PARZIALE DI ARCHITETTURE DEI CALCOLATORI 2019/2020
2 dicembre 2019

NOME: **COGNOME:**

MATRICOLA:

Scrivere in stampatello NOME, COGNOME e MATRICOLA su ogni foglio.
Al termine, si DOVRANNO consegnare tutti i fogli ricevuti.

NOTA: *La prova si considera superata con esito positivo solo se sono verificate entrambe le seguenti condizioni:*

a) *Votazione complessiva maggiore o uguale a 18/30;*
b) *Esercizi 2 e 3 svolti in modo sufficiente.*

ESERCIZIO 1 [3 punti] Effettuare la sintesi della rete logica che implementa la funzione booleana

$$F(A,B,C,D) = m_1+m_4+m_5+m_6+m_9+m_{11}+m_{12}+m_{13}+m_{14}$$

ESERCIZIO 2 [7 punti] Scrivere una procedura ricorsiva in assembler MIPS che dato un array di interi `A` di lunghezza `m`, con `m` pari ad una potenza di 2, e due interi `k` e `h`: 1) sostituisce nell'array `A` tutte le occorrenze di `k` con `h`; 2) restituisce il numero di sostituzioni effettuate.

Possibile soluzione in C

```
int sost_conta(int *A, int m, int k, int h) {
      if (m==1) {
            if (A[0]==k) {
                  A[0]=h;
                  return 1;
            }
            else
                  return 0;
      }
      else
            return sost_conta(A,m/2,k,h) + sost_conta(A+m/2,m/2,k,h);
}
```

Gli argomenti alla procedura sono passati tramite i registri `$a0, $a1, $a2`, e `$a3`. La procedura restituisce il numero di sostituzioni effettuate in `$v0`. Si richiede di commentare il codice.
Nota Bene: 1) Non si possono usare pseudoistruzioni; 2) è richiesto l'uso delle convenzioni di chiamata a procedura usate dall'assembler MIPS.

ESERCIZIO 3 [5 punti] Scrivere una procedura in assembler MIPS che, dato un array di interi `A` di lunghezza `m`, restituisce il valore $v=\sum_{i=0}^{m-1}\sqrt{A[i]}$. Per il calcolo della radice quadrata si può assumere l'esistenza di una procedura di libreria `radiceq` che, passato in `$a0` un argomento intero, restituisce in `$v0` la parte intera della radice quadrata di `$a0`. Si richiede di commentare il codice.
Nota Bene: 1) Non si possono usare pseudoistruzioni; 2) è richiesto l'uso delle convenzioni di chiamata a procedura usate dall'assembler MIPS.

ESERCIZIO 4 [2 punti] In relazione all'implementazione in assembler della procedura MIPS dell'esercizio precedente, si chiede di indicare il contenuto dei campi immediate/address delle istruzioni: `beq`, `bne`, `j` e `jal`, al termine della fase di "collegamento", nell'ipotesi che le due funzioni vengano posizionate in memoria, una subito dopo l'altra, a partire dall'indirizzo 0x00480A40 (segmento testo).

ESERCIZIO 5 [8 punti] Scrivere una funzione in C che riceve come parametri di ingresso una matrice *A* sparsa (ovvero una matrice i cui valori sono quasi tutti uguali a 0) di interi, il numero di righe di *A*, il numero di colonne di *A* ed il numero di elementi diversi da zero (*nz*) presenti in *A* e restituisce in output una matrice *B*, che è una versione più compatta di *A*. La matrice *B* ha dimensione $nz \times 3$ e contiene nella prima colonna i valori di *A* diversi da 0 e nelle restanti due colonne rispettivamente gli indici di riga e di colonna degli elementi di *A* diversi da 0.
Esempio: data in ingresso la seguente matrice *A,* la funzione restituisce in output la matrice *B*:

$$A = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 5 & 8 & 0 & 0 \\ 0 & 0 & 3 & 0 \\ 0 & 6 & 0 & 0 \end{pmatrix} \qquad B = \begin{pmatrix} 5 & 1 & 0 \\ 8 & 1 & 1 \\ 6 & 3 & 1 \\ 3 & 2 & 2 \end{pmatrix}$$

ESERCIZIO 6 [4 punti] Illustrare lo schema a livello di porte logiche e descrivere il funzionamento di un Flip Flop D master slave.

ESERCIZIO 7 [3 punti] Dato il seguente programma in C, indicare il valore stampato dall'istruzione `printf`, motivando adeguatamente la risposta fornita.

```c
#include <stdio.h>

void swap1(int *a, int *b)
{
	int tmp;
	tmp = *a;
	*a = *b;
	*b = tmp;
}

void swap2(int a, int b)
{
	int tmp;
	tmp = a;
	a = b;
	b = tmp;
}
```

```c
void swap3(int *a, int *b)
{
	int *tmp;
	tmp = a;
	a = b;
	b = tmp;
}

int main(void)
{
	int x = 8, y = 7;

	if (x < y) swap1(&x, &y);
	if (x < y) swap2(x+1, y);
	if (x >= y) swap3(&x, &y);
	printf("%d, %d\n", x, y);
}
```

PARZIALE DI ARCHITETTURE DEI CALCOLATORI 2019/2020
2 dicembre 2019

NOME: **COGNOME:**

MATRICOLA:

Scrivere in stampatello NOME, COGNOME e MATRICOLA su ogni foglio.
Al termine, si DOVRANNO consegnare tutti i fogli ricevuti.

NOTA: *La prova si considera superata con esito positivo solo se sono verificate entrambe le seguenti condizioni:*

a) *Votazione complessiva maggiore o uguale a 18/30;*
b) *Esercizi 2 e 3 svolti in modo sufficiente.*

ESERCIZIO 1: [3 punti] Effettuare la sintesi della rete logica che implementa la funzione booleana

$$F(X,Y,W,Z)=m_1+m_3+m_4+m_5+m_6+m_7+m_9+m_{11}+m_{12}.$$

ESERCIZIO 2: [7 punti] Scrivere una procedura ricorsiva in assembler MIPS che dato un array di caratteri `A` di lunghezza `m`, con `m` pari ad una potenza di 2, e un carattere `c`, conta il numero di occorrenze del carattere `c` nell'array `A`.

Possibile soluzione in C

```
int conta(char *A, int m, char c) {
      if (m==1) {
            if (A[0]==c)
                  return 1;
            else
                  return 0;
      }
      else
            return conta(A,m/2,c)+ conta(A+m/2,m/2,c);
}
```

Gli argomenti alla procedura sono passati tramite i registri `$a0, $a1 e $a2`. La procedura restituisce il numero di occorrenze in `$v0`. Si richiede di commentare il codice.
Nota Bene: 1) Non si possono usare pseudoistruzioni; 2) è richiesto l'uso delle convenzioni di chiamata a procedura usate dall'assembler MIPS.

ESERCIZIO 3 [5 punti] Scrivere una procedura assembler che, dato un array di interi `A`, di lunghezza `m`, restituisce il valore $v=\sum_{i=0}^{m-1} e^{A[i]}$. Per il calcolo della funzione esponenziale si può assumere l'esistenza di una procedura di libreria `exp` che, passato in `$a0` un argomento intero, restituisce in `$v0` la parte intera della esponenziale di `$a0.` Si richiede di commentare il codice.
Nota Bene: 1) Non si possono usare pseudoistruzioni; 2) è richiesto l'uso delle convenzioni di chiamata a procedura usate dall'assembler MIPS.

ESERCIZIO 4 [2 punti] In relazione all'implementazione in assembler della procedura MIPS dell'esercizio precedente, si chiede di indicare il contenuto dei campi immediate/address delle istruzioni: `beq`, `bne`, `j` e `jal`, al termine della fase di "collegamento", nell'ipotesi che le due

funzioni vengano posizionate in memoria, una subito dopo l'altra, a partire dall'indirizzo 0x00407F00 (segmento testo).

ESERCIZIO 5 [8 punti] Scrivere una funzione in C che riceve come parametri di ingresso una matrice *A* sparsa (ovvero una matrice i cui valori sono quasi tutti uguali a 0) di interi, il numero di righe di *A*, il numero di colonne di *A* ed il numero di elementi diversi da zero (*nz*) presenti in *A* e restituisce in output una matrice *B*, che è una versione più compatta di *A*. La matrice *B* ha dimensione 3×*nz* e contiene nella prima riga i valori di *A* diversi da 0 e nelle restanti due righe rispettivamente gli indici di riga e di colonna degli elementi di *A* diversi da 0.
Esempio: data in ingresso la seguente matrice *A,* la funzione restituisce in output la matrice *B*:

$$A = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 5 & 8 & 0 & 0 \\ 0 & 0 & 3 & 0 \\ 0 & 6 & 0 & 0 \end{pmatrix} \qquad B = \begin{pmatrix} 5 & 8 & 3 & 6 \\ 1 & 1 & 2 & 3 \\ 0 & 1 & 2 & 1 \end{pmatrix}$$

ESERCIZIO 6 [4 punti] Illustrare lo schema a livello di porte logiche e descrivere il funzionamento di un Flip Flop D master slave.

ESERCIZIO 7 [3 punti] Dato il seguente programma in C, indicare il valore stampato dall'istruzione `printf`, motivando adeguatamente la risposta fornita.

```c
#include <stdio.h>

void swap1(int x, int y)
{
      int temp;
      temp = x;
      x = y;
      y = temp;
}

void swap2(int *x, int *y)
{
      int temp;
      temp = *x;
      *x = *y;
      *y = temp;
}

void swap3(int *x, int *y)
{
      int *temp;
      temp = x;
      x = y;
      y = temp;
}

int main(void)
{
      int a = 3, b = 4;

      if (a >= b) swap1(a+2, b);
      if (a >= b) swap2(&a, &b);
      if (a < b) swap3(&a, &b);
      printf("%d, %d\n", a, b);
}
```