ΕΡΓΑΣΤΗΡΙΟ ΑΡΧΙΤΕΚΤΟΝΙΚΗΣ

Τριαντάφυλλος Πράππας ΑΜ:1067504

ΕΡΓΑΣΤΗΡΙΑΚΗ ΑΣΚΗΣΗ 3

1)Μελέτη καταχωρητή κατάστασης

	N	C	Z	V	ΣΧΟΛΙΑ
@11	0	0	0	0	Στην συγκεκριμένη γραμμή αποθηκεύουμε στον καταχωρητή R2 το άθροισμα των καταχωρητών R0 (δηλαδή R0+R0) από την εντολή mov R0, #94 ξέρουμε ότι το περιεχόμενο του R0 είναι το 94 σε δεκαεξαδική μορφή δηλαδή 5E. Άρα το αποτέλεσμα του ADDS πρέπει να είναι το BC(δεκαεξαδικό). Πρέπει να τονίσουμε ότι αν χρησιμοποιούσαμε σκέτο ADD θα έβγαινε ακριβώς το ίδιο αποτέλεσμα αυτό συμβαίνει επειδή δεν έχουμε κρατούμενο και αυτό φαίνεται και από το CPSR που έχει μόνο 0, διάδοση κρατουμένου παρατηρούμε πχ σε longwords όπως στην προηγούμενη εργαστηριακή άσκηση που αν είχαμε πχ να προσθέσουμε το FF000000 2 φορές θα έπρεπε να κρατήσουμε ένα κρατούμενο και να διαδοθεί στην επόμενη πράξη.
@12	0	0	0	0	Στην συγκεκριμένη γραμμή αποθηκεύουμε στον καταχωρητή R2 το άθροισμα των καταχωρητών R1 (δηλαδή R1+R1) από την εντολή mov R1, R0, LSR #1 ξέρουμε ότι το περιεχόμενο του R1 είναι το 94 σε δεκαεξαδική μορφή δηλαδή 5E αλλά ολισθημένο κατά μια θέση δεξιά άρα το 2F. Άρα το αποτέλεσμα του ADDS πρέπει να είναι το 5E(δεκαεξαδικό). Πρέπει να τονίσουμε ότι αν χρησιμοποιούσαμε σκέτο ADD θα έβγαινε ακριβώς το ίδιο αποτέλεσμα αυτό συμβαίνει επειδή δεν έχουμε κρατούμενο και αυτό φαίνεται και από το CPSR που έχει μόνο 0
@13	0	0	0	0	Στην συγκεκριμένη γραμμή αποθηκεύουμε στον καταχωρητή R2 το άθροισμα των καταχωρητών R1 και R2 δηλαδή το αποτέλεσμα πρέπει να είναι το 8D(δεκαεξαδικό). Πρέπει να τονίσουμε ότι αν χρησιμοποιούσαμε σκέτο ADD θα έβγαινε ακριβώς το ίδιο αποτέλεσμα αυτό συμβαίνει επειδή δεν έχουμε κρατούμενο και αυτό φαίνεται και από το CPSR που έχει μόνο 0
@19	0	1	0	1	Στην συγκεκριμένη γραμμή αποθηκεύουμε στον καταχωρητή R3 την αφαίρεση των καταχωρητών R0 και R2 (R0-R2) το οποίο είναι το 7fffffff. Παρατηρούμε ότι το C είναι 1 δηλαδή η αφαίρεση δεν δημιουργεί δανεικό (το δανεικό είναι το ίδιο με

					το κρατούμενο στην πρόσθεση). Και το V είναι 1 επειδή έχουμε υπερχείλιση από την αφαίρεση.
@20	1	0	0	0	Στην συγκεκριμένη γραμμή αποθηκεύουμε στον καταχωρητή R3 την αφαίρεση των καταχωρητών R0 και R1 (R0-R1) το R1 από την εντολή ADD θα γίνει R0 + #0X80 (δηλαδή 0x80000000 + 0x80 = 0x80000080). Άρα παρατηρούμε ότι η αφαίρεση αυτή έχει ως αποτέλεσμα αρνητικό αριθμό αυτό μπορούμε να το καταλάβουμε και επειδή ενεργοποιείται η σημαία N στο CPSR που υποδηλώνει ότι έχουμε αρνητικό αριθμό. (Πρέπει επίσης να καταλάβουμε ότι με το N είναι λες και μετατρέπουμε αυτό που είναι στον καταχωρητή σε αρνητικο προσθέτοντας μπροστά από το περιεχόμενο του καταχωρητή, F τα οποία μετατρέπουν τον αριθμό σε αρνητικό).
@21	0	1	0	0	Στην συγκεκριμένη γραμμή αποθηκεύουμε στον καταχωρητή R3 την αφαίρεση των καταχωρητών R0 και R1 πρέπει να προσέξουμε ότι με την εντολή RSBS δεν εκτελούμε κανονική αφαίρεση αλλά αντεστραμμένη δηλαδή είναι λες και κάνουμε SUB R3, R1,R0 (R1-R0). Το αποτέλεσμα της αφαίρεσης αυτή είναι το 0x00000080 επίσης στον CPSR η σημαία C έχει 1 που σημαίνει ότι δεν έχουμε δανεικό (δηλαδή 1 που να πρέπει να χρησιμοποιηθεί σε άλλη πράξη).

2) Προσπέλαση διαδοχικών θέσεων μνήμης

```
.arm
 .text
 .global main

 main:
 STMDB R13!, {R0-R12, R14}
 LDR R0,=Stor
 MOV R1, #0
 STR R1,[R0], #4
 ADD R2, R0, #20
 Loop:
 ADD R1,R1, #1
 STR R1, [R0], #4
 CMP R0, R2
 BCC Loop

 LDMIA R13!, {R0-R12, PC}

 .data
  Stor:
 .word 0, 0, 0, 0, 0, 0
```

Για τον συγκεκριμένο κώδικα αφού πρώτα εισάγουμε τους καταχωρητές που θα χρησιμοποιήσουμε (μπορεί στον κώδικα να έχουμε εισάγει όλους τους καταχωρητές σε περίπτωση που χρειαζόντουσαν αρκετοί αλλά αυτό δεν είναι απαραίτητο καθώς θέλουμε μόνο 3 καταχωρητές). Στην συνέχεια καλό θα ήταν πρώτα να εξηγήσουμε την ετικέτα values στην οποία έχουμε εισάγει 6 μη μηδενικά word, αυτό το κάνουμε γιατί πρέπει να εισάγουμε τους αριθμούς από το 0 εως το 5 (χρησιμοποιούμε word και όχι byte έτσι ώστε το πρόγραμμα να αποθηκεύει στην μνήμη όλους τους πιθανούς αριθμούς πχ το 200 θέλει 3 ψηφία για να αναπαρασταθεί, βέβαια για να εισάγουμε πχ τον αριθμό 6 θα πρέπει στην ετικέτα Stor να εισάγουμε άλλο ένα μηδενικό word). Ύστερα με την LDR εισάγουμε στον καταχωρητή R0 τη διεύθυνση που σηματοδοτεί η ετικέτα Stor και με την MOV εισάγουμε στον R1 την τιμή 0 η οποία με το #4 θα «πιάσει» την πρώτη θέση μνήμης (πρώτο μηδενικό word). Η ADD θα μας βοηθήσει πιο μετά να βγούμε από το loop, με το #20 ουσιαστικά θα μας επιτρέψει να γίνει το loop 5 φορές έτσι ώστε να εμφανίσουμε τους αριθμούς από το 1 εως το 5 και να «πιάσουν» τις υπόλοιπες θέσεις μνήμης. Στην συνέχεια μέσα στο loop κάνουμε μια ADD η οποία θα προσθέτει στον καταχωρητή R1 κάθε φορά το 1 δηλαδή την πρώτη φορά θα γίνει 0+1 =1 κ.ο.κ. Και με την STR η τιμή της R1 θα αποθηκεύεται κάθε φορά στην διεύθυνση R0 στο αμέσως επόμενο word (λόγο του #4). Τέλος κάνουμε μια CMP με την οποία θα βγούμε τελικά από το loop και με την LDMIA θα καθαριστούν οι καταχωρητές που χρησιμοποιήσαμε.

3) Υπολογισμός αριθμών Fibonacci

```
.arm
 .text
 .global main
main:
 STMDB R13!, {R0-R12, R14}
 LDR R0,=Stor
 MOV R1, #1
 ADD R4, R0, #24
Loop:
 ADD R3,R1, R2
 STR R3, [R0], #4
 MOV R1, R2
 MOV R2, R3
 CMP R0, R4
 BCC Loop
 LDMIA R13!, {R0-R12, PC}
 .data
 Stor:
 .word 0, 0, 0, 0, 0, 0
```

Για αρχή θα πρέπει να πούμε ότι η ακολουθία Fibonacci είναι η εξής 0, 1, 1, 2, 3, 5, 8, 13…. Όμως από την εκφώνηση καταλαβαίνουμε ότι θα πρέπει να ξεκινήσουμε από το 1(βέβαια σε αρκετά βιβλία επίσης παραλείπεται το 0). Ο παραπάνω κώδικας μοιάζει με τον προηγούμενο απλώς όπως λέει και η εκφώνηση θα έχουμε 3 καταχωρητές έναν για να κρατά το an, ένα ν για το an−1 και έναν για το an−2. Στην αρχή ορίζουμε τον R1 ίσο με 1 έτσι ώστε να δημιουργήσουμε τον πρώτο αριθμό της ακολουθίας. Στην συνέχεια μεσα στο Loop βάζουμε στην αρχή το ADD έτσι ώστε να γίνεται η πρόσθεση του an-1+an-2 με το που μπούμε στο loop έχουμε μόνο το R1=1 και R2=0 άρα στον R3 αποθηκεύεται η τιμή 1 και με το STR αποθηκεύουμε αυτή την τιμή στην θέση μνήμης που περιέχει ο R0. Ύστερα με τις MOV

μεταφέρουμε το περιεχόμενο του R2 στον R1 και του R3 στον R2 και έτσι όταν θα εκτελεσθεί ξανά το ADD θα βγεί το δεύτερο αποτέλεσμα την ακολουθίας που είναι πάλι 1 κ.ο.κ. μέχρι να εμφανιστούν οι πρώτοι 6 αριθμοί της ακολουθίας.