Οργάνωση Υπολογιστών – Σετ Ασκήσεων #2 Ημερομηνία παράδοσης: 6-Ιουνίου-2021, 23:59

10

1. Έστω η οργάνωση του MIPS σε 1 κύκλο (χωρίς pipeline), όπου στην αρχή κάθε κύκλου ρολογιού μία νέα εντολή έρχεται στον επεξεργαστή και εκτελείται εξ'ολοκλήρου μέσα σε αυτόν τον ένα κύκλο. Τα ψηφιακά κυκλώματα για την ολοκλήρωση μιας εκτέλεσης έχουν καθυστέρηση 2000ps.

(α) Ποιό είναι το CPI αυτού του επεξεργαστή αν υποθέσουμε ότι κάθε εντολή load/store βρίσκει τις εντολές/δεδομένα στην κρυφή μνήμη εντολών/δεδομένων ?

(β) ποιό είναι το throughput σε εκατ. εντολές ανά δευτερόλεπτο?

10

2. Έστω η οργάνωση του MIPS σε 5-stage pipeline, όπου το πιο αργό από τα πέντε αυτά στάδια καθυστερεί 450ps.

(α) Ποιό είναι το CPI αυτού του επεξεργαστή αν υποθέσουμε ότι κάθε εντολή load/store βρίσκει τις εντολές/δεδομένα στην κρυφή μνήμη εντολών/δεδομένων, και αν υποθέσουμε ότι δεν υπάρχουν αλληλοεξαρτήσεις (ιδανικό pipeline)?,

(β) ποιό είναι το throughput σε εκατ. εντολές ανά δευτερόλεπτο?

(γ) ποιό είναι το speedup σε σχέση με τον επεξεργαστή στο (1) ερώτημα? Γιατί είναι μικρότερο από 5X?

15

3. Έστω η οργάνωση του MIPS σε 5-stage pipeline. Έστω ότι το pipeline δεν υποστηρίζει bypassing (forwarding). Δείξτε πως εκτελούνται σε κάθε στάδιο οι παρακάτω περιπτώσεις ακολουθιών εντολών. Πόσοι είναι οι κύκλοι καθυστέρησης (stall cycles) που πρέπει να εισηχθούν μεταξύ των συνεχόμενων εντολών?

α. add $1, $2, $3
   add $4, $1, $2

β. lw $1, 8($2)
   add $4, $1, $3

γ. lw $1, 8($2)
   sw $3, 8($1)

δ. lw $1, 8($2)
   sw $1, 8($4)

15

4. Έστω τρεις διαφορετικοί επεξεργαστές, CPU1, CPU2, CPU3, οι οποίοι εκτελούν το ίδιο σετ εντολών. Ο CPU1 έχει ένα ρυθμό ρολογιού 3GHz και CPI ίσο με 1.5, ο CPU2 έχει ρυθμό ρολογιού 2.5GHz και CPI ίσο με 1.0 και ο CPU3 έχει ρυθμό ρολογιού 4GHz και CPI ίσο με 2.2.

α. Ποιος επεξεργαστής έχει την καλύτερη επίδοση εκφρασμένη σε εντολές ανά δευτερόλεπτο?

β. Αν οι επεξεργαστές εκτελούν ένα πρόγραμμα σε 10 δευτερόλεπτα, βρείτε τον αριθμό των κύκλων και τον αριθμό των εντολών που απαιτούνται.

γ. Αν επιχειρήσουμε να μειώσουμε τον χρόνο εκτέλεσης κατά 30%, αυτό οδηγεί σε αύξηση κατά 20% στο CPI. Τι ρυθμό ρολογιού θα πρέπει να έχει ο κάθε επεξεργαστής για να πετύχουμε αυτήν την μείωση ?

50

5. Υλοποιήστε μία εφαρμογή σε C ή C++, η οποία διαβάζει ένα αρχείο που περιέχει προσβάσεις στην μνήμη, μία πρόσβαση ανά γραμμή στο αρχείο, με τη μορφή: "A 0x01020304", όπου A είναι είτε L, είτε S, δηλαδή Load ή Store, και ο αριθμός στο δεκαεξαδικό είναι η διεύθυνση μνήμης. Η εφαρμογή σας θα δέχεται σαν παραμέτρους (argv):
   i. Αριθμό sets στην cache (δύναμη του 2, πχ. 1, 2, 4, ... 1:πλήρως προσεταιριστική, ...)
   ii. Αριθμό blocks σε κάθε set (δύναμη του 2, πχ. 1: άμεσης απεικόνισης)
   iii. Αριθμό bytes σε κάθε block (με ελάχιστο το 4, δύναμη του 2)
   iv. Αλγόριθμο αντικατάστασης: lru (least-recently-used), fifo, ή random

- Προς το παρόν αγνοήστε τις πολιτικές εγγραφής στην μνήμη, write-allocate ή no-write-allocate, write-through ή write-back, δηλαδή όταν η εφαρμογή σας συναντήσει μία γραμμή που ξεκινά από S τότε την αγνοεί και διαβάζει την επόμενη.
- Επίσης στην περίπτωση που έχουμε κρυφή μνήμη άμεσης απεικόνισης, τότε η παράμετρος (iv), ο αλγόριθμος αντικατάστασης δεν έχει νόημα.

- Επέκταση της εφαρμογής ώστε να δέχεται ως παράμετρο και πολιτική εγγραφής στην μνήμη: write-through ή write-back. Αυτές οι πολιτικές θα μπορούσαν να συνδυαστούν με write-allocate ή no-write-allocate (https://en.wikipedia.org/wiki/Cache_(computing)).

Η εφαρμογή σας θα πρέπει να τυπώνει στατιστικά στην εξής μορφή:

```
Total loads:
Total stores:
Load hits:
Load misses:
Store hits:
Store misses:
```

Παραδοτέα: ο κώδικας της εφαρμογής και χαρακτηριστικά files εισόδου και αποτελέσματα της εφαρμογής.