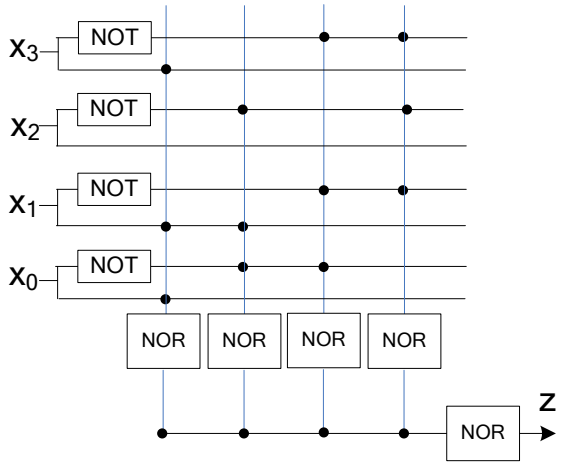
Reti Logiche, 21/7/2020

Esercizio 1

Si consideri la seguente rete combinatoria:



1) Disegnare la mappa di Karnaugh
2) Nell'ipotesi che non si presentino mai i due stati di ingresso $\{x_3, x_2, x_1, x_0\} = 1101$ e $\{x_3, x_2, x_1, x_0\} = 1000$, inserire nella mappa i corrispondenti *non specificati*
3) Sulla mappa di cui al punto precedente:
   a. individuare e classificare gli implicanti principali
   b. produrre *tutte* le liste di copertura irridondanti, ed indicare quali sono di costo minimo (criterio a porte)

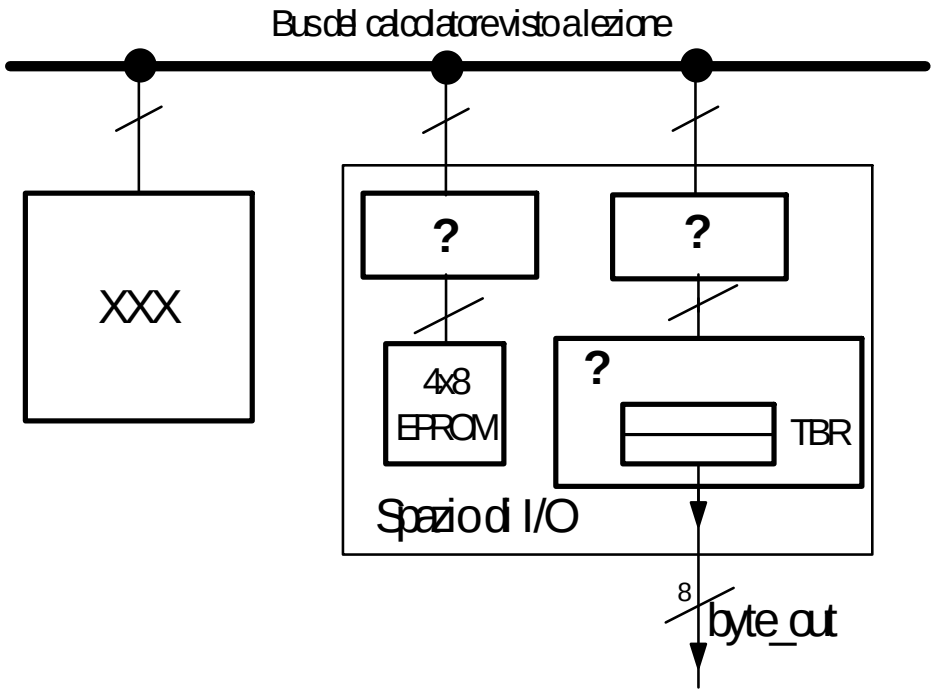
Esercizio 2

Specificare (con disegno o tramite Verilog) lo spazio di I/O specificando le tre scatole ? in modo che la EPROM sia sempre selezionata e risponda alle letture nello spazio di I/O e la scatola con il registro TBR sia sempre selezionata e risponda alle scritture nello spazio di I/O. Eliminare poi dal bus tutti i fili inutili.

Descrivere l'unità XXX in modo che ripeta ciclicamente, utilizzando un registro COUNT e con un ritmo pari a 20 periodi di clock, quanto segue: emettere tramite il registro TBR il contenuto della locazione della EPROM successiva a quella trattata nel precedente ciclo.

Disegnare il circuito della Parte Operativa relativo al registro COUNT.

Si assuma che la EPROM risponda molto velocemente, in modo che non siano necessari stati di wait.



Reti Logiche, 21/7/2020

Esercizio 1 – Soluzione

1) La rete combinatoria di figura sintetizza la seguente legge:

$$z = \overline{\overline{(x_3 + x_1 + x_0)} + \overline{(\overline{x_2} + x_1 + \overline{x_0})} + \overline{(\overline{x_3} + \overline{x_1} + \overline{x_0})} + \overline{(\overline{x_3} + \overline{x_2} + \overline{x_1})}}$$
$$= \overline{(\overline{x_3} \cdot \overline{x_1} \cdot \overline{x_0}) + (x_2 \cdot \overline{x_1} \cdot x_0) + (x_3 \cdot x_1 \cdot x_0) + (x_3 \cdot x_2 \cdot x_1)}$$

cui corrisponde la mappa di Karnaugh disegnata a sinistra

x_1x_0 \ x_3x_2	00	01	11	10
00	0	0	1	1
01	1	0	0	1
11	1	1	0	0
10	1	1	0	1

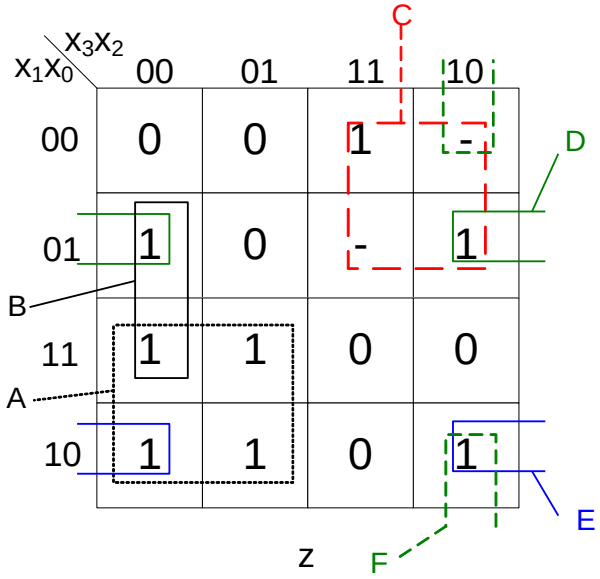
z

x_1x_0 \ x_3x_2	00	01	11	10
00	0	0	1	-
01	1	0	-	1
11	1	1	0	0
10	1	1	0	1

z

2) Dopo l'inserimento dei due non specificati, la mappa di Karnaugh diventa quella disegnata sopra a destra.

3) Per la mappa trovata si hanno gli implicanti principali elencati di seguito:



$$A = \overline{x_3} \cdot x_1,\ B = \overline{x_3} \cdot \overline{x_2} \cdot x_1,\ C = x_3 \cdot \overline{x_1},\ D = \overline{x_2} \cdot \overline{x_1} \cdot x_0,\ E = \overline{x_2} \cdot x_1 \cdot \overline{x_0},\ F = x_3 \cdot \overline{x_2} \cdot \overline{x_0}$$

Di questi, A e C sono implicanti essenziali; nessun implicante è assolutamente eliminabile; B, D, E, F sono implicanti semplicemente eliminabili. Le possibili liste di copertura irridondanti sono {A,C,D,E}, {A,C,D,F}, {A,C,B,E}, {A,C,B,F}, tutte di costo identico.

Esercizio 2 - Una Soluzione La scatola con il registro TBR è una interfaccia parallela di uscita senza handshake (chiamiamola ParallelOut) sempre selezionata. Il bus si riduce a: / ior , / iow , a 1_ a 0 (due bit per indirizzare la EPROM) e d 7_ d 0; Lo schema a blocchi dello spazio di I/O (descritto in Verilog) è il seguente:

```
module IO_space(d7_d0,a1_a0,ior_,iow_,byte_out);
  input[1:0]  a1_a0;
  inout[7:0]  d7_d0; // 8-bit data bus
  input       ior_,iow_;
  output[7:0] byte_out;
  wire sParallelOut_;  assign sParallelOut_=0;
  ParallelOut  PARALLELOUT(d7_d0,sParallelOut_,iow_,byte_out);
  wire  sEPROM_;       assign sEPROM_=0;
  Eprom EPROM(d7_d0,a1_a0,sEPROM_,ior_);
endmodule
```

Ciò premesso, l'Unità XXX ha la seguente struttura

```
module XXX(d7_d0,a1_a0, ior_,iow_,clock,reset_);
 input       clock,reset_;
 output      ior_,iow_;
 output[1:0] a1_a0;
 inout[7:0]  d7_d0;

 reg         DIR,IOR_,IOW_;  assign ior_=IOR_; assign   iow_=IOW_;
 reg[1:0]    A1_A0;          assign a1_a0=A1_A0;
 reg[7:0]    D7_D0;          assign d7_d0=(DIR==1)?D7_D0:'HZZ; //FORCHETTA
 reg [3:0]   COUNT;
 reg [2:0]   STAR;           parameter [2:0] S0=0,S1=1,S2=2,S3=3,S4=4,S5=5;

 parameter num_periodi=20;

 always @(reset_==0) #1 begin COUNT<=num_periodi; DIR<=0; IOR_<=1; IOW_<=1;
                            A1_A0<=0; STAR<=S0; end

 always @(posedge clock) if (reset_==1) #3
  casex(STAR)
   S0: begin COUNT<=COUNT-1; IOR_<=0; STAR<=S1; end
   S1: begin COUNT<=COUNT-1; D7_D0<=d7_d0; IOR_<=1; STAR<=S2; end
   S2: begin COUNT<=COUNT-1; DIR<=1; A1_A0<=(A1_A0)+1; STAR<=S3; end
   S3: begin COUNT<=COUNT-1; IOW_<=0; STAR<=S4; end
   S4: begin COUNT<=COUNT-1; IOW_<=1; STAR<=S5; end
   S5: begin COUNT<=(COUNT==1)?num_periodi:COUNT-1; DIR<=0;
             STAR<=(COUNT==1)?S0:S5; end
  endcase
endmodule
```