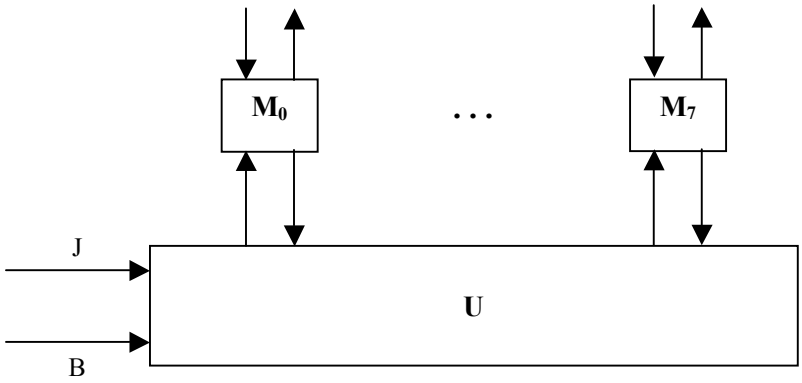
Architettura degli Elaboratori - Correzione

Appello del 27 giugno 2006, a.a. 2005/06

Domanda 1

Un sistema di elaborazione contiene le seguenti unità:



La memoria M, di capacità complessiva 1M parole (parole di 32 bit), ha organizzazione modulare con 8 moduli, M_0, …, M_7, e distribuzione degli indirizzi tra i moduli di tipo interallacciato. J è un indirizzo di tale memoria. B è di 2 bit.

U ha il seguente funzionamento: ricevuto un messaggio (J, B), se il byte B-esimo della parola di M avente indirizzo J è uguale a zero, allora scrive zero in tutti gli altri byte di tale parola.

a) Scrivere e spiegare il microprogramma di M.

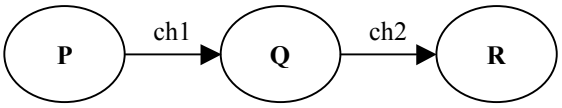
b) Mostrare e spiegare la Parte Operativa di U.

c) Ricavare l'espressione booleana della variabile di uscita della Parte Controllo di U che abilita la scrittura in uno dei registri di interfaccia di uscita (a scelta).

d) Ricavare il ciclo di clock di U in funziona del ritardo t_p di una porta logica con al più 8 ingressi.

Domanda 2

Si consideri il seguente programma concorrente:



Il canale *ch1* è sincrono; i messaggi trasmessi su *ch1* sono array unidimensionali di M = 10K interi. Il canale *ch2* è asincrono con grado di asincronia uguale a uno; i messaggi trasmessi su *ch2* sono interi.

Il processo Q ha il seguente funzionamento: ricevuto un messaggio A da *ch1*, invia su *ch2* il valore $x = \sum_{i=0}^{M-1} A[i]$.

a) Scrivere il processo Q nel linguaggio concorrente delle Note sul Livello dei Processi.

b) Compilare Q in assembler Risc, realizzando la funzione che calcola *x* come procedura.

c) Mostrare in dettaglio, e spiegare, lo spazio di indirizzamento e la memoria virtuale di Q. La spiegazione deve specificare qual è l'indirizzo base di ogni oggetto facente parte dello spazio di indirizzamento e quanti indirizzi occupa, inclusi i descrittori di processo.

d) Spiegare in seguito a quali eventi il processo Q può passare in stato di attesa e, in tali casi, a quale indirizzo si sospende.

Traccia di soluzione

Domanda 1

Si noti che l'Unità U è collegata con M_0 .. M_7 mediante 8 interfacce indipendenti, come descritto nella schema (e come esplicitamente suggerito durante i commenti al testo in fase di esame). Si noti che i moduli M_0 .. M_7 costituiscono logicamente una memoria M (unico spazio di indirizzamento), ma questo non permette di assumere che M sia acceduta mediante una sola interfaccia.

a) Notazione: dato un registro R

$$R_{disp} = R[0..16],\ R_{mod} = R[17..19]$$

0. $(RDYIN = 0)\ nop, 0;$
   $(=1)\ reset\ RDYIN, set\ ACKIN, J \rightarrow A, B \rightarrow C, J_{disp} \rightarrow IND[J_{mod}]\ 'read' \rightarrow OP[J_{mod}]\ set\ RDYOUT[J_{mod}]1$
1. $(RDY[A_{mod}]\ or\ (DATAIN[A_{mod}][C]) = 0-)\ nop, 1;$
   $(=10)\ resetRDY[A_{mod}]\ A_{disp} \rightarrow IND[A_{mod}]\ 'write' \rightarrow OP[A_{mod}]0 \rightarrow DATAOUT[A_{mod}]\ set\ RDYOUT[A_{mod}]2$
   $(=11)\ resetRDY[A_{mod}]0$
2. $(RDYIN[A_{mod}] = 0)\ nop, 2;$
   $(=1)\ reset\ RDYIN[A_{mod}]0$

b)

PO con commutatori e selezionatori sulle interfacce con M (commutatori dagli array di registri RDY e DATAIN, selezionatori verso gli array di registri IND, RDYOUT, OP, DATAOUT).

La condizione logica nella 2 ottimizza le prestazioni. Realizzazione nella PO: commutatore dai byte di DATAIN[A_{mod}] comandato da C.

c)

Consideriamo l'array di registri IND. C'è un unico β di scrittura che viene selezionato in funzione di J_{mod} o di A_{mod}:

$$\beta_{BIND} = \beta_{IND} = \overline{y_0}\,\overline{y_1}\,RDYIN + \overline{y_0}\,y_1\,RDY[A_{mod}]\overline{or(...)}$$

d)

Ciclo di clock:

$T_{\omega PO}$:

Il commutatore dai RDY ha ritardo 2tp.

Il commutatore dai DATIN ha ritardo 2tp. I quattro byte vengono commutati in funzione di C con un ritardo 2tp. La porta OR finale ha ritardo tp.

$$T_{\omega PO} = 5tp$$

$T_{\sigma PO}$:

C'è solo il selezionatore sui β dei registri di uscita, le cui variabili di controllo (J_{mod} o di A_{mod}) sono uscite di un commutatore: $T_{\sigma PO} = 4tp$.

Le funzioni di PC hanno ritardo 2tp.

$$\tau = 5tp + 2tp + 4tp + tp = 12\ tp$$

Domanda 2

Il testo richiede eplicitamente che la somma $x = \sum_{i=0}^{M-1} A[i]$ sia implementata mediante una procedura (punto b). Non si deve dimenticare che il risultato della compilazione dipende dal sorgente in Lc (punto a), che quindi dovrà necessariamente specificare la sommatoria con una procedura (in caso contrario, il codice al punto b non sarebbe la compilazione del codice al punto a).

```
const int M = 10K; <dichiarazioni canali ed altre dichiarazioni>
int riduci (int * arr) {
        int i, sum = 0;
        for (i=0; i<M; i++)
                sum+=arr[i];
        return (sum);
}

while (true} {
        receive(ch1,A);
        s = F(A);
        send(ch2,s);
}
<trattamento_interruzioni> <trattamento_eccezioni>
```

b) Si tratta di specificare in quail registri sono contenuti gli indirizzi delle procedure (riduci, send, receive) e in quail registri ogni procedura utilizza per i parametri di ingresso e di uscita.

c) Si tratta di determinare dimensioni realistiche per l'occupazione di codice, dati, e strutture di supporto a Lc quali PCB (di tutti i processi), tabelle dei canali, etc.

d) Q può andare in stato di attesa sia sulla receive che sulla send. Q si sopenderà all'indirizzo successivo della primitiva che ne ha determinato la sospensione (nel caso di receive si assume una implementazione alla pari, e quindi il lavoro di copia sarà effettuato da P, nel caso di send asincrona il lavoro di copia sarà effeuato da R). In entrambi i casi, Q al momento della sospensione ha completato il suo lavoro e deve solo aspettare che il partner faccia la sua parte.