

PROVA SCRITTA DEL CORSO DI
CALCOLATORI ELETTRONICI
NUOVO ORDINAMENTO DIDATTICO
11 Settembre 2008

NOME:

COGNOME:

MATRICOLA:

ESERCIZIO 1 (8 punti)

Si consideri la rete combinatoria caratterizzata da tre ingressi A, B, C e da due uscite le cui funzioni sono:

$$Y_1 = \overline{A}\overline{B}C + \overline{A}B\overline{C} + A\overline{B}\overline{C} + ABC$$

$$Y_2 = \overline{A}BC + A\overline{B}C + AB\overline{C} + ABC$$

- 1) (2 punti) Scrivere la tabella di verità di Y_1 e Y_2 .
- 2) (2 punti) Calcolare le forme minime per mezzo delle mappe di Karnaugh.
- 3) (2 punti) Riscrivere le equazioni delle reti logiche risultanti dal punto 2 usando soltanto porte NAND.
- 4) (2 punti) Progettare, con le mappe di Karnaugh, una terza uscita Y_3 che, in funzione di A, B, C realizzi la funzione:

$$Y_3 = Y_1 + Y_2.$$

ESERCIZIO 2 (8 punti)

Si consideri un calcolatore che dispone di una memoria principale di 256 Mbyte e di una memoria cache di 512 Kbyte. E' possibile accedere al singolo byte e la memoria è suddivisa in blocchi da 16 byte.

1) Spiegare, precisando il significato e la funzione dei diversi campi, come vengono interpretati gli indirizzi logici per recuperare l'informazione contenuta nella cache nel caso in cui venga usata la modalità di indirizzamento:

(a) (2 punti) diretto;

(b) (2 punti) "associativo su insiemi", e ciascun insieme contenga due blocchi.

2) (4 punti) Si consideri la cache di cui alla domanda precedente. Ipotizzare che il processore acceda in sequenza ai byte dall'indirizzo 00000000 a 00003FFF e da 03800000 a 03803FFF (in questo ordine) e ripeta la sequenza di accesso data per 10 volte consecutive. Si ipotizzi inoltre che la cache sia inizialmente vuota. Calcolare l'hit rate considerando la cache descritta al punto precedente, sia nel caso di indirizzamento diretto sia nel caso di indirizzamento "associativo su insiemi".

ESERCIZIO 3 (9 punti)

Scrivere una funzione Assembly MIPS che ricevendo in ingresso un vettore di interi v di dimensione N e un intero t , sposti all'inizio del vettore tutti i valori di v minori di t .

Si supponga che $\&v[0]$ sia memorizzato nella locazione 1000, $\$5 \leftarrow t$, $\$6 \leftarrow N$.

Ad esempio, se $v = \{11, 10, 20, 9, 5\}$ e $t = 10$, allora, dopo l'esecuzione della funzione, si avrà $v = \{9, 5, 20, 11, 10\}$.

In altri termini, la funzione MIPS potrebbe implementare la seguente funzione C:

```
void verifica(int *v, int t, int N) {
    int i, j, temp;

    j=0;
    for(i=0; i<N; i++)
        if(v[i]<t) {
            temp=v[j];
            v[j]=v[i];
            v[i]=temp;
            j++;
        }
}
```

ESERCIZIO 4 (8 punti)

(a) (6 punti) Il sistema di memoria virtuale di un calcolatore viene gestito mediante la tecnica di "paginazione su richiesta". Il sistema operativo indirizza 8 pagine virtuali, mentre la memoria del calcolatore contiene 4 pagine fisiche. La dimensione di ciascuna pagina è uguale a 1024 parole. Ad un certo istante si abbia il contenuto della PMT riportata a lato.

Elencare tutti gli indirizzi virtuali, espressi in decimale, che provocano un "page fault".

Indirizzo di pagina virtuale	Indirizzo di pagina fisica
0	2
1	-
2	-
3	-
4	3
5	-
6	1
7	0

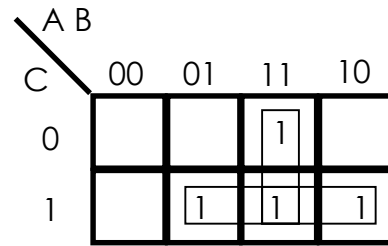
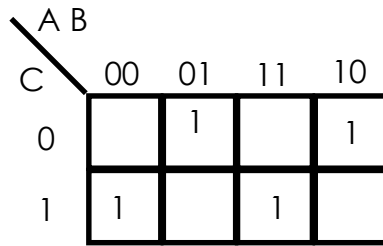
(b) (2 punti) Spiegare in modo chiaro e sintetico in cosa consiste il problema del "trashing" nella paginazione su richiesta, e indicare i principali algoritmi di "trashing" utilizzati.

ESERCIZIO 1

Soluzione.

1) Tabella di verità e mappe di Karnaugh.

Tabella di verità					
A	B	C	Y ₁	Y ₂	
0	0	0	0	0	
0	0	1	1	0	
0	1	0	1	0	
0	1	1	0	1	
1	0	0	1	0	
1	0	1	0	1	
1	1	0	0	1	
1	1	1	1	1	



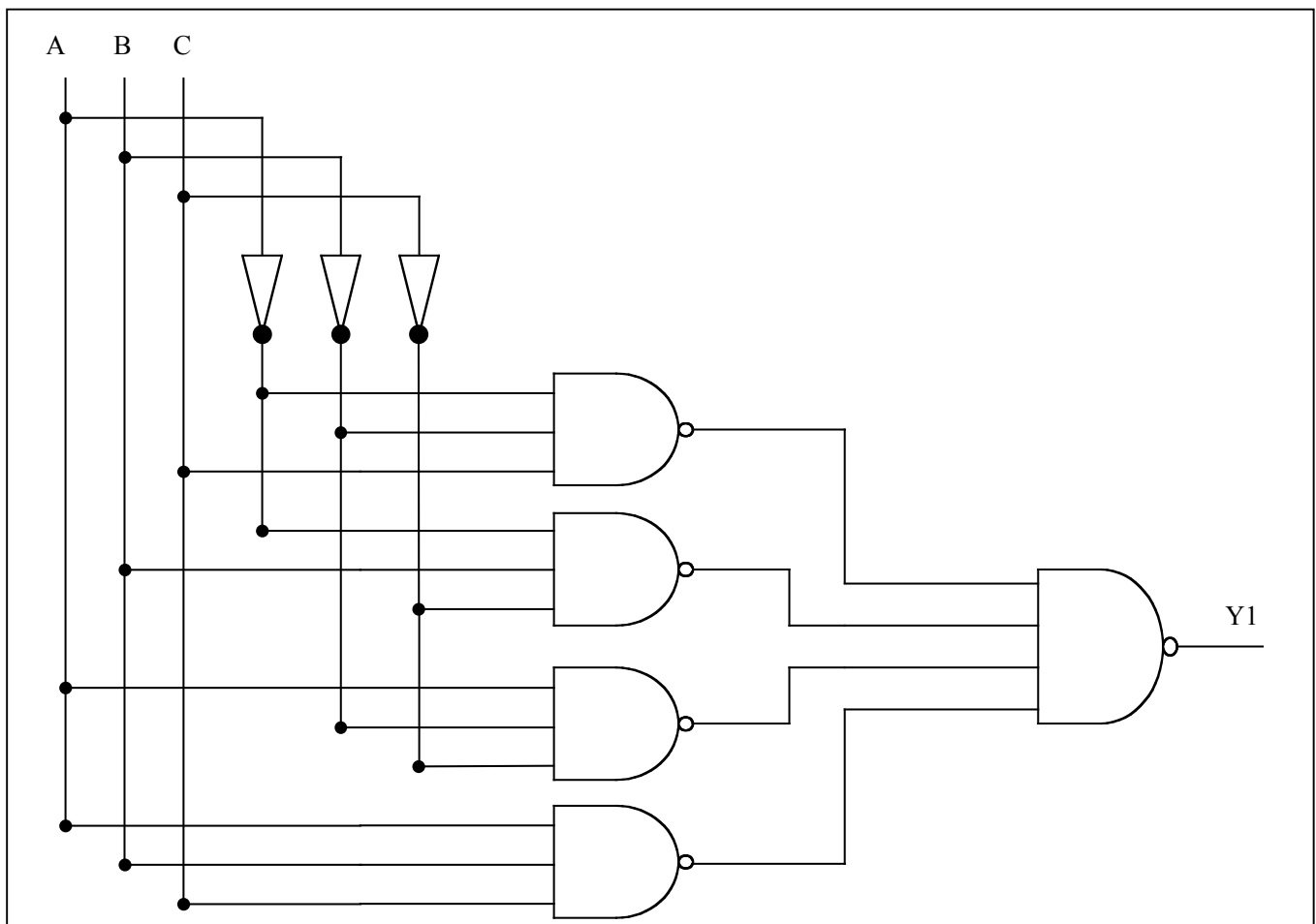
Y₁ è già espressa nella forma minima, mentre $Y_2 = AB + AC + BC$.

2) Rete logica con sole porte NAND.

Per brevità riportiamo la rete logica relativa a Y₁. La rete logica di Y₂ presenta la stessa struttura (cambiano le connessioni di ingresso e il numero di ingressi supportati da ciascuna NAND del primo livello di logica).

$$Y_1 = \overline{\overline{A}\overline{B}\overline{C}} \cdot \overline{\overline{A}\overline{B}C} \cdot \overline{\overline{A}B\overline{C}} \cdot \overline{\overline{A}BC} = (\overline{A} \uparrow \overline{B} \uparrow \overline{C}) \uparrow (\overline{A} \uparrow B \uparrow \overline{C}) \uparrow (\overline{A} \uparrow B \uparrow C) \uparrow (\overline{A} \uparrow \overline{B} \uparrow C)$$

Riportiamo (per completezza / non era richiesto) il disegno di questa rete:



3) OR di $Y1$ e $Y2$ a partire da A, B, C con le mappe di Karnaugh.

Dal momento che viene richiesto un OR, è sufficiente sovrapporre le due mappe precedenti ottenendo:

A B					
C		00	01	11	10
	0		1	1	1
	1	1	1	1	1

Il risultato è dunque: $Y3 = A + B + C$.

ESERCIZIO 2

Soluzione

1) Per indirizzare 256 Mbyte occorre un indirizzo di 28 bit. Per indirizzare il singolo byte all'interno di un blocco occorrono 4 bit ($16 = 2^4$), che coincidono con i 4 bit meno significativi dell'indirizzo di memoria primaria. I restanti 24 bit costituiscono l'indirizzo del "block frame". Per indirizzare la cache, il "block frame" viene interpretato diversamente a seconda che l'indirizzamento sia di tipo "diretto" o "associativo su insiemi".

(a) Indirizzamento diretto. In questo caso devo poter indirizzare ciascuno dei 32K blocchi contenuti nella cache ($512\text{Kbyte}/(16\text{byte}/\text{blocco})$). Occorrono 15 bit che coincidono con i 15 bit meno significativi del "block frame". Pertanto i 28 bit di indirizzo della memoria primaria vengono interpretati come:

tag	cache index	offset
9 bit	15 bit	4 bit

(b) Indirizzamento "associativo su insiemi". In questo caso devo poter indirizzare ciascuno dei 16 insiemi in cui sono suddivisi i blocchi contenuti nella cache. Occorrono 14 bit che coincidono con i 14 bit meno significativi del "block frame". Pertanto i 28 bit di indirizzo della memoria primaria vengono interpretati come:

tag	cache index	offset
10 bit	14 bit	4 bit

2) La memoria è divisa in blocchi di 16 byte ciascuno in modo che la richiesta di un dato non presente in cache causa il trasferimento del blocco a cui appartiene il dato richiesto dalla memoria principale alla cache. Nel caso proposto i dati richiesti sono così suddivisi (N.B. per semplicità i valori di index sono riportati in formato decimale, mentre gli indirizzi in esadecimale):

indirizzi 00000000 ÷ 0000000F ⇒ index 0; indirizzi 00000010 ÷ 0000001F ⇒ index 1; ... ; 00003F0 ÷ 00003FF ⇒ index 63; 03800000 a 0380000F ⇒ index 0; ... ; 03803F0 a 03803FF ⇒ index 63.

(N.B. in questa sequenza l'index è identico nel caso di indirizzamento diretto e associativo su insiemi).

La prima volta che viene richiesta la parola 00000000 avremo un "cache miss" che provoca il caricamento del blocco 0 nella linea di cache di indirizzo 0. Le successive richieste dei dati di indirizzo 00000001 ÷ 0000000F vengono quindi soddisfatte dalla cache ("cache hit"). Analogamente avviene per tutti i blocchi fino al blocco 63, che vengono allocati in linee consecutive della cache fino alla 63. Quando viene richiesta la parola 03800000 (index 0), nel caso di indirizzamento diretto questa sovrascrive il blocco 00000000 ÷ 0000000F e così via per tutte le richieste consecutive.

Nel caso di indirizzamento associativo su insiemi a due vie uno stesso index corrisponde a un insieme di due blocchi. Pertanto le parole da 03800000 a 03803FF non sovrascrivono quelle precedentemente inserite, ma vengono memorizzate nel secondo blocco disponibile in ciascun insieme memorizzato. Nei cicli successivi al primo il processore richiede nuovamente tutti i dati, a partire da 0. Nel caso di indirizzamento diretto avremo un miss per il caricamento della prima parola di ciascun blocco e 15 hit per le parole dello stesso blocco, per tutti i 128 blocchi. Nel caso di indirizzamento associativo su insiemi si hanno solo hit perché tutti i blocchi sono presenti in cache. In sintesi:

Nel caso di indirizzamento diretto avremo per ciascun ciclo 128 miss e $15 \cdot 128$ hit. L'hit ratio H della gerarchia di memoria risulta pertanto pari a:

$$H = \text{cachehit} / \text{richieste totali} = (10 \cdot 15 \cdot 128) / (10 \cdot 16 \cdot 128) = 0.9375$$

Nel caso di indirizzamento associativo su insiemi a due vie avremo nel primo ciclo 128 miss e 15*128 hit, mentre nei 9 cicli successivi avremo 16*128 hit. L'hit ratio H della gerarchia di memoria risulta pertanto pari a:

$$H = \text{cachehit} / \text{richieste totali} = (15 \cdot 128 + 9 \cdot 16 \cdot 128) / 10 \cdot 16 \cdot 128 = 0.99375$$

ESERCIZIO 3

Soluzione.

Assumiamo che:

$i \rightarrow \$8, j \rightarrow \$9, \text{temp} \rightarrow \$10, (v[i] < t) \rightarrow \$11, v[i] \rightarrow \$12$

verifica:

```
addi $29, $29, -20    #salvataggio del contesto
sw $8, 0($29)
sw $9, 4($29)
sw $10, 8($29)
sw $11, 12($29)
sw $12, 16($29)
```

```
mul $6, $6, 4
move $9, $0           #j=0
move $8, $0           #i=0
```

for:

```
beq $8, $6, exit      #if(i==N) exit
lw $12, 1000($8)      #$12 ← v[i]
slt $11, $12, $5      #$11 ← v[i]<t
beq $11, $0, update_i #if(!(v[i]<t)) update_i
lw $10, 1000($9)      #$10 ← v[j], ovvero temp=v[j]
sw $12, 1000($9)      #v[j]=v[i]
sw $10, 1000($8)      #v[i]=temp
addi $9, $9, 4        #j++
```

update_i:

```
addi $8, $8, 4
j for
```

exit:

```
lw $8, 0($29)         #ripristino del contesto
lw $9, 4($29)
lw $10, 8($29)
lw $11, 12($29)
lw $12, 16($29)
addi $29, $29, 20
jr $31                #ritorno al chiamante
```

ESERCIZIO 4

Soluzione

- (a) Gli indirizzi virtuali che generano un "page fault" sono quelli che corrispondono a pagine virtuali non caricate in memoria. Nell'esempio riportato si tratta delle pagine 1, 2, 3, 5, cui corrispondono rispettivamente gli indirizzi virtuali da 1024 a 2047, da 2048 a 3071, da 3072 a 4095 e da 5120 a 6143 (NB: l'indirizzo della prima parola della pagina x si calcola come $x \cdot 1024$, mentre l'indirizzo dell'ultima parola è $(x + 1) \cdot 1024 - 1$).
- (b) Vedi lucidi del corso.