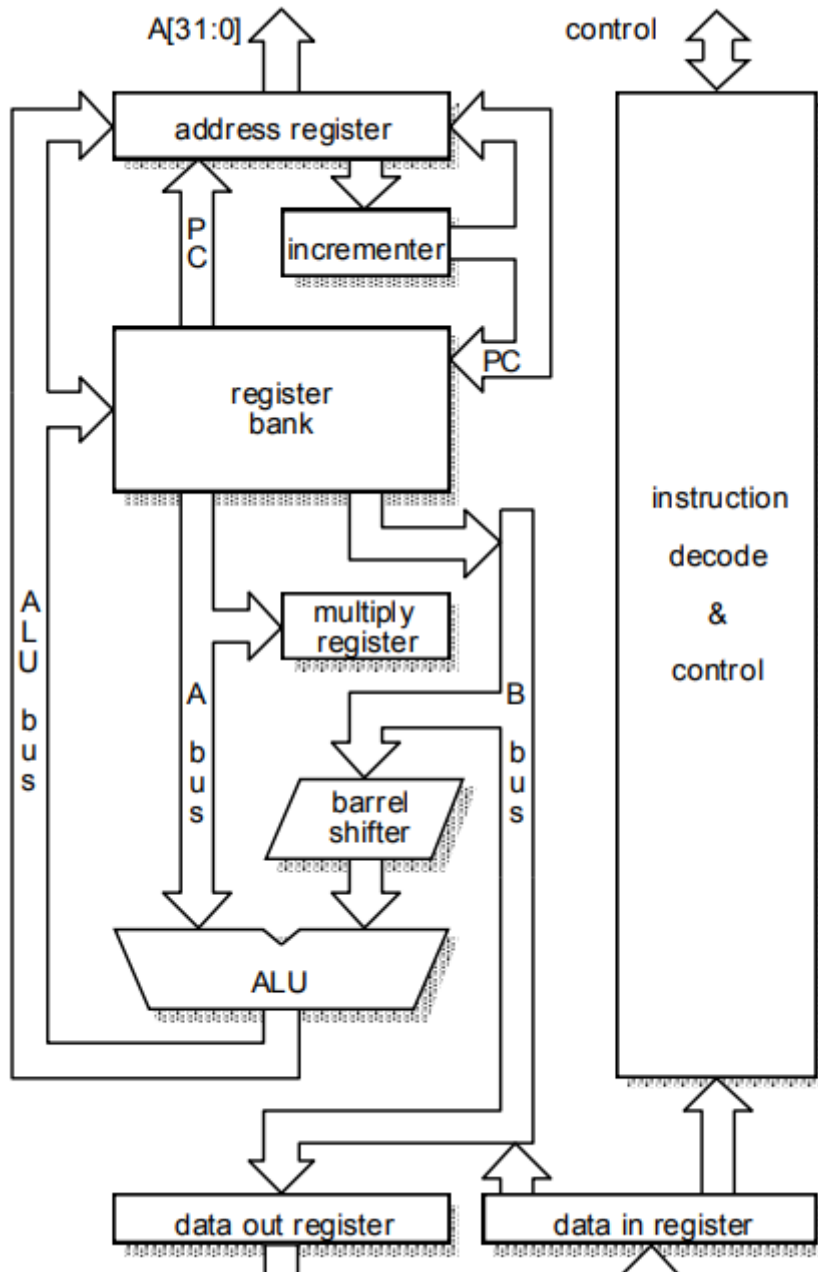


Due tipi di core per i SoC:

- **soft core:** descrizione ad alto livello attraverso linguaggi di sintesi(VHDL etc)
- **hard core:** descrizione del layout fisico

Architettura ARM v7-M

Architettura **generica** ARM:



Il *barrel shifter* permette di generare numeri *immediate* più grandi.
Notare come il register bank abbia due *writing port*

Data processing(ALU) instructions

Register-Register

1. Due operandi vengono letti dal register bank(Rn e Rm)
2. Un operando viene potenzialmente ruotato/shiftato
3. L'ALU genera il risultato
4. Il risultato viene salvato in Rd
5. Viene fetchata un'altra istruzione
6. Il PC viene aggiornato

Register-Immediate

Come Register-Register ma al passo 1 il secondo operando è un immediate

Data transfer instructions

Load and store operations

They require two clock cycles for the **Execute** stage.

In the first stage, the address is computed using one register and one immediate(offset).

In the second clock cycle:

- **memory** is accessed
- **source register** is sent to the memory(in case of a **store**)

Branch Instructions.

Prima viene calcolato il *target* address aggiungendo un'immediate(**lsI 2** di posizioni) al PC, infine la **pipeline** viene *flushata* e *refillata*

Branch and link

Un ulteriore ciclo di clock viene usato per salvare l'indirizzo di ritorno in **R14**(link register).

ARM Cortex-M3

Il Cortex-M3 usa l'architettura **v7-m**, un miscuglio tra **Thumb**(16-bit) e **Thumb-2**(16/32-bit)

Pipeline

Ha 3 stadi:

- **Fetch**

- **Decode:**
Instruction decode e lettura dei registri.
Calcolo del *target address* per il branch.
Speculazione e branch forwarding.
- **Execute:** Operazioni dell'ALU, load/store, writeback

Branch Pipeline

Impiega 3 cicli per completarlo e nel caso di un branch non preso, viene **sempre** flushata e refillata la pipeline

Programmer View

Panoramica

- Il **Cortex-M3** ha 18 registri da 32 bit.
- Gestione efficiente dell'handling degli interrupt.
- Efficiente sistema di debug e sviluppo (breakpoints etc)
- Forte supporto per l'esecuzione di sistemi operativi (modalità **user** e modalità **supervisor**).
- Pensato per essere **completamente** programmato in **C**.

Registri

Come detto sopra, sono a 32 bit.

Supportano i seguenti tipi:

- 8 bit (byte)
- 16 bit (halfword)
- 32 bit (word)

I registri da **R0** a **R12** sono completamente general purpose,

- **R13** è lo stack pointer (**sp**)
- **R14** è il link register (**lr**)
- **R15** è il program counter (**pc**)

questi 3 registri possono comunque essere usati in modo generale, tuttavia bisogna prestare attenzione.

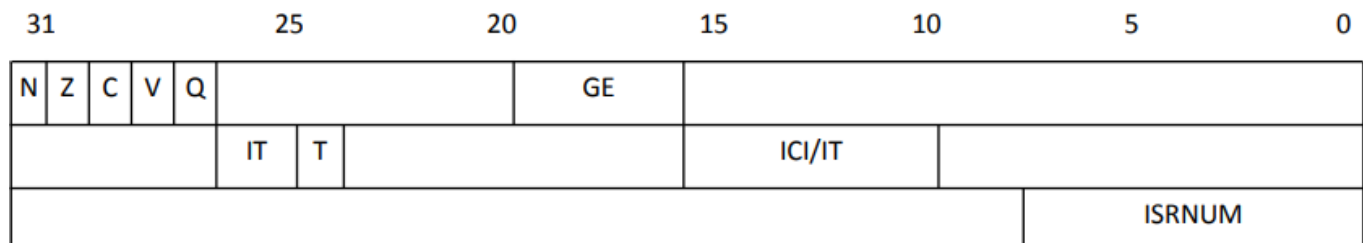
Il **R13** viene replicato per avere una migliore gestione degli interrupt e per un migliore supporto del programmatore

Lo **SP** principale viene detto **Master SP**, la replica **PSP**.

C'è anche un ulteriore registro, **SPR**, che contiene vari flag.

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)



Il flag Q(sticky) ci dà visibilità su qualche elemento architetturale

APSR

Contiene:

- **N**: ultima operazione ha dato risultato negativo
- **Z**: ultima operazione ha dato risultato uguale a 0
- **C**: ultima operazione ha dato un carry
- **V**: ultima operazione ha generato overflow
- **Q**: detto anche sticky.

EPSR

Contiene:

- **IT**: IF-THEN bits
- **T**: per passare da Thumb mode o Arm mode

IPSR

Contiene un *exception number* usato nella gestione delle eccezioni

Supporto ai sistemi operativi

Ci sono due modi operativi per la CPU:

- **Thread mode:** modalità d'esecuzione *normale*.
Di default o dopo un'eccezione
- **Handler mode:** quando un eccezione accade

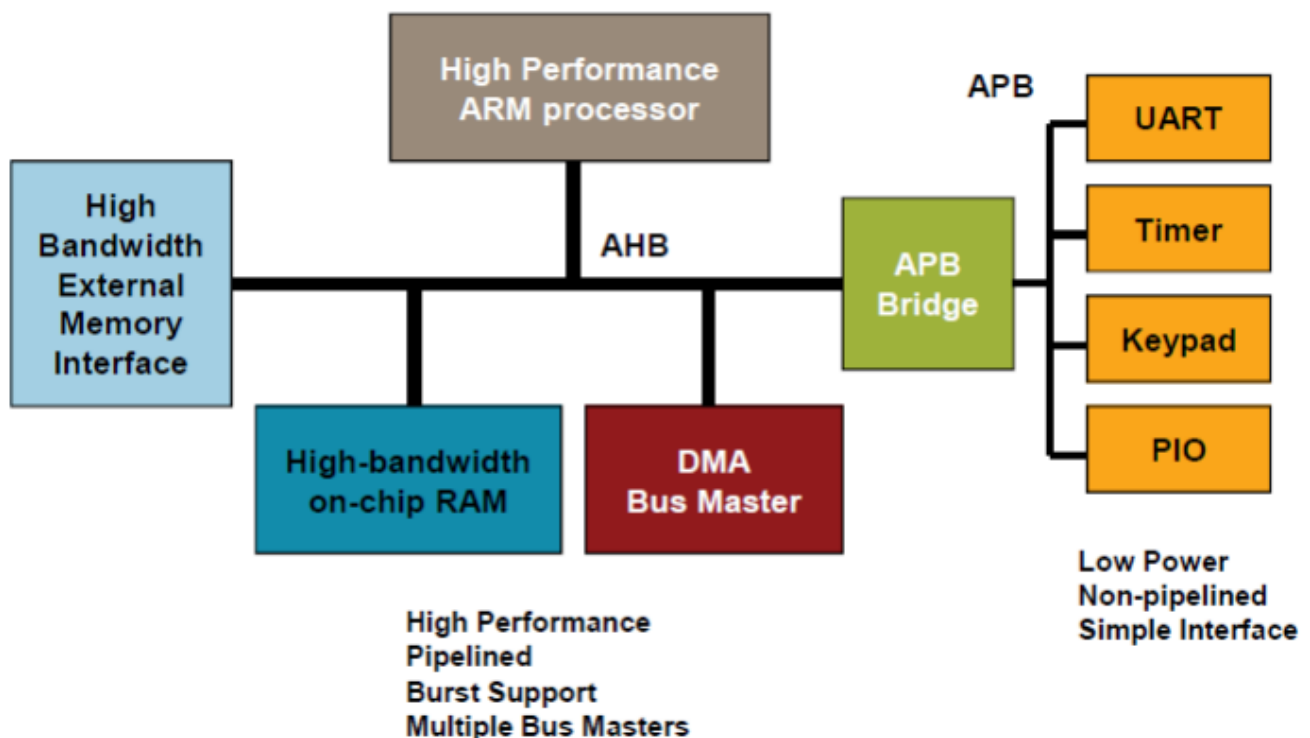
e due livelli d'accesso:

- **user level:** accesso limitato alle risorse
- **privileged level:** accesso a tutte le risorse.
La modalità **handler** è sempre **privileged**

Sistema di bus AMBA

Il bus **AHB - Advanced High-performance Bus** viene usato per la comunicazione con componenti ad alte prestazioni.(o comunicazione interna); lavora sui circa 100MHz

Il bus **APB - Advanced Peripheral Bus** viene usato per la comunicazione con le periferiche(o comunicazione esterna);lavora sui circa 25MHz



Modulo di power management

Modulo periferico che permette al sistema di andare in sleep mode.

Ci sono diverse sleep mode:

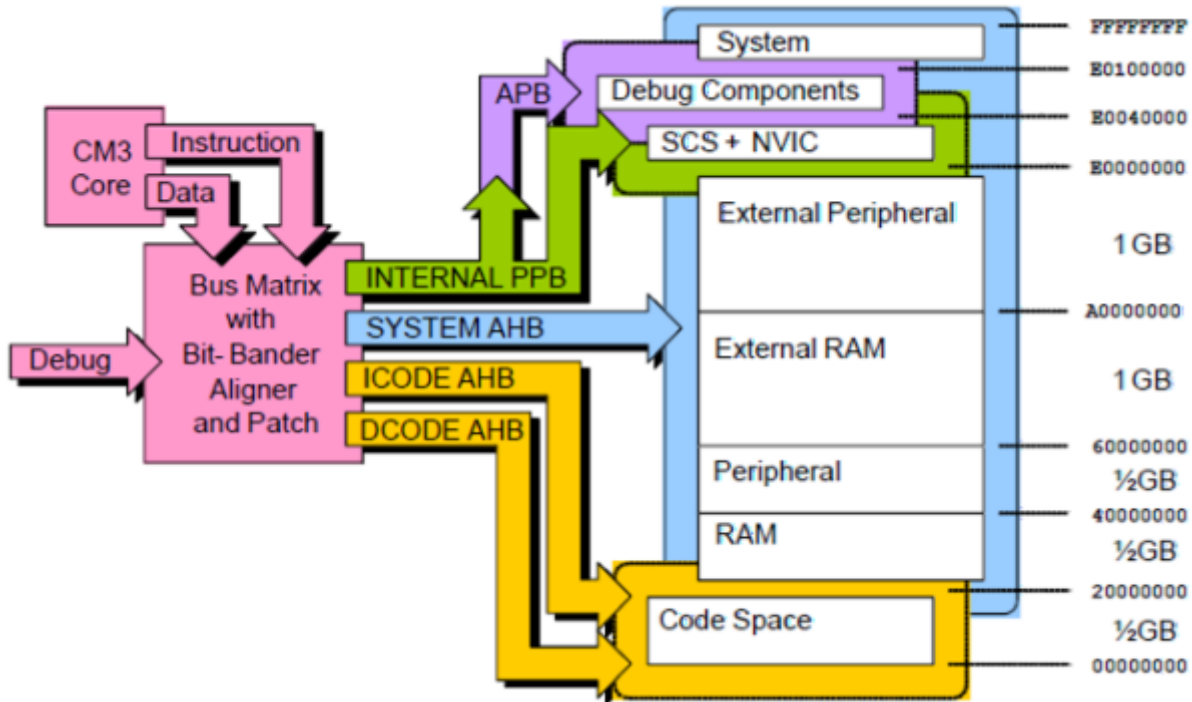
- **sleep now:** aspetta per interrupt/event instructions
- **deep sleep:** fa addormentare tutto il sistema(memorie,debugger etc).
Funziona attraverso la tecnica di **clock-gating**, ovvero sospendendo il clock.

Per svegliare il sistema si usa un WIC, Wakeup Interrupt Controller.

Memory map organization

Memory map lineare di **4GB**.

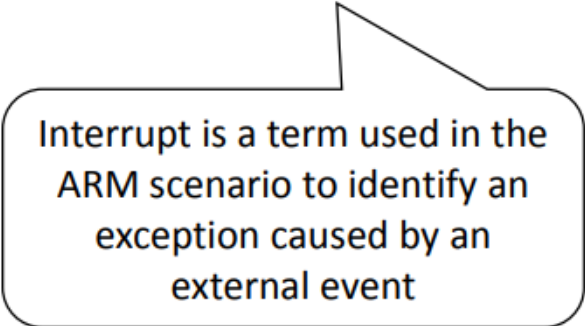
Non possiede cache, infatti i dati vengono messi nelle **SRAM**, mentre il codice nel **FLASH**.



Exception Handling

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

Reset è una eccezione speciale che si attiva all'avvio del sistema.

NMI sono interrupt non mascherabili(non disattivabili), ad esempio come quelle legate all'alimentazione(caduta di tensione etc).

Interrupt Handling

Gli interrupt sono eventi in arrivo dall'esterno.

Vengono gestiti tramite il **NVIC**, Nested Vectored Interrupt Controller, che è strettamente collegato al core processor.

Ci sono in totale 240 interrupt gestibili.

IVT - Interrupt Vector Table

Il **IVT** è un elemento cruciale nella gestione delle interrupt.

E' una struttura dati che associa una lista di interrupt con i suoi handler.

Due possibilità riguardo il suo contenuto:

- istruzioni di branch che puntano a degli handler

- indirizzo delle prime istruzioni degli handler che viene salvato **poi** nel PC.

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