Informatica Generale
Appello Straordinario del 19 novembre 2009

Testo del compito con soluzioni

[Esercizio 1]

(1.a) Scrivere la tavola di verità della seguente funzione booleana in tre variabili:

```
G(A,B,C) = (A xor B) xor (C and not B)
```

Soluzione: La seguente tabella mostra nell'ultima colonna la tavola di verità della funzione **`G(A,B,C)`** [si intente che TRUE=1 e FALSE=0]. Le colonne D, E e F sono dei valori intermedi, utili per il calcolo della funzione.

A	B	C	D	E	F	G(A,B,C)
			A xor B	not B	C and E	E xor F
0	0	0	FALSO	VERO	FALSO	FALSO
0	0	1	FALSO	VERO	VERO	VERO
0	1	0	VERO	FALSO	FALSO	VERO
0	1	1	VERO	FALSO	FALSO	VERO
1	0	0	VERO	VERO	FALSO	VERO
1	0	1	VERO	VERO	VERO	FALSO
1	1	0	FALSO	FALSO	FALSO	FALSO
1	1	1	FALSO	FALSO	FALSO	FALSO

(1.b) Sfruttando la tavola di verità, disegnare un circuito combinatorio che calcola la funzione G usando solo porte AND, OR e NOT. Se si desidera, si possono usare porte AND e OR con più di due ingressi.

Soluzione: Il circuito che si richiedeva era quello basato sulla lettura della tavola di verità di F(A, B, C) come una espressione booleana, costituita dall'OR delle righe per cui la funzione vale TRUE, e per ogni riga l'AND delle variabili A, B e C oppure del loro negato. Quindi si chiedeva il circuito che rappresentava **G(A, B, C) = (¬A and ¬B and C) or (¬A and B and ¬C) or (¬A and B and C) or (A and ¬B and ¬C)**.

(per un disegno del circuito rivolgersi al docente.)

[Esercizio 2]

(2.a) Si consideri la seguente configurazione di bit, che rappresenta un numero in virgola mobile in un formato che prevede 1 bit per il segno, 4 bit per l'esponente (in complemento a 2) e 11 bit per la mantissa.
Quale numero in base 10 rappresenta?

1	0	1	0	1	1	0	1	1	0	1	1	0	0	0	0

Soluzione: -22,75

(2.b) Usando lo stesso formato del punto (2.a), con quale configurazione di bit si rappresenta il numero decimale `0.375` ?

Soluzione:

0	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0

(2.c) Sempre usando il formato del punto (2.a), qual è il massimo numero rappresentabile? E qual è il minimo numero strettamente maggiore di zero rappresentabile?

Soluzione: massimo numero rappresentabile 127,9375 (127 + 15/16)

0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Minimo numero rappresentabile 0,001953125 = 1/512 = 2^{-9}

0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0

[Esercizio 3]

(3.a) La tabella seguente mostra una porzione della memoria del computer contenente un programma in linguaggio macchina. Si assuma che l'esecuzione cominci con il valore **00** nel registro PC. Scrivere la sequenza di istruzioni che vengono eseguite finché la macchina non si fermi, mostrando il contenuto del PC e dei registri generali dopo ogni istruzione. Rispondere poi alle domande che seguono.

indirizzo	contenuto
00	25
01	03
02	A5
03	02
04	35
05	03
06	24
07	00
08	34
09	04
0A	B0
0B	03
0C	C0
0D	00

Soluzione:

Contenuto di PC	Contenuto di IR	Note
00	<sconosciuto>	Indirizzo della prima istruzione da eseguire
02	**2503**	Carico **03** in R5
04	**A502**	Ruoto il contenuto di R5 di due posizioni verso destra: R5 conterrà 1100 0000 cioè **C0**
06	**3503**	Scrivo il contenuto di R5 (cioè **C0**) nella cella **03**
08	**2400**	Carico **00** in R4
0A	**3404**	Scrivo il contenuto di R4 (cioè **00**) nella cella **04**
03	**B003**	Salto incondizionato (se R0 = R0) a **03**
05	**C000**	L'istruzione in **03-04** è **C000**, quindi la CPU si ferma

(3.b) Quale sequenza di bit contiene il registro 5 quando la macchina si ferma?

Soluzione: 1100 0000, cioè **C0**

(3.c) Quale sequenza di bit contiene il program counter quando la macchina si ferma?

Soluzione: **05** (l'istruzione successiva a **03**)

(3.d) Quale sequenza di bit contiene la cella di memoria di indirizzo 04 quando la macchina si ferma?

Soluzione: **00**

[Esercizio 4] Quale sequenza di valori viene stampata se si esegue l'istruzione

```
Prima(9);
```

assumendo che siano definite le seguenti procedure **Prima** e **Seconda**? Motivare la risposta.

```
procedure Prima (int N)
     while (N > 0) do (
          Seconda(N);
          N ← N - 5;
          print(N);
     )
```

```
procedure Seconda (int M)
     while (M > 0) do (
          M ← M - 5;
          print(M);
     )
```

Soluzione: viene stampato **4, -1, -1, 4, -1**

[Esercizio 5] La seguente procedura è progettata per cercare un numero (il primo argomento) in un array di interi (il secondo argomento) usando la ricerca lineare; il terzo argomento è la lunghezza dell'array. La procedura dovrebbe restituire TRUE se il numero è presente nell'array, e FALSE altrimenti.
Purtroppo la procedura non è corretta.

```
boolean procedure Ricerca(int N, int [] seq, int K)
     int i ← 1;
     while (i <= K) do (
          if (seq[i] = N) then return TRUE;
          else return FALSE;
          X ← X + 1;
          )
     return FALSE;
```

(5.a) Mostrare che la procedura non è corretta fornendo un array di interi e un numero per i quali la procedura non restituisce il valore atteso

---

Soluzione: Con l'array [4,5] ed N = 5, la procedura restituisce FALSE invece di TRUE

(5.b) Indicare come si può modificare la procedura in modo da correggere l'errore.

Soluzione: Occorre cambiare l'istruzione **`X ← X + 1;`** in **`i ← i + 1;`** e togliere l'istruzione **`else return FALSE;`**

Attenzione: Per l'esercizio (6.b) che segue, si chiede di scrivere una procedura usando lo pseudocodice visto a lezione (usando ***assegnamenti***, ***if-then-else***, ***while-do*** e ***procedure***). Si richiede che ogni variabile utilizzata nella procedura venga prima introdotta in una dichiarazione che ne fissi il tipo e, eventualmente, un valore iniziale. Le sequenze di dati devono essere rappresentate con array. Anche i parametri delle procedure devono essere preceduti dal tipo corrispondente, e la parola chiave **`procedure`** deve essere preceduta dal tipo del risultato, se si attende un risultato.

[Esercizio 6]

(6.a) Descrivere in modo informale, ma più preciso possibile, un algoritmo che risolva il seguente problema:
Data una sequenza di numeri interi a[1], a[2], ..., a[n], si calcoli la differenza tra la somma degli elementi di posizione dispari (cioè, quelli di indice 1, 3, 5, ...) e la somma degli elementi di posizione pari (quelli di indice 2, 4, 6, ...). Per esempio, se la sequenza contiene i numeri [3, 5, 7, 9, 11], l'algoritmo deve restituire il numero **7** (la somma dei numeri di posizione dispari è 3 + 7 + 11 = 21, quella dei numeri in posizione pari è 5 + 9 = 14, e 21 - 14 = 7).

(6.b) Scrivere una procedura **`DiffPariDispari`** che realizza l'algoritmo del punto precedente. La procedura **`DiffPariDispari`** deve avere come parametri un array di interi e la sua lunghezza, e deve restituire un intero come descritto nel punto precedente

---

Una possibile soluzione è la seguente, dove **%** è l'operatore di 'modulo' (resto della divisione intera):

```
int procedure DiffPariDispari(int [] seq, int N)
     int i ← 1;
     int sommaDispari ← 0;
     int sommaPari ← 0;
     while (i <= N) do (
          if (i % 2 = 1)
               then sommaDispari ← sommaDispari + a[i];
               else sommaPari ← sommaPari + a[i];
     )
     int risultato = sommaDispari - sommaPari;
     return risultato;
```
