

Errata

MSPM33C32xx、MSPM33C32xx-Q1 微控制器



摘要

本文档介绍了功能规格的已知例外情况（公告）。

内容

1 功能公告.....1

2 预编程软件公告.....2

3 仅调试公告.....2

4 编译器修复公告.....2

5 器件命名规则.....2

 5.1 器件编号法和修订版本标识.....3

6 公告说明.....4

7 商标.....17

8 修订历史记录.....17

1 功能公告

影响器件运行、功能或参数的公告。

✓ 复选标记表示指定版本中存在该问题。

勘误编号	修订版 A
AES_ERR_01	✓
CPU_ERR_05	✓
FLASH_ERR_02	✓
GPIO_ERR_05	✓
GSC_ERR_01	✓
IOMUX_ERR_01	✓
I2S_ERR_01	✓
KEYSTORE_ERR_01	✓
PMCU_ERR_15	✓
RST_ERR_01	✓
RST_ERR_02	✓
SYSCTL_ERR_01	✓
SYSPLL_ERR_01	✓
TIMER_ERR_04	✓
TIMER_ERR_06	✓
TIMER_ERR_07	✓
TIMER_ERR_08	✓
UNICOMMI2CC_ERR_01	✓
UNICOMMI2CC_ERR_02	✓
UNICOMMI2CC_ERR_03	✓
UNICOMMSPI_ERR_02	✓

勘误编号	修订版 A
UNICOMMSPI_ERR_03	✓
UNICOMMSPI_ERR_04	✓
UNICOMMUART_ERR_01	✓
UNICOMMUART_ERR_02	✓
UNICOMMUART_ERR_03	✓
UNICOMMUART_ERR_04	✓
UNICOMMUART_ERR_05	✓
UNICOMMUART_ERR_06	✓
UNICOMMUART_ERR_07	✓
UNICOMMUART_ERR_10	✓
UNICOMMUART_ERR_12	✓
UNICOMMUART_ERR_13	✓
UNICOMMUART_ERR_14	✓
UNICOMMUART_ERR_15	✓

2 预编程软件公告

影响出厂编程软件的公告。

✓ 复选标记表示指定版本中存在该问题。

3 仅调试公告

仅影响调试操作的公告。

✓ 复选标记表示指定版本中存在该问题。

4 编译器修复公告

由编译器权变措施解决的公告。请参阅每个公告，以了解 IDE 和编译器版本及权变措施。

✓ 复选标记表示指定版本中存在该问题。

5 器件命名规则

为了标示产品开发周期所处的阶段，TI 为所有 MSP MCU 器件的器件型号分配了前缀。每个 MSP MCU 商用系列产品都具有以下两个前缀之一：**MSP** 或 **XMS**。这些前缀代表了产品开发的发展阶段，即从工程原型 (**XMS**) 直到完全合格的生产器件 (**MSP**)。

XMS - 实验器件，不一定代表最终器件的电气规格

MSP - 完全合格的生产器件

支持工具命名前缀：

X：还未经德州仪器 (TI) 完整内部质量测试的开发支持产品。

null：完全合格的开发支持产品。

XMS 器件和 **X** 开发支持工具在供货时附带如下免责条款：

“开发中的产品用于内部评估用途。”

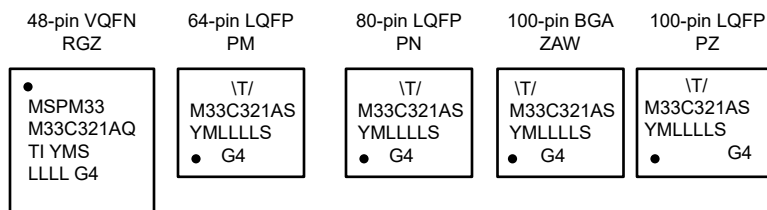
MSP 器件的特性已经全部明确，并且器件的质量和可靠性已经完全论证。TI 的标准保修证书对该器件适用。

预测显示原型器件 (**XMS**) 的故障率大于标准生产器件。由于这些器件的预计最终使用故障率尚不确定，德州仪器 (TI) 建议不要将它们用于任何生产系统。请仅使用合格的生产器件。

TI 的器件命名规则还包含具有器件产品系列名称的后缀。此后缀表示温度范围、封装类型和配送形式。

5.1 器件编号法和修订版本标识

下面的封装图展示了封装编号法方案，表 5-1 定义了器件修订版到版本 ID 的映射。



TI = TI Letters
YM = Year, month date code
S = Assembly site
= Die revision
LLLL = Assembly lot code
G4 = ECAT

图 5-1. 封装符号

表 5-1. 芯片修订版本

修订版字母	版本 (在器件的出厂常量存储器中)
A	0

修订版字母表示产品硬件修订版本。根据修订字母，本文档中的公告标记为“适用于”或“不适用于”给定器件。该字母映射到器件存储器中存储的整数，可用于使用应用软件或已连接的调试探针来查找修订版本。

6 公告说明

AES_ERR_01 *AES 模块*

类别

功能

功能

AES 保存的上下文就绪中断未按预期生成

说明

未生成已保存的上下文就绪中断。如果对任何 AES 寄存器进行了访问（读取或写入），则会生成中断。

权变措施

使用基于轮询的机制，检查 CTRL 寄存器中已保存的上下文就绪的状态位，而不是中断。

CPU_ERR_05 *CPU 模块*

类别

功能

功能

MTB_BASE 寄存器值读取不正确

说明

读取 MTB_BASE 寄存器时，MTB_SRAM 基地址将右移一位。例如，如果 MTB_SRAM 显示 0x80806000，则实际 MTB_SRAM 基地址为 0x40403000。

解决方法

当读取 MTB_BASE 寄存器以查找 MTB_SRAM 基址时，请将读取到的值右移一位。请使用以下代码解决问题：uint32_t mtb_base = (*(uint32_t*)(0xE004300C) >> 1);

FLASH_ERR_02 *FLASH 模块*

类别

功能

功能

可以使用密码重新启用 NONMAIN 中的调试禁用

说明

如果在 NONMAIN 配置中禁用了调试 (DEBUGACCESS = 0x5566)，则仍可以使用编程的密码访问器件（如果未明显编程，则使用默认密码）。

权变措施

权变措施 1：

将 DEBUGACCESS 设置为使用密码启用调试选项 (DEBUGACCESS = 0xCCDD)，并在 PWDDEBUGLOCK 字段中提供唯一的密码。为了提高安全性，建议使用加密随机的器件唯一密码。这将允许使用正确的 128 位密码进行调试访问，但仍可允许某些调试命令以及对 CFG-AP 和 SEC-AP 的访问。

权变措施 2：

通过禁用 SWDP_MODE 来完全禁用物理 SW 调试端口。这可完全防止对器件进行任何调试访问或发送申请，但可能会影响故障分析和返回流程。

GPIO_ERR_05

GPIO 模块

类别

功能

功能

当 DMA 传输正在进行时，可能会错过对 GPIO DOUTTGL 寄存器的写入

说明

当并发 DMA 传输正在进行时，GPIO DMAMASK 寄存器信息被错误地应用于 CPU 对 DOUTTGL 寄存器的写入。

权变措施

在应用程序代码中，确保针对 DOUTTGL 寄存器中的相应位将 GPIO DMAMASK 位设置为 1，然后再发出 CPU 对 DOUTTGL 寄存器的写入访问权限。如果不需要向任何 GPIO 寄存器进行 DMA 传输，则 GPIO DMAMASK 可在 IO 初始化步骤期间配置为 0xFFFFFFFF。这将解决此勘误表中的冲突。如果应用程序还需要对 GPIO 寄存器进行 DMA 写入传输，建议应用程序不要同时使用 DMA 和 CPU 来写入器件中同一 GPIO 模块的 DOUTTGL 寄存器。如果器件具有多个 GPIO 模块，则 DMA 和 CPU 可以同时向不同 GPIO 模块的 DOUTTGL 寄存器写入数据（同时仍要求为 CPU 要写入的 GPIO 模块配置 GPIO DMAMASK）。

GSC_ERR_01

GSC 模块

类别

功能

功能

某些计时器具有相同的安全属性

说明

在 PPC_SECATTRIB_TIMER 寄存器中，SEC_TIMA0_0 和 SEC_TIMA0_1 寄存器绑定在一起，因此 TIMERA0_0 和 TIMERA0_1 的安全属性必须相同。TIMERG4_2 和 TIMERG4_1 也具有相同的配置，但仅当 DMA 尝试访问这些计时器时才如此。

权变措施

应尽量将 TIMERA0_0 与 TIMERA0_1 保持为相同的安全级别。如果无法做到这一点，请改用其他计时器实例对。对于 TIMERG4_2 和 TIMERG4_1，请将它们保持在相同的安全级别，尤其是在使用 DMA 访问 TIMERG4_2 或 TIMERG4_1 时。

IOMUX_ERR_01

IOMUX 模块

类别

功能

功能

复位后 TDO 引脚会驱动低电平信号。

说明

退出复位状态后，TDO 引脚作为低电平信号被驱动。其余引脚退出复位后均为高阻状态。

权变措施

如需将此引脚配置为高阻态，请将 PC 位和 PF 位均置零。否则，请将此 IO 引脚连接到您计划使用的外设。

I2S_ERR_01	I2S 模块
类别	功能
功能	使能 I2S/TDM 模块后首次 TXIFG 中断未触发
说明	在使能 I2S/TDM 模块后，首次 TXIFG 中断不会被触发。
解决方法	向 INTCTL.INTEVAL 位写入 1，以生成初始 TXIFG 中断。
KEYSTORE_ERR_01	密钥库模块
类别	功能
功能	STATUS.STAT 值可以是 0 或 1，无需密钥访问
说明	STATUS.STAT 的复位值为 1 并在以下条件下变为 0：1.复位后，调试器通过寄存器窗口访问将返回 0x00。2.复位后，第一次 CPU 读取将返回 0x01，后续 CPU 读取将返回 0x00。3) 复位后，首先读取任何其他密钥库寄存器，然后读取 STATUS.STAT 将返回 0x00。
权变措施	STATUS.STAT = 0x0 表示“无错误”。要检查插槽是否有效（是否有密钥），请检查 STATUS.VALID。
PMCU_ERR_15	PMCU 模块
类别	功能
功能	启用 SYSPLL 之前的 GENCLKEN 时钟相关配置将被忽略
说明	配置 SYSPLL 时，需先置位 HSCLKEN 寄存器中的 SYSPLLEN 位，之后再对 GENCLKEN 寄存器进行配置。如果在设置 SYSPLLEN 位之前对 GENCLKEN 寄存器中的 EXTDIVCAN、MCLKEXTDIVEN、I2SPLLCLKDIVEN 和 I2SPLLCLKDIVCFG 位进行编程，则 GENCLKEN 寄存器中的配置将无法正确生效，并且 EXTDIVCAN 和 I2SPLLCLKDIVCFG 配置将被忽略。
权变措施	必须在 SYSPLLEN 位设置完成后，再配置 GENCLKEN 寄存器中的 EXTDIVCAN、MCLKEXTDIVEN 和 I2SPLLCLKDIVEN、I2SPLLCLKDIVCFG 位。
RST_ERR_01	RST 模块
类别	功能

RST_ERR_01 (续) *RST 模块*

功能

在 LFXT 或 LFCLK_IN 丢失后，NRST 释放沿与 LFOSCGOOD 上升沿重合时，器件会保持复位状态

说明

在特定的极端情况下，如果 NRST 信号的释放沿（低电平到高电平转换）与 LFOSCGOOD 的上升沿重合，则器件无法检测到 NRST 置为无效，并会无限期地保持复位状态。可能会触发该问题的情况：当 LFCLK 源从 LFCLK_IN 或 LFXT 切换到 LFOSC 时，LFOSC 将重新启用，并且 LFOSCGOOD 状态在重新启用事件的 250us（最小值）至 1.5ms（最大值）内置为高电平。如果施加的 NRST 低电平脉冲使其释放沿（低电平到高电平转换）在 50ns 窗口内与 LFOSCGOOD 上升沿对齐，则不会检测到 NRST 置为无效，且器件保持复位状态。

解决方法

保持 NRST 低电平脉冲宽度高于 2ms 即可避免此问题

RST_ERR_02 *RST 模块*

类别

功能

功能

当 NRST 置为无效与初始 LFOSCGOOD 置为有效重合时，器件可能无法上电

说明

在上电事件中，一旦内部内核电压 (VCORE) 达到稳定电平，就会启用内部低频振荡器 (LFOSC)。LFOSCGOOD 信号在 VCORE 达到稳定后的 250us（最小值）至 1.5ms（最大值）内置为高电平。VDD 的 VBOR+ 阈值可用作 VCORE 变为稳定时的基准点。如果 NRST 置为无效（低电平到高电平转换）与 LFOSCGOOD 上升沿在 200ns 窗口内重合，则不会检测到 NRST 置为无效，且器件保持复位状态，无法退出启动序列的引导阶段。由于 NRST 置为无效时序由 NRST 电路上的 VDD 斜坡速率以及外部电阻器和电容器 (RC) 网络控制，因此实际上可能会触发这种情况 — 已知易受影响的配置包括 NRST 电路上的 47kΩ 电阻器和 10nF 电容器。

解决方法

权变措施 1：在 RC 滤波器中使用足够小的上拉电阻器，以确保 NRST 信号在 LFOSCGOOD 最小置为有效时间 (250us) 之前至少 50us 上升到 VDD 的 90%，从而确保 NRST 置为无效沿与 LFOSCGOOD 置为有效窗口之间有足够的间隔。请注意，所需的上拉最小电阻取决于外部复位控制器（如果适用）的 IO 驱动强度。已知符合标准的配置是 NRST 电路上的 4.7kΩ 电阻器以及 10nF 电容器。Workaround2：采用一个外部复位控制器，使 NRST 在 VDD 斜坡开始后保持在逻辑低电平至少 2.0ms，从而确保 NRST 置为无效发生在 LFOSCGOOD 最大置为有效时间 (1.5ms) 之后。

SYSCCTL_ERR_01 *SYSCCTL 模块*

类别

功能

功能

SW-POR 功能与 HW-POR 结合使用

SYSCTL_ERR_01

(续)

SYSCTL 模块**说明**

当用户使用正确的密钥写入 LFSSRST 寄存器以生成软件触发的 POR 时，RSTCAUSE 寄存器将显示 0x2 (指示 NRST 触发的 POR)，而不是预期的 0x3 (软件触发的 POR)。这是因为 SW-POR 功能是与 HW-POR 路径结合使用的。

权变措施

否

SYSPLL_ERR_01**SYSPLL 模块****类别**

功能

功能

启用后，SYSPLL 频率可能无法锁定到正确的频率。

说明

当在 SYSCTL HSCLKEN 寄存器中将 SYSPLLEN 位设置为 1 时，SYSPLL 将运行锁相环搜索。如果频率不会设置为正确的值，搜索可能会失败，相反，生成的频率将与配置的频率大不相同。

权变措施**频率验证流程**

只要 SYSPLLEN 位设置为 1，就会使用频率时钟计数器 (FCC) 监控 SYSPLL 的频率输出。设置正确的频率后，它将保持稳定，直到 SYSPLL 被禁用并重新启用 (SYSPLLEN 位从 0 切换到 1)。如果检测到频率不正确，请禁用并重新启用 SYSPLL 以再次执行验证。

权变措施 1：FCC 计数检查

使用 LFCLK 作为 FCC 触发源，以计算 SYSPLL 输出时钟频率。执行 FCC，并使用 LFCLK 作为参考，将测量值与配置的 SYSPLL 频率进行比对。

计算示例：

- SYSPLLCLK0 = 160MHz；LFCLK = 32.768kHz
- 测量的 FCC 计数 = $160,000,000 / 32,768 = 4,882$

FCC 计数容许：

实际 FCC 计数会因组合的时钟精度 (SYSPLLCLK0 和 LFCLK) 而有所变化。建议为允许的 FCC 检查范围增加 +/- 5~10%。

- FCC 计数上限 = $4,882 * 1.05 = 5,126$
- FCC 计数下限 = $4,882 * 0.95 = 4,638$

时序注意事项：

- 时钟同步时间：5-6 个 LFCLK 周期
- FCC 触发时间：1-32 个 LFCLK 周期 (用户可配置)

寄存器配置：

- FCC 设置：SYSCTL.GENCLKCFG.FCCTRIGSRC = 1;
- SYSCTL.GENCLKCFG.FCCLVLTRIG = 0;
- SYSCTL.GENCLKCFG.FCCTRIGCNT = 0;
- SYSCTL.GENCLKCFG.FCCSELCLK = 4;
- 起始 FCC：SYSCTL.FCCCMD = 0x0E000001U
- 检查 FCC 完成状态：SYSCTL.CLKSTATUS.FCCDONE
- 读取 FCC 计数：SYSCTL.FCC

超时保护：

在监控 FCC 完成状态期间，实现软件超时机制，以避免当未锁定的 SYSPLL 频率低于

SYSPLL_ERR_01

(续)

SYSPLL 模块

FCC 触发时钟 (LFCLK) 时等待时间延长。

```
fccTimeOutCounter = 0;
while (DL_SYSCTL_isFCCDone() == 0) {
    delay_cycles(977); /* 1 个 LFCLK 周期 = 32MHz/32.768kHz */
    fccTimeOutCounter++;
    if(fccTimeOutCounter > 65) break;
    /* 超时设置为大约 2ms (用户可配置) */
}
```

FCC 检查重新启动：

如果 FCC 测量值超出预期范围，请禁用并重新启用 SYSPLL (将 SYSPLLEN 设为 0，再设为 1)，然后重复 FCC 验证。

权变措施 2：FCC 比率检查

使用 LFCLK 作为 FCC 触发源，以计算 SYSPLL 输出和输入时钟频率。执行 FCC，并对比预期比率，验证测量的输出时钟与输入时钟的 FCC 检查值比率。

计算示例：

- SYSPLL = 160MHz；HFCLK = 40MHz；LFCLK = 32.768kHz
- 预期时钟比率 = 160MHz/40MHz = 4.0000
- 测量的 FCC 计数 (SYSPLL) = 160,000,000/32,768 = 4,882
- 测量的 FCC 计数 (HFCLK) = 40,000,000/32,768 = 1,220
- 测量的时钟比率 = 4,882/1,220 = 4.0016

FCC 比率容许：

FCC 比率方法可消除组合时钟精度造成的误差，仅取决于 FCC 不确定性 (2 个计数时钟周期) 及计算取整误差。与 FCC 计数检查方法相比，这可实现更严格的容许范围，例如 +/- 0.3%。

时序注意事项：

- 时钟同步时间：5-6 个 LFCLK 周期
- FCC 触发时间：1-32 个 LFCLK 周期 (用户可配置)
- 每次完整 FCC 比率检查的总时间：2* (同步时间 + 触发时间)

FCC 比率检查流程：

1. 设置 FCC 以检测 SYSPLL 输出时钟 (SYSPLL0 或 SYSPLL2X)
2. 启动 FCC 并等待 FCC 完成 (添加超时保护，请参阅 FCC 计数检查)
3. 读取 FCC 检查计数
4. 设置 FCC 以检测 SYSPLL 输入时钟 (SYSOSC 或 HFCLK)
5. 启动 FCC 并等待 FCC 完成 (添加超时保护，请参阅 FCC 计数检查)
6. 读取 FCC 检查计数
7. 计算 FCC 检查比率，并与预期比率范围进行比较
8. 如果 FCC 比率超出预期范围，请禁用并重新启用 SYSPLL (将 SYSPLLEN 设为 0，再设为 1)，然后重复 FCC 比率验证。

TIMER_ERR_04

TIMER 模块

类别

功能

功能

如果接近零事件，则可能会错过计时器的重新启用

TIMER_ERR_04

(续)

TIMER 模块**说明**

在单次模式下使用计时器时，如果接近零事件，则可能会错过计时器的重新启用。对计时器使能位进行硬件更新将需要一个功能时钟周期。例如，如果计时器的时钟源为 32.768kHz，时钟分频器为 3，则需要 ~100us 才能将使能位正确设置为 0。

权变措施

在重新启用计时器之前等待 1 个功能时钟周期，或者可以先禁用计时器，然后再重新启用。

通过 CTRCTL.EN = 0 禁用计数器，然后通过 CTRCTL.EN = 1 重新启用

TIMER_ERR_06**TIMER 模块****类别**

功能

功能

向 CLKEN 位写入 0 不会禁用计数器

说明

向计数器时钟控制寄存器 (CCLKCTL) 时钟使能位 (CLKEN) 写入 0 不会停止定时器。

权变措施

通过向计数器控制 (CTRCTL) 使能 (EN) 位写入 0 来停止定时器。

TIMER_ERR_07**初始重复计数器的周期比下一个重复模块少 1 个****类别**

功能

功能

计时器

说明

使用计时器重复计数器模式时，第一次重复的计数将比后续重复的计数少 1，因为以下重复计数器将包括 0 和加载值之间的转换。例如，如果 TIMx.RCLD = 0x3，则第一个重复计数器上将出现 3 个可观察到的零事件，并在以下重复计数器序列上显示 4 个可观察到的零事件。

权变措施

将初始 RCLD 值设置为比预期的 RCLD 大 1，然后在重复计数器归零事件 (REPC) 的 ISR 中将 RCLD 设置为预期的 RCLD 值。例如，如果打算重复 4 次，请将初始 RCLD 值设置为 RCLD = 0x5，然后在 REPC 中断的计时器 ISR 中设置 RCLD = 0x4。现在，所有计时器重复将具有相同数量的零/加载事件。

TIMER_ERR_08**TIMx 模块****类别**

功能

功能

在高级计时器的顺序事件生成过程中，GEN_EVENT1 输出会被忽略

TIMER_ERR_08

(续)

TIMx 模块

说明

在高级计时器 (TIMA) 中，GEN_EVENT1 仅适用于单个事件，并且在第一个事件触发后，无法通过事件处理程序正确传播。

权变措施

请勿将 GEN_EVENT1 用于高级计时器上的重复事件输出。请使用 GEN_EVENT0 或采用通用计时器 (TIMG) 实例。

UNICOMMI2CC_E RR_01

UNICOMMI2CC 模块

类别

功能

功能

I2C 控制器繁忙状态轮询问题

说明

当通过设置 BUSTRUN/FRAME_START 位来启动 I2C 控制器传输时，繁忙状态标志需要大约 2-3 个 I2C 功能时钟周期才能置为有效。如果在设置 BUSTRUN/FRAME_START 后立即轮询繁忙位，则应用可能会在该状态得到正确设置之前检查该状态。在 CLKDIV 值较高（导致 I2C 功能时钟较慢）或编译器优化级别较高的情况下，该问题会更加明显。

权变措施

在轮询繁忙状态之前添加软件延迟。建议的延迟应为：软件延迟 = $3 \times \text{I2C 功能时钟} = 3 \times \text{clock_divider} \times (\text{CPU_CLK}/\text{所选时钟源频率})$ ，例如 clock_divider 为 2、时钟源为 4MHz (MFCLK)、CPU_CLK 为 80MHz：软件延迟 = $3 \times 2 \times (80 \text{ MHz} / 4 \text{ MHz}) = 120$ 个 CPU 周期

UNICOMMI2CC_E RR_02

UNICOMMI2CC 模块

类别

功能

功能

当 I2C 控制器的 ACKOEN 位使能时，RXDONE 中断会被触发两次

说明

当 I2C 控制器的 CTR.ACKONE 位使能时，软件可以通过向 CTR.ACK 位写入 0 或 1 来控制 ACK/NACK 信号的发送。这一操作可以在停止条件发生前的 RXDONE 中断中进行。在写入 CTR.ACK 位之后，I2C 总线上会产生有效的 ACK/NACK 信号，但在随后的 I2C 停止状态下，还会再次触发一次额外的 RXDONE 中断。

解决方法

在写入 CTR.ACK 位之前，先禁用 RXDONE 中断；直到下一次传输开始前，再手动重新使能该中断。

UNICOMMI2CC_E RR_03

UNICOMMI2CC 模块

类别

功能

UNICOMMI2CC_ER R_03 (续)

UNICOMMI2CC 模块

功能

启用时钟延展功能时，I2C 控制器的 SCL 频率较低

说明

- 请务必不要建议使用 ETW 功能作为解决方法。- 请确保 ERRATA 提案不会影响此器件修订版或先前修订版的其他现有 ERRATA。启用时钟扩展功能 (CR.CLKSTRETCH = 1) 时，SCL 频率将低于设置值。例如，I2C 控制器总线速度设置为 400kHz 时，如果启用了时钟延展功能，则 I2C 控制器总线时钟将低于 400kHz (例如 360kHz 或 390kHz)。在其他总线速度 (例如 100kHz 或 1MHz) 下，也会出现相同现象。

解决方法

不建议禁用时钟延展功能，因为这对 I2C 通信功能不会产生很大影响。禁用时钟延展功能后，即可看到预期的 I2C 总线时钟速度。

UNICOMMSPI_ER R_02

UNICOMMSPI 模块

类别

功能

功能

控制器端错误读取繁忙状态

说明

SPI 繁忙状态并不取决于 TXFIFO 的数据元素，因此本质上，在我们向 SPI TXFIFO 加载某些数据后，如果 CLKSEL 选择的是较慢的时钟 (例如 LFCLK)，或者 CLKDIV > /1，则繁忙位不会立即变为高电平。由于 SPI 源时钟较慢，这会导致在向 TXFIFO 加载数据之后到通信实际开始之前的这段时间内读取到错误的状态。

解决方法

可以在写入应用程序代码时引入对 TXFIFO 空状态位的检查。

UNICOMMSPI_ER R_03

UNICOMMSPI 模块

类别

功能

功能

使用 CPU 清除 RXFIFO 时，第二次不会设置 RTOUT

说明

在启用 rxtimeout 的情况下完成 SPI 通信并设置 RTOUT 后，如果使用 CPU 读出 RXFIFO 数据并读取 IIDX 以清除中断，但未执行任何复位或禁用 IP 的操作，那么当第二次进行相同传输时，即使 rxtimeout 计数器为“0”且 RXFIFO 非空，也不会设置 RTOUT 中断。

解决方法

使用 DMA 读出 RXFIFO 数据，而不要直接使用 CPU。或在直接使用 CPU 读出 RXFIFO 数据后复位 UNICOMM SPI。

UNICOMMSPI_ER R_04

UNICOMMSPI 模块

类别

功能

功能

在 DSS 不匹配的上一次传输之后重新进行传输时，RX 中存储的数据不正确

说明

如果控制器端与外设端使用不同的数据包大小设置 (DSS) 发送数据，随后执行 RXCLR 清除 FIFO，再以正确的数据包大小重新发送数据，在第一个 SCLK 脉冲到来时，仍会有旧数据进入 RXFIFO。

解决方法

请确保控制器端和外设端的数据包大小设置 (DSS) 相同。

UNICOMMUART_E RR_01

UNICOMMUART 模块

类别

功能

功能

在 IrDA 模式中引入干扰时的数据完整性问题

说明

UNICOMM-UART 支持 IrDA，但在嘈杂的环境中有限制。由于没有干扰滤波器，当数据线路上出现干扰时，IrDA 模式可能会发生数据完整性问题。由于 IrDA 依赖于边沿检测逻辑，因此这些干扰会被误解为有效脉冲，从而导致 FIFO 中出现意外数据和 LTOUT 计算问题。

权变措施

否

UNICOMMUART_E RR_02

UNICOMMUART 模块

类别

功能

功能

IDLELINE 模式下有干扰时的空闲周期检测问题

说明

空闲期间的干扰可能会破坏接收器的空闲检测，从而导致 Rx 侧丢弃后续地址字节。重要的是，此故障模式与时序有关：只有在特定时间插入的相对于空闲期间的干扰才会触发问题。并非空闲期间内的每个干扰都会导致损坏。

解决方法

否

UNICOMMUART_E RR_03

UNICOMMUART 模块

类别

功能

UNICOMMUART_E

RR_03 (续)

UNICOMMUART 模块

功能

Manchester 模式下的干扰数据完整性问题

说明

Manchester 编码数据流中的干扰会破坏信号的精确边沿时序，从而导致错误的接收数据以及数据完整性问题。

权变措施

否

UNICOMMUART_E

RR_04

UNICOMMUART 模块

类别

功能

功能

在 LIN 模式下的中断域和/或同步域期间，数据将因干扰而丢失

说明

中断场景（高脉冲干扰）：在中断域周期期间，负边沿会将计数器复位为零，导致中断域检测失败并造成数据丢失。同步域方案：同步字段期间的干扰将触发错误的 LINC0/LINC1 中断，在 Rx 线路负边沿复位计数器，导致同步字段验证失败并造成数据丢失。

权变措施

否

UNICOMMUART_E

RR_05

UNICOMMUART 模块

类别

功能

功能

UART 干扰滤波器不存在导致的问题

说明

1. IRDA 模式数据完整性：由于没有干扰滤波器，当数据线路上出现干扰时，IrDA 模式可能会发生数据完整性问题。由于 IrDA 依赖于边沿检测逻辑，因此这些干扰会被误解为有效脉冲。2. Ideline 检测：在 IDLELINE 检测期间，信号干扰可能会通过将内部空闲线路计数器复位为零来破坏检测过程。这种干扰会导致 IDLELINE 状态标志无法正确置位，从而使模块无法识别通信线路上的有效空闲状态。但是，如果在这种中断后，空闲线路状态持续保持，且没有额外的干扰，则检测机制最终将恢复并正确检测空闲状态。3. Manchester 模式数据完整性：在曼彻斯特编码传输的数据采样期间，如果出现信号干扰，接收器将捕获错误数据，从而导致位被错误解析。这些采样错误会导致接收缓冲区中存储损坏的数据值，从而导致接收消息中出现数据完整性故障。在采样窗口之外出现的干扰通常不会影响数据解释。4. LIN 模式下 BREAK / SYNC 字段的敏感性：a. BREAK 字段：在 LIN 通信期间，如果信号干扰刚好发生在 BREAK 字段的时钟采样期间，检测机制可能会提前终止。但是，该系统具有复原力，如果中断过后 BREAK 字段的状态仍在持续且没有新的干扰出现，检测逻辑将会恢复并成功识别出 BREAK 字段。b. SYNC 字段：该系统在 SYNC 字段校准期间也表现出类似的脆弱性。在此期间，干扰会干扰下降沿和上升沿（RXNE 和 RXPE）的检测。5. 干扰导致的 LTOUT/RTOUT 检测问题：当空闲线路检测期间出现信号干扰时，这些干扰会暂时中断 LTOUT/RTOUT 的时间计算，并将超时计数器重置为 0。如

UNICOMMUART_E

RR_05 (续)

UNICOMMUART 模块

果线路在中断后没有其他干扰，则系统将在配置的时间段之后检测到 LTOUT/RTOUT 条件。

权变措施

没有可用的解决方法。

UNICOMMUART_E

RR_06

UNICOMMUART 模块

类别

功能

功能

由于停止位处理而导致 RTOUT/LTOUT 计算问题

说明

在接收器侧，功能状态机在停止位中间从停止位转换为空闲。这会使接收超时 (RTOUT) 和线路超时 (LTOUT) 计数器在停止位的中间而非末端开始计数。这会导致 RTOUT/LTOUT 提前半个波特周期触发。这在 UART 功能时钟频率较高的低波特率下尤其明显。

权变措施

向 RTOUT 计数器添加具有半个停止位周期的补偿。

UNICOMMUART_E

RR_07

UNICOMMUART 模块

类别

功能

功能

如果在 RS - 232 模式下禁用 UART，则 RTS 线路不会变为高电平

说明

当 UART 被禁用时，RTS 线路无法恢复到其空闲状态（高电平），仍然卡在低电平。

权变措施

使用软件来启用内部上拉电阻，并将 RTS 线路 IO 设置为 Hiz 模式。

UNICOMMUART_E

RR_10

UNICOMMUART 模块

类别

功能

功能

LIN 寄存器 CLKDIV 限制

说明

当 CLKDIV 值不是 0 时，写入 LINC0/1 将不会产生任何影响。

权变措施

要正确配置 LIN 寄存器：1.首先将 CLKDIV 设置为“0” 2.使用所需的值更新 LINC0/1 寄存器配置 3.将 CLKDIV 恢复至其预期工作值

UNICOMMUART_E RR_12

UNICOMMUART 模块

类别

功能

功能

CTL0.TXE 位在帧中间设置为 0 会导致当前传输出错

说明

开始发送数据包，然后在传输过程中，将 CTL0.TXE 位设置为 0，当前数据包传输不会完成，TX 线路将下拉至低电平，显示 BUSY 行为（表示繁忙）。仅当 CTL0.TXE = 1 时才会释放 TX 线，在此之前，BUSY 保持为 1。

解决方法

用户必须轮询 BUSY 状态以变为低电平和 CTL0.TXE = 0，以避免出现此问题。

UNICOMMUART_E RR_13

UNICOMMUART 模块

类别

功能

功能

RTOUT 中断提前 1 个 fclk 周期出现

说明

RTOUT（接收超时）中断比应出现的确切时间提前了 1 个 fclk 周期。

解决方法

用户可以通过预判 RTOUT 中断会提前一个 fclk 周期出现来处理这一情况。

UNICOMMUART_E RR_14

UNICOMMUART 模块

类别

功能

功能

CTS 中断在 CTS 线路的下降沿和上升沿均会切换

说明

CTS 中断在 CTS 线路的上升沿和下降沿均会切换。理想情况下，它应该仅在 CTS 线路的下降沿切换。

解决方法

用户可以监视 STAT.CTS 位切换为 1，而不是 RIS 位，因为 STAT.CTS = 1 的这种切换表示遇到了 CTS 线路下降。

UNICOMMUART_E RR_15

UNICOMMUART 模块

类别

功能

功能

未生成 fclk 和新 IDLELINE 帧之间的 LCRH.SENDIDLE 访问丢失

UNICOMMUART_E

RR_15 (续)

UNICOMMUART 模块

说明

如果在两个 fclk 边沿之间将 LCRH.SENDIDLE 从 0 切换到 1，然后从 1 切换到 0；或者从 1 切换到 0，然后从 0 切换回 1，则将错过切换，并且不会发送新的 IDLELINE 帧。

解决方法

切换 LCRH.SENDIDLE 后，不要立即将其切回，请维持多个 fclk 周期，因为 LCRH.SENDIDLE 是在 fclk 上进行采样的。

7 商标

所有商标均为其各自所有者的财产。

8 修订历史记录

注：以前版本的页码可能与当前版本的页码不同

Changes from June 30, 2026 to September 16, 2026 (from Revision B (June 2026) to Revision C (September 2026))

Page

• 添加了 I2S_ERR_01 公告.....	6
• 已更新 RST_ERR_01 权变措施.....	6
• 添加了 RST_ERR_02 公告.....	7
• 添加了 UNICOMMI2CC_ERR_02 公告.....	11
• 添加了 UNICOMMI2CC_ERR_03 公告.....	11
• 添加了 UNICOMMSPI_ERR_02 公告.....	12
• 添加了 UNICOMMSPI_ERR_03 公告.....	12
• 添加了 UNICOMMSPI_ERR_04 公告.....	13
• 添加了 UNICOMMUART_ERR_12 公告.....	16
• 添加了 UNICOMMUART_ERR_13 公告.....	16
• 添加了 UNICOMMUART_ERR_14 公告.....	16
• 添加了 UNICOMMUART_ERR_15 公告.....	16

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任何明示或暗示的担保，包括但不限于对适销性、与某特定用途的适用性或不侵犯任何第三方知识产权的暗示担保。

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

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

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

TI 反对并拒绝您可能提出的任何其他或不同的条款。

版权所有 © 2026，德州仪器 (TI) 公司

最后更新日期：2025 年 10 月