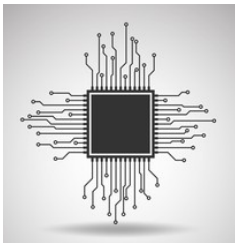
Εθνικο & Καποδιστριακο Πανεπιστημιο Αθηνων
Τμημα Πληροφορικης & Τηλεπικοινωνιων
Μαθημα: **Αρχιτεκτονικη Υπολογιστων ΙΙ**



Εργασία 2 (*υποχρεωτική*) – Κρυφές Μνήμες

Ακαδημαϊκο Ετος 2022 – 2023
(Εκφωνηση) Τριτη 6 Δεκεμβριου 2022
(Παραδοση στο eclass μεχρι) **Παρασκευη 20 Ιανουαριου 2023**

Επώνυμο	Όνομα	Αριθμός Μητρώου	Email
Ζερμπίνος	Ιωάννης	1115202000053	sdi2000053@di.uoa.gr
Ρούσσος	Βασίλειος	1115200700224	sdi0700224@di.uoa.gr
Κουτσουκής	Λεωνίδας	1115201500235	sdi1500235@di.uoa.gr

Πληροφορίες για τις Υποχρεωτικές Εργασίες του μαθήματος

- Οι υποχρεωτικές εργασίες του μαθήματος είναι **δύο**. Σκοπός τους είναι η κατανόηση των εννοιών του μαθήματος με χρήση αρχιτεκτονικών προσομοιωτών. Η πρώτη υποχρεωτική εργασία αφορά τη διοχέτευση (pipelining) και η δεύτερη (αυτή) αφορά τις κρυφές μνήμες (cache memories).
- Οι δύο εργασίες είναι υποχρεωτικές και η βαθμολογία του μαθήματος θα προκύπτει από το γραπτό (60%), την εργασία της διοχέτευσης (20%), και την εργασία των κρυφών μνημών (20%).
- Κάθε ομάδα μπορεί να αποτελείται **από 1 έως και 3 φοιτητές** (η υποβολή να γίνει μόνο από έναν φοιτητή εκ μέρους όλης της ομάδας). Συμπληρώστε τα στοιχεία όλων των μελών της ομάδας στον παραπάνω πίνακα. Όλα τα μέλη της ομάδας πρέπει να έχουν ισότιμη συμμετοχή και να γνωρίζουν τις λεπτομέρειες της υλοποίησης.
- Για την εξεταστική Σεπτεμβρίου δεν θα δοθούν άλλες εργασίες. Το Σεπτέμβριο εξετάζεται μόνο το γραπτό.
- Σε περίπτωση αντιγραφής θα μηδενίζονται όλες οι ομάδες που μετέχουν σε αυτή.
- Η παράδοση της **Εργασίας Κρυφών Μνημών** πρέπει να γίνει **ηλεκτρονικά μέχρι τα μεσάνυχτα της προθεσμίας και μόνο στο eclass** (να ανεβάσετε ένα μόνο αρχείο zip ή rar με την τεκμηρίωσή σας σε PDF και τους κώδικές σας). **Μην περιμένετε μέχρι την τελευταία στιγμή – κάθε εργασία απαιτεί τον χρόνο της.**

Ζητούμενο

Το ζητούμενο της εργασίας είναι η **σχεδίαση μιας «οικογένειας» τριών μικροεπεξεργαστών** που να διαφέρουν στην απόδοση και το κόστος για την ίδια υπολογιστική εργασία. Η σχεδίαση και η αξιολόγηση των επεξεργαστών θα γίνει στον προσομοιωτή QtMips.

Η υπολογιστική εργασία είναι **ο πολλαπλασιασμός τετραγωνικών πινάκων τάξης n** (δηλαδή διαστάσεων nxn) και η αναζήτηση ενός αριθμού στον τελικό πίνακα-γινόμενο των δύο αρχικών πινάκων. Τα στοιχεία των αρχικών πινάκων είναι μικροί προσημασμένοι ακέραιοι με τιμές στο διάστημα [–64, +63]. Το πρόγραμμά σας πρέπει να εκτελεί δύο ξεχωριστά βήματα:

(α) Δεδομένων δύο τετραγωνικών πινάκων X και Y διαστάσεων nxn να υπολογίζει το γινόμενο τους στον πίνακα Z (τα στοιχεία των πινάκων μπορούν να έχουν μέγεθος όσα byte θέλετε). Να εξετάζεται το ενδεχόμενο υπερχείλισης κατά τη διάρκεια του υπολογισμού του Z και να τερματίζει εάν συμβεί.

(β) Στον πίνακα Z να αναζητά έναν (οποιονδήποτε) ακέραιο K και να αποθηκεύει στην μεταβλητή C το πλήθος των εμφανίσεών του.

Οι τρεις επεξεργαστές έχουν κοινή σχεδίαση στον πυρήνα του pipeline: διαθέτουν branch predictor των 2-bit με BHT που προσπελάζεται με 5 bit και επίλυση των διακλαδώσεων στο στάδιο EX και πλήρη μονάδα κινδύνων με προώθηση. Διαφέρουν στο σύστημα της μνήμης.

Ο πρώτος επεξεργαστής (codename: *Qatar*) είναι μια φθηνή έκδοση της οικογένειας με το ελάχιστο δυνατό κόστος που σημαίνει ότι δεν χρησιμοποιεί κρυφή μνήμη (cache). Η προσπέλαση της κύριας μνήμης διαρκεί 40 κύκλους ρολογιού. Το κόστος του επεξεργαστή είναι 20 ευρώ και ο ρυθμός του ρολογιού του είναι 500 MHz.

Ο δεύτερος επεξεργαστής (codename: *Portugal*) είναι μια μεσαία έκδοση της οικογένειας που χρησιμοποιεί μόνο ένα επίπεδο κρυφής μνήμης (L1) για εντολές και δεδομένα. Η προσπέλαση της κύριας μνήμης και σε αυτόν τον επεξεργαστή διαρκεί 40 κύκλους. Οι κρυφές μνήμες εντολών/προγράμματος (L1 program cache στην ορολογία του QtMips) και δεδομένων (L1 data cache) έχουν μέγεθος 4, 8, ή 16 KB η καθεμία (ίδιο μέγεθος). Επίσης μπορούν να περιέχουν 4, 8, ή 16 λέξεις ανά μπλοκ η καθεμία (ίδιες επιλογές και για τις δύο). Μπορείτε να επιλέξετε όποια συσχετιστικότητα (associativity) επιθυμείτε (και πάλι ίδια για τις δύο μνήμες L1) και η

πολιτική αντικατάστασης να είναι LRU. Στην L1 data cache η πολιτική εγγραφής είναι ετερόχρονη (write back) και η πολιτική κατανομής είναι κατανομή σε εγγραφή (write allocate). Το κόστος αυτού του επεξεργαστή είναι 20, 25, ή 30 ευρώ υψηλότερο από τον προηγούμενο αν οι κρυφές μνήμες L1 έχουν μέγεθος 4, 8, ή 16KB καθεμία, αντίστοιχα. Ο ρυθμός ρολογιού είναι 500 MHz αν η συσχετιστικότητα είναι 1 (direct mapped) αλλά μειώνεται κατά 10 MHz για κάθε διπλασιασμό της συσχετιστικότητας.

Ο τρίτος επεξεργαστής (codename: *Argentina*) είναι μια προηγμένη έκδοση της οικογένειας που χρησιμοποιεί δύο επίπεδα κρυφής μνήμης (ξεχωριστή L1 για εντολές και δεδομένα και ενιαία L2). Η προσπέλαση της κύριας μνήμης και σε αυτόν τον επεξεργαστή διαρκεί 40 κύκλους και η προσπέλαση της L2 διαρκεί 5 κύκλους. Οι κρυφές μνήμες L1 είναι ίδιες με του προηγούμενου επεξεργαστή (θα κρατήσετε τη σχεδίαση στην οποία καταλήξατε – συνεπώς και το κόστος και τον ρυθμό ρολογιού). Η κρύφη μνήμη L2 έχει μέγεθος 16, 32, ή 64 KB και έχει ίδιο μέγεθος μπλοκ με τις L1 caches του προηγούμενου επεξεργαστή. Μπορείτε να επιλέξετε όποια συσχετιστικότητα επιθυμείτε για την L2 (δηλαδή μπορεί να είναι διαφορετική από των L1). Η πολιτική εγγραφής και κατανομής πρέπει να είναι ετερόχρονη (write back) και με κατανομή σε εγγραφή (write allocate) και η αντικατάσταση LRU. Το κόστος αυτού του επεξεργαστή είναι 50, 75 ή 100 ευρώ υψηλότερο από τον προηγούμενο αν η κρυφή μνήμη L2 έχει μέγεθος 16, 32, ή 64 KB, αντίστοιχα. Ο ρυθμός ρολογιού είναι ίσος με αυτόν του προηγούμενου επεξεργαστή.

- Να συμπληρώσετε τον παρακάτω πίνακα με **τα χαρακτηριστικά της σχεδίασής σας** για τον καθένα από τους τρεις επεξεργαστές.
- Να μετρήσετε **τον χρόνο εκτέλεσης** του προγράμματός σας στις τρεις σχεδιάσεις σας χρησιμοποιώντας τον αριθμό των κύκλων ρολογιού από τον QtMips και τον ρυθμό ρολογιού της σχεδίασής σας. Εάν χρησιμοποιήσετε διάφορα σύνολα δεδομένων να περιγράψετε τον τρόπο με τον οποίο παίρνετε τις μετρήσεις σας (μπορείτε να χρησιμοποιήσετε μέσες τιμές για παράδειγμα).
- Να υπολογίσετε **τον λόγο απόδοσης προς κόστος** του προγράμματός σας και στις τρεις σχεδιάσεις.
- Να εξετάσετε τα παραπάνω για **διαφορετικές τιμές του n** που να είναι δυνάμεις του 2 ξεκινώντας από n=16 και με όποιο μέγιστο n θέλετε.

Χαρακτηριστικό	**Πρώτος επεξεργαστής (*Qatar*)**	**Δεύτερος επεξεργαστής (*Portugal*)**	**Τρίτος επεξεργαστής (*Argentina*)**
L1 caches (μέγεθος, μέγεθος μπλοκ, συσχετιστικότητα)	-	8,8,1	8,8,1
L2 cache (μέγεθος, μέγεθος μπλοκ, συσχετιστικότητα)	-	-	16,16,2
Κόστος	20 ευρώ	45 ευρώ	95 ευρώ
Ρυθμός ρολογιού	500MHz	500MHz	500MHz
Κύκλοι εκτέλεσης	3992680	339139	70104
Χρόνος εκτέλεσης	0.007985 sec	0.0006783 sec	0.00014151 sec
Λόγος απόδοσης/κόστους	6.2617	32.7616	74.3857

Τεκμηρίωση

[Σύντομη τεκμηρίωση της λύσης σας ενδεικτικά (όχι αυστηρά) μέχρι **10 σελίδες ξεκινώντας από την επόμενη σελίδα** – μην αλλάζετε τη μορφοποίηση του κειμένου (**και παραδώστε την τεκμηρίωση σε αρχείο PDF**). Η τεκμηρίωσή σας πρέπει να περιλαμβάνει παραδείγματα ορθής εκτέλεσης και σχολιασμό για την επίλυση του προβλήματος και την επίτευξη του ζητούμενου. Μπορείτε να χρησιμοποιήσετε εικόνες, διαγράμματα και ό,τι άλλο μπορεί να βοηθήσει στην εξήγηση της δουλειάς σας.]

Σχολιασμός Κώδικα

- Στην αρχή ξεκινάμε φορτώνοντας τις διευθύνσεις των πινάκων και αρχικοποιώντας τους counters που θα χρησιμοποιήσουμε μετέπειτα.
- Μετά ξεκινάμε το τριπλό loop που κάνει ολόκληρη τη δουλειά.
- Στο loop3 ξεκινάμε με το να φορτώνουμε τα δύο πρώτα στοιχεία των δύο πινάκων και τα πολλαπλασιάζουμε.
- Μετά κάνουμε shift 31 θέσεις δεξιά το $t3 και το $t4, για να πάρουμε το 1ο bit που καθορίζει αν ο αριθμός είναι θετικός ή αρνητικός και κάνουμε xor για να δούμε αν είναι ίσα 2 δύο bit. Δηλαδή αν είναι και τα δύο αυτά αθροίσματα ομόσημοι ή όχι.
- Έπειτα προσθέτουμε τους δύο αριθμούς που φορτώσαμε και τους καταχωρούμε στο $t4 που κρατάει το άθροισμα για κάθε γραμμή * στήλη.
- Μετά ελέγχουμε αν τελικά οι αριθμοί είναι ομόσημοι ή όχι. Αν είναι δεν ελέγχουμε για overflow γιατί δεν μας ενδιαφέρει αυτή η περίπτωση αφού δεν θα συμβεί ποτέ.
- Αν όμως είναι, τότε κάνουμε shift 31 θέσεις δεξιά το τωρινό άθροισμα και το συγκρίνουμε με το προηγούμενο άθροισμα. Αν δεν είναι τα ίδια τα πρόσημα, τότε έχουμε overflow.
- Αυξάνουμε κατά 4 τον pointer για να πάει στο επόμενο element του πρώτου πίνακα.
- Εδώ όμως προσθέτουμε 64 για να πάει στο επόμενο στοιχείο της γραμμής αυτήν τη φορά του δεύτερου πίνακα.
- Αυξάνουμε τον counter για το loop3.
- Μόλις τελειώσει το loop3, αποθηκεύουμε τον αριθμό που προέκυψε από τις πράξεις στο matrix3, επαναφέρουμε την γραμμή του πρώτου - που έχει φτάσει στην θέση 16 - πάλι στην αρχή οπότε μειώνουμε 64(16*4) θέσεις στον pointer.

- Το ίδιο και για τις στήλες του δεύτερου πίνακα που ο pointer τώρα έχει φτάσει στην θέση 1024(16*64). Αλλά το επαναφέρουμε κατα 1020 θέσεις και όχι 1024 για να πάει στην επόμενη στήλη αυτήν την φορά και να κάνει την ίδια δουλειά του πολλαπλασιασμού.
- Αυξάνουμε counter για τις στήλες, επαναφέρουμε τον counter για τα elements, και του sum και τέλος, προσθέτουμε +4 στον pointer του matrix3 για να γίνει η επόμενη αποθήκευση.
- Όταν τελειώσουμε με τις στήλες, και έχουν πλέον επαναφερθεί τα υπόλοιπα που θέλουμε, αυξάνουμε τον pointer της γραμμής του πρώτου πίνακα κατά 64 για να πάει στην επόμενη γραμμή και να συνεχίσει η δουλειά από κει.
- Κάνουμε reset τις στήλες με la, επαναφέρουμε και τον counter των στηλών, και αυξάνουμε τον counter γραμμών κατά 1.
- Τέλος, φορτώνουμε έναν αριθμό που ορίζουμε να ψάχνει ο κώδικας και με ένα loop διατρέχουμε το matrix3 και άμα τον βρει αυξάνει τον counter του κατά 1.

Επίσης έχουμε στείλει και έναν κώδικα για την προσπάθεια που κάναμε για να κάνουμε την ίδια εργασία με blocking.