Architettura degli Elaboratori

Appello del 6 luglio 2012

Riportare su tutti i fogli consegnati nome, cognome, numero di matricola, corso A/B, e la sigla NEW (per nuovo ordinamento), oppure OLD-0 (per vecchio ordinamento, nuovo programma), oppure OLD-1 (per vecchio ordinamento, vecchio programma). I risultati verranno pubblicati sulle pagine web del corso/dei docenti appena disponibili.

Domanda 1 (tutti)

Dato il seguente microprogramma:

0. (RDY1, zero(M[IND] – IN) = 0 –) nop, 0;

   (= 1 1) reset RDY1, set ACK1, 0 $\rightarrow$ I, 1;

   (= 1 0) reset RDY1, set ACK1, IN $\rightarrow$ M[IND], 0

1. (I_0, segno(M[I_m]), ACK2 = 0 0 -) I + 1 $\rightarrow$ I, 1;

   (= 0 1 -) I + 1 $\rightarrow$ I, 0 $\rightarrow$ M[I_m],1;

   (= 1 - 0) nop, 1;

   (= 1 - 1) set RDY2, reset ACK2, 0

a) definire completamente l'automa Parte Controllo, giustificando formalmente come si arriva a tale definizione, e progettare la rete sequenziale Parte Controllo;

b) spiegare formalmente la ragione per cui la Parte Controllo contiene un registro impulsato;

c) ricavare la funzione di transizione dello stato interno della Parte Operativa relativamente alla generica locazione della memoria M e giustificare formalmente il procedimento adottato.

Domanda 2 (tutti)

La seguente computazione

int A[N]; int x = 0;

$\forall i = 0 .. N\text{-}1:$ $x = x + F(A[i])$

è eseguita su un elaboratore con gerarchia di memoria memoria principale – cache primaria su domanda e con blocchi di σ parole. Ricavare il valore minimo della banda della memoria principale che permetta di ottenere un determinato valore del tempo di completamento T_c e un determinato valore dell'efficienza relativa della cache ε.

Sono noti il ciclo di clock τ della CPU e la latenza di trasmissione T_{tr} dei collegamenti inter-chip. Non sono note informazioni sulla funzione F e sul suo tempo di servizio, eccetto che è lecito trascurare il numero di fault di cache per il codice del processo.

Domanda 3 (NEW, OLD-0)

Per una architettura di CPU pipeline scalare D-RISC dare una descrizione dettagliata del funzionamento dell'Unità Istruzioni e, sulla base di tale descrizione, dimostrare che il tempo di servizio ideale di tale unità è uguale a un ciclo di clock.

Domanda 3 (OLD-1)

Una architettura D-RISC con dati a 64 bit utilizza un coprocessore matematico, visto come unità di I/O; per realizzare l'operazione di radice quadrata di un numero reale. Si suppone che i numeri reali siano rappresentati su una parola di 64 bit.

a) Scrivere una procedura D-RISC che calcola la radice quadrata di un numero reale in memoria e restituisce il risultato in memoria, senza prevedere commutazione di contesto della CPU. Conoscendo il tempo di servizio T_{sqrt} del coprocessore, valutare il tempo di completamento.

b) Scrivere una versione LC della procedura.

Sintesi della soluzione

(da completare con spiegazioni ulteriori)

Commento
TUTTE le domande del compito sono fortemente riconducibili a domande delle raccolte di esercizi e quesiti che sono state distribuiti e raccomandati, e in gran parte svolti a lezione:
Domanda 1:raccolta 1, n. 1 e qualunque esercizio di realizzazione di unità di elaborazione.
Domanda 2: raccolta 3, qualunque esercizio sulla valutazione di programmi con gerarchia memoria principale – cache.
Domanda 3: raccolta 3, n. 5.
Domanda 3, OLD-1: raccolta 2, n. 7.

Domanda 1 (tutti)

a) Come ogni automa, PC è una quintupla (X, Z, S, ω, σ), con X insieme degli stati d'ingresso, Z insieme degli stati di uscita, S insieme degli stato interni, ω funzione delle uscite:

$$Z(t) = \omega\,(X(t), S(t))$$

e σ funzione di transizione dello stato interno:

$$S(t+1) = \sigma\,(X(t), S(t))$$

Per l'automa PC la determinazione della quintupla dipende dal microporgramma, ed è ricavabile formalmente dal microprogrammma stesso.

A tale scopo, viene prima fissato come determinare l'insieme degli stati interni: questi sono in corrispondenza biunivoca con le etichette delle microistruzioni. Nel nostro caso S = (0, 1). Fatto questo, tutta la rimanente definizione dell'automa deriva automaticamente dalla struttura di frase del microprogramma.

L'insieme degli stati d'ingresso X è dato da tutte le combinazioni di valori delle variabili d'ingresso (variabili di condizionamento) RDY1, zero(M[IND] – A), I_0, ACK2, H, quindi

X = (RDY1, zero(M[IND] – A), I_0, segno(M[I_m]), ACK2, = 0 - - - -, 11 - - -, 10 - - -,
- - 0 0 -, - - 0 1 -, - - 1 – 0, - - 1 – 1)

Gli stati di uscita sono tutte le combinazioni dei valori delle variabili di uscita (variabili di controllo). La loro conoscenza richiede che sia stata realizzata la PO a partire dal microprogramma; le variabili sono le seguenti:

$$\beta_{RDY1} = \beta_{ACK1},\ \alpha_I,\ \beta_I,\ \alpha_M,\ \beta_M,\ \beta_{RDY2} = \beta_{ACK2}$$

Per soddisfare la condizione necessaria di correttezza della PO, la memoria M deve essere realizzata a doppio indirizzamento (IND, I_m - doppia lettura e doppia scrittura); la variabile α_M controlla il commutatore sull'ingresso del dato (da IN oppure 0).

L'insieme degli stati di uscita di PC, in corrispondenza con le micro-operazioni da eseguire nelle frasi, è dato da:

Z = (β_{RDY1}, α_I, β_I, α_M, β_M, β_{RDY2} = 0 – 0 – 0 0, // nop//
1 0 1 – 0 0, // micro-operazione seconda frase di 0 //
1 – 0 0 1 0, // micro-operazione terza frase di 0 //
0 1 1 – 0 0, // micro-operazione prima frase di 1 //
0 1 1 1 1 0, // micro-operazione seconda frase di 1 //
0 – 0 – 0 1, // micro-operazione quarta frase di 1 //)

Le funzione dell'automa sono definite secondo il seguente procedimento:

- ω: per ogni stato interno presente (ad esempio 0) e per ogni stato d'ingesso significativo in quello stato interno (ad esempio (RDY1, zero(M[IND] – IN) = 1 1)), risulta determinata univocamente la micro-operazione (reset RDY1, set ACK1, 0 → I) e quindi lo stato di uscita ((β_{RDY1}, α_I, β_I, α_M, β_M, β_{RDY2} = 1 0 1 – 0 0). Si tratta di un automa di Mealy;
- σ: per ogni stato interno presente (ad esempio 0) ed ogni stato d'ingesso significativo in quello stato interno (ad esempio (RDY1, zero(M[IND] – IN) = 1 1)), risulta determinata univocamente la microistruzione successiva e quindi lo stato interno successivo (1).

Codificata la variabile dello stato interno come y =0 (stato 0), y = 1 (stato 1), le funzioni dell'automa sono quindi implementate come segue:

ω:

$$\beta_{RDY1} = \bar{y}\ RDY1$$
$$\alpha_I = y\,\overline{I_0}$$
$$\beta_I = \bar{y}\ RDY1\ zero + y\ \overline{I_0}$$
$$\alpha_M = y\ \overline{I_0}\ segno$$
$$\beta_M = \bar{y}\ RDY1\ \overline{zero} + y\ \overline{I_0}\ segno$$
$$\beta_{RDY2} = y\ I_0\ ACK2$$

σ:

$$Y = \bar{y}\ RDY1 + y\,\overline{I_0} + y\,I_0\ ACK2$$

La realizzazione della PC consiste nelle corrispondenti reti combinatorie e da un registro impulsato avente in ingresso la variabile dello stato interno successivo Y e in uscita la variabile dello stato interno presente y.

b) Come in ogni rete sequenziale, la memorizzazione dello stato interno in un registro impulsato ha lo scopo di stabilizzare le variabili dello stato interno presente in modo che la loro variazione avvenga contemporaneamente alla variazione delle variabili d'ingresso anche in presenza di ritardi non costanti della parte combinatoria. Completare la risposta secondo la sez. 2.3, Cap. III.

c) La Parte Operativa è una rete sequenziale i cui stati interni sono tutte le possibili combinazioni dei contenuti dei registri (inclusi quelli d'ingresso), i cui stati d'ingresso sono tutte le possibili combinazioni dei valori delle variabili di controllo e degli ingressi esterni, e i cui stati di uscita sono tutte le possibili combinazioni delle variabili di condizionamento e delle uscite esterne. La rete sequenziale realizza un automa di Moore.

Una volta costruito il grafo della PO a partire dal microprogramma, le funzioni ω_{PO} e σ_{PO} si ricavano con un procedimento di analisi che applica i concetti sulle reti sequenziali (vedi precedente punto *b)* in particolare). Ad ogni ciclo di clock, per ogni registro R, lo stato successivo è dato dal valore stabile assunto dalle variabili d'ingresso del registro stesso in_R a condizione che, in tale ciclo di clock, la scrittura sia abilitata. Il valore delle variabili d'ingresso è il risultato di una funzione applicata allo stato interno presente e allo stato d'ingresso della PO; la funzione corrisponde al sottografo che si ottiene percorrendo il grafo a ritroso dall'ingresso del registro fino ad incontrare, in tutti i possibili percorsi, solo registri o costanti.

Per la generica locazione della memoria M, la definizione della funzione σ_{PO}, mediante un formalismo di programmazione, è la seguente:

$$in_{M[i]} = \textbf{\textit{when}}\ (\beta_M\ \textbf{\textit{and}}\ i = IND\ \textbf{\textit{or}}\ i = I_m)\ \textbf{\textit{do if}}\ \alpha_M\ \textbf{\textit{then}}\ 0\ \textbf{\textit{else}}\ IN$$

Domanda 2 (tutti)

Per la definizione di efficienza relativa e per il modo di valutare il tempo di completamento, vale il seguente sistema di equazioni:

$$\begin{cases} \varepsilon = \dfrac{T_{c-id}}{T_c} \\ T_c = T_{c-id} + T_{fault} \end{cases}$$

Eliminando $T_{c\text{-}id}$ si ottiene:

$$T_{fault} = (1 - \epsilon)\, T_c$$

La computazione è caratterizzata da sola località su A; l'insieme di lavoro è quindi dato dal solo blocco corrente di A più i blocchi del codice. Non ci sono scritture in memoria che abbiano impatto sul tempo di servizio (al più una sola scrittura all'uscita del loop).

Detta B_M la banda della memoria principale interallacciata su *m* moduli, l'espressione per T_{fault} è:

$$T_{fault} = N_{fault} * T_{trasf} = \frac{N}{\sigma}\,(2\,T_{tr} + \frac{\sigma}{B_M} + m\tau)$$

Essendo:

$$\frac{N}{\sigma}\left(2\,T_{tr} + \frac{\sigma}{B_M} + m\tau\right) > \frac{N}{\sigma}\,(2\,T_{tr} + \frac{\sigma}{B_M})$$

si ha:

$$\frac{N}{\sigma}\left(2\,T_{tr} + \frac{\sigma}{B_M}\right) < (1 - \epsilon)\, T_c$$

da cui si ricava il valore minimo della banda di memoria:

$$B_M > \frac{\sigma\, N}{(1 - \varepsilon)\, T_c\, \sigma - 2\, T_{tr}\, N}$$

Domanda 3 (NEW, OLD-0)

Ad ogni ciclo di clock IU testa tutte le interfacce per verificare le seguenti condizioni:

- presenza di una nuova istruzione da IM, sua validità e sua abilitazione
- presenza di un nuovo valore da EU.

La condizione di validità prevede il test della situazione di delayed branch in atto e, se non lo è, il confronto tra l'identificatore unico (IC) associato all'istruzione e quello corrente in IU. Se l'istruzione è valida, viene decodificata ed eseguita *nello stesso ciclo di clock* secondo le seguenti azioni:

- se si tratta di un'istruzione aritmetica, viene incrementato il semaforo del registro destinazione e l'istruzione viene inviata alla EU;
- per le istruzioni di salto operanti su registri generali e di Load/Store, viene valutata l'abilitazione testando i semafori associati ai registri di input riferiti. Se l'istruzione è abilitata viene eseguita secondo la semantica (inserire spiegazioni sintetiche), effettuando le comunicazioni opportune a IM, EU, DM. Altrimenti l'istruzione viene posta in una struttura dati di attesa W (memoria di registri): un campo del semaforo SEM[i] contiene un bit di presenza e il riferimento alla posizione in W nella quale è memorizzata una istruzione in attesa del valore aggiornato di RG[i]. Nel ciclo di clock successivo IU passa quindi a testare la presenza di una nuova istruzione da IM, come indicato all'inizio.

Nello stesso ciclo di clock, se viene ricevuta una coppia (Val, j) da EU, viene scritto il valore Val in RG[j], viene decrementato SEM[j] e, se prima del decremento SEM[j] valeva uno e contiene un riferimento significativo ad una istruzione in attesa, viene verificata l'abilitazione di tale istruzione; se abilitata, viene eseguita nel ciclo di clock successivo secondo il comportamento descritto sopra.

È stato quindi dimostrato che il tempo di servizio ideale, cioè il tempo di servizio della IU considerata isolata dal contesto dell'intera CPU, è uguale ad un ciclo di clock in quanto tutte le azioni relative ad *una* istruzione sono eseguite in parallelo in una sola microistruzione.

Domanda 3 (OLD-1)

a) La procedura utilizza Memory Mapped I/O per la cooperazione tra CPU e unità di I/O mediante istruzioni di LOAD e STORE. Per verificare se il risultato della radice quadrata è pronto, e senza affidare compiti particolari all'unità di I/O, si può utilizzare un flag nella memoria condivisa di I/O, testandolo ripetutamente finché l'unità non lo pone ad uno.

```
        LOAD   Rparam, 0, Rx            // memoria principale //
        STORE  Ri/o, Roffset_1, Rx      // memoria I/O //
LOOP:   LOAD   Rflag, 0, Rsync          // memoria I/O //
        IF = 0 Rsync, LOOP
        STORE  Rflag, 0, Rzero          // memoria I/O //
        LOAD   Ri/o, Roffset_2, Rz      // memoria I/O //
        STORE  Rris, 0, Rz              // memoria principale //
        GOTO   Rret
```

Alternativamente, si può utilizzare una segnalazione esplicita da parte dell'unità via interruzione, usando l'istruzione WAITINT del D-RISC e mascherando preventivamente tutte le altre interruzioni.

```
        LOAD   Rparam, 0, Rx
        STORE  Ri/o, Roffset_1, Rx
        MASKINT  Rmask
        WAITINT  Ri/0, classe_coprocessore, -, -
        LOAD   Ri/o, Roffset_2, Rz
        STORE  Rris, 0, Rz
        GOTO   Rret
```

Nella prima versione, il tempo di completamento è dato da:

$$T_c = T_{sqrt} + 2\,T_{LD/ST\text{-}MEM} + 3\,T_{LD/ST\text{-}I/O} + T_{GOTO}$$

Non sono state valutate le due istruzioni del loop di sincronizzazione (LOAD Rflag, IF), il cui ritardo è assorbito in T_{sqrt} (una valutazione del tutto analoga si avrebbe con la seconda versione). Quindi:

$$T_c = T_{sqrt} + 6\,Tch + 2\,T_{ex\text{-}LD/ST\text{-}MEM} + 3\,T_{ex\text{-}LD/ST\text{-}I/O} + T_{ex\text{-}GOTO} = T_{sqrt} + 8\,(2\tau + t_M) + 3\,(2\tau + t_{I/O}) + \tau$$

dove

$$T_{I/O} = 2\,T_{bus} + \tau_{I/O}$$

b) La versione LC considera l'unità di I/O come un processo esterno, con il quale il processo centrale coopera mediante comunicazioni:

```
        send (CH_I/O_1, param);
        receive (CH_I/O_2, ris);
        return
```

Il supporto a tempo di esecuzione delle send-receive, che per le ipotesi esiste sia su CPU che su I/O; utilizza la memoria condivisa di I/O e realizza automaticamente la sincronizzazione e la condizione di attesa, in generale passiva, del processo centrale.