

## Θέμα 1

- i. (1 μονάδα) Περιγράψτε συνοπτικά τις write-through και write-back πολιτικές εγγραφής σε caches. Ποια τα πλεονεκτήματα/μειονεκτήματα σε κάθε περίπτωση?
- ii. (0.5 μονάδες) Έστω ότι έχετε δύο επεξεργαστές, ο πρώτος με 5 στάδια pipeline και ο δεύτερος με 12 στάδια pipeline. Και στους δύο υπολογιστές υπάρχει ένας branch predictor με σχετικά χαμηλή ακρίβεια. Σε ποιον από τους δύο υπολογιστές θεωρείτε ότι είναι περισσότερο απαραίτητο να βελτιωθεί ο branch predictor και γιατί?
- iii. (1 μονάδα) Περιγράψτε εν συντομία ποια είδη pipeline hazards γνωρίζετε? Αναφέρεται τι είναι τα dependencies και ποια είδη dependencies γνωρίζετε?
- iv. (1 μονάδα) Τι επίδραση έχει η αύξηση του block size της cache σε κάθε μια από τις 3 κατηγορίες των cache misses και γιατί? Μπορείτε να σκεφτείτε μία επιπλέον κατηγορία από cache misses?
- v. (1.5 μονάδα) Τι σημαίνει pipeline, superpipeline & superscalar αρχιτεκτονικές? Δώστε σχηματικά τα διαγράμματα σε κάθε περίπτωση.
- vi. (0.5 μονάδες) Ποια η διαφορά του ILP από το IPC?
- vii. (bonus 0.5 μονάδες) Αναφέρετε εάν έχει νόημα (από την άποψη της αρχιτεκτονικής του συστήματος) να κατασκευαστεί μια cache με associativity ίσο με 3 (associativity που δεν είναι δύναμη του 2)?
- viii. (bonus 1 μονάδα) Οι σύγχρονες κάρτες γραφικών (π.χ. CUDA GPU) μπορούν να υποστηρίξουν ταυτόχρονα πολλά threads (εκατοντάδες χιλιάδες). Σε αυτό το χαρακτηριστικό βασίζεται η επιτυχία τους. Μπορείτε να εξηγήσετε γιατί?

## Θέμα 2

(1.5 μονάδα) Θέλουμε να συγκρίνουμε 2 επεξεργαστές με τα εξής χαρακτηριστικά: Ο πρώτος επεξεργαστής έχει 5 στάδια pipeline (IF, ID, EX, MEM, WB) ενώ ο δεύτερος έχει 4 (IF, ID, EX+MEM, WB) ενοποιώντας την φάση εκτέλεσης με την φάση προσπέλασης στην μνήμη σε ένα στάδιο. Επίσης ο δεύτερος επεξεργαστής έχει 50% μεγαλύτερο κύκλο ρολογιού. Το ποσοστό εντολών που παρεμποδίζονται από stalls είναι 5% λόγω branch stalls και 4% λόγω load stalls. Βοηθή ποιος επεξεργαστής είναι πιο γρήγορος και πόσο, εάν υποθέσουμε ότι στον δεύτερο επεξεργαστή εξαλείφονται τα load stalls.

420

2

*Ταρίχης/φυσία του pred.*

### Θέμα 3

(2 μονάδες) Δίνονται οι παρακάτω εντολές διακλάδωσης:

B1 : T, N, T, T, T, N, N, N, T, N, T, N, T, T, N, T, N

B2 : N, T, N, N, N, T, T, T, N, T, N, T, N, N, T, N, T

Έστω ότι τώρα έχετε στην διάθεσή σας έναν  $(m,n)$  global history predictor, όπου  $m,n = \{1,2\}$ . Ποιο μηχανισμό πρόβλεψης θα προτεινάτε για τη διακλάδωση B2 και γιατί (Τα branches του B2 εκτελούνται πάντα μετά από τα αντίστοιχα branches του B1) ?

### Θέμα 4

Εξετάζουμε την εκτέλεση του ακόλουθου βρόχου ο οποίος αντιμετωπίζει έναν  $256 \times 256$  πίνακα  $a$  και αποθηκεύει το αποτέλεσμα σε έναν πίνακα  $b$ .

```
for(i=0; i<256; i++)
  for(j=0; j<256; j++)
    b[i][j] = a[j][i];
```

Οι πίνακες περιέχουν στοιχεία κινητής υποδιαστολής διπλής ακρίβειας, μεγέθους 8 bytes το καθένα. Κάνουμε τις εξής υποθέσεις:

- Το πρόγραμμα εκτελείται σε έναν επεξεργαστή με μόνο ένα επίπεδο κρυφής μνήμης δεδομένων, η οποία αρχικά είναι άδεια. Η κρυφή μνήμη είναι πλήρως συσχετιστική (fully associative), έχει μέγεθος 16 KB, και έχει LRU πολιτική αντικατάστασης. Το μέγεθος του block είναι 64 bytes.

- Οι πίνακες είναι αποθηκευμένοι στην κύρια μνήμη κατά γραμμές. Επιπλέον, είναι "ευθυγραμμισμένοι" ώστε το πρώτο στοιχείο του καθενός να απεικονίζεται στην αρχή μιας γραμμής της κρυφής μνήμης.

- i. (2 μονάδες) Βρείτε το συνολικό miss rate για τις αναφορές που γίνονται στην μνήμη στον παραπάνω κώδικα.
- ii. (bonus 1 μονάδα) Εφαρμόστε την τεχνική του blocking στον παραπάνω κώδικα παρουσιάζοντας τον βελτιστοποιημένο κώδικα. Ποιο block size θα επιλέγατε ως καταλληλότερο και γιατί;?

Καλή επιτυχία