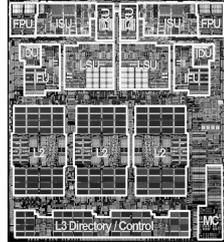




Università degli Studi
di Cassino



Corso di
Calcolatori Elettronici II

Data path multicycle

Anno Accademico 2006/2007

Francesco Tortorella

Problemi dell'implementazione singolo ciclo

Arithmetic & Logical



Load



← Critical Path →

Store



Branch



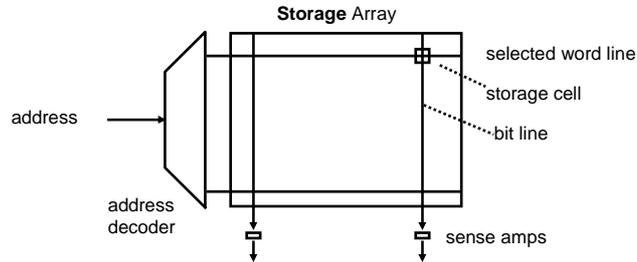
- Tempo di ciclo lungo
 - Tutte le istruzioni hanno un tempo di esecuzione uguale all'istruzione più lenta
 - Memoria reale più lenta della memoria ideale
- Risorse duplicate

Calcolatori Elettronici II
Datapath multicycle - 1

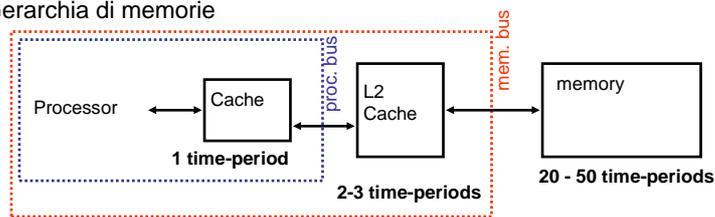
F. Tortorella © 2007
Università degli Studi
di Cassino

Tempo di accesso delle memorie

- Limite fisico: le memorie veloci sono piccole (e viceversa)



- => Gerarchia di memorie

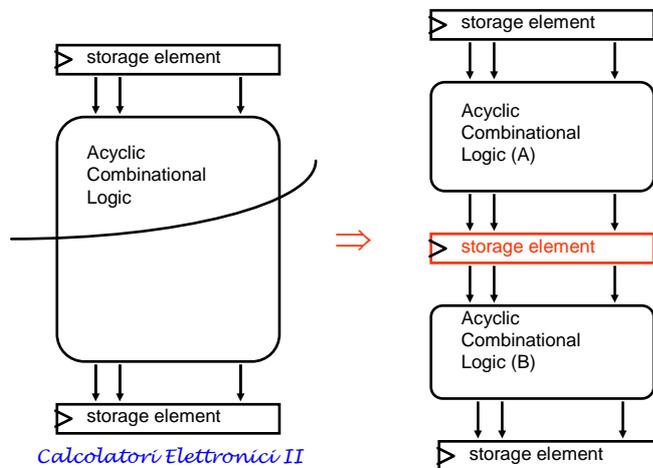


*Calcolatori Elettronici II
Datapath multiciclo - 2*

*F. Tortorella © 2007
Università degli Studi
di Cassino*

Riduzione del tempo di ciclo

- Si partiziona la parte combinatoria in più parti combinatorie più piccole, separate da registri.
- La stessa operazione viene realizzata in due cicli di breve durata invece che in uno solo molto lungo.

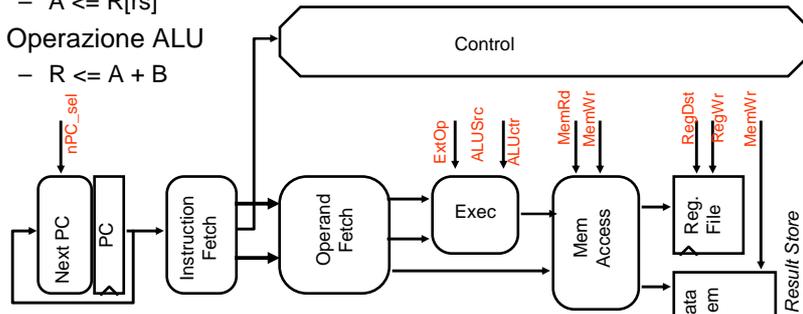


*Calcolatori Elettronici II
Datapath multiciclo - 3*

*F. Tortorella © 2007
Università degli Studi
di Cassino*

Vincoli sul tempo di ciclo

- Logica per calcolare l'indirizzo successivo
 - $PC \leq \text{branch} ? PC + \text{offset} : PC + 4$
- Instruction Fetch
 - $\text{InstructionReg} \leq \text{Mem}[PC]$
- Accesso ai registri
 - $A \leq R[\text{rs}]$
- Operazione ALU
 - $R \leq A + B$

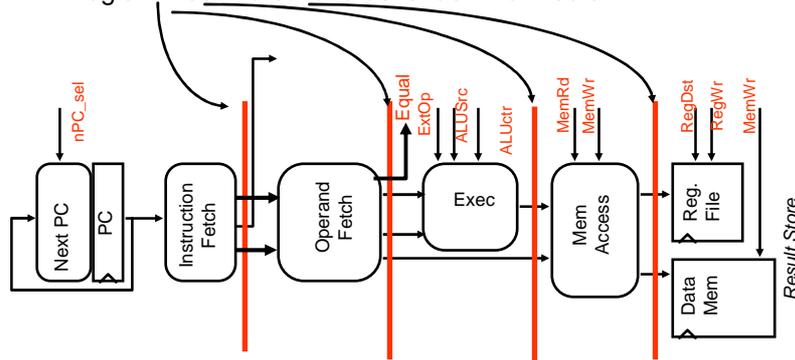


Calcolatori Elettronici II
Datapath multiciclo - 4

F. Tortorella © 2007
Università degli Studi
di Cassino

Partizionamento del datapath

- Registri inseriti a cavallo delle fasi intermedie

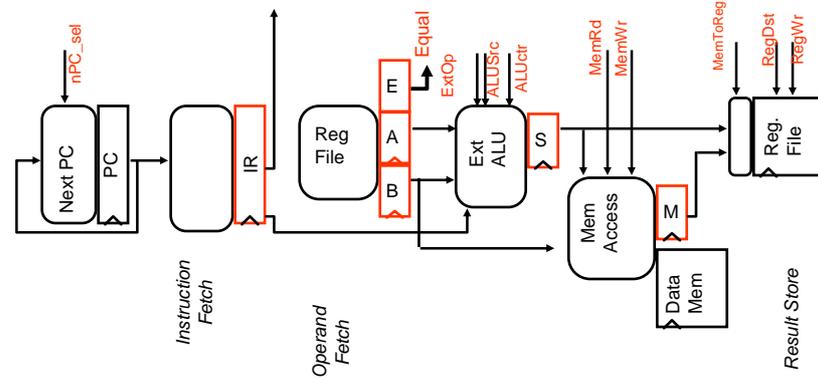


1. Registri inseriti in modo da bilanciare la lunghezza del ciclo di clock (parti combinatorie tra i registri con lo stesso ritardo)
2. Registri inseriti per conservare le informazioni necessarie in passi successivi dell'esecuzione dell'istruzione.

Calcolatori Elettronici II
Datapath multiciclo - 5

F. Tortorella © 2007
Università degli Studi
di Cassino

Datapath multiciclo (visione logica)



- Critical Path ?

Calcolatori Elettronici II
Datapath multiciclo - 6

F. Tortorella © 2007
Università degli Studi
di Cassino

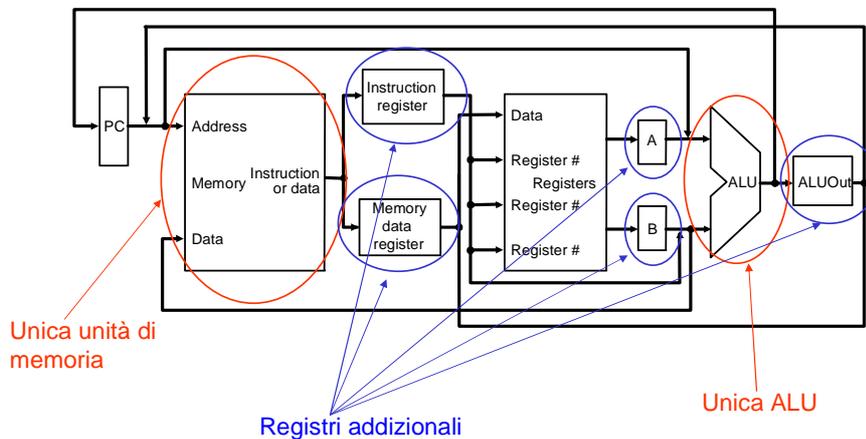
Datapath multiciclo

- Nell'implementazione che stiamo considerando, assumiamo che in un ciclo di clock si possa realizzare al più una tra le seguenti operazioni:
 - Accesso in memoria (lett. istruzione o lett./scritt. dati)
 - Accesso al register file (2 letture o 1 scrittura)
 - Operazione ALU
- Al termine di ogni ciclo di clock, i registri intermedi contengono i dati necessari alle fasi da realizzare nei cicli seguenti.

Calcolatori Elettronici II
Datapath multiciclo - 7

F. Tortorella © 2007
Università degli Studi
di Cassino

Implementazione del datapath multiciclo (vista ad alto livello)



Calcolatori Elettronici II
Datapath multiciclo - 8

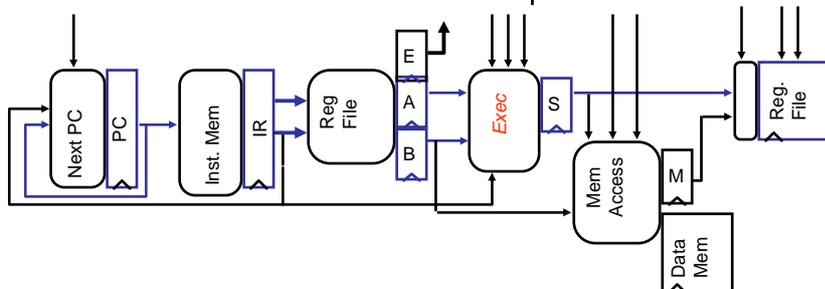
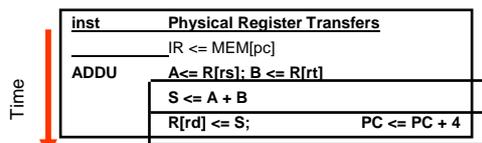
F. Tortorella © 2007
Università degli Studi
di Cassino

Istruzione R-rtipe (add, sub, . . .)

- Logical Register Transfers

inst Logical Register Transfers
ADDU $R[rd] \leftarrow R[rs] + R[rt]; PC \leftarrow PC + 4$

- Physical Register Transfers



Calcolatori Elettronici II
Datapath multiciclo - 9

F. Tortorella © 2007
Università degli Studi
di Cassino

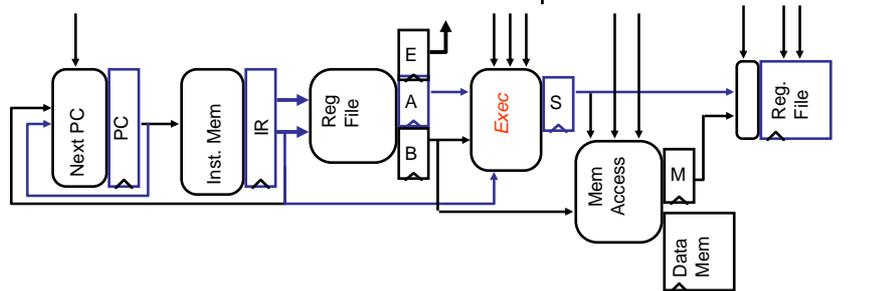
Istruzione Logica con immediato

- Logical Register Transfer

inst Logical Register Transfers
ORI $R[rt] \leftarrow R[rs] \mid ZExt(Im16); PC \leftarrow PC + 4$

- Physical Register Transfers

inst Physical Register Transfers
 $IR \leftarrow MEM[pc]$
ORI $A \leftarrow R[rs]; B \leftarrow R[rt]$
 $S \leftarrow A \mid ZExt(Im16)$
 $R[rt] \leftarrow S; PC \leftarrow PC + 4$



Calcolatori Elettronici II
Datapath multiciclo - 10

F. Tortorella © 2007
 Università degli Studi
 di Cassino

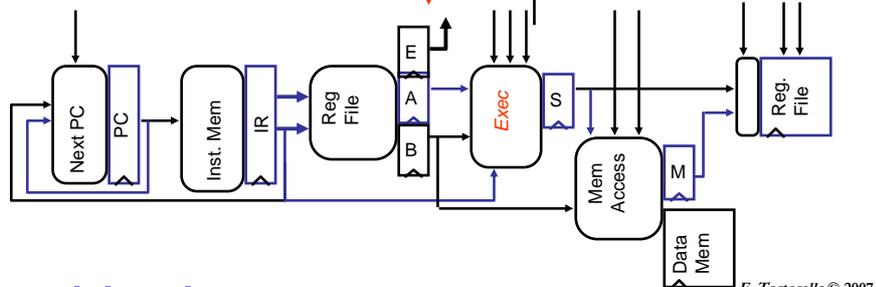
Load

- Logical Register Transfer

inst Logical Register Transfers
LW $R[rt] \leftarrow MEM[R[rs] + SExt(Im16)];$
 $PC \leftarrow PC + 4$

- Physical Register Transfers

inst Physical Register Transfers
 $IR \leftarrow MEM[pc]$
LW $A \leftarrow R[rs]; B \leftarrow R[rt]$
 $S \leftarrow A + SExt(Im16)$
 $M \leftarrow MEM[S]$
 $R[rd] \leftarrow M; PC \leftarrow PC + 4$



Calcolatori Elettronici II
Datapath multiciclo - 11

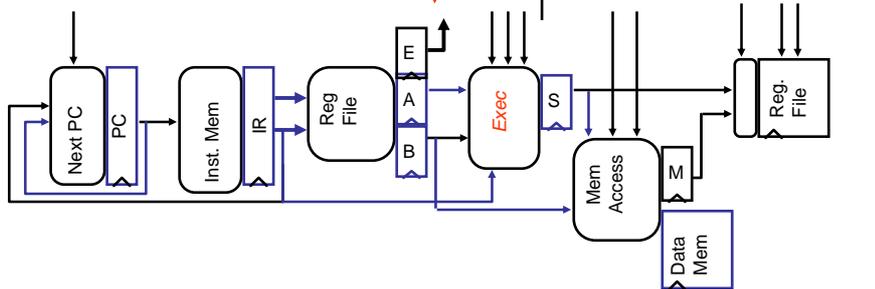
F. Tortorella © 2007
 Università degli Studi
 di Cassino

Store

- Logical Register Transfer
- Physical Register Transfers

inst Logical Register Transfers
SW $MEM[R[rs] + SExt(Im16)] \leftarrow R[rt];$
 $PC \leftarrow PC + 4$

inst Physical Register Transfers
 $IR \leftarrow MEM[pc]$
SW $A \leftarrow R[rs]; B \leftarrow R[rt]$
 $S \leftarrow A + SExt(Im16);$
 $MEM[S] \leftarrow B$ $PC \leftarrow PC + 4$



Calcolatori Elettronici II
Datapath multiciclo - 12

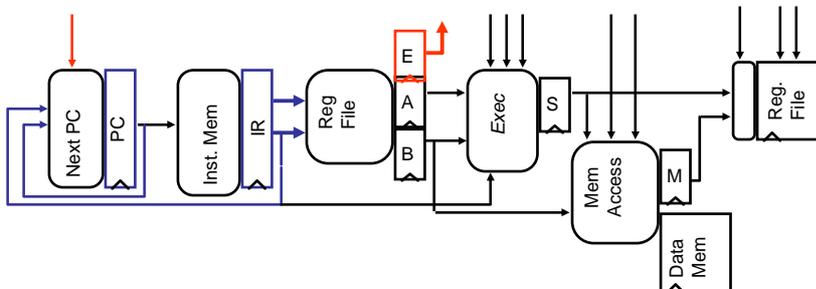
F. Tortorella © 2007
 Università degli Studi
 di Cassino

Branch

- Logical Register Transfer
- Physical Register Transfers

inst Logical Register Transfers
BEQ if $R[rs] == R[rt]$
 then $PC \leftarrow PC + 4 + (SExt(Im16), 2'b00)$
 else $PC \leftarrow PC + 4$

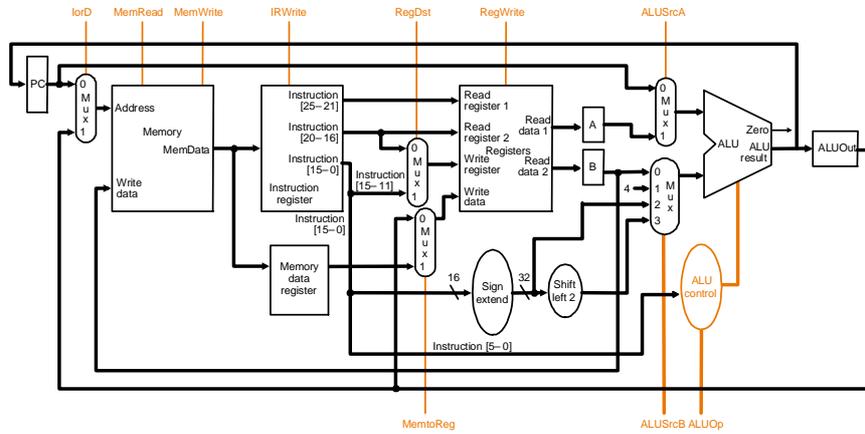
inst Physical Register Transfers
 $IR \leftarrow MEM[pc]$
BEQ $E \leftarrow (R[rs] == R[rt])$
 if $(!E) PC \leftarrow PC + 4;$
 else $PC \leftarrow PC + 4 + (SExt(Im16), 2'b0)$



Calcolatori Elettronici II
Datapath multiciclo - 13

F. Tortorella © 2007
 Università degli Studi
 di Cassino

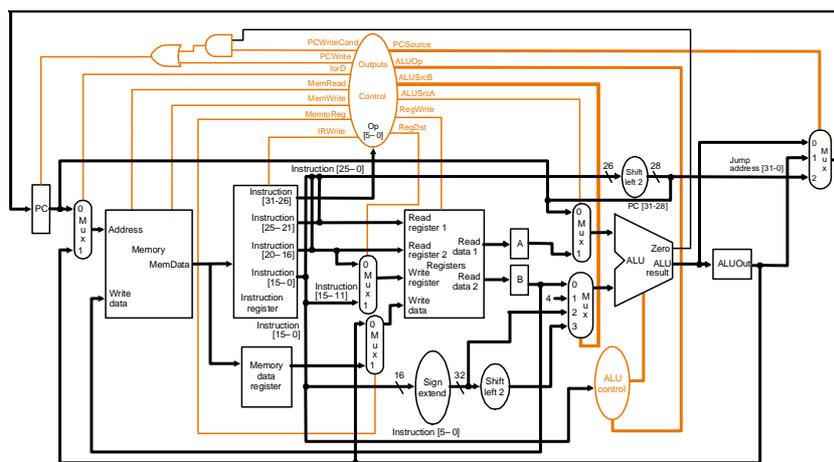
Implementazione del datapath multiciclo



Calcolatori Elettronici II
Datapath multiciclo - 14

F. Tortorella © 2007
Università degli Studi
di Cassino

Implementazione del datapath multiciclo



Calcolatori Elettronici II
Datapath multiciclo - 15

F. Tortorella © 2007
Università degli Studi
di Cassino

Esecuzione dell'istruzione in 5 passi

1. Instruction fetch
2. Instruction decode and register fetch
3. Execution / Memory address computation / Branch completion / Jump completion
4. Memory access / R-type instruction completion
5. Memory read completion

Calcolatori Elettronici II
Datapath multicycle - 16

F. Tortorella © 2007
Università degli Studi
di Cassino

1. Instruction fetch

- $IR \leq Memory[PC]$
- $PC \leq PC+4$

- Quali segnali attivi ?
- Perché si incrementa il PC ?

Calcolatori Elettronici II
Datapath multicycle - 17

F. Tortorella © 2007
Università degli Studi
di Cassino

2. Instruction decode and register fetch

- $A \leq \text{Reg}[\text{IR}[25:21]]$
- $B \leq \text{Reg}[\text{IR}[20:16]]$
- $\text{ALUOut} \leq \text{PC} + (\text{signext}(\text{IR}[15:0])) \ll 2$

- Non è ancora nota l'istruzione da eseguire
- Si preparano possibili operandi (rs->A, rt->B) e si calcola l'offset dell'indirizzo destinazione di una eventuale istruzione di branch

3. Execution / Memory address computation / Branch completion / Jump completion

- Le operazioni eseguite da questo passo in poi dipendono dalla particolare istruzione
- **Load/Store**
 - $\text{ALUOut} \leq A + \text{signext}(\text{IR}[15:0])$
- **Istruzione Logico-Aritmetica (R-type)**
 - $\text{ALUOut} \leq A \text{ op } B$
- **Branch (completa)**
 - $\text{If } (A == B) \text{ PC} \leq \text{ALUOut}$
 - La condizione è fornita dal flag Zero dell'ALU
- **Jump (completa)**
 - $\text{PC} \leq \{ \text{PC}[31:28] , \{ \text{IR}[25:0] , '00' \} \}$

4. Memory access / R-type instruction completion

- Load
 - $\text{MDR} \leq \text{Memory}[\text{ALUOut}]$
- Store (completa)
 - $\text{Memory}[\text{ALUOut}] \leq \text{B}$
- Istruzione Logico-Aritmetica (R-type) (completa)
 - $\text{Reg}[\text{IR}[15:11]] \leq \text{ALUOut}$

5. Memory read completion

- Load
 - $\text{Reg}[\text{IR}[20:16]] \leq \text{MDR}$

Passi realizzati per eseguire una classe di istruzioni. Vista complessiva

Step name	Action for R-type instructions	Action for memory-reference instructions	Action for branches	Action for jumps
Instruction fetch	IR = Memory[PC] PC = PC + 4			
Instruction decode/register fetch	A = Reg [IR[25-21]] B = Reg [IR[20-16]] ALUOut = PC + (sign-extend (IR[15-0]) << 2)			
Execution, address computation, branch/ jump completion	ALUOut = A op B	ALUOut = A + sign-extend (IR[15-0])	if (A ==B) then PC = ALUOut	PC = PC [31-28] (IR[25-0]<<2)
Memory access or R-type completion	Reg [IR[15-11]] = ALUOut	Load: MDR = Memory[ALUOut] or Store: Memory [ALUOut] = B		
Memory read completion		Load: Reg[IR[20-16]] = MDR		

Calcolatori Elettronici II
Datath multicycle - 22

F. Tortorella © 2007
Università degli Studi di Cassino

Valutazione del data path multicycle

- Ricordiamo la distribuzione stimata per le istruzioni:
 - Load: 25 %
 - Store: 10 %
 - Istruzioni logico-aritmetiche: 45 %
 - Branch: 15 %
 - Jump: 5 %

Calcolatori Elettronici II
Datath multicycle - 23

F. Tortorella © 2007
Università degli Studi di Cassino

Valutazione del data path multiciclo

Tipo di istruzione	Numero di cicli
Logico aritmetiche	4
Load	5
Store	4
Branch	3
Jump	3

Valutiamo il numero medio di cicli per istruzione (CPI):

$$\begin{aligned} \text{CPI} &= \\ &.45 \cdot 4 + .25 \cdot 5 + .1 \cdot 4 + .15 \cdot 3 + .05 \cdot 3 \\ &= 4.05 \end{aligned}$$

Se avessimo adottato lo stesso numero di cicli avremmo avuto un CPI=5.

Calcolatori Elettronici II
Datapath multiciclo - 24

F. Tortorella © 2007
Università degli Studi
di Cassino

Definizione del controllo

- I valori dei segnali di controllo (abilitazioni) dipendono da:
 - Istruzione in esecuzione
 - Passo corrente
- Si usano le informazioni raccolte per specificare una macchina a stati finiti (MSF) che sia capace di generare le sequenze opportune di abilitazioni.

Calcolatori Elettronici II
Datapath multiciclo - 25

F. Tortorella © 2007
Università degli Studi
di Cassino

Reti sequenziali

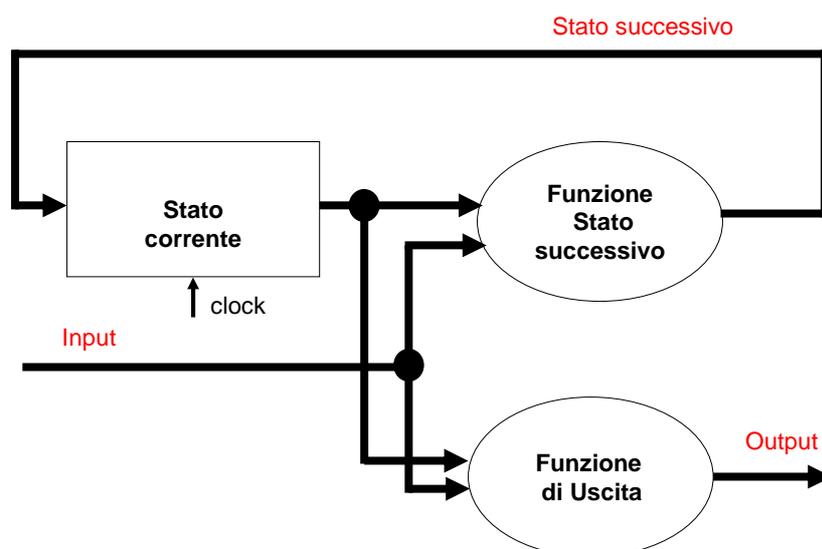
- una rete sequenziale è un circuito logico avente n ingressi (x_1, x_2, \dots, x_n) ed m uscite (y_1, y_2, \dots, y_m) , ciascuno dei quali assume valori binari (0/1).
- ad ogni istante, il valore delle uscite dipende sia dagli ingressi presenti, sia dalla sequenza degli ingressi precedenti.
- in ogni istante, si può identificare uno *stato* S in cui la rete si trova, in funzione della sequenza degli ingressi precedenti.
- le uscite possono quindi essere definite come funzioni degli ingressi correnti e dello stato corrente.
- ugualmente, lo stato successivo può essere determinato in funzione dello stato corrente e degli ingressi correnti.



Modello di macchina a stati finiti

Calcolatori Elettronici II
Datath multicycle - 26

F. Tortorella © 2007
Università degli Studi
di Cassino



Calcolatori Elettronici II
Datath multicycle - 27

F. Tortorella © 2007
Università degli Studi
di Cassino

Esempi di macchine sequenziali

- Registri (RS, D, JK)
- Registri a scorrimento
- Contatori

Come si definisce lo stato ?
 Quanti stati diversi può assumere la macchina ?
 Come si definisce la funzione stato successivo ?
 Come si definisce la funzione di Uscita ?

Registro JK

Stato: valore di Q

Num. di stati: 2

F. di stato successivo:

F. di uscita: valore di Q

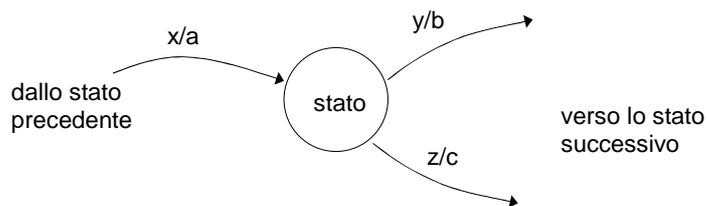
$$\left\{ \begin{array}{ll} J=0 \text{ e } K=0 \implies & Q'=Q \\ J=1 \text{ e } K=0 \implies & Q'=1 \\ J=0 \text{ e } K=1 \implies & Q'=0 \\ J=1 \text{ e } K=1 \implies & Q'=\bar{Q} \end{array} \right.$$

Calcolatori Elettronici II
Datapath multiciclo - 28

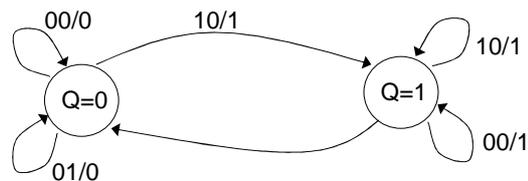
F. Tortorella © 2007
 Università degli Studi
 di Cassino

Diagramma di stato

Strumento per descrivere graficamente una macchina sequenziale.



Registro JK

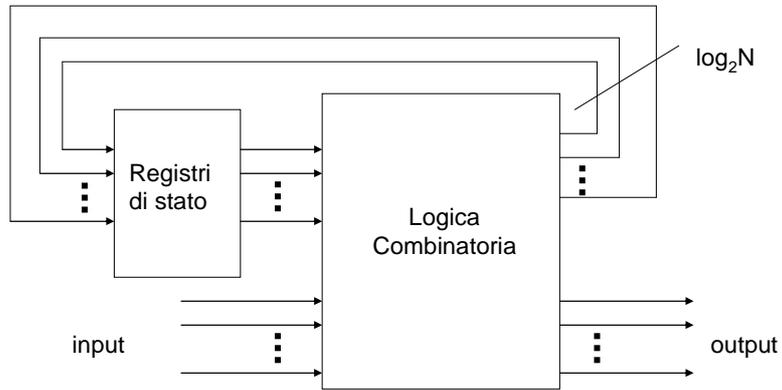


Calcolatori Elettronici II
Datapath multiciclo - 29

F. Tortorella © 2007
 Università degli Studi
 di Cassino

Realizzazione di una macchina a stati finiti

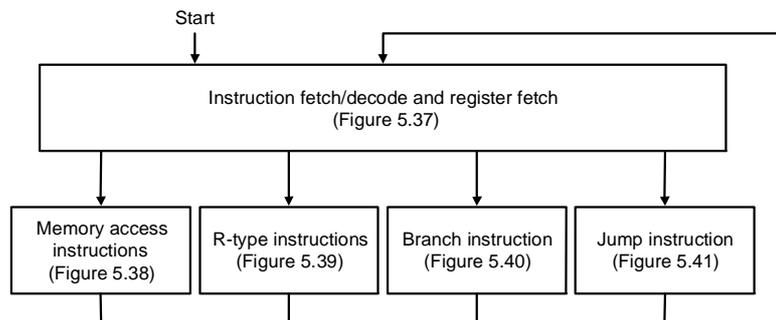
MSF ad N stati \Rightarrow lo stato è rappresentabile con $\log_2 N$ bit



Calcolatori Elettronici II
Datapath multiciclo - 30

F. Tortorella © 2007
Università degli Studi
di Cassino

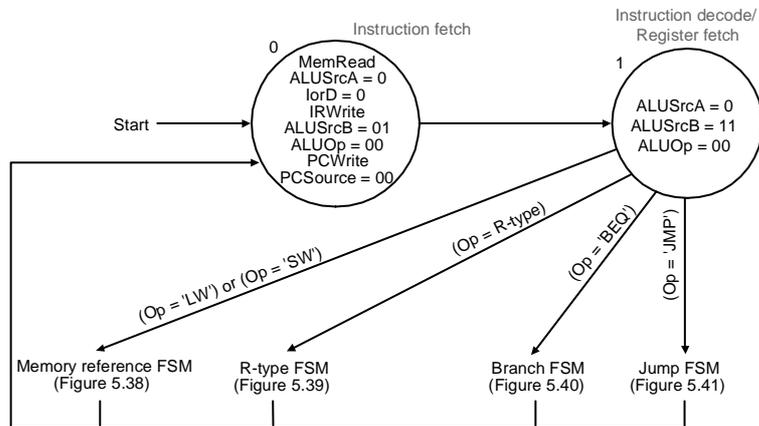
Vista ad alto livello della MSF



Calcolatori Elettronici II
Datapath multiciclo - 31

F. Tortorella © 2007
Università degli Studi
di Cassino

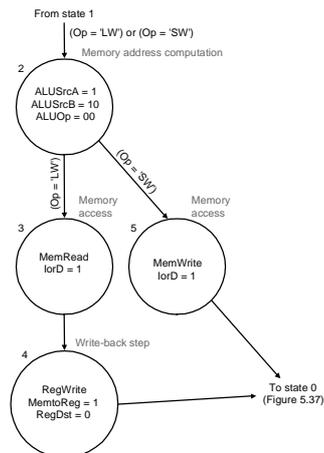
MSF: i primi due stati (comuni)



Calcolatori Elettronici II
Datapath multiciclo - 32

F. Tortorella © 2007
Università degli Studi
di Cassino

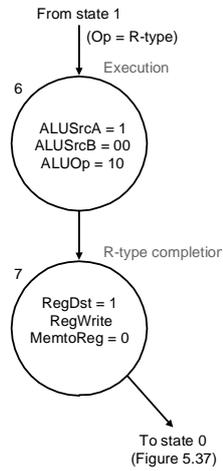
MSF: istruzioni lw/sw



Calcolatori Elettronici II
Datapath multiciclo - 33

F. Tortorella © 2007
Università degli Studi
di Cassino

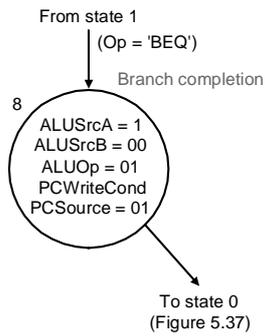
MSF: istruzioni di tipo R



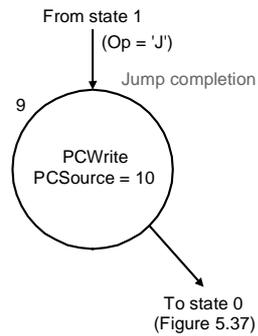
Calcolatori Elettronici II
Datath multicycle - 34

F. Tortorella © 2007
Università degli Studi
di Cassino

MSF: istruzioni di salto

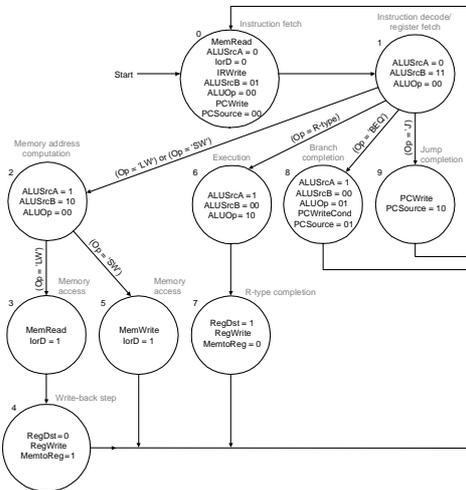


Calcolatori Elettronici II
Datath multicycle - 35



F. Tortorella © 2007
Università degli Studi
di Cassino

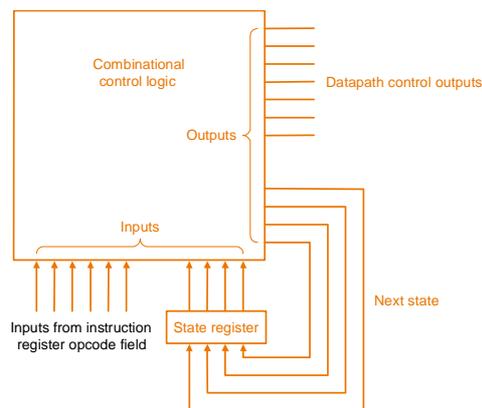
MSF: diagramma a stati finiti completo



Calcolatori Elettronici II
Datapath multiciclo - 36

F. Tortorella © 2007
Università degli Studi
di Cassino

Implementazione della MSF



Calcolatori Elettronici II
Datapath multiciclo - 37

F. Tortorella © 2007
Università degli Studi
di Cassino

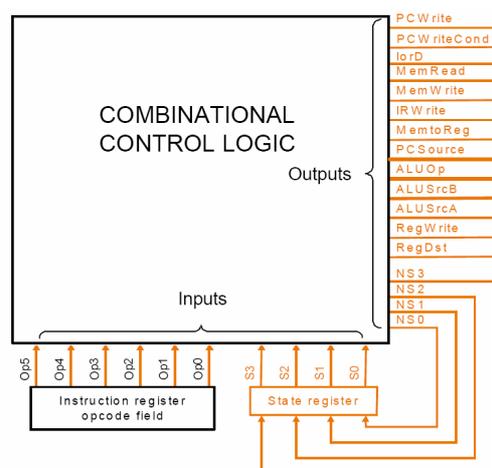
Implementazione della MSF

- Codifica degli stati
 - 10 stati → 4 bit per codificare lo stato
- Definizione tabella dello stato prossimo
 - Sulla base dello stato corrente e dell'ingresso (valore di op)
- Definizione delle uscite (abilitazioni)
 - Funzione dello stato corrente (macchina di Moore)

Calcolatori Elettronici II
Datapath multicycle - 38

F. Tortorella © 2007
Università degli Studi
di Cassino

Implementazione della MSF



Calcolatori Elettronici II
Datapath multicycle - 39

F. Tortorella © 2007
Università degli Studi
di Cassino

Implementazione della MSF

Output	Current states	Op
PCWrite	State0+State 9	
IorD	State3+State5	
MemRead	State0+State3	
IRWrite	State0	
		Definizione stato prossimo
NextState0	State4+State5+State7+State8+State9	
NextState1	State0	
NextState2	State1	(Op='lw')+(Op='sw')
...	...	
NextState5	State2	(Op='sw')
...	...	

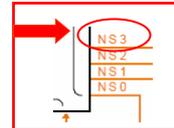
Definizione delle uscite

Definizione stato prossimo

Calcolatori Elettronici II
Datapath multiciclo - 40

F. Tortorella © 2007
Università degli Studi di Cassino

Implementazione della MSF



Output	Current states	Op
NextState0	State4+State5+State7+State8+State9	
NextState1	State0	
NextState2	State1	(Op='lw')+(Op='sw')
NextState3	State2	(Op='lw')
NextState4	State3	
NextState5	State2	(Op='sw')
NextState6	State1	(Op='R-type')
NextState7	State6	
NextState8	State1	(Op='beq')
NextState9	State1	(Op='jmp')

Stati che hanno $NS_3=1$

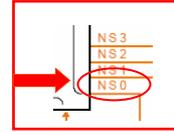
$$NS_3 = State1 \cdot (Op = 'beq') + State1 \cdot (Op = 'jmp')$$

$$NS_3 = \bar{S}_3 \bar{S}_2 \bar{S}_1 S_0 \bar{Op}_5 \bar{Op}_4 \bar{Op}_3 \bar{Op}_2 \bar{Op}_1 \bar{Op}_0 + \bar{S}_3 \bar{S}_2 \bar{S}_1 S_0 \bar{Op}_5 \bar{Op}_4 \bar{Op}_3 \bar{Op}_2 \bar{Op}_1 \bar{Op}_0$$

Calcolatori Elettronici II
Datapath multiciclo - 41

F. Tortorella © 2007
Università degli Studi di Cassino

Implementazione della MSF



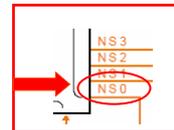
Output	Current states	Op
NextState0	State4+State5+State7+State8+State9	
NextState1	State0	
NextState2	State1	(Op='lw')+(Op='sw')
NextState3	State2	(Op='lw')
NextState4	State3	
NextState5	State2	(Op='sw')
NextState6	State1	(Op='R-type')
NextState7	State6	
NextState8	State1	(Op='beq')
NextState9	State1	(Op='jmp')

Stati che hanno
NS₀=1

Calcolatori Elettronici II
Datapath multiciclo - 42

F. Tortorella © 2007
Università degli Studi
di Cassino

Implementazione della MSF



NS₀ è la somma
logica di questi 5
termini

$$\begin{aligned}
 \text{NextState1} &= \text{State0} = \bar{S}_3 \cdot \bar{S}_2 \cdot \bar{S}_1 \cdot \bar{S}_0 \\
 \text{NextState3} &= \text{State2} \cdot (\text{Op}[5-0] = \text{lw}) \\
 &= \bar{S}_3 \cdot \bar{S}_2 \cdot S_1 \cdot \bar{S}_0 \cdot \text{Op}_5 \cdot \bar{\text{Op}}_4 \cdot \bar{\text{Op}}_3 \cdot \bar{\text{Op}}_2 \cdot \text{Op}_1 \cdot \text{Op}_0 \\
 \text{NextState5} &= \text{State2} \cdot (\text{Op}[5-0] = \text{sw}) \\
 &= \bar{S}_3 \cdot \bar{S}_2 \cdot S_1 \cdot \bar{S}_0 \cdot \text{Op}_5 \cdot \bar{\text{Op}}_4 \cdot \text{Op}_3 \cdot \bar{\text{Op}}_2 \cdot \text{Op}_1 \cdot \text{Op}_0 \\
 \text{NextState7} &= \text{State6} = \bar{S}_3 \cdot S_2 \cdot S_1 \cdot \bar{S}_0 \\
 \text{NextState9} &= \text{State1} \cdot (\text{Op}[5-0] = \text{jmp}) \\
 &= \bar{S}_3 \cdot \bar{S}_2 \cdot \bar{S}_1 \cdot S_0 \cdot \text{Op}_5 \cdot \bar{\text{Op}}_4 \cdot \bar{\text{Op}}_3 \cdot \bar{\text{Op}}_2 \cdot \text{Op}_1 \cdot \bar{\text{Op}}_0
 \end{aligned}$$

Calcolatori Elettronici II
Datapath multiciclo - 43

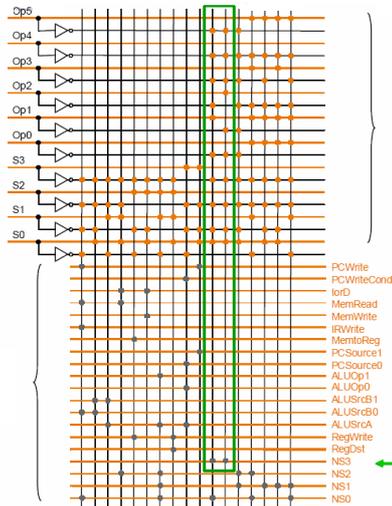
F. Tortorella © 2007
Università degli Studi
di Cassino

Implementazione della MSF

Rete combinatoria tramite PLA

OR plane:
Calcola le somme

*Calcolatori Elettronici II
Datapath multiciclo - 44*



AND plane:
Calcola i mintermini

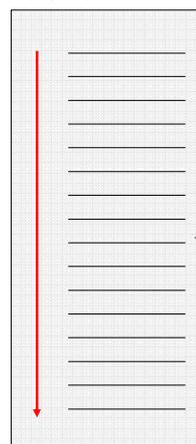
← NS₃

*F. Tortorella © 2007
Università degli Studi
di Cassino*

Eccezioni

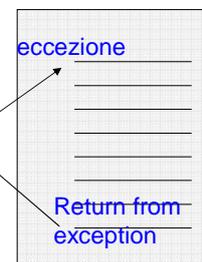
- Trasferimento imprevisto del controllo sulla base del verificarsi di un evento.
- Il sistema risponde con un'azione opportuna all'eccezione.
 - Necessario salvare l'indirizzo dell'istruzione in corrispondenza della quale si è generata l'eccezione
- Al termine viene restituito il controllo al programma interrotto
- Necessario salvare e poi ripristinare lo stato del programma interrotto.

Programma utente



Flusso di controllo normale: sequenza, branch, jump

System exception handler o Interrupt Service Routine (ISR)



*Calcolatori Elettronici II
Datapath multiciclo - 45*

*F. Tortorella © 2007
Università degli Studi
di Cassino*

Tipi di eccezioni

- **Interruzioni**

- Causate da eventi esterni
- Asincrone rispetto al programma in esecuzione
- Possono essere gestite tra due istruzioni consecutive
- Semplice sospensione e ripresa del programma utente

- **Eccezioni**

- Causate da eventi interni
 - Condizioni di eccezione (overflow)
 - Errori (parity)
 - Faults (pagine non residenti in memoria)
- Sincrone rispetto al programma in esecuzione
- La condizione di eccezione deve essere risolta dall'handler
- Se la condizione è risolubile, l'istruzione può essere rieseguita ed il programma continuato. In caso contrario, il programma deve essere terminato prematuramente.

Calcolatori Elettronici II
Datapath multicycle - 46

F. Tortorella © 2007
Università degli Studi
di Cassino

Gestione delle eccezioni

- Due esigenze vanno considerate per una corretta gestione delle eccezioni:
 - Salvare lo stato del programma interrotto
 - PC, altre informazioni
 - Individuare la routine da eseguire in funzione della causa che ha generato l'eccezione.
- Salvataggio dello stato
 - Push sullo stack (Vax, Motorola 68K, 80x86)
 - Salvataggio in registri speciali (MIPS)
 - Registri ombra (Motorola 88K)

Calcolatori Elettronici II
Datapath multicycle - 47

F. Tortorella © 2007
Università degli Studi
di Cassino

Gestione delle eccezioni: indirizzamento dell'ISR

- Diverse soluzioni:
- **Interrupt vettorizzato** (68K, 80x86,...)
 - Viene fornito un valore (vector) che è un indice in una tabella (vector table) che contiene gli indirizzi delle varie ISR:
 $PC = Mem[IVT_base + vector][00]$. Causa \rightarrow vector
- **Tabella delle ISR** (Sparc, 88K,...)
 - Viene fornito uno spiazzamento all'interno di un'area di memoria che contiene tutte le ISR:
 $PC = IT_base + offset | 0000$. Causa \rightarrow offset
- **Indirizzo fisso** (MIPS)
 - Si salta ad un unico indirizzo: unica ISR che si occupa di identificare la causa e avviare la routine opportuna (effettivamente nel MIPS si considerano due entry).
 $PC = Exc_address$

Calcolatori Elettronici II
Datapath multicycle - 48

F. Tortorella © 2007
Università degli Studi
di Cassino

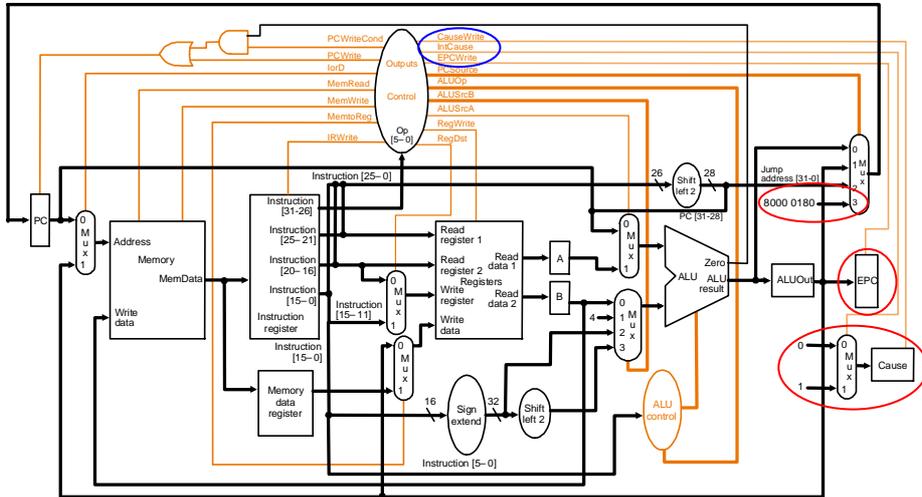
Gestione delle eccezioni: come cambia il controllo ?

- Consideriamo per semplicità due sole cause possibili di eccezione:
 - Istruzione non riconosciuta (generata dal controllo a valle della decodifica dell'istruzione)
 - Overflow (generata dall'ALU nella fase execute di un'istruzione R-type)
- Nuovi compiti da eseguire:
 - Salvare in un registro opportuno (**Cause**) la causa dell'eccezione
 - Salvare l'indirizzo dell'istruzione che ha causato l'eccezione (PC-4) in un registro opportuno (**EPC**)
 - Saltare all'entry point dell'ISR

Calcolatori Elettronici II
Datapath multicycle - 49

F. Tortorella © 2007
Università degli Studi
di Cassino

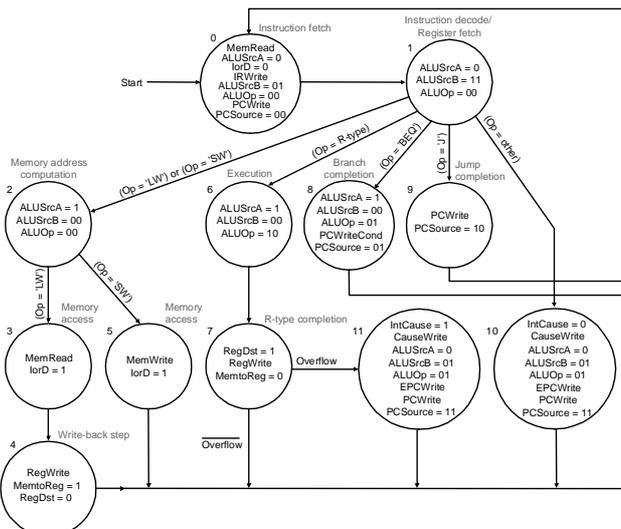
Eccezioni: Modifica del datapath e del controllo



Calcolatori Elettronici II
Datapath multiciclo - 50

F. Tortorella © 2007
Università degli Studi
di Cassino

Eccezioni: Modifica della MSF



Calcolatori Elettronici II
Datapath multiciclo - 51

F. Tortorella © 2007
Università degli Studi
di Cassino

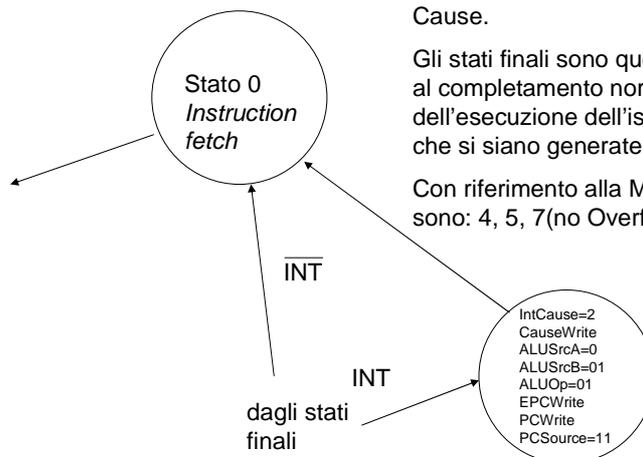
Interruzioni

- Sono essere gestite tra due istruzioni consecutive
- Semplice sospensione e ripresa del programma utente
- Al termine dell'esecuzione dell'istruzione corrente si verifica se è attivo un segnale di interruzione
- In tal caso si esegue il cambio di contesto descritto per le eccezioni, con la differenza che nel registro Cause viene codificato l'insorgere di un'interruzione.

Calcolatori Elettronici II
Datapath multicycle - 52

F. Tortorella © 2007
Università degli Studi
di Cassino

Interruzioni: Modifica della MSF



Assumiamo che l'interruzione sia codificata con il valore 2 nel registro Cause.

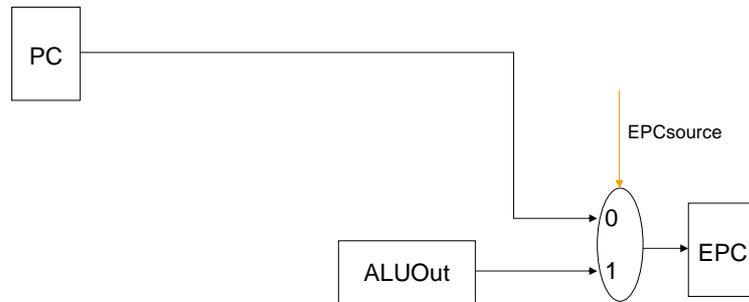
Gli stati finali sono quelli corrispondenti al completamento normale dell'esecuzione dell'istruzione (senza che si siano generate eccezioni).

Con riferimento alla MSF precedente, sono: 4, 5, 7(no Overflow), 8, 9

Calcolatori Elettronici II
Datapath multicycle - 53

F. Tortorella © 2007
Università degli Studi
di Cassino

Interruzioni: Modifica del datapath e del controllo



Calcolatori Elettronici II
Datapath multicycle - 54

F. Tortorella © 2007
Università degli Studi
di Cassino