Hello, and welcome to this presentation of the STM32L4 Flash memory. All STM32L4 Flash features will be presented.
Please note that this presentation has been written for STM32L47x/48x devices. Key differences with other devices are indicated at the end of the presentation unless otherwise specified.
The STM32L4 embeds up to 1 Mbyte of dual-bank Flash memory.
The Flash memory interface manages all memory access (read, programming, erasing) as well as memory protection and option bytes.
Applications using this Flash interface benefit from its high performance together with low-power access. It supports read-while-write, has a small erase granularity, a short programming time and allows dual-bank booting.
The STM32L4’s Flash memory has several key features. It has up to 1 Mbyte of dual-bank Flash memory, with a read-while-write capability that can program or erase one bank while executing code from the other bank. The erase granularity, corresponding to the page size, is only 2 Kbytes. A page, bank or mass erase operation requires only 22 ms, and the programming time is only 82 µs for a double-word. The adaptive real-time memory accelerator, with an instruction cache, a data cache and a prefetch buffer, allows a linear performance in relation to frequency. The Flash memory supports Error Code Correction (ECC) which is 8 bits long for each 64-bit double word. A single error is detected and corrected. A double error is detected, but not corrected.
The Flash memory is divided into 2 banks, each having a main memory block containing 256 pages of 2 Kbytes each. Each page is made of 8 rows of 256 bytes.

Each main memory block has an information block which contains 3 parts. The first part is 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.

The second part is a 1-Kbyte one-time programmable area. This area is located in Bank 1 only. The OTP area cannot be erased and can be written to only once. If one bit is at ‘0’, the entire double-word can no longer be written, even with the value ‘0’.

The last part contains the option bytes for configuring user options.
This slide shows the Flash memory map. There are 256 pages in Bank 1, starting from page 0, and 256 pages in the Bank 2, starting from page 256. The page number MSB corresponds to the bank number. The page number is used in the software procedure to erase a page.
The Flash is dual-bank memory with read-while-write and dual-bank boot capability, able to boot from either Bank 1 or Bank 2.

The BFB2 option in the user option bytes is used to select the dual-bank boot mode. When the BFB2 option is set, the device boots either in Bank 2 or Bank 1 depending on the valid bank. When the BFB2 option is cleared, the device always boots from Bank 1.

The DUALBANK option is used to select either a single bank or a dual bank for the 256-Kbyte and 512-Kbyte device part numbers. For instance, when DUALBANK is selected for 512-Kbyte devices, 128 pages are in Bank 1 and 128 pages are in Bank 2.

The first page name in Bank 2 is always page 256 regardless of the device’s memory size, as the page name MSB refers to the bank number.
With a dual-bank memory, it is possible to read from one bank while programming or erasing the other bank. Code execution is not stopped when the Flash memory is being programmed. When programming or erasing data in the same bank, the AHB is stalled as long as the Flash memory controller is busy. Using the FB_MODE bit in the System Configuration memory remap register, the two Flash bank addresses can be swapped. When this bit is cleared, Bank 1 is mapped at address 0x0800 0000 and aliased at address 0. When this bit is set, Bank 2 is mapped at address 0x0800 0000 and aliased at address 0 which allows the device to boot into Bank 2.
The dual-bank boot allows a safe firmware upgrade as the previous firmware version is still present in the other memory bank. The dual-bank boot is managed by the bootloader. The device boots in Bank 2 using the BFB2 option bit, programmed in the Flash option bytes. The boot pin and boot option are configured for booting in Flash memory. If the BFB2 option bit is cleared, the device boots in Flash Bank 1. If the BFB2 option bit is set, the device boots in the System Flash memory. The bootloader checks the bank’s first address, as it must read there the stack pointer at that location. If Bank 2’s first address is a valid SRAM address, the bootloader swaps the banks to remap the Bank 2 at address 0 and jumps into Bank 2. If it is not valid, the bootloader swaps the banks to remap Bank 1 at address 0 and jumps into Bank 1. Note that the bootloader uses resources in SRAM1 from address 0x2000 0000 to address 0x2000 1000, so this SRAM area must not be used by the application when the BFB2
option bit is set.
The Flash memory embeds an Error Code Correction function to ensure robust memory integrity and safety. The ECC is 8 bits long for a 64-bit word. In case of a single error, it is corrected. The ECCC bit is set in the Flash ECC register, and an interrupt is generated if it is enabled. In case of a double error, it is detected but not corrected. The ECCD bit is set in the Flash ECC register, and a non-maskable interrupt is generated. When an ECC error is detected, the failure address and associated bank are saved in the Flash ECC register.

The programming granularity is 64 bits, in fact it’s 72 bits with the 8 bit ECC. There are 2 programming modes: Standard mode for the main memory and OTP, and Fast mode for the main memory only. In Standard mode, the Flash memory checks that the double-word is erased before launching the programming. In Fast mode, 32 double-words are programmed without verifying the Flash location.
The Flash programming time is only 82 µs for 64-bit double-words. To program one page (2 Kbytes), 20.9 ms are needed in Standard mode and 15.3 ms in Fast mode. For the complete bank, it requires 3.9 s in Fast mode.

The page erase time is 22 ms. It also requires only 22 ms to erase one or both banks, as both banks can be erased simultaneously.

The short programming and erase time, plus the small page size, make it convenient for data EEPROM emulation.
A fast programming mode allows to program 32 double-words faster than in standard programming mode.
Only the main memory can be programmed in Fast programming mode.
The Flash address location contents are not verified by hardware before programming in Fast mode.
The 32 double-words must be written successively. The high voltage is kept on the Flash memory for all programming. The maximum time between two double-word write requests is the programming time, which is approximately 20 µs. Consequently, interrupts should be disabled to ensure that the 20 µs between the two word write requests is not overpassed. The minimum clock frequency must be at least 8 MHz in Fast programming mode.
This slide compares standard and fast programming modes. Standard mode can be used to program the main memory and OTP areas while Fast mode cannot be used for OTP programming. Standard mode allows programming 64-bit double-words, or 8 bytes, whereas Fast mode only allows programming 32-bit double-words, or only 256 bytes. In Fast mode, the address location content is not checked before programming, the CPU clock frequency must be greater than 8 MHz and interrupts are prohibited. It takes 2.6 ms to program 256 bytes in Standard mode and 1.9 ms in Fast mode.
The Flash memory is guaranteed for a minimum of 10,000 cycles up to 105 °C. Data retention is 30 years after 10,000 cycles at 55 °C, 15 years after 10,000 cycles at 85 °C and 10 years after 10,000 cycles at 105 °C. It is 30 years after 1,000 cycles at 85 °C, 15 years after 1,000 cycles at 105 °C and 7 years after 1,000 cycles at 125 °C.
In order to read the Flash memory, it is required to configure the number of wait states to be inserted in a read access, depending on the clock frequency. The number of wait states also depends on the voltage scaling range. In Range 1, the Flash memory can be accessed up to 80 MHz with 4 wait states. It can be accessed with 0 wait states up to 16 MHz. For Range 2, it is up to 26 MHz, with 3 wait states. Thanks to the adaptive real-time memory accelerator, the ART accelerator, the program can be executed with 0 wait states independent of the clock frequency. This provides an almost linear performance in relation to frequency and allows to reach 100 Dhrystone MIPS at 80 MHz.

### Flash read access

**100 DMIPS at 80 MHz**

- Adaptive real-time memory accelerator (ART Accelerator™) allowing linear performance versus frequency, regardless of Flash memory access time.

<table>
<thead>
<tr>
<th>Wait states (WS) (Latency)</th>
<th>HCLK (MHz)</th>
<th>V\textsubscript{CORE} Range 1</th>
<th>V\textsubscript{CORE} Range 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 WS</td>
<td>≤ 16</td>
<td>≤ 6</td>
<td></td>
</tr>
<tr>
<td>1 WS</td>
<td>≤ 32</td>
<td>≤ 12</td>
<td></td>
</tr>
<tr>
<td>2 WS</td>
<td>≤ 48</td>
<td>≤ 18</td>
<td></td>
</tr>
<tr>
<td>3 WS</td>
<td>≤ 64</td>
<td>≤ 26</td>
<td></td>
</tr>
<tr>
<td>4 WS</td>
<td>≤ 80</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
The ART accelerator brings outstanding performance and reduces dynamic power consumption. It consists of a 1-KByte instruction cache, 256 bytes of data cache and a prefetch buffer. The instruction cache contains 32 lines of 4 double-words and the data cache has 8 lines of 4 double-words. Once all the instruction cache memory lines have been filled, the LRU (least recently used) policy is used to determine the line to replace in the instruction memory cache. This feature is particularly useful when code contains loops. This architecture is chosen to provide the best tradeoff between cache size, power consumption and performance.

After each miss, the cache is updated with only the requested double-word in order to limit the Flash access for power-saving. In a line, the 4 double-words may be not all be valid.
In case of a miss, the Cortex M4 code takes the instruction directly from the Flash memory. In parallel, the 64-bit line is copied into the current buffer enabled and I-Cache if enabled. So the next sequential access is taken directly from the current buffer.
If prefetch is enabled, another 64-bit Flash access is performed to fill the Prefetch buffer with sequential data.

When the data is present in the current buffer, the CPU reads the current buffer. The next sequential read is performed in the Prefetch buffer, which is copied into the current buffer, so that it is free to be filled with the next sequential data.

If the data is not present in the current buffer, it is read from the Prefetch buffer if it is present. If not, it is read from the instruction cache if there is a cache hit. Otherwise, a Flash access is performed.
The Instruction Cache behaves differently depending on if the prefetch buffer is enabled or not.

If the prefetch buffer is enabled, the ART instruction cache behaves like a branch cache. The cache is modified each time a branch or a jump occurs in the execution flow. Sequential accesses are issued by the current instruction buffer and the prefetch buffer; each time the prefetch buffer is hit, its contents are transferred to current instruction buffer and a new Flash access to fill the prefetch buffer is performed ... In this case, the cache content is not altered.

If the prefetch buffer is disabled, the ART instruction cache behaves like a normal cache. Since no prefetch buffer is available, even a sequential access will modify the cache content.

The power and performance tradeoff must be evaluated for each application to know whether it is better to enable or disable the prefetch buffer. For most of applications, enabling
the prefetch buffer allows to increase slightly the performance but with a higher consumption. Most of the time, the best energy efficiency is provided with caches enabled and prefetch buffer disabled, as it often reduces the number of Flash accesses.
This slide shows the number of cycles needed to execute sequential 16-bit instructions without prefetch, when 3 wait states are needed to access the Flash memory. Every Flash access provides 64 bits or 4 instructions; 3 wait states are therefore inserted every 4 instructions, at every Flash access.
This slide shows the number of cycles needed to execute sequential 16-bit instructions with prefetch enabled, when 3 wait states are needed to access the Flash memory. After each Flash access, another Flash access is performed to fill the prefetch buffer. So after all instructions are fetched from the current buffer, the next sequential instruction is read from the prefetch buffer and no wait state is inserted as long as the instruction flow is sequential.
Several Flash memory protection options can be configured using the option bytes.
The readout protection is configured using the RDP option byte. The readout protection prohibits any access to the Flash memory, the SRAM2 and the backup registers by the debug interface or when booting from SRAM1 or when the bootloader is selected.
The proprietary code protection is configured using the PCROP option byte. This option protects a specific code area from any read or write access; the code can only be executed. The protected area can be defined with 64-bit granularity and one area can be defined in each bank.
The write protection is configured using the WRP option byte. This option protects specific code areas from unwanted write access. The write-protected area can be defined with 2-Kbyte granularity.
Please refer to the specific training about System protections
for more details about these protection options.
Several option bytes are available in the Flash memory to configure certain specific features of the device.

The user option bytes are loaded in two cases: either after a power or brown-out reset, when exiting from Standby or Shutdown modes, or when the OBL_LAUNCH bit is set in the Flash control register (FLASH CR).

Three option bits are used to configure the brown-out reset threshold.

Three options are available to prohibit or allow the Stop, Standby and Shutdown low-power modes.

Four options configure if the watchdogs are enabled by hardware or after a software configuration, and if the independent watchdog is frozen or not in Stop and Standby modes.

2 options are used to enable dual-bank boot and to configure the 512-Kbyte or 256-Kbyte devices in dual-bank configuration.
The nBOOT1 option is used together with the BOOT0 pin to configure the memory used for booting.
2 options are used to configure if the SRAM2 is erased with the system reset, and to enable the SRAM2 parity check.
Several option bytes are used for memory protection options: the RDP for readout protection, PCROP for the start and end addresses of each bank and WRP for the start and end addresses for each of the 2 areas of each bank. The PCROP_RDP bit is used to preserve or erase the PCROP area when the readout protection is removed from Level 1 to Level 0.
Four interrupts can be generated by the Flash memory. The end-of-operation interrupt, which is triggered when one or more Flash program or erase operations is completed successfully. The operation error interrupt is triggered when a Flash memory program or erase operation failed. The read error interrupt is triggered when an address read through the Core Data Bus belongs to an area of the Flash protected by the PCROP option. The ECC interrupt is triggered when one ECC error is detected and corrected. When two ECC errors are detected, a non-maskable interrupt is generated.

<table>
<thead>
<tr>
<th>Interrupt event</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>End of operation</strong></td>
<td>Set by hardware when one or more Flash memory operations (programming / erase) is completed successfully.</td>
</tr>
<tr>
<td><strong>Operation error</strong></td>
<td>Set by hardware when a Flash memory operation (program / erase) is unsuccessful.</td>
</tr>
<tr>
<td><strong>Read error</strong></td>
<td>Set by hardware when an address to be read through the D-bus belongs to a Read-protected area of the Flash (PCROP protection).</td>
</tr>
<tr>
<td><strong>ECC correction</strong></td>
<td>Set by hardware when one ECC error has been detected and corrected.</td>
</tr>
<tr>
<td><strong>ECC detection</strong></td>
<td>Set by hardware when two ECC errors have been detected.</td>
</tr>
</tbody>
</table>

**Non-maskable interrupt (NMI)**
The Flash memory’s consumption can be reduced when the code is not executed from Flash.
The Flash clock can be gated off in Run and low-power run modes. It can also be configured to be gated off in Sleep and low-power sleep modes. The Flash clock is configured in the Reset and Clock controller. It is enabled by default.
The Flash memory can be configured in Power-down mode during the Sleep and low-power sleep modes.
It can also be configured in power-down mode during Run and low-power run modes, when the code is executed from SRAM. Gating the clock and putting the Flash memory in Power-down mode significantly reduces power consumption.
In Run and low-power run modes, the Flash memory is active. Its clock can be disabled if code is executed from SRAM and the Flash memory is in Power-down mode.

In Sleep and low-power sleep modes, the Flash clock can be disabled and the Flash memory configured in Power-down mode.

In Stop 0, Stop 1 and Stop 2 modes, the Flash clock is off. The content of the Flash interface registers is retained.

In Standby and Shutdown modes, the content of the Flash interface registers is lost and must be reinitialized after exiting the mode.
The performance of the Flash memory is almost linear with the frequency using the ART accelerator. The CoreMark score is 268 at 80 MHz, which corresponds to 3.35 CoreMark / MHz when the Instruction Cache, Data Cache are enabled and the Prefetch buffer is disabled.

In Range 1 at 80 MHz, the performance is 3.32 CoreMark / MHz when the Instruction and Data caches are enabled but the Prefetch buffer is disabled. When the ART is disabled, the performance is only 1.55 CoreMark / MHz. When comparing the energy efficiency, enabling the Cache is very interesting as the results are 24.4 CoreMark / mA when the ART accelerator is enabled, and 13.2 CoreMark / mA when disabled.

In Range 2, energy efficiency rises up to 28.6 CoreMark/mA at 26 MHz.
Related peripherals

- Refer to these peripheral trainings linked to this peripheral
  - System configuration controller (SYSCFG)
  - System protections
  - Reset and clock controller (RCC)
  - Power controller (PWR)
  - Interrupts (NVIC and EXTI)

This is a list of peripherals related to the Flash memory. Please refer to these peripheral trainings for more information if needed.
For more details, please refer to the following document

- AN2606: STM32 microcontroller system memory boot mode – Application note

For more details, please refer to application note AN2606 about the STM32 microcontroller system memory boot mode.
This slide presents the key differences between baseline STM32L47x/48x devices and other devices.

The maximum size of the Flash memory and the number of banks differ for each device.

<table>
<thead>
<tr>
<th>Max Flash size</th>
<th>L49x/4Ax</th>
<th>L47x/48x</th>
<th>L45x/46x</th>
<th>L43x/44x</th>
<th>L41x/42x</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of banks</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

- Of course features (read-while-write capability, dual bank boot) and option bytes related to the availability of 2 banks are not applicable on STM32L41x/42x/43x/44x/45x/46x devices.
- Also, the option bytes for boot are a bit different on STM32L11x/12x/13x/14x/15x/16x/19x/4Ax compared to STM32L47x/48x devices and detailed in the System configuration controller (SYSCFG) presentation.