Dežurni nastavnik: _____

Elektrotehnički fakultet u Beogradu
Katedra za računarsku tehniku i informatiku

Predmet: Multiprocesorski sistemi (SI4MPS)
Nastavnik: dr Milo Tomašević, vanr. prof.
Asistent: dipl. ing. Marko Mišić
Ispitni rok: Drugi kolokvijum (novembar 2012.)
Datum: 28.11.2012.

Kandidat:* _____

Broj Indeksa:* _____

Kolokvijum traje 105 minuta, prvih sat vremena nije dozvoljeno napuštanje kolokvijuma. Upotreba literature nije dozvoljena.

Zadatak 1	_____/10	*Zadatak 5*	_____/15
Zadatak 2	_____/15	*Zadatak 6*	_____/10
Zadatak 3	_____/15	*Zadatak 7*	_____/15
Zadatak 4	_____/20		

Ukupno na kolokvijumu: _____/100

Napomena: Ukoliko u zadatku nešto nije dovoljno precizno definisano, student treba da uvede razumnu pretpostavku, da je uokviri (da bi se lakše prepoznala prilikom ocenjivanja) i da nastavi da izgrađuje preostali deo svog odgovora na temeljima uvedene pretpostavke. Kod pitanja koja imaju ponuđene odgovore treba **samo zaokružiti** jedan odgovor. Na ostala pitanja odgovarati **čitko, kratko i precizno**.

* popunjava student.

1. [10] Objasniti *model cene komunikacije.* Na osnovu njega definisati indikatore performanse bitne sa aspekta procesora.

2. [15] Objasniti prednosti i nedostatke korišćenja zajedničke keš memorije u multiprocesorskim sistemima.

3. [15] Procesi A i B se izvršavaju na dva procesora sa privatnim keš memorijama i sinhronizuju se preko deljene promenljive S čija je početna vrednost 0 na sledeći način:
   - A čeka da S bude 0, onda nešto radi i postavi ga na 1
   - B čeka da S bude 1, onda nešto radi i postavi ga na 0.

   Ispisati niz transakcija na magistrali koje se javljaju ukoliko se koristi strategija a) *invalidacije* i b) *ažuriranja*. Na osnovu toga, komentarisati koja strategija je pogodnija u ovakvim slučajevima.

4. [20] Kod protokola Dragon precizno objasniti:
   a) kakva je uloga stanja *Shared-Modified* i kako bi se ono moglo izbeći,
   b) precizno opisati šta se dešava prilikom pogotka pri upisu (*write hit*)

5. [15] Neka je data matrica dimenzija MxN koja sadrži podatke tipa `int`. Program treba da vrši određenu obradu nad kolonama matrice. Nakon obrade kolone, ona ima iste dimenzije, ali izmenjene elemente. Proces upravljač šalje neobrađene kolone matrice slobodnim procesima radnicima i prihvata rezultate od radnika, sve dok ima neobrađenih kolona. Procesi radnici prihvataju po jednu kolonu matrice, obrađuju ih i vraćaju nazad sve dok ih upravljač ne obavesti da više nema posla. Proces upravljač je implementiran funkcijom `master()` koja je data u prilogu. Koristeći rutine iz MPI biblioteke napisati funkciju `slave()` koja implementira proces radnika u opisanom *manager – worker* modelu obrade podataka.

```
enum tags {ROW_INDEX_TAG = 11111, RESULT_TAG, END_TAG};
void readMatrix (int matrix[][N], int row, int col);
printMatrix(matrix, rows, cols);
void processColumn (int column[], int n);

void master (int size) {
  int matrix[ROW][COL];
  int rows, cols, i, colSent = 0, resultReceived = 0 ;
  MPI_Datatype columnType;
  MPI_Status status;

  scanf("%d%d", &rows, &cols);
  readMatrix(matrix, rows, cols);

  MPI_Bcast(&rows, 1, MPI_INT, MASTER, MPI_COMM_WORLD);

  MPI_Type_vector(rows, 1, COL, MPI_INT, &columnType);
  MPI_Type_commit(&columnType);

  for (i = 1; i < size; i++) {
    if (colSent < cols) {
      MPI_Send(&matrix[0][colSent], 1,
               columnType, i, colSent++, MPI_COMM_WORLD);
    } else {
      MPI_Send(&matrix[0][0], 1, columnType, i, END_TAG, MPI_COMM_WORLD);
    }
  }

  while (resultReceived < cols) {
    int colRecv;

    MPI_Recv(&colRecv, 1, MPI_INT, MPI_ANY_SOURCE,
             ROW_INDEX_TAG, MPI_COMM_WORLD, &status);
    MPI_Recv(&matrix[0][colRecv], 1, columnType,
             status.MPI_SOURCE, RESULT_TAG, MPI_COMM_WORLD, &status);
    resultReceived++;

    if (colSent < cols) {
      MPI_Send(&matrix[0][colSent], 1, columnType,
               status.MPI_SOURCE, colSent++, MPI_COMM_WORLD);
    } else {
        MPI_Send(&matrix[0][0], 1, columnType,
                 status.MPI_SOURCE, END_TAG, MPI_COMM_WORLD);
    }
  }

  printMatrix(matrix, rows, cols);
  MPI_Type_free(&columnType);
}
```

6. [10] U čemu je razlika između blokirajućih i neblokirajućih rutina za *point-to-point* komunikaciju u MPI standardu? Da li rutine za slanje i prijem moraju biti istog tipa ili je moguće njihovo kombinovanje u tom smislu?

7. [15] Dat je multiprocesorski sistem sa 4 identična procesora, koji koristi MOESI protokol za održavanje koherencije keš memorije. Svaka keš memorija ima po 2 ulaza, koji su veličine jedne reči. Preslikavanje je direktno. Početne vrednosti podataka su 0. Svaki upis uvećava vrednost izmenjenog podatka za 1. Na početku su sve keš memorije prazne. Data je sledeća sekvenca pristupa memoriji:

1. P0,R,A0 2. P1,W,A2	3. P1,R,A1 4. P2,R,A2	5. P0,W,A0 6. P0,W,A2	7. P0,R,A2 8. P2,W,A1

Napisati stanja koherencije u svim procesorima i stanje memorije posle svake promene i skicirati opisani sistem u trenutku 8. [10 poena]
Koliko puta koji od procesora pristupa memoriji? Za svaki pristup navesti razlog. [3 poena]
Koliki je Hit Rate za svaki od procesora (brojati i čitanje i upis, prikazati zbirno)? [2 poena]

CPU	Broj pogodaka	Ukupan broj pristupa	Hit rate
P0			
P1			
P2			
P3			

Pristupi memoriji

Trenutak 1

P0			P1			P2			P3		

Memorija	
A0	
A1	
A2	
A3	

Pristupi memoriji:

_____

Trenutak 2

P0			P1			P2			P3		

Memorija	
A0	
A1	
A2	
A3	

Pristupi memoriji:

_____

Trenutak 3

P0			P1			P2			P3		

Memorija	
A0	
A1	
A2	
A3	

Pristupi memoriji:

_____

Trenutak 4

P0			P1			P2			P3		

Memorija	
A0	
A1	
A2	
A3	

Pristupi memoriji:

_____

Trenutak 5

P0			P1			P2			P3		

Memorija	
A0	
A1	
A2	
A3	

Pristupi memoriji:

_____

Trenutak 6

P0			P1			P2			P3		

Memorija	
A0	
A1	
A2	
A3	

Pristupi memoriji:

_____

Trenutak 7

P0			P1			P2			P3		

Memorija	
A0	
A1	
A2	
A3	

Pristupi memoriji:

_____

Trenutak 8

P0			P1			P2			P3		

Memorija	
A0	
A1	
A2	
A3	

Pristupi memoriji:

_____