**Quali sono i concetti fondamentali di un processore MIPS?**

Il processore MIPS utilizza un set di istruzioni di tipo RISC (istruzioni semplici offrendo elevate prestazioni). Ciascuna operazione aritmetica MIPS esegue solo un’operazione e deve contenere esattamente tre variabili. Il processore MIPS non riesce ad operare aritmeticamente direttamente nella memoria ma utilizza dei registi da tramite. Gli operandi delle istruzioni aritmetiche del MIPS devono obbedire ad alcune restrizioni, devono essere scelti tra un numero limitato di locazioni particolari chiamate registri. Il MIPS ha 32 registri a 32 bit.

Le istruzioni aritmetiche del MIPS nello stesso ciclo di clock possono leggere due registri, eseguire l’operazione e scrivere il risultato.

Le istruzioni di trasferimento dati del MIPS, invece possono leggere e scrivere un solo operando, senza potere eseguire nessuna operazione su di esso. Nel MIPS le parole devono iniziare sempre a indirizzi multipli di 4, questo requisito si chiama vicolo di allineamento. Nel posizionamento in memoria dei dati il MIPS utilizza il posizionamento Big Endian.

Nel MIPS le istruzioni hanno tutte la stessa lunghezza però esistono formati differenti di istruzioni.

Formato di tipo R è utilizzato per istruzioni aritmetiche e logiche, è composta da 5 campi,

* op: codice operativo
* rs: primo operando sorgente (registro)
* rt: secondo operando sorgente (registro)
* rd: registro destinazione
* shamt: shift amount (per operazioni di scalamento)
* funct: seleziona una variante specifica dell'operazione base definita nel campo op.

Formato I utilizzato per le istruzioni di trasferimento contiene quattro campi codice operativo, due operandi e indirizzo.

Formato J utilizzato per l’istruzione jump, contiene solo due campi codice operativo e indirizzo di salto.

**Cosa differenzia ARM con MIPS?**

A differenza del MIPS, l’ARM non riserva un registro al valore 0.

L’ARM utilizza le condition code (cosa che un processore RISC non dovrebbe fare), nei processori tradizionali (tipo X86) il risultato di un istruzione di confronto non va a finire in un registro con nel MIPS ma quando si effettuano dei confronti nei processori tradizionali si settano dei bit in una word (word detta word di stato del processore che contiene lo stato interno del processore) quindi ogni volta che viene fatto un confronto vengono settati dei bit che ci dicono se il risultato del confronto era uguale, diverso, minore o maggiore a zero e il risultato viene codificato in una parola di stato (che è un registro interno al processore in cui ci sono tutte le informazioni di tutto quello che avviene internamente). In particolare questi bit che esprimono le condizioni minore, maggiore, uguale e diverso vengono detti condition code. Quindi i condition code sono dei bit interni alle parole di stato del processore.

Utilizzare i condition code all’interno di un processore moderno non è una grande idea, perché se in un processore moderno riesco a fare 2-3 operazioni per volta nel senso che dispongo di 2-3 ALU quindi riesco a fare più somme ognuna di queste tenterà di modificare i condition code e quindi un unico set di condition code non andrà bene, è meglio mettere il risultato ogni volta in un registro diverso e in questa maniera riesco a far andare completamente in parallelo questa cosa.

Ottima cosa che possiede l’ARM a differenza del MIPS è che ogni istruzione può essere condizionale, significa che l’esecuzione di un’istruzione dipende dal risultato di un confronto. Serve ad evitare i branch. Evitare i branch è una grandissima cosa perché evita al processore di fare previsioni.

Il MIPS ha solamente 3 semplici modalità di indirizzamento dei dati l’ARM ne ha 9 di cui alcune ottenute tramite dei calcoli complessi.

L’ARM ha 16 registri mentre il MIPS 32 registri.

**Differenza tra processore MIPS e X86?**

Nei processori X86 le istruzioni hanno sempre un operando che funge sia da sorgente che da destinazione.

Questa restrizione rende complesso il problema del numero limitato dei registri dato che il contenuto di uno dei registri sorgente viene necessariamente modificato.

MIPS ed ARM consentono di definire registri distinti per operandi di sorgente e destinazione.

Uno degli operandi può essere contenuto in memoria perciò tutte le istruzioni possono prendere uno dei due operandi dalla memoria a differenza di ARM e MIPS.

I processori X86 sono più difficili da implementare rispetto a MIPS ed ARM ma l’ampiezza del mercato delle loro CPU permettono di avere più risorse da investire per risolvere la loro complessità.

Inoltre i componenti architetturali di X86 utilizzati più frequentemente sono più facili da implementare.

**Istruzione che dura più tempo nel MIPS?**

Load.