

PROVA SCRITTA DEL CORSO DI
CALCOLATORI ELETTRONICI
NUOVO ORDINAMENTO DIDATTICO (5 CFU)
07 Giugno 2011

NOME:

COGNOME:

MATRICOLA:

ESERCIZIO 1 (9 punti)

Progettare una rete sequenziale che presenti un ingresso X e un'uscita Z posta a 1 ogni volta che viene riconosciuta la sequenza 01101.

Si richiede:

1. (6 punti) il diagramma degli stati, la tabella di flusso e la tabella delle transizioni;
2. (3 punti) il calcolo delle forme minime delle variabili di eccitazione dei flip flop con le mappe di Karnaugh. Si usino flip flop JK. Calcolare anche la rete combinatoria per l'uscita Z.

ESERCIZIO 2 (8 punti)

Si implementi in Assembly MIPS una funzione che, dato in ingresso un numero intero non negativo Y (in \$4), scriva in \$5 il valore 0 se esso è pari, 1 altrimenti. Nell'implementare la funzione si faccia uso della funzione div(X,Y) che, ricevendo X in \$4 e Y in \$5, scrive nel registro \$6 il quoziente e in \$7 il resto della divisione X/Y.

ESERCIZIO 3 (9 punti)

Si consideri un calcolatore che dispone di una memoria principale di 256 Mbyte e di una memoria cache di 512 Kbyte. E' possibile accedere al singolo byte e la memoria è suddivisa in blocchi da 16 byte.

1. (4 punti) Spiegare, precisando il significato e la funzione dei diversi campi, come vengono interpretati gli indirizzi logici per recuperare l'informazione contenuta nella cache nel caso venga usata la modalità di indirizzamento:
 - a. diretto;
 - b. "associativo su insiemi", e ciascun insieme contenga due blocchi.
2. (5 punti) Si considerino le due parole di indirizzo rispettivamente 21A0x43 e 10y0F4D. Calcolare i valori delle cifre x e y tali per cui le due parole si trovino:
 - a. nella stessa "linea" di cache nel caso di indirizzamento diretto;
 - b. nello stesso insieme nel caso di indirizzamento associativo su insiemi a due vie.

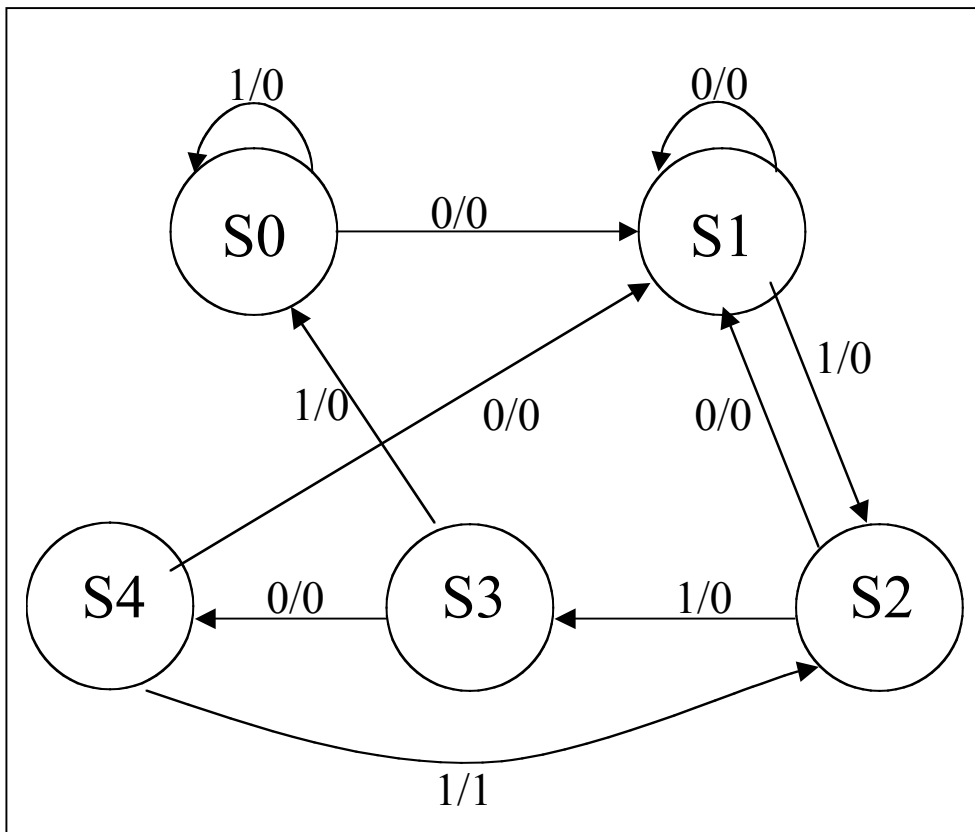
ESERCIZIO 4 (7 punti)

1. (5 punti) Si rappresentino i valori $(63.5)_{10}$ e $(31.25)_{10}$ nella forma $1.M * 2^E$ e li si sommi con l'algoritmo dei calcolatori.
2. (2 punti) Descrivere gli elementi di una ALU che effettua somme fra numeri in virgola mobile.

ESERCIZIO 1

Soluzione

Il diagramma degli stati è il seguente:



La tabella di flusso è data da:

Stato presente	Stato successivo/Uscita	
	X=0	X=1
S0	S1/0	S0/0
S1	S1/0	S2/0
S2	S1/0	S3/0
S3	S4/0	S0/0
S4	S1/0	S2/1

Per codificare tre stati occorrono due flip flop. La codifica è la seguente:
 $S0 \rightarrow 000$; ...; $S4 \rightarrow 100$. Nel seguito indicheremo ciascun bit della codifica con le lettere A, B, C. L'apice indicherà il bit nell'istante successivo a quello considerato.

A partire dalla tabella di eccitazione del flip flop JK:

Q	Q'	J	K
0	0	0	D
0	1	1	D
1	0	D	1
1	1	D	0

A	B	C	X	A'	Ja	Ka	B'	Jb	Kb	C'	Jc	Kc	Z
0	0	0	0	0	0	D	0	0	D	1	1	D	0
0	0	0	1	0	0	D	0	0	D	0	0	D	0
0	0	1	0	0	0	D	0	0	D	1	D	0	0
0	0	1	1	0	0	D	1	1	D	0	D	1	0
0	1	0	0	0	0	D	0	D	1	1	1	D	0
0	1	0	1	0	0	D	1	D	0	1	1	D	0
0	1	1	0	1	1	D	0	D	1	0	D	1	0
0	1	1	1	0	0	D	0	D	1	0	D	1	0
1	0	0	0	0	D	1	0	0	D	1	1	D	0
1	0	0	1	0	D	1	1	1	D	0	0	D	1
1	0	1	0	D	D	D	D	D	D	D	D	D	0
1	0	1	1	D	D	D	D	D	D	D	D	D	0
1	1	0	0	D	D	D	D	D	D	D	D	D	0
1	1	0	1	D	D	D	D	D	D	D	D	D	0
1	1	1	0	D	D	D	D	D	D	D	D	D	0
1	1	1	1	D	D	D	D	D	D	D	D	D	0

Ora possiamo disegnare le mappe di Karnaugh

		AB			
		00	01	11	10
CX	00			d	d
	01			d	d
	11			d	d
	10		1	d	d

$$J_A = BC\bar{X}$$

		AB			
		00	01	11	10
CX	00	d	d	d	1
	01	d	d	d	1
	11	d	d	d	d
	10	d	d	d	d

$$K_A = 1$$

		AB			
		00	01	11	10
CX	00		d	d	
	01		d	d	1
	11	1	d	d	d
	10		d	d	d

$$J_B = CX + AX$$

		AB			
		00	01	11	10
CX	00	d	1	d	d
	01	d		d	d
	11	d	1	d	d
	10	d	1	d	d

$$K_B = \bar{X} + C$$

		AB			
		00	01	11	10
CX	00	1	1	d	1
	01		1	d	
	11	d	d	d	d
	10	d	d	d	d

$$J_C = \bar{X} + B$$

		AB			
		00	01	11	10
CX	00	d	d	d	d
	01	d	d	d	d
	11	1	1	d	d
	10		1	d	d

$$K_C = B + X$$

L'uscita Z si ricava facilmente dalla tabella delle transizioni: $Z = \bar{A}\bar{B} \cdot \bar{C}X$.

ESERCIZIO 2**Soluzione**

```
pari_dispari:  addi $29, $29, -8
                sw  $5, 0($29)
                sw  $31, 4($29)
                addi $5, $0, 2
                jal  div
                move $5, $7
                lw  $31, 0($29)
                lw  $5, 4($29)
                addi $29, $29, 8
                jr  $31
```

ESERCIZIO 3

Soluzione

1. Per indirizzare 256 Mbyte occorre un indirizzo di almeno 28 bit. Per indirizzare il singolo byte all'interno di un blocco occorrono 4 bit ($16 = 2^4$), che coincidono con i 4 bit meno significativi dell'indirizzo di memoria primaria. I restanti 24 bit costituiscono l'indirizzo del "block frame". Per indirizzare la cache, il "block frame" viene interpretato diversamente a seconda che l'indirizzamento sia di tipo "diretto" o "associativo su insiemi".

(a) Indirizzamento diretto. In questo caso devo poter indirizzare ciascuno dei 32K blocchi contenuti nella cache ($512\text{Kbyte}/(16\text{byte}/\text{blocco})$). Occorrono 15 bit che coincidono con i 15 bit meno significativi del "block frame". Pertanto i 28 bit di indirizzo della memoria primaria vengono interpretati come:

tag	cache index	Offset
9 bit	15 bit	4 bit

(b) Indirizzamento "associativo su insiemi". In questo caso devo poter indirizzare ciascuno dei 16 insiemi in cui sono suddivisi i blocchi contenuti nella cache ($\frac{512\text{Kbyte}}{\frac{2\text{blocchi}}{\text{insieme}} \cdot \frac{16\text{byte}}{\text{blocco}}} = 16\text{K insiemi}$). Occorrono 14 bit che coincidono con i 14 bit meno significativi del "block frame"

Pertanto i 28 bit di indirizzo della memoria primaria vengono interpretati come:

tag	cache index	offset
10 bit	14 bit	4 bit

2. Perché le due parole si trovino nella stessa "linea" di cache o nello stesso insieme è necessario eguagliare l'index delle due parole. Faremo riferimento alla suddivisione in campi dell'indirizzo mostrato nella soluzione della prima parte dell'esercizio.

Nel caso di indirizzamento diretto avremo:

tag	cache index	offset
9 bit	15 bit	4 bit
001000011	0100000xxxx0100	0011
00010000y	yyy000011110100	1101

Cioè, in esadecimale, $x = F$, mentre y può essere pari a 2 od A perché la cifra più significativa è nel tag.

Analogamente, per il caso di indirizzamento associativo su insiemi a due vie, si ha:

tag	cache index	offset
10 bit	14 bit	4 bit
0010000110	100000xxxx0100	0011
00010000yy	yy000011110100	1101

Cioè, in esadecimale, $x = F$, mentre y può essere pari a 2, 6, A, E perché le due cifre più significative sono nel campo tag.

ESERCIZIO 4

Soluzione

1. Innanzi tutto rappresentiamo in virgola mobile i valori dati:

$$\begin{aligned}(63.5)_{10} &= 111111.1 = 1.111111 \cdot 2^5 \\ (31.25)_{10} &= 11111.01 = 1.111101 \cdot 2^4\end{aligned}$$

Poiché il primo ha esponente maggiore del secondo ($5 > 4$) di quest'ultimo si fa scorrere la mantissa a destra di una posizione, ovvero si moltiplica e divide per 2^1 .

Sommiamo ora le mantisse:

$$\begin{array}{r} 1.1111110 + \\ 0.1111101 = \\ \hline 10.1111011 \end{array}$$

Normalizzando la mantissa ottenuta, la somma si rappresenta:

$$1.01111011 \cdot 2^6 \rightarrow (93.75)_{10}$$

2. [cap.6, pag 39-40] La "floating point unit" (FPU) della ALU di un calcolatore può essere realizzata usando due unità aritmetiche in virgola fissa, una per l'esponente e una per la mantissa, che vengono accoppiate. L'unità per la mantissa si occupa di eseguire le operazioni aritmetiche di base sulla mantissa (somma algebrica, moltiplicazione e divisione). L'unità per l'esponente è più semplice in quanto deve eseguire solo operazioni di somma algebrica e di confronto fra numeri interi. Il confronto può essere effettuato attraverso una sottrazione degli esponenti.

Per eseguire la somma di due numeri in FP si fa la differenza degli esponenti. Il segno del risultato indica quale dei due esponenti è il più piccolo mentre il modulo indica il numero di scorrimenti verso destra che devono essere eseguiti sulla mantissa del numero più piccolo. Il registro che contiene la mantissa deve dunque essere del tipo a scorrimento. Gli scorrimenti possono essere pilotati da un contatore caricato con la differenza fra gli esponenti: ad ogni scorrimento il contatore viene decrementato finché non viene raggiunto il valore zero.