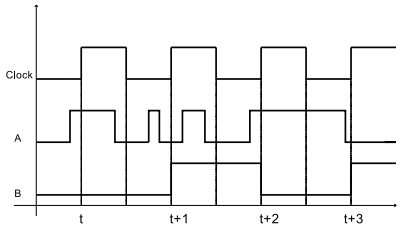
Corso di Architettura degli Elaboratori
Prova parziale del 17 Dicembre 2007
Tempo a disposizione: ore 2:00.

Si ricorda che:

- Per quanto possibile, occorre scrivere in bella calligrafia (il testo illeggibile non verrà preso in considerazione).
- Su tutti i fogli che vi abbiamo consegnato occorre riportare cognome, nome e numero di matricola.
- Occorre riportare in modo chiaro tutti i passi che portano alla determinazione del risultato.
- Il numero dell'esercizio che si sta svolgendo va sempre riportato in modo chiaro.
- Non è consentita la consultazione di appunti, libri, etc.
- Non è consentito l'uso di calcolatrici, telefoni cellulari, etc.
- Non è concesso chiedere alcunché ai docenti e agli altri studenti.

Esercizio 1. *(Punti 6)*
Si vuole costruire un circuito sequenziale sincrono con una singola entrata A e una singola uscita B. Il valore di B deve sempre essere identico al valore che A aveva in corrispondenza del *penultimo* fronte di salita del segnale di clock. Il diagramma temporale seguente mostra il comportamento atteso di B in funzione di un possibile andamento per A.

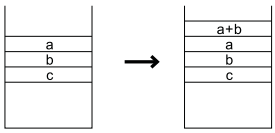


Nei primi due cicli di clock, B deve valere sempre 0 (indipendentemente dal valore di A). Si proceda come segue:

a) Si costruisca un diagramma di stato per il circurito sequenziale.
b) Si costruisca la relativa tabella di stato.
c) Si dica se il circuito che si otterrebbe dalla tabella di stato di cui al punto precedente è un circuito di Mealy oppure un circuito di Moore.

Esercizio 2. *(Punti 4, la risposta occupi al massimo 5 righe)*
Si parli dei principali problemi che si presentano nella progettazione dei chip di memoria.

Esercizio 3. *(Punti 4)*
Si scriva un frammento di codice IJVM che ha l'effetto di caricare in cima allo stack la somma delle due parole che si trovano in cima allo stack (le quali, però, *non* devono essere rimosse). Graficamente:



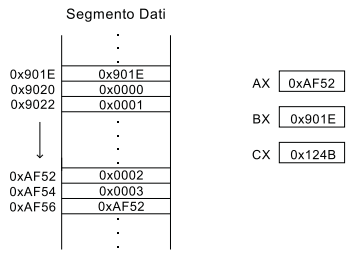
Si supponga di poter utilizzare un numero arbitrario di variabili locali.

Esercizio 4. *(Punti 6)*
L'istruzione `IADDWR` non esiste in IJVM. L'istruzione `IADDWR` ha lo stesso effetto del frammento di codice IJVM descritto nell'esercizio precedente. In altre parole, `IADDWR` calcola la somma delle due parole che si trovano in cima allo stack e mette il risultato in cima allo stack, senza rimuovere i due operandi. Si estenda il microcodice di Mic-1 con una sequenza di microistruzioni per `IADDWR`. Si valuti poi il miglioramento in performance indotto dall'adozione di `IADDWR` rispetto all'utilizzo del frammento di codice IJVM ottenuto nell'esercizio precente. A tal fine si faccia riferimento alla tabella seguente che elenca, per ogni istruzione IJVM, il numero di microistruzioni Mic-1 necessarie:

BIPUSH	4	IF_ICMPEQ	13	ISUB	4
DUP	3	IINC	7	LDC_W	8
GOTO	7	ILOAD	6	NOP	2
IADD	4	INVOKEVIRTUAL	23	POP	4
IAND	4	IOR	4	SWAP	7
IFEQ	11	IRETURN	9	WIDE ILOAD	10
IFLT	11	ISTORE	7	WIDE ISTORE	11

Esercizio 5. *(Punti 4, la risposta occupi al massimo 10 righe)*
Si parli dei principali fattori che influenzano la scelta del formato di istruzioni dei linguaggi del livello ISA, con particolare riferimento alla scelta della lunghezza delle istruzioni.

Esercizio 6. *(Punti 6)*
Si supponga che il segmento dati e i registri dati di una macchina 8088 abbiano l'aspetto seguente (ogni casella del segmento dati, nel disegno, rappresenta una parola a 16 bit, mentre il prefisso `0x` indica che si sta utilizzando la notazione esadecimale).



Si determini quale sia il valore dei registri `AX`, `BX` e `CX` dopo che uno dei due frammenti di codice Assembly seguenti (che sono equivalenti) è eseguito:

```
MOV SI,AX                    MOV SI,AX
MOV DI,2(BX)                 MOV DI,[2+BX]
MOV BX,4(SI)                 MOV BX,[4+SI]
MOV CX,2(BX)(DI)             MOV CX,[2+BX+DI]
MOV AX,(BX)                  MOV AX,[BX]
MOV BX,2(SI)                 MOV BX,[2+SI]
```