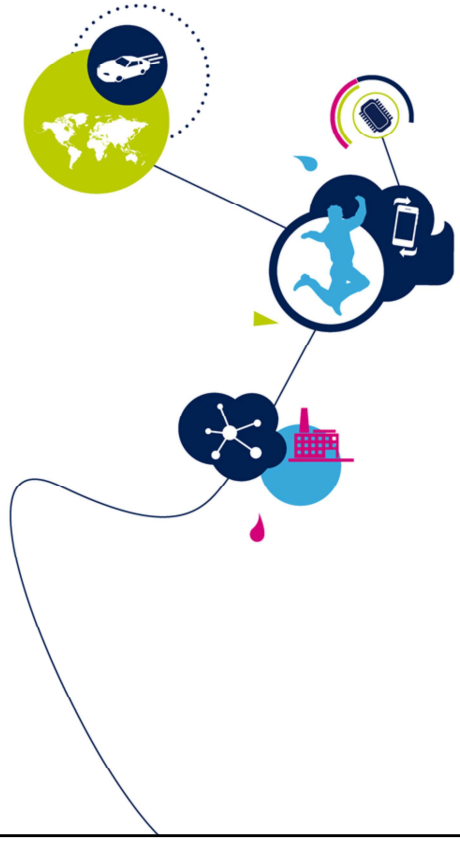
STM32H7- FLASH

Embedded FLASH

Revision 1.0



life.augmented

Hello, and welcome to this presentation of the STM32 Flash memory interface. It covers all the new features of the STM32H7 Flash memory.

Key Features (1/2)

- Up to 2 Mbytes of Flash memory
- 256-bit-wide Flash words
- Error Code Correction (ECC): 10 ECC bits per Flash word
- Program:
  - Flash programming by 256 bits
  - Double-word, word, half-word and byte write operations
- Dual-bank organization supporting simultaneous operations:
  - Two read/program/erase operations can be executed in parallel on the two banks

life.augmented

STM32H7 devices embed high-speed embedded memories with a dual-bank Flash memory up to 2 Mbytes that can be used for storing programs and data.
The Flash memory is organized as 266-bit Flash words that can be used for storing both code and data constants. Each word consists of:
One Flash word (8 words, 32 bytes or 256 bits)
10 ECC bits
The STM32H7 series has a Flash memory with Dual-bank organization supporting simultaneous operations: two read/program/erase operations can be executed in parallel on the two banks.

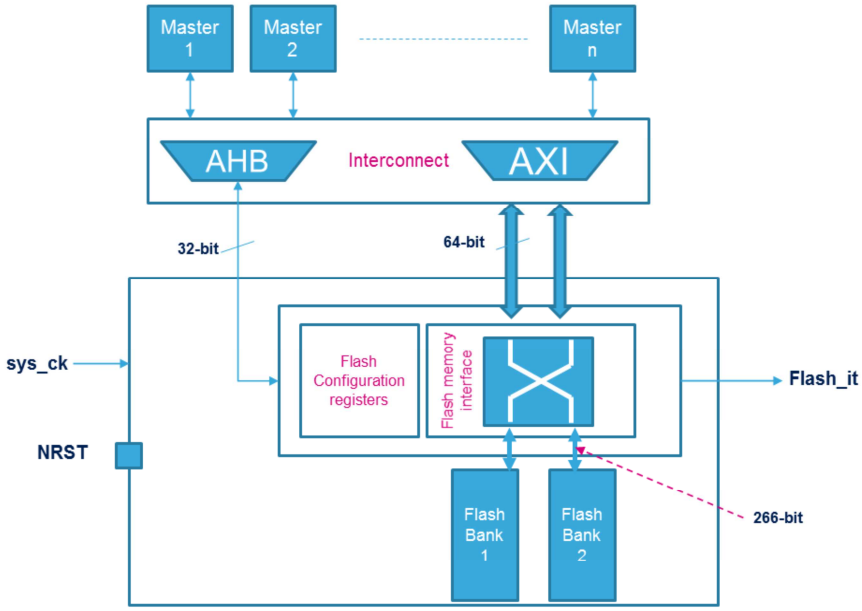
Key Features (2/2)

- Enhanced protection features
  - Readout protection,
  - Sector write protection
  - 2 PCROP protection area (1 per bank) (execute-only memory)
- 2 secure areas in user Flash memory (1 per bank)
- Bank swapping: the address mapping of the user Flash memory of each bank can be swapped
- Option bytes

life.augmented

The Flash memory supports Enhanced Security features which can be configured using the option bytes: the Readout protection (RDP), the Sector write protection and 2 PCROP protection area (1 per bank) (execute-only memory).
The Flash interface allows swapping Bank 1 and Bank 2 memory mapping. This feature can be used after a firmware upgrade to restart the device on the new firmware after a system reset.
The Flash interface allows also setting a “secure” area in each Flash bank. The data and program stored in this area cannot be accessed unless the secure mode is set. The secure area helps isolating secure user code from non-secure application code.
Please refer to the specific training about System protections for more details about these protection options.

Flash memory block diagram



life.augmented

This figure shows the Flash memory high-level block diagram.
The Flash interface implements a dual AXI bus interface for code/data accesses and an AHB interface to configure the Flash memory interface. The AXI interface can request a read/program operation at the same time.

Flash memory architecture

5

- The Flash memory is organized as 266-bit words:
  - 256-bit Data Flash word (8 words or 32 bytes)
  - And 10 ECC bits
- Flash memory is divided into two independent banks. Each bank has:
  - A 1-Mbyte user Flash memory block divided into 128 Kbytes sectors (8x sectors)
  - 128 Kbytes of System Flash memory
  - 2 Kbytes (64 Flash words) of user option bytes for user configuration
    - Available only in Bank 1
    - Accessed only through the Flash register interface (It is not memory mapped)
- The Flash interface can drive different operations in parallel on each bank, but only one read or write operation at a time on a given bank

life.augmented

The Flash memory is organized as 266-bit Flash words that can be used for storing both code and data constants. Each word consists of:
• One Flash word (8 words, 32 bytes or 256 bits)
• 10 ECC bits
The Flash memory is divided into two independent banks. Each bank is organized as follows:
• A 1-Mbyte user Flash memory block containing eight user sectors of 128 Kbytes (4 K Flash words)
• 128 Kbytes of System Flash memory from which the device can boot: This area contains root secure services (RSS) and the bootloader, which are used respectively for secure or non-secure Flash memory programming through one of the following interfaces: USART, USB (DFU), I2C, SPI or Ethernet.
The System Flash memory is reserved for use by STMicroelectronics. It is programmed by STMicroelectronics when the device is manufactured, and then protected

against spurious program/erase operations. For further details, please refer to application note AN2606 “STM32 microcontroller System Flash memory boot mode” available from *http://www.st.com*.

• 2 Kbytes (64 Flash words) of user option bytes for user configuration: This area is available only in Bank 1. Unlike user Flash memory and System Flash memory, it is not mapped to any memory address and can be accessed only through the Flash register interface.

Flash Memory Organization

Block		Name	Block base address	Size
Bank 1	Main Memory	Sector 0	0x0800 0000 - 0x0801 FFFF	128 Kbytes
		Sector1	0x0802 0000 - 0x0803 FFFF	128 Kbytes
			...	
		Sector 7	0x080E 0000 - 0x080F FFFF	128 Kbytes
	System Flash	System memory	0x1FF0 0000- 0x1FF1 FFFF	128 Kbytes
	Option byte Sector		Not memory mapped	2 Kbytes
Bank 2	Main Memory	Sector 0	0x0810 0000 - 0x0811 FFFF	128 Kbytes
		Sector 1	0x0812 0000 - 0x0813 FFFF	128 Kbytes
			...	
		Sector 7	0x081E 0000 - 0x081F FFFF	128 Kbytes
	System Flash	System memory	0x1FF4 0000- 0x1FF5 FFFF	128 Kbytes

life.augmented

This table describes the Flash memory organization that is divided into 2 banks each having a main memory block containing 8 sectors of 128 Kbytes each.
Each main memory block has a System Flash block that contains the system memory which is reserved for use by STMicroelectronics and contains the bootloader. When selected, the device boots in System memory to execute the bootloader.
2 Kbytes (64 Flash words) of user option bytes for user configuration available only in Bank 1.

Flash read operations

- Flash interface support the following access types:
  - Double-word (64 bits)
  - Single-word (32 bits)
  - Half-word (16 bits)
  - Byte (8 bits)
- The Flash interface implements:
  - A dual AXI bus interface for code/data accesses
  - An AHB interface for Flash interface configuration
- To ensure correct Flash interface read operations, the number of wait states (LATENCY) must be correctly configured in the FLASH_ACR register according to the Flash memory interface frequency

life.augmented

The Flash interface can be accessed by Double-word (64 bits), by Single-word (32 bits), by Half-word (16 bits) or by Byte (8 bits).
The Flash interface clock must be enabled and running when reading information from Flash memory. To ensure correct Flash interface read operation, the number of wait states
(LATENCY) must be correctly configured in the FLASH_ACR register according to the Flash memory interface frequency.
The Flash interface implements a dual AXI bus interface for code/data accesses and an AHB interface for Flash interface configuration. The AXI interfaces can request a read/program operation at the same time.

Flash read operations

- After power-on, the clock used is the HSI (64 MHz) and 7WS are configured by default in the FLASH_ACR register
- Relation between the AXI clock frequency and Flash memory read time

V_{CORE} range		Wait States(WS) (LATENCY)	Maximum AXI Frequency
VOS0/VOS 1	1.15V – 1.35V	0WS	70 MHz
		1WS	140 MHz
		2WS	210 MHz
		3WS	225 MHz
		4WS	240 MHz
VOS2	1.05V – 1.15V	0WS	55 MHz
		1WS	110 MHz
		2WS	165 MHz
		3WS	225 MHz
VOS3	0.95V – 1.05V	0WS	45 MHz
		1WS	90 MHz
		2WS	135 MHz
		3WS	180 MHz
		4WS	225 MHz

life.augmented

This table shows the correspondence between the number of wait states, the bus clock frequency and VCORE range. After power-on, the clock used is the HSI (64 MHz) and 7 wait-states are configured by default in the FLASH_ACR register.

Flash read operations

- The Flash interface has a read command buffer
  - Buffer depth is fixed to 3 requests.
  - When it is full (3 read requests queued in the buffer), any new read request will stall the bus interface and consequently the master.
- The Flash interface has a read data buffer
  - Any system read request for data that belongs to the same Flash data word (256 bits) does not trigger additional Flash read operations, data is directly read from the current data read buffer.
  - Any system read request for data that is not available in the read buffer triggers a Flash read operation.
  - Read commands to each bank are associated with a read data buffer of 256 bits.

life.augmented

The read mechanism is the following:
• The read command buffer depth is fixed to 3 requests. When it is full (3 read requests queued in the buffer), any new read request will stall the bus interface and consequently the master.
• Any system read request for data that is not available in the read buffer triggers a Flash read operation. This data is buffered inside the read data buffer.
If several consecutive read accesses request data that belongs to the same Flash data word (256 bits), data is directly read from the current data read buffer and does not trigger additional Flash read operations.

Flash read operations

- The read operations are executed in the Flash interface reception order
  - Read request buffer is free as soon as the last data of current read transaction is transferred from the Flash memory to the read data buffer inside the Flash memory interface.

life.augmented

The read command queue buffer is free as soon as the last data of the current read transaction is transferred from the Flash memory to the read data buffer inside the Flash memory interface.

Error code correction (1/2)

- The ECC mechanism is based on the SECDED algorithm. It supports:
  - Single error correction
  - Double error detection
- When an ECC error is detected:
  - Address of the failing Flash word is saved in the FLASH_ECC_FA1/2R register.
  - In case of successive error detections, only the address corresponding to the first error will be saved.
- When an error is detected and corrected:
  - The SNECCERR1/2 flag is set in the FLASH_SR1/2 register.
  - An interrupt is generated if the SNECCERRIE bit is set in the FLASH_CR1/2 register.

life.augmented

Data in Flash memory are 266-bit words: 10 ECC bits are added per Flash word of 256 bits.
The ECC mechanism is based on the SECDED algorithm. It supports:
• Single error correction
• Double error detection
When an error is detected and corrected, the SNECCERR1/2 flag is set in the FLASH_SR1/2 register. An interrupt is generated if the SNECCERRIE bit is set in the FLASH_CR1/2 register.
When an ECC error is detected, the address of the failing Flash word is saved in the FLASH_ECC_FA1/2R register. In case of successive error detections, only the address corresponding to the first error will be stored. This register is automatically cleared when the associated flag that generated the error has been reset.

Error code correction (2/2)

- When two errors are detected (in this case the received data is not corrected):
  - The DBECCERR1/2 flag is set in the FLASH_SR1/2 register
  - A bus error is generated.
  - An interrupt is generated if the DBECCERRIE1/2 bit is set in the FLASH_CR1/2 register

life.augmented

When two errors are detected, the DBECCERR1/2 flag is set in the FLASH_SR1/2 register and a bus error is generated. In this case, the received data is not corrected. An interrupt is generated if the DBECCERRIE1/2 bit is set in the FLASH_CR1/2 register.

Cyclic redundancy check (1/3)

- The Flash interface embeds one cyclic redundancy check hardware (CRC) module
  - Only one CRC check operation on Bank 1 or 2 can be launched at a time
- CRC checked area can be defined either by sectors or by start/end addresses.

life.augmented

The Flash interface embeds a cyclic redundancy check (CRC) hardware module. This module allows checking the integrity of a Flash area content. This area can be defined either by sectors or by start/end addresses.
Only one CRC check operation on Bank 1 or 2 can be launched at a time.
The CRC operation cannot be concurrent with any option byte change operation. This means that if a CRC operation is requested while an option byte change is ongoing, the option byte change operation must be completed before serving the CRC operation, and vice-versa.

Cyclic redundancy check (2/3)

- CRC hardware module processes Flash data by chunks of 128, 512, 2048 or 8192 bytes
  - The Flash interface issues 4, 16, 64 or 256 consecutive Flash-word read accesses
  - CRC burst length is configurable by software.
- These transactions are queued into the read command queue together with other AXI read requests:
  - Thus avoiding to deny AXI read commands.
  - The queue command buffer can contain only one CRC command

life.augmented

The Flash interface issues 4, 16, 64 or 256 consecutive Flash-word read accesses. These transactions are queued into the read command buffer together with other AXI read requests, thus avoiding to deny AXI read commands. The queue command buffer can contain only one CRC command.

Cyclic redundancy check (3/3)

- The configuration steps for a CRC check:
  1. Enable the CRC feature in the FLASH_CR1/2 register
  2. Program the desired data burst size in the FLASH_CRCCR1/1 register
  3. Define the Flash area on which the CRC has to be computed (sector or address)
  4. Start the CRC operation setting the START_CRC bit
  5. Wait until the CRC_BUSY flag is reset.
  6. Retrieve the CRC result in the FLASH_CRCDATAR register.
- The CRC can be computed for an entire bank by setting the ALL_BANK bit in the FLASH_CRCCR1/2 register.
- Running a CRC on PCROP- or secure-protected user Flash area may alter the expected CRC value.

life.augmented

The recommended sequence to configure a CRC operation in Bank 1/2 is the following:
1. Enable the CRC feature by setting the CRC_EN bit in the FLASH_CR1/2 register.
2. Program the desired data size in the CRC_BURST field of the FLASH_CRCCR1/2 register.
3. Define the Flash area on which the CRC has to be computed. Two solutions are possible:
• Define the area start and end addresses by programming registers FLASH_CRCSADD1/2R and FLASH_CRCEADD1/2R, respectively
• Or select the targeted sectors by setting the CRC_BY_SECT bit in the FLASH_CRCCR1/2 register and by programming consecutively the target sector numbers in the CRC_SECT field of the FLASH_CRCCR1/2 register. Set the ADD_SECT bit after each CRC_SECT programming.
4. Start the CRC operation by setting the START_CRC bit.
5. Wait until the CRC_BUSY flag is reset.

6. Retrieve the CRC result in the FLASH_CRCDATAR register.
The CRC can be computed for a whole bank by setting the ALL_BANK bit in the FLASH_CRCCR1/2 register.

Flash program and erase operations (1/2)

- The Flash memory interface supports multiple program operations:
  - Write to user sectors
  - Erase user sectors
  - Erase Bank 1, Bank 2 or both banks
  - Change user option bytes
- The Flash interface write queue buffer can contain 2 requests:
  - Write accesses are accepted until the write queue buffer becomes full.
  - When it is full, the Flash interface stalls the AXI bus.
  - The write operations are executed in the order in which they have been received by the Flash interface.

life.augmented

The Flash memory interface supports multiple program operations:

- Write to user sectors
- Erase user sectors
- Erase Bank 1, Bank 2 or both banks
- Change user option bytes

The write accesses issued through the AXI interface can be considered as bufferable and not cacheable except that it is not possible to read back the write buffer inside the Flash interface.

The embedded Flash memory can be programmed using in-circuit programming or in-application programming.

Flash program and erase operations (2/2)

- A program or erase operation can be executed on Bank 1 while another program or erase operation is executed on Bank 2
  - An exception for option byte change when a level regression is required: in this case, the availability of both banks
- Parallelism must be set up for any program or erase operation
- The parallelism is the maximum number of bits that can be written to '0' in one shot during a write operation. The programming parallelism is also used during sector and bank erase operations.

life.augmented

A program or erase operation can be executed on Bank 1 while another program or erase operation is executed on Bank 2.
Note that the programming parallelism is a parameter that must be configured prior to performing a program or erase operation.

Configuring the programming parameters

- Unlock and program the Flash configuration registers (FLASH_CR1/2)
- Set the programming parallelism for each bank through the PSIZE1/2 bits in the FLASH_CR1/2 registers
  - There is no hardware limitation on programming parallelism (Byte, half-word, word, or Double-word)
  - The lower the parallelism, the lower the peak consumption during a programming operation
  - The lower the parallelism, the longer the execution time
- Set the programming delay through the WRHIGHFREQ parameter in the FLASH_ACR register
  - Programming operation timing constraints depend of the Flash interface frequency

life.augmented

The user application must configure the programming parameters prior to performing a program/erase operation.

Flash programming sequence

Simple write sequence	Optimal block write sequence
Set the PG1/2 bit in the FLASH_CR1/2 register of the targeted bank (Bank 1/2)	
Check the protection of the target memory area	
Write one Flash word corresponding to 32-byte data starting at 32-byte aligned address	Write successively 32 data bytes (Flash words) until the entire block is transferred. Each Flash word must start at an 32-byte aligned address
Check that the QW1/2 flag has been raised and wait until it is reset	

life.augmented

Two write operation mode are possible.
The simple write sequence (recommended) for which the steps are:
1. Set the PG1/2 bit in the FLASH_CR1/2 register of the targeted bank (Bank1/2).
2. Check the protection of the target memory area.
3. Write one Flash word corresponding to 32-byte data starting at 32-byte aligned address.
4. Check that the QW1/2 flag has been raised and wait until it is reset.

This sequence can be used to program a block in Flash memory:
1. Set the PG1/2 bit in the FLASH_CR1/2 register of the corresponding bank (Bank1/2).
2. Check the protection of the target memory area.
3. Write successively 32 data bytes (Flash words) until the whole block is transferred.

Each Flash word must start at a 32-byte aligned address.

Write status busy flags

- Three different status flags located in the FLASH_SR1/2 registers are available for each bank. They indicate the ongoing write operation status:

Flags	Description
BSYx	This flag indicates that an effective write, erase or option byte change operation is ongoing to the Flash memory.
QWx	This flag indicates that a program, erase or option byte change operation is pending. This bit remains high until the write operation is complete. It supersedes the BSYx status bit.
WBNEx	This flag indicates that the write buffer is not empty. It is reset as soon as the write command is queued.

life.augmented

Three different status flags located in FLASH_SR1/2 registers are available for each bank. They indicate the ongoing write operation status:

• BSY1/2: This flag indicates that an effective write, erase or option byte change operation is ongoing to the Flash memory.

• QW1/2: This flag indicates that a program, erase or option byte change operation is pending. This bit remains high until the write operation is complete. It supersedes the BSY1/2 status bit.

• WBNE1/2: This flag indicates that the write buffer is not empty. It is reset as soon as the write command is queued.

Write status busy flags

- If one of the busy flags is active, the MCU cannot switch the D1 domain to Stop or Standby mode
- To release the Flash interface, the BSYx and QWx busy bits must be cleared by:
  - Completing the write buffer with missing data.
  - Forcing the write without filling the missing data by setting the **FW**x bit in the FLASH_CRx register.
  - Terminating the write by resetting the **PG**x bit in the FLASH_CRx register: this leads to the loss of data in the write buffer.

life.augmented

If one of the busy flags is active, the MCU cannot switch the D1 domain to Stop or Standby mode.
To release the Flash interface, the BSYx and QWx busy bits must be cleared.

User option bytes

- The user option byte change operation can be used to modify the configuration and the protection settings saved in the Flash memory option byte area.
- To increase the robustness of option byte storage in the Flash interface, each option byte data is associated to an error code correction (ECC) inside the Flash memory.
- The Flash interface features two sets of option byte registers
  - FLASH_XXX_CUR: All "_CUR" registers are read-only. Their values are automatically loaded after power-on or after an option byte change operation.
  - FLASH_XXX_PRG: All "_PRG" registers are in read/write mode. Setting this register allows modifying the option bytes.

life.augmented

The user option byte change operation can be used to modify the configuration and the protection settings saved as the Flash memory option byte area.
The Flash interface features two sets of option byte registers:
• The first register set contains the current values of the option bytes. Their names has the _CUR extension. All "_CUR" registers are read-only. Their values are automatically loaded after power-on or after an option byte change operation.
• The second register set allows modifying the option bytes. Their names contains the PRG extension. All "_PRG" registers can be accessed in read/write mode.

Interrupts

- Each flag error can generate an interrupt if the corresponding interrupt enable bit has been set in the FLASH_CR1/2 register.

Interrupt event	Description
End of operation	Set by hardware when one or more Flash memory operation (programming / erase) has been completed successfully.
Programming error	Set by hardware when a Flash memory operation (program / erase) completes unsuccessfully.
Write protection error	Set by hardware when attempt to program or erase in a write protected area, System memory area or OTP area.
Operation error	Set by hardware when the Flash memory detects an error during a write or erase operation. This error may be caused by an incorrect Flash memory behavior.

life.augmented

Four interrupts can be generated by the Flash memory.
The end-of-operation interrupt is triggered when one or more Flash program or erase operations is completed successfully.
The programming error interrupt is triggered when a Flash memory program or erase operation failed.
The write protection error interrupt is triggered when a write access is attempted to a write-protected area of the Flash memory.
The operation error interrupt is triggered when an error is detected during a write or erase operation.