，其中 I/O 寄存器保持执行 STOP 指令前的状态。执行 STOP 指令使 MCU 进入 STOP2 模式前，应将 CPU 寄存器状态和 I/O 寄存器状态保存在 RAM 中。在从 STOP2 模式恢复时，用户访问任何 I/O 前，都应检查 SPMSC2 寄存器里的 PPDF 位状态。如果 PPDF 位是 0，I/O 必须按发生过上电复位那样进行初始化。如果 PPDF 位是 1，外围设备可能要求通过初始化恢复为它们的停止前状态。如果执行 STOP 指令前数据已保留在 RAM 中，就可以完成上述操作。然后用户必须向 SPMSC2 寄存器中的 PPDACK 位写入 1。现在，用户应用程序再次被允许访问 I/O。
- STOP3 模式保留了所有 I/O，因为内部逻辑电路处于通电状态。恢复时，用户可以使用正常的 I/O 功能。

## 6.5 并行 I/O 和管脚控制寄存器

本节介绍与并行 I/O 端口相关的寄存器报文。数据和数据方向寄存器位于存储器映射的 0 页面中。上拉、斜率、驱动强度和中断控制寄存器都位于存储器映射的高页部分。

如需了解所有并行 I/O 及其管脚控制寄存器的绝对地址分配报文，请参见本产品说明的第 4 章，“[存储器](#)”中的表格。本节只按照寄存器和控制位的名称参考寄存器和控制位。飞思卡尔提供的等式或头文件一般用来把这些名称转换到适当的绝对地址。

### 6.5.1 A 端口寄存器

A 端口由下面列出的寄存器控制。

### 6.5.1.1 A 端口数据寄存器 (PTAD)

	7	6	5	4		3	2	1	0
R W	PTAD7	PTAD6	PTAD5	PTAD4		PTAD3	PTAD2	PTAD1	PTAD0
复位：	0	0	0	0		0	0	0	0

图 6-3. A 端口数据寄存器 (PTAD)

表 6-1. PTAD 寄存器字段描述

字段	描述
7:0 PTAD[7:0]	<b>A 端口数据寄存器位</b> — 对于配置为输入的 A 端口管脚，读数返回管脚上的逻辑电平。对于配置为输出的 A 端口管脚，读数返回写入寄存器的最后一个值。 写入值被锁定在本寄存器的所有位中。对于配置为输出的 A 端口管脚，逻辑电平驱动相应的 MCU 管脚。 复位强制 PTAD 都为 0，但是这些 0 未被驱出相应的管脚，因为复位还会将所有端口管脚配置为上拉 / 下拉被禁止的高抗阻输入。

### 6.5.1.2 A 端口数据方向寄存器 (PTADD)

	7	6	5	4		3	2	1	0
R W	PTADD7	PTADD6	PTADD5	PTADD4		PTADD3	PTADD2	PTADD1	PTADD0
复位：	0	0	0	0		0	0	0	0

图 6-4. 端口数据方向寄存器 (PTADD)

表 6-2. PTADD 寄存器字段描述

字段	描述
7:0 PTADD[7:0]	<b>A 端口位的数据方向</b> — 这些读 / 写位控制着 A 端口管脚的方向以及为 PTAD 读数读取的内容。 0 输入（输出驱动被禁止），读数返回管脚值。 1 A 端口位 - 输出驱动使能，PTAD 读数返回 PTADn 内容。

### 6.5.1.3 A 端口上拉使能寄存器 (PTAPE)

	7	6	5	4		3	2	1	0
R	PTAPE7	PTAPE6	PTAPE5	PTAPE4		PTAPE3	PTAPE2	PTAPE1	PTAPE0
W									

复位： 0 0 0 0 0 0 0 0 0 0

图 6-5. A 端口寄存器的内部上拉使能 (PTAPE)

表 6-3. PTAPE 寄存器字段描述

字段	描述
7:0 PTAPE[7:0]	<b>A 端口内部上拉使能位</b> — 对于 PTA 管脚，这些控制位决定着为相关 PTA 管脚使能内部上拉还是使能下拉器件。对于配置为输出的 A 端口管脚，这些位不会产生影响，同时内部上拉器件被禁止。 0 A 端口位 - 内部上拉 / 下拉器件被禁止。 1 A 端口位 - 内部上拉 / 下拉器件使能。

#### 注意

只有当使用管脚中断功能且配置了相应的边沿选择和管脚选择功能时，才能使用下拉器件。

### 6.5.1.4 A 端口斜率使能寄存器 (PTASE)

	7	6	5	4		3	2	1	0
R	PTASE7	PTASE6	PTASE5	PTASE4		PTASE3	PTASE2	PTASE1	PTASE0
W									

复位： 0 0 0 0 0 0 0 0 0 0

图 6-6. A 端口寄存器的斜率使能 (PTASE)

表 6-4. PTASE 寄存器字段描述

字段	描述
7:0 PTASE[7:0]	<b>A 端口位输出斜率使能</b> — 这些控制位决定着是否为相关 PTA 管脚使能输出斜率控制。对于配置为输入的 A 端口管脚，这些位不会产生任何影响。 0 A 端口位 - 输出斜率控制被禁止。 1 A 端口位 - 输出斜率控制使能。

注意：工程样品和最终成品的斜率复位默认值可能不同。一定要将斜率控制初始化为所需的值，以确保正确的操作。

### 6.5.1.5 A 端口驱动强度选择寄存器 (PTADS)

	7	6	5	4	3	2	1	0
R	PTADS7	PTADS6	PTADS5	PTADS4	PTADS3	PTADS2	PTADS1	PTADS0
W	0	0	0	0	0	0	0	0

复位： 0 0 0 0 0 0 0 0 0

图 6-7. A 端口寄存器的驱动强度选择 (PTADS)

表 6-5. PTADS 寄存器字段描述

字段	描述
7:0 PTADS[7:0]	<b>A 端口位的输出驱动强度选择</b> — 这些控制位为相关 PTA 管脚选择低输出驱动和高输出驱动。对于配置为输入的 A 端口管脚，这些位不会产生任何影响。 0 A 端口位 - 选择的低输出驱动强度。 1 A 端口位 - 选择的高输出驱动强度。

### 6.5.1.6 A 端口中断状态和控制寄存器 (PTASC)

	7	6	5	4	3	2	1	0
R	0	0	0	0	PTAIF	0	PTAIE	PTAMOD
W						PTAACK		
复位：	0	0	0	0	0	0	0	0

= 未实现或已被预留

图 6-8. A 端口中断状态和控制寄存器 (PTASC)

表 6-6. PTASC 寄存器字段描述

字段	描述
3 PTAIF	<b>A 端口中断标志</b> — PTAIF 显示是否检测到 A 端口中断。写入对 PTAIF 没有任何影响。 0 未检测到 A 端口中断。 1 检测到 A 端口中断。
2 PTAACK	<b>A 端口中断确认</b> — 向 PTAACK 写入 1 是标记清除机制的一部分。PTAACK 的读数总为 0。
1 PTAIE	<b>A 端口中断使能</b> — PTAIE 决定是否请求 A 端口中断。 0 A 端口中断请求禁止。 1 A 端口中断请求使能。
0 PTAMOD	<b>A 端口检测模式</b> — PTAMOD (同 PTAES 位一起) 控制着 A 端口中断管脚的检测模式。 0 A 端口管脚只检测边沿。 1 A 端口管脚同时检测边沿和电平。

### 6.5.1.7 A 端口中断管脚选择寄存器 (PTAPS)

	7	6	5	4		3	2	1	0
R W	PTAPS7	PTAPS6	PTAPS5	PTAPS4		PTAPS3	PTAPS2	PTAPS1	PTAPS0
复位：	0	0	0	0		0	0	0	0

图 6-9. A 端口中断管脚选择寄存器 (PTAPS)

表 6-7. PTAPS 寄存器字段描述

字段	描述
7:0 PTAPS[7:0]	<b>A 端口中断管脚选择</b> — 每个 PTAPSn 位都使能相应的 A 端口中断管脚。 0 管脚禁止中断。 1 管脚允许中断。

### 6.5.1.8 A 端口中断边沿选择寄存器 (PTAES)

	7	6	5	4		3	2	1	0
R W	PTAES7	PTAES6	PTAES5	PTAES4		PTAES3	PTAES2	PTAES1	PTAES0
复位：	0	0	0	0		0	0	0	0

图 6-10. A 端口边沿选择寄存器 (PTAES)

表 6-8. PTAES 寄存器字段描述

字段	描述
7:0 PTAES[7:0]	<b>A 端口边沿选择</b> — 每个 PTAESn 位都具有双重功能，选择中断边沿的极性以及选择上拉或下拉器件（使能的话）。 0 上拉器件与相关的管脚相连，检测中断生成的下降边沿 / 低电平。 1 下拉器件与相关的管脚相连，检测中断生成的上升边沿 / 高电平。

### 6.5.2 B 端口寄存器

B 端口由下面列出的寄存器控制。

### 6.5.2.1 B 端口数据寄存器 (PTBD)

	7	6	5	4		3	2	1	0
R W	PTBD7	PTBD6	PTBD5	PTBD4		PTBD3	PTBD2	PTBD1	PTBD0
复位：	0	0	0	0		0	0	0	0

图 6-11. B 端口数据寄存器 (PTBD)

表 6-9. PTBD 寄存器字段描述

字段	描述
7:0 PTBD[7:0]	<b>B 端口数据寄存器位</b> — 对于配置为输入的 B 端口管脚，读数返回管脚上的逻辑电平。对于配置为输出的 B 端口管脚，读数返回写入寄存器的最后一个值。 写入值被锁定在本寄存器的所有位中。对于配置为输出的 B 端口管脚，逻辑电平被输出到相应的 MCU 管脚。 复位强制 PTBD 都为 0，但是这些 0 未被输出到相应的管脚，因为复位还会将所有端口管脚配置为上拉 / 下拉被禁止的高抗阻输入。

### 6.5.2.2 B 端口数据方向寄存器 (PTBDD)

	7	6	5	4		3	2	1	0
R W	PTBDD7	PTBDD6	PTBDD5	PTBDD4		PTBDD3	PTBDD2	PTBDD1	PTBDD0
复位：	0	0	0	0		0	0	0	0

图 6-12. B 端口数据方向寄存器 (PTBDD)

表 6-10. PTBDD 寄存器字段描述

字段	描述
7:0 PTBDD[7:0]	<b>B 端口位的数据方向</b> — 这些读 / 写位控制着 B 端口管脚的方向以及为 PTBD 读数读取的内容。 0 输入（输出驱动被禁止），读数返回管脚值。 1 B 端口位 - 输出驱动使能，PTBD 读数返回 PTBDn 内容。

### 6.5.2.3 B 端口上拉使能寄存器 (PTBPE)

	7	6	5	4		3	2	1	0
R	PTBPE7	PTBPE6	PTBPE5	PTBPE4		PTBPE3	PTBPE2	PTBPE1	PTBPE0
W									

复位： 0 0 0 0 0 0 0 0 0 0

图 6-13. B 端口寄存器内部上拉使能 (PTBPE)

表 6-11. PTBPE 寄存器字段描述

字段	描述
7:0 PTBPE[7:0]	<b>B 端口内部上拉使能位</b> — 这些控制位决定着为相关 PTB 管脚使能内部上拉还是使能下拉器件。对于配置为输出的 B 端口管脚，这些位不会产生影响，同时内部上拉器件被禁止。 0 B 端口位 - 内部上拉 / 下拉器件被禁止。 1 B 端口位 - 内部上拉 / 下拉器件使能。

#### 注意

只有当使用管脚中断功能且配置了相应的边沿选择和管脚选择功能时，才能使用下拉器件。

### 6.5.2.4 B 端口斜率使能寄存器 (PTBSE)

	7	6	5	4		3	2	1	0
R	PTBSE7	PTBSE6	PTBSE5	PTBSE4		PTBSE3	PTBSE2	PTBSE1	PTBSE0
W									

复位： 0 0 0 0 0 0 0 0 0 0

图 6-14. B 端口寄存器斜率使能 (PTBSE)

表 6-12. PTBSE 寄存器字段描述

字段	描述
7:0 PTBSE[7:0]	<b>B 端口位输出斜率使能</b> — 这些控制位决定着是否为相关 PTB 管脚使能输出斜率控制。对于配置为输入的 B 端口管脚，这些位不会产生任何影响。 0 B 端口位 - 输出斜率控制禁止。 1 B 端口位 - 输出斜率控制使能。

注意：工程样品和最终成品的斜率复位默认值可能不同。一定要将斜率控制初始化为所需的值，以确保正确的操作。

### 6.5.2.5 B 端口驱动强度选择寄存器 (PTBDS)

	7	6	5	4		3	2	1	0
R	PTBDS7	PTBDS6	PTBDS5	PTBDS4		PTBDS3	PTBDS2	PTBDS1	PTBDS0
W									

复位： 0 0 0 0 0 0 0 0 0 0

图 6-15. B 端口寄存器驱动强度选择 (PTBDS)

表 6-13. PTBDS 寄存器字段描述

字段	描述
7:0 PTBDS[7:0]	<b>B 端口位的输出驱动强度选择</b> — 这些控制位为相关 PTB 管脚选择低输出驱动和高输出驱动。对于配置为输入的 B 端口管脚，这些位不会产生任何影响。 0 B 端口位 - 选择的低输出驱动强度。 1 B 端口位 - 选择的高输出驱动强度。

### 6.5.2.6 B 端口中断状态和控制寄存器 (PTBSC)

	7	6	5	4		3	2	1	0
R	0	0	0	0		PTBIF	0		
W							PTBACK	PTBIE	PTBMOD
复位：	0	0	0	0		0	0	0	0

= 未实现或已被预留

图 6-16. B 端口中断状态和控制寄存器 (PTBSC)

表 6-14. PTBSC 寄存器字段描述

字段	描述
3 PTBIF	<b>B 端口中断标志</b> — PTBIF 显示是否检测到 B 端口中断。写入对 PTBIF 没有任何影响。 0 未检测到 B 端口中断。 1 检测到 B 端口中断。
2 PTBACK	<b>B 端口中断确认</b> — 向 PTBACK 写入 1 是标记清除机制的一部分。PTBACK 的读数总为 0。
1 PTBIE	<b>B 端口中断使能</b> — PTBIE 决定是否请求 B 端口中断。 0 B 端口中断请求禁止。 1 B 端口中断请求使能。
0 PTBMOD	<b>B 端口检测模式</b> — PTBMOD (同 PTBES 位一起) 控制着 B 端口中断管脚的检测模式。 0 B 端口管脚只检测边沿。 1 B 端口管脚同时检测边沿和电平。

### 6.5.2.7 B 端口中断管脚选择寄存器 (PTBPS)

	7	6	5	4		3	2	1	0
R W	PTBPS7	PTBPS6	PTBPS5	PTBPS4		PTBPS3	PTBPS2	PTBPS1	PTBPS0
复位：	0	0	0	0		0	0	0	0

图 6-17. B 端口中断管脚选择寄存器 (PTBPS)

表 6-15. PTBPS 寄存器字段描述

字段	描述
7:0 PTBPS[7:0]	<b>B 端口中断管脚选择</b> — 每个 PTBPSn 位都使能相应的 B 端口中断管脚。 0 管脚禁止中断。 1 管脚允许中断。

### 6.5.2.8 B 端口边沿选择寄存器 (PTBES)

	7	6	5	4		3	2	1	0
R W	PTBES7	PTBES6	PTBES5	PTBES4		PTBES3	PTBES2	PTBES1	PTBES0
复位：	0	0	0	0		0	0	0	0

图 6-18. B 端口边沿选择寄存器 (PTBES)

表 6-16. PTBES 寄存器字段描述

字段	描述
7:0 PTBES[7:0]	<b>B 端口边沿选择</b> — 每个 PTBESn 位都具有双重功能，选择活动中断边沿的极性以及选择上拉或下拉器件（使能的话）。 0 上拉器件与相关的管脚相连，检测中断生成的下降边沿 / 低电平。 1 下拉器件与相关的管脚相连，检测中断生成的上升边沿 / 高电平。

### 6.5.3 C 端口寄存器

C 端口由下列寄存器控制。

### 6.5.3.1 C 端口数据寄存器 (PTCD)

	7	6	5	4	3	2	1	0
R W	PTCD7	PTCD6	PTCD5	PTCD4	PTCD3	PTCD2	PTCD1	PTCD0
复位：	0	0	0	0	0	0	0	0

图 6-19. C 端口数据寄存器 (PTCD)

表 6-17. PTCD 寄存器字段描述

字段	描述
7:0 PTCD[7:0]	<b>C 端口数据寄存器位</b> — 对于配置为输入的 C 端口管脚，读数返回管脚上的逻辑电平。对于配置为输出的 C 端口管脚，读数返回写入寄存器的最后一个值。 写入值被锁定在本寄存器的所有位中。对于配置为输出的 C 端口管脚，逻辑电平被输出到相应的 MCU 管脚。 复位强制 PTCD 都为 0，但是这些 0 未被输出到驱出相应的管脚，因为复位还会将所有端口管脚配置为上拉 / 下拉禁止的高抗阻输入。

### 6.5.3.2 C 端口数据方向寄存器 (PTCDD)

	7	6	5	4	3	2	1	0
R W	PTCDD7	PTCDD6	PTCDD5	PTCDD4	PTCDD3	PTCDD2	PTCDD1	PTCDD0
复位：	0	0	0	0	0	0	0	0

图 6-20. C 端口数据方向寄存器 (PTCDD)

表 6-18. PTCDD 寄存器字段描述

字段	描述
7:0 PTCDD[7:0]	<b>C 端口位的数据方向</b> — 这些读 / 写位控制着 C 端口管脚的方向以及为 PTCD 读数读取的内容。 0 输入（输出驱动禁止），读数返回管脚值。 1 C 端口位 - 输出驱动使能，PTCD 读数返回 PTCDn 内容。

### 6.5.3.3 C 端口上拉使能寄存器 (PTCPE)

	7	6	5	4		3	2	1	0
R W	PTCPE7	PTCPE6	PTCPE5	PTCPE4		PTCPE3	PTCPE2	PTCPE1	PTCPE0
复位：	0	0	0	0		0	0	0	0

图 6-21. C 端口寄存器内部上拉使能 (PTCPE)

表 6-19. PTCPE 寄存器字段描述

字段	描述
7:0 PTCPE[7:0]	<b>C 端口内部上拉使能位</b> — 这些控制位决定着是否为相关 PTC 管脚使能内部上拉器件。对于配置为输出的 C 端口管脚，这些位不会产生影响，同时内部拉器件被禁止。 0 C 端口位 - 内部上拉器件被禁止。 1 C 端口位 - 内部上拉器件使能。

## 注意

只有当使用管脚中断功能且配置了相应的边沿选择和管脚选择功能时，才能使用下拉器件。

### 6.5.3.4 C 端口斜率使能寄存器 (PTCSE)

	7	6	5	4		3	2	1	0
R W	PTCSE7	PTCSE6	PTCSE5	PTCSE4		PTCSE3	PTCSE2	PTCSE1	PTCSE0
复位：	0	0	0	0		0	0	0	0

图 6-22. C 端口寄存器斜率使能 (PTCSE)

表 6-20. PTCSE 寄存器字段描述

字段	描述
7:0 PTCSE[7:0]	<b>C 端口位输出斜率使能</b> — 这些控制位决定着是否为相关 PTC 管脚使能输出斜率控制。对于配置为输入的 C 端口管脚，这些位不会产生任何影响。 0 C 端口位 - 输出斜率控制禁止。 1 C 端口位 - 输出斜率控制使能。

注意：工程样品和最终成品的斜率复位默认值可能不同。一定要将斜率控制初始化为所需的值，以确保正确的操作。

### 6.5.3.5 C 端口驱动强度选择寄存器 (PTCDS)

	7	6	5	4	3	2	1	0
R W	PTCDS7	PTCDS6	PTCDS5	PTCDS4	PTCDS3	PTCDS2	PTCDS1	PTCDS0
复位：	0	0	0	0	0	0	0	0

图 6-23. C 端口寄存器的驱动强度选择 (PTCDS)

表 6-21. PTCDS 寄存器字段描述

字段	描述
7:0 PTCDS[7:0]	<b>C 端口位的输出驱动强度选择</b> — 这些控制位为相关 PTC 管脚选择低输出驱动和高输出驱动。对于配置为输入的 C 端口管脚，这些位不会产生任何影响。 0 C 端口位 - 选择的低输出驱动强度。 1 C 端口位 - 选择的高输出驱动强度。

### 6.5.4 D 端口寄存器

D 端口由下列寄存器控制。

#### 6.5.4.1 D 端口数据寄存器 (PTDD)

	7	6	5	4	3	2	1	0
R W	PTDD7	PTDD6	PTDD5	PTDD4	PTDD3	PTDD2	PTDD1	PTDD0
复位：	0	0	0	0	0	0	0	0

图 6-24. D 端口数据寄存器 (PTDD)

表 6-22. PTDD 寄存器字段描述

字段	描述
7:0 PTDD[7:0]	<b>D 端口数据寄存器位</b> — 对于配置为输入的 D 端口管脚，读数返回管脚上的逻辑电平。对于配置为输出的 D 端口管脚，读数返回写入寄存器的最后一个值。 写入值被锁定在本寄存器的所有位中。对于配置为输出的 D 端口管脚，逻辑电平被输出到相应的 MCU 管脚。 复位强制 PTDD 都为 0，但是这些 0 未被输出到相应的管脚，因为复位还会将所有端口管脚配置为上拉 / 下拉禁止的高抗阻输入。

### 6.5.4.2 D 端口数据方向寄存器 (PTDDD)

	7	6	5	4		3	2	1	0
R W	PTDDD7	PTDDD6	PTDDD5	PTDDD4	PTDDD3	PTDDD2	PTDDD1	PTDDD0	
复位：	0	0	0	0	0	0	0	0	0

图 6-25. D 端口数据方向寄存器 (PTDDD)

表 6-23. PTDDD 寄存器字段描述

字段	描述
7:0 PTDDD[7:0]	<b>D 端口位的数据方向</b> — 这些读 / 写位控制着 D 端口管脚的方向以及为 PTDD 读数读取的内容。 0 输入（输出驱动被禁止），读数返回管脚值。 1 D 端口位 - 输出驱动使能，PTDD 读数返回 PTDDn 内容。

### 6.5.4.3 D 端口上拉使能寄存器 (PTDPE)

	7	6	5	4		3	2	1	0
R W	PTDPE7	PTDPE6	PTDPE5	PTDPE4	PTDPE3	PTDPE2	PTDPE1	PTDPE0	
复位：	0	0	0	0	0	0	0	0	0

图 6-26. D 端口寄存器内部上拉使能 (PTDPE)

表 6-24. PTDPE 寄存器字段描述

字段	描述
7:0 PTDPE[7:0]	<b>D 端口内部上拉使能位</b> — 这些控制位决定相关的 PTD 管脚是使能内部上拉还是起用下拉器件。对于配置为输出的 D 端口管脚，这些位不会产生影响，同时内部拉器件被禁止。 0 D 端口位 - 内部上拉 / 下拉器件被禁止。 1 D 端口位 - 内部上拉 / 下拉器件使能。

#### 注意

只有当使用管脚中断功能且配置了相应的边沿选择和管脚选择功能时，才能使用下拉器件。

### 6.5.4.4 D 端口斜率使能寄存器 (PTDSE)

	7	6	5	4		3	2	1	0
R W	PTDSE7	PTDSE6	PTDSE5	PTDSE4		PTDSE3	PTDSE2	PTDSE1	PTDSE0
复位：	0	0	0	0		0	0	0	0

图 6-27. D 端口寄存器斜率使能 (PTDSE)

表 6-25. PTDSE 寄存器字段描述

字段	描述
7:0 PTDSE[7:0]	<b>D 端口位输出斜率使能</b> — 这些控制位决定着是否为相关的 PTD 管脚使能输出斜率控制。对于配置为输入的 D 端口管脚，这些位不会产生任何影响。 0 D 端口位 - 输出斜率控制禁止。 1 D 端口位 - 输出斜率控制使能。

注意：工程样品和最终成品的斜率复位默认值可能不同。一定要将斜率控制初始化为所需的值，以确保正确的操作。

### 6.5.4.5 D 端口驱动强度选择寄存器 (PTDDS)

	7	6	5	4		3	2	1	0
R W	PTDDS7	PTDDS6	PTDDS5	PTDDS4		PTDDS3	PTDDS2	PTDDS1	PTDDS0
复位：	0	0	0	0		0	0	0	0

图 6-28. D 端口寄存器驱动强度选择 (PTDDS)

表 6-26. PTDDS 寄存器字段描述

字段	描述
7:0 PTDDS[7:0]	<b>D 端口位输出驱动强度选择</b> — 这些控制位为相关 PTD 管脚选择低输出驱动和高输出驱动。对于配置为输入的 D 端口管脚，这些位不会产生任何影响。 0 D 端口位 - 选择的低输出驱动强度。 1 D 端口位 - 选择的高输出驱动强度。

### 6.5.4.6 D 端口中断状态和控制寄存器 (PTDSC)

	7	6	5	4		3	2	1	0
R	0	0	0	0	PTDIF	0			
W						PTDACK		PTDIE	PTDMOD
复位：	0	0	0	0		0	0	0	0
	= 未实现或已被预留								

图 6-29. D 端口中断状态和控制寄存器 (PTDSC)

表 6-27. PTDSC 寄存器字段描述

字段	描述
3 PTDIF	<b>D 端口中断标志</b> — PTDIF 显示是否检测到 D 端口中断。写入对 PTDIF 没有任何影响。 0 未检测到 D 端口中断。 1 检测到 D 端口中断。
2 PTDACK	<b>D 端口中断确认</b> — 向 PTDACK 写入 1 是标记清除机制的一部分。PTDACK 的读数总为 0。
1 PTDIE	<b>D 端口中断功能</b> — PTDIE 决定是否请求 D 端口中断。 0 D 端口中断请求禁止。 1 D 端口中断请求使能。
0 PTDMOD	<b>D 端口检测模式</b> — PTDMOD (同 PTDES 位一起) 控制着 D 端口中断管脚的检测模式。 0 D 端口管脚只检测边沿。 1 D 端口管脚同时检测边沿和电平。

### 6.5.4.7 D 端口中断管脚选择寄存器 (PTDPS)

	7	6	5	4		3	2	1	0
R	PTDPS7	PTDPS6	PTDPS5	PTDPS4	PTDPS3	PTDPS2	PTDPS1	PTDPS0	
W									
复位：	0	0	0	0	0	0	0	0	0
	= 未实现或已被预留								

图 6-30. D 端口中断管脚选择寄存器 (PTDPS)

表 6-28. PTDPS 寄存器字段描述

字段	描述
7:0 PTDPS[7:0]	<b>D 端口中断管脚选择</b> — 每个 PTDPSn 位都允许相应的 D 端口中断管脚。 0 管脚禁止中断。 1 管脚允许中断。

### 6.5.4.8 D 端口中断边沿选择寄存器 (PTDES)

	7	6	5	4	3	2	1	0
R W	PTDES7	PTDES6	PTDES5	PTDES4	PTDES3	PTDES2	PTDES1	PTDES0
复位：	0	0	0	0	0	0	0	0

图 6-31. D 端口边沿选择寄存器 (PTDES)

表 6-29. PTDES 寄存器字段描述

字段	描述
7:0 PTDES[7:0]	<b>D 端口边沿选择</b> — 每个 PTDESn 位都具有双重功能，选择活动中断边沿的极性以及选择上拉或下拉器件（使能的话）。 0 上拉器件与相关的管脚相连，检测中断生成的下降边沿 / 低电平。 1 下拉器件与相关的管脚相连，检测中断生成的上升边沿 / 高电平。

### 6.5.5 E 端口寄存器

E 端口由下列寄存器控制。

#### 6.5.5.1 E 端口数据寄存器 (PTED)

	7	6	5	4	3	2	1	0
R W	PTED7	PTED6	PTED5	PTED4	PTED3	PTED2	PTED1 <sup>1</sup>	PTED0
复位：	0	0	0	0	0	0	0	0

图 6-32. E 端口数据寄存器 (PTED)

<sup>1</sup> 读取这个位总是要返回相关管脚的管脚值，与端口数据方向位中保存的值无关。

表 6-30. PTED 寄存器字段描述

字段	描述
7:0 PTED[7:0]	<b>E 端口数据寄存器位</b> — 对于配置为输入的 E 端口管脚，读数返回管脚上的逻辑电平。对于配置为输出的 E 端口管脚，读数返回写入寄存器的最后一个值。 写入值被锁定在本寄存器的所有位中。对于配置为输出的 E 端口管脚，逻辑电平被输出到驱出相应的 MCU 管脚。 复位强制 PTED 都为 0，但是这些 0 未被输出到驱出相应的管脚，因为复位还会将所有端口管脚配置为上拉 / 下拉禁止的高抗阻输入。

### 6.5.5.2 E 端口数据方向寄存器 (PTEDD)

	7	6	5	4	3	2	1	0
R W	PTEDD7	PTEDD6	PTEDD5	PTEDD4	PTEDD3	PTEDD2	PTEDD1 <sup>1</sup>	PTEDD0
复位：	0	0	0	0	0	0	0	0

图 6-33. E 端口数据方向寄存器 (PTEDD)

<sup>1</sup> PTEDD1 对输入 PTE1 管脚没有影响。

表 6-31. PTEDD 寄存器字段描述

字段	描述
7:0 PTEDD[7:0]	<b>E 端口位的数据方向</b> — 这些读 / 写位控制着 E 端口管脚的方向以及为 PTED 读数读取的内容。 0 输入（输出驱动被禁止），读数返回管脚值。 1 E 端口位 - 输出驱动使能，PTED 读取返回 PTEDn 内容。

### 6.5.5.3 E 端口上拉使能寄存器 (PTEPE)

	7	6	5	4	3	2	1	0
R W	PTEPE7	PTEPE6	PTEPE5	PTEPE4	PTEPE3	PTEPE2	PTEPE1	PTEPE0
复位：	0	0	0	0	0	0	0	0

图 6-34. E 端口寄存器内部上拉使能 (PTEPE)

表 6-32. PTEPE 寄存器字段描述

字段	描述
7:0 PTEPE[7:0]	<b>E 端口内部上拉使能位</b> — 这些控制位决定是否为相关的 PTE 管脚使能内部上拉器件。对于配置为输出的 E 端口管脚，这些位不会产生影响，同时内部拉器件被禁止。 0 E 端口位 - 内部上拉器件禁止。 1 E 端口位 - 内部上拉器件使能。

#### 注意

只有当使用管脚中断功能且配置了相应的边沿选择和管脚选择功能时，才能使用下拉器件。

### 6.5.5.4 E 端口斜率使能寄存器 (PTESE)

	7	6	5	4		3	2	1	0
R W	PTESE7	PTESE6	PTESE5	PTESE4	PTESE3	PTESE2	PTESE1 <sup>1</sup>	PTESE0	
复位：	0	0	0	0	0	0	0	0	0

图 6-35. E 端口寄存器斜率使能 (PTESE)

<sup>1</sup> PTESE1 对输入 PTE1 管脚没有影响。

表 6-33. PTESE 寄存器字段描述

字段	描述
7:0 PTESE[7:0]	<b>E 端口位输出斜率使能</b> — 这些控制位决定是否为相关的 PTE 管脚使能输出斜率控制。对于配置为输入的 E 端口管脚，这些位不会产生任何影响。 0 E 端口位 - 输出斜率控制禁止。 1 E 端口位 - 输出斜率控制使能。

注意：工程样品设计采样和最终成品的斜率复位默认值可能不同。一定要将斜率控制初始化为规定的值，确保正确的操作。

### 6.5.5.5 E 端口驱动强度选择寄存器 (PTEDS)

	7	6	5	4		3	2	1	0
R W	PTEDS7	PTEDS6	PTEDS5	PTEDS4	PTEDS3	PTEDS2	PTEDS1 <sup>1</sup>	PTEDS0	
复位：	0	0	0	0	0	0	0	0	0

图 6-36. E 端口寄存器驱动强度选择 (PTEDS)

<sup>1</sup> PTEDS1 对输入 PTE1 管脚没有影响。

表 6-34. PTEDS 寄存器字段描述

字段	描述
7:0 PTEDS[7:0]	<b>E 端口位的输出驱动强度选择</b> — 这些控制位为相关 PTE 管脚选择低输出驱动和高输出驱动。对于配置为输入的 E 端口管脚，这些位不会产生任何影响。 0 E 端口位 - 选择的低输出驱动强度。 1 E 端口位 - 选择的高输出驱动强度。

### 6.5.6 F 端口寄存器

F 端口由下列寄存器控制。

### 6.5.6.1 F 端口数据寄存器 (PTFD)

	7	6	5	4	3	2	1	0
R W	PTFD7	PTFD6	PTFD5	PTFD4	PTFD3	PTFD2	PTFD1	PTFD0
复位：	0	0	0	0	0	0	0	0

图 6-37. F 端口数据寄存器 (PTFD)

表 6-35. PTFD 寄存器字段描述

字段	描述
7:0 PTFD[7:0]	<b>F 端口数据寄存器位</b> — 对于配置为输入的 F 端口管脚，读数返回管脚上的逻辑电平。对于配置为输出的 F 端口管脚，读数返回写入寄存器的最后一个值。 写入值被锁定在本寄存器的所有位中。对于配置为输出的 F 端口管脚，逻辑电平被输出到驱出相应的 MCU 管脚。 复位强制 PTFD 都为 0，但是这些 0 未被输出到驱出相应的管脚，因为复位还会将所有端口管脚配置为上拉 / 下拉被禁止的高抗阻输入。

### 6.5.6.2 F 端口数据方向寄存器 (PTFDD)

	7	6	5	4	3	2	1	0
R W	PTFDD7	PTFDD6	PTFDD5	PTFDD4	PTFDD3	PTFDD2	PTFDD1	PTFDD0
复位：	0	0	0	0	0	0	0	0

图 6-38. F 端口数据方向寄存器 (PTFDD)

表 6-36. PTFDD 寄存器字段描述

字段	描述
7:0 PTFDD[7:0]	<b>F 端口位的数据方向</b> — 这些读 / 写位控制着 F 端口管脚的方向以及为 PTFD 读数读取的内容。 0 输入（输出驱动被禁止），读数返回管脚值。 1 B 端口位 - 输出驱动使能，PTFD 读数返回 PTFDn 内容。

### 6.5.6.3 F 端口上拉使能寄存器 (PTFPE)

	7	6	5	4		3	2	1	0
R W	PTFPE7	PTFPE6	PTFPE5	PTFPE4		PTFPE3	PTFPE2	PTFPE1	PTFPE0
复位：	0	0	0	0		0	0	0	0

图 6-39. 端口寄存器内部上拉使能 (PTFPE)

表 6-37. PTFPE 寄存器字段描述

字段	描述
7:0 PTFPE[7:0]	<b>F 端口的内部上拉使能位</b> — 这些控制位决定着是否为相关的 PTF 管脚使能内部上拉器件。对于配置为输出的 F 端口管脚，这些位不会产生影响，同时内部拉器件被禁止。 0 F 端口位 - 内部上拉器件被禁止。 1 F 端口位 - 内部上拉器件使能。

#### 注意

只有当使用管脚中断功能且配置了相应的边沿选择和管脚选择功能时，才能使用下拉器件。

### 6.5.6.4 F 端口斜率使能寄存器 (PTFSE)

	7	6	5	4		3	2	1	0
R W	PTFSE7	PTFSE6	PTFSE5	PTFSE4		PTFSE3	PTFSE2	PTFSE1	PTFSE0
复位：	0	0	0	0		0	0	0	0

图 6-40. F 端口寄存器斜率使能 (PTFSE)

表 6-38. PTFSE 寄存器字段描述

字段	描述
7:0 PTFSE[7:0]	<b>F 端口位的输出斜率使能</b> — 这些控制位决定着是否为相关的 PTF 管脚使能输出斜率控制。对于配置为输入的 F 端口管脚，这些位不会产生任何影响。 0 F 端口位 - 输出斜率控制禁止。 1 F 端口位 - 输出斜率控制使能。

**注意：**工程样品和最终成品的斜率复位默认值可能不同。一定要将斜率控制初始化为所需的值，以确保正确的操作。

### 6.5.6.5 F 端口驱动强度选择寄存器 (PTFDS)

	7	6	5	4		3	2	1	0
R	PTFDS7	PTFDS6	PTFDS5	PTFDS4		PTFDS3	PTFDS2	PTFDS1	PTFDS0
W									

复位： 0 0 0 0 0 0 0 0 0 0

图 6-41. F 端口寄存器驱动强度选择 (PTFDS)

表 6-39. PTFDS 寄存器字段描述

字段	描述
7:0 PTFDS[7:0]	<b>F 端口位的输出驱动强度选择</b> — 这些控制位为相关的 PTF 管脚选择低输出驱动和高输出驱动。对于配置为输入的 F 端口管脚，这些位不会产生任何影响。 0 F 端口位 - 选择的低输出驱动强度。 1 F 端口位 - 选择的高输出驱动强度。

### 6.5.7 G 端口寄存器

G 端口由下列寄存器控制。

#### 6.5.7.1 G 端口数据寄存器 (PTGD)

	7	6	5	4		3	2	1	0
R	0	0	PTGD5	PTGD4		PTGD3	PTGD2	PTGD1	PTGD0
W									

复位： 0 0 0 0 0 0 0 0 0 0

[ ] = 未实现或已被预留

图 6-42. G 端口数据寄存器 (PTGD)

表 6-40. PTGD 寄存器字段描述

字段	描述
5:0 PTGD[5:0]	<b>G 端口数据寄存器位</b> — 对于配置为输入的 G 端口管脚，读数返回管脚上的逻辑电平。对于配置为输出的 G 端口管脚，读数返回写入寄存器的最后一个值。 写入值被锁定在本寄存器的所有位中。对于配置为输出的 G 端口管脚，逻辑电平被输出到相应的 MCU 管脚。 复位强制 PTGD 都为 0，但是这些 0 未被输出到相应的管脚，因为复位还会将所有端口管脚配置为上拉 / 下拉被禁止的高抗阻输入。

### 6.5.7.2 G 端口数据方向寄存器 (PTGDD)

	7	6	5	4		3	2	1	0
R	0	0	PTGDD5	PTGDD4	PTGDD3	PTGDD2	PTGDD1	PTGDD0	
W									
复位：	0	0	0	0	0	0	0	0	0
	= 未实现或已被预留								

图 6-43. G 端口数据方向寄存器 (PTGDD)

表 6-41. PTGDD 寄存器字段描述

字段	描述
5:0 PTGDD[5:0]	<b>G 端口位的数据方向</b> — 这些读 / 写位控制着 G 端口管脚的方向以及为 PTGD 读数读取的内容。 0 输入（输出驱动被禁止），读数返回管脚值。 1 G 端口位 - 输出驱动使能，PTGD 读数返回 PTGDn 内容。

### 6.5.7.3 G 端口上拉使能寄存器 (PTGPE)

	7	6	5	4		3	2	1	0
R	0	0	PTGPE5	PTGPE4	PTGPE3	PTGPE2	PTGPE1	PTGPE0	
W									
复位：	0	0	0	0	0	0	0	0	0
	= 未实现或已被预留								

图 6-44. G 端口寄存器内部上拉使能 (PTGPE)

表 6-42. PTGPE 寄存器字段描述

字段	描述
5:0 PTGPE[5:0]	<b>G 端口的内部上拉使能位</b> — 这些控制位决定着是否为相关的 PTG 管脚使能内部上拉器件。对于配置为输出的 G 端口管脚，这些位不会产生影响，同时内部拉器件被禁止。 0 G 端口位 - 内部上拉器件被禁止。 1 G 端口位 - 内部上拉器件使能。

#### 注意

只有当使用管脚中断功能且配置了相应的边沿选择和管脚选择功能时，才能使用下拉器件。

### 6.5.7.4 G 端口斜率使能寄存器 (PTGSE)

	7	6	5	4		3	2	1	0
R	0	0	PTGSE5	PTGSE4	PTGSE3	PTGSE2	PTGSE1	PTGSE0	
W									
复位：	0	0	0	0	0	0	0	0	0
	= 未实现或已被预留								

图 6-45. G 端口寄存器斜率使能 (PTGSE)

表 6-43. PTGSE 寄存器字段描述

字段	描述
5:0 PTGSE[5:0]	<b>G 端口位的输出斜率使能</b> — 这些控制位决定是否为相关的 PTG 管脚使能输出斜率控制。对于配置为输入的 G 端口管脚，这些位不会产生任何影响。 0 G 端口位 - 输出斜率控制禁止。 1 G 端口位 - 输出斜率控制使能。

注意：工程样品和最终成品的斜率复位默认值可能不同。一定要将斜率控制初始化为所需的值，以确保正确的操作。

### 6.5.7.5 G 端口驱动强度选择寄存器 (PTGDS)

	7	6	5	4		3	2	1	0
R	0	0	PTGDS5	PTGDS4	PTGDS3	PTGDS2	PTGDS1	PTGDS0	
W									
复位：	0	0	0	0	0	0	0	0	0
	= 未实现或已被预留								

图 6-46. G 端口寄存器的驱动强度选择 (PTGDS)

表 6-44. PTGDS 寄存器字段描述

字段	描述
5:0 PTGDS[5:0]	<b>G 端口位的输出驱动强度选择</b> — 这些控制位为相关的 PTG 选择低输出驱动和高输出驱动。对于配置为输入的 G 端口管脚，这些位不会产生任何影响。 0 G 端口位 - 选择的低输出驱动强度。 1 G 端口位 - 选择的高输出驱动强度。

# 第 7 章

## 中央处理器 (S08CPUV3)

### 7.1 介绍

本节简要地介绍了 HCS08 系列的寄存器、寻址模式和 CPU 指令集。如需了解更多信息，请参见 HCS08 系列参考手册第 1 卷，飞思卡尔半导体文档订单号 HCS08RMV1/D。

HCS08 CPU 和 M68HC08 CPU 的源和目标代码完全兼容。在 HCS08 的 CPU 中增加了几个指令和增强型寻址模式来提高 C 编译器效率、支持取代了早期 M68HC08 微控制器 (MCU) 监控模式的新背景调试系统。

#### 7.1.1 特性

HCS08 CPU 的特性包括：

- 目标代码完全向上兼容 M68HC05 和 M68HC08 系列
- 所有寄存器和存储器都被映射到一个 64Kb 地址空间
- 16 位堆栈指针 (64Kb 地址空间内任意规模的堆栈)
- 16 位索引寄存器 (H:X)，具有强大的索引寻址模式
- 8 位累加器 (A)
- 很多指令将 X 当作备用的通用 8 位寄存器
- 7 个寻址模式：
  - Inherent — 内部寄存器里的操作数
  - Relative — 分支目的地的 8 位带符号偏移
  - Immediate — 下一个目标代码字节里的操作数
  - Direct — 0x0000 - 0x00FF 存储器的操作数
  - Extended — 64-Kb 地址空间里的操作数
  - Indexed relative to H:X — 5 个子模式，包括自动累加
  - Indexed relative to SP — 大幅提高 C 效率
- 存储器至存储器数据移动指令，具有 4 个地址模式组合
- 溢出、半进位、负数、零和进位条件代码支持在带符号、不带符号和十进制计数法 (BCD) 运算结果上的有条件转移
- 有效的位操控指令
- 快速的 8-bit by 8-bit 乘法和 16-bit by 8-bit 除法指令
- 调用低功率运行模式的 STOP 和 WAIT 指令

## 7.2 程序员模型和 CPU 寄存器

图 7-1 显示了 5 个 CPU 寄存器。CPU 寄存器不属于存储器映射。

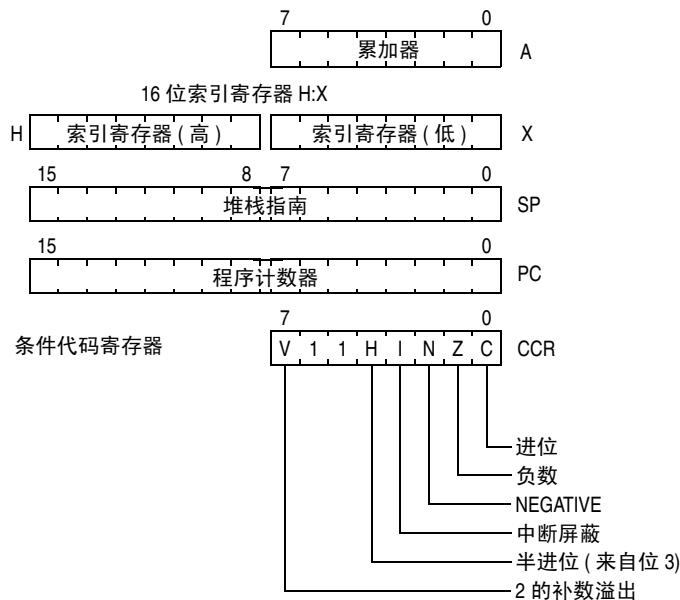


图 7-1. CPU 寄存器

### 7.2.1 累加器 (A)

A 累加器是通用 8 位寄存器。算术逻辑单元 (ALU) 中的其中一个输入操作数会来自于累加器，在完成算术和逻辑运算后，ALU 结果通常保存在 A 累加器中。使用不同寻址模式可以将存储器中指定地址的数据加载到累加器；或者使用不同寻址模式将 A 的内容保存到指定的存储器地址。复位不会对 A 累加器的内容产生影响。

### 7.2.2 索引寄存器 (H:X)

这个 16 位寄存器实际上是两个独立的 8 位寄存器 (H 和 X)，它们通常以 16 位地址指针的形式在一起工作，其中，H 保存地址的高字节，X 保存地址的低字节。所有索引寻址模式指令均使用 H:X 中的整个 16 位值，作为索引参考指针。然而，为了实现与早期 M68HC05 系列的兼容，有些指令只在低阶 8 位部分 (X) 上操作。

许多指令将 X 作为备用通用 8 位寄存器，该寄存器可以用来保留 8 位数据值。X 可以被清除、累加、减少、补数、忽略、移位或旋转。传输指令允许数据从 A 那里进行传输，也可以传输到 A，然后在这里进行算术和逻辑运算。

为了实现与早期 M68HC05 系列的兼容，复位期间 H 被强制为 0x00。复位不会对 X 的内容产生影响。

### 7.2.3 堆栈指针 (SP)

这个 16 位地址指针寄存器指向自动后进先出 (LIFO) 堆栈上的下一个可用位置。该堆栈可以位于 64Kb 地址空间的任意位置，64Kb 地址空间具有 RAM，大小可以是可用 RAM 量的任意值。该堆栈用于自动保存子程序调用的返回地址，以及在中断期间供本地变量使用的返回地址和 CPU 寄存器。AIS (立即值加到堆栈指针) 指令向 SP 添加一个 8 位带符号的立即值。这通常供用于堆栈上的本地变量的空间分配或取消分配。

SP 在复位时被强制放在 0x00FF 上，以实现与早期 M68HC05 系列的兼容性。在复位初始化期间，HCS08 程序通常将 SP 中的值更改为片上 RAM 最后位置（最高地址）的地址，以释放直接页面 RAM（从片上寄存器末端到 0x00FF）。

为了实现与 M68HC05 系列的兼容，指令中还包括 RSP（复位堆栈指针）指令，但很少在 HCS08 程序中使用，因为它只影响堆栈指针的低阶部分。

### 7.2.4 程序计数器 (PC)

程序计数器是一个 16 位寄存器，它包含将要获取的下一个指令或操作数的地址。

在正常的程序执行过程中，每次获取指令或操作数时，程序计数器就会自动累加到下一个顺序存储器位置。跳转、分支、中断和返回操作加载地址到程序计数器，而非下一个顺序位置的存储器地址，这被称为 **change-of-flow**（流程转换）。

复位时，程序计数器被加载位于 0xFFFF 和 0xFFFF 的复位向量。这里保存的向量是退出复位状态后将要执行的第一个指令的地址。

### 7.2.5 条件码寄存器 (CCR)

8 位条件码寄存器包括中断屏蔽 (I) 和 5 个显示刚执行指令的结果的标记。位 6 和 5 永远设为 1。下面的几个段落描述了一般条件下的条件码位功能。如需了解各个指令如何设置 CCR 位的更多信息，请参见“*HCS08 系列参考手册第 1 卷*”，飞思卡尔半导体文档订购编号 HCS08RMv1。

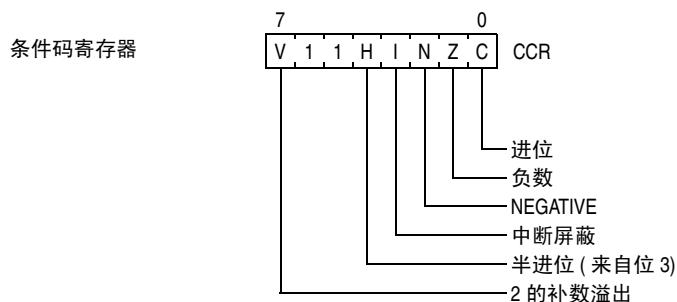


图 7-2. 条件码寄存器

表 7-1. CCR 寄存器字段描述

字段	描述
7 V	<b>2 的补数溢出标记</b> — 当出现 2 的补数溢出时, CPU 设置溢出标记。带符号的分支指令 BGT、BGE、BLE 和 BLT 使用溢出标记。 0 无溢出 1 溢出
4 H	<b>半进位标记</b> — 在 add-without-carry (ADD) 或 add-with-carry (ADC) 运算中, 当累加器位 3 和 4 间发生进位时 CPU 设置半进位标记。十进制计数码 (BCD) 算术运算中要求半进位标记。DAA 指令使用 H 和 C 条件码位状态, 将更正数值从原来 BCD 操作数上的 ADD 或 ADC 自动添加到结果中, 以便将结果更正为有效的 BCD 值。 0 位 3 和 4 间无进位 1 位 3 和 4 间有进位
3 I	<b>中断屏蔽位</b> — 当设置了中断屏蔽时, 禁止所有可屏蔽 CPU 中断。清除中断屏蔽后, CPU 中断使能。发生 CPU 中断时, 中断屏蔽会在 CPU 寄存器保存在堆栈上后但执行中断服务程序第一个指令执行前被自动设置。 在执行了任何清除 I (CLI 或 TAP) 的指令后, 中断都不能在指令边界识别。这样可以保证当设置了 I 时, CLI 或 TAP 后的下一个指令总被执行, 而不会出现介于其间的中断。 0 中断使能 1 中断禁止
2 N	<b>负数标志</b> — 算术运算、逻辑运算或数据处理产生一个负数结果且设置该结果的位 7 时, CPU 设置负数标记。如果已加载或已保存值的最高位是 1, 加载或保存 8 位或 16 位值都会导致设置 N。 0 非负数结果 1 负数结果
1 Z	<b>零标志</b> — 当算术运算、逻辑运算和数据处理产生 0x00 或 0x0000 结果时, CPU 设置零标记。如果已加载或已保存值的全为 0, 加载或保存 8 位或 16 位值都会导致设置 Z。 0 非 0 结果 1 0 结果
0 C	<b>进位 / 借位标记</b> — 当加法运算在累加器的位 7 外产生一个进位, 或者减法运算需要一个借位时, CPU 设置进位 / 借位标记。有些指令, 如位测试和分支、移位和旋转, 也清除或设置进位 / 借位标记。 0 位 7 无进位 1 位 7 有进位

## 7.3 寻址模式

寻址模式定义 CPU 访问操作数和数据的方式。在 HCS08 中，所有存储器、状态和控制寄存器、输入 / 输出 (I/O) 端口共用一个 64Kb 线性地址空间，因此 16 位二进位地址能够识别任意存储器位置。这种安排意味着，访问 RAM 中变量的相同指令还可以用来访问 I/O 和控制寄存器或非易失性程序空间。

有些指令使用一个以上的寻址模式。例如，移动指令用一种寻址模式来指定源操作数，用另外一种寻址模式来指定目的地地址。而诸如 BRCLR、BRSET、CBEQ 和 DBNZ 这样的指令则使用一种寻址模式来指定测试用操作数的位置，当测试条件成立时，又使用相关寻址模式来指定分支目的地地址。对于 BRCLR、BRSET、CBEQ 和 DBNZ 而言，指令集表里列出的寻址模式是访问将要测试的操作数所需的寻址模式，而相关寻址模式则意味着分支地址寻址模式。

### 7.3.1 固有寻址模式 (INH)

在这种寻址模式中，完成指令（如果有的话）所需的操作数位于 CPU 寄存器上，因此 CPU 不需要访问存储器以获得任何操作数。

### 7.3.2 关联寻址模式 (REL)

关联寻址模式用来指定分支指令的目的地址。一个带符号的 8 位偏移值位于紧接操作码的存储器位置。在执行期间，如果分支条件成立，带符号的偏移被符号扩展为 16 位值，并被添加到程序计数器的当前内容中，造成在分支目的地地址上继续执行程序。

### 7.3.3 立即寻址模式 (IMM)

在立即寻址模式中，完成指令所需的操作数包含在紧跟存储器的指令操作码的目标码中。如果是 16 位直接操作数，高阶字节位于操作码后的下一个存储器位置，低阶字节位于高阶存储器后的下一个存储器位置。

### 7.3.4 直接寻址模式 (DIR)

在直接寻址模式中，指令包括直接页面 (0x0000 – 0x00FF) 地址的低阶 8 位。在执行过程中，16 位地址的高 8 位部分隐含为 0x00，低 8 位地址从指令中直接获得，以此构成 16 位地址指向指令所需的操作数。这比为操作数指定整个 16 位地址更快、存储器也更有效。

### 7.3.5 扩展寻址模式 (EXT)

在扩展寻址模式中，操作数的整个 16 位地址都位于紧接操作码后的程序存储器的两个字节上（高阶字节优先）。

### 7.3.6 索引寻址模式

索引寻址模式带有 7 个变种，5 个使用 16 位 H:X 索引寄存器对和两个使用堆栈指针作为基本参考。

### 7.3.6.1 有索引、无偏移 (IX)

这个索引寻址变种将 H:X 索引寄存器对的 16 位地址作为完成指令所需的操作数地址。

### 7.3.6.2 有索引、无带后增量的偏移 (IX+)

这个索引寻址变种将 H:X 索引寄存器对的 16 位值作为完成指令所需的操作数地址。在获得操作数后，索引寄存器对然后被增加 ( $H:X = H:X + 0x0001$ )。这种寻址模式只用于 MOV 和 CBEQ 指令。

### 7.3.6.3 有索引、8 位偏移 (IX1)

这个索引寻址变种将 H:X 索引寄存器对和指令中不带符号的 8 位偏移作为完成指令所需的操作数地址。

### 7.3.6.4 有索引、带后增量的 8 位偏移 (IX1+)

这个索引寻址变种将 H:X 索引寄存器对和指令中不带符号的 8 位偏移作为完成指令所需的操作数地址。在获得操作数后，索引寄存器对然后被增加 ( $H:X = H:X + 0x0001$ )。这种寻址模式只用于 CBEQ 指令。

### 7.3.6.5 有索引、16 位偏移 (IX2)

这个索引寻址变种将 H:X 索引寄存器对和指令中的 16 位偏移作为完成指令所需的操作数地址。

### 7.3.6.6 SP 相关、8 位偏移 (SP1)

这个索引寻址变种将堆栈指针 (SP) 和指令中不带符号的 8 位偏移作为完成指令所需的操作数地址。

### 7.3.6.7 SP 相关、16 位偏移 (SP2)

这个索引寻址变种将堆栈指针 (SP) 和指令中的 16 位偏移作为完成指令所需的操作数地址。

## 7.4 特殊运算

CPU 执行一些特殊运算，这些运算和指令类似，但不像其他 CPU 指令那样有操作码。此外，有些指令，如 STOP 和 WAIT，还会直接影响其他 MCU 电路。本节提供了有关这些运算报文。

### 7.4.1 复位顺序

上电复位 (POR) 事件、内部条件 (如 COP 看门狗) 或外部低效复位管脚有效状态等都可以导致复位。发生复位事件时，CPU 立即停止正在处理的任何操作 (MCU 在响应复位事件前，无需等待指令边界)。如需了解 MCU 如何识别复位和决定源的详细信息，请参见“复位，中断和系统配置”章。

当确定复位是否来自内部源的顺序已经确定且复位管脚的电平不再处于复位有效状态时，复位事件就视为已经结束。复位事件结束后，CPU 执行一个 6 周期顺序，从 0xFFFFE 和 0xFFFF 中获取复位向量，并填写指令队列，为执行第一个程序指令做准备。

### 7.4.2 中断时序

发生中断请求时，CPU 在响应中断前会完成当前指令。此时，程序计数器指向下一个指令的开始位置，这个位置也是 CPU 完成中断操作后的返回位置。CPU 通过执行与软件中断 (SWI) 指令一样的运算顺序响应中断，但不同的是，用于向量获取的地址由中断时序开始时处于等待状态优先级最高的中断决定。

CPU 中断的时序为：

1. 把 PCL、PCH、X、A 和 CCR 的内容顺序保存到堆栈上；
2. 在 CCR 中设置 I 位；
3. 获取中断向量高阶部分；
4. 获取中断向量低阶部分；
5. 延迟一个空闲总线周期；
6. 获取从中断向量显示的地址开始的 3 个字节程序报文来填写指令队列，为执行中断服务程序的第一个指令做好准备。

当 CCR 内容被推送到堆栈后，在 CCR 中 I 位被置位，防止中断服务程序中出现其他中断。尽管可以用中断服务程序的指令来清除 I 位，但这样会发生中断嵌套（不推荐使用该方法，因为它会让程序难以调试和维护）。

为了实现与早期 M68HC05 的兼容，H:X 索引寄存器对 (H) 的高阶部分未作为中断顺序的一部分保存在堆栈上。用户必须在服务程序开始时使用 PSHH 指令来保存 H，然后在结束中断服务程序的 RTI 前使用 PULH 指令。如果您确定中断服务程序不使用任何指令或可能修改 H 值的自动增量寻址模式，就没有必要保存 H。

软件中断 (SWI) 指令类似硬件中断，只是它不是由 CCR 中的全局 I 位进行屏蔽，它与程序中的指令操作码有关，因此它不同步于程序执行。

### 7.4.3 等待模式操作

**WAIT** 指令通过清除 CCR 中的 I 位来使能中断。然后它暂停 CPU 时钟，以减少整体功耗，CPU 此时正在等待将把 CPU 从等待模式唤醒的中断或复位事件。当发生中断或复位事件时，CPU 时钟会重新开始工作，中断或复位事件被正常处理。

如果串行 **BACKGROUND** 命令通过背景调试接口发送到 MCU，与此同时 CPU 处于等待模式，那么 CPU 时钟会重新开始工作，CPU 进入活动后台模式，可以处理其他串行后台命令。这样就确保了即使主机开发系统处于等待模式，它仍能访问目标 MCU。

### 7.4.4 停止模式操作

在通常情况下，包括晶体振荡器（使用时）在内的所有系统时钟在停止模式中都会暂停，以最大限度地减少功耗。在这类系统中，需要外部电路来控制停止模式所花费的时间，并且在需要重新开始处理时发出一个信号来唤醒目标 MCU。与早期 M68HC05 和 M68HC08 MCU 不同的是，HCS08 可以在经过配置后使停止模式以最少的时钟运行。这同样允许内部周期信号将 MCU 从停止模式唤醒。

如果主机调试系统与背景调试管脚（**BKGD**）连接，且串行命令通过后台接口设置了 **ENBDM** 控制位（或者因为 MCU 被重置为活动后台模式），那么当 MCU 进入停止模式时，振荡器就被迫保持活动状态。这时，当通过背景调试接口将串行 **BACKGROUND** 命令发送到 MCU，而与此同时 CPU 处于停止模式时，CPU 时钟会重新开始工作，CPU 进入可以处理其他串行后台命令的活动后台模式。这样就确保了即使 MCU 处于等待模式主机开发系统仍能访问目标 MCU。

停止模式恢复取决于特殊 HCS08 及振荡器是否在停止模式中停止。更多信息请参见“运行模式”。

### 7.4.5 BGND 指令

相对于 M68HC08，**BGND** 指令是 HCS08 的新增指令。普通用户程序中不会使用 **BGND**，因为它强制 CPU 停止处理用户指令而进入活动后台模式。重新执行用户的唯一方法是通过复位，或由主机调试系统通过背景调试接口发出 **GO**、**TRACE1** 或 **AGGO** 串行命令。

基于软件的断点可以通过用 **BGND** 操作码在所需断点地址上取代操作码的方式进行设置。当程序到达该断点地址时，CPU 会被迫进入活动后台模式，而不是继续用户程序。

## 7.5 HCS08 指令集小结

表 7-2 概括地介绍了所有可能的寻址模式中的 HCS08 指令集。表中显示了各个指令的每个寻址模式变种的操作数构造、内部总线时钟周期的执行时间和逐周期详情。

表 7-2. 指令集小结 (第 1 页, 共 9 页)

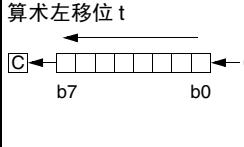
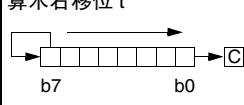
Source Form	Operation	Address Mode	Object Code	Cycles	Cyc-by-Cyc Details	Affect on CCR	
						V	1
ADC #opr8i ADC opr8a ADC opr16a ADC oprx16,X ADC oprx8,X ADC ,X ADC oprx16,SP ADC oprx8,SP	进位添加 A" (A) + (M) + (C)	IMM DIR EXT IX2 IX1 IX SP2 SP1	A9 ii B9 dd C9 hh ll D9 ee ff E9 ff F9 9E D9 ee ff 9E E9 ff	2 3 4 4 3 3 5 4	pp rpp prpp prpp rpp rfp pprpp prpp	↑ 1 1 ↓	- ↑ ↓ ↑
ADD #opr8i ADD opr8a ADD opr16a ADD oprx16,X ADD oprx8,X ADD ,X ADD oprx16,SP ADD oprx8,SP	无进位添加 A" (A) + (M)	IMM DIR EXT IX2 IX1 IX SP2 SP1	AB ii BB dd CB hh ll DB ee ff EB ff FB 9E DB ee ff 9E EB ff	2 3 4 4 3 3 5 4	pp rpp prpp prpp rpp rfp pprpp prpp	↑ 1 1 ↓	- ↑ ↓ ↑
AIS #opr8i	在堆栈指针上添加立即值 (带符号) SP" (SP) + (M)	IMM	A7 ii	2	pp	- 1 1 -	-----
AIX #opr8i	在索引寄存器 (H:X) 上添加立即值 (带符号) )H:X H:X" (H:X) + (M)	IMM	AF ii	2	pp	- 1 1 -	-----
AND #opr8i AND opr8a AND opr16a AND oprx16,X AND oprx8,X AND ,X AND oprx16,SP AND oprx8,SP	逻辑 AND A" (A) & (M)	IMM DIR EXT IX2 IX1 IX SP2 SP1	A4 ii B4 dd C4 hh ll D4 ee ff E4 ff F4 9E D4 ee ff 9E E4 ff	2 3 4 4 3 3 5 4	pp rpp prpp prpp rpp rfp pprpp prpp	0 1 1 -	↑ ↓ ↑ -
ASL opr8a ASLA ASLX ASL oprx8,X ASL ,X ASL oprx8,SP	算术左移位 t  (同 LSL)	DIR INH INH IX1 IX SP1	38 dd 48 58 68 ff 78 9E 68 ff	5 1 1 5 4 6	rfwpp p p rfwpp rfwp prfwpp	↑ 1 1 -	- ↑ ↓ ↑
ASR opr8a ASRA ASRX ASR oprx8,X ASR ,X ASR oprx8,SP	算术右移位 t 	DIR INH INH IX1 IX SP1	37 dd 47 57 67 ff 77 9E 67 ff	5 1 1 5 4 6	rfwpp p p rfwpp rfwp prfwpp	↑ 1 1 -	- ↑ ↓ ↑
BCC rel	如果进位位清除, 分支 (如果 C = 0)	REL	24 rr	3	ppp	- 1 1 -	-----

表 7-2. 指令集小结 (第 2 页, 共 9 页)

Source Form	Operation	Address Mode	Object Code	Cycles	Cyc-by-Cyc Details	Affect on CCR	
						V	I
BCLR <i>n,opr8a</i>	存储器里的清除位 <i>n</i> (Mn = 0)	DIR (b0) DIR (b1) DIR (b2) DIR (b3) DIR (b4) DIR (b5) DIR (b6) DIR (b7)	11 dd 13 dd 15 dd 17 dd 19 dd 1B dd 1D dd 1F dd	5 5 5 5 5 5 5 5	rfwpp rfwpp rfwpp rfwpp rfwpp rfwpp rfwpp rfwpp	- 1 1 -	----
BCS <i>rel</i>	如果进制位设置, 分支 (如果 C = 1) (同 BLO)	REL	25 rr	3	ppp	- 1 1 -	----
BEQ <i>rel</i>	如果相等, 分支 (如果 Z = 1)	REL	27 rr	3	ppp	- 1 1 -	----
BGE <i>rel</i>	如果大于或等于, 分支 (如果 N > V = 0) (带符号)	REL	90 rr	3	ppp	- 1 1 -	----
BGND	如果 ENBDM=1, 进入活动后台, 等待处理 BDM 命令, 直到 GO, TRACE1 或 TAGGO	INH	82	5+	fpp...ppp	- 1 1 -	----
BGT <i>rel</i>	如果大于, 分支 (如果 Z > (N > V) = 0) (带符号)	REL	92 rr	3	ppp	- 1 1 -	----
BHCC <i>rel</i>	如果半进制位清除, 分支 (如果 H = 0)	REL	28 rr	3	ppp	- 1 1 -	----
BHCS <i>rel</i>	如果半进制位设置, 分支 (如果 H = 1)	REL	29 rr	3	ppp	- 1 1 -	----
BHI <i>rel</i>	如果高于, 分支 (如果 C > Z = 0)	REL	22 rr	3	ppp	- 1 1 -	----
BHS <i>rel</i>	如果高于或相同, 分支 (如果 C = 0) (同 BCC)	REL	24 rr	3	ppp	- 1 1 -	----
BIH <i>rel</i>	如果 IRQ 管脚高, 分支 (如果 IRQ 管脚 = 1)	REL	2F rr	3	ppp	- 1 1 -	----
BIL <i>rel</i>	如果 IRQ 管脚低, 分支 (如果 IRQ 管脚 = 0)	REL	2E rr	3	ppp	- 1 1 -	----
BIT #opr8i BIT opr8a BIT opr16a BIT oprx16,X BIT oprx8,X BIT ,X BIT oprx16,SP BIT oprx8,SP	位测试 (A) & (M) (CCR 已更新, 但操作数没变)	IMM DIR EXT IX2 IX1 IX SP2 SP1	A5 ii B5 dd C5 hh ll D5 ee ff E5 ff F5 9E D5 ee ff 9E E5 ff	2 3 4 4 3 3 5 4	pp rpp prpp prpp rpp rpp pprpp prpp	0 1 1 -	↑ ↓ -
BLE <i>rel</i>	如果小于或等于, 分支 (如果 Z < (N > V) = 1) (带符号)	REL	93 rr	3	ppp	- 1 1 -	----
BLO <i>rel</i>	如果小于, 分支 (如果 C = 1) (同 BCS)	REL	25 rr	3	ppp	- 1 1 -	----
BLS <i>rel</i>	如果小于或等于, 分支 (如果 C < Z = 1)	REL	23 rr	3	ppp	- 1 1 -	----
BLT <i>rel</i>	如果小于, 分支 (如果 Z < (N > V) = 1) (带符号)	REL	91 rr	3	ppp	- 1 1 -	----
BMC <i>rel</i>	如果中断屏蔽清除, 分支 (如果 I = 0)	REL	2C rr	3	ppp	- 1 1 -	----
BMI <i>rel</i>	如果减, 分支 (如果 N = 1)	REL	2B rr	3	ppp	- 1 1 -	----
BMS <i>rel</i>	如果中断屏蔽设置, 分支 (如果 I = 1)	REL	2D rr	3	ppp	- 1 1 -	----
BNE <i>rel</i>	如果不等于, 分支 (如果 Z = 0)	REL	26 rr	3	ppp	- 1 1 -	----
BPL <i>rel</i>	如果加, 分支 (如果 N = 0)	REL	2A rr	3	ppp	- 1 1 -	----

表 7-2. 指令集小结 (第 3 页, 共 9 页)

Source Form	Operation	Address Mode	Object Code	Cycles	Cyc-by-Cyc Details	Affect on CCR	
						V	I
BRA rel	总是分支 (如果 I = 1)	REL	20 rr	3	ppp	- 1 1 -	---
BRCLR n,opr8a,rel	如果存储器的位 n 清除, 分支 (如果 (Mn) = 0)	DIR (b0) DIR (b1) DIR (b2) DIR (b3) DIR (b4) DIR (b5) DIR (b6) DIR (b7)	01 dd rr 03 dd rr 05 dd rr 07 dd rr 09 dd rr 0B dd rr 0D dd rr 0F dd rr	5 5 5 5 5 5 5 5	rpppp rpppp rpppp rpppp rpppp rpppp rpppp rpppp	- 1 1 -	---
BRN rel	从不分支 (如果 I = 0)	REL	21 rr	3	ppp	- 1 1 -	---
BRSET n,opr8a,rel	如果存储器的位 n 设置, 分支 (如果 (Mn) = 1)	DIR (b0) DIR (b1) DIR (b2) DIR (b3) DIR (b4) DIR (b5) DIR (b6) DIR (b7)	00 dd rr 02 dd rr 04 dd rr 06 dd rr 08 dd rr 0A dd rr 0C dd rr 0E dd rr	5 5 5 5 5 5 5 5	rpppp rpppp rpppp rpppp rpppp rpppp rpppp rpppp	- 1 1 -	---
BSET n,opr8a	在存储器里设置位 n(Mn ^ 1)	DIR (b0) DIR (b1) DIR (b2) DIR (b3) DIR (b4) DIR (b5) DIR (b6) DIR (b7)	10 dd 12 dd 14 dd 16 dd 18 dd 1A dd 1C dd 1E dd	5 5 5 5 5 5 5 5	rfwpp rfwpp rfwpp rfwpp rfwpp rfwpp rfwpp rfwpp	- 1 1 -	---
BSR rel	分支到子程序 PC " (PC) + \$0002 推 (PCL); SP " (SP) - \$0001 推 (PCH); SP " (SP) - \$0001 PC " (PC) + rel/	REL	AD rr	5	ssppp	- 1 1 -	---
CBEQ opr8a,rel CBEQA #opr8i,rel CBEQX #opr8i,rel CBEQ oprx8,X+,rel CBEQ ,X+,rel CBEQ oprx8,SP,rel	比较 ... 分支, 如果 (A) = (M) 分支, 如果 (A) = (M) 分支, 如果 (X) = (M) 分支, 如果 (A) = (M) 分支, 如果 (A) = (M) 分支, 如果 (A) = (M)	DIR IMM IMM IX1+ IX+ SP1	31 dd rr 41 ii rr 51 ii rr 61 ff rr 71 rr 9E 61 ff rr	5 4 4 5 5 6	rpppp pppp pppp rpppp rfppp prpppp	- 1 1 -	---
CLC	清除进位 (C ^ 0)	INH	98	1	p	- 1 1 -	--- 0
CLI	清除中断屏蔽位 (I ^ 0)	INH	9A	1	p	- 1 1 -	0 ---
CLR opr8a CLRA CLRX CLRH CLR oprx8,X CLR ,X CLR oprx8,SP	清除 M " \$00 A " \$00 X " \$00 H " \$00 M " \$00 M " \$00 M " \$00	DIR INH INH INH IX1 IX SP1	3F dd 4F 5F 8C 6F ff 7F 9E 6F ff	5 1 1 1 5 4 6	rfwpp p p p rfwpp rfwp prfwpp	0 1 1 -	- 0 1 -

表 7-2. 指令集小结 (第 4 页, 共 9 页)

Source Form	Operation	Address Mode	Object Code	Cycles	Cyc-by-Cyc Details	Affect on CCR		
						V	1	1
CMP #opr8i CMP opr8a CMP opr16a CMP oprx16,X CMP oprx8,X CMP ,X CMP oprx16,SP CMP oprx8,SP	比较存储器和累加器 A - M (CCR 已更新, 但操作数未变)	IMM DIR EXT IX2 IX1 IX SP2 SP1	A1 ii B1 dd C1 hh ll D1 ee ff E1 ff F1 9E D1 ee ff 9E E1 ff	2 3 4 4 3 3 5 4	pp rpp prpp prpp rpp rfp pprpp prpp	↑ 1 1 -	-	↑↑↑
COM opr8a COMA COMX COM oprx8,X COM ,X COM oprx8,SP	补数 (1 的补数) M ~ (M) = \$FF - (M) A ~ (A) = \$FF - (A) X ~ (X) = \$FF - (X) M ~ (M) = \$FF - (M) M ~ (M) = \$FF - (M) M ~ (M) = \$FF - (M)	DIR INH INH IX1 IX SP1	33 dd 43 53 63 ff 73 9E 63 ff	5 1 1 5 4 6	rfwpp p p rfwpp rfwp prfwp	0 1 1 -	-	↑↑↑
CPHX opr16a CPHX #opr16i CPHX opr8a CPHX oprx8,SP	比较索引寄存器 (H:X) 和存储器 (H:X) - (M:M + \$0001) (CCR 已更新, 但操作数未变)	EXT IMM DIR SP1	3E hh ll 65 jj kk 75 dd 9E F3 ff	6 3 5 6	prrfpp ppp rrfpp prrfpp	↑ 1 1 -	-	↑↑↑
CPX #opr8i CPX opr8a CPX opr16a CPX oprx16,X CPX oprx8,X CPX ,X CPX oprx16,SP CPX oprx8,SP	比较 X (索引储存器低) 和存储器 X - M (CCR 已更新, 但操作数未变)	IMM DIR EXT IX2 IX1 IX SP2 SP1	A3 ii B3 dd C3 hh ll D3 ee ff E3 ff F3 9E D3 ee ff 9E E3 ff	2 3 4 4 3 3 5 4	pp rpp prpp prpp rpp rfp pprpp prpp	↑ 1 1 -	-	↑↑↑
DAA	十进调整累加器 BCD 值的 ADD 或 ADC 后	INH	72	1	p	U 1 1 -	-	↑↑↑
DBNZ opr8a,rel DBNZA rel DBNZX rel DBNZ oprx8,X,rel DBNZ ,X,rel DBNZ oprx8,SP,rel	减量 A, X, 或 M, 如果非 0, 分支 (如果 (result) ¼ 0) DBNZX 影响 X 而非 H	DIR INH INH IX1 IX SP1	3B dd rr 4B rr 5B rr 6B ff rr 7B rr 9E 6B ff rr	7 4 4 7 6 8	rfwpppp fppp fppp rfwpppp rfwppp prfwpppp	- 1 1 -	-	- - - -
DEC opr8a DECA DECX DEC oprx8,X DEC ,X DEC oprx8,SP	减量 M ~ (M) - \$01 A ~ (A) - \$01 X ~ (X) - \$01 M ~ (M) - \$01 M ~ (M) - \$01 M ~ (M) - \$01	DIR INH INH IX1 IX SP1	3A dd 4A 5A 6A ff 7A 9E 6A ff	5 1 1 5 4 6	rfwpp p p rfwpp rfwp prfwpp	↑ 1 1 -	-	↑ -
DIV	除 A ~ (H:A)^(X); H ~ 余数	INH	52	6	fffffp	- 1 1 -	-	- ↑↓
EOR #opr8i EOR opr8a EOR opr16a EOR oprx16,X EOR oprx8,X EOR ,X EOR oprx16,SP EOR oprx8,SP	存储器和累加器 " 异或 " A ~ (A Y M)	IMM DIR EXT IX2 IX1 IX SP2 SP1	A8 ii B8 dd C8 hh ll D8 ee ff E8 ff F8 9E D8 ee ff 9E E8 ff	2 3 4 4 3 3 5 4	pp rpp prpp prpp rpp rfp pprpp prpp	0 1 1 -	-	↑ -

表 7-2. 指令集小结 (第 5 页, 共 9 页)

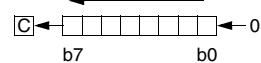
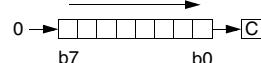
Source Form	Operation	Address Mode	Object Code	Cycles	Cyc-by-Cyc Details	Affect on CCR	
						V	I
INC opr8a INCA INCX INC oprx8,X INC ,X INC oprx8,SP	增量 M "(M) + \$01 A "(A) + \$01 X "(X) + \$01 M "(M) + \$01 M "(M) + \$01 M "(M) + \$01	DIR INH INH IX1 IX SP1	3C dd 4C 5C 6C ff 7C 9E 6C ff	5 1 1 5 4 6	rfwpp p p rfwpp rfwp prfwpp	↑ 1 1 -	- ↑ ↓ -
JMP opr8a JMP opr16a JMP oprx16,X JMP oprx8,X JMP ,X	跳转 PC " 跳转地址	DIR EXT IX2 IX1 IX	BC dd CC hh ll DC ee ff EC ff FC	3 4 4 3 3	ppp pppp pppp ppp ppp	- 1 1 -	- - - -
JSR opr8a JSR opr16a JSR oprx16,X JSR oprx8,X JSR ,X	跳转到子程序 PC "(PC) + n (n = 1, 2, 或 3) 推 (PCL); SP "(SP) - \$0001 推 (PCH); SP "(SP) - \$0001 PC " 无条件地址	DIR EXT IX2 IX1 IX	BD dd CD hh ll DD ee ff ED ff FD	5 6 6 5 5	ssppp pssppp pssppp ssppp ssppp	- 1 1 -	- - - -
LDA #opr8i LDA opr8a LDA opr16a LDA oprx16,X LDA oprx8,X LDA ,X LDA oprx16,SP LDA oprx8,SP	从存储器那里加载累加器 A "(M)	IMM DIR EXT IX2 IX1 IX SP2 SP1	A6 ii B6 dd C6 hh ll D6 ee ff E6 ff F6 9E D6 ee ff 9E E6 ff	2 3 4 4 3 3 5 4	pp rpp prpp prpp rpp rfp pprpp prpp	0 1 1 -	- ↑ ↓ -
LDHX #opr16i LDHX opr8a LDHX opr16a LDHX ,X LDHX oprx16,X LDHX oprx8,X LDHX oprx8,SP	加载索引寄存器 (H:X) H:X "(M:M + \$0001)	IMM DIR EXT IX IX2 IX1 SP1	45 jj kk 55 dd 32 hh ll 9E AE 9E BE ee ff 9E CE ff 9E FE ff	3 4 5 5 6 5 5	ppp rrpp prrpp prrfp pprrpp prrpp prrpp	0 1 1 -	- ↑ ↓ -
LDX #opr8i LDX opr8a LDX opr16a LDX oprx16,X LDX oprx8,X LDX ,X LDX oprx16,SP LDX oprx8,SP	从存储器那里加载 X (索引寄存器低) X "(M)	IMM DIR EXT IX2 IX1 IX SP2 SP1	AE ii BE dd CE hh ll DE ee ff EE ff FE 9E DE ee ff 9E EE ff	2 3 4 4 3 3 5 4	pp rpp prpp prpp rpp rfp pprpp prpp	0 1 1 -	- ↑ ↓ -
LSL opr8a LSLA LSLX LSL oprx8,X LSL ,X LSL oprx8,SP	逻辑左移位 t  (同 ASL)	DIR INH INH IX1 IX SP1	38 dd 48 58 68 ff 78 9E 68 ff	5 1 1 5 4 6	rfwpp p p rfwpp rfwp prfwpp	↑ 1 1 -	- ↑ ↓ ↓
LSR opr8a LSRA LSRX LSR oprx8,X LSR ,X LSR oprx8,SP	逻辑右移位 t 	DIR INH INH IX1 IX SP1	34 dd 44 54 64 ff 74 9E 64 ff	5 1 1 5 4 6	rfwpp p p rfwpp rfwp prfwpp	↑ 1 1 -	- 0 ↑ ↓

表 7-2. 指令集小结 (第 6 页, 共 9 页)

Source Form	Operation	Address Mode	Object Code	Cycles	Cyc-by-Cyc Details	Affect on CCR	
						V	I
MOV opr8a,opr8a MOV opr8a,X+ MOV #opr8i,opr8a MOV ,X+,opr8a	移动 (M) <sub>destination</sub> " (M) <sub>source</sub> 在 IX+/DIR 和 DIR/IX+ 模式, H:X " (H:X) + \$0001	DIR/DIR DIR/IX+ IMM/DIR IX+/DIR	4E dd dd 5E dd 6E ii dd 7E dd	5 5 4 5	rwpwpp rfwpp pwpp rfwpp	0 1 1 -	-↑↑-
MUL	不带符号的乘法 X:A " (X) × (A)	INH	42	5	fffffp	- 1 1 0	- - - 0
NEG opr8a NEGA NEGX NEG oprx8,X NEG ,X NEG oprx8,SP	否定 (2 的补数) M " - (M) = \$00 - (M) A " - (A) = \$00 - (A) X " - (X) = \$00 - (X) M " - (M) = \$00 - (M) M " - (M) = \$00 - (M) M " - (M) = \$00 - (M)	DIR INH INH IX1 IX SP1	30 dd 40 50 60 ff 70 9E 60 ff	5 1 1 5 4 6	rfwpp p p rfwpp rfwp prfwpp	↑ 1 1 -	-↑↑↑
NOP	无操作 — 使用 1 总线周期	INH	9D	1	p	- 1 1 -	- - - -
NSA	半位元组交换累加器 A " (A[3:0]:A[7:4])	INH	62	1	p	- 1 1 -	- - - -
ORA #opr8i ORA opr8a ORA opr16a ORA oprx16,X ORA oprx8,X ORA ,X ORA oprx16,SP ORA oprx8,SP	累加器或存储器 " 兼或 " A " (A)   (M)	IMM DIR EXT IX2 IX1 IX SP2 SP1	AA ii BA dd CA hh ll DA ee ff EA ff FA 9E DA ee ff 9E EA ff	2 3 4 4 3 3 5 4	pp rpp prpp prpp rpp rfp pprpp prpp	0 1 1 -	-↑↑-
PSHA	将累加器推送到堆栈 推 (A); SP " (SP) - \$0001	INH	87	2	sp	- 1 1 -	- - - -
PSHH	将 H (索引寄存器高) 推送到堆栈上 推 (H); SP " (SP) - \$0001	INH	8B	2	sp	- 1 1 -	- - - -
PSHX	将 X (索引寄存器低) 推送到堆栈上 推 (X); SP " (SP) - \$0001	INH	89	2	sp	- 1 1 -	- - - -
PULA	从堆栈拉累加器 SP " (SP + \$0001); 拉 (A)	INH	86	3	ufp	- 1 1 -	- - - -
PULH	从堆栈拉 H (索引寄存器高) SP " (SP + \$0001); Pull (H)	INH	8A	3	ufp	- 1 1 -	- - - -
PULX	从堆栈拉 X (索引寄存器低) SP " (SP + \$0001); 拉 (X)	INH	88	3	ufp	- 1 1 -	- - - -
ROL opr8a ROL A ROLX ROL oprx8,X ROL ,X ROL oprx8,SP	通过进位左旋转 	DIR INH INH IX1 IX SP1	39 dd 49 59 69 ff 79 9E 69 ff	5 1 1 5 4 6	rfwpp p p rfwpp rfwp prfwpp	↑ 1 1 -	-↑↑↑
ROR opr8a RORA RORX ROR oprx8,X ROR ,X ROR oprx8,SP	通过进位右旋转 	DIR INH INH IX1 IX SP1	36 dd 46 56 66 ff 76 9E 66 ff	5 1 1 5 4 6	rfwpp p p rfwpp rfwp prfwpp	↑ 1 1 -	-↑↑↑

表 7-2. 指令集小结 (第 7 页, 共 9 页)

Source Form	Operation	Address Mode	Object Code	Cycles	Cyc-by-Cyc Details	Affect on CCR		
						V	1	1
RSP	复位堆栈指针 (低字节) SPL " \$FF (高字节未受影响)	INH	9C	1	p	-	1	-
RTI	从中断返回 SP " (SP) + \$0001; Pull (CCR) SP " (SP) + \$0001; Pull (A) SP " (SP) + \$0001; Pull (X) SP " (SP) + \$0001; Pull (PCH) SP " (SP) + \$0001; Pull (PCL)	INH	80	9	uuuuuufppp	↑	1	↑
RTS	从子程序返回 SP " SP + \$0001; Pull (PCH) SP " SP + \$0001; Pull (PCL)	INH	81	5	ufppp	-	1	-
SBC #opr8i SBC opr8a SBC opr16a SBC oprx16,X SBC oprx8,X SBC ,X SBC oprx16,SP SBC oprx8,SP	减去进位 A " (A) - (M) - (C)	IMM DIR EXT IX2 IX1 IX SP2 SP1	A2 ii B2 dd C2 hh ll D2 ee ff E2 ff F2 9E D2 ee ff 9E E2 ff	2 3 4 4 3 3 5 4	pp rpp prpp prpp rpp rfp pprpp prpp	↑	1	1
SEC	设置进位 (C " 1)	INH	99	1	p	-	1	-
SEI	设置中断屏蔽位 (I " 1)	INH	9B	1	p	-	1	-
STA opr8a STA opr16a STA oprx16,X STA oprx8,X STA ,X STA oprx16,SP STA oprx8,SP	将累加器保存在存储器中 M " (A)	DIR EXT IX2 IX1 IX SP2 SP1	B7 dd C7 hh ll D7 ee ff E7 ff F7 9E D7 ee ff 9E E7 ff	3 4 4 3 2 5 4	wpp pwpp pwpp wpp wp ppwpp pwpp	0	1	1
STHX opr8a STHX opr16a STHX oprx8,SP	保存 H:X (索引寄存器) (M:M + \$0001) " (H:X)	DIR EXT SP1	35 dd 96 hh ll 9E FF ff	4 5 5	wwpp pwwpp pwwpp	0	1	1
STOP	使能中断：停止处理 参见 MCU 文档 I 位 " 0; 停止处理	INH	8E	2	fp...	-	1	-
STX opr8a STX opr16a STX oprx16,X STX oprx8,X STX ,X STX oprx16,SP STX oprx8,SP	将 X (索引寄存器的低 8 位) 保存在存储器中 M " (X)	DIR EXT IX2 IX1 IX SP2 SP1	BF dd CF hh ll DF ee ff EF ff FF 9E DF ee ff 9E EF ff	3 4 4 3 2 5 4	wpp pwpp pwpp wpp wp ppwpp pwpp	0	1	1

表 7-2. 指令集小结 (第 8 页, 共 9 页)

Source Form	Operation	Address Mode	Object Code	Cycles	Cyc-by-Cyc Details	Affect on CCR	
						V	I
SUB #opr8i SUB opr8a SUB opr16a SUB oprx16,X SUB oprx8,X SUB ,X SUB oprx16,SP SUB oprx8,SP	减 A" (A) - (M)	IMM DIR EXT IX2 IX1 IX SP2 SP1	A0 ii B0 dd C0 hh ll D0 ee ff E0 ff F0 9E D0 ee ff 9E E0 ff	2 3 4 4 3 3 5 4	pp rpp prpp prpp rpp rfp pprpp prpp	↑ 1 1 -	-↑↑↑
SWI	软件中断 PC" (PC) + \$0001 推 (PCL); SP" (SP) - \$0001 推 (PCH); SP" (SP) - \$0001 推 (X); SP" (SP) - \$0001 推 (A); SP" (SP) - \$0001 推 (CCR); SP" (SP) - \$0001 I" 1; PCH" 中断向量高字节 PCL" 中断向量低字节	INH	83	11	sssssvvfppp	- 1 1 -	1 ---
TAP	将累加器转移到 CCR CCR" (A)	INH	84	1	p	↑ 1 1 ↑	↑↑↑↑
TAX	将累加器转移到 X (索引寄存器低) X" (A)	INH	97	1	p	- 1 1 -	--- --
TPA	将 CCR 转移到累加器 A" (CCR)	INH	85	1	p	- 1 1 -	--- --
TST opr8a TSTA TSTX TST oprx8,X TST ,X TST oprx8,SP	负数或 0(M) 测试 (M) - \$00 (A) - \$00 (X) - \$00 (M) - \$00 (M) - \$00 (M) - \$00	DIR INH INH IX1 IX SP1	3D dd 4D 5D 6D ff 7D 9E 6D ff	4 1 1 4 3 5	rfpp p p rfpp rfp prfpp	0 1 1 -	-↑↑-
TSX	将 SP 转移到索引寄存器 . H:X" (SP) + \$0001	INH	95	2	fp	- 1 1 -	--- --
TXA	将 X(索引寄存器低) 转移到累积器上 A" (X)	INH	9F	1	p	- 1 1 -	--- --

表 7-2. 指令集小结 (第 9 页, 共 9 页)

Source Form	Operation	Address Mode	Object Code	Cycles	Cyc-by-Cyc Details	Affect on CCR		
						V	1	1
TXS	将索引寄存器转移到 SP SP " (H:X) - \$0001	INH	94	2	f <sub>p</sub>	-	1	1
WAIT	使能中断; 等待中断 I 位 = 0; 暂停 CPU	INH	8F	2+	f <sub>p</sub> ...	-	1	1

源形式：“源形式”栏中的所有内容（斜体字符除外）都是文字报文，必须严格按照栏中显示的那样出现在汇编源文件中。开始的 3 到 5 个字母助记符和字符（#，( ) 和 +）通常是文字字符。

*n* 任何值为 0-7 间的一个整数的标签或表达

*opr8i* 任何值为 8 位立即值的标签或表达

*opr16i* 任何值为 16 位立即值的标签或表达

*opr8a* 任何值为 8 位直接页面地址 (\$00xx) 的标签或表达

*opr16a* 任何值为 16 位地址的标签或表达

*opr8x* 任何值为不带符号的 8 位值（用于索引寻址）的标签或表达

*opr16x* 任何值为 16 位值（用于索引寻址）的标签或表达

*rel* 任何从下一个指令开始，参考 -128 至 +127 位置内地址的标签或表达

#### 运算符号：

A 累加器  
CCR 条件码寄存器  
H 索引寄存器高字节  
M 存储器位置  
*n* 任意位  
*opr* 操作数（1 个或 2 个字节）  
PC 程序计数器  
PCH 程序计数器高字节  
PCL 程序计数器低字节  
*rel* 相关程序计数器偏移字节  
SP 堆栈指针  
SPL 堆栈指针低字节  
X 索引寄存器低字节  
& 和  
| 或  
Y 异或  
( ) 内容  
+ 加  
- 减, 否 (2 的补数)  
× 乘  
÷ 除  
# 立即值  
.. 加载  
: 级联

#### 寻址模式：

DIR 直接寻址模式  
EXT 扩展寻址模式  
IMM 立即寻址模式  
INH 固有寻址模式  
IX 有索引、无偏移寻址模式  
IX1 有索引、8 位偏移寻址模式  
IX2 有索引、16 位偏移寻址模式  
IX+ 有索引、无偏移、后增量寻址模式  
IX1+ 有索引、8 位偏移、后增量寻址模式  
REL 相关寻址模式  
SP1 堆栈指针、8 位偏移寻址模式  
SP2 堆栈指针、16 位偏移寻址模式

#### 逐周期代码：

f 空闲周期。这表示 CPU 不需要使用系统总线的周期。  
f 周期通常是系统总线时钟的一个周期，且总是读取周期。  
程序获取，从程序内存的下一个连续位置读取  
p 程序获取，从程序内存的下一个连续位置读取  
r 读取 8 位操作数  
s 推送（写入）1 个字节到堆栈  
u 从堆栈上弹出（读）一个字节  
v 从 \$FFxx 中读取向量（高字节优先）  
w 写 8 位操作数

#### CCR 位：

V 溢出位  
H 半进位  
I 中断屏蔽  
N 负数位  
Z 0 位  
C 进位 / 借位

#### CCR 影响：

↑ 设置或清除  
- 无影响  
U 未定义

表 7-3. 操作码映射 (第1页, 共2页)

Bit-Manipulation	Branch	Read-Modify-Write										Control			Register/Memory																					
		00	5	10	5	20	3	30	5	40	1	50	1	60	5	70	4	80	9	90	3	A0	2	B0	3	C0	4	D0	4	E0	3	F0	3			
BRSETO 3 DIR	BSETO 2 DIR	5	20	3	30	5	40	1	NEGA 1 INH	50	1	NEGX 1 INH	1	60	5	70	4	RTI 1 INH	9	90	3	A0	2	B0	3	C0	4	D0	4	E0	3	F0	3			
BRCLR0 3 DIR	BCLR0 2 DIR	5	21	3	31	5	41	4	CBEQ 3 DIR	51	4	CBEQA 3 IMM	1	61	5	71	5	CBEQ 2 IX+	81	6	91	3	A1	2	B1	3	C1	4	D1	4	E1	3	F1	3		
BRSET1 3 DIR	BSET1 2 DIR	5	22	3	32	5	42	5	MUL 3 EXT	52	6	DIV 1 INH	1	62	1	72	1	DAA 1 INH	82	5+	92	3	A2	2	B2	3	C2	4	D2	4	E2	3	F2	3		
BRCLR1 3 DIR	BCLR1 2 DIR	5	23	3	33	5	43	1	COM 2 DIR	53	1	COMA 1 INH	1	63	5	73	4	SWI 1 IX	83	11	93	3	A3	2	B3	3	C3	4	D3	4	E3	3	F3	3		
BRSET2 3 DIR	BSET2 2 DIR	5	24	3	34	5	44	1	LSR 2 DIR	54	1	LSRA 1 INH	1	64	5	74	4	TAP 1 IX	84	1	94	2	A4	2	B4	3	C4	4	D4	4	E4	3	F4	3		
BRCLR2 3 DIR	BCLR2 2 DIR	5	25	3	35	4	45	3	STHX 2 DIR	55	4	LDHX 3 IMM	2	65	3	75	5	CPHX 2 DIR	85	1	95	2	A5	2	B5	3	C5	4	D5	4	E5	3	F5	3		
BRSET3 3 DIR	BSET3 2 DIR	5	26	3	36	5	46	1	ROR 2 DIR	56	1	RORA 1 INH	1	66	5	76	4	ROR 2 IX1	86	3	96	5	A6	2	B6	3	C6	4	D6	4	E6	3	F6	3		
BRCLR3 3 DIR	BCLR3 2 DIR	5	27	3	37	5	47	1	ASR 2 DIR	57	1	ASRX 1 INH	1	67	5	77	4	PSHA 1 IX	87	2	97	1	A7	2	B7	3	C7	4	D7	4	E7	3	F7	2		
BRSET4 3 DIR	BSET4 2 DIR	5	28	3	38	5	48	1	LSL 2 DIR	58	1	LSLA 1 INH	1	68	5	78	4	PULX 1 IX1	88	3	98	1	A8	2	B8	3	C8	4	D8	4	E8	3	F8	3		
BRCLR4 3 DIR	BCLR4 2 DIR	5	29	3	39	5	49	1	ROL 2 DIR	59	1	ROLX 1 INH	1	69	5	79	4	PSHX 1 IX1	89	2	99	1	A9	2	B9	3	C9	4	D9	4	E9	3	F9	3		
BRSET5 3 DIR	BSET5 2 DIR	5	2A	3	3A	5	4A	1	DEC 2 DIR	5A	1	DECX 1 INH	1	6A	5	7A	4	PULH 1 IX1	90	3	9A	1	AA	2	BA	3	CA	4	DA	4	EA	3	FA	3		
BRCLR5 3 DIR	BCLR5 2 DIR	5	2B	3	3B	7	4B	4	DBNZ 2 DIR	5B	4	DBNZ 2 INH	2	6B	7	7B	6	PSHH 1 IX1	91	2	9B	1	AB	2	BB	3	CB	4	DB	4	EB	3	FB	3		
BRSET6 3 DIR	BSET6 2 DIR	5	2C	3	3C	5	4C	1	INC 2 DIR	5C	1	INCA 1 INH	1	6C	5	7C	4	CLRH 1 IX1	92	1	9C	1	RSP 1 INH		BC	3	CC	4	DC	4	EC	3	FC	3		
BRCLR6 3 DIR	BCLR6 2 DIR	5	2D	3	3D	4	4D	1	TST 2 DIR	5D	1	TSTA 1 INH	1	6D	4	7D	3	TST 2 IX1	93	1	9D	1	NOP 1 INH	2	BSR	5	BD	5	CD	6	DD	6	ED	5	FD	5
BRSET7 3 DIR	BSET7 2 DIR	5	2E	3	3E	6	4E	5	CPHX 3 EXT	5E	5	MOV 2 DD	4	6E	4	7E	5	MOV 2 IX+D	8E	2+	9E	2	AE	2	BE	3	CE	4	DE	4	EE	3	FE	3		
BRCLR7 3 DIR	BCLR7 2 DIR	5	2F	3	3F	5	4F	1	CLR 2 DIR	5F	1	CLRA 1 INH	1	6F	5	7F	4	CLR 2 IX1	8F	2+	9F	1	AF	2	BF	3	CF	4	DF	4	EF	3	FF	2		

INH 固有  
IMM 立即  
DIR 直接  
EXT 扩展  
DD DIR 至 DIR  
IX+ DIR 至 DIR

REL 相关  
IX 索引、无偏移  
IX1 索引、8位偏移  
IX2 索引、16位偏移  
IMD IMM 至 DIR  
DIX+ DIR 至 IX+

SP1 堆栈指针, 8位偏移  
SP2 堆栈指针, 16位偏移  
IX+ 有索引、无偏移、  
带后增量  
IX1+ 有索引、1字节偏移、  
带后增量

十六进制操作码 

F0	3
SUB	IX

 HCS08 周期  
字节数 1 指令助记符  
寻址模式

表 7-3. 操作码映射 (第 2 页, 共 2 页)

Bit-Manipulation	Branch	Read-Modify-Write			Control			Register/Memory					
				9E60 6 NEG 3 SP1				9ED0 5 SUB 4 SP2	9EE0 4 SUB 3 SP1				
				9E61 6 CBEQ 4 SP1				9ED1 5 CMP 4 SP2	9EE1 4 CMP 3 SP1				
								9ED2 5 SBC 4 SP2	9EE2 4 SBC 3 SP1				
				9E63 6 COM 3 SP1				9ED3 5 CPX 4 SP2	9EE3 4 CPX 3 SP1	9EF3 6 CPHX 3 SP1			
				9E64 6 LSR 3 SP1				9ED4 5 AND 4 SP2	9EE4 4 AND 3 SP1				
				9E66 6 ROR 3 SP1				9ED5 5 BIT 4 SP2	9EE5 4 BIT 3 SP1				
				9E67 6 ASR 3 SP1				9ED6 5 LDA 4 SP2	9EE6 4 LDA 3 SP1				
				9E68 6 LSL 3 SP1				9ED7 5 STA 4 SP2	9EE7 4 STA 3 SP1				
				9E69 6 ROL 3 SP1				9ED8 5 EOR 4 SP2	9EE8 4 EOR 3 SP1				
				9E6A 6 DEC 3 SP1				9ED9 5 ADC 4 SP2	9EE9 4 ADC 3 SP1				
				9E6B 8 DBNZ 4 SP1				9EDA 5 ORA 4 SP2	9EEA 4 ORA 3 SP1				
				9E6C 6 INC 3 SP1				9EDB 5 ADD 4 SP2	9EEB 4 ADD 3 SP1				
				9E6D 5 TST 3 SP1									
				9E6F 6 CLR 3 SP1				9EAE 5 LDHX 2 IX	9EBE 6 LDHX 4 IX2	9ECE 5 LDHX 3 IX1	9EDE 5 LDX 4 SP2	9EEE 4 LDX 3 SP1	9EFE 5 LDHX 3 SP1
											9EDF 5 STX 4 SP2	9EEF 4 STX 3 SP1	9EFF 5 STHX 3 SP1

INH 固有  
IMM 即时  
DIR 直接  
EXT 扩展  
DD DIR 至 DIR  
IX+D IX+ 全 DIR

REL 相关  
IX 索引、无偏移  
IX1 索引、8位偏移  
IX2 索引、16位偏移  
IM/D IMM 至 DIR  
DIX+ DIR 全 IX+

SP1 堆栈指针、8位偏移  
SP2 堆栈指针、16位偏移  
IX+ 有索引、无偏移、  
带后增量  
IX1+ 有索引、1字节偏移、  
带后增量

注意：第 2 页的所有操作码前面都带有 Page 2 Prebyte (9E)

十六进制操作码 | 9E60 6  
NEG  
字节数 3 SP1  
HCS08 周期  
指令助记符  
寻址模式



# 第 8 章

## 多功能时钟发生器（S08MCGV1）

### 8.1 介绍

多功能时钟发生器（MCG）模块为 MCU 提供了几个时钟源选项。MCG 模块中包含 1 个锁频环（FLL）和 1 个锁相环（PLL），可以由内部或外部参考时钟控制。模块可以选择 FLL 或 PLL 时钟作为 MCU 系统时钟，也可以选择内部或外部参考时钟作为 MCU 系统时钟。无论选择哪个时钟源，它都要通过降阶总线分频器，该分频器允许生成更低的输出时钟频率。MCG 还控制一个外部振荡器（XOSC），以便把晶体或共鸣器用作外部参考时钟。

MC9S08DZ60 系列的所有器件都含有 MCG 模块。

#### 注意

如需了解整个芯片的分配时钟源的更多信息，请参见 [1.3，“系统时钟分配”](#)。

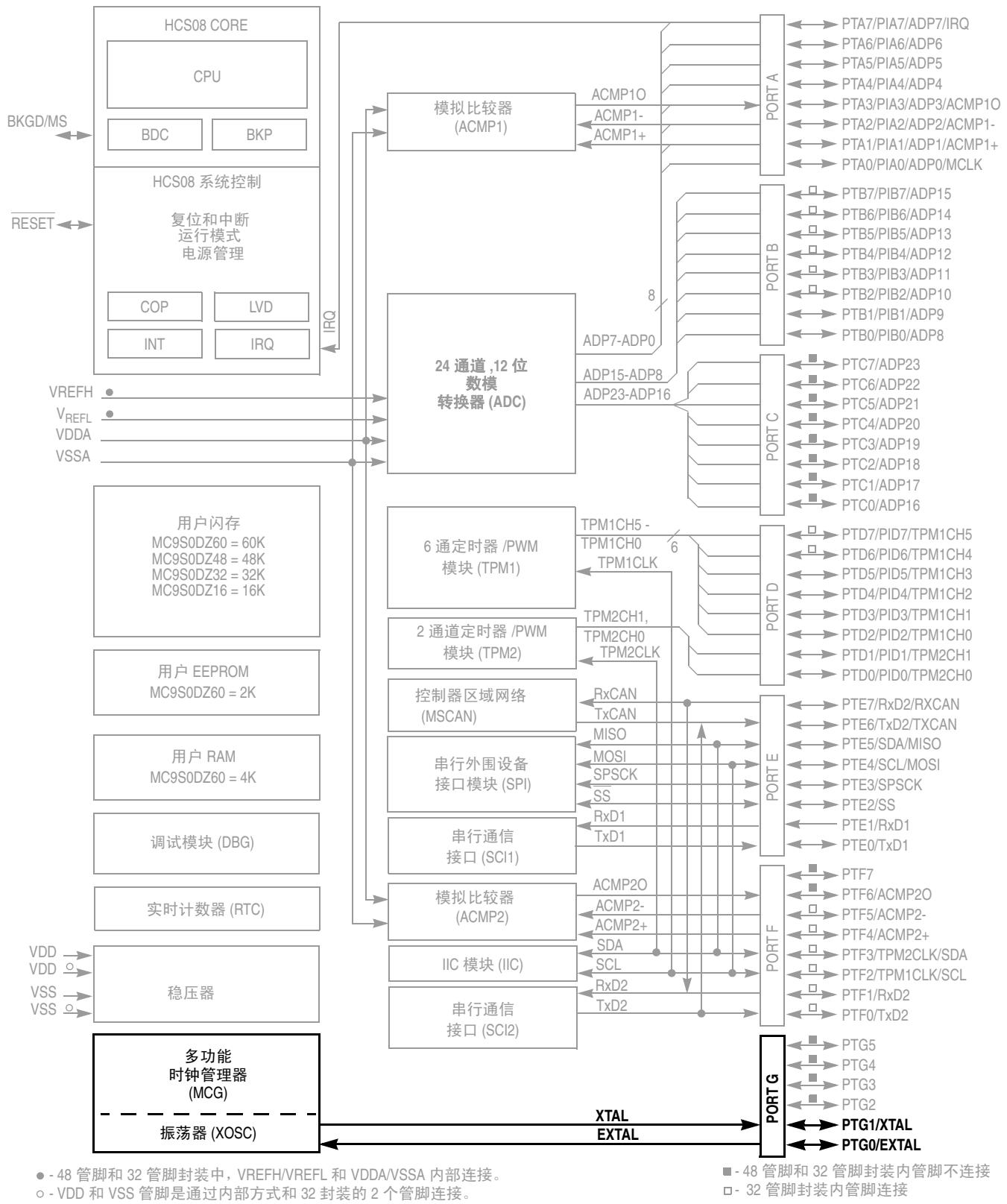


图 8-1. MC9S08DZ60 结构图

## 8.2.1 特性

MCG 模块的主要特性：

- 锁频环 (FLL)
  - 使用内部 32-kHz 参考时， 0.2% 分辨率
  - 使用内部 32-kHz 参考时， 全电压和温度范围内 2% 的偏差
  - 可以使用内部或外部参考控制 FLL
- 锁相环 (PLL)
  - 压控振荡器 (VCO)
  - 模数 VCO 分频器
  - 相位 / 频率检测器
  - 集成环路滤波器
  - 带中断功能的锁定检测器
- 内部参考时钟
  - 9 个调整位，确保精确度
  - 可选择为 MCU 的时钟源
- 外部参考时钟
  - 外部振荡器控制
  - 具有复位功能的时钟监控器
  - 可选择为 MCU 的时钟源
- 提供参考分频器
- 所选的时钟源可以除以 124 或 8
- 无论在 FLL 还是 PLL 模式中， BDC 时钟 (MCGLCLK) 是一个由 DCO 输出除以 2 得出的常量。

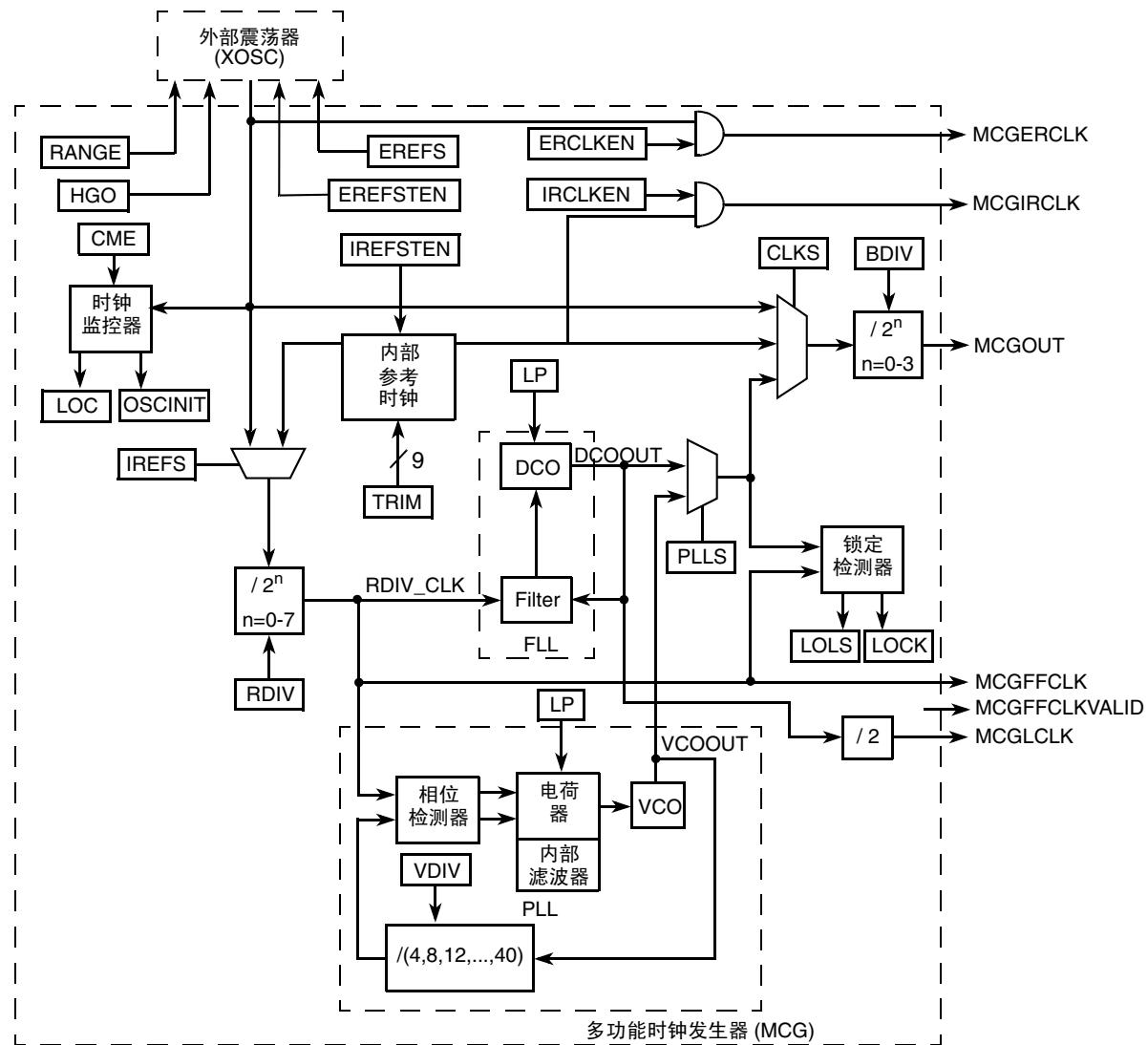


图 8-2. 多功能时钟发生器 (MCG) 结构图

## 8.2.2 运行模式

MCG 有 9 种运行模式：

- FLL Engaged Internal (FEI)
- FLL Engaged External (FEE)
- FLL Bypassed Internal (FBI)
- FLL Bypassed External (FBE)
- PLL Engaged External (PEE)
- PLL Bypassed External (PBE)
- Bypassed Low Power Internal (BLPI)
- Bypassed Low Power External (BLPE)
- Stop

如需了解更多信息 [8.5.1, “运行模式”](#)。

## 8.3 外部信号描述

没有片外连接的 MCG 信号

## 8.4 寄存器定义

### 8.4.1 MCG 控制寄存器 1 (MCGC1)

	7	6	5	4	3	2	1	0
R W	CLKS		RDIV		IREFS	IRCLKEN	IREFSTEN	
复位：	0	0	0	0	0	1	0	0

图 8-3. MCG 控制寄存器 1 (MCGC1)

表 8-1. MCG 控制寄存器 1 字段描述

字段	描述
7:6 CLKS	<b>时钟源选择</b> — 选择系统时钟源 00 Encoding 0 — 选择 FLL 或 PLL 输出。 01 Encoding 1 — 选择内部参考时钟。 10 Encoding 2 — 选择外部参考时钟。 11 Encoding 3 — 预留的， 默认为 00。 .
5:3 RDIV	<b>参考分频器</b> — 选择要分配给 IREFS 位选定参考时钟的量。如果选择 FLL，得到的频率必须在 31.25 kHz --39.0625 kHz 之间；如果选择 PLL，得到的频率必须在 1 MHz -- 2 MHz 之间。 000 Encoding 0 — 参考时钟除以 1 (复位默认) 001 Encoding 1 — 参考时钟除以 2 010 Encoding 2 — 参考时钟除以 4 011 Encoding 3 — 参考时钟除以 8 100 Encoding 4 — 参考时钟除以 16 101 Encoding 5 — 参考时钟除以 32 110 Encoding 6 — 参考时钟除以 64 111 Encoding 7 — 参考时钟除以 128
2 IREFS	<b>内部参考选择</b> — 选择参考时钟源 1 选择内部参考时钟 0 选择外部参考时钟
1 IRCLKEN	<b>内部参考时钟使能</b> — 使能内部参考时钟，用作 MCGIRCLK。 1 MCGIRCLK 使能 0 MCGIRCLK 禁止
0 IREFSTEN	<b>内部参考停止使能</b> — 控制着当 MCG 进入停止模式时，内部参考时钟是否仍保持使能状态。 1 如果设置了 IRCLKEN 或者在 MCG 进入停止状态前已处于 FEI、 FBI 或 BLPI 模式，那么内部参考时钟在停止状态中保持使能。 0 内部参考时钟在停止状态中被禁止。

#### 8.4.2 MCG 控制寄存器 2 (MCGC2)

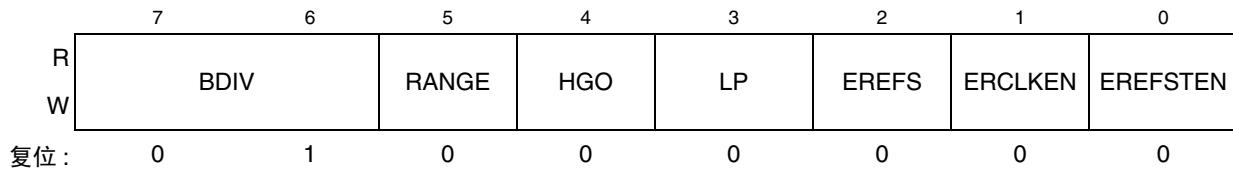


图 8-4. MCG 制寄存器 2 (MCGC2)

表 8-2. MCG 控制寄存器 2 字段描述

字段	描述
7:6 BDIV	<b>总线分频器</b> — 选择被 MCGC1 寄存器中的 CLKS 位所选的时钟源除的数字。这样可以控制总线频率。 00 Encoding 0 — 用所选时钟除以 1 01 Encoding 1 — 用所选时钟除以 2 (复位默认) 10 Encoding 2 — 用所选时钟除以 4 11 Encoding 3 — 用所选时钟除以 8
5 RANGE	<b>频率范围选择</b> — 选择外部振荡器或外部时钟源的频率范围。 1 为外部振荡器选择 1 MHz --16 MHz 的高频率范围 (外部时钟源为 1 MHz-- 40 MHz) 0 为外部振荡器选择 32 kHz --100 kHz 的低频率范围 (外部时钟源为 32 kHz --1 MHz)
4 HGO	<b>高增益振荡器选择</b> — 控制外部振荡器的运行模式 1 配置外部振荡器的高增益运行 0 配置外部振荡器的低增益运行
3 LP	<b>低功率选择</b> — 控制是否在旁路模式中禁止 FLL (或 PLL)。 1 旁路模式中禁止 FLL (或 PLL) (低功率). 0 旁路模式中激活旁路模式中激活 FLL (或 PLL).
2 EREFS	<b>外部参考选择</b> 选择外部参考源。 1 选择振荡器 0 选择外部时钟源
1 ERCLKEN	<b>外部参考使能</b> — 使能外部参考时钟, 用作 MCGERCLK。 1 MCGERCLK 活动 0 MCGERCLK 未活动
0 EREFSTEN	<b>外部参考停止使能</b> — 控制着当 MCG 进入停止模式时, 内部参考时钟是否仍保持使能状态。 1 如果设置了 IRCLKEN 或者在 MCG 进入停止状态前已处于 FEI、FBI 或 BLPI 模式, 那么外部参考时钟在停止状态中保持使能。 0 外部参考时钟在停止状态中禁用。

### 8.4.3 MCG 修正寄存器 (MCGTRM)

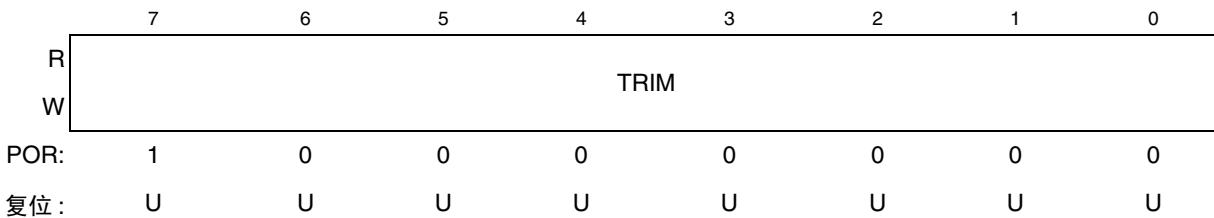


图 8-5. MCG 修正寄存器 (MCGTRM)

表 8-3. MCG 修正寄存器字段描

字段	描述
7:0 TRIM	<b>MCG 修正设置</b> — 通过控制内部参考时钟时段来控制内部参考时钟频率。TRIM 位是加权二进位 (即位 1 的调整次数是位 0 调整次数的两倍)。增大 TRIM 中的二进位值将延长这个时段, 减小 TRIM 值会缩短该时段。  MCGSC 中还提供了另外一个微调位, 即 FTRIM 位 t.  如果使用保存在非易失性存储器中的 TRIM[7:0] 值, 用户就有责任将这个值从非易失性存储器位置复制到该寄存器。r.

## 8.4.4 MCG 状态和控制寄存器 (MCGSC)

	7	6	5	4	3	2	1	0
R	LOLS	LOCK	PLLST	IREFST	CLKST		OSCINIT	FTRIM
W								
POR:	0	0	0	1	0	0	0	0
复位:	0	0	0	1	0	0	0	U

图 8-6. MCG 状态和控制寄存器 (MCGSC)

表 8-4. MCG 状态和控制寄存器字段描述

字段	描述
7 LOLS	<b>锁定状态丢失</b> — 该位是 FLL 或 PLL 锁定状态的标志。当锁定检测使能时，并且时钟已经锁定所定时，就设置 LOLS。锁定后，PLL 或 PLL 输出频率超出未锁定频率容限 Dunl 的范围。当 LOLIE 置位时，它决定是否在设置了 LOLS 时发送中断请求。当设置了 LOLS 时，可以通过复位或向 LOLS 写入逻辑 1 的方式清除 LOLS。向 LOLS 写入逻辑 0 不会产生影响。 0 自从上次清除 LOLS 以来，FLL 或 PLL 没有丢失锁定。 1 自从上次清除 LOLS 以来，FLL 或 PLL 丢失锁定。
6 LOCK	<b>锁定状态</b> — 显示 FLL 或 PLL 是否已获得锁定。当 PLL 和 FLL 都被禁止时，锁定检测也被禁止。如果设置了锁定状态位，那么修改 IREFS、PLLS、RDIV[2:0]、TRIM[7:0]（如果为 FEI 或 FBI 模式）或 VDIV[3:0]（如果为 PBE 或 PEE 模式）中的任何一个值都可能造成锁定状态位清除，并在 FLL 或 PLL 重新获得锁定之前一直保持清除状态。进入停止模式也会造成锁定状态位清除，并在 FLL 或 PLL 重新获得锁定之前一直保持清除状态。进入 BLPI 或 BLPE 模式也会造成锁定状态位清除，并在 MCG 退出这些模式前一直保持清除状态，直到 FLL 或 PLL 重新获得锁定。 0 FLL 或目前未被锁定。 1 FLL 或目前被锁定。
5 PLLST	<b>选择状态</b> — PLLST 位显示 PLLS 时钟的当前源。由于时钟域间的内部同步，在向 PLLST 位进行写入后，PLLST 位不会立即更新。 0 PLLS 时钟源是 FLL 时钟 1 PLLS 时钟源是 PLL 时钟
4 IREFST	<b>内部参考状态</b> — IREFST 位显示当前参考时钟的源。由于时钟域间的内部同步，在向 IREFST 位进行写入后，IREFS 位不会立即更新。 0 参考时钟源是外部参考时钟（振荡器或外部时钟源由 MCGC2 寄存器中的 EREFS 位决定 1 参考时钟源是内部参考时钟
3:2 CLKST	<b>时钟模式状态</b> — CLKST 位显示当前时钟模式。由于时钟域间的内部同步，在向 CLKST 位进行写入后，CLKS 位不会立即更新。 00 Encoding 0 — 选择 FLL 输出 01 Encoding 1 — 选择内部参考时钟 10 Encoding 2 — 选择外部参考时钟 11 Encoding 3 — 选择 PLL 输出
1 OSCINIT	<b>OCS 初始化</b> — 如果 ERCLKEN 选择了外部参考时钟源或者 MCG 正处于 FEE、FBE、PEE、PBE 或 BLPE 模式，并且设置了 EREFS，那么在完成了外部振荡器时钟的初始化周期后就要设置该位。只有当 EREFS 被清除或者 MCG 处于 FEI、FBI 或 BLPI 模式且 ERCLKEN 被清除时，这个位才被清除。
0 FTRIM	<b>MCG 微调</b> — 控制内部参考时钟频率最细微的调节。设置 FTRIM 会以最小的幅度延长该时段，清除 FTRIM 会以最小的幅度缩短该时段。  如果保存在非易失性存储器中 FTRIM 值被使用，用户有责任将这个值从非易失性存储器位置复制到该寄存器的 FTRIM 位上。

### 8.4.5 MCG Control Register 3 (MCGC3)

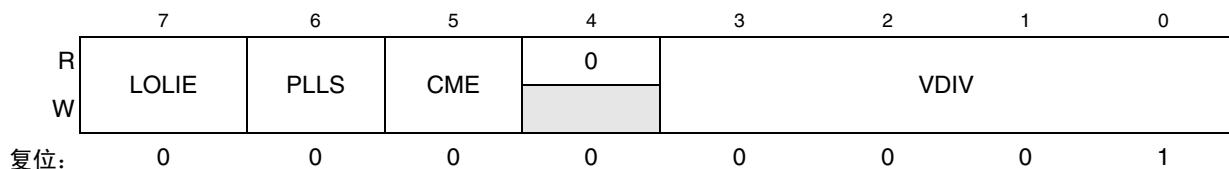


图 8-7. MCG PLL 寄存器 (MCGPLL)

表 8-5. MCG PLL 寄存器字段描述

字段	描述
7 LOLIE	<b>锁定中断丢失使能</b> — 决定是否在锁定丢失后产生中断请求。LOLIE 位只有在设置 LOLS 后才产生作用。 0 锁定丢失不生成请求 1 锁定丢失生成请求
6 PLLS	<b>PLL 选择</b> — 控制是选择 PLL 还是选择 FLL。如果 PLLS 位清除, PLL 在所有模式中都被禁止。如果设置了 PLLS, FLL 在所有模式中被禁止。 1 选择 PLL 0 选择 FLL
5 CME	<b>时钟监控器使能</b> 调决定是否在外部时钟显示丢失后是否发送复位请求。当 MCG 处于使用外部时钟 (FEE, FBE, PEE, PBE 或 BLPE) 的运行模式或使能外部参考 (在 MCGC2 寄存器中 ERCLKEN=1) 时, CME 位只能设置为逻辑 1。只要 CME 位设置为逻辑 1, MCGC2 寄存器中的 RANGE 位的值都不能更改。 0 时钟监控器禁止 1 外部时钟丢失生成复位请求
3:0 VDIV	<b>VCO 分频器</b> — 选择用来除 PLL 的 VCO 输出的值。VDIV 位确定被应用到参考时钟频率的倍频因子 (M)。 0000 ENCODING 0 — 预留的 0001 ENCODING 1 — 乘以 4. 0010 ENCODING 2 — 乘以 8. 0011 ENCODING 3 — 乘以 12. 0100 ENCODING 4 — 乘以 16. 0101 ENCODING 5 — 乘以 20. 0110 ENCODING 6 — 乘以 24. 0111 ENCODING 7 — 乘以 28. 1000 ENCODING 8 — 乘以 32. 1001 ENCODING 9 — 乘以 36. 1010 ENCODING 10 — 乘以 40. 1011 ENCODING 11 — 预留的 (默认设置为 M=40) 11xx ENCODING 12-15 — 预留的 (默认设置为 M=40)

## 8.5 特性描述

### 8.5.1 运行模式

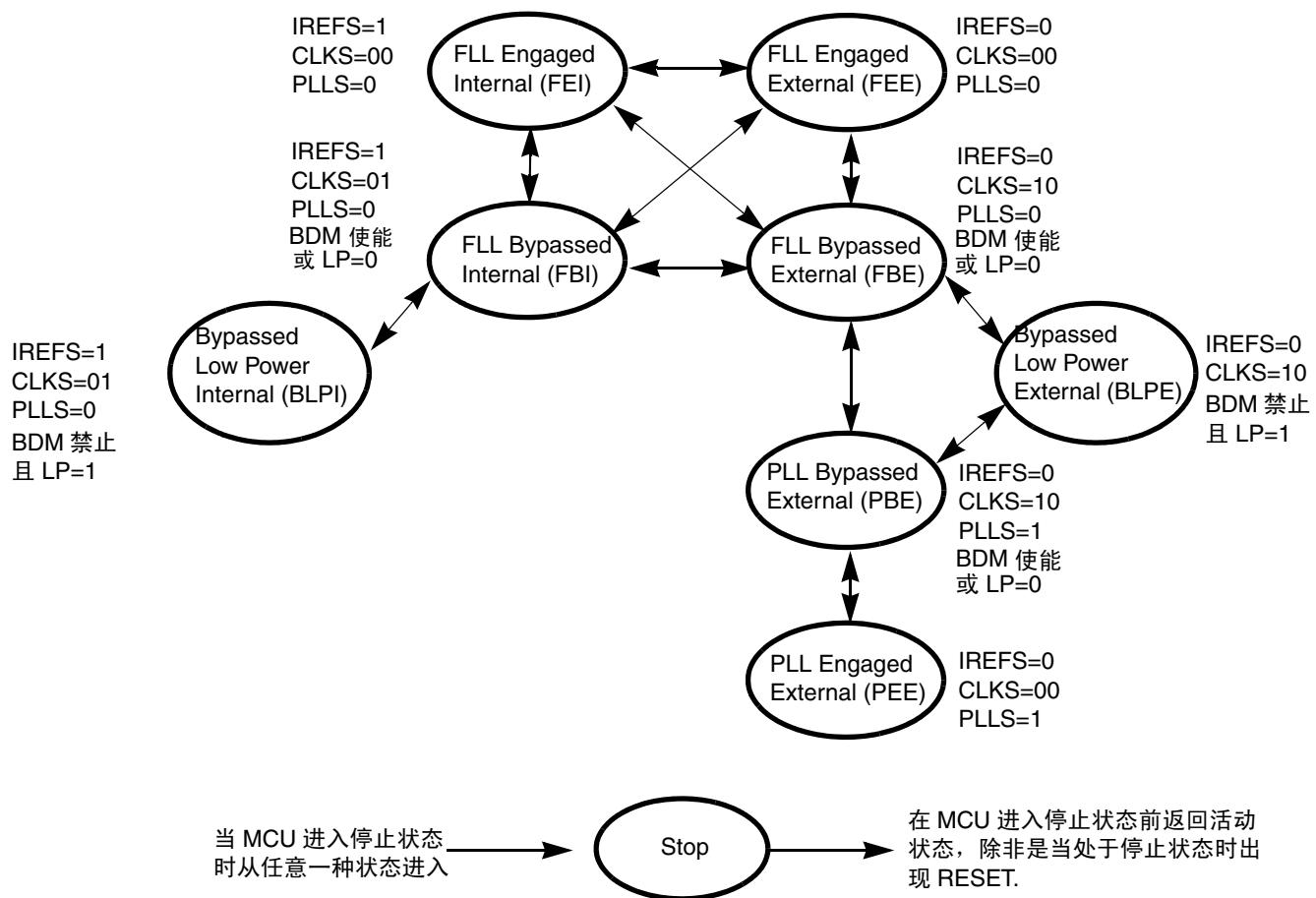


图 8-8. 时钟切换模式

MCG 的 9 种状态如状态示意图所示，并在下面做了详细地介绍。箭头显示状态间的允许移动方向。

### 8.5.1.1 FLL Engaged Internal (FEI)

FLL Engaged Internal (FEI) 是默认运行模式并且当满足下列条件时就进入该模式：

- CLKS 位写入 00
- IREFS 位写入 1
- PLLS 位写入 0
- RDIV 位写入 000。因为内部参考时钟频率在修正后应介于 31.25 kHz- 39.0625 kHz 之间，所以不需要进一步分频。

在 FLL Engaged Internal 模式中，MCGOUT 时钟源自 FLL 时钟，由内部参考时钟控制。FLL 时钟频率是由 RDIV 位选择的参考频率的 1024 倍。MCGLCLK 来自 FLL，PLL 被禁止并处于低功率状态。

### 8.5.1.2 FLL Engaged External (FEE)

当满足下列条件时就进入 FLL Engaged External (FEE) 模式：

- CLKS 位写入 00
- IIREFS 位写入 0
- PLLS 位写入 0
- RDIV 位写入介于 31.25 kHz- 39.0625 kHz 频率范围内的分频参考时钟。

在 FLL Engaged External 模式中，MCGOUT 时钟来自 FLL 时钟，由外部参考时钟控制。使能的外部参考时钟可以是外部晶体 / 谐振器，也可以是另外一个外部时钟源。FLL 时钟频率是由 RDIV 位选择的参考频率的 1024 倍。MCGLCLK 来自 FLL，PLL 被禁止并处于低功率状态。

### 8.5.1.3 FLL Bypassed Internal (FBI)

在 FLL Bypassed Internal (FBI) 模式中，MCGOUT 时钟来自内部参考时钟，FLL 处于运行状态但其输出时钟未使用。该模式对允许 FLL 获得目标频率非常有用，同时 MCGOUT 时钟由内部参考时钟驱动。

当满足以下条件时就进入 FLL Bypassed Internal 模式：

- CLKS 位写入 01
- IREFS 位写入 1
- PLLS 位写入 0
- RDIV 位写入 000。由于内部参考时钟频率在修正后应已经介于 31.25 kHz- 39.0625 kHz 之间，所以不需要进一步的分频。
- LP 位写入 0

在 FLL Bypassed Internal 模式中，MCGOUT 时钟源自内部参考时钟。FLL 时钟由内部参考时钟控制。FLL 时钟频率是由 RDIV 位选择的参考频率的 1024 倍。MCGLCLK 来自 FLL，PLL 被禁止并处于低功率状态。

### 8.5.1.4 FLL Bypassed External (FBE)

在 FLL Bypassed External (FBE) 模式中，MCGOUT 时钟来自外部参考时钟，FLL 处于运行状态但其输出时钟未使用。该模式对允许 FLL 获得目标频率非常有用，同时 MCGOUT 时钟由内部参考时钟驱动。

当满足以下条件时就进入 FLL Bypassed External 模式：

- CLKS 位写入 10
- IREFS 位写入 0
- PLLS 位写入 0
- 位写入介于 31.25 kHz- 39.0625 kHz 频率范围内的分频参考时钟。
- LP 位写入 0

在 FLL Bypassed External 模式中，MCGOUT 时钟源自 FLL 时钟。使能的外部参考时钟可以是外部晶体 / 谐振器，也可以是另外一个外部时钟源。FLL 时钟由外部参考时钟控制，FLL 时钟频率是由 RDIV 位选择的参考频率的 1024 倍。MCGLCLK 来自 FLL，PLL 被禁止处于低功率状态。

#### 注意

可以用大于指定最大频率的 FLL 参考时钟频率在 FBE 模式中短时间运行。这在使用频率大于 5 MHz 的外部晶体运行于 PEE 模式中是必需的。如需了解详细的示例信息，请参见 8.6.2.4，“[示例 4：从 FEI 转换到 PEE 模式：外部晶体 = 8 MHz、总线频率 = 8 MHz](#)”。

### 8.5.1.5 PLL Engaged External (PEE)

当满足以下条件时就进入 PLL Engaged External (PEE) 模式：

- CLKS 位写入 00
- IREFS 位写入 0
- PLLS 位写入 1
- RDIV 位写入介于 1 MHz - 2 MHz 频率范围内的分频参考时钟。

在 PLL Engaged External 模式中，MCGOUT 时钟源自 PLL 时钟，由外部参考时钟控制。使能的外部参考时钟可以是外部晶体 / 谐振器，也可以是另外一个外部时钟源。PLL 时钟频率是参考频率 (RDIV 位所选) 和倍频因子 (VDIV 位所选) 乘积。如果使能 BDM，MCGLCLK 值就是 DCO 除以 2 (开放环路模式) 的得数。如果禁止 BDM，那么 FLL 被禁止且处于低功率状态。

### 8.5.1.6 PLL Bypassed External (PBE)

在 PLL Bypassed External (PBE) 模式中，MCGOUT 时钟源自外部参考时钟，PLL 处于运行状态但其输出时钟未使用。该模式对允许 PLL 获得目标频率非常有用，同时 MCGOUT 时钟由内部参考时钟驱动。

当满足以下条件时就进入 PLL Bypassed External 模式:

- CLKS 位写入 00
- IREFS 位写入 0
- PLLS 位写入 1
- RDIV 位写入介于 1 MHz - 2 MHz 频率范围的分频参考时钟。
- LP 位写入 0

在 PLL Bypassed External 模式中, MCGOUT 时钟源自外部参考时钟。使能的外部参考时钟可以是外部晶体 / 谐振器, 也可以是另外一个外部时钟源。PLL 时钟频率是参考频率 (RDIV 位所选) 和倍频因子 (VDIV 位所选) 乘积。如果使能 BDM, MCGLCLK 值就是 DCO 除以 2 (开放环路模式) 的得数。如果禁止 BDM, 那么 FLL 被禁止且处于低功率状态。

#### **8.5.1.7 Bypassed Low Power Internal (BLPI)**

当满足以下条件时就进入 Bypassed Low Power Internal (BLPI) 模式:

- CLKS 位写入 01
- IREFS 位写入 1
- PLLS 位写入 0
- LP 位写入 1
- BDM 模式未活动

在 Bypassed Low Power Internal 模式中, MCGOUT 时钟源自内部参考时钟。

在 BLPI 模式中, PLL 和 FLL 总是无效的, 且 MCGLCLK 不能用于 BDC 通信。如果 BDM 进入活动状态, 该模式将切换为由 PLLS 位状态决定的另外一种内部旁路模式。

#### **8.5.1.8 Bypassed Low Power External (BLPE)**

当满足以下条件时就进入 Bypassed Low Power External (BLPE) 模式:

- CLKS 位写入 10
- IREFS 位写入 0
- PLLS 位写入 0 或 1
- LP 位写入 1
- BDM 模式未活动

在 Bypassed Low Power External 模式中, MCGOUT 时钟源自外部参考时钟。使能的外部参考时钟可以是外部晶体 / 谐振器, 也可以是另外一个外部时钟源。

在 BLPE 模式中, PLL 和 FLL 总是无效的, 且 MCGLCLK 不能用于 BDC 通信。如果 BDM 进入活动状态, 该模式将切换到由 PLLS 位状态决定的另外一种外部旁路模式。

### 8.5.1.9 Stop

每当 MCU 进入 STOP 状态就进入 STOP 模式。在该模式中，FLL 和 PLL 被禁止，所有 MCG 时钟信号静止，但下列情况除外：

当满足下列条件时，MCGIRCLK 将在停止模式中使能：

- IRCLKEN = 1
- IREFSTEN = 1

当满足下列条件时，MCGERCLK 将在停止模式中使能：

- ERCLKEN = 1
- EREFSTEN = 1

### 8.5.2 模式切换

当在 Engaged Internal 和 Engaged External 间进行模式切换时，IREFS 位可以随时修改，但必须同时修改 RDIV 位，这样参考频率处于 PLLS 位状态所要求的范围内（如果选择 FLL，则介于 31.25 kHz -39.0625 kHz 之间；如果选择 PLL，则介于 1 MHz -2 MHz 之间）。在 IREFS 值更改后，FLL 或 PLL 在切换完成后会被再次锁定。切换完成由 IREFST 位标志。

对于切换到 FBE 模式后立即进入停止模式的特殊情况，如果外部时钟和内部时钟在停止模式中被禁止（EREFSTEN = 0，IREFSTEN = 0），在清除 IREFST 位后必须允许 100us 来切断内部参考。在大多数情况下，由于指令执行次数造成的延迟都足够用。

CLKS 位也可以随时修改，但为了正确配置 MCGLCLK，必须同时修改 RDIV 位，这样参考频率就处于 PLLS 位状态所要求的范围内（如果选择 FLL，则介于 31.25 kHz -39.0625 kHz 之间；如果选择 PLL，则介于 1 MHz -2 MHz 之间）。实际切换到刚选择的时钟由 CLKST 位表示。如果刚选择的时钟不可用，则仍选择原来的时钟。

如需了解更多信息，请参见图 8-8。

### 8.5.3 总线分频器

BDIV 位可以随时修改，实际切换到新频率将立即进行。

### 8.5.4 低功率位使用

提供低功率位 (LP) 的目的是为了在不需要使用 FLL 或 PLL 时禁止它们，从而达到节电目的。然而，在有些应用中可能需要使能 FLL 或 PLL，使它在切换到下一个 Engaged 模式前锁定，以实现最大精确度，只需向 LP 位写入 0，即可完成该操作。

### 8.5.5 内部参考时钟

当设置了 IRCLKEN 时，内部参考时钟信号将作为 MCGIRCLK 出现，作为另外一个时钟源使用。通过调整内部参考时钟时段，MCGIRCLK 的目标频率可以重新设定。在 MCGTRM 寄存器的 FRIM 位中写入一个新值就可以完成该操作。向 MCGTRM 寄存器写入一个更大的值将降低 MCGIRCLK 频率，写入一个更小的值将提高 MCGIRCLK 频率。如果 MCG 处于 FLL Engaged Internal (FEI)、FLL Bypassed Internal (FBI) 或 Bypassed Low Power Internal (BLPI) 模式，TRIM 位会影响 MCGOUT 频率。TRIM 和 FTRIM 值由 POR 初始化，但不会受其他复位影响。

如果 MCGIRCLK 未调整，编程低参考分频器 (RDIV) 因子可能导致 MCGOUT 频率超过芯片级最高频率，且违反芯片级时钟定时技术规范 (参见 [Device Overview](#))

如果 IREFSTEN 和 IRCLKEN 位均已设置，内部参考时钟将在停止模式期间保持运行，以便在退出停止模式时快速恢复。

### 8.5.6 外部参考时钟

MCG 模块可以支持 FEE 和 FBE 模式中频率在 31.25 kHz-5 MHz, PEE 和 PBE 模式中频率在 1 MHz -16 MHz 之间，BLPE 模式中频率在 0 - 40 MHz 之间的外部参考时钟。当设置了 ERCLKEN 时，内部参考时钟信号将作为 MGERCLK 出现，作为另外一个时钟源使用。当 IREFS = 1 时，FLL 或 PLL 不使用外部参考时钟，外部参考时钟只能用作 MGERCLK。在这些模式中，该频率可能等于芯片级定时规范支持的最大频率 (参见 [Device Overview](#))

如果 EREFSTEN 和 ERCLKEN 位均已设置，或者 MCG 处于 FEE、FBE、PEE、PBE 或 BLPE 模式，外部参考时钟将在停止模式期间保持运行，以便在退出停止模式时快速恢复。

如果 CME 位写入 1，时钟监控器使能。如果外部参考降到某一频率（根据 MCGC2 中的 RANGE 位可以是 floc\_high 或 floc\_low）以下，MCU 将复位。系统复位状态 (SRS) 寄存器中的 LOC 位将用来标志错误。

### 8.5.7 固定频率时钟

MCG 将分频参考时钟作为 MCGFFCLK，作为另外一个时钟源使用。MCGFFCLK 频率小于或等于 MCGOUT 频率的 1/4 才有效。正是因为这个要求，MCGFFCLK 在旁路模式中的如下 BDIV 和 RDIV 值组合中无效：

- BDIV=00 (除以 1)， RDIV
- BDIV=01 (除以 2)， RDIV

## 8.6 初始化 / 应用报文

本节描述了如何在应用中初始化和配置的 MCG 模块。后面几节给出了几个如何对 MCG 进行初始化、如何在不同模式间进行适当切换的例子。

## 8.6.1 MCG 模块初始化顺序

MCG 来自于为 FEI 模式配置的复位，其中为 BDIV 除以 2。在 FLL 获取锁定之前，内部参考能在 trefst 毫秒内稳定。一旦内部参考稳定，FLL 就会在 tfll\_lock 毫秒内获取锁定。

在 POR 时，内部参考需要进行调整以确保精确的时钟。飞思卡尔推荐使用闪存位置 0xFFAE 来保存 MCGSC 寄存器中的微调位 FTRIM、推荐 0xFFAF 来保存 MCGTRM 寄存器中的 8 位调整值。MCU 不会自动将这些闪存位置的数值复制到各自的寄存器中。因此，用户代码必须将这些值从闪存复制到寄存器中。

### 注意

在没有进行首次调整内部参考前，BDIV 值不应被更改为 divide-by-1。不这样做可能会导致 MCU 不符合技术规范。

### 8.6.1.1 初始化 MCG

由于 MCG 在复位后处于 FEI 模式，复位后可以直接切换到的 MCG 模式有 FEE、FBE 和 FBI 模式（参见图 8-8）。要直接切换到任何其他模式需要首先配置 MCG 为这三种初始模式中的一种。必须留意检查 MCGSC 寄存器中标志各个模式中所有配置更改的相关状态位。

要从 FEI 模式更改为 FEE 或 FBE 模式，请按下列步骤操作：

1. 使能在 MCGC2 中适当的位来使能外部时钟源；
2. 写至 MCGC1 以选择时钟模式：
  - 如果进入 FEE 模式，适当设置 RDIV、清除 IREFS 位，以切换到外部参考，让 CLKS 位停留在 %00，这样就可以把 FLL 输出选择为系统时钟源。
  - 如果进入 FBE，清除 IREFS 位以切换到外部参考，将 CLKS 位更改为 %10，这样就可以把外部参考选择为系统时钟源。这里还应根据外部参考频率适当设置 RDIV 位，因为尽管 FLL 被旁通，但它仍然处于 FBE 模式。
  - 内部参考可以通过设置 IRCLKEN 位保持运行。如果应用中需要在内部和外部模式之间来回切换，这就十分有用。为了实现最低功耗，当处于外部时钟模式时应禁止内部参考。
3. 在设置了正确的配置位后，等待 MCGSC 寄存器中受影响的位适当地改变，因为它们反应了 MCG 已经切换到正确模式。
  - 如果第 1 步中已经设置了 ERCLKEN，或者 MCG 处于 FEE、FBE、PEE、PBE 或 BLPE 模式，且第 1 步中也设置了 EREFS，等待 OSCINIT 位置位，OSCINIT 位的置位表明外部时钟源已经完成初始化周期且稳定下来。附录 A “电气性能” 中给出了正常情况下的晶体启动时间。
  - 如果是 FEE 模式，一定要确保在进一步操作前，IREFST 位已经清除且 LOCK 位置位。
  - 如果是 FBE 模式，请确保 IREFST 位已经清除，LOCK 位已经置位，CLKST 位已经更改为 %10，这样表明已经正确选择了外部参考时钟。尽管在 FBE 模式中 FLL 被旁通，但它仍处于打开，将在 FBE 模式中锁定。

要从 **FEI** 时钟模式转换到 **FBI** 时钟模式, 请按下列步骤操作:

1. 将 **CLKS** 位更改为 %01, 这样内部参考时钟就可以选择为系统时钟源。
2. 等待把 **MCGSC** 寄存器中的 **CLKST** 位更改为 %01, 表明已经正确选择了内部参考时钟。

## 8.6.2 MCG 模式切换

当在 MCG 的运行模式间切换时, 必须更改某些配置位, 以便从一种模式转到另一种模式。每次当更改这些位 (**PLLS**、**IREFS**、**CLKS** 或 **EREFs**) 中的任意一个时, 在进一步操作之前, 应用软件必须检查 **MCGSC** 寄存器中相应的位 (**PLLST**、**IREFST**、**CLKST** 或 **OSCINIT**)。

此外, 还必须确保适当设置参考时钟分频器 (**RDIV**), 使之适应将被切换到的模式。例如, 在 **PEE** 模式中, 如果使用 4 MHz 晶体, **RDIV** 必须设置为 %001 (除以 2) 或 %010 (除以 4), 以便将外部参考分频为 1-2MHz 之间的所需频率。

在更改 **PLLS** 位前, 一定要对 **RDIV** 和 **IREFS** 位进行适当设置, 这样 **FLL** 或 **PLL** 时钟才有适当的参考时钟频率进行切换。

下表表明使用 **RDIV**、**BDIV** 和 **V DIV** 设置, 为每种时钟模式进行了 **MCGOUT** 频率计算。总线频率等于 **MCGOUT** 除以 2。.

**表 8-6. MCGOUT 频率计算选项**

时钟模式	$f_{MCGOUT}^1$	注意
FEI (FLL engaged internal)	$(f_{int} * 1024) / B$	在复位后, 典型 $f_{MCGOUT} = 16$ MHz。RDIV 位设置为 %000。.
FEE (FLL engaged external)	$(f_{ext} / R * 1024) / B$	$f_{ext} / R$ 必须在 31.25 kHz-39.0625 kHz 的频率范围内。
FBE (FLL bypassed external)	$f_{ext} / B$	$f_{ext} / R$ 必须在 31.25 kHz -39.0625 kHz 的频率范围内。
FBI (FLL bypassed internal)	$f_{int} / B$	典型 $f_{int} = 32$ kHz
PEE (PLL engaged external)	$[(f_{ext} / R) * M] / B$	$f_{ext} / R$ 必须在 1MHz 和 2 MHz 的频率范围内。
PBE (PLL bypassed external)	$f_{ext} / B$	$f_{ext} / R$ 必须介于 1MHz - 2 MHz 的频率范围内。
BLPI (Bypassed low power internal)	$f_{int} / B$	
BLPE (Bypassed low power external)	$f_{ext} / B$	

<sup>1</sup> R 是 RDIV 位选择的参考分频器, B 是 BDIV 位选择的总线分频器, M 是 VDIV 位选择的多路复用器。

本节包括三种使用 4 MHz 外部晶体的模式切换示例。如果使用小于 1 MHz 的外部时钟源, 都不应将 MCG 配置成任意一种 PLL 模式 (PEE 和 PBE)。

### 8.6.2.1 示例 1: 从 FEI 切换到 PEE 模式: 外部晶体 = 4 MHz、总线频率 = 8 MHz

本例中, MCG 将通过适当的运行模式从 PEE 转换到 BLPI 模式, 直到设置了 4 MHz 晶体参考频率来实现 8 MHz 的总线频率。因为 MCG 复位后处于 FEI 模式, 本例还显示了在复位后如何初始化 MCG, 实现进入 PEE 模式。示例中首先介绍了代码序列, 然后提供了一个演示该顺序的流程图。

1. 首先, FEI 必须转换到 FBE 模式:
  - a)  $\text{MCGC2} = 0x36$  (%00110110)
    - BDIV (位 7 和 6) 设置为 %00 或除以 1
    - RANGE (位 5) 设置为 1, 因为 4 MHz 的频率位于高频范围。
    - HGO (位 4) 设置为 1, 配置外部振荡器以实现高增益运行;
    - EREFS (位 2) 设置为 1, 因为正在使用晶体;
    - ERCLKEN (位 1) 设置为 1, 确保外部参考时钟处于活动状态;
  - b) 循环检测, 直到 MCGSC 中 OSCINIT (位 1) 是 1, 表明 EREFS 位选择的晶体已经完成初始化。
  - c)  $\text{MCGC1} = 0xB8$  (%10111000)
    - CLKS (位 7 和 6) 设置为 %10, 以便选择外部参考时钟为系统时钟源。
    - RDIV (位 5-3) 设置为 %111 或 divide-by-128, 因为  $4 \text{ MHz} / 128 = 31.25 \text{ kHz}$ , 这在 FLL 要求的 31.25 kHz-- 39.0625 kHz 频率范围内。
    - IREFS (位 2) 清除至 0, 选择外部参考时钟。
  - d) 循环检测, 直到 MCGSC 中的 IREFST (位 4) 是 0, 表明外部参考是当前的参考时钟源。
  - e) 循环检测, 直到 MCGSC 中的 CLKST (位 3 和 2) 是 %10, 表明已经选择外部参考时钟为当前时钟模式的 MCGOUT 馈电。
2. 然后, FBE 必须直接转换到 PBE 模式或先转换到 BLPE 模式, 然后再转换到 PBE 模式:
  - a) BLPE: 如果需要从 BLPE 模式中转换, 首先把 MCGC2 中的 LP (位 3) 设置为 1。
  - b) BLPE/PBE:  $\text{MCGC1} = 0x90$  (%10010000)
    - RDIV (位 5-3) 设置为 %010 或除以 4, 因为  $4 \text{ MHz} / 4 = 1 \text{ MHz}$ , 这在 PLL 要求的 1MHz - 2 MHz 频率范围内。在 BLPE 模式中, RDIV 的配置不重要, 因为 FLL 和 PLL 都被禁止。更改它们只会建立供 PLL 在 PBE 模式中使用的分频器。
  - c) BLPE/PBE:  $\text{MCGC3} = 0x44$  (%01000100)
    - PLLS (位 6) 设置为 1, 选择 PLL。在 BLPE 模式中, 更改该位只会让 MCG 准备在 PBE 模式中的 PLL 使用
    - VDIV (位 3-0) 设置为 %0100 或乘以 16, 因为  $1 \text{ MHz} \text{ 参考} * 16 = 16 \text{ MHz}$ 。在 BLPE 模式中, VDIV 位的配置不重要, 因为 PLL 被禁止。更改它们只会为 PBE 模式中的 PLL 使用乘积因子。

- d) BLPE: 如果通过 BLPE 模式转换, 将 MCGC2 的 LP (位 3) 转换到 0, 切换到 PBE 模式。
  - e) PBE: 循环检测, 直到 MCGSC 中的 PLLST (位 5) 已经设置, 表明 PLLS 时钟的当前源是 PLL。
  - f) PBE: 循环检测, 直到 MCGSC 中的 LOCK (位 6) 已经设置, 表明 PLL 已经获得锁定。
3. 最后, PBE 模式转换到 PEE 模式:
- a) MCGC1 = 0x10 (%00010000)
    - MCGSC1 中的 CLKS (位 7 和 6) 设置为 %00, 以便将 PLL 输出选择为系统时钟源。
    - b) 循环检测, 直到 MCGSC 中的 CLKST (位 3 和 2) 是 %11, 表明已经选择 PLL 输出为当前时钟模式的 MCGOUT 馈电。
    - 这样, RDIV 除以 4、BDIV 除以 1、VDIV 乘以 16,  $MCGOUT = [(4 \text{ MHz} / 4) * 16] / 1 = 16 \text{ MHz}$ , 总线频率是 MCGOUT / 2 或 8 MHz

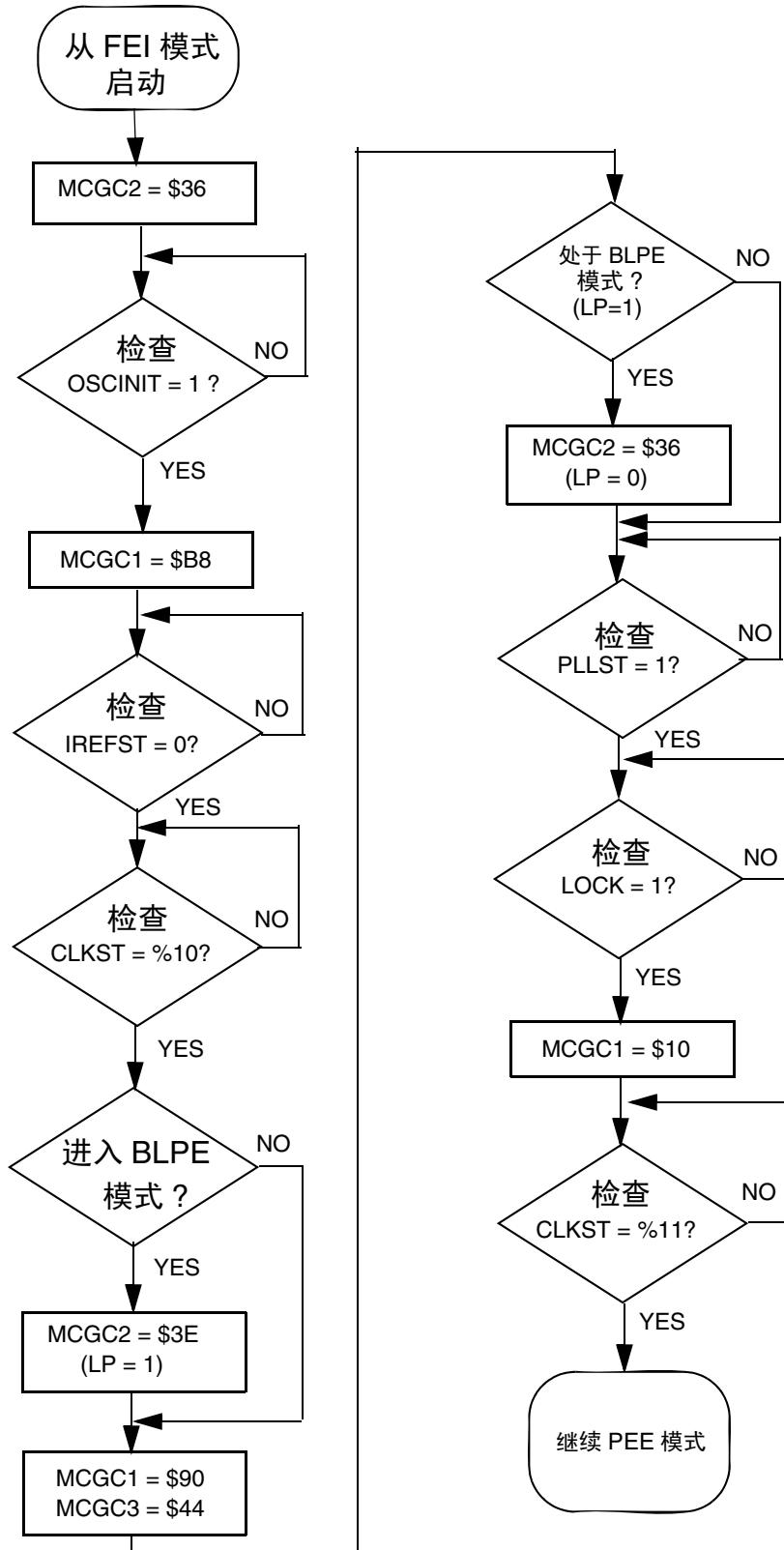


图 8-9. 使用 4 MHz 晶体从 FEI 转换到 PEE 模式的流程图

### 8.6.2.2 示例 2: 从 PEE 切换到 BLPI 模式: 外部晶体 = 4 MHz、总线频率 = 16 kHz

本例中, MCG 将通过适当的运行模式, 从晶体频率为 4MHz、总线频率为 8MHz 的 PEE 模式(参见前一示例)切换到总线频率为 16kHz 的 BLPI 模式。示例中首先介绍了代码序列, 然后提供了一个演示该顺序的流程图。

1. 首先, PEE 必须转换到 PBE 模式:
  - a) MCGC1 = 0x90 (%10010000)
    - CLKS (位 7 和 6) 设置为 %10, 以便把系统时钟源切换到外部参考时钟。
  - b) 循环检测, 直到 MCGSC 中 CLKST (位 3 和 2) 是 %10, 表明已经选择外部参考时钟为 MCGOUT 馈电。
2. 然后, PBE 必须要么直接转换到 FBE 模式, 要么先转换到 BLPE 模式, 然后再转换到 FBE 模式:
  - a) BLPE: 如果需要从 BLPE 模式转换, 首先把 MCGC2 中的 LP (位 3) 设置为 1。
  - b) BLPE/FBE: MCGC1 = 0xB8 (%10111000)
    - RDIV (位 5-3) 设置为 %111 或除以 128, 因为  $4 \text{ MHz} / 128 = 31.25 \text{ kHz}$ , 这在 FLL 要求的  $31.25 \text{ kHz} - 39.0625 \text{ kHz}$  频率范围内。在 BLPE 模式中, RDIV 的配置不重要, 因为 FLL 和 PLL 都被禁止。更改它们只会建立供 FLL 在 FBE 模式中使用的分频器。
    - c) BLPE/FBE: MCGC3 = 0x04 (%00000100)
      - PLLS (位 6) 清除至 0, 选择 FLL。在 BLPE 模式中, 更改该位只会让 MCG 准备在 FBE 模式中的 FLL 使用。如果 PLLS = 0, VDIV 值不重要。
    - d) BLPE: 如果通过 BLPE 模式转换, 将 MCGC2LP (位 3) 中的 LP 清除至 0, 切换到 FBE 模式。
    - e) FBE: 循环检测, 直到 MCGSC 中的 PLLST (位 5) 已经清除, 表明 PLLS 时钟的当前源是 FLL。
    - f) FBE: 循环检测, 直到在 MCGSC 中的 LOCK (位 6) 已经置位, 表明 FLL 已经获得锁定。尽管在 FBE 模式中 FLL 被旁通, 但它仍使能且在运行。
3. 接下来, FBE 模式转换到 FBI 模式:
  - a) MCGC1 = 0x44 (%01000100)
    - MCGSC1 中的 CLKS (位 7 和 6) 设置为 %01, 以便将系统时钟切换到内部参考时钟。
    - IREFS (位 2) 设置为 1, 选择内部参考时钟为参考时钟源。
    - RDIV (位 5-3) 设置为 %000 或除以 1, 因为调整后的内部参考应在 FLL 要求的  $31.25 \text{ kHz} - 39.0625 \text{ kHz}$  频率范围内。
  - b) 循环检测, 直到 MCGSC 中的 IREFST (位 4) 是 1, 表明已经选择内部参考时钟为参考时钟源。
  - c) 循环检测, 直到 MCGSC 中的 CLKST (位 3 和 2) 是 %01, 表明已经选择内部参考时钟为 MCGOUT 馈电。

4. 最后, FBI 转换到 FBILP 模式。  
 a) MCGC2 = 0x08 (%00001000)  
 — LP 中的 LP (位 3) 是 1。

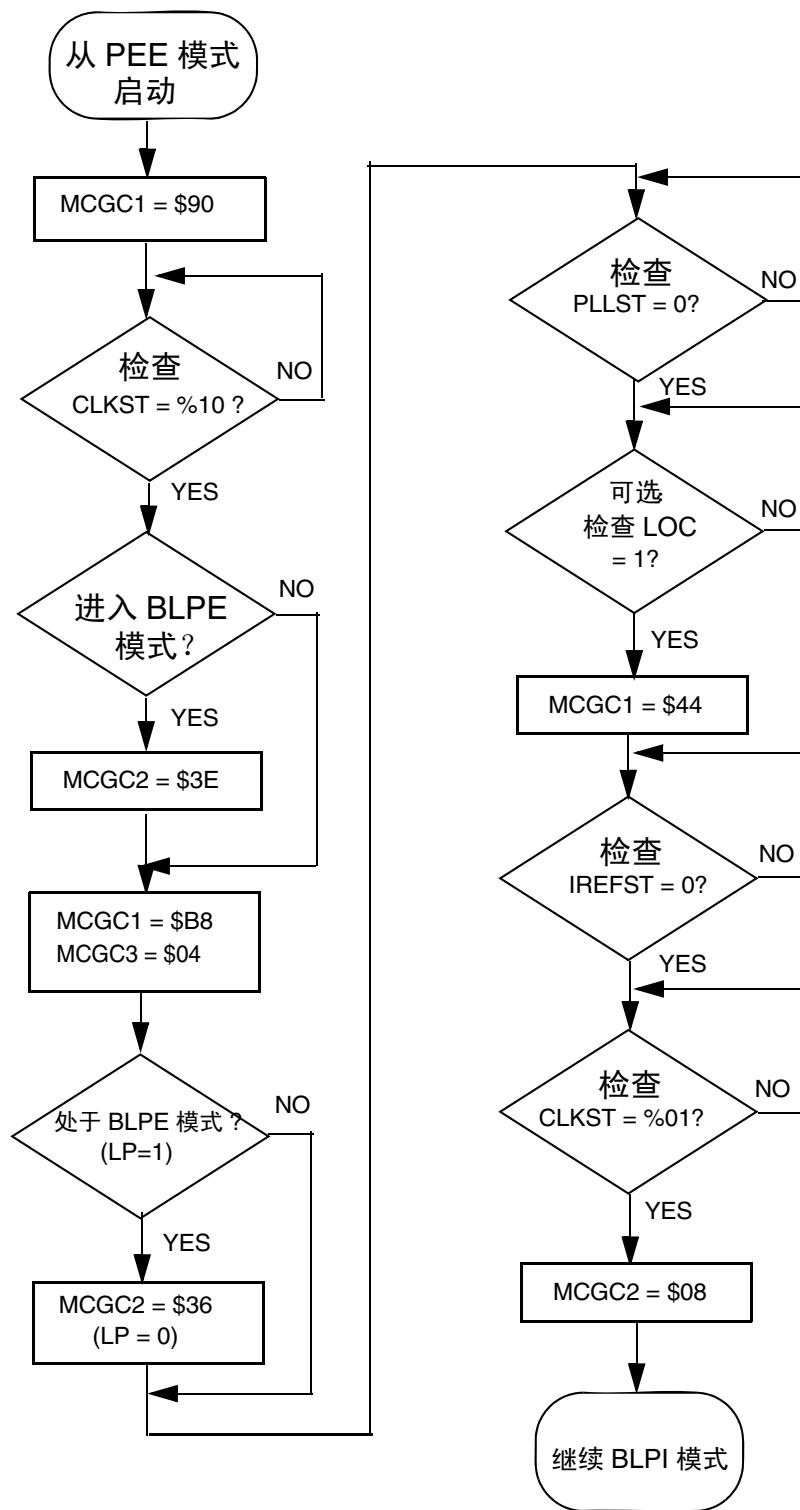


图 8-10. 使用 4 MHz 晶体从 PEE 转换到 BLPI 模式的流程图

### 8.6.2.3 示例 3: 从 BLPI 转换到 FEE 模式: 外部晶体 = 4 MHz、总线频率 = 16 MHz

本例中, MCG 将选择适当的运行模式, 从以基于内部参考时钟, 运行于 16 kHz 总线频率的 BLPI 模式 (参见前例) 转换到 4MHz 晶体频率、16 MHz 总线频率的 FEE 模式。示例中首先介绍了代码序列, 然后提供了一个演示该顺序的流程图。

1. 首先, BLPI 必须转换到 FBI 模式。
  - a) MCGC2 = 0x00 (%00000000)
    - MCGSC 中的 LP (位 3) 是 0
  - b) 循环检测, 直到 MCGSC 中的 LOCK (位 6) 置位, 表明 FLL 已经获得锁定。尽管在 FBI 模式中 FLL 被旁通, 但它仍使能并运行。
2. 接下来, FBI 将转换到 FEE 模式。
  - a) MCGC2 = 0x36 (%00110110)
    - RANGE (位 5) 设置为 1, 因为 4 MHz 频率在高频范围内。
    - HGO (位 4) 设置为 1, 为高增益运行配置外部振荡器。
    - EREFS (位 2) 设置为 1, 因为正在使用晶体。
    - ERCLKEN (位 1) 设置为 1, 确保外部参考时钟处于活动状态。
  - b) 循环检测, 直到 MCGSC 中的 OSCINIT (位 1) 是 1, 表明 EREFS 位选择的晶体已经完成初始化。
  - c) MCGC1 = 0x38 (%00111000)
    - CLKS (位 7 和 6) 设置为 %00, 以便将 FLL 输出选为系统时钟源。
    - RDIV (位 5-3) 设置为 %111 或除以 128, 因为  $4 \text{ MHz} / 128 = 31.25 \text{ kHz}$ , 这在 FLL 要求的  $31.25 \text{ kHz} \sim 39.0625 \text{ kHz}$  频率范围内。
    - IREFS (位 1) 清除至 0, 选择外部参考时钟
  - d) 循环检测, 直到 MCGSC 中的 IREFST (位 4) 是 0, 表明外部参考时钟是参考时钟的当前源。
  - e) 循环检, 直到 MCGSC 中的 LOCK (位 6) 置位, 表明 FLL 重新获得了锁定。
  - f) 循环检测, 直到 MCGSC 中的 CLKST (位 3 和 2) 是 %00, 表明已经选择 FLL 输出为 MCGOUT 馈电。

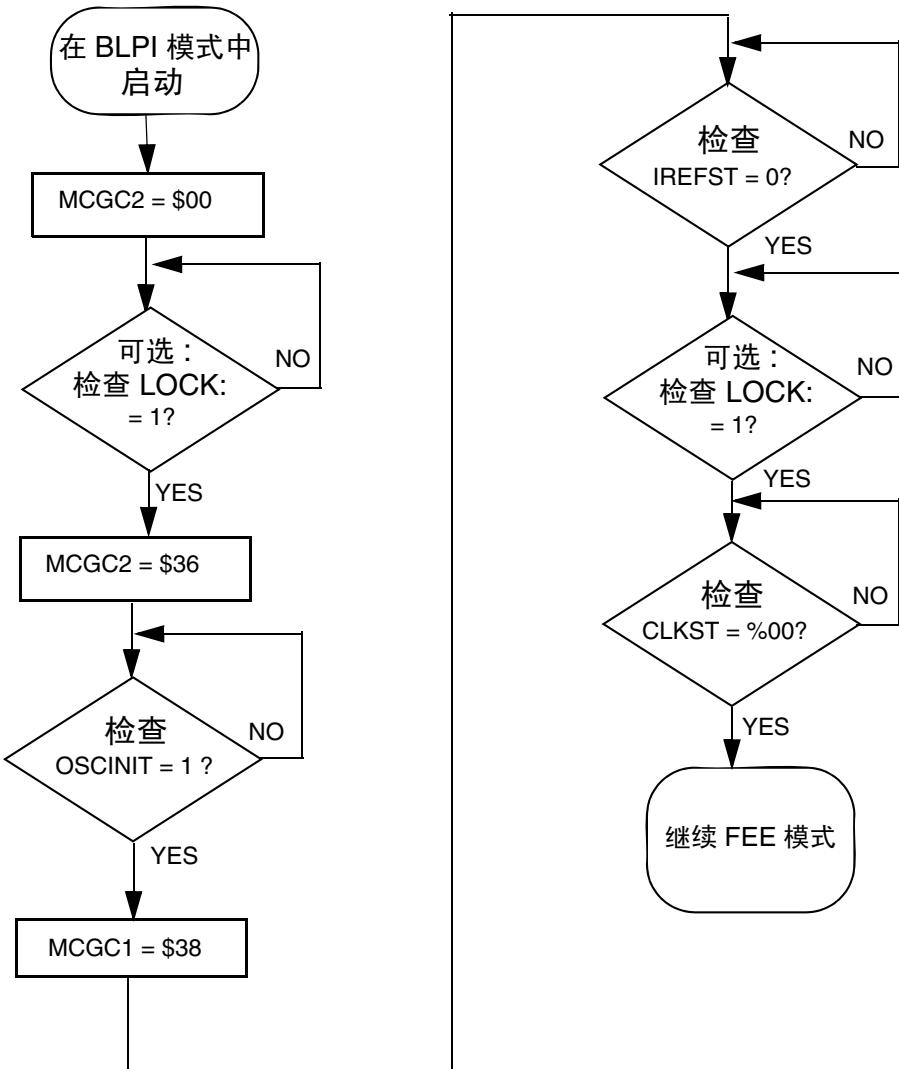


图 8-11. 使用 4 MHz 晶体从 BLPI 转换到 FEE 模式的流程图

#### 8.6.2.4 示例 4: 从 FEI 转换到 PEE 模式: 外部晶体 = 8 MHz、总线频率 = 8 MHz

本例中, MCG 将通过适当的运行模式从 FEI 转换到 PEE 模式, 直到设置 8 MHz 晶体参考频率来获得 8 MHz 的总线频率。

这个例子和第一个例子相似, 只是外部晶体频率从 4 MHz 变成了 8 MHz。在这个例子中必须特别注意, 因为从 FEI 模式转换到 PEE 模式的过程中, FLL 有时会基于高于 FLL 的最大允许频率的参考时钟运行。之所以出现这种情况, 是因为 8 MHz 的外部晶体和 128 的最大参考分频器因子, FLL 由此得出的参考时钟的频率是 62.5 kHz (大于 39.0625 kHz 这一最大允许值)。

当 FLL 在该条件下运行时, 在应用软件中应最大限度地减少在这一状态中的时间。

下列代码顺序描述了如何从 **FEI** 模式转换到 **PEE** 模式，直到设置 8 MHz 晶体参考频率来获得 8 MHz 总线频率。因为 MCG 复位后处于 **FEI** 模式，本例还显示了如何初始化 MCG，以实现在复位后进入 **PEE** 模式。示例中首先介绍了代码序列，然后提供了一个演示该顺序的流程图。

1. 首先，**FEI** 必须转换到 **FBE** 模式：

- a)  $\text{MCGC2} = 0x36$  (%00110110)
  - BDIV 位 7 和 6) 设置为 %00 或除以 1。
  - RANGE (位 5) 设置为 1，因为 8 MHz 的频率属于高频范围。
  - HGO (位 4) 设置为 1，为高增益运行配置外部振荡器。
  - EREFS (位 2) 设置为 1，因为正在使用晶体。
  - ERCLKEN (位 1) 设置为 1，确保外部参考时钟处于活动状态。
- b) 循环检测，直到 MCGSC 中的 OSCINIT (位 1) 是 1，表明 EREFS 位选择的晶体已经完成初始化。
- c) 禁止中断 (如果适用，在 CCR 中设置中断位)。
- d)  $\text{MCGC1} = 0xB8$  (%10111000)
  - CLKS (位 7 和 6) 设置为 %10，以便将外部参考时钟选择为系统时钟源。
  - RDIV (位 5-3) 设置为 %111 或除以 128。

#### 注意

$8 \text{ MHz} / 128 = 62.5 \text{ kHz}$ ，这大于 FLL 要求的 31.25 kHz -- 39.0625 kHz 频率范围。因此，当 **FBE** 的转换完成后，必须通过在软件中设置 MCGC2 中 LP 位，让 MCG 立即进入 **BLPE** 模式。

- IREFS 位 2) 清除至 0，选择外部参考时钟。
- e) 循环检测，直到 MCGSC 中的 IREFST (位 4) 是 0，表明外部参考是参考时钟的当前源。
- f) 循环检测，直到 MCGSC 中的 CLKST (位 3 和 2) 是 %10，表明已经选择外部参考时钟为 MCGOUT 馈电。
- 2. 然后，从 **FBE** 模式转换到 **BLPE** 模式：

- a)  $\text{MCGC2} = 0x3E$  (%00111110)
  - MCGC2 中的 LP (位 3) 为 1 (已进入 **BLPE** 模式)

#### 注意

在 1d 和 2a 步骤间没有额外步骤 (包括中断)。

- b) 使能中断 (如果适用，清除 CCR 中的中断位)
- c)  $\text{MCGC1} = 0x98$  (%10011000)
  - RDIV (位 5-3) 设置为 %011 或除以 8，因为  $8 \text{ MHz} / 8 = 1 \text{ MHz}$ ，这在 PLL 要求的 1MHz -- 2 MHz 频率范围内。在 **BLPE** 模式中，RDIV 的配置不重要，因为 FLL 和 PLL 都被禁止。更改它们只会为 PLL 建立在 PBE 模式中使用的分频器。
- d)  $\text{MCGC3} = 0x44$  (%01000100)
  - PLLS (位 6) 设置为 1，选择 PLL。在 **BLPE** 模式中，更改该位只会使 MCG 准备在 PBE 模式中的 PLL 使用。

- VDIV (位 3-0) 设置为 %0100 或乘以 16, 因为 1 MHz 参考 \* 16 = 16 MHz。在 BLPE 模式中, VDIV 位的配置不重要, 因为 PLL 被禁止。更改它们只建立 PBE 模式中的 PLL 使用乘积因子。
  - e) 循环检测, 直到 MCGSC 中的 PLLST (位 5) 置位, 表明 PLLS 时钟的当前源是 PLL。
3. 然后, 从 BLPE 模式转换到 PBE 模式:
    - a) 将 MCGC2 中的 LP (位 3) 清除至 0, 切换到 PBE 模式。
    - b) 循环检测, 直到 MCGSC 中的 LOCK (位 6) 置位, 表明 PLL 已经获得锁定。
  4. 最后, 从 PBE 模式转换到 PEE 模式:
    - a) MCGC1 = 0x18 (%00011000)
    - MCGSC1 中的 CLKS (位 7 和 6) 设置为 %00, 以便将 PLL 输出选择为系统时钟源。
    - b) 循环检测, 直到 MCGSC 中的 CLKST (位 3 和 2) 是 %11, 表明已经选择 PLL 输出, 在当前时钟模式中为 MCGOUT 馈电。
    - 这样, RDIV 除以 8、BDIV 除以 1、VDIV 乘以 16,  $MCGOUT = [(8 \text{ MHz} / 8) * 16] / 1 = 16 \text{ MHz}$ , 总线频率是  $MCGOUT / 2$  或 8 MHz。

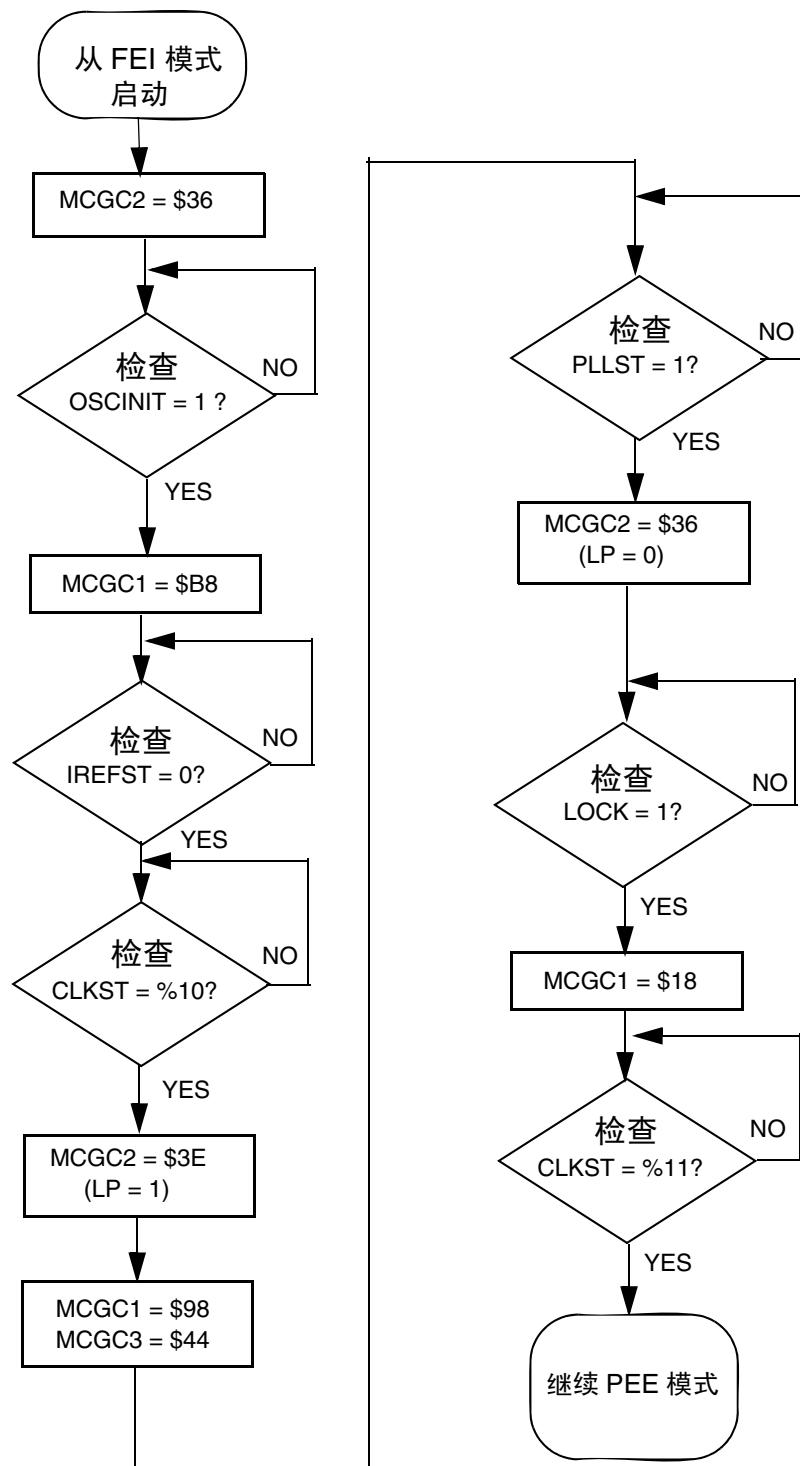


图 8-12. 使用 8 MHz 晶体从 FEI 转换到 PEE 模式的流程图

### 8.6.3 校准内部参考时钟 (IRC)

IRC 校准首先通过写入 MCGTRM 寄存器，然后使用 FTRIM 位“微调”频率。我们把这个全部 9 位值作为调整值，范围从 0x000 到 0x1FF，其中，FTRIM 位是 LSB。

POR 后的调整值通常是 0x100（MCGTRM = 0x80 和 FTRIM = 0）。写入更大的值会降低频率；写入更小的值会增加频率。调整值与时段呈线性关系，但晶圆加工的微小差异也会导致调整值和时段间存在轻微的非线性。这些非线性就是我们推荐使用迭代修正法来实现最佳调整值的原因。该方法在本节后面的示例 4 中有相关介绍。

当已经为器件找到调整值后，该值就保存在闪存里。如果该器件断电，只要将保存的值从闪存复制到 MCG 寄存器上，就能轻松地重新调整 IRC。飞思卡尔在每个 MCU 上注明了保存调整值的建议闪存位置。参见本产品说明的存储器映射，了解它们的具体位置。对于出厂时已调整的器件，出厂调整值将保存在这些位置中。

#### 8.6.3.1 示例 5：内部参考时钟调整

对于那些需要严格频率容限的应用而言，我们提供了一个调整流程，它将实现一个非常精确的内部时钟源。本节用一个示例概括地描述了内部振荡器修正。众多其他调整流程也非常有效，可以使用。

在下面的例子里，将为 9 位 MCGTRM 和 FTRIM 的合计值校准 MCG 调整。这个值被称为 TRMVAL。

初始条件:

- 1) ATE 供应的时钟的占空比为  $500 \mu s$ 。
- 2) 配置 MCG 使用内部参考源，并且总线时钟达到 8MHz。

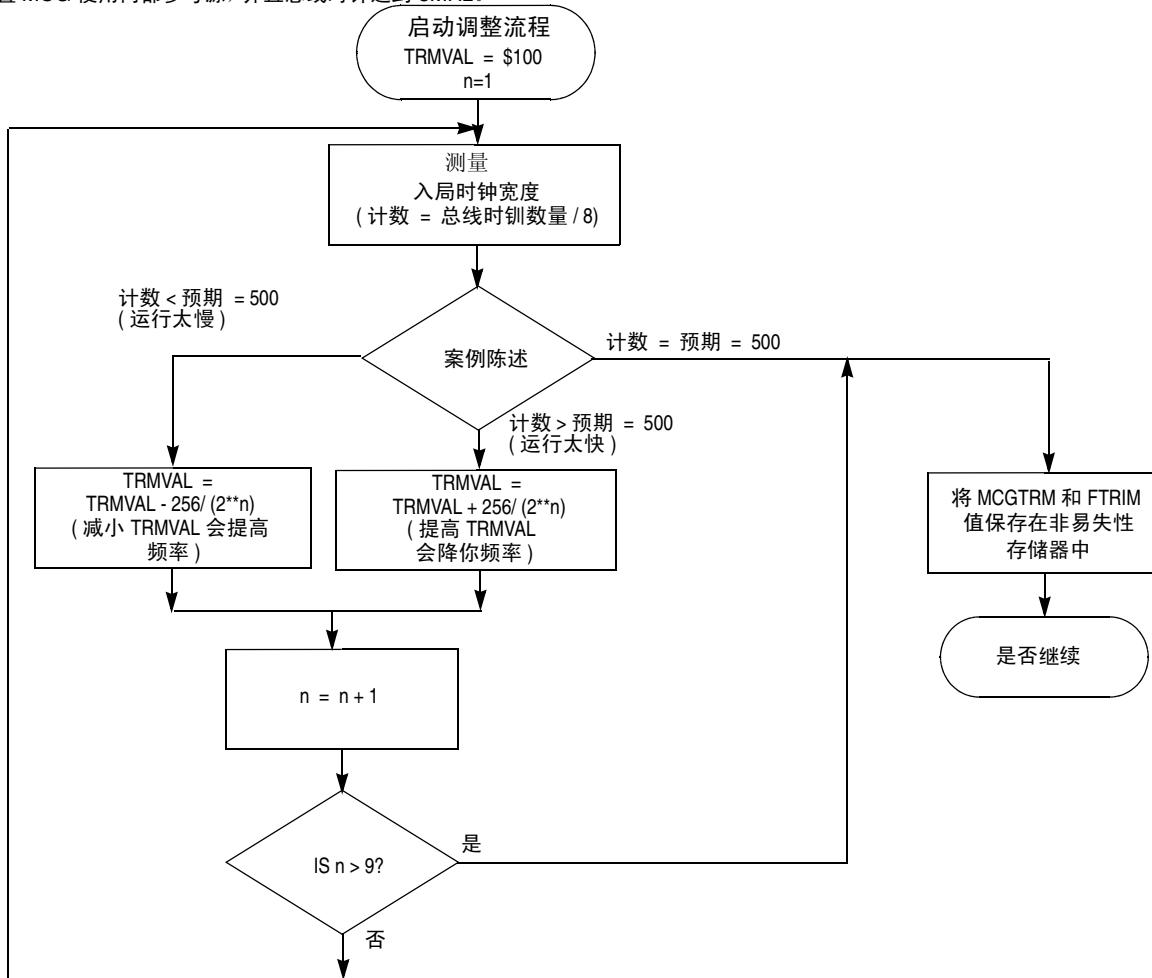


图 8-13. 修正步骤

在这个特例中，MCU 已经焊接到 PCB 上，且整个 PCB 焊接已经结束，正在用自动测试设备进行最后测试。当运行在用户提供的软件控制下时，会为 MCU 提供单独的信号或消息。MCU 发起 图 8-13 所示的调整流程，同时测试设备会提供精确的参考信号。

如果想要的总线频率接近器件允许的最大值，建议使用两倍于最终值的参考分频器值 (RDIV 设置) 进行调整。调整流程完成后，可以恢复参考分频器。这样就能防止意外超越最大时钟频率。



# 第 9 章

## 模拟比较器 (S08ACMPV3)

### 9.1 介绍

模拟比较器模块 (ACMP) 提供用来比较两个模拟输入电压，或者一个输入电压和一个内部参考电压的电路。比较器电路能够在整个电源电压范围内操作（轨到轨操作）。

MC9S08DZ60 系列的所有 MCU 都能在 64 管脚的封装中提供两个全功能 ACMP。48 管脚封装的 MCU 有两个 ACMP，但 ACMP2 的输出管脚没有引出。32 管脚封装的 MCU 只有一个全功能 ACMP。

#### NOTE

MC9S08DZ60 系列器件的工作电压范围较高 (2.7 V --5.5 V)，不支持 STOP1 模式。请忽略 STOP1 的参考。

#### 9.1.1 ACMP 配置报文

当使用带死区参考电压为 ACMP+ 输入时，用户必须通过在 SPMSC1 中设置 BGBE =1，使能死区缓冲，详细内容 [5.8.7，“系统电源管理状态和控制寄存器 1 \(SPMSC1\)”](#)。如需了解死区电压参考报文 [A.6，“DC 特性”](#)。

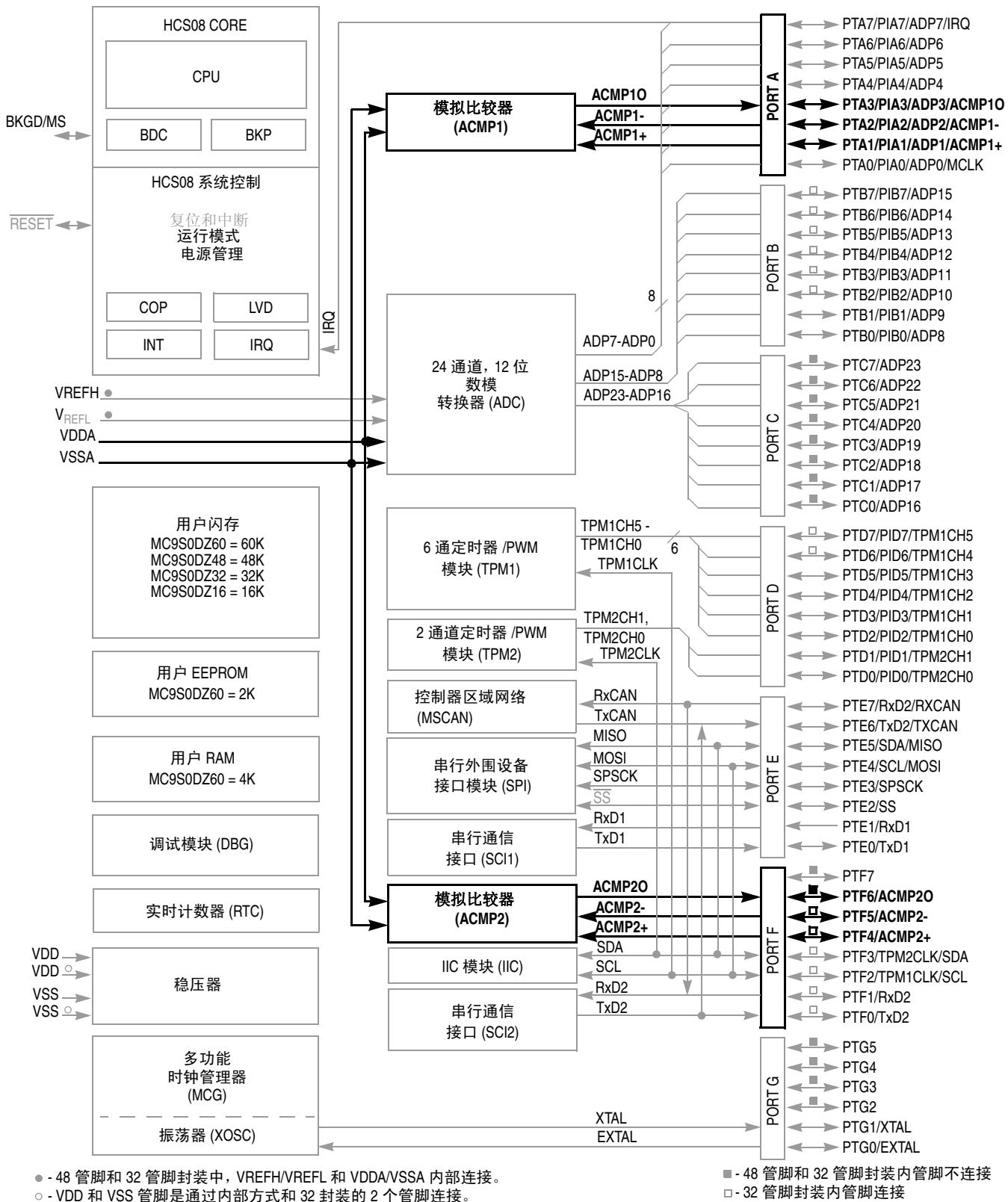


图 9-1. MC9S08DZ60 结构图

## 9.1.2 特性

ACMP 具有以下特性：

- 完全的轨到轨供电操作。
- 可选择的比较器输出上升沿中断、下降沿中断，或上升沿及下降沿中断。
- 与内部固定的带死区参考电压比较选项。
- 允许在管脚 ACMPxO 上看到比较器结果输出的选项。

## 9.1.3 运行模式

本节介绍等待、停止和背景调试模式中的 ACMP 运行。

### 9.1.3.1 等待模式中的 ACMP

如果在进入等待模式前已经使能 ACMP，ACMP 将继续在等待模式中运行。因此，如果使能 ACMP 中断（ACIE 已设置），可以用 ACMP 使 MCU 退出等待模式。为了实现尽可能低的功耗，如果等待模式中不需要 ACMP 作为中断源，应通过软件关闭 ACMP。

### 9.1.3.2 停止模式中的 ACMP

ACMP 在所有停止模式中都被禁止，无论执行停止指令前的设置如何。因此，ACMP 不能作为停止模式的唤醒源。

在 STOP2 模式中，ACMP 模块的电源完全关闭。当从 STOP2 模式中唤醒时，ACMP 模块处于复位状态。

在 STOP3 模式中，ACMP 模块的时钟暂停。寄存器不受影响。此外，ACMP 比较器电路进入低功耗状态。STOP3 中不会发生比较操作。

如果 STOP3 由于复位而退出，ACMP 将进入复位状态。如果 STOP3 由于中断而退出，ACMP 将从进入 STOP3 时的状态继续运行。

### 9.1.3.3 使能背景调试模式中的 ACMP

当微控制器处于使能背景调试模式时，ACMP 继续正常运行。

### 9.1.4 结构图

模拟比较器模块的结构图如图 9-2。

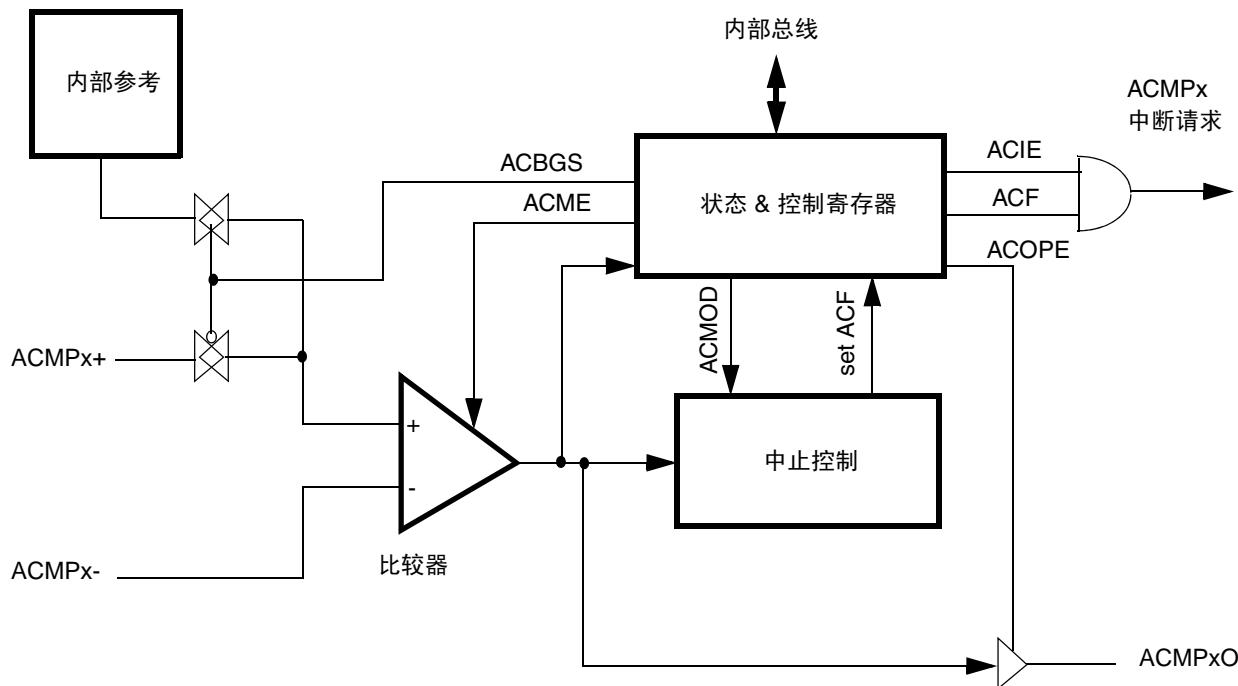


图 9-2. 模拟比较器 (ACMP) 结构图

### 9.2 外部信号描述

ACMP 有两个模拟输入管脚  $\text{ACMPx+}$  和  $\text{ACMPx-}$ ，有一个数字输出管脚  $\text{ACMPxO}$ 。输入管脚可接受的电压范围在 MCU 正常工作电压范围内。如图 9-2 所示， $\text{ACMPx-}$  管脚连接比较器的反相输入，如果  $\text{ACBGS}$  是 0， $\text{ACMPx+}$  管脚连接比较器的同相输入。如图 9-2 所示，可以使能  $\text{ACMPxO}$  管脚来驱动外部输出。

ACM 的信号属性如表 9-1 所示。

表 9-1. 信号属性

信号	功能	I/O
$\text{ACMPx-}$	ACMP 的反相模拟输入（负输入）	I
$\text{ACMPx+}$	ACMP 的同相模拟输入（正输入）	I
$\text{ACMPxO}$	ACMP 的数字输出	O

## 9.3 存储器映射 / 寄存器定义

ACMP 包括一个寄存器：

- 一个 8 位状态和控制寄存器

如需了解 ACMP 寄存器的绝对地址分配，请参见本文档的存储器节“直接页面寄存器概述”。本节仅按寄存器和控制位的名称及相关地址偏移进行参考。

有些 MCU 的 ACMP 可能不止一个，因此寄存器名称包括占位符 (x)，以明确正在参考哪个 ACMP。

表 9-2. ACMP 寄存器摘要

名称		7	6	5	4	3	2	1	0
ACMPxSC	R	ACME	ACBGS	ACF	ACIE	ACO	ACCOPE	ACMOD	
	W								

### 9.3.1 ACMPx 状态和控制寄存器 (ACMPxSC)

ACMPxSC 包括状态标记和使能及配置 ACMP 所需的控制位。

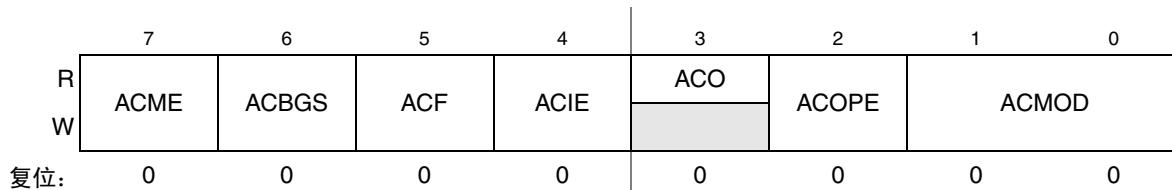


图 9-3. ACMPx 状态和控制寄存器 (ACMPxSC)

表 9-3. ACMPxSC 字段描述

字段	描述
7 ACME	模拟比较器模块使能。使能 ACMP 模块。 0 ACMP 关闭 1 ACMP 使能
6 ACBGS	模拟比较器死区选择。选择带死区参考电压或 ACMPx+ 管脚作为模拟比较器同相输入的输入。 0 外部管脚 ACMPx+ 选择为比较器的同相输入 1 内部参考选择为比较器的同相输入
5 ACF	模拟比较器标记。每次发生比较事件时都设置 ACF。比较事件由 ACMOD 定义。通过在 ACF 上写入 1 来清除 ACF。 0 未发生比较事件。 1 已发生比较事件。
4 ACIE	模拟比较器中断使能。从 ACMP 那里使能中断。设置了 ACIE 后，在 ACF 置位时中断被触发。 0 中断禁止 1 中断使能
3 ACO	模拟比较器输出。ACO 读数返回模拟比较器输出的当前值。ACO 复位为 0，在 ACMP 禁止时 (ACME = 0) 读数为 0。

表 9-3. ACMPxSC 字段描述

字段	描述
2 ACOPE	模拟比较器输出管脚使能。将使能在外部管脚 ACMPxO 上的比较器输出。 0 模拟比较器输出在 ACMPxO 上不可用 1 模拟比较器输出被驱动到 ACMPxO 上
1:0 ACMOD	模拟比较器模式。ACMOD 择置位 ACF 的比较器事件类型。 00 Encoding 0 — 比较器输出下降沿 01 Encoding 1 — 比较器输出上升沿 10 Encoding 2 — 比较器输出下降沿 11 Encoding 3 — 比较器输出上升或下降沿 9.4 功能描述

## 9.4 功能描述

模拟比较器可以比较 ACMPx+ 和 ACMPx- 管脚上的模拟输入电压，或者它也可以将 ACMPx- 管脚上的模拟输入电压和内部带死区参考电压比较。ACBGS 选择带死区参考电压或 ACMPx+ 管脚为模拟比较器同相输入的输入。当同相输入大于反相输入时，比较器输出为高电平；当同相输入小于反相输入时，比较器输出为低电平。ACMOD 选择置位 ACF 的条件。ACF 可以在比较器输出的上升沿、比较器输出的下降沿、或上升及下降沿（切换）时置位。比较器输出可以通过 ACO 直接读取。使用 ACOPE，比较器输出可以直接驱动到 ACMPxO 管脚上。。

# 第 10 章

## 数模转换器 (S08ADC12V1)

### 10.1 介绍

12 位数模转换器 (ADC) 是逐次逼近式 ADC，为集成微控制器片上系统设计。

#### 注意

MC9S08DZ60 系列器件的工作电压 (2.7 V -5.5 V) 较高，不支持 STOP1 模式。请忽略 STOP1 参考。

#### 10.1.1 模拟功率和接地信号名称

本章的  $V_{DDAD}$  和  $V_{SSAD}$  参考分别对应  $V_{DDA}$  和  $V_{SSA}$  信号。

#### 10.1.2 信道分配

#### 注意

MC9S08DZ60 系列器件的 ADC 通道分配如表 10-1 所示。保留通道转换结果未知。

本章介绍了所有 S08ADC12V1 通道位。MC9S08DZ60 系列 MCU 并不全部使用所有这些通道。器件上不可用通道对应的所有位都被保留。

表 10-1. ADC 通道分配

ADCH	通道	输入
00000	AD0	PTA0/ADP0/MCLK
00001	AD1	PTA1/ADP1/ACMP1+
00010	AD2	PTA2/ADP2/ACMP1P-
00011	AD3	PTA3/ADP3/ACMP1O
00100	AD4	PTA4/ADP4
00101	AD5	PTA5/ADP5
00110	AD6	PTA6/ADP6
00111	AD7	PTA7/ADP7
01000	AD8	PTB0/ADP8
01001	AD9	PTB1/ADP9
01010	AD10	PTB2/ADP10
01011	AD11	PTB3/ADP11
01100	AD12	PTB4/ADP12
01101	AD13	PTB5/ADP13
01110	AD14	PTB6/ADP14

ADCH	通道	输入
01111	AD15	PTB7/ADP15
10000	AD16	PTC0/ADP16
10001	AD17	PTC1/ADP17
10010	AD18	PTC2/ADP18
10011	AD19	PTC3/ADP19
10100	AD20	PTC4/ADP20
10101	AD21	PTC5/ADP21
10110	AD22	PTC6/ADP22
10111	AD23	PTC7/ADP23
11000–	AD24 through AD25	预留
11001		
11010	AD26	温度传感器 <sup>1</sup>
11011	AD27	内部隙带 <sup>2</sup>
11100	预留	预留
11101	V <sub>REFH</sub>	V <sub>REFH</sub>
11110	V <sub>REFL</sub>	V <sub>REFL</sub>

### 10.1.3 替代时钟

ADC 模块的时钟源可以是 MCU 总线时钟，总线时钟二分频，模块内的本地异步时钟 (ADACK) 或替代时钟 ALTCLK。MC9S08DZ60 系列 MCU 器件的替代时钟是外部参考时钟 (MGERCLK)。

所选的时钟源必须运行在一定的频率范围内，这样 ADC 转换时钟 (ADCK) 就可以通过 ADIV 位的设置，在从 ALTCLK 分频后，运行在指定的频率范围 ( $f_{ADCK}$ ) 内。

当 MCU 处于等待模式时，ALTCLK 是使能的 (满足以上条件)。这使得当 MCU 处于等待模式时，ALTCLK 可以用作 ADC 的工作时钟源。

当 MCU 处于 STOP2 或 STOP3 时，ALTCLK 不能用作 ADC 工作时钟源。

### 10.1.4 硬件触发

ADC 硬件触发（即 ADHWT）是来自实时时钟（RTC）的输出。RTC 计数器可以用 MGERCLK 为时钟源，也可以用 1 kHz 时钟源计时。

RTC 的周期由输入时钟频率、RTCPMS 位和 RTCMOD 寄存器决定。当 ADC 硬件触发使能时，RTC 计数器溢出触发 ADC 转换。

RTC 可以在经过配置后，在 MCU 运行、等待和 STOP3 模式中引发硬件触发。

### 10.1.5 温度传感器

要使用片上温度传感器，用户必须实施以下操作：

- 设置 ADC 为长采样模式，最高 1MHz 时钟
- 采集带死区参考电压通道（AD27）
  - 通过采集带死区参考电压通道的电压 VBG，用户可以确定 VDD。有关带死区参考电压值的报文，请 [A.6，“DC 特性”](#)。
- 转换温度传感器通道（AD26）
  - 使用计算得到的 VDD，将 AD26 的数字量转换成电压  $V_{TEMP}$

[等式 10-1](#) 提供了一个温度传感器的近似传递函数。

$$\text{Temp} = 25 - ((V_{TEMP} - V_{TEMP25}) \div m)$$

[等式 10-1](#)

其中：

- $V_{TEMP}$  是在当前环境温度下温度传感器通道的电压。
- $V_{TEMP25}$  是 25 °C 时的温度传感器通道的电压。
- 是以 V/ °C 表示的热 / 冷电压与温度斜率比较值

如需进行温度计算，请使用 **ADC Electricals** 表中的  $V_{TEMP25}$  和  $m$  值。

在应代码中，用户读取温度传感器通道、计算  $V_{TEMP}$  值并与  $V_{TEMP25}$  进行比较。如果  $V_{TEMP}$  大于  $V_{TEMP25}$ ，冷端斜率值就应用于 [等式 10-1](#)。如果  $V_{TEMP}$  小于  $V_{TEMP25}$ ，热端斜率值就应用于 [等式 10-1](#)。为了提高准确性，用户应标定死区参考电压和温度传感器。

在 25 °C 上标定将精度提高到 ± 4.5 °C。

-40 °C、25 °C 和 125 °C 三个点上的标定能够将精度提高到 ± 2.5 °C。一旦完成了标定，用户就需要计算热端斜率和冷端斜率。在应用代码中，用户使用 [等式 10-1](#) 计算温度，然后决定温度是高于还是低于 25 °C。一旦确定了温度，用户就能够使用标定期间获得的热 / 冷斜率，重新计算温度。

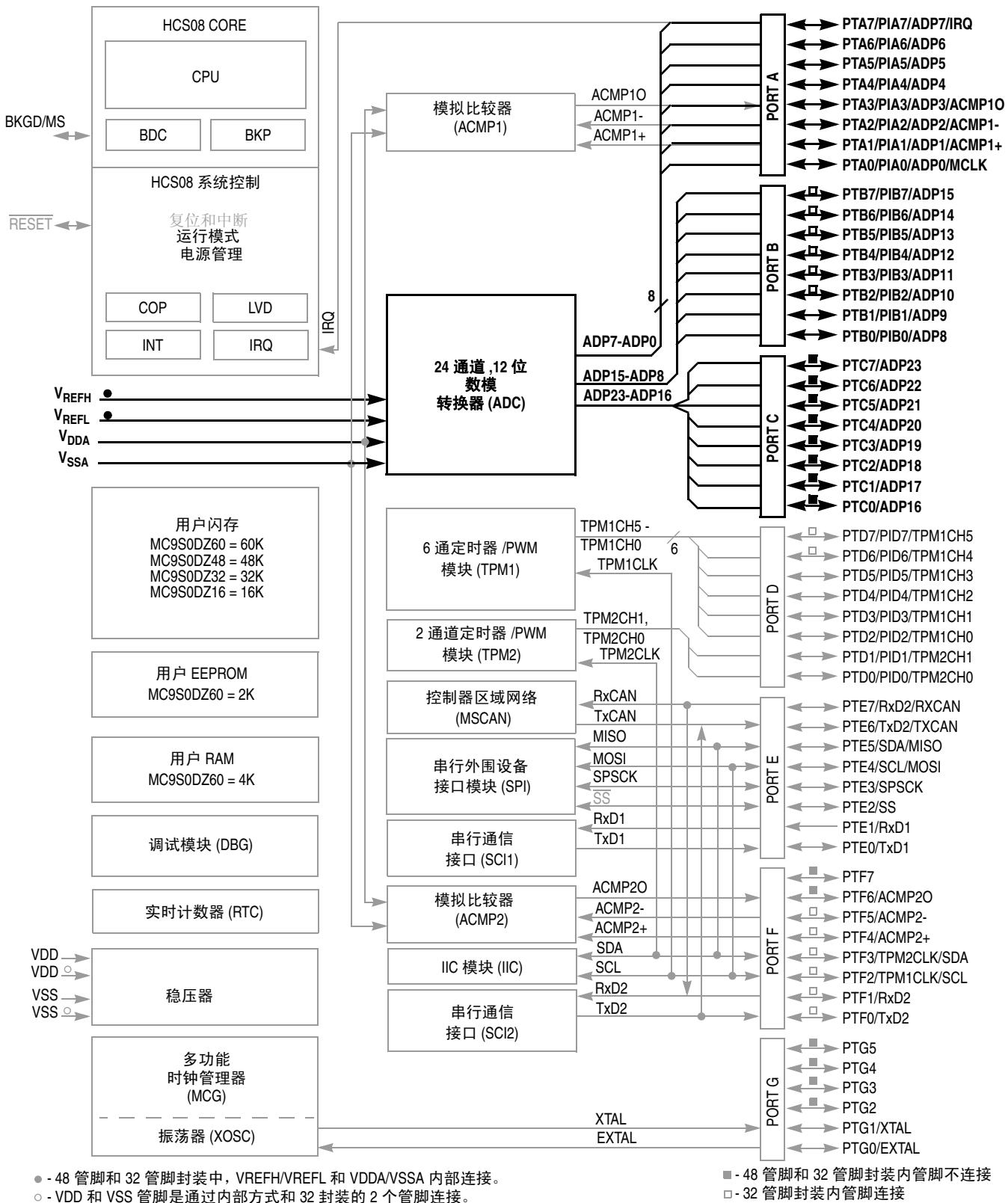


图 10-1. 强调 ADC 模块和管脚的 MC9S08DZ60 结构图

### 10.2.6 特性

ADC 模块的特性包括：

- 具有 12 位分辨率的线性逐次逼近算法；
- 高达 28 个模拟输入；
- 12、10 或或 8 位右对齐输出格式；
- 单次转换或连续转换（单转换后自动返回空闲状态）；
- 采样时间和转换速度 / 功率可配置；
- 转换完成标志和中断；
- 最多可选择 4 个输入时钟源；
- 在等待或 STOP3 模式下实现了低噪音运行；
- 异步时钟源实现了低噪音运行；
- 可选的异步硬件转换触发；
- 与小于、大于或等于可编程值自动比较的中断；

### 10.2.7 结构图

图 10-2 是 ADC 模块的结构图

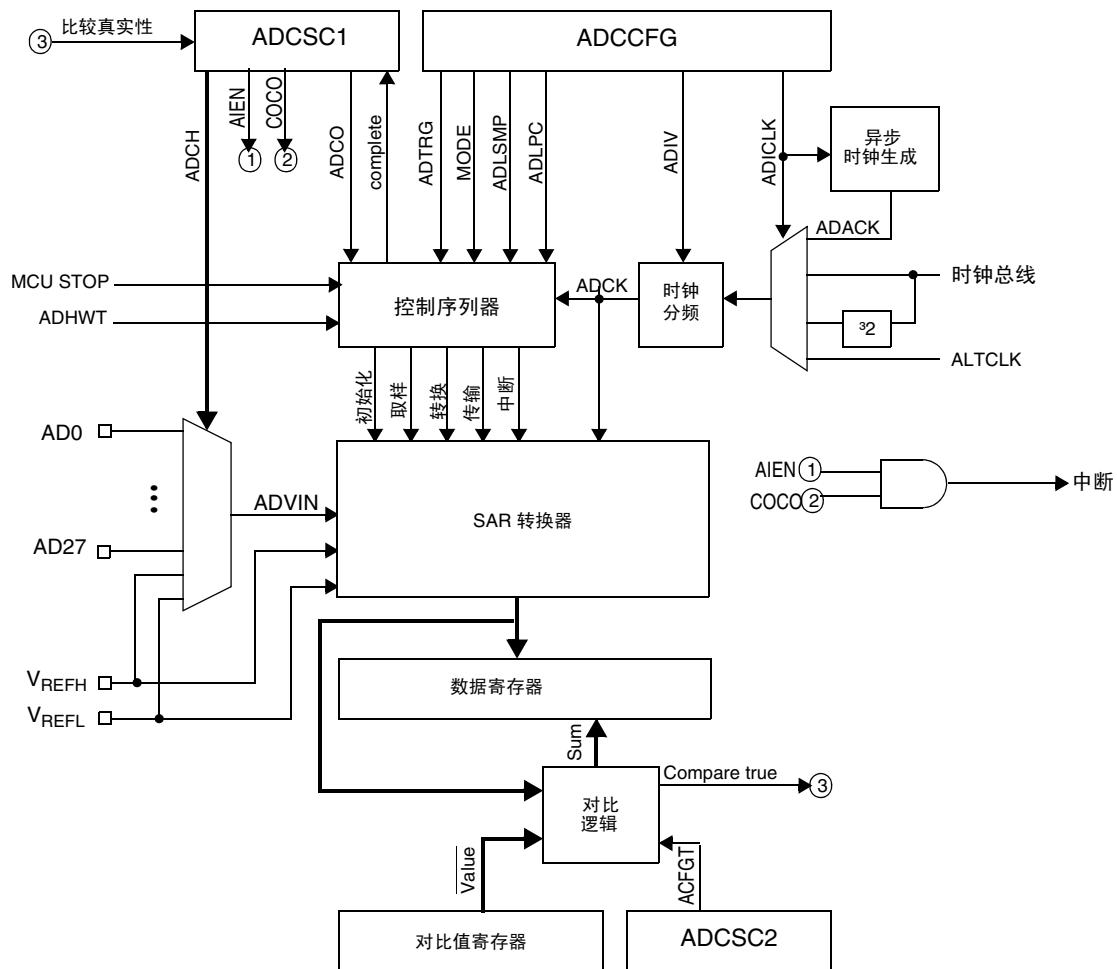


图 10-2. ADC 结构图

## 10.3 外部信号描述

ADC 模块最多可支持 28 个独立模拟输入。它还需要 4 个电源 / 参考 / 接地连接。

表 10-2. 信号属性

名称	功能
AD27-AD0	模拟通道输入
$V_{REFH}$	高参考电压
$V_{REFL}$	低参考电压
$V_{DDAD}$	模拟电源
$V_{SSAD}$	模拟接地

### 10.3.1 模拟电源 ( $V_{DDAD}$ )

ADC 模拟部分使用  $V_{DDAD}$  作为其电源连接。在有些封装中， $V_{DDAD}$  与  $V_{DD}$  是内部连接。如果是外部连接，将  $V_{DDAD}$  管脚连到与  $V_{DD}$  相同的电平。为了确保干净的  $V_{DDAD}$  信号，可能还需要外部滤波。

### 10.3.2 模拟接地 ( $V_{SSAD}$ )

ADC 模拟部分使用  $V_{SSAD}$  作为其接地连接。在有些封装中， $V_{SSAD}$  与  $V_{SS}$  是内部连接。如果是外部连接，将  $V_{SSAD}$  管脚连到与  $V_{SS}$  相同的电平。

### 10.3.3 参考电压高 ( $V_{REFH}$ )

$V_{REFH}$  是转换器的高参考电压。在有些封装中， $V_{REFH}$  与  $V_{DDAD}$  是内部连接。如果是外部连接， $V_{REFH}$  可以连接到与  $V_{DDAD}$  相同的电平，或者由介于  $V_{DDAD}$  最低限值和  $V_{DDAD}$  电平之间的外部源驱动 ( $V_{REFH}$  必须不能超过  $V_{DDAD}$ )。

### 10.3.4 参考电压低 ( $V_{REFL}$ )

$V_{REFL}$  是转换器的低参考电压。在有些封装中， $V_{REFL}$  与  $V_{SSAD}$  是内部连接。如果是外部连接，将  $V_{REFL}$  管脚连到和  $V_{SSAD}$  相同的电平。

### 10.3.5 模拟通道输入 (ADx)

ADC 模块最多可支持 28 个独立的模拟输入。通过 ADCH 通道选择位选择转换。

## 10.4 寄存器定义

以下这些存储器映射寄存器控制和显示 ADC 的运行状态：

- 状态和控制寄存器, ADCSC1
- 状态和控制寄存器, ADCSC2
- 数据结果寄存器, ADCRH 和 ADCRL
- 比较值寄存器, ADCCVH 和 ADCCVL
- 配置寄存器, ADCCFG
- 管脚使能寄存器, APCTL1、APCTL2 和 APCTL3

### 10.4.1 状态和控制寄存器 1 (ADCSC1)

本节介绍 ADC 状态和控制寄存器 (ADCSC1) 的功能。写入 ADCSC1 会中断当前转换，并发起一个新转换（如果 ADCH 位等于全 1 以外的值）。

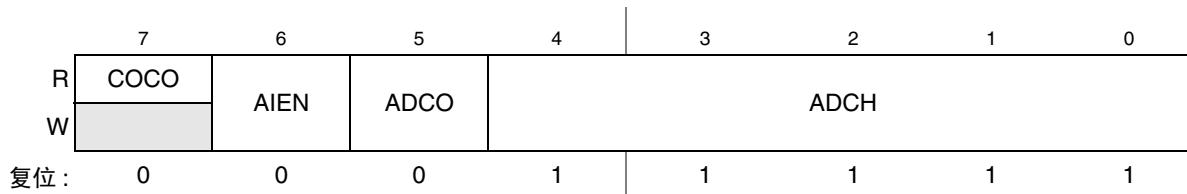


图 10-3. 状态和控制寄存器 (ADCSC1)

表 10-3. ADCSC1 寄存器字段描述

字段	描述
7 COCO	<b>转换完成标志</b> — COCO 标记是只读位，在每次转换完成时置位，这时比较功能被禁止 (ACFE = 0)。当比较功能使能 (ACFE = 1) 时，只有当比较结果为真时才在转换完成时设置 COCO 标记。每次当写入 ADCSC1 或读取 ADCRL 时，该位就被清除。 0 转换未完成 1 转换完成
6 AIEN	<b>中断使能</b> — AIEN 用来使能转换完成中断。当 COCO 已置位且 AIEN 为 1 时，中断将被触发。 0 转换完成中断禁止 1 转换完成中断使能
5 ADCO	<b>连续转换使能</b> — ADCO 用来使能连续转换。 0 当选择软件触发时写入 ADCSC1，进行一次转换，或者当选择硬件触发时 ADHWT 触发后，进行一次转换。 1 当选择软件触发时写入 ADCSC1，开始连续转换，或者当选择硬件触发时 ADHWT 触发后，开始连续转换。
4:0 ADCH	<b>输入通道选择</b> — ADCH 位形成一个 5 位字段，该字段用来选择其中的一个输入通道。图 10-5 详细地描述了输入通道。 通道选择位全部设置为 1 时，逐次逼近转换器子系统关闭。该功能能够彻底禁止 ADC，隔离所有输入通道。用这种方式终止连续转换，可以防止多余的的一次单次转换。当禁止连续转换时，就不需要通过把通道选择位都设置为 1 的方式使 ADC 处于低功耗状态，因为当转换完成时模块会自动进入低功耗状态。

图 10-4. 输入通道选择

ADCH	输入选择
00000–01111	AD0–15
10000–11011	AD16–27
11100	预留
11101	$V_{REFH}$
11110	$V_{REFL}$
11111	模块禁止

### 10.4.2 状态和控制寄存器 2 (ADCSC2)

ADCSC2 寄存器用来控制 ADC 模块的比较功能、转换触发和转换状态。

	7	6	5	4		3	2	1	0
复位：	0	0	0	0		0	0	0	0

图 10-5. 状态和控制寄存器 2 (ADCSC2)

表 10-4. ADCSC2 寄存器字段描述

字段	描述
7 ADACT	<b>转换状态</b> — ADACT 显示正在进行一个转换。当开始转换时就设置 ADACT，当转换完成或中止时就清除 ADACT。 0 转换未进行 1 转换正在进行
6 ADTRG	<b>转换触发选择</b> — ADTRG 用来选择转换的触发类型。 有 2 种触发可供选择：软件触发和硬件触发。当选择软件触发时，写入 ADCSC1 就能发起转换。当选择硬件触发时，ADHWT 触发后就能发起转换。 0 选择软件触发 1 选择硬件触发
5 ACFE	<b>比较功能使能</b> — ACFE 用来使能比较功能。 0 比较功能禁止 1 比较功能使能
4 ACFGT	<b>比较功能大于使能</b> — ACFGTE 用来配置使当前的转换结果大于或等于比较值时触发，默认的比较功能为在当前的转换结果小于比较值时触发。 0 当输入小于比较值时触发 1 当输入大于或等于比较值时触发

### 10.4.3 数据结果高地址寄存器 (ADCRH)

	7	6	5	4		3	2	1	0
R	0	0	0	0	ADR11	ADR10	ADR9	ADR8	
W									
复位：	0	0	0	0	0	0	0	0	0

图 10-6. 数据结果高地址寄存器 (ADCRH)

在 10 位模式中，ADCRH 包含 10 位转换结果中的高 2 位。在配置 10 位模式时，ADR11 - ADR10 等于 0。当为 8 位模式进行配置时，ADR11 - ADR8 等于 0。

在 12 位和 10 位模式中，每次完成转换时 ADCRH 都被更新，除非使能了自动比较但不满足比较条件。在 12 位和 10 位模式中，读取 ADCRH 能够防止在读取 ADCRL 之前 ADC 将后续转换结果传输到结果寄存器。如果 ADCRL 是在下一次转换完成后才被读取，那么中间转换结果会被丢失。在 8 位模式中，没有与 ADCRL 的互锁。

当 MODE 位被更改时，ADCRH 中的任何数据都将无效。数据结果低地址寄存器 (ADCRL)

ADCRL 包含 12 位或 10 位转换结果中的低 8 位。每次转换完成时，8 位转换寄存器的 8 个位都被更新，除非使能了自动比较但不满足比较条件。在 12 位和 10 位模式中，读取 ADCRH 能够防止在读取 ADCRL 之前 ADC 将后续转换结果传输到结果寄存器。如果 ADCRL 是在下一次转换完成后才被读取，那么中间转换结果会被丢失。在 8 位模式中，没有与 ADCRH 的互锁。当 MODE 位被更改时，ADCRL 中的任何数据都将无效。

	7	6	5	4		3	2	1	0
R	ADR7	ADR6	ADR5	ADR4	ADR3	ADR2	ADR1	ADR0	
W									
复位：	0	0	0	0	0	0	0	0	0

图 10-7. 数据结果低地址寄存器 (ADCRL)

### 10.4.4 比较值高地址寄存器 (ADCCVH)

在 12 位模式中，ADCCVH 寄存器包含 12 位比较值的高 4 位。当比较功能使能时，这些位与 12 位模式转换结果的高 4 位进行比较。

	7	6	5	4		3	2	1	0
R	0	0	0	0	ADCV11	ADCV10	ADCV9	ADCV8	
W									
复位：	0	0	0	0	0	0	0	0	0

图 10-8. 比较值高地址寄存器 (ADCCVH)

在 10 位模式中，ADCCVH 寄存器包含 10 位比较值 (ADCV9 - ADCV8) 的高 2 位。当比较功能使能时，这些位就与 10 位模式转换结果的高 2 位进行比较。

在 8 位模式中，比较过程中不使用 ADCCVH。

#### 10.4.5 比较值低地址寄存器 (ADCCVL)

寄存器包含 12 位或 10 位比较值的低 8 位或者 8 位比较值的所有 8 位。位 ADCV7:ADCV0 与 12 位、10 位或 8 位模式转换结果的低 8 位进行比较。

	7	6	5	4		3	2	1	0
R W	ADCV7	ADCV6	ADCV5	ADCV4	ADCV3	ADCV2	ADCV1	ADCV0	
复位：	0	0	0	0	0	0	0	0	0

图 10-9. 比较值低地址寄存器 (ADCCVL)

#### 10.4.6

#### 10.4.7 配置寄存器 (ADCCFG)

ADCCFG 用来选择运行模式、时钟源、时钟分频、低功率或长采样时间配置。

	7	6	5	4		3	2	1	0
R W	ADLPC	ADIV	ADLSMP		MODE		ADICLK		
复位：	0	0	0	0	0	0	0	0	0

图 10-10. 配置寄存器 (ADCCFG)

表 10-5. ADCCFG 寄存器字段描述

字段	描述
7 ADLPC	<b>低功率配置</b> — ADLPC 控制逐次逼近转换器的速度和功率配置。当不需要更高采样率时，ADLPC 可以用来优化功耗。 0 高速配置 1 低功率配置：{FC31} 功率的降低以最大时钟速度为代价。
6:5 ADIV	<b>时钟分频选择</b> — ADIV 选择 ADC 生成内部时钟 ADCK 所使用的分频率。 <a href="#">表 10-6</a> 显示了可用时钟配置。
4 ADLSMP	<b>长采样时间配置</b> — ADLSMP 选择长采样时间和短采样时间。这样可以调节采样时间，实现对高阻抗输入的精确采样，或者最大限度地提高对低阻抗输入的转换速度。如果在连续转换模式下不需要高转换速率，更长的采样时间还可以用来降低整体功耗。 0 短采样时间 1 长采样时间
3:2 模式	<b>转换模式选择</b> — MODE 位用来选择是 12、10 或 8 位转换。请参见 <a href="#">表 10-7</a> 。
1:0 ADICLK	<b>输入时钟选择</b> — ADICLK 位选择生成内部时钟 ADCK 的输入时钟源。请参见 <a href="#">表 10-8</a> 。

表 10-6. 时钟分频选择

ADIV	分频率	时钟率
00	1	输入时钟
01	2	输入时钟 $\div 2$
10	4	输入时钟 $\div 4$
11	8	输入时钟 $\div 8$

表 10-7. 转换模式

模式	模式描述
00	8 位转换 ( $N=8$ )
01	12 位转换 ( $N=12$ )
10	10 位转换 ( $N=10$ )
11	保留

表 10-8. 输入时钟选择

ADICLK	所选的时钟源
00	总线时钟
01	总线时钟除以 2
10	替代时钟 (ALTCLK)
11	异步时钟 (ADACK)

#### 10.4.8 管脚控制寄存器 1 (APCTL1)

管脚控制寄存器用来禁止对模拟输入的 MCU 管脚作为 I/O 端口控制，APCTL1 用来控制这些管脚与 ADC 模块通道 0-7 的连接。

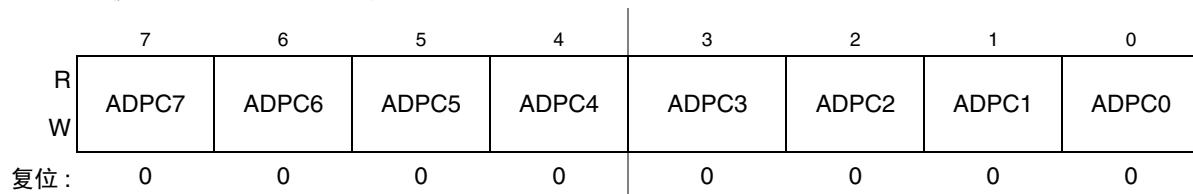


图 10-11. 管脚控制寄存器 1 (APCTL1)

表 10-9. APCTL1 寄存器字段描述

字段	描述
7 ADPC7	<b>ADC 管脚控制 7</b> — ADPC7 用来控制与通道 AD7 连接的管脚。 0 AD7 管脚 I/O 控制使能 1 AD7 管脚 I/O 控制禁止
6 ADPC6	<b>ADC 管脚控制 6</b> — ADPC6 用来控制与通道 AD6 连接的管脚。 0 AD6 管脚 I/O 控制使能 1 AD6 管脚 I/O 控制禁止
5 ADPC5	<b>ADC 管脚控制 5</b> — ADPC5 用来控制与通道 AD5 连接的管脚。 0 AD5 管脚 I/O 控制使能 1 AD5 管脚 I/O 控制禁止
4 ADPC4	<b>ADC 管脚控制 4</b> — ADPC4 用来控制与通道 AD4 连接的管脚。 0 AD4 管脚 I/O 控制使能 1 AD4 管脚 I/O 控制禁止
3 ADPC3	<b>ADC 管脚控制 3</b> — ADPC3 用来控制与通道 AD3 连接的管脚。 0 AD3 管脚 I/O 控制使能 1 AD3 管脚 I/O 控制禁止
2 ADPC2	<b>ADC 管脚控制 2</b> — ADPC2 用来控制与通道 AD2 连接的管脚。 0 AD2 管脚 I/O 控制使能 1 AD2 管脚 I/O 控制禁止
1 ADPC1	<b>ADC 管脚控制 1</b> — ADPC1 用来控制与通道 AD1 连接的管脚。 0 AD1 管脚 I/O 控制使能 1 AD1 管脚 I/O 控制禁止
0 ADPC0	<b>ADC 管脚控制 0</b> — ADPC0 用来控制与通道 AD0 连接的管脚。 0 AD0 管脚 I/O 控制使能 1 AD0 管脚 I/O 控制禁止

#### 10.4.9 管脚控制寄存器 2 (APCTL2)

APCTL2 用来控制 ADC 模块的通道 8-15。

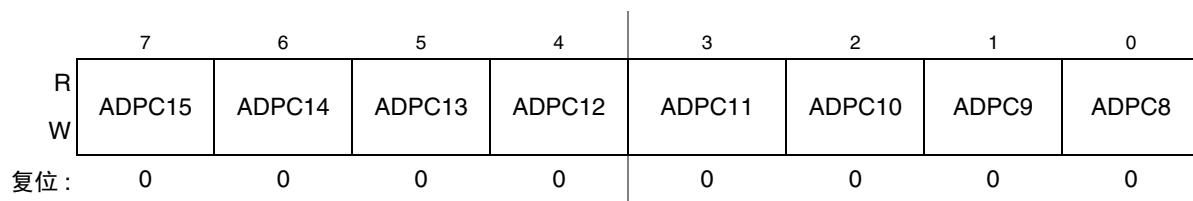


图 10-12. 管脚控制寄存器 2 (APCTL2)

表 10-10. APCTL2 寄存器字段描述

字段	描述
7 ADPC15	<b>ADC 管脚控制 15</b> — ADPC15 用来控制与通道 AD15 连接的管脚。 0 AD15 管脚 I/O 控制使能 1 AD15 管脚 I/O 控制禁止
6 ADPC14	<b>ADC 管脚控制 14</b> — ADPC14 用来控制与通道 AD14 连接的管脚。 0 AD14 管脚 I/O 控制使能 1 AD14 管脚 I/O 控制禁止
5 ADPC13	<b>ADC 管脚控制 13</b> — ADPC13 用来控制与通道 AD13 连接的管脚。 0 AD13 管脚 I/O 控制使能 1 AD13 管脚 I/O 控制禁止
4 ADPC12	<b>ADC 管脚控制 12</b> — ADPC12 用来控制与通道 AD12 连接的管脚。 0 AD12 管脚 I/O 控制使能 1 AD12 管脚 I/O 控制禁止
3 ADPC11	<b>ADC 管脚控制 11</b> — ADPC11 用来控制与通道 AD11 连接的管脚。 0 AD11 管脚 I/O 控制使能 1 AD11 管脚 I/O 控制禁止
2 ADPC10	<b>ADC 管脚控制 10</b> — ADPC10 用来控制与通道 AD10 连接的管脚。 0 AD10 管脚 I/O 控制使能 1 AD10 管脚 I/O 控制禁止
1 ADPC9	<b>ADC 管脚控制 9</b> — ADPC9 用来控制与通道 AD9 连接的管脚。 0 AD9 管脚 I/O 控制使能 1 AD9 管脚 I/O 控制禁止
0 ADPC8	<b>ADC 管脚控制 8</b> — ADPC8 用来控制与通道 AD8 连接的管脚。 0 AD8 管脚 I/O 控制使能 1 AD8 管脚 I/O 控制禁止

#### 10.4.10 管脚控制寄存器 3 (APCTL3)

APCTL3 用来控制 ADC 模块的通道 16-23。

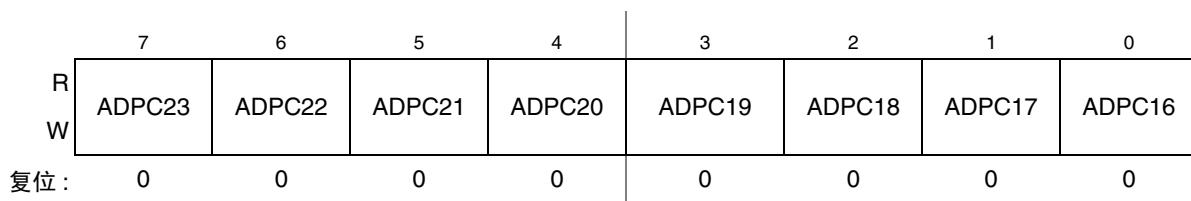


图 10-13. 管脚控制寄存器 3 (APCTL3)

表 10-11. APCTL3 寄存器字段描述

字段	描述
7 ADPC23	<b>ADC 管脚控制 23</b> — ADPC23 用来控制与通道 AD23 连接的管脚。 0 AD23 管脚 I/O 控制使能 1 AD23 管脚 I/O 控制禁止
6 ADPC22	<b>ADC 管脚控制 22</b> — ADPC22 用来控制与通道 AD22 连接的管脚。 0 AD22 管脚 I/O 控制使能 1 AD22 管脚 I/O 控制禁止
5 ADPC21	<b>ADC 管脚控制 21</b> — ADPC21 用来控制与通道 AD21 连接的管脚。 0 AD21 管脚 I/O 控制使能 1 AD21 管脚 I/O 控制禁止
4 ADPC20	<b>ADC 管脚控制 20</b> — ADPC20 用来控制与通道 AD20 连接的管脚。 0 AD20 管脚 I/O 控制使能 1 AD20 管脚 I/O 控制禁止
3 ADPC19	<b>ADC 管脚控制 19</b> — ADPC19 用来控制与通道 AD19 连接的管脚。 0 AD19 管脚 I/O 控制使能 1 AD19 管脚 I/O 控制禁止
2 ADPC18	<b>ADC 管脚控制 18</b> — ADPC18 用来控制与通道 AD18 连接的管脚。 0 AD18 管脚 I/O 控制使能 1 AD18 管脚 I/O 控制禁止
1 ADPC17	<b>ADC 管脚控制 17</b> — ADPC17 用来控制与通道 AD17 连接的管脚。 0 AD17 管脚 I/O 控制使能 1 AD17 管脚 I/O 控制禁止
0 ADPC16	<b>ADC 管脚控制 16</b> — ADPC16 用来控制与通道 AD16 连接的管脚。 0 AD16 管脚 I/O 控制使能 1 AD16 管脚 I/O 控制禁止

## 10.5 功能描述

复位期间或当 ADCH 位都高时，ADC 模块禁止。当已经完成当前转换，而下一次转换还未发起时，模块进入空闲状态。空闲时，模块处于最低功耗状态。

ADC 可以对软件选择的任意通道实施模数转换。在 12 位和 10 位模式中，所选的通道电压通过逐次逼近算法被转换成 12 位数字结果。在 8 位模式中，所选的通道电压通过逐次逼近算法被转换成 9 位数字结果。

转换完成后，结果保存在数据寄存器（ADCRH 和 ADCRL）中。在 10 位模式中，结果被圆整到 10 位并保存在数据寄存器（ADCRH 和 ADCRL）中。在 8 位模式中，结果被圆整到 8 位并保存在 ADCRL 中。然后设置转换完成标记（COCO），如果已经使能了转换完成中断（AIEN = 1），则触发中断。

ADC 模块能够自动地把转换结果与比较寄存器的内容进行比较。通过设置 ACFE 位并结合任意一种转换模式和配置一起运行，就使能了比较功能。

### 10.5.1 时钟选择和分频控制

ADC 模块可选择 4 个时钟源之一，然后由可配置值进行分频，生成转换器的输入时钟（ADCK）。时钟源通过 ADICLK 位设置从以下源中选择。

- 总线时钟，等于软件运行的频率。这是复位后的默认选择。
- 总线时钟除以 2，如果总线时钟很高，允许总线时钟最大除以 16。
- ALTCLK，由此 MCU 定义（参见本章节的概述部分）。
- 异步时钟（ADACK）- 该时钟从 ADC 模块内部时钟源产生。当 MCU 则处于等待或 STOP3 模式时，此时钟源仍然有效，从而实现此模式下的低噪音转换。

无论选择哪种时钟，其频率必须在 ADCK 的指定频率范围内。如果可用时钟太慢，ADC 将无法保证正常运行。如果可用时钟太快，那么时钟必须分频为适当的频率。除数由 ADIV 位指定，可以是 1、2、4 或 8。

### 10.5.2 输入选择和管脚控制

管脚控制寄存器（APCTL3, APCTL2 和 APCTL1）用来禁止对作为模拟输入的管脚的 I/O 控制功能。当置位管脚控制寄存器相应位时，对应的 MCU 管脚进入以下状态：

- 输出缓冲器进入高阻抗状态。
- 输入缓冲器禁止。对于其输入缓冲器被禁止的任何管脚，I/O 端口读数均返回 0。
- 上拉禁止。

### 10.5.3 硬件触发

ADC 模块有一个可选的异步硬件转换触发 ADHWT，当设置了 ADTRG 位时，ADHWT 使能。并不是所有 MCU 上都有这个源。如需了解该 MCU 的特定 ADHWT 源的更多报文，请参见本章概述部分。

当提供了 ADHWT 源且已使能了硬件触发 (ADTRG=1) 时, ADHWT 的上升沿上就发起转换。如果转换进行过程中再出现上升沿, 该上升沿就被忽略。在连续转换模式下, 只是触发连续转换的初始上升沿有效。硬件触发功能可以结合任意转换模式和配置一起运行。

## 10.5.4 转换控制

转换可以在 12 位、10 位或 8 位模式下进行, 由 MODE 位决定。转换可以由软件或硬件触发发起。此外, ADC 模块也可以设置为低功率操作、长采样时间、连续转换或将转换结果与软件设定的比较值自动比较的方式。

### 10.5.4.1 发起转换

转换的发起:

- 如果选择软件触发操作, 在 ADCSC1 写入 (ADCH 位不全为 1) 后。
- 即使选择硬件触发操作, 在硬件触发 (ADHWT) 事件后。
- 当使能连续转换时, 在把结果传输到数据寄存器后。

如果使能连续转换, 在当前转换完成会就自动发起一个新转换。在软件触发操作中, 连续转换在写入 ADCSC1 后就开始, 并且一直持续直到被中止。在硬件触发操作中, 连续转换在硬件触发事件后开始, 并且一直持续直到被中止。

### 10.5.4.2 完成转换

当转换结果被传输到数据结果寄存器 ADCRH 和 ADCRL 中时, 转换完成。这通过置位 COCO 标识。如果置位 COCO 时 AIEN 高, 就会触发中断。

如果有原有数据正在在 12 位或 10 位 MODE 中被读取, 拦截机制可以防止新结果覆盖 ADCRH 和 ADCRL 中的原有数据 (ADCRH 寄存器已被读取, 但 ADCRL 寄存器还没有)。当拦截处于工作状态时, 数据传输被拦截, COCO 未被置位而且新转换结果丢失。在使能了比较功能但比较条件不成立的单次转换中, 拦截不会起作用, ADC 操作终止。在其他模式下, 当数据传输被拦截时, 都会发起另另外一次转换, 而无论 ADCO 处于何种状态 (单次或连续转换使能)。

如果单次转换使能, 拦截机制可能会导致几个转换被丢失, 且功耗过高。为了避免这一问题, 数据寄存器必须在发起单转换且完成了转换后再读取。

### 10.5.4.3 中止转换

当出现下列情况时, 进行中的任何转换都将中止:

- ADCSC1 写入 (当前转换被中止, 如果 ADCH 不都是 1, 则会发起一个新转换。)
- ADCSC2、ADCCFG、ADCCVH 或 ADCCVL 写入。这表明出现运行模式更改, 当前转换因此无效。
- MCU 复位。
- MCU 进入停止模式, ADACK 被禁止。

当转换被中止时, 数据寄存器、ADCRH 和 ADCRL 的内容都不变, 保持上次成功转换完成后传输的值。如果由于复位中断了转换, ADCRH 和 ADCRL 返回其复位状态。

#### 10.5.4.4 功率控制

ADC 模块在转换发起前一直保持空闲状态。如果 ADACK 被选为转换时钟源，ADACK 时钟发生器也会被使能。

可以通过设置 ADLPC 降低运行功耗。这也会降低 fADCK 的最大值（参见电气规范）。

#### 10.5.4.5 采样时间和总转换时间

总转换时间取决于采样时间（由 ADLSMP 决定）、MCU 总线频率、转换模式（8 位、10 位或 12 位）和转换时钟的频率（fADCK）。模块使能后，输入采样开始。ADLSMP 用来选择短（3.5 ADCK 周期）和长（23.5 ADCK 周期）采样时间。采样完成时，转换器与输入通道隔离，实施逐次逼近算法来决定模拟信号的数字值。一旦转换算法结束，转换结果就传输到 ADCRH 和 ADCRL。

如果总线频率小于 fADCK 频率，在使能短采样时间（ADLSMP=0）时，不能保证连续转换模式下采样时间的准确性。如果总线频率小于 fADCK 频率的 1/11，在使能长采样时间（ADLSMP=1）时，不能保证连续转换模式下采样时间的准确性。

表 10-12 概括介绍了不同条件下的最长总转换时间。

表 10-12. 总转换时间与控制条件之比较

转换类型	ADICLK	ADLSMP	最长总转换时间
单次转换或连续转换的第一次 8 位	0x, 10	0	20 个 ADCK 周期 + 5 个总线时钟周期
单次转换或连续转换的第一次 10 位或 12 位	0x, 10	0	23 个 ADCK 周期 + 5 个总线时钟周期
单次转换或连续转换的第一次 8 位	0x, 10	1	40 个 ADCK 周期 + 5 个总线时钟周期
单次转换或连续转换的第一次 10 位或 12 位	0x, 10	1	43 个 ADCK 周期 + 5 个总线时钟周期
单次转换或连续转换的第一次 8 位	11	0	5ms + 20 ADCK + 5 个总线时钟周期
单次转换或连续转换的第一次 10 位或 12 位	11	0	5ms + 23 ADCK + 5 个总线时钟周期
单次转换或连续转换的第一次 8 位	11	1	5ms + 40 ADCK + 5 个总线时钟周期
单次转换或连续转换的第一次 10 位或 12 位	11	1	5ms + 43 ADCK + 5 个总线时钟周期
后续的连续转换 8 位； $f_{BUS} \geq f_{ADCK}$	xx	0	17 ADCK 周期
后续的连续转换 10 位或 12 位； $f_{BUS} \geq f_{ADCK}$	xx	0	20 ADCK 周期
后续的连续转换 8 位； $f_{BUS} \geq f_{ADCK}/11$	xx	1	37 ADCK 周期
后续的连续转换 10 位或 12 位； $f_{BUS} \geq f_{ADCK}/11$	xx	1	40 ADCK 周期

最长总转换时间由所选的时钟源和分频率决定。时钟源可以由 ADICLK 位选择，分频率由 ADIV 位指定。例如，在 10 位模式中，总线时钟选为输入时钟源、输入时钟分频率选为除以 1、总线频率为 8 MHz，那么单转换的转换时间为：

$$\text{转换时间} = \frac{23 \text{ ADCK Cyc}}{8 \text{ MHz}/1} + \frac{5 \text{ bus Cyc}}{8 \text{ MHz}} = 3.5 \text{ ms}$$

总线周期数 = 3.5 ms × 8 MHz = 28 周期

#### 注意

ADCK 频率必须介于 fADCK 最小值和最大值之间才符合 ADC 技术规范

### 10.5.5 自动比较功能

可以配置比较功能来检查上限或下限。在进行完输入采样并转换后，结果与比较值（ADCCVH 和 ADCCVL）补数相加。比较上限（ACFGT = 1）时，如果结果大于或等于比较值，就设置 COCO。比较下限（ACFGT = 0）时，如果结果小于比较值，就设置 COCO。转换结果和比较值的补数相加得到的值被传输到 ADCRH 和 ADCRL 中。

当转换完成而比较功能使能时，如果比较条件不成立，就不设置 COCO，且没有数据传输到结果寄存器。如果使能了 ADC 中断（AIEN = 1），那么当设置 COCO 时就会生成 ADC 中断。

#### 注意

这个比较功能可以用来监控通道的电压，这时 MCU 可能处于等待模式或 STOP3 模式。在满足比较条件时，ADC 中断会唤醒 MCU。

### 10.5.6 MCU 等待模式运行

WAIT 指令使 MCU 处于更低功耗的待机模式，待机模式的恢复非常快，因为时钟源保持有效状态。如果正在进行转换时 MCU 进入等待模式，那么转换会继续，直到完成。在 MCU 处于等待模式时，通过硬件触发或使能连续转换，可以发起转换。

在等待模式中，总线时钟、总线时钟除以 2 和 ADACK 都可以作为转换时钟源。在等待模式中，是否将 ALTCLK 作为时钟源取决于该 MCU 对 ALTCLK 的定义。有关该 MCU 的特定 ALTCLK 的更多信息，请参阅本章概述。

如果 ADC 中断使能（AIEN = 1），转换完成事件就会设置 COCO，生成 ADC 中断，把 MCU 从等待模式中唤醒。

### 10.5.7 MCU STOP3 模式运行

STOP 指令使 MCU 进入低功耗待机模式，在该模式中，MCU 上的大多数甚至所有时钟源都被禁止。

### 10.5.7.1 ADACK 禁止的 STOP3 模式

如果异步时钟 ADACK 未被选为转换时钟，执行 STOP 指令将中止当前转换，并把 ADC 置入空闲状态。ADCRH 和 ADCRL 内容不受 STOP3 模式的影响。从 STOP3 模式退出后，需要软件或硬件触发来重新开始转换。

### 10.5.7.2 ADACK 使能的 STOP3 模式

如果异步时钟 ADACK 已被选为转换时钟，ADC 会在 STOP3 模式中继续运行。为了保证 ADC 操作，MCU 的电压调节器在 STOP3 模式中必须保持工作状态。有关该 MCU 的配置报文，请参阅模块介绍。

如果正在进行转换时 MCU 进入 STOP3 模式，那么转换会继续，直到完成。在 MCU 处于 STOP3 模式时，通过硬件触发或使能连续转换，可以发起转换。

如果 ADC 中断使能 (AIEN = 1)，转换完成事件就会设置 COCO，生成 ADC 中断，把 MCU 从 STOP3 模式中唤醒。

#### 注意

ADC 模块可以从停止状态唤醒系统，让 MCU 开始以运行级电流工作，而不产生系统级中断。为了防止这一情况，在进入 STOP3 并继续 ADC 转换时，软件应当确保数据传输拦截机制（在 11.4.4.2 节“完成转换”中做过介绍）已经清除。

### 10.5.8 MCU STOP1 和 STOP2 模式运行

当 MCU 进入 STOP1 或 STOP2 模式时，ADC 模块会被自动禁止。所有模块寄存器在退出 STOP1 或 STOP2 模式后都包含它们的复位值。因此，在从 STOP1 或 STOP2 退出后，模块必须重新使能和重新配置。

## 10.6 初始化报文

本节给出了一个为用户提供如何初始化和配置 ADC 模块的一些基本指导的示例。用户可以从众多选项中灵活选择配置模块的 8 位、10 位或 12 位分辨率、单或连续转换、查询或中断法。

请参见表 11-8、表 11-9 和表 11-10，获取示例中使用的报文。

### 注意

十六进制值前加了一个 0x，二进制值前加了一个 %，十进制值没有前置字符。

### 10.6.1 ADC 模块初始化示例

#### 10.6.1.1 初始化顺序

在使用 ADC 模块完成转换前，必须进行初始化步骤。初始化的常见顺序如下：

1. 更新配置寄存器 (ADCCFG)，选择输入时钟源和用来生成内部时钟 ADCK 的分频率。该寄存器也可用于选择采样时间和低功率配置。
2. 更新状态和控制寄存器 2 (ADCSC2)，选择转换触发（硬件或软件）与比较功能选项，如使能的话。
3. 更新状态和控制寄存器 1 (ADCSC1)，选择转换是连续转换还是只完成一次，并使能或禁止转换完成中断。同时还选择执行转换的输入通道。

#### 10.6.1.2 伪代码示例

在本例中，ADC 模块设置为：中断使能，在低功率情况下实施 10 位转换，该转换在输入通道 1 上有一个长采样时间，这里的内部 ADCK 时钟用总线时钟除以 1 得来。

##### ADCCFG = 0x98 (%10011000)

位 7	ADLPC	1	配置用于低功率（降低最快时钟速度）
位 6:5	ADIV	00	将 ADCK 设置为输入时钟除以 1
位 4	ADLSMP	1	配置用于长采样时间
位 3:2	MODE	10	在 10 位转换上设置模式
位 1:0	ADICLK	00	选择总线时钟为输入时钟源

##### ADCSC2 = 0x00 (%00000000)

位 7	ADACT	0	标记表明是否正在进行转换
位 6	ADTRG	0	软件触发已选
位 5	ACFE	0	比较功能禁止
位 4	ACFGT	0	本例中未使用
位 3:2		00	未实施或保留，总是读取 0
位 1:0		00	供飞思卡尔内部使用，总是写入 0

##### ADCSC1 = 0x41 (%01000001)

位 7	COCO	0	转换完成时设置的只读标记
位 6	AIEN	1	转换完成中断使能
位 5	ADCO	0	仅一次转换（连续转换禁止）
位 4:0	ADCH	00001	输入通道 1 选为 ADC 输入通道

**ADCRH/L = 0xxx**

保留转换结果。读取低字节 (ADCRL) 前的高字节 (ADCRH)，这样转换数据就不会被下一次转换的数据覆盖。

**ADCCVH/L = 0xxx**

当比较功能使能时保留比较值

**APCTL1=0x02**

AD1 管脚 I/O 控制禁止。其他 AD 管脚仍为通用 I/O 管脚

**APCTL2=0x00**

所有其他 AD 管脚仍为通用 I/O 管脚

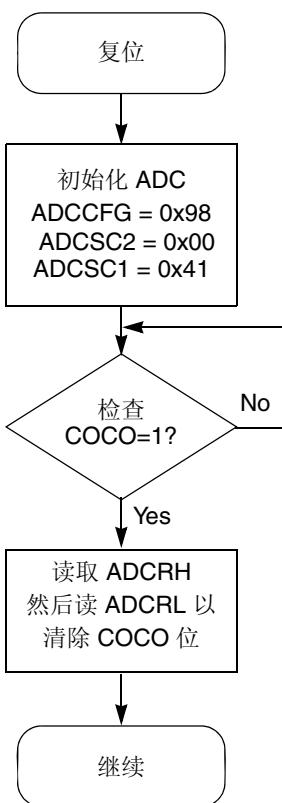


图 10-14. 初始化流程图示例

## 10.7 应用报文

本节介绍了在应用中使用 ADC 模块的相关报文。ADC 已被集成到微控制器中，供需要 A/D 转换器的嵌入式控制应用使用。

### 10.7.1 外部管脚和布线

以下几节讨论与 ADC 模块相关的外部管脚以及为获得最佳结果，应该如何使用它们。

#### 10.7.1.1 模拟电源管脚

ADC 模块有模拟电源和模拟地 ( $V_{DDAD}$  和  $V_{SSAD}$ )，在有些器件上它们作为独立管脚。在其余器件上， $V_{SSAD}$  与 MCU 数字  $V_{SS}$  共用同一管脚。还有一些器件， $V_{SSAD}$  和  $V_{DDAD}$  同时共用 MCU 数字电源管脚。在这些情况下，模拟电源就使用单独的电极极片，与相应的数字电源管脚内部相连，这样电源之间就保持一定程度的隔离。

当作为独立管脚出现时， $V_{DDAD}$  和  $V_{SSAD}$  必须连接到与它们相应的 MCU 数字电源 ( $V_{DD}$  和  $V_{SS}$ ) 相同的电压水平上，并且在布线时必须小心，以实现最好隔离效果，滤波电容要尽可能靠近芯片布置。

当为模拟和数字电源使用独立的电源时，这些电源间的接地连接必须在  $V_{SSAD}$  管脚位置。这应当是这些电源间唯一的接地连接（如果可能的话）。 $V_{SSAD}$  管脚是很好的单点接地位置。

#### 10.7.1.2 模拟参考管脚

除模拟电源外，ADC 模块还与两个参考电压输入连接。高参考是  $V_{REFH}$ ，在有些器件上可能被与  $V_{DDAD}$  相同的管脚共用。低参考是  $V_{REFL}$ ，在有些器件上可能被与  $V_{SSAD}$  相同的管脚共用。

当作为独立管脚出现时， $V_{REFH}$  可能连接到与  $V_{DDAD}$  相等的电压水平上，或者可能由介于  $V_{DDAD}$  最小规范和  $V_{DDAD}$  电平间的外部源驱动 ( $V_{REFH}$  必须不能超过  $V_{DDAD}$ )。当作为独立管脚出现时， $V_{REFL}$  必须连接到与  $V_{SSAD}$  相同的电压水平上，并且在布线时必须小心，以实现最好隔离效果，滤波电容要尽可能靠近芯片布置。

在每个逐次逼近步骤中用来给电容阵列充电所需的峰值电流型交流电通过  $V_{REFH}$  和  $V_{REFL}$  环路获取。满足这一电流要求的最佳外部组件是  $0.1\text{mF}$  电容器，必须有出色的高频特征。该电容器连接  $V_{REFH}$  和  $V_{REFL}$ ，必须尽可能靠近封装管脚。不建议在电路中使用电阻，因为会导致压降，进而可能导致转换错误。这个路径中的电感必须最小（仅寄生）。

#### 10.7.1.3 模拟输入管脚

外部模拟输入通常与 MCU 器件上的数字 I/O 管脚共用。在管脚控制寄存器中设置相应的控制位就能禁止管脚 I/O 控制。转换也可以在管脚控制寄存器位没设置时进行，建议在把管脚作为模拟输入使用时，最好设置管脚控制寄存器位。这样就可以避免可能的问题，因为输出缓冲器处于高电阻状态，且禁止上拉。此外，当输入缓冲器的输入不在  $V_{DD}$  或  $V_{SS}$  时，会消耗 DC 电流。因而为用作模拟输入的所有管脚设置管脚控制寄存器位，可以实现最低的工作电流。

试验数据显示，有出现噪音或源阻抗很高时，模拟输入上的电容器有助于改进性能。使用  $0.01\text{mF}$  电容器（带有出色的高频性能）完全可以做到这一点。不是所有情况都需要这些电容器，但使用时，必须尽可能地将它们放在封装管脚旁边，并且参考到  $V_{SSA}$ 。

为了进行正确转换，输入电压必须在  $V_{REFH}$  和  $V_{REFL}$  之间。如果输入等于或高于  $V_{REFH}$ ，转换器电路将把信号转换成  $0xFFFF$ （全范围 12 位）、 $0x3FF$ （全范围 10 位）或  $0xFF$ （全范围 8 位）。如果输入等于或低于  $V_{REFL}$ ，转换器电路将把信号转换成  $0x000$ 。 $V_{REFH}$  and  $V_{REFL}$  间的输入电压是直线线性转换。当采样电容充电时，会出现与  $V_{REFL}$  相关的短暂电流。当  $ADLSMP$  低时，输入采样为 3.5 个周期的  $ADCK$  源；当  $ADLSMP$  高时，输入采样为 23.5 个周期。

为了把由于灌电流带来的精度性损失降到最低，转换期间靠近模拟输入管脚的引脚不要有跳变。

## 10.7.2 错误源

A/D 转换中存在几个错误源。我们将在以下几节中讨论。

### 10.7.2.1 采样错误

为了实现正确转换，采样时间必须足够长才能获得适当的精度。假设最大输入阻抗约为  $7\text{k}\Omega$ ，输入电容约为  $5.5\text{pF}$ ，最小采样窗口（8KHz 的最大  $ADCK$  频率时，3.5 个周期）可以实现不超过  $1/4\text{LSB}$ （分辨率为 12 位）的采样，但外部模拟源 ( $R_{AS}$ ) 的电阻需小于  $2\text{k}\Omega$ 。

通过设置  $ADLSMP$ （将采样窗口增加到 23.5 个周期），或降低  $ADCK$  频率来增加采样时间，可以获得更高输入阻抗或更高精度采样。

### 10.7.2.2 管脚漏电流错误

如果外部模拟源电阻 ( $R_{AS}$ ) 高，I/O 管脚上的漏电流就可能造成转换错误。如果应用不能容忍该错误，应一直使  $R_{AS}$  低于  $V_{DDAD} / (2^N * I_{LEAK})$ ，实现低于  $1/4\text{LSB}$  的漏电流错误（在 8 位中  $N = 8$ ；在 10 位中  $N=10$ ；在 12 位中  $N=12$ ）。

### 10.7.2.3 噪音引发的错误

采样或转换过程中出现的系统噪音可能会影响到转换的准确性。只有当满足下列条件时，才能保证 ADC 达到指定的精度：

- 从  $V_{REFH}$  到  $V_{REFL}$  有  $0.1\text{\mu F}$  的低 ESR 电容器
- 从  $V_{DDAD}$  到  $V_{SSAD}$  有  $0.1\text{\mu F}$  的低 ESR 电容器
- 如果主电源使用感性隔离， $V_{DDAD}$  至  $V_{SSAD}$  间还需要额外放置一个  $1\text{\mu F}$  电容器。
- $V_{SSAD}$ （和  $V_{REFL}$ ，如果连接）与地线网络的  $V_{SS}$  连接。
- 在发起 ADC 转换前进入等待或 STOP3 模式（硬件触发转换），或发起 ADC 转换后（硬件或软件触发转换）立即进入等待或 STOP3 模式。
  - 对于软件触发转换，写入  $ADCSC1$  后紧跟 WAIT 指令或 STOP 指令。
  - 对于 STOP3 模式运行，选择  $ADACK$  作为时钟源。STOP3 中的运行降低了  $V_{DD}$  噪音，但会延长有效转换时间，因为从 STOP 恢复的时间。
- 转换期间，MCU 上没有 I/O 口跳变，无论是输入还是输出。

可能会出现外部系统工作造成辐射或传导噪音，或者过多  $V_{DD}$  噪音被耦合到 ADC 的情况。在这些情况下，或者当 MCU 不能置于等待或 STOP3 模式，或 I/O 跳变不能暂停时，下面这些推荐操作会降低噪音对精度的影响：

- 将所选输入通道与  $V_{REFL}$  或  $V_{SSAD}$  之间放置  $0.01 \mu F$  电容器 ( $C_{AS}$ )，（这将改善噪音问题，但仍会影响采样速率，基于模拟源的阻抗）。
- 通过对连续多次转换结果求算术平均。要消除 1LSB，一次采样的错误，需要 4 次转换。
- 通过关闭异步时钟 (ADACK) 和算术平均可以降低同步噪音的影响。与 ADCK 同步的噪音无法通过平均消除。

#### 10.7.2.4 编码宽度和量化错误

ADC 将理想的直线传递函数量化为 4096 个步骤（在 12 位模式中）。每个步骤刚好都有相同的高度（1 个代码）和宽度。宽度的定义是两个代码转换点间的 delta。N 位转换器（本例中，N 可以为 8、10 或 12）理想的代码宽度是（就像 1LSB 定义的那样）：

$$1 \text{ lsb} = (V_{REFH} - V_{REFL}) / 2^N$$

等式 10-2

由于结果的数字化原因，可能会出现固有的量化错误。对于 8 位或 10 位转换，当电压位于两点间的中点时，代码就会进行切换，此时直线传递函数与实际的传递函数相一致。因此，在 8 位或 10 位模式中，量化错误是  $\pm 1/2 \text{ lsb}$ 。但是，由此导致的第一次转换 (0x000) 的代码宽度为  $1/2 \text{ lsb}$ ，而最后一次转换 (0xFF or 0x3FF) 的代码宽度为  $1.5 \text{ lsb}$ 。

对于 12 位转换，代码仅在整个代码宽度出现后才进行转换，因此量化错误在 -1 lsb 至 0 lsb 之间，每个步骤的代码宽度均为 1 lsb。

#### 10.7.2.5 线性错误

ADC 还可能呈现几种形式的非线性。大量的努力被用来减少这些错误，但系统应知道这些错误，因为它们会影响总体精度。这些错误包括：

- 零刻度错误 ( $E_{ZS}$ ) ——这个错误的定义是第一次转换的实际代码宽度和理想代码宽度（在 8 位/10 位中为  $1/2 \text{ lsb}$ ，在 12 位模式中为  $1 \text{ LSB}$ ）之间的差异。注意，如果第一次转换是 0x001，那就使用实际的 0x001 代码宽度及其理想宽度（ $1 \text{ LSB}$ ）之间的差异。
- 满刻度错误 ( $E_{FS}$ ) ——这个错误的定义是最后一次转换的实际代码宽度和理想代码宽度（在 8 位/10 位中为  $1.5 \text{ lsb}$ ，在 12 位模式中为  $1 \text{ LSB}$ ）之间的差异。注意，如果最后一次转换是 0x3FE，那么就使用实际的 0x3FE 代码宽度及其理想宽度（ $1 \text{ LSB}$ ）之间的差异。
- 差分非线性错误 (DNL) ——这个错误的定义是所有转换的实际代码宽度和理想代码宽度之间的最大差异。
- 积分非线性错误 (INL) ——这个错误的定义是 DNL 求和达到的最高值（绝对值）。更简单地说，对于所有代码来说，这是给定代码的实际转换电压和相应的理想转换电压之间的最大差异。
- 总不可调整错误 (TUE) ——这个错误的定义是实际转换功能和理想直线转换功能之间的差异，因此包括所有形式的错误。

### 10.7.2.6 代码抖动、非单调性和丢码

数模转换器容易受三种特殊形式的错误影响，它们是代码抖动、非单一性和丢码。

代码抖动是把某些点的给定输入电压在重复采样时的转换出两个值。在理想情况下，当输入电压低于转换电压时，转换器会产生更低代码（反之亦然）。但是，对于转换电压附近的一系列输入电压来说，即使非常小的系统噪音也可能造成转换器的抖动（两个代码之间）。在 8 位或 10 位模式中，这个范围通常是  $1/2 \text{ lsb}$ ；在 12 位模式中，这个范围通常是  $2 \text{ lsb}$ ，并会随着噪音的提高而提高。

通过重复进行输入采样和算术平均，这个错误可能会减小。此外，11.6.2.3 节中也讨论了一些减小这一错误的技巧。

非单调性的定义是转换器将高的输入电压转换到低的代码（代码抖动除外）。丢码是那些没有任何输入值进行转换的值。

在 8 位或 10 位模式中，ADC 保证具有单调性，并且没有丢码。

# 第 11 章

## IIC 模块 (S08IICV2)

### 11.1 介绍

IIC 模块 (IIC) 提供了不同器件间的通信方法。接口可以在最大的总线负载和时序下，支持最高 100kbps 的传输速率。器件可以在较低总线负载下、以更高的波特率（最高时钟 /20）运行。最大通信长度和可以连接的器件数量受 400 pF 的最高总线电容限制。

MC9S08DZ60 系列的所有 MCU 都具有 IIC 功能，如以下结构图所示。

#### 注意

在使用 IIC 模块时，禁止为 IIC 管脚使用驱动强度（DSE=0），以实现正确操作。

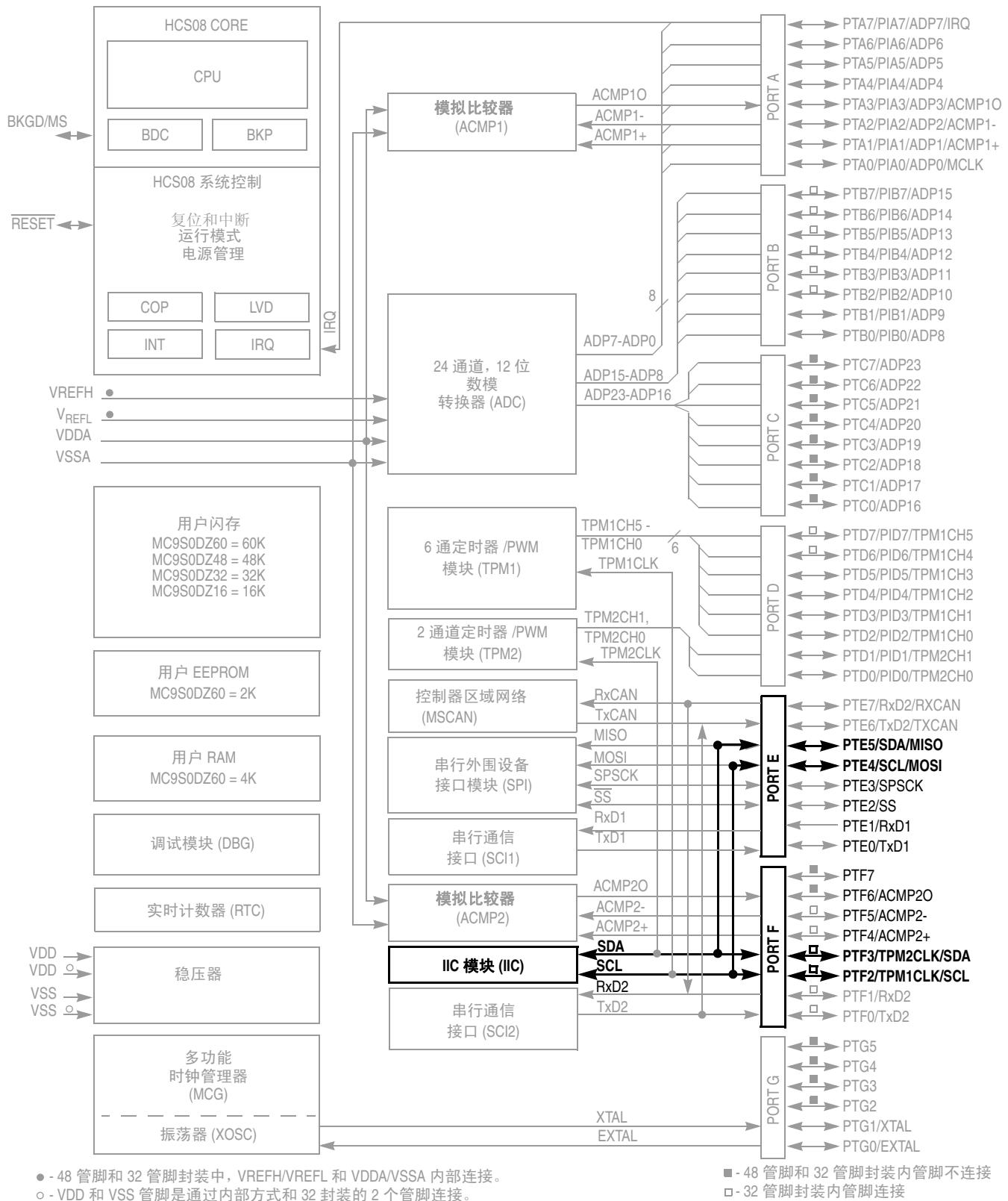


图 11-1. MC9S08DZ60 结构图

### 11.2.1 特性

IIC 包括以下明显的特性：

- 兼容 IIC 总线标准
- 多主操作
- 可以软件编程为 64 种不同串行时钟频率的任意一种
- 软件可选应答位
- 中断驱动的逐字节数据传输
- 仲裁丢失中断，可以自动地从主模式切换到从模式
- 主叫地址识别中断
- 开始和停止信号生成 / 检测
- 重复生成启动信号
- 应答位生成 / 检测
- 总线忙检测
- 通用呼叫识别
- 10 位地址扩展

### 11.2.2 运行模式

不同 MCU 模式中的 IIC 的简要描述如下：

- 运行模式 — 这是基本运行模式。要降低这种模式下的功耗，只需禁止模块。
- 等待模式 — 当 MCU 处于等待模式时，模块继续运行并能够提供唤醒中断。
- 停止模式 — IIC 在 STOP3 模式中是不停止的，以降低功耗。停止指令不会影响 IIC 寄存器状态。STOP2 复位寄存器内容。

### 11.2.3 结构图

图 11-3 是 ICC 的结构图。

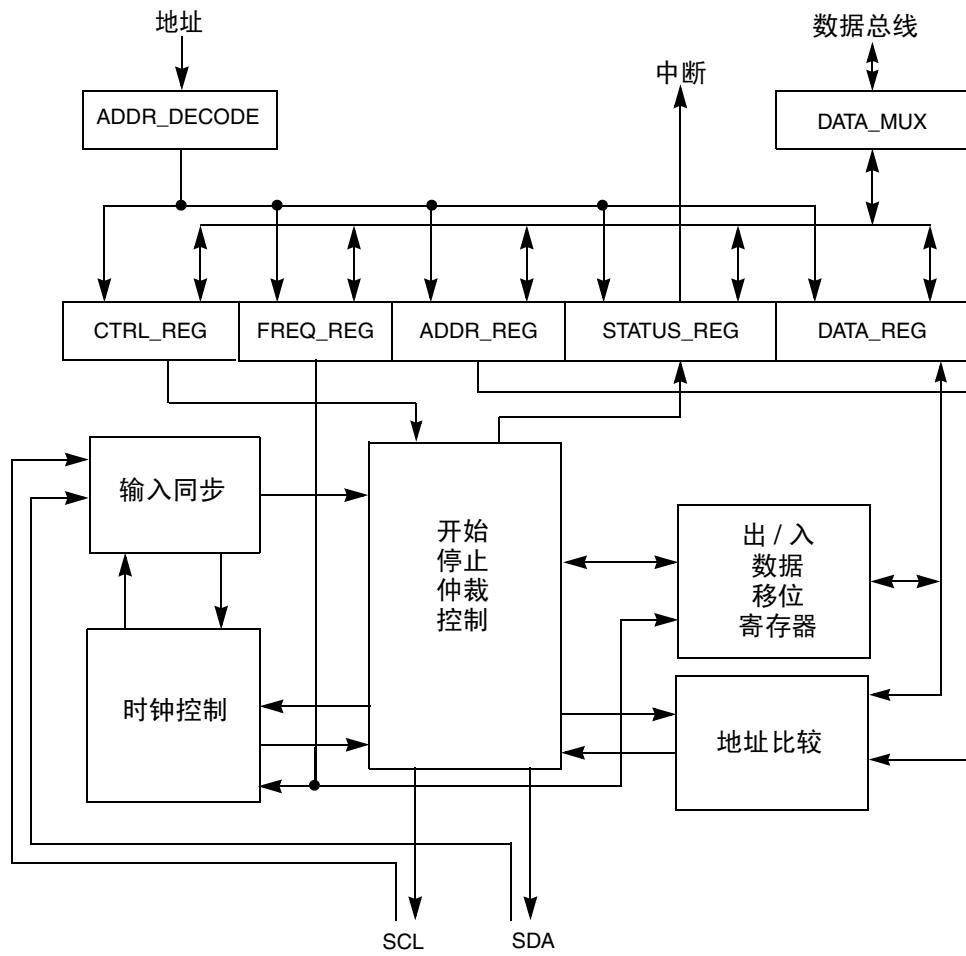


图 11-2. IIC 功能结构图

## 11.3 外部信号描述

本节描述了用户可连接的各个管脚信号。

### 11.3.1 SCL — 串行时钟线

双向 SCL 是 IIC 系统的串行时钟线。

### 11.3.2 SDA — 串行数据线

双向 SDA 是 IIC 系统的串行数据线。

## 11.4 寄存器定义

本节按地址顺序对 IIC 寄存器进行了描述。

如需了解所有 IIC 寄存器的绝对地址分配报文, 请参见本文档存储器章的直接页面寄存器概述。本节仅按参考寄存器和控制位名称对其进行参考。飞思卡尔提供的等式或标头文件用于把这些名称转换成适当的绝对地址。

### 11.4.1 IIC 地址寄存器 (IICA)

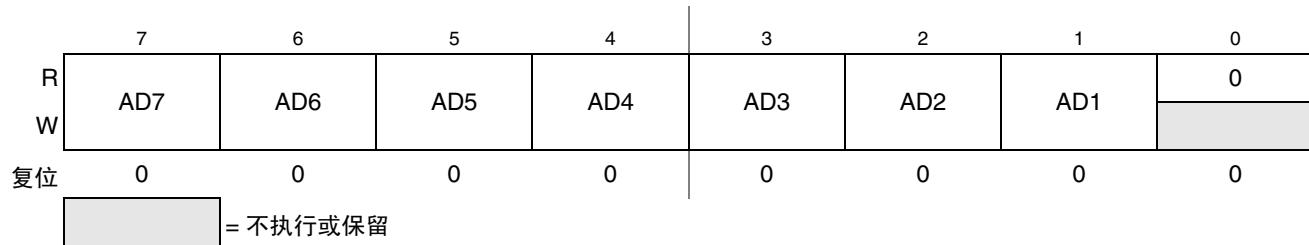


图 11-3. IIC 地址寄存器 (IICA)

表 11-1. IICA 字段描述

字段	描述
7-1 AD[7:1]	从机地址。AD[7:1] 字段包含 IIC 模块将要使用的从机地址。该字段在 7 位地址机制和 10 位地址机制的低 7 位中使用。

### 11.4.2 11.3.2 IIC 分频器寄存器 (IICF)

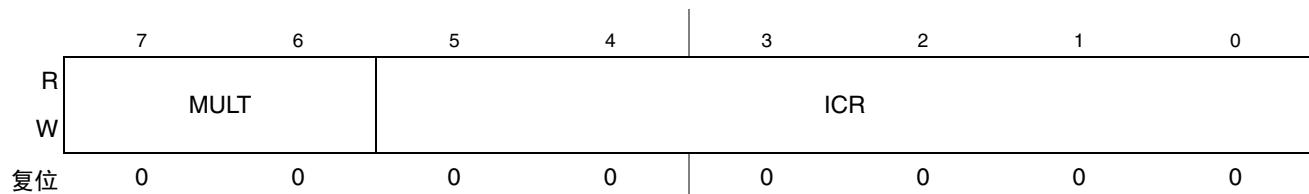


图 11-4. IIC 分频器寄存器 (IICF)

表 11-2. IICF 字段描述

字段	描述
7-6 MULT	IIC 倍频因子。MULT 位定义倍频因子 mul。该倍频因子与 SCL 分频器一起，共同生成 IIC 波特率。MULT 位定义的倍频因子 mul 如下： 00 mul = 01 01 mul = 02 10 mul = 04 11 保留
5-0 ICR	IIC 时钟率。IIC 位用来预分频总线时钟，以便进行比特率选择。这些位和 MULT 位确定 IIC 波特率、SDA 保持时间、SCL 开始保持时间和 SCL 停止保持时间。表 11-4 为 ICR 的相应值提供了 SCL 分频器和保持值。  由倍频因子 mul 和 SCL 分频器生成 IIC 波特率。  $\text{IIC baud rate} = \frac{\text{bus speed (Hz)}}{\text{mul} \times \text{SCLdivider}}$ 等式 11-1 SDA 保持时间是从 SCL (IIC 时钟) 的下降边沿到 SDA (IIC 数据) 变化的延迟。  $\text{SDA 保持时间} = \text{总线周期 (s)} \times \text{mul} \times \text{SDA 保持值}$ 等式 11-2 SCL 开始保持时间是从 SDA (IIC 数据) 的下降边沿 (这时 SCL 处于高一开始状态) 到 SCL (IIC 时钟) 下降边沿的延迟。  $\text{SCL 开始保持时间} = \text{总线周期 (s)} \times \text{mul} \times \text{SCL 开始保持值}$ 等式 11-3 SCL 停止保持时间是从 SCL (IIC 时钟) 的上升边沿到 SDA (IIC 数据) 上升边沿的延迟 (这时 SCL 处于高一停止状态)  $\text{SCL 停止保持时间} = \text{总线周期 (s)} \times \text{mul} \times \text{SCL 停止保持值}$ 等式 11-4

例如，如果总线速率为 8 MHz，下表显示了不同 ICR 和 MULT 选择为达到 100kbps 的 IIC 波特率的可能保持时间值。

表 11-3. 8 MHz 总线速度保持时间值

MULT	ICR	保持时间 (μ s)		
		SDA	SDA SCL 开始	SCL 停止
0x2	0x00	3.500	3.000	5.500
0x1	0x07	2.500	4.000	5.250
0x1	0x0B	2.250	4.000	5.250
0x0	0x14	2.125	4.250	5.125
0x0	0x18	1.125	4.750	5.125

表 11-4. IIC 分频器和保持值

ICR (hex)	SCL 分 频器	SDA 保 持 值	SCL 保 持 (开始) 值	SDA 保 持 (停止) 值
00	20	7	6	11
01	22	7	7	12
02	24	8	8	13
03	26	8	9	14
04	28	9	10	15
05	30	9	11	16
06	34	10	13	18
07	40	10	16	21
08	28	7	10	15
09	32	7	12	17
0A	36	9	14	19
0B	40	9	16	21
0C	44	11	18	23
0D	48	11	20	25
0E	56	13	24	29
0F	68	13	30	35
10	48	9	18	25
11	56	9	22	29
12	64	13	26	33
13	72	13	30	37
14	80	17	34	41
15	88	17	38	45
16	104	21	46	53
17	128	21	58	65
18	80	9	38	41
19	96	9	46	49
1A	112	17	54	57
1B	128	17	62	65
1C	144	25	70	73
1D	160	25	78	81
1E	192	33	94	97
1F	240	33	118	121

ICR (hex)	SCL 分 频器	SDA 保 持 值	SCL 保 持 (开始) 值	SDA 保 持 (停止) 值
20	160	17	78	81
21	192	17	94	97
22	224	33	110	113
23	256	33	126	129
24	288	49	142	145
25	320	49	158	161
26	384	65	190	193
27	480	65	238	241
28	320	33	158	161
29	384	33	190	193
2A	448	65	222	225
2B	512	65	254	257
2C	576	97	286	289
2D	640	97	318	321
2E	768	129	382	385
2F	960	129	478	481
30	640	65	318	321
31	768	65	382	385
32	896	129	446	449
33	1024	129	510	513
34	1152	193	574	577
35	1280	193	638	641
36	1536	257	766	769
37	1920	257	958	961
38	1280	129	638	641
39	1536	129	766	769
3A	1792	257	894	897
3B	2048	257	1022	1025
3C	2304	385	1150	1153
3D	2560	385	1278	1281
3E	3072	513	1534	1537
3F	3840	513	1918	1921

### 11.4.3 IIC 控制寄存器 (IICC1)

	7	6	5	4		3	2	1	0
R	IICEN	IICIE	MST	TX	TXAK	0	0	0	0
W					RSTA				
复位	0	0	0	0	0	0	0	0	0
	= 不执行或保留								

图 11-5. IIC 控制寄存器 (IICC1)

表 11-5. IICC1 字段描述

字段	描述
7 IICEN	IIC 使能。IICEN 位确定是否使能 IIC 模块。 IIC 禁止 1 IIC 使能
6 IICIE	IIC 中断使能。IICIE 位确定是否请求 IIC 中断。 0 IIC 中断请求禁止 1 IIC 中断请求使能
5 MST	主模式选择。当 MST 位从 0 变为 1，总线上产生启动信号，主机模式被选择。当 MST 位从 1 变为 0，生成停止信号，运行模式从主机模式变为从机模式 0 从机模式 1 主机模式
4 TX	发送模式选择。TX 位选择主从传输的方向。在主模式中，TX 位应根据所需传输类型进行设置。因此对于地址周期来说，TX 位始终很高。当作为从机时，TX 位应由软件根据状态寄存器中的 SRW 位进行设置。 0 接收 1 发送
3 TXAK	发送应答使能。在主从接收器的数据应答周期中，该位设置驱动输出到 SDA 的值。 0 收到一个数据字节后，向总线发送应答信号 1 不发送应答信号响应
2 RSTA	重复开始。向该位写入 1 产生重复启动条件，假设它是当前主机的情况。该位始终读取为 0。在错误时间试图重复会导致仲裁丢失。

### 11.4.4 IIC 状态寄存器 (IICS)

	7	6	5	4		3	2	1	0
R	TCF	IAAS	BUSY	ARBL	0	SRW		IICIF	RXAK
W									
复位	1	0	0	0	0	0	0	0	0
	= 不执行或保留								

图 11-6. IIC 状态寄存器 (IICS)

表 11-6. IICS 字段描述

字段	描述
7 TCF	传输完成标志。该位在字节传输完成时置位。该位只有在 IIC 模块往返传输过程中或刚传输完成后才有效。TCF 位的清除通过在接收模式中读取 IICD 寄存器或在发送模式中写入 IICD。 0 传输正在进行 1 传输完成
6 IAAS	匹配从机。当主叫地址匹配已编程的从机地址或当设置了 GCAEN 位且收到通用呼叫时，设置 IAAS 位。写 IICC 寄存器则清除该位。 0 未寻址 1 已寻址从机
5 BUSY	总线忙。BUSY 位表示总线的状态，无论是处于从模式还是主模式。当检测到启动信号时设置 BUSY 位，当检测到停止信号时清除。 0 总线空闲 1 总线忙
4 ARBL	仲裁丢失。当仲裁丢失时，该位由硬件设置。ARBL 位必须通过由软件向其中写入 1 来清除。 0 标准总线运行 1 仲裁丢失
2 SRW	从机读 / 写。当寻址到从机时，SRW 位指示发送给主叫地址主机的 R/W 命令位的值。 0 从机接收，主机写入从机 1 从机发送，主机从从机中读取
1 IICIF	IIC 中断标记。当有中断等待时，设置 IICIF 位。该位必须通过由软件向其中写入 1 来清除。以下事件可以设置 IICIF 位： <ul style="list-style-type: none"><li>• 一个字节传输完成</li><li>• 从机地址匹配主叫地址</li><li>• 仲裁丢失</li></ul> 0 无等待中断 1 有等待中断
0 RXAK	接收确认。当 RXAK 位低时，表示向总线传输一个字节数据完成后收到应答信号。如果 RXAK 位高，表示没有检测到应答信号。 0 收到应答 1 未收到应答

#### 11.4.5 IIC 数据 I/O 寄存器 (IICD)



图 11-7. IIC 数据 I/O 寄存器 (IICD)

表 11-7. IICD 字段描述

字段	描述
7-0 DATA	数据—在主传输模式中，当数据写入 IICD 时，数据传输被发起。最高位先传送。在主接收模式中，读取该寄存器启动接收下一字节的数据。

### 注意

当退出主接收模式时，应当在读取 IICD 寄存器之前切换 IIC 模式，以防止意外启动主接收数据传输。

在从机模式中，地址匹配完成后提供相同功能。

IICC 中的 TX 位必须正确反应主从模式的传输方向，以便开始传输。例如，如果为 IIC 配置了主发送，但需要的是主接收，读取 IICE 就不会启动接收。

当 IIC 配置为主接收或从接收模式时，读取 IICD 将返回所接收的最后一个字节。IICD 不反应 IIC 总线上发送的每个字节，软件也不能通过回读的方式确认字节是否已经正确写入 IICD。

在主传输模式中，置位 MST 后写入 IICD 的第一个字节数据用于传输地址，应当包含主叫地址（位 7 或位 1）及所需 R/W 位（在位 0 位置）。

#### 11.4.6 IIC 控制寄存器 2 (IICC2)

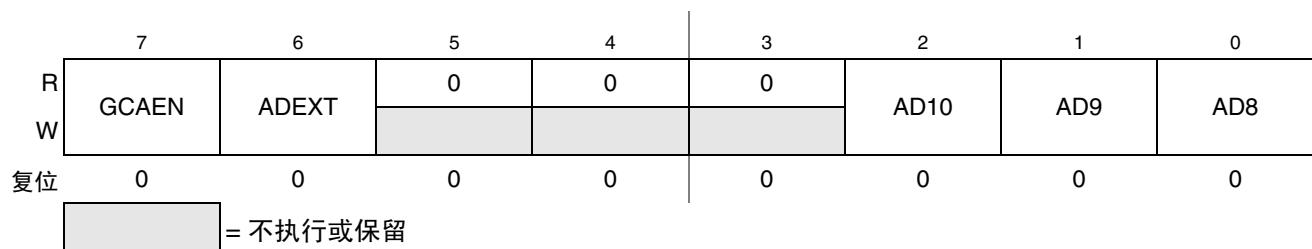


图 11-8. IIC 控制寄存器 (IICC2)

表 11-8. IICC2 字段描述

字段	描述
7 GCAEN	通用呼叫地址使能。 GCAEN 位使能或禁止通用呼叫地址。 0 通用呼叫地址禁止 1 通用呼叫地址使能
6 ADEXT	地址扩展名。 ADEXT 位控制着从机地址使用的位的数量。 0 7 位地址模式 1 10 位地址模式
2-0 AD[10:8]	AD[10:8] 从机地址。 AD[10:8] 字段包含 10 位地址模式中从机地址的高三位。当 ADEXT 位设置时，本字段才有效。

## 11.5 功能描述

本小节详细描述了 IIC 模块的全部功能。

### 11.5.1 IIC 协议

IIC 总线系统为数据传输使用串行数据线（SDA）和串行时钟线（SCL）。与其连接的所有器件必须具有开漏或开极输出。逻辑与功能通过外部上拉电阻在两条线上执行。这些电阻的值与系统相关。

一般地，标准通信由以下四部分组成：

- 启动信号
- 从机地址发送
- 数据传输
- 停止信号

停止信号不应与 CPU 停止指令相混淆。IIC 总线系统通信将在后面进行简要地描述，并在图 11-9 中进行了阐释。

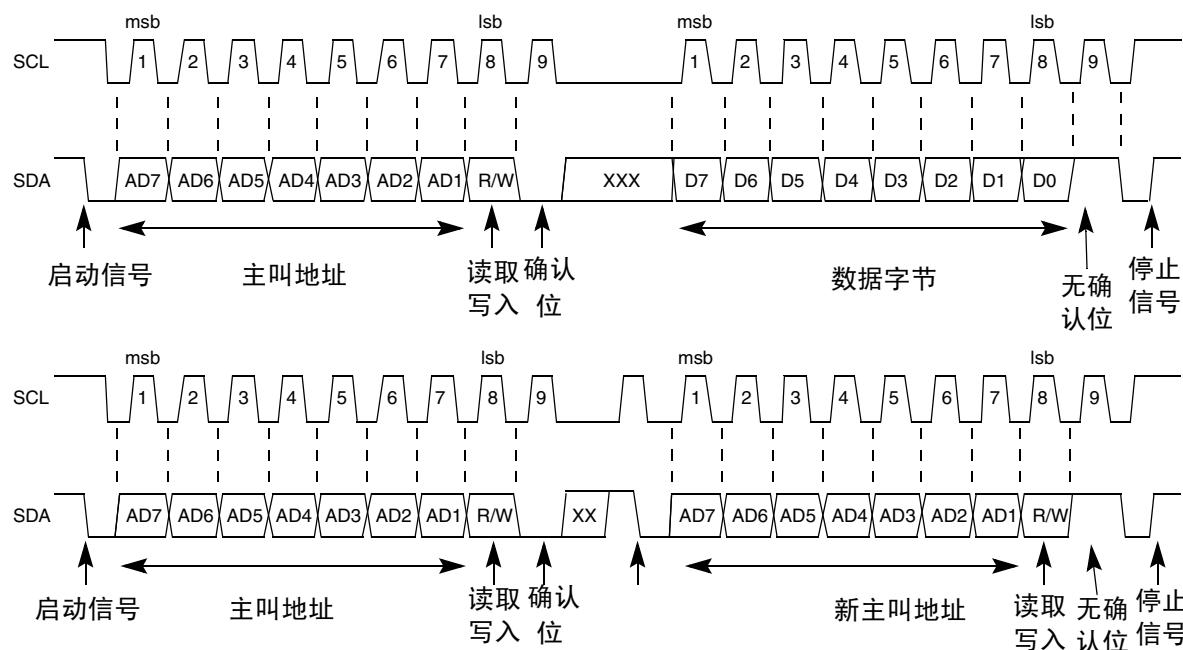


图 11-9. IIC 总线传输信号

### 11.5.1.1 启动信号

当总线空闲时，没有主机占用总线（**SCL** 和 **SDA** 线处于逻辑高电平），主机可以通过发送启动信号发起通信。如图 11-9 所示，启动信号定义为，当 **SCL** 在高电平时，**SDA** 从高到低的跳变。该信号表示开始新的数据传输（每次数据传输可能包含几个字节的数据），并使所有从机退出空闲状态。

### 11.5.1.2 从机地址发送

启动信号发出后传输的第一个字节数据是主机发送的从机地址。这是 **R/W** 位之前的 7 位主叫地址。**R/W** 位告知从机数据传输的方向。

**1** = 读取传输，从机向主机传输数据

**0** = 写入传输，主机向从机传输数据

当主机发送的地址与某从机地址匹配时，此从机发回应答位进行响应。通过将 **SDA** 第 9 个时钟拉低实现（见图 11-9）。

系统中不能有两个地址相同的从机。如果 IIC 模块是主机，它就不能发送与其自己的从机地址相同的地址。IIC 不能同时既是主机又是从机。但是，如果仲裁在寻址周期中丢失，IIC 就重新返回到从机模式并正确运行，即便它正被另一个主机寻址。

### 11.5.1.3 数据传输

成功实现从机寻址之前，就可以按照主叫主机发送的 **R/W** 位指定的方向逐字节地进行数据传输。

地址周期后的所有传输都被称为数据传输，即使它们包含从机的子地址报文。

每个数据字节的长度均为 8 位。只有当 **SCL** 处于低时数据才可以更改，如果 **SCL** 处于高位，那么它必须保持稳定，如图 11-9 所示。**SCL** 的一个时钟脉冲传输一个数据位，最高位被首先传输。每个数据字节后面都有一个第 9（确认）位，该位由从机发出信号，通过把 **SDA** 拉低到第 9 个时钟实现。总之，一个完整数据传输需要 9 个时钟脉冲。

如果从机在第 9 个位时间时未应答主机，从机必须保留 **SDA** 线在高电平。主机将未接收应答信号解释为不成功的数据传输。

如果主机接收器在一个数据字节传输后未应答从机发送器，从机把这种情况理解为数据传输结束，并释放 **SDA** 线。

对于这两种情况，数据传输都被中止，主机会进行以下两种操作之一：

- 发送停止信号，放弃总线
- 发送重复启动信号，开始新呼叫

### 11.5.1.4 停止信号

主机可以通过发送停止信号终止通信，以释放总线。然而，主机可以直接发送启动信号和呼叫命令，而无需首先发送停止信号。这被称为重复启动。停止信号的定义是 SCL 在逻辑 1 位置时的从低到高的 SDA 跳变（见图 11-9）。

即便从机发出一个应答，主机也可以发送停止信号，此时从机必须释放总线。

### 11.5.1.5 重复启动信号

如图 11-9 所示，重复启动信号是在无需首先生成停止信号以终止通信的情况下发送的信号。该信号由主机用来与另外一个从机进行通信，或者在不同模式（传输 / 接收模式）中与同一从机进行通信，而不需要释放总线。

### 11.5.1.6 仲裁程序

IIC 总线是真正的多主控总线，允许一个以上的主机连接到 IIC 上。如果两个甚至多个主机试图同时控制总线，那么就由时钟同步过程来决定总线时钟，总线低周期等于最长的时钟低段，总线高周期等于最短的时钟高段。竞争主机的相对优先级由数据仲裁过程确定，如果一个总线主机发出逻辑 1，而另一个发出逻辑 0，那么前者就丢失仲裁。失败的一方立即切换到从机接收模式，停止驱动 SDA 输出。在这种情况下，从主模式到从模式的转换不会生成停止条件。与此同时，会通过硬件设置一个状态位，表示仲裁丢失。

### 11.5.1.7 时钟同步

因为线与逻辑在 SCL 线上执行，所以 SCL 上的从高到低的跳变会影响连接到总线上的所有器件。节点开始计数它们的低态周期，当节点的时钟进入低态后，它将一直保持 SCL 线的低态，直到达到时钟高态。然而，如果另一个节点时钟仍处于低态时段，这时此节点时钟从低到高的变化就不会改变 SCL 线的状态。因此，经过同步的时钟 SCL 由具有最长低态周期的节点保持。低态周期较短的节点在该时段进入高态等待（见图 11-10）。当所有有关节点均已完成它们的低态周期时，同步时钟 SCL 线被释放和拉高。这样，节点时钟和 SCL 线的状态之间就没有任何差异，所有节点开始计数它们的高态周期。第一个完成其高态周期的节点再次拉低 SCL 线。

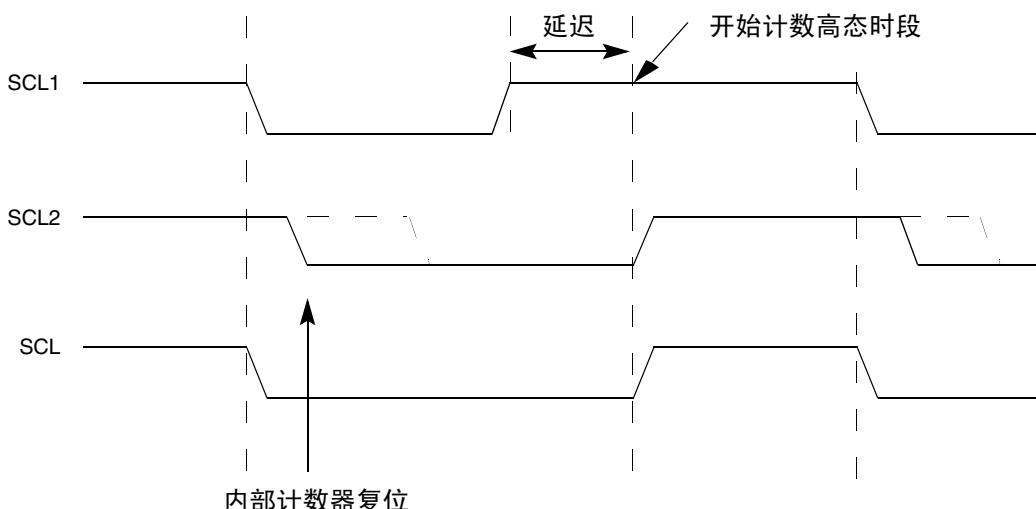


图 11-10. IIC 时钟同步

### 11.5.1.8 握手

时钟同步机制可以用作数据传输中的握手。在完成一个字节的传输（9 个位）后，从机可以保持 SCL 低位。在这种情况下，它会暂停总线时钟，强迫主机时钟进入等待状态，直到从机释放 SCL 线。

### 11.5.1.9 时钟延展

时钟同步机制可以被从机用于减缓传输的比特速率。在主机已经拉低 SCL 后，从机可以继续拉低 SCL 一定时间，然后再释放它。如果从机 SCL 低态周期长于主机 SCL 低态周期，那么就会将 SCL 总线信号低态周期延展。

## 11.5.2 10 位地址

对于 10 位寻址，0x11110 用于地址首字节的前 5 位。10 位寻址传输过程中可能出现不同的读/写格式组合。

### 11.5.2.1 主发送器寻址从接收器

传输方向不变（见表 11-9）。当 10 位地址跟随开始信号发送时，每个从机会把该从机地址首字节的前 7 位与其自己的地址进行比较，并测试第 8 个位（R/W 方向位）是否为 0。一个以上的从机能够匹配并生成应答（A1）。然后，匹配的每个从机会把该从机地址第二个字节的 8 个位与其自己的地址进行比较。只有一个从机找到匹配并生成应答（A2）。匹配的从机与此主机通信，直到收到停止信号（P）或跟随着其他从机地址的重复开始信号（Sr）。

S	从机前面 7 位 11110 + AD10 + AD9	R/W 0	A1	从机前面 7 位 AD[8:1]	A2	数据	A	...	数据	A/A	P
---	--------------------------------	----------	----	---------------------	----	----	---	-----	----	-----	---

表 11-9. 主发射器寻址 10 位地址的辅接收器

在主机发送器已经发送了 10 位地址的第一个字节后，从机接收器产生 IIC 中断。软件必须确保 IICD 的内容被忽略，且不作为该中断的有效数据对待。

### 11.5.2.2 主接收器寻址从发送器

如果传输方向被第二个 R/W 位改变（见表 11-10）。一直到应答位 A2（包括应答位 A2）前，该流程与主发送器寻址从接收器中的描述都相同。在重复开始条件（Sr）后，匹配的从接收器记得它以前曾被寻址过。从接收器然后就检查 Sr 后的从机地址首字节的前 7 位是否与开始条件（S）后的前 7 位相同，并测试第 8（R/W）个位是否为 1。如果匹配，从接收器就认为它已经被确认为发送器，并生成应答 A3。从发送器保持匹配，直到收到停止信号（P）或跟随着其他从机地址的重复开始信号（Sr）。

重复开始条件（Sr）之后，所有其他从机也将从机地址首字节的前 7 位与它们自己的地址进行比较，测试第 8（R/W）位。然而，这些从机都没有匹配，因为 R/W=1（用于 10 位器件）或 11110XX 从机地址（用于 7 位节点）不匹配。

S	从机前 7 位 11110 + AD10 + AD9	R/W 0	A1	从机第二个字节 AD[8:1]	A2	Sr	从机前 7 位 11110 + AD10 + AD9	R/W 1	A3	数据	A	...	数据	A	P
---	-------------------------------	----------	----	--------------------	----	----	-------------------------------	----------	----	----	---	-----	----	---	---

表 11-10. 主接收器寻址 10 位地址的从发射器

在主机发送器已经发送了 10 位地址的第一个字节后，从机接收器产生 IIC 中断。软件必须确保 IICD 的内容被忽略，且不作为该中断的有效数据对待。

### 11.5.3 通用呼叫地址

通用呼叫可以是 7 位地址或 10 位地址。如果设置了 GCAEM 位，IIC 就匹配通用呼叫地址及其自己的从机地址。当 IIC 响应通用呼叫时，它用作从接收器，且在地址周期后设置 IAAS 位。传输完首字节后，软件必须读取 IICD 寄存器，以确定是地址匹配其自己的从机还是通用呼叫。如果值为 00，匹配是通用呼叫。如果 GCAEN 位为 0，IIC 则通过不发送应答的方式忽略通用呼叫地址提供的任何数据。

## 11.6 复位

IIC 在复位后被禁止，IIC 不能引起 MCU 复位。

## 11.7 中断

IIC 只产生一个中断。

假设设置了 IICIE 位，当发生表 11-11 中的任意一个事件时，IIC 就生成中断。中断由位 IICIF (IIC 状态寄存器的位) 驱动，用位 IICIE (IIC 控制寄存器的位) 屏蔽。IICIF 位必须通过软件在中断程序中向其写入 1 来清除。您可以通过读取状态寄存器确定中断类型。

表 11-11. 中断摘要

中断源	状态	标记	本地使能
完成 1 字节传输	TCF	IICIF	IICIE
匹配到收到的主叫地址	IAAS	IICIF	IICIE
仲裁丢失	ARBL	IICIF	IICIE

### 11.7.1 字节传输中断

TCF (传输完成标记) 位在第 9 时钟的下降边沿设置，表示字节传输完成。

### 11.7.2 地址检测中断

当主叫地址匹配已编程的从机地址 (IIC 地址寄存器) 或者当设置了 GCAEN 位且收到通用呼叫时，就设置状态寄存器中的 IAAS 位。假设设置了 IICIE，CPU 就被中断。CPU 必须检查 SRW 位并相应设置其 Tx 模式。

### 11.7.3 仲裁丢失中断

IIC 是真正的多主控总线，允许一个以上的主机连接到 IIC 上。如果两个甚至多个主机试图同时控制总线，那么竞争主机的相对优先级就由数据仲裁流程决定。IIC 模块在丢失数据仲裁时触发中断，同时状态寄存器中的 ARBI 位置位。

当出现以下情况时，仲裁丢失：

- 当主机在地址或数据传输周期中驱动为高时，SDA 采样为低。
- 当主机在数据接收周期的应答位期间驱动为高时，SDA 采样为低。
- 总线忙时尝试开始信号。
- 从模式下请求重复开始周期。
- 当主机未请求时检测到停止信号。

该位必须通过用软件在其中写入 1 来清除。

## 11.8 初始化 / 应用报文

### 模块初始化（从模块）

1. 写入: IICC2
  - 使能或禁止通用呼叫
  - 选择 10 位或 7 位寻址模式
2. 写入: IICA
  - 设置从机
3. 写入: IICC1
  - 使能 IIC 和中断
4. 初始化 RAM 变量 ( $\text{IICEN} = 1$ ,  $\text{IICIE} = 1$ ) 以便发送数据
5. 初始化用来实现图 11-12 所示程序的 RAM 变量

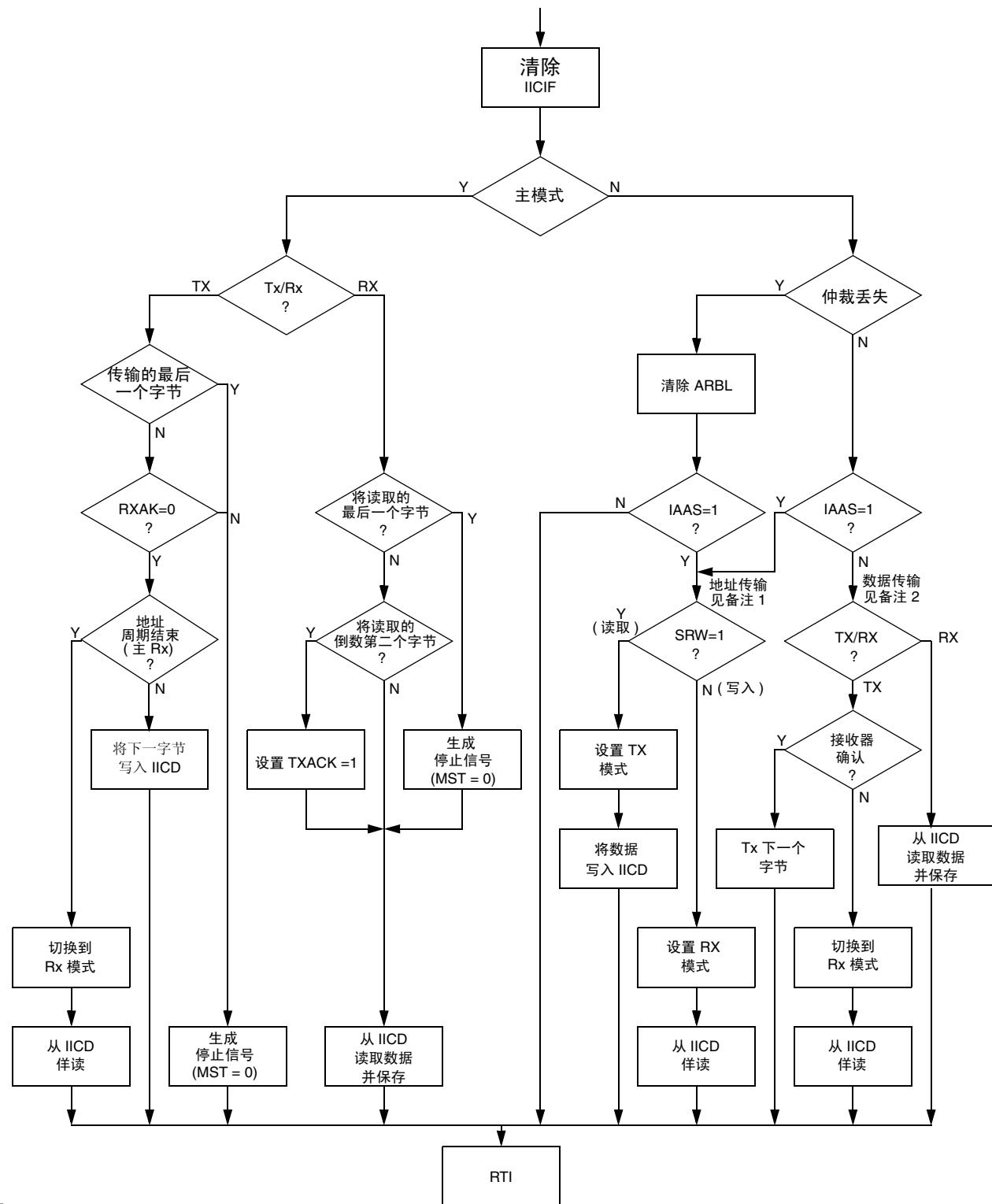
### 模块初始化（主机）

1. 写入: IICF
  - 设置 IIC 波特率（本章提供了示例）
2. 写入: IICC1
  - 使能 IIC 和中断
3. 初始化 RAM 变量 ( $\text{IICEN} = 1$ ,  $\text{IICIE} = 1$ ), 以便发送数据
4. 初始化用来实现图 11-12 所示程序的 RAM 变量
5. 写入: IIIC1
  - 使能 TX

### 寄存器模式

IICA	AD[7:1]							0
寻址为从机（在从模式中）时，模块响应该地址								
IICF	MULT	ICR						
波特率 = $\text{BUSCLK} / (2 \times \text{MULT} \times (\text{SCL DIVIDER}))$								
IICC1	IICEN	IICIE	MST	TX	TXAK	RSTA	0	0
模块配置								
IICS	TCF	IAAS	BUSY	ARBL	0	SRW	IICIF	RXAK
模块状态标记								
IICD	数据							
数据寄存器：通过写操作传输 IIC 数据阅读，读取 IIC 数据								
IICC2	GCAEN	ADEXT	0	0	0	AD10	AD9	AD8
地址配置								

图 11-11. IIC 模块快速启动



备注：

- 如果使能了通用呼叫，必须进行检查，以确定收到的地址是否为通用呼叫地址（0x00）。如果收到的地址是通用呼叫地址，那么通用呼叫必须由用户软件处理。
- 当使用 10 位寻址来寻址从器件时，从器件在扩展地址的首字节后发现中断。用户软件必须为该中断确保这一点，那就是忽略 IICD 的内容，且不把它作为有效数据传输对待。

图 11-12. 典型的 IIC 中断程序

# 第 12 章

## 飞思卡尔控制器局域网 (S08MSCANV1)

### 12.1 介绍

飞思卡尔控制器局域网（**MSCAN**）是一种通信控制器，它按照 1991 年 9 月定义的 **Bosch** 规范执行 CAN 2.0A/B 协议。为了全面了解 **MSCAN** 规范，我们建议首先阅读 **Bosch** 规范，熟悉本文档包含的一些条款和概念。

尽管 CAN 协议并非是汽车应用的专用协议，但它旨在满足车辆串行数据总线的特定规范，如实时处理、车辆在 **EMI** 环境中的可靠运行、成本高效性和所需带宽等。

**MSCAN** 使用先进的缓冲器安排，实现了可预测的实时性，并简化了应用软件。

**MSCAN** 模块应用在 MC9S08DZ60 系列的所有器件上。

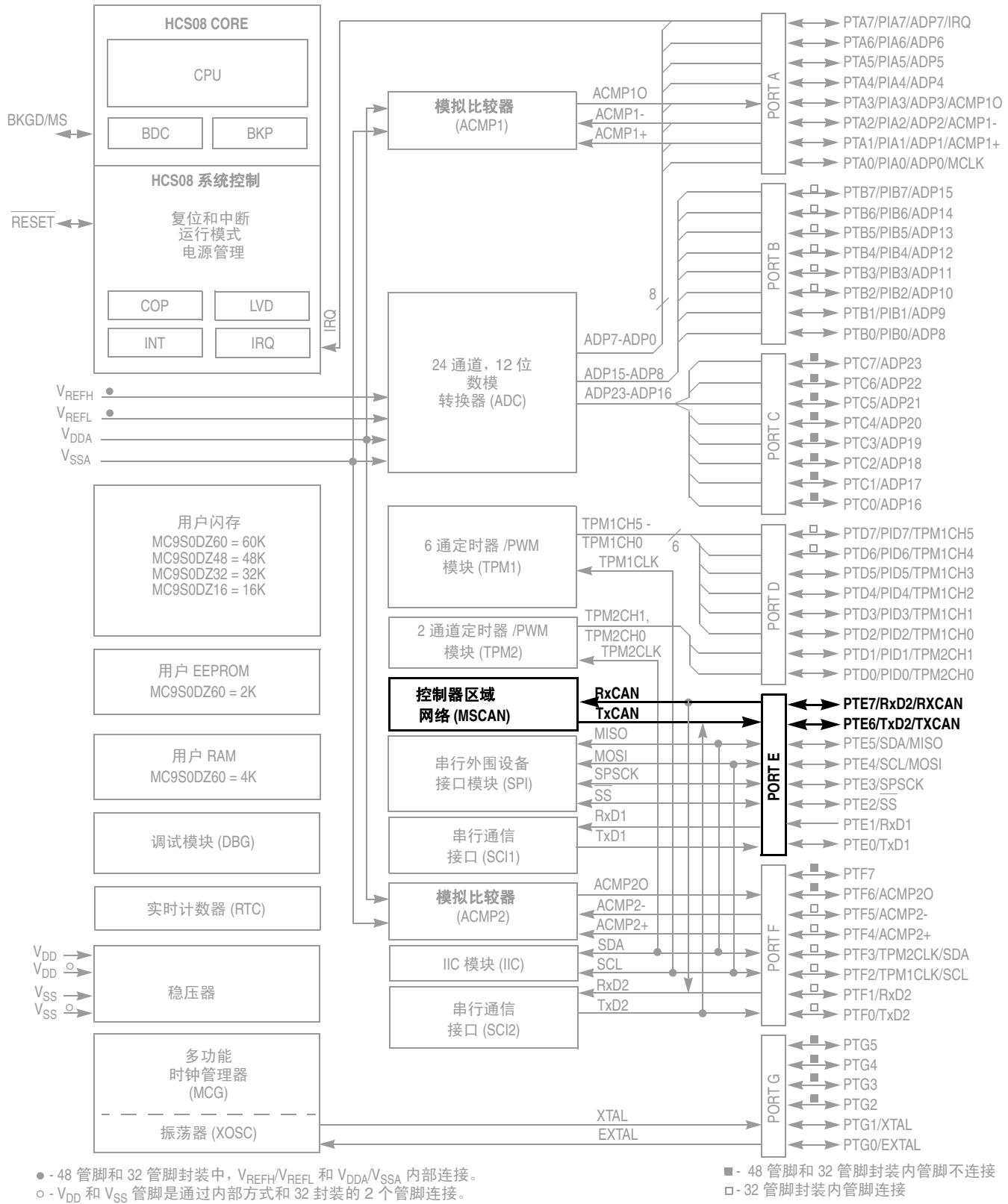


图 12-1. MC9S08DZ60 结构图

## 12.1.1 特性

MSCAN 的基本特性如下：

- 实施 CAN 协议—2.0A/B 版
  - 标准和扩展数据帧
  - 0-8 字节数据长度
  - 高达 1 Mbps<sup>1</sup> 的可编程比特率<sup>1</sup>
  - 支持远程帧
- 5 个具有 FIFO 存储机制的接收缓冲器
- 3 个具有使用“本地优先”概念的内部优先顺序的发送缓冲器 t
- 灵活可掩码标识符滤波器支持 2 个全尺寸（32 位）扩展标识符滤波器或 4 个 16 位滤波器或 8 个 8 位滤波器
- 集成低通滤波器的可编程唤醒功能 r
- 可编程环回模式支持自测操作
- 可编程监听模式用于 CAN 总线监控
- 可编程总线脱离恢复功能
- 独立的信号和中断功能适用于所有 CAN 接收器和发射器错误状态（警报、错误严重状态、总线脱离）
- 可编程 MSCAN 时钟源，采用总线时钟或振荡器时钟
- 内部计时器提供给接收和发送的报文的时间标签
- 三种低功耗模式：睡眠、关机和 MSCAN 使能
- 配置寄存器的全局初始化

## 12.1.2 运行模式

以下运行模式是 MSCAN 的特定运行模式。详情 [12.5，“功能描述”](#)。

- 监听模式
- MSCAN 睡眠模式
- MSCAN 初始化模式
- MSCAN 关机模式
- 环回自测模式

---

<sup>1</sup>. Depending on the actual bit timing and the clock jitter of the PLL.

### 12.1.3 结构图

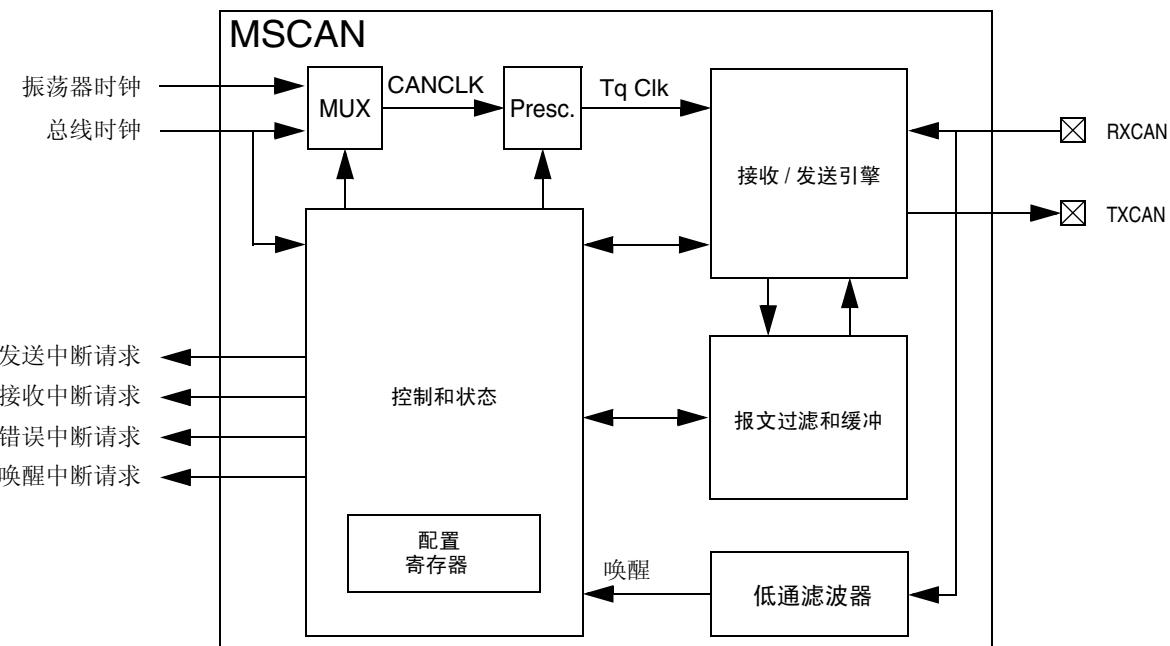


图 12-2. MSCAN 结构图

## 12.2 外部信号描述

MSCAN 使用两个外部管脚：

### 12.2.1 RXCAN — CAN 接收器输入管脚

RXCAN 是 MSCAN 接收器输入管脚。

### 12.2.2 TXCAN — CAN T 发射器输出管脚

TXCAN 是 MSCAN 发送器输出管脚。TXCAN 输出管脚代表 CAN 总线上的逻辑层：

0 = 显性状态

1 = 隐性状态

### 12.2.3 CAN 系统

图 12-3. 显示了一个具有 MSCAN 的典型 CAN 系统。每个 CAN 节点通过收发器物理连接到 CAN 总线线路。e. 收发器能够驱动 CAN 总线所需的大电流，并具有对故障 CAN 或故障节点的电流保护。

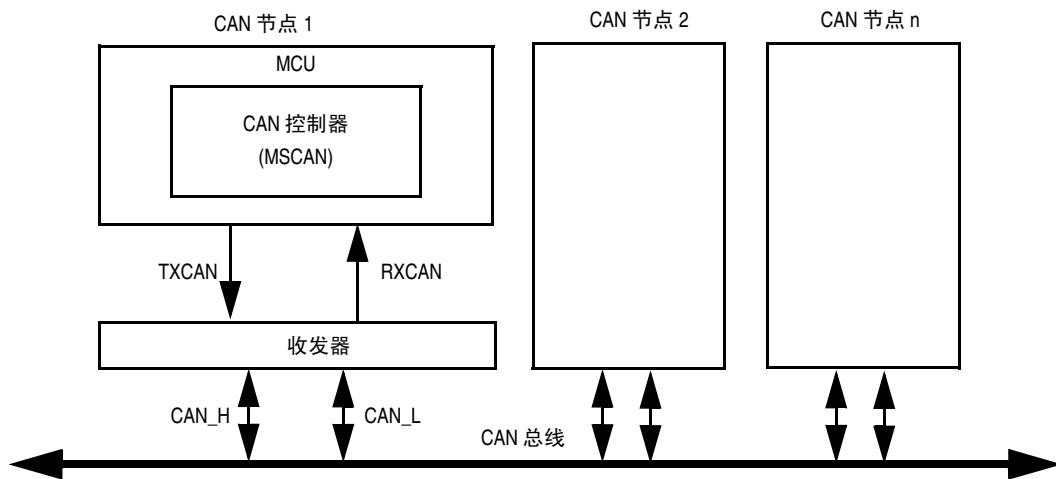


图 12-3. CAN 系统

## 12.3 寄存器定义

本节详细描述 MSCAN 模块中的所有寄存器和寄存器位。每个描述都包括带有相关图形编号的标准寄存器示意图。寄存器位和字段功能的详细说明在寄存器图后面，按位顺序。该模块中所有寄存器的所有位在寄存器读取过程中都与内部时钟完全同步。

### 12.3.1 MSCAN 控制寄存器 0 (CANCTL0)

The CANCTL0 寄存器提供了如下所述的 MSCAN 模块的各种位控制。

	7	6	5	4	3	2	1	0
R	RXFRM	RXACT	CSWAI	SYNCH	TIME	WUPE	SLPRQ	INITRQ
W								
复位：	0	0	0	0	0	0	0	1
	= Unimplemented							

图 12-4. 控制寄存器 0 (CANCTL0)

#### 注意

当初始化模式处于有效 (INITRQ = 1 and INITAK = 1) 时，除 WUPE、INITRQ 和 SLPRQ 外的所有 CANCTL0 寄存器位都处于复位状态。只要退出初始化模式 (INITRQ = 0, INITAK = 0)，该寄存器可以再次写入。

读取：任何时间

写入：退出初始化模式的任何时间；例外是只读 RXACT、SYNCH、RXFRM（只由该模块设置）和 INITRQ（也可以在初始化模式中写入）。

表 12-1. CANCTL0 寄存器字段描述

字段	描述
7 RXFRM <sup>1</sup>	已收到帧标记—该位是只读和只清除位。当接收器正确收到有效报文（独立于滤波器配置）时，设置该位。设置后，该位一直保持设置，直到通过软件或复位将其清除。清除通过写入 1 完成。写 0 被忽略。该位在环回模式中无效。 0 自上次清除该标记以来未收到有效报文 1 自上次清除该标记以来收到有效报文
6 RXACT	接收器使能状态—该只读标记表示 MSCAN 正在接收报文。该标记由接收器前端控制。该位在环回模式中无效。 0 MSCAN 正在发送或空闲 <sup>2</sup> 1 MSCAN 正在接收报文（包括仲裁丢失时） <sup>2</sup>
5 CSWAI <sup>2</sup>	在等待模式中 CAN 停止—设置此位，可以在等待模式中通过禁止 MSCAN 模块与 CPU 总线接口的所有时钟而降低功耗。 0 在等待模式中，CAN 模块不受影响 cei 1 等待模式中，CAN 模块停止计时
4 SYNCH	同步状态—该只读标记显示 MSCAN 是否与 CAN 总线同步，是否能够参与通信流程。其设置和清除通过 MSCAN 进行。 0 MSCAN 与 CAN 总线不同步 1 MSCAN 与 CAN 总线同步 s
3 TIME	计时器使能—该位使能内部 16 位字宽自由运行计时器，由位时钟速率计时。如果计时器被使能，16 位时间标签将分配给有效 TX/RX 缓冲器内的每条发送 / 接收报文。一旦报文在 CAN 总线上确认，时间标签将被写入适当缓冲器（参见 12.4，“报文存储模式”）的最高字节（0x000E, 0x000F）。禁止时，内部计时器复位（所有位都设置为 0）。该位在初始化模式中保持低。 0 禁止内部 MSCAN 计时器 r 1 使能内部 MSCAN 计时器 r
2 WUPE <sup>3</sup>	唤醒使能—当检测到 CAN 上有流量时（参见 12.5.5.4，“MSCAN 睡眠模式”），该配置位能够让 MSCAN 从睡眠模式中重启。为了让所选功能发挥作用，在该位进入睡眠模式前必须进行配置。 0 唤醒禁止—MSCAN 忽略 CAN 上的流量 1 唤醒使能—MSCAN 能够重启

表 12-1. CANCTL0 寄存器字段描述 (continued)

字段	描述
1 SLPRQ <sup>4</sup>	睡眠模式请求—该位请求 MSCAN 进入睡眠模式，这是一个内部节电模式（参见 12.5.5.4，“MSCAN 睡眠模式”）。当 CAN 总线空闲时，也就是说该模块不接收任何报文且所有发送缓冲器空，睡眠模式请求被受理。通过设置 SLPAK = 1（参见 12.3.2，“控制寄存器 1 (CANCTL1)”），表示该模块进入睡眠模式。当设置了 WUPIF 标记时（参见 12.3.4.1，“MSCAN 接收器标志寄存器 (CANRFLG)”），不能设置 SLPRQ。睡眠模式维持有效，直到 SLPRQ 被 CPU 清除或者根据 WUPE 的设置，MSCAN 检测到 CAN 总线上有有效并自行清除。 0 运行中 — MSCAN 正常工作 1 睡眠模式请求—当 CAN 总线空闲时 MSCAN 进入睡眠模式
0 INITRQ <sup>5,6</sup>	INITRQ6、7 初始化模式请求—当 CPU 设置该位时，MSCAN 切换至初始化模式（参见 12.5.5.5，“MSCAN I 初始化模式”）。任何正在进行的发送或接收都将被中止，与 CAN 总线的同步也丢失。通过设置 INITAK = 1（见 12.3.2 小节“MSCAN 控制寄存器 1 (CANCTL1)”), 表示该模块进入初始化模式。 以下寄存器进入其硬复位状态并恢复它们的默认值：CANCTL08、CANRFLG9、CANRIER10、CANTFLG、CANTIER、CANTARQ、CANTAAK 和 CANTBSEL。 μ SCAN 处于初始化模式 (INITRQ = 1, INITAK = 1) 时，寄存器 CANCTL1、CANBTR0、CANBTR1、CANIDAC、CANIDAR0-7 和 CANIDMR0-7 只能通过 CPU 写入。错误计数器的值不受初始化模式的影响。 当该位通过 CPU 清除时，MSCAN 重启，然后试图与 CAN 总线同步。如果 MSCAN 未处于总线脱离状态，它在 CAN 总线上出现 11 个连续隐性位后同步。如果 MSCAN 处于总线脱离状态，它将继续等待 11 个连续隐性位重复出现 128 次。 只有当退出初始化模式后，才可以在 CANCTL0、CANRFLG、CANRIER、CANTFLG 或 CANTIER 中写入其他位，这时 INITRQ = 0, INITAK = 0。 0 正常运行 1 MSCAN 处于初始化模式

1 要设置该位，MSCAN 必须处于正常模式。

2 如需了解发送器和接收器状态的详细定义，（参见 12.5.5.2，“等待模式中的操作”）和 12.5.5.3，“停止模式中的操作”）。

3 为了防止意外违反 CAN 协议，当 CPU 进入等待 (CSWAI = 1) 或停止模式（参见 12.3.5，“MSCAN 接收器中断使能寄存器 (CANRIER)”），立即强制 TXCAN 管脚进入隐性状态。

4 如果需要从停止或等待模式中进行恢复的机制，CPU 必须确保 WUPE 位和 WUPIE 唤醒中断使能位（参见 12.3.5，“MSCAN 接收器中断使能寄存器 (CANRIER)”）(CANRIER) 被使能。

5 在 MSCAN 进入睡眠模式 (SLPRQ = 1, SLPAK = 1) 前，CPU 不能清除 SLPRQ。

6 在 MSCAN 进入初始化模式 (INITRQ = 1, INITAK = 1) 前，CPU 不能清除 INITRQ。

### 12.3.2 控制寄存器 1 (CANCTL1)

CANCTL1 寄存器如下所述提供了 MSCAN 模块的各种控制位和握手状态报文。

	7	6	5	4	3	2	1	0
R	CANE	CLKSRC	LOOPB	LISTEN	BORM	WUPM	SLPAK	INITAK
W								
复位：	0	0	0	1	0	0	0	1
	[Unimplemented]							

图 12-5. 控制寄存器 1(CANCTL1)

读取：任何时间

写入：当 INITRQ = 1 和 INITAK = 1 时的任何时间， CANE 例外，可以在正常情况下写入一次，以及当 MSCAN 处于初始化模式 (INITRQ = 1, INITAK = 1) 的特殊系统运行模式时任何时间写入。

表 12-2. 寄存器字段描述

字段	描述
7 CANE	<b>MSCAN 使能</b> 0 MSCAN 模块禁止 1 MSCAN 模块使能
6 CLKSRC	<b>MSCAN 时钟源</b> — 该位定义 MSCAN 模块的时钟源 (仅适用于具有时钟发生模块的系统; <a href="#">12.5.3.3, “时钟系统”</a> 和 <a href="#">图 12-42., “MSCAN 时钟机制”</a> ). 0 MSCAN 时钟源是振荡器时 1 MSCAN 时钟源是总线时钟
5 LOOPB	环回自测模式 — 当设置了该位时，MSCAN 执行可用于自测操作的内部环回。T 发送器的位流输出从内部流回接收器。 <a href="#">12.5.4.6, “环回自测模式”</a> . 0 环回自测禁止 1 环回自测使能
4 LISTEN	° 监听模式 — 该位把 SCAN 配置为 CAN 总线监控器。当设置了 LISTEN 时，会收到带有匹配 ID 的所有有效 CAN，但不发出确认或错误帧 (参见 <a href="#">12.5.4.4, “监听模式”</a> )。此外，错误计数器停止计数。监听模式可以支持需要“热插拔”或“吞吐量分析”的应用。当监听模式处于有效状态时，MSCAN 不能发送任何报文。 0 正常运行 1 监听模式使能
3 BORM	总线脱离恢复模式 — 该位配置 MSCAN 的总线关断恢复模式。更多报文总线脱离恢复模式 — 该位配置 MSCAN 的总线关断恢复模式。更多报文 <a href="#">12.6.2, “总线脱离恢复”</a> 。 0 自动总线脱离恢复 (参见 Bosch CAN 2.0A/B 协议规范) 1 用户请求的总线脱离恢复
2 WUPM	唤醒模式 — 如果 CANCTL0 中的 WUPE 被使能，该位决定是否应用集成低通滤波器来防止 MSCAN 出现假唤醒 (参见 <a href="#">12.5.5.4, “MSCAN 睡眠模式”</a> ) 0 MSCAN 被 CAN 总线上的任意显性信号唤醒 1 MSCAN 只有在 CAN 总线上的显性脉冲长度为 Twup 时才唤醒。

表 12-2. 寄存器字段描述

字段	描述
1 SLPAK	睡眠模式确认—该标记显示 MSCAN 模块是否已经进入睡眠模式(参见 12.5.5.4, “MSCAN 睡眠模式”)。它用作 SLPRQ 睡眠模式请求的握手标志。t。 当 SLPRQ = 1、SLPAK = 1 时, 睡眠模式是有效的。根据 WUPE 设置, 如果在处于睡眠模式检测到 CAN 总线有信号, MSCAN 将清除该标志。CPU 清除 SLPRQ 位也将复位 SLPAK 位。 0 正在运行—MSCAN 正常运行 1 睡眠模式使能—MSCAN 已经进入睡眠模式
0 INITAK	初始化模式确认—该标志显示 MSCAN 模块是否处于初始化模式(参见 12.5.5.5, “MSCAN I 初始化模式”)。它用作 INITRQ 初始化模式请求的握手标志。当 INITRQ = 1, INITAK = 1 时, 初始化模式被使能。当 MSCAN 处于初始化模式时, 寄存器 CANCTL1、CANBTR0、CANBTR1、CANIDAC、CANIDAR0 - CANIDAR7 和 CANIDMR0 - CANIDMR7 只能通过 CPU 写入 0 正在运行—MSCAN 正常运行 1 初始化模式使能—MSCAN 处于初始化模式

### 12.3.3 MSCAN 总线计时寄存 0 (CANBTR0)

CANBTR0 寄存器配置 MSCAN 模块的各种 CAN 总线计时参数。

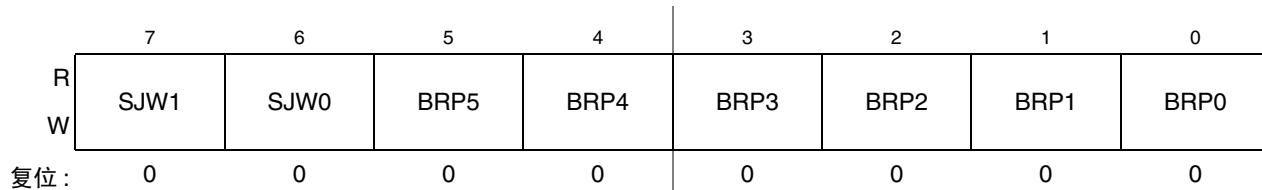


图 12-6. MSCAN 总线计时寄存器 0(CANBTR0)

读取: 任何时间

写入: 处于初始化模式 (INITRQ = 1, INITAK = 1) 的任何时间

表 12-3. CANBTR0 寄存器字段描述

字段	描述
7:6 SJW[1:0]	同步跳转宽度—同步跳转宽度决定要实现 CAN 总线上的数据传输重新同步, 一个位可以缩短或延长的时间冲量 ( $T_q$ ) 的最大值(参见表 12-4)。
5:0 BRP[5:0]	波特率预分频器—该位确定用来构建位计时的时间冲量 ( $T_q$ ) 时钟(参见表 12-5)。

表 12-4. 同步跳转宽度

SJW1	SJW0	同步跳转宽度
0	0	1 Tq 时钟周期
0	1	2 Tq 时钟周期
1	0	3 Tq 时钟周期
1	1	4 Tq 时钟周期

表 12-5. 波特率预分频器

BRP5	BRP4	BRP3	BRP2	BRP1	BRP0	预分频器值 (P)
0	0	0	0	0	0	1
0	0	0	0	0	1	2
0	0	0	0	1	0	3
0	0	0	0	1	1	4
:	:	:	:	:	:	:
1	1	1	1	1	1	64

### 12.3.4 MSCAN 总线计时寄存器 (CANBTR1)

T 每个标志只有在造成该设置的条件不再有效时才能通过软件清除（将 1 写入相应位位置）。每个标志在 CANRIER 寄存器中都有相关的中断使能位。

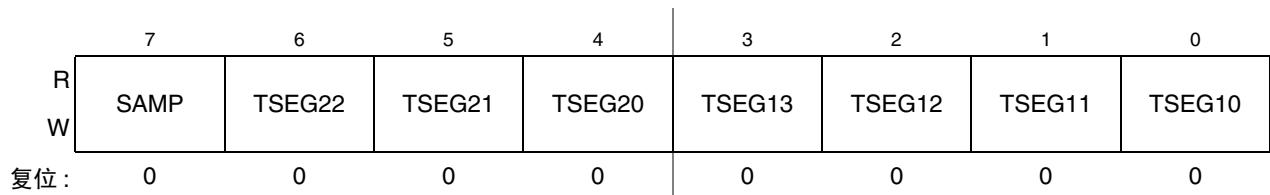


图 12-7. MSCAN 总线计时寄存器 1 (CANBTR1)

读取：任何时间

写入：处于初始化模式 (INITRQ = 1 and INITAK = 1)

表 12-6. CANBTR1 寄存器字段描述

字段	描述
7 SAMP	采样 — 该位确定每位时间所采集的 CAN 总线样本数量 0 每位 1 个样本 1 每位 3 个样本 1。 如果 SAMP = 0, 得到的位值等于采样点上定位的单个位的值。如果 SAMP = 1, 得到的位值通过在总共三个采样点上使用多数规则来决定。要实现更高比特速率, 建议每个位时间只采集一个样本 (SAMP = 0)。).
6:4 TSEG2[2:0]	TSEG2[2:0] 时间段 2—一位时间内的时间段固定每个位时间的时钟周期数和采样点的位置 (参见图 12-43). 时间段 2 (TSEG2) 值可以如表 12-7 所示进行编程。
3:0 TSEG1[3:0]	时间段 1—一位时间内的时间段固定每个位时间的时钟周期数和采样点的位置 (参见图 12-43). 时间段 1 (TSEG1) 值可以如表 12-8 所示进行编程。.

表 12-7. 时间段 2 值

TSEG22	TSEG21	TSEG20	时间段 2
0	0	0	1 Tq 时钟周期 <sup>1</sup>
0	0	1	2 Tq 时钟周期
:	:	:	:
1	1	0	7 Tq 时钟周期
1	1	1	8 Tq 时钟周期

<sup>1</sup> This setting is not valid. Please refer to 表 12-35 for valid settings.

表 12-8. 时间段 1 值

TSEG13	TSEG12	TSEG11	TSEG10	时间段 1
0	0	0	0	1 Tq c 时钟周期 <sup>1</sup>
0	0	0	1	2 Tq 时钟周期 <sup>1</sup>
0	0	1	0	3 Tq 时钟周期 <sup>1</sup>
0	0	1	1	4 Tq 时钟周期
:	:	:	:	:
1	1	1	0	15 Tq 时钟周期
1	1	1	1	16 Tq 时钟周期

<sup>1</sup> 该设置无效, 请参见表 12-35 查看有效设置。

位时间由振荡器频率、波特率预分频器和每位的时间冲量 (Tq) 数量确定 (如表 12-7 和表 12-8)。

等式 12-1

$$\text{Bit Time} = \frac{(\text{Prescaler value})}{f_{\text{CANCLK}}} \cdot (1 + \text{TimeSegment1} + \text{TimeSegment2})$$

### 12.3.4.1 MSCAN 接收器标志寄存器 (CANRFLG)

每个标志只有在造成该设置的条件不再有效时才能通过软件清除（将 1 写入相应位位置）。每个标志在 CANRIER 寄存器中都有相关的中断使能位。

	7	6	5	4		3	2	1	0
R	WUPIF	CSCIF	RSTAT1	RSTAT0	TSTAT1	TSTAT0	OVRIF	RXF	
W									
复位：	0	0	0	0	0	0	0	0	0
		= 不执行							

图 12-8. MSCAN 接收器标志寄存器 (CANRFLG)

#### 注意

当初始化模式处于有效状态时 CANRFLG 寄存器保持复位状态 1 (INITRQ = 1, INITAK= 1)。一旦退出初始化模式，该寄存器就可以重新写入 (INITRQ = 0 and INITAK = 0)。

读取：任何时间

写入：退出初始化模式的任何时间，除非 RSTAT[1:0] 和 TSTAT[1:0] 标志是只读；写入 1 表示清除标志，写入 0 表示忽略标志。

表 12-9. CANRFLG 寄存器字段描 ^

字段	描述
7 WUPIF	唤醒中断标志—如果在处于睡眠模式时 MSCAN 检测到 CAN 总线上面有效 (参见 12.5.5.4, “MSCAN 睡眠模式” ) 且 CANCTL0 中的 WUPE = 1(参见 12.3.1, “MSCAN 控制寄存器 0 (CANCTL0)” ), 那么该模块将设置 WUPIF。如果未被屏蔽，当设置了该标志时有一个唤醒中断产生。 0 处于睡眠模式时未观察到唤醒有效 1 MSCAN 检测到 CAN 总线上有有效并请求唤醒
6 CSCIF	CAN 状态变化中断标志—当 MSCAN 由于发送错误计数器 (TEC) 和接收错误计数器的实际值而更改其当前 CAN 总线状态时，设置该标志。另外一个为 TEC/REC 分出几个独立段的 4 位 (RSTAT[1:0]、TSTAT[1:0]) 状态寄存器告知系统实际的 CAN 总线状态 (参见 12.3.5, “MSCAN 接收器中断使能寄存器 (CANRIER)” )。如果未被屏蔽，当设置了该标志时有一个错误中断产生。CSCIF 提供一个拦截中断，这保证了接收器 / 发送器状态位 (RSTAT/TSTAT) 只有在无 CAN 状态变化中断产生时才进行更新。如果 TEC/REC 在 CSCIF 置位后更改其当前值，就会引起 RSTAT/TSTAT 位的其他状态变化。这些位会一直保持它们的状态，直到当前 CSCIF 中断被再次清除。 0 自上次中断以来 CAN 中线状态未发生变化 1 MSCAN 更改了当前 CAN 总线状态
5:4 RSTAT[1:0]	接收器状态位— 错误计数器的值控制着 MSCAN 的实际 CAN 总线状态。只要设置了状态变化中断标志 (CSCIF)，这些位就显示 MSCAN 的与接收器有关的适当 CAN 总线状态。位 RSTAT1、RSTAT0 的编码是： 00 RxOK: $0 \leq$ 接收错误计数器 $\leq 96$ 01 RxWRN: $96 < \text{接收错误计数器} \leq 127$ 10 RxERR: $127 < \text{发送错误计数器}$ 11 Bus-off <sup>1</sup> : 发送错误计数器 $> 255$

表 12-9. CANRFLG 寄存器字段描述 (continued)

字段	描述
3:2 TSTAT[1:0]	发送器状态位 — 错误计数器的值控制着 MSCAN 的实际 CAN 总线状态。只要设置了状态变化中断标志 (CSCIF)，这些位就显示 MSCAN 的与接收器有关的适当 CAN 总线状态。位 TSTAT1、TSTAT0 的编码是： 00 TxOK: $0 \leq$ 接收错误计数器 $\leq 96$ 01 RxOK: $0 < \text{接收错误计数器} \leq 127$ 10 RxERR: $127 < \text{接收错误计数器} \leq 255$ 11 Bus-off1: 发送错误计数器 $> 255$
1 OVRIF	溢出中断标志 — 该标志在出现数据溢出情况时设置。如果没有被屏蔽，当设置了该标志时有一个错误中断产生。 0 无数据溢出情况 1 检测到数据溢出
0 RXF <sup>2</sup>	接收缓冲器已满标志 — 当新报文被转移到接收器 FIFO 中时，RXF 由 MSCAN 进行置位。该标志表示移位缓冲器是否接收了正确的报文（匹配标识符，匹配循环冗余代码（CRC）和未检测到其他错误）。在 CPU 从接收器 FIFO 中的 RxFG 缓冲器那里读取了该报文后，RXF 标志必须清除，以释放缓冲器。已设置的 RXF 标志禁止下一个 FIFO 条目转移到前景缓冲器（RxFG）。如果未被屏蔽，当设置了该标志时有一个接收中断产生。 0 RxFG 中没有新报文 1 接收器 FIFO 非空。RxFG 中有报文

<sup>1</sup> 处于“总线脱离”状态的最重要 CAN 总线状态的冗余报文。只有当 Tx 错误计数器的错误超过 255 个时才会出现这种情况。总线脱离会影响接收器状态。一旦发送器离开其总线脱离状态，接收器状态也立即跳到 RxOK。也请参见本寄存器中的 TSTAT[1:0] 编码。

<sup>2</sup> T 为确保数据完整性，当 RXF 标志清除时，不要读取接收缓冲器寄存器。对于那些有双 CPU 的 MCU 来说，当 RXF 标志被清除时读取接收缓冲器寄存器可能会导致 CPU 故障。

### 12.3.5 MSCAN 接收器中断使能寄存器 (CANRIER)

该寄存器包含用于 CANRFLG 寄存器中描述的中断标志的中断使能位。

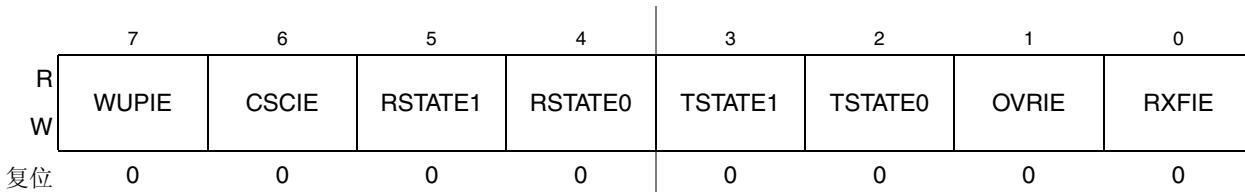


图 12-9. MSCAN 接收器中断使能寄存器 (CANRIER)

#### 注意

当初始化模式处于有效状态时，CANRIER 寄存器保持复位状态 (INITRQ=1 and INITAK=1)。当未处于初始化模式时，该寄存器可以写入 (INITRQ=0, INITAK=0)。

The RSTATE[1:0], TSTATE[1:0] 位不受初始化模式影响。

读取：任何时间

写入：未处于初始化模式的任何时间。

表 12-10. CANIER 寄存器字段描述

字段	描述
7 WUPIE <sup>1</sup>	<b>唤醒中断使能</b> 0 无中断请求从该事件中产生。 1 唤醒事件引起唤醒中断请求。
6 CSCIE	<b>CAN 状态变化中断使能</b> 0 无中断请求从该事件中产生。 1 CAN 状态变化事件引起错误中断请求。
5:4 RSTATE[1:0]	接收器状态变化使能—这些 RSTAT 使能位控制接收器状态变化而引起 CSCIF 中断的电平状态。独立于所选电平状态, RSTAT 标志继续显示实际接收器状态, 且只有在没有 CSCIF 中断产生时才会更新。 00 未生成由于接收器状态变化而引起的任何 CSCIF 中断。 01 只有当接收器进入或离开“总线脱离”状态时才会生成 CSCIF 中断。为生成 CSCIF 中断丢弃其他接收器状态变化。 10 只有当接收器进入或离开“RxErr”或“总线脱离” <sup>2</sup> 状态时才会生成 CSCIF 中断。为生成 CSCIF 中断丢弃其他接收器状态变化。 11 所有状态变化都生成 CSCIF 中断。
3:2 TSTATE[1:0]	T发送器状态变化使能—这些 TSTAT 使能位控制发送器状态变化而引起 CSCIF 中断的电平状态。独立于所选电平状态, TSTAT 标志继续显示实际发送器状态, 且只有在没有 CSCIF 中断产生时才会更新。 00 未生成由于接收器状态变化而引起的任何 CSCIF 中断。 01 只有当发送器进入或离开“总线脱离”状态时才会生成 CSCIF 中断。为生成 CSCIF 中断丢弃其他接收器状态变化。 10 只有当发送器进入或离开“总线脱离”状态时才会生成 CSCIF 中断。为生成 CSCIF 中断丢弃其他接收器状态变化。 11 所有状态变化都生成 CSCIF 中断。
1 OVRIE	<b>溢出中断使能</b> 0 无中断请求从该事件中生成。 1 溢出事件引起错误中断请求。
0 RXFIE	接收器已满中断使能 0 无中断请求从该事件中生成。 1 接收缓冲器已满(成功报文接收)事件引起接收器中断请求。

<sup>1</sup> 如果需要从停止或等待模式中进行恢复的机制, 必须同时使能 WUPIE 和 WUPE(参见 12.3.1, “MSCAN 控制寄存器 0 (CANCTL0)” )。

### 12.3.6 MSCAN 发送器标志寄存器 (CANTFLG)

每个发送缓冲器空标志在 CANTIER 寄存器中都有相关的中断使能位。 r.

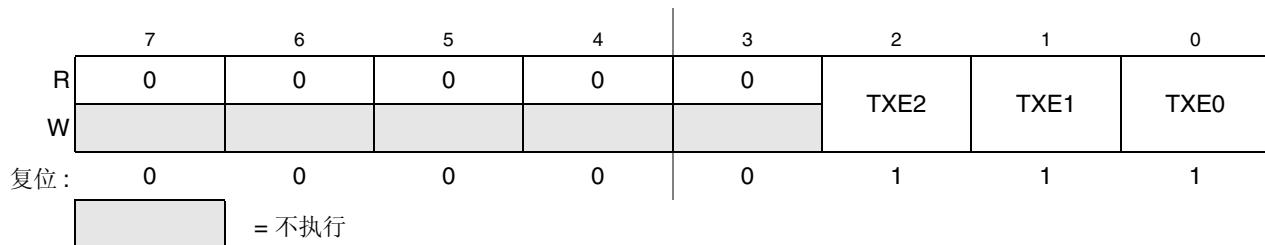


图 12-10. MSCAN 发送器标志寄存器 (CANTFLG)

### 注意

当初始化模式处于有效状态时, CANTFLG 寄存器保持复位状态 (INITRQ=1, INITAK=1)。当未处于初始化模式时, 该寄存器可以写入 (INITRQ = 0 and INITAK = 0).

读取: 任何时间

写入: TXEx 标志不处于初始化模式的任何时间; 写入 1 清除标志, 写入 0 忽略标志。

表 12-11. CANTFLG 寄存器字段描述

字段	描述
2:0 TXE[2:0]	<p>发送器缓冲器空—该标志表示相关发送报文缓冲器空, 因此没有安排用于发送。在发送缓冲器中放好报文并准备好发送后, CPU 必须清除该标志。报文发送成功后, MSCAN 设置该标志。当发送请求由于中止请求而被成功中止时, MSCAN 也设置该标志 (参见 12.3.8, “MSCAN Transmitter 发送器报文中止请求寄存器 (CANTARQ)”). 如果未被屏蔽, 当设置了该标志时产生发送中断。</p> <p>清除 TXEx 标志也会清除相应的 ABTAKx (参见 12.3.9, “MSCAN 发送器报文中止确认寄存器 (CANTAAK)”). 当设置了 TXEx 标志时, 相应的 ABTRQx 位被清除 (参见 12.3.8, “MSCAN Transmitter 发送器报文中止请求寄存器 (CANTARQ)”).</p> <p>当监听模式处于有效状态时 (参见 12.3.2, “控制寄存器 1 (CANCTL1)”) TXEx 标志不能清除且不进行任何发送。</p> <p>当相应的 TXEx 位被清除 (TXEx = 0) 且缓冲器被安排用于发送时, 对发送缓冲器的读写操作会被拦截。</p> <p>0 相关报文缓冲器已满 (加载了准备发送的报文) 1 相关报文缓冲器空 (未安排)</p>

### 12.3.7 MSCAN 发送器中断使能寄存器 (CANTIER)

该寄存器包含发送缓冲器空中断标志的中断使能位。

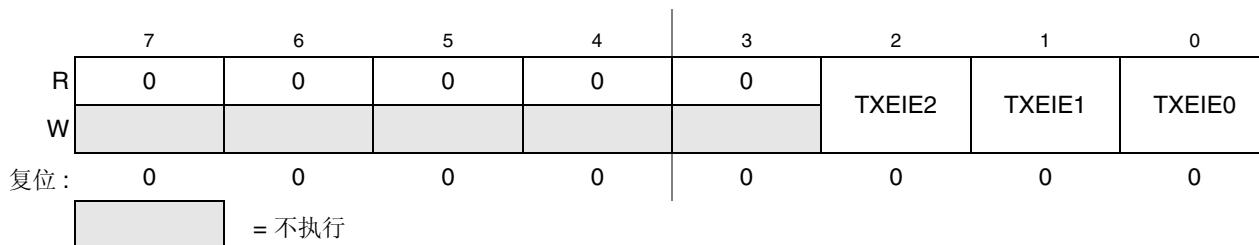


图 12-11. MSCAN 发送器中断使能寄存器 (CANTIER)

### 注意

当初始化模式处于有效状态时, CANTIER 寄存器保持复位状态 (INITRQ=1, INITAK=1)。当未处于初始化模式时, 该寄存器可以写入 (INITRQ=0, INITAK=0)。

读取: 任何时间

写入: 未处于初始化模式的任何时间

表 12-12. CANTIER 寄存器字段描述

字段	描述
2:0 TXEIE[2:0]	发送器空中断使能 0 无中断请求从该事件中生成。 1 发送器空（发送缓冲器可用于发送）事件引起发送器空中断请求。更多报文参见 12.5.2.2，“发送结构”。

### 12.3.8 MSCAN Transmitter 发送器报文中止请求寄存器 (CANTARQ)

The CANTARQ 寄存器中止报文发送队列的请求。

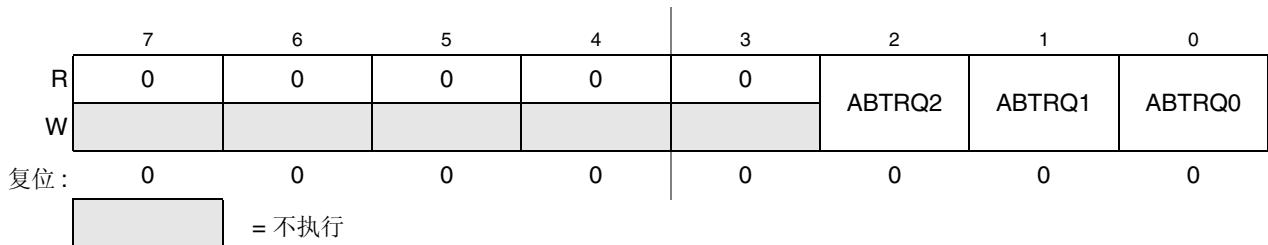


图 12-12. MSCAN 发送器报文中止请求寄存器 (CANTARQ)

#### 注意

当初始化模式处于有效状态时，CANTARQ 寄存器保持复位状态（INITRQ=1，INITAK=1）。当未处于初始化模式时，该寄存器可以写入（INITRQ=0，INITAK=0）。

读取：任何时间

写入：未处于初始化模式的任何时间

表 12-13. CANTARQ 寄存器字段描述

字段	描述
2:0 ABTRQ[2:0]	中止请求 —CPU 设置 ABTRQx 位，请求中止预定的报文缓冲器 (TXEx = 0)。如果报文还没有开始发送，或者如果发送没有成功（仲裁丢失或错误），MSCAN 就同意请求。当报文被中止时，相关 TXE（参见 12.3.6，“MSCAN 发送器标志寄存器 (CANTFLG)”）和中止确认标志 (ABTAK，参见 12.3.9，“MSCAN 发送器报文中止确认寄存器 (CANTAAK)”）被设置，且若使能就触发发送中断。CPU 不能复位 ABTRQx。每当设置了相关的 TXE 标志时，ABTRQx 就被复位。 0 无中止请求 1 中止请求产生

### 12.3.9 MSCAN 发送器报文中止确认寄存器 (CANTAAK)

CANTAAK 寄存器表示成功中止报文发送队列的请求，如果由 CANTARQ 寄存器中的适当位请求的话。

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	ABTAK2	ABTAK1	ABTAK0
W								
复位：	0	0	0	0	0	0	0	0

= 不执行

图 12-13. MSCAN 发送器报文中止确认寄存器 (CANTAAK)

#### 注意

当初始化模式处于有效状态时，CANTAAK 寄存器保持复位状态 (INITRQ = 1, INITAK = 1)。

读取：任何时间

写入：不为 ABTAKx 标志执行

表 12-14. CANTAAK 寄存器字段描述

字段	描述
2:0 ABTAK[2:0]	中止确认 — 该标志确认由于 CPU 的产生发送中止请求而中止报文。当某个报文缓冲器标空时，软件可以使用该标志来确认报文是成功中止还是已发送出去。每当相应 TXE 标志被清除时，ABTAKx 标志就会清除。 0 报文未被中止。 1 报文被中止。

### 12.3.10 MSCAN 发送缓冲器选择寄存器 (CANTBSEL)

实际发送报文缓冲器的 CANTBSEL 选择，该缓冲器可以在 CANTXFG 寄存器空间访问。

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	TX2	TX1	TX0
W								
复位：	0	0	0	0	0	0	0	0

= 不执行

图 12-14. MSCAN 发送缓冲器选择寄存器 (CANTBSEL)

#### 注意

当初始化模式处于有效状态时，CANTARQ 寄存器保持复位状态 (INITRQ = 1, INITAK = 1)。当未处于初始化模式时，该寄存器可以写入 (INITRQ = 0 and INITAK = 0)。

读取：发现最低排列顺序位设置为 1，所有其他位读为 0

写入：未处于初始化模式的任何时间

表 12-15. CANTBSEL 寄存器字段描述

字段	描述
2:0 TX[2:0]	发送缓冲器选择 — 在 CANTXFG 寄存器空间里置位为 1 的最低位（例如 TX1 = 1、TX0 = 1 选择发送缓冲器 TX0；TX1 = 1、TX0 = 0 选择发送缓冲器 TX1）。如果相应 TXEx 位被清除，缓冲器被安排用于传输，所选发送缓冲器的读写接入会被拦截。（参见 12.3.6，“MSCAN 发送器标志寄存器 (CANTFLG)”）。 0 相关报文缓冲器不被选择 1 选择了相关报文缓冲器，如果是最低置 1 位

下面给出了一个 CANTBSEL 寄存器使用的简短编程示例。

Tx 为了获得下一个可用发送缓冲器，应用软件必须读取 CANTFLG 寄存器，并将该值重新写入 CANTBSEL 寄存器。在该示例中，Tx 缓冲器 TX1 和 TX2 可用。从 CANTFLG 读取的值因此为 0b0000\_0110。当该值重新写入 CANTBSEL 时，CANTXFG 中选择 Tx 缓冲器 TX1，因为设置为 1 的最低位处于位 1。从 CANTBSEL 重新读取该值会导致 0b0000\_0010，因为只有设置为 1 的最低位显示。这种机制简化了应用软件选择下一个可用 Tx 缓冲器的逻辑。

- LDD CANTFLG; 读取值为 0b0000\_0110
- STD CANTBSEL; 写入值为 0b0000\_0110
- LDD CANTBSEL; 读取值为 0b0000\_0010

如果取消了所有发送报文缓冲器选择，则不允许访问 CANTXFG 缓冲器寄存器

### 12.3.11 MSCAN 标识符验收控制寄存器 (CANIDAC)

CANIDAC 寄存器如下所述用于标识符滤波器验收控制。

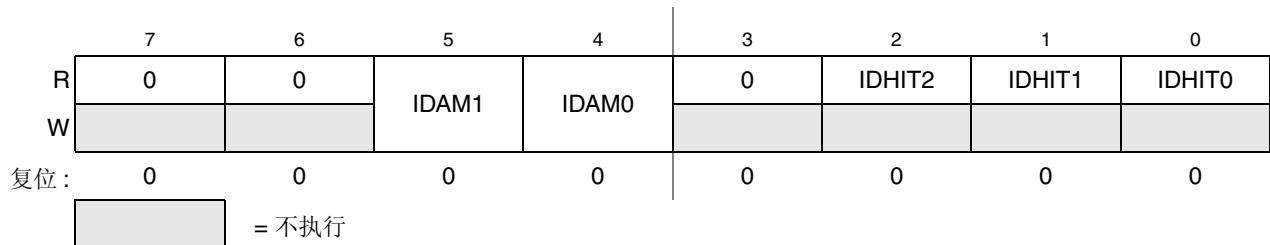


图 12-15. MSCAN 标识符验收控制寄存器 r (CANIDAC)

读取：任何时间

写入：处于初始化模式的任何时间 (INITRQ = 1 and INITAK = 1), except bits IDHITx, 只读位 IDHITx 除外。

表 12-16. CANIDAC 寄存器字段描述

字段	描述
5:4 IDAM[1:0]	标识符接收模式 — CPU 设置这种标志来定义标识符接收滤波器结构（参见 12.5.3，“标识符接收滤波器”）。表 12-17 总结了不同设置。在滤波器关闭模式中，不接收任何报文，因此前景缓冲器永远不会重载。 2:0
2:0 IDHIT[2:0]	标识符接收有效标志指示器 — MSCAN 设置这些标志来显示标识符接收有效标志（参见 12.5.3，“标识符接收滤波器”）。表 12-18 总结了不同设置。

表 12-17. 标识符接收模式设置

IDAM1	IDAM0	标识附接收模式
0	0	2 个 32 位接收 滤波器
0	1	4 个 16 位接收 滤波器
1	0	8 个 8 位接收 滤波器
1	1	滤波器关闭

表 12-18. 标识符接收有效标志指示器

IDHIT2	IDHIT1	IDHITO	标识附接收有效标志
0	0	0	滤波器 0 有效标志
0	0	1	滤波器 1 有效标志
0	1	0	滤波器 2 有效标志
0	1	1	滤波器 3 有效标志
1	0	0	滤波器 4 有效标志
1	0	1	滤波器 5 有效标志
1	1	0	滤波器 6 有效标志
1	1	1	滤波器 7 有效标志

IDHITx 指示器总是与前景缓冲器 (RxFG) 中的报文有关。当报文被转移到接收器 FIFO 的前景缓冲器时，指示器也相应更新。

### 12.3.12 MSCAN 其他寄存器 (CANMISC)

这种寄存器提供了一些其他功能。

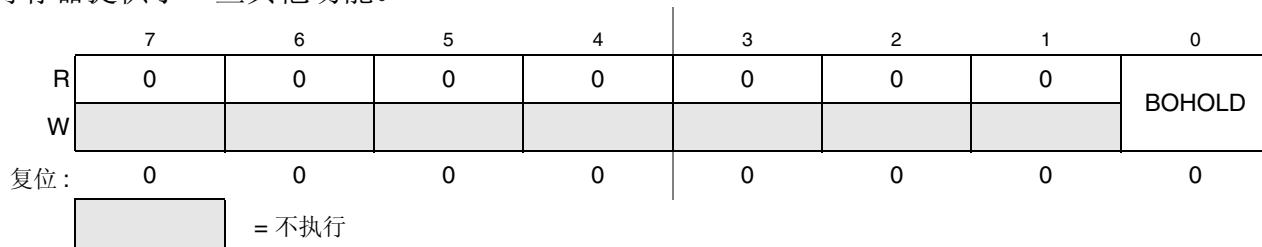


图 12-16. MSCAN 其他寄存器 (CANMISC)

读取：任何时间

写入：任何时间；写入 ‘1’ 清除标志，写入 ‘0’ 忽略标志

表 12-19. CANMISC 寄存器字段描 ^

字段	描述
0 BOHOLD	总线脱离状态持续到用户请求 —12.3.2，“控制寄存器 1 (CANCTL1)”，“MSCAN 控制寄存器 1 (CANCTL1) 中设置了 BORM，此标志位显示模块是否已经进入总线脱离状态。清除该位则请求从总线脱离恢复。如需了解详细报文，12.6.2，“总线脱离恢复”。 0 模块未总线脱离，或在总线脱离状态并已请求恢复 1 模块总线脱离，并保持该状态直到用户请求

### 12.3.13 MSCAN 接收错误计数器 (CANRXERR)

该寄存器反应 MSCAN 接收错误计数器的状态。

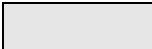
	7	6	5	4		3	2	1	0
R	RXERR7	RXERR6	RXERR5	RXERR4	RXERR3	RXERR2	RXERR1	RXERR0	
W									
复位：	0	0	0	0	0	0	0	0	0
		= 不执行							

图 12-17. MSCAN 接收错误计数器 (CANRXERR)

读取：仅在睡眠模式 (SLPRQ = 1, SLPAK = 1) 或初始化模式 (INITRQ = 1 and INITAK = 1)  
写入：不执行

#### 注意

在非睡眠或初始化模式外的任意其他模式中读取该寄存器会返回错误值。对于那些具有双 CPU 的 MCU 来说，这可能会引起 CPU 故障情况。

在特殊模式中写入该寄存器可能改变 MSCAN 功能。

### 12.3.14 MSCAN 发送错误计数器 (CANTXERR)

该寄存器反应 MSCAN 发送错误计数器的状态。

	7	6	5	4		3	2	1	0
R	TXERR7	TXERR6	TXERR5	TXERR4	TXERR3	TXERR2	TXERR1	TXERR0	
W									
复位：	0	0	0	0	0	0	0	0	0
		= 不执行							

图 12-18. MSCAN 发送错误计数 ~ (CANTXERR)

读取：仅在睡眠模式 (SLPRQ = 1, SLPAK = 1) 或初始化模式 (INITRQ = 1, INITAK = 1)。  
写入：不执行

#### 注意

在非睡眠或初始化模式外的任意其他模式中读取该寄存器会返回错误值。对于那些具有双 CPU 的 MCU 来说，这可能会引起 CPU 故障情况。

在特殊模式中写入该寄存器可能改变 MSCAN 功能。

### 12.3.15 MSCAN 标识符接收寄存器 (CANIDAR0-7)

一旦接受，每条报文将写入后台接收缓冲器。只有当报文通过了标识符接收和标识符掩码寄存器中的滤波，CPU 才被告知读取报文（接受），否则报文会被下一条报文覆盖（丢弃）。

MSCAN 的接收寄存器采用逐位方式（参见 12.5.3，“标识符接收滤波器”），应用于 IDR0 – IDR3 寄存器（参见 12.4.1，“标识符寄存器 (IDR0 鑫 DR3)”）of incoming messages in a bit by bit manner（参见 12.5.3，“标识符接收滤波器”）。

对于扩展标识符，要应用所有四个接收和掩码寄存器。对于标准标识符，只应用前两个 (CANIDAR0/1、CANIDMR0/1)。

	7	6	5	4	3	2	1	0
R W	AC7	AC6	AC5	AC4	AC3	AC2	AC1	AC0
复位	0	0	0	0	0	0	0	0

图 12-19. MSCAN 标识符接收寄存器（第一页）— CANIDAR0-CANIDAR3

读取：任何时间

写入：处于初始化模式的任何时间 (INITRQ = 1 and INITAK = 1)

表 12-20. CANIDAR0-CANIDAR3 寄存器字段描述

字段	描述
7:0 AC[7:0]	接收码位— AC[7:0] 由用户定义的位顺序组成，通过这种方式，接收报文缓冲器的相关标识符寄存器 (IDRn) 的相应位进行比较。比较结果然后用相应标识符掩码寄存器进行掩码屏蔽 r。

	7	6	5	4	3	2	1	0
R W	AC7	AC6	AC5	AC4	AC3	AC2	AC1	AC0
复位	0	0	0	0	0	0	0	0

图 12-20. MSCAN 标识符接收标识符（第二页）— CANIDAR4-CANIDAR7

读取：任何时间

写入：处于初始化模式的任何时间 (INITRQ = 1 and INITAK = 1)

表 12-21. CANIDAR4-CANIDAR7 寄存器字段描述

字段	描述
7:0 AC[7:0]	接收码位— AC[7:0] 由用户定义的位顺序组成，通过这种方式，接收报文缓冲器的相关标识符寄存器 (IDRn) 的相应位进行比较。比较结果然后用相应标识符掩码寄存器进行掩码屏蔽。

### 12.3.16 MSCAN 标识符掩码寄存器 (CANIDMR0-CANIDMR7)

标识符掩码寄存器指定标识符接收寄存器中的哪些相应位与接收过滤比较。为了在 32 位滤波器模式中接收标准标识符，需要把掩码寄存器 CANIDMR1 和 CANIDMR5 中最后三个位 (AM[2:0]) 编程为“不比较”。为了在 16 位滤波器模式中接收标准标识符，需要把掩码寄存器 CANIDMR1、CANIDMR3、CANIDMR5 和 CANIDMR7 中的最后三个位 (AM[2:0]) 编程为“不比较”。

	7	6	5	4		3	2	1	0
R W	AM7	AM6	AM5	AM4		AM3	AM2	AM1	AM0
复位	0	0	0	0		0	0	0	0

图 12-21. MSCAN 标识符掩码寄存器 (第一页) — CANIDMR0-CANIDMR3

读取：任何时间

写入：处于初始化模式的任何时间 (INITRQ = 1 and INITAK = 1)

表 12-22. CANIDMR0-CANIDMR3 寄存器字段描述

字段	描述
7:0 AM[7:0]	接收掩码位—如果该寄存器中的某位被清除，这表示检测到匹配前，标识符接收寄存器中的相应位必须和它的标识符位相同。如果所有类似位均匹配，报文被接受，如果此位置 1，这表示标识符接收寄存器中的相应位的状态不会影响报文是否被接受。 0 匹配相应接收码寄存器和标识符位 1 忽略相应接收码寄存器位（不比较）

	7	6	5	4		3	2	1	0
R W	AM7	AM6	AM5	AM4		AM3	AM2	AM1	AM0
复位	0	0	0	0		0	0	0	0

图 12-22. MSCAN 标识符掩码寄存器 (第二页) — CANIDMR4-CANIDMR7

读取：任何时间

写入：处于初始化模式的任何时间 (INITRQ = 1 and INITAK = 1)

表 12-23. CANIDMR4-CANIDMR7 寄存器字段描述

字段	描述
7:0 AM[7:0]	接收掩码位—如果该寄存器中的某位被清除，这表示检测到匹配前，标识符接收寄存器中的相应位必须和它的标识符位相同。如果所有类似位均匹配，报文被接受，如果此位置 1，这表示标识符接收寄存器中的相应位的状态不会影响报文是否被接受。 0 匹配相应接收码寄存器和标识符位 1 忽略相应接收码寄存器位（不比较）

## 12.4 报文存储模式

下面这一节详细介绍了接收和发送报文缓冲器的结构，以及相关控制寄存器。

为了简化程序员界面，接收和发送报文缓冲器的轮廓相同。每个报文缓冲器都在包含 13 字节数据结构的存储器映射中都分配 16 个字节。

我们还为发送缓冲器定义了一个发送缓冲器优先级寄存器（TBPR）。在该存储器映射的最后两个字节中，MSCAN 保存一个特殊的 16 位时间标签，采样于报文成功传输或接收后的内部计时器。如果设置了 TIME 位，这种功能只出现于发送和接收器缓冲器（参见 12.3.1，“[MSCAN 控制寄存器 0 \(CANCTL0\)](#)”）。

时间标签寄存器由 MSCAN 写入。CPU 只能读这些寄存器。

**表 12-24. 报文缓冲器结构**

Offset Address	Register	Access
0x00X0	Identifier Register 0	
0x00X1	Identifier Register 1	
0x00X2	Identifier Register 2	
0x00X3	Identifier Register 3	
0x00X4	Data Segment Register 0	
0x00X5	Data Segment Register 1	
0x00X6	Data Segment Register 2	
0x00X7	Data Segment Register 3	
0x00X8	Data Segment Register 4	
0x00X9	Data Segment Register 5	
0x00XA	Data Segment Register 6	
0x00XB	Data Segment Register 7	
0x00XC	Data Length Register	
0x00XD	Transmit Buffer Priority Register <sup>1</sup>	
0x00XE	Time Stamp Register (High Byte) <sup>2</sup>	
0x00XF	Time Stamp Register (Low Byte) <sup>3</sup>	

<sup>1</sup> Not applicable for receive buffers

<sup>2</sup> Read-only for CPU

<sup>3</sup> Read-only for CPU

[图 12-23](#) 为扩展标识符显示了接收和发送缓冲器常用的 13 字节数据结构。标准标识符在 IDR 寄存器中的映射 [图 12-24](#)。

由于基于 RAM1，接收和发送缓冲器的所有位在复位时均为 ‘x’。接收和发送缓冲器的所有保留或不使用位始终读为 ‘x’。

Register Name	Bit 7	6	5	4	3	2	1	Bit0	
IDR0	R W	ID28	ID27	ID26	ID25	ID24	ID23	ID22	ID21
IDR1	R W	ID20	ID19	ID18	SRR <sup>(1)</sup>	IDE <sup>(1)</sup>	ID17	ID16	ID15
IDR2	R W	ID14	ID13	ID12	ID11	ID10	ID9	ID8	ID7
IDR3	R W	ID6	ID5	ID4	ID3	ID2	ID1	ID0	RTR <sup>2</sup>
DSR0	R W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
DSR1	R W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
DSR2	R W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
DSR3	R W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
DSR4	R W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
DSR5	R W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
DSR6	R W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
DSR7	R W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
DLR	R W					DLC3	DLC2	DLC1	DLC0

= Unused, always read '0'

图 12-23. 接收 / 发送报文缓冲器 — 扩展标识符映射

<sup>1</sup> SRR 和 IDE 都为 1。

<sup>2</sup> RTR 的位置在扩展和标准标识符映射间存在差异。

读取：对于发送缓冲器，当设置了 TXEx 标志（参见 12.3.6，“MSCAN 发送器标志寄存器 (CANTFLG)”）且在 CANTBSEL 中选择了相应发送缓冲器（参见 12.3.10，“MSCAN 发送缓冲器选择寄存器 (CANTBSEL)”）。的任何时间。对于接收缓冲器，仅当设置了 RXF 标志（参见 12.3.4.1，“MSCAN 接收器标志寄存器 (CANRFLG)”）。

写入：对于发送缓冲器，当设置了 TXEx 标志（参见 12.3.6，“MSCAN 发送器标志寄存器 (CANTFLG)”）且在 CANTBSEL 中选择了相应发送缓冲器（参见 12.3.10，“MSCAN 发送缓冲器选择寄存器 (CANTBSEL)”）。的任何时间。对于接收缓冲器，不执行。

复位：因为基于 RAM，未定义（0x00XX）。

图 12-24. 接收 / 发送报文缓冲器 — 标准标识符映射

Register Name	Bit 7	6	5	4	3	2	1	Bit 0	
IDR0	R W	ID10	ID9	ID8	ID7	ID6	ID5	ID4	ID3
IDR1	R W	ID2	ID1	ID0	RTR <sup>1</sup>	IDE <sup>2</sup>			
IDR2	R W								
IDR3	R W								
= 不使用，始终读为 ‘x’									

<sup>1</sup> The position of RTR differs between extended and standard identifier mapping.

<sup>2</sup> IDE is 0.

## 12.4.1 标识符寄存器 (IDR0–IDR3)

扩展格式标识符的标识符寄存器共由 32 个位组成； ID[28:0]、 SRR、 IDE 和 RTR 位。标准格式标识符的标识符寄存器共由 13 个位组成； ID[10:0]、 RTR 和 IDE 位。

### 12.4.1.1 扩展标识符映射的 IDR0 – IDR3

	7	6	5	4		3	2	1	0
R W	ID28	ID27	ID26	ID25		ID24	ID23	ID22	ID21
复位：	x	x	x	x		x	x	x	x

图 12-25. 标识符寄存器 1 (IDR1) — 扩展标识符映射

表 12-25. IDR0 寄存器字段描述 — 扩展

字段	描述
7:0 ID[28:21]	扩展格式标识符—该标识符由 29 个扩展格式位 (ID[28:0]) 组成。ID28 是最高的位，仲裁流程期间最先在 CAN 总线上发送。标识符的优先级定义为处于最高位的最小二进制数。

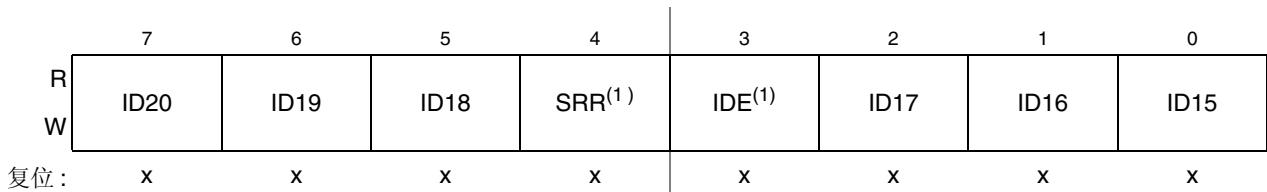


图 12-26. IDR3 — 扩展标识符映射

<sup>1</sup> 1 SRR 和 IDE 都为 1。

表 12-26. 标识符寄存器 1 (IDR1) — 扩展标识符映射

字段	描述
7:5 ID[20:18]	扩展格式标识符—该标识符由 29 个扩展格式位 (ID[28:0]) 组成。ID28 是最高的位，仲裁流程期间最先在 CAN 总线上发送。标识符的优先级定义为处于最高位的最小二进制数。
4 SRR	替代远程请求—该固定隐性位仅用于扩展格式。它必须由用户为传输缓冲器置位为 1，，并为接收缓冲器在 CAN 总线上置位为接收。
3 IDE	扩展—该标志显示扩展或标准标识符格式是否应用于该缓冲器。在接收缓冲器中，该标志设置为已接收，并向 CPU 显示如何处理缓冲器标识符寄存器。在发送缓冲器中，该标志向 MSCAN 显示将发送的标识符类型。 0 标准格式 (11 位) 1 扩展格式 (29 位) 2:0
2:0 ID[17:15]	扩展格式标识符—该标识符由 29 个扩展格式位 (ID[28:0]) 组成。ID28 是最高的位，仲裁流程期间最先在 CAN 总线上发送。标识符的优先级定义为处于最高位的最小二进制数。

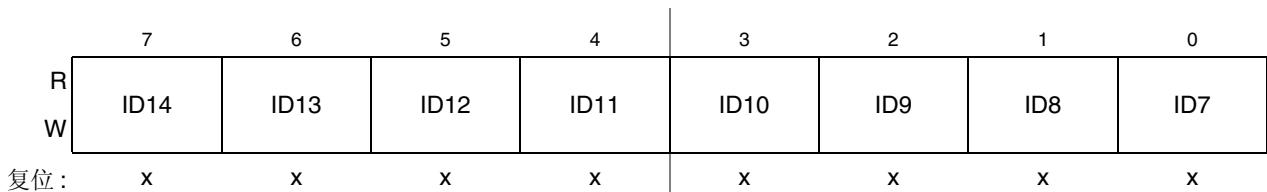


图 12-27. IDR2 — 扩展标识符映射

表 12-27. IDR2 寄存器字段说明—扩展

字段	描述
7:0 ID[14:7]	扩展格式标识符—该标识符由 29 个扩展格式位 (ID[28:0]) 组成。ID28 是最高的位，仲裁流程期间最先在 CAN 总线上发送。标识符的优先级定义为处于最高位的最小二进制数。

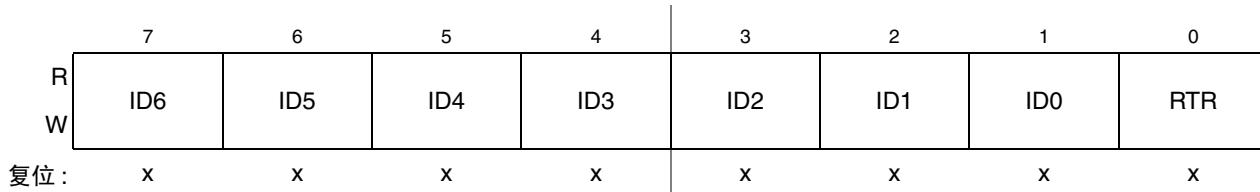


图 12-28. 标识符寄存器 3 (IDR3) — 扩展标识符映射

表 12-28. 标识符寄存器 0 — 标准映射

字段	描述
7:1 ID[6:0]	<b>扩展格式标识符</b> — 该标识符由 29 个扩展格式位 (ID[28:0]) 组成。ID28 是最高的位，仲裁流程期间最先在 CAN 总线上发送。标识符的优先级定义为处于最高位的最小二进制数。
0 RTR	远程发送请求 — 该标志反应 CAN 帧中远程发送请求的状态。在接收缓冲器中，它显示已接收帧的状态，并在软件中支持应答帧的发送。在发送缓冲器中，该标志定义将要发送的 RTR 位的设置。 0 数据帧 1 远程帧

## 12.4.2 标准标识符映射的 IDR0 - IDR3

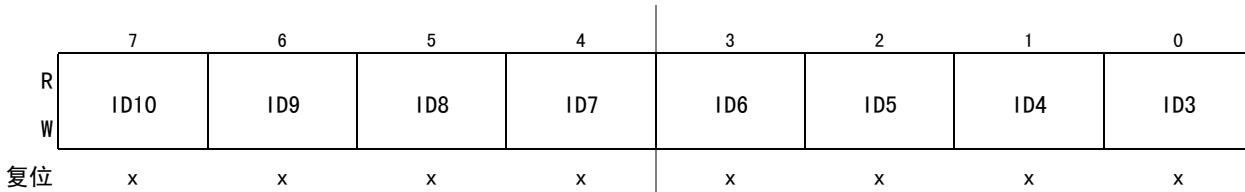


图 12-29. 标识符寄存器 0 — 标准映射

表 12-29. IDR0 寄存器字段描述 — 标准

字段	描述
7:0 ID[10:3]	<b>标准格式标识符</b> — 该标识符由 11 个扩展格式位 (ID[10:0]) 组成。ID10 是最高位，仲裁流程期间最先在 CAN 总线上发送。标识符的优先级定义为处于最高位的最小二进制数。也可参见表 12-30 中的 ID 位。



= 不使用，始终读为 ‘x’

图 12-30. 标识符寄存器 1 — 标准映射

<sup>1</sup> IDE 为 0.

表 12-30. IDR1 寄存器字段描述

字段	描述
7:5 ID[2:0]	<b>标准格式标识符</b> — 标准格式标识符 — 该标识符由 11 个扩展格式位 (ID[10:0]) 组成。ID10 是最高位，仲裁期间最先在 CAN 总线上发送。标识符的优先级定义为处于最高位的最小二进制数。也可参见表 12-29 中的 ID 位。
4 RTR	远程发送请求 — 该标志反应 CAN 帧中远程发送请求的状态。在接收缓冲器中，它显示已接收帧的状态，并在软件中支持应答帧的发送。在发送缓冲器中，该标志定义将要发送的 RTR 位的设置。 0 数据帧 1 远程帧
3 IDE	<b>ID 扩展</b> — 该标志显示扩展或标准标识符格式是否应用于该缓冲器。在接收缓冲器中，该标志设置为已接收，并向 CPU 显示如何处理缓冲器标识符寄存器。在发送缓冲器中，该标志向 MSCAN 显示将发送的标识符类型。 0 标准格式 (11 位) 1 扩展格式 (29 位)

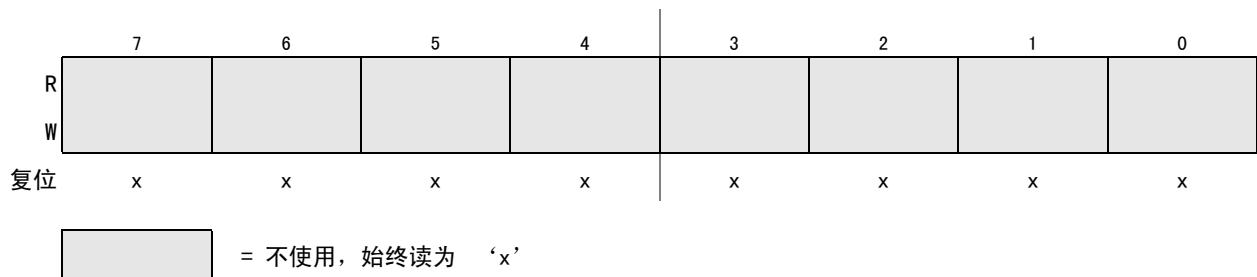


图 12-31. 标识符寄存器 2 — 标准映射

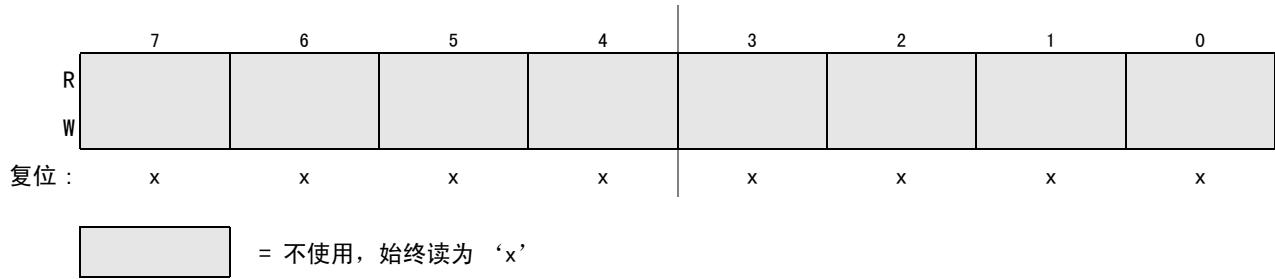


图 12-32. 标识符寄存器 3 — 标准映射

### 12.4.3 数据段寄存器 (DSR0-7)

8 个数据段寄存器（每个都带有位 DB[7:0]）包含将要发送或接收的数据。将要发送或接收的字节数由相应 DLR 寄存器中的数据长度代码决定。

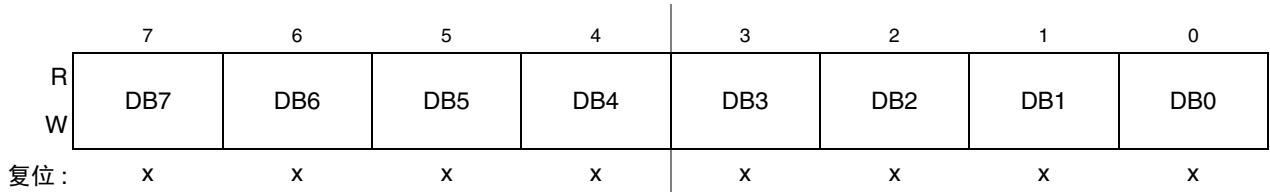


图 12-33. 数据段寄存器 (DSR0 - DSR7) — 扩展标识符映射

表 12-31. DSR0 – DSR7 寄存器字段描述

字段	描述
7:0 DB[7:0]	数据位 7:0

#### 12.4.4 数据长度寄存器 (DLR)

该寄存器保存 CAN 帧的数据长度字段。

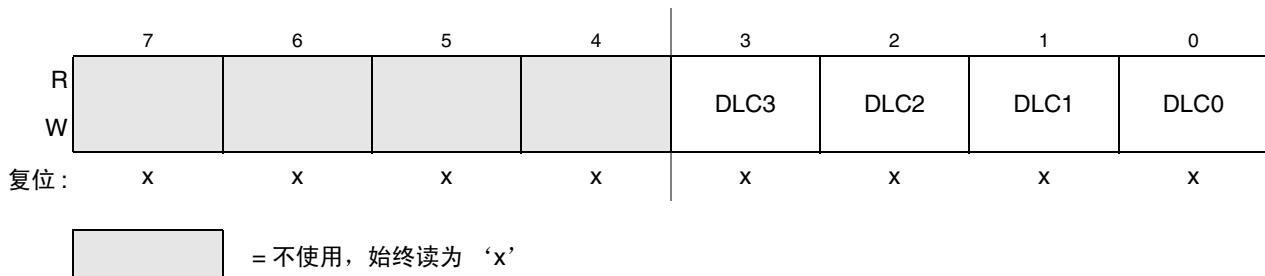


图 12-34. 数据长度寄存器 (DLR) — 扩展标识符映射

表 12-32. DLR 寄存器字段描述

字段	描述
3:0 DLC[3:0]	数据长度代码位 — T 数据长度代码位包含各自报文的字节数（数据字节计数）。在远程帧发送过程中，数据长度代码作为已编程码发送，而已发送的数据字节数始终为 0。数据帧的数据字节计数从 0 到 8 不等。表 12-33 显示设置 DLC 位的影响。

## 12.4.5 发送缓冲器优先寄存器 (TBPR)

表 12-33. 数据长度代码

数据长度代码				数据字节计数 Count
DLC3	DLC2	DLC1	DLC0	
0	0	0	0	0
0	0	0	1	1
0	0	1	0	2
0	0	1	1	3
0	1	0	0	4
0	1	0	1	5
0	1	1	0	6
0	1	1	1	7
1	0	0	0	8

该寄存器定义相关报文发送缓冲器的本地优先级。本地优先级用于 MSCAN 的内部优先级排队程序，优先级定义为最小二进制数字取得最高优先级。MSCAN 执行下列内部优先级排队机制：

- 带有 TXEx 清除标志的所有发送缓冲器在发送 SOF (帧开始) 前立即参与优先级排队。
- 带有最低本地优先级字段的发送缓冲器优先。

当出现一个以上的缓冲器具有相同最低优级的情况时，索引编号较小的报文缓冲器优先。

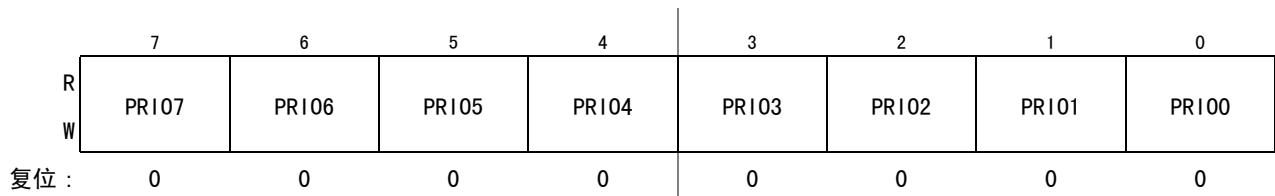


图 12-35. 发送缓冲器优先寄存器 (TBPR)

读取：当设置了 TXEx 标志（参见 12.3.6，“MSCAN 发送器标志寄存器 (CANTFLG)”）且在 CANTBSEL 中选择了相应发送缓冲器（见 12.3.10，“MSCAN 发送缓冲器选择寄存器 (CANTBSEL)”）的任何时间。

写入：当设置了 TXEx 标志（参见 12.3.6，“MSCAN 发送器标志寄存器 (CANTFLG)”）且在 CANTBSEL 中选择了相应发送缓冲器（参见 12.3.10，“MSCAN 发送缓冲器选择寄存器 (CANTBSEL)”）的任何时间。

## 12.4.6 时间标签寄存器 (TSRH – TSRL)

如果使能了 TIME 位，只要报文已经在 CAN 总线上得到确认，MSCAN 就把时间标签写入有效发送或接收缓冲器中的各自寄存器（参见 12.3.1，“MSCAN 控制寄存器 0 (CANCTL0)”）。发送时，CPU 只有在各自发送缓冲器标志空后才可以读取时间标签。

用于标签印的计时器值从自由运行的内部 CAN 位时钟中获取。计时器溢出不通过 MSCAN 显示。计时器在初始化模式中复位（所有位设置为 0）。CPU 只能读取时间标签。

	7	6	5	4		3	2	1	0
R	TSR15	TSR14	TSR13	TSR12		TSR11	TSR10	TSR9	TSR8
W									
复位：	x	x	x	x		x	x	x	x

图 12-36. 时间标签寄存器—高字节 (TSRH)

	7	6	5	4		3	2	1	0
R	TSR7	TSR6	TSR5	TSR4		TSR3	TSR2	TSR1	TSR0
W									
复位 t:	x	x	x	x		x	x	x	x

图 12-37. 时间标签寄存器—低字节 (TSRL)

读取：当设置了 TXEx 标志（参见 12.3.6，“MSCAN 发送器标志寄存器 (CANTFLG)”）且在 CANTBSEL 中选择了相应发送缓冲器（参见 12.3.10，“MSCAN 发送缓冲器选择寄存器 (CANTBSEL)”）的任何时间。

写入：不执行

## 12.5 功能描述

### 12.5.1 概述

本小节提供了 MSCAN 的完整功能描述，它描述了介绍中列出的每种功能和模式。

## 12.5.2 报文存储

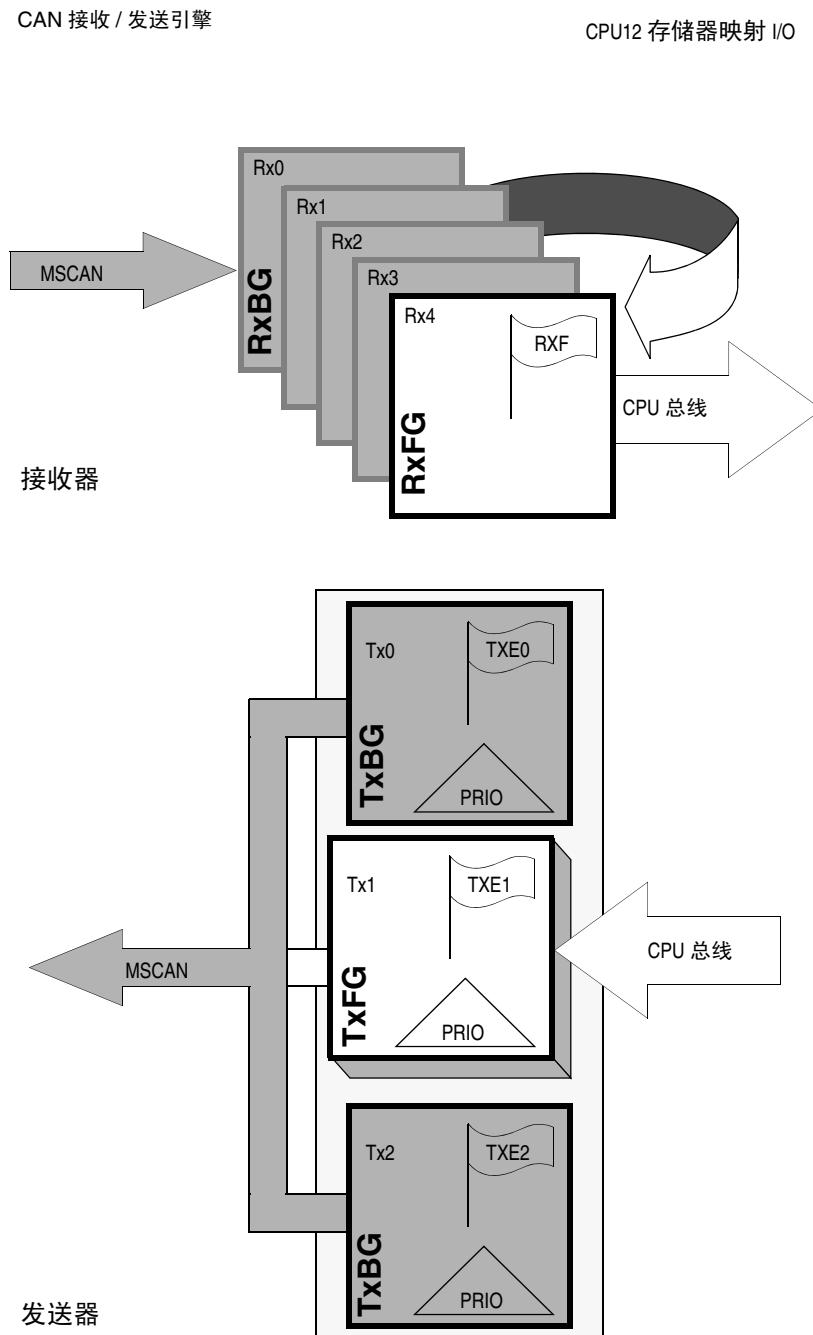


图 12-38. 报文缓冲器结构的用户模型

MSCAN 促进了一个能够满足一系列网络应用需求的先进报文存储系统。

### 12.5.2.1 报文发送基础

现代应用层软件的建立基于两个基本假设：

- 任何 CAN 节点都能够发送出安排好的报文流，而不需要在两条报文间释放 CAN 总线。这些节点在发送上一条报文后立即仲裁 CAN 总线，只当仲裁丢失时释放 CAN 总线。
- 安排 CAN 节点内的内部报文队列，这样，如果有多条报文准备发送时，最高优先级报文首先发出。

以上中描述的行为不能用单个发送缓冲器来实现。该缓冲器在上一条报文发送后必须立即重新加载。加载流程的持续时间有限，必须在帧间顺序 (IFS) 内完成，以便能够发送不中断报文流。即便这对于有限总线速度的 CAN 来说可行，但它要求 CPU 有最短的发送中断延迟时间。

双缓冲器机制能够把发送缓冲器的重新加载和实际的报文发送分开，因此降低了 CPU 的响应要求。问题可能出在完成报文的发送时 CPU 正重新加载第二个缓冲器，这时没有缓冲器做好发送准备，CAN 总线会被释放。

无论在什么情况下，至少需要三个发送缓冲器来满足上述第一个要求。MSCAN 有三个发送缓冲器。

第二个要求需要某些类型的内部优先排队，MSCAN 用 [12.5.2.2，“发送结构”](#) 中描述的“本地优先级”来执行该优先排队。

### 12.5.2.2 发送结构

MSCAN 三重发送缓冲器机制允许提前建立多条报文，从而优化了实时性能。这三个缓冲器的安排如图 [12-38](#). 所示。

这三个缓冲器都具有类似接收缓冲器的 13 字节数据结构（参见 [12.4，“报文存储模式”](#)）。[12.4.5，“发送缓冲器优先寄存器 \(TBPR\)”](#) 包含 8 位本地优先级字段 (PRIO)（参见 [12.4.5，“发送缓冲器优先寄存器 \(TBPR\)”](#)）。剩下的两个字节用于报文的时间标签，如果需要的话（参见 [12.4.6，“时间标签寄存器 \(TSRH - TSRL\)”](#)）。

要发送报文，CPU 必须确定可用的发送缓冲器，这由置位的发送器缓冲器空 (TXEx) 标志（参见 [12.3.6，“MSCAN 发送器标志寄存器 \(CANTFLG\)”](#)）表示。如果发送缓冲器可用，CPU 必须通过写入 CANTBSEL 寄存器（参见 [12.3.10，“MSCAN 发送缓冲器选择寄存器 \(CANTBSEL\)”](#)），为该缓冲器设置一个指针。这使得各自的缓冲器能够在 CANTXFG 地址空间内访问（参见 [12.4，“报文存储模式”](#)）。与 CANTBSEL 寄存器有关的算法功能简化了发送缓冲器选择。此外，这种机制使程序软件处理更为简单，因为发送流程只需访问一个地址，节省所需地址空间。

然后，CPU 将标识符、控制位和数据内容保存到一个发送缓冲器。最后，通过清除相关 TXE 标志，缓冲器标志为发送准备就绪。

MSCAN 然后安排报文发送，并通过设置相关 TXE 标志，通知缓冲器成功发送。当设置了 TXEx，可触发发送中断（参见 [12.5.7.2，“发送中断”](#)<sup>1</sup>），能够用来使应用软件重新加载缓冲器。

1. 只有当未屏蔽时才会发生发送中断。轮询机制也可应用于 TXEx。

当 CAN 总线赢得仲裁时，如果有一个以上的缓冲器等待发送，MSCAN 则使用三个缓冲器的本地优先级设置来决定优先顺序。因此，每个发送缓冲器都有 8 位本地优先级字段（PRIO）。当报文建立时，应用软件就编辑该字段。本地优先级反应了在从该节点发送的有关报文之间的优先级顺序。具有最低二进制代码的 PRIO 字段占最高优先级。每当 MSCAN 为 CAN 总线进行仲裁时，就会引发内部调度程序。当出现发送错误时也会如此。

当应用软件安排了高优先级报文时，可能有必要中止三个发送缓冲器的某一个低优先级报文。由于正发送的报文不能中止，因此用户必须通过设置相应中止请求位（ABTRQ）请求中止（参见 [12.3.8，“MSCAN Transmitter 发送器报文中止请求寄存器 \(CANTARQ\)”](#)）。可能的话，MSCAN 通过以下方式同意该请求：

1. 在 CANTAAK 寄存器中设置相应的中止确认标志（ABTAK）。
2. 设置相关的 TXE 标志来释放缓冲器。
3. 生成发送中断。发送中断处理程序软件能够根据 ABTAK 标志的设置确定是报文中止（ABTAK = 1）还是已发送（ABTAK = 0）。

### 12.5.2.3 接收结构

收到的报文保存在 5 级输入 FIFO 中。5 个报文缓冲器被交替映射到单个存储器区域（参见 [图 12-38](#)）。后台接收缓冲器（RxBG）只与 MSCAN 相关，但前景接收缓冲器可以通过 CPU 寻址（参见 [图 12-38](#)）。这种机制简化了处理程序软件，因为接收流程只需访问一个地址。

如使能的话，所有接收缓冲器都有 15 字节大小空间来保存 CAN 控制位、标识符（标准或扩展）、数据内容（参见 [12.4，“报文存储模式”](#)）。

接收器已满标志（RXF）（参见 [12.3.4.1，“MSCAN 接收器标志寄存器 \(CANRFLG\)”](#)）显示前景接收缓冲器的状态。当缓冲器包含带有匹配标识符的正确接收报文时，设置该标志。

接收时，检查每条报文，看看它是否通过滤波器（参见 [12.5.3，“标识符接收滤波器”](#)），同时被写入有效 RxBG。成功接收到有效报文后，MSCAN 将 RxBG 的内容转移到接收器 FIFO2，设置 RXF 标志并向 CPU3 生成一个接收中断（参见 [12.5.7.3，“接收中断”](#)）。用户的接收处理程序必须从 RxFG 读取收到的报文，然后复位 RXF 标志，确认中断、释放前景缓冲器。在某些情况下，紧跟 CAN 帧的 IFS 字段后的的新报文，将被接收到下一个可用 RxBG 中。如果 MSCAN 在其 RxBG 中接收到无效报文（错误标识符、发送错误等），缓冲器的实际内容将被下一条报文覆盖。缓冲器随后不会转移到 FIFO。

当 MSCAN 模块正在发送报文时，MSCAN 把其自己发送的报文接收到后台接收缓冲器 RxBG，但不会将它转移到接收器 FIFO，生成接收中断或在 CAN 总线上响应其自己的报文。这一规则的例外是在环回模式（参见 [12.3.2，“控制寄存器 1 \(CANCTL1\)”](#)）中，这时 MSCAN 会完全按照同所有其他报文一样的方式处理其自己的报文。当仲裁丢失时，MSCAN 接收其自己发送的报文。如果仲裁丢失，MSCAN 必须做好成为接收器的准备。

当 FIFO 中的所有接收报文缓冲器充满了带有已接收标识符的正确接收报文，且从 CAN 总线中正确接收到另外一条带有已接收标识符的报文时，就会出现溢出。后面这一条报文被丢弃，并生成带有溢出标志的错误中断（参见 [12.5.7.5，“错误中断”](#)）。当接收器 FIFO 已满时，MSCAN 仍能发送报文，但所有进入报文会被丢弃。一旦 FIFO 中的接收缓冲器重新可用，就能接收新的有效报文。

### 12.5.3 标识符接收滤波器

MSCAN 标识符接收寄存器（参见 12.3.11，“[MSCAN 标识符验收控制寄存器 \(CANIDAC\)](#)”）定义标准或扩展标识符（ID[10:0] 或 ID[28:0]）的可接受模式。这些位中的任意一个都可以在 MSCAN 标识符掩码寄存器中标志为“不比较”（参见 12.3.16，“[MSCAN 标识符掩码寄存器 \(CANIDMR0-CANIDMR7\)](#)”）。

一次滤波器匹配可由接收缓冲器已满标志（RXF = 1）和 CANIDAC 寄存器中的 3 个位（参见 12.3.11，“[MSCAN 标识符验收控制寄存器 \(CANIDAC\)](#)”）。通知给应用软件。这些标识符匹配标志（IDHIT[2:0]）能够清晰识别引起接收的滤波寄存器。它们简化了应用软件处理接收器中断来源的任务。如果出现一次以上的匹配（两个或多个滤波器匹配），低地址的寄存器具有优先权。

非常灵活的可编程通用标识符接收滤波器可以有效降低 CPU 的中断负载，该滤波器在经过编程后可在四种不同模式中运行（Bosch CAN 2.0A/B 协议规范）：

- 两个标识符接收滤波器，每个将应用于：
  - 扩展标识符的全部 29 位和 CAN2.0B 帧的以下位：
    - 远程发送请求 (RTR)
    - 标识符扩展 (IDE)
    - 替代远程请求 (SRR)
  - 标准标识符的 11 位，加上 CAN 2.0A/B 报文的 RTR 和 IDE 位<sup>1</sup>。这种模式为符合 CAN 2.0B 标准的长扩展标识符提供两个滤波器。图 12-39 显示第一个 32 位滤波器页（CANIDAR0 - CANIDAR3、CANIDMR0 - CANIDMR3）如何产生滤波器 0 匹配。同样，第二个滤波器页（CANIDAR4 - CANIDAR7、CANIDMR4 - CANIDMR7）产生滤波器 1 匹配。
- 4 个标识符接收滤波器，每个应用于：
  - a) 扩展标识符的 14 个最重要位，加上 CAN 2.0B 报文的 SRR 和 IDE 位，或
  - b) 标准标识符的 11 位、CAN 2.0A/B 报文的 RTR 和 IDE 位。图 12-40 显示第一个 32 位滤波器页（CANIDAR0 - CANIDA3、CANIDMR0 - CANIDMR3）如何产生滤波器 0 和 1 匹配。同样，第二个滤波器页（CANIDAR4 - CANIDAR7、CANIDMR4 - CANIDMR7）产生滤波器 2 和 3 匹配。
- 8 个标识符接收滤波器，每个应用于标识符的前 8 位。这种模式为符合 CAN 2.0A/B 的标准标识符或符合 CAN 2.0B 的扩展标识符的前 8 个位实施 8 个独立的滤波器。图 12-41 显示第一个 32 位滤波器页（CANIDAR0 - CANIDAR3、CANIDMR0 - CANIDMR3）如何产生滤波器 0-3 匹配。同样，第二个滤波器页（CANIDAR4 - CANIDAR7、CANIDMR4 - CANIDMR7）产生滤波器 4-7 匹配。
- 关闭滤波器。没有 CAN 报文被复制到前景缓冲器 RxFG，且从不设置 RXF 标志。

<sup>1</sup>. 尽管这种模式可以用于标准标识符，但我们还是建议为标准标识符使用 4 个或 8 个标识符接收滤波器。

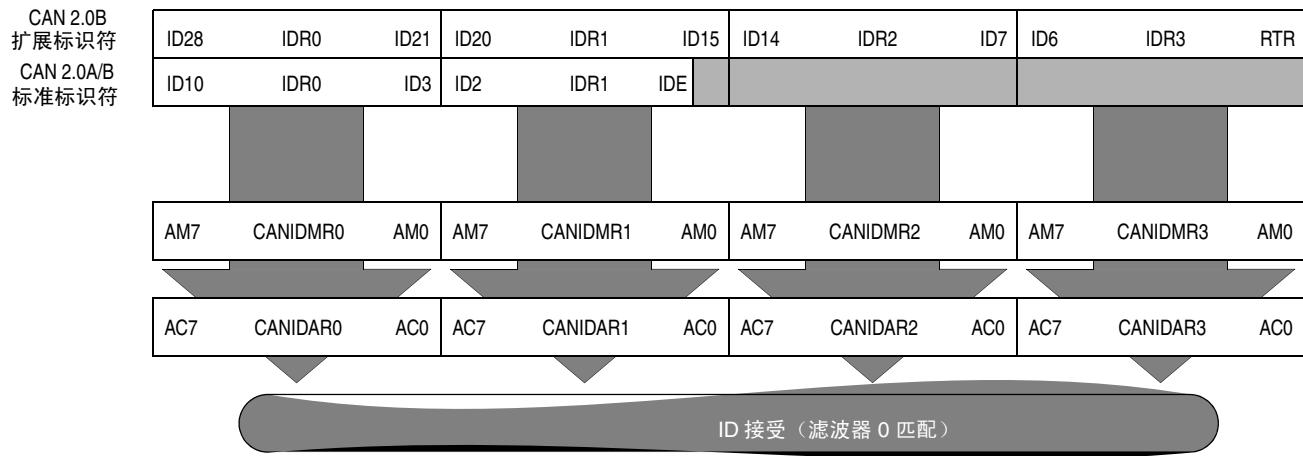


图 12-39. 32 位可屏蔽标识符接收滤波器

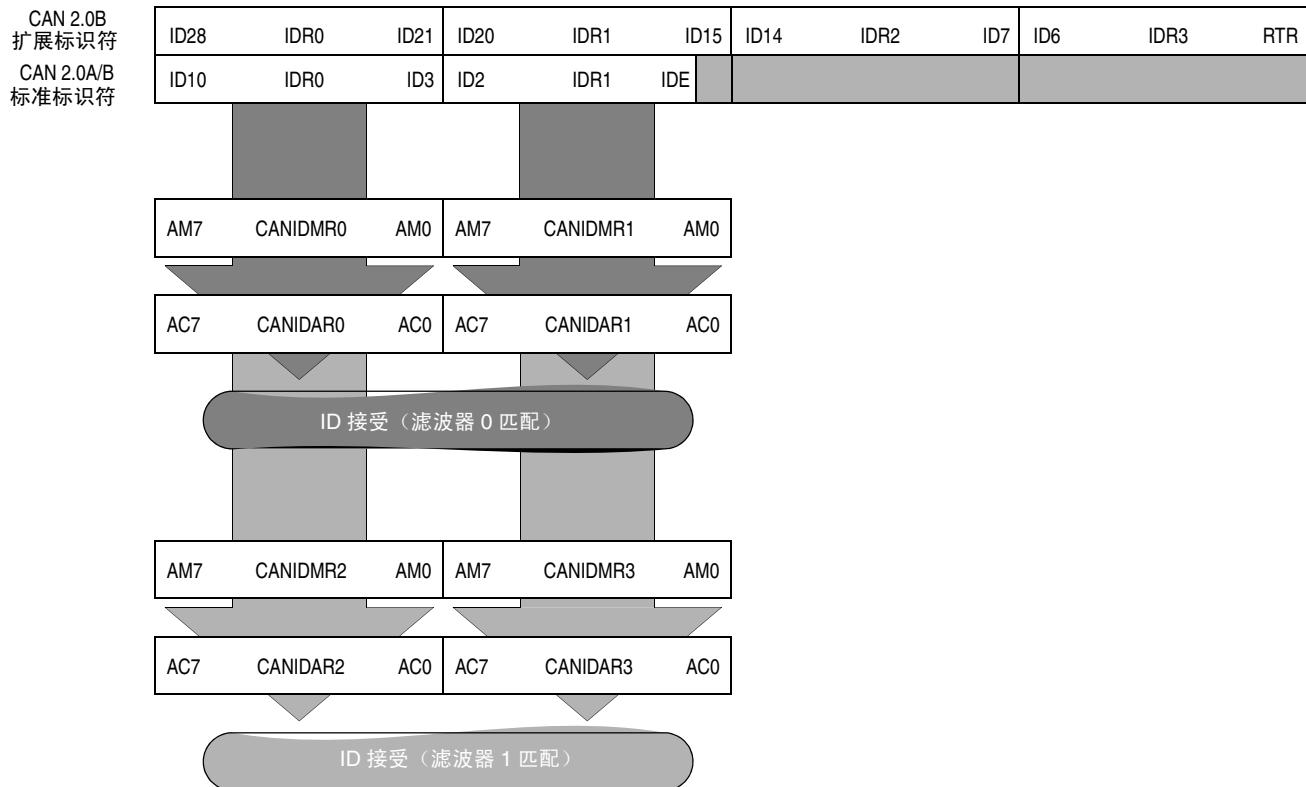


图 12-40. 16 位可屏蔽标识符接收滤波器

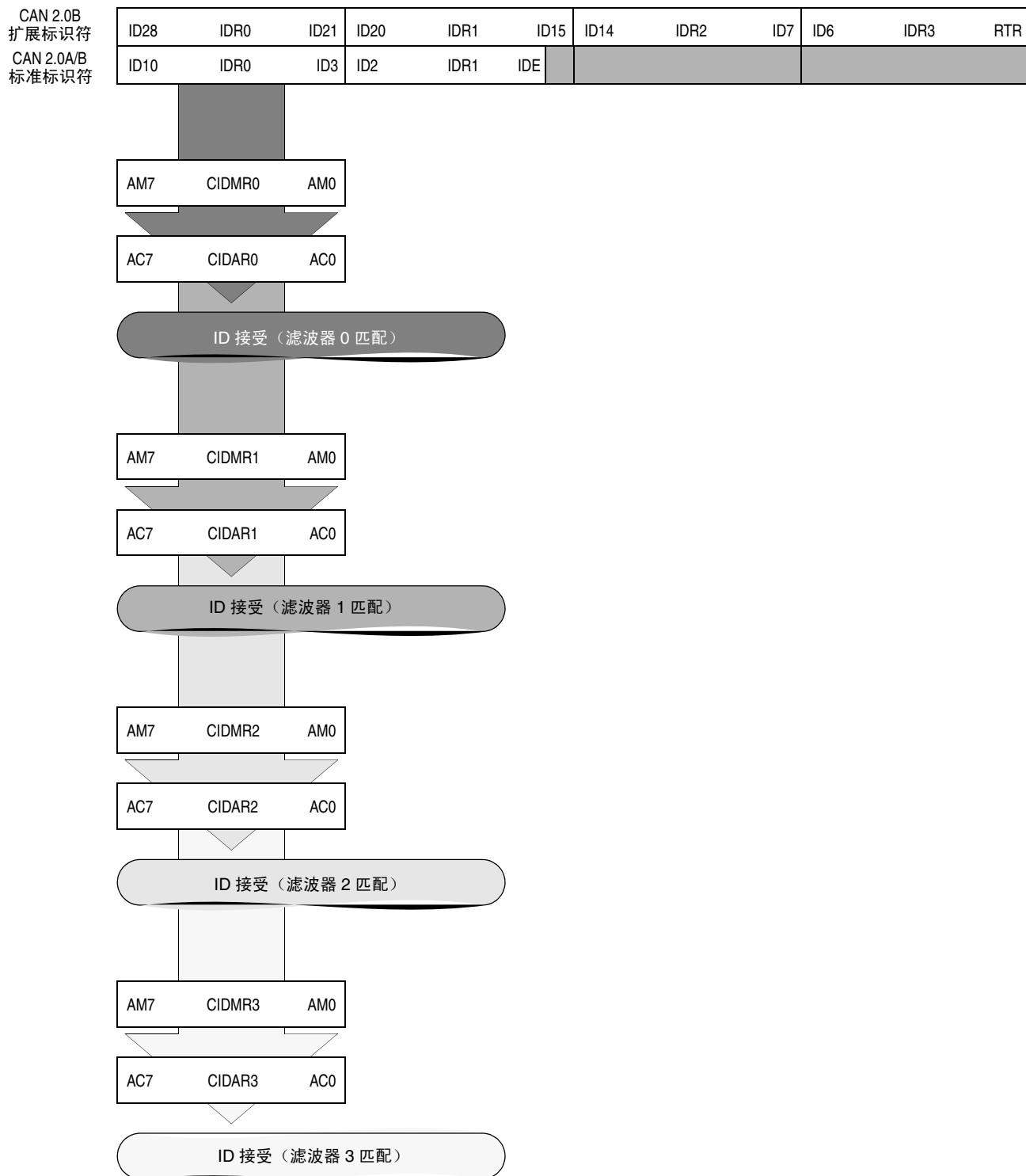


图 12-41. 8 位可屏蔽标识符接收滤波器

MSCAN 滤波器使用三组寄存器来提供滤波器配置。首先，CANIDAC 寄存器决定页配置中的滤波器大小和滤波器数量；其次，寄存器 CANIDMR0/1/2/3 通过把 ‘0’ 放在滤波器寄存器中的

适当位置来决定将比较的滤波器位；最后，寄存器 CANIDAR0/1/2/3 决定 CANIDMR0/1/2/3 所决定的位的值。

例如，当滤波器值为：

0001x1001x0

The CANIDMR0/1/2/3 寄存器将被配置为：

00001000010

因此，所有报文标识符位（除位 1 和位 6）会与 CANIDAR0/1/2/3 寄存器进行比较。这些寄存器将配置为：

00010100100

在这种情况下，位 1 和 6 设置为 ‘0’，但由于它们被忽略，因此效力等同于把它们设置为 ‘1’。

### 12.5.3.1 标识符接收滤波器示例

正如上面所描述的那样，滤波器是通过与 CAN 报文标识符字段中的个别位的比较进行工作的。滤波器将检查标准 CAN 报文标识符的 11 位的每个位。假设滤波器值为 0001x1001x0。在这个简单示例中，只可能有三种 CAN 报文。

滤波器值：0001x1001x0

报文 1: 00011100110

报文 2: 00110100110

报文 3: 00010100100

由于第三高位不是 ‘0’ - 001，报文 2 被拒绝。滤波器只是提供了 CPU 需要接收的报文组的便捷方式。对于扩展 CAN 报文标识符的全部 29 位，滤波器能辨认两组报文：一组是它接收的报文，一组是它拒绝的报文。此外，滤波器可以一分为二。这允许 MSCAN 只检查报文标识符的前 16 位，但允许让两个独立滤波器执行检查。见下面的示例：

滤波器值 A: 0001x1001x0

滤波器值 B: 00x101x01x0

报文 1: 00011100110

报文 2: 00110100110

报文 3: 00010100100

MSCAN 会接受所有三条报文。滤波器 A 像以前一样将接受报文 1 和 3，滤波器 B 接受报文 2。实践中，哪个滤波器接受报文并不重要，任意一个滤波器接受的报文都将放入输入缓冲器。一条报文可由多个滤波器接受。

### 12.5.3.2 协议违反保护

MSCAN 能够防止用户由于编程错误而意外违反 CAN 协议。保护逻辑实施以下功能：

- 接收和发送错误计数器不能写入或以别的方式操作。
- 当 MSCAN 在线时，控制 MSCAN 的配置的所有寄存器均不能被修改。MSCAN 必须处于初始化模式。CANCTL0/CANCTL1 寄存器中的相应 INITRQ/INITAK 握手位（参见 12.3.1，“MSCAN 控制寄存器 0 (CANCTL0)”）作为一个锁来保护以下寄存器：
  - MSCAN 控制 1 寄存器 (CANCTL1)
  - MSCAN 总线定时寄存器 0 和 1 (CANBTR0, CANBTR1)
  - MSCAN 标识符接收控制寄存器 (CANIDAC)
  - MSCAN 标识符接收寄存器 (CANIDAR0 – CANIDAR7)
  - MSCAN 标识符掩码寄存器 (CANIDMR0 – CANIDMR7)
- 当 MSCAN 进入节电模式或初始化模式时，TXCAN 管脚立即被强制进入隐性状态（参见 12.5.6，“MSCAN 断电模式”和 12.5.5，“MSCAN I 初始化模式”）。
- MSCAN 使能位 (CANE) 在正常系统操作模式下只能写入一次，从而为意外禁止 MSCAN 提供了进一步保护。

### 12.5.3.3 时钟系统

图 12-42 显示 MSCAN 时钟发生电路的结构。

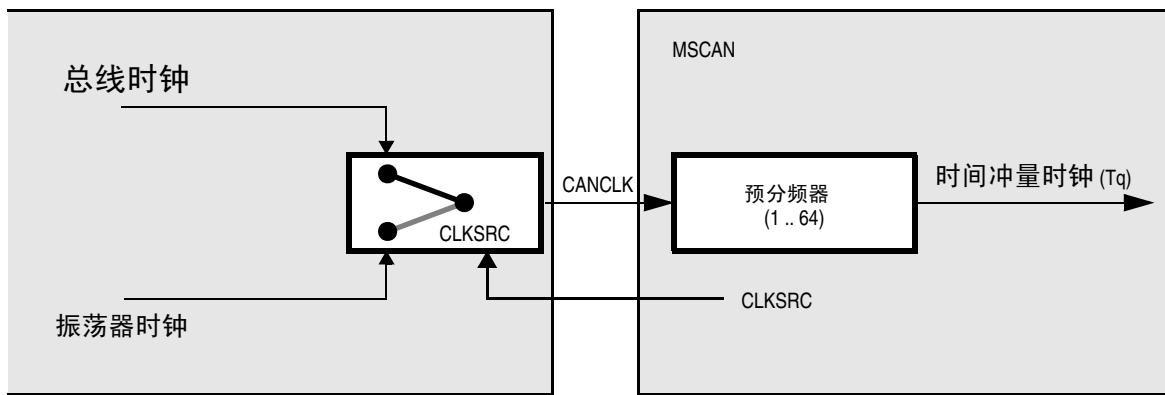


图 12-42. MSCAN 时钟机制

CANCTL1 寄存器 (12.3.2/-214) 中的时钟源位 (CLKSRC) 决定内部 CANCLK 是连接到晶体振荡器 (振荡器时钟) 输出还是连接到总线时钟。

必须选择能满足 CAN 协议的振荡器精度要求 (高达 0.4%) 的时钟源。此外，对于高 CAN 总线速率 (1 Mbps) 来说，要求 45%-55% 的时钟占空比。

如果总线时钟从 PLL 中生成，由于抖动，建议选择振荡器时钟而不是总线时钟，特别是以较快的 CAN 总线速率时。PLL 锁可能太宽，不能确保所需的时钟精度。

对于那些没有时钟和复位发生器 (CRG) 的微控制器，CANCLK 的驱动则来自晶体振荡器 (振荡时钟)。

可编程预分频器从 CANCLK 生成时间冲量 (Tq) 时钟。时间冲量是 MSCAN 所处理时间的原子单位。

等式 12-2

$$f_{Tq} = \frac{f_{CANCLK}}{(\text{Prescaler value})^l}$$

位时间再分成三段，如 Bosch CAN 规范所述（见 图 12-43）：

- **SYNC\_SEG**: 该段有一个长度固定的时间冲量，信号边沿预计出现在本段。
- 时段 1：本段包括 CAN 标准的 PROP\_SEG 和 PHASE\_SEG1。通过设置参数 TSEG1，使之包含 4-16 个时间冲量，可以对其进行编程。
- 时段 2：本段表示 CAN 标准的 PHASE\_SEG2。通过设置 TSEG2 参数，使之具有 2-8 个时间冲量长，可以对其进行编程。

等式 12-3

$$\text{Bit Rate} = \frac{f_{Tq}}{(\text{number of Time Quanta})}$$

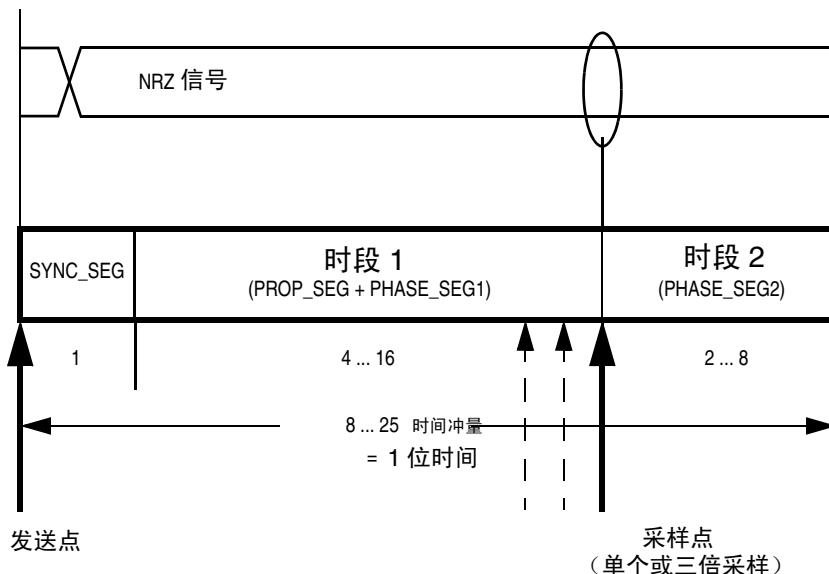


图 12-43. 位时间内的段

表 12-34. 时段句法

名称	描述
SYNC_SEG	系统希望该时段内在 CAN 总线上出现电平转换。
发送点	正处于发送模式的节点在该点上向 CAN 总线传输一个新值。
采样点	正处于接收模式的节点在该点采样 CAN 总线。如果选择了每位采样三次模式，那么该点标志第三采样点的位置。

同步跳转宽度（如需了解详细报文，参见 Bosch CAN 规范）可以通过设置 SJW 参数，在 1-4 个时间冲量范围内进行编程。

SYNC\_SEG、TSEG1、TSEG2 和 SJW 参数通过编程 MSCAN 总线计时寄存器（CANBTR0、CANBTR1）进行设置（参见 12.3.3，“[MSCAN 总线计时寄存 0 \(CANBTR0\)](#)”和 12.3.4，“[MSCAN 总线计时寄存器 \(CANBTR1\)](#)”）。

表 12-35 概括地描述了 CAN 段设置和相关参数值。

#### 注意

用户有责任确保位时间设置遵从 CAN 标准。

表 12-35. 遵从 CAN 标准的位时段设置

时段 1	TSEG1	时段 2	TSEG2	同步跳转宽度	SJW
5 .. 10	4 .. 9	2	1	1 .. 2	0 .. 1
4 .. 11	3 .. 10	3	2	1 .. 3	0 .. 2
5 .. 12	4 .. 11	4	3	1 .. 4	0 .. 3
6 .. 13	5 .. 12	5	4	1 .. 4	0 .. 3
7 .. 14	6 .. 13	6	5	1 .. 4	0 .. 3
8 .. 15	7 .. 14	7	6	1 .. 4	0 .. 3
9 .. 16	8 .. 15	8	7	1 .. 4	0 .. 3

## 12.5.4 运行模式

### 12.5.4.1 正常模式

在普通系统模式中，MSCAN 模块如本规范所述运行。

### 12.5.4.2 特殊模式

在特殊系统模式中，MSCAN 模块如本规范所述运行。

### 12.5.4.3 仿真模式

在所有仿真模式中，MSCAN 模块如在普通系统模式下一样，如本规范所述运行。

#### 12.5.4.4 监听模式

在可选的 CAN 总线监控模式（监听）中，CAN 节点能够接收有效数据帧和有效远程帧，但它只发送 CAN 总线上的“隐性”位。此外，它不能启动发送。如果 MAC 层需要发送“显性”位（ACK 位、超载标志或有效错误标志），该位将在内部传输，这样 MAC 层就监控该“显性”位，此时 CAN 总线在外部仍保持隐性状态。

#### 12.5.4.5 保密模式

MSCAN 模块没有保密功能。

#### 12.5.4.6 环回自测模式

环回自测模式独立于外部系统的连接，有时用于检查软件，以帮助隔离系统问题。在该模式中，发送器输出内部连接到接收器输入。**RXCAN** 输入管脚被忽略，**TXCAN** 输出进入隐性状态（逻辑 1）。发送时，MSCAN 的运行如常，把它自己发送的报文作为从远程节点接收的报文。在该状态中，MSCAN 将忽略 CAN 帧应答场中 ACK 间隙中发送的位，以确保正确接受它自己的报文。同时生成发送和接收中断。

### 12.5.5 低功耗选项

如果 MSCAN 禁止 (**CANE** = 0)，MSCAN 时钟停止，以节省功率。

如果 MSCAN 使能 (**CANE** = 1)，MSCAN 还有两种与正常模式相比功耗更低的模式：睡眠模式和断电模式，在睡眠模式中，可以通过停止所有时钟（除了 CPU 端访问寄存器的时钟外）来降低功耗。在节电模式中，所有时钟停止，没有功率消耗。

**表 12-36** 总结了 MSCAN 和 CPU 模式的各种组合。模式的特殊组合通过 **CSWAI** 和 **SLPRQ/SLPAK** 位上的设置决定。

对所有模式来说，只有当 MSCAN 处于睡眠模式 (**SLPRQ** = 1, **SLPAK** = 1)、唤醒功能使能 (**WUPE** = 1) 且唤醒中断使能 (**WUPIE** = 1) 时，MSCAN 唤醒中断才可能发生。

表 12-36. CPU 与 MSCAN 运行模式之比较

CPU 模式	MSCAN 模式			
	正常	功耗降低		
		睡眠	断电	禁止 (CANE=0)
运行	CSWAI = X <sup>1</sup> SLPRQ = 0 SLPAK = 0	CSWAI = X SLPRQ = 1 SLPAK = 1		CSWAI = X SLPRQ = X SLPAK = X
等待	CSWAI = 0 SLPRQ = 0 SLPAK = 0	CSWAI = 0 SLPRQ = 1 SLPAK = 1	CSWAI = 1 SLPRQ = X SLPAK = X	CSWAI = X SLPRQ = X SLPAK = X
Stop3		CSWAI = X <sup>2</sup> SLPRQ = 1 SLPAK = 1	CSWAI = X SLPRQ = 0 SLPAK = 0	CSWAI = X SLPRQ = X SLPAK = X
Stop1 or 2			CSWAI = X SLPRQ = X SLPAK = X	CSWAI = X SLPRQ = X SLPAK = X

<sup>1</sup> ‘X’ 表示 “不关心”<sup>2</sup> 为了从睡眠模式安全唤醒，进入 STOP3 模式前，必须把 SLPRQ 和 SLPAK 设置为 1。

### 12.5.5.1 运行模式中的操作

如表 12-37 所示，当 CPU 处于运行模式时，只有 MSCAN 睡眠模式作为低功率选项。

### 12.5.5.2 等待模式中的操作

WAIT 指令将 MCU 置入低功耗待机模式中。如果设置了 CSWAI 位，还可以在断电模式下节省更多电力，因为 CPU 时钟停止。退出断电模式后，MSCAN 重新启动其内部控制器，并再次进入正常模式。

当 CPU 处于等待模式时，MSCAN 可以在正常模式下运行并生成中断（寄存器可以通过背景调试模式访问）。根据 SLPRQ/SLPAK 和 CSWAI 位的值，MSCAN 也可以在任意一种低功率模式下运行，如在表 12-37 中看到的那样。

### 12.5.5.3 停止模式中的操作

STOP 指令将 CPU 置入低功耗待机模式中。在 STOP1 或 STOP2 模式中，MSCAN 在被置为断电模式，无论 SLPRQ/SLPAK 的值如何。在 STOP3 模式中，断电或睡眠模式由进入 STOP3 前设置的 SLPRQ/SLPAK 值决定。CSWAI 位在任意一种停止模式中都不发挥作用。参见表 12-37。

### 12.5.5.4 MSCAN 睡眠模式

通过在 CANCTL0 寄存器中确定 SLPRQ 位, CPU 可以请求 MSCAN 进入这种低功率模式。MSCAN 进入睡眠模式的时间取决于固定的同步延迟及其当前状态:

- 如果有一个或多个报文缓冲器等待发送 ( $\text{TXEx} = 0$ ), MSCAN 将继续发送, 直到所有发送报文缓冲器空 ( $\text{TXEx} = 1$ , 成功发送或中止), 然后再进入睡眠模式。
- 如果 MSCAN 正在接收, 它继续接收, 并且一旦 CAN 总线空闲, 就立即进入睡眠模式。
- 如果 MSCAN 既不在发送也不在接收, 它会立即进入睡眠模式。

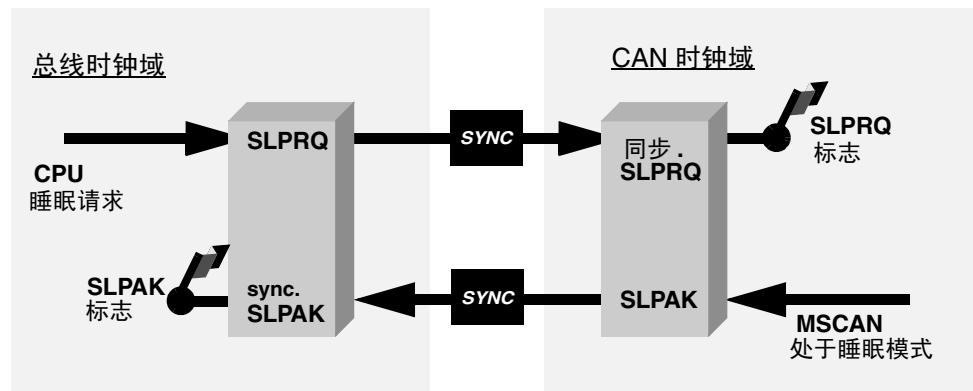


图 12-44. 睡眠请求 / 确认周期

#### 注意

应用软件必须避免建立发送 (通过清除一个或多个  $\text{TXEx}$  标志) 后立即请求睡眠模式 (通过设置  $\text{SLPRQ}$ )。MSCAN 是启动发送还是直接进入睡眠模式取决于实际的操作顺序。

如果睡眠模式激活,  $\text{SLPRQ}$  和  $\text{SLPAK}$  置位 (图 12-44)。应用软件必须把  $\text{SLPAK}$  作为请求 ( $\text{SLPRQ}$ ) 的握手标志, 以进入睡眠模式。

当处于睡眠模式 ( $\text{SLPRQ} = 1$ ,  $\text{SLPAK} = 1$ ) 时, MSCAN 停止其内部时钟。然而, CPU 访问寄存器的时钟继续运行。

如果 MSCAN 处于总线脱离状态, 由于时钟停止, 它就停止计数 11 个连续隐性位的 128 次出现。 $\text{TXCAN}$  管脚保持隐性状态。如果  $\text{RXF} = 1$ , 可以读取报文且可以清除  $\text{RXF}$ 。当处于睡眠模式时, 不会出现新报文转移到接收器 FIFO ( $\text{RxFG}$ ) 的前景缓冲器的情况。

访问发送缓冲器和清除相关  $\text{TXE}$  标志是允许的。当处于睡眠模式时, 不会出现报文中止的情况。

如果 CANCTL0 中的 WUPE 位还未置位, MSCAN 将屏蔽它在 CAN 上检测到的任何信号。 $\text{RXCAN}$  管脚因此在内部设置为隐性状态。这将把 MSCAN 锁在睡眠模式 (图 12-45)。WUPE 必须在进入睡眠模式前设置, 以便发挥作用。

只有当出现以下情形时，MSCAN 才能够退出睡眠模式（唤醒）：

- 出现 CAN 总线有效和 WUPE = 1
- 或
- CPU 清除 SLPRQ 位

#### 注意

在使能睡眠模式 ( $SLPRQ = 1$ ,  $SLPAK = 1$ ) 前，CPU 不能清除 SLPRQ 位。

唤醒之后，MSCAN 等待 11 个连续隐性位与 CAN 总线同步。因此，如果 MSCAN 被 CAN 帧唤醒，就不会收到该帧。

如果在进入睡眠模式前已经收到报文，接收报文缓冲器（RxFG 和 RxBG）就包含该报文。所有挂起操作在唤醒后执行，复制 RxBG 至 RxFG，报文中止和报文发送。如果在退出睡眠模式后 MSCAN 仍处于总线脱离状态，它将继续计数 128 次 11 个连续隐性位的出现。

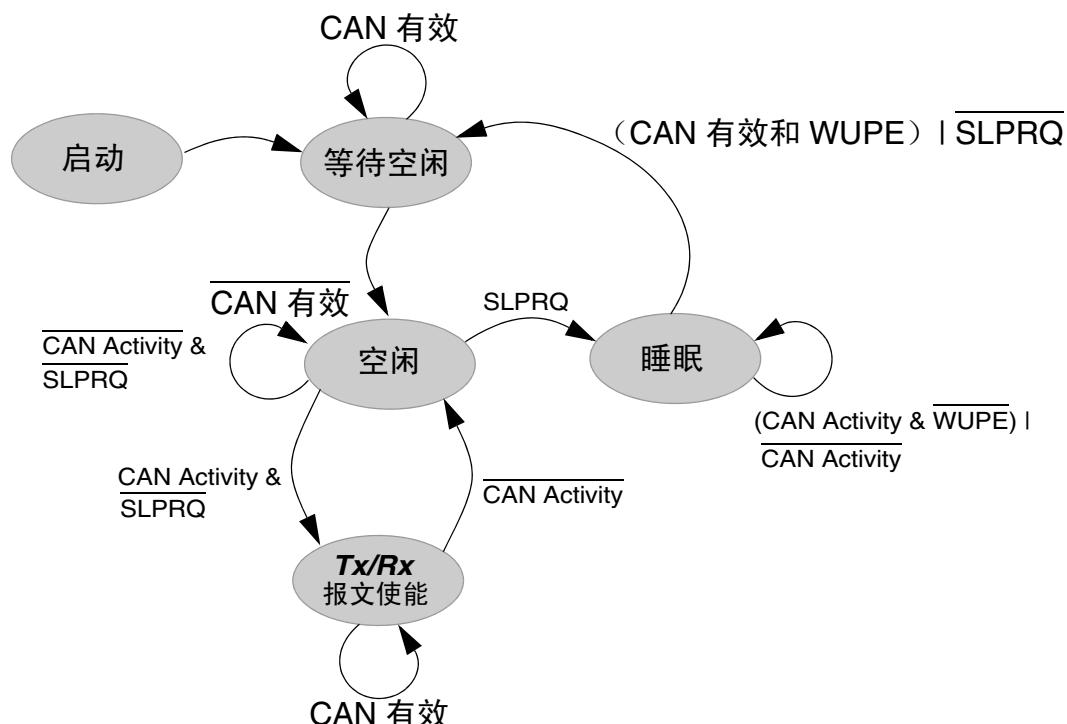


图 12-45. 进入 / 退出睡眠模式的简单状态转换

### 12.5.5.5 MSCAN I 初始化模式

在初始化模式中，正在进行的任何发送或接收都会立即中止，与 CAN 总线的同步丢失，并可能引起 CAN 协议违反。为了防止 CAN 总线系统出现严重的违反后果，MSCAN 立即驱动 TXCAN 管脚进入隐性状态。

#### 注意

进入初始化模式时，用户负责保证 MSCAN 不在工作态。推荐步骤是在 CANCTL0 寄存器中设置 INITRQ 位前，把 MSCAN 置入睡眠模式（SLPRQ = 1, SLPAK = 1）。否则，中止正在发送的报文可能导致错误情况，并影响到其他 CAN 总线节点。

在初始化模式中，MSCAN 被停止。然而，接口寄存器仍然可以访问。这种模式用来将 CANCTL0、CANRFLG、CANRIER、CANTFLG、CANTIER、CANTARQ、CANTAACK 和 CANTBSEL 寄存器复位为它们的默认值。此外，MSCAN 还使能 CANBTR0、CANBTR1 位计时寄存器的配置以及 CANIDAC、CANIDAR 和 CANIDMR 报文滤波器。参见 12.3.1，“[MSCAN 控制寄存器 0 \(CANCTL0\)](#)”，有关初始化模式的详细描述。

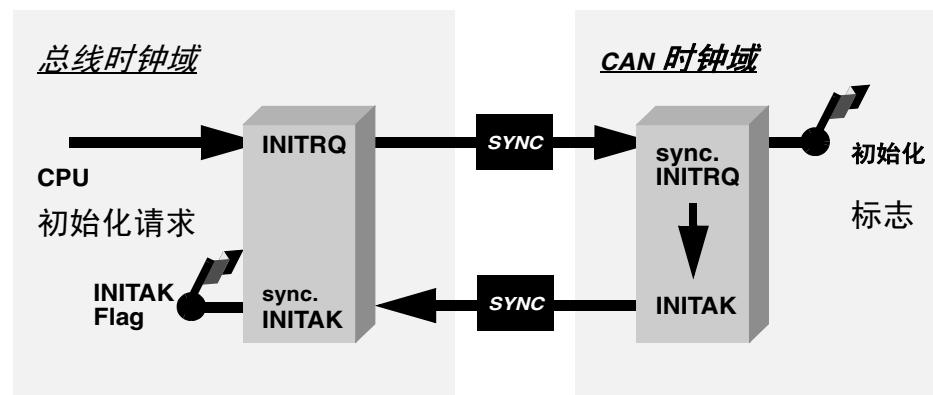


图 12-46. 初始请求 / 确认周期

由于 MSCAN 内的独立时钟域，INITRQ 必须通过采用特殊握手机制与所有时钟同步。这种握手导致了进一步的同步延迟（参见图 12-46., “[初始化请求 / 确认周期](#)”）。

如果 CAN 总线上没有正在传输的报文，最小延迟将是两个额外的总线时钟和三个额外的 CAN 时钟。当 MSCAN 的所有部件都处于初始化模式时，t.INITAK 标志置位。应用软件必须将 INITAK 作为握手标志，以便请求（INITRQ）进入初始化模式。

#### 注意

在使能初始化模式 (INITRQ = 1 and INITAK = 1) 前，CPU 不能清除 INITRQ。

### 12.5.5.6 MSCAN 断电模式

当出现以下情况时，MSCAN 处于断电模式（表 12-36）

- CPU 处于停止模式，  
或
- CPU 处于等待模式且设置了 CSWAI 位

当进入断电模式时，MSCAN 立即停止正在进行的所有发送和接收，可能造成违反 CAN 协议。为了防止 CAN 总线系统出现违反上述规则的严重后果，MSCAN 立即驱动 TXCAN 管脚进入隐性状态。

#### 注意

进入初始化模式时，用户负责保证 MSCAN 不在工作态。推荐步骤是在 CANCTL0 寄存器中设置 INTRQ 位前，把 MSCAN 置入睡眠模式（SLPRQ = 1, SLPAK = 1）。否则，中止正在发送的报文可能导致错误情况，并影响到其他 CAN 总线节点。

在断电模式中，所有时钟停止，且不能访问寄存器。如果在断电模式有效前 MSCAN 未处于睡眠模式，通电后该模块执行一个内部恢复周期。这会给模块再次进入正常模式带来某些固定延迟。

### 12.5.5.7 可编程唤醒功能

只要检测到 CAN 总线有效（参见 12.3.1，“MSCAN 控制寄存器 0 (CANCTL0)” 中的控制位 WUPE）。就可以对 MSCAN 进行编程以唤醒 MSCAN。当处于睡眠模式时，通过将低通滤波器功能应用于 RXCAN 输入，可以更改 CAN 总线检测的灵敏度（参见 12.3.2，“控制寄存器 1 (CANCTL1)” 中的控制位 WUPM）。

该功能可以用来防止由于 CAN 总线线路上的短脉冲而唤醒 MSCAN。例如，嘈杂环境中的电磁干扰可以引起尖峰脉冲。

### 12.5.6 复位初始化

各个单个位的复位状态在 12.3，“寄存器定义”，其中详细阐述了所有寄存器及其位字段。

### 12.5.7 中断

本小节描述了由 MSCAN 引发的所有中断，列出了使能位和触发标志。文中单独列出并描述了每个中断。

#### 12.5.7.1 中断运行描述

MSCAN 支持四个中断矢量（参见表 12-37），任意一个矢量都可以单独屏蔽。12.3.5，“MSCAN 接收器中断使能寄存器 (CANRIER)” 至 12.3.7，“MSCAN 发送器中断使能寄存器 (CANTIER)”。

#### 注意

专用的中断矢量地址在 Resets and Interrupts 章中有详细说明。

表 12-37. 中断矢量

中断源	CCR 掩码	本地使能
唤醒中断 (WUPIF)	1 位	CANRIER (WUPIE)
错误中断 (CSCIF, OVRIF)	1 位	CANRIER(CSCIE, OVRIE)
接入中断 (RXF)	1 位	CANRIER (RXFIE)
发送中断 (TXE[2:0])	1 bit	CANRIER (TXEIE[2:0])

### 12.5.7.2 发送中断

三个发送缓冲器中至少有一个空（未安排发送），并可以写入报文发送。空报文缓冲器的 TXEx 标志已置位。

### 12.5.7.3 接收中断

报文成功接收，并转移到接收器 FIFO 的前景缓冲器 (RxFG)。收到 EOF 符号后，立即生成该中断。RXF 标志已置位。如果接收器 FIFO 中有多条报文，一旦下一条报文转移到前景缓冲器，就立即设置 RXF 标志。

### 12.5.7.4 唤醒中断

如果在 MSCAN 内部睡眠模式期间 CAN 总线上有信号，就生成唤醒中断。WUPE（参见 12.3.1，“[MSCAN 控制寄存器 0 \(CANCTL0\)](#)”）必须使能。

### 12.5.7.5 错误中断

如果出现了接收器 FIFO 溢出、错误、警报或总线脱离情况，就出现错误中断。[12.3.4.1，“\[MSCAN 接收器标志寄存器 \\(CANRFLG\\)\]\(#\)”](#) 显示以下情况中的一种：

- 溢出 — 出现了如 [12.5.2.3，“\[接收结构\]\(#\)”所述的接收器 FIFO 的溢出情况。](#)
- CAN 状态变化 — 实际值控制着 MSCAN 的 CAN 总线状态。只要错误计数器进入关键范围 (Tx/Rx 警报、Tx/Rx 错误、总线脱离)，MSCAN 就标志错误情况。造成错误情况的状态变化用 TSTAT 和 RSTAT 标志表示 (参见 [12.3.4.1，“\[MSCAN 接收器标志寄存器 \\(CANRFLG\\)\]\(#\)”](#) 和 [12.3.5，“\[MSCAN 接收器中断使能寄存器 \\(CANRIER\\)\]\(#\)”](#))。

### 12.5.7.6 中断响应

中断与 [12.3.4.1，“\[MSCAN 接收器标志寄存器 \\(CANRFLG\\)\]\(#\)”](#) 或 [12.3.6，“\[MSCAN 发送器标志寄存器 \\(CANTFLG\\)\]\(#\)”](#) 中的一个或多个状态标志相关。CANRFLG 和 CANTFLG 中的标志必须在中断处理程序内复位。将 1 写入相应位来清零标志。如果中断条件仍然存在，标志不能清除。只要设置了相应标志中的一个，中断就产生。

#### 注意

必须确保 CPU 只清除引起当前中断的位。正是因为这个原因，不能用位操作指令 (BSET) 清除中断标志。这种指令可能造成意外清除进入当前中断服务程序后设置的中断标志。

### 12.5.7.7 从恢复停止或 等待

MSCAN 可以通过唤醒中断从停止或等待中恢复。只有当 MSCAN 在进入断电模式前处于睡眠模式 ( $SLPRQ = 1$ ,  $SLPAK = 1$ ) 时, 唤醒选项被使能 ( $WUPE = 1$ ), 唤醒中断使能 ( $WUPIE = 1$ ), 这种中断才能发生。

## 12.6 初始化 / 应用信息

### 12.6.1 MSCAN 初始化

系统复位后, 初始化 MSCAN 模块的流程如下:

1. 置位 CANE
2. 写入处于初始化模式的配置寄存器
3. 清除 INITRQ, 离开初始化模式, 进入正常模式

当 MSCAN 模块处于正常模式下, 需要更改只能在初始化模式中写入的寄存器:

1. CAN 总线空闲后, 通过设置 SLPRQ 并等待 SLPAK 进行确认, 将模块置入睡眠模式。
2. 进入初始化模式: 确定 INITRQ 并等待 INITAK
3. 写入处于初始化模式的配置寄存器
4. 清除 INITRQ, 离开初始化模式, 继续保持正常模式

### 12.6.2 总线脱离恢复

用户可配置总线脱离恢复功能。总线脱离状态既可以自动退出, 也可以在用户的请求下退出。

出于向前兼容原因, 复位后, MSCAN 默认为自动恢复。在这种情况下, 在计数 128 次 CAN 总线上 11 个连续隐性位的出现后, MSCAN 将重新变成 ERROR ACTIVE (详情请参见 Bosch CAN 规范)。

如果 MSCAN 配置为用于用户请求模式 [12.3.2, “控制寄存器 1 \(CANCTL1\)” 中设置的 BORM](#), 从总线脱离中恢复依赖于以下两个独立事件都成立后:

- 发现 128 次 CAN 总线上的 11 个连续隐性位
- [12.3.12, “MSCAN 其他寄存器 \(CANMISC\)” 中的 BOHOLD 已经被用户清除](#)

这两个事件的发生顺序任意。



# 第 13 章

## 串行外围器件接口 (S08SPIV3)

### 13.1 介绍

串行外围器件接口 (SPI) 模块提供 MCU 和外围器件间的全双工、同步和串行通信。这些外围器件可以包括其他微控制器、模数移位器、移位寄存器、传感器和存储器等。

SPI 运行在主模式中最高可运行在总线时钟除以 2 的波特率上，在辅模式中最高可运行在总线时钟除以 4 的波特率上。

MC9S08DZ60 系列 MCU 中的所有器件包含一个 SPI 模块，如下面的结构图所示。

#### 注意

在位更改为 CPHA 位的同时，确保 SPI 不得被禁止 (SPE=0)。这些更改应作为独立操作执行，否则就可能发生意外。

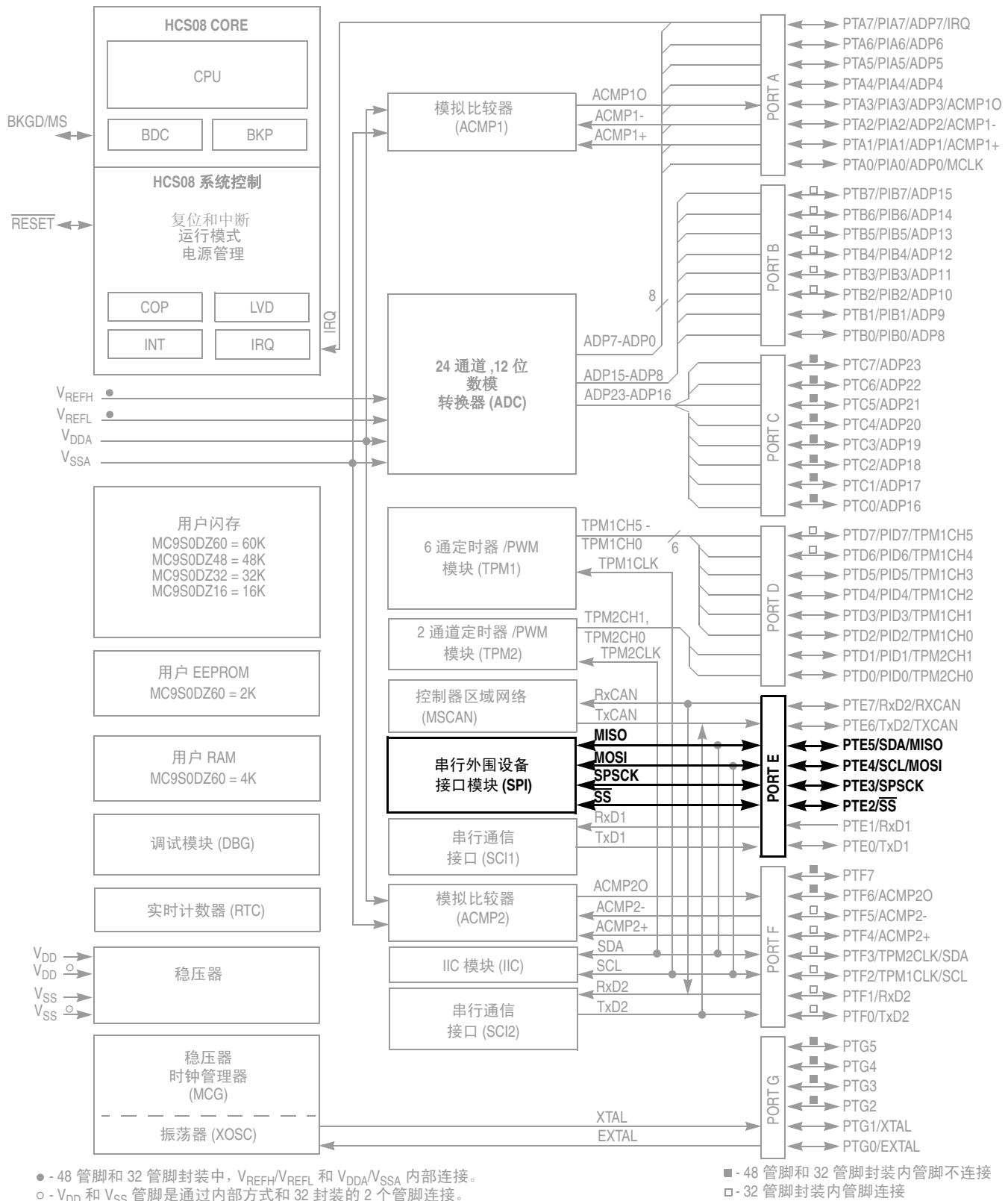


图 13-1. 强调 ADC 模块和管脚的 MC9S08DZ60 结构图

### 13.1.1 特性

SPI 模块的特性包括：

- 主或辅模式运行
- 全双工或单线双向选项
- 可编程发送波特率
- 双缓冲发送和接收
- 串行时钟相位和极性选项
- 辅选择输出
- 可选择的 MSB 先或 LSB 先移位

### 13.1.2 结构图

本小节包括显示 SPI 系统连接、SPI 模块内部结构、控制主模式波特率的 SPI 时钟分频器的结构图。

#### 13.1.2.1 SPI 系统结构图

图 13-2 显示主从安排中连接的两个 MCU 的 SPI 模块。主器件发起所有 SPI 数据传输。在传输过程中，主器件将数据（在 MOSI 管脚上）从辅器件中移出来，同时又把数据从辅器件中转入（在 MISO 管脚上）。这个传输有效交换位于两个 SPI 系统的 SPI 移位寄存器中的数据。

SPSCK 信号是来自主器件的时钟输出和到辅器件的时钟输入。辅器件必须由辅选择输入（SS 管脚）上的低电平进行选择。在该系统中，主器件把其 SS 管脚配置为可选的辅选择输出。

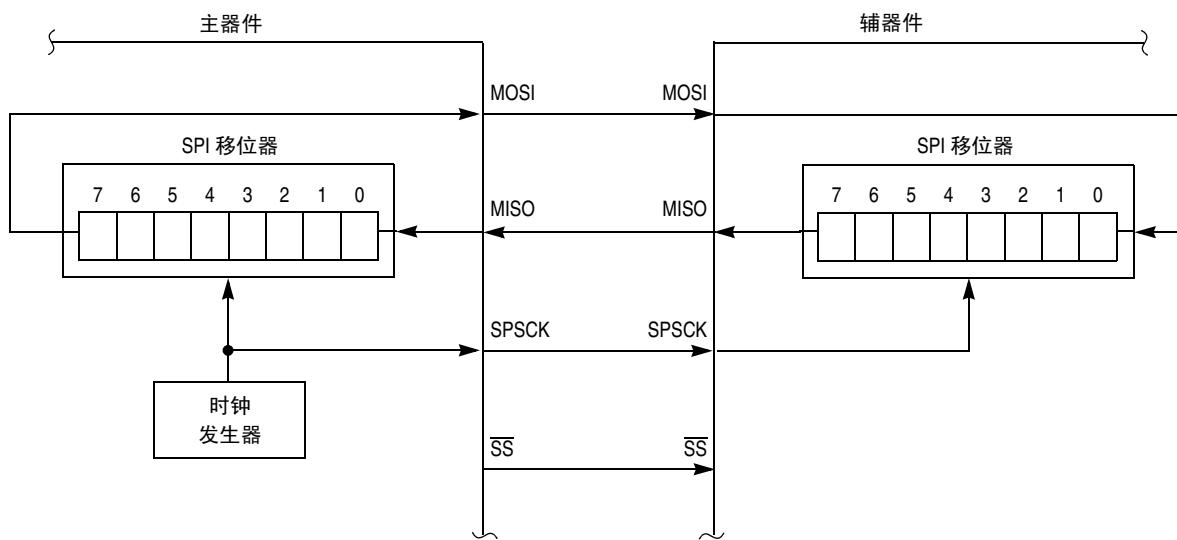


图 13-2. SPI 系统连接

SPI 系统最常见的使用包括连接简单移位寄存器，以增加输入或输出端口，或者连接小型外围器件，如串行 A/D 或 D/A 移位器。尽管图 13-2 显示了一个在两个 MCU 间交换数据的系统，但许多实际的系统的连接更简单，数据要么从主 MCU 单向传输到从 MCU，要么从从 MCU 单向传输到主 MCU。

### 13.1.2.2 SPI 模块结构图

图 13-3 是 SPI 模块的结构图。SPI 的中心元件是 SPI 移位寄存器。数据写入双缓冲发射器（写入 SPID），然后转移到位于数据传输起点的 SPI 移位寄存器。在数据字节中转换后，数据被传输到双缓冲接收器，在这里数据可以被读取（从 SPID 读取）。管脚复用逻辑控制着 MCU 管脚和 SPI 模块间的连接。

当 SPI 配置为主 SPI 时，时钟输出被路由到 SPSCK 管脚，移位器输出被路由到 MOSI，移位器输入从 MISO 管脚路由出来。

当 SPI 配置为从 SPI 时，SPSCK 管脚为时钟输出。MOSI 为移位输出，MISO 管脚为移位器输入。

在外部 SPI 系统，只需将所有 SPSCK 管脚彼此连接，所有 MISO 管脚连接起来，所有 MOSI 管脚连接起来就可以来。外围器件通常为这些管脚使用略有不同的名称。

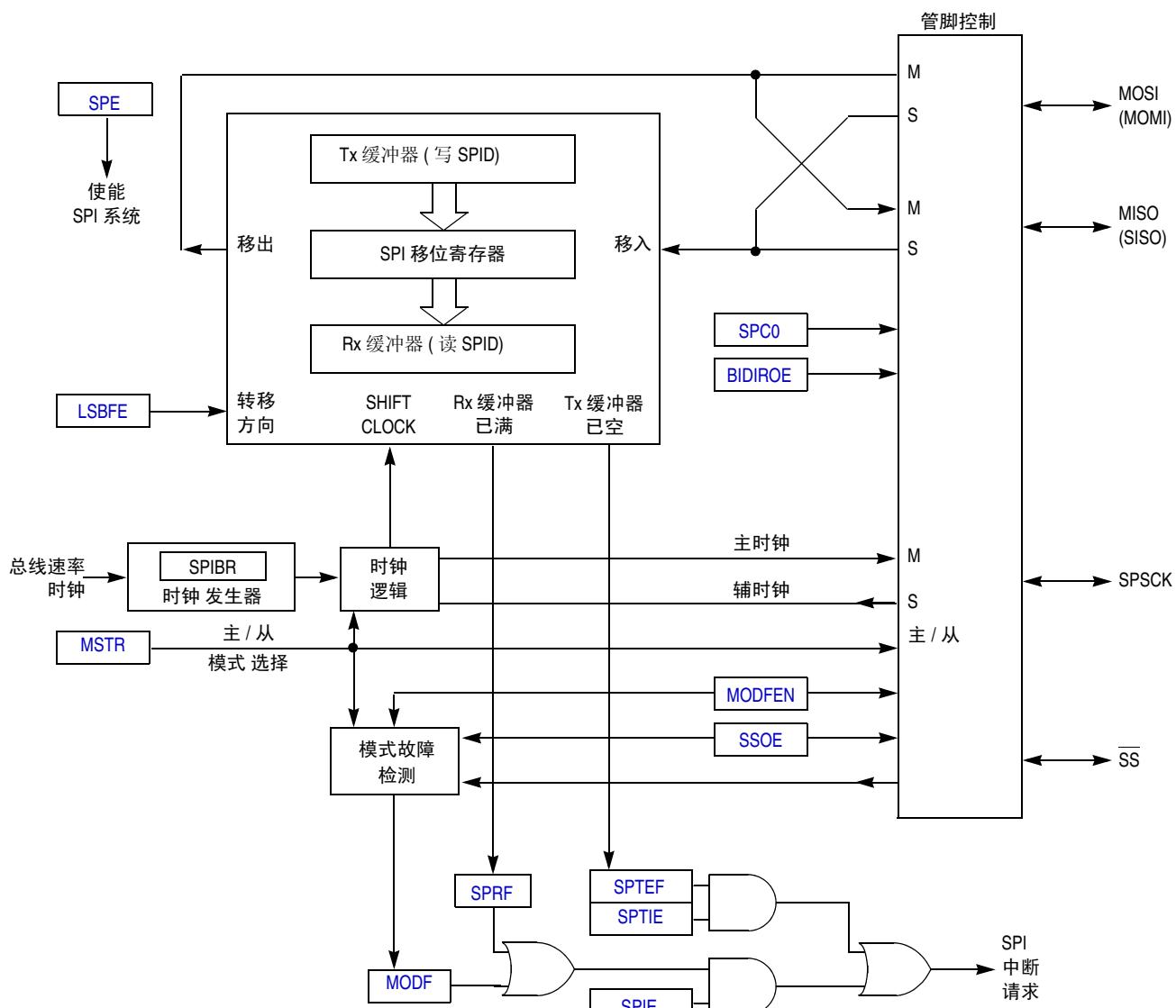


图 13-3. SPI 模块结构图

### 13.1.3 SPI 波特率生成

图 13-4 所示，SPI 波特率发生器的时钟源是总线时钟。三个预分频位 (SPPR3:SPPR2:SPPR1:SPPR0) 选择 1, 2, 3, 4, 5, 6, 7 或 8 作为预分频系数。3 个速率选择位 (SPR2:SPR1:SPR0) 分别用 2, 4, 8, 16, 32, 64, 128, 256 或 512 来除预分频器阶段的输出，以获取内部 SPI 主模式波特率时钟。

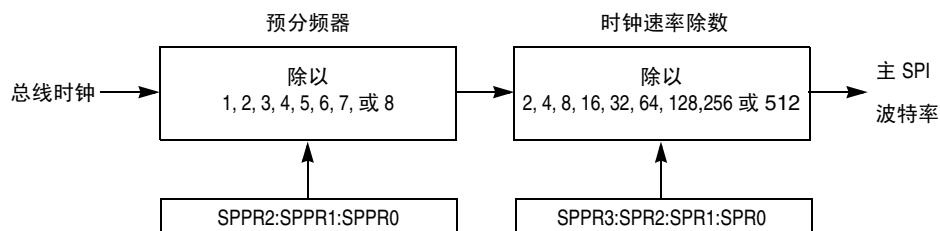


图 13-4. SPI 波特率生成

## 13.2 外部信号描述

SPI 共享 4 个端口管脚。这些管脚的功能取决于 SPI 控制位的设置。当 SPI 禁止 ( $SPE = 0$ ) 时，这 4 个管脚恢复为不受 SPI 控制的通用端口 I/O 管脚。

### 13.2.1 SPSCK — SPI 串行时钟

当 SPI 作为从 SPI 被使能时，该管脚是串行时钟输入。当 SPI 作为主 SPI 被使能时，该管脚是串行时钟输出。

### 13.2.2 MOSI — Master Data Out, Slave Data In

当 SPI 作为主 SPI 被使能且 SPI 管脚控制零 ( $SPC0 = 0$ ) 为 0 (不是双向模式) 时，该管脚是串行数据输出。当 SPI 作为从 SPI 被使能且  $SPC0 = 0$  时，该管脚为串行数据输入。如果  $SPC0 = 1$ ，选择单线双向模式并选择了主模式，该管脚就成为双向数据 I/O 管脚 (MOMI)。同样，双向模式输出使能位决定该管脚作为输入 ( $BIDIROE = 0$ ) 管脚还是输出管脚 ( $BIDIROE = 1$ )。如果  $SPC0 = 1$  且选择了辅模式，该管脚不会被 SPI 使用，并恢复为通用端口 I/O 管脚。

### 13.2.3 MISO — Master Data In, Slave Data Out

当 SPI 作为主 SPI 被使能且 SPI 管脚控制零 ( $SPC0 = 0$ ) 为 0 (不是双向模式) 时，该管脚是串行数据输入。当 SPI 作为从 SPI 被使能且  $SPC0 = 0$  时，该管脚是串行数据输出。如果  $SPC0 = 1$ ，选择单线双向模式并选择了辅模式，该管脚变成双向数据 I/O 管脚 (SISO)，双向模式输出使能位决定该管脚作为输入 ( $BIDIROE = 0$ ) 管脚还是输出管脚 ( $BIDIROE = 1$ )。如果  $SPC0 = 1$  且选择了辅模式，该管脚不会被 SPI 使用，并恢复为通用端口 I/O 管脚。

### 13.2.4 SS — 从选择

当 SPI 作为从 SPI 被使能时，该管脚是低电平有效的从选择输入。当 SPI 作为主 SPI 被使能且模式默认使能关闭 ( $MODFEN = 0$ ) 时，该管脚不被 SPI 使用，并恢复为通用端口 I/O 管脚。当 SPI 作为主 SPI 被使能时， $MODFEN = 1$ ，辅选择输出使能位决定该管脚是作为模式默认输入 ( $SSOE = 0$ ) 还是辅选择输出 ( $SSOE = 1$ )。

## 13.3 运行模式

### 13.3.1 处于停止模式的 SPI

SPI 在所有停止模式禁止，无论在执行 STOP 指令前的设置如何。在 STOP1 或 STOP2 模式期间，SPI 模块将完全关闭。一旦从 STOP1 或 STOP2 模式唤醒，SPI 模块将进入复位状态。在 STOP3 模式期间，SPI 模块的时钟暂停。寄存器没受到影响。如果通过复位退出 STOP3，SPI 被置复位状态。如果通过中断退出 STOP3，SPI 继续保持其进入 STOP3 时所处的状态。

## 13.4 寄存器定义

SPI 有 5 个 8 位寄存器来选择 SPI 选项、控制波特率、报告 SPI 状态和发送 / 接收数据。

如需了解所有 SPI 寄存器的绝对地址分配，参见本文 [Memory](#) 章的直接页面寄存器概述。本小节只提及了寄存器和控制位的名称，飞思卡尔提供的对照表或头文件用来把这些名称转换成适当的绝对地址。

### 13.4.1 SPI 控制寄存器 1 (SPIC1)

该读 / 写寄存器包括 SPI 使能控制、中断使能和配置选项。

	7	6	5	4		3	2	1	0
R W	SPIE	SPE	SPTIE	MSTR	CPOL	CPHA	SSOE	LSBFE	
复位	0	0	0	0	0	1	0	0	0

图 13-5. SPI 控制寄存器 1 (SPIC1)

表 13-1. SPIC1 字段描述

字段	描述
7 SPIE	<b>SPI 中断使能 (用于 SPRF 和 MODF)</b> — 这是 SPI 接收缓冲器已满 (SPRF) 和模式故障 (MODF) 事件的中断使能。 0 禁止从 SPRF 和 MODF 中中断 (使用轮询) 1 当 SPRF 或 MODF 为 1 时，请求硬件中断
6 SPE	<b>SPI 系统使能</b> — 禁止 SPI 将暂停正在进行的任何传输、清除数据缓冲器、初始化内部状态设备。 SPRF 被清除，并设置 SPTEF 来显示 SPI 发送数据缓冲器空。 0 SPI 系统禁止 1 SPI 系统使能
5 SPTIE	<b>SPI 发送中断使能</b> — 这是 SPI 发送缓冲器空 (SPTEF) 的中断使能位。 0 禁止从 SPTEF 中中断 (使用轮询) 1 当 SPTEF 为 1 时，请求硬件中断
4 MSTR	<b>主 / 辅模式选择</b> 0 SPI 模块配置为辅 SPI 器件 1 SPI 模块配置为主 SPI 器件
3 CPOL	<b>时钟极性</b> — 该位在来自主 SPI 或去往从 SPI 的时钟信号间有效地以串联方式放置一个取反逻辑。详情请参见 13.5.1，“ <a href="#">SPI 时钟格式</a> ”。 0 SPI 时钟高有效 (闲置低态) 1 SPI 时钟低有效 (闲置高态)
2 CPHA	<b>时钟相位</b> — 该位选择两种时钟格式的一种用于不同类型的同步串行外围器件。详情请参见 13.5.1，“ <a href="#">SPI 时钟格式</a> ”。 0 SPSCK 上的第一个边沿出现在 8 周期数据传输的第一个周期的中央 1 SPSCK 上的第一个边沿出现在 8 周期数据传输的第一个周期的起点

表 13-1. SPIC1 字段描述 (continued)

字段	描述
1 SSOE	辅选择输出使能 — 该位的使用结合 SPCR2 中的模式故障使能 (MODFEN) 位和主从 (MSTR) 控制位, 以确定 SS 管脚的功能, 如表 13-2 所示。
0 LSBFE	<b>LSB 先发 (移位器方向)</b> 0 SPI 串行数据传输始于最高位 1 SPI 串行数据传输始于最低位

表 13-2. SS 管脚功能

MODFEN	SSOE	主模式	辅模式
0	0	通用 I/O (非 SPI)	从选择输入
0	1	通用 I/O (非 SPI)	从选择输入
1	0	模式故障的 SS 输入	从选择输入
1	1	自动 SS 输出	从选择输入

**注意**

确保在位更改为 CPHA 位的同时 SPI 不得禁止 (SPE=0)。这些更改应作为独立操作执行, 否则可能发生意外。

**13.4.2 SPI 控制寄存器 2 (SPIC2)**

该读 / 写寄存器用来控制 SPI 系统的可选功能。位 7、6、5 和 2 不执行, 始终读为 0。

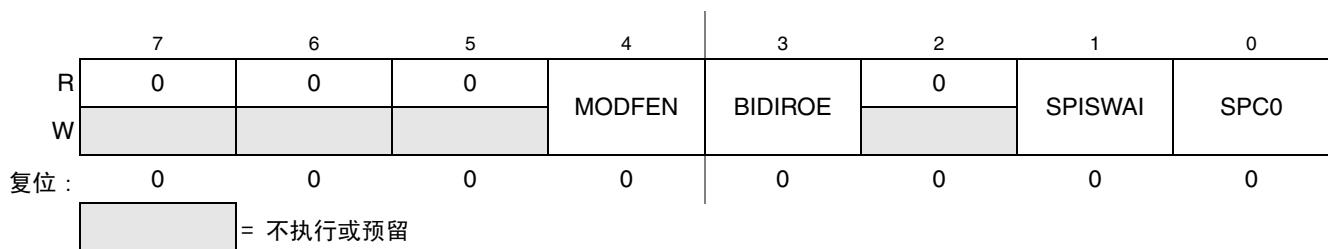


图 13-6. SPI 控制寄存器 2 (SPIC2)

表 13-3. SPIC2 寄存器字段描述

字段	描述
4 MODFEN	<b>主模式故障功能使能</b> — 当为辅模式配置 SPI 时, 该位没有意义或影响 (SS 管脚是从选择输入)。在主模式中, 该位决定 SS 管脚的使用方式 (如需了解更多信息, 参见表 13-2)。 0 模式故障功能禁止, 主 SS 管脚恢复为不受 SPI 控制的通用 I/O。 1 模式故障功能使能, 主 SS 管脚用作模式故障输入或辅选择输出
3 BIDIROE	<b>双向模式输出使能</b> — 双向模式由 SPI 管脚控制 0 (SPC0 = 1) 使能时, BIDIROE 决定 SPI 数据输出驱动器是否被使能为单个双向 SPI I/O 管脚。根据 SPI 是配置为主 SPI 还是从 SPI, 它将 MOSI (MOMI) 或 MISO (SISO) 管脚分别用作单个 SPI 数据 I/O 管脚, 当 SPC0 = 0, BIDIROE 没有意义或影响。 0 输出驱动器禁止, 因此 SPI 数据 I/O 管脚作为输入 1 SPI I/O 管脚作为输出使能

表 13-3. SPIC2 寄存器字段描述 (continued)

字段	描述
1 SPISWAI	<b>SPI 停止在等待模式中</b> 0 在等待模式中 SPI 时钟继续运行 1 当 MCU 进入等待模式时 SPI 时钟停止
0 SPC0	<b>管脚控制 0 — SPC0 位选择单线双向模式。</b> 如果 MSTR = 0 (辅模式), SPI 将 MISO (SISO) 管脚用于双向 SPI 数据传输。如果 MSTR = 1 (主模式), SPI 将 MOSI (MOMI) 管脚用于双向 SPI 数据传输。当 SPC0 = 1, BIDIROE 用于使能或禁止单个双向 SPI I/O 管脚的输出驱动器。 0 SPI 为数据输入和数据输出使用独立管脚 1 SPI 配置用于单线双向运行

### 13.4.3 SPI 波特率寄存器 (SPIBR)

该寄存器用来为 SPI 主器件设置预分频器和波特率系数。该寄存器可以随时读取或写入。

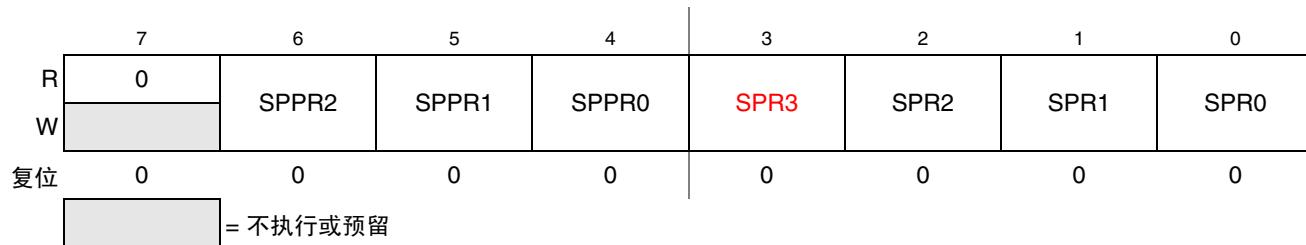


图 13-7. SPI 波特率寄存器 (SPIBR)

表 13-4. SPIBR 寄存器字段描述

字段	描述
6:4 SPPR[2:0]	<b>SPI 波特率预分频系数</b> — 该 3 位字段为 SPI 波特率预分频器选择 8 个系数中的一个, 如表 13-5 所示。该预分频器的输入是总线速率时钟 (BUSCLK)。该预分频器的输出驱动 SPI 波特率系数的输入 (图 13-4)。
2:0 SPR[3:0]	<b>SPI 波特率系数</b> — 该 4 位字段为 SPI 波特率系数选择 8 个系数中的一个, 如表 13-6 所示。该被除数的输入来自 SPI 波特率预分频器 (见图 13-4)。该被除数的输出是主模式的 SPI 波特率时钟。

表 13-5. SPI Baud 波特率预分频器系数

SPPR2:SPPR1:SPPR0	预分频器系数
0:0:0	1
0:0:1	2
0:1:0	3
0:1:1	4
1:0:0	5
1:0:1	6
1:1:0	7
1:1:1	8

表 13-6. SPI 波特率系数

SPR3:SPR2:SPR1:SPR0	速率系数
0:0:0:0	2
0:0:0:1	4
0:0:1:0	8
0:0:1:1	16
0:1:0:0	32
0:1:0:1	64
0:1:1:0	128
0:1:1:1	256
1:0:0:0	512
All other combinations	512

### 13.4.4 SPI 状态寄存器 (SPIS)

该寄存器有 3 个只读状态位。位 6, 3, 2, 1 和 0 不执行，始终读 0。写没有意义或影响。

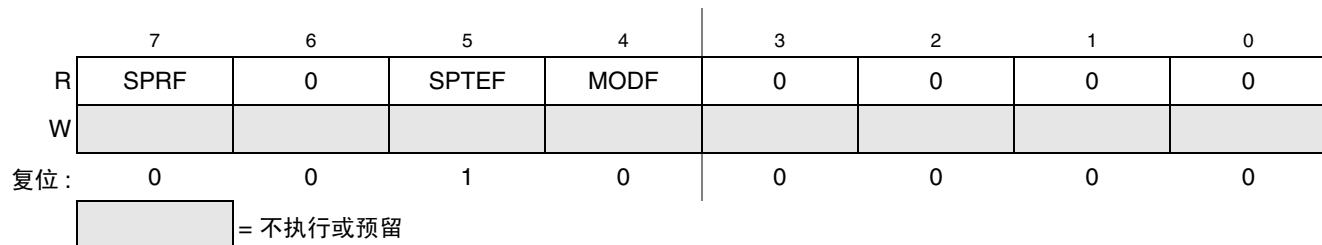


图 13-8. SPI 状态寄存器 (SPIS)

表 13-7. SPIS 寄存器字段描述

字段	描述
7 SPRF	<b>SPI 读缓冲器已满标记</b> — 在完成 SPI 传输时设置 SPRF，显示所接收数据可以从 SPI 数据寄存器（SPID）中读取。在设置 SPRF 的同时读取它可以清除 SPRF，然后读 SPI 数据寄存器。 0 接收数据缓冲器中无数据可用 1 接收数据缓冲器中有数据
5 SPTEF	<b>SPI 发送缓冲器空标记</b> — 该位在发送数据缓冲器中有空间时设置。通过读取已设置了 SPTEF 的 SPIS 可以清除该位，然后将数据值写入 SPID 上的发送缓冲器。在向 SPID 写入数据前，SPIS 必须被读到 SPTEF=1 标记，否则写 SPID 将被忽略。如果还设置了 SPIC1 中的 SPTIE 位，SPTEF 生成 SPTEF CPU 中断请求。当数据字节从发送缓冲器传输到发送移位寄存器时，SPTEF 被自动设置。对于闲置 SPI（发送缓冲器或移位寄存器中没有数据，且没有正在进行的传输）来说，写入 SPID 的数据会被立即传输到移位器，这样 SPTEF 就在两个总线周期内被设置，从而允许第二个 8 位数据值排队进入发送缓冲器。转换寄存器中的值传输完成后，来自发送缓冲器的排队值将自动移到移位器，同时设置 SPTEF，显示发送缓冲器为空。如果发送缓冲器中没有新数据在等待，SPTEF 只需保持设置，数据不会从缓冲器移位到移位器。 0 SPI 发送缓冲器不空 1 SPI 发送缓冲器空
4 MODF	<b>主模式故障标记</b> — 如果 SPI 配置为主 SPI 且从选择输入进入低态，就设置 MODF，显示其他一些 SPI 器件也配置为主 SPI。只有当 MSTR = 1、MODFEN = 1、SSOE = 0 时，SS 管脚才作为模式故障错误输入。否则，将永远不会设置 MODF。当 MODF 为 1 时读 MODF 可以清除 MODF，然后写入 SPI 控制寄存器 1 (SPIC1)。 0 无模式故障错误 1 检测到模式故障错误

### 13.4.5 SPI 数据寄存器 (SPID)

	7	6	5	4	3	2	1	0
R W	Bit 7	6	5	4	3	2	1	Bit 0
复位	0	0	0	0	0	0	0	0

图 13-9. SPI 数据寄存器 (SPID)

读该寄存器会返回从接收数据缓冲器读取的数据。写该寄存器会把数据写入发送数据缓冲器。当 SPI 配置为主 SPI 时，向发送数据缓冲器写入数据将发起一个 SPI 传输。

数据不应写入发送数据缓冲器，除非设置了 SPI 发送缓冲器空标记 (SPTEF)，显示发送缓冲器中有一定的空间来排队新的发送字节。

在设置了 SPRF 后、完成另外一个传输前，可以随时从 SPID 中读取数据。如果在新传输结束前未能从接收数据缓冲器读取数据，就会导致接收溢出，新传输的数据丢失。

## 13.5 功能描述

检查 SPI 发送缓冲器空标记 (SPTEF = 1) 然后将数据字节写入主 SPI 器件中的 SPI 数据寄存器，这样就发起 SPI 传输。当 SPI 移位寄存器可用时，该数据字节从发送数据缓冲器移到移位器，设置 SPTEF，显示缓冲器中有一定的空间来排队另外一个发送字符（如需要的话），SPI 串行传输开始。

在 SPI 传输过程中，数据在一个 SPSCK 边沿从 MOSI 管脚上进行采样（读取）和转移，半个 SPSCK 周期后改变 MOSI 管脚上的位值。在 8 个 SPSCK 周期后，主 SPI 的移位寄存器中的数据已经从 MOSI 管脚转移到从 SPI，同时 MISO 管脚里的 8 位数据被转移到主 SPI 的移位寄存器中。传输结束时，收到的数据字节从移位器转移到接收数据缓冲器，并设置 SPRF，显示数据可以通过读取 SPID 进行读取。如果传输结束时发送缓冲器中有另外一个数据字节在等待，它就被移到移位器，设置 SPTEF，启动新传输。

在正常情况下，SPI 数据首先传输最高位 (MSB)。如果设置了最低位先发使能位 (LSBFE)，SPI 数据首先移位 LSB。

当 SPI 配置为从 SPI 时，传输开始前其 SS 管脚必须驱动到低态，整个传输过程 SS 必须保持低态。如果选择了 CPHA = 0 的时钟格式，SS 必须在两次连续传输间被驱动到逻辑 1。如果 CPHA = 1，SS 在两次连续传输间必须保持低态。如需了解更多详细信息，参见 [13.5.1，“SPI 时钟格式”](#)。

因为发送器和接收器被双缓冲，因此第二个字节（除了当前正在移出的字节外）可以排队进入发送数据缓冲器，原来接收的字符可以放在接收数据缓冲器中，同时新字符被转移进来。

SPTEF 标记显示发送缓冲器何时有新字符空间。SPRF 标记显示已接收的字符何时出现在接收数据缓冲器。在下一次传输完成或导致接收溢出错误前，已接收字符必须从接收缓冲器读出（读 SPID）。

当出现接收溢出时，新数据丢失，因为接收缓冲器仍留有以前的字符，不准备接受新数据。这种溢出没有任何显示，因此应用系统设计人员必须确保在发起新传输前以前的数据已经从接收缓冲器中读出。

### 13.5.1 SPI 时钟格式

为了适应不同制造商的各种同步串行外围器件，SPI 系统有一个时钟极性（CPOL）位和一个时钟相位（CPHA）控制位，从四种时钟格式中选择一种进行数据传输。CPOL 有选择性地插入了与时钟串行的取反逻辑。CPHA 选择时钟和数据间的两种不同时钟相位关系。

图 13-10 显示了 CPHA = 1 时的时钟格式。在图的顶部，显示了 8 个位时间，作为参考。第一个位始于第一个 SPSCK 边沿，第八个位结束于第 16 个 SPSCK 边沿后的半个 SPSCK 周期。

MSB First 和 LSB First 线根据 LSBFE 中的设置显示了 SPI 数据位的顺序。SPSCK 极性的两个变化都显示了出来，但这两个波形中只有一个适用于特定传输，具体哪一个取决于 CPOL 中的值。SAMPLE IN 波形适用于辅器件的 MOSI 输入或主器件的 MISO 输入。MOSI 波形适用于主器件的 MOSI 输出管脚，MISO 波形适用于辅器件的 MISO 输出。SS OUT 波形适用于主器件的辅选择输出（如果 MODFEN, SSOE = 1）。在传输开始前的半个 SPSCK 周期，主 SS 输出信号进入有效低电平状态，在传输的第 8 个位时间结束时返回有效高态。SS IN 波形适用于辅器件的辅选择输入。

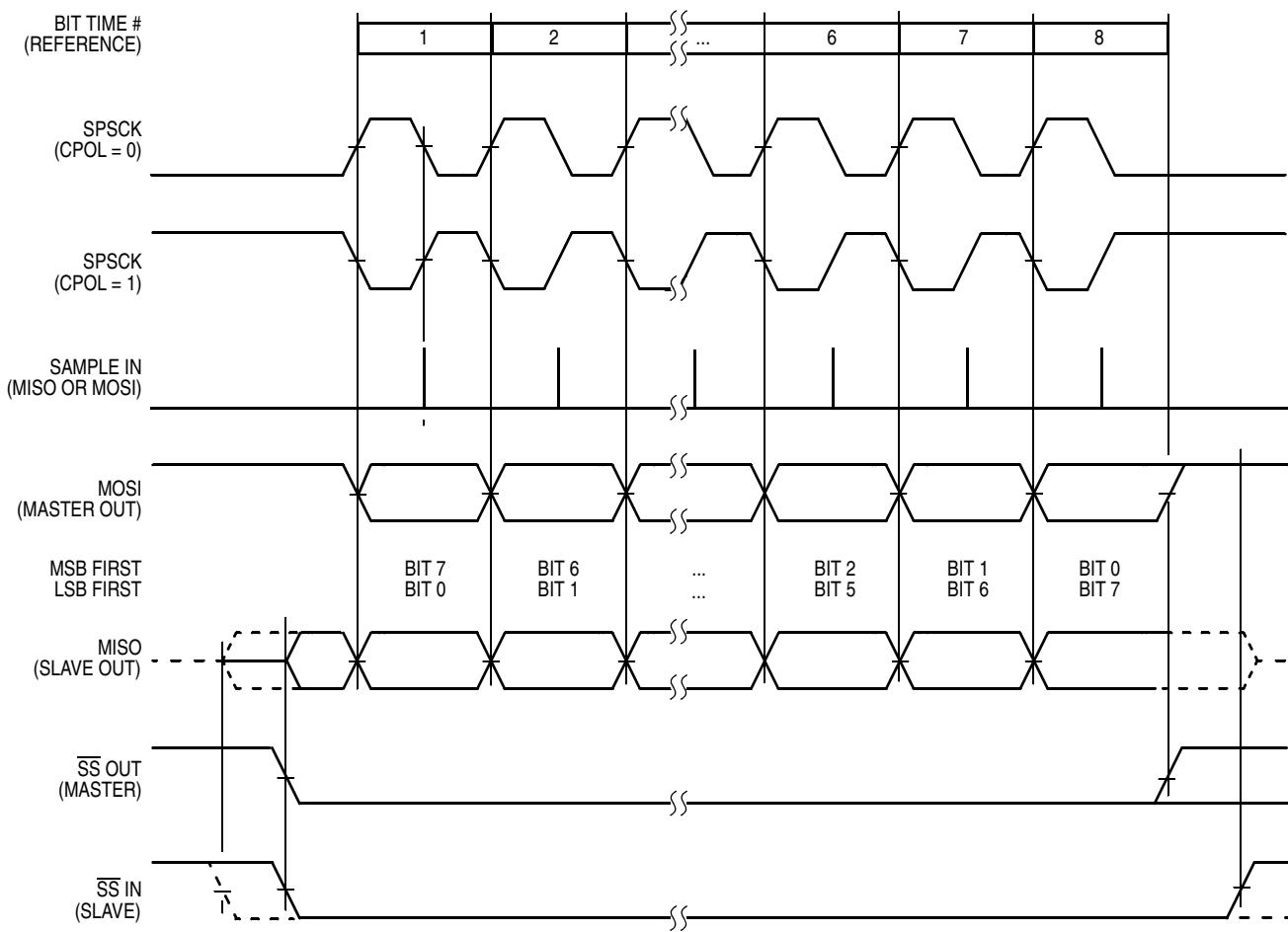


图 13-10. SPI 钟格式 (CPHA = 1)

若  $\text{CPHA} = 1$ ，则当处于活跃低态时，辅器件开始驱动其 MISO 输出，但直到出现第一个 SPSCK 边沿时才定义数据。第一个 SPSCK 边沿将数据的第一位从移位器转移到主 SPI 器件的 MOSI 输出和辅 SPI 器件的 MISO 输出。第二个 SPSCK 边沿促使主 SPI 器件和辅 SPI 器件分别在它们的 MISO 和 MOSI 输入上进行数据位值采样。在第三个 SPSCK 边沿，SPI 移位器移动 1 个位位置，移到刚刚采样的位值中，将第二个数据位值移出移位器的另一端，分别移到主 SPI 器件和辅 SPI 器件的 MOSI 和 MISO 输出。若  $\text{CPHA} = 1$ ，不需要辅 SPI 器件的 SS 输入在两个传输之间进入非激活的高电平状态。

图 13-11 显示了  $\text{CPHA} = 0$  时的时钟格式。在图的顶部，显示了 8 个位时间，作为参考。当选择辅时钟时（SS IN 进入低态），第一个位就开始，第八个位结束于最后一个 SPSCK 边沿。**MSB First** 和 **LSB First** 线根据 LSBFE 中的设置显示了 SPI 数据位的顺序。SPSCK 极性的两个变化都显示了出来，但这两个波形中只有一个适用于特定传输，具体哪一个取决于 CPOL 中的值。**SAMPLE IN** 波形适用于辅器件的 MOSI 输入或主器件的 MISO 输入。**MOSI** 波形适用于主器件的 MOSI 输出管脚，**MISO** 波形适用于辅器件的 MISO 输出。**SS OUT** 波形适用于主器件的辅选择输出（如果 MODFEN，SSOE = 1）。传输的第一位时间开始时，主器件的 SS 输出处于活跃低态，在传输的第 8 个位时间结束后的半个 SPSCK 周期时返回高态。**SS IN** 波形适用于辅器件的辅选择输入。

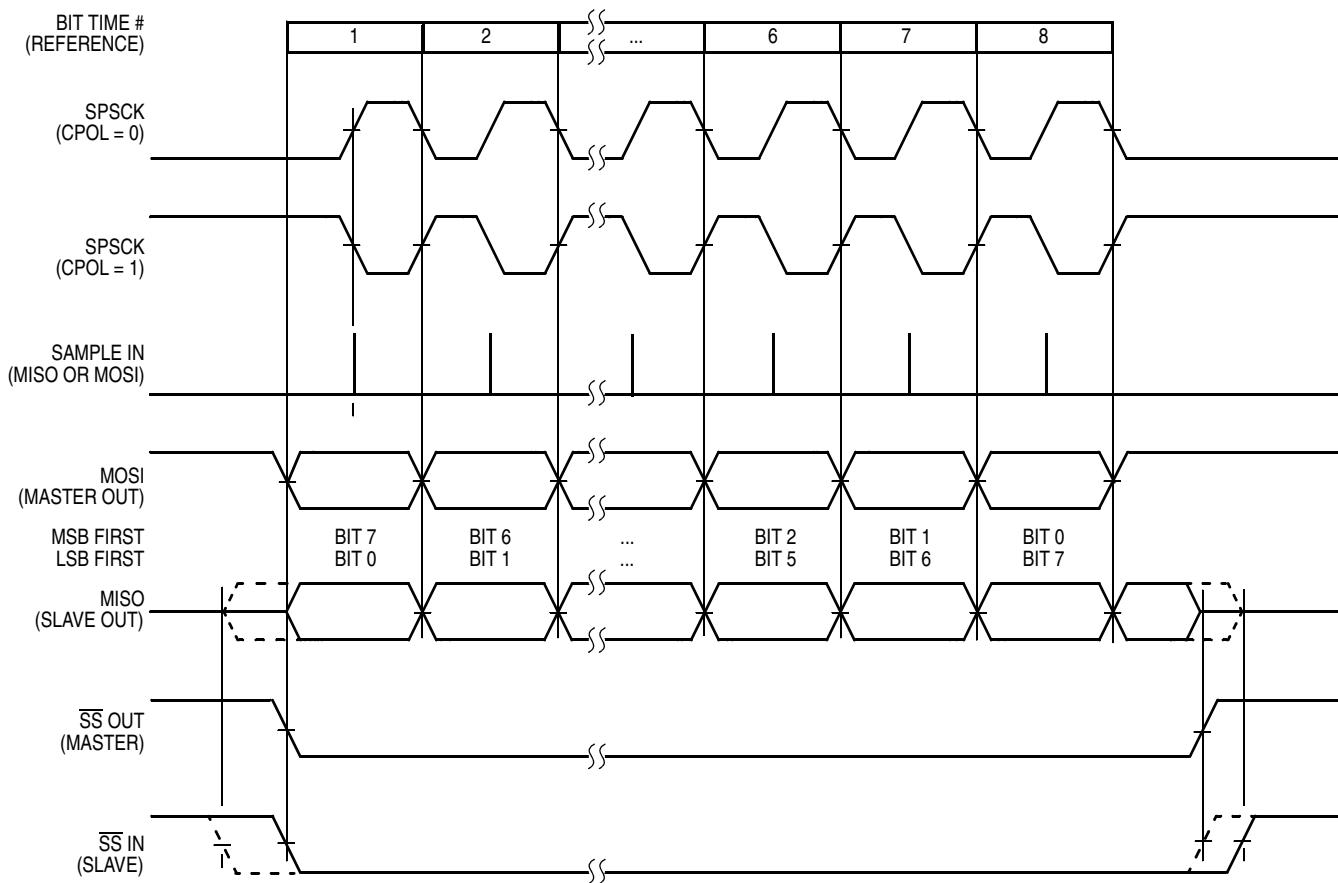


图 13-11. SPI 时钟格式 (CPHA = 0)

若  $\text{CPHA} = 0$ , 则当处于活跃低态时, 辅器件开始用第一个数据位值 (MSB 或 LSB, 取决于  $\text{LSBFE}$ ) 驱动其  $\text{MISO}$  输出。第一个  $\text{SPSCK}$  边沿促使主 SPI 器件和辅 SPI 器件分别在它们的  $\text{MISO}$  和  $\text{MOSI}$  输入管脚上进行数据位值采样。在第二个  $\text{SPSCK}$  边沿, SPI 移位器移动一个位位置, 移到刚刚采样的位值, 将第二个数据位值移出移位器的另一端, 分别移动到主辅 SPI 器件的  $\text{MOSI}$  和  $\text{MISO}$  输出。若  $\text{CPHA} = 0$ , 辅 SPI 器件的  $\text{SS}$  输入在两个传输间必须进入非激活的高电平状态。

### 13.5.2 SPI 中断

有三个标记位、两个中断屏蔽位和一个与 SPI 系统有关的中断向量。SPI 中断使能位 (**SPIE**) 允许来自 SPI 接收器已满标记 (**SPRF**) 和模式故障标记 (**MODF**) 的中断发生。SPI 发送中断使能位 (**SPTIE**) 允许来自 SPI 发送缓冲器空标记 (**SPTEF**) 的中断发生。当设置了一个标记位且设置了相关中断使能位, 硬件中断请求就被发送到 CPU。如果中断使能位被清除, 软件可以轮询相关标记位, 而不发生中断。SPI 中断服务程序 (**ISR**) 应检查标记位, 确定引起中断的事件。在从 **ISR** (通常在 **ISR** 起点的附近) 返回前, 服务程序还应清除标记位。

### 13.5.3 模式故障检测

当主 SPI 器件在  $\text{SS}$  管脚上检测到错误时 (假设  $\text{SS}$  管脚配置为模式故障输入信号), 就会发生模式故障并设置模式故障标记 (**MODF**)。当  $\text{MSTR} = 1$ , 设置模式故障位使能 (**MODFEN** = 1), 辅助 SPI 选择输出使能位清零 (**SSOE** = 0) 时,  $\text{SS}$  管脚配置为模式故障输入信号。

模式故障检测功能可用于一个以上的 SPI 器件可能同时成为主要 SPI 的系统中。当主 SPI 的  $\text{SS}$  管脚低时检测到错误, 就表明有其他 SPI 器件正尝试寻址该主 SPI, 就好像它是从 SPI 器件一样。这可以显示出一个有害的输出驱动器冲突, 因此当检测到这种错误时, 模式故障逻辑被设计成能够禁止所有 SPI 输出驱动器。

当检测到模式故障时, 设置 **MODF** 并清除 **MSTR**, 以便把 SPI 配置变回辅模式。**SPSCK**、**MOSI** 和 **MISO** 上的输出驱动器 (如果不是双向模式) 被禁止。

当 **MODF=1** 时读它可以清除 **MODF**, 然后写入 SPI 控制寄存器 1 (**SPIC1**)。用户软件应在把 SPI 变回主模式前, 确认已经更正了错误。

# 第 14 章

## 串行通信接口 (S08SCIV4)

### 14.1 介绍

MC9S08DZ60 系列中的所有 MCU 都包括 SCI1 和 SCI2。

#### 注意

MC9S08DZ60 系列器件在较高电压范围 (2.7 V~5.5 V) 中运行，不包括 STOP1 模式。请忽略 STOP1 参考。

#### 14.1.1 SCI2 配置信息

使用 SOPT1 寄存器中的 SCI2PS 位，SCI2 模块的 TxD2 和 RxD2 管脚的位置可以在软件控制下重新安排，如表 14-1 所示。SOPT1 中的 SCI2PS 位选择哪两个通用 I/O 口参予 SCI2 通讯。

表 14-1. SCI2 位置选项

SOPT1 中的 SCI2PS	TxD2 的端口管脚	RxD2 的端口管脚
0 (默认)	PTF0	PTF1
1	PTE6	PTE7

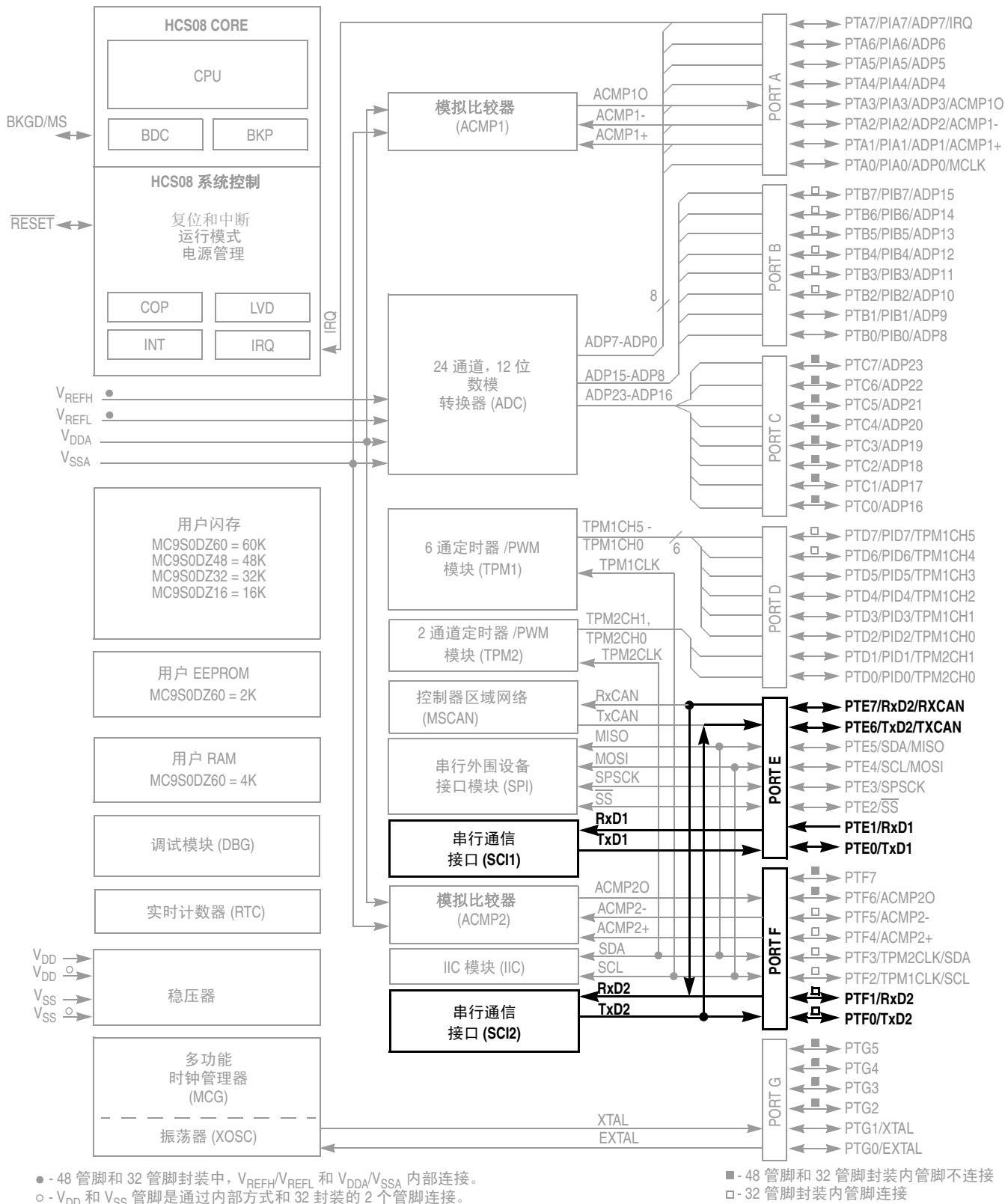


图 14-1. MC9S08DZ60 结构图

### 14.1.2 特性

SCI 模块的特性包括：

- 全双工、标准的不归零（NRZ）格式
- 具有独立使能的双缓冲发射器和接收器
- 可编程波特率（13 位模数分频器）
- 中断驱动型或轮询操作：
  - 发送数据寄存器空，发送完成
  - 接收数据寄存器已满
  - 接收溢出、奇偶效验错误、成帧错误和噪音错误
  - 闲置接收器检测
  - 接收管脚上的活动边沿
  - 支持 LIN 的断点检测
- 硬件奇偶效验生成和检查
- 可编程 8 位或 9 位字符长度
- 按闲置线路或地址标记的接收器唤醒
- 可选的 13 位中止字符生成 / 11 位中止字符检测
- 可选的发射器输出极性

### 14.1.3 运行模式

如需了解有关以下模式中的 SCI 操作的详细信息，参见 [14.3，“功能描述”](#)

- 8 位和 9 位数据模式
- 停止模式操作
- 循环模式
- 单线模式

### 14.1.4 结构图

[图 14-2](#) 显示了 SCI 的发射器部分。

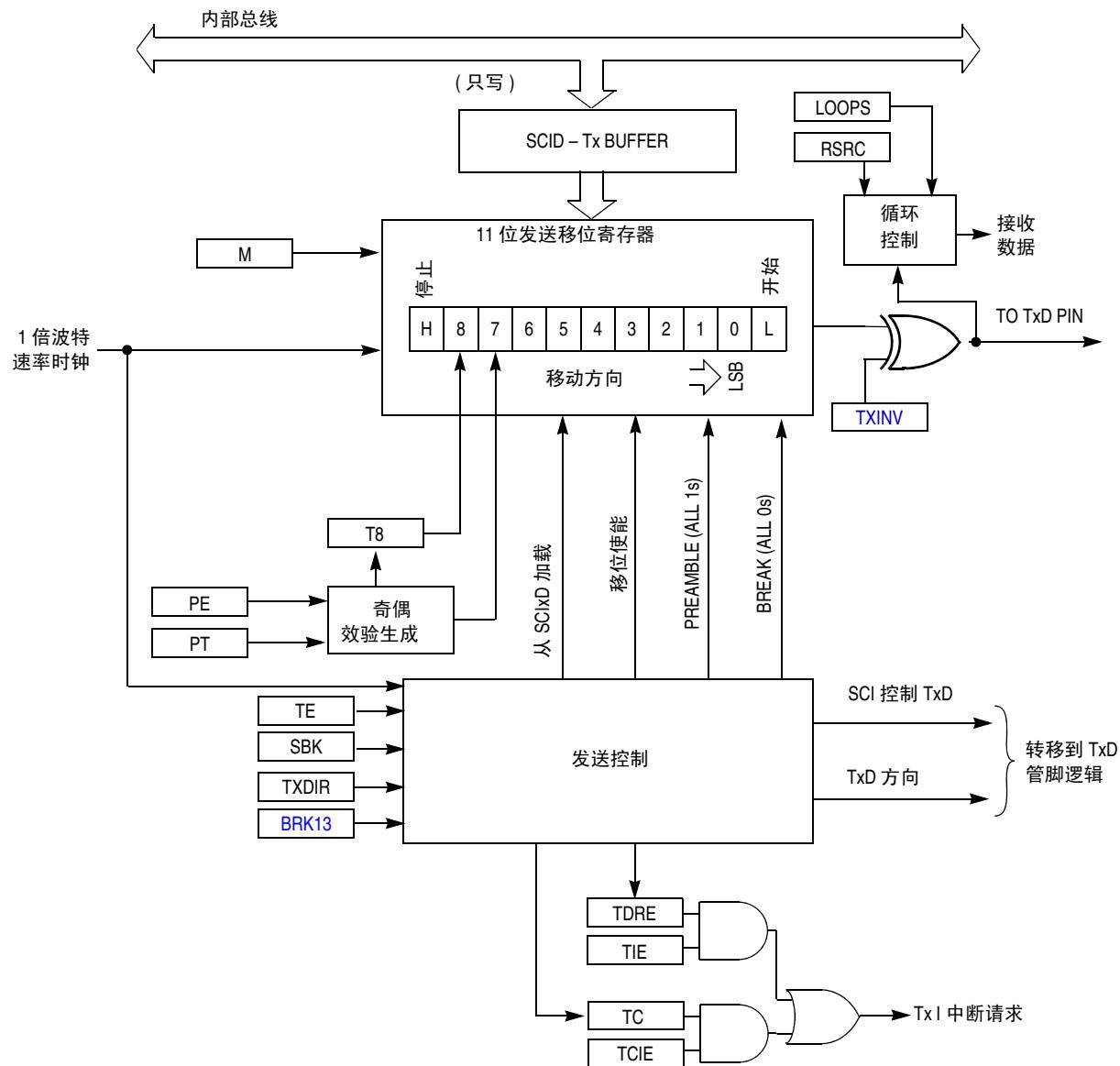


图 14-2. SCI 发射器结构图

图 14-3 SCI 接收器结构图

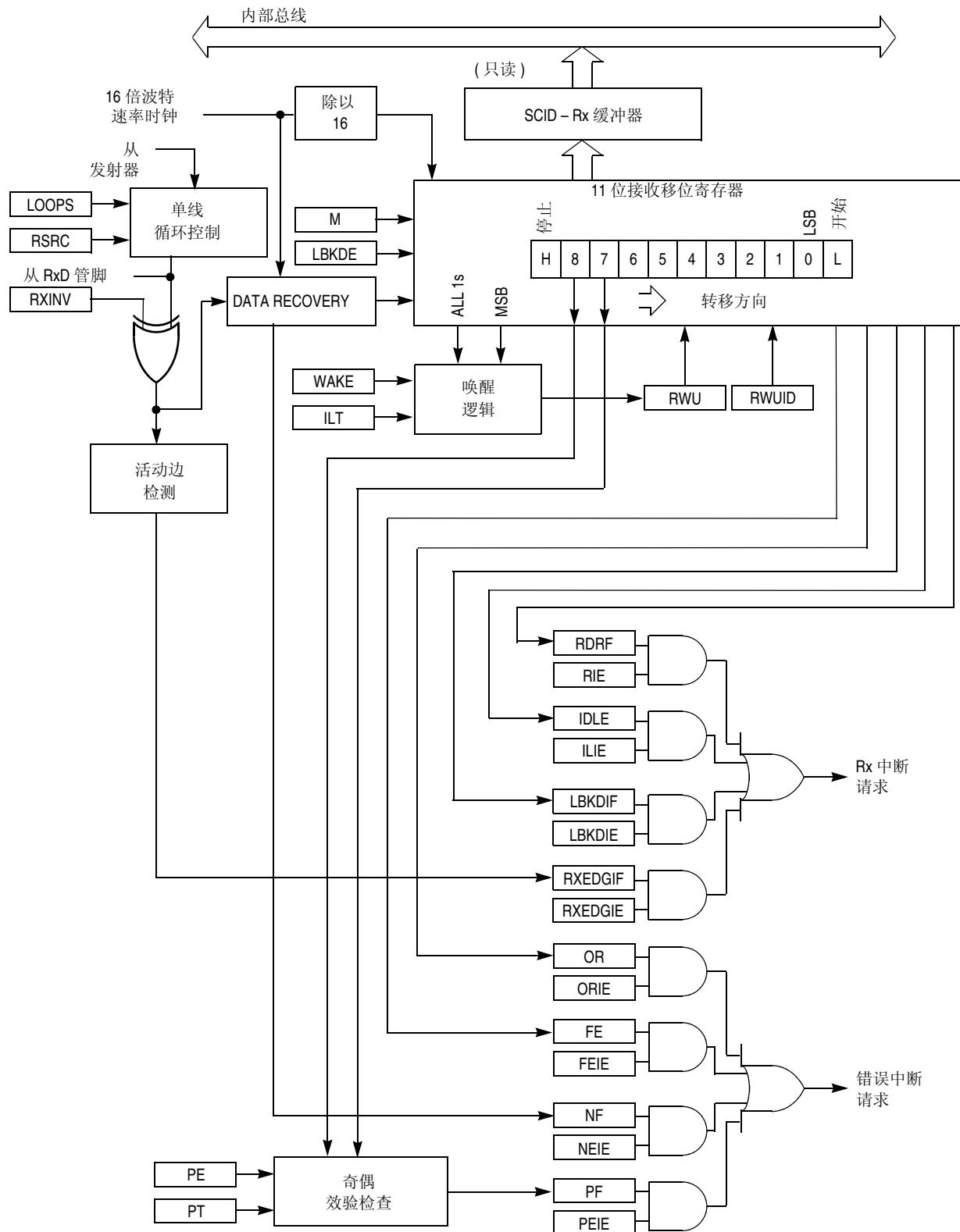


图 14-3. SCI 接收器结构图

## 14.2 寄存器定义

SCI 有 8 个 8 位寄存器，来控制波特速率、选择 SCI 选项、报告 SCI 状态和发送 / 接收数据。

如需了解所有 SCI 寄存器的绝对地址分配，参见本文 [Memory](#) 章的直接页面寄存器概述。本小节只提及了寄存器和控制位的名称，飞思卡尔提供的对照表或头文件用来把这些名称转换成适当的绝对地址。

### 14.2.1 SCI 波特率寄存器 (SCIxBDH, SCIxBDL)

这两个寄存器控制着生成 SCI 波特率的预分频系数。要更新 13 位波特率设置 [SBR12:SBR0]，首先写入 SCIxBDH，缓存生成的高位字节，然后再写入 SCIxBDL。SCIxBDH 中的值在写入 SCIxBDL 前不会变化。

SCIxBDL 复位为非零值，因此复位后，波特率发生器一直保持禁止，直到第一次使能接收器或发射器（SCIxC2 中的 RE 或 TE 位写为 1）。

	7	6	5	4		3	2	1	0
R	LBKDIIE	RXEDGIE	0	SBR12	SBR11	SBR10	SBR9	SBR8	
W				0	0	0	0	0	
复位	0	0	0	0	0	0	0	0	
	= 不执行或预留								

图 14-4. SCI 波特率寄存器 (SCIxBDH)

表 14-2. SCIxBDH 字段描述

字段	描述
7 LBKDIIE	<b>LIN 断点检测中断使能（用于 LBKDIF）</b> 0 来自 LBKDIF 的硬件中断禁止（使用轮询） 1 当 LBKDIF 标记为 1 时，硬件中断允许
6 RXEDGIE	<b>RxD 输入活动边中断使能（用于 RXEDGIF）</b> 0 禁止来自 RXEDGIF 的中断（使用轮询）。 1 当 RXEDGIF 标记为 1 时，允许硬件中断。
4:0 SBR[12:8]	波特率模数除数 — SBR[12:0] 中的 13 个位统称为 BR，它们为 SCI 波特率发生器设置模数除数系数。当 BR = 0，SCI 波特率发生器禁止，以降低电源电流。当 BR = 1~8191 时，SCI 波特率 = BUSCLK/ (16xBR)。也可参见中的 BR 位。 <a href="#">表 14-3</a> 中的 BR 位。

	7	6	5	4		3	2	1	0
R	SBR7	SBR6	SBR5	SBR4	SBR3	SBR2	SBR1	SBR0	
W					0	1	0	0	
复位	0	0	0	0	0	1	0	0	
	= 不执行或预留								

图 14-5. 波特率寄存器 (SCIxBDL)

表 14-3. SCIxBDL 字段描述

字段	描述
7:0 SBR[7:0]	波特率模数系数 — SBR[12:0] 中的这 13 个位统称为 BR，它们为 SCI 波特率发生器设置模数除数系数。当 BR = 0，SCI 波特率发生器禁止，以降低电源电流。当 BR = 1~8191 时，SCI 波特率 = BUSCLK / (16xBR)。也可参见表 14-2. 中的 BR 位。

## 14.2.2 SCI 控制寄存器 1(SCIxC1)

该读 / 写寄存器用于控制 SCI 系统的各种可选功能。

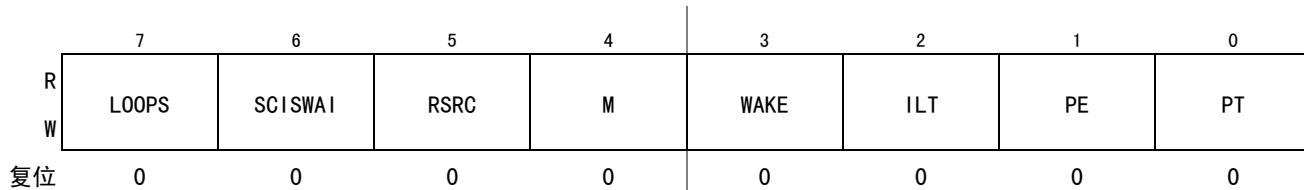


图 14-6. SCI 控制寄存器 1 (SCIxC1)

表 14-4. SCIxC1 字段描述

字段	描述
7 LOOPS	循环模式选择 — 在环回模式和正常的 2 管脚全双工模式之间选择。当 LOOPS = 1，发射器输出内部连接到接收器输入。 0 正常运行 — RxD 和 TxD 使用独立管脚。 1 循环模式或单线模式，发射器输出内部连接到接收器输入（见 <st-blue>RSRC 位）。SCI 不使用 RxD 管脚。
6 SCISWAI	等待模式中的 SCI 停止 0 SCI 时钟继续在等待模式中运行，因此 SCI 可以是唤醒 CPU 的中断源。 1 SCI 时钟在 CPU 处于等待模式时冻结。
5 RSRC	接收器源选择 — 该位没有任何意义或影响，除非 LOOPS 位设置为 1。当 LOOPS = 1 时，接收器输入内部连接到 TxD 管脚，RSRC 决定该连接是否也连接到发射器输出。 0 假设 LOOPS = 1，RSRC = 0 选择内部环回模式，SCI 不使用 RxD 管脚。 1 单线 SCI 模式，其中 TxD 管脚连接到发射器输出和接收器输入。
4 M	9 位或 8 位模式选择 0 正常 — 启动 + 8 个数据位 (LSB 先发) + 停止 1 接收器和发射器使用 9 位数据字符 启动 + 8 个数据位 (LSB 先发) + 第 9 个数据位 + 停止
3 WAKE	接收器唤醒方法选择 — 详情请参见 14.3.3.2，“接收器唤醒操作” 0 闲置线路唤醒 1 地址标记唤醒
2 ILT	闲置线路类型选择 — 将该位设置为 1，确保字符末端的停止位和逻辑 1 位不会计数闲置线路检测逻辑所需的逻辑高电平的 10 或 11 个位时间。如需了解更多信息，请参见 14.3.3.2.1，“闲置线路唤醒” 0 开始位后闲置字符位计数开始。 1 停止位后闲置字符位计数开始。

表 14-4. SCIxC1 字段描述

字段	描述
1 PE	奇偶效验使能 — 使能硬件奇偶效验生成和检查。当使能奇偶效验时，数据字符（第 8 或第 9 数据位）的最高位（MSB）视为奇偶校验位。 0 无硬件奇偶效验生成或检查。 1 奇偶效验使能。
0 PT	奇偶效验类型 — 如果使能奇偶效验（PE = 1），该位选择奇或偶效验。奇效验表示数据字符中 1 的总数（包括奇偶校验位）是奇数。偶数表示数据字符中 1 的总数（包括奇偶校验位）是偶数。 0 偶效验 1 奇效验

### 14.2.3 SCI 控制寄存器 2 (SCIxC2)

该寄存器可以随时读取或写入。

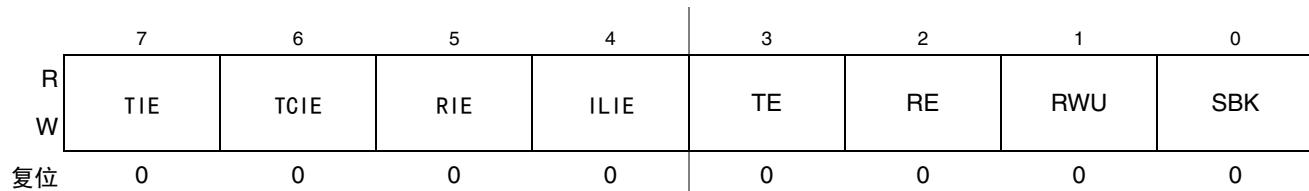


图 14-7. SCI 控制寄存器 2 (SCIxC2)

表 14-5. SCIxC2 字段说明

字段	描述
7 TIE	<b>发送中断使能 (用于 TDRE)</b> 0 来自 TDRE 的硬件中断禁止（使用轮询）。 1 当 TDRE 标记为 1 时允许硬件中断。
6 TCIE	<b>发送完成中断使能 (用于 TC)</b> 0 来自 TC 的硬件中断禁止（使用轮询）。 1 当 TC 标记为 1 时允许硬件中断。
5 RIE	<b>接收器中断使能 (用于 RDRF)</b> 0 来自 RDRF 的硬件中断禁止（使用轮询）。 1 当 RDRF 标记为 1 时允许硬件中断。
4 ILIE	<b>闲置线路中断使能 (用于 IDLE)</b> 0 来自 IDLE 的硬件中断禁止（使用轮询）。 1 当 IDLE 标记为 1 时允许硬件中断。
3 TE	<b>发射器使能</b> 0 发射器关闭。 1 发射器打开。 要使用 SCI 发射器，TE 必须是 1。当 TE = 1 时，SCI 强制 TxD 管脚作为 SCI 系统的输出。 当 SCI 配置用于单线运行（LOOPS = RSRC = 1）时，TXDIR 控制单 SCI 通信线路（TxD 管脚）上的流量方向。 当正在进行发送时通过写 TE = 0，然后写 TE=1，TE 也可以用来排队闲置字符。详情请 14.3.2.1，“发送中断和排队闲置”。 当 TE 写入 0，发射器保持对端口 TxD 管脚的控制，直到任何数据、队列闲置或队列中止字符在允许管脚恢复为通用 I/O 管脚前完成传输。

表 14-5. SCIxC2 字段说明 (continued)

字段	描述
2 RE	接收器使能 — 当 SCI 接收器关闭时, RxD 管脚恢复为通用端口 I/O 管脚。如果 LOOPS = 1, RxD 管脚恢复为通用 I/O 管脚, 即使 RE = 1。 0 接收器关闭。 1 接收器打开。
1 RWU	接收器唤醒控制 — 该位可以写入 1, 将 SCI 接收器置于待机状态, 等待所选唤醒条件的自动硬件检测。唤醒条件既可以是信息间的闲置线路 (WAKE = 0, 闲置线路唤醒), 也可以是某个字符中最高数据位中的逻辑 1 (WAKE = 1, 地址标记唤醒)。应用软件设置 RWU, (正常情况下) 且所选的硬件条件自动清除 RWU。如需了解更多信息, <a href="#">14.3.3.2, “接收器唤醒操作”</a> 。 0 正常的 SCI 接收器运行。 1 处于待机状态的 SCI 接收器等待唤醒条件。
0 SBK	发送中止字符 — 先后将 1 和 0 写入 SBK, 即在发送数据流中排入了一个中止字符。只要 SBK=1, 多达 10 或 11 (如果 BRK13 = 1, 则为 13 或 14 位) 位时间的逻辑 0 中止字符被加入队列。根据当前正在发送信息有关的 SBK 的设置和清除时间, 第二个中止字符可以在软件清除 SBK 前排入队列。如需了解更多信息, <a href="#">14.3.2.1, “发送中断和排队闲置”</a> 。 0 正常的发射器运行。 1 将发送的队列中止字符。

#### 14.2.4 SCI 状态寄存器 1 (SCIxS1)

该寄存器有 8 种只读状态标记。写没有影响, 特殊软件顺序 (不包括写入该寄存器) 用来清除这些状态标记。

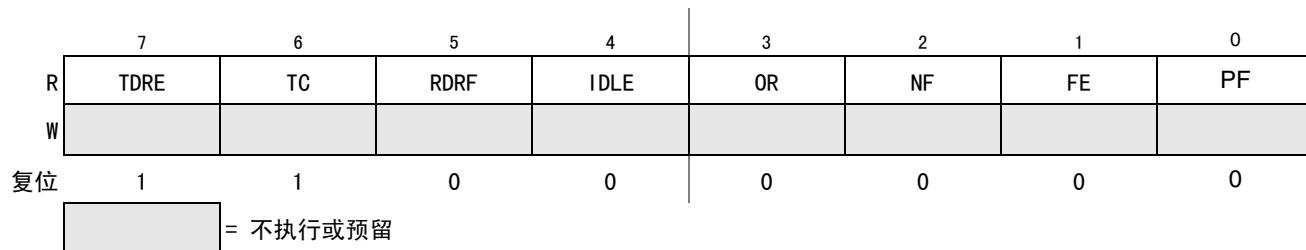


图 14-8. SCI 状态寄存器 1(SCIxS1)

表 14-6. SCIxS1 字段描述

字段	描述
7 TDRE	发送数据寄存器空标记 — TDRE 设置于复位, 当发送数据值从发送数据缓冲器传输到发送移位器时, 就在缓冲器中为新字符留出空间。要清除 TDRE, 当 TDRE = 1 时读 SCIxS1, 然后写入 SCI 数据寄存器 (SCIxD)。 0 发送数据寄存器 (缓冲器) 已满。 1 发送数据寄存器 (缓冲器) 为空。
6 TC	发送完成标记 — TC 设置于复位, 当 TDRE = 1 时, 无数据、前导信号或中止字符在发送。 0 发射器活动 (发送数据、前导信号或中止字符)。 1 发射器闲置 (发送活动完成) 当 TC = 1 时读取 SCIxS1 时可以自动清除 TC, 然后进行以下三种操作中一种: <ul style="list-style-type: none"><li>• 写入 SCI 数据寄存器 (SCIxD), 以发送新数据</li><li>• 通过把 TE 从 0 变为 1, 排队前导信号</li><li>• 将 1 写入 SCIxC2 中的 SBK, 排队中止字符。</li></ul>

表 14-6. SCIxS1 字段描述

字段	描述
5 RDRF	接收数据寄存器已满标记 — 当字符从接收移位器传输到接收数据寄存器 (SCIxD) 时, 设置 RDRF。要清除 RDRF, 当 RDRF = 1 时读 SCIxS1, 然后读 SCI 数据寄存器 (SCIxD)。 0 接收数据寄存器空。 1 接收数据寄存器已满。
4 IDLE	闲置线路标记 — 在一段时间的活动后, 当 SCI 接收线路已经闲置了一个全字符时间时, 就设置 IDLE。当 ILT = 0, 接收器在起始位后开始计数闲置位时间。因此, 如果接收字符都为 1, 这些位时间和停止位时间计数入接收器用于探测一个闲置线路所需逻辑高态 (10 或 11 个位时间, 取决于 M 控制位) 的全字符时间。当 ILT = 1, 接收器直到停止位后才开始计数闲置位时间。因此, 停止位和前一字符末端的任何逻辑高态位时间不会计数入接收器用于探测一个闲置线路所需逻辑高态的全字符时间。 要清除 IDLE, 当 IDLE = 1 时读取 SCIxS1, 然后读取 SCI 数据寄存器 (SCIxD)。清除 IDLE 后, 不能再次进行设置, 直到接收到新字符且已设置了 RDRF。IDLE 只设置一次, 即便接收线路闲置了很长一段时间。 0 没有检测到闲置线路 1 检测到闲置线路
3 OR	接收器溢出标记 — 当新的串行字符做好了传输到接收数据寄存器 (缓冲器) 的准备时, 但原来接收的字符还没有从 SCIxD 读取, 设置 OR。在这种情况下, 新字符 (和所有相关错误信息) 丢失, 因为没有空间将它们移到 SCIxD。要清除 OR, 当 OR = 1 时读 SCIxS1, 然后读 SCI 数据寄存器 (SCIxD)。 0 没有溢出 1 接收溢出 (新 SCI 数据丢失)
2 NF	噪音标记 — 接收器中采用的先进的采样技术在起始位中提取 7 个样本, 在每个数据位和停止位中提取 3 个样本。如果这些样本中任何一个样本与帧中任何位时间内的其余样本不一致, 就要在 RDRF 为这个字符而置 1 的同时设置标记 NF。要清除 NF, 读 SCIxS1, 然后读 SCI 数据寄存器 (SCIxD)。 0 没有检测到噪音 1 SCIxD 中的已接收字符中检测到噪音
1 FE	成帧错误标记 — 当接收器在应该是停止位的时候检测到逻辑 0 时, 同时设置 FE 和 RDRF。这表示接收器与字符帧没有完全统一。要清除 FE, 当 FE = 1 时读 SCIxS1, 然后读 SCI 数据寄存器 (SCIxD)。 0 未检测到成帧错误, 这不能保证成帧正确。 1 成帧错误。
0 PF	奇偶效验错误标记 — 当奇偶效验使能 (PE = 1) 且已接收字符中的奇偶校验位与预期奇偶效验值不一致时, 同时设置 PF 和 RDRF。要清除 PF, 读 SCIxS1, 然后读 SCI 数据寄存器 (SCIxD)。 0 没有奇偶效验错误 1 奇偶效验错误

### 14.2.5 SCI 状态寄存器 2 (SCIxS2)

该寄存器有一个只读状态标记。

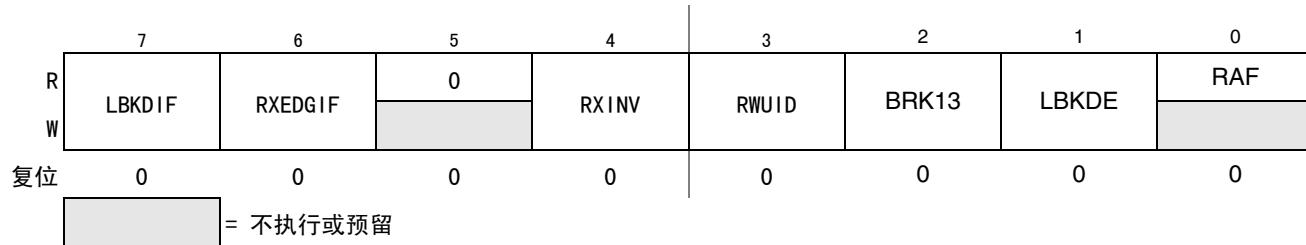


图 14-9. SCI 状态寄存器 2 (SCIxS2)

表 14-7. SCIxS2 字段描述

字段	描述
7 LBKDIF	中止符检测中断标记 — 当使能了 LIN 中止符检测电路且检测到 LIN 中止字符时，就设置 LBKDIF。将 “1” 写入其中可以清除 LBKDIF。 0 未检测到 LIN 中止字符。 1 检测到 LIN 中止字符。
6 RXEDGIF	RxD 管脚活动边沿中断标记 — 当 RxD 管脚上出现活动边沿（如果 RXINV = 0，下降；如果 RXINV = 1，上升）时，就设置 RXEDGIF。将 “1” 写入其中清除 RXEDGIF。 0 接收管脚上未出现活动边沿。 1 接收管脚上出现活动边沿。
4 RXINV <sup>1</sup>	接收数据颠倒 — 设置该位反转已接收数据输入的极性。 0 接收数据未被反转 1 接收数据被反转
3 RWUID	接收唤醒闲置检测 — RWUID 控制着唤醒接收器的闲置字符是否设置 IDLE 位。 0 在接收待机状态 (RWU = 1) 期间，检测到闲置字符时不设置 IDLE 位。 1 在接收待机状态 (RWU = 1) 期间，检测到闲置字符时设置 IDLE 位。
2 BRK13	中止字符生成长度 — BRK13 用于选择较长的发送中止字符长度。成帧错误的检测不受该位状态的影响。 0 中止字符用 10 位时间（如果 M = 1，则是 11 位时间）长度发送 1 中止字符用 13 位时间（如果 M = 1，则是 14 位时间）长度发送
1 LBKDE	LIN 中止字符检测使能 — LBKDE 用来选择较长中止字符检测长度。当设置了 LBKDE 时，防止设置成帧错误 (FE) 和接收数据寄存器已满 (RDRF) 标记。 0 中止字符在 10 位时间（如果 M = 1，则是 11 位时间）长度上检测。 1 中止字符在 11 位时间（如果 M = 1，则是 12 位时间）长度上检测。
0 RAF	接收器活动标记 — 当 SCI 接收器检测到有效起始位开始时，设置 RAF，并且当接收器检测到闲置线路时，RAF 被自动清除。这种状态标记可以用来检查在引导 MCU 进入停止模式前，是否正在接收 SCI 字符。 0 SCI 接收器闲置，正在等待起始位。 1 SCI 接收器活动 (RxD 输入不闲置)。

<sup>1</sup> 设置 RXINV 会反转所有情况下的 RxD 输入：数据位、起始位和停止位、中止字符和闲置。

当在 LIN 系统中使用内部振荡器时，需要把中止符检测阈值提高 1 个位时间。在 LIN 中所允许的最坏计时情况下，0x00 数据字符在运行速度比主器件快 14% 的辅器件上的长度可能达到 10.26 位时间。这将触发旨在检测 10 位中断符号的常规中止符检测电路。当设置了 LBKDE 位时，成帧错误被禁止，中止符检测阈值从 10 位变为 11 位，从而防止把 0x00 数据字符错误检测为一个 LIN 的中止符。

## 14.2.6 SCI 控制寄存器 3 (SClxC3)

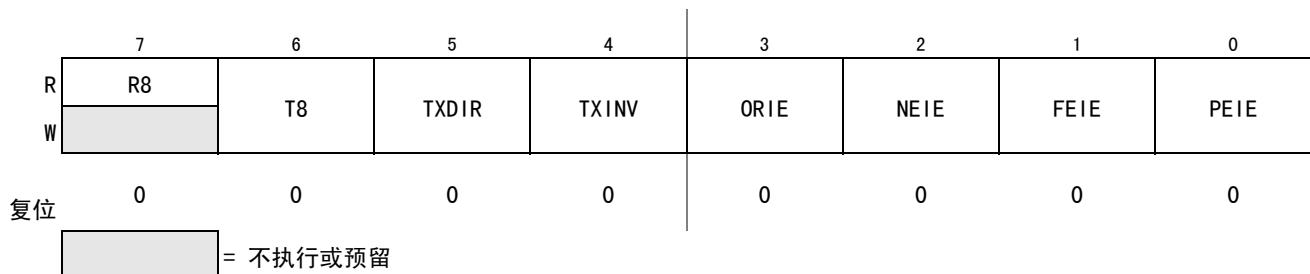


图 14-10. SCI 控制寄存器 3 (SClxC3)

表 14-8. SClxC3 字段描述

字段	描述
7 R8	接收器的第 9 个数据位 — 当 SCI 配置用于 9 位数据 ( $M = 1$ ) 时, R8 可以视为 SClxD 寄存器中缓冲数据的 MSB 左侧的第 9 个接收数据位。读 9 位数据时, 读 SClxD 前读取 R8, 因为读 SClxD 能够完成自动的标记清除顺序, 允许 R8 和 SClxD 被新数据覆盖。
6 T8	9 个数据位发射器 — 当 SCI 配置用于 9 位数据 ( $M = 1$ ) 时, T8 可以视为 SClxD 寄存器中缓冲数据的 MSB 左侧的第 9 个接收数据位。写 9 位数据时, 整个 9 位值在 SClxD 写入后被传输到 SCI 移位寄存器, 因此, T8 应在 SClxD 写入前写入 (如果它需要从它的原来值中修改)。如果 T8 不需要在新值 (例如当它用于生成标记或空间奇偶效验) 中修改, 它就不需要在每次写 SClxD 时写入。
5 TXDIR	单线模式中的 TxD 管脚方向 — 当 SCI 配置用于单线半双工运行 ( $LOOPS = RSRC = 1$ ) 时, 该位决定 TxD 管脚上数据的方向。 0 TxD 管脚是单线模式中的输入。 1 TxD 管脚是单线模式中的输出。
4 TXINV <sup>1</sup>	发送数据反转 — 设置该位反转已发送数据输出的极性。 0 发送数据未被反转 1 发送数据被反转
3 ORIE	溢出中断使能 — 该位使能溢出标记 (OR) 以生成硬件中断请求。 0 OR 中断禁止 (使用轮询) 1 当 OR = 1 时允许硬件中断
2 NEIE	噪音错误中断使能 — 该位使能噪音标记 (NF) 以生成硬件中断请求。 0 NF 中断禁止 (使用轮询) 1 当 NF = 1 时允许硬件中断
1 FEIE	成帧错误中断使能 — 该位使能成帧错误标记 (FE) 以生成硬件中断请求。 0 FE 中断禁止 (使用轮询) 1 当 FE = 1 时允许硬件中断
0 PEIE	奇偶效验错误中断使能 — 该位使能奇偶错误标记 (PF) 以生成硬件中断请求。 0 PF 中断禁止 (使用轮询) 1 当 PF = 1 时允许硬件中断

<sup>1</sup> 设置 TXINV 会反转所有情况下的 TxD 输出: 数据位、起始位和停止位、中止符、闲置。

### 14.2.7 SCI 数据寄存器 (SCIxD)

该寄存器实际上是两个独立寄存器。读返回只读接收数据缓冲器的内容，写进入只写发送数据缓冲器。该寄存器的读写还涉及 SCI 状态标记的自动标记清除机制。

	7	6	5	4	3	2	1	0
R	R7	R6	R5	R4	R3	R2	R1	R0
W	T7	T6	T5	T4	T3	T2	T1	T0
复位	0	0	0	0	0	0	0	0

图 14-11. SCI 数据寄存器 (SCIxD)

## 14.3 功能描述

SCI 允许在 MCU 和远程器件（包括其他 MCU）间进行全双工、异步、NRZ 串行通信。SCI 由波特速率发生器、发射器和接收时钟组成。发射器和接收器独立运行，尽管它们使用同一波特率发生器。在正常运行期间，MCU 监控 SCI 的状态，写将要发送的数据，处理已接收的数据。下面就简要地介绍一下 SCI 的每个块。

### 14.3.1 波特率生成

如图 14-12 所示，SCI 波特率发生器的时钟源是总线速率时钟。

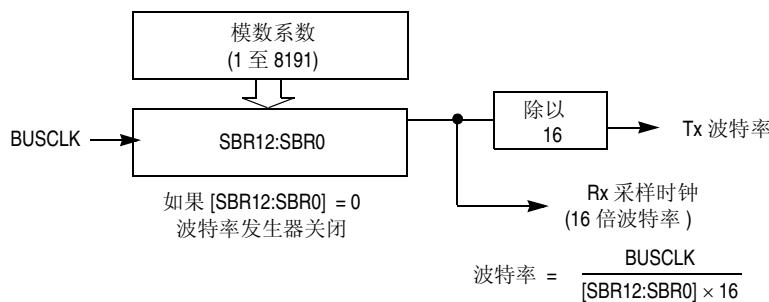


图 14-12. SCI 波特率生成

SCI 通信要求发射器和接收器（一般情况下从独立时钟源中获取波特率）使用相同的波特率。该波特频率的允许容限取决于接收器与起始位的前沿以及位采样执行的具体情况。

每次从高态转换到低态时，MCU 都重新同步位边界，但在最坏的情况下，整个 10 或 11 位时间字符帧中没有这种转换，所以波特率中的任何不匹配都累积到整个字符时间。对于总线频率由晶振驱动的飞思卡尔半导体 SCI 系统，允许的波特率不匹配对 8 位数据格式来说大约为 4.5%、对 9 位数据格式来说大约 4%。尽管波特率模数除数设置不会永远生成与标准速率严格匹配的波特率，但一般情况下都在一个很小的百分比内，是可靠通信可以接受的。

### 14.3.2 发射器功能描述

本小节描述 SCI 发射器的整体结构图，以及发送中断和闲置字符的一些专用功能。发射器结构图如图 14-2。

发射器输出 (TxD) 闲置状态默认为逻辑高态（复位后  $\text{TXINV} = 0$ ）。如果  $\text{TXINV} = 1$ ，发射器输出就被颠倒。通过在  $\text{SClxC2}$  中设置  $\text{TE}$  位，发射器被使能。这会排队前导信号字符，前导信号字符是闲置状态的一个完整字符帧。发射器然后保持闲置状态，直到发送数据缓冲器中出现数据。通过把数据写入 SCI 数据寄存器 ( $\text{SClxD}$ )，程序把数据保存到发送数据缓冲器。

SCI 发射器的中心元件是长度为 10 或 11 位（取决于  $\text{M}$  控制位中的设置）的发送移位寄存器。对于本小节的剩余部分，我们假设  $\text{M} = 0$ ，选择正常的 8 位数据模式。在 8 位数据模式中，移位寄存器中有 1 个起始位、8 个数据位和 1 个停止位。当发送移位寄存器可以用于新 SCI 字符时，在发送数据寄存器中等待的值被传输到移位寄存器（与波特率时钟同步），同时设置发送数据寄存器空 ( $\text{TDRE}$ ) 状态标记，显示另外一个字符可以写入  $\text{SClxD}$  的发送数据缓冲器。

如果停止位移出 TxD 管脚后发送数据缓冲器中没有新字符在等待，发射器设置发送完成标记，进入闲置模式，TxD 处于高态，等待发送更多字符。

将 0 写入  $\text{TE}$  不会立即释放管脚使其成为通用 I/O 管脚。正在进行的任何发送活动必须首先完成，这包括正在发送的数据字符、已进入队列的闲置字符和已进入队列的中止字符。

#### 14.3.2.1 发送中断和排队闲置

$\text{SClxC2}$  中的  $\text{SBK}$  控制位用来发送中止字符，中止字符最初用来引起旧式电传打字接收器的注意。中止字符是逻辑 0（10 位时间，包括启动和停止位）的全字符时间。13 位时间的较长中止符可以通过设置  $\text{BRK13} = 1$  进行使能。一般来说，程序要等待  $\text{TDRE}$  进行设置，以显示信息的最后一个字符已经移动到发送移位器，然后依次把 1 和 0 写入  $\text{SBK}$  位。一旦移位器可用，该操作就立即对将发送的中止字符进行排队。如果当已进入队列的中止符进入移位器（与波特率时钟同步）时  $\text{SBK}$  仍然为 1，额外的中止字符会进入队列。如果接收器件是另一个飞思卡尔半导体 SCI，中止字符将作为所有 8 个数据位中的 0 进行接收，并出现成帧错误 ( $\text{FE} = 1$ )。

当使用闲置线路唤醒时，信息之间就需要闲置（逻辑 1）的全字符时间，以唤醒正处于睡眠状态的任何接收器。在正常情况下，程序会等待  $\text{TDRE}$  进行设置，显示信息的最后字符已经移动到发送移位器，然后依次把 0 和 1 写入  $\text{TE}$  位。一旦移位器可用，该操作就立即对将发送的闲置字符进行排队。只要  $\text{TE} = 0$  时移位器中的字符没有完成，SCI 发射器永远不会真正放弃 TxD 管脚的控制。如果移位器在  $\text{TE} = 0$  时有完成的可能，则设置通用 I/O 控制，这样与 TxD 共享的管脚就是驱动逻辑 1 的输出。这确保了 TxD 线路看起来像是正常闲置线路，即便在向  $\text{TE}$  写入 0 和 1 的过程中 SCI 失去对端口管脚的控制。

中止字符的长度受  $\text{BRK13}$  和  $\text{M}$  位的影响，如下表所示。

表 14-9. 中止字符长度

$\text{BRK13}$	$\text{M}$	中止字符长度
0	0	10 位时间
0	1	11 位时间
1	0	13 位时间
1	1	14 位时间

### 14.3.3 接收器功能描述

在本小节，首先把接收器结构图（图 14-3）作为接收器总体功能描述的指南使用。然后详细描述了用来重建接收器数据的数据采样方法。最后解释了接收器唤醒功能的两个变种。

通过设置 **RXINV = 1**，接收器输入被反转。通过设置 **SCIxC2** 中的 **RE** 位，接收器被使能。字符帧由逻辑 0 的起始位、8 个（或 9 个）数据位（**LSB 先发**）和逻辑 1 的停止位组成。如需了解 9 位数据模式的有关信息，参见 [14.3.5.1，“8 位和 9 位数据模式”](#)。对于本小节的其余部分，我们假设 **SCI** 配置用于正常的 8 位数据模式。

在把停止位接收到接收移位器后，如果接收数据寄存器还未满，数据字符就被传输到接收数据寄存器，设置接收数据寄存器已满（**RDRF**）状态标记。如果已经设置了显示接收数据寄存器（缓冲器）已满的 **RDRF**，就设置溢出（**OR**）状态标记，新数据丢失。因为 **SCI** 接收器是双缓冲的，程序在设置 **RDRF** 后、读取接收数据缓冲器的数据前，有一个全字符时间，以避免接收器溢出。

当程序检测到接收数据寄存器已满（**RDRF = 1**）时，它通过读 **SCIxD** 从接收数据寄存器中获取数据。**RDRF** 标记由一个 2 步式顺序自动清除，这个 2 步式顺序通常在处理接收数据的用户程序中满足。如需了解标记清除的更多详细信息，参见 [14.3.4，“中断和状态标记”](#)。

#### 14.3.3.1 数据采样方法

**SCI** 接收器使用 16 倍波特率时钟进行采样。接收器通过以 16 倍波特率提取逻辑电平样本，以搜索 **RxD** 串行数据输入管脚上的下降边沿。下降边沿的定义是 3 个连续逻辑 1 采样后的逻辑 0 样本。16 倍波特率时钟用来把位时间划分为 16 个段，分别标记为 **RT1** 到 **RT16**。当定位了下降边沿时，还要从 **RT3**、**RT5** 和 **RT7** 中提取三个样本，以确保这是真正的起始位，而不仅仅噪音。如果这三个样本至少有两个样本为 0，接收器假设它与接收器字符同步。

接收器然后在 **RT8**、**RT9** 和 **RT10** 的每个位时间上进行采样，包括起始启位和停止位，以决定该位的逻辑电平。逻辑电平是位时间期间提取的绝大多数样本的逻辑电平。在起始位中，如果 **RT3**、**RT5** 和 **RT7** 上的样本中至少有 2 个样本为 0，那么就假设该位为 0，即便在 **RT8**、**RT9** 和 **RT10** 上提取的一个或所有样本均为 1。如果字符帧的任意位时间（包括起始和停止位）中的任意样本不能与该位的逻辑电平保持一致，当收到的字符传输到接收数据缓冲器时，都设置噪音标记（**NF**）。

下降边沿检测逻辑不断寻找下降边沿，如果检测到边沿，样本时钟重新同步位时间。这样当出现噪音或不匹配波特率时，就可以提高接收器的可靠性。它不能改进最坏情况分析，因为有些字符在字符帧的任何地方都没有额外的下降边沿。

在成帧错误情况下，假设收到的字符不是中止字符，搜索下降边沿的采样逻辑就充满 3 个逻辑 1 样本，这样一个新起始位几乎可以立即检测到。

在成帧错误情况下，接收器禁止接收任何新字符，直到成帧错误标记被清除。如果仍设置 **FE**，接收移位寄存器继续发挥作用，但整个字符不能传输到接收数据缓冲器。

### 14.3.3.2 接收器唤醒操作

接收器唤醒是一种硬件机制，允许 SCI 接收器忽略用于不同 SCI 接收器的信息中的字符。在这种系统中，所有接收器都估计每条信息的第一个字符，一旦确定该信息旨在用于不同接收器，它们就立即将逻辑 1 写入 SCIx C2 中的接收器唤醒 (RWU) 控制位。当设置了 RWU 位时，禁止设置与接收器有关的状态标记（当设置了 RWUID 位时，闲置位 IDLE 除外），因此消除了处理不重要信息字符的软件开销。在信息结束或在下一条信息开始时，所有接收器自动强制 RWU 清零，这样所有接收器及时唤醒，以查看下一条信息的首字符。

#### 14.3.3.2.1 闲置线路唤醒

当 **WAKE = 0** 时，接收器配置用于闲置线路唤醒。在该模式中，当接收器检测到闲置线路级的某个全字符时间时，RWU 被自动清除。**M** 控制位选择 8 位或 9 位数据模式，确定构成全字符时间所需的闲置位时间（10 或 11 位时间，由于起始和停止位）。

当 **RWU = 1**、**RWUID = 0** 时，唤醒接收器的闲置条件不会设置 **IDLE** 标记。接收器唤醒并等待下一条信息的第一个数据字符，这将设置 **RDRF** 标记并生成中断（如使能的话）。当 **RWUID = 1** 时，任何闲置条件都设置闲置标记并生成中断（如使能的话），无论 **RWU** 是 0 还是 1。

闲置线路类型 (**ILT**) 控制位选择以下两种方式中的一种来检测闲置线路。当 **ILT = 0** 时，闲置位计数器在起始位后启动，这样停止位和字符末端的任何逻辑 1 计数闲置的全字符时间。当 **ILT = 1** 时，闲置位计数器直到停止位时间结束后才启动，这样闲置检测不受上一条信息的最后一个字符中的数据的影响。

#### 14.3.3.2.2 地址标记唤醒

当 **WAKE = 1** 时，接收器配置用于地址标记唤醒。在该模式中，当接收器检测到已接收字符的最高位（在 **M = 0** 模式中是第 8 个位；在 **M = 1** 模式中是第 9 个位）中的逻辑 1 时，RWU 被自动清除。

地址标记唤醒允许信息包含闲置字符，但要求预留 **MSB**，以便在地址帧中使用。在收到停止位前，地址帧中 **MSB** 的逻辑 1 会清除 **RWU** 位，并设置 **RDRF** 标记。在这种情况下，会收到设置了 **MSB** 的字符，即便接收器在该字符时间的大部分时间中处于睡眠状态。

### 14.3.4 中断和状态标记

SCI 系统有三种独立的中断向量，以减少隔离中断原因所需的软件数量。一个中断向量与 **TDRE** 和 **TC** 事件的发射器相关，一个中断向量与 **RDRF**、**IDLE**、**RXEDGIF** 和 **LBKDIF** 事件的接收器相关，第三个向量用于 **OR**、**NF**、**FE** 和 **PF** 错误情况。这 10 个中断源的每个都可以由本地中断使能分别进行屏蔽。当清除本地使能以禁止生成硬件中断请求时，标记仍然可以用软件进行轮询。

SCI 发射器有两种状态标记，它们都可以生成硬件中断请求。发送数据寄存器空 (**TDRE**) 显示发送数据缓冲器何时有空间将其他发送字符写入 **SCIx D**。如果设置了发送中断使能 (**TIE**) 位，每当 **TDRE = 1** 时都请求硬件中断。发送完成 (**TC**) 表示发射器完成发送所有数据、前导信号和中止字符，且它处于闲置状态，**TxD** 不活动。该标记通常用于带有调制解调器的系统，以决定何时可以安全关闭调制解调器。如果设置了发送完成中断使能 (**TCIE**) 位，每当 **TC = 1** 时请求硬件中断。

如果相应 **TIE** 或 **TCIE** 本地中断允许位为 0，那么就使用软件轮询来监控 **TDRE** 和 **TC** 状态标记，而不是发生软件中断。

当程序检测到接收数据寄存器已满 (**RDRF = 1**) 时，它通过读 **SClxS1** 从接收数据寄存器获取数据。 **RDRF = 1** 时读 **SClxS1**，这样 **RDRF** 标记就被清除，然后再读 **SClxD**。

当使用轮询时，该顺序自然在用户程序的正常过程中得到满足。如果使用硬件中断，就必须在中断服务程序 (**ISR**) 中读 **SClxS1**。在正常情况下，这无论如何都要在 **ISR** 中完成，以检查接收错误，这样该顺序就自动满足了。

当 **RxD** 线路在很长一段时间内保持闲置时，**IDLE** 状态标记包括可以防止其进行重复设置的逻辑。当 **IDLE = 1** 时，读 **SClxS1** 可以清除 **IDLE**，然后再读 **SClxD**。在已经清除 **IDLE** 后，它不能再次进行设置，直到接收器已经收到至少一个新字符并已设置 **RDRF**。

如果在造成设置 **RDRF** 的已接收字符中检测到有关错误，就在设置 **RDRF** 的同时设置错误标记，如噪音标记 (**NF**)、成帧错误 (**FE**) 和奇偶效验错误标记 (**PF**)。这些标记不会在溢出情况下设置。

如果当一个新字符准备好从接收移位器传输到接收数据缓冲器时已经设置了 **RDRF**，就设置溢出 (**OR**) 标记，而数据及任何有关的 **NF**、**FE** 或 **PF** 条件丢失。

任何时候，**RxD** 串行数据输入管脚上的活动边沿都会引起 **RXEDGIF** 标记设置。把 1 写入 **RXEDGIF** 会清除该标记。该功能取决于正被使能 (**RE = 1**) 的接收器。

### 14.3.5 其他 SCI 功能

以下几节描述其他 SCI 功能。

#### 14.3.5.1 8 位和 9 位数据模式

通过在 **SClxC1** 中设置 **M** 控制位，SCI 系统（发射器和接收器）在经过配置后可以运行在 9 位数据模式中。在 9 位模式中，在 SCI 数据寄存器的 **MSB** 的左侧有一个第 9 数据位。对发送数据缓冲器来说，该位保存在 **SClxC3** 中的 **T8**。对接收器来说，第 9 位保存在 **SClxC3** 中的 **R8**。

为了连贯写入发送数据寄存器，写入 **SClxD** 前请先写入 **T8** 位。

如果作为新字符第 9 位发送的位值和上一字符的位值相同，则不需要重新写入 **T8**。当数据从发送数据缓冲器传输到发送移位器时，在数据从 **SClxD** 传输到移位器的同时 **T8** 中的值被复制。

9 位数据模式通常和奇偶效验一起使用，以允许数据的 8 个位加第 9 位中的奇偶效验位。或者与地址标记唤醒一起使用，这样第 9 数据位可以作为唤醒位。在自定义协议中，第 9 位还可以作为软件控制标记。

### 14.3.5.2 停止模式运行

在所有停止模式中，SCI 模块的时钟都被暂停。

在 STOP1 和 STOP2 模式中，所有 SCI 寄存器数据丢失，当从这两种停止模式恢复时必须重新初始化。任何 SCI 模块寄存器在 STOP3 模式中都不受影响。

接收输入活动边沿检测电路在 STOP3 模式中仍然是活动的，但在 STOP2 模式中不活动。如果中断未屏蔽 ( $RXEDGIE = 1$ )，接收输入上的活动边沿将把 CPU 带离 STOP3 模式。

注意，由于时钟被暂停，当从停止模式（仅在 STOP3 模式）退出时，SCI 模块会重新开始运行。当 ISC 模块正在发送或接收字符时，软件应确保不会进入停止模式。

### 14.3.5.3 循环模式

当  $LOOPS = 1$  时，相同寄存器中的 RSRC 位选择循环模式 ( $RSRC = 0$ ) 或单线模式 ( $RSRC = 1$ )。循环模式独立于外部系统连接，有时用于检查软件，以帮助隔离系统问题。在该模式中，发射器输出内部连接到接收器输入，且 SCI 不使用 RxD 管脚，因此它恢复为通用端口 I/O 管脚。

### 14.3.5.4 单线运行

当  $LOOPS = 1$  时，相同寄存器中的 RSRC 位选择循环模式 ( $RSRC = 0$ ) 或单线模式 ( $RSRC = 1$ )。单线模式用来执行半双工串行连接。接收器内部连接到发射器输出和 TxD 管脚。RxD 管脚不使用并恢复为通用端口 I/O 管脚。

在单线模式中， $SCIxC3$  中的 TXDIR 位控制着 TxD 管脚上的串行数据方向。当  $TXDIR = 0$  时，TxD 管脚是 SCI 接收器的输入，发射器与 TxD 管脚的连接被暂时断开，因此外部器件就可以向接收器发送串行数据。当  $TXDIR = 1$  时，TxD 管脚是一个由发射器驱动的输出。在单线模式中，辅发射器到接收器的内部环回连接使接收器接收由发射器发送出来的字符。.

# 第 15 章

## 实时计数器 (S08RTCV1)

### 15.1 简介

RTC 模块包括一个 8 位计数器、一个 8 位比较器、几个二进制和十进制预分频器、三个时钟源和一个可编程定期中断。该模块可用于时刻、日历或任何任务调度功能。此外，它可以从低功率模式中提供周期性叫醒服务而不需要外部组件。

MC9S08DZ60 系列的所有器件都带有 RTC。

#### 15.1.1 RTC 时钟信号名称

本章中提及的 ERCLK 和 IRCLK 分别对应信号 MCGERCLK 和 MCGIRCLK。

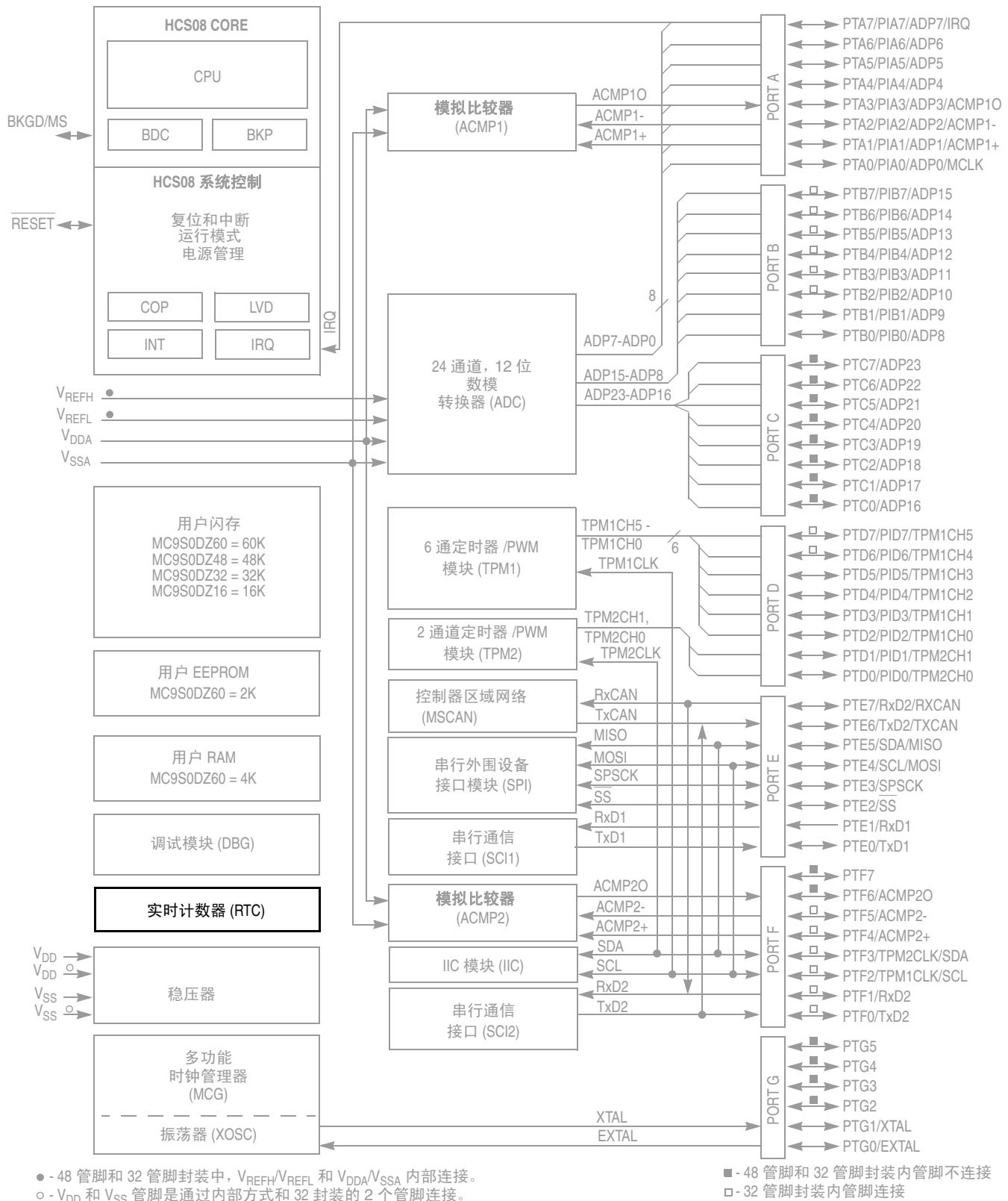


图 15-1. MC9S08DZ60 结构图

## 15.1.2 功能

RTC 模块的功能包括：

- 8 位向上计数器
  - 8 位模数匹配限制
  - 匹配时软件可控制的定期中断
- 三个软件可选时钟源，向预分频器输入可选的二进制和十进制分频器值
  - 1-kHz 的内部低功率振荡器（LPO）
  - 外部时钟（ERCLK）
  - 32-kHz 内部时钟（IRCLK）

## 15.1.3 运行模式

本小节定义停止、等待和后台调试模式的操作。

### 等待模式

如果在执行等待（WAIT）指令前被启动，RTC 继续以等待模式运行。因此，如果启动了实时中断，RTC 可用于使 MCU 脱离等待模式。为了最大限度地降低耗电，如果在等待模式下不需要作为中断源，RTC 应通过软件关闭。

### 停止模式

如果在执行停止指令前 RTC 被启动，它会继续以停止 2 或停止 3 模式运行。因此，如果启动了实时中断，RTC 可用于使 MCU 脱离停止模式而无需外部组件。

LPO 时钟可用于停止 2 和停止 3 模式。**ERCLK 和 IRCLK 时钟只能用于停止 3 模式。**

所有时钟源被关闭时，功耗较低；但在这种情况下，实时中断不能从停止模式中唤醒 MCU。

### 激活后台模式

RTC 会在激活后台模式期间暂停所有计数，直到微控制器返回到正常的用户运行模式。只要 RTMOD 寄存器没有写入，并且 RTCPMS 和 RTCLKS 位没有改变，计数就会从暂停值恢复。

### 15.1.4 结构图

RTC 模块的结构图如图 15-2 所示。

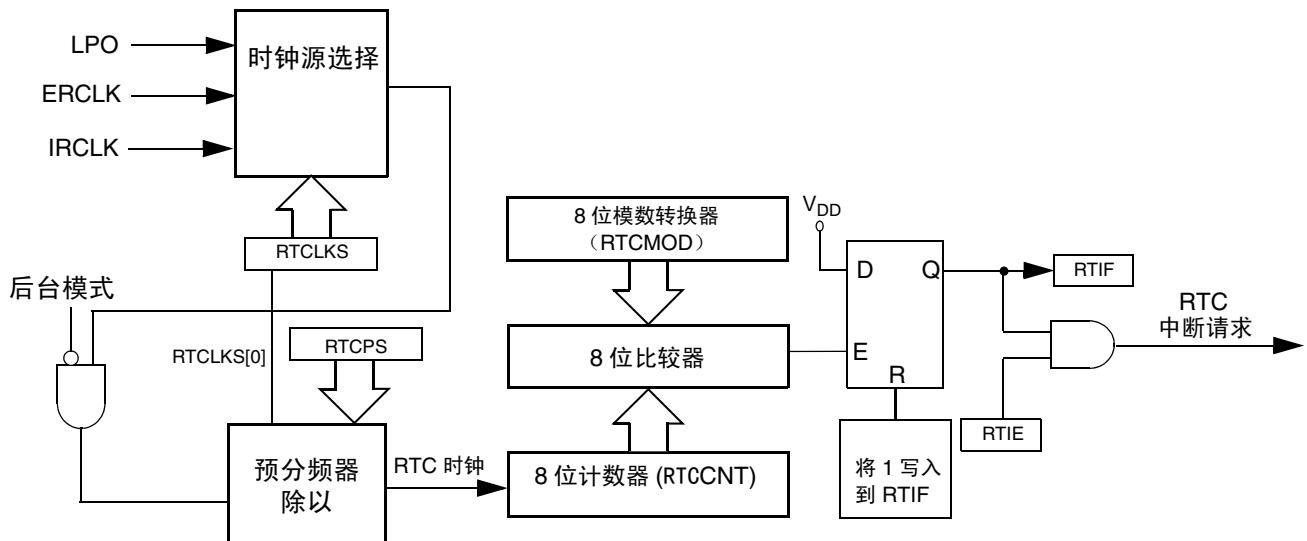


图 15-2. 实时计数器 (RTC) 结构图

### 15.2 外部信号描述

RTC 不包括任何片外信号。

### 15.3 寄存器定义

RTC 包括一个状态和控制寄存器、一个 8 位计数器寄存器和一个 8 位模数寄存器。

请参考本产品介绍内存部分的“直接页面寄存器总结”小节，了解所有 RTC 寄存器的绝对地址分配。

表 15-1 为 RTC 寄存器的总结。

表 15-1. RTC 寄存器总结

Name		7	6	5	4	3	2	1	0							
RTCSC	R	RTIF	RTCLKS	RTIE	RTCPSS											
	W															
RTCCNT	R	RTCCNT														
	W															
RTCMOD	R	RTCMOD														
	W															

### 15.3.1 RTC 状态和控制寄存器 (RTCSC)

RTCSC 包含实时中断状态标记 (RTIF)、时钟选择位 (RTCLKS)、实时中断启动位 (RTIE) 和预分频器选择位 (RTCPS)。

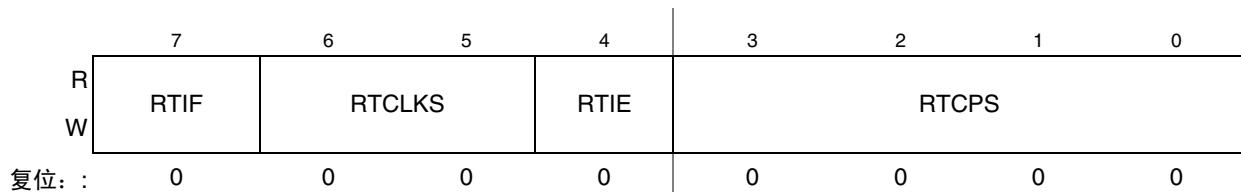


图 15-3. RTC 状态和控制寄存器 (RTCSC)

表 15-2. RTCSC 字段描述

字段	描述
7 RTIF	实时中断标记 — 本状态位指示 RTC 计数器寄存器计数值达到 RTC 模数寄存器中设定的值。写入逻辑数 0 是无效的。写入逻辑数 1 可清除此位和实时中断请求。复位将 RTIF 清除为 0。 0 RTC 计数器未达到 RTC 模数寄存器中规定的值。 1 RTC 计数器已达到 RTC 模数寄存器中规定的值。
6–5 RTCLKS	实时时钟源选择 — 这两个读 / 写位为 RTC 预分频器选择时钟源输入。更换时钟源可清除预分频器和 RTCCNT 计数器。选择时钟源时，确保时钟源被正常使能（如果适用），以确保 RTC 的正确运行。复位将把 RTCLKS 清除为 00。 00 实时时钟源为 1-kHz 低功率振荡器 (LPO) 01 实时时钟源为外部时钟 (ERCLK) 1x 实时时钟源为内部时钟 (IRCLK)
4 RTIE	实时中断使能 — 这个读 / 写位使能实时中断。如果 RTIE 被设置，那么 RTIF 被设置时会生成中断。复位将把 RTIE 清除为 0。 0 实时中断请求被关闭。采用软件轮询。 1 实时中断请求被允许。
3–0 RTCPS	实时时钟预分频器选择 — 这四个读 / 写位为时钟源选择二进制或十进制除数值。请参见 表 15-3. 修改预分频器可清除预分频器和 RTCCNT 计数器。复位将把 RTCPS 清除为 0000。

表 15-3. RTC 预分频器除数值

RTCLKS[0]	RTCPS															
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	Off	$2^3$	$2^5$	$2^6$	$2^7$	$2^8$	$2^9$	$2^{10}$	1	2	$2^2$	10	$2^4$	$10^2$	$5 \times 10^2$	$10^3$
1	Off	$2^{10}$	$2^{11}$	$2^{12}$	$2^{13}$	$2^{14}$	$2^{15}$	$2^{16}$	$10^3$	$2 \times 10^3$	$5 \times 10^3$	$10^4$	$2 \times 10^4$	$5 \times 10^4$	$10^5$	$2 \times 10^5$

### 15.3.2 RTC 计数器寄存器 (RTCCNT)

RTCCNT 是 8 位计数器的当前 RTC 计数的只读值。

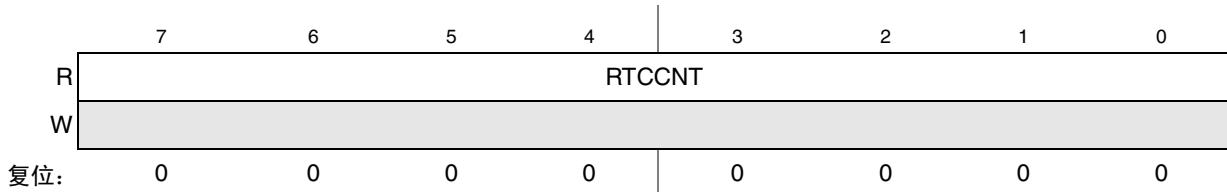


图 15-4. RTC 计数器寄存器 (RTCCNT)

表 15-4. RTCCNT 字段描述

字段	描述
7:0 RTCCNT	RTC 计数—这 8 个只读位包含 8 位计数器的当前值。写入操作对该寄存器无效。复位、写入 RTCMOD 或向 RTCLKS 和 RTCPS 写入不同值将把计数清除为 0x00。

### 15.3.3 RTC 模数寄存器 (RTCMOD)

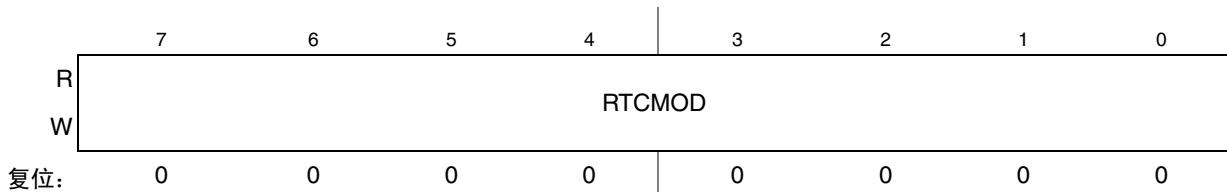


图 15-5. RTC 模数寄存器 (RTCMOD)

表 15-5. RTCMOD 字段描述

字段	描述
7:0 RTCMOD	RTC 模数—这 8 个读 / 写位包含用于在比较匹配后将计数复位为 0x00 并设置 RTIF 状态位的模数值。0x00 值会在预分频器输出的每个上升沿设置 RTIF 位。写入 RTCMOD 将把预分频器和 RTCCNT 计数器复位为 0x00。复位将模数寄存器设置为 0x00。

## 15.4 功能描述

RTC 由一个带有 8 位模数寄存器的主 8 位向上计数器、一个时钟源选择器和一个带有二进制和十进制可选值的预分频器组件组成。该模块还包含软件可选的中断逻辑。

任何 MCU 复位后，计数器被停止并复位为 0x00；模数寄存器被设置为 0x00，而且预分频器将关闭。1-kHz 内部振荡器时钟被选择为默认时钟源。要启动预分频器，向预分频器选择位 (RTCPS) 写入零以外的任何值。

三种时钟源可由软件选择：低功率振荡器时钟（LPO）、外部时钟（ERCLK）和内部时钟（IRCLK）。RTC 时钟选择位（RTCLKS）用于选择想要的时钟源。如果一个不同值被写入到 RTCLKS 中，预分频器和 RTCCNT 计数器将复位为 0x00。

RTCPs 和 RTCLKS[0] 位选择想要的除数值。如果不同值被写入到 RTCPs 中，预分频器和 RTCCNT 计数器将复位为 0x00。表 15-6 显示了不同的预分频器周期值。

表 15-6. 预分频器周期

RTCPs	1-kHz 内部时钟源 (RTCLKS = 00)	1-MHz 外部时钟源 (RTCLKS = 01)	32-kHz 内部时钟源 (RTCLKS = 10)	32-kHz 外部时钟源 (RTCLKS = 11)
0000	Off	Off	Off	Off
0001	8 ms	1.024 ms	250 μs	32 ms
0010	32 ms	2.048 ms	1 ms	64 ms
0011	64 ms	4.096 ms	2 ms	128 ms
0100	128 ms	8.192 ms	4 ms	256 ms
0101	256 ms	16.4 ms	8 ms	512 ms
0110	512 ms	32.8 ms	16 ms	1.024 s
0111	1.024 s	65.5 ms	32 ms	2.048 s
1000	1 ms	1 ms	31.25 μs	31.25 ms
1001	2 ms	2 ms	62.5 μs	62.5 ms
1010	4 ms	5 ms	125 μs	156.25 ms
1011	10 ms	10 ms	312.5 μs	312.5 ms
1100	16 ms	20 ms	0.5 ms	0.625 s
1101	0.1 s	50 ms	3.125 ms	1.5625 s
1110	0.5 s	0.1 s	15.625 ms	3.125 s
1111	1 s	0.2 s	31.25 ms	6.25 s

RTC 模数寄存器（RTCMOD）允许将比较值设置为从 0x00 到 0xFF 的任何值。当计数器处于有效状态时，计数器以所选速率递增，直到计数与模数值匹配。当这些值匹配时，计数器复位为 0x00 并继续计数。任何时候发生匹配时，实时中断标记（RTIF）会被设置。该标记在从模数值过渡为 0x00 时设置。写入 RTCMOD 将使预分频器和 RTCCNT 计数器复位为 0x00。

RTC 允许在设置 RTIF 时生成中断。要使能实时中断，在 RTCSC 中设置实时中断使能位（RTIE）。向 RTIF 写入 1 可以清除 RTIF。

### 15.4.1 操作实例

这一部分显示了计数器达到模数寄存器中的匹配值时 RTC 的运行情况。

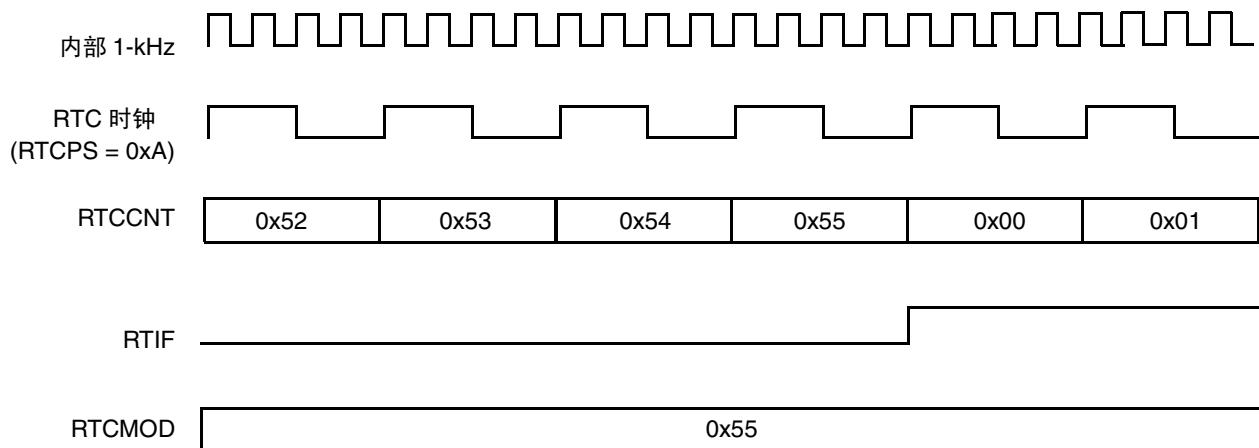


图 15-6. RTC 计数器溢出实例

在图 15-6 所示的例子中，所选时钟源为内部时钟源。预分频器 (RTCPS) 设置为 0xA 或 divide-by-4。RTCMOD 寄存器中的模数值设置为 0x55。当计数器 RTCCNT 达到模数值 0x55 时，计数器溢出为 0x00 并继续计数。. 实时中断标记 RTIF 会在计数器值从 0x55 变为 0x00 时设置。如果如果 RTIE = 1， RTIE 被设置时会生成一个实时中断。

## 15.5 初始化 / 应用信息

本小节提供示例代码，为用户提供如何初始化和配置 RTC 模块方面的一些基本指导。示例软件采用 C 语言实施。

下面的示例介绍了如何通过使用 1-kHz 时钟源的 RTC 实现时间计时，以实现尽可能低的功耗。由于 1-kHz 时钟源不如晶振准确，我们可以添加软件来进行必要的调整。为了在没有以额外的功耗为代价进行调整的情况下实现精确性，可选择具有适当预分频器和模数值的外部时钟 (ERCLK) 或内部时钟 (IRCLK)。

```

/* Initialize the elapsed time counters */
Seconds = 0;
Minutes = 0;
Hours = 0;
Days=0;

/* Configure RTC to interrupt every 1 second from 1-kHz clock source */
RTCMOD.byte = 0x00;
RTCSC.byte = 0x1F;

```

```
*****
Function Name : RTC_ISR
Notes : Interrupt service routine for RTC module.
*****
#pragma TRAP_PROC
void RTC_ISR(void)
{
    /* Clear the interrupt flag */
    RTCSC.byte = RTCSC.byte | 0x80;
    /* RTC interrupts every 1 Second */
    Seconds++;
    /* 60 seconds in a minute */
    if (Seconds > 59){
        Minutes++;
        Seconds = 0;
    }
    /* 60 minutes in an hour */
    if (Minutes > 59){
        Hours++;
        Minutes = 0;
    }
    /* 24 hours in a day */
    if (Hours > 23){
        Days++;
        Hours = 0;
    }
}
```



# 第 16 章

## 定时器脉冲宽度调节器 (S08TPMV3)

### 注意

本章的描述适用于 S08TPM 版本 3，该版本适用于 OM74K 及本器件的更新掩码集。3M05C 和更早版本的掩码集器件使用 S08TPM 版本 2。如果您的器件使用掩码 3M05C 或更早版本，请参考 [364 页上的附录 B，“定时器脉宽调制器 \(TPMV2\)”，以了解该模块的相关信息。](#)

### 16.1 简介

TPM 为每个通道使用一个输入 / 输出 (I/O) 管脚，即 TPM<sub>x</sub>CH<sub>n</sub>，其中 x 为 TPM 编号 (如 1 或 2) 而 n 为通道编号 (如 0–5)。TPM 与通用 I/O 端口管脚共享其 I/O 管脚 (请参考 [Pins and Connections 章节](#) 了解更多信息)。

C9S08DZ60 系列 MCU 有两种 TPM 模块。在所有封装中，TPM2 有 2 个通道。TPM1 中外部管脚上的可用通道数取决于封装：

- 64 管脚和 48 管脚封装中有 6 个通道
- 32 管脚封装中有 4 个通道。

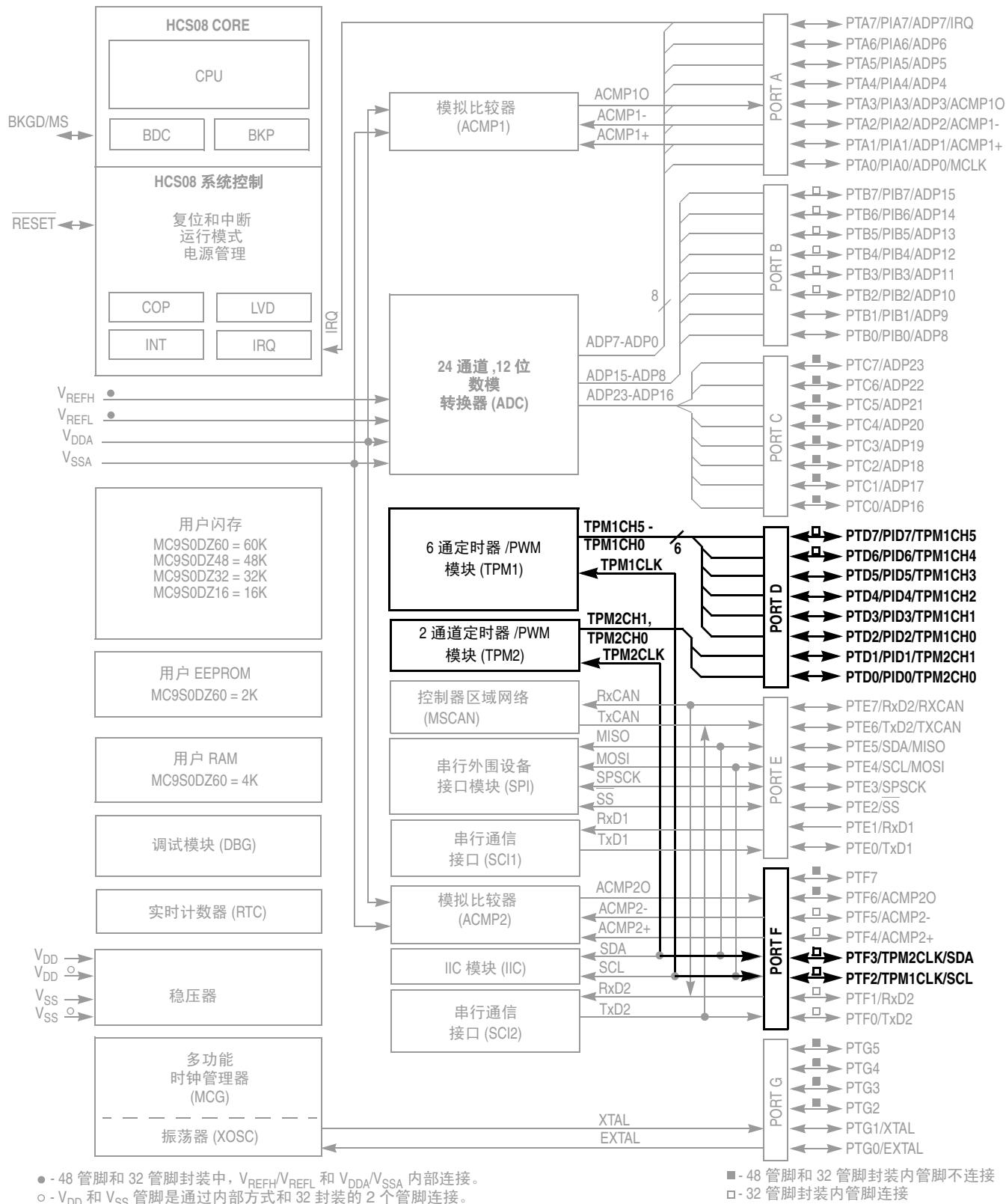


Figure 16-1. MC9S08DZ60 结构图

## 16.1.1 功能

TPM 包括以下独特功能：

- 1 至 8 个通道：
  - 每个通道可以是输入捕捉、输出比较或边缘对齐 PWM 模式
  - 上升边、下降边或任何边输入捕捉触发
  - 设置、清除、切换输出比较操作
  - PWM 输出上的可选极性
- 在所有通道上，模块可配置为缓冲、中央对齐脉冲宽度调制 (cpwm)
- 定时器时钟源，可选为预分频的总线时钟、固定系统时钟或外部时钟管脚
  - 除以 1, 2, 4, 8, 16, 32, 64, 或 128 的预分频器值相位
  - 固定系统时钟源通过片上同步电路与总线时钟保持同步
  - 外部时钟管脚可与任何定时器通道管脚或独立的输入管脚共享。
- 16 位自由运行或模数向上 / 向下计数操作
- 定时器系统启动
- 每个通道一个中断以及终端计数中断

## 16.1.2 运行模式

一般来说，TPM 通道可以独立配置，以便以输入捕捉、输出比较或边缘对齐的 PWM 模式运行。控制位使整个 TPM（所有通道）可以切换为中央对齐的 PWM 模式。选择中央对齐 PWM 模式后，输入捕捉、输出比较和边缘对齐 PWM 功能不能在本 TPM 模式的任何通道上使用。

当微控制器处于激活的 BDM 后台模式或 BDM 前台模式时，TPM 会临时暂停所有计数操作，直到微控制器返回到正常用户运行模式。在停止模式下，包括主振荡器的所有系统时钟都会停止；因此在时钟恢复前，TPM 被有效关闭。在等待模式下，TPM 继续正常运行。如果 TPM 不需要产生实时参考或提供从等待模式中唤醒 MCU 的中断源，那么用户在进入等待模式前可以通过关闭 TPM 功能来节约电源。

- 输入捕捉模式
 

关联 MCU 管脚上发生所选边沿事件时，16 位定时器计数器当前值被捕捉到通道值寄存器中，同时会设置一个中断标志位。上升边沿、下降边沿、任何边沿或无边沿（关闭通道）可选择作为触发输入捕捉的活动边沿。
- 输出比较模式
 

定时计数器寄存器中的值与通道值寄存器相匹配时，会设置一个中断标志位，并且会在管理管脚上强制执行所选的输出操作。输出比较操作可选择用于强制将管脚设置为零或 1、反转管脚电平或忽略管脚（用于软件定时功能）。
- 边缘对齐 pwm 模式
 

16 位模数寄存器值加 1 设置 PWM 输出信号的周期。通道值寄存器设置 PWM 输出信号的占空比。此外，用户还可选择 PWM 输出信号的极性。中断可在周期结束和占空比过渡点上提供。这类 PWM 信号被称为边缘对齐，因为所有 PWM 信号的前沿是在周期开始时对齐的，该周期对 TPM 内的所有通道是相同的。

- 中央对齐 pwm 模式

16 位模数寄存器值的两倍设置 PWM 输出周期，而通道值寄存器设置一半占空比持续时间。定时器计数器向上计数，直到达到模数值，然后向下计数直到达到 0。向下计数的情况下，计数与通道值寄存器匹配时，PWM 输出进入活动状态。向上计数的情况下，计数与通道值寄存器匹配时，PWM 输出进入非活动状态。这类 PWM 信号被称为中央对齐，因为所有通道的活动占空比的中心与计数值 0 对齐。用于小型设备中的发动机类型需要这类 PWM 应用。

只是一个简要介绍。运行模式的详细介绍请参见后面的各小节。

### 16.1.3 结构图

TPM 为每个通道使用一个输入 / 输出 (I/O) 管脚，即 TPMxCH<sub>n</sub> (定时器通道 n)，其中 n 为通道编号 (1-8)。TPM 与通用 I/O 端口管脚分享其 I/O 管脚 (请参考全芯片规范中的输入 / 输出管脚描述，了解如何完成具体芯片执行)。

图 16-2 显示了 TPM 结构。TPM 的中心组件是 16 位计数器。该计数器既可作为自由运行的计数器运行，又可作为模数向上 / 向下计数器运行。TPM 计数器 (以正常的向上计数模式运行时) 为输入捕捉、输出比较和边缘对齐 PWM 功能提供定时参考。定时器计数器模数寄存器 TPMxMODH:TPMxMODL 控制计数器的模数值 (0x0000 或 0xFFFF 值有效地使计数器自由运行)。软件可随时读取计数器值而不影响计数序列。向 TPMxCNT 计数器的任何一半写入任何数据值都会复位计数器。

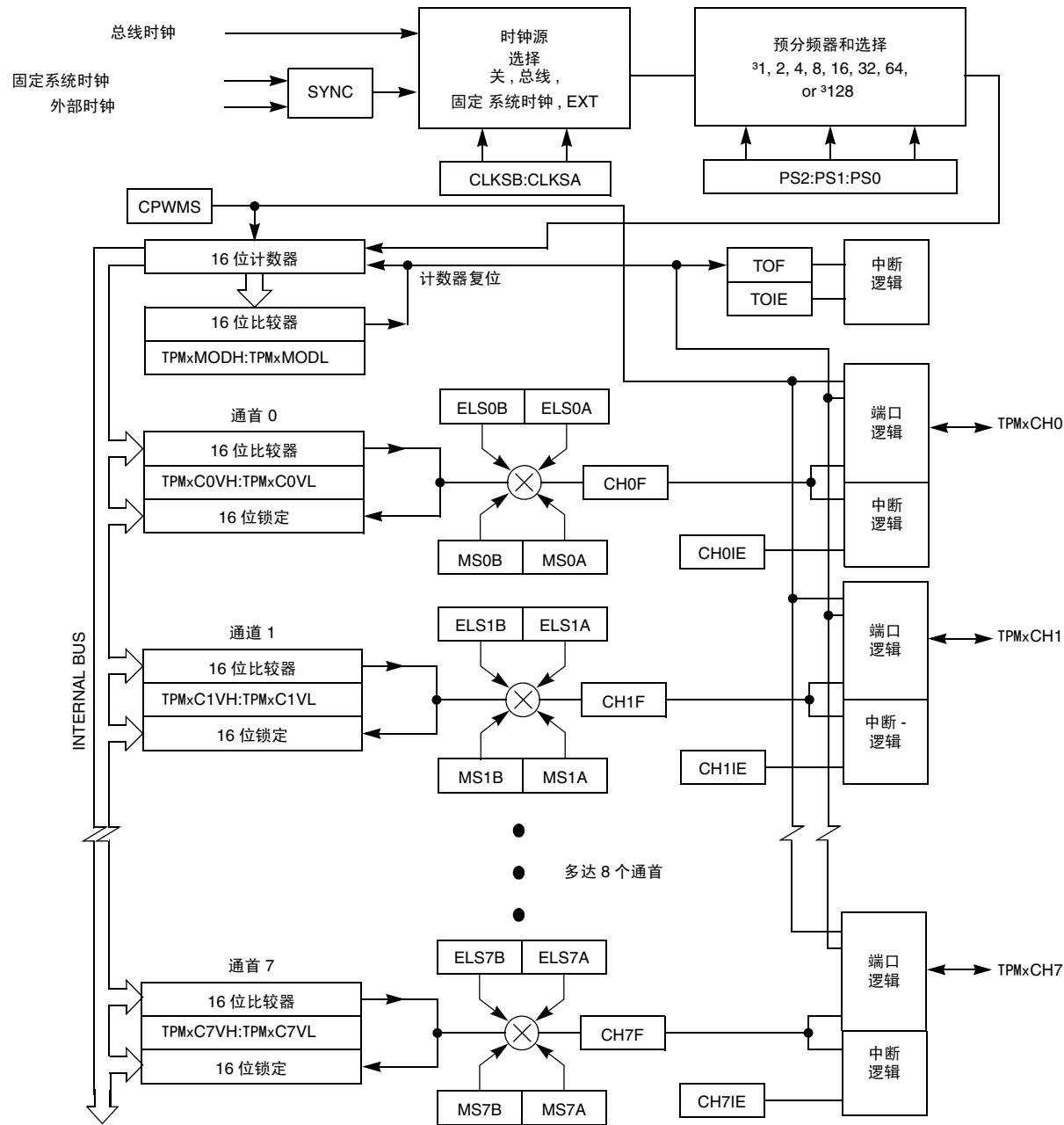


图 16-2. TPM 结构图

作为输入捕捉、输出比较或边缘对齐的 **TPM** 通道，**PWM** 通道是可独立编程的。或者，**TPM** 也可配置用于在所有通道上产生 **CPWM** 输出。当 **TPM** 被配置用于 **CPWM** 时，计数器作为向上 / 向下计数器运行；这种情况下输入捕捉、输出比较和 **EPWM** 功能是不可用的。

如果一个通道被配置为输入捕捉，那个该通道的内部上拉器件可以被使能。一个模块如何与管脚控制交互的细节取决于芯片实施，因为输入 / 输出管脚和相关通用输入 / 输出控制不是 **PWM** 模块的一部分。请参考全芯片规范中有关输入 / 输出端口逻辑的讨论。

因为中央对齐 **PWM** 通常用于驱动 3 相交流感应电机和无刷直流电机，它们一般用在 3 个或 6 个通道组中。

## 16.2 信号描述

**表 16-1** 显示了 **TPM** 的用户可接入信号。通道数目可从 1 到 8 之间变化。包含外部时钟时，它可与任何 **TPM** 通道一样通过相同的管脚共享；然而，它可以连接到独立的输入管脚。请参考全芯片规范中的输入 / 输出管脚介绍，了解具体的芯片执行。

**表 16-1. 信号属性**

名称	功能
EXTCLK <sup>1</sup>	可选择用于驱动 <b>TPM</b> 计数器的外部时钟源。
TPMxCHn <sup>2</sup>	与 <b>TPM</b> 通道 n 相关的输入 / 输出管脚

<sup>1</sup> 预设后，该信号可分享任何通道管脚；然而根据全芯片实施，这个信号可以连接到独立的外部管脚。

<sup>2</sup> n= 通道数目（1 至 8）

请参考全芯片文档，了解关于复位状态、端口连接的详细信息，以及这些管脚上是否有上拉器件。

当 **TPM** 或通用输入 / 输出控制将关联管脚配置为输入时，**TPM** 通道管脚可与通用输入 / 输出管脚相关，并且可以使用可通过控制位使能的被动上拉器件。当没有 **TPM** 功能被使能以使用关联的管脚时，管脚恢复到由通用输入 / 输出控制的状态，包括端口数据和数据方向寄存器。复位后不会有 **TPM** 功能立即使能，因此所有相关管脚都恢复到通用输入 / 输出控制。

### 16.2.1 详细信号描述

本节详细介绍了每种用户可接入的管脚信号。虽然**表 16-1** 对所有通道管脚进行了分组，但任何 **TPM** 管脚都可以和外部时钟源信号共享。由于输入 / 输出管脚逻辑不是 **TPM** 的一部分，请参考全芯片文档的具体描述来了解有关 **TPM** 管脚功能和包括端口数据、数据方向和上拉控制的通用输入 / 输出控制交互的更多详情。

#### 16.2.1.1 EXTCLK — 外部时钟源

通过定时器状态和控制寄存器中的控制位，用户可以选择无（定时器关闭）、总线速率时钟（正常默认源）、晶振相关时钟或外部时钟作为驱动 **TPM** 预分频器和随后的 16 位 **TPM** 计数器的时钟。外部时钟源在 **TPM** 中实现同步。总线时钟对同步器进行定时；外部源的频率必须不能超过总线速率时钟频率的四分之一，以满足 Nyquist 标准并允许抖动。

外部时钟信号与通道输入 / 输出管脚共享相同的管脚，因此选择为外部时钟源时通道管脚不能用于通道输入 / 输出功能。用户应负责避免这种设置。如果这个管脚被用作外部时钟源 ( $\text{CLKSB:CLKSA} = 1:1$ )，通道仍可作为软件定时器 ( $\text{ELSnB:ELSnA} = 0:0$ ) 用于输出比较模式。

### 16.2.1.2 TPMxCHn — 通道 n 输入 / 输出管脚

每个 TPM 通道都与 MCU 上的一个输入 / 输出管脚相关联。这个管脚的功能取决于通道配置。TPM 管脚与通用输入 / 输出管脚共享，其中每个管脚都有一个端口数据寄存器位和一个数据方向控制位，而且端口有可选的被动上拉器件。该上拉器件可在端口管脚作为输入设备时使能。

当 ( $\text{ELSnB:ELSnA} = 0:0$ ) 或 ( $\text{CLKSB:CLKSA} = 0:0$ ) 时，TPM 通道不会控制输入 / 输出管脚，因此通常恢复到由通用输入 / 输出控制的状态。当  $\text{CPWMS} = 1$  (and  $\text{ELSnB:ELSnA not} = 0:0$ ) 时，TPM 中的所有通道被配置用于中央对齐 PWM，而 TPMxCHn 管脚全部由 TPM 系统控制。当  $\text{CPWMS}=0$  时， $\text{MSnB:MSnA}$  控制位决定通道配置用于输入捕捉、输出比较还是边缘对齐 PWM。

当通道被配置用于输入捕捉 ( $\text{CPWMS}=0$ ,  $\text{MSnB:MSnA} = 0:0$  and  $\text{ELSnB:ELSnA not} = 0:0$ ) 时，TPMxCHn 管脚被强制用作 TPM 的对边缘敏感的输入。 $\text{ELSnB:ELSnA}$  控制位决定哪个或哪些极性边将触发输入捕捉事件。一个基于总线时钟的同步器用于同步输入边和总线时钟。这意味着输入捕捉管脚上可以可靠检测的最小脉冲宽度是 4 个总线时钟周期（可检测尽可能靠近 2 个总线时钟的最佳时钟脉冲）。TPM 使用该管脚作为输入捕捉输入，为相同管脚改写端口数据和数据方向控制。

当通道被配置用于输出比较 ( $\text{CPWMS}=0$ ,  $\text{MSnB:MSnA} = 0:1$  and  $\text{ELSnB:ELSnA not} = 0:0$ ) 时，相关数据方向控制被改写；TPMxCHn 管脚被看作是由 TPM 控制的输出， $\text{ELSnB:ELSnA}$  控制位决定如何控制管脚。 $\text{ELSnB:ELSnA}$  的其余三个组合决定在每次 16 位通道值寄存器与定时器计数器匹配时是否切换、清除或者设置 TPMxCHn 管脚。

刚完成选择输出比较切换模式时，管脚上的以前的值一直被驱动，直到发生下一个输出比较事件，然后管脚被切换。

当通道被配置用于边缘对齐 PWM (CPWMS=0, MSnB=1 and ELSnB:ELSnA not = 0:0) 时, 数据方向被修改; TPMxCHn 管脚被强制用作受 TPM 控制的输出, 而 ELSnA 控制管脚上 PWM 输出信号的极性。当 ELSnB:ELSnA=1:0 时, TPMxCHn 管脚在每个新周期开始时被强制进入高态 (TPMxCNT=0x0000); 管脚在通道值寄存器与定时器计数器匹配时强制进入低态。当 n ELSnA=1 时, TPMxCHn 管脚在每个新周期开始时被强制进入低态 (TPMxCNT=0x0000); 而管脚在通道值寄存器与定时器计数器匹配时强制进入高态。

TPMxMODH:TPMxMODL = 0x0008  
TPMxCnVH:TPMxCnVL = 0x0005

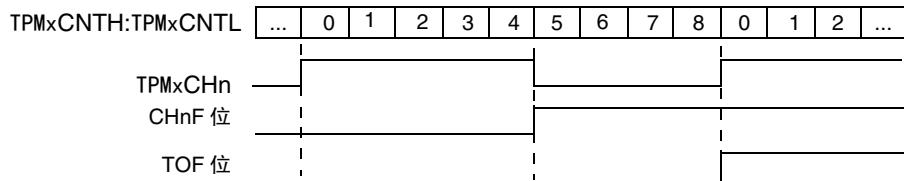


图 16-3. 边缘对齐 PWM 的 High-True 脉冲

TPMxMODH:TPMxMODL = 0x0008  
TPMxCnVH:TPMxCnVL = 0x0005

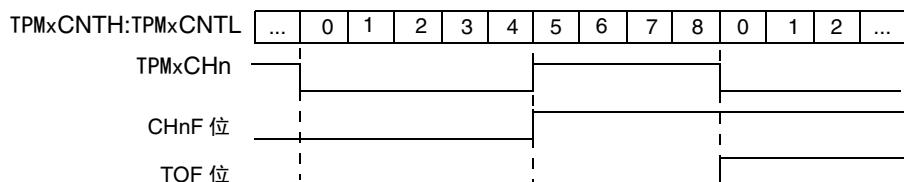


图 16-4. 边缘对齐 PWM 的 Low-True 脉冲

当 TPM 被配置用于中央对齐 PWM (and ELSnB:ELSnA not = 0:0) 时, 该 TPM 中所有通道的数据方向被改变; TPMxCHn 管脚被强制用作受 TPM 控制的输出, 而 ELSnA 位控制每个 TPMxCHn 输出的极性。如果 ELSnB:ELSnA=1:0, 那么在定时器计数器向上计数、通道值寄存器与定时器计数器匹配时, 相应的 TPMxCHn 管脚被清除; 当定时器计数器向下计数、通道值寄存器与定时器的计数器匹配时, TPMxCHn 管脚被设置。如果 ELSnA=1, 在定时器计数器向上计数、通道值寄存器与定时器的计数器匹配时, 相应的 TPMxCHn 管脚被设置; 当定时器计数器向下计数、通道值寄存器与定时器计数器匹配时, TPMxCHn 管脚被清除。

TPMxMODH:TPMxMODL = 0x0008  
TPMxCnVH:TPMxCnVL = 0x0005

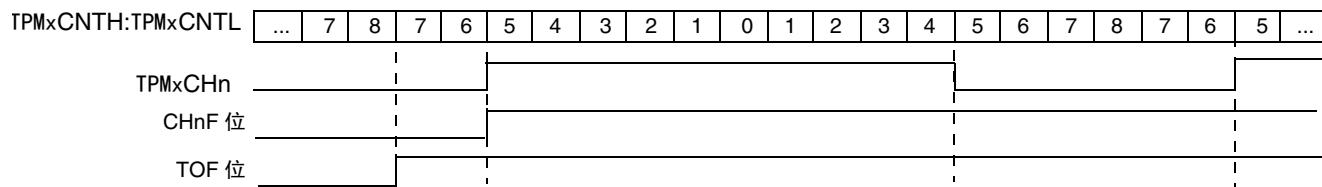


图 16-5. 中央对齐 PWM 的 High-True 脉冲

TPMxMODH:TPMxMODL = 0x0008  
TPMxCnVH:TPMxCnVL = 0x0005

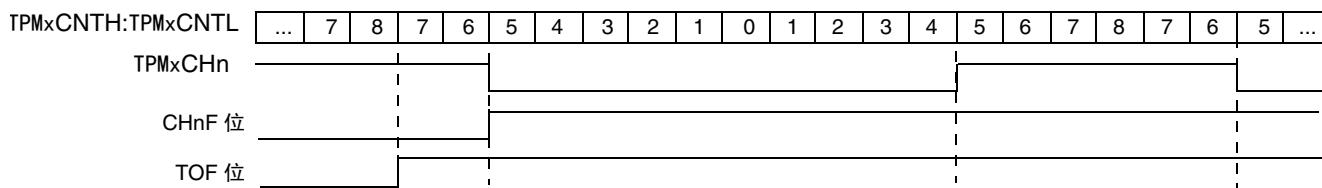


图 16-6. 中央对齐 PWM 的 Low-True 脉冲

## 16.3 寄存器定义

本小节包括按地址顺序排列的寄存器描述。

### 16.3.1 TPM 状态和控制寄存器 (TPMxSC)

TPMxSC 包含用于配置中断使能、TPM 配置、时钟源和预分频器等因素的溢出状态标志和控制位。这些控制与本定时器模块中的所有通道相关。

	7	6	5	4	3	2	1	0
R	TOF	TOIE	CPWMS	CLKSB	CLKSA	PS2	PS1	PS0
W	0							
复位	0	0	0	0	0	0	0	0

图 16-7. TPM 状态和控制寄存器 (TPMxSC)

表 16-2. TPMxSC 字段描述

字段	描述
7 TOF	定时器溢出标志。这个读 / 写标记在 TPM 定时器达到 TPM 计数器模数寄存器中设置的模数值后复位为 0x0000 时被设置。设置了 TOF 的情况下，读取 TPM 状态和控制寄存器，然后将逻辑 0 写入 TOF 可清除 TOF。如果清除序列完成前发生另一个 TPM 溢出，则序列被复位，因此为较早 TOF 完成清除序列后 TOF 仍将保持设置状态。这样做的目的是确保 TOF 中断请求在为前一个 TOF 完成清除序列期间不会丢失。复位可清除 TOF。向 TOF 写入逻辑数 1 是无效的 t。 0 TPM 计数器未达到模数值或未溢出 1 TPM 计数器已溢出。
6 TOIE	定时器溢出中断使能。这个读 / 写位使能 TPM 溢出中断。如果 TOIE 被设置，那么在 TOF 等于 1 时会生成中断。 复位可清除 TOIE。 0 TOF 中断关闭（用于软件轮询） 1 TOF 中断允许
5 CPWMS	中央对齐 PWM 选择。如果存在，这个读 / 写位选择 CPWM 运行模式。默认情况下，TPM 在执行输入捕捉、输出比较和边缘对齐 PWM 功能时以向上计数模式运行。设置 CPWMS 可重新配置 TPM，以便在执行 CPWM 功能时以向上 / 向下计数模式运行。复位可清除 CPWMS。 0 所有通道以输入捕捉、输出比较或边缘对齐 PWM 模式运行，即按每个通道的状态和控制寄存器中 MSnB:MSnA 控制位所选的模式运行。 1 所有通道以中央对齐 PWM 模式运行。
4-3 CLKSB[A]	时钟源选择。表 16-3 所示，这个 2 位字段用于关闭 TPM 系统或选择三个时钟源之一来驱动计数器预分频器。固定系统时钟源仅在采用基于 PLL 的系统时钟的系统中有意义。没有 PLL 时，固定系统时钟源与总线速率时钟相同。TPM 模块使外部源与总线时钟保持同步，而片上同步电路使固定系统时钟源（PLL 存在时）与总线时钟保持同步。当 PLL 存在但未使能时，固定系统时钟源与总线速率时钟相同。
2-0 PS[2:0]	预分频器因子选择。这个 3 位字段 表 16-4 所示为 TPM 时钟输入选择 8 个系数之一。任何时钟源同步或时钟源选择后，这个预分频器被定位，以便影响所选的驱动 TPM 系统的时钟源。新值被更新到寄存器位上之后，这个新的预分频器因子将在下一个系统时钟周期内影响时钟源。

表 16-3. TPM- 时钟源选择

CLKSB:CLKSA	预分频器输入的 TPM 时钟源
00	未选择时钟 (TPM 计数器关闭)
01	总线速率时钟
10	固定系统时钟
11	外部源

表 16-4. 预分频器因子选择

PS2:PS1:PS0	TPM 时钟源除数
000	1
001	2
010	4
011	8
100	16
101	32
110	64
111	128

### 16.3.2 计数器的寄存器 (TPMxCNTH:TPMxCNTL)

两个只读 TPM 计数器寄存器包含 TPM 计数器中值的高低字节。读取任何一个字节（无论是 TPMxCNTH 还是 TPMxCNTL）都会使两个字节的内容锁入到缓冲器中。这些内容一直被锁定在那里，直到另一半被读取。这允许按从小到大或从大到小的顺序进行连贯的 16 位读取，使它更加友好地适应各种编译器编译。一致性机制可通过 MCU 复位或定时器状态 / 控制寄存器 (TPMxSC) 写入操作自动重启。

复位可清除 TPM 计数器寄存器。此外，向 TPMxCNTH 或 TPMxCNTL 中写入任何值也可清除 TPM 计数器 (TPMxCNTH:TPMxCNTL) 并复位一致性机制，而不管写入操作所涉及的数据。

R	Bit 15	6	5	4	3	2	1	0
W	14	13	12	11	10	9	Bit 8	
任何向 TPMxCNTH 的写入操作都会清除 16 位计数器								
复位	0	0	0	0	0	0	0	0

图 16-8. TPM 计数器寄存器高字节 (TPMxCNTH)

	7	6	5	4		3	2	1	0	
R	Bit 7	6	5	4		3	2	1	Bit 0	
W	任何向 TPMxCNTL 的写入操作都会清除 16 位计数器									
复位	0	0	0	0		0	0	0	0	

图 16-9. TPM 计数器寄存器低字节 (TPMxCNTL)

当 BDM 处于有效状态，定时器计数器被冻结（这是用户将读取的值）；一致性机制是冻结的，这样当 BDM 处于有效状态时，缓冲器中锁定的内容仍将保持锁定状态，即使计数器的一半或两个一半都在 BDM 处于有效状态时被读取。这确保了如果 BDM 处于有效状态时用户正在读取 16 位寄存器，那么返回到正常操作后将从 16 位值的另一半中读取适当值。

在 BDM 模式下，向 TPMxSC, TPMxCNTH 或 TPMxCNTL 寄存器写入任何值可复位 TPMxCNTH:L 寄存器的读取一致性机制而不受写入涉及的数据影响。

### 16.3.3 TPM 计数器模数寄存器 (TPMxMODH:TPMxMODL)

这个读 / 写 TPM 模数寄存器包含 TPM 计数器的模数值。TPM 计数器达到模数值后，TPM 计数器在下一个时钟周期内又从 0x0000 开始计数，同时会设置一个溢出标记 (TOF)。向 TPMxMODH 或 TPMxMODL 中写入一个值可禁止 TOF 位和溢出中断，直到另一个字节也被写入。复位操作将 TPM 计数器的模数寄存器设置为 0x0000，进而导致自由运行的定时器计数器（模数被关闭）。

写入任何一个字节（无论是 TPMxMODH 还是 TPMxMODL）都会使值锁入到缓冲器中，同时寄存器会根据 CLKSB:CLKSA bits, so 位的值以它们的写入缓冲器的值得到更新，因此：

- 如果 (clkbs:clksa = 0:0)，那么寄存器在第二个字节被写入时更新。
- 如果 (clkbs:clksa not = 0:0)，那么寄存器在两个字节都被写入后更新，计数器从 (tpmxmodh:tpmxmodl - 1) 变为 (tpmxmodh:tpmxmodl)。如果 tpm 计数器为自由运行的计数器，那么 tpm 计数器从 0xffff 变为 0xffff 时会进行更新。

锁定机制可通过向 TPMxSC 地址（无论 BDM 是否活动）中写入一个值来手动复位。

当 BDM 处于有效状态时，一致性机制是冻结的（除非通过写入 TPMxSC register 寄存器复位）。这样，当 BDM 处于有效状态时，缓冲器中锁定的内容仍然保持锁定状态，即使模数寄存器的一半或两个一半都在 BDM 处于有效状态时被写入。当 BDM 处于有效状态时，任何向模数寄存器的写入行为都会绕过缓冲器锁定并直接写入到模数寄存器中。

	7	6	5	4		3	2	1	0	
R	Bit 15	14	13	12		11	10	9	Bit 8	
W	任何向 TPMxMODH 的写入操作都会清除 16 位计数器									
复位	0	0	0	0		0	0	0	0	

图 16-10. TPM 计数器模数寄存器高字节 (TPMxMODH)

	7	6	5	4	3	2	1	0
R	Bit 7	6	5	4	3	2	1	Bit 0
W								
复位	0	0	0	0	0	0	0	0

写入 TPM 模数寄存器前复位 TPM m 计数器，以避免造成首次计数器溢出发生时间的混乱。

### 16.3.4 TPM 通道 n 状态和控制寄存器 (TPMxCnSC)

TPMxCnSC 包含用于配置中断使能、通道配置和管脚功能的通道中断状态标志和控制位。

	7	6	5	4	3	2	1	0
R	CHnF	CHnIE	MSnB	MSnA	ELSnB	ELSnA	0	0
W	0							
复位	0	0	0	0	0	0	0	0
= 未执行或保留								

图 16-12. TPM 通道 n 状态和控制寄存器 (TPMxCnSC)

表 16-5. TPMxCnSC 字段描述

字段	描述
7 CHnF	通道 n 标志。通道 n 用作输入捕捉通道的情况下，通道 n 管脚上发生有效触发边沿时会设置这个读 / 写位。通道 n 为输出比较或边缘对齐 / 中央对齐 PWM 通道时，TPM 计数器寄存器中的值与 TPM 通道 n 值寄存器中的值匹配时会设置 CHnF。通道 n 用作边缘对齐 / 中央对齐 PWM 通道而占空比被设置为 0% 或 100% 的情况下，TPM 计数器寄存器中的值与 TPM 通道 n 值寄存器中的值匹配时将不设置 CHnF。 设置了 CHnF 而且使能了中断 (CHnIE = 1) 时会请求相应的中断。CHnF 可通过在 (CHnIE = 1) 时读取 TPMxCnSC，然后将逻辑 0 写入到 0 中来清除。如果清除序列完成前出现另一个中断请求，则序列被复位，以确保前一个 CHnF 的清除序列完成后 CHnF 仍被设置。这样做的目的是确保 CHnF 中断请求不会因清除以前 CHnF 而丢失。 重启可清除 CHnF 位。将逻辑数 1 写入 CHnF 是无效的。 0 通道 n 上没有发生输入捕捉或输出比较事件 1 通道 n 上发生输入捕捉或输出比较事件
6 CHnIE	通道 n 中断使能。这个读 / 写位使能来自通道 n 的中断。复位可清除 CHnIE。 0 通道 n 中断请求关闭 (用于软件轮询) 1 通道 n 中断请求允许
5 MSnB	TPM 通道 n 的模式 B 选择位。当 CPWMS=0 时，MSnB=1 为边缘对齐 TPM 模式配置 PWM 通道 n。请参考表 16-6 中的通道模式和设置控制总结。.

表 16-5. TPMxCnSC 字段描述 (continued)

字段	描述
4 MSnA	TPM 通道 n 的模式 A 选择位。当 CPWMS=0, MSnB=0 时, MSnA 为输入捕捉模式或输出比较模式配置 TPM 通道 n。请参见 表 16-6 中关于通道模式和设置控制的总结。 <b>注意:</b> 如果相关端口管脚在变为输入捕捉模式前至少 2 个总线时钟周期内是不稳定的, 则可能获得一个边缘触发的意外指示。
3-2 ELSnB ELSnA	边沿 / 电平选择位。根据 CPWMS:MSnB:MSnA 设置、表 16-6, 中所示的定时器通道的运行模式, 这些位选择触发输入捕捉事件的输入边的极性, 选择满足输出比较匹配后将驱动的电平, 或选择 PWM 输出的极性。 将 ELSnA 设置为 0:0 可将关联的定时器管脚配置为与任何定时器功能无关的通用输入 / 输出管脚。当关联的定时器通道被设置为不请求使用管脚的软件定时器时, 本功能常用于临时关闭输入捕捉通道或使定时器管脚可用作通用输入 / 输出管脚。

表 16-6. 模式、边沿和电平选择

CPWMS	MSnB:MSnA	ELSnB:ELSnA	模式	配置
X	XX	00	不用于 TPM 的管脚 - 恢复为通用输入 / 输出或其他外围设备控制	
0	00	01	输入捕捉	仅在上升边沿捕捉
		10		仅在下降边沿捕捉
		11		在上升或下降边沿捕捉
	01	01	输出比较	切换比较输出
		10		清除比较输出
		11		设置比较输出
	1X	10	边缘对齐 PWM	High-true p 脉冲 (清除比较输出)
		X1		Low-true p 脉冲 (设置比较输出)
	XX	10	中央对齐 PWM	High-true 脉冲 (清除向上比较输出)
		X1		Low-true 脉冲 (设置向上比较输出)

### 16.3.5 TPM 通道值寄存器 (TPMxCnVH:TPMxCnVL)

这些读 / 写寄存器包含输入捕捉功能捕捉的 TPM 计数器值, 或输出比较或 PWM 功能的输出比较值。该通道寄存器可通过复位清除。

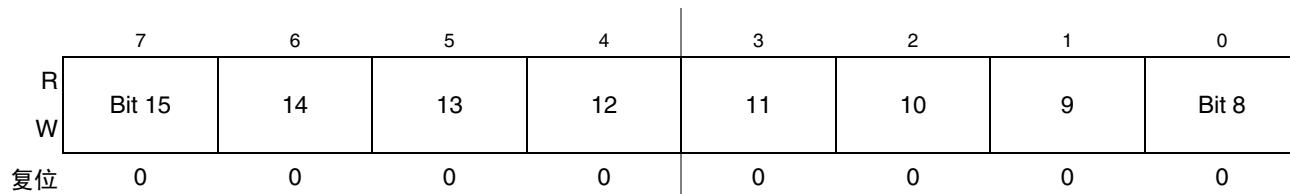


图 16-13. TPM 道值寄存器高字节 (TPMxCnVH)

	7	6	5	4		3	2	1	0
R	Bit 7	6	5	4	3	2	1	Bit 0	
W	0	0	0	0	0	0	0	0	0
复位	0	0	0	0	0	0	0	0	0

图 16-14. TPM 通道值寄存器低字节 (TPMxCnVL)

在输入捕捉模式下，读取任何一个字节（无论是 TPMxCnVH 还是 TPMxCnVL）都会使两个字节的内容锁定到缓冲器中。这些内容一直锁定在那里，直到另一半被读取。当 TPMxCnSC 寄存器被写入时（不管 BDM 模式是否使能），锁存机制可复位（为未锁定状态）。向通道寄存器的任何写入操作在输入捕捉模式下将被忽视。

BDM 处于有效状态时，一致性机制是冻结的（除非通过写入 TPMxCnSC 寄存器复位），这样缓冲器中锁定的内容将一直保持 BDM 处于有效状态时的锁定状态，即使通道寄存器的一半或两个一半都在 BDM 处于有效状态时被读取。这确保了如果 BDM w 处于有效状态时用户正在读取 16 位寄存器，那么在返回正常运行后它将从 16 位值的另一半中读取适当的值。在 BDM 模式下从 TPMxCnVH 和 TPMxCnVL 存器中读取的值是这些寄存器的值而不是它们的读取缓冲器的值。

在输出比较或 PWM 模式下，写入任何一个字节（无论是 TPMxCnVH 还是 TPMxCnVL）都会将该值锁定到缓冲器中。两个字节都被写入后，他们根据 CLKSB:CLKSA 位的值和所选模式作为连贯的 16 位值发送到定时器通道寄存器中，因此：

- 如果 (clksb:clksa = 0:0)，那么寄存器在第二个字节被写入时更新。
- 如果 (clksb:clksa not = 0:0 而且在 epwm 或 cpwm 模式下)，那么寄存器在第二个字节被写入后和 tpm 计数器下次发生变化（预分频器计数结束）时更新。
- 如果 (clksb:clksa not = 0:0 而且在 epwm 或 cpwm 模式下)，寄存器在两个字节被写入后以及 tpm 计数器从 (tpmxmodh:tpmxmodl - 1) 变为 (tpmxmodh:tpmxmodl) 时更新。如果 tpm 计数器为自由运行的计数器，那么更新在 tpm 计数器从 0xffff 变为 0xffff 时进行。

锁存机制可通过写入 TPMxCnSC 寄存器（不管 BDM 是否处于活动状态）来手动复位。这种锁定机制允许按从小到大或从大到小的顺序进行连贯的 16 位写入，这对各种编译器的编译都非常友好。

当 BDM 处于有效状态时，一致性机制被冻结，这样缓冲器中锁定的内容仍可保持 BDM 处于有效状态时的锁定状态，即使通道寄存器的一半或两个一半都在 BDM 处于有效状态时被写入。当处于有效状态时，向通道寄存器的任何写入操作都会绕过缓冲器锁定并直接写入到通道寄存器中。一旦恢复正常运行，BDM 处于有效状态时写入到通道寄存器的值用于 PWM 和输出比较操作。处于有效状态时，通道寄存器写入不会干扰一致性序列的部分完成。一致性机制全部运行完毕后，可通过用户写入的缓冲值（BDM 未处于活动状态时）更新通道寄存器。

## 16.4 功能描述

所有 TPM 功能都与一个允许灵活选择时钟源和预分频器因数的中央 16 位计数器相关。此外，还有一与主计数器关联的 16 位模数寄存器。

CPWMS 控制位可在 PWM 中所有通道的中央对齐 TPM 操作 (CPWMS=1) 或通用定时功能 (CPWMS=0) 间选择。在后一种情况下，每个通道可独立配置，以输入捕捉、输出比较或边缘对齐 PWM 模式运行。CPWMS 控制位位于主 TPM 状态和控制寄存器中，因为它会影响 TPM 中的所有通道，而且会影响主计数器的运行方式。(在 CPWM 模式下，计数器变为向上 / 向下模式，而不是用于通用定时器功能的向上计数模式。)

后面各小节介绍了主计数器及计数器的每一种运行模式（输入捕捉、输出比较、边缘对齐 PWM 和中央对齐 PWM）。因为管脚运行和中断活动的细节取决于运行模式，这些主题将在相关模式的说明部分中介绍。

### 16.4.1 计数器

所有定时器功能都基于 16 位主计数器 (TPMxCNTH:TPMxCNTL)。这一部分讨论时钟源的选择、计数终止溢出、向上计数和向下计数以及手动计数器复位。

#### 16.4.1.1 计数器时钟源

计数器状态和控制寄存器 (TPMxSC) 中的 2 位字段 CLKSB:CLKSAs 从三个可能的时钟源中进行选择或选择 OFF (可有效地关闭 TPM)。请参见表 16-3。任何 MCU 复位后，CLKSB:CLKSA=0:0，因此不会选择任何时钟源，TPM 处于非常低功耗的状态。这些控制位可随时读取或写入，关闭定时器 (将 00 写入到 CLKSB:CLKSA f 字段) 不会影响计数器或其他定时器寄存器中的值。

表 16-7. TPM 时钟源选择

CLKSB:CLKSA	预分频器输入的 TPM 时钟源
00	没有选择时钟 (TPM 计数器被关闭)
01	总线速率时钟
10	固定系统时钟
11	外部源

总线速率时钟是 MCU 的主系统总线时钟。这个时钟源不要求同步，因为它是用于所有内部 MCU 活动（包括 CPU 和总线运行）的时钟。

在没有 PLL 或没有使用 PLL 的 MCU 中，固定系统时钟源与总线速率时钟源相同，不需要经过同步器。当存在并使用了 PLL 时，在晶振 2 分时钟源和定时器计数器之间要求同步器，以确保计数器过渡与总线时钟过渡同步。同步器将用于芯片一级，以便将晶振相关源时钟与总线时钟同步。

外部时钟源可与任何 TPM 通道管脚连接。这种时钟源必须始终通过同步器，确保计数器过渡与总线时钟过渡保持同步。总线速率时钟驱动同步器；因此，要满足 Nyquist 标准甚至抖动，外部时钟源的频率必须不能高于总线速率的四分之一。使用适当时钟时，外部时钟可与总线时钟的四分之一一样快。

当外部时钟源共享 TPM 通道管脚时，该管脚不应用于其他通道计数功能。例如，当 TPM 通道 0 管脚同时也用作定时器外部时钟源时，将通道 0 配置用于输入捕捉会造成不明确。（用户应负责避免这种设置。）TPM 通道仍可用于输出比较模式以支持软件计时功能（管脚控制位不影响 TPM 通道管脚）。

#### 16.4.1.2 计数器溢出和模数复位

中断标记和使能与 16 位主计数器相关。标记 (TOF) 是显示定时器计数器溢出的软件可接入指标。不论何时 TOF 标记等于 1，使能信号都在软件轮询 (TOIE=0)（无硬件中断被生成）或中断驱动操作 (TOIE=1)（生成静态硬件中断）间选择。

导致 TOF 被设置的条件取决于 TPM i 是否被配置为中央对齐 PWM (CPWMS=1)。在最简单的模式下没有模数限制，TPM 也不在 CPWMS=1 模式中。在这种情况下，16 位定时器计数器从 0x0000 计数到 0xFFFF，然后在下一个计数时钟周期内溢出为 0x0000。在从 0xFFFF 过渡到 0x0000 时，TOF 被设置。设置了模数限制时，TOF 在从模数寄存器中设置的值向 0x0000 过渡时设置。当 TPM 处于中央对齐 PWM 模式时 (CPWMS=1)，TOF 标记在计数器到达模数寄存器中设置的计数值结束改变方向时被设置（也就是说从模数寄存器中设置的值向下一个更低计数值过渡时）。这与 PWM 周期结束对应（0x0000 计数值与周期中央对应）。

#### 16.4.1.3 计数模式

主定时器的计数器有两种计数模式。选择中央对齐 PWM 时，计数器以向上 / 向下计数模式运行。否则，计数器作为简单的向上计数器运行。作为向上计数器时，定时器计数器从 0x0000 计数到其终端计数，然后从 0x0000 重新开始。终端计数为 0xFFFF 或 TPMxMODH:TPMxMODL 中的模数值。

规定中央对齐 PWM 操作时，计数器从 0x0000 向上计数到其终端计数，然后向下计数到 0x0000，在那里又开始向上计数。0x0000 和终端计数值均为正常长度计数（一个定时器时钟周期长度）。在这种模式下，定时器溢出标记 (TOF) 在终端计数周期结束时被设置（当计数器变为下一个更低计数值时）。

#### 16.4.1.4 手动计数器复位

主定时器计数器可随时通过将任何值写入 TPMxCNTH 或 TPMxCNTL 的任何一半来手动复位。如果复位计数前只有一半计数器被读取，这种计数器复位方法还会复位一致性机制。

#### 16.4.2 通道模式选择

如果 Provided CPWMS=0，通道 n 状态和控制寄存器中 MSnA 和 MSnA 控制位为相应通道确定基本运行模式。选择包括输入捕捉、输出比较或边缘对齐 PWM。

##### 16.4.2.1 输入捕捉模式

利用输入捕捉功能，TPM 可捕捉外部事件发生的时间。当输入捕捉通道的管脚上发生活动边沿时，TPM 可将 TPM 计数器的内容锁入到通道值寄存器 (TPMxCnVH:TPMxCnVL) 中。上升边沿、下降边沿或任何边沿都可选择作为触发输入捕捉的使能边沿。

在输入捕捉模式下，TPMxCnVH 和 TPMxCnVL 寄存器为只读。

当 16 位捕捉寄存器的任何一半被读时，另一半被锁入到缓冲器中，以按从小到大或从大到小顺序支持连贯的 16 位接入。一致序列可通过向通道状态 / 控制寄存器 (TPMxCnSC) 中写入值来手动复位。

输入捕捉事件设置标记位 (CHnF)。该位可选择生成 CPU 中断请求。

在 BDM 中时，输入捕捉功能按用户配置的方式运行。发生外部事件时，TPM 将 TPM 计数器的内容（因为这在 BDM 模式下是冻结的）锁入到通道值寄存器中，并设置标志位。

##### 16.4.2.2 输出比较模式

利用输出比较功能，TPM 可生成具有可编程位置、极性、持续时间和频率的定时脉冲。当计数器达到输出比较通道的通道值寄存器中的值时，TPM 可设置、清除或切换通道管脚。

在输出比较模式下，根据 CLKSB:CLKSA 位值，仅在 16 位寄存器的两个一半（8 位）被写入后，值被传输到相应的定时器通道寄存器中，因此：

- 如果 (clkbs:clksa = 0:0)，寄存器在第二个字节被写入时更新。
- 如果 (clkbs:clksa not = 0:0)，寄存器在第二字节被写入后 tpm 计数器下次发生变化（预分频器计数结束）时更新。

一致性序列可通过向通道状态 / 控制寄存器 (TPMxCnSC) 中写入值来手动复位。

输出比较事件设置标记位 (CHnF)。该位可选择生成 CPU 中断请求。

### 16.4.2.3 边缘对齐 PWM 模式

这类 PWM 输出使用定时器计数器的正常向上计数模式 (CPWMS=0)；当相同 TPM 中的其他通道被配置用于输入捕捉或输出比较功能时，也可使用它。这个 PWM 信号的周期由模数寄存器 (TPMxMODH:TPMxMODL) 的值加 1 确定。占空比由定时器通道寄存器 (TPMxCnVH:TPMxCnVL) 中的设置确定。这个 PWM 信号的极性由 ELSnA 控制位中的设置确定。0% 和 100% 的占空比都是可能的。

TPM 通道寄存器中的输出比较值决定 PWM 信号的脉冲宽度（占空比）（图 16-15）。T 模数溢出和输出比较间的时间间隔为脉冲宽度。如果 ELSnA=0，计数器溢出强迫 PWM 信号进入高态；而输出比较强制 PWM 信号进入低态。如果 ELSnA=1，计数器溢出强制 PWM 信号进入低态；而输出比较强制 PWM 信号进入高态。

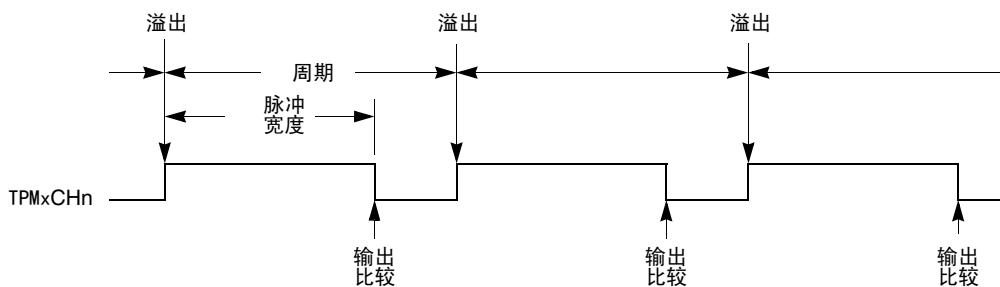


图 16-15. PWM 周期和脉冲宽度 (ELSnA=0)

当通道值寄存器被设为 0x0000, 时，占空比为 0%. 100%。通过将定时器通道计数器 (TPMxCnVH:TPMxCnVL) 设置为大于模数设置的值可实现 . 的占空比。这意味着要实现 100% 的占空比，模数设置必须小于 0x FFFF。

因为 TPM 可用在 8 位 MCU 中，定时器通道寄存器中的设置被缓存，以确保连贯的 16 位更新并避免意外的 PWM 脉冲宽度。写入到任何寄存器 TPMxCnVH 和 TPMxCnVL 中意味着实际写入到缓冲器寄存器中。在边缘对齐 PWM 模式下，值根据 CLKSB:CLKSA 位的值被传输到相应的定时器通道寄存器中，因此：

- 如果 (clkbs:clksa = 0:0)，寄存器在第二个字节被写入时更新
- 如果 (clkbs:clksa not = 0:0)，寄存器在两个字节都被写入，tpm 计数器从 (tpmxmodh:tpmxmodl - 1) 变为 (tpmxmodh:tpmxmodl) 后更新。如果 tpm 计数器为自由运行的计数器，那么更新在 tpm 计数器从 0xffffe 变为 0xffff 时进行更新。

### 16.4.2.4 中央对齐 PWM 模式

这类 PWM 输出使用定时器计数器 (CPWMS=1) 的向上 / 向下计数模式。

TPMxCnVH:TPMxCnVL 中的输出比较值决定 PWM 信号的脉冲宽度（占空比）而 TPMxMODH:TPMxMODL 中的值决定周期。TPMxMODH:TPMxMODL 应保持在 0x0001 至 0x7FFF 之间，因为这一范围外的值可生成模糊结果。ELSnA 将决定 CPWM 输出的极性。

$$\text{脉冲宽度} = 2 \times (\text{TPMxCnVH:TPMxCnVL})$$

$$\text{周期} = 2 \times (\text{TPMxMODH:TPMxMODL}) ; \text{TPMxMODH:TPMxMODL}=0x0001-0x7FFF$$

如果通道值寄存器 TPMxCnVH:TPMxCnVL 为零或负数（位 15 被设置），占空比将为 0%。如果 TPMxCnVH:TPMxCnVL 是正值（位 15 被清除）并大于模数设置（非零），占空比将为

100%，因为占空比比较将不会发生。这意味着模数寄存器设置的可用范围周期为 0x0001 至 0x7FFE（如果不需要 100% 的占空比，则为 0x7FFF）。这不是一个重大的限制。所能产生的周期将远远长于正常应用所需的周期。

TPMxMODH:TPMxMODL=0x0000 是不用于中央对齐 PWM 模式的特例。当 CPWMS=0 时，这一情况与计数器从 0x0000 自由运行到 0xFFFF 的情况相对应，但当 CPWMS=1 时，计数器需要与 0x0000 以外的模数寄存器有效匹配，以便将方向从向上计数变为向下计数。

TPM 通道寄存器（2 倍）中的输出比较值决定 CPWM 信号的脉冲宽度（占空比）（图 16-16）。如果 ELSnA=0，当向上计数时发生数值比较会强制 CPWM 输出信号进入低态；当向下计数时发生数值比较会强制输出进入高态。计数器达到 TPMxMODH:TPMxMODL 中的模数设置后才开始向上计数；然后向下计数直到 0。这将周期设置为 TPMxMODH:TPMxMODL 的两倍。

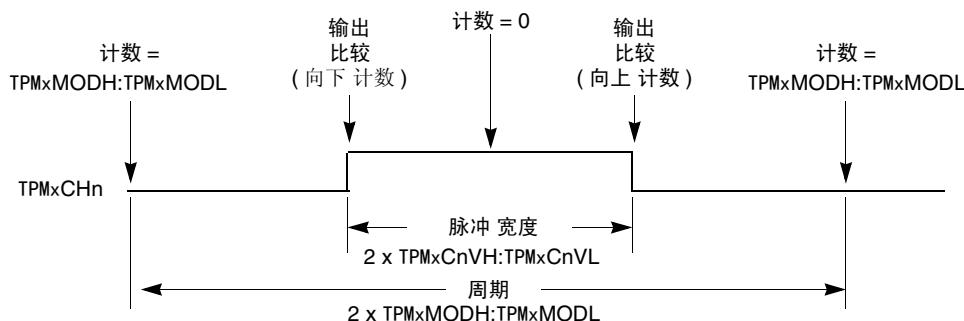


图 16-16. CPWM 周期和脉冲宽度 (ELSnA=0)

中央对齐 PWM 输出的噪音一般比边缘对齐 PWM 小，因为相同系统时钟边上的输入 / 输出管脚过渡更少。有些类型的电机需要这类 PWM 应用。

当计数器以向上 / 向下计数模式运行时，输入捕捉、输出比较和边缘对齐 PWM 功能没有意义。因此这意味着当 CPWMS=1 时，TPM 中的所有使能通道必须用于 CPWM 模式中。

TPM 可用在 8 位 MCU 中。定时器通道寄存器中的设置被缓冲，以确保连贯的 16 位更新并避免意外的 PWM 脉冲宽度。向任何寄存器 TPMxMODH、TPMxMODL、TPMxCnVH 和 TPMxCnVL 中写入实际上就是写入到缓冲器寄存器中。

在中央对齐 PWM 模式下，TPMxCnVH:L 寄存器根据 CLKSB:CLKSA 位的值通过写入缓冲器的值得到更新，因此：

- 如果 (clkbs:clksa = 0:0)，寄存器在第二个字节被写入时更新。
- 如果 (clkbs:clksa not = 0:0)，寄存器在两个字节都被写入，tpm 计数器从 (tpmxmodh:tpmxmodl - 1) 变为 (tpmxmodh:tpmxmodl) 后更新。如果 tpm 计数器为自由运行的计数器，那么更新在 tpm 计数器从 0xffff 变为 0xffff 时进行。

当 TPMxCNTH:TPMxCNTL=TPMxMODH:TPMxMODL 时，TPM 可选择生成 TOF 中断（在该计数结束时）。

写入 TPMxSC 的操作会取消写入到 TPMxMODH 和 / 或 TPMxMODL 中的任何值，并且复位模数寄存器的一致性机制。写入 TPMxCnSC 的操作会取消写入到通道值寄存器中的任何值，并且为 TPMxCnVH:TPMxCnVL 复位一致性机制。

## 16.5 复位概述

### 16.5.1 概况

MCU 复位时 TPM 就会被复位。

### 16.5.2 复位操作介绍

复位清除 TPMxSC 寄存器，导致关闭 TPM 的时钟源，并禁止定时器溢出中断（TOIE=0）。CPWMS、MSnB、MSnA、ELSnB 和 ELSnA 可被全部清除，这使相关的被配置为输入捕捉功能的所有 TPM 通道与 I/O 口逻辑断开（因此与 TPM 相关的所有 MCU 管脚恢复到通用输入 / 输出管脚）。

## 16.6 中断

### 16.6.1 General

TPM 为主计数器溢出产生可选的中断，或为每个通道生成中断。通道中断的意义取决于每个通道的运行模式。通道中断的意义取决于每个通道的运行模式。如果通道配置用于输入捕捉，那么中断标志在每次所选的输入捕捉边沿被识别时设置。如果通道配置用于输出比较或 PWM 模式，那么中断标志在每次主定时器计数器与 16 位通道值寄存器中的值匹配时被设置。

表 16-8 中列出所有 TPM 中断，其中显示了中断名称及任何本地使能的名称。这些本地使能可促使中断请求离开 TPM 或被不同的处理逻辑识别。

表 16-8. 中断总结

中断	本地启动	源	描述
TOF	TOIE	定时器溢出	每次定时器计数器达到其终端计数（过渡到通常为 0x0000 的下一计数值）时设置
CHnF	CHnIE	通道事件	通道 n 上发生输入捕捉或输出比较事件 TPM 模块将提供 high-true 中断信号。

向量和优先级在中断模块中进行芯片集成时确定，因此请参考用户指南，查看有关中断模块或芯片的全部文档了解更详细信息。

### 16.6.2 中断操作描述

对于 TPM 中的每个中断源，标志位在识别到中断条件后设置，如定时器溢出、通道输入捕捉或输出比较事件等。这个标志可被软件读取（轮询），以确定操作已经发生，或者相关使能位（TOIE 或 CHnIE）可设置以便使能硬件中断生成。设置了中断使能位时，不论何时相关中断标记等于 1，就会生成静态中断。从中断服务程序中返回前，用户的软件必须执行一系列步骤来清除中断标志。

TPM 中断标志通过两步清除：一是标志位被设置（为 1）时读取，二是向其中写入 0。如果这两步之间检测到新事件，序列被复位，并且在完成第二步后中断标志仍被设置以避免丢失新事件。

### 16.6.2.1 定时器溢出中断 (TOF) 介绍

根据 TPM 系统的运行模式（通用定时功能和中央对齐 PWM 操作），TOF 中断操作的意义和细节会有细微变化。标志通过上述两步序列清除。

#### 16.6.2.1.1 常见情况

当定时器计数器从 0xFFFF 变为 0x0000 时，TOF 通常被设置。当 TPM 没有被配置为中央对齐 PWM 时（CPWMS=0），TOF 在定时器计数器从终端计数（模数寄存器中的值）变为 0x0000 时被设置。这种情况与计数器溢出的正常意义相对应。.

#### 16.6.2.1.2 中央对齐 PWM 情况

当 CPWMS=1 时，TOF 在定时器计数器方向在终端计数（模数寄存器中的值）结束时从向上计数变为向下计数时被设置。在这种情况下，TOF 与 PWM 周期结束相对应。

### 16.6.2.2 通道事件中断描述

通道中断的意义取决于通道的当前模式（输入捕捉、输出比较、边缘对齐 PWM 或中央对齐 PWM）。

#### 16.6.2.2.1 输入捕捉事件

当通道被配置为输入捕捉通道时，ELSnB:ELSnA 控制位选择无边沿（关）、上升边沿、下降边沿或任何边沿作为触发输入捕捉事件的边沿。检测到选定边沿时，中断标志被设置。标志通过 [16.6.2，“中断操作描述”](#) 中所述的两步序列清除。

#### 16.6.2.2.2 输出比较事件

如果通道被配置为输出比较通道，中断标志在每次主定时器计数器与通道值寄存器中的 16 位值匹配时被设置。标志通过 [16.6.2，“中断操作描述”](#) 中所述的两步序列清除。

#### 16.6.2.2.3 PWM 占空比结束事件

对于配置用于 PWM 操作的通道，有两种可能性。当通道配置用于边缘对齐 PWM 时，通道标志在定时器计数器与标志占空比周期结束的通道值寄存器匹配时被设置。当通道配置为中央对齐 PWM 时，定时器计数在每个 PWM 周期内两次与通道值寄存器相匹配。在这种 CPWM 情况中，通道标志在占空比周期时间开始和结束时（定时器计数器与通道值寄存器匹配时）被设置。标志通过 [16.6.2，“中断操作描述”](#) 中所述的两步序列清除。

## 从 TPM v2 到 TPM v3 的差异

### 1. 写入到 TPMxCnTH:L 寄存器中 (16.3.2, “计数器的寄存器 (TPMxCNTH:TPMxCNTL) ”) [SE110-TPM 案例 7]

向 TPM v3 中任何 TPMxCNTH 或 TPMxCNTL 寄存器的写入可清除 TPM 计数器 (TPMxCNTH:L) 和预分频器计数器。然而，在这种情况下，TPM v2 中只有 TPM 计数器被清除。

### 2. TPMxCNTH:L 寄存器的读取 (16.3.2, “计数器的寄存器 (TPMxCNTH:TPMxCNTL) ”)

- 在 TPM v3 中，BDM 模式下 TPMxCNTH:L 寄存器的任何读取都会返回冻结的 TPM 计数器的值。在 TPM v2 中，如果 BDM 模式使能前只有 TPMxCNTH:L 寄存器的一个字节被读取，那么 BDM 模式下对 TPMxCNTH:L 寄存器的任何读取都将从读取缓冲器而不是冻结的 TPM 计数器值中返回 TPMxCNTH:L 的锁定值。
- 如果有向 TPMxSC、TPMxCNTH 或 TPMxCNTL 的写入，那么该读取一致性机制将在 BDM 模式下从 TPM v3 中清除。然而，在这些情况下，TPM v2 不会清除这种读取一致性机制。

### 3. TPMxCnVH:L 寄存器的读取 (16.3.5, “TPM 通道值寄存器 (TPMxCnVH:TPMxCnVL) ”)

- 在 TPM v3 中，BDM 模式下对 TPMxCnVH:L 寄存器的任何读取都会返回 TPMxCnVH:L 寄存器中的值。在 TPM v2 中，如果 BDM 模式使能前只有 TPMxCnVH:L 寄存器的一个字节被读取，那么 BDM 模式下对 TPMxCnVH:L 寄存器的任何读取都将从读取缓冲器而不是 TPMxCnVH:L 寄存器值中返回 TPMxCNTH:L 的锁定值。
- 如果有 TPMxCnSC 写入操作，该读取一致性机制在 BDM 模式下在 TPM v3 中被清除。然而，在这种情况下，TPM v2 不会清除这种读取一致性机制。

### 4. 写入到 TPMxCnVH:L 寄存器

#### — 输入捕捉模式 (16.4.2.1, “输入捕捉模式”)

在此模式下，TPM v3 不允许向 TPMxCnVH:L 寄存器写入任何值。然而，TPM v2 允许写入。

#### — 输出比较模式 (16.4.2.2, “输出比较模式”)

在此模式下，如果 (CLKSB:CLKSA not = 0:0)，那么 TPM v3 会在第二个字节被写入、TPM 计数器再次发生变化（预分频器计数结束）时使用写入缓冲器的值更新 TPMxCnVH:L 寄存器。然而，TPM v2 总是在它们的第二个字节被写入时更新这些寄存器。

#### — 边缘对齐 PWM (16.4.2.3, “边缘对齐 PWM 模式”)

在此模式下，如果 (CLKSB:CLKSA not = 0:0)，TPM v3 在两个字节都被写入，TPM 计数器从 (TPMxMODH:L - 1) 变为 (TPMxMODH:L) 时使用写入缓冲器的值更新 TPMxCnVH:L 寄存器。如果 TPM 计数器为自由运行的计数器，那么更新在 TPM 计数器从 \$FFFE 变为 \$FFFF 时进行。然而，TPM v2 在两个字节都被写入，且当 TPM 计数器从 TPMxMODH:L 变为 \$0000 时进行更新。

- 中央对齐 PWM ([16.4.2.4, “中央对齐 PWM 模式”](#))

在此模式下, 如果 (CLKSB:CLKSA not = 0:0), TPM v3 在两个字节都被写入, 且 TPM 计数器从 (TPMxMODH:L - 1) 变为 (TPMxMODH:L) 时使用写入缓冲器的值更新 TPMxCnVH:L 寄存器。如果 TPM 计数器为自由运行的计数器, 那么更新在 TPM 计数器从 \$FFFE 变为 \$FFFF 时进行。然而, TPM v2 在两个字节都被写入, 且 TPM 计数器从 TPMxMODH:L 变为 (TPMxMODH:L - 1) 时进行更新。

5. 中央对齐 PWM ([16.4.2.4, “中央对齐 PWM 模式”](#))

- TPMxCnVH:L = TPMxMODH:L [SE110-TPM 案例 1]

在这种情况下, TPM v3 生成 100% 的占空比。然而, TPM v2 生成 0% 的占空比。

- TPMxCnVH:L = (TPMxMODH:L - 1) [SE110-TPM 案例 2]

在这种情况下, TPM v3 生成几乎 100% 的占空比。然而, TPM v2 生成 0% 的占空比。

- TPMxCnVH:L 从 0x0000 变为非零值 [SE110-TPM 案例 3 和 5]

在这种情况下, TPM v3 等待新的 PWM 时期开始, 以便开始使用新占空比设置。然而, TPM v2 在当前 PWM 周期的中间改变通道输出 (当计数达到 0x0000 时)。

- TPMxCnVH:L 从非零值变为 0x0000 [SE110-TPM 案例 4]

在这种情况下, TPM v3 使用旧的占空比设置完成当前的 PWM 周期。然而, TPM v2 使用新占空比设置完成当前的 PWM 周期。

6. 在 BDM 模式下写入到 TPMxMODH:L 寄存器 ([16.3.3, “TPM 计数器模数寄存器 \(TPMxMODH:TPMxMODL\)”](#))

在 TPM v3 中, BDM 模式下向 TPMxSC 寄存器的写入可清除 TPMxMODH:L 寄存器的写入一致性机制。然而, 在 TPM v2 中, TPMxSC 寄存器写入操作不会清除这个一致性机制。

# 第 17 章 开发支持

## 17.1 介绍

HCS08 中的开发支持系统包括背景调试控制器 (BDC) 和片上调试模块 (DBG)。BDC 提供单线调试接口，与目标连接，通过这个接口可以方便地进行片上闪存和其它非易失性存储器的编程。BDC 也是开发用的主要调试接口，允许以非侵入式方式存取存储器数据和传统调试功能，如 CPU 寄存器修改、断点和单指令跟踪命令等。

在 HCS08 产品系列中，外部管脚不包括地址和数据总线信号（即使在测试模式也不包括）。调试的执行是通过单线背景调试接口向目标 MCU 传输命令来实现的。调试模块提供了一种有选择性地触发和捕获总线信息的方式，这样外部开发系统可以对 MCU 内发生的事件按周期进行重现，而不需要从外部存取 MCU 的地址和数据信号。

### 17.1.1 强制激活背景调试

强制激活背景调试模式的方式取决于具体的 HCS08 衍生产品。对，你可以在上电复位后强制激活背景调试，即当器件退出复位时保持 BKGD 引脚为低。你还可以通过将 SBDFR 寄存器的 BDFR 位置 1 的串行背景调试命令之后拉低 BKGD 管脚从而强制激活背景调试中的 BKGD 管脚后立即使低。如果没有调试盒连接到 BKGD 管脚，MCU 将总是复位到正常操作模式。

## 17.1.2 特性

BDC 模块的特性包括：

- 单引脚进行模式选择和背景调试通信
- BDC 的寄存器不位于存储器地址中
- SYNC 命令确定目标通信速率
- 非侵入式命令进行存储器存取
- 供 CPU 寄存器存取的激活背景调试模式命令
- GO 和 TRACE1 命令
- 背景调试命令可以将 CPU 从停止模式或等待模式中唤醒
- BDC 内置一个硬件地址断点
- 如果 BDC 使能，则振荡器运行在停止模式
- 处于激活背景调试模式时，COP 看门狗禁止

ICE 系统的特性包括：

- 两个触发比较器：两个地址 + 读 / 写 (R/W) 或一个完整地址 + 数据 + R/W
- 灵活的 8-word x 16-bit FIFO (先进先出) 缓存，用于捕获信息：
  - 流程变化的地址 或
  - 纯事件数据
- 两个类型的断点：
  - 指令操作码的标记断点
  - 任何地址存取的强制断点
- 九个触发模式：
  - 基本：只有 A, A 或 B
  - 顺序：A 然后 B
  - 全部：A 和 B 数据，A 和非 B 数据
  - 事件 (存储数据)：纯事件 B, A 然后纯事件 B
  - 范围：在范围以内 ( $A \leqslant$  地址  $\leqslant B$ )，在范围以外 ( 地址  $< A$  或地址  $> B$ )

## 17.2 背景调试控制器 (BDC)

HCS08 系列中的所有 MCU 都包含一个单线背景调试接口，它支持片上非易失性存储器的在线编程和先进的非侵入式调试功能。与早期的 8-位 MCU 的调试接口不同，这个系统不干扰正常的应用资源。它不使用任何用户存储器或存储器映射中的地址，也不分享任何片上外设。

BDC 命令分为两大组：

- 激活背景调试模式命令要求目标MCU处于激活背景调试模式(用户程序未运行)。激活背景调试模式命令允许读写 CPU 寄存器，允许用户一次跟踪一个用户指令，或从激活背景调试模式进入用户程序。
- 非侵入式命令可以随时执行，即使用户的程序正在运行。  
非侵入式命令允许用户在背景调试控制器中读写 MCU 存储器位置或存取状态和控制寄存器。

一般地，可以用相当简单的接口盒将来自主机的命令转换为与单线背景调试系统连接所需的串行命令。根据开发工具供应商的不同，这个接口盒可以采用标准 232 串行端口，或是并行打印端口，或是其它类型的通信端口，如用来与 PC 通信的 USB 接口。这个接口盒一般通过接地、

**BKGD** 管脚、**RESET**，有时还有 **VDD** 与目标系统连接。**RESET** 引脚的开漏连接允许主机强制目标系统复位，这有助于重新获得对已失目标系统的控制，或在片上非易失性存储器重新编程之前，控制目标系统的启动。有时可以用 **VDD** 来允许接口盒使用目标系统的电源，避免再使用一个电源。但是，如果单独对接口盒供电，它可以连接到一个正在运行的目标系统，而不必强制目标系统复位，否则会干扰正在运行的应用程序。

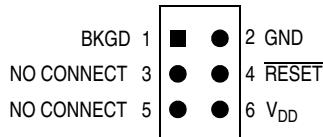


图 17-1. 工具接口

### 17.2.1 BKGD 管脚描述

**BKGD** 是单线背景调试接口管脚。这个管脚的主要功能是实现激活背景调试模式命令和数据的双向串行通信。在复位过程中，这个管脚用来选择激活背景调试模式启动或启用用户的应用程序。这个管脚还用来请求定时同步响应脉冲，允许主机开发工具确定背景调试串行通信的正确时钟频率。

**BDC** 串行通信采用首先引入在微处理器 M68HC12 系列上的定制串行协议。这个协议假定主机知道通信时钟速率，这个速率按目标 **BDC** 所有通信通过主机启动和控制所有通信，主机驱动高到低边沿发出每个位时间开始信号。命令和数据以最重要的位先发 (**MSB 先发**) 的方式发送。有关通信协议的详细信息，请参见 [17.2.2，“通信详细介绍”](#)。

如果主机尝试与 **BDC** 时钟速率未知的目标 MCU 沟通，可以发送 **SYNC** 命令给目标 MCU，请求定时同步响应信号，通过这个信号，主机可以判断正确的通信速率。

**BKGD** 是伪开漏管脚，有一个片上上拉，因此不需要外部上拉电阻。与典型的开漏管脚不同，管脚上的外部 RC 时间常数（受外部容性的影响），在信号上升时间上几乎不起作用。定制协议提供瞬态加速脉冲，强制提高这个管脚的上升时间，而没有驱动电平冲突风险。参见 [17.2.2，“通信详细介绍”](#)，了解更多详情。

当没有调试盒连接 6- 管脚的 BDM 接口连接器时，BKGD c 的内部上拉会选择正常的操作模式。当调试盒连接到 BKGD 时，可以在 MCU 复位后强制它进入激活背景调试模式。强制激活背景调试的具体条件取决于 HCS08 衍生产品（参见“开发支持”小节的介绍）。不必复位目标 MCU 来通过背景调试接口来与之通信。

## 17.2.2 通信详细介绍

BDC 串行接口需要外部控制器来生成 BKGD 管脚上的下降沿，指示每个位时间的开始。无论数据是发送或接收，外部控制器都会提供这个下降边沿。

BKGD 是伪开漏管脚，可以被外部控制器或 MCU 来驱动。数据以 MSB 先发的形式且以每位 16 个 BDC 时钟周期的速率（标定速率）发送。如果来自主机的下降边沿之间产生 512 BDC 时钟周期，则该接口超时。后果出现超时，任何正在进行的 BDC 命令被中止，对目标 MCU 系统的存储器或操作模式没有影响。

定制串行协议要求调试盒知道目标 BDC 通信时钟速率。

BDC 状态和控制寄存器中的时钟开关 (CLKSW) 控制位允许用户选择 BDC 时钟源。BDC 时钟源可以是总线，或备用的 BDC 时钟源。

BKGD 管脚可以接收高或低电平，或发送高或低电平。下图显示了每种情况的时序。接口时序与目标 BDC 中的时钟同步，但是与外部主机异步。显示的内部 BDC 时钟信号是计数周期的参考。

图 17-2 显示了外部主机将逻辑 1 或 0 发送到目标 HCS08 MCU 的 BKGD 管脚。主机与目标异步，因此主机生成的 BKGD 下降边沿与目标所认为的位时间起始点有 0- 到 -1 周期的延迟。10 个目标 BDC 时钟周期后，目标获得 BKGD 管脚的电平。一般地，主机在主机到目标方向的传输过程中驱动 BKGD 管脚，以加快上升边沿。由于目标在主机至目标方向的传输周期中不驱动 BKGD 管脚，因此没有必要在此期间将线路作为开漏信号。

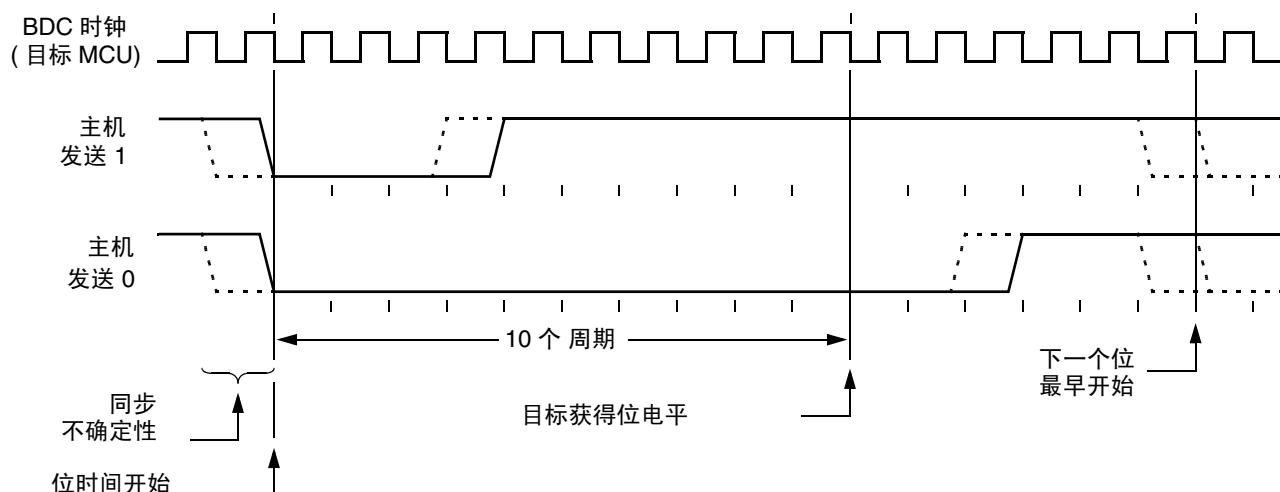


图 17-2. BDC 主机到目标方向串行位时序

图 17-3 显示主机从目标 HCS08 MCU 收到逻辑 1。由于主机与目标异步，因此主机生成的 BKGD 上的下降边沿与目标 MCU 所认为的位时间起始点有 0 到 1 个周期的延迟。主机保持低 BKGD 管脚足够长的时间，使目标识别它（至少两个目标 BDC 周期）。主机必须在目标 MCU 在其认为的位计时开始后驱动瞬时高态加速脉冲七个周期前，释放低电平驱动。主机应该在其启动位时间约 10 个周期后采样位电平。

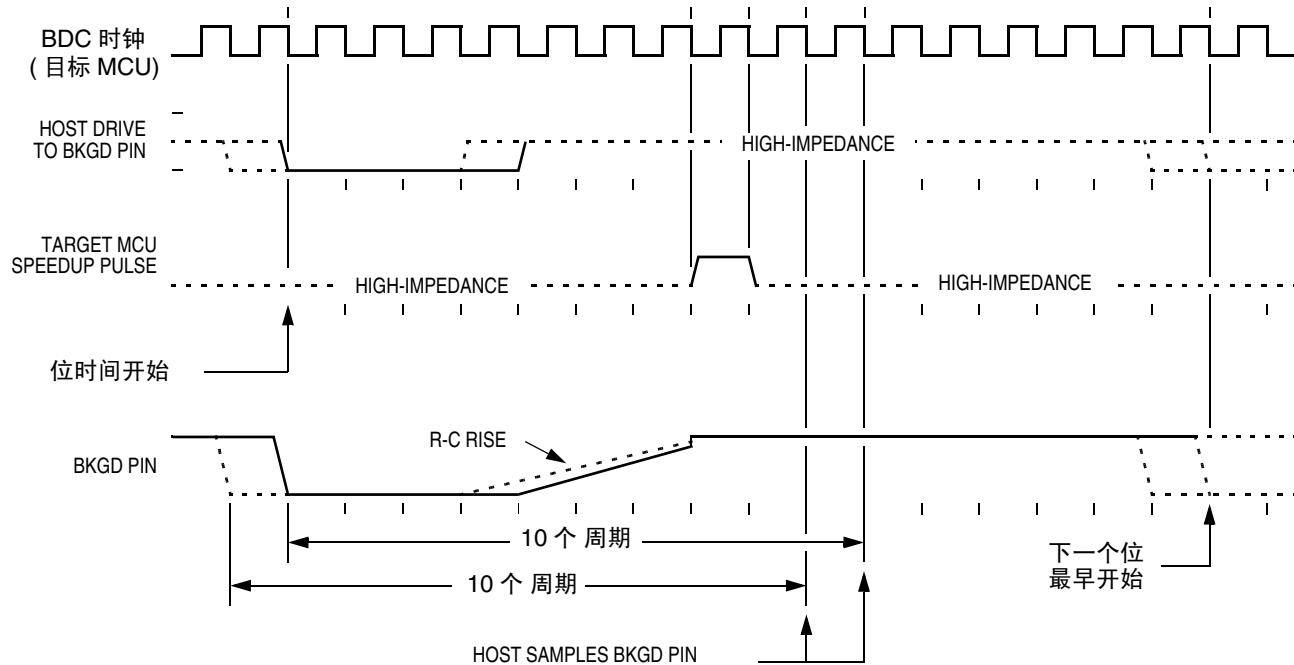


图 17-3. BDC 目标 - 到 - 主机串行位时序 (逻辑 1)

图 17-4 显示了主机从目标 HCS08 MCU 收到逻辑 0。由于主机与目标异步，因此主机生成的 BKGD 上的下降边沿与目标 MCU 所认为的位时间起始点有 0 到 1 个周期的延迟。主机启动位时间，但是目标 HCS08 MCU 完成它。由于目标希望主机接收逻辑 0，它保持低 BKGD 管脚 13 个 BKGD 管脚周期，然后驱动引脚置高，加速上升沿。主机在启动位时间约 10 个周期后采样位电平。

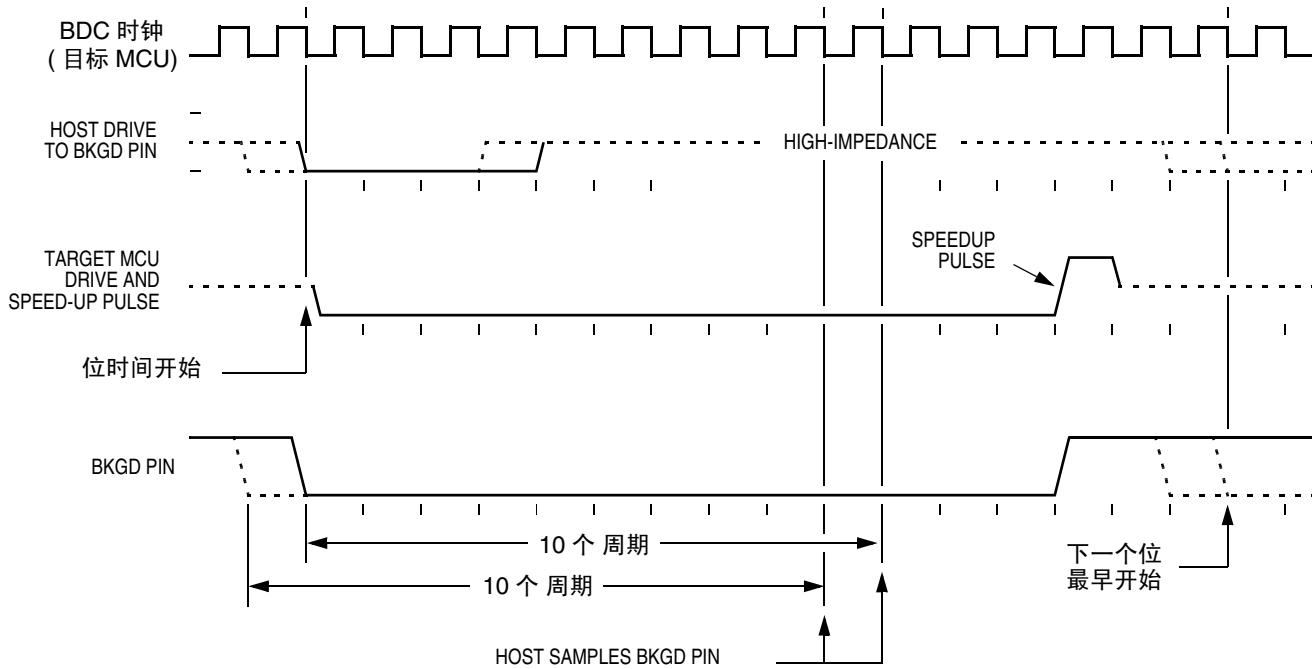


图 17-4. BDM 目标 - 到 - 主机串行位时序 (逻辑 0)

### 17.2.3 BDC 命令

BDC 命令以串行形式从主机发送到目标 HCS08 MCU 的 BKGD 管脚。所有命令和数据都采用定制 BDC 通信协议以 MSB- 先发的形式发送。激活背景调试模式命令要求目标 MCU 当前处于激活背景调试模式，而非侵入式命令可以随时发出，无论目标 MCU 是处于激活背景调试模式还是运行用户应用程序。

表 17-1 显示了所有 HCS08 BDC 命令，并简要描述了它们的编码结构，以及每个命令的含义。

#### 编码结构术语

表 17-1 中所用的术语描述了 BDC 命令的编码结构。

	命令在主机 - 到 - 目标方向上始于一个 8- 位十六进制命令代码 (MSB 先发)
/	= 将命令的各部分分开
d	= 延迟 16 个目标 BDC 时钟周期
AAAA	= 主机 - 到 - 目标方向上的一个 16- 位地址
RD	= 目标 - 到 - 主机方向上的 8- 位读数据
WD	= 主机 - 到 - 目标方向上的 8- 位写数据
RD16	= 目标 - 到 - 主机方向上的 16 位读数据
WD16	= 主机 - 到 - 目标方向上的 16 位写数据
SS	= 目标 - 到 - 主机方向 (STATUS) 上的 BDCSCR 内容
CC	= 主机 - 到 - 目标方向 (CONTROL) 方向上的 8 位写数据
RBKP	= 目标 - 到 - 主机方向 (从 BDCBKPT 断点寄存器) 上的 16 位读数据
WBKP	= 主机 - 到 - 目标方向 (至 BDCBKPT 断点寄存器) 16 位写数据

表 17-1. BDC 命令一览

命令术语	激活 BDM/ 非侵入式	编码结构	描述
SYNC	非侵入式	n/a <sup>1</sup>	要求定时参考脉冲来确定目标 BDC 通信速率
ACK_ENABLE	非侵入式	D5/d	使能响应协议。参见飞思卡尔文档编号 HCS08RMv1/D。
ACK_DISABLE	非侵入式	D6/d	禁止响应协议。参见飞思卡尔文档编号 HCS08RMv1/D。
BACKGROUND	非侵入式	90/d	如果使能，则进入激活背景调试模式（如果 ENBDM 位等于 0，则忽略）
READ_STATUS	非侵入式	E4/SS	从 BDCSCR 读取 BDC 状态
WRITE_CONTROL	非侵入式	C4/CC	向 BDCSCR 中写入对 BDC 的控制
READ_BYTE	非侵入式	E0/AAAA/d/RD	从目标存储器读取字节
READ_BYTE_WS	非侵入式	E1/AAAA/d/SS/RD	读字节和报告状态
READ_LAST	非侵入式	E8/SS/RD	从地址重新读字节，仅读和报告状态
WRITE_BYTE	非侵入式	C0/AAAA/WD/d	将字节写入到目标存储器
WRITE_BYTE_WS	非侵入式	C1/AAAA/WD/d/SS	写入字节和报告状态
READ_BKPT	非侵入式	E2/RBKP	读 BDCBKPT 断点寄存器
WRITE_BKPT	非侵入式	C2/WBKP	写 BDCBKPT 断点寄存器
GO	激活 BDM	08/d	从 PC 当前的地址执行用户应用程序
TRACE1	激活 BDM	10/d	在 PC 的地址跟踪 1 条用户指令，然后返回到激活后台模式
TAGGO	激活 BDM	18/d	与 GO 相同，但激活外部标签（HCS08 器件没有外部标签管脚）
READ_A	激活 BDM	68/d/RD	读累积器（A）
READ_CCR	激活 BDM	69/d/RD	读条件代码寄存器（CCR）
READ_PC	激活 BDM	6B/d/RD16	程序计数器（PC）
READ_HX	激活 BDM	6C/d/RD16	读 H 和 X 寄存器对（H:X）
READ_SP	激活 BDM	6F/d/RD16	读堆栈指针（SP）
READ_NEXT	激活 BDM	70/d/RD	以 1 为基数递增 H:X，然后读位于 H:X 的存储器字节
READ_NEXT_WS	激活 BDM	71/d/SS/RD	以 1 为基数递增 H:X，然后读位于 H:X 的存储器字节。报告状态和数据。
WRITE_A	激活 BDM	48/WD/d	写累积器（A）
WRITE_CCR	激活 BDM	49/WD/d	写条件代码寄存器（CCR）
WRITE_PC	激活 BDM	4B/WD16/d	写程序计数器（PC）
WRITE_HX	激活 BDM	4C/WD16/d	写 H 和 X 寄存器对（H:X）
WRITE_SP	激活 BDM	4F/WD16/d	写堆栈指针（SP）
WRITE_NEXT	激活 BDM	50/WD/d	以 1 为基数递增 H:X，然后写位于 H:X 的存储器字节
WRITE_NEXT_WS	激活 BDM	51/WD/d/SS	以 1 为基数递增 H:X，然后写位于 H:X 的存储器字节。报告状态和数据。1 SYNC 命令是特殊操作，不需要命令代码。

<sup>1</sup> SYNC 命令是特殊操作，不需要命令代码。

**SYNC** 命令与其它 **BDC** 命令不同，因为主机不必要知道 **BDC** 通信的正确通信速率，直到它分析完 **SYNC** 命令的响应后。

要发出 **SYNC** 命令，主机：

- 保持 **BKGD** 管脚为低电平至少 128 周期，而且是以最慢的 **BDC** 时钟来计(最慢的时钟一般是参考振荡器 /64 或自时钟速率 /64。)
- 驱动 **BKGD** 达到高电平，实现瞬态加速，快速上升时间 (这个加速脉冲一般是系统中最快的时钟的一个周期)
- 去除 **BKGD** 管脚的所有驱动，这样它可回复到高阻抗。
- 监视 **BKGD** 管脚得到同步响应脉冲

当检测到主机的 **SYNC** 请求（比在正常 **BDC** 通信过程中发生的慢时钟要长），则目标：

- 等待 **BKGD** 返回到逻辑高电平
- 延迟 16 个周期，允许主机停止驱动高电平加速脉冲
- 驱动 **BKGD** 低态 128 **BDC** 时钟周期
- 驱动一个周期的高电平加速脉冲，在 **BKGD** 上实现快速上升时间
- 去除 **BKGD** 管脚的所有驱动，这样它可回复到高阻抗。

主机测量这个 128 周期的响应脉冲的低电平时间，判断速率，进行后续的 **BDC** 通信。主机一般可以确定正确的通信速率，与实际目标速率的误差只有百分之几，通信协议能够接受百分之几的速率误差。

#### 17.2.4 BDC 硬件断点

**BDC** 包括一个相对简单的硬件断点，将 **CPU** 地址总线与 **BDCBKPT** 寄存器中的 16- 位匹配值进行比较。这个断点可以生成强制断点或标记断点。强制断点使 **CPU** 在存取断点地址后的第一个指令边界进入激活背景调试模式。标记的断点使指令操作码在断点地址被标记，这样当 **CPU** 到达指令队列的终点时，将进入激活后台模式，而不是执行该指令。这意味着标记的断点可能放置在指令操作代码的地址上，而强制断点可以设置在任何地址。

**BDC** 状态和控制寄存器 (**BDCSCR**) 中的断点使能 (**BKPTEN**) 控制位用来激活断点逻辑 (**BKPTEN = 1**)。当 **BKPTEN = 0** (复位后它的默认值)，断点逻辑禁止，无论其它 **BDC** 断点中的值是多少，也不管控制位如何，均不请求断点。**BDCSCR** 中的强制 / 标记选择 (**FTS**) 控制位用来选择强制 (**FTS = 1**) 或标记 (**FTS = 0**) 类型断点。

片上调试模块 (**DBG**) 包括两个额外的硬件断点的电路，这两个硬件断点比 **BDC** 模块中的简单断点更灵活。

## 17.3 片上调试系统 (DBG)

由于 HCS08 器件没有外部地址和数据总线，在线仿真器最重要的功能已经构建在 MCU 的芯片上。这种调试系统包含可以灵活地存储地址或数据总线信息的 8- 级 FIFO，和一个确定何时捕获总线信息以及捕获哪些总线信息的灵活触发系统。这个系统依赖单线背景调试系统来存取调试控制寄存器，读取 8 级阶 FIFO 的结果。

调试模块包括控制和状态寄存器，可以在用户存储器映射中存取。这些寄存器位于高地址寄存器空间中，避免使用宝贵的直接页面存储器空间。

大多数调试模块的功能在开发过程使用，用户程序很少存取调试模块的任何控制和状态寄存器。一个例外就是调试系统可以提供一种手段来实施某种形式的 ROM 补丁。[17.3.6，“硬件断点”](#) 中对此有更详细的描述。

### 17.3.1 比较器 A 和 B

两个 16- 位比较器器 (A 和 B) 可以选择用 R/W 信号或一个操作码跟踪电路来鉴定。比较器单独的控制位允许你忽略每个比较器的 R/W。操作码跟踪电路可选地允许你规定，如果操作码在规定的地址实际执行，而不是只从存储器读到指令队列中，则触发将发生。比较器还能够进行庞大的比较，支持范围内和范围外触发模式。在所有 BDC 存取过程中，比较器临时禁止。

比较器 A 总是与 16- 位 CPU 地址相关联。比较器 B 根据所选的触发模式比较 CPU 地址或 8- 位 CPU 数据总线。由于 CPU 数据总线分为只读数据和写数据总线，RWAEN 和 RWA 控制位有一个额外的目的，在完整地址加上数据比较中，它们被用来确定其中哪些总线用在比较器 B 数据总线比较中。如 RWAEN = 1 (激活)，RWA = 0 (写)，则使用 CPU 的写数据总线，否则用 CPU 的只读数据。

当前选择触发模式确定当比较器检测到合格的匹配条件时，调试器逻辑做什么。匹配可以导致以下情况：

- 生成 CPU 断点
- 将数据总线值存储到 FIFO 中
- 开始将流变化地址存储到 FIFO 中 (开始类型跟踪)
- 停止将流变化地址存储到 FIFO 中 (结束类型跟踪)

### 17.3.2 总线捕获信息和 FIFO 操作

使用 FIFO 的通常方式是建立触发模式和其它控制选项，然后打开调试器。当 FIFO 填满后，或调试器停止将数据存储到 FIFO 后，你可以按信息存储的顺序从中读取信息。状态位指示数据所在的 FIFO 中的有效信息的字数。如果在满 (CNT = 1:0:0:0) 之前将 ARM 写为 0，以人工停止跟踪，信息移动一个位置，主机必须执行 ((8 - CNT) - 1)FIFO 虚读操作，使信息进入到 FIFO 中的第一个重要入口。

在大多数触发模式中，存储在 FIFO 中的信息包含 16- 位流变化地址。在这些情况中，先读 DBGFH 然后读 DBGFL，从 FIFO 中获得一个一致的信息字。读 DBGFL (FIFO 数据端口的低阶字节) 会使 FIFO 移动，这样下一个信息字可以在 FIFO 数据端口提供。在纯事件触发模式 (参见 [17.3.5，“触发模式”](#)) 中，8 位数据信息存储在 FIFO 中。在这些情况下，FIFO(DBGFH) 的

上半部分没有被使用，仅仅通过读 **DBGFL** 来从 **FIFO** 中读出数据。每次读 **DBGFL** 时，**FIFO** 都会移动，这样通过 **DBGFL** 的 **FIFO** 数据端口可以获得下一个数据值。

在触发模式中，**FIFO** 保存流变化地址，**CPU** 地址与 **FIFO** 的输入端有一个延迟。由于这个延迟，如果触发事件本身是一个流变化地址或在触发事件启动 **FIFO** 后下两个周期中出现了流变化地址，它将不保存在 **FIFO** 中。如果是结束 - 跟踪的情况，当触发事件是一个流变化，则它将保存为运行的调试器的最后一个流变化入口。

当调试器没有打开时，**FIFO** 还可以用来生成所执行指令地址的分析。当 **ARM = 0**，读 **DBGFL** 会使最近获取的操作码的地址保存在 **FIFO** 中。采用分析功能，主机调试器将从 **FIFO** 中读取地址，即以常规的间隔先读 **DBGFH** 然后读 **DBGFL**。前 8 个值将被丢弃，因为它们对应于初始需要填充 **FIFO** 的 8 个 **DBGFL** 读取。**DBGFH** 和 **DBGFL** 的其它周期读取则返回关于所执行指令的延迟信息，这样主机调试器可以对执行指令地址进行分析。

### 17.3.3 流变化信息

为了减少存储在 **FIFO** 中的信息数量，只保存与使正常的指令执行顺序发生变化的指令相关的信息。知道存储在目标系统中的源和对象代码程序后，外部调试器可以通过来自 **FIFO** 中存储的大量流变化信息的许多指令来重现执行路径。

对于采用了分支的条件分支指令（分支条件为真），则保存源地址（条件分支操作码的地址）。由于 **BRA** 和 **BRN** 指令不是条件的，这些事件不会使流变化信息存储在 **FIFO** 中。

间接 **JMP** 和 **JSR** 指令采用 **H:X** 间址寄存器对的当前内容，确定目的地址，这样调试系统为任何间接 **JMP** 或 **JSR** 保存运行时的目的地址。对于中断，**RTI** 或 **RTS**，目的地址作为流变化信息存储在 **FIFO** 中。

### 17.3.4 标记 vs. 强制断点和触发器

标记一词指当指令操作码被取到指令队列时识别它，但是不采取任何其它操作，直到且除非指令被 **CPU** 真正执行。这种区分非常重要，因为任何因跳转、分支、子例程调用、或中断而发生的流变化都会导致一些指令被取到指令队列，未执行就被丢弃。

强制类型的断点等待当前指令完成，然后执行断点请求操作。通常操作是进入激活背景调试模式，而不是继续用户应用程序中的下一个指令。

标记 vs. 强制这一术语在调试模块的两种情况下使用。第一种情况指从调试模块向 **CPU** 发送断点请求。第二种情况指从比较器向调试控制逻辑发送匹配信号。当标记类断点发送给 **CPU** 时，信号与操作码一起进入指令队列，这样当这个操作码被执行时，**CPU** 将有效地用 **BGND** 操作码代替被标记的操作码，这样 **CPU** 进入激活背景调试模式，而不是执行被标记的指令。当 **DBGTR** 寄存器中的 **TRGSEL** 控制位被设置为选择标记类操作，比较器 A 或 B 的输出被调试模块中的逻辑块鉴定，这个逻辑块跟踪操作码，如果比较地址的操作码被实际执行，则只向该调试器生成一个触发。每个比较器都有单独的操作码跟踪逻辑，这样整个指令队列一次不只一个比较事件被跟踪。

### 17.3.5 触发模式

触发模式控制调试器运行的整体行为。DBGT 寄存器中的 4-位 TRG 字段选择九个触发模块中的一个。当 DBGT 寄存器中的 TRGSEL = 1，比较器的输出必须在触发 FIFO 操作前通过操作码跟踪电路传播。DBGT 中的 BEGIN 位选择当检测到合格的触发时 FIFO 是否开始存储数据（开始跟踪），或 FIFO 从其打开之时开始循环存储数据，直到检测到合格的触发（结束触发）。

将 1 写入到寄存器中的 ARM 位便可启动调试运行，它设置 DBGS 中的 ARMF 标记，并清除 AF 和 BF 标记及 CNT 位。开始跟踪调试运行当 FIFO 满时结束。结束跟踪运行则在所选触发事件发生时结束。任何调试运行均可通过将 0 写入到 DBGC 中的 ARM 或 DBGEN 位停止。

除纯事件模式外的所有触发模式中，FIFO 都存储流变化地址。在纯事件触发模式中，FIFO 将数据存储在 FIFO 的八低八位。

控制位在纯事件触发模式中被忽略，而且所有这样的调试运行都是开始类型跟踪。当 TRGSEL = 1 选择操作码获取触发器，没有必要在比较中使用 R/W，因为操作码标签只应用于操作码获取，而这一直都是读周期。在采用全模式触发器时，规定 TRGSEL = 1 也是不正常的，因为操作码的值通常在特定的地址可以知道。

下面的触发模式描述只说明了导致触发的主要比较器条件。比较器 A 或 B 通常都可以被 R/W 进一步鉴定，通过将 RWAEN (RWBEN) 和相应的 RWA (RWB) 值设置为与 R/W 相匹配。如果 BRKEN = 1，来自比较器的带可选 R/W 鉴定的信号，用来请求 CPU 断点，TAG 决定 CPU 请求是标记请求还是强制请求。

**只 A — 当地址匹配比较器 A 的值时触发**

**A 或 B — 当地址匹配比较器 A 或 B 的值时触发**

**A 然后 B — 当地址匹配比较器 B 但只能在另一个周期的地址匹配比较器 A 的值以后，触发。可能在 A 匹配后 B 匹配前有许多周期。**

**A 和 B 数据（全模式）—**这称为全模式，因为地址，数据和 R/W（可选）必须在同一个总线周期内匹配，才能产生触发事件。比较器 A 检查地址，比较器的低阶字节检查数据，如果 RWAEN = 1，R/W 对照 RWA 进行检查。比较器 B 的高半部分没有使用。

在全触发模式中，规定标签类 CPU 断点 (BRKEN = TAG = 1) 没有用，但是如果你这样做了，就会忽略比较器 B 数据匹配，以例向 CPU 发送标签请求，当比较器 A 地址匹配时发送 CPU 断点。

**A 但非 B 数据（全模式）—**地址必须匹配比较器 A，数据必须不能匹配比较器 B 的低阶部分，如果 RWAEN = 1，R/W 必须匹配 RWA。所有三个条件必须在同一个总线周期中达到才能引起触发。

在全触发模式中，规定标签类 CPU 断点 (BRKEN = TAG = 1) 没有用，但是如果你这样做了，就会忽略比较器 B 数据匹配，以例向 CPU 发送标签请求，当比较器 A 地址匹配时发送 CPU 断点。

**纯事件 B（存储数据）—**当地址每次匹配比较器 B 的值时，触发事件发生。触发事件导致数据被捕获到 FIFO 中。当 FIFO 满时调试运行结束。

**A 然后纯事件 B（存储数据）—**当地址匹配比较器 A 中的值后，每次地址匹配比较器 B 中的值时，触发事件发生。触发事件导致数据被捕获到 FIFO 中。当 FIFO 满时调试运行结束。

**范围内 ( $A \leq 地址 \leq B$ )** — 当地址大于或等于比较器 A 的值，且小于等于比较器 B 的值时，触发发生。

**范围外 (地址 < A 或 地址 > B)** — 当地址小于比较器 A 的值，或大于比较器 B 的值时，触发发生。

### 17.3.6 硬件断点

DBGC 寄存器中的 BRKEN 控制位可以设置为 1，来允许使用 17.3.5，“触发模式”所描述的任何触发条件，向 CPU 生成硬件断点请求。DBGC 中的 TAG 控制断点请求是否处理为标记类断点或强制类断点。标记断点使当前的操作码进入指令队列时被标记。如果标记的操作码达到队列的末端，CPU 执行 BGND 指令，进入激活背景调试模式，而不是执行被标记的操作码。强制类断点使 CPU 完成当前指令，然后进入激活背景调试模式。

如果背景调试模式没有被通过 BKGD 管脚的串行 WRITE\_CONTROL 命令激活 (ENBDM = 1)，CPU 将执行 SWI 指令，而不是进入激活背景调试模式。

## 17.4 寄存器定义

本小节描述了 BDC 和 DBG 寄存器及控制位。

参见本文的器件概述章节中的 high-page 寄存器一览，了解所有 DBG 寄存器的绝对地址分配。本小节只按名字参考了寄存器和控制位。使用飞思卡尔提供的等式或头文件，将这些名称翻译为相应的绝对地址。

### 17.4.1 BDC 寄存器和控制位

BDC 有两个寄存器：

- 状态和控制寄存器 (BDCSCR) 是一个包含背景调试控制器控制和状态位的 8- 位寄存器。
- BDC 断点匹配寄存器 (BDCBKPT) 拥有一个 16- 位断点匹配地址。

这些寄存器通过专门的串行 BDC 命令接入，没有位于目标 MCU 的存储器空间中（因此，它们没有地址，用户程序不能接入）。

BDCSCR 中的一些位有写限制，否则这些寄存器可以随时被读或写。例如，当 MCU 处于激活背景调试模式中时，ENBDM 控制位不能被写。（这防止了在 MCU 已经处于激活后台模式时，禁止激活后台模式的控制位的模糊条件）而且，有四个状态位 (BDMACT, WS, WSF, 和 DVF) 是只读状态指示符，永远也不能被 WRITE\_CONTROL 串行 BDC 命令写。时钟开关 (CLKSW) 控制位随时都可读或写。

#### 17.4.1.1 BDC 状态和控制寄存器 (BDCSCR)

这个寄存器可以被串行 BDC 命令 (READ\_STATUS 和 WRITE\_CONTROL) 读或写，但是用户程序不能存取它，因为它不位于 MCU 的正常的存储器映射空间中。

	7 R W	6 ENBDM	BDMACT	5 BKPTEN	4 FTS	3 CLKSW	2 WS	1 WSF	0 DVF
正常复位		0	0	0	0	0	0	0	0
在激活 BDM 中复位		1	1	0	0	1	0	0	0

= 未实施或预留

图 17-5. BDC 状态和控制寄存器 (BDCSCR)

表 17-2. BDCSCR 寄存器字段描述

字段	描述
7 ENBDM	<b>激活 BDM (允许激活背景调试模式)</b> — 一般而言，这个位在调试开始后不久，或只要调试主机复位目标，由调试主机写为 1，并保留 1，直到通过正常的复位清除它。 0 BDM 不能激活（非侵入式命令仍然被允许） 1 BDM 可以激活，允许激活后台模式命令
6 BDMACT	<b>背景调试模式激活状态</b> — 这是只读状态位。 0 BDM 未激活（用户应用程序运行） 1 BDM 激活并等待串行命令
5 BKPTEN	<b>BDC 断点激活</b> — 如果这个位清零，BDC 断点处于处活状态，FTS（强制标签选择）控制位和 BDCBKPT 匹配寄存器被忽略。 0 BDC 断点禁止 1 BDC 断点激活
4 FTS	<b>强制 / 标签选择</b> — 当 FTS = 1，只要 CPU 地址总线匹配 BDCBKPT 匹配寄存器，则请求断点。当 FTS = 0，CPU 地址总线与 BDCBKPT 寄存器之间的匹配会造成获取的操作码被标记。如果标记的操作码到达指令队列的末端，CPU 则进入激活后台模式，而不是执行标记的操作码。 0 在断点地址标记操作码，如果 CPU 试图执行该指令，则进入激活后台模式 1 断点匹配强制在下一个指令边界进入激活后台模式（地址不必是操作码）
3 CLKSW	<b>选择 BDC 通信时钟的源</b> — CLKSW 默认 0，选择其它 BDC 时钟源。 0 其它 BDC 时钟源 1 MCU 总线时钟表
2 WS	<b>等待或停止状态</b> — 当目标 CPU 处于等待或停止状态时，大多数 BDC 命令不起作用。但是可以用后台命令来强制目标 CPU 从等待或停止状态进入激活后台模式，这样所有 BDC 命令都可以起作用。只要主机强制目标 MCU 进入激活背景调试模式，主机应该发出 READ_STATUS 命令，在尝试其它 BDC 命令前，检查 BDMACT = 1。 0 目标 CPU 运行用户应用代码，或处于激活背景调试模式（当后台激活时，它不处于等待或停止模式） 1 目标 CPU 处于等待或停止模式，或者后台命令用来将其从等待或停止状态改变为激活背景调试模式
1 WSF	<b>等待或停止失败状态</b> — 如果这存储器存取命令因目标 CPU 在大约相同时间执行等待或停止指令而失败，则设置这个状态位。通常的恢复策略是发出后台命令，从等待或停止模式进入激活后台模式，重复失败的命令，然后返回到用户程序。（一般地，主机应该恢复 CPU 寄存器，准备值，重新执行等待或停止指令。） 0 存储器存取与等待或停止指令不冲突 1 存储器存取命令失败，因为 CPU 已进入等待或停止模式
0 DVF	<b>数据有效失败状态</b> — 这个状态位没有在 MC9S08DZ60 系列中使用，因为它没有慢存取存储器。 0 存储器存取与慢存储器接入不冲突 1 存储器存取命令失败，因为 CPU 没有完成慢存储器接入

### 17.4.1.2 BDC 断点匹配寄存器 (BDCBKPT)

6- 位寄存器保留 BDC 中的硬件断点的地址。BDCSCR 中的 BKPTEN 和 FTS 控制位用来使能和配置断点逻辑。专门的串行 BDC 命令 (READ\_BKPT 和 WRITE\_BKPT) 用来读和写 BDCBKPT 寄存器，但是用户程序不能存取它，因为它不位于 MCU 的普通存储器映射空间中。当目标 MCU 处于激活背景调试模式时，断点一般在运行用户应用程序前设置。关于建立和使用 BDC 中的硬件断点逻辑的更多信息，请参见 17.2.4，“BDC 硬件断点”。

### 17.4.2 系统背景调试强制复位寄存器 (SBDFR)

这个寄存器包含单个只写控制位。必须要用一个串行后台模式命令，如 WRITE\_BYTE，来写 SBDFR。从用户程序写该寄存器的尝试被忽略。读总是返回 0x00。

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0
W								BDFR <sup>1</sup>
复位	0	0	0	0	0	0	0	0
= 未实施或预留								

<sup>1</sup> BDFR 只有通过串行后台模式调试命令才可写，不能通过用户程序来写。

图 17-6. 系统背景调试强制复位寄存器 (SBDFR)

表 17-3. 寄存器字段描述

字段	描述
0 BDFR	背景调试强制复位 —— 一系列激活后台模式命令，如 WRITE_BYTE 等，允许外部调试主机强制目标系统复位。将 1 写到这个位，强制 MCU 复位。这个位 不能从用户程序写。

### 17.4.3 DBG 寄存器和控制位

这个调试模块包括 9 个字节的寄存器空间，用于三个 16- 位寄存器和三个 8- 位控制和状态寄存器。这些寄存器位于存储器空间的高地址空间中，这样它们可以存取正常的应用程序。普通用户应用程序几乎从不接入这些寄存器，除了使用断点逻辑的 ROM patching 机制。

#### 17.4.3.1 调试比较器 A 高寄存器 (DBGCAH)

这个寄存器包含比较器 A 的高 8 位的比较值位。在复位时，这个寄存器被强制设置为 0x00，可以随时被读或写，除非 ARM = 1。

#### 17.4.3.2 调试比较器 A 低寄存器 (DBGCAL)

这个寄存器包含比较器 A 的低 8 位的比较值位。在复位时，这个寄存器被强制设置为 0x00，可以随时被读或写，除非 ARM = 1。

### 17.4.3.3 调试比较器 B 高寄存器 (DBGCBH)

这个寄存器包含比较器 B 的高 8 位的比较值位。在复位时，这个寄存器被强制设置为 0x00，可以随时被读或写，除非 ARM = 1。

### 17.4.3.4 调试比较器 B 低寄存器 (DBGCBL)

这个寄存器包含比较器 B 的低 8 位的比较值位。在复位时，这个寄存器被强制设置为 0x00，可以随时被读或写，除非 ARM = 1。

### 17.4.3.5 调试 FIFO 高寄存器 (DBGFH)

这个寄存器提供对 FIFO 的高 8 位的只读接入。写到这个寄存器没有意义或无效果。在纯事件触发模式中，FIFO 只将数据存储在每个 FIFO 字的低字节，因此这个寄存器不能使用，将读 0x00。

读 DBGFH 不会导致 FIFO 移动到下一个字。当从 FIFO 中读出 16- 位字时，在读 DBGFL 前先读 DBGFH，因为读 DBGFL 会导致 FIFO 先于下个字的信息。

### 17.4.3.6 调试 FIFO 低寄存器 (DBGFL)

这个寄存器提供对 FIFO 的低 8 位的只读存取。写到这个寄存器没有意义或无效果。

读 DBGFL 会导致 FIFO 移动到下一个字的信息。当调试模块以纯事件模式运行时，只有 8- 位数据存储在 FIFO（每个 FIFO 字的高字节部分没有使用）。当从 FIFO 中读出 8- 位字时，只需重复地读 BDGFL，从 FIFO 中获得数据的连续的字节。在这种情况下，没有必要读 DBGFH。

当 FIFO 仍然打开时（打开后，但 FIFO 充满或 ARMF 被清除前）不要试图从其中读数据，因为在 DBGL 读取过程中，FIFO 不能进一步操作。这可以干扰正常的 FIFO 的读取顺序。

在调试器没有打开的情况下读会使最近获取的操作码的地址存储到 FIFO 中的最后的位置。读取 DBGFL，然后定期 DBGFL，外部主机软件可以开发程序执行的概况。在对 FIFO 进行八次读取后，第九次读取将返回第一次读取结果的信息。要使用分析功能，则需要读取 FIFO 八次，且不使用启动顺序的数据，然后开始使用数据来获取已执行地址的延迟概貌。存储在 FIFO 中的关于 DBGFL（且 FIFO 没有打开）读取的信息就是最近所获操作码的地址。

### 17.4.3.7 调试控制寄存器 (DBG C)

这个寄存器可以在任何时间读或写。

	7	6	5	4	3	2	1	0
R	DBGEN	ARM	TAG	BRKEN	RWA	RWAEN	RWB	RWBEN
复位	0	0	0	0	0	0	0	0

图 17-7. 调试控制寄存器 (DBG C)

表 17-4. DBG C 寄存器字段描述

字段	描述
7 DBGEN	调试模块启用 — 来用启用调试模块。 DBGEN 不能设置为 1, 如果 MCU 是安全的。 0 DBG 禁用 1 DBG 启用
6 ARM	打开控制 — 控制调试器是否在 FIFO 中比较和存储信息。采用写操作来设置该位 ( 和 ARMF), 完成调试运行就是自动清除它。将 ARM 或 DBGEN 写为 0, 可以停止任何调试运行。 0 调试器没有打开 1 调试器被打开
5 TAG	标记 / 强制选择 — 控制送到 CPU 的中断请求是否为标签或强制型请求。如果 BRKEN = 0, 这个位就没有意义或无效。 0 CPU 中断请求作为强制型请求 1 CPU 中断请求作为标签型请求
4 BRKEN	中断启用 — 控制触发事件是否向 CPU 生成中断请求。触发事件可以使信息存储在 FIFO 中而不必向 CP 生成中断请求。对于结束跟踪, 如果比较器 (s) 和 R/W 满足触发条件, 则发出 CPU 中断请求。对于起始跟踪, 则当 FIFO 满时发出 CPU 中断请求。 TRGSEL 不影响 CPU 中断请求的定时。 0 CPU 不断请求未启用 1 触发器触发向 CPU 发出中断请求
3 RWA	比较器 A 的 R/W 比较值 — 当 RWAEN = 1, 这个位确定是否用读或写接入来鉴定比较器 A, 当 RWAEN = 0, , RWA 和 R/W 信号不影响比较器 A。 0 比较器 A 只在写周期上匹配 1 比较器 A 只在读周期上匹配
2 RWAEN	启用比较器 A 的 R/W — 控制比较器 A 的匹配是否考虑这个水平的 R/W 。 0 R/W 未用在比较 A 中 1 R/W 用在比较 A 中
1 RWB	比较器 B 的 R/W 比较值 — 当 RWBEN = 1, 这个位确定是否用读或写接入来鉴定比较器 B。当 RWBEN = 0, RWA 和 R/W 信号不影响比较器 B。 0 比较器 B 只在写周期上匹配 1 比较器 B 只在读周期上匹配
0 RWBEN	启用比较器 B 的 R/W B — 控制比较器 B 的匹配是否考虑这个水平的 R/W 。 0 R/W 未用在比较 B 中 1 R/W 用在比较 B 中

### 17.4.3.8 调试触发寄存器 (DBGTR)

这个寄存器在任何时候都可以读，但是只有当 ARM = 0 时才可以写，除非位 4 和位 5 硬件线与至 0。

	7	6	5	4		3	2	1	0
R	TRGSEL	BEGIN	0	0		TRG3	TRG2	TRG1	TRG0
W									
复位	0	0	0	0		0	0	0	0
	= 未实施或预留								

图 17-8. 调试触发寄存器 (DBGTR)

表 17-5. DBGTR 寄存器字段描述

字段	描述
7 TRGSEL	触发类型 — 控制比较器 A 和 B 的匹配输入是否与调试模块中的操作码跟踪逻辑匹配。如果 TRGSEL 已设置，比较器 A 或 B 的匹配信号必须通过操作码跟踪逻辑传播，如果匹配地址的操作码实际已执行，则只有触发事件发送到 FIFO 逻辑。 0 存取比较地址时触发 ( 强制 ) 1 如果比较地址的操作码已执行 ( 标签 )，则触发
6 BEGIN	开始 / 结束触发选择 — 控制 FIFO 在触发时开始填充还是以循环形式填充直到触发结束信息的捕获。在纯事件触发模式中，忽略这个位，所有调试运用都假定为起始跟踪。 0 数据存储在 FIFO，直到触发 ( 结束跟踪 ) 1 触发启动数据存储 ( 起始跟踪 )
3:0 TRG[3:0]	选择触发模式 — 选择下面 9 个触发模式中的一个。 0000 只有 A 0001 A 或 B 0010 A 然后 B 0011 只有事件 B ( 存储数据 ) 0100 A 然后只有事件 B ( 存储数据 ) 0101 A 和 B 数据 ( 满模式 ) 0110 A 和非 B 数据 ( 满模式 ) 0111 I 范围内：A ≤ 地址 ≤ B 1000 范围外：地址 < A 或 地址 > B 1001 – 1111 ( 无触发 )

### 17.4.3.9 调试状态寄存器 (DBGS)

这是一个只读状态寄存器。

	7	6	5	4		3	2	1	0
R	AF	BF	ARMF	0	CNT3	CNT2	CNT1	CNT0	
W									
复位	0	0	0	0		0	0	0	0
	= 未实施或预留								

图 17-9. 调试状态寄存器 (DBGS)

表 17-6. DBGS 寄存器字段描述

字段	描述
7 AF	<b>触发匹配 A 标记</b> — 在调试运行开始时清除 AF，指示武装后是否满足触发 匹配 A 条件。 0 比较器 A 未匹配 1 比较器 A 匹配
6 BF	<b>触发匹配 B 标记</b> — 在调试运行开始时清除 BF，指示武装后是否满足触发 匹配 B 条件。 0 比较器 B 未匹配 1 比较器 B 匹配
5 ARMF	<b>打开标记</b> — 当 DBGEN=1 时，这个位为 DBG_C 中 ARM 的只读镜像。将 DBG_C 中的 ARM 控制位写为 1（当 DBGEN = 1）可设置该位，在调试运行结束时自动清除它。当 FIFO 为满时（始起跟踪），或当探测到触发事件时（结束跟踪），调度运行完成。将 DBG_C 中的 ARM 或 DBGEN 写为 0，可以人工停止调试运行。 0 调试器没有打开 1 调试器被打开
3:0 CNT[3:0]	<b>FIFO 有效计数</b> — 这些位在调试运行开始时清除，指示调试运行结束时 FIFO 中的有效数据的字数。当数据从 FIFO 中读出时，CNT 中的值不减少。当信息从 FIFO 中读出时，外部调试主机负责计数的跟踪。 0000 FIFO 中的有效字数 = 无有效数据 0001 FIFO 中的有效字数 = 1 0010 FIFO 中的有效字数 = 2 0011 FIFO 中的有效字数 = 3 0100 FIFO 中的有效字数 = 4 0101 FIFO 中的有效字数 = 5 0110 FIFO 中的有效字数 = 6 0111 FIFO 中的有效字数 = 7 1000 FIFO 中的有效字数 = 8



# 附录 A 电气特征

## A.1 简介

本小节包含本文档出版时 MC9S08DZ60 系列微控制器最精确的电气和时序信息。

## A.2 参数分类

本附件中显示的电气参数通过各种方法得到了保证。为了让客户更好地理解，我们进行了如下分类，并在合适的地方对表中的参数进行了相应地标注：

表 A-1. 参数分类

P	在各个器件上进行生产测试时这些参数都得到了保证。
C	这些参数通过设计特性表征来实现的，方法是从统计上测量整个工艺参数变化的相关样本尺寸。
T	这些参数通过对典型条件下的典型设备（除非另有说明）进行小规模采样进行设计特性表征而获得。典型列中显示的所有值均属本类别。
D	这些参数主要源自于仿真。

### 注意

参数表“C”栏中显示了相应的分类。

## A.3 绝对最大额定值

绝对最大额定值仅仅强调额定值，不保证最大值的功能操作。超过表 A-2 中指定限度的压力可能会影响设备可靠性或造成设备的永久损坏。有关功能操作条件的信息，请参考本小节中的其它表格。

本芯片含有防高静电压或电场的电路，保护设备免遭损坏。然而，我们还是建议采取一些正常的防范措施，以避免任何高于最大额定电压的电压应用到该高阻电路。如果未使用的输入固定到一个适当的逻辑电压水平（例如， $V_{SS}$  或  $V_{DD}$ ）。

表 A-2. 绝对最大额定值

编号	参数	符号	值	单位
1	电源电压	$V_{DD}$	-0.3 to + 5.8	V
2	输入电压	$V_{In}$	-0.3 to $V_{DD} + 0.3$	V
3	瞬时最大电流 单管脚极限 (适用于所有端口管脚) <sup>1, 2, 3</sup>	$I_D$	$\pm 25$	mA
4	$V_{DD}$ 中的最大电流	$I_{DD}$	120	mA
5	存储温度	$T_{stg}$	-55 to +150	°C

<sup>1</sup> 输入必须是限定为指定值的电流。要确定所需的电流限流电阻器的值，需要先计算正 ( $V_{DD}$  和负 ( $V_{SS}$ ) 钳位电压的电阻值，然后使用两个电阻值中的较大者。

<sup>2</sup> 所有功能性非电源管脚内部均钳位在  $V_{SS}$  和  $V_{DD}$ 。

<sup>3</sup> 在瞬时和操作最大电流条件下，电源必须维持在操作  $V_{DD}$  范围内。如果正注入电流 ( $V_{In} > V_{DD}$ ) 大于  $I_{DD}$ ，则注入电流就可能超出  $V_{DD}$ ，并导致外部电源不可调控。确保外部  $V_{DD}$  载荷分流大于最大注入电流的电流。当 MCU 不消耗功率时，就会有最大的风险，这样的例子包括：如果当前无系统时钟，或者如果时钟速率非常低，这都会降低总功耗。

## A.4 热特性

本小节提供有关操作温度范围、功耗和封装热阻的信息。 $I/O$  管脚上的功耗一般要比片上逻辑的功耗小，它由用户自己决定而非受 MCU 设计的控制。为了在功率计算中把  $P_{I/O}$  考虑进去，先需要确定实际管脚电压和  $V_{SS}$  or  $V_{DD}$  间的差，并乘以每个  $I/O$  管脚的管脚电流。除非出现异常高的管脚电流（大负载），管脚电压和  $V_{SS}$  or  $V_{DD}$  间的差非常小。

表 A-3. 热特征

编号	C	参数	符号	值	单位	温度代码
1	D	操作温度范围 (打包后)	$T_A$	-40 至 125 -40 至 105 -40 至 85	°C	M V C
2	T	最高结温度 <sup>1</sup>	$T_J$	135	°C	—
3	D	热阻 <sup>2</sup>				
		单层板				
		64- 管脚 LQFP	$\theta_{JA}$	69	°C/W	
		48- 管脚 LQFP	$\theta_{JA}$	75	°C/W	
		32- 管脚 LQFP	$\theta_{JA}$	80	°C/W	
		四层板				
		64- 管脚 LQFP	$\theta_{JA}$	51	°C/W	
		48- 管脚 LQFP	$\theta_{JA}$	51	°C/W	
		32- 管脚 LQFP	$\theta_{JA}$	52	°C/W	

<sup>1</sup> 结温度是晶元尺寸、片上功耗、封装热阻、安装点（主板）温度、周围温度、气流、主板上的其他组件功耗及主板热阻的函数。

<sup>2</sup> 结与环境的自然对流。

平均芯片接面温度 ( $T_J$ ) (单位  $^{\circ}\text{C}$ ) 可以用如下等式计算:

$$T_J = T_A + (P_D \times \theta_{JA}) \quad \text{等式 A-1}$$

其中:

$T_A$  = 周围温度,  $^{\circ}\text{C}$

$\theta_{JA}$  = 封装热阻, 结到环境,  $^{\circ}\text{C}/\text{W}$

$P_D = P_{int} + P_{I/O}$

$P_{int} = I_{DD} \times V_{DD}$ , 瓦特 — 芯片内部功率

$P_{I/O}$  = 输入和输出管脚上的功耗 — 由用户决定

对大多数应用来说,  $P_{I/O} \ll P_{int}$ , 可以忽略不计。 $P_D$  和  $T_J$  间的近似关系 (如果  $P_{I/O}$  忽略不计) 是:

$$P_D = K \div (T_J + 273^{\circ}\text{C}) \quad \text{等式 A-2}$$

将等式 1 代入和等式 2, 求出  $K$  为:

$$K = P_D \times (T_A + 273^{\circ}\text{C}) + \theta_{JA} \times (P_D)^2 \quad \text{等式 A-3}$$

其中,  $K$  是一个与特殊部件有关的常量。 $K$  可以通过测量已知  $T_A$  的  $P_D$  (平衡时), 从等式 3 计算出来。如果  $K$  值已知, 用等式 1 和等式 2 相式代入, 就可以得出任意  $T_A$  值的  $P_D$  和  $T_J$ 。

## A.5 ESD 保护和抗闭锁方法

尽管这些器件上的静电放电 (ESD) 损害要比早期的 CMOS 电路的 ESD 要小得多, 但仍应采取一些正常的处理防范措施, 防止静电。要执行一些鉴定测试, 以确保这些器件可以忍受合理水平的静电, 而不会造成任何永久的损害。

整个 ESD 测试符合 AEC-Q100 汽车集成电路的应力测试鉴定。在进行器件鉴定过程中, 要执行人体放电模式 (HBM) 和充电器件模式 (CDM) 的 ESD 应力测试。

如果暴露给 ESD 脉冲后, 器件不再符合器件规范, 那么就认定器件测试失败。在进行完高温测试后, 还要在室温下根据每个可适用的器件规范进行完整的 DC 参数及功能测试, 除非设备规范中另有指定。

表 A-4. ESD 和闭锁测试条件

模式	描述	符号	值	单位
人体	串联电阻	R1	1500	W
	存储电容	C	100	pF
	每管脚脉冲数	-	3	
闭锁	最小输入电压限制		-2.5	V
	最大输入电压限制		7.5	V

表 A-5. ESD 和闭锁保护特性

编号	参数	符号	最小值	最大值	单位
1	人体模式 (HBM)	$V_{HBM}$	+/- 2000	—	V
2	充电器件模式 (CDM)	$V_{CDM}$	+/- 500	—	V
3	$T_A = 125^\circ\text{C}$ 时的闭锁电流	$I_{LAT}$	+/- 100	—	mA

## A.6 DC 特性

本小节介绍了电源要求、I/O 管脚特性及各种操作模式中的电源电流信息。

表 A-6. DC 特性

编号	C	特性	符号	条件	最小值	典型值 <sup>1</sup>	最大值	单位
1	—	操作电压	$V_{DD}$		2.7	—	5.5	V
2	P	所有 I/O 管脚、低驱动强度 高压输出 所有 I/O 管脚、高驱动强度	$V_{OH}$	5 V, $I_{Load} = -2 \text{ mA}$	$V_{DD} - 1.5$	—	—	V
	C			3 V, $I_{Load} = -0.6 \text{ mA}$	$V_{DD} - 1.5$	—	—	
	C			5 V, $I_{Load} = -0.4 \text{ mA}$	$V_{DD} - 0.8$	—	—	
	C			3 V, $I_{Load} = -0.24 \text{ mA}$	$V_{DD} - 0.8$	—	—	
	P			5 V, $I_{Load} = -10 \text{ mA}$	$V_{DD} - 1.5$	—	—	
	C			3 V, $I_{Load} = -3 \text{ mA}$	$V_{DD} - 1.5$	—	—	
	C			5 V, $I_{Load} = -2 \text{ mA}$	$V_{DD} - 0.8$	—	—	
	C			3 V, $I_{Load} = -0.4 \text{ mA}$	$V_{DD} - 0.8$	—	—	
3	C	高电流输出 所有端口的最大总 $I_{OH}$	$I_{OHT}$	5 V	0	—	-100	mA
				3 V	0	—	-60	
4	P	所有 I/O 管脚、低驱动强度 低压输出 所有 I/O 管脚、高驱动强度	$V_{OL}$	5 V, $I_{Load} = 2 \text{ mA}$	—	—	1.5	V
	C			3 V, $I_{Load} = 0.6 \text{ mA}$	—	—	1.5	
	C			5 V, $I_{Load} = 0.4 \text{ mA}$	—	—	0.8	
	C			3 V, $I_{Load} = 0.24 \text{ mA}$	—	—	0.8	
	P			5 V, $I_{Load} = 10 \text{ mA}$	—	—	1.5	
	C			3 V, $I_{Load} = 3 \text{ mA}$	—	—	1.5	
	C			5 V, $I_{Load} = 2 \text{ mA}$	—	—	0.8	
	C			3 V, $I_{Load} = 0.4 \text{ mA}$	—	—	0.8	
5	C	低电流输出 所有端口的最大总 $I_{OL}$	$I_{OLT}$	5 V	0	—	100	mA
				3 V	0	—	60	
6	C	高压输入；所有数字输入	$V_{IH}$	5V	$0.65 \times V_{DD}$	—	—	V
7	C	低压输入；所有数字输入	$V_{IL}$	5V	—	—	$0.35 \times V_{DD}$	
8	C	输入滞后	$V_{hys}$		$0.06 \times V_{DD}$			mV
9	P	输入漏电流（每管脚） 仅针对所有输入管脚	$ I_{In} $	$V_{In} = V_{DD} \text{ or } V_{SS}$	—	0.1	1	$\mu\text{A}$

表 A-6. DC 特性 (续)

编号	C	特性	符号	条件	最小值	典型值 <sup>1</sup>	最大值	单位
10	P	Hi-Z (关态) 漏电流 (每管脚) 所有输入 / 输出	$ I_{OZ} $	$V_{IN} = V_{DD}$ or $V_{SS}$	—	0.1	1	$\mu\text{A}$
11	P	上拉电阻 (或下拉电阻 <sup>2</sup> , 如果启用的话)	$R_{PU}, R_{PD}$	5 V	20	45	65	$\text{k}\Omega$
	C			3 V	20	45	65	
12	T	输入电容、所有管脚	$C_{IN}$		—	—	8	$\text{pF}$
13	D	RAM 保持电压	$V_{RAM}$		0.9	1.4	2.0	V
14	D	POR re-arm 电压 <sup>3</sup>	$V_{POR}$		0.9	1.4	2.0	V
15	D	POR re-arm 时间 <sup>4</sup>	$t_{POR}$		10	—	—	$\mu\text{s}$
16	P	低压探测阈值 — 高量程 $V_{DD}$ 下降 $V_{DD}$ 上升	$V_{LVD1}$		3.9 4.0	4.0 4.1	4.1 4.2	V
17	P	低压探测阈值 — 低量程 $V_{DD}$ 下降 $V_{DD}$ 上升	$V_{LVD0}$		2.48 2.54	2.56 2.62	2.64 2.70	V
18	C	低压报警阈值 — 高量程 1 $V_{DD}$ 下降 $V_{DD}$ 上升	$V_{LVW3}$		4.5 4.6	4.6 4.7	4.7 4.8	V
19	P	低压报警阈值 — 高量程 0 $V_{DD}$ 下降 $V_{DD}$ 上升	$V_{LVW2}$		4.2 4.3	4.3 4.4	4.4 4.5	V
20	P	低压报警阈值 低量程 1 $V_{DD}$ 下降 $V_{DD}$ 上升	$V_{LVW1}$		2.84 2.90	2.92 2.98	3.00 3.06	V
21	C	低压报警阈值 — 低量程 0 $V_{DD}$ 下降 $V_{DD}$ 上升	$V_{LVW0}$		2.66 2.72	2.74 2.80	2.82 2.88	V
22	T	低压禁止复位 / 恢复滞后	$V_{hys}$	5 V	—	100	—	$\text{mV}$
				3 V	—	60	—	
23	D	dc 注入电流 <sup>5,6,7,8</sup> 单管脚限制 总 MCU 限制, 包括所有加应用的管脚	$I_{IC}$	$V_{IN} > V_{DD}$	0	—	2	$\text{mA}$
				$V_{IN} < V_{SS}$	0	—	-0.2	
				$V_{IN} > V_{DD}$	0	—	25	
				$V_{IN} < V_{SS}$	0	—	-5	

表 A-6. DC 特性 (续)

编号	C	特性	符号	条件	最小值	典型值 <sup>1</sup>	最大值	单位
24	C	$V_{DD} = 3.0 \text{ V}$ , Temp = 25 °C 时的带隙电压参考源	$V_{BG}$		1.19	1.20	1.21	V

- 1 典型值是温度在 25°C 时测量的。描述性，未经测试。
- 2 当配置管脚中断来探测上升沿时，就使用下拉电阻来代替上拉电阻。
- 3 最大值指保证 POR 时的最高电压。
- 4 模拟的，未测试。
- 5 在瞬时和操作最大电流条件下，电源必须维持在操作 VDD 范围内。如果正注入电流 ( $V_{In} > V_{DD}$ ) 大于  $I_{DD}$ ，注入电流就可能超出  $V_{DD}$ ，并导致外部电源不可调控。确保外部  $V_{DD}$  载荷分流大于最大注入电流的电流。当 MCU 不消耗功率时，就会有最大的风险。这样的例子包括：如果当前无系统时钟，或者如果时钟速率非常低，这都会降低总体功耗。
- 6 所有功能性非供应管脚均内部限制到  $V_{SS}$  和  $V_{DD}$ 。
- 7 输入必须是限定为指定值的电流。要确定所需的电流限定电阻器的值，需要先计算正和负钳位电压的电阻值，然后使用两个电阻值中的较大者。
- 8 PTE1 没有  $V_{DD}$  钳位二极管，不要把 PTE1 提高到  $V_{DD}$ 。

## A.7 电源电流特性

表 A-7. 电源电流特性

编号	C	参数	符号	$V_{DD}$ (V)	典型值 <sup>1</sup>	最大值 <sup>2</sup>	单位
1	C	当 (CPU 时钟 = 2 MHz、 $f_{Bus} = 1 \text{ MHz}$ ) 时测量的电源电流 <sup>3</sup>	$RI_{DD}$	5	3	7.5	mA
	C			3	2.8	7.4	
2	P	当 (CPU 时钟 = 16 MHz、 $f_{Bus} = 8 \text{ MHz}$ ) 时测量的电源电流 <sup>3</sup>	$RI_{DD}$	5	7.7	11.4	mA
	C			3	7.4	11.2	
3	P	当 (CPU 时钟 = 40 MHz、 $f_{Bus} = 20 \text{ MHz}$ ) 时测量的电源电流 <sup>3</sup>	$RI_{DD}$	5	15	24	mA
	C			3	14	23	
4	$P^3$	停止 3 模式 电源电流 $-40^\circ\text{C}$ (C, V, & M 后缀)	$S3I_{DD}$	5	0.9	—	$\mu\text{A}$
	$P^4$	$25^\circ\text{C}$ (所有部件)			1.0	—	
	P	$105^\circ\text{C}$ (仅 V 后缀)			26	39	
	P	$125^\circ\text{C}$ (仅 M 后缀)			62	90	
	C	$-40^\circ\text{C}$ (C, V, & M 后缀)	$S3I_{DD}$	3	0.8	—	
	C	$25^\circ\text{C}$ (所有部件)			0.9	—	
	C	$105^\circ\text{C}$ (仅 V 后缀)			21	32	
	C	$125^\circ\text{C}$ (仅 M 后缀)			52	80	

表 A-7. 电源电流特性 (续)

编号	C	参数	符号	V <sub>DD</sub> (V)	典型值 <sup>1</sup>	最大值 <sup>2</sup>	单位
5	P <sup>4</sup>	停止 2 模式 电源电流 -40 °C (C, V, & M 后缀)	S2I <sub>DD</sub>	5	0.8	—	μA
	P <sup>4</sup>	25 °C (所有部件)			0.9	—	
	P	105 °C (仅 V 后缀)			25	37	
	P	125 °C (仅 M 后缀)			46	70	
	C	-40 °C (C, V, & M 后缀)		3	0.7	—	
	C	25 °C (所有部件)			0.8	—	
	C	105 °C (仅 V 后缀)			20	30	
	C	125 °C (仅 M 后缀)			40	60	
6	C	增加 RTC 时的停止 2 或停止 3 <sup>4</sup> 的加法器、25°C		5	300	—	nA
				3	300	—	nA
7	C	增加 LVD 的停止 3 (LVDE = LVDSE = 1)		5	110	—	μA
				3	90	—	μA
8	C	增加振荡器启用时 <sup>5</sup> 的停止 3 (IRCLKEN = 1 和 IREFSTEN = 1 或 ERCLKEN = 1 和 EREFSTEN = 1)		5	5	—	μA
				3	5	—	μA

1 典型值典型值在 25°C 时测量的值，除非另有说明。

2 本列中的最大值适用于设备的整个操作温度范围，除非另有说明。

3 25°C 时在所有部件上进行停止电流测试。在其他温度上的测试取决于部件编号后缀及产品的成熟度。一旦收集到足够的数据且被批准，飞思卡尔可能会把特殊温度下的测试从生产测试流程中消除。

4 大多数客户都期望可以使用停止 2 或停止 3 的自动唤醒，而非更高电流的等待模式。

5 以下条件下的给定值：低量程操作 (RANGE = 0)、低功率模式 (HGO = 0)

## A.8 模拟比较器 (ACMP) 电气特性

表 A-8. 模拟比较器电气规范

编号	C	参数	符号	最小值	典型值	最大值	单位
9	—	电源电压	V <sub>DD</sub>	2.7	—	5.5	V
10	D	电源电流 (活动)	I <sub>DDAC</sub>	—	20	35	μA
11	D	模拟输入电压	V <sub>A1N</sub>	V <sub>SS</sub> - 0.3	—	V <sub>DD</sub>	V
12	D	模拟输入偏移电压	V <sub>A1O</sub>		20	40	mV
13	D	模拟比较器滞后	V <sub>H</sub>	3.0	6.0	20.0	mV
14	D	模拟输入漏电流	I <sub>ALKG</sub>	--	--	1.0	μA
15	D	模拟比较器初始化延迟	t <sub>AINIT</sub>	—	—	1.0	μs

## A.9 ADC 特性

表 A-9. 12 位 ADC 操作条件

特性	条件	符号	最小值	典型值 <sup>1</sup>	最大值	单位	注释
电源电压	绝对	V <sub>DDAD</sub>	2.7	—	5.5	V	
	Delta to V <sub>DD</sub> (V <sub>DD</sub> -V <sub>DDAD</sub> ) <sup>2</sup>	DV <sub>DDAD</sub>	-100	0	+100	mV	
接地电压	Delta to V <sub>SS</sub> (V <sub>SS</sub> -V <sub>SSAD</sub> ) <sup>2</sup>	DV <sub>SSAD</sub>	-100	0	+100	mV	
参考电压 高		V <sub>REFH</sub>	2.7	V <sub>DDAD</sub>	V <sub>DDAD</sub>	V	仅在 64 管脚封装中适用 {V <sub>REFH</sub> < V <sub>DDAD</sub> 描述性的, 未在生产中测试 }
参考电压 低		V <sub>REFL</sub>	V <sub>SSAD</sub>	V <sub>SSAD</sub>	V <sub>SSAD</sub>	V	不适用于 64 管脚封装 (只适用于 32 和 48 管脚封装)
输入电压		V <sub>ADIN</sub>	V <sub>REFL</sub>	—	V <sub>REFH</sub>	V	
输入电容		C <sub>ADIN</sub>	—	4.5	5.5	pF	
输入电阻		R <sub>ADIN</sub>	—	3	5	kΩ	
模拟信号源电阻	12 位模式 f <sub>ADCK</sub> > 4MHz f <sub>ADCK</sub> < 4MHz	R <sub>AS</sub>	—	—	2	kΩ	MCU 外部
	10 位模式 f <sub>ADCK</sub> > 4MHz f <sub>ADCK</sub> < 4MHz		—	—	5		
	8 位模式 (所有有效 f <sub>ADCK</sub> )		—	—	10		
ADC 转换时钟频率	高速 (ADLPC=0)	f <sub>ADCK</sub>	0.4	—	8.0	MHz	
	低速 (ADLPC=1)		0.4	—	4.0		

<sup>1</sup> 典型值假设 V<sub>DDAD</sub> = 5.0V、温度 = 25°C、f<sub>ADCK</sub>=1.0MHz, 除非另有其他说明。典型值仅用于参考, 并在生产中测试。

<sup>2</sup> DC 潜在差。

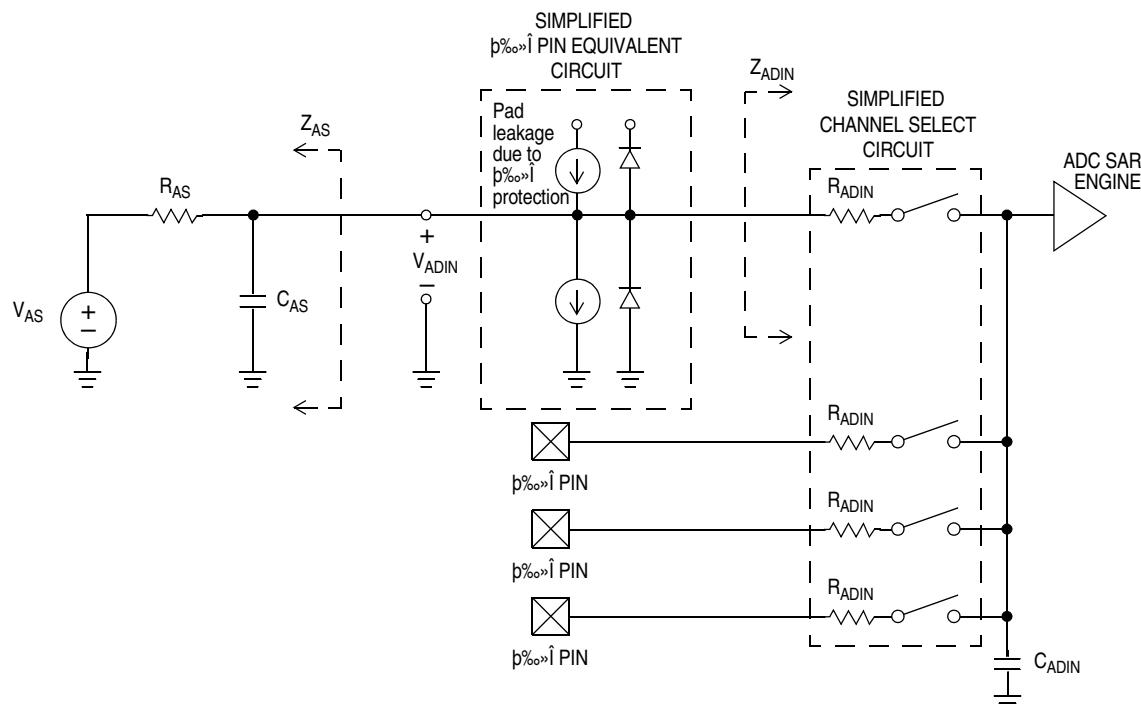


图 A-1. ADC 输入抗阻等效示意图

表 A-10. 12-位 ADC 特性 ( $V_{REFH} = V_{DDAD}$ ,  $V_{REFL} = V_{SSAD}$ )

特性	条件	C	符号	最小值	典型值 <sup>1</sup>	最大值	单位	注释
电源电流	ADLPC=1 ADLSMP=1 ADCO=1	T	$I_{DD} + I_{DDAD}$	—	133	—	$\mu A$	仅 ADC 电流
电源电流	ADLPC=1 ADLSMP=0 ADCO=1	T	$I_{DD} + I_{DDAD}$	—	218	—	$\mu A$	仅 ADC 电流
电源电流	ADLPC=0 ADLSMP=1 ADCO=1	T	$I_{DD} + I_{DDAD}$	—	327	—	$\mu A$	仅 ADC 电流
电源电流	ADLPC=0 ADLSMP=0 ADCO=1	D	$I_{DD} + I_{DDAD}$	—	0.582	1	mA	仅 ADC 电流
电源电流	停止、复位、模块关		$I_{DD} + I_{DDAD}$	—	0.011	1	$\mu A$	仅 ADC 电流
ADC 同步时钟源	高速 (ADLPC=0)	P	$f_{ADACK}$	2	3.3	5	MHz	$t_{ADACK} = 1/f_{ADACK}$
	低速 (ADLPC=1)			1.25	2	3.3		

表 A-10. 12-位 ADC 特性 ( $V_{REFH} = V_{DDAD}$ ,  $V_{REFL} = V_{SSAD}$ ) (续)

特性	条件	C	符号	最小值	典型值 <sup>1</sup>	最大值	单位	注释
转换时间 (包括采样时间)	短采样 (ADLSMP=0)	D	$t_{ADC}$	—	20	—	ADCK 周期	参见表 A-12, 查看转换时间 变量
	长采样 (ADLSMP=1)			—	40	—		
采样时间	短采样 (ADLSMP=0)	D	$t_{ADS}$	—	3.5	—	ADCK 周期	
	长采样 (ADLSMP=1)			—	23.5	—		
未调整误差总数	12 位模式	T	$E_{TUE}$	—	$\pm 3.0$	$\pm 10$	LSB <sup>2</sup>	包括基本倍数
	10 位模式	P		—	$\pm 1$	$\pm 2.5$		
	8 位模式	T		—	$\pm 0.5$	$\pm 1.0$		
差分非线性误差	12 位模式	T	DNL	—	$\pm 1.75$	$\pm 4.0$	LSB <sup>2</sup>	
	10 位模式 <sup>3</sup>	P		—	$\pm 0.5$	$\pm 1.0$		
	8 位模式 <sup>3</sup>	T		—	$\pm 0.3$	$\pm 0.5$		
积分非线性误差	12 位模式	T	INL	—	$\pm 1.5$	$\pm 4.0$	LSB <sup>2</sup>	
	10 位模式	T		—	$\pm 0.5$	$\pm 1.0$		
	8 位模式	T		—	$\pm 0.3$	$\pm 0.5$		
零刻度误差	12 位模式	T	$E_{zs}$	—	$\pm 1.5$	$\pm 6.0$	LSB <sup>2</sup>	$V_{ADIN} = V_{SSAD}$
	10 位模式	P		—	$\pm 0.5$	$\pm 1.5$		
	8 位模式	T		—	$\pm 0.5$	$\pm 0.5$		
满刻度误差	12 位模式	T	$E_{FS}$	—	$\pm 1$	$\pm 4.0$	LSB <sup>2</sup>	$V_{ADIN} = V_{DDAD}$
	10 位模式	T		—	$\pm 0.5$	$\pm 1$		
	8 位模式	T		—	$\pm 0.5$	$\pm 0.5$		
量化误差	12 位模式	D	$E_Q$	—	-1 to 0	-1 to 0	LSB <sup>2</sup>	
	10 位模式			—	—	$\pm 0.5$		
	8 位模式			—	—	$\pm 0.5$		
输入漏电流误差	12 位模式	D	$E_{IL}$	—	$\pm 1$	$\pm 10.0$	LSB <sup>2</sup>	Pad leakage <sup>4 *</sup> $R_{AS}$
	10 位模式			—	$\pm 0.2$	$\pm 2.5$		
	8 位模式			—	$\pm 0.1$	$\pm 1$		
温度传感器范围	-40°C–25°C	D	m	—	3.266	—	mV/ °C	
	25°C–125°C			—	3.638	—		
温度传感器电压	25°C	D	$V_{TEMP25}$	—	1.396	—	V	

<sup>1</sup> 典型值假设  $V_{DDAD} = 5.0V$ 、温度 =  $25 \times C$ ,  $f_{ADCK}=1.0MHz$ , 除非另有其他说明。典型值仅用于参考，并在生产中测试。

<sup>2</sup> 1 LSB =  $(V_{REFH} - V_{REFL}) / 2^N$

<sup>3</sup> 10 位和 8 位模式中保证单调和无丢码。

<sup>4</sup> 基于输入引脚 (input pad) 漏电流。请参考板电气 (pad electricals)。

## A.10 外部振荡器 (XOSC) 特性

表 A-11. 振荡器电气规范 (温度范围 = -40 - 125 °C)

编号	C	参数	符号	最小值	典型值 <sup>1</sup>	最大值	单位	
16	C	振荡器晶体或共鸣器 (EREFS = 1, ERCLKEN = 1)						
		低量程 (RANGE = 0)	$f_{lo}$	32	—	38.4	kHz	
		高量程 (RANGE = 1) FEE 或 FBE 模式 <sup>2</sup>	$f_{hi-fll}$	1	—	5	MHz	
		高量程 (RANGE = 1) PEE 或 PBE 模式 <sup>3</sup>	$f_{hi-pll}$	1	—	16	MHz	
		高量程 (RANGE = 1, HGO = 1) BLPE 模式	$f_{hi-hgo}$	1	—	16	MHz	
		高量程 (RANGE = 1, HGO = 0) BLPE 模式	$f_{hi-lp}$	1	—	8	MHz	
17	—	载荷电容器	$C_1$ $C_2$	参见晶体或共鸣器制造商的推荐。				
18	—	反馈电阻器						
		低量程 (32 kHz to 100 kHz)	$R_F$	—	10	—	MΩ	
		高量程 (1 MHz to 16 MHz)		—	1	—	MΩ	
19	—	串行电阻器						
		低量程, 低增益 (RANGE = 0, HGO = 0)	$R_S$	—	0	—	kΩ	
		低量程, 高增益 (RANGE = 0, HGO = 1)		—	100	—		
		高量程, 低增益 (RANGE = 1, HGO = 0)		—	0	—		
		高量程, 高增益 (RANGE = 1, HGO = 1)		≥ 8 MHz	0	0		
				4 MHz	0	10		
				1 MHz	0	20		
20	T	晶体启动时间 <sup>4</sup>						
		低量程, 低增益 (RANGE = 0, HGO = 0)	$t_{CSTL-LP}$	—	200	—	ms	
		低量程, 高增益 (RANGE = 0, HGO = 1)	$t_{CSTL-HGO}$	—	400	—		
		高量程, 低增益 (RANGE = 1, HGO = 0) <sup>5</sup>	$t_{CSTH-LP}$	—	5	—		
		高量程, 高增益 (RANGE = 1, HGO = 1) <sup>4</sup>	$t_{CSTH-HGO}$	—	15	—		
21	T	方波输入时钟频率 (EREFS = 0, ERCLKEN = 1)						
		FEE 或 FBE 模式 <sup>2</sup>	$f_{extal}$	0.03125	—	5	MHz	
		PEE 或 PBE 模式 <sup>3</sup>		1	—	16		
		BLPE 模式		0	—	40		

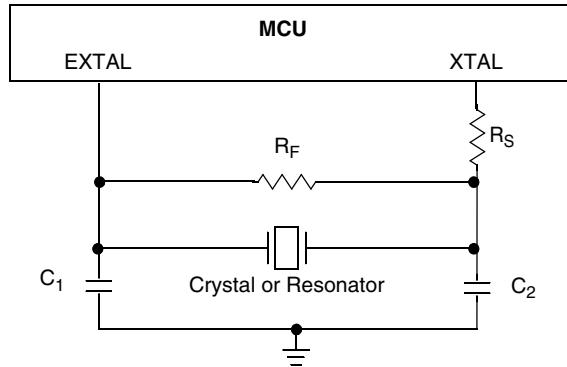
<sup>1</sup> 典型数据是电压为 3.0 V、温度为 25°C 时的数据或推荐值。

<sup>2</sup> 当为 FEE 或 FBE 模式配置 MCG 时, 必须能够用 RDIV 将输入时钟源分割在 31.25 kHz 至 39.0625 kHz 的范围内。

<sup>3</sup> 当为 PEE 或 PBE 模式配置 MCG 时, 能够能够用 RDIV 将输入时钟源分割在 1 kHz 至 2 kHz 的范围内。

<sup>4</sup> 该参数是描述性数据, 未在每个器件上进行测试。要达到规范, 必须遵守正确的 PC 主板布局流程。

<sup>5</sup> 4 MHz 晶体。



## A.11 MCG 规范

表 A-12. CG 频率规范 (温度范围 = -40 至 125 °C )

编号	C	参数	符号	最小值	典型值	最大值	单位
1	P	当 $V_{DD} = 5 \text{ V}$ 、温度 = 25 °C 时的工厂调整值	$f_{int\_ft}$	—	31.25	—	kHz
2	P	平均内部参考频率 - 未调整 <sup>1</sup>	$f_{int\_ut}$	25	32.7	41.66	kHz
3	P	平均内部参考频率 - 用户调整	$f_{int\_t}$	31.25	—	39.0625	kHz
4	D	内部参考启动时间	$t_{irefst}$	—	60	100	us
5	—	DCO 输出频率范围 - 为参考提供的未调整值 <sup>1</sup> : $f_{dco\_ut} = 1024 \times f_{int\_ut}$	$f_{dco\_ut}$	25.6	33.48	42.66	MHz
6	P	DCO 输出频率范围 - 已调整	$f_{dco\_t}$	32	—	40	MHz
7	C	电压和温度固定时经调整的 DCO 输出频率的分辨率 (使用 FTRIM)	$\Delta f_{dco\_res\_t}$	—	± 0.1	± 0.2	% $f_{dco}$
8	C	电压和温度固定时经调整的 DCO 输出频率的分辨率 (不使用 FTRIM)	$\Delta f_{dco\_res\_t}$	—	± 0.2	± 0.4	% $f_{dco}$
9	P	已调整 DCO 输出频率的电压和温度总误差	$\Delta f_{dco\_t}$	—	+ 0.5 -1.0	± 2	% $f_{dco}$
10	C	0 - 70 °C 的温度范围内时已调整 DCO 输出频率的总误差	$\Delta f_{dco\_t}$	—	± 0.5	± 1	% $f_{dco}$
11	C	FLL 获取时间 <sup>2</sup>	$t_{fll\_acquire}$	—	—	1	ms
12	D	PLL 获取时间 <sup>3</sup>	$t_{pll\_acquire}$	—	—	1	ms
13	C	输出时钟的长时间抖动 (平均间隔为 2ms) <sup>4</sup>	$C_{Jitter}$	—	0.02	0.2	% $f_{dco}$
14	D	VCO 操作频率	$f_{vco}$	7.0	—	55.0	MHz
15	D	PLL 参考频率范围	$f_{pll\_ref}$	1.0	—	2.0	MHz
16	T	输出时钟的长期准确性 (平均为 2 ms)	$f_{pll\_jitter\_2ms}$	—	0.590 <sup>5</sup>	—	% $f_{pll}$
17	T	基于 625 ns <sup>6</sup> 测量的 PLL 输出时钟抖动	$f_{pll\_jitter\_625ns}$	—	0.566 <sup>5</sup>	—	% $f_{pll}$
18	D	锁定进入频率容限 <sup>7</sup>	$D_{lock}$	± 1.49	—	± 2.98	%
19	D	锁定退出频率容限 <sup>8</sup>	$D_{unl}$	± 4.47	—	± 5.97	%

表 A-12. CG 频率规范 (温度范围 = -40 至 125 °C )

编号	C	参数	符号	最小值	典型值	最大值	单位
20	D	锁定时间 - FLL	$t_{\text{fll\_lock}}$	—	—	$t_{\text{fll\_acquire+}} + 1075$ $(1/f_{\text{int\_t}})$	s
21	D	锁定时间 - PLL	$t_{\text{pll\_lock}}$	—	—	$t_{\text{pll\_acquire+}} + 1075$ $(1/f_{\text{pll\_ref}})$	s
22	D	外部时钟最小频率损失 - RANGE = 0	$f_{\text{loc\_低}}$	$(3/5) \times f_{\text{int}}$	—	—	kHz
23	D	外部时钟最小频率损失 - RANGE = 1	$f_{\text{loc\_高}}$	$(16/5) \times f_{\text{int}}$	—	—	kHz

<sup>1</sup> TRIM 寄存器的默认值为 (0x80), FTRIM 控制位的默认值为 (0x0)。.

<sup>2</sup> 本规范适用于更改 FLL 参考源或参考分频器、更改 trim 值或从 FFL 禁用 (BLPE、BLPI) 更改为 FLL 启用 (FEI、FEE、FBE、FBI) 的任何情况。如果晶体 / 共鸣器正作为参考使用，本规范假设它已经在运行。

<sup>3</sup> 本规范适用于 PLL VCO 分频器或参考分频器、或从 PLL 禁用 (BLPE、BLPI) 更改为 PLL 启用 (PBE、PEE) 的任何情况。如果晶体 / 共鸣器正作为参考使用，本规范假设它已经在运行。

<sup>4</sup> 抖动是当  $f_{\text{BUS}}$  最大时在指定间隔内测量的与编辑频率的平均偏差。测量采用由已过滤电源供电的器件，并由稳定的外部时钟信号给出时间。因  $V_{\text{DD}}$  和  $V_{\text{SS}}$  和晶体振荡器频率的变化注入 FLL 电路的噪声会提高给定间隔内的  $C_{\text{jitter}}$  百分比。

<sup>5</sup> 抖动测量基于 48 MHz MCGOUT 时钟频率。

<sup>6</sup> 625 ns 代表着 CAN 应用的 5 个时间量子，且在 8 MHz CAN 总线时钟、1 Mbps CAN 总线速度和位时间设置的每位 8 个时间量子最差条件下。5 个时间量子是同步边缘与位（使用 8 个时间量子 / 位）的采样点间的最长时间

<sup>7</sup> 在  $D_{\text{lock}}$  最小值以下，MCG 保证进入锁定状态；在  $D_{\text{lock}}$  最大值以上，MCG 将不进入锁定状态。但是，如果 MCG 已经处于锁定状态，那么 MCG 可能就保持锁定状态。

<sup>8</sup> 在  $D_{\text{unl}}$  最小值以下，MCG 将不退出锁定状态，如果已经处于锁定状态的话。在  $D_{\text{unl}}$  最大值以上，MCG 保证退出锁定状态。

## A.12 AC 特性

本小节描述每个外围系统的 AC 定时特性。

## A.12.1 控制时序

表 A-13. 控制时序

编号	C	参数	符号	最小值	典型值	最大值	单位
1		总线频率 ( $t_{cyc} = 1/f_{Bus}$ )	$f_{Bus}$	dc	—	20	MHz
2		内部低功率振荡器时间	$t_{LPO}$	700		1300	$\mu s$
3		外部复位脉冲宽度 <sup>1</sup>	$t_{extrst}$	$1.5 \times t_{cyc}$		—	ns
4		复位低驱动器 <sup>2</sup>	$t_{rstdrv}$	$34 \times t_{cyc}$		—	ns
5		激活背景调试模式锁定建立时间	$t_{MSSU}$	25		—	ns
6		激活背景调试模式锁定保持时间	$t_{MSH}$	25		—	ns
7		IRQ/PIAx/ PIBx/PIDx 脉冲宽度 异步路径 <sup>2</sup> 同步路径 <sup>3</sup>	$t_{ILIH}, t_{IHIL}$	100 $1.5 t_{cyc}$	—	—	ns
8	T	端口升降时间 — 低输出驱动器 ( $PTxDS = 0$ ) (载荷 = 50 pF) <sup>3</sup> 斜率控制禁用 ( $PTxSE = 0$ ) 斜率控制启用 ( $PTxSE = 1$ )	$t_{Rise}, t_{Fall}$	— —	40 75		ns
		端口升降时间 — 高输出驱动器 ( $PTxDS = 1$ ) (载荷 = 50 pF) <sup>3</sup> 斜率控制禁用 ( $PTxSE = 0$ ) 斜率控制启用 ( $PTxSE = 1$ )	$t_{Rise}, t_{Fall}$	— —	11 35		ns

<sup>1</sup> 这是被识别为复位管脚请求所保证的最短脉冲。不保证更短脉冲来覆盖来自内部源的复位请求。

<sup>2</sup> 当发起任意复位时，内部电路驱动 复位 pin 低管脚下降，实现大约 34 个  $t_{cyc}$  周期。在 POR 复位后，总线时钟频率更改为未调整的 DCO 频率 ( $f_{复位} = (f_{dco\_ut}) / 4$ )，因为 TRIM 被复位为 0x80，FTRIM 被复位为 0。而且还额外除以 2，因为 BDIV 被复位为 0:1。在其他复位后，trim 始终保持在预先设定的值上。

<sup>3</sup> 定时分别显示为 20%  $V_{DD}$  和 80%  $V_{DD}$  水平。温度范围是 -40°C 至 125°C。

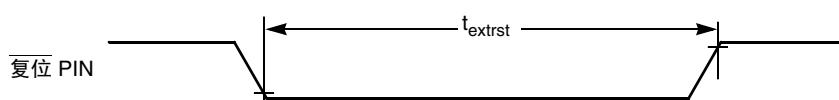


图 A-2. 复位时序

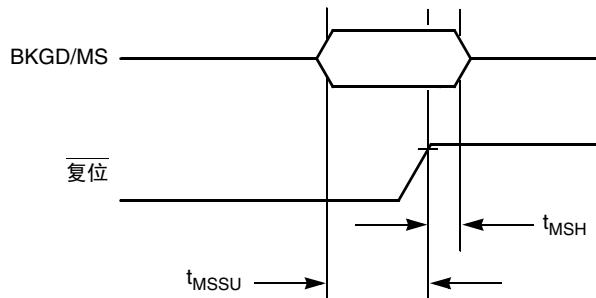


图 A-3. 激活背景调试模式锁定时序

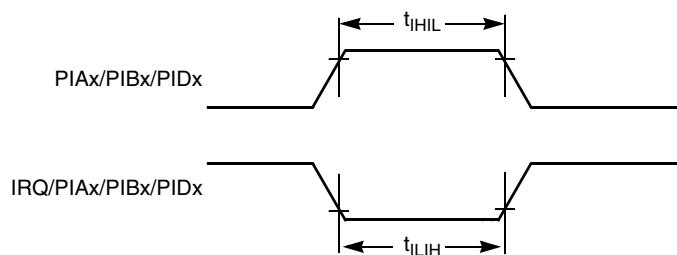


图 A-4. 管脚中断时序

## A.12.2 定时器 /PWM

同步器电路决定着可被识别的最短输入脉冲或可以用作定时器计数器的可选外部源的最快时钟。这些同步器当前的总线速率时钟中进行操作。

表 A-14. TPM 输入时序

编号	C	参数	符号	最小值	最大值	单位
9	—	外部时钟频率	$f_{TCLK}$	dc	$f_{Bus}/4$	MHz
10	—	外部时钟时间	$t_{TCLK}$	4	—	$t_{cyc}$
11	D	外部时钟高时间	$t_{clkh}$	1.5	—	$t_{cyc}$
12	D	外部时钟低时间	$t_{clkL}$	1.5	—	$t_{cyc}$
13	D	输入捕捉脉冲宽度	$t_{ICPW}$	1.5	—	$t_{cyc}$

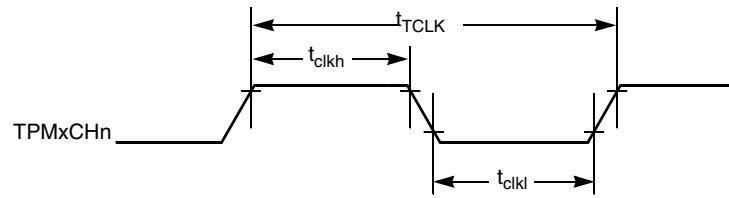


图 A-5. 定时器外部时钟

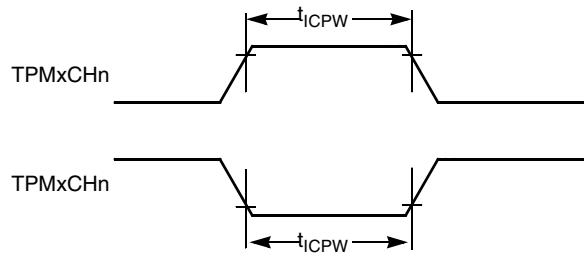


图 A-6. 定时器输入捕捉脉冲

### A.12.3 MSCAN

表 A-15. MSCAN 唤醒脉冲特性

编号	C	参数	符号	最小值	典型值	最大值	单位
14	D	MSCAN 唤醒显性脉冲过滤	$t_{WUP}$	—	—	2	ms
15	D	MSCAN 唤醒显性脉冲通过	$t_{WUP}$	5	—	—	ms

## A.12.4 SPI

表 A-16 和图 A-7 至图 A-10 描述了 SPI 系统的定时要求。

表 A-16. SPI 电气特性

编号 <sup>1</sup>	C	参数 <sup>2</sup>	符号	最小值	最大值	单位
1	D	周期时间 主从	$t_{SCK}$ $t_{SCK}$	2 4	— 2048	$t_{cyc}$ $t_{cyc}$
2	D	使能前置时间 主从	$t_{Lead}$ $t_{Lead}$	— 1/2	1/2 —	$t_{SCK}$ $t_{SCK}$
3	D	使能落后时间 主从	$t_{Lag}$ $t_{Lag}$	— 1/2	1/2 —	$t_{SCK}$ $t_{SCK}$
4	D	时钟 (spck) 高电平时间 主和从	$t_{SCKH}$	(1/2 $t_{SCK}$ ) - 25	—	ns
5	D	时钟 (spck) 低电平时间 主和从	$t_{SCKL}$	(1/2 $t_{SCK}$ ) - 25	—	ns
6	D	数据建立时间 (输入) 主从	$t_{SI(M)}$ $t_{SI(S)}$	30 30	— —	ns ns
7	D	数据保持时间 (输入) 主从	$t_{HI(M)}$ $t_{HI(S)}$	30 30	— —	ns ns
8	D	存取时间, 从 <sup>3</sup>	$t_A$	0	40	ns
9	D	禁止时间, 从 <sup>4</sup>	$t_{dis}$	—	40	ns
10	D	数据设置时间 (输出) 主从	$t_{SO}$ $t_{SO}$	25 25	— —	ns ns
11	D	数据建立时间 (输出) 主从	$t_{HO}$ $t_{HO}$	-10 -10	— —	ns ns
12	D	操作频率 <sup>5</sup> 主从	$f_{op}$ $f_{op}$	$f_{Bus}/2048$ dc	5 $f_{Bus}/4$	MHz

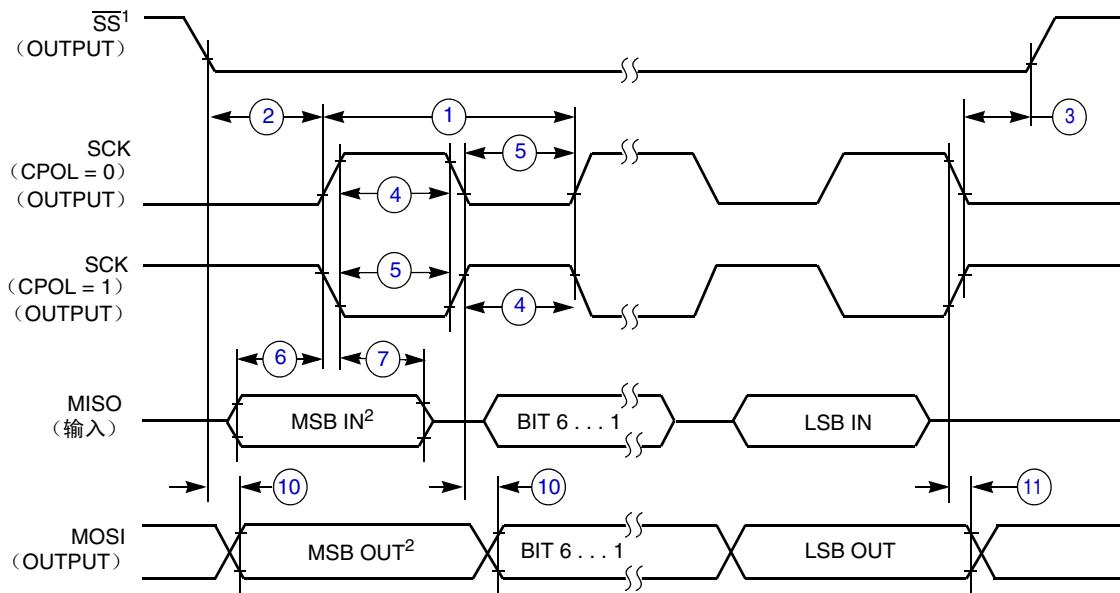
<sup>1</sup> 参见图 A-7 至图 A-10。

<sup>2</sup> 对于 20%  $V_{DD}$  和 70%  $V_{DD}$ , 显示所有时钟, 除非另有说明。所有 SPI 管脚上 100 pF 载荷。所有时钟假设 SPI 输出管脚禁用斜率控制, 启用高驱动强度。

<sup>3</sup> 从高抗阻状态到数据激活的时间。

<sup>4</sup> 高抗阻状态的保持时间。

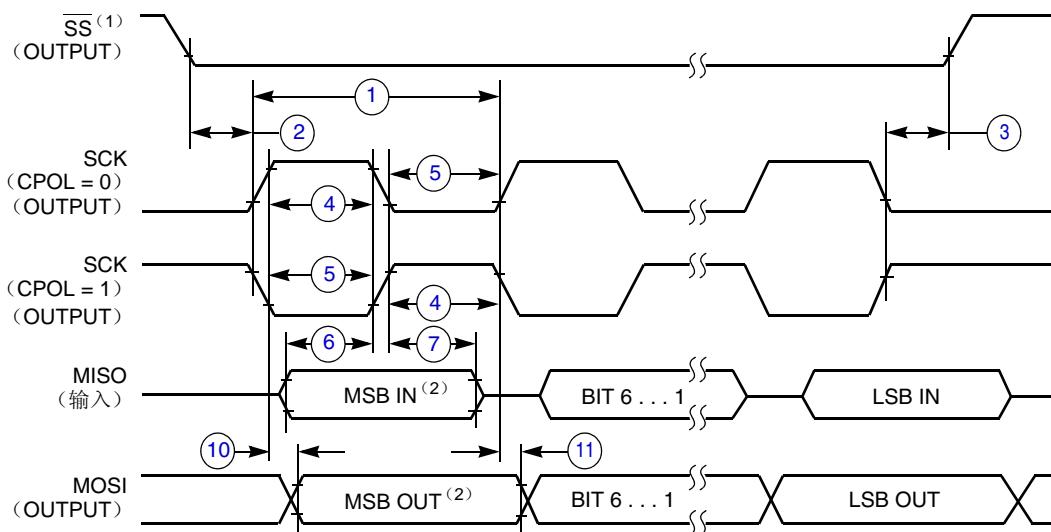
<sup>5</sup> 由于引脚输入 (pad) 特性, 最大波特率必须限定为 5 MHz。



注释:

1. SS 输出模式 (MODFEN = 1, SSOE = 1).
2. LSBF = 0。当 LSBF = 1 时, 位顺序是 LSB、位 1、...、位 6、MSB。

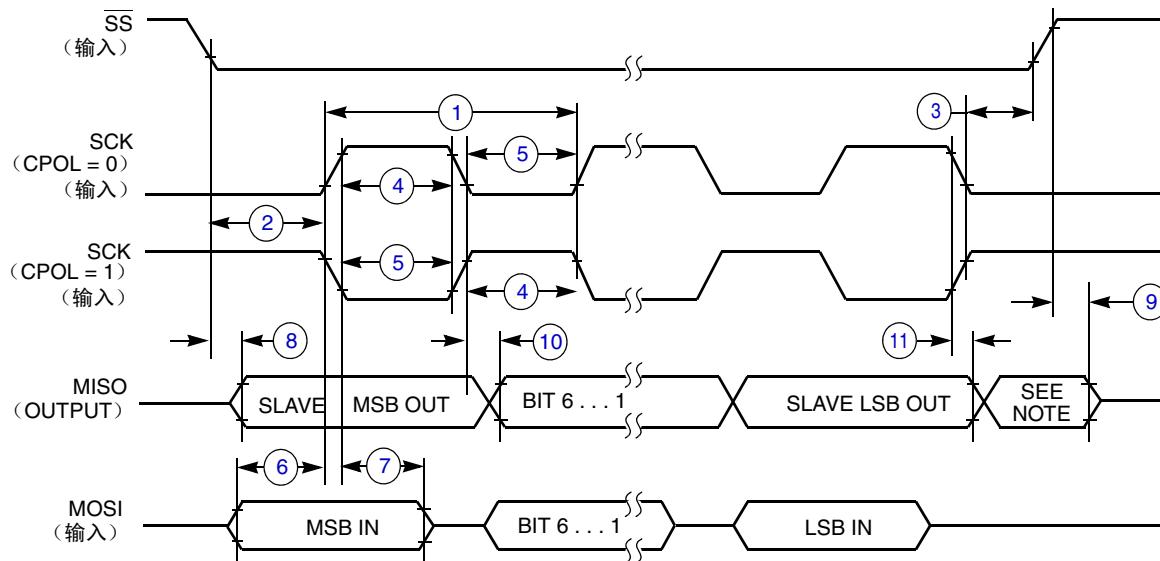
图 A-7. SPI 主时时序 (CPHA = 0)



注释:

1. SS输出模式 (MODFEN = 1, SSOE = 1)
2. LSBF = 0。当 LSBF = 1 时, 位顺序是 LSB、位 1、...、位 6、MSB。B.

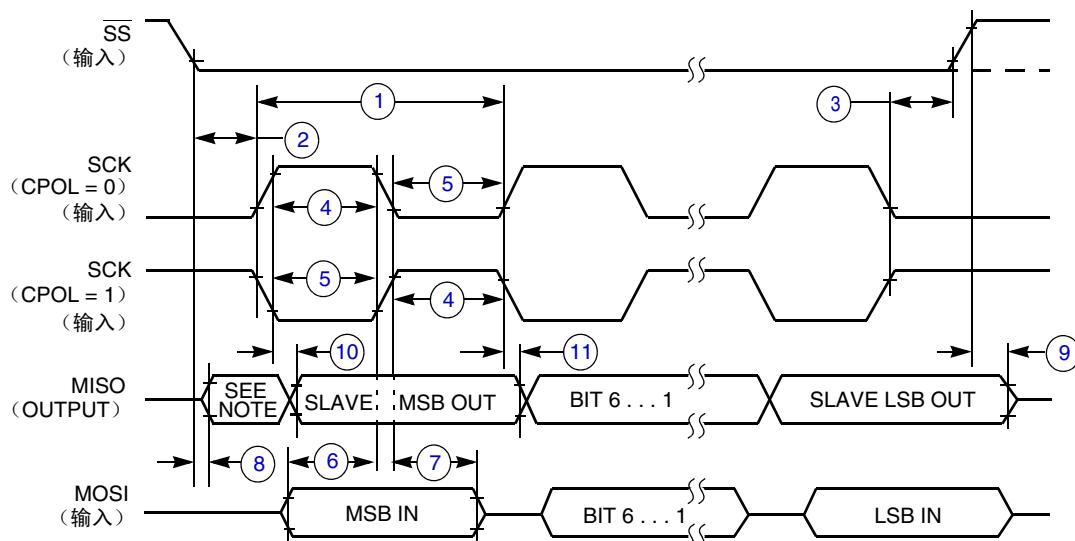
图 A-8. SPI 主时时序 (CPHA = 1)



注释:

1. 未定义, 但在正常情况下只接收 MSB 字符

图 A-9. SPI 从时序 (CPHA = 0)



注释:

1. 未定义, 但在正常情况下只接收 LSB 字符

图 A-10. SPI 从时序 (CPHA = 1)

## A.13 闪存和 EEPROM

本小节详细地描述闪存和 EEPROM 存储器的编程 / 擦除次数及编程 - 擦除容限。

编程和擦除操作除正常  $V_{DD}$  supply 电源外不需要任何特殊电源。有关编程 / 擦除操作的更多信息，请参见第 4 章，“存储器”。

表 A-17. 闪存和 EEPROM 特性

编号	C	参数	符号	最小值	典型值	最大值	单位
16	—	编程 / 擦除的电源电压	$V_{prog/erase}$	2.7		5.5	V
17	—	读取操作的电源电压 $0 < f_{Bus} < 8 \text{ MHz}$ $0 < f_{Bus} < 20 \text{ MHz}$	$V_{Read}$	2.7		5.5	V
18	—	内部 FCLK 频率 <sup>1</sup>	$f_{FCLK}$	150		200	kHz
19	—	内部 FCLK 时间 (1/FCLK)	$t_{Fcyc}$	5		6.67	$\mu\text{s}$
20	—	字节编程时间 (任意位置) <sup>(2)</sup>	$t_{prog}$	9			$t_{Fcyc}$
21	—	字节编程时间 (突发模式) <sup>(2)</sup>	$t_{Burst}$	4			$t_{Fcyc}$
22	—	页面擦除时间 <sup>2</sup>	$t_{Page}$	4000			$t_{Fcyc}$
23	—	块擦除时间 <sup>(2)</sup>	$t_{Mass}$	20,000			$t_{Fcyc}$
24	C	闪存编程 / 擦除次数 <sup>3</sup> $T_L$ 至 $T_H = -40^\circ\text{C}$ to $+125^\circ\text{C}$ $T = 25^\circ\text{C}$	$n_{FLPE}$	10,000 —	— 100,000	— —	cycles
25	C	编程 / 擦除次数 <sup>3</sup> $T_L$ 至 $T_H = -40^\circ\text{C}$ to $+0^\circ\text{C}$ $T_L$ to $T_H = 0^\circ\text{C}$ to $+125^\circ\text{C}$ $T = 25^\circ\text{C}$	$n_{EEPE}$	10,000 50,000 —	— — 100,000	— — —	cycles
26	C	数据保留时间 <sup>4</sup>	$t_{D\_ret}$	15	100	—	years

<sup>1</sup> 该时钟的频率由软件设置控制。

<sup>2</sup> 这些值是硬件状态设备控制的值。用户代码无需计周期数。提供该信息的目的是为了计算编程和擦除的大约时间。

<sup>3</sup> 闪存和 EEPROM 的典型容限基于内在的位信元性能。有关飞思卡尔半导体如何定义典型容限的更多信息，请参考 Engineering Bulletin EB619, 非易失性存储器的典型容限。

<sup>4</sup> 典型数据保留时间值基于在高温时测量的技术的内在能力，并使用阿伦尼乌斯公式降到  $25^\circ\text{C}$ 。有关飞思卡尔半导体如何定义典型数据保留时间的更多信息，请参考 Engineering Bulletin EB618, 非易失性存储器的典型数据保留时间。

## A.14 EMC 性能

电磁兼容性（EMC）性能在很大程度上取决于 MCU 所在的环境。主板设计及布局、电路拓扑选择、外部组件的位置和特性以及 MCU 软件操作等都在 EMC 性能中起到重要的作用。系统设计人员应参照飞思卡尔应用笔记，如 AN2321、AN1050、AN1263、AN2764 和 AN1259，获取优化 EMC 性能的建议和指导。

### A.14.1 辐射放射性

微控制器 RF 辐射放射性根据 IEC 61967-2 和 SAE J1752/3 标准，用 TEM/GTEM Cell 方法在 150 kHz 至 1 GHz 中进行测量。测量用安装在定制 EMC 评估板上的微控制器执行，同时运行专门的 EMC 测试软件。微控制器的放射性排放在 TEM 信元的两个封装方位（北和东）中测量。有关评估结果、条件和设置等的更多信息，请参考本器件的 EMC 评估报告。

所有方位中已测试配置的最大 RF 辐射放射性应小于或等于已报告的排放水平。

表 A-18. 3M05C Mask Set 的辐射放射性

参数	符号	条件	频率	$f_{osc}/f_{CPU}$	水平 <sup>1</sup> (最大值)	单位
放射性排放、电场 — 条件 - TBD	$V_{RE\_TEM}$	$V_{DD} = 5$ $T_A = +25^\circ C$ 64 LQFP	0.15 – 50 MHz	16 MHz 晶体 I 20 MHz Bus	18	dB $\mu$ V
			50 – 150 MHz		18	
			150 – 500 MHz		13	
			500 – 1000 MHz		7	
			IEC Level		L	
			SAE Level		2	

<sup>1</sup> 基于条件测试结果的数据。



## 附录 B 定时器脉宽调制器 (TPMV2)

### 注意

本章节参考 S08TPM 第二版本，它适用于该器件的 3M05C 及更旧的掩膜版本。0M74K 和更新的掩膜版本采用 S08TPM 第三版本。如果你的器件采用 0M74K 或更新的掩码，请参见 [299 页上的第 16 章，“定时器脉冲宽度调节器 \(S08TPMV3\)”](#)，了解该模块的信息。

### B.1 介绍

TPM 采用每通道一个 I/O 管脚，TPMxCHn，其中 x 是 TPM 数量 (如 1 或 2)，n 是通道的数量 (如 0 - 4)。TPM 与通用 I/O 端口管脚共享其 I/O 管脚 (参见 [Pins and Connections](#) 章节，了解更多信息)。

### B.2 特性

TPM 提供以下特性：

- 每个 TPM 可以配置为所有通道上缓冲的且中央对齐的脉宽调制 (CPWM)
- 可以为每个 TPM 独立选择时钟源 (多个 TPM 器件)
- 时钟源可选择 (根据器件选择): 总线时钟、固定系统时钟、外部管脚
- 时钟预分频点按 1, 2, 4, 8, 16, 32, 64, 或 128 分
- 16- 位自由运行或上 / 下 (CPWM) 计数操作
- 16- 位模量寄存器控制计数器范围
- 定时器系统使能
- 每个通道一个中断加上每个 TPM 模块的终端计数中断 (多个 TPM 器件)
- 通道特性：
  - 每个通道可以是输入捕捉，输出比较，或缓冲边沿对齐的 PWM
  - 上升边，下降边或任意边输入捕捉触发器
  - 设置、清除或固定输出比较行动
  - 在 PWM 输出上可选择极性

### B.3 结构图

图 B-1 显示了 TPM 的结构。一些 MCU 包括不止一个 TPM，通道数量也不同。

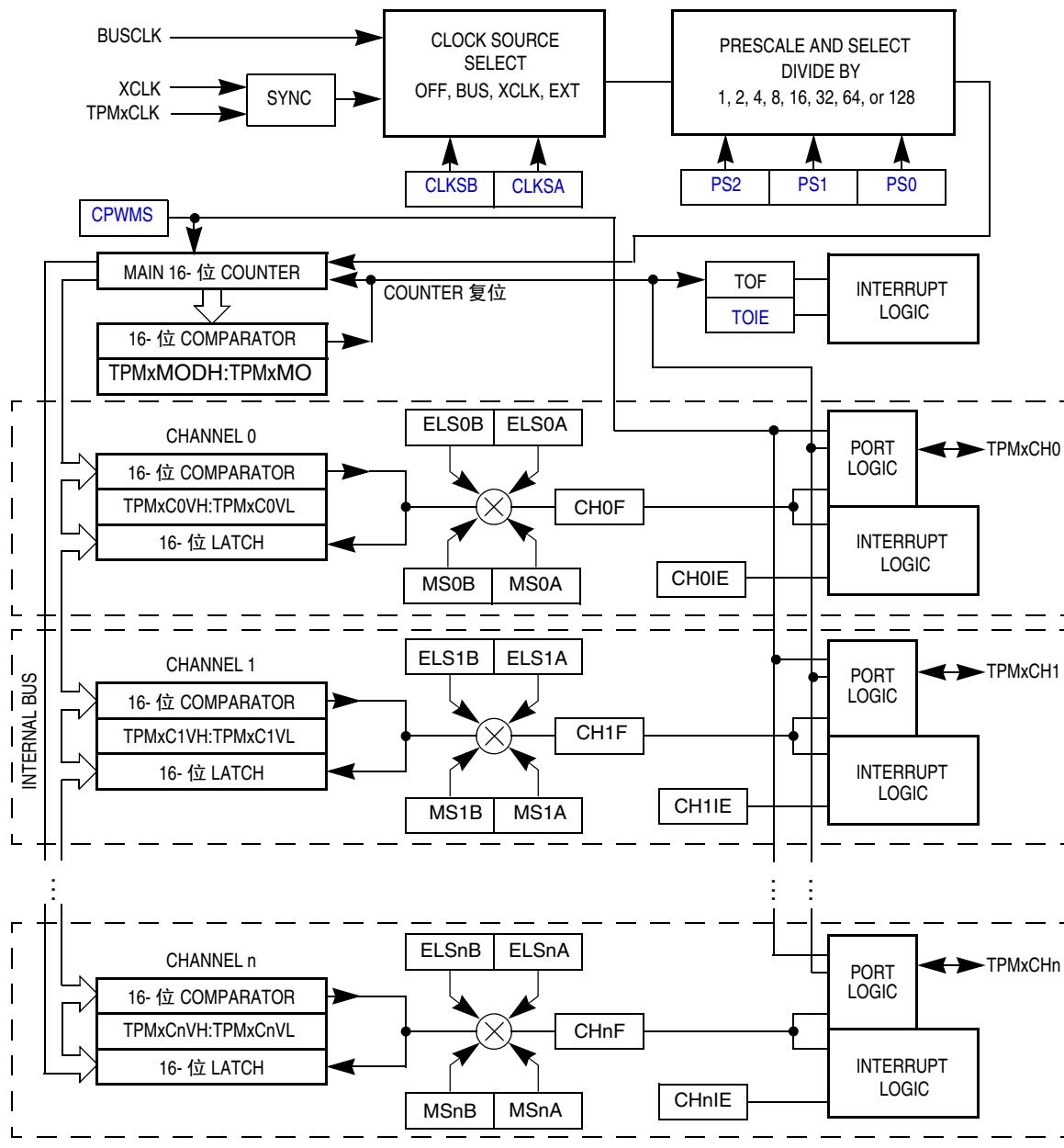


图 B-1. TPM 结构图

TPM 的核心组件是 16 位计数器，当 TPM 配置用于中央对齐的 PWM 时，它可以用为自由运行的计数器，或模数计数器或上 / 下计数器。TPM 计数器（当以普通的向上计数模式运行时）为输入捕捉、输出比较和边沿对齐 PWM 等功能提供时钟参考。定时器计数器模数寄存器，**TPMxMODH:TPMxMODL**，控制计数器的模数值。（0x0000 或 0xFFFF 值有效地使计数器处于自由运行状态。）软件可以在任何时候读取计数器的值，而不会影响计数顺序。**TPMxCNT** 计数器字节的任何写入会复位计数器，无论写入的什么数据值。

所有 **TPM** 通道都可以独立编程为输入捕捉、输出比较或缓冲边沿 **PWM** 通道。

## B.4 外部信号描述

与定时器关联的任意管脚配置为定时器输入时，被动上拉功能使能。复位后，**TPM** 模块禁止，所有管脚默认设置为被动上拉功能禁止的通用输入。

### B.4.1 外部 **TPM** 时钟源

当定时器状态和控制器寄存器里的控制位 **CLKSB:CLKSA** 设置为 1:1 时，预分频器及 **TPMx** 的 16 位计数器由外部时钟源 **TPMxCLOCK** 驱动。该时钟源与 I/O 管脚连接。外部时钟和 **TPM** 的其余部分之间需要一个同步装置。该同步装置采用总线时钟，因而外部源的频率必须低于总线速率时钟频率的 1.5 倍。这个外部时钟源的频率上限明确规定为总线频率的 1/4，以便能适应负载循环和相位锁定环路（PLL）/ 频率锁定环路（FLL）频率抖动产生的影响。

在部分设备上，外部时钟输入由其中一个 **TPM** 通道共享。当 **TPM** 通道作为外部时钟输入共享时，关联 **TRM** 通道不能使用该管脚。（该通道仍然可以作为软件定时器用于输出比较模式）。此外，如果其中一个 **TPM** 通道用作外部时钟输入，对应的 **ELSnB:ELSnA** 控制位必须设置为 0:0，因此该通道不会使用同一个管脚。

### B.4.2 **TPMxCHn** — **TPMx** 通道 n I/O 管脚

所有 **TPM** 通道都与 **MCU** 上的一个 I/O 管脚关联。这个管脚的功能与通道配置有关。部分情况不需要管脚功能，因此该管脚由通用 I/O 控制装置进行控制。当定时器拥有某个端口管脚的控制时，端口数据和数据方向寄存器不会对关联管脚产生影响。如需了解共享管脚功能的相关信息，请参见 [Pins and Connections](#) 章节。

## B.5 寄存器定义

**TPM** 包括：

- 8 位状态和控制寄存器 (**TPMxSC**)
- 16 位计数器 (**TPMxCNTH:TPMxCNTL**)
- 16 位模量寄存器 (**TPMxMODH:TPMxMODL**)

每个定时器通道都包括：

- 8 位状态和控制寄存器寄存器 (**TPMxCnSC**)
- 16 位通道值寄存器 (**TPMxCnVH:TPMxCnVL**)

如需了解所有 **TPM** 寄存器的绝对地址分配信息，请参见本文 [Memory](#) 章节的直接地址页寄存器概况。

## B.5.1 定时器状态和控制寄存器 (TPMxSC)

TPMxSC 包含溢出状态标志和控制位，用来配置中断使能、TPM 配置、时钟源和预分频器除数。这些控制与本定时器模块中的所有通道相关。

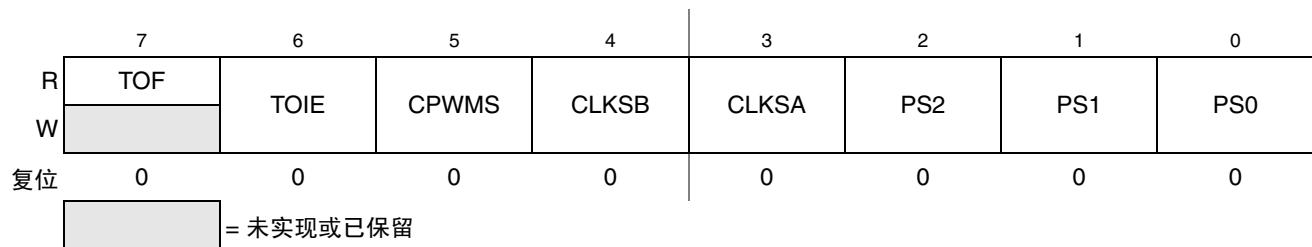


图 B-2. 定时器状态和控制寄存器 (TPMxSC)

表 B-1. TPMxSC 寄存器字段描述

字段	描述
7 TOF	<b>定时器溢出标记</b> — 该标记用于 TPM 计数器变更为 x0000 时，并且已经到达 TPM 计数器模量寄存器中设定的模量值。当 TPM 配置用于 CPWM 时，计数器达到模量寄存器的值后设置 TOF，这时转化为下一个较低的计数值。TOF 设置完毕后，读取 TPM 状态和控制寄存器，在 TOF 中写入 0，可以清除 TOF。如果在清除顺序完成之前，发生另一个 TPM 溢出，该顺序要复位，这样前一个 TOF 的清除顺序完成后，后一个 TOF 仍然能保持为设置状态。复位可清除 TOF。在 TOF 中写入 1 不会产生影响。 0 TPM 计数器没有达到模量值或没有溢出 1 TPM 计数器已经溢出
6 TOIE	<b>定时器溢出中断使能</b> — 这个读 / 写位使能 TPM 溢出中断。如果 TOIE 被设置，那么在 TOF 等于 1 时会生成中断。复位可清除 TOIE。 0 TOF 中断关闭（用于软件轮询） 1 TOF 中断允许
5 CPWMS	<b>中心对齐 PWM 选择</b> — 这个读写位选择 CPWM 的操作模式。复位清除该位，这样 TPM 就在向上计数模式中运行，完成输入捕捉、输出比较和边沿对齐 PWM 功能。设置 CPWMS 可以重新配置 TPM，使它以向上 / 下计数模式操作完成 CPWM 功能。复位清除 CPWMS。 0 所有 TPMx 通道以输入捕捉、输出比较或边沿对齐 PWM 模式操作，由各通道的状态和控制寄存器中 MSnB:MSnA 控制位的选择 1 所有 TPMx 通道以中心对齐 PWM 模式操作
4:3 CLKS[B:A]	<b>时钟源选择</b> — 如表 A-2 所示，2 位字段用来禁止 TPM 系统或在 3 个时钟源中任选一个驱动计数器预分频器。外部源和 XCLK 通过片上同步电路，完成与总线的时钟同步。
2:0 PS[2:0]	<b>预分频器除数选择</b> — 这个 3 位字段从 8 个除数中选择一个，用作 TPM 时钟输入，如表 B-3 所示。完成时钟源同步或选定时钟源后，这个预分频器的位置也就确定了，因此无论选择什么时钟源驱动 TPM 系统都会产生影响。

表 B-2. TPM 时钟源选择

CLKSB:CLKSA	连接预分频器输入的 TPM 时钟源
0:0	未选中任何时钟 (TPMx 禁止)
0:1	线速率时钟 (BUSCLK)
1:0	固定系统时钟 (XCLK)
1:1	外部源 (TPMxCLK) <sup>1,2</sup>

<sup>1</sup> 外部时钟的最大允许频率是总线频率的 1/4。

<sup>2</sup> 如果外部时钟输入共享通道 n，并且选用未 TPM 时钟源，对应的 ELSnB:ELSnA 控制位应当设为 0:0，这样通道 n 就不会使用相同管脚，从而避免了冲突。

表 B-3. 预分频器除数选择

PS2:PS1:PS0	TPM 时钟源除以
0:0:0	1
0:0:1	2
0:1:0	4
0:1:1	8
1:0:0	16
1:0:1	32
1:1:0	64
1:1:1	128

## B.5.2 定时器计数器寄存器 (TPMxCNTH:TPMxCNTL)

这两个只读 TPM 计数器寄存器包括 TPM 计数器数值的高字节和低字节。读取任何一个字节 (TPMxCNTH 或 TPMxCNTL) 都能将两个字节的内容锁定到缓冲器中。它们将在该缓冲器中保持锁定状态，直到另一个字节被读取为止。在 MCU 复位，在 TPMxCNTH 或 TPMxCNTL 中写入任意值或者定时器状态 / 控制寄存器 (TPMxSC) 进行任何写入操作时，一致性机制都会自动重启。

复位清除 TPM 计数器寄存器。

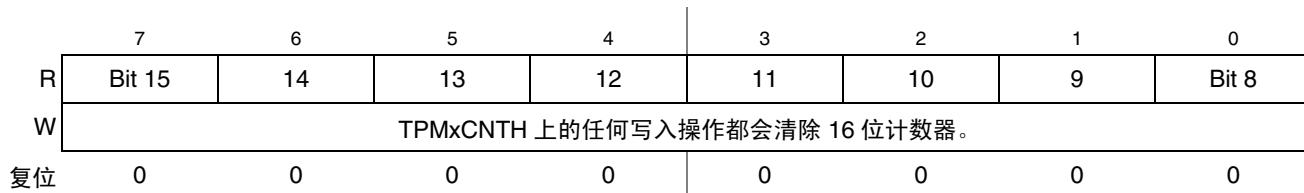


图 B-3. 定时器计数器寄存器高 (TPMxCNTH)

R	7	6	5	4	3	2	1	0
W	Bit 7	6	5	4	3	2	1	Bit 0
TPMxCNTL 上的任何写入操作都会清除 16 位计数器。								
复位	0	0	0	0	0	0	0	0

图 B-4. 定时器计数器寄存器低 (TPMxCNTL)

当背景模式处于活动状态时，定时器计数器和一致性机制被冻结，以便缓冲器锁定保持背景模式使能时的状态，无论背景模式使能时读取计数器的一个字节还是两个字节。

### B.5.3 定时器计数器模量寄存器 (TPMxMODH:TPMxMODL)

读写 TPM 模量寄存器包括 TPM 计数器使用的模量值。在 TPM 计数器达到模量值后，TPM 计数器在下个时钟 ( $CPWMS = 0$ ) 位置从 0x0000 重新计数，或者向下计数 ( $CPWMS = 1$ )，完成溢出标记 (TOF) 设置。写入 TPMxMODH 或 TPMxMODL 会抑制 TOF，溢出中断，直到书写另一个字节为止。复位操作会把 TPM 计数器模量寄存器设置为 0x0000，由此产生一个空运转定时器计数器（模量禁止）。

R	7	6	5	4	3	2	1	0
W	Bit 15	14	13	12	11	10	9	Bit 8
TPMxMODH 上的任何写入操作都会抑制溢出标记 (TOF)。								
复位	0	0	0	0	0	0	0	0

图 B-5. 定时器计数器模量寄存器高 (TPMxMODH)

R	7	6	5	4	3	2	1	0
W	Bit 7	6	5	4	3	2	1	Bit 0
TPMxMODL 上的任何写入操作都会抑制溢出标记 (TOF)。								
复位	0	0	0	0	0	0	0	0

图 B-6. 定时器计数器模量寄存器低 (TPMxMODL)

最好的方法是等待溢出中断，这样模量寄存器的两个字节都可以在发生新溢出之前写入。另一种方法是在写入 TPM 模量寄存器前复位 TPM 计数器，避免第一个计数器溢出时发生混淆。

## B.5.4 定时器通道 n 状态和控制寄存器 (TPMxCnSC)

TPMxCnSC 包含通道中断状态标记和控制位，用于配置中断启动、通道配置和管脚功能。

	7	6	5	4	3	2	1	0
R W	CHnF	CHnIE	MSnB	MSnA	ELSnB	ELSnA	0	0
复位	0	0	0	0	0	0	0	0
			= 未实施或预留					

图 B-7. 定时器通道 n 状态和控制寄存器 (TPMxCnSC)

表 B-4. TPMxCnSC 寄存器字段描述

字段	描述
7 CHnF	<b>通道 n 标记</b> — 通道 n 配置用于输入捕获的情况下，通道 n 管脚上发生活动边时设置该标记位。通道 n 为输出对比或边缘对齐 PWM 通道时，TPM 计数器寄存器中的值与 TPM 通道 n 值寄存器中的值匹配时，CHnF 被设置。此标记很少与中央对齐 PWM 一起使用，因为它是每次计数器与通道值寄存器（与活动工作循环周期的两个边相对应）相匹配时设置的。 当 CHnF 被设置且中断 (CHnIE = 1) 被启动时，会请求相应的中断。在 CHnF 被设置时，您可以通过读取 TPMxCnSC 并将一个 0 写入 CHnF 中来清除 CHnF。如果清除序列完成前出现另一个中断请求，则序列将被复位，因此早先 CHnF 的清除序列完成后 CHnF 仍将被设置。这样做的目的是确保清除以前的 CHnF 不会导致 CHnF 中断请求的丢失。复位可清除 CHnF 位。将 1 写入 CHnF 是无效的。 0 通道 n 上没有输入捕获或输出对比事件 1 通道 n 上发生输入捕获或输出对比事件
6 CHnIE	<b>通道 n 中断使能</b> — 这个读 / 写位从通道 n 中启动中断。复位可清除 CHnIE。 1 通道 n 中断请求使能
5 MSnB	<b>TPM 通道 n 的模式选择 B</b> — 当 CPWMS=0 时，MSnB=1 为边缘对齐 PWM 模式配置 TPM 通道 n。请参考表 B-5 中来查看通道模式和设置控制总结。
4 MSnA	<b>TPM 通道 n 的模式选择 A</b> — 当 CPWMS=0 而 MSnB=0 时，MSnA 为输入捕获模式或输出对比模式配置 TPM 通道 n。请参考表 B-5，查看通道模式和设置控制总结。
3:2 ELSn[B:A]	<b>边缘 / 电平选择位</b> — 根据 CPWMS:MSnB:MSnA 设置的定时器通道运行模式（如表 B-5 所示），这些位选择触发输入捕获事件的输入边极性，选择根据输出对比匹配将驱动的电平，或者选择 PWM 输出的极性。  将 ELSnB:ELSnA 设置为 0:0 可把相关定时器管脚配置为与任何定时器通道功能无关的通用输入 / 输出管脚。当相关定时器通道被设置为不要求使用管脚的软件定时器时，本功能常用于临时关闭输入捕获通道或允许将定时器管脚用作通用输入 / 输出管脚。

表 B-5. 模式、边和电平选择

CPWMS	MSnB:MSnA	ELSnB:ELSnA	模式	配置
X	XX	00	不用于 TPM 通道的管脚；作为 TPM 的外部时钟使用或恢复为通用输入 / 输出	
0	00	01	输入捕获	仅在上升边捕获
		10		仅在下降边捕获
		11		在上升或下降边捕获
	01	00	输出对比	仅对比如软件
		01		切换对比输出
		10		清除对比输出
		11		设置对比输出
	1X	10	边缘对齐 PWM	高保真脉冲（清除对比输出）
		X1		低保真脉冲（设置对比输出）
1	XX	10	中央对齐 PWM	高保真脉冲（清除向上对比输出）
		X1		低保真脉冲（设置向上对比输出）

如果相应的端口管脚在改变成输入捕获模式前至少在两个总线时钟周期内不稳定，系统可能会提供边沿触发器的意外指示。一般，在改变通道配置位之后和启动通道中断之前，程序会清除状态标记，或使用状态标记避免任何意外行为。。

### B.5.5 TPM 通道值寄存器 (TPMxChVH:TPMxChVL)

这些读 / 写寄存器包含输入捕获功能捕获的 TPM 计数器值，或输出对比或 PWM 功能的输出对比值。通过复位可清除通道值寄存器。

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

图 B-8. 定时器通道值寄存器高 (TPMxChVH)

R	7	6	5	4	3	2	1	0
W	Bit 7	6	5	4	3	2	1	Bit 0
复位	0	0	0	0	0	0	0	0

图 B-9. 定时器通道值寄存器低 (TPMxChVL)

在输入捕获模式中，读取任何一个字节（无论是 TPMx $CnVH$  还是 TPMx $CnVL$ ）都会使两个字节的内容被锁入到缓冲器中。这些内容一直锁定在这个缓冲器中，直到另一个字节被读取。当 TPMx $CnSC$  寄存器被写入时，锁存机制可复位（变为未锁存状态）。

在输出对比或 PWM 模式中，写入任何一个字节（无论是 TPMx $CnVH$  还是 TPMx $CnVL$ ）都会使该值被锁入到缓冲器中。两个字节都被写入后，它们会作为连贯的 16 位值传输到定时器通道值寄存器中。这一锁存机制可以通过写入 TPMx $CnSC$  寄存器来人工进行复位。

这种锁定机制允许以任何顺序进行连贯的 16 位写入，这对各种编译器实施方案都很友好。

## B.6 功能介绍

所有 TPM 功能都与允许灵活选择时钟源和预分频器的 16 位主计数器相关。此外，16 位模数寄存器还与 TPM 中的 16 位主计数器相关。每个 TPM 通道可与 MCU 管脚及可屏蔽的中断功能相关。

TPM 具有中央对齐的功能（由 TPMx $SC$  中 CPWMS 控制位控制）。当 CPWMS 被设置为 1 时，定时器计数器 TPMxCNT 改变为向上 / 向下计数器并且相关 TPM 中的所有通道都作为中央对齐的 PWM 通道。当 CPWMS=0 时，每个通道可独立配置，以便以输入捕获、输出对比或缓冲的边缘对齐 PWM 模式运行。

后面各小节介绍 16 位主计数器和计数器的每种运行模式（输入捕获、输出对比、边缘对齐 PWM 和中央对齐 PWM）。因为管脚运行和中断活动的细节取决于操作模式，这些主题将在相关模式的章节中介绍。

### B.6.1 计数器

所有定时器功能都基于 16 位主计数器（TPMxCNTH:TPMxCNTL）。本小节讨论时钟源选择、向上计数和向下计数、计数结束溢出和手动计数器复位。

在任何 MCU 复位后，CLKSB:CLKSA = 0:0，所以没有选择时钟源，并且 TPM 是不活动的。正常情况下，CLKSB:CLKSA 将设置为 0:1，使总线时钟驱动定时器计数器。TPM 的时钟源可以选为关闭、总线时钟（BUSCLK）、固定系统时钟（XCLK）或外部输入。外部时钟方法的最大允许频率为总线速率的 1/4。参见 B.5.1，“定时器状态和控制寄存器 (TPMx $SC$ )”及表 B-2 来了解有关时钟源的更多信息。

当微控制器处于活动后台模式时，TPM 会临时挂起所有计数，直到微控制器返回到正常用户操作模式。在停止模式下，所有 TPM 时钟被停止；因此在时钟恢复前，TPM 一直被有效地关闭。在等待模式期间，TPM 继续正常运行。

16 位主计数器有两种计数模式。选择中央对齐 PWM 时（CPWMS = 1），计数器以向上 / 向下计数模式运行。否则，计数器作为简单的向上计数器运行。用作向上计数器时，16 位主计数器从 0x0000 开始计数，直到终端计数，然后重新从 0x0000 开始。最大计数为 0xFFFF 或 TPMxMODH:TPMxMODL 中的模数值。

当规定了中央对齐 PWM 操作时，计数器从 0x0000 向上计数，直到达到终端计数，然后向下计数到 0x0000，再从这里向上计数。0x0000 和终端计数值（TPMxMODH:TPMxMODL 中的值）为正常长度计数（一个定时器时钟周期长度）。

中断标记和启动与 16 位主计数器相关。定时器溢出标记（TOF）是一种显示定时器计数器溢出的软件可接入指示。TOF 标记等于 1 时自动生成静态硬件中断的情况下，启动信号都在软件轮询（TOIE=0）（无硬件中断生成）或中断驱动操作（TOIE=1）之间选择。

导致 TOF 被设置的条件取决于（向上或向上 / 向下）计数模式。在向上计数模式中，16 位主计数器从 0x0000 计数到 0xFFFF，并且在下一次计数时钟中溢出到 0x0000。在从 0xFFFF 向 0x0000 过渡时，TOF 被设置。设置了模数限制时，TOF 会在从模数寄存器中设置的值向 0x0000 过渡时设置。当 16 位主计数器以向上 / 向下模式运行的情况下，计数器在从模数寄存器中设置的值和下一个更低的计数值过渡而改变方向时 TOF 标记被设置。这对应 PWM 周期的结束。（0x0000 计算值对应周期的中央。）

因为 HCS08 MCU 是一种 8 位的架构，所以一致性机制被设计到定时器计数器中以进行读取操作。当计数器的任何一个字节（TPMxCNTH or TPMxCNTL）被读取时，两个字节都被捕获到缓冲器中，这样当另一个字节被读取时，值会显示读取第一个字节时计数的另一个字节。计数器继续正常计数，但是没有新的值从任意字节中读取，直到旧的计数的两个字节都被读取。

主定时器的计数器可随时通过将任何值写入 TPMxCNTH 或 TPMxCNTL 计数的任一比特来手动复位。如果在复位计数前只有计数器的一个字节被读取，那么这种计数器复位方式还会复位一致性机制。

## B.6.2 通道模式选择

如果 CPWMS=0（未规定中央对齐的 PWM 操作），那么通道 n 状态和控制寄存器中 MSnB 和 MSnA 控制位为相应通道确定基本运行模式。选择包括输入捕获、输出对比或缓冲的边缘对齐 PWM。

### B.6.2.1 输入捕获模式

利用输入捕获功能，TPM 可捕获外部事件发生的时间。当输入捕获通道的管脚上发生激活边时，TPM 会将 TPM 计数器的内容锁入到通道值寄存器中（TPMxCnVH:TPMxCnVL）。上升边、下降边或任何边可被选为触发输入捕获的活动边。

当 16 位捕获寄存器的任何一个字节被读取时，两个字节都被锁入到缓冲器中，以支持连贯的 16 位接入而不受顺序的影响。一致性序列可通过向通道状态 / 控制寄存器（TPMxCnSC）中写入值来手动复位。

输入捕获事件会设置标记位（CHnF），该标记可选择生成一个 CPU 中断请求。

### B.6.2.2 输出比较模式

通过输出比较功能，TPM 可生成具有可编程位置、极性、持续时间和频率的定时脉冲。当定时器达到输出对比通道的通道值寄存器中的值时，TPM 可以置 1，置 0，翻转引脚状态通道。

在输出比较模式中，值只有在 16 位寄存器的两个 8 位字节都被写入后被传输到相应定时器的通道寄存器中。这种一致性序列可通过向通道状态 / 控制寄存器（TPMxCnSC）中写入值来手动复位。

输出比较事件会设置标记位（CHnF），该标记可选择产生 CPU 中断请求。

### B.6.2.3 边缘对齐 PWM 模式

这类 PWM 输出使用定时器计数器的正常向上计数模式 (CPWMS=0)，而且可在相同 TPM 中的其他通道被配置为在输入捕获或输出对比时使用。这个 PWM 信号的周期由模数寄存器 (TPMxMODH:TPMxMODL) 中的设置确定。工作周期由定时器通道值 (TPMxCnVH:TPMxCnVL) 的设置确定。这个 PWM 信号的极性由 ELSnA 控制位的设置确定。0% 和 100% 工作周期都是可能的。

如图 B-10 所示，TPM 通道寄存器中的输出对比值决定 PWM 信号的脉冲宽度（工作周期）。模数溢出和输出对比之间的时间间隔就是脉冲宽度。如果 ELSnA=0，计数器溢出强制进入 PWM 信号高态；而输出对比强制进入 PWM 信号低态。如果 ELSnA=1，则计数器溢出强制进入 PWM 信号低态；而输出对比强制进入 PWM 信号高态。

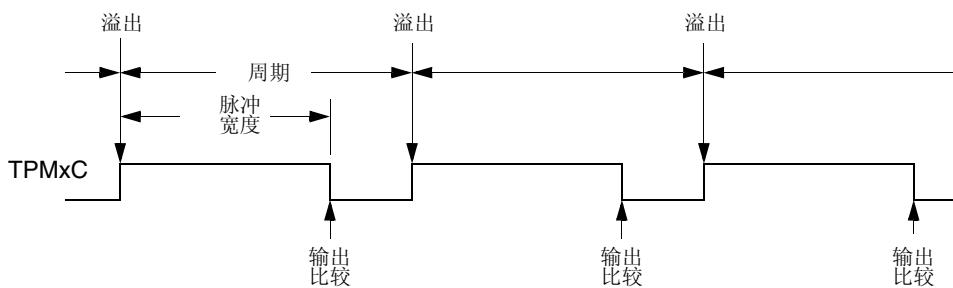


图 B-10. PWM 周期和脉冲宽度 (ELSnA = 0)

当通道值寄存器被设为 0x0000 时，工作周期为 0%。通过将定时器的通道值计数器 (TPMxCnVH:TPMxCnVL) 设为大于模数设置的值，可实现 100% 的工作周期。这意味着模数设置必须小于 0xFFFF 才能实现 100% 的工作周期。

HCS08 是一个 8 位 MCU 系列，定时器通道寄存器中的设置被缓冲，以确保连贯的 16 位更新并避免意外的 PWM 脉冲宽度。将值写入到任一寄存器 (TPMxCnVH 或 TPMxCnVL) 中也就是写入到缓冲器寄存器中。在边缘对齐 PWM 模式中，只有在 16 位寄存器的两个 8 位字节都被写入并且 TPMxCNTH:TPMxCNTL 计数器中的值为 0x0000 时，值才会被发送到相应定时器通道寄存器中。（新的工作周期直到下一个完全周期才生效）

### B.6.3 中央对齐 PWM 模式

这类 PWM 输出使用定时器计数器的向上 / 向下计数模式 (CPWMS = 1)。TPMxCnVH:TPMxCnVL 中的输出比较值决定 PWM 信号的脉冲带宽（工作周期），而周期是由 TPMxMODH:TPMxMODL 中的值决定的。TPMxMODH:TPMxMODL 应保持在 0x0001 至 0x7FFF 之间的范围内，因为这一范围以外的值可能会导致模糊结果。ELSnA 将决定 CPWM 输出的极性。

$$\text{脉冲宽度} = 2 \times (\text{TPMxCnVH:TPMxCnVL}) \quad \text{等式 17-1}$$

$$\begin{aligned} \text{周期} &= 2 \times (\text{TPMxMODH:TPMxMODL}); \\ \text{TPMxMODH:TPMxMODL} &= 0x0001-0x7FFF \end{aligned} \quad \text{等式 17-2}$$

如果通道值寄存器  $\text{TPMxChVH:TPMxChVL}$  为零或负数（位 15 被设置），工作周期将是 0%。如果  $\text{TPMxChVH:TPMxChVL}$  是正值（位 15 被清除）并大于（非零）模数设置，工作周期将为 100%，因为工作周期比较将不会发生。这意味着模数寄存器设置的可用周期范围为 0x0001 至 0x7FFE（如果不需要 100% 的工作周期，则为 0x7FFF）。这不是一个重要的限制条件，因为结果周期远远长于正常应用所需的周期。

$\text{TPMxMODH:TPMxMODL} = 0x0000$  是不应与中央对齐 PWM 模式一同使用的特例。当  $\text{CPWMS}=0$  时，这一情况与在 0x0000 和 0xFFFF 之间自由运行的计数器对应，然而当  $\text{CPWMS}=1$  时，计数器需要与 0x0000 以外的某个模数寄存器值有效匹配，以便将方向从向上计数改变为向下计数。

图 B-11 显示了 TPM 通道寄存器（乘以 2）中的输出比较值。该值决定 CPWM 信号的脉冲宽度（工作周期）。如果  $\text{ELSnA}=0$ ，向上计数时的比较匹配会强制 CPWM 输出信号降低；而向下计数时的比较匹配会强制输出升高。计数器向上计数，直到达到中的模数设置，然后向下计数，直到 0。这样就可将周期设置为  $\text{TPMxMODH:TPMxMODL}$  的两倍。

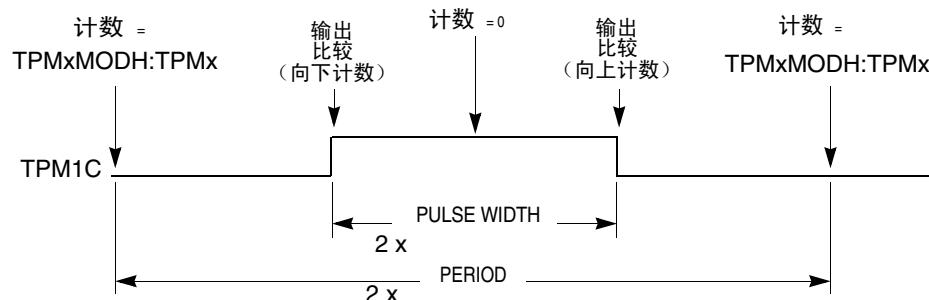


图 B-11. CPWM 周期和脉冲宽度 ( $\text{ELSnA} = 0$ )

中央对齐 PWM 生成的噪音一般比边缘对齐 PWM 小，因为相同系统时钟边上排列的输入 / 输出过渡更少。某些类型的电机也需要这类 PWM。

因为 HCS08 是一个 8 位 MCU 系列。定时器通道寄存器中的设置被缓存，以确保连贯的 16 位更新并避免意外的 PWM 脉冲宽度。写入到任何寄存器，不管是  $\text{TPMxMODH}$ 、 $\text{TPMxMODL}$ 、 $\text{TPMxChVH}$ 、还是  $\text{TPMxChVL}$ ，实际就是写入到缓冲器寄存器。只有在 16 位寄存器的两个 8 位字节都被写入而且定时器计数器溢出后（在模数寄存器上的终端计数结束后从向上计数改变为向下计数），值才被发送到相应的定时器通道寄存器中。这一  $\text{TPMxCNT}$  溢出要求只适用于 PWM 通道，而不是输出比较。

当  $\text{TPMxCNTH:TPMxCNTL} = \text{TPMxMODH:TPMxMODL}$  时，TPM 可在计数结束时生成一个 TOF 中断。用户可以选择重新上载任何数量的 PWM 缓冲器，并且它们可以在新的周期开始时同时更新。

$\text{TPMxSC}$  写入操作会取消写入到  $\text{TPMxMODH}$  and/or  $\text{TPMxMODL}$  中的任何值，并且为模数寄存器复位一致性机制。 $\text{TPMxChSC}$  写入操作会取消写入到通道值寄存器中的任何值，并且为  $\text{TPMxChVH:TPMxChVL}$  复位一致机制。

## B.7 TPM 中断

TPM 为主计数器溢出生成可选的中断，为每个通道生成一个中断。通道中断的意义取决于每个通道的运行模式。如果通道被配置用于输入捕获，所选的输入捕获边每次被识别时中断标记被设置。如果通道配置用于输出比较或 PWM 模式，中断标记会在每次主定时器计数器与 16 位通道值寄存器中的值匹配时被设置。参见[复位、中断和系统配置](#)一章了解绝对中断向量地址、优先级和本地中断掩码控制位。

对于 TPM 中的每个中断源，会在识别到中断条件（如定时器溢出、通道输入捕获或输出比较事件等）后设置标记位。这个标记可被软件读取（轮询）以确定操作已经发生，或者也可设置相关的启动位（TOIE 或 CHnIE）以启动硬件中断生成。中断启动位被设置时，相关中断标记等于 1 时会生成静态中断。从中断服务程序中返回前，用户软件必须执行一系列步骤来清除中断标记。

### B.7.1 清除定时器中断标记

TPM 中断标记通过两个步骤来清除：标记位被设置（1）时被读取，然后是向该位中写入一个 0。如果在这两步间检测到新事件，序列被复位，并且在第二步后中断标记仍被设置以避免错过新事件的可能性。

### B.7.2 定时器溢出中断描述

导致 TOF 被设置的条件取决于计数模式（向上或向上 / 向下）。在向上计数模式中，16 位定时器计数器从 0x0000 计数到 0xFFFF，然后在下一个计数时钟上溢出到 0x0000。在从 0xFFFF 过渡到 0x0000 时 TOF 被设置。设置了模数限制的情况下，TOF 标记会在从模数寄存器中设置的值过渡到 0x0000 时被设置。当计数器以向上 / 向下模式运行时，TOF 标记会在计数器从模数寄存器中设置的计数值和下一个更低计数值过渡而改变方向时被设置。这与 PWM 周期的结束对应（0x0000 计数值与周期中央对应）。

### B.7.3 通道事件中断描述

通道中断的含义取决于通道的当前模式（输入捕获、输出比较、边缘对齐 PWM 或中央对齐 PWM）。

当通道被配置为输入捕获通道时，ELSnB:ELSnA 控制位选择上升边、下降边、任何边或无边（关）作为触发输入捕获事件的边。检测到选定的边之后，中断标记被设置。标记通过[B.7.1，“清除定时器中断标记”](#)中所述的两步序列清除。

如果通道被配置为输出比较通道，每次主定时器计数器与通道值寄存器中的 16 位值匹配时会设置中断标记。标记通过[B.7.1，“清除定时器中断标记”](#)中所述的两步序列清除。

## B.7.4 PWM 占空比结束事件

对于配置用于 PWM 运行的通道，有两种可能性：

- 当通道配置用于边缘对齐 PWM 时，定时器计数器与标志活动工作周期结束的通道值寄存器匹配时，通道标记被设置。
- 当通道配置用于中央对齐 PWM 时，定时器计数与每个 PWM 周期的通道值寄存器的两倍相匹配。在这种 CPWM 情况下，通道标记会在定时器计数器与通道值寄存器匹配时，即在活动工作周期开始和结束时被设置。

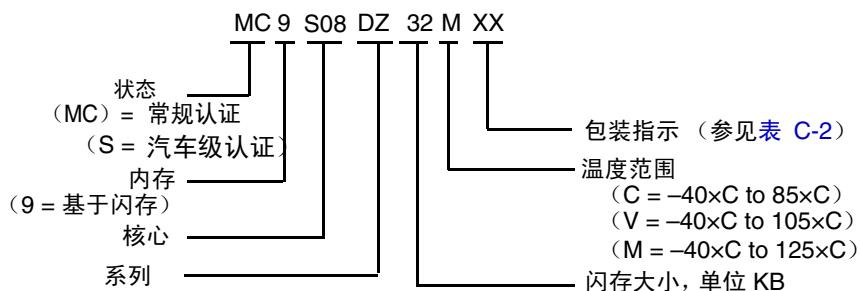
这种标记通过 [B.7.1，“清除定时器中断标记”](#) 中所述的两步序列清除。

# 附录 C 订购信息和机械图

## C.1 订购信息

本章包含 MC9S08DZ60 系列设备的订购信息。

设备编号体制举例：



### C.1.1 MC9S08DZ60 Series 设备

表 C-1. MC9S08DZ60 Series 设备

设备编号	内存			可供包装 <sup>1</sup>
	FLASH	RAM	EEPROM	
MC9S08DZ60	60,032	4096	2048	64-LQFP, 48-LQFP, 32-LQFP
MC9S08DZ48	49,152	3072	1536	
MC9S08DZ32	33,792	2048	1024	
MC9S08DZ16	16,896	1024	512	

<sup>1</sup> 包装信息请参见表 C-2。

## C.2 机械图

以下各页显示了下表中描述的包装的机械图：

表 C-2. Package 描述

Pin Count	典型值值 e	Abbreviation	Designator	Document No.
64	低 Quad Flat Package	LQFP	LH	98ASS23234W
48	低 Quad Flat Package	LQFP	LF	98ASH00962A
32	低 Quad Flat Package	LQFP	LC	98ASH70029A





## **联系我们：**

**主页：**  
[www.freescale.com](http://www.freescale.com)

**电子邮件：**  
[support@freescale.com](mailto:support@freescale.com)

### **美国 / 欧洲或未列出的地点：**

飞思卡尔半导体  
技术信息中心，CH370  
1300 N. Alma School Road  
Chandler, Arizona 85224  
+1-800-521-6274 or +1-480-768-2130  
[support@freescale.com](mailto:support@freescale.com)

**欧洲、中东和非洲：**  
Freescale Halbleiter Deutschland GmbH  
技术信息中心  
Schatzbogen 7  
81829 Muenchen, Germany  
+44 1296 380 456 (English)  
+46 8 52200080 (English)  
+49 89 92103 559 (German)  
+33 1 69 35 48 48 (French)  
[support@freescale.com](mailto:support@freescale.com)

**日本：**  
飞思卡尔半导体日本公司。  
总部  
ARCO Tower 15F  
1-8-1, Shimo-Meguro, Meguro-ku,  
Tokyo 153-0064  
Japan  
0120 191014 or +81 3 5437 9125  
[support.japan@freescale.com](mailto:support.japan@freescale.com)

**亚太地区：**  
飞思卡尔半导体香港公司  
技术信息中心  
2 Dai King Street  
Tai Po Industrial Estate  
Tai Po, N.T., Hong Kong  
+800 2666 8080  
[support.asia@freescale.com](mailto:support.asia@freescale.com)

**索取技术材料：**  
飞思卡尔半导体手册发布中心  
PO. Box 5405  
Denver, Colorado 80217  
1-800-441-2447 or 303-675-2140  
Fax: 303-675-2150  
[LDCFForFreescaleSemiconductor@hibbertgroup.com](mailto:LDCFForFreescaleSemiconductor@hibbertgroup.com)

Information in this document is provided solely to enable system and software implementers to use Freescale Semiconductor products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits or integrated circuits based on the information in this document.

Freescale Semiconductor reserves the right to make changes without further notice to any products herein. Freescale Semiconductor makes no warranty, representation or guarantee regarding the suitability of its products for any particular purpose, nor does Freescale Semiconductor assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. "Typical" parameters that may be provided in Freescale Semiconductor data sheets and/or specifications can and do vary in different applications and actual performance may vary over time. All operating parameters, including "Typicals", must be validated for each customer application by customer's technical experts. Freescale Semiconductor does not convey any license under its patent rights nor the rights of others. Freescale Semiconductor products are not designed, intended, or authorized for use as components in systems intended for surgical implant into the body, or other applications intended to support or sustain life, or for any other application in which the failure of the Freescale Semiconductor product could create a situation where personal injury or death may occur. Should Buyer purchase or use Freescale Semiconductor products for any such unintended or unauthorized application, Buyer shall indemnify and hold Freescale Semiconductor and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs, damages, and expenses, and reasonable attorney fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use, even if such claim alleges that Freescale Semiconductor was negligent regarding the design or manufacture of the part.

RoHS-compliant and/or Pb-free versions of Freescale products have the functionality and electrical characteristics as their non-RoHS-compliant and/or non-Pb-free counterparts. For further information, see <http://www.freescale.com> or contact your Freescale sales representative.

For information on Freescale's Environmental Products program, go to <http://www.freescale.com/epp>.

Freescale™ and the Freescale logo are trademarks of Freescale Semiconductor, Inc. All other product or service names are the property of their respective owners.  
© Freescale Semiconductor, Inc. 2007. All rights reserved.

