TRAVEO™ T2G MCU のセキュアなファームウェア オーバーザエア (FOTA) アップデート

本書について

適用範囲と目的

AN229058 は、TRAVEO™ T2G MCU における「セキュア」で信頼性の高いファームウェア オーバーザエア (FOTA) アップデートの開発に関連するさまざまな手順を説明しています。このドキュメントでは、無線でアップデートを実行するためのワイヤレスインタフェースの設定については説明しません。しかし、TRAVEO™ T2G MCU の論理フラッシュバンクを新しいイメージでアップデートする方法と、(TRAVEO™ T2G MCU のセキュアブートメカニズムを使用して) 認証に成功した場合、ブートコードが新しいイメージにジャンプできる信頼性について説明します。

対象者

このドキュメントは、TRAVEO™ T2G MCU のファームウェア オーバーザエア アップデートを使用するすべての人を対象とします。

目次

目次

1 はじめに

このアプリケーションノートでは、TRAVEO™ T2G MCU で FOTA (firmware over-the-air) アップデートを安全に実行する方法について説明します。このシリーズは、強化された拡張「セキュア」ハードウェア (eSHE) またはハードウェアセキュリティモジュール (HSM) 、CAN FD、メモリ 、アナログおよびデジタル周辺機能を備えた Arm® Cortex®-M CPU がシングルチップに含まれています。

自動車組込みシステムでは、FOTA アップデートは、デバイスでワイヤレスファームウェアのアップグレードを実行するのに役立つリモートソフトウェア管理テクノロジです。特にシステムが重大なバグ修正、新機能の追加、既存機能の削除などを要求する場合、フィールドに出た後のデバイスファームウェアのアップグレードは不可欠な場合があります。

FOTA の完全な実装は、デバイスアーキテクチャ (ここでは TRAVEO™ T2G デバイス) に大きく依存します。TRAVEO™ T2G MCU の内部コードフラッシュアーキテクチャは、主に FOTA アプリケーションを対象とするデュアルバンクモードをサポートします。このアーキテクチャにより、FOTA アップデートが行われている間、既存のアプリケーションが中断されることはなく、新しいアプリケーションは安全に起動されます。FOTA アップデートでは、新しいアップデートが何らかの理由で破損または中断した場合、古いファームウェアへのロールバックメカニズムを義務付ける必要があります。

このアプリケーションノートは、主にデュアルバンクマネージャコードの開発に焦点を当てています。これは、新しいアプリケーションイメージの有効性を検証 (暗号化認証機能を使用して) するのに役立ちます。そして、フラッシュバンクおよびリマップレジスタを正しく変更して、検証されたイメージが起動することを確実にします。

これは高度なアプリケーションノートです。続行する前に、以下のアプリケーションノートを参照してください。

- AN220242 - TRAVEO™ T2G ファミリのフラッシュアクセス手順
- AN228680 - Secure system configuration in TRAVEO™ T2G family

2 FOTA の概要

図 1 に、自動車ネットワークにおける標準的な FOTA システムの例を示します。

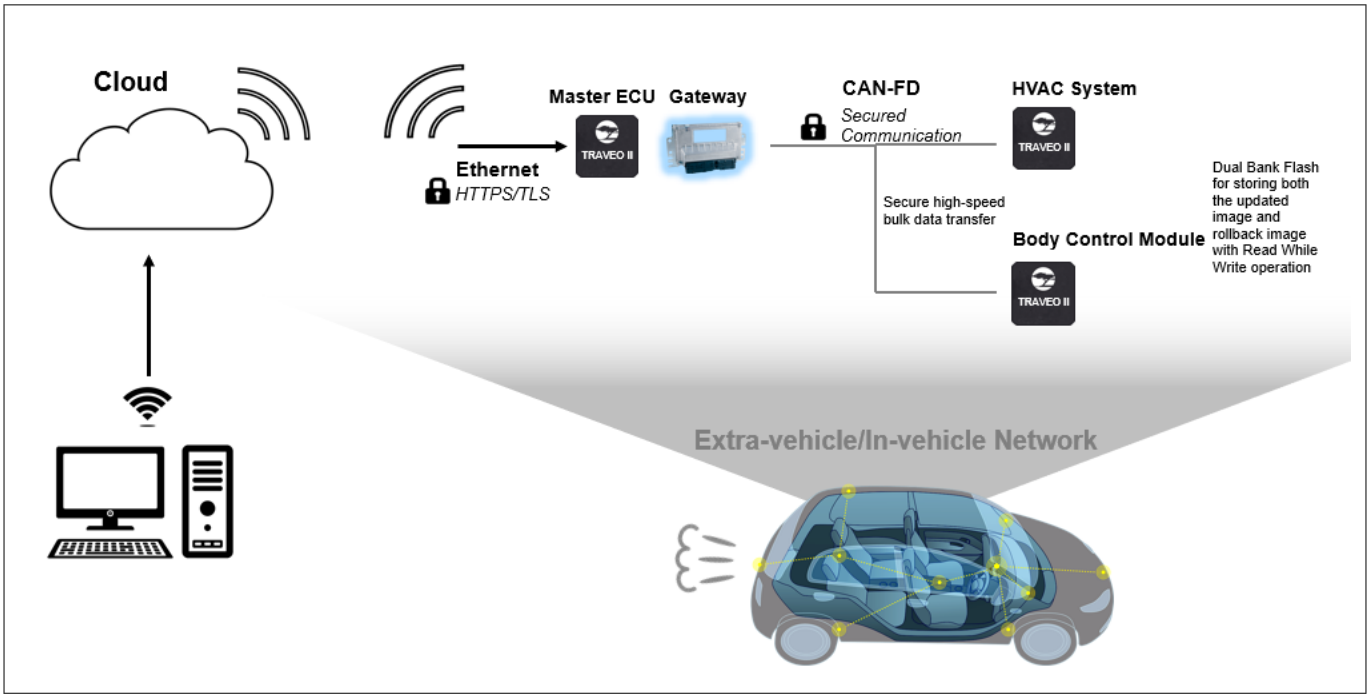

図 1 標準的な FOTA システム

マスタ電子制御ユニット (ECU) は、クラウドと通信して新しいイメージを受信します。次に、ゲートウェイ(CAN-FD などに基づく) を実装して、HVAC システムやボディコントロールモジュールなどの他の TRAVEO™ T2G サブシステムのイメージをアップデートできます。この通信は、暗号化と復号化、および保護ユニット (ソフトウェア保護ユニット (SWPU) など) を使用して保護することもできます。これにより、フラッシュ書き込みの権限を持つ ECU のみがフラッシュアップグレードを実行できます。

TRAVEO™ T2G デバイスは真の FOTA に対応します。これは、次のことを意味します。

- ソフトウェアイメージのアップデートはバックグラウンドで行われます。アプリケーションサービスの中断はありません。
- 失敗した場合、またはアプリケーションの必要に応じて、アップデートをロールバックできます。

この真の FOTA は、以下をサポートする TRAVEO™ T2G デバイスで可能です。

- デュアルバンクフラッシュ
- プログラム中 (書き込み) のソフトウェアの実行 (読み出し) を許可する、Read-while-write メモリ。
- 複数レベルのセキュリティと「セキュアブート」

2.1 従来の FOTA (「セキュアブート」なし) の課題

図 2 に、TRAVEO™ T2G デバイスの 8MB メモリのコードフラッシュメモリマッピングを示します。

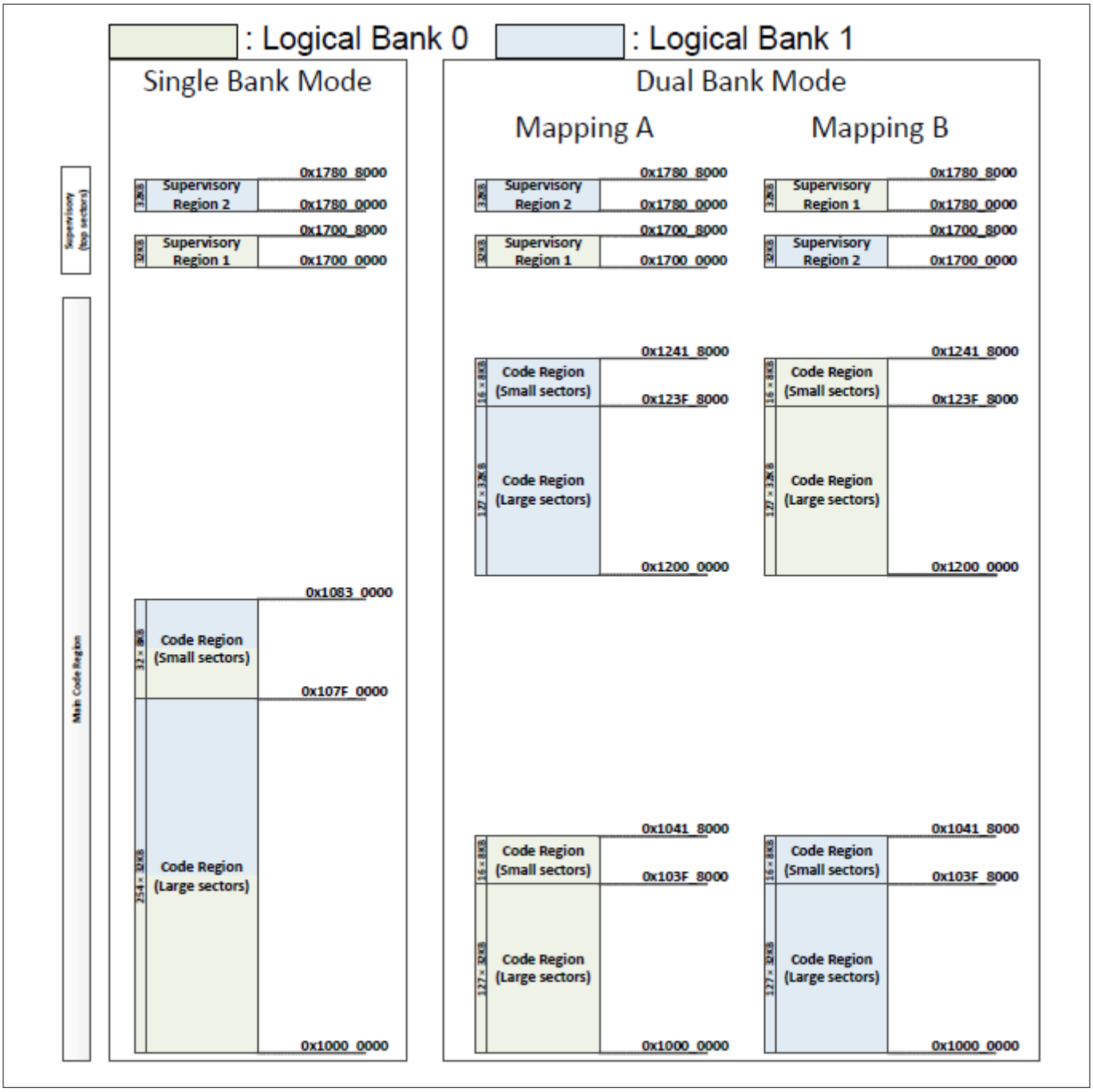

図 2　　8MB コードフラッシュのコードフラッシュメモリマッピング

FOTA アーキテクチャはデュアルバンクモードを使用します。FOTA のアップグレード中に電源障害またはその他のハードウェアの問題が発生すると、TRAVEO™ T2G デバイスは破損状態になる可能性があります。例えば、アプリケーションがマッピング B でコードを実行しているときに FOTA のリクエストがあった場合、新しいイメージを論理フラッシュバンク 0 に書き込む必要があります。ここで、最初のいくつかのセクタのアップデート中に電源障害または破損が発生した場合、TRAVEO™ T2G デバイスは次のリセット後にロックアップまたは破損状態になる可能性があります。これは、Flash ブートは常にシングルバンクモードで実行され、デフォルトでは TOC2 (TOC2 の詳細を参照) 内の「最初のユーザアプリケーションオブジェクトのアドレス」で指定された(破損の可能性のあるベクタテーブルを持つ)アドレスにジャンプするためです。このアプリケーションノートで述べるテクニックに基づくセキュアブートメカニズムを実装すると、この問題を軽減するのに役立ちます。

3 TRAVEO™ T2G のバンク切替えメカニズム

TRAVEO™ T2G ファミリの MCU において、OTA 機能は FLASHC_FLASH_CTL レジスタの 2 つの別々のビットによって処理されます。MAIN_BANK_MODE ビットは、フラッシュバンクモードをシングルバンクモード、またはデュアルバンクモードに設定します。MAIN_MAP ビットは、デュアルバンクモード時のマッピングを A、または B に設定します。これら両ビットともリセットでクリアされます。ROM ブートおよび Flash ブートは、これらの構成に触れません。つまり、TRAVEO™ T2G ファミリの MCU は、Arm® Cortex®-M0+プログラムが起動する前に、常にシングルバンクモードで起動します。アプリケーションは、必要に応じてデュアルバンクとリマップ機能を構成する必要があります。

***注**: MAIN_BANK_MODE ビットは FLASHC_FLASH_CTL レジスタの一部です。FLASHC_FLASH_CTL.MAIN_BANK_MODE = 1 の場合、デュアルバンクモードが有効になります。*

***注**: MAIN_MAP ビットも FLASHC_FLASH_CTL レジスタの一部です。このビットが"0"の場合はマッピング A が使用され、このビットが"1"の場合はマッピング B が使用されます。*

FLASHC_FLASH_CTL レジスタの詳細については、レジスタテクニカルリファレンスマニュアル (TRM) を参照してください。

本アプリケーションノートで詳しく説明されるデュアルバンクマネージャソフトウェアは、特定の基準に基づいてバンクモードとマッピングレジスタを構成することに注意してください。このデュアルバンクマネージャソフトウェアは、ユーザの SFlash メモリにプログラムされ、通常の Flash ブートオペレーション後に CM0+によってトリガされます。

バンク切り替えの詳細は、AN220242 - TRAVEO™ T2G ファミリのフラッシュアクセス手順を参照してください。

また、MAIN_BANK_MODE および MAIN_MAP ビットは、フラッシュアドレス空間の突然の変更を伴うため、コードフラッシュまたは SFlash から実行されるコードによって変更してはならないことに注意してください。このアプリケーションノートで説明する実装では、デュアルバンクマネージャがコードを SFlash から SRAM にコピーし、SRAM から実行されるコードがこれらのビットを構成します。

4 「セキュアブート」の概要

ここでは、TRAVEO™ T2G MCU の起動シーケンスを紹介します。TRAVEO™ T2G MCU のセキュリティ機能、さまざまなライフサイクルステージ、および「セキュアブート」シーケンスの実装の詳細は、AN228680 - Secure system configuration in TRAVEO™ T2G family を参照してください。

TRAVEO™ T2G MCU のブートシーケンス (図 3) は、さまざまなライフサイクルステージに対して実装される ROM ブートコードと Flash ブートコードに基づきます。図 3 に、CM0+のオペレーションがリセットから開始される方法を示します。リセット後、CM0+は ROM ブートから実行を開始します。ROM ブートは SFlash を検証します。SFlash の検証が完了すると、Flash ブートにジャンプし、保護状態の必要に応じて DAP を構成します。データとコードが存在するメモリタイプを表す色分けに注意してください。

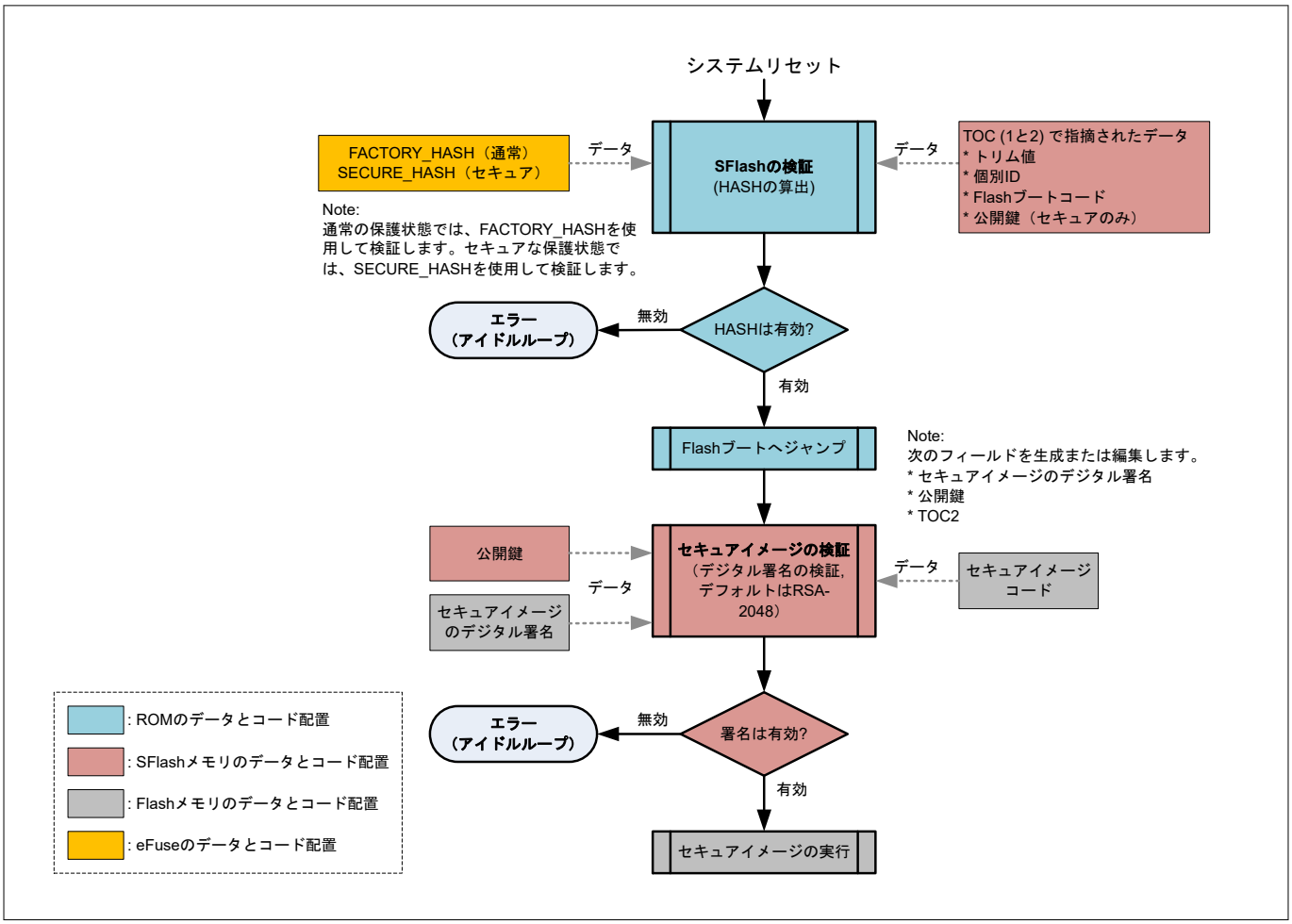

図 3 TRAVEO™ T2G ブートシーケンス

次に、Flash ブートは、TOC2 にリストされている最初のアプリケーションを検証し、検証されている場合は、そのエントリーポイントにジャンプします。このイメージは「セキュア」イメージです。「セキュア」イメージが無効または破損していることが判明した場合、CPU はアイドルループにジャンプし、デバイスがリセットされるまでアイドルループに留まります。

4.1 セキュアな FOTA の「セキュアブート」メカニズム

このアプリケーションノートで提案するセキュア FOTA アーキテクチャとサンプルコードは、「セキュアブート」プロセスに対して少し異なるメカニズムを使用します。提案するアーキテクチャとソフトウェアは、以下のアップデート後、TRAVEO™ T2G デバイスのライフサイクルステージを「SECURE」または「SECURE_W_DEBUG」に移行することを前提とします。

- ユーザの SFlash 行 (4 から 7 行) は、デュアルバンクマネージャコードでプログラムされます。SFlash 行の詳細は、TRAVEO™T2G MCU プログラミング仕様のドキュメントを参照してください。
- パブリックキーは SFlash 行 (50 から 55 行) にプログラムされています。
- TOC2 は SFlash の 62 行にプログラムされています。

TRAVEO™ T2G デバイスが SECURE ライフサイクルステージに移行すると、SFlash 行をプログラムできなくなることに注意してください。また、SECURE ライフサイクルステージに移行すると、「SECURE_HASH」が計算され、関連する e-Fuse ビットが切断されます。

4.1.1 デュアルバンクマネージャ

このアプリケーションノートは、主にデュアルバンクマネージャコードの開発に焦点をあてています。これは、CM0+のアプリケーションイメージの検証と、それに応じてフラッシュバンクおよびリマップレジスタをアップデートし、正しいフラッシュバンクからコードを実行することに役立ちます。このデュアルバンクマネージャのソフトウェアの詳細と完全な実装は、デュアルバンクマネージャを参照してください。このデュアルバンクマネージャコードは、SFlash の一部としてユーザ SFlash 行に実装されていることに注意してください (SFlash の 4 行から 7 行)。

デュアルバンクマネージャは、Cypress Secure Application Format (CySAF) に従って開発されていますが、追加のデジタル署名やデジタル署名に限定されたスペースはありません。これは、利用可能な SFlash ユーザ行に他のユーザ固有のデータを格納する追加の利点を提供します。このデュアルバンクマネージャコードは、TOC2 の追加オブジェクトとして追加され、ROM ブートによって認証されます。TOC2 構造の詳細については、TOC2 の詳細を参照してください。

4.1.2 FOTA の ROM ブート機能

TRAVEO™ T2G デバイスが SECURE ライフサイクルステージにある場合、ROM ブートは E-Fuse の「SECURE_HASH」に対して TOC2 の内容も検証します。デュアルバンクマネージャコードは TOC2 の追加オブジェクトとして追加され、その信頼性は ROM ブートコードによって自動的に検証されます。デュアルバンクマネージャコードは SFlash メモリの一部で、SECURE ライフサイクルステージを超えてプログラムできないため、デュアルバンクマネージャが破損する可能性も最小限に抑えられます。ROM ブートがすべての必要なオブジェクトを正常に検証すると、ROM ブートコードは Flash ブートコードにジャンプします。

4.1.3 FOTA の Flash ブート機能

このアプリケーションノートで提案されているアーキテクチャでは、Flash ブートによるアプリケーション認証制御は無効です。これは TOC2_FLAGS.APP_AUTH_CTL = 1 を意味します。また、デュアルバンクマネージャアドレスは、TOC2 の CM0+の最初のアプリケーションオブジェクトとして提供されます。したがって、Flash ブートコードはデュアルバンクマネージャに直接ジャンプします。デュアルバンクマネージャコードは ROM ブートによってすでに認証されているため、認証が無効になってもリスクはありません。

4.1.4 FOTA のデュアルバンクマネージャ機能

このアプリケーションノートで提案されるアーキテクチャでは、CM0+イメージは CySAF に従って構築されています。これは、このイメージに関連付けられたデジタル署名 (秘密鍵を使用して暗号化された) と SFlash の関連付けられた公開鍵があることを意味します。デュアルバンクマネージャは、この「セキュア」イメージの有効性をチェックします。最初に、FOTA を介してアップデートされる最新のイメージの有効性をチェックします。新しいイメージが無効の場合は、古いイメージの有効性を確認します。デュアルバンクマネージャは、次にフラッシュバンクとリマップレジスタをアップデートして正しい論理バンク/マッピングを設定し、CM0+アプリケーションイメージにジャンプします。両方のイメージが無効の場合、デュアルバンクマネージャは無限ループになります。

4.1.5 FOTA の CM0+アプリケーション機能

デュアルバンクマネージャによって認証された CM0+イメージは、CM4 または CM7 アプリケーションイメージをさらに認証できます。これは CM0+アプリケーションイメージによって実行される必要があり、TRAVEO™ T2G ブートコードまたはデュアルバンクマネージャによって自動的に実行されることはありません。

4.1.6 セキュアな FOTA の信頼の連鎖 (Chain of Trust:CoT)

ステップ 4.1.2 から 4.1.5 をまとめると、デバイスブート中の CoT は、「SECURE」保護状態で以下の順で進行します。

表 1 に、セキュアな FOTA のブートフローに関係する各コンポーネントの保護メカニズムのまとめを示します。

表 1 セキュアな FOTA に関連するコンポーネント

ブートコンポーネント	起動元	保護メカニズム	機能
ROM ブート	リセット	継承 (これは設計により固定されます)	ROM ブートは、Flash ブートとデュアルバンクマネージャコード (TOC2 で追加オブジェクトとして追加される) を認証し、Flash ブートにジャンプします。
Flash ブート	ROM ブート	ROM ブートによって実行される「SECURE_HASH」チェック。	Flash ブートは、デュアルバンクマネージャ機能をトリガします。
デュアルバンクマネージャ	Flash ブート	ROM ブートによって実行される「SECURE_HASH」チェック。このコードは TOC2 に追加オブジェクトとして追加されています。	デュアルバンクマネージャは、FOTA を介して新しい CM0+イメージを検証し、それに応じてフラッシュバンク/マッピングレジスタをアップデートします。次に、デュアルバンクマネージャは CM0+イメージにジャンプします。
CM0+イメージ	デュアルバンクマネージャ	デュアルバンクマネージャは、SFlash にプログラムされた公開鍵を使用して CM0+イメージのデジタル署名認証をチェックします。	CM0+イメージは、CM4/CM7 アプリケーションイメージの信頼性を検証でき、有効な場合はそれにジャンプできます。
CM4/CM7 イメージ	CM0+イメージ	CM0+イメージは、SFlash にプログラムされた公開鍵を使用して CM4/CM7 イメージのデジタル署名認証をチェックします。	CM4/CM7 イメージは、メインのアプリケーションイメージです。

4.1.7 ワークフラッシュマーカー

FLASHC_FLASH_CTL レジスタの MAIN_BANK_MODE ビットと MAIN_MAP ビットは揮発性であるため、これらのビットの値はリセット後に保持されません。したがって、デュアルバンクマネージャでこのレジスタをアップデートするには、一部の不揮発性メモリのサポートが必要です。ワークフラッシュの小セクタ (128 バイト) は、この目的に適したメモリです。

このアプリケーションノートで説明される実装では、ワークフラッシュの最初の小セクタが使用されます。このセクタは FOTA 用に予約されており、FOTA アップデートが成功するたびに消去および書き込みされるため、このセクタに他の情報は保存されないことに注意してください。このセクタを意図しない消去やプログラムから保護するよ

うに、SWPU オブジェクトを構成することを推奨します。マップ A で FOTA が完了すると、このワークフラッシュセクタの最初の 32 ビットが 0xAAAAAAAA (マジックワード) にアップデートされ、マップ B で FOTA が完了すると 0xFFFFFFFF にアップデートされます。

ワークフラッシュのアップデート中に FOTA アップデートが中断されると、ワークフラッシュセクタが消去される可能性があります。消去されたワークフラッシュセクタの内容は予測できません。消去状態のワークフラッシュの読み出し中に ECC エラーおよび関連するバス障害を回避するには、FLASHC_FLASH_CTL を使用します。WORK_ERR_SILENT は、ワークフラッシュを読み出す前にデュアルバンクマネージャで設定されます。このビットは、アプリケーションによって後に有効にされることに注意してください。

***注**:* *このワークフラッシュ マーカーは、FLASHC_FLASH_CTL レジスタの MAIN_MAP ビットに適用されるマッピングのみを示します。アプリケーションの開始アドレス (認証チェック用) は SFlash に保存されます。アドレスは CY_APP_START_ADDR_LB と CY_APP_START_ADDR_UB です。付録 8.1 を参照してください。*

5 デュアルバンクマネージャ

図 4 に、デュアルバンクマネージャのフローチャートを示します。`flashmarker` の詳細については、付録 8.4.1 を参照してください。デュアルバンクマネージャは、CySAF フォーマットに従って開発され、SFlash の位置 0x17000800 に配置される必要があることに注意してください。これは、必要なリンカスクリプトのサポートによって実行できます。

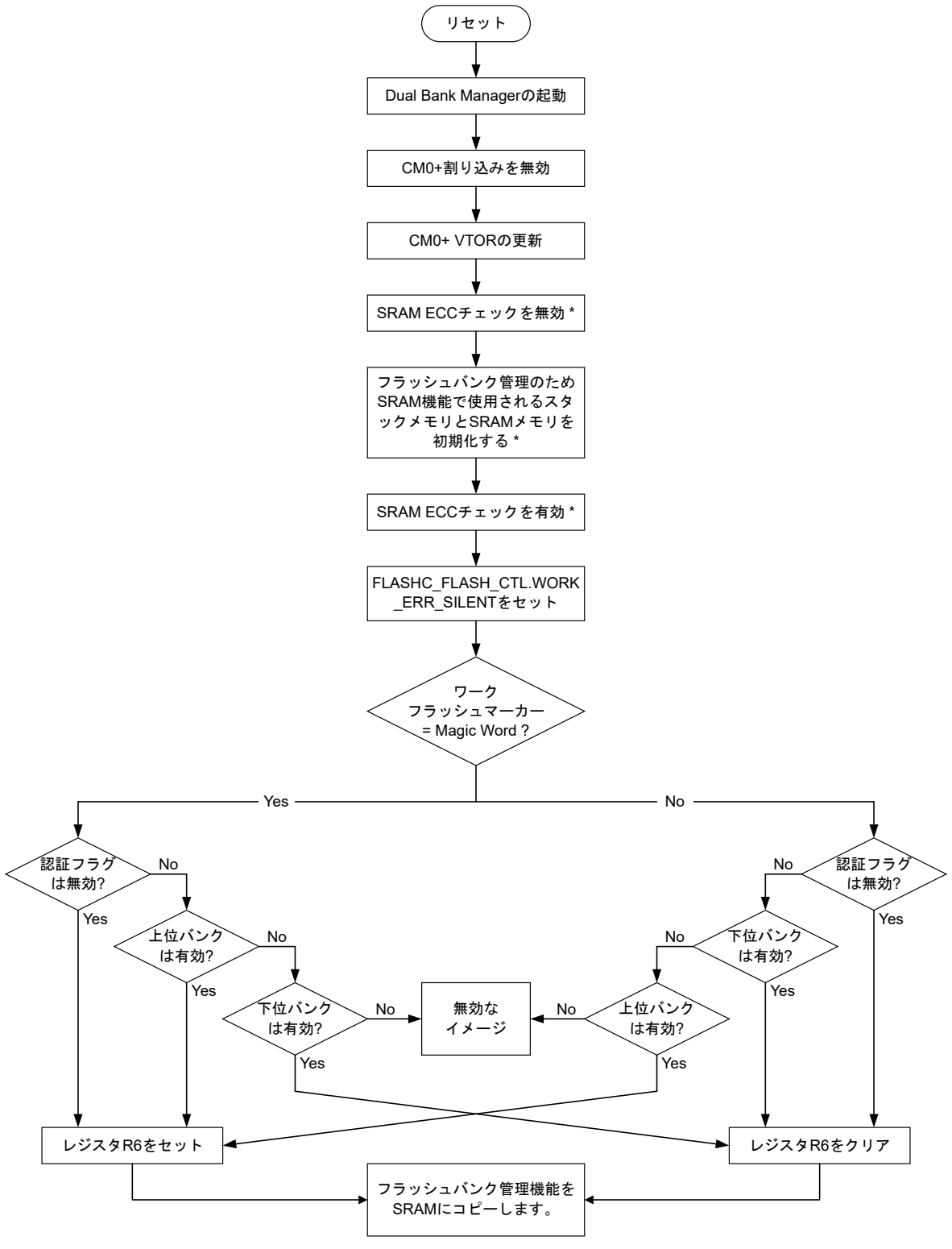

図 4　　デュアルバンクマネージャのフローチャート

5.1 割込みの無効化

最初に、グローバル割込みが無効にされます。

```
; Disable global interrupts
     CPSID I
```

5.2 VTOR のアップデート

CM0+のベクトルテーブルオフセットレジスタ (VTOR) は、デュアルバンクマネージャのベクタテーブルアドレスにアップデートされます。割込みが無効になっているため、これはデュアルバンクマネージャの実行中に発生したハードフォールトを検知するのに役立ちます。その他の割込み/例外ハンドラは、ベクタテーブルでゼロに固定されます。ベクタテーブルについては、付録 8.2 を参照してください。

```
; Update Vector Table Offset Register with address of SFlash vector table
     LDR  r0, =__sflashvector_table
     LDR  r1, =VTOR
     STR  r0, [r1]
     DSB
```

5.3 SRAM ECC チェックの無効化

デュアルバンクマネージャで使用されるスタック領域は、関数が呼び出される前に初期化されます。この領域の初期化は、32 ビット書き込みのみを許可する CM0+によって行われるため、64 ビット ECC に基づく SRAM を備えた TRAVEO™ T2G デバイスは、Read-modify-write オペレーションを実行します。そのため、SRAM の ECC チェックは一時的に無効になります。

```
; CM0+ bus width is 32-bit, but SRAM is built with 64-bit based ECC on Traveo T2G parts with
CM7 core
; Set CPUSS->RAMx_CTL0.ECC_CHECK_DIS bits to avoid causing unintentional ECC faults during
startup while SRAM ECC has not been initialized yet
; Done for all SRAM macros as the stack/function could be placed any where based on linker
script; this could also be modified only to specific SRAM macros
#if (defined (tviibh4m) || defined (tviibh8m) || defined (tviic2d6m) || defined (tviic2d4m))
     MOVS r4, #1
     LSLS r4, r4, #19
     LDR  r1, =CPUSS_RAM0_CTL0
     LDR  r2, [r1]
     ORRS r2, r4
     STR  r2, [r1]
     LDR  r1, =CPUSS_RAM1_CTL0
     LDR  r2, [r1]
     ORRS r2, r4
     STR  r2, [r1]
     LDR  r1, =CPUSS_RAM2_CTL0
     LDR  r2, [r1]
     ORRS r2, r4
     STR  r2, [r1]
#endif
```

5.4 スタックの初期化

スタックメモリはゼロに初期化されます。スタックのサイズは、STARTUP_STACK_SIZE_DOUBLE_WORDS マクロによって構成されます (付録 8.1 を参照)。

```
; Initialize ECC of startup stack
     MOVS r0, #0 ; clear value
     MOVS r1, #0 ; clear value
     LDR  r2, flashmarkers+28

startup_stack_ecc_init_loop:
     STM  r2!, {r0, r1}
     CMP  r2, sp
     BNE  startup_stack_ecc_init_loop
```

5.5 SRAM 関数メモリ領域の初期化と ECC チェックの有効化

デュアルバンクマネージャ機能は、フラッシュバンクを構成するのに必要なコードを SRAM にコピーします。コピーは 32 ビットの書き込みで行われ、64 ビット ECC に基づく SRAM を搭載した TRAVEO™ T2G デバイスの場合も、この領域はゼロに初期化されます。その後、ECC チェックが再び有効になります。

```
; For Traveo T2G devices with 64-bit based ECC SRAM, initialize ECC of the region where SRAM
function shall be copied later from SFlash
#if (defined (tviibh4m) || defined (tviibh8m) || defined (tviic2d6m) || defined (tviic2d4m))

     LDR  r2, flashmarkers
; Subtract 1 from R2 due to Thumb alignment
     SUBS r2, 1
     LDR  r3, flashmarkers+4

; loop to clear the SRAM
SRAM_function_ecc_init_loop:
     STR  r1, [r2,r0]
     ADDS r0, #8
     CMP  r0, r3
     BCC  SRAM_function_ecc_init_loop

; Clear CPUSS->RAMx_CTL0.ECC_CHECK_DIS bits to enable ECC check/correction; r4 already has the
bit position.

     LDR  r1, =CPUSS_RAM0_CTL0
     LDR  r2, [r1]
     BICS r2, r4
     STR  r2, [r1]
     LDR  r1, =CPUSS_RAM1_CTL0
     LDR  r2, [r1]
     BICS r2, r4
     STR  r2, [r1]
     LDR  r1, =CPUSS_RAM2_CTL0
     LDR  r2, [r1]
     BICS r2, r4
     STR  r2, [r1]
#endif
```

5.6 ワークフラッシュバスエラーの無効化

詳細については、ワークフラッシュマーカーを参照してください。

```
; Load the address for FLASHC_FLASH_CTL
     LDR  r4, flashmarkers+8

; Set FLASHC_FLASH_CTL.WORK_ERR_SILENT
; Reading the Work Flash in erased state can cause bus fault when
FLASHC_FLASH_CTL.WORK_ERR_SILENT is 0
     MOVS r1, #1
     LSLS r1, r1, #22
     ORRS r0, r1
     STR  r0, [r4]
     LDR  r0, [r4]
```

5.7 疑似コード

以下は、デュアルバンクマネージャによって実行される次のステップの疑似コードです。疑似コードの Authentication_Flag は、付録 8.1 にリストされている AUTHENTICATION_CHECK マクロであり、TOC2_FLAGS.APP_AUTH_CTL とは異なることに注意してください。

```
if (WorkFlash_Marker == MAGIC_WORD)
{
    if (Authentication_Flag == DISABLE)
    {
        set reg
        RAM func (MAP_B)
    }
    else
    {
        if (UpperImage == VALID)
        {
            set reg
            RAM func (MAP_B)
        }
        else if (LowerImage == VALID)
        {
            clear reg
            RAM func(MAP_A)
        }
        else
            ImageInvalid
    }
}
else
{
    if (Authentication_Flag == DISABLE)
    {
        clear reg
        RAM func(MAP_A)
    }
    else
    {
        if (LowerImage == VALID)
        {
            clear reg
            RAM func(MAP_A)
        }
        else if (UpperImage == VALID)
        {
            set reg
            RAM func (MAP_B)
        }
        else
            ImageInvalid
    }
}
```

5.7.1 マジックワードチェック

ワークフラッシュアドレスを読み、マジックワードが含まれているかどうかを確認します。

```
MagicWordCheck:

; Check if the magic word is present in Work Flash address and branch to appropriate label

    LDR  r2, flashmarkers+12   ; Load the address of WorkFlash marker
    LDR  r0, [r2]              ; Load the marker value to r0
    LDR  r1, flashmarkers+16   ; Load the magic word
    LDR  r3, flashmarkers+20   ; Load the flag for authentication check
    LDR  r4, flashmarkers+24   ; Load the AUTHENTICATION_DISABLE for comparison
    CMP  r0, r1
    BNE  Check_MapA_LB
```

5.7.2 認証チェック

認証フラグが有効になっている場合 (付録 8.4.1 を参照)、Cy_FB_VerifyApplication 関数を使用して、マップ A またはマップ B アプリケーションに有効なデジタル署名があるかどうかを確認します。_checkLB および_checkUB 関数の詳細については、デジタル署名検証機能関数を参照してください。

```
Check_MapB_UB:
; Check if the authentication check is disabled. If disabled directly jump to Auth_Skip_MapB.
If enabled, perform the authentication.
        CMP  r3, r4
        BEQ  Auth_Skip_MapB

; Check if the result of authentication of upper bank image is successful, if not jump to
Check_MapB_LB and check lower bank
        BL   _checkUB
        CMP  r0, #1
        BNE  Check_MapB_LB

Auth_Skip_MapB:
; Jump to the FlashtoRAM_copy function with register r6 set
        MOVS r6, #1
        B    FlashtoRAM_copy

Check_MapB_LB:
; Check if the result of authentication of lower bank image is successful, if not jump to
ImageInvalid, and no application is launched.
        BL   _checkLB
        CMP  r0, #1
        BNE  ImageInvalid

; Jump to the FlashtoRAM_copy copy function with register r6 cleared
        MOVS r6, #0
        B    FlashtoRAM_copy

Check_MapA_LB:

; Check if the authentication check is disabled. If disabled directly jump to Auth_Skip_MapA.
If enabled, perform the authentication.
        CMP  r3, r4
        BEQ  Auth_Skip_MapA

; Check if the result of authentication of lower bank image is successful, if not jump to
Check_MapA_UB and check upper bank
        BL   _checkLB
        CMP  r0, #1
        BNE  Check_MapA_UB

Auth_Skip_MapA:
; Jump to the FlashtoRAM_copy function with register r6 cleared
        MOVS r6, #0
        B    FlashtoRAM_copy

Check_MapA_UB:
```

```
; Check if the result of authentication of upper bank image is successful, if not jump to
ImageInvalid, no application is launched.
        BL   _checkUB
        CMP  r0, #1
        BNE  ImageInvalid

; Jump to the FlashtoRAM_copy function with register r6 set
        MOVS r6, #1
        B    FlashtoRAM_copy
```

5.7.3 認証の失敗

あるバンクのイメージの認証が失敗した場合、デュアルバンクマネージャは別のバンクのイメージをチェックします。両方のイメージが無効な場合、デュアルバンクマネージャは無限ループに入りデッド状態になります。

```
; Enters this handler when both banks have invalid images

ImageInvalid:
     B    ImageInvalid
```

5.7.4 SFlash から SRAM へのコピー

有効なイメージがいずれかのフラッシュバンクで使用可能な場合、または認証フラグが無効になっている場合、_flashbankswitch 関数が SFlash から SRAM にコピーされ、デュアルバンクマネージャはこの SRAM 関数にジャンプします。

```
FlashtoRAM_copy:

; Copy the _flashbankswitch function to SRAM and Dual Bank Manager jumps to SRAM where the dual
bank configuration is done

; Load the register with the source address, destination address and size
     ADR  r3, _flashbankswitch
     LDR  r2, flashmarkers
     LDR  r1, flashmarkers+4

; Subtract 1 from R2 due to Thumb alignment
     SUBS r2, #1

; R0 is used for compare operation
     MOVS r0, #0
     B    compare

; loop to copy the complete SFlash function to SRAM
copyloop:
     LDR  r4, [r3,r0]
     STR  r4, [r2,r0]
     ADDS r0, #4
compare:
     CMP  r0, r1
     BCC  copyloop

; Jump to the _flashbankswitch copy function in SRAM with valid r6
     LDR  r7, flashmarkers
     BX   r7
```

5.8 デジタル署名検証機能関数

デュアルバンクマネージャは、LowerBankCheck および UpperBankCheck 関数を使用して、コードフラッシュアプリケーションの認証をチェックします。以下は LowerBankCheck のコードです。

```
; Performs authentication of the lower Bank
LowerBankCheck:
.align 4

_checkLB:

     PUSH {LR}

; Check if the Application is valid with Cy_FB_VerifyApplication()

; Load the registers with the lower bank parameters for Cy_FB_VerifyApplication() function
     LDR  r0, addrmarker+4        ; address of the lower bank image
     LDR  r1, [r0]                ; length of the lower bank image
     ADDS r2, r0, r1              ; address of the digital signature
     BCS  skip_LB_check           ; carry bit detected, boundary violation
     LDR  r4, addrmarker+16       ; end address for the lower bank
     CMP  r2, r4                  ; boundary condition check
     BHI  skip_LB_check           ; skip Cy_FB_VerifyApplication
     LDR  r3, addrmarker+8        ; address of the public key

; Get the address of Cy_FB_VerifyApplication() from SFlash marker
     LDR  r4, addrmarker
     LDR  r7, [r4]

; Call Cy_FB_VerifyApplication() function
     BLX  r7

skip_LB_check:
     POP  {PC}

     .endf _checkLB

; Performs authentication of the upper Bank
UpperBankCheck:
.align 4

_checkUB:

     PUSH {LR}

; Check if the Application is valid with Cy_FB_VerifyApplication()

; Load the registers with the upper bank parameters for Cy_FB_VerifyApplication() function
     LDR  r0, addrmarker+12       ; address of the upper bank image
     LDR  r1, [r0]                ; length of the upper bank image
     ADDS r2, r0, r1              ; address of the digital signature
     BCS  skip_UB_check           ; carry bit detected, boundary violation
     LDR  r4, addrmarker+20       ; end address for the upper bank
```

```
    CMP  r2, r4                 ; boundary condition check
    BHI  skip_UB_check          ; skip Cy_FB_VerifyApplication
    LDR  r3, addrmarker+8       ; address of the public key

; Get the address of Cy_FB_VerifyApplication() from SFlash marker
    LDR  r4, addrmarker
    LDR  r7, [r4]

; Call Cy_FB_VerifyApplication() function
    BLX  r7

skip_UB_check:
    POP  {PC}

    .endf _checkUB
```

UpperBankCheck も、上位バンクに対して同様のコンセプトを使用していることに注意してください。この関数は、SFlash で提供される Cy_FB_VerifyApplication 関数を内部で使用します。この関数のパラメータはチェックされ、次にレジスタ r0～r3 にロードされます。関数のアドレスがレジスタ r4 にロードされ、そして BLX を使用してジャンプが実行されます。コードフラッシュメモリの境界条件も、Cy_FB_VerifyApplication が呼び出される前にバンク内でチェックされます。境界条件に違反した場合、Cy_FB_VerifyApplication は呼び出されず、認証は失敗します。コードフラッシュアプリケーションのデジタル署名が有効な場合、レジスタ r0 は 1 に設定されます。それ以外の場合は 0 になります。addrmarker の詳細については、8.4.2 を参照してください。

6 フラッシュバンク管理用の SRAM 関数

TRAVEO™ T2G のバンク切替えメカニズムで説明したように、フラッシュバンク管理オペレーションは SRAM メモリから実行されます。これは、SFlash で使用可能な_flashbankswitch 関数のコピーです。図 5 に、この関数の実装を示します。ramdatamarker の詳細については、付録 8.4.3 を参照してください。

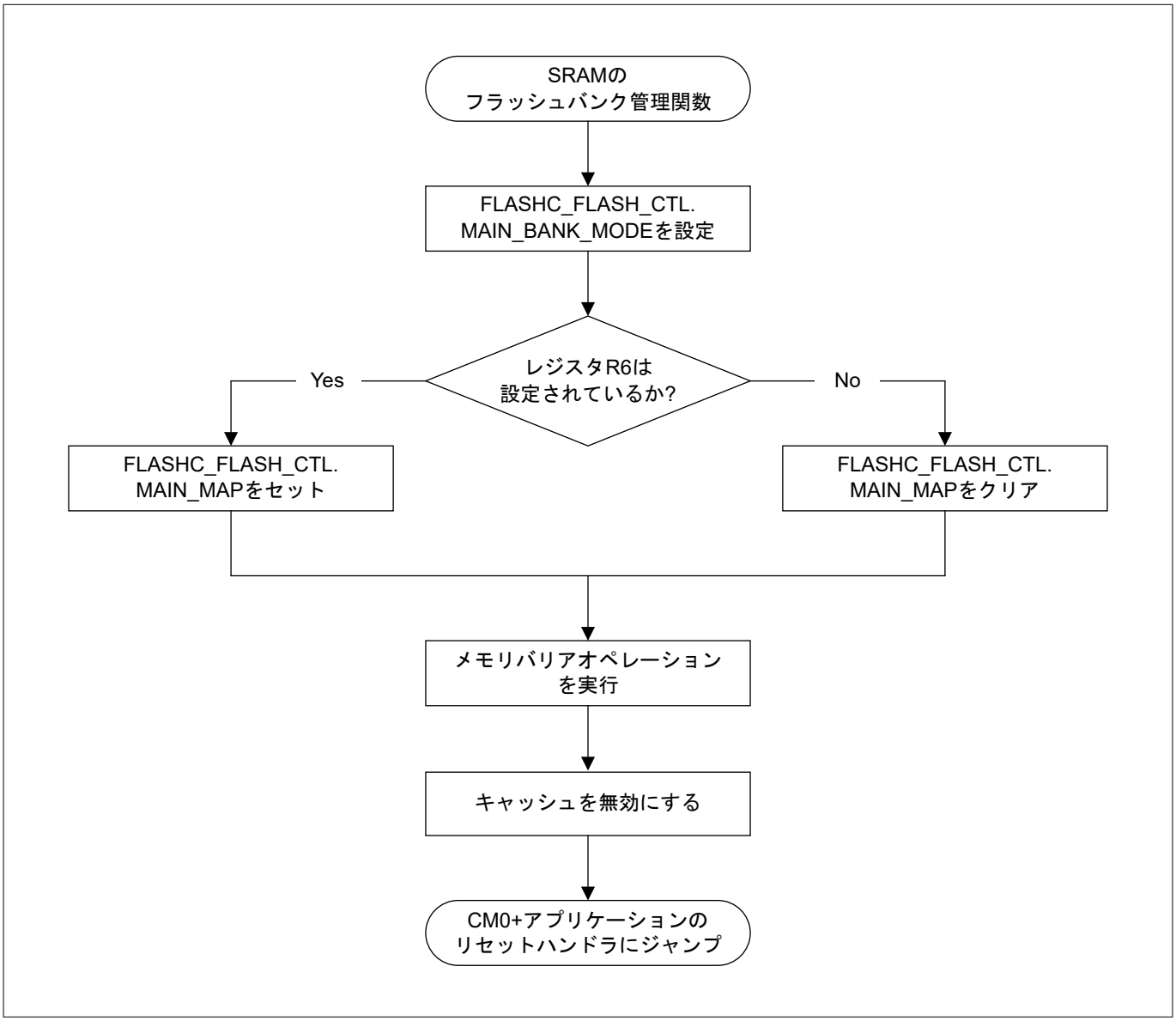

図 5 SRAM 関数のフローチャート

6.1 フラッシュをデュアルバンクとして設定

FLASHC_FLASH_CTL.MAIN_BANK_MODE ビットを設定し、デュアルバンクモードを設定してください。

```
; Load the address for FLASHC_FLASH_CTL
     LDR  r4, ramdatamarker

; Set FLASHC_FLASH_CTL.MAIN_BANK_MODE for dual bank mode
     LDR  r0, [r4]
     MOVS r1, #1
     LSLS r1, r1, #12
     ORRS r0, r1
     STR  r0, [r4]
     LDR  r0, [r4]
```

6.2 フラッシュマップの設定

以前に実行されたマジックワードと認証チェックに基づき、マップ A またはマップ B で動作するように FLASHC_FLASH_CTL.MAIN_MAP を設定してください。

```
; Set '1' for bit corresponding to FLASHC_FLASH_CTL.MAIN_MAP in register R1
     MOVS r1, #1
     ADDS r1, #255

; Check if register R6 is set
     CMP  r6, #1
     BNE  MapA

; Map B is valid, set the bit corresponding to FLASHC_FLASH_CTL.MAIN_MAP
MapB:
     ORRS r0, r1
     B    common

; Map A is valid, clear the bit corresponding to FLASHC_FLASH_CTL.MAIN_MAP
MapA:
     BICS r0, r1

common:

; configure FLASHC_FLASH_CTL.MAIN_MAP
     STR  r0, [r4]

; read back FLASHC_FLASH_CTL
     LDR  r0, [r4]
```

6.3 メモリバリアオペレーションとキャッシュの無効化

メモリバリア命令を発行してから、キャッシュとバッファを無効にしてください。

```
; instruction barrier operation
     ISB  SY

; invalidate cache and buffers
     LDR  r0, [r4,8]
     MOVS r1, #1
     ORRS r0, r1
     STR  r0, [r4,8]

; wait till cache and buffers are invalidated
cache_clearloop:
     LDR  r0, [r4,8]
     LSLS r0, r0, #31
     BMI  cache_clearloop
```

6.4 CM0+アプリケーションへの分岐

CySAF 形式を使用して識別されたコードフラッシュアプリケーションのリセットハンドラに分岐し、それにジャンプしてください。

```
; jump to the _startcontd function which is the reset handler of the CM0+ application placed at
0x10000000 based on CySAF
     LDR  r5, ramdatamarker+4         ; Load the starting address of CySAF
     LDR  r1, [r5,16]
     ADDS r1, #16
     ADDS r5, r1
     LDR  r7, [r5,4]
     BX   r7
     .endf _flashbankswitch
```

7 まとめ

このアプリケーションノートは、セキュアな FOTA アップデートの設計に含まれるさまざまな手順について説明しました。FOTA アップデート後、CM0+アプリケーションを安全に起動することに重点が置かれていることに注意してください。この CM0+アプリケーションは、CM4 または CM7 に基づくホストアプリケーションを追加で認証できます。

8 付録 A. デュアルバンクマネージャ: 依存関係

ここでは、セキュアな FOTA を含むプロジェクトの開発に関連する追加の依存関係について説明します。また、必要な定数、マクロ 、ベクタテーブル構成などについても説明します。これらは通常、デュアルバンクマネージャで使用されます。

8.1 定数

```
STARTUP_STACK_SIZE_DOUBLE_WORDS     .equ 128

VTOR                                .equ 0xe000ed08
CPUSS_RAM0_CTL0                     .equ 0x40201300
CPUSS_RAM1_CTL0                     .equ 0x40201380
CPUSS_RAM2_CTL0                     .equ 0x402013a0

; Macros used for flash configurations
MAGIC_WORD                          .equ 0xAAAAAAAA
ADDR_MAKER_ON_WORK                  .equ 0x14012000
FLASHC_FLASH_CTL                    .equ 0x40240000

; Macros used for application authentication

CY_FB_VERIFYAPP_ADDR                .equ 0x17002040
CY_PUBLIC_KEY_ADDR                  .equ 0x17006408
CY_APP_START_ADDR_LB                .equ 0x10000000

; Macro to enable application authentication by Dual Bank Manager
AUTHENTICATION_ENABLE               .equ 0x00000001

; Macro to disable application authentication by Dual Bank Manager
AUTHENTICATION_DISABLE              .equ 0x55555555

; The following macros should be set by user based on the device and requirement

; Set AUTHENTICATION_CHECK to AUTHENTICATION_ENABLE or AUTHENTICATION_DISABLE
AUTHENTICATION_CHECK                .equ AUTHENTICATION_ENABLE

; Mid-Point of the Large sectors in single bank mode for Traveo T2G 1M device. This value shall
be modified based on the Target Device. Application intended to be executed out of Map B shall
be placed at this starting address.

CY_APP_START_ADDR_UB                .equ 0x10078000

; Macro used for the flash boundary check. This macro has the value ( Map A/B Large sector end
address in single bank mode - size of the digital signature (256 bytes)) for Traveo T2G 1M
device. These values shall be modified based on the Target Device

CY_APP_END_ADDR_LB                  .equ 0x10077F00             ; 0x10078000 - 0x100
CY_APP_END_ADDR_UB                  .equ 0x100EFF00             ; 0x100F0000 - 0x100

; For CYT2B78CAE the user reset handler is _startcontd which is continuation of the SFlash
_start

User_Reset_Handler                  .equ _startcontd
```

8.2 デュアルバンクマネージャのベクタテーブル

```
; This is the vector table required for CySAF of the Dual Bank Manager. Only the first 4
entries are used, others are filled with zero.
        .section ".sflashintvec", "a"
        ; align to 256 bytes, because CM0_VECTOR_TABLE_BASE register only supports address bits
[31:8]
       .align 256

__sflashvector_table:

        DCD     __ghsend_stack
        DCD     _start
        DCD     Exception_handler
        DCD     Exception_handler
        DCD     0
        DCD     0
        DCD     0
__sflashvector_table_0x1c:
        DCD     0
        DCD     0
        DCD     0
        DCD     0
        DCD     0
        DCD     0
        DCD     0
        DCD     0
        DCD     0

        ; External interrupts            PowerMode    Description

        DCD     0                       ; DeepSleep    CPU User Interrupt #0
        DCD     0                       ; DeepSleep    CPU User Interrupt #1
        DCD     0                       ; DeepSleep    CPU User Interrupt #2
        DCD     0                       ; DeepSleep    CPU User Interrupt #3
        DCD     0                       ; DeepSleep    CPU User Interrupt #4
        DCD     0                       ; DeepSleep    CPU User Interrupt #5
        DCD     0                       ; DeepSleep    CPU User Interrupt #6
        DCD     0                       ; DeepSleep    CPU User Interrupt #7

        ; These IRQs can only be triggered by SW via NVIC regs
        DCD     0                       ; Active       CPU User Interrupt #8
        DCD     0                       ; Active       CPU User Interrupt #9
        DCD     0                       ; Active       CPU User Interrupt #10
        DCD     0                       ; Active       CPU User Interrupt #11
        DCD     0                       ; Active       CPU User Interrupt #12
        DCD     0                       ; Active       CPU User Interrupt #13
        DCD     0                       ; Active       CPU User Interrupt #14
        DCD     0                       ; Active       CPU User Interrupt #15

        .endo __sflashvector_table
```

8.3 SRAM 関数空間の予約

```
; SRAM section to reserve space for SRAM function needed for flash bank configurations
        .section ".ramprog", "awx"
        .align 8

; number of bytes for to allocated in ramprog for _flashbankswitchramcopy
_flashbankswitchsize           .equ (_flashbankswitchend - _flashbankswitch)

_flashbankswitchramcopy:
        .space        _flashbankswitchsize
#endif
```

8.4 マクロ

8.4.1 flashmarkers

```
; Markers for flash copy operation
.align 4
flashmarkers:
     DCD    _flashbankswitchramcopy                          ; flashmarkers
     DCD    _flashbankswitchsize                             ; flashmarkers+4
     DCD    FLASHC_FLASH_CTL                                 ; flashmarkers+8
     DCD    ADDR_MAKER_ON_WORK                               ; flashmarkers+12
     DCD    MAGIC_WORD                                       ; flashmarkers+16
     DCD    AUTHENTICATION_CHECK                             ; flashmarkers+20
     DCD    AUTHENTICATION_DISABLE                           ; flashmarkers+24
     DCD    (__ghsend_stack - (#STARTUP_STACK_SIZE_DOUBLE_WORDS * 8)); flashmarkers+28

.endo flashmarkers
```

8.4.2 addrmarker

```
; Markers used by the LowerBankCheck and UpperBankCheck functions
.align 4
addrmarker:
     DCD    CY_FB_VERIFYAPP_ADDR      ; addrmarker
     DCD    CY_APP_START_ADDR_LB      ; addrmarker+4
     DCD    CY_PUBLIC_KEY_ADDR        ; addrmarker+8
     DCD    CY_APP_START_ADDR_UB      ; addrmarker+12
     DCD    CY_APP_END_ADDR_LB        ; addrmarker+16
     DCD    CY_APP_END_ADDR_UB        ; addrmarker+20
.endo addrmarker
```

8.4.3 ramdatamarker

```
; Markers used by the RAM function
.align 4
ramdatamarker:
     DCD    FLASHC_FLASH_CTL          ; ramdatamarker
     DCD    CY_APP_START_ADDR_LB      ; ramdatamarker+4
.endo ramdatamarker
```

8.5 CySAF

TRAVEO™ T2G デバイスが「SECURE」保護状態にある場合、Flash ブートは CySAF 形式のみに基づいてアプリケーションを起動できます。CySAF で使用される形式の詳細については、アーキテクチャ TRM のアプリケーションフォーマットのセクションを参照してください。

Flash ブートによって起動されるデュアルバンクマネージャは、CySAF 形式に基づきます。

8.6 TOC2 の詳細

ROM ブートと Flash ブートは、TOC2 のコンテンツをさまざまなオペレーションに使用します。表 2 に、セキュアな FOTA の一部として TOC2 に加えられた変更を示します。

表 2 セキュアな FOTA の TOC2

オフセット	目的	デフォルト値	変更された値	コメント
0x00	オフセット 0x00 から始まる CRC 計算のバイト単位のオブジェクトサイズ。	0x00001FC		
0x04	マジックナンバー (固定: 0x01211220)	0x01211220		
0x08	SMIF 設定構成を表すヌル終了ポインタのテーブル	0x00000000		
0x0C	**最初のユーザアプリケーションオブジェクトのアドレス**	**0x10000000**	**0x17000800**	**Flash ブートにより起動されるデュアルバンクマネージャ**
0x10	**最初のユーザアプリケーションオブジェクトの形式** **0: Basic, 1: Infineon standard, 2: Simplified**	**0x00000000**	**0x1**	**CySAF 形式は「SECURE」保護状態で使用する必要があります**
0x14	2 番目のユーザアプリケーションオブジェクトのアドレス。 最初のアプリケーションの検証が失敗した場合、2 番目のユーザーアプリケーションが検証されます	0x00000000		
0x18	2 番目のユーザアプリケーションオブジェクトの形式。 0: Basic, 1: Infineon standard, 2: Simplified	0x00000000		
0x1C	最初の CM4 または CM7 core1 のユーザアプリケーションオブジェクトのアドレス	0x00000000		

(続く)

表 2 (続き) セキュアな FOTA の TOC2

オフセット	目的	デフォルト値	変更された値	コメント
0x20	2 番目の CM4 または CM7 core1 のユーザアプリケーションオブジェクトのアドレス	0x00000000		
0x24	最初の CM4 または CM7 core2 のユーザアプリケーションオブジェクトのアドレス	0x00000000		
0x28	2 番目の CM4 または CM7 core2 のユーザアプリケーションオブジェクトのアドレス	0x00000000		
0xFC	マジックナンバーが有効な場合、セキュリティエンハンス用の保護構成を有効にします。	0x00000000		詳細については、AN228680 - Secure system configuration in TRAVEO™ T2G family [5]を参照してください。
0x100	**「SECURE_HASH」で検証される追加オブジェクトの数**	**0x00000003**	**4**	**デュアルバンクマネージャが追加オブジェクトとして追加されています。**
0x104	**署名検証キーのアドレス (ない場合は 0)。 オブジェクトは署名固有のキーです。RSA の場合は公開鍵です。**	**0x00000000**	**0x17006400**	**署名認証に使用される公開鍵 (APP_AUTH_CTL が無効になっている場合は使用されません) が使用可能なアドレス。**
0x108	アプリケーション保護のアドレス	0x17007600		この領域は変更禁止
0x10C	予約済み	0x00000000		この領域は変更禁止
0x110	**追加オブジェクト**	**0x00000000**	**0x17000800**	**デュアルバンクマネージャ**
0x114 ～ 0x1F0	必要に応じて追加のオブジェクト (ない場合は 0)	0x00000000		
0x1F8	デフォルト構成を制御します。 ビット[1:0]: CLOCK_CONFIG クロック周波数設定を示すフラグ。Flash ブートの実行後、クロックは同じままである必要があります。 **値[1:0]** / **説明** 0x0 / 8 MHz, IMO, FLL なし	デフォルト値については使用するレジスタ TRM [2][3][4]を参照してください。	**0x000002C2**	**Flash ブートによる認証が無効になります (ビット [8:7]は 0x1 です)**

(続く)

表 2 (続き) セキュアな FOTA の TOC2

オフセット	目的		デフォルト値	変更された値	コメント
	0x1	25 MHz, IMO + FLL			
	0x2	50 MHz, IMO + FLL			
	0x3	ROM ブートクロック設定 (100 MHz) を使用			
	ビット[4:2]: LISTEN_WINDOW デバッグポートを取得するのに十分な時間を与えるためにリッスンウィンドウを決定するフラグ。				
	値[4:2]	**説明**			
	0x0	20ms			
	0x1	10ms			
	0x2	1ms			
	0x3	0 ms (リッスンウィンドウなし)			
	0x4	100ms			
	Others	予約			
	ビット[6:5]: SWJ_PINS_CTL SWJ ピンが Flash ブートによって SWJ モードで構成されているかどうかを判別するためのフラグ。 *注: SWJ ピンは、ユーザコードの後半で有効になる場合があります。*				
	値[6:5]	**説明**			
	0x0	Flash ブートで SWJ ピンを有効にしない。リッスンウィンドウはスキップされます			
	0x1	Flash ブートで SWJ ピンを有効にしない。リッスンウィンドウはスキップされます			
	0x2	Flash ブートで SWJ ピンを有効にする			
	0x3	Flash ブートで SWJ ピンを有効にしない。リッスンウィンドウはスキップされます			
	ビット[8:7]: APP_AUTH_CTL アプリケーションイメージのデジタル署名検証 (認証) が実行されるかどうかを決定するフラグ。				
	値[8:7]	**説明**			
	0x0	認証は有効			

(続く)

表 2 (続き) セキュアな FOTA の TOC2

オフセット	目的		デフォルト値	変更された値	コメント
	0x1	認証は無効			
	0x2	認証は有効 (推奨)			
	0x3	認証は有効			
	ビット[10:9]: FB_BOOTLOADER_CTL Flash ブートの内部ブートローダが無効になっているかどうかを判別するためのフラグ。				
	値[9]	**説明**			
	0x0	内部ブートローダは無効			
	0x1	他のブートローダ条件が満たされた場合、内部ブートローダが起動される			
	0x2	内部ブートローダは無効			
	0x3	内部ブートローダは無効			

8.7 「SECURE」への移行

TRAVEO™ T2G デバイスは、NORMAL_PROVISIONED ライフサイクル状態でお客様に出荷されます。「SECURE」への移行システムコールは、「SECURE」ライフサイクル状態への移行に使用されます。「SECURE」ライフサイクルに移行した後は SFlash の書き込みが許可されないため、「SECURE」ライフサイクルに移行する前に SFlash メモリを正しくプログラムしなければいけないことに注意してください。

詳細は、アーキテクチャ TRM の「SECURE」への移行セクションを参照してください。SECURE ライフサイクルステージへ移行する間、SECURE_HASH は、TOC2 の追加オブジェクトとして追加されるデュアルバンクマネージャを含むすべての必要なオブジェクトに対して計算されます。

8.8 必要なツール

以下は、さまざまなオペレーションをサポートするために必要ないくつかのツールです。

- OpenSSL
- Cypress Auto Flash Utility 1.0
- cymcuelftool
- Python 3

例えば、Cypress Auto Flash Utility 1.0 は SFlash プログラミングに対応しています。それは、デュアルバンクマネージャと TOC2 のプログラミングに使用できます。OpenSSL は秘密鍵と公開鍵の生成に使用でき、cymcuelftool はアプリケーションのデジタル署名の生成に対応します。これらのツールはデモンストレーションのみを目的としており、製造設計には使用できないことに注意してください。

これらのツールの詳細については、AN228680 - Secure system configuration in TRAVEO™ T2G family を参照してください。

9 用語集

用語	説明
CPU	Central processing unit (中央処理装置)
NVIC	Nested vectored interrupt controller (統合ネスト型ベクタ割込みコントローラ)
NMI	Non-maskable interrupt (ノンマスカブル割込み)
IRQ	Interrupt request (割込み要求)
FOTA	Firmware over the air (ファームウェアオーバーザエア)
OTA	Over-the-air (オーバーザエア)
SWPU	Software protection unit (ソフトウェア保護ユニット)
ECC	Error correction code (誤り訂正符号)
ROM	Read-only memory (読み出し専用メモリ)
SFlash	Supervisory flash (監視フラッシュ)
SRAM	Static random-access memory (スタティックラム)
TOC	Table of contents (目次)
CySAF	Cypress secure application format (サイプレスセキュアアプリケーションフォーマット)

10 関連資料

以下は、TRAVEO™ T2G ファミリのデータシートおよびテクニカルリファレンスマニュアルです。これらのドキュメントを入手するには、テクニカルサポートに連絡してください。

[1] デバイスデータシート

- CYT2B7 datasheet 32-bit Arm® Cortex®-M4F microcontroller TRAVEO™ T2G family
- CYT2B9 datasheet 32-bit Arm® Cortex®-M4F microcontroller TRAVEO™ T2G family
- CYT4BF datasheet 32-bit Arm® Cortex®-M7 microcontroller TRAVEO™ T2G family
- CYT4DN datasheet 32-bit Arm® Cortex®-M7 microcontroller TRAVEO™ T2G family (Doc No. 002-24601)
- CYT3BB/4BB datasheet 32-bit Arm® Cortex®-M7 microcontroller TRAVEO™ T2G family
- CYT3DL datasheet 32-bit Arm® Cortex®-M7 microcontroller TRAVEO™ T2G family (Doc No. 002-27763)

[2] ボディコントローラ Entry ファミリ

- TRAVEO™ T2G automotive body controller entry family architecture technical reference manual (TRM)
- TRAVEO™ T2G automotive body controller entry registers technical reference manual (TRM) for CYT2B7
- TRAVEO™ T2G automotive body controller entry registers technical reference manual (TRM) for CYT2B9

[3] ボディコントローラ High ファミリ

- TRAVEO™ T2G automotive body controller high family architecture technical reference manual (TRM)
- TRAVEO™ T2G automotive body controller high registers technical reference manual (TRM) for CYT4BF
- TRAVEO™ T2G automotive body controller high registers technical reference manual (TRM) for CYT3BB/4BB

[4] クラスタ 2D ファミリ

- TRAVEO™ T2G automotive cluster 2D family architecture technical reference manual (TRM) (Doc No. 002-25800)
- TRAVEO™ T2G automotive cluster 2D registers technical reference manual (TRM) for CYT4DN (Doc No. 002-25923)
- TRAVEO™ T2G automotive cluster 2D registers technical reference manual (TRM) for CYT3DL (Doc No. 002-29854)

[5] アプリケーションノート

- AN228680 – Secure system configuration in TRAVEO™ T2G family
- AN220242 - TRAVEO™ T2G ファミリのフラッシュアクセス手順

改訂履歴

版数	発行日	変更内容
**	2021-01-06	これは英語版 002-29058 Rev. **を翻訳した日本語版 002-31969 Rev. **です。
*A	2021-05-18	テンプレートの変更を実施。これは英語版 002-29058 Rev. *A を翻訳した日本語版 Rev. *A です。
*B	2021-11-08	これは英語版 002-29058 Rev. *B を翻訳した日本語版 Rev. *B です。英語版の改訂内容: Changed TOC2 table for security marker Added supported device (CYT3DL)
*C	2022-08-30	これは英語版 002-29058 Rev. *C を翻訳した日本語版 Rev. *C です。英語版の改訂内容: Changed 'Public key is programmed to the SFlash Rows' in section セキュアな FOTA の「セキュアブート」メカニズム From (Row 45 to Row 50) to (Row 50 to Row 55)
*D	2024-07-03	これは英語版 002-29058 Rev. *D を翻訳した日本語版 Rev. *D です。英語版の改訂内容: Template update; no content update

Trademarks

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

Edition 2024-07-03

Published by

Infineon Technologies AG

81726 Munich, Germany

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

Email: erratum@infineon.com

Document reference

IFX-nga1686137686525

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。