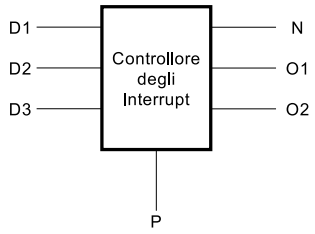
Corso di Architettura degli Elaboratori
Prova scritta del 3 Giugno 2008
Tempo a disposizione: ore 1:50.

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.
- Occorre consegnare anche la brutta copia ai docenti.

Esercizio 1. *(Punti 2, la risposta occupi al massimo 5 righe)*
Si parli della Macchina di Von Neumann. Ci si soffermi, in particolare, sull'influenza che questo modello ha nei calcolatori moderni.

Esercizio 2. *(Punti 9)*
Si vuole costruire un circuito combinatorio con quattro ingressi $D1$, $D2$, $D3$ e P e tre uscite $O1$, $O2$ e N. Più in dettaglio, si vuole progettare un controllore degli interrupt, di tipo molto semplice;



In particolare, le tre linee in input $D1$, $D2$ e $D3$ provengono da altrettanti dispositivi: se il dispositivo collegato a Dn vuole interrompere la CPU, setta Dn a 1 e in caso contrario setta Dn a 0. La linea in uscita N segnala se *almeno* un dispositivo vuole interrompere la CPU, mentre le due linee $O1$ e $O2$ segnalano *quale* dispositivo ha diritto ad interrompere la CPU, risolvendo eventuali conflitti utilizzando una tra le seguenti due politiche:
- Se l'ingresso P vale 0, allora si darà priorità al dispositivo con indice più basso (Dn avrà priorità maggiore di Dm ogniqualvolta $n < m$);
- Se l'ingresso P vale 1, allora si darà priorità al dispositivo con indice più alto.

Se il dispositivo selezionato è quello collegato a Dn, allora $O1$ e $O2$ saranno valorizzate in modo da rappresentare il numero n in binario puro ($O1$ è il bit meno significativo e $O2$ è il bit più significativo). Se $N = 0$ allora anche $O1$ e $O2$ valgono 0. Se per esempio $D1 = D2 = D3 = 1$ e $P = 0$, allora $N = 1$, $O1 = 1$ e $O2 = 0$.
Si proceda come segue:
a) Si costruisca una tabella di verità per $O1$, $O2$ e N in funzione di $D1$, $D2$, $D3$ e P.
b) Si costruiscano tre espressioni booleane in somma di mintermini per $O1$, $O2$ e N.
c) Si applichi a $O1$, $O2$ e N la procedura di semplificazione formale basata sulle mappe di Karnaugh e sugli implicanti primi essenziali, ottenendo un espressione in somma di prodotti.
d) Si costruisca un circuito combinatorio a due livelli a partire dalle espressioni ottenute al punto precedente.

Esercizio 3. *(Punti 3)*
Si costruisca un codice correttore di errore in modo tale che il codice comprenda almeno 8 parole di codice e che la distanza di Hamming tra esse sia almeno 2. Nel definire il codice si cerchi di minimizzare la lunghezza delle parole di codice.

Esercizio 4. *(Punti 8)*
In IJVM esiste l'istruzione

`IFEQ` *offset*

L'istruzione `IFEQ` è un'istruzione di salto. Il salto si effettua solo se la parola che si trova in cima allo stack è 0. Il parametro *offset* è uno spiazzamento (relativo al registro `PC`) che permette di determinare a quale locazione saltare. Il microcodice Mic1 che interpreta `IFEQ` è il seguente:

```
ifeq1   MAR=SP=SP-1;rd
ifeq2   OPC=TOS
ifeq3   TOS=MDR
ifeq4   Z=OPC; if (Z) goto T; else goto F
T       OPC=PC-1; goto goto2
F       PC=PC+1
F2      PC=PC+1;fetch
F3      goto Main1
goto1   OPC=PC-1
goto2   PC=PC+1;fetch
goto3   H=MBR«8
goto4   H=MBRU OR H
goto5   PC=OPC+H;fetch
goto6   goto Main1
```

Si consideri poi l'istruzione

`IFEQVAR`

che non esiste in IJVM. L'istruzione `IFEQVAR` si comporta in modo del tutto simile a `IFEQ`: realizza un salto condizionato e la condizione da verificare è la stessa, ossia l'uguaglianza tra la parola che si trova in cima allo stack (che chiameremo a) e 0. C'è però una piccola differenza: l'indirizzo dell'istruzione cui saltare (nel caso in cui la condizione appena descritta risulti verificata) si trova in cima allo stack, immediatamente sotto a. In questo modo il parametro *offset* non è più necessario. Si semplifichi il microcodice di cui sopra in modo che interpreti `IFEQVAR` anziché `IFEQ`.

Esercizio 5. *(Punti 3, la risposta occupi al massimo 10 righe)*
Si spieghi in che senso subroutine e coroutine sono diverse tra loro.

Esercizio 6. *(Punti 5)*
Si scriva una subroutine nel linguaggio assembly 8088 che, preso in input un argomento intero $n \geq 1$, stampi a video i primi $3n - 2$ valori interi. Per esempio, se $n = 2$, l'esecuzione della subroutine deve produrre in output sul terminale la seguente sequenza:

```
1
2
3
4
```

Il passaggio del parametro alla subroutine deve avvenire attraverso lo stack. Si ricordi che il codice della chiamata di sistema `PRINTF` è 127.