

RXファミリ

RX74M ファーストステージブートローダを使ったアプリケーションの開発

要旨

RX74M グループには、変更不能なファーストステージブートローダ(First Stage Bootloader、FSBL)が実装されています。FSBL はリセット後に実行され、MRAM に格納された OEM ブートローダ(OEM_BL)を検証し、検証に成功した場合にのみ OEM_BL へ制御を渡します。これにより、変更不可能な FSBL を起点とするブートチェーンを構築できます。

RX74M の FSBL は、CRC32 によって OEM_BL の完全性を確認する CRC ブートと、署名検証およびデバイス固有鍵を用いた HMAC-SHA256 によって OEM_BL の真正性と完全性を確認する HMAC ブートをサポートします。HMAC ブートでは、製品プログラミング時に ECDSA P-256 署名を使用して OEM_BL を検証します。

本アプリケーションノートでは、RX74M で FSBL を使用するための設定、検証に必要な鍵および証明書の生成、Renesas Flash Programmer を使用した製品プログラミング、ならびに付属のサンプルプロジェクトを使用した CRC ブートおよび HMAC ブートの動作確認手順を説明します。

必要なリソース

対象デバイス

- RX74M(RSIP-E50D 搭載)

ソフトウェアと開発ツール

- e² studio IDE v2026-07
- Renesas Flexible Software Package (FSP) v1.0.0
ダウンロードはこちらから<https://github.com/renesas/rx-fsp>.
- Renesas Security Key Management Tool (SKMT) v1.12
- Renesas Flash Programmer (RFP) v3.24
- GNU Privacy Guard for Windows: [Gpg4win](https://gnupg.org/why-gpg4win/)
- ルネサス KeyWrap サーバ

ハードウェア

- EK-RX74M、RX74M グループ用評価キット([renesas.com/rx/ek-rx74m](https://www.renesas.com/rx/ek-rx74m))
- Windows®11 と Tera Term コンソールまたは同様のアプリケーションを実行可能な PC
- USB Type-C®ケーブル(USB Type-C®オス - USB Type-C®オス)1 本

本書で使用する用語の説明を以下に示します。

表 用語説明

用語	説明
RSIP	Renesas Secure IP ルネサス製 MCU に搭載されているセキュリティエンジン。 RX74M では RSIP-E50D が搭載されている。
HUK	Hardware Unique Key デバイス内部に存在する、鍵の保護のために使用するデバイス固有の暗号鍵。
UFPK	User Factory Programming Key 鍵注入においてユーザ鍵を暗号化するために使用する、ユーザが設定する鍵束。デバイス上では使用しない。
W-UFPK	Wrapped UFPK UFPK を HRK(RSIP 内部とルネサス内セキュアルームのみに存在する共通の暗号鍵)によりラッピングすることで生成される鍵情報。RSIP 内部にて HRK で UFPK に復号されて使用される。
MRAM	磁気抵抗メモリ 一定以上のプロセスノードにおいてフラッシュに替わり NVRAM として使用される。RX74M にはコード MRAM とデータ MRAM が搭載されている。
RFP	Renesas Flash Programmer ルネサス製 MCU の NVRAM にプログラムを書き込むためのツール。
SKMT	Security Key Management Tool セキュアな鍵注入やファームウェアアップデートを支援するツール。
OEM_BL	起動する前に FSBL で検証するプログラム OEM が開発する Second Stage Bootloader またはアプリケーションプログラム。
SSBL	Second Stage Bootloader FSBL により検証されてから実行されるブートローダ。

参考情報

本アプリケーションノートに付属のデモプロジェクトは、ルネサスの統合開発環境である e² studio を使用しています。また、アプリケーションブートのローディングや暗号化アルゴリズムに関する知識があることが望ましいです。必要に応じて、RX74M グループ ユーザーズマニュアル ハードウェア編の以下の章を参照してください。

- 「セキュリティ機能」
- 「オプション設定メモリ」
- 「MRAM」

本アプリケーションノートは、ルネサス RX ファミリ MCU のファーストステージブートローダを使ったアプリケーション開発に携わる製品開発者、製品メーカー、製品サポート、またはエンドユーザが読むことを想定して作成されています。

目次

1. FSBLの概要	6
1.1 FSBLの目的と役割	6
1.2 RX74MにおけるFSBL	7
1.3 製品プログラミングから起動までの全体フロー	8
1.3.1 CRCブートにおける認証情報の適用フロー	8
1.3.2 HMACブートにおける認証情報の適用フロー	9
2. RX74M FSBLのセキュリティモデルと動作	10
2.1 信頼の基点とChain of Trust	10
2.2 HMACブートで使用する鍵と認証情報	12
2.2.1 OEM_BL検証ルート鍵とOEM_BL検証鍵	12
2.2.2 鍵証明書	13
2.2.3 コード証明書	14
2.3 シングルチップモードでの起動時検証	16
3. CRCブートのサンプルプロジェクト	19
3.1 RX74M FSBLのCRCブート設定	19
3.2 コード証明書の作成	21
3.3 RFPIによるプログラミングと動作確認	23
4. HMACブートのサンプルプロジェクト	26
4.1 鍵の準備	26
4.1.1 OpenSSLを使用したECC鍵ペアの生成	26
4.1.2 ECC 鍵ペアとするNIST CAVP テストベクタ	27
4.2 Root of Trustの導入	28
4.2.1 ラッピングされたUFPK(Wrapped User Factory Programming Key、W-UFPK)の生成	28
4.2.2 SKMT CLIを使用したラッピングされたOEM_ROOT_PKの生成	30
4.2.3 SKMT GUIを使用したラッピングされたOEM_ROOT_PKの生成	31
4.3 RX74M FSBLのHMACブート設定	33
4.4 アンチロールバックカウンタ	36
4.5 証明書の生成	38
4.5.1 OpenSSLとSKMTコマンドラインインタフェースの使用	38
4.5.2 NIST ECCの鍵ペアとSKMT GUIの使用	39
4.6 認証付き製品プログラミング	42
5. 付録	46
5.1 ルネサスセキュアファクトリプログラミングでのFSBL使用	46
5.2 FSBLのレジスタ設定	46
6. References	48
7. ウェブサイトとサポート	49

改訂記録.....50

1. FSBL の概要

セキュアな組み込みシステムでは、実行するソフトウェアが意図した提供者によって作成され、かつ保存後に改変されていないことを、実行開始前に確認する必要があります。この確認をブートローダが行う場合、そのブートローダ自身も信頼できる状態でなければなりません。

ファーストステージブートローダ（FSBL）は、リセット後に最初に実行され、次に実行するソフトウェアイメージを検証するブートローダです。本書では、FSBL が直接検証して制御を渡すイメージを OEM_BL と呼びます。OEM_BL には、製品アプリケーションを直接配置する構成と、MCUboot などのセカンドステージブートローダを配置する構成があります。

RX74M には、ユーザが変更できない FSBL が実装されています。RX74M の FSBL は、OEM_BL に対する CRC32 ベースの完全性検証、または署名認証と HMAC-SHA256 を組み合わせた HMAC ブートを提供します。本章では FSBL の役割と RX74M における全体的な処理を説明し、認証情報と詳細な検証処理については2章で説明します。

1.1 FSBL の目的と役割

セキュアシステムでは、アプリケーションプログラムが実行される前に、プログラムが意図的、または偶発的に変更されていないことを確認する必要があります。この確認は通常、ブートローダによって行われ、簡単な完全性チェックや、信頼性を保証するための署名検証を行うものがあります。ただし、ブートローダ自体の正当性も保証されなければならないことに注意が必要です。

以下は、MCU シングルチップモードで FSBL を使用した際の処理の概要を示したものです。

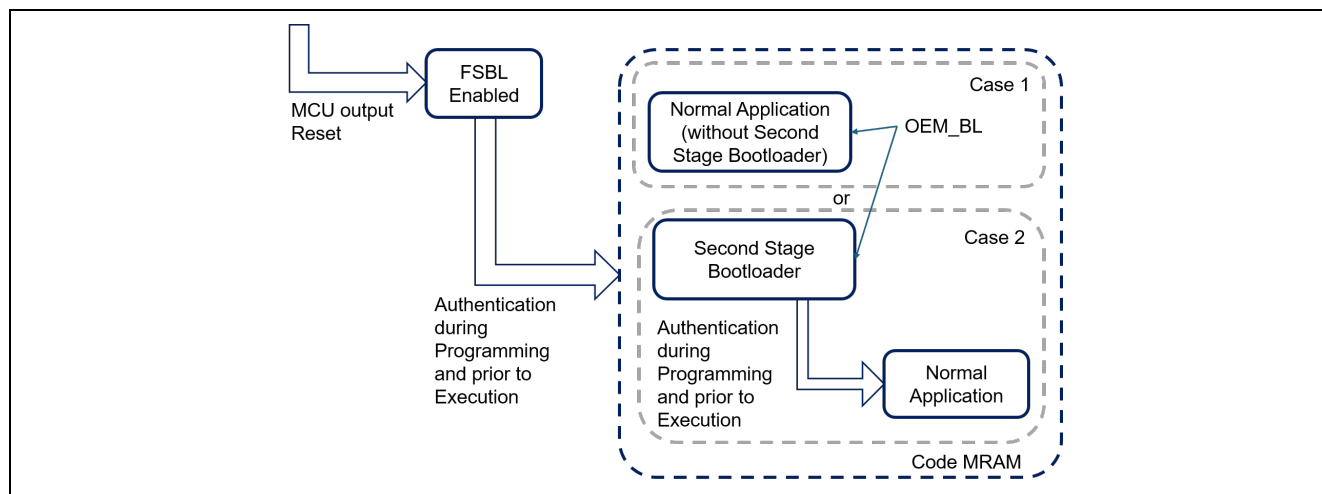


図1-1 シングルチップモードでの FSBL の使用方法

1.2 RX74M における FSBL

FSBL が有効になっている場合、OEM_BL を検証する 2 つのオプションがあります。

- CRC ブート：CRC32 を使用した完全性チェックをします。MCU の初期プログラミング時と実行前に行われます。
- HMAC ブート：真正性のチェックをします。MCU の初期プログラミング時、OEM_BL は secp256r1(NIST P-256)ECC 曲線を使用した ECDSA で認証されます。また、実行前には MCU の HUK で HMAC-SHA256 を使用して認証されます。

HMAC ブートシーケンスは強力な Root of Trust を確保するために、不変のエンティティから始めて段階的に実装されます。RX74M には不変の FSBL が書き込まれていて、HMAC ブートシーケンスのスタートとして Root of Trust を提供します。最初に FSBL は OEM_BL を検証します。その後 OEM_BL が通常のアプリケーションプログラムを検証して、システムを起動するため信頼の連鎖(Chain of Trust、認証パスについては図1-1 Case 2 を参照)を形成します。

CRC ブートのプロセスは HMAC ブートと比較して簡単になっています。HMAC ブートと比較した際の CRC ブートの特徴は以下のとおりです。

表 1-1 CRC ブートと HMAC ブートの比較

項目	CRC ブート	HMAC ブート
Root of Trust	不要	必須
鍵証明書	不要	必須
コード証明書	必須	必須
ECDSA 署名	不要	必須
アンチロールバック機能	不要	必須
完全性検証	CRC32	HMAC-SHA256
真正性検証	×	○

1.3 製品プログラミングから起動までの全体フロー

前節までに説明した Root of Trust、鍵証明書、およびコード証明書は、それぞれ独立して存在するものではなく、製品へのプログラミング時および起動時の認証プロセスにおいて連携して使用されます。FSBL はこれらの認証情報を利用して Chain of Trust を構築し、OEM_BL が改ざんされていないこと、および正規の開発者によって生成されたものであることを検証します。

認証に使用される認証情報および検証手順は、CRC ブートと HMAC ブートで異なります。CRC ブートではコード証明書に格納された CRC 値を用いて OEM_BL の完全性を確認します。一方、HMAC ブートでは Root of Trust、鍵証明書、コード証明書、および HMAC ダイジェストを組み合わせることで、OEM_BL の真正性と完全性を検証します。

以下では、製品へのプログラミング時において各認証情報がどのように使用されるかを、CRC ブートと HMAC ブートそれぞれについて説明します。

1.3.1 CRC ブートにおける認証情報の適用フロー

RFP による OEM_BL のプログラミング段階では、以下の項目が MCU にプログラムされます。

- OEM_BL(アプリケーションイメージ)
- コード証明書

以下の図は、CRC ブートを使用した製品へのプログラミング中に SKMT と RFP を使用した全体的なワークフローをまとめたものです。SKMT と RFP の使い方についての詳しい手順は、3章で説明します。

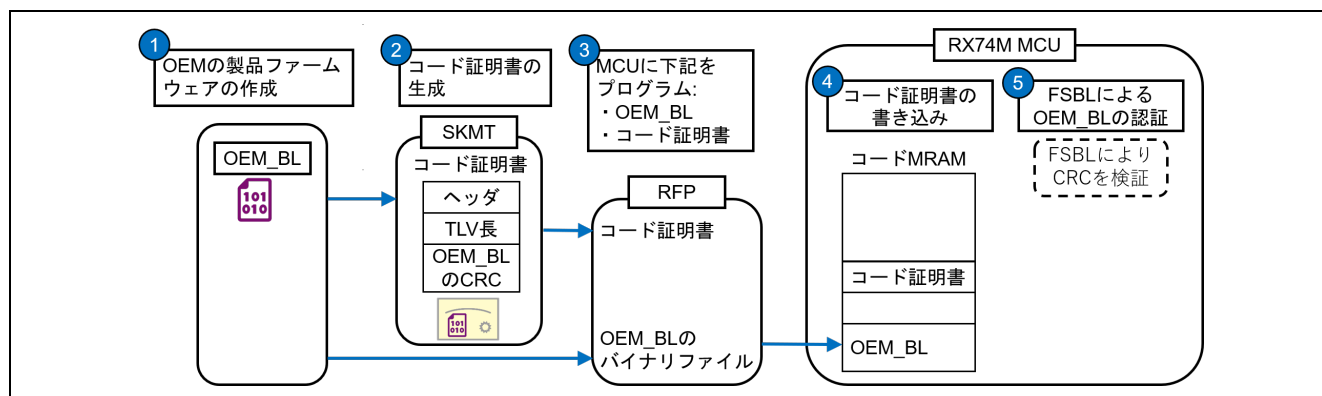


図1-2 CRC ブート製品へのプログラミング認証情報の概要

1.3.2 HMAC ブートにおける認証情報の適用フロー

OEM_BL のプログラミング段階では、以下の項目が MCU にプログラムされます。

- OEM_ROOT_PK のハッシュ
- OEM_BL(アプリケーションイメージ)
- コード証明書

以下の図は、HMAC ブートを使用した製品のプログラミングに SKMT と RFP を使用した全体的なワークフローをまとめたものです。SKMT と RFP の使用法についての詳しい手順は、4章で説明します。

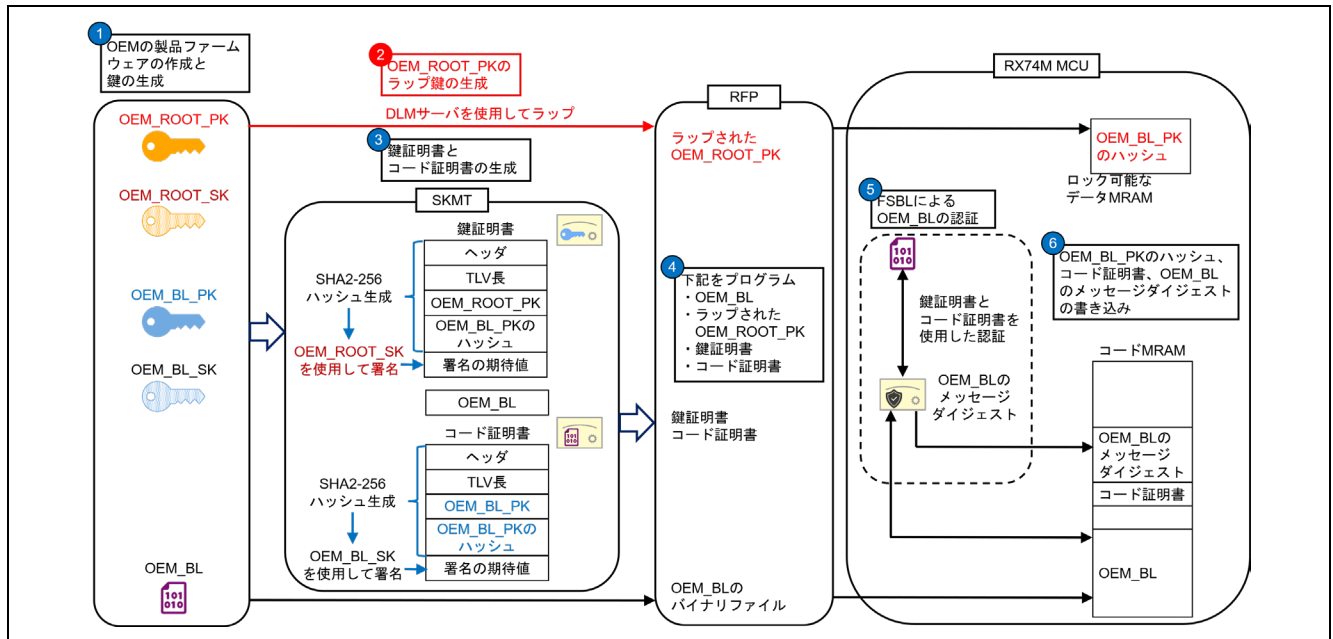


図1-3 HMAC ブート製品のプログラミング認証情報の概要

2. RX74M FSBL のセキュリティモデルと動作

HMAC ブートでは、OEM_BL の真正性と完全性を保証するために、複数の認証情報を用いた Chain of Trust を構築します。システムの起動時には、それぞれのコンポーネントが次のコンポーネントを検証することで、最終的に実行される OEM_BL およびアプリケーションが正当なものであることを保証します。

本章では、HMAC ブートのセキュリティモデルについて説明します。まずシステムの信頼の起点となる Root of Trust を説明し、その後、鍵証明書、コード証明書、およびアンチロールバック機能による保護方法について説明します。これらの認証情報がどのように連携して Chain of Trust を形成するかを理解することで、以降の証明書生成や製品プログラミング手順を容易に理解できます。

FSBL の HMAC ブートにおける Chain of Trust の概要を以下に示します。Chain of Trust は、変更できない領域に書かれた FSBL を起点として構築されます。FSBL は OEM_BL 検証ルート鍵ペアの公開鍵に基づいて OEM_BL を検証し、検証に成功した場合のみ OEM_BL の実行を許可します。

ここで、CRC ブートについては、OEM_BL の正当性を検証するものではないため、本章では HMAC ブートを中心に説明を記載しています。

2.1 信頼の基点と Chain of Trust

HMAC ブートでは、システムが起動する前に実行対象のソフトウェアが正規の開発者によって生成されたものであり、かつ改ざんされていないことを確認する必要があります。この仕組みは Chain of Trust と呼ばれ、信頼できるコンポーネントが次のコンポーネントを順次検証することで実現されます。

RX74M の FSBL を使用したブート処理では、変更できない領域に書かれた FSBL を信頼の起点とし、OEM_BL 検証ルート鍵ペア(OEM_ROOT_PK)の公開鍵、鍵証明書、コード証明書を利用して OEM_BL の真正性および完全性を検証します。検証に成功した場合のみ OEM_BL が実行されます。

図2-1に FSBL における Chain of Trust を示します。

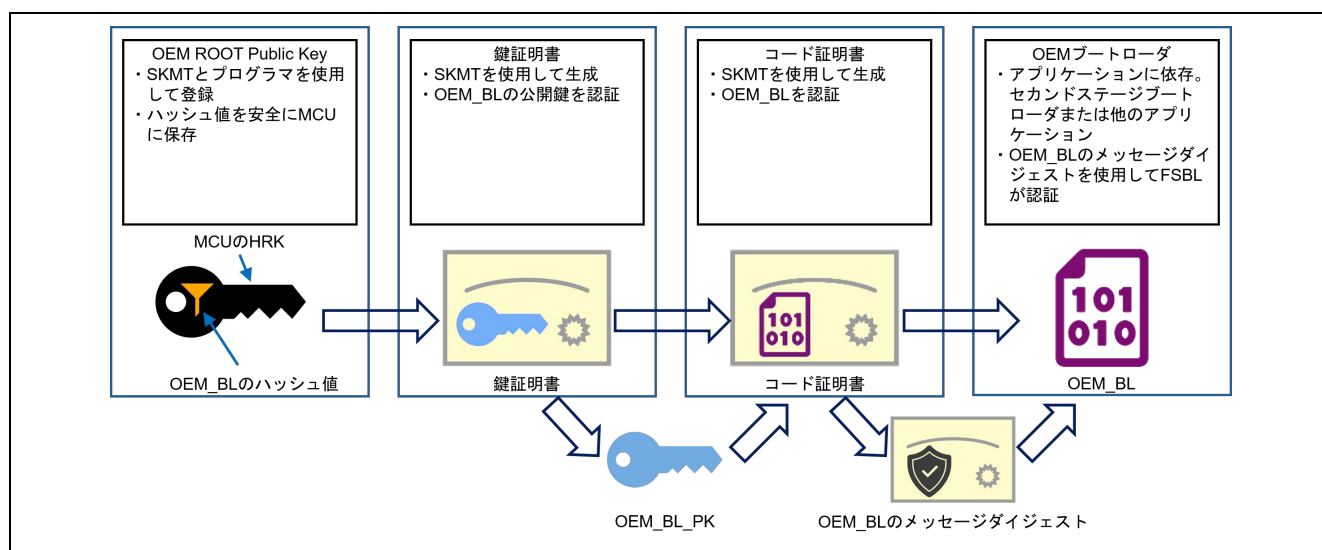


図2-1 Chain of Trust

Chain of Trust を構成する主要な認証情報は以下のとおりです。

- ・ Root of Trust
システムの信頼の起点となる OEM_ROOT_PK の SHA-256 ハッシュ値。
- ・ 鍵証明書
OEM_BL 検証鍵ペアの公開鍵(OEM_BL_PK)の正当性を保証するために使用される証明書。
- ・ コード証明書
OEM_BL イメージの検証情報を保持する証明書。
- ・ OEM_BL_Digest
HMAC ブート有効時に実行時認証へ使用される HMAC-SHA256 ダイジェスト。

以下の節では、これらの認証情報の構造および役割について説明します。

2.2 HMAC ブートで使用する鍵と認証情報

2.2.1 OEM_BL 検証ルート鍵と OEM_BL 検証鍵

HMAC ブートを使用する場合は、2 セットの secp256r1 ECC 鍵ペアを生成する必要があります。

- OEM_BL 検証ルート鍵ペア
 - OEM_ROOT_PK: OEM_BL 検証ルート鍵ペアの公開鍵(Public Key)
 - OEM_ROOT_SK: OEM_BL 検証ルート鍵ペアの秘密鍵(Secret Key)
- OEM_BL 検証鍵ペア
 - OEM_BL_PK: OEM_BL 検証鍵ペアの公開鍵(Public Key)
 - OEM_BL_SK: OEM_BL 検証鍵ペアの秘密鍵(Secret Key)

OEM_ROOT_PK の SHA256 ハッシュは、RX74M グループ ユーザーズマニュアル ハードウェア編の「セキュア鍵注入」章で説明されているプロセスを使用して、オプション設定メモリ(HOEMRTPK)に格納されません。Root of Trust の注入は、アプリケーション製品へプログラミング中に行われます。

2.2.2 鍵証明書

鍵証明書は、HMAC ブートを使用する製品のプログラミング時に、OEM_BL_PK の正当性を検証するために使用されます。

以下は、鍵証明書の作成と使用に関する重要な情報です。

- 鍵証明書は OEM_ROOT_SK によって署名されます。
- OEM_ROOT_PK は署名された鍵証明書に含まれ、HOEMRTPK に格納された値によって検証されます。
- OEM_BL_PK のハッシュは鍵証明書に含まれており、OEM_BL によって OEM_BL_PK を検証するために使用されます。
- 鍵証明書は OEM_BL_PK が検証された後に破棄されます。

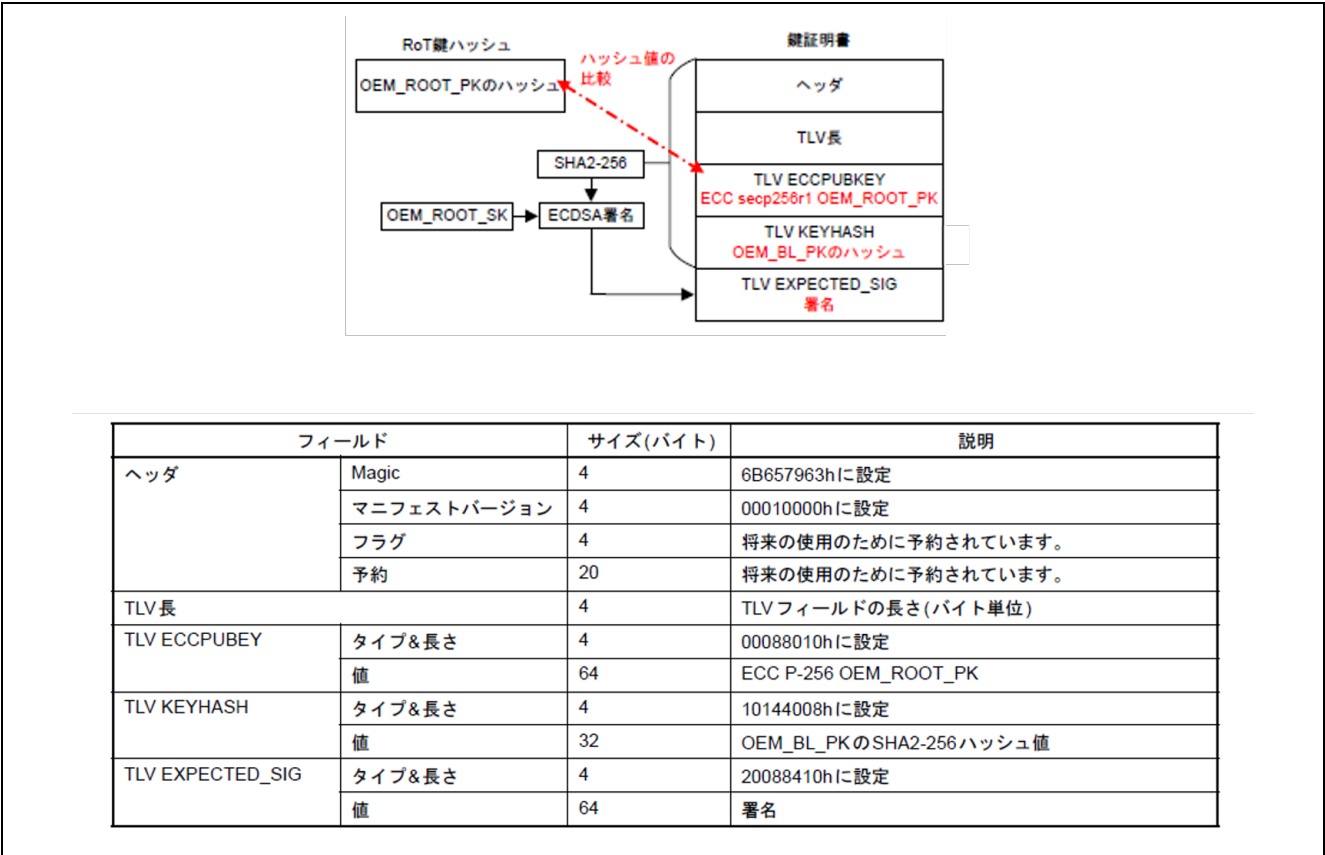


図2-2 OEM_BL_PK を検証するための鍵証明書

2.2.3 コード証明書

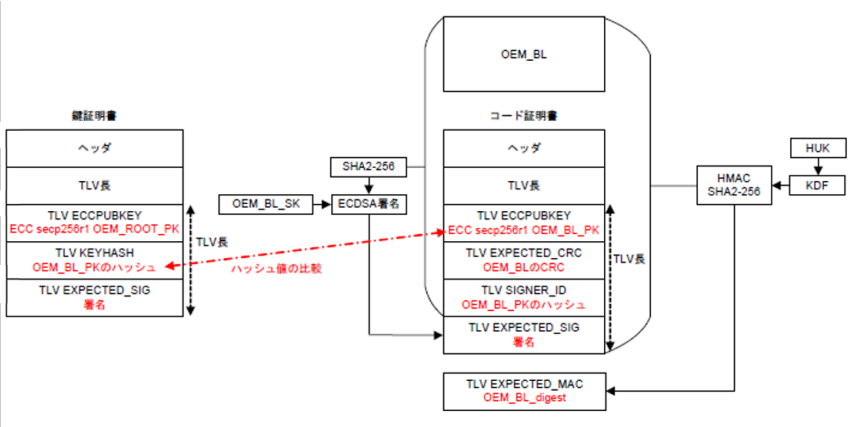
コード証明書は、OEM_BL の検証に必要な情報が格納されています。コード証明書は CRC ブートと HMAC ブートの両方で使用され、製品プログラミング時およびシングルチップモードでの起動時に参照されます。

- コード証明書は OEM_BL_SK によって署名されています。
- OEM_BL_PK はコード証明書に含まれています。鍵証明書に保存されたハッシュによって検証されます。その後、OEM_BL_PK はコード証明書に含まれている OEM_BL の署名検証に使用します。
- コード証明書には、シングルチップモード中の FSBL 検証プロセスに使用される認証情報が含まれています。OEM_BL がプログラムされた後、MRAM(コード MRAM またはデータ MRAM)にプログラムする必要があります。
- HMAC ブートを使用する際、アプリケーションのイメージバージョンはコード証明書の一部となります。有効なイメージバージョンは FSBL の検証対象によって異なります。FSBL の検証対象が SSBL の場合 1～64、アプリケーションの場合 1～1024 になります。FSBL の検証対象が SSBL の場合、64 回更新することになります。FSBL はアンチロールバックスキームを実装しています。したがって、新しい OEM_BL イメージに更新するには、次の新しいイメージのバージョンが前のアプリケーションイメージバージョンより新しくなければなりません。イメージバージョンの操作や使用方法についての詳細は、RX74M グループ ユーザーズマニュアル ハードウェア編の「アンチロールバックカウンタ」章を参照してください。
- MRAM 内のコード証明書の場所は、SACC0 レジスタ、SACC1 レジスタで指定する必要があります。SACC0 レジスタは、デュアルバンクモードで下位バンクがバンク 0 の場合、またはリニアモードでスタートアップ領域が通常エリアの場合に、コード証明書の開始アドレスを指定します。SACC1 レジスタは、デュアルバンクモードで下位バンクがバンク 1 の場合、またはリニアモードでスタートアップ領域が代替エリアの場合に、開始アドレスを指定します。SACCx レジスタ設定は、RX FSP のボードサポートパッケージ(BSP)によって設定することができます。
- 図3-1と図4-6にコード証明書プログラミングの位置の構成例を示しています。特定のアプリケーションでは必要に応じて調整してください。
- HMAC ブートでは、TLV EXPECTED_CRC フィールドは使用されませんが、コード証明書の構成要素としては残ります。
- 製品へプログラミング中に OEM_BL の検証に成功すると、OEM_BL は MCU の HUK を使用して、OEM_BL とコード証明書の HMAC-SHA256 ダイジェスト(OEM_BL_digest)を計算します。OEM_BL は、OEM_BL_digest をコード証明書の直後のメモリ領域にプログラムします。

フィールド		サイズ(バイト)	説明
TLV EXPECTED_MAC	タイプ&長さ	4	固定値 30184008h
	値	32	各 MCU の一意の OEM_BL_digest

図2-3 OEM_BL_digest の詳細

HMAC ブートのコード証明書の構成を、図2-4に示します。



フィールド		サイズ(バイト)	説明
ヘッダ	Magic	4	636F6463hに設定
	マニフェストバージョン	4	00010000hに設定
	フラグ	4	00000000hに設定
	Load Addr	4	OEM_BLの開始アドレスに設定
	Dest Addr	4	OEM_BLの開始アドレスに設定
	イメージサイズ	4	OEM_BLのバイトサイズ。16の倍数で設定してください。最小サイズは64バイト
	イメージバージョン	4	OEM_BLのバージョン番号 設定値はARCMDの設定に依存
	ビルド番号	4	00000000hに設定
TLV長		4	TLVフィールドのバイト長
TLV ECCPUBKEY	タイプ&長さ	4	01088010hに設定
	値	64	ECC secp256r1 OEM_BL_PK
TLV EXPECTED_CRC	タイプ&長さ	4	40000001hに設定
	値	4	OEM_BLのCRC32
TLV SIGNER_ID	タイプ&長さ	4	10144008hに設定
	値	32	OEM_BL_PKのSHA2-256ハッシュ値
TLV EXPECTED_SIG	タイプ&長さ	4	25088410hに設定
	値	64	OEM_BL_SKによって署名した[コード証明書 OEM_BL]署名の生成時、コード証明書、OEM_BLの順番でハッシュ関数に入力します。

図2-4 HMAC ブートを使用して OEM_BL を検証するためのコード証明書

CRC ブートのコード証明書を、図2-5に示します。TLV ECCPUBKEY と TLV_EXPECTED_SIG フィールドは、コード証明書には存在しません。また CRC ブートではアンチロールバックポリシーは適用されないため、イメージバージョンフィールドは使用されません。

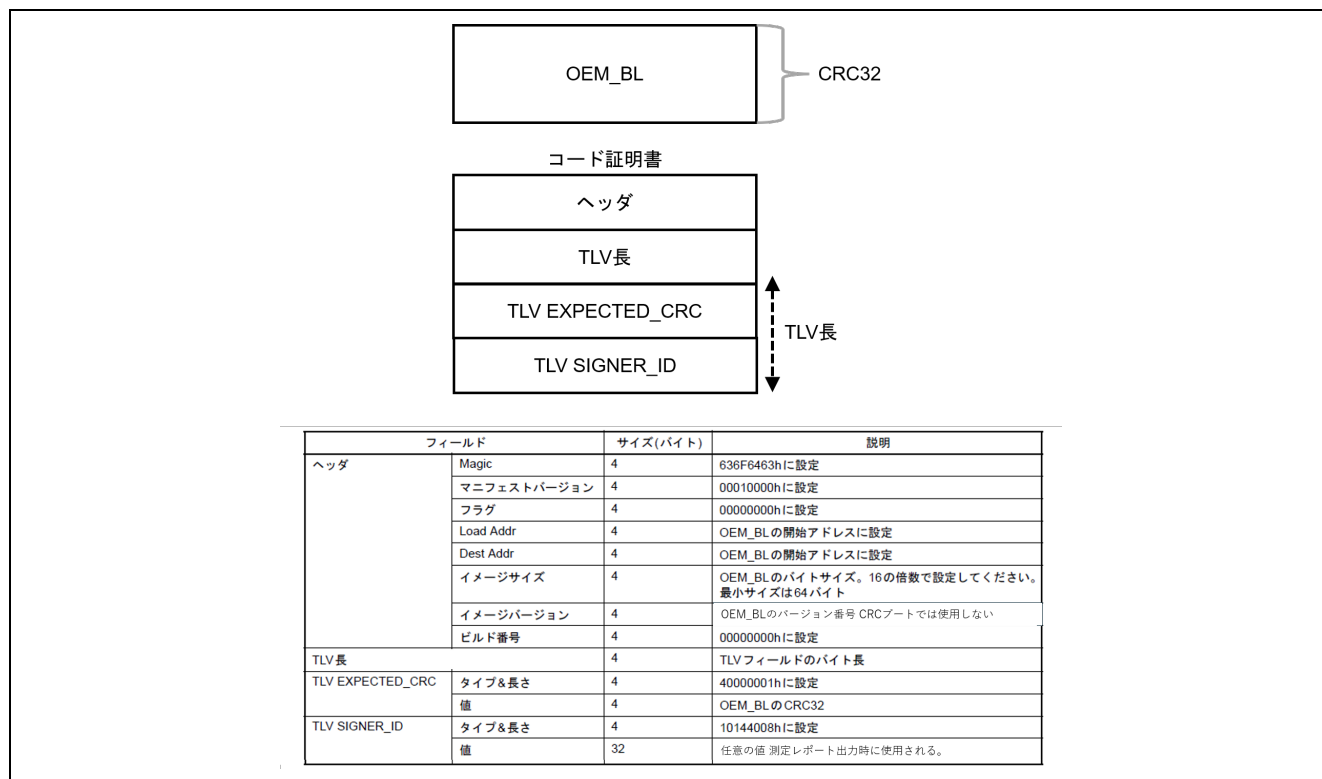


図2-5 CRC ブートを使用して OEM_BL を検証するためのコード証明書

2.3 シングルチップモードでの起動時検証

HMAC ブートの場合、製品へのプログラミング中、OEM_BL は ECDSA を使用してブートファームウェアによって検証されます。RX74M のブートモードは、JTAG、SCI、および USB インタフェースから利用できます。ブートモードおよび各インタフェースの詳細については、RX74M グループ ユーザーズマニュアル ハードウェア編の「動作モード」章を参照してください。

OEM_BL を起動する前に、ユーザは例えば RFP を使用して、OEM_ROOT_PK のハッシュをシステムの HOEMRTPKn というエリアにプログラムします。その後、OEM_BL が MCU にプログラムされる時に、ブートファームウェアはコード証明書のイメージバージョンを読み取ります。

イメージバージョンが以前にプログラムされたバージョンよりも新しいバージョンである場合、ブートファームウェアは、OEM_ROOT_PK(鍵証明書に保存されている)のハッシュを計算し、MCU にプログラムされた OEM_ROOT_PK のハッシュと比較します。

これにより、コード証明書の OEM_BL_PK が検証されます。OEM_BL_PK が検証されると、ブートファームウェアがコード証明書の署名を使用して OEM_BL を検証します。

OEM_BL が検証されると、ブートファームウェアはアンチロールバックカウンタをコード証明書にプログラムされたイメージバージョンまでインクリメントし、OEM_BL_digest を計算します。その後、コード証明書と OEM_BL_digest が MCU にプログラムされます。

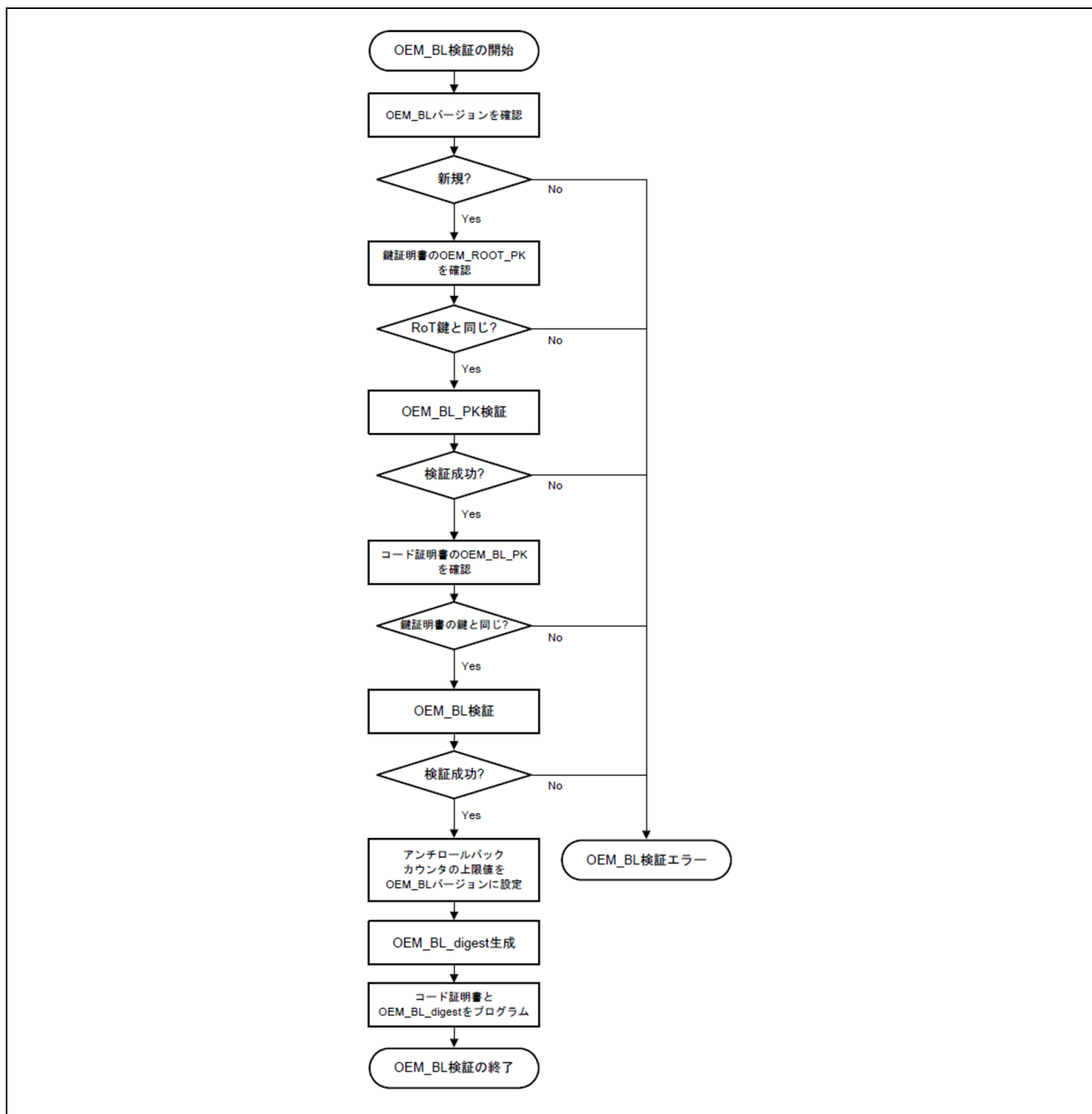


図2-6 製品プログラミングモードでの OEM_BL 検証フロー

製品プログラミングが正常に完了した後、RX74M をシングルチップモードで起動し、FSBL が有効になっていると、リセット後に FSBL が実行されます。FSBL は、選択されているブート方式に従って OEM_BL を検証します。

CRC ブートを使用する場合、FSBL は MRAM に格納されている OEM_BL の CRC32 値を計算し、コード証明書に格納されている期待値と比較します。

HMAC ブートを使用する場合、FSBL は OEM_BL およびコード証明書を対象として HMAC-SHA256 を計算し、製品プログラミング時に MRAM へ格納された OEM_BL_digest と比較します。検証値が一致した場合、FSBL は OEM_BL へ制御を渡します。

図2-7に、シングルチップモードにおける OEM_BL の検証フローを示します。

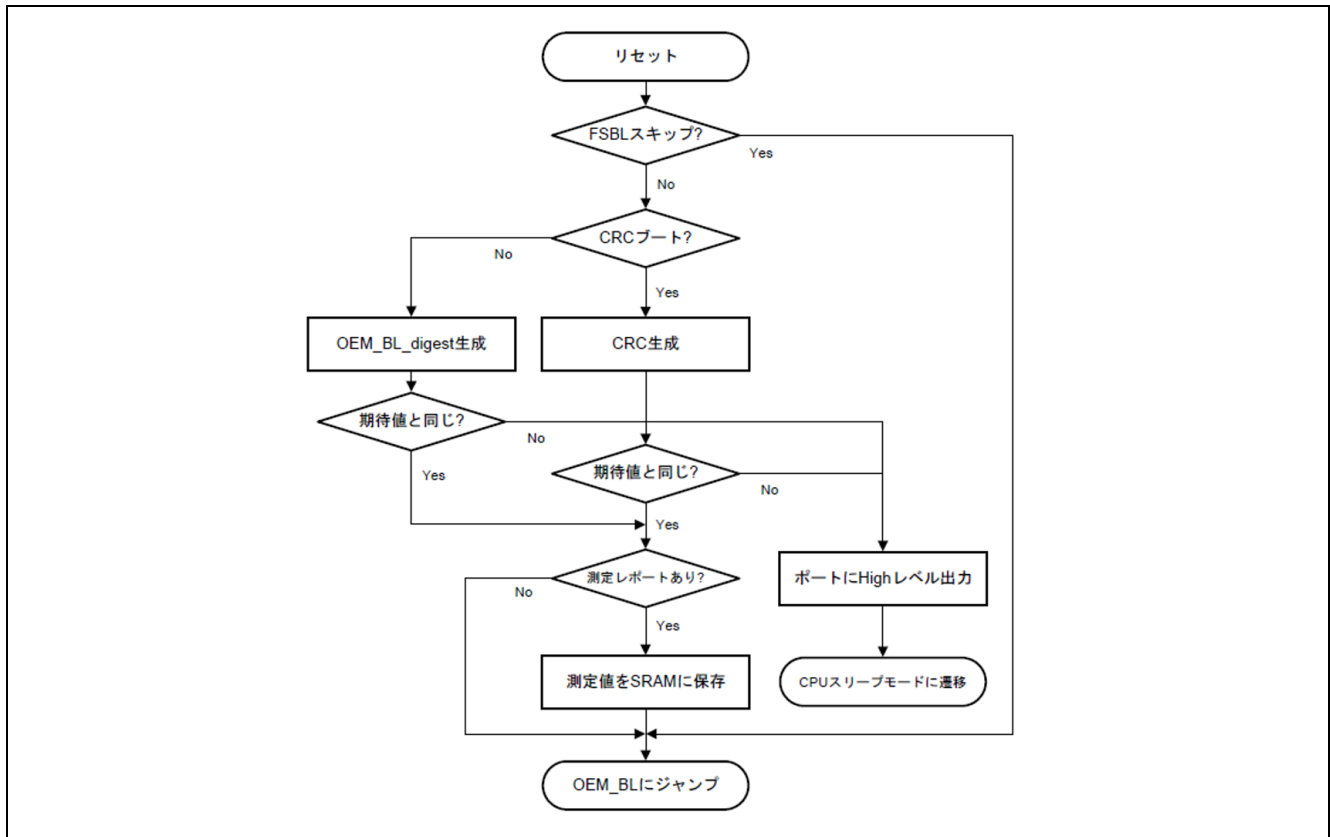


図2-7 シングルチップモードでの OEM_BL の検証

各ブート方式では、検証結果を測定レポートとして出力する構成も選択できます。設定方法とレポート形式は5.2節を参照してください。

3. CRC ブートのサンプルプロジェクト

本章では、CRC ブートのサンプルプロジェクトの使用方法を説明します。

本章においては、コンパイラとして CC-RX を使用する場合を記載していますが、GCC においても操作内容は同様です。

3.1 RX74M FSBL の CRC ブート設定

CRC ブートを使用する Blinky プロジェクト(`blinky_crc_boot_rx74m_ccrx`)は、FSP の Blinky プロジェクトを元に作成したプロジェクトです。FSBL を有効にするため、図3-1のように FSBL Execution Mode を **CRC boot with report measurement** に設定します。また、エラーレポート用端子として P604 端子も設定しています。

`blinky_crc_boot_rx74m_ccrx` をワークスペースにインポートし、`configuration.xml` ファイルを実行して、**Generate Project Content** をクリックします。`blinky_crc_boot_rx74m_ccrx` をビルドして実行し、その機能を確認します。

- 生成されたモトローラヘキサファイルは、3.2節でコード証明書の作成に使用されます。
- 3つのLEDが点滅していることを確認することで、サンプルプロジェクトが起動していることを確認することができます。

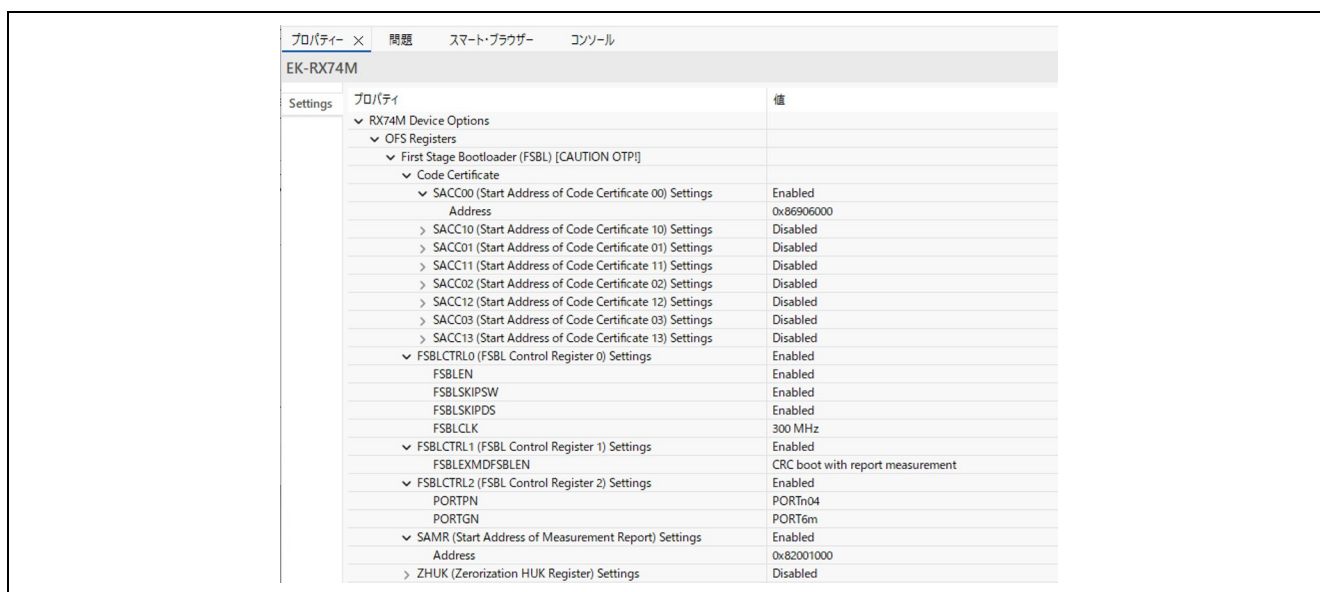


図3-1 CRC ブート用の FSBL 設定

Tera Term を開き、スピードを **115200**、データを **8 bit** に設定します。

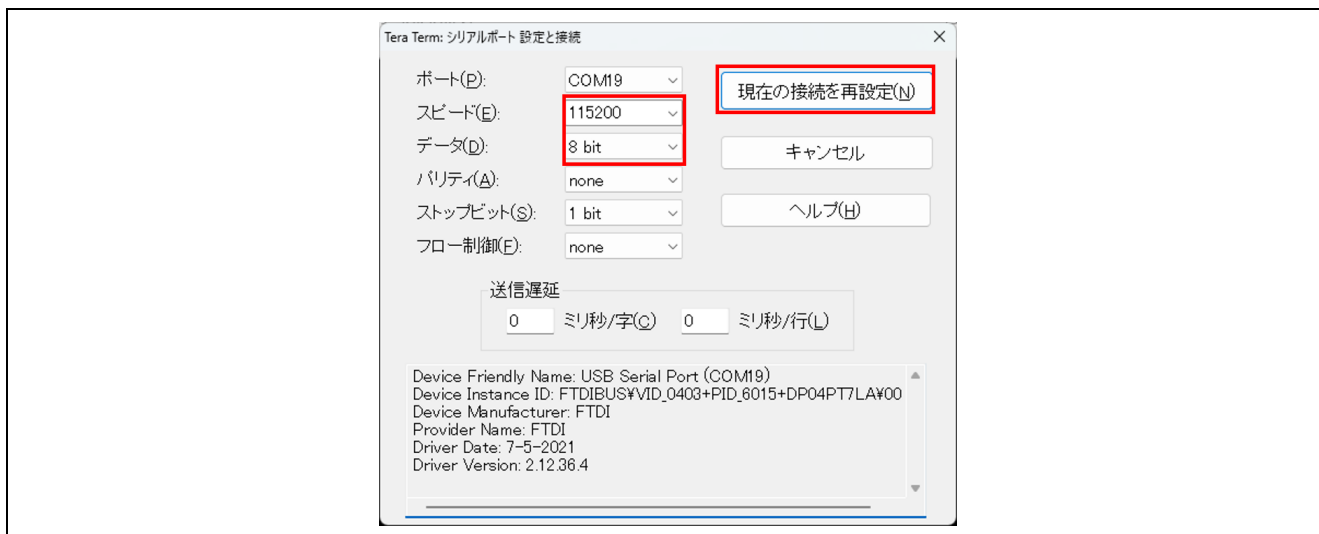


図3-2 Tera Term の設定

次の Tera Term の出力を確認してください。コード証明書はまだプログラムされていないので、CRC の出力は不定値になります。また、イメージバージョン情報は CRC ブートでは使用されないため、`version_num_oem_bl` の出力はすべて 0 です。

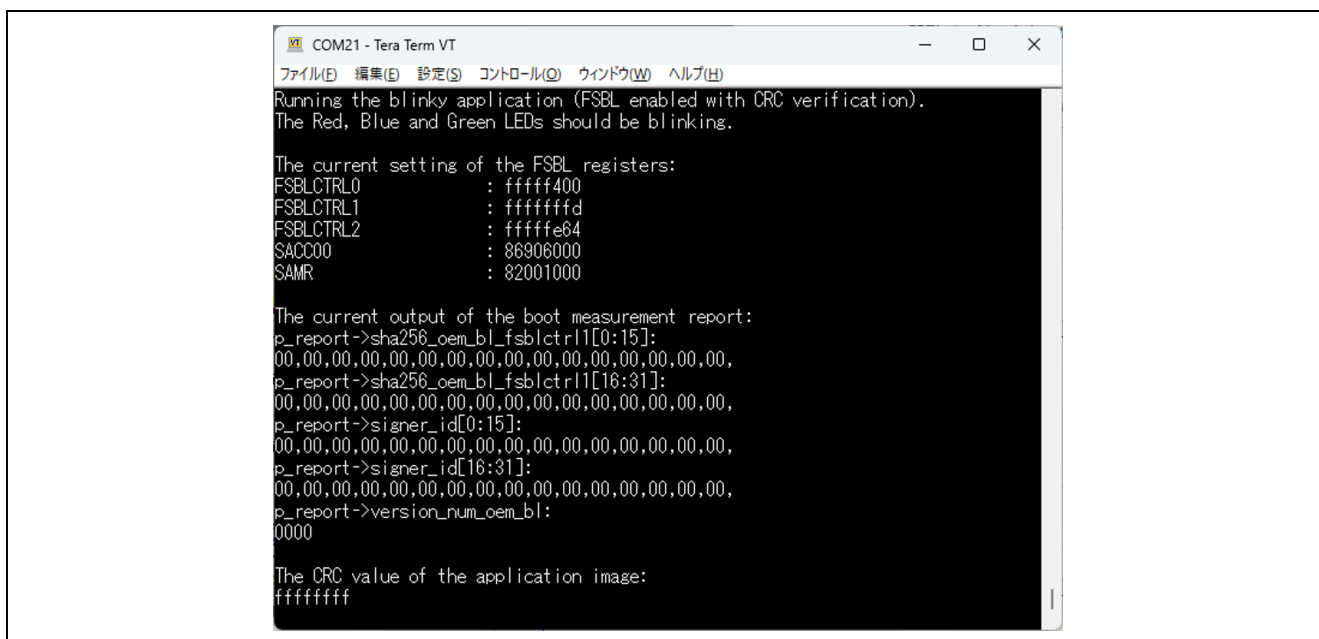


図3-3 CRC ブートのコンソール出力

3.2 コード証明書の作成

CRC ブート用のコード証明書には、OEM_BL の CRC 期待値が含まれている必要があります。MCU リセット時、完全性チェックが実行されている場合、FSBL は OEM_BL の CRC を計算し、保存されているコード証明書の値と比較します。

コード証明書は、CLI または GUI の SKMT を使用して作成します。

SKMT CLI を使用した CRC ブート用コード証明書の作成

コード証明書を生成するには、いくつかのプロパティを設定する必要があります。

- 入力:/loadaddr OEM_BL のロードアドレス。
- 入力:/cfsize 使用するデバイスのコード MRAM サイズ。
- 入力:/oembl OEM_BL のバイナリファイル。
- 出力:/output_codecert コード証明書。

コマンドウィンドウを開き、\SecurityKeyManagementTool\cli フォルダに移動し、以下のコマンドを実行して、3.1節で作成したプロジェクトに基づいてコード証明書を生成します。

```
C:\Renesas\SecurityKeyManagementTool\cli>skmt.exe /gencert /mode "CRC" /loadaddr "FFF00000"  
/cfsize "100000" /oembl  
"C:\PARENT_LOC\blinkycrcboot_rx74m_ccrx\Debug\blinkycrcboot_rx74m_ccrx.mot"  
/output_codecert "C:\RX74M_FSBL\certificates\blinkycrcboot_rx74m_code_cert.bin"  
Output File: C:\RX74M_FSBL\certificates\blinkycrcboot_rx74m_code_cert.bin
```

SKMT GUI を使用した CRC ブート用コード証明書の作成

GUI を開き、FSBL タブに移動します。プログラム検証方法で **CRC** を選択し、コード証明書で出力ファイルを設定します。ファイルを生成する をクリックするとコード証明書を生成します。

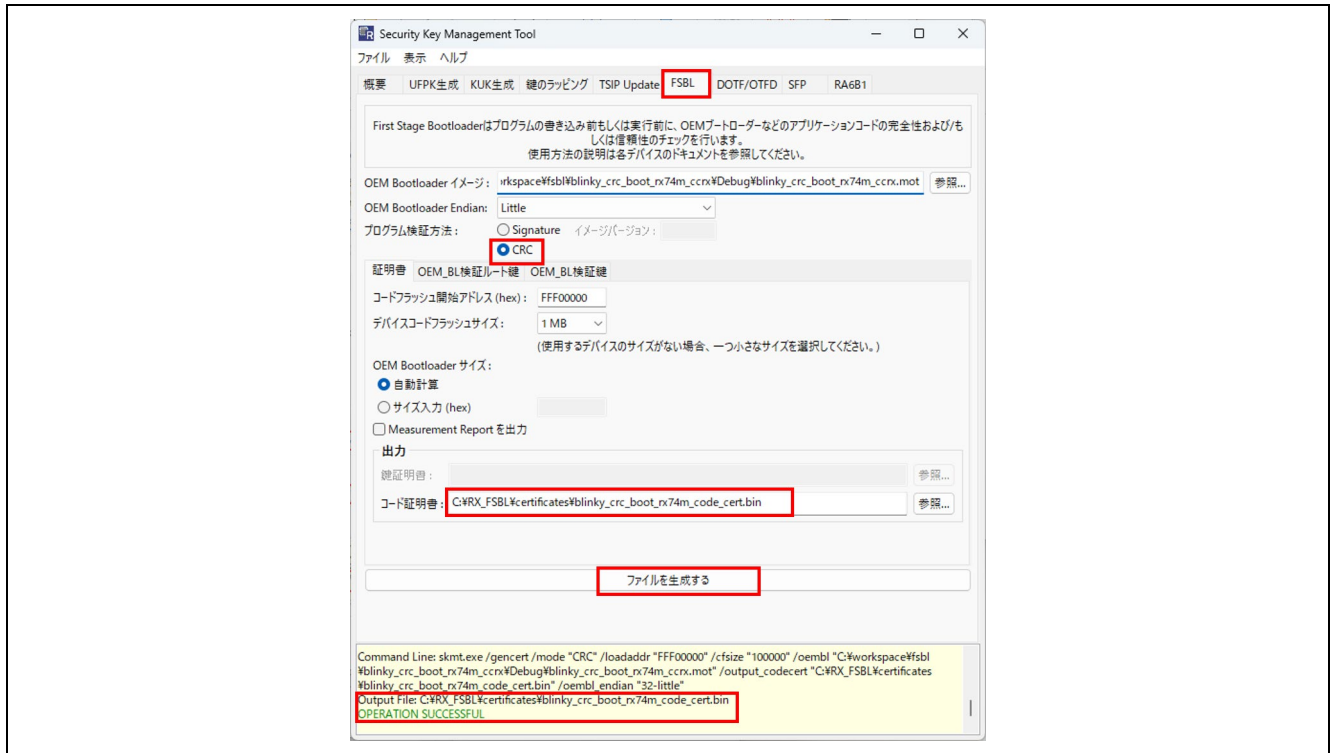


図3-4 CRC ブート用コード証明書の作成

CRC を指定してコード証明書を生成した場合、CRC 値はコード証明書の署名者 ID フィールドのダミー値として出力されます。CRC 検証のために作成される測定レポートでは署名者 ID は CRC 値の 8 倍の長さで使用されます。CRC32 は 4 バイト(32 ビット)で、署名者 ID(SHA256)は 32 バイトなので、CRC 値は署名者 ID フィールドに 8 回書き込まれます。

3.3 RFP によるプログラミングと動作確認

RFP を起動し、付属の `rfp_rx74m.rpj` を開きます。アプリケーションを MCU にダウンロードし、MCU の FINE インタフェースブートモードを介してアプリケーションを認証して起動するための認証情報を提供します。

以下の操作設定タブが構成されていることを確認します。

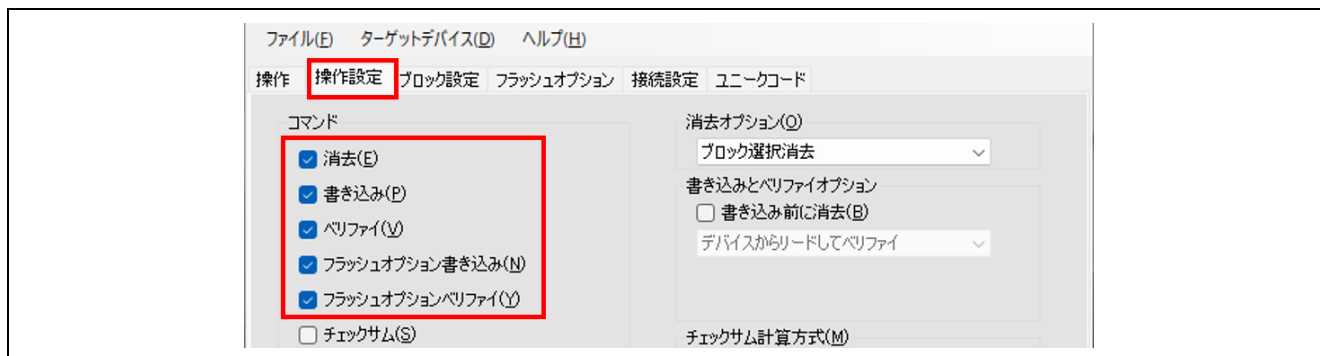


図3-5 アプリケーションをダウンロードする前の動作設定の構成

操作タブで、MCU にプログラムするアプリケーションプロジェクトを選択します。

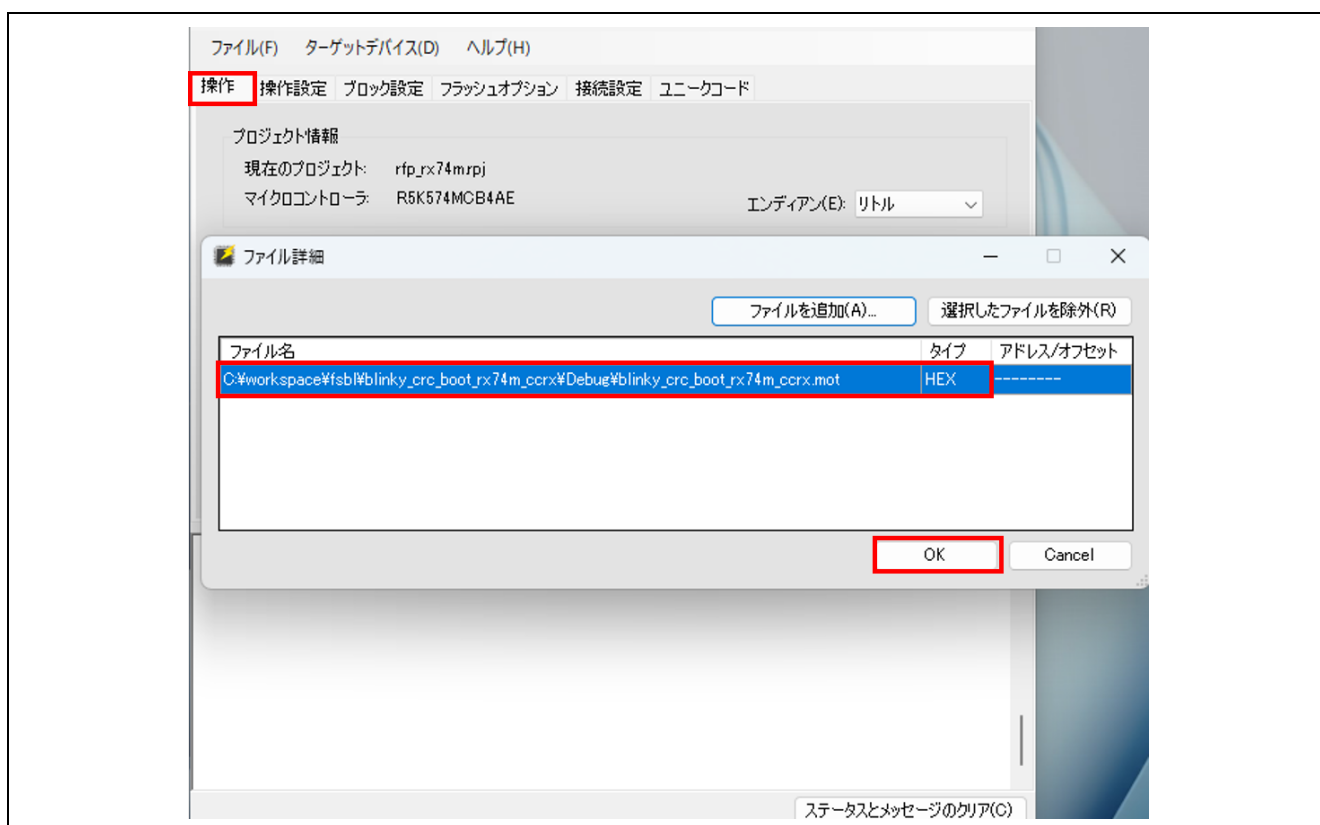


図3-6 アプリケーションプロジェクトの選択

OK をクリックし、フラッシュオプションタブに移動して **Certificate** の項目を設定します。

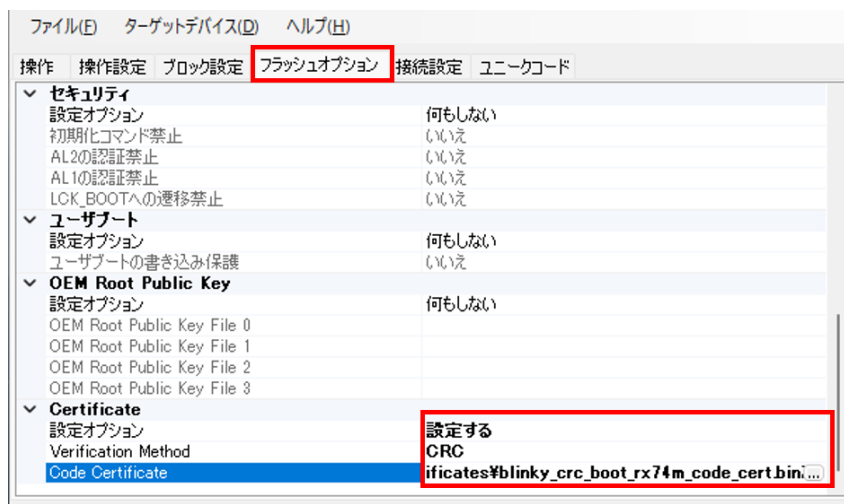


図3-7 CRC コード証明書の設定

次に、操作タブに移動し、スタートを選択すると、コード証明書と同様にアプリケーションがプログラミングされます。3つのLEDが点滅し、正常動作を示します。

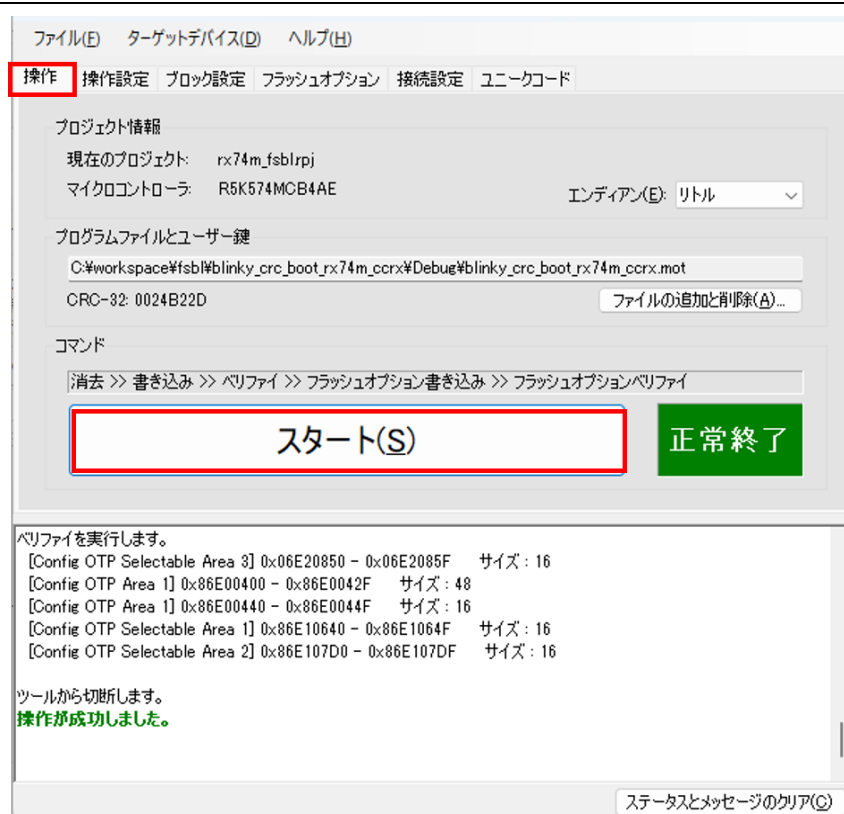
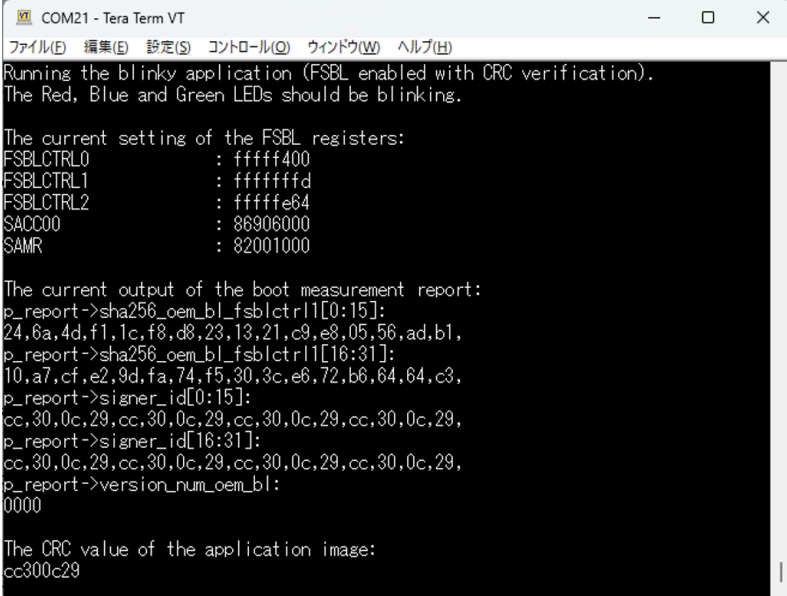


図3-8 CRC ブートアプリケーションのダウンロード

Tera Term のコンソールには、以下のようなメッセージが表示されます。CRC 値が `signer_id` として 8 回繰り返されていることに注意してください。



```
COM21 - Tera Term VT
ファイル(F) 編集(E) 設定(S) コントロール(C) ウィンドウ(W) ヘルプ(H)
Running the blinky application (FSBL enabled with CRC verification).
The Red, Blue and Green LEDs should be blinking.

The current setting of the FSBL registers:
FSBLCTRL0      : fffff400
FSBLCTRL1      : ffffffff
FSBLCTRL2      : fffffe64
SACCC00        : 86906000
SAMR           : 82001000

The current output of the boot measurement report:
p_report->sha256_oem_bl_fsblctrl1[0:15]:
24,6a,4d,f1,1c,f8,d8,23,13,21,c9,e8,05,56,ad,b1,
p_report->sha256_oem_bl_fsblctrl1[16:31]:
10,a7,cf,e2,9d,fa,74,f5,30,3c,e6,72,b6,64,64,c3,
p_report->signer_id[0:15]:
cc,30,0c,29,cc,30,0c,29,cc,30,0c,29,cc,30,0c,29,
p_report->signer_id[16:31]:
cc,30,0c,29,cc,30,0c,29,cc,30,0c,29,cc,30,0c,29,
p_report->version_num_oem_bl:
0000

The CRC value of the application image:
cc300c29
```

図3-9 証明書を使用した CRC ブートのコンソール出力

RFP で計算される CRC32 値と、SKMT が計算するコード証明書の CRC32 値は一致しません。RFP はファイル全体を対象に CRC32 を計算します。例えば、`.srec` ファイルではすべてのレコードフィールドが計算対象となります。一方、コード証明書の CRC 値は `OEM_BL` のバイナリデータのみを対象に CRC32 を計算します。

CRC ブートの失敗を確認する場合、RFP の初期化機能を使用してデバイスを初期化し、3.3節の手順を実行してください。CRC ブートは失敗し、P604 端子は”High”になり、赤色の LED が点灯します。

4. HMAC ブートのサンプルプロジェクト

本章では、HMAC ブートのサンプルプロジェクトの使用方法を説明します。

Root of Trust(OEM_ROOT_PK の SHA256 ハッシュ)を導入する手順は、アプリケーションノート R01AN8067 で説明されているセキュア鍵注入プロセスと多くの共通点があるので、必要に応じて R01AN8067 の各章を参照してください。

本アプリケーションノートのサンプルプロジェクトでは、NIST CAVP テストベクタの値を例として使用しています。元のテストベクタの値を確認される場合は、以下のリンクからダウンロードしてください。

[Cryptographic Algorithm Validation Program | CSRC \(nist.gov\)](https://csrc.nist.gov/Cryptographic-Algorithm-Validation-Program)

本章においては、コンパイラとして CC-RX を使用する場合を記載していますが、GCC においても操作内容は同様です。

4.1 鍵の準備

2.2.1節で説明したように、システムの Chain of Trust を構築するには2セットの ECC secp256r1 鍵ペアが必要です。本アプリケーションノートでは、OpenSSL を使用して鍵ペアを生成する方法ならびに、NIST CAVP テストベクタの鍵ペアの使用方法について説明します。

4.1.1 OpenSSL を使用した ECC 鍵ペアの生成

評価用 PC の OS が Linux の場合は、次のリンクから OpenSSL を取得できます

<https://www.openssl.org/source/>

Windows の場合は、次のリンクから OpenSSL を取得できます

<https://slproweb.com/products/Win32OpenSSL.html>

OpenSSL のダウンロード後、インストールし、コマンドプロンプトから任意のフォルダに移動し、OpenSSL のコマンドを使用します。

以下のコマンドで OEM_BL 検証ルート鍵ペアを生成します。

```
openssl ecparam -name prime256v1 -genkey -noout -out oem_root_private_key.pem
```

以下のコマンドで OEM_BL 検証ルート鍵ペアの公開鍵(OEM_ROOT_PK)を生成します。

```
openssl ec -in c:\RX74M_FSBL\openssl_keys\oem_root_private_key.pem -pubout -out oem_root_public_key.pem
```

同じコマンドを使用して OEM_BL 検証鍵ペアを生成することができます。

```
openssl ecparam -name prime256v1 -genkey -noout -out oem_bl_private_key.pem
```

以下のコマンドで、OEM_BL 検証鍵ペアの公開鍵(OEM_BL_PK)を生成します。

```
openssl ec -in c:\RX74M_FSBL\openssl_keys\oem_bl_private_key.pem -pubout -out oem_bl_public_key.pem
```

4.1.2 ECC 鍵ペアとする NIST CAVP テストベクタ

本アプリケーションノートのサンプルプロジェクトでは、NIST CAVP テストベクタの値を例として使用しています。元のテストベクタの値を確認される場合は、以下のリンクからダウンロードしてください。

[Cryptographic Algorithm Validation Program | CSRC \(nist.gov\)](https://csrc.nist.gov/Cryptographic-Algorithm-Validation-Program)

ここでは以下の鍵ペアを使用します。

OEM_BL 検証ルート鍵ペア用の ECC P-256(secp256r1)鍵ペア

以下の鍵ペアは、OEM_BL 検証ルート鍵ペアに使用されます。これは4.2節でラッピングされた OEM_ROOT_PK の生成に使用されます。また、4.5節で鍵証明書を生成する際にも使用されます。

```
d = c9806898a0334916c860748880a541f093b579a9b1f32934d86c363c39800357
Qx = d0720dc691aa80096ba32fed1cb97c2b620690d06de0317b8618d5ce65eb728f
Qy = 9681b517b1cda17d0d83d335d9c4a8a9a9b0b1b3c7106d8f3c72bc5093dc275f
```

OEM_BL 検証鍵ペア用の ECC P-256(secp256r1)鍵ペア

以下の鍵ペアは、OEM_BL 検証鍵ペアに使用されます。これは、4.5節でコード証明書を生成するために使用されます。

```
d = 710735c8388f48c684a97bd66751cc5f5a122d6b9a96a2dbe73662f78217446d
Qx = f6836a8add91cb182d8d258dda6680690eb724a66dc3bb60d2322565c39e4ab9
Qy = 1f837aa32864870cb8e8d0ac2ff31f824e7beddc4bb7ad72c173ad974b289dc2
```

4.2 Root of Trust の導入

前述のように、OEM_ROOT_PK のハッシュはシステムの Root of Trust です。FSBL が有効になっている場合、アプリケーションのプログラミングが成功するとこのハッシュが注入されます。

本節の操作を実施する前に、ルネサス Key Wrap サービスを使用してユーザ・ファクトリ・プログラミング・キー (User Factory Programming Key、UFPK) をラップする手順を実行してください。UFPK ラップのセキュリティ考慮事項やプロセスについては、アプリケーションノート R01AN8067 の以下の章を参照してください。使用背景を理解し、ルネサス Key Wrap サーバとの PGP 暗号化通信を確立することができます。

- 顧客の PGP 鍵ペアを生成：「PGP 鍵ペアの作成」章
- ルネサス Key Wrap サーバに登録：「ルネサス Key Wrap サーバへの登録」章
- ルネサス Key Wrap サーバと暗号化通信を確立：「ユーザとルネサスの PGP 公開鍵交換」章

4.2.1 ラッピングされた UFPK (Wrapped User Factory Programming Key、W-UFPK) の生成

OEM_ROOT_PK を RX74M へ安全に注入するには、UFPK を使用して OEM_ROOT_PK をラッピングする必要があります。UFPK および W-UFPK の生成方法については、アプリケーションノート R01AN8067 の「UFPK の生成と W-UFPK の取得」を参照してください。RX74M の OEM_ROOT_PK の鍵注入のための、UFPK および W-UFPK を生成する場合は、R01AN8067 の手順に対して次の設定を適用してください。

1. SKMT の概要画面で **RX Family, RSIP-E50D Security Functions and Protected Mode** を選択します。



図4-1 RSIP-E50D セキュリティ機能とプロテクトモードの選択

2. ルネサス Key Wrap サービスでは、**RSIP Compatibility Mode** ではなく、**RSIP Protected Mode** にある RX74M 用の項目を選択します。

PGP鍵交換 処理履歴表示 再表示 戻る

RSIP Protected Mode

Protected Modeおよび下記セキュリティ機能で使用する鍵のインジェクション向け

- ・デバイスライフサイクル管理 (DLM)
- ・セキュアファクトリプログラミング(SFP)
- ・オンザフライ復号(DOTF) (Protected Modeを使用する場合)
- ・ファーストステージブートローダ(FSBL)

RX74M Wrap UFPK

RX74N Wrap UFPK

RX261 Wrap UFPK

RSIP Compatibility Mode

Compatibility Modeおよび下記セキュリティ機能で使用する鍵のインジェクション向け

- ・オンザフライ復号(DOTF) (Compatibility Modeを使用する場合)

RX74M Wrap UFPK

RX74N Wrap UFPK

RX261 Wrap UFPK

図4-2 RSIP Protected Mode の RX74M グループの選択

SKMT と Key Wrap サービスでは、同じセキュリティモードを選択する必要があります。異なるモードで生成した UFPK と W-UFPK の組み合わせは使用できません。

その他の操作については、R01AN8067 に記載された手順に従ってください。

以降の説明では、次のファイルが C:\RX74M_FSBL\ufpk フォルダに保存されているものとします。

- UFPK : rx74m_ufpk.key
- W-UFPK : rx74m_ufpk.key_enc.key

4.2.2節および4.2.3節では、これらのファイルを使用して OEM_ROOT_PK をラッピングします。

4.2.2 SKMT CLI を使用したラッピングされた OEM_ROOT_PK の生成

SKMT がインストールされたら、コマンドラインウィンドウを開き、\cli フォルダに移動し、以下のコマンドを使用してラッピングされた OEM_ROOT_PK を生成します。この鍵は、RFP を使用したセキュア鍵注入プロセスを使用して MCU に注入されます。

```
C:\Renesas\SecurityKeyManagementTool\cli>skmt.exe /genkey /ufpk
file="C:\RX74M_FSBL\ufpk\rx74m_ufpk.key" /wufpk
file="C:\RX74M_FSBL\ufpk\rx74m_ufpk.key_enc.key" /mcu "RX-RSIP-E50D" /keytype
"OEM_ROOT_PK" /filetype "rfp" /iv "55AA55AA55AA55AA55AA55AA55AA55AA55AA" /key
file="C:\RX74M_FSBL\openssl_keys\oem_root_private_key.pem" /output
"C:\RX74M_FSBL\openssl_keys\oem_root_pk_cli.rkey"

Output File: C:\RX74M_FSBL\openssl_keys\oem_root_pk_cli.rkey

UFPK: 000102030405060708090A0B0C0D0E0F000102030405060708090A0B0C0D0E0F
W-UFPK: 00000000F77A67AB713D174EB9D7C6639DAADB1F77A67AB713D174EB9D7C6639DAADB1
IV: 55AA55AA55AA55AA55AA55AA55AA55AA55AA

Encrypted key:
F714A8B68CB80637F9EE1F35A310EBC262DE6D4311E1E56B762ECD4BA709BD94827A9B7906628B71
1E3CC8CCBA68BA8CE8B3392DEF2591574D647219F8F0DC11599EA9637BC7A94A53B260B5C248AB2C
```

ここで例示されている oem_root_pk_cli.rkey は、本アプリケーションノートのサンプルプロジェクトとは一致しないので、サンプルプロジェクトでは使用しないでください。

RXファミリ

RX74M ファーストステージブートローダを使ったアプリケーションの開発

4.2.3 SKMT GUI を使用したラッピングされた OEM_ROOT_PK の生成

ここでは、4.1.2節の OEM_BL 検証ルート鍵ペアを使用して、ラッピングされた OEM_ROOT_PK を生成します。

SKMT を起動し、概要画面で **RX Family, RSIP-E50D Security Functions and Protected Mode** を選択します(図4-1を参照)。次に、SKMT の鍵のラッピングタブに移動し、鍵の種類で **OEM Root public** を選択し、ラッピング鍵 (UFPK、W-UFPK)を入力します(図4-3参照)。

The screenshot shows the 'Key Wrapping' tab in the SKMT GUI. The 'Key Type' section has 'OEM Root public' selected. The 'Wrapping Key' section has 'UFPK' selected, with file paths for UFPK and W-UFPK.

Key Type	Key Data
☐ DLM/AL	AL2_KEY
☐ KUK	
☒ OEM Root public	secp256r1, public
☐ ECC	
☐ HMAC	

Wrapping Key	File Path
☒ UFPK	C:\RX74M_FSB\ufpk\rx74m_ufpk.key
☐ W-UFPK	C:\RX74M_FSB\ufpk\rx74m_ufpk.key_enc.key
☐ KUK	

図4-3 OEM_ROOT_PK のラップ

鍵データタブで、OEM Root Public Key を入力します。

The screenshot shows the 'Key Data' tab in the SKMT GUI. The 'Plain Data' section has 'Qx' and 'Qy' fields with hexadecimal values.

Key Data	Value
☒ 平文データ	Qx: d0720dc691aa80096ba32fed1cb97c2b620690d06de0317b8618d5ce65eb728f
	Qy: 9681b517b1cda17d0d83d335d9c4a8a9a9b0b1b3c7106d8f3c72bc5093dc275f
☐ 乱数を使用 - 出力ファイル	

図4-4 OEM_ROOT_PK の指定

RXファミリ

RX74M ファーストステージブートローダを使ったアプリケーションの開発

次に、出力タイプとして **RFP** を選択し、出力先を指定して、ファイル名を付けてから、**ファイル** を生成するをクリックし、ラッピングされた **OEM_ROOT_PK** を生成します。

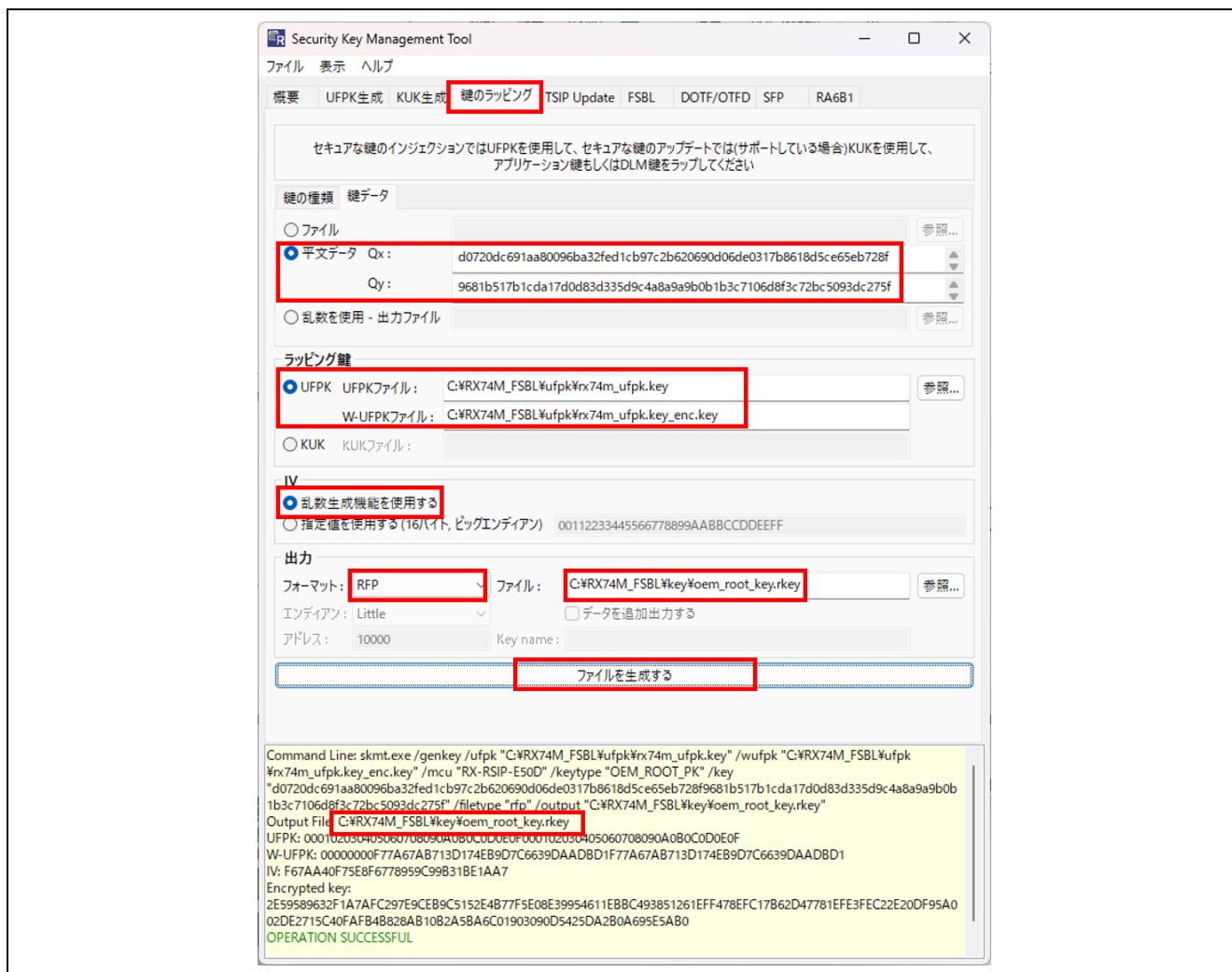


図4-5 ラッピングされた OEM_ROOT_PK の生成

このラッピングされた OEM_ROOT_PK (oem_root_key.rkey)は、システム評価を便利にするために提供されています。同じ公開鍵が鍵証明書およびコード証明書の生成に使用されている場合に使用できます。

4.3 RX74M FSBL の HMAC ブート設定

通常、FSBL を有効にしたブートを使用するためには、最初に FSBL を無効にした状態で OEM_BL を開発する必要があります。OEM_BL が完全にテストされてから、FSBL を有効にすることができます。

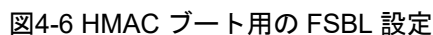
RX Flexible Software Package Documentation の「Create a New Project for Blinky」章に記載されているように、付属のサンプル Blinky プロジェクト `blinky_hmac_boot_rx74m_ccrx` は、RX74M 用の RTOS サポートなしで作成されています。デフォルトでは、Protect Zone の境界設定がデバッグ設定で有効になっています。

FSBL のプロパティは、図4-6の設定に基づいて構成されています。アプリケーションのデバッグを簡易にするために、ソフトウェアリセットやディープソフトウェアスタンバイモードからの復帰時に FSBL スキップが有効になっています。さらに、プロジェクトはエラーレポート用端子として P604 端子を使用しており初期状態で Low に設定していて、FSBL がブート失敗時に P604 端子を High にすることができます。e² studio のサンプルプロジェクトでは FSBL が有効になっていますが、e² studio でアプリケーションイメージがダウンロードされた後には FSBL は有効になりません。FSBL は次の電源投入後にのみ有効になります。

EK-RX74M のオンボード E2 エミュレータを使用するため、SW6-Pin1 および SW6-Pin2 を OFF に設定し、J8 と開発用 PC を USB Type-C®ケーブルで接続します。RDPM を使用する際は、SCI、FINE、SWD インタフェースのいずれかを選択できますが、この接続では FINE を使用します。

RX74M ファーストステージブートローダを使ったアプリケーションの開発

Blinky プロジェクト(`blinky_hmac_boot_rx74m_ccrx`)をワークスペースにインポートし、`configuration.xml` ファイルを開きます。次に、Generate Project Content をクリックします。



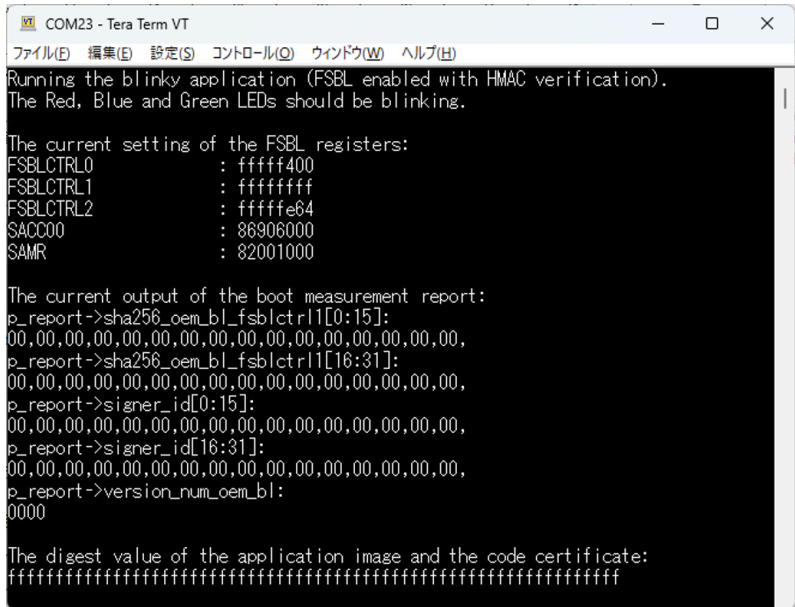
- コンパイル結果は、4.5節でコード証明書の作成に使用されます。
- 3つのLEDが点滅します。

図4-7 Tera Term の設定

RXファミリ

RX74M ファーストステージブートローダを使ったアプリケーションの開発

OK をクリックし、コンソールで以下の出力を確認します。FSBL のレジスタ設定、ブート測定レポート、HMAC ダイジェストの出力に注目してください。出力の内容については、RX74M グループ ユーザーズマニュアル ハードウェア編の「セキュアブート」章の内容(図2-4)を参照してください。FSBL はまだ有効になっていないため、version_num_oem_bl フィールドは 0、ダイジェスト領域は不定値になります。



```
COM23 - Tera Term VT
ファイル(F) 編集(E) 設定(S) コントロール(C) ウィンドウ(W) ヘルプ(H)
Running the blinky application (FSBL enabled with HMAC verification).
The Red, Blue and Green LEDs should be blinking.

The current setting of the FSBL registers:
FSBLCTRL0      : fffff400
FSBLCTRL1      : ffffffff
FSBLCTRL2      : fffffe64
SACC00         : 86906000
SAMR           : 82001000

The current output of the boot measurement report:
p_report->sha256_oem_bl_fsblctrl1[0:15]:
00,00,00,00,00,00,00,00,00,00,00,00,00,00,00,00,
p_report->sha256_oem_bl_fsblctrl1[16:31]:
00,00,00,00,00,00,00,00,00,00,00,00,00,00,00,00,
p_report->signer_id[0:15]:
00,00,00,00,00,00,00,00,00,00,00,00,00,00,00,00,
p_report->signer_id[16:31]:
00,00,00,00,00,00,00,00,00,00,00,00,00,00,00,00,
p_report->version_num_oem_bl:
0000

The digest value of the application image and the code certificate:
fffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffff
```

図4-8 HMAC ブートのコンソール出力

この時点では Root of Trust および証明書はまだ MCU へプログラムされていません。そのため、パワーオンリセット後に FSBL が実行されると OEM_BL の認証に失敗し、アプリケーションは起動しません。アプリケーションを正常に起動するためには、Root of Trust の注入と認証付き製品プログラミングを実施する必要があります。詳細は4.6節で説明します。

4.4 アンチロールバックカウンタ

RX74M は、古いバージョンの OEM_BL への書き戻しを防止するために、アンチロールバックカウンタを備えています。HMAC ブートを使用する場合、ブートファームウェアはイメージの書き込み時に、コード証明書のイメージバージョンとアンチロールバックカウンタを比較し、書き込むイメージがロールバックに該当していないことを確認します。

使用するアンチロールバックカウンタは、アンチロールバックカウンタ設定レジスタ(ARCCS)の ARC 利用モード選択ビット(ARCMD)で選択します。ARCMD の設定によって、HMAC ブートの検証対象を OEM_BL またはセキュアアプリケーションのいずれかに設定できます。

表 4-1 ARCMD の設定と使用するカウンタ

ARCMD の設定	モード	検証対象	使用するカウンタ	設定バージョン範囲
01b	モード 1	SSBL	ARCOEMBL	1~64
00b	モード 2	アプリケーション	ARCS	1~1024

7.2.33 アンチロールバックカウンタ設定レジスタ (ARCCS)

アドレス OFSM.ARCCS 86E3 0E30h

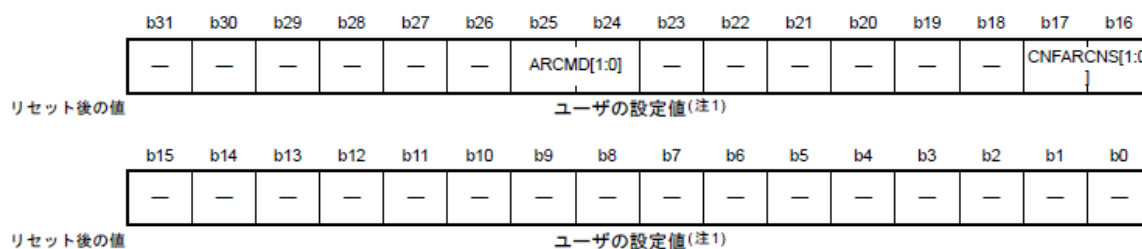


図4-9 アンチロールバックカウンタ設定レジスタ

ARCMD[1:0]=01b(モード 1)を設定した場合、HMAC ブートの検証対象は SSBL になります。このモードでは、コード証明書のイメージバージョンと OEM ブートローダ用アンチロールバックカウンタ (ARCOEMBL) の現在値を比較します。

コード証明書のイメージバージョンが ARCOEMBL の現在値以上である場合に限り、SSBL をプログラムできます。SSBL の検証に成功し、コード証明書のイメージバージョンが ARCOEMBL の現在値より大きい場合、ブートファームウェアは ARCOEMBL をそのイメージバージョンまで更新します。

7.2.37 OEM ブートローダ用アンチロールバックカウンタ [i] (ARCOEMBL[i]) (i = 0, 1)

アドレス OFSM.ARCOEMBL[0] 86F4 1F00h, OFSM.ARCOEMBL[1] 86F4 1F04h

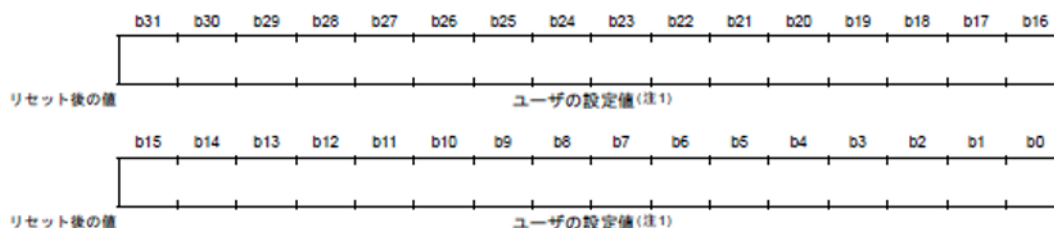


図4-10 OEM ブートローダ用アンチロールバックカウンタ

ARCMD[1:0]=00b(モード 2)を設定した場合、HMAC ブートの検証対象はアプリケーションになります。このモードでは、コード証明書のイメージバージョンと Secure アプリケーション用アンチロールバックカウンタ (ARCS) の現在値を比較します。

コード証明書のイメージバージョンが ARCS の現在値以上である場合に限り、セキュアアプリケーションをプログラムできます。セキュアアプリケーションの検証に成功し、コード証明書のイメージバージョンが ARCS の現在値より大きい場合、ブートファームウェアは ARCS をそのイメージバージョンまで更新します。

7.2.34 Secure アプリケーション用アンチロールバックカウンタ [i] (ARCS[i]) (i = 0 ~ 31)

アドレス OFSM.ARCS[0] 86F4 1D00h~OFSM.ARCS[31] 86F4 1D7Ch

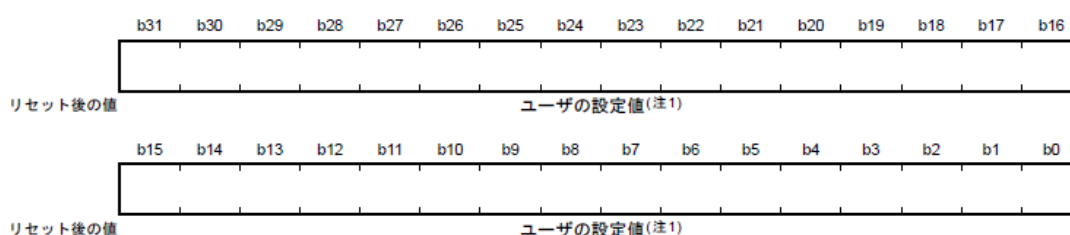


図4-11 Secure アプリケーション用アンチロールバックカウンタ

アンチロールバックカウンタは、カウンタの値を更新する際に、バージョンに対応する未使用のビットを 1 に設定することで、イメージバージョンを記録します。例えば、最初の 3 ビットが設定されている場合、レジスタ値は 0x00000007 となり、イメージバージョン 3 に対応します。

イメージを更新する場合は、アンチロールバックカウンタに記録されている現在のバージョンより大きい値を、コード証明書のイメージバージョンに設定してください。

HMAC ブートを使用する場合、ARCMD を設定する必要があります。用途に応じて、RFP の **Anti-Rollback Counter Mode** で **Mode1** または **Mode2** を選択してください。



図4-12 アンチロールバックカウンタ利用モードの設定

アンチロールバックカウンタの構成、値の読み方、および更新条件の詳細については、RX74M グループ ユーザーズマニュアル ハードウェア編の「アンチロールバックカウンタ」章を参照してください。

4.5 証明書の生成

前述のように、HMAC ブートでは、鍵とコード証明書を生成するには、2 セットの ECC P-256 secp256r1 の鍵ペアが必要です。本節では、2 つの方法を示しています。セキュリティの観点から、例示された鍵は製品サポートには使用しないでください。

- OpenSSL と SKMT コマンドラインインタフェースの使用
- NIST ベクタと SKMT GUI の使用

4.5.1 OpenSSL と SKMT コマンドラインインタフェースの使用

鍵証明書とコード証明書を生成するには、いくつかのプロパティを設定する必要があります。

- 入力:/loadaddr アプリケーションのロードアドレス。
- 入力:/oembl_size OEM_BL のサイズ。
- 入力: /ver アプリケーションのバージョン。以前に FSBL を介して古いバージョンのアプリケーションから起動した MCU の場合、次のバージョンはより高いバージョンである必要があります。バージョン番号は、生成されるコード証明書に含まれます(図2-4参照)。この例では 3 に設定されており、ARC_OEMBLn の値が 0、1、2 の場合に使用できます。
- 入力:/cfsize 使用するデバイスのコード MRAM サイズ。
- 入力:/oembl OEM_BL のバイナリファイル。
- 入力:/oemroot_private OEM_ROOT_SK。
- 入力:/oembl_private OEM_BL_SK。
- 出力:/output_codecert コード証明書。
- 出力:/output_keycert 鍵証明書。

コマンドウィンドウを開き、\SecurityKeyManagementTool\cli フォルダに移動し、以下のコマンドを実行して鍵証明書とコード証明書を生成します。ファイルの場所は必要に応じて変更してください。

```
C:\Renesas\SecurityKeyManagementTool\cli>skmt.exe /gencert /mode "signature" /loadaddr
"02000000" /oembl_size "8000" /cfsize "100000" /ver "3" /oembl
"C:\RX74M_FSBL\fsp_v510\blinky_hmac_boot_rx74m\Debug\blinky_hmac_boot_rx74m.srec
" /oembl_private file="C:\RX74M_FSBL\openssl_keys\oem_bl_private_key.pem"
/oemroot_private file="C:\RX74M_FSBL\openssl_keys\oem_root_private_key.pem"
/output_codecert "C:\RX74M_FSBL\openssl_keys\blinky_hmac_openssl_ccert.bin"
/output_keycert "C:\RX74M_FSBL\openssl_keys\blinky_hmac_openssl_kcert.bin"

Output File: C:\RX74M_FSBL\openssl_keys\blinky_hmac_openssl_kcert.bin
Output File: C:\RX74M_FSBL\openssl_keys\blinky_hmac_openssl_ccert.bin
```

4.5.2 NIST ECC の鍵ペアと SKMT GUI の使用

本項では、証明書の生成を説明するために、4.1.2節で提供された NIST CAVP ECC P-256(secp256r1)の 2 セットの鍵ペアを使用します。鍵証明書とコード証明書を生成するために、鍵ペアの秘密鍵の部分と公開鍵の部分の両方を SKMT に提供する必要があります。

SKMT GUI を起動します。概要画面で、**RX Family, RSIP-E50D Security Functions and Protected Mode** を選択します。

次に **FSBL** タブに移動し、**OEM Bootloader イメージ**の右側にある**参照**ボタンをクリックします。4.3節で生成された.srec ファイルを選択します。**プログラム検証方法**で **Signature** を選択し、**イメージバージョン**でアプリケーションのバージョンを定義します。

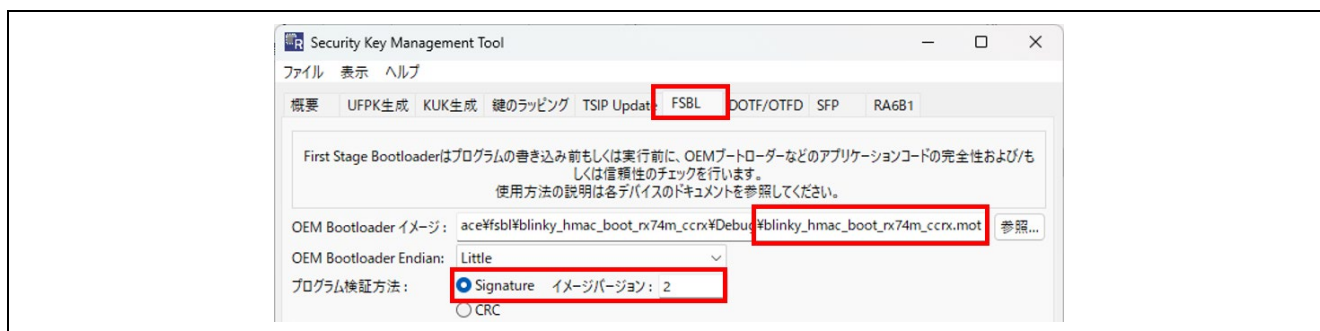


図4-13 アプリケーションとバージョン番号の選択

図4-13で定義されたイメージバージョンは、コード証明書に保存されます。2.2.3節で説明したように、許可されるバージョン番号は FSBL の検証対象によって異なります。FSBL の検証対象が SSBL の場合 1~64、アプリケーションの場合 1~1024 になります。更新イメージは現在プログラムされているイメージよりも新しいバージョンである必要があります。FSBL HMAC ベースのブートを初めて実行する場合、イメージバージョンは 1 に設定できます。

FSBL タブで **OEM_BL 検証ルート鍵**を設定します。以下の例では、いずれも**平文データ**を選択して鍵値を設定します。



図4-14 OEM_BL 検証ルート鍵の指定

次に、**OEM_BL 検証鍵**を設定します。以下の例では、いずれも**平文データ**を選択して鍵値を設定します。

図4-15 OEM_BL 検証鍵の指定

次に、**証明書**タブに移動し、**鍵証明書**と**コード証明書**で名前と場所を定義し、**ファイルを生成する**をクリックします。鍵証明書とコード証明書が生成されます。以下の例では、**OEM Bootloader サイズ**で**自動計算**を選択します。

図4-16 鍵証明書とコード証明書の生成

RXファミリ

RX74M ファーストステージブートローダを使ったアプリケーションの開発

SKMT は内部的に ECC 鍵ペアも生成できます。これは、**乱数を使用- 出力ファイル**を選択することで可能です。

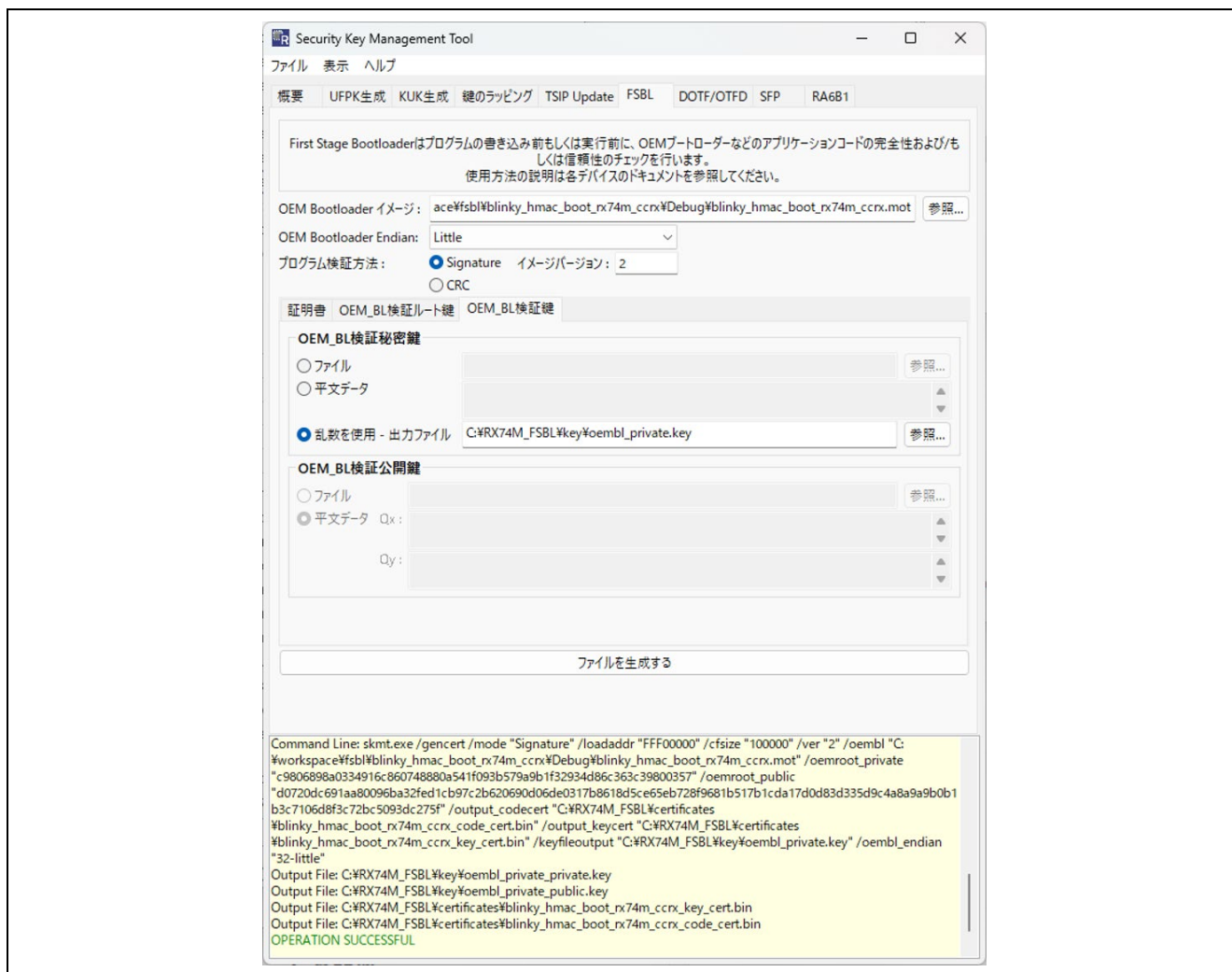


図4-17 SKMT で生成された ECC ベクトルの出力

図4-16、図4-17で生成された証明書は、本アプリケーションノートで説明されている他の操作に使用できます。

4.6 認証付き製品プログラミング

rfp_rx74m.zip をダウンロードし、その内容をローカルフォルダに解凍してください。次に、RFP を起動し、rfp_rx74m.rpj プロジェクトを開きます。このサンプル RFP プロジェクトでは、FINE インタフェースがブートインタフェースとして選択されていて、USB Type-C[®]ケーブルを使用して FINE インタフェースブートモードにアクセスします。

以前のアプリケーションで FSBL 設定になっている場合や、Protect Zone 境界が設定されている場合は、次の項目に進む前に初期化コマンドを実行する必要があります。これにより、前の FSBL 設定や Protect Zone 設定が削除されます。

次に、RFP を使用して MCU にアプリケーションをダウンロードし、アプリケーションを認証するための認証情報を提供します。フラッシュオプションのプログラミングを通じて、いくつかの認証情報が MCU に注入されます。以下の**操作設定**が構成されていることを確認し、以前のアプリケーションを消去し、新しいアプリケーションをダウンロードして、フラッシュオプションをプログラムします。

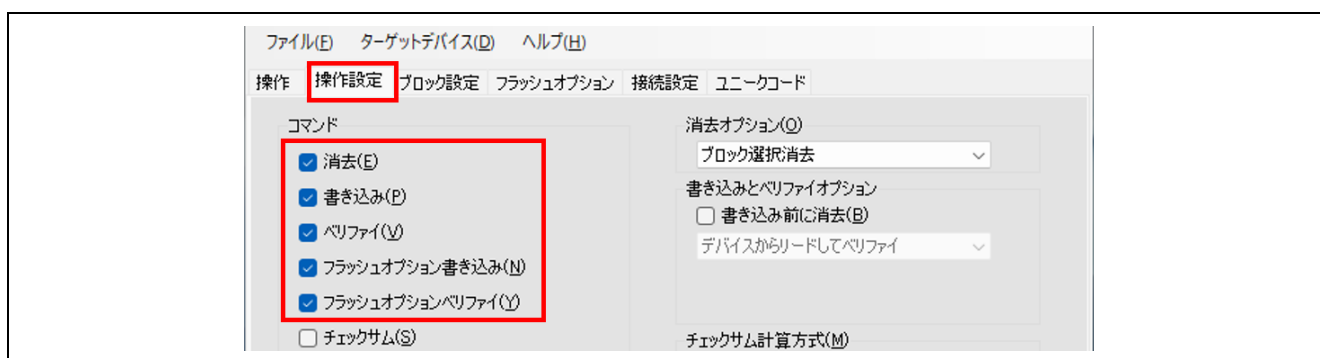


図4-18 動作設定の構成

RXファミリ

RX74M ファーストステージブートローダを使ったアプリケーションの開発

MCUにプログラムするアプリケーションファイルを選択します。操作タブで**ファイルの追加と削除**をクリックします。**ファイルを追加**をクリックして `blinky_hmac_boot_rx74m.mot` を選択します。加えて、ARCMD の設定に合わせて、`ARCCS_ARCMD_mode1.mot` または `ARCCS_ARCMD_mode2.mot` を選択します。**OK** をクリックします。

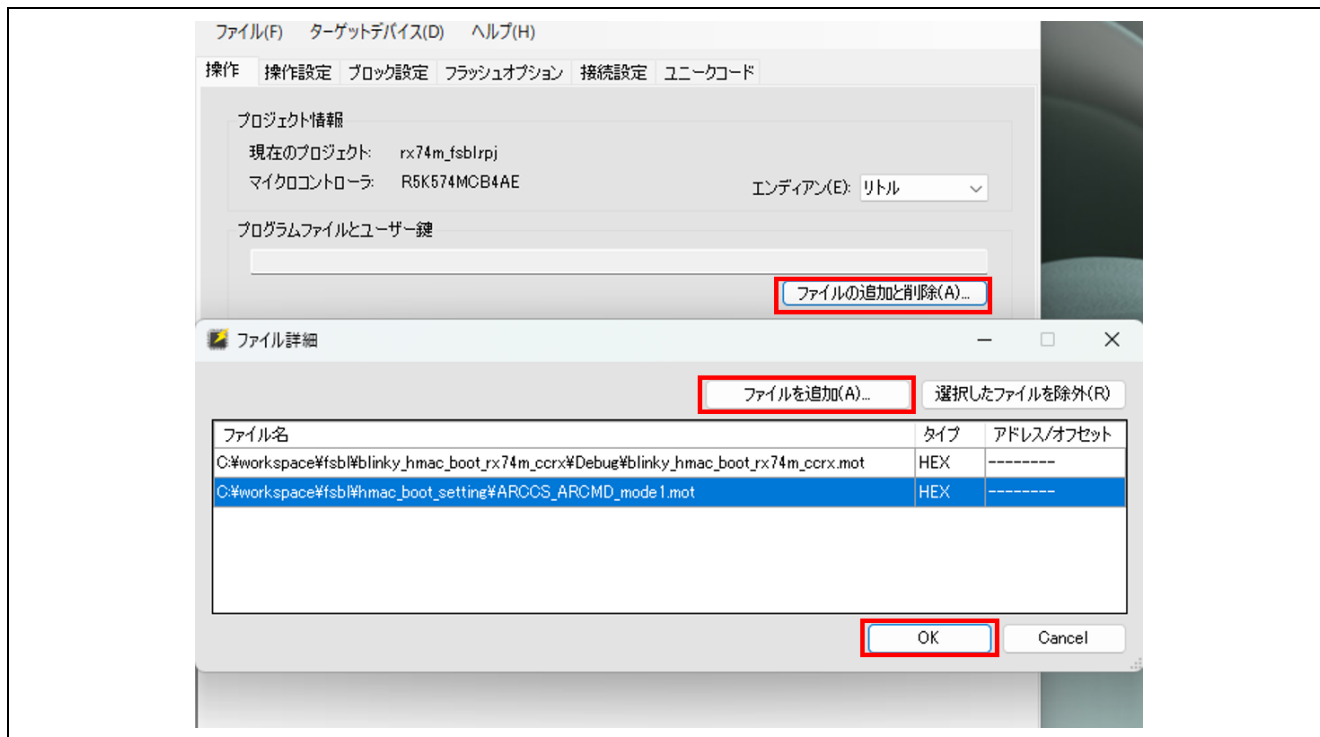


図4-19 アプリケーションプロジェクトを選択

フラッシュオプションタブで **OEM Root Public Key**、**Code Certificate**、**Key Certificate** を設定します。

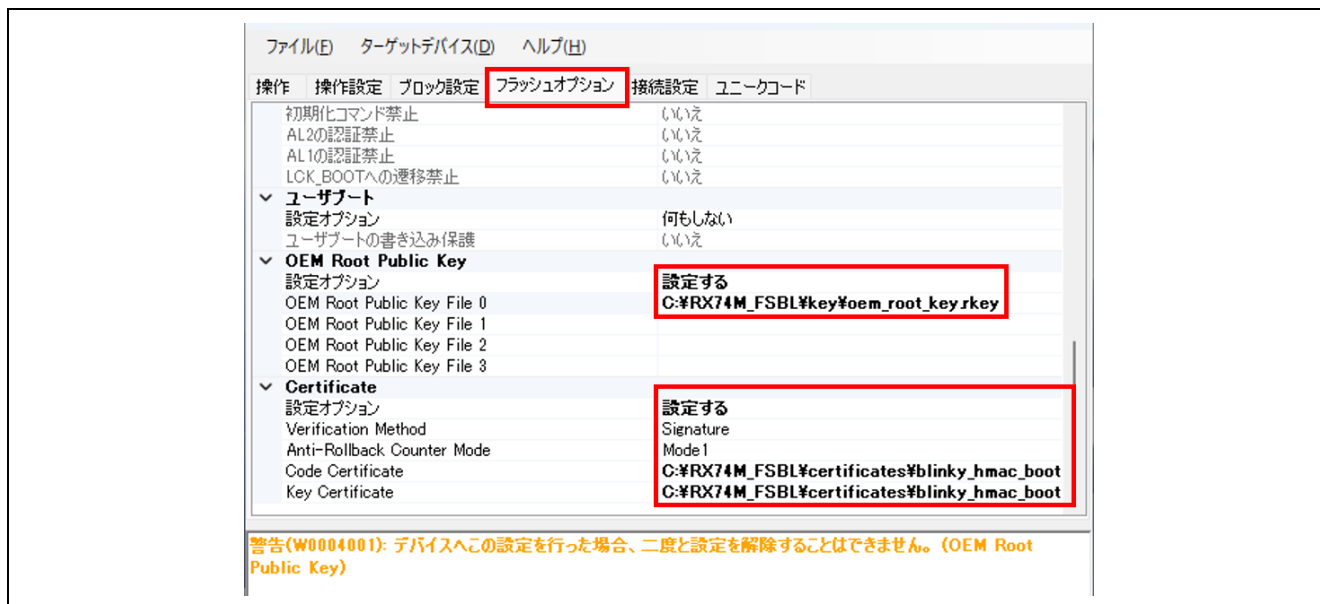


図4-20 OEM_ROOT_PK と証明書の設定

RX74M ファーストステージブートローダを使ったアプリケーションの開発

ここで**操作**タブに移動し、**スタート**ボタンをクリックします。RFP は.mot ファイルから SACC0 レジスタ値を読み取り、指示されたメモリアドレスにコード証明書をプログラムします。FSBL はコード証明書のバージョン番号を比較し、バージョン番号の方が新しければ OEM_BL の認証を進めます。認証が成功すると、OEM_BL ダイジェストがコード証明書の後の領域にプログラムされます。次の電源投入で、FSBL は保存されたダイジェストと FSBL が計算したダイジェストを使用してアプリケーションを認証し、比較が成功すると、新しいアプリケーションが起動し、3 つの LED が点滅し、正常動作を確認することができます。

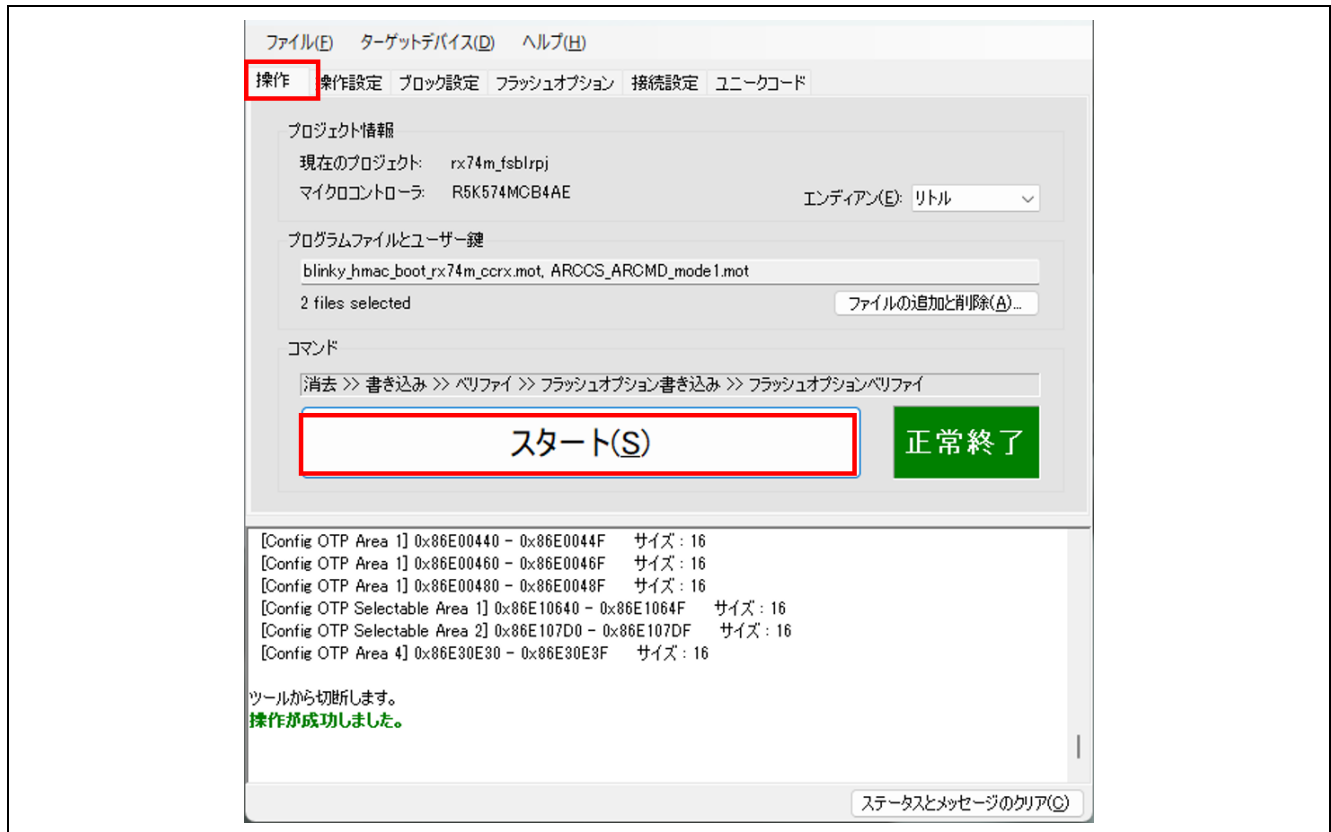
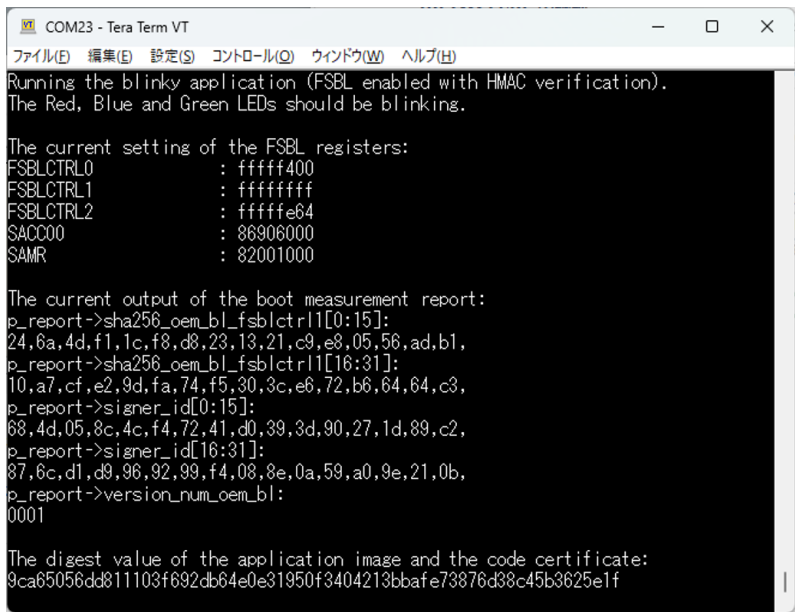


図4-21 HMAC SHA2-256 検証に基づくアプリケーションの更新成功

RXファミリ

RX74M ファーストステージブートローダを使ったアプリケーションの開発

起動に成功すると、Tera Term コンソールは以下のような情報を出力します。version_num_oem_bl とイメージダイジェストは、キットの状態とプログラムされたアプリケーションによって異なります。



```
COM23 - Tera Term VT
ファイル(F) 編集(E) 設定(S) コントロール(C) ウィンドウ(W) ヘルプ(H)
Running the blinky application (FSBL enabled with HMAC verification).
The Red, Blue and Green LEDs should be blinking.

The current setting of the FSBL registers:
FSBLCTRL0      : fffff400
FSBLCTRL1      : ffffffff
FSBLCTRL2      : fffffe64
SACCC00        : 86906000
SAMR           : 82001000

The current output of the boot measurement report:
p_report->sha256_oem_bl_fsblctrl1[0:15]:
24,6a,4d,f1,1c,f8,d8,23,13,21,c9,e8,05,56,ad,b1,
p_report->sha256_oem_bl_fsblctrl1[16:31]:
10,a7,cf,e2,9d,fa,74,f5,30,3c,e6,72,b6,64,64,c3,
p_report->signer_id[0:15]:
68,4d,05,8c,4c,f4,72,41,d0,39,3d,90,27,1d,89,c2,
p_report->signer_id[16:31]:
87,6c,d1,d9,96,92,99,f4,08,8e,0a,59,a0,9e,21,0b,
p_report->version_num_oem_bl:
0001

The digest value of the application image and the code certificate:
9ca65056dd811103f692db64e0e31950f3404213bbafe73876d38c45b3625e1f
```

図4-22 証明書を使用した HMAC ブートのコンソール出力

HMAC ブートの失敗は容易に把握できます。まずデバイスを初期化し、4.6節の手順を実行します。ただし、アプリケーションイメージが不正な場合、例えば OEM_BL の一部を書き換えた場合には、図4-21の RFP プログラミングは失敗し、エラーレポートとして P604 端子が” High” になり、赤色の LED が点灯します。

5. 付録

5.1 ルネサスセキュアファクトリプログラミングでの FSBL 使用

RX74M は、セキュアファクトリプログラミングにより、暗号化されたアプリケーションプログラムイメージのプログラミングをサポートし、非セキュアな環境でもセキュアなアプリケーションプログラムのプログラミングを可能にします。

ルネサスセキュアファクトリプログラミングは、FSBL の有無にかかわらずアプリケーションプログラムを実装できます。FSBL をセキュアファクトリプログラミングで使用する場合、アプリケーションは FSBL を設定する必要があります。暗号化されたイメージが MCU にプログラムされた後、リセットを実行する前に FSBL を起動します。詳細については、RX74M グループ ユーザーズマニュアル ハードウェア編の「セキュアファクトリプログラミング」章を参照してください。

5.2 FSBL のレジスタ設定

FSBL を設定するオプションは、スマートコンフィギュレータの **BSP** タブから選択できます。以下の FSBL の設定では、**プロパティ** タブの **FSBLEXMDFSBLEN** で **HMAC boot with report measurement** を選択した例を示しています。開発するアプリケーションのセキュリティの目的に応じて、これらの設定を調整してください。各種スタックプロパティの詳細については、ルネサスの GitHub ページ(<https://github.com/renesas>)の RX Flexible Software Package Documentation > API Reference > BSP > MCU Board Support Package の Build Time Configurations を参照してください。

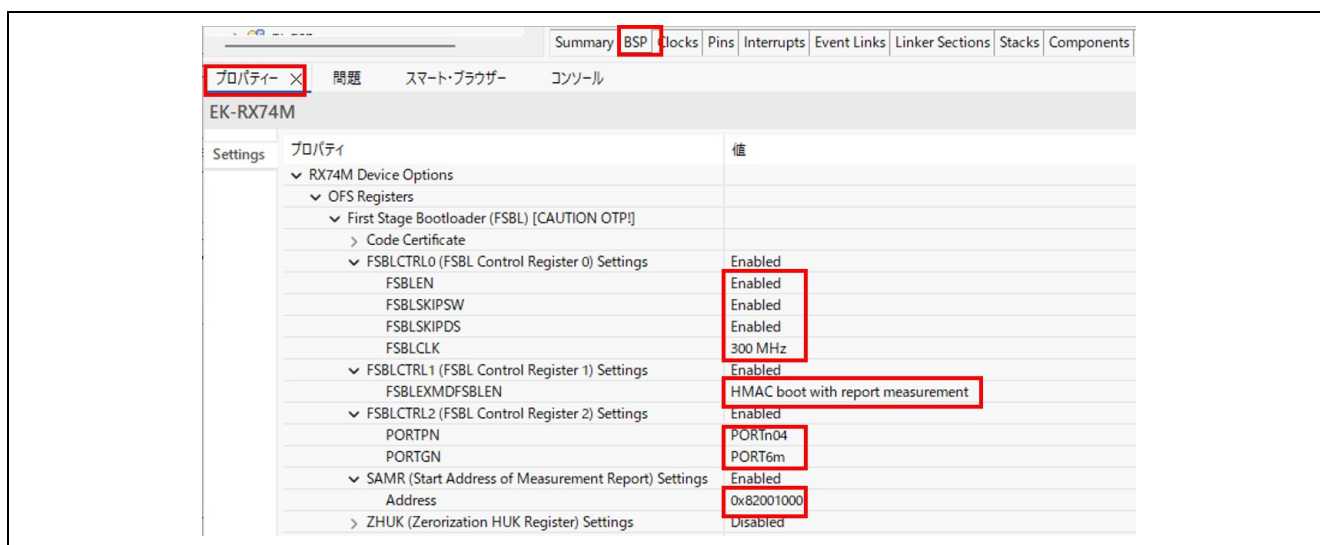


図5-1 FSBL の設定

RX74M を対象とするアプリケーションでは、FSBL を必ず使用する必要はなく、**FSBLEN** プロパティを **Disabled** に設定することで FSBL を未使用にできます。この場合、プログラミング中または実行前のアプリケーションの検証は行われません。FSBL を未使用に設定している MCU は、FSBL を実装していない他の RX MCU と同じように使用できます。

FSBL の検証の実行には時間がかかるため、特定のタイプのリセット後に検証プロセスをスキップしても問題ない場合があります。ソフトウェアリセット時に FSBL をスキップするには、**FSBLSKIPSW** プロパティを **Enabled** に設定します。同様に、ディープソフトウェアスタンバイリセットからのリセット時に FSBL をスキップするには、**FSBLSKIPDS** プロパティを **Enabled** に設定します。

CRC ブートと HMAC ブートの両方で、図5-2の FSBL を使用した検証オプションに示されているように、**report measurement** オプションの有無を選択することができます。

▼ FSBLCTRL1 (FSBL Control Register 1) Settings	Enabled
FSBLEXMDFSBLEN	HMAC boot with report measurement
▼ FSBLCTRL2 (FSBL Control Register 2) Settings	CRC boot without report
PORTPN	CRC boot with report measurement
PORTGN	HMAC boot without report
SAMR (Control Register 3) Settings	HMAC boot with report measurement

図5-2 FSBL を使用した検証オプション

report measurement を選択した場合、図5-3の情報が SAMR レジスタに格納されているメモリアドレスに出力されます。BSP タブの **FSBL Measurement Report Address** プロパティを使用して設定されます(図5-1を参照)。測定レポートの内容は、[Arm Platform Security Architecture Attestation API](#)の要件を満たすように生成されます。使用方法については、このリンクの最新バージョンの Arm PSA Certified Attestation API マニュアルを参照してください。

SAMRレジスタで指定したSRAMアドレス + 00h	OEM_BLと{000000b + FSBLCTRL1[1:0]} のSHA2-256ハッシュ値
SAMRレジスタで指定したSRAMアドレス + 20h	署名者ID (OEM_BL_PKの SHA2-256ハッシュ値)
SAMRレジスタで指定したSRAMアドレス + 40h	OEM_BLのバージョン番号
SAMRレジスタで指定したSRAMアドレス + 43h	

注: OEM_BLとFSBLCTRL[7:0]はこの順番でハッシュ関数に入力されます。

図5-3 測定レポートの内容と場所

CRC ブートの場合、署名者 ID には計算された CRC を 8 回繰り返した値が出力されます。各 CRC32 は 4 バイトで、署名者 ID は 32 バイトです。

ブートに失敗すると、**FSBLCTRL2** レジスタで設定された端子に “High” 出力されます。

FSBL のレジスタは**エクストラ MRAM オプション設定メモリ領域**にマッピングされています。FSBL のレジスタのプログラミングは、シリアルプログラミング(JTAG/SWD または FINE)を使用して IDE(e² studio など)で可能です。RFP を使用してアプリケーションをプログラムする場合、プログラミングインタフェースを介して行われます。これにより、FSBL レジスタで設定された内容によって、リセット後に FSBL が起動するか、OEM_BL に直接ジャンプするかが決まります。

6. References

1. RX ファミリ セキュアユーザキーのインジェクションとアップデート(R01AN8067)
2. RX74M グループ ユーザーズマニュアル ハードウェア編(R01UH1055)
3. Renesas Security Key Management Tool ユーザーズマニュアル(R20UT5349)
4. Renesas Flash Programmer フラッシュ書き込みソフトウェア ユーザーズマニュアル(R20UT5820)
5. RX Flexible Software Package Documentation Release v1.0.0

7. ウェブサイトとサポート

RX ファミリのマイクロコントローラの詳細、ツール、ドキュメントのダウンロード、サポートについては、以下の URL をご覧ください。

EK-RX74M リソース	renesas.com/rx/ek-rx74m
RX 製品情報	renesas.com/rx
Flexible Software Package (FSP)	renesas.com/rx-fsp
RX 製品サポートフォーラム	renesas.com/rx/forum
ルネサスサポート	renesas.com/support

改訂記録

Rev.	発行日	改訂内容	
		ページ	ポイント
1.00	2026.10.22	-	初版発行

製品ご使用上の注意事項

ここでは、マイコン製品全体に適用する「使用上の注意事項」について説明します。個別の使用上の注意事項については、本ドキュメントおよびテクニカルアップデートを参照してください。

1. 静電気対策

CMOS 製品の取り扱いの際は静電気防止を心がけてください。CMOS 製品は強い静電気によってゲート絶縁破壊を生じることがあります。運搬や保存の際には、当社が出荷梱包に使用している導電性のトレーやマガジンケース、導電性の緩衝材、金属ケースなどを利用し、組み立て工程にはアースを施してください。プラスチック板上に放置したり、端子を触ったりしないでください。また、CMOS 製品を実装したボードについても同様の扱いをしてください。

2. 電源投入時の処置

電源投入時は、製品の状態は不定です。電源投入時には、LSI の内部回路の状態は不確定であり、レジスタの設定や各端子の状態は不定です。外部リセット端子でリセットする製品の場合、電源投入からリセットが有効になるまでの期間、端子の状態は保証できません。同様に、内蔵パワーオンリセット機能を使用してリセットする製品の場合、電源投入からリセットのかかる一定電圧に達するまでの期間、端子の状態は保証できません。

3. 電源オフ時における入力信号

当該製品の電源がオフ状態のときに、入力信号や入出力ブルアップ電源を入れないでください。入力信号や入出力ブルアップ電源からの電流注入により、誤動作を引き起こしたり、異常電流が流れ内部素子を劣化させたりする場合があります。資料中に「電源オフ時における入力信号」についての記載のある製品は、その内容を守ってください。

4. 未使用端子の処理

未使用端子は、「未使用端子の処理」に従って処理してください。CMOS 製品の入力端子のインピーダンスは、一般に、ハイインピーダンスとなっています。未使用端子を開放状態で動作させると、誘導現象により、LSI 周辺のノイズが印加され、LSI 内部で貫通電流が流れたり、入力信号と認識されて誤動作を起こす恐れがあります。

5. クロックについて

リセット時は、クロックが安定した後、リセットを解除してください。プログラム実行中のクロック切り替え時は、切り替え先クロックが安定した後、に切り替えてください。リセット時、外部発振子（または外部発振回路）を用いたクロックで動作を開始するシステムでは、クロックが十分安定した後、リセットを解除してください。また、プログラムの途中で外部発振子（または外部発振回路）を用いたクロックに切り替える場合は、切り替え先のクロックが十分安定してから切り替えてください。

6. 入力端子の印加波形

入力ノイズや反射波による波形歪みは誤動作の原因になりますので注意してください。CMOS 製品の入力がノイズなどに起因して、 V_{IL} (Max.) から V_{IH} (Min.) までの領域にとどまるような場合は、誤動作を引き起こす恐れがあります。入力レベルが固定の場合はもちろん、 V_{IL} (Max.) から V_{IH} (Min.) までの領域を通過する遷移期間中にチャタリングノイズなどが入らないように使用してください。

7. リザーブアドレス（予約領域）のアクセス禁止

リザーブアドレス（予約領域）のアクセスを禁止します。アドレス領域には、将来の拡張機能用に割り付けられている リザーブアドレス（予約領域）があります。これらのアドレスをアクセスしたときの動作については、保証できませんので、アクセスしないようにしてください。

8. 製品間の相違について

型名の異なる製品に変更する場合は、製品型名ごとにシステム評価試験を実施してください。同じグループのマイコンでも型名が違うと、フラッシュメモリ、レイアウトパターンの相違などにより、電気的特性の範囲で、特性値、動作マージン、ノイズ耐量、ノイズ幅射量などが異なる場合があります。型名が違う製品に変更する場合は、個々の製品ごとにシステム評価試験を実施してください。

ご注意書き

1. 本資料に記載された回路、ソフトウェアおよびこれらに関連する情報は、半導体製品の動作例、応用例を説明するものです。回路、ソフトウェアおよびこれらに関連する情報を使用する場合、お客様の責任において、お客様の機器・システムを設計ください。これらの使用に起因して生じた損害（お客様または第三者いずれに生じた損害も含みます。以下同じです。）に関し、当社は、一切その責任を負いません。
2. 当社製品または本資料に記載された製品データ、図、表、プログラム、アルゴリズム、応用回路例等の情報の使用に起因して発生した第三者の特許権、著作権その他の知的財産権に対する侵害またはこれらに関する紛争について、当社は、何らの保証を行うものではなく、また責任を負うものではありません。
3. 当社は、本資料に基づき当社または第三者の特許権、著作権その他の知的財産権を何ら許諾するものではありません。
4. 当社製品を組み込んだ製品の輸出入、製造、販売、利用、配布その他の行為を行うにあたり、第三者保有の技術の利用に関するライセンスが必要となる場合、当該ライセンス取得の判断および取得はお客様の責任において行ってください。
5. 当社製品を、全部または一部を問わず、改造、改変、複製、リバースエンジニアリング、その他、不適切に使用しないでください。かかる改造、改変、複製、リバースエンジニアリング等により生じた損害に関し、当社は、一切その責任を負いません。
6. 当社は、当社製品の品質水準を「標準水準」および「高品質水準」に分類しており、各品質水準は、以下に示す用途に製品が使用されることを意図しております。

標準水準： コンピュータ、OA 機器、通信機器、計測機器、AV 機器、家電、工作機械、パーソナル機器、産業用ロボット等

高品質水準： 輸送機器（自動車、電車、船舶等）、交通制御（信号）、大規模通信機器、金融端末基幹システム、各種安全制御装置等

当社製品は、データシート等により高信頼性、Harsh environment 向け製品と定義しているものを除き、直接生命・身体に危害を及ぼす可能性のある機器・システム（生命維持装置、人体に埋め込み使用するもの等）、もしくは多大な物的損害を発生させるおそれのある機器・システム（宇宙機器と、海底中継器、原子力制御システム、航空機制御システム、プラント基幹システム、軍事機器等）に使用されることを意図しておらず、これらの用途に使用することは想定していません。たとえ、当社が想定していない用途に当社製品を使用したことにより損害が生じて、当社は一切その責任を負いません。

7. あらゆる半導体製品は、外部攻撃からの安全性を 100%保証されているわけではありません。当社ハードウェア／ソフトウェア製品にはセキュリティ対策が組み込まれているものもありますが、これによって、当社は、セキュリティ脆弱性または侵害（当社製品または当社製品が使用されているシステムに対する不正アクセス・不正使用を含みますが、これに限られません。）から生じる責任を負うものではありません。当社は、当社製品または当社製品が使用されたあらゆるシステムが、不正な改変、攻撃、ウイルス、干渉、ハッキング、データの破壊または窃盗その他の不正な侵入行為（「脆弱性問題」といいます。）によって影響を受けないことを保証しません。当社は、脆弱性問題に起因またはこれに関連して生じた損害について、一切責任を負いません。また、法令において認められる限りにおいて、本資料および当社ハードウェア／ソフトウェア製品について、商品性および特定目的との合致に関する保証ならびに第三者の権利を侵害しないことの保証を含め、明示または黙示のいかなる保証も行いません。
8. 当社製品をご使用の際は、最新の製品情報（データシート、ユーザーズマニュアル、アプリケーションノート、信頼性ハンドブックに記載の「半導体デバイスの使用上の一般的な注意事項」等）をご確認の上、当社が指定する最大定格、動作電源電圧範囲、放熱特性、実装条件その他指定条件の範囲内でご使用ください。指定条件の範囲を超えて当社製品をご使用された場合の故障、誤動作の不具合および事故につきましては、当社は、一切その責任を負いません。
9. 当社は、当社製品の品質および信頼性の向上に努めていますが、半導体製品はある確率で故障が発生したり、使用条件によっては誤動作したりする場合があります。また、当社製品は、データシート等において高信頼性、Harsh environment 向け製品と定義しているものを除き、耐放射線設計を行っておりません。仮に当社製品の故障または誤動作が生じた場合であっても、人身事故、火災事故その他社会的損害等を生じさせないよう、お客様の責任において、冗長設計、延焼対策設計、誤動作防止設計等の安全設計およびエージング処理等、お客様の機器・システムとしての出荷保証を行ってください。特に、マイコンソフトウェアは、単独での検証は困難なため、お客様の機器・システムとしての安全検証お客様の責任で行ってください。
10. 当社製品の環境適合性等の詳細につきましては、製品個別に必ず当社営業窓口までお問合せください。ご使用に際しては、特定の物質の含有・使用を規制する RoHS 指令等、適用される環境関連法令を十分調査のうえ、かかる法令に適合するようご使用ください。かかる法令を遵守しないことにより生じた損害に関して、当社は、一切その責任を負いません。
11. 当社製品および技術を国内外の法令および規則により製造・使用・販売を禁止されている機器・システムに使用することはできません。当社製品および技術を輸出、販売または移転等する場合は、「外国為替及び外国貿易法」その他日本国および適用される外国の輸出管理関連法規を遵守し、それらの定めるところに従い必要な手続きを行ってください。
12. お客様が当社製品を第三者に転売等される場合には、事前に当該第三者に対して、本ご注意書き記載の諸条件を通知する責任を負うものいたします。
13. 本資料の全部または一部を当社の文書による事前の承諾を得ることなく転載または複製することを禁じます。
14. 本資料に記載されている内容または当社製品についてご不明な点がございましたら、当社の営業担当者までお問合せください。

注 1. 本資料において使用されている「当社」とは、ルネサス エレクトロニクス株式会社およびルネサス エレクトロニクス株式会社が直接的、間接的に支配する会社をいいます。

注 2. 本資料において使用されている「当社製品」とは、注 1 において定義された当社の開発、製造製品をいいます。

(Rev.5.0-1 2020.10)

本社所在地

〒135-0061 東京都江東区豊洲 3-2-24（豊洲フォレシア）

www.renesas.com

お問合せ窓口

弊社の製品や技術、ドキュメントの最新情報、最寄の営業お問合せ窓口に関する情報などは、弊社ウェブサイトをご覧ください。

www.renesas.com/contact/

商標について

ルネサスおよびルネサスロゴはルネサス エレクトロニクス株式会社の商標です。すべての商標および登録商標は、それぞれの所有者に帰属します。