

Errata

MSPM0L111x マイクロコントローラ



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.....	1
2 プログラム済みのソフトウェア アドバイザリ.....	3
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.....	3
6 アドバイザリの説明.....	4
7 改訂履歴.....	20

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
ADC_ERR_05	✓
I2C_ERR_05	✓
I2C_ERR_06	✓
I2C_ERR_07	✓
I2C_ERR_08	✓
I2C_ERR_09	✓
I2C_ERR_10	✓
I2C_ERR_11	✓
I2C_ERR_13	✓
PMCU_ERR_11	✓
PMCU_ERR_13	✓
PMCU_ERR_10	✓
SPI_ERR_04	✓
SPI_ERR_05	✓
SPI_ERR_06	✓
SPI_ERR_07	✓
SYSOSC_ERR_01	✓
SYSOSC_ERR_02	✓

エラッタ番号	リビジョン A
TIMER_ERR_04	✓
TIMER_ERR_05	✓
TIMER_ERR_06	✓
TIMER_ERR_07	✓
UART_ERR_01	✓
UART_ERR_02	✓
UART_ERR_03	✓
UART_ERR_04	✓
UART_ERR_05	✓
UART_ERR_06	✓
UART_ERR_07	✓
UART_ERR_08	✓
UART_ERR_10	✓
UART_ERR_11	✓
FLASH_ERR_02	✓
FLASH_ERR_03	✓
FLASH_ERR_05	✓
FLASH_ERR_08	✓
SYSCTL_ERR_01	✓
SYSCTL_ERR_02	✓
SYSCTL_ERR_04	✓
CPU_ERR_02	✓
CPU_ERR_03	✓
AES_ERR_01	✓
KEYSTORE_ERR_01	✓
RST_ERR_01	✓

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。

エラー番号	リビジョン A
GPIO_ERR_03	✓

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。X マーク付きの開発サンプルは、-40°C ~ 85°C の温度範囲内で動作させる必要があることに注意してください。

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。

6 アドバイザリの説明

PMCU_ERR_10 *PMCU* モジュール

カテゴリ

機能

機能

特定の動作条件下では、VBOOST により大きな遅延が発生する可能性があります

説明

アナログマルチプレクサの VBOOST は、VDD < 1.8V で大きな遅延が発生しました。このため、HFXT、COMP、SYSOSC (FCL-EXTERNAL R)、OPA、GPAMP などのその他のモジュールのセトリングタイムが遅延します。

回避方法

VDD を 1.8V 以上に維持し、GENCLKCFG[23:22] = 0x2 を設定して、VBOOST を ONALWAYS モードで使用します。

PMCU_ERR_11 *PMCU* モジュール

カテゴリ

機能

機能

NRST < 1 のパルスにより、シャットダウンモードで誤った rstcause が発生

概要

次の条件下で rstcause の値が正しくありません。予想される rstcause は 0x05 です。
 (i) デバイスをシャットダウンモードに構成済み
 (ii) WFI()を呼び出し
 (iii) デバイスをシャットダウンモードから復帰させるために NRST < 1 秒のパルスを与えます

回避方法

回避方法はありません。

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。エラーは、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。

ADC_ERR_05	ADC モジュール
カテゴリ	機能
機能	IP (周辺モジュール) が有効化される前にハードウェアイベントが生成された場合、その ADC トリガはキューに保持されたままになります
説明	ADC を HW イベントトリガモードに構成されていて、ADC が有効になる前にトリガが生成されると、ADC トリガはキュー内にとどまります。ADC が有効になると、サンプリングおよび変換がトリガされます。
回避方法	ADC をハードウェアトリガ モードで設定した後、外部トリガを与える前に、まず ADC を有効にします。
GPIO_ERR_03	GPIO および <i>DEBUGSS</i> モジュール
カテゴリ	デバッグのみ
機能	デバッガで GPIO EVENT0 IIDX を読み取ると、割り込みがクリアされます。
説明	GPIO の EVENT0 の IIDX をデバッガで読み取ると、CPU による読み取りと見なされ、割り込みがクリアされます。
回避方法	GPIO 割り込みとイベントが予想される場合は、デバッガを介して IIDX レジスタを使用して読み取らないでください。代わりに、RIS レジスタのソフトウェア読み取りを利用します。
I2C_ERR_04	I2C モジュール
カテゴリ	機能
機能	SCL が Low で SDA が High の状態では、ターゲット I2C はストレッチを解除できません。
概要	<p>1: SCL ラインを接地して解放し、デバイスは無制限に SCL を Low にプルします。</p> <p>2: ポストクロックストレッチ、タイムアウト、解放。ライン上に別のクロック Low がある場合、本デバイスは無期限に SCL を Low にプルします。</p>
回避方法	<p>I2C ターゲットアプリケーションで、非同期高速クロック要求を使用した低電力モードでのデータ受信が不要な場合は、SWUEN をデフォルトで無効にすることを推奨します (リセット時や電源サイクル時を含む)。この場合、バグの説明 1 と 2 は発生しません。</p> <p>I2C ターゲットアプリケーションで、非同期高速クロック要求を使用した低電力モードでのデータ受信が必要な場合は、低電力モードへ移行する直前に SWUEN を有効にし、復帰後に SWUEN をクリアします。このシナリオでも、I2C ターゲットが低消費電力のときにバグ説明 1 および 2 が発生するおそれがあります。バス上の他のデバイスによって連続的なクロックストレッチングまたはタイムアウトが発生すると、SCL ラインが無期限にストレッチされます。この状況から回復</p>

I2C_ERR_04 (続き) I2C モジュール

するには、I2C ターゲットデバイスで Low タイムアウト割り込みを有効にし、低タイムアウト ISR 内で I2C モジュールをリセットして再初期化します。

I2C_ERR_05 I2C モジュール

カテゴリ

機能

機能

進行中のトランザクション中に ACTIVE ビットをトグルすると、I2C SDA が 0 に固定化されるおそれがあります

説明

進行中の転送中に ACTIVE ビットがトグルされると、ステート マシンはリセットされます。ただし、マスターによって駆動される SDA と SCL 出力はリセットされません。SDA が 0 の状態でマスターが IDLE 状態に遷移すると、マスターは IDLE 状態から先へ進めず、SDA の値も更新できなくなります。スレーブの BUSBUSY がセットされ (ACTIVE ビットのトグルによってライン上で開始が検出されます)、BUSBUSY はクリアされません。これは、マスターが STOP を駆動してクリアできないためです。

回避方法

進行中のトランザクション中は、ACTIVE ビットをトグルしないでください。

I2C_ERR_06 I2C モジュール

カテゴリ

機能

機能

SMBus の High タイムアウト機能は、I2C クロックが 24 kHz 未満になると動作しません

説明

SMBus の High タイムアウト機能は、I2C クロックレートが 24 kHz 未満 (20 kHz、10 kHz など) では正常に動作しません。SMBus 仕様から、アクティブトランザクション中の SCL High 時間の上限は 50 μ s です。開始 MMR ビットの書き込みから SCL Low までに要する合計時間は 60 μ s で、50 μ s 以上です。タイムアウト イベントがトリガされ、転送開始時にトランザクションを完了することなく I2C マスターが IDLE に移行できるようにします。以下は詳細な説明です。SCL が 20 kHz に構成されている場合、SCL の Low 期間と High 期間はそれぞれ 30 μ s および 20 μ s です。まず、High タイムアウトカウンタでデクリメントが開始し、同時に MMR ビットの書き込みが開始します。その後、START MMR ビットの書き込みから SDA が Low (スタート条件) になるまでに、1 SCL Low 期間 (30 μ s) かかります。次に、SDA が Low (スタート条件) になってから SCL が Low になる (データ転送が開始) までにさらに別の SCL Low 期間 (30 μ s) がかかり、この時点で High タイムアウトカウンタが停止します。合計で、カウンタの開始から終了まで 60 μ s かかります。ただし、高タイムアウトカウンタには上限 (50 μ s) により、I2C トランザクションは問題なく正常に動作しますが、タイムアウトイベントがトリガされます。

回避方法

I2C クロックが 24KHz 未満の場合は、SMBus High タイムアウト機能を使用しないでください。

I2C_ERR_07	I2C モジュール
カテゴリ	機能
機能	コントローラの制御レジスタへの連続書き込みを行うと、I2C 通信が開始されない可能性があります。
説明	連続 CTR レジスタへの書き込みでは、次の CTR .START によって正しく開始条件が発生しません。
回避方法	CTR.START を含むすべての CTR ビットを 1 回の書き込みで書き込むか、CTR 書き込みと CTR.START 書き込みの間に 1 クロック サイクル待機します。
I2C_ERR_08	I2C モジュール
カテゴリ	機能
機能	RXDONE 割り込みの直後に FIFO を読み出すと、誤ったデータが取得されます
概要	RXDONE 割り込みが発生したとき、FIFO は最新のデータに対して常に更新されるとは限りません。
回避方法	最新のデータが FIFO に確実に反映されるように、2 つの I2C クロックサイクル分待機してください。I2C CLK は、I2C レジスタの CLKSEL レジスタに基づいています。
I2C_ERR_09	I2C モジュール
カテゴリ	機能
機能	I2C を低速で動作させている場合、割り込みサービスルーチン (ISR) 内での読み取り時に、開始アドレス一致ステータスがタイミング的に更新されていない可能性があります。
説明	I2C 速度が 100kHz 未満で動作している場合、ADDRMATCH ビット (TSR レジスタのアドレス一致) が割り込みによる読み取りに間に合うように設定されない可能性があります。
回避方法	I2C で 100kHz 未満で実行している場合は、ADDRMATCH ビットを読み取る前に少なくとも 1 つの I2C CLK サイクルを待機します。
I2C_ERR_10	I2C モジュール
カテゴリ	機能
機能	低消費電力に移行しないよう、I2C ビジーステータスは有効になっています

I2C_ERR_10 (続き) I2C モジュール

概要

I2C ターゲットモードでは、STOP ビットがない場合、トランザクションの後、I2C ビジーステータスは High のままです。

回避方法

STOP ビットを送信するように I2C コントローラをプログラムします。最後のバイトに対して NACK を送信しないでください。すべての I2C 転送は STOP 条件で終了し、適切な BUSY ステータスと非同期クロック要求の動作を維持してください(低消費電力モードへの再移行に備えるため)。

UNICOMMI2CC_ERR_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

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 = $3 \times 8 \times (32\text{MHz} / 4\text{MHz}) = 192 \text{ CPU サイクル}$

I2C_ERR_13

I2C モジュール

カテゴリ

機能

機能

I2C BUSY ビットをポーリングしても、コントローラの転送が完了したことが保証されない場合があります。

説明

I2C コントローラ転送を開始するために CCTR.BURSTRUN ビットを設定した後、BUSY ステータスがアサートされるまでに約 3 回の I2C 機能クロックサイクルがかかります。CCTR.BURSTRUN を設定した後すぐに転送完了を待つために BUSY ビットのポーリングを使用すると、BUSY ステータスが設定される前にチェックされる可能性があります。この問題は、CLKDIV 値が高い場合(I2C 機能クロックが遅くなる)、またはコンパイラの最適化レベルが高い場合に発生する可能性が高くなります。

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てください。ソフトウェア遅延 = $3 \times \text{CPU CLK} / \text{I2C 機能クロック} = 3 \times \text{CPU CLK} / (\text{CLKSEL} / \text{CLKDIV})$ 。例えば、クロック分周器(CLKDIV)が 8、クロックソース(MFCLK)が 4 MHz、CPU CLK が 32 MHz の場合、ソフトウェア遅延 = $3 \times 32 \text{ MHz} / (4 \text{ MHz} / 8) = 192 \text{ CPU サイクル}$

SPI_ERR_04	SPI モジュール
カテゴリ	機能
機能	SPI ペリフェラルが受信モードのみの場合、各フレーム受信後の IDLE/BUSY ステータスグル。
説明	SPI ペリフェラルが受信モードのみの場合、SPI がデータを連続的に受信している間に、各フレーム受信の後で、IDLE 割り込みおよび BUSY ステータスがトグルされます (SPI_PHASE = 1)。ここでは、ペリフェラル (スレーブ) の TXFIFO にロードされるデータはなく、TXFIFO は空です。
回避方法	SPI ペリフェラルのみの受信モードを使用しないでください。SPI をペリフェラル (スレーブ) 同時送受信モードに設定します。
SPI_ERR_05	SPI モジュール
カテゴリ	機能
機能	SPI ペリフェラルの受信タイムアウト割り込みは、RXFIFO のデータの有無にかかわらず発生します
説明	SPI タイムアウト割り込みを使用すると、ペリフェラルが SPI クロックの受信を停止した時点から RXTIMEOUT カウンタがデクリメントを開始し、RXFIFO にデータが存在するかどうかに関係なく RXTIMEOUT 割り込みをセットしますが、これは TRM の説明と一致しません。SPI ペリフェラルの受信タイムアウト (RTOUT) 割り込みは、受信 FIFO が空でなく、CTL1.RXTIMEOUT で指定された時間内にそれ以上のデータが受信されない場合にアサートされます。
回避方法	受信 FIFO が空の間は負荷 RXTIMEROUT カウンタ値を繰り返し、受信 FIFO がデータを取得した場合にのみタイムアウトカウントを開始します。
SPI_ERR_06	SPI モジュール
カテゴリ	機能
機能	デバッグ HALT がアサートされている場合、IDLE/BUSY ステータスは SPI IP の正しい状態を反映しません
概要	IDLE/BUSY は HALT とは無関係で、RXFIFO/TXFIFO の書き込み/読み取りストロブのみをゲーティングします。つまり、コントローラがデータ送信中であっても、そのデータが FIFO にラッチされていない状態で BUSY ステータスが設定されてしまいます。POCI 回線は、停止中に以前に送信されたデータを回線上で送信します
回避方法	SPI IP が停止しているときは、IDLE/BUSY ステータスを使用しないでください。

SPI_ERR_07

SPI モジュール

カテゴリ

機能

機能

SPI ペリフェラルで TXFIFO への読み取り / 書き込みが同時に発生した場合、SPI アンダーフロー イベントは生成しない場合があります。

説明

SPI.CTL0.SPH = 0 であり、本デバイスが SPI ペリフェラルとして構成されている場合。

SPI コントローラからの読み取り要求がある間に TXFIFO への書き込みが発生した場合、読み取り / 書き込み要求が同時に発生するため、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。

回避方法

SPI コントローラによるデバイスのアドレス指定中、TXFIFO が確実に空でないようにします。これは、同じ TXFIFO アドレスへの書き込みと読み取りを避けるために、データを事前ロードすることで実現できます。あるいは、CRC のようなデータチェック戦略を使用してパケットが確実に正しく送信されるようにし、CRC が一致しない場合にデータを再送信することもできます。

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

特定の条件では、MFCLK の起動に失敗します。

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/STANBY1)。
- 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クロック要求が受信されます。

回避方法

上記のシナリオを回避します。非同期高速クロック要求がイネーブルの間、システムが FCL および MFCLK とリストされている低消費電力モードを使用していないことを確認してください。

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

GPTIMER をワンショット モードで使用し、CLKDIV.RATIO が 0 でない場合、ゼロ イベント直前に TIMER を再有効化すると、再有効化が失われることがあります。

回避方法

タイマは、最初に再度イネーブルにする前に無効にできます。

TIMER_ERR_06	タイマ モジュール
カテゴリ	機能
機能	CLKEN 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 (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_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。
UART_ERR_01	UART モジュール
カテゴリ	機能

UART_ERR_01 (続 き)

UART モジュール

機能

STANDBY1 モードへの遷移時に、UART のスタート条件が検出されないことがあります

概要

デバイスが STANDBY1 モードのときに、UART 送信によって開始された非同期高速クロック要求を処理した後、デバイスは STANDBY1 モードに戻ります。STANDBY1 モードへの復帰中に別の UART 送信が開始されると、デバイスはそのデータを正しく検出および受信できません。

回避方法

UART のスタート条件が繰り返し発生することが想定される場合は、STANDBY0 モードまたはそれ以上の低消費電力モードを使用してください。

UART_ERR_02

UART モジュール

カテゴリ

機能

機能

TXE のみが有効な場合、UART 送信終了の割り込みは設定されません

概要

デバイスを送信のみに設定すると (CTL0.TXE = 1、CTL0.RXE = 0)、UART 送信終了 (EOT) 割り込みのトリガはかかりません。デバイスが送受信に設定されている場合 (CTL0.TXE = 1、CTL0.RXE = 1)、EOT は正常にトリガされます

回避方法

UART 送信終了割り込みを使用するときは、CTL0.TXE ビットおよび CTL0.RXE ビットの両方を設定します。ピンを UART 受信として割り当てる必要はないので注意してください。

UART_ERR_03

UART モジュール

カテゴリ

機能

機能

3x オーバーサンプリングと MFCLK または BUSCLK をクロック ソースとして使用して、UART RX 割り込みが誤って設定されます

説明

UART および 3x オーバーサンプリングに BUSCLK または MFCLK を使用する場合、RXINT が誤って設定されます。BUSCLK または MFCLK をソースとして UART モジュールを使用し、3x オーバーサンプリング モードの場合、RX 割り込みが誤って設定される可能性があります。これらの条件下では、TX データも破損する可能性があります。

回避方法

BUSCLK または MFCLK を使用する場合は、より高いオーバーサンプリング レートを使用してください。3x オーバーサンプリングが必要な場合は、LFCLK を利用します。

UART_ERR_04

UART モジュール

カテゴリ

機能

UART_ERR_04 (続き)

UART モジュール

機能

クロックが SYSOSC から LFOSC に遷移する際、高速クロック要求が無効になっていると、UART データが誤って受信される可能性があります

概要

シナリオ:

1. UART の機能クロックとして LFCLK が選択されます 2.3 倍オーバーサンプリングで構成された 9600 のボーレート 3. UART 高速クロック要求が無効になっている状態で、UART 受信転送中に ULPCCLK が SYSOSC から LFOSC に切り替わると、1 ビットが誤って読み取られることがあります

回避方法

LPM モードで UART を使用する場合は、UART 高速クロック要求を有効にしてください。

UART_ERR_05

UART モジュール

カテゴリ

機能

機能

UART モジュールのデバッグ停止機能の制限

概要

本来は既存のフレームを完了して停止することが期待されますが、すべての Tx FIFO 要素が送信されてから通信が停止します。

回避方法

デバッグ停止がアサートされた後は、データが TX FIFO に書き込まれないようにしてください。

UART_ERR_06

UART モジュール

カテゴリ

機能

機能

UART 9 ビットモードでの予期しない RTOUT/Busy/Async の動作

説明

UART 受信タイムアウト (RTOUT) は、マルチノード構成では正しく動作しません。この構成では、1 つの UART がコントローラとして動作し、他の UART ノードはペリフェラルとして機能し、各ペリフェラルは 9 ビット UART モードで異なるアドレスに設定されます。

最初の UART コントローラが UART ペリフェラル 1 と通信し、ペリフェラル 1 のアドレスを最初のバイトとして送信してからデータを送信することで、ペリフェラル 1 がアドレスの一致を確認してデータを受信しました。コントローラがペリフェラル 1 との通信を終了した後、バス上で異なるアドレスに構成された別の UART ペリフェラル (ペリフェラル 2) との通信を直ちに開始すると、ペリフェラル 1 は設定されたタイムアウト期間が経過しても RTOUT を設定しません。ペリフェラル 2 との通信中もペリフェラル 1 の RTOUT カウンタはリセットされ続け、RTOUT が設定されるのは、コントローラがペリフェラル 2 との通信を完了した後になります。

BUSY 要求と Async 要求で同様の動作が確認観察されました。コントローラがバス上の別のペリフェラルと通信中で、アドレスが一致しない場合でも、Busy および Async 要求が設定されます。

UART_ERR_06 (続き)

UART モジュール

回避方法

1 つのコントローラが複数のペリフェラルに接続されたマルチノード UART 通信では、RTOUT / BUSY / 非同期クロック要求の動作は使用しないでください。

UART_ERR_07

UART モジュール

カテゴリ

機能

機能

IDLE LINE モードにおいて、RTOUT カウンタが期待どおりにカウントされません

概要

UART のアイドルラインモードでは、ラインがアイドル状態で、FIFO に何らかの要素がある場合でも、RTOUT カウンタはスタックします。つまり、IDLE LINE モードでは RTOUT 割り込みは動作しません。
アドレスが一致しない場合、Rx ラインでトグルの発生を検出すると RTOUT カウンタがリロードされます。
マルチレスポンス構成の場合、コマンドと他のレスポンス間で通信が行われていると、RTOUT イベントの取得に不定の遅延が発生するおそれがあります。

回避方法

UART モジュールを IDLLINE モード/マルチノード UART アプリケーションのいずれかで使用する場合、RTOUT 機能を有効にしないでください。

UART_ERR_08

UART モジュール

カテゴリ

機能

機能

STAT BUSY は、UART モジュールの正しいステータスを表していません

概要

UART モジュールが無効で TXFIFO でデータが利用可能である場合でも、STAT BUSY は High のままです。

回避方法

TXFIFO ステータスと CTL0.ENABLE レジスタビットをポーリングして、ビジーステータスを識別します。

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 モードは、57MHz UARTCLK 未満の 9600 ボーレートをサポートできません

UNICOMMUART_E

RR_09 (続き)

UNICOMMUART モジュール**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 UARTCLK = $9600 * 372 * 16 = 57.139\text{MHz}$

回避方法

なし

UART_ERR_11**UART** モジュール**カテゴリ**

機能

機能

UART 受信タイムアウトが、STOP ビット転送中に、予期したタイミングよりも早くカウントを開始する

説明

STOP ビット転送時に、受信タイムアウトが STOP ビット転送の途中でカウントを開始する場合があります。その結果、RXTSEL の設定値が小さすぎる場合、意図しない RTOUT 割り込みが発生する可能性があります。たとえば、ボーレートが 1Mbps で、RXTSEL が 1 に設定されている場合、想定される RTOUT は STOP ビット転送の 1 μ s 後に発生するはずですが、実際には RTOUT 割り込みが 0.5 μ s で設定されます。

回避方法

UART.IFLS.RXTSEL レジスタは、受信タイムアウト (RTOUT) 割り込みが発生するまでのビット時間を選択します。早期割り込みを防止するには、RXTSEL の値を 1 より大きくする必要があります。受信タイムアウト時間は次のように計算できます。受信タイムアウト = $(\text{RXTSEL} - 0.5) / \text{ボーレート}$

FLASH_ERR_02**FLASH** モジュール**カテゴリ**

機能

機能

NONMAIN でのデバッグの無効は、パスワードを使用して再度有効にできる

説明

デバッグが NONMAIN 設定 (DEBUGACCESS = 0x5566) により無効化されている場合でも、プログラムされたパスワードを使用してデバイスにアクセスできる可能性があります (パスワードが明示的に設定されていない場合はデフォルト値が使用される)。

回避方法

回避方法 1:

DEBUGACCESS を Debug Enabled with Password オプション (DEBUGACCESS = 0xCCDD) に設定し、PWDDEBUGLOCK フィールドに一意のパスワードを入力します。より高度のセキュリティを確保するために、暗号化されたランダムなデバイス固有のパスワードを使用することをお勧めします。これにより、適切な 128 ビットのパスワードでデバッグアクセスが可能になりますが、一部のデバッグコマンドで、CFG-AP と SEC-AP にアクセスすることもできます。

FLASH_ERR_02

(続き)

FLASH モジュール

回避方法 2:

SWDP_MODE を無効にして、物理的な SW デバッグポートを完全に無効にします。これにより、デバイスへのデバッグアクセスや要求は完全に防止されますが、**Failure Analysis** やリターンフローに影響が出るおそれがあります。

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_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

FLASH_ERR_08

(続き)

FLASH モジュール

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

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_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

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クセス (SHUTDNSTORE) を発行します。これは SYSCTL.SOCLOCK.SHUTDNSTORE0; で実行できます。

メモリアクセスが完了すると、プリフェッチャーは無効になります。

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クセス

CPU_ERR_03

CPU モジュール

カテゴリ

機能

機能

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

説明

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

AES_ERR_01

AES モジュール

カテゴリ

機能

AES_ERR_01 (続き) AES モジュール

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ステディのステータスビットを確認します。

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。

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。

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。

Changes from NOVEMBER 30, 2024 to NOVEMBER 30, 2025 (from Revision * (November 2024) to Revision A (November 2025))

	Page
• PMCU_ERR_12 を削除.....	1
• VREF_ERR_03 を削除.....	1
• PMCU_ERR_11 の説明を更新しました.....	4
• PMCU_ERR_13 カテゴリを更新しました.....	4
• PMCU_ERR_13 モジュールを更新しました.....	4
• PMCU_ERR_13 機能を更新しました.....	4
• PMCU_ERR_13 の説明を更新しました.....	4
• PMCU_ERR_13 回避策を更新しました.....	4
• 説明および回避策が更新されました.....	5
• I2C_ERR_07 カテゴリを更新しました.....	7
• I2C_ERR_07 モジュールを更新しました.....	7
• I2C_ERR_07 機能を更新しました.....	7
• I2C_ERR_07 の説明を更新しました.....	7
• I2C_ERR_07 回避策を更新しました.....	7
• I2C_ERR_08 カテゴリを更新しました.....	7
• I2C_ERR_08 モジュールを更新しました.....	7
• I2C_ERR_08 機能を更新しました.....	7
• I2C_ERR_08 回避策を更新しました.....	7
• I2C_ERR_08 の説明を更新しました.....	7
• I2C_ERR_09 カテゴリを更新しました.....	7
• I2C_ERR_09 モジュールを更新しました.....	7
• I2C_ERR_09 機能を更新しました.....	7
• I2C_ERR_09 の説明を更新しました.....	7
• I2C_ERR_09 回避策を更新しました.....	7
• I2C_ERR_10 カテゴリを更新しました.....	7
• I2C_ERR_10 モジュールを更新しました.....	7
• I2C_ERR_10 機能を更新しました.....	7
• I2C_ERR_10 の説明を更新しました.....	7
• I2C_ERR_10 回避策を更新しました.....	7
• UNICOMMI2CC_ERR_1 カテゴリを更新しました.....	8
• UNICOMMI2CC_ERR_1 機能を更新しました.....	8
• UNICOMMI2CC_ERR_1 の説明を更新しました.....	8
• UNICOMMI2CC_ERR_1 回避策を更新しました.....	8
• UNICOMMI2CC_ERR_1 モジュールを更新しました.....	8
• I2C_ERR_13 カテゴリを更新しました.....	8
• I2C_ERR_13 モジュールを更新しました.....	8
• I2C_ERR_13 機能を更新しました.....	8
• I2C_ERR_13 回避策を更新しました.....	8
• I2C_ERR_13 の説明を更新しました.....	8
• SPI_ERR_06 カテゴリを更新しました.....	9
• SPI_ERR_06 モジュールを更新しました.....	9
• SPI_ERR_06 機能を更新しました.....	9
• SPI_ERR_06 回避策を更新しました.....	9
• SPI_ERR_06 の説明を更新しました.....	9
• SPI_ERR_07 カテゴリを更新しました.....	10
• SPI_ERR_07 モジュールを更新しました.....	10
• SPI_ERR_07 機能を更新しました.....	10
• SPI_ERR_07 の説明を更新しました.....	10
• SPI_ERR_07 回避策を更新しました.....	10

• UART_ERR_04 カテゴリを更新しました.....	13
• UART_ERR_04 モジュールを更新しました.....	13
• UART_ERR_04 機能を更新しました.....	13
• UART_ERR_04 の説明を更新しました.....	13
• UART_ERR_04 回避策を更新しました.....	13
• UART_ERR_05 カテゴリを更新しました.....	14
• UART_ERR_05 モジュールを更新しました.....	14
• UART_ERR_05 機能を更新しました.....	14
• UART_ERR_05 の説明を更新しました.....	14
• UART_ERR_05 回避策を更新しました.....	14
• UART_ERR_06 カテゴリを更新しました.....	14
• UART_ERR_06 モジュールを更新しました.....	14
• UART_ERR_06 機能を更新しました.....	14
• UART_ERR_06 回避策を更新しました.....	14
• UART_ERR_06 の説明を更新しました.....	14
• UART_ERR_07 カテゴリを更新しました.....	15
• UART_ERR_07 モジュールを更新しました.....	15
• UART_ERR_07 機能を更新しました.....	15
• UART_ERR_07 回避策を更新しました.....	15
• UART_ERR_07 の説明を更新しました.....	15
• UART_ERR_08 カテゴリを更新しました.....	15
• UART_ERR_08 モジュールを更新しました.....	15
• UART_ERR_08 機能を更新しました.....	15
• UART_ERR_08 の説明を更新しました.....	15
• UART_ERR_08 回避策を更新しました.....	15
• UNICOMMUART_EER_9 カテゴリを更新しました.....	15
• UNICOMMUART_EER_9 機能を更新しました.....	15
• UNICOMMUART_EER_9 の説明を更新しました.....	15
• UNICOMMUART_EER_9 回避策を更新しました.....	15
• UNICOMMUART_EER_9 モジュールを更新しました.....	15
• UART_ERR_11 カテゴリを更新しました.....	16
• UART_ERR_11 モジュールを更新しました.....	16
• UART_ERR_11 機能を更新しました.....	16
• UART_ERR_11 の説明を更新しました.....	16
• UART_ERR_11 回避策を更新しました.....	16
• FLASH_ERR_02 カテゴリを更新しました.....	16
• FLASH_ERR_02 モジュールを更新しました.....	16
• FLASH_ERR_02 機能を更新しました.....	16
• FLASH_ERR_02 の説明を更新しました.....	16
• FLASH_ERR_02 回避策を更新しました.....	16
• FLASH_ERR_03 カテゴリを更新しました.....	17
• FLASH_ERR_03 モジュールを更新しました.....	17
• FLASH_ERR_03 機能を更新しました.....	17
• FLASH_ERR_03 の説明を更新しました.....	17
• FLASH_ERR_03 回避策を更新しました.....	17
• FLASH_ERR_05 カテゴリを更新しました.....	17
• FLASH_ERR_05 モジュールを更新しました.....	17
• FLASH_ERR_05 機能を更新しました.....	17
• FLASH_ERR_05 の説明を更新しました.....	17
• FLASH_ERR_05 回避策を更新しました.....	17
• FLASH_ERR_08 カテゴリを更新しました.....	17

• FLASH_ERR_08 モジュールを更新しました.....	17
• FLASH_ERR_08 機能を更新しました.....	17
• FLASH_ERR_08 の説明を更新しました.....	17
• FLASH_ERR_08 回避策を更新しました.....	17
• SYSCTL_ERR_01 カテゴリを更新しました.....	18
• SYSCTL_ERR_01 モジュールを更新しました.....	18
• SYSCTL_ERR_01 機能を更新しました.....	18
• SYSCTL_ERR_01 の説明を更新しました.....	18
• SYSCTL_ERR_01 回避策を更新しました.....	18
• SYSCTL_ERR_02 カテゴリを更新しました.....	18
• SYSCTL_ERR_02 モジュールを更新しました.....	18
• SYSCTL_ERR_02 機能を更新しました.....	18
• SYSCTL_ERR_02 の説明を更新しました.....	18
• SYSCTL_ERR_02 回避策を更新しました.....	18
• SYSCTL_ERR_04 カテゴリを更新しました.....	18
• SYSCTL_ERR_04 回避策を更新しました.....	18
• SYSCTL_ERR_04 モジュールを更新しました.....	18
• SYSCTL_ERR_04 機能を更新しました.....	18
• SYSCTL_ERR_04 の説明を更新しました.....	18
• CPU_ERR_02 カテゴリを更新しました.....	19
• CPU_ERR_02 モジュールを更新しました.....	19
• CPU_ERR_02 の説明を更新しました.....	19
• CPU_ERR_02 機能を更新しました.....	19
• CPU_ERR_02 回避策を更新しました.....	19
• CPU_ERR_03 カテゴリを更新しました.....	19
• CPU_ERR_03 モジュールを更新しました.....	19
• CPU_ERR_03 機能を更新しました.....	19
• CPU_ERR_03 の説明を更新しました.....	19
• CPU_ERR_03 回避策を更新しました.....	19
• AES_ERR_01 カテゴリを更新しました.....	19
• AES_ERR_01 モジュールを更新しました.....	19
• AES_ERR_01 機能を更新しました.....	19
• AES_ERR_01 の説明を更新しました.....	19
• AES_ERR_01 回避策を更新しました.....	19
• KEYSTORE_ERR_01 カテゴリを更新しました.....	20
• KEYSTORE_ERR_01 モジュールを更新しました.....	20
• KEYSTORE_ERR_01 機能を更新しました.....	20
• KEYSTORE_ERR_01 の説明を更新しました.....	20
• KEYSTORE_ERR_01 回避策を更新しました.....	20
• RST_ERR_01 カテゴリを更新しました.....	20
• RST_ERR_01 モジュールを更新しました.....	20
• RST_ERR_01 回避策を更新しました.....	20
• RST_ERR_01 機能を更新しました.....	20
• RST_ERR_01 の説明を更新しました.....	20

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、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 © 2025, Texas Instruments Incorporated

最終更新日：2025 年 10 月