


L'unità di controllo di CPU multi-ciclo

Prof. Alberto Borghese
Dipartimento di Scienze dell'Informazione
borghese@dsi.unimi.it

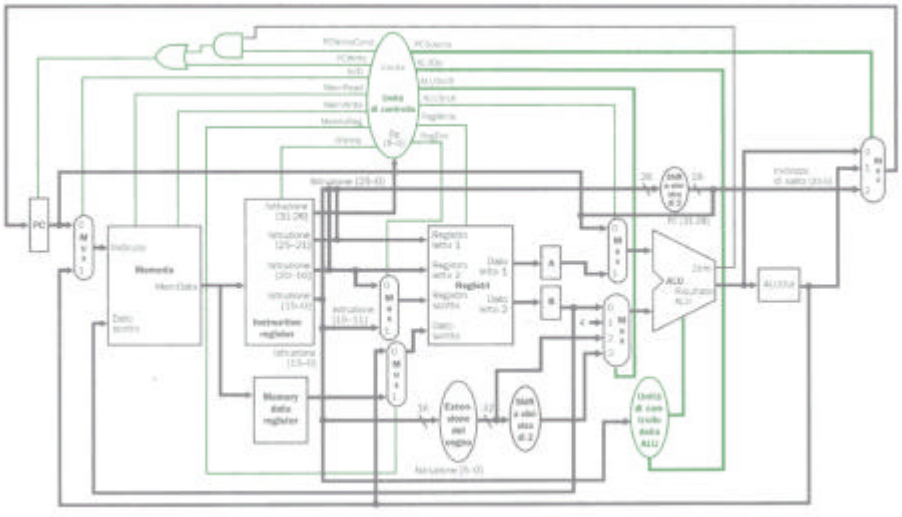
Università degli Studi di Milano


Sommario

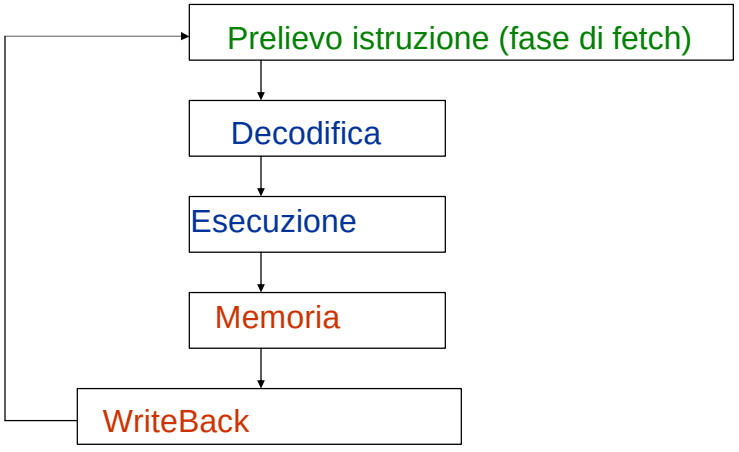
I segnali di controllo della CPU multi-ciclo

Sintesi dell'Unità di Controllo come Macchina a Stati Finiti

CPU multi-ciclo



Ciclo di esecuzione di un'istruzione



Le istruzioni richiederanno da 3 a 5 cicli di clock

Principio della suddivisione in passi

•Tutte le operazioni elementari che hanno bisogno di unità funzionali diverse possono essere eseguite in parallelo.

•Tutte le operazioni elementari che hanno bisogno della stessa unità funzionale devono essere eseguite in serie (in passi di esecuzione successivi).

Riassunto dell'esecuzione

Nome del passo	Azioni per Istruzioni di Tipo R	Azioni per istruzioni di accesso alla memoria	Azioni per salti condizionati	Azioni per salti non condizionati
Fetch	IR = Memory[PC] PC = PC + 4			
decodifica & Prelievo dati dai registri	A = Reg[IR[25-21]] B = Reg[IR[20-16]] ALUOut = PC+4 + (sign_extend(IR[15-0] << 2)			
Esecuzione	ALUOut = A oper B	ALUOut = A + sign_extend(IR[15-0])	If (A == B) then PC = ALUOut	PC = PC[31-28] || (IR[25-0]<<2)
Memoria o WriteBack		Load: MDR = Memory[ALUOut] Store: Memory[ALUOut] = B		
WriteBack	Reg([IR[15-11]] = ALUOut	Load: Reg[IR[20-16]] = MDR		

Le istruzioni richiedono da 3 a 5 cicli di clock

Segnali di controllo

Nome del segnale	Valore	Effetto
ALUSrcA (1 bit)	0	Il primo operando è il valore attuale del PC
	1	Il primo operando proviene dalla prima porta di lettura del Register File
ALUSrcB (2 bit)	00	Il secondo operando proviene dalla seconda porta di lettura del RF
	01	Il secondo operando è la costante + 4
	10	Il secondo operando è l'estensione del segno del campo offset
	11	Il secondo operando proviene dall'estensione del segno e dallo shift a sx di due posizioni, dell'offset
IorD	0	L'indirizzo della memoria proviene dal PC
	1	L'indirizzo della memoria proviene dalla ALU (ALUOut)
PCSource	00	In PC viene scritta l'uscita della ALU (PC+4)
	01	In PC viene scritta il contenuto di ALUOut (indirizzo di una branch)
	10	In PC viene scrittto l'indirizzo di destinazione della jump
PCWrite	0	Non abilitazione della scrittura
	1	Viene scritto il registro PC. L'indirizzo scritto in PC è controllato da PCSource
PCWriteCond	0	Non abilitazione della scrittura.
	1	Il PC viene scritto se anche l'uscita Zero della ALU è affermata

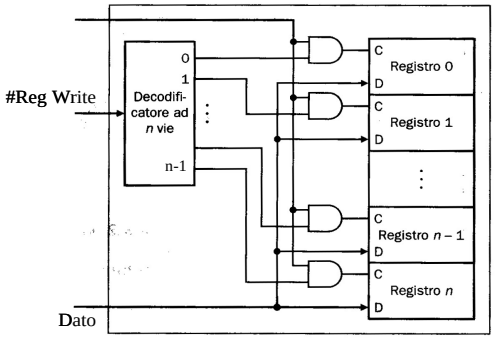
Altri segnali di controllo

Scrittura del Register File – solamente nelle istruzioni con fase di WriteBack.

Scrittura / lettura Memoria – solamente nelle istruzioni di accesso alla memoria.

Scrittura nei registri A, B, ALUOut – ad ogni colpo di clock.

Scrittura del IR – Avviene durante la fase di fetch.



Sommario

I segnali di controllo della CPU multi-ciclo

Sintesi dell'Unità di Controllo come Macchina a Stati Finiti

Segnali di controllo

Passo esecuzione	IorD	MemRead	MemWrite	IRWrite	ALUScrA	ALUSrcB	ALUop	PCSource	PCWrite	PCWriteCond	RegWrite	RegDst	MemtoReg
Fase fetch													
Decodifica													
Exec I – lw													
Exec II – lw													
Exec III – lw													
Exec I – sw													
Exec II – sw													
Exec I – R													
Exec II – R													
Exec I – beq													
Exec I – j													

Segnali di controllo

Passo esecuzione	IorD	MemRead	MemWrite	IRWrite	ALUScrA	ALUSrcB	ALUop	PCSource	PCWrite	PCWriteCond	RegWrite	RegDst	MemtoReg
Fase fetch	0	1	0	1	0	01	10	00	1	0	0	X	X
Decodifica	X	0	0	0	0	11	10	X	0	0	0	X	X
Exec I – lw	X	0	0	0	1	10	00	X	0	0	0	X	X
Exec II – lw	1	1	0	0	X	X	X	X	0	0	0	X	X
Exec III – lw	X	0	1	0	X	X	X	X	0	0	1	0	1
Exec I – sw	X	0	0	0	1	10	00	X	0	0	0	X	X
Exec II – sw	1	0	1	0	X	X	X	X	0	0	0	X	X
Exec I – R	X	0	0	0	1	00	10	X	0	0	0	X	X
Exec II – R	X	0	0	0	X	X	X	X	0	0	1	1	0
Exec I – beq	X	0	0	0	1	00	01	01	0	1	0	X	X
Exec I – j	X	0	0	0	X	X	X	10	1	0	0	X	X

Sintesi della FSM della CPU

•Stato – passo di esecuzione.
•Uscita – segnali di controllo.
•Ingressi – Codice operativo.

I valori dei segnali di controllo dipendono:
•dal passo dell'istruzione (stato)

Il passo successivo dell'istruzione (stato prossimo) dipende:
• dal codice operativo (ingresso).
• dal passo presente (stato presente).

•Uscita = f(Stato)
•Stato_prossimo = f(Ingressi, Stato_presente)

Macchina a Stati Finiti (di Moore)

La Macchina di Moore è definita, in teoria degli automi, dalla quintupla :

< X, I, Y, f(.), g(.) >

X: insieme degli stati (in numero finito).

I: alfabeto di ingresso: tutti i simboli che si possono presentare in ingresso. Se abbiamo n ingressi, avremo 2^n possibili simboli da leggere in ingresso (configurazioni).

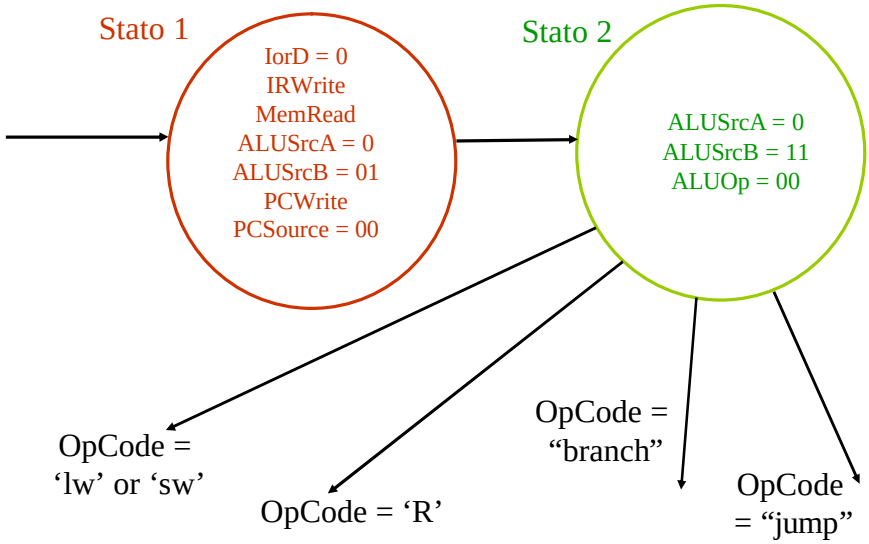
Y: alfabeto di uscita: tutti i simboli che si possono generare in uscita. Se abbiamo m uscite, avremo 2^m possibili simboli da presentare in uscita (configurazioni).

f(.): funzione stato prossimo: X' = f(X,I). Definisce l'evoluzione della macchina nel tempo. L'evoluzione è deterministica.

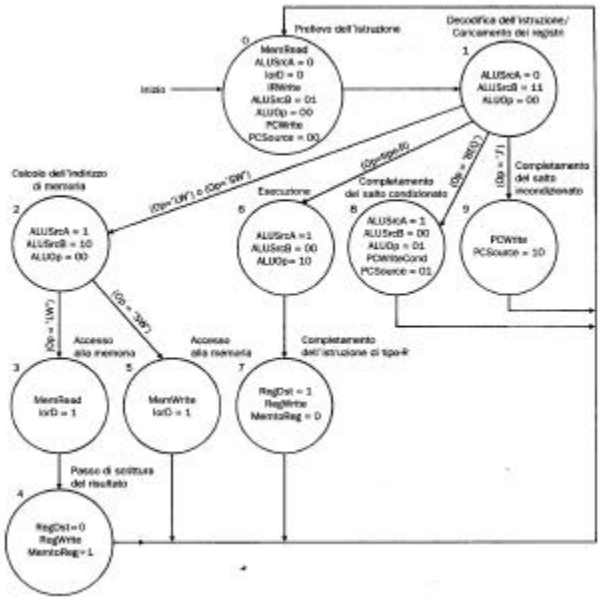
g(.): funzione di uscita: Y= g(X) nelle macchien di Moore.

Per il buon funzionamento della macchina è previsto uno stato iniziale, al quale la macchina può essere portata mediante un comando di reset.

Fase di fetch e decodifica



FSM - STG



FSM - STT

Passo esecuzione	OpCode = R	OpCode = sw	OpCode = lw	OpCode = beq	OpCode = j	IorD	MemRead	MemWrite	IRWrite	ALUScrA	ALUSrcB	ALUop	PCSource	PCWrite	PCWriteCond	RegWrite	RegDst	MemtoReg
Fase fetch – 0						0	1	0	1	0	01	10	00	1	0	0	X	X
Decodifica – 1						X	0	0	0	0	11	10	X	0	0	0	X	X
Exec I – sw/lw – 2						X	0	0	0	1	10	00	X	0	0	0	X	X
Exec II – lw – 3						1	1	0	0	X	X	X	X	0	0	0	X	X
Exec III – lw - 4						X	0	1	0	X	X	X	X	0	0	1	0	1
Exec II – sw - 5						1	0	1	0	X	X	X	X	0	0	0	X	X
Exec I – R – 6						X	0	0	0	1	00	10	X	0	0	0	X	X
Exec II – R - 7						X	0	0	0	X	X	X	X	0	0	1	1	0
Exec I – beq – 8						X	0	0	0	1	00	01	01	0	1	0	X	X
Exec I – j - 9						X	0	0	0	X	X	X	10	1	0	0	X	X

FSM - STT

Passo esecuzione	OpCode = R	OpCode = sw	OpCode = lw	OpCode = beq	OpCode = j	IorD	MemRead	MemWrite	IRWrite	ALUScrA	ALUSrcB	ALUop	PCSource	PCWrite	PCWriteCond	RegWrite	RegDst	MemtoReg
Fase fetch – 0	1	1	1	1	1	0	1	0	1	0	01	10	00	1	0	0	X	X
Decodifica – 1	6	2	2	8	9	X	0	0	0	0	11	10	X	0	0	0	X	X
Exec I – sw/lw – 2	X	5	3	X	X	X	0	0	0	1	10	00	X	0	0	0	X	X
Exec II – lw – 3	X	X	4	X	X	1	1	0	0	X	X	X	X	0	0	0	X	X
Exec III – lw - 4	X	X	0	X	X	X	0	1	0	X	X	X	X	0	0	1	0	1
Exec II – sw - 5	X	0	X	X	X	1	0	1	0	X	X	X	X	0	0	0	X	X
Exec I – R – 6	7	X	X	X	X	X	0	0	0	1	00	10	X	0	0	0	X	X
Exec II – R - 7	0	X	X	X	X	X	0	0	0	X	X	X	X	0	0	1	1	0
Exec I – beq – 8	X	X	X	0	X	X	0	0	0	1	00	01	01	0	1	0	X	X
Exec I – j - 9	X	X	X	X	0	X	0	0	0	X	X	X	10	1	0	0	X	X

FSM – sintesi della funzione di uscita

Passo esecuzione	IorD	MemRead	MemWrite	IRWrite	ALUScrA	ALUSrcB	ALUop	PCSource	PCWrite	PCWriteCond	RegWrite	RegDst	MemtoReg
Fase fetch – 0	0	1	0	1	0	01	10	00	1	0	0	X	X
Decodifica – 1	X	0	0	0	0	11	10	X	0	0	0	X	X
Exec I – beq – 8	X	0	0	0	1	00	01	01	0	1	0	X	X
Exec I – j - 9	X	0	0	0	X	X	X	10	1	0	0	X	X
Exec I – R – 6	X	0	0	0	1	00	10	X	0	0	0	X	X
Exec II – R - 7	X	0	0	0	X	X	X	X	0	0	1	1	0
Exec I – sw/lw – 2	X	0	0	0	1	10	00	X	0	0	0	X	X
Exec II – sw - 5	1	0	1	0	X	X	X	X	0	0	0	X	X
Exec II – lw – 3	1	1	0	0	X	X	00	X	0	0	0	X	X
Exec III – lw - 4	X	0	1	0	X	X	X	X	0	0	1	0	1

Esempi:
Y_2 (RegWrite) = (Stato == 7) OR (Stato == 4)
Y_{11} (ALUSrcA) = (Stato == 8) OR (Stato == 6)
OR (Stato == 2)

FSM – codifica della STT (stato futuro)

Passo esecuzione	OpCode = R 000000	OpCode = sw 101011	OpCode = lw 100011	OpCode = beq 000100	OpCode = j 000010
Fase fetch – 0 - 0000	0 0 0 1	0 0 0 1	0 0 0 1	0 0 0 1	0 0 0 1
Decodifica – 1 – 0001	0 1 1 0	0 0 1 0	0 0 1 0	1 0 0 0	1 0 0 1
Exec I – lw/sw – 2- 0010	X X X X	0 1 0 1	0 0 1 1	X X X X	X X X X
Exec II – lw – 3- 0011	X X X X	X X X X	0 1 0 0	X X X X	X X X X
Exec III – lw – 4 - 0100	X X X X	X X X X	0 0 0 0	X X X X	X X X X
Exec II – sw – 5 - 0101	X X X X	0 0 0 0	X X X X	X X X X	X X X X
Exec I – R – 6 - 0110	0 1 1 1	X X X X	X X X X	X X X X	X X X X
Exec II – R - 7 - 0111	0 0 0 0	X X X X	X X X X	X X X X	X X X X
Exec I – beq – 8 - 1000	X X X X	X X X X	X X X X	0 0 0 0	X X X X
Exec I – j – 9 - 1001	X X X X	X X X X	X X X X	X X X X	0 0 0 0

FSM – circuito dello stato futuro

Esempio: $S_0(t+1)=(\bar{S}_0\bar{S}_1\bar{S}_2\bar{S}_3)+(\bar{S}_0\bar{S}_1\bar{S}_2\bar{S}_3)(\bar{i}_0\,\bar{i}_1\,\bar{i}_2\,\bar{i}_3\bar{i}_4\,\bar{i}_5)+(\bar{S}_0\bar{S}_1\bar{S}_2\bar{S}_3)(i_0\,\bar{i}_1\,\bar{i}_2\,\bar{i}_4\,\bar{i}_5)+(\bar{S}_0\bar{S}_1\bar{S}_2\bar{S}_3)$

Passo esecuzione	OpCode = R 000000	OpCode = sw 101011	OpCode = lw 100011	OpCode = beq 000100	OpCode = j 000010
Fase fetch – 0 - 0000	0 0 0 1	0 0 0 1	0 0 0 1	0 0 0 1	0 0 0 1
Decodifica – 1 – 0001	0 1 1 0	0 0 1 0	0 0 1 0	1 0 0 0	1 0 0 1
Exec I – lw/sw – 2- 0010	X X X X	0 1 0 1	0 0 1 1	X X X X	X X X X
Exec II – lw – 3- 0011	X X X X	X X X X	0 1 0 0	X X X X	X X X X
Exec III – lw – 4 - 0100	X X X X	X X X X	0 0 0 0	X X X X	X X X X
Exec II – sw – 5 - 0101	X X X X	0 0 0 0	X X X X	X X X X	X X X X
Exec I – R – 6 - 0110	0 1 1 1	X X X X	X X X X	X X X X	X X X X
Exec II – R - 7 - 0111	0 0 0 0	X X X X	X X X X	X X X X	X X X X
Exec I – beq – 8 - 1000	X X X X	X X X X	X X X X	0 0 0 0	X X X X
Exec I – j – 9 - 1001	X X X X	X X X X	X X X X	X X X X	0 0 0 0

Sommario

I segnali di controllo della CPU multi-ciclo

Sintesi dell'Unità di Controllo come Macchina a Stati Finiti