ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΤΟΜΕΑΣ ΤΕΧΝΟΛΟΓΙΑΣ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΥΠΟΛΟΓΙΣΤΩΝ
ΕΡΓΑΣΤΗΡΙΟ ΥΠΟΛΟΓΙΣΤΙΚΩΝ ΣΥΣΤΗΜΑΤΩΝ
www.cslab.ece.ntua.gr

ΑΡΧΙΤΕΚΤΟΝΙΚΗ ΥΠΟΛΟΓΙΣΤΩΝ

Κανονική Εξέταση Φεβρουαρίου 2013
Διάρκεια 2,5 ώρες

Οι εξετάσεις θα πραγματοποιηθούν ΧΩΡΙΣ την παρουσία βιβλίων, βοηθημάτων ή άλλου είδους σημειώσεων. Το μόνο που επιτρέπεται να έχετε μαζί σας είναι ένα φύλλο Α4 στο οποίο μπορείτε να έχετε γράψει ό,τι έχετε κρίνει πιο σημαντικό για το μάθημα και θέλετε να το έχετε ως βοήθημά σας. Απαγορεύεται η ανταλλαγή οποιουδήποτε αντικειμένου κατά την ώρα της εξέτασης, ούτε και των φύλλων Α4 που είναι ατομικά.

Θέμα 1ο (20%)

Α) Καλείστε να συμμετέχετε στην κατασκευή ενός καινούριου mp3 player και αναλαμβάνετε τη σχεδίαση του επεξεργαστή που χρησιμοποιεί το datapath του MIPS που είδαμε στο μάθημα. Μετά από μετρήσεις, ανακαλύπτετε τα παρακάτω για τα διαφορετικά κομμάτια του datapath:

- Οι αναγνώσεις/εγγραφές στο Register File απαιτούν 2ns.
- Οι λειτουργίες της ALU απαιτούν 3ns.
- Οι προσβάσεις στη μνήμη (αναγνώσεις/εγγραφές) απαιτούν 12ns.
- Οι εντολές άλματος υπό συνθήκη επιλύονται (και θέτουν τη σωστή τιμή στο PC) στο στάδιο EX.

Οι προγραμματιστές σας ενημερώνουν ότι το λογισμικό του συστήματος έχει τα εξής χαρακτηριστικά:

- 15% των εντολών είναι εντολές ADD
- 15% των εντολών είναι εντολές NAND
- 40% των εντολών είναι εντολές BEQ
- 20% των εντολών είναι εντολές LW
- 10% των εντολών είναι εντολές SW
- Το 30% των εντολών LW ακολουθείται από μια εξαρτώμενη εντολή (RAW hazard)

Αναλύστε τις τρεις παρακάτω πιθανές διαφορετικές υλοποιήσεις του επεξεργαστή:

- Single-cycle processor
- Multi-cycle processor
- Pipelined processor, όπου η προσθήκη των pipeline registers (IF/ID, ID/EX κτλ) έχει σαν αποτέλεσμα να απαιτούνται 2ns για την ανάγνωση των δεδομένων τους και 1 ns για την εγγραφή τους. Ταυτόχρονα, η εκτέλεση μιας εντολής άλματος υπό συνθήκη έχει ως αποτέλεσμα την εισαγωγή 2.65 stalls κατά μέσο όρο στο pipeline.

Για κάθε μία από τις παραπάνω υλοποιήσεις υπολογίστε τα εξής:

(i) CPI
(ii) Τη μέγιστη συχνότητα του ρολογιού του επεξεργαστή (σε MHz)
(iii) Το χρόνο εκτέλεσης (σε ms) ενός προγράμματος με 100000 εντολές.

	Single-cycle	Multi-cycle	Pipelined
CPI	1	Οι ADD, NAND, SW χρειάζονται 4 κύκλους. Οι BEQ 3 και η LW 5. Έτσι για το CPI έχουμε: $(0.15 + 0.15 + 0.1) * 4 + 0.4 * 3 + 0.2 * 5 = 3,8$	Το CPI ενός pipelined επεξεργαστή χωρίς εξαρτήσεις είναι 1. Εδώ stalls εισάγουν τα branches καθώς και το ποσοστό των εντολών LW που ακολουθούνται από εξαρτώμενη εντολή και άρα εμφανίζουν RAW hazard. Καθώς υπάρχουν σχήματα προώθησης τα RAW hazards αυτά εισάγουν 1 stall κάθε φορά. Έτσι θα έχουμε για το CPI: $1 + 0.4 * 2.65 + 0.2 * 0.3 * 1 = 2.12$
f_{MAX}	Ο κύκλος θα είναι ίσος με αυτόν της πιο αργής εντολής, δηλαδή 12+2+3+12+2 = 31ns. Άρα η μέγιστη συχνότητα θα είναι: $\frac{1}{31 * 10^{-9}} = \frac{1000}{31} MHz \approx 32.26MHz$	Ο κύκλος θα είναι ίσος με το χρόνο που απαιτεί το πιο αργό στάδιο, δηλαδή 12ns. Άρα η μέγιστη συχνότητα θα είναι: $\frac{1}{12 * 10^{-9}} = \frac{1000}{12} MHz \approx 83.33MHz$	Ο κύκλος θα είναι ίσος με το χρόνο που απαιτεί το πιο αργό στάδιο, δηλαδή 2+12+1 = 15ns. Άρα η μέγιστη συχνότητα θα είναι: $\frac{1}{15 * 10^{-9}} = \frac{1000}{15} MHz \approx 66.67MHz$
t_{exec}	$t_{exec} = CPI * instr * t_{cycle} = 1 * 10^5 * 31 * 10^{-9} = 3.1ms$	$t_{exec} = CPI * instr * t_{cycle} = 3.8 * 10^5 * 12 * 10^{-9} = 4.56ms$	$t_{exec} = CPI * instr * t_{cycle} = 2.12 * 10^5 * 15 * 10^{-9} = 3.18ms$

Β)Δώστε τους ορισμούς των WAW, WAR και RAW εξαρτήσεων. Εξηγήστε ποιοι από τους κινδύνους αυτούς εμφανίζονται και ποιοι όχι στην κλασσική αρχιτεκτονική σωλήνωσης 5 σταδίων του MIPS.

Ο WAW κίνδυνος εμφανίζεται όταν μια εντολή προσπαθεί να γράψει σε έναν καταχωρητή πριν γράψει στον ίδιο καταχωρητή μια προηγούμενη εντολή.
πχ. ADD R1, R2, R3
ADD R1, R4, R5
Στον MIPS ο κίνδυνος αυτός δεν εμφανίζεται γιατί εγγραφή επιτρέπεται μόνο σε ένα στάδιο και οι εντολές εκτελούνται πάντα in-order. Επομένως πάντα θα γράφει η πρώτη εντολή.

Ο WAR κίνδυνος εμφανίζεται όταν μια εντολή προσπαθεί να γράψει σε έναν καταχωρητή πριν τον διαβάσει μια προηγούμενη εντολή.
πχ. ADD R1, R2, R3
ADD R3, R4, R5
Στον MIPS ο κίνδυνος αυτός δεν εμφανίζεται γιατί ο MIPS εκτελεί τις εντολές in-order και οι εντολές διαβάζουν τα ορίσματα τους στο στάδιο ID, το οποίο προηγείται του σταδίου WB όπου γίνονται οι εγγραφές στους καταχωρητές. Επομένως πάντα θα διαβάζει τον καταχωρητή η πρώτη εντολή.

Ο RAW κίνδυνος εμφανίζεται όταν μια εντολή προσπαθεί να διαβάσει από έναν καταχωρητή πριν τον γράψει μια προηγούμενη εντολή.
πχ. ADD R1, R2, R3
ADD R5, R4, R1
Στον MIPS ο κίνδυνος αυτός εμφανίζεται και επιλύεται με χρήση forwarding και stalls.

Θέμα 2ο (30%)

Υποθέστε την κλασσική αρχιτεκτονική σωλήνωσης του MIPS αποτελούμενη από τα στάδια IF, ID, EX, MEM, WB. Όλα τα στάδια διαρκούν ένα κύκλο. Κατά τον εντοπισμό μιας εντολής άλματος υπό συνθήκη, ο επεξεργαστής κάνει stall τη σωλήνωση μέχρι την επίλυση η οποία πραγματοποιείται στο στάδιο EX. Τέλος, υποθέστε ότι η εγγραφή σε ένα καταχωρητή γίνεται στο πρώτο μισό ενός κύκλου, ενώ η ανάγνωση από τον ίδιο καταχωρητή πραγματοποιείται στο δεύτερο μισό του κύκλου.

Δίνεται το ακόλουθο κομμάτι κώδικα :

```
1.    LOOP: ADDI  $t5, $t5, #4
2.          LW    $t3, 8($t5)
3.          LW    $t4, 4($t3)
4.          SUB   $t1, $t3, $t4
5.          SUB   $t2, $t7, $t5
6.          SW    $t1, 200($t2)
7.          BNEZ  $t2, LOOP
```

Δίνεται επίσης ότι η αρχική τιμή του καταχωρητή $t7 είναι ίση με **$t5+156**.

Α) Υποθέστε ότι δεν υπάρχουν σχήματα προώθησης. Εκτελέστε την $1^{η}$ επανάληψη του βρόχου (μέχρι και τη $1^{η}$ εντολή της $2^{ης}$ επανάληψης) και χρησιμοποιήστε ένα διάγραμμα χρονισμού για να δείξετε τα διάφορα στάδια της σωλήνωσης από τα οποία διέρχονται οι παραπάνω εντολές. Πόσοι κύκλοι απαιτούνται για την εκτέλεση ολόκληρου του βρόχου;

Β) Υποθέστε τώρα ότι υπάρχουν όλα τα δυνατά σχήματα προώθησης. Δείξτε όπως και πριν το διάγραμμα χρονισμού για την $1^{η}$ επανάληψη του βρόχου, υποδεικνύοντας τις προωθήσεις που γίνονται. Πόσοι κύκλοι απαιτούνται τώρα για την εκτέλεση του κώδικα;

Γ) Θεωρώντας την ίδια σωλήνωση με το ερώτημα Β, μπορείτε να επιτύχετε καλύτερη επίδοση αναδιατάσσοντας τον κώδικα (με τις απαραίτητες βέβαια μετατροπές για να μην αλλάξετε την σημασιολογία του προγράμματος); Δείξτε όπως και πριν το διάγραμμα χρονισμού για την $1^{η}$ επανάληψη του βρόχου, υποδεικνύοντας τις προωθήσεις που γίνονται. Πόσοι κύκλοι απαιτούνται τώρα για την εκτέλεση του βρόχου;

A. Το loop θα εκτελεστεί συνολικά 156/4 = 39 φορές. Άρα έχουμε συνολικά 38*17+19 = 665 κύκλους.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
ADDI $t5,$t5, #4	F	D	X	M	W																	
LW $t3, 8($t5)		F	D	-	-	X	M	W														
LW $t4, 4($t3)			F	-	-	D	-	-	X	M	W											
SUB $t1, $t3, $t4						F	-	-	D	-	-	X	M	W								
SUB $t2, $t7, $t5									F	-	-	D	X	M	W							
SW $t1, 200($t2)												F	D	-	-	X	M	W				
BNEZ $t2, LOOP													F	-	-	D	X	M	W			
ADDI $t5,$t5, #4																		F	D	M	X	W

B. Το loop θα εκτελεστεί συνολικά 156/4 = 39 φορές. Άρα έχουμε συνολικά 38*11+13 = 431 κύκλους.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
ADDI $t5,$t5, #4	F	D	X	M	W											
LW $t3, 8($t5)		F	D	X	M	W										
LW $t4, 4($t3)			F	D	-	X	M	W								
SUB $t1, $t3, $t4				F	-	D	-	X	M	W						
SUB $t2, $t7, $t5						F	-	D	X	M	W					
SW $t1, 200($t2)								F	D	X	M	W				
BNEZ $t2, LOOP									F	D	X	M	W			
ADDI $t5,$t5, #4												F	D	M	X	W

C. Το loop θα εκτελεστεί συνολικά 156/4 = 39 φορές. Άρα έχουμε συνολικά 38*9+11 = 353 κύκλους.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14
LW $t3, 12($t5)	F	D	X	M	W									
ADDI $t5,$t5, #4		F	D	X	M	W								
LW $t4, 4($t3)			F	D	X	M	W							
SUB $t2, $t7, $t5				F	D	X	M	W						
SUB $t1, $t3, $t4					F	D	X	M	W					
SW $t1, 200($t2)						F	D	X	M	W				
BNEZ $t2, LOOP							F	D	X	M	W			
ADDI $t5,$t5, #4										F	D	M	X	W

Θέμα 3ο (30%)

Α) Δίνεται ένα πρότυπο κινητής υποδιαστολής 16 bits αντίστοιχο με το IEEE 754, όπου όμως ο εκθέτης έχει μήκος 7 bits και η μαντίσσα 8.

(i) Ποια η πόλωση του συστήματος;

Η πόλωση είναι:

$2^{7-1} - 1 = 63$

(ii) Βρείτε τον μεγαλύτερο και τον μικρότερο αριθμό που μπορούν να παρασταθούν.

$V_{MAX} = \pm 2^{126-63} * 1.11111111 \approx \pm 2^{63} * 2 = \pm 2^{64}$
$V_{MIN} = \pm 2^{1-63} * 1.00000000 = \pm 2^{-62}$

(iii) Δώστε την αναπαράσταση του -0.001647949(που είναι ίσος με $-27/2^{14}$)

$$-\frac{27}{2^{14}} = -(11011) * 2^{-14} = -(1.1011) * 2^{-10}$$

Επομένως έχουμε:

- sign = 1
- exp = -10 + 63 = $53_{<10>}$ = 0110101 $_{<2>}$
- mantissa = 10110000

και η ζητούμενη αναπαράσταση είναι η :

1	0	1	1	0	1	0	1	1	0	1	1	0	0	0	0

Β) Δίνεται το παρακάτω πρόγραμμα σε C καθώς και η αντίστοιχη μετάφραση του σε assembly MIPS. Συμπληρώστε τα κενά. Σας υπενθυμίζουμε ότι ο καταχωρητής $zero είναι πάντα μηδέν.

```
int boo (int *a, int b){
      int c;
      c = boo2(*a >> 6);
      *a = c + b;
      if ((b == 0) || (c == 0))
            return c;
      else
            return c & b;
   }
```

```
boo:    addi  $sp, $sp, -12
        sw    $a0, 0($sp)
        sw    $a1, 4($sp)
        sw    $ra, 8($sp)
        lw    $a0, 0($a0)
        sra   $a0, $a0, 6
        jal   boo2
        lw    $t0, 0($sp)
        lw    $t1, 4($sp)
        add   $t2, $v0, $t1
        sw    $t2, 0($t0)
        beq   $t1, $zero, exit
        beq   $v0, $zero, exit
        and   $v0, $v0, $t1
exit:   lw    $ra, 8($sp)
        addi  $sp, $sp, 12
        jr    $ra
```

Γ) Έστω μία cache με τα ακόλουθα χαρακτηριστικά:

- Η ελάχιστη μονάδα δεδομένων που μπορεί να διευθυνσιοδοτηθεί είναι το 1 byte.
- Οι διευθύνσεις έχουν μήκος 8 bits.
- Το μέγεθος της cache είναι 64 bytes.
- Χρησιμοποιεί LRU πολιτική αντικατάστασης.
- Είναι write-back.

Υποθέστε ότι αρχικά η cache είναι άδεια. Στον παρακάτω πίνακα δίνεται μια ακολουθία προσπελάσεων σε διευθύνσεις μνήμης, καθώς και η συμπεριφορά της cache σε κάθε προσπέλαση (hit ή miss).

Εντολή	**Διεύθυνση (hex)**	**Αποτέλεσμα**
lw	0xA5	miss
sw	0x66	miss
sw	0x6B	miss
lw	0xA3	hit
sw	0xE7	miss
sw	0xC3	miss
lw	0x61	miss

Σκοπός σας είναι να ανακαλύψετε τη δομή της cache βασιζόμενοι στη συμπεριφορά αυτή. Απαντήστε στις παρακάτω ερωτήσεις δικαιολογώντας την απάντησή σας:

(i) Ποιο είναι το μέγεθος του block;
(ii) Ποιο είναι το associativity;
(iii) Ποιο είναι το μέγεθος του tag;
(iv) Ποιες από τις προσπελάσεις του παραπάνω πίνακα οδηγούν τελικά σε εγγραφή κάποιου block πίσω στην κύρια μνήμη, σαν αποτέλεσμα της write-back πολιτικής;

Έχουμε τις διευθύνσεις:

lw 1 0 1 0 0 1 0 1 Miss
sw 0 1 1 0 0 1 1 0 Miss
sw 0 1 1 0 1 0 1 1 Miss
lw 1 0 1 0 0 0 1 1 Hit
sw 1 1 1 0 0 1 1 1 Miss
sw 1 1 0 0 0 0 1 1 Miss
lw 0 1 1 0 0 0 0 1 Miss

(i) Το μέγεθος του block είναι τουλάχιστον 8 ώστε το 0xA3 να κάνει hit στο 0xA5. Αν όμως είναι μεγαλύτερο ή ίσο του 16 τότε το 0x6B θα ήταν hit στο 0x66. Επομένως block_size = 8 και block_offset = 3

(ii) Έχουμε συνολικά 64/8 = 8 blocks. Επομένως η cache μπορεί να είναι DM, 2-way, 4-way ή FA. FA ή 4-way δεν μπορεί να είναι γιατί τότε το 0x61 θα ήταν hit. Αντίστοιχα, αν ήταν DM τότε το 0xA3 θα ήταν miss. Επομένως associativity = 2 και άρα για το index χρειαζόμαστε 2 bits.

(iii) tag = 8 – 2 – 3 = 3 bits.

(iv) Το 0xE7 έχει σαν αποτέλεσμα να γίνει write back το 0x66 και το 0x61 προκαλεί το write back του 0xE7.

Θέμα 4ο (20%)

Δίνεται ο παρακάτω κώδικας γραμμένος σε C.

```
double x[20][8], y[8][8], z[20][8];

for(i=0; i<8; i++)
  for(j=0; j<8; j++)
    y[i][j] = x[i][j] + z[i+8][j] * x[i+1][j];
```

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 συσχέτισης δύο δρόμων (2-way set associative), write-allocate, αποτελείται από 32 blocks δεδομένων, και έχει LRU πολιτική αντικατάστασης. Το μέγεθος του block είναι 32 bytes, ενώ η μικρότερη μονάδα δεδομένων που μπορεί να διευθυνσιοδοτηθεί είναι το 1 byte.
- Υποθέτουμε ότι όλες οι μεταβλητές, πλην των στοιχείων των πινάκων, μπορούν να αποθηκευτούν σε καταχωρητές του επεξεργαστή, οπότε οποιαδήποτε αναφορά σε αυτές δεν συνεπάγεται προσπέλαση στην κρυφή μνήμη. Επίσης, σε επίπεδο εντολών assembly οι αναγνώσεις γίνονται με τη σειρά που εμφανίζονται στον κώδικα.
- Οι πίνακες είναι αποθηκευμένοι στην κύρια μνήμη κατά γραμμές. Το πρώτο στοιχείο του πίνακα x βρίσκεται στη διεύθυνση **0x00000f40**, του πίνακα y στη διεύθυνση **0x08001940**, και του πίνακα z στη διεύθυνση **0x80004d40**.

A) Βρείτε το συνολικό αριθμό hits και misses για όλη την εκτέλεση του παραπάνω κώδικα. Υποδείξτε ποια misses είναι compulsory, ποια είναι capacity και ποια conflict.

B) Αν η κρυφή μνήμη γίνει write-no-allocate, διατηρώντας τα υπόλοιπα χαρακτηριστικά της ίδια, πώς θα επηρεαστούν τα hits και τα misses; Δικαιολογήστε την απάντησή σας δίνοντας όπως και πριν το συνολικό τους αριθμό.

32 bytes block size => 5 bits byte offset, 4 στοιχεία ανά block
32 blocks => 16 sets => 4 bits index

x: 0x0000 0f40 = 0000 0000 0000 0000 0000 1111 0100 0000
y: 0x0800 1940 = 0000 1000 0000 0000 0001 1001 0100 0000
z: 0x8000 4d40 = 1000 0000 0000 0000 0100 1101 0100 0000

Το x[1][0] θα απεικονίζεται 2 sets πιο κάτω σε σχέση με το x[0][0] :
(addr. x[1][0] = 0x00000f40 + 8_elements*8bytes = 0x00000f80 = 0000000000000000000 1111 1000 0000)

Το z[8][0] θα απεικονίζεται στο ίδιο set με το z[0][0] :
(addr. z[8][0] = 0x80004d40 + 64_elements*8bytes = 0x80004f40 = 10000000000000000100111101000000)

A)
x00 z80 x10 y00 m m m m
x01 z81 x11 y01 m m h m
x02 z82 x12 y02 m m h m
x03 z83 x13 y03 m m h m
x04 z84 x14 y04 m m m m
x05 z85 x15 y05 m m h m
x06 z86 x16 y06 m m h m
x07 z87 x17 y07 m m h m 26 misses + 6 hits

x10 z90 x20 y00 **h** m m m
x11 z91 x21 y01 m m h m
x12 z92 x22 y02 m m h m
x13 z93 x23 y03 m m h m
x14 z94 x24 y04 **h** m m m
x15 z95 x25 y05 m m h m
x16 z96 x26 y06 m m h m
x17 z97 x27 y07 m m h m 24 misses + 8 hits

Συνολικά:
misses = 26*1 + 24*7 = 194
hits = 6 + 8*7 = 62

B)

x00 z80 x10 y00 m m m m (no write to cache)
x01 z81 x11 y01 h h h m
x02 z82 x12 y02 h h h m
x03 z83 x13 y03 h h h m
x04 z84 x14 y04 m m m m
x05 z85 x15 y05 h h h m
x06 z86 x16 y06 h h h m
x07 z87 x17 y07 h h h m 14 misses + 18 hits

x10 z90 x20 y00 **h** m m m
x11 z91 x21 y01 h h h m
x12 z92 x22 y02 h h h m
x13 z93 x23 y03 h h h m
x14 z94 x24 y04 **h** m m m
x15 z95 x25 y05 h h h m
x16 z96 x26 y06 h h h m
x17 z97 x27 y07 h h h m 12 misses + 20 hits

Συνολικά:
misses = 14 + 12*7 = 98
hits = 18 + 20*7 = 158