Πανεπιστήμιο Κρήτης | ΗΥ-225
Τμήμα Επιστήμης Υπολογιστών | Άνοιξη 2000
ΗΥ-225: Οργάνωση Υπολογιστών | 2000-02-25 – βδομάδα 4
Δ. Πνευματικάτος | φυλλάδιο 4

2η Σειρά Ασκήσεων

Προθεσμία έως Τετάρτη 8 Μαρτίου (βδομάδα 6)

Διαβάστε το κεφάλαιο 3 του βιβλίου και απαντήστε τις παρακάτω ερωτήσεις.

2.1: Συνθέστε τις παρακάτω εντολές, οι οποίες υπάρχουν σε άλλους υπολογιστές αλλά όχι στον MIPS, χρησιμοποιώντας μία ή περισσότερες εντολές του MIPS: *(i)* clear register – μηδένισε τον καταχωρητή \$rd, *(ii)* clear memory word indirect – μηδένισε τη λέξη μνήμης στην οποία δείχνει το περιεχόμενο μιας δοσμένης (σταθερής) διεύθυνση (ο compiler γνωρίζει τη διεύθυνση, και ας υποθέσουμε ότι αυτή χωράει σε 16 bits, π.χ. διεύθυνση=1424).

2.2: Στην § 3.5, είδαμε το βρόχο:

```
while ( save[i] == k )
    i = i + j ;
```

μεταφρασμένο σε Assembly κατά τέτοιο τρόπο ώστε σε κάθε ανακύκλωση να εκτελείται τόσο μία διακλάδωση υπό συνθήκη (branch), όσο και ένα πήδημα χωρίς συνθήκη (jump). Μόνον οι εξαιρετικά απλοϊκοί compilers όμως θα έφτιαχναν τέτοιο κώδικα. Ξαναγράψτε το βρόχο σε Assembly, ούτως ώστε μόνο ένα branch ή jump να εκτελείται σε κάθε ανακύκλωση.
Έστω ότι ο βρόχος εκτελείται 10 φορές. Πόσες εντολές συνολικά εκτελούνταν με τον παλαιό κώδικα, και πόσες συνολικά με τον νέο;

2.3: Έστω ότι η μεταβλητή `i` βρίσκεται στον καταχωρητή \$17, η μεταβλητή `j` στον καταχωρητή \$21 και οτι CONST σημαίνει μια αυθαίρετη σταθερή μέχρι και 16 bits. Δώστε τον κώδικα Assembly MIPS (μόνο κανονικές εντολές του MIPS, όχι ψευδοεντολές), που υλοποιεί τις εξής περιπτώσεις:

- (Α) `if ( i == j ) goto L1;` (ίσο)
- (Β) `if ( i != j ) goto L1;` (διάφορο)
- (Γ) `if ( i < j ) goto L1;` (μικρώτερο)
- (Δ) `if ( i <= j ) goto L1;` (μικρώτερο ή ίσο)
- (Ε) `if ( i > j ) goto L1;` (μεγαλύτερο)
- (Ζ) `if ( i >= j ) goto L1;` (μεγαλύτερο ή ίσο)
- (Η) `if ( i >= CONST ) goto L1;` (μεγαλύτερο ή ίσο)
- (Θ) `if ( i == CONST ) goto L1;` (ίσο)

2.4: Δείξτε την εντολή MIPS – ή την ελάχιστη δυνατή ακολουθία από εντολές MIPS – που αντιστοιχεί στην εξής εκχώρηση (γραμμένη σε C):

```
z[8] = z[13] + c;
```

όπου η μεταβλητή `c` βρίσκεται στον καταχωρητή \$13, ο δέ πίνακας (ακεραίων) `z[]` αρχίζει στη διεύθυνση μνήμης 4,000,000 (τέσσερα εκατομμύρια) δεκαδικό. Δώστε την εντολή ή τις εντολές σε Assembly καθώς και σε γλώσσα μηχανής, στο δυαδικό, δείχνοντας τα όρια των διαφόρων πεδίων της/των εντολής/λών. Χρησιμοποιήστε εντολές κατασκευής/χρήσης άμεσων σταθερών (immediates), και μην θεωρείστε ότι οι σταθερές που χρειάζεστε προϋπάρχουν σε κάποιο καταχωρητή ή στη μνήμη.

2.5: Γράψτε και τρέξτε στο προσομοιωτή spim/xspim μια υπορουτίνα η οποία θα υπολογίζει το άθροισμα, το ελάχιστο και το μέγιστο στοιχείο ενός πίνακα μή-αρνητικών ακεραίων. Η υπορουτίνα δέχεται δύο παραμέτρους, ένα δείκτη στην αρχή του πίνακα (π.χ. `x[]`) στον καταχωρητή $5 και το μέγεθος του πίνακα (σε θέσεις) στον καταχωρητή $6. Το άθροισμα επιστρέφετε στον καταχωρητή $2 και το ελάχιστο και το μέγιστο στοιχείο στην στοίβα στην πρώτη και δεύτερη θέση αντίστοιχα, δηλαδή εκεί που δείχνει ο stack pointer κατά την κλήση της υπορουτίνας το ελάχιστο και στην επόμενη (μεγαλύτερη) θέση το μέγιστο. Ο απαιτούμενος χώρος στην στοίβα έχει δεσμευτεί ήδη από την υπορουτίνα που καλεί την υπορουτίνα σας.

Θεωρήστε ότι όλοι οι καταχωρητές είναι της κατηγορίας "callee save" (εκτός φυσικά από τους καταχωρητές που κρατάνε ορίσματα και στους οποίους επιστρέφουμε τιμές).

Χρησιμοποιήστε τον σκελετό προγράμματος από το directory ˜hy225/ask2. Στο ίδιο directory θα βρείτε και οδηγίες για την χρήση του spim και του xspim. Παραδώστε αυτή και την επόμενη άσκηση (μαζί) ηλεκτρονικά χρησιμοποιόντας το πρόγραμμα ˜hy225/bin/submit. Οδηγίες για την ηλεκτρονική παράδωση στο αρχείο ˜hy225/ask2/README

Σύσταση: Γράψτε την υπορουτίνα για μία από τις τρείς περιπτώσεις (π.χ. για το άθροισμα μόνο), ελέγξτε την, και μόνο αφού βεβαιωθήτε οτι δουλεύει προσθέστε τον υπόλοιπο κώδικα που ζητάει η άσκηση.

2.6: Γράψτε και τρέξτε στο προσομοιωτή spim/xspim μια αναδρομική συνάρτηση η οποία θα υλοποιεί την συνάρτηση του Fibonacci:

$$f(n) = f(n-1) + f(n-2)$$

με οριακές τιμές: $f(0) = 0$, και $f(1) = 1$. Η συνάρτηση δέχεται το όρισμά της στον καταχωρητή $4 και επιστρέφει το αποτέλεσμά της στον καταχωρητή $2. Η συνάρτηση σας θα πρέπει να επιστρέφει την τιμή -1 για αρνητικούς αριθμούς. Θεωρήστε ότι όλοι οι καταχωρητές είναι της κατηγορίας "callee save" (εκτός φυσικά από τους καταχωρητές που κρατάνε ορίσματα και στους οποίους επιστρέφουμε τιμές).

Χρησιμοποιήστε τον σκελετό προγράμματος από το directory ˜hy225/ask2. Στο ίδιο directory θα βρείτε και οδηγίες για την χρήση του spim και του xspim. Παραδώστε την αυτή και την προηγούμενη άσκηση (μαζί) ηλεκτρονικά χρησιμοποιόντας το πρόγραμμα ˜hy225/bin/submit. Οδηγίες για την ηλεκτρονική παράδωση στο αρχείο ˜hy225/ask2/README