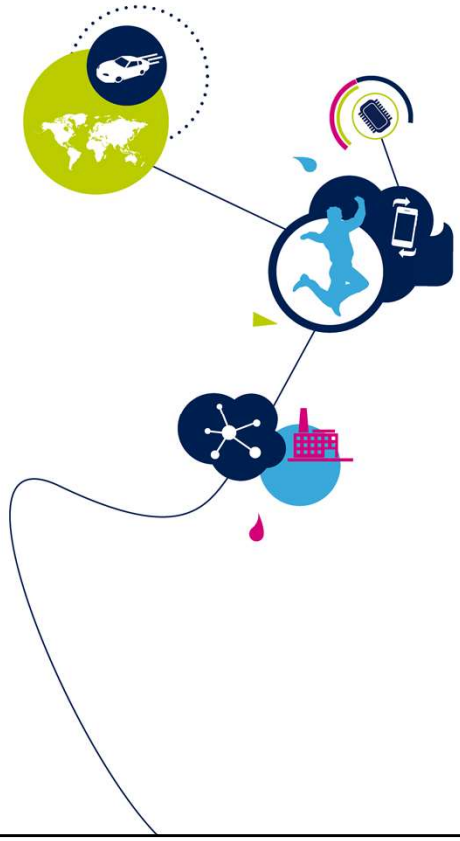
STM32MP1 - HASH

ハッシュプロセッサ：SHA-1、SHA-2、MD5 エンジン
1.0 版

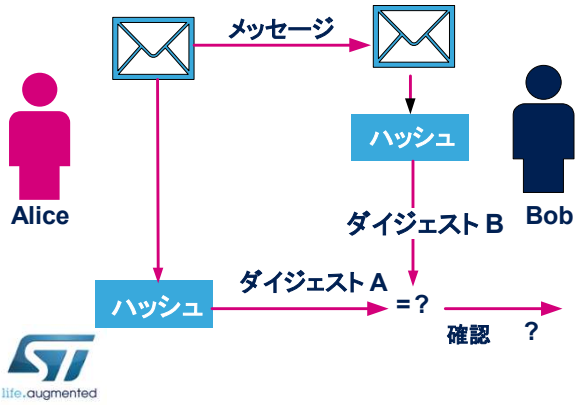


life.augmented

STM32MP1 ハッシュプロセッサのプレゼンテーションにようこそ。

概要

- ハッシュ処理
  - メッセージから固定長のダイジェストを計算します。
  - メッセージはダイジェストから取得できません。
  - 同じダイジェストで 2 つのメッセージを発見することは実質不可能です（衝突）。
- 2 つのインスタンス：HASH1 および HASH2



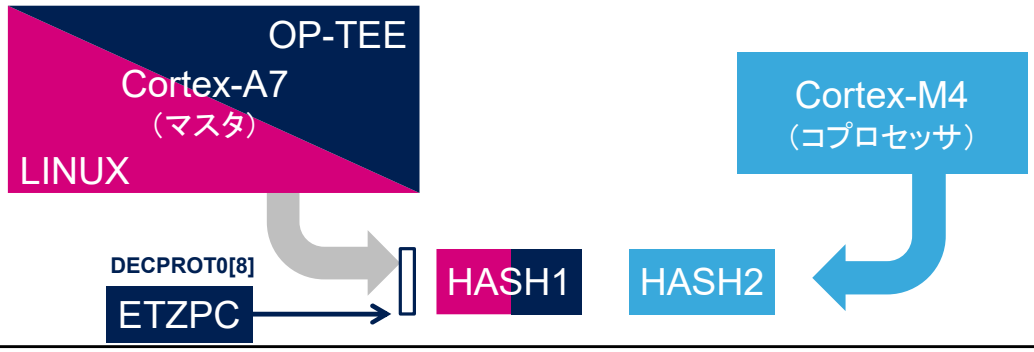
アプリケーション側の利点

- 次を保証してトランザクションを保護するために使用されます。
  - メッセージ整合性（HASH）
  - メッセージ認証（HMAC）
- CPU の処理時間が低減します。

ハッシュペリフェラルは、メッセージダイジェストの効率的な計算を実行します。
ダイジェストとは、入力メッセージから計算される固定長の値です。ダイジェストは一意で、同じダイジェストで 2 つのメッセージを発見することは実質不可能です。元のメッセージはダイジェストから取得できません。
ハッシュダイジェストとハッシュベースメッセージ認証コード（HMAC）は、転送の整合性と認証を保証するために使用されるため、通信で広く使用されています。

HASH 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
  - 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 HASH ペリフェラル割当て：



HASH1 はセキュアペリフェラル（ETZPC_DECPROT0 bit 8 で ETZPC 制御下）ですが、HASH2 は非セキュアペリフェラルです。
HASH1 インスタンスは、次に対して割り当てられます。

- HASH 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 暗号化フレームワークを搭載した Linux® で使用するための Arm Cortex-A7 非セキュアコア

HASH2 インスタンスは、STM32Cube HASH ドライバを使用して 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。

HASH1 インスタンスは、バイナリ認証をサポートするブートデバイスとして使用されます。

主な機能

- ハッシュプロセッサは次のものをサポート
  - 次のハッシュ関数の高速計算
    - MD5
    - SHA-1
    - SHA-224 および SHA-256
  - シンプルなハッシュダイジェストや鍵付きハッシュメッセージ認証コード（HMAC）の計算
  - ビッグエンディアンおよびリトルエンディアンに準拠する自動バイトスワッピング
  - ダイレクトメモリアクセス（DMA）およびマスタダイレクトメモリアクセスコントローラ（MDMA）の両方をサポートする自動データフロー制御
  - FIFO がフルになるごとの部分ダイジェスト計算の管理
  - 自動パディングを含む最終ダイジェスト計算の管理
    - 最終ダイジェスト計算は、HASH1 と HASH2 で異なります。
      - レジスタ HASH_HWCFGR は、HASH1 の場合 0x1、HASH2 の場合 0x0 となります。

life.augmented

ハッシュプロセッサは、メッセージダイジェスト 5（MD5）、セキュアハッシュアルゴリズム SHA-1、224bit と 256bit のダイジェスト長バージョンを持つさらに新しい SHA-2 を含め、広く使用されるハッシュ関数をサポートしています。
ハッシュを秘密鍵で生成して、メッセージ認証コード（MAC）を生成することもできます。
このプロセッサは、ビット、バイト、およびハーフワードスワッピングをサポートしています。また、ブロック配置での入力データの自動パディングもサポートしています。
このプロセッサは、自動プロセッサ供給を実現するために DMA と組み合わせて使用できます。

ダイジェスト計算は、部分（FIFO がフルになるたび）または最終（それ以上追加するバイトがない場合）のいずれかに設定できます。アプリケーションは、ケースに応じて最終ダイジェストを違う形で管理します。

- 単一の DMA 転送（MDMAT ビット= “0”）
- 複数の DMA 転送（MDMAT ビット= “1”）
- MDMA での転送（HASH1 のみ）

ハッシュ関数

- ブロックベースアルゴリズム
  - サポートされるハッシュ関数は、512bit のデータブロックで動作します。元のメッセージは、必要に応じてパディングされた後、512bit の連続したブロックに分けられます。
    - 有効ビット数が 32bit ではない場合、NBLW を更新して最後のワードで有効ビット数を定義します。
  - 衝突に対する堅牢性は、ダイジェスト長で上がります。
    - MD5 および SHA-1 は、NIST によって安全なダイジェストとみなされていません。

ハッシュ関数		ダイジェスト（bit）	強度（衝突）	安全なダイジェスト(*)
MD5		128	2^64	不可
SHA-1		160	2^80	
SHA-2	SHA-224	224	2^112	可能
	SHA-256	256	2^128	

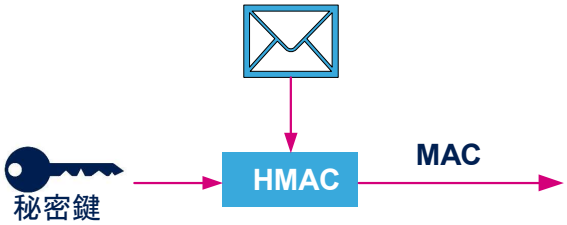
life.augmented

(*) NIST による

すべてのサポートされるハッシュ関数は、512bit のデータブロックで動作します。入力メッセージは、ハッシュプロセッサに供給するために必要な数に分けられます。連続したブロックが順番に計算されます。
MD5 は、わずか 128bit ダイジェストの堅牢性の低い関数です。
SHA 標準には、SHA-1 と、224bit と 256bit のダイジェスト長バージョンを持つさらに新しい SHA-2 の 2 バージョンがあります。
MD5 および SHA-1 は、NIST によって安全なダイジェストとみなされていません。

メッセージ認証コードのための HMAC 鍵付きハッシング

- IETF RFC2104 および NIST FIPS PUB198-1 で定義されている HMAC
  - 整合性に加えて、メッセージの認証も保証します。
  - 送信側と受信側の両方で共有される秘密鍵が含まれます。
- このアルゴリズムは 2 つのネストされたハッシュ操作で構成
  - HMAC(message) = Hash[((key | pad) ⊕ 0x5C) | Hash(((key | pad) ⊕ 0x36) | message)]
  - ハッシュ関数は、ペリフェラルによってサポートされるうちのいずれかになります。



life.augmented

ハッシュベースのメッセージ認証コード（HMAC）は、メッセージの認証と整合性の確認に使用されます。HMAC 関数は、送信側と受信側の両方で共有される秘密鍵がある 2 つのネストされたハッシュ関数で構成されます。
HMAC 計算に含まれるハッシュ関数は、ペリフェラルによってサポートされるものに設定できます。MD5、SHA-1、SHA-2 のいずれかです。

標準

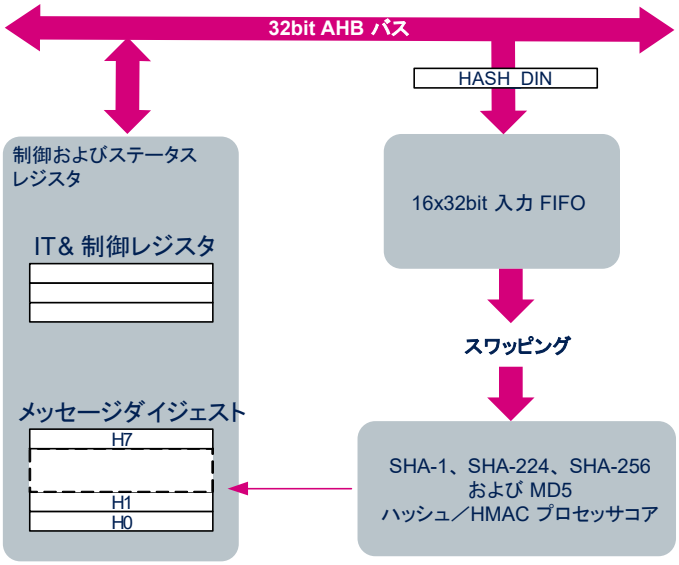
標準の準拠

- このハッシュプロセッサはデータ認証アプリケーションに適しており、次の標準に準拠
  - セキュアハッシュ標準（SHA-1、SHA-224 および SHA-256）では FIPS PUB 180-4（連邦情報処理規格公報 180-4）
  - IETF RFC 1321（インターネット技術タスクフォース RFC 1321） MD5 メッセージダイジェストアルゴリズム
  - 鍵付きハッシュメッセージ認証コード（HMAC）では NIST FIPS PUB 198-1 および IETF RFC 2104

life.augmented

ハッシュプロセッサは、セキュアハッシュアルゴリズム（SHA）、メッセージダイジェストアルゴリズム（MD5）、鍵付きハッシュメッセージ認証コード（HMAC）の国際標準に準拠しています。

ブロック図



life.augmented

こちらのハッシュプロセッサの簡略化されたブロック図には、基本的なデータフローと制御モジュールを示しています。
ハッシュプロセッサは、512bit データブロックを処理して、アルゴリズムに応じて最大 256bit のダイジェストを生成します。
入力データは、コアユニットに移行し、そこで処理されてシンプルなハッシュやメッセージ認証コード（MAC）を生成する前にスワッピングできます。

割込みおよび DMA

割込みイベント	説明
ハッシュダイジェスト計算完了	ダイジェストがレディになったとき（メッセージ全体が処理済）にセットされます。
ハッシュデータ入力レディ	入力バッファが 512bit の新しいブロックを取得する準備ができたときにセットされます（空き領域が 16 か所）。

- DMA 機能：メモリからの入力データ用のチャネル 1 個
  - シングルリクエストと 4 ワードの固定バーストリクエストをサポートしています。
  - ハッシュプロセッサが、512bit のデータブロック 1 個をロードするために DMA リクエスト（HASH_IN）を発生させます。
  - DMA の転送完了割込みをデータフロー制御に使用できます。

life.augmented

ネスト化されたベクタ割込みコントローラ（NVIC）での割込みは、ハッシュダイジェストの計算が完了したとき、またはハッシュプロセッサが新しいデータブロックを受け入れる準備ができたときにトリガされます。

ダイレクトメモリアクセス（DMA）モードでは、入力データに対するリクエストが内部で生成されます。DMA チャネルは、512bit のデータサイズでメモリからペリフェラルモードに設定する必要があります。シングルリクエストと 4ワードの固定バーストリクエストをサポートしています。

パフォーマンス

- ハッシュのパフォーマンス
  - メッセージの中間ブロック(512bit)の計算について、以下にまとめています。

	MD5	SHA-1	SHA-224	SHA-256
処理(サイクル)	**66**	**82**	**66**	
ダイジェストサイズ (bit)	128	160	224	256

  - 以下に示すような追加処理が期待されます。
    - 最終ダイジェスト計算:部分ダイジェスト計算と比較すると最大 2.5 倍
    - HMAC のネスト化された操作:ショートキーが使用される場合は～2.5 倍(ロングキーが使用される場合は最大 5 倍)
  - このペリフェラルによって、速度が上がり、アルゴリズムのソフトウェアバージョンと比較して電力が削減されます。

life.augmented

これらは、選択したアルゴリズムに応じて 1 つのデータブロックの処理にかかる時間です。
HCLK は CPU クロックで、216MHz ハイにできます。
ハードウェアアクセラレータの主なメリットは、速度が上がり、ハッシュ関数の完全ソフトウェア実装と比較して電力が削減されるということに注意してください。
中間ブロックの処理と比べて以下の倍数増える場合があります。
- ハッシュメッセージは 1 から 2.5 倍
- HMAC 入力キーは～2.5 倍
- HMAC メッセージは 1 から 2.5 倍
- ショートキーの場合の HMAC 出力キーは～2.5 倍
- ロングキーの場合のHMAC 出力キーは約 3.5 から 5 倍

低電力モード

モード	説明
RUN	アクティブ
SLEEP	オプションで RCC では無効です。
STOP	停止。ペリフェラルレジスタの内容は保たれます。
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。

life.augmented

ここでは、各低電力モードでのハッシュプロセッサのステータス概要を示します。
デバイスが STOP モードおよび STANDBY モードの場合、ハッシュ操作は実行できません。

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。
  - HASH1 用のマスタダイレクトメモリアクセスコントローラ（MDMA）
  - HASH2 用のダイレクトメモリアクセスコントローラ（DMA）
  - 暗号プロセッサ（CRYP）

life.augmented

これは、ハッシュプロセッサに関連するペリフェラルの一覧です。
ハッシュチャネルの設定方法に関する詳細については、DMA ペリフェラルのトレーニング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 トレーニングを参照してください。

参照

- 詳細および追加情報については、次の文書を参照してください。
  - ユーザマニュアル UM0586：STM32 暗号ライブラリ

詳細については、弊社ウェブサイトで利用できるユーザマニュアルを参照してください。