COMPITINO di ARCHITETTURA DEI CALCOLATORI del 05-11-2025

MATRICOLA_____

COGNOME_____

NOME_____

DA RESTITUIRE INSIEME AGLI ELABORATI e A TUTTI I FOGLI
→ NON USARE FOGLI NON TIMBRATI
→ ANDARE IN BAGNO PRIMA DELL'INIZIO DELLA PROVA
→ NO FOGLI PERSONALI, NO TELEFONI, SMARTPHONE/WATCH, ETC

NOTE: I FOGLI UTILIZZATI PER RAGIONAMENTI VANNO RICONSEGNATI ANCHE SE BIANCHI; PER I FILE:

- per l'esercizio 4 consegnare DUE files: il file del programma VERILOG di nome <COGNOME>.v
  e il file del diagramma temporale (screenshot o copy/paste → usare tasto 'STAMP') <COGNOME>.png

3) [12/30] Data una funzione booleana con 4 ingressi x3,x2,x1,x0 e una uscita z tale che l'uscita vale 1 qualora gli ingressi rappresentino in binario una delle cifre decimali 2, 3, 5, 7:
   a. [4/30]Utilizzando le mappe di Karnaugh, sintetizzare tale funzione in forma algebrica (suggerimento: gli ingressi sono 4 cifre booleane x3, x2, x1, x0 ei valori 10,11,12,13,14,15 sono non-specificati).
   b. [4/30] Realizzare la funzione del punto a utilizzando solo porte NAND
   c. [4/30] Assumendo 150ps per il ritardo di ogni porta NAND, calcolare il ritardo del cammino più critico e il ritardo del cammino più veloce.

4) [18/30] Realizzare in Verilog una macchina a stati di tipo Mealy-ritardato con un ingresso X a 1 bit e un'uscita Z a 1 bit, che riconosca esclusivamente sequenze non interallacciate del tipo 1, 0, 1, 0. All'inizio e dopo ogni riconoscimento, la macchina si resetta automaticamente. Al reset l'uscita Z è 0. L'uscita Z diventa 1 esattamente per un ciclo di clock al riconoscimento della sequenza. Tracciare il diagramma di temporizzazione [punti 8/30] corrispondente alla esecuzione del modulo Verilog realizzato, come verifica della correttezza dell'unità riportando i segnali clock, /reset, ingresso X, uscita Z e stato STAR per la durata complessiva. Nota: salvare una copia dell'output (diagramma temporale) e del programma Verilog su USB-drive del docente.

```
module Toplevel;
  reg reset_; initial begin reset_=0; #22 reset_=1; #300; $stop; end
  reg clock; initial clock=0; always #5 clock<=(!clock);
  reg   X;
  wire  Z;
  wire [2:0] STAR=Xxx.STAR;
  initial begin X=0;
    wait(reset_==1);
    @(posedge clock); X<=1;@(posedge clock); X<=0;@(posedge clock); X<=1;
    @(posedge clock); X<=0;@(posedge clock); X<=1;@(posedge clock); X<=0;
    @(posedge clock); X<=1;@(posedge clock); X<=1;@(posedge clock); X<=0;
    @(posedge clock); X<=0;@(posedge clock); X<=1;@(posedge clock); X<=0;
    @(posedge clock); X<=1;@(posedge clock); X<=0;@(posedge clock); X<=1;
    @(posedge clock); X<=0;@(posedge clock); X<=1;@(posedge clock); X<=0;
    @(posedge clock); X<=0;@(posedge clock); X<=0;@(posedge clock); X<=0;
  end
  ricseq1010mealyrit Xxx(Z,  X,clock,reset_);
endmodule
```

COMPITINO di ARCHITETTURA DEI CALCOLATORI del 05-11-2025

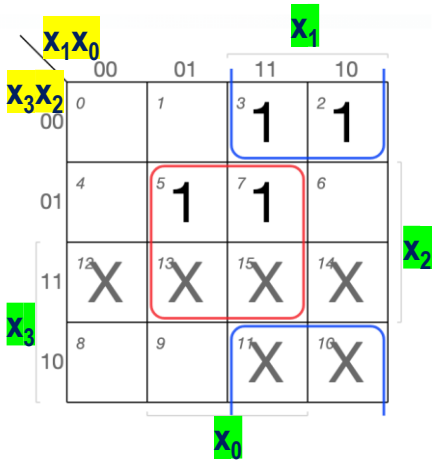
MATRICOLA_____

COGNOME_____

NOME_____

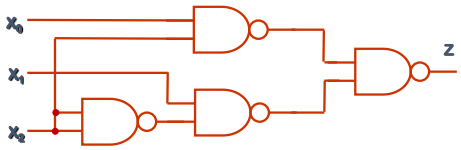
ESERCIZIO 3

3a) Sintesi con le mappe di Karnaugh:



$z = x_2x_0 + /x_2x_1$

3b) Realizzazione solo con porte NAND.
Utilizzando più volte De Morgan:
z = x2x0+/x2x1 =
= //(x2x0+/x2x1) = /(/(x2x0)/(/x2x1))=

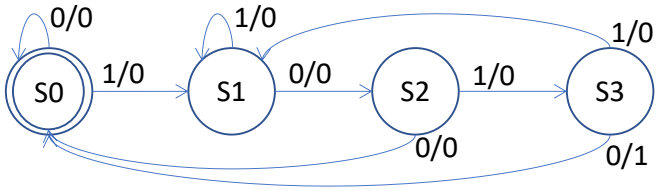


3c)
Cammino più critico (peggiore):
da x2 a z (3 livelli) = 3·150 ps = 450 ps

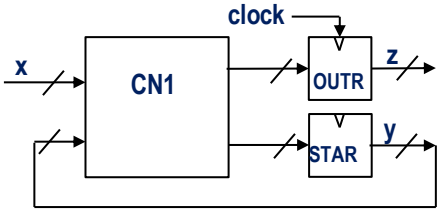
Cammino più veloce:
da x1→z o x0→z (2 livelli) = 2·150 ps = 300 ps

ESERCIZIO 4

Diagramma degli Stati



Schema di riferimento per Mealy Ritardato



Codice Verilog del modulo da realizzare (Mealy-ritardato):

```
module ricseq1010mealyrit(z,  x, clock, reset_);
  input             clock, reset_;
  input             x;
  output            z;
  reg         [2:0] STAR;
  reg               OUTR;

  // Stati: S4 = stato "pulse" dopo il riconoscimento
  parameter S0 = 3'b000, S1 = 3'b001, S2 = 3'b010, S3 = 3'b011;

  // Reset asincrono attivo basso
  always @(reset_==0) #1 begin STAR <= S0; OUTR<= 1'b0; end
  assign z = OUTR;

  // Transizioni + uscita registrata secondo il template
  always @(posedge clock) if (reset_==1) #3
    casex (STAR)
      S0: begin OUTR <= 1'b0;                    STAR <= (x ? S1 : S0); end
      S1: begin OUTR <= 1'b0;                    STAR <= (x ? S1 : S2); end
      S2: begin OUTR <= 1'b0;                    STAR <= (x ? S3 : S0); end
      S3: begin OUTR <= (x ? 1'b0 : 1'b1);       STAR <= (x ? S1 : S0); end // match (x=0)
    endcase
endmodule
```

Diagramma di Temporizzazione: