Architettura degli Elaboratori, 2008-09

Prima prova di verifica intermedia

4 novembre 2008

Una unità di elaborazione U ha al suo interno due memorie, A e B, di capacità N = 16K parole, dove la parola è di 32 bit.

U riceve dall'unità U_0 valori OP di 1 bit, dall'unità U_1 coppie (IND, H), dove IND è un indirizzo delle memorie e H è un valore di 5 bit. U invia all'unità U_2 valori $Z0$ di 1 bit, e ad U_3 valori $Z1$ rappresentati su un numero di bit opportuno.

Le operazioni esterne sono definite come segue:

- *prima operazione esterna* ($OP = 0$): considerando le due locazioni di A e B indirizzate da IND, se esse hanno il bit H-esimo uguale: copiare la locazione di A nella locazione di B e inviare 1 a U_2; altrimenti non modificare le memorie e inviare 0 a U_2;
- *seconda operazione esterna* ($OP = 1$): considerando A e B suddivise logicamente in blocchi di 512 parole, per ogni locazione i ($i = 0 \ldots 511$) del blocco H-esimo di A e di B:

  $$\textit{se } A[i] > B[i] \textit{ eseguire } B[i] = A[i] + B[i]/4$$

  Inviare a U_3 il numero di modifiche apportate al blocco di B.

É noto il ritardo t_p di una porta logica con al massimo 8 ingressi. Una ALU ha ritardo uguale a $5t_p$. Le memorie A e B sono realizzate partendo da componenti logici memoria aventi capacità 256 parole e tempo di accesso $3t_p$.

1) Con il vincolo che la prima operazione esterna sia eseguita in un singolo ciclo di clock, scrivere il microprogramma di U e valutarne il tempo medio di elaborazione nell'ipotesi che la probabilità della prima operazione esterna sia uguale a 1/4 e quella della seconda uguale a 3/4.
   È necessario fornire opportune e sintetiche spiegazioni sulle scelte più significative effettuate.
2) Il sistema è modificato come segue: U_0 e U_3 sono eliminate, e U esegue solo la prima operazione esterna. Realizzare U come singola rete sequenziale esprimendone le funzioni che la definiscono, *spiegando sinteticamente come si è ragionato*.
3) Con riferimento al punto 1):
   a) esprimere la funzione di transizione dello stato interno della Parte Operativa relativamente alla generica locazione della memoria B, e la funzione di transizione dello stato interno della Parte Controllo;
   b) spiegare se, con gli stessi vincoli e nelle stesse ipotesi, lo stesso microprogramma potrebbe essere eseguito in un tempo minore rispetto a quello valutato.

Soluzione

(1)

Il vincolo sul tempo di elaborazione della prima operazione esterna impone la scelta di una variabile di condizionamento complessa:

exor(A[IND][H], B[IND][H])

implementata mediante due commutatori a 32 ingressi comandati da *H* (le uscite delle memorie sono viste come array di 32 bit), seguiti da un confrontatore su singolo bit.

La scelta di variabili di condizionamento complesse va quindi adottata anche per la seconda operazione esterna, nella quale useremo:

segno(A[J] – B[J]) *zero(A[J] – B[J])*

scrivendo il microprogramma in modo da minimizzare il numero di cicli di clock allo scopo di minimizzare il tempo medio di elaborazione. Sarebbe del tutto inutile cercare soluzioni per operare anche sul ciclo di clock, che è imposto dal vincolo sulla prima operazione esterna.

Nella seconda operazione esterna useremo un registro *J* per indirizzare le memorie, inizializzato con il concatenamento del valore di *H* e di una costante di nove zeri. Il risultato è ottenuto in un registro *C* di 10 bit. Il controllo del loop *for* è effettuato con un registro *I* di 10 bit, inizializzato a zero e incrementato ad ogni passo, con condizione di terminazione ($I_0 = 1$).

L'espressione *A[i] + B[i]/4* si valuta come *A[i] +* sh_{R-2}*(B[i])*, implementata mediante due ALU in cascata: questa soluzione minimizza il tempo medio di elaborazione rispetto a quella in cui il calcolo viene spezzato in due cicli di clock.

Il microprogramma è il seguente:

0. (RDY0, OP, RDY1, *exor*(A[IND][H], B[IND][H]), ACK2 = 0 – – – –, 1 – 0 – –, 1 0 1 – 0) nop. 0;

// prima operazione esterna //

(= 1 0 1 0 1) reset RDY0, set ACK0, reset RDY1, set ACK1, reset ACK2, set RDY2, 1 → Z0,
A[IND] → B[IND], 0;

(= 1 0 1 1 1) reset RDY0, set ACK0, reset RDY1, set ACK1, reset ACK2, set RDY2, 0 → Z0, 0;

// seconda operazione esterna: inizializzazione //

(= 1 1 1 – –) reset RDY0, set ACK0, reset RDY1, set ACK1, H ∘ nove_zeri → J, 0 → I, 0 → C, 1

// seconda operazione esterna: ciclo for //

1. (I_0, *segno*(A[J] – B[J]), *zero*(A[J] – B[J]), ACK3 = 0 0 0 –) A[J] + sh_{R-2}(B[J]) → B[J], C + 1 → C, I + 1 → I,
J + 1 → J, 1;

(= 0 0 1 –, 0 1 – –) I + 1 → I, J + 1 → J, 1;

(= 1 – – 0) nop, 1;

(= 1 – – 1) reset ACK3, set RDY3, C → Z1, 0

Per soddisfare la condizione necessaria per la correttezza della PO (modello di Moore), le variabili di condizionamento complesse *segno(...)* e *zero(...)* sono implementate da due ALU dedicate e le memorie *A* e *B* sono a doppio indirizzamento (J, IND). Per entrambi gli indirizzi, *A* è in sola lettura e *B* in lettura/scrittura. *B* ha quindi due variabili di controllo β distinte (β_{B1}, β_{B2}). *B* ha un commutatore sull'ingresso del dato (variabile di controllo α_B).

Il tempo di accesso delle memorie modulari, con 64 moduli, è $t_a = 6t_p$, in quanto i commutatori di uscita ai 64 moduli hanno un livello di logica AND e due livelli OR.

Le variabili di condizionamento *exor (...), segno (...), zero (...)* hanno tutte un ritardo

$$T_{\omega PO} = t_a + 5t_p = 11\ t_p$$

Infatti, il ritardo di stabilizzazione dei commutatori per ottenere $A[IND][H]$ e $B[IND][H]$ è $3t_p$ (un livello di logica AND e due livelli OR), cui va sommato il ritardo di un confrontatore su bit.

L'operazione di trasferimento tra registri con ritardo maggiore è $A[J] + sh_{R\text{-}2}(B[J]) \rightarrow B[J]$ che, tenendo conto della completa sovrapposizione della scrittura in B, ha un ritardo:

$$T_{\sigma PO} = t_a + 2\ T_{ALU} + T_K = 18\ t_p$$

Operazioni di incremento di registri hanno un ritardo $T_{ALU} + 2\ T_K$.

$T_{\omega PC} = 2t_p$, in quanto il massimo numero di variabili di condizionamento specificate in una condizione logica è 5 (un livello AND) e il massimo numero di frasi che operano su uno stesso registro è 3 (un livello OR).

Quindi:

$$\tau = 32t_p$$

La prima operazione esterna è eseguita in $T_0 = 1\tau$, la seconda in $T_1 = \sim 512\tau$. Con le probabilità date si ottiene un tempo medio di elaborazione:

$$T = (1/4)\ T_0 + (3/4)\ T_1 \sim (3/4)\ 512\ \tau = 12.228t_p$$

(2)

La fattibilità della realizzazione come singola rete sequenziale è dovuta al fatto che il microprogramma consta di una sola microistruzione, quindi PC è una rete combinatoria:

0. (RDY1, *exor*(A[IND][H], B[IND][H]), ACK2 = 0 – – , 1 – 0) nop. 0;
   (= 1 0 1) reset RDY1, set ACK1, reset ACK2, set RDY2, 1 → Z0, A[IND] → B[IND], 0;
   (= 1 1 1) reset RDY1, set ACK1, reset ACK2, set RDY2, 0 → Z0, 0;

Con il modello di Moore si ha:

ω ::

$$\begin{cases} out0 = Z0 \\ out\ 1 = ACK1 \\ out2 = RDY2 \end{cases}$$

σ ::

$$\begin{cases} \forall i = 0\ ..\ N\text{–}1 : in_{B[i]} = when\ (\beta = 1)\ and\ (i = IND)\ do\ A[IND] \\ in_{Z0} = when\ (\beta = 1)\ do\ \alpha \\ \text{per ogni indicatore di interfaccia: } in = \beta \end{cases}$$

dove:

$$\beta = RDY1\ ACK2$$

$$\alpha = RDY1\ \overline{exor(A[IND][H], B[IND][H])}\ ACK2$$

corrispondono alle espressioni logiche delle variabili di controllo valutate dalla ex-PC.

Il ciclo di clock vale:

$$\tau = T_\sigma + \delta = t_a + T_{K\text{-}32} + T_\oplus + \delta = 12\ t_p$$

(3)

a) Le funzioni richieste per l'unità del punto *1)* si esprimono come segue:

$\sigma_{PO/B[i]}$ ∴

$in_{B[i]}$ = *when* (β_{B1} = 1) *and* (i = IND) *do* *if not* α_B *then* A[IND] *else* A[IND] + $sh_{R\text{-}2}$(B[IND])

or when (β_{B2} = 1) *and* (i = J) *do* *if not* α_B *then* A[J] *else* A[J] + $sh_{R\text{-}2}$(B[J])

ricavata dallo schema parziale della PO per quanto riguarda la memoria B;

σ_{PC} ∴

$$Y = \bar{y}\ RDY0\ OP\ RDY1 + \overline{I_0} + I_0\ \overline{ACK3}$$

ricavata direttamente dal microprogramma, usato come tabella di verità delle funzioni di PC.

b) La frase condizionale nella quale viene calcolato *A[J] + $sh_{R\text{-}2}$(B[J])* → *B[J]* è caratterizzata contemporaneamente dal massimo valore di $T_{\omega PO}$ e dal massimo valore di $T_{\sigma PO}$.

Il ciclo di clock calcolato con la formula fornisce una valutazione esatta, in quanto coincidente con la valutazione che si otterrebbe considerando separatamente tutte le frasi (e quindi il tempo di elaborazione valutato non è riducibile), a condizione che, nell'applicare la formula, non si tenga conto del fatto che il ritardo della lettura dalle due memorie viene pagato una sola volta all'inizio del ciclo di clock.

Considerando questo aspetto, il ciclo di clock può essere ridotto di t_a e di conseguenza il tempo di elaborazione.