Πανεπιστήμιο Θεσσαλίας
Τμήμα Πληροφορικής

Αρχιτεκτονική Υπολογιστών

Ασκήσεις Χειμερινού Εξαμήνου 2018-2019
(μέρος Γ΄)

Άσκηση 11 (εργαστηριακή):

Γράψτε ένα πρόγραμμα προσομοίωσης για την αξιολόγηση διαφόρων μηχανισμών δυναμικής πρόβλεψης διακλαδώσεων. Χρησιμοποιήστε τη λίστα δεκαεξαδικών διευθύνσεων διαδοχικών εντολών διακλάδωσης και αντίστοιχων διευθύνσεων προορισμού (branch traces) από το αρχείο trace.txt που προκύπτει από την εκτέλεση κάποιου μετροπρογράμματος. Προσέξτε ότι στην εκτέλεση αυτή οι διακλαδώσεις έχουν έναν κύκλο καθυστέρησης, ενώ όλες οι διευθύνσεις είναι ευθυγραμμισμένες σε πολλαπλάσια του 4, εφόσον οι εντολές έχουν μέγεθος λέξης.
Επιλέξτε μεγέθη πίνακα ιστορίας 1Kbits, 2Kbits, 8Kbits και 16Kbits. Ανάλογα με το μέγεθος του πίνακα, η προσπέλασή του γίνεται με τον κατάλληλο αριθμό από τα λιγότερο σημαντικά ψηφία της διεύθυνσης εντολής, όπως δείχνεται στο βιβλίο, που αναφέρεται σε ένα μηχανισμό πρόβλεψης (2,2).

Α. Εκτελέστε το πρόγραμμα προσομοίωσης, για να αξιολογήσετε την επίδοση των μηχανισμών (0,2), (0,4), (1,2), (1,4), (2,2) και (2,4) για τα τέσσερα παραπάνω μεγέθη πίνακα ιστορίας. Σχολιάστε τα αποτελέσματα.

Β. Εξετάστε την επίδοση ενός μηχανισμού (N,2) με πίνακα ιστορίας μίας μόνο θέσης. Στο μηχανισμό αυτό δε χρησιμοποιείται η διεύθυνση εντολής για την προσπέλαση του πίνακα, και η πρόβλεψη γίνεται από τα 2 ψηφία τοπικής ιστορίας, με βάση την επιλογή που κάνει η καθολική ιστορία. Πόσο τουλάχιστον πρέπει να είναι το N, ώστε ο μηχανισμός αυτός να έχει την ίδια επίδοση (i) με ένα μηχανισμό (0,2) με 256 θέσεις, (ii) με ένα μηχανισμό (0,2) με 512 θέσεις και (iii) με ένα μηχανισμό (0,4) με 256 θέσεις;

Άσκηση 12:

Έστω οι πιο κάτω βρόχοι σε γλώσσα C:

Α.
```
for (i=1; i<100; i=i+1) {
      a[i] = b[i] + c[i+1];   /* S1 */
      b[i] = a[i] + d[i];     /* S2 */
      c[i] = b[i] + e[i-1];   /* S3 */
      d[i] = c[i+1] + a[i+1]; /* S4 */
}
```

Β.
```
for (i=1; i<100; i=i+1) {
      a[i] = b[i] + c[i];     /* S1 */
      b[i] = a[i-1] + d[i];   /* S2 */
      c[i+1] = b[i] + e[i-1]; /* S3 */
}
```

Γ.
```
for (i=1; i<100; i=i+1) {
      a[i] = b[i] + c[i];     /* S1 */
      b[i] = a[i+1] + d[i];   /* S2 */
      c[i+1] = b[i] + e[i-1]; /* S3 */
      d[i-1] = c[i] + a[i-1]; /* S4 */
}
```

Δ.
```
for (i=1; i<100; i=i+1) {
```

```
        a[i] = b[i] + c[i];      /* S1 */
        b[i] = a[i-1] + d[i];    /* S2 */
        a[i+1] = a[i] + e[i-1]; /* S3 */
    }
```

Ε.
```
    for (i=1; i<100; i=i+1) {
        a[i] = b[i] + c[i];      /* S1 */
        b[i+1] = a[i-1] + d[i]; /* S2 */
        c[i] = b[i] + e[i-1];    /* S3 */
        d[i+1] = c[i-1] + a[i]; /* S4 */
    }
```

Για καθένα από τους παραπάνω βρόχους: Βρείτε τις εξαρτήσεις μεταξύ των στοιχείων πίνακα που αναφέρονται στο σώμα του. Στη συνέχεια εξετάστε αν αυτός είναι παράλληλος στη μορφή που δίνεται, αν δηλαδή οι κώδικες διαδοχικών επαναλήψεων μπορούν να δρομολογηθούν παράλληλα, χωρίς περιορισμούς στη διάταξη εντολών διαφορετικών επαναλήψεων. Διαφορετικά, εξετάστε εάν και πώς μπορεί να μετατραπεί σε ένα μοναδικό παράλληλο βρόχο, ή σε συνδυασμό πολλαπλών βρόχων όπου τουλάχιστον ο ένας να είναι παράλληλος. Πώς αλλάζει η απάντησή σας, εάν η δρομολόγηση των εντολών μπορεί να δεχτεί περιορισμούς στη διάταξη των εντολών διαφορετικών επαναλήψεων;

Άσκηση 13:

Στην άσκηση αυτή θα μελετήσουμε την εκτέλεση με στατική δρομολόγηση του βρόχου διανυσματικής επεξεργασίας που είδαμε και στην άσκηση 6. Υποθέστε τώρα ότι ο κώδικας αυτός εκτελείται σε έναν επεξεργαστή MIPS, όπου:

- Όλες οι εντολές σταθερής υποδιαστολής – εκτός από τις εντολές προσπέλασης μνήμης – έχουν διάρκεια εκτέλεσης 1 κύκλο μηχανής.
- Οι εντολές προσπέλασης μνήμης εκτελούνται σε 2 κύκλους μηχανής. Έτσι, ένα δεδομένο που φορτώνεται από τη μνήμη μπορεί να χρησιμοποιηθεί με καθυστέρηση ενός κύκλου μηχανής. Από την άλλη μεριά, κάποιο δεδομένο που αποθηκεύεται στη μνήμη μπορεί να σταλεί με το μηχανισμό παροχέτευσης απ' ευθείας στη φάση προσπέλασης μνήμης της αντίστοιχης εντολής αποθήκευσης.
- Οι εντολές κινητής υποδιαστολής εκτελούνται σε χρόνους 3, 8 και 20 κύκλων μηχανής χωρίς επικάλυψη, για πράξεις πρόσθεσης, πολλαπλασιασμού και διαίρεσης, αντίστοιχα.
- Οι εντολές διακλάδωσης εκτελούνται στη φάση αποκωδικοποίησης με έναν κύκλο καθυστέρησης. Ο μηχανισμός παροχέτευσης στέλνει όποιο δεδομένο απαιτείται, απ' ευθείας στη φάση αποκωδικοποίησης της αντίστοιχης εντολής διακλάδωσης. Η εντολή που ακολουθεί μια εντολή διακλάδωσης εκτελείται πάντα, γι' αυτό και αν ο μεταγλωττιστής δε βρει άλλη εντολή, εισάγει εντολή nop στη θέση καθυστέρησης. Ο κώδικας της άσκησης 6 δίνεται *πριν* τη συμπλήρωση της θέσης καθυστέρησης των διακλαδώσεων.

Α. Θεωρήστε ένα βαθμωτό επεξεργαστή MIPS. Πόσους κύκλους μηχανής χρειάζεται κάθε επανάληψη του βρόχου, (i) όταν οι εντολές του έχουν τη δρομολόγηση που προκύπτει από την παραπάνω σειρά, και (ii) όταν ο μεταγλωττιστής αναδιατάξει τις εντολές, ώστε να επιτύχει το συντομότερο χρόνο εκτέλεσης ανά επανάληψη; Στην πρώτη περίπτωση θεωρήστε ότι ο μεταγλωττιστής συμπληρώνει τη θέση καθυστέρησης των διακλαδώσεων με εντολή nop.

Β. Στη συνέχεια, θεωρήστε ένα βαθμωτό επεξεργαστή MIPS, που όμως είναι βελτιωμένος σε σχέση με τον παραπάνω με προσθήκη μεγάλου αριθμού υπομονάδων εκτέλεσης πράξεων κινητής υποδιαστολής. Ξεδιπλώστε το βρόχο όσες φορές χρειάζεται και αναδιατάξτε τις εντολές του νέου βρόχου, συμπληρώνοντας κατάλληλα τη θέση καθυστέρησης των εντολών διακλάδωσης, ώστε να μην υπάρχει κανένα πάγωμα ή χαμένος κύκλος μηχανής στην εκτέλεσή του. Για να επιτευχθεί ο συντομότερος χρόνος εκτέλεσης, πρέπει οι εντολές ελέγχου του βρόχου – μεταβολές δεικτών και έλεγχος τερματισμού – να εκτελούνται μια φορά ανά επανάληψη και να είναι κατάλληλα δρομολογημένες, ώστε να μην αυξάνουν το χρόνο εκτέλεσης του βρόχου. Όμως, επειδή δεν υπάρχει άλλη υποστήριξη υλικού, δε μπορείτε να μετακινήσετε εντολές που

προκαλούν ειδική περίπτωση – εδώ προσπελάσεις μνήμης και διαιρέσεις – παραβιάζοντας διαδικασιακές εξαρτήσεις. Πόσες φορές πρέπει να ξεδιπλωθεί ο βρόχος ώστε να επιτευχθεί ο πιο πάνω στόχος; Δώστε το χρονισμό στην εκτέλεση των εντολών μιας επανάληψης του νέου βρόχου. Πόσους κύκλους μηχανής χρειάζεται κάθε επανάληψη του νέου βρόχου και σε πόσους κύκλους μηχανής ανά επανάληψη του αρχικού βρόχου αντιστοιχεί ο χρόνος αυτός; Υποθέστε ότι με ένα ξεδίπλωμα k φορές, ο μεταγλωττιστής δημιουργεί κατάλληλο κώδικα στις διευθύνσεις exfoo1, exfoo2, …, exfook που αποτελούν τις διευθύνσεις προορισμού των αντίστοιχων διακλαδώσεων στο σώμα του βρόχου, ο οποίος συμπληρώνει προηγούμενες επαναλήψεις του αρχικού βρόχου που δεν έχουν ακόμα τελειώσει.

Γ. Θεωρήστε τώρα ότι ο προηγούμενος επεξεργαστής διαθέτει επιπλέον τη δυνατότητα υποθετικής φόρτωσης μέσω των εντολών:

```
ldc1z  $FRdest, offset($Rbase), $FRcond
ldc1nz $FRdest, offset($Rbase), $FRcond
```

όπου η φόρτωση ολοκληρώνεται, μόνο αν το τελευταίο τελούμενο – καταχωρητής κινητής υποδιαστολής – έχει μηδενική ή μη μηδενική τιμή, αντίστοιχα. Να επαναλάβετε τα ζητούμενα του προηγούμενου ερωτήματος, φροντίζοντας, ώστε με την αναδιάταξη των εντολών να χρησιμοποιείτε όπου πρέπει τις παραπάνω εντολές υποθετικής φόρτωσης, αντικαθιστώντας απλές εντολές φόρτωσης. Υποθέστε και πάλι κατάλληλο κώδικα στις διευθύνσεις exfoo1, exfoo2, …, exfook, ώστε να συμπληρώνονται ημιτελείς επαναλήψεις του αρχικού βρόχου.

Δ. Θεωρήστε τώρα έναν VLIW επεξεργαστή, ο οποίος σε κάθε λέξη εντολής κωδικοποιεί 5 απλές εντολές MIPS, από τις οποίες οι δύο είναι εντολές κινητής υποδιαστολής, οι δύο είναι εντολές προσπέλασης μνήμης, και η πέμπτη είναι εντολή σταθερής υποδιαστολής. Ο επεξεργαστής δεν αντιμετωπίζει εξαρτήσεις μεταξύ των επιμέρους εντολών MIPS της ίδιας λέξης εντολής. Υποθέστε ότι οι εντολές διακλάδωσης δεν έχουν καθυστέρηση, ενώ οι εντολές κινητής υποδιαστολής εκτελούνται σε οσεσδήποτε υπομονάδες είναι απαραίτητο.
(1) Να ξεδιπλώσετε το βρόχο και να δώστε μια κωδικοποίηση των εντολών MIPS σε εντολές VLIW, των οποίων η δρομολόγηση να μην οδηγεί σε πάγωμα ή χαμένους κύκλους μηχανής.
(2) Για βαθμό ξεδιπλώματος 4 φορές: Πόσους κύκλους μηχανής χρειάζεται κάθε επανάληψη του βρόχου και σε πόσους κύκλους μηχανής ανά επανάληψη του αρχικού βρόχου αντιστοιχεί ο χρόνος αυτός; Πόσο ποσοστό θέσεων επιμέρους εντολών MIPS καλύπτεται και πόσοι καταχωρητές απαιτούνται στον κώδικα VLIW που βρήκατε;
(3) Τι βαθμός ξεδιπλώματος απαιτείται για μέγιστη πλήρωση θέσεων στις εντολές VLIW, και σε πόσους κύκλους μηχανής για κάθε επανάληψη του νέου και του αρχικού βρόχου αντιστοιχεί; Πόσο διαφέρει το μέγεθος του κώδικα σε αριθμό εντολών VLIW μεταξύ αυτής και της προηγούμενης περίπτωσης;

Ε. Επιστρέψτε τώρα στο βαθμωτό επεξεργαστή MIPS του ερωτήματος Γ.
(1) Δώστε τον κώδικα που προκύπτει με συμβολικό ξεδίπλωμα του αρχικού βρόχου για αυτόν τον επεξεργαστή, αντικαθιστώντας απλές εντολές φόρτωσης με υποθετικές, όπου αυτό είναι απαραίτητο. Πόσους κύκλους μηχανής χρειάζεται κάθε επανάληψη του βρόχου;
(2) Επαναλάβετε, εφαρμόζοντας πριν το συμβολικό όσες φορές πραγματικό ξεδίπλωμα απαιτείται, ώστε να μην εμφανίζεται πάγωμα ή χαμένος κύκλος στην εκτέλεση του βρόχου.
Και για τις δύο περιπτώσεις αγνοήστε τον πρόσθετο κώδικα αποκατάστασης που απαιτείται, για την περίπτωση που κάποια διακλάδωση του σώματος του αρχικού βρόχου εκτελέσει άλμα σε μια από τις διευθύνσεις exfoo1, exfoo2, …, exfook. Να δώσετε όμως τόσο τον πρόλογο, όσο και τον επίλογο του συμβολικού ξεδιπλώματος.

Άσκηση 14:

Η υποθετική εκτέλεση εντολών σε μια κλασική αρχιτεκτονική MIPS γίνεται μέσω εντολών διακλάδωσης, οπότε οι εντολές που ακολουθούν μια διακλάδωση εκτελούνται υποθετικά μέχρι την αποτίμηση της συνθήκης της διακλάδωσης. Μια μέθοδος που χρησιμοποιείται συχνά σαν

εναλλακτική των εντολών διακλάδωσης είναι η βεβαιωμένη εκτέλεση (predication). Με τη μέθοδο αυτή, δεν απαιτείται εντολή διακλάδωσης πριν την εντολή που εκτελείται υποθετικά, αλλά η συνθήκη εκτέλεσης ενσωματώνεται στην εντολή με κατάλληλη μορφή προθέματος. Για παράδειγμα, η υποθετική εκτέλεση μιας εντολής πρόσθεσης με συνθήκη εκτέλεσης τη μη μηδενική τιμή ενός καταχωρητή, έστω του $8, θα γίνει με τη βοήθεια εντολής διακλάδωσης από τον κώδικα:

```
      beq   $8,$0,L
      add   $1,$2,$3
L:    <άλλη εντολή>
```

ενώ με βεβαιωμένη εκτέλεση από τον κώδικα:

```
      ($8) add $1,$2,$3
```

οπότε η εντολή add εκτελείται μόνο αν ο καταχωρητής $8 δεν έχει τιμή 0, διαφορετικά η εντολή συμπεριφέρεται σαν τη μηδενική εντολή nop.

Για την υποστήριξη της βεβαιωμένης εκτέλεσης στην αρχιτεκτονική MIPS, εισάγουμε έναν αριθμό από καταχωρητές βεβαίωσης μεγέθους 1 bit, οι οποίοι λαμβάνουν τιμή από εντολές σύγκρισης των περιεχομένων δύο καταχωρητών που έχουν τη γενική μορφή:

```
      cmp.xx p1,p2=$rs,$rt
```

όπου p1 και p2 καταχωρητές βεβαίωσης, από τους ο οποίους ο πρώτος λαμβάνει τιμή 1 αν το αποτέλεσμα της σύγκρισης είναι "Αληθές", και 0 αν το αποτέλεσμα της σύγκρισης είναι "Ψευδές", και ο δεύτερος – προαιρετικός – λαμβάνει τη συμπληρωματική τιμή του πρώτου. Η σύγκριση καθορίζεται από την τιμή του επιθέματος xx, και για παράδειγμα η εντολή cmp.eq εκτελεί σύγκριση για ισότητα, ενώ η εντολή cmp.ne εκτελεί σύγκριση για ανισότητα. Οι καταχωρητές που συγκρίνονται είναι οι $rs και $rt. Μια εντολή σύγκρισης δέχεται κι αυτή βεβαίωση, και αν ο καταχωρητής βεβαίωσής της έχει τιμή 0, συμπεριφέρεται σαν εντολή nop.

Α. Ξαναγράψτε τον πιο κάτω κώδικα MIPS[1] με τη μέθοδο της βεβαιωμένης εκτέλεσης, έτσι ώστε να μην περιέχει διακλαδώσεις, εκτός από την τελευταία που κλείνει το βρόχο:

```
L:    lw    $13,0($1)
      subu  $2,$13,$14
      bne   $2,$0,L1
      lw    $2,24($6)
      beq   $2,$0,L2
L1:   ldc1  $f0,0($2)
      slt.d $8,$f0,$f8
      beq   $8,$0,L3
      mul.d $f0,$f0,$f2
L3:   add.d $f0,$f0,$f4
      sdc1  $f0,0($2)
L2:   addiu $1,$1,4
      addiu $6,$6,4
      bne   $1,$9,L
```

Υποθέστε ότι διαθέτετε 8 καταχωρητές βεβαίωσης p0 - p7.
Ποιες είναι οι εξαρτήσεις στον αρχικό κώδικα, και ποιες στον τελικό; Πώς διαφέρουν αυτές και τι πλεονέκτημα προσφέρει η βεβαιωμένη εκτέλεση στην απόδοση του κώδικα;

Β. Υποθέστε ότι οι δύο κώδικες εκτελούνται σε ένα βαθμωτό επεξεργαστή MIPS, όπου:

- Όλες οι εντολές σταθερής υποδιαστολής – εκτός από τις εντολές προσπέλασης μνήμης – έχουν διάρκεια εκτέλεσης 1 κύκλο μηχανής.
- Οι εντολές προσπέλασης μνήμης εκτελούνται σε 2 κύκλους μηχανής. Έτσι, ένα δεδομένο που φορτώνεται από τη μνήμη μπορεί να χρησιμοποιηθεί με καθυστέρηση ενός κύκλου μηχανής. Από την άλλη μεριά, κάποιο δεδομένο που αποθηκεύεται στη μνήμη μπορεί να

[1] Η εντολή slt.d δεν ανήκει στο σύνολο εντολών MIPS, και απλά επεκτείνει την ακέραια εντολή slt για αριθμούς κινητής υποδιαστολής μεγέθους d (double), με ακέραιο λογικό αποτέλεσμα.

σταλεί με το μηχανισμό παροχέτευσης απ' ευθείας στη φάση προσπέλασης μνήμης της αντίστοιχης εντολής αποθήκευσης.
- Οι εντολές κινητής υποδιαστολής εκτελούνται σε χρόνους 2, 3 και 8 κύκλων μηχανής χωρίς επικάλυψη, για πράξεις σύγκρισης, πρόσθεσης και πολλαπλασιασμού, αντίστοιχα.
- Οι εντολές διακλάδωσης εκτελούνται στη φάση εκτέλεσης χωρίς καθυστέρηση.

Πόσους κύκλους μηχανής χρειάζεται κάθε επανάληψη του βρόχου για καθέναν από τους δύο κώδικες, όταν οι διακλαδώσεις χρησιμοποιούν στατική πρόβλεψη με βάση το πρόσημο της μετατόπισης και οι τρεις διακλαδώσεις στο σώμα του βρόχου του πρώτου κώδικα εκτελούν το άλμα τους σε ποσοστό 60% η πρώτη, 5% η δεύτερη και 25% η τρίτη;

Γ. Εφαρμόστε την τεχνική του συμβολικού ξεδιπλώματος στον κώδικα με βεβαιωμένη εκτέλεση για τον ίδιο επεξεργαστή. Διαχωρίστε τον κώδικα στα λιγότερα δυνατά επίπεδα εξαρτήσεων, ώστε να μην έχετε κανένα πάγωμα στην εκτέλεση του κώδικα, αλλά και να χρειαστείτε τον ελάχιστο αριθμό πρόσθετων εντολών αντιγραφής. Πόσους κύκλους μηχανής χρειάζεται τώρα κάθε επανάληψη του βρόχου;

Δ. Πώς θα διαμορφωνόταν ο κώδικας που βρήκατε στο ερώτημα Α, για εκτέλεση σε μια αρχιτεκτονική σαν την IA-64, η οποία δέχεται VLIW εντολές με τη μορφή που ορίζονται στην IA-64 (δέσμες με σχεδιότυπο, συλλαβές και ακινητοποιήσεις), αλλά με επιμέρους εντολές (συλλαβές) τις παραπάνω εντολές MIPS; Πριν τη μετατροπή σε κώδικα VLIW, εφαρμόστε πραγματικό ξεδίπλωμα, ώστε να λάβετε υψηλότερο βαθμό ILP. Πόσους κύκλους μηχανής χρειάζεται τώρα κάθε επανάληψη του βρόχου;

Άσκηση 15:

Θεωρήστε την τεχνική πρόβλεψης πλαισίου (way prediction) για κρυφές μνήμες με οργάνωση συνόλου συσχέτισης (set associative). Με την τεχνική αυτή, ο χρόνος επιτυχημένης προσπέλασης μειώνεται σε αυτόν που αντιστοιχεί στις πιο γρήγορες κρυφές μνήμες με οργάνωση άμεσης απεικόνισης (direct mapped). Η τεχνική πρόβλεψης πλαισίου μπορεί όμως να χρησιμοποιηθεί και για κάποιον δεύτερο σκοπό: τη μείωση του κόστους συσκευασίας του chip.

Έστω ο επεξεργαστής MIPS R10K. Όσο αφορά το σύστημα μνήμης, το υλικό του επεξεργαστή αυτού περιλαμβάνει: κρυφή μνήμη L1, κύκλωμα σύγκρισης ετικετών για κρυφή μνήμη L2, καθώς και κύκλωμα πρόβλεψης πλαισίου για κρυφή μνήμη L2. Έτσι, ετικέτες από τα πλαίσια της μνήμης L2 φτάνουν στον επεξεργαστή για ανίχνευση επιτυχίας ή αποτυχίας στην L2. Το κύκλωμα πρόβλεψης πλαισίου περιλαμβάνει έναν πίνακα πρόβλεψης μεγέθους 8K bits, με κάθε bit να επιλέγει ένα από δύο πλαίσια. Η κρυφή μνήμη L2 υλοποιείται εξωτερικά του επεξεργαστή και πρέπει να είναι οργανωμένη σε σύνολα συσχέτισης 2 τρόπων.

Α. Εξηγήστε πώς η τεχνική πρόβλεψης πλαισίου μπορεί να μειώσει το κόστος συσκευασίας του chip όσο αφορά τον αριθμό ακίδων (pins) του επεξεργαστή που απαιτούνται για την επικοινωνία με την κρυφή μνήμη L2.

Β. Συγκρίνετε την απόδοση του επεξεργαστή στην επικοινωνία με τη μνήμη αυτή, σε σχέση με έναν επεξεργαστή MIPS R10K που δε χρησιμοποιεί πρόβλεψη πλαισίου, όταν ο δεύτερος έχει (α) όσες ακίδες απαιτούνται για την προσπέλαση της L2 χωρίς πρόβλεψη πλαισίου, και (β) τον ίδιο αριθμό ακίδων με τον πρώτο.

Γ. Έστω ότι η πρόβλεψη πλαισίου γίνεται με βάση τον αλγόριθμο του πιο πρόσφατα χρησιμοποιηθέντος (most recently used) πλαισίου. Δώστε σε μορφή γράφου τις μεταβάσεις στην κατάσταση πρόβλεψης ενός συνόλου, όπου να φαίνονται μεταβάσεις σε επιτυχία πρόβλεψης, σε αποτυχία πρόβλεψης, καθώς και σε αποτυχία στην προσπέλαση της κρυφής μνήμης L2.

Δ. Αν η κρυφή μνήμη L2 έχει μέγεθος 512KB και πλαίσια των 64 bytes, πόσο μέγεθος πίνακα πρόβλεψης απαιτείται; Επαναλάβετε για μέγεθος μνήμης 4MB. Πώς θα απεικονίζατε τυχόν περισσότερα σύνολα, και πώς θα χρησιμοποιούσατε τυχόν περισσευούμενα bits στον πίνακα των 8K bits που διατίθεται στον επεξεργαστή;