Università degli Studi di Padova

Facoltà di Ingegneria

Corso di Laurea in Ingegneria Informatica

Tesina di Laurea Triennale

I PROCESSORI ARM CORTEX

Laureando: Toniolo Massimo

Relatore: Congiu Sergio

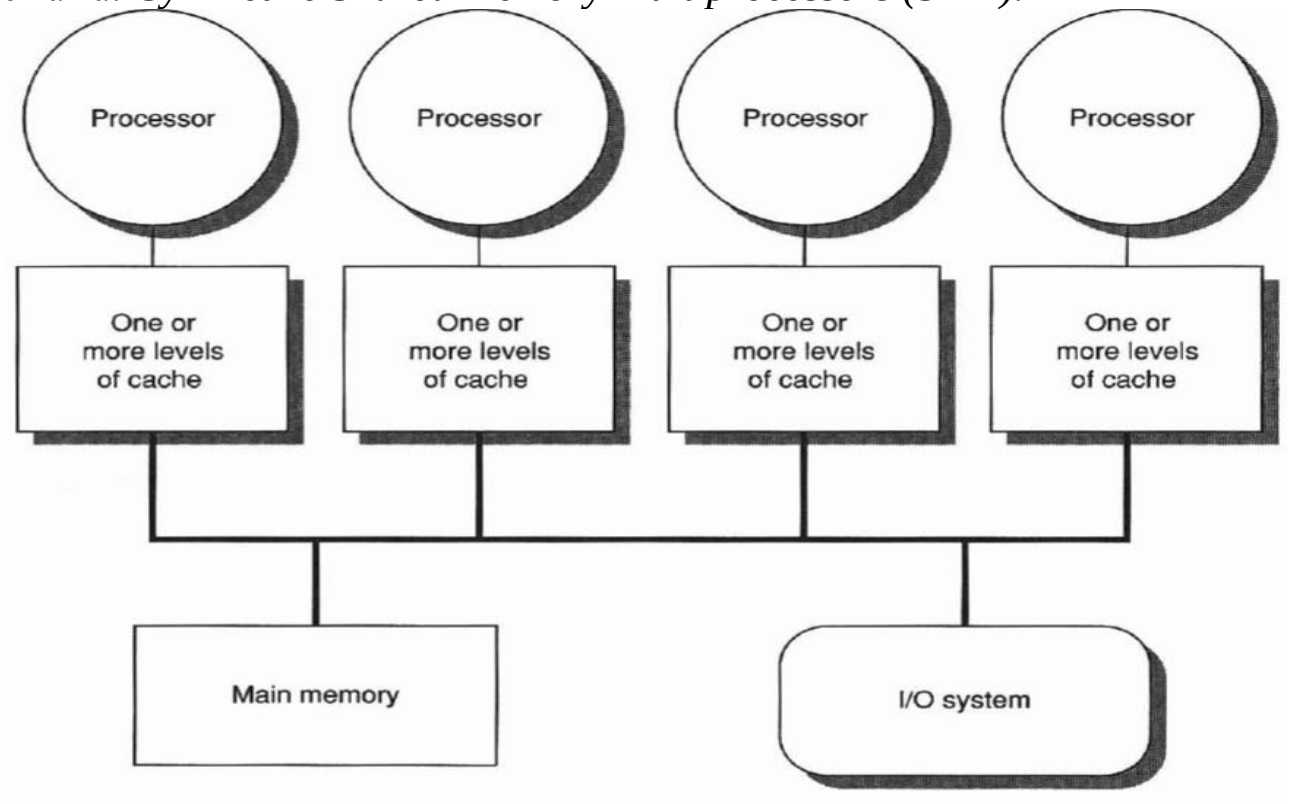
Anno Accademico 2010/2011

INDICE

1. MULTIPROCESSORI

Una architettura con più CPU che condividono la stessa memoria primaria viene detto **multiprocessore**. In un sistema multiprocessore tutti i processi che girano sulle varie CPU condividono un unico spazio di indirizzamento logico,mappato su una memoria fisica che può però anche essere distribuita fra i vari processori. Ogni processo può leggere e scrivere un dato in memoria semplicemente usando una load o una store, e la comunicazione fra processi avviene attraverso la memoria condivisa. In un multiprocessore, la struttura della memoria dipende dal numero di processori ma, dato che tale valore è destinato a variare nel tempo, si sceglie di classificare i multiprocessori in base all'organizzazione della memoria. Esistono due tipi di architetture:

_ Architettura a Memoria Condivisa Centralizzata detta **Uniform Memory Access (UMA):** Questo tipo di architettura viene così chiamata perché tutti i processori condividono un'unica memoria primaria centralizzata, e quindi *ogni CPU ha lo stesso tempo di accesso alla memoria.* I processori di questa architettura sono chiamati *Symmetric Shared-memory Multiprocessors (SMP).*



Struttura base di un multiprocessore con memoria condivisa centralizzata.

Quando una CPU vuole leggere una locazione di memoria verifica prima che il bus sia libero, invia la richiesta al modulo di interfaccia della memoria e attende sul bus che arrivi il valore richiesto. Di fatto, i processori Arm Cortex dual (o quad) core sono sostanzialmente dei piccoli sistemi multiprocessore di tipo UMA, in cui è la cache L2 a costituire il canale di comunicazione.

_ Architettura a Memoria Distribuita detta **Non Uniform Memory Access (NUMA):** in questi sistemi i vari processori vedono ancora uno spazio di indirizzamento logico unico, ma la memoria fisica è distribuita fra le varie CPU, e quindi *i tempi di accesso ai dati variano a seconda che siano nella RAM locale o in una remota* (da cui appunto il termine NUMA).I processori in questa architettura sono chiamati anche *Distributed Shared Memory (DSM).*

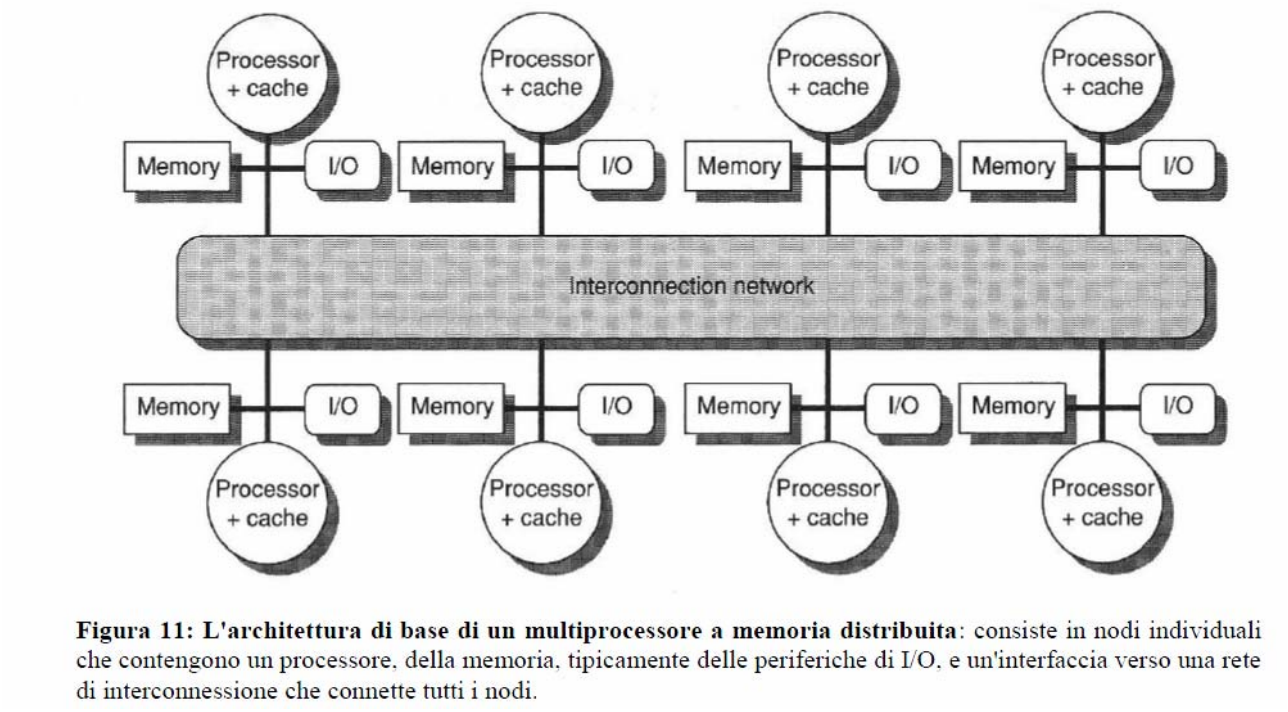


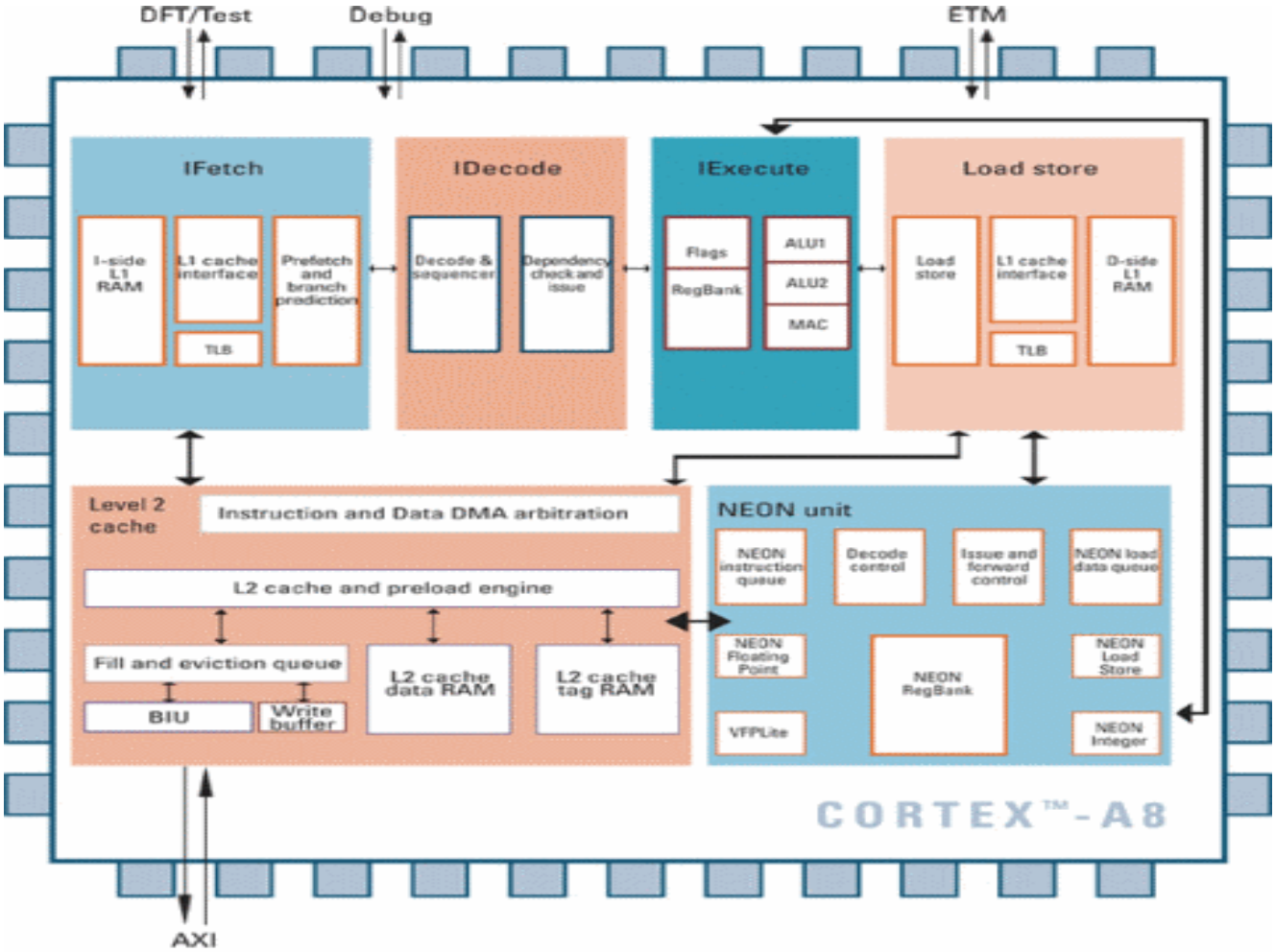
Figura 11: L'architettura di base di un multiprocessore a memoria distribuita: consiste in nodi individuali che contengono un processore, della memoria, tipicamente delle periferiche di I/O, e un'interfaccia verso una rete di interconnessione che connette tutti i nodi.

La comunicazione in questi sistemi deve essere effettuata con il passaggio di messaggi in maniera esplicita tra i processori. L'utilizzo di cache a più livelli può aiutare a ridurre la latenza per l'accesso locale ad un'informazione e le richieste di larghezza di banda di memoria di un processore. Tuttavia insorgono problemi per i dati condivisi, che possono venire replicati in più cache distinte, poiché si deve garantire che le informazioni condivise siano viste nello stesso modo da tutti i processori che accedono alla propria cache locale; questi problemi prendono il nome di coerenza e consistenza della cache, e vengono risolti tramite l'impiego di hardware e protocolli appositi.

2. Introduzione agli ARM Cortex

L'**ARM Cortex** è una famiglia di microprocessori presentata nel 2005 dalla ARM Holdings e basati sul set di istruzioni ARMv7 e ARMv6(quest'ultimo implementato soltanto nei processori Cortex M0 e M1). La famiglia Cortex è formata da una serie di blocchi funzionali che possono essere collegati tra loro al fine di soddisfare le esigenze dei clienti, quindi un specifico processore Cortex non ha necessariamente tutte le unità funzionali della famiglia. I processori Cortex sono disponibili in configurazione singolo core o multicore e per ogni famiglia esistono più core con prestazioni diversi. Rispetto alla versione 6(ARMV6) la famiglia Cortex introduce le seguenti novità:

- **L'unita NEON** sviluppata per eseguire operazioni SIMD su vettori di 64 o 128 bit. L'unità è dotata di registri dedicati ed i vettori possono contenere numeri interi a 16 o 32 bit o numeri in virgola mobile a singola precisione a 32 bit. L'unità opera in parallelo alla pipeline principale, la pipeline principale interviene solo durante il caricamento delle istruzioni da eseguire.Questa unità è utilizzata soprattutto nei processori Cortex-A series, di cui un esempio è riportato nella figura seguente:

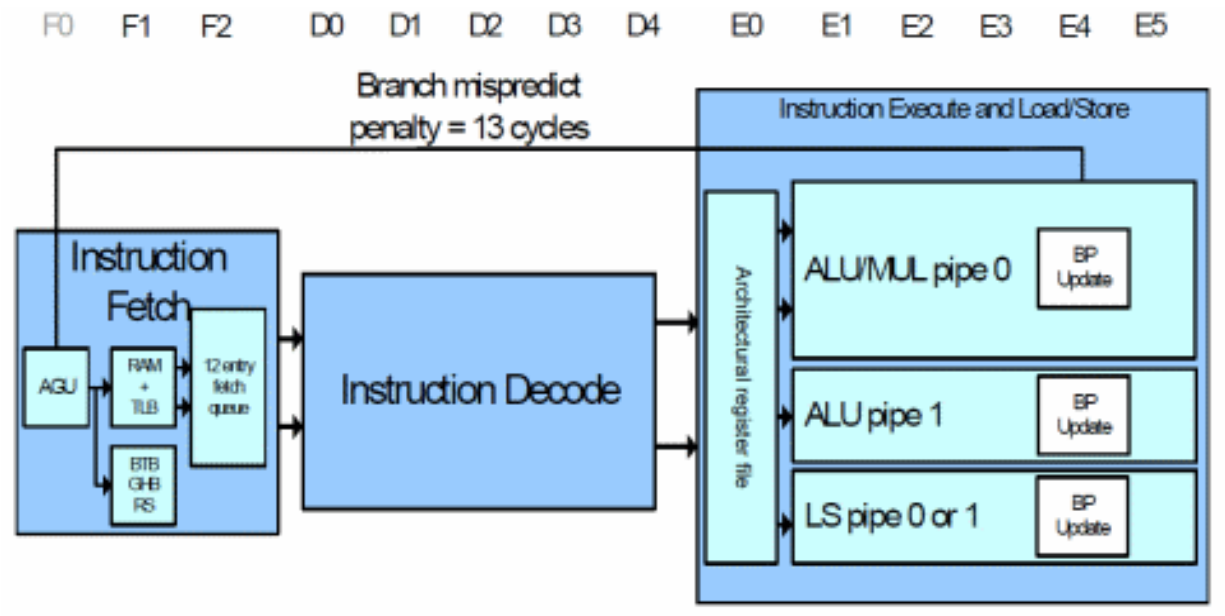


- L'unità in virgola mobile **VFPv3** raddoppia i registri della precedente versione portandoli a 32 e introduce alcune nuove operazioni. La tecnologia VFP è nata per fornire operazioni in grado di trattare dati in virgola mobile a singola e doppia precisione in modo economico ma pienamente compatibile con la standard ANSI/IEEE Std 754-1985 Standard for Binary Floating-Point Arithmetic. VFPv3 fornisce istruzioni per applicazioni tipo compressioni, decompressioni, grafica tridimensionale, analisi audio e altro. Questa risulta utile per dispositivi tipo PDA, smartphone, set-top box e applicazioni di automazione e controllo.
- Il set di istruzioni **Thumb-EE** è un derivato del set di istruzioni Thumb-2 ed è nato per sostituire le istruzioni Jazelle. Queste istruzioni vengono utilizzate per accelerare l'esecuzione di codice eseguito da macchine virtuali come quello richiesto dal linguaggio Java (codice java). Thumb-2 è stato specificatamente progettato per gestire codice generato in tempo reale (per esempio durante la esecuzione di codice Just in time). La tecnologia Thumb 2EE è stata progetta per linguaggi come Java, C++, Perl e python in modo da generare codice compilato di dimensioni ridotte ma senza impattare sulle prestazioni.Le nuove istruzioni fornite permettono di controllare automaticamente i puntatori nulli prima di ogni load o store, permettono di gestire l'eventuale sfondamento degli array, la gestione delle diramazioni e molte altre caratteristiche fornite da linguaggi ad alto livello come l'istanziamento di memoria per i nuovi oggetti.
- **TrustZone** è una modalità di esecuzione sicura nata per permettere l'esecuzione di codice sicuro o per eseguire meccanismi di digital rights management (DRM)(ovvero di gestione dei diritti digitali(copyright)).Rappresenta dunque un sistema di sicurezza per le piattaforme di calcolo ad alte prestazioni volto a garantire inoltre la sicurezza dei dati(per esempio dei dati personali attraverso meccanismi di autenticazione e autorizzazione dell'utente), la sicurezza nei pagamenti,e lo scambio dei messaggi e delle comunicazioni attraverso dei canali sicuri.
- **pipeline superscalare** costituita da 13 stadi e in grado di eseguire due istruzioni *in-order.*

L' Architettura Cortex, **è suddivisa in tre gruppi**, ognuno contraddistinto da una lettera che ne richiama l'appartenenza: ***Application* (A), *Realtime* (R) e *Microcontroller* (M)**.L'idea è quella di avere un core comune da specializzare specificamente in base al target d'utilizzo. Il più ricco è **il gruppo A,** che **include tutta una serie di caratteristiche ed estensioni necessarie per l'utilizzo in ambito "tradizionale" (netbook, server, telefoni cellulari evoluti (smartphone)**e più in generale le applicazioni che necessitano di potenza di calcolo e flessibilità**)**: Questa è la serie più completa e oltre al set di istruzioni classico ARM gestisce le istruzioni Thumb-2, Thumb-EE, include le unità Vector Floating Point (VFP) e NEON (unita SIMD). Il processore è dotato di cache di primo e secondo livello(cache L1 ed L2), PMMU e tecnologia TrustZone per la sicurezza e i DRM.**La serie R** è sviluppata per applicazioni **realtime**, il set di istruzioni Thumb-2 è presente, le istruzioni Vector

Floating Point sono opzionali e la cache è configurabile. La protezione della memoria è presente opzionalmente tramite la MPU, un'unità più limitata della PMMU. **La serie M** è la serie più ridotta, implementa il set di istruzioni Thumb oThumb-2 o entrambi, la cache non è presente e la MPU è opzionale. **Notiamo subito un punto di rottura col passato** rappresentato dal fatto che l'approccio usato in precedenza per implementare funzionalità di tipo SIMD è stato molto più "morbido", modificando il core aggiungendo nuovi flag e istruzioni per accomodare l'applicazione di operazioni (in genere dello stesso tipo) su più dati al medesimo tempo, ma continuando a sfruttare i registri del processore.Altro cambiamento sensibile è dovuto alla sostanziale **rimozione della vecchia modalità *Jazelle***. In realtà viene mantenuta per retrocompatibilità, ma nessun *bytecode* viene accelerato in hardware: ognuno di essi comporta l'invocazione dell'*handler* che si dovrà occupare della sua interpretazione e relativa esecuzione. Al posto di Jazelle si sfrutta una modifica a Thumb-2, che offre **un ambiente più flessibile e generale** (non legato esclusivamente a Java).Non sarà gradita a molti, invece, la soluzione rappresentata da TrustZone. Quanti avevano pensato di abbandonare *x86* & affini a causa della famigerata **tecnologia *Trusted Computing* o *Palladium***, confidando quindi nella "liberazione" grazie ad ARM, si ritroveranno già ad ardere nella brace del **software *"trusted"***, magari senza neppure saperlo e addirittura vantandosi dell'acquisto. Piacevolissimo e sicuramente apprezzato da tutti è stato, invece, l'arrivo della *superpipeline*. A vent'anni dall'introduzione del primo esemplare della famiglia, e dopo che tante altre case sono saltate sul carro dell'esecuzione di più istruzioni per ciclo di clock, **ARM compie un passo storico e sceglie la soluzione che, 12 anni prima, ha reso famoso il Pentium di Intel: eseguirne due in ordine**.Non che fosse impossibile farlo prima, sia chiaro, ma considerato il mercato di riferimento di Acorn prima e di ARM Ltd poi, la ricerca delle prestazioni elevate a tutti i costi non era certo uno dei suoi obiettivi, per cui ha preferito conservare le peculiarità della sua architettura (in primis un core ridotto all'osso: sui 35mila transistor circa), aggiornandola, estendendola, passando a processi produttivi migliori e innalzando il clock.Certamente **negli ultimi anni** e con la sempre più larga e variegata adozione dei suoi microprocessori, **è diventato sempre più pressante aumentarne la velocità di esecuzione**. Infatti uno dei cavalli di battaglia, la *pipeline* corta (appena 3 stadi), era stata sacrificato già da qualche tempo sull'altare della frequenza del clock.Si sono già viste, infatti, CPU con pipeline lunghe da 5 fino a 9 stadi, e sappiamo bene i problemi che ciò comporta: il rischio di uno svuotamento della pipeline con relativo nuovo caricamento delle istruzioni è sempre dietro l'angolo, e il decadimento prestazionale assicurato.**I Cortex segnano il record assoluto per questa famiglia, arrivando a ben 13 stadi.** Tanti (troppi, direi considerata la sua storia), se pensiamo che un vecchio *Athlon* ne aveva 10, e 12 i più recenti *Athlon64*. Ciò ha richiesto l'introduzione di meccanismi di *branch prediction* efficaci (una *global history* di 512 entry, un *branch target* di 4096 a 2 bit, e un *return stack* di 8). Lo sviluppo di un processore a 13 stadi si è reso necessario inoltre per innalzare la frequenza di funzionamento del processore. Al fine di evitare che i salti condizionati deprimessero eccessivamente le prestazioni per via della pipeline lunga tutti i core implementano un'unità di predizione delle diramazioni che secondo le

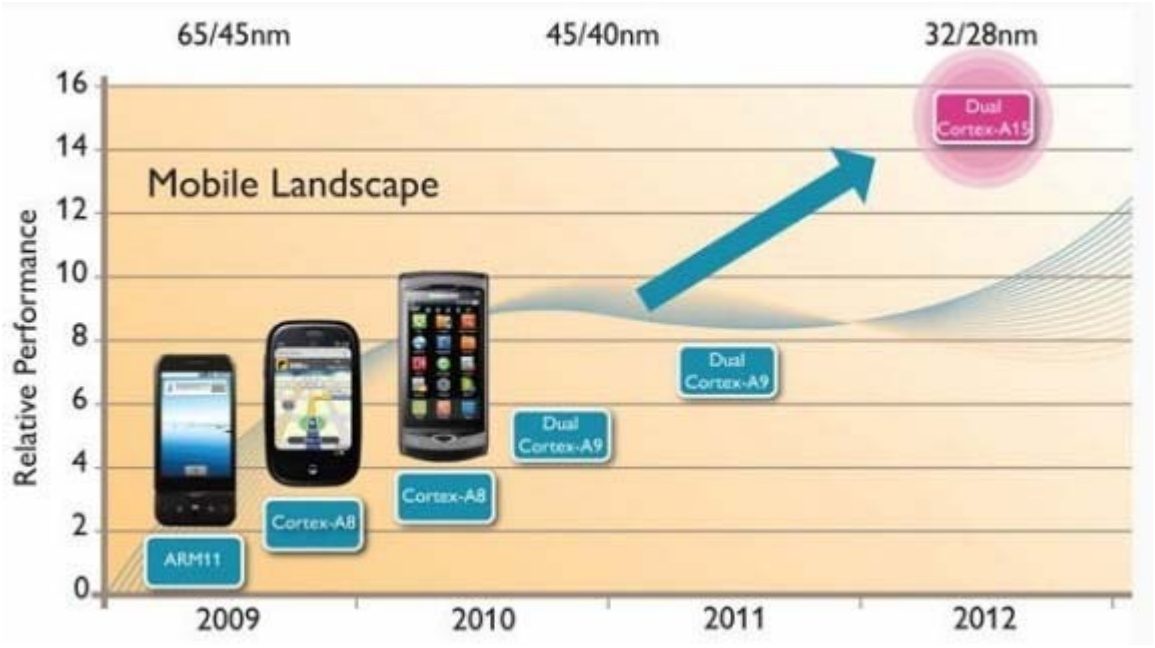
dichiarazioni del costruttore **predice correttamente il 95% dei salti**: un risultato di tutto rispetto, che garantisce un ottimo livello di prestazioni raggiunte. A titolo di confronto, un Athlon raggiunge gli stessi risultati, ma con una branch history di 2048 entry, un branch target di altrettanti elementi, e un return stack di 12.C'è però da dire che gli Athlon hanno raggiunto frequenze elevatissime (2,3Ghz con un modello della serie *Barton*) considerati gli appena 10 stadi e il fatto che si siano fermati a un processo produttivo "da preistoria" (130nm), mentre i Cortex-A8 hanno toccato il Ghz, i Cortex-A9 i 2Ghz, solamente gli ultimi Cortex-A15 (la cui uscita è prevista per il 2012) riusciranno a superare in frequenza gli Athlon con una frequenza di 2,5Ghz. Tornando alla **pipeline superscalare,** l'**implementazione *in-order*** è molto semplice:



Pipeline Superscalare: Esecuzione di 2 istruzioni *in order*

la prima istruzione viene smistata alla prima *ALU* (la 0) e quella seguente alla seconda (ALU 1). Può sembrare una forte limitazione, ma entrambe le ALU possono eseguire qualunque operazione (a parte la moltiplicazione, affidata soltanto alla prima, ma si tratti di un'operazione rara) e possono accedere all'unità di *Load/Store*. **Si tratta di un design conservativo, ma anche molto semplice e poco costoso da implementare**, che può garantire un buon miglioramento prestazionale specialmente se i compilatori faranno il loro dovere cercando di ridurre le dipendenze dei risultati e smistando le istruzioni alle due ALU in maniera ottimale. Concludendo la famiglia Cortex A è stata sviluppata al fine di ottenere prestazioni elevate e consumi ridotti. I processori di questa serie si suddividono in tre core, il core 5, 8 e 9. Il core 5 ha una pipeline a 8 stadi, il core 8 ha due pipeline a 13 stadi e il core 9 ha due pipeline a 8 stadi. Il processore al fine di non richiedere un numero eccessivo di transistor non implementa nel core 8 l'esecuzione fuori ordine delle istruzioni, mentre il core 9

introduce anche questa caratteristica, il core 3 avendo una sola pipeline non può eseguire più di un'istruzione per ciclo di clock. Nel core 8 la prima istruzione viene caricata dalla prima pipeline, la seconda istruzione viene caricata dalla seconda pipeline; nel caso di vincoli l'istruzione vincolata viene bloccata fino a quando l'altra istruzione non è completata e quindi il vincolo è risolto. Il core 9 gestendo l'esecuzione fuori ordine analizza il codice e ricerca due istruzioni non vincolate per eseguirle in parallelo, fornendo prestazioni migliori del core 8. Il core 9 gestisce anche la ridenominazione dei registri al fine di ridurre i vincoli e migliorare l'esecuzione parallela delle istruzioni. I core 5 e 9 possono essere assemblati in integrati che possono contenere fino a 4 core. Il processore futuro della serie Cortex A sarà il multi-core Cortex A15 (Eagle), in grado di lavorare fino a una frequenza 2,5 GHz in configurazione quad-core - ma saranno possibili anche soluzioni a 8/16 core a seconda dell'ambito in cui verranno implementati i chip. Secondo l'azienda **il nuovo system on chip è cinque volte più potente di dual-core Cortex A8 a 1 GHz.** Al centro dei questa architettura c'è Amba 4, bus integrato nel chip che permette a cluster di quattro core di comunicare tra loro.



3. ARM CORTEX A-SERIES

_ARM Cortex A5;

_ARM Cortex A8;

_ARM Cortex A9.

PROCESSORE ARM CORTEX A5

Di seguito sono riportati dei diagrammi a blocchi di un processore Cortex A5.

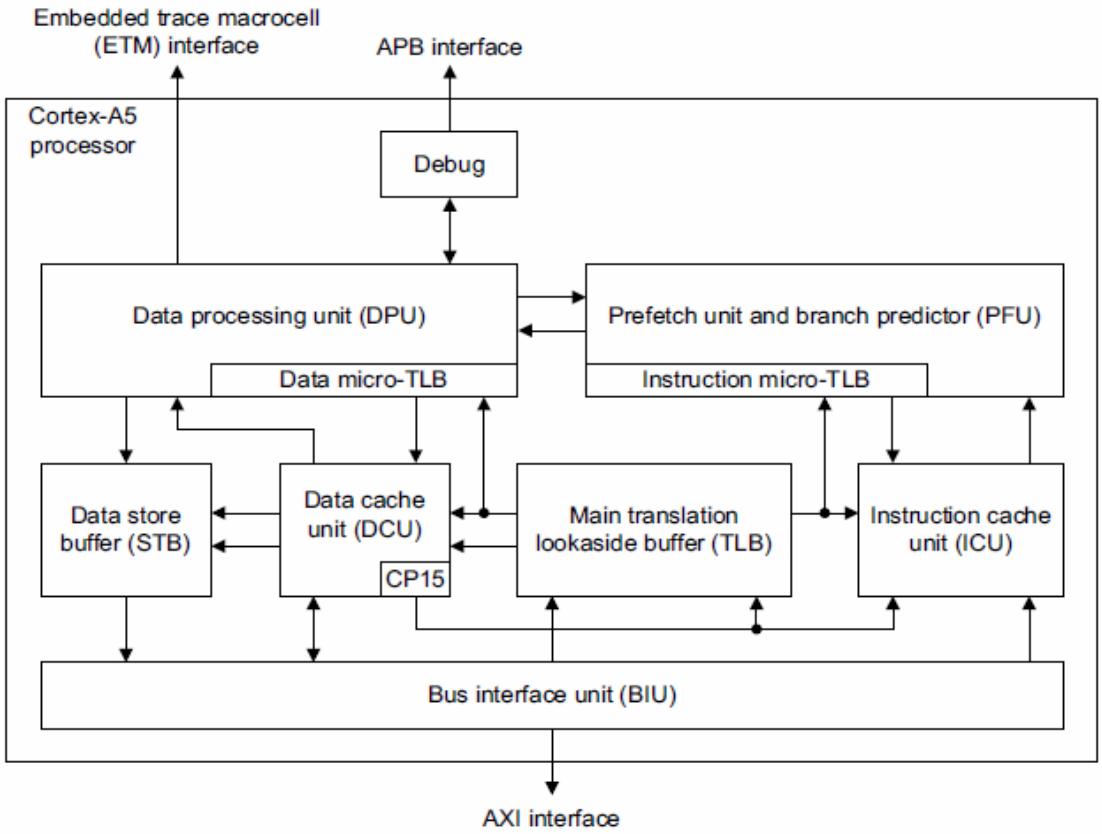
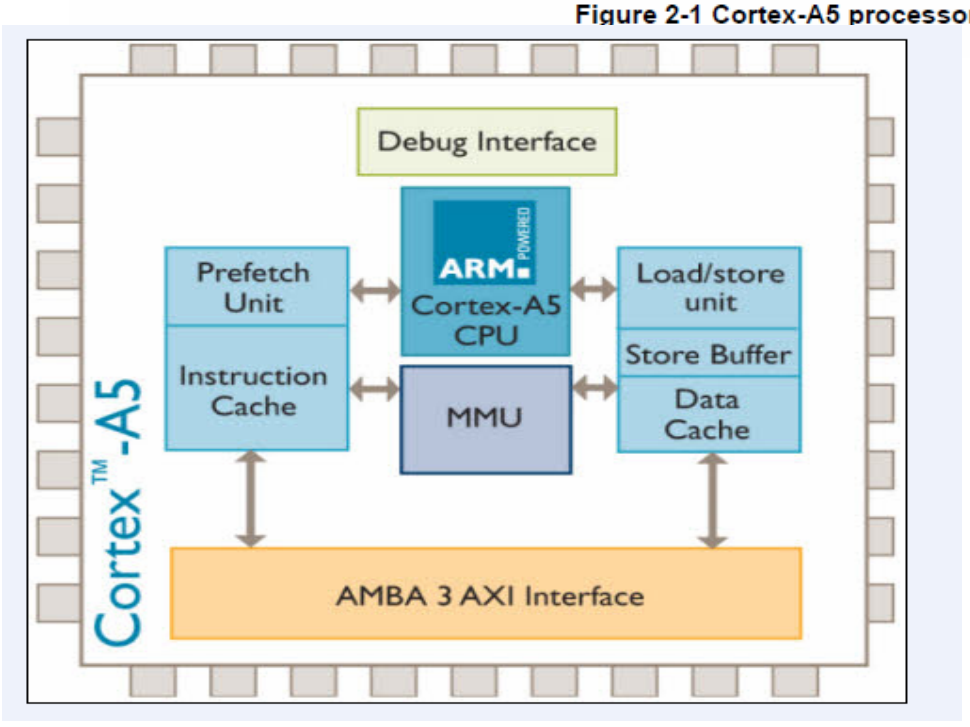


Figure 2-1 Cortex-A5 processor top-level diagram



Di seguito sono illustrati i relativi blocchi e le loro funzioni:

_ *Data Processing Unit(DPU):*Conserva lo stato del processore, per esempio i registri general-purpose, i registri di stato(status register) e i registri di controllo(controller register).Decodifica ed esegue le istruzioni, ed opera sui dati contenuti nei registri. Le istruzioni sono fornite alla DPU dalla Prefetch Unit (PFU),infatti quest'ultima ha il compito di prelevare le istruzioni dalla Instruction Cache Unit (ICU).La DPU esegue le istruzioni che richiedono il trasferimento dei dati da o verso la memoria del sistema interfacciandosi con la Data Cache Unit (DCU), che gestisce tutte le operazioni di load and store(cioè di caricamento e d'immagazzinamento dei dati).

_ *System Control Coprocessor :* Il coprocessore di controllo del sistema, CP15, fornisce la configurazione e il controllo della memoria di sistema e le sue funzionalità associate.

_*Instruction side memory system:*La memoria di sistema delle istruzioni e composta da:

_*Prefetch Unit(PFU):* ottiene le istruzioni dalla cache delle istruzioni (ICU) o dalla memoria esterna, e predice l'esito dei salti(branches) nell'instruction stream(flusso di istruzioni),in seguito passa le istruzioni alla DPU perché sia processate.La PFU contiene inoltre 4 cache in cui sono conservati gli indirizzi dei salti *Branch Target Address Cache* (BTAC).La PFU contiene anche uno stack di ritorno *Return Stack a 4 entry* in cui sono inseriti gli indirizzi di ritorno dalla procedura chiamante, e un *Branch Predictor* che è un componente della BPU(Branch prediction unit contenuta sempre all'interno della PFU), che ha il compito di prevedere l'esito di un'operazione su cui si basa l'accettazione di una istruzione di salto condizionato , cioè in poche parole ha il compito di prevedere la direzione del salto.

_*Instruction Cache Unit:* L'unità di cache delle istruzioni contiene il controller di cache istruzioni (Instruction cache controller) e i suoi relativi buffer.Le ICache dei Cortex A5 sono a due vie set associative e usano dei tag fisici indicizzati Virtually Indexed Physically Tagged (VIPT), e contengono fino a otto istruzioni ARM e fino a un massimo di sedici istruzioni Thumb.

_ *Data side memory system* : La data side memory system è composta da:

_ *Data Cache Unit DCU(cache contenente i dati)* : suddivisa nei seguenti sottoblocchi:

_*il livello (L1) data cache controller*, che genera i segnali di controllo per i tag embedded, per i dati e per la memoria RAMs,e controlla gli accessi alle risorse di memoria.La Data Cache è una memoria set associativa a 4 vie che usa uno schema a tag *Physically Indexed Physically Tagged* (PIPT) che consente attraverso il TAG,la data-micro TLB(nella DPU),e la main TLB di verificare se un determinato dato è presente nella memoria cache.

_ Una *pipeline di load and store* che si interfaccia con la TLB e la DPU.

_Un coprocessore di sistema *system coprocessor (CP15) controller* che esegue operazioni di manutenzione della cache direttamente sulla cache di dati e, indirettamente, sulla cache istruzioni attraverso un'interfaccia con l'ICU.

_ *Store Buffer(STB)*: Conserva i dati, e tutte le operazioni in esecuzione nella DPU che sono state prelevate dalla pipeline di load and store.Per immagazzinare un dato o una operazione nella STB, può richiedere un accesso alla cache RAMs nella DCU,richiedendo al BIU di iniziare la procedura di linefills(ovvero di lettura del dato nella memoria cache DCU e se non è presente nella memoria esterna), oppure richiedendo al BIU di scrivere i dati nell'interfaccia esterna AXI interface.

_*Bus Interface Unit and AXI interface:* contiene le interface esterne di tipo master e vari Buffer e separa le interfacce della DCU da quelle di STB. Contiene una porta AMBA AXI a 64 bit condivisa sia dai dati che dalle operazioni.

_*L1 memory system:* La memoria di sistema L1 possiede le seguente caratteristiche:

_ è composta dall'instruction side memory system, e data side memory system;

_ separa la cache delle istruzioni da quella dei dati;

_ Esporta gli attributi di memoria per la memoria di sistema L2;

_ Dimensione della cache dei dati e delle istruzioni compresa tra 4KB e 64KB;

_ Politica di sostituzione della cache pseudo-casuale;

_ Possibilità di disabilitare ogni cache indipendentemente;

_Streaming dei dati sequenziali attraverso le operazioni LDM e LDRD, e attraverso le operazioni di lettura;

_Critical word first filling della cache in presenza di una cache miss:in presenza di read miss la parola mancante viene richiesta alla memoria di livello inferiore e inviata al processore non appena arriva. Il processore inizia ugualmente l'esecuzione nonostante sia in attesa dei dati richiesti;

_Implementazione dei blocchi della RAM e dei TAG associati usando dei compilatori standar ASIC RAM.

_ *L2 AXI interfaces :* L'interfaccia L2 AXI abilita la memoria di sistema L1 ad avere accesso alle periferiche e alla memoria esterna usando una porta master AXI.

_*Media Processing Engine:* L'opzionale Media Processing Engine implementa la tecnologia NEON, ovvero un architettura di elaborazione del segnale che aggiunge nuove istruzioni mirate ai processi di elaborazione dell'audio, video,immagini e grafica 3-D, e di trattamento della parola.Le Istruzioni discusse sono quelle Avanzate SIMD.

_*Floating-Point Unit:* l'unità floating point FPU implementa l'architettura ARMv7 VFPv4-D16 ed include il file di registro VFP e I registri di stato (status register). Questà unita esegue le operazioni di tipo floating-point sui dati conservati nel file di registro VFP.

_Debug: Il processore Cortex A5 ha un interfaccia di debug chiamata *Advanced Peripheral Bus version 3*(APBv3).Ciò consente l'accesso al sistema per eseguire il debug delle risorse, per esempio il settaggio dei breakpoints e dei watchpoints.
_Performance monitoring(monitoraggio delle prestazioni): il processore contiene un misuratore delle prestazioni del processore e quindi un monitor configurato per raccogliere i dati sul funzionamento del processore e della memoria di sistema.
_Virtualization extensions: il processore contiene un numero di estensioni dell'ARMv7 che garantiscono l'efficienza nell'ambito della virtualizzazione.Queste estensioni sono implementate usando dei campi contenuti nei registri di configurazione *Secure Configuration Register SCR*, e una coppia di nuovi registri the *virtualization control register VCR* e *virtualization interupt register VIR.* Lo stato delle interruzioni fisiche o virtuali è contenuto nel *Interupt Status Register.*
*_MMU:*La memory Management Unit lavora con le memorie di sistema L1 e L2 per tradurre gli indirizzi virtuali *virtual addresses VA* usati dai set d'istruzioni dell'architettura in indirizzi fisici *physical addresses PA* usati dalla memoria di sistema . Per la traduzione di questi indirizzi virtuali in indirizzi fisici vengono usati 2 livelli di MMU. Il primo livello di MMU usa un micro TLB (*Translation Lookside Buffer*) contenuta nella PFU (*IuTLB)* per la lettura degli indirizzi fisici delle istruzioni ,e una seconda micro TLB nella DPU (*DuTLB*) per la lettura degli indirizzi fisici dei dati. Il secondo livello di MMU è composto dalla main TLB(che contiene i TAG e gli indici delle pagine fisiche di tutte le istruzioni e dati).Un miss in una qualsiasi delle due micro TLB si traduce in una richiesta alla Main TLB. Sia le micro TLB, e sia la main TLB in caso di hit(cioè l'indirizzo è presente nella tabella) mi restituiscono il relativo indirizzo fisico in memoria. Le micro TLB sono delle memorie set associative a due vie a 10 entry, mentre la main TLB è una memoria set associativa a due vie a 128 entry.

INTERFACCE

Il processore ha le seguenti interfacce:
• AMBA AXI interfaces
• APB CoreSight Debug interface
• DFT interface
• ETM interface.

ETM Interface

L'interfaccia *Embedded Trace Macrocell* (ETM) permette di collegare un'unità esterna ETM al processore, per codice in tempo reale e dati di analisi del core in un sistema embedded.
L'interfaccia di ETM raccoglie i segnali di processori diversi e instrada questi segnali verso altri dispositivi. Funziona alla massima velocità del processore. Di seguito sono riportati i segnali dell'interfaccia ETM .

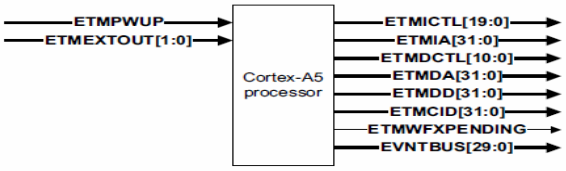


Figure 2-2 ETM interface signals

CARATTERISTICHE DEL PROCESSORE

I processori Cortex A5 implementano l'architettura ARMv7 che include:

- Set d'istruzioni ARM a 32 bit;
- Set d'istruzioni Thumb, con lunghezza delle istruzioni può essere di 16 o 32 bit;
- Set d'istruzioni ThumbEE;
- Tecnologia TrustZone e Jazelle RCT;
- VFPv4 Floating point con 16 registri VFPv4-D16 a doppia precisione, e estensioni avanzate SIMD NEON.La FPU consente di eseguire operazioni a singola precisione e a doppia precisione su dati in virgola mobile quali,addizione, sottrazione, moltiplicazione,divisione, moltiplicazione e accumulo,radice quadrata;
- Tipi di dati: doubleword a 64-bit,word a 32-bit(4byte),halfword a 16-bit(2 byte),byte a 8-bit (1 byte); e rappresentazione dei dati di tipo little endian o big endian;
- Pipeline a 8 Stadi.

CARATTERISTICHE TECNICHE E PRESTAZIONI

	Cortex-A5
Architecture	**ARMv7-A Cortex**
Dhrystone Performance	1.57 DMIPS / MHz per core
Multicore	1-4 cores Single core version also available
Debug & Trace	CoreSight™ DK-A5

Il primo processore Cortex A5 MPCore per telefoni cellulari

La serie di Processori Cortex A5 è stata presentata in due versioni di chip: la **TSMC 40G**, indirizzata alla produzione di architetture complesse, votate alla prestazioni, mentre il **TSMC 40LP** sarà indirizzato ai chip che del risparmio energetico fanno un punto di valore imprescindibile. Entrambi i chip hanno un processo produttivo a 40 nm. Di seguito sono riportate le prestazioni:

ARM Cortex-A5 Performance, Power, and Area		
	TSMC 40LP	**TSMC 40G**
Process Type/Nominal Voltage	low leakage, 1.1V	performance, 1.0V
Performance or Frequency Optimized	Frequency	Frequency
Frequency	530~600 MHz	>1GHz
Area excluding RAMs/cache	0.27mm²	0.27mm²
Area with 16K/16K cache	0.53mm²	0.53mm²
Area with 16K/16K cache + NEON	0.68mm²	0.68mm²
Dynamic Power	0.12 mW/MHz	<0.08mW/MHz
Energy Efficiency	13 DMIPS/mW	>20 DMIPS/mW

Applicazioni D'uso

Il processore ARM Cortex ™-A5 è il processore a più basso consumo energetico, e a più a basso costo in grado di fornire internet alla più ampia gamma possibile di dispositivi: dagli smartphone entry level, i portatili a basso costo e dispositivi mobili intelligenti (embedded), dispositivi industriali e di consumo .

Applicazioni

- **Telefoni Cellulari :**
  SmartPhones Entry Level(cioè di fascia di mercato bassa, per esempio Apple Iphone),Apple Ipad,Feauture Phones.

- **Dispositivi Audio**: Lettori MP3, Dispositivi GSM,Radio Digitale,Registratori.

- **Home/ Consumer :** TV Digitale via cavo, Console di Gioco(Nintendo DS),Macchine Fotografiche digitali,Netbook a basso costo.
- **Embedded/Industrial :** Smart Meter(contatori domestici della luce o del gas),Dispositivi MPU(Programmatori di chip per esempio di EEPROM).

PROCESSORE ARM CORTEX A8

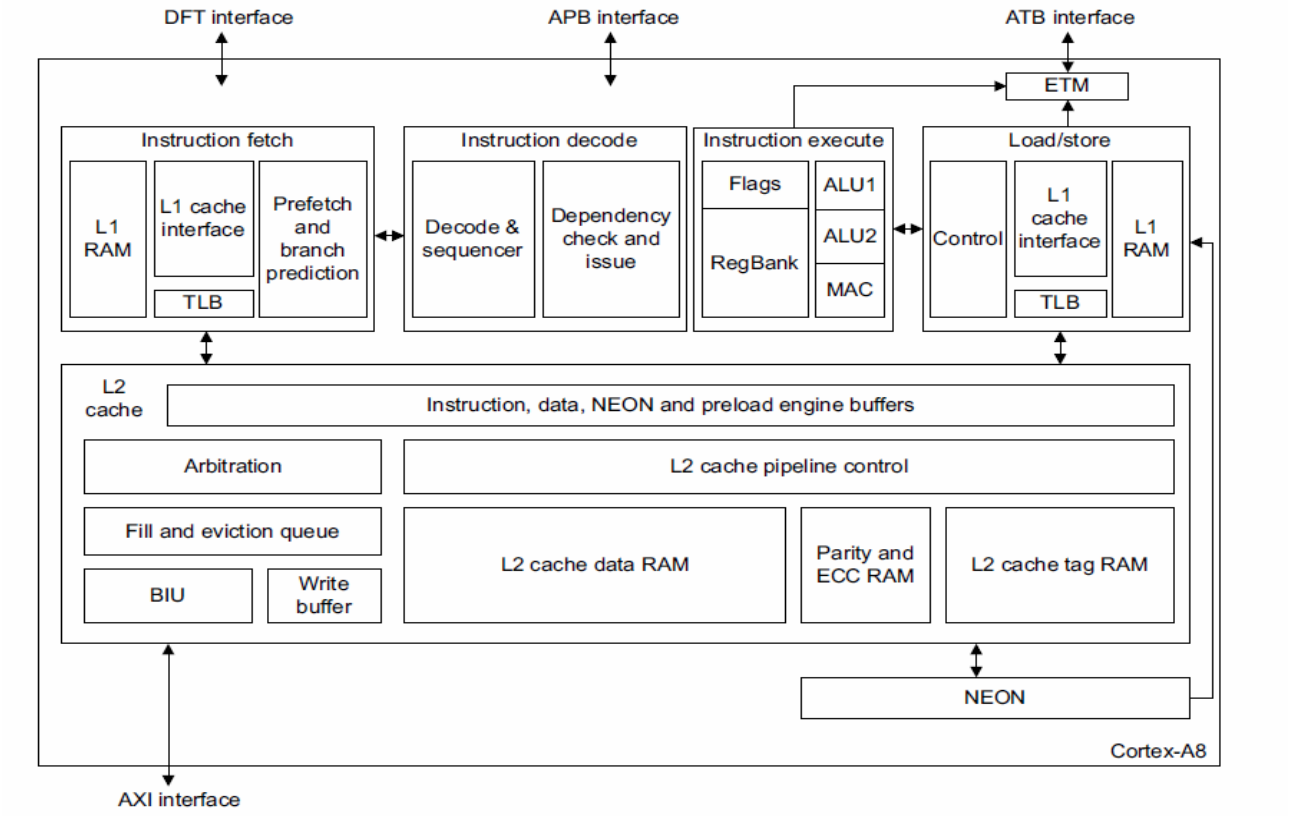


Figure 1-1 Cortex-A8 block diagram

I principali componenti del processore sono:

_*Instruction fetch :* L'unità instruction fetch(unità di prelievo delle istruzioni) prevede il flusso delle istruzioni, recupera le istruzioni dalla memoria cache delle istruzioni L1 , e colloca le istruzioni in un buffer per il consumo da parte della pipeline di decodifica.
_*Instruction decode:* L'unità di decodifica delle istruzioni(instruction decode unit) decodifica le istruzioni e le sequenze di tutti le istruzioni ARM e Thumb-2 incluse quelle del coprocessore debug di controllo, CP14, e le istruzioni del sistema di controllo del coprocessore, CP15.L'unità di decodifica delle istruzioni gestisce inoltre:

- Le eccezioni;
- Gli eventi di Debug;
- L'inizializzazione del Reset;
- La memoria *Memory Built-In Self Test(MBIST);*
- L'attesa per le interruzioni *wait-for-interrupt;*
- Altri eventi insoliti.

_*Instruction execute:*L'unità di esecuzione delle istruzioni è composta da due pipeline simmetriche *Arithmetic Logical Unit* (ALU),da un generatore di indirizzi per le istruzioni di load and store,e da una pipeline di moltiplicazione. L'esecuzione sulle pipeline opera su registri di tipo write back.
L'unità di esecuzione delle istruzioni:

- Esegue tutti gli interi della ALU e le operazioni di moltiplicazione incluse le operazioni di generazione dei FLAG;
- Genera gli indirizzi virtuali per le istruzioni di load and store, e un valore di base write-back quando richiesto;
- Fornisce i dati formattati per le operazioni di load e store ed anche i FLAG.
- Gestisce i salti(branches)e le variazioni nel flusso delle istruzioni e valuta i codici delle istruzioni di condizione.

_*Load and Store:* L'unità Load and store include l'intera memoria di sistema dei dati L1 e una pipeline di load and store.Include:

- Una cache dei dati L1;
- Una tabella TLB;
- Un buffer d'immagazzinamento dei dati *store;*
- Un buffer NEON sempre di tipo store;
- I dati interi formattati dalla Instruction execute che deve essere caricati(*load*);
- I dati interi formattati dalla Instruction execute che devono essere salvati(*store*).

_ *cache L2*: l'unità cache L2 è composta da una cache L2 e da un buffer *Buffer Interface Unit BIU* . Gestisce i Miss nella chache L1 attraverso l'unità load and store e l'unità Instruciton fetch.
_ *NEON:* L'unità Neon è composta da pipeline Neon a 10 stadi che decodificano ed eseguono il set delle istruzioni Advanced SIMD.L'unità include:

- The Neon Instruction queue , cioè una coda per le istruzioni;
- the NEON load data queue, cioè una coda per i dati di caricamento;

- due pipeline per la decodifica;
- tre pipeline per l'esecuzione delle istruzioni Advanced SIMD su dati di tipo integer;
- due pipeline per l'esecuzione delle istruzioni Advanced SIMD su dati di tipo floating-point;
- una pipeline per l'esecuzione delle istruzioni Advanced SIMD e le istruzioni VFP load and store;
- un'unità VFP per la piena esecuzione di tutto il set di istruzioni VFPv3.

_ *ETM* : l'unità ETM è una macrocella che filtra e comprime i dati e le istruzioni in modo che possano essere utilizzati nei sistemi di debugging e di profiling.Ha un interfaccia esterna al processore chiamata *Advanced Trace Bus* (ATB).

INTERFACCE

Il processore ha le seguenti interfacce esterne:

- AMBA AXI interface: L'interfaccia Bus AXI è l'interfaccia principale per il bus di sistema. Consente di accedere alla cache L2,con accessi di tipo non cacheable alle istruzioni e ai dati.Supporta dati da 64bit o 128bit.
- AMBA APB interface :il Cortex A8 implementa un interfaccia APB di tipo slave che abilita gli accessi all'unità ETM,e ai registri di debug.
- AMBA ATB interface : interfaccia da cui escono le informazioni di debugging emesse dall'unità ETM.
- DFT interface: *Design For Test,* viene utilizzata per effettuare dei test sul core usando due tipi di test *Memory Built-In Self Test* (MBIST) and *Automatic Test Pattern Generation*(ATPG).

CARATTERISTICHE DEL PROCESSORE

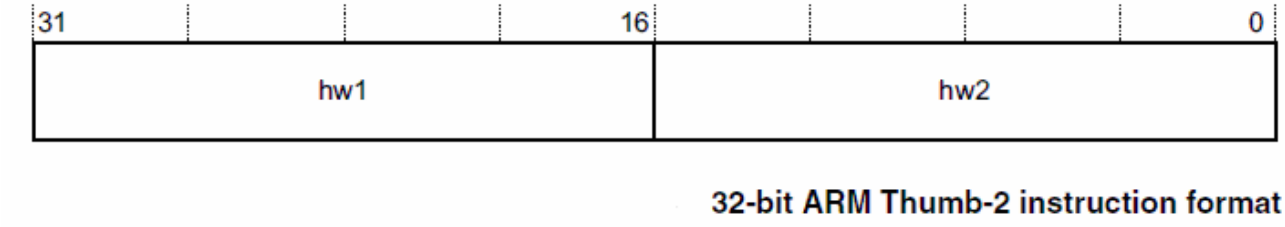
Il processore Cortex A8 implementa l'architettura ARMv7A ed include:
_ *un set d'istruzioni ARM a 32 bit.*
_ *un set d'istruzioni Thumb2 a 16 e 32 bit:*rispetto al set Thumb la maggior parte delle istruzioni Thumb2 sono a 32 bit.La differenza principale rispetto alle istruzioni ARM è che le istruzioni ARM sono principalmente incondizionate, mentre quelle Thumb2 possono essere anche condizionali cioè del tipo IT (if then else).Le istruzioni Thumb 2 sono accessibili solamente quando il processore è nello stato Thumb, ovvero quando il bit T nel registro CPSR è a 1 e il bit J nel registro CPSR è a 0. Oltre alle istruzioni a 32-bit Thumb, ci sono diverse istruzioni a 16-bit Thumb e alcune istruzioni a 32-bit ARM. I miglioramenti principali sono:

1. L'aggiunta di istruzioni a 32 bit alle istruzioni Thumb consente:
   _ di gestire le eccezioni nello stato Thumb;
   _ l'accesso ai Coprocessori;
   _ di includere istruzioni del tipo DSP *Digital Signal Processing;*

_ migliorare le prestazioni nei casi in cui una singola istruzione a 16 bit limita le funzioni a disposizione del compilatore.

2. L'aggiunta di un istruzione a 16 bit consente la gestione delle operazioni di tipo condizionale.
3. L'aggiunta di un istruzione a 16 bit migliora la densità di codice sostituendo sequenza di due istruzioni con una singola istruzione.

Di seguito viene riportato il formato di un istruzione Thumb2 a 32 bit:



32-bit ARM Thumb-2 instruction format

Il primo *halfword* hw1 determina la lunghezza dell' istruzione e le funzionalità.Se il processore vede che l'istruzione è lunga 32 bit, va a prendersi anche il secondo halfword (hw2) leggendo l'indirizzo dell' istruzione.

_ *un set d'istruzioni Thumb EE:* ThumbEE è una variante del set di istruzioni Thumb-2. È stato progettato con l'obiettivo di generare codice dinamicamente. Questo è il codice compilato su un dispositivo poco prima o durante l'esecuzione del bytecode . E 'particolarmente adatto per linguaggi che utilizzano puntatori gestiti da array, e che richiedono la gestione di branch, di subroutine, e l'intercettazioni di puntatori nulli. Il processore entra nello stato ThumbEE quando i bit T e J nel registro CPSR sono entrambi impostati a 1.
Due registri forniscono la configurazione ThumbEE:
_ ThumbEE Configuration Register: Contiene un bit XED che stabilisce o meno l'accesso al registro ThumbEE HandlerBase;
_ ThumbEE HandlerBase Register: che contiene gli indirizzi base degli Handlers Thumb EE.
Di seguito sono riportate le relative configurazioni dei due registri:

ThumbEE Configuration Register

The purpose of the ThumbEE Configuration Register is to control access to the ThumbEE HandlerBase Register.

The ThumbEE Configuration Register is:

- in CP14 register c0
- a 32-bit register, with access rights that depend on the current privilege:
  - the result of an unprivileged write to the register is Undefined
  - unprivileged reads, and privileged reads and writes, are permitted.

Figure 2-2 shows the bit arrangement of the ThumbEE Configuration Register.

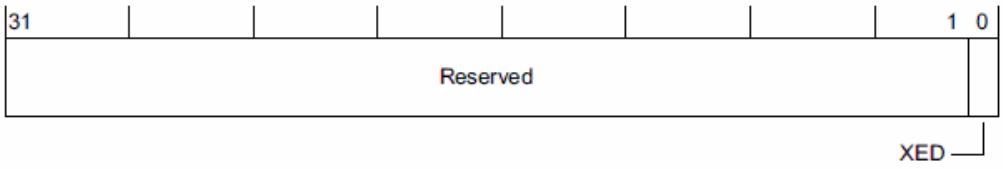


Figure 2-2 ThumbEE Configuration Register format

Table 2-1 shows how the bit values correspond with the ThumbEE Configuration Register.

Table 2-1 ThumbEE Configuration Register bit functions

Bits	Field	Function
[31:1]	-	Reserved. *Unpredictable* (UNP), *Should-Be-Zero* (SBZ).
[0]	XED	eXecution Environment Disable bit. Controls unprivileged access to the ThumbEE HandlerBase Register: 0 = Unprivileged access permitted. 1 = Unprivileged access disabled. The reset value of this bit is 0.

Any change to this register is only guaranteed to be visible to subsequent instructions after the execution of an ISB instruction. However, a read of this register always returns the last value written to the register.

To access the ThumbEE Configuration Register, read or write CP14 with:

```
MRC p14, 6, <Rd>, c0, c0, 0 ; Read ThumbEE Configuration Register
MCR p14, 6, <Rd>, c0, c0, 0 ; Write ThumbEE Configuration Register
```

ThumbEE HandlerBase Register

The purpose of the ThumbEE HandlerBase Register is to hold the base address for ThumbEE handlers.

The ThumbEE HandlerBase Register is:

- in CP14 register c0
- a 32-bit read/write register, with unprivileged access that depends on the value of the ThumbEE Configuration Register.

Figure 2-3 shows the bit arrangement of the ThumbEE HandlerBase Register.

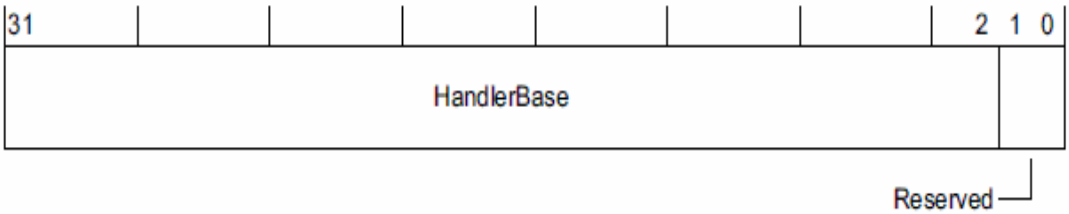


Figure 2-3 ThumbEE HandlerBase Register format

Table 2-2 shows how the bit values correspond with the ThumbEE HandlerBase Register.

Table 2-2 ThumbEE HandlerBase Register bit functions

Bits	Field	Function
[31:2]	HandlerBase	The address of the ThumbEE Handler_00 implementation. This is the address of the first of the ThumbEE handlers. The reset value of this field is Unpredictable.
[1:0]	-	Reserved. UNP, SBZ.

Any change to this register is only guaranteed to be visible to subsequent instructions after the execution of an ISB instruction. However, a read of this register always returns the last value written to the register.

To access the ThumbEE HandlerBase Register, read or write CP14 with:

```
MRC p14, 6, <Rd>, c1, c0, 0 ; Read ThumbEE HandlerBase Register
MCR p14, 6, <Rd>, c1, c0, 0 ; Write ThumbEE HandlerBase Register
```

Access to ThumbEE registers

Table 2-3 shows the access permissions for the ThumbEE registers, and how unprivileged access to the ThumbEE HandlerBase Register depends on the value of the ThumbEE Configuration Register.

Table 2-3 Access to ThumbEE registers

	Unprivileged access		Privileged access
Register	**XED == 0[a]**	**XED == 1[a]**	
ThumbEE Configuration	Read access permitted, write access Undefined	Read access permitted, write access Undefined	Read and write access permitted
ThumbEE HandlerBase	Read and write access permitted	Read and write access Undefined	Read and write access permitted

a. Value of XED bit in the ThumbEE Configuration Register

_ *Jazelle Extension*: Il processore Cortex-A8 fornisce un'implementazione banale del Extension Jazelle. Questo significa che il processore non accelera l'esecuzione di qualsiasi bytecode, e tutti i bytecode sono eseguiti da una routine software. Se fosse stata supportata l'estensione il bit J del CPSR sarebbe stato uguale a 1.
Nell'implementazione:
• lo stato Jazelle non è supportato quindi bit J= 0 nel CPSR;
• L'istruzione BXJ si comporta come una istruzione BX.

Vengono forniti 3 registri per l'implementazione:

Jazelle Identity Register
Consente al software di determinare il tipo di implementazione del estensione jazelle prevista dal processore.
Il registro d'identita Jazelle:
_nel coprocessore di debug CP14 è il registro c0;
_ è un registro a 32 bit di sola lettura, e accessibile da tutti i processori in qualsiasi stato.

Figure 2-4 shows the bit arrangement of the Jazelle Identity Register.

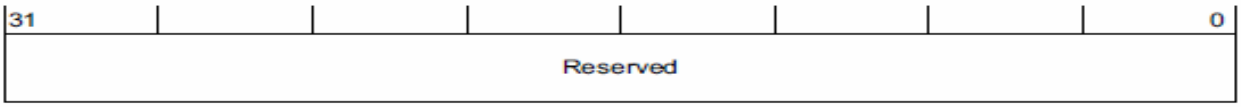


Figure 2-4 Jazelle Identity Register format

Table 2-4 shows how the bit values correspond with the Jazelle Identity Register.

Table 2-4 Jazelle Identity Register bit functions

Bits	Field	Function
[31:0]	-	*Read-As-Zero* (RAZ)

To access this register, read CP14 with:

```
MRC p14, 7, <Rd>, c0, c0, 0 ; Read Jazelle Identity Register
```

Jazelle Main Configuration Register

E' il registro di configurazione principale:
_in CP14 è il registro c0;
_è a 32 bit e l'accesso dipende dai privilegi:

- sola scrittura (WO) in modalità User Mode;
- sola Lettura(RO) in modalità Privileged Mode.

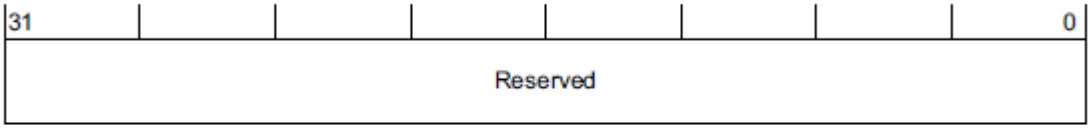


Figure 2-5 Jazelle Main Configuration Register format

Table 2-5 shows how the bit values correspond with the Jazelle Main Configuration Register.

Table 2-5 Jazelle Main Configuration Register bit functions

Bits	Field	Function
[31:0]	-	RAZ

To access this register, read or write CP14 with:

```
MRC p14, 7, <Rd>, c2, c0, 0 ; Read Main Configuration Register
MCR p14, 7, <Rd>, c2, c0, 0 ; Write Main Configuration Register
```

Jazelle OS Control Register

Il Jazelle OS Control Register permette ai sistemi operativi di controllare l'accesso all'estensione hardware Jazelle.
Il Registro Jazelle OS Control è:
• in CP14 è il registro c0.
• un registro a 32-bit, con diritti di accesso, che dipendono dal privilegio corrente:

_Il risultato di un accesso in modalità utente (User mode) è un'eccezione Undefined Instruction ;
_Il registro è di lettura / scrittura (R / W)accessibile solo in modalità privilegiata (Privileged mode).

Figure 2-6 shows the bit arrangement of the Jazelle OS Control Register.

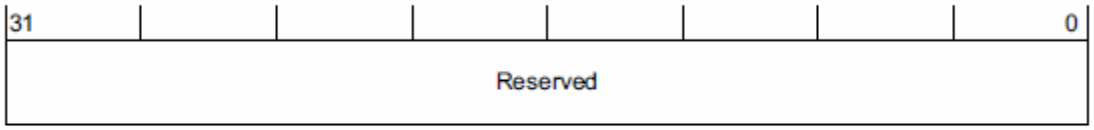


Figure 2-6 Jazelle OS Control Register format

Table 2-6 shows how the bit values correspond with the Jazelle OS Control Register.

Table 2-6 Jazelle OS Control Register bit functions

Bits	Field	Function
[31:0]	-	RAZ

To access this register, read or write CP14 with:

```
MRC p14, 7, <Rd>, c1, c0, 0 ; Read OS Control Register
MCR p14, 7. <Rd>, c1, c0, 0 ; Write OS Control Register
```

_*Security Extensions architecture:* il processore implementa l'estensione di sicurezza TrustZone per facilitare lo sviluppo di applicazioni sicure.Queste Estensioni di sicurezza sono basate su questi principi fondamentali:

- definire una classe di funzionamento di base dalla quale è possibile passare da uno stato Sicuro(Secure)a uno stato non sicuro (Nonsecure) . La maggior parte del codice viene eseguito in stato non sicuro. Solo il codice fidato viene eseguito nello stato sicuro;
- definire una parte della memoria come protetta (Secure memory),alla quale il Core può accedervi solo se è nello stato Secure;
- Entrata nello Stato Secure controllata;
- Nello stato Non Secure non vi è alcuna perdita dei dati;
- L'uscita dallo Stato Secure può avvenire solo in alcuni punti programmati;
- Il debug è rigorosamente controllato;
- Il processore entra nello stato Secure durante il reset.

Nella figura 2.7 sono riportati i relativi stati.
Dalla figura si nota che vi è un monitor che collega i due stati secure e nonsecure, e gestisce il flusso del programma.Il sistema Operativo(Kernel) è diviso in kernel sicuro e kernel non sicuro.
In una operazione normale non sicura, cioè quando sono nello stato non secure, il sistema operativo avvia il task nel modo consueto. Quando il processo utente richiede l'esecuzione sicura fa una richiesta al Secure Kernel che opera nella modalità Privileged mode. Il secure Kernel chiama il monitor chiedendo che

trasferisca l'esecuzione nel Secure State.Questo significa che il punto d'uscita per un processo in Non Secure state è il Monitor.

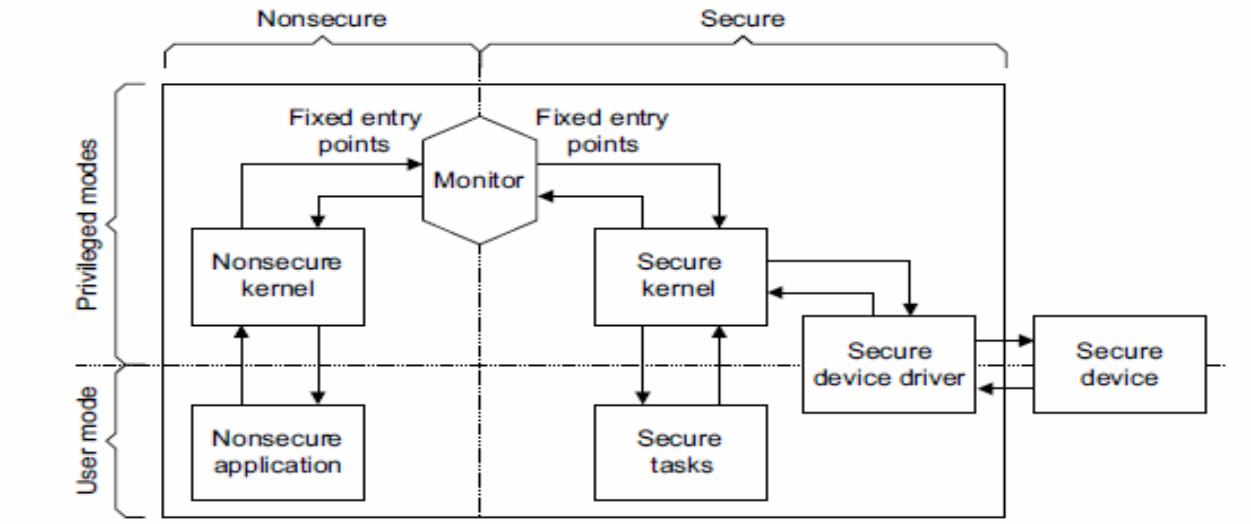
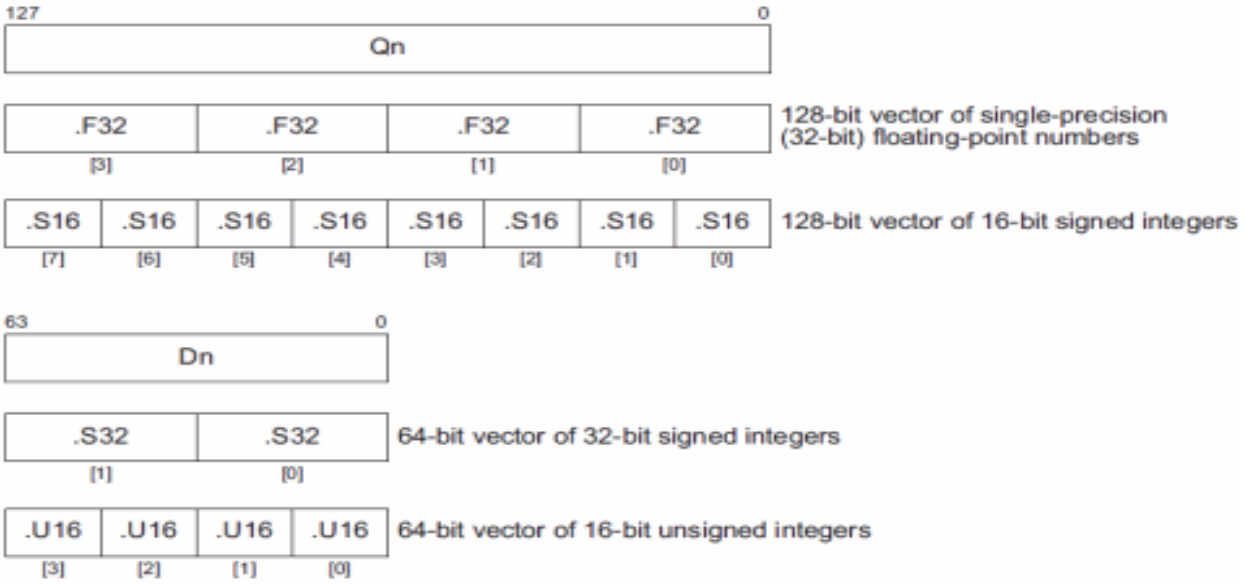


Figure 2-7 Secure and Nonsecure states

_ *Advanced SIMD Architecture*: architettura di elaborazione del segnale che aggiunge nuove istruzioni mirate ai processi di elaborazione dell'audio, video,immagini e grafica 3-D, e di trattamento della parola. Queste istruzioni assieme alle istruzioni VPF e a quelle del coprocessore NEON utilizzano lo stesso registro *registry bank*.Le istruzioni sono eseguiti su vettori in parallelo di elementi dello stesso tipo, e aventi registri da 64 bit o 128 bit. Gli elementi possono essere:

1. a 32 bit per i numeri in virgola mobile(floating point)e a singola precisione;
2. a 8, 16,32,64 bit per gli interi con o senza segno;
3. a 8,16,32,64bit per i bitfields;
4. a 8 o 16 bit per i polinomi.



Examples of Advanced SIMD vectors

_*L'architettura VFP v3* è un miglioramento per l'architettura VFP v2. Le modifiche principali sono:

• il raddoppio del numero di registri a doppia precisione a 32 bit;
• l'introduzione di un'istruzione che pone una costante a virgola mobile in un registro, e istruzioni che consentono di eseguire conversioni tra numeri in virgola fissa e numeri in virgola mobile ;
• l'introduzione di architettura VFP v3 variante, che non trattiene le eccezioni in virgola mobile .

_ *Processor operating states:*

CPSR J and T bit encoding

J	T	Instruction set state
0	0	ARM
0	1	Thumb
1	0	Jazelle
1	1	ThumbEE

La modalità jazelle nel cortex A8 è supportata in modo banale. E' possibile cambiare :

1. Da Arm a Thumb state usando le istruzioni BX e BLX;
2. Da Thumb a ThumbEE state usando le istruzioni ENTERX e LEAVEX;
3. Da ARM a JAZELLE state o da Thumb a Jazelle state usando l'istruzione BXJ.

_ *Tipi di Dati*: doubleword a 64-bit,word a 32-bit(4byte),halfword a 16-bit(2 byte),byte a 8-bit (1 byte).Quando uno di questi tipi di dati è unsigned(senza segno), il valore degli N-bit di dati rappresenta un intero nel range che va da 0- +2N-1. Quando uno di questi tipi di dati è signed(con segno), il valore degli N-bit di dati rappresenta un intero nella gamma-2^(N-1)- +2 (N-1)-1, utilizzando il formato complemento a due .

_ *Rappresentazione dei dati:* Big Endian o Little Endian.

Big Endian: Il byte più significativo di una parola è collocato all'indirizzo più basso:

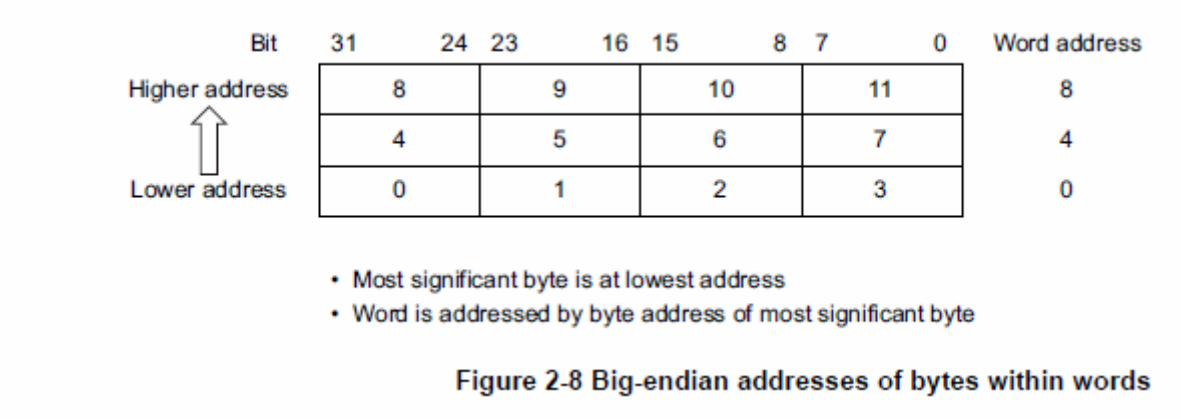


Figure 2-8 Big-endian addresses of bytes within words

Little Endian : il byte meno significativo della parola è collocato all'indirizzo più basso:

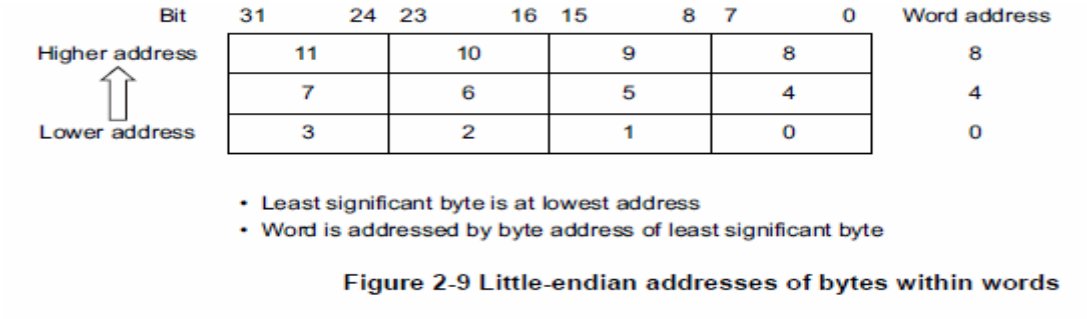


Figure 2-9 Little-endian addresses of bytes within words

_ *Addresses in the processor* **:** Esistono 4 tipi di indirizzi:

- *Virtual Address* (VA)
- *Modified Virtual Address* (MVA)
- *Physical Address* (PA).

Per effettuare la traduzione da VA a PA , il core nello stato Secure affida il compito alla TLB.

Table 2-7 Address types in the processor system

Processor	Caches	TLBs	AXI bus
Virtual Address	Virtual index physical tag[a]	Translates Virtual Address to Physical Address	Physical Address[b]

a. L1 cache is virtual index physical tag.
b. L2 cache is physical address physical tag.

Di seguito è riportato un esempio di manipolazione dell'indirizzo quando il processore richiede un istruzione:

1. Il processore emette il VA dell'istruzione come Secure o Non Secure secondo lo stato del processore.
2. Il VA è tradotto in base all'ID del processo Secure o Non Secure(l'ID è fornito dal coprocessore di sistema CP15 nel registro c13 di CP15) in un MVA e poi in un PA attraverso la TLB.Il TLB esegue la traduzione in parallelo con la cache di ricerca.La traduzione utilizza dei descrittori sicuri se il core è nello stato Secure, altrimenti usa quelli non sicuri in caso contrario.
3. Se il TLB esegue un controllo sulla corretta protezione del MVA, e il tag PA è nella cache istruzioni, i dati dell' istruzione vengono restituiti al processore.
4. il PA è passato alla cache L2. Se la cache L2 contiene l'indirizzo fisico del istruzione richiesto, la cache L2 fornisce i dati dell' istruzione.
5. il PA è passato all' interfaccia bus AXI per eseguire un accesso esterno, in caso di cache miss. L'accesso esterno è sempre Nonsecure quando il core è nel Nonsecure stato. Nello stato protetto, l'accesso esterno è sicuro o non sicuri secondo il valore dell' attributo nel descrittore selezionato.

_ *Registri :* il processore ha 40 registri, di cui 33 registri general purpose a 32 bit , e 7 registri di stato a 32 bit. Questi registri non sono tutti accessibili, allo stesso tempo. Lo stato del processore e le modalità di funzionamento determinano i registri che sono disponibili per il programmatore.In ARM, 16 registri di dati e uno o due registri di stato sono accessibili in qualsiasi momento.Nella figura seguente sono mostrati i registri che sono disponibili per le relative modalità.

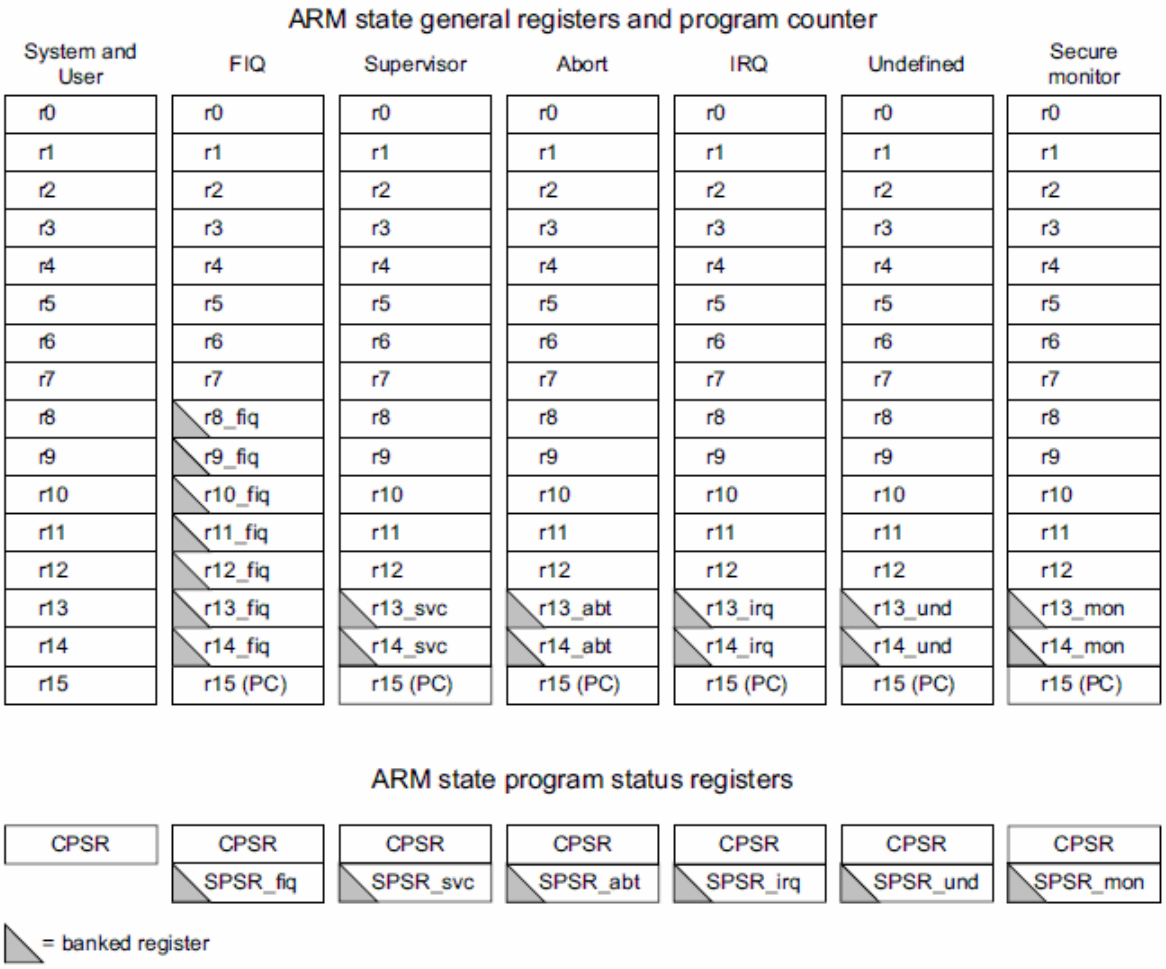


Figure 2-10 Register organization in ARM state

Table 2-9 Register mode identifiers

Mode	Mode identifier
User	usr
Fast interrupt	fiq
Interrupt	irq

Table 2-9 Register mode identifiers (continued)

Mode	Mode identifier
Supervisor	svc
Abort	abt
System	usr
Undefined	und
Monitor	mon

Processor mode			Descrizione	Codifica M[4:0]
1	User	(usr)	Modo d'esecuzione dei programmi comuni.	0b10000
2	FIQ	(fiq)	Gestione di *interrupt* veloce.	0b10001
3	IRQ	(irq)	Gestione di *interrupt* generico.	0b10010
4	Supervisor	(svc)	Modo protetto per l'esecuzione di codice del sistema operativo.	0b10011
5	Abort	(abt)	Errore nell'accesso di memoria (anche per implementare memoria virtuale o protezione della memoria).	0b10111
6	Undefined	(und)	Istruzione illegale.	0b11011
7	System	(sys)	Modo provilegiato d'esecuzione di un *task* del sistema operativo.	0b11111

Tabella - Modi di esecuzione

Tuttavia, le istruzioni a 16-bit offrono solo un accesso limitato ad alcuni dei registri. Nessuna limitazione esiste per le istruzioni a 32 bit Thumb-2 e ThumbEE. I Registri da r0 a r13 sono registri generici utilizzato per contenere dati o valori di indirizzo.Il registro R14 *Link Counter* contiene l'indirizzo di ritorno da una subroutine. Il registro R14 riceve inoltre l'indirizzo di ritorno quando il processore esegue un istruzione *Branch with link*(BL o BLX).Allo stesso modo, i corrispondenti registri bank register(cioè associati ad un modo identificativo) r14_mon , r14_svc, r14_irq, r14_fiq, r14_abt, e r14_und ,hanno un valore di ritorno quando il processore riceve interrupt ed eccezioni, o quando si esegue le istruzioni BL o BLX all'interno di routine di interrupt o eccezione. Il registro R15 invece è quello che contiene il PC(*program counter*)ovvero l'indirizzo successivo dell'istruzione.In Modalità privilegiata si può accedere al registro di stato del programma salvato (SPSR), che contiene the condition code flags(flag di condizione), status bits(bit di stato), and current mode bits(bit di modalità corrente) , quest'ultimo abilitato quando ho un eccezione .
Infine bisogna considerare anche il **Current program status registers CPSR** riportato nella figura seguente**:**

Figure 2-12 shows the bit arrangements of the program status registers.

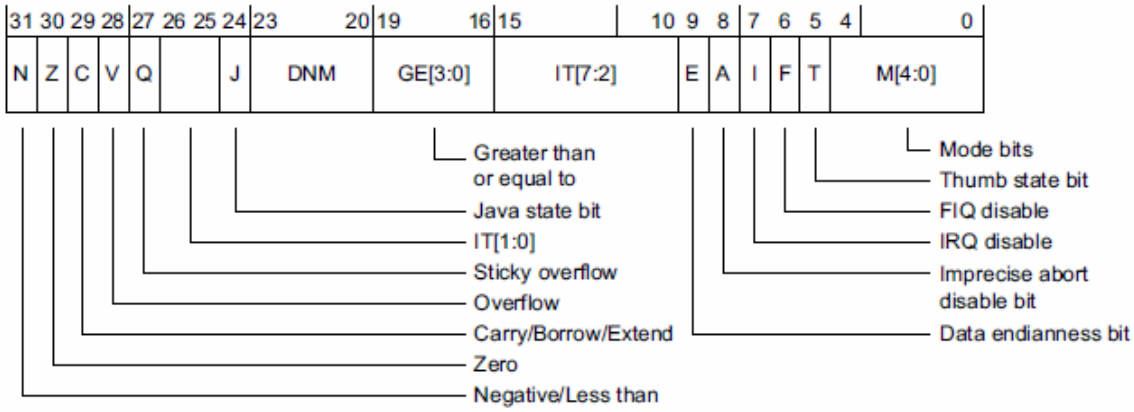


Figure 2-12 Program status register

Note

The bits identified in Figure 2-12 as *Do Not Modify* (DNM) must not be modified by software. These bits are:

- Readable, to enable the processor state to be preserved, for example, during process context switches.
- Writable, to enable the processor state to be restored. To maintain compatibility with future ARM processors, and as good practice, you are strongly advised to use a read-modify-write strategy when you change the CPSR.

M0-M4 definiscono il modo di processore (Vedi Tabella Modi di esecuzione)

F indica lo stato di disabilitazione (valore 1) dei "Fast Interrupt" (FIQ)

I indica lo stato di disabilitazione (valore 1) delle interruzioni "generiche" (IRQ).

V Overflow: bit di condizione *overflow* per l'aritmetica con segno in complemento a due.

C Carry: bit di condizione *carry* per l'aritmetica senza segno e *shift*.

Z Zero: indica il risultato di un'operazione pari a zero.

N Negative: indica il risultato di un'operazione negativo.

Q Sticky Overflow: è impostato a 1 quando eseguo le seguenti istruzioni di moltiplicazione e aritmetiche:

- QADD
- QDADD
- QSUB
- QDSUB

- SMLAD
- SMLAxy
- SMLAWy
- SMLSD
- SMUAD
- SSAT
- SSAT16
- USAT
- USAT16.

IT bits: IT [7:5] contiene b000 quando il blocco IT non è attivo.Codifica il codice di condizione base per il corrente blocco IT , se presente. IT [4:0] codifica il numero di istruzioni che devono essere eseguito sotto condizione, contiene b00000 se il blocco IT non è attivo.

The GE[3:0] bits

Some of the SIMD instructions set GE[3:0] as greater than or equal bits for individual halfwords or bytes of the result, as Table 2-10 shows.

Table 2-10 GE[3:0] settings

Instruction	GE[3] A op B >= C	GE[2] A op B >= C	GE[1] A op B >= C	GE[0] A op B >= C
Signed				
SADD16	[31:16] + [31:16] ≥ 0	[31:16] + [31:16] ≥ 0	[15:0] + [15:0] ≥ 0	[15:0] + [15:0] ≥ 0
SSUB16	[31:16] - [31:16] ≥ 0	[31:16] - [31:16] ≥ 0	[15:0] - [15:0] ≥ 0	[15:0] - [15:0] ≥ 0
SADDSUBX	[31:16] + [15:0] ≥ 0	[31:16] + [15:0] ≥ 0	[15:0] - [31:16] ≥ 0	[15:0] - [31:16] ≥ 0
SSUBADDX	[31:16] - [15:0] ≥ 0	[31:16] - [15:0] ≥ 0	[15:0] + [31:16] ≥ 0	[15:0] + [31:16] ≥ 0
SADD8	[31:24] + [31:24] ≥ 0	[23:16] + [23:16] ≥ 0	[15:8] + [15:8] ≥ 0	[7:0] + [7:0] ≥ 0
SSUB8	[31:24] - [31:24] ≥ 0	[23:16] - [23:16] ≥ 0	[15:8] - [15:8] ≥ 0	[7:0] - [7:0] ≥ 0
Unsigned				
UADD16	[31:16] + [31:16] ≥ 2^{16}	[31:16] + [31:16] ≥ 2^{16}	[15:0] + [15:0] ≥ 2^{16}	[15:0] + [15:0] ≥ 2^{16}
USUB16	[31:16] - [31:16] ≥ 0	[31:16] - [31:16] ≥ 0	[15:0] - [15:0] ≥ 0	[15:0] - [15:0] ≥ 0
UADDSUBX	[31:16] + [15:0] ≥ 2^{16}	[31:16] + [15:0] ≥ 2^{16}	[15:0] - [31:16] ≥ 0	[15:0] - [31:16] ≥ 0
USUBADDX	[31:16] - [15:0] ≥ 0	[31:16] - [15:0] ≥ 0	[15:0] + [31:16] ≥ 2^{16}	[15:0] + [31:16] ≥2^{16}
UADD8	[31:24] + [31:24] ≥ 2^{8}	[23:16] + [23:16] ≥ 2^{8}	[15:8] + [15:8] ≥ 2^{8}	[7:0] + [7:0] ≥ 2^{8}
USUB8	[31:24] - [31:24] ≥ 0	[23:16] - [23:16] ≥ 0	[15:8] - [15:8] ≥ 0	[7:0] - [7:0] ≥ 0

CARATTERISTICHE TECNICHE E PRESTAZIONI

Cortex-A8	
Architecture	**ARMv7-A Cortex**
Dhrystone Performance	2.0 DMIPS / MHz
Multicore	No - Single core only
Debug and Trace	CoreSight DK-A8 (available separately)

	65nm LP process		65nm G+ process	
	Optimized	**Synthesized**	**Optimized**	**Synthesized**
Frequency (MHz)	600	500	1 GHz	750
Frequency conditions	at ss, 1.08v, 125 C		at SS, 0.9v, 125C	
Area with L1 Cache (mm2)	<4	<4	<4	<4
Cache Size (I/D)	32K/32K	32K/32K	32K/32K	32K/32K

Applicazioni D'Uso:
Il processore Cortex-A8 è in grado di soddisfare i requisiti per i dispositivi mobili che necessitano di potenza ottimizzati per il funzionamento in meno di 300mW, e le applicazioni consumer che richiedono prestazioni ottimizzate di 2.000 Dhrystone MIPS.

Applicazioni
SmartPhone, Apple Iphone 4,Apple IPAD,Netbook,Tv Digitale(decoder digitale,decoder satellitare),home networking(router,modem,switch,gateway),Storage Networking(Hard disk),Stampanti.

Modem

Router

SmartPhone

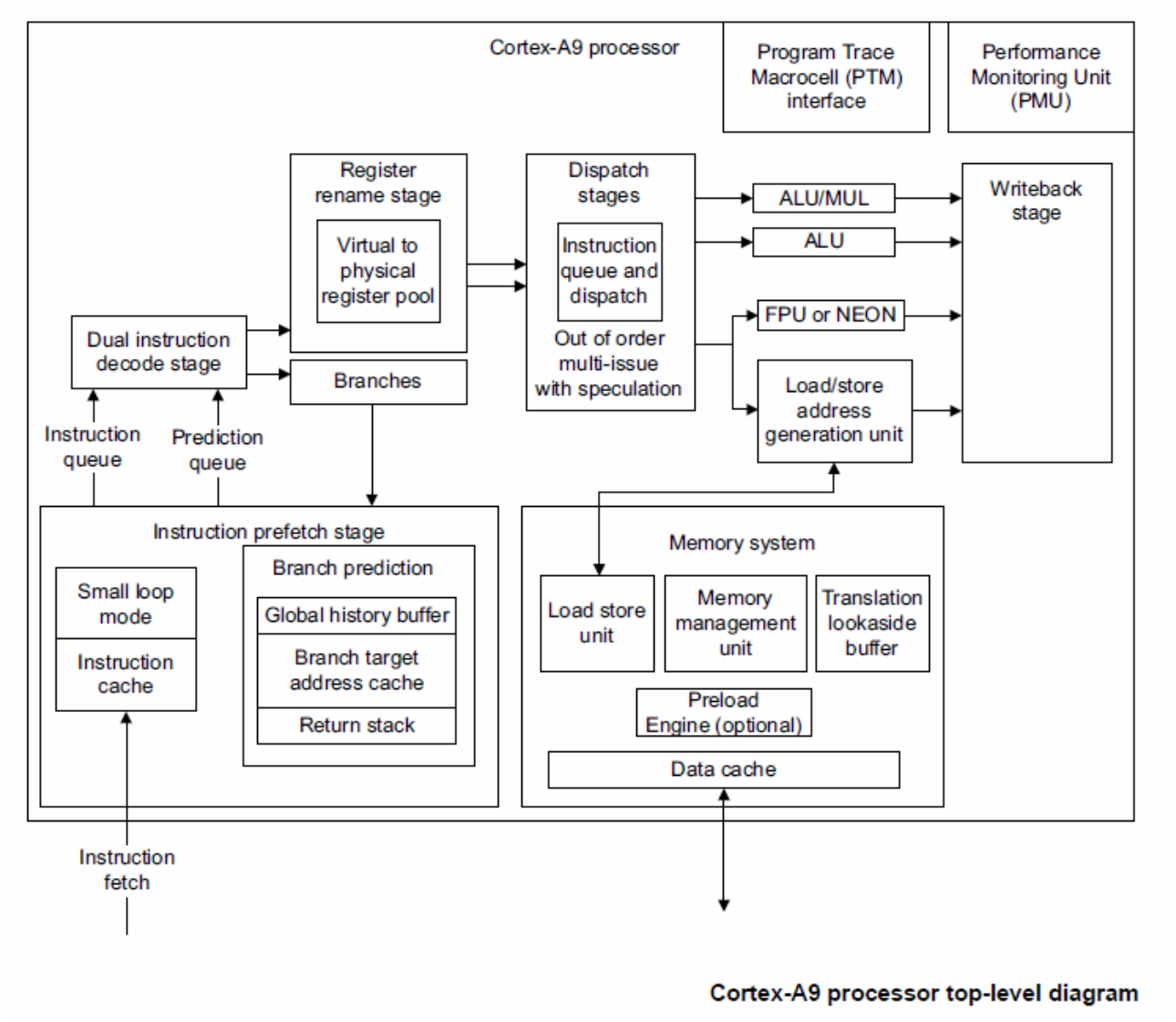
NetBook

Stampante

Apple
Iphone4

PROCESSORE ARM CORTEX A9

Di seguito riporto il diagramma a blocchi del Cortex A9



Cortex-A9 processor top-level diagram

A differenza di tutti gli altri schemi a blocchi troviamo delle nuove unità che descrivo di seguito:
_*Performance Monitoring Unit*: Unità di monitoraggio delle prestazioni del processore.Questa unità dispone di sei contatori che raccolgono le statistiche di funzionamento del processore e della memoria. Ogni contatore può contare qualsiasi dei 58 eventi disponibili nel processore Cortex-A9. I contatori PMU, e i loro registri di controllo associati, sono accessibili dal interfaccia interna del coprocessore di sistema CP15 oppure dalla Debug APB interface.
_ *Register Rename*: questo registro ridenomina durante lo stato d'esecuzione WAR (Write after Read)e WAW(write after write) i registri d'uso comune e i bit dei flag del registro di stato CPSR. Inoltre mappa i 32 registri ARM in un pool di 56 registri fisici a 32 bit. Ridenomina i flag del CPSR(N, Z, C, V, Q, e GE) utilizzando 8 registri a 9 bit ciascuno.
_*Small loop mode*: il processore entra in questa modalità quando deve eseguire una sequenza ciclica e ripetuta di istruzioni. Oltre a questa modalità troviamo la modalità Wait for interupt(il processore rimane in attesa che venga gestita l'eccezione o interruzione),Dormant mode(il processo è inattivo, solo la cache è in funzione),Shutdown mode(il processore sta per essere arrestato), Run mode(è in funzione).
_*Program Trace Macrocell interface*(PTM): questa interfaccia utilizza l'istruzione PFT *program flow trace,* che raccoglie tutti i waypoints ovvero tutti i cambiamenti nel flusso del programma o negli eventi per esempio i branches(i salti) oppure il cambiamento del ID di un processo.

_*Virtualization of Interupts*: Con la virtualizzazione degli interupt è possibile modificare il comportamento di una eccezione velocizzando la gestione degli interupt.Bisogna prima di tutto prendere in considerazione il registro *Virtualization Control Register* che è accessibile solo in modalità privilegiata e stato Secure.

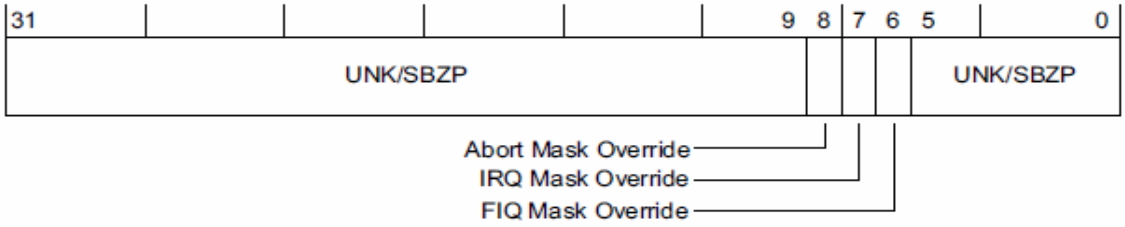


Figure 4-11 VCR bit assignments

Table 4-13 shows the VCR bit assignments.

Table 4-13 VCR bit assignments

Bits	Name	Description
[31:9]	-	UNK/SBZP
[8]	AMO	Abort Mask Override When the processor is in Non-secure state and the SCR.EA bit is set, if the AMO bit is set, this enables an asynchronous Data Abort exception to be taken regardless of the value of the CPSR.A bit. When the processor is in Secure state, or when the SCR.EA bit is not set, the AMO bit is ignored.
[7]	IMO	IRQ Mask Override When the processor is in Non-secure state and the SCR.IRQ bit is set, if the IMO bit is set, this enables an IRQ exception to be taken regardless of the value of the CPSR.I bit. When the processor is in Secure state, or when the SCR.IRQ bit is not set, the IMO bit is ignored.
[6]	IFO	FIQ Mask Override When the processor is in Non-secure state and the SCR.FIQ bit is set, if the IFO bit is set, this enables an FIQ exception to be taken regardless of the value of the CPSR.F bit. When the processor is in Secure state, or when the SCR.FIQ bit is not set, the IFO bit is ignored.
[5:0]	-	UNK/SBZP

To access the VCR, use:

```
MRC p15, 0,<Rd>, c1, c1, 3; Read VCR data
MCR p15, 0,<Rd>, c1, c1, 3; Write VCR data
```

Se il processore è nello stato Secure, quando occorre un eccezione i bit AMO,IMO,IFO del registro VCR sono ignorati. Se il processore è nello stato Non Secure l'eccezione viene sollevata se non sono stati impostati i bit del CPSR E,A,FIQ,e IRQ. Se il bit di CPSR E,A, FIQ,IRQ è impostato, l'eccezione relativa è intrappolato in modalità Monitor. In questo caso, l'eccezione relativa è gestita o meno a seconda dei bit A,I,F in CPSR e AMO,IMO,IFO in VCR.

INTERFACCE

_PTM;

_2 interfacce esterne AMBA AXI da 64bit, una Master0 che fa da bus dei dati, e l'altra Master1 che fa da bus delle istruzioni;
_un interfaccia di DEBUG APB CoreSight;
_un interfaccia DFT.

CARATTERISTICHE DEL PROCESSORE

Implementa l'architettura ARMv7 e possiede le caratteristiche seguenti:
_ Supporta i set d'istruzioni ARM, *Thumb*, *ThumbEE*;
_ La *rappresentazione dei dati* è Big Endian o Little Endian;
_Estensioni di sicurezza *TrustZone*;
_*Memory Management Unit* (MMU) composta da 2 microTLB da 32 entry(una per i dati e una per le istruzioni)e da una MainTLB da 2x32Entry set-associativa a 2 vie;
_ Una unita Opzionale *Preload Engine(PLE)* che trasferisce delle regioni selezionate in memoria verso la memoria L2 secondo una politica FIFO(first input first output).
_ Opzionale è l' accelerazione hardware *Jazelle;*
_ Opzionali: *Media Processing Engine* con tecnologia NEON, e architettura *VFPv3*;
_ *Pipeline Superscalare* di lunghezza variabile con Branch Prediction;
_ *Instruction Cache* e *Data Cache* da 16KB,32KB o 64KB;
_ Opzionale *FPU*(floating point unit);
_ *Registri* : gli stessi del Cortex A8 più il registro VCR(Virtualization Control Register) quindi in totale 41 registri.

CARATTERISTICHE TECNICHE E PRESTAZIONI

Cortex-A9	
Architecture	ARMv7-A Cortex
Dhrystone Performance	2.50 DMIPS/MHz per core
Multicore	1-4 cores Single core version also available
Debug and Trace	CoreSight™ DK-A9 (available separately)

Per analizzare le prestazioni consideriamo i due processori con chip TSMC 65G e TSMC 40G della famiglia Cortex A9:

ARM Cortex-A9 Performance Power & Area			
	Cortex-A9 Single Core Soft Macro Trial Implementation	**Cortex-A9 Dual Core Hard Macro Implementations**	
Process	TSMC 65G	TSMC 40G	
Optimization method	Performance Optimized	Performance Optimized	Power Optimized
Standard Cell Library	ARM SC12	ARM SC12 + High Performance Kit	ARM SC12 + High Performance Kit
Performance (Total DMIPS)	2,075 DMIPS	10,000 DMIPS	4,000 DMIPS
Frequency	830 MHz	2000 MHz (typical)	800 MHz (wc/ss)
Energy Efficiency (DMIPS/mW)	5.2	5.26	8.0
Total power at target frequency	0.4 W	1.9 W	0.5 W
Silicon Area	1.5 mm^2 (excludes caches)	6.7 mm^2 (including L1 parity and all DFT/DFM)	4.6 mm^2 (including all DFT/DFM)

Applicazioni D'Uso:

Il processore ARM Cortex-A9 ™ offre livelli senza precedenti di prestazioni ed efficienza di potenza, è quindi la soluzione ideale per progetti che richiedono elevate prestazioni a basso consumo.

Applicazioni

SmartPhone e cellulari di fascia di mercato medio-alta(blackberry come RIM Playbook),Apple IPAD2,Computer Portatili per esempio NetBook Apple con Windows Mobile CE,Server ad elevate prestazioni,Console di gioco portatili(per esempio PlayStation Portable come Sony PSP2).

Apple IPAD2

RIM Playbook

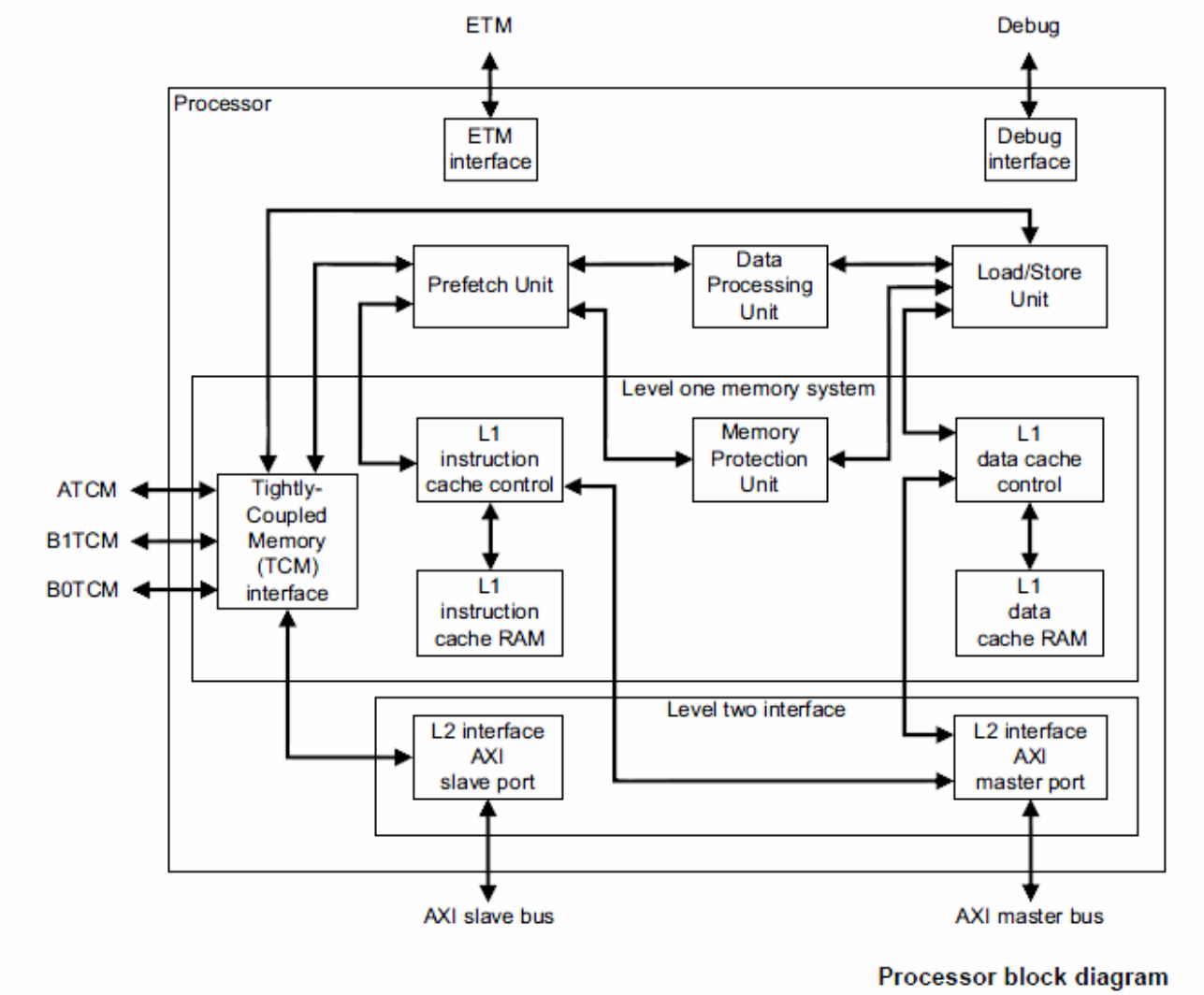
Sony PSP2

4. ARM CORTEX R-SERIES(per applicazioni Real-Time cioè in tempo reale)

_ARM Cortex R4;
_ARM Cortex R5.

PROCESSORE ARM CORTEX R4

I componenti del processore sono illustrati nella figura di seguito.



Processor block diagram

La maggior parte delle unità in figura sono già state incontrate nella famiglia dei Cortex A, quindi di seguito espongo soltanto un breve riassunto. Dall'analisi del diagramma a blocchi si nota che la DPU decodifica ed esegue le istruzioni che gli sono state inviate dalla prefetch Unit PFU. La PFU infatti ha il compito di prelevare le istruzioni dalla cache delle istruzioni L1 e se in tale cache non sono presenti: effettua un accesso alla memoria esterna collegata all' interfaccia Tightly Coupled Memory (TCM) oppure un accesso ad una eventuale seconda memoria esterna tramite l'interfaccia L2 AXI master port. Le istruzioni prelevate dalla PFU sono poi trasferite alla DPU tramite un instruction stream(flusso di istruzioni).La PFU dovrà inserire nell'instruction stream anche la predizione del salto(cioè la DPU dovrà sapere subito a quale indirizzo in memoria dovrà recarsi se incontra un istruzione di salto,ovvero dovrà saperlo prima che decodifichi l'istruzione di salto, in questo modo si evita che la DPU si ritrovi ad effettuare un ciclo(ovvero ad eseguire più volte delle istruzioni)o ad eseguire delle operazioni errate).La DPU può eseguire anche operazioni di trasferimento dei dati da o verso la memoria di sistema L1, per fare questo si affida all'unita load and store LSU che si collega alla L1 data cache control.

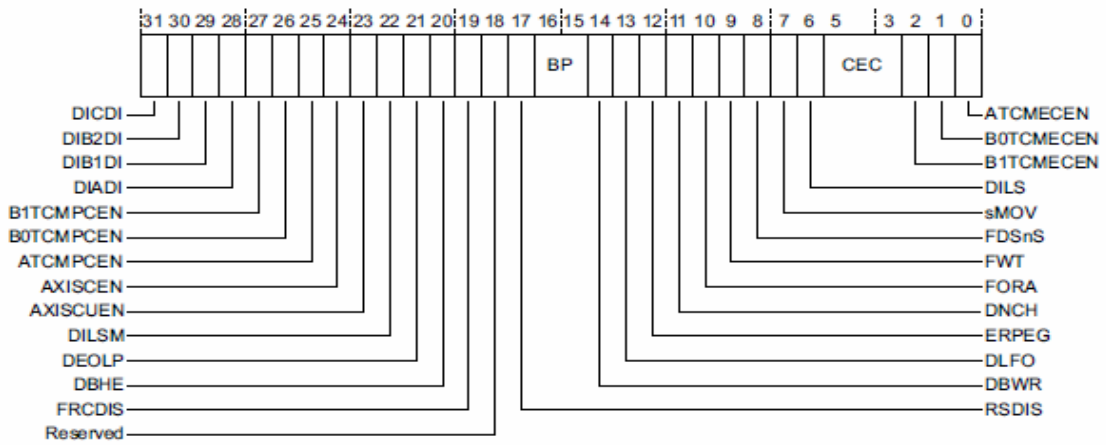
Per quanto riguarda le memorie di sistema troviamo la memoria L1 composta dalla cache delle istruzioni(suddivisa in L1 instruction cache control e L1 instruction cache RAM), dalla cache dei dati (L1 data cache control e L1 data cache RAM), dall'interfaccia TCM e da un'unità di protezione della memoria MPU .La memoria d'interfaccia L2 invece è formata da due interfacce AXI: una porta slave e una master usate per comunicare con le memorie e dispositivi esterni al processore. All'interno della memoria L1 sono presenti due nuove unità che non abbiamo mai incontrato sinora la TCM interface e la MPU che è opportuno analizzare di seguito.

TCM Interface

L'interfaccia TCM è utile quando bisogna reperire dei dati da memorie esterne o scrivere i dati su queste memorie .I dispositivi che utilizzano questa interfaccia sono la PFU, la LSU e l'interfaccia AXI.Questa interfaccia si connette solo a memorie esterne del tipo *Tightly-Coupled Memory* (TCM) attraverso due interfacce (ATCM e BTCM).L'interfaccia ATCM ha 1 sola porta mentre l'interfaccia BTCM ha 2 porte di comunicazione.Le memorie TCM connesse a queste interfacce possono essere delle Memorie RAM o ROM. L'interfaccia ATCM contiene il codice per la gestione delle interruzioni o delle eccezioni che possono essere sollevate in seguito ad un errore. L'interfaccia BTCM invece contiene i blocchi di dati che devono essere elaborati o trasferiti. Su queste interfacce possono essere utilizzati anche dei meccanismi di rilevamento e correzione degli errori che si possono presentare nella lettura di dati o istruzioni nelle memorie TCM. In particolare possono essere implementati meccanismi di rilevazione degli errori (su un bit)attraverso lo schema *parity*, e di correzione dell'errore ECC(Error Correction Code) a 32 o 64bit su uno o due bit d'errore.

Error detection and correction schemes

Il controllo di parità dell'errore *parity error checking* e l'ECC sono possibili solamente se sono stati abilitati i bit del registro di controllo ausiliario *Auxiliary Control Register* (riportato di seguito)B1TCMPCEN, B1TCMECEN se l'errore è sull'interfaccia B1TCM,B0TCMPCEN, B0TCMECEN se l'errore è sull' interfaccia B0TCM , e ATCMPCEN, ATCMECEN se l'errore è sul interfaccia ATCM .



Auxiliary Control Register format

Auxiliary Control Registers

The Auxiliary Control Registers control:

- branch prediction
- performance features
- error and parity logic.

c1, Auxiliary Control Register

The Auxiliary Control Register is:

- a read/write register
- accessible in Privileged mode only.

To access the Auxiliary Control Register, read or write CP15 with:

```
MRC p15, 0, <Rd>, c1, c0, 1 ; Read Auxiliary Control Register
MCR p15, 0, <Rd>, c1, c0, 1 ; Write Auxiliary Control Register
```

Auxiliary Control Register bit functions

Bits	Field	Function
[31]	DICDI[a]	Case C dual issue control: 0 = Enabled. This is the reset value. 1 = Disabled.
[30]	DIB2DI[a]	Case B2 dual issue control: 0 = Enabled. This is the reset value. 1 = Disabled.
[29]	DIB1DI[a]	Case B1 dual issue control: 0 = Enabled. This is the reset value. 1 = Disabled.
[2]	B1TCMECEN	B1TCM external error enable: 0 = Disabled 1 = Enabled. The primary input **ERRENRAM[2]** defines the reset value.
[1]	B0TCMECEN	B0TCM external error enable: 0 = Disabled 1 = Enabled. The primary input **ERRENRAM[1]** defines the reset value.
[0]	ATCMECEN	ATCM external error enable: 0 = Disabled 1 = Enabled. The primary input **ERRENRAM[0]** defines the reset value.

Bits	Field	Function
[28]	DIADI[a]	Case A dual issue control: 0 = Enabled. This is the reset value. 1 = Disabled.
[27]	B1TCMPCEN	B1TCM parity or ECC check enable: 0 = Disabled 1 = Enabled. The primary input **PARECCENRAM[2]**[b] defines the reset value. If the BTCM is configured with ECC, you must always set this bit to the same value as **B0TCMPCEN**.
[26]	B0TCMPCEN	B0TCM parity or ECC check enable: 0 = Disabled 1 = Enabled. The primary input **PARECCENRAM[1]**[b] defines the reset value. If the BTCM is configured with ECC, you must always set this bit to the same value as **B1TCMPCEN**.
[25]	ATCMPCEN	ATCM parity or ECC check enable: 0 = Disabled 1 = Enabled. The primary input **PARECCENRAM[0]**[b] defines the reset value.
[24]	AXISCEN	AXI slave cache RAM access enable: 0 = Disabled. This is the reset value. 1 = Enabled. **Note** When AXI slave cache access is enabled, the caches are disabled and the processor cannot run any cache maintenance operations. If the processor attempts a cache maintenance operation, an Undefined instruction exception is taken.
[23]	AXISCUEN	AXI slave cache RAM non-privileged access enable: 0 = Disabled. This is the reset value. 1 = Enabled.
[22]	DILSM	Disable *Low Interrupt Latency* (LIL) on load/store multiples: 0 = Enable LIL on load/store multiples. This is the reset value. 1 = Disable LIL on all load/store multiples.
[21]	DEOLP	Disable end of loop prediction: 0 = Enable loop prediction. This is the reset value. 1 = Disable loop prediction.
[20]	DBHE	Disable *Branch History* (BH) extension: 0 = Enable the extension. This is the reset value. 1 = Disable the extension.
[19]	FRCDIS	Fetch rate control disable: 0 = Normal fetch rate control operation. This is the reset value. 1 = Fetch rate control disabled.

Bits	Field	Function
[17]	RSDIS	Return stack disable: 0 = Normal return stack operation. This is the reset value. 1 = Return stack disabled.
[16:15]	BP	This field controls the branch prediction policy: b00 = Normal operation. This is the reset value. b01 = Branch always taken. b10 = Branch always not taken. b11 = Reserved. Behavior is Unpredictable if this field is set to b11.
[14]	DBWR	Disable write burst in the AXI master: 0 = Normal operation. This is the reset value. 1 = Disable write burst optimization.
[13]	DLFO	Disable linefill optimization in the AXI master: 0 = Normal operation. This is the reset value. 1 = Limits the number of outstanding data linefills to two.
[12]	ERPEG<	Enable random parity error generation: 0 = Random parity error generation disabled. This is the reset value. 1 = Enable random parity error generation in the cache RAMs. Note This bit controls error generation logic during system validation. A synthesized ASIC typically does not have such models and this bit is therefore redundant for ASICs.
[11]	DNCH	Disable data forwarding for Non-cacheable accesses in the AXI master: 0 = Normal operation. This is the reset value. 1 = Disable data forwarding for Non-cacheable accesses.
[10]	FORA	Force outer read allocate (ORA) for outer write allocate (OWA) regions: 0 = No forcing of ORA. This is the reset value. 1 = ORA forced for OWA regions.
[9]	FWT	Force write-through (WT) for write-back (WB) regions: 0 = No forcing of WT. This is the reset value. 1 = WT forced for WB regions.
[8]	FDSnS	Force D-side to not-shared when MPU is off: 0 = Normal operation. This is the reset value. 1 = D-side normal Non-cacheable forced to Non-shared when MPU is off.
[7]	sMOV	sMOV of a divide does not complete out of order. No other instruction is issued until the divide is finished. 0 = Normal operation. This is the reset value. 1 = sMOV out of order disabled.
[6]	DILS	Disable low interrupt latency on all load/store instructions. 0 = Enable LIL on all load/store instructions. This is the reset value. 1 = Disable LIL on all load/store instructions.
[5:3]	CEC	Cache error control for cache parity and ECC errors. The reset value is b100.

I bit per il controllo e la correzione degli errori possono essere abilitati anche quando il processore è in modalità Reset attraverso il comando PARECCENRAM.

MPU(Memory Protection Unit)

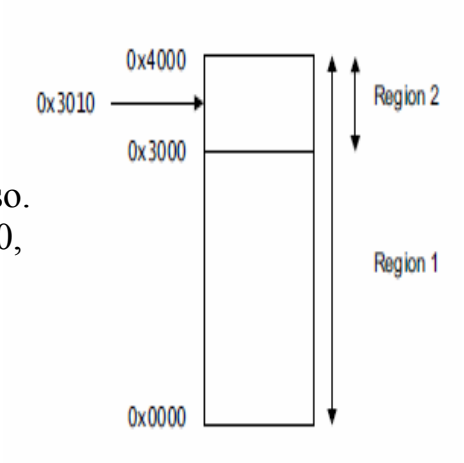
L'unità di protezione della memoria lavora con la memoria L1 per controllarne gli accessi a quest'ultima. Può essere partizionata in 0 , 8 o 12 regioni.Se la memoria ha 0 regioni non può essere riprogrammata. Ogni regione possiede degli attributi, che stabiliscono quali periferiche e memorie esterne possono accedere alla regione.
Gli attributi sono :

Memory type attribute	Shared or Non-shared	Other attributes	Description
Strongly Ordered	-	-	All memory accesses to Strongly Ordered memory occur in program order. All Strongly Ordered accesses are assumed to be shared.
Device	Shared	-	For memory-mapped peripherals that several processors share.
	Non-shared	-	For memory-mapped peripherals that only a single processor uses.
Normal	Shared	Non-cacheable Write-through Cacheable Write-back Cacheable	For normal memory that is shared between several processors.
	Non-shared	Non-cacheable Write-through Cacheable Write-back Cacheable	For normal memory that only a single processor uses.

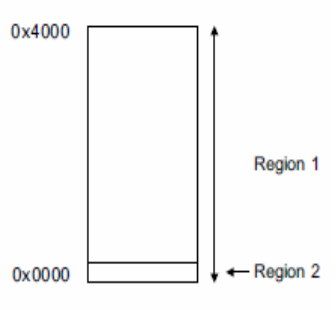
Se una regione ha degli attributi device-shared vuol dire che potranno accedervi soltanto le periferiche che hanno più processori condivisi. Se l'attributo è device-non shared possono accedervi solo le periferiche che hanno un solo processore.Se l'attributo è normal-shared può accedervi una memoria condivisa da più processori, mentre normal non shared solo le memorie con un solo processore. Ogni regione stabilisce anche delle autorizzazioni d'accesso per le modalità User Mode e Privileged mode: no access, read-only access, or read/write access. In una MPU le regioni possono essere anche sovrapposte tra loro.Ogni regione ha una lunghezza che varia dai 32 bytes fino ai 4GB.Di seguito riporto un esempio di possibile organizzazione della memoria MPU:

_ Regione 2 : dimensione di 4KB,indirizzo iniziale 0x3000, Privileged mode hanno pieno accesso , User Mode hanno solo l'accesso Read-only.

_ Regione 1: dimensione 16KB, indirizzo iniziale 0X0000,Privileged e User Mode hanno pieno accesso. Se un User Mode vuole scrivere all'indirizzo 0x3010, questo provoca un interruzione.



Qui a destra riporto invece delle regioni
Sovrapposte, dove la Regione 2 ha una dimensione di
Soli 32 bytes e corrisponde all'indirizzo 0x0000.



CARATTERISTICHE DEL PROCESSORE

_il processore implementa l'*architettura* ARMv7-R;
_*Set d'istruzioni* Arm a 32 bit,Thumb a 16 bit e Thumb2 a 32 bit;
_ *Modi d'esecuzione* :USR,FIQ,IRQ,SVC,ABT,SYS,UND.A differenza dei Cortex A non c'è la modalità Monitor;
_*Tipi di dati*:byte(8bit), halfword(16 bit),word(32 bit),doubleword(64 bit);
_*Rappresentazione dei dati*: little endian o big endian;
_ *Registri:*31 registri comuni(general purpose) a 32 bit: R0-R15 più R8_fiq , R9_fiq, R10_fiq , R11_fiq, R12_fiq, R13_fiq ,R14_fiq ,R13_svc,R14_svc,R13_abt , R14_abt, R13_irq, R14 _irq ,R13_und,R14_und.
6 registri di stato a 32 bit: CPSR , SPSR_fiq , SPSR_svc ,SPSR_abt ,SPSR_irq , SPSR_und.
In totale ci sono 37 registri;
_*Jazelle Extension* con i soliti 3 registri visti per i processori Cortex A ovvero:Jazelle ID,Jazelle main,e jazelle OS.

Jazelle register instruction summary

Register	Instruction	Response
Jazelle ID	`MRC p14, 7, <Rd>, c0, c0, 0`	Read as zero
	`MCR p14, 7, <Rd>, c0, c0, 0`	Ignore writes
Jazelle main configuration	`MRC p14, 7, <Rd>, c2, c0, 0`	Read as zero
	`MCR p14, 7, <Rd>, c2, c0, 0`	Ignore writes
Jazelle OS control	`MRC p14, 7, <Rd>, c1, c0, 0`	Read as zero
	`MCR p14, 7, <Rd>, c1, c0, 0`	Ignore writes

_*Floating point unit* opzionale;
_*Branch prediction* nella PFU con un global history buffer e uno stack di ritorno *return stack* a 4 entry;
_ *una memoria d'interfaccia L2* composta da una interfaccia AXI di tipo master a 64 bit, e da una interfaccia AXI di tipo slave a 64 bit;
_un *interfaccia di DEBUG DAP*, e un *interfaccia ETM*;
_ un unità *Performance Monitoring Unit* (PMU);
_una porta *Vectored Interrupt Controller* (VIC) ;
_Memoria di sistema *Harvard Level one (L1)* composta da:

1. Instruction cache e data cache di dimensione compresa dai 4 ai 64KB;
2. Un interfaccia TCM con dimensione massima di 8 MB con supporto alla correzione dell'errore e rilevamento d'errore di parità;
3. MPU con 8 o 12 regioni ognuna da 32 Bytes.

_*pipeline* a 8 stadi.

CARATTERISTICHE TECNICHE E PRESTAZIONI

Il processore Cortex ™-R4 è il primo processore profondamente embedded real-time cioè sviluppato per applicazioni in tempo reale embedded. Questo processore è ampiamente utilizzato dall'azienda statunitense **Broadcom Corporation** operante nel settore dei semiconduttori, nei circuiti integrati e nelle reti di telecomunicazione.È usato in grandi quantità anche nelle applicazioni come hard disk, prodotti di consumo e unità elettroniche di controllo per i sistemi automobilistici(automotive systems),dispositivi wireless e connessioni LAN Ethernet ed applicazioni senza-fili, quali WLAN, Bluetooth, GPS.

PRESTAZIONI

Per analizzare le prestazioni dei Cortex R4 prendiamo in considerazione due configurazioni:

1. Configurazione con Instruction Cache e Data cache di 8KB,3 porte TCM,MPU con 8 regioni,nessuna FPU,parity checking e ECC sulla memoria L1 e sulle interfacce AXI,debug con 1 watchpoint e due breakpoints;
2. Configurazione con 8KB di Instruction cache e Data cache, nessuna porta TCM,bus AXI slave,MPU,FPU,ECC e parità checking e debug con 1 watchpoint e 1 breakpoints.

Implementation Target	Performance optimized[1]	Power optimized[2]	Area optimized[2]
Process technology	65 nm GP	65 nm LP	65 nm GP
Standard cell library	Artisan™ SC10	Artisan SC10	Artisan SC10
Clock frequency	620 MHz[3]	270 MHz[4]	380 MHz[4]
Performance	1,030 DMIPS	450 DMIPS	630 DMIPS
Core dynamic power[5]	0.12 mW/MHz	0.17 mW/MHz[6]	0.09 mW/MHz
Core leakage power[5]	4.4 mW	0.02 mW	1.4 mW
Core layout area[5]	0.8 sq mm	0.5 sq mm	0.4 sq mm
Core efficiency	13.8 DMIPS/mW	9.8 DMIPS/mW	18.4 DMIPS/mW

Applicazioni D'Uso

- Nei **dispositivi automobilistici**(auto e camion) con riferimento alle unita di controllo elettroniche *electronic control units* ECUs(airbag,controllo di stabilità della vettura,servosterzo,antibloccaggio in frenata,dispositivi di gestione del motore, sistemi di controllo di azionamento del motore, sistemi informatici).
- Supporti di **Storage** come gli hard disk.
- **applicazioni di comunicazione digitale di dati**: modem,applicazioni senza fili quali WLAN,Bluetooth, GPS.
- **Cellulari** : 3G e 4G(tecnologie di terza e quarta generazione che consentono il trasferimento sia di dati "voce" (telefonate digitali) che di dati "non-voce" (ad esempio, *download* da internet, invio e ricezione di email,instant messaging e videochiamata)), WiMax smartphones.
- **Home**: Tv via cavo, Lettori Blu Ray e lettori multimediali portatili per esempio lettori mp3.

ARM CORTEX R5

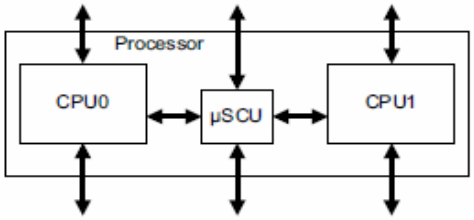
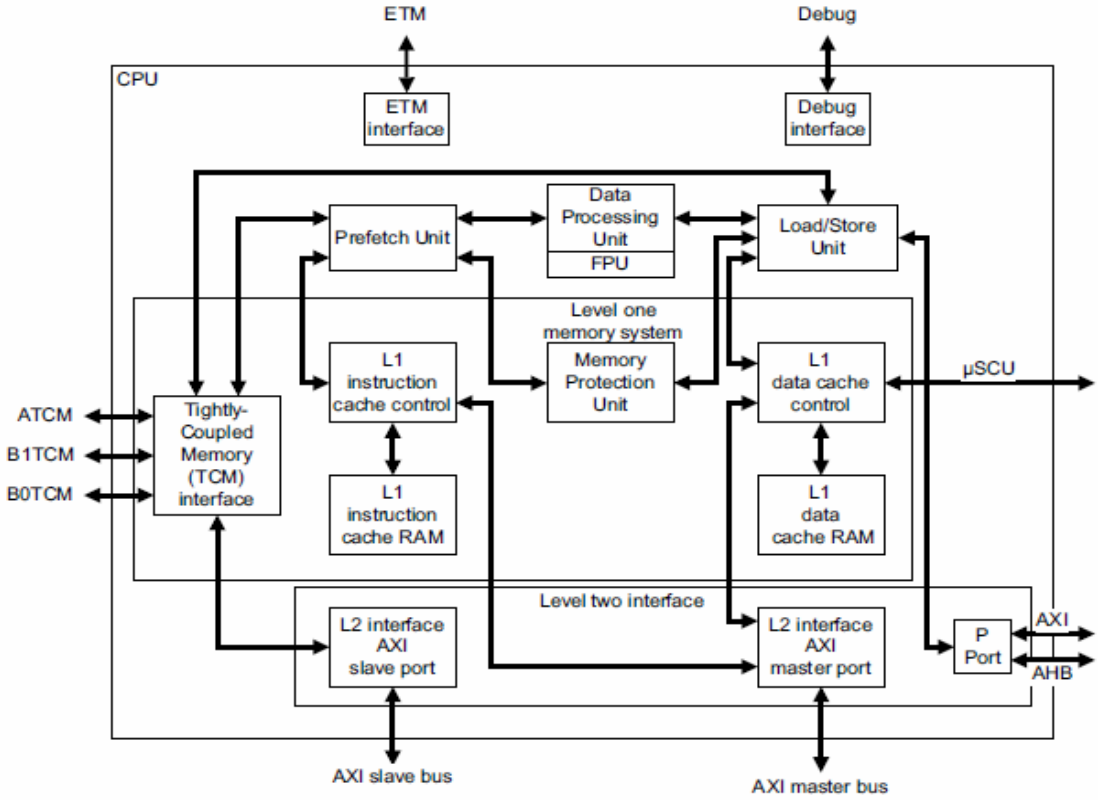


Figure 2-1 Processor block diagram



CPU block diagram

Il diagramma a blocchi è simile a quello del processore Cortex R4, si notano però alcune importanti novità:

_ L'unità load and store LSU può connettersi direttamente alla memoria L2 attraverso la porta P port.

_La cache dei dati L1 si collega con la µSCU per effettuare la manutenzione della cache.

_ la porta *P port* è usata per comunicare con un secondo processore o con dei dispositivi o periferiche esterne.Questa porta è suddivisa a sua volta in due porte: AXI e AHB, in cui sono memorizzati i dati e gli indirizzi in entrata/uscita al processore.Le porte AXI e AHB dispongono ognuna di un buffer dei dati e di un buffer degli indirizzi. Ogni campo nel buffer dei dati contiene 32 bit di dati , e ogni campo nel buffer degli indirizzi contiene 32 bit di indirizzo.Gli accessi a queste porte AXI e AHB sono controllati dalla MPU che consente accessi del tipo: Strongly-ordered e Device per le periferiche, e normal per le memorie.

_*Dual-redundant core(Dual Core):* Il processore può essere implementato con una seconda copia dello stesso processore(core). Quindi due core collegati tra di loro.Esistono due modalità di configurazione :

- **Redundant CPU :**Il secondo core condivide le memorie cache RAM (cache RAMs)e i pin d'ingresso(input pins) del primo core, in questo modo gli accessi avvengono su un'unica memoria cache .Il secondo core comunica con il primo core attraverso i segnali d'ingresso DCCMINP [07:00] e DCCMINP2 [07:00] e i segnali d'uscita DCCMOUT[7:0] and DCCMOUT2[7:0]. Ognuno dei due core può avere una porta d'interfaccia Accelerator Coherency port ACP.In questa configurazione è come se ci fosse un'unica CPU condivisa.
- **Twin-CPU:** in questa configurazione ci sono due CPU operative ed ogni CPU ha le proprie Memorie Cache RAM, le proprie interfacce , e le proprie unità di DEBUG. Vi è però una sola porta ACP(che può essere opzionale).In questa configurazione l'unica memoria condivisa è la memoria L2. Si potrebbe sollevare il problema della coerenza dei dati *coherency*.Per analizzare questo problema consideriamo prima come avviene la manutenzione della cache in una CPU.

  Manutenzione della cache

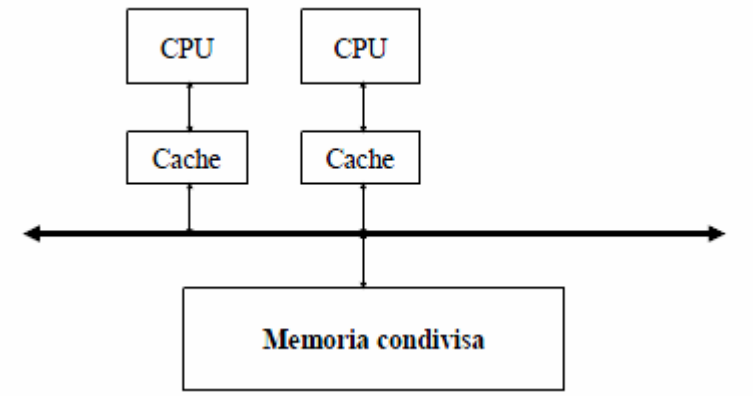
  La CPU ha normalmente un canale di connessione diretto con la memoria principale; questo permette 2 possibili meccanismi di aggiornamento della memoria principale:

  • *write-back:* Per ogni blocco nella cache viene tenuto aggiornato un flag *(dirty bit*), che ricorda se il blocco è stato modificato da quando è stato caricato nella cache.

  Quando un blocco viene eliminato dalla cache ed il dirty bit è settato, il blocco viene copiato dalla cache nella memoria principale .

  Svantaggi:nei sistemi multiprocessore si può avere inconsistenza tra le cache di diversi processori.

  • *write-through:* Ogni volta che la CPU esegue un'operazione di scrittura, la esegue sia sul dato nella cache che in quello nella memoria principale .La perdita di efficienza che ne deriva è limitata dal fatto che le operazioni di scrittura sono di solito molto meno numerose di quelle di lettura.



Nello schema la memoria condivisa è la memoria principale.

Ora facciamo un esempio per chiarire il problema della coerenza in una configurazione Twin-CPU: supponiamo un configurazione con due cpu A e B che adottano la manutenzione di tipo write-back.
A modifica un dato nella memoria cache L1 , ma nella memoria principale e condivisa la tecnologia Write-back
fa si che il nuovo dato risiede solo nella cache L1 . Di conseguenza se B va a leggere il dalla memoria principale
trova il vecchio valore e non il nuovo(problema di coerenza e inconsistenza dei dati).
Una soluzione a questo problema è quella di ricorrere ad una tecnologia di manutenzione write-through, in questo modo il dato è modificato anche nella memoria principale e non vi sono quindi problemi di coerenza.La coerenza dei dati non avviene inoltre nella configurazione *Redundant CPU* in quanto essendo la memoria cache condivisa con il secondo core, non vi sono problemi di inconsistenza e coerenza dei dati.

INTERFACCE

Il processore ha le seguenti interfacce per l'accesso all'esterno:
_ un Interfaccia AXI di tipo Master una di tipo Slave;
_ le Interfacce delle periferiche *Peripheral interfaces:* ovvero la porta P con le sue porte AXI e AHB;
_L'interfaccia TCM;
_l'interfaccia *Accelerator Coherency port* ACP;
_Interfaccia VIC;
_Interfaccia di Debug APB;
_Interfaccia ETM.

CARATTERISTICHE DEL PROCESSORE

Il processore implementa l'*architettura* ARMv7-R ed ha le seguenti caratteristiche:
_ *Set di istruzioni* ARM a 32 bit, Thumb a 16bit, e Thumb2 a 32 bit;
_*Modi d'esecuzione*: *User* (USR),*Fast Interupt* (FIQ),*Interupt* (IRQ),*Supervisor* (SVC),*Abort* (ABT), *System* (SYS),*Undefined* (UND);
_*Rappresentazione dei dati:* Big Endian o Little Endian;
_ *Tipi di dati*:byte(8bit), halfword(16 bit),word(32 bit),doubleword(64 bit);
_ *Registri:* 31 registri comuni(general purpose) a 32 bit: R0-R15 più R8_fiq , R9_fiq, R10_fiq
,R11_fiq,R12_fiq,R13_fiq,R14_fiq,R13_svc,R14_svc,R13_abt,R14_abt,R13_irq, R14 _irq ,R13_und,R14_und.
6 registri di stato a 32 bit: CPSR , SPSR_fiq , SPSR_svc ,SPSR_abt ,SPSR_irq , SPSR_und.
In totale ci sono 37 registri;
_ *Jazelle Extension* stessa implementazione vista nel processore Cortex R4;

_ *Floating point unit* opzionale;
_*Branch prediction* nella PFU con un global history buffer e uno stack di ritorno *return stack* a 4 entry;
_*interfaccia di memoria L2* composta:

_da una interfaccia AXI3 di tipo master a 64 bit, e da una interfaccia AXI3 di tipo slave a 64 bit;

_Un interfaccia AXI3 e AHB entrambe di tipo master a 32 bit collegate alla porta P;

_un interfaccia di DEBUG DAP, e un interfaccia ETM;

_ un unità *Performance Monitoring Unit* (PMU);

_una porta *Vectored Interrupt Controller* (VIC) .

_Memoria di sistema *Harvard Level one (L1)* composta:

1. Instruction cache e data cache di dimensione compresa dai 4 ai 64KB;
2. Un interfaccia TCM con dimensione massima di 8 MB con supporto alla correzione dell'errore e rilevamento d'errore di parità;
3. MPU con dimensione fino a 32 Bytes.

CARATTERISTICHE TECNICHE E PRESTAZIONI

Il processore Cortex-R5 espande le caratteristiche del processore Cortex-R4 per consentire livelli più elevati di prestazioni del sistema, una maggiore efficienza e affidabilità, e una gestione degli errori migliore in sistemi real-time.

Per analizzare le prestazioni dei Cortex R5 prendiamo in considerazione le due configurazioni viste per il processore Cortex R4 implementate nel processore in esame:

1. Configurazione con Instruction Cache e Data cache di 8KB,3 porte TCM,MPU con 8 regioni,nessuna FPU,parity checking e ECC sulla memoria L1 e sulle interfacce AXI,debug con 1 watchpoint e due breakpoints;
2. Configurazione con 8KB di Instruction cache e Data cache, nessuna porta TCM,bus AXI slave,MPU,FPU,ECC e parità checking e debug con 1 watchpoint e 1 breakpoint.

Implementation Target[1]	Performance optimized[1]	Power optimized[2]	Area optimized[2]
Process technology	65 nm GP	65 nm LP	65 nm GP
Standard cell library	Artisan™ SC10	Artisan SC10	Artisan SC10
Clock frequency	620 MHz[3]	270 MHz[4]	380 MHz[4]
Performance	1,030 DMIPS	450 DMIPS	630 DMIPS
Core dynamic power[5]	0.13 mW/MHz	0.18 mW/MHz[6]	0.09 mW/MHz
Core leakage power[5]	4.4 mW	0.02 mW	1.4 mW
Core layout area[5]	0.8 sq mm	0.5 sq mm	0.4 sq mm
Core efficiency	13.6 DMIPS/mW	9.7 DMIPS/mW	18.2 DMIPS/mW

Osservando le prestazioni si nota che i valori della Performance, del Core efficiency e della frequenza di clock sono gli stessi del Processore Cortex R4. La differenza principale tra il Cortex R4 e R5, sta nel fatto che il processore R5 si può presentare in versione Dual Core. Sicuramente le prestazioni di un Cortex R5 Dual Core sono nettamente migliori di quelle del Cortex R4.

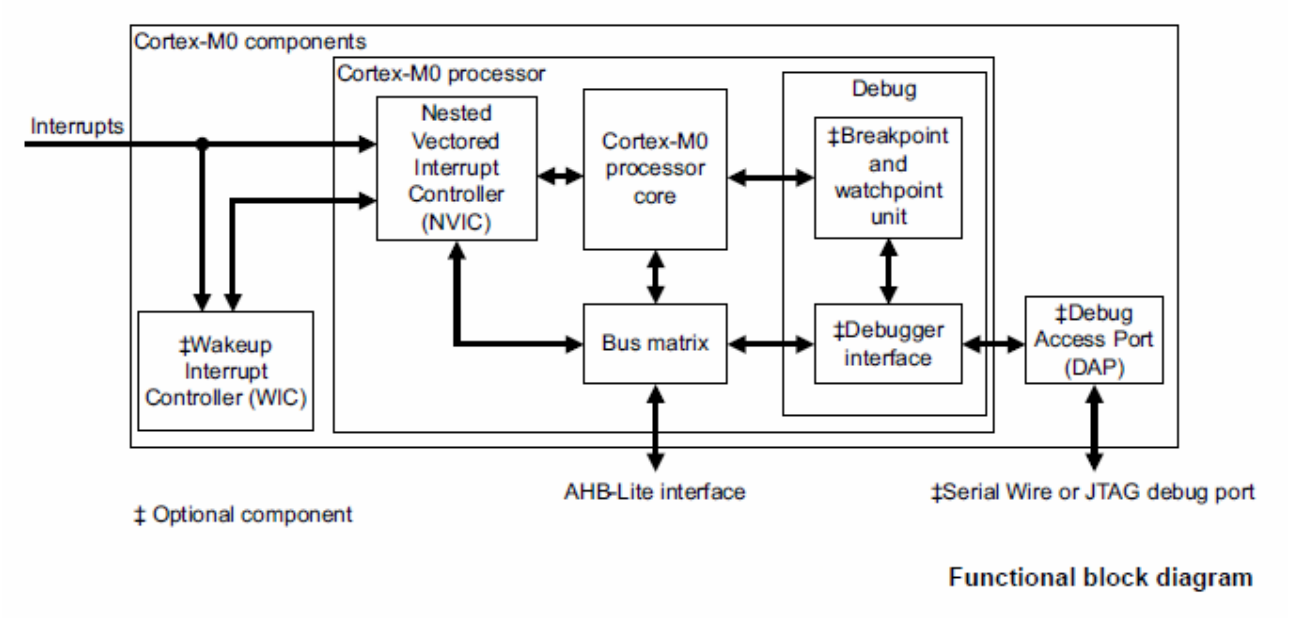
Applicazioni d'uso

Le applicazioni d'uso sono le stesse viste nel Cortex R4 ovvero nei dispositivi automobilistici, Storage, Applicazioni di comunicazione digitale dei dati,Cellulari e Home.

5. ARM CORTEX M-SERIES

_ARM Cortex M0;
_ARM Cortex M1;
_ ARM Cortex M3;
_ARM Cortex M4.

ARM CORTEX M0



Functional block diagram

Il processore Cortex M0 è un processore RISC a 32 bit multistadio configurabile. Contiene un unità NVIC,un interfaccia AMBA AHB Lite, e un unità di Debug Hardware opzionale. Dallo schema a blocchi emerge:

- Un processore low gate count (a basso numero di porte) che dispone:
  _ di un set d'istruzioni Thumb dell'architettura ARMv6-M;
  _di un set d'istruzioni Thumb 2 a 32 bit;
  _di un opzionale timer SysTick a 24 bit;
  _Moltiplicatore hardware a 32 bit: possibilità di scegliere tra un moltiplicatore standard (3 cicli) e uno più piccolo, con prestazioni più basse (32 cicli);
  _Un interfaccia di sistema con rappresentazione dei dati little Endian o Big Endian;
  _Le operazioni Load/Store multiple o multiciclo sono abbandonate per facilitare la gestione degli interupt;
  _*C Application Binary Interface* (C-ABI) che utilizza funzioni di tipo C ovvero quelle di gestione delle interruzioni;
  _il processore entra nella modalità *Low power sleep-mode* quando esegue le istruzioni *Wait For Interrupt* (WFI), *Wait For Event* (WFE) ovvero quando è in attesa di gestire un interruzione o evento in modalità a basso consumo.
- Presenza di un controllore di interrupts integrato (il processore e questa unità lavorano a bassa latenza sugli interupts), *Nested Vectored Interrupt Controller (NVIC),* con le seguenti caratteristiche:
  _Configura fino a 32 interruzioni sui 32 ingressi esterni di interupt, e ogni ingresso ha uno dei 4 livelli di priorità;
  _contiene un opzionale controllore *Wake-up Interrupt Controller* (WIC), che abilita il processore e la NVIC ad essere messi nella modalità *sleep mode*.

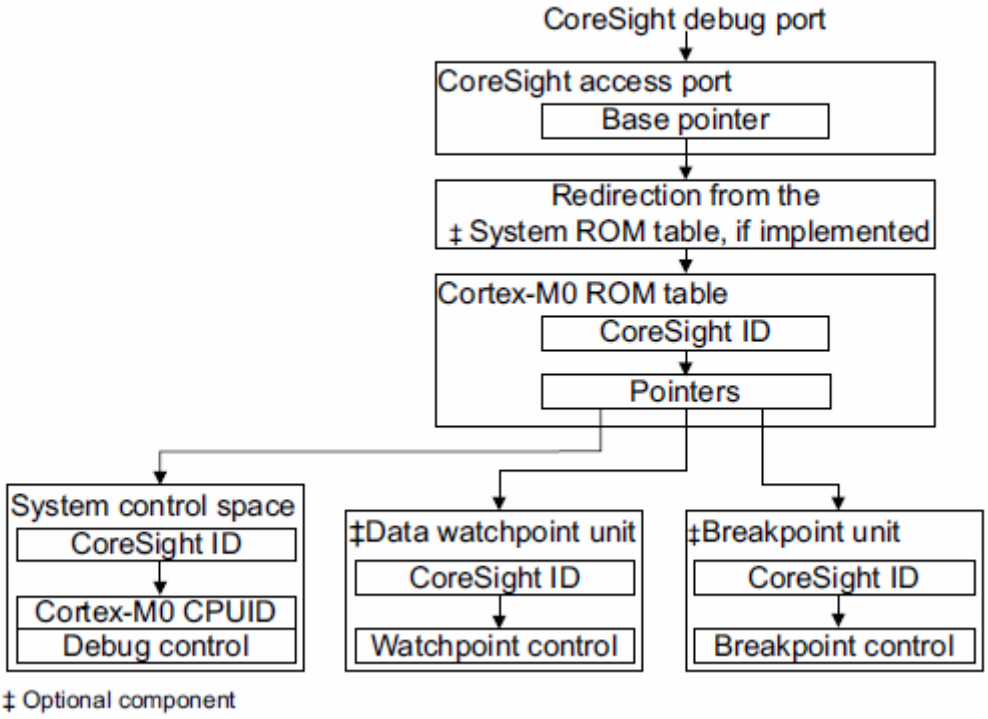
Inoltre identifica le interruzioni in entrata al processore e assegna un livello di priorità ad ogni interruzione.
L'unità NVIC lavora assieme al controllore WIC ed ha il compito quindi di rilevare tutte i segnali d'interruzione esterni, assegnare una priorità di gestione, e configurare le apposite interruzioni tramite gli appositi registri NVIC che riporto di seguito:

NVIC registers

Name	Description
ISER	*Interrupt Set-Enable Register* in the *ARMv6-M ARM*
ICER	*Interrupt Clear-Enable Register* in the *ARMv6-M ARM*
ISPR	*Interrupt Set-Pending Register* in the *ARMv6-M ARM*
ICPR	*Interrupt Clear-Pending Register* in the *ARMv6-M ARM*
IPR0-IPR7	*Interrupt Priority Registers* in the *ARMv6-M ARM*

Le interruzioni una volte configurate vengono poi gestite dal processore.
Nei Registri NVIC possono essere scritti solo Word di dati, dati in formato halfword,e DoubleWord non sono ammessi.

- Un unità opzionale di *Debug* che contiene i breakpoints nella BreakPoint unit e i watchpoints nella Data watchpoint unit. L'identificazione del relativo indirizzo in memoria in cui viene gestito il breakpoint o watchpoint viene recuperato nel modo seguente:



_ si legge prima di tutto attraverso il CoreSight ID di un determinato componente l'indirizzo in memoria in cui si trova la Cortex-M0 Rom table

associata a quel componente. Per fare questo vado a leggere nella tabella Cortex-M0 ROM table identification values il relativo valore per un determinato componente o periferica.Ad esempio la tabella Cortex-M0 ROM del componente Component ID0 si trova all'indirizzo in memoria 0X0000000D. Nella Tabella Cortex-M0 ROM Table troviamo gli indirizzi in memoria a cui si trovano le tabelle SCS,DWT,e BPU.

Cortex-M0 ROM table identification values

Register	Value	Description
Peripheral ID4	0x00000004	*Component and peripheral ID register formats* in the *ARMv6-M ARM*
Peripheral ID0	0x00000071	
Peripheral ID1	0x000000B4	
Peripheral ID2	0x0000000B	
Peripheral ID3	0x00000000	
Component ID0	0x0000000D	
Component ID1	0x00000010	
Component ID2	0x00000005	
Component ID3	0x000000B1	

Table 6-2 shows the CoreSight components that the Cortex-M0 ROM table points to. The values depend on the implemented debug configuration.

Table 6-2 Cortex-M0 ROM table components

Component	Value
SCS	0xFFF0F003
DWT	0xFFF02003[a]
BPU	0xFFF03003[b]
End marker	0x00000000
MemType	0x00000001

Nella tabella SCS riportata di seguito vado a leggere per un determinato CoreSight ID (l'ID del componente o periferica) l'indirizzo in memoria in cui si trova il Debug Control.

SCS CoreSight identification

Table 6-3 shows the SCS CoreSight identification registers and values for debugger detection. Final debugger identification of the Cortex-M0 processor is through the CPUID register in the SCS

Table 6-3 SCS identification values

Register	Value	Description
Peripheral ID4	0x00000004	*Component and Peripheral ID register formats* in the *ARMv6-M ARM*
Peripheral ID0	0x00000008	
Peripheral ID1	0x000000B0	
Peripheral ID2	0x0000000B	
Peripheral ID3	0x00000000	
Component ID0	0x0000000D	
Component ID1	0x000000E0	
Component ID2	0x00000005	
Component ID3	0x000000B1	

Invece nella *Data Watchpoint Unit (DWT)* riportata di seguito vado a leggere i relativi indirizzi dei relativi watchpoints. Questi indirizzi sono riportati negli appositi registri.
In questa tabella possono essere contenuti da 0 a 2 watchpoints.Se un processore ha 0 watchpoints questa tabella non esiste.

DWT CoreSight identification

Table 6-4 shows the DWT identification registers and values for debugger detection.

Table 6-4 DWT identification values

Register	Value	Description
Peripheral ID4	0x00000004	*Component and Peripheral ID register formats* in the *ARMv6-M ARM*
Peripheral ID0	0x0000000A	
Peripheral ID1	0x000000B0	
Peripheral ID2	0x0000000B	
Peripheral ID3	0x00000000	
Component ID0	0x0000000D	
Component ID1	0x000000E0	
Component ID2	0x00000005	
Component ID3	0x000000B1	

Nella *Breakpoint unit(BPU)* riportata di seguito sono contenuti da 0 a 4 BreakPoints. Per ogni Breakpoint ne è riportato l'indirizzo in memoria in cui viene gestito.

Table 6-5 shows the BPU identification registers and their values for debugger detection.

Table 6-5 BPU identification registers

Register	Value	Description
Peripheral ID4	0x00000004	*Component and Peripheral ID register formats* in the *ARMv6-M ARM*
Peripheral ID0	0x0000000B	
Peripheral ID1	0x000000B0	
Peripheral ID2	0x0000000B	
Peripheral ID3	0x00000000	
Component ID0	0x0000000D	
Component ID1	0x000000E0	
Component ID2	0x00000005	
Component ID3	0x000000B1	

- I seguenti Bus d'interfaccia:
  _ l'interfaccia AMBA-3 AHB-Lite a 32 bit utilizzata per accedere alle periferiche esterne ;
  _ L'accesso all'unità di Debug avviene attraverso la porta Debug Access Port (DAP).

CARATTERISTICHE DEL PROCESSORE

_Implementa l'architettura *ARMv6-M ARM* più le unita NVIC e Debug già viste prima.
_*Stati Operativi*:
_Thumb state: stato in cui il processore si trova durante la normale esecuzione del codice;
_Debug state: stato in cui il processore si trova durante il debug.
_*Modi d'esecuzione*:
_Thread mode: il processore entra in questa modalità al reset e può rientrarci come risultato di ritorno di un'eccezione;
_ Handler mode: il processore entra in questa modalità a seguito di un'eccezione;
_Low power Sleep Mode: quando è in attesa di gestire un interruzione o evento.
_*Set d'istruzioni:* Tutte le istruzioni Thumb dell'architettura ARMv6-M.Tutte le istruzioni Thumb a 16 bit dell'architettura ARMv7-M ad esclusione di CBZ, CBNZ e IT. Tutte le seguenti istruzioni Thumb-2 a 32 bit BL, DMB, DSB, ISB, MRS e MSR.

Di seguito riporto una tabella con tutte le istruzioni incluse:

Table shows the Cortex-M0 instructions and their cycle counts. The cycle counts are based on a system with zero wait-states.

Table Cortex-M0 instruction summary

Operation	Description	Assembler	Cycles
Move	8-bit immediate	MOVS Rd, #<imm>	1
	Lo to Lo	MOVS Rd, Rm	1
	Any to Any	MOV Rd, Rm	1
	Any to PC	MOV PC, Rm	3
Add	3-bit immediate	ADDS Rd, Rn, #<imm>	1
	All registers Lo	ADDS Rd, Rn, Rm	1
	Any to Any	ADD Rd, Rd, Rm	1
	Any to PC	ADD PC, PC, Rm	3
	8-bit immediate	ADDS Rd, Rd, #<imm>	1
	With carry	ADCS Rd, Rd, Rm	1
	Immediate to SP	ADD SP, SP, #<imm>	1
	Form address from SP	ADD Rd, SP, #<imm>	1
	Form address from PC	ADR Rd, <label>	1
Subtract	Lo and Lo	SUBS Rd, Rn, Rm	1
	3-bit immediate	SUBS Rd, Rn, #<imm>	1
	8-bit immediate	SUBS Rd, Rd, #<imm>	1
	With carry	SBCS Rd, Rd, Rm	1
	Immediate from SP	SUB SP, SP, #<imm>	1

Table Cortex-M0 instruction summary (continued)

Operation	Description	Assembler	Cycles
Subtract	Negate	RSBS Rd, Rn, #0	1
Multiply	Multiply	MULS Rd, Rm, Rd	1 or 32[a]
Compare	Compare	CMP Rn, Rm	1
	Negative	CMN Rn, Rm	1
	Immediate	CMP Rn, #<imm>	1
Logical	AND	ANDS Rd, Rd, Rm	1
	Exclusive OR	EORS Rd, Rd, Rm	1
	OR	ORRS Rd, Rd, Rm	1
	Bit clear	BICS Rd, Rd, Rm	1
	Move NOT	MVNS Rd, Rm	1
	AND test	TST Rn, Rm	1
Shift	Logical shift left by immediate	LSLS Rd, Rm, #<shift>	1
	Logical shift left by register	LSLS Rd, Rd, Rs	1
	Logical shift right by immediate	LSRS Rd, Rm, #<shift>	1
	Logical shift right by register	LSRS Rd, Rd, Rs	1
	Arithmetic shift right	ASRS Rd, Rm, #<shift>	1
	Arithmetic shift right by register	ASRS Rd, Rd, Rs	1
Rotate	Rotate right by register	RORS Rd, Rd, Rs	1
Load	Word, immediate offset	LDR Rd, [Rn, #<imm>]	2
	Halfword, immediate offset	LDRH Rd, [Rn, #<imm>]	2
	Byte, immediate offset	LDRB Rd, [Rn, #<imm>]	2
	Word, register offset	LDR Rd, [Rn, Rm]	2
	Halfword, register offset	LDRH Rd, [Rn, Rm]	2
	Signed halfword, register offset	LDRSH Rd, [Rn, Rm]	2
	Byte, register offset	LDRB Rd, [Rn, Rm]	2

Table Cortex-M0 instruction summary (continued)

Operation	Description	Assembler	Cycles
Load	Signed byte, register offset	LDRSB Rd, [Rn, Rm]	2
	PC-relative	LDR Rd, <label>	2
	SP-relative	LDR Rd, [SP, #<imm>]	2
	Multiple, excluding base	LDM Rn!, {<loreglist>}	1+N[b]
	Multiple, including base	LDM Rn, {<loreglist>}	1+N[b]
Store	Word, immediate offset	STR Rd, [Rn, #<imm>]	2
	Halfword, immediate offset	STRH Rd, [Rn, #<imm>]	2
	Byte, immediate offset	STRB Rd, [Rn, #<imm>]	2
	Word, register offset	STR Rd, [Rn, Rm]	2
	Halfword, register offset	STRH Rd, [Rn, Rm]	2
	Byte, register offset	STRB Rd, [Rn, Rm]	2
	SP-relative	STR Rd, [SP, #<imm>]	2
	Multiple	STM Rn!, {<loreglist>}	1+N[b]
Push	Push	PUSH {<loreglist>}	1+N[b]
	Push with link register	PUSH {<loreglist>, LR}	1+N[b]
Pop	Pop	POP {<loreglist>}	1+N[b]
	Pop and return	POP {<loreglist>, PC}	4+N[c]
Branch	Conditional	B<cc> <label>	1 or 3[d]
	Unconditional	B <label>	3
	With link	BL <label>	4
	With exchange	BX Rm	3
	With link and exchange	BLX Rm	3
Extend	Signed halfword to word	SXTH Rd, Rm	1
	Signed byte to word	SXTB Rd, Rm	1
	Unsigned halfword	UXTH Rd, Rm	1

Table Cortex-M0 instruction summary (continued)

Operation	Description	Assembler	Cycles
Extend	Unsigned byte	UXTB Rd, Rm	1
Reverse	Bytes in word	REV Rd, Rm	1
	Bytes in both halfwords	REV16 Rd, Rm	1
	Signed bottom half word	REVSH Rd, Rm	1
State change	Supervisor Call	SVC #<imm>	-[e]
	Disable interrupts	CPSID i	1
	Enable interrupts	CPSIE i	1
	Read special register	MRS Rd, <specreg>	4
	Write special register	MSR <specreg>, Rn	4
	Breakpoint	BKPT #<imm>	-[e]
Hint	Send event	SEV	1
	Wait for event	WFE	2[f]
	Wait for interrupt	WFI	2[f]
	Yield	YIELD[g]	1
	No operation	NOP	1
Barriers	Instruction synchronization	ISB	4
	Data memory	DMB	4
	Data synchronization	DSB	4

a. Depends on multiplier implementation.
b. N is the number of elements.
c. N is the number of elements in the stack-pop list including PC and assumes load or store does not generate a HardFault exception.
d. 3 if taken, 1 if not-taken.
e. Cycle count depends on core and debug configuration.
f. Excludes time spent waiting for an interrupt or event.
g. Executes as NOP.

_*Mappaggio della memoria*:di seguito sono riportate le aree di memoria riservate per i dati(data),per il codice delle istruzioni(code), e per i dispositivi(device):

Memory map usage

Address range	Code	Data	Device
0xF0000000 - 0xFFFFFFFF	No	No	Yes
0xE0000000 – 0xEFFFFFFF	No	No	No[a]
0xA0000000 – 0xDFFFFFFF	No	No	Yes
0x60000000 – 0x9FFFFFFF	Yes	Yes	No
0x40000000 – 0x5FFFFFFF	No	No	Yes
0x20000000 - 0x3FFFFFFF	Yes[b]	Yes	No
0x00000000 - 0x1FFFFFFF	Yes	Yes	No

a. Space reserved for Cortex-M0 NVIC and debug components.
b. Cortex-M1 devices implementing data *Tightly-Coupled Memories* (TCMs) in this region do not support code execution from the data TCM.

_*Registri:*

Table Processor core register set summary

Name	Description
R0-R12	R0-R12 are general-purpose registers for data operations.
MSP (R13) PSP (R13)	The *Stack Pointer* (SP) is register R13. In Thread mode, the CONTROL register indicates the stack pointer to use, *Main Stack Pointer* (MSP) or *Process Stack Pointer* (PSP).
LR (R14)	The *Link Register* (LR) is register R14. It stores the return information for subroutines, function calls, and exceptions.
PC (R15)	The *Program Counter* (PC) is register R15. It contains the current program address.
PSR	The *Program Status Register* (PSR) combines: • *Application Program Status Register* (APSR) • *Interrupt Program Status Register* (IPSR) • *Execution Program Status Register* (EPSR). These registers provide different views of the PSR.
PRIMASK	The PRIMASK register prevents activation of all exceptions with configurable priority.
CONTROL	The CONTROL register controls the stack used when the processor is in Thread mode.

Tutti i registri sono a 32 bit.

CARATTERISTICHE TECNICHE E PRESTAZIONI

Il processore ARM Cortex-M0 ™ è il più piccolo(cioè utilizza dispositivi di soli 8 o 16 bit), ha la potenza più bassa e la più alta efficienza energetica dei processore ARM disponibili.

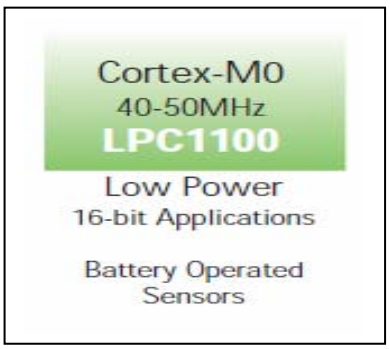
E un processore **Ultra Low Power**: cioè consuma pochissimo, solo 85 µW / MHz (0,085 milliwatt).

Optimized connectivity: Progettato per supportare la connettività a bassa potenza come il Bluetooth Low Energy (BLE), IEEE 802.15 e Z-Wave, in particolare nei dispositivi analogici che stanno aumentando la loro funzionalità digitale di pre-trattamento e la comunicazione dati in modo efficiente.

Cortex-M0	
Architecture	ARMv6-M (Von Neumann)
ISA support	Thumb®/ Thumb-2 technology*
Pipeline	3-stage
Dhrystone	0.9 DMIPS/MHz
Interrupts	NMI + 1 to 32 physical interrupts
Interrupt latency	16 cycles
Sleep modes	Integrated WFI and WFE instructions Sleep & Deep Sleep Signals Optional Retention Mode with Power Management Kit
Enhanced Instructions	Single-cycle (32x32) multiply
Debug	JTAG or Serial-Wire Debug ports

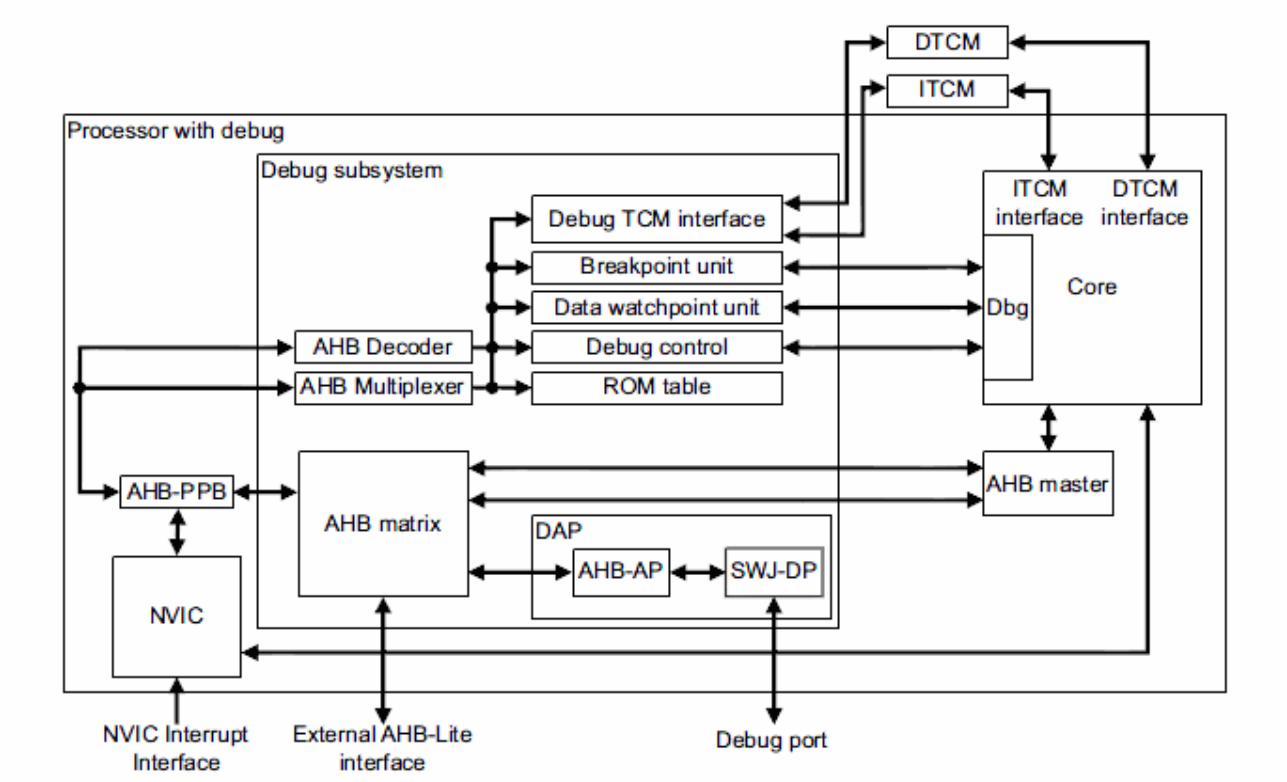
Applicazioni d'uso

Gli ambiti d'uso riguardano principalmente i microcontrollori (MCU) a basso consumo per esempio i microcontrollori LPC11xx della casa di semiconduttori NXP: un esempio è il microcontrollore ARM Cortex M0 NSP LPC1100 che ha una frequenza di 40-50MHZ.



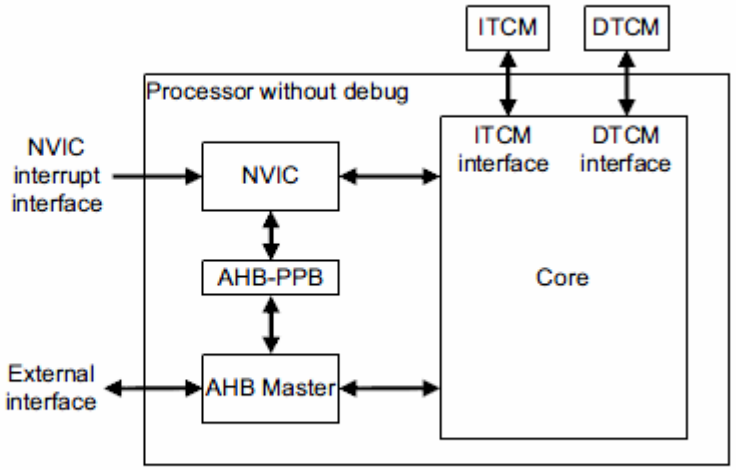
Gli usi riguardano anche quei microconttrollori utilizzati nelle applicazioni embedded e SoC a basso consumo quali:
_ negli accessori dei giochi(gamepad,joypad);
_nelle applicazioni e-Metering;
_nei dispositivi d'illuminazione;
_nelle automobili: i dispositivi di controllo del motore per esempio i dispositivi engine control module (ECM) e powertrain control unit/module (PCU)(dispositivi per il controllo della combustione e delle emissioni inquinanti nel motore),i sensori ovvero i sensori della temperatura del motore, i sensori di velocità, i sensori di rotazione del motore;
_ nelle applicazioni Smart Control;
_nei dispositivi Medici;
_nei dispositivi di controllo della potenza(Power Control);
_nei dispositivi di segnali analogici e misti(Analog and Mixed Signal Devices);
_nei dispositivi touch-screens.

ARM CORTEX M1



Processor with debug block diagram

Figure shows the structure of the processor without debug.



Processor block diagram

Dallo schema a blocchi emerge:
_ il *Core*(processore)le cui caratteristiche saranno descritte più avanti nella sezioni caratteristiche.
_ le seguenti *Interfacce di memoria TCM*: ITCM interface e DTCM interface usate dal processore per accedere alle memorie esterne : dei dati DTCM e delle istruzioni ITCM.
_*BUS:* un bus interno *AHB-PPB* per accedere all'unità NVIC e all'unità di Debug se è presente. Un bus AHB Master che riceve i segnali dai dispositivi esterni tramite l'External Interface e si connette alla porta AHB-PPB, al processore, e all'unità di debug se è presente.
_Un unità NVIC che lavora con il processore nella gestione delle interruzioni ed eccezioni:

_Configura tramite gli appositi registri NVIC 1, 8, 16, or 32 interruzioni esterne;
_assegna uno dei 4 livelli di priorità alle interruzioni;
_Lo stato del processore è salvato automaticamente al verificarsi di un interruzione e ripristinato all'uscita dell'interruzione.

Il controllore possiede i seguenti registri:

NVIC registers

Name of register	Type	Address	Reset value
Interrupt Set Enable Register	R/W	0XE000E100	0x00000000
Interrupt Clear Enable Register	R/W	0XE000E180	0x00000000
Interrupt Set Pending Register	R/W	0XE000E200	0x00000000
Interrupt Clear Pending Register	R/W	0XE000E280	0x00000000
Priority 0 Register	R/W	0XE000E400	0x00000000
Priority 1 Register	R/W	0XE000E404	0x00000000
Priority 2 Register	R/W	0XE000E408	0x00000000
Priority 3 Register	R/W	0XE000E40C	0x00000000
Priority 4 Register	R/W	0XE000E410	0x00000000
Priority 5 Register	R/W	0XE000E414	0x00000000
Priority 6 Register	R/W	0XE000E418	0x00000000
Priority 7 Register	R/W	0XE000E41C	0x00000000

Interupt Set Enable Register

Viene usato per abilitare le interruzione , e stabilisce quale interruzione è attualmente abilitata.Gli indirizzi in memoria del registro, il tipo accesso al registro, e il valore nello stato di Reset del registro sono riportati di seguito:
Il registro ha 32 bit, ogni bit corrisponde ad un interruzione. Abilitando uno dei 32 bit abilito una delle 32 interruzioni.Un bit è abilitato se presenta il valore 1, se ha 0 è disabilitato.

Interupt Clear Enable Register

Viene usato per disabilitare le interruzioni. Mi permette di capire quali interruzioni sono abilitate. Il registro è a 32 bit, ed ogni bit corrisponde ad una delle 32 interruzioni. Disabilitando l'i bit disabilito l'i interruzione, con 0<=i<=32. Un bit è disabilitato se presenta il valore 1, abilitato se presenta il valore 0.

Interupt Set Pending Register

Stabilisce quali interruzioni sono nello stato d'attesa. Anche qui il registro ha 32 bit e ogni i-esimo bit corrisponde alla i-esima interruzione. Scrivendo il valore 1 in un bit metto l'interruzione in attesa, scrivendo 0 nel bit l'interruzione non è più in attesa.

Interupt Clear Pendine Register

Stabilisce quali interruzioni non sono più nello stato d'attesa. Anche qui il registro ha 32 bit e ogni i-esimo bit corrisponde alla i-esima interruzione.Impostando nel i-esimo bit il valore 1 cancello quel i-esima interruzione dallo stato d'attesa.Impostando il valore 0 per l'i-esimo bit lascio la i-esima interruzione nello stato d'attesa.

Interupt Priority Register

Questo registro a 32 bit assegna una priorità da 0 a 3 ad ognuna delle interruzioni. 0 indica la priorità più alta, 3 quella più bassa.Questi 2 bit di priorità sono contenuti rispettivamente nei bit 7-6 di ogni byte.

	31 30 29	24 23 22 21	16 15 14 13	8 7 6 5	0
E000E400	IP_3	IP_2	IP_1	IP_0	
E000E404	IP_7	IP_6	IP_5	IP_4	
E000E408	IP_11	IP_10	IP_9	IP_8	
E000E40C	IP_15	IP_14	IP_13	IP_12	
E000E410	IP_19	IP_18	IP_17	IP_16	
E000E414	IP_23	IP_22	IP_21	IP_20	
E000E418	IP_27	IP_26	IP_25	IP_24	
E000E41C	IP_31	IP_30	IP_29	IP_28	
	Reserved	Reserved	Reserved	Reserved	

Interrupt Priority Registers 0-7 bit assignments

Interrupt Priority Registers 0-31 bit assignments

Bits	Field	Function
[7:6]	IP_*n*	Priority of interrupt *n*

_Un unità opzionale di *Debug* che può presentarsi in 2 tipi di configurazione:

1. Configurazione full debug, che ha 4 breakpoint e 2 watchpoint.E'la configurazione di default;
2. Configurazione reduce debug: che ha 2 breakpoint ed 1 watchpoint.

L'unità di debug è formata dalle seguenti Unità:

- Una AHB decoder che decodifica i segnali in ingresso, questi segnali decodificati sono poi inoltrati dal AHB-Multiplexer in una delle 5 uscite a cui è collegato.
- Un unità AHB Matrix che controlla tutti gli accessi provenienti dal processore, dalla porta DAP,dalla porta AHB-PPB, e dall'interfaccia esterna AHB-Lite.
- Una porta DAP composta da 2 porte : la porta SWJ-DP e la porta AHB-AP. La porta SWJ-DP fornisce l'accesso a tutti i registi di sistema inclusi quelli del processore.
- Un interfacce di debug TCM per interfacciarsi con le memorie ITCM e DTCM.
- Un unità Debug Control che ha il compito di monitorare lo stato del processore effettuando dei test sul processore.

- Una ROM Table che contiene i puntatori alle tabelle BPU,DW,SCS ,e i valori che assumono i registri dei componenti e delle periferiche con i rispettivi indirizzi di memoria.

Table ROM memory

Address	Value	Name	Bits	Description
0xE00FF000	0xFFF0F003	SCS	[31:0]	Points to the *System Control Space* (SCS) at 0xE000E000. This includes core debug control registers.
0xE00FF004	0xFFF02003	DW	[31:0]	Points to the DW unit at 0xE0001000.
0xE00FF008	0xFFF03003	BPU	[31:0]	Points to the BPU at 0xE0002000.
0xE00FF00C	0x00000000	end	[31:0]	Marks of end of table. Because adding more debug components is not permitted, this value is fixed.
0xE00FFFCC	0x00000001	MEMTYPE	[7:0]	System memory map is always accessible from the DAP. Always set to 0x1.
0xE00FFFD0	0x00000004	Peripheral ID4	[31:8]	Reserved.
			[7:4]	Indicates the size of the ROM table: 0x0 = 4KB ROM table.
			[3:0]	JEP106 continuation code: 0x4
0xE00FFFD4	0x00000000	Peripheral ID5	-	Reserved.
0xE00FFFD8	0x00000000	Peripheral ID6	-	
0xE00FFFDC	0x00000000	Peripheral ID7	-	
0xE00FFFE0	0x00000070	Peripheral ID0	[31:8]	Reserved.
			[7:0]	Contains bits [7:0] of the part number: 0x70.
0xE00FFFE4	0x000000B4	Peripheral ID1	[31:8]	Reserved.
			[7:4]	Contains bits [3:0] of the JEP106 ID code: 0xB.
			[3:0]	Contains bits [11:8] of the part number 0x4.

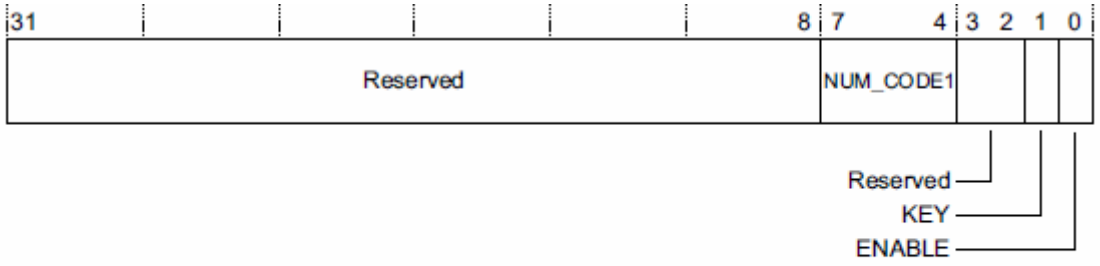
Address	Value	Name	Bits	Description
0xE00FFFE8	0x0000000B	Peripheral ID2	[31:8]	Reserved.
			[7:4]	Indicates the revision: 0x0 = r0p0 0x1 = r0p1.
			[3]	Indicates JEDEC assigned ID fields: 0x1.
			[2:0]	Contains bits [6:4] of the JEP106 ID code: 0x3.
0xE00FFFEC	0x00000000	Peripheral ID3	[31:8]	Reserved.
			[7:4]	Indicates minor revision field RevAnd.
			[3:0]	Indicates block unmodified: 0x0.
0xE00FFFF0	0x0000000D	Component ID0	[31:8]	Reserved.
			[7:0]	Preamble[a].
0xE00FFFF4	0x00000010	Component ID1	[31:8]	Reserved.
			[7:4]	Indicates component class: 0x1 = ROM table.
			[3:0]	Preamble[a].
0xE00FFFF8	0x00000005	Component ID2	[31:8]	Reserved.
			[7:0]	Preamble[a].
0xE00FFFFC	0x000000B1	Component ID3	[31:8]	Reserved.
			[7:0]	Preamble[a].

a. Preamble enables a debugger to detect the presence of the ROM table.

- Una Breakpoint Unit(BPU) che implementa 4 istruzioni per i 4 breakpoints in configurazione full debug,e 2 istruzioni per i 2 breakpoints in configurazione reduce debug. La BPU utilizza due registri: il **Breakpoint Control Register, e il Breakpoint Comparator Registers.**

Breakpoint Control Register

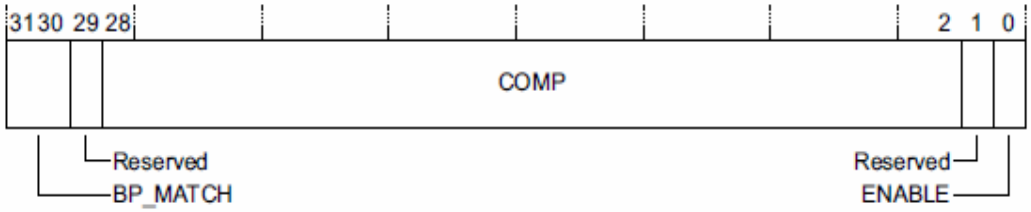
Questo registro abilita i relativi blocchi di breakpoint e ne indica il numero di istruzioni utilizzate.



Breakpoint Control Register bit assignments

Bits	Field	Type	Function
[31:8]	-	RO	Reserved.
[7:4]	NUM_CODE1	RO	Number of comparators. This read-only field and contains either: b0100 = four instruction comparators in use b0010 = two instruction comparators in use.
[3:2]	-	RO	Reserved.
[1]	KEY	WO	Key field. To write to the Breakpoint Control Register, you must write a 1 to this write-only bit. This bit is reads as zero.
[0]	ENABLE	R/W	Breakpoint unit enable bit: 1 = Breakpoint unit enabled 0 = Breakpoint unit disabled. **DBGRESETn** clears the ENABLE bit.

Breakpoint Comparator Registers
In questo registro sono conservati gli indirizzi da confrontare con gli indirizzi delle istruzioni.



Breakpoint Comparator Registers bit assignments

Table 8-10 Breakpoint Comparator Registers bit assignments

Bits	Field	Function
[31:30]	BP_MATCH	This field selects what happens when the COMP address is matched.It is interpreted as: b00 = no breakpoint matching b01 = set breakpoint on lower halfword, upper is unaffected b10 = set breakpoint on upper halfword, lower is unaffected b11 = set breakpoint on both lower and upper halfwords.
[29]	-	Reserved.
[28:2]	COMP	Comparison address. Although it is architecturally Unpredictable whether breakpoint matches on the address of the second halfword of a 32-bit instruction to generate a debug event, in this processor it is predictable and a debug event is generated.
[1]	-	Reserved.
[0]	ENABLE	Compare enable for Breakpoint Comparator Register *n*: 1 = Breakpoint Comparator Register *n* compare enabled 0 = Breakpoint Comparator Register *n* compare disabled. The ENABLE bit of BPU_CTRL must also be set to enable comparisons. **DBGRESETn** clears the ENABLE bit.

- DW Unit(Unità Data Watch): Possono essere implementati due watchpoints(comparators) in modalità full debug, e 1 watchpoint in modalità reduced debug. Un watchpoint(comparator) può essere effettuato sull'hardware oppure su un indirizzo di dati.L'unità utilizza i registri seguenti:

DW Control Register

Per indicare quanti watchpoints(comparators) sono stati implementati.

DW Control Register bit assignments

Bits	Field	Function
[31:28]	NUMCOMP	Number of comparators field. This read-only field contains: • b0010 to indicate two comparators in the full debug configuration • b0001 to indicate one comparator in the reduced debug configuration.
[27:0]	-	Reserved.

DW Comparator Registers

Questo registro contiene un campo a 32 bit COMP, ovvero l'indirizzo da confrontare.

Field	Name	Definition
[31:0]	COMP	DW_COMP to compare against PC or the data address as given by DW_FUNCTION Register. DW_COMP is always masked using the DW Mask Register value before a compare is done.

DW Mask Registers

Quando il confronto tra l'indirizzo COMP e l'indirizzo dei dati o di una istruzione è positivo(match), viene applicata una maschera (Mask) sull'indirizzo dei dati o dell'istruzione:

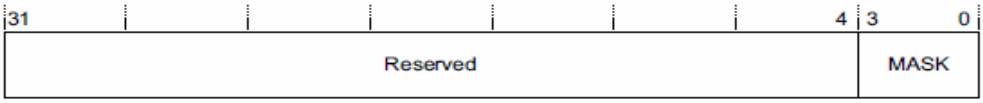


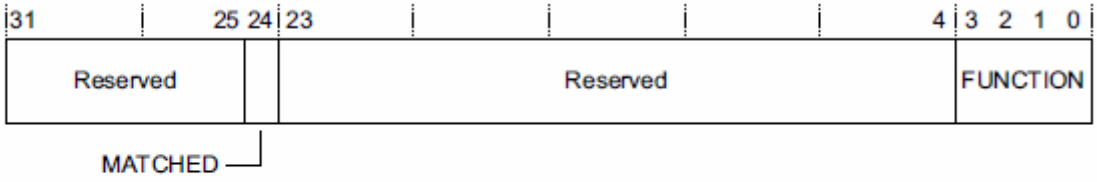
Figure 8-8 DW Mask Registers 0-1 format

DW Mask Registers bit assignments

Bits	Field	Function
[31:5]	-	Reserved.
[4:0]	MASK	Mask on data address when matching against COMP. This is the size of the ignore mask. So, ~0<<MASK forms the mask against the address to use. That is, DW matching is performed as: (ADDR & (~0 << MASK)) == (COMP & (~0 << MASK)) For word accesses the two least significant bits are not compared. For halfword accesses the least significant bit is not compared. For PC matches the least significant bit is not compared.

DW Function Registers

Questo registro nel campo FUNCTION contiene la funzione che deve svolgere il relativo watchpoint(comparator)(per esempio un watchpoint di lettura degli indirizzi).



DW Function Registers bit assignments

DW Function Registers bit assignments

Bits	Field	Function
[31:25]	-	Reserved.
[24]	MATCHED	This bit is set when the comparator matches this bit is cleared on read.
[23:4]	-	Reserved.
[3:0]	FUNCTION	See Table 8-16 for FUNCTION settings.

Table 8-16 Settings for DW Function Registers

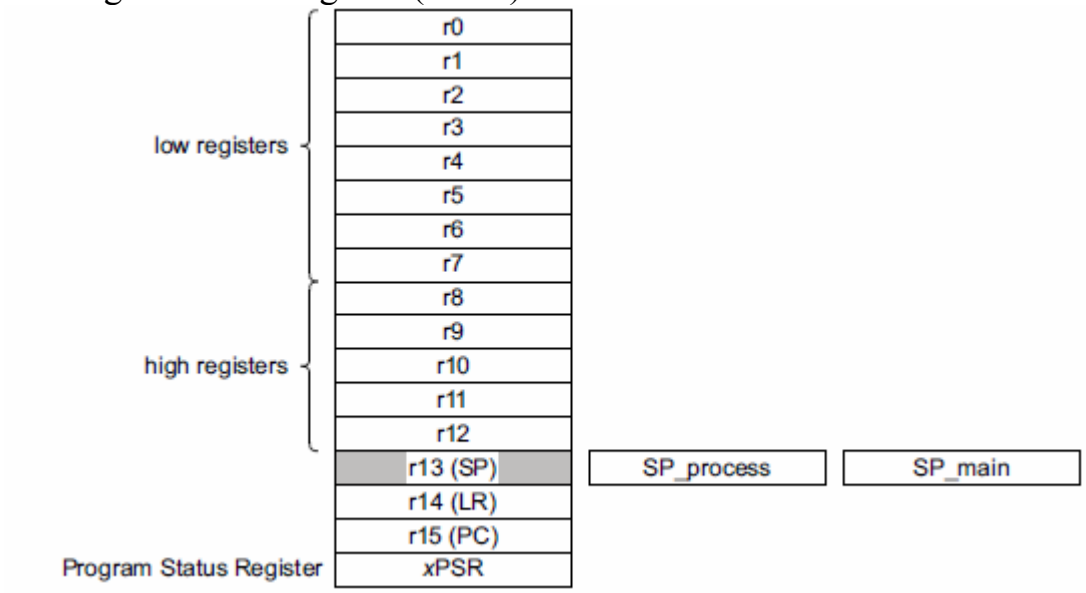
Value	Function
b0000	Disabled
b0001-b0011	Reserved
b0100	Watchpoint on PC match
b0101	Watchpoint on read address
b0110	Watchpoint on write address
b0111	Watchpoint on read or write address
b1000-b1111	Reserved

CARATTERISTICHE DEL PROCESSORE

_implementa l'*architettura* ARM v6-M;
_*Set d'istruzioni* composto da un sottoinsieme del set Thumb e Thumb-2 (architettura ARMv6-M). Non implementa il set d'istruzioni ARM.

_ Architettura *pipeline* a 3 stadi;
_*Stati Operativi*:
_Thumb state: stato in cui il processore si trova durante la normale esecuzione del codice;
_Debug state: stato in cui il processore si trova durante il debug.
_*Modi d'esecuzione*:
_Thread mode: il processore entra in questa modalità al reset e può rientrarci come risultato di ritorno di un'eccezione;
_ Handler mode: il processore entra in questa modalità a seguito di un'eccezione.
_*Rappresentazione dei dati* in formato Big endian o little endian (le istruzioni sono sempre little endian); *tipi di dati* supportati:word (32 bit), half word e byte;
_Presenza di 13 *registri* general purpose a 32 bit indicati con R0-R12, divisi in *low* (R0-R7), accessibili da tutte le istruzioni che specificano un registro general purpose, e *high* (R8-R12), non accessibili da istruzioni a 16 bit; e dei seguenti registri:

- Link register (LR, R14);
- Program counter (PC, R15);
- Fino a 2 (nel caso in cui venga supportato un sistema operativo) Stack pointer (SP, R13);
- Program Status register (xPSR).



_ *Memorie esterne* ITCM e DTCM con dimensione che varia da 1KB fino a 1MB.Nella configurazione di default sono entrambe a 32KB.
_*Moltiplicatore hardware* a 32 bit: possibilità di scegliere tra un moltiplicatore standard (3 cicli) e uno più piccolo, con prestazioni più basse (33 cicli).

CARATTERISTICHE TECNICHE E PRESTAZIONI

I cortex M1 sono implementati su diverse FPGA(circuiti integrati digitali) di case produttrici diverse. In tabella 3.6 vengono riportati alcuni esempi di FPGA su cui è possibile implementare il processore e i relativi software di sviluppo. Da notare che questo processore (grazie ai tool messi a disposizioni) può essere direttamente programmato nei linguaggi C/C++.

FPGA Device Compatibility	Implementation Tool Compatibility
Actel ProASIC3L & ProASIC3/E	Actel Libero
Actel Fusion	
Actel IGLOO/e	
Altera Cyclone-II	Altera Quartus-II
Altera Cyclone-III	
Altera Stratix-II	Synopsys Synplify Pro
Altera Stratix-III	
Xilinx Spartan-3	Mentor Precision
Xilinx Virtex-2	
Xilinx Virtex-3	
Xilinx Virtex-4	Xilinx ISE

Tabella 3.6:esempi di tool di sviluppo

Di seguito vengono elencate le prestazione del processore implementato in alcuni dei dispositivi FPGA della tabella 3.6:

FPGA Type	Example	Frequency (MHz)	Area (LUTS)
65 nm	Altera Stratix-III, Xilinx Virtex-5	200	1900
90 nm	Altera Stratix-II, Xilinx Virtex-4	150	2300
65 nm	Altera Cyclone-III	100	2900
90 nm	Altera Cyclone-II, Xilinx Spartan-3	80	2600
130 nm	Actel ProASIC3, Actel Fusion	70	4300 Tiles

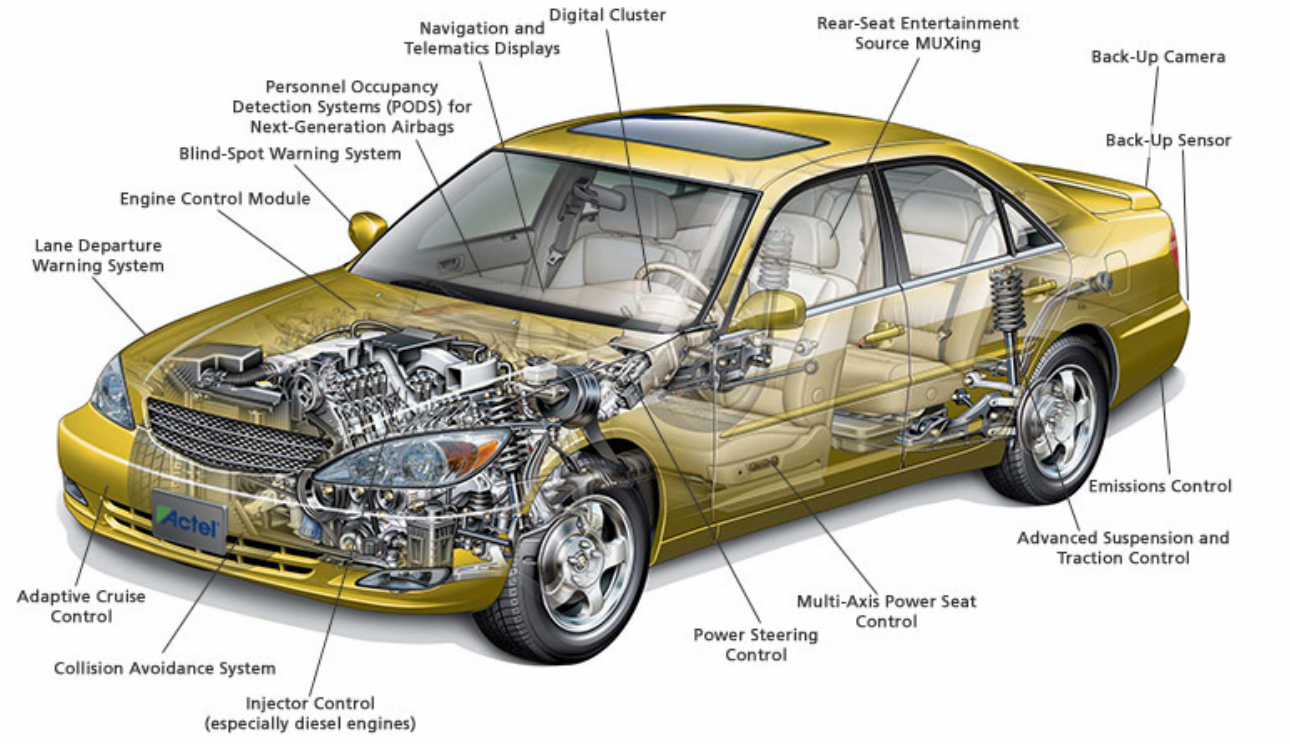
Il processore ha una performance media di 0.8 DMIPS/MHz, ed un consumo di 0.085mw/MHZ.

Applicazioni d'uso

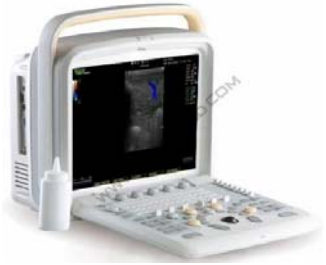
Queste diverse FPGA che fanno uso del processore Cortex M1 sono implementate nelle seguenti applicazioni:

- **Storage:** Hard Disk, Secure Digital (SD) Flash Memory **(**utilizzate nelle machine fotografiche digitale,nelle videocamere, nei cellulari**),**Compact Flash(CF)utilizzate nelle videocamere.

- **Console di gioco portatili**:joystick,gamepad.
- **Automotive**: Dispositivi come quelli riportati nella figura seguente, utilizzati nel settore delle automobili:



- Dispositivi nel settore **Militare/Aerospaziale**.
- Dispositivi portatili nel **settore Medico**: come i sistemi ad ultrasuono portatili come l'ecografo ad ultrasuoni portatile, oppure il microinfusore(utilizzati per liberare insulina) e i defibrillatori esterni automatici.



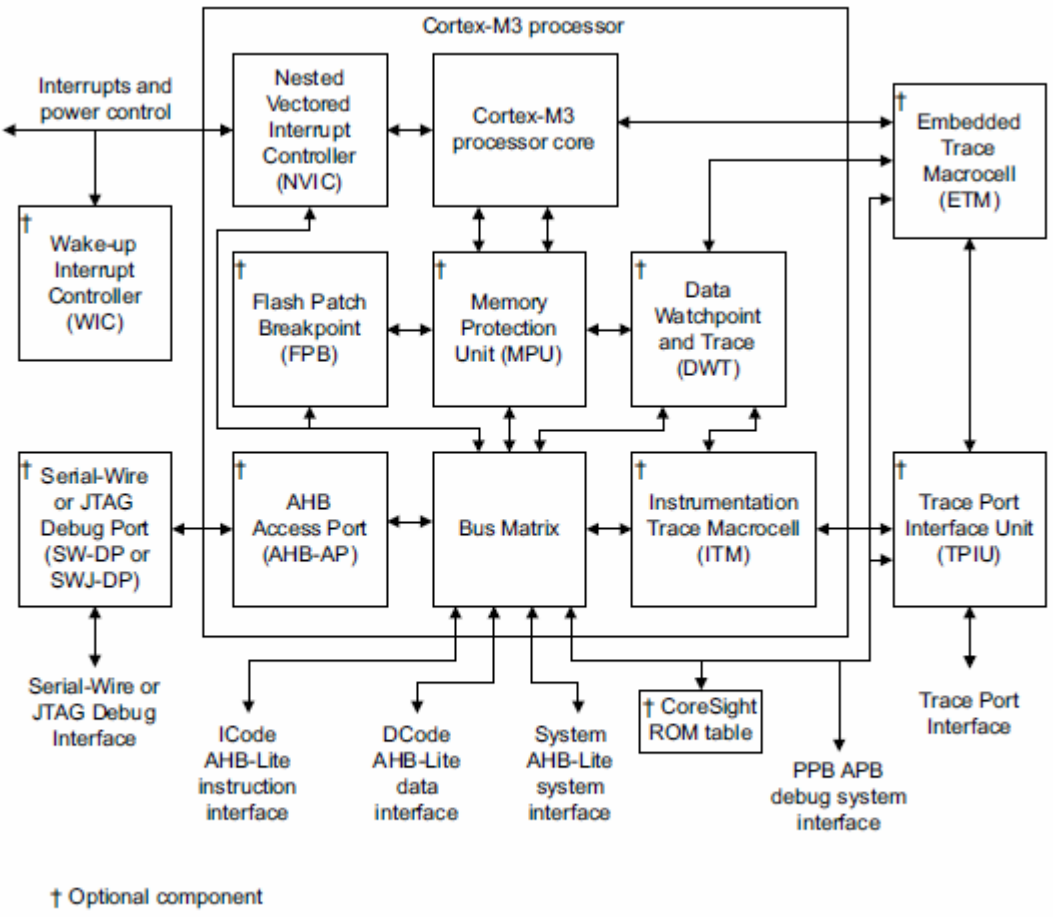
Ecografo ad ultrasuono portatile

Un microinfusore

Defribillatore esterno automatico

- Dispositivi **HMI e Motor Control**: Tastiere, dispositivi touch-screen, altoparlanti, motori in miniatura.
- Dispositivi **Display LCD**: per esempio macchina fotografica digitale, videocamere, palmari.

ARM CORTEX M3



Cortex-M3 block diagram

Dallo schema a blocchi emerge:
_Un *controllore delle interruzioni NVIC* che lavora assieme al processore a bassa latenza sugli interupts.

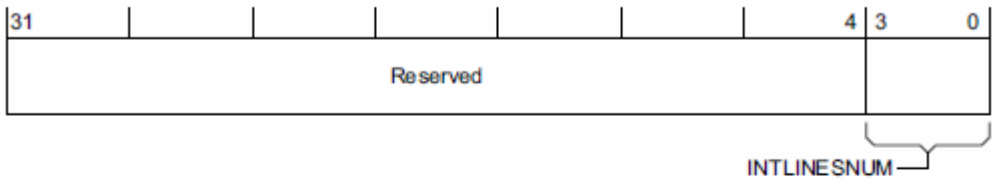
_Configura da 1 a 240 interruzioni esterne tramite gli appositi registri NVIC;
_ Ogni interruzione ha uno dei 256 livelli di priorità;
_ Contiene dai 3 agli 8 bit di priorità;
_Dynamic reprioritization of interruptus:è possibile modificare la priorità di un interupt in modo dinamico;

_Priority grouping: Suddivisione delle interruzioni in più gruppi, ad esempio gruppo1 tutte le interruzioni dalla 0..32, gruppo2 tutte le interruzioni dal 32..64, ecc;
_ Gestione delle interruzioni che arrivano in ritardo nella unità NVIC;
_ Lo stato del processore è salvato automaticamente all'ingresso dell'interruzione e ripristinato all'uscita dell'interruzione;
_ Si affida all'unità WIC(wake-up interupt controller) che ha il compito di identificare e assegnare una priorità alle interruzioni.Inoltre abilita il processore e la NVIC ad essere posti nella modalità low-power sleep mode per la gestione delle interruzioni;
_ Ai registri NVIC è possibile accedere solo in modalità privilegiata(privileged mode);
_Nei registri NVIC possono essere scritti solo byte,halfword,e word di dati o di indirizzi, inoltre questi registri presentano una rappresentazione dei dati di tipo little endian;
_I registri NVIC sono gli stessi visti nel processore Cortex M1 più il registro **Interrupt Controller Type Register, ICTR:** che mi permette di capire tramite il campo INTLINESNUM, quale gruppo di interruzioni bisogna gestire.

NVIC registers

Address	Name	Type	Reset	Description
0xE000E004	ICTR	RO	-	*Interrupt Controller Type Register, ICTR*
0xE000E100 - 0xE000E11C	NVIC_ISER0 - NVIC_ISER7	RW	0x00000000	Interrupt Set-Enable Registers
0xE000E180 - 0E000xE19C	NVIC_ICER0 - NVIC_ICER7	RW	0x00000000	Interrupt Clear-Enable Registers
0xE000E200 - 0xE000E21C	NVIC_ISPR0 - NVIC_ISPR7	RW	0x00000000	Interrupt Set-Pending Registers
0xE000E280 - 0xE000E29C	NVIC_ICPR0 - NVIC_ICPR7	RW	0x00000000	Interrupt Clear-Pending Registers
0xE000E300 - 0xE000E31C	NVIC_IABR0 - NVIC_IABR7	RO	0x00000000	Interrupt Active Bit Register
0xE000E400 - 0xE000E41F	NVIC_IPR0 - NVIC_IPR59	RW	0x00000000	Interrupt Priority Register

Interrupt Controller Type Register, ICTR



ICTR bit assignments

Bits	Name	Function
[31:4]	-	Reserved.
[3:0]	INTLINESNUM	Total number of interrupt lines in groups of 32: b0000 = 0...32 b0001 = 33...64 b0010 = 65...96 b0011 = 97...128 b0100 = 129...160 b0101 = 161...192 b0110 = 193...224 b0111 = 225...256[a]

a. The processor supports a maximum of 240 external interrupts.

_Un unità di protezione della memoria MPU che contiene 8 regioni ciascuna con i relativi attributi e modalità di accesso. Le regioni possono anche essere sovrapposte tra di loro. Può essere implementata anche una configurazione *Sub Region Disable* (SRD), cioè quando si accede ad una regione possono essere disabiltate tutte le sotto regioni.

_ Le seguenti *interfacce BUS:*

_ICode memory interface: interfaccia AHB-Lite a 32 bit utilizzata per leggere le istruzioni (Instruction fetches) nello spazio di memoria che va dall'indirizzo 0x00000000 a 0x1FFFFFFF;

_DCode memory interface: interfaccia AHB-Lite a 32 bit utilizzata per leggere i dati ed effettuare gli accessi di debug nello spazio di memoria che va dall'indirizzo 0x00000000 a 0x1FFFFFFF;

_Interfaccia di sistema *System AHB-Lite system interface*: interfaccia AHB-Lite a 32 bit per la lettura delle istruzioni(Instruction fetches), e per effettuare gli accessi ai dati e all'unità di debug. Gli accessi avvengono nello spazio di memoria compreso tra 0x20000000 a 0xDFFFFFFF e tra 0xE0100000 a 0xFFFFFFFF;

_Private Peripheral Bus (PPB): Bus utilizzato per accedere ai dati e alle unità esterne utilizzando lo spazio di memoria compreso tra 0xE0040000 a 0xE00FFFFF. Ci sono due tipi di questo bus:*PPB APB Debug System Interface* per accedere ad una unità di debug esterna,e *Trace Port Interface Unit* (TPIU) per accedere/uscire dall'unità ITM.

_ Un unità di debug opzionale ETM , che accede al processore e all'unità data watch time DWT se presente, per tenere traccia di tutte le istruzioni che vengono eseguite dal processore.
_ Un unità opzionale *Instrumentation Trace Macrocell Unit ITMU,* che tiene traccia di tutte le caratteristiche del processore: ovvero l'hardware (hardware trace),il software utilizzato(software trace) e il timestamps(utilizzando un contatore a 21 bit).
_ L'unità di debug interna opzionale è costituita dalla *DataWatchpoint and Trace Unit (DWT)* e dalla *Flash Patch and Breakpoint Unit (FPB).*
La Flash Patch and Breakpoint Unit (FPB) implementa 6 punti d'interruzione hardware (hardware breakpoints), e corregge (Patch) il codice delle istruzioni oppure direttamente i dati. Questa unità può effettuare anche dei confronti per verificare se è presente un dato o un istruzione in memoria, oppure raccogliere le caratteristiche sull'esecuzione di una determinata istruzione. I registri utilizzati nella FPB sono riportati nella tabella seguente:

FPB register summary

Address	Name	Type	Reset	Description
0xE0002000	FP_CTRL	RW	0x130	FlashPatch Control Register
0xE0002004	FP_REMAP	RW	-	FlashPatch Remap Register
0xE0002008	FP_COMP0	RW	1'b0[a]	FlashPatch Comparator Register0
0xE000200C	FP_COMP1	RW	1'b0	FlashPatch Comparator Register1
0xE0002010	FP_COMP2	RW	1'b0	FlashPatch Comparator Register2
0xE0002014	FP_COMP3	RW	1'b0	FlashPatch Comparator Register3
0xE0002018	FP_COMP4	RW	1'b0	FlashPatch Comparator Register4
0xE000201C	FP_COMP5	RW	1'b0	FlashPatch Comparator Register5
0xE0002020	FP_COMP6	RW	1'b0	FlashPatch Comparator Register6
0xE0002024	FP_COMP7	RW	1'b0	FlashPatch Comparator Register7
0xE0002FD0	PID4	RO	0x04	Peripheral identification registers
0xE0002FD4	PID5	RO	0x00	
0xE0002FD8	PID6	RO	0x00	
0xE0002FDC	PID7	RO	0x00	
0xE0002FE0	PID0	RO	0x03	
0xE0002FE4	PID1	RO	0xB0	
0xE0002FE8	PID2	RO	0x2B	
0xE0002FEC	PID3	RO	0x00	
0xE0002FF0	CID0	RO	0x0D	Component identification registers
0xE0002FF4	CID1	RO	0xE0	
0xE0002FF8	CID2	RO	0x05	
0xE0002FFC	CID3	RO	0xB1	

a. For FP_COMP0 to FP_COMP7, bit 0 is reset to 0. Other bits in these registers are not reset.

DataWatchpoint and Trace Unit (DWT):implementa 4 watchpoints che possono essere configurati come: un hardware watchpoint, un Trigger ETM,un data address event Trigger, un PC event Trigger. Questa unità contiene anche dei contatori che contano :

_ I cicli di clock (CYCCNT);
_ il numero di istruzioni;
_ il numero di operazioni Load and Store;
_ i cicli d'istruzione;
_il numero di interruzioni.

I registri dell'unità DWT sono riportati di seguito:

DWT register summary

Address	Name	Type	Reset	Description
0xE0001000	DWT_CTRL	RW	See [a]	Control Register
0xE0001004	DWT_CYCCNT	RW	0x00000000	Cycle Count Register
0xE0001008	DWT_CPICNT	RW	-	CPI Count Register
0xE000100C	DWT_EXCCNT	RW	-	Exception Overhead Count Register
0xE0001010	DWT_SLEEPCNT	RW	-	Sleep Count Register
0xE0001014	DWT_LSUCNT	RW	-	LSU Count Register
0xE0001018	DWT_FOLDCNT	RW	-	Folded-instruction Count Register
0xE000101C	DWT_PCSR	RO	-	Program Counter Sample Register
0xE0001020	DWT_COMP0	RW	-	Comparator Register0
0xE0001024	DWT_MASK0	RW	-	Mask Register0
0xE0001028	DWT_FUNCTION0	RW	0x00000000	Function Register0
0xE0001030	DWT_COMP1	RW	-	Comparator Register1
0xE0001034	DWT_MASK1	RW	-	Mask Register1
0xE0001038	DWT_FUNCTION1	RW	0x00000000	Function Register1
0xE0001040	DWT_COMP2	RW	-	Comparator Register2
0xE0001044	DWT_MASK2	RW	-	Mask Register2
0xE0001048	DWT_FUNCTION2	RW	0x00000000	Function Register2
0xE0001050	DWT_COMP3	RW	-	Comparator Register3
0xE0001054	DWT_MASK3	RW	-	Mask Register3
0xE0001058	DWT_FUNCTION3	RW	0x00000000	Function Register3
0xE0001FD0	PID4	RO	0x04	Peripheral identification registers
0xE0001FD4	PID5	RO	0x00	
0xE0001FD8	PID6	RO	0x00	
0xE0001FDC	PID7	RO	0x00	
0xE0001FE0	PID0	RO	0x02	
0xE0001FE4	PID1	RO	0xB0	
0xE0001FE8	PID2	RO	0x3B	
0xE0001FEC	PID3	RO	0x00	

DWT register summary (continued)

Address	Name	Type	Reset	Description
0xE0001FF0	CID0	RO	0x0D	Component identification registers
0xE0001FF4	CID1	RO	0xE0	
0xE0001FF8	CID2	RO	0x05	
0xE0001FFC	CID3	RO	0xB1	

a. Possible reset values are:
0x40000000 if four comparators for watchpoints and triggers are present
0x4F000000 if four comparators for watchpoints only are present
0x10000000 if only one comparator is present
0x1F000000 if one comparator for watchpoints and not triggers is present
0x00000000 if DWT is not present.

_ L'opzionale *Serial Wire Debug Port* (SW-DP) or *Serial Wire JTAG Debug Port* (SWJ-DP): ovvero delle porte esterne al processore utilizzate per accedere/uscire alla porta interna AHB Access port del processore.

CARATTERISTICHE DEL PROCESSORE

_Implementa *l'architettura* ARMv7-M ed una pipeline a 3 stadi con branch predictor.
_Implementa il *set d'istruzioni* Thumb ARMv7-M.Nella tabella 3.1 sono mostrate tutte le istruzioni e il loro numero di cicli.Di seguito riportiamo le abbreviazioni usate nella tabella 3.1:

_P: il numero di cicli richiesti per ricaricare la pipeline.Varia da 1 a 3 in base alla larghezza del Target Instruction;
_B: Il numero di cicli necessari per eseguire l'operazione. Per DSB e DMB, il numero minimo di cicli è pari a zero. Per ISB, il numero minimo di cicli è equivalente al numero richiesto per ricaricare la pipeline;
_N: il numero di registri utilizzato;
_W: il numero di cicli d'attesa di un appropriato evento.

Table 3-1 Cortex-M3 instruction set summary

Operation	Description	Assembler	Cycles
Move	Register	MOV Rd, <op2>	1
	16-bit immediate	MOVW Rd, #<imm>	1
	Immediate into top	MOVT Rd, #<imm>	1
	To PC	MOV PC, Rm	1 + P
Add	Add	ADD Rd, Rn, <op2>	1
	Add to PC	ADD PC, PC, Rm	1 + P
	Add with carry	ADC Rd, Rn, <op2>	1
	Form address	ADR Rd, <label>	1

Table 3-1 Cortex-M3 instruction set summary (continued)

Operation	Description	Assembler	Cycles
Subtract	Subtract	SUB Rd, Rn, <op2>	1
	Subtract with borrow	SBC Rd, Rn, <op2>	1
	Reverse	RSB Rd, Rn, <op2>	1
Multiply	Multiply	MUL Rd, Rn, Rm	1
	Multiply accumulate	MLA Rd, Rn, Rm	2
	Multiply subtract	MLS Rd, Rn, Rm	2
	Long signed	SMULL RdLo, RdHi, Rn, Rm	3 to 5[a]
	Long unsigned	UMULL RdLo, RdHi, Rn, Rm	3 to 5[a]
	Long signed accumulate	SMLAL RdLo, RdHi, Rn, Rm	4 to 7[a]
	Long unsigned accumulate	UMLAL RdLo, RdHi, Rn, Rm	4 to 7[a]
Divide	Signed	SDIV Rd, Rn, Rm	2 to 12[b]
	Unsigned	UDIV Rd, Rn, Rm	2 to 12[b]
Saturate	Signed	SSAT Rd, #<imm>, <op2>	1
	Unsigned	USAT Rd, #<imm>, <op2>	1
Compare	Compare	CMP Rn, <op2>	1
	Negative	CMN Rn, <op2>	1
Logical	AND	AND Rd, Rn, <op2>	1
	Exclusive OR	EOR Rd, Rn, <op2>	1
	OR	ORR Rd, Rn, <op2>	1
	OR NOT	ORN Rd, Rn, <op2>	1
	Bit clear	BIC Rd, Rn, <op2>	1
	Move NOT	MVN Rd, <op2>	1
	AND test	TST Rn, <op2>	1
	Exclusive OR test	TEQ Rn, <op1>	
Shift	Logical shift left	LSL Rd, Rn, #<imm>	1
	Logical shift left	LSL Rd, Rn, Rs	1
	Logical shift right	LSR Rd, Rn, #<imm>	1
	Logical shift right	LSR Rd, Rn, Rs	1
	Arithmetic shift right	ASR Rd, Rn, #<imm>	1
	Arithmetic shift right	ASR Rd, Rn, Rs	1

Table 3-1 Cortex-M3 instruction set summary (continued)

Operation	Description	Assembler	Cycles
Rotate	Rotate right	`ROR Rd, Rn, #<imm>`	1
	Rotate right	`ROR Rd, Rn, Rs`	1
	With extension	`RRX Rd, Rn`	1
Count	Leading zeroes	`CLZ Rd, Rn`	1
Load	Word	`LDR Rd, [Rn, <op2>]`	2[c]
	To PC	`LDR PC, [Rn, <op2>]`	2[c] + P
	Halfword	`LDRH Rd, [Rn, <op2>]`	2[c]
	Byte	`LDRB Rd, [Rn, <op2>]`	2[c]
	Signed halfword	`LDRSH Rd, [Rn, <op2>]`	2[c]
	Signed byte	`LDRSB Rd, [Rn, <op2>]`	2[c]
	User word	`LDRT Rd, [Rn, #<imm>]`	2[c]
	User halfword	`LDRHT Rd, [Rn, #<imm>]`	2[c]
	User byte	`LDRBT Rd, [Rn, #<imm>]`	2[c]
	User signed halfword	`LDRSHT Rd, [Rn, #<imm>]`	2[c]
	User signed byte	`LDRSBT Rd, [Rn, #<imm>]`	2[c]
	PC relative	`LDR Rd,[PC, #<imm>]`	2[c]
	Doubleword	`LDRD Rd, Rd, [Rn, #<imm>]`	1 + N
	Multiple	`LDM Rn, {<reglist>}`	1 + N
	Multiple including PC	`LDM Rn, {<reglist>, PC}`	1 + N + P
Store	Word	`STR Rd, [Rn, <op2>]`	2[c]
	Halfword	`STRH Rd, [Rn, <op2>]`	2[c]
	Byte	`STRB Rd, [Rn, <op2>]`	2[c]
	Signed halfword	`STRSH Rd, [Rn, <op2>]`	2[c]
	Signed byte	`STRSB Rd, [Rn, <op2>]`	2[c]
	User word	`STRT Rd, [Rn, #<imm>]`	2[c]
	User halfword	`STRHT Rd, [Rn, #<imm>]`	2[c]
	User byte	`STRBT Rd, [Rn, #<imm>]`	2[c]
	User signed halfword	`STRSHT Rd, [Rn, #<imm>]`	2[c]
	User signed byte	`STRSBT Rd, [Rn, #<imm>]`	2[c]
	Doubleword	`STRD Rd, Rd, [Rn, #<imm>]`	1 + N

Table 3-1 Cortex-M3 instruction set summary (continued)

Operation	Description	Assembler	Cycles
Push	Push	PUSH {<reglist>}	1 + N
	Push with link register	PUSH {<reglist>, LR}	1 + N
Pop	Pop	POP {<reglist>}	1 + N
	Pop and return	POP {<reglist>, PC}	1 + N + P
Semaphore	Load exclusive	LDREX Rd, [Rn, #<imm>]	2
	Load exclusive half	LDREXH Rd, [Rn]	2
	Load exclusive byte	LDREXB Rd, [Rn]	2
	Store exclusive	STREX Rd, Rt, [Rn, #<imm>]	2
	Store exclusive half	STREXH Rd, Rt, [Rn]	2
	Store exclusive byte	STREXB Rd, Rt, [Rn]	2
	Clear exclusive monitor	CLREX	1
Branch	Conditional	B<cc> <label>	1 or 1 + P[d]
	Unconditional	B <label>	1 + P
	With link	BL <label>	1 + P
	With exchange	BX Rm	1 + P
	With link and exchange	BLX Rm	1 + P
	Branch if zero	CBZ Rn, <label>	1 or 1 + P[d]
	Branch if non-zero	CBNZ Rn, <label>	1 or 1 + P[d]
	Byte table branch	TBB [Rn, Rm]	2 + P
	Halfword table branch	TBH [Rn, Rm, LSL#1]	2 + P
State change	Supervisor call	SVC #<imm>	-
	If-then-else	IT... <cond>	1[e]
	Disable interrupts	CPSID <flags>	1 or 2
	Enable interrupts	CPSIE <flags>	1 or 2
	Read special register	MRS Rd, <specreg>	1 or 2
	Write special register	MSR <specreg>, Rn	1 or 2
	Breakpoint	BKPT #<imm>	-
Extend	Signed halfword to word	SXTH Rd, <op2>	1
	Signed byte to word	SXTB Rd, <op2>	1
	Unsigned halfword	UXTH Rd, <op2>	1
	Unsigned byte	UXTB Rd, <op2>	1

Table 3-1 Cortex-M3 instruction set summary (continued)

Operation	Description	Assembler	Cycles
Bit field	Extract unsigned	`UBFX Rd, Rn, #<imm>, #<imm>`	1
	Extract signed	`SBFX Rd, Rn, #<imm>, #<imm>`	1
	Clear	`BFC Rd, Rn, #<imm>, #<imm>`	1
	Insert	`BFI Rd, Rn, #<imm>, #<imm>`	1
Reverse	Bytes in word	`REV Rd, Rm`	1
	Bytes in both halfwords	`REV16 Rd, Rm`	1
	Signed bottom halfword	`REVSH Rd, Rm`	1
	Bits in word	`RBIT Rd, Rm`	1
Hint	Send event	`SEV`	1
	Wait for event	`WFE`	1 + W
	Wait for interrupt	`WFI`	1 + W
	No operation	`NOP`	1
Barriers	Instruction synchronization	`ISB`	1 + B
	Data memory	`DMB`	1 + B
	Data synchronization	`DSB <flags>`	1 + B

a. `UMULL`, `SMULL`, `UMLAL`, and `SMLAL` instructions use early termination depending on the size of the source values. These are interruptible, that is abandoned and restarted, with worst case latency of one cycle.
b. Division operations use early termination to minimize the number of cycles required based on the number of leading ones and zeroes in the input operands.
c. Neighboring load and store single instructions can pipeline their address and data phases. This enables these instructions to complete in a single execution cycle.
d. Conditional branch completes in a single cycle if the branch is not taken.
e. An `IT` instruction can be folded onto a preceding 16-bit Thumb instruction, enabling execution in zero cycles.

Stati Operativi:

_Thumb state: stato in cui il processore si trova durante la normale esecuzione del codice;

_Debug state: stato in cui il processore si trova durante il debug.

_*Modi d'esecuzione*:

_Thread mode: il processore entra in questa modalità al reset e può rientrarci come risultato di ritorno di un'eccezione;

_ Handler mode: il processore entra in questa modalità a seguito di un'eccezione.

_*Rappresentazione dei dati* in formato Big endian o little endian (le istruzioni sono sempre little endian); *tipi di dati* supportati:word (32 bit), half word e byte;

_ Registri: sono gli stessi del Processore Cortex M1.

CARATTERISTICHE TECNICHE E PRESTAZIONI

Cortex-M3 Features	
Architecture	ARMv7-M (Harvard)
ISA Support	Thumb® / Thumb-2
Pipeline	3-stage + branch speculation
Dhrystone	1.25 DMIPS/MHz
Memory Protection	Optional 8 region MPU with sub regions and background region
Interrupts	Non-maskable Interrupt (NMI) + 1 to 240 physical interrupts
Interrupt Latency	12 cycles
Inter-Interrupt Latency	6 cycles
Interrupt Priority Levels	8 to 256 priority levels
Wake-up Interrupt Controller	Up to 240 Wake-up Interrupts
Sleep Modes	Integrated WFI and WFE Instructions and Sleep On Exit capability. Sleep & Deep Sleep Signals. Optional Retention Mode with ARM Power Management Kit
Bit Manipulation	Integrated Instructions & Bit Banding
Enhanced Instructions	Hardware Divide (2-12 Cycles) & Single-Cycle (32x32) Multiply.
Debug	Optional JTAG & Serial-Wire Debug Ports. Up to 8 Breakpoints and 4 Watchpoints.
Trace	Optional Instruction Trace (ETM), Data Trace (DWT), and Instrumentation Trace (ITM)

Per analizzare le prestazioni consideriamo il processore Cortex M3 con tecnologia TSMC 180nm G e TSMC 90nm G.

Cortex-M3 Performance, Power & Area				
Process	**TSMC 180nm G**		**TSMC 90nm G**	
Optimization Type	Speed Optimized	Area Optimized	Speed Optimized	Area Optimized
Standard Cell Library	ARM SC7	ARM SC7	ARM SC9	ARM SC9
Performance (Total DMIPS)	125	75	340	75
Frequency (MHz)	100	50	275	50
Power Efficiency (DMIPS/mW)	3.75	6.25	TBD	12.5
Area (mm^2)	0.37	0.25	0.083	0.047

Applicazioni D'Uso.

I processori Cortex M3 sono impiegati principalmente in diversi FPGA di diverse case quali Actel, STMicroelectronics,NXP,e ATMEL,Sitara.Questi diversi FPGA sono implementati su dispositivi embedded, in generale su piattaforme a basso costo e in grado di fornire un alto rendimento per le seguenti tipologie di applicazioni:

- Microcontrollori: l'esempio nella figura seguente mostra il microcontrollore *LTC3588* che è stato messo assieme ad un processore Cortex M3 realizzando un dispositivo di monitoraggio dell'energia. Questo dispositivo può essere impiegato nei sistemi ad energia fotovoltaica o termica.

Il microcontrollore ARM Cortex-M3 a 32 bit e l'LTC3588 di Linear Technology sono stati messi insieme per raggiungere un obiettivo comune: la raccolta di energia per far funzionare sistemi a basso consumo. L'MCU monitora i livelli di energia, per garantire in qualsiasi momento che il consumo totale del circuito rimanga al di sotto dei livelli di energia raccolta.

- Dispositivi di controllo del motore: il processore Cortex M3 è impiegato in tutti i dispositivi SmartFusion della Actel come quelli riportati nelle figure di seguito.

SmartFusion Motor Control Development Kit

Miniature Motor Control Daughter Card

TRINAMIC TMCM-AC-840 Motor Control Daughter Board Kit

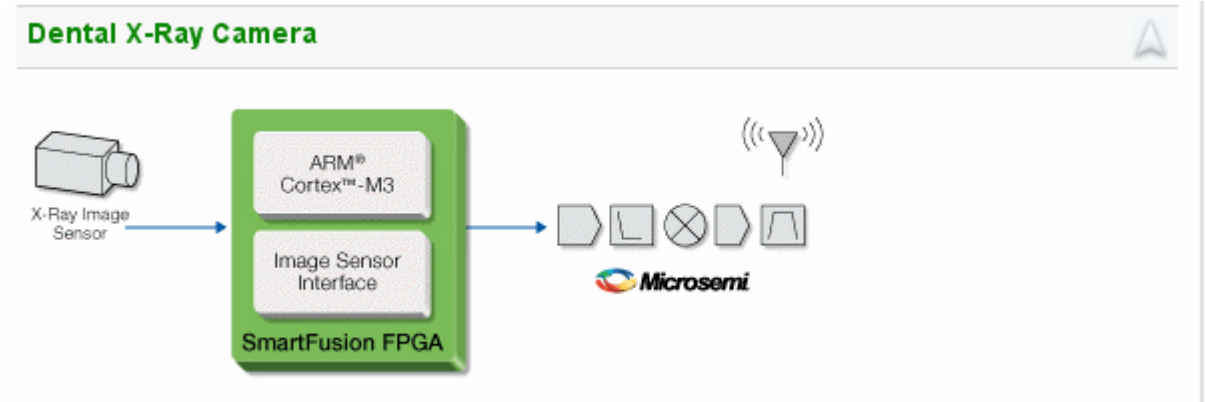
Smartfusion Motor Control development kit, e Trinamic TMC-AC-840 sono utilizzati nei settori delle automobili, nei settori industriali, nella robotica(per il controllo del movimento del braccio di un robot).Mentre Miniature Motor Control Daughter Card in tutti i dispositivi portatili nell'ambito Medico.

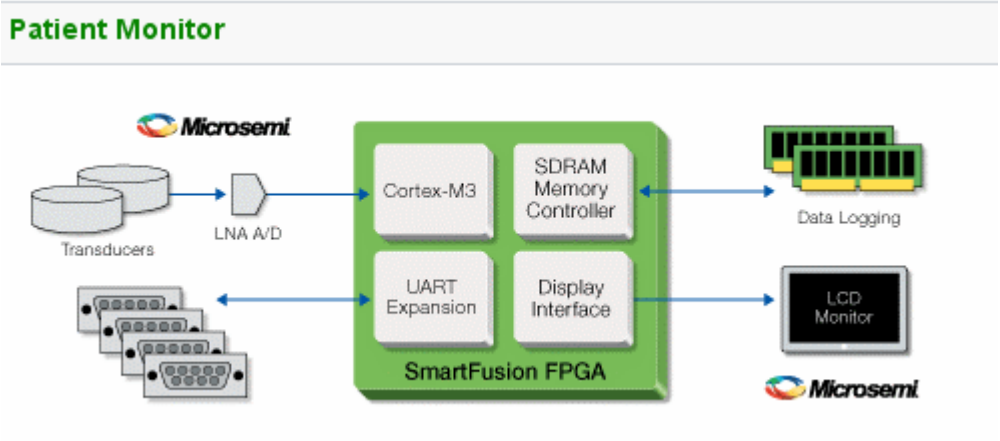
- Dispositivi di monitoraggio della potenza(Power Management): per esempio la combinazione del processore Cortex M3 + Actel SmartFusion Evaluation Kit+ MPM Daughter Card il tutto riportato nella figura seguente:

- Dispositivi Medici: ad esempio la telecamera intraorale(Dental X-Ray Camera, utilizzata per trasmettere le immagini del cavo orale nel computer),e il Patient Monitor(monitoraggio dei segni vitali del paziente).

Patient Monitor

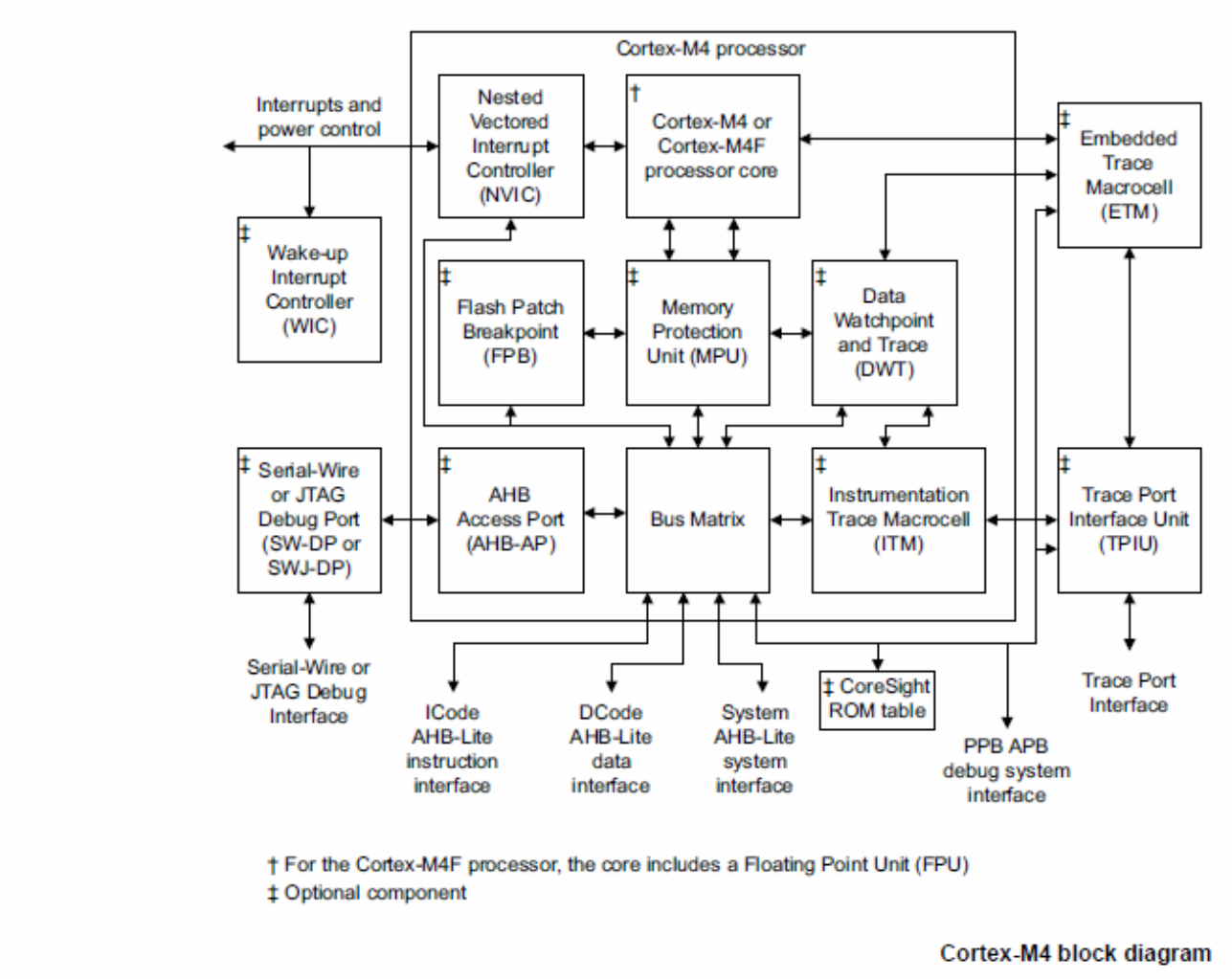
Telecamera intraorale





- Nel settore automobistico(vedi dispositivi precedenti più quelli implementati nel processore Cortex M1),e nel settore aeronautico/militare(Flight computers, mission computers, weapon systems, radar control systems).
- Nel Settore Industriale e di consumo: Radio portatile,Lettore RFID,Scanner,Tester,Lettore Codice a Barre.

ARM CORTEX M4



Cortex-M4 block diagram

Il Diagramma a blocchi è lo stesso del Processore Cortex M3. Le unità FPB,DWT,MPU,NVIC sono le stesse viste nel Cortex M3, con caratteristiche uguali, tabelle uguali, stessi registri e stessi indirizzi di memoria.L'unica differenza è che può essere implementata una Floating Point Unit nel processore e da qui ne deriva il nome di Cortex M4F.

CARATTERISTICHE DEL PROCESSORE

Implementa l'architettura ARMv7-M, inoltre i registri, stato operativo, modi d'esecuzione,tipi di dati,rappresentazione dei dati sono gli stessi del processore Cortex M3.Per quanto riguarda il set di istruzioni è lo stesso visto nel processore Cortex M3 se viene implementato un processore senza la Floating point unit FPU. Se viene implementata la FPU troviamo il set di istruzioni del Cortex M3 più tutte le istruzioni previste dalla FPU.
La FPU fornisce le seguenti caratteristiche:
_Istruzioni a 32 bit per operazioni a singola precisione su dati in virgola mobile con o senza segno(signed/unsigned);
_Istruzioni di moltiplicazione(Multiply) e accumulazione(Accumulate) fuse in un'unica istruzione per operazioni ad alta precisione;
_Fornisce le conversioni da integer a float e viceversa e le conversioni tra fixed-point e floating point;
_Contiene 32 registri a singola precisione che possono essere visti come:16 registri doubleword da 64 bit(D0-D15) oppure 32 registri di word a 32 bit (S0-S31).

FPU instruction set

Operation	Description	Assembler	Cycles
Absolute value	of float	VABS.F32	1
Addition	floating point	VADD.F32	1
Compare	float with register or zero	VCMP.F32	1
	float with register or zero	VCMPE.F32	1
Convert	between integer, fixed-point, half-precision and float	VCVT.F32	1
Divide	Floating-point	VDIV.F32	14
Load	multiple doubles	VLDM.64	1+2*N, where N is the number of doubles.
	multiple floats	VLDM.32	1+N, where N is the number of floats.
	single double	VLDR.64	3
	single float	VLDR.32	2

FPU instruction set (continued)

Operation	Description	Assembler	Cycles
Move	top/bottom half of double to/from core register	VMOV	1
	immediate/float to float-register	VMOV	1
	two floats/one double to/from two core registers or one float to/from one core register	VMOV	2
	floating-point control/status to core register	VMRS	1
	core register to floating-point control/status	VMSR	1
Multiply	float	VMUL.F32	1
	then accumulate float	VMLA.F32	3
	then subtract float	VMLS.F32	3
	then accumulate then negate float	VNMLA.F32	3
	then subtract then negate float	VNMLS.F32	3
Multiply (fused)	then accumulate float	VFMA.F32	3
	then subtract float	VFMS.F32	3
	then accumulate then negate float	VFNMA.F32	3
	then subtract then negate float	VFNMS.F32	3
Negate	float	VNEG.F32	1
	and multiply float	VNMUL.F32	1
Pop	double registers from stack	VPOP.64	1+2*N, where N is the number of double registers.
	float registers from stack	VPOP.32	1+N where N is the number of registers.
Push	double registers to stack	VPUSH.64	1+2*N, where N is the number of double registers.
	float registers to stack	VPUSH.32	1+N, where N is the number of registers.
Square-root	of float	VSQRT.F32	14
Store	multiple double registers	VSTM.64	1+2*N, where N is the number of doubles.
	multiple float registers	VSTM.32	1+N, where N is the number of floats.
	single double register	VSTR.64	3
	single float registers	VSTR.32	2
Subtract	float	VSUB.F32	1

CARATTERISTICHE TECNICHE E PRESTAZIONI

Cortex-M4 Features	
Architecture	ARMv7E-M (Harvard)
ISA Support	Thumb® / Thumb-2
DSP Extensions	Single cycle 16,32-bit MAC Single cycle dual 16-bit MAC 8,16-bit SIMD arithmetic Hardware Divide (2-12 Cycles)
Floating Point Unit	Single precision floating point unit IEEE 754 compliant
Pipeline	3-stage + branch speculation
Dhrystone	1.25 DMIPS/MHz
Memory Protection	Optional 8 region MPU with sub regions and background region
Interrupts	Non-maskable Interrupt (NMI) + 1 to 240 physical interrupts
Interrupt Latency	12 cycles
Inter-Interrupt Latency	6 cycles
Interrupt Priority Levels	8 to 256 priority levels
Wake-up Interrupt Controller	Up to 240 Wake-up Interrupts
Sleep Modes	Integrated WFI and WFE Instructions and Sleep On Exit capability. Sleep & Deep Sleep Signals. Optional Retention Mode with ARM Power Management Kit
Bit Manipulation	Integrated Instructions & Bit Banding
Debug	Optional JTAG & Serial-Wire Debug Ports. Up to 8 Breakpoints and 4 Watchpoints.
Trace	Optional Instruction Trace (ETM), Data Trace (DWT), and Instrumentation Trace (ITM)

Cortex-M4 Performance, Power & Area		
Process	**65nm low power process**	
Optimization Type	Speed Optimized	Area Optimized
Standard Cell Library	ARM SC12	ARM SC9
Performance (Total DMIPS)	375	185
Frequency (MHz)	300	150
Power Efficiency (DMIPS/mW)	24	38
Area (mm^2)	0.21	0.11
FPU Area (if included) (mm^2)	0.08	0.06

Si nota che rispetto al Cortex M3 abbiamo una maggiore frequenza 300 Mhz contro i 275 Mhz del M3, e una maggiore efficienza energetica(Power Efficiency) 38 DMIPS/mw contro i 12.5 DMIPS/mw del M3.Il valore della Dhrystone invece è lo stesso.

Applicazioni D'uso

ARM Cortex™-M4 è il core nato come evoluzione del core ARM Cortex™-M3. Il core M4 offre prestazioni avanzate di controllo digitale e quindi ideale per applicazioni DSC, ovvero applicazioni di controllo ed elaborazione del segnale.Basso assorbimento, bassi costi ed alta efficienza lo porteranno ad essere un core usato nel **controllo motori**, **nel settore automobilistico**, nella **gestione di potenza**, nelle **applicazioni audio** e **nell'automazione industriale**(Robotica, ed elettrodomestici quali lavatrici, condizionatori, ventilatori , ecc). Il core M4 è in produzione nei microcontrollori Freescale con il progetto Kinetis (che prevede più di 7 famiglie di microcontrollori e più di 200 dispositivi compatibili).Un esempio di microcontrollore Freescale con processore Cortex M4 è il modello TWR-K60N512-KIT riportato nella figura seguente ed utilizzato nelle piattaforme di sviluppo di applicazioni di controllo:

Anche NXP utilizza il Cortex M4 nello sviluppo dei suoi microcontrollori. Nella figura seguente è riportato un dispositivo di controllo del motore realizzato con il microcontrollore NXP LPC4300 avente due processori: un Cortex M4 e un Cortex M0.

6.Conclusioni

Osservando le prestazioni di ogni processore si nota che la famiglia di processori Cortex A è quella con le più elevate prestazioni:confrontando una versione ad un solo core, nella famiglia dei Cortex A il processore con le maggiori performance è il Cortex A9 con 2,50 DMIPS/MHZ per core contro l'1,030 DMIPS/MHZ dei Cortex R4 e R5 e l'1,25DMIPS/MHZ del Cortex M4(la migliore versione in prestazioni della famiglia Cortex M).Inoltre i processori Cortex A5 e A9 si possono presentare oltre che nella versione single-core, in modalità Dual Core e Quad Core. Dei restanti processori solo il processore Cortex R5 può essere implementato in versione Multicore (single o Dual Core). Ne consegue che la famiglia di Processori Cortex A sono ampiamente utilizzati nei dispositivi di ultima generazione ad elevate prestazioni: dagli Smartphone, Iphone, IPAD,Netbook sino ai server di ultima generazione. Mentre I Cortex R ed M sono ampiamente richiesti nei semiconduttori, circuiti integrati e microcontrollori di diverse case produttrici. Questi semiconduttori e microcontrollori vengono poi implementati in svariate applicazioni : Settore automobilistico, Settore Industriale e dell'automazione, Settori dell'elettronica di consumo,Settore Medico, Settore Aeronautico/Militare. Riguardo invece i consumi energetici, osservando il valore della potenza di ogni processore in configurazione single core e nelle migliori prestazioni: Il Cortex A5 è il processore con il minor consumo energetico (ha un valore di potenza <0.08mw/MHZ). Se consideriamo invece i Cortex A5 e A9 in configurazione Dual o Quad Core i consumi energetici sono nettamente superiori ai Processori Cortex R e M.

7. Bibliografia

Di seguito riportate le fonti da cui ho attinto per la composizione di questo documento.

- Multiprocessori:
  www.di.unito.it/~gunetti/DIDATTICA/.../07-multiprocessors-1.pdf
  http://it.wikipedia.org/wiki/Multiprocessore
- Introduzione agli ARM Cortex:
  http://it.wikipedia.org/wiki/ARM_Cortex
  http://www.appuntidigitali.it/5037/cortex-a8-una-nuova-via-per-arm/
- Diagramma a Blocchi e caratteristiche di ogni processore attraverso i Manuali ARM CORTEX reperiti nel sito seguente:
  http://infocenter.arm.com/help/index.jsp?noscript=1
- Caratteristiche tecniche e prestazioni di ogni processore:
  http://www.arm.com/products/processors/cortex-a/cortex-a5.php
  http://www.arm.com/products/processors/cortex-a/cortex-a8.php
  http://www.arm.com/products/processors/cortex-a/cortex-a9.php
  http://www.arm.com/products/processors/cortex-r/cortex-r4.php
  http://www.arm.com/products/processors/cortex-r/cortex-r5.php
  http://www.arm.com/products/processors/cortex-m/cortex-m0.php
  http://www.arm.com/products/processors/cortex-m/cortex-m1.php
  http://www.arm.com/products/processors/cortex-m/cortex-m3.php
  http://www.arm.com/products/processors/cortex-m/cortex-m4-processor.php
- Applicazioni d'uso per ogni processore:
  http://it.emcelettronica.com/arm-cortex-m3-mcu-e-riduzione-del-consumo-di-energia
  http://www.actel.com/products/mpu/cortexm3/default.aspx#features
  http://www.actel.com/products/mpu/cortexm1/default.aspx
  http://www.actel.com/products/solutions/portable/default.aspx
  http://www.actel.com/products/solutions/industrial/default.aspx
  http://www.actel.com/products/solutions/medical/default.aspx
  http://www.actel.com/products/solutions/milaero/default.aspx
  http://www.actel.com/products/solutions/auto/default.aspx
  http://www.arm.com/products/processors/cortex-m/cortex-m4-processor.php
  http://www.arm.com/products/processors/cortex-m/cortex-m3.php
  http://www.arm.com/products/processors/cortex-m/cortex-m1.php
  http://www.arm.com/products/processors/cortex-m/cortex-m0.php
  http://www.arm.com/products/processors/cortex-r/cortex-r4.php
  http://www.arm.com/products/processors/cortex-r/cortex-r5.php
  http://www.arm.com/products/processors/cortex-a/cortex-a5.php
  http://www.arm.com/products/processors/cortex-a/cortex-a8.php
  http://www.arm.com/products/processors/cortex-a/cortex-a9.php

http://www.nxp.com/
http://www.freescale.com/
http://www.st.com/internet/com/home/home.jsp
http://www.elettronicanews.it/articoli/0,1254,40_ART_2778,00.html
http://www.ti.com/ww/it/embedded/arm/index.html
http://netbookitalia.it/rim-playbook-da-64gb.html
http://telephonino.altervista.org/sony-psp-2-hardware-piu-potente-di-iphone-5-secondo-le-indiscrezioni/