ADP32F1x 数字信号处理器

勘误表

编号：JXDZ7.381.002 KWSM

Advancechip Electronics

湖南进芯电子科技有限公司

2023 年 06 月

V1.0

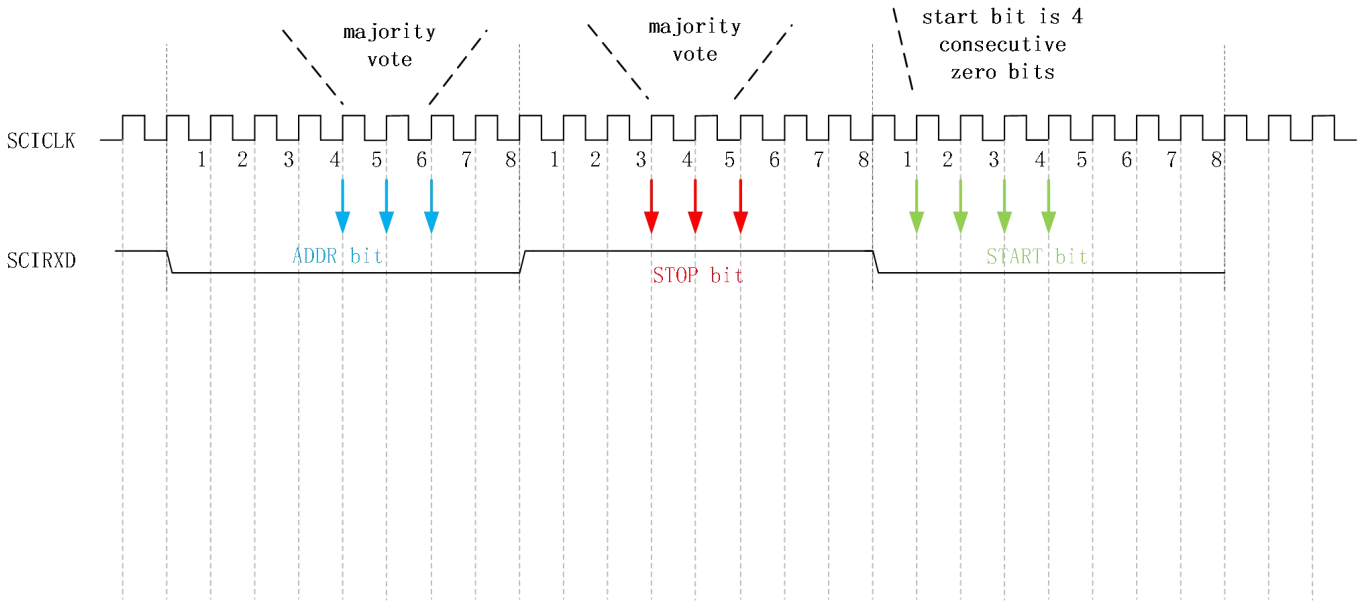
目 次

1 SCI：SCI 地址位模式的错误操作

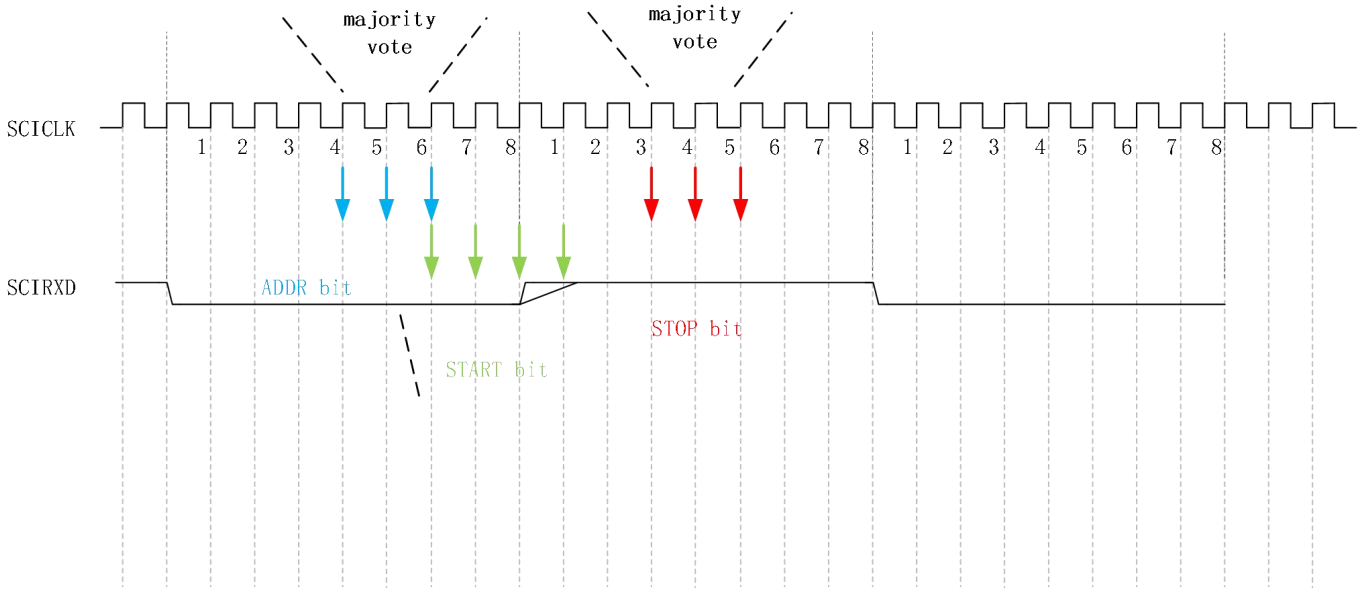
1.1 问题描述

SCI 不会在地址位之后寻找停止位。相反，从地址位的子样本 6 开始寻找起始位。由于开始位的第四个子样本可能会被认为是低，可能会导致一个错误的开始位发生，如下图所示：

预期操作：



错误操作：



1.2 建议

降低 SCI 波特率。这将导致错误启动位的第 4 个子样本在时间上被延迟，因此更多地发生在 STOP 位的中间位置（远离信号过渡区域）。所需的波特率降低量取决于系统中信号的上升时间。或者，如果适用，也可以使用 SCI 模块的 IDLE 模式。

2 SCI：引导加载程序在自动波特率锁定后没有清除 ABD 位

2.1 问题描述

在自动波特处理完成后，SCI 引导加载程序代码不会清除 SCIFFCT 寄存器中的自动波特率检测（ABD）位。如果在执行引导加载程序后使用 SCI-A 端口，则传输中断（SCITXINTA）将无法发生，SCI-A 的自动波特锁定特性也不能正常工作。

2.2 建议

如果 SCI 引导加载程序已经执行，用户的应用程序应该在启用 SCITXINTA 中断之前和使用自动波特功能之前，通过在 SCIFFCT 寄存器中写入一个 1 到 ABDCLR（bit14）来清除 ABD 位。

3 SCI：SCI 引导加载程序在自动波特锁定之前没有清除 ABD 位

3.1 问题描述

在自动波特处理开始之前，SCI 引导加载程序没有正确地清除 ABD 位。引导加载程序的代码片段如下图所示：

```
// Prepare for autobaud detection
// Set the CDC bit to enable autobaud detection
// and clear the ABD bit
SCIARegs.SCIFFCT.all = 0x2000;
```

注释错误地说明了 ABD 位已被清除。通过对 ABD_CLR 位（bit14）写 1 来清除该位。这种情况不会阻止上电或复位的操作，因为 ABD 位在复位后默认被清除。但是，如果第二次从软件中调用引导加载程序，那么 ABD 位将不会被清除，自动波特锁定将不会正常发生。

3.2 建议

如果引导加载程序将被软件重新调用，那么用户的代码必须在调用引导加载程序之前首先清除 ABD 位。为此，需要向 SCIFFCT 寄存器中的 ABD_CLR 位（bit14）写入一个 1。

4 WD：WDFLAG 位不能按预期的方式工作

4.1 问题描述

WDFLAG 位不能用于辨别来自上电（加热）的看门狗复位。这是因为该设备期望在看门狗启动的复位脉冲（512OSCLK 周期）结束后的 4 个 SYSCLKOUT 周期（8 个 OSCLK 周期）内拉高 XRS 引脚（通过外部复位电路）。大多数外部 XRS 电路不能提供快速上升时间要求（由于电容）；因此，在应用中不应使用 WDFLAG 位。

4.2 建议

不使用 WDFLAG 位。

5 ADC：当 INT MOD SEQ1/2 启用时，在测序器 1/2 的转换结束时，ADCST 中的 EOS BUF1/2 位错误

5.1 问题描述

根据用户指南，在 ADCTRL2 中设置 INT MOD SEQx 位会导致每隔一个序列转换结束时而不是在每次转换结束时生成 INT SEQx。此外，在每次转换结束时都将设置 EOS BUFx，以跟踪正在使用中的 SEQx 的状态。然而，如果为该排序器启用了 INT_MOD_SEQ2，即使 INT_MOD_SEQ1 没有启用，则 SEQ1 上的转换都将导致 EOS_BUF2 被设置。

例如，如果设置了 INT_MOD_SEQ2，则在 SEQ1 上的转换，将导致 EOS_BUF2 被错误地设置。这将导致在下一次 SEQ2 完成后，INT SEQ2 被错误地设置。如果 EOS_BUF2 已经被设置（来自先前的 SEQ 转换），在 SEQ1 上的一个转换将导致 EOS_BUF2 被清除，导致中断丢失。上述关系也适用与影响 SEQ1 的 SEQ2。在所有情况下，排序器都能正常工作，除了 EOS BUFx 被损坏。

5.2 建议

如果在 INT_MOD_SEQ 选择的排序器上完成两个序列之前同时使用 SEQ1 和 SEQ2，则不要使用 INT_MOD_SEQx 功能。

6 ADC：ADCASEQSR 寄存器的保留位

6.1 问题描述

SEQ2 STATE2-0 和 SEQ1 STATE3-0 位字段（bit6:0）分别是 SEQ2 和 SEQ1 的指针。这些位被保留用于内部测试，不应用于客户应用程序。

6.2 建议

不使用这些位。

7 ADC：当双排序器运行时，排序器复位

7.1 问题描述

片上有 SEQ1 和 SEQ2 两个排序器用于执行 ADC 的转换,如果其中一个排序器在另一个排序器正在运行时复位，它将导致正在运行中的排序器永远不会完成其当前序列。排序器繁忙位（ADCST 的 bit3:2）将保持活动状态，正在运行的排序器的“EOS”中断将永远不会产生。例如，如果在 SEQ2 执行一个序列转换时，复位 SEQ1，那么 SEQ2 将永远不会完成转换。

7.2 建议

如果启用了双排序器，那么软件应该在 BYS 位没有设置的时候复位任何一个排序器。

8 ADC：结果寄存器延迟更新

8.1 问题描述

ADCST 寄存器中的标志 INT_SEQ1 和 INT_SEQ2 位字段（bit1:0）表示在转换和启动 ADC 中断后，新的 ADC 结果的可用性。

在标志 INT_SEQ1 和 INT_SEQ2 被设置后，ADC 结果寄存器的更新需要一个额外的 ADC 周期来完成。在此额外 ADC 周期之前去读取 ADC 结果寄存器的值将导致旧数据被读取。

如果在 MAXCONV 寄存器中使用非零值启用自动排序器，则最后一个结果寄存器的更新将花费额外的一个 ADC 周期 。

8.2 建议

延迟至少一个 ADC 时钟周期去读取 ADC 结果寄存器的值，可以使用软件延时。

如果使用 ADC 中断而不是轮询来读取 ADC 结果寄存器，则由 ISR（中断服务例程）引入的等待周期可以尽量减少软件中的延时。这个 ISR 的分支延迟通常大于 8 个系统时钟周期。

9 McBSP：接收 FIFO 读取冲突

9.1 问题描述

McBSP 带有 FIFO 或没有 FIFO 的操作。接收 FIFO 具有中断生成逻辑，该逻辑基于 MFFRX 寄存器中的 5 位 FIFO 状态位（bit12:8）和中断水平位（bit4:0）启动中断。

如果 CPU 在 McBSP 模块将新数据写入 FIFO 时读取接收 FIFO，则会有一个潜在的冲突。CPU 读取将不会停止，读取数据将无效。FIFO 写入获得优先级。在 DRR2/1 寄存器中接收到每个字后，接收 FIFO 将被更新。DRR1/2 寄存器更新时间将主要取决于字的大小和 CLKR 速率。例如，对于 8 位字，它通常应该是 CLKR 周期的 8 倍。如果在接收通道上传输的数据在字与字之间是连续没有延迟的，那么这种冲突更加明显。

9.2 建议

接收 FIFO 应该根据接收中断并在下一个字的接收时间内被读取。为了避免读取冲突，可以在启动接收 FIFO 读取之前使用其他检查。在大多数 McBSP 配置中，FSR 是一个接收同步脉冲，该脉冲要么高要么低（基于 FSR 的极性位），并且在字传输期间将不活动。这些活动和不活动的相位可以通过检查 FSR 标志位（MCFFST 的 bit3）或者 FSR 引脚的状态来被检测。

10 McBSP：读操作减少 McBSP FIFO

10.1 问题描述

对以下任一地址进行读取操作都将导致 McBSP FIFO 内容减 1，就好像 McBSP DRR1 寄存器已经被读取一样：

0x7001 Reserved

0x7401 EV-A T1CNT

0x7C01 Reserved

从该位置读取的实际值是正确的并且不受该问题的影响。

10.2 建议

1.在从这些地址执行任何读取操作之前，确保 McBSP 接收的 FIFO 为空。

2. 如果 McBSP 传输在应用程序中很常见，并且一个定时计数器需要被监视，可以考虑使用除 EV Timer1 以外的定时器。

11 SPI：从机模式操作

11.1 问题描述

在从机模式下，SPI 不会基于 SPISTE 重新同步接收到的字。因此，一个虚假的 SPICLK 脉冲可能会使数据流不同步。

11.2 建议

如果电路板没有足够的噪声来产生虚假的 SPICLK 脉冲，那么这就不是问题，如果噪声很大，则可以使用 SPI 从机模式下的 McBSP，因为 McBSP 会在每个新字上重新同步。

12 时钟：对于 XCLKIN 引脚的逻辑高电平

12.1 问题描述

该问题仅适用于使用外部振荡器时钟设备。这个 X1/XCLKIN 引脚是参考内核电源供电电压（Vdd），而不是 3.3V I/O 供电电压。因此，输入时钟的逻辑高电平不应超过 Vdd。这个要求对于以后的版本也保持不变。

12.2 建议

一个钳位二极管可以用来钳位缓冲时钟信号，以确保逻辑高电平不超过 VDD (1.8V 或 1.9V)。否则，1.8V 的振荡器才能够被使用。

13 QEP：QEP 电路

13.1 问题描述

在 DSP 重置之后，QEP 模块无法检测到发生在 QEP 输入引脚上的第一次转换。（EV 定时器使用外部时钟时，也会出现这个问题。）因此，如果第一次转换发生在 GP 定时器已经初始化并使能作为 QEP 计数器（也就是说，使用 QEP 作为时钟源）之后，那么第一次转换将不被 GP 定时器计算在内。其结果是，在 GP 定时器 256 行译码器的 1024 个计数中或在 1024 行译码器的 4096 个计数中，有一个计数错误。然而，这个问题和下列任何情况都无关：

- 第一次转换发生在 GP 定时器已经初始化并使能作为 QEP 计数器之前。这可以确保在初始化后计算所有的转换。
- 在接收到第一个索引脉冲之后，并且如果该索引脉冲用于重新校准 GP 定时器（通过捕获中断）。重新校准纠正了 GP 定时器中的错误；因此，从接收到第一索引脉冲开始，QEP 计数器变得精确。

13.2 建议

- 确保第一次转换发生在 GP 定时器已经初始化并使能作为 QEP 计数器之前。这是一种很常见的情况，因为通常在 GP 定时器初始化之前，转子轴被锁定在已知位置。锁定转子轴将在 QEP 输入引脚上产生转换，除非转子轴完全对准到已知位置（这是一种罕见的情况）。可以故意扰乱转子轴的位置，以处理这种罕见情况。
- 使用编码器的索引脉冲来重新校准用作 QEP 计数器的 GP 定时器。
- 在应用程序真正使用 QEP 之前，强制计数器计数。在初始化期间，配置内部时钟（HSPCLK）作为计数器时钟源。在第一次计数完成之后，计数器应该重新配置为外部信号（QEP/TCLKIN），并重置为 0。计数器也将计数 QEP 的第一个边沿。

14 QEP：在 GPIO 异步模式下的 QEP 输入

14.1 问题描述

如果任何一个 QEP 输入引脚通过 GPxQSELn 寄存器被配置为 GPIO 异步输入模式，则 QEP 模块可能无法正常运行。例如，QPOSCNT 可能无法正确地复位或锁存，并且输入引脚上的脉冲可能会丢失。这是因为 QEP 外设假定输入到模块的系统时钟的外部同步的存在。

为了实现 QEP 的正常运行，输入 GPIO 引脚通过 GPxQSELn 寄存器配置为同步输入模式（有或没有限定）。这是在 GPxQSEL 寄存器的默认状态。我们提供的所有 QEP 外设的例程也配置了 GPIO 输入位同步输入模式。QEP 模块输入引脚不应该使用异步模式。

14.2 建议

QEP 的 GPIO 输入配置为非异步模式（除“11b = 异步”）

15 内存：预取超过了有效的地址

15.1 问题描述

C28x CPU 预取指令超出了当前在 pipeline 活动的指令。如果预取发生在有效内存结束之后，那么 CPU 可能会收到一个无效的操作码。

15.2 建议

预取的队列深度为 8 个 16 位字。因此，代码不应该在有效内存结束的 8 个字以内。此限制适用于设备上的所有内存区域和所有内存类型（Flash/ROM、OTP、SARAM、XINTF）。可以跨越两个有效内存块之间的边界进行预取。

案例 1：M1 以地址 0x7ff 结束，后面没有另一个内存块。M1 中的代码存储范围不超过 0x7f7。地址 0x7f8-0x7ff 不应用于代码。

案例 2：M0 以地址 0x3ff 结束，后面是有效内存（M1）。M0 中的代码可以存储的并包括地址 0x3ff。代码也可以跨越到 M1，并包括地址 0x7f7。

16 内存：从 Flash/ROM 内存的程序读取

16.1 问题描述

当程序代码执行从地址范围 0x3f7ff0 到 0x3f7ff7 的指令时，发生中断，则从 Flash/ROM 读取的后续数据可能会返回 0。

16.2 建议

不要将程序代码放在此地址范围内。此地址范围可用于数据变量的存储。

17 内存：Flash 和 OTP 预取缓冲区溢出

17.1 问题描述

此建议适用于启用了 Flash 预取缓冲区的在 Flash 或 OTP 执行的代码。

如果在 8 个 16 位字宽的 SBF 或 BF 指令中其中一个使用了直接或间接程序地址寻址操作，则 Flash 预取缓冲区可能会溢出。发生此情况的窗口如下所示：

```
Address
Offset
0x0000 BF LSW (32-bit opcode)
0x0001 BF MSW or SBF (16-bit opcode)
--------------------------------------
0x0002 SBF/BF + 1 word //
0x0003 SBF/BF + 2 words //
0x0004 SBF/BF + 3 words // If an instruction within this window
0x0005 SBF/BF + 4 words // uses program-memory addressing, it
0x0006 SBF/BF + 5 words // can cause the flash prefetch buffer to
0x0007 SBF/BF + 6 words // overflow.
0x0008 SBF/BF + 7 words //
0x0009 SBF/BF + 8 words //
--------------------------------------
0x0010 SBF/BF + 9 words
```

溢出是否发生实际取决于指令序列、flash 等待状态和 CPU pipeline stalls。如果发生溢出，它将导致一个无效的操作码执行。使用程序内存寻址的指令有 MAC/XMAC,DMAC/XMACD,QMACL,IMACL,PREAD/XPREAD 和 PWRITE/XPWRITE。

17.2 建议

1.手动汇编代码

对于在 flash 或 OTP 执行的代码，使用 SB/B 指令去替代 SBF/BF。SB/B 指令在等待状态的内存中更有效，因此也可以看到性能有所提高

2.编译器生成汇编代码

使用编译器开关 -me 来强制编译器生成 SB/B 指令，而不是 SBF/BF 指令。在大量的等待状

态内存中，SB/B 指令比 SBF/BF 更有效。在 SARAM 中，SBF/BF 指令更有效。因此，这个开关因此，该开关应该应用到如下：

- 对从 flash 或 OTP 运行的源代码使用编译器-me 开关。
- 不要对从 SARAM 运行的源代码使用-me 开关。
- 如果一个文件包含 flash 运行的函数以及从 SARAM 运行的函数，那么请使用-me 开关。

-me 开关在 C28x 编译器 V4.1.4 及之后的版本中可用。

18 XINTF：XBANK 没有正确地扩展访问权限

18.1 问题描述

当 XTIMCLK 不等于系统时钟，XBANK 逻辑可能不会正确的延迟挂起的访问。这种情况发生在 XINTF 区域等待状态和 XBANK 延迟周期的某些组合中。有两种情况会发生：

情况 1：当 XTIMCLK = 1/2 SYSCLKOUT 和 XCLKOUT = XTIMCLK 时

如果有以下任何情况，XBANK 逻辑可能不会延迟挂起的访问：

- WLEAD + WACTIVE + WTRAIL <= XBANK[BCYC]
- RLEAD + RACTIVE + RTRAIL <= XBANK[BCYC]

其中 WLEAD ,WACTIVE ,WTRAIL ,RLEAD ,RACTIVE ,RTRAIL 定义如下表所示：

	X2TIMING = 0	**X2TIMING = 1**
WLEAD	XTIMING x [XWRLEAD]	XTIMING x [XWRLEAD] x 2
WACTIVE	XTIMING x [XWRACTIVE] + 1	XTIMING x [XWRACTIVE] x 2 + 1
WTRAIL	XTIMING x [XWRTRAIL]	XTIMING x [XWRTRAIL] x 2
RLEAD	XTIMING x [XRDLEAD]	XTIMING x [XRDLEAD] x 2
RACTIVE	XTIMING x [XRDACTIVE] + 1	XTIMING x [XRDACTIVE] x 2 + 1
RTRAIL	XTIMING x [XRDTRAIL]	XTIMING x [XRDTRAIL] x 2

在上表中，XTIMINGx 指的是 Zone x 的 XTIMING 定时寄存器。当在两次访问之间添加 XBANK 延迟周期时，Zone x 是指序列中的第一个区域。例如：如果 XBANK[BANK] = 7，则延迟周期将被添加到进出 Zone 7 的任何访问中。这就意味着：

- 在 Zone 7 之后接着访问 Zone 0：Zone 0 的时间非常关键
- 在 Zone 7 之后接着访问 Zone 1：Zone 1 的时间非常关键
- 在 Zone 0 之后接着访问 Zone 7：Zone 7 的时间非常关键

因此，必须考虑涉及 bank 转换的任何区域的时间。

情况 2：当 XTIMCLK = 1/2 SYSCLKOUT 和 XCLKOUT = 1/2 XTIMCLK 时

如果 XBANK[BCYC] = 4 或 XBANK[BCYC] = 6，则挂起访问的 XBANK 逻辑可能不会正确延迟。

18.2 建议

情况 1：当 XTIMCLK = 1/2 SYSCLKOUT 和 XCLKOUT = XTIMCLK 时，则选择：

- XBANK[BCYC] <= WLEAD + WACTIVE + WTRAIL 和
- XBANK[BCYC] <= RLEAD + RACTIVE + RTRAIL

当在两次访问之间添加 XBANK 延迟周期时，时间限制适用于前面访问的第一个区域。必须考虑涉及 bank 转换的任何区域的时间。

下表显示了有效的 XBANK[BCYC]选择的示例。这个列表不是很详细。

XWRLEAD XRDLEAD	WRACTIVE XRDACTIVE	XWRTRAIL XRDTRAIL	X2TIMING	WLEAD + WACTIVE + WTRAIL	Choose XBANK[BCYC]
1	2	1	0	5	<5
1	3	1	0	6	<6
2	3	1	0	7	<7
1	0	1	1	3	<3
1	1	0	1	5	<5
1	1	1	1	7	<7

情况 2：当 XTIMCLK = 1/2 SYSCLKOUT 和 XCLKOUT = XTIMCLK 时，则选择：

- XBANK[BCYC] != 4 和
- XBANK[BCYC] != 6

19 eCAN：Abort Acknowledge（AAn） 位没有置 1

19.1 问题描述

为了终止消息，在设置传输请求重置（TRR）寄存器位后，有一些罕见的情况：TRRn 和 TRSn 位将被清除，而没有设置终止确认（AAn）位。传输本身已正确终止，但没有产生中断，也没有中断挂起操作的迹象。

为了使这种罕见的情况发生，必须发生以下所有情况：

1. 先前的消息没有成功，要么是因为仲裁丢失，要么是因为总线上没有任何节点能够承认它，要么是因为传输导致了一个错误帧。先前的消息没有必要来自当前正在尝试进行传输终止的相同邮箱。

2. 邮箱的 TRRn 位应该在 TRSn 位被设置周期之后的一个 CPU 周期立即设置。由于也满足 TRSn 位已经被设置，但传输仍不完整这个条件的不完整传输，TRSn 位仍然被设置

3. TRRn 位必须在精确的系统周期时设置，此时 ，每个周期 CAN 模块处于空闲状态。当 CAN 模块不接收/发送数据时，CAN 模块处于空闲状态。

如果出现这些情况，则邮箱的 TRRn 和 TRSn 位将在 TRR 位设置之后的 Tclr = （（mailbox_number）*2）+3 SYSCLKOUT cycles 内被清除。

如果出现此情况，则不会设置 TAn 和 AAn 位。通常，在 TRR 位变为 0 之后，要么 TA 位被设置，要么 AA 位被设置。

19.2 建议

当此问题发生时，TRRn 和 TRSn 位将在 Tclr 系统周期内清除。为了检查此情况，请先首先禁用中断。在设置了 TRRn 位后，检查 TRRn 位的 Tclr 系统周期，以确保它仍然被设置好。TRRn 位的设置表示问题没有发生。

如果 TRRn 位被清除，则可能是由于消息的正常结束，并且设置了相应的 TAn 或 AAn 位。同时检查 TAn 和 AAn 位。如果设置了其中任何一个位，那么问题就没有发生。如果都为 0，那么就出现了问题。

如果设置了 TAn 或 AAn 位，那么当重新启用中断时，将会发生正常的中断例程。

20 eCAN：如果与 eCAN 寄存器的 eCAN 访问发生冲突，则对 eCAN 寄存器的 CPU 访问可能会失败

20.1 问题描述

如果 CPU 和 eCAN 控制器之间存在对某些 eCAN 寄存器区域的访问，CPU 的读取可能会错误地读取所有的 0，并且 CPU 的写操作可能无效。具体是以下四种情况：

- 情况 1：如果 CPU 在 eCAN 控制器访问（读取或写入）LAM/MOTO/MOTS 寄存器区域的同时，读取 eCAN 邮箱 RAM 区域（MSGID、MSGCTRL、MDL 或 MDH 寄存器），CPU 可能会错误地读取所有 0（0x00000000）
- 情况 2：如果 CPU 在 eCAN 控制器访问（读取或写入）LAM/MOTO/MOTS 寄存器区域的同时，写入 eCAN 邮箱 RAM 区域（MSGID、MSGCTRL、MDL 或 MDH 寄存器），则 CPU 写入可能无法执行。
- 情况 3：如果 CPU 在 eCAN 控制器访问（读取或写入）eCAN 邮箱 RAM 区域（MSGID、MSGID、MDL 或 MDH 寄存器）的同时读取 LAM/MOTO/MOTS 寄存器区域，则 CPU 可能错误地读取所有 0（0x00000000）。
- 情况 4：如果 CPU 在 eCAN 控制器访问（读取或写入）eCAN 邮箱 RAM 区域（MSGID、MSGCTRL、MDL 或 MDH 寄存器）的同时写入 LAM/MOTO/MOTS 寄存器区域，则 CPU 写入可能无法执行。

20.2 建议

这四种情况的解决方法如下：

- 情况 1：对于从 eCAN 邮箱 RAM 区域进行的所有 CPU 读取，检查读取是否返回所有零。如果是这样，CPU 应该执行第二次读取。如果第二次读取也返回零，那么数据实际就是零。

如果第二次读取返回一个非零值，则第二个数据是正确的值。请注意，在连续的 CPU 读取期间，必须禁用中断。

- 情况 2：对于所有发送到 eCAN 邮箱 RAM 区域的 CPU 写入操作，CPU 应该写入两次数据。请注意，在连续的 CPU 写入期间，必须禁用中断。
- 情况 3：对于从 LAM/MOTO/MOTS 寄存器区域进行的所有 CPU 读取，请检查该读取是否返回所有零。如果是这样，CPU 应该执行第二次读取。如果第二次读取也返回零，那么数据实际就是零。如果第二次读取返回一个非零值，则第二个数据是正确的值。请注意，在连续的 CPU 读取期间，必须禁用中断。
- 情况 4：对于所有到 LAM/MOTO/MOTS 寄存器区域的 CPU 写入，CPU 应该写入这两次数据，在写入之间至少有 4 个 CPU 周期。请注意，在连续的 CPU 写入期间，必须禁用中断。

21 eCAN：意外停止传输操作

21.1 问题描述

在极少数情况下，观察到来自 eCAN 模块的消息传输的停止（而接收操作继续正常进行）。

这种异常状态可能发生在总线上不会出现任何错误帧的时候。

21.2 建议

可以使用 eCAN 模块的超时功能（MOTO）来检测此情况。当发生这种情况时，设置并清除 CCR 位（使用 CCE 位进行验证），以消除异常情况。

22 内存：RAM 区域 L0、L1 不能混用

22.1 问题描述

在极少数情况下，ADP32F1X 系列芯片 L0、L1 RAM 区域出现程序和数据空间混用，芯片出现程序跑飞无法正常运行现象。

根本原因：器件 L0、L1 区域，在程序执行过程中，程序总线和数据总线不能同时访问，对同一个区域块（L0 或 L1）只能作为程序空间或数据空间使用。

22.2 建议

修改 CMD 文件，将 0x008000~0x008FFF(L0)区域全部分配到 PAGE0(程序空间)或 PAGE1(数据空间）。同理 0x009000~0x009FFF(L1)区域。

23 ADC：并发采样模式下，同序列两组采样值，B 通道采样值与 A 通道采样值偏差

23.1 问题描述

对比 ACQ_ps=0 及其它参数，ACQ_ps=0 配置条件下的 A、B 通道采样值偏差部分呈现阶梯型，个别通道相对一致性较好如 B1-B2-B3; 对比 ACQ_ps=5 及 ACQ_ps=10 测试结果，相同 ADC_clk 配置下增加采样开窗有利于稳定并发采样模式下 A、B 通道差值，且不同温度对应相同输入电压下的采样偏差上下限有变小趋势；但差值随着输入电压信号增加而变大的趋势无明显改善。

23.2 建议

相同 ADC_clk 配置下增加采样开窗（增大 ACQ_ps 设置值）；并发采样同序列 A 组通道采样值较接近真值，针对 B 通道采样值，用户在做采样值校准时，需要补偿 B 通道自身采样偏差。

24 时钟：Flash 运行速度配置超频导致程序跑飞

24.1 问题描述

Flash 随机等待时间、页等待时间配置小于 5 时，会造成程序跑飞现象。

24.2 建议

150MHz 主频下，确保 InitFlash()函数中 Flash 配置随机等待和页等待时间赋值为 5。

FlashRegs.FBANKWAIT.bit.RANDWAIT = 5;

FlashRegs.FBANKWAIT.bit.PAGEWAIT = 5;

联系方式

公司网址：**www.advancechip.com**

联系邮箱：**sales@advancechip.com**

销售联系电话：**0731-88731027**（长沙）

025-66051670（南京）

公司总部地址：长沙市高新开发区尖山路 **39** 号中电软件园总部大楼 **10** 楼

南京销售中心：南京市雨花台区软件大道 **106** 号 **2** 号楼 **802** 室