Errata

TMS320F28003x 实时 MCU 器件勘误表
器件修订版本 0

TEXAS INSTRUMENTS

摘要

本文档介绍了功能规格的已知例外情况（公告）。本文档也包含了使用说明。在使用说明中介绍了器件行为可能与假定或记录的行为不匹配的情况。这可能包括影响器件性能或功能正确性的行为。

内容

1 使用说明和公告模型.....2
1.1 使用说明汇总表.....2
1.2 公告汇总表.....2
2 命名规则、封装编号法和修订版本标识.....3
2.1 器件和开发支持工具命名规则.....3
2.2 支持的器件.....3
2.3 封装编号法和修订版本标识.....4
3 器件修订版本 0 使用说明和公告.....7
3.1 器件修订版本 0 使用说明.....7
3.2 器件修订版本 0 公告.....10
4 文档支持.....39
5 商标.....39
6 修订历史记录.....39

插图清单

图 2-1. PZ 封装的封装编号法.....4
图 2-2. PN 封装的封装编号法.....4
图 2-3. PM 封装的封装编号法.....5
图 2-4. PT 封装的封装编号法.....5
图 3-1. 具有 AGPIO 和 AIO 模拟引脚类型的模拟子系统图.....17
图 3-2. 不良跳闸事件和消隐窗口过期.....20
图 3-3. 可能会产生不良的 ePWM 输出.....20
图 3-4. 流水线中没有停滞时的问题流水线图.....23
图 3-5. 指令 I1 的 E3 时隙中存在停滞时的问题流水线图.....24
图 3-6. 带解决方法的流水线图.....25

表格清单

表 1-1. 使用说明汇总表.....2
表 1-2. 公告汇总表.....2
表 2-1. 版本标识.....6
表 3-1. ADCCTL2 寄存器.....11
表 3-2. 特定模拟输入引脚的用例组合.....17
表 3-3. C2000 作为带有标准模式主机的目标发送器时的数据上升时间要求.....26
表 3-4. 公共总线电容 (C_b) 的上拉电阻 (R_p) 值.....27
表 3-5. 受公告影响的存储器.....29

1 使用说明和公告模型

表 1-1 列出了所有使用说明和适用的器件修订版本。表 1-2 列出了所有公告、受影响的模块以及适用的器件修订版本。

1.1 使用说明汇总表

表 1-1. 使用说明汇总表

编号	标题	受影响的器件修订版本
		0
节 3.1.1	PIE：背对背 PIEACK 写入和手动 CPU 中断屏蔽清除之后的伪波嵌套中断	是
节 3.1.2	将嵌套中断与重复块一起使用时的注意事项	是
节 3.1.4	安全性：主要的防御层是构建芯片安全边界，从启用 JTAGLOCK 和零引脚引导至闪存功能开始	是
节 3.1.3	I2C 从机模式：在中断响应时间较长的应用中管理状态标志	是

1.2 公告汇总表

表 1-2. 公告汇总表

模块	说明	受影响的器件修订版本
		0
ADC	ADC：如果未设置 INTxCONT（继续中断模式），中断可能会停止	是
ADC	ADC：使用 ADCCLK 小数分频器时 ADC 性能下降	是
ADC	ADC：DMA 读取陈旧结果	是
模拟子系统	模拟子系统：对 ADCDACLOOPBACK 寄存器第 1 位的写入不起作用	是
BOR	BOR：2.45V 至 3.0V 之间的 VDDIO 可产生多个 XRSn 脉冲	是
C28	C28：快速整数除法 (FID) 和 DIVF64 结果	是
CMPSS	CMPSS：在某些情况下，COMPxLATCH 可能无法正确清除	是
CMPSS	CMPSS：如果比较器输入引脚具有 AGPIO 功能并且 ADC 正在对输入引脚进行采样，则可能会发生 CMPSS 干扰	是
DCAN	DCAN：在 DCAN FIFO 模式下，接收到的报文在 FIFO 缓冲区中可能未按顺序存放	是
MCAN	MCAN：当从配置有相同报文 ID 的专用 Tx 缓冲区发送时，报文顺序反转	是
ePWM	ePWM：如果跳闸在消隐窗口结束时保持活动状态，则可能会发生 ePWM 干扰	是
ePWM	ePWM 单次/CBC 跳闸事件 DCxEVTy.force 不会设置跳闸条件	是
ePWM	ePWM：在消隐窗口启动后的前 3 个周期内，消隐窗口不会过滤跳闸事件	是
eQEP	eQEP：索引期间方向变化时位置计数器复位错误	是
FPU	FPU：FPU 至 CPU 寄存器移动操作之前是任何 FPU 2p 操作	是
I2C	I2C：目标发送器模式、标准模式 SDA 时序限制	是
LIN	LIN：当同步字段错误时，同步字段错误 (ISFE) 标志/中断未设置	是
存储器	存储器：在有效存储器之外进行预取	是
引导 ROM	引导 ROM：TMX 器件上 MCAN 引导加载程序的数据溢出	是
引导 ROM	安全实时固件更新 (LFU) 引导模式已弃用	是
闪存	闪存：在未禁用闪存预取的情况下执行 Fapi_setActiveFlashBank () 可能导致 ITRAP	是
系统	系统：在 Pagesel=0 模式下，HIC 非法读取错误标志不会被置位	是
系统	系统：多次连续写入 CLKSRCCTL1 可能导致系统挂起	是
诊断	在使用 HWBIST 时避免杂散中断	是
PLL	PLL 基准时钟丢失检测：时钟丢失标志可能被错误激活	是
SDFM	SDFM：动态更改阈值设置（LLT、HLT）、滤波器类型或 COSR 设置将触发虚假比较器事件	是
SDFM	SDFM：动态更改数据滤波器设置（例如滤波器类型或 DOSR）将触发虚假数据响应事件	是
SDFM	SDFM：在三个 SD 调制器时钟周期内对 SDCPARMx 寄存器位字段 CEVT1SEL、CEVT2SEL 和 HZEN 进行两次背对背写入可能会损坏 SDFM 状态机，从而导致虚假比较器事件	是
看门狗	看门狗：WDKEY 寄存器不受 EALLOW 保护	是

2 命名规则、封装编号法和修订版本标识

2.1 器件和开发支持工具命名规则

为了标示产品开发周期所处的阶段，TI 为所有 DSP 器件和支持工具的器件型号分配了前缀。每个 DSP 商用产品系列成员都具有以下三个前缀之一：TMX、TMP 或 TMS（例如，TMS320F280039C）。德州仪器 (TI) 为其支持工具推荐使用三种可能的前缀指示符中的两个：TMDX 和 TMDS。这些前缀代表了产品开发的发展阶段，即从工程原型（TMX 和 TMDX）直到完全合格的生产器件和工具（TMS 和 TMDS）。

器件开发演变流程：

TMX 试验器件不一定代表最终器件的电气规范标准，并且可能不使用生产组装流程。

TMP 原型器件不一定是最终器件模型，并且不一定符合最终电气标准规范。

TMS 完全合格的芯片模型的生产版本。

支持工具开发演变流程：

TMDX 还未经德州仪器 (TI) 完整内部质量测试的开发支持产品。

TMDS 完全合格的开发支持产品。

TMX 和 TMP 器件和 TMDX 开发支持工具供货时附带如下免责条款：

“开发的产品用于内部评估用途。”

生产器件和 TMDS 开发支持工具已进行完全特性描述，并且器件的质量和可靠性已经完全论证。TI 的标准保修证书适用。

预测显示原型器件（X 或者 P）的故障率大于标准生产器件。由于这些器件的预期最终使用故障率仍未确定，故德州仪器 (TI) 建议请勿将这些器件用于任何生产系统。请仅使用合格的生产器件。

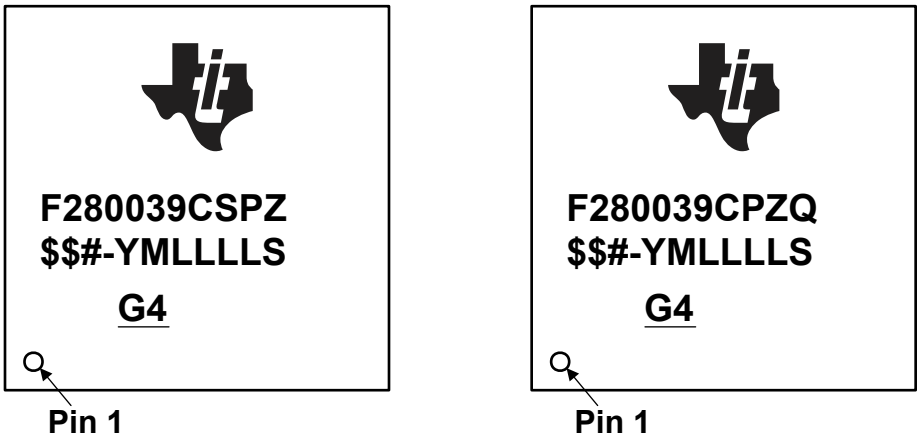
2.2 支持的器件

本文档支持以下器件：

- TMS320F280039C
- TMS320F280039C-Q1
- TMS320F280039
- TMS320F280039-Q1
- TMS320F280038C-Q1
- TMS320F280038-Q1
- TMS320F280037C
- TMS320F280037C-Q1
- TMS320F280037
- TMS320F280037-Q1
- TMS320F280036C-Q1
- TMS320F280036-Q1
- TMS320F280034
- TMS320F280034-Q1
- TMS320F280033
- TMS320F280033-Q1

2.3 封装编号法和修订版本标识

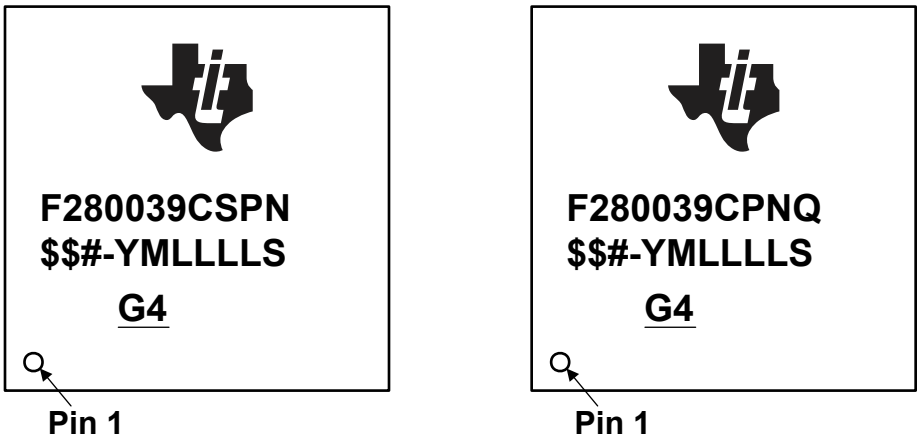
图 2-1、图 2-2、图 2-3 和图 2-4 展示了封装编号法。表 2-1 列出了器件修订版本代码。



$$ = Wafer Fab Code (one or two characters)
# = Silicon Revision Code
YM = 2-digit Year/Month Code
LLLL = Assembly Lot Code
S = Assembly Site Code per QSS 005-120

G4 = Green (Low Halogen and RoHS-compliant)

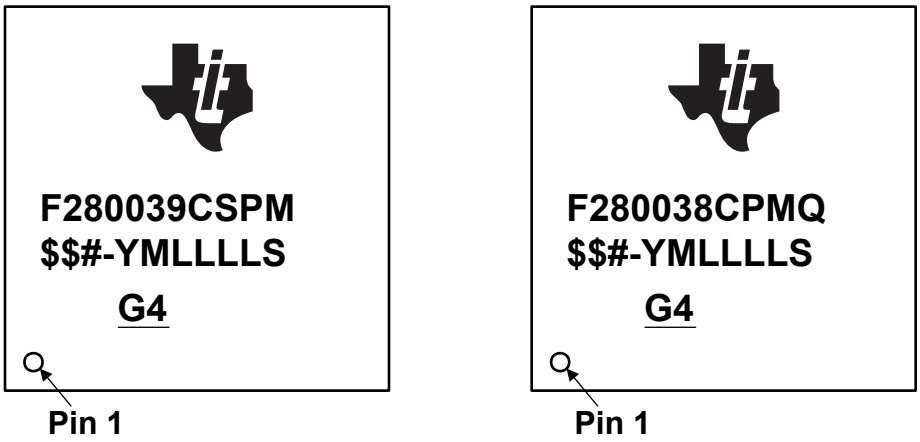
图 2-1. PZ 封装的封装编号法



$$ = Wafer Fab Code (one or two characters)
# = Silicon Revision Code
YM = 2-digit Year/Month Code
LLLL = Assembly Lot Code
S = Assembly Site Code per QSS 005-120

G4 = Green (Low Halogen and RoHS-compliant)

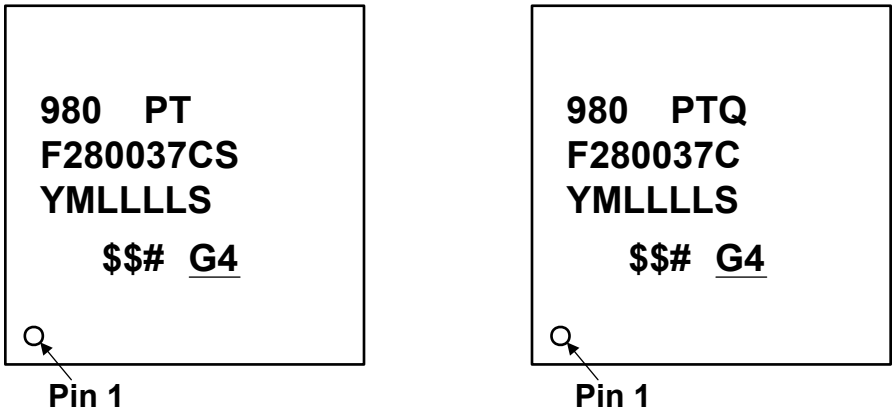
图 2-2. PN 封装的封装编号法



$$ = Wafer Fab Code (one or two characters)
# = Silicon Revision Code
YM = 2-digit Year/Month Code
LLLL = Assembly Lot Code
S = Assembly Site Code per QSS 005-120

G4 = Green (Low Halogen and RoHS-compliant)

图 2-3. PM 封装的封装编号法



980 = TI EIA Code
YM = 2-digit Year/Month Code
LLLL = Assembly Lot Code
S = Assembly Site Code per QSS 005-120
$$ = Wafer Fab Code (one or two characters)
= Silicon Revision Code

G4 = Green (Low Halogen and RoHS-compliant)

图 2-4. PT 封装的封装编号法

表 2-1. 版本标识

器件修订版本代码	器件修订版本	REVID(1) 地址：0x5D00C	注释(2)
空白	0	0x0000 0000	该器件修订版本的代码为 TMX 和 TMS。

(1) 器件修订版本 ID
(2) 有关可订购器件型号，请参阅 TMS320F28003x *实时微控制器数据表中*的“封装信息”表。

3 器件修订版本 0 使用说明和公告

3.1 器件修订版本 0 使用说明

本节列出了适用于器件修订版本 0 的所有使用说明。

3.1.1 PIE：背对背 PIEACK 写入和手动 CPU 中断屏蔽清除之后的伪波嵌套中断

受影响修订版：0

某些用于嵌套中断的代码序列允许 CPU 和 PIE 进入不一致状态，从而触发不必要的中断。进入该状态所需的条件为：

1. PIEACK 清除后，立即执行全局中断使能（EINT 或 ASM (" CLRC INTM") ）。
2. 嵌套中断会清除其组的一个或多个 PIEIER 位。

是否触发不必要的中断，取决于系统中其他中断的配置和时序。在大多数应用中，预计这种事件很罕见或根本不存在。如果发生这种情况，不必要的中断将是嵌套中断的 PIE 组中的第一个中断、并将在嵌套中断重新启用 CPU 中断（EINT 或 asm ("CLRC INTM") ）后触发。

解决方法：在 PIEACK 写入和 CPU 中断使能之间添加一个 NOP（无操作）。以下显示了示例代码。

```
//Bad interrupt nesting code
PieCtrlRegs.PIEACK.all = 0xFFFF;      //Enable nesting in the PIE
EINT;                                 //Enable nesting in the CPU

//Good interrupt nesting code
PieCtrlRegs.PIEACK.all = 0xFFFF;      //Enable nesting in the PIE
asm(" NOP");                          //Wait for PIEACK to exit the pipeline
EINT;                                 //Enable nesting in the CPU
```

3.1.2 将嵌套中断与重复块一起使用时的注意事项

受影响修订版：0

如果用户为了使用嵌套特性而在中断处理例程 (ISR) 内使用 EINT 指令启用中断，那么用户必须在退出 ISR 之前通过使用 DINT 汇编指令来禁用中断。否则可能会导致 RB 寄存器中的位无法正确恢复，从而导致异常代码行为。

如果应用程序中未使用 RPTB ASM 指令，则没有问题。对于 C 语言源代码，需要对生成的反汇编代码进行分析以验证是否发生此情况。

如果 ISR 用 C 语言编码，那么 C28x C 编译器可以处理上述问题，无需执行任何操作。如果 ISR 用 C28x 汇编语言编码，则必须遵循上述指南。

备注

CGT v15.12.2.LTS（2016 年 4 月发布）或更高版本的 CGT 封装会自动满足此要求。只有早期版本的 CGT 工具才需要添加 DINT。

3.1.3 I2C 从机模式：在中断响应时间较长的应用中管理状态标志

受影响修订版：0

如果应用在 I2C 总线活动继续进行时延迟处理 I2C 中断，后续总线事件可能导致 I2C 状态寄存器 (I2CSTR) 中的多个状态标志同时变为活动状态。如果 I2CIER 中启用了这些中断事件中的多个，后续中断可能会丢失。

- 以 I2C 主机模式工作时，此问题不适用。
- 如果每个 I2CSTR 标志在另一个标志置位之前已由软件清除，则此问题不适用。

解决方法：轮询 I2CISRC.INTCODE 字段，并针对每个中断源执行相应操作，直到 I2CISRC.INTCODE 字段的值为 0（无挂起中断）。

```
__interrupt void I2CA_ISR(void){
    uint16_t isrc;
    while((isrc = I2C_getInterruptSource(I2CA_BASE)) != I2C_INTSRC_NONE){
        switch(isrc) {
            case I2C_INTSRC_ARB_LOST:
                    /* ARBLINT ISR code */
                    break;
            case I2C_INTSRC_NO_ACK:
                    /* NACKINT ISR code */
                    break;
            case I2C_INTSRC_REG_ACCESS_RDY:
                    /* ARDYINT ISR code */
                    I2C_clearStatus(I2CA_BASE, I2C_STS_REG_ACCESS_RDY);
                    break;
            case I2C_INTSRC_RX_DATA_RDY:
                    /* RRDYINT ISR code, including I2C_getData() call */
                    break;
            case I2C_INTSRC_TX_DATA_RDY:
                    /* XRDYINT ISR code, including I2C_putData() call */
                    break;
            case I2C_INTSRC_STOP_CONDITION:
                    /* SCDINT ISR code */
                    break;
            case I2C_INTSRC_ADDR_TARGET:
                    /* AASINT ISR code */
                    break;
        }
        NOP;
        NOP;
    }
    Interrupt_clearACKGroup();
}
```

3.1.4 安全性：主要的防御层是构建芯片安全边界，从启用 JTAGLOCK 和零引脚引导至闪存功能开始

受影响的版本：0

在任何情况下都不允许未经授权的代码进入器件并执行，这是器件安全的前提条件。为此，器件提供了两项功能，担心安全性的用户应该始终启用这两项功能。

- **JTAGLOCK**

  当在闪存的 USER OTP 区域中启用时，JTAGLOCK 功能会禁用对器件的资源进 JTAG 访问（例如调试器连接），阻止未经授权的一方使用 JTAG 接口将任何代码下载到器件中。启用 JTAGLOCK 后，用户仍然可以通过输入密码来允许授权方解锁，也可以通过将密码值全部设置为零来永久锁定。

- **零引脚引导至闪存**

  内置于 TI ROM 中的外部引导加载程序不会对所下载的代码执行任何身份验证。在 USER OTP 中启用零引脚引导选项以及闪存引导模式，则在引导时会阻止运行所有基于引脚的外部引导加载程序选项（例如 SCI、CAN、并行），这是因为在基本引导 ROM 执行结束后，会强制使引导过程立即跳转到内部闪存。为了保障最高的安全性，可选择安全闪存启动模式。这可以在跳转到闪存代码之前通过基本引导 ROM 对闪存代码进行预检查。

如果 JTAG 被永久锁定并且启用了零引脚引导至闪存选项，则通过 JTAG 或内置引导加载程序与器件进行通信的编程工具将失效。如果需要能够执行固件升级，则用户必须将代码预存储在闪存中以安全地管理和执行更新。

3.2 器件修订版本 0 公告

本节列出了适用于器件修订版本 0 的所有公告。

公告 ***ADC：如果未设置 INTxCONT（继续中断模式），中断可能会停止***

受影响版本

0

详细信息

在 ADCINTSELxNx[INTxCONT]= 0 时，设置 ADCINTFLG 后，中断将停止，并且不会发生其他 ADC 中断。

若在 ADCINTFLGCLR 寄存器进行软件写入的同时发生 ADC 中断，则 ADCINTFLG 将意外保持为设置状态，阻止将来发生 ADC 中断。

解决方法

1. 使用 “继续中断” 模式，则 ADCINTFLG 无法阻止其他 ADC 中断：

```
ADCINTSEL1N2[INT1CONT] = 1;
ADCINTSEL1N2[INT2CONT] = 1;
ADCINTSEL3N4[INT3CONT] = 1;
ADCINTSEL3N4[INT4CONT] = 1;
```

2. 为了避免发生这种情况，请确保下一次发生 ADC 中断之前，始终有足够的时间为 ADC ISR 提供服务并清除 ADCINTFLG。
3. 清除 ADCINTFLG 时，请检查 ISR 中是否存在溢出情况。在写入到 ADCINTFLGCLR 后立即检查 ADCINTOVF；如果已设置，则再次写入 ADCINTFLGCLR 以确保 ADCINTFLG 已被清除。若再设置 ADCINTOVF 寄存器，则表示已丢失 ADC 转换中断。

```
AdcaRegs.ADCINTFLGCLR.bit.ADCINT1 = 1;        //clear INT1 flag
if(1 == AdcaRegs.ADCINTOVF.bit.ADCINT1)       //ADCINT overflow
{
    AdcaRegs.ADCINTFLGCLR.bit.ADCINT1 = 1;    //clear INT1 again
// If the ADCINTOVF condition will be ignored by the application
// then clear the flag here by writing 1 to ADCINTOVFCLR.
// If there is a ADCINTOVF handling routine, then either insert
// that code and clear the ADCINTOVF flag here or do not clear
// the ADCINTOVF here so the external routine will detect the
// condition.
//    AdcaRegs.ADCINTOVFCLR.bit.ADCINT1 = 1;  // clear OVF
}
```

公告

ADC：使用 ADCCLK 小数分频器时 ADC 性能下降

受影响版本

0

详细信息

已证明使用分数 SYSCLK 转 ADCCLK 分频器（由 ADCCTL2.PRESCALE 字段控制）会导致该器件的 ADC 性能下降。请参阅表 3-1。

表 3-1. ADCCTL2 寄存器

性能下降			
位	**字段**	**值**	**说明**
3-0	预分频	0001	ADCCLK = SYSCLK/1.5
		0003	ADCCLK = SYSCLK/2.5
		...	
性能正常			
位	**字段**	**值**	**说明**
3-0	预分频	0000	ADCCLK = SYSCLK/1.0
		0002	ADCCLK = SYSCLK/2.0
		...	

解决方法

使用偶数预分频时钟分频器值。偶数预分频值会产生整数时钟分频，而不会影响 ADC 性能。

公告

ADC：DMA 读取陈旧结果

受影响版本

0

详细信息

ADCINT 标志可在 ADCRESULT 值锁存之前置位（请参阅 *TMS320F28003x 实时微控制器* 数据表 ADC 时序表中的 t_{LAT} 和 $t_{INT(LATE)}$ 列）。 DMA 可在 ADCINT 触发置位后仅 3 个周期即读取 ADCRESULT 值。因此，若满足以下所有条件，当用户期望获得最新结果时，DMA 可能读取到之前的 ADCRESULT 值：

- ADC 处于延迟中断模式。
- ADC 在 $t_{INT(LATE)}$ 比 t_{LAT} 提前 3 个或更多周期的模式下工作 (ADCCTL2 [PRESCALE] > 2)。
- DMA 由 ADCINT 信号触发。
- DMA 在未先读取任何其他值的情况下，立即读取与该 ADCINT 信号关联的 ADCRESULT 值。
- DMA 在收到 ADCINT 触发时处于空闲状态。

仅上述 DMA 读取可能导致读取到陈旧数据；以下非 DMA 方法将始终读取预期数据：

- ADCINT 标志触发 CLA 任务。
- ADCINT 标志触发 CPU ISR。
- CPU 轮询 ADCINT 标志。

解决方法

从 ADCINT 标志触发两个 DMA 通道。第一个通道作为虚拟事务。这将产生足够的延迟，使得第二个通道总能读取到最新的 ADC 结果。

公告	***模拟子系统：对 ADCDACLOOPBACK 寄存器第 1 位的写入不起作用***
受影响版本	0
详细信息	寄存器位 ADCDACLOOPBACK.ENLB2ADCB 不起作用，对该位的写入无效。因此，无法使用该寄存器激活 ADCB 的 ADC-DAC 环回功能。
解决方法	• 要将 ADC 功能作为诊断安全措施进行测试，请使用两个缓冲 DAC 中的一个输出已知电压，并使用 ADCB 对该 DAC 电压进行采样。DACA 的输出与通道 B15 在同一引脚上，DACB 的输出在 B7 上。 • 使用 INTERNALTEST/ADC 增益平衡机制对 0.9*VREFHI 进行采样。有关更多详细信息，请参阅 *TMS320F28003x 实时微控制器技术参考手册*中“模数转换器 (ADC)”一章的“内部测试模式”一节。

公告 **BOR：2.45V 至 3.0V 之间的 VDDIO 可产生多个 XRSn 脉冲**

受影响版本

0

详细信息

当 VDDIO 电源电压介于 2.45V 和 3.0V 之间时，BOR 会生成重复的 XRSn 置为有效和置为无效 。建议*不要*直接将 XRSn 引脚用作系统中任何其他器件的复位。

即使在这些 XRSn 脉冲发生时，F28003x BOR 也能有效地在内部将器件保持在已知的复位状态。在 VDDIO 供电电压升至 3.0 V 以上之前，该器件不会转移应用代码或引导加载程序，所有其他引脚都将保持其复位状态。

解决方法

1. 上电、断电和 BOR 事件期间，忽略额外的 XRSn 切换。额外的 XRSn 脉冲将不会影响 F28003x 器件本身的运行。
2. 如果 XRSn 脉冲会导致其他系统元件出现不良的系统行为，则请勿使用 XRSn 驱动其他器件。这些应用可使用外部电压监控器。
3. 对于需要在正常上电和断电期间避免这些脉冲的应用：
   a. 上电：请遵循 *TMS320F28003x 实时微控制器*数据表的建议运行条件表中的 SR_{SUPPLY} 要求；不会出现额外的 XRSn 低电平脉冲。
   b. 断电：为避免 XRSn 在断电期间失效，在对电源进行设计时应确保 VDDIO 在 25 µs 内支持 3.0V 至 2.45V 的电压范围 。如果 XRSn 上的电压上升是可以接受的，则可以计算 XRSn 上实现的 RC 电路的时间常数，以确保电压不会上升到系统指定的阈值以上。

公告 **C28：快速整数除法 (FID) 和 DIVF64 结果**

受影响版本

0

详细信息

使用 22.6.3.LTS 或 21.6.2.LTS 之前的编译器版本时，某些双周期 FPU 指令可能出现不正确的结果。

当以下受影响的 2p FPU 指令后跟任一 FID 或 DIVF64 指令且延迟槽少于两个时，可能发生寄存器损坏，从而导致结果错误。

在 FID 或 DIVF64 指令之前需要两个延迟槽的双周期 (2p) FPU 指令：

- ADDF32
- SUBF32
- MPYF32
- MACF32

快速整数除法 (FID) 指令：

- ABSI32DIV32
- ABSI32DIV32U
- ABSI64DIV32
- ABSI64DIV32U
- ABSI64DIV64
- ABSI64DIV64U
- ENEGI32DIV32
- ENEGI64DIV32
- ENEGI64DIV64
- MNEGI32DIV32
- MNEGI64DIV32
- MNEGI64DIV64
- NEGI32DIV32
- NEGI64DIV32
- NEGI64DIV64
- SUBC2UI64
- SUBC4UI32

DIVF64 指令：

- PREDIVF64
- SUBC3F64
- POSTDIVF64

问题示例：

可使用如下序列重现此问题：

```
ADDF32    R0H, R0H, #32768    ; 2p FPU instruction
NOP                           ; Only one delay slot (insufficient)
ABSI32DIV32 R2H, R1H, R3H     ; Fast Integer Division instruction
```

在这种情况下，R0H 中的值可能泄漏到 R1H，从而导致结果错误。添加一个额外的 NOP 可解决此问题：

```
ADDF32    R0H, R0H, #32768    ; 2p FPU instruction
NOP                           ; First delay slot
```

公告 (续)

C28：快速整数除法 (FID) 和 DIVF64 结果

```
NOP                          ; Second delay slot (required)
ABSI32DIV32 R2H, R1H, R3H    ; Fast Integer Division instruction
```

解决方法

1. 更新至已修复此问题的编译器版本 22.6.3.LTS、21.6.2.LTS 或更高版本。
2. 如果使用汇编代码，请确保受影响的指令组合之间有两个延迟槽。

公告 **CMPSS：在某些情况下，COMPxLATCH 可能无法正确清除**

受影响版本

0

详细信息

CMPSS 锁存路径旨在在本地锁存器 (COMPxLATCH) 中保持跳闸状态，直到由软件（通过 COMPSTSCLR）或 PWMSYNC 清除。

在信号经过数字滤波器数字化和鉴定后，COMPxLATCH 由比较器输出间接设置。比较器输出到达 COMPxLATCH 的预期最大延时可以用 CMPSS 模块时钟周期表示为：

LATENCY = 1 + (1 x FILTER_PRESCALE) + (FILTER_THRESH x FILTER_PRESCALE)

当 COMPxLATCH 由软件或 PWMSYNC 清除时，锁存本身根据需要清除，但 COMPxLATCH 之前的数据路径可能不会反映额外延时数的模块时钟周期的比较器输出值。如果在 COMPxLATCH 被清除时数字滤波器输出解析为逻辑 1，则锁存将在下一个时钟周期再次设置。

解决方法

在清除 COMPxLATCH 之前，让数字滤波器输出解析为逻辑 0。

如果软件清除了 COMPxLATCH，那么在清除锁存之前可以通过 COMPSTS 寄存器确认数字滤波器的输出状态。对于较大的延时值会产生不可容忍的延时的情况，可以通过重新初始化数字滤波器（通过 CTRIPxFILCTL）来刷新滤波器 FIFO。

如果 COMPxLATCH 被 PWMSYNC 清除，设计用户应用程序时应使得比较器跳闸条件在 PWMSYNC 生成前能至少清除延时周期。

公告

CMPSS：如果比较器输入引脚具有 AGPIO 功能并且 ADC 正在对输入引脚进行采样，则可能会发生 CMPSS 干扰

受影响版本

0

详细信息

需要特别注意特定模拟输入引脚的用例组合，如表 3-2 所示。如此表所示，对于 CMPSS 输入、ADC 采样和 AGPIO 的组合，需要使用特殊注意事项或权变措施。

表 3-2. 特定模拟输入引脚的用例组合

特定模拟引脚上使用的功能	使用的元件				
CMPSS 比较器输入	是	-	是	-	是
ADC 采样	是	是	-	是	是
AGPIO 模拟引脚类型	是	是	是	-	-
AIO 模拟引脚类型	-	-	-	是	是
结果	**需要权变措施**	**无需特殊分析或权变措施**			

AGPIO 模拟引脚路径包含一个额外的 53Ω 串联开关。这会创建一个由 ADC 和 CMPSS 比较器共享的低电容隔离式节点，如图 3-1 所示。当 ADC 对通道进行采样时，该节点可能会受到干扰（取决于 ADC 采样保持电容器上先前存储的电压），这种干扰可能会导致高达 50ns 的错误 CMPSS 事件。为了适应这种潜在的干扰，可以实施以下权变措施。

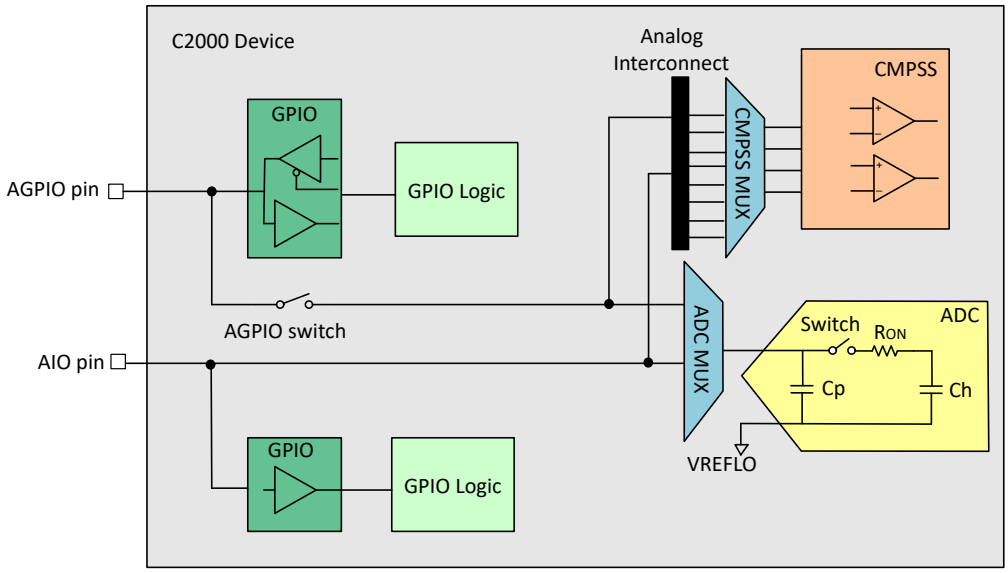


图 3-1. 具有 AGPIO 和 AIO 模拟引脚类型的模拟子系统图

应变方法

1. 对于同时需要 ADC 和 CMPSS 的模拟通道，使用不同的引脚（即 AIO 引脚类型）。
2. 使用设置为 50ns 或更大的 CMPSS 数字滤波器，从而将滤除临时干扰。
3. 预处理 ADC 的采样保持电容器，从而将使干扰不会导致误跳闸。例如，在读取受影响的通道之前，立即对 ADC 上不同通道的 3.3V 连接执行虚拟读取，从而使干扰为正向，远离误跳闸。如果误跳闸极性反转，则将使用 0V 信号的相反虚拟读取。

公告	***DCAN：在 DCAN FIFO 模式下，接收到的报文在 FIFO 缓冲区中可能未按顺序存放***
受影响版本	0
详细信息	在 DCAN FIFO 模式下，具有相同仲裁 ID 和掩码 ID 的接收报文应按照接收顺序存放在 FIFO 中。随后 CPU 通过 IF1/IF2 接口寄存器从 FIFO 中取回接收到的报文。某些报文可能未按接收顺序存放在 FIFO 中。如果报文顺序对应用程序处理至关重要，则此行为将妨碍 DCAN FIFO 模式的正常使用。
解决方法	使用 DMA 通过 IF3 寄存器读出 FIFO。每次报文被接收到 FIFO 时，数据也会被复制到 IF3 寄存器中，并向 DMA 模块生成 DMA 请求以读出数据。

公告	***MCAN：当从配置有相同报文 ID 的专用 Tx 缓冲区发送时，报文顺序反转***
受影响版本	0
详细信息	多个 Tx 缓冲区配置有相同的报文 ID。按升序顺序提出发送这些 Tx 缓冲区的请求，每个 Tx 请求之间会有一定延迟。根据各个 Tx 请求之间的延迟，Tx 缓冲区可能不会按 Tx 缓冲区编号的预期升序顺序进行发送。
解决方法	首先，将具有相同报文 ID 的一组 Tx 报文写入报文 RAM。然后，通过对 **TXBAR** 的一次写访问，同时请求发送所有这些报文。 使用 Tx FIFO 而不是专用 Tx 缓冲区来按特定顺序发送多条具有相同报文 ID 的报文。

公告 **ePWM：如果跳闸在消隐窗口结束时保持活动状态，则可能会发生 ePWM 干扰**

受影响版本

0

详细信息

消隐窗口通常用于屏蔽转换期间将导致系统误跳闸的任何 PWM 跳闸事件。如果在消隐窗口周期结束后，ePWM 跳闸事件在少于三个 ePWM 时钟内保持活动状态，则 ePWM 输出上可能会存在不良干扰脉冲。

图 3-2 展示了可能会导致不良 ePWM 输出的时间段。

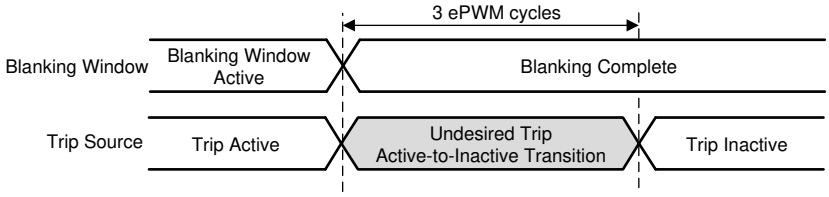


图 3-2. 不良跳闸事件和消隐窗口过期

图 3-3 展示了如果跳闸事件在消隐窗口关闭之前的 1 个周期或关闭之后的 3 个周期内结束，可能出现的两个 ePWM 输出。

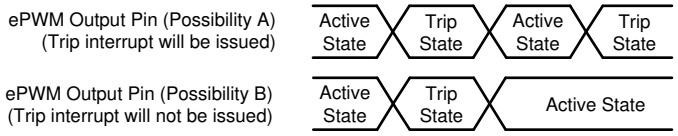


图 3-3. 可能会产生不良的 ePWM 输出

解决方法

扩展或减小消隐窗口，以避免任何不良的跳闸操作。

公告	***ePWM：ePWM 单次/CBC 跳闸事件 DCxEVTy.force 不设置跳闸条件***
受影响版本	0
详细信息	DCxEVTy.force 信号可用作单次跳闸事件和逐周期跳闸事件的输入。当设置 DCxCTL[EVT1/2FRCSYNCSEL] = 1（通过异步路径传递 DCxEVT 信号）时，无法正确设置单次或逐周期跳闸事件条件，并且会漏掉跳闸事件。
解决方法	对于具有 DCxCTL.EVT1/2FRCSYNCSEL 和 DCxCTL.EVT1/2LATSEL 的器件，存在两个选项。否则，对于单次或逐周期跳闸事件，同步路径是唯一可用的选项。 **异步路径**（通过异步路径传递 DCxEVT 信号） 要通过异步路径锁存单次或逐周期跳闸条件，必须进行以下配置： 要触发 PWM，可以使用 XBAR_invertEPWMSignal() 反转 EPWMXBAR 状态。 1. DCxCTL.EVT1/2FRCSYNCSEL = 1 2. DCxCTL.EVT1/2LATSEL = 1 **同步路径**（通过同步路径传递 DCxEVT 信号） 1. DCxCTL.EVT1/2FRCSYNCSEL = 0

公告	***ePWM：在消隐窗口启动后的前 3 个周期内，消隐窗口不会过滤跳闸事件***
受影响版本	0
详细信息	在启动消隐窗口后的前 3 个周期内，消隐窗口不会消隐跳闸事件。DCEVTFILT 可能继续反映 DCxEVTy 信号的变化。如果启用 DCEVTFILT，则可能会影响配置的后续子系统（例如跳闸区域子模块、TZ 中断、ADC SOC 或 PWM 输出）。
解决方法	在需要消隐之前的 3 个周期启动消隐窗口。如果在周期边界需要消隐窗口，则在下一周期开始前的 3 个周期启动消隐窗口。这是因为消隐窗口在整个期间边界中持续存在。

公告	***eQEP: 索引期间方向变化时位置计数器复位错误***
受影响版本	0
详细信息	在使用 PCRM = 0 的配置时，若索引输入处于活动状态时方向发生变化，则位置计数器 (QPOSCNT) 可能会错误地复位，从而导致计数器值发生意外变化。若如此，可能会与位置计数器的预期值相差多达 ±4 个数值，并随后意外设置错误标志。 在使用 PCRM = 0 的配置时[即发生索引事件时位置计数器复位 (QEPCTL[PCRM] = 00)]，若在正向运动期间发生索引事件，则位置计数器在下一个 eQEP 时钟时复位为 0。若在反向移动期间发生索引事件，则位置计数器将在下一个 eQEP 时钟时复位为 QPOSMAX 寄存器中的值。eQEP 外围设备会记录第一个索引标记 (QEPSTS[FIMF]) 的出现情况以及 QEPSTS 寄存器中第一个索引事件标记 (QEPSTS[FIDF]) 方向。此外，其还会记录第一个索引标记上的正交边沿以便在索引事件复位操作中使用相同的相对正交跃迁。 如果在索引脉冲活动时方向发生改变，则该模块仍将继续查找相对正交跃迁以执行位置计数器复位操作。若如此，位置计数器的值会发生意外变化。 未同时发生方向改变的下一个索引事件将正确复位计数器并按预期工作。
解决方法	在索引处于活动状态时，若方向发生变化且位置计数器值的相应变化可能会影响应用程序，请勿使用 PCRM = 0 的配置。 若适用于此应用程序，则位置计数器复位所运用的其他选项[例如索引事件初始化(IEI)]不会出现此问题。

公告

FPU：FPU 至 CPU 寄存器移动操作之前是任何 FPU 2p 操作

受影响版本

0

详细信息

当多周期 (2p) FPU 指令后跟 FPU 到 CPU 寄存器转移时，此公告适用。如果 FPU 到 CPU 读取指令源寄存器与 2p 指令目标相同，则在 2p 指令完成之前读取的可能是 FPU 寄存器的值。这是因为 2p 指令依赖于流水线 E3 阶段期间结果的数据转发。如果在 E3 阶段碰巧发生流水线停滞，则读取指令不会及时转发结果。

受此公告影响的 2p 指令是 MPYF32、ADDF32、SUBF32 和 MACF32。FPU 寄存器读取的目标必须是一个 CPU 寄存器（ACC、P、T、XAR0...XAR7）。如果寄存器读取是 FPU 到 FPU 寄存器转移，则此公告不适用。

在下面的示例中，2p 指令 MPYF32 使用 R6H 作为其目标。FPU 寄存器读取 MOV32 使用同一个寄存器 R6H 作为其源，并使用 CPU 寄存器作为目标。如果在 E3 流水线阶段发生停滞，则 MOV32 将在 MPYF32 指令完成之前读取 R6H 的值。

问题示例：

```
   MPYF32 R6H, R5H, R0H  ; 2p FPU instruction that writes to R6H
|| MOV32 *XAR7++, R4H
   F32TOUI16R R3H, R4H   ; delay slot
   ADDF32 R2H, R2H, R0H
|| MOV32 *--SP, R2H      ; alignment cycle
   MOV32 @XAR3, R6H      ; FPU register read of R6H
```

图 3-4 展示了流水线中没有停滞时的问题流水线图。

	Instruction	F1	F2	D1	D2	R1	R2	E	W		Comments
		FPU pipeline--->				R1	R2	E1	E2	E3	
I1	MPYF32 R6H, R5H, R0H || MOV32 *XAR7++, R4H	I1									
I2	F32TOUI16R R3H, R4H	I2	I1								
I3	ADDF32 R3H, R2H, R0H || MOV32 *--SP, R2H	I3	I2	I1							
I4	MOV32 @XAR3, R6H	I4	I3	I2	I1						
			I4	I3	I2	I1					
				I4	I3	I2	I1				
					I4	I3	I2	I1			
						I4	I3	I2	I1		
							I4	I3	I2	**I1**	I4 samples the result as it enters the R2 phase. The product R6H=R5H*R0H (I1) finishes computing in the E3 phase, but is **forwarded** as an operand to I4. This makes I4 appear to be a 2p instruction, but I4 actually takes 3p cycles to compute.
								I4	I3	I2	
									I4	I3	

图 3-4. 流水线中没有停滞时的问题流水线图

公告 (续)

FPU：FPU 至 CPU 寄存器移动操作之前是任何 FPU 2p 操作

图 3-5 展示了指令 I1 的 E3 时隙中存在停滞时的问题流水线图。

	Instruction	F1	F2	D1	D2	R1	R2	E	W		Comments
		FPU pipeline-->				R1	R2	E1	E2	E3	
I1	MPYF32 R6H, R5H, R0H || MOV32 *XAR7++, R4H	I1									
I2	F32TOUI16R R3H, R4H	I2	I1								
I3	ADDF32 R3H, R2H, R0H || MOV32 *--SP, R2H	I3	I2	I1							
I4	MOV32 @XAR3, R6H	I4	I3	I2	I1						
			I4	I3	I2	I1					
				I4	I3	I2	I1				
					I4	I3	I2	I1			
						I4	I3	I2	I1		
							I4	I3	I2	I1 (STALL)	I4 samples the result as it enters the R2 phase, but I1 is stalled in E3 and is unable to forward the product of R5H*R0H to I4 (R6H does not have the product yet due to a design bug). So, I4 reads the old value of R6H.
							I4	I3	I2	I1	There is no change in the pipeline as it was stalled in the previous cycle. I4 had already sampled the old value of R6H in the previous cycle.
								I4	I3	I2	Stall over

图 3-5. 指令 I1 的 E3 时隙中存在停滞时的问题流水线图

解决方法

在本例中，应将 MPYF32、ADDF32、SUBF32 和 MACF32 视为 3p 周期指令。必须将三个 NOP 或非冲突指令放置在指令的延迟时隙中。

C28x 代码生成工具 v.6.2.0 及更高版本都将生成正确的指令序列并检测汇编代码中的错误。在以前的版本 v6.0.5（适用于 6.0.x 分支）和 v.6.1.2（适用于 6.1.x 分支）中，编译器将生成正确的指令序列，但汇编器不会检测汇编代码中的错误。

解决方法示例：

```
   MPYF32 R6H, R5H, R0H
|| MOV32 *XAR7++, R4H      ; 3p FPU instruction that writes to R6H
   F32TOUI16R R3H, R4H     ; delay slot
   ADDF32 R2H, R2H, R0H
|| MOV32 *--SP, R2H        ; delay slot
   NOP                     ; alignment cycle
   MOV32 @XAR3, R6H        ; FPU register read of R6H
```

图 3-6 展示了带解决方法的流水线图。

公告 (续)

FPU：FPU 至 CPU 寄存器移动操作之前是任何 FPU 2p 操作

	Instruction	F1	F2	D1	D2	R1	R2	E	W		Comments
		FPU pipeline-->				R1	R2	E1	E2	E3	
I1	MPYF32 R6H, R5H, R0H || MOV32 *XAR7++, R4H	I1									
I2	F32TOUI16R R3H, R4H	I2	I1								
I3	ADDF32 R3H, R2H, R0H || MOV32 *--SP, R2H	I3	I2	I1							
I4	NOP	I4	I3	I2	I1						
I5	MOV32 @XAR3, R6H	I5	I4	I3	I2	I1					
			I5	I4	I3	I2	I1				
				I5	I4	I3	I2	I1			
					I5	I4	I3	I2	I1		
						I5	I4	I3	I2	I1 (STALL)	Due to one extra NOP, I5 does not reach R2 when I1 enters E3; thus, forwarding is not needed.
						I5	I4	I3	I2	I1	There is no change due to the stall in the previous cycle.
							I5	I4	I3	I2	I1 moves out of E3 and I5 moves to R2. R6H has the result of R5H*R0H and is read by I5. There is no need to forward the result in this case.
								I5	I4	I3	

图 3-6. 带解决方法的流水线图

公告 ***I2C：目标发送器模式、标准模式 SDA 时序限制***

受影响版本

0

详细信息

MCU 上的 I2C 外设是一款快速模式器件；当与标准模式主机一同使用时，其将对 SCL（时钟）线路执行时钟延展。

I2C 规范对于标准模式系统中使用的快速模式器件有一项要求，即在释放 SCL 线路之前满足 $t_{SU:DAT}$（数据设置时间）+ $t_{r\ (max)}$（上升时间）。请参阅 NXP 半导体 *I²C 总线规范和用户手册 (UM10204)* 中“标准、快速和快速模式 Plus I²C 总线器件的 SDA 和 SCL 总线特性”表的脚注 4。

然而，在上述情况下， C2000 I2C 时钟会将 SCL 线延展到一个固定量 = 6 * f_{mod} 时钟（C2000 的 I2C 时钟速率）。当 C2000™微控制器充当标准模式主机的目标发送器时，如果 SDA 的 t_r 过长，则 C2000 可能会在数据 (SDA) 就绪之前释放时钟线路 (SCL)。

NXP Semiconductors *I²C 总线规范和用户手册*(UM10204) 中的“上拉电阻器大小调整"部分提供了有关根据上升时间 (t_r) 和总线电容 (C_b) 选择适当 pu 电阻器 (R_p) 的更多详细信息，如方程式 1 所示。

$$R_{p(max)} = \frac{t_r}{0.8473 \times C_b} \quad (1)$$

解决方法

1. **通过使用强上拉电阻来减小 t_r**
   为了确保满足 $t_{SU:DAT}$ + $t_{r\ (max)}$，用户可以在 SDA 线路上配置上拉电阻，以便根据其系统中的 f_{mod} 时钟值，满足表 3-3 的“SDA 数据上升时间要求”列中列出的限制。这将确保在 C2000 释放 SCL 信号时，SDA 线路上存在的数据有效。

   表 3-4 给出了在给定 f_{mod} 时钟 (MHz) 和 C_b（总线电容）下的建议 R_p 电阻值。对于其他 C_b 值，请使用方程式 1 来计算系统中所需的 R_p 值。

表 3-3. C2000 作为带有标准模式主机的目标发送器时的数据上升时间要求

f_{mod} 时钟 (MHz)	f_{mod} 周期 (ns)	C2000 I2C 的 SCL 时钟延展延迟 (ns)：（6*f_{mod} 时钟）	数据建立时间 (ns)：$t_{SU:DAT}$（标准模式）	SDA 数据上升时间要求 (ns)：t_r
7	142.9	857	250	607
8	125	750		500
9	111	666		416
10	100	600		350
11	90.9	545		295
12	83.3	500		250

公告 (续)

I2C：目标发送器模式、标准模式 SDA 时序限制

表 3-4. 公共总线电容 (C_b) 的上拉电阻 (R_p) 值

f_{mod} 时钟 (MHz)	SDA 数据上升时间要求 (ns)：t_r	R_p (kΩ)，对于 C_b = 100 pF	R_p (kΩ)，对于 C_b = 200 pF	R_p (kΩ)，对于 C_b = 300 pF	R_p (kΩ)，对于 C_b = 400 pF
7	607	7.1	3.5	2.3	1.7
8	500	5.9	2.9	1.9	1.4
9	416	4.9	2.4	1.6	1.2
10	350	4.1	2.0	1.3	1.0
11	295	3.4	1.7	1.1	0.8
12	250	2.9	1.4	0.9	0.7

2. **T_r = 1000ns**

   由于一般 I2C 使用方面的限制、因此不推荐使用此权变措施、请尽可能使用权变措施 1。

   如果系统在 SDA 线路上需要 1000 ns 的上升时间、则可以将 C2000 I2C f_{mod} 时钟配置为 4.8 MHz，以便时钟延展（6 * f_{mod} 时钟）满足该要求。由此可得，t_r = (1/4.8 MHz) * 6 = 1000 ns。此权变措施仅在 C2000 I2C 是 I2C 总线上目标的系统中有效。请注意，4.8 MHz 超出了数据表中 f_{mod} 时钟 7 MHz 至 12 MHz 的所需范围。在 4.8 MHz 时使用 f_{mod} ，尽管其已超出数据表的所需范围，也将适用于标准模式主机总线上的目标模式下的 C2000 I2C。除了此权变措施中列出的配置外，在任何其他配置中使用 f_{mod} = 4.8 MHz 都将导致违反其他时序参数的要求，因此不允许如此操作。

公告 **LIN: 当同步字段错误时，同步字段错误 (ISFE) 标志/中断未设置**

受影响版本

0

详细信息

在 LIN 通信期间，如果接收到的同步字段（在 RX 上）是错误的（即同步字段接收到除 0x55 之外的任何值），则 LIN 不会设置 SCIFLR.ISFE 寄存器中的 ISFE 标志或触发 ISFE 中断。在未接收到数据或设置 RX 接收中断的情况下，通信会终止。应用程序无法检测同步字段中的错误。该应用程序可以检测同步字段是否完全为空，或同步字段是否未在给定容差范围内接收到（如 *TMS320F28003x 实时微控制器技术参考手册*中所述），但该应用程序无法检测同步字段值中的任何错误。

解决方法

方法 1：如果未在特定时间内设置该标志，则继续轮询 SCIFLR.RXRDY 标志并触发超时。

将以下步骤作为指南执行：

1. 轮询要设置的 SCIFLR.BUSY 标志。
2. 一旦 BUSY 标志变为高电平，轮询 SCIFLR.RXRDY 标志。在此循环中同时还有一个 SW 超时，如果 RXRDY 标志未在用户定义的时间间隔内设置，则会超时并退出循环。

方法 2：将 CPU 计时器配置为在 RX 中断未被触发时中断。此方法不使用 CPU 带宽。

将以下步骤作为指南执行：

1. 配置 XINT 以在 LINRX 从高电平变为低电平时触发 ISR（表示 LIN 繁忙）。
2. 在 XINT ISR 内部配置 CPU 计时器，随后该计时器开始对帧完成计时。
3. 如果使用正确的同步字段正确接收到帧，则将触发 LIN RX ISR。您可以在其中关闭计时器，从而不会得到错误的超时。
4. 如果帧接收不正确，则不会触发 LIN RX ISR，而会触发 CPU 计时器 ISR（发生超时），这表示同步字段中存在错误。

公告 *存储器：在有效存储器之外进行预取*

受影响版本

0

详细信息

C28x CPU 预取指令的范围超出其流水线中当前正在活动的指令范围。如果预取发生在有效存储器结束之后，则 CPU 可能会接收到无效的操作码。

解决方法

M1、GS3 – 预取队列的深度为 8 x16 字。因此，代码不应在有效存储器末尾的 8 个字以内。可以在两个有效存储器块之间实现跨边界预取。

示例 1：M1 在地址 0x7FF 处结束，后面不跟随另一个存储器块。M1 中代码的存储地址不应超过 0x7F7。地址 0x7F8-0x7FF 不应用于代码。

示例 2：M0 结束于地址 0x3FF，有效存储器 (M1) 紧随其后。M0 中的代码可存储在 0x3FF 及以下的地址。代码也可以交叉进入 M1，最高到地址 0x7F7（含地址 0x7F7）。

闪存 – 预取队列的深度为 16 x16 字。因此，代码不应在有效存储器末尾的 16 个字以内；否则，将生成闪存 ECC 不可纠正的错误。

表 3-5. 受公告影响的存储器

存储器类型	受影响的地址
M1	0x0000 07F8-0x0000 07FF
GS3	0x0000 FFF8 – 0x0000 FFFF
闪存	0x000A FFF0 – 0x000A FFFF

公告	***引导 ROM：TMX 器件上 MCAN 引导加载程序的数据溢出***
受影响版本	0 *（仅限 TMX）*
详细信息	在 TMX 器件中，当使用 MCAN 引导加载程序时，F28003x 侧会发生通信数据溢出，从而导致引导加载程序故障。
解决方法	主机需要确保在连续帧之间设置 400 µs 延迟，以确保成功通信。此问题在 TMS 器件中已得到修复。

公告	*安全实时固件更新 (LFU) 引导模式已弃用*
受影响版本	0
详细信息	如果配置了安全 LFU 引导模式，并且存储器已加密保护，则器件将复位。
解决方法	最新的固件组选择算法可以在闪存自定义引导加载程序的一部分实现，而不是从 ROM 调用。在完成最新固件组选择后，同一自定义引导加载程序可调用位于安全 ROM 中的 CMAC 认证算法来认证闪存内容。应将保存此自定义引导加载程序的闪存配置为 Z1 安全存储器。

公告	***闪存：在未禁用闪存预取的情况下执行 Fapi_setActiveFlashBank () 可能导致 ITRAP***
受影响版本	0
详细信息	闪存预取机制只能由从 RAM 执行的代码禁用，且仅在对闪存没有任何活动取指访问时才应禁用。Fapi_setActiveFlashBank () 会禁用预取机制。当预取禁用过程正在进行时，任何闪存执行都可能导致 ITRAP；因此，在执行 Fapi_setActiveFlashBank() 之前应禁用预取。
解决方法	在调用 Fapi_setActiveFlashBank() 之前禁用预取机制。可以在执行 Fapi_setActiveFlashBank () 后启用预取机制。禁用和启用预取机制的代码应从 RAM 执行。实施此解决方法时，请注意 Fapi_setActiveFlashBank() 可从闪存（而不是从擦除/编程操作所针对的存储体）执行，因为预取已被禁用。 C2000Ware_5_02_00_00（及更高版本）中提供的闪存 API 使用示例展示了此解决方法的实现。 TMS320F28003x 示例路径： C2000Ware_5_02_00_00\driverlib\f28003x\examples\flash\flashapi_ex1_programming.c 在本例中，从 RAM 执行 Flash_DisablePrefetch_SW_Workaround() 来禁用预取，以在调用 Fapi_setActiveFlashBank() 之前禁用预取。调用 Fapi_setActiveFlashBank () 后，从 RAM 执行 Flash_EnablePrefetch_SW_Workaround() 以启用预取。

公告	***系统：在 Pagesel=0 模式下，HIC 非法读取错误标志不会被置位***
受影响版本	0
详细信息	当主机对挂起写入位置的同一地址发起读取访问（非法访问序列）时，在 Pagesel=0 模式下，非法读错误标志不会置位。在 Pagesel=1 模式下，相同序列会置位该错误标志。影响很低，因为它是一个非法序列，并且在常规应用流程中，软件不应发起对写挂起位置的读取。
解决方法	无

公告	***系统：多次连续写入 CLKSRCCTL1 可能导致系统挂起***
受影响版本	0
详细信息	在 写入 CLKSRCCTL1 寄存器时，在写入操作之间多次未延迟，则系统可能会发生挂起的情况，并且只能通过外部 XRSn 复位或 Watchdog 复位才能恢复。在 SYSCLK 与 OSCCLKSRCSEL 所选时钟之间的时钟比有问题时，就会出现这种情况，然而，并非每次都会出现这个问题。 如果在使用调试器时遇到此问题，那么在点击暂停后，程序计数器将位于 Boot ROM 复位向量处。 实施此权变措施，可避免由 SYSCLK 与 OSCCLK 时钟比引起的这种情况。
解决方法	每次写入 CLKSRCCTL1 寄存器后，使用 NOP 指令添加 300 个 SYSCLK 周期的软件延迟。 示例： C2000Ware_3_00_00_00 和更高版本将实施此权变措施。

```
ClkCfgRegs.CLKSRCCTL1.bit.INTOSC2OFF=0;         // Turn on INTOSC2
asm("  RPT #250 || NOP");                       // Delay of 250 SYSCLK Cycles
asm("  RPT #50 || NOP");                        // Delay of 50 SYSCLK Cycles
ClkCfgRegs.CLKSRCCTL1.bit.OSCCLKSRCSEL = 0;     // Clk Src = INTOSC2
asm("  RPT #250 || NOP");                       // Delay of 250 SYSCLK Cycles
asm("  RPT #50 || NOP");                        // Delay of 50 SYSCLK Cycles
```

公告 **在使用 HWBIST 时避免杂散中断**

受影响版本

0

详细信息

HWBIST 能够记录 CPU 处于测试状态时接收到的中断，并在 HWBIST 完成后重新发出这些中断。在中断日志记录启用之前的时钟周期中接收到的中断将在 HWBIST 运行之前执行。在下一个周期中，中断日志记录启用，中断将被记录，并在 HWBIST 完成后重新发出。

CPU 计时器 1 和 CPU 计时器 2 的中断事件有效期为 2 个 SYSCLK 周期内。如果第一个周期出现在中断日志记录启用之前的一个周期，并且第二个周期与中断日志记录启用同时发生，则中断将在日志记录之前执行一次 (清除 CPU 计时器 TCR.TIF 标志) ，但随后将被中断记录器记录，并在 HWBIST 完成后再次触发。由于 TCR.TIF 标志已被上一个 ISR 清除，因此这是意外的杂散中断。

请注意，此场景仅适用于非 PIE CPU 计时器中断。CPU 计时器 0 中断由 PIE 管理，其脉冲宽度仅为一个 SYSCLK 周期。

解决方法

在中断日志记录启用之前禁用 CPU 计时器 1 和 2 中断，稍后再恢复这些中断。此解决方法已在 C2000 软件诊断库的 stl_hwbist_s.asm 文件中实现。相关步骤如下：

1. 清除 CPU 计时器 1 和 2 的计时器中断使能位 TCR.TIE。
2. 运行正常的 HWBIST 序列：保存寄存器、启用中断日志记录、运行 HWBIST、恢复寄存器、结束中断日志记录。
3. 检查 CPU 计时器的 TCR.TIF 标志是否置位。如果这些标志已置位，则设置相应的 CPU IFR 位来触发中断。
4. 恢复 TCR.TIE。

公告	***PLL 基准时钟丢失检测：时钟丢失标志可能被错误激活***
受影响版本	0
详细信息	寄存器位 SYSPLLSTS.REF_LOST 可能设置错误，因此错误地指示基准时钟丢失。
解决方法	请勿使用 PLL 基准时钟丢失检测功能。或者，可以使用时钟丢失检测(MCD)功能或双时钟比较器(DCC)来检测基准时钟丢失。 • 对于 MCD 方法、请参阅 SYSCTL 文件夹下的 C2000Ware 示例 sysctl_ex1_missing_clock_detection。 • 对于 DCC 方法、请参阅 DCC 文件夹下的 C2000Ware 示例 dcc_ex4_clock_fail_detect。

公告	***SDFM：动态更改阈值设置（LLT、HLT）、滤波器类型或 COSR 设置将触发虚假比较器事件***
受影响版本	0
详细信息	当 SDFM 比较器设置（例如滤波器类型、下限/上限阈值或比较器 OSR (COSR) 设置）在运行期间动态更改时，将触发虚假比较器事件。如果配置得当，虚假比较器事件将触发相应的 CPU 中断、CLA 任务、ePWM X-BAR 事件和 GPIO 输出 X-BAR 事件。
解决方法	当需要动态更改比较器设置时，请执行以下程序，以确保虚假比较器事件不会产生 CPU 中断、CLA 任务或 X-BAR 事件（ePWM X-BAR/GPIO 输出 X-BAR 事件）： 1. 禁用比较器滤波器。 2. 至少延迟比较器滤波器的延时 + 3 个 SD-Cx 时钟周期。 3. 更改比较器滤波器设置，例如滤波器类型、COSR 或下限/上限阈值。 4. 至少延迟比较器滤波器的延时 + 5 个 SD-Cx 时钟周期。 5. 启用比较器滤波器。

公告	***SDFM：动态更改数据滤波器设置（例如滤波器类型或 DOSR）将触发虚假数据响应事件***
受影响版本	0
详细信息	当 SDFM 数据设置（例如滤波器类型或 DOSR 设置）在运行期间动态更改时，将触发虚假数据滤波器就绪事件。如果配置得当，虚假数据就绪事件将触发相应的 CPU 中断、CLA 任务和 DMA 触发器。
解决方法	当需要动态更改 SDFM 数据滤波器设置时，请执行以下程序，以确保不会产生虚假数据滤波器就绪事件： 1. 禁用数据滤波器。 2. 至少延迟数据滤波器的延时 + 3 个 SD-Cx 时钟周期。 3. 更改数据滤波器设置，如滤波器类型和 DOSR。 4. 至少延迟数据滤波器的延时 + 5 个 SD-Cx 时钟周期。 5. 启用数据滤波器。

公告	***SDFM：在三个 SD 调制器时钟周期内对 SDCPARMx 寄存器位字段 CEVT1SEL、CEVT2SEL 和 HZEN 进行两次背对背写入可能会损坏 SDFM 状态机，从而导致虚假比较器事件***
受影响版本	0
详细信息	在三个 SD 调制器时钟周期内对 SDCPARMx 寄存器位字段 CEVT1SEL、CEVT2SEL 和 HZEN 进行背对背写入可能会损坏 SDFM 状态机，从而导致虚假比较器事件，如果配置得当，这可能触发 CPU 中断、CLA 任务、ePWM XBAR 事件和 GPIO 输出 X-BAR 事件。
解决方法	避免在三个 SD 调制器时钟周期内背对背写入或在一次寄存器写操作中配置 SDCPARMx 寄存器位字段。

公告	*看门狗：WDKEY 寄存器不受 EALLOW 保护*
受影响版本	0
详细信息	WDKEY 寄存器不受 EALLOW 保护。无需发出 EALLOW 和 EDIS 指令来写入该寄存器。要在 WDKEY 受 EALLOW 保护的其他器件上启用软件重用，建议使用 EALLOW 和 EDIS。
解决方法	无

4 文档支持

欲获得器件专用数据表和相关文档，请访问 TI 网站：https://www.ti.com。

欲了解有关 TMS320F28003x 器件的更多信息，请参阅以下文档：

- *TMS320F28003x 实时微控制器*数据表
- *TMS320F28003x 实时微控制器技术参考手册*

5 商标

C2000™ is a trademark of Texas Instruments.
所有商标均为其各自所有者的财产。

6 修订历史记录

Changes from May 30, 2024 to August 31, 2026 (from Revision D (May 2024) to Revision E (August 2026)) **Page**

- 节 3.1.2 使用说明：标题从“*嵌套中断使用注意事项*”改为“*将嵌套中断与重复块一起使用时的注意事项*”。更新了“使用说明”描述。 7
- 新增 I2C 从机模式使用说明：节 3.1.3。 7
- 增加了公告 - C28：快速整数除法 (FID) 和 DIVF64 结果 14
- 增加了公告 - ePWM：ePWM 单次/CBC 跳闸事件 DCxEVTy.force 不会设置跳闸条件 21

重要通知和免责声明

TI“按原样”提供技术和可靠性数据（包括数据表）、设计资源（包括参考设计）、应用或其他设计建议、网络工具、安全信息和其他资源，不保证没有瑕疵且不做出任何明示或暗示的担保，包括但不限于对适销性、与某特定用途的适用性或不侵犯任何第三方知识产权的暗示担保。

这些资源可供使用 TI 产品进行设计的熟练开发人员使用。您将自行承担以下全部责任：(1) 针对您的应用选择合适的 TI 产品，(2) 设计、验证并测试您的应用，(3) 确保您的应用满足相应标准以及任何其他安全、安保法规或其他要求。

这些资源如有变更，恕不另行通知。TI 授权您仅可将这些资源用于研发本资源所述的 TI 产品的相关应用。严禁以其他方式对这些资源进行复制或展示。您无权使用任何其他 TI 知识产权或任何第三方知识产权。对于因您对这些资源的使用而对 TI 及其代表造成的任何索赔、损害、成本、损失和债务，您将全额赔偿，TI 对此概不负责。

TI 提供的产品受 TI 销售条款)、TI 通用质量指南 或 ti.com 上其他适用条款或 TI 产品随附的其他适用条款的约束。TI 提供这些资源并不会扩展或以其他方式更改 TI 针对 TI 产品发布的适用的担保或担保免责声明。 除非德州仪器 (TI) 明确将某产品指定为定制产品或客户特定产品，否则其产品均为按确定价格收入目录的标准通用器件。

TI 反对并拒绝您可能提出的任何其他或不同的条款。

最后更新日期：2025 年 10 月