Si consideri una unità U che realizza una coda FIFO implementata mediante un vettore circolare in una memoria interna ad U, M, di 1024 parole di 32 bit, che permette la lettura di una parola e la scrittura della stessa o di un' altra parola nello stesso ciclo di clock. U è collegata ad U1, U2 ed U3 mediante collegamenti dedicati. Il collegamento con U1 e quello con U2 è in ingresso, l'interfaccia è dotata di indicatori a transizione di livello e permette la trasmissione di una singola parola da 32 bit. U3 interagisce con U secondo un protocollo a domanda risposta: invia richieste (senza invio di dati) ad U, e riceve come risposta la parola in cima alla coda.

U attende una parola da U1 ed una da U2 e memorizza in M, secondo la disciplina FIFO (cioè in fondo alla coda), la media dei due valori (arrotondata per difetto).
Quando U riceve una richiesta da U3, trasmette una parola di M, prelevata sempre utilizzando una disciplina FIFO.

Se entro TO cicli di clock (con TO costante di sistema, non più grande del massimo valore memorizzabile in una parola di M) dalla sua ultima richiesta, U3 non effettua nuove richieste, l'elemento in cima alla coda viene comunque eliminato.

Si progetti U **e** se ne calcoli la lunghezza del ciclo di clock, in termini di tp, sapendo che una ALU ha un ritardo di 5tp, che il tempo di accesso alla memoria è dominato dal tempo di lettura (che quindi va calcolato in termini di tp), che la tecnologia hardware utilizzata permette la scrittura e la lettura nello stesso registro nel medesimo ciclo di clock (come avviene per la memoria), e che le porte logiche utilizzate per la costruzione di *tutti* i componenti di U hanno al massimo **6** ingressi.

Algoritmo e variabili usate

Come richiesto, organizziamo la coda FIFO come un vettore gestito in maniera circolare. Il puntatore INS indica la posizione di inserimento. Il puntatore ESTR, indica la posizione di estrazione. Un contatore NE indica il numero di elementi presenti nella coda. TIMER è il registro utilizzato per fare il countdown da TO a 0, ed è di 32 bit. INS, ESTR ed NE sono inizializzati a 0. INS ed ESTR sono da 10 bit e tutte le operazioni che li interessano saranno a 10 bit, dunque realizzando operazioni il cui risultato è automaticamente modulo 1024. NE è da 11 bit. NE0 (bit più significativo) sarà l'indicatore di coda piena (1=piena), OR(NEm) (10 bit meno significativi) sarà l'indicatore di coda vuota (0=vuota). OR(TIMER)=0 => TIMER è 0.

Le operazioni relative all'inserimento e all'estrazione sono:

- per l'inserimento di una parola X: X → M[INS], (INS+1)%1024 → INS, (NE+1) → NE, oltre agli eventuali set e reset sugli indicatori a transizione di livello necessari ad implementare il corretto protocollo di comunicazione. L'inserzione potrà avvenire se e solo se NE0 = 0.
- per l'estrazione di una parola Y: M[ESTR] → Y, (ESTR+1)%1024 → ESTR, (NE-1) → NE, oltre agli eventuali set e reset sugli indicatori a transizione di livello necessari ad implementare il corretto protocollo di comunicazione. L'estrazione potrà avvenire se e solo se OR(NEm) = 1.

I casi possibili sono:

- se si riceve solo una richiesta di estrazione con coda non vuota, si esegue l'estrazione e si ritorna allo stato iniziale
- se si ricevono una parola da U1 ed una parola da U2 e la coda non è piena, si inserisce la media delle due parole e si ritorna nello stato iniziale
- se si ricevono una parola da U1 ed una parola da U2 e una richiesta da U3 distinguiamo tre casi:
  - la coda è vuota: la media delle due parole ricevute da U1 ed U2 viene passata ad U3 e la coda rimane vuota, si ritorna allo stato iniziale
  - la coda contiene almeno un elemento ma non è piena: si inserisce la media delle due parole ricevute da U1 ed U2 e si passa il più vecchio elemento in coda ad U3, aggiornando di conseguenza INS, ESTR ma non NE, si ritorna allo stato iniziale
  - la coda è piena: si passa l'elemento più vecchio in coda ad U3 e si sostituisce con la media delle due parole ricevute da U1 ed U2, aggiornando INS ed ESTR, ma non NE, si ritorna allo stato iniziale.

• in tutti gli altri casi, si rimane nello stesso stato senza fare niente. In ogni caso, si decrementa di 1 il valore di timer e, se questo diviene 0 senza che ci sia una richiesta di U3, si estrae l'elemento in cima alla coda.
Le condizioni da testare (a parte quella di TIMER, e la sua gestione) sono dunque (assumiamo che la coppia di indicatori a transizione di livello Ri Ai sia associata all'interfaccia con Ui e che Ri sia l'indicatore in ingresso ed Ai quello in uscita):
• R1,R2,R3,NE0,OR(NEm)=001-1,011-1,101-1 → estrazione
• R1,R2,R3,NE0,OR(NEm)=1100- → inserzione (e, se TIMER=0, estrazione)
• R1,R2,R3,NE0,OR(NEm)=11100 → passaggio della media dei valori ricevuti ad U3
• R1,R2,R3,NE0,OR(NEm)=11101 → inserzione+estrazione
• R1,R2,R3,NE0,OR(NEm)=1111- → inserzione + estrazione (stessa posizione nella memoria M, possibile per ipotesi)

Sotto queste ipotesi, il **microprogramma** potrebbe essere il seguente:

0. *// sola richiesta di estrazione, coda non vuota*
*(R1,R2,R3,N0,or(N),**OR(TIMER)**=001-1**1**,011-1**1**,101-1**1**)*
M[ESTR]→OUT, (ESTR+1)%1024→ ESTR, (NE-1)→NE,TO→ TIMER, set A3, reset R3, 0
*(R1,R2,R3,N0,or(N),**OR(TIMER)**=001-1**0**,011-1**0**,101-1**0**)*
M[ESTR]→OUT, (ESTR+1)%1024→ ESTR, (NE-1)→NE,TO→ TIMER, set A3, reset R3, 0
// sola richiesta di inserzione, coda non piena
(R1,R2,R3,N0,OR(N),OR(TIMER)=1100-1) SHR(X1+X2)→M[INS], (INS+1)%1024→INS, (NE+1)→NE, TIMER-1→ TIMER, set A1, set A2, reset R1, reset R2, 0
(R1,R2,R3,N0,or(N)OR(TIMER)=1100-0) SHR(X1+X2)→M[INS], (INS+1)%1024→INS, TO→ TIMER, (ESTR+1)%1024→ ESTR,set A1, set A2, reset R1, reset R2, 0
// inserzione ed estrazione con coda vuota
(R1,R2,R3,NE0,OR(NEm),OR(TIMER)=111001) SHR(X1+X2)→ OUT, TO→ TIMER, set A1, set A2, set A3, reset R1, reset R2, reset R3, 0
(R1,R2,R3,NE0,OR(NEm),OR(TIMER)=111000) SHR(X1+X2)→ OUT, TO→ TIMER, set A1, set A2, set A3, reset R1, reset R2, reset R3, 0
// inserzione ed estrazione da coda non vuota e non piena e lo stesso per inserzione ed estrazione da coda piena

(R1,R2,R3,NE0,OR(NEm),OR(TIMER)=111011,1111-1)
SHR(X1+X2) → M[INS], (INS+1)%1024 → INS, TO → TIMER, M[ESTR]→OUT, (ESTR+1)%1024→ESTR, set A1, set A2, set A3, reset R1, reset R2, reset R3, 0
(R1,R2,R3,NE0,OR(NEm),OR(TIMER)=111010,1111-0)
SHR(X1+X2) → M[INS], (INS+1)%1024 → INS, TO → TIMER, M[ESTR]→OUT, (ESTR+1)%1024→ESTR, set A1, set A2, set A3, reset R1, reset R2, reset R3, 0
// attesa se non ho richieste o non ho richieste di inserimento complete (da entrambe U1 e U2)
(R1,R2,R3,NE0,OR(NEm),OR(TIMER)=000--1, 010--1, 100--1) TIMER-1 → TIMER, 0
(R1,R2,R3,NE0,OR(NEm),OR(TIMER)=000--0, 010--0, 100--0) TO → TIMER, (ESTR+1)%1024 → ESTR, 0

Essendoci una sola microistruzione (un solo stato nell'automa di controllo) la PC sparisce come rete sequenziale. Rimane solo la ωPC come rete combinatoria. Il ritardo introdotto dalla ωPC (rete con 5 ingressi e uscite distinte per al massimo 5 frasi escluso il timer, 10 frasi per il timer) sarà di 2tp.
Il ciclo di clock viene dunque calcolato come

$$\tau = T\omega PO + T\omega PC + T\sigma PO + \delta$$

Le variabili di condizionamento richiedono al massimo 2tp (calcolo di OR(NEm) con NEm da 10 bit, calcolo di OR(TIMER) con TIMER di 32 bit: servono due livelli di porte OR da 6 bit).
Le risorse di stato della PO sono:
- INS, scritto sempre e solo con il risultato di una ALU che calcola l'incremento modulo 1024
- ESTR, idem
- NE, scritto sempre e solo con una ALU che calcola incremento o decremento

TIMER, scritto sempre e solo con una ALU che calcola solo decremento, oppure con TO. Quindi, è necessario un commutatore in ingresso al timer.
- M, scritta sempre all'indirizzo INS e letta dall'indirizzo ESTR, il valore in ingresso è sempre (IN1+IN2)/2 calcolato come descritto nel seguito. Nel caso estrazione+inserzione è evidente la necessità di avere una memoria a doppia porta. Nel caso di lettura e scrittura che avvengono nella stessa posizione di memoria, la cosa è possibile utilizzando una tecnologia che

effettua la lettura sul fronte di discesa del clock (inizio del ciclo) e la scrittura avviene al prossimo clock alto (fine del ciclo): questo è possibile solo se le due operazioni avvengono in modo estremamente veloce
- gli indicatori a transizione di livello, su cui si opera esclusivamente con le operazioni di set e reset.

Dunque non vi sono commutatori in ingresso alle risorse di stato, eccettuato quello per il timer.
Le risorse di calcolo nella PO sono:
- una ALU per incremento INS e ESTR, con operazioni a 10 bit (modulo 1024). Se si usa una sola ALU, occorre un commutatore in ingresso per scegliere fra INS ed ESTR. L'uscita è comunque ingresso sia di INS che di ESTR (scritture comandate da due β distinti, ad 1 in frasi diverse). Il ritardo introdotto con una ALU sola è 2tp + 5tp = 7tp e con due ALU è 5tp.
- una ALU in grado di calcolare incremento o decremento con ingresso ed uscita da/su NE. Il ritardo introdotto è 5 tp.
- una ALU in grado di calcolare decremento con ingresso ed uscita da/su TIMER. Il ritardo introdotto è 5 tp.
- Il calcolo della media richiede una ALU e uno shift destro di una posizione, realizzabile scartando il bit meno significativo dell'uscita della ALU ed inserendo uno 0 come bit più significativo (questo a costo 0). Il ritardo introdotto è 5pt.

Tempi
Occorre calcolare il tempo di accesso alla memoria M per poter valutare TσPO. La memoria ha 1K posizioni. Dunque il commutatore di lettura (tempo più lento per M, da ipotesi) deve lavorare su 1K + 10 ingressi. La tabella di verità avrà 1024 righe e su ogni riga (corrispondente ad un 1 nella colonna delle uscite) avrà 11 bit specificati. Avendo a disposizione porte da 6 ingressi serviranno 2 livelli AND e $\lceil \log_6(1024) \rceil$ livelli di porte OR. Considerando che $6^3=216$, $6^4=1296$, serviranno 4 livelli di porte OR e dunque ta = 6tp.
Con questi risultati, TσPO sarà al massimo talu + ta (tempo della microoperazione più lenta, quella necessaria per calcolare (IN1+IN2)/2 → M[INS]; il calcolo di TIMER-1→ TIMER richiede talu+tk<talu+ta), ovvero 5tp + 6tp = 11tp.
Dunque il ciclo di clock sarà pari a:

$$\tau = T\omega PO + T\omega PC + T\sigma PO + \delta = 2tp + 2tp + 11tp + tp = 16tp$$