life.augmented

STM32MP13

HASH

Revision 1.0

Hello and welcome to this presentation of the STM32MP13's hash processor.

HASH key features

- The Hash processor supports:
  - Fast computation of the following hash functions
  - Computation of a simple hash digest or a hash-based message authentication code (HMAC)
  - Automatic byte swapping to comply with big and little endianness
  - Automatic padding to complete the input bit string to fit the minimum digest block size
  - Automatic data flow control with support for direct memory access (DMA)
- Hash performances
  - See the table on the right

Hash function		Digest size (bits)
SHA-1		160
SHA-2/ SHA-3	224	224
	256	256
	384	384
	512	512
SHAKE/ RawSHAKE	128	Variable
	256	

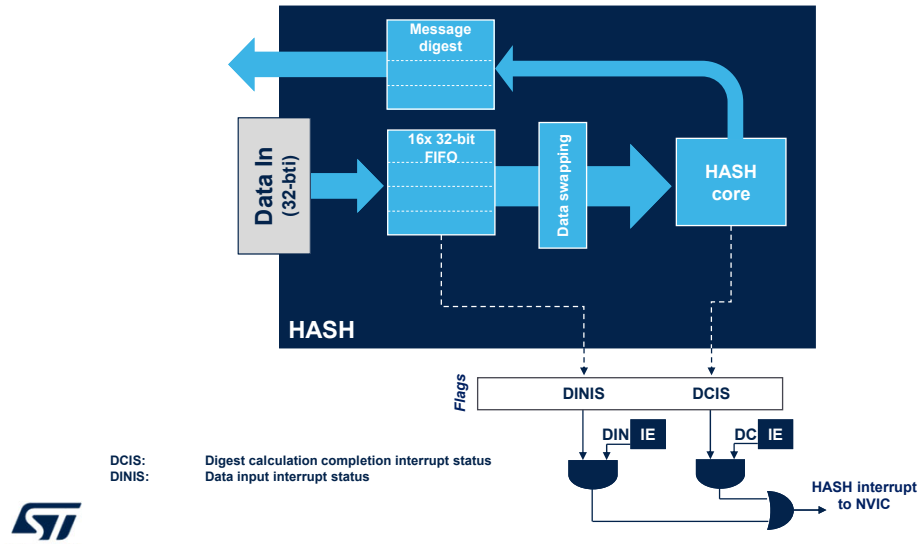
Number of cycles to process a block					
Mode	SHA-1	SHA-2 256	SHA-2 512	SHA-3 256	SHA-3 512
Block size*	64		128	136	72
Normal	82	66	98	58	42
HMAC	Increase time up to: ➢ x2.5 (short key) ➢ x5 (long key)				

* In bytes

The hash processor supports widely used hash functions including Secure Hash Algorithm SHA-1, SHA-2 and the more recent SHA-3 with its SHAKE and RawSHAKE versions.
When a message of any length is provided as an input, the HASH processing core produces a fixed-length output string called a message digest, depending on the algorithm. The size of the message digest is indicated in the upper table.
A hash can also be generated with a secret-key to produce a message authentication code (MAC).
The processor supports bit, byte and half-word swapping. It also supports automatic padding of input data for block alignment.

The hash processor can be used in conjunction with the DMA for automatic feeding of data.
The times it takes to process a single block of data depends on the chosen algorithms.
The bottom table summarizes the time required to process an intermediate block for each mode of operation.
It is increased by a factor of 2.5 or 5 when an HMAC is also generated.

HASH block diagram



This simplified block diagram shows that the hash processor processes data blocks through a FIFO and generates digests once the message has been fully loaded.
Input data may be swapped before entering the core unit.
The hash processor manages two individual maskable interrupt sources in the event of a digest calculation completion (DCIS) or a data input buffer ready (DINIS).

Thank you

life.augmented

Thank you for attending this presentation.
For more details and additional information, refer to the User Manual STM32 Cryptographic Library