

Errata

MSPM0L112x および MSPM0L211x マイコン



概要

この文書では、機能仕様に対する既知の例外 (アドバイザリ) について説明します。

目次

1 機能アドバイザリ.....2

2 プログラム済みのソフトウェア アドバイザリ.....2

3 デバッグ専用のアドバイザリ.....3

4 コンパイラ アドバイザリによって修正.....3

5 デバイスの命名規則.....4

    5.1 デバイスの記号表記とリビジョンの識別.....4

6 アドバイザリの説明.....6

7 改訂履歴.....18

商標

すべての商標は、それぞれの所有者に帰属します。

## 1 機能アドバイザー

デバイスの動作、機能、またはパラメータに影響するアドバイザー。

✓ チェック マークは、指定されたリビジョンに問題が存在することを示します。

エラッタ番号	Rev B
<a href="#">AES_ERR_01</a>	✓
<a href="#">CPU_ERR_02</a>	✓
<a href="#">CPU_ERR_03</a>	✓
<a href="#">FLASH_ERR_03</a>	✓
<a href="#">FLASH_ERR_04</a>	✓
<a href="#">FLASH_ERR_05</a>	✓
<a href="#">FLASH_ERR_08</a>	✓
<a href="#">KEYSTORE_ERR_01</a>	✓
<a href="#">PMCU_ERR_13</a>	✓
<a href="#">RST_ERR_01</a>	✓
<a href="#">SYSCTL_ERR_01</a>	✓
<a href="#">SYSCTL_ERR_02</a>	✓
<a href="#">SYSCTL_ERR_03</a>	✓
<a href="#">SYSCTL_ERR_04</a>	✓
<a href="#">SYSOSC_ERR_01</a>	✓
<a href="#">SYSOSC_ERR_02</a>	✓
<a href="#">TIMER_ERR_04</a>	✓
<a href="#">TIMER_ERR_06</a>	✓
<a href="#">TIMER_ERR_07</a>	✓
<a href="#">UNICOMMI2CC_ERR_01</a>	✓
<a href="#">UNICOMMI2CT_ERR_01</a>	✓
<a href="#">UNICOMMI2CT_ERR_02</a>	✓
<a href="#">UNICOMMI2CT_ERR_03</a>	✓
<a href="#">UNICOMMSPI_ERR_01</a>	✓
<a href="#">UNICOMMUART_ERR_01</a>	✓
<a href="#">UNICOMMUART_ERR_02</a>	✓
<a href="#">UNICOMMUART_ERR_03</a>	✓
<a href="#">UNICOMMUART_ERR_04</a>	✓
<a href="#">UNICOMMUART_ERR_05</a>	✓
<a href="#">UNICOMMUART_ERR_06</a>	✓
<a href="#">UNICOMMUART_ERR_07</a>	✓
<a href="#">UNICOMMUART_ERR_08</a>	✓
<a href="#">UNICOMMUART_ERR_09</a>	✓
<a href="#">UNICOMMUART_ERR_10</a>	✓
<a href="#">UNICOMMUART_ERR_11</a>	✓

## 2 プログラム済みのソフトウェア アドバイザリ

工場出荷時にプログラムされたソフトウェアに影響を及ぼすアドバイザー。

✓ チェック マークは、指定されたリビジョンに問題が存在することを示します。

このカテゴリに該当するエラッタは、このデバイスには存在しません。

### 3 デバッグ専用のアドバイザリ

デバッグ動作のみに影響するアドバイザリ。

✓ チェック マークは、指定されたリビジョンに問題が存在することを示します。

このカテゴリに該当するエラッタは、このデバイスには存在しません。

### 4 コンパイラ アドバイザリによって修正

コンパイラの回避方法により解決されるアドバイザリ各アドバイザリについては、回避策が適用されている IDE およびコンパイラのバージョンを参照してください。

✓ チェック マークは、指定されたリビジョンに問題が存在することを示します。

## 5 デバイスの命名規則

製品開発サイクルの段階を示すため、TI はすべての MSP MCU デバイスの型番に接頭辞を割り当てています。MSP MCU 商用ファミリの各番号には、MSP、X のいずれかの接頭辞があります。MSP または XMS。これらの接頭辞は、製品開発の進展段階を表します。段階には、エンジニアリング プロトタイプ(XMS)から、完全認定済みの量産デバイス(MSP)までがあります。

**XMS** - 実験段階のデバイスであり、必ずしも最終製品の電气的特性を表しているとは限りません

**MSP** — 完全に認定済みの量産版デバイス

サポートツールの名前付けプレフィックス:

**X**: 開発サポート製品。テキサス・インスツルメンツの社内認定試験はまだ完了していません。

**null**: 完全に認定済みの開発サポート製品です。

**XMS** デバイスと **MSPX** 開発サポート ツールは、以下の免責事項に基づいて出荷されます:

「開発中の製品は、社内での評価用です。」

**MSP** デバイスの特性は完全に明確化されており、デバイスの品質と信頼性が十分に示されています。テキサス・インスツルメンツの標準保証が適用されます。

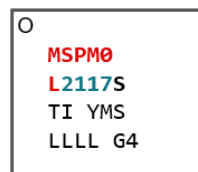
プロトタイプ デバイス (**XMS**) は、標準の量産デバイスよりも故障率が高いことが予想されます。これらのデバイスは、予測される最終使用時の故障率が未定義であるため、テキサス・インスツルメンツはそれらのデバイスを量産システムで使用しないよう推奨しています。認定済みの量産デバイスのみを使用する必要があります。

TI デバイスの項目表記には、デバイス ファミリの接尾辞も含まれます。この接尾辞は、温度範囲、パッケージタイプ、配布形式を示しています。

### 5.1 デバイスの記号表記とリビジョンの識別

次のパッケージ図はパッケージ記号化スキームを示すと同時に図 5-1、デバイスリビジョンからバージョン ID へのマッピングを定義しています。

**48QFN (RGZ):**



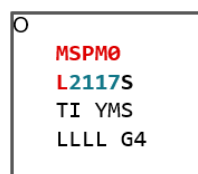
**64QFP (PM):**



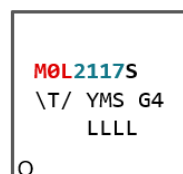
**32VSSOP (DGS32):**



**32QFN (RHB):**



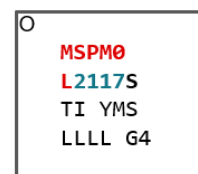
**48QFP (PT):**



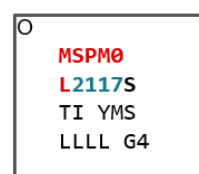
**28VSSOP (DGS28):**



**24QFN (RGE):**



**28QFN (RUY):**



TI = TI Letters  
\T/ = TI LOGO  
YM = Year Month Date  
S = Site code  
LLLL = Lot code  
T = "Q" character

LLLL = Lot code  
# = Die revision  
O = Pin 1 marked  
G4 = ECAT  
CAV = Cavity number

図 5-1. パッケージの記号表記

表 5-1. ダイ リビジョン

リビジョンレター	バージョン(デバイスの工場出荷時定数メモリ内)
B	1

## 6 アドバイザリの説明

<b>AES_ERR_01</b>	<b>AES モジュール</b>
<b>カテゴリ</b>	機能
<b>機能</b>	AES Saved Context Ready 割り込みが予想どおりに生成されていません
<b>説明</b>	Saved Context Ready 割り込みが生成されていません。いずれかの AES レジスタに対してアクセス(読み取りまたは書き込み)が行われた場合に、割り込みが生成されます。
<b>回避方法</b>	ポーリングベースのメカニズムを使用して、割り込みをせず、CTRL レジスタの保存済みコンテキストステディのステータスビットを確認します。
<b>CPU_ERR_02</b>	<b>CPU モジュール</b>
<b>カテゴリ</b>	機能
<b>機能</b>	CPUSS のプリフェッチ機能を無効にする制限
<b>説明</b>	保留中のフラッシュメモリアccessがある場合、CPU プリフェッチを無効にしても無効にはなりません。
<b>回避方法</b>	<p>プリフェッチャーを無効にし、SYSCTL でシャットダウンメモリへのメモリアccess (SHUTDNSTORE) を発行します。これは SYSCTL.SOCLOCK.SHUTDNSTORE0; で実行できます。</p> <p>メモリアccessが完了すると、プリフェッチャーは無効になります。</p> <p>例:            CPUSS.CTL.PREFETCH = 0x0、プリフェッチャーを無効にします            SYSCTL.SOCLOCK.SHUTDNSTORE0、シャットダウンメモリへのメモリアccess</p>
<b>CPU_ERR_03</b>	<b>CPU モジュール</b>
<b>カテゴリ</b>	機能
<b>機能</b>	低電力モードへの遷移時に、プリフェッチャが誤った命令を読み取る可能性がある
<b>説明</b>	低電力モードへ遷移する際に保留中のプリフェッチがある場合、プリフェッチャが誤って正しくないデータ(すべて 0)をフェッチする可能性があります。デバイスがウェイクアップした際、もしプリフェッチャおよびキャッシュが ISR コードによって上書きされない場合、フラッシュから実行されるメインコードが破損する可能性があります。たとえば、ISR が SRAM 内にある場合、フラッシュからプリフェッチされた誤ったデータは上書きされません。ISR から復帰する際に、プリフェッチャ内の破損したデータが CPU によってフェッチされ、誤った命令が実行されるおそれがあります。ハ

## CPU\_ERR\_03 (続き) CPU モジュール

ードウェア イベント ウェイクアップは、デバイスをウェイクアップするがプリフェッチャをフラッシュしないプロセスのもう 1 つの例です。

### 回避方法

低電力モードに入る前にプリフェッチャを無効にします。

例:

```
CPUSS.CTL.PREFETCH = 0x0; // プリフェッチャを無効化
SYSCTL.SOCLOCK.SHUTDNSTORE0; // シャットダウン メモリから読み出し
__WFI(); // または __WFE(); この関数は低電力モードへの遷移を呼び出す
CPUSS.CTL.PREFETCH = 0x1; // プリフェッチャを再有効化
```

## FLASH\_ERR\_03 FLASH モジュール

### カテゴリ

機能

### 機能

2 待機状態のフラッシュ アクセスの直後に無効なブートコード領域へのアクセスが行われると、次のフラッシュ アクセスでも違反が発生する可能性があります

### 説明

2 待機状態が設定されている状態で、フラッシュ アクセスの直後に **BOOTCODE** 領域へのアクセスを行うと、その次のフラッシュ アクセスでも違反が発生する可能性があります。

### 回避方法

ブート フェーズ終了後は、ブートコード領域へのアクセスを行わないでください。そうしない場合、ブートコード違反の後に正しいフラッシュ アクセスを行うまでに、少なくとも 4 クロック サイクルの間隔を空ける必要があります。

## FLASH\_ERR\_04 FLASH モジュール

### カテゴリ

機能

### 機能

NONMAIN または **Factory** 領域でエラーが発生した場合、SYSCTL\_DEDERRADDR に誤ったアドレスが報告されます

### 説明

FLASHDED エラーが発生すると、データの最上位バイト (**MSB**) が切り捨てられます。デバイスのメモリ制限では、最上位バイトは **MAIN** フラッシュの復帰アドレスに影響を与えません。**NONMAIN** フラッシュまたは **Factory** 領域の場合、MSB は 0x41xx.xxxx である必要があります。

### 回避方法

SysCtl\_DEDERRADDR の戻りアドレスで 0x00Cxxxxx が返る場合は、0x41000000 で OR 演算を実行して、NONMAIN または工場出荷時領域の復帰アドレスに適切なアドレスを取得します。たとえば、SYSCTL\_DEDERRADDR = 0x00C4013C の場合、実際のアドレスは 0x41C4013C となります。メインフラッシュ DED の場合、SYSCTL\_DEDERRADDR をそのまま使用できます。

## FLASH\_ERR\_05 *FLASH* モジュール

### カテゴリ

機能

### 機能

DEDERRADDR に誤ったリセット値が設定される可能性があります

### 説明

SYSCTL -> DEDERRADDR のリセット値では、正しい 0x00000000 のかわりに 0x00C4013C が返されることがあります。エラーが発生している場所はファクトリトリム領域であり、故障を示すものではありません。そのため、この値は無視して問題ありません。デバイスに **NONMAIN** をプログラムされると、リセット値が変化する傾向があります。

### 回避方法

0x00C4013C を別のリセット値として受け入れ、ブートからのデフォルト値を 0x00000000 または 0x00C4013C にすることができます。戻り値はデバイス上の **MAIN** フラッシュの範囲外であるため、実際のフラッシュ DED ステータスから返された可能性はありません。

## FLASH\_ERR\_08 *FLASH* モジュール

### カテゴリ

機能

### 機能

通常の無効なメモリ領域に対してハード フォルトは生成されません

### 説明

不正なメモリ アドレス空間へのアクセス中は、以下に示すようにハード フォルトは生成されません。1. 0x010053FF ~ 0x20000000 2. 0x40BFFFFFF ~ 0x41C00000 3. 0x41C007FF ~ 0x41C40000

### 回避方法

番号

## KEYSTORE\_ERR\_01 *キーストアモジュール*

### カテゴリ

機能

### 機能

STATUS.STAT の値は、キーアクセスがない場合、0 または 1 になります。

### 説明

STATUS.STAT のリセット値は 1 で、以下の条件で 0 になります。1. リセット後、レジスタウィンドウを介したデバッガアクセスは 0x00 を返します。2. リセット後、最初の CPU 読み取りは 0x01 を返し、その後の CPU 読み取りは 0x00 を返します。3) リセット後、最初に他の キーストアレジスタを読み取り、次に STATUS.STAT を読み取ると、0x00 が返されます。

### 回避方法

STATUS.STAT=0x0 は「エラーなし」を意味します。スロットが有効かどうか (キーが存在するかどうか) を確認するには、STATUS.VALID を確認してください。



<b>PMCU_ERR_13</b>	<b>PMCU モジュール</b>
<b>カテゴリ</b>	機能
<b>機能</b>	STOP2 または STANDBY0 からのウェークアップ時に MCU がスタックする可能性があります
<b>説明</b>	デバイスが STOP2 または STANDBY に移行するときにプリフェッチ アクセスが保留されている場合、デバイスがウェークアップしたときに、保留中のプリフェッチにより、デバイスが通常実行を再開できない可能性があります。エラッタは、WFI 命令がワードアライメントされておらず、フラッシュの待機ウェイト状態が 2 の場合に発生します。このような場合、DMA 転送も保留中の割り込みも処理されません。
<b>回避方法</b>	ユーザーはプリフェッチを無効化し、シャットダウン ストア メモリ読み取りを発行する必要があります。これにより、新しいプリフェッチが発行されなくなり、保留中のプリフェッチを完了させることができます。
<b>RST_ERR_01</b>	<b>RST モジュール</b>
<b>カテゴリ</b>	機能
<b>機能</b>	LFCLK_IN が LFCLK のソースとして選択されており、かつ LFCLK_IN が無効になっている場合、NRST リリースは検出されません
<b>説明</b>	LFCLK = LFCLK_IN で、LFCLK_IN を無効にすると、NRST パルスエッジ検出を見逃されし、デバイスがリセットから復帰しないコーナーシナリオが発生します。この問題は、NRST パルス幅が 608µs 未満のときに見られます。NRST パルスが 608µs を超える場合は、リセットは通常どおり表示されます。
<b>回避方法</b>	この問題を回避するため、608µs よりも高い NRST パルス幅を維持します。
<b>SYSCTL_ERR_01</b>	<b>SYSCTL モジュール</b>
<b>カテゴリ</b>	機能
<b>機能</b>	SW-POR 機能は、HW_POR と組み合わせて使用できます
<b>説明</b>	ソフトウェアトリガ POR を生成するために正しいキーを使って LFSSRST レジスタに書き込むと、RSTCAUSE レジスタには、予測される 0x3 (ソフトウェアトリガ POR) ではなく 0x2 (NRST トリガ POR) が表示されます。これは、SW-POR 機能が HW-POR パスと組み合わされているためです。
<b>回避方法</b>	番号

## SYSCTL\_ERR\_02 SYSCTL モジュール

---

### カテゴリ

機能

### 機能

BOOTRST の後には、SYSSTATUS.FLASHSEC はゼロ以外になります

### 説明

BOOTRST/ブートコード完了後、SYSSTATUS.FLASHSEC はゼロ以外になります。これは、お客様がブートコードが完了した後に表示されます。

### 回避方法

番号

## SYSCTL\_ERR\_03 SYSCTL モジュール

---

### カテゴリ

機能

### 機能

DEDERRADDR は、SYSRESET または SYSSTATUSCLR への書き込みの後にも持続します

### 説明

SYSRESET または SYSSTATUSCLR レジスタへの書き込みの後も、DEDERRADDR は持続します。この値は、新しい FLASHDED エラーが発生した場合にのみ書き込まれます。この挙動は、初期リセット値をゼロに規定されているテクニカル リファレンス マニュアル (TRM) に矛盾します。

### 回避方法

なし

## SYSCTL\_ERR\_04 SYSCTL モジュール

---

### カテゴリ

機能

### 機能

SYSRESET 後に SYSSTATUS.FLASHSEC がクリアされません。

### 説明

SYSSTATUS.FLASHSEC は、SYSRESET 後にクリアされず、SYSSTATUSCLR レジスタに書き込まれることでのみクリアされます。

### 回避方法

番号

## SYSOSC\_ERR\_01 SYSOSC モジュール

---

### カテゴリ

機能

### 機能

STOP1 モードと SYSOSC の FCL を併用すると、MFCLK にドリフトが発生する可能性があります

## SYSOSC\_ERR\_01

(続き)

### SYSOSC モジュール

#### 説明

MFCLK が有効になっており、SYSOSC が周波数補正ループ (FCL) モードを使用しており、STOP1 の低電力動作モードが使用されている場合、SYSOSC が 4MHz から 32MHz に切り替わる際 (STOP1 モードから RUN モードへの移行時、または SYSOSC を 32MHz に強制する非同期の高速クロック要求時)、MFCLK が 2 サイクル分ドリフトする可能性があります。

#### 回避方法

STOP1 モードではなく STOP0 モードを使用します。STOP0 モードを使用する場合、MFCLK ドリフトは発生しません。

または

STOP1 を使用する場合は、FCL モードで SYSOSC を使用しないでください (FCL を無効のままにしておきます)。

## SYSOSC\_ERR\_02 SYSOSC モジュール

#### カテゴリ

機能

#### 機能

SYSOSC が FCL モードで無効化されている LPM 中に非同期クロック要求を受信しても、MFCLK は動作しません

#### 説明

以下のシナリオでは、MFCLK はトグルを開始しません:

- 1.FCL モードを有効にした後、MFCLK を有効にします
- 2.SYSOSC が無効になる低消費電力モードに移行します (SLEEP2/STOP2/STANDBY0/STANDBY1)。
- 3.MFCLK を機能クロックとして使用する一部のペリフェラルから非同期要求を受信されます。ASYNC 要求を受信すると、SYSOSC は有効になり、ulpclock は 32MHz になります。ただし、デバイスが依然として LPM に設定されているため、MFCLK はゲートオフの状態となり、一切トグルしません。

#### 回避方法

SYSOSC が FCL モードを使用している場合は、通常 SYSOSC がオフになる LPM モードへ移行する際に、ペリフェラル用の MFCLK を有効にしないでください。

## TIMER\_ERR\_04 TIMER モジュール

#### カテゴリ

機能

#### 機能

TIMER をゼロ イベントの直前に再有効化すると、再有効化が失われる可能性があります

#### 説明

タイマーをワンショット モードで使用している場合、ゼロ イベント付近で再有効化を行うと再有効化が失われる可能性があります。タイマー有効ビットのハードウェア更新には、1 機能クロック サイクルが必要です。たとえば、タイマーのクロック ソースが 32.768kHz で、クロック分周比が 3 の場合、有効ビットが正しく 0 に設定されるまでに約 100μs かかります。

**TIMER\_ERR\_04 (続  
き)**
***TIMER* モジュール**


---

**回避方法**

タイマーを再有効化する前に 1 機能クロック サイクル分待機するか、一度タイマーを無効化してから再度有効化してください。

CTRCTL.EN = 0 でカウンタを無効化してから、CTRCTL.EN = 1 で再度有効化します

**TIMER\_ERR\_06**
***TIMG* モジュール**


---

**カテゴリ**

機能

**機能**

CLKEN ビットに 0 を書き込んでも、カウンタは無効化されません

**説明**

カウンタ クロック制御レジスタ (CCLKCTL) のクロック イネーブル ビット (CLKEN) に 0 を書き込んでも、タイマは停止しません。

**回避方法**

カウンタ制御 (CTRCTL) イネーブル (EN) ビットに 0 を書き込むことで、タイマを停止します。

**TIMER\_ERR\_07**
***TIMG* モジュール**


---

**カテゴリ**

機能

**機能**

初期リピート カウンタの周期は、次のリピートより 1 回だけ少なくなる

**説明**

タイマ リピート カウンタ モードを使用する場合、以下のリピート カウンタには 0 とロード値の間の遷移が含まれるため、最初のリピートのカウントは後続のリピートより 1 回少なくなります。たとえば、TIMx.RCLD = 0x3 の場合、観測可能な 3 つのゼロ イベントが最初のリピート カウンタに現れ、観測可能な 4 つのゼロ イベントが後続するリピート カウンタ シーケンスに現れます。

**回避方法**

初期 RCLD 値を想定される RCLD より 1 だけ大きく設定し、リピート カウンタ ゼロ イベント (REPC) の ISR 内で、RCLD を目的の値に設定します。

たとえば、4 回の繰り返しを行う場合は、初期 RCLD 値を RCLD = 0x5 に設定し、REPC 割り込み用のタイマ ISR 内で、RCLD = 0x4 に設定します。これで、すべてのタイマーの繰り返しで、ゼロ/ロード イベントの数が同一になります。

**UNICOMMI2CC\_E  
RR\_01**
***UNICOMMI2CC* モジュール**


---

**カテゴリ**

機能

**機能**

I2C BUSY ビットのポーリングでは、コントローラ転送の完了を確実に保証できない場合がある

## UNICOMMI2CC\_E RR\_01 (続き)

### UNICOMMI2CC モジュール

#### 説明

BUSRTRUN/FRAME\_START ビットを設定して I2C コントローラ転送を開始した後、BUSY ステータスがアサートされるまでに、約 2 クロック分の I2C 機能クロックサイクルを要します。転送完了を待つように BUSRTRUN/FRAME\_START を設定した直後に BUSY ビットをポーリングすると、BUSY ステータスが設定される前にチェックされることがあります。この問題は、CLKDIV の値が大きい (その結果、I2C 機能クロックが遅くなる) 場合やコンパイラ最適化レベルが高い場合に、発生する可能性がより高くなります。

#### 回避方法

BUSY ステータスをポーリングする前に、ソフトウェア遅延を追加します。ソフトウェア遅延 =  $3 \times \text{I2C 機能クロック} = 3 \times \text{clock\_divider} \times (\text{CPU\_CLK} / \text{選択したクロックソース周波数})$ 。たとえば、clock\_divider が 8、クロックソースが 4MHz(MFCLK)、CPU\_CLK が 32MHz の場合: ソフトウェア遅延 =  $3 \times 8 \times (32\text{MHz} / 4\text{MHz}) = 192 \text{ CPU サイクル}$

## UNICOMMI2CT\_ER R\_01

### UNICOMMI2CT モジュール

#### カテゴリ

機能

#### 機能

CTR が別々に書き込まれると、I2C ターゲットはアイドル状態を保持します

#### 説明

I2C ターゲットは、次のシナリオでアイドル状態を維持します: 1. I2C は MFCLK をクロックソースとして使用します。2. CLKDIV = 0。3. 最初のレジスタ書き込みで、CTR.START、CTR.STOP、または CTR.BLEN ビットを設定します。4. 2 回目のレジスタ書き込みで、CTR.FRM\_START を設定します。

#### 回避方法

I2C がクロックソースとして MFCLK を使用し、かつ CLKDIV = 0 の場合は、CTR.START、CTR.STOP、CTR.BLEN、または CTR.FRM\_START の各ビットを 1 回のレジスタ書き込みで設定することを推奨します。

## UNICOMMI2CT\_ER R\_02

### UNICOMMI2CT モジュール

#### カテゴリ

機能

#### 機能

I2C のクロックが CPU のクロックよりもはるかに低い場合に、CPU が RXFIFO を読み取る際に失敗します

#### 説明

UNICOMMI2CT の RX モードでは、受信フレームの最後のビットが受信された直後に RXDONE 割り込みフラグがアサートされます。ただし、RXFIFO バッファの内容更新は、RXDONE のアサートに対して 2 クロック サイクルのレイテンシを伴います。RXDONE フラグがトリガされてから 2 UNICOMMI2CT クロック サイクル以上経過して CPU が RXFIFO の読み取りを開始した場合、古い FIFO データにアクセスしてしまい、データ整合性の違反が発生する可能性があります。これは、UNICOMMI2CT のクロックが CPU クロックよりも大幅に低い場合に、特に問題となります。

**UNICOMMI2CT\_ER****R\_02 (続き)****UNICOMMI2CT モジュール****回避方法**

RXFIFO を読み取る前に、固定の遅延を追加します。

**UNICOMMI2CT\_ER****R\_03****UNICOMMI2CT モジュール****カテゴリ**

機能

**機能**

7 ビット モードでは、SR.ADDRMATCH よりも先に TSTART フラグがセットされます

**説明**

UNICOMMI2CT バスでは、SR.ADDRMATCH ステータスが更新される 3 バス クロック サイクル前に、RIS.TSTART フラグが設定されます。ただし、TSTART がアサートされてから CPU が ADDRMATCH を読み取るまでの間隔が UNICOMMI2CT クロックで 2 サイクルを超えると、CPU が古い ADDRMATCH 値を読み取ってしまい、結果としてエラーが発生する可能性があります。

**回避方法****UNICOMMSPI\_ER****R\_01****UNICOMMSPI モジュール****カテゴリ**

機能

**機能**

TXFIFO への読み書きが同時に行われた場合、SPI のアンダーフロー イベントが生成されない可能性があります。

**説明**

UNICOMMSPI の TXFIFO では、以下の条件が同時に発生するクリティカル ウィンドウ中に、アンダーフロー イベントが生成されない場合があります: 1. CPU または DMA が TXFIFO 2 にデータを書き込みます。SPI ペリフェラルが送信データを送るために TXFIFO を読み出している場合、アンダーフロー イベント機能の使用は推奨されません。ユーザーは、CPU または DMA が TXFIFO にアクセスする際に、UNICOMMSPI の TXFIFO が決して空にならないように確保する必要があります。

**回避方法**

なし

**UNICOMMUART\_E****RR\_01****UNICOMMUART モジュール****カテゴリ**

機能

**機能**

IrDA モードでグリッチが発生した場合のデータ完全性の問題

## UNICOMMUART\_E

RR\_01 (続き)

**UNICOMMUART** モジュール

### 説明

UNICOMM-UART は IrDA をサポートしていますが、ノイズの多い環境では制限があります。グリッチ フィルタがないため、データラインにグリッチが発生すると、IrDA モードではデータ整合性の問題が発生する可能性があります。IrDA はエッジ検出ロジックに依存しているため、これらのグリッチが有効なパルスとして誤認識され、その結果 FIFO に予期しないデータが入り、LTOUT の計算に問題が生じます。

### 回避方法

なし

## UNICOMMUART\_E

RR\_02

**UNICOMMUART** モジュール

### カテゴリ

機能

### 機能

IDLELINE モードにおけるグリッチによる IDLE 期間検出の問題

### 説明

アイドル期間中にグリッチが発生すると、受信側のアイドル検出が乱され、その結果、後続のアドレス バイトを Rx 側が破棄してしまう可能性があります。重要なのは、この故障モードがタイミング依存である点です。アイドル期間に対して特定のタイミングでグリッチが挿入された場合にのみ、この問題が発生します。それで、アイドル期間内のすべてのグリッチが破損を引き起こすわけではありません。

### 回避方法

なし

## UNICOMMUART\_E

RR\_03

**UNICOMMUART** モジュール

### カテゴリ

機能

### 機能

マンチェスター モードでのグリッチによるデータ完全性の問題

### 説明

マンチェスター符号化されたデータ ストリーム中のグリッチにより、信号の正確なエッジ タイミングが乱され、誤った受信データやデータ整合性の問題が発生します。

### 回避方法

なし

## UNICOMMUART\_E

RR\_04

**UNICOMMUART** モジュール

### カテゴリ

機能

### 機能

LIN モードでは、ブレーク フィールドおよび / または同期フィールド中にグリッチが発生すると、データが欠落します

**UNICOMMUART\_E****RR\_04 (続き)****UNICOMMUART モジュール****説明**

ブレーク フィールドのシナリオ (ハイパルス グリッチ) : ブレーク フィールド期間中に負のエッジが発生すると、カウンタがゼロにリセットされ、ブレーク フィールドの検出に失敗し、その結果データが失われます。同期フィールドのシナリオ: 同期フィールド中にグリッチが発生すると、誤った LINC0/LINC1 割り込みがトリガされ、Rx ラインの負のエッジでカウンタがリセットされ、同期フィールドの検証に失敗し、その結果データが失われます。

**回避方法**

なし

**UNICOMMUART\_E****RR\_05****UNICOMMUART モジュール****カテゴリ**

機能

**機能**

ラインがアイドル状態のときにグリッチが発生すると、LTOUT/RTOUT の期間計算に問題が生じます

**説明**

通常の UART モードでは、アイドル ライン状態中にグリッチが発生すると、LTOUT イベントが 1 ~2 ボー クロック分ずれて発生し、LTOUT のタイミングが乱されます。

**回避方法**

なし

**UNICOMMUART\_E****RR\_06****UNICOMMUART モジュール****カテゴリ**

機能

**機能**

ストップ ビットの長さが一致していないため、RTOUT/LTOUT 周期の計算に問題が発生します

**説明**

レシーバ側では、機能ステート マシンがストップ ビットの途中でストップ ビット状態からアイドル状態へ遷移するように設計されています。このため、RTOUT カウンタが本来開始されるべきタイミングよりも早くカウントを開始してしまいます。この結果、RTOUT の期間が不正確となり、RTOUT 割り込みが想定よりも早く発生します。

**回避方法**

RTOUT カウンタに、ストップ ビット半周期分の補正を追加します

**UNICOMMUART\_E****RR\_07****UNICOMMUART モジュール****カテゴリ**

機能

**機能**

RS-232 モードでは、UART が無効化されている場合、RTS ラインは HIGH になりません



## UNICOMMUART\_E

### RR\_07 (続き) *UNICOMMUART モジュール*

#### 説明

UART が無効化されている場合、RTS ラインはアイドル状態 (HIGH) に戻らず、LOW のまま固定されます。

#### 回避方法

ソフトウェアを使用して内部プルアップ抵抗を有効にし、RTS ラインの IO を Hi-Z モードに設定します。

## UNICOMMUART\_E

### RR\_08 *UNICOMMUART モジュール*

#### カテゴリ

機能

#### 機能

LTOUT 割り込みは、タイムアウトが期限切れになるたびに継続して発生します

#### 説明

1 回目の LTOUT 割り込みが発生した後、カウンタは再スタートしてカウントを継続し、その間に Rx フレームが存在しなくても、再び LTOUT 割り込みを発生させます。その結果、タイムアウト満了間隔ごとに LTOUT 割り込みが連続して発生することになります。この動作は RTOUT の挙動とは異なります。RTOUT 割り込みは、新たな Rx トランザクションが行われるまで一度しか発生しません。この制約は、RTOUT と LTOUT の両イベントで同一のカウンタを共有して使用していることに起因しています。

#### 回避方法

## UNICOMMUART\_E

### RR\_09 *UNICOMMUART モジュール*

#### カテゴリ

機能

#### 機能

ISO-7816 スマートカード モードは、57MHz UARTCLK 未満の 9600 ボーレートをサポートできません

#### 説明

ISO-7816 スマートカード モードで 9600 のボーレートを実現するには、以下の制限を考慮して、57MHz を超える UARTCLK 周波数が必要です。1.ISO-7816 規格では、1 ビットに 372 クロック サイクルが必要であると規定されています。2.MSPM0 ISO-7816 モードでは、オーバーサンプリング レート (OVS) は UART ペリフェラルで 16x に固定されています。最小 UARTCLK の計算式を以下に示します。必要な  $\text{UARTCLK} = 9600 * 372 * 16 = 57.139\text{MHz}$

#### 回避方法

なし

## UNICOMMUART\_E

### RR\_10 *UNICOMMUART モジュール*

#### カテゴリ

機能

**UNICOMMUART\_E****RR\_10 (続き)** *UNICOMMUART モジュール***機能**

CLKDIV が 11 に設定されている場合にのみ、LIN レジスタにアクセスできます

**説明**

UNICOMMUART (UART + LIN) バリエントでは、CLKDIV が 1 に設定されている場合にのみ、LIN レジスタの設定が有効に維持されます。CLKDIV 値が 1 以外の場合、これらの設定は破棄されます。

**回避方法**

なし

**UNICOMMUART\_E****RR\_11** *UNICOMMUART モジュール***カテゴリ**

機能

**機能**

UNICOMMUART が無効化されており、かつ txfito にデータが存在する場合、STAT.BUSY ビットは high のままになります

**説明**

UNICOMMUART の電源が無効化されており、かつ txfito にデータが存在する場合、STAT.BUSY ビットは high のままになります。

**回避方法**

すべての UNICOMMUART レジスタを初期化するには、UNICOMMUART をリセットします。

**7 改訂履歴**

資料番号末尾の英字は改訂を表しています。その改訂履歴は英語版に準じています。

日付	改訂	注
November 2025	*	初版リリース

## 重要なお知らせと免責事項

TI は、技術データと信頼性データ (データシートを含みます)、設計リソース (リファレンス デザインを含みます)、アプリケーションや設計に関する各種アドバイス、Web ツール、安全性情報、その他のリソースを、欠陥が存在する可能性のある「現状のまま」提供しており、商品性および特定目的に対する適合性の黙示保証、第三者の知的財産権の非侵害保証を含みいかなる保証も、明示的または黙示的にかかわらず拒否します。

これらのリソースは、TI 製品を使用する設計の経験を積んだ開発者への提供を意図したものです。(1) お客様のアプリケーションに適した TI 製品の選定、(2) お客様のアプリケーションの設計、検証、試験、(3) お客様のアプリケーションに該当する各種規格や、その他のあらゆる安全性、セキュリティ、規制、または他の要件への確実な適合に関する責任を、お客様のみが単独で負うものとし、TI は一切の責任を拒否します。

上記の各種リソースは、予告なく変更される可能性があります。これらのリソースは、リソースで説明されている TI 製品を使用するアプリケーションの開発の目的でのみ、TI はその使用をお客様に許諾します。これらのリソースに関して、他の目的で複製することや掲載することは禁止されています。TI や第三者の知的財産権のライセンスが付与されている訳ではありません。お客様は、これらのリソースを自身で使用した結果発生するあらゆる申し立て、損害、費用、損失、責任について、TI およびその代理人を完全に補償するものとし、TI は一切の責任を拒否します。

TI の製品は、[TI の販売条件](#)、[TI の総合的な品質ガイドライン](#)、[ti.com](https://www.ti.com) または TI 製品などに関連して提供される他の適用条件に従い提供されます。TI がこれらのリソースを提供することは、適用される TI の保証または他の保証の放棄の拡大や変更を意味するものではありません。TI がカスタム、またはカスタマー仕様として明示的に指定していない限り、TI の製品は標準的なカタログに掲載される汎用機器です。

お客様がいかなる追加条項または代替条項を提案する場合も、TI はそれらに異議を唱え、拒否します。

Copyright © 2026, Texas Instruments Incorporated

最終更新日：2025 年 10 月