参考資料

Application Note

MSPM0 マイコンを使用した USB 設計

概要

このガイドでは、MSPM0 マイコン (MCU) を使用して USB ベースのデバイスを設計する場合の概要を紹介します。本書では、TI の MSPM0 から入手できるハードウェアおよびソフトウェア リソースを取り上げ、MSPM0 デバイスの内蔵 USB 機能について説明し、開発を簡素化するツールの概要を示します。

このガイドの内容:

- USB システムの概要
- MSPM0 USB ハードウェア モジュールと、利用可能なマイコン ファミリ
- USB ハードウェア設計における推奨事項
- MSPM0 USB ソフトウェア開発者用キットおよびサポート システムの概要
- MSPM0 マイコンをベースとする USB デバイス向けハードウェア リファレンス デザイン

English Document: SDAA280

1 USB は複雑なシステムをシンプルに見せる

1.1 なぜ USB はこれほど成功したのか？

USB は、現在、最も広く使用されている業界標準の一つです。使いやすさと合理化されたエクスペリエンスにより、多くのユーザーにとって欠かせないツールとなっています。さらに、以下のようないくつかの利点もあります。

- 低コスト化：低コスト化により、USB はユーザーにとって魅力的なものとなり、大量普及を促しました
- 広範な互換性：USB は広く普及しており、多くの製品で標準的な機能になっています。
- 用途の拡張：基本的な機能以外に、USB は以下のような革新的な用途も可能にしています。
  - 効率的なエネルギーの供給：小型のガジェットやデバイスを充電するための電力を供給します。
  - ペリフェラルの接続：キーボード、マウスなどのさまざまなペリフェラルの接続を可能にします。
  - データ ストレージ：フラッシュ ドライバから外付けハード ドライブまで、さまざまなストレージ オプションを提供します。
- 信頼性：USB デバイス間でエラーや障害のない安定した接続を確立します。

エンド ユーザーにとってはシンプルに見える USB ですが、内部設計は実際には複雑です。この複雑さは、一般的な動作を自動化し、ホット プラグに対応できる高速で信頼性の高いデータ バスを提供する必要があるために生じます。そのためには、多層のプロトコルが必要です。

1.2 なぜ USB はシンプルに見えるのか？

USB はユーザーから複雑さを隠していますが、開発者はその内部構造を見ることがよくあります。UART、SPI、I2C などの他のプロトコルと比較すると、USB はデータ転送にはるかに多くの処理を必要とします。USB は、レジスタへの書き込みよりも、データ送信に多くの労力を要します。

オンチップ USB モジュールはこの複雑さの低減に役立ちますが、複雑さを解消することはできません。USB スタックの大部分では、複雑な USB 通信を管理するためのソフトウェアが依然として必要です。適切に設計されたソフトウェアは、アプリケーション開発者をこれらの複雑さの多くから守ることができますが、それでも以下のような重要な課題に直面します。

- デバイスの接続と接続解除イベントの処理
- ビジーや信頼性の低いバスなど難易度の高い条件下でも、安定した通信と信頼性の高いデータ転送を維持して、データ損失を防止すること。
- 複数のホスト オペレーティング システムを管理およびサポートするための戦略の開発。

これらの検討事項は、開発者が USB ベースのアプリケーションが、信頼性が高く、効率的で、ユーザーにやさしいことを検証するために不可欠です。

2 MSPM0 USB シリコン

MSPM0 マイコン ファミリには、USB モジュールが内蔵されています。このモジュールは、Tiny USB と呼ばれる開発パッケージを含む最新の MSPM0-SDK と互換性があります。

ビットバンギングによって USB 通信を実装することは可能ですが、このアプローチはフルスピード動作には適しておらず、さらにプロセッサの処理能力の多くを消費してしまう可能性があります。そのため、大半の USB アプリケーションは、USB 通信を管理するためのより効率的かつ信頼性の高い手段を提供するオンチップ USB モジュールを利用しています。

2.1 MSPM0 デバイスの文書化方法

MSPM0 デバイスのドキュメントは、MSPM0 デバイス ファミリ製品の次の 3 つの場所に分割されています。

- テクニカル リファレンス マニュアル：MSPM0 マイコン ファミリに関するすべてのアーキテクチャ情報が含まれています。すべての USB 搭載デバイスについては、MSPM0 G シリーズ 80MHz マイコン テクニカル リファレンス マニュアルを参照してください。
- データシート：このデータシートには、このデバイス ファミリの製品に固有のすべてのパラメータと詳細が掲載されており、機能や性能を詳しく理解することができます。
- SDK ユーザー ガイド：SDK ユーザー ガイドには、新しいプロジェクト用のソフトウェアを作成するための出発点として使用できる、さまざまなドライバとサンプル プログラムの概要が記載されています。

これらの資料を組み合わせると、MSPM0 デバイス ファミリについて包括的に理解することができます。

2.2 MSPM0 USB モジュール

MSPM0 USB モジュールは、以下の機能をサポートしています。

- ホスト モードとデバイス モードでの USB 2.0 フルスピード (12Mbps) 動作、およびホスト モードでの低速 (1.5Mbps) 動作
- USB-IF 認証規格に準拠
- 4 つの転送タイプ：制御、割り込み、バルク、アイソクロナス
- 16 個のエンドポイント
  - コントロール転送専用の IN エンドポイントおよび OUT エンドポイント各 1 個
  - 転送タイプが設定可能な IN エンドポイントおよび OUT エンドポイント各 7 個
- 2kB の専用エンドポイント メモリ
- DMA サポート用の専用ハードウェア トリガ
- ブートストラップ ローダ (BSL) を使用した USB ベースのデバイス ファームウェア更新 (DFU) をサポート
- USBFS モジュールにはフルスピード PHY を内蔵
- シグナリングの中断および再開機能をサポート

3 MSPM0 USB ハードウェア デザイン

3.1 ブロック図

USB ブロック図を、図 3-1 に示します。

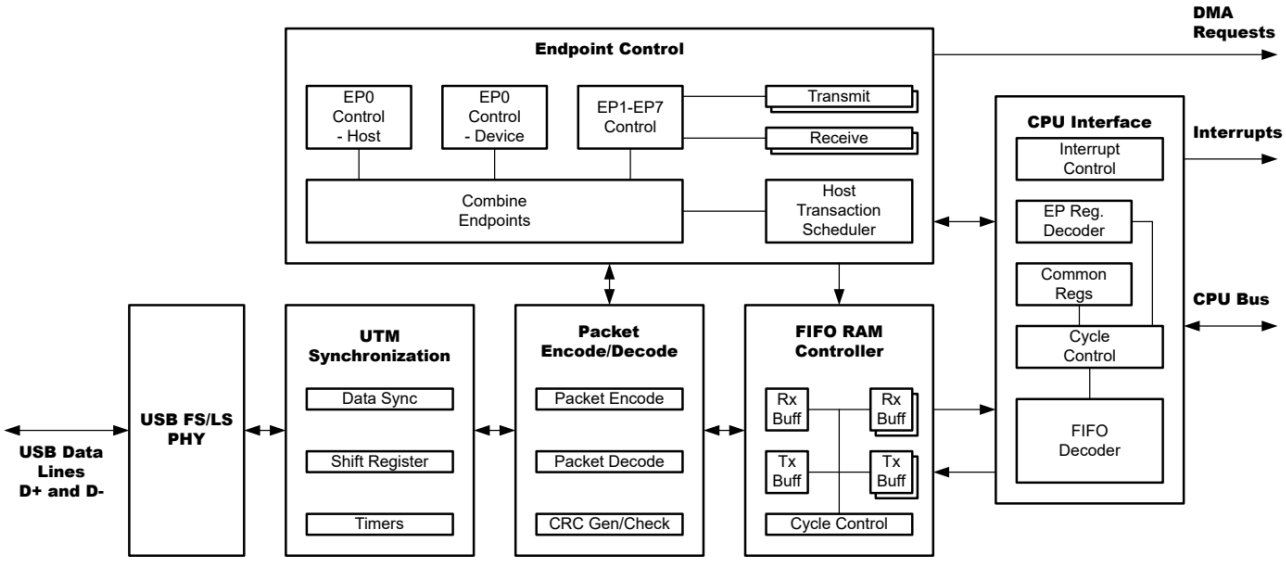

図 3-1. MSPM0 の USB ペリフェラルのブロック図

3.2 USB の動作モード

USB IP は、デバイス モードとホスト モードの両方を静的構成でサポートしています。

USB-C タイプのコネクタの場合、PCB システム上の CC1/CC2 ライン上のプルアップ / プルダウン抵抗でデバイスとホスト モードを示す必要があるため、モードの動的構成に対して USB IP が制限されます。したがって、USB IP は、デバイス モードとホスト モードとの間の On-the-Go (OTG) スイッチングをサポートしていません。

3.2.1 USB デバイス モード: バス パワー

バスパワー方式の USB アプリケーションでは、デバイスは USB コネクタ経由で USB ホストから電力を受け取ります。この電力供給方法は、USB デバイス アプリケーションに限定されており、USB ホストはバスに電力を供給する必要があるため、接続されている状態でのみ動作します。バスパワー型 USB アプリケーションの代表的な例として、USB マウスまたはキーボードがあります。これらのデバイスは、デバイスがコンピュータに接続され、デバイスが USB ホストから電源を供給されている場合にのみ動作します。

下の図は、バスパワー アプリケーションのシステム コンポーネントを示しています。このシステムでは、VBUS の電圧が 5V であるため、追加のオンボード コンポーネントに電力を供給するために外部 3.3V LDO が必要です。このモードでは、SoC に電源が供給されているときのみ動作するため、USB 接続検出回路は不要です。

3.2.2 USB デバイス モード: セルフパワー

USB デバイスのもう一つの電源方式は、「自己給電」です。これは、デバイス上に USB 電源以外に外部電源が存在し、必要に応じてチップと接続に電力を供給することを意味します。

このようなモードの代表的な例として、オーディオ デバイスまたはストレージ デバイスがあり、これらのデバイスでは接続が解除されていても電力が必要です。これらの設計では、バスパワー モードと比べて追加のパワーマネージメントが必要で、パワー ツリーの複雑さに応じて、何らかの形のメイン電源レギュレータとシーケンシングが使用されます。

多くの場合、セルフパワー デバイスは「ハイブリッド」と見なすこともできます。つまり、接続時にバス パワーを使用するか、接続時にバッテリを充電します。このため、多くの場合、接続を検出して、バスから供給されるパワー パスへの切り替えを

行うために、VBUS 監視回路が必要になります。これは、MSPM0G5187 の ADC ピンに抵抗デバイダを接続し、パワー パスのスイッチを切り替えるなどのシンプルなもので実現できます。

3.2.3 USB ホスト モードの電源に関する検討事項

MSP430 とは異なり、MSPM0Gx は USB 設計においてホストとして動作できます。複雑さの大半は、これらのアプリケーションのソフトウェア設計に由来しますが、ホストを設計する際には、追加のハードウェア要件もあります。

ホスト デバイスを設計する際、電力は主なニーズの一つです。これは、ホストは、接続されたデバイスに「5V VBUS 電力を供給する必要があり」、USB 2.0 フル スピードに必要な最大 500mA も供給する必要があるためです。多くの場合、これはデバイスを別のスイッチング電源などの専用電源とペアリングすることを意味します。USB デバイスは多くの場合、バスパワーで動作するため、一貫性のある動作を実現するには、ホストが任意のデバイスの電力要件をサポートできることを保証することが不可欠です。

接続されるデバイスが必ずしも正しく設計されているとは限らないため、保護も考慮する必要があります。誤ったデバイスが接続された場合、過電流や過電圧の状況が発生する可能性があります。電力線に保護回路と電流センス抵抗を内蔵することで、デバイスがこのような過電力状況にさらされていないことを確認できます。

3.2.4 ESD に関する考慮事項

ホストは、データ ラインおよび VBUS の ESD 保護において、より高い堅牢性を必要とします。これは、複数の未知のデバイスや信頼性が不確かなデバイスが下流側で ESD イベントを引き起こす可能性があるためです。通常、より高い ESD レーティングが推奨されており、接触放電で約 8kV、空気放電で約 15kV が目安とされています。

3.2.5 レイアウトに関する考慮事項

USB デバイスと同様に、2 つのラインが同期されていることを確認するために、データ ラインのトレース長を一致させることが重要です。

USB データ トレースはクリーンな状態に保ち、ビアを (使用する必要がある場合はペアで) 最小限に抑えて、高速信号から離して配線するように注意する必要があります。DP/DM 信号には、参照用に信号層の上または下の層にクリーンで完全なグランド プレーンを配置する必要があります。これらの信号の差動抵抗は、できるだけ 90Ω に近い値にする必要があります。ほとんどの PCBCAD ソフトウェアを使用して、この値を計算できます。

3.3 USB クロックの実装

USB には厳格なタイミング要件がありますが、これらの要件は、バスのタイミングと同期におけるそれぞれの役割があるため、USB ホストと USB デバイスとの間では大きく異なります。基本的に、ホストはバス上のタイミング制御権を持ち、デバイスはホストのタイミングに同期するようになります。

USB ホストは、フルスピードのデバイスに対して 1kHz の正確なレートで Start-of-Frame (SOF) パケットを生成します。SOF パケットは、USB バスのグローバル タイム ベースを定義するため、ホストのクロックは、バスからの外部リファレンスに頼らずに、個別に USB の精度要件を満たす必要があります。ホストのタイミング要件が厳しいため、高精度の外部発振器が必要です。この安定したクロックは、MSPM0 の FLL または PLL へのリファレンス入力として使用され、USB モジュールが必要とするより高い周波数のクロックを生成します。周波数誤差と温度ドリフトは、長期的な SOF タイミング精度に対して許容される値を超えているため、内部 LFOSC では不十分です。

一方、USB デバイスは独自の SOF パケットを生成しないため、タイミング マスタではありません。そのため、デバイスはホストが生成した SOF パケットからタイミングを取得できます。MSPM0 は、クロック ソースとして使用される USBFLL の入力基準として SOF パケットを利用します。デバイスは通常、ホストがタイミングを決定するため、非常に正確な外部発振器を必要としません。これにより、精度が低い内部クロックを使用できます。MSPM0 の USB ペリフェラルには、60MHz で動作するペリフェラル USBCLK に固有のクロックが含まれています。USBCLK は高精度のリファレンス クロックを必要とし、SYSPLL ブロックまたは USB 専用の USBFLL から供給できます。これは PLL ではなく FLL プロトコルを使用するため、より低い消費電力で動作します。クロック ツリーの詳細については、テクニカル リファレンス マニュアルのセクション 2.3 を参照してください。

3.3.1 クロック ソースの選択

クロック ソースを選択する際の最大の検討事項は、精度です。USB 仕様では、クロックの許容誤差が 2500ppm である必要があります。ソースがこの仕様の範囲外になると、安定した性能が低下したり、USB 準拠の検証に失敗する可能性があります。

リファレンス クロックを供給する方法としては、3 つの選択肢があります。

表 3-1. クロック ソースの違い

ソース	周波数範囲	使用に適した状況
外部クロック ソース	4 ～ 48MHz 水晶発振器、PLL からより高い周波数へ	水晶発振器は最高の精度と USB 準拠を備えているため、ユーザーは USB 仕様に準拠できます。
内部クロック ソース	4 ～ 32MHz、PLL からより高い周波数へ	内部クロックは 2500ppm の許容範囲を超えることが多いため、通常は推奨されませんが、プロトタイピングには使用可能です。
USBFLL	48～60MHz	このデバイスは SOF パケットを活用して同期を行うため、外部水晶振動子のためのスペースはありません。ただし、スペースとコストに制約がある設計では、クロックを同期するためにアクティブな USB 接続が必要です。

外部水晶振動子を使用する場合は、最高の性能を実現するため、『水晶発振器』ガイドに従っていることを確認してください。

3.3.2 クロック周波数の選択

MSPM0 の USBFLL クロック ソースは SOF パケットを利用して同期を行うため、幅広い周波数を使用することが可能です。各 MSPM0 ヘッダ ファイルには、SYSPLL または USBFLL ソース用に事前定義された定数が含まれています。また、SYSPLLCLK1 で追加の構成が可能で、最大 32 の除数を使用できます。

USB 機能では 48MHz が必要ですが、この目標を達成する方法は複数あります。多くの場合、48MHz 水晶振動子は、シグナル インテグリティとクロックの簡素さという利点があるにもかかわらず、高価で、EMI が高く、より厳格な設計要件を持つ傾向があるため、設計の範囲外になる可能性があります。

多くの場合、設計者は PLL (つまり SYSPLLCLK) に複数の HFXT 水晶を使用します。これにより、ユーザーは PLL に 6 つの 8MHz 水晶振動子、または PLL に 3 つの 16MHz 水晶振動子など、より低い周波数の水晶振動子の組み合わせを使用できます。この方法には、低コストであることと EMI を低減できるという利点があります。ただし、クロックツリーの複雑さとレイアウトの複雑さが増大します。

多くの場合、USBFLL クロック ソースを利用することはデバイスにとって信頼性の高いオプションです。水晶振動子を追加しないため (スペースとコストを削減)、スペースとコストが制限されたデバイスに最適です。ただし、FLL ではホストをロックして同期するために複数の SOF フレームが必要なため、レイテンシが大きくなることを考慮する必要があります。これは、USB に接続している間のみ動作するバスパワー デバイスに最適なオプションです。

3.4 実装例

図 3-2 は、評価キット LP-MSPM0G5187 に基づく疑似回路図です。

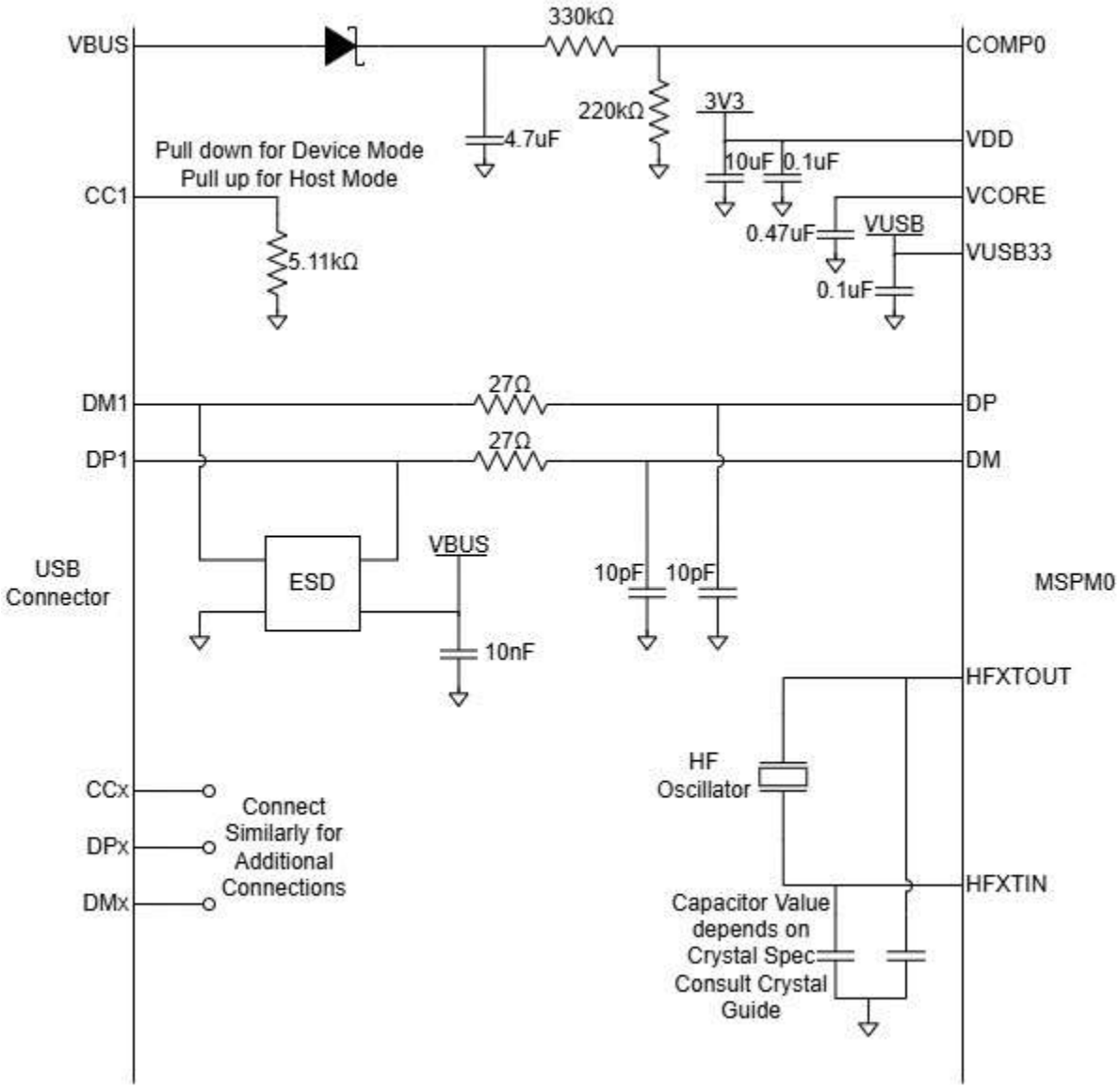

図 3-2. デバイス構成のリファレンス デザイン例

図 3-2 は簡略化されていますが、回路内の実装に注意して検証する必要がある複数の重要な機能があります。

- 前の例では、VDD、VCORE、VUSB33 は接続解除されたままにしています。デバイスのデータシートに従って、正しいデカップリング コンデンサの値が使用されていることを確認します。
- D+ は、新しい接続が確立されたときにホストに通知する信号であるため、デバイス側で DP にプル アップする必要があります。
- ESD は USB デバイス、特にマウス、キーボード、ヘッドセットなど人間が一般的に取り扱うデバイスでは重要です。図 3-2 のデバイスは TDP4E05U06 です。
- HFXTOUT および HFXTIN に接続された外部クロック ソースには、水晶振動子、共振器、または外部クロック ソースを使用できます。水晶振動子を使用する場合は、セクション 3.3.1 の水晶振動子ガイドに従ってください。

4 ソフトウェアの概要

4.1 USB スタック：特長

TinyUSB Library は、MSPM0SDK を使用した USB デバイス開発の基盤となります。この API は、最も一般的な USB デバイス クラスのうち、4 つをサポートしています。

- 通信デバイス クラス (CDC): シリアル通信デバイスに使用される USB デバイス クラス。CDC を使用すると、USB デバイスが従来のシリアル ポート (COM) をエミュレートすることができ、ホストとデバイス間でのデータ交換が可能になります。
- ヒューマン インターフェイス デバイス (HID) クラス：キーボード、マウス、ゲーム コントローラなどのユーザー入力デバイス用に設計された USB デバイス クラスです。HID デバイスは標準化されたプロトコルを使用して、ほとんどのオペレーティング システムへのドライバレス インストールが可能です。
- オーディオ デバイス クラス (UAC): ホストとデバイス間でデジタル オーディオ データを送受信するための USB デバイス クラス。UAC を使用すると、USB デバイスをホストに対して標準オーディオ IO デバイスとして認識できます。
- マス ストレージ クラス (MSC): デバイスをホスト オペレーティング システムの外部ストレージ ドライブとして認識できるようにする USB デバイス クラス。標準のファイル システム操作を可能にします。

これらのクラスは、汎用の用途に適した選択肢を提供します。インターフェイスの選択方法については、デバイス クラスの選択セクションを参照してください。

API には次のような特長があります。

- クロスプラットフォーム サポート
- ホストおよびデバイス スタック
- 複数のデバイス クラス
- クリーンで読みやすいコードベース
- MIT ライセンス取得済み
- 最小限のリソース要件
- マルチ構成のサポート

MSPM0SDK にはいくつかの例があり、本文書の作成時点では以下の例が含まれています。

表 4-1. MSPM0SDK で提供されている例

CDC の例	HID の例	MSC の例	UAC の例	汎用的な例
cdc_acm_uart	hid_cdc_composite_ti	msc_dual_lun	uac_microphone_i2s	device_billboard
cdc_dual_ports	device_hid_composite	sd_card_bridge	uac2_headset	
hid_cdc_composite_ti	device_hid_generic_inout	msc_file_explorer	uac2_speaker_fb	
billboard_cdc	hid_keyboard_ti		audio_test	
	hid_mouse_ti			
	hid_multiple_interface			

これらの例は、MSPM0G5187 LaunchPad を使用した迅速なテストと開発に利用できます。プロトタイピングの開始方法の詳細については、セクション 5 を参照してください。

4.2 SysConfig ディスクリプタ ツール

TI は、*SysConfig* を使用するディスクリプタのシンプルな構成を提供しています。このツールを使用すると、MSPM0 デバイス上で多くのペリフェラルを構成できます。USB 対応 MSPM0 デバイスについては、*TinyUSB* というタイトルのセクションがあり、ディスクリプタを手動でコーディングする必要なく、構成と生成が可能です。

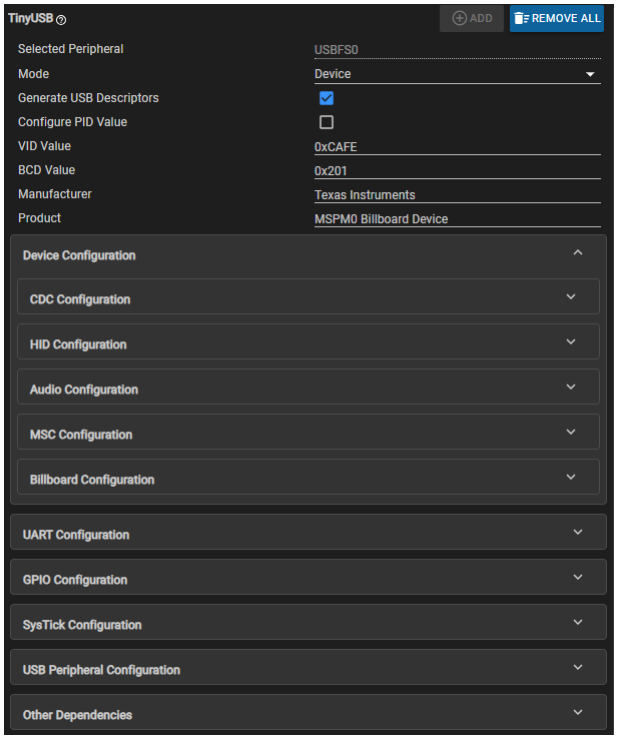

図 4-1. Code Composer Studio 経由で提供される SysConfig エディタのスクリーン キャプチャ

このツールを使用すると、以下の機能を構成できます。

- モード: デバイス / ホスト
- VID 値
- BCD 値
- 製品コードおよびメーカーコード
- デバイス クラスの構成
  - RX FIFO サイズ (CDC のみ)
  - TX FIFO サイズ (CDC のみ)
  - エンドポイント転送バッファのサイズ
  - スピーカー フィードバック、エンコード、およびデコードの有効化 (UAC のみ)
- デフォルトの Billboard 識別子と代替 Billboard 識別子
- UART インスタンスとピンマルチプレクサの選定の構成
  - クロック ソース
  - クロック分周比
  - ボー レート
  - ワード長
  - パリティ
  - ストップ ビット
  - HW フロー制御
  - DMA の構成
  - 割り込み構成
- GPIO 依存関係の構成
  - 方向

  - 初期値
  - ピンマルチプレクサ
  - 割り込み
  - 内部抵抗
- Systick の依存関係の構成
- USB クロック ソースの構成し

ディスクリプタの作成は面倒で、誤りが増える可能性があるため、SysConfig を利用して USB ディスクリプタを生成することは大きな助けとなります。間違ったディスクリプタに起因する障害は難読化される傾向があり、それらのミスを追跡すると、デバッグ時間が大幅に増加する可能性があります。

SysConfig は、CDC、HID、UAC、MSC インターフェイスの任意の組み合わせに対して、最初の試行で大幅に少ない労力で信頼性の高いディスクリプタを生成します。

このツールは、アプリケーションが相互作用する USB インターフェイスを構築するものと考えてください。これは、MSPM0 TinyUSB プロジェクトを開発するための最初の手順です。

4.3 デバイス クラスの選択

USB 設計を作成する最初の手順の一つは、適切なデバイス クラスを選択することです。この選択によって、デバイスは表 4-2 に掲載されている特定の特性セットに制限されます。

表 4-2. 4 つのサポートされているデバイス クラスの比較

特性	CDC (通信デバイス クラス)	HID (ヒューマン インターフェイス デバイス クラス)	MSC (マス ストレージ クラス)	UAC (オーディオ デバイス クラス)
ホストで生成されたインターフェイス	仮想 COM ポート	ヒューマンインターフェイスデバイスです	ストレージのボリューム	オーディオ デバイス
このインターフェイスに関する業界の専門知識	COM ポートは業界で一般的であり、広くサポートされ、よく理解されています	COM ポートやストレージ ボリュームとは異なり、HID インターフェイスはある程度 USB に固有で、業界ではあまり知られていません	ストレージ ボリュームは業界で一般的であり、広くサポートされ、理解されています	UAC デバイスは 20 年近くにわたって使用されてきました。広くサポートされ、よく理解されています
ホストへのインストール	Windows PC は、エンド ユーザーとの相互作用が必要なデバイスのインストール プロセスを実行する必要があります。(1) この Windows PC の管理者権限が必要です 実際のバイナリ ファイルはすでに Windows に存在していますが、ユーザーは INF ファイルを提供する必要があります	ほとんどのオペレーティング システムで静かに読み込まれ、すぐに動作を開始します。 ドライバ ファイルは必要ありません	ほとんどのオペレーティング システムで静かに読み込まれ、すぐに動作を開始します。 ドライバ ファイルは必要ありません	ほとんどのホストでクラス準拠の読み込みを行い、インストールは不要です
エンド ユーザーがどのように相互作用するか	COM ポートとのインターフェイスとして機能する、ホスト上のアプリケーション アプリケーションは、カスタム、または COM ポートを使用する既存のアプリケーションのいずれかです	HID デバイスとインターフェイスするホスト上のカスタム アプリケーション	デバイスは、ストレージボリュームをシステムにマウントします。アプリケーションは、ボリューム上のファイルの読み取りと書き込みを行います。アプリケーションは、カスタム、またはファイルの読み取りまたは書き込みを行う任意のアプリケーションです	UAC インターフェイスを使用して、デバイスとの間でオーディオがストリーミングされます
ドライバ認証の必要性	INF ファイルが WHQL 認証 (署名済み) されていない限り、Windows はドライバが「未認証」であることを報告します	*未認証*メッセージは生成されません	• *未認証*メッセージは生成されません	• *未認証*メッセージは生成されません

表 4-2. 4 つのサポートされているデバイス クラスの比較 (続き)

特性	CDC (通信デバイス クラス)	HID (ヒューマン インターフェイス デバイス クラス)	MSC (マス ストレージ クラス)	UAC (オーディオ デバイス クラス)
コードのフットプリントと複雑さ	小型のコード フットプリント (4K ～ 6K) シンプルなアーキテクチャ	小型のコード フットプリント (4K ～ 6K) シンプルなアーキテクチャ	より大きいコード フットプリント (8K ～ 15K) ファイル システムが必要です。コスト、サイズ、複雑さが増大します。	さらに大きなフットプリント インターフェイスのアイソクロナス転送が必要になり、複雑さが増します
スループット	高速 (数百 KB / 秒) USB のバルク転送が使用されます。	遅い (64KB / 秒) は割り込み USB 転送を使用します。	高速 (数百 KB / 秒) USB のバルク転送が使用されます。	サンプル レートとビット深度に依存しています 通常は数百 KB / 秒です
ホストとデバイス間のポイント ツー ポイント通信に適しています	はい	はい	いいえ	はい
一括データ転送に適しています	はい	いいえ	はい	はい (オーディオ)

4.3.1 USB デバイス クラスを決定するプロセスの例

デバイス クラスの選択が明確な場合があります。それ以外の場合、アプリケーションは汎用的であるとみなされ、開発者に選択肢が与えられる場合もあります。デバイス クラスの選択にはさまざまな方法がありますが、ユーザーは図 4-2 を使用してデバイス クラスを選択できます。

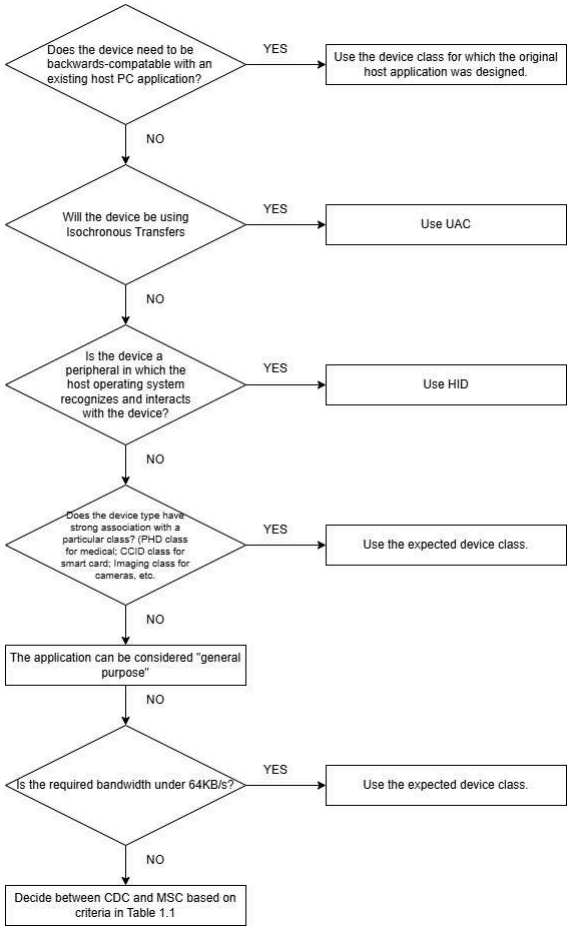

図 4-2. 正しいデバイス クラスを選択するためのデシジョン ツリーの例

4.4 ベンダ ID (VID) と製品 ID (PID) の選択方法

USB に関してよくある質問の一つは、VID と PID の選択方法です。USB デバイスがホストに接続されると、ホストは USB ディスクリプタの入力を求めます。これらはホストに、デバイスの性質とデバイスの機能を伝えます。

ディスクリプタには、16 ビットの VID および PID 値が含まれています。VID は特定のベンダおよび OEM に関連付けられており、PID はそのベンダが販売する製品に関連付けられています。

たとえば、ベンダ Vendor1 が最初の USB 製品 (Product1) を販売した場合、ベンダはその会社に関連付けられている VID を取得し、Product1 に関連付ける PID を選択する必要があります。ベンダが後で Product2 をリリースすると、ベンダは同じ VID を使用しますが、現在は新しい PID を使用する必要があります。フィールドで競合が発生する可能性がある PID が重複しないようにするのはベンダの責任です。

したがって、VID と PID の固有の組み合わせにより、USB ホストは USB 製品タイプを区別できます。Product1 と Product2 の VID と PID が同じで、フィールドのホストが両方の製品に遭遇した場合、ホストが 2 つの製品を混乱させ、不適切なドライバをロードして競合する可能性があります。一般的なルールとして、デバイスに USB ディスクリプタの違いがある場合、そのデバイスには異なる PID が必要です。

4.4.1 VID および PID の選択と取得

VID は、USB を監視する標準団体である USB Implementers Forum (USB-IF) によって割り当てられます。ベンダは、USB-IF に参加して VID を入手するか、参加せずに VID のライセンスを取得するかを選択できます。本文書の執筆時点では、前者は年間 $5000、後者は 2 年間のライセンスで $3500 かかります。(詳細については http://www.usb.org/developers/vendor/ を参照してください。)

前世代の MSP430 デバイスとは異なり、MSP では VID 共有プログラムを実行していません。このプログラムを利用するお客様がオンラインにアクセスして、固有の PID を請求し、TI の VID 下で使用することができますお客様は、前述のように USB-IF に VID を登録し、当社のデバイスで VID と関連する PID を使用する必要があります。

4.4.2 開発中の VID および PID の使用

USB デバイスに固有の VID と PID のペアを用意することは、競合を防ぐために重要です。特定の USB ホストは、最初に与えられた VID および PID を見つけた後に、USB デバイスのドライバ要件に関する情報を保存します。同じ VID と PID を持つ後続のデバイスでは、同じホストドライバのセットアップが必要であると想定できる必要があります。したがって、市場にリリースされた後は、製品の VID と PID を変更してはなりません。

ただし、開発中に開発者が最終的な USB ディスクリプタ セットに到着するにつれて、VID と PID を変更する必要が生じる場合があります。開発者は、使用中のホスト マシンで競合を防止する必要があります。これは、USB ディスクリプタが変更されるたびに新しい PID 値を使用することによって実行できます。または、元の PID を使用することもできますが、デバイスをシステムからアンインストールして再インストールする必要があります。詳細については、USB 開発者パッケージの USB API プログラマ ガイドを参照してください。

4.5 TinyUSB API プログラマ ガイドおよび例

SDK に含まれる TinyUSB ユーザー ガイドには、TinyUSB を使用したソフトウェア開発の詳細情報や、表 4-1 に記載されている多くの例の解説や、図 4-1 の USB ディスクリプタ ツールの解説が含まれています。

加えて、TinyUSB の Web サイトを通じて、オンラインで多数の例を公開しており、API の詳細について説明しています。これには、アプリケーション設計に関連する多数の追加定義、ソフトウェア コンセプト、リソースが含まれます。

5 設計開始の手引き MSPM0 USB の評価

LP-MSPM0G5187 評価キットには、ホストとデバイスの両方の設計の評価と開発を開始するために必要なブレークアウトがすべて装備されています。

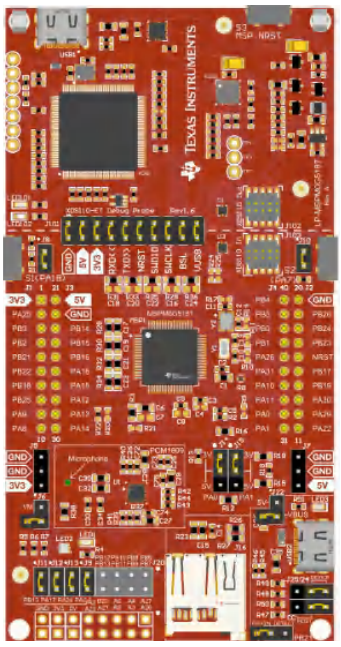

図 5-1. LP-MSPM0G5187 評価キット

この評価キットは複数のジャンパを搭載しており、以下のような多数のハードウェア / ソフトウェア機能をデバッグすることができます。

- UAC デバイス向けのマイクと I2S ベースのオーディオ ADC
- MSC アプリケーション用の microSD スロット
- 2 個の LED (1 個の RGB 対応 LED)
- 3 個のボタン
- 汎用デバイスのテストに使用できる 40 個以上のピン。
- 2 個の USB-C コネクタ (1 個はデバッグ用、もう 1 個は MSPM0 接続用)
- プログラミング、デバッグ、EnergyTrace™ テクノロジーに適したオンボード デバッグ プローブ。

LaunchPad 開発キットには 40 ピンの拡張ヘッダが搭載されており、さまざまな BoosterPack™ プラグイン モジュールとの接続が可能なため、迅速なプロトタイピングが簡単になります。ワイヤレス、ディスプレイ、センサなどの機能も迅速に追加できます。BoosterPack プラグイン モジュールを設計するか、TI やその他の企業からすでに提供されている多数のモジュールから選択します。この 40 ピン インターフェイスは標準に準拠したすべての 20 ピン BoosterPack プラグイン モジュールと互換性を持ちます。

TI の Code Composer Studio IDE (CCS) は、LaunchPad を使用した設計とテストのための主要な開発環境です。USB API の例は MSPM0SDK を介して提供されており、最新の例を利用できるようにするために、通常は最新バージョンをダウンロードすることをお勧めします。

6 まとめ

MSPM0 ファミリの USB 対応デバイスは、マスストレージ ホストとともに、オーディオ デバイス、ヒューマン インターフェイス デバイス、通信デバイスなど、複数のタイプのアプリケーションをサポートできます。USB 2.0 フルスピード モジュールは、最大 16 個のエンドポイント、専用のエンドポイント RAM、DMA トリガ、統合型フルスピード PHY、DFU/BSL サポートを搭載しています。これらの各モジュールの詳細については、テクニカル リファレンス マニュアルと USB パーツに関連するデータシートを参照してください。MSPM0 の USB ペリフェラルは、CCS 開発環境および TinyUSB アーキテクチャと組み合わせることで、本アプリケーション ノート全体のハードウェア ガイドラインとともに使用すれば、USB アプリケーションの迅速な開発を可能にする簡便な開発環境を実現できます。こうした設計には、USB-C モードの表示、セルフ パワーおよびバス パワー電源アーキテクチャの検討事項、配線とレイアウトの規則など、多くの重要な検討事項があります。MSPM0 内のクロック モジュールはデバイス アプリケーションを処理できますが、USB プロトコルへの準拠を維持するためには慎重に検討する必要があります。

7 参考資料

1. テキサス インスツルメンツ、『MSP430 マイコン (MCU) を使用した USB 設計の開始』、アプリケーション ノート。
2. テキサス インスツルメンツ、LP-MSPM0G5187 LaunchPad、開発キット。
3. テキサス インスツルメンツ、MSPM0 G シリーズ 80MHz マイコン、テクニカル リファレンス マニュアル。
4. テキサス インスツルメンツ、MSPM0 Academy、Resource Explorer。
5. TinyUSB、TinyUSB の概念、ウェブページ。

8 USB 用語集

1. バルク転送:USB バス上の 4 つのデータ転送タイプの一つ。バルク転送は、大量のデータを移動するために設計されています。バス上の空き帯域幅 (つまり、他の転送タイプでまだ使用されていない帯域幅) を使用できます。これにより、最高のデータ レートを達成できますが、予約済みの帯域幅がないため、ビジーなバスでは、バルク転送は小さな帯域幅を受信したり、高いレイテンシを発生させたりすることがあります。転送タイプは、USB インターフェイスのタイプの選択によって決まります。たとえば、CDC および MSC インターフェイスではバルク転送が使用されます。
2. 複合 USB デバイス:複数の USB インターフェイスを搭載した物理的 USB デバイス (USB コネクタ 1 個)。たとえば、CDC インターフェイス 2 個や CDC+HID などです。ホストは、各インターフェイスを個別の論理エンティティとして列挙します。
3. 制御転送:USB バス上の 4 つのデータ転送タイプの一つ。制御転送は、USB ディスクリプタのレポートなど、接続設定の管理タスクを処理します。また、ホストはその他の USB デバイスの要求も送信し、デバイスは制御転送を使用して応答します。これらの転送専用の USB エンドポイント、エンドポイント 0 (EP0) があります。
4. デバイス クラス:特定のデバイス クラスに定義された USB プロトコル。一般的なデバイス クラスには、通信デバイス クラス (CDC)、ヒューマン インターフェイス デバイス (HID) クラス、USB オーディオ クラス (UAC)、マス ストレージ クラス (MSC) があります。
   a. CDC: シリアル通信デバイスに使用される USB デバイス クラス。CDC を使用すると、USB デバイスが従来のシリアル ポート (COM) をエミュレートすることができ、ホストとデバイス間でのデータ交換が可能になります。
   b. HID: キーボード、マウス、ゲーム コントローラなどのユーザー入力デバイス用に設計された USB デバイス クラス。HID デバイスは標準化されたプロトコルを使用して、ほとんどのオペレーティング システムへのドライバレス インストールが可能です。
   c. UAC: ホストとデバイス間でデジタル オーディオ データを送受信するための USB デバイス クラス。UAC を使用すると、USB デバイスをホストに対して標準オーディオ IO デバイスとして認識できます。
   d. MSC: デバイスをホスト オペレーティング システムの外部ストレージ ドライブとして認識できるようにする USB デバイス クラス。標準のファイル システム操作を可能にします。
5. デバイスのインストール:USB デバイスを初めて列挙するとき、ホストは 1 回限りの機能を実行してデバイスをインストールする場合があります。たとえば、Windows はデバイスの VID と PID をインデックスとして使用して、システム レジストリにデバイスに関する情報を記録します。以降の列挙では、ホストはデバイスに関する情報の大部分をレジストリから取得します。デバイスのインストールはサイレント (ほとんどエンド ユーザーから見えない) である場合や、Windows 上の CDC の場合、ユーザーの操作が必要になる場合があります。
6. エンドポイント:パイプの端。これは、そのパイプの USB デバイス上のメールボックスとして機能します。通常、デバイスには複数のアクティブ エンドポイントがあります。ホストがバス上で通信を行うとき、ホストはまず物理的 USB デバイスを識別し、次に通信を希望するデバイス内のエンドポイント番号を識別します。エンドポイントには、作成された USB インターフェイスに従って特定の機能が割り当てられます。HID/MSC はそれぞれ 1 つの IN エンドポイントと 1 つの OUT エンドポイントを使用し、CDC は 2 つの IN エンドポイントと 1 つの OUT エンドポイントを使用します。MSP430 API スタックのエンドポイント管理は、ディスクリプタ ツールによって完全に自動化されています。
7. 列挙:ホストが物理 USB デバイスを識別するためにデバイスを問い合わせ、適切なドライバを読み込むことで、ホスト アプリケーションがデバイスとインターフェイスを確立できるようにするプロセス。列挙は、デバイスが接続されるたびに行われます。
8. 割り込み転送:4 つの USB データ転送タイプの一つ。割り込み転送は、レイテンシ、帯域幅、および配信用に設計されています。ただし、帯域幅は、フレーム (1ms) ごとに 1 つの USB パケット (フルスピード USB の場合は 64 バイト) のみに制限されます。転送タイプは、USB インターフェイスのタイプの選択によって決まります。たとえば、HID インターフェイスでは割り込み転送が使用されます。
9. アイソクロナス転送:4 つの USB データ転送タイプの一つ。アイソクロナス転送では、レイテンシと帯域幅が保証されますが、配信はできません。つまり、エラー チェックで破損したデータが表示された場合、再試行は行われません。このタイプは、オーディオおよびビデオのストリーミングを目的としています。再試行によって中断が発生し、パケットの損失よりもユーザーに顕著になるアプリケーションです。
10. INF (*.inf) ファイル:Windows での USB デバイスのインストール中に必要なテキストベースのファイル。Windows がデバイスを特定のドライバに関連付けることができます。一部のデバイス クラスでは、Windows に INF が含まれており、デバイスのサイレント インストールが可能です。CDC の場合、Windows はエンド ユーザーに INF ファイルの入力を求めるプロンプトを表示します。

11. パイプ：ホストとデバイス間の単一通信線。パイプは、IN (ホストへ) または OUT (ホストから) のいずれかです。それらは特定の転送タイプ (バルクまたは割り込みなど) によって特徴付けられています。
12. ディスクリプタ：列挙時に USB デバイスがホストに提供するデータ構造。デバイスの機能、構成、インターフェイス、エンドポイントが記述されています。一般的なディスクリプタには、デバイス、構成、インターフェイス、およびエンドポイント ディスクリプタがあります。
13. フルスピード USB: USB は 12Mbps で動作し、USB 1.1 では標準速度、USB 2.0 デバイスではオプションです。
14. ホスト：バスを管理し、すべてのデータ転送を開始する USB コントローラ。通常は PC ですが、USB ホスト機能を備えた組込みシステムの場合もあります。

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