ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΤΟΜΕΑΣ ΤΕΧΝΟΛΟΓΙΑΣ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΥΠΟΛΟΓΙΣΤΩΝ
ΕΡΓΑΣΤΗΡΙΟ ΥΠΟΛΟΓΙΣΤΙΚΩΝ ΣΥΣΤΗΜΑΤΩΝ
www.cslab.ece.ntua.gr

24 Μαΐου 2005

ΠΡΟΗΓΜΕΝΑ ΘΕΜΑΤΑ ΟΡΓΑΝΩΣΗΣ ΥΠΟΛΟΓΙΣΤΩΝ

2η Σειρά Ασκήσεων (παράδοση μέχρι 14/6/05)

Οι κρυφές μνήμες χωρίς κλειδώματα (lockup-free caches) σχεδιάστηκαν για να επιταχύνουν το ρυθμό εκτέλεσης των εντολών στον επεξεργαστή, επικαλύπτοντας το χειρισμό των αστοχιών κρυφής μνήμης (cache misses) με την επεξεργασία άλλων εντολών ή άλλων αστοχιών κρυφής μνήμης. Στο πρόβλημα αυτό υποθέστε ότι η ιεραρχία μνήμης διαθέτει κρυφή μνήμη ενός επιπέδου, με μέγεθος cache line 64bytes και διάδρομο επικοινωνίας κύριας μνήμης με την κρυφή μνήμη (memory bus) 16bytes. Υποθέστε, επίσης, ότι ο χρόνος εύρεσης των ζητούμενων δεδομένων στη μνήμη είναι t_{miss} = 5 cycles, και ο χρόνος μεταφοράς δεδομένων στη κρυφή μνήμη είναι 1 cycle/16 bytes, δηλαδή για την μεταφορά ολόκληρης της cache line: $t_{transfer}$ = 4 cycles. Τέλος, δίνεται ότι στο σύστημα 1word = 4 bytes.

α) Εξετάζουμε την εκτέλεση του εξής κώδικα:

```
for (i = 0; i < 1000; i++) {
          a += A[i] + B[i];
```

Θεωρούμε ότι οι τιμές των μεταβλητών `4·i`, `a` βρίσκονται στους καταχωρητές `$s4`, `$t0` και οι διευθύνσεις των πινάκων `A` και `B` στους `$s1` και `$s2`. Το περιεχόμενο του καταχωρητή `$s4` είναι αρχικά 0. Ο αντίστοιχος κώδικας σε assembly είναι ο εξής:

```
Loop:     add  $t1, $s1, $s4
          add  $t2, $s2, $s4
          lw   $t3, 0($t1)
          lw   $t4, 0($t2)
          add  $t0, $t0, $t3
          add  $t0, $t0, $t4
          addi $s4, $s4, 4
          bne  $s4, $s5, Loop
Exit:
```

Στον πίνακα που ακολουθεί φαίνεται η ροή εκτέλεσης του παραπάνω βρόχου, υποθέτοντας ότι κανένα από τα στοιχεία των πινάκων `A` και `B` δεν βρίσκονται ήδη στην κρυφή μνήμη (`m` = miss latency, `t` = transfer latency, `A` = lw from A, `B` = lw from B, `a` = add/addi, `b` = branch). Επιπλέον, διαθέτουμε απλή κρυφή μνήμη.

											1									
cycles	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9
array A				m	m	m	m	m	t	t	t	t								
array B														m	m	m	m	m	t	t
Execute	a	a	A										B							

cycles	2										3									
	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9
array A																				
array B	t	t																		
Execute			a	a	a	b	a	a	A	B	a	a	a	b	a	a	A	B	a	…

Πόσοι κύκλοι απαιτούνται για την εκτέλεση των 1000 επαναλήψεων του `for loop`;

β) Θεωρείστε ότι ο σχεδιασμός της κρυφή μνήμης επιτρέπει την προώθηση στον επεξεργαστή της ζητούμενης λέξης αμέσως μόλις αφιχθεί στην κρυφή μνήμη, χωρίς να χρειάζεται να ολοκληρωθεί η μεταφορά ολόκληρης της cache line. Ωστόσο, μόνο μία αστοχία μπορεί να βρίσκεται σε εκκρεμότητα. Εξετάστε δύο ενδεχόμενα:
i) Πρέπει να ολοκληρωθεί η μεταφορά ολόκληρης της cache line στην κρυφή μνήμη. Μόνο τότε μπορεί να αποσταλεί η επόμενη αίτηση αστοχίας στη μνήμη.
ii) Αρκεί να αφιχθεί η ζητούμενη από τον επεξεργαστή λέξη, για να υπάρχει η δυνατότητα αποστολής της επόμενης αίτησης αστοχίας στη μνήμη.
Φτιάξτε από ένα διάγραμμα χρονισμού για κάθε μία από τις περιπτώσεις (βi) και (βii), όπως αυτό του παραπάνω πίνακα. Πόσοι κύκλοι απαιτούνται για την εκτέλεση των 1000 επαναλήψεων του `for loop`; Στην περίπτωση (βi) υπάρχει τρόπος να μειώσουμε τον αριθμό των κύκλων που απαιτούνται για την εκτέλεση του βρόχου με βελτιστοποίηση του κώδικα;

γ) Θεωρείστε ότι ο σχεδιασμός της κρυφή μνήμης επιτρέπει την προώθηση στον επεξεργαστή της ζητούμενης λέξης αμέσως μόλις αφιχθεί στην κρυφή μνήμη, χωρίς να χρειάζεται να ολοκληρωθεί η μεταφορά ολόκληρης της cache line, και επιπλέον την εκκρεμότητα πολλών αστοχιών. Φτιάξτε ένα διάγραμμα χρονισμού για αυτόν τον σχεδιασμό της κρυφής μνήμης, όπως αυτό του παραπάνω πίνακα. Πόσοι κύκλοι απαιτούνται για την εκτέλεση των 1000 επαναλήψεων του `for loop`;