Esercitazione 3 – Architettura dei Sistemi di Elaborazione – 09/01/2017
Soluzioni

Calcolatori e processi

1. **In un computer con 1 GB di memoria, il sistema operativo occupa 512 MB e i processi occupano mediamente 64 MB, se l'attesa media dell'I/O è del 60%, qual è l'utilizzo della CPU? Aggiungendo 256 MB di RAM, quale sarà il nuovo utilizzo della CPU?**

Stiamo parlando, in questo caso, di sistemi per la gestione della memoria e, in particolare, di come la dimensione della memoria RAM influenzi l'utilizzo della CPU. Il tempo di attesa del 60% è ovviamente riferito all'attesa di completamento di operazioni di I/O. Se abbiamo n processi in memoria e il tempo di attesa medio è p, allora la probabilità che tutti i processi siano in attesa (e quindi la CPU sia intattiva) è p^n. L'utilizzo della CPU è, dunque, $1- p^n$.
Ora sta tutto a stabilire quanti processi possono trovarsi contemporaneamente in memoria e quanto valga p. Ovviamente p vale 0.6, mentre il numero n di processi è ottenibile dividendo la dimensione di RAM libera per lo spazio occupato naturalmente da un solo processo. La quantità di RAM libera è, in modo triviale, 1GB – 512 MB = 512 MB. Il valore n, dunque, è 512/64 = 8.
Ora dobbiamo solo sostituire i valori, quindi:

$$\text{Utilizzo CPU} = 1 - 0.6^8 = 0.983$$

In caso si aggiunga altra RAM, dobbiamo solo comprendere come si modifica il valore n. È facile intuire che la nuova RAM libera sarà di 768MB, che divisi per 64MB ci offriranno la nuova n, 12.
Il nuovo utilizzo sarà:

$$\text{Utilizzo CPU} = 1 - 0.6^{12} = 0.997.$$

[Riferimento sul libro: Moderni sistemi operativi, paragrafo 4.1.3 *Modello della multiprogrammazione*, pag 176 sull'edizione in mio possesso]

2. **Supponendo di utilizzare l'algoritmo first-come first-served per lo scheduling in un sistema batch, se arrivano 4 job (in ordine A, poi B dopo 2ms, C dopo 4ms e D dopo 5ms) con i seguenti tempi di esecuzione A=8, B=4, C=2 e D=4 quali sono i tempi di turnaround? Qual è il tempo medio di turnaround e di attesa?**

Ora siamo passati nell'ambito dei processi e, in particolar modo, nell'ambito della schedulazione. Questo esercizio è davvero molto semplice: basta conoscere la definizione di *tempo di turnaround* e di *tempo di attesa*. Il *tempo di turnaround* è il tempo che intercorre dal momento in cui un processo entra in coda per essere eseguito fino al termine della sua esecuzione. Il *tempo di attesa*, invece, è il tempo che un processo trascorre in coda, prima di essere eseguito. Ora che abbiamo chiarito la teoria necessaria per svolgere l'esercizio, andiamo ad analizzare i nostri dati. Nella tabella che segue, TA rappresenta il momento in cui i processi entrano in cosa, mentre TE rappresenta il tempo necessario per eseguire il processo.

Processo	TA	TE
A	0	8
B	2	4
C	4	2
D	5	4

Quindi, possiamo assumere che, seguendo la strategia FCFS (first come- first served) o FIFO, se

preferite, i processi saranno eseguiti nell'ordine: A-B-C-D. A inizierà la sua esecuzione al tempo 0, B al termine dell'esecuzione di A (quindi al tempo 8), C al termine di B e così via. Per comodità, vi fornisco una seconda tabella con il TS, ovverosia il tempo assoluto in cui il processo inizia la propria esecuzione. Ovviamente, il TS di B sarà il tempo in cui A termina la propria esecuzione, e lo stesso varrà per ogni coppia di processi successivi. Allo stesso modo, TF sarà il tempo in cui ogni processo termina.

Processo	**TA**	**TE**	**TS**	**TF**
A	0	8	0	8
B	2	4	8	12
C	4	2	12	14
D	5	4	14	18

Ora è facile vedere che, per ogni processo, il tempo di turnaround (TT) è TF-TA e, allo stesso modo, TW (il tempo d'attesa) è pari a TS-TA. I risultati dunque, saranno:

Processo	**TA**	**TE**	**TS**	**TF**	**TW**	**TT**
A	0	8	0	8	0	8
B	2	4	8	12	6	10
C	4	2	12	14	8	10
D	5	4	14	18	9	13

Abbiamo risposto, dunque, alla prima domanda. Per rispondere alla seconda domanda, basta applicare la media aritmetica (ovverosia sommare i valori e poi dividere per il numero di processi) sulle colonne che sono state prodotte in questo ultimo passaggio. I valori voluti, dunque, saranno:

$$TW(medio) = \frac{0+6+8+9}{4} = 5.75$$
$$TT(medio) = \frac{(8+10+10+13)}{4} = 10.25$$

3. **Supponendo di dover valutare la fattibilità di un sistema soft real-time con eventi periodici P0=300µs, P1=900µs, P2=45ms, P3=150ms e rispettivi tempi di elaborazione C0=25µs, C1=300µs, C2=9ms, C3=50µs il sistema è sostenibile? Se si aggiunge un nuovo evento periodico P4=110ms, quanto è il tempo massimo di elaborazione affinché il sistema rimanga sostenibile?**

Stiamo ora valutando un sistema real-time, di cui potete trovare tutta la teoria che vi serve sempre sul libro *I moderni sistemi Operativi*. Questo tipo di sistemi è contraddistinto dal fatto che le operazioni devono tassativamente essere completate entro un certo tempo (denominato tipicamente *deadline*). Attenzione, questo non significa che questi sistemi debbano essere obbligatoriamente *veloci*, quanto piuttosto che siano *affidabili*. I sistemi real-time, infatti, sono generalmente utilizzati in sistemi con alta criticità. Al di là di questa semplice introduzione, esistono degli eventi detti *periodici*, che vanno eseguiti a ciclo continuo e con una cadenza ben stabilita. Nel nostro caso conosciamo, per ogni evento periodico, il suo periodo (ossia, ogni quanto deve essere ripetuto) e il suo tempo d'esecuzione. Ora, perché un sistema real-time sia sostenibile, sappiamo che, se ci sono

m processi periodici con periodo Pi e tempo d'esecuzione Ci, deve valere la seguente disuguaglianza:

$$\sum_{i=1}^{m}(\frac{C_i}{P_i})\leq 1$$

Quindi, ora dobbiamo solo inserire i dati nella diseguaglianza e chiederci se è verificata. Lo facciamo in questo semplice passaggio:

$$\frac{C_0}{P_0}+\frac{C_1}{P_1}+\frac{C_2}{P_2}+\frac{C_3}{P_3}=\frac{25}{300}+\frac{300}{900}+\frac{9}{45}+\frac{50}{150}=\frac{57}{60}\leq 1$$

Quindi il sistema è sostenibile.
Ora, per far sì che il sistema sia ancora sostenibile, dopo aver aggiunto il processo periodico P4, dobbiamo fare in modo che la disuguaglianza sia ancora verificata. Quindi:

$$\frac{C_0}{P_0}+\frac{C_1}{P_1}+\frac{C_2}{P_2}+\frac{C_3}{P_3}+\frac{C_4}{P_4}=\frac{57}{60}+\frac{C_4}{110}\leq 1$$

$$\frac{C_4}{110}\leq 1-57/60$$

$$C_4\leq(\frac{3}{60})*110\Rightarrow C_4\leq 5.5$$

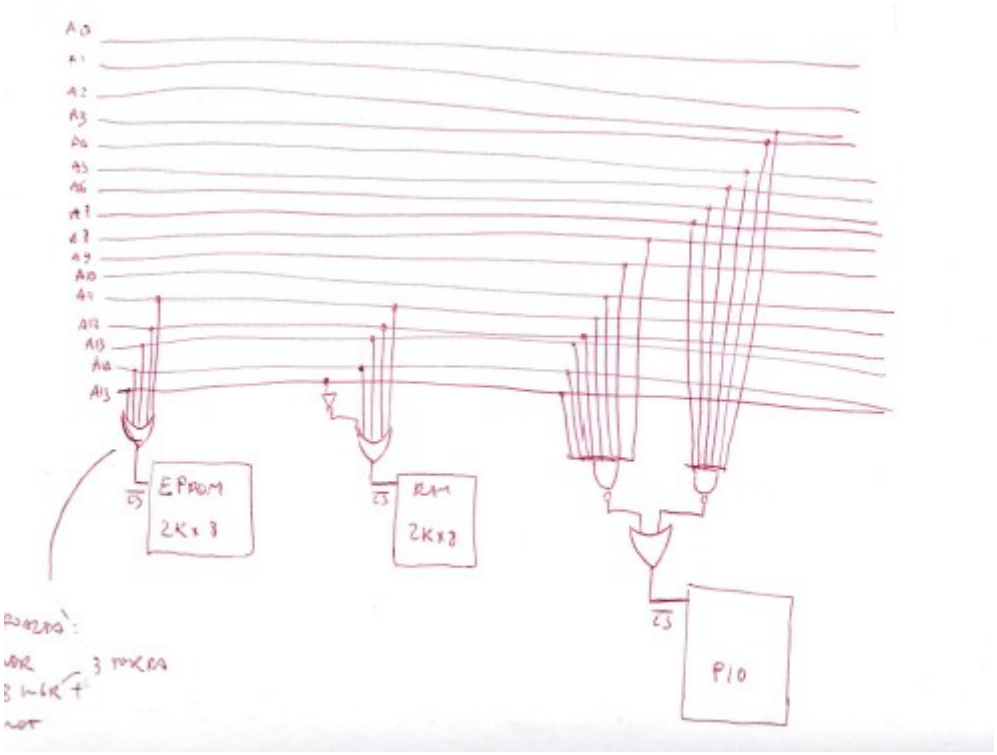
Quindi il massimo tempo d'elaborazione per il nuovo evento periodico sarà pari a 5.5ms.
Attenzione alle unità di misura: non fate confusione e cercate di scriverle sempre.

[Riferimento sul libro: Moderni sistemi operativi, paragrafo 2.5.4. *Schedulazione nei sistemi real time*, pag 135 e successive sull'edizione in mio possesso]

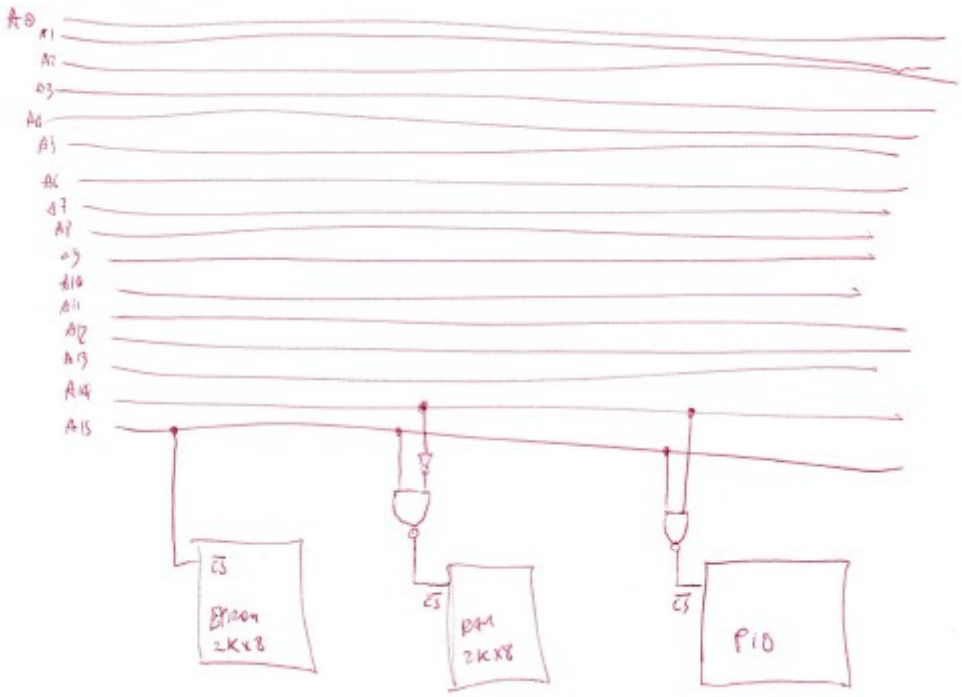
4. **Si supponga di avere un calcolatore monoprocessore con 16-bit di indirizzamento (A0÷A15), una EPROM di 2 KB × 8 byte per il programma, una RAM di 2 KB × 8 byte per i dati, una PIO tipo Intel 8255A con 24 porte e un registro di controllo.**
   Descrivere il circuito che abilita il chip di I/O in modalità Memory-Mapped I/O, se la PIO è posizionata a partire dall'indirizzo FFFCH della memoria.
   Suggerimento: porre la EPROM all'indirizzo 0 dello spazio di indirizzamento e la RAM all'indirizzo 8000H.

Questo esercizio è estremamente lungo, se lo si volesse spiegare per filo e per segno ci vorrebbe davvero molto tempo. Vi consiglio di studiare bene tutta la parte teorica, in modo tale da poter comprendere meglio quello che sto per scrivere. Sul libro (questa volta *Architettura dei Calcolatori* di Tanenbaum) lo trovate, praticamente identico, alle pagine 215 e seguenti.
Partiamo con l'idea che, se il calcolatore ha 16 bit di indirizzamento, allora è lecito pensare che il bus che lo collega agli altri componenti abbia lo stesso numero di bit di dimensione, che chiameremo allo stesso modo (A0÷A15). Ora, conoscendo la teoria, sappiamo che ls EPROM e la RAM descritte nell'esercizio richiedono uno spazio degli indirizzi di 2KB, mentre per la PIO (in modalità Memory-Mapped) richiede solo 4 byte. Quindi, ora sappiamo che tutti gli indirizzi del tipo 00000xxxxxxxxxxx sono destinati alla EPROM, quelli del tipo 10000xxxxxxxxxxx alla RAM e quelli del tipo 1111111111111xx alla PIO. Questo semplicemente trasformando l'indirizzo iniziale di ogni componente in binario. Ora si deve solo collegare le componenti al bus. Ricordiamo, che per ogni componente va attivata (con valore 0) la ports di controllo $\overline{CS}$. Basta, dunque collegare $\overline{CS}$ tramite una porta OR alle linee del bus corrisondenti a A_{11}, A_{12}, A_{13}, A_{14} e A_{15}. Discorso uguale si può fare per la RAM, negando, però, il segnale relativo alla linea A_{15}, in modo da avere quell'1 all'inizio dell'indirizzo. Per la PIO, infine, vanno lasciate scollegate solo le linee A_0 e A_1, in modo da

avere quella sfilza di 1 iniziale. Idealmente, dunque, vorremmo negarli tutti e poi porli all'interno di una porta OR a 14 ingressi. Una soluzione equivalente (e più realistico, visto che porte simili non vengono prodotte, nella realtà) è quella di usare due porte NAND, una a 8 ingressi e una a 6, e poi inserire il loro output in una porta OR, da collegare alla porta della scheda PIO.
Questa è la prima soluzione. Vi allego anche una orribile immagine fatta da me a mano, con la penna rossa (senza motivo). Sul libro ne trovate una ancora più realistica, che utilizza porte NOR invece della OR.



Ora, si può fornire anche una soluzione diversa, più facile. Si deve notare che, gli indirizzi della EPROM sono tutti e soli quelli che iniziano per 0 (quindi, ha 0 sulla linea A_{15}). Quindi, si collega direttamente la porta della EPROM a A_{15}. Per la RAM, usando un ragionamento simile, si nota che gli indirizzi sono tutti e solo quelli che iniziano per 10 (quindi, hanno 1 su A_{15} e 0 su A_{14}) quindi, si nega in segnale di A_{14} e si pone insieme a quello di A_{15} in una porta NAND a 2 ingressi, il cui output è collegato con la RAM. Per la PIO, invece vale lo stesso ragionamento, ma con doppio 1 iniziale. Quindi, semplicemente, si collegano A_{14} e A_{15} alla PIO tramite una porta NAND. Ovvimanete, tramite questa soluzione non è possibile aggiungere altri componenti a questa macchina, rendendola molto meno flessibile a modifiche. Vi metto sempre l'immagine brutta fatta a mano, ma questa volta la trovate identica sul libro.


A0
A1
A2
A3
A4
A5
A6
A7
A8
A9
A10
A11
A12
A13
A14
A15
CS
EPROM
2K x 8
CS
RAM
2K x 8
CS
FIO