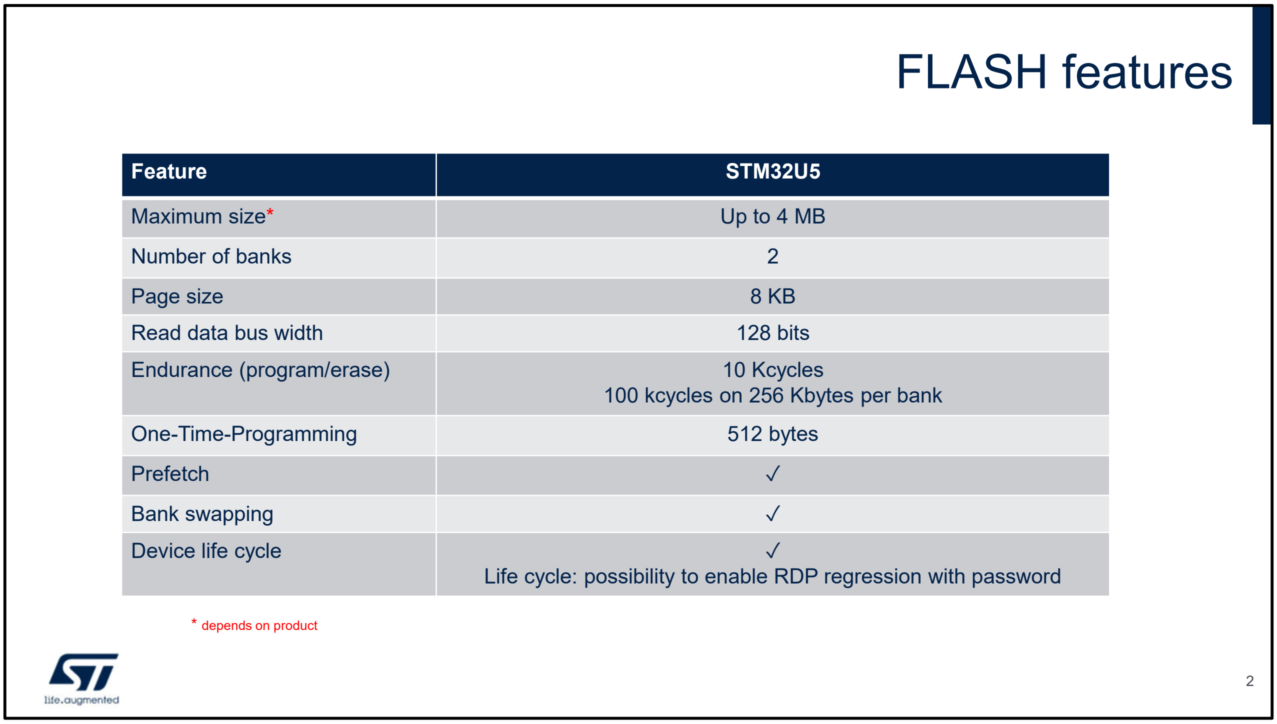
life.augmented

STM32U5

Flash

Hello, and welcome to this presentation of the embedded Flash memory which is included in all products of the STM32U5 microcontroller Series.



FLASH features

Feature	STM32U5
Maximum size*	Up to 4 MB
Number of banks	2
Page size	8 KB
Read data bus width	128 bits
Endurance (program/erase)	10 Kcycles 100 kcycles on 256 Kbytes per bank
One-Time-Programming	512 bytes
Prefetch	✓
Bank swapping	✓
Device life cycle	✓ Life cycle: possibility to enable RDP regression with password

* depends on product

This table summarizes the features of the flash memory existing in STM32U5. Depending on the product, the flash size can be up to 4MB. It also embeds a one-time-programming area of 512 bytes.

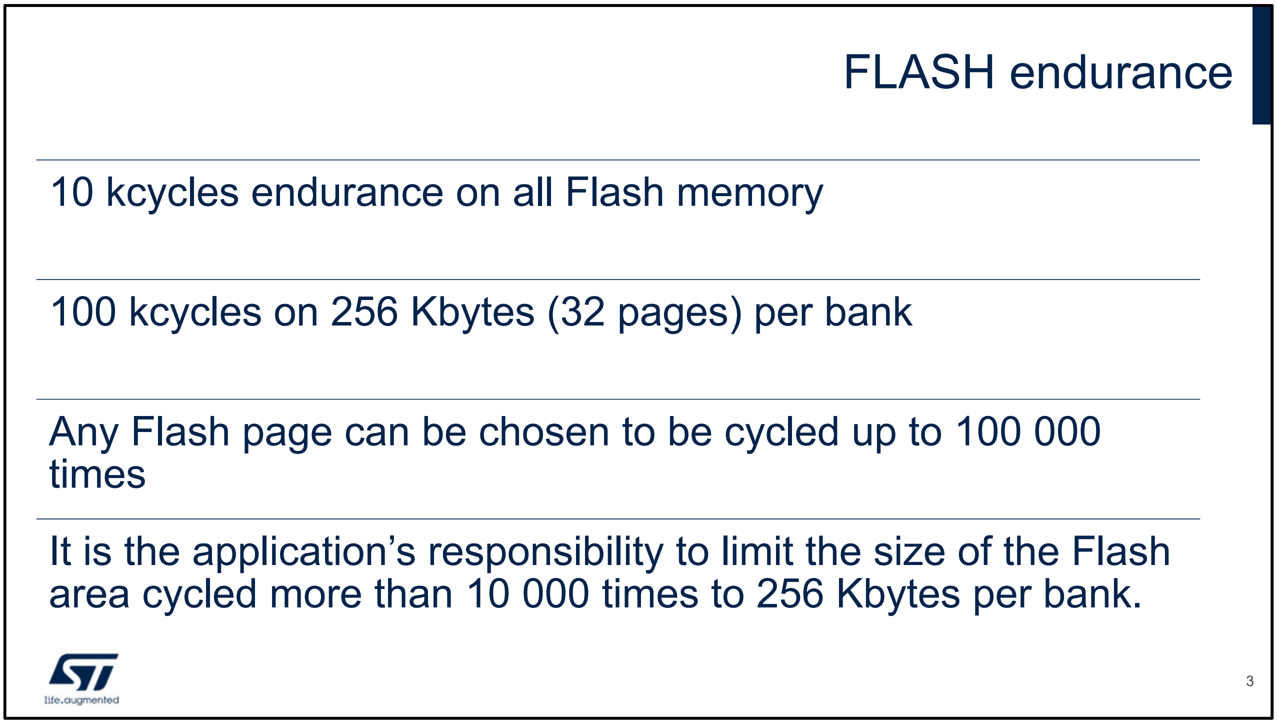
The flash read data bus width is 128-bit. STM32U5 always supports a dual bank architecture. The SWAP-BANK option in the user option bytes is used to swap Bank 1 and Bank 2 addresses.

Note that read-while-write capability (or RWW) is therefore always supported by the STM32U5.

The page size which provides the minimum erase granularity is 8 kilobytes.

STM32U5 has an increased endurance of up to 100 kilocycles on 256 kilobytes per bank.

It also supports a read prefetch unit, that increases the efficiency of Cortex M33 C-AHB bus.
Finally, STM32U5 implements a Flexible life-cycle scheme with readout protection (RDP), including support for product decommissioning even from Level 2, using passwords.



FLASH endurance

10 kcycles endurance on all Flash memory

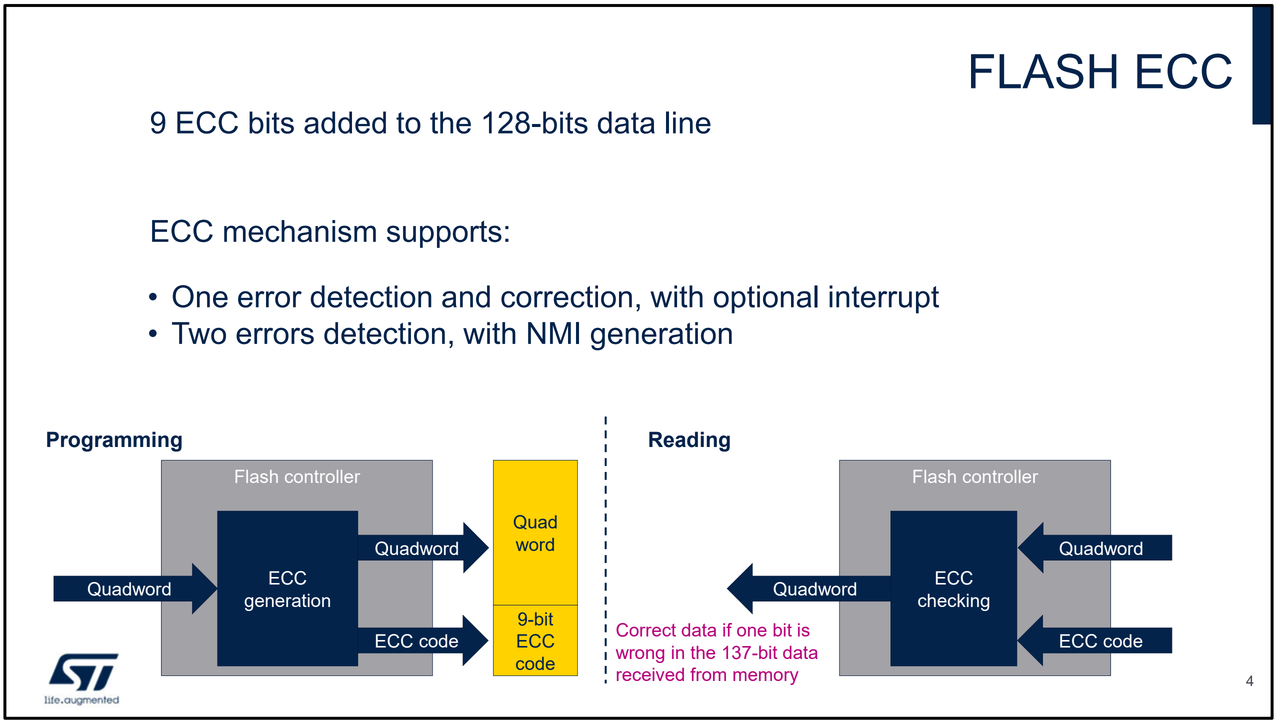
100 kcycles on 256 Kbytes (32 pages) per bank

Any Flash page can be chosen to be cycled up to 100 000 times

It is the application’s responsibility to limit the size of the Flash area cycled more than 10 000 times to 256 Kbytes per bank.

life.augmented
3

Each program / erase operation can degrade the Flash memory cell. After an accumulation of program / erase cycles, memory cells can become non-functional, causing memory errors. Endurance is the maximum number of erase/programming sequences that the Flash memory can support without affecting its reliability. 256 Kbytes (32 pages) per bank feature an increased endurance of 100 kcycles, that can be used for data storage that usually needs more intensive cycling capability than code storage. Any Flash page can be chosen to be cycled more than 10 000 times, up to 100 000 times.
It is the application’s responsibility to limit the size of the Flash area cycled more than 10 000 times to 256 Kbytes per bank.



FLASH ECC

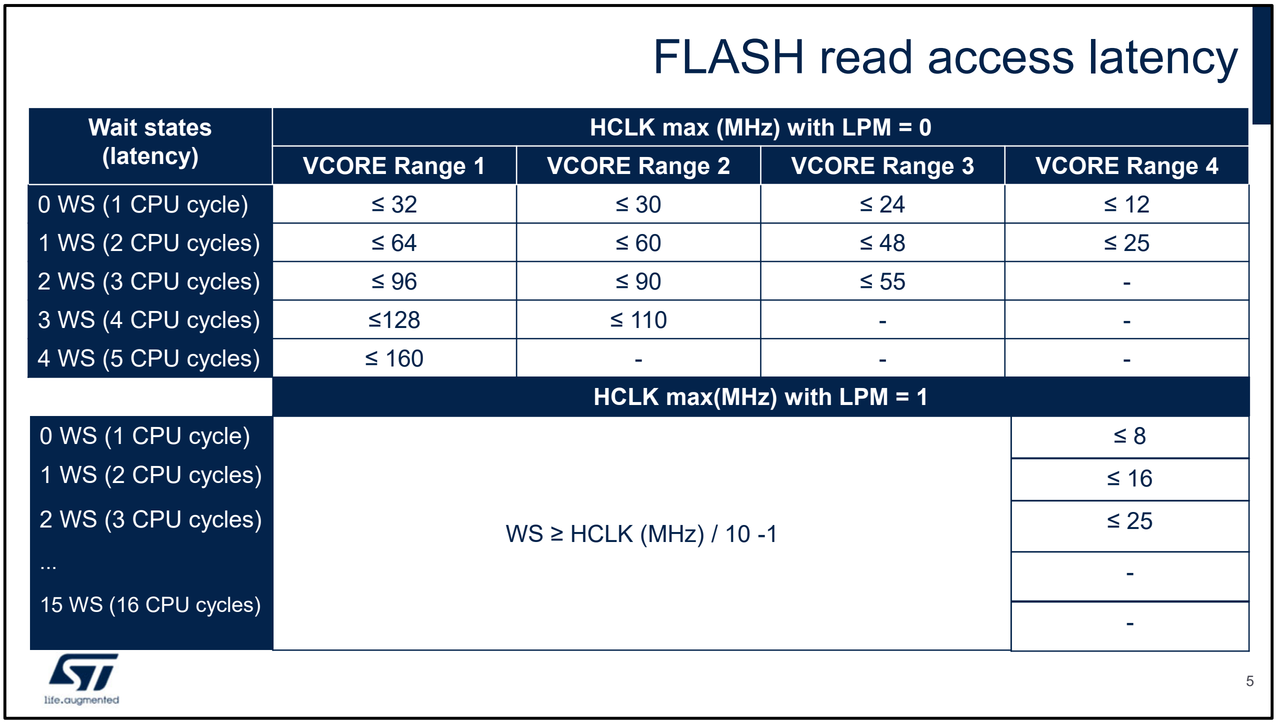
9 ECC bits added to the 128-bits data line

ECC mechanism supports:

- One error detection and correction, with optional interrupt
- Two errors detection, with NMI generation

Correct data if one bit is wrong in the 137-bit data received from memory

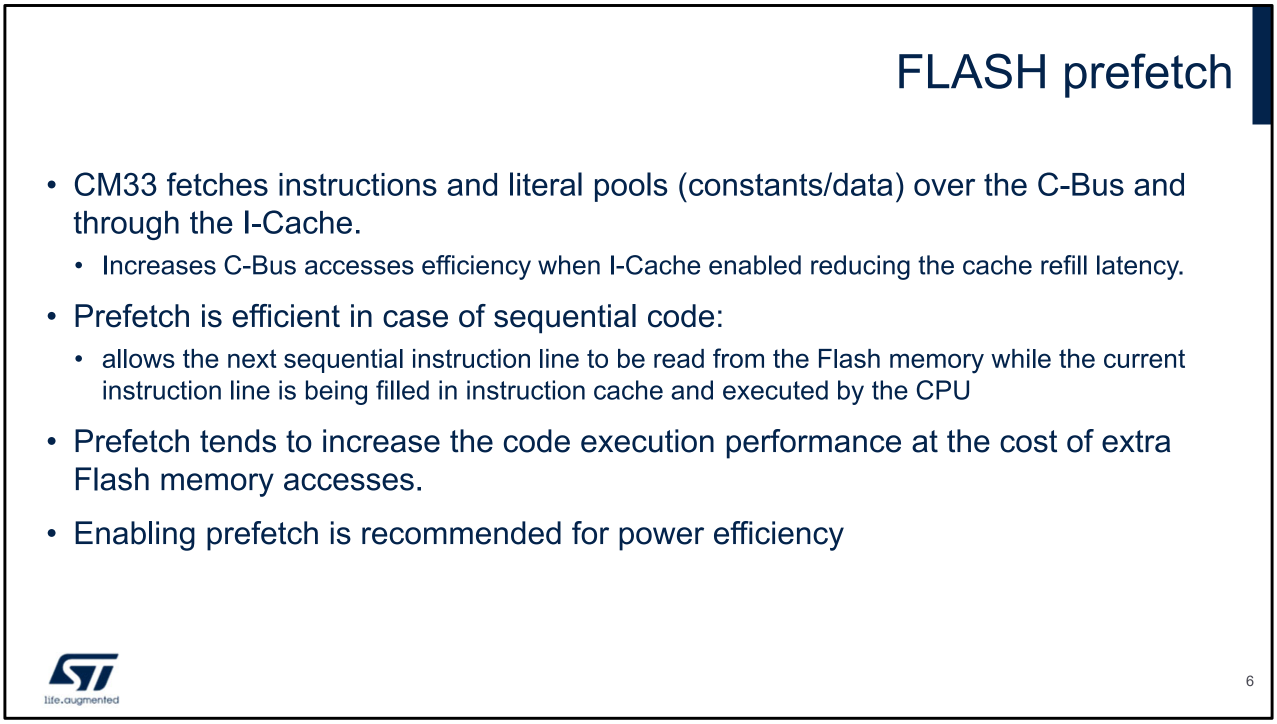
Data in Flash memory are 137-bits wide: nine bits are added per each quad word of 128 bits. The ECC mechanism supports: • One error detection and correction • Two error detection When one error is detected and corrected, the ECCC flag (ECC correction) is set in the Flash ECC register. An interrupt can be generated. When two errors are detected, the ECCD flag (ECC detection) is set in the Flash ECC register. In this case, an NMI is generated. The address and bank number at which the error has been detected are captured in status registers for further investigation.



FLASH read access latency

Wait states (latency)	HCLK max (MHz) with LPM = 0			
	VCORE Range 1	VCORE Range 2	VCORE Range 3	VCORE Range 4
0 WS (1 CPU cycle)	≤ 32	≤ 30	≤ 24	≤ 12
1 WS (2 CPU cycles)	≤ 64	≤ 60	≤ 48	≤ 25
2 WS (3 CPU cycles)	≤ 96	≤ 90	≤ 55	-
3 WS (4 CPU cycles)	≤128	≤ 110	-	-
4 WS (5 CPU cycles)	≤ 160	-	-	-
	HCLK max(MHz) with LPM = 1			
0 WS (1 CPU cycle)	WS ≥ HCLK (MHz) / 10 -1			≤ 8
1 WS (2 CPU cycles)				≤ 16
2 WS (3 CPU cycles)				≤ 25
…				-
15 WS (16 CPU cycles)				-

- To correctly read data from the Flash memory, the number of wait states (latency) must be correctly programmed according to the frequency of the CPU clock (HCLK) and the internal voltage range of the device VCORE. The table below shows the correspondence between wait states and CPU clock frequency.
- **LPM**: The Flash memory supports a low-power read mode when setting the LPM bit in the *FLASH access control register (FLASH_ACR)*.

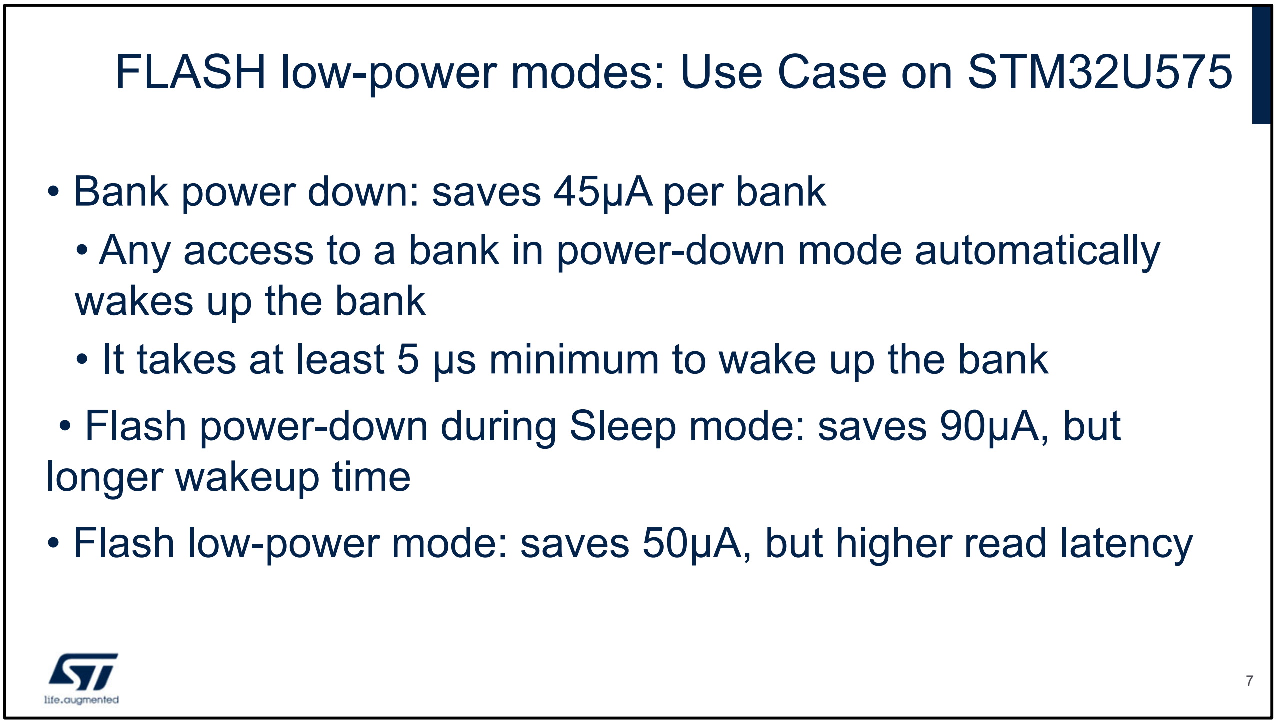


FLASH prefetch

- CM33 fetches instructions and literal pools (constants/data) over the C-Bus and through the I-Cache.
  - Increases C-Bus accesses efficiency when I-Cache enabled reducing the cache refill latency.
- Prefetch is efficient in case of sequential code:
  - allows the next sequential instruction line to be read from the Flash memory while the current instruction line is being filled in instruction cache and executed by the CPU
- Prefetch tends to increase the code execution performance at the cost of extra Flash memory accesses.
- Enabling prefetch is recommended for power efficiency

The Cortex-M33 fetches instructions and literal pool constants over the C-Bus and through the instruction cache if it is enabled.
The prefetch block increases the efficiency of C-Bus accesses when the instruction cache is enabled by reducing the cache refill latency. Prefetch is efficient in the case of sequential code; prefetch in the Flash memory allows the next sequential instruction line to be read from the Flash memory while the current instruction line is being filled in instruction cache and executed by the CPU.
Prefetch is enabled by setting the PRFTEN bit in theFLASH access control register (FLASH_ACR).PRFTEN must be set only if at least one wait state is needed to access the Flash memory.

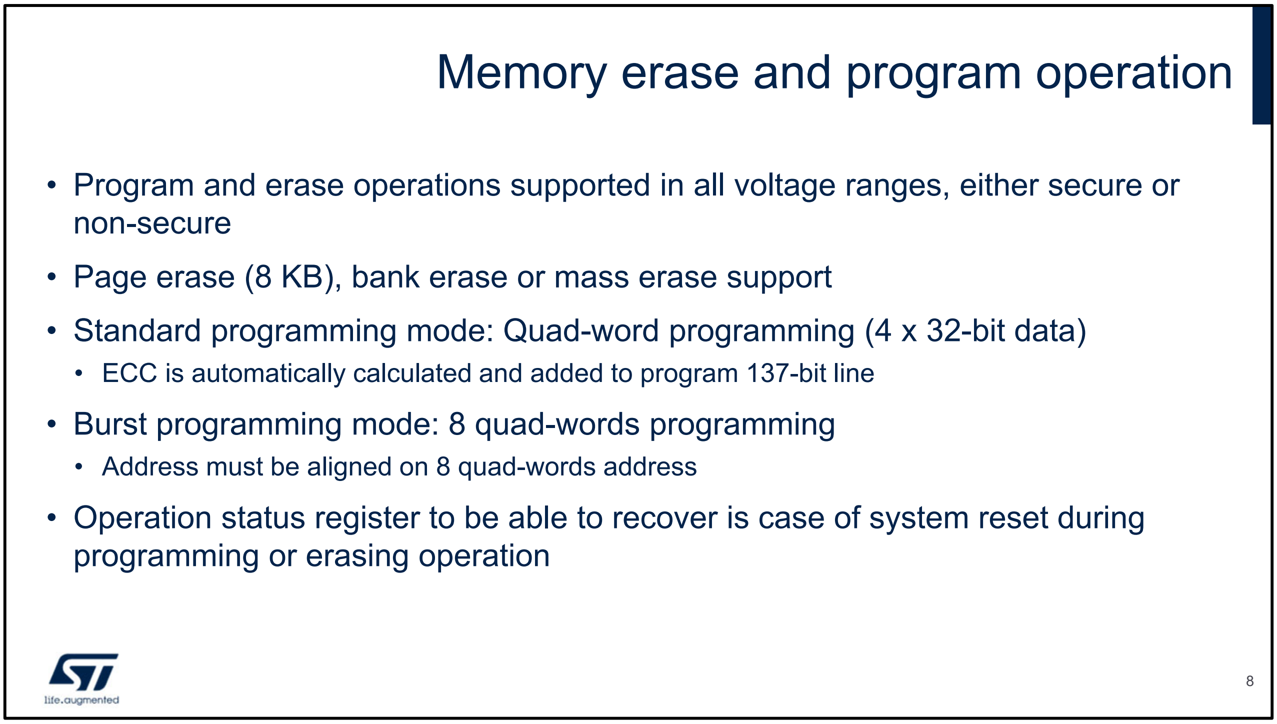
Note that Prefetch tends to increase the code execution performance at the cost of extra Flash memory accesses. It may impact power consumption when activated, but power efficiency is better thanks to the increased performance. Here are some performance metrics expressed in Coremark per megahertz Performance when icache is off and prefetch is off is 2.2. Performance when icache is off and prefetch is on is 2.7. This illustrates the performance increase thanks to prefetch in the case of a Cache miss. As Coremark code is entirely in icache (no cache miss after the first iteration),the prefetch has no impact on the Coremark score when icache is enabled.



FLASH low-power modes: Use Case on STM32U575

- Bank power down: saves 45µA per bank
  - Any access to a bank in power-down mode automatically wakes up the bank
  - It takes at least 5 µs minimum to wake up the bank
- Flash power-down during Sleep mode: saves 90µA, but longer wakeup time
- Flash low-power mode: saves 50µA, but higher read latency

The Flash memory consumption can be reduced when the code is not executed from Flash. After reset, both banks are in normal mode. In order to reduce power consumption, each bank can be independently put in power-down mode by setting the PDREQx bit. Any access to a bank in power-down mode automatically wakes up the bank. It takes at least 5µs to wake up the bank. A powered down bank saves 45 microamperes, power downed flash in sleep mode saves 90 microamperes.
Activating the low-power read mode by setting the LPM bit in the FLASH access control register (FLASH_ACR) saves 50 microamperes, at the expense of an increased latency.



Memory erase and program operation

- Program and erase operations supported in all voltage ranges, either secure or non-secure
- Page erase (8 KB), bank erase or mass erase support
- Standard programming mode: Quad-word programming (4 x 32-bit data)
  - ECC is automatically calculated and added to program 137-bit line
- Burst programming mode: 8 quad-words programming
  - Address must be aligned on 8 quad-words address
- Operation status register to be able to recover is case of system reset during programming or erasing operation

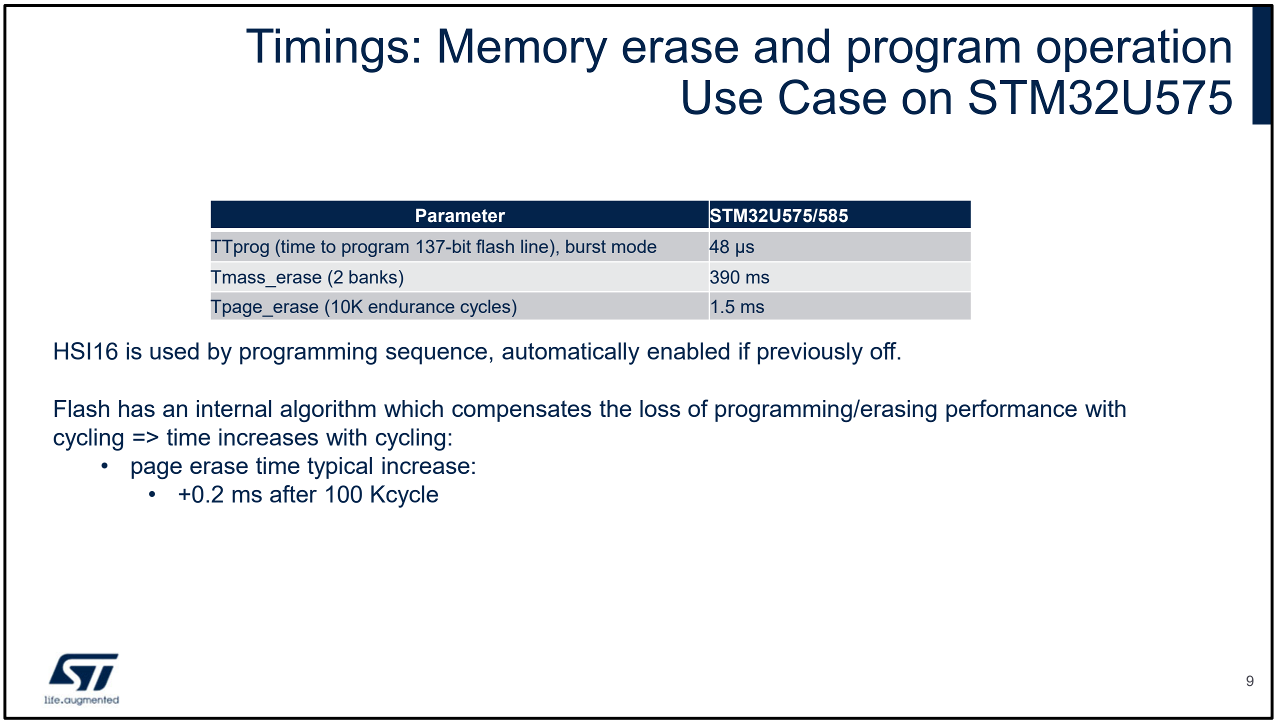
Read, program and erase operations are supported in all voltage ranges. When trustzone is enabled, the non-secure software is only permitted to access the non-secure part of the flash.
Erase can be performed with a page granularity, for one bank or both banks. In the latter case, this is called a mass-erase.
The flash controller implements two programming modes:
-Single quadword, called normal mode
-Eight quadwords representing 128 bytes, called burst mode.
In both cases, the ECC code is calculated and added to the data, so that 137 bits are actually programmed.
Programming 1 megabyte at 160 MHz takes 7.7 seconds

in normal mode, 3.1 seconds in burst mode.

The contents of the Flash memory currently being accessed are not guaranteed if a reset occurs during a Flash memory program or erase operation.

The status of the Flash memory can be recovered from the FLASH operation status register when a system reset occurs during a Flash memory program or erase operation.

It is the software’s responsibility to check the Flash memory status and to take corrective actions.



Timings: Memory erase and program operation Use Case on STM32U575

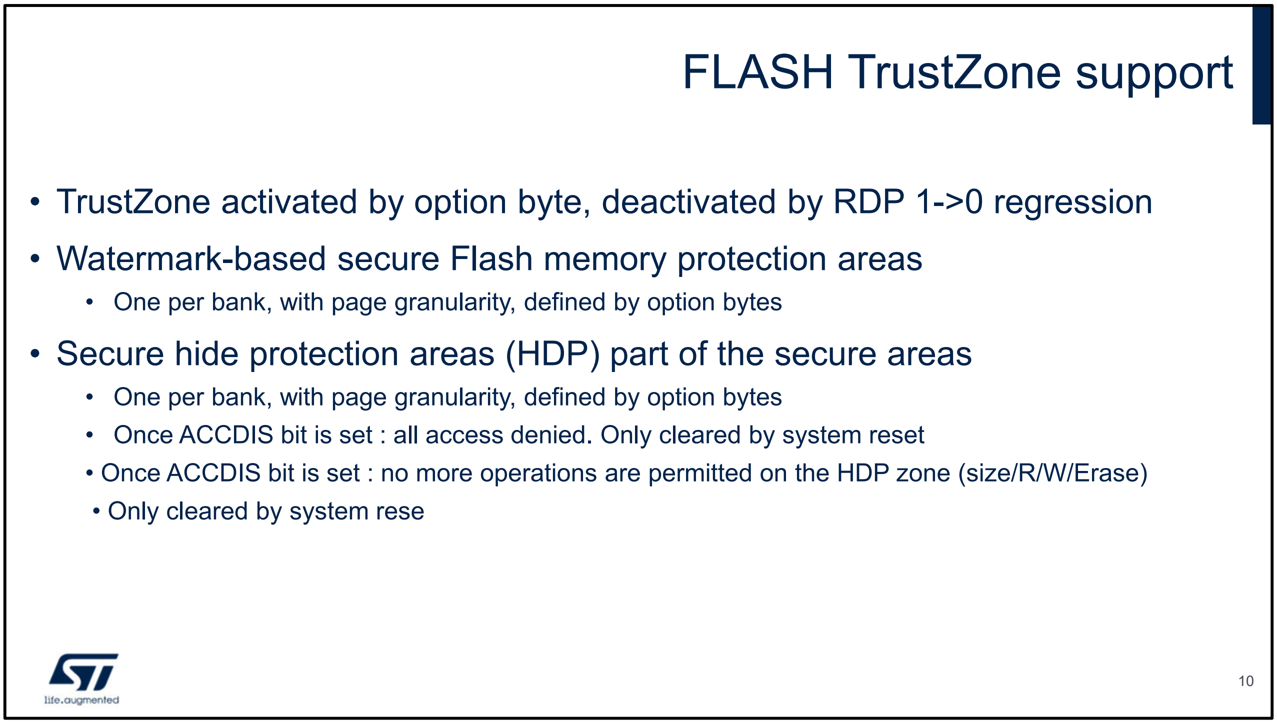
Parameter	STM32U575/585
TTprog (time to program 137-bit flash line), burst mode	48 µs
Tmass_erase (2 banks)	390 ms
Tpage_erase (10K endurance cycles)	1.5 ms

HSI16 is used by programming sequence, automatically enabled if previously off.

Flash has an internal algorithm which compensates the loss of programming/erasing performance with cycling => time increases with cycling:

- page erase time typical increase:
  - +0.2 ms after 100 Kcycle

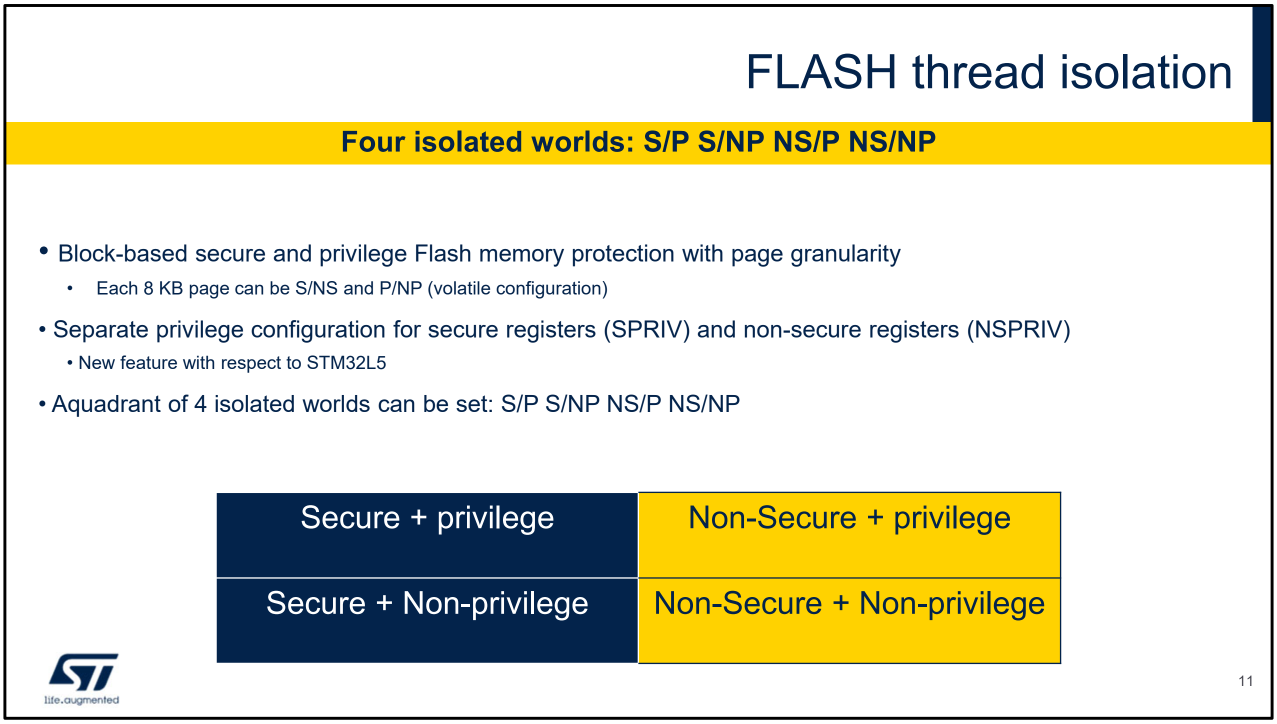
This slide provides some metrics regarding flash program and erase operations. The time to program a quadword + ECC code is 48 microseconds when burst mode is used. The time to fully erase the two banks is 390 milliseconds. The time to erase one page assuming 10 kilo endurance cycles is 1.5 millisecond. An internal algorithm manages the erase sequence, and the erase time increases when the number of endurance cycles increases. An additional 0.2 millisecond is typically required when erasing a page with 100 kilo cycles. The internal 16-Megahertz oscillator HSI16 is automatically enabled when an erase or programming sequence starts, and automatically disabled when this sequence completes, except if the HSI16 was previously enabled.



FLASH TrustZone support

- TrustZone activated by option byte, deactivated by RDP 1->0 regression
- Watermark-based secure Flash memory protection areas
  - One per bank, with page granularity, defined by option bytes
- Secure hide protection areas (HDP) part of the secure areas
  - One per bank, with page granularity, defined by option bytes
  - Once ACCDIS bit is set : all access denied. Only cleared by system reset
  - Once ACCDIS bit is set : no more operations are permitted on the HDP zone (size/R/W/Erase)
  - Only cleared by system rese

When TrustZone security is active, a part of the Flash memory can be protected against non-secure read and write accesses. Deactivation of TrustZone is only possible when the Readout protection, or RDP, is changed from level 1 to level 0. Up to two different non-volatile secure areas can be defined by option bytes and can be read or written only by a secure access : one area per bank with a page granularity. Each of them supports a secure hide protection area, starting at the same start page offset and ending at a programmable end page offset. The contents of the secure hide protection area is marked as non-accessible after the corresponding HDP_ ACCDIS bit is set to one. This is used to prevent subsequent access to a part of the flash and is used to isolate the secure boot code and data from both secure and non-secure application codes.



FLASH thread isolation

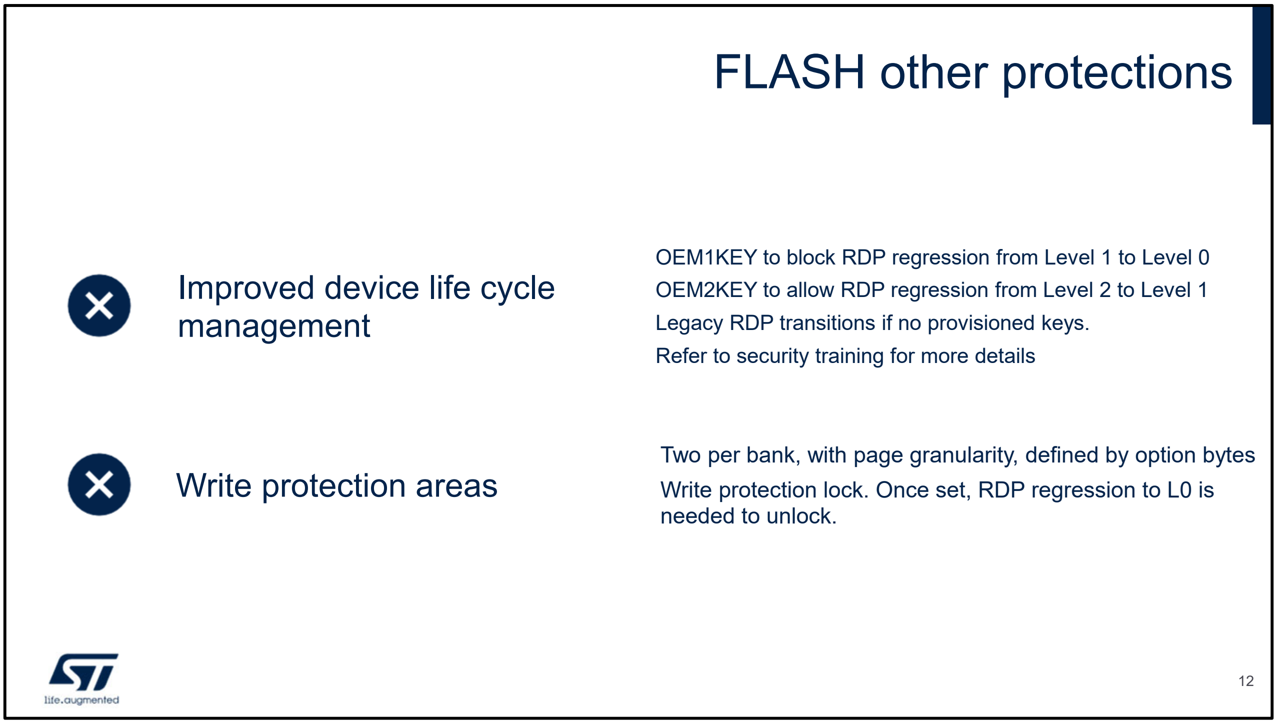
Four isolated worlds: S/P S/NP NS/P NS/NP

- Block-based secure and privilege Flash memory protection with page granularity
  - Each 8 KB page can be S/NS and P/NP (volatile configuration)
- Separate privilege configuration for secure registers (SPRIV) and non-secure registers (NSPRIV)
  - New feature with respect to STM32L5
- Aquadrant of 4 isolated worlds can be set: S/P S/NP NS/P NS/NP

Secure + privilege	Non-Secure + privilege
Secure + Non-privilege	Non-Secure + Non-privilege

life.augmented

Any flash page can be set as secure/non-secure thanks to dedicated secure registers in the flash interface: FLASH_SECBB1Rx (with x=1 to 8) and FLASH_SECBB2Rx (with x=1 to 8). At reset these registers are cleared (non-secure). A page which already belongs to a secure watermark area will be secure whatever its block-based bit configuration. In each security domain, the privilege level of each flash page is programmable: either unprivileged or privileged, by means of FLASH_PRIVBB1Rx (with x=1 to 8)
and FLASH_PRIVBB2Rx (with x=1 to 8) registers. 4 quadrants of isolated worlds are thus obtained:
- Secure privilege
- Secure non-privilege
- Non-secure privilege
- Non-secure non-privilege



FLASH other protections

Improved device life cycle management

OEM1KEY to block RDP regression from Level 1 to Level 0

OEM2KEY to allow RDP regression from Level 2 to Level 1

Legacy RDP transitions if no provisioned keys.

Refer to security training for more details

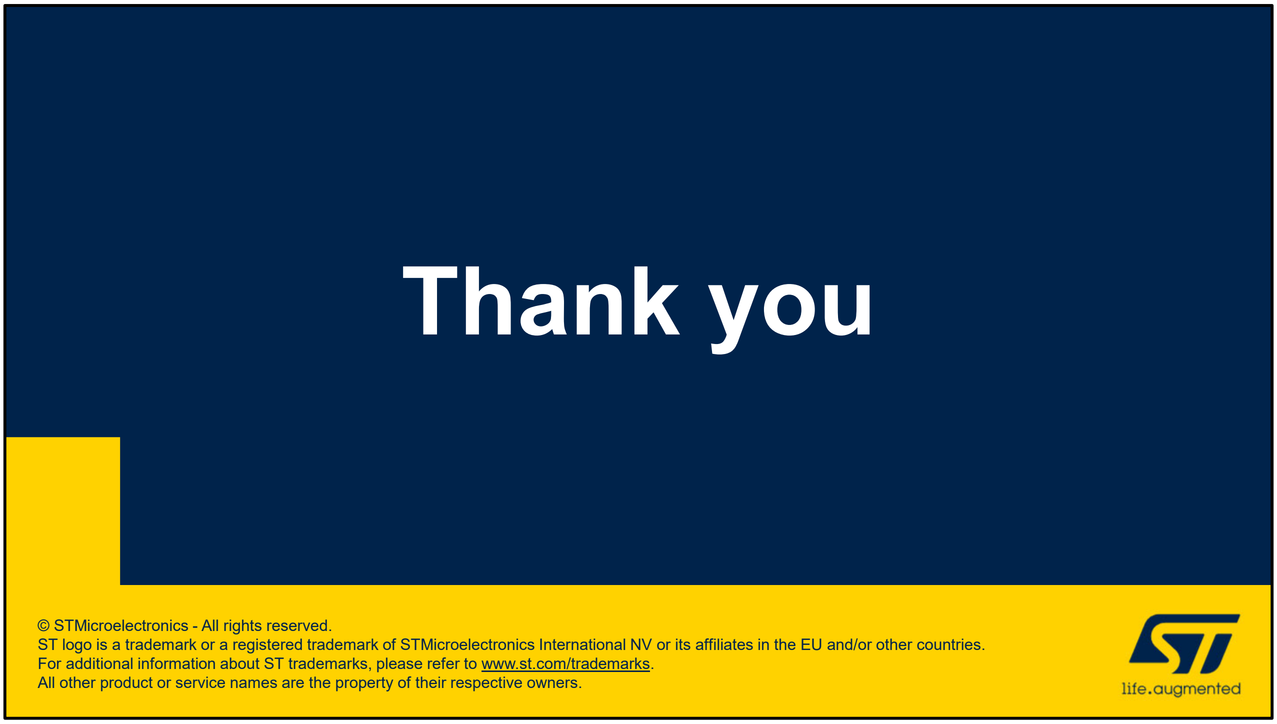
Write protection areas

Two per bank, with page granularity, defined by option bytes

Write protection lock. Once set, RDP regression to L0 is needed to unlock.

Regarding the RDP state machine, STM32U5 implements a new feature: OEM1/OEM2 lock activation. Two 64-bit keys OEM1KEY and OEM2KEY can be defined in order to lock the RDP regression from Level 1, or to allow the regression from Level 2. Each 64-bit key is coded on two registers. OEM1KEY and OEM2KEY cannot be read through these registers. In order to regress from RDP level 1 to RDP level 0, the debugger has to provide the correct OEM1 key value. In order to regress from RDP level 1 to RDP level 0.5, the debugger has to provide the correct OEM2 key value. In order to regress from RDP level 2 to RDP level 1, the debugger has to provide the correct OEM2 key value. When these keys are not provisioned, the STM32U5 only implements the legacy transitions.

When the RDP is set to Level 2 and the OEM2 key is not provisioned, JTAG and SWD are definitively disabled. If the OEM2 key is provisioned, the JTAG and SWD remain enabled under reset only to obtain device identification and provide the OEM2 key to request RDP regression. Refer to the security training for more information about the device life cycle. Four write protection areas are supported: two per bank. Program and erase operations are prohibited in write protection areas. Consequently, a software mass erase cannot be performed if one area is write-protected. Each area is defined by a start page offset and an end page offset related to the physical Flash bank base address. Each write-protection area can be independently locked. In this case it is not possible to modify the area settings, and the unlock can be done only thanks to RDP regression to level 0. The Write protection attribute is orthogonal to the secure and HDP settings.



Thank you

life.augmented