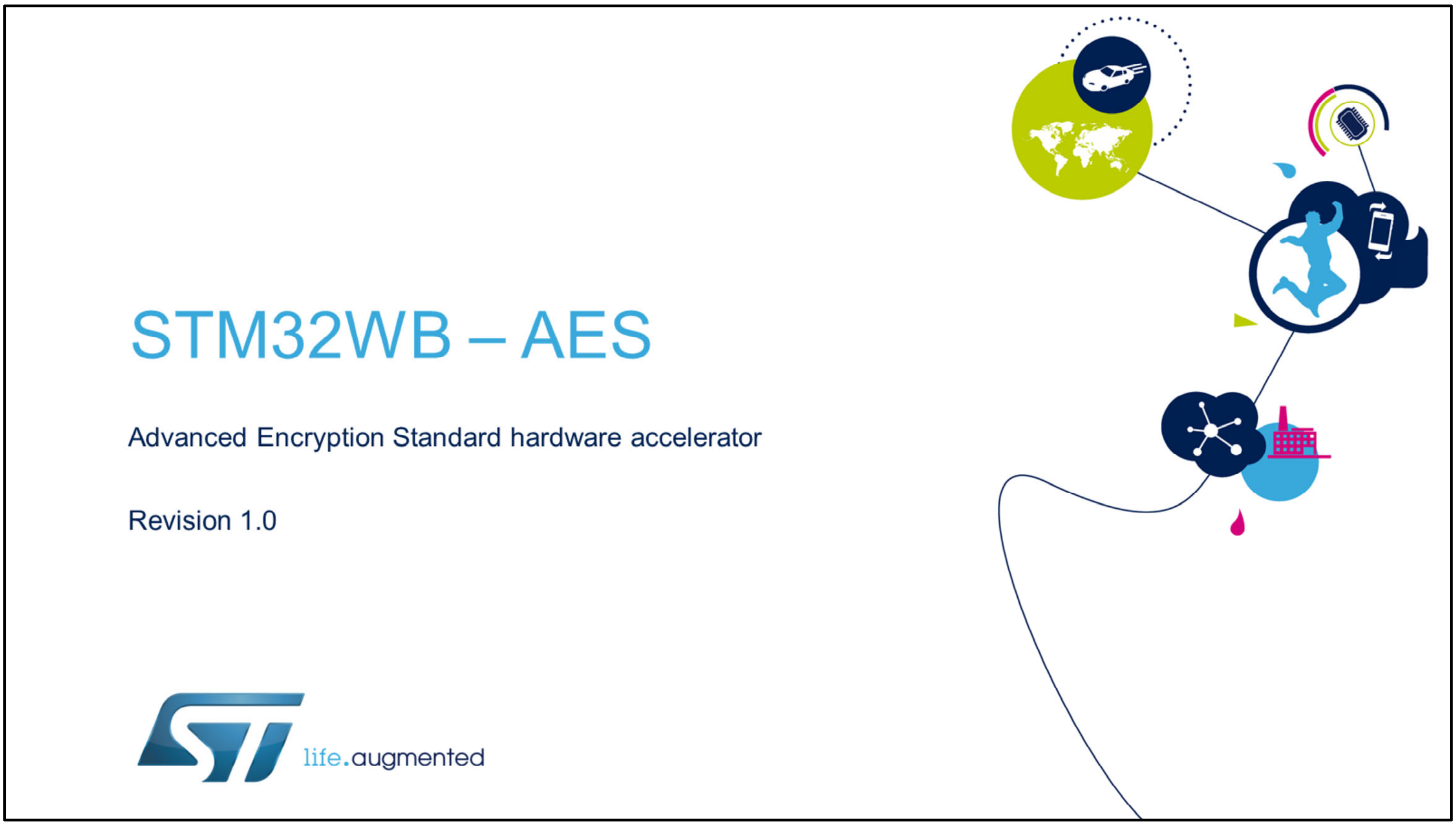


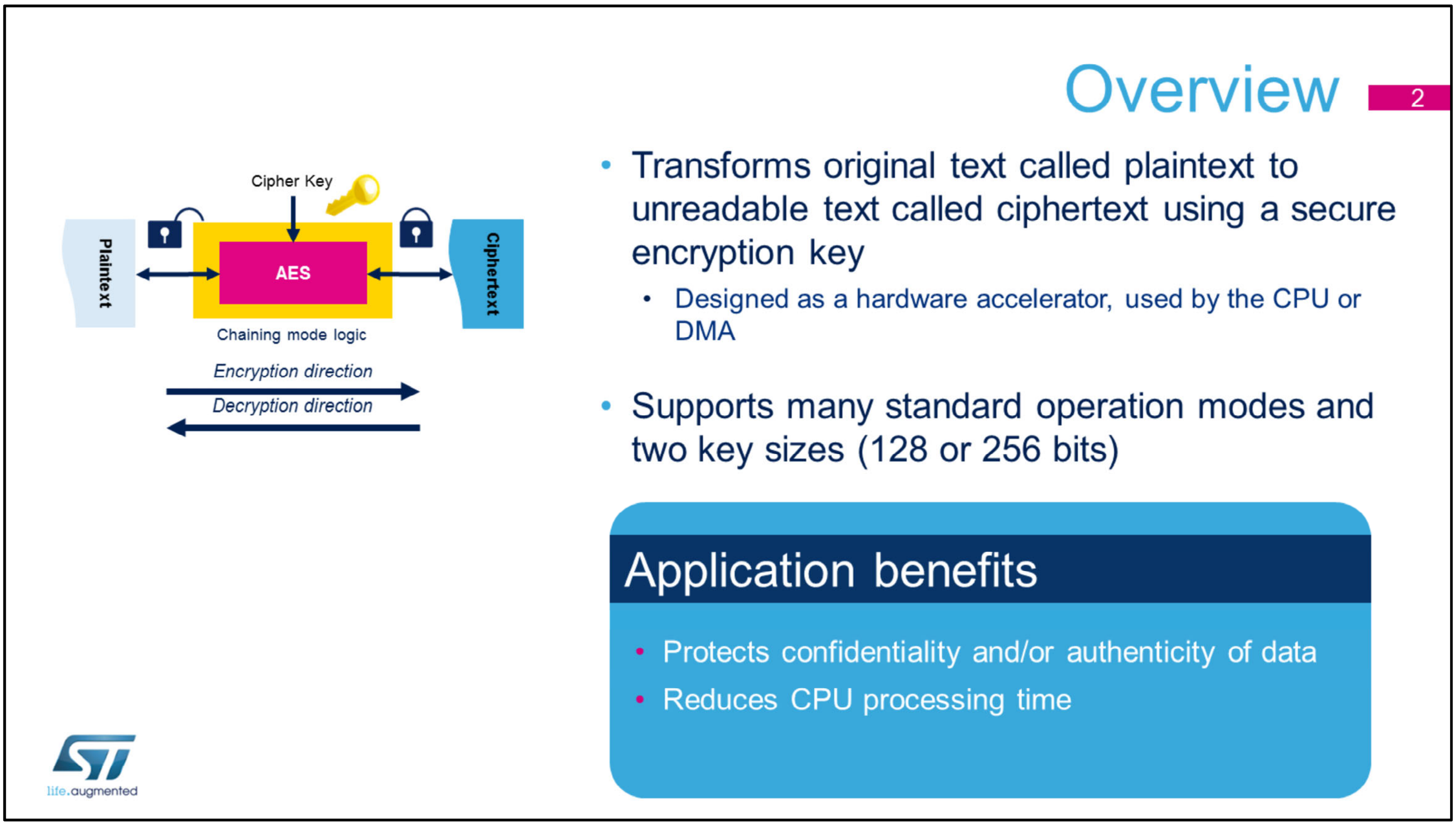
# STM32WB – AES

Advanced Encryption Standard hardware accelerator

Revision 1.0

life.augmented

Hello and welcome to this presentation of the STM32 Advanced Encryption Standard hardware accelerator. It covers the features of the AES interface, which is widely used for cryptographic applications.



Cipher Key
Plaintext
AES
Ciphertext
Chaining mode logic
Encryption direction
Decryption direction
life.augmented

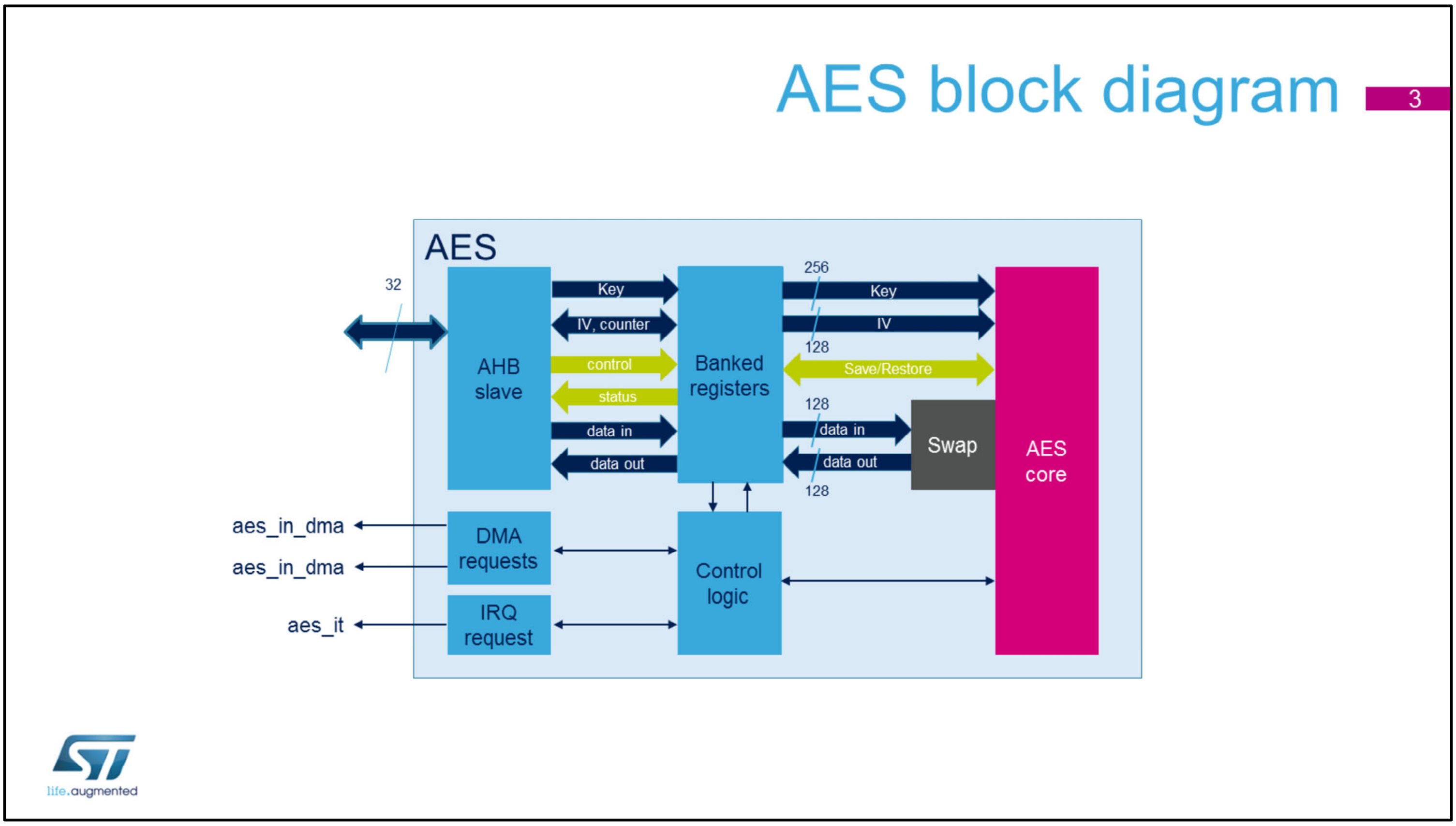
# Overview

- Transforms original text called plaintext to unreadable text called ciphertext using a secure encryption key
  - Designed as a hardware accelerator, used by the CPU or DMA
- Supports many standard operation modes and two key sizes (128 or 256 bits)

## Application benefits

- Protects confidentiality and/or authenticity of data
- Reduces CPU processing time

The AES algorithm is a symmetric block cipher used to encrypt and decrypt information using a secret cryptographic key that is 128 or 256 bits long. Encryption converts data to an unintelligible format called ciphertext; while decrypting converts the ciphertext back into its original format, called plaintext.
The AES peripheral is a NIST FIPS 197 compliant implementation of the AES algorithm, more efficient than a software library in terms of processing time. The AES peripheral supports multiple chaining modes, protecting data confidentiality or data confidentiality + authenticity, depending on the mode.



# AES block diagram

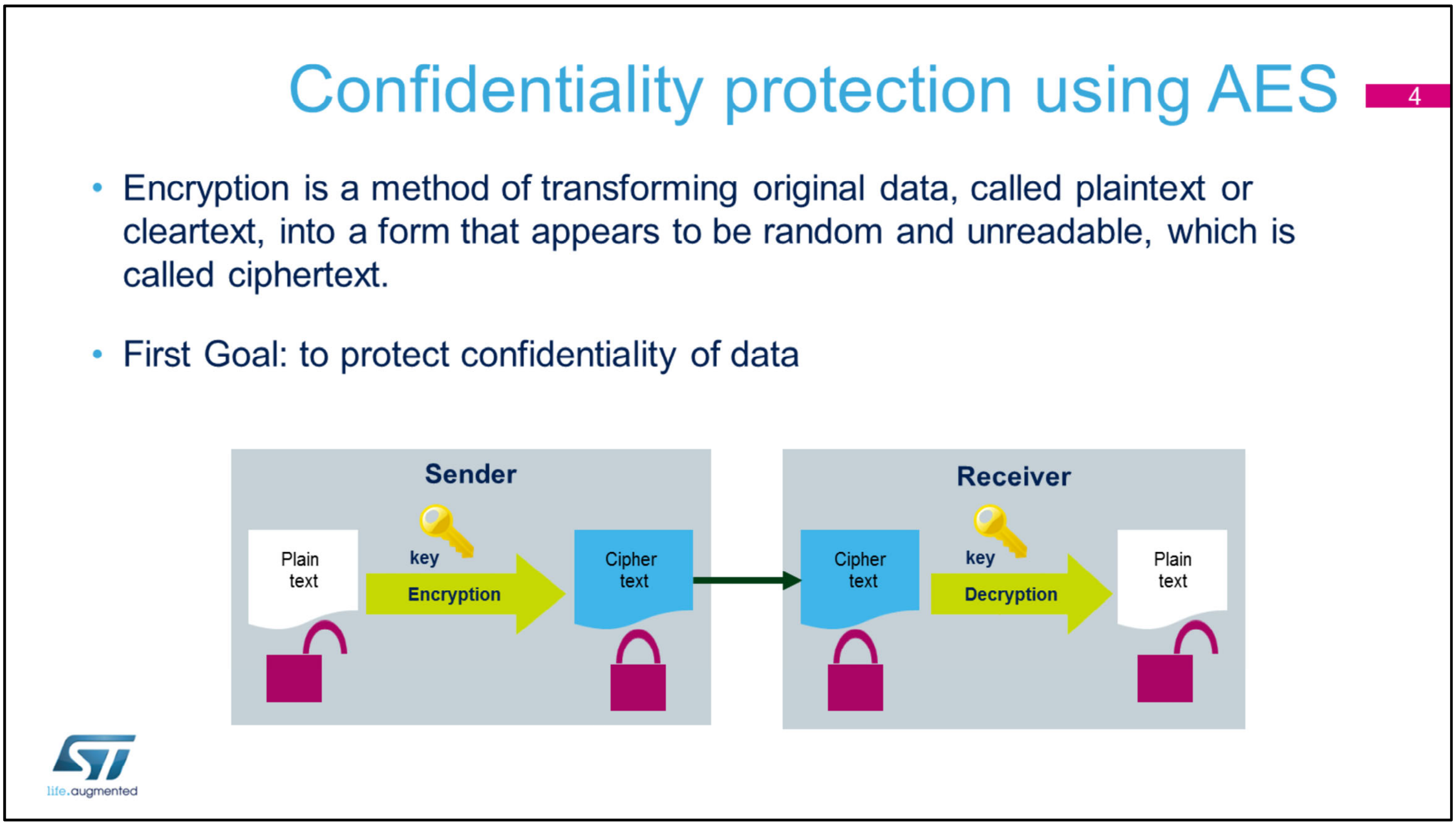
Encrypting plaintext data into ciphertext, and inversely decrypting ciphertext into plaintext, requires intensive computing which represents a huge workload when done entirely by software. The AES hardware accelerator lightens the CPU's workload by performing encryption/decryption operations in the AES core.

The AES block is an AHB slave. Either the CPU passes the data, key and initialization vector to the AES block by writing to memory-mapped registers and gets the result by reading registers, or data movement can be ensured by two DMA channels: one for writing data to the AES, the second to read the result.

Software can suspend a message if the AES needs to process another message with a higher priority, then resume the original message.

The AES core is the unit in charge of processing data. Its logic supports 1-, 8-, 16- or 32-bit data swapping.

Internal data paths are 128 bits wide for data and initialization values and 256 bits for keys. 128-bit keys are also supported.



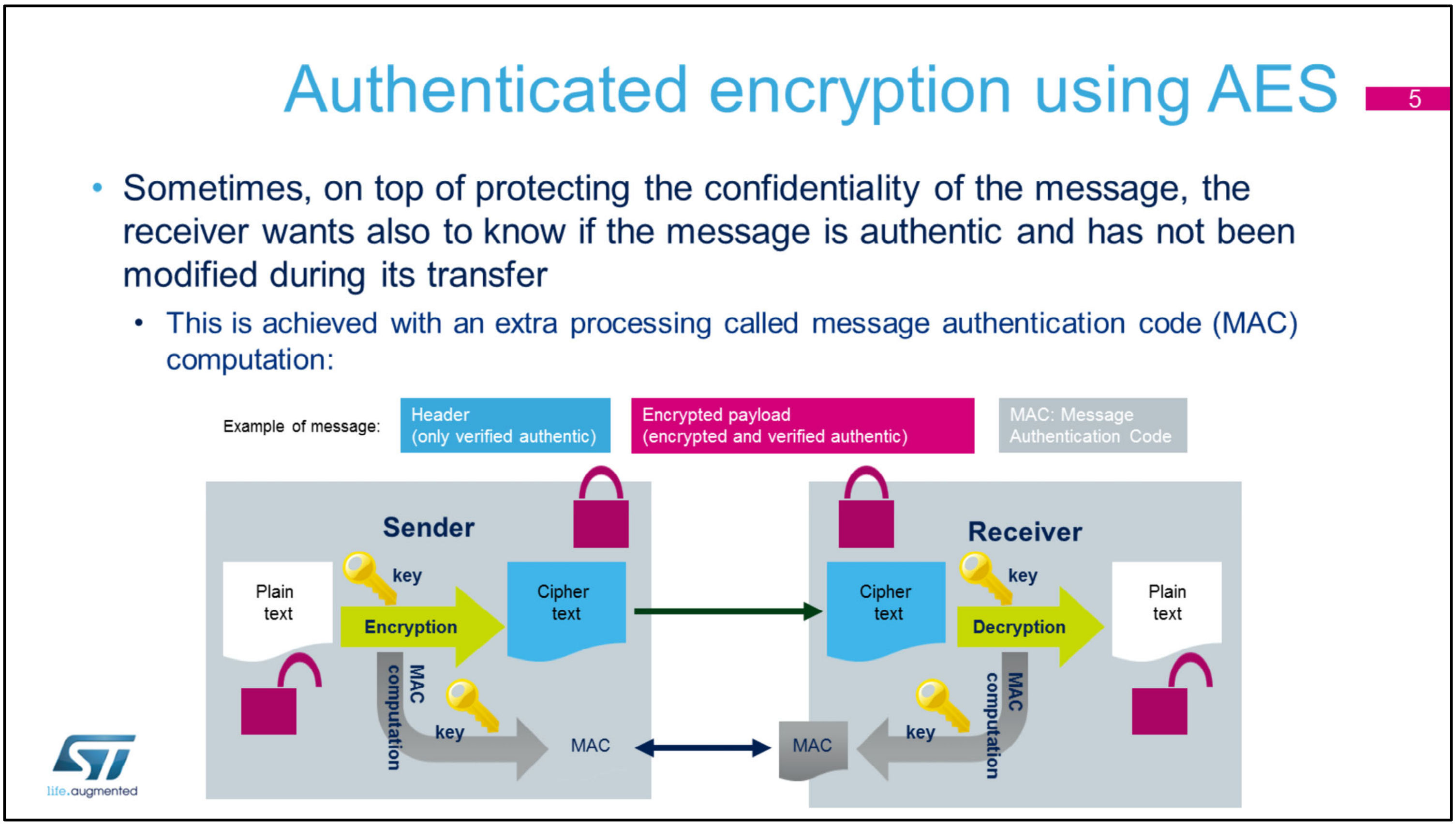
## Confidentiality protection using AES

- Encryption is a method of transforming original data, called plaintext or cleartext, into a form that appears to be random and unreadable, which is called ciphertext.
- First Goal: to protect confidentiality of data

The AES encryption and decryption algorithms are suitable for a variety of applications such as secure networking routers, wireless communications, encrypted data storage including secure smartcards, secure video surveillance systems, secure electronic financial transactions, etc.

The sender encrypts a plaintext message using a secret key. And the receiver decrypts the message using the same secret key.

AES is therefore based on symmetric keys: the same key is used for both encryption and decryption.

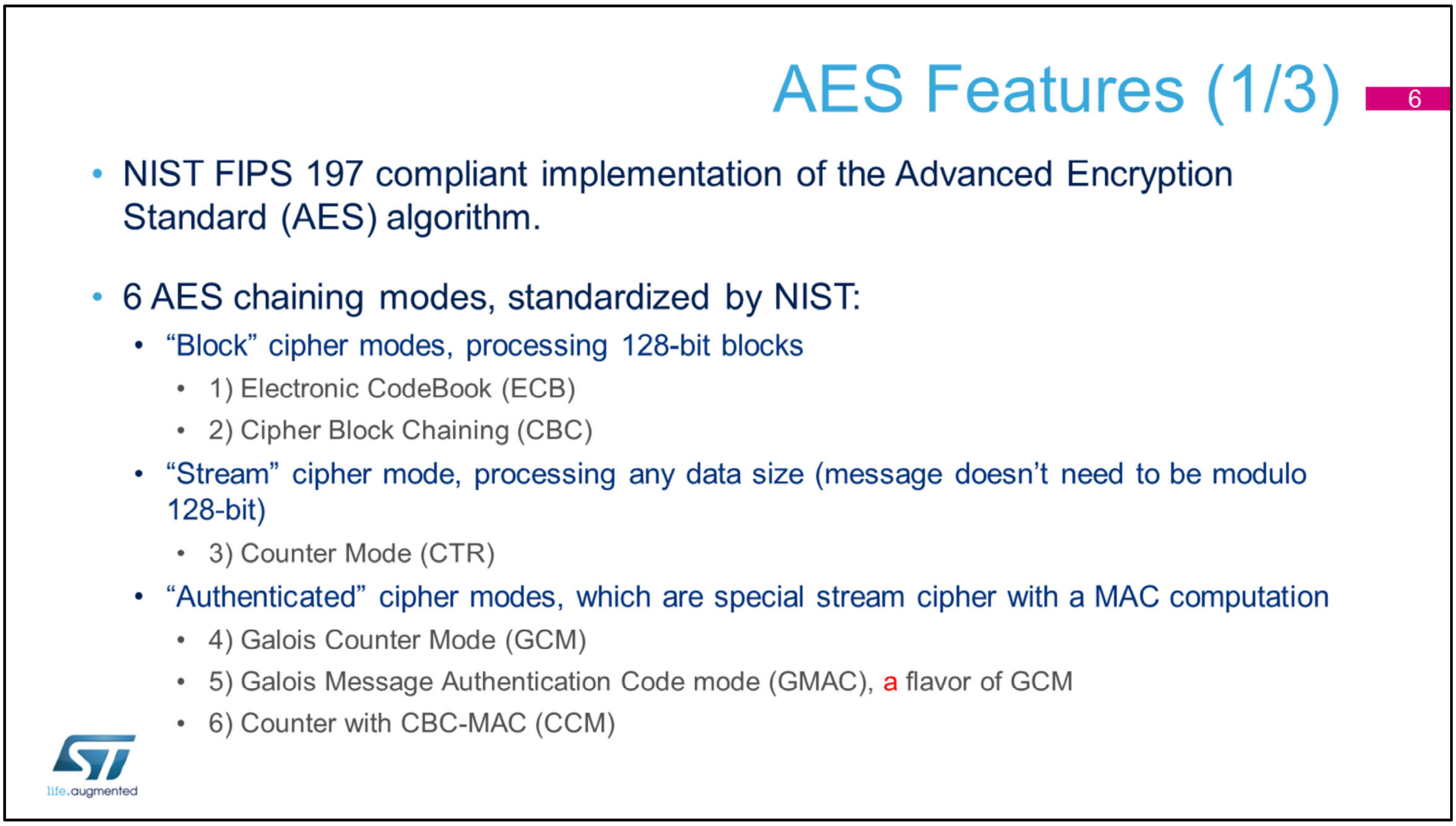


# Authenticated encryption using AES

- Sometimes, on top of protecting the confidentiality of the message, the receiver wants also to know if the message is authentic and has not been modified during its transfer
  - This is achieved with an extra processing called message authentication code (MAC) computation:

| Example of message: | Header<br>(only verified authentic) | Encrypted payload<br>(encrypted and verified authentic) | MAC: Message Authentication Code |
|---|---|---|---|

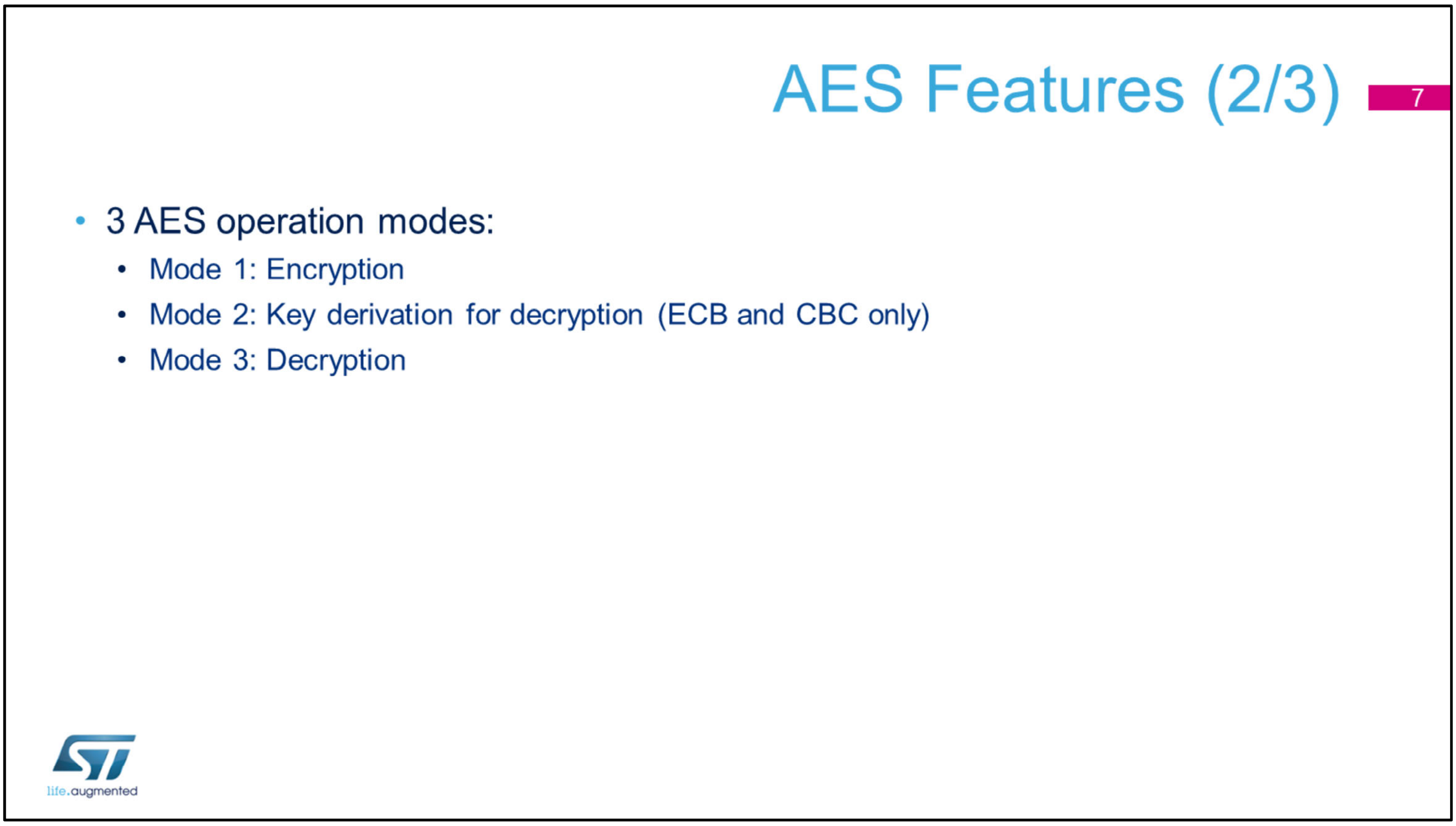
Appending a Message Authentication Code to the cipher text enables the receiver to confirm that the message has been originated by the expected sender.
The AES block is capable of generating the MAC along with data encryption.



# AES Features (1/3)

- NIST FIPS 197 compliant implementation of the Advanced Encryption Standard (AES) algorithm.
- 6 AES chaining modes, standardized by NIST:
  - “Block” cipher modes, processing 128-bit blocks
    - 1) Electronic CodeBook (ECB)
    - 2) Cipher Block Chaining (CBC)
  - “Stream” cipher mode, processing any data size (message doesn’t need to be modulo 128-bit)
    - 3) Counter Mode (CTR)
  - “Authenticated” cipher modes, which are special stream cipher with a MAC computation
    - 4) Galois Counter Mode (GCM)
    - 5) Galois Message Authentication Code mode (GMAC), a flavor of GCM
    - 6) Counter with CBC-MAC (CCM)

The National Institute of Standards and Technology (NIST) develops Federal Information Processing Standards (FIPS) publications specifying cryptographic standards.
Block cipher modes are useful when data to be encrypted has been stored in buffers.
Stream cipher mode is useful to efficiently encrypt or decrypt data at bit level (not block level). This mode does not require key scheduling.
Authenticated modes are used to generate a message authentication code (MAC), along with encrypted data (if enabled).

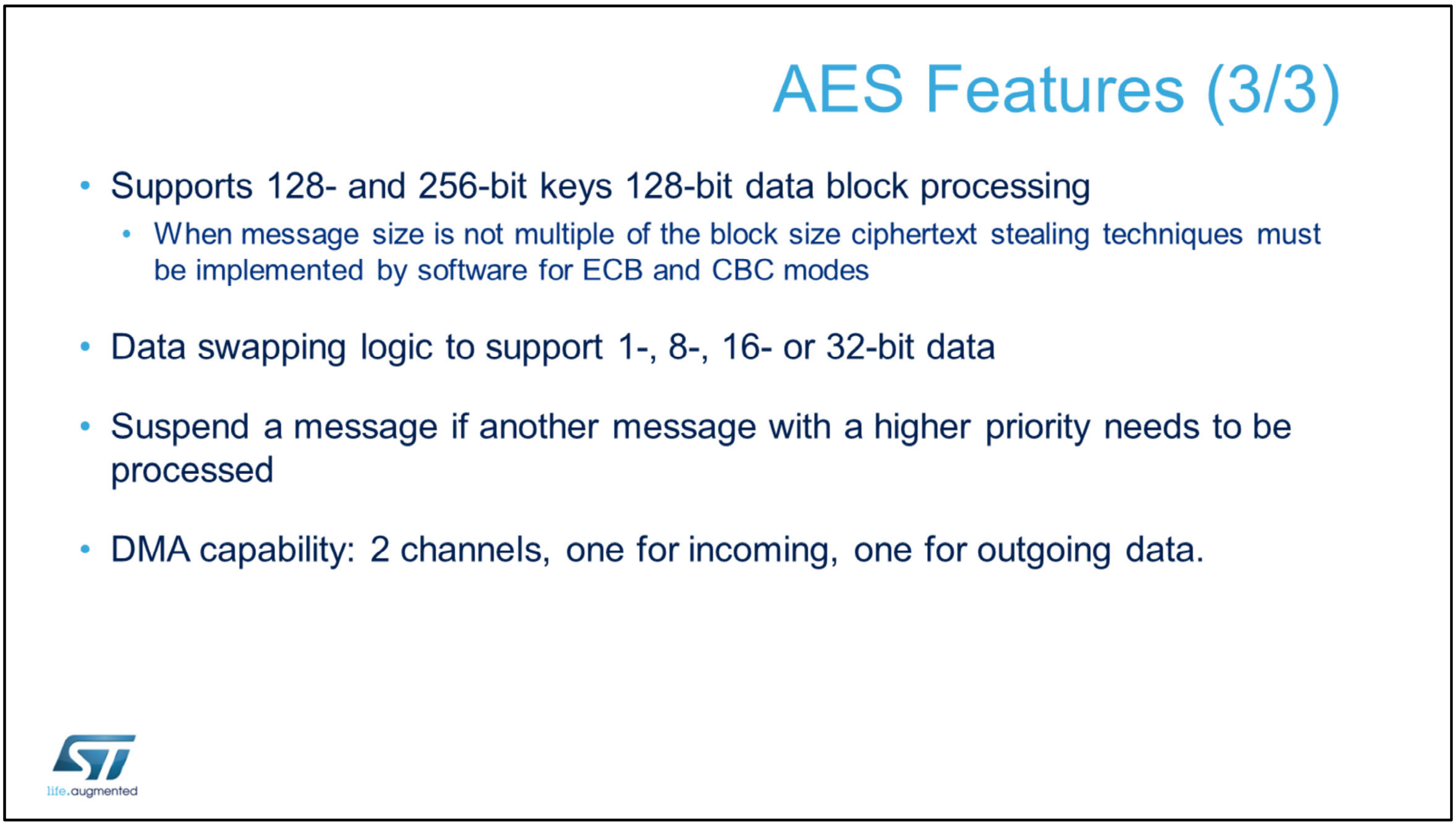


## AES Features (2/3)

- 3 AES operation modes:
  - Mode 1: Encryption
  - Mode 2: Key derivation for decryption (ECB and CBC only)
  - Mode 3: Decryption

The AES features three modes of operation:

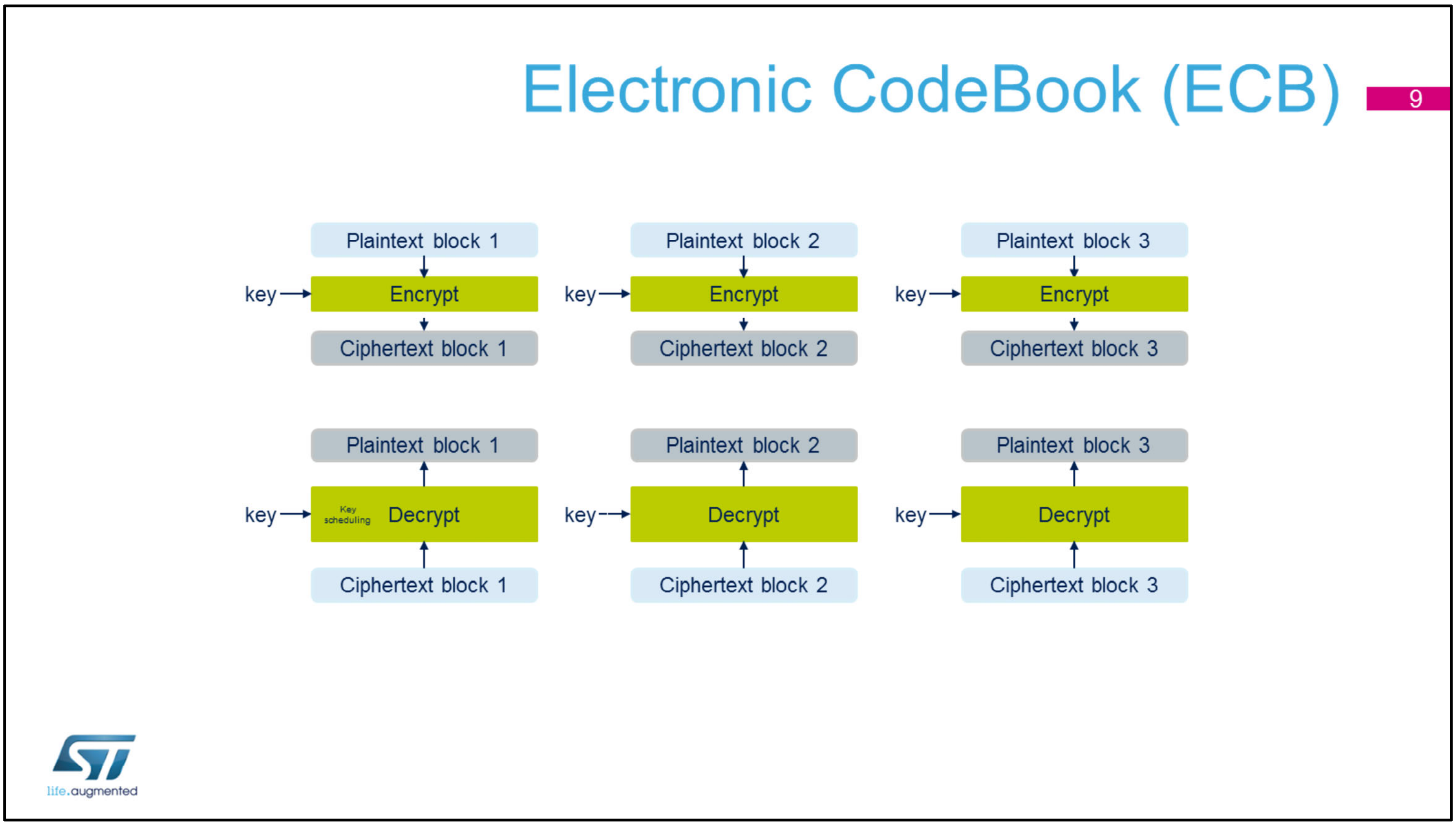
- Mode 1: Plaintext encryption
- Mode 2: Electronic Codebook (ECB) or Cipher Block Chaining (CBC) decryption key derivation. It must be used prior to selecting Mode 3 with ECB or CBC chaining modes. Key derivation derives a new key based on the value stored in the AES Key registers before enabling the AES accelerator.
- Mode 3: Ciphertext decryption



## AES Features (3/3)

- Supports 128- and 256-bit keys 128-bit data block processing
  - When message size is not multiple of the block size ciphertext stealing techniques must be implemented by software for ECB and CBC modes
- Data swapping logic to support 1-, 8-, 16- or 32-bit data
- Suspend a message if another message with a higher priority needs to be processed
- DMA capability: 2 channels, one for incoming, one for outgoing data.

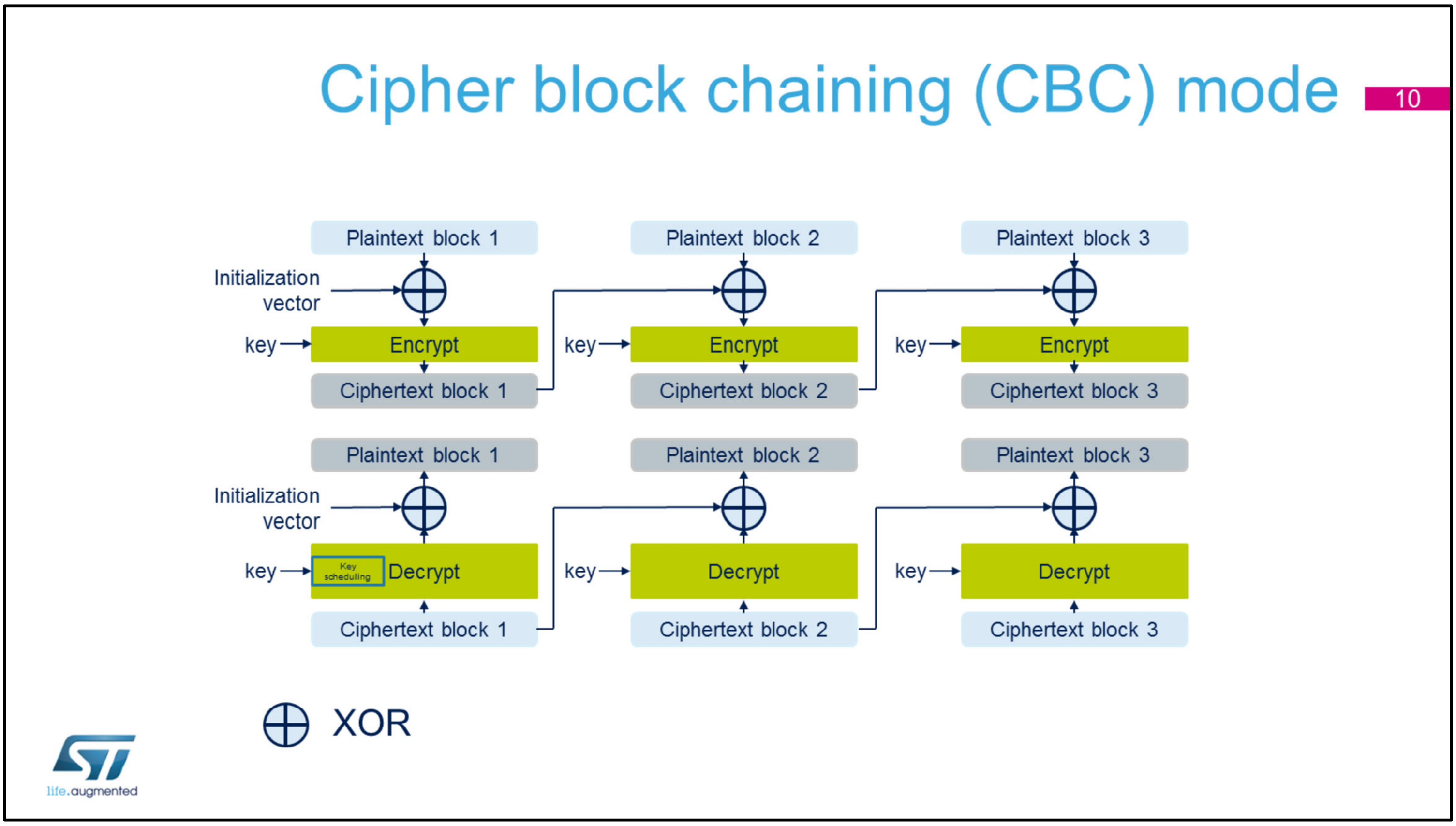
AES keys are 128 or 256 bits long.
Data swapping supports 1-, 8-, 16-, or 32-bit swapping within 128-bit data blocks.
The suspend / resume mechanism enables preemption depending on the priority of the message to handle.
When managing messages of a size that is not a multiple of the block size (128 bits), software must implement ciphertext stealing techniques, such as those described in the Addendum to NIST Special Publication 800-38A.



# Electronic CodeBook (ECB)

9

ECB is the simplest mode of operation. There are no chaining operations and no special initialization stage.
The message is divided into blocks and each block is encrypted or decrypted separately.
For an ECB decryption, a key for the first round of decryption must be derived from the key of the last round of encryption. This is why a complete key schedule of encryption is required before performing the decryption.



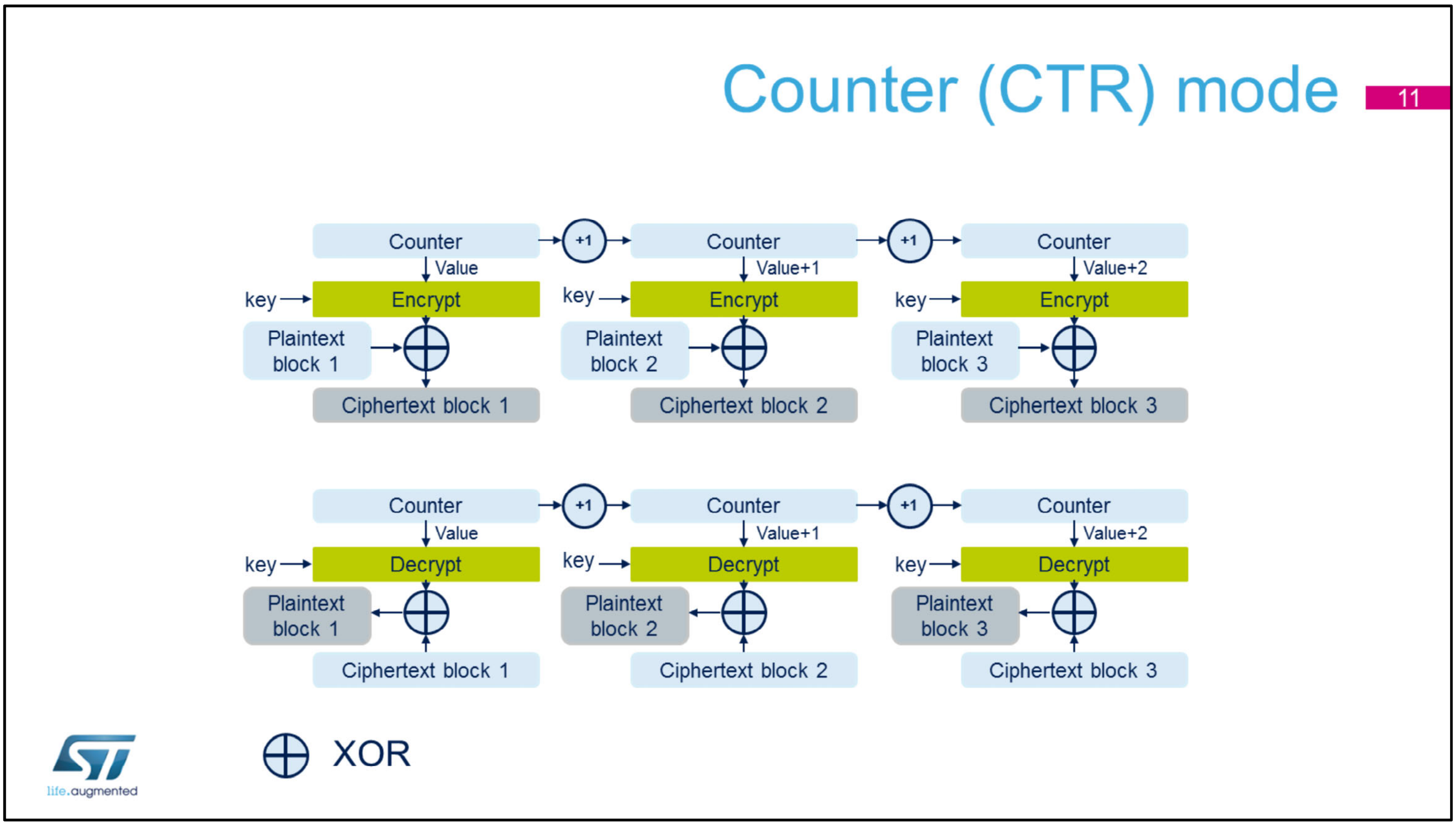
# Cipher block chaining (CBC) mode

⊕ XOR

life.augmented

In CBC mode, each block of plaintext is XORed with the previous ciphertext block before being encrypted. To make each message unique, an initialization vector is used during the first block processing.

For a CBC decryption, a key for the first round of decryption must be derived from the key of the last round of encryption. This is why a complete key schedule of encryption is required before performing the decryption.



# Counter (CTR) mode

11

Counter → +1 → Counter → +1 → Counter

Value | Value+1 | Value+2

key → Encrypt | key → Encrypt | key → Encrypt

Plaintext block 1 | Plaintext block 2 | Plaintext block 3

Ciphertext block 1 | Ciphertext block 2 | Ciphertext block 3

Counter → +1 → Counter → +1 → Counter

Value | Value+1 | Value+2

key → Decrypt | key → Decrypt | key → Decrypt

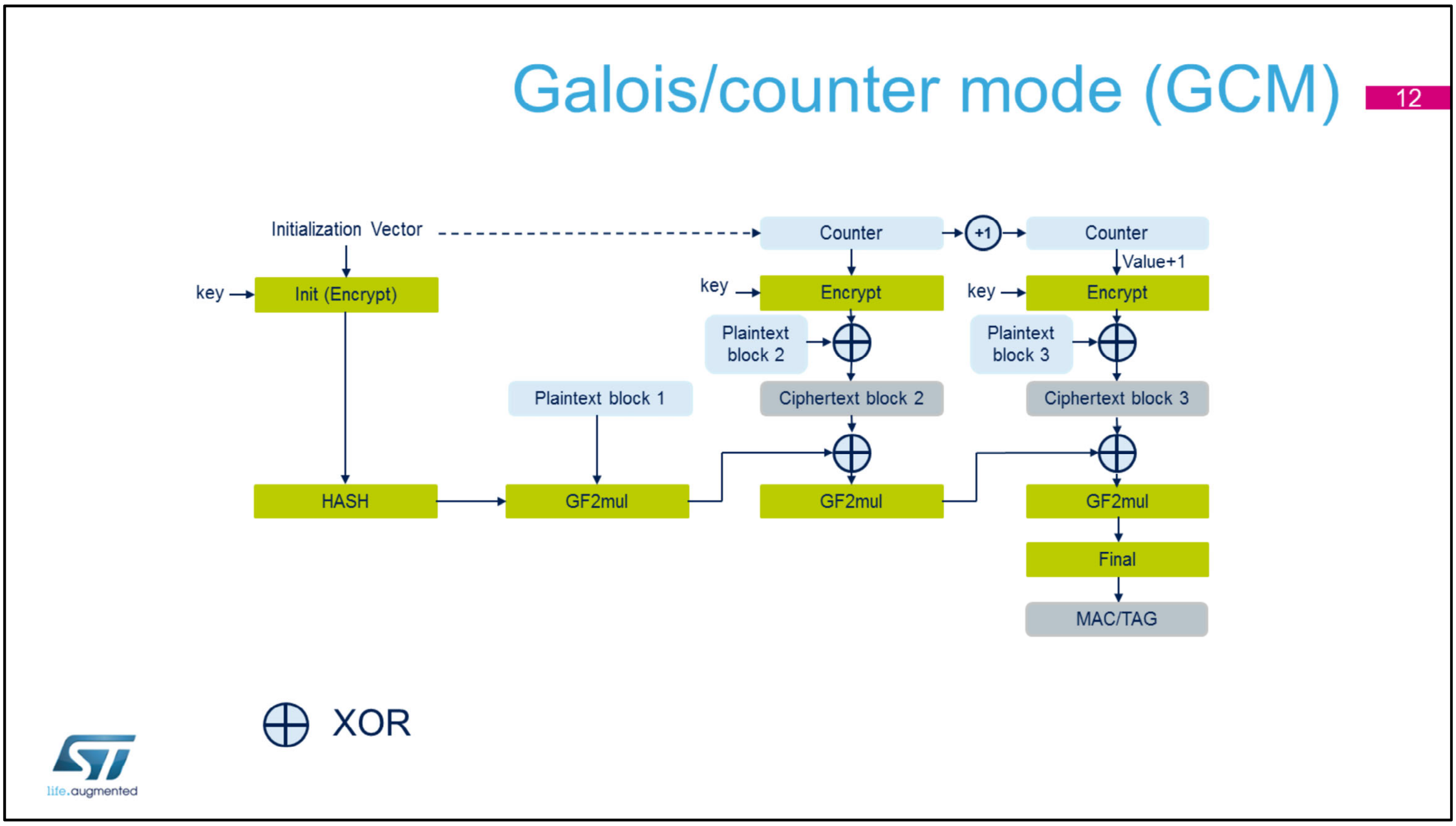
Plaintext block 1 | Plaintext block 2 | Plaintext block 3

Ciphertext block 1 | Ciphertext block 2 | Ciphertext block 3

⊕ XOR

life.augmented

The Counter (CTR) mode uses the AES core to generate a key stream. The keys are then XORed with the plaintext to obtain the ciphertext.
Unlike ECB and CBC modes, no key scheduling is required for the CTR decryption, since in this chaining scheme the AES core is always used in encryption mode for producing the key stream, or counter blocks.



# Galois/counter mode (GCM)

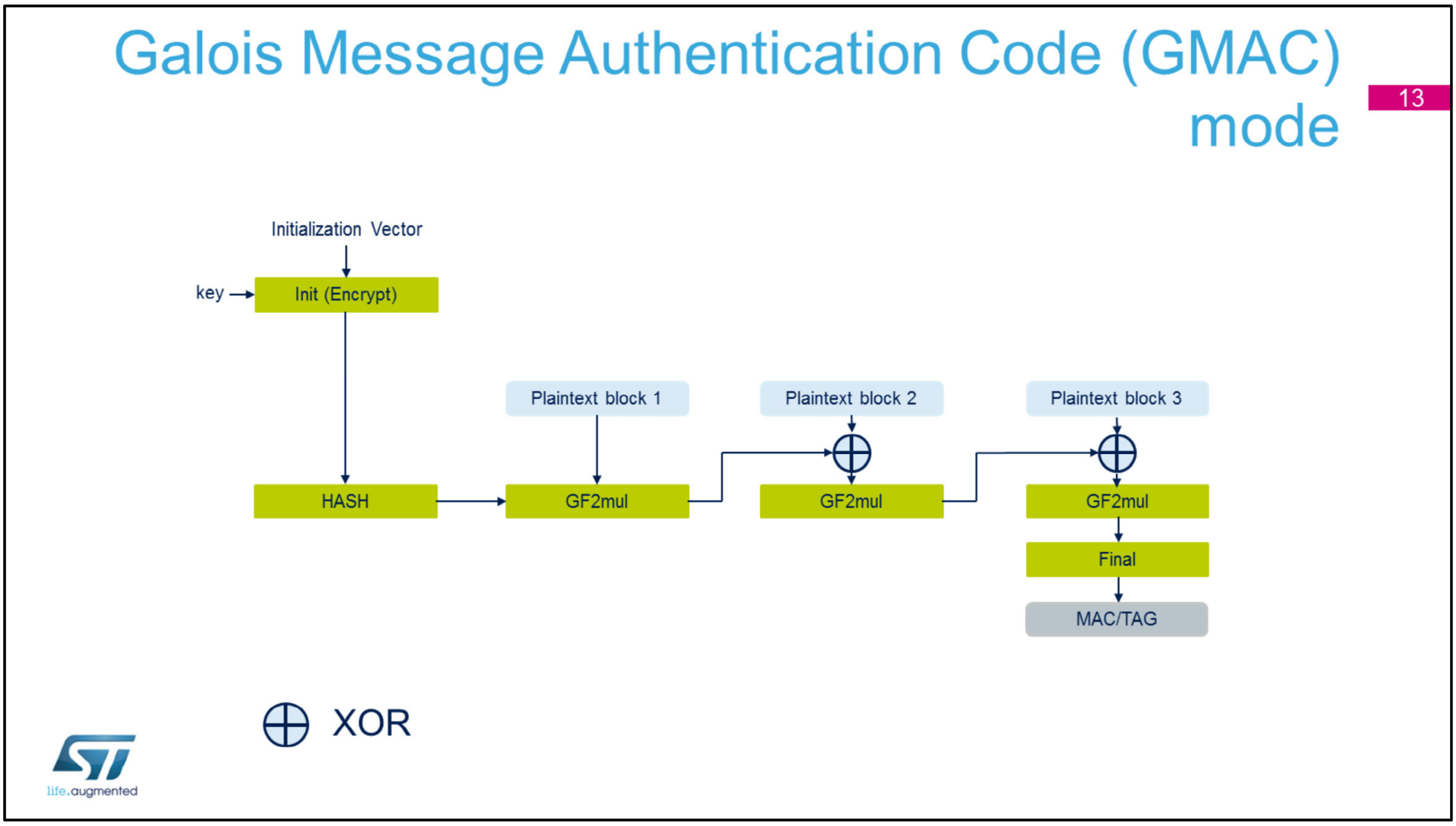
Initialization Vector
key
Init (Encrypt)
HASH
Plaintext block 1
GF2mul
Counter
+1
Counter
Value+1
key
Encrypt
key
Encrypt
Plaintext block 2
Ciphertext block 2
GF2mul
Plaintext block 3
Ciphertext block 3
GF2mul
Final
MAC/TAG

⊕ XOR

life.augmented

In Galois/counter mode (GCM), the plaintext message is encrypted while a message authentication code (MAC) is computed in parallel, thus generating the corresponding ciphertext and its MAC (also known as authentication tag). It is based on the AES's counter mode for confidentiality and uses a multiplier over a fixed finite field for generating the tag. It requires an initialization vector at the beginning.

Part of the GCM message, here block 1, might not be encrypted (it is called the authenticated header).



# Galois Message Authentication Code (GMAC) mode

Initialization Vector

key

Init (Encrypt)

HASH

Plaintext block 1

GF2mul

Plaintext block 2

GF2mul

Plaintext block 3

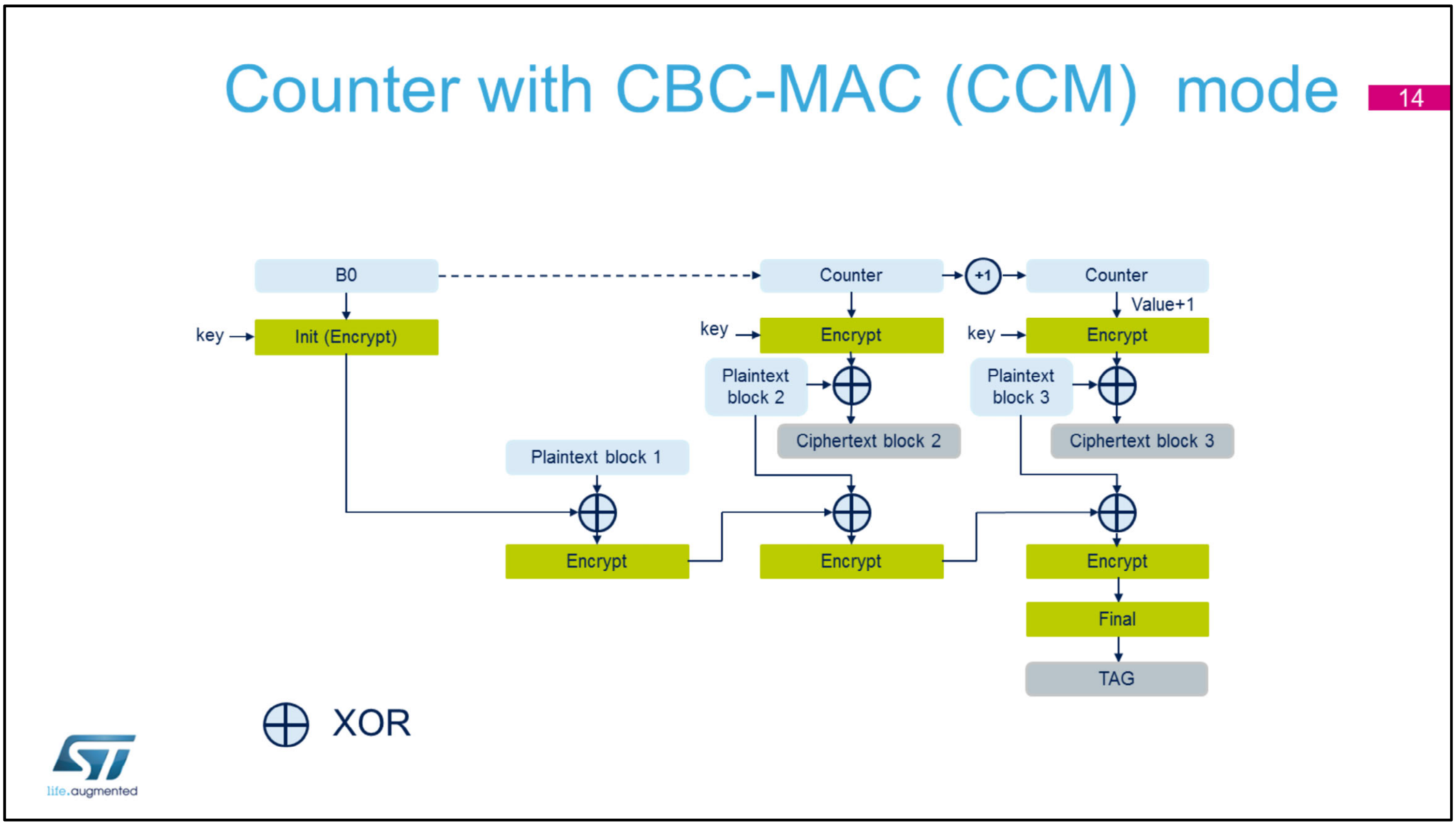
GF2mul

Final

MAC/TAG

⊕ XOR

Galois message authentication code (GMAC) allows authenticating a message and generating the corresponding message authentication code (MAC). GMAC is similar to GCM, except that it is applied to a message that only contains the plaintext authenticated header (so no payload).
All steps and settings are the same as GCM except that the payload phase will not be used.



# Counter with CBC-MAC (CCM) mode

B0
Counter
+1
Counter
Value+1
key
Init (Encrypt)
key
Encrypt
key
Encrypt
Plaintext block 2
Plaintext block 3
Ciphertext block 2
Ciphertext block 3
Plaintext block 1
Encrypt
Encrypt
Encrypt
Final
TAG

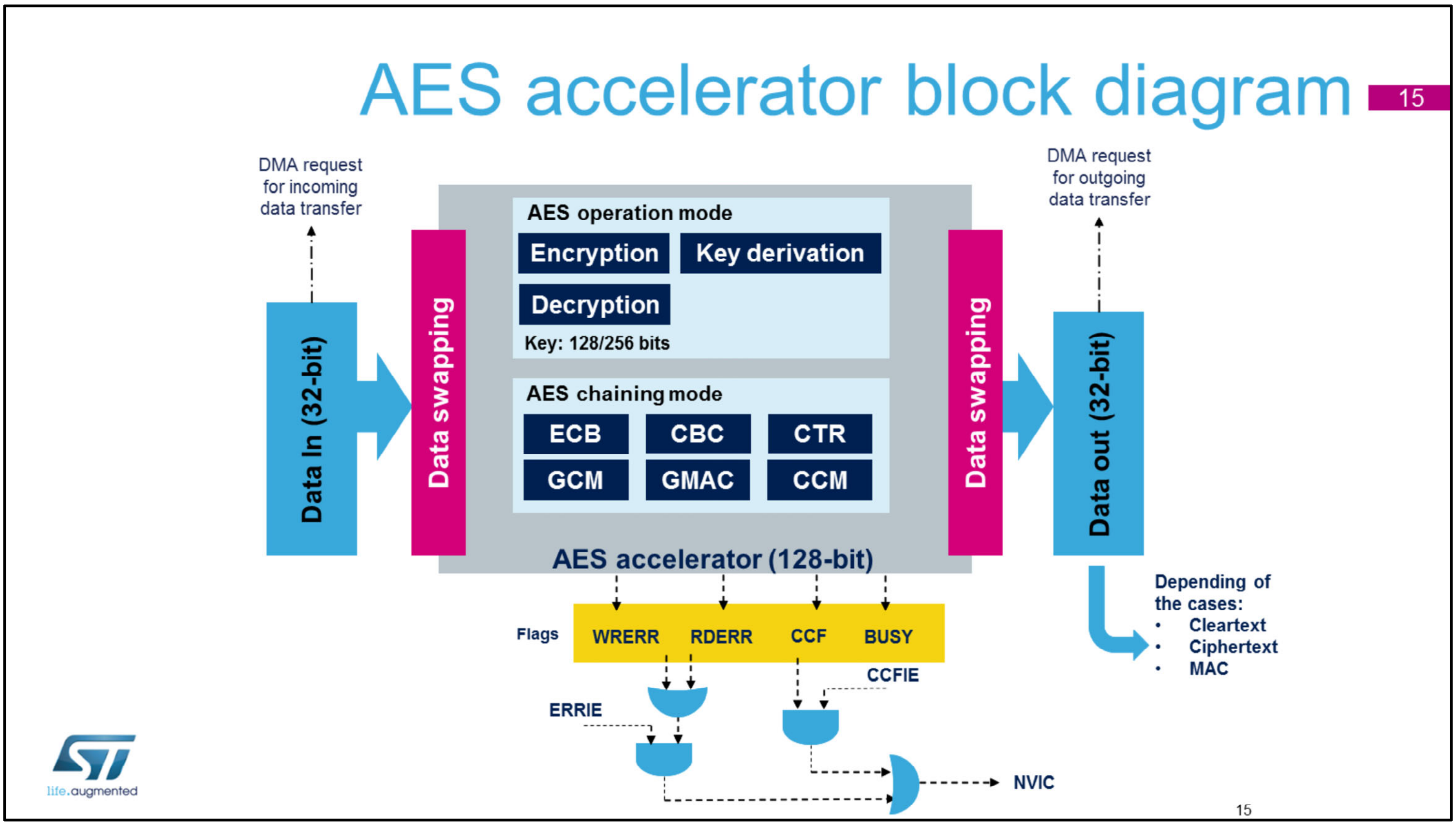
⊕ XOR

In Counter with cipher block chaining-message authentication code (CCM) mode, the payload part of the plaintext message is encrypted while a message authentication code (MAC) is computed for the complete message in parallel, thus generating the corresponding ciphertext and the corresponding MAC (also known as a tag).

CCM mode is based on the AES in Counter mode for confidentiality and it uses CBC for computing the message authentication code. It requires an initial value.

The CCM standard defines specific encoding rules for the first authentication block (called B0 in the standard). In particular, the first block includes flags, a nonce and the payload length expressed in bytes.

Like GCM, the CCM chaining mode could be applied on a message composed only by plaintext authenticated data (that is, only header, no payload), but it is not recommended by NIST. Note that this way of using CCM is not called CMAC (it is not similar to GCM/GMAC). CMAC is a different NIST mode, specified in SP800-38B.



# AES accelerator block diagram

15

DMA request for incoming data transfer

Data In (32-bit)

Data swapping

AES operation mode

Encryption

Key derivation

Decryption

Key: 128/256 bits

AES chaining mode

ECB

CBC

CTR

GCM

GMAC

CCM

AES accelerator (128-bit)

Data swapping

DMA request for outgoing data transfer

Data out (32-bit)

Depending of the cases:

- Cleartext
- Ciphertext
- MAC

Flags

WRERR

RDERR

CCF

BUSY

CCFIE

ERRIE

NVIC

life.augmented

15

This simplified block diagram of the AES accelerator shows the data path from data in on the left to data out on the right.

The AES accelerator processes 128-bit data blocks using an encryption key with a length of either 128 or 256 bits, with or without a data swapping option.

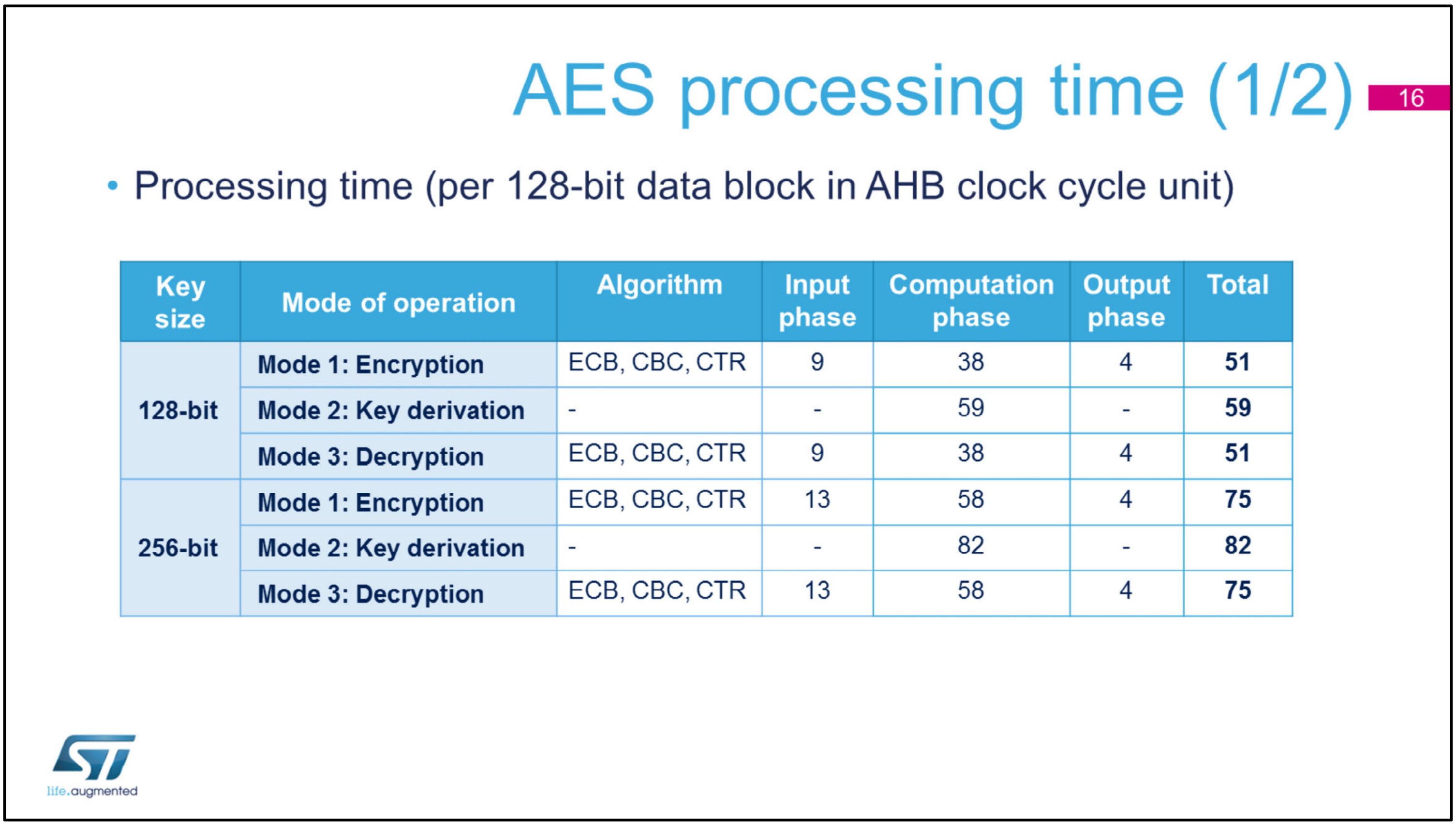
The Error Flags block checks the behavior of the AES accelerator via two different flags:

The Read Error flag (RDERR) is set in the AES Status register when an unexpected read operation is detected during the computation phase or during the input phase.

The Write Error flag (WRERR) is set in the AES Status register when an unexpected write operation is detected during the output phase or during the computation phase.

An interrupt can be generated when one of these two error flags is set if the Error Interrupt Enable (ERRIE) bit

in the AES Control register was previously set.
The Computation Complete flag (CCF) is set by hardware when the computation is complete. An interrupt is generated if the CCF Interrupt Enable bit was previously set.
The Busy flag, used only with GCM mode, indicates that a higher priority message can interrupt the current message during GCM payload phase for encryption mode.

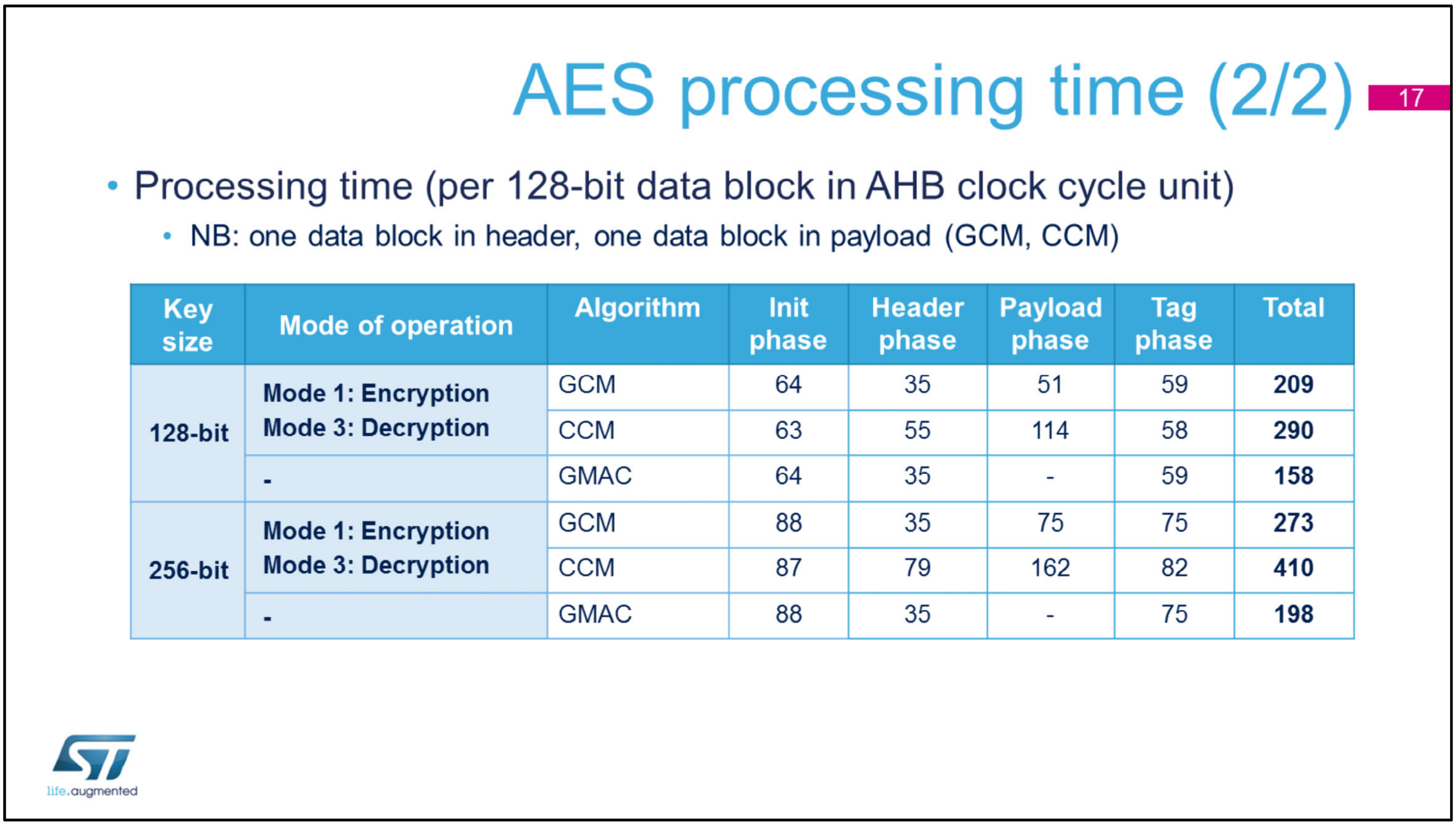


# AES processing time (1/2)

- Processing time (per 128-bit data block in AHB clock cycle unit)

| Key size | Mode of operation | Algorithm | Input phase | Computation phase | Output phase | Total |
|---|---|---|---|---|---|---|
| **128-bit** | **Mode 1: Encryption** | ECB, CBC, CTR | 9 | 38 | 4 | **51** |
| | **Mode 2: Key derivation** | - | - | 59 | - | **59** |
| | **Mode 3: Decryption** | ECB, CBC, CTR | 9 | 38 | 4 | **51** |
| **256-bit** | **Mode 1: Encryption** | ECB, CBC, CTR | 13 | 58 | 4 | **75** |
| | **Mode 2: Key derivation** | - | - | 82 | - | **82** |
| | **Mode 3: Decryption** | ECB, CBC, CTR | 13 | 58 | 4 | **75** |

Here are the processing times for different key sizes and algorithms.



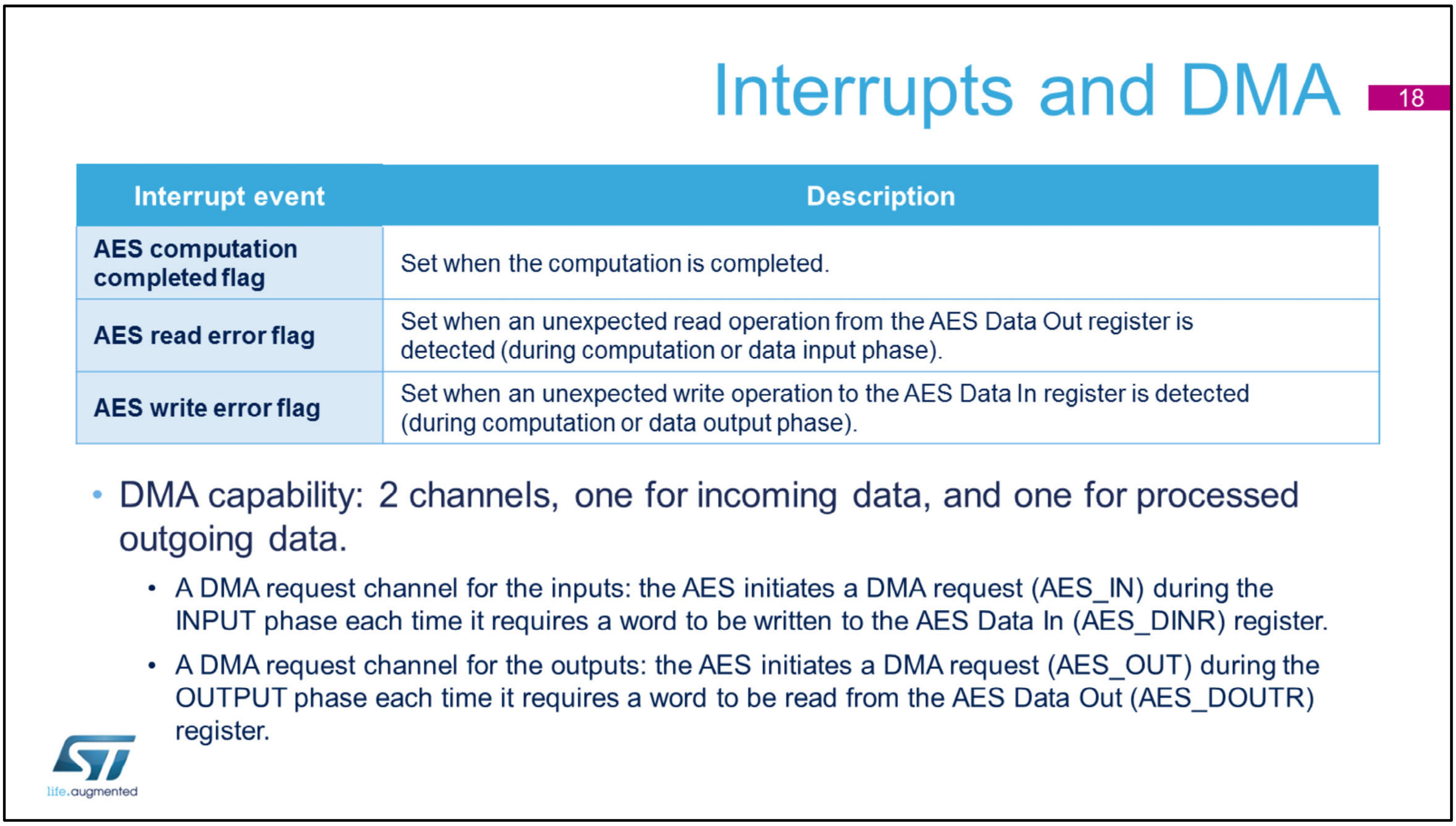
# AES processing time (2/2)

- Processing time (per 128-bit data block in AHB clock cycle unit)
  - NB: one data block in header, one data block in payload (GCM, CCM)

| Key size | Mode of operation | Algorithm | Init phase | Header phase | Payload phase | Tag phase | Total |
|---|---|---|---|---|---|---|---|
| **128-bit** | **Mode 1: Encryption**<br>**Mode 3: Decryption** | GCM | 64 | 35 | 51 | 59 | **209** |
| | | CCM | 63 | 55 | 114 | 58 | **290** |
| | **-** | GMAC | 64 | 35 | - | 59 | **158** |
| **256-bit** | **Mode 1: Encryption**<br>**Mode 3: Decryption** | GCM | 88 | 35 | 75 | 75 | **273** |
| | | CCM | 87 | 79 | 162 | 82 | **410** |
| | **-** | GMAC | 88 | 35 | - | 75 | **198** |

life.augmented

Here are the processing times for different key sizes and algorithms.



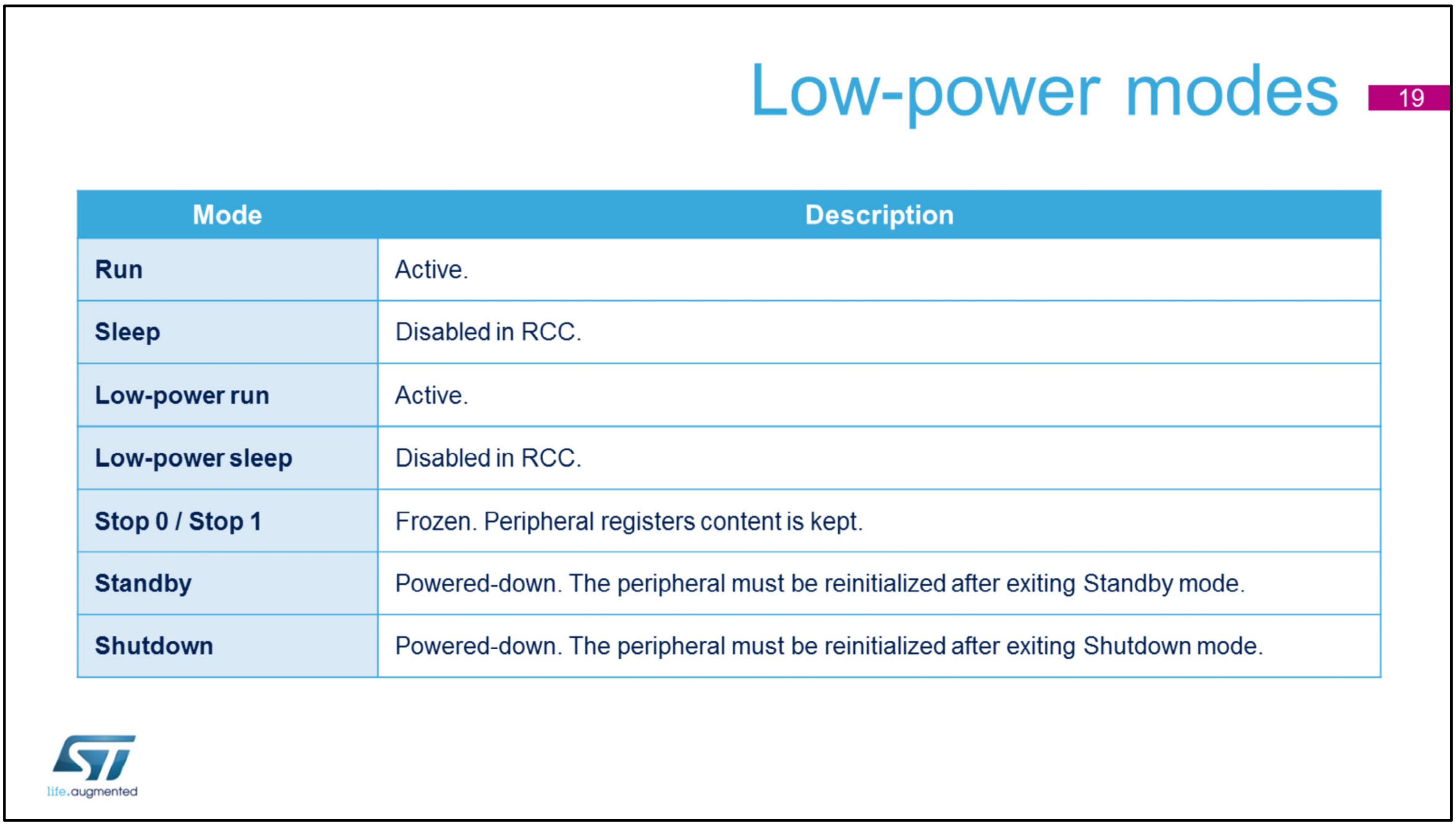
# Interrupts and DMA

| Interrupt event | Description |
|---|---|
| **AES computation completed flag** | Set when the computation is completed. |
| **AES read error flag** | Set when an unexpected read operation from the AES Data Out register is detected (during computation or data input phase). |
| **AES write error flag** | Set when an unexpected write operation to the AES Data In register is detected (during computation or data output phase). |

- DMA capability: 2 channels, one for incoming data, and one for processed outgoing data.
  - A DMA request channel for the inputs: the AES initiates a DMA request (AES_IN) during the INPUT phase each time it requires a word to be written to the AES Data In (AES_DINR) register.
  - A DMA request channel for the outputs: the AES initiates a DMA request (AES_OUT) during the OUTPUT phase each time it requires a word to be read from the AES Data Out (AES_DOUTR) register.

Here is a summary of the events able to trigger an interrupt in the nested vectored interrupt controller: AES computation completed, AES read error, and AES write error.

Direct memory access requests are generated internally for both incoming and outgoing data. The DMA channel must be configured in Memory-to-peripheral or Peripheral-to-memory mode with a data size equal to 32 bits.

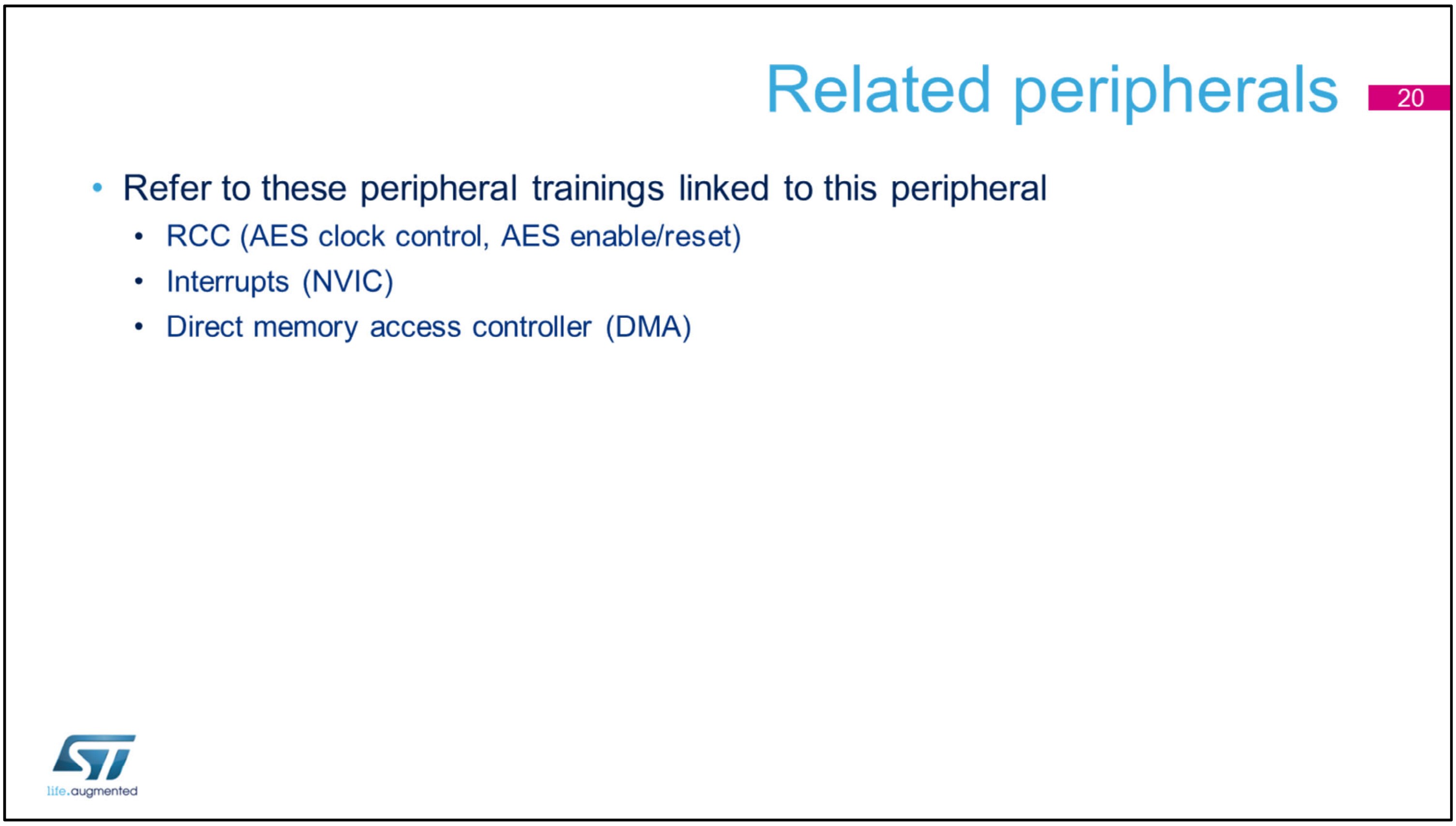


# Low-power modes

| Mode | Description |
|---|---|
| **Run** | Active. |
| **Sleep** | Disabled in RCC. |
| **Low-power run** | Active. |
| **Low-power sleep** | Disabled in RCC. |
| **Stop 0 / Stop 1** | Frozen. Peripheral registers content is kept. |
| **Standby** | Powered-down. The peripheral must be reinitialized after exiting Standby mode. |
| **Shutdown** | Powered-down. The peripheral must be reinitialized after exiting Shutdown mode. |

life.augmented

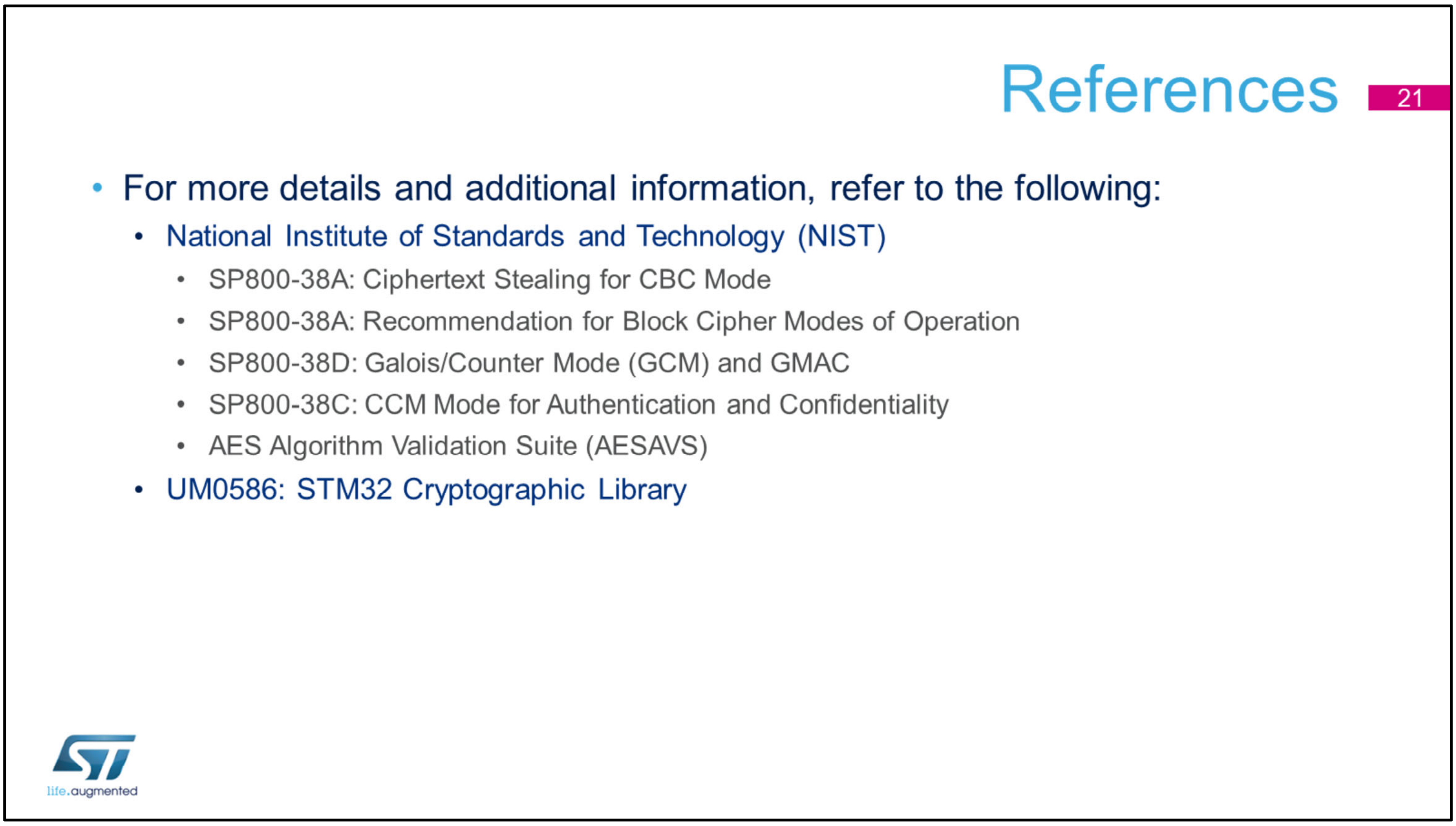
Here is an overview of the status of the AES accelerator in each of the low-power modes.
AES operations are not possible when the device is in Stop mode.



## Related peripherals

- Refer to these peripheral trainings linked to this peripheral
  - RCC (AES clock control, AES enable/reset)
  - Interrupts (NVIC)
  - Direct memory access controller (DMA)

This is a list of peripherals related to the AES accelerator. Please refer to these peripheral trainings for more information if needed.



## References

- For more details and additional information, refer to the following:
  - National Institute of Standards and Technology (NIST)
    - SP800-38A: Ciphertext Stealing for CBC Mode
    - SP800-38A: Recommendation for Block Cipher Modes of Operation
    - SP800-38D: Galois/Counter Mode (GCM) and GMAC
    - SP800-38C: CCM Mode for Authentication and Confidentiality
    - AES Algorithm Validation Suite (AESAVS)
  - UM0586: STM32 Cryptographic Library

For more details, please refer to these application notes and user manuals available on our website.