

# Elementi di Architettura

Tiziano Villa

6 Settembre 2022

Nome e Cognome:

Matricola:

Posta elettronica:

problema	punti massimi	i tuoi punti
problema 1	10	
problema 2	10	
problema 3	10	
totale	30	

1. Si consideri la funzione combinatoria che vale 1 per i mintermini 0, 1, 4, 5, 7, 12, 13, non e' specificata per i mintermini 6, 10 e 15, vale 0 altrimenti.
- (a) Si rappresenti la funzione specificata con una somma di prodotti avente un numero minimo di termini prodotto. Si usi la mappa di Karnaugh per la minimizzazione.
  - (b) Si scriva la definizione d'implicante primo. Si elenchino tutti gl'implicanti primi di tale funzione.
  - (c) Si scriva la definizione d'implicante essenziale. Quali tra i precedenti implicanti primi sono essenziali ? Si motivi la risposta.

Traccia di soluzione.

Si veda l'allegato per la minimizzazione di tutte e tre le specifiche della funzione combinatoria data.

Si riveda attentamente la definizione di implicante primo, implicante primo essenziale.

Si noti che la funzione della prima specifica ha 4 implicanti primi di cui solo 2 essenziali. Ci sono due possibili forme minime ottenute dai due primi essenziali uniti all'uno o all'altro dei due primi non essenziali.

- (d) Si ripeta l'intero esercizio assumendo che la funzione valga 0 anche per i mintermini 6, 10 e 15, confrontando il risultato con quello ottenuto nel caso precedente. Come cambiano il numero di termini prodotto e il numero di letterali ?

- (e) Si ripeta l'intero esercizio assumendo che la funzione valga 1 anche per i mintermini 6, 10 e 15, confrontando il risultato con quelli ottenuti nei due casi precedenti. Come cambiano il numero di termini prodotto e il numero di letterali ?

2. Si progetti un circuito sequenziale che individua una stringa in ingresso con la seguente specifica:

- C'è una variabile binaria in ingresso  $X$ , e una variabile binaria in uscita  $Z$ .
- Il circuito pone a 1 per un'unità di tempo il valore dell'uscita  $Z$  quando rileva nell'ingresso  $X$  la stringa 1101 e poi azzerata l'uscita  $Z$  fino alla prossima rilevazione di 1101. In altri termini, ogni volta che si rileva in ingresso la sottosuccessione 1101, l'uscita  $Z$  è asserita a 1 nel ciclo successivo, poi è azzerata fino a che non si ricomincia a cercare tale sottosuccessione.

Si noti che le sottosuccessioni 1101 oltre che ripetersi separatamente possono essere innestate l'una nell'altra, come nell'esempio 1101101, dove l'uscita vale 1 dopo la rilevazione dell'uno finale della prima occorrenza di 1101 e anche dopo la rilevazione dell'uno finale della seconda occorrenza di 1101, la quale inizia dall'uno finale della prima occorrenza.

- (a) Si progetti la macchina a stati finiti (tipo **Moore**) che modella la specifica disegnando il grafo delle transizioni. S'indichi lo stato iniziale.

Traccia di soluzione.

Si veda il grafo degli stati nell'allegato.

- (b) Si minimizzi il numero degli stati della macchina proposta, applicando l'algoritmo di minimizzazione degli stati.
- (c) Si scriva la tavola delle transizioni con gli stati futuri e le uscite e la si codifichi.

- (d) Supponendo di usare bistabili di tipo D, si derivino le equazioni minimizzate di eccitazione degl'ingressi dei bistabili e le equazioni minimizzate delle uscite. Si esegua e mostri la minimizzazione con le mappe di Karnaugh.

- (e) Si realizzi il circuito sequenziale corrispondente con bistabili di tipo D campionati sul fronte di salita, invertitori e porte NAND. Si etichettino con chiarezza i segnali.



3. Si consideri il seguente codice LC-3.

```
        .ORIG    x3000
        LD R2, INIZIO
        LD R3, ASCII
        LD R4, FINE
RIPETI  TRAP x23
        ADD R1, R2, R0
        BRn USCITA
        ADD R1, R4, R0
        BRp USCITA
        ADD R0, R0, R3
        TRAP x21
        BRnzp RIPETI
USCITA  TRAP x25
INIZIO  .FILL xFFBF
FINE    .FILL xFFA6
ASCII   .FILL x0020
        .END
```

Si spieghi il funzionamento di questo programma, descrivendo il comportamento delle singole istruzioni e poi deducendo da esse il suo comportamento globale.

Traccia di soluzione.

```
        .ORIG x3000
        LD R2, INIZIO    ; carica -A (in ASCII A=$x41$)
        LD R3, ASCII     ; carica la differenza in ASCII
        LD R4, FINE      ; carica -Z (in ASCII A=$x5A$)
RIPETI  TRAP x23          ; richiedi un carattere da tastiera
        ADD R1, R2, R0
        BRn USCITA
        ADD R1, R4, R0
        BRp EXIT
        ADD R0, R0, R3    ; cambia maiuscola in minuscola
        TRAP x21          ; visualizza sullo schermo
        BRnzp RIPETI     ; salta a ripeti
```

```

USCITA TRAP x25          ; fermati
INIZIO  .FILL xFFBF      ; FFBF = -A
FINE    .FILL xFFA6      ; FFA6 = -Z
ASCII   .FILL x0020      ; differenza tra maiuscola
                                e minuscola
                                ;
                                .END

```

Il programma riceve maiuscole da tastiera e visualizza le corrispondenti minuscole sullo schermo, e termina se non riceve in ingresso una maiuscola.

Ad ogni carattere immesso, se esso corrisponde a una posizione nella tabella ASCII minore di  $65 = x41$  o maggiore di  $90 = x5A$  il programma termina perché non è una maiuscola; altrimenti converte la maiuscola in minuscola e la visualizza sullo schermo.

Si noti che:

- (a) Se  $R1 = R2 + R0 < 0$  vuol dire che il carattere letto viene prima di *A* nella tabella ASCII (quindi non è una maiuscola);
- (b) Se  $R1 = R4 + R0 > 0$  vuol dire che il carattere letto viene dopo *Z* nella tabella ASCII (quindi non è una maiuscola).

La differenza tra maiuscole (da posizione 65) e minuscole (da posizione 97) vale  $97 - 65 = 32_{10} = 20_{16}$ .