Compito di Reti Logiche dell'11/01/2016

Cognome e Nome _____ Matricola: _____

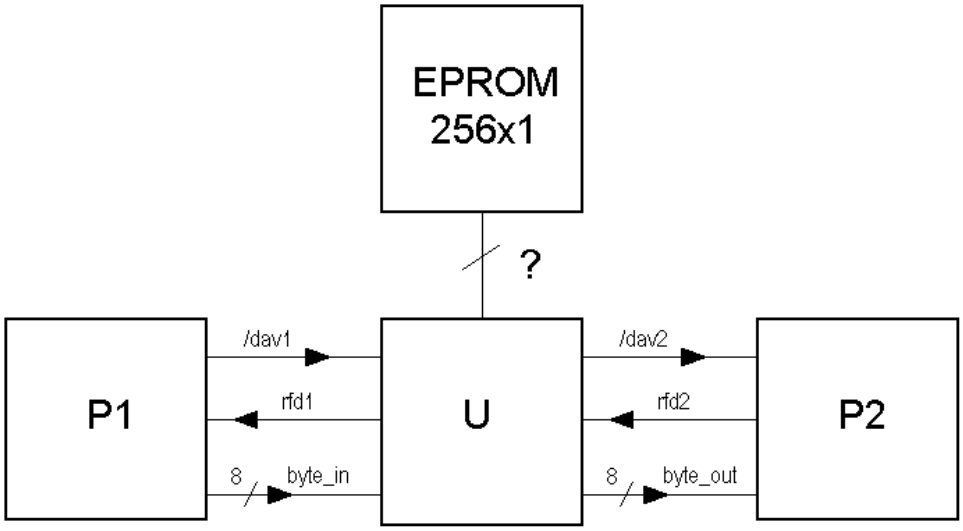
Esercizio 1

Sintetizzare, secondo il modello strutturale con elementi neutri di ritardo, la rete sequenziale asincrona descritta dalla seguente tabella di flusso. Per la sintesi delle reti combinatorie utilizzare solo porte NOR.

x_1x_0	00	01	11	10	z
S_0	(S_0)	(S_0)	(S_0)	S_1	0
S_1	S_0	—	S_2	(S_1)	0
S_2	—	S_0	(S_2)	S_1	1

Esercizio 2

Descrivere e <u>sintetizzare</u> l'Unità **U**, definita funzionalmente come segue.



1) Riceve numeri naturali ad 8 bit da P1 ed invia numeri naturali ad 8 bit a P2, instaurando con entrambi un protocollo di handshake del tipo */dav*, *rfd*.
2) Ogni volta che riceve un nuovo numero naturale *x*, lo interpreta un indirizzo per accedere (in lettura) alla EPROM da 256x1 bit.
3) Se il bit ritornato dalla EPROM vale 1, il numero naturale *x* viene trasmesso a P2, altrimenti viene ignorato e viene iniziato un nuovo ciclo di acquisizione di un nuovo numero naturale da P1, e cosi via all'infinito.

Specificare i collegamenti con la EPROM.

Soluzione esercizio 1

La rete sequenziale è un riconoscitore della sequenza di stati di ingresso 10, 11. Adottando le codifiche S_0 = 00, S_1 = 01, S_2 = 11, si rende necessario uno stato ponte fra S_2 e S_0, nel passaggio dello stato d'ingresso da 'B11 a 'B01. Poiché per lo stato di ingresso 'B01 lo stato successivo corrispondente allo stato S_1 è non specificato, quest'ultimo può essere usato come stato ponte. La tabella diventa quindi:

x_1x_0	00	01	11	10	z
S_0	(S0)	(S0)	(S0)	S1	0
S_1	S0	S0	S2	(S1)	0
S_2	—	S1	(S2)	S1	1

Con riferimento al modello strutturale con elementi neutri di ritardo, le mappe di Karnaugh relative alle uscite della rete CN1 sono riportate in figura. Le forme PS corrispondenti (esenti da alee statiche) sono $a_1 = x_1 \cdot x_0 \cdot y_0$, $a_0 = (x_1 + y_1) \cdot (\overline{x}_0 + y_0)$, da cui si ottiene: $a_1 = \overline{\overline{x_1 \cdot x_0 \cdot y_0}} = \overline{\overline{x_1} + \overline{x_0} + \overline{y_0}}$,

$a_0 = \overline{\overline{(x_1 + y_1)} + \overline{(\overline{x}_0 + y_0)}}$.

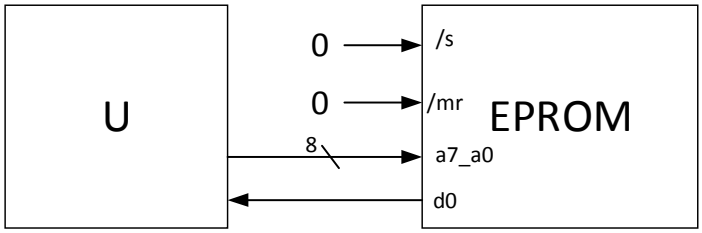
y_1y_0 \ x_1x_0	00	01	11	10	z
00	00	00	00	01	0
01	00	00	11	01	0
11	--	01	11	01	1
10	--	--	--	--	-

a_1a_0

Per CN2, è immediato verificare che $z = y_1$.

Soluzione esercizio 2

Il collegamento tra l'unità U e la EPROM richiede le seguenti linee:



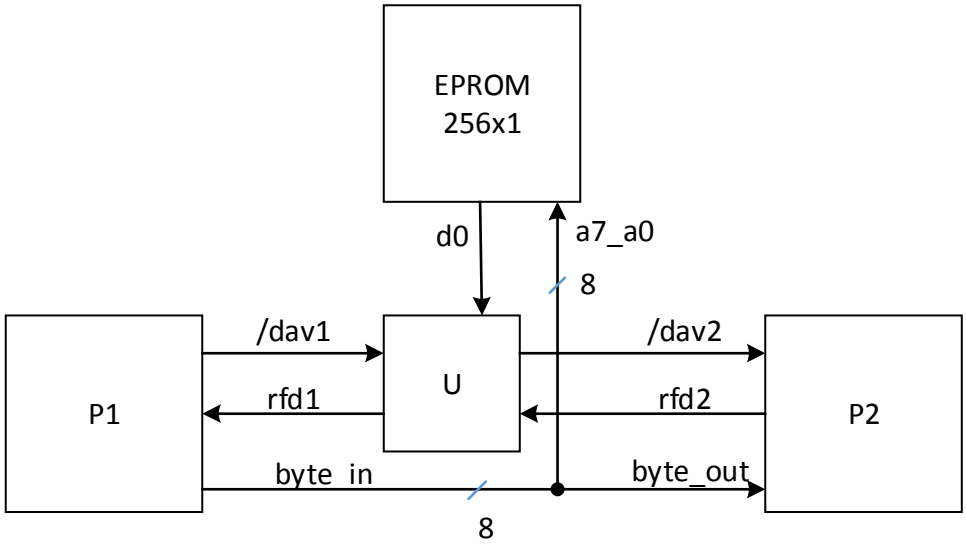
Una possibile descrizione è la seguente:

```
module Unit_U (dav1_,rfd1,x,dav2_,rfd2,byte_out,d0,a7_a0,clock,reset_);
 input         clock, reset_;
 input         dav1_, rfd2;
 output        rfd1,  dav2_;
 input  [7:0]  x;
 output [7:0]  byte_out;
 input         d0;
 output [7:0]  a7_a0;

 reg        RFD1, DAV2_;  assign rfd1=RFD1; dav2_=DAV2_;
 reg [7:0]  MAR;          assign byte_out= MAR; assign a7_a0=MAR;
 reg [2:0]  STAR;   parameter S0=0,S0=1,S1=2,S3=3,S3=4,S4=5;

 always @(reset_==0) begin RFD1<=1; DAV2_<=1; STAR=S0; end
 always @(posedge clock) if (reset_==1) #3
  casex(STAR)
   S0: begin RFD1<=1; MAR<=x; STAR<=(dav1_==1)?S0:S1; end
   S1: begin RFD1<=0; STAR<=(dav1_==0)?S1:S2; end
   S2: begin STAR<=(d0==1)?S3:S0; end
   S3: begin DAV2_<=0; STAR<=(rfd2==1)?S3:S4; end
   S4: begin DAV2_<=1; STAR<=(rfd2==0)?S4:S0; end
  endcase
endmodule
```

Si può semplificare la descrizione osservando che la variabile di ingresso *x* può essere connessa direttamente alle uscite a7_a0 e byte_out, come mostrato in figura.



In questo caso, non è più necessario il registro MAR, e i filli x, a7_a0 e byte_out non entrano nella descrizione di U, ma è necessario posporre l'handshake con P1 a dopo che P2 ha prelevato byte_out. La descrizione può quindi essere riscritta come segue:

```
module Unit_U (dav1_,rfd1, dav2_,rfd2,clock,reset_);
 input          clock, reset_;
 input          dav1_, rfd2;
 output         rfd1,  dav2_;
 input          d0;

 reg         RFD1, DAV2_;  assign rfd1=RFD1; dav2_=DAV2_;
 reg [2:0]   STAR;   parameter S0=0,S0=1,S1=2,S3=3,S3=4,S4=5;

 always @(reset_==0) begin RFD1<=1; DAV2_<=1; STAR=S0; end
 always @(posedge clock) if (reset_==1) #3
  casex(STAR)
   S0: begin RFD1<=1; STAR<=(dav1_==1)?S0:S1; end
   S1: begin STAR<=S2; end
   S2: begin STAR<=(d0==1)?S3:S4; end
   S3: begin DAV2_<=0; STAR<=(rfd2==1)?S3:S4; end
   S4: begin DAV2_<=1; RFD1<=0; STAR<=({dav_1,rfd2}=='B00)?S4:S0; end
  endcase
endmodule
```