Scritto di Architettura degli Elaboratori

A.A. 1996/97, 13/6/1997

Cognome :

Nome :

Matricola :

Regole Generali

Questo compito é composto da 2 diversi tipi di domande: alcune a risposte multiple, altre in cui dovete dare voi la risposta corretta mediante lo svolgimento di un semplice esercizio. Nel secondo tipo di domande si richiede di riportare solo il risultato finale.
Le risposte vanno riportate sul MODULO DI RISPOSTA distribuito a parte. Si consiglia comunque di riportare le risposte anche all'interno del testo in modo che il candidato possa tenere una copia dopo la consegna per la verifica delle risposte ed il calcolo del punteggio. Il calcolo dei punteggi relativi ai due tipi di domande é spiegato più avanti.
NON USARE FOGLI ADDIZIONALI né per la consegna del compito né per la "brutta"! E' ammesso utilizzare il rovescio dei fogli del testo come "brutta copia" (che in ogni caso non sarà presa in considerazione per la correzione).
Usare solo penne e NON MATITE per la compilazione delle risposte, in quanto NON VERRANNO ACCETTATE CONTESTAZIONI DI NESSUN TIPO riguardo eventuali correzioni o cancellazioni (comprese quelle effettuate col bianchetto).
Non é ammessa nessuna forma di comunicazione tra i candidati. Non sono ammessi "prestiti" di materiale di nessun tipo tra i candidati.
E' possibile assentarsi dall'aula solo previo deposito del testo del compito presso il docente "di guardia". Anche in caso di ritiro anticipato dal compito il testo dell'esercizio NON PUÒ essere portato fuori dall'aula prima dello scadere del termine di consegna.

Compilazione delle risposte

Si consiglia di scrivere le risposte inizialmente sul testo e di utilizzare gli ultimi 5–10 minuti per ricopiare le risposte sull'apposito foglio da consegnare.

Domande a risposte multiple

Rispondere segnando sul testo con una crocetta le risposte che ritenete corrette e lasciando in bianco quelle che ritenete errate. Ad ogni domanda corrisponde almeno una risposta corretta (ci possono essere più risposte corrette per una domanda).
Domande di questo tipo hanno associati due valori di punteggio: uno negativo e uno positivo. Ogni risposta errata segnata con una crocetta comporta la detrazione del valore negativo dal punteggio complessivo accumulato. Il valore positivo corrisponde al punteggio globalmente assegnato a tutte le risposte corrette ad una domanda.
Per esempio, se una domanda etichettata coi valori $(-1, 4)$ prevede 4 risposte possibili di cui **a.** **d.** errate e **b.** **c.** corrette, allora se il candidato segna le risposte **b.** **c.** e non segna le risposte **a.** **d.** ottiene il massimo punteggio 4, mentre se segna le risposte **a.** **b.** e non **c.** **d.** allora ottiene il punteggio $-1+2=1$; nel caso segnasse tutte le risposte otterrebbe in questo esempio il punteggio $-1+2+2-1=2$. Notare

che in generale, CONTRARIAMENTE a quanto succede nell'esempio, la somma algebrica dei punteggi ottenuti segnando tutte le risposte é ≤ 0!

Esercizi

In questo caso é il candidato che deve calcolare e scrivere la risposta finale. Ogni domanda é accompagnata da un solo valore positivo che indica il punteggio massimo assegnato per la risposta corretta a tutti i quesiti. In caso di risposte parziali o parzialmente corrette il punteggio viene ridotto in proporzionalità diretta rispetto al numero di risposte corrette (con troncamento al più grande numero intero minore o uguale nel caso di risultato frazionario).

Tattica suggerita

Come nel caso di altri test da completarsi in tempi predefiniti, risulta normalmente conveniente completare il compito in due passate: in una prima fase dedicare una quantità predefinita e limitata di tempo alla lettura di tutte le domande, svolgendo solo gli esercizi più semplici. In un secondo passo (dopo aver risposto rapidamente a tutte le domande semplici) tornate a considerare quelle più difficili in ordine decrescente di punteggio. In questo modo riuscirete a dedicare la maggior quantità di tempo alla soluzione degli esercizi difficili ma più redditizi. Per le domande a risposte multiple con punteggi negativi evitate di segnare risposte di cui non siete assolutamente certi in quanto i valori dei punteggi sono studiati per penalizzare i "tentativi a caso". Per gli esercizi senza punteggi negativi invece le risposte sbagliate non vi penalizzeranno.

Calcolo del punteggio

Il voto dello scritto espresso in trentesimi sarà calcolato applicando la seguente formula, nella quale s rappresenta la somma dei punti totalizzati per le varie domande:

$$\text{Voto} = \text{troncamento}\,(3.8 + \min(s, 27) \times 0.6 + (s - 27) \times 0.4)$$

in modo da raggiungere la sufficienza (18/30) totalizzando almeno $s = 25$ punti ed il massimo (30/30) totalizzando almeno $s = 52$ punti (su un massimo teorico di 69). Il voto finale verrà stabilito a partire dal voto dello scritto con un riaggiustamento di tale valore fino ad una variazione massima di 4/30 in più o in meno, a seconda dell'esito della prova orale. Voti superiori a 30 verranno poi considerati equivalenti alla "lode".

Domande

1. (punti: -1, 4)
   Una microarchitettura di tipo microprogrammato:

   (a) si pone come uno degli scopi principali la realizzazione di un insieme di istruzioni molto sofisticate anche partendo da un data path relativamente semplice; per questo normalmente richiede l'uso di una control store di dimensioni relativamente elevate per contenere il microcodice;

   (b) si pone come uno degli scopi principali la minimizzazione dell'occupazione di memoria RAM per il codice dei programmi; per questo adotta tecniche di codifica ad espansione per le microistruzioni in modo da minimizzare il numero di bit utilizzati a parità di numero di parametri;

   (c) si pone come uno degli scopi principali la minimizzazione del numero di accessi in memoria RAM durante l'esecuzione dei programmi; per questo adotta normalmente un insieme di istruzioni di tipo Load/Store e contiene un numero relativamente elevato di registri di uso generale;

   (d) si pone come uno degli scopi principali la massimizzazione del numero di istruzioni eseguite a parità di clock, per questo adotta normalmente frequenze di clock inferiori rispetto a microarchitetture RISC;

   (e) si pone come uno degli scopi principali la massimizzazione della frequenza di clock, per questo adotta normalmente tecniche di caching delle microistruzione per velocizzare l'accesso alla control store;

   (f) nessuna delle affermazioni precedenti é corretta.

2. (punti: -1, 4)
   La tecnica di rilocazione usata da un programma Linker per combinare in un solo programma eseguibile più "moduli oggetto" assemblati separatamente:

   (a) richiede il supporto di un meccanismo di memoria virtuale a segmentazione e paginazione durante l'esecuzione del programma;

   (b) é inutile sul segmento stack nel caso del processore VM-2 per via del meccanismo di indirizzamento indicizzato rispetto al registro FP;

   (c) é indispensabile sul segmento dati nel caso del processore VM-2 a causa della presenza di istruzioni con modo di indirizzamento diretto;

   (d) consente di ritardare fino al momento dell'esecuzione del programma la specifica dell'indirizzo fisico dei dati, facilitando quindi la gestione della memoria da parte del sistema operativo;

   (e) obbliga i programmi assemblatore o compilatore di linguaggi ad alto livello a prevedere a priori quali e quanti moduli potranno essere agganciati al modulo che si sta assemblando o compilando in quel momento;

   (f) nessuna delle affermazioni precedenti é corretta.

3. (punti: -1, 4)
Il codice ASCII per la rappresentazione di caratteri su 7 bit:

   (a) può essere associato ad un bit di parità per definire una rappresentazione ridondante su un byte con capacità di rilevare la presenza di un singolo carattere errato in una stringa;

   (b) é tale per cui rappresentazioni di cifre decimali consencutive corrispondono a valori numerici che differiscono di una unità;

   (c) usa la rappresentazione in modulo e segno per codificare cifre decimali;

   (d) é tale per cui la distanza di Hamming tra le rappresentazioni di un qualsiasi carattere maiuscolo (alfabeto inglese) e del corrispondente carattere minuscolo (es. "A" ed "a") é uguale a 1;

   (e) é tale per cui i 5 bit meno significativi delle rappresentazioni dei caratteri alfabetici dell'alfabeto italiano coincidono con la rappresentazione binaria del numero d'ordine secondo l'ordinamento lessicografico;

   (f) nessuna delle affermazioni precedenti é corretta.

4. (punti: -1, 4)
Il modo di indirizzamento "diretto":

   (a) in generale consente un minor tempo di accesso al dato rispetto al modo "indiretto mediante registro";

   (b) nel caso del processore VM-2 consente un minor tempo di accesso al dato rispetto al modo "indicizzato";

   (c) viene usato per accedere a valori costanti durante l'esecuzione di un programma;

   (d) trova un esempio di realizzazione nella istruzione `STOL 2` della macchina VM-2;

   (e) trova un esempio di realizzazione nella istruzione `LOC8 5` della macchina VM-2;

   (f) trova un esempio di realizzazione nella istruzione SPARC `lduh [%g0+2],%g2`.

5. (punti: -1, 4)
Una memoria cache da 512KB organizzata in 4 insiemi di associatività con parole da 64 bit, linee da 2 parole, indirizzamento della RAM da parte del processore a livello di byte con la convenzione "Big Endians" (come lo SPARC):

   (a) nel caso adotti un campo "tag" di 10 bit permette il "caching" di memoria RAM di dimensione non superiore a 128MB;

   (b) nel caso adotti un campo "tag" di 12 bit permette il "caching" di memoria RAM di dimensione fino a 128M word;

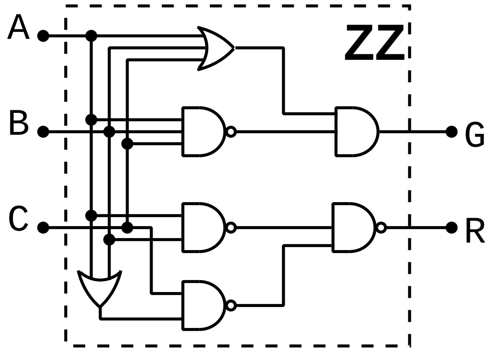
   (c) consente di mappare qualsiasi byte della RAM in 4 posizioni diverse relative alla ripartizione in 32 byte di ogni linea;

   (d) consente di mappare qualsiasi byte della RAM in 64 posizioni diverse nella cache;

   (e) consente di realizzare un livello di multiscalarità di 4, parallelizzando il fetch delle istruzioni consecutive contenute in una stessa linea di cache;

   (f) nessuna delle affermazioni precedenti é corretta.

6. Considerare il seguente modulo circuitale ZZ con 3 ingressi booleani (A, B e C) e 2 uscite (R e G):



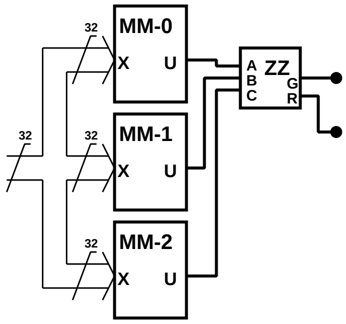
(a) (punti: -1, 1)
Il circuito é di tipo Combinatorio, Sequenziale Asincrono o Sequenziale Sincrono?

(b) (punti: 4) ricavare i valori delle uscite data la seguente sequenza di valori in ingresso:

n.	A	B	C	G	R
1	0	0	0		
2	0	0	1		
3	0	1	0		
4	0	1	1		
5	1	0	0		
6	1	0	1		
7	1	1	0		
8	1	1	1		

(c) (punti: 4)
Disegnare la rappresentazione BDD canonica della funzione G al variare dei valori di ingresso; assumere l'ordinamento delle variabili così come elencato da sinistra verso destra nella tabella al punto (b).

7. Considerare il seguente circuito contenente una istanziazione del modulo ZZ della domanda precedente, oltre a 3 istanziazioni di un ulteriore modulo MM combinatorio a 32 ingressi ed una uscita che calcola una generica funzione booleana $f(X0, ...X31)$:



(a) (punti: -1, 1)
Il circuito é di tipo Combinatorio, Sequenziale Asincrono o Sequenziale Sincrono?

(b) (punti: 2)

Dire quali funzioni vengono calcolate sulle uscite G ed R, nell'ipotesi che i 3 moduli MM realizzino correttamente la funzione f.

(c) (punti: 2)

Spiegare quale scopo si prefigge il circuito in figura.

(d) (punti: 9)

completare la seguente specifica in linguaggio VHDL per definire un circuito analogo a quello in figura ma basato su moduli NN che realizzano una funzione combinatorie a 32 ingressi e 2 uscite:

```
entity my_entity is
port(I: in bit_vector(0 to 31); O: out bit_vector(0 to 1); E: out bit);
end my_entity;

library ttl;
use ttl.or2;

entity NN_entity is
port(X: in bit_vector(0 to 31); U: out bit_vector(0 to 1));
end NN_entity;

entity ZZ_entity is
port ...
end ZZ_entity;

architecture structure of my_entity is

        component or2_gate
         port(a,b: in bit; c:out bit);
        end component;

        for all:or2_gate use entity ttl.or2;

signal sa0,sb0,sc0,sg0: bit;
signal ...

begin

nn0: NN_entity port map ...

nn1: ...

nn2: ...
```

```
zz0: ZZ_entity port map ...

zz1: ...

org: ...

end structure;
```

8. Considerare la seguente procedura "ppp" definita in Assembler SPARC:

```
.section        ".text"
        .align 4
        .global ppp
ppp:    save %sp,-112,%sp
        mov 0,%l1
        add %i1,4,%l2
.LL9:   ldub [%i0],%l3
        call vvv
MANCA1:
        mov %o0,%l0
        ldub [%i1],%l4
        call vvv
        mov %l4,%o0
        cmp %l0,%o0
        bne,a .LL12
        add %i1,1,%i1
        sub %l3,%l4,%l5
        add %l1,%l5,%l1
MANCA2:
.LL12:  cmp %i1,%l2
        ble .LL9
        add %i0,1,%i0
        sethi %hi(.LLC0),%o0
        or %o0,%lo(.LLC0),%o0
        call printf
        mov %l1,%o1
MANCA3:
        restore
.LLfe1: .size   ppp,.LLfe1-ppp
        .global vvv
vvv:    cmp %o0,97
        be .LL3
        cmp %o0,101
        be .LL3
        cmp %o0,105
        be .LL3
        cmp %o0,111
        be .LL3
        cmp %o0,117
        bne .LL2
```

```
        mov 0,%o0
.LL3:   mov 1,%o0
.LL2:   retl
        nop
.LLfe2: .size    vvv,.LLfe2-vvv
```

`ppp` viene chiamata da un programma principale previa inserzione in `%o0` dell'indirizzo del primo elemento di un vettore di 4 byte, contenente la codifica ASCII delle prime 4 lettere del vostro nome ed in `%o1` dell'indirizzo del primo elemento di un vettore di 4 byte, contenente la codifica ASCII delle prime 4 lettere del vostro cognome (senza spazi, caratteri minuscoli). Nel caso il vostro nome e/o cognome fosse composto da meno di 4 caratteri riempire i byte inutilizzati col valore 0.

(a) (punti: 9)
Completare il programma con le tre istruzioni mancanti, individuate dalle etichette `MANCA1`, `MANCA2` e `MANCA3`.

(b) (punti: 8)
Sapendo che il "formato di stampa" identificato dall'etichetta `.LLC0` definisce la stampa di un numero intero, scrivere cosa verrebbe stampato dall'esecuzione della procedura con i vostri dati.
Suggerimento: si ricorda che la codifica ASCII del carattere "a" vale 97, e che il codice segue l'ordine lessicografico dell'alfabeto inglese.

9. (punti: 9)
Completare la seguente traduzione della procedura `ppp` della domanda precedente in assembler VM-2 con le 3 istruzioni mancanti alle etichette `X1`, `X2` e `X3`:

```
ppp:   LOC8 0
X1:
       LOC8 4
       ADDL 2
X2:
       PUSH
       LODL 2
       PUSH
       LODL 1
       PUSH
su:    PSHI
       CALL vvv
       PUSH
       LODL -4
       PSHI
       CALL vvv
       SIGN
       ADDL -7
       JNZE giu
       LODL -6
       SIGN
       ADDL -8
       ADDL -2
       STOL -2
```

```
giu:   LOC8 1
       ADDL -4
       STOL -4
       ADDL -3
       JPOS via
X3:
       LOC8 1
       ADDL -5
       STOL -5
       JUMP su
via:   INSP 6
       LOC8 low(LLC0)
       LOCH high(LLC0)
       PUSH
       CALL printf
       RETN
```

I parametro in ingresso a `ppp` vengono in questo caso passati sullo stack, così come quelli alla funzione `vvv` ed alla procedura `printf`. Per convenzione la funzione `vvv` ritorna in questo caso il valore nel registro accumulatore.

Risposte Scritto di Architettura degli Elaboratori, 13/6/1997

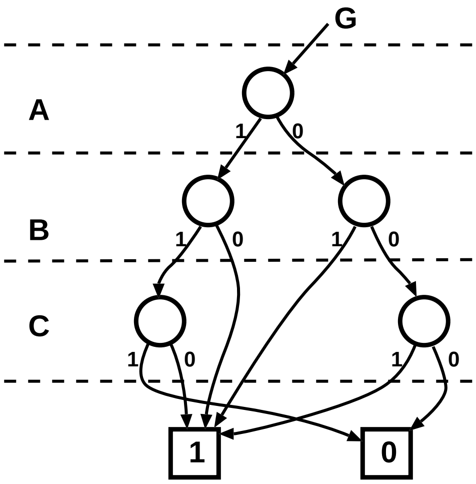
Cognome : **Nome:** **Matricola:**

1. (punti: -1, 4) **X a** **b** **c** **d** **e** **f**
2. (punti: -1, 4) **a** **X b** **X c** **d** **e** **f**
3. (punti: -1, 4) **a** **X b** **c** **X d** **e** **f**
4. (punti: -1, 4) **a** **X b** **c** **d** **e** **X f**
5. (punti: -1, 4) **X a** **b** **c** **d** **e** **f**
6.
   (a) (punti: -1, 1) X Combinatorio Asincrono Sincrono

   (b) (punti: 4)

n.	A	B	C	G	R
1	0	0	0	0	0
2	0	0	1	1	0
3	0	1	0	1	0
4	0	1	1	1	1
5	1	0	0	1	0
6	1	0	1	1	1
7	1	1	0	1	1
8	1	1	1	0	1

   (c) (punti: 4)



7.
   (a) (punti: -1, 1) X Combinatorio Asincrono Sincrono

   (b) (punti: 2) $G = 0$ $R = f(X)$

   (c) (punti: 2)
   consenso "2 su 3" per tolleranza ai guasti del modulo MM: continua a funzionare correttamente anche quando una delle 3 repliche di MM si guasta; G=1 indica la condizione di guasto, nella quale le 3 repliche non concordano sul risultato.

(d) (punti: 9)

```
port(A: in bit; B: in bit; C: in bit; G: out bit; R: out bit);
...
signal sa1,sb1,sc1,sg1: bit;
...
nn0: NN_entity port map (X=>I,U(0)=>sa0,U(1)=>sa1);
nn1: NN_entity port map (X=>I,U(0)=>sb0,U(1)=>sb1);
nn2: NN_entity port map (X=>I,U(0)=>sc0,U(1)=>sc1);
zz0: ZZ_entity port map (A=>sa0,B=>sb0,C=>sc0,G=>sg0,R=>O(0));
zz1: ZZ_entity port map (A=>sa1,B=>sb1,C=>sc1,G=>sg1,R=>O(1));
org: or2_gate port map (a=>sg0,b=>sg1,c=>E);
```

8.

(a) (punti: 9)

```
        call vvv
MANCA1: mov %l3,%o0
        mov %o0,%l0
...
        add %l1,%l5,%l1
MANCA2: add %i1,1,%i1
.LL12:  cmp %i1,%l2
...
        mov %l1,%o1
MANCA3: ret
        restore
```

(b) (punti: 8)
la somma delle differenze lessicografiche tra vocali o consonanti in posizioni corrispondenti (vocali corrispondenti a consonanti e viceversa vengono saltate).
Esempio: "giov" e "chio" producono $(g - c = 4) + (o - i = 6) = 10$

9. (punti: 9)

```
ppp:   LOC8 0
X1:    PUSH
       LOC8 4
       ADDL 2
X2:    SIGN
       PUSH
       ...
       JPOS via
X3:    INSP 3
       LOC8 1
```