RX ファミリ

TSIP ドライバを用いた AES 暗号の利用方法

要旨

Trusted Secure IP(TSIP)ドライバは、多くの暗号アルゴリズムを提供しています。本アプリケーションノートでは、RX ファミリ TSIP ドライバを用いて AES 暗号を実装する方法を説明します。本アプリケーションノートには、RX FIT の BSP と TSIP ドライバを使ったサンプルプロジェクトを添付しています。サンプルプロジェクトでは、TSIP ドライバを使用して ECB および CBC モードで AES 暗号化および復号を行います。

サンプルプロジェクトには、サンプルプログラムをすぐに実行できるように、あらかじめラップされた UFPK が含まれています。この UFPK は実製品では利用することができません。お客様の実製品では、お客様が作成した UFPK を Key Wrap サービスでラップして使用する必要があります。

目的

本アプリケーションノートでは、TSIP ドライバを使用して ECB および CBC モードで AES 128 暗号化を行うサンプルを提供します。TSIP やその他の周辺機能を追加・構成する際には、スマート・コンフィグレータを使用します。提供されるサンプルプロジェクトは Renesas Starter Kit+ for RX65N （RX65N RSK）および RX72N Envision Kit 用のプロジェクトです。スマート・コンフィグレータを使用することにより、他の TSIP を搭載した RX ファミリマイクロコントローラへの移植が可能です。

動作確認デバイス

本アプリケーションノートに付属するサンプルプログラムは以下のデバイスで動作確認しています。

RX65N：R5F565NE

RX72N：R5F572NN

動作環境

本アプリケーションノートに付属するサンプルプログラムは以下の環境で動作確認しています。統合開発環境とツールチェーンは、アプリケーションノート「RX ファミリ TSIP(Trusted Secure IP)モジュール Firmware Integration Technology (R20AN0371)」に記載のバージョンを使用しています。

統合開発環境	e^2 studio
ツールチェーン	CCRX コンパイラ
ターゲットボード	Renesas Starter Kit+ for RX65N-2MB（製品型名：RTK50565N2CxxxxxBE） RX72N Envision Kit（製品型名：RTK5RX72N0CxxxxxBJ）
デバッガ	E2 Lite エミュレータ
TSIP ドライバ	バージョン 1.21
Tera-Term	バージョン 4.104

目次

1. 概要 3
1.1 AES アルゴリズム 3
1.2 ブロック暗号のオペレーションモード 3

2. TSIP の概要 5
2.1 用語の定義 5
2.2 Trusted Secure IP（TSIP） 6
2.2.1 Trusted Secure IP ドライバパッケージ 7
2.2.2 TSIP ドライバのインストール 8

3. AES 暗号プロジェクトの作成方法 9
3.1 TSIP を使用するプロジェクトの作成 9
3.2 TSIP FIT モジュールの追加 10
3.3 BSP 構成 11
3.4 AES を使用するための TSIP ドライバ設定 11
3.5 鍵のラップと Wrapped Key 12

4. デモプロジェクト 13
4.1 デモコマンド 13
4.2 ファイルおよびフォルダの一覧 13
4.3 AES 実装用 TSIP ドライバ API 関数の一覧 14

5. プロジェクトのビルドおよび実行 15
5.1 プロジェクトのインポートおよびビルド 15
5.2 ハードウェアの設定 17
5.2.1 ターミナルソフトウェア通信 17
5.2.2 プログラムのダウンロード 19
5.3 プログラムの実行 21

6. Security Key Management Tool の使用方法 23

改訂記録 25

1. 概要

1.1 AES アルゴリズム

AES アルゴリズムは、情報を暗号化および復号できる対称ブロック暗号です。暗号化はデータを解読不可能な形式（暗号文）に変換し、復号はデータを元の形式（平文）に逆変換します。当初、NIST は、DES*1 アルゴリズムに代わる新しい暗号アルゴリズムとして、AES アルゴリズムを発表しました。

Feistel および SPN*2 構造は、ブロック暗号アルゴリズムの代表的な構造です。AES アルゴリズムには SPN 構造が採用されました。SPN 構造では、置換と転置が繰り返し適用されます（ラウンド処理）。通常、SPN 構造によるランダム化は、Feistel 構造よりも優れた効率性を発揮します。つまり、必要な処理ラウンド数が少なく、全体的な処理が速くなります。

【注】 *1 DES：The Data Encryption Standard（DES）アルゴリズムは、暗号鍵を使用する対称ブロック暗号です。この標準は、1977 年にアメリカの国家暗号規格として採用されました。
*2 SPN：Substitution Permutation Network（SPN）は、AES アルゴリズムに採用されているブロック暗号アーキテクチャの一種です。

AES は、アメリカ国立標準技術研究所（NIST）によって 2001 年 11 月に公開されました。現在、AES には 3 種類のバージョンがあります。そのすべてにおいてブロックの長さは 128 ビットで統一されており、使用可能な鍵の長さは、128、192、または 256 ビットです。使用する鍵の長さが長くなるほど、アルゴリズムを解読して暗号化データを取得することが難しくなります。本アプリケーションノートでは、128 ビットの鍵長のみ説明します。

AES アルゴリズムでは、128、192、または 256 ビットの暗号鍵を使用して 128 ビットの暗号文が生成されます。各ラウンドではラウンドキーと呼ばれる別の鍵が使用されます。この鍵は、所定の 128、192、または 256 ビットの暗号鍵から生成されます。

1.2 ブロック暗号のオペレーションモード

NIST SP 800-38A には、平文ブロックを暗号化することで、暗号文ブロックを生成するために、数種類のブロック暗号モードが含まれます。ここで紹介するデモプロジェクトの AES 暗号では、ECB モードと CBC モードをサポートします。

- ECB（Electronic Code Book）モード：暗号文は、暗号鍵を使用して平文の各ブロックを暗号化するだけで作成され、対応する暗号文ブロックを形成します。

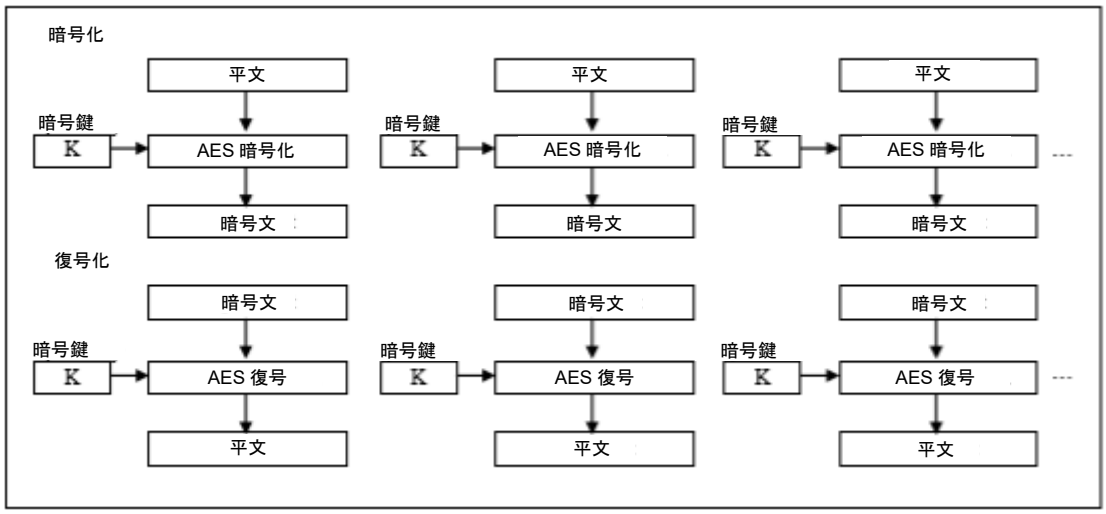


図 1.1　Electronic Code Book（ECB）

- CBC（Cipher Block Chaining）モード：暗号化対象の平文の各ブロックは、前の暗号文のブロックとの XOR を取ります。その結果、暗号鍵を使用して暗号化され、対応する暗号文ブロックが作成されます。最初のブロックには暗号文の前のブロックがないため、前のブロックの代わりに初期化ベクトル（IV）が使用されます。

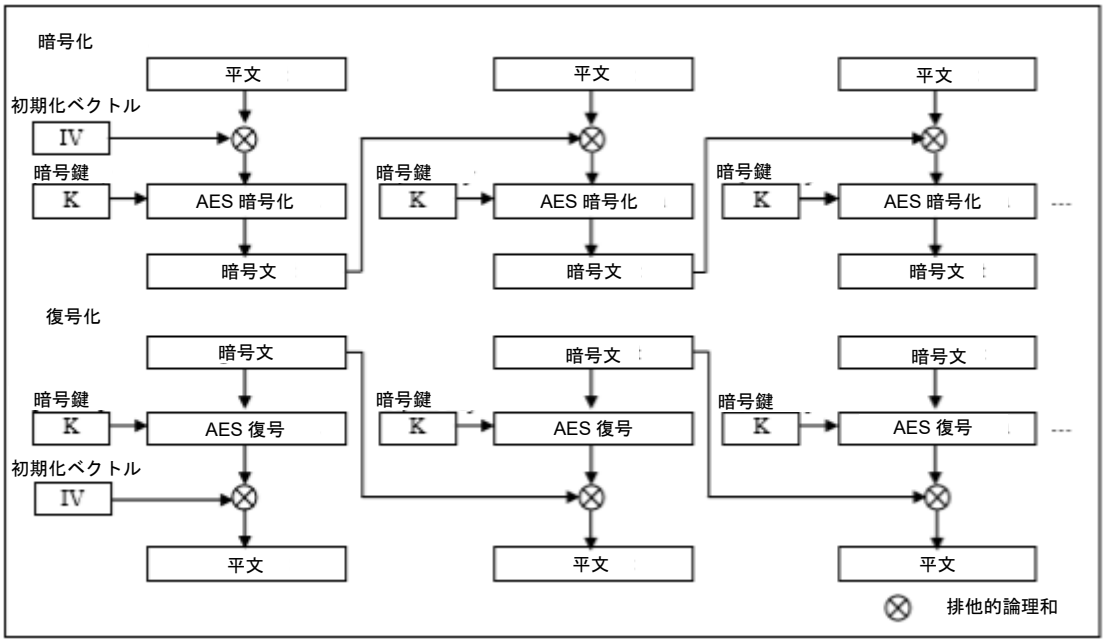


図 1.2　Cipher Block Chaining（CBC）

2. TSIP の概要

2.1 用語の定義

本文書で使用される用語の定義を以下に示します。

表 2-1　用語

用語	説明
ユーザ鍵、User Key	ユーザがデバイス上で暗号機能への入力として使う鍵。ユーザが生成する。
Encrypted Key	UFPK を使用して User Key を AES128 で暗号化、MAC 付与することで生成される鍵情報。Security Key Management Tool が生成する。
Wrapped Key	User Key などを TSIP ドライバで使用可能な形式に変換したデータ。 TSIP が生成する。
UFPK	User Key から Encrypted Key を生成するために必要な鍵。ユーザが生成する。
Hardware Root Key（HRK）	TSIP 内部およびルネサスのセキュアルームのみに存在する鍵。
Key Wrap サービス （https://dlm.renesas.com/）	Renesas 鍵管理サーバ。UFPK をラップするのに使用する。

2.2　Trusted Secure IP（TSIP）

RX ファミリ内の Trusted Secure IP（TSIP）ブロックは、不正アクセスを監視することで、MCU 内部に安全な領域を作成します。これにより、TSIP は暗号化エンジンおよび暗号鍵（User Key）を確実に安全に使用できるようにしています。信頼できる安全な暗号処理において最も重要な要素である暗号鍵（User Key）は、安全で解読不可能な形式でフラッシュメモリ内に保存する必要があります。TSIP は、TSIP ブロックの外部において、暗号鍵（User Key）を安全で解読不可能な Wrapped Key と呼ばれる形式で扱います。

各 TSIP ブロックには安全領域があり、鍵の暗号化に使用される、暗号化エンジン、平文鍵用のストレージおよび Hardware Root Key が格納されています。

TSIP ハードウェアは、TSIP 内部で Wrapped Key から暗号演算に使用する暗号鍵（User Key）を復元します。Wrapped Key は、Unique ID に紐付けられている鍵を使って生成されているため、デバイス固有の値です。したがって、あるデバイスの Wrapped Key を別のデバイスにコピーした場合、TSIP はその Wrapped Key を使用して動作することができません。TSIP ドライバソフトウェアを使用すると、アプリケーションから TSIP ハードウェアにアクセスできます。

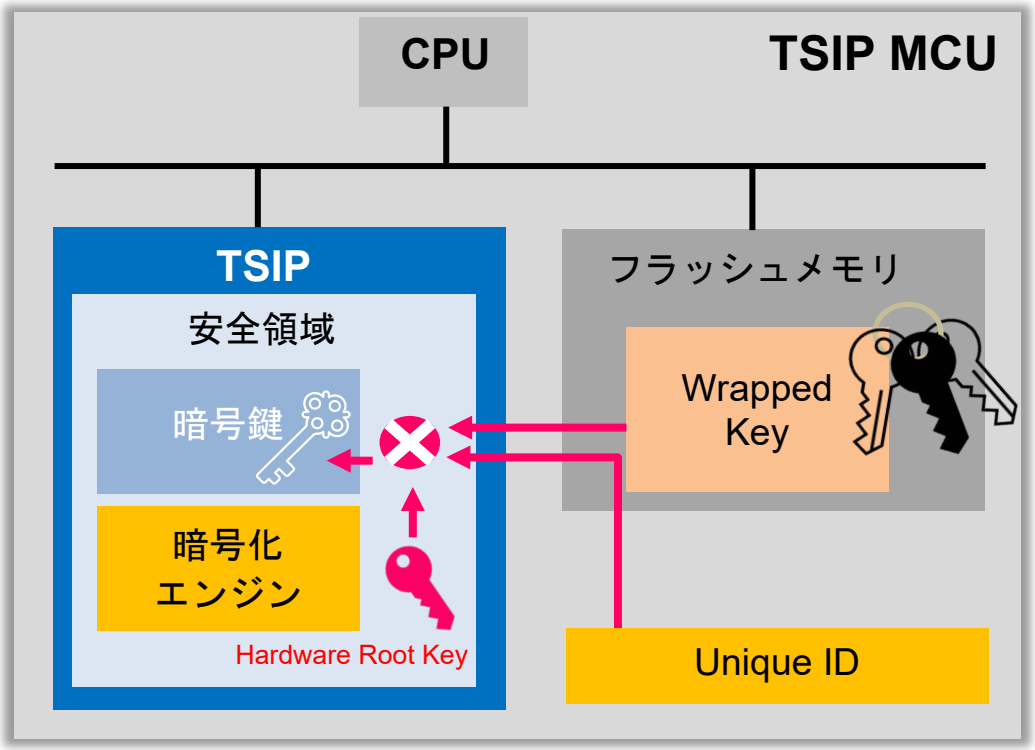


図 2.1　TSIP ハードウェア

2.2.1　Trusted Secure IP ドライバパッケージ

TSIP ドライバパッケージは、ルネサスのウェブサイトからダウンロードすることができます。このパッケージには、TSIP または TSIP Lite をサポートするすべての RX ファミリ用 TSIP ドライバ、アプリケーションノート「RX ファミリ TSIP(Trusted Secure IP) Module Firmware Integration Technology (R20AN0371)」、サンプルコードが含まれます。Trusted Secure IP ドライバパッケージのフォルダ構造については、上記アプリケーションノートの「1.3 製品構成」に記載しています。

2.2.2　TSIP ドライバのインストール

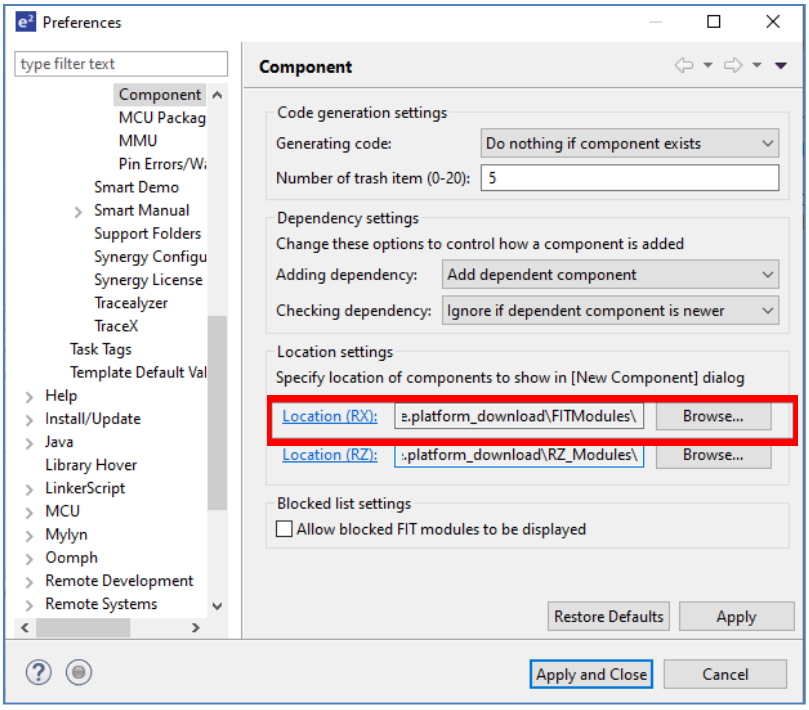
TSIP ドライバは、FIT モジュールとして提供されます。TSIP FIT モジュールは RX Driver Package に同梱されているので、e² studio から RX Driver Package をダウンロードしてください。

RX Driver Package をダウンロードしていない場合は、スマート・コンフィグレータを使用する前に、IDE の FIT モジュールディレクトリに TSIP FIT モジュールを入れておく必要があります。 TSIP FIT モジュールは、TSIP ドライバパッケージの FITModules フォルダに格納されています。TSIP ドライバパッケージを解凍して、その中の FITModules フォルダを FIT モジュールインストールディレクトリにコピーしてください。

以下の 3 つのファイルは、TSIP ドライバパッケージから FIT モジュールインストールフォルダにコピーしてください。

1. r_tsip_rx_v1.xx.xml
2. r_tsip_rx_v1.xx.zip
3. r_tsip_rx_v1.xx_extend.mdf

FIT モジュールインストールフォルダの場所は、スマート・コンフィグレータウィザード内の [Component]（コンポーネント）タブ（下の図を参照）に表示されます。



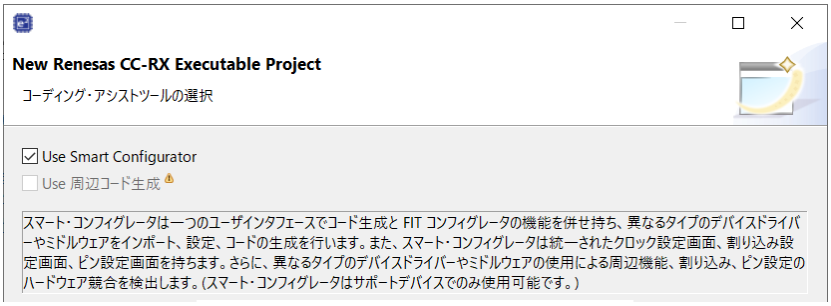
3.　AES 暗号プロジェクトの作成方法

以下の画面は e² studio 2024-10 を使用した画面ならびに手順です。e² studio のバージョンによっては、画面や手順が異なります。

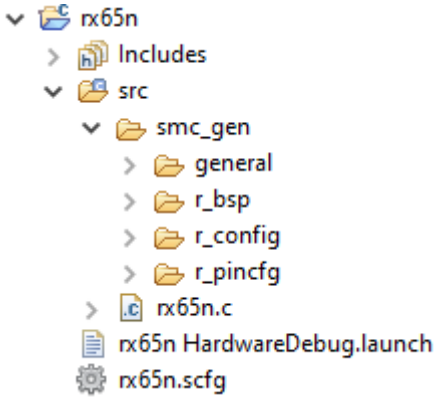
3.1　TSIP を使用するプロジェクトの作成

TSIP を使用するプロジェクトを簡単に作成する方法は、e² studio プロジェクトウィザードを使用して、次にスマート・コンフィグレータで必要な周辺機能を追加する方法です。

- e² studio で[ファイル（F)]→[新規(N) Alt+Shift+N]→[C/C++ project]へ移動します。
- [Renesas RX]→[Renesas CC-RX C/C++ Executable Project]を選択します。
- プロジェクト名とディレクトリを入力します。
- コンパイラと対象デバイス（RX65N RSK の場合：R5F565NEHDFC）を選択します。選択したデバイスに TSIP を使用するためのセキュリティ機能があることを確認します。例えば、RX65N ファミリの R5F565NEHDFC はハードウェア暗号機能を搭載しています。
- [Use Smart Configurator]にチェックマークを付けて、FIT モジュールドライバの追加を許可するコードジェネレータ機能を使用します。



- [終了(F)]をクリックしてプロジェクトの作成を完了します。サンプルプロジェクトは、スマート・コンフィグレータ用のファイル（拡張子*.scfg）です。

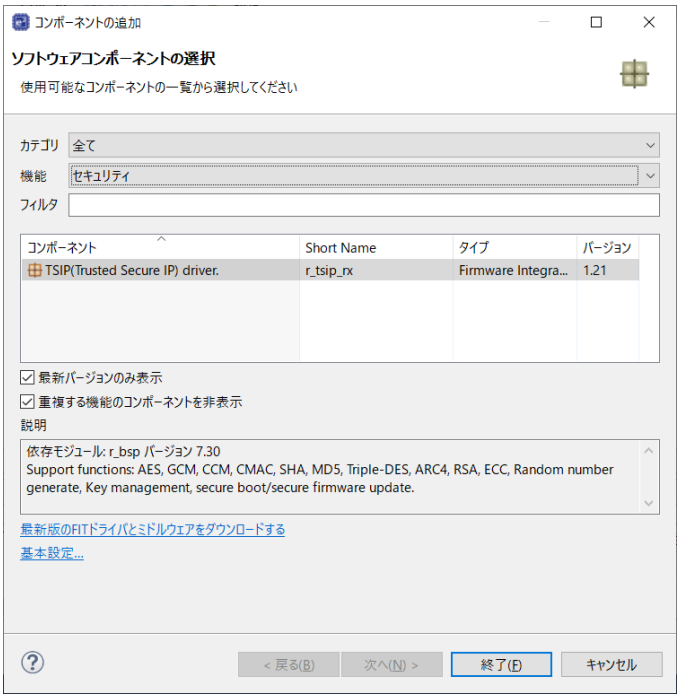


3.2　TSIP FIT モジュールの追加

プロジェクトウィザードで作成できるのは、基本プロジェクトと BSP のみです。TSIP ドライバとアプリケーションコードに必要なその他のペリフェラルドライバは、スマート・コンフィグレータを使用して追加できます。

TSIP ドライバは、データフラッシュ内に Wrapped Key を保存するためのフラッシュ FIT モジュール（r_flash_rx）に依存するので、このドライバを TSIP ドライバと一緒に追加する必要があります。フラッシュ FIT モジュールには、コードやデータフラッシュをプログラム、消去、書き込み、ブランクチェックするための API が含まれます。アプリケーションプロジェクトに必要なその他の FIT モジュールは、スマート・コンフィグレータを使用して追加することもできます。

スマート・コンフィグレータを起動するには、[プロジェクト・エクスプローラー]ウィンドウでプロジェクト内の*.scfg ファイルをダブルクリックします。[コンポーネント]タブを開いて[コンポーネントの追加]ボタンをクリックすると、FIT モジュールを追加できます。コンポーネントは、アプリケーションに必要な関数別に絞り込むことができます。下の例では、「セキュリティ（ドライバ）」別に関数を絞り込むことで、TSIP ドライバ（r_tsip_rx）を追加しています。同じプロセスを使用して、フラッシュ FIT モジュールを追加できます。



デモコードは、UART インタフェースを使用してターミナルを介して通信します。したがって、SCI FIT モジュール（r_sci_rx）および ByteQ FIT モジュール（SCI ドライバに必要）もプロジェクトに追加しています。ByteQ FIT モジュールは、ユーザアプリケーションによって提供されるバッファのインデックス処理を行うために、Open()の呼び出しでキュー制御ブロックを割り当てます。また、このモジュールには、バッファキューからデータを配置・取得するための関数、使用可能または使用中のバイト数を検査するための関数、キューをフラッシュするための関数も含まれます。サポートされるキューの数に制限はなく、キュー制御ブロックはビルド時に静的に割り当てるか、実行時に動的に割り当てることができます。RX65N RSK では送受信に SCI チャネル 8、RX72N Envision Kit では SCI チャネル 2 を使用します。

3.3　BSP 構成

BSP のデフォルト設定は、セキュリティ機能が無効になっています。TSIP を使用するには e² studio プロジェクトウィザードによって生成される BSP 設定ファイルで有効にする必要があります。セキュリティ設定は、BSP の Config ファイル「r_bsp_config_h」にあります。

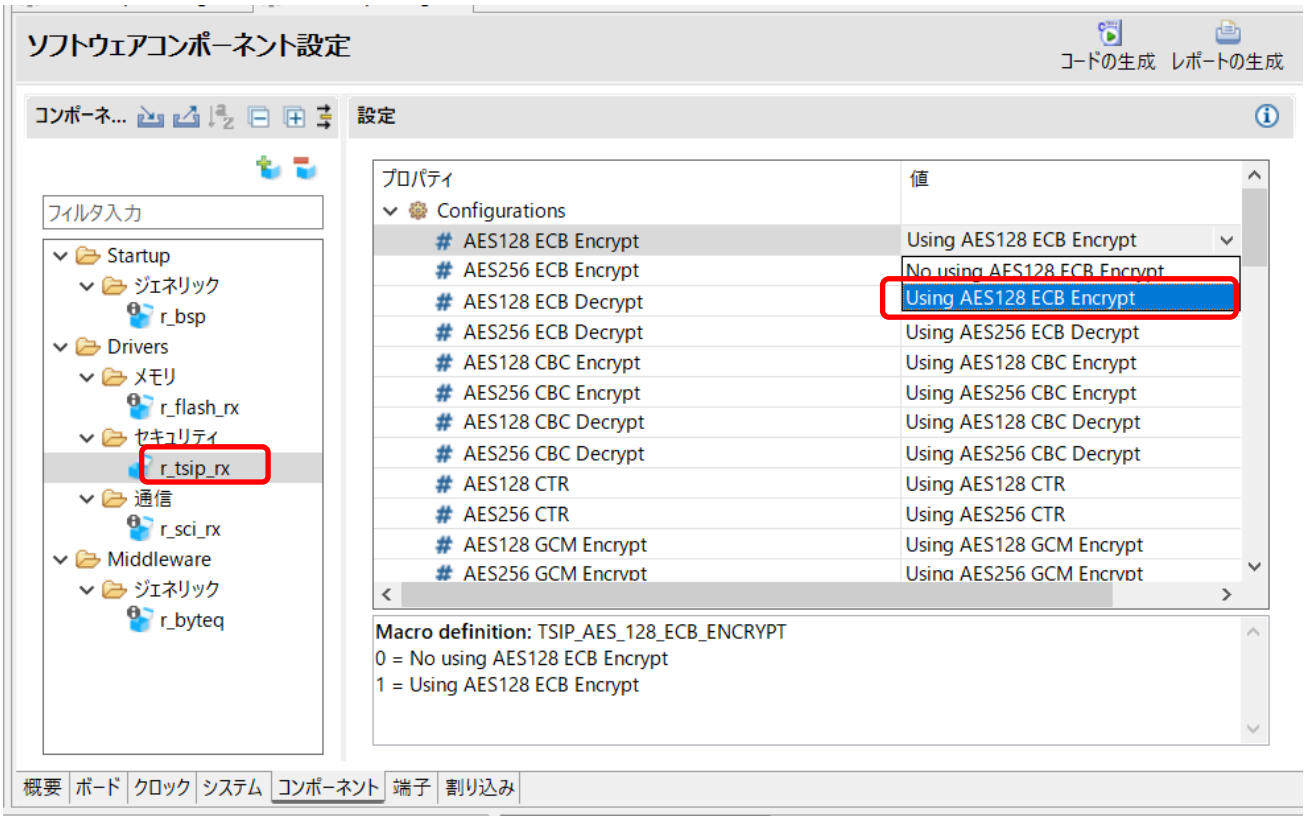
TSIP API 関数を使用する前に、r_bsp_config.h の以下のマクロを表 **3-1** に示すように設定してください。r_bsp_config.h は、本来は手動での編集は行わないファイルですが、本設定値については必要に応じて手動での編集をしてください。

表 3-1 BSP 設定ファイルの設定値

使用 MCU	設定するマクロ	設定する値
RX65N	BSP_CFG_MCU_PART_ENCRYPTION_INCLUDED	true
RX72N	BSP_CFG_MCU_PART_FUNCTION	0x11

3.4　AES を使用するための TSIP ドライバ設定

デフォルトでは、TSIP FIT モジュールを追加すると、TSIP FIT モジュールがサポートする機能がすべて有効になっています。これには、暗号、ハッシュ、乱数生成などが含まれます。AES 暗号のみを必要とするアプリケーションの場合、下図に示されているように AES コンポーネントを有効にして、その他すべてのコンポーネントを、スマート・コンフィグレータで各コンポーネントのドロップダウンメニューから[No using...]（...を使用しない）とすることで使用メモリの容量を削減できます。



3.5　鍵のラップと Wrapped Key

TSIP ドライバは、平文のユーザ鍵を入力として受け入れないため、鍵をラップして TSIP ドライバが受け入れられる形式に変換する必要があります。ここでは、ユーザ鍵を TSIP ドライバが受け入れられる形式に変換してデバイスに書き込む方法について説明します。

ユーザ鍵をより安全に使用するために、TSIP ドライバで使用するユーザ鍵は、Key Wrap サービス（https://dlm.renesas.com/）および Security Key Management Tool を使用して、以下の手順でラップします。

1. まず、ユーザ鍵と UFPK の 2 つの鍵を用意します。ユーザ鍵は暗号処理で使用する鍵、UFPK はユーザ鍵をラップするために使用する鍵です。

2. 次に、UFPK を Key Wrap サービスでラップし、ラップされた UFPK を生成します。Security Key Management Tool を使用して、Key Wrap サービスが受け付けるフォーマットの UFPK ファイルを作成することができます。UFPK ファイルの作成方法は、Security Key Management Tool のユーザーズマニュアルおよびアプリケーションノート「RX ファミリ TSIP(Trusted Secure IP)モジュール Firmware Integration Technology (R20AN0371)」を参照してください。Key Wrap サービスの使用方法は、Key Wrap サービスのトップページにある FAQ より、操作マニュアルを参照してください。

3. その後、ラップされた UFPK、平文の UFPK、ユーザ鍵の 3 つを Security Key Management Tool に入力すると、ラップされた UFPK と UFPK でラップされたユーザ鍵（Encrypted Key）が出力された暗号化鍵ファイルが生成されます。詳細な手順については、Security Key Management Tool のユーザーズマニュアルおよびアプリケーションノート「RX ファミリ TSIP(Trusted Secure IP)モジュール Firmware Integration Technology (R20AN0371)」を参照してください。

作成した暗号化鍵ファイルをプロジェクトに含める必要があります。ラップされた鍵を使用する利点は、平文のユーザ鍵をプログラムに含める必要がないため、ユーザによって平文のユーザ鍵の安全を保証できるということです。

各 TSIP ドライバ API には「Wrapped Key」が必要です。これは、他の API が呼び出される前の最初のステップで生成する必要があります。Wrapped Key は、UFPK と Encrypted Key をもとに Wrapped Key 生成 API によって生成されます。生成された Wrapped Key はデータフラッシュ領域（0x00100000）等にインストールすることができます。

Wrapped Key の生成には Unique ID に紐づく鍵が使用されます。Unique ID はデバイス固有の値であるため、「Wrapped Key」は同じユーザ鍵から生成されたとしてもデバイスごとに異なります。したがって、「Wrapped Key」が攻撃者によってデバイスから読み取られても、他のデバイスでは利用できないため、システムのセキュリティ上の脅威とはなりません。

サンプルプロジェクトでは、以下のユーザ鍵と UFPK を使用しています。デモを簡単に行えるように、Key Wrap サービスと Security Key Management Tool を使用して作成済みの暗号化鍵ファイル（euk_aes128.c および euk_aes128.h）をプロジェクトに追加しています。

表 3-2　使用する鍵

鍵の種別	平文鍵（16 進数）
ユーザ鍵 （AES-128 ビット）	11111111222222223333333344444444
UFPK	22222222222222222222222222222222111111111111111111111111111111111

4. デモプロジェクト

デモプログラムは、ターミナルウィンドウのプロンプトでユーザコマンド入力を実行します。

4.1 デモコマンド

表 4-1 は、デモでサポートされるユーザコマンドの一覧です。コマンドでは大文字/小文字が区別されるので注意してください。

表 4-1　コマンドリスト

コマンド	説明
ecbdemo <plain text>	16 進数 16 バイトの入力平文を ECB モードで暗号化します
cbcdemo <plain text>	16 進数 16 バイトの入力平文を CBC モードで暗号化します
function	サンプルデータを使用して AES ECB/CBC API をテストし、結果を出力します

4.2 ファイルおよびフォルダの一覧

表 4-2 は、プロジェクト内のファイルおよびフォルダの一覧です。ペリフェラルドライバは、スマート・コンフィグレータから生成されます。

表 4-2　ファイルおよびフォルダの一覧

フォルダ名	ファイル名/フォルダ名	説明
src/genkey	euk_aes128.c euk_aes128.h	Security Key Management Tool から生成された鍵データ
src/smc_gen	• general • r_bsp • r_byteq • r_config • r_flash_rx • r_pincfg • r_sci_rx • r_tsip_rx	スマート・コンフィグレータから生成されるコード • BSP を含む • ペリフェラルドライバ • 端子構成 • BSP および周辺機能構成
src	main.c	メイン関数
src	key_data.c key_data.h	サンプルプログラムで使用する Wrapped Key と状態を管理する構造体
src	tsip_sample_aes.c	AES 関数テスト用サンプルプログラム
src	secure_boot.c secure_boot.h	AES の Wrapped Key を生成し、データフラッシュにインストールするためのサンプルプログラム
src	tsip_sample.c tsip_sample.h	データ出力用ユーティリティ関数

4.3　AES 実装用 TSIP ドライバ API 関数の一覧

サンプルコードは、ECB および CBC モードの実装に以下の AES 128 用 TSIP ドライバ API 関数を使用します。これらの関数は、TSIP 初期化、Wrapped Key の生成、ECB および CBC モードでの入力データの暗号/復号に使用します。

表 4-3　AES 用 API 関数

型	API	説明
TSIP 初期化用 API 関数	R_TSIP_Open	TSIP 機能を有効にします。
	R_TSIP_Close	TSIP 機能を無効にします。
ユーザ鍵の Wrapped Key 生成用 API	R_TSIP_GenerateAes128KeyIndex	128 ビット AES ユーザ鍵の Wrapped Key を生成します。
ECB モード暗号化および復号化用 API	R_TSIP_Aes128EcbEncryptInit	128 ビット AES ユーザ鍵の Wrapped Key を使用して AES128-ECB モードでデータを暗号化する準備をします。
	R_TSIP_Aes128EcbEncryptUpdate	AES128-ECB モードでデータを暗号化します。
	R_TSIP_Aes128EcbEncryptFinal	AES128-ECB モードで暗号化の最終処理を実行します。
	R_TSIP_Aes128EcbDecryptInit	128 ビット AES ユーザ鍵の Wrapped Key を使用して AES128-ECB モードでデータを復号化する準備をします。
	R_TSIP_Aes128EcbDecryptUpdate	AES128-ECB モードでデータを復号化します。
	R_TSIP_Aes128EcbDecryptFinal	AES128-ECB モードで復号化の最終処理を実行します
CBC モード暗号化および復号化用 API	R_TSIP_Aes128CbcEncryptInit	128 ビット AES ユーザ鍵の Wrapped Key を使用して AES128-CBC モードでデータを暗号化する準備をします。
	R_TSIP_Aes128CbcEncryptUpdate	AES128-CBC モードでデータを暗号化します。
	R_TSIP_Aes128CbcEncryptFinal	AES128-CBC モードで暗号化の最終処理を実行します。
	R_TSIP_Aes128CbcDecryptInit	128 ビット AES ユーザ鍵の Wrapped Key を使用して AES128-CBC モードでデータを復号化する準備をします。
	R_TSIP_Aes128CbcDecryptUpdate	AES128-CBC モードでデータを復号化します。
	R_TSIP_Aes128CbcDecryptFinal	AES128-CBC モードで復号化の最終処理を実行します。

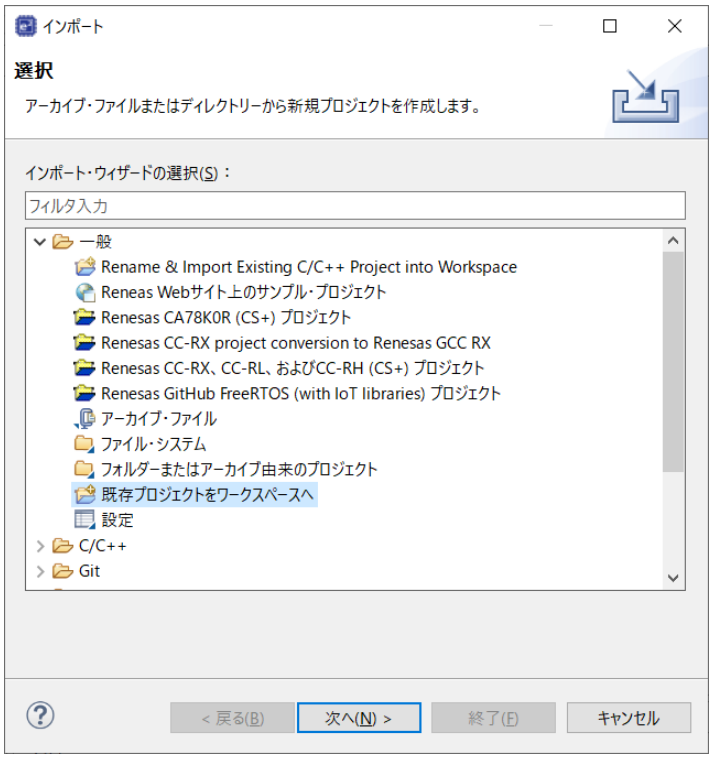
5.　プロジェクトのビルドおよび実行

本章では、RX65N RSK を使う場合の手順を説明いたします。他の RX ファミリデバイスを使用される場合は、プロジェクト名をご使用になるプロジェクト名に置き換えてお読みください。

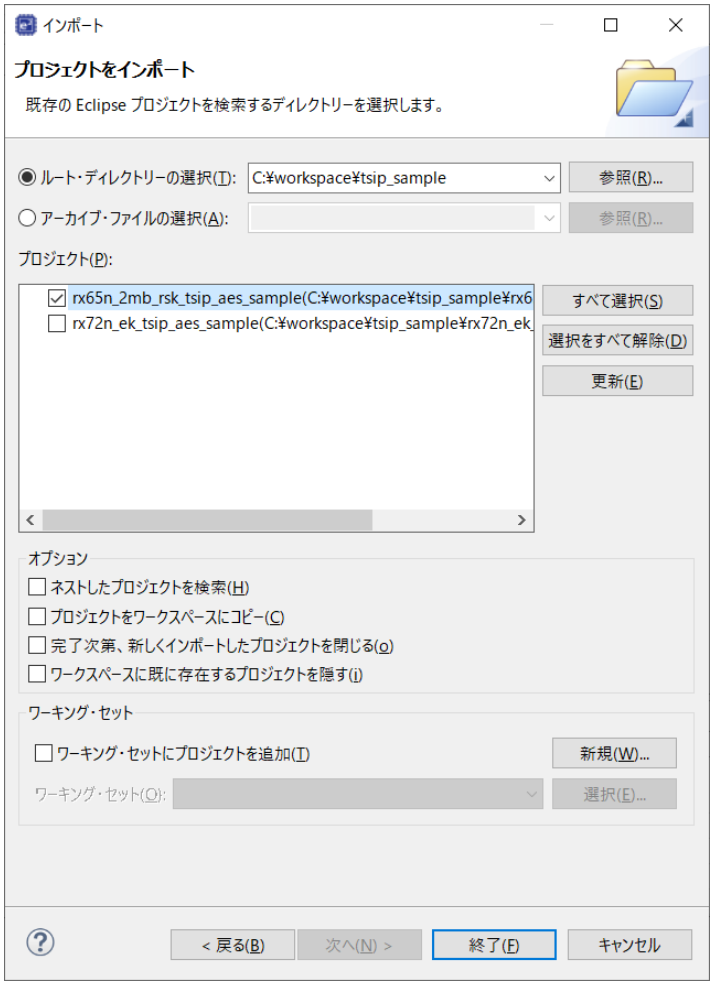
以下の画面は e² studio 2024-10 を使用した画面ならびに手順です。e² studio のバージョンによっては、画面や手順が異なります。

5.1　プロジェクトのインポートおよびビルド

e² studio を開いて任意のワークスペースに移動し、[ファイル(F)]→[インポート(I)…]をクリックします。次に、下に示すように[一般]→[既存プロジェクトをワークスペースへ]を選択します。



[ルート・ディレクトリの選択(T)]を選択し、[(参照(R) …]ボタンを押して、プロジェクトが格納されているディレクトリを選択します。rx65n_2mb_rsk_tsip_aes_sample を選択し、画面が下のように表示されていることを確認したら[終了(F)]をクリックします。「rx65n_2mb_rsk_tsip_aes_sample project」がワークスペースにインポートされます。[プロジェクト]→[プロジェクトのビルド(B) Ctrl+B]を選択してプロジェクトをビルドします。


インポート
プロジェクトをインポート
既存の Eclipse プロジェクトを検索するディレクトリーを選択します。
ルート・ディレクトリーの選択(T):
C:¥workspace¥tsip_sample
参照(R)...
アーカイブ・ファイルの選択(A):
参照(R)...
プロジェクト(P):
rx65n_2mb_rsk_tsip_aes_sample(C:¥workspace¥tsip_sample¥rx6
rx72n_ek_tsip_aes_sample(C:¥workspace¥tsip_sample¥rx72n_ek
すべて選択(S)
選択をすべて解除(D)
更新(E)
オプション
ネストしたプロジェクトを検索(H)
プロジェクトをワークスペースにコピー(C)
完了次第、新しくインポートしたプロジェクトを閉じる(o)
ワークスペースに既に存在するプロジェクトを隠す(i)
ワーキング・セット
ワーキング・セットにプロジェクトを追加(T)
新規(W)...
ワーキング・セット(O):
選択(E)...
< 戻る(B)
次へ(N) >
終了(F)
キャンセル

5.2　ハードウェアの設定

E1 または E2Lite エミュレータを使用してコードをダウンロードできるように、下の図で示すようにハードウェアを接続します。短絡を防止するために、ボードに物理的な変更を加える前に電源が接続されていないことを確認します。また、RX65N RSK では MCU をシングルチップモードにするために、SW4-1 が OFF であることも確認します。RX72N Envision Kit では、エミュレータモードにするために、SW1-2 が OFF であることも確認してください。

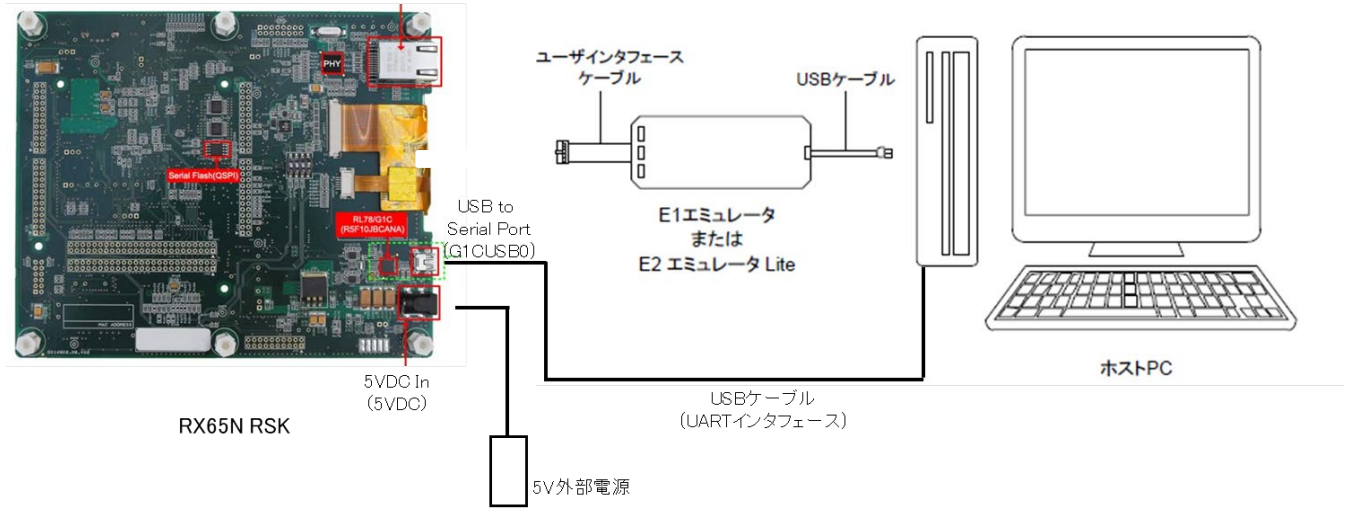


図 5.1 RX65N RSK の接続関係

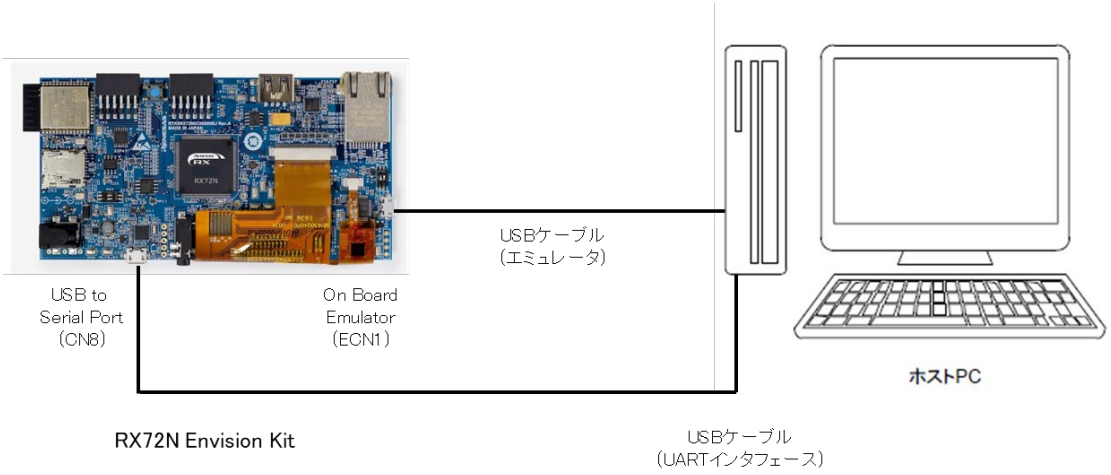
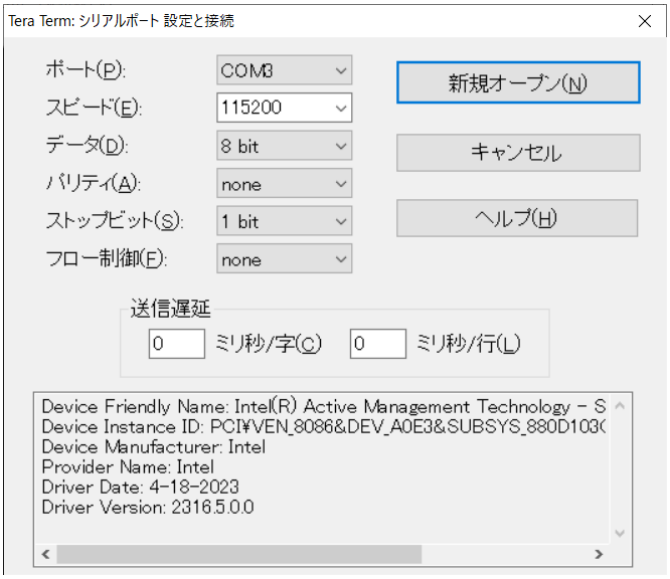


図 5.2 RX72N Envision Kit の接続関係

5.2.1　ターミナルソフトウェア通信

ターミナルソフトウェアでの UART 通信用 PC に USB ケーブルを接続します。「シリアル」通信用に Tera-Term を構成します。まず端末の設定で送信時の改行コードを「CR」とします。次にシリアルポートを下の図のように設定します。

スピード：115200
パリティ：なし
フロー制御：なし

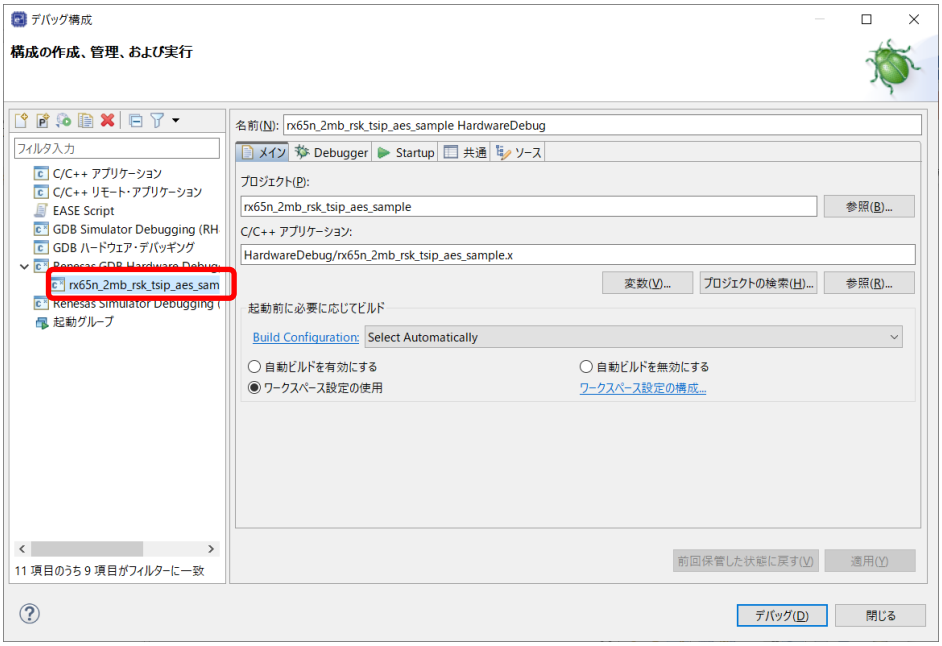


5.2.2　プログラムのダウンロード

プログラムのダウンロードは、e² studio か Renesas Flash Programmer のいずれかを使用して実施します。

5.2.2.1　e² studio を使用したプログラムのダウンロード

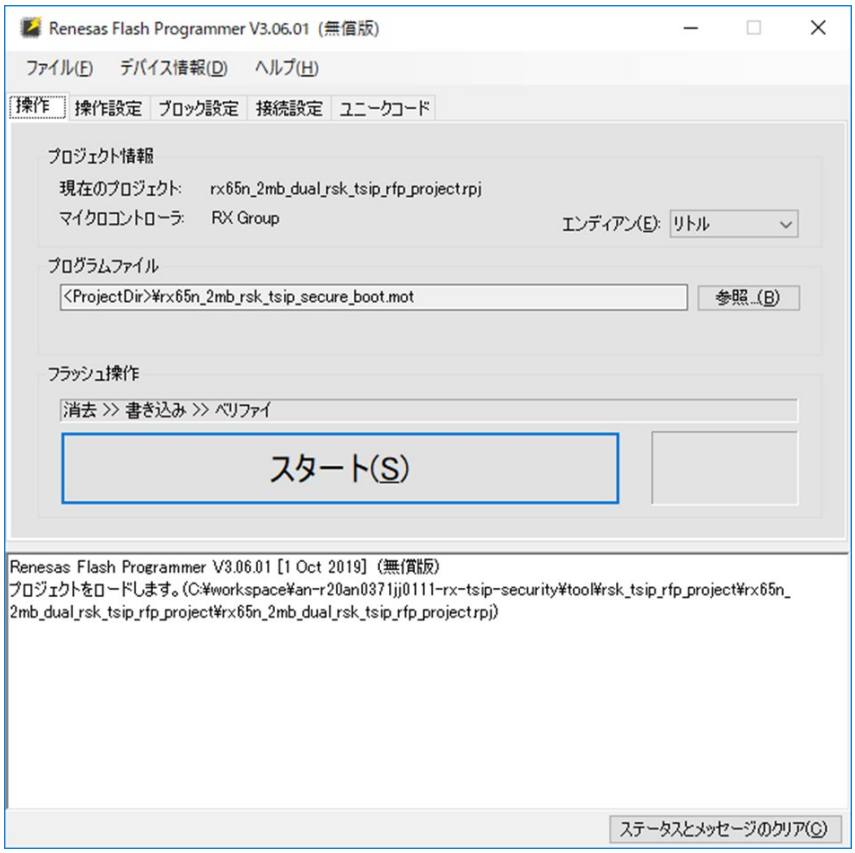
ハードウェア設定が完了したら、[デバッグ(D)]→[デバッグ構成(B)...]をクリックし、「rx65n_2mb_rsk_tsip_aes_sample HardwareDebug」を選択して、[デバッグ(D)]ボタンをクリックするとプログラムがダウンロードされます。



5.2.2.2　Renesas Flash Programmer を使用したプログラムのダウンロード

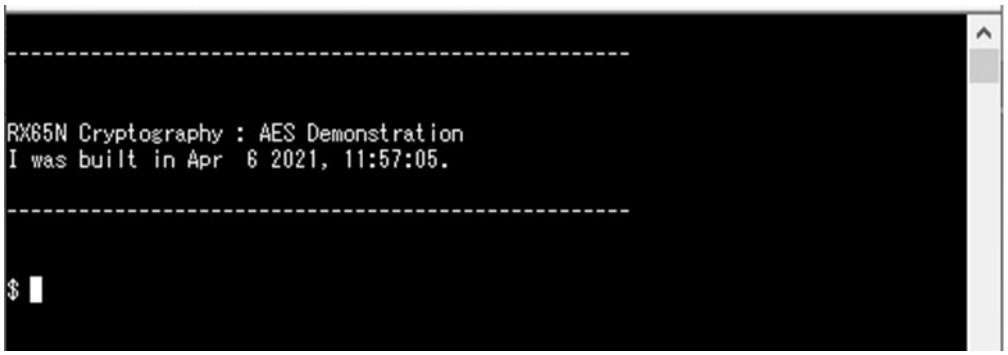
本プロジェクトのプログラムは、デバッガを使用しないスタンドアロンで動作可能です。スタンドアロンで動作を行う場合は、Renesas Flash Programmer(RFP)を使用してください。RFP の使用方法は以下サイトをご参照ください。

https://www.renesas.com/rfp



5.3　プログラムの実行

e² studio からプログラムを実行するために、[Resume]（再開）ボタンをクリックします。最初にクリックすると、main 関数でブレークします。[Resume]（再開）ボタンを再度クリックしてプログラムを実行します。以下の出力が Tera-Term に表示されます。



- $プロンプトがユーザコマンドを受け入れる準備が整いました。プロンプトで以下のコマンドを入力します。

  1. ecbdemo <plain text>
  2. cbcdemo <plain text>
  3. function

- コマンドを以下のように入力すると、図のようにターミナルウィンドウに出力が表示されます。

  1. $ ecbdemo aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa

     平文（input data (plain text)）を表 3-2 のユーザ鍵を使用して AES128-ECB モードで暗号化した暗号文（output data (cipher text)）が表示されます。Wrapped Key（key_index）はデバイスごとに異なる値となります。



  2. $ cbcdemo aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa

     平文（input data (plain text)）を表 3-2 のユーザ鍵と初期化ベクタ（=12345678fedcba0955555555aaaaaaaa） を使用して AES128-CBC モードで暗号化した暗号文（output data (cipher text)）が表示されます。Wrapped Key（key_index）はデバイスごとに異なる値となります。



3. $ function
   サンプルデータを使用して AES ECB/CBC API をテストし、結果を出力します。

```
$ function
R_TSIP_Open() returns:                          TSIP_SUCCESS
R_TSIP_Aes128EcbEncryptInit() returns:          TSIP_SUCCESS
R_TSIP_Aes128EcbEncryptUpdate() returns:        TSIP_SUCCESS
R_TSIP_Aes128EcbEncryptFinal() returns:         TSIP_SUCCESS
aes128_ecb_enc_input:
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaabbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbccccccccccccccccc
ccccccccccccccccdddddddddddddddddddddddddddddddd
aes128_ecb_enc_output:
f4136afc2a9df52b31c447f4d13a78b4c1c759e6e44a6de1d23e6efb5c48fb55dcdb81139b2a6ca3
14ca50c7e4fa1a179310094c38bd4e2c93ea63670bd92cb3
aes128_ecb_enc_expect:
f4136afc2a9df52b31c447f4d13a78b4c1c759e6e44a6de1d23e6efb5c48fb55dcdb81139b2a6ca3
14ca50c7e4fa1a179310094c38bd4e2c93ea63670bd92cb3

R_TSIP_Aes128EcbDecryptInit() returns:          TSIP_SUCCESS
R_TSIP_Aes128EcbDecryptUpdate() returns:        TSIP_SUCCESS
R_TSIP_Aes128EcbDecryptFinal() returns:         TSIP_SUCCESS
aes128_ecb_dec_input:
f4136afc2a9df52b31c447f4d13a78b4c1c759e6e44a6de1d23e6efb5c48fb55dcdb81139b2a6ca3
14ca50c7e4fa1a179310094c38bd4e2c93ea63670bd92cb3
aes128_ecb_dec_output:
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaabbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbccccccccccccccccc
ccccccccccccccccdddddddddddddddddddddddddddddddd
aes128_ecb_dec_expect:
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaabbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbccccccccccccccccc
ccccccccccccccccdddddddddddddddddddddddddddddddd

R_TSIP_Aes128CbcEncryptInit() returns:          TSIP_SUCCESS
R_TSIP_Aes128CbcEncryptUpdate() returns:        TSIP_SUCCESS
R_TSIP_Aes128CbcEncryptFinal() returns:         TSIP_SUCCESS
aes128_cbc_enc_input:
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaabbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbccccccccccccccccc
ccccccccccccccccdddddddddddddddddddddddddddddddd
aes128_cbc_enc_output:
f5c065e3cfdb49f4d62c1011baf001851d70ad50b9ca17af41cec50fe152e8db1a8bc5d62baf765e
d3d064a3109cf05ccd78f719397e05ae2341cb816a2764fa
aes128_cbc_enc_expect:
f5c065e3cfdb49f4d62c1011baf001851d70ad50b9ca17af41cec50fe152e8db1a8bc5d62baf765e
d3d064a3109cf05ccd78f719397e05ae2341cb816a2764fa

R_TSIP_Aes128CbcDecryptInit() returns:          TSIP_SUCCESS
R_TSIP_Aes128CbcDecryptUpdate() returns:        TSIP_SUCCESS
R_TSIP_Aes128CbcDecryptFinal() returns:         TSIP_SUCCESS
aes128_cbc_dec_input:
f5c065e3cfdb49f4d62c1011baf001851d70ad50b9ca17af41cec50fe152e8db1a8bc5d62baf765e
d3d064a3109cf05ccd78f719397e05ae2341cb816a2764fa
aes128_cbc_dec_output:
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaabbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbccccccccccccccccc
ccccccccccccccccdddddddddddddddddddddddddddddddd
aes128_cbc_dec_expect:
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaabbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbccccccccccccccccc
ccccccccccccccccdddddddddddddddddddddddddddddddd

R_TSIP_Close() returns:                         TSIP_SUCCESS
```

4. 上のすべてのコマンドでは大文字と小文字が区別されます。大文字/小文字が一致しない場合、エラーが出力されます。下の出力は、$ fun と入力した場合のものです。

```
$ fun
command format error.

$
```

6. Security Key Management Tool の使用方法

暗号化されたユーザ鍵(Encrypted Key)を生成するために、Security Key Management Tool を使用することが可能です。

Security Key Management Tool はコマンドラインインタフェース版(CLI 版)も用意しているため、工場などの生産工程でも簡単に扱うことが可能です。

Security Key Management Tool

https://www.renesas.com/software-tool/security-key-management-tool

Security Key Management Tool の使用方法の詳細はユーザーズマニュアルをご確認ください。

ホームページとサポート窓口

ルネサス エレクトロニクスホームページ
https://www.renesas.com/jp/ja/

お問合せ先
https://www.renesas.com/jp/ja/support/contact.html

改訂記録

Rev.	発行日	改訂内容	
		ページ	ポイント
1.00	Mar.31.21	—	初版発行
1.01	Jun.30.21	— 24 - 25	RX72N Envision Kit の説明の追加 6 章 Renesas Secure Flash Programmer の使用方法の追加
1.02	Nov.30.23	—	使用するユーザ鍵暗号化ツールを Security Key ManagementTool に変更
1.03	Apr. 10.25	—	用語に関する変更を適用

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　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。

© 2025 Renesas Electronics Corporation. All rights reserved.