Alunno: Matricola:

Università degli studi di Napoli

Parthenope

Corso di laurea in Informatica

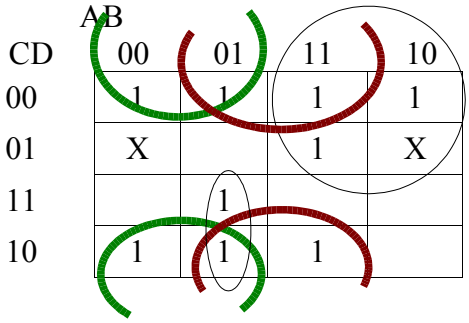
Architettura dei Calcolatori e Laboratorio – a.a. 2005-2006

Prova di esame del 7-2-2006

Esercizio 1

Data la seguente funzione booleana avente 4 ingressi ed una uscita:
On-set = [0,2,4,6,7,8,12,13,14]
Dc-set = [1,9]
Utilizzando il metodo delle mappe di Karnaugh sintetizzarla in forma minima come somma di prodotti.

Soluzione



F=!A!D+B!D+A!C+!ABC

Esercizio 2

Risolvere l'esercizio precedente utilizzando il programma SIS. Riportare il file blif e l' equazione ottenuta dopo la minimizzazione.

Soluzione

```
.model es2
.inputs a b c d
.outputs x
.names a b c d x
0000 1
0010 1
0100 1
0110 1
0111 1
1000 1
1100 1
1101 1
1110 1
```

Alunno: Matricola:

```
.exdc
.names a b c d x
0001 1
1001 1
.end
```

Dopo full_simplify

F=!A!D+B!D+A!C+!ABC

Esercizio 3

Scrivere i sottoprogramma assembly che eseguono le strutture indicate di seguito:

1- do
   D0=D0+3
   While D0<40

   Soluzione

```
        org $8000
start   add.l #3,D0
        cmp.l #40,D0
        blt     start
        stop #$2700
        end start
```

2- spostare un vettore di 10 elementi dagli indirizzi di memoria 8100-8109 agli indirizzi 8110-8119 (ciclo for).

Soluzione

```
                org $8000
start   clr.b D0
        move.b #10,D1
        lea     array1,A0
        lea     array2,A1
ciclo   move.b          (A0)+,D0
        move.b          D0,(A1)+
        addi #-1,D1
        bne ciclo
        stop #$2700
        org $8100
array1  dc.b    10,1,2,3,4,5,6,7,8,9
        org $8110
array2  ds.b    10
        end start
```

Esercizio 4

Sia data la seguente specifica funzionale di una macchina a stati finiti sincrona con due ingressi ed una uscita.
Indicati con x(k), y(k) e z(k) rispettivamente i due ingressi e l'uscita della macchina al k-esimo ciclo di clock si ha che z(k) = 1 se y(k)=x(k-1) e se x(k)=y(k-1), altrimenti z(k)=0.

Alunno: Matricola:

Utilizzando il programma SIS:
1. Tracciare il diagramma degli stati.
2. Minimizzare il diagramma degli stati.
3. Sintetizzare le funzioni di stato prossimo e di uscita utilizzando come elementi di memoria flip-flop di tipo D.

Soluzione

Z=1 quando si hanno le combinazioni degli ingressi:

ingressi precedenti	ingressi attuali
00	00
01	10
10	01
11	11

Indicando con R lo stato di reset e con a, b, c, d gli altri stati, si ha:

	ingressi			
	00	01	10	11
R	a/0	b/0	c/0	d/0
a	a/1	b/0	c/0	d/0
b	a/0	b/0	c/1	d/0
c	a/0	b/1	c/0	d/0
d	a/0	b/0	c/0	d/1

```
.model ESERC_4
.inputs x y
.outputs z
.start_kiss
.i 2
.o 1
.s 5
00 R a 0
00 a a 1
00 b a 0
00 c a 0
00 d a 0
01 R b 0
01 a b 0
01 b b 0
01 c b 1
01 d b 0
10 R c 0
10 a c 0
10 b c 1
10 c c 0
10 d c 0
11 R d 0
11 a d 0
11 b d 0
11 c d 0
11 d d 1
.end_kiss
.end
```

Con i comandi state_minimize, state_assign, stg_to_network si ottengono le risposte ai punti 2 e 3.

Alunno: Matricola:

Esercizio 5

Realizzare un programma in assembly che svolga la stessa funzione della macchina a stati finiti precedente.
Il valore degli ingressi x e y sono rispettivamente il bit 0 ed il bit 1 del byte dati all'indirizzo $8050.
L'uscita z è il bit 4 dello stesso byte.
La lettura degli ingressi è fatta tramite sottoprogramma che salva il byte dati nel registro D1.

Soluzione (non ottimizzata ma funzionale)

```
        org $8050
rd      ds.b 1
        org $8200
start   jsr lettura     lettura primo valore
reset   cmpi #0,d1
        beq stato_a
        cmpi #1,d1
        beq stato_b
        cmpi #2,d1
        beq stato_c
        bra stato_d
stato_a         jsr lettura     stato 00
        cmpi #0,d1
        bne altro1
        bset #4,rd              se input=00 setta bit 4
        bra stato_a             resta nello stato 00
altro1  bclr #4,rd              se input<>00 resetta bit 4
        cmpi #1,d1              vai nello stato corrispondente
        beq stato_b
        cmpi #2,d1
        beq stato_c
        bra stato_d
stato_b         jsr lettura     stato 01
        cmpi #2,d1
        bne altro2
        bset #4,rd
        bra stato_c
altro2  bclr #4,rd
        cmpi #0,d1
        beq stato_a
        cmpi #1,d1
        beq stato_b
        bra stato_d
stato_c                 jsr lettura     stato 10
        cmpi #1,d1
        bne altro3
        bset #4,rd
        bra stato_b
altro3  bclr #4,rd
        cmpi #0,d1
        beq stato_a
        cmpi #2,d1
        beq stato_c
        bra stato_d
stato_d         jsr lettura     stato 11
        cmpi #3,d1
        bne altro4
        bset #4,rd
```

Alunno: Matricola:

```
        bra stato_d
altro4  bclr #4,rd
        cmpi #1,d1
        beq stato_b
        cmpi #2,d1
        beq stato_c
        bra stato_a
lettura move.b (rd),d1          sub lettura - dato in d1
        andi #3,d1              reset di tutti i bit esclusi bit 0 e bit 1
        rts
        stop #$2700
        end start
```