Elementi di Architettura e Sistemi Operativi

Bioinformatica - Tiziano Villa

10 Luglio 2019

Nome e Cognome:
Matricola:
Posta elettronica:

problema	punti massimi	i tuoi punti
problema 1	6	
problema 2	9	
problema 3	5	
problema 4	10	
totale	30	

1. Si consideri il seguente programma che crea processi invocando la chiamata di sistema `fork`.

```
#include <sys/types.h>
#include <stdio.h>
#include <unistd.h>

int main()
{
  pid_t pid;

  /* crea mediante fork un processo figlio */
  pid = fork();

  if (pid < 0) { /* si e' verificato un errore */
    fprintf(stderr, "Fork fallita");
    return 1;
  }
  else if (pid == 0) /* processo figlio */
    execlp("bin ls", "ls", NULL);
    printf("LINE J");
  }
  else { /* processo padre */
  /* il padre attende che il figlio termini */
    wait(NULL);
    printf("Figlio completato"));
  }

  return 0;
}
```

(a) Si spieghi la semantica della chiamata di sistema `fork`.

Si analizzi il codice precedente spiegandone il funzionamento.

Traccia.

La famiglia di funzioni `exec` rimpiazza i'immagine del processo corrente con un nuovo processo specificato in argomento.

Traccia di soluzione.

La chiamata di sistema fork() crea un nuovo processo figlio che avra' una copia dello spazio degl'indirizzi del processo genitore. Entrambi i processi genitore e figlio continuano l'esecuzione dall'istruzione successiva alla chiamata di sistema fork(), con una differenza: la chiamata di sistema fork() restituisce il valore 0 nel processo figlio, ma restituisce l'identificatore del processo figlio (il PID diverso da 0) nel processo genitore. Il processo genitore invocando la chiamata di sistema wait() si autorimuove dalla coda dei processi pronti fino alla terminazione del figlio. Quando il processo figlio termina, il processo genitore chiude la chiamata di sistema wait() e continua con il resto del codice.

(b) In quali circostanze si eseguira' la linea di codice `printf("LINE J")` ?

Si argomenti la risposta.

Traccia di soluzione.

Il processo figlio eseguira' `printf("LINE J")` solo se la chiamata precedente alla funzione `exec` fallisce. Se la `exec` va a buon fine, essa sovrascrive l'immagine in memoria del processo figlio con quella del programma `ls`, e quindi la `printf("LINE J")` non e' eseguita.

2. Il sistema operativo BTV ha indirizzi virtuali a 21 cifre binarie, anche se su alcune versioni per applicazioni industriali ha indirizzi fisici di 16 cifre binarie. La dimensione di pagina di 2 KB. Quante voci ci sono in ciascuna delle seguenti strutture dati ? Si argomentino le risposte.

   (a) Una tavola delle pagine convenzionale, a livello singolo.

   (b) Una tavola delle pagine invertita.

   In entrambi i casi, si spieghi concisamente il modello (tavola delle pagine a un livello, tavola della pagine invertita).

   Traccia di soluzione.

   (a) La dimensione dello spazio d'indirizzamento logico e' data da
   $2^m B$ = numero di pagine x dimensione pagina B
   dove m e' il numero di cifre binarie degl'indirizzi virtuali. Nel nostro caso $m = 21$ e la dimensione di pagina e' $2KB = 2.2^{10} = 2^{11}B$.
   Segue che $2^{21}B$ = numero di pagine x $2^{11}B$, da cui il numero di pagine e' 2^{10}.
   Dato che in una pagina a livello singolo il numero di voci (elementi) e' uguale al numero delle pagine, esso e' 2^{10}.

   (b) La dimensione dello spazio d'indirizzamento fisico e' data da
   $2^{16}B$ = numero pagine x dimensione pagina B, e quindi nel nostro caso $2^{16}B$ = numero pagine x $2^{11}B$.
   Segue che il numero di pagine e' 2^5, che e' il numero di voci (elementi) della tavola delle pagine invertita.

3. Si consideri il seguente codice LC-3

```
        LDR R0, R5, #0
        BRz SALTA

        LDR R0, R5, #-1
        ADD R0, R0, #1
        STR R0, R5, #-1

        LDR R0, R5, #-2
        ADD R0, R0, #-1
        STR R0, R5, #-2
        BR FINE

SALTA LDR R0, R5, #-1
      ADD R0, R0, #-1
      STR R0, R5, #-1

      LDR R0, R5, #-2
      ADD R0, R0, #1
      STR R0, R5, #-2
FINE ...
```

Si spieghi il suo funzionamento, sia commentando le singole istruzioni che la procedura complessiva.

Traccia di soluzione

Si stia attenti alla semantica di LDR e STR.

LDR legge il valore in memoria all'indirizzo contenuto in R5 (o all'indirizzo precedente di 1 o di 2 quello contenuto in R5) e lo salva in R0: $R0 \leftarrow mem[R5]$ (o $R0 \leftarrow mem[R5-1]$ o $R0 \leftarrow mem[R5-2]$).

STR scrive in memoria il contenuto di R0 all'indirizzo precedente di 1 o di 2 quello contenuto in R5: $mem[R5-1] \leftarrow R0$ o $mem[R5-2] \leftarrow R0$.

Se chiamiamo rispettivamente x, y e z le variabili cui puntano $R5, R5-1$ e $R5-2$, (cioe' x stia per $mem[R5]$, y per $mem[R5-1]$ e z per $mem[R5-2]$), allora possiamo dire che il frammento di codice LC-3 precedente e' la compilazione del seguente frammento di codice C:

```
if (x) {
  y++;
  z--
} else {
  y--;
  z++
)
```

nell'ipotesi appunto che $R5$ punti alla variabile x (cioe' ne contenga l'indirizzo), che $R5 - 1$ punti alla variabile y e che $R5 - 2$ punti alla variabile z.

4. Si progetti un circuito sequenziale che funziona come contatore di Johnson a 4 cifre binarie secondo la sequenza seguente: $0000 \rightarrow 1000 \rightarrow 1100 \rightarrow 1110 \rightarrow 1111 \rightarrow 0111 \rightarrow 0011 \rightarrow 0001 \rightarrow 0000$ e via ripetendo.

   (a) Si disegni il grafo delle transizioni di una macchina a stati finiti che corrisponde alla specifica. S'indichi lo stato iniziale.

   Si minimizzi il numero degli stati della macchina proposta.

(b) Si scriva la tavola delle transizioni con gli stati futuri e le uscite e la si codifichi.

(c) Supponendo di usare bistabili di tipo D, si derivino le equazioni minimizzate di eccitazione degl'ingressi dei bistabili e le equazioni minimizzate delle uscite.

(d) Si realizzi il circuito sequenziale corrispondente con bistabili di tipo D campionati sul fronte di salita, invertitori e porte NAND (a 2, 3, o 4 ingressi). Si etichettino con chiarezza i segnali.