Architettura degli Elaboratori, 2006-07

Terza Esercitazione di Verifica Intermedia

Consegna: lezione di venerdì 15 dicembre, ore 11

L'elaborato, da presentare in una forma leggibile agevolmente, deve contenere spiegazioni chiare ed esaurienti, utilizzando la corretta terminologia ed i concetti del corso.
Insieme a nome e cognome indicare l'anno di corso di ogni studente.

Domanda 1

Si consideri il programma della Domanda 1 della Seconda Esercitazione. Il programma viene eseguito da un sistema che, oltre alle caratteristiche di quello della Domanda 2 della Seconda Esercitazione, possiede una memoria cache con le seguenti caratteristiche:

- operante su domanda,
- capacità di 64K parole e blocchi di 8 parole,
- indirizzamento con il metodo associativo su insiemi, con 2 blocchi per insieme,
- scritture gestite con il metodo Write Through.

La memoria principale è interallacciata con 4 moduli. Non esiste cache secondaria.

a) Valutare il tempo di completamento del programma, la performance, e l'efficienza relativa nell'uso della cache. Spiegare il procedimento seguito.

b) Ricavare la Parte Operativa dell'unità cache di questo sistema, spiegando come è stata ottenuta, e verificare che il tempo di accesso in cache, in assenza di fault, sia quello ipotizzato al punto *a).*

Domanda 2

a) Si consideri il programma corrispondente alla seguente computazione:

$$i = 0 \,..\, N - 1 : A[i] = F\,(A[i], B)$$

con A e B array di N = 32K interi, ed F funzione data (libreria). Spiegare come, per questo programma, l'uso della cache operante su domanda può essere ottimizzato, e determinare il corrispondente numero di fault. Nel far questo si consideri che il set di istruzioni è arricchito da opzioni associate alle istruzioni LOAD e STORE per indicare che il blocco contenente la parola riferita, una volta allocato in cache, può essere deallocato dalla cache in qualunque momento, oppure che non deve essere deallocato (finché non viene eseguita una istruzione che indica che il blocco può essere deallocato in qualunque momento).

b) Dimostrare che dei tre metodi di indirizzamento, visti a proposito della memoria cache, solo il metodo completamente associativo può essere utilizzato (e di fatto lo è) anche nella gerarchia memoria virtuale - memoria principale.

Domanda 3

Il messaggio di interruzione di un certo sottoinsieme di unità di I/O è costituito dalla coppia di parole (codice evento = sveglia processo, nome di processo). L'Handler associato a questo evento provvede a svegliare il processo il cui nome unico è dato dalla seconda parola. Il sistema prevede 4 code dei processi pronti, ognuna delle quali corrisponde ad una diversa classe a cui che i processi possono appartenere. La funzionalità di sveglia inserisce il PCB del processo da svegliare in fondo alla coda pronti che compete a quel processo.

a) Scrivere l'Handler, spiegando accuratamente come si è ragionato, le strutture dati utilizzate, ed il codice assembler.

b) Valutare l'intervallo di tempo necessario a trattare questo tipo di interruzione, a partire dall'istante in cui l'interruzione viene rilevata fino all'istante in cui si conclude l'elaborazione dell'Handler.

Domanda 4

Il programma della Domanda 1 è completato in modo che gli array *A, B, C* siano letti, uno alla volta, da un certo dispositivo USER_DEV e che l'array *C* modificato sia scritto sullo stesso dispositivo.

Si suppone che, a livello del linguaggio delle applicazioni, USER_DEV sia visto attraverso i comandi *leggi(n, x)* e *scrivi (n, x),* dove *x* è il nome di una variabile rappresentata da *n* byte.

Il sistema operativo è a processi cooperanti a scambio di messaggi.

a) Mostrare e spiegare la *compilazione* in linguaggio concorrente del programma *in un processo* MY_PROG, supponendo che all'unità I/O_USER_DEV sia associato un processo Driver_USER_DEV. In seguito all'eventuale esito anomalo dei comandi *leggi* e *scrivi*, il processo MY_PROG invia un messaggio "errore" al dispositivo SYSTEM_DEV dopo di che termina. *Mostrare e spiegare tutti gli oggetti contenuti nella memoria virtuale del processo MY_PROG*.

b) Spiegare in seguito a quali eventi il processo MY_PROG può *transire in stato di attesa*, ed in seguito a quali eventi può essere *risvegliato*.

c) Spiegare se la versione *assembler* di MY_PROG va modificata o meno, ed eventualmente come, nel caso che sia eliminato il processo Driver_USER_DEV,

d) Spiegare se, nella versione *c),* è visibile a MY_PROG quale modello, DMA o Memory Mapped I/O, è adottato per i trasferimenti di I/O.