

ALU = Arithmetic Logic Unit

SIMD = Single instruction, multiple data

MPU = Memory Protection Unit

MAC = multiply–accumulate operation

LSU = load store unit

DPU = data processing unit

DTCM & ITCM = The memory system includes support for the connection of local Tightly Coupled Memory called ITCM (16K) and DTCM (64K)

STB = store buffer

BUI = Bus Interface Unit

TCU = Tightly-Coupled interface Unit

EPPB = External Private Peripheral Bus-The APB External PPB

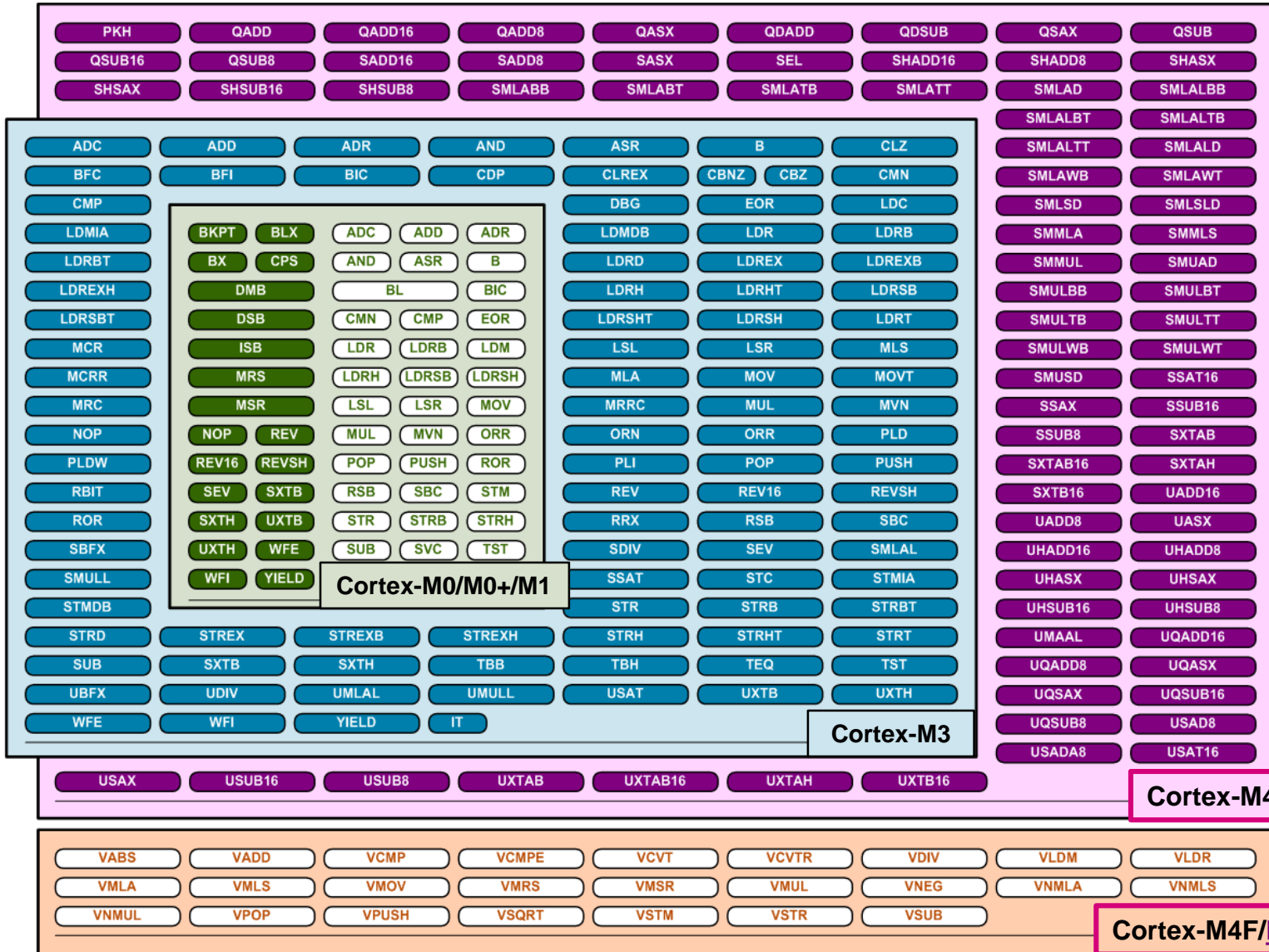
Superscalar architecture first appeared on Intel Pentium 5 in 1993 => it means it is able to process multiple instructions in parallel (in our case, Cortex M7, up to 2, that's why **dual-issue**)

- 32-bit RISC architecture
- Designed to be fully programmed in **C-language**
- Very low interrupt latency (**deterministic**)
- **Small code memory footprint**
- **High energy efficiency** and support for system low-power modes
- Single cycle multiply
- RTOS support and 24-bit SysTick timer

Included in every STM32

Cortex-Mx instruction set

4

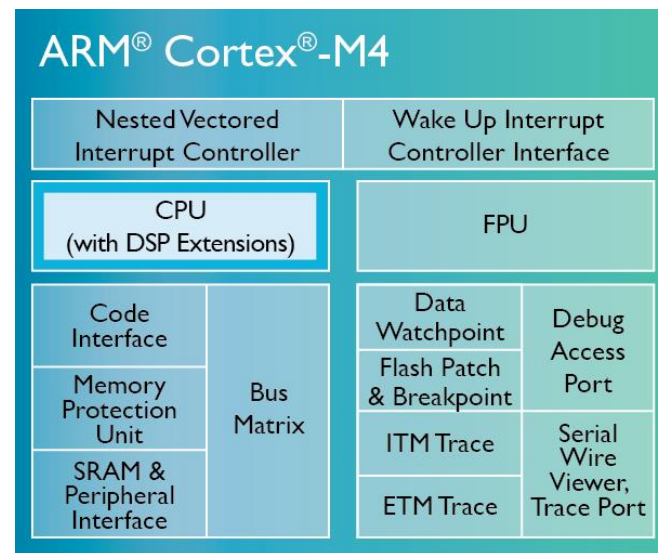


Cortex-M4 Processor Overview

5

- **ARMv7E-M** Architecture
- **Harvard** architecture, **3-stage pipeline**
- **DIV in 12-cycles max, SIMD instructions**
- **Memory Protection Unit (MPU)**
- (Optional) Single Precision floating point (float)

⇒ Included in current STM32 based on ARM Cortex-M4



12-cycles interrupt latency

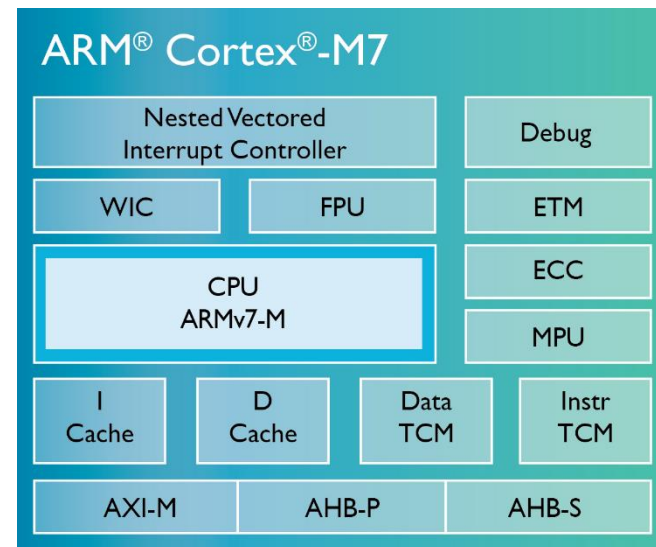
SIMD - Single instruction, multiple data

MPU - Memory Protection Unit

Cortex-M7 Processor Overview

6

- **ARMv7E-M Architecture**
- **Harvard architecture, 6-stage pipeline**
- **Dual-issue superscalar architecture!**
- **DIV in 12-cycles max, SIMD instructions**
- **Memory Protection Unit (MPU)**
- **Floating point unit**



⇒ Included in current STM32 based on ARM Cortex-M4 **and Cortex-M7**

12-cycles interrupt latency still

One step closer to DSPs

Load and store in parallel with arithmetic

Zero overhead loops

One step closer to Real-Time processors

Tightly Coupled Memories

AXI-M interface with Cache memory

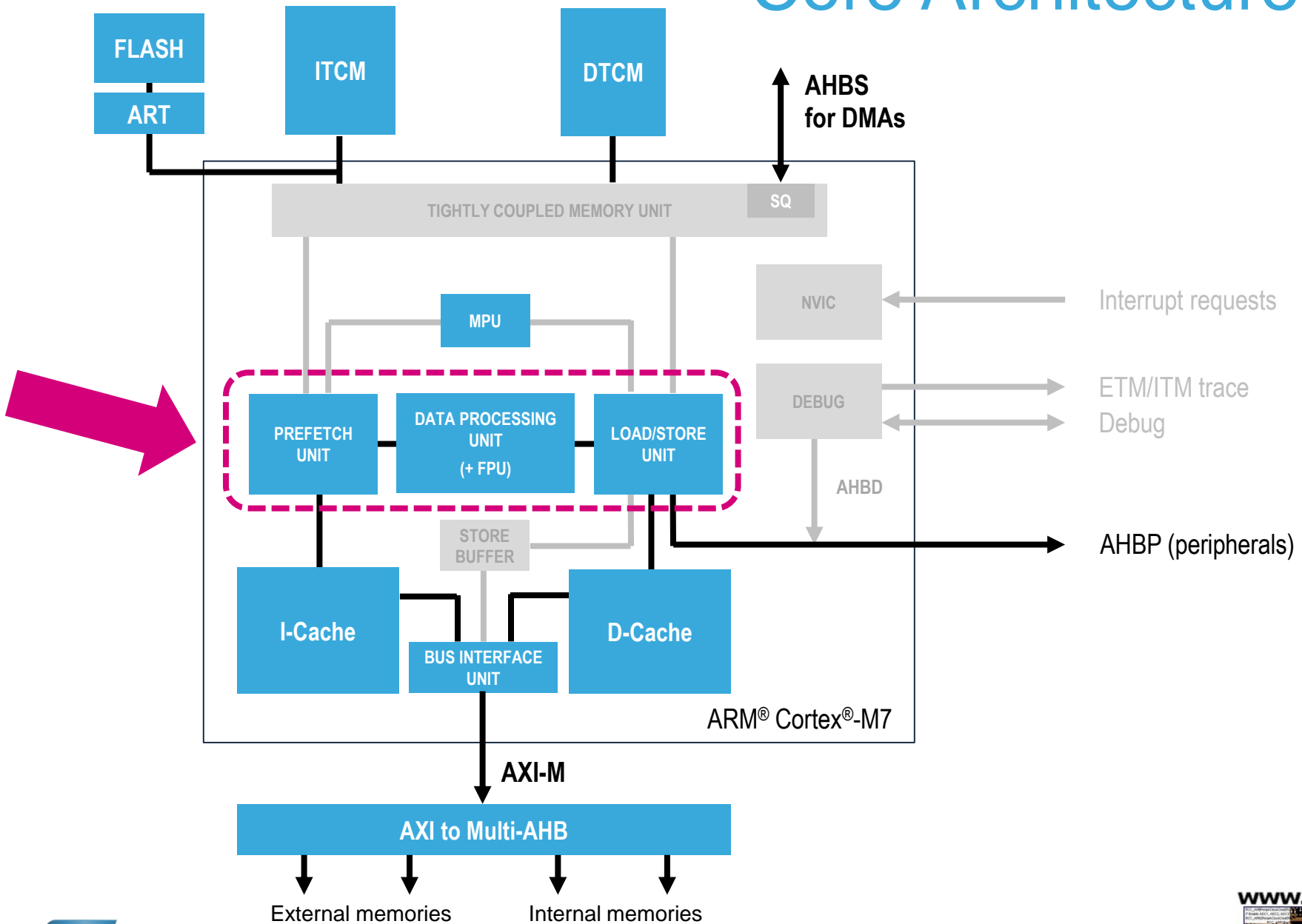


One step closer to Digital-Signal Processor

(but still universal easy-to-use STM32 MCU)

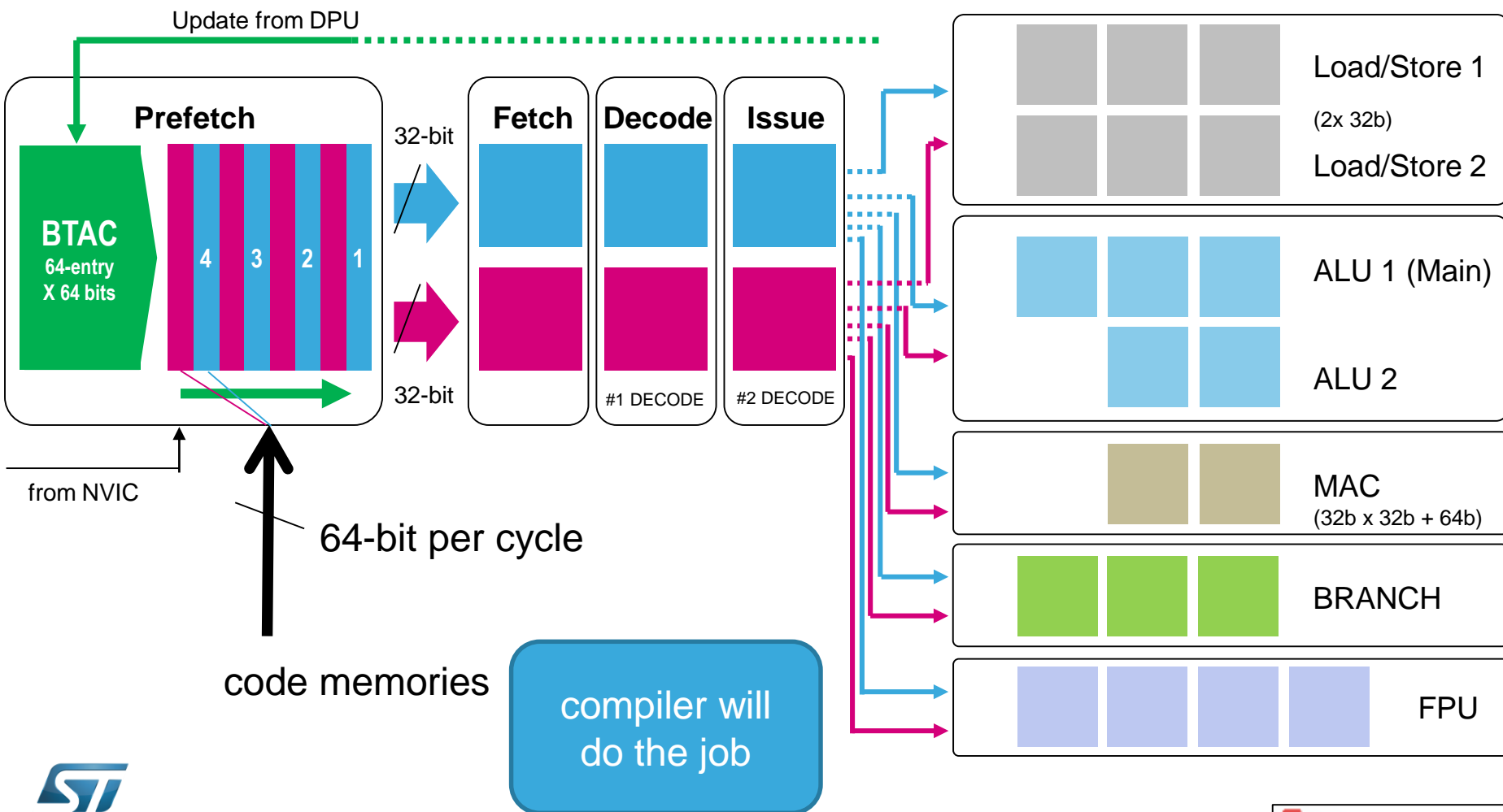
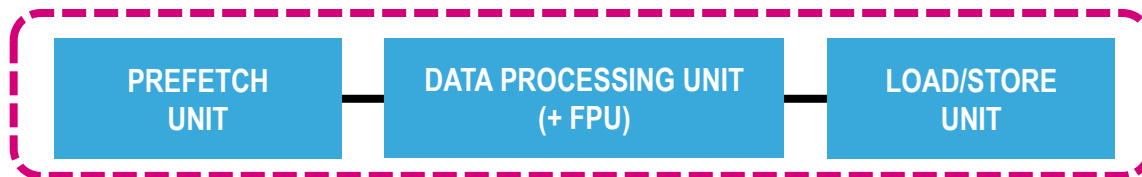
Core Architecture

8



ARM Cortex-M7 → dual-issue

9



Load and store in parallel with arithmetic

10

- **Cortex-M4**

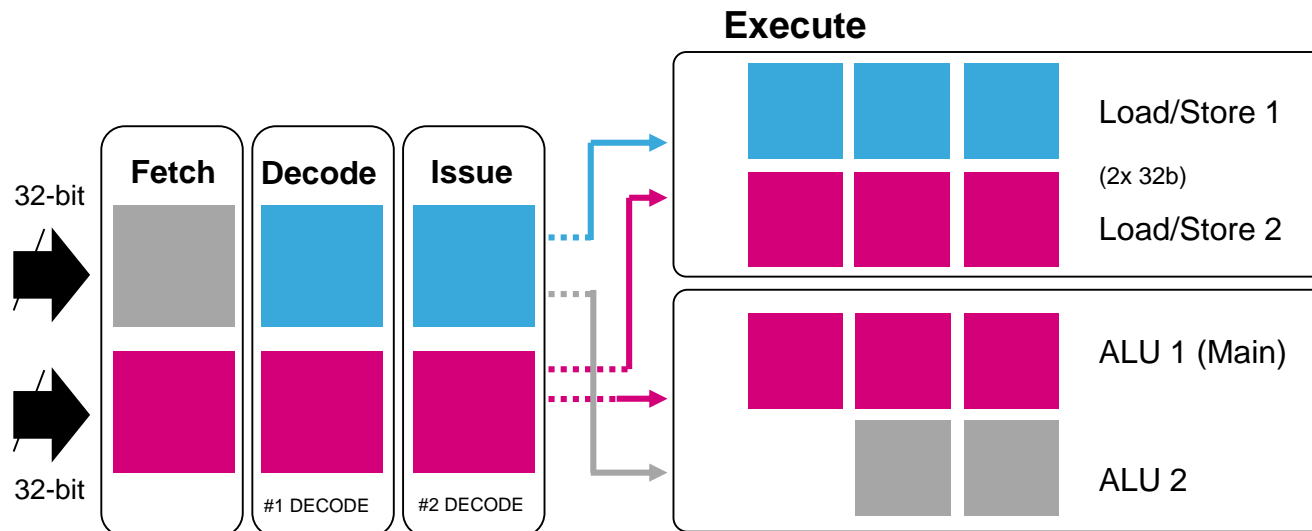
- Single load or store instructions take 2 cycles
- N consecutive loads or stores take N+1 cycles

Group as many loads and stores together

- **Cortex-M7**

- Load and store operations can occur in parallel with math
- Memory access possible without penalty

Interleave memory accesses with computation



Load and store in parallel with arithmetic

11

• Cortex-M4

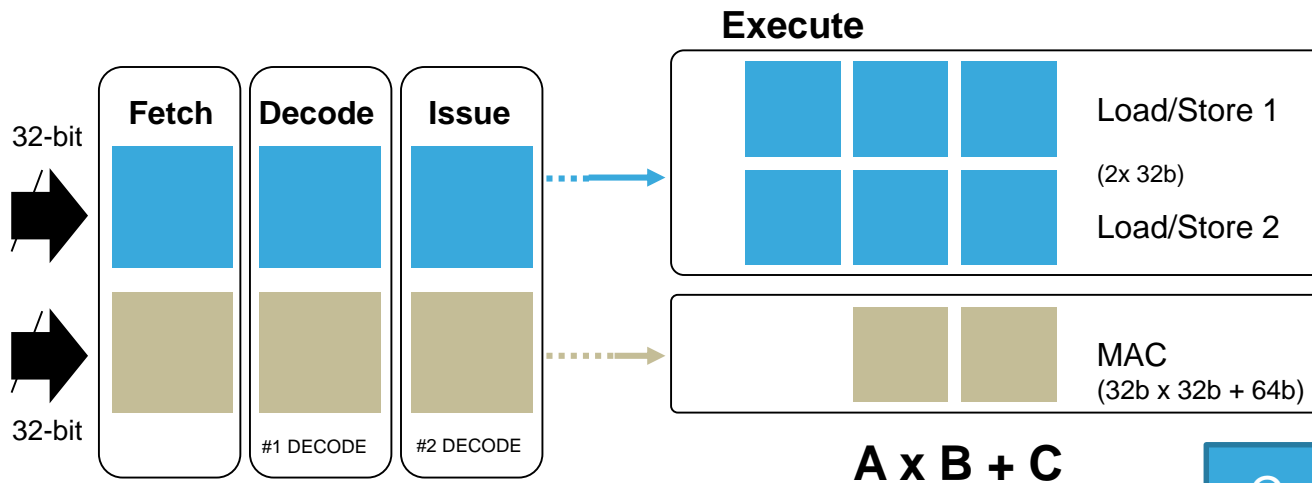
- Multiplication or addition takes 1 or 2 cycles depending if the result is used or not by the next instruction
- Multiply and Accumulate takes 2 to 4 instructions

Use individual multiplies or adds and reorder to avoid stalls

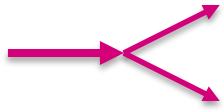
• Cortex-M7

- Multiplication or addition takes 1 or 2 cycles depending if the result is used or not by the next instruction
- MAC requires 1 cycle

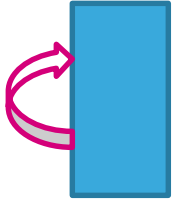
Use MACs as much as possible



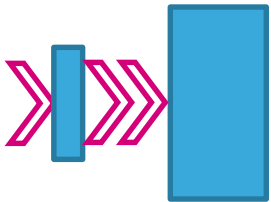
Compiler job!



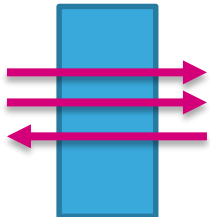
1. Superscalar → 2 instructions at 1 cycle



2. Branch in 1 cycle



3. Cache system to compensate slow memories



4. High system bandwidth for every application



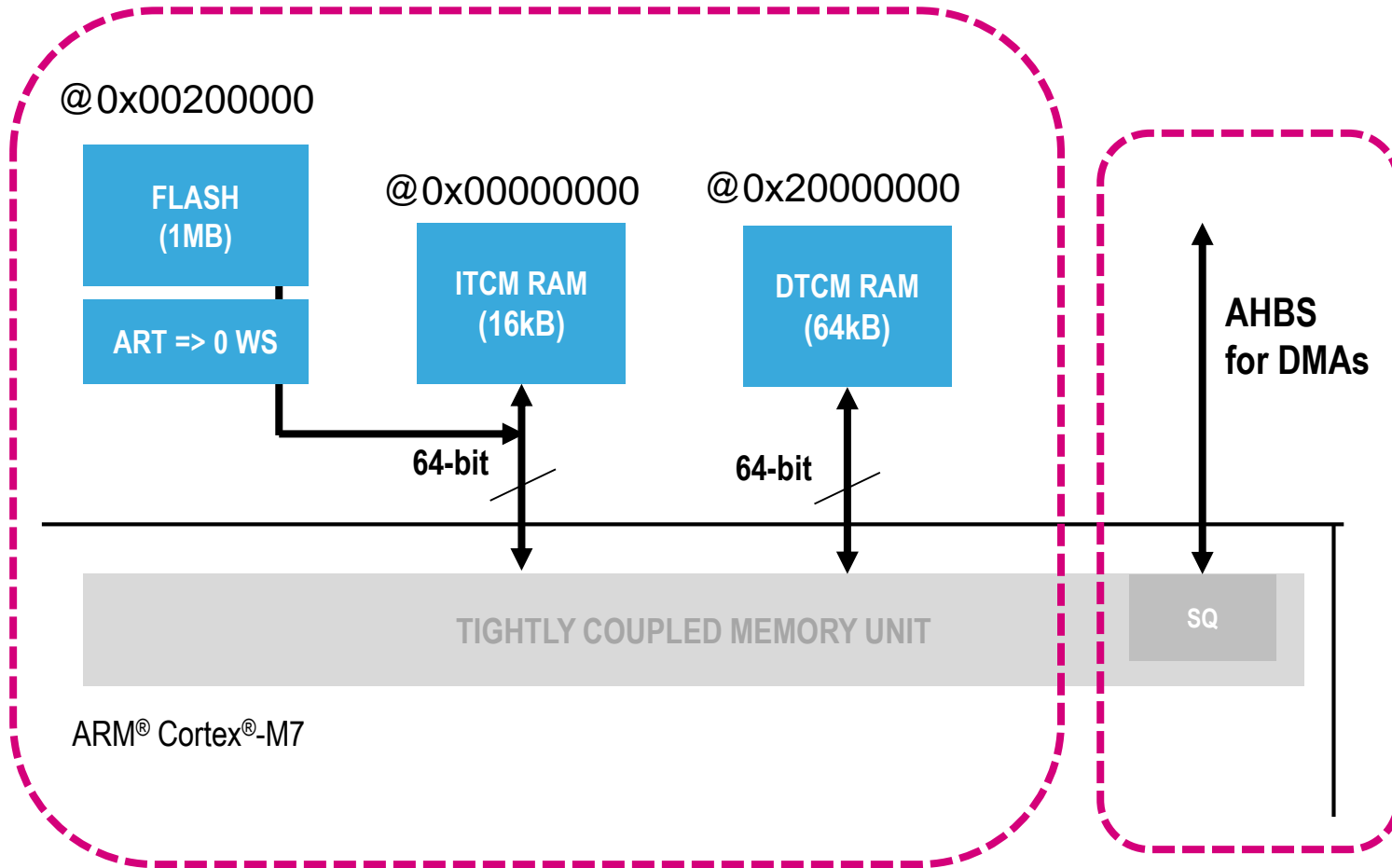
One step closer to Real-Time Processor

(but still universal easy-to-use STM32 MCU)



Tightly Coupled Memories (TCM)

15



TCM memories are strictly 0 wait states

Tightly Coupled Memories (TCM)

16

ITCM RAM
(16kB)

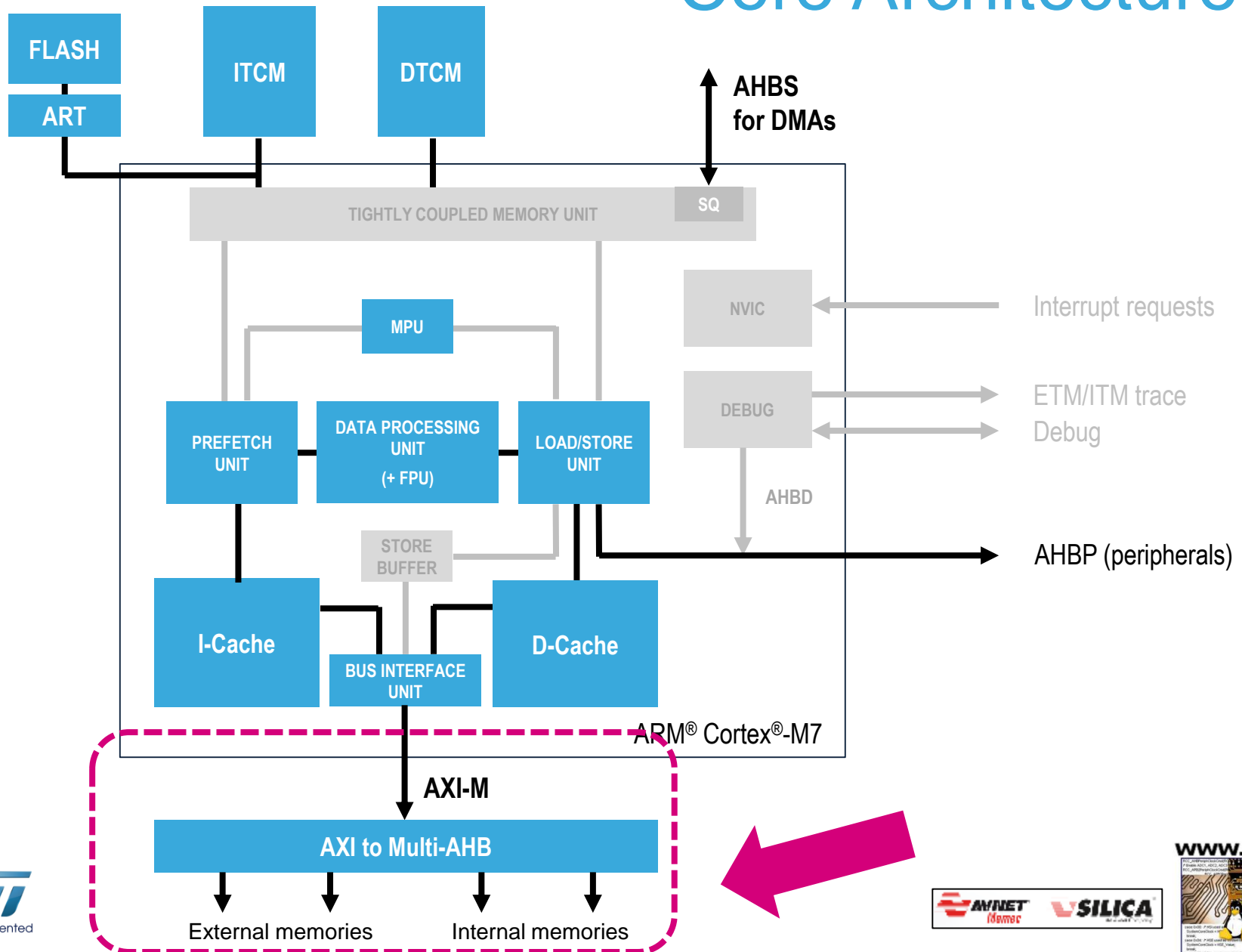
- Critical code
- Interrupt service routines
- Highly deterministic

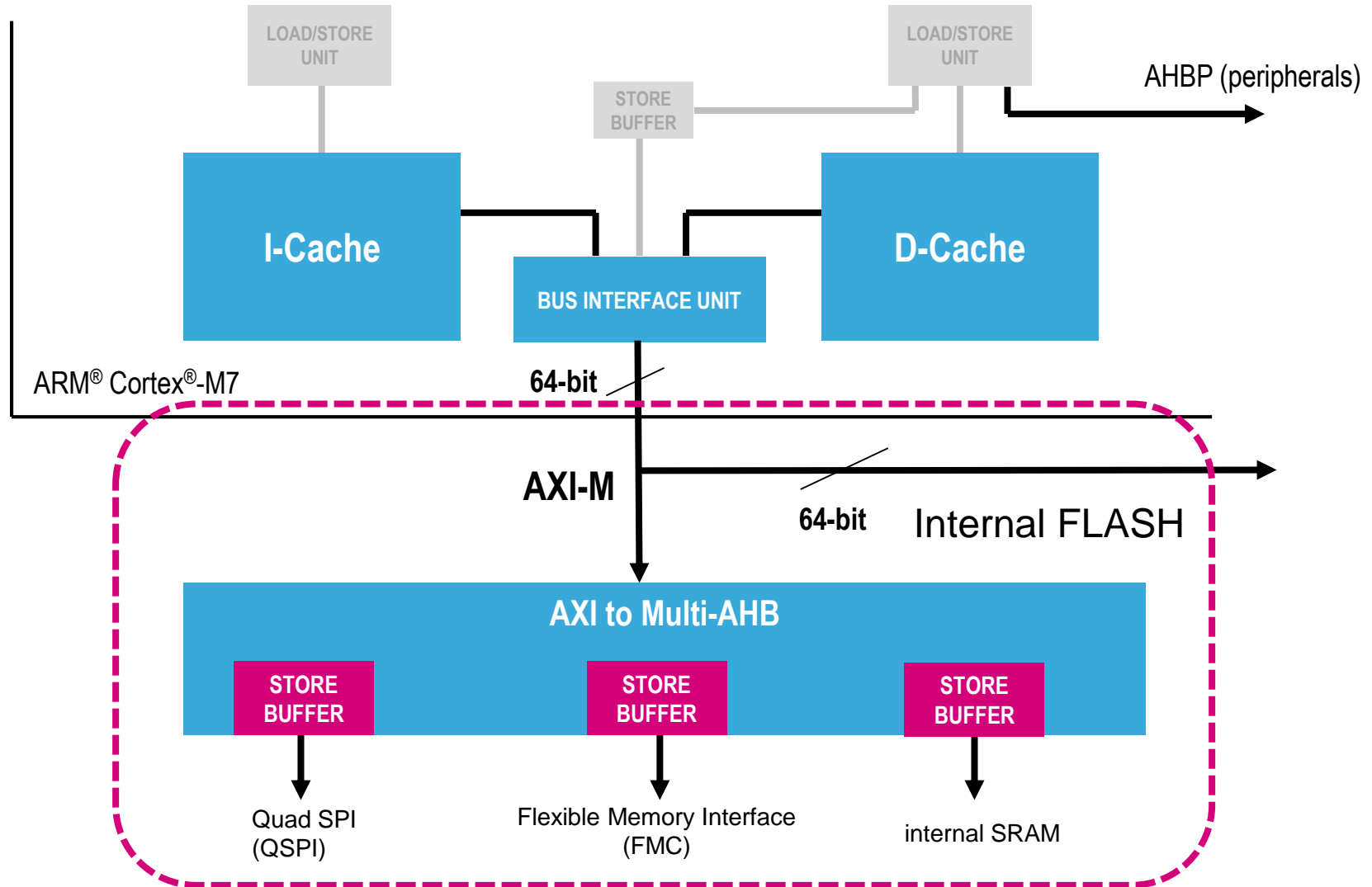
DTCM RAM
(64kB)

- Frequently used data
- Stack/Heap
- DSP coefficients

Core Architecture

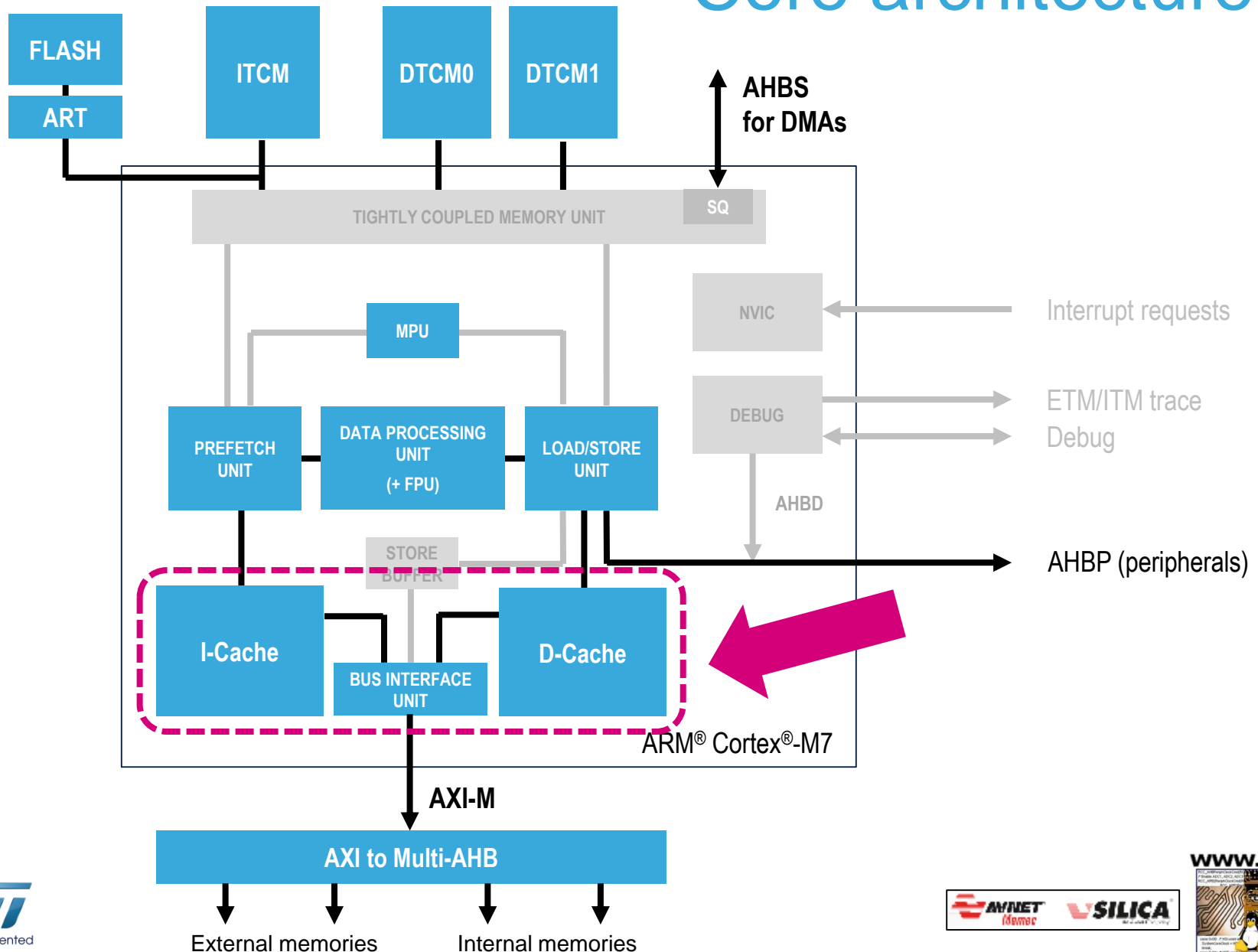
17





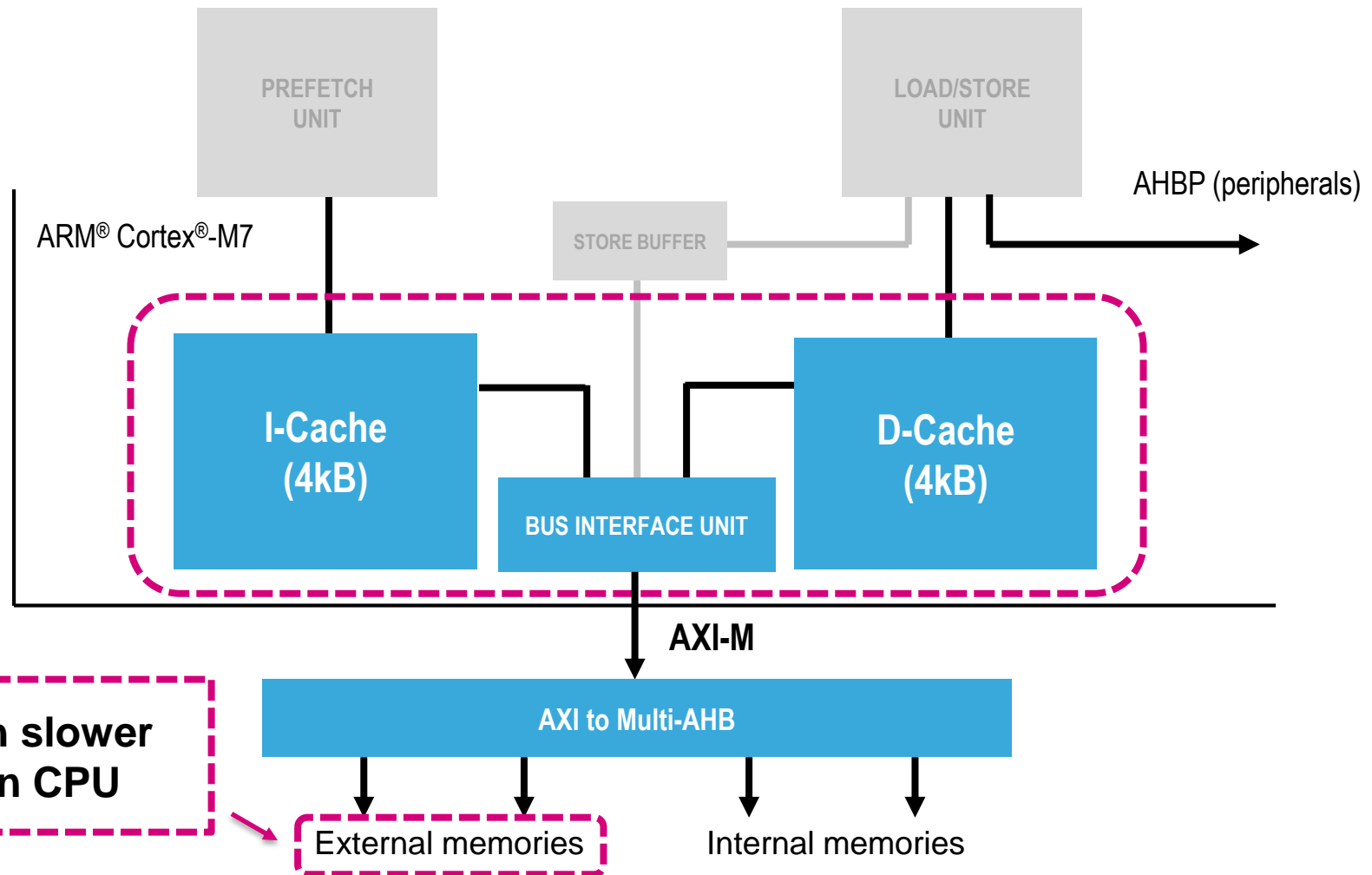
Core architecture

19



L1 Cache memory on AXI-M

20



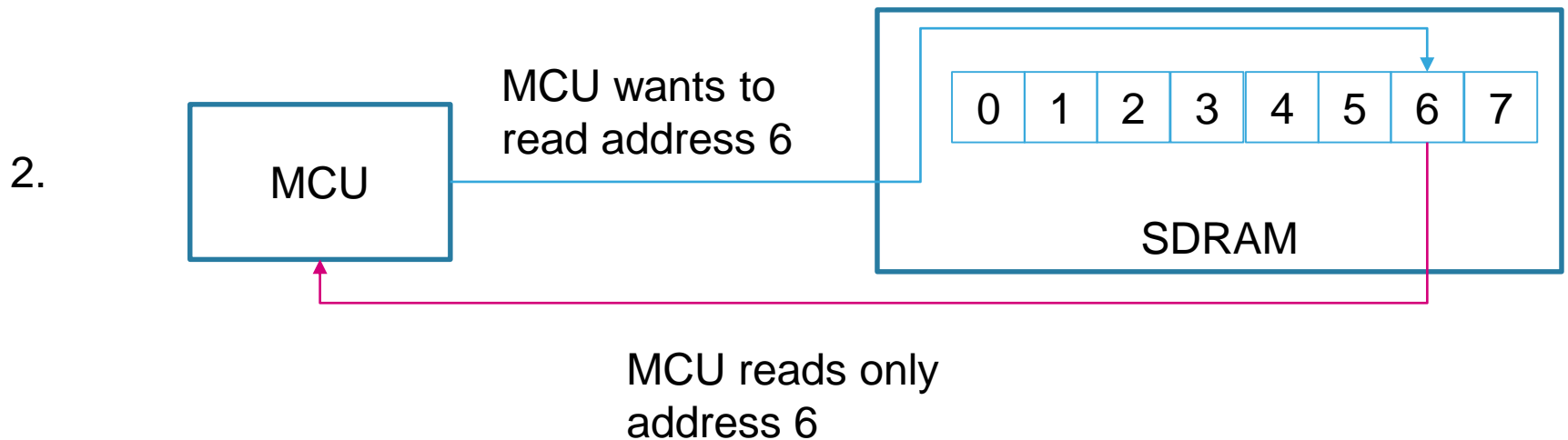
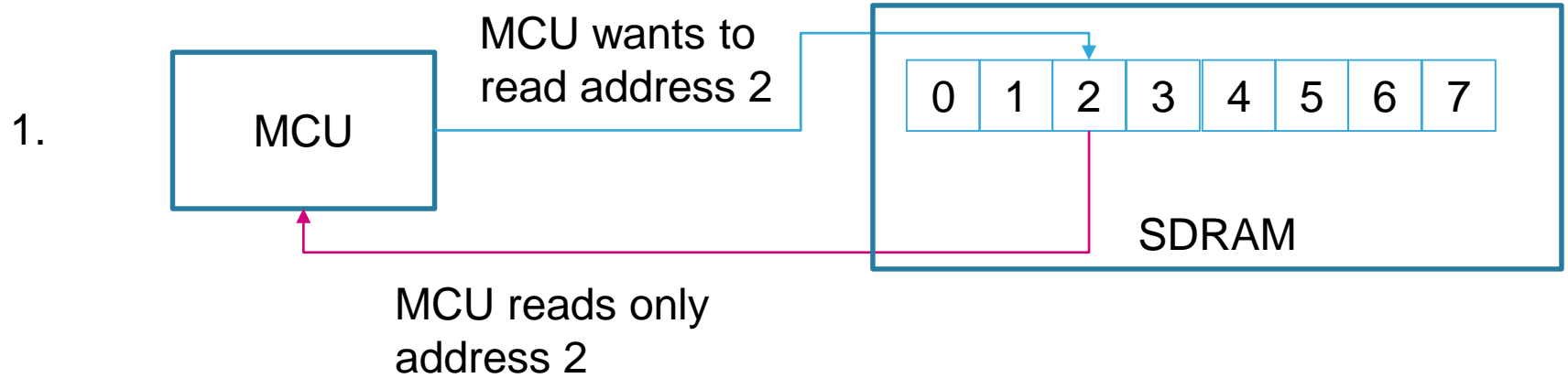
- Result is:

Data placement	D-Cache	Cycles for 500 iterations
DTCM	N/A	1018
SDRAM	Disabled	6604
SDRAM	Enabled 1 st run	2954
SDRAM	Enabled 2 nd run	1017

Conditions: System clock = 200MHz

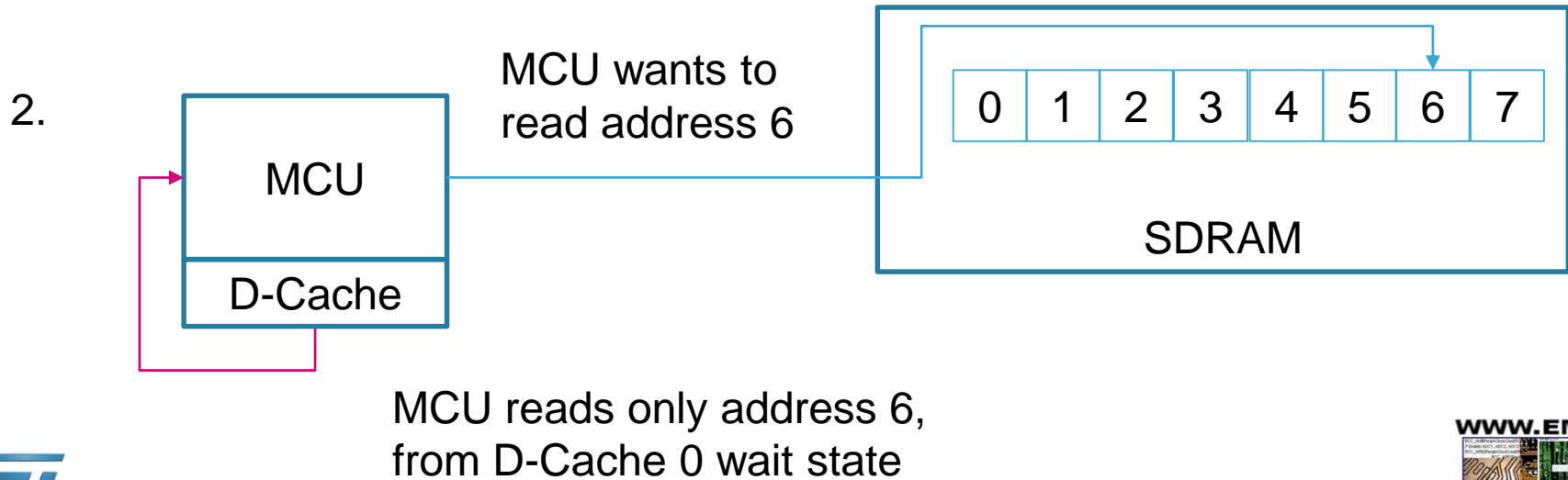
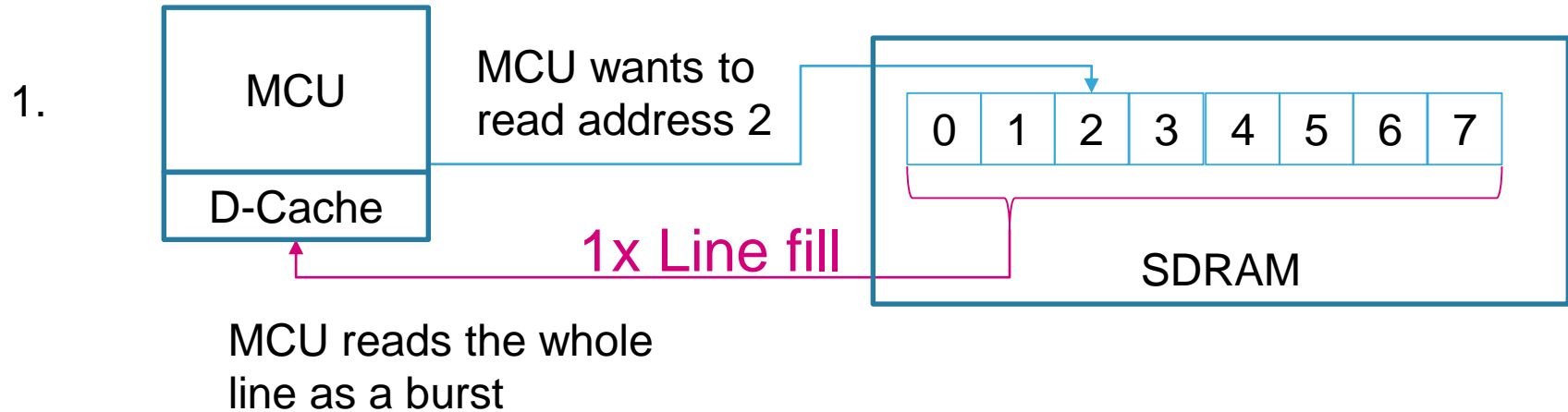
Random read without cache

22



Random read with D-Cache

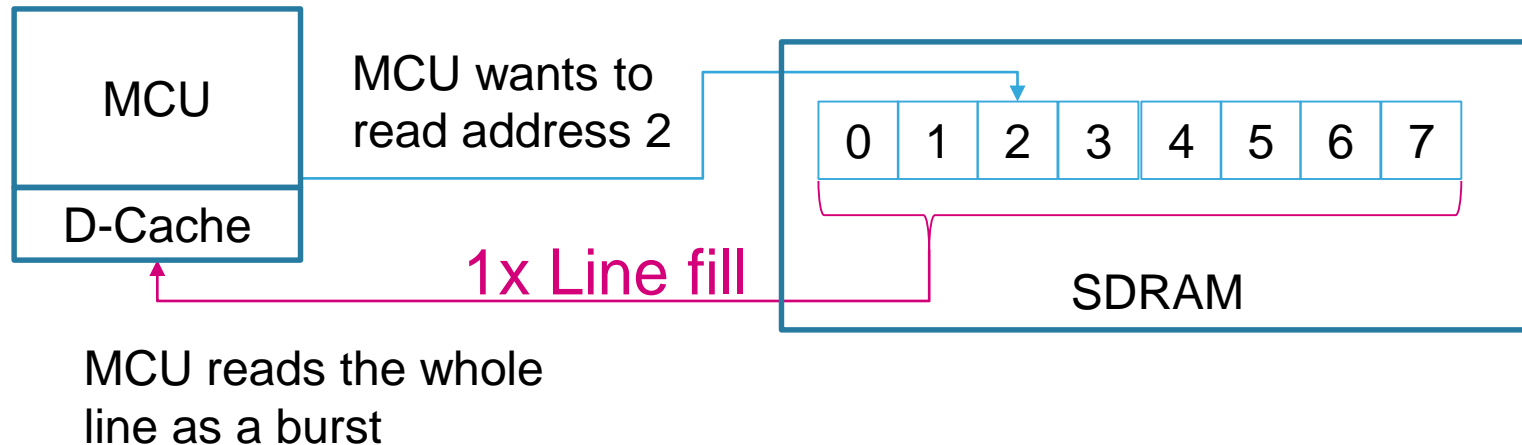
23



Data cache – line fill

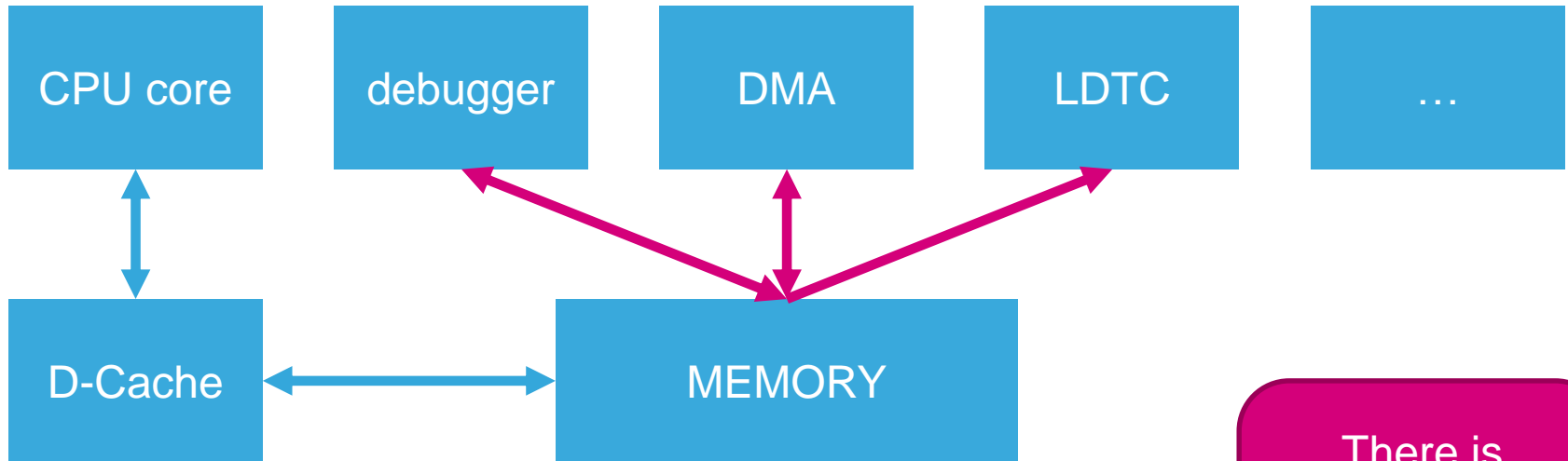
24

- When there is **cache miss**, there is a complete cache **line fill** action.
- The cache loads full line of data (8 words) in burst mode, which is more efficient for SDRAM than random read



Data cache – coherency

25



There is
CMSIS
function to do
all steps

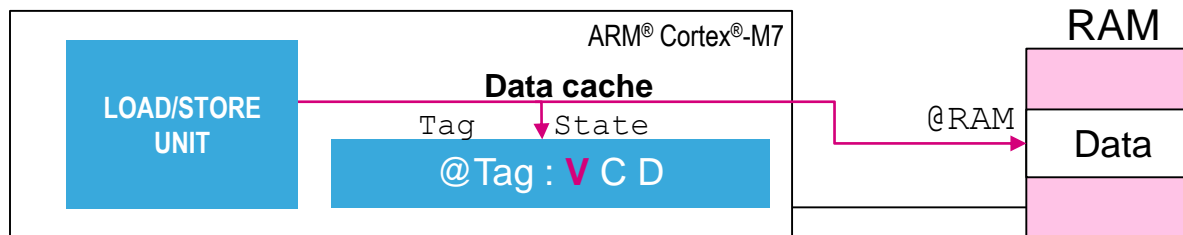
To overcome cache coherency issue

1. Don't use cache at all
2. **Invalidate cache** when you pass control to other master
3. In case of write only, you can use **Write Through** policy

- Description of the cache policy:

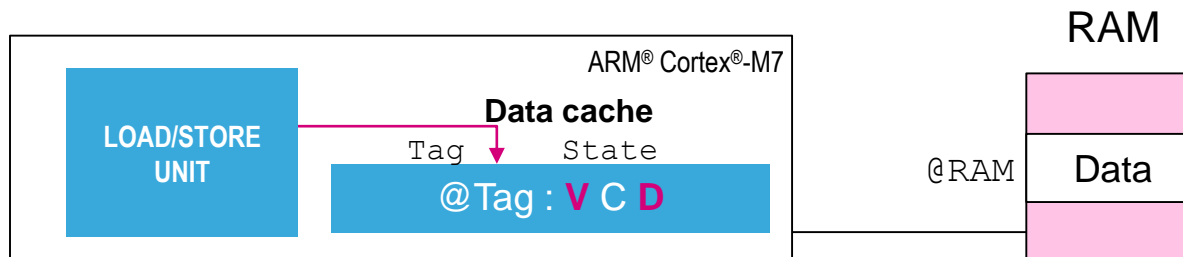
1. Write-through policy (read or write allocate)

- Data writes occur simultaneously to the cache and to next level memory



2. Write-back policy (read or write allocate)

- Data write occurs only to the cache, next level memory written on eviction



- The internal or external flash has latency. So we may get penalty if we execute the code from it.
- To overcome this problem – let's use instruction cache
 - **ART on ITCM bus**
 - **I-Cache on AXI bus**
- Once executed instruction will be 0-wait state next time.

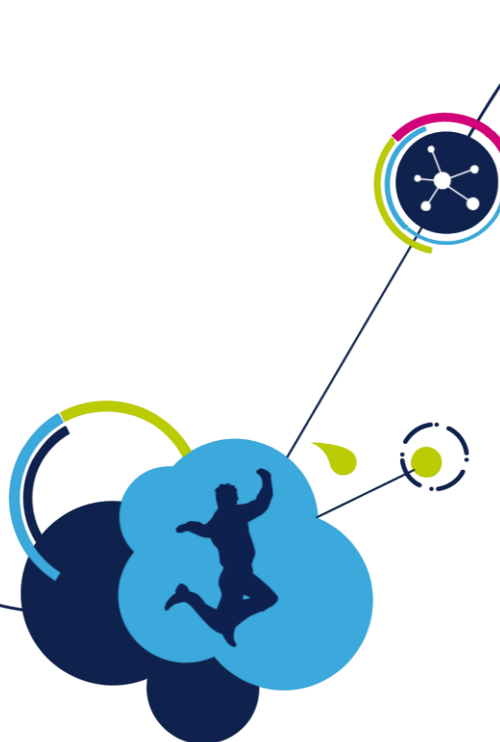
- Result is:

Code placement	System Clock[MHz]	I-Cache or ART	Cycles for 500 iterations
AXI	16	OFF	1058
AXI	16	ON	1054
AXI	200	OFF	1095
AXI	200	ON	1078
ITCM	16	OFF	1035
ITCM	16	ON	1035
ITCM	200	OFF	1075
ITCM	200	ON	1064

- Result is:

Code placement	System Clock[MHz]	I-Cache or ART	Cycles for 500 iterations
ITCM - RAM	200	N/A	3526
AXI	200	ON	3588
ITCM – flash	200	ON	3568
M4 - Flash	16	ON	6522

- ART and I-cache usage result in 0 wait state execution

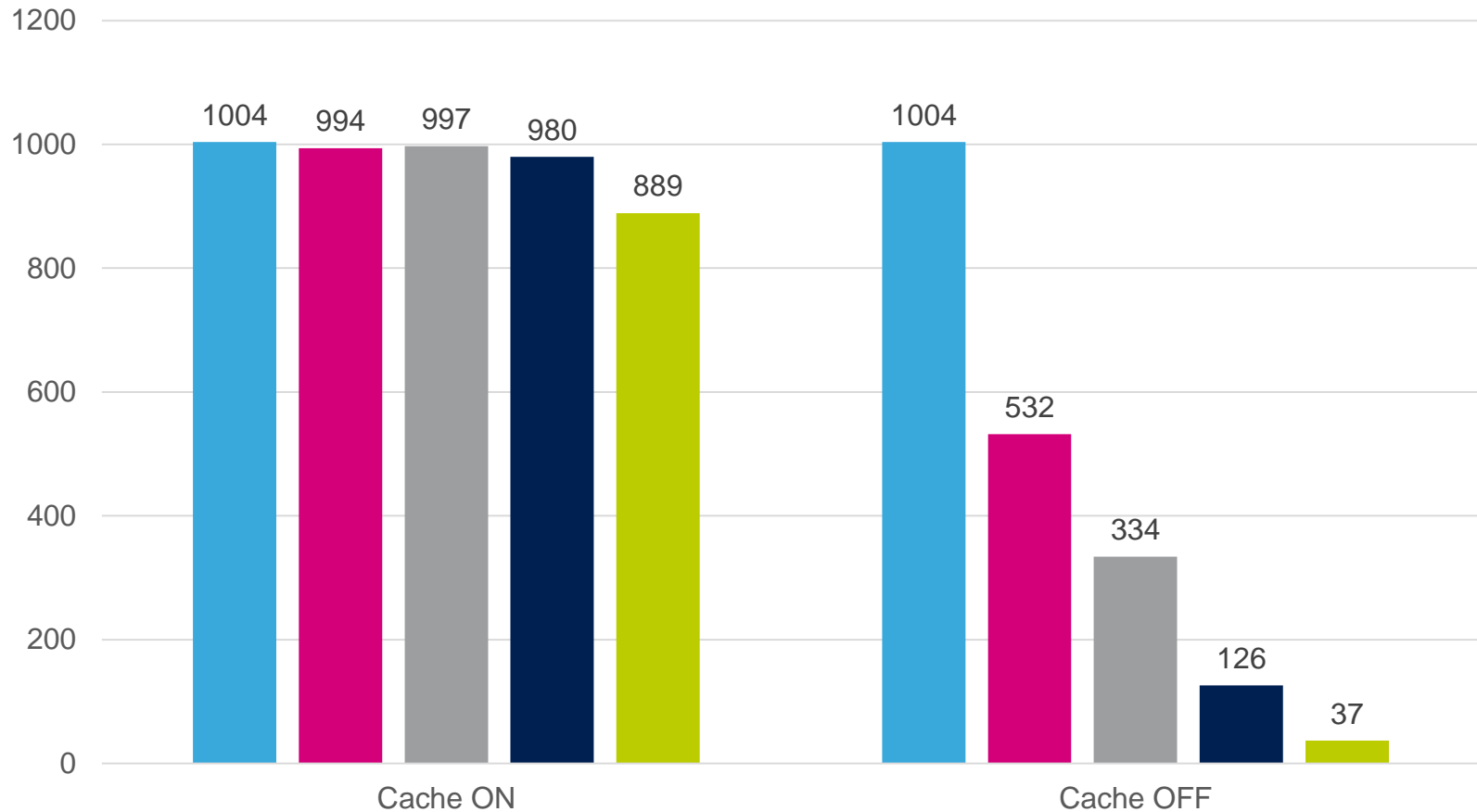


Coremark®

- As you can see the MCU performance is quite difficult to assess, because it depends on how the code looks like.
- compare different architectures, with different settings → necessity of “standard” code.
- Coremark® is now industry standard of CPU benchmarking.
- <http://www.eembc.org/coremark/>
- Sources available



Iterations/s = Coremark[®] score

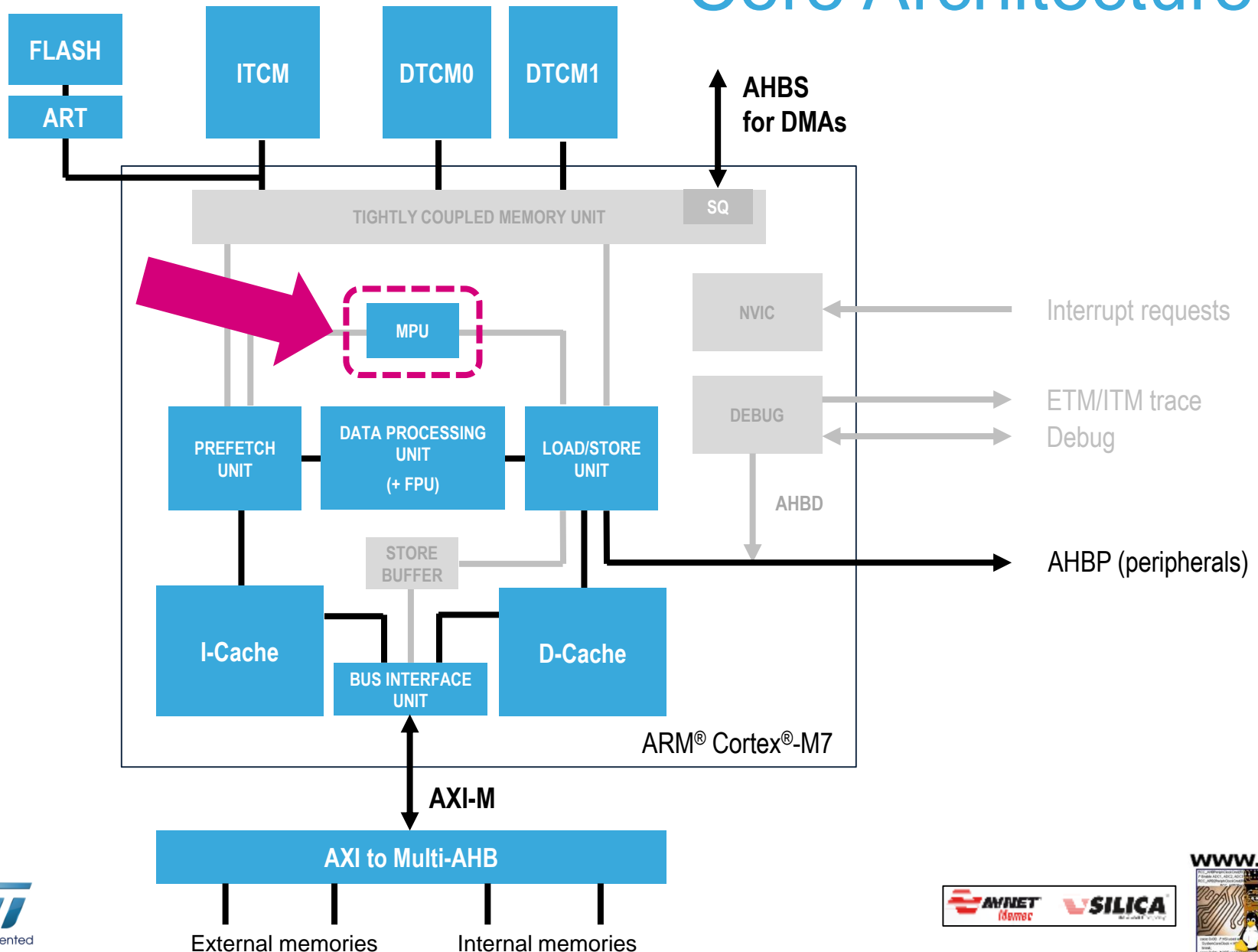


■ RAM ITCM
■ Flash AXI

■ Flash ITCM
■ SDRAM 16bit

Core Architecture

33



Memory Protection Unit and Cache

34

- MPU attributes settings **affect** AXI-M/Cache memory system **behavior**
- **8 independent memory regions**
 - Cache **On / Off**
 - Cache **Policy**

- Allocate Policy

- **Read and Write Allocate** → cache is filled on miss both read and write
 - Best for overall performance
- **Read Allocate** → cache is filled only on read miss
 - Specific use cases (e.g. memset())

- Write Policy

- **Write Back** → writes go only to cache
 - **Write through** → lower performance than WB.
- For I-Cache, all allocate policies are treated as read-allocate (no instruction write)

Memory management attributes

36

TEX	C	B	Description	Memory type
000	0	0	Strongly Ordered	Strongly Ordered and shared
000	0	1	Shared Device	Device
000	1	0	Cacheable, write-through, no write allocate	Normal
000	1	1	Cacheable write-back, no write allocate	Normal
001	1	1	Cacheable write-back; write and read allocate	Normal
010	0	0	Non-shared device	Device

MPU Default attributes

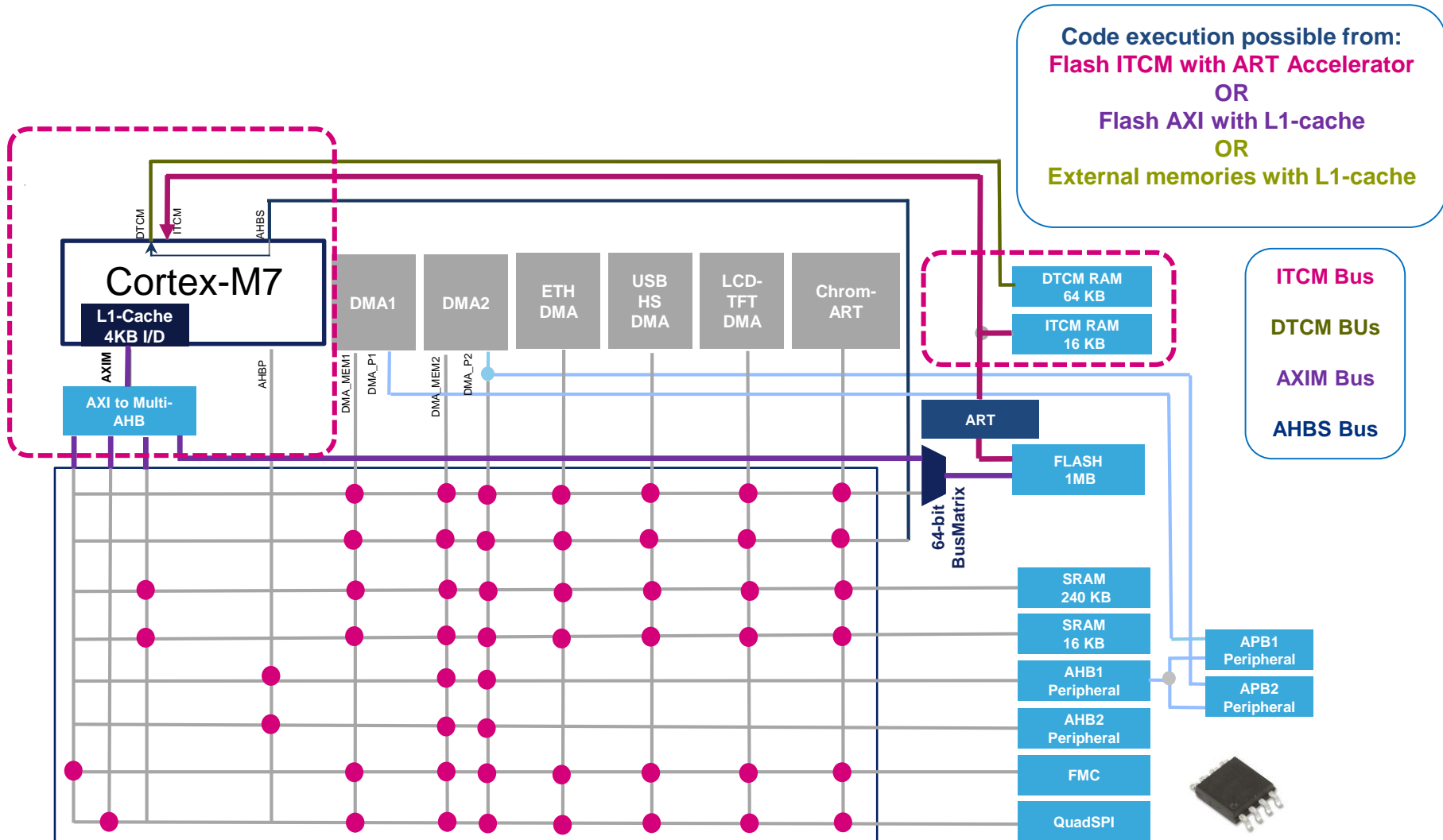
37

Address	Name	Memory Type	XN?	Cache	Description
0x0000 0000 0x1FFF FFFF	Code	Normal		WT	Typically ROM or flash memory. Memory required from address 0x0 to support the vector table for system boot code on reset
0x2000 0000 0x3FFF FFFF	SRAM	Normal	-	WBWA	SRAM region typically used for on-chip RAM
0x4000 0000 0x5FFF FFFF	Periph.	Device	XN	-	On chip peripheral address space
0x6000 0000 0x7FFF FFFF	RAM	Normal	-	WBWA	Memory with write-back, write allocate cache attribute for L2/L3 cache support
0x8000 0000 0x9FFF FFFF	RAM	Normal	-	WT	Memory with write-through cache attribute
0xA000 0000 0xBFFF FFFF	Device	Device, Shareable	XN	-	Shared device space
0xC000 0000 0xDFFF FFFF	Device	Device, non-shareable	XN	-	Non-shared device space
0xE000 0000 0xE00F FFFF	PPB	Strongly-Ordered	XN	-	1MB region reserved as the PPB. This supports key resources, including the System Control Space and debug features.

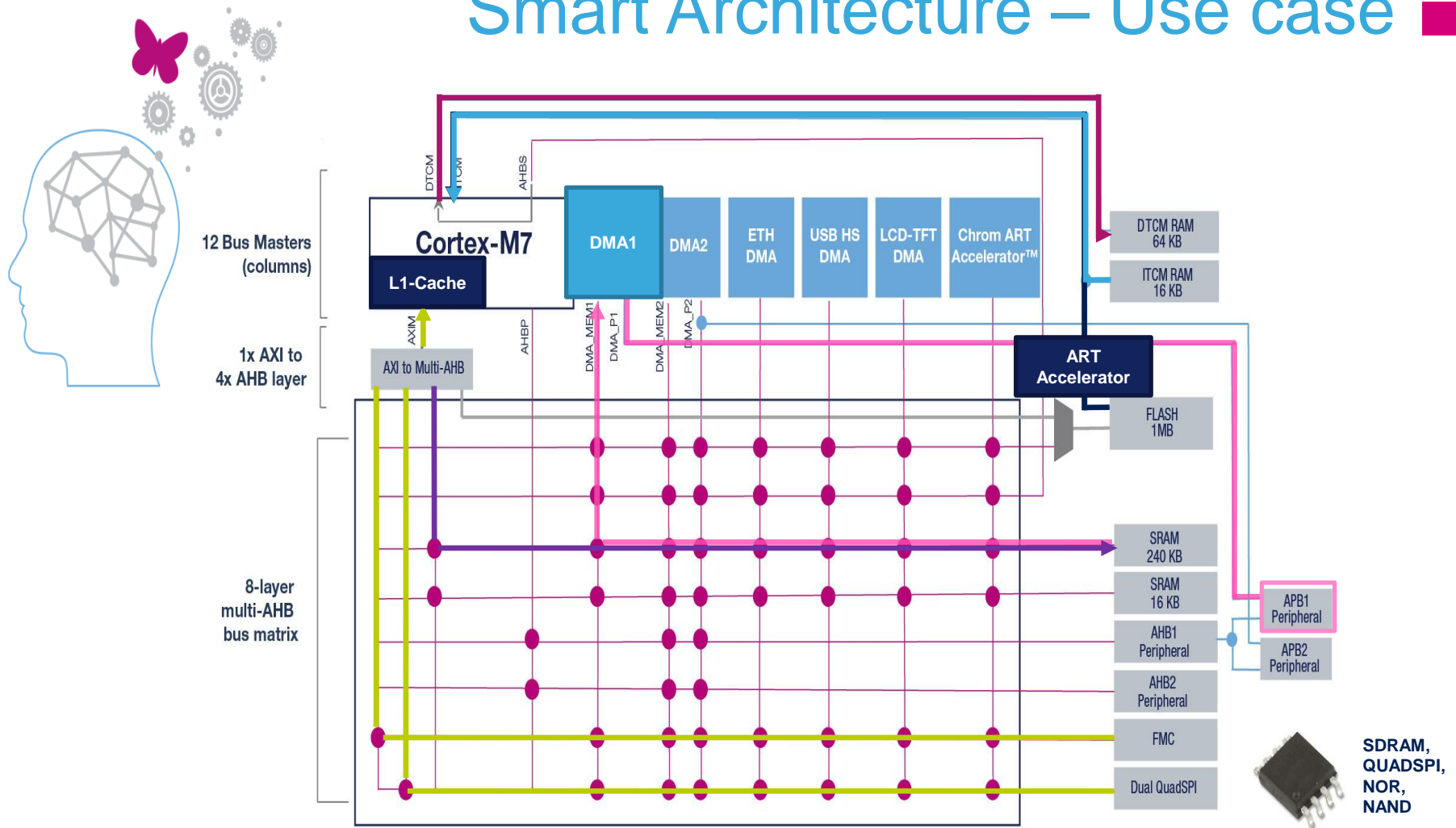


STM32F7x6 Smart Architecture

38



32-bit Bus Matrix



ART Accelerator: 0-wait state execution from Flash

Legend: ITCM: Critical Code with deterministic execution (ISRs, frequently used routines, main loop)

L1 cache for external memories: data manipulation or code execution

DTCM RAM: Critical real time data (Stack, heap ..)

System SRAM: Concurrent data transfer CPU or DMA

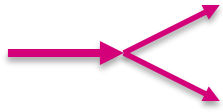
Internal memory mapping

Reserved	0x2005 0000
SRAM2(16 KB)	0x2004 C000
SRAM1(240 KB)	0x2001 0000
DTCM (64 KB)	0x2000 0000
Reserved	0x1FFF 0020
Option Bytes	0x1FFF 0000
Reserved	0x1FF0 EDC0
System memory on AXIM	0x1FF0 0000
Reserved	0x0820 0000
Flash memory on AXIM interface	0x0800 0000
Reserved	0x0030 0000
Flash memory on ITCM interface	0x0020 0000
Reserved	0x0011 0000
System memory on ITCM	0x0010 0000
Reserved	0x0000 4000
ITCM RAM	0x0000 0000

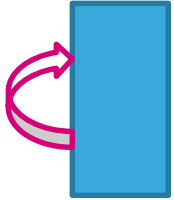
External memory mapping

SDRAM Bank2	0xD000 0000
SDRAM Bank1	0xC000 0000
QSPI registers	0xA000 1000
FMC registers	0xA000 0000
QSPI	0x9000 0000
Reserved	0x8000 0000
NOR/RAM 256MB	0x6000 0000

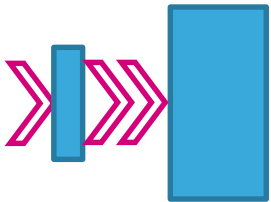
You can find on the leaflet



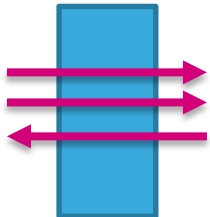
1. Superscalar → 2 instructions at 1 cycle



2. Branch in 1 cycle



3. Cache system to compensate slow memories



4. High system bandwidth for every application

Need more info ?

42

For more info contact:

enrico.marinoni@avnet.eu

(Digital FAE for STM - MCU, WireLess (IoT), MEMS, PLM, etc)

roberto.rossetti@avnet.eu

(B.D.M.)





Thank you

www.st.com/stm32