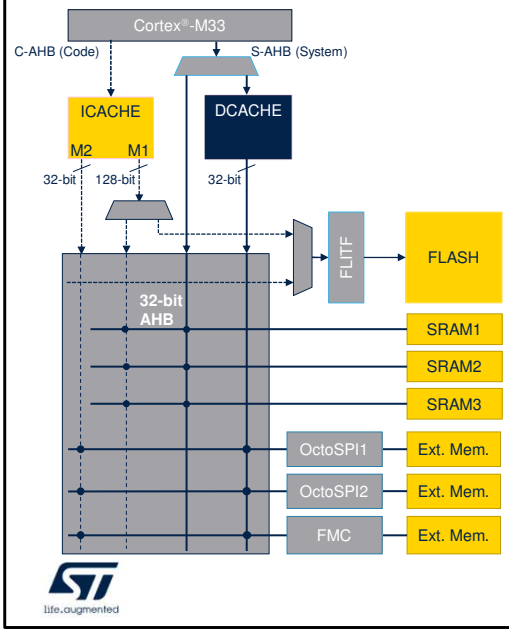
Hello, and welcome to this presentation of the DCACHE module which is embedded in all products of the STM32U5 microcontroller family.

Overview



DCACHE is a 4KB data cache, on S-AHB System bus of Cortex®-M33 to improve performance of data traffic to/from external memories

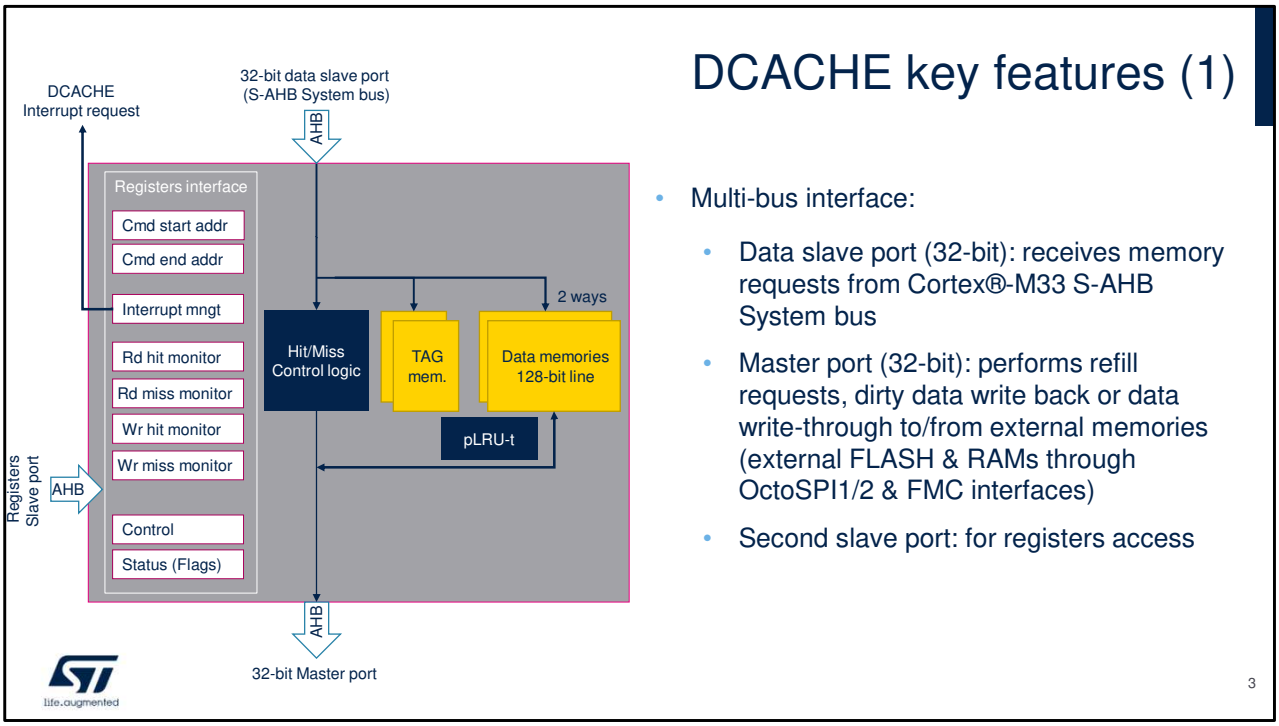
Application benefits

- Higher Performance allowed by close to zero wait state data accesses
- Lower power consumption: hitting data reads/writes to/from small internal DCACHE, rather than bigger and external memories

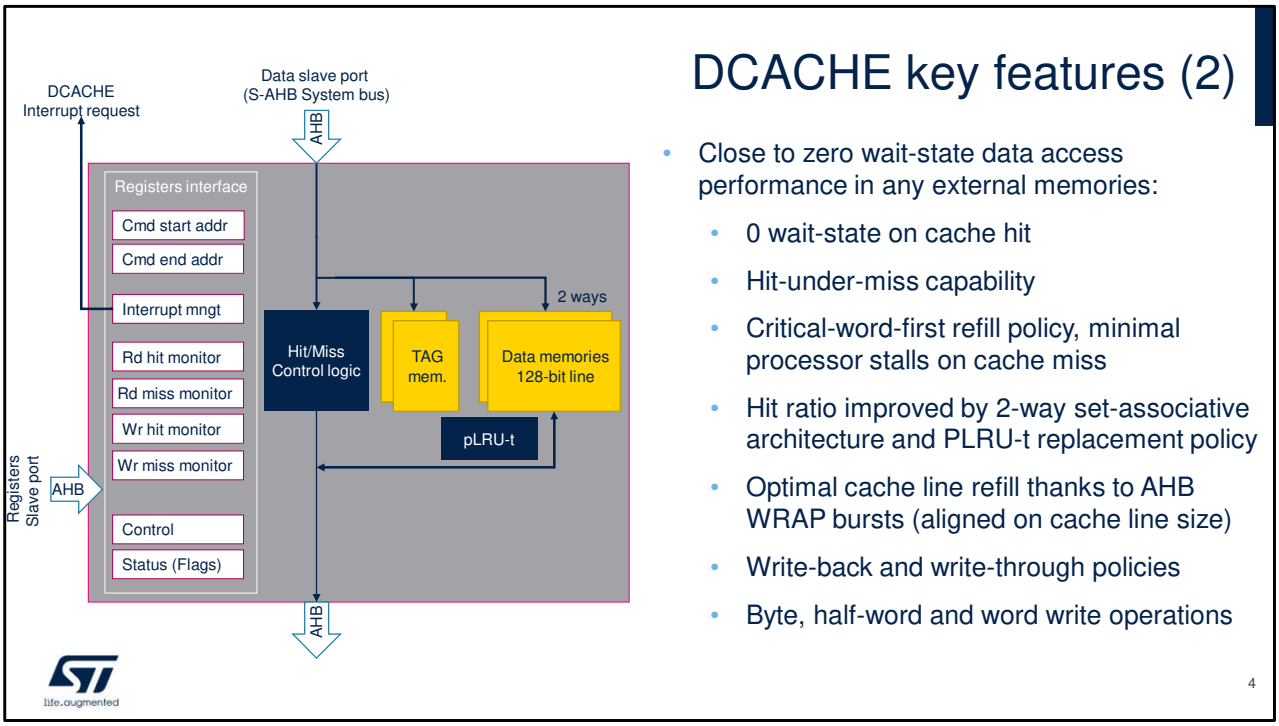
2

The data cache (DCACHE) is introduced on the S-AHB system bus of the Cortex®-M33 processor to improve performance when accessing data and instructions read from and written to external memories through OctoSPI1/2 or FMC interfaces.
S-AHB memory transactions targeting embedded SRAMs or peripherals are not routed towards DCACHE and thus, are not cached.
DCACHE allows a close to zero wait-state performance on data read or write operations in most use cases, due to intrinsic caching operation.
This performance is achieved through the two following features: hit-under-miss support and critical-word-first refill policy.

The concurrence between DCACHE accesses to external memories and core accesses to internal SRAMs also improves the overall performance of the microcontroller.
In the figure, the DCACHE vertical bus used to access external memories is completely independent of the bus used to access internal SRAMs.
The DCACHE autonomously manages cache line refills, cache line evictions and write-through stores to external memories.
The data cache contributes to reducing the consumption of the microcontroller by accessing data in the internal DCACHE, rather than from the larger and then more power consuming main memories.



The multi-bus interface minimizes potential conflicts between memory traffic:
- The 32-bit data slave port receives instruction and data memory requests from the Cortex®-M33 S-AHB System bus
- The 32-bit master port performs refill of missing requests, dirty data write-back or data write-through to/from external memories. These memories are external FLASH & RAMs accessed through OctoSPI and FMC controllers.
- The second slave port is used for registers access.
When an external memory access is marked as non-cacheable by the MPU, the DCACHE is bypassed. The request is forwarded unchanged to the external memory on the DCACHE master port in the same clock cycle.



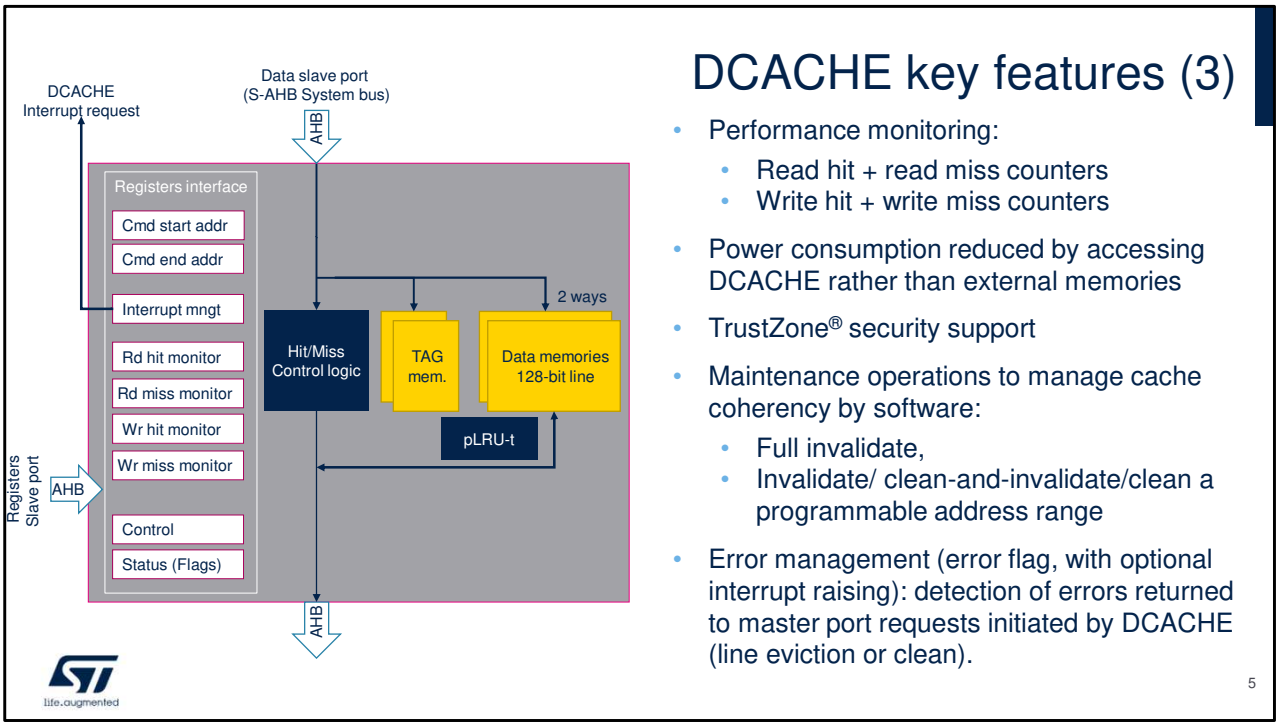
The DCACHE offers close to zero wait states data read/write access performance due to:

- Zero wait-state on cache hit
- Hit-under-miss capability, that allows to serve new processor requests while a line refill (due to a previous cache miss) is still ongoing;
- And critical-word-first refill policy, which minimizes processor stalls on cache miss.

The hit ratio is improved by:

- The 2-way set-associative architecture and
- The pseudo-least-recently-used, based on binary tree (or pLRU-t) replacement policy. This algorithm is a good tradeoff between hardware complexity and performance.

Cache lines are transferred with the critical word first, by implementing the WRAP4 AHB transaction ordering, in order to deliver the data requested by the processor's pipeline first.
Write-back and write-through policies are supported, selection depends on the MPU setting for the addressed data region.
The DCACHE supports all data sizes: byte, halfword and word.



The DCACHE implements performance counters: two 32-bit hit counters, one for read and one for write transactions and two 16-bit miss counters, one for read and one for write transactions.
This performance monitoring allows to analyze and optimize data placement in accordance with cacheability and write-back/write-through policy to achieve the most performant data traffic.
Power consumption is reduced: most data accesses are performed to/from internal cache memory rather to/from bigger and external main memories.
A dedicated Secure-bit in TAG RAM of each cache line prevents non secure requests from hitting secure DCACHE entries.

Software cache coherency is performed through maintenance operations controlled by memory-mapped registers.
These are:

- The full cache invalidation operation, which is a fast command
- The invalidate, clean-and-invalidate and clean operations that are related to a programmable address range.

The data cache is automatically invalidated after a reset.
The address range maintenance operations are typically used to maintain the coherency of buffers shared by DMA channels and the processor core.
These commands are not interruptible, with an end of operation raising a specific flag and possibly an interrupt.
An error flag and possibly an interrupt are raised whenever a bus error is returned to the master port of the DCACHE, when the request is initiated by the DCACHE itself: either a line eviction or a clean operation.
When the master port forwards a request received on the slave port, the DCACHE simply forwards the AHB response received on the master port back to the processor.

SUMMARY

Cache line size	16 bytes	
Cache size	4 KB	
Organization	2-way set associative	
Write & allocate policies	Write through no write allocate	Write-back write allocate
Maintenance operations	Global	Per address range
Clean		✓
Clean & invalidate		✓
Invalidate	✓	✓

life.augmented

6

This table summarizes the characteristics of the data cache:

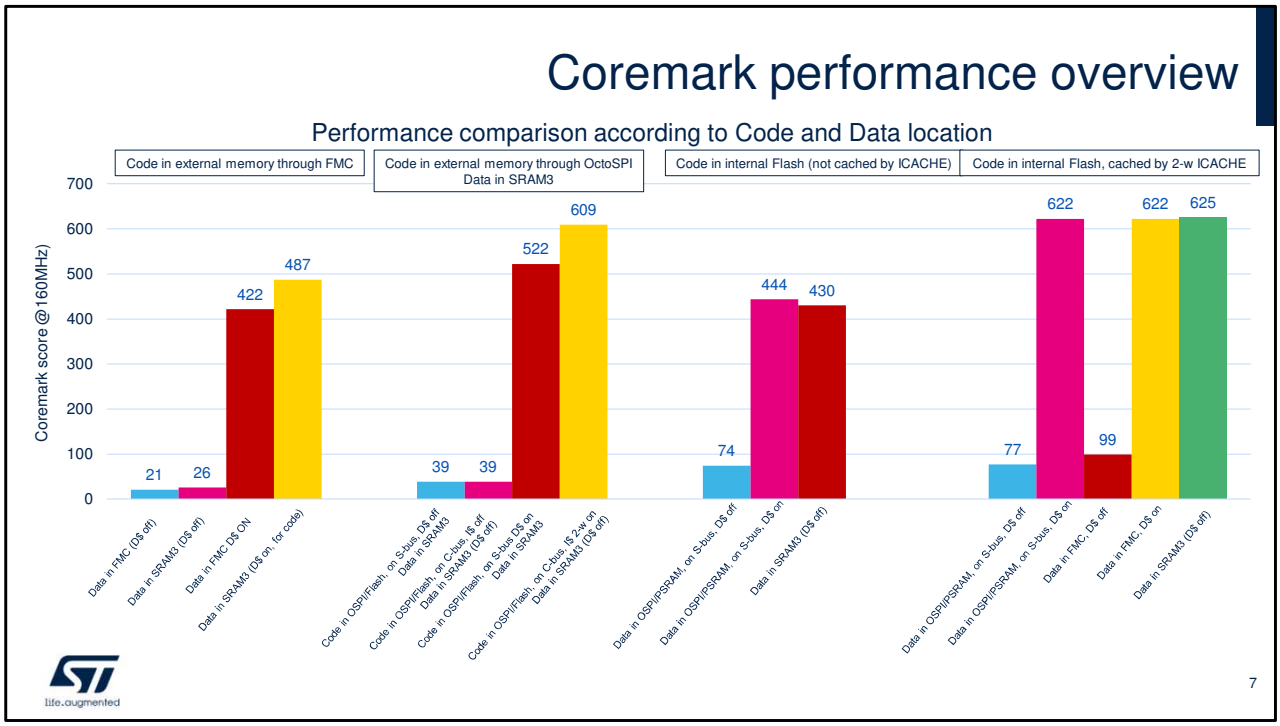
- 16 byte cache line size, transferred using a burst of four words
- 2-way set associative, 4-KB cache

The data cache implements the following write and allocate policies:

- Write-through no write allocate. When a store miss occurs, the DCACHE is bypassed, the data is directly written to memory
- Write-back write allocate. When a store miss occurs, the cacheline is acquired from memory, updated with data received from the processor. The resulting cacheline is then written to the data cache with the dirty bit set.

The supported maintenance operations are:

- Invalidate: global and per address range
- Clean and invalidate, per address range
- And Clean, per address range.



This slide demonstrates the influence of data cache and instruction cache on performance, expressed in Coremark, when the processor core frequency is 160 MHz.
Four scenarios are described, for which the location of data and instructions varies as well as their cacheability in ICACHE and DCACHE.

In the first case, code is in external memory, accessible through the FMC on S-AHB bus, ICACHE is not involved
Lowest performance is obtained when code and data are accessed through the FMC, not cached by DCACHE, and therefore code and data are multiplexed on the S-AHB bus
.
This is slightly better when data is in the SRAM3, but still

transferred on the S-AHB which is also used to fetch code from the FMC.
When DCACHE is enabled, the performance increases a lot, but is still limited by code and data sharing, both on the S-AHB bus and in the DCACHE.
The best performance, which is 487 coremark, is achieved when data is in the SRAM3: code and data still share S-AHB bus, but the DCACHE is dedicated to code storage only.
Code accessible through the FMC could advantageously be cached by ICACHE, through ICACHE address remapping, which would split code and data traffics between C-AHB and S-AHB buses.

In the second case, code is in external memory, accessible through the OctoSPI on the S-AHB bus or on the C-AHB bus
Performance is low when data is in the SRAM3 and code is in the OctoSPI External Flash, accessible through the S-AHB bus and not cached by DCACHE.
Performance is low also when code is in the OctoSPI External Flash accessible through the C-AHB bus and not cached by ICACHE.
In both cases, the score in coremarks is only 39.
The performance is drastically increased when the code accessible through the S-AHB bus is cached by the DCACHE, but the S-AHB bus remains a bottleneck, because it is used to transport both code and data.
The almost optimal performance is achieved when the code accessible through the C-AHB bus is cached by ICACHE. This requires the implementation of address

remapping in the ICACHE. In this case, code is transferred over the C-AHB bus while data is transferred over the S-AHB bus.

In the third case, code is stored in the internal Flash, not cached by ICACHE
When data is stored in OctoSPI PSRAM, the performance is low when data is not cached.
When data is cached in the DCACHE, the performance increases a lot and becomes even better than having the data in SRAM3: 444 Coremark instead of 430.

In the last case, code is in internal Flash, cached by ICACHE configured in 2-way set associative mode.
Performance is low when data stored in OctoSPI PSRAM is non cacheable and a bit better when data is stored in an SRAM accessible through the FMC but still non cacheable.
In both cases, marking the address range containing this data as cacheable, leads to the same good performance of 622 coremark.
The best performance is obtained by having data in SRAM3 and instructions in cacheable internal flash.

DCACHE errors and interrupts

Interrupt vector	Interrupt event	Event Flag	Interrupt Enable bit	Interrupt Clear bit	Description
DCACHE	Functional Error	DCACHE_SR[ERRF]	DCACHE_IER[ERRIE]	DCACHE_FCR[CERRF]	Error on master port transaction initiated by DCACHE itself (eviction or clean operation)
	End of Busy State	DCACHE_SR[BSYENDF]	DCACHE_IER[BSYENDIE]	DCACHE_FCR[CBSYENDF]	When cache-busy state is finished, at the end of the cache full invalidate operation
	End of Cache Operation	DCACHE_SR[CMDENDF]	DCACHE_IER[CMDENDIE]	DCACHE_FCR[CCMDENDF]	When command-busy state is finished, at the end of the cache range operation (invalidate and/or clean)

DCACHE does not manage AHB bus errors on Master port transactions that result from a data request on the slave port (that received the initial Core S-AHB bus transaction), but propagates these errors back to the Slave port

life.augmented

8

The three sources of DCACHE global interrupt are:

- Error detection on data request initiated by the DCACHE itself, either for dirty cache line eviction or clean operations, which sets the ERRF bit in the DCACHE status register
- End of full Invalidate operation, which sets the BSYENDF bit in the DCACHE status register
- End of cache range maintenance operation (invalidate, clean & invalidate, or clean), which sets the CMDENDF bit in the DCACHE status register.

There is no DCACHE management of errors occurring on a Master port request, when this request results from an initial S-AHB data request issued by the processor. Instead this erroneous response is propagated through DCACHE

back to the Cortex-M33.
A typical case is an erroneous refill request initiated by an initial data request that misses in cache.

Low-power modes

DCACHE clocked by the Cortex®-M33 S-AHB bus clock.

- Same clock domain as Cortex®-M33 core: same clock frequency and same behavior regarding the power modes

Low power mode	State of the DCACHE
Run	Active
Sleep	Active
Stop	Frozen, DCACHE registers content is kept Option: a dedicated control bit in Power Controller to power-down DCACHE in Stop mode
Standby	Powered-down The peripheral must be reinitialized after exiting Standby mode

When disabled, DCACHE is bypassed and internal TAG and Data memories are not accessed

- Almost no power consumption in DCACHE, with the drawback that each data is read from/written to the more power consuming main (external) memory

life.augmented

DCACHE is clocked at the same frequency as the Cortex M33 core, because the DCACHE only caches data requested by the Cortex-M33.
Consequently the DCACHE and the Cortex-M33 have the same state in the various low power modes.
When the microcontroller is in stop mode, the user can decide to power-down the DCACHE, which may require a complete clean and invalidate maintenance operation.
When the DCACHE is disabled, the DCACHE is bypassed. The system bus input requests are just forwarded to the master port.
So, the DCACHE consumes less, because TAG and Data memory are not accessed, but each data is read from/written to the more power consuming targeted (external) main memory.
To reduce power consumption, the performance monitor is disabled by default.

Thank you

life.augmented

In addition to this presentation, you can refer to the following presentations:

- Instruction cache
- Security
- FMC
- OCTOSPI.