参考資料

Application Note

STM32® から Arm® ベースの MSPM0 への移行ガイド

TEXAS INSTRUMENTS

概要

このアプリケーション・ノートは、STMicroelectronics STM32® プラットフォームからテキサス・インスツルメンツの MSPM0 MCU エコシステムへの移行を支援します。このガイドでは、MSPM0 の開発およびツール・エコシステム、コア・アーキテクチャ、ペリフェラルに関する考慮事項、およびソフトウェア開発キットについて説明します。この目的は、2 つのファミリの違いを強調し、STM32 エコシステムに関する既存の知識を活用して、MSPM0 シリーズの MCU で迅速に開発を開始することです。

目次

商標

1 MSPM0 製品ラインアップの概要

1.1 概要

MSP430™ MCU は、テキサス・インスツルメンツの従来型マイコンとして、30 年近く使用されてきました。最新世代では MSPM0 ファミリが導入されています。MSPM0 マイクロコントローラ (MCU) は、強化された ARM® Cortex®-M0+ 32 ビット・コア・プラットフォームをベースとする MSP の高集積超低消費電力 32 ビット MCU ファミリの一部です。コスト最適化されたこれらの MCU は、高性能アナログ・ペリフェラルの統合、拡張温度範囲のサポート、および小型フットプリント・パッケージを実現します。テキサス・インスツルメンツの MSPM0 ファミリの低消費電力 MCU は、アナログとデジタルの統合度が異なるデバイスで構成されており、エンジニアはプロジェクトのニーズを満たす MCU を見つけることができます。MSPM0 MCU ファミリは、ARM Cortex-M0+ プラットフォームと超低消費電力のシステム・アーキテクチャを組み合わせたもので、システム設計者は性能向上と消費電力低減を同時に実現できます。

MSPM0 MCU は、STM32 MCU に代わる競争力のある選択肢を提供します。このアプリケーション・ノートは、デバイスの機能とエコシステムを比較することで、STM32 MCU から MSPM0 MCU への移行を支援します。

1.2 STM32 MCU と MSPM0 MCU の製品ラインアップの比較

表 1-1. テキサス・インスツルメンツ MSPM0Gx/LX と STM32G0/F0 シリーズの比較

	ST Micro STM32G0 シリーズ	ST Micro STM32F0 シリーズ	テキサス・インスツルメンツ MSPM0 MSPM0Gx シリーズ	テキサス・インスツルメンツ MSPM0 MSPM0Lx シリーズ
コア / 周波数	CM0+ / 64MHz	CM0 / 48MHz	CM0+ / 80MHz	CM0+ / 32MHz
電源電圧	1.7V～3.6V	2V～3.6V	1.62V～3.6V	1.62V～3.6V
温度	-40℃～125℃	-40℃～105℃	-40℃～125℃	-40℃～125℃
メモリ	512KB～16KB	256KB～16KB	128KB～32KB	64KB～8KB
RAM	最大 144KB	最大 32KB	最大 32KB	最大 4KB
GPIO (最大)	90	88	60	28
アナログ	1 個の 2.5Msps 12 ビット ADC 1 個の 12 ビット DAC 3 個のコンパレータ	1 個の 1Msps 12 ビット ADC 1 個の 12 ビット DAC 2 個のコンパレータ	2 個の 4 Msps 12 ビット ADC 1 個の 12 ビット DAC 3 個の高速コンパレータ 2 個のオペアンプ	1 個の 1Msps 12 ビット ADC 1 個の高速コンパレータ 2 個のオペアンプ
通信 (最大)	3 個の SPI 3 個の I²C Fast+ 6 個の UART (LIN) 2 個の CAN-FD 1 個の USB	2 個の SPI 2 個の I²C Fast+ 8 個の UART (LIN) 1 個の CAN	2 個の SPI 2 個の I²C Fast+ 4 個の UART (LIN) 1 個の CAN-FD	1 個の SPI 2 個の I²C Fast+ 2 個の UART (LIN)
タイマ	8	4	7	4
アドバンス・タイマ	あり (1)	あり (1)	あり (3x)	なし
ハードウェア・アクセラレータ	該当なし	該当なし	オプション	該当なし
セキュリティ	CRC、TRNG、AES256	CRC	CRC、TRNG、AES256	CRC
低消費電力	アクティブ：100µA/MHz スタンバイ (RTC)：1.5µA	アクティブ：281µA/MHz スタンバイ (RTC)：2.5µA	アクティブ：85µA/MHz スタンバイ (RTC)：1.5µA	アクティブ：85µA/MHz スタンバイ：1.5µA

2 エコシステムと移行

MSPM0 MCU は、ハードウェアおよびソフトウェアの大規模なエコシステムによってサポートされており、リファレンス・デザインやサンプル・コードを利用して設計をすぐに開始できます。MSPM0 MCU は、オンライン・リソース、MSP Academy を使用したトレーニング、TI E2E™ サポート・フォーラムによるオンライン・サポートによってもサポートされています。

2.1 ソフトウェア・エコシステムの比較

表 2-1. MSPM0 と同等の STM32 ソフトウェア・ツール

	STM32	MSPM0
IDE	CubeIDE	Code Composer Studio™ IDE (CCS)
ソフトウェアの設定	CubeMX	SysConfig
スタンドアロン・プログラミング	CubeProgrammer	UniFlash
ディスプレイ / デモ GUI エディタ	CubeMonitor	GuiComposer

2.1.1 MSPM0 ソフトウェア開発キット (MSPM0 SDK)

MSPM0 SDK には、テキサス・インスツルメンツの MSPM0+ マイコン・デバイス上でアプリケーションを迅速に開発するのに役立つソフトウェア API、サンプル、資料、ライブラリが含まれています。各機能分野とすべてのサポート・デバイスの使用法を解説しており、設計を開始する際の出発点になります。さらに、MSPM0 SDK には対話型の MSP Academy トレーニングが付属しており、ガイド付きの学習パスを提供します。

サンプル・フォルダは RTOS と非 RTOS のサブフォルダに分割されています (現在、非 RTOS のみがサポートされています)。これらのフォルダには、各 LaunchPad™ 開発キットのサンプルが含まれており、下位レベルの DriverLib サンプル、上位レベルのテキサス・インスツルメンツ・ドライバ・サンプル、GUI Composer、LIN、IQMath などのミドルウェアのサンプルなど、カテゴリ別に整理されています。詳細については、『MSPM0 SDK ユーザー・ガイド』を参照してください。

2.1.2 CubeIDE と Code Composer Studio IDE (CCS) の比較

テキサス・インスツルメンツの Code Composer Studio IDE (CCS) は、STM32 の CubeIDE に相当します。CCS は無償の Eclipse ベースの IDE で、テキサス・インスツルメンツのマイコン (MCU) と組込みプロセッサ・ポートフォリオをサポートしています。CCS は、最適化された C/C++ コンパイラ、ソース・コード・エディタ、プロジェクトのビルド環境、デバッガ、プロファイラ、その他の多くの機能を含む、組込みアプリケーションの開発とデバッグに使用する一連のツールで構成されています。CCS は、デスクトップまたはクラウド・ベースの IDE として利用できます。

CCS は、統合型の TI Resource Explorer に、MSPM0 デバイス構成と SysConfig からの自動コード生成、MSPM0 サンプル・コードとアカデミー・トレーニングを統合しています。CCS は、一体型の開発ツールを提供します。

CCS に加えて、MSPM0 デバイスは以下の表に示す業界標準 IDE でもサポートされています。

表 2-2. MSPM0 でサポートされている IDE

IDE	MSPM0
CCS	✓
IAR	✓
Keil	✓

2.1.3 CubeMX と SysConfig の比較

SysConfig は、ピン、ペリフェラル、無線、サブシステム、他の機能を構成するために使用できる、直観的で包括的なグラフィカル・ユーティリティのコレクションです。これは、STM32 CubeMX と同等のテキサス・インスツルメンツのツールです。SysConfig を使用すると、コンフリクトの管理、表面化、解決をビジュアルな方法で実行できるので、より多くの時間をアプリケーションの差異化に割り当てることができます。このツールの出力には C ヘッダとコード・ファイルが含まれており、MSPM0 SDK サンプルと組み合わせて使用することも、カスタム・ソフトウェアの構成に使用することもできます。SysConfig は CCS に統合されていますが、スタンドアロン・プログラムとしても使用できます。

詳細については、『MSPM0 SysConfig ガイド』を参照してください。

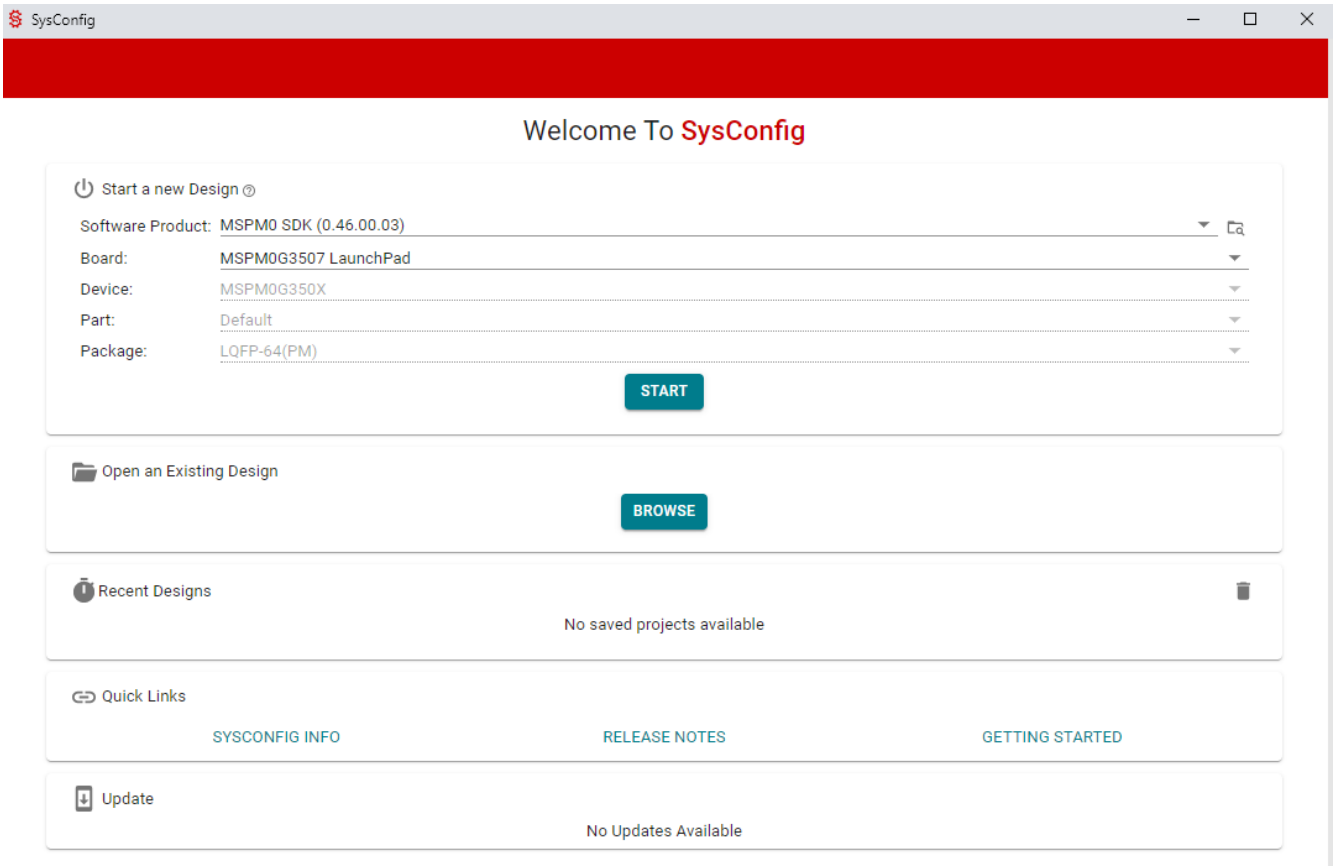

図 2-1. MSPM0 SysConfig

2.2 ハードウェア・エコシステム

MSPM0 の評価基板は LaunchPad 開発キットのみです。LaunchPad キットは使いやすい評価基板で、MSPM0 の開発を開始するために必要なものがすべて含まれています。これには、EnergyTrace™ テクノロジーを使用したプログラミング、デバッグ、消費電力測定用のオンボード・デバッグ・プローブが含まれています。MSPM0 LaunchPad には、オンボード・ボタン、LED、温度センサなどの回路も搭載されています。さまざまなブースタパック・プラグイン・モジュールをサポートする 40 ピンのブースターパック™・プラグイン・モジュール・ヘッダーにより、迅速で簡単なプロトタイプ製作が可能になります。ワイヤレス接続、グラフィカル・ディスプレイ、環境センシングなどの機能も迅速に追加できます。

- LP-MSPM0G3507 LaunchPad 開発キット
- LP-MSPM0L1306 LaunchPad 開発キット

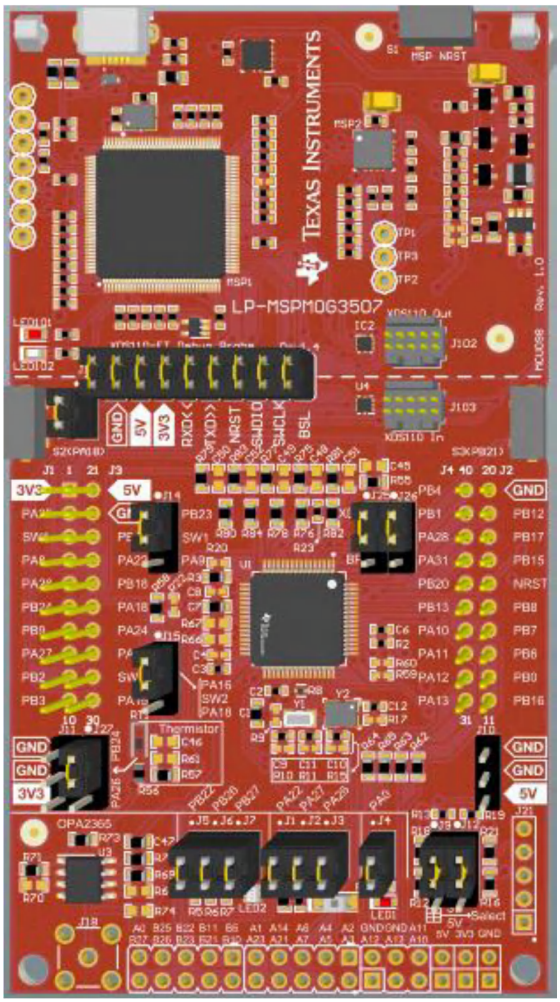

図 2-2. LP-MSPM0G3507 LaunchPad 開発キット

2.3 デバッグ・ツール

デバッグ・サブシステム (DEBUGSS) は、シリアル・ワイヤ・デバッグ (SWD) の 2 線式物理インターフェイスを、デバイス内の複数のデバッグ機能に接続します。MSPM0 デバイスは、プロセッサの実行、デバイスの状態、電力状態 (EnergyTrace テクノロジーを使用) のデバッグをサポートしています。図 2-3 に、デバッガの接続を示します。

MSPM0 は、標準的なシリアル・ワイヤ・デバッグ用の XDS110 および J-Link デバッガをサポートしています。

テキサス・インスツルメンツの XDS110 は、テキサス・インスツルメンツの組込みプロセッサ向けに設計されています。XDS110 は、テキサス・インスツルメンツの 20 ピン・コネクタ (テキサス・インスツルメンツの 14 ピン、ARM 10 ピン、ARM 20 ピンを接続するための複数のアダプタ付属) を経由してターゲット・ボードに接続し、USB 2.0 ハイスピード (480Mbps) を経由してホスト PC に接続します。単一のポッドで幅広い規格 (IEEE1149.1、IEEE1149.7、SWD) をサポートしています。すべての XDS デバッグ・プローブは、ETB (Embedded Trace Buffer、組込みトレース・バッファ) 搭載のすべての ARM と DSP プロセッサに対し、コア・トレースとシステム・トレースをサポートしています。詳細については、XDS110 デバッグ・プローブを参照してください。

J-Link デバッグ・プローブは、デバッグとフラッシュ・プログラミングの経験を最適化するための最も一般的な選択肢です。記録的なブレークダウンを実現するフラッシュ・ローダ、最大 3MiB/s の RAM ダウンロード速度、MCU のフラッシュ・メモリ内に無制限のブレークポイントを設定する機能を活用できます。また、J-Link は CortexM0+ を含む幅広い CPU とアーキテクチャもサポートしています。詳細については、Segger J-Link デバッグ・プローブのページを参照してください。

図 2-3 に、XDS110 プローブから MSPM0 ターゲットへの主要な機能領域とインターフェイスの概略図を示します。

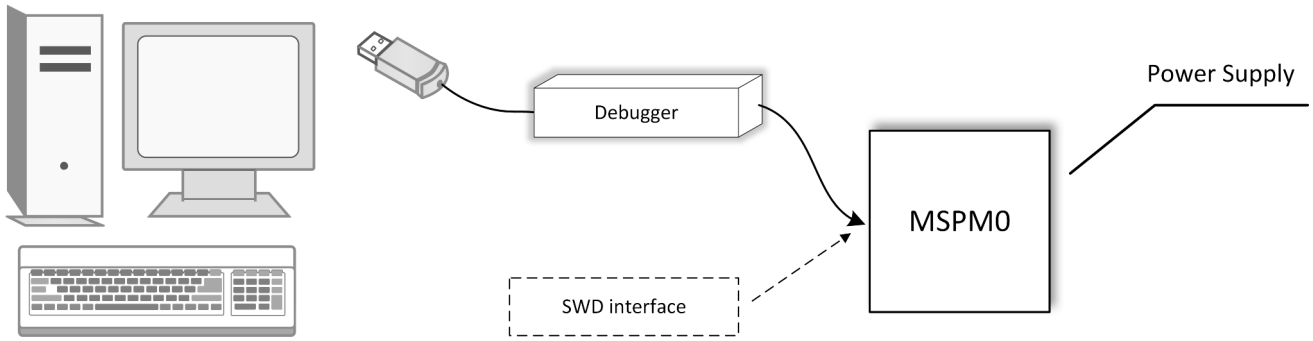

図 2-3. MSPM0 のデバッグ

2.4 移行プロセス

移行の最初のステップは、ポートフォリオを確認し、最適な MSPM0 MCU を選択することです。MSPM0 MCU を選択した後、開発キットを選択します。開発キットには、購入可能な LaunchPad キットとターゲット・ソケット・ボード用の設計ファイルが含まれています。また、テキサス・インスツルメンツは無償の MSPM0 ソフトウェア開発キット (SDK) も提供しており、Code Composer Studio IDE デスクトップのコンポーネントとして利用できます。TI Resource Explorer ではクラウド・バージョンも利用可能です。STM32 から MSPM0 へのソフトウェアの移植については、このアプリケーション・ノートのペリフェラル・セクションを参照してください。最後に、ソフトウェアが移植された後、デバッグ・ツールを使用してアプリケーションをダウンロードし、デバッグします。

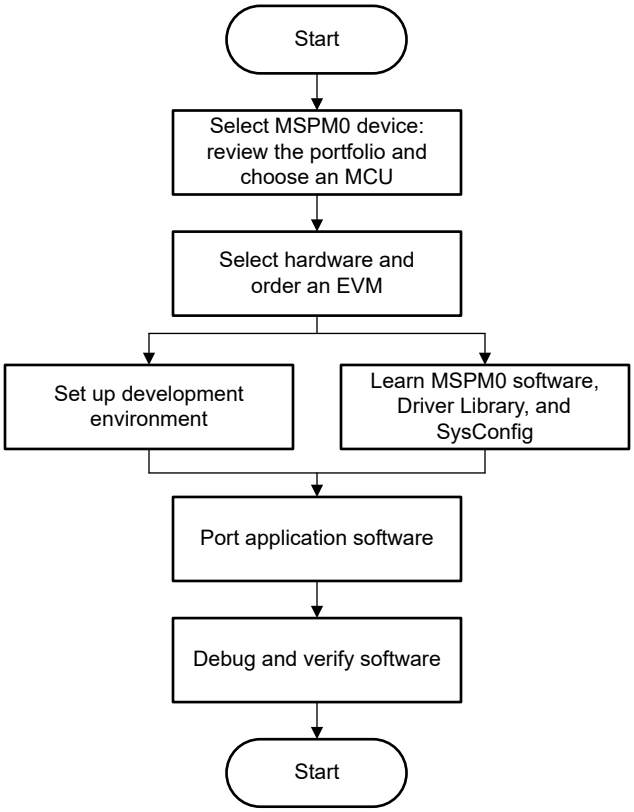

図 2-4. MSPM0 の移行フローチャート

2.5 移行と移植の例

このセクションでは、テキサス・インスツルメンツのエコシステムを理解し、MSPM0 を使用して最善の開発を開始する方法を説明するために、基本的なアプリケーションの段階的な移行プロセスについて説明します。

STM32 から MSPM0 への移植プロセスを示すために、この説明には、既存の ST UART の例を出発点として使用し、基本的な低消費電力 UART モニタ・アプリケーションを STM32G0x から MSPM0 デバイスに移植する手順が含まれています。

ステップ 1.適切な MSPM0 MCU を選択する

移行の最初のステップは、アプリケーションに適した MSPM0 デバイスを選択することです。そのために、このガイドのポートフォリオ・セクションを使用して MSPM0 ファミリを選択できます。製品選択ツールを使用して特定のデバイスに絞り込むことができます。STM32G0 および MSPM0 は両方とも M0+ コアを使用しますが、メモリ・サイズ、電力、主要ペリフェラルなどの機能も考慮する必要があります。また、MSPM0 には多くのピン互換のスケーラブルなオプションがあり、システム内の他の何も変更することなく、より大容量または小型のメモリ・デバイスに簡単に変更できます。

この例では、アプリケーションに最適な MSPM0G3507 を選択しました。

ステップ 2.ハードウェアを選択し、評価基板を注文する

評価基板 (EVM) を使用すると、移行プロセスを迅速化できます。MSPM0 MCU の場合、LaunchPad キットは、開発を開始するための最も簡単なハードウェアです。LaunchPad キットにはプログラマが内蔵されており、迅速な開発を可能にするように設計されているため、使いやすくなっています。

MSPM0G3507 には、LaunchPad 開発キット (LP-MSPM0G3507) が付属しており、ソフトウェアの移植に使用できます。

ステップ 3.ソフトウェア IDE と SDK をセットアップする

ソフトウェアを移植する前に、ソフトウェア開発環境を選択してセットアップする必要があります。セクション 2.1 に MSPM0 でサポートされているすべての IDE を示します。移行と移植のプロセスは、選択したすべての IDE で同様です。MSPM0 SDK の最新バージョンを使用する必要があります。

この例では、テキサス・インスツルメンツの CCS が選択された IDE です。

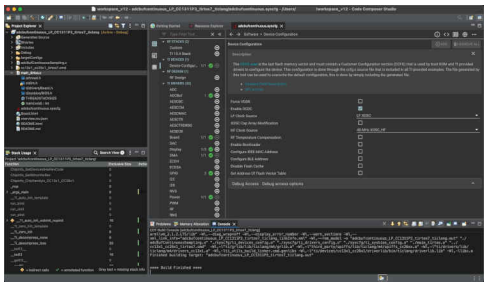

図 2-5. Code Composer Studio IDE

ステップ 4.ソフトウェアの移植

環境の準備ができたら、MSPM0 SDK の使用を開始します。すでに説明したように、MSPM0 SDK は STM32Cube ソフトウェア・パッケージと類似しています。MSPM0 SDK は、ソフトウェア開発のためのさまざまなレイヤを提供します。MSPM0 テキサス・インスツルメンツ・ドライバは STM32Cube HAL と同じレベルで動作しますが、MSPM0 DriverLib は STM32Cube 低レベル・ドライバと同等です。ほとんどの MSPM0 ユーザーは、DriverLib レベルのソフトウェアがアプリケーションに最適であると判断しています。そのため、ほとんどの MSPM0 ソフトウェア・サンプルも DriverLib ベースです。この例では DriverLib を使用しています。

プロジェクトを移植する際の 1 つのオプションは、コードの各セクションを、等価な MSPM0 DriverLib API で置き換えようとすることですが、一般的にはこれが最も簡単な方法ではありません。一般的に、最初に移植するアプリケーション・コードを理解することが最善です。次に、最も近い MSPM0 サンプル・プロジェクトから開始し、元のコード機能に合わせて変更します。STM32CubeG0 の低消費電力 UART の例を使用してこのプロセスを以下に示します。多くのペリフェラルを使用するより複雑なプロジェクトでは、通常、各ペリフェラルに対してこのプロセスが繰り返されます。

ステップ 4a:アプリケーションを理解する

次の説明は、STM32CubeG0 の「LPUART_WakeUpFromStop_Init」というサンプル・プロジェクトからのものです。

```
@par サンプルの説明

LPUART RX ピンで受信した文字が低消費電力モードから MCU をウェークアップできるように、GPIO および LPUART ペリフェラルを構成します。この例は、LPUART LL API をベースにしています。ペリフェラルの初期化では、LL 初期化機能を使用して LL 初期化の使用法を示します。

LPUART ペリフェラルは非同期モード (9600 ボー、8 データ・ビット、1 スタート・ビット、1 ストップ・ビット、パリティなし) に構成されています。
ハードウェア・フロー制御は使用されません。
LPUART クロックは、HSI をベースにしています。

実行例:
リセットおよびシステム構成から起動した後、3 秒間 LED3 がすばやく点滅し、MCU は「Stop 0」モードに移行します (LED3 オフ)。「Stop 0」モード期間の後、PC Com ポート (例:ハイパーターミナルを使用) から LPUART が最初に受信した文字で、MCU は「Stop 0」モードからウェークアップします。

受信した文字の値がチェックされます。
- 特定の値 (「S」または「s」) で LED3 がオンになり、プログラムが終了します。
- 「S」または「s」と異なる場合、プログラムは 3 秒間 LED3 をすばやく点滅させ、「Stop 0」モードに戻り、次の文字がウェークアップするのを待ちます。
```

最初のステップは、MCU の主な設定を理解することです。これは一般に、クロック速度と電力ポリシーです。この例では、UART が低消費電力の Stop0 モードで動作することが唯一の重要な設定であるため、一般的なクロック周波数は指定されていません。低消費電力の UART クロックは、「HIS」または高速の内部発振器に基づいていることを示しています。つまり、外部水晶振動子は使用されていません。UART は、9600 ボー、8 データ・ビット、1 スタート・ビットおよび 1 ストップ・ビット、パリティなしで動作します。ハードウェア・フロー制御は使用していません。アプリケーション側で受信する「S」または「s」をチェックし、LED を点滅させます。

ステップ 4b:最も近い MSPM0 の例を見つける

次のステップは、STM32G0 と MSPM0 の UART モジュールの違いを理解し、MSPM0 SDK で最も近い例を見つけることです。これは、セクション 4 の「UART」セクションを参照することで簡単に実現できます。このセクションでは、UART モジュール間の違いと、UART 関連の MSPM0 SDK サンプル・コードへのリンクについて説明します。この例の SDK で最も近い例は、おそらく uart_echo_interrupts_stanby で、「デバイスが STANDBY モードのとき、割り込みを使用して UART RX/TX がエコーする」ものです。

この MSPM0 の例は、移植されているものと類似していますが、完全に同じではありません。この例では、Stop モードよりも低消費電力モードであるスタンバイ・モードに移行します。UART 通信設定と、どの GPIO が使用されているかを確認する必要があります。最後に、特定の文字を監視するアプリケーション層を追加する必要があります。

ステップ 4c:サンプルをインポートして変更する

類似の例が見つかったら、CCS を開き、「**Project > Import CCS Project...**」の順に選択してサンプル・コードをインポートし、MSPM0 SDK のサンプル・フォルダに移動します。サンプルをインポートします。これは、インポートされた uart_echo_interrupts_stanby の例です。これは SysConfig プロジェクトなので、メインの C ファイルはシンプルです。最初に SysConfig driverlib 初期化を呼び出します。これは、SysConfig によって自動生成される機能であり、デバイスを構成します。その後、UART 割り込みをイネーブルにします。最後に、UART トランザクションを待ちながら、スリープ状態に移行します。UART トランザクションを受信すると、データをすぐにエコーし、ウェークアップします。

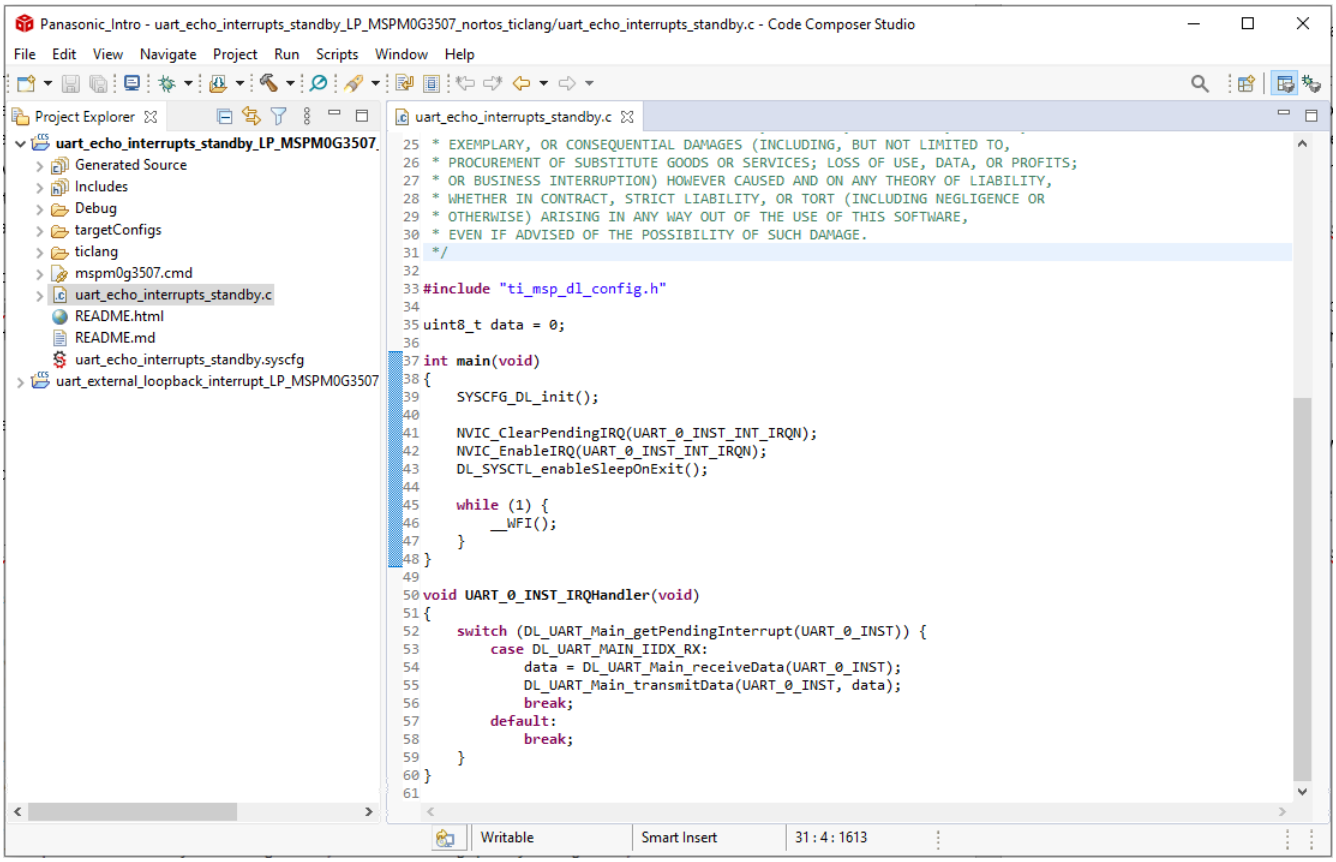

図 2-6. uart_echo_interrupts_standby の例

SysConfig の構成を表示するには、.syscfg ファイルを開きます。このファイルはデフォルトで SYSCTL タブで開きます。SysConfig の使用方法の詳細については、MSPM0 SDK にある『SysConfig ガイド』を参照してください。

最初に注意するのは、電力ポリシーです。この MSPM0 の例では、Standby0 モードを使用していますが、Stop0 モードを使用することを目標としています。ドロップダウン・リストをクリックすると、適切な低消費電力モードを選択できます。このタブではすべてのクロックと発振器を構成することもできますが、現時点では較正する必要はありません。

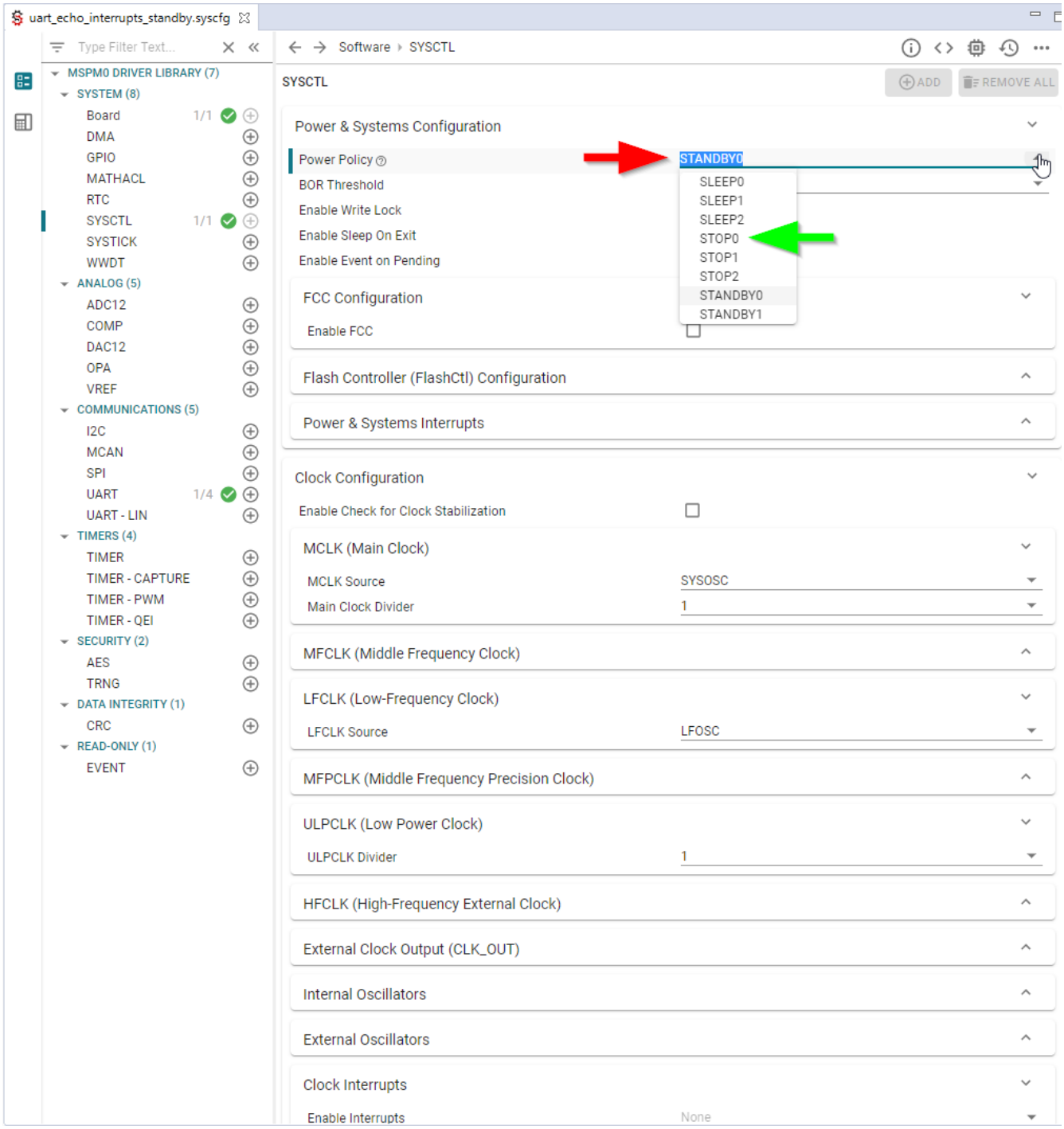

図 2-7. 電力モードの構成

次に、UART タブで UART 通信設定を確認します (図 2-8 を参照)。この場合、ボーレートはすでに 9600 に設定されており、他の通信設定も正確です。受信割り込みはすでにイネーブルになっており、メイン・プログラムで使用されます。また、右上にあるチップ・アイコンをクリックし、UART で強調表示されているピンをチェックして、使用中の UART モジュールとピンを確認します。MSPM0G3507 LaunchPad キットのバックチャネルの UART にすでに接続されているため、ここでは何も変更する必要はありません。

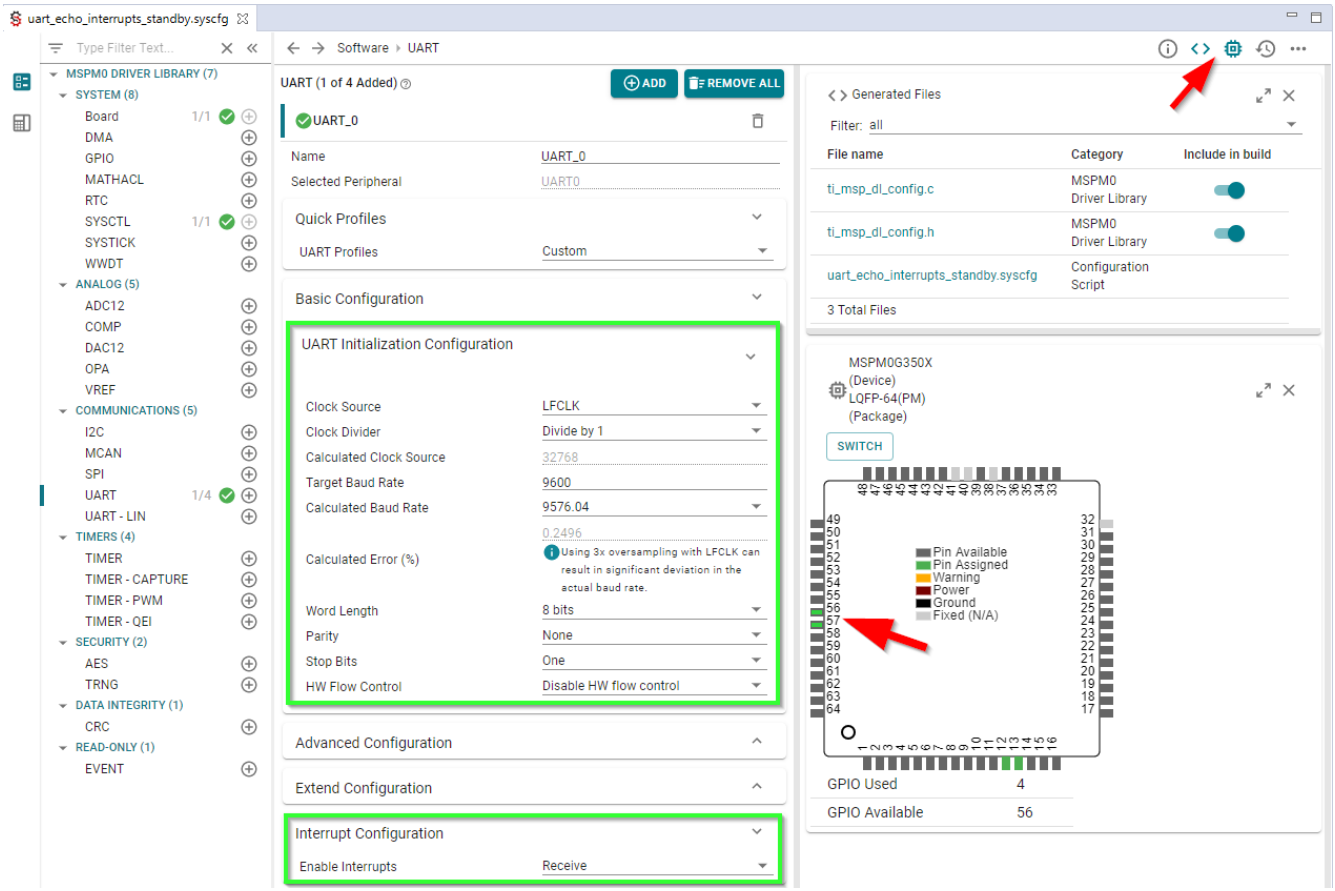

図 2-8. UART の構成

この例では現在、LED 駆動用に GPIO が設定されていませんが、設定は簡単に追加できます (図 2-9 を参照)。GPIO は、ページの上部にある **+ADD** ボタンを使用して追加できます。GPIO ポートとピンには、名前をつけることができ、この場合はそれぞれ「LED」と「RED」です。この GPIO は出力として設定され、ポート A のピン 0 (PA0) に配置されます。LaunchPad キットでは、この GPIO はシンプルな赤い LED に接続されています。

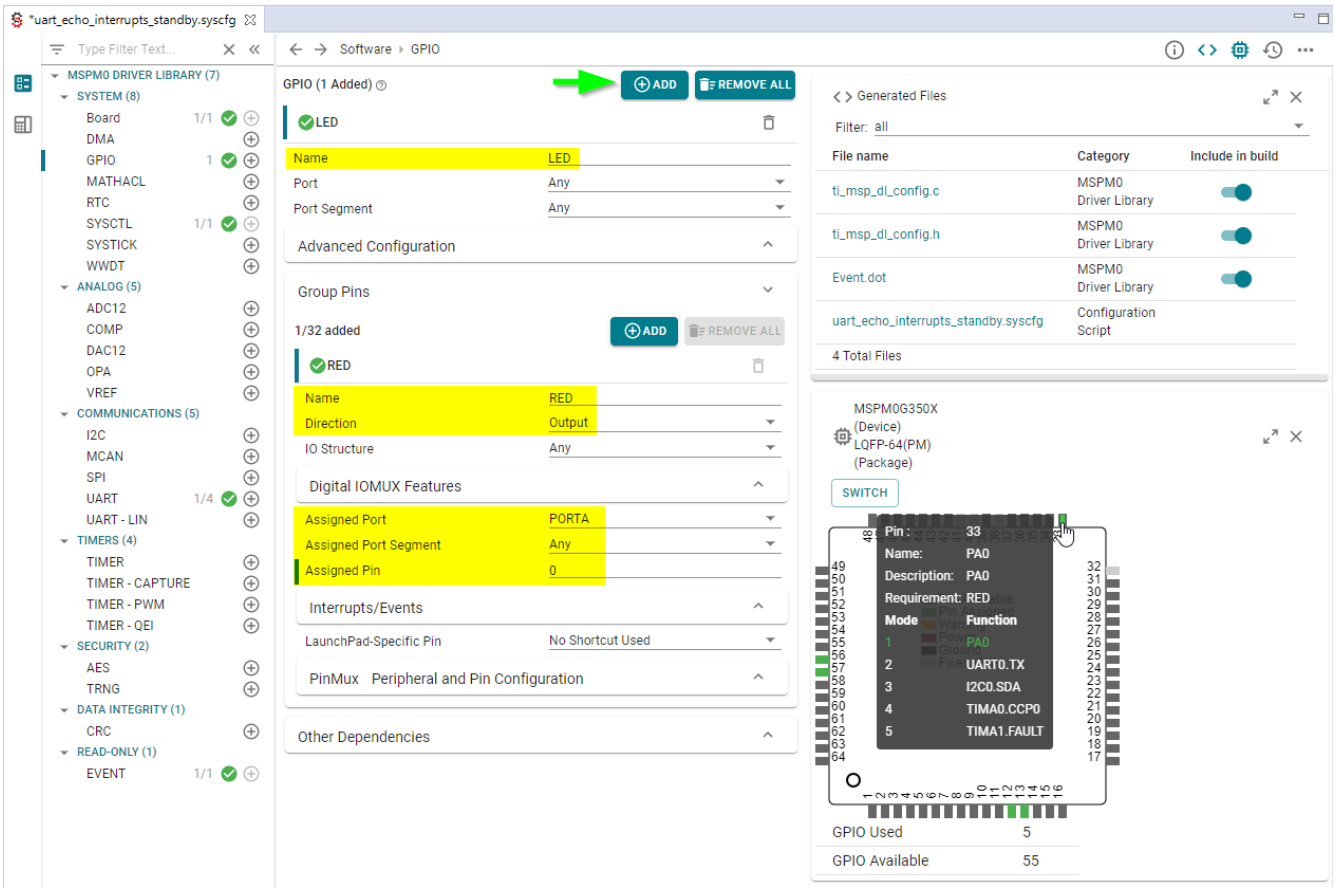

図 2-9. GPIO の構成

プロジェクトを保存して再ビルドすると、SysConfig によって、たとえば ti_msp_dl_config.c および ti_msp_dl_config.h ファイルが更新されます。この時点で、サンプル・ハードウェア構成が変更され、移植対象の元のソフトウェアのすべての機能が一致するようになりました。残りの作業は、受信した UART バイトをチェックし、LED を切り替えるためのアプリケーション・レベルのソフトウェアのみです。これは、少量のコードをメインの C ファイルに移動することで実現されます。

*uart_echo_interrupts_standby.c uart_echo_interrupts_standby.syscfg

```
*/

#include "ti_msp_dl_config.h"

uint8_t data = 0;

int main(void)
{
    SYSCFG_DL_init();

    DL_GPIO_clearPins(LED_PORT, LED_RED_PIN);

    NVIC_ClearPendingIRQ(UART_0_INST_INT_IRQN);
    NVIC_EnableIRQ(UART_0_INST_INT_IRQN);
//    DL_SYSCTL_enableSleepOnExit();
    DL_SYSCTL_disableSleepOnExit();

    while (1) {
        __WFI();

        if((data == 'S')||( data == 's')){
            DL_GPIO_setPins(LED_PORT, LED_RED_PIN);

        }else{
            DL_GPIO_clearPins(LED_PORT, LED_RED_PIN);
        }
    }
}

void UART_0_INST_IRQHandler(void)
{
    switch (DL_UART_Main_getPendingInterrupt(UART_0_INST)) {
        case DL_UART_MAIN_IIDX_RX:
            data = DL_UART_Main_receiveData(UART_0_INST);
            DL_UART_Main_transmitData(UART_0_INST, data);
            break;
        default:
            break;
    }
}
```

図 2-10. アプリケーション・コードの変更

アプリケーション・コードに 2 つの変更が加えられています。まず、DL_SYSCTL_disablesSleepOnExit() を使用して、MSPM0 が各 UART RX によって短時間でウェークアップするようにします。次に、UART RX データの簡単なチェックが追加され、「S」または「s」を受信すると、赤い LED が点灯します。それ以外の場合はオフになります。

ステップ 5：デバッグと検証

以下の図は、9600 ボーでの UART 通信と、赤の LED が正しくオン・オフされていることを示すロジック・アナライザからのキャプチャです。このコードはすべての UART 文字をエコーしていますが、正しい文字を受信したときのみ LED をオンにします。

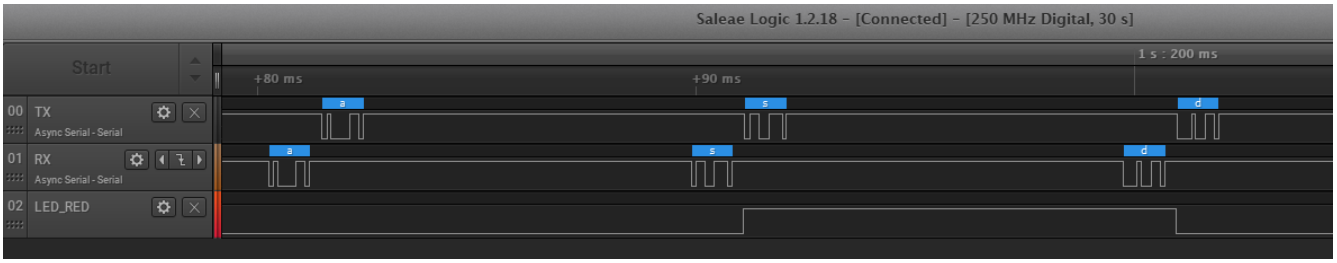

図 2-11.

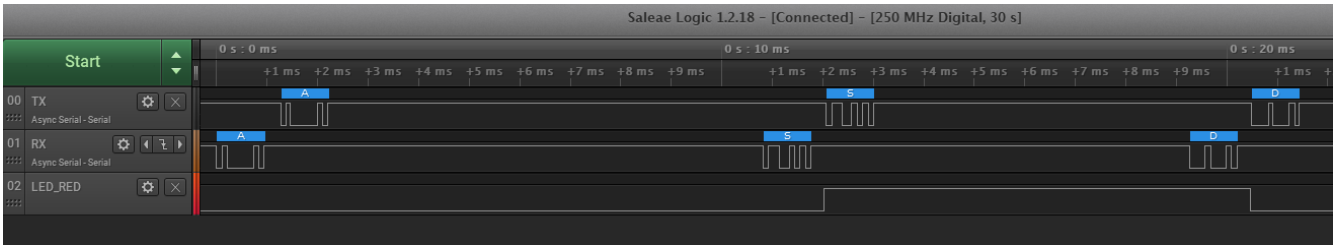

図 2-12.

ソフトウェアは正常に移植されました。これが多くのデバイスの最初のペリフェラルである場合は、このプロセスを繰り返し、SysConfig を使用して各ブロックを結合します。

3 コア・アーキテクチャの比較

3.1 CPU

STM32G0 および MSPM0 ファミリの部品はどちらも、Arm Cortex® M0+ CPU コア・アーキテクチャと命令セットをベースにしています。以下の表に、MSPM0G および MSPM0L ファミリの CPU の一般的な機能と STM32G0 の比較に関する大まかな概要を示します。割り込みと例外では、割り込みと例外の比較、および各デバイスの M0 アーキテクチャに含まれているネスト型ベクタ割り込みコントローラ (NVIC) ペリフェラルでそれらをどのようにマップするかを示します。

表 3-1. CPU 機能セットの比較

機能	STM32G0	MSPM0G	MSPM0L
アーキテクチャ	ARM Cortex-M0+	ARM Cortex-M0+	ARM Cortex-M0+
最大 MCLK	64MHz	80MHz	32MHz
CPU 命令キャッシュ	2 つの x64 ビット・ライン (16 バイト)	4 つの x64 ビット・ライン (32 バイト)	2 つの x64 ビット・ライン (16 バイト)
プロセッサのトレース機能	なし	あり、マイクロ・トレース・バッファを内蔵	なし
メモリ保護ユニット (MPU)	あり	あり	なし
システム・タイマ (SYSTICK)	あり	あり、24 ビット	あり、24 ビット
NVM の事前読み取り	あり	あり	あり
ハードウェアの多重化	あり	あり	あり
ハードウェア・ブレークポイント / ウォッチポイント	4/2	4/2	4/2
ブート・ルーチンの保存	フラッシュ (システム・メモリ)	ROM	ROM
ブートストラップ・ローダのストレージ	フラッシュ (システム・メモリ)	ROM	ROM
ブートローダ・インターフェイスのサポート(1) (2)	UART、I2C、SPI、USB、FDCAN	UART、I2C、ユーザー拡張可能	UART、I2C、ユーザー拡張可能
DMA	あり	あり	あり

(1) 利用可能かどうかについては、デバイス固有のデータシートを参照してください。
(2) その他のインターフェイスは、今後のデバイス・リリースで利用可能になります。

3.2 組み込みメモリの比較

3.2.1 フラッシュの特長

MSPM0 および STM32G0 ファミリの MCU には、実行可能なプログラム・コードとアプリケーション・データの保存に使用される不揮発性フラッシュ・メモリが搭載されています。

表 3-2. フラッシュ機能の比較

特長	STM32G0	MSPM0
フラッシュ・メモリ	STM32G0B1xx、G0C1xx (最大 512KB) STM32G071xx、G081xx (最大 128KB) STM32G031xx、G041xx、G051xx、G061xx (最大 64KB)	MSPM0Gx x の範囲は 128KB～32KB です MSPM0Lxx の範囲は 64KB～8KB です
メモリ構成	1 バンク – 最大 128KB のデバイス 2 バンク – 128KB を超えるデバイス	1 バンク – 最大 256KB のデバイス 2 バンク – 256KB を超えるデバイス
フラッシュのウェイト状態	0 (HCLK ≤ 24MHz) 1 (HCLK ≤ 48MHz) 2 (HCLK ≤ 64MHz)	0 (MCLK、CPUCLK ≤ 24MHz) 1 (MCLK、CPUCLK ≤ 48MHz) 2 (MCLK、CPUCLK ≤ 80MHz)
フラッシュ・ワード・サイズ	64 ビット + 8 個の ECC ビット	同一
プログラミング分解能	シングル・ワード・サイズ	シングル・ワード、32、16、または 8 ビット (バイト)
マルチワード・プログラミング	32 ワード (256 バイト)	2、4、8 ワード (最大 64 バイト)

表 3-2. フラッシュ機能の比較 (continued)

特長	STM32G0	MSPM0
消去	ページ・サイズ = 2KB バンク消去 (シングル・バンク) 大量消去 (すべてのバンク)	セクター・サイズ = 1KB バンク消去 (最大 256KB)
書き込み保護	あり (バンクごとに 2 つの書き込み保護領域)	あり、静的と動的
読み取り保護	あり	あり
フラッシュ・メモリの読み取り動作	64 ビットのフラッシュ・ワード・サイズと 8 個の ECC ビット	同一 - オプションの ECC が存在する場合
フラッシュ・メモリの書き込み動作	64 ビットのフラッシュ・ワード・サイズと 8 個の ECC ビット	同一 - オプションの ECC が存在する場合
エラー・コード訂正 (ECC)	64 ビットの場合は 8 ビット	同一
保護可能なメモリ領域	あり、メイン・メモリ	なし
Info メモリ	あり	あり (NONMAIN)
OTP データ領域	1KB	なし
事前読み取り	あり	あり
CPU 命令キャッシュ	2 つの 64 ビット・キャッシュ・ライン (16 バイト) 4x 32 ビット命令群、または 8x 16 ビットの命令群	4 つの 64 ビット・キャッシュ・ライン (32 バイト) 8x 32 ビット命令群、または 16x 16 ビットの命令群

前の表に示したフラッシュ・メモリ機能に加えて、MSPM0 フラッシュ・メモリには以下の機能もあります。

- 電源電圧範囲全体にわたって、インサーキット・プログラムと消去がサポートされています
- 内部プログラミング電圧の生成
- EEPROM エミュレーションは、フラッシュ・メモリの下位 32KB で最大 100,000 回のプログラム / 消去サイクルをサポートし、残りのフラッシュ・メモリで最大 10,000 回のプログラム / 消去サイクルをサポート (32KB のデバイスはフラッシュ・メモリ全体で 100,000 サイクルをサポート)

3.2.2 フラッシュの構成

フラッシュ・メモリは、アプリケーション・コードとデータ、デバイスのブート構成、および工場出荷時にテキサス・インスツルメンツが事前にプログラムしたパラメータを保存するために使用されます。フラッシュ・メモリは 1 つ以上のバンクに編成され、各バンクのメモリはさらに 1 つ以上の論理メモリ領域にマップされ、アプリケーションで使用できるようにシステム・アドレス空間が割り当てられます。

メモリ・バンク

ほとんどの MSPM0 デバイスは、単一のフラッシュ・バンク (BANK0) を実装しています。単一のフラッシュ・バンクを持つデバイスでは、継続的なプログラム / 消去動作により、フラッシュ・メモリに対するすべての読み取り要求が停止し、動作が完了してフラッシュ・コントローラがバンクの制御を解放するまで保持されます。複数のフラッシュ・バンクを持つデバイスでも、バンク上のプログラム / 消去動作により、プログラム / 消去動作を実行しているバンクに対して発行された読み取り要求は停止されますが、他のバンクに対して発行された読み取り要求を停止することはありません。したがって、複数のバンクが存在するため、以下のようなアプリケーション・ケースが可能になります。

- デュアル・イメージ・ファームウェアの更新 (アプリケーションは、1 つのフラッシュ・バンクからコードを実行でき、アプリケーションの実行を停止せずに、もう 1 つのイメージをもう 1 つの対称型フラッシュ・バンクにプログラムすることが可能)
- EEPROM エミュレーション (アプリケーションは 1 つのフラッシュ・バンクからコードを実行でき、もう 1 つのフラッシュ・バンクを使用してアプリケーションの実行を停止せずにデータを書き込むことができます)

フラッシュ・メモリ領域

各バンク内のメモリは、各バンクのメモリがサポートする機能に基づいて、1 つまたは複数の論理領域にマップされます。4 つの領域があります。

- FACTORY - デバイス ID およびその他のパラメータ
- NONMAIN - デバイス・ブート構成 (BCR および BSL)
- MAIN - アプリケーションのコードとデータ
- DATA - データまたは EEPROM エミュレーション

1 つのバンクを持つデバイスは、FACTORY、NONMAIN、および MAIN 領域を BANK0 (存在する唯一のバンク) に実装しており、データ領域は利用できません。複数のバンクを持つデバイスは、FACTORY、NONMAIN、および MAIN 領域も BANK0 に実装していますが、MAIN 領域または DATA 領域を実装できる追加のバンク (BANK1 から BANK4) が含まれています。

NONMAIN メモリ

NONMAIN は、デバイスをブートするために BCR と BSL が使用する構成データを格納する、フラッシュ・メモリの専用領域です。この領域は、他の目的では使用されません。BCR と BSL にはどちらも構成ポリシーがあり、NONMAIN フラッシュ領域にプログラムされた値を変更することで、デフォルト値 (開発および評価時の標準値) をそのままにするか、特定の目的 (量産プログラミング時の標準値) に変更することができます。

3.2.3 内蔵 SRAM

MSPM0 および STM32G0 ファミリの MCU には、アプリケーション・データの保存に使用される SRAM が搭載されています。

表 3-3. SRAM 機能の比較

機能	STM32G0	MSPM0
SRAM メモリ	STM32G0B1xx、G0C1xx：144KB (SRAM パリティ有効時 128KB) STM32G071xx、G081xx：36KB (SRAM パリティ有効時 32KB) STM32G051xx、G061xx：18KB (SRAM パリティ有効時 16KB) STM32G031xx、G041xx：8KB (SRAM パリティ有効時 8KB) ゼロ待機状態	MSPM0Gxx：32KB～16KB MSPM0Lxx：4KB～2KB ゼロ待機状態 選択されたデバイスには、SRAM のパリティと ECC が含まれる。詳細についてはデバイス・データ・シートを参照
最大 CPU クロック周波数でゼロ待機状態	あり	あり
アクセス分解能	バイト、ハーフワード (16 ビット)、またはフルワード (32 ビット)	バイト、ハーフワード (16 ビット)、またはフルワード (32 ビット)
パリティ・チェック	あり	あり

MSPM0 MCU には、低消費電力の高性能 SRAM が搭載されており、デバイスでサポートされている CPU 周波数範囲全体にわたってゼロ・ウェイト・ステートに対応します。SRAM は、コードに加えて、呼び出しスタック、ヒープ、グローバル・データなどの揮発性情報を格納するために使用できます。SRAM の内容は、実行、スリープ、停止、スタンバイ動作モードでは完全に保持されますが、シャットダウン・モードでは失われます。書き込み保護メカニズムが搭載されているため、アプリケーションは 1KB の分解能で下位 32KB の SRAM を動的に書き込み保護できます。32KB 未満の SRAM を搭載したデバイスでは、SRAM 全体に対して書き込み保護が提供されます。書き込み保護は、実行可能コードを SRAM に配置するときに役立ちます。これは、CPU または DMA によってコードが意図せず上書きされることを防止するレベルの保護を提供するためです。SRAM にコードを配置すると、ゼロ待機状態動作と低消費電力を実現することで、重要なループの性能を向上できます。

3.3 電源投入とリセットの概要と比較

STM32G0 デバイスと同様に、MSPM0 デバイスには最小動作電圧があり、デバイスまたはデバイスの一部をリセット状態に保持することでデバイスを正しく起動させるためのモジュールが搭載されています。表 3-4 に、2 つのファミリ間でどのように行われ、どのモジュールが、ファミリ間で電源投入プロセスとリセットを制御するかを比較したものを示します。

表 3-4. 電源投入の比較

STM32G0 デバイス		MSPM0 デバイス	
電源投入とリセットを管理するモジュール	PWR (電源) および RCC (リセットおよびクロック制御) モジュール	電源投入とリセットを管理するモジュール	PMCU (パワー・マネージメントおよびクロック・ユニット)
電圧レベル・ベースのリセット			
POR (パワーオン・リセット)	デバイスのリセットを完了します。電源投入用の第 1 レベル電圧リリース。パワーダウン時の最小電圧レベル。	POR (パワーオン・リセット)	デバイスのリセットを完了します。電源投入用の第 1 レベル電圧リリース。パワーダウン時の最小電圧レベル。
レベルを設定可能な BOR (ブラウンアウト・リセット)	プログラマブルな場合もある。電源投入時にリセット状態を解放する電圧レベル、または電源オフ時にデバイスをリセットする電圧レベルを設定します。	BOR (ブラウンアウト・リセット) を構成可能	リセットまたは割り込みとして構成でき、異なる電圧スレッショルドを使用して、STM32G0 の BOR と PVD の機能を組み合わせることができます。
PVD (プログラマブル電圧検出器)	割り込みを提供できる構成可能な電圧モニタ。		

STM32G0 はさまざまなリセット・ドメインを定義しており、MSPM0 デバイスはさまざまなレベルのリセット状態を持っています。MSPM0 デバイスの場合、リセット・レベルには設定された順序があり、レベルがトリガされると、デバイスが RUN モードに解放されるまで、それ以降のすべてのレベルがリセットされます。表 3-5 に、STM32G0 リセット・ドメインと MSPM0 リセット状態の概要と比較を示します。図 3-1 に、MSPM0 のすべてのリセット状態の関係を示します。

表 3-5. リセット・ドメインの比較

STM32G0 のリセット・ドメイン		MSPM0 のリセット状態(1)	
電源リセット・ドメイン	代表的なトリガは POR、BOR で、スタンバイ・モードまたはシャットダウン・モードから終了します。VCORE ドメイン外のレジスタを除くすべてのレジスタがリセットされます。	POR	代表的なトリガ：POR 電圧レベル、SW トリガ、NRST が 1 秒を上回る間 LOW に保持される。シャットダウン・メモリをリセット、NRST と SWD を再度イネーブル、BOR をトリガ
		BOR	代表的なトリガ：POR または BOR の電圧レベル、シャットダウン・モードの終了。PMU、VCORE、および関連するロジックのリセット。BOOTRST のトリガ。
完全に同等ではありません。ブート構成は、リセット後に SYSCLK の 4 番目のクロック・サイクルで読み取られます。		ブート・リセット (BOOTRST)	代表的なトリガ：BOR またはソフトウェアのトリガ、致命的なクロック障害、NRST が 1 秒未満の間 low に保持される。ブート構成ルーチンの実行。RTC、クロック、IO 構成を含む、コア・ロジックとレジスタの大部分をリセットします。(2)SRAM の電源サイクルおよび損失。SYSRST をトリガ。
システム・リセット・ドメイン	システム・リセットでは、クロック制御およびステータス・レジスタ (RCC_CSR) のリセット・フラグと RTC ドメインのレジスタを除き、すべてのレジスタがリセット値に設定されます。	システム・リセット (SYSRST)	代表的なトリガ：BOOTRST、BSL の開始または終了、ウォッチドッグ・タイマ、ソフトウェアのトリガ、デバッグ・サブシステム。RTC、LFCLK、LFXT、SYSOSC 周波数補正ループを除くすべてのペリフェラルと CPU の状態をリセットします。デバイスは終了時に RUN モードに移行します。
同等ではありません		CPU のみのリセット (CPURST)	ソフトウェアおよびデバッグ・サブシステムのトリガのみ。CPU ロジックのみをリセットします。ペリフェラルの状態は影響を受けません。
RTC ドメイン	ソフトウェアによってトリガされます。または、両方の電源がオフになっていた場合、VDD または VBAT の電源オンによってトリガされます。LSE 発振器、RTC、バックアップ・レジスタ、および RCC RTC ドメイン制御レジスタのみをリセットします。	RTC および関連クロックは、BOOTRST、BOR、または POR によってリセットされます。(2)	

(1) すべてのリセット・トリガについて説明しているわけではありません。利用可能なすべてのリセット・トリガについては、デバイス TRM の PMCU の章を参照してください。
(2) BOOTRST の原因が NRST またはソフトウェア・トリガの場合、RTC、LFCLK、LFXT/LFCLK_IN 構成および IOMUX 設定はリセットされず、RTC は外部リセットにより動作を維持できます。

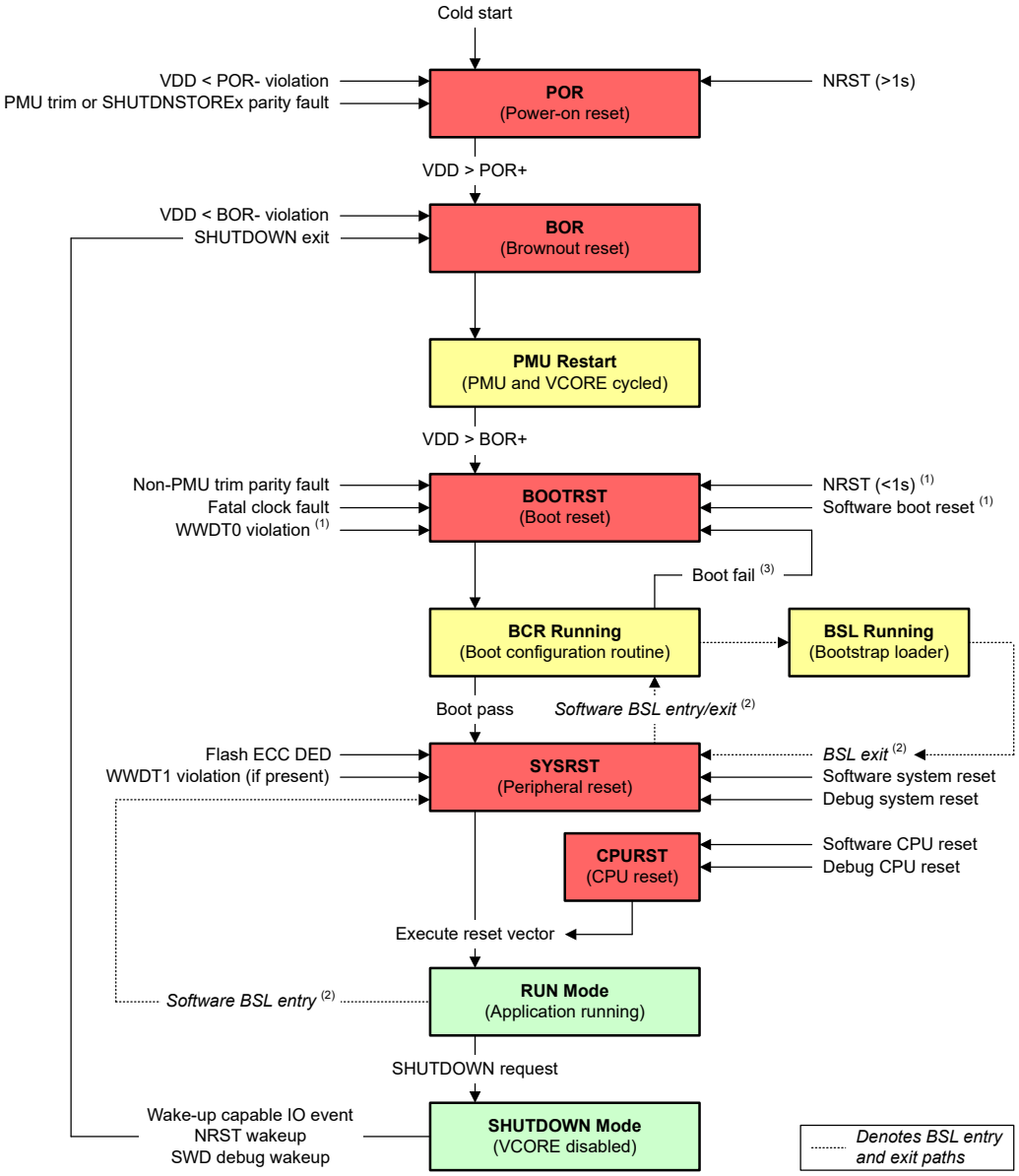

図 3-1. MSPM0 のリセット・レベル

3.4 クロックの概要と比較

STM32G および MSPM0 には、1 次クロックを供給する内部発振器が内蔵されています。クロックを分周して他のクロックを供給し、多数のペリフェラルに分配することができます。

表 3-6. 発振器の比較

STM32G0 発振器	MSPM0 発振器
HSI16RC 16MHz	SYSOSC(1)
HSI48RC 48MHz	SYSOSC
LSI RC 32kHz	LFOSC
HSE OSC 4～48MHz	HFXT
LSE OSC 32kHz	LFXT

表 3-6. 発振器の比較 (continued)

STM32G0 発振器	MSPM0 発振器
I2S_CLKIN	HFCLK_IN (デジタル・クロック)

(1) SYSOSC は 32MHz、24MHz、16MHz、4MHz にプログラム可能です。

表 3-7. クロックの比較

STM32G クロック	MSPM0 クロック
HSISYS	該当なし
PLLPCLK	SYSPLLCLK1
PLLQCLK	SYSPLLCLK1
PLLRCLK	SYSPLLCLK0
該当なし	SYSPLLCLK2x(1)
SYSCLK	BUSCLK(2)
HCLK	MCLK
HCLK8	CPUCLK
PCLK	BUSCLK
TIMPCLK	BUSCLK
LPTIMx_IN	LFCLK_IN

(1) SYSPLLCLK2x は PLL モジュールの出力速度の 2 倍で、分周できます。
(2) BUSCLK はパワー・ドメインに依存します。パワー・ドメイン 0 の場合、BUSCLK は ULPCLK です。パワー・ドメイン 1 の場合、BUSCLK は MCLK です。

表 3-8. ペリフェラル・クロック・ソース

ペリフェラル	STM32G クロック・ソース	MSPM0 クロック・ソース
RTC	LSI、LSE、HSE/32	LFCLK (LFOSC、LFXT)
UART	PCLK、LSE、HSI16、SYSCLK	BUSCLK、MFCLK、LFCLK
SPI	見つける必要がある	BUSCLK、MFCLK、LFCLK
I2C	PCLK、HSI16、SYSCLK	BUSCLK、MFCLK
ADC	HSI16、SYSCLK、PLLPCLK	ULPCLK、HFCLK、SYSOSC
CAN	PCLK、HSE、PLLQCLK	PLLCLK1、HFCLK
タイマ	PCLK、TIMPCLK、PLLQCLK	BUSCLK、MFCLK、LFCLK
LPTIM 1/2 (TIMG0/1)	PCLK、LSI、LSE、HSI16、LPTIMX_IN	LFCLK、ULPCLK、LFCLK_IN
RNG	HSI48、PLLQCLK、HSI16/8、SYSCLK	MCLK

各デバイス・ファミリの TRM には、クロック・システムの視覚化に役立つクロック・ツリーがあります。Sysconfig は、ペリフェラルのクロック分割とソーシングのオプションをサポートできます。

3.5 MSPM0 の動作モードの概要と比較

MSPM0 MCU には 5 つのメイン動作モード (電力モード) があり、アプリケーションの要件に基づいてデバイスの消費電力を最適化できます。消費電力を低減するためのモードは次のとおりです。RUN、SLEEP、STOP、STANDBY、SHUTDOWN。CPU は RUN モードではコードをアクティブに実行しています。ペリフェラル割り込みイベントにより、デバイスを SLEEP、STOP、または STANDBY モードから RUN モードにウェークアップできます。SHUTDOWN モードでは、内部コア・レギュレータが完全にディセーブルされ、消費電力が最小化されます。また、NRST、SWD、または特定の

IO での論理レベルの一致によってのみウェークアップが可能です。RUN、SLEEP、STOP、STANDBY の各モードには、複数の構成可能なポリシー・オプション (例:RUN.x) も含まれており、性能と消費電力のバランスを確保できます。

性能と消費電力のバランスをさらに高めるために、MSPM0 デバイスには次の 2 つの電力ドメインが実装されています。PD1 (CPU、メモリ、高性能ペリフェラル用) と PD0 (低速、低消費電力ペリフェラル用)。PD1 は、RUN モードと SLEEP モードで常に電源が供給されますが、他のすべてのモードではディスエーブルになります。PD0 は、RUN、SLEEP、STOP、STANDBY の各モードで常に電源が供給されます。SHUTDOWN モードでは、PD1 と PD0 の両方がディスエーブルになります。

動作モードの比較

STM32G0 デバイスは、同様の動作モードを備えています。次の表に、STM32G0 デバイスと MSPM0 デバイスの簡単な比較を示します。

表 3-9. STM32G0 デバイスと MSPM0 デバイスの動作モードの比較

STM32G0			MSPM0		
モード		概要	モード	概要	
実行		フル・クロックとペリフェラルを利用可能	実行	0	フル・クロックとペリフェラルを利用可能
LP RUN		2MHz に制限された CPU		1	SYSOSC は設定された周波数、CPUCLK および MCLK は 32kHz に制限
				2	SYSOSC はディスエーブル、CPUCLK および MCLK は 32kHz に制限
スリープ		CPU にクロックが供給されない	スリープ	0	CPU にクロックが供給されない
LP スリープ		LP RUN と同じだが、CPU にクロックが供給されない		1	Run1 と同じだが、CPU にクロックが供給されない
				2	Run2 と同じだが、CPU にクロックが供給されない
ストップ	0	VCORE ドメイン・クロックがディスエーブル	ストップ	0	スリープ 0 + PD1 はディスエーブル
				1	スリープ 1 + SYSOSC ギアが 4MHz にシフト
	1	ストップ 0 + メイン電源レギュレータをオフ		2	スリープ 2 + 32kHz に制限された ULPCLK
スタンバイ		BOR 機能付きで最小の消費電力、RTC 利用可能、レジスタ設定は消失。	スタンバイ	0	BOR 機能付きで最小の消費電力、すべての PD0 ペリフェラルは 32kHz で ULPCLK と LFCLK を受信、RTC は RTCCLK で使用可能
				1	32kHz で ULPCLK または LFCLK を受信できるのは、TIMG0 および TIMG1 のみ、RTC は RTCCLK で使用可能
シャットダウン		クロックまたは BOR なし。コア・レギュレーション・オフ。RTC ドメインは引き続きアクティブ可能。終了するとリセットがトリガされる。	シャットダウン	クロック、BOR、または RTC なし。コア・レギュレーション・オフ。PD1 および PD0 はディスエーブル。終了するとリセット・レベル BOR がトリガされる。	

低消費電力モードでの MSPM0 機能

表 3-9 に示すように、MSPM0 のペリフェラル・モードは、低消費電力動作モードにおいては、利用可能な機能または動作速度を制限できます。具体的な詳細については、MSPM0 デバイス固有のデータシートに掲載されている「動作モードでサポートされる機能」の表を参照してください。例:

MSPM0G350x ミックスド・シグナル・マイクロコントローラ・データシート

MSPM0L134x、MSPM0L130x ミックスド・シグナル・マイクロコントローラ・データシート

MSPM0 デバイスの追加機能は、一部のペリフェラルが非同期高速クロック要求を実行できることです。これにより、MSPM0 デバイスを、ペリフェラルがアクティブでない低消費電力モードに移行しつつ、ペリフェラルをトリガまたはアクティブにすることもできます。非同期高速クロック要求が発生した場合、MSPM0 デバイスは内部発振器を高速にすばやく立ち上げたり、一時的に高い動作モードに移行して差し迫った動作を処理したりすることができます。これにより、最小消費

電力モードでのスリープ中に、タイマ、コンパレータ、GPIO、および RTC からの CPU の高速ウェークアップ、SPI、UART、I2C の受信、または DMA 転送と ADC 変換のトリガを行うことができます。非同期クロック要求の実装とペリフェラルのサポートおよび目的の具体的な詳細については、MSPM0 TRM の該当する章を参照してください。

MSPM0 G シリーズ 80MHz マイクロコントローラ・テクニカル・リファレンス・マニュアル

MSPM0 L シリーズ 32MHz マイクロコントローラ・テクニカル・リファレンス・マニュアル

低消費電力モードへの移行

STM32G0 デバイスと同様に、MSPM0 デバイスは、イベントの待機、`__WFE();`、または割り込みの待機、`__WFI();`、の命令を実行するときに、低消費電力モードに移行します。低消費電力モードは、現在の電力ポリシー設定によって決定されます。デバイスの電力ポリシーは、ドライバ・ライブラリ関数によって設定されます。次の関数呼び出しは、電力ポリシーをスタンバイ 0 に設定します。

```
DL_SYSCTL_setPowerPolicySTANDBY0();
```

`STANDBY0` は、任意の動作モードに置き換えることができます。電源ポリシーを管理する driverlib API の全リストについては、『MSPM0 SDK API ガイド』の該当するセクションを参照してください。また、さまざまな動作モードの開始方法を示す以下のサンプル・コードも参照してください。すべての MSPM0 デバイスで、同様のサンプルを利用できます。

低消費電力モードのサンプル・コード

SDK のインストール先に移動し、低消費電力モードのサンプル・コードを「examples > nortos > LP name > driverlib」から検索します

3.6 割り込みとイベントの比較

割り込みと例外

MSPM0 と STM32G0 は両方とも、デバイスで利用可能なペリフェラルに応じて、割り込みベクトルと例外ベクトルを登録し、マップします。各デバイス・ファミリの割り込みベクトルの概要と比較を表 3-10 に示します。割り込みまたは例外の優先度の値が低いほど、優先度の高い値を持つ割り込みよりも優先度が高くなります。これらのベクトルの中には、優先順位をユーザーが選択できるものもあれば、固定されているものもあります。

MSPM0 および STM32G0 では、NMI、リセット、ハード・フォルト・ハンドラなどの例外に負の優先度の値が与えられ、常にペリフェラル割り込みよりも優先度が高いことを示します。割り込み優先度を選択可能なペリフェラルの場合、両方のデバイス・ファミリで最大 4 つのプログラム可能な優先レベルを使用できます。

表 3-10. 割り込みの比較

NVIC 番号	STM32G0		MSPM0x	
	割り込み / 例外	優先順位	割り込み / 例外	優先順位
-	リセット	固定:-3	リセット	固定:-3
-	NMI ハンドラ	固定:-2	NMI ハンドラ	固定:-2
-	ハード・フォルト・ハンドラ	固定:-1	ハード・フォルト・ハンドラ	固定:-1
-	SVCall ハンドラ	選択可能	SVCall ハンドラ	選択可能
-	PendSV	選択可能	PendSV	選択可能
-	SysTick	選択可能	SysTick	選択可能
0	ウィンドウ・ウォッチドッグ割り込み	選択可能	INT_GROUP0:WWDT0、DEBUGSS、FLASHCTL、WUC FSUBx、および SYSCTL	選択可能
1	電源電圧検出割り込み	選択可能	INT_GROUP1:GPIO0 と COMP0	選択可能
2	RTC およびタイムスタンプ	選択可能	タイマ G1 (TIMG1)	選択可能
3	Flash グローバル割り込み	選択可能	UART3(1)	選択可能
4	RCC グローバル割り込み	選択可能	ADC0	選択可能
5	EXTI0 および EXTI1 割り込み	選択可能	ADC1(1)	選択可能

表 3-10. 割り込みの比較 (continued)

NVIC 番号	STM32G0		MSPM0x	
	割り込み / 例外	優先順位	割り込み / 例外	優先順位
6	EXTI2 および EXTI3 割り込み	選択可能	CANFD0(1)	選択可能
7	EXTI4-EXTI15 割り込み	選択可能	DAC0(1)	選択可能
8	UCPD1/UCPD2/USB	選択可能	予約済み	選択可能
9	DMA1 チャネル 1	選択可能	SPI0	選択可能
10	DMA1 チャネル 2 および 3	選択可能	SPI1(1)	選択可能
11	DMA1 チャネル 4-6、DMA2 チャネル 1-5	選択可能	予約済み	選択可能
12	ADC とコンパレータ	選択可能	予約済み	選択可能
13	タイマ 1 (TIM1)、ブレーク、更新、トリガ、整流	選択可能	UART1	選択可能
14	TIM1 キャプチャの比較	選択可能	UART2(1)	選択可能
15	TIM2 グローバル割り込み	選択可能	UART0	選択可能
16	TIM3 および TIM4 グローバル割り込み	選択可能	TIMG0	選択可能
17	TIM6、LPTIM1、DAC 割り込み	選択可能	TIMG10(1)	選択可能
18	TIM6 および LPTIM2 グローバル割り込み	選択可能	TIMA0(1)	選択可能
19	TIM14 グローバル割り込み	選択可能	TIMA1	選択可能
20	TIM15 グローバル割り込み	選択可能	TIMA2(2)	選択可能
21	TIM16 および FDCAN0 グローバル割り込み	選択可能	TIMH0(1)	選択可能
22	TIM17 および FDCAN1 グローバル割り込み	選択可能	予約済み	選択可能
23	12C1 グローバル割り込み	選択可能	予約済み	選択可能
24	I2C2 および I2C3 グローバル割り込み	選択可能	I2C0	選択可能
25	SPI1 グローバル割り込み	選択可能	I2C1	選択可能
26	SPI2 および SPI3 グローバル割り込み	選択可能	予約済み	選択可能
27	USART1 グローバル割り込み	選択可能	予約済み	選択可能
28	USART2 および LPUART2 グローバル割り込み	選択可能	AES(1)	選択可能
29	USART 3-6 および LPUART1 グローバル割り込み	選択可能	予約済み	選択可能
30	CEC グローバル割り込み	選択可能	RTC(1)	選択可能
31	AES および RNG グローバル割り込み	選択可能	DMA	選択可能

(1) MSPM0G ファミリのデバイスでのみ利用できます。
(2) MSPM0L デバイス・ファミリ上の TIMG4

イベント・ハンドラと EXTI (拡張割り込みおよびイベント・コントローラ)

MSPM0 デバイスには専用のイベント・マネージャ・ペリフェラルが搭載されており、NVIC の概念を拡張して、ペリフェラルからのデジタル・イベントを割り込みとして CPU に転送したり、トリガとして DMA に転送したり、ハードウェア・アクションをトリガするために他のペリフェラルに転送したりできます。また、イベント・マネージャはパワー・マネージメントおよびクロック・ユニット (PMCU) とのハンドシェイクを実行し、トリガされたイベント・アクションを実行するために必要なクロックと電力ドメインが存在することを確認することもできます。

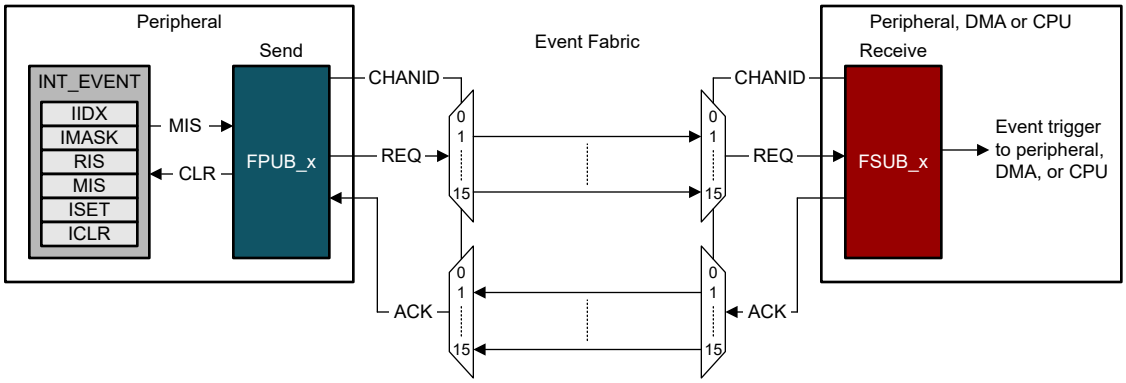

図 3-2. 汎用イベント・ルート

MSPM0 イベント・マネージャでは、イベントを生成するペリフェラルをパブリッシャーと呼び、パブリッシャーに基づいて動作するペリフェラル、DMA、CPU を加入者と呼びます。利用可能なパブリッシャーと加入者の潜在的な組み合わせは非常に柔軟で、ソフトウェアを移行するときに、以前は割り込みベクトルと CPU によって処理されていた機能を置き換えるために使用でき、CPU 全体をバイパスできます。たとえば、I^2C から UART へのブリッジは、以前は I^2C ストップを受信時に、ISR を使用してフラグを設定したとき、または UART TX バッファを直接ロードしたときに、UART 送信をトリガしていた可能性があります。MSPM0 イベント・ハンドラを使用すると、I^2C トランザクション完了イベントによって DMA がトリガして、UART TX バッファが直接ロードできるため、CPU によるアクションが不要になります。

MSPM0 でのイベント・ハンドラの使用方法の詳細については、『MSPM0G テクニカル・リファレンス・マニュアル』または『MSPM0L テクニカル・リファレンス・マニュアル』の「イベント」セクションを参照してください。

MSPM0 イベント・ハンドラと混同しないように、STM32G0 ファミリのデバイスには拡張割り込みおよびイベント・コントローラ (EXTI) が実装されており、IO またはペリフェラルからの構成可能なイベントにより、システムを STOP モードからウェイクアップさせることができます。STM32G0 EXTI のウェークアップ機能は、IO ウェークアップ機能 (『MSPM0 テクニカル・リファレンス・マニュアル』の「IOMUX」セクションを参照) と GPIO FastWake (『MSPM0 テクニカル・リファレンス・マニュアル』の「GPIO セクション」を参照) を使用して、MSPM0 で最適に複製できます。ウェークアップが単一アクションである場合、イベント・ハンドラ・ペリフェラルは、ペリフェラル動作の発生に必要な PMCU リソースを要求し、その後、適切な低消費電力モードに戻すことができます。

3.7 デバッグとプログラミングの比較

Arm SWD 2 線式 JTAG ポートは、MSPM0 と STM32G0 の両方のデバイスのメインのデバッグおよびプログラミング・インターフェイスです。このインターフェイスは通常、アプリケーション開発時および量産プログラミング時に使用されます。表 3-11 は、2 つのデバイス・ファミリの機能を比較したものです。MSPM0 デバッグ・インターフェイスのセキュリティ機能の詳細については、『MSPM0 MCU のサイバーセキュリティ・イネーブラ』アプリケーション・ノートを参照してください。

表 3-11. ARM SWD JTAG 機能の比較

	STM32G0	MSPM0
デバッグ・ポート	ARM SWD ポート (2 線式)	ARM SWD ポート (2 線式)
ブレーク・ポイント・ユニット (BPU)	4 つのハードウェア・ブレークポイント	4 つのハードウェア・ブレークポイント
データ・ウォッチ・ユニット (DWT)	2 つのウォッチポイント	2 つのウォッチポイント
マイクロ・トレース・バッファ (MTB)	なし	MTB は 4 つのトレース・パケットをサポートしています(1)
低消費電力デバッグのサポート	あり	あり
EnergyTrace のサポート	なし	EnergyTrace+ のサポート (電力プロファイリングによる CPU の状態)
デバッグ中のペリフェラルの実行サポート	あり	あり
デバッグ・インターフェイスのロック	デバッグの読み取りアクセスを一時的にブロックできます	デバッグ機能を永続的に無効にすることも、パスワードでロックすることもできます

(1) MSPM0Gxxxx デバイスのみ

ブートストラップ・ローダ (BSL) のプログラミング・オプション

ブートストラップ・ローダ (BSL) プログラミング・インターフェイスは、ARM SWD に対する代替プログラミング・インターフェイスです。このインターフェイスはプログラミング機能のみを提供し、通常は標準の組込み通信インターフェイスを通して使用します。これにより、システム内または外部ポート内の他の組み込みデバイスへの既存の接続を経由して、ファームウェアを更新できます。プログラミングの更新はこのインターフェイスの主な目的ですが、初期の量産プログラミングにも利用できます。表 3-12 に MSPM0 と STM32G0 の各デバイス・ファミリのさまざまなオプションと機能の比較を示します。

表 3-12. BSL 機能の比較

BSL の特長	STM32G0	MSPM0
BSL はブランク・デバイスで開始されました	あり	あり
プログラミング・インターフェイスの自動検出	あり	あり
セキュリティ	メモリのセキュリティとアクセス制限オプション	セキュア・ブート・オプション、CRC 保護
カスタム化可能	なし	あり、ピン起動とプラグイン機能を構成可能
メソッドを起動する	リセット時、SW エントリ時に、最大 2 本のピンとデバイス・レジスタ設定を使用するパターン(1)	BOOTRST、SW エントリで 1 ピン HIGH
サポートされているインターフェイス		
UART	あり	あり
I2C	あり	あり
SPI	あり(2)	カスタム・プラグインが必要
CAN	あり(2)	プラグインを計画中(2)
USB	あり(2)	現時点で USB 機能を搭載した MSPM0 デバイスはありません。

(1) パターン・オプションが利用可能かどうかは、デバイスによって異なります。
(2) 一部のデバイスのみ

4 デジタル・ペリフェラルの比較

4.1 汎用 I/O (GPIO、IOMUX)

MSPM0 GPIO 機能は、STM32G0 GPIO が提供するほぼすべての機能を網羅しています。STM32G0 で使用されている GPIO という用語は、デバイスのピンを管理するすべての機能を指しています。ただし、MSPM0 では、以下のように多少異なる命名規則が使用されます。

- MSPM0 GPIO とは、IO の読み取りと書き込み、割り込みの生成などを実行できるハードウェアのことです。
- MSPM0 IOMUX とは、さまざまな内部デジタル・ペリフェラルをピンに接続するために使用するハードウェアのことです。IOMUX は、GPIO など、多くの異なるデジタル・ペリフェラルにサービスを提供しますが、これらに限定されるものではありません。

MSPM0 GPIO と IOMUX を組み合わせることで、STM32G0 GPIO と同じ機能を実現できます。さらに、MSPM0 は、DMA 接続、制御可能な入力フィルタリング、イベント機能など、STM32G0 デバイスでは利用できない機能を提供します。

表 4-1. GPIO 機能の比較

機能	STM32G0	MSPM0G および MSPM0L
出力モード	プッシュプル プルアップまたはプルダウン付きのオープン・ドレイン	同等
GPIO 速度の選択	各 I/O の速度選択	類似 MSPM0 は、すべての IO ピンに標準 IO (SDIO) を提供する。SDIO は、STM GPIO の速度 = 01 と同等またはそれ以上。 MSPM0 高速 IO (HSIO) は、選択されたピンで利用可能。HSIO は、STM GPIO の速度 = 10 と同等。
High-drive GPIO	おおよそ 20mA	同等で、High Drive IO (HDIO) と呼ばれる
入力モード	フローティング プルアップまたはプルダウン アナログ	同等
アトミック・ビットのセットとリセット	あり	同等
GPIO ロック	レジスタ・ロック機構	*MSPM0 の同等はない*
代替機能	選択レジスタ	同等 MSPM0 は IOMUX を使用する
高速トグル	2 つのクロックごとに変更可能	同等、MSPM0 はクロック・サイクルごとにピンを切り替えることが可能
ウェークアップ	GPIO ピンの状態変更	同等
GPIO は DMA によって制御される	なし	*MSPM0 でのみ利用可能*
ユーザー制御の入力フィルタリングにより、1、3、8 ULPCLK 周期未満のグリッチを除去する	なし	*MSPM0 でのみ利用可能*
ユーザー制御可能な入力ヒステリシス	なし	*MSPM0 でのみ利用可能*

GPIO サンプル・コード

GPIO サンプル・コードの詳細については、『MSPM0 SDK サンプル・ガイド』を参照してください。

4.2 UART (Universal Asynchronous Receiver-Transmitter)

STM32G0 と MSPM0 はどちらも、非同期 (クロックレス) 通信を実行するためのペリフェラルを備えています。これらの UART ペリフェラルには、標準機能と高度な機能を備えた 2 つのバリエーションがあります。命名の違いは、表 4-2 に示されています。

表 4-2. STM32G0 と MSPM0 の UART 命名法の違い

	STM32G0 の命名法	MSPM0 の命名法
標準機能	基本	メイン
高度な機能	フル	拡張

表 4-3. UART の高度な機能セットの比較

機能	STM32G0 USART フル機能セット	MSPM0L および MSPM0G UART 拡張機能セット
ハードウェア・フロー制御	あり	あり
DMA を使用した連続通信	あり	あり
マルチプロセッサ	あり	あり
同期モード	あり	なし
スマート・カード・モード (ISO7816)	あり	あり
1 線式半二重通信	あり	あり(1)
IrDA ハードウェア・サポート	あり	あり
LIN ハードウェアのサポート	あり	あり
DALI ハードウェアのサポート	なし	あり
マンチェスター・コードのハードウェア・サポート	なし	あり
低消費電力モードからのウェークアップ	あり	あり
自動ボーレート検出	あり	なし
ドライバがイネーブル	あり	あり
データ長	7、8、9	5、6、7、8
Tx/Rx FIFO の深度	8	4

(1) 伝送と受信の間でペリフェラルを再構成する必要があります

表 4-4. UART 標準機能セットの比較

機能	STM32G0 USART の基本機能セット	MSPM0 UART メイン機能セット
ハードウェア・フロー制御	あり	あり
DMA を使用した連続通信	あり	あり
マルチプロセッサ	あり	あり
同期モード	あり	なし
1 線式半二重通信	あり	あり(1)
低消費電力モードからのウェークアップ	なし	あり
ドライバがイネーブル	あり	あり
データ長	7、8、9	5、6、7、8
Tx/Rx FIFO の深度	なし	4

(1) 伝送と受信の間でペリフェラルを再構成する必要があります

UART サンプル・コード

UART サンプル・コードの詳細については、『MSPM0 SDK サンプル・ガイド』を参照してください。

4.3 シリアル・ペリフェラル・インターフェイス (SPI)

MSPM0 と STM32G0 はどちらも、シリアル・ペリフェラル・インターフェイス (SPI) をサポートしています。全体として、MSPM0 および STM32G0 SPI のサポートは、表 4-5 に示す相違と同等です。

表 4-5. SPI 機能の比較

機能	STM32G0x	MSPM0L と MSPM0G
コントローラまたはペリフェラルの動作	あり	あり
データ・ビット幅 (コントローラ・モード)	4～16 ビット	4～16 ビット
データ・ビット幅 (ペリフェラル・モード)	4～16 ビット	7～16 ビット
最大速度	32MHz	MSPM0L：16MHz
		MSPM0G：32MHz
全二重通信	あり	あり
半二重通信 (双方向データ・ライン)	あり	なし
シンプレックス通信 (単方向データ・ライン)	あり	あり
マルチ・コントローラ機能	あり	なし
ハードウェア・チップ・セレクト・マネージメント	あり (1 つのペリフェラル)	あり (4 つのペリフェラル)
クロックの極性と位相をプログラム可能	あり	あり
MSB ファーストまたは LSB ファーストのシフトによるプログラマブル・データ順序	あり	あり
SPI フォーマットのサポート	Motorola、テキサス・インスツルメンツ	Motorola、テキサス・インスツルメンツ、MICROWIRE
ハードウェア CRC	あり	なし、MSPM0 は SPI パリティ・モードを備えています
TX FIFO の深さ	データ・サイズに依存	4
RX FIFO の深さ	データ・サイズに依存	4

SPI サンプル・コード

SPI サンプル・コードの詳細については、『MSPM0 SDK サンプル・ガイド』を参照してください。

4.4 I²C

MSPM0 と STM32G0 はどちらも I²C をサポートしています。MSPM0 および STM32G0 I²C の全体的なサポートは、次の表に概要を示した大きな違いと同等です。

表 4-6. I²C 機能の比較

機能	STM32G0	MSPM0L と MSPM0G
コントローラ・モードとターゲット・モード	あり	あり
マルチ・コントローラの機能	あり	あり
標準モード (最大 100kHz)	あり	あり
高速モード (最大 400kHz)	あり	あり
高速モード・プラス (最大 1MHz)	あり	あり
アドレッシング・モード	7、10 ビット	7 ビット
ペリフェラル・アドレス	2 つのアドレスと 1 つの構成可能マスク	2 つのアドレス
ゼネラル・コール	あり	あり
プログラム可能なセットアップ時間とホールド時間	あり	なし
イベント管理	あり	あり
クロック・ストレッチ	あり	あり
ソフトウェア・リセット	あり	あり

表 4-6. I²C 機能の比較 (continued)

機能	STM32G0	MSPM0L と MSPM0G
FIFO / バッファ	1 バイト	TX:8 バイト
		RX:8 バイト
DMA	あり	あり
プログラム可能なアナログおよびデジタル・ノイズ・フィルタ	あり	あり

I²C サンプル・コード

I2C サンプル・コードの詳細については、『MSPM0 SDK サンプル・ガイド』を参照してください。

4.5 タイマ (TIMGx、TIMAx)

STM32G0 と MSPM0 はどちらも、さまざまなタイマを搭載しています。MSPM0 は低消費電力監視から高度なモーター制御までのユースケースをサポートする、さまざまな機能を備えたタイマを提供しています。

表 4-7. タイマの命名

STM32G0		MSPM0	
タイマ名	**略称**	**タイマ名**	**略称**
高度な制御	TIM1	高度な制御	TIMA0
汎用	TIM2-4、TIM14/-17	汎用	TIMG0-11
		高解像度	TIMG12
基本	TIM6/7		
低消費電力	LPTIM		

表 4-8. タイマ機能の比較

機能	STM32G0 タイマ	MSPM0G タイマ	MSPM0L タイマ
分解能	16 ビット、32 ビット	16 ビット、32 ビット	16 ビット
PWM	あり	あり	あり
キャプチャ	あり	あり	あり
比較	あり	あり	あり
ワンショット	あり	あり	あり
アップ・ダウン・カウント機能	あり	あり	あり
電力モード	あり	あり	あり
QEI サポート	あり	あり	なし
プログラマブル・プリスケーラ	あり	あり	あり
シャドウ・レジスタ・モード	あり	あり	あり
イベント / 割り込み	あり	あり	あり
フォルト・イベント・メカニズム	あり	あり	なし
自動リロード機能	あり	あり	あり

表 4-9. タイマ・モジュールの代替品

STM32G0 タイマ	MSPM0 同等品	推論
TIM1	TIMA、TIMG8-12	高度な制御、共に 16 ビット分解能、QEI サポート
TIM2	TIMG12	32 ビット分解能
TIM3/4	TIMG0-7	汎用、16 ビット分解能
TIM6/7	任意	ベーシック・タイマ
TIM14	任意	TIM3/4 と同じ機能
TIM15/16/17	任意	汎用

表 4-9. タイマ・モジュールの代替品 (continued)

STM32G0 タイマ	MSPM0 同等品	推論
LPTIM	PD0 内の任意のタイマ	LPTIM は、LFCLK、PD0 (MSPM0 の低消費電力モード) に電力を供給

表 4-10. タイマの使用事例の比較

機能	STM32G0 タイマ	MSPM0 タイマ
PWM	TIM1-4 にはエッジ・アラインおよびセンター・アラインのオプションがあり、TIM6-7 には PWM 機能がない。TIM15-17 エッジ・アラインのみのオプション。	すべてのタイマには、エッジ・アラインまたはセンター・アラインのオプションがある
キャプチャ	大きな違いはない	大きな違いはない
比較	大きな違いはない	大きな違いはない
ワンショット	大きな違いはない	大きな違いはない
プリスケーラ	16 ビット・プリスケーラ、LPTIM (3 ビット・プリスケーラ) 以外	8 ビット・プリスケーラ
同期	TIM1-4、TIM15	すべてのタイマにこの機能がある

タイマ・サンプル・コード

タイマ・サンプル・コードの詳細については、『MSPM0 SDK サンプル・ガイド』を参照してください。

4.6 ウィンドウ付きウォッチドッグ・タイマ (WWDT)

STM32G0 と MSPM0 はどちらもウィンドウ・ウォッチドッグ・タイマを備えています。ウィンドウ・ウォッチドッグ・タイマ (WWDT) は、指定された時間内にアプリケーションがチェックインに失敗した場合、システム・リセットを開始します。

表 4-11. WWDT の命名法

キー	STM32G0	MSPM0
名称	独立したウォッチドッグ・タイマ、ウィンドウ付きウォッチドッグ・タイマ	ウィンドウ付きウォッチドッグ・タイマ
省略名 (同じ順序)	IWDG、WWDG	WWDT

表 4-12. WDT 機能の比較

機能	STM32G0	MSPM0G	MSPM0L
ウィンドウ・モード	あり	あり	あり
インターバル・タイマ・モード	あり	あり	あり
LFCLK ソース	あり	あり	あり
割り込み	あり	あり	あり
カウンタの分解能	7 ビット	25 ビット	25 ビット
クロック・デバイダ	WWDG なし、IWDG あり	あり	あり

WWDT サンプル・コード

WWDT サンプル・コードの詳細については、『MSPM0 SDK サンプル・ガイド』を参照してください。

4.7 リアルタイム・クロック (RTC)

STM32G0 および MSPM0 [1] は、どちらもリアルタイム・クロック (RTC) を提供しています。リアルタイム・クロック (RTC) モジュールは、選択可能な 2 進数または 2 進化 10 進数形式での、秒、分、時間、曜日、日、年のカウンタを使用することで、アプリケーションのタイム・トラッキング機能を提供します。

[1] RTC をサポートしているのは MSPM0G デバイスのみです。

表 4-13. RTC 機能の比較

機能	STM32G0	MSPM0G
電力モード	あり	あり
2 進数符号化形式	あり	あり
うるう年補正	あり	あり
カスタマイズ可能なアラームの数	2	2
内部および外部の水晶振動子	あり	あり
水晶振動子のオフセット較正	あり	あり
プリスケーラ・ブロック	あり	あり
割り込み	あり	あり

RTC サンプル・コード

RTC サンプル・コードの詳細については、『MSPM0 SDK サンプル・ガイド』を参照してください。

5 アナログ・ペリフェラルの比較

5.1 A/D コンバータ (ADC)

STM32G0 と MSPM0 はどちらも、アナログ信号を同等のデジタル信号に変換するための ADC ペリフェラルを搭載しています。どちらのデバイス・ファミリも、12 ビット ADC を搭載しています。以下の表は、ADC のさまざまな機能とモードを比較したものです。

表 5-1. 機能セットの比較

機能	STM32G0	MSPM0G	MSPM0L
分解能 (ビット)	12	12	12
変換レート (Msps)	2.5	4	1.4
オーバーサンプリング (ビット)	16	14	該当なし
ハードウェア・オーバーサンプリング	256x	128x	該当なし
FIFO	なし	あり	あり
ADC 基準電圧 (V)	内部:2.048、2.5	内部:1.4、2.5、VDD	内部:1.4、2.5、VDD
	V_{DD} < 2 のとき 外部:V_{REF} = V_{DD}	外部: 1.4 ≤ V_{REF} ≤ V_{DD}	外部: 1.4 ≤ V_{REF} ≤ V_{DD}
	V_{DD} ≥ 2 のとき 外部:2 ≤ V_{REF} ≤ V_{DD}		
動作時電力モード	動作、スリープ	動作、スリープ、停止、スタンバイ(1)	動作、スリープ、停止、スタンバイ(1)
自動パワーダウン	あり	あり	あり
外部入力チャネル(2)	最大 16	最大 16	最大 16
内部入力チャネル	温度センサ、VREF、VBAT	温度センサ、電源監視、アナログ・シグナル・チェーン	温度センサ、電源監視、アナログ・シグナル・チェーン
DMA サポート	あり	あり	あり
ADC ウィンドウ・コンパレータ・ユニット	なし	あり	あり
同時サンプリング	なし	あり	なし
ADC の数(3)	最大 1	最大 2	最大 1

(1) ADC はスタンバイ・モードでトリガできるため、動作モードが変化します。
(2) 外部入力チャネルの数は、デバイスごとに異なります。
(3) ADC の数はデバイスごとに異なります。

表 5-2. 変換モード

STM32G0	MSPM0	備考
シングル変換モード	シングル・チャネル・シングル変換	ADC は、1 つのチャネルを 1 回サンプリングして変換します
チャネルのシーケンスをスキャンします	チャネル変換のシーケンス	ADC は一連のチャネルをサンプリングし、1 回変換します。
連続変換モード	シングル・チャネル変換を繰り返します	1 つのチャネルを連続的にサンプリングし、1 つのチャネルを変換することを繰り返します
	チャネル変換のシーケンスを繰り返します	一連のチャネルをサンプリングして変換した後、同じシーケンスを繰り返します
不連続モード	チャネル変換のシーケンスを繰り返します	不連続な一連のチャネルのサンプリングと変換を行います。これは MSPM0 では、MEMCTRLx をさまざまなチャネルにマッピングすることで実行できます。

ADC サンプル・コード

ADC サンプル・コードの詳細については、MSPM0 SDK サンプル・ガイドを参照してください。

5.2 コンパレータ (COMP)

STM32G0 および MSPM0 ファミリの部品はどちらも、一部のデバイスでオプションのペリフェラルとしてコンパレータを内蔵しています。どちらのファミリのデバイスでも、これらは COMPx と呼ばれます。ここで、「x」の最後の文字は、検討対象のコンパレータ・モジュールを表します。STM32G0 ファミリでは、これらに 1～3 の番号が付けられ、MSPM0 ファミリでは 0～2 の番号が付けられています。コンパレータ・モジュールは、複数のコンパレータを備えたデバイスでウィンドウ付きコンパレータ機能を提供することも、さまざまな内部および外部ソースから入力を取得することも、パワー・モードの変更をトリガしたり、PWM 信号を切り詰めたり制御したりするために使用することもできます。MSPM0 および STM32G0 コンパレータ・モジュールの機能別の比較については、表 5-3 を参照してください。

表 5-3. COMP 機能セットの比較

機能	SMT32G0	MSPM0G	MSPM0L
使用可能なコンパレータ	最大 3	最大 3	最大 1
出力のルーティング	多重化された I/O ピン	多重化された I/O ピン	多重化された I/O ピン
	EXTI 割り込み	割り込み / イベント・インターフェイス	割り込み / イベント・インターフェイス
非反転入力ソース	多重化された I/O ピン	多重化された I/O ピン	多重化された I/O ピン
		DAC12 出力(1)	DAC8 出力
		DAC8 出力	OPA1 出力(2)
		内部 V_{REF}：1.4V、2.5V	
		OPA1 出力(2)	
反転入力ソース	多重化された I/O ピン	多重化された I/O ピン	多重化された I/O ピン
	DAC チャネル 1 および 2	内部温度センサ	内部温度センサ
	内部 V_{REF}：2.048V、2.5V	内部 V_{REF}：1.4V、2.5V	DAC8 出力
	バッファ付き V_{REF} デバイダ：1/4 V_{REF}、1/2 V_{REF}、および 3/4 V_{REF}	DAC8 出力	OPA0 (3)出力
		OPA0 出力(3)	
ヒステリシスをプログラム可能	なし、10mV、20mV、30mV	なし、10mV、20mV、30mV	なし、10mV、20mV、30mV
		DAC8 を使用するその他の値は、0V～V_{REF}/V_{DD} です	DAC8 を使用するその他の値は、0V～V_{DD} です
レジスタ・ロック	あり、すべての COMP レジスタ (デバイス・リセット時に無効)	あり、COMP レジスタの一部 (書き込みにはキーが必要)	あり、COMP レジスタの一部 (書き込みにはキーが必要)
ウィンドウ・コンパレータの構成	あり	あり	なし (シングル COMP)
入力短絡モード	なし	あり	あり
動作モード	高速、中速	高速、低消費電力	高速、低消費電力
PWM の高速シャットダウン	あり	あり (TIMA フォルト・ハンドラを使用)	なし
出力フィルタリング	ブランキング・フィルタ	ブランキング・フィルタ	ブランキング・フィルタ
		調整可能なアナログ・フィルタ	調整可能なアナログ・フィルタ
出力極性の制御	あり	あり	あり
割り込み	立ち上がりエッジ	立ち上がりエッジ	立ち上がりエッジ
	立ち下がりエッジ	立ち下がりエッジ	立ち下がりエッジ
	両エッジ	出力準備完了	出力準備完了
入力交換モード	なし	あり	あり

(1) DAC12 ペリフェラルを搭載したデバイスのみ
(2) OPA1 ペリフェラルを搭載したデバイスのみ
(3) OPA0 ペリフェラルを搭載したデバイスのみ

COMP サンプル・コード

COMP サンプル・コードの詳細については、MSPM0 SDK サンプル・ガイドを参照してください。

5.3 D/A コンバータ (DAC)

STM32G0 および MSPM0 ファミリの部品はどちらも 12 ビットの DAC ペリフェラルを搭載しており、さまざまなアプリケーションで D/A 変換を実行できます。STM32G0 の資料では、このペリフェラルは DAC と呼ばれています。『MSPM0 テクニカル・リファレンス・マニュアル』、『MSPM0 シリーズ・データシート』、および『MSPM0 SDK』では、12 ビット DAC ペリフェラルを DAC12 と呼びます。これにより、DAC12 は、特定の MSPM0 デバイスに搭載されている各コンパレータ・ペリフェラルで使用できる 8 ビット DAC と差別化します。追加の 8 ビット DAC については、このドキュメントのコンパレータのセクションで説明します。この DAC12 ペリフェラルは、MSPM0G ファミリのデバイスでのみ利用できます。

STM32G0 および MSPM0G の 12 ビット DAC ペリフェラルの機能を表 5-4 にまとめます。

表 5-4. DAC 機能セットの比較

機能	STM32G0	MSPM0
分解能	12 ビット (11.4～11.5ENOB)	12 ビット (11ENOB)
出力レート	1MSPS	1MSPS
出力チャネル数	2(1)	1(2)
データ形式	8 ビットの右揃え、12 ビットの右揃え、12 ビットの左揃え	8 ビットの右揃え、12 ビットの右揃え、2 の補数、またはストレート・バイナリ
DMA の統合	あり	あり
出力のルーティング	外部ピン	外部ピン
	内部ペリフェラル接続：COMP IN-、ADC	内部ペリフェラル接続：OPA IN+、COMP IN+、ADC0
内部基準電圧	あり、2.5V または 2.048V	あり、2.5V または 1.4V
外部基準電圧	あり	あり
FIFO	なし	あり
出力バッファ	あり	あり
出力オフセットの設定が可能	あり	あり
自己校正モード	あり	あり
自動波形生成	ノイズ波、三角波	なし
サンプル・アンド・ホールド・モード	あり	なし
トリガ・ソース	外部ピン、内部タイマ信号、DAC ホールド・クロック、DMA アンダーラン	内部専用のサンプル・タイム・ジェネレータ、DMA 割り込み / イベント、FIFO スレッショルド割り込み / イベント、2 つのハードウェア・トリガ (イベント・ファブリックから利用可能)

(1) 一部のデバイスでのみ使用できます。
(2) デュアル DAC チャネルは、将来の MSPM0G デバイス用に計画されています。

DAC12 サンプル・コード

DAC12 のサンプル・コードの詳細については、『MSPM0 SDK サンプル・ガイド』を参照してください。

5.4 オペアンプ (OPA)

STM32G0 ファミリのデバイスにはオペアンプ (OPA) ペリフェラルが内蔵されていませんが、STM32G0 から MSPM0 ファミリに移行するときは、MSPM0 内部 OPA を使用して外部のディスクリート・デバイスを置き換えるか、必要に応じて内部信号をバッファすることができます。MSPM0 OPA モジュールは非常に柔軟性が高く、センシングまたは制御アプリケーションでは、個別に、または組み合わせて、多くのディスクリート・アンプを置き換えることができます。MSPM0 OPA モジュールの主な機能を 表 5-5 に示し、再現可能な一般的な OPA 構成の例を OPA のサンプル・コードに示します。

表 5-5. MSPM0 OPA 機能セット

機能	MSPM0 の実装
入力タイプ	レール・ツー・レール (イネーブルまたはディスエーブル可能)
ゲイン帯域幅	1MHz (低消費電力モード)
	6MHz (標準モード)

表 5-5. MSPM0 OPA 機能セット (continued)

機能	MSPM0 の実装
アンプの構成	汎用モード
	バッファ・モード
	PGA モード (反転または非反転)
	差動アンプモード
	カスケード・アンプ・モード
入力 / 出力の配線	外部ピンの配線
	ADC および COMP モジュールへの内部接続
フォルト検出	バーンアウト電流源 (BCS)
チョッパ安定化	標準 (チョッピング周波数を選択可能)
	ADC 支援チョップ
	無効化
基準電圧	内部 VREF (MSPM0G デバイスのみ)
	DAC12 (MSPM0G デバイスのみ)
	DAC8 (COMP モジュール搭載デバイスのみ)

OPA サンプル・コード

OPA サンプル・コードの詳細については、『MSPM0 SDK サンプル・ガイド』を参照してください。

5.5 基準電圧 (VREF)

STM32G0x と MSPM0 はどちらも内部基準電圧を備えており、内部ペリフェラルに基準電圧を供給したり、外部ペリフェラルに出力したりするために使用できます。

表 5-6. 機能セットの比較

機能	STM32G0	MSPM0G	MSPM0L
内部基準電圧 (V)	2.048、2.5	1.4、2.5	1.4、2.5
外部基準電圧 (V)	V_{DD} < 2 の場合、$V_{REF} = V_{DD}$	外部：$1.4 \le V_{REF} \le V_{DD}$	外部：$1.4 \le V_{REF} \le V_{DD}$
	$V_{DD} \ge 2$ のとき、$2 \le V_{REF} \le V_{DD}$		
内部基準電圧の出力	あり	あり	あり
ADC への内部接続	あり	あり	あり
DAC への内部接続	あり	あり	なし
COMP への内部接続	なし	あり	なし
OPA への内部接続	該当なし	あり	なし

表 5-7. 制御ビットの比較

STM32G0x VREFBUF ビット	MSPM0 同等品
VREFBUF Bit3 (VRR)	CTL1 Bit0 (READY)
VREFBUF Bit2 (VRS)	CTL0 Bit7 (BUFCONFIG)
VREFBUF Bit1 (HIZ)	該当なし
VREFBUF Bit0 (ENVR)	CTL0 Bit0 (ENABLE)
	サンプル / ホールド・モードの場合：CTL0 Bit8 (SHMODE)

MSPM0 VREF の場合、パワー・ビット、PWREN Bit0 (ENABLE) をイネーブルにする必要があります。

VREF サンプル・コード

VREF を使用したサンプル・コードについては、『MSPM0 SDK サンプル・ガイド』を参照してください。

6 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	リビジョン	注
2023 年 3 月	A	最初の公開リリース

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、または ti.com やかかる TI 製品の関連資料などのいずれかを通じて提供する適用可能な条項の下で提供されています。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。

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

郵送先住所：Texas Instruments, Post Office Box 655303, Dallas, Texas 75265