Άσκηση 4 - Απαντήσεις

Έλεγχος ροής εκτέλεσης – Εκτέλεση εντολών υπό συνθήκη

Β. ΠΕΙΡΑΜΑΤΙΚΟ ΜΕΡΟΣ

1. Δοκιμάστε και σχολιάστε το ακόλουθο πρόγραμμα:

Διεύθυνση	Εντολή
cs:0100	**mov al,0f0**
cs:0102	**inc al**
cs:0104	**jmp 0102**

Απάντηση

Στο παραπάνω πρόγραμμα, η εντολή **jmp 0102** επαναφέρει συνεχώς τη ροή εκτέλεσης στη διεύθυνση 0102 στην οποία γίνεται αύξηση κατά 1 του καταχωρητή al. Έτσι, αυξάνεται συνεχώς το περιεχόμενο του al. Όταν το περιεχόμενο φτάσει στην τιμή FF, τότε γυρνάει στο 00 μετά την επόμενη αύξηση, κ.ο.κ..

2. Δοκιμάστε το ακόλουθο πρόγραμμα:

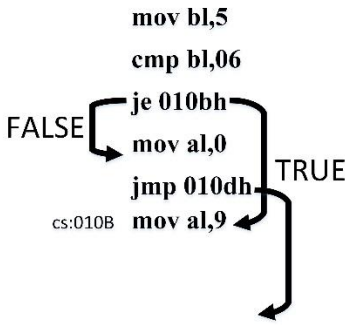
cs:0100	**mov bl,5**	Αν bl=06, τότε al=9, διαφορετικά al=0
cs:0102	**cmp bl,06**	
cs:0105	**je 010bh**	α) Σχολιάστε το πρόγραμμα
cs:0107	**mov al,0**	β) Σχεδιάστε διάγραμμα ροής
cs:0109	**jmp 010dh**	
cs:010B	**mov al,9**	

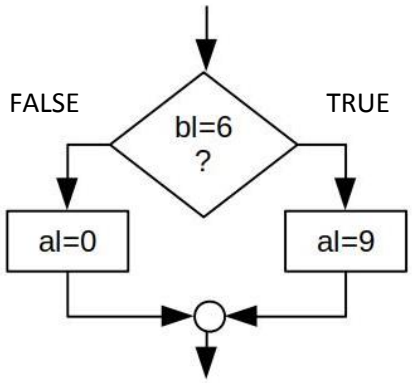
Απάντηση

Αρχικά, το περιεχόμενο του bl γίνεται 5 (βάζουμε διάφορες τιμές και δοκιμάζουμε το πρόγραμμα). Εμείς εστιάζουμε από τη σύγκριση και μετά. Η εντολή **cmp bl,6** συγκρίνει το περιεχόμενο του bl με το 6 (θυμηθείτε ότι, ο μικροεπεξεργαστής αφαιρεί τα δύο ορίσματα προκαλώντας την ενημέρωση ορισμένων bit του καταχωρητή κατάστασης, τα οποία αξιοποιεί η εντολή άλματος προκειμένου να προχωρήσει η ροή εκτέλεσης, βάσει του αποτελέσματος της σύγκρισης). Η εντολή **je** ελέγχει ισότητα (e=equal), δηλαδή αν **bl=6**. Αν αυτό ισχύει (συνθήκη true), τότε η ροή εκτέλεσης συνεχίζει από τη διεύθυνση 010b στην οποία εκχωρείται η τιμή 9 στο al. Αν όμως **bl≠6**, τότε δεν γίνεται άλμα στη διεύθυνση 010b και η ροή εκτέλεσης συνεχίζει από την επόμενη εντολή, δηλαδή την **mov al,0**. Σε αυτή την περίπτωση, το al γίνεται 0 αλλά θα πρέπει επίσης να αποφύγουμε την εντολή της διεύθυνσης 010b, αφού αναφέρεται στην περίπτωση true. Έτσι, με την εντολή **jmp** (άλμα χωρίς συνθήκη), παρακάμπτουμε το mov al,9 και φεύγουμε από αυτή την ενότητα προγράμματος.

Η λογική (αλγόριθμος) που υλοποιεί το παραπάνω πρόγραμμα είναι:

IF (BL=6) THEN
  AL=9
ELSE
  AL=0





3. Δίνεται το ακόλουθο πρόγραμμα:

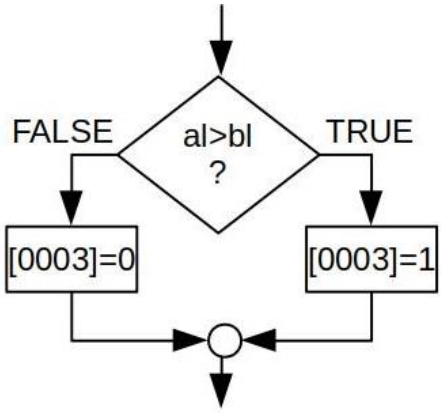
Πριν τη δοκιμή του προγράμματος, καταχωρήστε στο περιβάλλον του debugger τις τιμές 02 και 01 στις θέσεις μνήμης [0000] και [0001] αντίστοιχα.

```
cs:0100   mov al,[0000]
cs:0103   mov bl,[0001]
cs:0107   cmp al,bl
cs:0109   jg 0112
cs:010B   mov [0003],0
cs:0110   jmp 0117
cs:0112   mov [0003],1
```

α) Σχολιάστε το πρόγραμμα
β) Σχεδιάστε διάγραμμα ροής
γ) Δοκιμάστε το πρόγραμμα για διάφορες τιμές στις θέσεις μνήμης [0000] και [00001]

Απάντηση

Αρχικά, φορτώνεται το περιεχόμενο των θέσεων μνήμης [0000] και [0001] στους καταχωρητές al και bl αντίστοιχα. Στη συνέχεια, συγκρίνεται το περιεχόμενο των καταχωρητών (στην ουσία το περιεχόμενο των παραπάνω θέσεων μνήμης). Αν al>bl, τότε φορτώνεται στη θέση μνήμης [0003] ο αριθμός 1. Διαφορετικά (περίπτωση False), φορτώνεται το μηδέν και η ροή εκτέλεσης οδηγείται στην έξοδο (εκτός προγράμματος, διεύθυνση 0117).



Δοκιμάζοντας διάφορες τιμές, βλέπουμε την αντίστοιχη ροή εκτέλεσης με τη συνθήκη να είναι TRUE ή FALSE

4. Ποιο έλεγχο υλοποιεί το ακόλουθο πρόγραμμα;

```
mov al,[0000]
mov bl,[0001]
cmp al,1
je syneheia
jmp exit
syneheia:
        cmp bl,2
        je ok_label
        jmp exit
ok_label:
mov [0000],00
mov [0001],00
exit:
```

Απάντηση

Αρχικά, φορτώνεται το περιεχόμενο των θέσεων μνήμης [0000] και [0001] στους καταχωρητές al και bl αντίστοιχα. Το περιεχόμενο του al συγκρίνεται με το 1. Αν είναι ίσο, τότε προχωράμε και σε δεύτερο έλεγχο, αυτή τη φορά του bl με το 2. Αν και ο δεύτερος έλεγχος είναι TRUE, τότε η ροή εκτέλεσης οδηγείται στο ok_label με αποτέλεσμα να μηδενίζεται το περιεχόμενο των θέσεων μνήμης [0000] και [0001]. Αν οποιαδήποτε συνθήκη βρεθεί να είναι FALSE, τότε δεν γίνεται καμία αλλαγή στα περιεχόμενα των παραπάνω θέσεων μνήμης.

Το πρόγραμμα περιλαμβάνει τον έλεγχο δυο συνθηκών (αν al=1 και αν bl=2). Ο έλεγχος των συνθηκών είναι «συνδεδεμένος» με τη λογική AND, αφού πρέπει να είναι TRUE και οι δύο προκειμένου να μηδενιστεί το περιεχόμενο των αντίστοιχων θέσεων μνήμης.

Συνολικά, η λογική ελέγχου που υλοποιείται, είναι:

```
IF (AL=1 and BL=2) THEN
        [0000]=00
        [0001]=00
```