Architettura degli Elaboratori

a.a. 2012/13 - secondo appello, 1 luglio 2013

Riportare nome, cognome, numero di matricola e corso A/B

Domanda 1

Dato il seguente microprogramma (l'unità ha registri d'ingresso IN, IND):

0. (RDYIN, OP = 0 –) nop, 0;
   (= 1 0) reset RDYIN, set ACKIN, 0 → I, 0 → S, 1;
   (= 1 1) reset RDYIN, set ACKIN, IN → M[IND], 0
1. (I_0, OR(M[I_m]), ACKOUT = 0 0 –) I + 1 → I, 1;
   (= 0 1 –) I + 1 → I, S + M[I_m] → S,1;
   (= 1 – 0) nop, 1;
   (= 1 – 1) set RDYOUT, reset ACKOUT, S → OUT, 0

definire l'*automa* Parte Controllo dell'unità, e progettare la *rete sequenziale* Parte Controllo.

Domanda 2

Si consideri il sottosistema Memoria Istruzioni (IM) di una CPU D-RISC pipeline scalare con comunicazioni a singola bufferizzazione.

a) Spiegare l'architettura concreta di IM.

b) Scrivere il microprogramma dell'unità, facente parte di IM, preposta alla generazione dello stream di indirizzi fisici delle istruzioni. Per semplicità, trascurare i controlli di protezione e le azioni per la commutazione di contesto. Sono predefinite le funzioni:

- *traduci(identificatore_pagina_logica)*, che restituisce la coppia di valori *(esito, ipf)* dove *esito* = (0 = successo, 1 = fault di pagina, 2 = fault di memoria associativa). Se *esito* = 0, *ipf* è l'identificatore di pagina fisica;
- *rimpiazza ( )*, che restituisce la posizione dell'entrata della memoria associativa da rimpiazzare.

Determinare il tempo di servizio ideale di tale unità nell'ipotesi che l'evento 2 abbia probabilità trascurabile.

c) Spiegare perché l'unità di cui al punto *b)* contiene una memoria associativa e non una memoria RAM.

Domanda 3

La seguente computazione, operante sugli array di interi *A[N]*, *B[N]*, *C[N]:*

$$\forall\ i = 0\ ..\ N-1\text{: } C[i] = A[i] * B[i]$$

è eseguita su una architettura D-RISC con CPU pipeline scalare in-order, con ciclo di clock τ, comunicazioni a singola bufferizzazione, e unità funzionali per operazioni aritmetiche su interi a 4 stadi.

La Cache Dati primaria è su domanda, Write-Through, con blocchi di σ parole. La cache secondaria on-chip è assunta avere probabilità di fault trascurabile per questa computazione. La memoria esterna è interallacciata con 8 moduli e ciclo di clock uguale a 100τ.

Determinare il tempo di servizio ideale e il tempo di servizio effettivo dell'*unità Cache Dati*.

Soluzione

Domanda 1

L'automa PC è definito dalla quintupla (S, X, Z, ω, σ), dove:

- l'insieme S degli stati interni è in corrispondenza biunivoca con le etichette del microprogramma:

  S0 $\Leftrightarrow$ 0

  S1 $\Leftrightarrow$ 1

- l'insieme X degli stati di ingresso è dato dalle combinazioni delle variabili di condizionamento RDYIN, OP, I_0, OR(M[I_m]), ACKOUT. Gli stati d'ingresso distinti sono quindi in corrispondenza biunivoca con le condizioni logiche distinte:

  X0 $\Leftrightarrow$ (RDYIN, OP = 0 –)

  X1 $\Leftrightarrow$ (RDYIN, OP = 1 0)

  X2 $\Leftrightarrow$ (RDYIN, OP = 1 1)

  X3 $\Leftrightarrow$ (I_0, OR(M[I_m]), ACKOUT = 0 0 –)

  X4 $\Leftrightarrow$ (I_0, OR(M[I_m]), ACKOUT = 0 1 –)

  X5 $\Leftrightarrow$ (I_0, OR(M[I_m]), ACKOUT = 1 – 0)

  X6 $\Leftrightarrow$ (I_0, OR(M[I_m]), ACKOUT = 1 – 1)

- l'insieme Z degli stati di uscita è dato dalle combinazioni delle variabili di controllo; tali configurazioni di variabili sono a loro volta associate alle microoperazioni che vengono eseguite in PO. Gli stati di uscita distinti sono quindi in corrispondenza biunivoca con le microoperazioni distinte:

  Z0 $\Leftrightarrow$ nop

  Z1 $\Leftrightarrow$ reset RDYIN, set ACKIN, 0 $\rightarrow$ I, 0 $\rightarrow$ S

  Z2 $\Leftrightarrow$ reset RDYIN, set ACKIN, IN $\rightarrow$ M[IND]

  Z3 $\Leftrightarrow$ I + 1 $\rightarrow$ I

  Z4 $\Leftrightarrow$ I + 1 $\rightarrow$ I, S + M[I_m] $\rightarrow$ S

  Z5 $\Leftrightarrow$ set RDYOUT, reset ACKOUT, S $\rightarrow$ OUT

- la funzione ω delle uscite esprime la seguente corrispondenza S × X $\rightarrow$ Z:

	X0	X1	X2	X3	X4	X5	X6
S0	**Z0**	**Z1**	**Z2**	**–**	**–**	**–**	**–**
S1	**–**	**–**	**–**	**Z3**	**Z4**	**Z0**	**Z5**

- la funzione σ di transizione dello stato interno esprime la seguente corrispondenza S × X $\rightarrow$ S:

	X0	X1	X2	X3	X4	X5	X6
S0	**S0**	**S1**	**S0**	**–**	**–**	**–**	**–**
S1	**–**	**–**	**–**	**S1**	**S1**	**S1**	**S0**

La progettazione della rete sequenziale PC implementa le funzioni ω e σ dell'automa PC mediante espressioni booleane. Queste si ricavano facilmente dal microprogramma applicando le corrispondenze definite per l'automa:

- funzione di transizione dello stati interno σ: detta *y* la variabile dello stato interno presente e *Y* la variabile dello stato interno successivo, si ha

$$Y = \bar{y}\, RDYIN\, \overline{OP} + y\, \bar{I}_0 + y\, I_0\, \overline{ACKOUT}$$

- funzione delle uscite ω: le variabili di controllo sono state ricavate all'atto della progettazione della PO; per questa, si osservi che la memoria M deve essere realizzata a doppio indirizzamento. Le variabili di controllo sono definite dalle seguenti espressioni booleane:

$$\beta_{RDYIN} = \beta_{ACKIN} = \bar{y}\, RDYIN$$

$$\beta_{RDYOUT} = \beta_{ACKOUT} = \beta_{OUT} = y\, I_0\, ACKOUT$$

$$\alpha_S = y\, \bar{I}_0\, OR(M[I_m]) \quad , \quad \beta_S = \bar{y}\, RDYIN\, \overline{OP} + y\, \bar{I}_0\, OR(M[I_m])$$

$$\alpha_I = y\, \bar{I}_0 \quad , \quad \beta_I = \bar{y}\, RDYIN\, \overline{OP} + y\, \bar{I}_0$$

$$\beta_M = \bar{y}\, RDYIN\, OP$$

Domanda 2

a) L'architettura di IM ha una struttura pipeline a due stadi, dei quali il primo (MMU_I) genera lo stream di indirizzi fisici delle istruzioni e il secondo è la Cache Istruzioni (CI). MMU_I ha anche un collegamento in ingresso da IU e un collegamento a domanda-risposta con la Cache Dati. La descrizione dettagliata di MMU_I è data al punto *b).*

CI è eventualmente decomponibile a sua volta in due stadi, dei quali il primo effettua la traduzione da indirizzo fisico a indirizzo di cache e il secondo contiene la memoria cache vera e propria. CI può essere implementata con un qualsiasi metodo di indirizzamento della cache, incluso il metodo diretto.

b) L'unità da studiare è MMU_I. Essa contiene una copia del contatore istruzioni (*IC1*) il cui valore è l'indirizzo logico dell'istruzione da inviare nello stream. *IC1* è incrementato ad ogni passo di generazione dello stream. Nondeterministicamente, e con priorità, MMU_I ascolta la comunicazione da IU (*newIC*) contenente il valore del nuovo indirizzo logico della sequenza di istruzioni in seguito al verificarsi di un salto; *newIC* è usato direttamente per la traduzione e *newIC + 1* viene scritto in *IC1*.

(La comunicazione da IU può anche contenere il nuovo indirizzo della Tabella di Rilocazione, nel caso IU abbia eseguito l'istruzione START_PROCESS alla fine della procedura di commutazione di contesto.)

L'indirizzo logico dell'istruzione (*IC1* oppure *newIC*) viene tradotto in indirizzo fisico, *se* l'entrata corrispondente della Tabella di Rilocazione è presente nella memoria associativa (*MA*) *e* la funzione di traduzione ha successo. La coppia (indirizzo fisico *INDFIS*, indirizzo logico *INDLOG*) è inviata a CI: l'indirizzo logico funge da identificatore unico dell'istruzione, in modo che IU ne possa verificare la validità.

Quando si verifica un fault di memoria associativa, l'entrata della Tabella di Rilocazione (il cui indirizzo base *IND_TABRIL* è noto alla MMU_I) viene richiesta (*come per ogni dato riferito dal processo*) *alla Cache Dati* (sottosistema DM), la quale eventualmente provocherà (*come per ogni dato riferito dal processo*) un trasferimento di blocco dai livelli superiori della gerarchia di memoria.

Quando si verifica un fault di pagina, l'elemento dello stream consiste nel messaggio di eccezione (ECC), segnalata alla IU via CI, insieme all'indirizzo logico.

Con comunicazioni a singola bufferizzazione la latenza di comunicazione intra-chip vale L_{com} = 2τ. Il tempo di servizio ideale è quindi:

$$T_{id-MMU_I} = \max(T_{calc-MMU_I}, L_{com}) = 2\,\tau$$

purché $T_{calc-MMU_I} \leq 2\,\tau$. *In realtà*, è importante progettare MMU_I in modo che:

$$\boldsymbol{T_{calc-MMU_I} = \tau}$$

per garantire il minimo tempo di risposta alle richieste di IU e massimizzare la probabilità che l'ampiezza media della bolla causata da salti sia uguale a $t = 2\tau$.

Per rispettare il vincolo suddetto, quando i dati sono pronti il microprogramma deve essere eseguito in un singolo ciclo di clock, tranne (per ipotesi) quando si verifica il fault di memoria associativa. A questo scopo, le funzioni *traduci(IC1.IPL).esito* e *traduci(newIC.IPL).esito* sono usate, in modo mutuamente esclusivo, come variabili di condizionamento complesse: questo non causa ritardi addizionali all'interno del ciclo di clock in quanto i risultati *traduci(IC1.IPL).ipf* e *traduci(newIC.IPL).ipf* delle stesse funzioni sono ovviamente anch'essi stabili all'inizio del ciclo di clock. Il microprogramma è:

0. (ACKCI, RDYIU, *traduci(IC1.IPL).esito*, *traduci(newIC.IPL).esito* = 0 - - - - -) nop, 0;

   // non presente messaggio da IU, traduzione eseguita con successo //

   (= 1 **0** **00** - -) *traduci(IC1.IPL).ipf* ° IC1.DISPL → INDFIS, IC1 → INDLOG, 0 → ECC, set RDYCI, reset ACKCI, IC1 + 1 → IC1, 0;

   // non presente messaggio da IU, fault di pagina //

   (= 1 **0** **01** - -) IC1 → INDLOG, 1 → ECC, set RDYCI, reset ACKCI, 0;

   // non presente messaggio da IU, fault di memoria associativa: comunicazione a domanda-risposta con la Cache Dati per ottenere l'entrata di TABRIL //

   (= 1 **0** **1 –** - -) IND_TABRIL + IC1.IPL → OUT_DM, set RDYOUT_DM, IC1 → SAVE_IC, 1;

   // presente messaggio da IU: azioni analoghe su *newIC* invece che su IC1 //

   (= 1 **1** - - **0 0**) reset RDYIU, set ACKIU, *traduci(newIC.IPL).ipf* ° newIC.DISPL → INDFIS, newIC → INDLOG, 0 → ECC, set RDYCI, reset ACKCI, newIC + 1 → IC1, 0;

   (= 1 **1** - - **0 1**) reset RDYIU, set ACKIU, newIC → INDLOG, 1 → ECC, set RDYCI, reset ACKCI, 0;

   (= 1 **1** - - **1 –**) reset RDYIU, set ACKIU, IND_TABRIL + newIC.IPL → OUT_DM, set RDYOUT_DM, newIC → SAVE_IC, 1

// le microistruzioni 1 e 2 seguenti sono eseguite quando si è verificato fault di MA; l'entrata ricevuta in IN_DM, concatenata con l'identificatore di pagina logica, viene scritta nella posizione *rimpiazza( )* di MA //

1. (RDYIN_DM = 0) nop, 1;

   (= 1) reset RDYIN_DM, SAVE_IC.IPL ° IN_DM → MA[*rimpiazza( )*], 2

2. (*traduci(SAVE_IC.IPL).esito* = **0 0**) *traduci(SAVE_IC.IPL).ipf* ° SAVE_IC.DISPL → INDFIS, SAVE_IC → INDLOG, 0 → ECC, set RDYCI, reset ACKCI, SAVE_IC + 1 → IC1, 0;

   (= **0 1**) SAVE_IC → INDLOG, 1 → ECC, set RDYCI, reset ACKCI, 0;

   (= 1 –) // impossibile // nop, 0

c) Una qualunque MMU potrebbe contenere una RAM per bufferizzare le entrate della Tabella di Rilocazione del processo ed accedervi per indice, ma questo non comporterebbe alcun vantaggio prestazionale, in quanto il numero di pagine logiche effettivamente riferite è minore (spesso molto minore) del numero di pagine logiche facenti parte della memoria virtuale del processo stesso. Se, addirittura, TABRIL fosse completamente copiata all'atto della commutazione di contesto, il numero di trasferimenti di entrate sarebbe (molto) maggiore rispetto a quello che si ha copiando le entrate su domanda.

D'altra parte, la memoria associativa, necessaria per effettuare la ricerca per contenuto, è accedibile all'interno del singolo ciclo di clock, allo stesso modo di quanto avviene con una RAM.

In conclusione, ad una maggiore area necessaria per una RAM (in D-RISC: capacità di 4M con pagine di 1K) corrisponderebbe solo un maggior tempo di servizio.

Domanda 3

Essendo operante su stream, il tempo di servizio effettivo dell'unità Cache Dati (CD) è valutato come:

$$T_{CD} = \max(T_{CD-id}, T_A)$$

dove T_A è il tempo medio di interarrivo delle richieste da IU (via MMU_D). T_A va determinato in funzione del tempo di servizio di IU, e quindi della CPU, per il determinato programma.

Il tempo medio di servizio ideale dell'unità CD è valutabile come:

$$T_{CD-id} = (1-p)\ 2\ \tau\ +\ p\ T_{trasf_{C2\_C1}}$$

dove p è la probabilità di effettuare un trasferimento blocco dalla cache secondaria. I tre array godono della sola proprietà di località, quindi l'insieme di lavoro è di 3 blocchi. Poiché l'array C è utilizzato in sola scrittura, si ha che su *3N* richieste ricevute da IU quelle che provocano trasferimento di blocco sono in numero di $2N/\sigma$. Quindi:

$$p = \frac{2}{3\ \sigma}$$

In accordo alle ipotesi sulla cache secondaria, si ha:

$$T_{trasf_{C2\_C1}} = 2\ \sigma\ \tau$$

da cui:

$$T_{CD-id} = 2\ \tau\ [1 + p\ (\sigma - 1)] \sim \frac{10}{3}\ \tau$$

valido per valori tipici di σ (ad esempio, per σ = 8 il risultato sarebbe 38 τ/12).

Valutiamo il tempo di completamento del programma. La compilazione ottimizzata per CPU in-order è:

```
LOOP:   LOAD   RA, Ri, Ra
        LOAD   RB, Ri, Rb
        INCR   Ri
        MUL    Ra, Rb, Ra
        IF <   Ri, RN, LOOP, delayed_branch
        STORE  RC, Ri, Ra
```

La dipendenza logica IU-EU indotta da INCR su IF (distanza 2, probabilità 1/6, N_Q = 2, L_{pipe} = 0) è intrecciata con quella indotta da MUL su STORE (distanza 2, probabilità 1/6, N_Q = 2, L_{pipe} = 4): la seconda ha latenza maggiore; inoltre non ci sono ulteriori ritardi indotti da dipendenze EU-EU nella sequenza critica. Quindi il ritardo dovuto a dipendenze logiche è dato da:

$$\Delta = \Delta_1 = t\, d_k \left(L_{pipe-k} + N_{Qk} + 1 - k\right) = \frac{5}{6}\, t$$

Per effetto del delayed branch (λ = 0), il tempo di servizio per istruzione è dato da:

$$T = t + \Delta = \frac{11}{6}\, t$$

Il tempo di completamento, incluso il ritardo dovuto ai fault di cache, vale:

$$T_c = T_{c-0} + T_{fault} = 6\, N\, T + 2\, \frac{N}{\sigma}\, 2\, \sigma\, \tau = 22\, N\, \tau + 4\, N\, \tau = 26\, N\, \tau$$

Questo risultato è valido in quanto si verifica che la memoria esterna non costituisce collo di bottiglia per le scritture con il metodo Write-Through. Infatti, la banda di richieste di scrittura (una per iterazione) è:

$$B_{richiesta} = \frac{1}{T_{iter}} = \frac{N}{T_c} = \frac{1}{26\, \tau}$$

che risulta minore della banda offerta dalla memoria interallacciata per scritture a indirizzi consecutivi:

$$B_{offerta} = B_{M-max} = \frac{m}{\tau_M} = \frac{1}{12{,}5\, \tau}$$

A questo punto, il tempo medio di interarrivo delle richieste di IU a DM (tre richieste per iterazione) è dato da:

$$T_A = \frac{T_{iter}}{3} = \frac{26}{3}\, \tau$$

In conclusione:

$$T_{CD} = \max(T_{CD-id}\, , T_A) = T_A = \frac{26}{3}\, \tau$$