

RA8 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

はじめに

以下に示す Renesas RA8 シリーズ MCU は、「First Stage Bootloader(FSBL)」を ROM にプログラムして おり、マスクされています。リセット後に実行され、チップ内のフラッシュメモリにプログラムされた OEM アプリケーションプログラムをシングルチップモードで検証できます。フラッシュメモリの不揮発性 により、FSBL はシリコンベースで Root of Trust(RoT)を提供します。

FSBLを使用するには、OEM アプリケーションプログラムが製造時のプログラミング中に検証される必要があります。MCU のファクトリブートアプリケーションプログラムがこの機能を提供します。

このアプリケーションプロジェクトは、ファクトリブートアプリケーションプログラムおよび FSBL が OEM アプリケーションプログラムを検証するために使用する資格情報の設定方法を説明します。その後、 アプリケーションプロジェクトは、Cyclic Redundancy Check 32(CRC32)整合性検証または Elliptic Curve Cryptography(ECC)NIST P256 署名認証と HMAC-SHA256 認証のいずれかの技術を使用してアプリケーシ ョンプログラムの検証を実演します。

必要なリソース

対象デバイス

以下は、本資料の情報が適用されるルネサス MCU 製品です。:

- RA8M1
- RA8D1
- RA8T1

ソフトウェアと開発ツール

- e² studio IDE v2023-10
- ルネサスフレキシブルソフトウェアパッケージ(FSP)v5.1.0
 ダウンロードはこちらから <u>https://github.com/renesas/fsp</u>.
- •
- SEGGER J-Link[®] USB driver v7.92o 以降(<u>SEGGER J-Link</u>)
- Renesas Secure Key Management Tool v1.05 以降
- Renesas Flash Programmer v3.13 以降
- GNU Privacy Guard for Windows: <u>Gpg4win</u>
- Renesas Device Lifecycle Management Server
- •

ハードウェア

- EK-RA8M1、RA8M1 MCU グループ用評価キット(renesas.com/ra/ek-ra8m1)
- Windows[®] 10 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 デバイスケーブル(タイプ A オス-マイクロ B オス)1 本



RA8 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

前提条件と対象読者

このアプリケーションプロジェクトをお使いいただくには、ルネサス e² studio の使用経験があることを前 提としています。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。このアプリケーションプロジェクトを実行する前に、ユーザは[ソフトウェアと開発 ツール]に記載されているすべてのツールをインストールし、RA8M1 ユーザーズマニュアル ハードウェア 編の以下セクションをお読みいただく必要があります。これにより、このアプリケーションプロジェクトの いくつかの説明の背景にある情報が得られ、ユーザがより深く掘り下げたり、学習を拡張するための便利さ も提供されます。

- セクション:セキュリティ機能
- セクション:オプション設定メモリ
- セクション:フラッシュメモリ

さらに、本デモンストレーションには非 TrustZone[®]アプリケーションが使用されています。もし、FSBL を TrustZone[®]プロジェクトで使用したい場合は、アプリケーションノート Injecting and Updating Secure User Keys (R11AN0496)のセクション 4.1 を参照してください。



RA8 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

目次

1.	ファーストステージブートローダの紹介	5
1.1	概要	5
1.2	FSBL ベースの起動オプション	8
1.2.1	製品へのプログラミング中の OEM_BL 検証の概要	10
1.2.2	シングルチップ動作時の OEM_BL 検証	12
1.2.3	シングルチップモードで HMAC-SHA256 を使用する利点	13
1.3	認証チェックに使用される認証情報の詳細	13
1.3.1	Root of Trust	14
1.3.2	キー証明書	14
1.3.3	コード証明書	15
1.3.4	製品へのプログラミングとセキュアブートにおける認証情報の概要	17
1.4	整合性チェックで使用される認証情報の詳細	18
1.4.1	コード証明書	18
1.4.2	製品へのプログラミング、CRC ブートにおける認証情報の概要	19
1.5	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	20
2.	認証された製品プログラミングと HMAC-SHA256 ブートデモンストレーション	21
2.1	ECC secp256r1 キーペア 2 セットの準備	21
2.1.1	OpenSSL を使用した ECC キーペアの生成	21
2.1.2	・ ECC キーペアとした NIST CAVP テストベクタのダウンロード	
2.2	Root of Trust の導入	
2.2.1	Wrapped User Factory Programming Key の生成	
2.2.2	OpenSSL と SKMT CLI インタフェースを使用した Wrapped OEM Root Public Key の生成…	
2.2.3	。 SKMT GUI インタフェースを使用した Wrapped OEM Root Public Key の生成	
2.3	HMAC-SHA256 ブートを有効にした Blinky アプリケーションの準備	
2.4	MCU OEM_BL アンチロールバックカウンタ値の取得	
2.5	鍵証明書とコード証明書の生成	
2.5.1	OpenSSL と SKMT コマンドラインインターフェイスの使用	
2.5.2	NIST ECC のキーペアと SKMT GUI インタフェースの使用の使用	
2.6	認証付き製品プログラミングと HMAC ブートのデモンストレーション	
2	CPC チェックト CPC ゴートデエンフトレーション制日々 プログラミング	10
ວ. ວ.4		43
3.1	CRC ノート FSBL を使用した DIINKy サノノルノロジェクト準備	
ა.∠ ეე	コート証明者のTFR	
J.J	RFF を使用したアノリクーションの GRG ノートの夫証	
4.	付録	50
4.1	TrustZone [®] プロジェクトでの使用上の注意点	50
4.2	RFP 使用時のエラーのデバッグ	53



RA8 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

5.	References	54
6.	ウェブサイトとサポート	55
改訂		56



RA8 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

1. ファーストステージブートローダの紹介

1.1 概要

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、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なわれ、簡単 な整合性チェックや、信頼性を保証するための署名検証を含む場合があります。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。

FSBL が有効になっている場合、ユーザの製品アプリケーションプログラムを検証する2つのオプションがあります。

- CRC32 を使用した整合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。
- 信頼性のチェックをします。MCUへのプログラミングの初期時、アプリケーションは secp256r1(NIST P256)ECC 曲線を使用して ECDSA で認証されます。また、実行前には MCU のハードウェアユニーク キー(以下 HUK)で HMAC-SHA256 を使用して認証されます。

セキュア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。RA8 MCU には不変の FSBL が書き込まれていて、セキュア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 のブートローダを検証します。その後 OEM 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 Chain of Trust(認証パスについては図 1 case 2 を参照)を形成します。したがって、FSBL を使用すると、OEM ブートローダは安全に更新できま す。



以下は、MCU シングルチップモードでの FSBL を使用した概要を示したものです。

図 1. シングルチップモードでの FSBL の使用方法

RA8 MCU ユーザーズマニュアル ハードウェア編では、セカンドステージブートローダーが存在する case2 に関する説明に焦点を当てています。また、FSBL(ファーストステージブートローダ)に関連する操作につい てファーストステージ、セカンダリステージ両方で「OEM_BL」と呼んでいますが、アプリケーションプロ ジェクトのリリースでは、最初のケース(FSBL が存在しない場合)のみが対象です。ユーザーズマニュアル ハードウェア編と一致させるため、「OEM_BL」は OEM 製品のアプリケーションプログラムを指す用語と して使用します。

以下の図のように、FSBL のレジスタは**データフラッシュオプション設定メモリ領域**にマッピングされま す。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 または SWD)を使用して IDE(e² studio など)で可能です。RFP(ルネサスフラッシュプログラマ)を使用してアプリケーションをプログラムする場 合、ブートアクセスモードを介して行われます。これにより、FBSL レジスタで Setting された内容によっ て、リセット後の MCU の動作状態と、OEM_BL に直接ジャンプするかが決まります。





図 2. データフラッシュオプション設定メモリ領域 RA8x1 の場合



「ユーザーロック可能領域は、製造時やその他の製造プログラム中に Renesas Flash Programming ツール を使用してロックすることができます。

但し、誤ってこれらの領域をロックしたら、ロックを解除することはできません。このため、アプリケーション開発中にこれらのレジスタをロックすることは推奨しません。

File Target Device Help			
Operation Operation Settings Block Settings	Hash Options Connect Settings Ur	nique Code	
Disable Initialize Command	No	~	
Disable AL2 authentication	No		
Disable Al 1 a theotication	No		
Disable LCK_BOOT transition	and the second se		
 Configuration Data Lock Bit 			
Set Option	Set		
Lock bits	FFFFFFFFFFFFFFFFFFFFFFFF	FFFFFFFFFFF	
 Anti-Rollback Settings 		Mania 04/00040041	~
Set Option	Do Nothing	warning(wood4001)	~
ARCLS	FFFF FFFF		
ARCCS	FFFF FFF	A Once this setting is set to a device the setting	
V OEM Root Public Key		removed. (Configuration Data Lock Bit)	annot be
Set Option	Do Nothing	removed. (comiguration bata cock big	
OEM Root Public Key File 1			
Disable Rewriting	No		

図 3. RFP を使用したデータフラッシュオプション設定レジスタのロック方法



RA8 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

1.2 FSBL ベースの起動オプション

FSBL が有効になっている場合、OEM_BLの検証には2つのスキームがサポートされています。

- CRC32 ブート:アプリケーションの整合性は CRC32 を使用して検証されます。
- セキュアブート:アプリケーションの信頼性は secp256r1 ECC 曲線(NIST P256)を使用してプログラミン グ時に検証されます。MCU の HUK で HMAC-SHA256 を使用して実行時に検証されます。

FSBL の Setting 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。以下のように FSBL の Setting は、**Secure 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基づいて、これらの Setting を調整してください。さまざまなスタッ クプロパティの詳細については、GitHub の RA Flexible Software Package Documentation > API Reference > BSP > MCU Board Support Package の <u>Build Time Configurations</u>を参照してください。

🖹 Problem	ns 📮 Console 🔲 Properties 🗙 🏟 Smart Browser	🥋 Smart Manual 🔋 Memory
EK-RA8	M1	
Settings	Property	Value
	> R7FA8M1AHECBD	
	✓ RA8M1	
	series	8
	✓ RA8M1 Family	
	> Security	
	> OFS0 register settings	
	> OFS1_SEL register settings	
	> OFS1 register settings	
	> OFS2 register settings	
	> Block Protection Settings (BPS)	
	> Permanent Block Protection Settings (PBPS)	
	 First Stage Bootloader (FSBL) 	
	 FSBL Control 0 (FSBLCTRL0) 	
	FSBLEN	Enabled
	FSBLSKIPSW	Enabled
	FSBLSKIPDS	Enabled
	FSBLCLK	240 MHz
	 FSBL Control 1 (FSBLCTRL1) 	
	FSBLEXMDFSBLEN	Secure boot with report measurement
	 FSBL Control 2 (FSBLCTRL2) 	
	PORTPN	PORTn07
	PORTGN	PORT1m
	 Code Certificates (SACCn) 	
	SACC0	0x2006000
	SACC1	0x2000000
	FSBL Measurement Report Address (SAMR)	0x22001000

図 4. FSBL の Setting

RA8 MCU 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 Setting 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。MCU は、FSBL を含まない他の RA 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 Setting 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 Setting します。

CRC とセキュアブートの両方で、図 5 の FSBL を使用した検証オプションに示されているように、レポート測定オプションの有無を選択することができます。



ルネサス RA ファミリ RA8 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

 FSBL Control 1 (FSBLCTRL1) FSBLEXMDFSBLEN FSBL Control 2 (FSBLCTRL2) PORTPN 	Secure boot with report measurement CRC boot without report CRC boot with report measurement
PORTGN	Secure boot without report

図 5. FSBL を使用した検証オプション

レポート測定を選択した場合、図6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 Setting されます(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 ARM PSA Certified Attestation API マニュアルを参 照してください。

SRAM address specified by SAMR register + 0x00	SHA2-256 hash value of OEM_BL and FSBLCTRL1[7:0]
SRAM address specified by SAMR register + 0x20	Signer ID (SHA2-256 hash value of OEM_BL_PK)
SRAM address specified by SAMR register + 0x40	Version number of OEM_BI
SRAM address specified by SAMR register + 0x43	
Note: OEM_BL and FSBLCTRL[7:0] are entered into the hash function	in this order.

図 6. 測定レポートの内容と場所

CRC ブートの場合、Signer ID は計算された CRC を 8 回出力します。各 CRC32 は 4 バイトで、Signer ID は 32 バイトです。

ブートに失敗すると、FSBLCTRL2 レジスタで Setting されたポートに "High"出力され(図4を参照)、 MCU は CPU スリープモードに移行します。例えば、図4では P107 ピンがセキュアブート失敗時に"High "を出力するピンとして選択されています。このピンはボード上の赤色の LED を制御し、ブートシーケン スが失敗すると LED を点灯します。



RA8 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

1.2.1 製品へのプログラミング中の OEM_BL 検証の概要

製品へのプログラミング中、OEM_BL は ECDSA を使用して MCU ブートアプリケーションプログラムによ って検証されます。RA8 MCU は、JTAG、SCI、および USB インタフェースを介して MCU ブート モード へのアクセスを提供します。ブートモードの詳細については、ユーザーズマニュアル ハードウェア編の動 作モードを参照してください。

ブートアプリケーションプログラムは、例えば RFP を通して通信が確立した後、OEM_ROOT_PK のハッシュをシステムの Root of Trust としてマッピングされていないフラッシュ領域にプログラムします。また OEM_BL が MCU にプログラムされると、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、ブートアプ リケーションプログラムは、OEM_ROOT_PK(キー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、ブートアプリケーションプログラム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、ブートアプリケーションプログラム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。



RA8 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



図 7. 製品プログラミングモードでの OEM_BL 検証フロー



1.2.2 シングルチップ動作時の OEM_BL 検証

シングルチップモードでは、OEM_BL は HMAC-SHA256 を使用して FSBL で検証されます。OEM_BL 製品 ヘプログラミングが成功したと仮定します。セキュアブートが有効になっている場合、MCU リセット後に ROM 内の FSBL が実行されます。FSBL は、OEM_BL とコード証明書の HMAC 値(OEM_BL_digest)を計算 し、製品ヘプログラミング中、コードフラッシュに格納される OEM_BL_digest と比較します。 OEM_BL_digest が一致する場合、FSBL は OEM_BL にジャンプします。一致しない場合、FSBL は CPU を スリープモードに移行させ、オプションでユーザが選択したポートピンに"High"を出力します。

CRC ブートが選択されると、FSBL は OEM_BL の CRC を計算し、コード証明書にあるとされる CRC 値と 比較します。CRC 値が一致した場合、FSBL は OEM_BL にジャンプします。



図 8. シングルチップモードでの OEM_BL の検証



1.2.3 シングルチップモードで HMAC-SHA256 を使用する利点

HMAC-SHA256 は、RA8 MCU のハードウェア機能に基づいて以下の利点を提供します。:

- HMAC の共有シークレット(MCU の HUK)が MCU の外部に公開されないため、セキュリティが維持されます。
- HUK は MCU 固有であり、複製の防止、保護を提供します。
- デジタル署名の検証よりも動作が大幅に高速です。
- HMAC-SHA256のオペレーションには量子耐性があります。

1.3 認証チェックに使用される認証情報の詳細

このセクションでは、ECDSA および HMAC-SHA256 操作に基づくシステムの Chain of Trust について説明 します。

システムの Chain of Trust は、以下のコンポーネントによって確立されます。



図 9. Chain of Trust

以下のセクションでは、図9に示されたさまざまな認証情報の形式と内容、製品のプログラミング中や製品 アプリケーションプログラムの実行前におけるその使用方法について説明します。



RA8 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

1.3.1 Root of Trust

セキュアブートを使用する場合は、2 セットの secp256r1 ECC キーペアを生成する必要があります。:

- OEM_ROOT_PK: OEM Root Key pair の公開部分
- OEM_ROOT_SK: OEM Root Key pair のプライベート(シークレット)部分
- OEM_BL_PK: OEM_BL Key pair の公開部分
- OEM_BL_SK: OEM_BL Key pair のプライベート(シークレット)部分

OEM_ROOT_PK の SHA256 ハッシュは、RA8 MCU のユーザーズマニュアル ハードウェア編のセキュア キーインジェクションで説明されているプロセスを使用して、Root of Trust としてロック可能なデータフラ ッシュ領域(HOEMRTPK)に格納されます。Root of Trust のインジェクションは、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。

1.3.2 キー証明書

キー証明書は、HMAC ブート対応アプリケーション製品へのプログラミング時に重要な役割を果たします。 以下は、キー証明書の作成と使用に関する重要な情報です。

- キー証明書は OEM_ROOT_SK によって署名されます。
- OEM_ROOT_PK は署名されたキー証明書に含まれ、HOEMRTPK によって検証されます。
- OEM_BL_PK のハッシュはキー証明書に含まれており、ブートアプリケーションプログラムによって OEM_BL_PK を検証するために使用されます。
- キー証明書は OEM_BL_PK が検証された後に破棄されます。



図 10. OEM_BL_PK を検証するためのキー証明書



RA8 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

1.3.3 コード証明書

コード証明書は、MCUの製品へプログラミング時、MCUのシングルチップ動作モード時において重要な役割を果たします。

- コード証明書は OEM_BL_S 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。キー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、フラッシュメモリ(コードフラッシュまたはデータフラッシュ) にプログラムする必要があります。
- セキュア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1~64であり、セキュアブート付きFSBLが有効な場合、アプリケーションは64回更新できることになります。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、新しいアプリケーション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、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を参照してください。
- フラッシュ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起動領域がデフォルトブロック(ブロック 0)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起動領域が代替ブロック(ブロック 1)の場合に、開始アドレスを指定します。SACCx レジスタ Settingは、RA スマートコンフィギュレータと FSP ボードサポートパッケージ(BSP)によって処理されます。図 4 に Code Certificate プログラミングの位置を示しています。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。
- FSBL 署名の認証では、TLV EXPECTED_CRC フィールドは使用されません。コード証明書の構成は変 更されません。
- 製品ヘプログラミング中に OEM_BL の検証に成功すると、ブートアプリケーションプログラムは 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。ブートアプリケーションプログラムは、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。

Field		Size [byte]	Description
TLV EXPECTED_MAC	Type & Length	4	Fixed value 0x30184008
	Value	32	Unique OEM_BL_digest in each MCU

図 11. OEM_BL_digestの詳細





図 12. 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



RA8 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

1.3.4 製品へのプログラミングとセキュアブートにおける認証情報の概要

アプリケーション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_BOOT_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、セキュア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、セクション2で説明しま す。



図 13. セキュアブート製品のプログラミング認証情報の概要



RA8 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

1.4 整合性チェックで使用される認証情報の詳細

CRC を使用した検証は比較的簡単です。上記のセキュアブートプロセスとの違いは以下のとおりです。:

- OEM Root Public Key(OEM_ROOT_PK)は使用されないので、OEM_ROOT キーペアを作成する必要 も、OEM_ROOT_PK をインジェクションして OEM_ROOT_PK のハッシュ(HOEMRTPK)を作成する必 要もありません。
- キー証明書は使用されず、OEM_BLは署名されないので OEM_BL キーペアを作成する必要はありません。
- アンチロールバックポリシーは適用されません。アンチロールバックカウンタ(ARC_OEMBLn)とコード 証明書のイメージバージョンフィールドは使用されません。
- 製品ヘプログラミング中に、ブートアプリケーションプログラムは OEM_BL の CRC32 を計算します。 計算された CRC32 の値がコード証明書の値と一致する場合、ブートアプリケーションプログラムは SACC0/1 レジスタで指定された場所にコード証明書をプログラムします。
- シングルチップモードでは、FSBLはOEM_BLのCRC32を計算し、コード証明書のCRCと比較します。CRCの値が一致すると、FSBLはOEM_BLにジャンプします。

1.4.1 コード証明書

CRC ブートのコード証明書には、以下のコンポーネントが含まれています。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。イメージバージョンフィールド は、ブート中には使用されません。



図 14. 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



RA8 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

1.4.2 製品へのプログラミング、CRC ブートにおける認証情報の概要

RFP によるアプリケーション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(ルネサスセキュアキーマネージメントツール)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、セクション 3 に説明します。



図 15. CRC ブート製品へのプログラミング認証情報の概要



RA8 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

1.5 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

RA8 MCU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 Setting 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。セキュアファクトリプログラミング後にデバイスを消去して FSBLをオフにするに は、MCU Initialize Device コマンドを実行します。(図 23 のように RFPを使用します)。詳細については、 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を参照してください。



RA8 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

2. 認証された製品プログラミングと HMAC-SHA256 ブートデモンストレーション

このセクションでは、認証された製品プログラミングと HMAC ブートの認証の手順を説明します。なお、 本 APN には以下も同梱しており、これらを使用したロセスを示しています。

blinky_hmac_boot_ra8m1.zip blinky_crc_boot_ra8m1.zip

Root of Trust を導入する手順は、アプリケーションノート R11AN0496 で説明されているセキュアキーイン ジェクションプロセスと多くの共通点があるので、必要に応じて R11AN0496 のセクションを参照してくだ さい。

2.1 ECC secp256r1 キーペア 2 セットの準備

前のセクションで説明したように、システムの Chain of Trust を構築するには 2 セットの ECC secp256r1 キーペアが必要です。このアプリケーションプロジェクトでは、OpenSSL を使用してキーペアを生成する 方法が示されています。さらに、NIST CAVP ECDSA テストベクタを使って Chain of Trust を構築する方法 も説明されています。

2.1.1 OpenSSL を使用した ECC キーペアの生成

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

Windows の場合は、次のリンクから OpenSSL を取得できます: <u>https://slproweb.com/products/Win32OpenSSL.html</u>

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\bin フォルダに移動します。

以下のコマンドで OEM Root Key Pair を生成します。:

C:\Program Files\OpenSSL-Win64\bin>openssl ecparam -name prime256v1 -genkey -noout - out c:\RA8_FSBL\openssl_keys\oem_root_private_key.pem

以下のコマンドで OEM Root Key Pair の Public Key を生成します。

C:\Program Files\OpenSSL-Win64\bin>openssl ec -in c:\RA8 FSBL\openssl keys\oem root private key.pem -pubout -out

c:\RA8 FSBL\openssl keys\oem root public key.pem

read EC key

writing EC key

同じコマンドを使用して OEM_BL Key Pair を生成することができます。:

C:\Program Files\OpenSSL-Win64\bin>openssl ecparam -name prime256v1 -genkey -noout - out c:\RA8_FSBL\openssl_keys\oem_bl_private_key.pem

以下のコマンドで、OEM_BL Key Pair の Public Key を生成します。

C:\Program Files\OpenSSL-Win64\bin>openssl ec -in c:\RA8_FSBL\openssl_keys\oem_bl_private_key.pem -pubout -out c:\RA8_FSBL\openssl_keys\oem_bl_public_key.pem

read EC key

writing EC key



2.1.2 ECC キーペアとした NIST CAVP テストベクタのダウンロード

このアプリケーションのプロジェクトでは、FSBL を使用した HMAC によるブートをデモンストレートする ために、以下の NIST Cryptographic Algorithm Validation Program(CAVP)テストベクタが使用されます。

NIST CAVP テストベクタは、以下のリンクからダウンロードできます。使用するのは ECDSA ベクタで す。

Cryptographic Algorithm Validation Program | CSRC (nist.gov)

lest vectors								
Use of these test vectors does not replace validation obtained through the CAVP.								
The test vectors linked 186-2 and FIPS 186-4) (The test vectors linked below can be used to informally verify the correctness of digital signature algorithm implementations (in F 186-2 and FIPS 186-4) using the validation systems <u>listed above</u> . Response files (.rsp) : the test vectors are properly formatted in response (.rsp) files. Vendor response files must match this formate exactly.							
Response files (.rsp): exactly.								
Intermediate results	Intermediate results files (.txt): files with intermediate results (.txt) are supplied to help with debugging.							
See the README file in	each zip file for details.							
	Publication	Algorithm Test Vestors						
	Publication FIPS 186-4	Algorithm Test Vectors						

図 16. ECDSA テストベクタ

zip ファイル 186-4ecdsatestvectors.zip をダウンロード後、解凍してプレーンテキストファイル KeyPair.rsp で以下のベクタを見つけてください。

OEM ルートキーペア用の ECC P256 secp256r1 キーペア

以下のベクタは、OEM Root Key Pair に使用されます。これはセクション 2.2 で Wrapped OEM Root Public Key の生成に使用されます。また、セクション 2.4 でキー証明書を生成する際にも使用されます。

d = c9806898a0334916c860748880a541f093b579a9b1f32934d86c363c39800357

Qx = d0720dc691aa80096ba32fed1cb97c2b620690d06de0317b8618d5ce65eb728f

Qy = 9681b517b1cda17d0d83d335d9c4a8a9a9b0b1b3c7106d8f3c72bc5093dc275f

OEM APP(BL)キーペア用の ECC P256 secp256r1 キーペア

以下のベクタは、OEM APP(BL)キーペアに使用されます。これは、セクション 2.4 でコード証明書の生成 するために使用されます。

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



RA8 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

2.2 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インジェクションされます。

OEM_ROOT_PK は、RFP を使用して MCU にインジェクションする前に、ルネサス DLM サーバを使って ラップする必要があります。OEM Root Public Key をラップする前に、Renesas Key Wrap Service を使用 して User Factory Programming Key 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、R11AN0496の以下のセクションを参照して、使用背景を理解し、ルネサ ス DLM サーバとの PGP 暗号化通信を確立してください。

- 顧客の PGP キーペアを生成するには、Create PGP Key Pair 章を参照してください。
- DLM サーバに登録するには、Registration with DLM Server 章を参照してください。
- DLM サーバと暗号化通信を確立するには、Exchange User and Renesas PGP Public Keys 章を参照 してください。

2.2.1 Wrapped User Factory Programming Key の生成

R11AN0496の上記のセクションに基づく操作完了後、次は R11AN0496の Wrapping the UFPK 章を参照 して W-UFPK を生成します。ただし、このプロセス中に更新すべき重要な手順がいくつかあります。

まず、Wrapping the UFPK 章を読み、その後のセクションを読む前に、OEM Root Public Key インジェク ションの目的で Wrapped UFPK を生成するために操作を更新する箇所を特定してください。その後、その フロー内で次に更新するべき操作を特定し、R11AN0496 に従って W-UFPK を生成してください。

- 特定のプレーンテキストの UPFK を入力すると、DLM サーバ(W-UFPK)の出力は、セキュリティエンジンで選択されたモード(互換モードや保護モード)に応じて変わります。OEM ルートキーをラップする時に DLM サーバで選択されたオプション(MCU 保護モードを使用)と一致させるため、SKMT でUFPK を生成する際には、Overview で RA Family、RSIP-E51A Security Functions、Protected Mode オプションを選択してください。
 - R11AN0496 は RSIP 互換モードを使用するため、SKMT で RA ファミリ、RSIP-E51A 互換モード を選択する必要があります。FSBL 使用時に必要なプロテクトモードへの変更にご注意ください。



図 17. RSIP-E51A セキュリティ機能とプロテクトモードの選択



 同じ理由で、DLM サーバを使用して UFPK をラップする場合は、互換モードではなく、DLM and Protected Mode で以下の RA8 の項目を選択してください。



図 18. RA8D1/RA8M1 MCU グループ DLM とプロテクトモードの選択

• Wrapped UFPK を生成するための他の手順は、R11AN0496 で説明されているものと同じです。

このアプリケーションプロジェクトの残りでは、以下の UFPK と W-UFPK がすでに作成されていることを 前提としています。:

ra8x1_ufpk.key:the UFPK

ra8x1_ufpk.key_enc.key: the DLM Wrapped W-UFPK

UFPK と W-UFPK が生成されたら、OEM Root Public Key のラッピングは、R11AN0496 の Wrap an Initial ECC Public Key with the UFPK のパートに記載されている ECC Public Key のラッピングプロセスと同様に 行わなければなりません。



RA8 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

2.2.2 OpenSSL と SKMT CLI インタフェースを使用した Wrapped OEM Root Public Key の生成

Secure Key Management Tool(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 Wrapped OEM Root Public Key を生成します。このキー は、RFP を使用したセキュアキーインジェクションプロセスを使用して MCU に照会されます。

C:\Renesas\SecurityKeyMangementTool\Cli>skmt.exe /genkey /ufpk

Output File: C:\RA8_FSBL\openssl_keys\oem_root_pk_cli.rkey

UFPK: 000102030405060708090A0B0C0D0E0F000102030405060708090A0B0C0D0E0F

W-UFPK: 00000000A7BF7EB27054D78E07C504291520678AA7BF7EB27054D78E07C504291520678A

IV: 55AA55AA55AA55AA55AA55AA55AA55AA

Encrypted key:

E71776A79F2BFF879CE3A434C5D0AEFBA934214518114AA89E7CAD10BD45D25623CE6710EC929971BAD200814 B6D633A670447B51347EA24EC7908C9C66AA933F7DD64E2DBDB1B831CED6E3B1347C69B

この例の oem_root_pk_cli.rkey は、このアプリケーションプロジェクトのサンプルプロジェクトとは 一致しないので、このキーを使用しないでください。



2.2.3 SKMT GUI インタフェースを使用した Wrapped OEM Root Public Key の生成

ここでは、セクション 2.1.2 のように OEM Root Public Key pair を使用して、Wrapped OEM Root Public Key を生成します。

SKMT を起動し Overview で RA Family、RSIP-E51A Security Functions、Protected Mode を選択しま す。(図 17 を参照)。次に、SKMT の Wrap Key タブに移動し、Key Type に OEM Root public を選択し、 Wrapping Key(UFPK、W-UFPK)を入力します(図 19 参照)。

Keys must	be wrapped by t	he UFPK for	r secure injection or by t	he KU	K for secure upda	ate.
Key Type Key Data						
O DLM/AL	AL2_KEY 🗸 🗸	○ AES	128 bits	\sim	O ARC4	
⊖ KUK	(⊖ RSA	2048 bits, public	\sim		
OEM Root public	(OECC	secp256r1, public	\sim		
	(SHA256-HMAC	\sim		
Wrapping Key						
UFPK UFPK File:	C:\RA8_FSBL\	ufpk\ra8x1	_ufpk.key			Browse
W-UFPK File	C:\RA8 FSBL	ufpk\ra8x1	ufpk.kev enc.kev			Province

図 19. OEM Root Public Key のラップ



Key Data タブで、OEM Root Public Key を入力します。



図 20. OEM Root Public Key の提供

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、Generate File をクリックし、Wrapped OEM Root Public Key を生成します。

Overview Ger	nerate UFPK	Generate KUK Wrap Key TSIP UPDATE	FSBL DOTE SEP	
k	Keys must be	e wrapped by the UFPK for secure injection or	by the KUK for secure update.	
Кеу Туре Ке	ey Data			
⊖ File				Browse
Raw Qx :	:	d0720dc691aa80096ba32fed1cb97c2b6206	90d06de0317b8618d5ce65eb72	8f 🔅
Qy	:	9681b517b1cda17d0d83d335d9c4a8a9a9b	0b1b3c7106d8f3c72bc5093dc2	′5f
O Random -	- Output File	e		Browse
Wrapping	Key			
● UFPK UF	PK File :	C:\RA8_FSBL\ufpk\ra8x1_ufpk.key		Browse
W-	-UFPK File :	C:\RA8_FSBL\ufpk\ra8x1_ufpk.key_enc.key		Browse
О кик ки	JK File :			Browse
IV				
Generate	random valu	ue		
O Use specif	fied value (1	o nex bytes, big endian format)	2200778899AABBCCDDEEFF	
Output	50	C\RA8_ESBL\K2\cem	root key rkey	Desuga
Address : 1	0000	Key name :	IOUL_KEY.IKEY	Browse
Hadresst				
		Generate file		
Output File: C:\l	RA8_FSBL\K	/2\oem_root_key.rkey 000A0D0C0D0E0F000102030405060708090A0B0	CODOEOF	^
W-UFPK: 000000	0002A8434C	A97D0313279389DD8F15523DB2A8434CA97D0	313279389DD8F15523DB	
Encrypted key:	70720 4 0021	1 A A EDE 40 4 70 40 A 3 75 F 7 A 7700 A C A 071 C 0 40 70		745026176
0506120200200	I DIZDA0021	IAAJD0494/B40A3/JF/A//B9ACA0/IC94B2F9	0005FJA4016DD4EA4330B3DEE	AF620170
85861E82DD399 1F02B3D6306D0	073282B820D	02F7CEE956F42BA63B0371C177A4B812D3FD15	B294831	

図 21. Wrapped OEM Root Public Key の生成



この Wrapped OEM Root Public Key(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。この Root Public Key がキー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。

また、同じ Root Public Key

2.3 HMAC-SHA256 ブートを有効にした Blinky アプリケーションの準備

通常、FSBL(First Stage Boot Loader)を有効にしてブートを行う場合、まず FSBL が無効な OEM_BL (Original Equipment Manufacturer Boot Loader)を開発します。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。

「Create a New Project for Blinky」に記載されているように、blinky_hmac_boot_ra8m1 というプロジェク ト例は、RA8M1 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、Trust 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、Setting 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便利にするために、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ポート P107 ピンを使用しており初期状態で Low に Setting していて、FSBL がブート失敗時にポート P107 ピンを 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 は次回の電源サイクル後にのみ有効になります。

初期動作で既に FSBL が有効になっている場合 MCU は、**MCU Initialize** コマンドを実行し、ブートアプリ ケーションプログラムを使用して、オプション設定メモリにある既存の FSBL とトラストゾーンの Setting を消去します。これは、**ルネサスデバイスパーティションマネージャ**(RDPM)(e² studio にネイティブでイン ストールされています。)または RFP を使用して実現できます。

e² studio を起動したら、**Run -> Renesas Debug Tools -> Renesas Device Partition Manager** を選択して、RDPM を起動できます。

EK-RA8M1 では、デフォルトのジャンパ Setting で SCI ブートモードが有効になっており、SCI 信号がデバ ッグインターフェイスに指定されています。RDPM を使用する際は、SCI インタフェース、SWD インタフ ェースのいずれかを選択できます。

以下の画像では、SCI インタフェースが選択されています。



ルネサス RA ファミリ RA8 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

Renesas Device Partition Manager			
Device Family: Renesas RA 🗸 🗸			
Action			
Set TrustZone secure / non-secure bound	Change debug state aries Initialize device back to factory default		
-			
Target MCU connection:	J-Link V		
Connection Type:	sci 🗸		
Emulator Connection:	Serial No V		
Serial No/IP Address:	0		
Debugger supply voltage (V):	0500		
Connection Speed (bps for SCI, Hz for SWD):	3000		
Debug state to change to:	secure software Development		
Memory partition sizes			
			Browse
			DIOWSCII
Code Flash Secure (KB)	32		
Code Flash NSC (KB)			
Data Flash Secure (KB)	0		
SRAM Secure (KB)	8		
SRAM NSC (KB)			
Command line tool:			
C:\Users\a5099044\.eclipse\com.renesas.plat	form_865760450\DebugComp\RA\DevicePartitionManager	RenesasDevicePartitionManagerCmd.exe	Browse
Display errors in : English			^
Connecting Loading library : SUCCESSFU Establishing connection : SUC Checking the device's TrustZone type : CONNECTED.	LI CESSFULI SUCCESSFULI		
Initializing device and rolling back Protection SUCCESSFUL!	Level to PL2		
Disconnecting DISCONNECTED.			
Connection : SUCCESSFULI Device initialization : SUCCESSFULI Device initialization : SUCCESSFULI 			*
<			>
3	Import	Export Run	Close

図 22. RDPM を使用した MCU の初期化

図 23 は、RFP を使用してデバイスを初期化することを選択するコマンドを示します。



図 23. RFP を使用した MCU の初期化

デバイスを初期化した後、以下の手順に従って付属のサンプルプロジェクトをビルドし、実行してください。



RA8 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_ra8m1 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。そして Generate Project Content」をクリックします。 次に、Micro USB ケーブルを使用して、J10 にある J-Link OB USB を評価ボードの USB に接続します。 blinky プロジェクト (blinky hmac boot ra8m1) をビルドして実行し、その動作を確認します。

- コンパイル結果は、セクション 2.4 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。

arm-none-eabi-size --format=berkeley "blinky_hmac_boot_ra8m1.elf"
 text data bss dec hex filename
 11576 92 2220 13888 3640 blinky_hmac_boot_ra8m1.elf
 16:53:29 Build Finished. 0 errors, 0 warnings. (took 396ms)



Tera Term を開き、ボーレートを 115200 に Setting します。

Tera Term: Serial port set	up and connection		×	
Port:	СОМ6 ~	New setting		
Speed:	115200 🗸		-	
Data:	8 bit 🗸 🗸 🗸	Cancel		
Parity:	none 🗸 🗸			
Stop bits:	1 bit \sim	Help		
Flow control:	none ~			
Device Friendly 1	nit delay msec/char 0 Name: JLink CDC UAF	msec/line	<u>\</u>	
Device Instance Device Manufact Provider Name: S Driver Date: 6-6-2	urer: SEGGER SEGGER 2019	D_1024&MI_00{0&2D10		
Driver Version: 1	.34.0.44950			
<		>		

図 25. Tera Term の Setting

New setting 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 Setting、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。出力の意味を理解するには、ユーザーズマニュ アル ハードウェア編のレジスタ定義、コード証明書の内容(図 12)を参照してください。FSBL はまだ有効に なっていないので、version num oem bl フィールドは 0、ダイジェスト領域は 0xFF になります。



RA8 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



図 26. HMAC ブートの J-Link コンソール出力

MCUに Power-ON リセットが発行されると、FSBL が起動しますが、Root of Trust とコード証明書が MCU にプログラムされていないため、blinky は起動しなくなります。アプリケーションを正常に起動するために は、POR が発行される前に OEM_ROOT_PK をインジェクションし、キー証明書とコード証明書を提示します。このプロセスについては、以降のセクションで説明します。セクション 2.6 では、FSBL を有効にし てアプリケーションを起動する方法について詳しく説明します。



2.4 MCU OEM_BL アンチロールバックカウンタ値の取得

セクション 1.3.3 で説明したように、HMAC ブート用のコード証明書には、1~64 までの範囲のイメージバ ージョンのフィールドを含める必要があります。このイメージバージョン番号は、オプション設定メモリ内 の OEM_BL のアンチロールバックカウンタ n(n=0,1)と呼ばれる 2 つのレジスタに永続的に保存されます。 (このレジスタの位置については図 2 を参照)

6.2.25	ARC_OEMBLn : Anti-Rollback Counter for OEMBL (n = 0, 1)						
Base address:	0x2703_0000	2703_0000					
Offset address:	0x878 + 0x004 × n (n =	78 + 0x004 × n (n = 0, 1)					
Bit position:	31		0				
Bit field:		ARC_OEMBL[32 × n + 31 : 32 × n]					
Value after reset: User setting ^{*1}							
Rit S	wmbol	Eurotion	D/M				
31:0	PC OEMBL [32*n	Anti-Bollback Counter for OEM BL Application	RAM				
	31 : 32*n]	The counter value is obtained by arranging the read values from the upper register (n = 1) to the lower register (n = 0). See section 52.12.5. Anti-Rollback Counter for detail					

図 27. OEMBL のアンチロールパック カウンタ

イメージ バージョン番号はこれら 2 つのレジスタに格納された値ではなく、解釈された値であることに注 意してください。このカウンタの動作の詳細については、ユーザーズマニュアル ハードウェア編 セクショ ン 52.12.5 アンチロールバックカウンタを参照してください。製品へのプログラミング中は、ブートアプリ ケーションプログラムのみがこのカウンタにプログラムすることができます。基本的に、アプリケーション の更新が成功するたびに、カウンタは 1 ビットに 1 を Setting していきます。例として、デバイスが 3 回更 新された場合、h'7(b 0111)が 2 つのカウンタに格納されます。これらの 32 ビットカウンタは、64 回のア プリケーション更新に使用できます。

アプリケーションプロジェクトには、このレジスタ値を読み取る J-Link スクリプトが含まれています。 read_arc_oembl_jlink_script.zip を解凍し、ra8m1.bat をダブルクリックして、これら2つのレ ジスタの値を読み取ります。図 28 に、J-Link スクリプトを実行した際の出力例を示しています。この例で は、レジスタ 0x27030878(ARC_OEMBL0)は 0xFFFFFFF の値を保持し、レジスタ 0x2703087C(ARC_OEMBL1)は 0x00000001 の値が保持されています。ARC_OEMBL0 と ARC_OEMBL1 レ ジスタには合計 33 ビットに Setting されています。従って、ユーザが次に選択できるイメージバージョンは 34 以上 64 以下である必要があります。



RA8 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

J-Link SEGGER J-Link Commander V7.920 (Compiled Nov 8 2023 15:47:59) DLL version V7.920, compiled Nov 8 2023 15:46:12 J-Link Command File read successfully. Processing script file... J-Link>device R7FA8M1AH J-Link>device R7FA8M1AH J-Link>device R7FA8M1AH J-Link connection not established yet but required for command. Connecting to J-Link via USB...O.K. Firmware: J-Link 0B-RA4M2 compiled Oct 30 2023 12:13:20 Hardware version: V1.00 J-Link uptime (since boot): 0d 10h 07m 36s S/N: 1082859018 USB speed mode: Full speed (12 MBit/s) VTref=3.300V J-Link>speed 12000 Selecting 12000 kHz as target interface speed J-Link>if swd Selecting SWD as current target interface. J-Link>Mm32 0x27030878 1 Target connection not established yet but required for command. Device "R7FA8M1AH" selected. Connecting to target via SWD ConfigTargetSettings() start ConfigTargetSettings() end - Took 147us InitTarget() start Identifying target device... SMD selected. Executing JTAG -> SWD switching sequence... Initializing DAP... DAP initialized successfully. Determining TrustZone configuration... Secure Debug: Enabled (SSD) Determining currently configured transfer type by reading the AHB-AP CSW register. Determining TrustZone configuration... Secure Debug: Enabled (SSD) Determining currently configured transfer type by reading the AHB-AP CSW register. --> Correct transfer type configured. Done. InitTarget() end - Took 5.7.2ms Found SW-DP with ID 0x6BA02477 DPIDR: 0x6BA02477 Corresignt SoC-400 or earlier Scanning AP map to find all available APs AP[2]: Stopped AP scan as end of AP map has been reached AP[0]: AHB-AP (IDR: 0x84770802) Iterasting through AP map to find AHB-AP to use AP[0]: AHB-AP (IDR: 0x54770802) Iterasting through AP map to find AHB-AP to use AP[0]: AHB-AP (KIDR: 0x54770802) Iterasting through AP map to find AHB-AP to use AP[0]: AHB-AP (M base: 0xE00FE000 CPUID register: 0x4010FD232. Implementer code: 0x41 (ARM) Feature set: Mainline Cache: L1 I/D-cache present Found Cortex-MBS r0p2, Little endian. FPUnit: 8 code (GP) slots and 0 literal slots Security extension: implemented Secure debug: enabled Coresight components: ROWTDL[0]: E00FF000 CID BI05100D PID 000BB4D4 ROM Table 000Tb1[1] & E00FF000 RowTb1[6] @ E00FF000 [0][0]: E00FF000 CID B105100D PID 000BB4D4 ROM Table RowTb1[1] @ E00FF000 [1][0]: E00F000 CID B105900D PID 000BBD23 DEVARCH 47702A04 DEVTYPE 00 ??? [1][1]: E0001000 CID B105900D PID 000BBD23 DEVARCH 47701A03 DEVTYPE 00 PFB [1][2]: E0002000 CID B105900D PID 000BBD23 DEVARCH 47701A03 DEVTYPE 00 FPB [1][3]: E0001000 CID B105900D PID 000BBD23 DEVARCH 47701A03 DEVTYPE 13 TIM [1][5]: E0041000 CID B105900D PID 000BBD23 DEVARCH 47704A03 DEVTYPE 13 TIM [1][5]: E0041000 CID B105900D PID 000BBD23 DEVARCH 47704A03 DEVTYPE 13 ETM [1][6]: E0042000 CID B105900D PID 000BBD23 DEVARCH 47704A13 DEVTYPE 13 ETM [1][7]: E0042000 CID B105900D PID 000BBD23 DEVARCH 47704A14 DEVTYPE 14 CSS60 [0][1]: E004000 CID B105900D PID 000BBD23 DEVARCH 47701A14 DEVTYPE 14 CSS60 [0][1]: E004000 CID B105900D PID 000BBD23 DEVARCH 47701A14 DEVTYPE 14 CSS60 [0][1]: E004000 CID B105900D PID 000BBD23 DEVARCH 47701A14 DEVTYPE 14 CSS60 [0][1]: E004000 CID B105500D PID 000BBD23 DEVARCH 47701A14 DEVTYPE 14 CSS60 [0][1]: E004000 CID B105500D PID 000BBD23 DEVARCH 47701A14 DEVTYPE 14 CSS60 [0][1]: E004000 CID B105500D PID 000BBD23 DEVARCH 47701A14 DEVTYPE 14 CSS60 [0][1]: E004000 CID B105500D PID 000BBD23 DEVARCH 47701A14 DEVTYPE 14 CSS60 [0][1]: E004000 CID B105500D PID 000BBD23 DEVARCH 47701A14 DEVTYPE 14 CSS60 [0][1]: E004000 CID B105500D PID 000BBD23 DEVARCH 47701A14 DEVTYPE 11 TPIU T-cache L1; 16 KB, 25 Sets, 3; 28 VE56/Line, 2-Nav 600-CTI -Cache L1: 16 KB, 256 Sets, 32 Bytes/Line, 2-Way -Cache L1: 16 KB, 128 Sets, 32 Bytes/Line, 4-Way Wemory zones: Zone: "Default" Description: Default access mode Cortex-M85 identified. 27083078 = FFFFFFF D-Link>Mem32 0x2703087C 1 D-Link>Mem32 0x2703087C 1 2703087C = 00000001 D-Link>rx 100 Reset delay: 100 ms Reset type NORMAL: Resets core & peripherals via SYSRESETREQ & VECTRESET bit. Reset: RMNOM core with Security Extension enabled detected. Reset: Halt core after reset via DEMCR.VC_CORERESET. Reset: Reset device via AIRCR.SYSRESETREQ. ⊃Link>c Λεστιτη J-Link>g Memory map 'after startup completion point' is active Script processing completed.

図 28. J-Link スクリプトによる ARC_OEMBL レジスタのリード



2.5 鍵証明書とコード証明書の生成

前述のように、HMAC-SHA256 ベースのブートオプションでは、鍵とコード証明書を生成するには、2 セットの ECC-P256 secp256r1 の公開鍵と秘密鍵が必要です。このセクションでは、2 つのデモンストレーションを提供しています。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ーフェイスの使用
- NIST ベクタと SKMT GUI インタフェースの使用

2.5.1 OpenSSL と SKMT コマンドラインインターフェイスの使用

鍵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 Setting 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 12 参照)。この例では 3 に Setting されており、ARM_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フラッシュサイズ。
- 入力:/oembl アプリケーション バイナリ。
- 入力:/oemroot private OEM_ROOT_SK。
- 入力:/oembl private OEM_BL_ROOT_SK。
- 出力:/output codecert、/output keycert キー証明書とコード証明書。

コマンドウィンドウを開き、\SecurityKeyMangementTool\cliフォルダに移動し、以下のコマンドを 実行して鍵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\SecurityKeyMangementTool\cli>skmt.exe /gencert /mode "signature" /loadaddr "0200000" /oembl_size "8000" /cfsize "2000000" /ver "3" /oembl "C:\RA8_FSBL\fsp_v510\blinky_hmac_boot_ra8m1\Debug\blinky_hmac_boot_ra8m1.srec" /oembl_private file="C:\RA8_FSBL\openssl_keys\oem_bl_private_key.pem" /oemroot_private file="C:\RA8_FSBL\openssl_keys\oem_root_private_key.pem" /output_codecert "C:\RA8_FSBL\openssl_keys\blinky_hmac_openssl_ccert.bin" /output_keycert "C:\RA8_FSBL\openssl_keys\blinky_hmac_openssl_kcert.bin"

Output File: C:\RA8_FSBL\openssl_keys\blinky_hmac_openssl_kcert.bin

Output File: C:\RA8_FSBL\openssl_keys\blinky_hmac_openssl_ccert.bin

2.5.2 NIST ECC のキーペアと SKMT GUI インタフェースの使用

このセクションでは、証明書の生成を説明するために、セクション 2.1.2 で提供された NIST CAVP ECC secp25r1 の 2 セットのキーペアを使用します。キーとコード証明書を生成するために、キーペアのプライ ベートキー部分と公開キー部分の両方を SKMT に提供する必要があります。

SKMT GUI を起動します。Overview で、RA Family、RIP-E51A Security Functions、Protected Mode を 選択します。

次に、セクション2.1で生成された.srecファイルを選択し、アプリケーションのバージョンを定義します。



Recurity Key Management Tool	-		×
File View Help Overview Generate UFPK Generate KUK Wrap Key TSIP UPDATE FSBL D	OTF SFP		_
The First Stage Bootloader supports multiple use cases for checking applica Bootloader, for authenticity and/or integrity prior to programming ar Please refer to device-specific documentation for complete descript	ation code, such as nd prior to executio tions of all use case	an OEM on. s.	
	ac hoot ra8m1 sree	Brows	e

図 29. アプリケーションとバージョン番号の選択

図 29 のようにイメージバージョン定義は、コード証明書に保存されます。セクション 1.3.3 で説明したように、許可されるバージョン番号は 1~64 であり、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 Setting できます。

FSBL ページで OEM Root key を Setting します。



図 30. OEM ルートキーの提供

次に、OEM ブートローダキーを提供します。



図 31. OEM ブートローダキーの提供



Г

RA8 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、Certificates タブに移動し、キー証明書とコード証明書の名前と場所を定義し、Generate File(s)を クリックします。キー証明書とコード証明書が生成されます。

Certificates OEM Root Keys OEM Boot	loader Keys
Code Flash Start Address (hex) :	02000000
Device Code Flash Size :	2 MB 🗸 🗸
	(If exact size option is not available, select the next lower size)
OEM Bootloader Size :	ally
O Enter manually (hex))
Key Certificate : 48_FSBL\K2\oem_root_	_key_and_certs blinky_hmac_boot_ra8m1_key_cert.bin Browse
Code Certificate : 3_FSBL\K2\oem_root_k	xey_and_certs <mark>.</mark> blinky_hmac_boot_ra8m1_code_cert.bin Browse
	Generate File(s)
start File CARAO FCRIAKO - and shat have a	nd_certs blinky_hmac_boot_ra8m1_key_cert.bin
utput File: C:\KA8_FSBL\K2\0em_root_key_ar	
utput File: C:\RA8_FSBL\R2\0em_root_Rey_ar utput File: C:\RA8_FSBL\R2\0em_root_key_ar PERATION SUCCESSFUL	nd_certs <mark>blinky_hmac_boot_ra8m1_code_cert.bin</mark>

図 32. キーとコード証明書の生成



RA8 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 NIST ベクトルも出力できます。これは、Random – Output File を選択することで可能です。

OVERVIEW Generate CUPYA Generate CUPYA USIF DUF SFP The First Stage Bootloader supports multiple use cases for checking application code, such as an OEM Bootloader, for authenticity and/or integrity prior to programming and prior to execution. Please refer to device-specific documentation for complete descriptions of all use cases. OEM Bootloader Image: Nky_hmac_boot_ra&m1\Debug\blinky_hmac_boot_ra&m1.srec Browse Programming Verification Method : (*) Signature Image Version : 32 OEM Certificates OEM Rootloader Keys OEM Bootloader Private Key OEM Bootloader Private Key File Browse File @ Random - Output File C:\RA&_FSBL\K2\oem_root_key_and_certs\oembl_private.key Browse OEM Bootloader Public Key File Browse @ Raw (*) (*) (*) @ Raw (*) (*) (*)	Oversien Consult UEDV C	STATE FOR A STATE AND A STATE	
The First Stage Bootloader supports multiple use cases for checking application code, such as an OEM Bootloader, for authenticity and/or integrity prior to programming and prior to execution. Please refer to device-specific documentation for complete descriptions of all use cases. OEM Bootloader Image : kw_hmac_boot_ra8m1.sre Browse Programming Verification Method : sre Browse Programming Verification Method : Signature Image Version : 32 CRC Cettificates OEM Rootloader Private Key File Browse @ Random - Output File C\rRA8_FSBL\KZ\oem_root_key_and_certs\oembl_private.key Browse Browse	Overview Generate OFPK G	enerate KUK Wrap Key ISIP UPDATE FSBL DUTF SFP	
OEM Bootloader Image: 1ky_hmac_boot_ra8m1\Debug\blinky_hmac_boot_ra8m1.srec Browse Programming Verification Method: Image Version: 32 Image CRC Image Version: 32 Image Version: Image Version: Image Version: Image Version: Image Version: Image Version: Image Version: Image Version: Image Version: Image Version: Image Version: Image Version: Image Version: Image Version: Image Version: Image Version: Image Version: Image	The First Stage Bootloa Bootloader, for a Please refer to d	Jer supports multiple use cases for checking application code, su thenticity and/or integrity prior to programming and prior to exe evice-specific documentation for complete descriptions of all use	h as an OEM cution. cases.
Programming Verification Method:	OEM Bootloader Image :	1ky hmac boot ra8m1\Debug\blinky hmac boot ra8m1	.srec Browse
O CRC Certificates OEM Root Keys OEM Bootloader Private Key File Raw Image: Random - Output File CL\RA8_FSBL\KZ\oem_root_key_and_certs\oembl_private.key Browse OEM Bootloader Public Key File Image: Raw Qy:	Programming Verification M	ethod : Signature Image Version : 32	
Certificates OEM Root Keys OEM Bootloader Private Key File Raw C:RA8_FSBL\K2\oem_root_key_and_certs\oembl_private.key Browse Browse OEM Bootloader Public Key Browse File Browse OEM Bootloader Public Key Browse OIM Bootloader Public Key Browse OIM graw Qy :	2 2	⊖ CRC	
OEM Bootloader Private Key File Browse Raw C:\RA8_FSBL\K2\oem_root_key_and_certs\oembl_private.key Browse OEM Bootloader Public Key File Browse File 0 0 @ Raw 0 0 Year 0 0 OEM Bootloader Public Key Browse 0 Ogy: 0 0	Certificates OEM Root Ke	VS OEM Bootloader Keys	
File Browse Raw C:\RA8_FSBL\K2\oem_root_key_and_certs\oembl_private.key Browse Browse OEM Bootloader Public Key Browse File Browse @ Raw Qy:	OEM Bootloader Priv	ite Key	
Raw Image: Second sec	⊖ File		Browse
Random - Output File C:\RA8_FSBL\K2\oem_root_key_and_certs\oembl_private.key Browse File Raw 0x: Qy:	○ Raw		<u>^</u>
OEM Bootloader Public Key File Browse @ Raw Qx : Image: Constraint of the second	Random - Output File	C:\RA8_FSBL\K2\oem_root_key_and_certs\oembl_private.key	Browse
O File Browse @ Raw Qx : 0 Qy : 0	OFM Bootloader Pub	ir Kev	
Image: Operation of the second sec			Browse
Qy:	Raw Qx:		DIOWSE
Qy:			~
	Qy:		0
			~
		Generate File(s)	
Generate File(s)	Jutput File: C:\RA8_FSBL\K2\or Jutput File: C:\RA8_FSBL\K2\or Jutput File: C:\RA8_FSBL\K2\or Jutput File: C:\RA8_FSBL\K2\or Jutput File: C:\RA8_FSBL\K2\or	em_root_key_and_certs\oembl_private_private.key em_root_key_and_certs\oembl_private_public.key em_root_key_and_certs\blinky_hmac_boot_ra&m1_key_cert.bin em_root_key_and_certs\blinky_hmac_boot_ra&m1_code_cert.bin	

図 33. SKMT に保存された NIST テストベクトルの出力

図 32、図 33 で生成された証明書は、このアプリケーションプロジェクトで説明されている他の操作に使用 できます。

SKMT v1.05 には、イメージバージョン 64 でコード証明書を生成できないバグがあることに注意してください。イメージバージョン 64 が要求されると以下のエラーが表示されます。



ルネサス RA ファミリ RA8 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

File View Help Overview Generate	JFPK Generate KUK Wrap Key TSIP UPDATE FSBL DOTF SFP		^
The First Stage Bootload Please r	Bootloader supports multiple use cases for checking application code, su er, for authenticity and/or integrity prior to programming and prior to ex efer to device-specific documentation for complete descriptions of all us	uch as an OEM recution. e cases.	
OEM Bootloader Im Programming Verifi	age :	n1.srec Brow	se
Certificates OEM	Root Keys OEM Bootloader Keys		
Code Flash Start A Device Code Flash	ddress (hex) : 02000000 Size : 2 MB ~ (If exact size option is not available, select th	e next lower siz	ze)
OEM Bootloader S	2e : Calculate automatically Enter manually (hey)		
Key Certificate :	C:\RA8_FSBL\K2\oem_root_key_and_certs\blinky_hmac_boot_ra8m1_key	y_cer Browse	
Code Certificate :	C:\RA8_FSBL\K2\oem_root_key_and_certs\blinky_crc_boot_ra8m1_code	_cert. Browse	
	Generate File(c)		

図 34. イメージバージョン 64 でコード証明書生成に関する SKMT v1.05 のバグ



RA8 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

2.6 認証付き製品プログラミングと HMAC ブートのデモンストレーション

rfp_ra8m1.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_ra8m1.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、SWD 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るため、USB ケーブルを使用してブートモードにアクセ スできます。

以前のアプリケーションで FSBL 設定になっている場合や、TrustZone[®]境界が Setting されている場合は、 次のセクションに進む前に**デバイスの初期化**(図 23 を参照)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 TrustZone[®]設定が削除します。

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 にイ ンジェクションされます(セクション 1.3.4 を参照)。以下の動作**設定**が Setting 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。

Command	Erase Options
Erase	Erase Selected Blocks \sim
Program	Program & Verify Options
Verify	Erase Before Program
Program Bash Options	Verify by reading the device \sim
Verify Bash Ontions	
	Checksum Type
	CRC-32 method
Fill with 0xFF	
Code Area / User Boot Area	Error Settings
Data Area	Enable address check of program file

図 35. 動作設定の構成



RA8 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。

Operation Operation Settings Block Settings Flash Options Connect Settings Unique Code
Project Information
Current Project: rfp_ra8x1.rpj
Microcontroller: R7FA8M1AHECBD
Program and User Key Files
Add/Remove Files
Command ✓ File Details – X
Add File(s) Remove Selected File(s)
File Name Type Address/Offset
C:\RA8_FSBL\fsp_v510\blinky_hmac_boot_ra8m1\Debug\blinky_hmac_boot_ra8m1.srec HEX
O OK Cancel
5

図 36. アプリケーションプロジェクトを選択

Flash Options 設定で OEM Root Public Key、Code Certificate、Key Certificate を Setting します。

Disable Initialize Command No	
	^
Disable AL2 authentication No	
Disable AL1 authentication No	
Disable LCK_BOOT transition No	
Configuration Data Lock Bit	
Set Option Do Nothing	
Lock bits	FF
 Anti-Rollback Settings 	
Set Option Do Nothing	
ARCLS	
ARCCS HEX FFFF	
V OEM Root Public Key	_
Set Option Set	
OEM Root Public Key File 1 C:\RA8_FSBL\K2\oem_root_key.rkey	
Disable Rewriting	
✓ Certificate	_
Set Option Set	_
Verification Method Signature	
Code Certificate	ert.bin
Key Certificate C:\RA8_ESBI_K2\oem_root_key_and_ce	erts\blinky ∀

図 37. OEM ルートキーと証明書の Setting



RA8 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 Operation タブに移動し、Start ボタンをクリックします。RFP は SREC 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 HMAC 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。次の電源投入サイクルで、 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 が点滅するはずです。

Current Project: rfp_ra8x1.rpj		
Microcontroller: R7FA8M1AHECBD		
Program and User Key Files		
C:\RA8_FSBL\fsp_v510\blinky_hmac_boot_ra8m1\Debug\blink	y_hmac_boot_ra8m1.srec	
CHC-32: 918431B3	Add/Hemove Hies	
	OK	
Config Area 1] 0x0300A100 - 0x0300A11F size: 32 Config Area 1] 0x0300A130 - 0x0300A13F size: 16 Config Area 2] 0x0300A200 - 0x0300A20F size: 208 [Config Area 4] 0x27030080 - 0x2703035F size: 736		^
Config Area 1 0x0300A100 - 0x0300A11F size : 32 Config Area 1 0x0300A130 - 0x0300A13F size : 16 Config Area 2 0x0300A200 - 0x0300A20F size : 736 Config Area 1 0x27030080 - 0x2703035F size : 736 Verifying data Config Area 1 0x0300A100 - 0x0300A11F size : 32 Config Area 1 0x0300A100 - 0x0300A11F size : 32 size : 16 Config Area 1 0x0300A100 - 0x0300A11F size : 12 size : 16 Config Area 1 0x0300A100 - 0x0300A11F size : 232 size : 16 Config Area 2 0x0300A200 bx0300A205 size : 238 Config Area 4 0x27030080 bx0300A205 size : 238 Config Area 4 0x27030080 bx2703035F size : 736		^
IConfig Area 1] 0x0300A100 - 0x0300A11F size: 32 Config Area 1] 0x0300A100 - 0x0300A13F size: 16 Config Area 2] 0x0300A200 - 0x0300A25F size: 736 Config Area 4] 0x27030080 - 0x2703035F size: 736 Verifying data Config Area 1] 0x0300A100 - 0x0300A11F size: 128 Config Area 1] 0x0300A100 - 0x0300A11F size: 128 size: 128 Config Area 1] 0x0300A100 - 0x0300A11F size: 128 size: 128 Config Area 1] 0x0300A100 - 0x0300A25F size: 128 size: 128 Config Area 4] 0x0300A100 - 0x0300A25F size: 128 size: 128 Config Area 4] 0x27030080 - 0x2703035F size: 736 size: 736 Setting the target device size: 736 size: 736 Disconnecting the tool Operation completed. size: 108		

図 38. HMAC SHA2-256 検証に基づくアプリケーションの更新成功

起動に成功すると、J-Link 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。



ルネサス RA ファミリ RA8 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



図 39. HMAC ブートの J-Link コンソール出力

HMAC ブートの失敗は容易に把握できます。まず図 23 のようにデバイスを初期化し、セクション 2.6 の手順を実行します。ただし、間違ったアプリケーションイメージ、例えば CRC ブートコードのバイナリイメ ージや更新された HMAC ブートコードを指定することを除きます。いずれの場合も、図 38 の RFP プログ ラミングは失敗し、エラーレポートとして P107 ピンが"High"になり、赤色の LED が点灯します。



RA8 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

3. CRC チェックと CRC ブートデモンストレーション製品へプログラミング

CRC ブートを使用する場合、プログラミング中、MCU リセット後の通常実行中に CRC32 整合性チェック が実行されます。

3.1 CRC ブート FSBL を使用した blinky サンプルプロジェクト準備

blinky サンプルプロジェクト blinky_crc_boot_ra8m1 は、図 40 のように FSBL Execution Mode を CRC boot and report measurement に更新することで作成されます。同じくエラーレポートとしてポート P107 ピンが使用されます。さらに、J-Link コンソールメッセージは CRC ブートモードを示すように更新さ れます。

blinky_crc_boot_ra8m1 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 プロジェクト(blinky_crc_boot_ra8m1)をビ ルドして実行し、その機能を確認します。

- コンパイル結果は、セクション 3.2 でコード証明書の作成に使用されます。
- 3つの LED が点滅しているはずです。



図 40. CRC ブート用の FSBL 設定



RA8 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

図 25 のように Tera Term を Setting し、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 の出力はすべて 0xFF 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ので、version num oem_b1 の出力はすべて 0 です。



図 41. CRC ブートの RTT ビューア出力

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 アプリケーションの期待値が CRC が含まれている必要がありま す。MCU リセット時、整合性チェックが実行されている場合、FSBL はアプリケーションの 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。

SKMT CLI を使用した CRC ブート用のコード証明書の作成

コマンドラインを開き、SKMT\cli フォルダに移動し、以下のコマンドを使用して、セクション 3.1 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\SecurityKeyMangementTool\cli>skmt.exe /gencert /mode "CRC" /loadaddr "02000000" /oembl_size "8000" /cfsize "200000" /oembl

"C:\RA8_FSBL\fsp_v510\blinky_crc_boot_ra8m1\Debug\blinky_crc_boot_ra8m1.srec" /output_codecert

"C:\RA8_FSBL\K2\certificates\blinky_crc_boot_ra8m1_code_cert.bin"

Output File: C:\RA8_FSBL\K2\certificates\blinky_crc_boot_ra8m1_code_cert.bin



SKMT GUI を使用した CRC ブート用コード証明書の作成

The First Stage Bootloader supports multiple use cases for checking application code, such as Bootloader, for authenticity and/or integrity prior to programming and prior to executi Please refer to device-specific documentation for complete descriptions of all use case	an OEM on. s.
OEM Bootloader Image : Ø\blinky_crc_boot_ra8m1\Debug\blinky_crc_boot_ra8m1.sree Programming Verification Method : O Signature Image Version : 4	s Browse
Certificates OEM Root Keys OEM Bootloader Keys	
Code Flash Start Address (hex) : 02000000 Device Code Flash Size : 2 MB ~ (If exact size option is not available, select the next	t lower size)
OEM Bootloader Size :	
Kev Certificate : C:\RA8_FSBL\K2\certificates\blinky_hmac_boot_ra8m1_key_cert.bin	Browse
Code Certificate : C:\RA8_FSBL\K2\certificates\blinky_crc_boot_ra8m1_code_cert.bin	Browse
Generate File(s)	

図 42. CRC ブート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。CEC 検証のために作成される測定レポートでは Signer 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、Signer ID(SHA256)は 32 バイトなので、CRC 値は Signer ID フィールドに 8 回書き込まれます。



RA8 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

3.3 RFP を使用したアプリケーションの CRC ブートの実証

RFP を起動し、付属の rfp_ra8m1.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 SWD ブートインタフェースを介してアプリケーションを認証し起動するための認証情報を提供します。 以下の動作設定が構成されていることを確認します。

Command	Erase Options
🗹 Erase	Erase Selected Blocks 🗸
Program	Program & Verify Options
Verify	Erase Before Program
Read Date Online	Verify by reading the device \checkmark
Venily Hash Options	
Checksum	Checksum Type
	CRC-32 method
Fill with 0xFF	
Code Area / User Boot Area	Error Settings
Data Area	Enable address check of program file

図 43. アプリケーションをダウンロードする前の動作設定と

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。

Operation Operation Settin	gs Block Settings Flash	Options Connect Se	ttings Unique Code		
Project Information					
Current Project:	rfp_ra8x1.rpj				
Microcontroller:	R7FA8M1AHECBD				
Program and User Key	Files				
			Add/Rem	ove Files	
📕 File Details				-	
		Г	Add Filo(a)	Pomovo S	elected File/a)
		L	Add File(s)	Helliove Se	elected File(s)
CARAS ESPLAtes (E10)	hlinku oo haat m0m1\Dak	waltinku ere heet i		Type Ad	dress/Offset
C. INNO_FOR VOID		oug tollinky_cic_boot_	auintisiec	ILA	
			_		
			L	ОК	Cancel
Disconnecting the tool Operation completed.					

図 44. アプリケーションプロジェクトの選択



RA8 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、Flash Options ページに移動して Code Certificate を Setting します。

Ope	eration Operation Settings Block Settin	gs Flash Options Connect Settings Unique Code				
	Set Option	Do Nothing	-			
	Disable Initialize Command	No				
	Disable AL2 authentication	No				
	Disable AL1 authentication	No				
	Disable LCK_BOOT transition	No				
~	Configuration Data Lock Bit					
	Set Option	Do Nothing				
	Lock bits	HEX FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF				
×	Anti-Rollback Settings					
	Set Option	Do Nothing				
	ARCLS	HEX FFFF				
	ARCCS	HEX FFFF				
~	OEM Root Public Key					
	Set Option	Do Nothing				
	OEM Root Public Key File 1					
	Disable Rewriting	No				
~	Certificate		-			
	Set Option	Set				
	Verification Method	CRC				

図 45. CRC コード証明書の Setting

次に、Operation タブに移動し、Start 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 が点滅するはずです。



RA8 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



図 46. CRC ブートアプリケーションのダウンロード

J-Link 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 4 回繰り返されていることに注意してください。

The current setting of the FSBL registers: FSBLCTRL0 : fffffe00 FSBLCTRL1 : ffffffd SBCTRL2 : ffffffd SACC0 : 02000000 SACC1 : 02000000 SAMR : 22001000 HOEMRTPK : 22021000 HOEMRTPK : 22022a5a8 CFGDLOCK.CFGD0.CFGD_H : fffffff CFGDLOCK.CFGD1.CFGD_H : fffffff CFGDLOCK.CFGD1.CFGD_H : fffffff CFGDLOCK.CFGD2 : 0000ffff The current output of the boot measurement report: p_report->sha256_oem_bl_fsblctrl1[0:15]: 8e,51,51,70,d8,fb,c8,4e,58,c8,8b,86,d3,11,fc,f4, p_report->sha256_oem_bl_fsblctrl1[16:15]: 8e,51,51,70,d8,fb,c8,4e,58,c8,8b,86,d3,11,fc,f4, p_report->sha256_oem_bl_fsblctrl1[16:31]: ab,93,43,be,cf,b7,ae,38,f7,a8,c5,a6,79,dd,d0,37, p_report->signer_id[0:15]: 54,ba,fc,e1,54,ba,fc,e1,54,ba,fc,e1,		······································	
FSBLCTRL0 : ffffffe00 FSBLCTRL1 : fffffffa FSBLCTRL2 : fffffffa SACC0 : 02006000 SACC1 : 022001000 SAMR : 222201000 HOEMRTPK : 22223538 CFGDLOCK.CFGD0.CFGD_H : fffffff CFGDLOCK.CFGD1.CFGD_H : ffffffff CFGDLOCK.CFGD1.CFGD_H : ffffffff CFGDLOCK.CFGD2 : 0000ffff The current output of the boot measurement report: p_report->sha256_oem_bl_fsblctrl1[0:15]: 8e,51,51,70,48,fb,c8,4e,58,c8,8b,86,d3,11,fc,f4, p_report->sha256_oem_bl_fsblctrl1[11[15]1]; ab,93,43,be,cf,b7,ae,38,f7,a8,c5,a6,79,dd,d0,37, p_report->signer_id[0:15]; 54,ba,fc,e1,54,ba,fc,e1,54,ba,fc,e1,54,ba,fc,e1,54	he current setting of the FSE	L registers:	
FSBLCTRL1 : fffffffd FSBLCTRL2 : fffffffd SACC0 : 02006000 SACC1 : 02000000 SAMR : 222001000 SAMR : 222001000 HOEMRTPK : 2e22a5a8 CFGDLOCK.CFGD0.CFGD_H : fffffff CFGDLOCK.CFGD1.CFGD_H : ffffffff CFGDLOCK.CFGD1.CFGD_H : ffffffff CFGDLOCK.CFGD2 : 0000ffff The current output of the boot measurement report: p_report->sha256_oem_b1_fsblctrl1[0:15]: 8e,51,51,70,48,fb,c8,4e,58,c8,8b,86,43,11,fc,f4, p_report->sha256_oem_b1_fsblctrl1[16:15]: ab,93,43,be,cf,b7,ae,38,f7,a8,c5,a6,79,dd,d0,37, p_report->signer_id[0:15]: 54,ba,fc,e1,54,	SBLCTRLØ : fffffe	00	
FSBLC1RL2 : fffffffff SACC0 : 02006000 SACC1 : 02006000 SAMR : 22001000 SAMR : 920020 SAMR : 10000 SAMR : 90000ffff The current output of the boot measurement report: p_report->sha256_oem_bl_fsblctrl1[0:151: 8e,51,51,70,d8,fb,c8,4e,58,c8,4b,86,d3,11,fc,f4, n_penort->sha256_oem_bl_fsblctrl1[16:31]: ab	BLCTRL1 : ffffff	fd	
SACC0 : 02000000 SAMR : 22000000 HOEMRTPK : 2202a5a8 CFGDLOCK.CFGD0.CFGD_H : fffffff CFGDLOCK.CFGD1.CFGD_H : fffffff CFGDLOCK.CFGD1.CFGD_H : ffffffff CFGDLOCK.CFGD1.CFGD_H : ffffffff CFGDLOCK.CFGD2 : 0000ffff The current output of the boot measurement report: p_report->sha256_oem_bl_fsblctrl1[0:15]: 8e,51,51,70,d8,fb,c8,4e,58,c8,8b,86,d3,11,fc,f4, p_report->sha256_oem_bl_fsblctrl1[f6:31]: ab,93,43,be,cf,b7,ae,38,f7,a8,c5,a6,79,dd,d0,37, p_report->signer_id[0:15]: 54,ba,fc,e1,54,ba,fc,e1,54,ba,fc,e1,	SBLCIRL2 : ffffff	fa	
SAMR : 22001000 HOEMNTPK : 2223a8 CFGDLOCK.CFGD0.CFGD H : fffffff CFGDLOCK.CFGD1.CFGD H : fffffff CFGDLOCK.CFGD1.CFGD H : fffffff CFGDLOCK.CFGD1.CFGD H : fffffff CFGDLOCK.CFGD2 : 0000ffff The current output of the boot measurement report: $p_report->sha256_oem_bl_fsblctrl1[0:15]:$ 8e,51,51,70,48,fb,c8,4e,58,c8,8b,86,d3,11,fc,f4, $p_report->sha256_oem_bl_fsblctrl1[16:31]:$ ab,93,43,be,cf,b7,ae,38,f7,a8,c5,a6,79,dd,d0,37, $p_report->signer_id[0:15]:$ 54,ba,fc,e1,54,ba,fc,e1,54,ba,fc,e1,	ACC1 - 020000	100 100	
HOEMRIPK : 2222a5a8 CFGDLOCK.CFGD0.CFGD_H : fffffff CFGDLOCK.CFGD1.CFGD_H : fffffff CFGDLOCK.CFGD1.CFGD_H : fffffff CFGDLOCK.CFGD2 : 0000ffff The current output of the boot measurement report: p_report->sha256_oem_b1_fsblctrl1[0:15]: 8e,51,51,70,48,fb,c8,4e,58,c8,8b,86,d3,11,fc,f4, p_moprt->sha256_oem_b1_fsblctrl1[16;31]: ab.93,43,be,cf,b7,ae,38,f7,a8,c5,a6,79,dd,d0,37, p_report->signer_id[0:15]: 54,ba,fc,e1,54,ba,fc,e1,54,ba,fc,e1,	AMR = 220010	100	
CFGDLOCK.CFGDB.CFGD_H : fffffff CFGDLOCK.CFGDB.CFGD_H : fffffff CFGDLOCK.CFGD1.CFGD_H : fffffff CFGDLOCK.CFGD2 : 0000ffff The current output of the boot measurement report: p_report->sha256_oem_b1_fsblctr11[0:15]: 8e.51.51.70.d8.fb.c8.4e.58.c8.8b.86.d3.11.fc.f4. p_report->sha256_oem_b1_fsblctr11[16:31]: ab.93.43.be.cf.b7.ae.38.f7.a8.c5.a6.79.dd.d0.37. p_report->signer_id[0:15]: 54.ba.fc.e1.54.ba.fc.e1.54.ba.fc.e1.	DEMRTPK : 2e22a5	a8	
CFGDLOCK.CFGD1.CFGD_H : fffffff CFGDLOCK.CFGD2 : 0000ffff The current output of the boot measurement report: p_report->sha256_oem_b1_fsblctr11[0:15]: 8e.51,51.70,d8,fb,c8,4e.58,c8,8b,86,d3,11,fc,f4, p_report->sha256_oem_b1_fsblctr11[16:31]: ab.93,43,be,cf,b7,ae,38,f7,a8,c5,a6,79,dd,d0,37, p_report->signer_id[0:15]: 54,ba,fc,e1,54,ba,fc,e1,54,ba,fc,e1,	FGDLOCK.CFGD0.CFGD_H : ffffff Replock cRep0 cRep H · ffffff		
CFGDLOCK.CFGD1.CFGD_H : fffffff CFGDLOCK.CFGD2 : 0000ffff The current output of the boot measurement report: p_report->sha256_oem_b1_fsblctr11[0:15]: 8e,51,51,70,d8,fb,c8,4e,58,c8,8b,86,d3,11,fc,f4, m_report=>sha256_oem_b1_fsblctr11[6:31]: ab,93,43,be,cf,b7,ae,38,f7,a8,c5,a6,79,dd,d0,37, p_report->signer_id[0:15]: 54,ba,fc,e1,54,ba,fc,e1,54,ba,fc,e1,54,ba,fc,e1,	FGDLOCK.CFGD1.CFGD H : ffffff	ff	
CFGDLOCK.CFGD2 : 0000ffff The current output of the boot measurement report: p_report->sha256_oem_bl_fsblctrl1[0:15]: 8e,51,51,70,d8,fb,c8,4e,58,c8,8b,86,d3,11,fc,f4, n report->sha256_oem_bl_fsblctrl1[16:31]: ab,93,43,be,cf,b7,ae,38,f7,a8,c5,a6,79,dd,d0,37, p_report->signer_id[0:15]: 54,ba,fc,e1,54,ba,fc,e1,54,ba,fc,e1,54,ba,fc,e1,	FGDLOCK.CFGD1.CFGD_H : ffffff	ff	
The current output of the boot measurement report: p_report->sha256_oem_bl_fsblctrl1[0:15]: 8e,51,51,70,d8,fb,c8,4e,58,c8,8b,86,d3,11,fc,f4, m_report->sha256_oem_bl_fsblctrl1[f6:31]: ab,93,43,be,cf,b7,ae,38,f7,a8,c5,a6,79,dd,d0,37, p_report->signer_id[0:15]: 54,ba,fc,e1,54,ba,fc,e1,54,ba,fc,e1,	FGDLOCK.CFGD2 : 0000ff	ff	
p_report->sha256_oem_bl_fsblctrl1[0:15]: 8e,51,51,70,d8,fb,c8,4e,58,c8,8b,86,d3,11,fc,f4, n_report->sha256_oem_bl_fsblctrl1[16:31]: ab,93,43,be,cf,b7,ae,38,f7,a8,c5,a6,79,dd,d0,37, p_report->signer_id[0:15]: 54,ba,fc,e1,54,ba,fc,e1,54,ba,fc,e1,54,ba,fc,e1,	he current output of the boot	measurement report	:
8e,51,51,70,d8,fb,c8,4e,58,c8,8b,86,d3,11,fc,f4, n report=>sha256 oem hl fshlctr11[16:31]: ab,93,43,be,cf,b7,ae,38,f7,a8,c5,a6,79,dd,d0,37, p_report=>signer_id[0:15]: 54,ba,fc,e1,54,ba,fc,e1,54,ba,fc,e1,54,ba,fc,e1,	_report->sha256_oem_b1_fsblct	rl1[0:15]:	
ab,93,43,be,cf,b7,ae,38,f7,a8,c5,a6,79,dd,d0,37, p_report->signer_id[0:15]: 54,ba,fc,e1,54,ba,fc,e1,54,ba,fc,e1,54,ba,fc,e1,	8,51,51,70,d8,fD,c8,48,58,c8, Nevent-Scha256 com bl foblot	8D,86,d3,11,fC,f4,	
p_report->signer_id[0:15]: 54,ba,fc,e1,54,ba,fc,e1,54,ba,fc,e1,54,ba,fc,e1,	0.93.43.be.cf.b7.ae.38.f7.a8.	c5.a6.79.dd.d0.37.	
54, ba, fc, e1, 54, ba, fc, e1, 54, ba, fc, e1, 54, ba, fc, e1,	report->signer_id[0:15]:		
	4, ba, fc, e1, 54, ba, fc, e1, 54, ba,	fc,e1,54,ba,fc,e1,	
p_report=/signer_inite;si;; 54. ha_fc_e1.54. ha_fc_e1.54. ha_fc_e1.54. ha_fc_e1.	_report=/signer_10[16:31]; 4.ba.fc.e1.54.ba.fc.e1.54.ba.	fc_e1_54_ba_fc_e1_	
pp report - version_num_oem_or	report-/version_num_vem_bi.	10,01,01,04,10,01,	
0000	000		
The CRC value of the annlication image:	he CRC value of the applicati	on image:	

図 47. CRC ブートの成功



RA8 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

RFP から計算された CRC32 は、SKMT から計算された CRC32 とは異なることに注意してください。RFP はファイル全体の CRC32 を計算します。例えば、.srec ファイルの場合、すべてのレコードフィールドが CRC32 の計算に含まれます。しかし、SKMT の場合、CRC32 の計算には OEM_BL のバイナリ値のみが含まれます。

CRC ブートの失敗は容易に把握できます。図 23 のようにデバイスを初期化し、セクション 3.3 の手順を実行します。ただし、間違ったバイナリイメージを提供する場合は除きます。例えば、HMAC ブートコードの バイナリイメージを提供するか、CRC ブートコードのマイナーアップデートを実行します。いずれの場合 も、図 46 の RFP プログラミングは失敗し、ポート P107 ピンは"High"になり、赤色の LED が点灯します。



RA8 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. 付録

4.1 TrustZone[®]プロジェクトでの使用上の注意点

Trustzone[®]プロジェクトのセットが以下のとおりであると仮定します。:

- セキュアアプリケーション用 blinky s
- 一致する非セキュアアプリケーション用 blinky ns

FSBL を HMAC ブートの TrustZone[®]プロジェクトのセットで使用する場合、ECC キーペアと OEM ルート キーの生成は、セクション 2.1、2.2 と同じ方法で実行できます。セクション 2.4 で説明された同じスクリ プトを使用して、ARC_OEMBL レジスタの値を取得することもできます。

セクション 2.3、2.5、2.6の操作は、TrustZone[®]プロジェクトで FSBL を使用するように調整する必要があ ります。IDE から生成された.rdp ファイルには、セキュアプロジェクトのフラッシュと SRAM の使用サイド に基づいた TrustZon[®]境界が含まれており、TrustZone[®]境界の Setting に使用されます。

 セクション 2.3 では、FSBL を blinky_s プロジェクトで有効にする必要があります。SACCx が blinky_s と blinky_ns が使用するフラッシュ領域の外側にあることを確認してください。e² studio を使用する場合、以下の選択が有効になっていると、IDE は生成された blinky_s.rdp ファ イルを取得し TrustZone[®]境界をプログラムします。

me: blinky_hmac_boot_ra8m1 Debug_Flat	Source
Debug hardware: J-Link ARM V Target Device: I	R7FA8M1AH
GDB Settings Connection Settings Debug Tool Settin	ngs
✓ J-Link	
Туре	USB
J-Link Serial	(Auto)
Settings File	\${workspace_loc:/\${ProjName}}/\${LaunchConfigName}.jlink
Script File	
Log File	<pre>\${workspace_loc:/\${ProjName}}/JLinkLog.log</pre>
Low Power Handling	No
✓ IP Connection	
Connection Method	IP via LAN
Host Name/IP Address[:port number]	
Identifier	
Tunnel Server	
Port Number	
Password	
✓ Interface	
Туре	SWD
Speed (kHz)	4000
✓ JTAG Scan Chain	
Multiple Devices	No
IRPre	0
DRPre	0
✓ Connection	
Register initialization	No
Reset at the beginning of connection	Yes
Reset at the end of connection	No
Reset before download	Yes
Reset after download	Yes
ID Code (Bytes)	FFFFFFFFFFFFFFFFFFFFFFFFF
Hold reset during connect	Yes
Set CPSR(5bit) after download	No
Prevent Releasing the Reset of the CM3 Core	Yes
Secure Vector Address	
Non-secure Vector Address	
Hot Plug	No
Disconnection Mode	Stop
✓ SWV	
Core clock (MHz)	0
✓ TrustZone	
Set TrustZone secure/non-secure boundaries	Yes
Authenticate device to Authentication Level (AL)	None
Authentication key	

図 48. e² studio で TrustZone[®]境界プログラミングの有効化



RA8 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

TrustZone[®]ベースのアプリケーション開発に Keil と IAR を使用する場合は、RDPM を手動で起動する必要 があります。図 45 のように、RDPM は生成された.rdp ファイルを使用して TrustZone[®]境界をプログラムす るように Setting します。

Set TrustZone secure / no	mation n-secure bounda	Chan	ge debug state ize device back f	to factory de	e <mark>fault</mark>
Target MCU connection:		J-Link	~		
Connection Type:		SCI	~		
Emulator Connection:		Serial No	~		
Serial No/IP Address:					
Debugger supply voltage (V):		0	~		
Connection Speed (bps for S	CI, Hz for SWD):	9600	~		
Debug state to change to:		Secure Softwa	are Development	t v	
Memory partition sizes					
Use Renesas Partition Dat	a file				
C:\RA8_FSBL\K2\blinky_s\D	ebug\blinky_s.rp	bd			Browse
Code Flash Secure (KB)	32				
Code Flash NSC (KB)	0				
Data Flash Secure (KB)	0				
SRAM Secure (KB)	8				
SRAM NSC (KB)	0				
Command line tool: C:\Users\a5099044\.eclipse\c Connecting Loading library Establishing connection Checking the device's Tru	om.renesas.platf : SUCCESSFUI : SUC ustZone type	orm_86576045 L! CESSFUL! SUCCESSFUL!	0\DebugComp\	.RA\DeviceP	ar Brows
Programming secure/non-se - Code Flash Secure (- Data Flash Secure (SUCCESSFUL!	ecure memory pa kB) : 32 kB) : 0	ntitions with th	ne following sett	tings	
Disconnecting DISCONNECTED.	ЛТ				

図 49. RDPM を使用した TrustZone[®]境界の Setting



RA8 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

- セクション 2.5 では、キー証明書とコード証明書の生成に blinky_s のコンパイル済みバイナリを使用 します。非セキュアアプリケーションは、証明書の生成には使用されません。
- セクション 2.6 で、セキュアアプリケーションと非セキュアアプリケーションをプログラムする必要が あります。また TrustZone[®]境界は、セキュアプロジェクトから生成された blinky_s.rdp ファイルを 用いて Setting します。従って、図 36 のようにアプリケーションを提供することだけでなく、セクショ ン2.6 で記載されているように他のすべての Setting に加えて、図 50 のように RFP フラッシュオプシ ョンで blinky s.rdp ファイルを選択して TrustZone[®]境界を Setting する必要があります。

Op	eration	Operation Settings	Block Settings	Flash Options	Connect Settings	Unique Code		
~	DLM	Keys						^
	Set Op	ption		Do	Nothing			
	AL2 K	(ey File						
	AL1 K	(ey File						
	RMA	Key File						
~	Boun	idary						
	Set Op	ption		Se	st i i i i i i i i i i i i i i i i i i i			
	Use R	Renesas Partition Data	File	Ye	s			
	Renes	sas Partition Data File		R8	8 FSBL\K2\K2\Ł	olinky s\Debug	blinky s.rod	

図 50. RFP を使用した TrustZone[®]境界の Setting

FSBLをCRC ブートの TrustZone[®]プロジェクトのセットで使用する場合、コード証明書のみが生成される ことを除き、HMAC ブートと同様の方法でセクション 3.1、3.2、3.3の操作を調整する必要があります。

TrustZone[®]プロジェクトの一般的な使用方法については、アプリケーションプロジェクトの、RA8 MCU クイックデザインガイド R01AN7087、TrustZone[®]を使用したセキュリティデザイン R11AN01467 を参照してください。



RA8 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 RFP 使用時のエラーのデバッグ

RFP 操作から出力されるエラーコードを理解しておくには、RFP ユーザーズマニュアルを参照してくださ い。ドキュメント情報は参考資料のセクションに記載されています。参考までに、以下のエラーコードは、 イメージバージョン番号が正しく Setting されていないことを示しています。

> Writing data to the target device [Flash Options] Option Information : Code Certificate Setting the target device Disconnecting the tool Error(E1000016): A security error occurred in the device. (Command: 26, Response: DC) Operation failed.

> > 図 51. イメージバージョン番号のエラー



RA8 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

5. References

- 1. Rebesas RA Family Injecting and Updating Secure User Keys (R11AN0496)
- 2. RA8M1 グループ ユーザーズ マニュアル ハードウェア編(R01UH0994)
- 3. Renesas Secure Key Management Tool ユーザーズマニュアル(R20UT5349)
- 4. Renesas Flash Programmer Flash memory programming software ユーザーズマニュアル(R20UT5352)
- 5. Renesas RA Security Design using TrustZone with IP Protection (R11AN0467)
- 6. Renesas RA8 MCU Quick Design Guide (R01AN7087)



RA8 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

6. ウェブサイトとサポート

RA 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-RA8M1 リソース RA 製品情報 フレキシブルソフトウェアパッケージ(FSP) RA 製品サポートフォーラム ルネサスサポート renesas.com/ra/ek-ra8m1 renesas.com/ra renesas.com/ra/fsp renesas.com/ra/forum renesas.com/support



RA8 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

改訂履歴

		説明					
Rev.	日付	ページ	概要				
1.00	2024年11月13	-	初版バージョン				



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、入力信号や入出力プルアップ電源を入れないでください。入力信号や入出力プルアップ電源からの電流インジ ェクションにより、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_L (Max.)から V_H (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_L (Max.)から V_H (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

- 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。
- 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。
 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。
- 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も、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、これに限りません。)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、脆弱性問題に起因し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を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といたします。
- 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 2024.10)

本社

豊洲フォレシア、豊洲 3-2-24、 135-0061 東京都江東区豊洲 3-2-24 豊洲フォレシア TOYOSU FORESIA, 3-2-24

www.renesas.com

商標

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。

お問い合わせ先

製品、技術、ドキュメントの最新版、最寄りの営業所などに関する詳 しい情報は、こちらをご覧ください。 www.renesas.com/contact/