Άσκηση 6
Στοίβα – Πίνακες

Α. ΘΕΩΡΗΤΙΚΟ ΜΕΡΟΣ

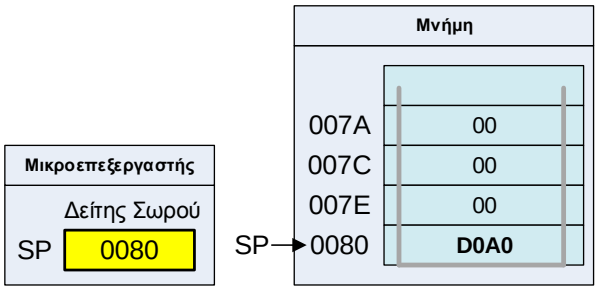
Σωρός ή Στοίβα

Ο σωρός είναι ένας τρόπος οργάνωσης δεδομένων που υλοποιείται μέσω του λογισμικού στην κεντρική μνήμη. Κάθε φορά που προωθείται κάτι στο σωρό, αυτό αποθηκεύεται πάντα στην κορυφή. Από την κορυφή επίσης γίνεται και η ανάκτηση. Για αυτούς τους λόγους, λέμε ότι στο σωρό ισχύει η λογική LIFO (Last In First Out), δηλαδή ότι αποθηκευτεί τελευταίο είναι αυτό που θα ανακτηθεί πρώτο.

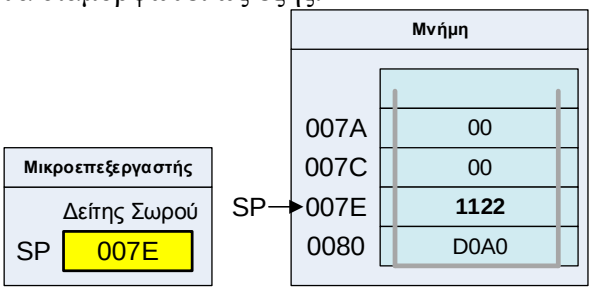
Η ορθή λειτουργία του σωρού προϋποθέτει την ύπαρξη ενός ειδικού δείκτη που θα δείχνει κάθε φορά την κορυφή του (SP-ΔΣ-Δείκτης Σωρού). Έτσι, θα είναι δυνατή η αποθήκευση και η ανάκτηση δεδομένων. Εφόσον ο σωρός υλοποιείται στη μνήμη, ο ΔΣ θα περιέχει πάντα κάποια διεύθυνση.

Στους μικροεπεξεργαστές INTEL, ο ΔΣ δείχνει το τελευταίο δεδομένο που έχει αποθηκευτεί στο σωρό και όχι την πρώτη ελεύθερη θέση στην οποία μπορεί να γίνει η επόμενη αποθήκευση. Αυτό σημαίνει ότι, πρώτα ενημερώνεται ο ΔΣ και στη συνέχεια γίνεται η αποθήκευση των νέων δεδομένων.

Ως παράδειγμα, θα υποθέσουμε την ακόλουθη μορφή σωρού, με το SP να δείχνει στη διεύθυνση 0080.



Η προώθηση δεδομένων στο σωρό γίνεται με την εντολή PUSH και την επιλογή ενός καταχωρητή. Αν AX=1122 και δοθεί η εντολή PUSH AX, τότε ο σωρός θα διαμορφωθεί ως εξής:



Η ανάκτηση δεδομένων γίνεται πάντα από την κορυφή του σωρού χρησιμοποιώντας την εντολή POP. Αν για παράδειγμα δοθεί τώρα η εντολή POP BX, τότε δύο byte από την κορυφή του σωρού θα αποθηκευτούν στον καταχωρητή BX. Έτσι, μετά την εντολή POP BX, θα ισχύει ότι BX=1122 και ο δείκτης σωρού θα δείχνει τώρα ξανά τη διεύθυνση 0080.

Οι εντολές που θα χρησιμοποιήσουμε για τη διαχείριση του σωρού είναι:

- ***PUSH καταχωρητής*** (π.χ. PUSH AX), προώθηση στην κορυφή του σωρού του περιεχομένου του καταχωρητή
- ***POP καταχωρητής*** (π.χ. POP BX), ανάκτηση από την κορυφή του σωρού και αποθήκευση σε καταχωρητή
- ***PUSHF***, προώθηση στο σωρό του περιεχομένου του καταχωρητή κατάστασης
- ***POPF***, ανάκτηση από το σωρό και ενημέρωση του καταχωρητή κατάστασης

Θα πρέπει να σημειωθεί ότι, σε όλες τις λειτουργίες (προώθηση, ανάκτηση), πάντα χρησιμοποιούνται 16bit δεδομένα.

Πίνακες

Η μνήμη του υπολογιστή (π.χ. Data Segment) αποτελεί ένα μονοδιάστατο πίνακα, αφού κάθε μοναδική διεύθυνση αντιστοιχεί σε μια θέση μνήμης.

Σε προηγούμενες ασκήσεις, διαβάζαμε ή γράφαμε στη μνήμη. Για παράδειγμα, γινόταν διάβασμα με την εντολή *mov bl,[0000]* ή γινόταν εγγραφή (αποθήκευση) με την εντολή *mov [0000], bl*. Επίσης, είχαμε δει εντολή της μορφής *mov [BX], al*, όπου η διεύθυνση μπορούσε να καθοριστεί από το περιεχόμενο του BX. Άρα, έχουμε κάνει ήδη πίνακες!.

Ας υποθέσουμε τώρα ότι, ένας πίνακας 5 θέσεων ξεκινά από τη διεύθυνση [0000] και θέλουμε σε όλες αυτές τις θέσεις να αποθηκεύσουμε την τιμή FF. Πώς θα γίνει αυτό;

Αν αρχικά AL=FF και μπορούμε να γράψουμε σε μια θέση μνήμης με την εντολή *mov [BX],AL*, πώς θα αναπτυχθεί το πρόγραμμα; Απλά θα φτιάξουμε ένα βρόχο επανάληψης, με το BX να παίρνει τις τιμές 0000 έως 0004 (συνολικά 5 θέσεις).

Άρα, το πρόγραμμα θα έχει την ακόλουθη μορφή:

```
mov AL, 0FF          Σταθερή τιμή που θα αποθηκευτεί σε όλες τις θέσεις
mov BX,0000          Αρχική διεύθυνση από την οποία θα ξεκινήσει η αποθήκευση
Start:               Σημείο επαναφοράς βρόχου
        mov [BX],AL  Αποθήκευση του FF (περιεχόμενο AL) στην τρέχουσα θέση (περιεχόμενο BX)
        inc BX       Αύξηση του BX κατά 1
        cmp BX,4     Σύγκριση του BX με το 4 (επιθυμητές τιμές 0 έως 4)
        JLE Start    Όσο δεν φτάσαμε στην τελευταία θέση, επιστροφή στο σημείο Start
```

Β. ΠΕΙΡΑΜΑΤΙΚΟ ΜΕΡΟΣ

1. Πρόγραμμα που φορτώνει το περιεχόμενο του BX στο CX μέσω του σωρού, θέτοντας όμως αρχικά τη διεύθυνση κορυφής του σωρού σε 007E

```
mov sp,007E
mov BX,1122
push BX
pop CX
```

α) Δοκιμάστε και σχολιάστε τη λειτουργία του προγράμματος
β) Πώς μεταβάλλεται το περιεχόμενο του sp;

2. Πρόγραμμα που ανταλλάσσει τα περιεχόμενα των καταχωρητών AX και BX μέσω του σωρού

```
mov AX,0ffff
mov BX,1234
push AX
push BX
pop AX
pop BX
```

α) Δοκιμάστε και σχολιάστε τη λειτουργία του προγράμματος
β) Σχεδιάστε μπλοκ διάγραμμα που να δείχνει τη λειτουργία του προγράμματος

3. Πρόγραμμα που θέτει 1 την τιμή όλων των bit του καταχωρητή κατάστασης και μετά επαναφέρει την αρχική

```
mov AX,0ffff
pushf
push AX
popf
popf
```

Δοκιμάστε και σχολιάστε τη λειτουργία του προγράμματος

4. Πρόγραμμα που ενεργοποιεί το CF του καταχωρητή κατάστασης (θεωρώντας ότι, το CF αποτελεί το λιγότερο σημαντικό ψηφίο του καταχωρητή κατάστασης)

```
pushf
pop AX
or AX,1
push AX
popf
```

α) Δοκιμάστε και σχολιάστε τη λειτουργία του προγράμματος
β) Εξηγήστε τη λειτουργία της λογικής πράξης ως προς το επιθυμητό αποτέλεσμα

5. Διπλός βρόχος με χρήση δύο καταχωρητών (5x3=15 επαναλήψεις)

```
mov AL,5
start:
        mov BL,3
        start2:
                dec BL
                jnz start2
        dec AL
        jnz start
```

Δοκιμάστε και σχολιάστε τη λειτουργία του προγράμματος

6. Διπλός βρόχος με χρήση ενός καταχωρητή (5x3=15 επαναλήψεις)

```
mov AX,5
start:
        push AX
        mov AX,3
        start2:
                dec AX
                jnz start2
        pop AX
        dec AX
        jnz start
```

Δοκιμάστε και σχολιάστε τη λειτουργία του προγράμματος

7. Πρόγραμμα που υπολογίζει το άθροισμα των στοιχείων ενός πίνακα 5 θέσεων που ξεκινά από τη διεύθυνση [0000]. Το άθροισμα αποθηκεύεται στη θέση [0005].

```
mov CL,0
mov BX,0
start:
        mov AL,[BX]
        add CL,AL
        inc BX
        cmp BX,4
        jle start
mov [0005],CL
```

α) Καταχωρήστε στο Data Segment αριθμούς της επιλογής σας (θέσεις [0000] έως [0004])
β) Δοκιμάστε και σχολιάστε τη λειτουργία του προγράμματος

8. Πρόγραμμα που μετρά το πλήθος των μηδενικών στοιχείων ενός πίνακα 16 θέσεων που ξεκινά από τη διεύθυνση [0000]. Το πλήθος των μηδενικών αριθμών αποθηκεύεται στη διεύθυνση [0005]

```
mov CL,0
mov BX,0
start:
        mov AL,[BX]
        cmp AL,0
        jne cont (διεύθυνση 010Dh)
        inc CL
cont:
        inc BX
        cmp BX,0F
        jle start (διεύθυνση 0105)
mov [0005],CL
```

α) Καταχωρήστε στο Data Segment αριθμούς της επιλογής σας (θέσεις [0000] έως [0004])

β) Δοκιμάστε και σχολιάστε τη λειτουργία του προγράμματος

Πραγματική μορφή καταχωρητή κατάστασης

Bit→	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	--	--	--	--	OF	DF	IF	TF	SF	ZF	--	AF	--	PF	--	CF

-- Δεν χρησιμοποιείται