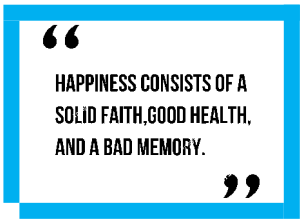

Εργασία 2 (*υποχρεωτική*) – Κρυφές Μνήμες

Ακαδημαϊκό Έτος 2019 - 2020

(Εκφώνηση) Παρασκευη 20 Δεκεμβριου 2019

(Παραδοση στο eclass μεχρι) **Παρασκευη 24 Ιανουαριου 2020**

Επώνυμο	Όνομα	Αριθμός Μητρώου	Email
Μαλέκας	Παντελεήμων	1115201600268	sdi1600268@di.uoa.gr
Πανταζή	Σωτηρία	1115201700241	sdi1700241@di.uoa.gr

Πληροφορίες για τις Υποχρεωτικές Εργασίες του μαθήματος

- Οι υποχρεωτικές εργασίες του μαθήματος είναι δύο. Σκοπός τους είναι η πλήρης κατανόηση των εννοιών του μαθήματος με χρήση αρχιτεκτονικών προσομοιωτών (architectural simulators). Η πρώτη υποχρεωτική εργασία αφορούσε τη διοχέτευση (pipelining) και η δεύτερη (αυτή) αφορά τις κρυφές μνήμες (cache memories).
- Η παράδοση των δύο εργασιών του μαθήματος είναι υποχρεωτική για τους φετινούς τριτοετείς φοιτητές καθώς επίσης και για όσους φοιτητές έχουν αριθμό μητρώου 2009 και μεταγενέστερο (δηλαδή πήραν το μάθημα για πρώτη φορά ως τριτοετείς από το εαρινό εξάμηνο του 2012 και μετά). Η βαθμολογία του μαθήματος προκύπτει από το γραπτό (60%), την εργασία της διοχέτευσης (20%), και την εργασία των κρυφών μνημών (20%). Καθένας από τους τρεις βαθμούς πρέπει να είναι προβιβάσιμος για να περαστεί προβιβάσιμος βαθμός στη γραμματεία. Οι προαιρετικές ασκήσεις που κατά καιρούς δίνονται δε βαθμολογούνται και δίνονται μόνο για να διευκολυνθεί η κατανόηση του μαθήματος.
- Για τους παλαιότερους φοιτητές (με αριθμό μητρώου 2008 και παλαιότερο) οι δύο εργασίες (pipeline, cache) είναι προαιρετικές. Αν κάποιος παλαιότερος φοιτητής δεν τις παραδώσει θα βαθμολογηθεί με ποσοστό 100% στο γραπτό. Αν τις παραδώσει, θα βαθμολογηθεί με τον παραπάνω τρόπο.
- Κάθε ομάδα μπορεί να αποτελείται **από 1 έως και 3 φοιτητές**. 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 της ομάδας.
- Για την εξεταστική του Σεπτεμβρίου δε θα δοθούν άλλες εργασίες. Το Σεπτέμβριο θα εξεταστεί μόνο το γραπτό.
- Σε περίπτωση αντιγραφής θα μηδενίζονται όλες οι ομάδες που μετέχουν σε αυτή.
- Η παράδοση της **Εργασίας Κρυφών Μνημών** πρέπει να γίνει μέχρι τα μεσάνυχτα της **Παρασκευής 24 Ιανουαρίου 2020** ηλεκτρονικά στο eclass (να ανεβάσετε ένα αρχείο 7z, zip ή rar με την τεκμηρίωσή σας σε Word ή PDF και τον πηγαίο κώδικά σας). Μόνο ένα από τα μέλη της ομάδας πρέπει να ανεβάσει την εργασία στο eclass, όχι καθένας ξεχωριστά.

Ζητούμενο

Γράψτε πρόγραμμα assembly MIPS για εκτέλεση στον προσομοιωτή EduMips64 και τον προσομοιωτή κρυφών μνημών DineroIV, το οποίο να εκτελεί τον παρακάτω απλό υπολογισμό:

- Εκτελεί την πράξη X=X+YZ, όπου X[i][j], Y[i][j], Z[i][j] τυχαίοι τετραγωνικοί πίνακες (8 byte κάθε ακέραιος – οι X, Y, Z αποθηκεύονται στο τμήμα δεδομένων του προγράμματος από την αρχή) με διαστάσεις 50x50 (δηλαδή i, j=0, 1, …, 49) και αφού ολοκληρώσει την πράξη X=X+YZ, υπολογίζει τον αριθμό των αρνητικών (n), μηδενικών (z) και θετικών (p) αριθμών. Στο τέλος εκτυπώνει τα τρία αυτά αποτελέσματα σε μία γραμμή.
- Ο χρόνος εκτέλεσης της εκτύπωσης του αποτελέσματος θα συνυπολογίζεται στο συνολικό χρόνο εκτέλεσης του προγράμματός σας.
- Ενεργοποιήστε την επιλογή του EduMIPS για την αυτόματη ανίχνευση υπερχείλισης (overflow) και μην γράψετε κώδικα σχετικά με την υπερχείλιση.

Φυσικά, το πρωταρχικό ζητούμενο είναι το πρόγραμμα να εκτελείται σωστά για οποιοδήποτε περιεχόμενο των πινάκων X, Y και Z. Πέρα όμως από την ορθή εκτέλεση πρέπει να ελαχιστοποιήσετε τον χρόνο εκτέλεσης του προγράμματός σας *γράφοντας κατάλληλα τον κώδικά* σας αλλά και *ρυθμίζοντας τις παραμέτρους* των κρυφών μνημών, υποθέτοντας ότι το σύστημα διαθέτει κρυφές μνήμες δύο επιπέδων, L1 και L2 Cache με βάση τα παρακάτω δεδομένα:

- Ο βασικός ρυθμός ρολογιού του μικροεπεξεργαστή είναι ίσος με 1GHz.
- Η κρυφή μνήμη του πρώτου επιπέδου (L1 Cache) μπορεί:
  - να είναι ενιαία ή ξεχωριστή για εντολές και δεδομένα
  - να έχει συνολικό μέγεθος (εντολές + δεδομένα) 8KB, 16KB ή 32KB
  - να έχει μέγεθος block 4, 8 ή 16 λέξεις (των 8 bytes)
  - να υποστηρίζει συσχετιστικότητα (associativity) Direct Map, 2-way, 4-way και 8-way.
- Η κρυφή μνήμη του δεύτερου επιπέδου (L2) μπορεί:
  - να έχει μέγεθος 64KB
  - να έχει μέγεθος block 8 ή 16 λέξεις (των 8 bytes)
  - να υποστηρίζει συσχετιστικότητα (associativity) μόνο 4-way.
- Ανάλογα με το μέγεθος της κρυφής μνήμης του πρώτου επιπέδου έχουμε και διαφορετική επιβάρυνση στον ρυθμό του ρολογιού. Δηλαδή:
  - για ξεχωριστές L1 Cache εντολών και δεδομένων μεγέθους 4KB καθεμία, ο ρυθμός ρολογιού του μικροεπεξεργαστή είναι ίσος με 1GHz
  - για ενιαία L1 Cache των 8KB ή για ξεχωριστές L1 Cache εντολών και δεδομένων μεγέθους 8KB καθεμία, ο ρυθμός ρολογιού του μικροεπεξεργαστή είναι ίσος με 950MHz
  - για ενιαία L1 Cache των 16KB ή για ξεχωριστές L1 Cache εντολών και δεδομένων μεγέθους 16KB καθεμία, ο ρυθμός ρολογιού του μικροεπεξεργαστή είναι ίσος με 900MHz
  - για ενιαία L1 Cache μεγέθους 32KB ο ρυθμός ρολογιού του μικροεπεξεργαστή είναι ίσος με 850MHz.
- Οι παραπάνω ρυθμοί ρολογιού ισχύουν για Direct Mapped συσχετιστικότητα, ενώ για κάθε διπλασιασμό της συσχετιστικότητας η επιβάρυνση του ρυθμού του ρολογιού είναι 2% σε σχέση με τον ρυθμό που έχει προκύψει από το μέγεθος της κρυφής μνήμης του πρώτου επιπέδου. Δηλαδή για παράδειγμα σε 4-way συσχετιστικότητα, ο ρυθμός του ρολογιού θα επιβαρυνθεί κατά 4%.
- Κάθε εντολή που εκτελείται διαρκεί 1 κύκλο ρολογιού αν δεν αστοχεί ούτε για εντολή ούτε για δεδομένα στην L1 Cache. Εάν όμως η εντολή αστοχήσει ή η αναφορά της σε δεδομένα αστοχήσει (miss) στη L1 cache, τότε επιβαρύνεται με ποινή 5/6/7 επιπλέον κύκλων ρολογιού αν το μέγεθος του μπλοκ είναι 4/8/16 λέξεις αντίστοιχα.
- Αντίστοιχα για την κρυφή μνήμη δεύτερου επιπέδου και ανάλογα με το μέγεθός του block της έχουμε και διαφορετική επιβάρυνση στην ποινή αστοχίας (που προστίθεται σε αυτή του πρώτου επιπέδου). Δηλαδή:
  - για μέγεθος block L2 Cache 8 λέξεων, η ποινή αστοχίας θα είναι 50 κύκλοι ρολογιού, ενώ
  - για μέγεθος block L2 Cache 16 λέξεων, η ποινή αστοχίας θα είναι 60 κύκλοι ρολογιού.
- Η πολιτική εγγραφής των κρυφής μνημών δεδομένων πρέπει να είναι write-back-allocate και για τα δύο επίπεδα κρυφής μνήμης.
- Ο αλγόριθμος αντικατάστασης πρέπει να είναι LRU και σε όλες τις κρυφές μνήμες (όταν φυσικά υπάρχει θέμα αντικατάστασης).

Συμπληρώστε τον παρακάτω πίνακα με τις ρυθμίσεις που επιλέξατε.

Ρυθμίσεις Κρυφή Μνήμης		
Ενιαία κρυφή μνήμη εντολών και δεδομένων 1ου επιπέδου (NAI/OXI)		NAI
L1 Instruction Cache	Μέγεθος	-
	Μέγεθος μπλοκ	-
	Associativity	-
L1 Data Cache ή ενιαία L1 Cache	Μέγεθος	32 KB
	Μέγεθος μπλοκ	8 λέξεις
	Associativity	Direct Mapped
L2 Cache	Μέγεθος μπλοκ	16 λέξεις

Συμπληρώστε τον παρακάτω πίνακα με τις πληροφορίες του τελικού σας προγράμματος (με βάση τα στατιστικά από τον προσομοιωτή και τα παραπάνω δεδομένα της εργασίας).

Συνολικό πλήθος εντολών που εκτελούνται	Συνολικοί κύκλοι ρολογιού για την πλήρη εκτέλεση του προγράμματος	CPI (*συνδυασμός EduMIPS+ Dinero)*	Ρυθμός ευστοχίας κρυφής μνήμης εντολών **L1**	Ρυθμός ευστοχίας κρυφής μνήμης δεδομένων ή ενιαίας **L1**	Ρυθμός ευστοχίας κρυφής μνήμης **L2**
1.076.155	1.147.931	1,0666967119	-	0.9956	0.9235

Τεκμηρίωση

[Σύντομη τεκμηρίωση της λύσης σας μέχρι 7 σελίδες – μην αλλάζετε τη μορφοποίηση του κειμένου. Η τεκμηρίωσή σας μπορεί να περιλαμβάνει παραδείγματα ορθής εκτέλεσης του προγράμματος, διαγράμματα και γενικά οποιονδήποτε σχολιασμό για την ανάπτυξη του κώδικά σας.]

Το πρόγραμμα το οποίο αναπτύχθηκε υπολογίζει την πράξη X = X + Y * Z όπου X, Y και Z είναι οι τρεις τυχαίοι τετραγωνικοί πίνακες με διαστάσεις 50X50. Επίσης το πρόγραμμα υπολογίζει πόσους θετικούς αριθμούς, πόσους αρνητικούς αριθμούς και πόσα μηδενικά έχει ο νέος πίνακας X και τα εκτυπώνει στο τέλος. Οι πίνακες έχουν οριστεί στο data τμήμα του κώδικα. Αποτελούνται από τυχαίους αριθμούς τύπου .word, αναγνωριστικό το οποίο αντιστοιχεί σε 8 byte. Θα παρατηρήσετε ότι αρκετοί αριθμοί στους πίνακες είναι ίδιοι. Επιλέξαμε αυτήν την σχεδιαστική επιλογή για να ολοκληρώσουμε σύντομα ό,τι αφορούσε την εκτέλεση στον Edumips, για να εστιάσουμε στις εκτελέσεις των κρυφών μνημών. Ωστόσο το γεγονός ότι οι αριθμοί επαναλαμβάνονται σε αρκετά σημεία δεν επηρεάζει την λειτουργικότητα του προγράμματος. Επίσης, ακριβώς μετά τον πίνακα X, έχει οριστεί το τελικό string που αφορά τους θετικούς, αρνητικούς και τα μηδέν, καθώς και τα ορίσματα τα οποία απαιτεί.

Στο τμήμα κώδικα, ξεκινάμε με τον υπολογισμό των αρχικών διευθύνσεων των Y και Z, οι οποίες είναι 20.088 και 40.088 αντίστοιχα. Αυτοί οι αριθμοί προκύπτουν επειδή κάθε πίνακας αποτελείται από 20.000 bytes και μετά από τον X έχουμε 88 bytes τα οποία αφορούν τις πληροφορίες του τελικού string. Επίσης στον r9 φορτώνουμε την τιμή 50, που είναι ο αριθμός γραμμής και στήλης για κάθε πίνακα, γιατί χρείαζεται σε διάφορα σημεία του κώδικα. Στην συνέχεια αρχικοποιείται ο καταχωρητής r16, ο οποίος χρησιμοποιείται για την αναπαράσταση του i από την πράξη X[i][j]=X[i][j] + Y[i][k] * Z[k][j], την πράξη που πρέπει να εκτελέσουμε δηλαδή. Αντιστοιχεί λοιπόν στην γραμμή των X και Y. Στην συνέχεια βλέπουμε το label L1 όπου αρχικοποιείται σε 0 ο καταχωρητής r17 που αναπαριστά το j, δηλαδή την στήλη των X και Z. Το L1 χρησιμοποιείται όταν έχει ολοκληρωθεί η απαιτούμενη πράξη σε κάθε στοιχείο από μια γραμμή του X και πρέπει να προχωρήσουμε στην επόμενη. Το j αρχικοποιείται σε 0 λοιπόν για υποδειξεί την πρώτη στήλη από μια νέα γραμμή. Ακολουθεί το label L2, όπου αρχικοποιείται σε 0 ο καταχωρητής r18 που αναπαριστά το k, δηλαδή την γραμμή του Z και την στήλη του Y. Το L2 χρησιμοποιείται όταν έχει ολοκληρωθεί η απαιτούμενη πράξη σε ένα στοιχείο του X και πρέπει να προχωρήσουμε στο επόμενο. Το k αρχικοποιείται σε 0 λοιπόν για υποδειξεί την πρώτη στήλη του Y και την νέα γραμμή του Z. Επίσης γίνονται οι απαραίτητες πράξεις για να βρούμε την θέση του επόμενου στοιχείου του X και το φορτώνουμε στον καταχωρητή r13. Ακολουθεί το label L3, που χρησιμοποιείται για να εκτελέσουμε την πράξη X[i][j] + Y[i][k] * Z[k][j]. Βρίσκουμε λοιπόν τις απαραίτητες θέσεις των στοιχείων Y και Z και τα φορτώνουμε στους καταχωρητές r15 και r14 αντίστοιχα. Αφού φορτωθούν αυξάνουμε τον καταχωρήτη r18, δηλαδή το k, μιας και έχουμε πάρει τα απαραίτητα στοιχεία των Y και Z. Εκτελούμε την πράξη Y[i][k] * Z[k][j], και το αποτέλεσμα της προστίθεται στον καταχωρητή r13, όπου περιέχεται το αντίστοιχο στοιχείο του X. Για να μειώσουμε τον χρόνο εκτέλεσης του προγράμματος εφαρμόσαμε την τεχνική του unrolling. Οπότε στο L3 το πεδίο κώδικα όπου φορτώνουμε τα στοιχεία των Y και Z, αυξάνουμε το k κατά 1 και εκτελούμε την πράξη X[i][j] + Y[i][k] * Z[k][j], έχει “ξετιλιχτεί” 25 συνολικά φορές. Άρα για να αλλάξει μία γραμμή του X αρκεί να εκτελεστεί 2 φορές το L3 label. Παρατηρούμε επίσης ότι για την φόρτωση των επόμενων στοιχείων των Y και Z, δεν χρείαζεται όπως στην αρχή του L3 να κάνουμε τις ίδιες πράξεις, μιας και αρκεί να αυξήσουμε κατά 400 τον r8, που τον χρησιμοποιούμε για να πάρουμε το επόμενο στοιχείο του Z (άρα και να αλλάξει γραμμή ο Z) και τον r11 να τον αυξήσουμε κατά 8, ο οποίος χρησιμοποιείται για την αντίστοιχη διαδικασία στον Y. Αφού γίνει λοιπόν η απαιτούμενη πράξη 25 φορές, ελέγχουμε αν ο r18 έχει την τιμή 50. Αν δεν είναι 50, ξαναπηγαίνουμε στην L3 για να κάνουμε την πράξη για τα υπόλοιπα στοιχεία των Y και Z. Αμά ο r18 είναι 50, αυτό σημαίνει ότι ολοκληρώσαμε την πράξη για όσα στοιχεία χρειαζόμασταν, άρα αποθηκεύουμε το καινούργιο περιεχόμενο του r13 στην αντίστοιχη θέση του X. Επίσης ελέγχουμε αν το νέο στοιχείο είναι θετικό, αρνητικό ή μηδέν και αυξάνουμε τους αντίστοιχους counter που θα χρειαστούν στην τελική εκτύπωση. Αφού γίνει αυτή η διαδικασία αυξάνεται ο r17, δηλαδή το j, κατά 1 για να προχωρήσουμε στην επόμενη στήλη του X με το L2, αν φυσικά ο καταχωρητής αυτός δεν είναι 50. Αν είναι 50 τότε έχουμε κάνει την πράξη για κάθε στοιχείο σε μια γραμμή του X, όποτε αυξάνουμε τον r16 κατά 1, δηλαδή το i. Αν ο r16 δεν έχει γίνει 50 πηγαίνουμε στην L1, για να κάνουμε τις απαραίτητες πράξεις για μια νέα γραμμή. Αν είναι 50 τότε έχουμε ολοκλήρωσει τις πράξεις για κάθε στοιχείο X. Αφού λοιπόν γίνουν όλες οι πράξεις στους πίνακες, αποθηκεύουμε τους

counters των θετικών, αρνητικών και μηδέν στα ορίσματα από το τελικό string και το εκτυπώνουμε με την χρήση του syscall 5.

Από τον Edumips παίρνουμε τα εξείς αποτελέσματα: 1.083.659 κύκλους, 1.076.155 εντολές και 1,006 CPI. Ο αριθμός κύκλων προκύπτει από των αριθμό των εντολών + τα instruction fetch που δεν ολοκληρώθηκαν λόγω branch εντολών + 4 λόγω του pipeline. Δεν υπάρχει κανένα stall μιας και έχει χρησιμοποιηθεί το Forwarding που παρέχει ο Edumips και εφαρμόσαμε αναδιάταξη εντολών όπου το θεωρήσαμε απαραίτητο. Επίσης για τα στοιχεία που έχουμε δώσει το output μας δίνει: Positive numbers: 1512, Zeros: 1, Negative numbers: 987. Αφού εκτελεστεί το πρόγραμμα μας στον Edumips παίρνουμε το trace file, έστω “Project2_Cache.s.xdin” και το χρησιμοποιούμε για να πάρουμε τα στοιχεία που χρειαζόμαστε από το Dinero.

Στο Dinero εκτελούμε το πρόγραμμα μας με την εντολή:

dineroiv -l1-usize 32K -l1-ubsize 64 -l1-uassoc 1 -l2-usize 64K -l2-ubsize 128 -l2-uassoc 4 -f "Project2_Cache.s.xdin"

Αφού δοκιμάσαμε κάθε συνδυασμό από τα δεδομένα που μας εδίνε η εκφώνηση στον Dinero, είδαμε ότι ο μικρότερος χρόνος με βάση τον τύπο: CPUtime = Cycles / ClockFrequency, ήταν 1.350.507 nsec. Οπότε επιλέξαμε αυτά τα δεδομένα. Για τον υπολογισμό των συνολικών κύκλων ρολογιού προσθέσαμε στους κύκλους από τον Edumips, τους κύκλους από τις αστοχίες (L1: (5932 misses)*6 λόγω του μεγέθους των block της L1, L2: (478 misses)*60 λόγω του μεγέθους των block της L2). Ο ρυθμός ρολογιού που προκύπτει από το μέγεθος της L1 cache με βάση τα δεδομένα της εκφώνησης είναι 0,85.

Επίσης, παρατηρήσαμε ότι ο ψηλός ρυθμός ρολογιού ήταν πιο σημαντικός παράγοντας στη μείωση του χρόνου εκτέλεσης, σε σχέση με το μέγεθος της cache. Επομένως επιλέξαμε μέτρια μεγέθη σε L1 cache και block. Επίσης, παρατηρήσαμε με το Dinero ότι με την ενιαία κρυφή μνήμη, για το συγκεκριμένο πρόγραμμα είχαμε καλύτερα αποτελέσματα όσον αφορά τις αστοχίες από ό,τι η ξεχωριστή για εντολές και δεδομένα, για αυτό επιλεξάμε την ενιαία.

Όσον αφορά τους ρυθμούς ευστοχίας στην L1 cache έχουμε 0,9956 (επειδή 1 – 0,0044, όπου 0,0044 ο ρυθμός αστοχίας σύμφωνα με τα αποτελέσματα του Dinero) και στην L2 έχουμε 0,9235 (επειδή 1-0,0765, για τον ίδιο λόγο με την L1). Για τον υπολογισμό του CPI προσθέσαμε στους κύκλους του Edumips, τους κύκλους από τις αστοχίες του Dinero, όπως αναφέραμε και προηγουμένως, και τους διαιρέσαμε με τον αριθμό των εντολών που μας έδωσε ο Edumips. Το CPI από την πράξη αυτή ήταν: 1,0666967119. Στον συνολικό αριθμό κύκλων τοποθετήσαμε τους κύκλους του Edumips + τους κύκλους από τις αστοχίες του Dinero, και για τον συνολικό αριθμό εντολών τοποθετήσαμε το αποτέλεσμα του Edumips.