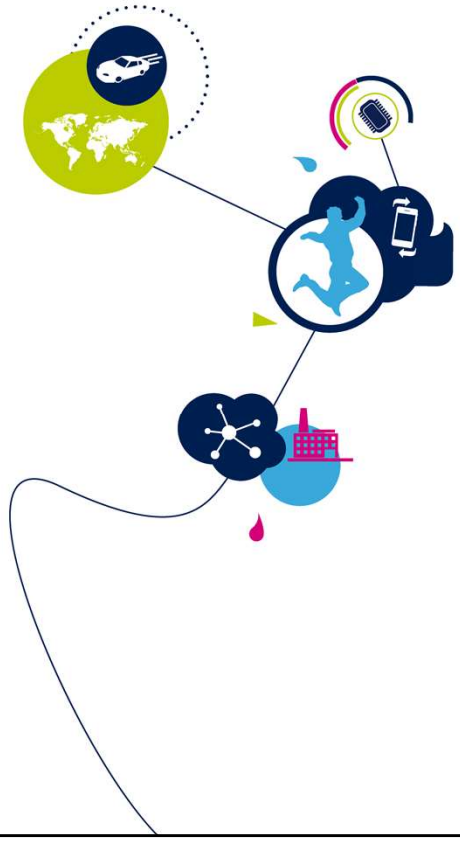
STM32MP1 – RNG

真性乱数発生器

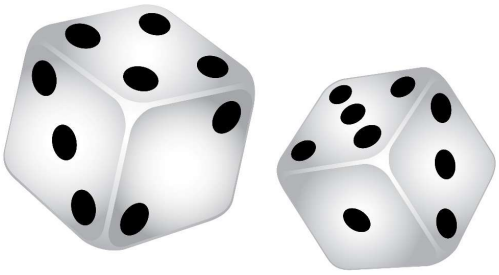
1.0 版



life.augmented

STM32 乱数発生器のプレゼンテーションにようこそ。このプレゼンテーションでは、乱数の生成に広く使用されている、このペリフェラルの機能について説明します。

概要



- 乱数の生成
  - 予測不可能な結果の生成が求められる場合に使用されます。
- 2 つのインスタンス：RNG1 および RNG2

アプリケーション側の利点

- 数字のランダム性を高めます。
- 値が推測される可能性を大幅に低減します。

life.augmented

STM32 製品内に組み込まれた乱数発生器（RNG）は、予測不可能な結果の生成が求められる場合に使用される乱数を生成します。アプリケーションでは、RNG によって数字のランダム性が高まり、特定の値が推測される可能性を下げるというメリットがあります。

RNG の使用と関連するソフトウェア

- STM32MP1 デバイスには、Arm® コアや関連するセキュリティモードの違いに応じて複数のランタイムコンテキストが存在します。
  - Cortex®-A7 セキュア（Trustzone）は、セキュアモニタや OP-TEE のようなセキュア OS で動作します。
  - Cortex-A7 非セキュアは、Linux で動作します。
  - Cortex-M4（非セキュア）は、STM32Cube で動作します。
- サポートされる RNG ペリフェラル割当て:

RNG1 はセキュアペリフェラルです（ETZPC_DECPROT0 bit 7 で ETZPC 制御下）。RNG2 は非セキュアペリフェラルです。

RNG1 インスタンスは、次に対して割り当てられます。

- RNG OP-TEE ドライバによって OP-TEE で制御する Arm Cortex-A7 セキュアコア
- Linux ハードウェア乱数フレームワークによって Linux® で使用するための Arm Cortex-A7 非セキュアコア

RNG2 インスタンスは、STM32Cube RNG ドライバによって STM32Cube MPU パッケージで制御する Arm Cortex-M4 コアに割り当てられます。

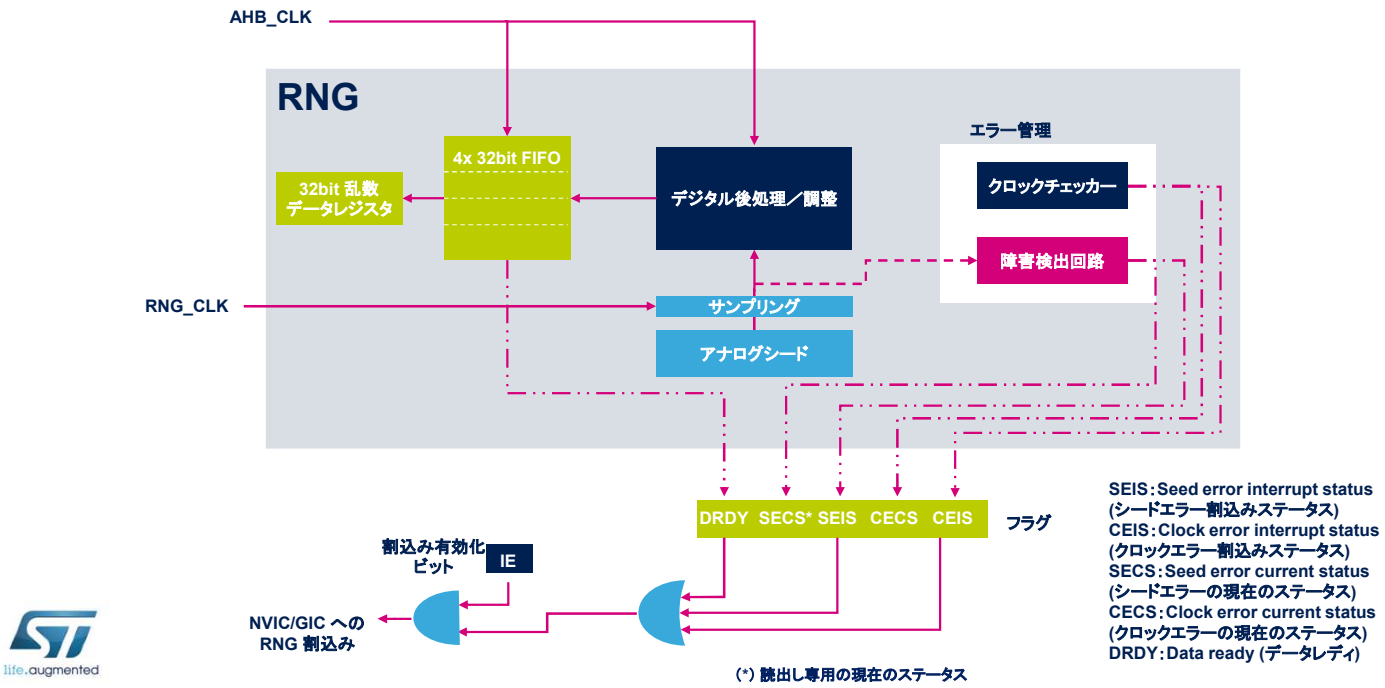
主な機能

- 32bit の乱数発生器はノイズソースに基づいています。
  - 4 個の 32bit 乱数のセットを最低周波数 213 クロックサイクルで生成できます。
    - 実際の値（213 を超えている場合）は 16 x f_{AHB} / f_{RNG}（システムクロックと RNG サンプルクロックの割合）です。
    - f_{AHB} = 266MHz および f_{RNG} = 4MHz の場合、サンプルは 1064AHB サイクルごとに使用できます。
  - 消費電力を低減するために無効にできます（RNG_CR で RNGEN = 0）。
- 次の場合に 3 つのフラグをトリガできます。
  - DRDY：有効な乱数データが準備できています。
  - SECS：シードで異常なシーケンスが発生しています（64bit 以上連続して“0”や“1”の同じ値、あるいは 32bit 以上連続して“01”や“10”のパターン）。
  - **CECS**：f_{RNG} 周波数が f_{AHB} / 32 未満です（このチェックは無効にできます）。
    - クロック供給設定（f_{AHB} = 266MHz and f_{RNG} = 4MHz）では、CED ビットをセットして CECS 検出を無効にする必要があります。
- 3 種類の割込み
  - CEIS：クロックエラーを示します。
  - SEIS：シードエラーを示します。
  - DRDY：有効な乱数データが準備できていることを示します。

life.augmented

RNG ペリフェラルは、後ほど詳細に説明する 32bit の乱数値を生成する連続アナログノイズに基づいています。RNG は、4 個の 32bit 乱数を最低周波数 213システムクロックサイクルで生成できます。おおまかに、RNG クロックが低くなると、サンプリングされる乱数ソースのエントロピーが向上します。
新しい乱数データのセットが準備されて確認されると、データレディフラグがステータスレジスタでセットされます。これは必ず使用する必要があります。
RNG は、生成されるデータのランダム性の基本的な検証を実行します。たとえば、64bit 以上連続して同じ値（0 や 1）である場合や、32bit 以上連続して 0 と 1 が入れ替わっている場合、シードエラーの現在のステータスフラグがセットされます。
RNG クロックが 32 で分周された HCLK クロックを下回る場合、クロックエラーの現在のステータスフラグがセットされます。このチェックは、特に RNG クロックがエントロピーを最大化するためにローで初期化される場合に無効にできます。
異常なシードシーケンスや周波数エラーを示すために、割込みソースを有効にすることもできます。

ブロック図



こちらの RNG の簡略化されたブロック図には、基本的な機能モジュールと制御モジュールを示しています。
乱数発生器は、複数のリングオシレータで構成されるアナログ回路に基づいています。その出力は、計算のラウンドごとに 4 個の 32bit 乱数を生成できるデジタル後処理ブロックを供給するシードを生成するためにサンプリングされて排他的論理和がとられます。
アナログシードのサンプリングは、専用の RNG クロック信号によってクロック供給されるため、乱数の品質は HCLK 周波数から独立しています。後処理ブロックの内容は、4ワード FIFO でデータレジスタに転送されます。FIFO がフルになると、すぐにデータレディフラグ（DRDY）がトリガされ、RNG から読み戻せるデータがなくなると、自動的にリセットされます。
同時に、エラー管理ブロックで、正しいシード動作と RNG ソースクロックの周波数が検証されます。
異常なシーケンスがシードで検出された場合や、RNG 周波数が低すぎる場合は、ステータスビットがセットされ、割込みがトリガされます。
RNG クロックが AHB_CLK/ 32 を下回って固定されている場合（品質上の理由など）、RNG 周波数エラーチェックを無効にする必要があります。

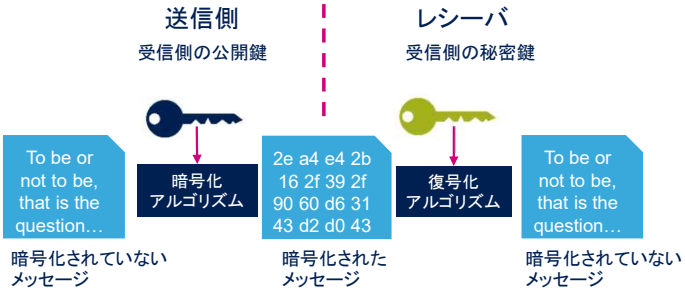
低電力モード

モード	説明
RUN	アクティブです。
SLEEP	RCC または RNG で無効にされます（RNGEN = 0）。RNG を有効にしたままにすると、RNG 初期化時間によって生じる、新しいランダムサンプルが利用可能となるまでの遅延を除去できます。
STOP	最低消費電力のために RCC で無効にされます。
LP-Stop	
LPLV-Stop	
STANDBY	パワーダウン状態です。ペリフェラルは、STANBY モード終了後に再初期化する必要があります。

真性乱数発生器は、RUN モードでのみアクティブです。初期化時間の遅延を回避するために、SLEEP モードで有効にしたままにできます。その他の低電力モードでは無効になり、STANDBY モードや SHUTDOWN モードでは完全にパワーダウン状態となります。

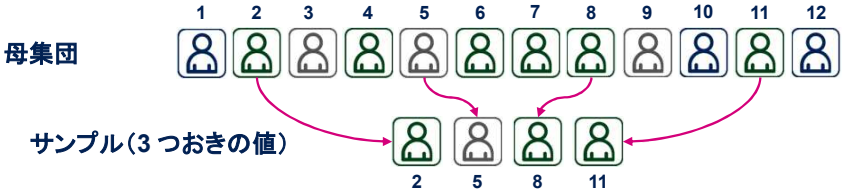
アプリケーション例

- 暗号化



- ゲーム

- 統計サンプリング



RNG は、暗号、ゲーム、統計サンプリングを含む幅広いアプリケーションに使用できます。たとえば、暗号アルゴリズムのセキュリティは、すべて鍵を推測できないようにすることにつながります。そのため、鍵は乱数である必要があり、そうしないと攻撃者が推測できてしまいます。

関連するペリフェラル

- 次のペリフェラルに関するペリフェラルのトレーニングを参照してください。
  - リセットおよびクロック制御（RCC）
  - 暗号プロセッサ（CRYP）
  - ハッシュプロセッサ（HASH）

life.augmented

これは、乱数発生器に関連するペリフェラルの一覧です。詳細については、必要に応じて RCC（RNG クロック制御、RNG 有効化／リセット）を参照してください。
また、暗号エンジンについて知りたい場合は、CRYP や HASH のトレーニングを参照してください。

参照

- AN4230：STM32 microcontrollers random number generation validation using NIST statistical test suite.
  - これには、STM32 マイクロコントローラのセレクションに組み込まれている乱数発生器ペリフェラルによって生成された数字のランダム性を検証するためのガイドラインがあります。
    - この検証は、NIST 統計テストスイート（STS）SP 800-22 に基づいており、SP800-22rev1a（2010 年 4 月）として発行および更新されています。
  - NIST テストスイートは、RNG ペリフェラルを組み込んだ STM32 ボードのセレクションで実行されました。 その結果は、ファームウェアフォルダ「NIST_Test_Suite_OutputExample」内にあります。

詳細については、NIST SP800-22 統計テストスイート（STS）に関するアプリケーションノート AN4230 を参照して、STM32 MCU のセレクションによって生成された乱数を検証してください。