参考資料

Errata

MSPM0G321x/MSPM0G320x ミックスド シグナル マイコン

TEXAS INSTRUMENTS

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

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。

エラッタ番号	リビジョン A
AES_ERR_01 AES モジュール	✓
CPU_ERR_02 CPU モジュール	✓
CPU_ERR_03 CPU モジュール	✓
FLASH_ERR_03 FLASH モジュール	✓
FLASH_ERR_04 FLASH モジュール	✓
FLASH_ERR_05 FLASH モジュール	✓
FLASH_ERR_08 FLASH モジュール	✓
GPIO_ERR_05 GPIO モジュール	✓
GPIO_ERR_06 GPIO モジュール	✓
KEYSTORE_ERR_01 キーストアモジュール	✓
MATHACL_ERR_01 MATHACL モジュール	✓
PMCU_ERR_13 PMCU モジュール	✓
RST_ERR_01 RST モジュール	✓
SYSCTL_ERR_01 SYSCTL モジュール	✓
SYSCTL_ERR_02 SYSCTL モジュール	✓
SYSCTL_ERR_03 SYSCTL モジュール	✓
SYSCTL_ERR_04 SYSCTL モジュール	✓
SYSCTL_ERR_05 LFCLK モジュール	
SYSCTL_ERR_06 CLK_OUT モジュール	
SYSOSC_ERR_01 SYSOSC モジュール	✓
SYSOSC_ERR_02 SYSOSC モジュール	✓

エラッタ番号	リビジョン A
SYSOSC_ERR_04 SYSOSC モジュール	✓
SYSPLL_ERR_01 SYSPLL モジュール	✓
TIMER_ERR_04 TIMER モジュール	✓
TIMER_ERR_06 TIMG モジュール	✓
TIMER_ERR_07 初期リピート カウンタの周期は、次のリピート モジュールより 1 回だけ少なくなる	✓
UNICOMMI2CC_ERR_01 UNICOMMI2CC モジュール	
UNICOMMUART_ERR_06 UNICOMMUART モジュール	✓
UNICOMMUART_ERR_07 UNICOMMUART モジュール	✓
UNICOMMUART_ERR_09 UNICOMMUART モジュール	
UNICOMMUART_ERR_10 UNICOMMUART モジュール	✓
VREF_ERR_05 VREF モジュール	✓

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。

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。

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気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 ファミリ名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。

6 アドバイザリの説明

AES_ERR_01 *AES モジュール*

カテゴリ

機能

機能

AES Saved Context Ready 割り込みが予想どおりに生成されていません

説明

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。

回避方法

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レディのステータスビットを確認します。

CPU_ERR_02 *CPU モジュール*

カテゴリ

機能

機能

CPUSS のプリフェッチ機能を無効にする制限

説明

保留中のフラッシュメモリアクセスがある場合、CPU プリフェッチを無効にしても無効にはなりません。

回避方法

プリフェッチを無効にした後、SYSCTL のシャットダウンメモリ（SHUTDNSTORE）へのメモリアクセスを実行してください。これは SYSCTL->SOCLOCK.SHUTDNSTORE0 にアクセスすることで行えます。メモリアクセスが完了すると、プリフェッチャは無効化されます。

CPU_ERR_03 *CPU モジュール*

カテゴリ

機能

機能

プリフェッチャは、SLEEP モードへの遷移時にデータ整合性の問題を引き起こす可能性があります

説明

SLEEP0 に移行するとき、プリフェッチャで不正なデータ（すべて 0）が誤ってフェッチされることがあります。スリープモードから復帰したときに、プリフェッチャとキャッシュが ISR コードによって上書きされない場合、フラッシュからのメインコード実行が破損するおそれがあります。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。

回避方法

SLEEP に入る前にプリフェッチャーを無効にします。

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

誤ったアドレスが SYSCTL -> DEDERRADDR で報告される

説明

FLASHDED エラーが表示されると、データの最上位バイト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 フラッシュまたは工場出荷時領域では、MSB は 0x41xx.xxxx と表記されます

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、実数アドレスは 0x41C4013C です。

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 をプログラムすされると、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。0x40BFFFFF ～ 0x41C00000 3。0x41C007FF～ 0x41C40000

回避方法

番号

GPIO_ERR_05 *GPIO モジュール*

カテゴリ

機能

機能

DMA 転送が進行中の間に、GPIO DOUTTGL レジスタへの書き込みが失われる可能性があります

説明

同時 DMA 転送が進行中の間に、DOUTTGL レジスタへの CPU 書き込みに GPIO DMAMASK レジスタ情報が誤って適用されます。

回避方法

アプリケーション コードで、DOUTTGL レジスタへの CPU 書き込みアクセスが発行される前に、DOUTTGL レジスタの対応するビットに対して GPIO DMAMASK ビットが 1 に設定されていることを確認します。GPIO レジスタへの DMA 転送が不要な場合は、IO 初期化ステップ中に GPIO DMAMASK を 0xFFFFFFFF として構成できます。これにより、このエラッタの競合が解決されます。アプリケーションで GPIO レジスタへの DMA 書き込み転送も必要な場合は、アプリケーションで DMA と CPU の両方を使用して、デバイス内の同じ GPIO モジュールの DOUTTGL レジスタに書き込むことを推奨します。デバイスに複数の GPIO モジュールがある場合、DMA と CPU は、異なる GPIO モジュールの DOUTTGL レジスタに同時に書き込むことができます (一方で、CPU が書き込み先の GPIO モジュール用に GPIO DMAMASK を構成する必要もあります)。

GPIO_ERR_06 *GPIO モジュール*

カテゴリ

機能

機能

DMA 転送が進行中の間に、GPIO DOUT、DOUTSET、DOUTCLR レジスタへの書き込みが失われる可能性があります

説明

GPIO DOUT、DOUTSET、DOUTCLR レジスタには DMA からアクセスできません。実装の誤りにより、同時 DMA 転送が進行中の間に、GPIO DOUT、DOUTSET、DOUTCLR への CPU アクセスもブロックされます。

GPIO_ERR_06 (続き)

GPIO モジュール

回避方法

アプリケーション コードでは、DOUT、DOUTSET、DOUTCLR レジスタに書き込む代わりに、ソフトウェアは DOUTTGL レジスタに対して等価な書き込みを実行する必要があります (DOUTTGL レジスタへの CPU 書き込みの制限については、「回避方法 GPIO_ERR_05」を参照)。

以下の疑似コードでは、「ピン」は、構成する GPIO モジュールのピンのビット ベクトルを示しています。

```
DL_GPIO_setPins(GPIO_Regs* gpio, uint32_t pins)
{
gpio->DOUTTGL31_0 = ~(gpio->DOUT31_0) & pins;
}

DL_GPIO_clearPins(GPIO_Regs* gpio, uint32_t pins)
{
gpio->DOUTTGL31_0 = gpio->DOUT31_0 & pins;
}

DL_GPIO_writePins(GPIO_Regs* gpio, uint32_t pins)
{
gpio->DOUTTGL31_0 = ~(gpio->DOUT31_0) & pins;
gpio->DOUTTGL31_0 = gpio->DOUT31_0 & (~pins);
}

DL_GPIO_writePinsVal(GPIO_Regs* gpio, uint32_t pinsMask, uint32_t pinsVal)
{
uint32_t doutVal = gpio->DOUT31_0;
doutVal &= ~pinsMask;
doutVal |= (pinsVal & pinsMask);
gpio->DOUTTGL31_0 = ~(gpio->DOUT31_0) & doutVal;
gpio->DOUTTGL31_0 = gpio->DOUT31_0 & (~doutVal);
}
```

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。

MATHACL_ERR_01

MATHACL モジュール

カテゴリ

機能

機能

MATHACL ステータスエラービットはクリアされません

概要

mathacl によってステータスエラーが生成された場合（例：0 で除算）、STATUS レジスタがはクリアされません。

回避方法

ペリフェラルをリセットして、STATUS ビットをクリアします。

PMCU_ERR_13

PMCU モジュール

カテゴリ

機能

機能

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

説明

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。

回避方法

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。

RST_ERR_01

RST モジュール

カテゴリ

機能

機能

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

説明

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。

回避方法

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。

SYSCTL_ERR_01 *SYSCTL モジュール*

カテゴリ

機能

機能

SW-POR 機能は、HW_POR と組み合わせて使用できます

説明

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。

回避方法

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

詳細

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上書きされます。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

回避方法はありません。

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。

SYSCTL_ERR_04 (続き) ***SYSCTL モジュール***

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

SYSCTL_ERR_05 ***LFCLK モジュール***

カテゴリ

機能

機能

シャットダウンからの終了時に LFCLK が動作しない

説明

LFCLK_IN ピンがプルアップ付きの汎用入力 (または) LFCLK_IN 機能として構成されている場合、この構成では、シャットダウン モードを終了すると、LFCLK がスタックします。

回避方法

次のいずれかを選択:1. この LFCLK_IN I/O では、プルアップの代わりにプルダウンをイネーブルにする、2. 入力として構成するのを避ける

SYSCTL_ERR_06 ***CLK_OUT モジュール***

カテゴリ

機能

機能

非同期クロックが CLK_OUT ソースとして選択されているときに、ユーザーが CLK_OUT をディスエーブルにすると、外部クロック出力 (CLK_OUT) でグリッチが発生する可能性があります

説明

CLK_OUT のクロック ソースが現在のバス クロックと非同期である場合 (たとえば、バス クロックが SYSOSC で、CLK_OUT のクロック ソースが LFCLK に選択されている場合)、この場合、CLK_OUT ピンが一定時間イネーブルになってからディスエーブルになるときにグリッチが発生することがあります

回避方法

回避方法はありません。

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

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

SYSOSC_ERR_01
(続き)

SYSOSC モジュール

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 サイクル分ずれる可能性があります。

回避方法

Workaround1：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。Workaround2：STOP1 を使用する場合、SYSOSC を FCL モードで使用しないでください(FCL はディセーブルのままにします)。

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が受信されます。
ASYNC 要求を受信すると、SYSOSC は有効になり、ulpclk 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。

SYSOSC_ERR_04 ***SYSOSC モジュール***

カテゴリ

機能

機能

SYSPLL を使用する場合、FCL ON モードで SYSOSC の精度が低下します

説明

内部発振器 SYSOSC に FCLON モードを使用する場合、FCL ON で SYSPLL を使用すると精度が最大 ±3% 低下する可能性があります。精度の低下は、4MHz SYSOSC サンプリング クロックと、システム内のノイズとの間の同期に起因します。

回避方法

SYSPLL FCL ON モードで使用する場合は、次のように SYSPLL 周波数に –4MHz 以外の倍数を使用します。78MHz

SYSPLL を 16、32、48、64、80MHz などにしないでください。

78MHz の場合：
SYSPLLCFG1.PDIV = 0x3、SYSPLLCFG1.QDIV を 38 に設定します

SYSPLL_ERR_01 *SYSPLL モジュール*

カテゴリ

機能

機能

SYSPLL 周波数が有効になっているとき、正しい周波数にロックされない場合がある。

説明

SYSCTL.HSCLKEN レジスタ内の SYSPLLEN ビットを 1 に設定すると、SYSPLL は位相同期ループ のサーチを実行します。周波数が正しい値に設定されないと、サーチ動作が失敗することがあります。その場合は、得られる周波数が設定値と大きく異なってしまいます。

回避方法

周波数確認プロセス
SYSPLLEN ビットを 1 に設定した場合は常に、周波数クロック カウンタ (FCC) を使用して SYSPLL の出力周波数を監視します。正しい周波数が確立されると、SYSPLL がディスエーブルになり、再度イネーブルになるまで、その周波数は安定した状態を維持します (SYSPLLEN ビットを 0 から 1 にトグル)。誤った周波数が検出された場合は、SYSPLL をディスエーブルにしてから再度イネーブルにして、別の検証を実行します。

回避方法 1:FCC カウントチェック
SYSPLL 出力クロック周波数をカウントするには、FCC トリガ ソースとして LFCLK を使用します。FCC を実行し、LFCLK をリファレンスとして使用して、設定された SYSPLL 周波数に対して測定値を検証します。
計算例:
- SYSPLLCLK0 = 80MHz ; LFCLK = 32.768kHz
- 測定 FCC カウント = 80,000,000/32,768 = 2,441
FCC カウント許容誤差:
実際の FCC 数は、合計クロック精度 (SYSPLLCLK0 および LFCLK) によって異なります。FCC チェック範囲を許可するには、±5 ～ 10% を追加することを推奨します。
- FCC カウント上限 = 2,441*1.05 = 2,563
- FCC カウント下限 = 2,441*0.95 = 2,318
タイミング考慮事項:
- クロック同期時間:5 ～ 6 LFCLK サイクル
- FCC トリガ時間:1 ～ 32 LFCLK サイクル (ユーザー構成可能)
レジスタ構成:
- FCC 設定:SYSCTL.GENCLKCFG.FCCTRIGSRC = 1;
- SYSCTL.GENCLKCFG.FCCLVLTRIG = 0;
- SYSCTL.GENCLKCFG.FCCTRIGCNT = 0;
- SYSCTL.GENCLKCFG.FCCSELCLK = 4;
- FCC を開始:SYSCTL.FCCCMD = 0x0E000001U
- FCC 完了ステータスのチェック:SYSCTL.CLKSTATUS.FCCDONE
- FCC カウントの読み取り:SYSCTL.FCC
タイムアウト保護:
FCC Done ステータス監視中にソフトウェア ベースのタイムアウトを実装し、無制限の待機を防止します。

```
fccTimeOutCounter = 0;
while (DL_SYSCTL_isFCCDone() == 0) {
delay_cycles(977); /* 1x LFCLK cycle = 32MHz/32.768kHz */
fccTimeOutCounter++;
if(fccTimeOutCounter > 65) break;
/* Timeout set to approximately 2ms (ユーザーによるカスタマイズ可能) */
}
```

SYSPLL_ERR_01
(続き)

***SYSPLL* モジュール**

FCC チェック再起動:
FCC 測定値が予想される範囲を超えている場合は、SYSPLL をディスエーブルにしてから再度イネーブルにし (SYSPLLEN を 0 に設定してから 1 に設定)、FCC 検証を繰り返します。

回避方法 2:FCC 比率チェック FCC
トリガ ソースとして LFCLK を使用し、SYSPLL 出力と入力クロック周波数の両方をカウントします。FCC を実行し、出力クロックと入力クロックとの間の FCC チェック値の測定された比率が予想される比率であることを確認します。
計算例:
- SYSPLL = 80MHz ; HFCLK = 40MHz ; LFCLK = 32.768kHz
- Expect clock ratio = 80MHz/40MHz = 2.0000
- Measured FCC count (SYSPLL) = 80,000,000/32,768 = 2,441
- Measured FCC count (HFCLK) = 40,000,000/32,768 =1,220
- 測定クロック比 = 2,441/1,220 = 2.0008

FCC 比率許容誤差:
FCC 比率方法を使用すると、結合クロック精度誤差を除去し、FCC の不確定性 (2 カウント クロック サイクル) および計算の丸め誤差のみに依存します。これにより、FCC カウント チェック方式に比べて、±0.3% などの許容範囲をはるかに狭くすることができます。
タイミングに関する考慮事項:
- クロック同期時間:5 ～ 6 LFCLK サイクル
- FCC トリガ時間:1 ～ 32 LFCLK サイクル (ユーザー構成可能)
- 完全な FCC 比チェックあたりの合計時間:2*(同期時間 + トリガ時間)

FCC 比チェック フロー:
1.SYSPLL 出力クロック用に FCC を構成します (SYSPLL0 または SYSPLL2X)
2.FCC を開始し、FCC Done を待機 (タイムアウト保護)
3.FCC チェック カウントを読み戻します
4.SYSPLL 入力クロックの FCC を構成します (SYSOSC または HFCLK)
5.FCC を開始し、FCC Done を待機 (タイムアウト保護)
6.FCC チェック カウントを読み戻します
7.FCC チェック比率を計算し、予測比率範囲と比較します
8.FCC 比が予想される範囲を超えている場合は、SYSPLL をディスエーブルにしてから再度イネーブルにし (SYSPLLEN を 0 に設定してから 1 に設定)、FCC 比検証を繰り返します。

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 イネーブル ビットへの HW 更新には、単一の機能クロック サイクルがかかります。32.768kHz と 3 の分周器を使用すると、イネーブル 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

初期リピート カウンタの周期は、次のリピート モジュールより 1 回だけ少なくなる

カテゴリ

機能

機能

TIMER

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_ERR_01

UNICOMMI2CC モジュール

カテゴリ

機能

機能

I2C コントローラの BUSY ステータス ポーリングの問題

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するとき、BUSY ステータス フラグがアサートするのに約 2 ～ 3 I2C 機能クロック サイクルが必要です。

UNICOMMI2CC_ERR_01 (続き)

UNICOMMI2CC モジュール

BUSRTRUN/FRAME_START を設定した直後に BUSY ビットをポーリングする場合、アプリケーションが適切に設定する前にステータスをチェックすることがあります。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より顕著になります。

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。遅延の推奨値は次のとおりです。ソフトウェア遅延 = 3 x I2C 機能クロック = 3 x clock_divider x (CPU_CLK / 選択したクロック ソース周波数)。たとえば、clock_divider が 2、クロック ソースが 4MHz(MFCLK)、CPU_CLK が 80MHz の場合:ソフトウェア遅延 = 3 x 2 x (80MHz / 4MHz) = 120 CPU サイクル

UNICOMMUART_ERR_06

UNICOMMUART モジュール

カテゴリ

機能

機能

ストップ ビット処理が原因で、RTOUT/LTOUT 計算に問題が発生

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 ビットから IDLE に遷移します。これにより、受信タイムアウト (RTOUT) およびライン タイムアウト (LTOUT) カウンタは、終了ではなく、ストップ ビットの中央でカウントを開始します。RTOUT/LTOUT は、ボー周期の半分前にトリガされることになります。これは、UART 機能クロック周波数が高いボーレートの場合に特に顕著です。

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_ERR_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

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_ERR_09

UNICOMMUART モジュール

カテゴリ

機能

機能

ISO-7816 スマートカード モードのボーレート制限

説明

ISO-7816 スマートカード モードで 9600 ボーレートを実現するには、以下の制約があるので、57MHz を超える UARTCLK 周波数が必要です。1.ISO-7816 規格では、ビット 2 あたり 372 クロック サイクルが必要です。MSPM0 ISO-7816 モードでは、オーバーサンプリング レート (OVS) は UART ペリフェラルの最大 UARTCLK 計算で 16x に固定されています。必要な UARTCLK = 9600 * 372 * 16 = 57.139MHz

回避方法

入力周波数が低い UART は、スマートカード モードをサポートできません。

UNICOMMUART_ERR_10

UNICOMMUART モジュール

カテゴリ

機能

機能

LIN レジスタの CLKDIV 制限

説明

CLKDIV 値が 0 以外の場合、LINC0/1 に書き込んでも効果はありません。

回避方法

LIN レジスタを適切に構成するには:1.まず、CLKDIV を '0' に設定します。2.目的の値を使用して、LINC0/1 レジスタ構成を更新します 3.CLKDIV を目的の動作値に復元します

VREF_ERR_05

VREF モジュール

カテゴリ

機能

機能

COMP がサンプリング (超低消費電力) モードに構成されていると、VREF0 の VREF READY0 のステータスが Low になります

説明

コンパレータがサンプリング (超低消費電力) モードに構成されている場合、VREF READY0 のステータスは VREF0 の Low に維持されます。REFMODE ビットが 1 に設定されているとき、コンパレータはサンプリング モードに構成されます。

回避方法

コンパレータをサンプリング モードに構成した後は、VREF READY0 ステータスをポーリングしないでください。VREF0 の READY0 ステータスをポーリングする前にコンパレータを静的 (高速) モードで構成するか、コンパレータを有効にする前に READY0 ステータスをポーリングします。

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
2026 年 2 月	*	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むいかなる保証も、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 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。

最終更新日：2025 年 10 月