Versione semplificata della domanda 1 del compito del primo appello A.A. 2018-19 (5/1/19)

L’unità firmware B riceve da una unità A interi da 32 bit e ne ricerca la prima occorrenza in una memoria interna M da 8K parole. La memoria M è ordinata, ovvero per ogni coppia di indirizzi i e j vale che se i<j allora sicuramente M[i]<=M[j]. A interagisce con B secondo un protocollo a domanda-risposta. B restituisce ad A un booleano che dice se il numero è stato trovato all’interno della memoria e un intero che indica l’indirizzo dell’intero cercato, in caso sia presente. B usa un algoritmo di ricerca binaria.

Si implementi l’unità B e se ne fornisca il ciclo di clock con le seguenti ipotesi:

- Sono a disposizione ALU che fanno operazioni fra interi in 5tp
- Il tempo di accesso alla memoria è ta = 10tp
- Le porte logiche hanno al massimo 4 ingressi
- Tutte le parole della memoria contengono dati validi (la memoria è piena).

Traccia di soluzione
Il microcodice dell'unità può essere scritto come segue (OUT e T sono i registri utilizzati per comunicare la risposta all'unità A: OUT contiene l'indice dell'elemento e T=1 se è stato trovato, =0 altrimenti).

Microprogramma NON ottimizzato

1. (RDY=0) nop, 1
   (=1) 0 → START, 8K-1 → END, 2
2. (segno(END-START)=1) 0 → T, reset RDY, set ACK, 1
   (=0) START +END → J, 3
3. SHR(J,1) → J, 4
4. (EQ(DATAIN,M[J])=0) J → OUT, 1 → T, reset RDY, set ACK, 1
   (=1) NOP, 5
5. (segno(M[J]-DATAIN)=0) J+1 → START, 2
   (=1) J-1 → END, 2

Microprogramma OTTIMIZZATO (commentato)

0. (RDY=0) nop, 0
// aspetta il dato da cercare
(=1) 0→ START, 8K-1→ END, 1
// dato arrivato: inizializzazioni (8K-1=8193)
1. (segno(END-START)=1) 0→T, reset RDY, set ACK, 0
// end<start: dato non trovato
(=0) SHR(START +END)→ J, 2
// start<=end: calcolo nuovo indice in cui cercare
2. (EQ(DATAIN,M[J]),(segno(M[J]-DATAIN)=0 -) J→OUT, 1→T,
reset RDY, set ACK, 0
// dato trovato: restituzione indice e T
(=1 0) J+1→START, 1
// dato non trovato: ricerca nella parte con indici più grandi
(=1 1) J-1→END, 1
// dato non trovato: ricerca nella parte con indici più piccoli

** perché non J→ START oppure J→ END ??*

EQ(X,Y) è una rete che utilizza 32 comparatori in parallelo per confrontare i bit corrispondenti di X e Y e calcola l'OR delle loro uscite in 5 tp (2 per i comparatori e 3 per l'OR), oppure è realizzata con una ALU (dedicata) di cui prendiamo zero(X-Y), sempre in 5tp, viste le specifiche del problema.
Abbiamo 3 stati interni (che richiedono 2 bit) e 4 variabili di condizionamento (di cui al massimo 2 sono testate contemporaneamente). Questo implica che per la ωPC e la σPC ci sia un solo livello di porte AND (al più 4 valori da considerare per ogni colonna in cui c'è un "1" (o uno "0", vedi dopo). Abbiamo in tutto 7 frasi. Questo implica che per la ωPC e la σPC ci sia un livello di porte OR (abbiamo al massimo 7 bit ad "1" in ogni colonna della tabella di verità: se sono più di 4, possiamo codificare gli "0" e complementare (negare) l'uscita). Abbiamo quindi TωPC = TσPC = 2tp. TωPO è 0 nella 0., talu nella 1., e talu+ta nella 2. TσPO è t_k nella 0., talu+tk nella 1. (l'operazione di shift destra di una posizione è realizzata collegando i 31 bit più significativi completati a sinistra con uno 0 all'ingresso del commutatore davanti a J) e talu+tk nella 2. (talu e tk sono i tempi di stabilizzazione della ALU e del commutatore, ta è il tempo di accesso alla memoria). La microistruzione più lenta è la 2. Il ciclo di clock è dunque $\tau = 15tp+(7tp+2tp)+\delta = 25tp$.