Scritto di Architettura prof. Chiola

21/6/1996

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:

$$\text{Voto} = \text{troncamento}(c1 + \min(s, c2) \times c3 + (s - c2) \times c4)$$

I coefficienti $c1$, $c2$, $c3$ e $c4$ sono valori costanti che vengono determinati tenendo conto della statistica dei valori s ottenuti dai candidati che hanno consegnato compiti precedenti in modo da ottenere una distribuzione equa dei voti tra 18/30 e 30/30 per coloro che hanno risposto correttamente ad una porzione sufficiente di domande. I valori delle costanti per questo compito sono:

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

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 di 67). 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 memoria Cache di primo livello:

   (a) deve essere organizzata in modo completamente associativo per ottenere la massima efficienza e la minima dimensione;

   (b) deve essere organizzata a corrispondenza diretta per semplificare l'algoritmo di ricerca e quindi aumentare la velocità di operazione;

   (c) deve usare la tecnica "write-through" per evitare la necessità di algoritmi di tipo "snooping cache" per mantenere la consistenza in sistemi multiprocessore;

   (d) deve usare la tecnica "write-back" per evitare di rallentare il completamento di istruzioni di scrittura da parte del processore rispetto alle operazioni di lettura;

   (e) può essere usata solo in associazione con un meccanismo di memoria virtuale per il segmento codice, sul quale il processore effettua solo operazioni di lettura (nella fase di fetch);

   (f) nessuna delle affermazioni precedenti é corretta.

2. (punti: -1, 4)
   Una codifica binaria in "complemento a due" per una architettura a 64 bit (come per esempio "UltraSPARC"):

   (a) richiede che ogni numero venga sempre rappresentato in memoria utilizzando una parola da 64 bit;

   (b) può consentire la definizione di formati diversi di rappresentazione in memoria (per esempio su 8, 16, 32 e 64 bit), a patto che vengano definite delle istruzioni di conversione da rappresentazioni con meno bit a rappresentazioni con più bit (estensione del segno);

   (c) può consentire la definizione di formati diversi di rappresentazione in memoria (per esempio su 8, 16, 32 e 64 bit), a patto che vengano definite delle istruzioni di conversione da rappresentazioni con più bit a rappresentazioni con meno bit (compressione mediante codice ad espansione);

   (d) richiede l'uso di almeno 12 bit per la rappresentazione del numero +2049;

   (e) non consente la rappresentazione del numero esadecimale $+8F41279A$;

   (f) nessuna delle affermazioni precedenti é corretta.

3. (punti: -1, 4)
   Il meccanismo di segnalazione fisica delle interruzioni:

   (a) permette ad un dispositivo connesso al bus di "chiedere" l'attenzione della CPU in modo asincrono; la CPU può rispondere alla richiesta interrompendo l'esecuzione del programma in corso ed eseguendo una procedura predefinita per la gestione delle interruzioni;

   (b) permette ad un dispositivo connesso al bus di ottenere l'attenzione della CPU in modo asincrono; la CPU rispondere interrompendo l'esecuzione del programma in corso ed eseguendo una procedura definita dal dispositivo stesso;

   (c) permette alla CPU di ottenere in modo asincrono informazioni sullo stato del dispositivo connesso al bus; quando il dispositivo interrompe la propria attività, la CPU rileva la condizione codificata sul filo di interruzione e può quindi iniziare l'esecuzione della procedura di gestione delle interruzioni;

   (d) permette al dispositivo di riconoscere la condizione di richiesta di interruzione dell'attività comandata dalla CPU mediante l'esecuzione della procedura di gestione delle interruzioni;

(e) permette la propagazione delle richieste di interruzione dai dispositivi meno prioritari ai dispositivi più prioritari mediante la connessione “daisy chain”;

(f) nessuna delle affermazioni precedenti é corretta.

4. (punti: -1, 4)
La gestione di tipo “sliding window” del “register file” in una CPU di tipo RISC:

(a) permette di inserire molti registri nel register file rispetto al numero di bit usati per la codifica degli operandi di tipo registro nel formato delle istruzioni;

(b) permette di eliminare l’uso dello stack per la memorizzazione dei valori delle variabili locali ad una procedura;

(c) permette di supportare a livello hardware l’attivazione di contesti locali con salvataggio e ripristo di valori come in una struttura di tipo “stack”, senza necessariamente far riferimento alla memoria RAM;

(d) consiste nello strutturare i registri interni della CPU come una memoria RAM con modo di indirizzamento diretto;

(e) consiste nello strutturare i registri interni della CPU come una memoria RAM con modo di indirizzamento autoincremento (o autodecremento);

(f) nessuna delle affermazioni precedenti é corretta.

5. (punti: -1, 4)
Il meccanismo di gestione delle Trap:

(a) consente di individuare errori di programma in fase di esecuzione attivando delle procedure di test opportunamente definite come gestori di trap ed interrompendo l’esecuzione del programma quando tali procedure di test falliscono;

(b) consente di attivare un meccanismo di allocazione dinamica di memoria virtuale, utilizzando il gestore di trap della condizione “page fault” per mappare una nuova pagina fisica nel segmento che ha generato l’eccezione;

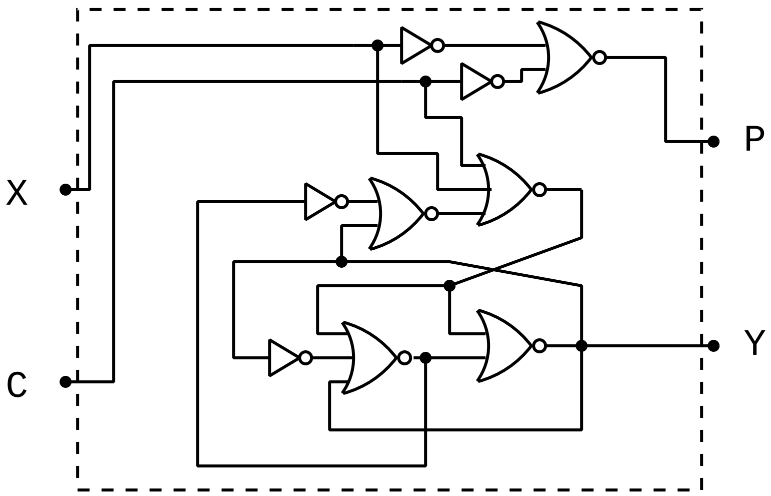
(c) consente ad un dispositivo attivato in DMA di segnalare alla CPU la fine del trasferimento di dati precedentemente attivato;

(d) consente alla CPU di riguadagnare un alto livello di privilegio per l’esecuzione di funzioni di gestione del sistema a seguito di una richiesta generata dall’esecuzione di un programma utente che non ha il diritto di manipolare direttamente quelle risorse;

(e) consente al programma “linker” di portare a termine le operazioni di rilocazione del codice in modo dinamico durante l’esecuzione del programma utente solo nel momento in cui il codice viene effettivamente eseguito (secondo una strategia di valutazione “lazy”);

(f) nessuna delle affermazioni precedenti é corretta.

6. Considerare il seguente modulo circuitale con 2 ingressi booleani (C e X) e 2 uscite booleane (Y e P):



(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:

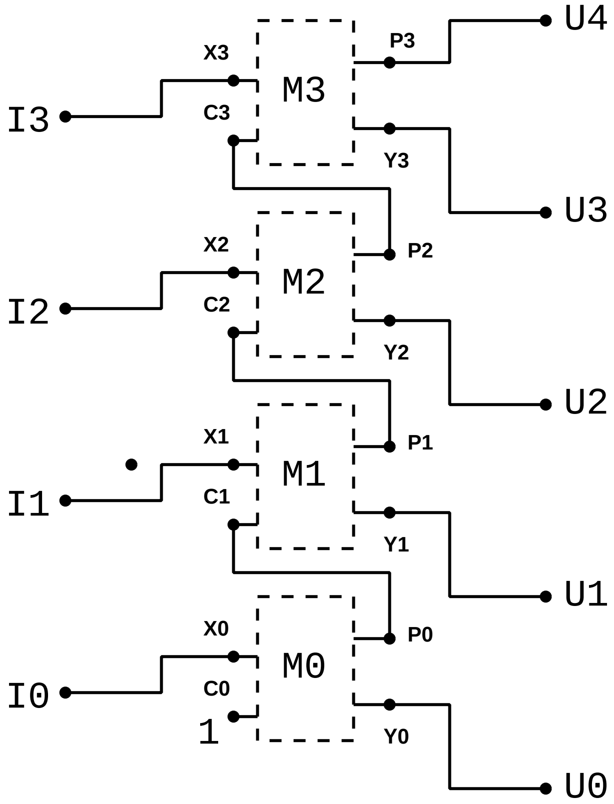
n.	C	X	Y	P
1	0	0		
2	0	1		
3	1	0		
4	1	1		
5	0	1		
6	1	1		
7	1	0		
8	0	0		

(c) (punti: 3)
Disegnare la rappresentazione BDD canonica delle funzioni Y e P di uscita, assumendo l'ordinamento delle variabili: prima C e poi X.

(d) (punti: 3)
Ridurre il circuito togliendo le componenti ridondanti, in modo da usare il minor numero possibile di funzioni elementari AND, OR, NOT.

(e) (punti: 4)

Considerare la seguente interconnessione di 4 moduli circuitali, ed interpretando gli insiemi di variabili booleane $I3, I2, I1, I0$ e $U4, U3, U2, U1, U0$ come rappresentazioni binarie di numeri naturali, dire quale funzione viene calcolata:



(f) (punti: 7)

completare la seguente descrizione in linguaggio VHDL del circuito a 4 ingressi e 5 uscite definito al punto (e):

```
entity M is
 port (C,X: in bit; Y,P: out bit);
end ha;
library lib; use lib.all;
architecture M_st of M is
component andg
    port (in1, in2 : bit; out1 : out bit);
end component;
        for all:andg use entity lib.andg;
component org
    port (in1, in2 : bit; out1 : out bit);
        for all:org use entity lib.org;
end component;
begin
  ...





end M_st;
```

```
entity my_entity is
port(I: in bit_vector(0 to 3);U: out bit_vector(0 to 3));
end my_entity;
architecture structure of my_entity is
        component M
        port (C,X: in bit; Y,P: out bit);
        end component;
        for all:M use entity work.M;
        signal ...

begin

   ...

end structure;
```

7. (punti: 10)
   Considerare le seguenti due procedure "p1" e "p2" definite in Assembler SPARC:

```
p1:   ldub [%o0],%g2
      ldub [%o1],%g3
      stb %g2,[%o1]
      stb %g3,[%o0]
      retl
      nop


p2:   save %sp,-104,%sp
      ldub [%i0],%o0
      cmp %o0,0
      be L5
      mov %i0,%l0
      b L2
      mov %i0,%l1
L0:   ldub [%l0],%o2
      cmp %o2,0
      be L5
      mov %i0,%l1
L1:   ldub [%l1],%o3
      cmp %o3,0
      be,a L0
      add %l0,1,%l0
```

```
L2:   sub %l0,%i0,%l2
      cmp %o2,%o3
      bleu L3
      sub %l1,%i0,%o1
      ldub [%i1+%l2],%o0
      ldub [%i1+%o1],%o1
      cmp %o0,%o1
      blu L4
      mov %l0,%o0
      cmp %o2,%o3
L3:   bgeu,a L1
      add %l1,1,%l1
      ldub [%i1+%l2],%o0
      sub %l1,%i0,%o1
      ldub [%i1+%o1],%o1
      cmp %o0,%o1
      bleu,a L1
      add %l1,1,%l1
      mov %l0,%o0
L4:   call p1
      mov %l1,%o1
      b L1
      add %l1,1,%l1
L5:   call print
      mov %i0,%o1
      ret
      restore
```

`p2` viene chiamata da un programma principale il quale inserisce nei registri `%o0` e `%o1` gli indirizzi di due vettori di 4 byte ciascuno; i valori contenuti nel primo vettore saranno le codifiche binarie delle prime tre cifre del vostro numero di matricola, ciascuna incrementata di 1, seguite dal valore 0; i valori contenuti nel secondo vettore saranno le codifiche binarie delle ultime tre cifre del vostro numero di matricola, ciascuna incrementata di 1, seguite dal valore 0; per esempio, supponendo che il vostro numero di matricola sia 1359642, il vettore indirizzato da `%o0` conterrà la codifica binaria dei valori 2, 4, 6, 0, mentre il vettore indirizzato da `%o1` conterrà la codifica binaria dei valori 7, 5, 3, 0. Indicare i valori contenuti nei due vettori prima e dopo l'esecuzione di `p2`.

8. (punti: 15)
Completare la seguente traduzione della procedura `p2` della domanda precedente in assembler VM-2 con le 5 istruzioni mancanti (ovviamente passando i parametri in ingresso a `p2` sullo stack):

```
p2:   LOC8 0
      PUSH
      LODL 2
l0:   PSHI
      POP
```

```
      JNZE l1
      RETN
l1:   SIGN
      PUSH
      LOC8 0
      PUSH
      LODL 1
l2:   PSHI
      POP
      JZER l6
      ADDL -3
      JNEG l4
      JZER l5
      LODL 1
      PUSH
      LODL -2
      PUSH
      LODL -4
      PUSH
      CALL test
X1:
      JNEG l5
l3:   LODL -2
      ADDL 2
      PUSH
      LODL -4
      ADDL 2
      PUSH
      CALL p1
      INSP 2
      JUMP l5
l4:   LODL 1
      PUSH
      LODL -4
      PUSH
X2:
      PUSH
      CALL test
      INSP 3
X3:
l5:   LOC8 1
      ADDL -4
      STOL -4
      ADDL 1
      JUMP l2
l6:   LOC8 1
      ADDL -2
      STOL -2
      ADDL 2
X4:
```

```
      JUMP 10
test: LODL 2
      ADDL 3
      PSHI
      POP
      SIGN
      PUSH
      LODL 1
X5:
      PSHI
      POP
      ADDL -2
      RETN
```

La funzione `test` restituisce il valore calcolato nel registro accumulatore.

Risposte Scritto di Architettura, 21/6/1996

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

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

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

(b) (punti: 4)

n.	C	X	Y	P
1	0	0	0	0
2	0	1	1	0
3	1	0	1	0
4	1	1	1	1
5	0	1	1	0
6	1	1	1	1
7	1	0	1	0
8	0	0	0	0

(c) (punti: 3)

Disegnare la rappresentazione BDD canonica delle funzioni Y e P di uscita, assumendo l'ordinamento delle variabili: prima C e poi X.

(d) (punti: 3)

Ridurre il circuito togliendo le componenti ridondanti, in modo da usare il minor numero possibile di funzioni elementari AND, OR, NOT.

(e) (punti: 4)

minimo numero strettamente maggiore, e tale che distanza di Hamming delle rappresentazioni é uguale ad 1.

(f) (punti: 7):

```
begin
 M_or: org
  port map(in1=>C,in2=>X,out1=>Y);
 M_and: andg
  port map(in1=>C,in2=>X,out1=>P);
end M_st;

 signal P_bus:bit_vector(0 to 2);
begin
 M0: M
  port map(C=>'1',X=>I(0),Y=>U(0),P=>P_bus(0));
 M1: M
  port map(C=>P_bus(0),X=>I(1),Y=>U(1),P=>P_bus(1));
 M2: M
  port map(C=>P_bus(1),X=>I(2),Y=>U(2),P=>P_bus(2));
 M3: M
  port map(C=>P_bus(2),X=>I(3),Y=>U(3),P=>U(4));
end structure;
```

7. (punti: 10) Ordina il primo vettore in modo che ogni coppia di elementi verifichi le stesse relazioni d'ordine della corrispondente coppia del secondo vettore; il secondo vettore rimane inalterato. Esempio: matricola=1359642, v1 = 2,4,6,0, v2 = 7,5,3,0; dopo l'esecuzione si ottiene v1 = 6,4,2,0, v2 = 7,5,3,0.

8. (punti: 15)

```
        ...
      CALL test
X1:   INSP 3
      JNEG 15
        ...
      PUSH
X2:   LODL -2
      PUSH
      CALL test
      INSP 3
X3:   JPOS 13
15:   LOC8 1
        ...
      ADDL 2
X4:   INSP 2
      JUMP 10
        ...
      LODL 1
X5:   ADDL 3
      PSHI
        ...
```

Risposte Scritto di Architettura, 21/6/1996

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

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

(a) (punti: -1, 1) Combinatorio Asincrono Sincrono

(b) (punti: 4)

n.	C	X	Y	P
1	0	0		
2	0	1		
3	1	0		
4	1	1		
5	0	1		
6	1	1		
7	1	0		
8	0	0		

(c) (punti: 3)

(d) (punti: 3)

(e) (punti: 4)

(f) (punti: 7)

```
architecture M_st of M is etc.
begin
  ...





end M_st;

entity my_entity is etc.
        signal ...

begin

   ...









end structure;
```

7. (punti: 10) matricola= , v1[0] = , v1[1] = , v1[2] = , v2[0] = , v2[1] = , v2[2] =
   dopo l'esecuzione di `p2`: v1[0] = , v1[1] = , v1[2] = , v2[0] = , v2[1] = , v2[2] =

8. (punti: 15)

```
      CALL test
X1:
      JNEG l5
       ...
      PUSH
X2:
      PUSH
      CALL test
      INSP 3
X3:
l5:   LOC8 1
       ...
      ADDL 2
X4:
      JUMP l0
       ...
      LODL 1
X5:
      PSHI
```