AIROC™ CYW20829 MCU の低消費電力モードと節電技術

本書について

適用範囲と目的

AN238282 では、ユースケースに基づいて電力消費を最適化するための、AIROC™ CYW20829 Bluetooth® LE システムオンチップ (SoC) のさまざまな電源モードの使用方法について説明します。このアプリケーションノートで取り上げる主なトピックには、CYW20829 で利用可能な電力測定手順, 低消費電力モード, 節電のヒント、電力管理技術, およびサンプルアプリケーションが含まれます。

対象者

このアプリケーションノートは、それぞれの Bluetooth®アプリケーションの平均消費電力を最小限に抑えることに関心のあるユーザーを対象とします。

目次

本書について . . . 1
目次 . . . 2
1 はじめに . . . 3
1.1 インフィニオンのリソース . . . 4
1.1.1 ModusToolbox™ . . . 4
1.1.2 開発キット . . . 4
2 CYW20829 パワーアーキテクチャ . . . 5
2.1 低消費電力システム機能 . . . 5
2.2 クロック . . . 6
2.3 CYW20829 電力モード . . . 6
2.4 MCU および Bluetooth®サブシステムのシステム電力モード遷移 . . . 7
2.5 システムの電力モード . . . 9
2.5.1 Device Configurator でのシステム電力モードの選択 . . . 10
3 ソフトウェアによる電力管理 . . . 14
3.1 ブート . . . 14
3.1.1 コールド ブート . . . 14
3.1.2 ウォーム ブート . . . 14
3.2 システム電源管理 (SysPm) ライブラリ . . . 14
3.2.1 PDL . . . 14
3.2.1.1 コールバック モード . . . 15
3.2.2 HAL . . . 16
3.3 低消費電力モード遷移の実装 . . . 16
3.3.1 ベアメタル アプリケーションへの低消費電力モードの実装 . . . 16
3.3.1.1 ディープスリープ RAM モードに入る前に実行する手順 . . . 17
3.3.1.2 ディープスリープ RAM モードからウェイクアップした後に実行する手順 . . . 17
3.3.2 FreeRTOS での低消費電力モードの実装 . . . 18
3.3.2.1 ディープスリープ RAM に入る前に実行する手順 . . . 19
3.3.2.2 ディープスリープ RAM からウェイクアップした後に実行する手順 . . . 19
3.4 PDL を使用したコールバック関数の登録 . . . 19
3.5 PDL を使用したカスタム コールバック関数の実装 . . . 21
3.6 アクティブ節電のヒント . . . 23
3.6.1 CYW20829 を使用した電流パスのゲート . . . 23
3.6.2 GPIO コンフィギュレーション . . . 23
3.6.3 クロックゲーティング . . . 24
3.6.4 DMA を使用してデータをコピー/移動する . . . 24
3.6.5 オーディオ サブシステム . . . 24
3.6.6 周期的ウェイクアップ タイマー . . . 24
3.6.7 SRAM . . . 25

3.6.8 SCB . . . 26
3.6.9 TCPWM . . . 27
3.6.10 RTOS ティックレスモード . . . 27
4 高度な電力管理 . . . 29
4.1 電力制御アーキテクチャ . . . 29
4.1.1 各 PPU でサポートされる電力状態 . . . 29
4.2 レギュレータートポロジー . . . 30
4.3 コア電圧選択 . . . 30
4.3.1 リニアレギュレーターおよび降圧レギュレーター . . . 30
4.3.2 レギュレーターの割り当て . . . 31
4.3.3 レギュレーターの動作 . . . 31
4.4 電源ドメインのバックアップ . . . 31
4.5 バックアップ データ レジスタ . . . 32
5 低消費電力サンプルコード . . . 33
5.1 CE237951 – AIROC™ CYW20829 スイッチング電力モード . . . 33
5.2 CE238052 – AIROC™ CYW20829 FreeRTOS スイッチング電力モード . . . 33
5.3 CE237278 – HAL RTC アラームの定期的なウェイクアップ . . . 34
6 電力測定 . . . 35
6.1 ハードウェア . . . 35
6.2 平均消費電流 . . . 35
6.2.1 スリープ モード . . . 36
6.2.2 ディープ スリープ モード . . . 37
6.2.3 ディープスリープ RAM モード . . . 38
6.2.4 ハイバネートモード . . . 40
6.3 さまざまな電力モードでの測定電流読み出し . . . 40
参考資料 . . . 42
改訂履歴 . . . 43
商標 . . . 44
免責事項 . . . 45

1 はじめに

AIROC™ CYW20829 は、IoT アプリケーション向けに構築された、高性能, 超低消費電力, および「セキュア」MCU + Bluetooth® LE プラットフォームです。高性能マイクロコントローラと Bluetooth® LE (5.4) 接続, 高性能アナログ-デジタル変換オーディオ入力, I2S/PCM, CAN, LIN などの車載用ユースケース向け、その他の標準的な通信およびタイミングペリフェラルを組み合わせています。CYW20829 は、外部コンポーネントを最小限に抑えるために高度な統合を採用しており、デバイスのフットプリントと Bluetooth® Low Energy ソリューションの実装に関連するコストを削減します。AIROC™ CYW20829 は、リモコン, キーボード, ジョイスティック, Bluetooth®メッシュ, 自動車, アセットトラッキングなどの無線入力デバイスや、照明やホームオートメーションなど、10 dBm の RF 出力を必要とする

1 はじめに

Bluetooth® LE IoT アプリケーションに最適なソリューションです。AIROC™ CYW20829 Bluetooth® LE システム オンチップは、演算性能を大幅に損なうことなく、最高の省電力機能とモードを提供します。

このアプリケーションノートでは、AIROC™の CYW20829 アーキテクチャに関する基本的な知識と、ModusToolbox™ ソフトウェアを使用して AIROC™ CYW20829 アプリケーションを開発する能力が必要です。ModusToolbox™環境に慣れるには、ModusToolbox™ユーザー ガイドを参照してください。

1.1 インフィニオンのリソース

インフィニオンは、www.infineon.com に大量のデータを掲載しており、ユーザがデザインに対して適切な IoT デバイスを選択し、迅速かつ効率的にデバイスをデザインに統合する手助けをします。インフィニオンは、技術ドキュメント, 回路図, 製品の部品表, PCB レイアウト情報, ソフトウェアアップデートなど、幅広い情報へのアクセスをお客様に提供します。お客様は、インフィニオン開発者コミュニティから技術ドキュメントとソフトウェアを入手できます。

1.1.1 ModusToolbox™

インフィニオンは、CYW208xx アプリケーションのソフトウェア開発プラットフォームとして ModusToolbox™を提供しています。ModusToolbox™は、インフィニオンデバイスを既存の開発方法に統合できるようにするツールのセットです。ツールの 1 つは、アプリケーションの設定と開発をサポートするマルチプラットフォームの Eclipse IDE です。**Help** > **Eclipse IDE for ModusToolbox™ Documentation** > **User Guide** を参照してください。

1.1.2 開発キット

CYW920829M2EVK-02 開発キットは、AIROC™ CYW20829 Bluetooth® LE デバイスを使用した Bluetooth®アプリケーションの評価を可能にします。キットおよび CYW20829 の詳細 (ユーザー ガイドや回路図を含む) については、AIROC™ CYW20829 製品情報を参照してください。

2 CYW20829 パワーアーキテクチャ

2.1 低消費電力システム機能

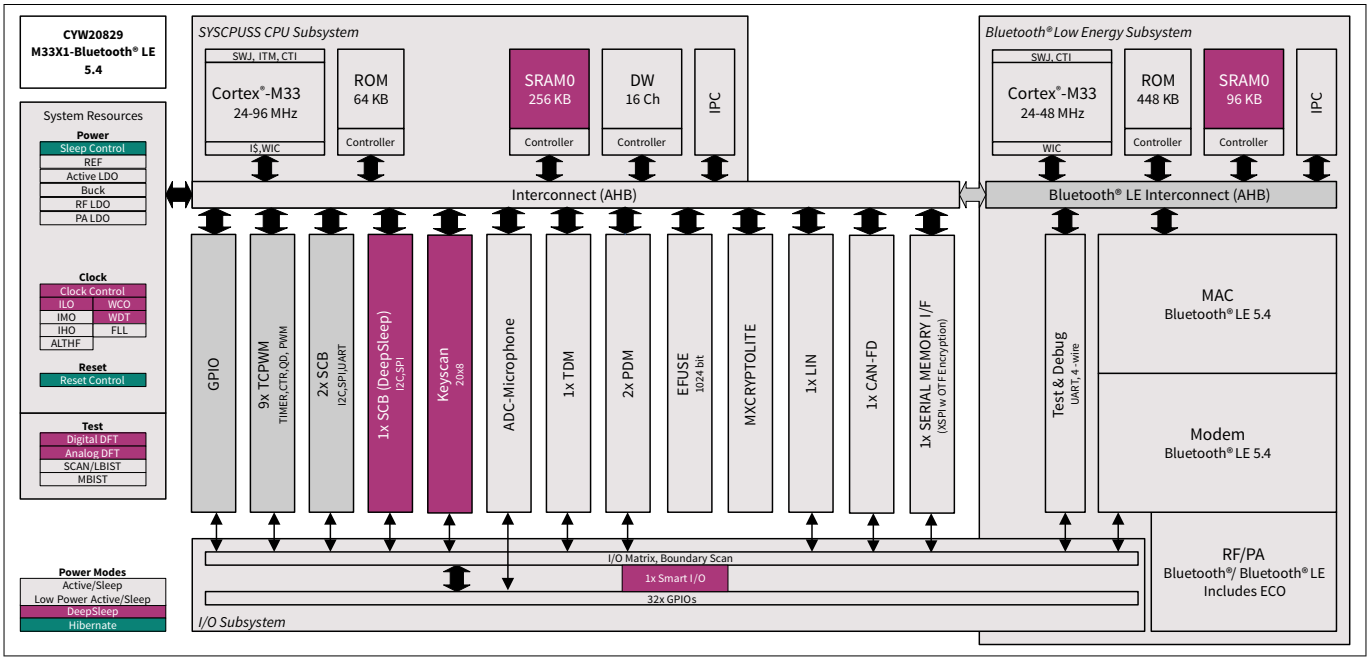

図 1 CYW20829 アーキテクチャ

2.2 クロック

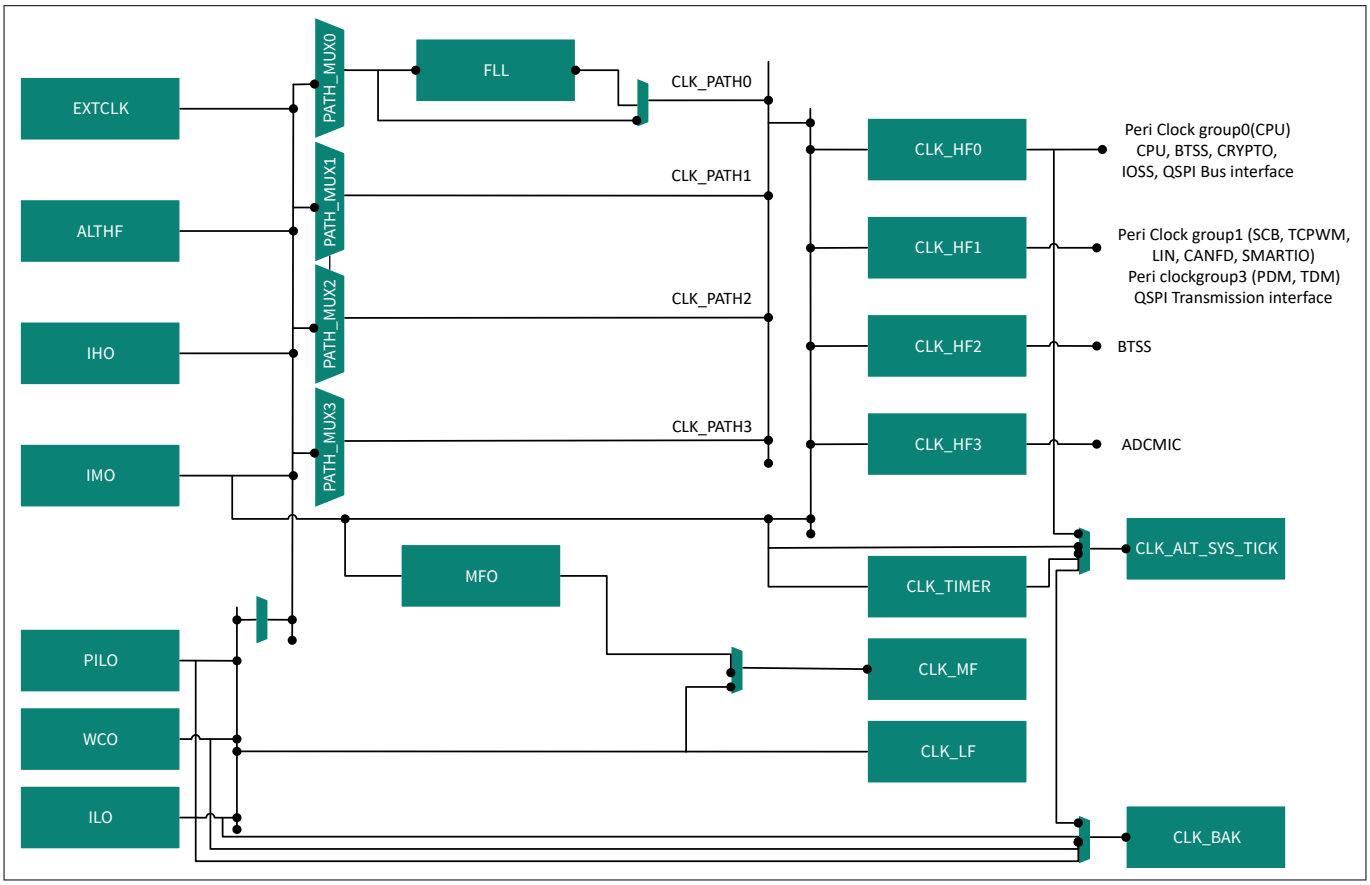

図 2 CYW20829 クロックツリー

2.3 CYW20829 電力モード

CYW20829 は、次のシステム電力モードで動作できます。これらのモードは、アプリケーションの平均電力消費を最小限に抑えることを目的とします。

CYW20829 の Arm® Cortex® M33 CPU コアは、次の 2 つの電圧選択モードで動作できます。

- **システム低消費電力 (LP)** - すべてのペリフェラルおよび CPU 電力モードは、1.1 V の安定化電圧でフルスピードで利用可能
- **システム超低消費電力 (ULP)** - すべての CPU 周辺機器と電力モードが利用可能ですが、速度と 1.0 V の電圧レギュレーションに制限があります

CYW20829 には、次の 5 つの SoC レベルの電力モードがあります。

- **CPU アクティブ:** CPU はシステム LP または ULP モードでコードを実行します。CPU コアへのクロックソースは、最小 6 MHz から最大 96 MHz まで選択できます。
- **CPU スリープ:** CPU コードの実行は、システム LP または ULP モードで停止します。このモードでは、CPU コアへのクロックソースがゲートされます。
- **システムディープスリープ:** MCU サブシステム (MCUSS) および Bluetooth®サブシステム (BTSS) の CPU が CPU ディープスリープモードに入った後は、低周波数ペリフェラルのみが使用可能です。
- **システムディープスリープ RAM:** RAM と I/O の状態のみが保持されます。選択した低電力周辺機器を除くすべてのシステムアクティビティは、システムがこの状態を終了するまで停止します。CPU は終了時にリセットされますが、RAM の内容が保持されるため、ソフトウェアの初期化をスキップできます。
- **システムハイバネート:** デバイスと I/O はフリーズ状態であり、デバイスはウェイクアップ時にリセットされます。ハイバネートモードから復帰すると、Bluetooth® LE アドバタイズメントのみが可能です。

図 3 に、CYW20829 での電力モードの遷移を示します。

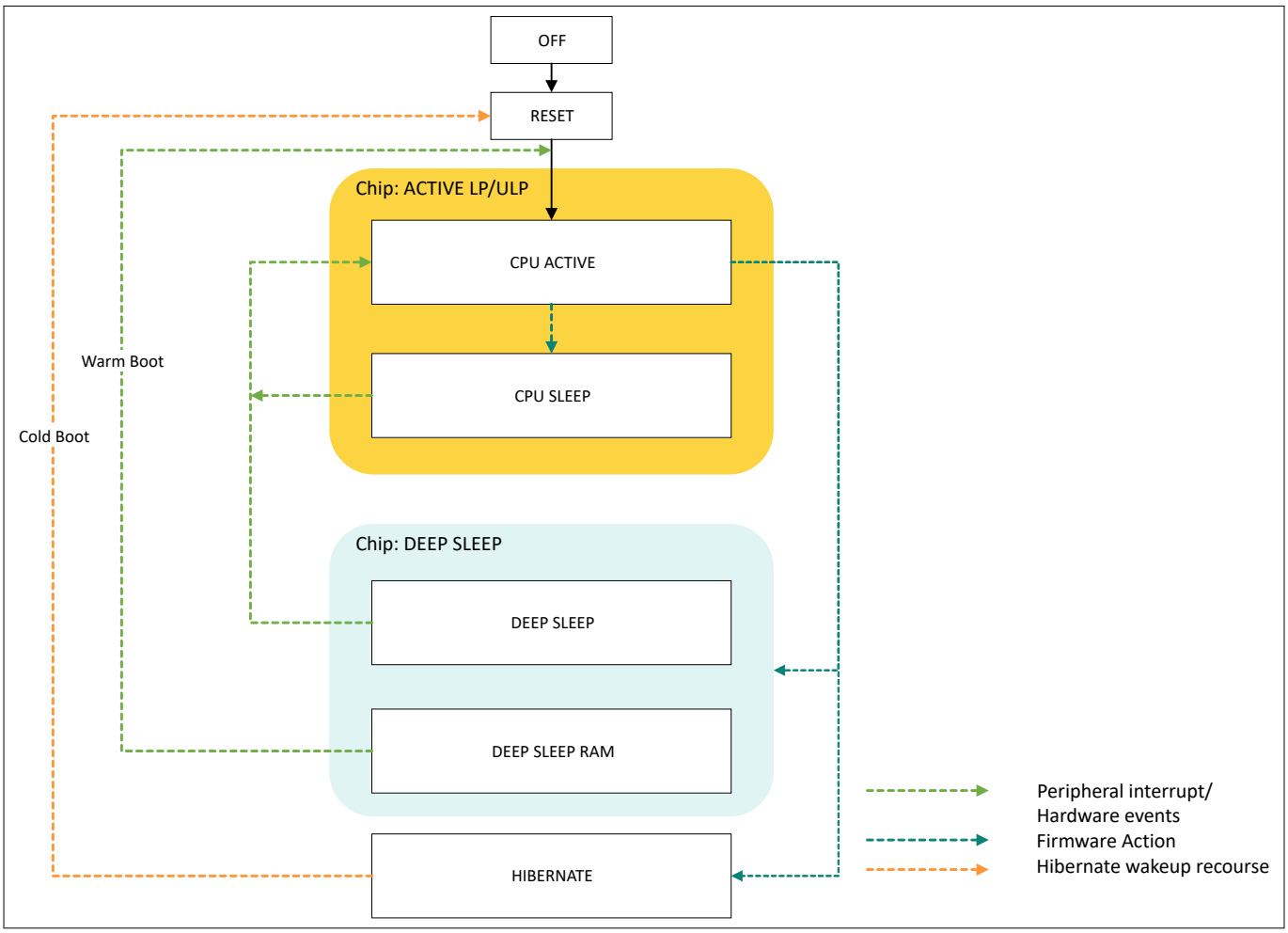

図 3　　電力モードの遷移

2.4　　MCU および Bluetooth®サブシステムのシステム電力モード遷移

電力モードの切り替えは MCU サブシステム (MCUSS) レベルで制御できますが、Bluetooth®サブシステム (BTSS) は独自の電力モードの状態遷移を内部で処理します。デバイスをディープスリープなどの低消費電力モードにするには、両方の CPU がディープスリープになっている必要があります。図 4 に、パワーオン リセット (POR) からのさまざまな電力モード間の遷移を示します。

2 CYW20829 パワーアーキテクチャ

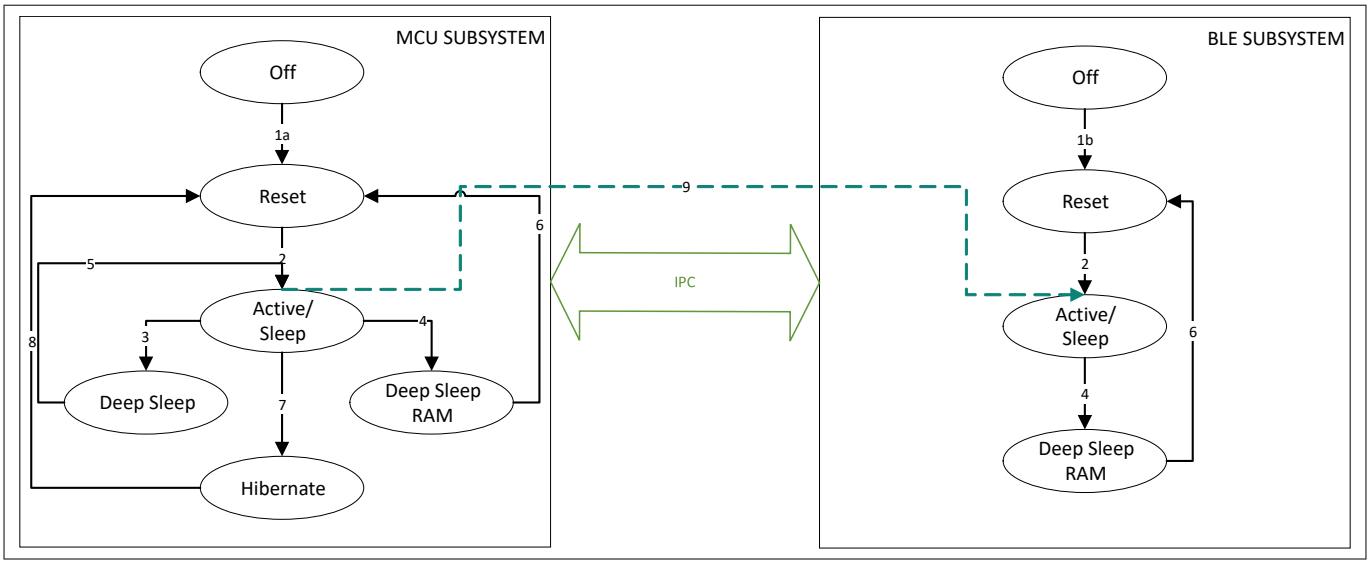

図 4　　MCUSS および Bluetooth®サブシステムの電力モード遷移

表 1　　MCUSS および Bluetooth®サブシステムの電力モード遷移とトリガー

遷移	説明	遷移トリガー	電力モード
1a	MCUSS OFF からリセットへの遷移	CYW20829 への電源適用	アクティブ
1b	Bluetooth®サブシステム OFF からリセットへの遷移 MCUSS が解放するまでリセットされたままになります	CYW20829 への電源適用	アクティブ
2	MCUSS/Bluetooth®サブシステムがアクティブ モード遷移にリセットされます。Bluetooth®サブシステムの遷移は、MCUSS がリセットから解放した後にのみ発生します。各サブシステムは、アクティブ モードとスリープ モードの間を個別に移動できます	リセットからアクティブモードへのファームウェア (ブート/スタートアップ) の遷移 MCUSS は、設定を Bluetooth®サブシステムにダウンロードします (プロセッサ間通信 (IPC) 経由)。両方のサブシステムがアクティブ モードに移行します。	アクティブ
3	MCUSS はディープスリープに移動します。Bluetooth®サブシステムがディープスリープ RAM 状態に移行します。 ディープ スリープおよびディープスリープ RAM の電力モードでは、Bluetooth®サブシステムの動作に違いはありません。	アクティブモードからディープスリープモードへのファームウェア (OS アイドルタスク) の遷移	ディープ スリープ
4	MCUSS と Bluetooth®サブシステムの両方がディープスリープ RAM 状態に移行します	アクティブモードからディープスリープ RAM モードへのファームウェア (OS アイドルタスク) の遷移	ディープスリープ RAM
5	MCUSS はディープスリープからアクティブに移動し、ディープスリープに入ったところから実行を再開します。Bluetooth®サブシステムは、ウォーム ブート ハンドラー (パス 6) を介してアクティブに移動します	ディープスリープのウェイクアップソース	アクティブ

(続く)

表 1 (続き) MCUSS および Bluetooth®サブシステムの電力モード遷移とトリガー

遷移	説明	遷移トリガー	電力モード
6	MCUSS と Bluetooth®サブシステムはどちらもディープスリープ RAM からアクティブに独立して遷移し、それぞれのウォーム ブート ハンドラを実行します	ディープスリープのウェイクアップソース	アクティブ
7	MCUSS はハイバネート モードへの遷移を開始します。Bluetooth®サブシステムは、この状態を無視できません。デバイスはハイバネート モードに入ります	アクティブモードからハイバネートモードへのファームウェアの遷移	ハイバネート
8	MCUSS がハイバネートからアクティブに遷移します	ハイバネートのウェイクアップソース	アクティブ
9	MCUSS は Bluetooth®サブシステムをリセットから解放します	Bluetooth® LE スタック初期化	アクティブ

***注**: Bluetooth® サブシステムの電力モードの遷移 (例えば、アクティブからスリープ) は、無線コントローラー ファームウェアによって内部的に処理されます。ユーザー アプリケーションは、これらの遷移を処理する必要はありません。*

2.5 システムの電力モード

CPU アクティブ, スリープ, およびディープスリープは Arm®標準により定義された電力モードであり、Arm® CPU 命令セットアーキテクチャ (ISA) によりサポートされます。システム LP, ULP, ディープスリープ, ディープスリープ RAM, およびハイバネートモードは、CYW20829 がサポートする追加の低消費電力モードです。

CPU は、Arm®で定義されているスリープおよびディープスリープ電力モードに選択的に配置できます。CPU は、RAM を除くほぼすべての回路の電源がオフになっているディープスリープ RAM (DS-RAM) モードも実装しています。RAM 内のデータは、状態を維持するために保持されます。終了すると、CPU はリセットされますが、RAM 内のデータを使用してソフトウェアの初期化をスキップできます。

表 2 に、各電力モードの詳細を示します。

表 2 電力モードのシステム状態

電力モード	説明	SRAM	利用可能なペリフェラル	復帰ソース	ウェイクアップ時間	ブート	再初期化が必要
低消費電力アクティブ	すべてのリソースは、最大のスピード (最大 96 MHz) で利用できます。	オン	すべてのペリフェラル	該当なし	該当なし	該当なし	無
超低消費電力アクティブ	すべてのリソースが利用可能ですが、速度は遅くなります (最大 48 MHz)	オン	すべてのペリフェラル	該当なし	該当なし	該当なし	無
CPU スリープ	CPU はコードの実行を停止します。 システムの残りの部分が利用可能です	オン	すべてのペリフェラル	すべてのペリフェラル	該当なし	再開	無

(続く)

表 2 (続き) 電力モードのシステム状態

電力モード	説明	SRAM	利用可能なペリフェラル	復帰ソース	ウェイクアップ時間	ブート	再初期化が必要
ディープ スリープ	CPU はコードの実行を停止します。 ほとんどのペリフェラルはオフになります。 高周波クロックはオフです。 低周波クロックは利用可能です。 SRAM は保持されます。 アクティブ・モード保持レジスタは保持されます	保持	1xSCB キースキャン RTC WDT/MCWDT	GPIO SCB 0 WDT/MCWDT RTC キースキャン Bluetooth® LE イベント	60 µs	再開	無
ディープスリープ RAM	CPU はコードの実行を停止します。 ほとんどのペリフェラルはオフ状態です。 高周波クロックはオフになります。 低周波クロックは利用可能です。 SRAM は保持されます。 アクティブ モード保持レジスタは保持されません	保持	1xSCB キースキャン RTC WDT/MCWDT	GPIO SCB 0 WDT/MCWDT RTC キースキャン Bluetooth® LE イベント	800 µs	Warm ブート	有
ハイバネート	CPU とクロックはオフです。 GPIO 出力状態はフリーズされます。 RTC アラームと専用の GPIO ウェイクアップ ピンを使用してシステムを起動できます。 バックアップ ドメインは使用可能。 SRAM は保持されません。	オフ	RTC WDT/MCWDT GPIO P0.5、P1.4	ハイバネートピン'0'と'1' RTC WDT/MCWDT	2000 µs	Cold ブート	無

2.5.1 Device Configurator でのシステム電力モードの選択

Device Configurator にはいくつかのタブがあり、各タブで特定のリソースにアクセスできます。以下の手順を実施してください。

1. Project Explorer で、プロジェクトを右クリックし、**ModusToolbox™** > **Device Configurator** を選択します
2. 選択したアプリケーションの Device Configurator を起動します。詳細は、Device Configurator ユーザーガイドを参照してください

3. システムを選択すると、システムクロック, 電力管理, およびデバッグインターフェースなど、チップ全体の設定のオプションが表示されます
4. システムの下の電力サブセクションを選択して、図 5 に示すように、CYW20829 で使用可能な設定可能な電力パラメータを表示します

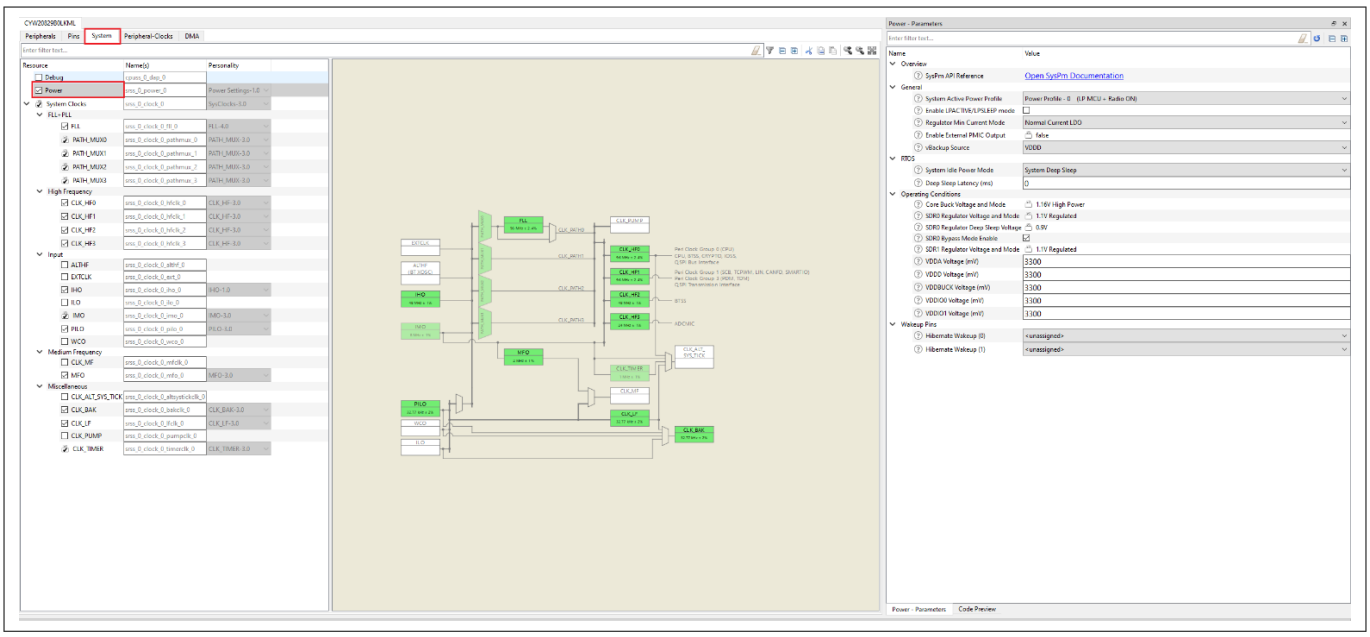

図 5　　CYW20829 の電力モードの選択

表 3　　システム電力パラメータ

Sl. 番号	パラメータ	説明	範囲/オプション
1	システムアクティブ電力プロファイル	システムアクティブ電力プロファイルの選択	オプション • 電力プロファイル - 0 (LP MCU + 無線オン) • 電力プロファイル - 1 (ULP MCU + 無線オン) • 電力プロファイル - 2A (LP MCU のみ) • 電力プロファイル - 2B (LP MCU のみ) • 電力プロファイル - 3 (ULP MCU のみ)
2	レギュレーターの最小電流モード	レギュレーターの最小電流モードをイネーブル	オプション • 通常電流 LDO • 最小電流 LDO

(続く)

表 3 (続き) システム電力パラメータ

Sl. 番号	パラメータ	説明	範囲/オプション
3	システムアイドル電力モード	システムがアイドル状態を維持するために必要な電力モードを選択します	オプション • アクティブ • CPU スリープ • システムディープスリープ • システムディープスリープ RAM
4	hibernate_wakeup[ ]	hibernate_wakeup[0]ピンを予約して、ハイバネート電力モードからのウェイクアップ ソースとして使用します。	P0.5 (固定)
5	hibernate_wakeup[1]	hibernate_wakeup[1]ピンを予約して、ハイバネート電力モードからのウェイクアップ ソースとして使用します。	P1.4 (固定)

- **電力プロファイル - 0 (LP MCU + 無線オン):** すべての周辺機器と CPU の電源モードは、96 MHz の最大構成可能な CPU クロックと、1.1 V の安定化電圧で利用できます。無線モジュールの電源がオンになっているため、Bluetooth®機能を利用できます
- **電力プロファイル - 1 (ULP MCU + 無線オン):** すべての CPU ペリフェラルと電源モードは、最大周波数 48 MHz と、1.0 V の電圧レギュレーションで動作します (レギュレーターの動作を参照)。無線モジュールの電源がオンになっているため、Bluetooth®機能を利用可能。
- **電力プロファイル - 2A (LP MCU のみ):** すべての周辺機器と CPU の電源モードは、1.1 V の安定化電圧でフルスピード (96 MHz) で使用できますが、無線モジュールがオフ状態であるため、Bluetooth®機能は使用できません。コア降圧電圧とモードは高電力 1.16 V、SDR0 はレギュレーションされた 1.1 V に設定されています
- **電力プロファイル – 2B (LP MCU のみ):** すべての周辺機器と CPU の電源モードは、1.1 V の安定化電圧でフルスピード (96 MHz) で使用できますが、無線モジュールがオフ状態であるため、Bluetooth®機能は使用できません。コア降圧電圧とモードは低消費電力の 1.1 V、SDR0 はバイパス 1.1 V に設定されています
- **電力プロファイル - 3 (ULP MCU のみ):** すべての CPU 周辺機器と電源モードは、最大周波数 48 MHz で、1.0 V の安定化電圧で動作します。Bluetooth®無線モジュールがオフになっています
- **レギュレーターの最小電流モード:** ユーザーは、アプリケーションでレギュレーターの電流制限が満たされていることを確認する責任があります。(1) 通常電流 LDO (2) 最小電流 LDO の 2 つのオプションがあります
  - **通常電流 LDO:** システムを通常レギュレーター電流モードに設定し、LP または ULP モードでのシステム動作を変更して、最大コア電流消費を提供します。LDO コアレギュレーターを使用している場合は、通常モードの出力電流制限を使用できます。降圧レギュレーターが使用されている場合でも、その低減された電流出力制限が適用されます
  - **最小電流 LDO:** システムを最小コア レギュレーター電流モードに設定し、システム コア ロジックで使用可能な最大電流を制限します。このモードは、LP モードまたは ULP モードでのシステム動作を変更して、消費電流をさらに削減します。システム電流がアクティブ コア電圧レギュレーター (LDO または降圧) のデータシートの電流制限を下回っている場合は、このモードに入ることができます
- **システムアイドル電力モード:** デバイスは、アイドル状態で選択したアイドル電源モード (アクティブ/スリープ/ディープスリープ/ディープスリープ RAM など) のままです
- **ハイバネートウェイクアップ:** 名前が示すように、ウェイクアップ ピンはデバイスをハイバネート電力モードからウェイクアップするために使用されます。デバイスがハイバネートからウェイクアップすると、コールドブート手順が実行されます。ブートを参照してください

Device Configurator は、レギュレーター, ペリフェラル, ピン, クロックソース, 電力モード, およびプロファイルのオプションを使用して、一連のマクロと初期化関数を生成します。これらの機能はすべて、以下で詳しく説明するシ

ステム電源管理 (SysPm) ライブラリを使用して実行することもでき、高度なユースケースを有効にするのに役立ちます。

3 ソフトウェアによる電力管理

3.1 ブート

ブート手順では、アプリケーションの main() 関数に入る前に、CYW20829 SoC のさまざまなブロックを初期化して基本動作モードに入ります。ブート手順は、POR,ソフトウェアリセット, ハイバネート ウェイクアップ, およびディープスリープ RAM ウェイクアップで呼び出されます。

CYW20829 は、電力モードに応じて 2 種類のブート手順をサポートします。

- コールド ブート: POR,ソフトウェアリセット, およびハーバネート ウェイクアップは、コールドブート手順で実行されます。
- ウォーム ブート: ディープスリープ RAM ウェイクアップは、ウォームブート手順で実行されます。

***注**: CPU スリープおよびディープスリープ電力モードからウェイクアップする場合、ブート手順は関与しません。ブート手順のタイプは、main() 関数の先頭で呼び出すことができる Cy_SysPm_GetBootMode() 関数を使用して特定できるため、実行されたブートのタイプに応じてアプリケーションで適切なアクションを実行できます。*

3.1.1 コールド ブート

このブートモードでは、システムクロックと関連するクロックパスが初期化され、main()関数に入る前に C ランタイム環境が設定されます。ハイバネート ウェイクアップの場合、デバイスはバックアップ電源ドメインリソースを除くすべてをリセットします。初期電源 ON には、バックアップ電源ドメインもリセットされるというわずかな違いがあります。バックアップ電源ドメインの詳細については、電源ドメインのバックアップを参照してください。

3.1.2 ウォーム ブート

ディープスリープ RAM モードからウェイクアップすると、ウォームブート手順になります。ウォームブートは、完全なブートプロセスの短縮版です。アプリケーションは、ディープスリープ RAM からウェイクアップしたときの CPU の実行開始点であるウォームブートのエントリポイントを指定できます。これにより、ブート手順の実行時間が短縮されます。

これは、Cy_Syslib_SetWarmBootEntryPoint() 関数を使用して行われます。この関数を使用すると、関数アドレスなどのエントリ ポイントを、復元する必要があるスタック ポインター値と共に指定できます。低消費電力モード遷移の実装を参照してください。この機能により、RTC アラーム割込みを設定した後にデバイスがディープスリープ RAM 状態に入るなど、さまざまな使用例を実装できます。アラーム割込みがトリガーされると、デバイスはディープスリープ RAM からウェイクアップし、CPU はウォームブート手順を実行し、制御はエントリポイント関数に移ります。

ウォーム ブート手順では Serial Memory Interface (SMIF) ブロックは初期化されないため、フラッシュ内の任意の関数にジャンプするには、エントリ ポイント関数で SMIF ブロックを初期化する必要があることに注意してください。したがって、エントリポイント機能は SRAM に格納されます。低電力モードに入った時点から CPU の実行を復元するには、ウェイク フォー割込み (WFI) の直後に Cy_Syslib_SetWarmBootEntryPoint() 関数を呼び出し、スタック ポインターを復元する必要があります。SMIF ブロックを初期化してから、ユースケースに応じて main 関数にジャンプできます。

3.2 システム電源管理 (SysPm) ライブラリ

3.2.1 PDL

ペリフェラル ドライバ ライブラリ (PDL) は、ペリフェラルとシステム レジスタを設定し、目的の機能を実装するための API を含んだ完全なソフトウェア ツールです。PDL は、ターゲットデバイスのほぼすべてのハードウェアリソー

スへの直接アクセスを提供します。これにより、レジスタとビット構造を理解して直接アクセスする必要が減ります。

PDL では System Power Management (SysPm) API が電力モード変更の関数を提供します。SysPm ドライバは、図 6 に示すように、電力モードの遷移の前後に実行するコールバック関数を登録する関数を提供します。PDL は、Project Creator ツールを使用して新しいアプリケーションを作成すると自動的にダウンロードされます。または、modus-shell から `make getlibs` コマンドを実行することもできます。このライブラリにはサブカテゴリ 1A と 1B のドライバがありますが、CAT1B PDL ドライバは CYW20829 でのみサポートされていることに注意してください。

SysPm PDL は、ModusToolbox™ Peripheral Driver Library (mtb-pdl-cat1) からダウンロードできます。

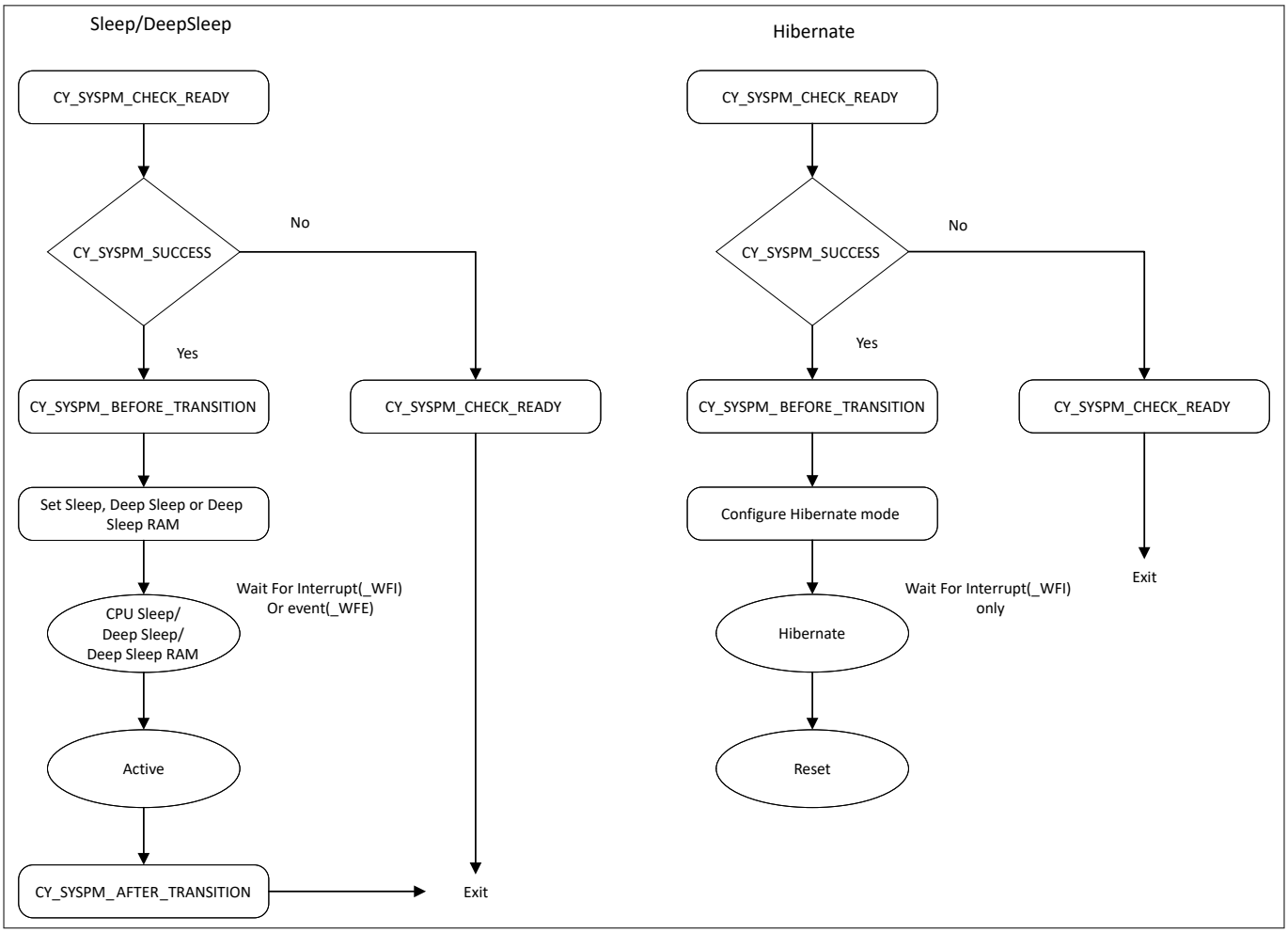

図 6　　電力モードの遷移

3.2.1.1　コールバック モード

表 4　　CYW20829 MCU 電力モードのコールバック操作

コールバック関数	説明
CY_SYSPM_CHECK_READY	他のモードに移行するための準備状態を確認します。 CY_SYSPM_FAIL を返した場合、遷移なしで終了します。
CY_SYSPM_CHECK_FAIL	コールバックは、CY_SYSPM_CHECK_READY を失敗した場合にのみ実行されます。これはロールバック アクションを実行します。

(続く)

表 4 (続き) CYW20829 MCU 電力モードのコールバック操作

コールバック関数	説明
CY_SYSPM_BEFORE_TRANSITION	コールバックは、モード遷移前に必要なアクションを実行し、設定します
CY_SYSPM_AFTER_DS_WFI_TRANSITION	ディープスリープに入った場合、ディープスリープを終了した後、割込みが有効になる前に実行するアクションを実行します。
CY_SYSPM_AFTER_TRANSITION	コールバックは、モード遷移または設定後に実行します。

SysPm ドライバは登録および実行という 2 つのコールバック用の関数を提供します。これらの関数は電力の最適化を支援するだけでなく、モード遷移後の異常なペリフェラル状態の防止にも有効です。PDL と HAL を使用する場合は、各電力モードのコールバックを登録する必要があります。ほとんどのペリフェラルドライバには、各電力モードに関連付けられた定義済みのコールバックがあります。定義されたペリフェラルコールバックを登録するか、またはカスタム コールバックの作成することを選択できます。SysPm 遷移関数は、登録されたコールバックを順番に実行します。最初に登録された関数が最初に実行されます。詳細については、PDL を使用したコールバック関数の登録を参照してください。

3.2.2 HAL

Hardware Abstraction Layer (HAL) は、MCU でハードウェアブロックを設定および使用するための高レベルのインターフェースを提供します。これは、複数の製品ファミリで使用できる汎用インターフェースです。使いやすさと移植性に重点を置いているということは、HAL ライブラリがすべての低レベルのペリフェラル機能を公開しているわけではないことを意味します。HAL 内では、システム電源管理 (SysPm) API がシステムレベルで電源管理を行う機能を提供します。この API を使用して、ハイバネート ウェイクアップ ソースの設定, CPU 電力状態の設定, エントリ/ウェイクアップ コールバックの登録, システム スリープとディープスリープへの移行などの機能を実行できます。

HAL ライブラリは、Project Creator ツールを使用して新しいアプリケーションを作成するか、modus-shell から make getlibs コマンドを実行すると、自動的にダウンロードされます。HAL ライブラリをできるだけ使用し、HAL ライブラリで特定の機能が使用できない場合にのみ PDL ライブラリに移行することを推奨します。

SysPm HAL ライブラリは、ModusToolbox™ Hardware Abstraction Layer (mtb-hal-cat1) ページからダウンロードできます。

3.3 低消費電力モード遷移の実装

3.3.1 ベアメタル アプリケーションへの低消費電力モードの実装

システムは、HAL API または PDL API を使用して、スリープ, ディープスリープ, ディープスリープ RAM, ハイバネートなどの電力モードに入ることができます。次の表に、さまざまなシステム電力モードを切り替えるための HAL API を示します。

表 5 API を使用した電力モード設定

電力モード	API
スリープ	cyhal_syspm_sleep()
ディープ スリープ	cyhal_syspm_Deep Sleep()
ディープ スリープ RAM	cyhal_syspm_Deep Sleep()
ハイバネート	cyhal_syspm_hibernate()

3 ソフトウェアによる電力管理

デバイスをディープスリープとディープスリープ RAM に移行させるために同じ API が使用されるため、CYW20829 をそれぞれのモードに移行するようにアイドル電力モードを設定します。アイドル電力モードは、Device Configurator (図 7 参照) を使用するか、Cy_SysPm_SetDeep SleepMode() API を使用して実行時に変更することで設定できます。

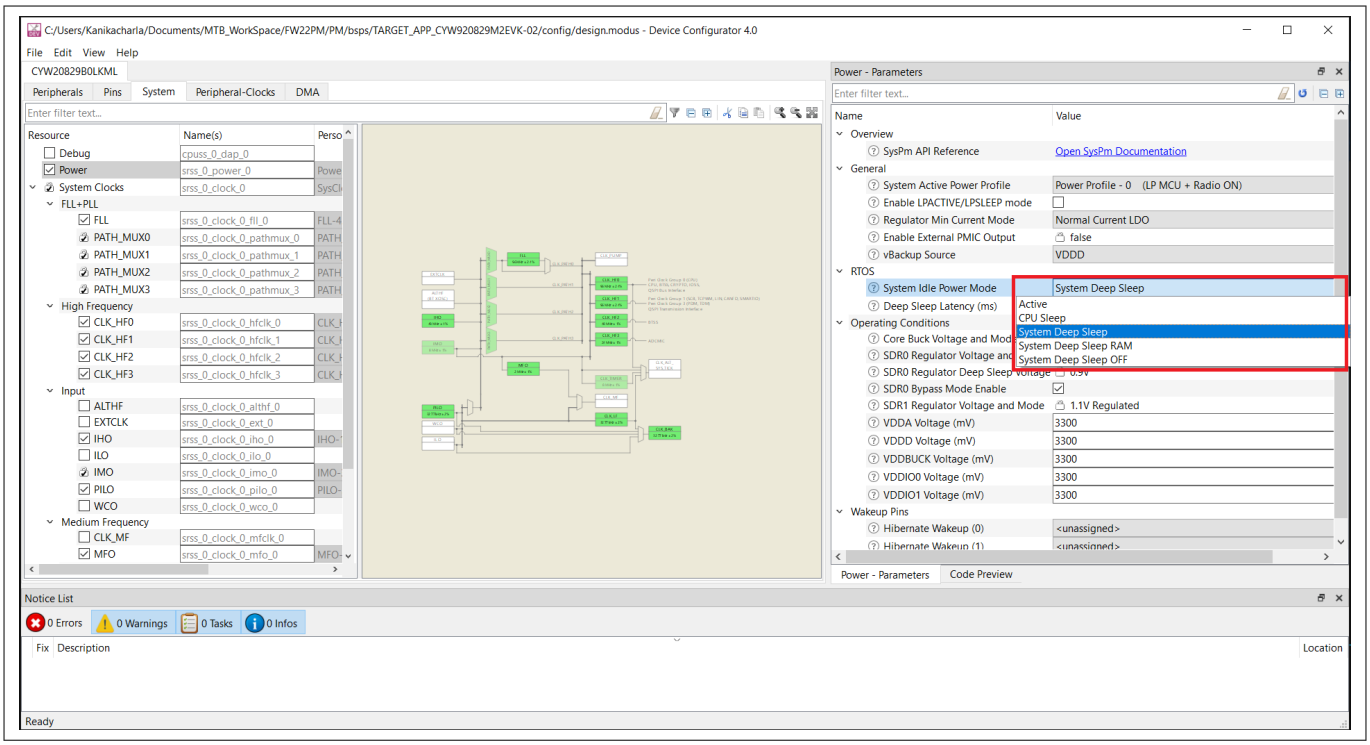

図 7　　システムアイドル電力モードの設定

3.3.1.1　ディープスリープ RAM モードに入る前に実行する手順

1. ウォームブートのエントリポイントが `Cy_Syslib_SetWarmBootEntryPoint()`関数を使用して設定されていることを確認します。ウォームブートエントリの詳細については、ウォーム ブートを参照してください。
2. HAL SysPm コールバックの優先度が「10」であるため、優先度を「0」から「9」に設定してディープスリープ RAM コールバックを登録します。
3. ディープスリープ RAM からウェイクアップすると、依存するソフトウェアフレームワークレイヤーによって登録されたシステムコールバック関数が最初に実行され、次にアプリケーションに登録されたコールバック関数が実行されることに注意してください。
4. `System_Store_NVIC_Reg()`関数を使用して NVIC レジスタを RAM に格納し、デバイスがディープスリープ RAM からウェイクアップした後も NVIC レジスタ データが保持されるようにします。

3.3.1.2　ディープスリープ RAM モードからウェイクアップした後に実行する手順

1. ペリフェラル制御レジスタを介してすべてのペリフェラル ブロックを有効にし、`SystemInit_Warmboot_CAT1B_CM33()`関数を使用してすべての I/O をアンフリーズします。
2. フラッシュを使用するように SMIF ブロックを初期化します。
3. `System_Restore_NVIC_Reg()`関数を使用して、RAM から NVIC データを復元します。
4. 低消費電力モードを終了した後、`Cy_SysPm_ExecuteCallback()`関数を使用してウェイクアップ後に登録済みのディープスリープ RAM コールバックを実行することにより、遷移後コールバックを実行します。
5. 登録されたディープスリープ RAM コールバックで、必要なペリフェラルを初期化解除して初期化します。

制御は、アプリケーションに基づいて他の機能にリダイレクトされます。次のプログラムコードは、DSRAM 電力モードから復帰した後に実行されるプロセスを示します。

ディープスリープ RAM モードからのウェイクアップ

```
/* DS-RAM Warmboot Re-entry */
    cy_stc_syspm_warmboot_entrypoint_t syspm_app_dsram_entry_point =
    {(uint32_t *)&__INITIAL_SP, (uint32_t *)cy_app_warmboot_handler};

CY_SECTION_RAMFUNC_BEGIN
int  cy_app_warmboot_handler(void)
{
    warm_boot=1;
    SystemInit_Warmboot_CAT1B_CM33();

    cybsp_smif_init();

    init_cycfg_all();

    System_Restore_NVIC_Reg();

    /* Suppress a compiler warning about unused return value */
    (void)Cy_SysPm_ExecuteCallback(((cy_en_syspm_callback_type_t)
 CY_SYSPM_DEEP SLEEP_RAM), CY_SYSPM_AFTER_TRANSITION);

    main();
    return 0;
}
CY_SECTION_RAMFUNC_END

cy_rslt_t cybsp_syspm_dsram_init(void)
{

    /* Sets the warm boot entry point and disables the debug pins */
    Cy_Syslib_SetWarmBootEntryPoint((uint32_t*)&syspm_app_dsram_entry_point,
 false);

     return CY_RSLT_SUCCESS;
}
```

3.3.2 FreeRTOS での低消費電力モードの実装

アイドル電力モードは、図 7 に示すように、FreeRTOS のすべての電力モード遷移を決定します。アイドル電力モードがディープスリープ RAM として設定されている場合、FreeRTOS フレームワークは、次の操作を実行できる方法で作成されます。

- アイドル タスクの現在のコンテキストを格納します
- ウォームブートエントリポイントを設定します
- NVIC レジスタを RAM に格納および再格納
- after-transition コールバック ハンドラーを実行する

3.3.2.1 ディープスリープ RAM に入る前に実行する手順

1. 2 番目のパラメータを'false'として `Cy_Syslib_SetWarmBootEntryPoint()`関数に渡すことで、確実にデバッグモードを無効にします。
2. アイドル電力モードがディープスリープに設定されている場合、実行時にディープスリープ RAM に入るには、次のコード ブロックを追加して、実行時にアイドル電力モードを変更します。

```
Cy_SysPm_SetDeep SleepMode(CY_SYSPM_MODE_DEEP SLEEP_RAM);
```

3. ディープスリープ RAM レイテンシを追加するには、アプリケーションの Makefile に次の設定を追加します。

```
# Add additional defines to the build process (without a leading -D).
DEFINES+=CY_CFG_PWR_DEEP SLEEP_RAM_LATENCY=20
```

3.3.2.2 ディープスリープ RAM からウェイクアップした後に実行する手順

移行後コールバックで必要なペリフェラルを再初期化します。

RTOS の電力遷移の詳細については、RTOS ティックレスモードを参照してください。

***注**: デバイスがディープスリープ RAM にある場合、デフォルトでは、ピン P1_0, P1_1, P1_2, および P1_3 がブート ROM JTAG アクティビティに使用されます。したがって、プログラムでこれらのピンを使用すると、誤ったブレークが発生する可能性があります。デバッグモードを無効にするには、2 番目のパラメータを'false'として `Cy_Syslib_SetWarmBootEntryPoint()` 関数に渡します。*

3.4 PDL を使用したコールバック関数の登録

次のプログラムコードに、SysPm コールバックを登録する方法を示します。

システム電力モードコールバックの登録

```
cy_stc_syspm_callback_params_t syspm_cpu_sleep_params;
cy_stc_syspm_callback_params_t syspm_deep_sleep_params;
cy_stc_syspm_callback_params_t syspm_hibernate_params;
cy_stc_syspm_callback_params_t syspm_dsram_params;

cy_stc_syspm_callback_t syspm_cpu_sleep_cb_handler =
{
    syspm_cpu_sleep_cb,
    CY_SYSPM_SLEEP,
    0u,
    &syspm_cpu_sleep_params,
    NULL,
    NULL,
    253
};

cy_stc_syspm_callback_t syspm_deep_sleep_cb_handler =
{
    syspm_ds_cb,
    CY_SYSPM_DEEP SLEEP,
    0u,
    &syspm_deep_sleep_params,
    NULL,
    NULL,
    253
};

cy_stc_syspm_callback_t syspm_dsram_cb_handler =
{
    syspm_dsram_cb,
    CY_SYSPM_DEEP SLEEP_RAM,
    0u,
    &syspm_dsram_params,
    NULL,
    NULL,
    0
};

cy_stc_syspm_callback_t syspm_hibernate_handler =
{
    syspm_hibernate_cb,
    CY_SYSPM_HIBERNATE,
    0u,
    &syspm_hibernate_params,
    NULL,
    NULL,
    253
};
```

システム電力モードコールバックの登録

```
Cy_SysPm_RegisterCallback(&syspm_cpu_sleep_cb_handler);
Cy_SysPm_RegisterCallback(&syspm_deep_sleep_cb_handler);
Cy_SysPm_RegisterCallback(&syspm_dsram_cb_handler);
Cy_SysPm_RegisterCallback(&syspm_hibernate_handler);
```

3.5 PDL を使用したカスタム コールバック関数の実装

SysPm コールバックでは、電力モードの切り替えの前後に実行するアクションを定義できます。例えば、次のプログラムコードでは、ディープスリープ RAM モードに入る前に CTS GPIO と RTS GPIO を High-Z に設定し、ウェイクアップ後にこれらの GPIO をデフォルトのプルアップ設定にします。

ユーザー定義の SysPm コールバックの実装

```
CY_SECTION_RAMFUNC_BEGIN
cy_en_syspm_status_t
syspm_dsram_cb( cy_stc_syspm_callback_params_t *callbackParams,
                cy_en_syspm_callback_mode_t mode)
{
    cy_en_syspm_status_t retVal = CY_SYSPM_FAIL;
    CY_UNUSED_PARAMETER(callbackParams);

    switch(mode)
    {
        case CY_SYSPM_CHECK_READY:
        {
            retVal = CY_SYSPM_SUCCESS;
        }
        break;

        case CY_SYSPM_CHECK_FAIL:
        {
            retVal = CY_SYSPM_SUCCESS;
        }
        break;

        case CY_SYSPM_BEFORE_TRANSITION:
        /* Performs the actions to be done before entering the low power mode */
        {
            /*CTS and RTS pins are configured as analog high-z as it is connected to
             * kitprog3 and also drawing more current */
            cyhal_gpio_configure(CYBSP_BT_UART_RTS, CYHAL_GPIO_DIR_INPUT,
CYHAL_GPIO_DRIVE_ANALOG);
            cyhal_gpio_configure(CYBSP_BT_UART_CTS, CYHAL_GPIO_DIR_INPUT,
CYHAL_GPIO_DRIVE_ANALOG);
            Cy_Syslib_SetWarmBootEntryPoint((uint32_t*)&syspmBspDeep SleepEntryPoint, false);
            retVal = CY_SYSPM_SUCCESS;
        }
        break;

        case CY_SYSPM_AFTER_DS_WFI_TRANSITION:
        {

            retVal = CY_SYSPM_SUCCESS;
        }
        break;

        case CY_SYSPM_AFTER_TRANSITION:
        /* Performs the actions to be done after entering the low power mode */
        {
            if(Cy_SysLib_IsDSRAMWarmBootEntry())
            {
            cy_retarget_io_deinit();
```

ユーザー定義の SysPm コールバックの実装

```
            cy_retarget_io_init(CYBSP_DEBUG_UART_TX,
CYBSP_DEBUG_UART_RX,CY_RETARGET_IO_BAUDRATE);
            /*CTS and RTS pins are configured to it's default configuration */
            cyhal_gpio_configure(CYBSP_BT_UART_RTS, CYHAL_GPIO_DIR_OUTPUT,
CYHAL_GPIO_DRIVE_PULL_NONE);
            cyhal_gpio_configure(CYBSP_BT_UART_CTS, CYHAL_GPIO_DIR_INPUT,
CYHAL_GPIO_DRIVE_PULL_NONE);
            }
            retVal = CY_SYSPM_SUCCESS;
        }
        break;

        default:
            break;
    }

    return retVal;
}
CY_SECTION_RAMFUNC_END
```

3.6 アクティブ節電のヒント

3.6.1 CYW20829 を使用した電流パスのゲート

ボードには、電力を使用する他のコンポーネントが含まれている場合があります。したがって、使用していないときに電流を流す可能性のあるコンポーネントを分離する必要があります。GPIO は、温度センサーやデジタルマイクなどのさまざまなコンポーネントに対して、より高い電流をソースおよびシンクできます。これらの GPIO は、一部の外部ペリフェラルに直接電力を供給するために使用できます (センサーが必要とする電流が GPIO がサポートする電流制限内の場合)。その場合、外部ペリフェラルへの電力は、CYW20829 デバイスを介して直接制御できます。不要な場合は、周辺機器を OFF にできるため、システムレベルで電力を節約できます。

3.6.2 GPIO コンフィギュレーション

GPIO は、低消費電力モードでロジック状態を保持できます。これは、外部ロジックを固定レベルに保持する必要がある時に役には立ちますが、ピンが電流を不必要に流し出したり、引き込んだりする時には、電力の無駄になります。この手法の具体的な省電力化は、特定の GPIO ピンに接続された回路によって異なります。

設計を解析し、低電力動作時の GPIO の最適な状態を判断します。

デジタル出力ピンを論理「1」または「0」に保持すると電流が最小になる場合、`cyhal_gpio_write()`を使用してそのデジタルレベルを一致させてください。

```
/* Set P0_0 to ‘0’ for low power using HAL APIs */
    cyhal_gpio_write(P0_0,0u);
```

別の駆動モードを使用する特別な理由がないかぎり、すべての未使用 GPIO をアナログ High-Z に設定します。

アナログ High-Z 駆動モードでは、GPIO ピンの電流が最小になります。ピンの駆動モードは、cyhal_gpio_configure() HAL 関数を使って設定できます。

```
/* Set P0_0 to Analog HI-Z for low power using HAL APIs. */
    cyhal_gpio_configure(P0_0, CYHAL_GPIO_DIR_INPUT, CYHAL_GPIO_DRIVE_ANALOG);
```

3.6.3 クロックゲーティング

一部のクロックとペリフェラルは、不要なときに無効にできます。

使用しないクロックソース: System Resources サブシステムは、多くの内部クロック ソースを提供します。ペリフェラル ブロックで使用されないクロック ソースは無効にできます。水晶発振器 (WCO) などの一部のクロックソースは、起動時間が比較的長いことに注意してください。これらの回路を OFF と ON に切り替えると、全体的な電流が増加する可能性があります。

周波数を下げるには、IMO などの低周波数ソースを選択するか、別の出力周波数 (FLL など) を選択するか、System Resources サブシステムのプリデバイダまたは PERI ディバイダを使用してクロックを分周します。

3.6.4 DMA を使用してデータをコピー/移動する

ダイレクトメモリアクセス (DMA) は、CPU を介さずにメモリとの間でデータ転送を可能にする手法です。CPU をバイパスすることで、DMA は CPU 使用率と電力消費を大幅に削減できます。DMA はブロック単位でデータを読み書きできるため、必要なメモリアクセスの数が減り、電力を節約できます。さらに、DMA はデータをバーストで転送できるため、バスの使用率が向上し、消費電力が削減されます。DMA は、CPU 処理と同時にデータ転送を行うことができるため、システムのレイテンシと消費電力を削減できます。また、DMA は、CPU が低速のデバイスがデータ転送を完了するのを待たなければいけないときに発生する待機状態を排除し、電力を節約することもできます。

さらに、DMA はシステムによって生成される割込みの数を減らすことができるため、CPU 使用率と電力消費の両方が削減されます。DMA は CPU よりも低いクロック周波数で動作できるため、消費電力が削減されます。最後に、DMA エンジンはシステム LP モードと ULP モードの両方で使用できるため、CPU を使用せずにデータを転送できます。全体として、DMA は、CPU とペリフェラル間の相互作用を最小限に抑えることにより、システムの消費電力を削減するいくつかの方法を提供します。

3.6.5 オーディオ サブシステム

I2S (TDM) および PDM/PCM ブロックは、通常 PLL から生成された高周波クロックによって供給されます。オーディオ サブシステムが使用されていない場合は、I2S ブロックや PDM/PCM ブロックを無効にします。また、高電流消費の主な原因である FLL を無効にします。フレームレートの精度がアプリケーションにとって重要でない場合は、FLL を使用しないでください。オーディオ サブシステムの割込みを使用してファームウェアの FIFO にアクセスする代わりに、FIFO レベルのトリガーによって制御される DMA を使用して、アプリケーションで必要な CPU サイクルの数を減らし、CPU がスリープ状態に長く留まるか、他のタスクを実行できるようにします。

3.6.6 周期的ウェイクアップ タイマー

CPU スリープモードからの定期的なウェイクアップは、消費電力を削減する最も一般的な方法です。平均消費電力は、CPU アクティブ期間の消費電力と CPU スリープ期間の消費電力の比率によって決まります。

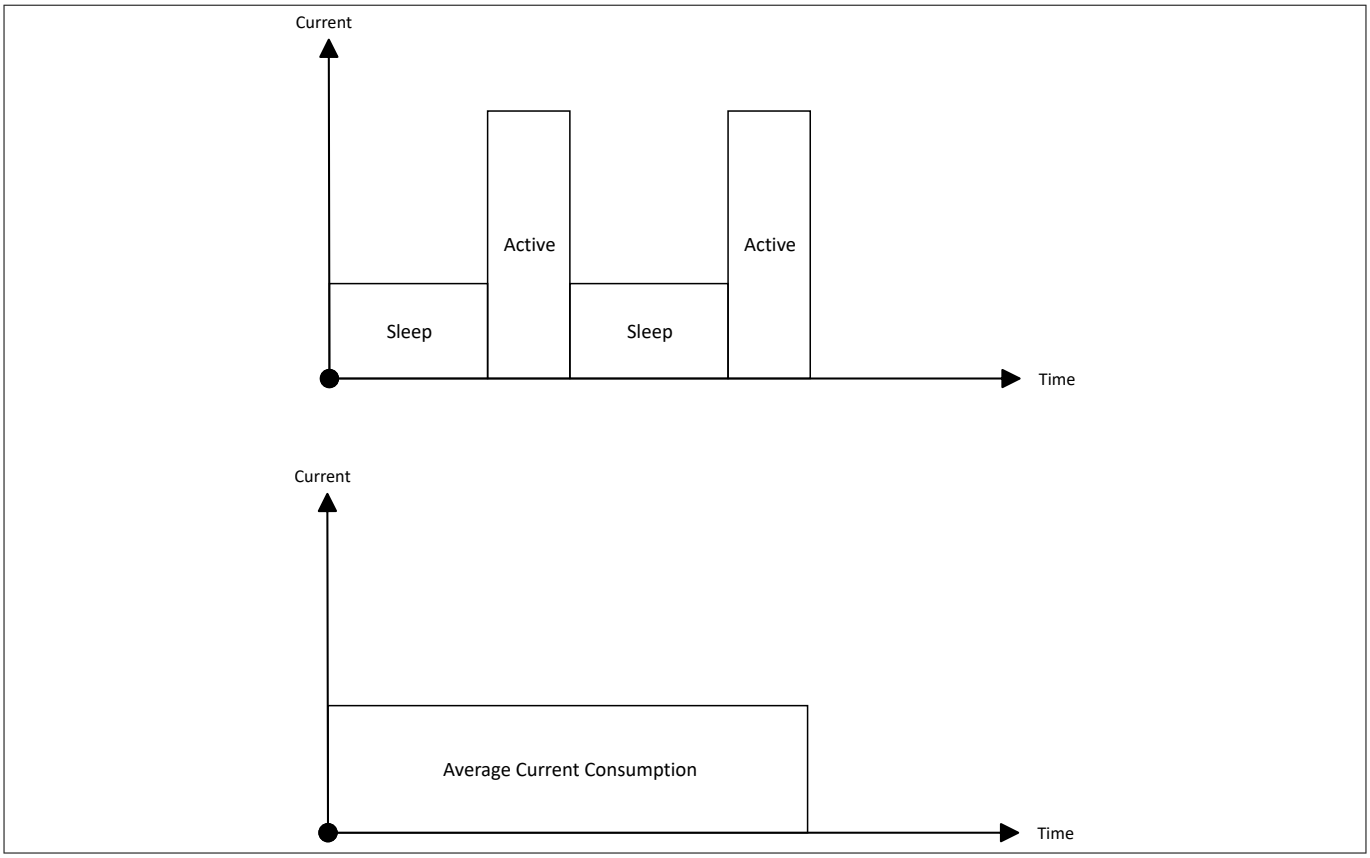

図 8　　周期的ウェイクアップ電流消費

3.6.7　SRAM

CYW20829 は、個々の SRAM バンクまたはバンク内のページの電源をオフにできます。バンク内のページサイズは、デバイスデータシートに詳述されているように、特定のデバイスとバンクに依存します。特定のデバイスには、1 つ以上の SRAM バンクがあります。未使用のページは、CPUSS 電源制御レジスタに書き込むことで無効にできます。この手法は、データシートに記載されているように、128 KB SRAM を保持すると 5 µA を消費し、256 KB の SRAM を保持すると 6 µA の動作電流を消費するシステムディープスリープモードで最も役立ちます。

CYW20829 SRAM は 256 KB のメモリで構成されています。図 9 に示すように、さらに 4 つのページに分割されています。各ページは、次のプログラムコードを使用して個別に電源を OFF にできます。これを行うには、それぞれのページにデータが保持されていないことを確認してください。

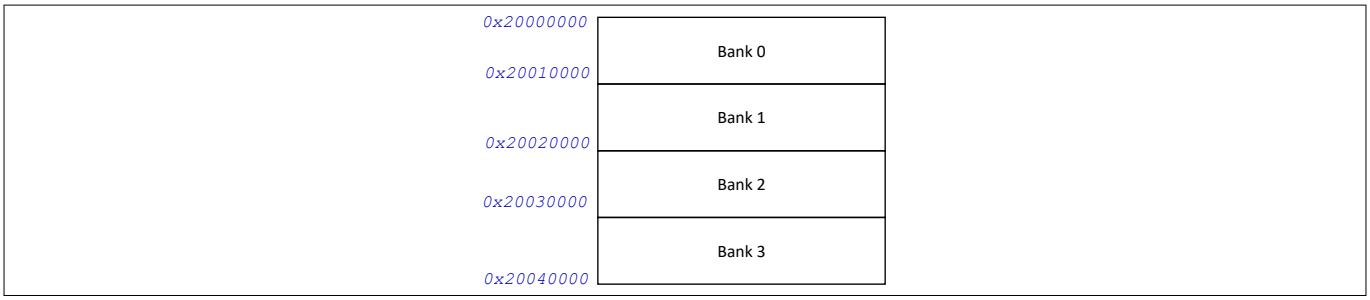

図 9　　SRAM ページ

SRAM バンク 0 の電源をオフにするには、アプリケーションで次の変更を行う必要があります。

1. linker_ns_flash.ld ファイルのリンカ スクリプトを更新して、そのデータが SRAM バンク 0 に配置されないようにします。

```
RAM_START_ADDR_SAHB          = 0x20010000;
RAM_START_ADDR_CBUS          = 0x20010000;
```

2. bsp.mk ファイル内の SRAM のサイズを更新します。

```
DEVICE_CYW20829B0LKML_SRAM_KB=192
```

3. 次の機能を使用して、SRAM バンク 0 の電源を OFF にします。

```
/* To power OFF the certain RAM section */
Cy_SysPm_SetSRAMMacroPwrMode(CY_SYSPM_SRAM0_MEMORY,0,CY_SYSPM_SRAM_PWR_MODE_OFF);
```

3.6.8 SCB

データを送受信する際にブロック機能を使用しないでください。割込みベースのイベントまたは RTOS を使用してデータを転送し、CPU を他のタスクに明け渡します。この戦略の背後にある考え方は、送信のステータスをポーリングするのではなく、CPU をスリープ状態に長く維持することです。図 10 に、ブロッキング関数と非ブロッキング関数の例を示します。

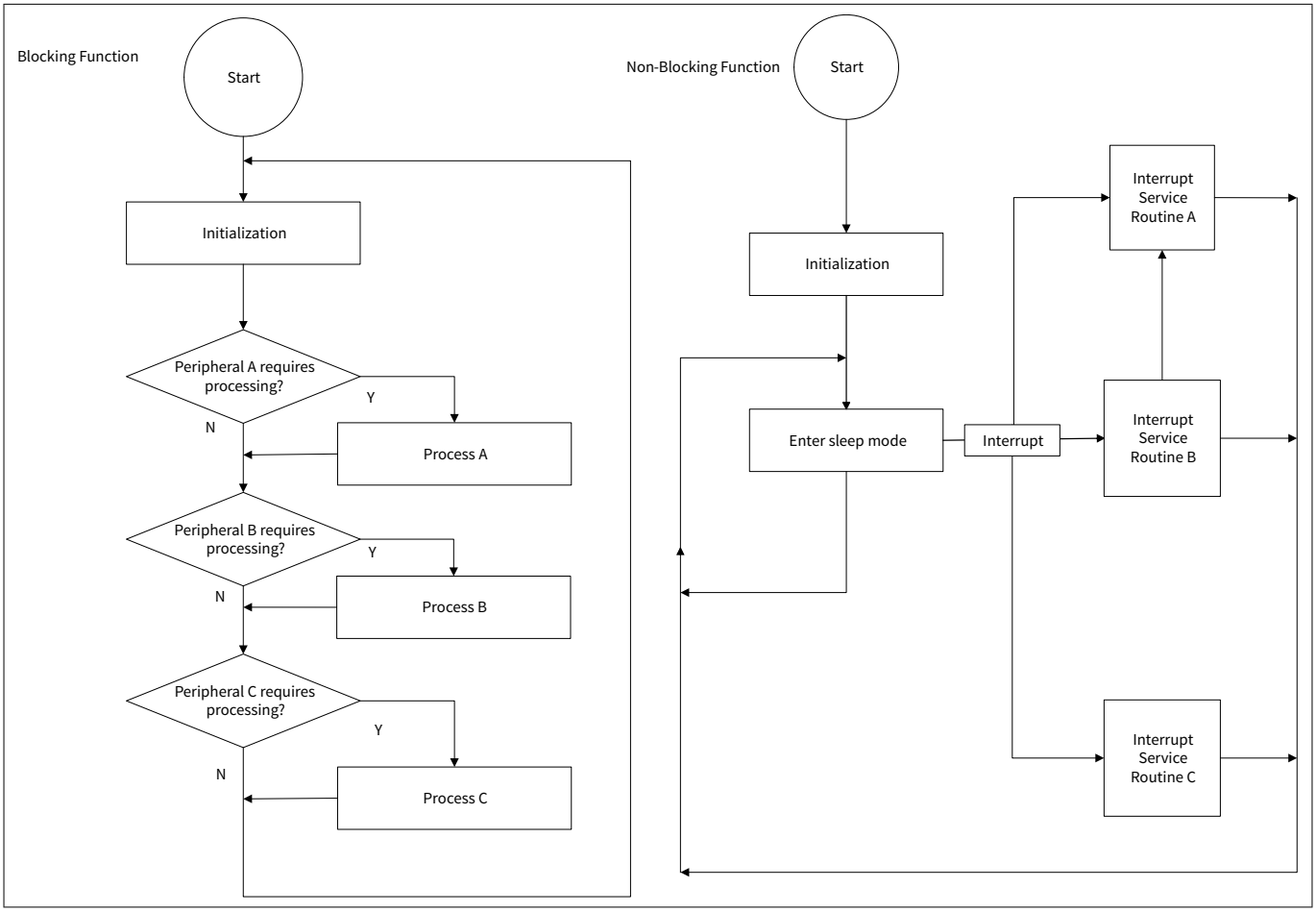

図 10 ブロッキングおよび非ブロッキング機能の例

SCB 割込みを使用してファームウェアで FIFO にアクセスする代わりに、FIFO レベルによって制御される DMA を使用して、アプリケーションに必要な CPU サイクル数を減らします。これにより、CPU はスリープ状態に長く留まり、電力消費を削減したり、他のタスクを実行したりできます。CYW20829 の SCB の 1 つは、ディープ スリープ モードで使用できます。この機能を使用すると、外部デバイスからのシリアル通信データを待機している間、システムをディープスリープモードに保つことができます。

3.6.9 TCPWM

カウンタ, タイマー, または PWM を使用する場合は、周波数と精度の要件を満たしながら、チャネルを可能なかぎり低くするクロックを設定する必要があります。例えば、タイマーで 1 秒の割込みを生成する必要がある場合は、周期が 1,000,000 カウントに等しい 1 MHz のクロック周波数よりも、周期が 1,000 カウントに等しい 1 kHz のクロック周波数を使用することを推奨します。TCPWM クロックの削減による電力節約は、ほとんどの場合、クロック周波数に基づいて線形です。

PWM を使用して LED を調光する場合も同じ考え方が当てはまります。LED が点滅していることを人間の目に認識させないように、可能な限り最小のクロック周波数を使用してください。60 Hz は、ほとんどのアプリケーションに適した周波数です。TCPWM ブロックのクロック設定の比較を図 11 に示します。

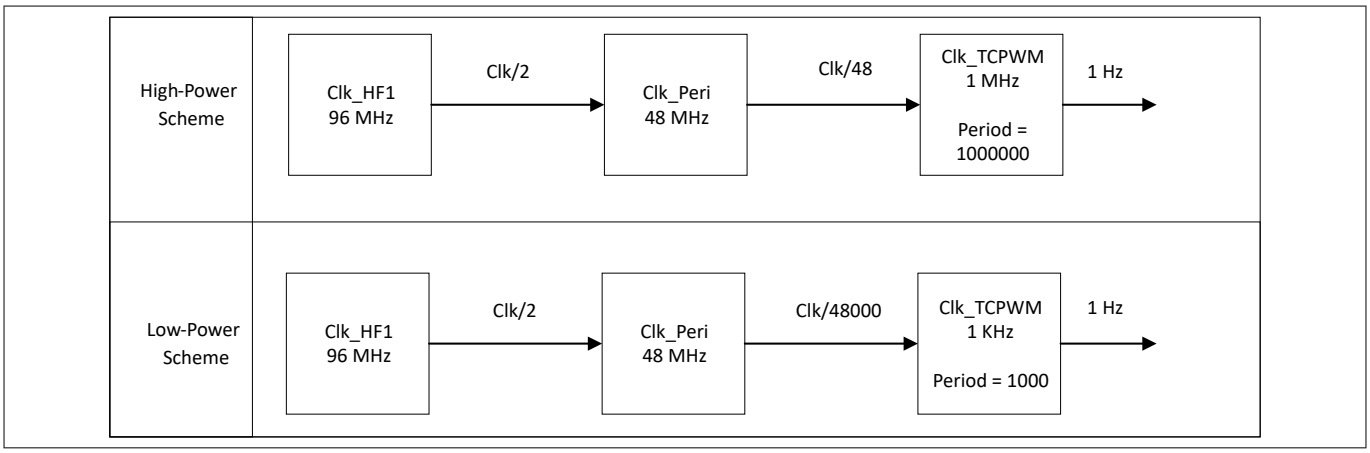

図 11 TCPWM クロック設定の比較

TCPWM ブロックには、クロック プリスケーラー機能があります。消費電力を最小限に抑えるには、TCPWM クロック プリスケーラーを使用する前に、まずペリフェラル クロック分周器を最大化します。TCPWM ブロックがピン接続を使用している場合、ブロックが無効になっているときにピンを Analog High-Z に設定します。

3.6.10 RTOS ティックレスモード

すべての RTOS には、タスクのタイミングと管理にクロックまたは「ティック」が必要です。これらのティックは通常、小さな RTOS カーネルタスクであり、定期的にウェイクアップし、遅延, タイムアウト, およびその他のタイミング関連タスクで使用される RTOS ティックカウントを更新します。多くの場合、これらの目盛りは ms で設定されます。これにより、有用なタスクが実行されていなくても、ティックタスクを処理するために CPU が頻繁にウェイクアップします。

遅延やタイムアウトを使用しない、または大きな遅延を使用するアプリケーションは、これを頻繁にウェイクアップする必要がなく、プラットフォームが提供するティックレスモードに入ることができます。ティックレスモードでは、RTOS はティックタスクを一時停止し、CPU を長時間低電力状態 (ディープスリープ) に移行させます。この期間は通常、遅延に依存する次のタスクをトリガーするのに必要な時間によって決まります。遅延に依存するタスクがない場合、外部イベント (GPIO 割込みなど) がデバイスをウェイクアップしてタスクをトリガするまで、デバイスは無期限に低電力状態を維持できます。

ティックレスモードに入るときに遅延が発生する場合、システムのディープスリープ状態でアクティブなタイマーは、遅延が経過した後に割込みを生成するように設定されます。

3 ソフトウェアによる電力管理

すべての RTOS プラットフォームは、ティックレス動作モードを提供します。さらに、アイドル時に OS がスリープ状態になるかディープスリープ状態になるかを制御できます。これは、OS (CPU) がアイドル状態であっても、ADC や UART などのペリフェラルが引き続き動作し、システムがディープスリープに入らないようにする必要がある場合に役立ちます。このような場合、ディープスリープエントリを「ロック」して、CPU が代わりにスリープ状態に入り、ペリフェラルが動作を完了した後でディープスリープエントリを「ロック解除」できます。

4 高度な電力管理

4.1 電力制御アーキテクチャ

CYW20829 は Arm®電力制御アーキテクチャを使用しています。1 つの電源ドメインを制御するために使用されるコンポーネントを図 12 に示します。

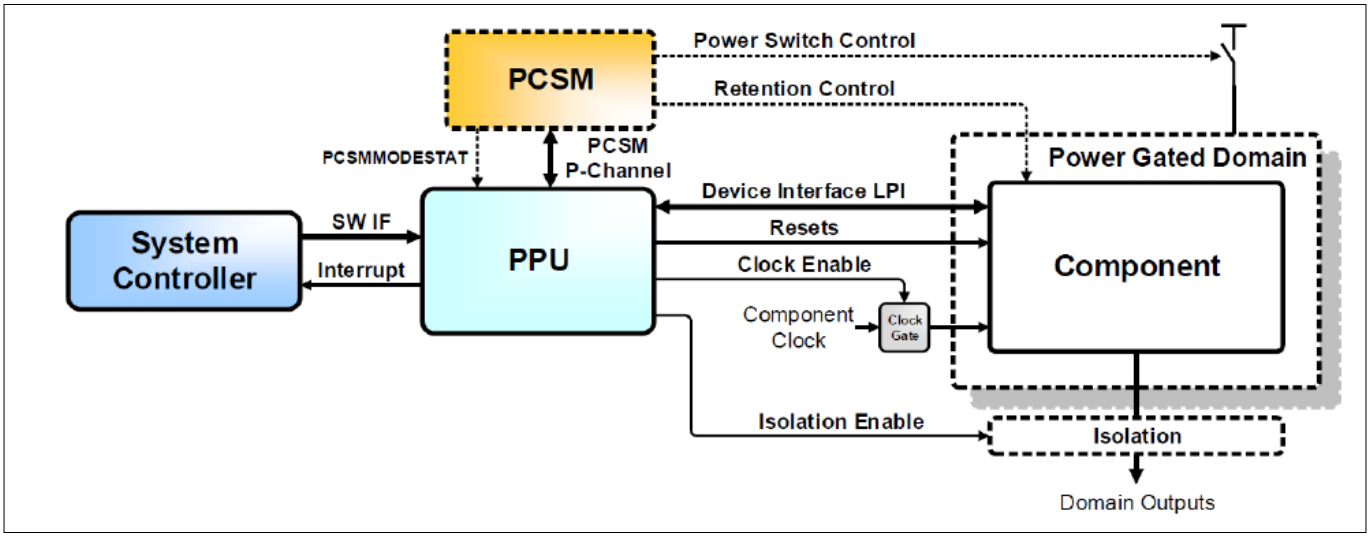

図 12 Arm®電力制御アーキテクチャにおけるパワー ゲーティング[5]

- 通常、**Power Policy Unit (PPU)** は電源ドメインを制御しますが、1 つの電源ドメイン内に個別のリセットドメインを実装するためにも使用できます。「SW IF」は、システムコントローラーからのソフトウェアインターフェースです (例: M33 プロセッサ)。これは PPU の Advanced Peripheral Bus (APB) インターフェースであり、非同期バス ブリッジを介して Advanced High-performance Bus (AHB) メモリ マッピング I/O (MMIO) にマッピングされます。「LPI」とラベル付けされたインターフェースは、Low Power Interface の略で、通常は Q チャネルまたは P チャネルとして実装されます。
- **Power Control State Machine (PCSM)** は、電源スイッチを制御するカスタムハードウェアモジュールです。System Controller とは、MCU (M33 など) を指します。VCCACT_PD を制御するメイン アクティブ PPU の特定のケースでは、システム コントローラーは電力ゲートドメイン内にあります。この場合、ソフトウェアインターフェース (SW IF) と割込み信号は必要に応じて隔離されます。

Arm®リファレンスデザインには、「Power Dependency Control Matrix (PDCM)」と呼ばれるもう 1 つのカスタムコンポーネントがあります。すべての PPU は、電源ドメイン間の依存関係を処理する PDCM とインターフェースします。

4.1.1 各 PPU でサポートされる電力状態

CYW20829 には 3 つの PPU があります。表 6 および表 7 を参照してください。

表 6 各 PPU でサポートされる電力モード

PPU	PPU 状態
PPU_MAIN	FULL_RETENTION
	MEMORY_RETENTION
	オフ
	オン

(続く)

表 6 (続き) 各 PPU でサポートされる電力モード

PPU	PPU 状態
PPU_CPUSS	FULL_RETENTION
	オフ
	オン
PPU_SRAM	MEMORY_RETENTION
	オフ
	オン

表 7 ディープスリープ状態への PPU 設定

ディープスリープ状態	PPU 電力モード
ディープスリープ	PPU_MAIN: FULL_RETENTION
	PPU_CPUSS: FULL_RETENTION
	PPU_SRAM: MEMORY_RETENTION
ディープスリープ RAM	PPU_MAIN: MEMORY_RETENTION
	PPU_CPUSS: オフ
	PPU_SRAM: MEMORY_RETENTION

4.2 レギュレータートポロジー

CYW20829 のレギュレーター ブロックは、リニアなステップダウン レギュレーターを備えたスイッチング ブート レギュレーターです。このトポロジーでは、OFF または XRES からの遷移後にコア降圧スイッチング レギュレーター (CBUCK) を使用します。CBUCK のターゲット電圧はプログラム可能です。

2 つのリニア ステップダウン レギュレーター (SDR0 および SDR1) は、システム使用のためのコアレールを生成します。

- SDR0 は、CPU コア電圧である VCCD を生成します
- SDR1 は一般的な使用のためのものです。独立したデジタルドメインとして、または Bluetooth® LE 無線のレギュレーターとして使用できます。

4.3 コア電圧選択

4.3.1 リニアレギュレーターおよび降圧レギュレーター

- **コア降圧レギュレーター (CBUCK):** このレギュレーターは、VCCD を直接生成することも、降圧レギュレーターで使用するための中間電圧 (VCCI) を生成することもできます。デバイスがディープ スリープになると、VCCD 電圧は 0.9 V に変化します。
- **リニア ステップダウン レギュレーター (SDR0 および SDR1):** 各 SDR は、VCCI を使用して低電圧のコア電源レール (VCCD など) を生成するリニア LDO レギュレーターです。どちらの CBUCK モード (低消費電力および低リップル) も、プラットフォーム電圧レベル (1.0 V および 1.1 V) に対して SDR が動作するのに十分なヘッドルームを持った電圧、またはバイパスモードをターゲットとします。SDR0 は、入力電圧がレギュレーションに対して低すぎて、接続されたロジックのタイミング クロージャ内にある場合にバイパスできます。
- **HV-LDO:** これは、VDDD を使用してプログラム可能な電圧を生成するリニア低ドロップアウト レギュレーターです

4.3.2 レギュレーターの割り当て

- SDR0 はデジタル ステップダウン レギュレーターとして割り当てられています。
- SDR1 は、Bluetooth®サブシステムで使用する RF ステップダウン レギュレータ (RF-SDR) として割り当てられています。ディープスリープ中、この SDR は無効になります。
- HV-LDO は、10 dBm 送信の場合に無線のパワーアンプに電力を供給する PA-LDO として割り当てられています。

4.3.3 レギュレーターの動作

- **CBUCK:** このレギュレーターは、低リップルおよび低消費電力モードで動作します。低リップルモードは、無線が動作しているときに使用されます
- **SDR0 (D-SDR):** このレギュレーターは、設定と入力電圧に応じて、バイパスモードまたはレギュレーションモードで動作します。これにより、CBUCK レギュレーターはデジタル電圧を直接ターゲットにでき、デジタルシステムが動作しているが無線が動作していないときの全体的な電力効率が向上します。CBUCK の低消費電力モードまたは低リップル モードを使用したバイパス モードとレギュレーション モードで動作することが期待されています。SDR0 電圧は、Device Configurator で System Active プロファイルを ULP に選択することで変更できます。これは、次のプログラムコードを使用して実行時に変更することもできます。

SDR0 電圧コンフィグレーションの設定

```
#define CY_CFG_PWR_SDR0_VOLT CY_SYSPM_SDR_VOLTAGE_1_000V

static cy_stc_syspm_sdr_params_t sdr0ConfigParam =
{
    .coreBuckVoltSel = CY_CFG_PWR_CBUCK_VOLT,
    .coreBuckMode = CY_CFG_PWR_CBUCK_MODE,
    .coreBuckDpSlpVoltSel = CY_SYSPM_CORE_BUCK_VOLTAGE_0_90V,
    .coreBuckDpSlpMode = CY_SYSPM_CORE_BUCK_MODE_LP,
    .sdr0DpSlpVoltSel = CY_SYSPM_SDR_VOLTAGE_0_900V,
    .sdrVoltSel = CY_CFG_PWR_SDR0_VOLT,
    .sdr0Allowbypass = CY_CFG_PWR_SDR0_MODE_BYPASS,
};

    /* SDR0 Regulator Configuration */
    Cy_SysPm_SdrConfigure(CY_SYSPM_SDR_0, &sdr0ConfigParam);
```

- **SDR1 (RF-SDR):** このレギュレーターは無線に電力を供給し、CBUCK が意図したものより高い出力電圧の低リップルモードで動作すると想定できます。CBUCK 低リップルモードを使用したバイパスモードで動作することは想定されていません

4.4 電源ドメインのバックアップ

バックアップ電源ドメインには、次の主要なコンポーネントが含まれます。

- バックアップクロック
  - ILO
  - WCO
  - PILO
- リアルタイム クロック (RTC)
- 少量のアプリケーション状態を保持するためのバックアップレジスタ (BREG)

バックアップ ドメインには、XRES を含むほとんどのリセットでも保持されるロジックが含まれています。バックアップ ドメイン電源ピン (VBACKUP) は VDDD から完全に独立しており、VDDD が存在しない場合でも実行を続行できます。VBACKUP は、機能と低電力モードとの互換性を最大化するために、常に使用可能な電源に接続する必要があります。これは、バッテリまたはスーパーキャパシタから得られる独立した電源にできます。

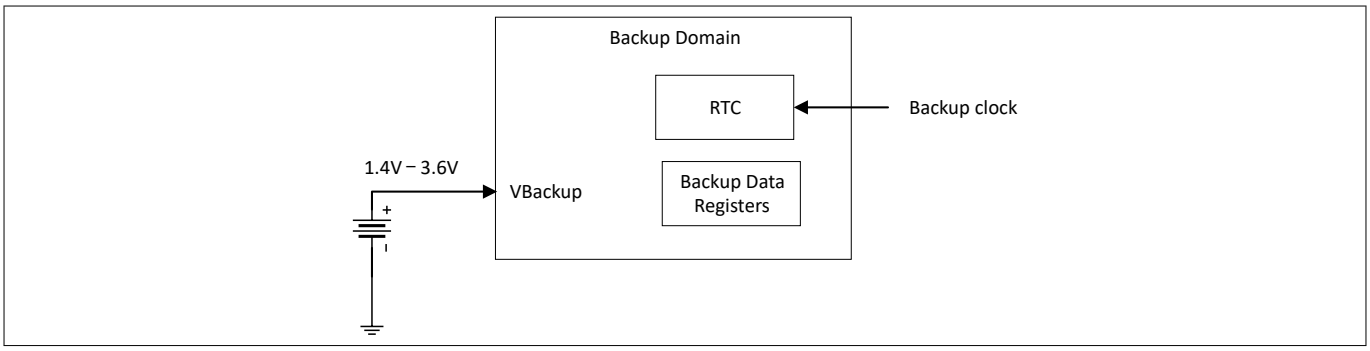

図 13　　バックアップ ドメインのブロック図

4.5　　バックアップ データ レジスタ

バックアップドメインには、VBACKUP 電源が有効である限り、その内容を保持する 16 個の 32 ビットレジスタ BACKUP_BREG[15:0]が含まれています。各レジスタは 4 バイト、合計 64 バイトのデータを保持します。これらのレジスタは、デバイス全体の電源オフ時に重要なシステム情報とフラグを保持するために使用されます。また、システムのハイバネート モード中にも使用できるため、デバイスのウェイクアップとリセット後にデータを読み出すことができます。これは、ユーザーがバックアップドメインに一部のデータを保存して、主電源の電源障害が発生した場合でも保持するために使用できます。バックアップ レジスタの処理は、低レベルのドライバ API によって処理されます。ユーザー アプリケーションは、これらのレジスタを制御する必要はありません。

5 低消費電力サンプルコード

5.1 CE237951 – AIROC™ CYW20829 スイッチング電力モード

このサンプルコードでは、デバイスを低消費電力および超低電力システムに移行し、アクティブからスリープ, アクティブからディープ スリープ, アクティブからディープ スリープ RAM, アクティブからハイバネートへの電力モードに移行する方法を示します。

このサンプルコードでは、ユーザー ボタンを使用して、電力モードをアクティブからスリープ, アクティブからディープ スリープ, アクティブからディープ スリープ RAM, アクティブからハイバネートに変更します。システムがハイバネート モードに入ると、ウェイクアップ リソースを待機します。別のユーザーボタンは、電力システムを低消費電力システムと超低消費電力システム、またはその逆に変更します。このサンプルコードでは、ウェイクアップ リソースとしてユーザー ボタン 1 を示します。図 14 に、トランジションを実行するためにファームウェアに実装されたステートマシンを示します。

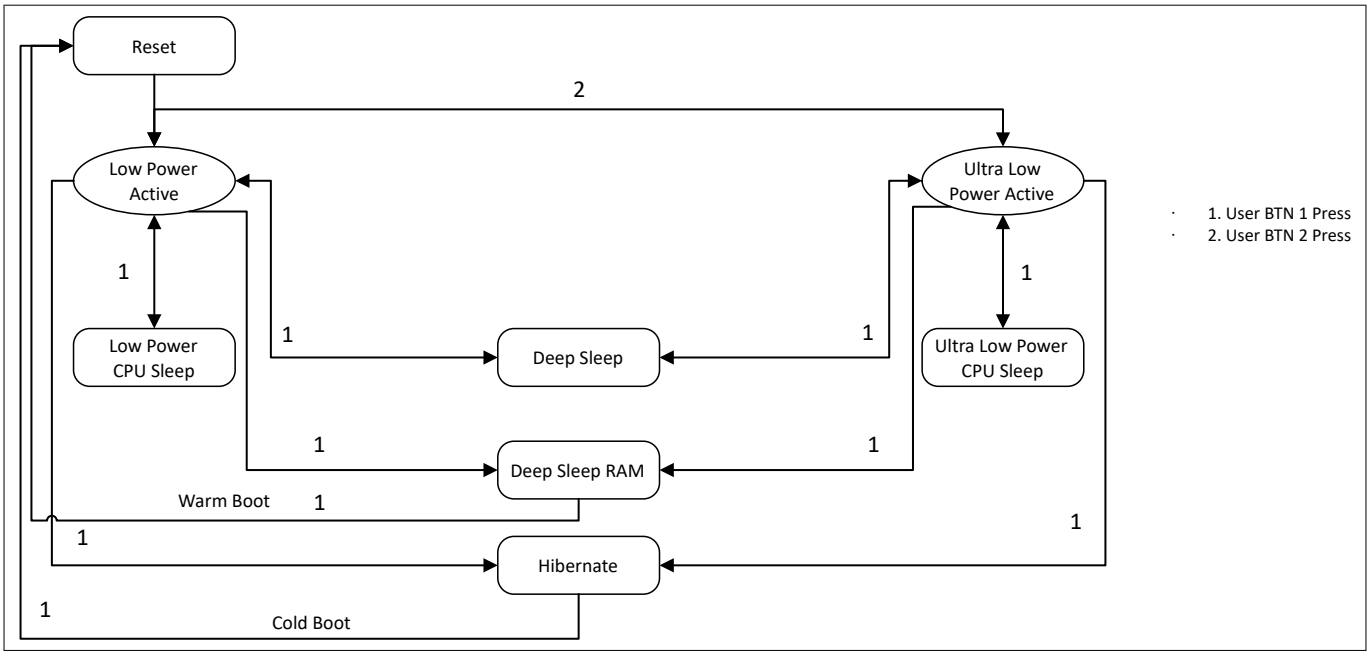

図 14 電力モードのステートマシン

5.2 CE238052 – AIROC™ CYW20829 FreeRTOS スイッチング電力モード

このサンプルコードでは、デバイスをアクティブからスリープ, アクティブからディープ スリープ, アクティブからディープ スリープ RAM, アクティブからハイバネートの電力モードに移行する方法を示します。

このサンプルコードでは、ユーザー ボタン 1 を使用して、電力モードをアクティブからスリープ, アクティブからディープ スリープ, アクティブからディープ スリープ RAM,およびアクティブからハイバネートに変更します。システムがハイバネート モードに入ると、ウェイクアップ リソースを待機します。このサンプルコードでは、ウェイクアップ リソースとしてユーザー ボタン 1 を示します。図 15 に、トランジションを実行するためにファームウェアに実装されたステートマシンを示します。

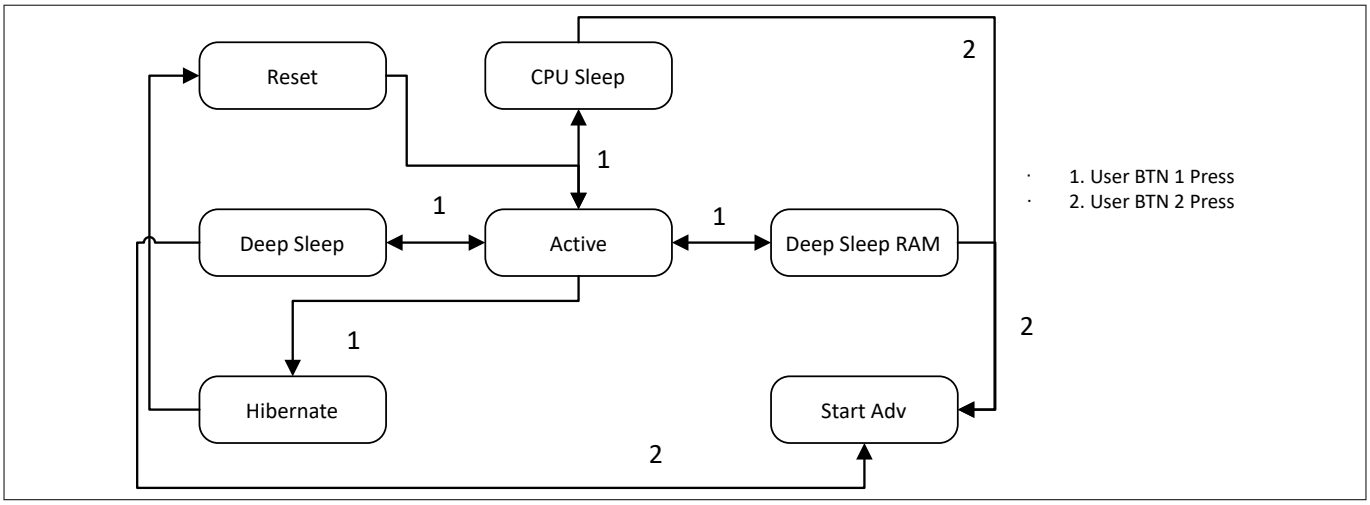

図 15　　電力モードのステートマシン

5.3　CE237278 – HAL RTC アラームの定期的なウェイクアップ

このサンプルコードでは、ディープ スリープ モードとハイバネート モードに入る方法と、RTC を使用して RTC アラームを生成し、ディープ スリープ モードとハイバネート モードから MCUSS をウェイクアップする方法を示します。

MCUSS は、ユーザー ボタンが押された時間に基づいて、ディープ スリープ モードまたはハイバネート モードのいずれかになります。RTC アラームはシステムをウェイクアップし、ウェイクアップ情報を UART 端末に出力します。図 16 に、トランジションを実行するためにファームウェアに実装されたステートマシンを示します。詳細については、CE237278 - HAL RTC アラームの定期的なウェイクアップを参照してください。

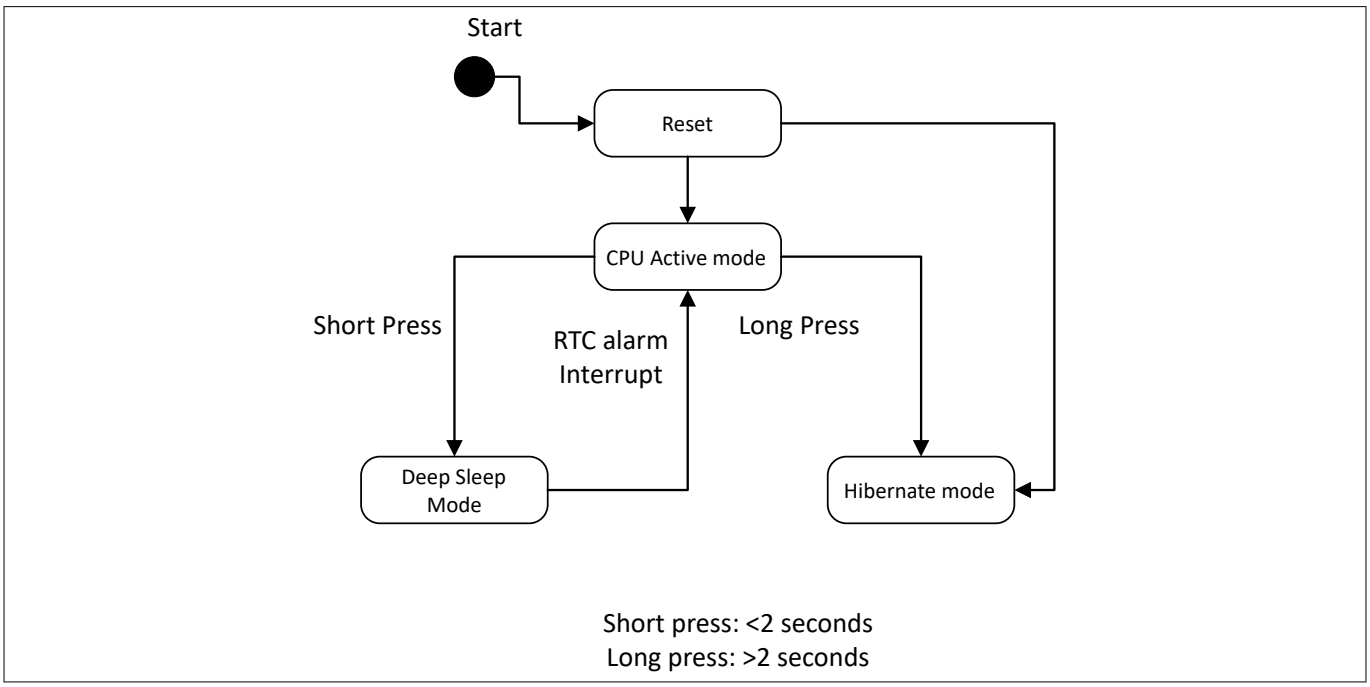

図 16　　割込みを使用した RTC 周期的ウェイクアップ タイマー

6 電力測定

6.1 ハードウェア

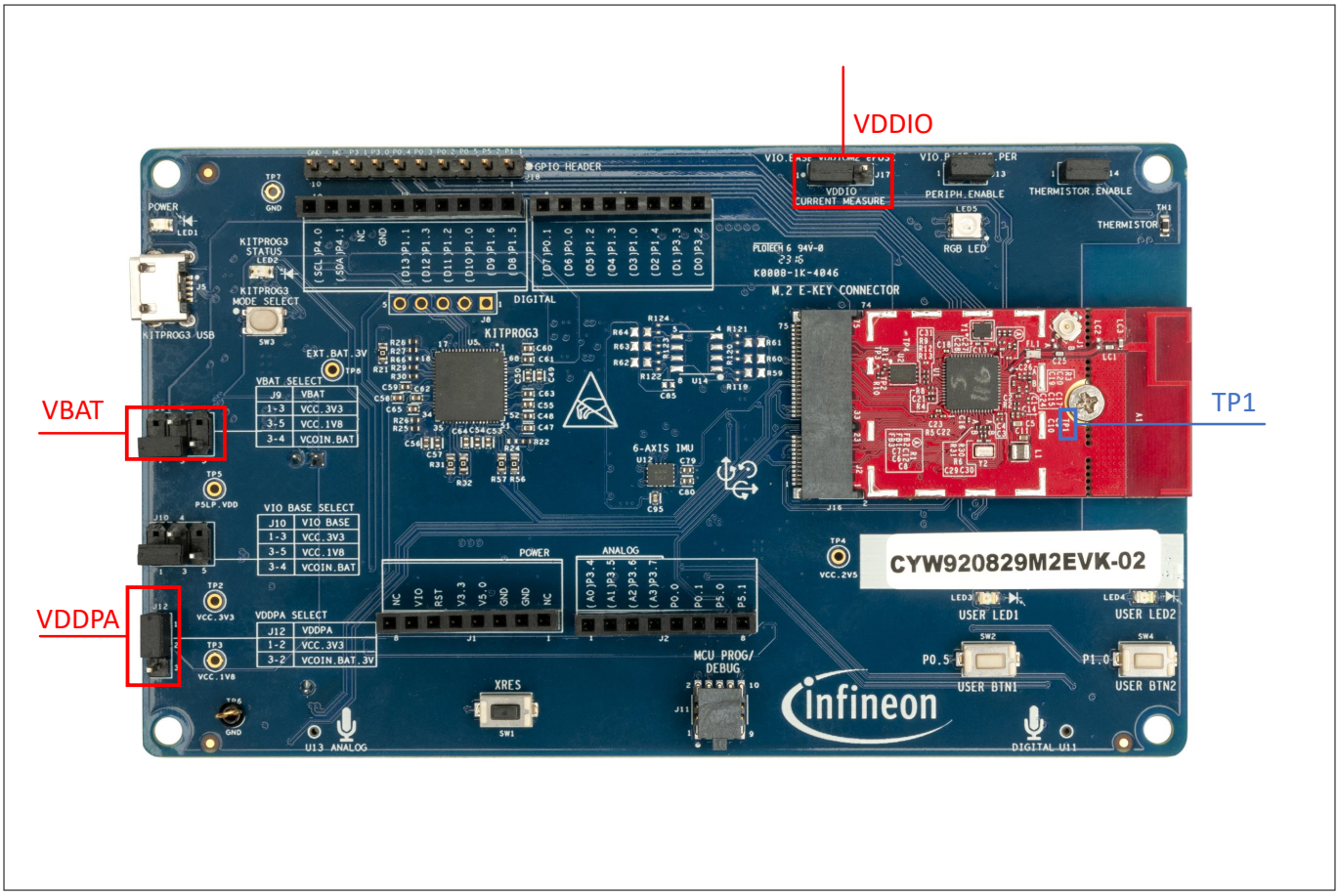

図 17 電流測定

- VIO_BASE ドメインで消費される電流を測定するには、ジャンパ J17 (VDDIO) のピン 1 とピン 2 の間に電流計を接続します。CYW920829M2EVK-02 では、外部フラッシュは VDDIO と電圧レギュレーターに接続されています。したがって、VDDIO で測定される電流は、選択した外部シリアルフラッシュおよび電圧レギュレーターに依存する場合があります。
- VBAT ドメインで消費される電流を測定するには、選択した電源に基づいて、J9.3 とピン J9.1, J9.4, J9.5 のいずれかに電流計を接続します。
- VDDPA ドメインで消費される電流を測定するには、選択した電源に基づいて、J12.3 と J12.1, J12.4 ピン, J12.5 のいずれかに電流計を接続します。
- VCCD 電圧を測定するには、GND を基準にして TP1 の両端に電圧計を接続します。これは、デバイスがディープスリープに入ったかどうかを確認するのに役立ちます。デバイスがディープスリープに入ると、TP1 の両端の電圧は 0.9 V です。CPU が ULP モードおよび LP モードの場合、TP1 の両端の電圧はそれぞれ 1.0 V と 1.1 V です。

6.2 平均消費電流

ここでは、低消費電力モードでの平均消費電流について説明します。各低消費電力モードで一定期間電流を測定し、平均消費電流を算出します。アクティブ モードの消費電力は、アクティブ ペリフェラルの数や動作クロック周波数など、さまざまな要因によって異なります。

6 電力測定

Bluetooth® LE GAP セントラルデバイスが推奨する接続間隔は、Bluetooth® LE GAP ペリフェラルとの接続を確立するために使用されます。他の値は GAP ペリフェラルによって要求される場合がありますが、GAP セントラル デバイスは引き続き接続パラメーターの決定を担当します。

接続間隔とペリフェラル レイテンシは、消費電力とデータスループットの間でトレードオフしながら、最適に選択する必要があります。例えば、接続間隔を長くする (4 秒など) と、電力使用量は減少しますが、スループットは低下します。したがって、接続間隔の値を小さくすることでデータスループットを増やすことができますが、そうすると消費電力も増加することに注意してください。

一部のペリフェラル デバイスでは、Bluetooth® LE スタックで最小接続間隔と最大接続間隔を選択できる場合があります。セントラルは、ペリフェラル デバイスの最小値と異なる値、もしくは場合によっては許容範囲を超える値を選択する代わりに、ペリフェラルによって提供される最大接続間隔値を使用して、範囲内の許容値を選択できます。

次の電流測定図では、黄色の線は VBAT 電流、緑の線は VDDPA、青の線は VDDIO を表しています。

6.2.1 スリープ モード



図 18 1 秒の接続間隔でのスリープ接続アイドル



図 19　　スリープ切断アイドル

6.2.2　　ディープ スリープ モード

消費電流は、デバイスがディープスリープにあるときに測定されます。図 22 に示すように、VBAT で 4.6 µA、VDDPA で 1.5 µA、VDDIO で 0.76 µA を消費し、アクティブモードよりも大幅に削減されます。

ディープスリープ時の接続アイドル状態の場合、図 20 に示すように、接続間隔が 300 ms の VBAT の消費電流は 35 µA です。接続間隔を長くすることで、接続アイドル時の消費電流をさらに低減できます。図 21 に示すように、ディープスリープの接続間隔を 300 ms から 1 秒に増やすと、平均消費電流は 18.8 µA に大幅に減少します。



図 20　　300 ms の接続間隔でのディープスリープ接続アイドル



図 21　　1 秒の接続間隔でのディープスリープ接続アイドル

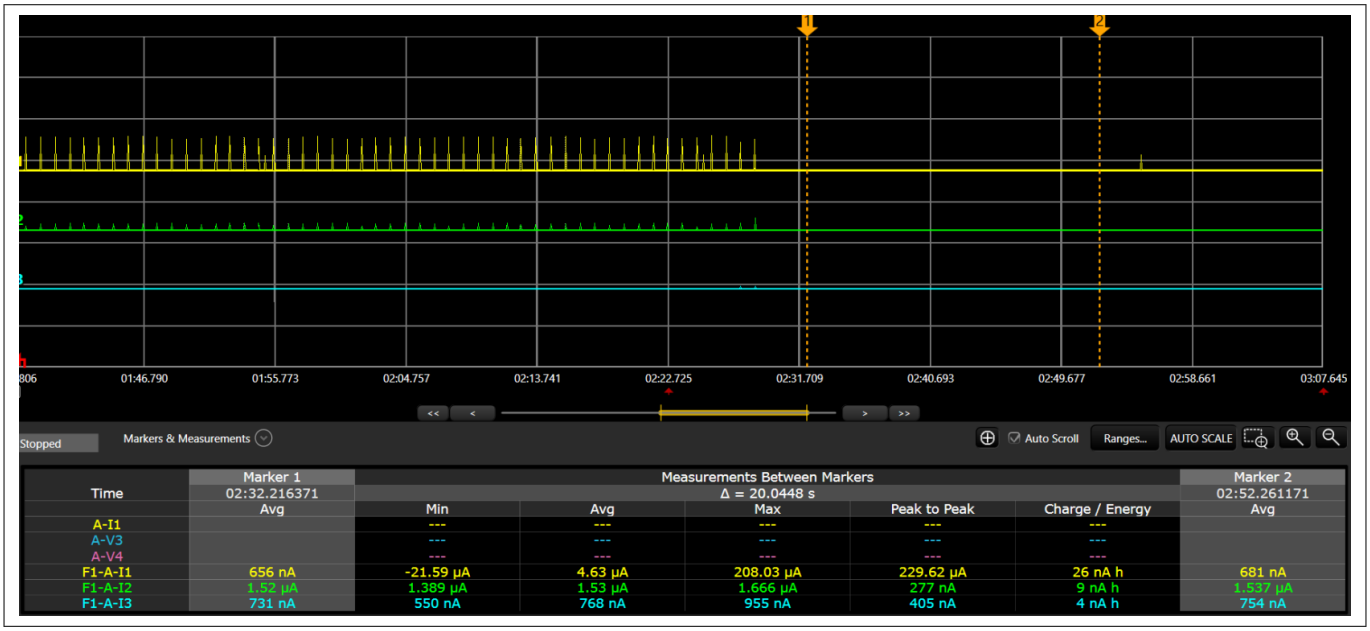

図 22　　ディープスリープ切断アイドル

6.2.3　ディープスリープ RAM モード

消費電流は、デバイスがディープスリープ RAM モードのときに測定されます。図 24 に示すように、VBAT で 4.1 µA、VDDPA で 1.5 µA、VDDIO で 0.77 µA を消費し、アクティブモードよりも大幅に低くなっています。

6 電力測定

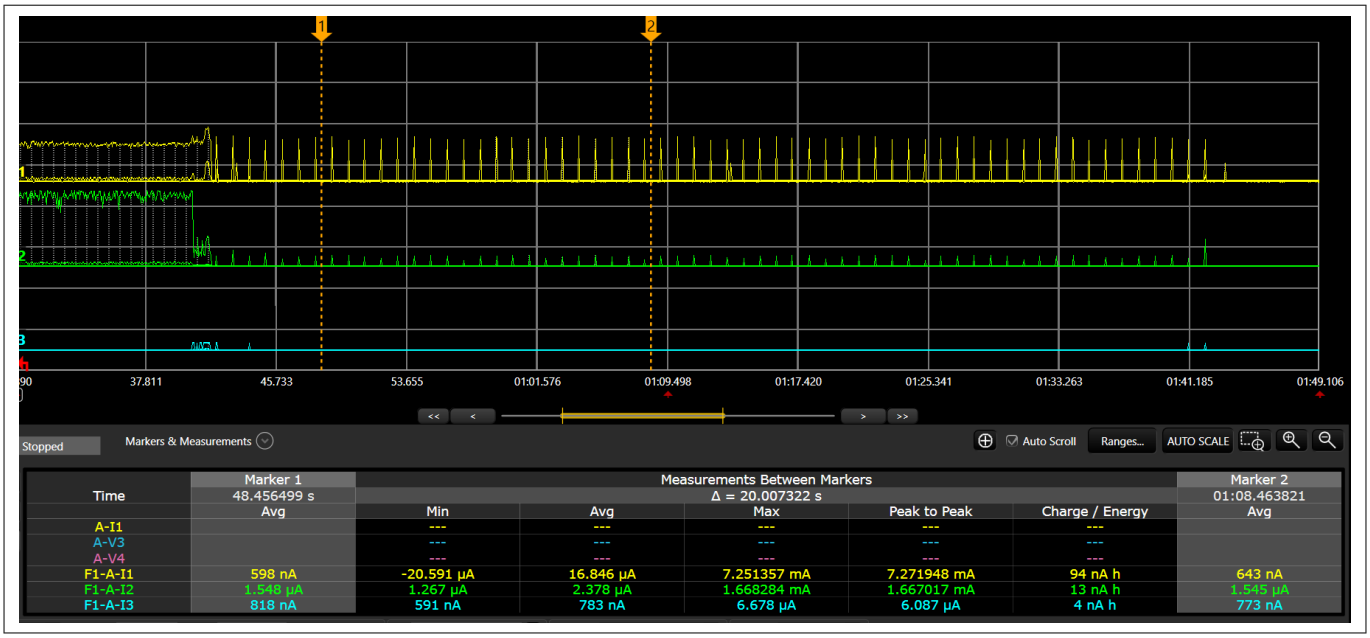

図 23　　1 秒の接続間隔でのディープ スリープ RAM 接続アイドル

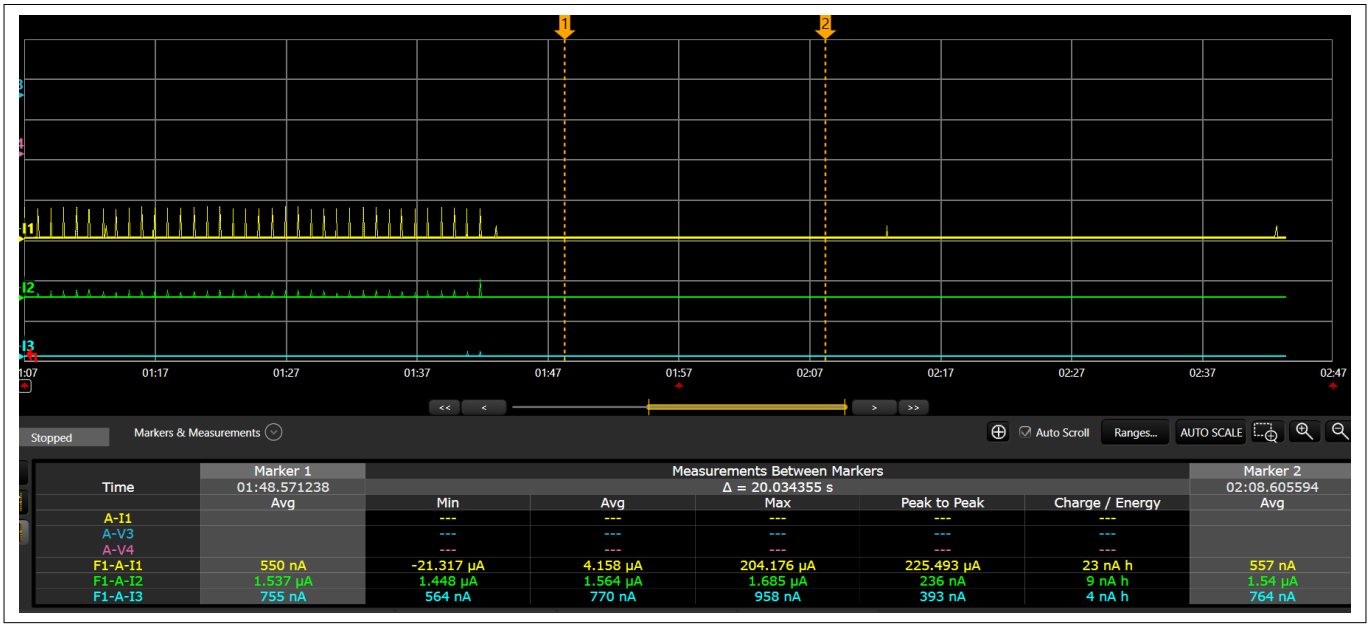

図 24　　ディープスリープ RAM の切断アイドル

6.2.4 ハイバネートモード

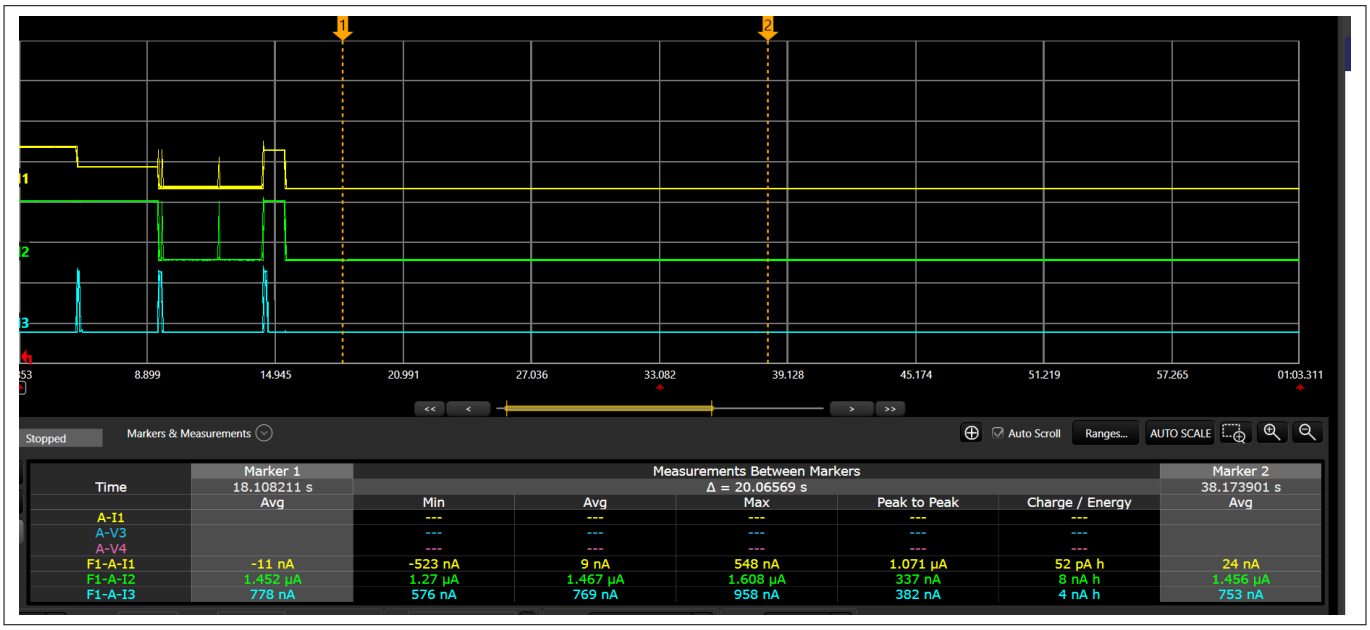

図 25 ハイバネート電流

注: 図 18 ～25 は、クロックソースとして WCO を使用してキャプチャされています。

6.3 さまざまな電力モードでの測定電流読み出し

表 8 WCO を使用した 20 秒間の接続時の平均電流読み出し

1 秒の接続間隔での接続アイドル

電力レール	スリープ	ディープスリープ	ディープスリープ RAM	ハイバネート
VBAT	2 mA	18.89 µA	16. 8µA	該当なし
VDDPA	56 µA	2.3 µA	2.3 µA	該当なし
VDDIO	45 µA	0.78 µA	0.78 µA	該当なし

表 9 WCO を使用した 20 秒間の接続なしの平均電流読み出し

切断アイドル

電力レール	スリープ	ディープスリープ	ディープスリープ RAM	ハイバネート
VBAT	1.9 mA	4.6 µA	4.1 µA	9 nA
VDDPA	55 µA	1.5 µA	1.5 µA	1.4 µA
VDDIO	44 µA	0.76 µA	0.77 µA	0.76 µA

表 10 PILO を使用した 20 秒間の接続有りの平均電流読み出し

1 秒の接続間隔での接続アイドル

電力レール	スリープ	ディープスリープ	ディープスリープ RAM	ハイバネート
VBAT	2 mA	20 µA	18.5 µA	該当なし

(続く)

表 10 (続き) PILO を使用した 20 秒間の接続有りの平均電流読み出し

1 秒の接続間隔での接続アイドル

電力レール	スリープ	ディープスリープ	ディープスリープ RAM	ハイバネート
VDDPA	56 µA	2 µA	2.1 µA	該当なし
VDDIO	45 µA	0.78 µA	0.78 µA	該当なし

表 11 PILO を使用した 20 秒間の接続なしの平均電流読み出し

切断アイドル

電力レール	スリープ	ディープスリープ	ディープスリープ RAM	ハイバネート
VBAT	1.9 mA	5.8 µA	5.3 µA	1.19 µA
VDDPA	55 µA	1.2 µA	1.3 µA	1.2 µA
VDDIO	44 µA	0.76 µA	0.76 µA	0.76 µA

参考資料

[1] サンプルコード

ModusToolbox™ソフトウェアのサンプルコード – サンプルコードの包括的なコレクションについては、このページをご覧ください。

[2] デバイス資料

AIROC™ CYW20829 Bluetooth® LE datasheet

[3] 開発キット

CYW920829M2EVK-02 キット ユーザーガイド

[4] ツールドキュメント

Eclipse IDE for ModusToolbox™ - インフィニオンの IoT 設計者向け IDE

[5] Arm®のドキュメント

Power Control System Architecture

改訂履歴

版数	発行日	変更内容
英語版**	-	本版は英語版のみの発行です。英語版の改訂内容: Initial release
**	2025-03-19	本版は英語版 002-38282 Rev. *A について、インフィニオン　DEVELOPER COMMUNITY の参画者によって日本語に翻訳されたドキュメントです。

商標

The Bluetooth® word mark and logos are registered trademarks owned by Bluetooth SIG, Inc., and any use of such marks by Infineon is under license.

Trademarks

All referenced product or service names and trademarks are the property of their respective owners.

Edition 2025-03-19

Published by

Infineon Technologies AG

81726 Munich, Germany

© 2025 Infineon Technologies AG

All Rights Reserved.

Do you have a question about any aspect of this document?

Email: erratum@infineon.com

Document reference
IFX-uoc1713252127398

重要事項

本手引書に記載された情報は、本製品の使用に関する手引きとして提供されるものであり、いかなる場合も、本製品における特定の機能性能や品質について保証するものではありません。本製品の使用の前に、当該手引書の受領者は実際の使用環境の下であらゆる本製品の機能及びその他本手引書に記された一切の技術的情報について確認する義務が有ります。インフィニオンテクノロジーズはここに当該手引書内で記される情報につき、第三者の知的所有権の不侵害の保証を含むがこれに限らず、あらゆる種類の一切の保証および責任を否定いたします。

本文書に含まれるデータは、技術的訓練を受けた従業員のみを対象としています。本製品の対象用途への適合性、およびこれら用途に関連して本文書に記載された製品情報の完全性についての評価は、お客様の技術部門の責任にて実施してください。

警告事項

技術的要件に伴い、製品には危険物質が含まれる可能性があります。当該種別の詳細については、インフィニオンの最寄りの営業所までお問い合わせください。

インフィニオンの正式代表者が署名した書面を通じ、インフィニオンによる明示の承認が存在する場合を除き、インフィニオンの製品は、当該製品の障害またはその使用に関する一切の結果が、合理的に人的傷害を招く恐れのある一切の用途に使用することはできないこと予めご了承ください。