


Architettura degli elaboratori - II
Introduzione

Prof. Alberto Borghese
Dipartimento di Informatica
borghese@di.unimi.it

Università degli Studi di Milano

Riferimento sul Patterson: capitolo 4.2 , 4.4, D1, D2.


Introduzione alla CPU

- Introduzione
- La CPU a ciclo singolo

Architetture II (6cfu)

Docente: Prof. N. Alberto Borghese: borghese@di.unimi.it
Laboratorio Assembler:
Dott. Iuri Frosio: frosio@di.unimi.it
Dott. Massimo Marchi: marchi@di.unimi.it

Orario e aule:

Lunedì	Ore 8.30-10.30	Aula G10, Via Celoria 20
Giovedì	Ore 8.30-10.30	Aula 301, Via Celoria 20
Venerdì	**Ore 10.30-12.30**	**Aula V307, Via Celoria 20 (Cognomi A-F)**
Venerdì	**Ore 10.30-12.30**	**Aula V309, Via Celoria 20 (Cognomi G-Z)**

Orario di ricevimento: su appuntamento.

Strumento principale di contatto: email!

Programma

Sito principale:
http://borghese.dsi.unimi.it/Teaching/Architettura_II/_Arch_II.html

Programma:
http://borghese.dsi.unimi.it/Teaching/Architettura_II/Programma_2012-2013.html

Argomenti principali:
- CPU (avanzate)
- Gerarchie di memoria
- Interconnessioni

Esame

Parte teorica (2/3 del voto).

- Prova scritta + orale. Appelli ogni 1 / 2 / 3 mesi, al di fuori dal periodo delle lezioni.
- 2 compitini in itinere durante l'anno. I compitini sostituiscono interamente scritto e orale.
- Per superare la parte di teoria con i compitini occorre avere preso almeno 17 in tutti e due i compitini e che la media dei 2 compitini sia ≥ 18. I compitini sono consigliati solo a chi frequenta.
- L'orale con i compitini è facoltativo.

Progetto di laboratorio in Assembler (PC-Spim, 1/3 del voto).

Materiale didattico

See web page

http://borghese.dsi.unimi.it/Teaching/Architettura_II/References.rtf

Testo di base (è disponibile sia in inglese che in italiano):
Struttura e progetto dei calcolatori: l'interfaccia hardware-software, D.A. Patterson and J.L. Hennessy, Terza edizione, Zanichelli, estate 2010 (Nota: la terza edizione Zanichelli è la traduzione della quarta edizione inglese).

"Computer Organization & Design: The Hardware/Software Interface", D.A. Patterson and J.L. Hennessy, Morgan Kaufmann Publishers, Revised Fourth Edition, 2012.

Potete trovare esercizi del testo svolti al seguente URL: http://books.elsevier.com/companions/1558606041/.

Obbiettivo di un'architettura

Elabora in modo adeguato un input per produrre l'output.

- Le unità di *ingresso* (tastiera, mouse, rete, interfacce con dispositivi di acquisizione, ecc.) permettono al calcolatore di acquisire informazioni dall'ambiente esterno.
- L'architettura di elaborazione.

- Le unità di *uscita* (terminale grafico, stampanti, rete, ecc.) consentono al calcolatore di comunicare i risultati ottenuti dall'elaborazione all'ambiente esterno.

Cosa fa un elaboratore?

- Algoritmi (sequenza di istruzioni).
  Calcoli (calcolatore).
  Operazioni logiche (elaboratore).
- Programma (Ada Lovelace, 1830) = *Algoritmi in Software.*

Come lo fa? *Hardware*.

Input ==> Elaborazione ==> Output

- Terza rivoluzione della nostra civiltà: la rivoluzione agricola, la rivoluzione industriale e la rivoluzione dell'informatica.

CPU di tipo CISC (*Complex Instruction Set Computer*)

- Caratterizzate da elevata complessità delle istruzioni eseguibili ed elevato numero di istruzioni che costituiscono l'insieme delle istruzioni.
- Numerose modalità di indirizzamento per gli **operandi** dell'*ALU* che possono provenire da registri oppure da memoria, nel qual caso l'indirizzamento può essere diretto, indiretto, con registro base, ecc.
- Dimensione *variabile* delle istruzioni a seconda della modalità di indirizzamento di ogni operando ⇒ complessità di gestione della fase di prelievo o *fetch* in quanto a priori non è nota la lunghezza dell'istruzione da caricare.
- Elevata complessità della *CPU* stessa (cioè dell'hardware relativo) in termini degli elementi che la compongono con la conseguenza di rallentare i tempi di esecuzione delle operazioni. Elevata profondità dell'albero delle porte logiche, utilizzato per la decodifica.

Utilizzo architettura Intel 80x86: le 10 istruzioni più frequenti

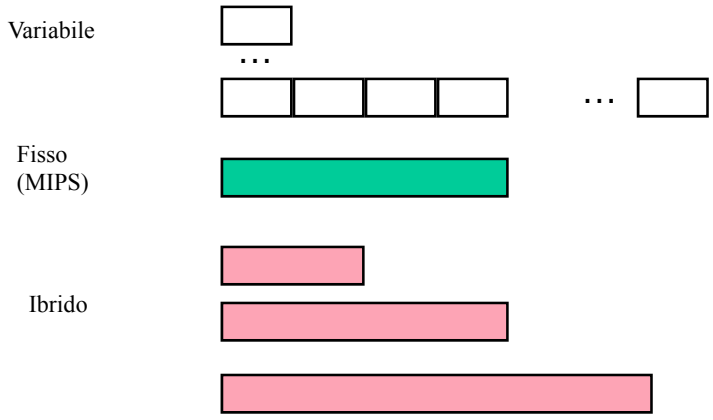
Rank	instruction	Integer Average Percent total executed
1	load	22%
2	conditional branch	20%
3	compare	16%
4	store	12%
5	add	8%
6	and	6%
7	sub	5%
8	move register-register	4%
9	call	1%
10	return	1%
	Total	96%

- **Simple instructions dominate instruction frequency** => RISC

Architetture di tipo RISC (*Reduced Instruction Set Computer*)

- Ispirate al principio di eseguire soltanto istruzioni semplici: le operazioni complesse vengono scomposte in una serie di istruzioni più semplici da eseguire in un ciclo base ridotto, con l'obiettivo di migliorare le prestazioni ottenibili dalle *CPU CISC*.
- Caratterizzate da istruzioni molto semplificate.
- Gli operandi dell'*ALU* possono provenire dai registri ma *non* dalla memoria. Per il trasferimento dei dati da memoria ai registri e viceversa si utilizzano delle apposite operazioni di caricamento (*load*) e di memorizzazione (*store)*
  ⇒ *architetture load/store*.
- *CPU* relativamente semplice ⇒ si riducono i tempi di esecuzione delle singole istruzioni, che sono però meno potenti delle istruzioni *CISC*.
- Dimensione *fissa* delle istruzioni ⇒ più semplice la gestione della fase di prelievo *(fetch)* e della codifica delle istruzioni da eseguire

I diversi formati di istruzioni

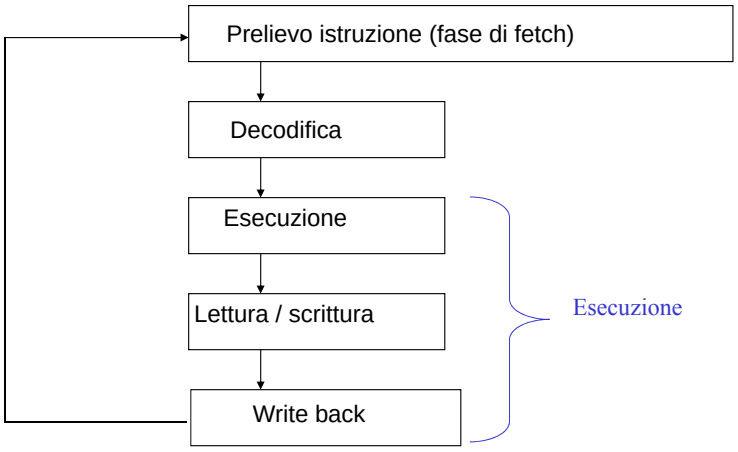


Il formato fisso consente di massimizzare la velocità, il formato ibrido consente di minimizzare la lunghezza del codice.

Architettura MIPS

- Architettura MIPS appartiene alla famiglia delle architetture **RISC (Reduced Instruction Set Computer)** sviluppate dal 1980 in poi
  - Esempi: Sun Sparc, HP PA-RISC, IBM Power PC, DEC Alpha, Silicon Graphics, AIBO-Sony, ARM.
- Principali obiettivi delle architetture RISC:
  - Semplificare la progettazione dell'hardware e del compilatore
  - Massimizzare le prestazioni
  - Minimizzare i costi

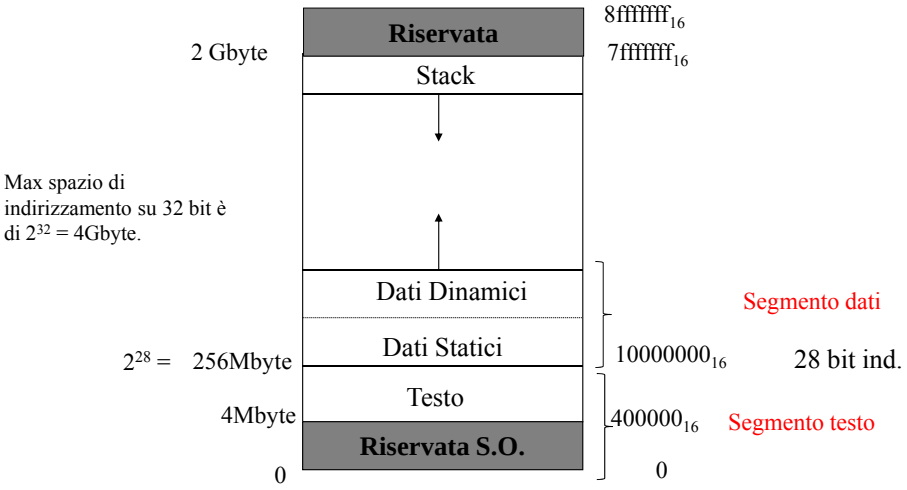
Ciclo di esecuzione di un'istruzione MIPS



Tipi di istruzioni

- Le istruzioni comprese nel linguaggio macchina di ogni calcolatore possono essere classificate nelle seguenti quattro categorie:
  - Istruzioni aritmetico-logiche;
  - Istruzioni di trasferimento da/verso la memoria *(load/store);*
  - Istruzioni di salto condizionato e non condizionato per il controllo del flusso di programma;
  - Istruzioni di trasferimento in ingresso/uscita (I/O).

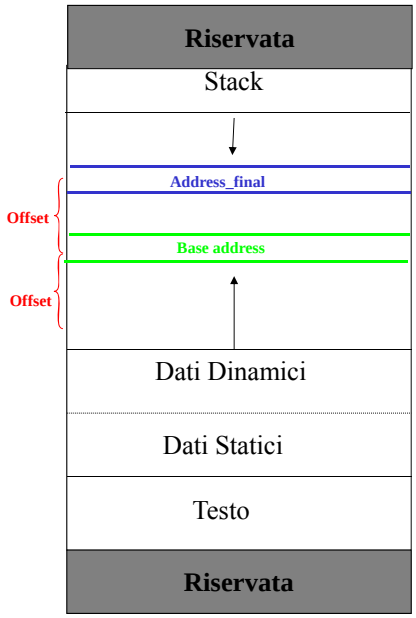
Organizzazione logica della memoria



Indirizzamento della memoria

Base + spiazzamento
Base + Offset

Address_final = Base_address + Offset



Istruzioni di trasferimento dati

- Gli operandi di una istruzione aritmetica devono risiedere nei registri che sono in numero limitato (32 nel MIPS). I programmi in genere richiedono un numero maggiore di variabili.
- Cosa succede ai programmi i cui dati richiedono più di 32 registri (32 variabili)?

  Alcuni dati risiedono in memoria.
- La tecnica di mettere le variabili meno usate (o usate successivamente) in memoria viene chiamata ***Register Spilling***.

Servono istruzioni apposite per trasferire dati da memoria a registri e viceversa

MIPS: Software conventions for Registers

0	zero	constant 0
1	at	reserved for assembler
2	v0	expression evaluation &
3	v1	function results
4	a0	arguments
5	a1	
6	a2	
7	a3	
8	t0	temporary: caller saves
. . .		(callee can clobber)
15	t7	
16	s0	callee saves
. . .	(caller can clobber)	
23	s7	
24	t8	temporary (cont'd)
25	t9	
26	k0	reserved for OS kernel
27	k1	
28	gp	Pointer to global area
29	sp	Stack pointer
30	fp	frame pointer (s8)
31	ra	Return Address (HW)

Tipi di istruzioni

- Le istruzioni comprese nel linguaggio macchina di ogni calcolatore possono essere classificate nelle seguenti quattro categorie:
  - Istruzioni aritmetico-logiche;
  - Istruzioni di trasferimento da/verso la memoria *(load/store);*
  - Istruzioni di salto condizionato e non condizionato per il controllo del flusso di programma;
  - Istruzioni di trasferimento in ingresso/uscita (I/O).

Contenuto di un'istruzione

Tutte le istruzioni MIPS hanno la **stessa** dimensione **(32 bit) – Architettura RISC.**

Alcune domande:
- Come e dove si specifica il tipo di istruzione?
- Come e dove si specifica da dove vengono letti i dati?
- Come e dove si specifica dove si scrivono i dati prodotti?
- Come viene gestita la memoria in lettura e scrittura?
- Come vengono gestiti i salti?

Codifica delle istruzioni

- I 32 bit hanno un significato diverso a seconda del formato (o tipo) di istruzione
  - il tipo di istruzione è riconosciuto in base al valore di alcuni bit **(6 bit)** più significativi **(codice operativo - *OPCODE*)**
- Le istruzioni MIPS sono di **3** tipi (formati):
  - **Tipo R (register) – Lavorano su 3 registri.**
    - Istruzioni aritmetico-logiche.
  - **Tipo I (immediate) – Lavorano su 2 registri. L'istruzione è suddivisa in un gruppo di 16 bit contenenti informazioni + 16 bit riservati ad una costante.**
    - Istruzioni di accesso alla memoria o operazioni contenenti delle costanti.
  - **Tipo J (jump) – Lavora senza registri: codice operativo + indirizzo di salto.**
    - Istruzioni di salto incondizionato.

	6-bit	5-bit	5-bit	5-bit	5-bit	6-bit
R	op	rs	rt	rd	shamt	funct
I	op	rs	rt	indirizzo		
J	op	indirizzo				

Tipi di istruzioni dell'ISA

Consideriamo istruzioni di tipo R, di tipo lw/sw, salto condizionato:

R

31-26	25-21	20-16	15-11	10-6	5-0
op	rs	rt	rd	shamt	funct

lw/sw

beq

31-26	25-21	20-16	15-0
35/43	rs	rt	offset

31-26	25-21	20-16	15-0
4	rs	rt	indirizzo

Architettura RISC:
Campo Op sempre contenuto nei primi 6 bit, inseguito Op[5-0].
Registri da leggere, rs ed rt, in posizione 25-21 e 20-16. Vengono sempre letti dal Register File.
Registro base per lettura / scrittura, rs, sempre in posizione 25-21.
Offset sempre in posizione 15-0.
Registro destinazione, rd (15-11) per le istruzioni di tipo R, rt (20-16) per le istruzioni lw.

Definizione di un'ISA

Definizione del funzionamento: insieme delle istruzioni (interfaccia verso i linguaggi ad alto livello).

- Tipologia di istruzioni.
- Meccanismo di funzionamento.

Definizione del formato: codifica delle istruzioni (interfaccia verso l'HW).

- Formato delle istruzioni.
- Suddivisione in gruppi omogenei dei bit che costituiscono l'istruzione.

Le istruzioni di un'ISA

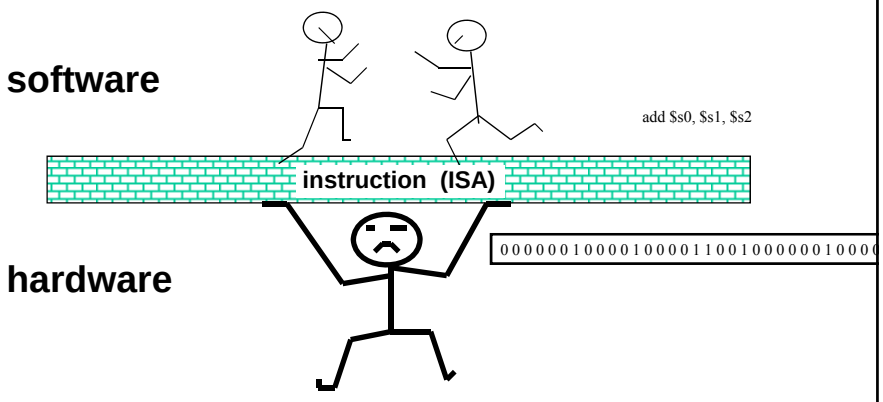
Devono contenere tutte le informazioni necessarie ad eseguire il ciclo di esecuzione dell'istruzione: registri, comandi, ...

Ogni architettura di processore ha il suo linguaggio macchina

- Architettura dell'insieme delle istruzioni elementari messe a disposizione dalla macchina (in linguaggio macchina).
  - **ISA (Instruction Set Architecture)**
- Due processori con lo stesso linguaggio macchina hanno la stessa architettura delle istruzioni anche se le implementazioni hardware possono essere diverse.
- Consente al SW di accedere direttamente all'hardware di un calcolatore.

L'architettura delle istruzioni, specifica come vengono costruite le istruzioni in modo tale che siano comprensibili alla macchina (in linguaggio macchina).

Insieme delle istruzioni

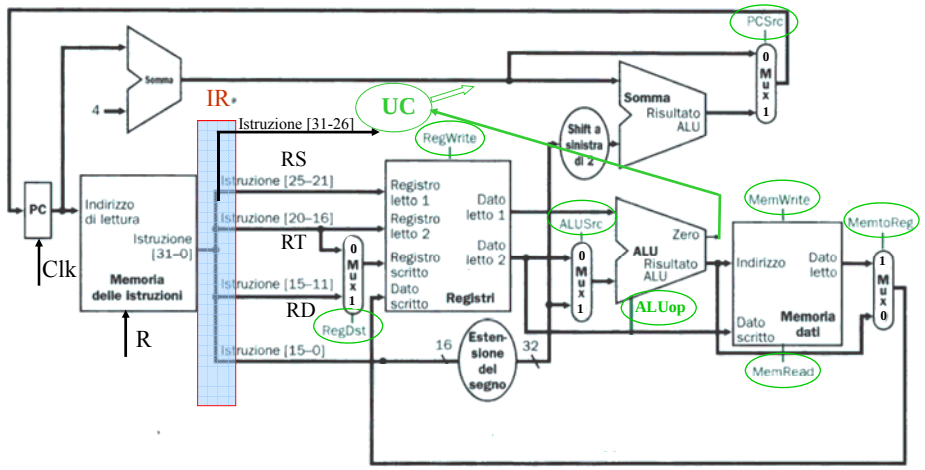


Quale è più facile modificare?

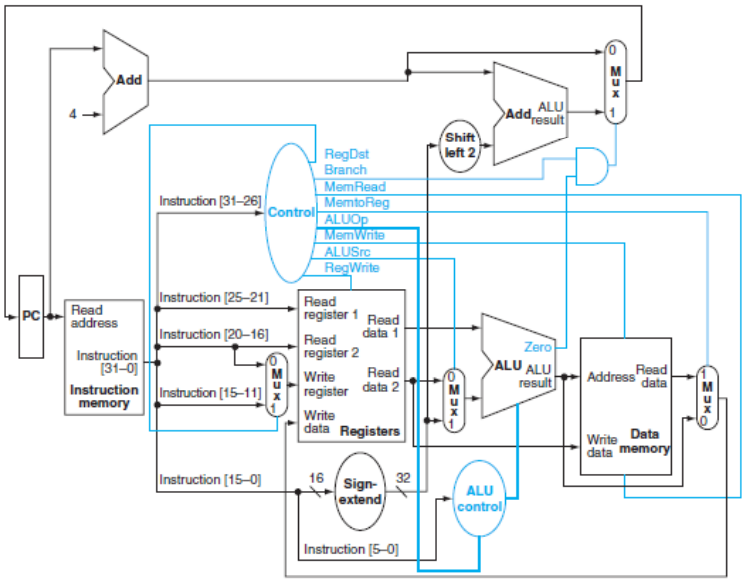
Introduzione alla CPU

- Introduzione
- La CPU a ciclo singolo

Schema generale (lw/sw + R + beq)



The control and data path



Osservazioni

Il ciclo di esecuzione di un'istruzione si compie in un unico ciclo di clock.

⇩

Ogni unità funzionale può essere utilizzata 1 sola volta.

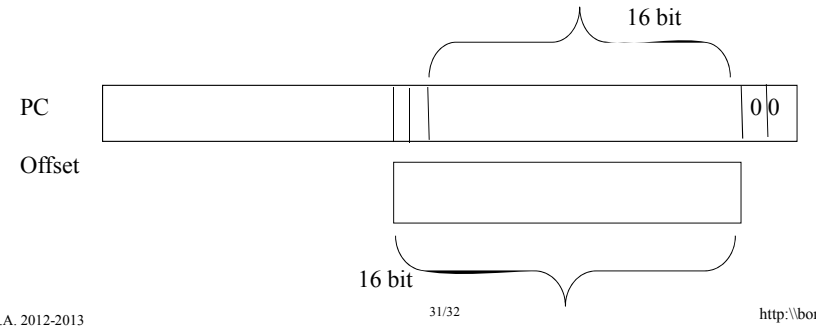
⇩

Duplicazione Memoria: Memoria dati e memoria istruzioni.
Triplicazione ALU: 3 ALU: 2 sommatori + 1 general purpose.

Allargamento dello spazio di indirizzamento

0000	0	0
0100	1	4
1000	2	8
1100	3	12

Considero 64Mword invece di 64Kbyte. Lo spazio indirizzabile all'interno del segmento di testo è di 64Kword * 4 = 256Kbyte.



Introduzione alla CPU

- Introduzione
- La CPU a ciclo singolo