



Politecnico di Milano – Sede di Cremona
Anno Accademico 2023/2024

Architettura dei Calcolatori e Sistemi Operativi

Esame – 19.07.2024

Prof. Carlo Brandolese

Cognome _____

Nome _____

Matricola _____

Firma _____

Istruzioni

1. Scrivere con cura, negli spazi sopra segnati, il proprio cognome, nome, numero di matricola e apporre la firma.
2. È vietato consultare libri, eserciziari, appunti ed utilizzare la calcolatrice e qualunque strumento elettronico (inclusi i cellulari), pena l'invalidazione del compito.
3. Il testo, debitamente compilato, deve essere riconsegnato in ogni caso.
4. Il tempo della prova è di 3 ore

Valutazione

Domanda	Voto	Note
A		
B		
C		
D		
E		
F		

Domanda A

Dato il codice mostrato nel seguito, si scriva un programma che calcola e salva in memoria il risultato della seguente espressione:

$$Z = X[n] * Y[n+1] + X[n+1] * Y[n]$$

Nella traduzione si cerchi di minimizzare il numero di istruzioni ed il numero di registri utilizzati.

```
.data
X:    .word    1,  2,  3,  4,  5,  6
Y:    .word   10, 20, 30, 40, 50, 60
Z:    .space   4
N:    .word    3

.text
main:
    # Prepares 4N
    la    t0, N           # t0 = &N
    lw    t0, (t0)        # t0 = N
    slli  t0, t0, 2       # t0 = 4N

    # Prepares base addresses
    la    t1, X           # t1 = &X
    add   t1, t1, t0      # t1 = &X + N
    la    t2, Y           # t2 = &Y
    add   t2, t2, t0      # t2 = &Y + N

    # Computes X[N] * Y[N+1]
    lw    t3, 0(t1)       # t3 = *t1 = X[N]
    lw    t4, 4(t2)       # t4 = *(t2+4) = Y[N+1]
    mul   t0, t3, t4      # t0 = X[N] * Y[N+1]

    # Computes X[N+1] * Y[N]
    lw    t3, 4(t1)       # t3 = *t1 = X[N]
    lw    t4, 0(t2)       # t4 = *(t2+4) = Y[N+1]
    mul   t3, t3, t4      # t3 = X[N+1] * Y[N]

    # Final sum and stores
    add   t3, t3, t0      # Result
    la    t0, Z           # t0 = &Z
    sw    t3, (t0)        # *t0 = Result
```

Domanda B

Si sviluppi in C un programma **mirror** che riceve sulla linea di comando un elenco di parole e stampa sullo standard output, una per riga, le parole ricevute scritte al contrario. Il programma deve essere parallelizzato in modo tale che ogni parola sia elaborata da un processo diverso.

A titolo di esempio si immagini che il programma **mirror** sia eseguito come segue:

```
$> ./mirror hello world this is an example
olleh
dlrow
siht
si
na
elpmaxe
```

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

int main( int argc, char** argv)
{
    for( int i = 1; i < argc; i++ )
    {
        if( fork() == 0 )
        {
            // Child
            int len = strlen( argv[i] );
            for( int j = len - 1; j >= 0; j-- ) {
                putchar( argv[i][j] );
            }
            putchar( '\n' );
            exit(0);
        }
    }

    for( int i = 1; i < argc; i++ )
    {
        wait( NULL );
    }

    exit(0);
}
```

Domanda C

Si consideri un sistema con uno spazio di indirizzamento complessivo pari a 4 GByte ed una struttura di cache a due livelli L0 ed L1, con le caratteristiche seguenti:

	L0 CACHE	L1 CACHE
Associatività	Diretta	Diretta
Dimensione totale	32 KB	128 KB
Dimensione linea	128 B	256 B
Tempo di accesso	1 ns	2 ns
Hit rate	99 %	98 %

Si indichi la struttura dell'indirizzo visto dalle cache, descrivendo i vari campi e il loro significato.

Struttura dell'indirizzo L0 CACHE

$$\text{Offset} = \log_2(128) = 7 \text{ bit}$$

$$\text{Index} = \log_2(32\text{K} / 128) = 8 \text{ bit}$$

$$\text{Tag} = \log_2(4\text{G}) - 7 - 8 = 32 - 7 - 8 = 17 \text{ bit}$$

Struttura dell'indirizzo L1 CACHE

$$\text{Offset} = \log_2(256) = 8 \text{ bit}$$

$$\text{Index} = \log_2(128\text{K} / 256) = 9 \text{ bit}$$

$$\text{Tag} = \log_2(4\text{G}) - 8 - 9 = 32 - 8 - 9 = 15 \text{ bit}$$

Sapendo che:

- L'accesso alla memoria ed alla cache L1 avviene a parole di 64 bit
- Il tempo di accesso alla RAM in modalità normale è di 100 ns
- Il tempo di accesso alla RAM in modalità burst è di
 - 200 ns per il primo accesso
 - 40 ns per gli accessi successivi

Si calcoli il tempo medio di accesso alla memoria.

Per la cache di livello zero si ha:

$$T = P_{\text{hit},L0} * T_{\text{hit},L0} + P_{\text{miss},L0} * [T_{\text{hit},L0} + (128/8) * T_{\text{miss},L0}]$$

In questa espressione tutte le grandezze sono note a meno di $T_{\text{miss},L0}$ che deve essere calcolato immaginando che vi sia solo la cache L1

$$\begin{aligned} T_{\text{miss},L0} &= P_{\text{hit},L1} * T_{\text{hit},L1} + P_{\text{miss},L1} * (T_{\text{hit},L1} + T_{\text{miss},L1}) \\ &= 0.98 * 2\text{ns} + 0.02 * [2\text{ns} + 200\text{ns} + (256/8-1) * 40\text{ns}] \\ &= 1.96\text{ns} + 0.02 * 1442\text{ns} \\ &= 30.8 \text{ ns} \end{aligned}$$

A questo punto si può calcolare il tempo medio complessivo:

$$\begin{aligned}
 T &= P_{hit,L0} * T_{hit,L0} + P_{miss,L0} * [T_{hit,L0} + (128/8) * T_{miss,L0}] \\
 &= 0,99 * 1ns + 0.01 * (1ns + 16 * 30.8 ns) \\
 &= 0.99ns + 0.01 * 493.8ns \\
 &= 5.918 ns
 \end{aligned}$$

Domanda D

Si considerino i seguenti processi:

Process	Arrival Time	Execution Time
P1	0	8
P2	2	6
P3	6	3
P4	12	3
P5	16	5

Si simuli lo scheduling di tali processi secondo gli algoritmi indicati e si calcoli il tempo medio di attesa nei due casi. In caso di preemption, per prima cosa il nuovo processo entra nella coda e solo dopo il processo corrente esce dallo stato di esecuzione.

Round Robin preemptive con periodo di 5 unità di tempo

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
P1	7	6				5						4	3			2					1	0				
P2			5	4	3		2	1	0																	
P3										2			1	0												
P4														2		1	0									
P5																		4	3			2	1	0		

Tempo medio di attesa = [(3+4+3+4) + (1) + (3+2) + (2+1) + (2+2)] / 5 = 27 / 5 = 5.4

First In First Out

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
P1	7	6	5	4	3	2	1	0																		
P2											5	4	3	2	1	0										
P3																2	1	0								
P4																		2	1	0						
P5																						4	3	2	1	0

Tempo medio di attesa = 0 + 6 + 8 + 5 + 4 = 23 / 5 = 4.6

Domanda E

Si consideri un sistema dotato di una memoria fisica di 128KB e di uno spazio di indirizzamento logico di 1MB. Tale sistema supporta la memoria virtuale con pagine di 2KB ed è dotato di una MMU con TLB di 32 elementi. Si indichi la dimensione in bit degli indirizzi:

Numero di pagina virtuale: $\log_2(1\text{M} / 2\text{K}) = 9$ bit

Offset nella pagina virtuale: $\log_2(2\text{K}) = 11$ bit

Numero di pagina fisica: $\log_2(128\text{K} / 2\text{K}) = 6$ bit

Offset nella pagina fisica: $\log_2(2\text{K}) = 11$ bit

Si considerino quindi due programmi X e Y così composti:

CX: 8K DX: 2K PX: 2K Entry point: 0x017E8
CY: 6K DY: 12K PY: 2K Entry point: 0x00AF0

Si rappresenti la memoria logica dei due processi.

Programma X

Numero Pagina Virtuale	Contenuto
0	CX0
1	CX1
2	CX2
3	CX3
4	DX0
5	HX0
6	HX1
511	PX0

Programma Y

Numero Pagina virtuale	Contenuto
0	CY0
1	CY1
2	CY2
3	DY0
4	DY1
5	DY2
6	DY3
511	PY0

Si supponga che:

- Lo heap viene allocato immediatamente dopo la sezione DATA
- L'esecuzione di un programma avviene caricando la prima pagina di codice necessaria, la prima pagina di dati e la pagina di stack corrente. Il resto delle pagine viene caricato in demand paging.
- La TLB consente di avere al più 8 pagine per ogni processo

Si risponda ai seguenti quesiti indicando il **contenuto** della pagina virtuale, ed il relativo **numero**, cioè l'NPV.

Quali sono le pagine virtuali di X caricate in memoria al momento della creazione del processo P che esegue il codice di X.

CX2 / 2, DX0 / 4, PX0 / 511

Quali sono le pagine virtuali destinate a contenere una zona di memoria di 2800 bytes allocata dinamicamente dal programma X in esecuzione in P.

HX0 / 5, HX1 / 6

Quali sono le pagine virtuali di Y caricate in memoria al momento della creazione del processo Q che esegue il codice di Y.

CY1 / 1, DY0 / 3, PY0 / 511

Domanda F

Si descriva il concetto di context switch e si spieghi come viene realizzato dal sistema operativo, facendo riferimento alle varie funzioni utilizzate a tale scopo.

Si veda il materiale didattico.

