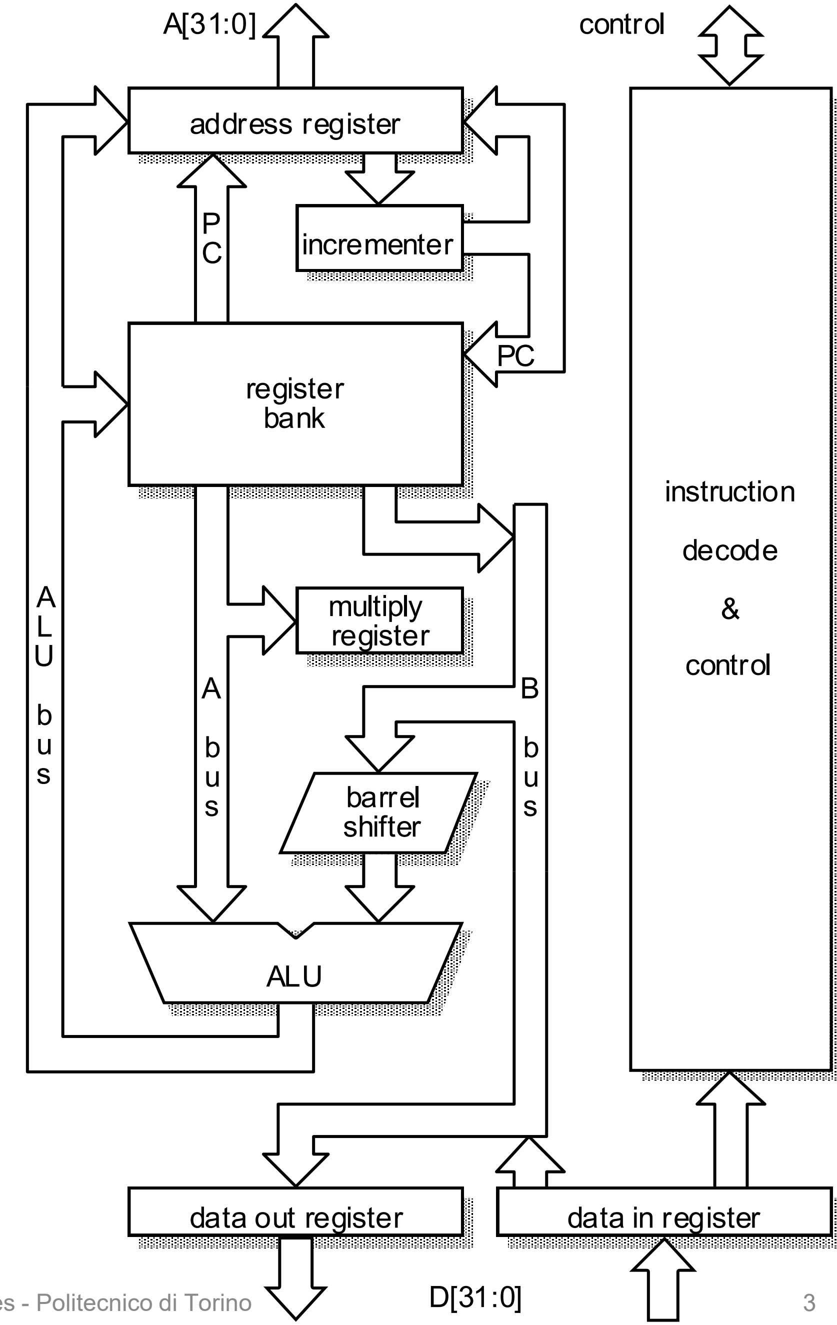
ARM v7-M Architecture

Renato Ferrero, Paolo Bernardi,
Matteo Sonza Reorda, Ernesto Sanchez
Politecnico di Torino
Dipartimento di Automatica e Informatica (DAUIN)
Torino - Italy

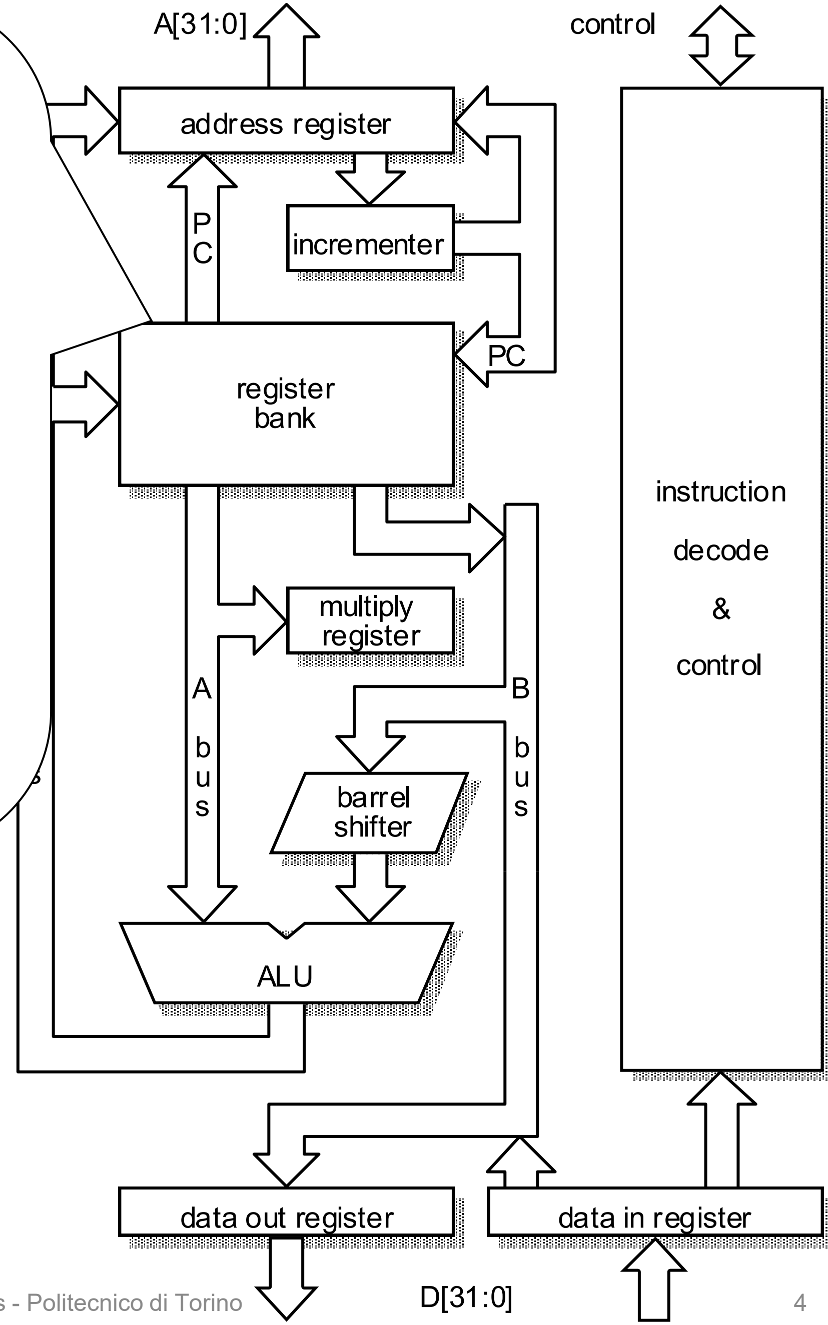
What's Happening in Microcontrollers?

- Microcontrollers are getting **cheap**
  - 32-bit ARM Cortex-M3 Microcontrollers@ $1
  - Some microcontrollers sell for as little as $0.65
- Microcontrollers are getting **powerful**
  - Lots of processing, memory, I/O in one package
  - Floating-point is even available in some!
- Microcontrollers are getting **interactive**
  - Internet connectivity, new sensors and actuators
  - LCD and display controllers are common

ARM generic Architecture

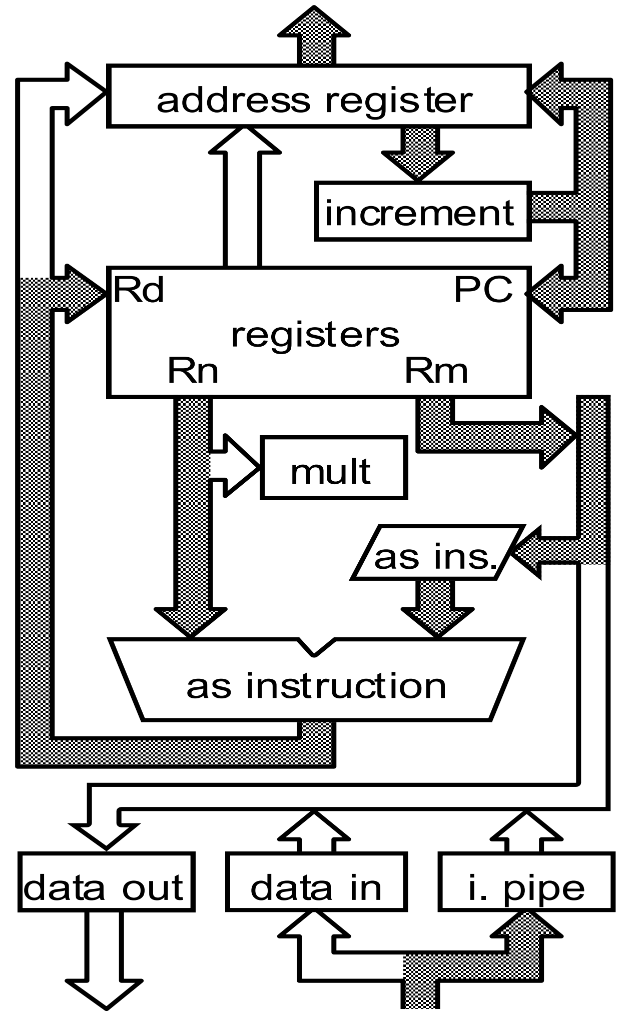


It has two read ports and one write port.
One additional read port and one additional write port are reserved for r15.



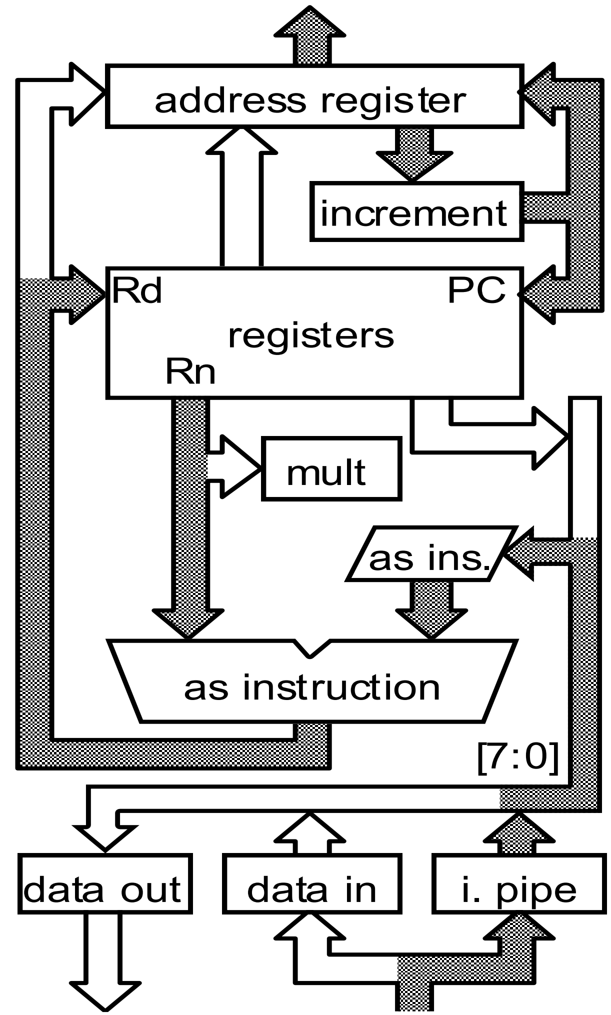
Data processing reg-reg instruction execution

- Instruction i is executed:
  - Two operands are read from registers *Rn* and *Rm*
  - One operand is possibly rotated
  - The ALU generates the result
  - The result is written to register *Rd*
  - A further instruction is fetched from memory
  - The PC is updated



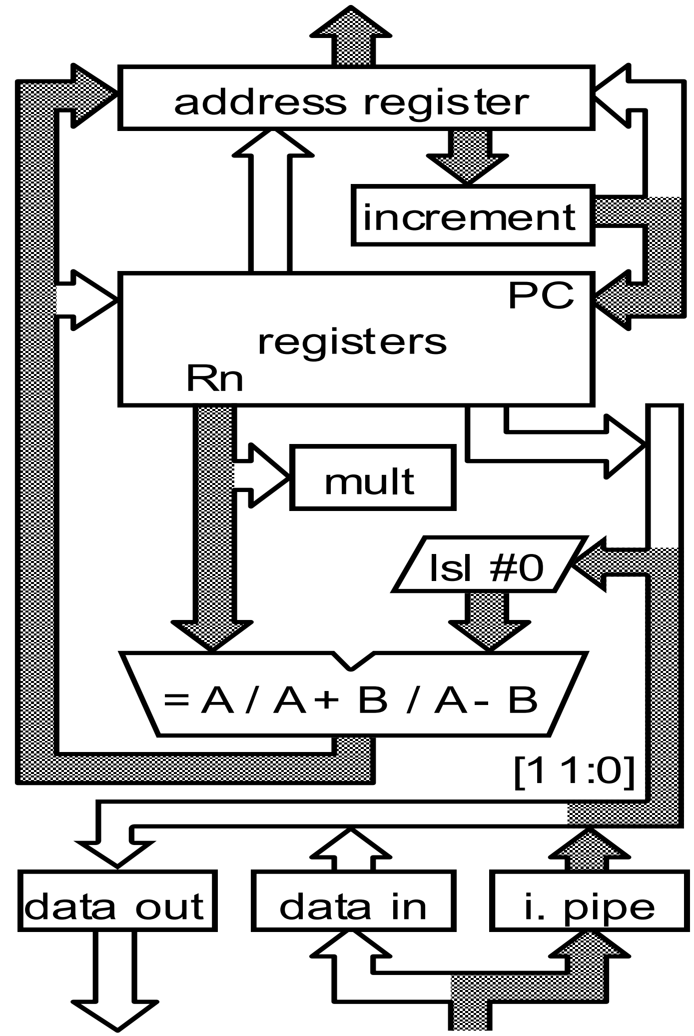
Data processing reg-imm instruction execution

- Instruction i is executed:
  - One operand is read from register *Rn*, the other is an immediate
  - One operand is possibly rotated
  - The ALU generates the result
  - The result is written to register *Rd*
  - A further instruction is fetched from memory
  - The PC is updated



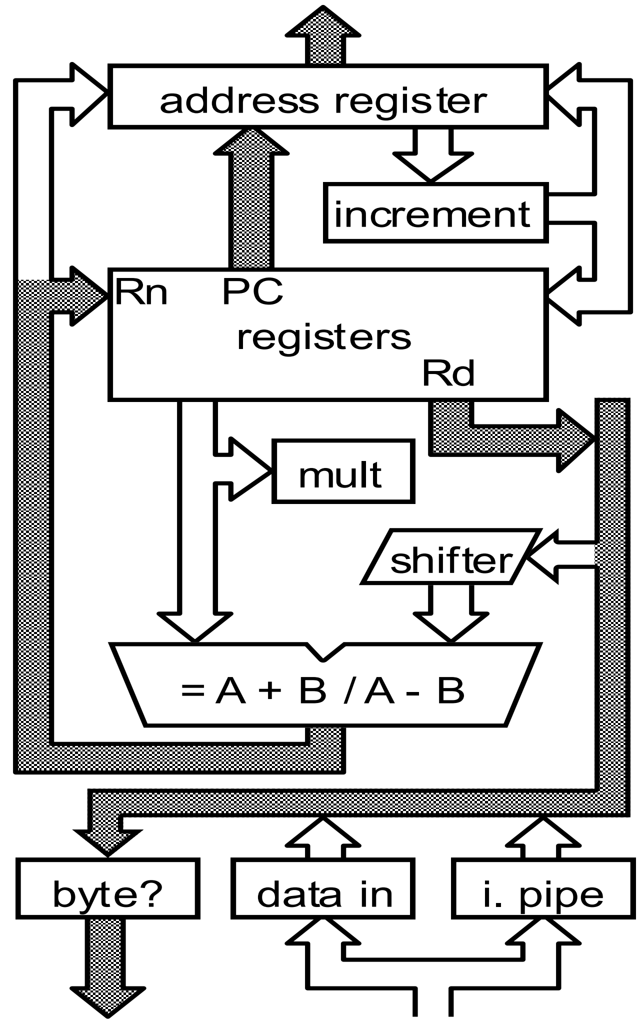
Data transfer instructions

- They require two clock cycles for the Execute stage
- In the first, the address is computed using one register and one immediate



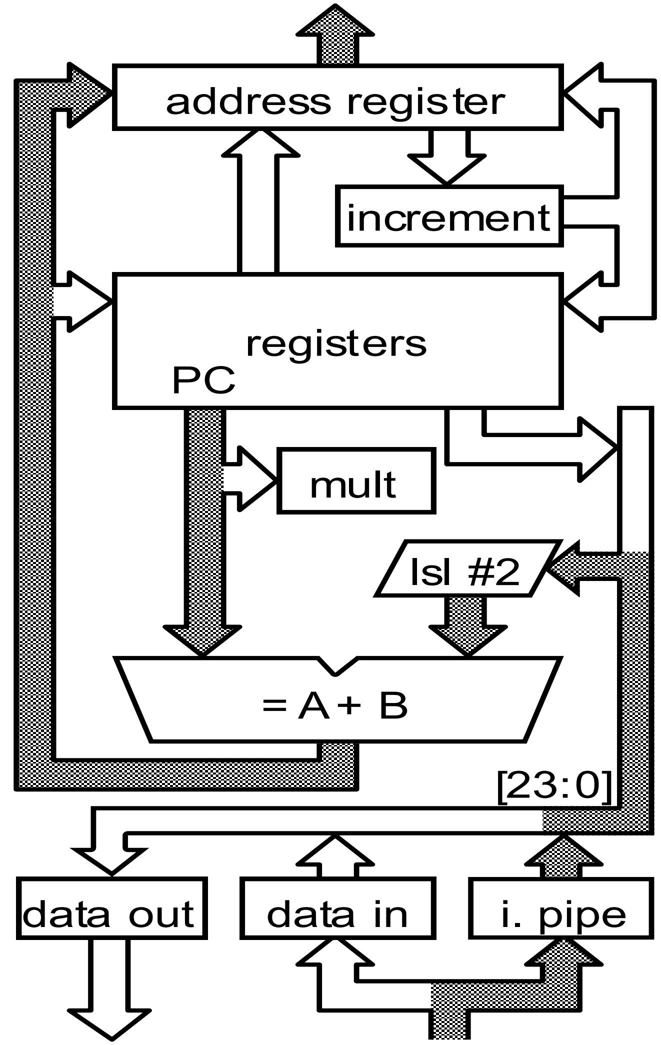
Data transfer instructions

- In the second clock cycle:
  - The memory is accessed
  - The source register is sent to the memory (STR instruction)



Branch instructions

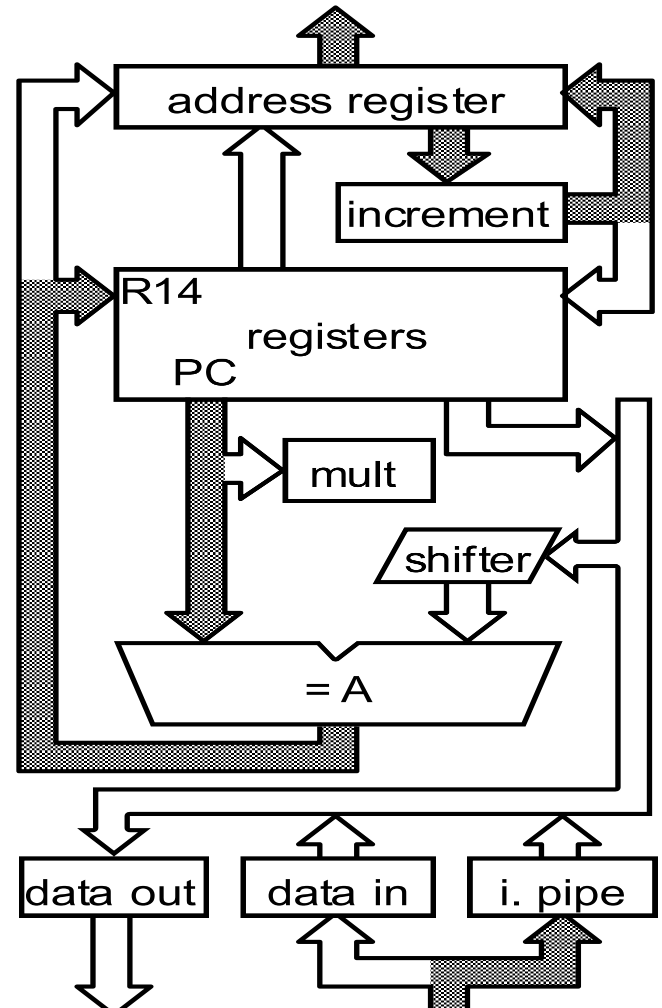
- These first compute the target address, adding an immediate (shifted by 2 positions) to the PC
- Then, the pipeline is flushed and refilled



(a) 1st cycle - compute branch target

Branch and link instructions

- In this case, a further clock cycle is required (while the pipeline is refilled) to save the return address in r14



(b) 2nd cycle - save return address

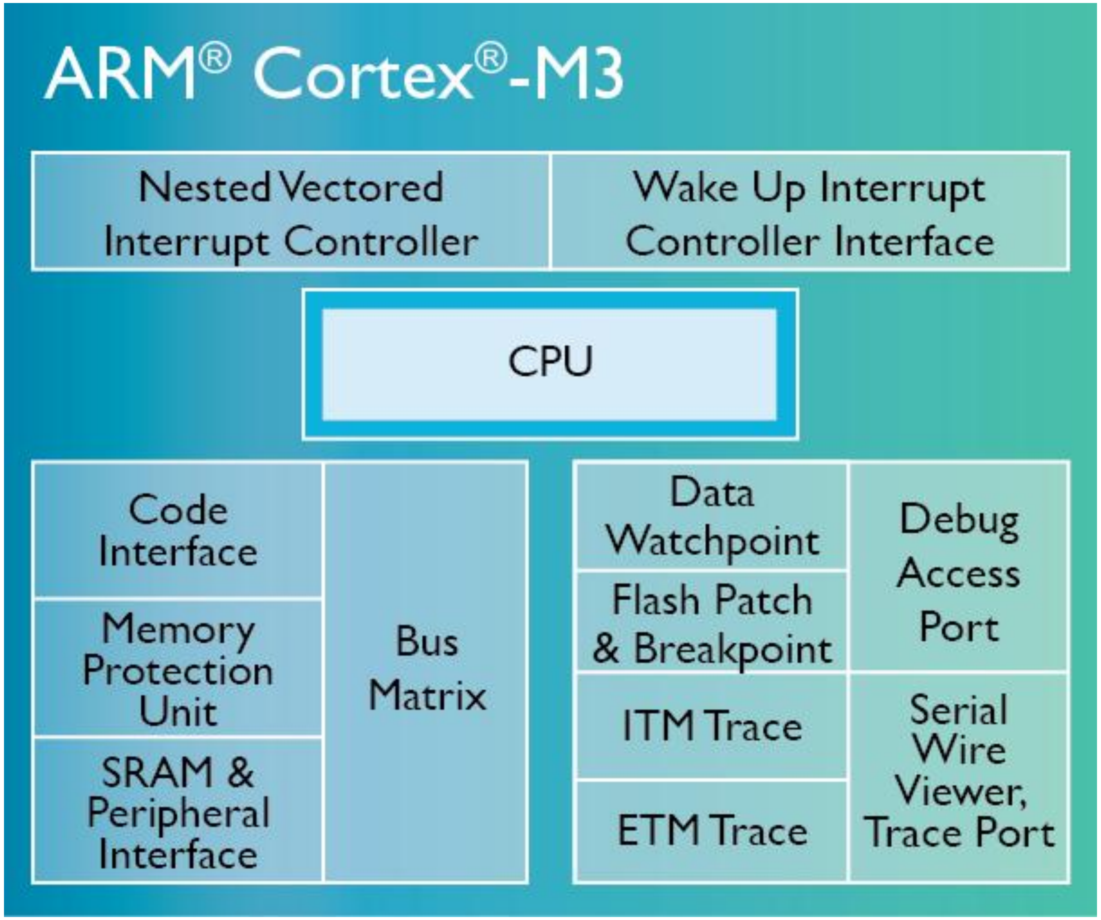
ARM Cortex–M3

Case of study for **Computer Architectures**

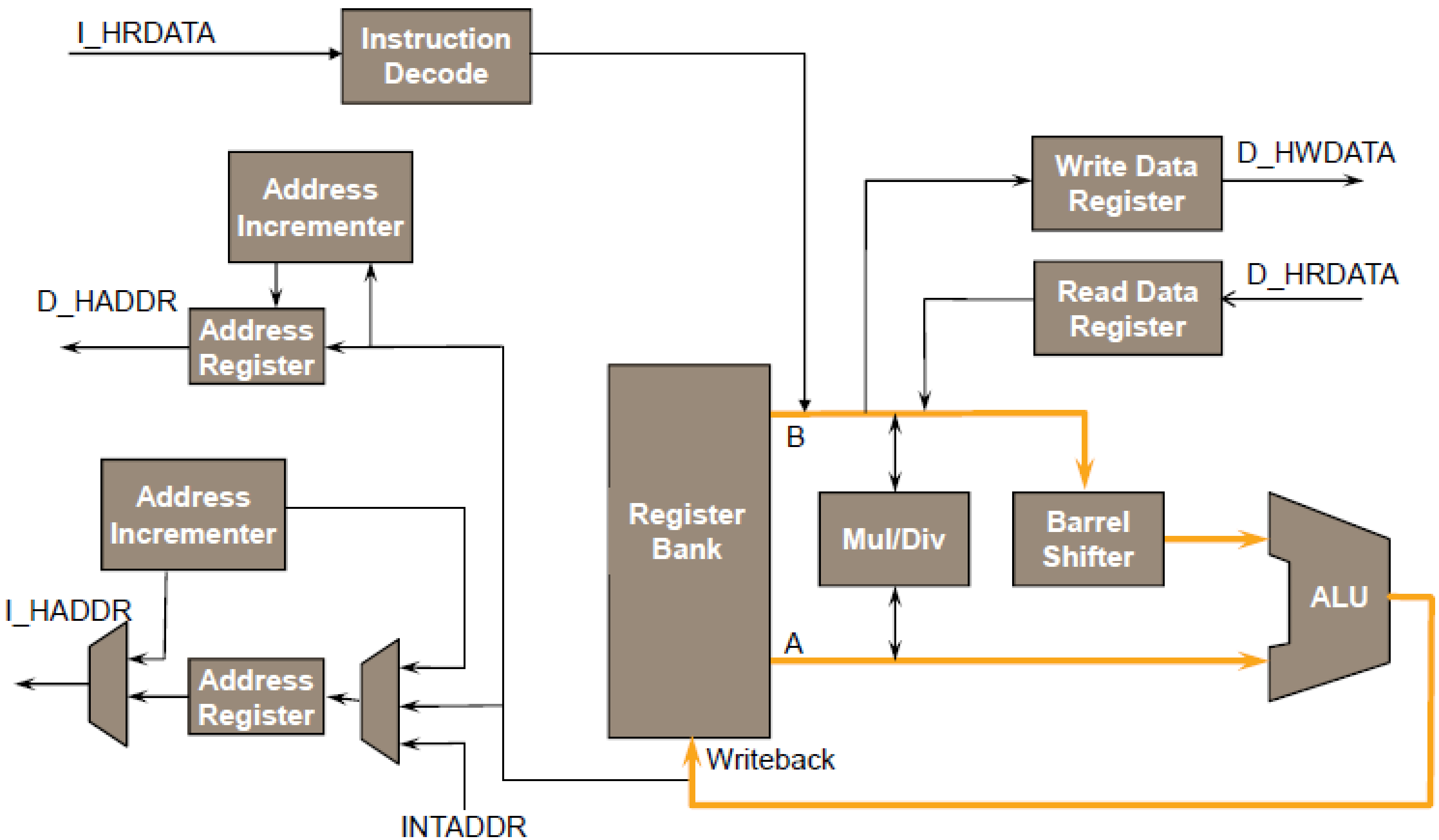
ARM family and architecture

				Cortex-A15			Application
				Cortex-A12			Embedded
				Cortex-A9			Classic
			ARM11MP	Cortex-A8			
Core			ARM176JZ	Cortex-A5		Cortex-M7	
		ARM926	ARM1176	Cortex-R7	SC000	SC300	
	SC100	ARM968	ARM1136	Cortex-R5	Cortex-M1	Cortex-M4	Cortex-A57
	ARM7TDMI	ARM946	ARM1156T2	Cortex-R4	Cortex-M0	Cortex-M3	Cortex-A53
Archi-tecture	v4	v5	v6	v7	v6-M	v7-M	v8
Instruc-tion set	**ARM 32 bit**						**ARM 64bit**
	Thumb 16 bit						
			Thumb-2			**Thumb-2**	

ARM Cortex–M3

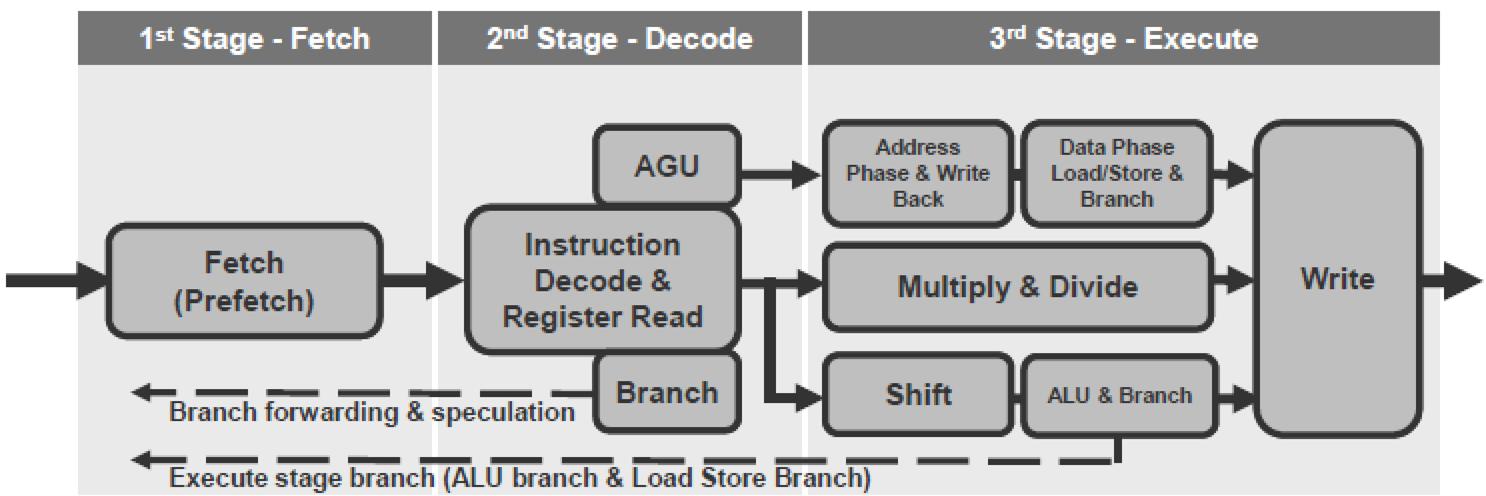


Cortex-M3 Datapath



Cortex-M3 Pipeline

- Cortex-M3 has 3-stage fetch-decode-execute pipeline



Branch Pipeline

- It takes 3 cycles to complete the branch
- Worst case scenario – indirect branch taken
  - They always flush and refill the pipeline
  - No delayed branch mechanism is supported

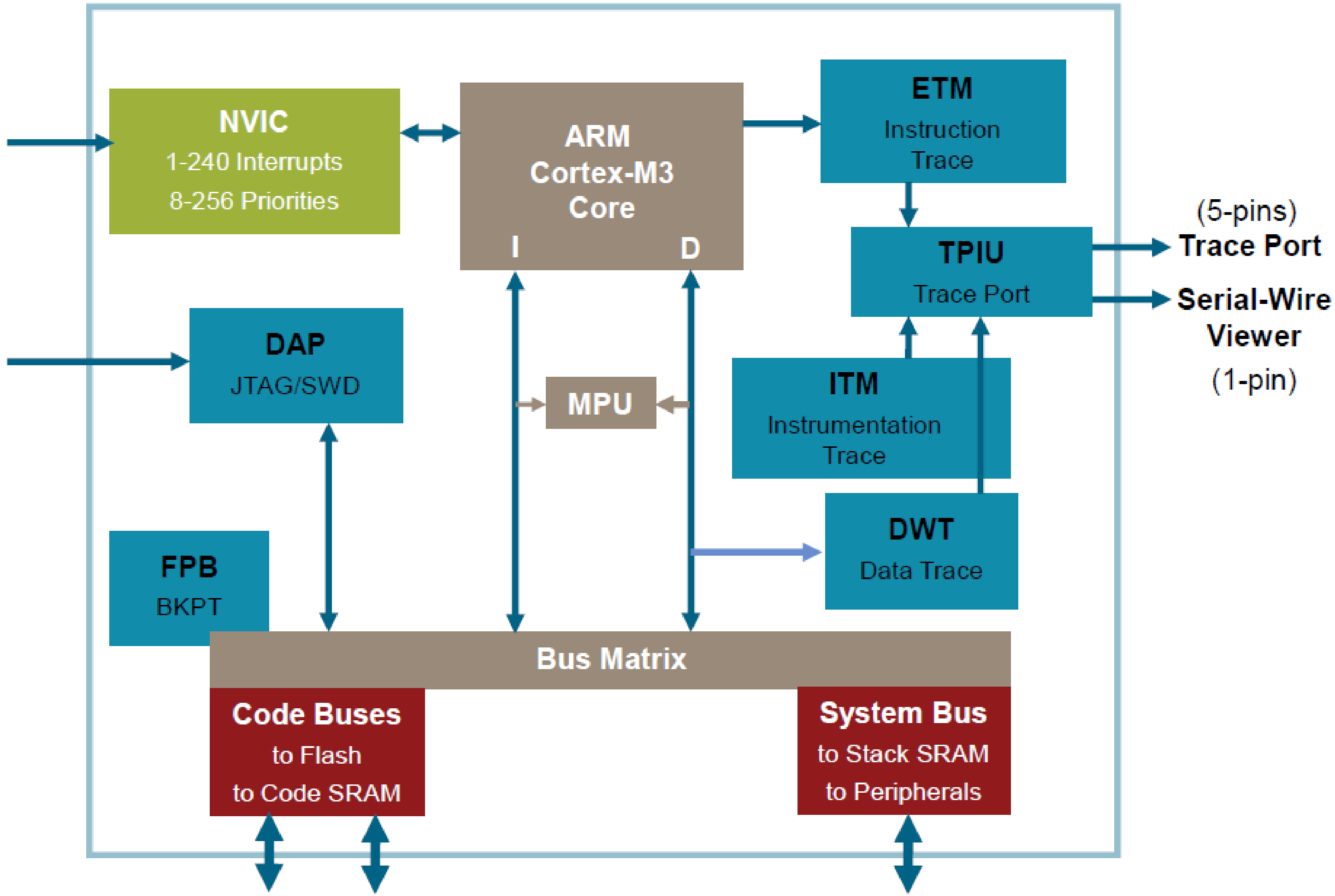
Cycle				1	2	3	4	5	6	7	8	9
Address	**Operation**											
0x8000	BX r5	F	D	E								
0x8002	SUB		F	D								
0x8004	ORR			F								
0x8FEC	AND				F	D	E					
0x8FEE	ORR					F	D	E				
0x8FF0	EOR						F	D	E			

LDR Pipeline

- The read cycle must complete on the bus before the LDR instruction can complete since there is only one write-back port in the register file

Cycle			1	2	3	4	5	6	7	8	9
Operation											
ADD	F	D	E								
SUB		F	D	E							
LDR			F	D	Ea	Ed					
AND				F	D	S	E				
ORR					F	S	D	E			
EOR							F	D	E		

ARM Cortex-M3 Processor block diagram with debug modules

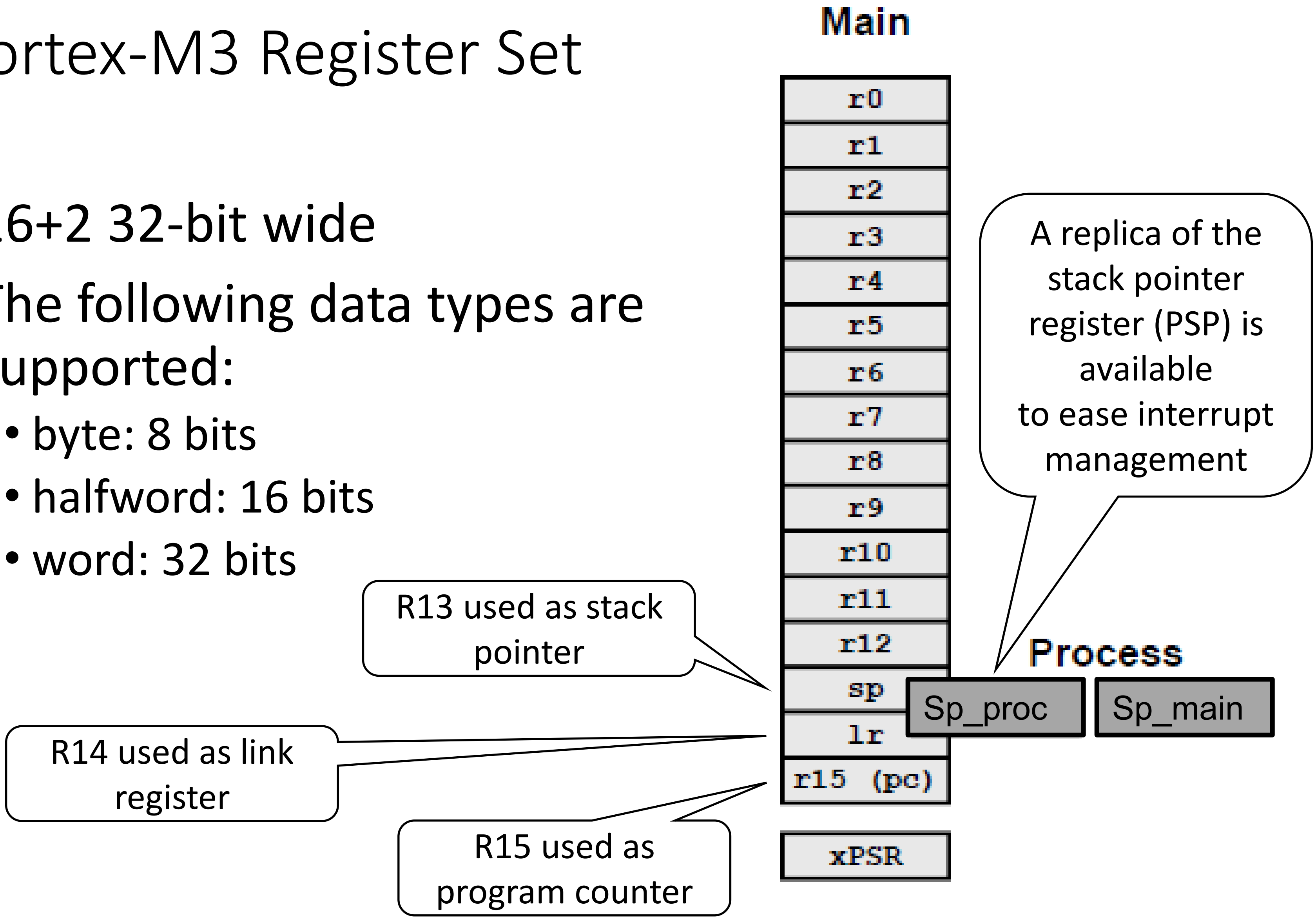


ARM Cortex-M3 Processor – programmer view

- 16+2 32-bit registers
- Efficient interrupt handling
- Power management enabling idle mode
- Efficient debug and development support features
  - Breakpoints - Watchpoints
  - Instruction Trace
- Strong OS support
  - User/Supervisor model
- Designed to be fully programmed in C, C++
  - even reset, interrupts and exceptions

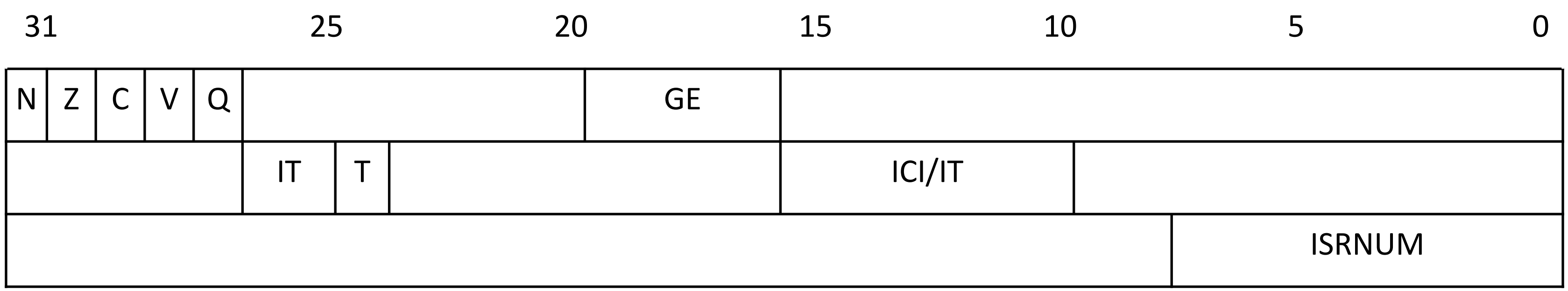
Cortex-M3 Register Set

- 16+2 32-bit wide
- The following data types are supported:
  - byte: 8 bits
  - halfword: 16 bits
  - word: 32 bits



PSR - Program Status Register

- It can be accessed all at once or as a combination of 3 registers:
  - Application Program Status Register (APSR)
  - Execution Program Status Register (EPSR)
  - Interrupt Program Status Register (IPSR)



Application Program Status Register

- It contains:
  - N: Negative result from ALU flag
  - Z: Zero result from ALU flag
  - C: ALU operation Carried out
  - V: ALU operation oVerflowed
  - GE: Greater Than or Equal flag
  - “sticky” Q flag

EPSR and IPSR

- The Execution Program Status Register contains:
  - IT: IF-THEN instruction status bits
  - ICI: Interrupt-Continuable Instruction bits
  - T: Thumb bit
- The Interrupt Program Status Register contains an exception number used in exception handling.

The T bit

- The mechanism to switch to/from Thumb instructions is driven by the T bit in the CPSR:
  - If T=1, the processor interprets the fetched code as a sequence of Thumb instructions
  - If T=0, the processor interprets the fetched code as a sequence of usual ARM instructions.
- The value of T can be changed via software.

The Thumb Instruction Set

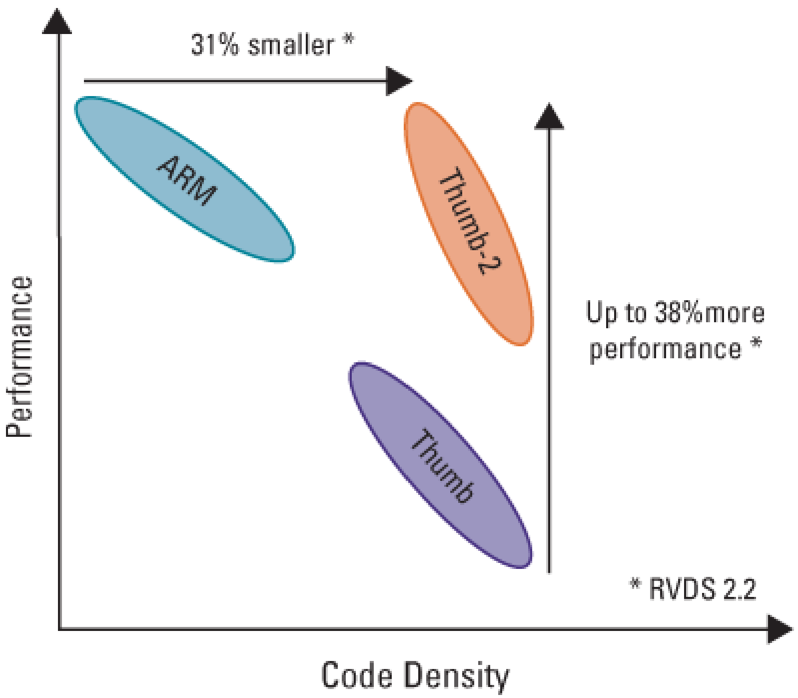
- Some of the ARM processors (those with a T in the acronym) support the Thumb instruction set (together with the standard ARM instruction set)
- In the Thumb instruction set
  - Instructions are encoded on 16 bits
  - Instructions are less powerful
  - Instructions are less.

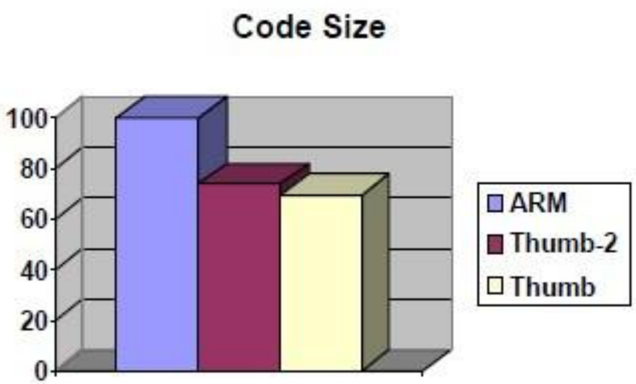
Thumb-2

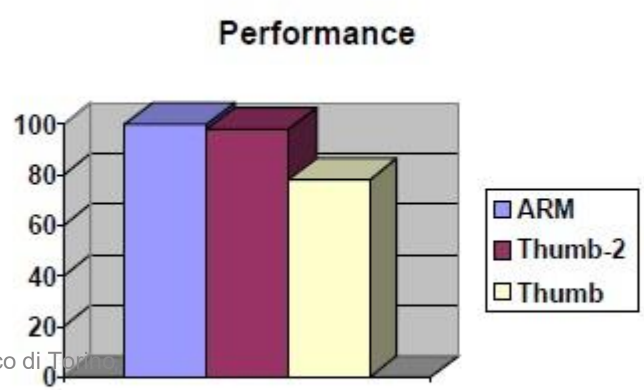
- Thumb-2 is a further instruction set, introduced by ARM in 2003
- Thumb-2 is supported by the latest ARM processor cores, which build on the ARM7 architecture
- Thumb-2
  - is a superset of Thumb (thus guaranteeing backward compatibility
  - includes new 16-bit instructions
  - includes some 32-bit instructions.

Thumb-2 vs. Thumb

- Thumb-2 is faster than Thumb, but still produces a very compact code







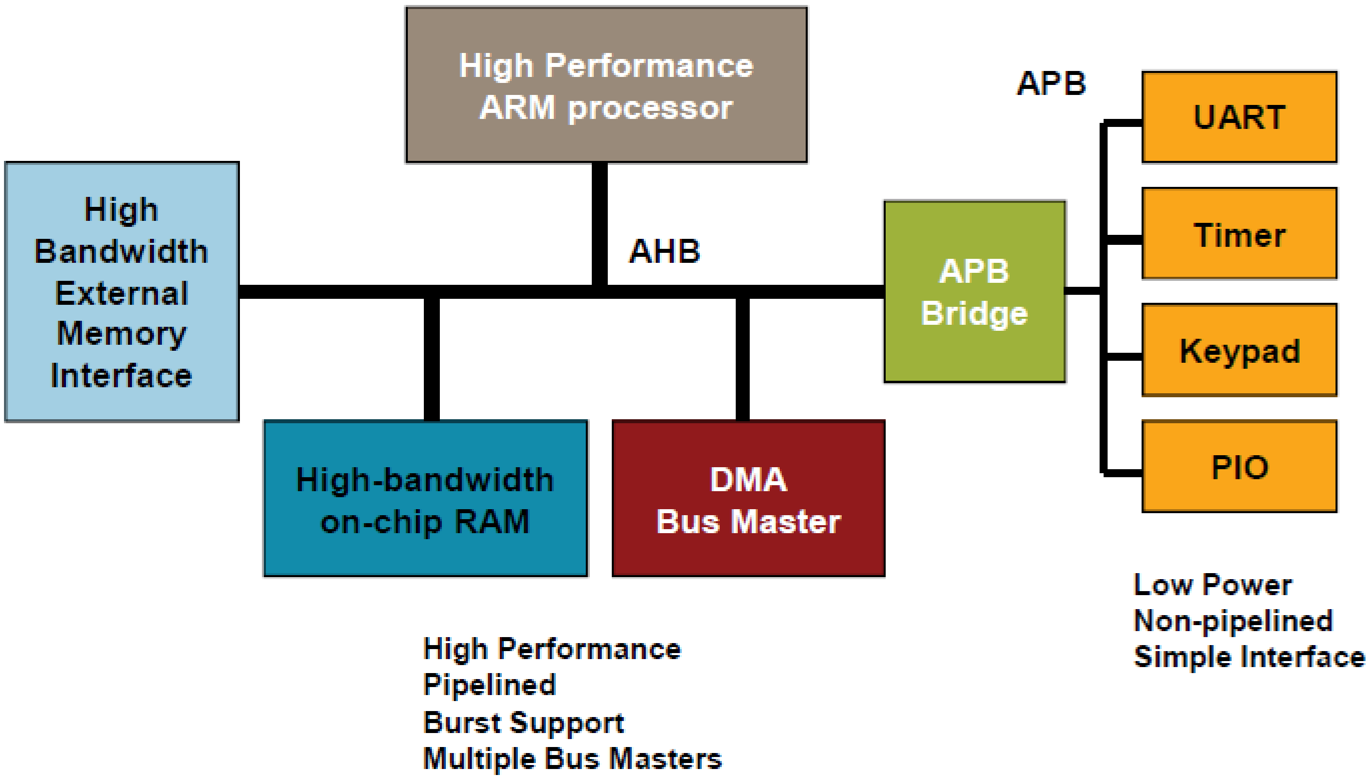
Processor operating modes and levels

- Two operating modes:
  - thread mode: on reset or after an exception
  - handler mode: when an exception occurs
- Two access levels:
  - user level: limited access to resources
  - privileged level: access to all resources
- Handler mode is always privileged.

AMBA Bus System

- The AMBA specification includes 3 busses:
  - The Advanced High-Performance Bus (AHB):
    - it is used to connect high-performance modules.
    - It supports burst mode data transfers and split transactions.
    - All timing is referenced to a single clock edge.
  - The *Advanced System Bus* (ASB):
    - it is an old specification, to be substituted by AHB *(kind of legacy type of bus you can even find in some systems based on old architectures)*
  - The *Advanced Peripheral Bus* (APB):
    - offers a simpler interface for low-performance peripherals.
    - APB is generally used as a local secondary bus which appears as a slave module on the AHB.

AMBA Bus System

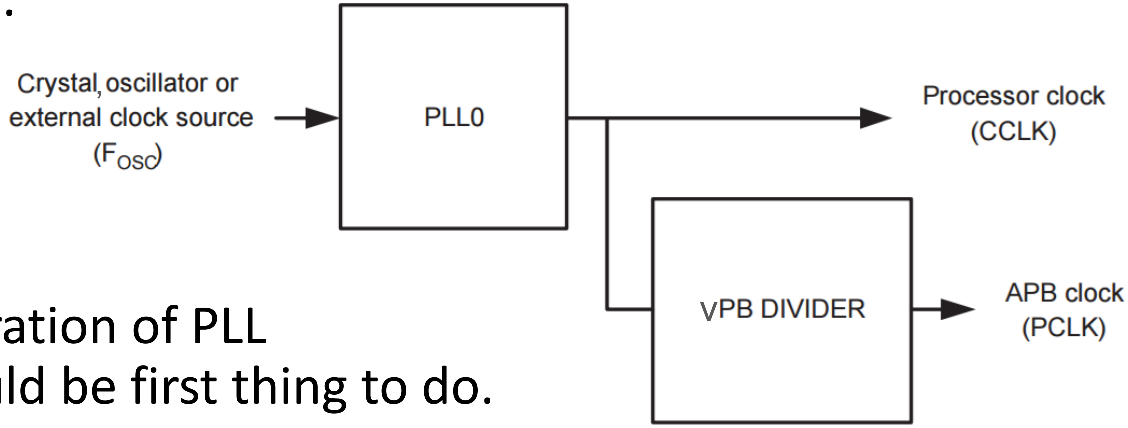


Clock distribution

- ARM systems like ARM v7-M then need two clocks
  - High frequency for CPU and high-speed system components
  - Low frequency for peripheral cores that requires less performance or must operate at limited speed (i.e., I/O communications)
- The CPU clock (CCLK) and peripheral clock (PCLK) gets clock input from a PLL (Phase Lock Loop), VPB (VLSI Peripheral Bus) Divider, or from external source.

Oscillator

Crystal



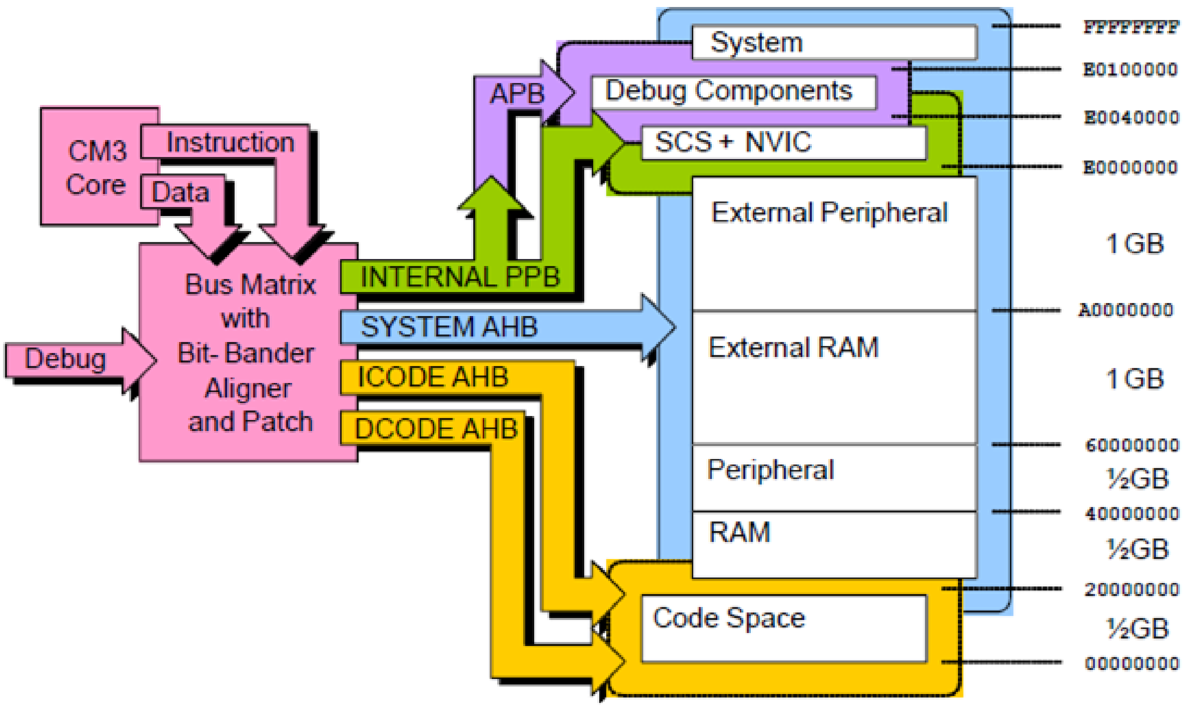
- After RESET, configuration of PLL and VPB Divider would be first thing to do.

Power Management capabilities

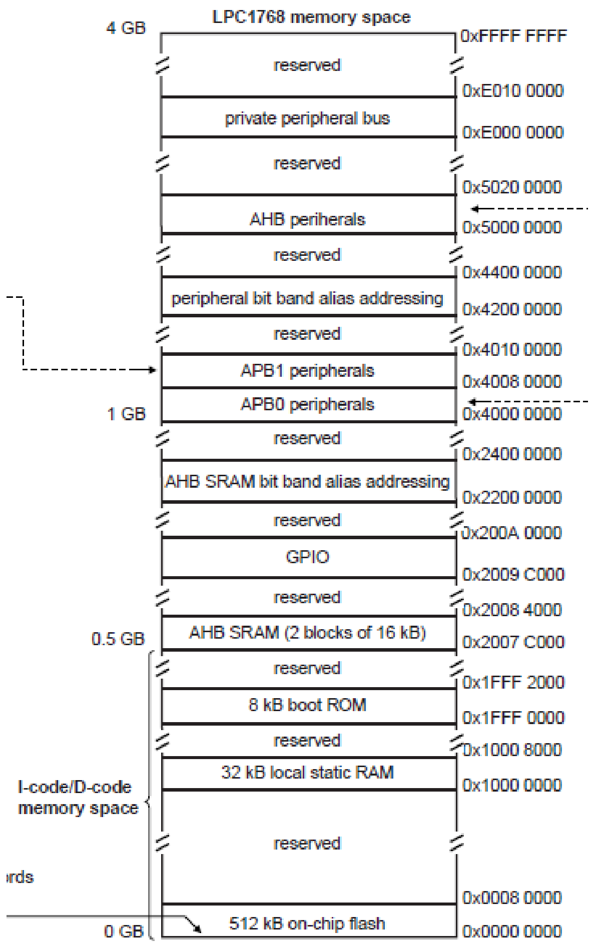
- Multiple sleep (idle) modes supported
  - Sleep Now – Wait for Interrupt/Event instructions
  - Sleep On Exit – Sleep immediately on return from last ISR
  - Deep Sleep
    - Long duration sleep, so PLL can be stopped
- Cortex-M3 system is clock gated in all sleep modes
  - Sleep signal is exported allowing external system to be clock gated also
  - NVIC interrupt Interface stays awake
- Wake-Up Interrupt Controller (WIC)
  - External wake-up detector allows Cortex-M3 to be fully powered down
  - Effective with State-Retention / Power Gating (SRPG) methodology

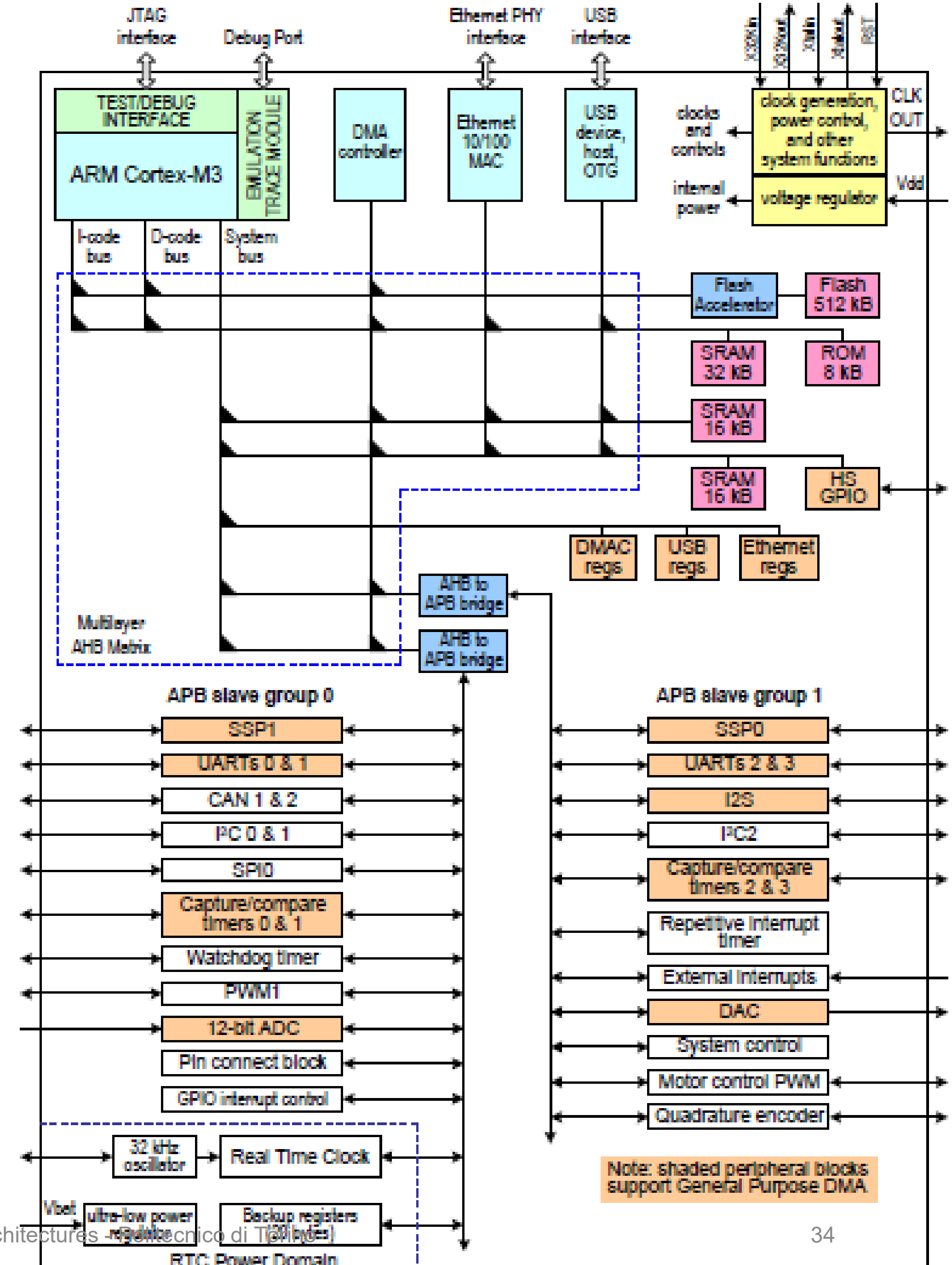
Memory Map organization

- Very simple linear 4GB memory map
- The Bus Matrix partitions memory access via the AHB and PPB buses



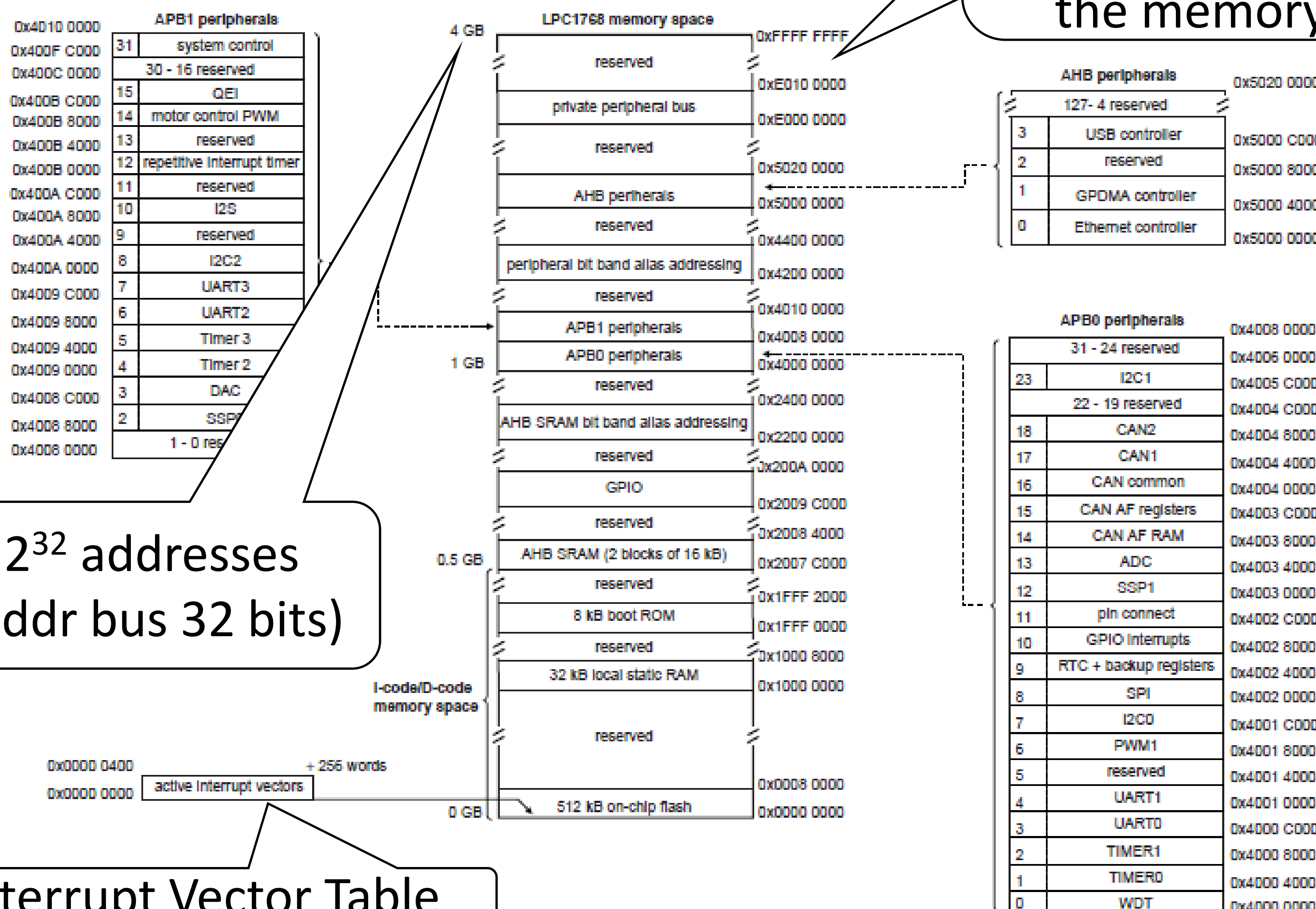
NXP LPC176x/5x block diagram and memory map





NXP LPC176x/5x memory map

Not all 4GB are used, there are some «holes» in the memory



2^{32} addresses (addr bus 32 bits)

Interrupt Vector Table

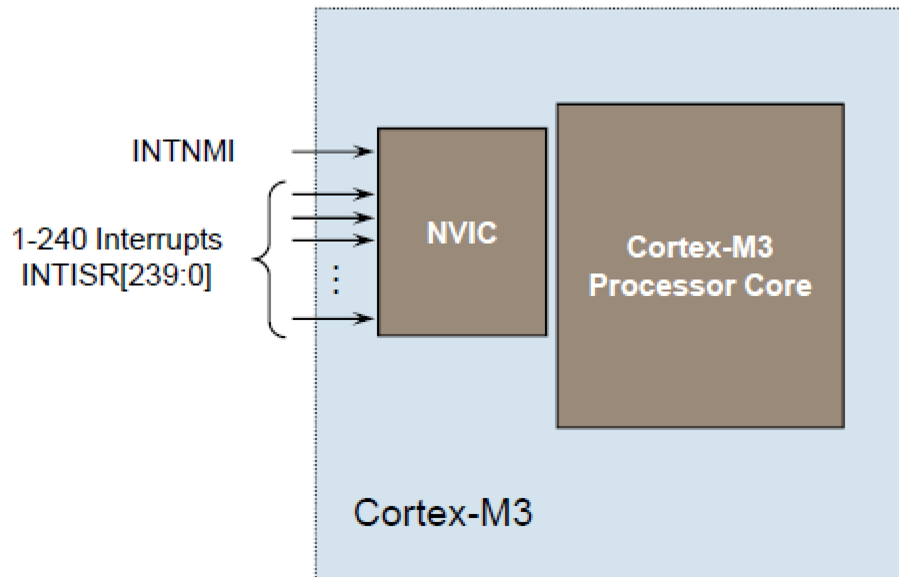
Exception Handling

- Reset
- NMI
- Faults
  - Hard Fault
  - Memory Manage
  - Bus Fault
  - Usage Fault
- SVCall
- Debug Monitor
- PendSV
- SysTick Interrupt
- External Interrupt

Interrupt is a term used in the ARM scenario to identify an exception caused by an external event

Interrupt Handling

- One Non-Maskable Interrupt (INTNMI) supported
- A Nested Vectored Interrupt Controller (NVIC) is tightly coupled with processor core
  - 1-240 prioritizable interrupts supported



Interrupt Vector table in v7-M architecture

- An "interrupt vector table" (IVT) is a data structure that associates a list of interrupt handlers with a list of interrupt requests in a table of interrupt vectors.
- Each entry manages an exception, interrupt, or other atypical event such as a reset.
- There are 2 possibilities:
  - The table content is composed of branch instructions to the specific handler
  - The table stores the addresses of the handler, which is loaded in the PC as soon as the exception arises.

Interrupt Vector table in v7-M architecture (II)

Exception Type	Index	Vector Address
(Top of Stack)	0	0x00000000
Reset	1	0x00000004
NMI	2	0x00000008
Hard fault	3	0x0000000C
Memory management fault	4	0x00000010
Bus fault	5	0x00000014
Usage fault	6	0x00000018
SVcall	11	0x0000002C
Debug monitor	12	0x00000030
PendSV	14	0x00000038
SysTick	15	0x0000003C
Interrupts	≥16	≥0x00000040

Each line contains an address to be copied in the PC in case a specific exception occurs.

The access mechanism to the table is hardware-based and «transparent» to the programmer

Anyway, it is a programmer duty to setup the IVT at boot time.

Features of ARM Instruction Sets

- Instructions are 32 (or 16) bits long.
- Every instruction can be conditionally executed.
- A load/store architecture
  - Data processing instructions act only on registers
  - Three operand format
  - Combined ALU and shifter
  - Memory access instructions with auto-indexing