Esonero del 4/2/2011

Compito A

Esercizio 1A

L'automa riconoscitore delle sequenze BCA, BAC, BAA anche se sovrapposte può essere realizzato usando gli stati che corrispondono alle diverse sottostringhe via via riconosciute: '-' (nulla), 'B', 'BC', 'BCA', 'BA', 'BAC', 'BAA'.

La tabella di transizione dell'automa sarà

	A	**B**	**C**
'-'	'-'/0	'B'/0	'-'/0
'B'	'BA'/0	'B'/0	'BC'/0
'BC'	'BCA'/1	'B'/0	'-'/0
'BCA'	'-'/0	'B'/0	'-'/0
'BA'	'BAA'/1	'B'/0	'BAC'/1
'BAC'	'-'/0	'B'/0	'-'/0
'BAA'	'-'/0	'B'/0	'-'/0

Osservando la tabella si vede subito che le righe degli stati '-', 'BCA', 'BAC' e 'BAA' sono uguali, quindi questi 4 stati sono sicuramente equivalenti. Le due righe degli stati 'BA' e 'BC' producono output diversi da tutte le altre, quindi questi stati NON sono equivalenti ad altri, l'unico confronto che resta da fare tra righe che hanno output uguali è tra la riga 'B' e la riga '-' (o una delle altre 3 ad essa equivalenti). Osservando gli stati di arrivo delle transizioni si vede che le due righe sono equivalenti SOLO SE sono equivalenti le righe '-' e 'BA' E le due righe '-' e 'BC'. Ma abbiamo visto che queste non possono esserlo per cui 'B' NON è equivalente a '-'.

Definiamo l'automa minimo con i nuovi stati, la cui tabella di transizione sarà:

	A	**B**	**C**
W0 = { '-', 'BCA', 'BAC', 'BAA' }	W0/0	W1/0	W0/0
W1 = { 'B' }	W3/0	W1/0	W2/0
W2 = { 'BC' }	W0/1	W1/0	W0/0
W3 = { 'BA' }	W0/1	W1/0	W0/1

A questo punto dobbiamo codificare gli stati e i simboli di ingresso (i simboli di uscita appartengono ad un alfabeto binario per cui li consideriamo già codificati). Ad esempio potremmo codificare

Stato	**Codifica**
W0	'00'
W1	'01'
W2	'10'
W3	'11'

Simbolo	**Codifica**
A	'00'
B	'01'
C	'10'

Dobbiamo anche scegliere quali flip-flop usare per rappresentare gli stati.
Usiamo un JK per il bit più significativo ed un SR per il meno significativo.
Abbiamo 2 bit per gli stati più 2 bit per i simboli e quindi la tabella di transizione avrà 16 righe.
Si noti che l'input '11' non potrà mai apparire, per cui le ultime 4 righe della tabella hanno solo dont' care.
Per riempire le colonne JK e SR si usano le tabelle inverse di transizione dei due tipi di flip-flop, che indicano quali segnali mandare in ingresso se si vuol ottenere la transizione desiderata, ovvero:

y	Y	J	K	S	R
0	0	0	-	0	-
0	1	1	-	1	0
1	0	-	1	0	1
1	1	-	0	-	0

Quindi la tabella di transizione della macchina sequenziale è:

x_1	x_0	y_1	y_0	Y_1	Y_0	Z	J_1	K_1	S_0	R_0
0	0	0	0	0	0	0	0	-	0	-
0	0	0	1	1	1	0	1	-	-	0
0	0	1	0	0	0	1	-	1	0	-
0	0	1	1	0	0	1	-	1	0	1
0	1	0	0	0	1	0	0	-	1	0
0	1	0	1	0	1	0	0	-	-	0
0	1	1	0	0	1	0	-	1	1	0
0	1	1	1	0	1	0	-	1	-	0
1	0	0	0	0	0	0	0	-	0	-
1	0	0	1	1	0	0	1	-	0	1
1	0	1	0	0	0	0	-	1	0	-
1	0	1	1	0	0	1	-	1	0	1
1	1	0	0	-	-	-	-	-	-	-
1	1	0	1	-	-	-	-	-	-	-
1	1	1	0	-	-	-	-	-	-	-
1	1	1	1	-	-	-	-	-	-	-

Esercizio 2A

Dovendo fare due trasferimenti **contemporaneamente**, con 4 sorgenti e 2 destinazioni, l'unico schema adatto è la mesh, ovvero un fascio di MUX (MUX0 e MUX1) per ciascuna delle destinazioni, con le 4 sorgenti che sono collegate in ingresso a entrambi i fasci di MUX.

A questo punto bisogna progettare due cose:

- come selezionare le due sorgenti dei due trasferimenti (ovvero quali segnali mandare agli ingressi di selezione di ciascuno dei due fasci di MUX)
- quando abilitare la scrittura nei registri destinazione

Il primo fascio MUX0 deve ricevere il valore i=RS0 mod 4, ovvero i 2 bit meno significativi di RS0.

Il valore j=3-i non è altro che il valore i con i bit invertiti, quindi al MUX1 mandiamo i due bit meno significativi di RS0 dopo averli negati.

Il bit di abilitazione del registro RD0 deve ricevere '1' se RS0>RS1, quindi usiamo un blocco comparatore tra RS1 ed RS0 ed otteniamo l'uscita '<' che ci permette di abilitare la scrittura in RD0.

Il bit di abilitazione di RD1 deve ricevere '1' quando RS3 = RS1+RS2. Usiamo un blocco sommatore ed un blocco comparatore, il primo per sommare RS1 ed RS2, il secondo per confrontare il risultato con RS3. L'uscita '=' del comparatore sarà collegata all'ingresso di abilitazione di RD1.

Compito B

Esercizio 1B

Le equazioni della parte combinatoria della macchina sequenziale da analizzare sono:

$$J = X \qquad K = \overline{y_0}\, XOR\, \overline{y_1}$$
$$S = y_0 \qquad R = Z\,\overline{y_0}$$
$$Z = \overline{X + y_1} = \overline{X}\,\overline{y_1}$$

Quindi la tabella di verità della macchina è

X	y_0	y_1	J	K	S	R	Z	Y_0	Y_1
0	0	0	0	0	0	1	1	0	0
0	0	1	0	1	0	0	0	0	1
0	1	0	0	1	1	0	1	0	1
0	1	1	0	0	1	0	0	1	1
1	0	0	1	0	0	0	0	1	0
1	0	1	1	1	0	0	0	1	1
1	1	0	1	1	1	0	0	0	1
1	1	1	1	0	1	0	0	1	1

Otteniamo dalla tabella sopra la tabella di transizione dell'automa (indicando con Si lo stato che ha codifica i)

	0	**1**
S0	S0/1	S2/0
S1	S1/0	S3/0
S2	S1/1	S1/0
S3	S3/0	S3/0

Minimizzando l'automa

S1	X		
S2	0 1 (X) 2 1 (X)	X	
S3	X	equivalente	X
	S0	**S1**	**S2**

Scopro che lo stato S1 è equivalente allo stato S3 mentre S2 non è equivalente ad S0

Per cui definisco i nuovi stati, la tabella di transizione dell'automa minimizzato è

	0	**1**
W0 = {S0}	W0/1	W2/0
W1 = {S1, S3}	W1/0	W1/0
W2 = {S2}	W1/1	W1/0

E la sequenza di stati che ottengo partendo da W1 (ex S3=11) è

W1--1-->W1--0-->W1--1-->W1--1-->W1--0-->W1--1-->W1--0-->W1--1-->W1--0-->W1--0-->W1

Esercizio 2B

Il circuito di trasferimento tra registri ha molte sorgenti e molte destinazioni, quindi può essere realizzato in 3 modi: con una mesh, con un bus oppure con un trasferimento molti-1-molti.

Visto che si richiede un solo trasferimento per volta non è necessario usare la mesh.

Il circuito più semplice per questo trasferimento è un bus che riceve i dati dagli 8 registri e li porta a tutti gli 8 registri.

Bisogna a questo punto costruire due cose:

- il circuito che fornisce ai three-state buffers i segnali per indicare qual'è la sorgente del trasferimento (i)
- il circuito che fornisce ai segnali di abilitazione dei registri l'indicazione di qual'è la destinazione del trasferimento (j)
- il circuito che indica se il trasferimento deve essere eseguito (R0-R1 sia pari che negativo)

Per attivare un solo registro sorgente usiamo un decodificatore che:

- riceve il valore i (ottenuto dai 3 bit meno significativi di R2)
- ha le sue 8 uscite collegate ai three-state buffers degli 8 registri

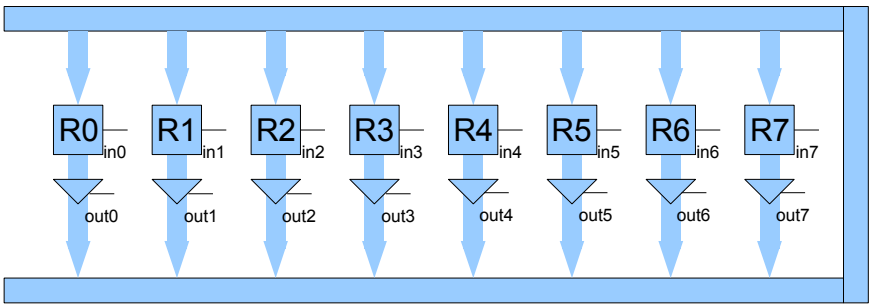
Notiamo che il valore j=7-i non è altro che il valore i con tutti i bit negati, quindi per selezionare la destinazione usiamo un secondo decodificatore che:

- riceve in ingresso il valore j ottenuto negando i 3 bit di i
- ha le sue 8 uscite collegate agli ingressi di abilitazione alla scrittura negli 8 registri

Per calcolare la condizione (R0-R1 sia pari che negativo) usiamo un blocco sommatore e prendiamo dal suo output:

- il bit più significativo (MSB, che vale 1 se il risultato è negativo)
- il bit meno significativo (LSB, che vale 1 se il risultato è dispari)

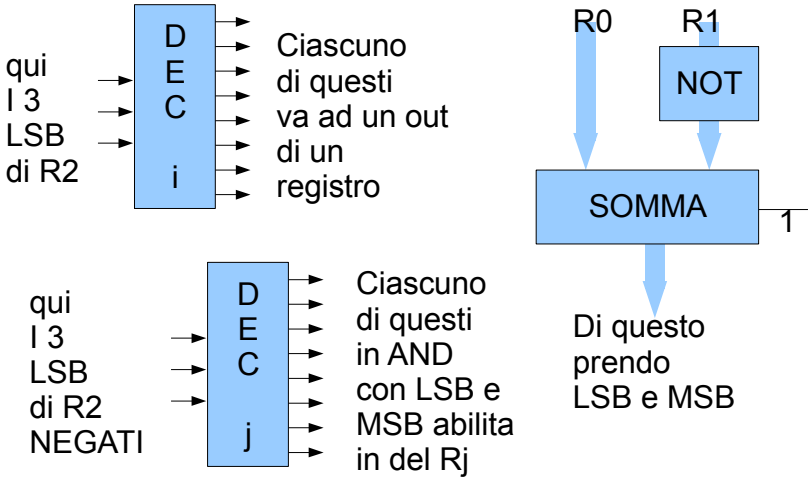
visto che dobbiamo abilitare la scrittura solo se la condizione è verificata, mettiamo in AND questi due bit con coi segnale di selezione alla scrittura dei registri. (ovvero ciascun registro riceve come segnale di abilitazione l'and tra questi due bit e il segnale di selezione che arriva dal decodificatore di j)



Discussione aggiuntiva:

Per permettere il trasferimento opposto, da Rj a Ri quando la condizione non è verificata basta scambiare i con j in quel caso. Ovvero basta negare i e non negare j a seconda del segnale di controllo.

Questo si ottiene usando un fascio di XOR sia all'ingresso del DEC i che all'ingresso del DEC j, uno comandato dalla condizione e l'altro dalla condizione negata. (lo XOR inverte un ingresso quando l'altro vale 1, altrimenti lo lascia uguale).



Compito C

Esercizio 1C

L'automa riconoscitore delle sequenze WXY, WXW, WWW può essere realizzato usando gli stati che corrispondono alle diverse sottostringhe via via riconosciute: '-' (nulla), 'W', 'WX', 'WXY', 'WXW', 'WW', 'WWW'. La tabella di transizione dell'automa sarà

	W	**X**	**Y**
'-'	'W'/0	'-'/0	'-'/0
'W'	'WW'/0	'WX'/0	'-'/0
'WX'	'WXW'/1	'-'/0	'WXY'/1
'WXY'	'W'/0	'-'/0	'-'/0
'WXW'	'WW'/0	'WX'/0	'-'/0
'WW'	'WWW'/1	'WX'/0	'-'/0
'WWW'	'WWW'/1	'WX'/0	'-'/0

Osservando la tabella si vede subito che alcune righe sono uguali, quindi equivalenti. Costruiamo comunque la tabella triangolare dei confronti.

'W'	W – WW (no) '-' - WX (no)					
'WX'	X	X				
'WXY'	equivalente		X			
'WXW'	W – WW (no) WX - '-' (no)	equivalente	X	W – WW (no) '-' - WX (no)		
'WW'	X	X	X	X	X	
'WWW'	X	X	X	X	X	equivalente
	'-'	**'W'**	**'WX'**	**'WXY'**	**'WXW'**	**'WW'**

Quindi sono equivalenti le coppie: 'WW' e 'WWW', 'WXW' e 'W', 'WXY' e '-'

Definiamo l'automa minimo con i nuovi stati, la cui tabella di transizione si ottiene da quella precedente sostituendo le vecchie destinazioni con i nuovi stati

	W	**X**	**Y**
W0 = { '-', 'WXY' }	W1/0	W0/0	W0/0
W1 = { 'W', 'WXW' }	W2/0	W3/0	W0/0
W2 = { 'WW', 'WWW' }	W2/1	W3/0	W0/0
W3 = { 'WX' }	W1/1	W0/0	W0/1

A questo punto dobbiamo codificare gli stati e i simboli di ingresso (i simboli di uscita appartengono ad un alfabeto binario per cui li consideriamo già codificati). Ad esempio potremmo codificare

Stato	**Codifica**
W0	'00'
W1	'01'
W2	'10'
W3	'11'

Simbolo	**Codifica**
W	'00'
X	'01'
Y	'10'

Dobbiamo anche scegliere quali flip-flop usare per rappresentare gli stati.
Usiamo un JK per il bit più significativo ed un SR per il meno significativo.
Abbiamo 2 bit per gli stati più 2 bit per i simboli e quindi la tabella di transizione quindi avrà 16 righe.
Si noti che l'input '11' non potrà mai apparire, per cui le ultime 4 righe della tabella hanno solo dont' care.
Per riempire le colonne JK e SR si usano le tabelle inverse di transizione dei due tipi di flip-flop, che indicano quali segnali mandare in ingresso se si vuol ottenere la transizione desiderata, ovvero.

y	**Y**	**J**	**K**	**S**	**R**
0	0	0	-	0	-
0	1	1	-	1	0
1	0	-	1	0	1
1	1	-	0	-	0

Quindi la tabella di transizione è:

x_1	x_0	y_1	y_0	Y_1	Y_0	**Z**	J_1	K_1	S_0	R_0
0	0	0	0	0	1	0	0	-	1	0
0	0	0	1	1	0	0	1	-	0	1
0	0	1	0	1	0	1	-	0	0	-
0	0	1	1	0	1	1	-	1	-	0
0	1	0	0	0	0	0	0	-	0	-
0	1	0	1	1	1	0	1	-	-	0
0	1	1	0	1	1	0	-	0	1	0
0	1	1	1	0	0	0	-	1	0	1
1	0	0	0	0	0	0	0	-	0	-
1	0	0	1	0	0	0	0	-	0	1
1	0	1	0	0	0	0	-	1	0	-
1	0	1	1	0	0	1	-	1	0	1
1	1	0	0	-	-	-	-	-	-	-
1	1	0	1	-	-	-	-	-	-	-
1	1	1	0	-	-	-	-	-	-	-
1	1	1	1	-	-	-	-	-	-	-

Esercizio 2C

I trasferimenti sono tutti separati, per cui si tratta di 5 trasferimenti singoli secondo lo schema circolare:

$$R0 \rightarrow R1 \rightarrow R3 \rightarrow R4 \rightarrow R2 \rightarrow R0$$

Rimane da realizzare il circuito di controllo che abilita ciascun registro destinazione in scrittura come segue:

- un sommatore per calcolare R3 + R4, del risultato si prende il LSB e lo si usa come abilitazione di R1
- un comparatore aritmetico per confrontare R0 ed R4, l'uscita '<' viene usata per abilitare R3
- un NOR tra i 3 bit meno significativi di R1 (ovvero R1 mod 8=0), per abilitare R0 quando sono tutti 0
- il bit più significativo (MSB) di R2 indica se è negativo, per abilitare R2
- un sommatore ed un comparatore per confrontare 3 = R2 – R1 ed abilitare R4

Compito D

Esercizio 1D

Le equazioni della parte combinatoria della macchina sequenziale da analizzare sono:

$$J = X \qquad K = \overline{y_0}\ XOR\ \overline{y_1}$$
$$S = y_0 \qquad R = \overline{y_0 + Z} = \overline{y_0}\,\overline{Z}$$
$$Z = X\ y_1$$

Quindi la tabella di verità della macchina è

X	y_0	y_1	J	K	S	R	Z	Y_0	Y_1
0	0	0	0	0	0	1	0	0	0
0	0	1	0	1	0	1	0	0	0
0	1	0	0	1	1	0	0	0	1
0	1	1	0	0	1	0	0	1	1
1	0	0	1	0	0	1	0	1	0
1	0	1	1	1	0	0	1	1	1
1	1	0	1	1	1	0	0	0	1
1	1	1	1	0	1	0	1	1	1

Otteniamo dalla tabella sopra la tabella di transizione dell'automa (indicando con Si lo stato che ha codifica i)

	0	**1**
S0	S0/0	S2/0
S1	S0/0	S3/1
S2	S1/0	S1/0
S3	S3/0	S3/1

Minimizzando l'automa

S1	X		
S2	0-1 (no) 2-1 (no)	X	
S3	X	0-3 (no)	X
	S0	**S1**	**S2**

Quindi l'automa è minimo
E la sequenza di stati che ottengo partendo da S0=00 se ricevo 0110101111 in input è:

S0--0-->S0--1-->S2--1-->S1--0-->S0--1-->S2--0-->S1--1-->S3--1-->S3--1-->S3--1-->S3

Esercizio 2D

Il trasferimento richiesto è da molti a molti, e dobbiamo traferire un solo valore, per cui possiamo usare sia una mesh che un trasferimento molti-1-molti che un bus.

Supponiamo di usare un trasferimento molti-1-molti (ovvero un fascio di MUX che seleziona la sorgente, seguito da un collegamento a tutte le destinazioni, di cui si abilita solo la destinazione giusta con un decodificatore).

Dobbiamo progettare:

- il circuito che fornisce al fascio di MUX il valore (i) di selezione della sorgente
  - i=(RS2+RS3) mod 4 si realizza con un sommatore dal quale prendiamo i soli due bit meno significativi del risultato

- il circuito che fornisce il valore di selezione (j) per il decodificatore che abilita la destinazione
  - si noti che per i compreso tra 0 e 3, j=3-i è lo stesso che negare tutti i bit di i, quindi basta usare un fascio di NOT per ottenere j e mandarlo al decodificatore
- il circuito di controllo che abilita la destinazione quando la condizione è verificata
  - servono due sommatori per calcolare RS0-RS1 ed RS2+RS3, più un comparatore per confrontare i risultati. L'uscita '<' del comparatore va messa in AND con i segnali che abilitano le destinazioni.