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

Εργασία 1 (*υποχρεωτική*) – Διοχέτευση

Ακαδημαϊκο Ετος 2024 – 2025
(Εκφωνηση) Παρασκευη 15 Νοεμβριου 2024
(Παραδοση στο eclass μεχρι) **Τριτη 3 Δεκεμβριου 2024**

Επώνυμο – Όνομα	Πατρώνυμο – Μητρώνυμο	Αριθμός Μητρώου	Email
Μανίκας Παύλος	Παναγιώτης-Γεωργία	1115202200092	sdi2200092@di.uoa.gr
Κωσταρέλλου Βάια	Αριστείδης-Δήμητρα	1115202200085	sdi2200085@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 (αυτή) αφορά τη διοχέτευση (pipelining) και η δεύτερη θα 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 2 φοιτητές**. Συμπληρώστε τα στοιχεία των μελών της ομάδας στον παραπάνω πίνακα. Τα μέλη της ομάδας πρέπει να έχουν ισότιμη συμμετοχή και να γνωρίζουν τις λεπτομέρειες της υλοποίησης της ομάδας.
- Για την εξεταστική Σεπτεμβρίου δε θα δοθούν άλλες εργασίες. Τον Σεπτέμβριο εξετάζεται μόνο το γραπτό.
- Σε περίπτωση αντιγραφής θα μηδενίζονται όλες οι ομάδες που μετέχουν σε αυτή.
- Η παράδοση της **Εργασίας Διοχέτευσης** πρέπει να γίνει μέχρι τα **μεσάνυχτα της προθεσμίας ηλεκτρονικά** και **μόνο στο eclass από ένα μόνο μέλος της ομάδας** (να ανεβάσετε ένα μόνο αρχείο zip ή rar με την τεκμηρίωσή σας σε PDF και τον κώδικά σας). **Μην περιμένετε μέχρι την τελευταία στιγμή. Η εκφώνηση της εργασίας 2 των κρυφών μνημών θα αναρτηθεί αμέσως**.

Ζητούμενο

Υποθέστε ότι λειτουργείτε ένα κέντρο δεδομένων με 10 000 MIPS CPUs και σας ενδιαφέρει το **συνολικό κόστος ιδιοκτησίας (total cost of ownership – TCO)** του κέντρου δεδομένων. Το TCO για ένα ημερολογιακό έτος ορίζεται απλουστευτικά για τις ανάγκες της άσκησης ως: **κόστος αγοράς των τσιπ των CPU + δαπάνη ενέργειας για την ετήσια λειτουργία όλων των CPU**. Κάθε 12 μήνες αγοράζετε καινούριες CPU και ανακυκλώνετε τις παλιές. Ως ετήσια **υπολογιστική απόδοση** για το κέντρο δεδομένων ορίζεται το πλήθος των πλήρων εκτελέσεων του ενός και μοναδικού προγράμματος που εκτελούν συνεχώς όλες οι CPU στο κέντρο δεδομένων στη διάρκεια ενός έτους. Το ζητούμενο είναι να βελτιστοποιήσετε (μεγιστοποιήσετε) τον λόγο:

Υπολογιστική Απόδοση προς Συνολικό Κόστος Ιδιοκτησίας

Το πρόγραμμα που εκτελούν συνεχώς για 24 ώρες κάθε μέρα όλες οι CPU στο κέντρο δεδομένων είναι ένα πρόγραμμα (που πρέπει να γράψετε σε assembly MIPS) το οποίο αναζητά σε έναν μη ταξινομημένο πίνακα 200 προσημασμένων ακεραίων αριθμών A[i] (i = 0, 1, … 199) την εξής πληροφορία: (α) πόσοι από τους αριθμούς του πίνακα είναι αρνητικοί (N), (β) πόσοι είναι θετικοί (P), (γ) πόσοι είναι ίσοι με το μηδέν (Z), (δ) πόσοι είναι άρτιοι (E), (ε) πόσοι είναι περιττοί (O). Τα A, N, P, Z, E, O να είναι μεταβλητές στο τμήμα δεδομένων του προγράμματός σας. Δεν πρέπει να δίνονται αριθμοί από την κονσόλα, ούτε και να εκτυπώνεται τίποτε σε αυτή.

Οι εναλλακτικές σχεδιάσεις CPU MIPS που εξετάζετε είναι:

(α) MIPSa: επεξεργαστής διαδρομής δεδομένων ενός κύκλου ρολογιού με ρολόι 50 MHz, κόστος 5 ευρώ, και ηλεκτρική ισχύ 20 watt

(β) MIPSb: επεξεργαστής με διοχέτευση πέντε σταδίων αλλά χωρίς προώθηση (μόνο ανίχνευση κινδύνων δεδομένων και προσθήκη καθυστερήσεων για τη σωστή εκτέλεση) και με προσέγγιση καθυστέρησης σε διακλάδωση (stall on branch) και επίλυση διακλαδώσεων στο στάδιο EX – το ρολόι του είναι 150 MHz, κοστίζει 8 ευρώ, και δαπανά ισχύ 35 watt

(γ) MIPSc: επεξεργαστής με διοχέτευση πέντε σταδίων, πλήρη προώθηση, και πρόβλεψη διακλάδωσης με 2-bit, BHT των 5 bit, και επίλυση στο στάδιο ID – το ρολόι του είναι 150 MHz, κοστίζει 12 ευρώ, και δαπανά ισχύ 45 watt

Και οι τρεις επεξεργαστές έχουν ιδανικό σύστημα μνήμης και συνεπώς προσπέλαση εντολών και δεδομένων σε έναν κύκλο ρολογιού. Το κόστος της ενέργειας ανά kWh (κιλοβατώρα) είναι 0.002 euro.

Ποια από τις τρεις CPU θα δώσει για το πρόγραμμα που γράψατε τον καλύτερο λόγο: Υπολογιστική Απόδοση προς Συνολικό Κόστος Ιδιοκτησίας;

Ο επεξεργαστής που έχει τον καλύτερο λόγο υπολογιστικής απόδοσης προς συνολικό κόστος είναι ο MIPSb αφού οι λόγοι τους είναι

MipsA: 2.43*10^9
MipsB: 3.42*10^9
MipsC: 3.23*10^9

Τεκμηρίωση

[Σύντομη τεκμηρίωση της λύσης σας **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 του προγράμματος 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.]

Έγινε η επιλογή να χρησιμοποιηθεί το ίδιο πρόγραμμα και για τις 3 σχεδιάσεις επεξεργαστή αφού το πρόγραμμα είναι βέλτιστο για όλες. Στον σχεδιασμό 1 χρησιμοποιούμε τον ελάχιστο αριθμό εντολών και στους συνδυασμούς 2 και 3 έχουμε ελαχιστοποιήσει τον αριθμό των κινδύνων ελέγχου αποφεύγοντας την χρήση εντολών branch. Επίσης οι κίνδυνοι δεδομένων έχουν εξαλειφθεί μέσο της αλλαγής σειράς εκτέλεσης των εντολών. Λόγο της πλήρης προώθησης στον σχεδιασμό 3 θα μπορούσαμε να κάνουμε πιο απλές αλλαγές στον κώδικα όμως αυτό δεν έχει κάποιο όφελος στην απόδοση οπότε και πάλι χρησιμοποιήθηκε το ίδιο πρόγραμμα.

Όσον αφορά την δομή του προγράμματος:

Γίνεται μία επανάληψη 200 φορές για κάθε αριθμό στον πίνακα. Βρίσκουμε αν είναι περιττός ελέγχοντας το τελευταίο bit του αριθμού και ενημερώνουμε τον κατάλληλο μετρητή. Επίσης ελέγχουμε αν ο αριθμός είναι θετικός με την χρήση της εντολής slt και ενημερώνουμε τον μετρητή απευθείας χωρίς να κάνουμε branch. Το ίδιο γίνεται και για να μετρήσουμε τους αρνητικούς αριθμούς. Για τους άρτιους και τα μηδενικά δεν είναι απαραίτητο να τους μετρήσουμε στην επανάληψη. Αφού γνωρίζουμε το μέγεθος του πίνακα μπορούμε να αφαιρέσουμε από το 200 τους περιττούς και τους μη μηδενικούς αντίστοιχα.

Ο έλεγχος της ορθότητας του προγράμματος έγινε με την χρήση προγράμματος python με την ίδια λειτουργικότητα με αυτό που καλούμαστε να υλοποιήσουμε. Παρακάτω βρίσκονται τα αποτελέσματα της εκτέλεσης του προγράμματος και στις δύο γλώσσες:

Σημείωση: Στον κώδικα mips ο καταχωριτής $t0 περιέχει το πλήθος των αρνητικών, στον $t1 το πλήθος των θετικών, στον $t2 το πλήθος των μηδενικών, στον $t3 το πλήθος των περιττών και στο $t4 το πλήθος των άρτιων

Οι πίνακες που χρησιμοποιήθηκαν για τον έλεγχο του προγράμματος βρίσκονται στο τέλος του αρχείου

Array1

$8/t0	99	$9/t1	97	$10/t2	4	$11/t3	86	$12/t4	114

```
positive: 97  negative: 99  zero: 4  odd: 86  even: 114
```

Array2

$8/t0	104	$9/t1	96	$10/t2	0	$11/t3	89	$12/t4	111

```
positive: 96  negative: 104  zero: 0  odd: 89  even: 111
```

Array3

$8/t0	102	$9/t1	96	$10/t2	2	$11/t3	102	$12/t4	98

```
positive: 96  negative: 102  zero: 2  odd: 102  even: 98
```

Array4

$8/t0	105	$9/t1	95	$10/t2	0	$11/t3	95	$12/t4	105

```
positive: 95  negative: 105  zero: 0  odd: 95  even: 105
```

Array5

$8/t0	104	$9/t1	96	$10/t2	0	$11/t3	104	$12/t4	96

```
positive: 96  negative: 104  zero: 0  odd: 104  even: 96
```

Το κόστος του κέντρου δεδομένων για κάθε σχεδίαση επεξεργαστή είναι:

10000*(price_of_chip + watts*8760*0.002)

Για το mipsA έχουμε:

10000*(5 + 20*8760*0.002) = 3.554.000

Για το mipsB έχουμε:

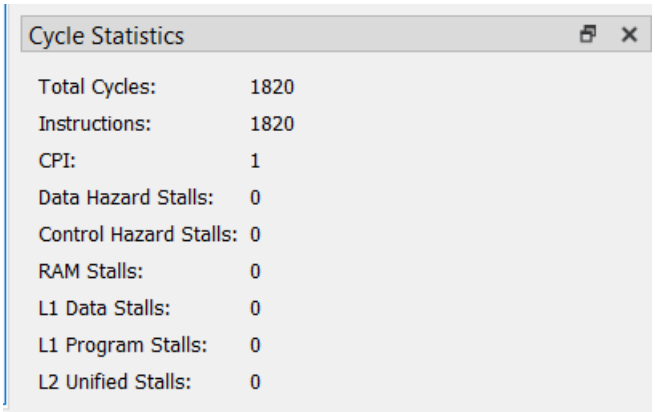
10000*(8 + 35*8760*0.002) = 6.212.000

Για το mipsC έχουμε:

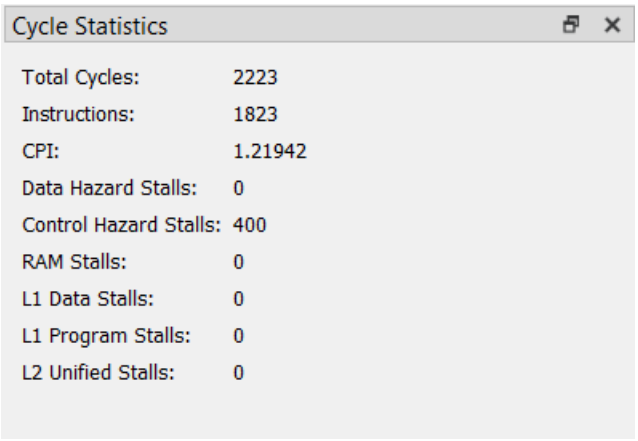
10000*(12 + 45*8760*0.002) = 8.004.000

Αφού τρέξουμε το πρόγραμμα και για τις 3 σχεδιάσεις CPU έχουμε τα cycle statistics της κάθε μιας τα οποία βρίσκονται παρακάτω

MIPSa:



MIPSb:



MIPSc:

Κάθε επεξεργαστής για να εκτελέσει το πρόγραμμα μία φορά χρειάζεται:

TotalCycles/ClockSpeed sec

Για το mipsA έχουμε:

1820/50mhz = 36.4 * 10^-6 sec

Για το mipsB έχουμε:

2223/150mhz = 14.82 * 10^-6 sec

Για το mipsC έχουμε:

1826/150mhz = 12.173 * 10^-6 sec

Άρα κάθε επεξεργαστής μπορεί σε έναν χρόνο να εκτελέσει το πρόγραμμα

(κάθε χρόνος έχει 365*24*3600 = 31536000 δευτερόλεπτα)

Για το mipsA:

31536000 sec/ 36.4 * 10^-6 sec = 866.37 * 10^9 φορές

Για το mipsB:

31536000 sec/ 14.82 * 10^-6 sec = 2127.9 * 10^9 φορές

Για το mipsC:

31536000 sec/ 12.173 * 10^-6 sec = 2590.6 * 10^9 φορές

Άρα η υπολογιστική απόδοση είναι:

10000*executions/cost

Για το mipsA:

10000*866.37*10^9/ 3.554.000 = 10000*0.243*10^9/10^3 = 2.43*10^9

Για το mipsB:

10000*2127.9*10^9/6.212.000 = 10000*0.342*10^9/10^3 = 3.42*10^9

Για το mipsC:

10000*2590.6*10^9/8.004.000 = 10000*0.323*10^9/10^3 = 3.23*10^9

Πίνακες:

array1 = [-89, 44, -69, -54, -56, -88, -42, -32, -94, -71, 0, -64, 73, 20, -44, -81, 67, -81, 100, -10, -12, 3, -3, -9, 72, -32, -65, 29, 34, -50, -77, 4, 13, 6, -69, 68, -60, -2, 7, 66, -46, -56, -39, -62, 10, -43, 38, -10, -62, -12, 2, 37, -76, -39, 27, -88, 94, -51, 70, 9, 68, -21, -55, 0, -13, -80, 96, -18, -31, 77, -9, 75, -46, 74, -76, -27, 84, 4, -46, -84, -34, 38, 48, -19, 22, -64, 76, -54, -56, -54, -20, -24, 76, -78, -82, -57, 90, 42, -84, -53, 22, -20, 20, 29, 59, -72, -75, 4, -33, 24, -54, 36, 53, -100, 96, -76, 95, 69, 100, 63, -2, 48, 51, 37, 99, 74, 22, -49, 25, 20, 77, 39, 0, 84, 23, 54, -23, -55, 41, 34, -80, -2, -98, 23, 28, 61, 55, -44, -22, 18, 97, -36, -32, 17, 70, 80, 9, -51, -99, -61, 69, 0, 39, 90, -12, -83, -19, -76, 83, 77, 35, -33, 31, 25, -99, -14, -3, -38, 43, -80, 39, -32, 44, 78, 35, 78, 5, 46, 42, -41, 1, 49, -87, 31, 22, 17, -81, -98, -58, -7]

array2 = [168, -130, 376, -452, 416, 97, 354, 236, 271, 222, 249, 53, -372, -311, -285, 224, 136, -430, 409, -255, 277, -302, -3, -426, -89, -420, -448, -401, 218, -488, -482, 85, 468, -21, -177, 250, 5, -346, 75, 488, 313, 462, 427, -272, -83, 189, -182, -356, 281, -142, 93, -269, -494, 213, -104, 215, -491, 68, 281, 84, -97, 302, -38, 36, -7, 231, -328, 436, -411, 443, 337, 426, -450, -367, 170, -7, 178, 177, 192, 456, 367, -380, 371, -188, -413, -22, 199, 32, 160, 329, 250, -372, -485, 454, -305, -442, 147, -486, -94, -54, -114, -166, 106, 215, -386, -383, -37, -213, -317, 313, -166, 438, 156, -283, 364, -103, -172, -335, -108, 323, -405, 38, -420, -247, -422, -481, 86, -18, 244, -111, -16, 477, 66, -328, 190, 40, -204, -163, -407, -284, -34, -217, -238, 381, 252, 496, -287, 316, 316, 443, -273, -219, 381, 452, -64, -75, -210, 280, -412, -1, 396, -311, 453, 222, 444, 49, -364, -86, -452, 369, -93, 456, -138, -204, 389, 388, -130, -452, -287, -119, 384, -210, 53, -459, 319, 367, 429, -183, 304, -335, 228, -224, -416, -421, 440, 127, 424, -465, 368, -152]

array3 = [81, 203, 122, -215, -68, -129, 464, -54, -152, -247, 119, -224, -483, 496, -103, -327, 44, 436, -237, -468, -415, 69, 397, 258, 240, -343, -261, 356, 273, 51, 196, 14, -18, -470, 498, -18, -305, 332, 85, 338, -172, -472, 480, -76, 252, 282, 179, -434, -383, 385, -102, -98, 111, -98, -151, 93, 282, 314, 137, 407, 285, 235, -344, 134, -110, -292, 115, -9, 38, 102, 45, -263, 423, -43, -300, 487, -160, 468, 23, -243, -321, 351, -222, -71, 167, -188, 474, 328, -180, 435, -219, 454, 160, -108, 447, 59, -399, 225, -383, -102, -230, -357, 136, -289, -12, -395, -182, 407, -237, -190, 52, 491, -238, 336, 103, 152, 433, 115, 186, -430, -456, 83, 256, -493, -290, -81, 195, -75, -241, -320, 228, -5, -493, -353, 101, -141, -150, 170, 493, -299, 431, -345, -429, -244, -406, -208, -474, -142, -452, 137, 211, 25, 168, -219, 245, 201, -36, -358, 180, 61, 379, -306, -239, -3, -323, 447, 397, -286, 0, 275, -21, -77, -214, -440, -329, 0, -185, -140, 142, 154, 456, 91, -290, -469, 143, 87, 476, -18, -200, 203, -5, 78, -232, 341, -454, -140, -331, 3, 7, 282]

array4 = [388, 54, 439, -168, -224, -103, -66, 494, 96, 97, -104, -308, -272, -480, -314, 171, -225, -418, 222, 273, -309, 209, 311, 368, 324, -84, 268, 325, 4, -110, -337, 249, 382, 19, -82, -127, 480, 496, 405, -311, 226, 349, -252, -470, -91, 203, -444, -113, -161, -489, -272, -55, -444, -374, -145, 478, 81, -411, 22, -156, -169, 361, -399, -275, 196, 188, 346, 298, 235, 415, -63, -434, -360, 27, -344, -272, 146, -173, 323, -91, -228, -379, 232, 72, -327, -161, 382, 487, -270, 40, -130, -491, 160, 47, 306, 95, -131, -236, -402, 439, -186, 352, -189, 466, -148, -7, -185, 380, 109, -401, -428, -245, 256, -183, -341, -142, -52, 451, -213, -388, -264, -332, -357, -208, 431, -303, -186, 266, -465, -497, -401, 257, -487, -280, 454, 128, 490, 88, -489, 148, -102, 190, -243, 37, 33, 156, 36, -159, -301, -308, 443, 209, 333, 302, -114, -38, -495, -393, 247, 378, 62, -295, -238, 125, 49, -69, -48, -455, 410, 420, 436, -174, -120, 363, 108, 47, 420, 162, 106, 13, 3, -474, 431, 84, -94, -381, 206, 327, 286, 62, -323, 42, -49, -314, -105, -477, -341, -497, 318, -169]

array5 = [-486, -319, -186, -397, -186, -409, -372, 445, -88, -323, -183, -38, 152, -450, -429, 216, -492, -174, 239, -170, -346, -367, -364, -338, -208, 462, -360, 299, 247, 146, 385, -464, 335, -157, -489, 198, -163, -325, -74, -424, 11, -309, 377, 482, -150, 95, -487, -492, 232, 292, 83, -420, -166, -300, 241, -1, 265, -158, -315, -14, -433, -229, 292, -153, 280, 123, -172, 346, 308, 108, 411, 7, 253, 171, -276, -55, -461, -350, 484, 38, 103, -411, 25, -493, 432, -15, 190, -223, -99, 314, -31, -392, 116, 116, 136, 326, -396, -72, -117, 492, -486, 455, 71, 22, -103, -390, 447, -68, 3, -389, -445, -117, -48, -320, -214, 449, 44, 410, 383, 212, 170, 261, 335, -204, -235, 268, -428, -330, -413, -101, 456, 465, -80, 116, 49, -253, -432, -373, -327, 432, -352, 428, -174, -323, 263, 379, 455, -170, 40, 239, -54, 163, -35, -340, 172, 480, -154, 313, 71, -431, 273, 448, 17, 423, 387, 153, -449, -223, 215, 355, -311, 313, 296, -472, 87, 177, 169, -10, 392, -74, 272, -219, -276, -209, 179, -391, -319, 83, -407, 466, 329, 117, -362, 31, 75, -310, -361, 162, 281, -299]