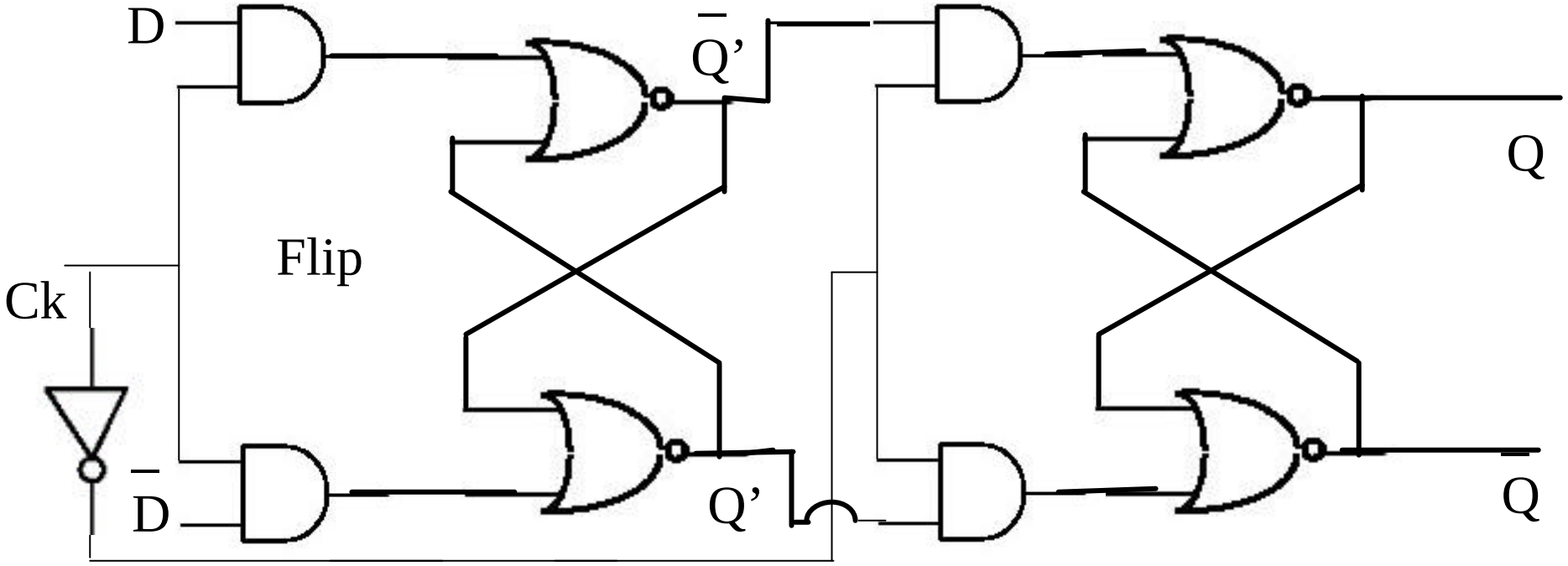
Architettura della CPU multi-ciclo

Architettura
degli Elaboratori e delle Reti,
Turno I

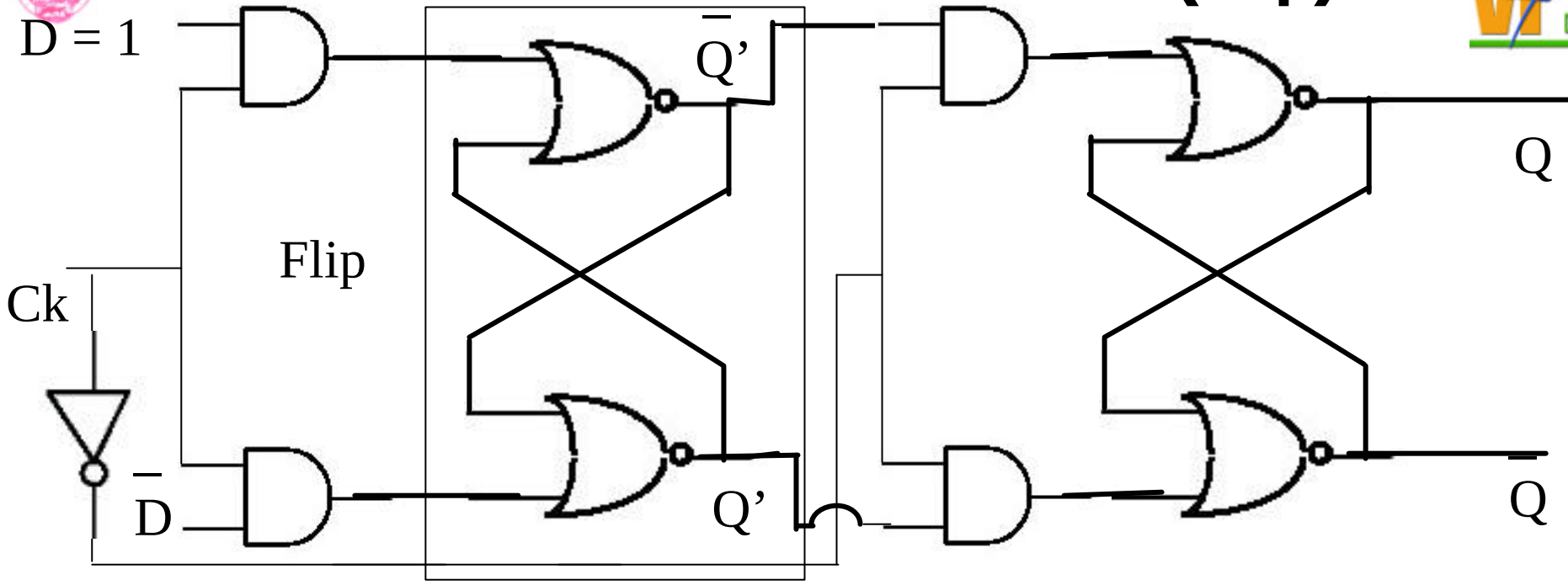
Alberto Borghese
Università degli Studi di Milano
Dipartimento di Scienze dell'Informazione
email: borghese@dsi.unimi.it

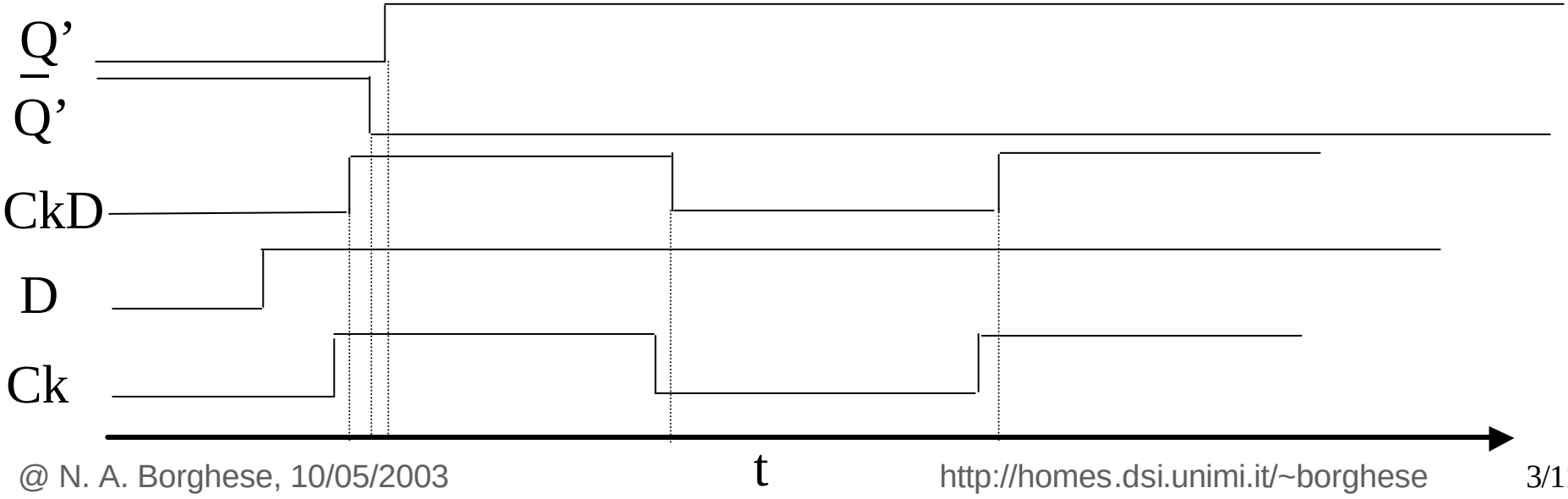
Funzionamento dei bistabili (flip-flop)



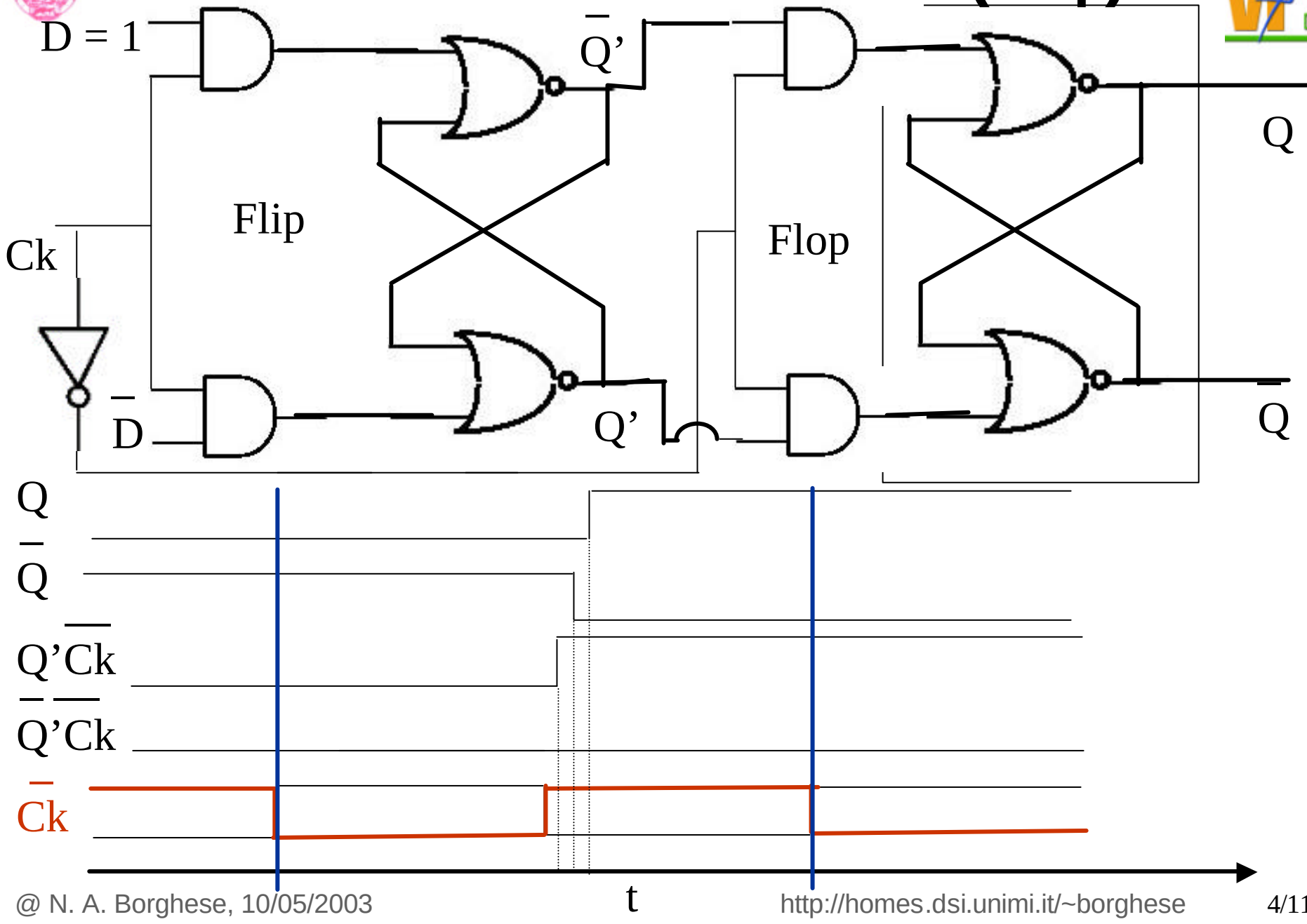
D	CkD	$\overline{Q}$'	$\overline{Ck}$	$\overline{Ck}\overline{Q}$'	Q
Ck	$Ck\overline{D}$	Q'		$\overline{Ck}Q$'	$\overline{Q}$

Funzionamento dei bistabili (flip)





Funzionamento dei bistabili (Flop)



Funzionamento dei registri

Clock alto. Registro abilitato. Quanto presente sulla linea di ingresso D (dati) viene trasferito allo stato intermedio del flip-flop (fase di flip).

Clock basso. Quanto presente sulla linea di ingresso dati viene bloccato dalla porta AND. Il master mantiene l'informazione grazie al feed-back dell'uscita. Nella fase di flop (clock basso), trasferisce lo stato del master allo slave.

Per questo motivo non c'è interferenza tra scrittura (che sarà disponibile a partire dal fronte di salita successivo del clock), e lettura dello stesso elemento.

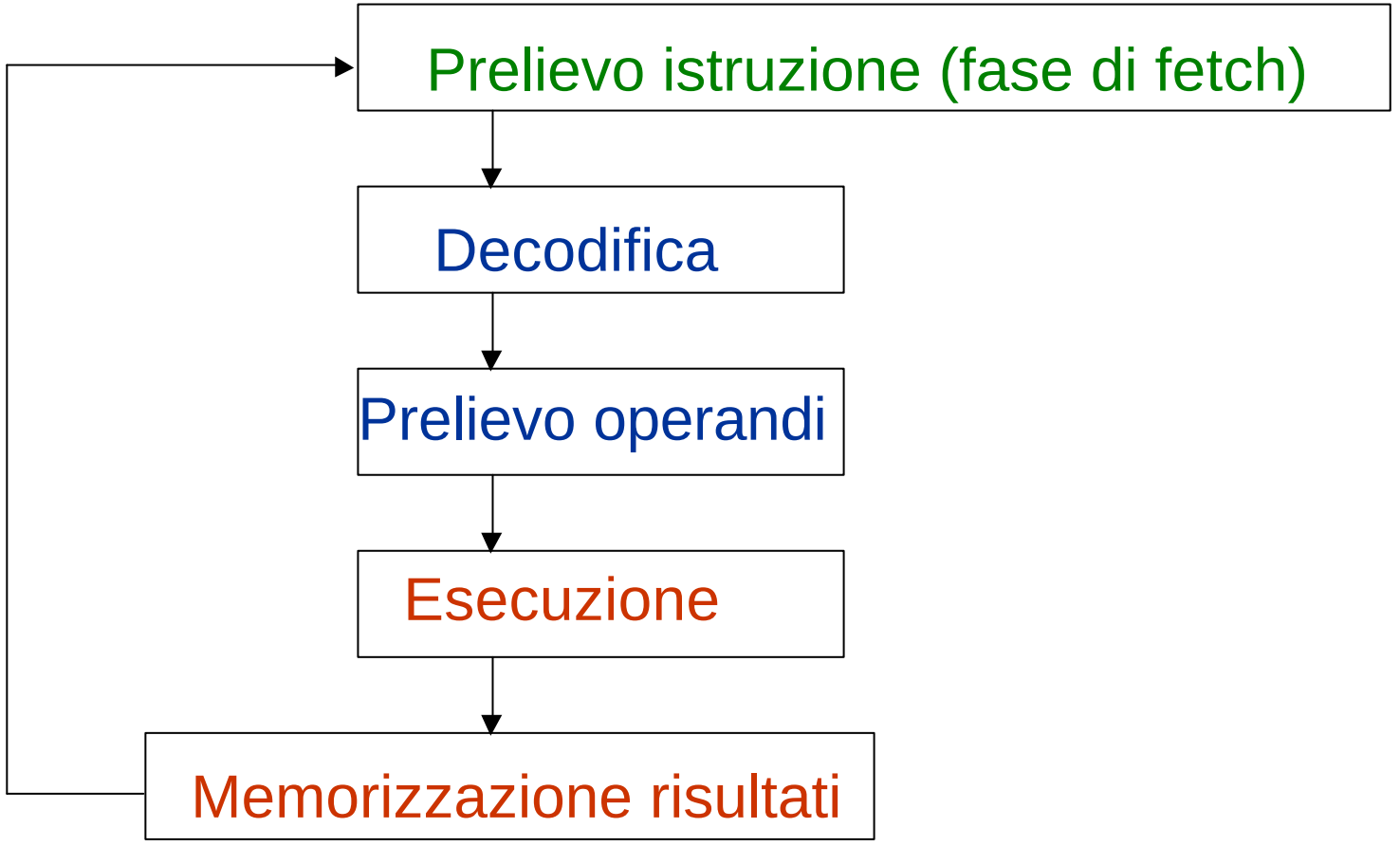
Scrittura nei registri della CPU multi-ciclo

Tutti i registri della CPU vengono scritti ad ogni colpo di clock. Non c'è bisogno di sintetizzare nella UC il segnale di scrittura dei registri.

TRANNE CHE PER l'IR.

Tutti i registri possono essere riscritti ad ogni passo dell'istruzione, tranne l'IR che deve mantenere il suo valore per tutta la durata dell'esecuzione di un'istruzione.

Ciclo di esecuzione di un'istruzione



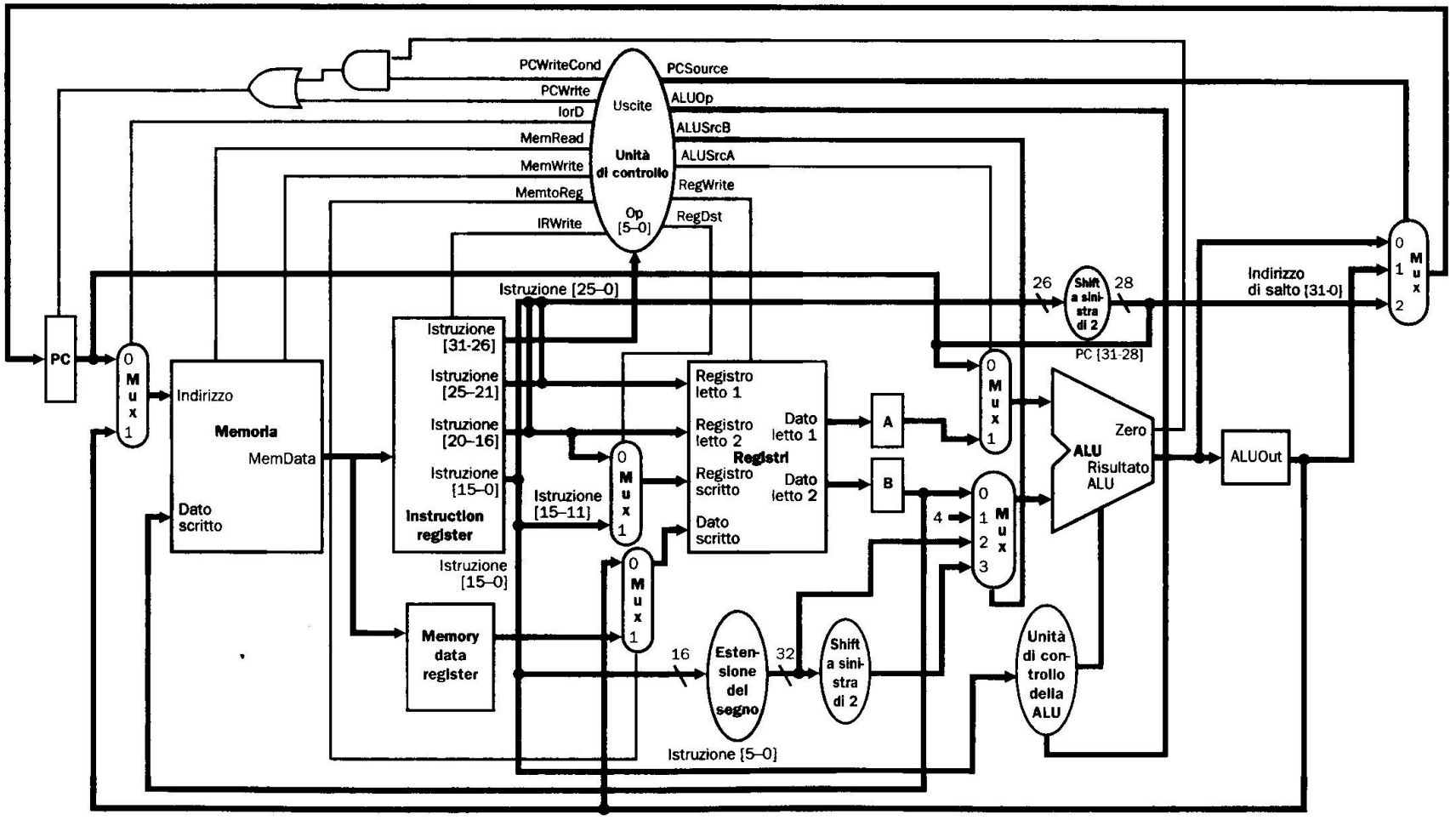
Le istruzioni richiederanno da 3 a 5 cicli di clock

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).

CPU multi-ciclo



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 +		(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)
Conclusione	Reg([IR[15-11]] = ALUOut	Load: MDR = Memory[ALUOut] Store: Memory[ALUOut] = B		
Scrittura finale		Load: Reg[IR[20-16]] = MDR		

Le istruzioni richiedono da 3 a 5 cicli di clock

Esecuzione multi-ciclo: esercizio

Quanti cicli richiede l'esecuzione del seguente frammento di codice:

```
        lw $t2, 0($t3)
        lw $t3, 4($t3)
        beq $t2, $t3, Label
        add $t5, $t2, $t3
        sw $t5, 8($t3)
Label:
```

Cosa succede durante l'ottavo ciclo e durante il tredicesimo?

In quale ciclo l'addizione tra $t2 e $t3 avviene?