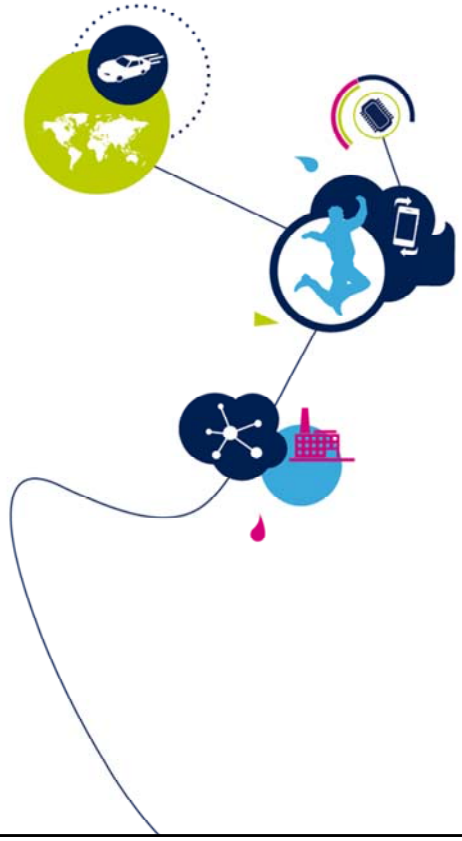
STM32MP1 - HASH

Hash processor: SHA-1, SHA-2 and MD5 engine

Revision 1.0

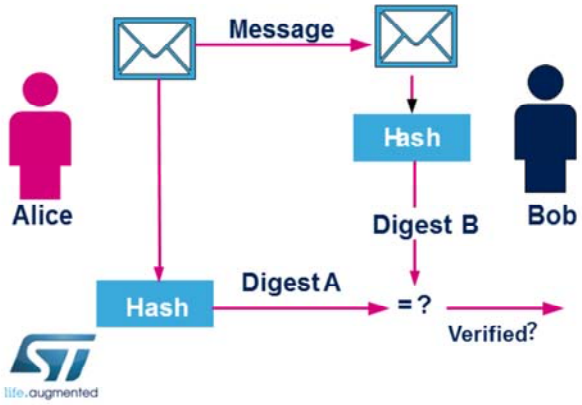


life.augmented

Hello, and welcome to this presentation of the STM32MP1 hash processor.

Overview

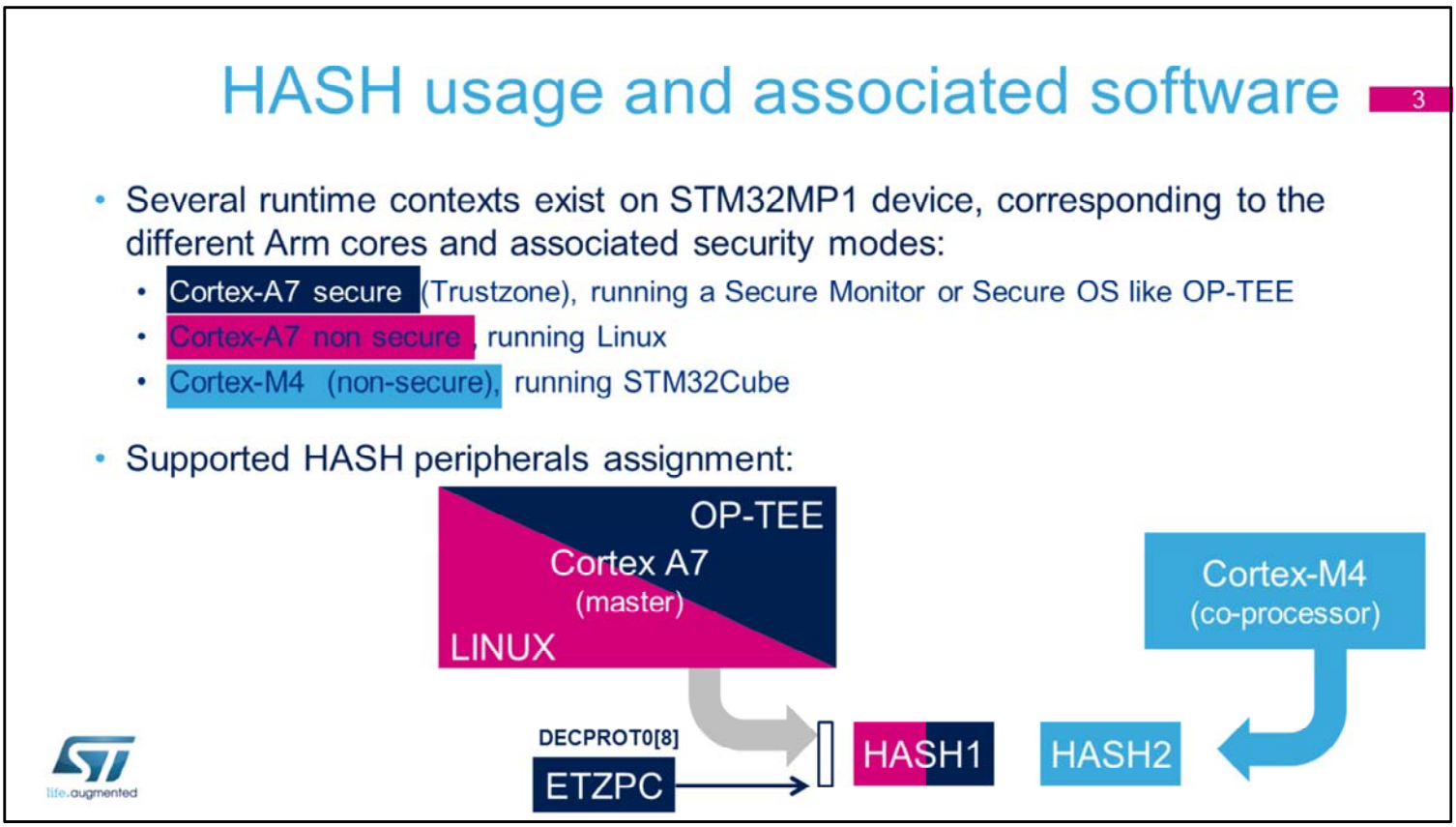
- Hash processing
  - Computes fixed-length digest from a message
  - Message cannot be retrieved from its digest
  - It is virtually impossible to find two messages with the same digest (collision)
- Two instances: HASH1 and HASH2



Application benefits

- Used to secure transaction by ensuring
  - Message Integrity (HASH)
  - Message Authentication (HMAC)
- Reduces CPU processing time

Hash peripheral is in charge of efficient computing of message digest.
A digest is a fixed-length value computed from an input message. A digest is unique - it is virtually impossible to find two messages with the same digest. The original message cannot be retrieved from its digest.
Hash digests and Hash-based Message Authentication Code (HMAC) are widely used in communication since they are used to guarantee the integrity and authentication of a transfer.



HASH1 is a secure peripheral (under ETZPC control through ETZPC_DECPROT0 bit 8) while HASH2 is a non secure peripheral.
HASH1 instance can be allocated to:

- The Arm® Cortex®-A7 secure core to be controlled in OP-TEE by the HASH OP-TEE driver or
- The Arm® Cortex® -A7 non-secure core for using in Linux® with Linux Crypto framework

HASH2 instance can be allocated to the Arm® Cortex®-M4 core to be controlled in the STM32Cube MPU Package using the STM32Cube HASH driver.

HASH1 instance is used as boot device to support binary authentication.

Key features

4

- The hash processor supports:
  - Fast computation of the following hash functions:
    - MD5
    - SHA-1
    - SHA-224 and SHA-256
  - Computation of a simple hash digest or keyed-hash message authentication code (HMAC)
  - Automatic byte swapping to comply with big and little endianness
  - Automatic data flow control supporting both direct memory access (DMA) and Master direct memory access controller (MDMA)
  - Management of partial digest computation each time the FIFO is full.
  - Management of final digest computation, including automatic padding
    - Final digest computation is different between HASH1 and HASH2
      - Register HASH_HWCFGR equals 0x1 for HASH1 and 0x0 for HASH2

life.augmented

The hash processor supports widely used hash functions including Message Digest 5 (MD5), Secure Hash Algorithm SHA-1 and the more recent SHA-2 with its 224- and 256-bit digest length versions.
A hash can also be generated with a secrete-key to produce a message authentication code (MAC).
The processor supports bit, byte and half-word swapping. It supports also automatic padding of input data for block alignment.
The processor can be used in conjunction with the DMA for automatic processor feeding.

Digest computation can be either partial (each time FIFO is full) or final (no more bytes to add). The application shall manage the final digest differently depending on the case:

- single DMA transfer (MDMAT bit=”0”)
- multiple DMA transfers (MDMAT bit=”1”)
- Any transfer with MDMA (HASH1 only)

Hash functions

- Block-based algorithms
  - Supported hash functions work on 512-bit blocks of data. The original message is split into subsequent blocks of 512 bits after padding if needed.
    - Update NBLW to define the number of valid bits in last word if it is different from 32-bits
  - Collision robustness increases with the digest length.
    - MD5 and SHA-1 are not qualified as secure digests by NIST

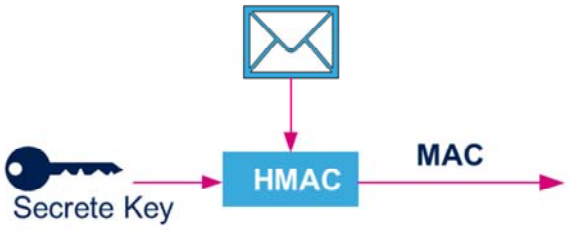
Hash function		Digest (bits)	Strength (collision)	Secure digest (*)
MD5		128	2^64	No
SHA-1		160	2^80	
SHA-2	SHA-224	224	2^112	Yes
	SHA-256	256	2^128	

(*) According to NIST

All supported hash functions work on 512-bit blocks of data. The input message is split as many times as needed to feed the hash processor. Subsequent blocks are computed sequentially.
MD5 is the less robust function with only a 128-bit digest.
The SHA standard has two versions SHA-1 and the more recent SHA-2 with its 224- and 256-bit digest length versions.
MD5 and SHA-1 are not qualified as secure digests according to NIST.

HMAC Keyed-hashing for message authentication code

- HMAC, as defined by IETF RFC2104 and NIST FIPS PUB198-1
  - Ensures authentication of messages in addition to integrity
  - Involves a secret key shared by both sender and receiver
- The algorithm consists of two nested hash operations:
  - HMAC(message) = Hash[((key | pad) ⊕ 0x5C) | Hash(((key | pad) ⊕ 0x36) | message)]
  - Hash function is any of the ones supported by the peripheral



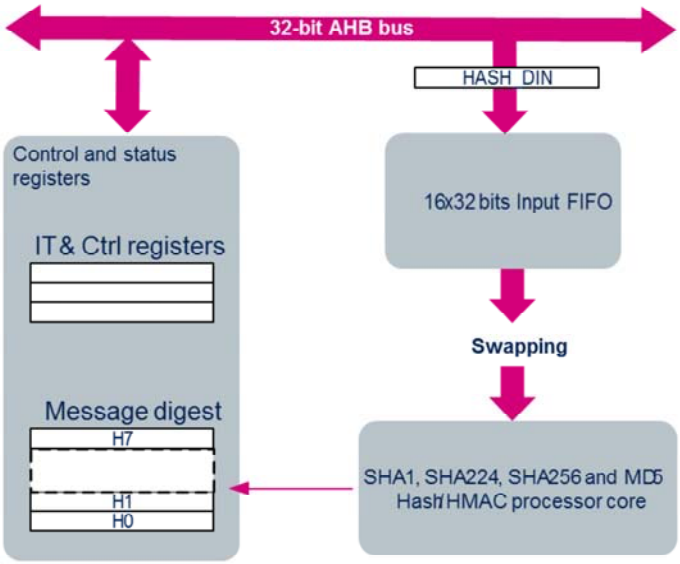
The hash-based message authentication code (HMAC) is used to authenticate messages and verify their integrity. The HMAC function consists of two nested Hash function with a secrete key that is shared by the sender and the receiver. The hash function involved in the HMAC computation can be any one supported by the peripheral: MD5, SHA-1 or SHA-2.

Standard compliance

- This hash processor , suitable for data authentication applications, is compliant with the following standards:
  - FIPS PUB 180-4 (Federal Information Processing Standards Publication 180-4) for Secure Hash Standard (SHA-1, SHA-224 and SHA-256)
  - IETF RFC 1321 (Internet Engineering Task Force Request For Comments number 1321) MD5 Message-Digest algorithm
  - NIST FIPS PUB 198-1 and IETF RFC 2104 for the Keyed-Hash Message Authentication Code (HMAC)

The hash processor complies with the international standards for Secure Hash Algorithms (SHA), Message Digest algorithms (MD5) and for keyed-hash Message Authentication Code (HMAC).

Block diagram



This simplified block diagram of the hash processor shows the basic data flow and control modules.
The hash processor processes 512-bit data blocks and generates digests of up to 256 bits depending on the algorithm.
Input data may be swapped before entering the core unit where they will be processed to generate a simple hash or a message authentication code (MAC).

Interrupts and DMA

Interrupt event	Description
Hash digest calculation completion	Set when a digest becomes ready (the whole message has been processed).
Hash data input ready	Set when the input buffer is ready to get a new block of 512 bits (16 locations are free).

- DMA capability: one channel for input data from memory
  - Single and fixed burst requests of 4 words are supported.
  - The hash processor initiates a DMA request (HASH_IN) to load data of one block of 512 bits.
  - DMA complete transfer interrupt can be used for data flow control.

life.augmented

An interrupt in the nested vectored interrupt controller (NVIC) is triggered when a hash digest has been successfully calculated or when the hash processor is ready to accept a new block of data.

In Direct memory access (DMA) mode, requests are generated internally for incoming data. The DMA channel must be configured in Memory-to-peripheral mode with a data size equal to 512 bits. Single or fixed burst requests of four words are supported.

Performance

- Hash performance
  - The computation of an intermediate block (512 bits) of a message is summarized below

	MD5	SHA-1	SHA-224	SHA-256
Processing (cycles)	**66**	**82**	**66**	
Digest size (bits)	128	160	224	256

  - Extra processing is expected as described below:
    - Final digest computation: up to x2.5 versus partial digest computation
    - HMAC nested operations: ~ x2.5 when short key is used (up to x5 when long key is used)
  - This peripheral increases speed and saves more power compared to the software version of the algorithms.

life.augmented

These are the times it takes top process a single block of data depending on the chosen algorithms.
HCLK is the CPU clock and can go as high as 216MHz.
Note that main benefit of using a hardware accelerator is to increase speed and save power compared to a full software implementation of the hash functions.
Compared to the processing of an intermediate block, it can be increased by the factor below:

- 1 to 2.5 for a hash message
- ~2.5 for an HMAC input-key
- 1 to 2.5 for an HMAC message
- ~2.5 for an HMAC output key in case of a short key
- 3.5 to 5 for an HMAC output key in case of a long key

Low-power modes

Mode	Description
Run	Active
Sleep	Optionally disabled in RCC
Stop	Frozen. Peripheral registers content is kept.
LP-Stop	
LPLV-Stop	
Standby	Powered-down. The peripheral must be reinitialized after exiting Standby mode.

life.augmented

Here is an overview of the status of the hash processor in each of the low-power modes.
Hash operations are not possible when the device is in Stop and Standby modes.

Related peripherals

- Refer to these peripherals trainings linked to this peripheral
  - Master direct memory access controller (MDMA) for HASH1
  - Direct memory access controller (DMA) for HASH2
  - Cryptographic processor (CRYP)

life.augmented

This is a list of peripherals related to the hash processor.
Refer to training on the DMA peripheral for information on how to configure the hash channel.
And please refer to CRYP training if you want to know more about cryptographic engines.

References

- For more details and additional information, refer to following:
  - User manual UM0586: STM32 Cryptographic Library

life.augmented

For more details, please refer to this user manual available on our website.