Una Macchina Nozionale Eseguibile per Architetture dei Calcolatori

D. D'Agostino, G. Delzanno, G. Guerrini, D. Traversaro

DIBRIS, Università degli Studi di Genova

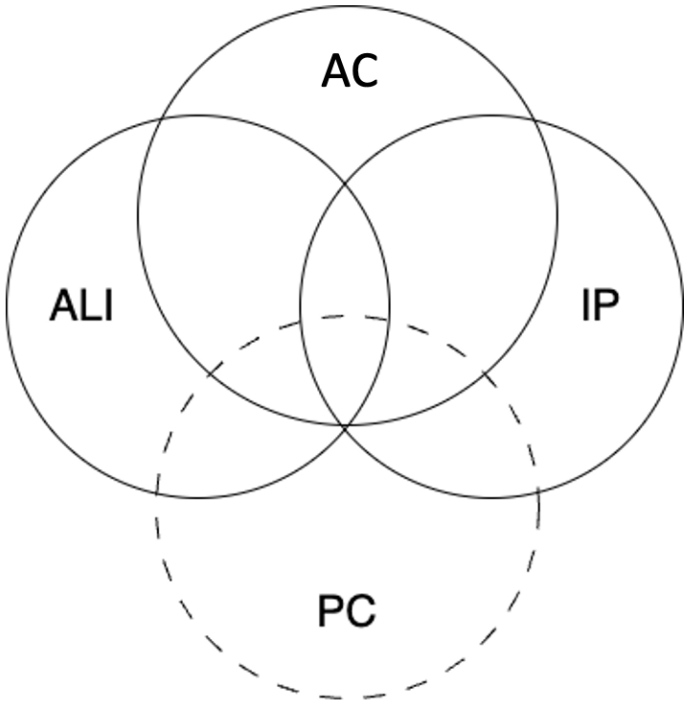
ITADINFO 2023

Contesto

- Il primo semestre del primo anno del Corso di Laurea in Informatica dell'Università di Genova è basato su tre insegnamenti: Introduzione alla Programmazione (IP), Architetture dei Calcolatori (AC), Algebra e Logica per Informatica (ALI)
- Il primo anno è da alcuni anni oggetto di studio per metodi didattici innovativi nel tentativo di ridurre gli abbandoni
- Il 2022/23 è stato un anno critico visto l'alto numero di studenti in aula (più di 300 in aule da 180)

Contesto

- In questo ambito principi e strumenti del Pensiero Computazionale (PC) sono stati spesso usati per attività di collegamento tra i corsi
- In questa presentazione descriveremo un esperimento nell'intersezione $AC \cap IP \cap ALI \cap PC$.



Contesto

- A partire dall'edizione 2022/23 il programma di AC segue un approccio basato su RISC-V come architettura di riferimento (*), un open standard ISA (Instruction Set Architecture), basato sul principio RISC (Reduced Instruction Set Computer)
- Dal punto di vista tecnico, RISC-V è una architettura load–store con istruzioni per accedere alla memoria (load e store tra memoria e registri) e istruzioni aritmetico-logiche che operano solo su registri.
- La programmazione in linguaggio macchina è uno degli argomenti ostici di questo tipo di corsi: ridurre al minimo l'instruction set rende molto complesso codificare anche algoritmi banali

(*) Hennessy & Patterson, Struttura e progetto dei calcolatori. Progettare con RISC-V, 2023

Verso una Macchina Nozionale per AC?

- Un modo per superare queste difficoltà iniziali può essere quello di accompagnare lo studente nella costruzione e nel raffinamento del proprio modello mentale di un macchina nozionale dell'elaboratore
- La macchina nozionale può essere interpretata come un'insieme di regole, non necessariamente formalizzate, per descrivere una versione astratta della vera architettura che possa aiutare uno studente ad apprendere il suo funzionamento, minizzando ad esempio misconcezioni ed errori comuni.

Tra sacro e profano: l'esperimento del 2022/23

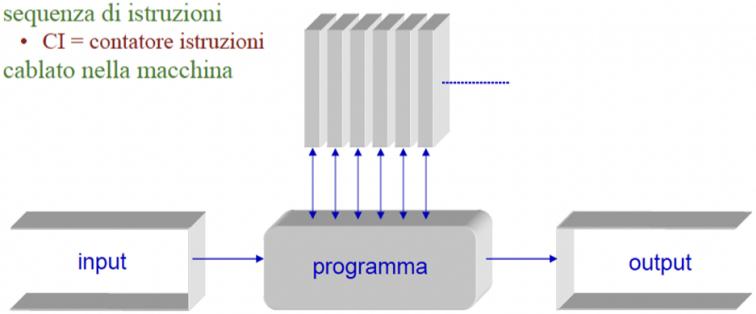
- Si è voluto adottare una macchina nozionale (NM) in grado di rappresentare i concetti di base di architetture load-store e relativi reduced instruction set, partendo dalla quale poter ricostruire i costrutti che nel corso di Introduzione alla Programmazione venivano nel frattempo presentati attraverso il C++.
- In alternativa a modelli basati su regole o diagrammi, si è pensato ad un approccio, ispirato al Pensiero Computazionale e al coding, in cui la NM fosse esplicita ed eseguibile tramite un simulatore
- Il simulatore doveva essere facilmente manipolabile anche da novizi per permettere di elaborare il proprio modello mentale e confrontarlo con quello proposto dai docenti ad esempio estendendo parti del simulatore

NM = Modello RAM di Cook-Reckow

Idealizzazione di un computer RISC basato sul modello delle Random Access Machine introdotte da Cook e Reckhow (*)

Random Access Machine

- memoria
  - costituita da un numero arbitrario di celle (registri) indirizzabili direttamente (RAM)
  - una cella può contenere un intero
  - la cella zero viene chiamata anche *accumulatore* e ha un ruolo speciale
- programma
  - sequenza di istruzioni
    - CI = contatore istruzioni
  - cablato nella macchina



LOAD	=valore | registro | *registro	*carica il valore nell'accumulatore*
STORE	registro | *registro	*memorizza l'accumulatore nel registro*
ADD	=valore | registro | *registro	*somma all'accumulatore*
SUB	=valore | registro | *registro	*sottrae all'accumulatore*
MULT	=valore | registro | *registro	*moltiplica l'accumulatore*
DIV	=valore | registro | *registro	*divide l'accumulatore*
READ	registro | *registro	*legge dall'input nel registro*
WRITE	=valore | registro | *registro	*scrive in output*
JUMP	etichetta	*salto non condizionato*
JGTZ	etichetta	*salto condizionato all'accumulatore > 0*
JZERO	etichetta	*salto condizionato all'accumulatore = 0*
HALT	etichetta	*termina la computazione*

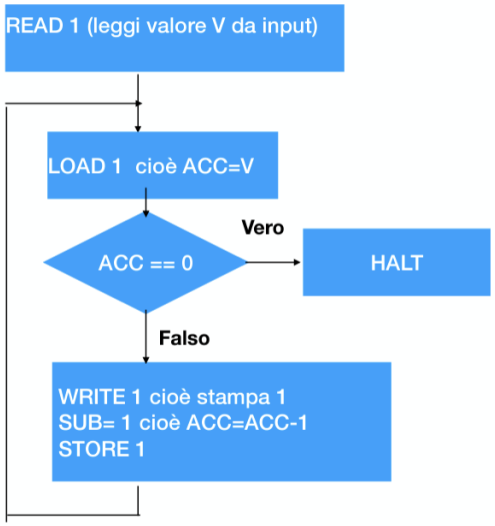
- gli operandi (valori, registri, etichette) sono tutti interi
- *registro realizza l'indirizzamento indiretto
  - l'indice del registro su cui operare è contenuto nel registro passato come operando

(*) S. Cook & R. Rechow
Time-Bounded Random Access Machines
STOC '72

Descrizione "eseguibile"

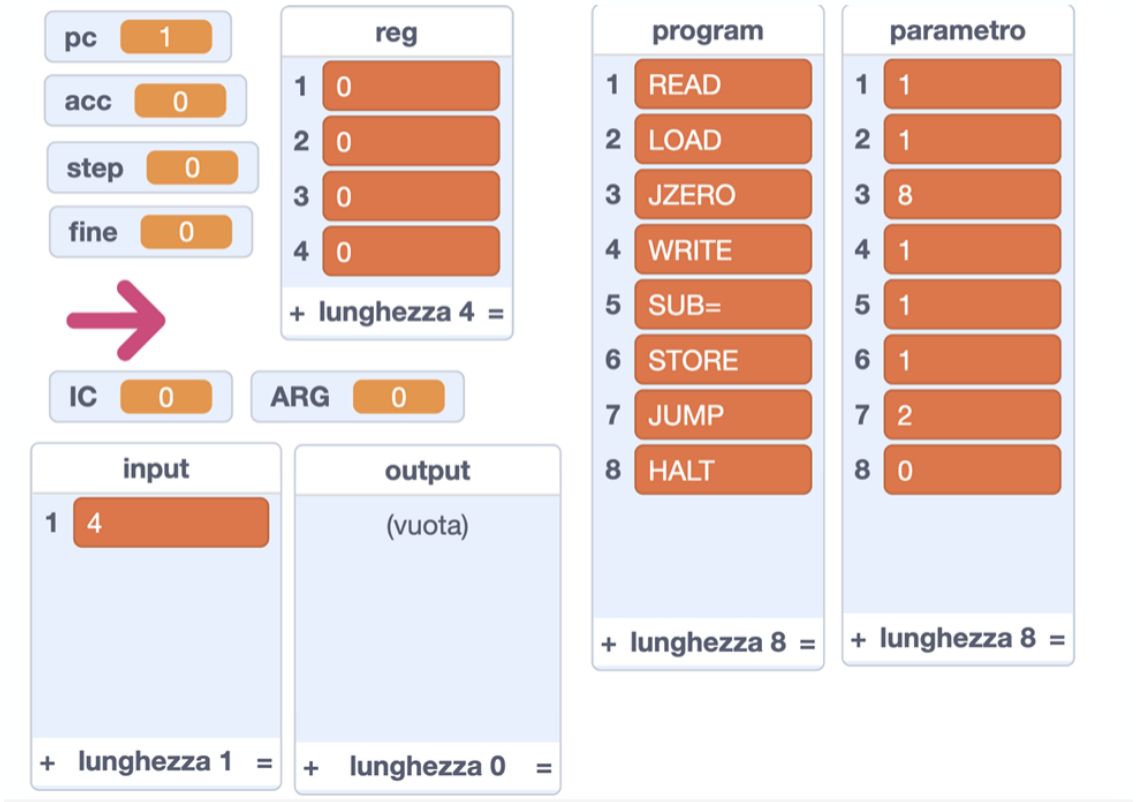
Descrizione del funzionamento della macchina RAM tramite un linguaggio di programmazione a blocchi con semantica intuitiva

	program	parametro
1	READ	1
2	LOAD	1
3	JZERO	8
4	WRITE	1
5	SUB=	1
6	STORE	1
7	JUMP	2
8	HALT	0



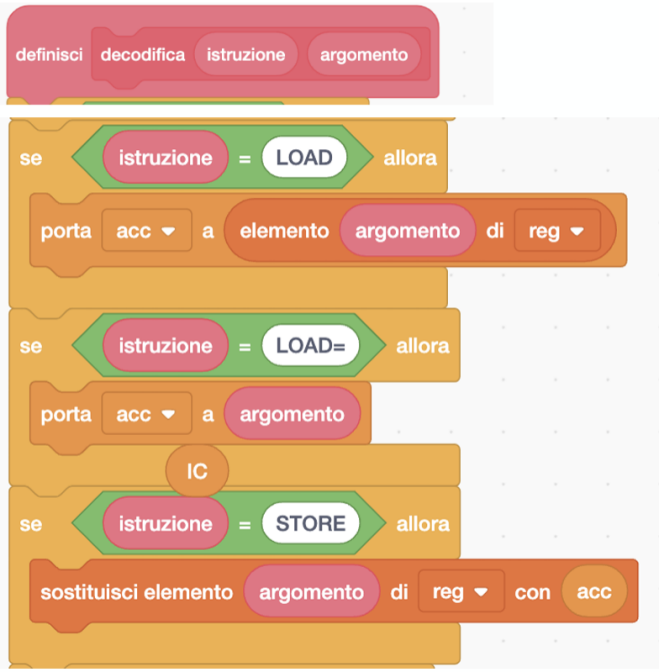
Registri e stato

Per rappresentare i registri di stato (es. program counter e accumulatore) e i registri di memoria abbiamo usato rispettivamente variabili e liste visualizzate tramite l'editor.



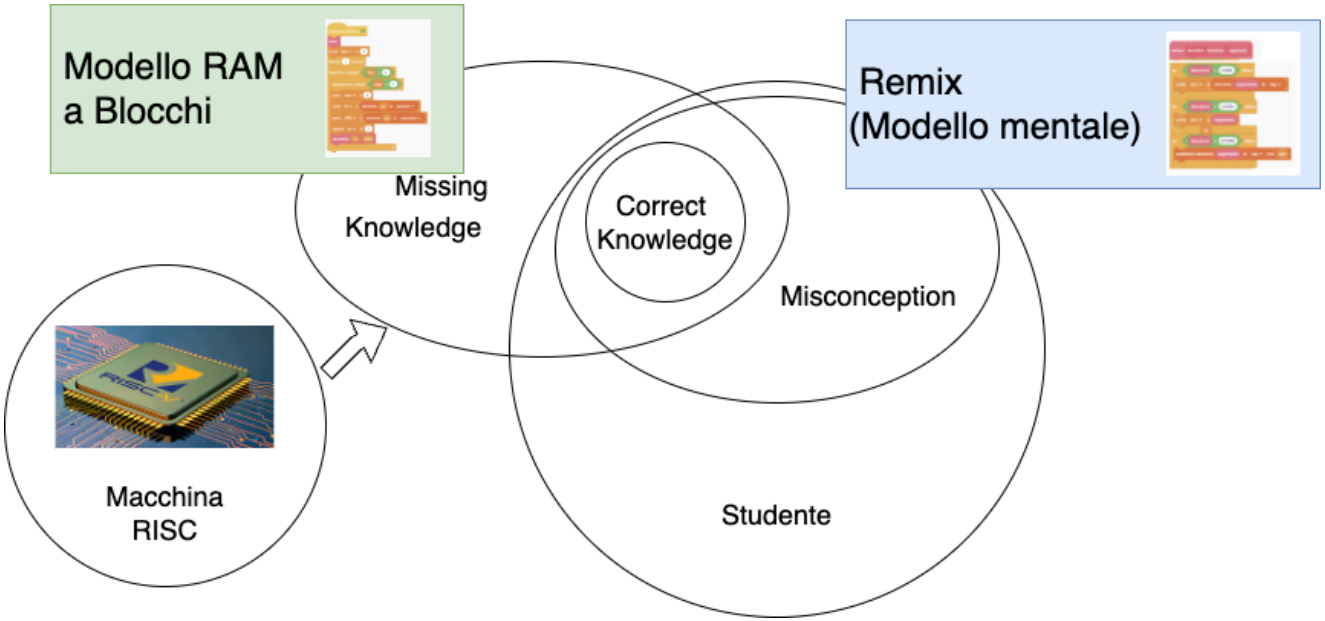
Ciclo di fetch e decodifica

La nozione di esecuzione è stata descritta tramite regole per definire il significato del ciclo di fetch e decodifica per le singole istruzioni



Metodo usato in classe

L'interprete è stato costruito, nelle prime lezioni del corso, insieme agli studenti sfruttando i blocchi come regole



Condividere progetto e codice ha permesso ai singoli studenti di elaborare il proprio modello mentale dell'ISA ed completarlo creando un cosidetto remix del progetto iniziale

Vantaggi

Il vantaggio di questo approccio è stato molteplice.

- Gli studenti inesperti hanno potuto cimentarsi subito (e quasi inconsciamente) con un interprete e comunque con un artefatto da usare per modellare il proprio modello mentale di calcolatore RISC.
- Questo ha permesso di esplorare misconception legate ad esempio al funzionamento di alcuni registri (accumulatore e program counter) e al funzionamento delle operazioni (reset di celle sorgenti di una load, funzionamento dell'indirizzamento indiretto anche con celle di memoria, ecc).
- Il legame tra RAM e RISC-V è stato poi ulteriormente evidenziato nel corso del semestre proponendo gli stessi esercizi proposti con la macchina anche in RISC-V.

Vantaggi

- Alcuni studenti esperti comunque trovato stimolante questo approccio trasformando l'esperimento proposto in un progetto per la creazione di un interprete per la RAM con linguaggio di sviluppo scelto a piacere (es. Python, Rust) trovando stimoli ulteriori rispetto agli esercizi tradizionali proposti nelle prime settimane di IP.
- Visto il ridotto set di istruzioni fornito dalla RAM, gli esercizi proposti si sono rivelati comunque concettualmente interessanti (es allocazione dinamica della memoria)

Edizione 2023/24

- Il primo anno è stato frazionato in due classi
- Di proverà a sfruttare l'esperienza dello scorso anno proponendo una macchina nozionale basata su una versione semplificata di RISC-V (senza passare da linguaggi intermedi), in realtà non distante dalla RAM di Cook e Reckhow
- Verrà poi estendesa con le parti più tecniche legate ai vari tipi di indirizzamento, alla codifica di procedure, ecc.

Edizione 2023/24

Come ho suggerito ai miei colleghi del primo anno. . .

. . . il linguaggio a blocchi era Scratch!