

ΣΗΜΕΙΩΣΕΙΣ ΑΡΧΙΤΕΚΤΟΝΙΚΗΣ II

(Σύγχρονα Θέματα Αρχιτεκτονικής Υπολογιστών)

Computer – Ανάλυση



Περιεχόμενα

Τυπολόγιο	1
Κεφάλαιο 3.3 – Σχεδίαση μονάδας ελέγχου και μονάδας επεξεργασίας δεδομένων	3
3.1 Επεξήγηση λειτουργίας μονάδας επεξεργασίας δεδομένων και μονάδας ελέγχου πολλών κ.ρ./ενός κ.ρ	3
Σήματα ελέγχου πολυπλέκτη Α και Β στην περίπτωση πολλών κ.ρ.	3
Ενέργειες κατά τη διάρκεια 1 ^{ου} και 2 ^{ου} κ.ρ. στην περίπτωση πολλών κ.ρ.	4
Θέμα για πλεονεκτήματα/μειονεκτήματα προσκόμισης και εκτέλεσης κάθε εντολής σε 1 κ. ρ.	4
Θέμα για συνύπαρξη εντολών LOAD r1, (r2)/LOAD r1, d(r2) σε 1 κ. ρ./πολλούς κ.ρ.	4
i. Ένας κύκλος ρολογιού	4
ii. Πολλοί κύκλοι ρολογιού	5
Θέμα θεωρίας Ιουνίου 2019 με έναν/πολλούς κύκλους ρολογιού	5
Θέμα θεωρίας Ιουνίου 2019 με ένα/πολλούς κύκλους ρολογιού	5
Θέμα θεωρίας Ιουνίου 2019 με ένα/πολλούς κύκλους ρολογιού	6
Θέμα θεωρίας Ιουνίου 2016 με έναν και πολλούς κύκλος ρολογιού	6
Θέμα Ιουνίου 2016 με τροποποίηση μονάδας επεξεργασίας δεδομένων για ένα κ.ρ.	6
Θέμα Σεπτεμβρίου 2019 με τροποποίηση μονάδας επεξεργασίας δεδομένων για πολλούς κ.ρ.	7
Θέμα Ιουνίου 2019 με τροποποίηση μονάδας επεξεργασίας δεδομένων για πολλούς κ.ρ.	8
Θέμα Σεπτεμβρίου 2016 με μονάδα επεξεργασίας δεδομένων για πολλούς κ.ρ.	9
Θέμα Σεπτεμβρίου 2014 με τροποποίηση μονάδα επεξεργασίας δεδομένων για πολλούς κ.ρ.	10
Τροποποίηση μονάδες ελέγχου για ένα/πολλούς κ.ρ. για εκτέλεση εντολής LOAD r1, d(r2).....	11
I. Ένας κύκλος ρολογιού.....	11
II. Πολλοί κύκλοι ρολογιού	12
Θέμα Σεπτεμβρίου 2016 με Σ/Λ για ένα/πολλούς κ.ρ.	12
Θέμα με τροποποίηση μονάδας επεξεργασίας δεδομένων για πολλούς κ.ρ.	14
Θέμα Ιουνίου 2019	15
Κεφάλαιο 4 – Μηχανισμός μερικώς επικαλυπτόμενων λειτουργιών και εξαρτήσεις.....	17
4.1 Θεωρία εξαρτήσεων	17
Θέμα για το μέγιστο ρυθμό ολοκλήρωσης εντολών σε ΜΕΛ	19
Θέμα αναφορικά με τα είδη εξαρτήσεων σε ένα ΜΕΛ.....	19
Θέμα αναφορικά με τη χρήση εντολών NOP	19
Θέμα αναφορικά με τους τρόπους επίλυσης εξαρτήσεων.....	19
Θέμα αναφορικά με τεχνική παροχέτευσης	20
Θέμα θεωρίας αναφορικά με τα μειονεκτήματα - πλεονεκτήματα του ΜΕΛ	20
Θέμα αναφορικά με επιτάχυνση του ρυθμού ολοκλήρωσης εντολών	20
Θέμα Σεπτεμβρίου 2019 για ΜΕΛ	21
Θέμα Ιουνίου 2014 θεωρίας	21
Θέμα Ιουνίου 2019 για ΜΕΛ	22
Θέμα Ιουνίου 2019 για ΜΕΛ	22
Θέμα θεωρίας Ιουνίου 2016 με εξαρτήσεις.....	23

Άσκηση 4.2 από λυσάρι με ΜΕΛ	23
Άσκηση 4.3 από λυσάρι με ΜΕΛ	23
Άσκηση 4.4 από λυσάρι με ΜΕΛ	24
Άσκηση 4.5 από λυσάρι με ΜΕΛ	24
Θέμα Ιουνίου 2014 με ΜΕΛ.....	26
Θέμα Ιουνίου 2012 με ΜΕΛ.....	26
Θέμα Σεπτεμβρίου 2014 με ιδανικό ΜΕΛ	27
Θέμα Σεπτεμβρίου 2020 με ΜΕΛ	27
Θέμα Σεπτεμβρίου 2020 με ΜΕΛ	28
Θέμα Ιουνίου 2016 με εξαρτήσεις	28
Άσκηση με όλα τα είδη εξαρτήσεων	30
Άσκηση από προφορική εξέταση πάνω στις εξαρτήσεις	32
Θέμα Ιουνίου 2019 με εξαρτήσεις	32
Θέμα Σεπτεμβρίου 2019 με εξαρτήσεις	33
Θέμα Σεπτεμβρίου 2020 με εξαρτήσεις.....	33
Θέμα Σεπτεμβρίου 2020 με εξαρτήσεις.....	34
Θέμα Σεπτεμβρίου 2020 με εξαρτήσεις.....	35
Θέμα Σεπτεμβρίου 2020 με εξαρτήσεις.....	35
Θέματα Ιανουαρίου 2020.....	36
Θέμα με εξαρτήσεις.....	38
Θέμα με εξαρτήσεις και επίλυση εξαρτήσεων.....	42
Κεφάλαιο 5.3 – Κρυφή μνήμη	45
Επεξήγηση τρόπου λειτουργίας κρυφής μνήμης (cache memory).....	45
1. Βασικές έννοιες κρυφής μνήμης	45
2. Βασικές οργανώσεις (είδη) κρυφής μνήμης:	45
3. Σύγκριση οργανώσεων κρυφής μνήμης	46
4. Θεωρία - Χαρακτηριστικά μεγέθη κρυφής μνήμης.....	46
Θέμα Ιουνίου 2016 με κρυφή μνήμη	48
Θέμα Ιουνίου 2019 με κρυφή μνήμη	49
Θέμα Σεπτεμβρίου 2019 με κρυφή μνήμη.....	50
Θέμα προόδου 2018 με κρυφή μνήμη.....	51
Θέμα προόδου 2018 με κρυφή μνήμη.....	52
Θέμα Σεπτεμβρίου 2021 με κρυφή μνήμη.....	54
Θέμα Σεπτεμβρίου 2020 με κρυφή μνήμη.....	55
Θέμα Φεβρουαρίου 2021 με κρυφή μνήμη.....	56
Θέμα Σεπτεμβρίου 2020 με κρυφή μνήμη.....	57
Θέμα Σεπτεμβρίου 2020 με κρυφή μνήμη.....	57
Θέμα Σεπτεμβρίου 2021 με κρυφή μνήμη.....	58
Θέμα με κρυφή μνήμη	60

Θέμα Σεπτεμβρίου 2021 με κρυφή μνήμη και ιδεατή μνήμη	62
Θέμα Σεπτεμβρίου 2021 με κρυφή μνήμη και ιδεατή μνήμη	62
Θέμα με κρυφή μνήμη	64
Θέμα Σεπτεμβρίου 2016 με κρυφή μνήμη.....	65
Θέμα με κρυφή μνήμη και γέμισμα πίνακα διευθύνσεων με στρατηγική LRU.....	66
Θέμα με κρυφή μνήμη και στρατηγικές απελευθέρωσης πλαισίων	68
Θέμα με κρυφή μνήμη και κώδικα Assembly	70
Θέμα με χωρητικότητα κρυφής μνήμης.....	72
Παράδειγμα κρυφής μνήμης με οργάνωση μονοσήμαντης απεικόνισης	73
Θέμα με κρυφή μνήμη και χωρητικότητα μνήμης ετικετών.....	74
Θέμα με κρυφή μνήμη και στρατηγική αντικατάστασης LRU	75
Θέμα 2019 με κρυφή μνήμη και δύο επεξεργαστές.....	76
Θέμα Ιουνίου 2015 με κρυφή μνήμη	78
Θέμα Φεβρουαρίου 2012 με κρυφή μνήμη.....	80
Άσκηση 5.8 με κρυφή μνήμη	82
Άσκηση 5.10 με κρυφή μνήμη	83
Άσκηση από Φροντιστήριο με κρυφή μνήμη	84
Άσκηση από Φροντιστήριο με κρυφή μνήμη	85
Κεφάλαιο 5.5 – Ιδεατή μνήμη	87
5.5.1 Λειτουργία ιδεατής μνήμης.....	87
5.5.2 Σύγκριση κρυφής μνήμης επεξεργαστή και κρυφής μνήμης Πίνακα Σελίδων	89
5.5.10 Φαινόμενο thrashing	92
5.5.11 Σημασία αντιστραμμένου πίνακα σελίδων (ΑΠΣ)	92
5.5.12. Σημασία κρυφής μνήμης πίνακα σελίδων (TLB)	92
5.5.13. Επιτυχία ιδεατής μνήμης.....	93
5.5.14. Θέμα σχετικά με μειονεκτήματα σελιδοποίησης	93
Παράδειγμα 5.10 – Εύρεση χωρητικότητας Π.Σ. με την τεχνική σελιδοποίησης	93
Α' Τρόπος μείωσης χωρητικότητας Π.Σ.: χρήση πολλών επιπέδων.....	94
Άσκηση 5.16 – Ιδεατή μνήμη με την τεχνική των πολλών επιπέδων	94
Άσκηση 5.17 – Ιδεατή μνήμη με την τεχνική των πολλών επιπέδων	97
Θέμα Ιουνίου 2019 - – Ιδεατή μνήμη με την τεχνική των πολλών επιπέδων και ΑΠΣ	99
Θέμα Ιουνίου 2020 με αντιστραμμένο πίνακα σελίδων.....	100
Θέμα Σεπτεμβρίου 2020 – Ιδεατή μνήμη με αντιστραμμένο πίνακα σελίδων	101
Θέμα Ιουνίου 2015 με ιδεατή μνήμη με ΑΠΣ.....	102
Παράδειγμα 5.11 με ιδεατή μνήμη και ΑΠΣ	103
Άσκηση 5.18 – Ιδεατή μνήμη με ΑΠΣ	104
Θέμα Ιανουαρίου 2020 – Ιδεατή μνήμη με ΑΠΣ και τεχνική πολλών επιπέδων	107
Θέμα προόδου Μαΐου 2022 με ΑΠΣ	109
Θέμα Ιουνίου 2016 – Ιδεατή μνήμη με χρήση ΑΠΜ	111

Θέμα Σεπτεμβρίου 2017 – Ιδεατή μνήμη με χρήση κρυφής μνήμης πίνακα σελίδων (TLB).....	111
Θέμα Σεπτεμβρίου 2015 με υβριδική κρυφή μνήμη επεξεργαστή	114
Θέμα Ιουνίου 2014 – Ιδεατή μνήμη με χρήση TLB.....	115
Άσκηση 5.21 από λυσάρι – Ιδεατή μνήμη με χρήση TLB	116
Άσκηση 5.22 – Ιδεατή μνήμη με χρήση TLB	120
Θέμα προόδου 2018 με κρυφή – Ιδεατή μνήμη	123
Κεφάλαιο 7 – Υπερβαθμωτοί επεξεργαστές και ΠΜΜΕ επεξεργαστές.....	124
7.1 Υπερβαθμωτοί επεξεργαστές	124
7.1.1. Βασικά σημεία θεωρίας	124
Θέμα με υπερβαθμωτούς και το μηχανισμό επαναδιάταξης αποτελεσμάτων	125
Θέμα θεωρίας Σεπτεμβρίου 2021 με υπερβαθμωτούς επεξεργαστές.....	126
Θέμα θεωρίας Σεπτεμβρίου 2021 με υπερβαθμωτούς επεξεργαστές.....	126
Θέμα θεωρίας Σεπτεμβρίου 2021 με υπερβαθμωτούς επεξεργαστές.....	126
Θέμα θεωρίας Ιουνίου 2015 με μηχανισμό επαναδιάταξης αποτελεσμάτων σε υπερβαθμωτούς	126
Θέμα για μηχανισμό επαναδιάταξης αποτελεσμάτων.....	126
Θέμα θεωρίας Σεπτεμβρίου 2019 με υπερβαθμωτούς επεξεργαστές.....	127
Θέματα θεωρίας Σεπτεμβρίου 2019 με υπερβαθμωτούς επεξεργαστές.....	127
7.1.2. Άσκηση σχετικά με υπερβαθμωτούς επεξεργαστές	127
7.1.3. Άσκηση σχετικά με υπερβαθμωτούς επεξεργαστές	128
7.1.4 Θέμα με υπερβαθμωτούς.....	129
Θέμα Σεπτεμβρίου 2021 με υπερβαθμωτούς επεξεργαστές	134
Θέμα Σεπτεμβρίου 2021 με υπερβαθμωτούς επεξεργαστές	135
Θέματα Σεπτεμβρίου 2021 με υπερβαθμωτούς επεξεργαστές.....	135
Θέματα Σεπτεμβρίου 2020 με υπερβαθμωτούς επεξεργαστές.....	138
Θέματα Φεβρουαρίου 2021 με υπερβαθμωτούς επεξεργαστές.....	139
Θέμα Σεπτεμβρίου 2020 με υπερβαθμωτούς επεξεργαστές	140
Θέμα με ασθενή/ισχυρή συνέπεια επεξεργαστή	141
7.2 ΠΜΜΕ επεξεργαστές.....	143
7.2.1. Βασικά σημεία θεωρίας	143
7.2.3. Θέμα με ΠΜΜΕ	144
7.2.3 Δεύτερο θέμα με ΠΜΜΕ	145
Θέμα θεωρίας Σεπτεμβρίου 2019 με ΠΜΜΕ	145
Θέμα θεωρίας Σεπτεμβρίου 2019 με ΠΜΜΕ	145
Θέμα αναφορικά με την κύρια διαφορά υπερβαθμωτών/ΠΜΜΕ	145
Θέματα Φεβρουαρίου 2023.....	146
Θέματα Προόδου 2023.....	148

Τυπολόγιο

Μετατροπή από δεκαεξαδικό σε δυαδικό/δεκαδικό σύστημα

Hex	Binary				Δεκαδικό
	2^3	2^2	2^1	2^0	
0	0	0	0	0	0
1	0	0	0	1	1
2	0	0	1	0	2
3	0	0	1	1	3
4	0	1	0	0	4
5	0	1	0	1	5
6	0	1	1	0	6
7	0	1	1	1	7
8	1	0	0	0	8
9	1	0	0	0	9
A	1	0	1	0	10
B	1	0	1	1	11
C	1	1	0	0	12
D	1	1	0	1	13
E	1	1	1	0	14
F	1	1	1	1	15

Μετατροπή από οκταδικό σε δυαδικό σύστημα

Octal	Binary		
	2^2	2^1	2^0
0	0	0	0
1	0	0	1
2	0	1	0
3	0	1	1
4	1	0	0
5	1	0	1
6	1	1	0
7	1	1	1



Τυπολόγιο δυνάμεων του 2

$2^0 = 1$	$2^{10} = 1024 = 1K$
$2^1 = 2$	$2^{11} = 2048 = 2K$
$2^2 = 4$	$2^{12} = 4096 = 4K$
$2^3 = 8$	$2^{13} = 8192 = 8K$
$2^4 = 16$	$2^{14} = 16384 = 16K$
$2^5 = 32$	$2^{15} = 32768 = 32K$
$2^6 = 64$	$2^{16} = 65536 = 64K$
$2^7 = 128$	
$2^8 = 256$	
$2^9 = 512$	

Μονάδες μέτρησης χωρητικότητας μνήμης

1 bit = 0 ή 1 (δυαδικό ψηφίο)

1 byte = 8 bits (ψηφιολέξη) = 1B  Πολλαπλάσια του byte

1 KB = 2^{10} bytes (Kilobyte)

1 byte (ψηφιολέξη) = 8 bits

1 MB = 2^{20} bytes (Megabyte)

1 halfword (μισή λέξη) = 16 bits = 2 bytes

1 GB = 2^{30} bytes (Gigabyte)

1 word (λέξη) = 32 bits = 4 bytes

1 TB = 2^{40} bytes (Terabyte)

1 double word (διπλή λέξη) = 64 bits = 8 bytes

1 quad word (τετραπλή λέξη) = 128 bits = 16 bytes

Κεφάλαιο 3.3 – Σχεδίαση μονάδας ελέγχου και μονάδας επεξεργασίας δεδομένων

3.1 Επεξήγηση λειτουργίας μονάδας επεξεργασίας δεδομένων και μονάδας ελέγχου πολλών κ.ρ./ενός κ.ρ

1. Στους **πολλούς κ.ρ.** απαιτούνται 3 ή 4 κ.ρ. ανάλογα με το αν η εντολή που εκτελείται, αποθηκεύει το αποτέλεσμά της σε καταχωρητή ή όχι. Αν το αποθηκεύει σε κάποιο καταχωρητή, τότε απαιτούνται 4 κ.ρ., διότι κατά τη διάρκεια του 4^{ου} κ.ρ. λαμβάνει χώρα η αποθήκευση σε κάποιο καταχωρητή του αρχείου καταχωρητών (π.χ. r1, r3). Αν όχι, απαιτούνται 3 κ.ρ.

Παραδείγματα εντολών που απαιτούν **4 κ.ρ.** είναι οι εντολές:

LOAD r1, (r2) //r1 ← M(r2), ADD r1, r2, r3 //r3 ← r1 + r2, SUB r1, r2, r3 //r3 ← r1 - r2, AND r1, r2, r3 //r3 ← r1 ^ r2.

Παραδείγματα εντολών που απαιτούν **3 κ.ρ.** είναι οι εντολές:

Store r1, (r2) //M(r2) ← r1, BRE r1, r2, d //Av r1 - r2 = 0 ⇒ r1 = r2 τότε MP ← MP + d.

2. Σε περίπτωση που εκτελούνται οι **παραλλαγές** των εντολών Load και Store, δηλαδή οι εντολές: **Load r1, d(r2)** και **Store r1, d(r2)**, ο αριθμός των απαιτούμενων κ.ρ. αυξάνεται κατά «1». Ο επιπλέον κ.ρ. χρειάζεται και στις δύο περιπτώσεις για τον υπολογισμό του αθροίσματος $r2 + d$. Πιο συγκεκριμένα:

Η εντολή Load r1, d(r2) // r1 ← M(r2 + d) απαιτεί 5 κ.ρ. και

Η εντολή Store r1, d(r2) // M(r2 + d) ← r1 απαιτεί 4 κ.ρ.

3. **Στους πολλούς κ.ρ. το κύκλωμα είναι ακολουθιακό**, διότι περιέχει στοιχεία μνήμης, όπως είναι οι καταχωρητές ειδικού σκοπού KE και ΚΑΛΜ, ενώ **στην περίπτωση του ενός κ.ρ. το κύκλωμα είναι συνδυαστικό**, διότι δεν περιέχει στοιχεία μνήμης, δηλαδή καταχωρητές ειδικού σκοπού. Στον KE αποθηκεύουμε - κατά τη διάρκεια του πρώτου κ.ρ.- τον κωδικό λειτουργίας της εντολής (opcode) και φροντίζουμε έτσι να τον διατηρήσουμε για όσο διάστημα (όσους κ.ρ.) εκτελείται η εντολή. Αυτό σημαίνει ότι το σήμα γKE είναι «1» μόνο κατά τη διάρκεια του πρώτου κ.ρ., ενώ σε όλους τους υπόλοιπους κ.ρ. είναι «0», για να μην μπορεί να γραφτεί μια νέα εντολή στον KE, για όσο διάστημα εκτελείται η τρέχουσα εντολή. Στον ΚΑΛΜ αποθηκεύουμε -στο δεύτερο κ.ρ.- το αποτέλεσμα MP + d **ανεξάρτητα** από το αν στον 3^ο κ.ρ. εκτελεστεί η εντολή BRE r1, r2, d. Επειδή οι δύο πρώτοι κ.ρ. είναι **κοινοί** για όλες ανεξαιρέτως τις εντολές, οι τιμές των σημάτων ελέγχου γMP, δME και γKE είναι επίσης **κοινές** για όλες τις εντολές σε όλους τους κ.ρ. Πιο συγκεκριμένα, οι τιμές των συγκεκριμένων σημάτων ελέγχου είναι: γMP = 1, 0, 0, 0 και δME = 1, 0, 0, 0 και γKE = 1, 0, 0, 0 για την περίπτωση των 4 κ.ρ. ή γMP = 1, 0, 0 και δME = 1, 0, 0 και γKE = 1, 0, 0 για την περίπτωση των 3 κ.ρ.

4. Τα σήματα ελέγχου που αρχίζουν με **δ** (**διάβασμα**) και **γ** (**γραφή**) **δεν** μπορούν να πάρουν ποτέ την αδιάφορη τιμή (X). Παίρνουν την τιμή «1» όταν χρησιμοποιούνται και «0» όταν **δεν** χρησιμοποιούνται. Αντίθετα, τα σήματα επιλογής των πολυπλεκτών, όπως πE&Δ, πA, πB μπορούν να πάρουν την αδιάφορη τιμή (X), όταν σε κάποιο κ.ρ. δεν χρησιμοποιούνται.

5. **Οι πολυπλέκτες E και Δ χρησιμοποιούνται μόνο όταν εκτελείται εντολή που αποθηκεύει σε καταχωρητή (LOAD, ADD, SUB, AND) καθώς η **έξοδος** του πολυπλέκτη E **συνδέεται με τη θύρα εγγραφής δE** του αρχείου καταχωρητών (περιέχει τους 16 καταχωρητές γενικού σκοπού). Σε εντολές που **δεν** αποθηκεύουν σε καταχωρητή (STORE, BRE) **οι πολυπλέκτες E και Δ δεν χρησιμοποιούνται** και ως εκ' τούτου το σήμα πE&Δ = x, x, x.**

6. Στη μονάδα επεξεργασίας δεδομένων και τη σχετική μονάδα ελέγχου **του ενός κ.ρ.** όταν πρέπει να γίνουν **δύο ταυτοτικές λειτουργίες**, π.χ. δύο προσθέσεις, **πρέπει να υπάρχει πάντα αντίστοιχο πλήθος λειτουργικών μονάδων** π.χ. **αθροιστές για την εκτέλεσή τους**. Αντίθετα, αυτό **δεν** είναι ποτέ αναγκαίο να συμβεί στη μονάδα επεξεργασίας δεδομένων και τη σχετική μονάδας ελέγχου των πολλών κ.ρ. διότι στην περίπτωση αυτή οι πράξεις εκτελούνται με τη βοήθεια της ΑΛΜ.

7. Μόνο οι εντολές Load r1, (r2), Load r1, d(r2), Store r1, (r2), Store r1, d(r2) προσπελαύνουν την κρυφή μνήμη δεδομένων. Όλες υπόλοιπες εντολές (ADD, SUB, MULT, AND, BRE, BRNE, BRM, BRA κ.λ.π.) **δεν** την χρησιμοποιούν.

Σήματα ελέγχου πολυπλέκτη A και B στην περίπτωση πολλών κ.ρ.



Στη μονάδα επεξεργασίας δεδομένων και τη σχετική μονάδας ελέγχου **των πολλών κ.ρ.** στον **πρώτο κ.ρ.** i) γίνεται πάντα η ανάγνωση της εντολής μέσα από την κρυφή μνήμη εντολών και ii) ο υπολογισμός του ΜΠ + 1 μέσω της ΑΛΜ. Για το λόγο αυτό στο συγκεκριμένο κ.ρ. πρέπει να περάσει από τον πολυπλέκτη A η τιμή του ΜΠ (επιλογή της τιμής «0» του σήματος ελέγχου) και από τον πολυπλέκτη B η τιμή «1» (επιλογή «01» των αντίστοιχων σημάτων ελέγχου). Στο **2^ο κ.ρ.** προκειμένου να υπολογιστεί η τιμή ΜΠ + d θα πρέπει να περάσει από τον πολυπλέκτη A **και πάλι** η τιμή του ΜΠ (επιλογή της τιμής «0» του σήματος ελέγχου) και από τον πολυπλέκτη B η τιμή d (επιλογή «10» των αντίστοιχων σημάτων ελέγχου).

Ενέργειες κατά τη διάρκεια 1^{ου} και 2^{ου} κ.ρ. στην περίπτωση πολλών κ.ρ.

Κατά τη διάρκεια του **1^{ου} κύκλου ρολογιού**, κατά την εκτέλεση μιας εντολής, λαμβάνουν χώρα **δύο ανεξάρτητες** ενέργειες:

1. Ανάγνωση της κρυφής μνήμης εντολών και η αποθήκευση της εντολής που διαβάστηκε στον καταχωρητή εντολών, KE.
2. Αύξηση του περιεχομένου του ΜΠ κατά «1».

Κατά τη διάρκεια του **2^{ου} κύκλου ρολογιού**, κατά την εκτέλεση μιας εντολής, λαμβάνουν χώρα **δύο ανεξάρτητες** ενέργειες:

1. Ανάγνωση των καταχωρητών γενικού σκοπού.
2. Υπολογισμός της διεύθυνσης ΜΠ + d.

Θέμα για πλεονεκτήματα/μειονεκτήματα προσκόμισης και εκτέλεσης κάθε εντολής σε 1 κ. ρ.

Να αναφέρετε τα **πλεονεκτήματα** και **μειονεκτήματα** μιας μονάδας επεξεργασίας δεδομένων σταθερής υποδιαστολής που η προσκόμιση και εκτέλεση κάθε εντολής γίνεται σε **ένα κύκλο ρολογιού**, σε σχέση με την αντίστοιχη μονάδα που η προσκόμιση και εκτέλεση κάθε εντολής γίνεται σε περισσότερους από ένα κύκλο ρολογιού.

Λύση

Το **πλεονέκτημα** της επιλογής η προσκόμιση και εκτέλεσης κάθε εντολής να γίνεται σε ένα κύκλο ρολογιού είναι η **ευκολία σχεδίασης της μονάδας ελέγχου**. Το **κύριο μειονέκτημα** της υλοποίησης στην οποία κάθε εντολή προσκομίζεται και εκτελείται σε ένα κύκλο ρολογιού, είναι ότι **η διάρκεια του κύκλου ρολογιού καθορίζεται από την εντολή που απαιτεί τον περισσότερο χρόνο για την εκτέλεση της**. Ένα ακόμα **μειονέκτημα** είναι ότι δεν έχει νόημα να εφαρμόσουμε τεχνικές που να κάνουν τις πιο συχνά εκτελούμενες εντολές να εκτελούνται πιο γρήγορα, αν αυτό δεν έχει ως συνέπεια να εκτελείται πιο γρήγορα και η πιο αργή εντολή. Επίσης, εάν για την προσκόμιση και εκτέλεση μιας εντολής απαιτείται να γίνουν δυο ή περισσότερες ταυτοτικές λειτουργίες π.χ. προσθέσεις, θα πρέπει να υπάρχει αντίστοιχος αριθμός λειτουργικών μονάδων για την εκτέλεση αυτών των λειτουργιών. Στις υλοποίησεις που οι εντολές εκτελούνται σε περισσότερους από ένα κύκλους, είναι δυνατόν οι ίδιες λειτουργίες να εκτελούνται σε **λιγότερες** λειτουργικές μονάδες σε διαφορετικούς κύκλους του ρολογιού. Αυτό οδηγεί στην **ελάττωση** του απαιτούμενου υλικού.

Θέμα για συνύπαρξη εντολών LOAD r1, (r2)/LOAD r1, d(r2) σε 1 κ. ρ./πολλούς κ.ρ.

i. Ένας κύκλος ρολογιού

Έστω μονάδα επεξεργασίας δεδομένων σταθερής υποδιαστολής όπου κάθε εντολή εκτελείται **σε ένα κύκλο ρολογιού**. Η μονάδα αυτή εκτελεί τις εντολές:

LOAD R1, (R2)

STORE R1, (R2)

ADD R1, R2, R3

SUB R1, R2, R3,

AND R1, R2, R3

BRE R1, R2, d.

a. Πως έγινε η επιλογή του σήματος της περιόδου χρονισμού;

b. Έχει νόημα στο σύνολο των εντολών του να υπάρχουν **και** οι δυο εντολές: LOAD R1, (R2), LOAD R1, d(R2) ή **STORE R1, (R2), STORE R1, d(R2)**;

Λύση



a. Η επιλογή της περιόδου του σήματος χρονισμού έγινε **έτσι ώστε η πιο αργή** από τις εντολές να προλαβαίνει να εκτελεστεί σε έναν κύκλο ρολογιού. Επίσης, η είσοδος ενεργοποίησης εγγραφής της κρυφής μνήμης δεδομένων να οδηγείται με τη ενεργό τιμή (λογική τιμή 1) αφού πρώτα οι είσοδοι διεύθυνσης και δεδομένων έχουν οδηγηθεί με την διεύθυνση στην οποία θα γίνει εγγραφή και τα δεδομένα που θα εγγραφούν.

b. Σε ένα υπολογιστή που η μονάδα επεξεργασίας δεδομένων σταθερής υποδιαστολής εκτελεί καθεμία **εντολή σε έναν κύκλο ρολογιού** δεν έχει νόημα στο σύνολο των εντολών να υπάρχει τόσο η εντολή **LOAD R1,(R2)** όσο και η εντολή **LOAD R1, d(R2)**. Η αιτία είναι αφενός ότι η εντολή LOAD R1,(R2) είναι **ισοδύναμη** της εντολής LOAD R1, 0(R2) και αφετέρου ότι οποιαδήποτε από αυτές τις εντολές εκτελείται στον ίδιο χρόνο, αφού κάθε εντολή εκτελείται σε ένα κύκλο ρολογιού. Επομένως, θα υλοποιήσουμε την εντολή LOAD R1, d(R2) που έχει περισσότερες δυνατότητες και για d=0 υλοποιεί και την εντολή LOAD R1, (R2).

ii. Πολλοί κύκλοι ρολογιού

Έστω μονάδα επεξεργασίας δεδομένων σταθερής υποδιαστολής όπου κάθε εντολή εκτελείται σε πολλούς κύκλο ρολογιού. Η μονάδα αυτή εκτελεί τις εντολές

LOAD R1, (R2)

STORE R1, (R2)

ADD R1, R2, R3,

SUB R1, R2, R3,

AND R1, R2, R3

BRE R1, R2, d.

a. Πως έγινε η επιλογή του **σήματος** της περιόδου **χρονισμού**;

b. Έχει νόημα στο σύνολο των εντολών του να υπάρχουν και οι δυο κάτωθι εντολές: LOAD R1, (R2) και LOAD R1, d(R2) ή STORE R1, (R2), STORE R1, d(R2);

Λύση

a. Για την εκτέλεση μιας εντολής απαιτείται η εκτέλεση μιας ακολουθίας λειτουργιών. Στη μονάδα επεξεργασίας δεδομένων σταθερής υποδιαστολής στην οποία κάθε εντολή εκτελείται σε διαφορετικό αριθμό κύκλων ρολογιού ανάλογα της πολυπλοκότητας της, **οι λειτουργίες ομαδοποιούνται και κάθε ομάδα λειτουργιών εκτελείται σε ένα κύκλο ρολογιού. Η ομάδα λειτουργιών, η εκτέλεση των οποίων απαιτεί το μεγαλύτερο χρονικό διάστημα** καθορίζει τη διάρκεια της περιόδου του σήματος χρονισμού, δηλαδή του ρολογιού.

b. Έχει νόημα διότι για την εκτέλεση της εντολής LOAD R1, (R2) απαιτούνται 4 κύκλοι ρολογιού, ενώ για την εκτέλεση της εντολής LOAD R1, d(R2) απαιτούνται 5 κύκλοι ρολογιού. Επομένως, αν και η εντολή LOAD R1,(R2) θα μπορούσε να αντικατασταθεί από την εντολή LOAD R1, 0(R2) ο χρόνος εκτέλεσής της θα αυξάνονταν κατά ένα κύκλο ρολογιού.

Θέμα Θεωρίας Ιουνίου 2019 με έναν/πολλούς κύκλους ρολογιού

Δίνονται οι εντολές: a. LOAD R1, (R2) //R1 ← M(R2) και b. LOAD R1, d(R2) //R1 ← M[(R2)+d]. Στο σύνολο των εντολών του επεξεργαστή **δεν** έχει νόημα να υπάρχει η εντολή **α** τόσο στην περίπτωση που η μονάδα επεξεργασίας δεδομένων εκτελεί κάθε εντολή σε ένα κύκλο ρολογιού όσο και στην περίπτωση που το πλήθος των κύκλων ρολογιού που απαιτούνται για την εκτέλεση μιας εντολής εξαρτάται από την πολυπλοκότητά της.

a) Σωστό

β) **Λάθος**

Λύση

Είναι **λάθος**, αφού η εντολή LOAD R1, (R2) έχει νόημα να υπάρχει **στην περίπτωση των πολλών κ.ρ., καθώς στον ένα κ.ρ. μπορεί ισοδύναμα να αντικατασταθεί από την εντολή LOAD R1, d(R2), για d = 0**. Αντίθετα, στους πολλούς κ.ρ. οι εντολές LOAD R1, (R2) και LOAD R1, d(R2) **δεν** είναι ισοδύναμες ακόμη και για d = 0, αφού η πρώτη χρειάζεται 4 κ.ρ. ενώ η δεύτερη 5 κ.ρ. Αυτά ακριβώς ισχύουν και για τις εντολές STORE R1, (R2) και STORE R1, d(R2).

Θέμα Θεωρίας Ιουνίου 2019 με ένα/πολλούς κύκλους ρολογιού



1. Της μονάδας επεξεργασίας δεδομένων σταθερής υποδιαστολής της οποίας κάθε εντολή εκτελείται σε τόσους κύκλους ρολογιού όσους απαιτείται, εάν κατά την προσκόμιση ή/και εκτέλεση της εντολής απαιτείται να γίνουν δύο ταυτοτικές λειτουργίες (π.χ. δύο προσθέσεις) δεν είναι πάντα αναγκαίο να υπάρχει αντίστοιχο πλήθος λειτουργικών μονάδων για την εκτέλεση αυτών των λειτουργιών.

- α) Σωστό
- β) **Λάθος**

Λύση

Είναι λάθος, **Η σωστή απάντηση είναι ότι δεν είναι ποτέ αναγκαίο να υπάρχει αντίστοιχο πλήθος λειτουργικών μονάδων (π.χ. αθροιστές) για την εκτέλεση αυτών των λειτουργιών.**

Θέμα Θεωρίας Ιουνίου 2019 με ένα/πολλούς κύκλους ρολογιού

Της μονάδας επεξεργασίας δεδομένων σταθερής υποδιαστολής της οποίας κάθε εντολή εκτελείται σε ένα κύκλο ρολογιού, εάν κατά την προσκόμιση και εκτέλεση της εντολής απαιτείται να γίνουν δύο ταυτοτικές λειτουργίες (π.χ. δύο προσθέσεις) δεν είναι ποτέ αναγκαίο να υπάρχει αντίστοιχο πλήθος λειτουργικών μονάδων για την εκτέλεση αυτών των λειτουργιών.

- α) Σωστό
- β) **Λάθος**

Λύση

Λάθος, είναι **πάντα** αναγκαίο στην περίπτωση αυτή.

Θέμα Θεωρίας Ιουνίου 2016 με έναν και πολλούς κύκλος ρολογιού

Θεωρείστε μία μονάδα επεξεργασίας δεδομένων σταθερής υποδιαστολής. Να σημειώσετε αν έχει νόημα να υπάρχουν και οι δύο εντολές LOAD r1, (r2) [$r1 \leftarrow M(r2)$], LOAD r1, d(r2) [$r1 \leftarrow M((r2)+d)$], σε κάθε μία από τις κάτωθι περιπτώσεις και να αιτιολογήσετε το πολύ σε πέντε γραμμές την απάντησή σας.

Σ/Λ. α. Έχει νόημα στην περίπτωση που κάθε εντολή εκτελείται σε ένα κύκλο ρολογιού.

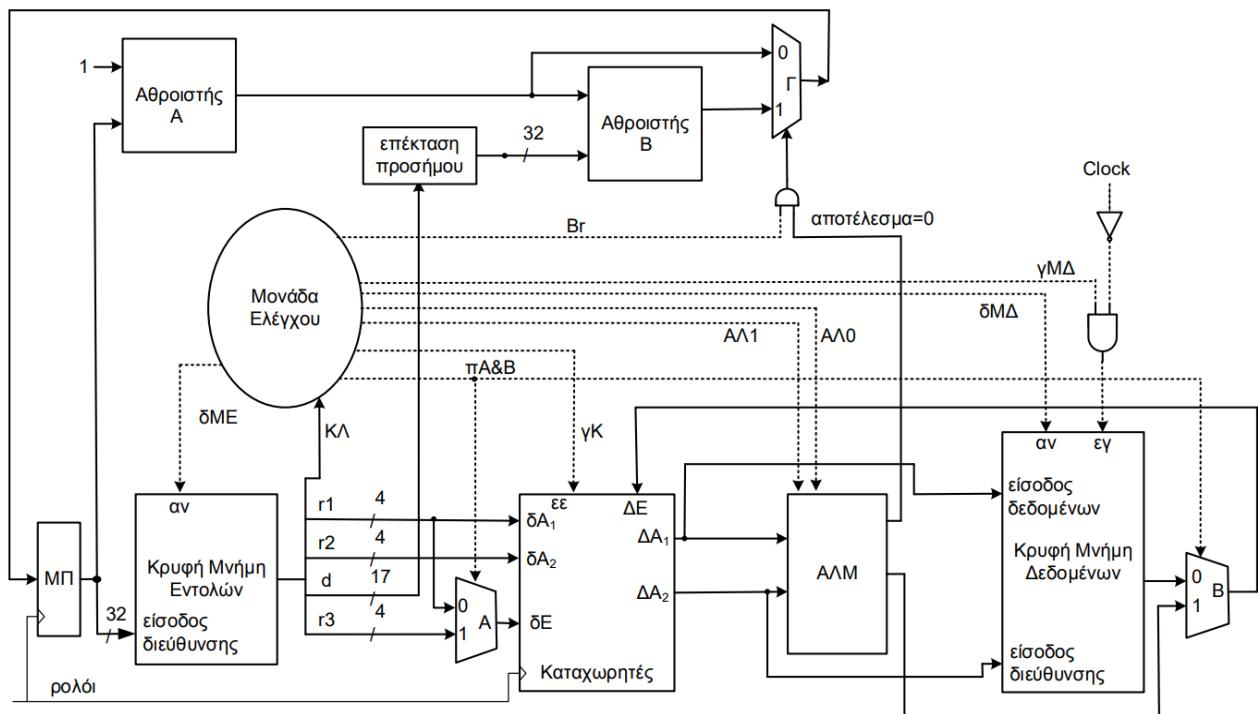
Σ/Λ β. Έχει νόημα στην περίπτωση που κάθε εντολή εκτελείται σε τόσους κύκλους ρολογιού όσους απαιτείται.

Λύση

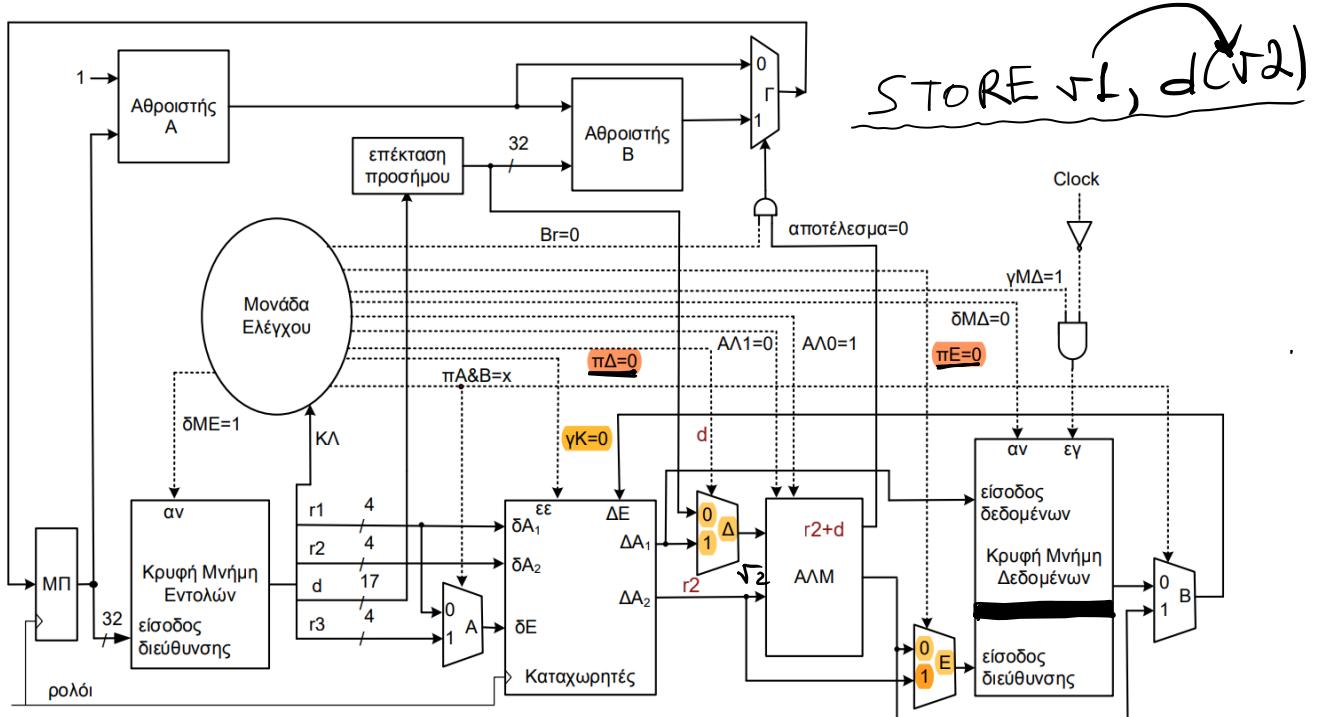
Η σωστή απάντηση είναι το β. Τότε έχει νόημα, διότι για την εκτέλεση της εντολής LOAD r1, (r2) απαιτούνται τέσσερις κ.ρ., ενώ για την εκτέλεση της εντολής LOAD r1, d(r2) απαιτούνται πέντε κ.ρ. Επομένως, αν και η εντολή LOAD r1, (r2) θα μπορούσε να αντικατασταθεί από την εντολή LOAD r1, 0(r2), ο χρόνος εκτέλεσης εντολών θα αυξανόταν κατά ένα κ.ρ.

Θέμα Ιουνίου 2016 με τροποποίηση μονάδας επεξεργασίας δεδομένων για ένα κ.ρ.

Η επόμενη μονάδα επεξεργασίας δεδομένων σταθερής υποδιαστολής έχει σχεδιαστεί για την εκτέλεση των εντολών: LOAD r1, (r2) [$r1 \leftarrow M(r2)$], STORE r1, (r2) [$r1 \rightarrow M(r2)$], ADD r1, r2, r3 [$r1+r2 \rightarrow r3$], SUB r1, r2, r3 [$r1-r2 \rightarrow r3$], AND r1, r2, r3 [$r1 \wedge r2 \rightarrow r3$], και BRE r1, r2, d [εάν $r1-r2=0$ τότε $MPI=MPI + d$, όπου d αριθμός των 17 δυαδικών ψηφίων σε αναπαράσταση συμπληρώματος ως προς 2], στις αγκύλες είναι η επεξήγηση της εντολής. Να κάνετε, πάνω στο σχήμα (δεν χρειάζεται αιτιολόγηση), **τις απαιτούμενες τροποποιήσεις** ώστε η μονάδα να μπορεί να εκτελέσει και την εντολή **STORE r1, d(r2) [r1 → M((r2)+d)]**, ο αριθμός d έχει τα ίδια χαρακτηριστικά με τον αριθμό d της εντολής BRE] και να θέσετε τις τιμές των σημάτων που απαιτούνται για την εκτέλεση αυτής της εντολής.



Λύση



Θέμα Σεπτεμβρίου 2019 με τροποποίηση μονάδας επεξεργασίας δεδομένων για πολλούς κ.ρ.

Θεωρείστε επεξεργαστή που διαθέτει τη μονάδα επεξεργασίας δεδομένων σταθερής υποδιαστολής του επόμενου σχήματος.

Μεταξύ των εντολών που μπορούν να εκτελεστούν περιλαμβάνονται και οι κάτωθι εντολές:

STORE r1,d(r2) //M (r2+d) ← r1

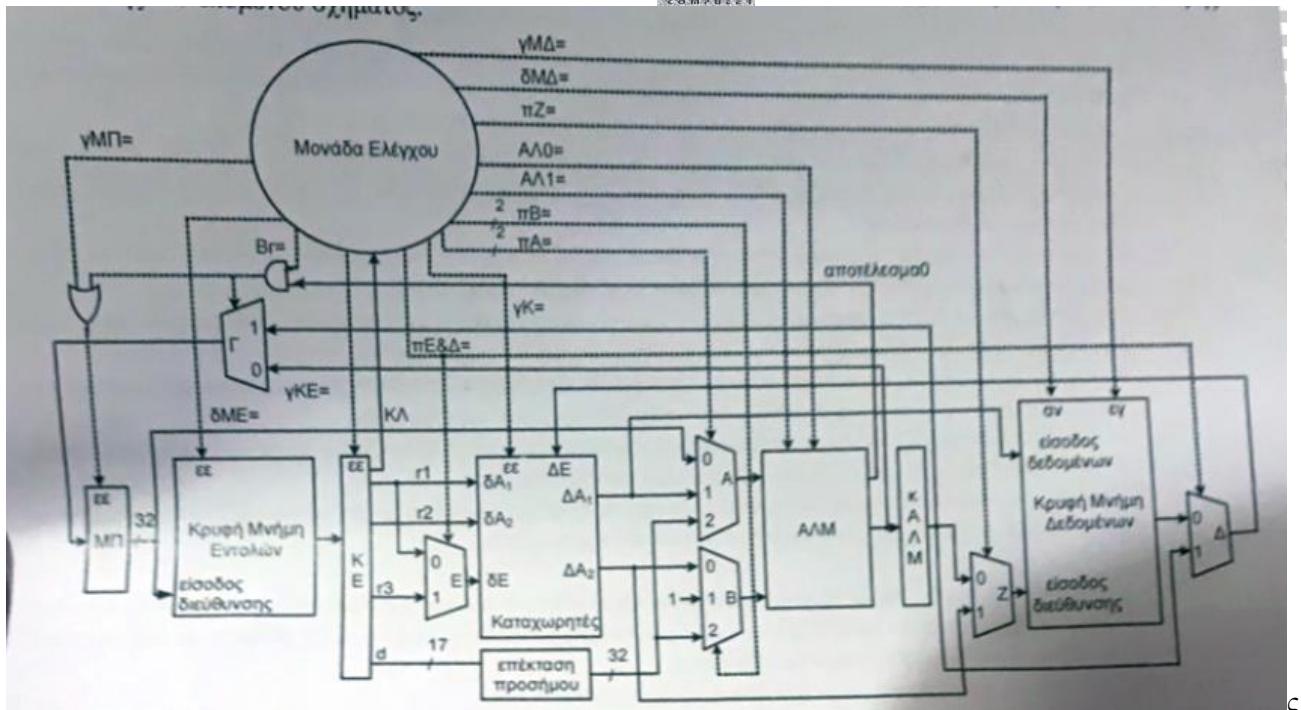
ADD r1,r2,r3 //r1+r2 → r3

SUB r1,r2,r3 // r1-r2 → r3

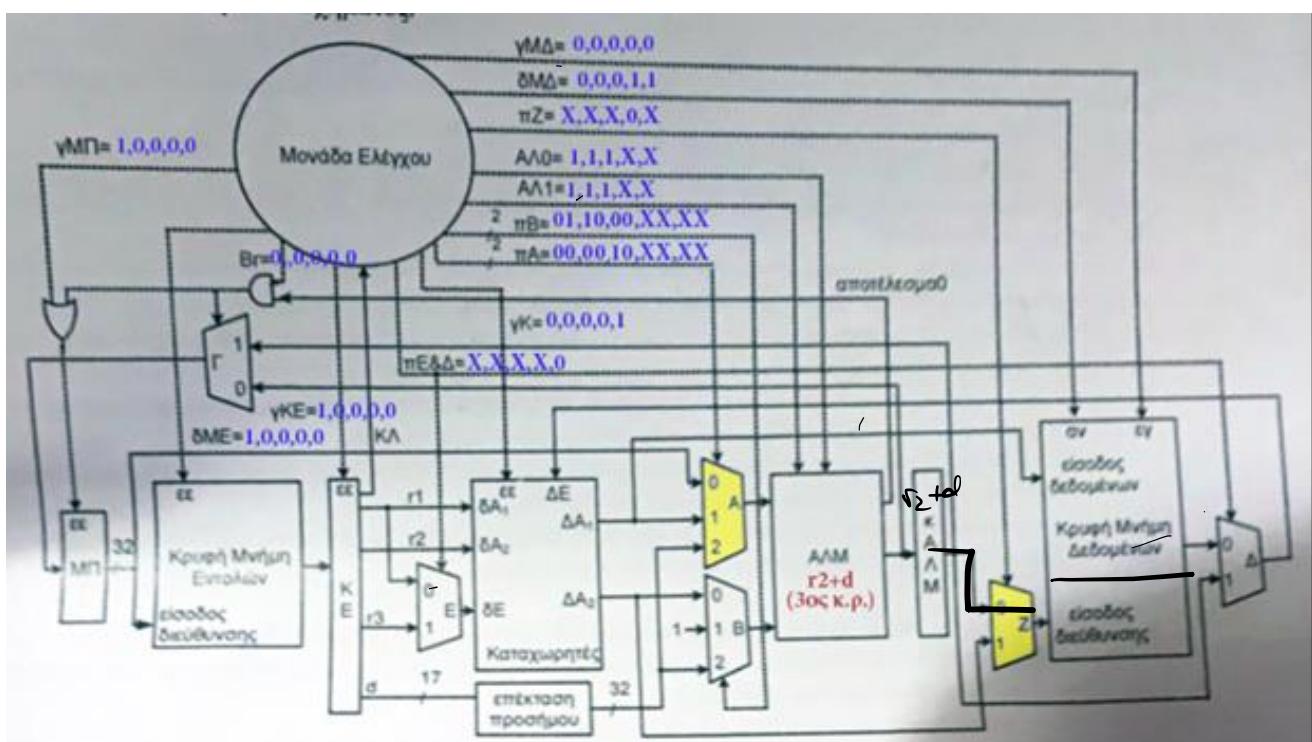
και BRE r1,r2,d // εάν r1-r2=0 τότε MΠ = MΠ + d

Να βάλετε πάνω στο σχήμα τις τιμές των σημάτων που απαιτούνται για την προσπέλαση και εκτέλεση της εντολής

LOAD r1,d(r2). Για πρόσθεση να χρησιμοποιηθούν οι τιμές **ΑΛ0 = 1**, **ΑΛ1 = 1**. Ο κωδικός λειτουργίας της εντολής **LOAD** είναι 11001. Δεν χρειάζεται αιτιολόγηση.



Λύση

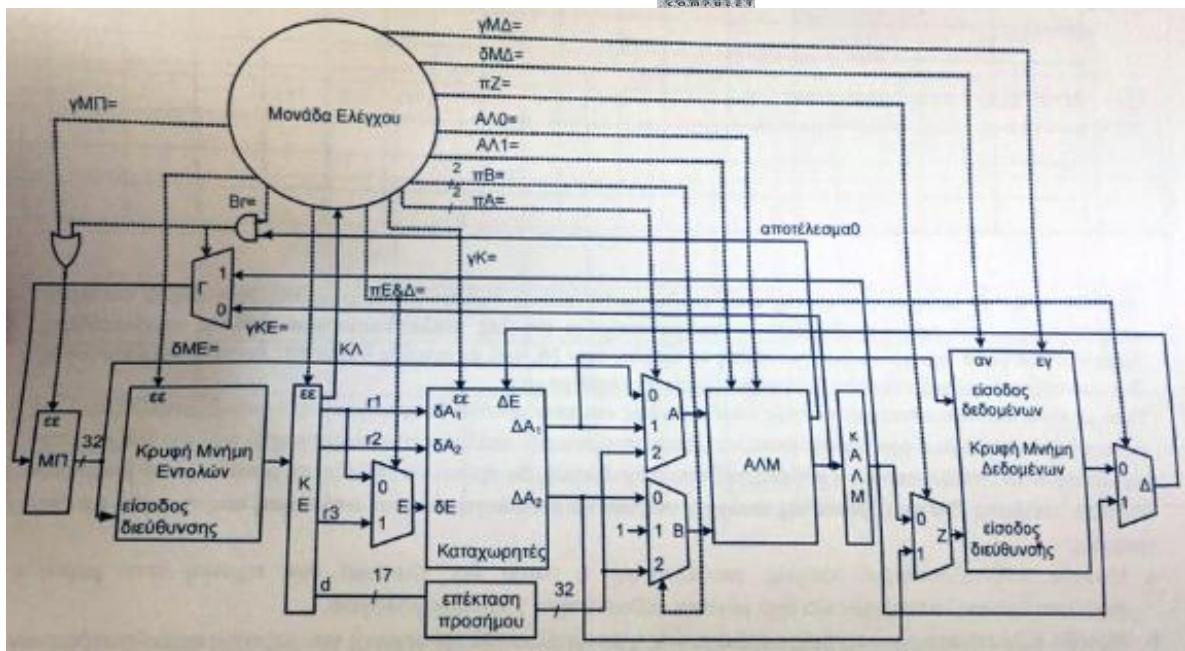


Θέμα Ιουνίου 2019 με τροποποίηση μονάδας επεξεργασίας δεδομένων για πολλούς κ.ρ.

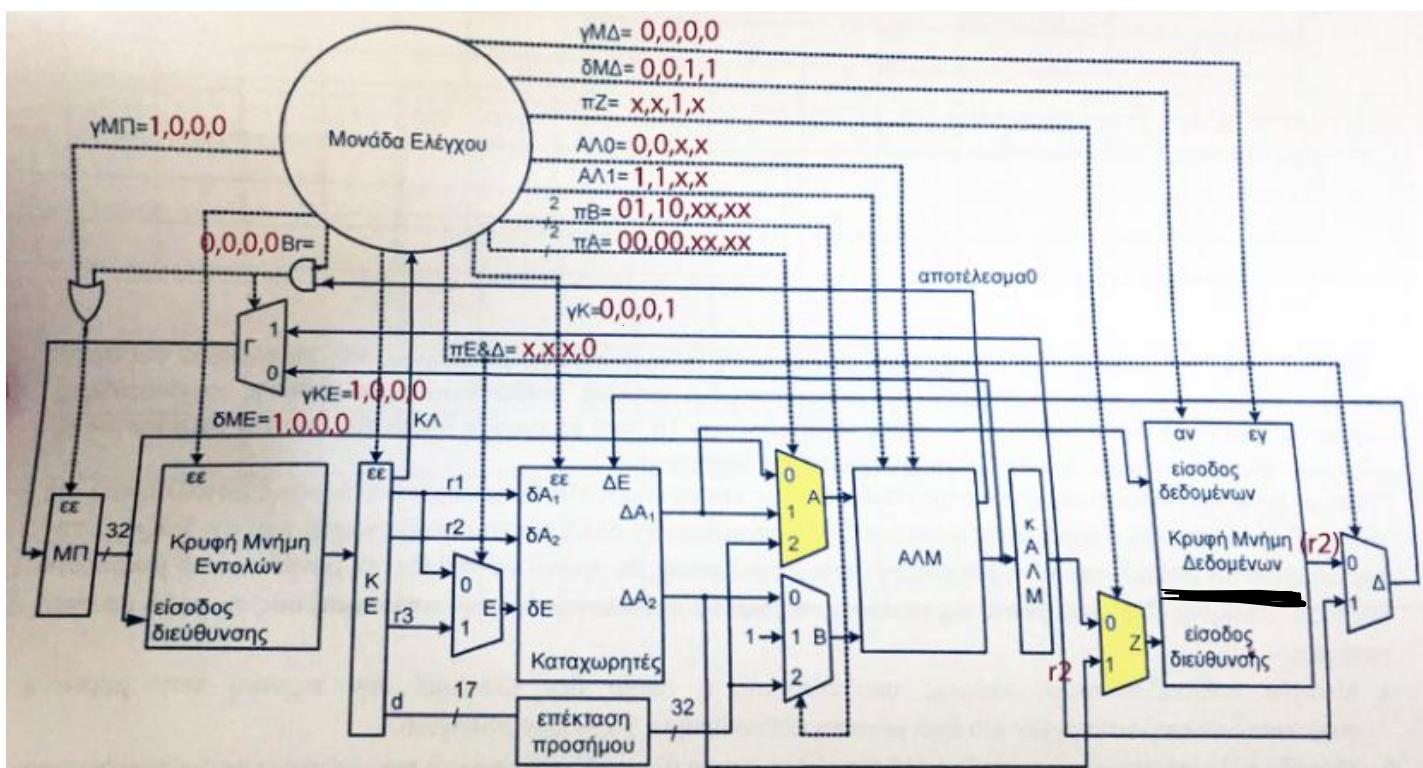
Στη μονάδα του επόμενου σχήματος μπορούν να εκτελεστούν μεταξύ άλλων οι εντολές:

LOAD r1,(r2)	//r1 \leftarrow M (r2)
STORE r1,(r2)	//r1 \rightarrow M (r2)
STORE r1,d(r2)	//r1 \rightarrow M (d+r2)
ADD r1,r2,r3	//r1+r2 \rightarrow r3
και BRE r1,r2,d	// εάν r1-r2=0 τότε MΠ=MΠ+d

Να βάλετε πάνω στο σχήμα τις τιμές των σημάτων που απαιτούνται για την προσπέλαση και την εκτέλεση της εντολής **LOAD r1, (r2)** η οποία εκτελείται σε 4 κύκλους ρολογιού. Αριστερά βρίσκεται η τιμή που αντιστοιχεί στον 1ο κύκλο ρολογιού. Για πρόσθεση να χρησιμοποιηθούν οι τιμές ΑΛ0=0, ΑΛ1=1. Ο κωδικός λειτουργίας της εντολής LOAD είναι 01111. Δεν χρειάζεται αιτιολόγηση.



Λύση

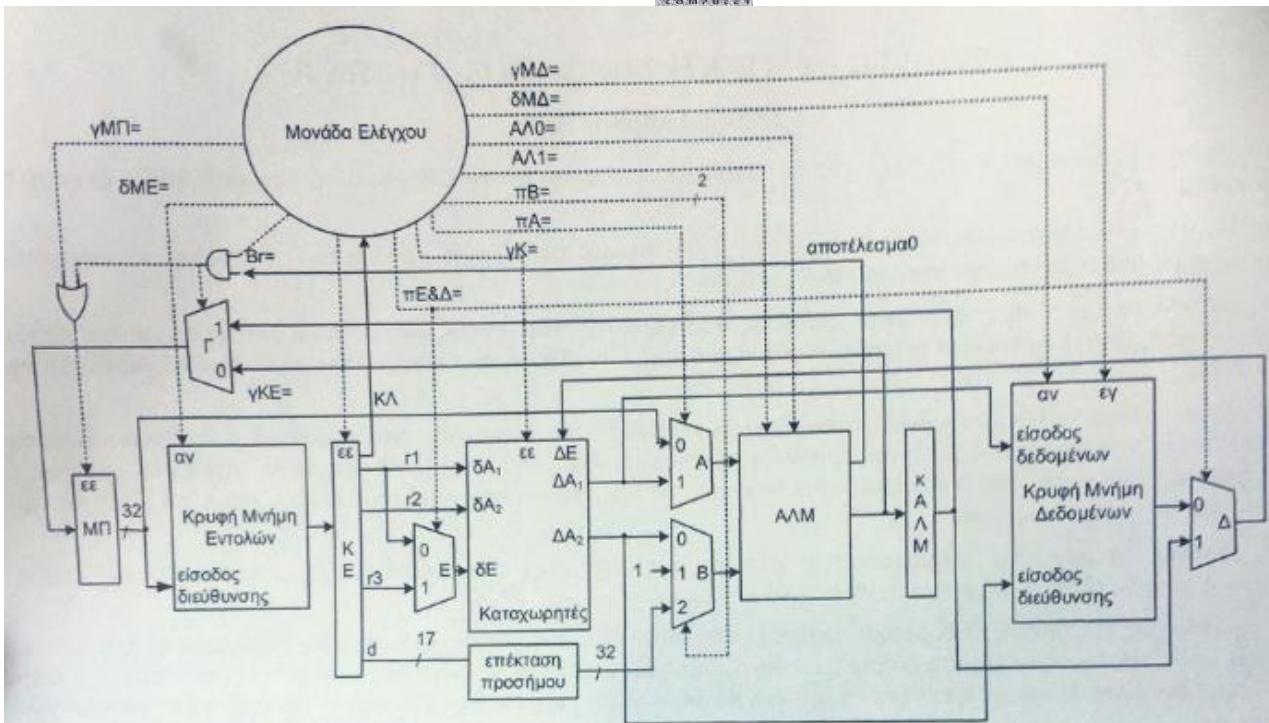


Θέμα Σεπτεμβρίου 2016 με μονάδα επεξεργασίας δεδομένων για πολλούς κ.ρ.

Δίνεται η επόμενη μονάδα επεξεργασίας δεδομένων σταθερής υποδιαστολής.

a. Να δηλώσετε και αιτιολογήσετε σε πέντε το πολύ γραμμές εάν εκτελεί κάθε εντολή σε έναν κύκλο ρολογιού ή σε περισσότερους ανάλογα με την πολυπλοκότητα της εντολής.

β. Να σημειώσετε πάνω στο σχήμα τις τιμές που πρέπει να έχουν τα σήματα ελέγχου για την εκτέλεση της εντολής STORE r1, (r2) [r1 → M(r2)] (για το ερώτημα β δεν απαιτείται καμία αιτιολόγηση)



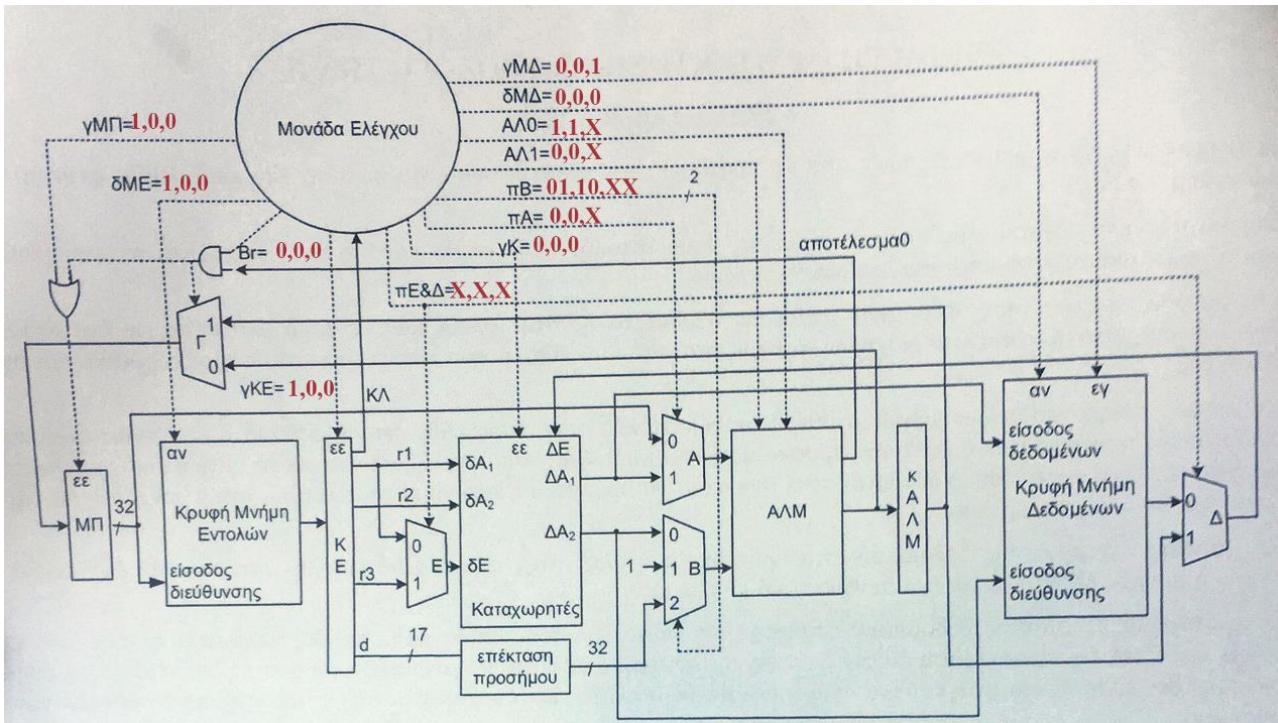
Άνση

α. Κάθε εντολή εκτελείται σε περισσότερους κ.ρ., διότι:

i) Δεν υπάρχουν οι αθροιστές που υπάρχουν στον έναν κύκλο ρολογιού.

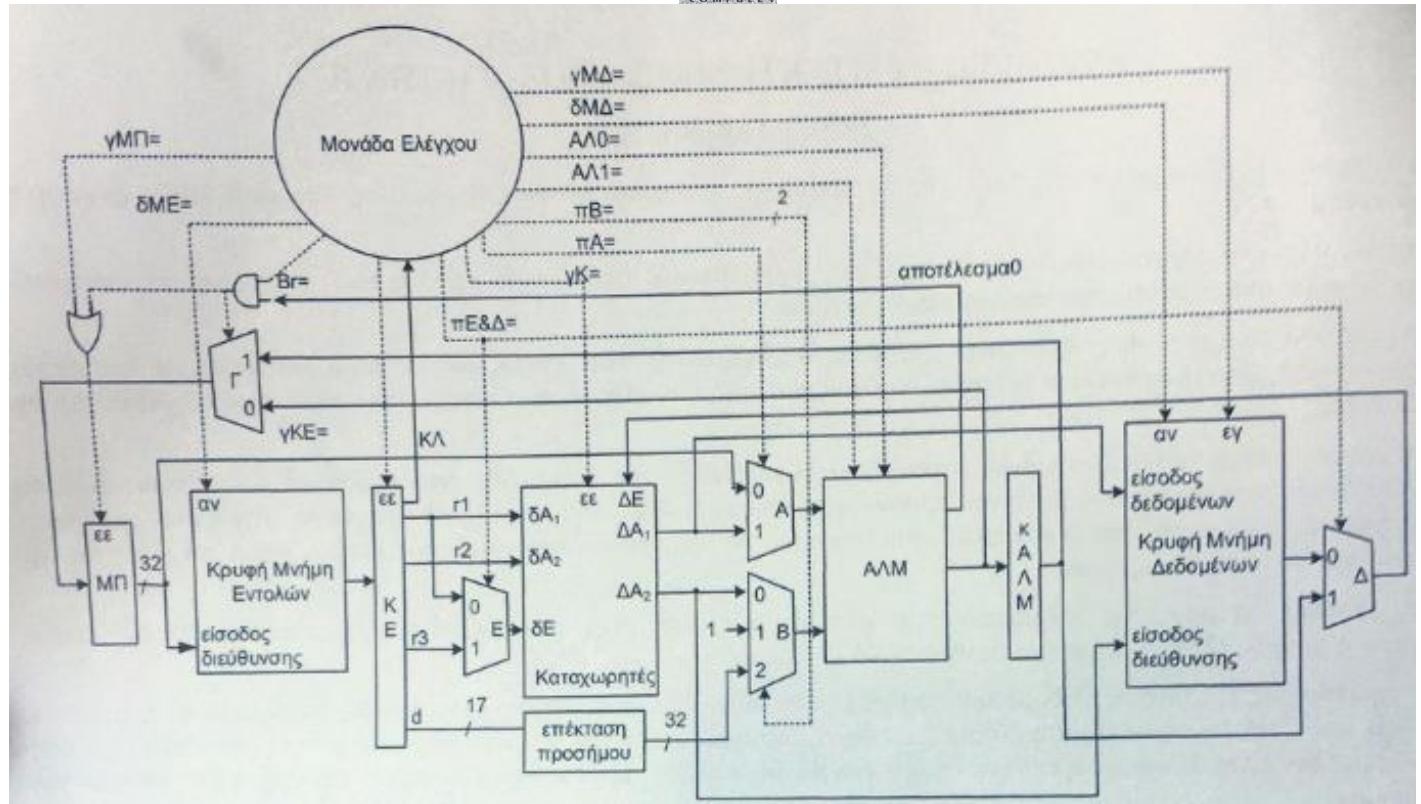
ii) Επίσης, υπάρχουν οι καταχωρητές ειδικού σκοπού KE και κΑΛΜ

β. Τιμές σημάτων ελέγχου για την εκτέλεση της εντολής STORE $r1$, $(r2)$

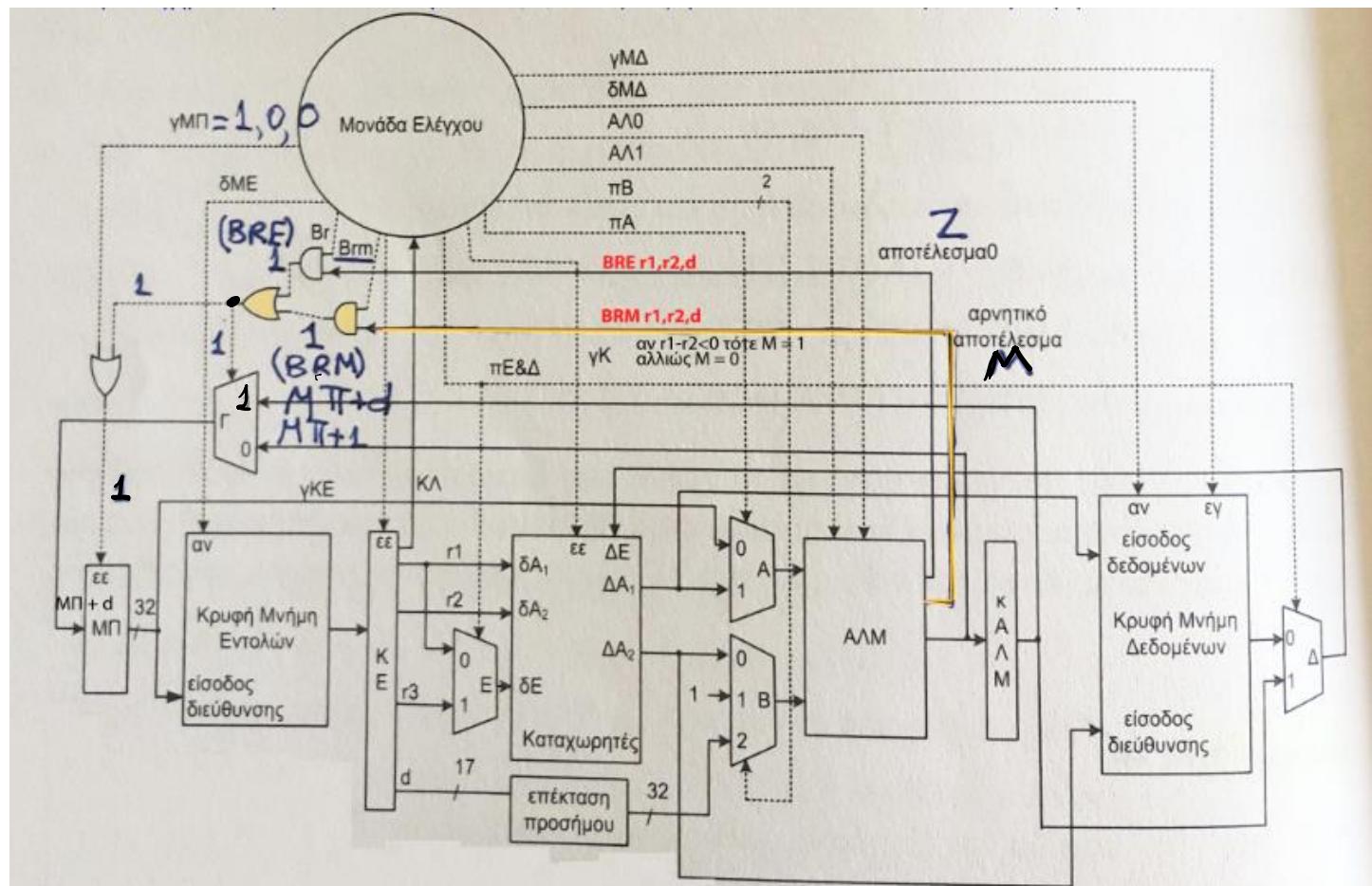


Θέμα Σεπτεμβρίου 2014 με τροποποίηση μονάδα επεξεργασίας δεδομένων για πολλούς κ.ρ.

Δίνεται η επόμενη μονάδα επεξεργασίας δεδομένων σταθερής υποδιαστολής που μπορεί να εκτελέσει οποιαδήποτε από τις κάτωθι εντολές: LOAD $r1, (r2)$ [$r1 \leftarrow M(r2)$], STORE $r1, (r2)$ [$r1 \rightarrow M(r2)$], ADD $r1, r2, r3$ [$r1+r2 \rightarrow r3$], SUB $r1, r2, r3$ [$r1-r2 \rightarrow r3$], AND $r1, r2, r3$ [$r1 \wedge r2 \rightarrow r3$] και BRE $r1, r2, d$ [εάν $r1-r2=0$ τότε $M_P=M_P+d$]. Να κάνετε τις απαιτούμενες τροποποιήσεις ώστε η μονάδα να μπορεί να εκτελέσει και την εντολή BRM $r1, r2, d$ [εάν $r1-r2<0$ τότε $M_P=M_P+d$].



Αύση

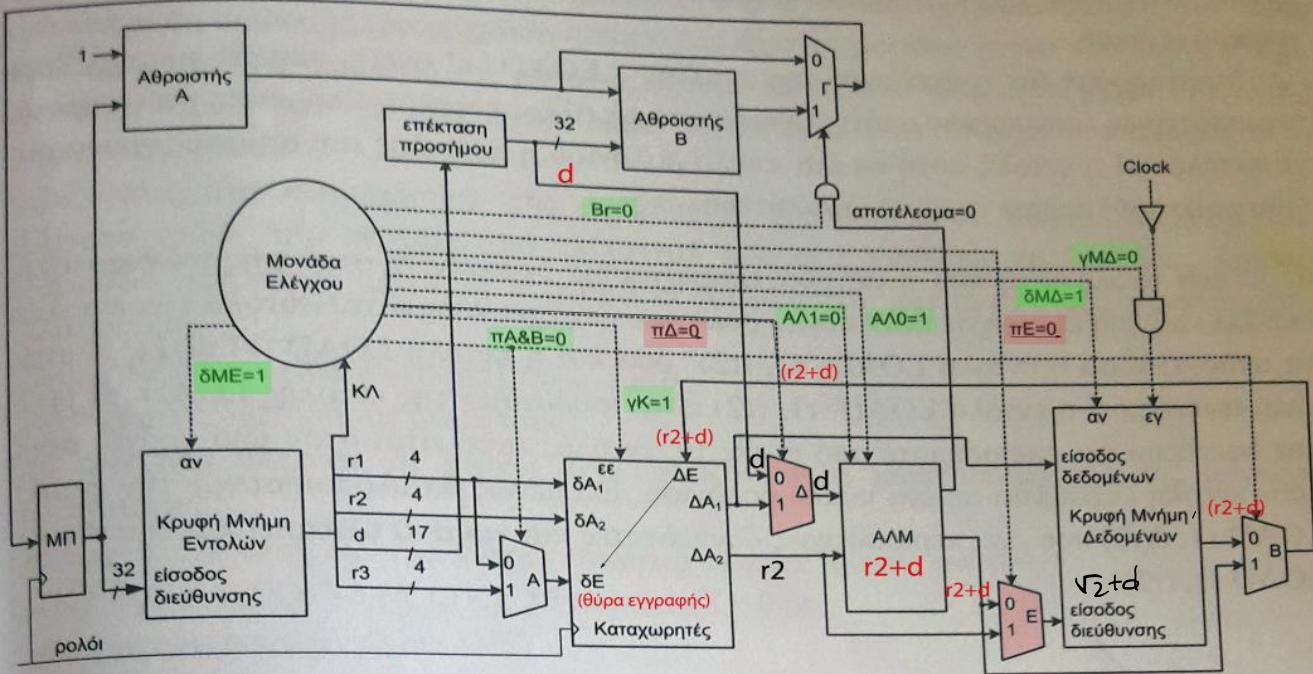


Τροποποίηση μονάδες ελέγχου για ένα/πολλούς κ.ρ. για εκτέλεση εντολής LOAD r1, d(r2)

I. Ένας κύκλος ρολογιού

Σε μονάδα ελέγχου που εκτελεί κάθε εντολή **σε ένα κύκλο ρολογιού**, να κάνετε τις απαιτούμενες τροποποιήσεις ώστε επιπλέον των κανονικών εντολών, να εκτελείται και η εντολή: LOAD r1, d(r2) // $r1 \leftarrow M[r2] + d$

Οι τιμές των σημάτων ελέγχου είναι για την περίπτωση εκτέλεσης της εντολής LOAD r1, d(r2)

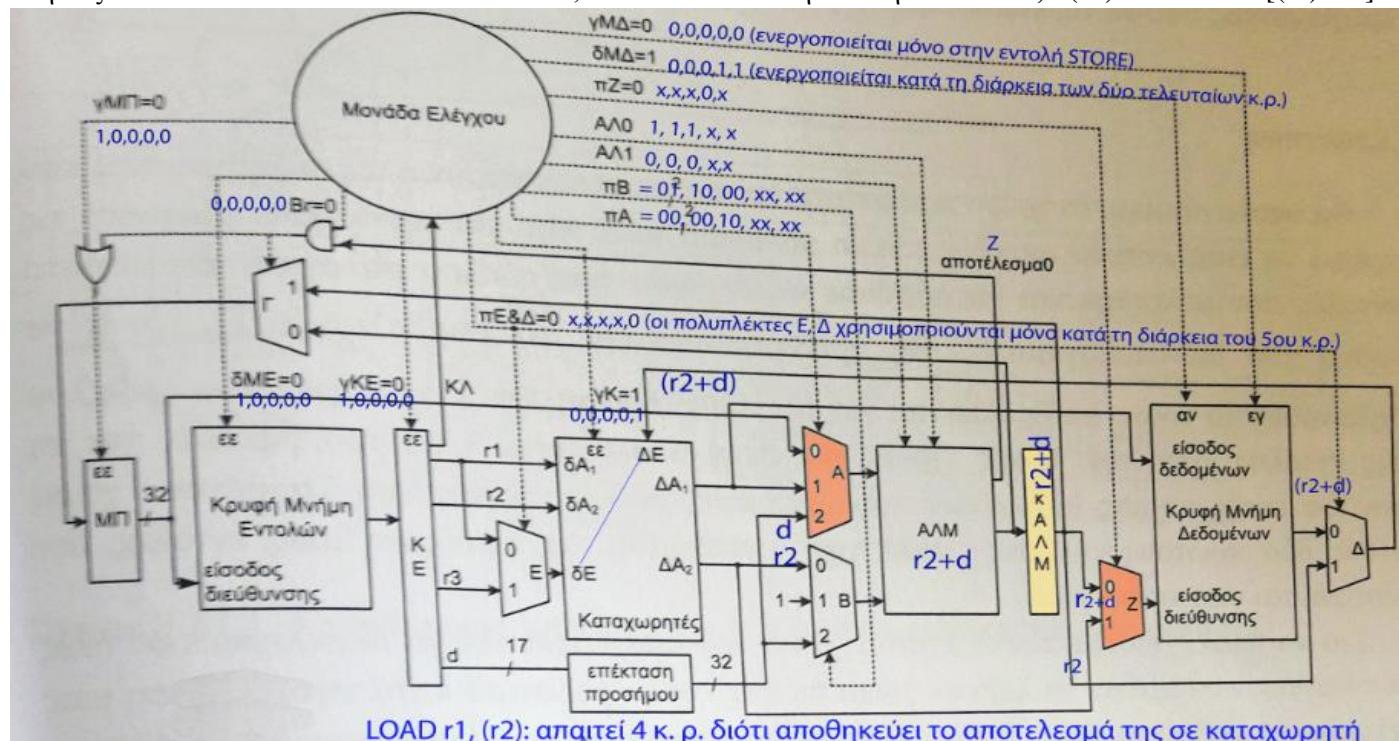


LOAD r1, d(r2) //r1 <- M(r2 + d)

Ο λόγος που προστέθηκε ο πολυπλέκτης Ε είναι για να υποστηρίζεται τόσο η εκτέλεση της κανονικής εντολής LOAD r1,(r2), όπου τότε το πΕ = 1, για να περάσει ο r2, όσο και η εντολή LOAD r1,d(r2), όπου τότε το πΕ = 0, για να περάσει το άθροισμα r2 + d,

II. Πολλοί κύκλοι ρολογιού

Σε μονάδα ελέγχου που εκτελεί κάθε εντολή **σε περισσότερους κύκλους ρολογιού**, να κάνετε τις απαιτούμενες τροποποιήσεις ώστε επιπλέον των κανονικών εντολών, να εκτελείται και η εντολή: **LOAD r1, d(r2) //r1 <- M[(r2) + d]**



LOAD r1, (r2): απαιτεί 4 κ. ρ. διότι αποθηκεύει το αποτέλεσμά της σε καταχωρητή

Θέμα Σεπτεμβρίου 2016 με Σ/Λ για ένα/πολλούς κ.ρ.

Να σημειώσετε ποιες από τις προτάσεις που ακολουθούν είναι σωστές.

A. Στη μονάδα επεξεργασίας δεδομένων σταθερής υποδιαστολής στην οποία κάθε εντολή εκτελείται σε ένα κύκλο ρολογιού, η μονάδα ελέγχου είναι ακολουθιακό κύκλωμα. → Λ (αυτό ισχύει για τους πολλούς κ.ρ.)

B. Στη μονάδα επεξεργασίας δεδομένων σταθερής υποδιαστολής στην οποία κάθε εντολή εκτελείται σε ένα κύκλο ρολογιού, η διάρκεια του κύκλου ρολογιού καθορίζεται από την εντολή που απαιτεί τον περισσότερο χρόνο για την εκτέλεσή της. → Σ (αυτό είναι μειονέκτημα του ενός κ.ρ.)

Γ. Στη μονάδα επεξεργασίας δεδομένων σταθερής υποδιαστολής στην οποία κάθε εντολή εκτελείται σε τόσους κύκλους ρολογιού όσους απαιτείται, εάν κατά την προσκόμιση ή/και εκτέλεση της εντολής απαιτείται να γίνουν δύο ταυτοτικές λειτουργίες (πχ. δύο προσθέσεις) δεν είναι ποτέ αναγκαίο να υπάρχει αντίστοιχο πλήθος λειτουργικών μονάδων για την εκτέλεση αυτών των λειτουργιών. → Σ (αυτό ισχύει στην περίπτωση του ενός κ.ρ.)

Δ. Στη μονάδα επεξεργασίας δεδομένων σταθερής υποδιαστολής στην οποία κάθε εντολή εκτελείται σε ένα κύκλο ρολογιού, η μονάδα ελέγχου είναι ένα συνδυαστικό κύκλωμα. → Σ

Ε. Στη μονάδα επεξεργασίας δεδομένων σταθερής υποδιαστολής στην οποία κάθε εντολή εκτελείται σε ένα κύκλο ρολογιού, εάν κατά την προσκόμιση ή/και εκτέλεση της εντολής απαιτείται να γίνουν δύο ταυτοτικές λειτουργίες (πχ. δύο προσθέσεις) δεν είναι πάντα αναγκαίο να υπάρχει αντίστοιχο πλήθος λειτουργικών μονάδων για την εκτέλεση αυτών των λειτουργιών. → Λ (είναι πάντα αναγκαίο στον 1 κ.ρ.)

Ζ. Στη μονάδα επεξεργασίας δεδομένων σταθερής υποδιαστολής στην οποία κάθε εντολή εκτελείται σε τόσους κύκλους ρολογιού όσους απαιτείται, η μονάδα ελέγχου είναι ακολουθιακό κύκλωμα. → Σ

Η. Στη μονάδα επεξεργασίας δεδομένων σταθερής υποδιαστολής στην οποία κάθε εντολή εκτελείται σε τόσους κύκλους ρολογιού όσους απαιτείται, η μονάδα ελέγχου είναι ένα συνδυαστικό κύκλωμα. → Λ

Θ. Στη μονάδα επεξεργασίας δεδομένων σταθερής υποδιαστολής στην οποία κάθε εντολή εκτελείται σε τόσους κύκλους ρολογιού όσους απαιτείται, εάν κατά την προσκόμιση ή/και εκτέλεση της εντολής απαιτείται να γίνουν δύο ταυτοτικές λειτουργίες (πχ. δύο προσθέσεις) είναι πάντα αναγκαίο να υπάρχει αντίστοιχο πλήθος λειτουργικών μονάδων για την εκτέλεση αυτών των λειτουργιών. → Λ

Ι. Στη μονάδα επεξεργασίας δεδομένων σταθερής υποδιαστολής στην οποία κάθε εντολή εκτελείται σε ένα κύκλο ρολογιού, δεν έχει νόημα να εφαρμόσουμε τεχνικές που κάνουν τις πιο συχνά εκτελούμενες εντολές να εκτελούνται πιο γρήγορα αν αυτό δεν έχει ως συνέπεια να εκτελείται πιο γρήγορα και η πιο αργή εντολή. → Σ

Κ. Στη μονάδα επεξεργασίας δεδομένων σταθερής υποδιαστολής στην οποία κάθε εντολή εκτελείται σε ένα κύκλο ρολογιού, εάν κατά την προσκόμιση ή/και εκτέλεση της εντολής απαιτείται να γίνουν δύο ταυτοτικές λειτουργίες (πχ. δύο προσθέσεις) δεν είναι ποτέ αναγκαίο να υπάρχει αντίστοιχο πλήθος λειτουργικών μονάδων για την εκτέλεση αυτών των λειτουργιών. → Λ

Λ. Στη μονάδα επεξεργασίας δεδομένων σταθερής υποδιαστολής στην οποία κάθε εντολή εκτελείται σε τόσους κύκλους ρολογιού όσους απαιτείται, η διάρκεια του κύκλου ρολογιού καθορίζεται από την εντολή που απαιτεί τον περισσότερο χρόνο για την εκτέλεσή της. → Λ

Μ. Στη μονάδα επεξεργασίας δεδομένων σταθερής υποδιαστολής στην οποία κάθε εντολή εκτελείται σε τόσους κύκλους ρολογιού όσους απαιτείται, δεν έχει νόημα να εφαρμόσουμε τεχνικές που κάνουν τις πιο συχνά εκτελούμενες εντολές να εκτελούνται πιο γρήγορα αν αυτό δεν έχει ως συνέπεια να εκτελείται πιο γρήγορα και η πιο αργή εντολή. → Λ

Ν. Στη μονάδα επεξεργασίας δεδομένων σταθερής υποδιαστολής στην οποία κάθε εντολή εκτελείται σε τόσους κύκλους ρολογιού όσους απαιτείται, εάν κατά την προσκόμιση ή/και εκτέλεση της εντολής απαιτείται να γίνουν δύο ταυτοτικές λειτουργίες (πχ. δύο προσθέσεις) δεν είναι πάντα αναγκαίο να υπάρχει αντίστοιχο πλήθος λειτουργικών μονάδων για την εκτέλεση αυτών των λειτουργιών. → Λ (δεν είναι ποτέ αναγκαίο)

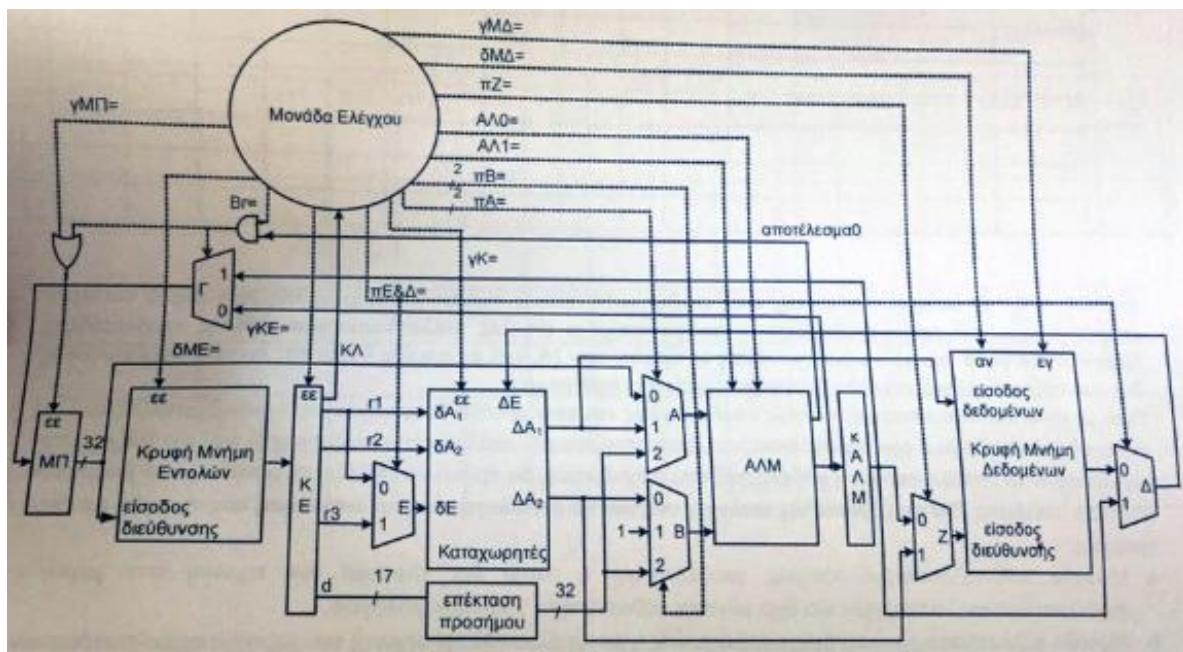
Ξ. Στη μονάδα επεξεργασίας δεδομένων σταθερής υποδιαστολής στην οποία κάθε εντολή εκτελείται σε ένα κύκλο ρολογιού, εάν κατά την προσκόμιση ή/και εκτέλεση της εντολής απαιτείται να γίνουν δύο ταυτοτικές λειτουργίες (πχ. δύο προσθέσεις) είναι πάντα αναγκαίο να υπάρχει αντίστοιχο πλήθος λειτουργικών μονάδων για την εκτέλεση αυτών των λειτουργιών. → Σ

Θέμα με τροποποίηση μονάδας επεξεργασίας δεδομένων για πολλούς κ.ρ.

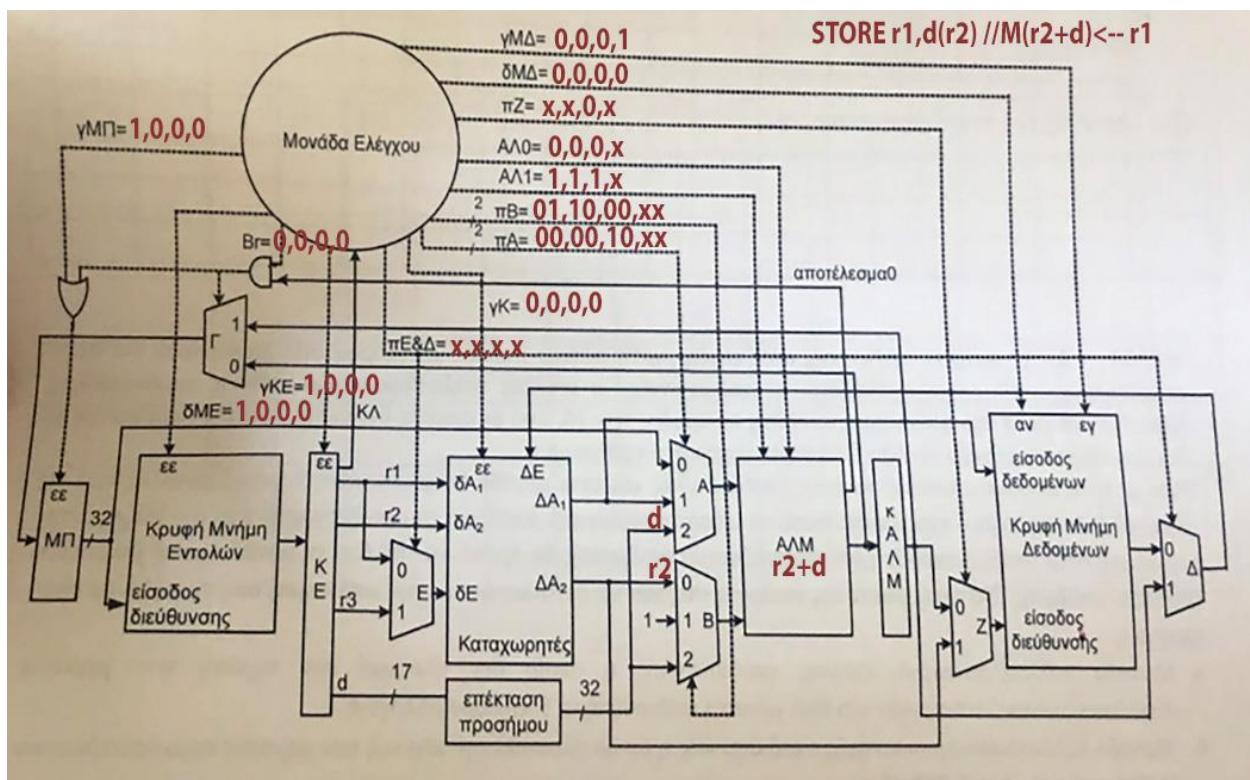
Στη μονάδα του επόμενου σχήματος μπορούν να εκτελεστούν μεταξύ άλλων οι εντολές:

LOAD r1,(r2)	// $r1 \leftarrow M(r2)$
STORE r1,(r2)	// $r1 \leftarrow M(r2)$
STORE r1,d(r2)	// $r1 \rightarrow M(d+r2)$
ADD r1,r2,r3	// $r1+r2 \rightarrow r3$
και BRE r1,r2,d	// εάν $r1=r2=0$ τότε $MPI=MPI+d$

Να βάλετε πάνω στο σχήμα τις τιμές των σημάτων που απαιτούνται για την προσπέλαση και την εκτέλεση της εντολής **Store r1, d(r2)** η οποία εκτελείται σε 4 κύκλους ρολογιού. Αριστερά βρίσκεται η τιμή που αντιστοιχεί στον 1ο κύκλο ρολογιού. Για πρόσθεση να χρησιμοποιηθούν οι τιμές **AΛ0=0, AΛ1=1**. Ο κωδικός λειτουργίας της εντολής LOAD είναι 01111. Δεν χρειάζεται αιτιολόγηση.

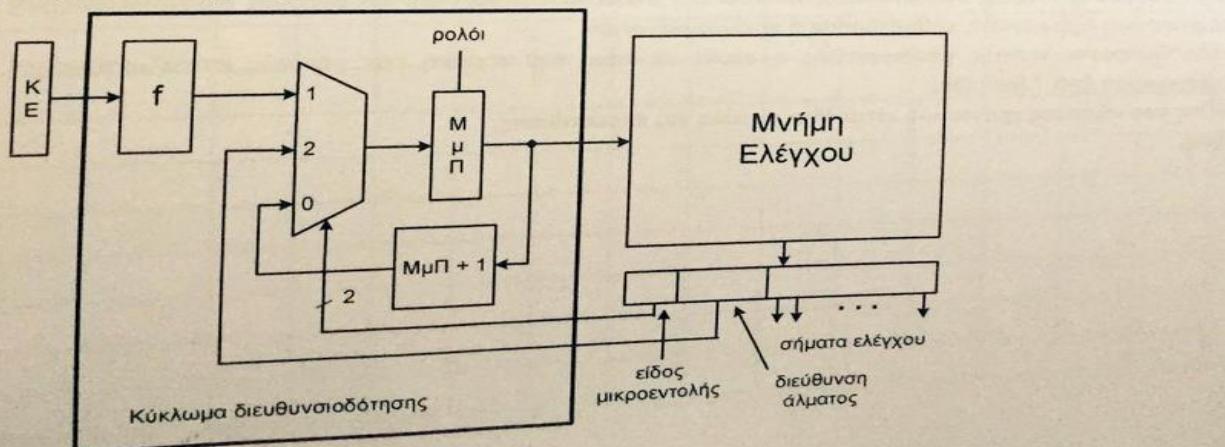


Λύση



Θέμα Ιουνίου 2019

β. Η μονάδα ελέγχου έχει υλοποιηθεί με την τεχνική των μικροτροφορματισμού (βλέπε επόμενο σχήμα).



Να γράψετε το μικροπρόγραμμα προσκόμισης εντολής και το μικροπρόγραμμα εκτέλεσης της εντολής LOAD (να τοποθετήσετε τα μικροπρογράμματα στον επόμενο πίνακα) και την τιμή εισόδου και εξόδου της μονάδας f. Θεωρήστε ότι οι διευθύνσεις της μνήμης ελέγχου είναι των 8 δυαδικών ψηφίων.

Λύση

διεύθυνση μνήμης ελέγχου	είδος μικρο-εντολής άλματος	διεύθυνση άλματος	περιεχόμενα μνήμης ελέγχου												
			γΜΠ	δΜΕ	Br	γΚΕ	πΕ&Δ	γΚ	πΑ	πΒ0	πΒ1	ΑΛ0	ΑΛ1	δΜΔ	γΜΔ
00000000	10	XX.....XX	1	1	0	1	0	0	0	1	0	1	0	0	0
00000001	01	XX....XX	0	0	0	0	0	0	0	0	1	1	0	0	0
00001100	10	XX....XX	0	0	0	0	0	0	0	0	1	1	0	0	0
00001101	01	0.....00	0	0	0	0	0	0	0	0	0	0	0	1	0
									1	0	0	0	0	0	1

Πίνακας 3.17
Περιεχόμενα Μνήμης Ελέγχου

	Διεύθυνση μνήμης ελέγχου	Είδος μικρο-εντολής άλματος	Διεύθυνση άλματος	γΜΠ	δΜΕ	Br	γΚΕ	πΕ&Δ	γΚ	πΑ	πΒ0	πΒ1	ΑΛ0	ΑΛ1	δΜΔ	γΜΔ
{ μικροπρόγραμμα προσκόμισης εντολών	0000000000	10	X ... XX	1	1	0	1	0	0	0	1	0	1	0	0	0
	0000000001	00	X ... XX	0	0	0	0	0	0	0	1	1	1	0	0	0
{ μικροπρόγραμμα εντολής LOAD	0000001100	10	X ... XX	0	0	0	0	0	0	0	0	0	0	0	1	0
	0000001101	01	0000000000	0	0	0	0	0	1	0	0	0	0	0	1	0
μικροπρόγραμμα εντολής ADD	0000001111	10	X ... XX	0	0	0	0	0	0	1	0	0	0	1	0	0
	0000010000	01	0000000000	0	0	0	0	1	1	0	0	0	0	0	0	0
μικροπρόγραμμα εντολής SUB	0011100000	10	X ... XX	0	0	0	0	0	0	1	0	0	0	1	0	0
	0011100001	01	0000000000	0	0	0	0	1	1	0	0	0	0	0	0	0
μικροπρόγραμμα εντολής AND	0011100011	10	X ... XX	0	0	0	0	0	0	1	0	0	0	1	1	0
	0011100100	01	0000000000	0	0	0	0	1	1	0	0	0	0	0	0	0
μικροπρόγραμμα εντολής STORE	0011101100	01	-0000000000	0	0	0	0	0	0	0	0	0	0	0	0	1
μικροπρόγραμμα εντολής BRE	0011111100	01	0000000000	0	0	1	0	0	0	1	0	0	0	1	0	0

Κεφάλαιο 4 – Μηχανισμός μερικώς επικαλυπτόμενων λειτουργιών και εξαρτήσεις

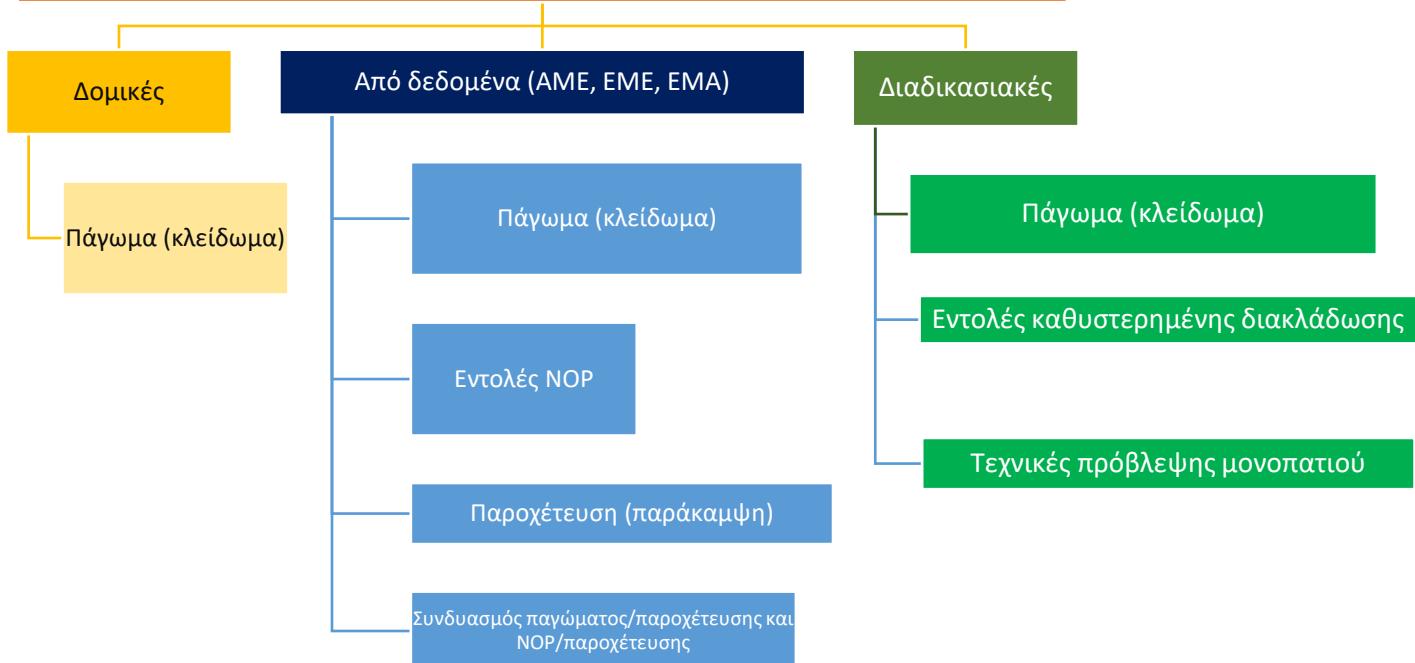
4.1 Θεωρία εξαρτήσεων

1. Αν στο μηχανισμό ΜΕΛ υπάρχει **κοινή (ενοποιημένη) κρυφή μνήμη εντολών και δεδομένων**, τότε θα υπάρχουν **δομικές εξαρτήσεις**, οι οποίες πάντα θα είναι ανάμεσα στο βήμα **ΠΜ της εντολής LOAD ή STORE** και το βήμα **ΠΕ** οποιασδήποτε άλλης εντολής. Αυτό συμβαίνει διότι τα συγκεκριμένα βήματα των συγκεκριμένων εντολών χρησιμοποιούν την κρυφή μνήμη και επειδή αυτή είναι μόνο **μία**, δημιουργείται εξάρτηση, η οποία και θα πρέπει να επιλυθεί.
2. Μπορεί μέσα σε ένα δοθέντα κώδικα του μηχανισμού ΜΕΛ να υπάρχουν (προκύπτουν) περισσότερες από **μια** δομικές εξαρτήσεις, οι οποίες θα έχουν πάντα ως **αφετηρία** μια εντολή **LOAD ή STORE** και τερματισμό οποιαδήποτε άλλη εντολή.
3. Η **επίλυση** των δομικών εξαρτήσεων γίνεται πάντα με το **πάγωμα (κλείδωμα)** του μηχανισμού ΜΕΛ για μία χρονική περίοδο (κύκλο ρολογιού) και αυτό λαμβάνει χώρα **πριν** από το βήμα **ΠΕ** της δεύτερης εντολής που εμπλέκεται στη δομική εξάρτηση, με την εισαγωγή ενός παγώματος **X**.
4. Η δημιουργία εξαρτήσεων από δεδομένα **τύπου AME** προκύπτει όταν μια εντολή χρησιμοποιεί το αποτέλεσμα μιας προηγούμενης.
5. Οι εξαρτήσεις από δεδομένα **τύπου AME** προκύπτουν είτε μεταξύ γειτονικών εντολών είτε μεταξύ μη - γειτονικών εντολών, που διαφέρουν μεταξύ τους το πολύ κατά μια εντολή.
6. Αν χρησιμοποιηθεί το **πάγωμα (κλείδωμα)** για την επίλυση των εξαρτήσεων από δεδομένα **τύπου AME** τότε τα δύο παγώματα συμβολίζονται με **ΑΕ'** και **X**, ενώ το ένα πάγωμα συμβολίζεται με **ΑΕ'**.
7. Σε περίπτωση που υποστηρίζεται η τεχνική της **παροχέτευσης** ή **παράκαμψης** για την επίλυση των εξαρτήσεων από δεδομένα **τύπου AME**, τότε **ξεκινάμε** από το βήμα **ΠΜ** (αν πρόκειται για εντολή **LOAD ή STORE**) ή από το βήμα **ΕΠ** (αν πρόκειται για οποιαδήποτε άλλη εντολή) και **καταλήγουμε πάντα** στο βήμα **ΕΠ** της εντολής με την οποία υπάρχει η εξάρτηση, η οποία θα πρέπει να έχει το βήμα **ΕΠ** μια τουλάχιστον θέση πιο δεξιά σε σχέση με τη θέση από την οποία ξεκινάμε, για να επιλυθεί η εξάρτηση. Αν απαιτηθεί, μπορεί να χρειαστεί να παρεμβάλουμε μια εντολή **NOP** ή να κάνουμε ένα πάγωμα.
8. Σε περίπτωση που υποστηρίζεται η τεχνική της **παροχέτευσης** ή **παράκαμψης** για την επίλυση των εξαρτήσεων από δεδομένα **τύπου AME** τότε οι εξαρτήσεις – αν υπάρχουν – θα υπάρχουν **μόνο** μεταξύ γειτονικών εντολών. Εξαρτήσεις μεταξύ **μη γειτονικών** εντολών επιλύονται **απευθείας** (αυτόματα) στην περίπτωση αυτή.
9. Οταν από την εκφώνηση της άσκησης δίνεται ότι υλοποιείται η τεχνική της **παροχέτευσης** εκ' των προτέρων, τότε ψάχνουμε για εξαρτήσεις τύπου AME μεταξύ των βημάτων **ΠΜ** και **ΕΠ** ή **ΕΠ** και **ΕΠ**. Σε διαφορετική περίπτωση, ψάχνουμε για εξαρτήσεις τύπου AME μεταξύ των βημάτων **ΑΑ** και **ΑΕ**.
10. Οι **διαδικασιακές εξαρτήσεις** προκύπτουν όταν μέσα στον κώδικα ενός ΜΕΛ εμφανιστεί μια εντολή διακλάδωσης, όπως π.χ. BRE, BRN, BRNE, BRC κ.λ.π.
11. Οι **διαδικασιακές εξαρτήσεις** επιλύονται είτε με πάγωμα, είτε με εντολές καθυστερημένης διακλάδωσης με καθυστέρηση (συνήθως) δύο χρονικών μονάδων (πιο συνήθης τρόπος επίλυσης), είτε με τεχνική πρόβλεψης μονοπατιού. Η τελευταία τεχνική προσθέτει δύο κ.ρ. μετά από μια συνθήκη που εξετάζεται και είναι ψευδής.
12. Σε περίπτωση **πολλαπλών εξαρτήσεων μέσα σε ένα δοθέντα κώδικα μηχανισμού ΜΕΛ**, αυτές επιλύονται με τη σειρά από **πάνω προς τα κάτω** και σε περίπτωση που μία εντολή περιέχει περισσότερες από μια εξαρτήσεις, αυτές επιλύονται με προτεραιότητα προς την **πλησιέστερη**.
13. Το **πάγωμα (κλείδωμα)** είναι ο **κοινός** τρόπος επίλυσης όλων **των ειδών εξαρτήσεων**.
14. Μπορεί στην **ίδια εντολή** να εμφανιστούν **διαφορετικά παγώματα**, τόσο για την επίλυση δομικών εξαρτήσεων όσο και για την επίλυση εξαρτήσεων από δεδομένα. Αυτό σημαίνει ότι μπορεί στην ίδια εντολή να βάλουμε στην αρχή ένα πάγωμα (**X**) για την επίλυση δομικής εξάρτησης και στη συνέχεια να βάλουμε και άλλα παγώματα με **ΑΕ'** και **X** για να λύσουμε εξαρτήσεις από δεδομένα τύπου AME.

15. Είναι δυνατόν στην επίλυση των εξαρτήσεων, να προκύψουν δομικές εξαρτήσεις μεταξύ εντολών που αρχικά δεν υπήρχαν. Αυτό συμβαίνει λόγω μετάθεσης των εντολών προς τα δεξιά, όταν εισάγουμε NOP.
16. Σε περίπτωση αναδιάταξης **εντολών** (δηλαδή αλλαγή της σειράς εκτέλεσής τους) με σκοπό τη μείωση του χρησιμοποιούμενου αριθμού εντολών NOP, τότε στην **αρχή** του κώδικα τοποθετούνται όλες οι εντολές LOAD και προς το **τέλος** οι εντολές STORE.

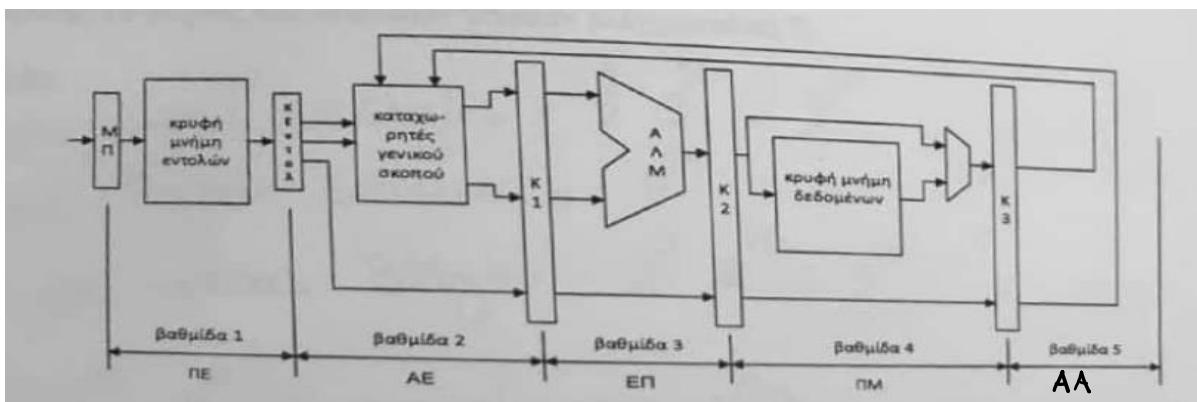
17. Τα είδη εξαρτήσεων περιγράφονται στη συνέχεια

Κατηγορίες (Είδη) εξαρτήσεων και τρόποι επίλυσης τους



18. Όταν σε μια εκφώνηση διατυπώνεται ότι χρησιμοποιείται η τεχνική της παροχέτευσης από την αρχή, τότε **θα πρέπει να τη λάβουμε υπόψη μας στον υπολογισμό των εξαρτήσεων** και να μη σημειώσουμε καθόλου εκείνες τις εξαρτήσεις που επιλύνονται αυτόματα.

19. Στο ΜΕΛ που ακολουθεί, δεν είναι δυνατό να προκύψουν δομικές εξαρτήσεις λόγω χρήσης των δύο κρυφών μνημών.



Επιπλέον δεν μπορούν να προκύψουν εξαρτήσεις από δεδομένα τύπου **EMA** (Εγγραφή-Μετά-από-Ανάγνωση) **και EME** (Εγγραφή-Μετά-από-Εγγραφή). Η εξάρτηση EMA προκύπτει όταν η εντολή B γράφει σε έναν καταχωρητή της KME ή σε μία θέση μνήμης, μετά από την ανάγνωση από τον καταχωρητή αυτό ή από αυτή τη θέση μνήμης από την εντολή A. Αυτό το είδος εξάρτησης δεν είναι δυνατόν να προκύψει στον ΜΕΛ που θεωρούμε διότι η ανάγνωση δεδομένων γίνεται σε κάποιον από τους πρώτους κύκλους εκτέλεσης της εντολής, ενώ η εγγραφή στους τελευταίους. Η εξάρτηση EME εμφανίζεται μόνο όταν στο μηχανισμό μερικώς επικαλυπτόμενων λειτουργιών εμφανίζονται περισσότερες από μια βαθμίδας στις οποίες μπορεί να γίνει εγγραφή στον ίδιο προορισμό, δηλαδή σε όλες τις περιπτώσεις στη μνήμη. Στο μηχανισμό μερικώς επικαλυπτόμενων λειτουργιών που θεωρούμε δεν μπορεί να προκύψει τέτοιου τύπου εξάρτηση διότι γίνεται εγγραφή σε

καταχωρητή ή σε μνήμη μόνο σε μια βαθμίδα του μηχανισμού. Εξαρτήσεις από δεδομένα τύπου AME μπορούν να εμφανιστούν. **Διαδικασιακές** εξαρτήσεις μπορούν να προκύψουν, διότι αυτές εξαρτώνται από την ύπαρξη αποκλειστικά και μόνο εντολών διακλάδωσης.

Θέμα για το μέγιστο ρυθμό ολοκλήρωσης εντολών σε ΜΕΛ

Να αναφέρετε τους λόγους για τους οποίους σε ένα επεξεργαστή μερικώς επικαλυπτόμενων λειτουργιών **δεν** μπορούμε να πετύχουμε το μέγιστο ρυθμό ολοκλήρωσης εντολών, δηλαδή **δεν** πετυχαίνουμε τη μέγιστη τιμή του EPO που είναι το N;

Λύση

- Δεν είναι δυνατόν να έχουν όλες οι βαθμίδες την ίδια καθυστέρηση
- Δεν χρησιμοποιούν όλες οι εντολές όλες τις βαθμίδες του μηχανισμού
- Οι καταχωρητές που υπάρχουν μεταξύ των βαθμίδων προσθέτουν μια καθυστέρηση
- Υπάρχουν εξαρτήσεις, οι οποίες μειώνουν την απόδοση των επεξεργαστών μερικώς επικαλυπτόμενων λειτουργιών
- Στις αρχικές χρονικές περιόδους δεν ολοκληρώνεται καμία διεργασία

Θέμα αναφορικά με τα είδη εξαρτήσεων σε ένα ΜΕΛ

Να αναφέρετε τα **είδη** των εξαρτήσεων σε ένα μηχανισμό μερικώς επικαλυπτόμενων λειτουργιών.

Λύση

- a) **Δομικές εξαρτήσεις:** εμφανίζονται αν υπάρχουν τουλάχιστον 2 εντολές που όταν βρεθούν για εκτέλεση στο μηχανισμό μερικώς επικαλυπτόμενων λειτουργιών κάποια βήματα τους χρειάζονται την **ίδια** βαθμίδα του μηχανισμού ταυτόχρονα. Συγκεκριμένα εμφανίζεται σε επεξεργαστές που διαθέτουν **κοινή** μνήμη εντολών και δεδομένων με δυνατότητα σε κάθε χρονική περίοδο να μπορεί να διαβαστεί μόνο μια θέση μνήμης.
- b) **Εξαρτήσεις από δεδομένα:** εμφανίζονται όταν υπάρχει εξάρτηση μεταξύ εντολών και αυτές είναι αρκετά κοντά ώστε η επικάλυψη που συμβαίνει, να οδηγεί σε αλλαγή της σειράς προσπέλασης κάποιων δεδομένων. Συγκεκριμένα όταν ο μηχανισμός μερικώς επικαλυπτόμενων λειτουργιών αλλάζει τη σειρά με την οποία εκτελούνται προσπελάσεις ανάγνωσης και εγγραφής δεδομένων. Υπάρχουν τρία είδη: AME, EME και EMA.
- c) **Διαδικασιακές εξαρτήσεις:** προκύπτουν λόγω της αλλαγής της ροής του προγράμματος που οφείλεται στην εκτέλεση εντολών διακλάδωσης.

Θέμα αναφορικά με τη χρήση εντολών NOP

Με τη χρησιμοποίηση **εντολών NOP** σε έναν επεξεργαστή μερικώς επικαλυπτόμενων εντολών μπορούμε να λύσουμε:

- a) Δομικές εξαρτήσεις που προκύπτουν λόγω της χρήσης κοινής κρυφής μνήμης εντολών και δεδομένων
- b) Εξαρτήσεις δεδομένων τύπου AME
- c) Διαδικασιακές εξαρτήσεις
- d) Καμία εκ των ανωτέρων

Λύση

Εξαρτήσεις δεδομένων τύπου AME (b) και διαδικασιακές εξαρτήσεις (c).

Θέμα αναφορικά με τους τρόπους επίλυσης εξαρτήσεων

Με ποιους τρόπους επιλύονται:

- a) Οι δομικές εξαρτήσεις
- b) Οι εξαρτήσεις από δεδομένα
- c) Οι διαδικασιακές εξαρτήσεις

Λύση

- a) Οι δομικές εξαρτήσεις επιλύονται μόνο με την τεχνική του παγώματος.
- b) Οι εξαρτήσεις από δεδομένα μπορούν επιλυθούν ως εξής.
 - (i) Με χρήση εντολών NOP. Κατά την εκτέλεση μιας εντολής NOP δεν γίνετε καμία λειτουργία.
 - (ii) Με την τεχνική του παγώματος.
 - (iii) Με την τεχνική της παροχέτευσης, χρησιμοποιείται για να γίνει καλύτερη αξιοποίηση του μηχανισμού μερικώς επικαλυπτόμενων λειτουργιών. Για την υλοποίηση της απαιτούνται κυκλώματα που έχουν ως σκοπό να αναγνωρίσουν πότε και πως πρέπει να γίνει η παροχέτευση και πολυπλέκτες που την υλοποιούν.
- (iv) Με συνδυασμό παροχέτευσης –NOP και παροχέτευσης – παγώματος.

- c) Ο απλούστερος τρόπος επίλυσης των διαδικασιών εξαρτήσεων είναι να παγώσουμε το μηχανισμό μόλις ανιχνεύσουμε ότι η εντολή που αποτελείται είναι εντολή διακλάδωσης. Εναλλακτικά, χρησιμοποιούμε εντολές καθυστερημένης διακλάδωσης ή τεχνικές πρόβλεψης μονοπατιού.

Θέμα αναφορικά με τεχνική παροχέτευσης

Να αναφέρετε το πρόβλημα το οποίο θέλουμε να λύσουμε με την τεχνική παροχέτευσης και πως το λύνουμε.

Λύση

Οι εξαρτήσεις από δεδομένα μπορούν να επιλυθούν με τη χρήση της τεχνικής της παροχέτευσης.

Πλεονέκτημα: Η τεχνική αυτή κάνει καλύτερη αξιοποίηση του μηχανισμού επικαλυπτόμενων λειτουργιών.

Μειονέκτημα: Σε μια τέτοια περίπτωση απαιτούνται κυκλώματα που έχουν ως σκοπό να αναγνωρίσουν πότε και πως πρέπει να γίνει η παροχέτευση και πολυπλέκτες που υλοποιούν ην παροχέτευση, άρα αυξάνεται το κόστος.

Στην τεχνική της παροχέτευσης οι εντολές αριθμητικών και λογικών πράξεων (π.χ. ADD, SUB, AND...) απαιτούν τα δεδομένα και τα έχουν διαθέσιμα στην βαθμίδα ΕΠ. Οι εντολές ανάκτησης δεδομένων και αποθήκευσης αποτελεσμάτων (LOAD, STORE) απαιτούν τα δεδομένα και τα έχουν διαθέσιμα στην βαθμίδα ΠΜ. Στην τεχνική αυτή έχει σημασία από που ξεκινάμε. Αν ξεκινάμε από εντολή ADD, SUB, MUL κ.λ.π. τότε οι εξαρτήσεις επιλύνονται απευθείας, διότι πηγαίνουμε από το ΕΠ στο ΕΠ.

εντολή	λ	λ+1	λ+2	λ+3	λ+4	λ+5
ADD r1, r2, r3	ΠΕ	ΑΕ	ΕΠ	ΠΜ	ΑΑ	
SUB r5, r3, r6	ΠΕ	ΑΕ	ΕΠ	ΠΜ	ΑΑ	

Αν όμως ξεκινάμε από εντολή LOAD, STORE, τότε οι εξαρτήσεις δεν επιλύνονται απευθείας διότι πηγαίνουμε από το ΠΜ στο ΕΠ και θα πρέπει πάντα το ΕΠ στο οποίο καταλήγουμε να είναι πιο δεξιά σε σχέση με τη θέση που ξεκινάμε.

εντολή	λ	λ+1	λ+2	λ+3	λ+4	λ+5	λ+6	λ+7
LOAD r1, (r4)	ΠΕ	ΑΕ	ΕΠ	ΠΜ	ΑΑ			
ADD r3, r1, r2	ΠΕ	ΑΕ'	ΑΕ	ΕΠ	ΠΜ	ΑΑ		
AND r5, r8, r9	ΠΕ	ΑΕ	ΕΠ	ΠΜ	ΑΑ			

Οι εξαρτήσεις μεταξύ μη – γειτονικών εντολών επιλύνονται απευθείας με την τεχνική της παροχέτευσης.

Θέμα Θεωρίας αναφορικά με τα μειονεκτήματα - πλεονεκτήματα του ΜΕΛ

Πλεονεκτήματα – Μειονεκτήματα ΜΕΛ. Να εξηγήσετε που βασίζεται η επιτυχία του ΜΕΛ.

Λύση

Μειονέκτημα ΜΕΛ: Στην τεχνική μερικώς επικαλυπτόμενων λειτουργιών, ο συνολικός χρόνος εκτέλεσης μιας διαδικασίας δεν είναι μικρότερος, αλλά μπορεί να είναι και μεγαλύτερος. Αυτό συμβαίνει διότι στον υπολογισμό του ΠΣΧ λαμβάνεται υπόψη η καθυστέρηση της πιο αργής βαθμίδας.

Πλεονέκτημα ΜΕΛ: Εντούτοις ο ρυθμός ολοκλήρωσης των διαδικασιών στην περίπτωση που δεν χρησιμοποιείται η τεχνική των μερικώς επικαλυπτόμενων λειτουργιών είναι μια διαδικασία ανά N χρονικές περιόδους (κύκλους) ενώ στην περίπτωση που χρησιμοποιείται, είναι μια διαδικασία ανά κύκλο. Επομένως έχουμε μια επιτάχυνση του ρυθμού εκτέλεσης διαδικασιών κατά N. Πρέπει να σημειώσουμε ότι αυτή είναι η μέγιστη επιτάχυνση του ρυθμού εκτέλεσης λειτουργιών που για διάφορους λόγους είναι πολύ δύσκολο να επιτευχθεί.

Πλεονέκτημα ΜΕΛ: Η κύρια συνεισφορά της τεχνικής των μερικών επικαλυπτόμενων λειτουργιών είναι ότι προσφέρει ένα τρόπο να ξεκινήσουμε μια νέα διαδικασία πριν ολοκληρωθεί μια παλιότερη. Συνεπώς, ο αριθμός ολοκλήρωσης δεν είναι συνάρτηση του συνολικού χρόνου επεξεργασίας αλλά μάλλον του πόσο γρήγορα μπορεί να ξεκινήσει μια νέα διαδικασία.

Θέμα αναφορικά με επιτάχυνση του ρυθμού ολοκλήρωσης εντολών

Να εξηγήσετε γιατί σε ένα μηχανισμό μερικώς επικαλυπτόμενων λειτουργιών με κ βαθμίδες δεν μπορούμε να πετύχουμε επιτάχυνση του ρυθμού εκτέλεσης εντολών ίσο με k σε σχέση με τον αντίστοιχο επεξεργαστή που δεν υλοποιεί την τεχνική των μερικών επικαλυπτόμενων λειτουργιών.

Λύση

Η μέγιστη τιμή της επιτάχυνσης του ρυθμού ολοκλήρωσης εντολών δεν ισούται με το πλήθος των βαθμίδων, δηλαδή k.

1. Η χρησιμοποίηση των καταχωρητών μεταξύ των βαθμίδων προσθέτουν καθυστέρηση.

2. Δεν μπορούμε να διαμερίσουμε την εντολή σε k βήματα με τέτοιο τρόπο ώστε κάθε βήμα για την εκτέλεσή του να απαιτεί τον ίδιο χρόνο, επομένως δεν έχουν όλες οι βαθμίδες την ίδια καθυστέρηση.

3. Όλες οι εντολές **δεν** χρησιμοποιούν και τις k βαθμίδες.
4. Υπάρχουν οι **εξαρτήσεις** που μειώνουν την απόδοση των επεξεργαστών μερικώς επικαλυπτόμενων λειτουργιών (Δομικές εξαρτήσεις, Εξαρτήσεις δεδομένων, Διαδικασιακές εξαρτήσεις).
5. Στις αρχικές χρονικές περιόδους δεν ολοκληρώνεται καμία διεργασία.

Ο χρόνος εκτέλεσης μίας εντολής ισούται με το γινόμενο της περιόδου του σήματος χρονισμού και του πλήθους

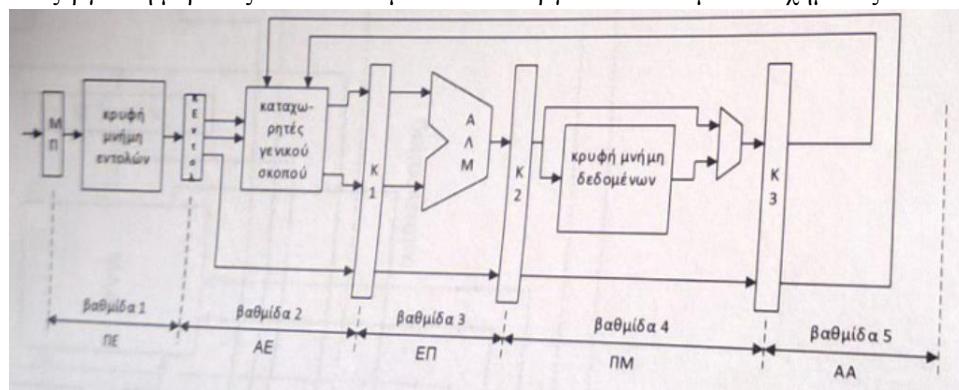
Θέμα Σεπτεμβρίου 2019 για ΜΕΛ

Στα παρακάτω να σημειώσετε τα Σ/Λ.

1. Η τεχνική των μερικώς επικαλυπτόμενων λειτουργιών (pipeline) αυξάνει το ρυθμό ολοκλήρωσης εντολών σε σχέση με ένα επεξεργαστή ο οποίος δεν υλοποιεί την τεχνική αυτή. $\rightarrow \Sigma$
2. Σε ένα επεξεργαστή μερικώς επικαλυπτόμενων εντολών με **κ** βαθμίδες εάν δεν υπήρχαν εξαρτήσεις μεταξύ των εντολών η επιτάχυνση του ρυθμού ολοκλήρωσης εντολών σε σχέση με αντίστοιχο επεξεργαστή που δεν υλοποιεί την τεχνική των μερικώς επικαλυπτόμενων εντολών θα ήταν **ίση** με **κ**. $\rightarrow \Lambda$
3. Σε ένα επεξεργαστή μερικώς επικαλυπτόμενων εντολών με **κ βαθμίδες** ακόμη και εάν δεν υπήρχαν εξαρτήσεις μεταξύ των εντολών η επιτάχυνση του ρυθμού ολοκλήρωσης εντολών σε σχέση με αντίστοιχο επεξεργαστή που δεν υλοποιεί την τεχνική των μερικώς επικαλυπτόμενων εντολών **θα ήταν μικρότερη** του **κ**. $\rightarrow \Sigma$
4. Ο χρόνος εκτέλεσης (latency) μίας εντολής σε επεξεργαστή που υλοποιεί την τεχνική των μερικώς επικαλυπτόμενων λειτουργιών (pipeline) είναι ίδιος με τον χρόνο εκτέλεσης σε ένα επεξεργαστή ο οποίος δεν υλοποιεί την τεχνική αυτή. $\rightarrow \Lambda$
5. Το κόστος υλοποίησης ενός επεξεργαστή μερικώς επικαλυπτόμενων λειτουργιών (pipeline) είναι μικρότερο από αυτό ενός επεξεργαστή ο οποίος δεν υλοποιεί την τεχνική αυτή. $\rightarrow \Lambda$
6. Η τεχνική των μερικώς επικαλυπτόμενων λειτουργιών (pipeline) μειώνει το χρόνο εκτέλεσης (latency) μίας εντολής σε σχέση με ένα επεξεργαστή ο οποίος δεν υλοποιεί την τεχνική αυτή. $\rightarrow \Lambda$
7. Η τεχνική των μερικώς επικαλυπτόμενων λειτουργιών (pipeline) αυξάνει το χρόνο εκτέλεσης (latency) μίας εντολής σε σχέση με ένα επεξεργαστή ο οποίος δεν υλοποιεί την τεχνική αυτή. $\rightarrow \Sigma$
8. Το κόστος υλοποίησης ενός επεξεργαστή μερικώς επικαλυπτόμενων λειτουργιών (pipeline) είναι μεγαλύτερο από αυτό ενός επεξεργαστή ο οποίος δεν υλοποιεί την τεχνική αυτή. $\rightarrow \Sigma$

Θέμα Ιοννίου 2014 θεωρίας

- Να αναφέρετε όλα τα είδη εξαρτήσεων που μπορούν να εμφανιστούν σε ένα επεξεργαστή μερικώς επικαλυπτόμενων λειτουργιών.
- Να αιτιολογήσετε ποια από τα είδη εξαρτήσεων που αναφέρατε στο ερώτημα α μπορούν να προκύψουν και ποια όχι στον επεξεργαστή μερικώς επικαλυπτόμενων λειτουργιών του επόμενου σχήματος



Λύση

α. - Δομικές εξαρτήσεις: εμφανίζονται αν υπάρχουν τουλάχιστον 2 εντολές που όταν βρεθούν για εκτέλεση στο μηχανισμό μερικώς επικαλυπτόμενων λειτουργιών κάποια βήματα τους χρειάζονται την **ίδια** βαθμίδα του μηχανισμού ταυτόχρονα. Συγκεκριμένα εμφανίζεται σε επεξεργαστές που διαθέτουν **κοινή** μνήμη εντολών και δεδομένων με δυνατότητα σε κάθε χρονική περίοδο να μπορεί να διαβαστεί μόνο μια θέση μνήμης.

- Εξαρτήσεις από δεδομένα: εμφανίζονται όταν υπάρχει εξάρτηση μεταξύ εντολών και αυτές είναι αρκετά κοντά ώστε η επικάλυψη που συμβαίνει, να οδηγεί σε αλλαγή της σειράς προσπέλασης κάποιων δεδομένων. Συγκεκριμένα όταν ο μηχανισμός μερικώς επικαλυπτόμενων λειτουργιών αλλάζει τη σειρά με την οποία εκτελούνται προσπελάσεις ανάγνωσης και εγγραφής δεδομένων. Υπάρχουν τρία είδη: AME, EME και EMA.

- Διαδικασιακές εξαρτήσεις: προκύπτουν λόγω της αλλαγής της ροής των προγράμματος που οφείλεται στην εκτέλεση εντολών διακλάδωσης.

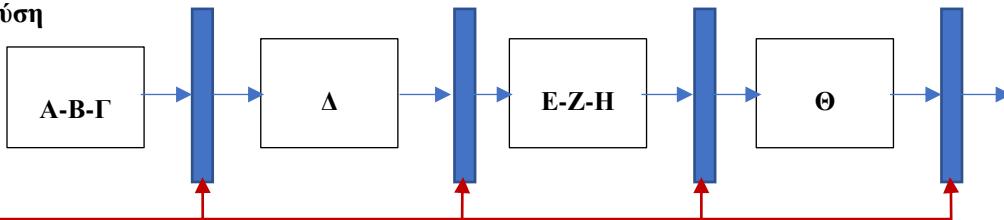
β. Από τα είδη εξαρτήσεων που αναφέρθηκαν στο ερώτημα α, **δεν** μπορούν να προκύψουν στο μηχανισμό μερικώς επικαλυπτόμενων λειτουργιών του σχήματος, οι **δομικές εξαρτήσεις** λόγω ύπαρξης ξεχωριστής κρυφής μνήμης εντολών και δεδομένων. Επίσης, **δεν** μπορούν να προκύψουν εξαρτήσεις **από δεδομένα τύπου EME**, διότι στο συγκεκριμένο μηχανισμό γίνεται εγγραφή σε καταχωρητή ή μνήμη μόνο σε μια βαθμίδα του. Τέλος, **δεν** μπορούν να προκύψουν εξαρτήσεις **από δεδομένα τύπου EMA**, διότι στο συγκεκριμένο μηχανισμό η ανάγνωση δεδομένων γίνεται σε κάποιον από τους πρώτους κύκλους εκτέλεσης μιας εντολής, ενώ η εγγραφή στους τελευταίους.

Θέμα Ιουνίου 2019 για ΜΕΛ

Για την εκτέλεση μιας διαδικασίας απαιτείται η εκτέλεση της ακολουθίας βημάτων $\alpha, \beta, \gamma, \delta, \varepsilon, \zeta, \eta$ και θ . Τα βήματα αυτά μπορούν να εκτελεστούν από τις αντίστοιχες μονάδες $A, B, \Gamma, \Delta, E, Z, H$ και Θ με καθυστέρηση 8 ns, 3 ns, 7 ns, 19 ns, 8 ns, 3 ns, 8 ns και 9 ns αντίστοιχα. Έχετε στη διάθεσή σας καταχωρητές που η καθυστέρηση τους είναι 1 nsec.

α) Να σχεδιάστε το μηχανισμό μερικώς επικαλυπτόμενων λειτουργιών (pipelining) που επιτυγχάνει το **μέγιστο ρυθμό ολοκλήρωσης διαδικασιών** και γι' αυτό το ρυθμό, τον **ελάχιστο χρόνο εκτέλεσης μιας διαδικασίας (latency)**. Δεν χρειάζεται αιτιολόγηση.

Λύση



$$\text{ΠΣΧ} = 19 \text{ nsec} + 1 \text{ nsec} = 20 \text{ nsec}$$

β) Να υπολογίσετε το ρυθμό ολοκλήρωσης διαδικασιών και το χρόνο που απαιτείται για την εκτέλεση μιας διαδικασίας.

Λύση: $P.O = 1$ διεργασία κάθε $\text{ΠΣΧ} = 1$ διεργασία κάθε 20 nsec και $X.O = \text{πλήθος βαθμίδων} * \text{ΠΣΧ} = 4 * 20 \text{ nsec} = 80 \text{ nsec}$.

γ) **Ποια είναι η επιτάχυνση του ρυθμού ολοκλήρωσης διαδικασιών που επιτυγχάνει ο μηχανισμός που σχεδιάσατε** (θεωρήστε ότι δεν υπάρχουν καθυστέρησεις λόγω εξαρτήσεων) σε σχέση με μια υλοποίηση που χρησιμοποιεί τις ίδιες μονάδες αλλά δεν υλοποιεί την τεχνική των μερικώς επικαλυπτόμενων λειτουργιών; Αγνοήστε τις αρχικές συνθήκες.

Λύση: $EPO = \frac{K * \text{χρόνος εκτέλεσης χωρίς pipeline}}{(N - 1 + K) * \frac{\text{χρόνος εκτέλεσης χωρίς pipeline}}{N}} = \frac{K * \text{χρόνος εκτέλεσης χωρίς pipeline}}{(N - 1 + K) * \text{ΠΣΧ}}$ Επειδή δεν έχουμε καθυστέρηση λόγω εξαρ-

τήσεων, μπορούμε να απλοποιήσουμε το χρόνο από αριθμητή και παρανομαστή και στο τέλος προκύπτει: $EPO = \frac{K * N}{N - 1 + K} = \frac{K * 4}{3 + K}$. Από την εκφώνηση δεν δίνεται ο αριθμός K των διεργασιών. Η μέγιστη θεωρητικά τιμή του EPO είναι 4.

Θέμα Ιουνίου 2019 για ΜΕΛ

Ο σχεδιαστής ενός επεξεργαστή ειδικού σκοπού με εξομοιώσεις διαπίστωσε ότι στην εφαρμογή, για την οποία σχεδιάζεται ο επεξεργαστής, **οι εντολές πολλαπλασιασμού κινητής υποδιαστολής εμφανίζονται μέσα στο πρόγραμμα συνήθως σε ομάδες των 16**, ενώ οι εντολές διαίρεσης κινητής υποδιαστολής δεν εμφανίζονται συχνά και είναι διάσπαρτες μέσα στο πρόγραμμα. Ποια μονάδα πολλαπλασιασμού κινητής υποδιαστολής και ποια μονάδα διαίρεσης κινητής υποδιαστολής από τις ακόλουθες θα επιλέξει ο σχεδιαστής ώστε να μεγιστοποιήσει την απόδοση του επεξεργαστή που σχεδιάζει; Στην περίπτωση στην οποία η απόδοση δεν επηρεάζεται ο σχεδιαστής θα πρέπει να επιλέξει τη μονάδα με το μικρότερο κόστος υλοποίησης. Να κυκλώσετε τις επιλογές σας και να αιτιολογήσετε την απάντησή σας το πολύ σε επτά γραμμές.

α. Μονάδα πολλαπλασιασμού κινητής υποδιαστολής η οποία δεν υλοποιεί την τεχνική των μερικώς επικαλυπτόμενων λειτουργιών και έχει μέγιστη καθυστέρηση 5 κύκλων ρολογιού.

β. Μονάδα πολλαπλασιασμού κινητής υποδιαστολής η οποία υλοποιεί την τεχνική των μερικώς επικαλυπτόμενων λειτουργιών και έχει 6 βαθμίδες.

γ. Μονάδα διαίρεσης κινητής υποδιαστολής η οποία δεν υλοποιεί την τεχνική των μερικώς επικαλυπτόμενων λειτουργιών και έχει μέγιστη καθυστέρηση 6 κύκλων ρολογιού.

δ. Μονάδα διαίρεσης κινητής υποδιαστολής η οποία υλοποιεί την τεχνική των μερικώς επικαλυπτόμενων λειτουργιών και έχει 7 βαθμίδες.

Η περίοδος του σήματος χρονισμού είναι ίδια σε όλες τις περιπτώσεις

Λύση

Οι σωστές απαντήσεις είναι η (b) και η (c). Η αιτιολόγηση είναι ότι στον πολλαπλασιασμό, όπου οι εντολές εμφανίζονται συχνά (εμφανίζονται σε ομάδες των «16») απαιτείται αύξηση του ρυθμού ολοκλήρωσης διεργασιών πολλαπλασιασμού και για το λόγο αυτό επιλέγουμε **πολλαπλασιαστή που υποστηρίζει την τεχνική των ΜΕΛ** (ως γνωστό ο ρυθμός ολοκλήρωσης είναι 1 εντολή

ανά κύκλο ρολογιού), ενώ για τη διαίρεση, από τη στιγμή που οι εντολές διαίρεσης δεν είναι συχνά εμφανιζόμενες μέσα στο πρόγραμμα και λαμβάνοντας υπόψη και το κόστος, επιλέγουμε μονάδα διαίρεσης που δεν χρησιμοποιεί την τεχνική των ΜΕΔ.

Θέμα Θεωρίας Ιουνίου 2016 με εξαρτήσεις

Θεωρήστε επεξεργαστή μερικώς επικαλυπτόμενων λειτουργιών (pipeline) με πέντε βαθμίδες.

α. Ποια είναι η μέγιστη τιμή της επιτάχυνσης του ρυθμού ολοκλήρωσης εντολών σε σχέση με αντίστοιχο επεξεργαστή που δεν υλοποιεί την τεχνική των μερικώς επικαλυπτόμενων λειτουργιών;

β. Να αναφέρετε τους λόγους για τους οποίους δεν μπορούμε να πετύχουμε αυτή την τιμή.

γ. Ποιος είναι ο χρόνος εκτέλεσης μίας εντολής;

Λύση

α. Η μέγιστη τιμή της επιτάχυνσης του ρυθμού ολοκλήρωσης εντολών ισούται με το πλήθος των βαθμίδων, δηλαδή 5.

β. 1. Η χρησιμοποίηση των καταχωρητών μεταξύ των βαθμίδων.

2. Δεν μπορούμε να διαμερίσουμε την εντολή σε 5 βήματα με τέτοιο τρόπο ώστε κάθε βήμα για την εκτέλεσή του να απαιτεί τον ίδιο χρόνο, επομένως δεν έχουν όλες οι βαθμίδες την ίδια καθυστέρηση.

3. Όλες οι εντολές δεν χρησιμοποιούν και τις 5 βαθμίδες.

4. Δομικές εξαρτήσεις.

5. Εξαρτήσεις δεδομένων.

6. Διαδικασιακές εξαρτήσεις.

γ. Ο χρόνος εκτέλεσης μίας εντολής ισούται με το γινόμενο της περιόδου του σήματος χρονισμού και του πλήθους των βαθμίδων του μηχανισμού.

Άσκηση 4.2 από λυσάρι με ΜΕΔ

Για την εκτέλεση μιας διαδικασίας απαιτείται η εκτέλεση της ακολουθίας βημάτων α, β, γ, δ, ε, ζ, η και θ. Τα βήματα αυτά μπορούν να εκτελεστούν από αντίστοιχες μονάδες A, B, Γ, Δ, E, Z, H και Θ με καθυστέρηση 10 ns, 5 ns, 30 ns, 10 ns, 10 ns, 5 ns, 8 ns και 9 ns αντίστοιχα.

α. Να σχεδιάσετε το μηχανισμό μερικώς επικαλυπτόμενων λειτουργιών (pipelining) που επιτυγχάνει το **μέγιστο ρυθμό ολοκλήρωσης διαδικασιών** και για αυτό το ρυθμό, **τον ελάχιστο χρόνο ολοκλήρωσης μιας διαδικασίας (latency)**. Να δικαιολογήσετε την σχεδίαση.

β. Να υπολογίσετε το ρυθμό ολοκλήρωσης διαδικασιών και το χρόνο που απαιτείται για την ολοκλήρωση μιας διαδικασίας. Έχετε στη διάθεσή σας καταχωρητές που η καθυστέρησή τους είναι 1 nsec.

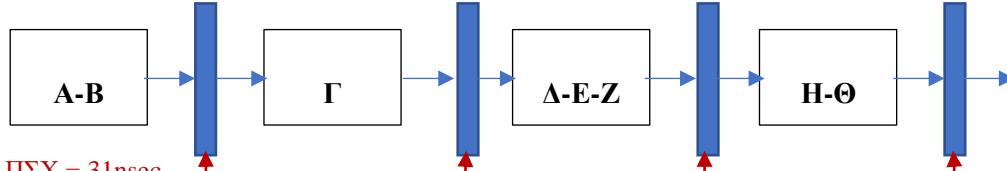
Λύση

α) Οι όροι «**ελάχιστος χρόνος ολοκλήρωσης διαδικασιών**» και «**μέγιστος ρυθμός ολοκλήρωσης διαδικασιών**» είναι **ισοδύναμοι** μεταξύ τους και κάθε φορά που στην εκφρώνηση εμφανίζεται οποιοσδήποτε από αυτούς, σημαίνει ότι πρέπει να **ενοποιηθούμε** (ενώσουμε) μονάδες σε βαθμίδες, προσέχοντας δύο σημεία:

i) Οι μονάδες που ενώνονται να είναι διαδοχικές (συνεχόμενες)

ii) Από την ενοποίηση αυτή να μην ξεπερνάμε τη μέγιστη καθυστέρηση (χωρίς την καθυστέρηση των καταχωρητών). Για παράδειγμα, στην προκειμένη περίπτωση **η μέγιστη καθυστέρηση είναι 30 nsec** και όχι 31 nsec.

Στην προκειμένη περίπτωση, επειδή οι χρόνοι των βαθμίδων είναι **10 5 30 10 10 5 8 9**



ΠΣΧ = πολ άργη μονάδα (μονάδα με τη μεγαλύτερη καθυστέρηση) + χρόνος καταχωρητών = 30 nsec + 1 nsec = 31 nsec

β) Ο ρυθμός ολοκλήρωσης (P.O.) διαδικασιών (διεργασιών) είναι **1 διεργασία κάθε ΠΣΧ**, δηλ. **P.O. = 1 διεργασία κάθε 31 nsec**. Ο χρόνος ολοκλήρωσης μιας διεργασίας (ή χρόνος εκτέλεσης) είναι: **X.O. (ή X.E.) = πλήθος βαθμίδων * ΠΣΧ = 4 * 31 nsec = 124 nsec**.

Άσκηση 4.3 από λυσάρι με ΜΕΔ

Για την εκτέλεση μιας διαδικασίας N απαιτείται η εκτέλεση της ακολουθίας βημάτων α, β, γ, δ, ε, ζ, η και θ. Για την εκτέλεση των βημάτων διατίθενται οι αντίστοιχες μονάδες A, B, Γ, Δ, E, Z, H και Θ με καθυστέρηση, 19 nsec, 5 nsec, 10 nsec, 15 nsec, 9 nsec, 5 nsec, 8 nsec και 7 nsec αντίστοιχα.

α. Να σχεδιάσετε το μηχανισμό μερικώς επικαλυπτόμενων λειτουργιών (pipelining) που επιτυγχάνει το **μέγιστο ρυθμό ολοκλήρωσης διαδικασιών** τύπου N, υπό την προϋπόθεση ότι ο χρόνος ολοκλήρωσης μιας διαδικασίας (latency) θα είναι μικρότερος των 105 nsec. Να δικαιολογήσετε την σχεδίαση.

β. Να υπολογίσετε το ρυθμό ολοκλήρωσης διαδικασιών και το χρόνο που απαιτείται για την ολοκλήρωση μιας διαδικασίας.

Έχετε στη διάθεσή σας καταχωρητές που η καθυστέρησή τους είναι 1 nsec.

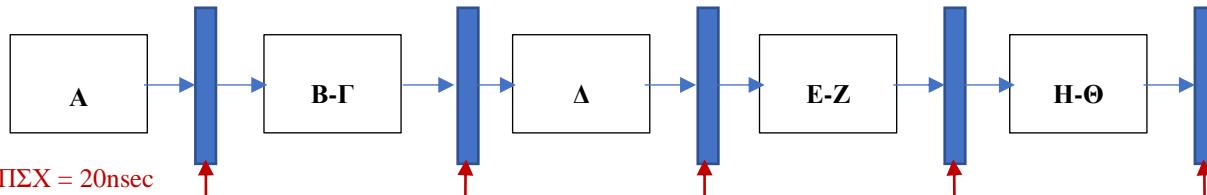
Λύση

Όταν **ενοποιούμε** μονάδες σε βαθμίδες, προσέχουμε δύο σημεία:

i) Οι μονάδες που ενώνονται να είναι διαδοχικές (συνεχόμενες)

ii) Από την ενοποίηση αυτή να μην ξεπερνάμε τη μέγιστη καθυστέρηση (χωρίς την καθυστέρηση των καταχωρητών). Για παράδειγμα, στην προκειμένη περίπτωση **η μέγιστη καθυστέρηση είναι 19 nsec** και όχι 20 nsec.

Ισχύει ότι $X.O. = \text{πλήθος βαθμίδων} * \Pi\Sigma X \leq 105 \Rightarrow \text{πλήθος βαθμίδων} \leq \left\lfloor \frac{105}{20} \right\rfloor = [5.25] = 5$. Αυτό σημαίνει ότι κατά μέγιστο μπορούμε να έχουμε 5 βαθμίδες, τις οποίες επαληθεύουμε στη συνέχεια: **19 | 5 | 10 | 15 | 9 | 5 | 8 | 7**



$\Pi\Sigma X = \piο αργή μονάδα (\muονάδα με τη μεγαλύτερη καθυστέρηση) + χρόνος καταχωρητών = 19 \text{ nsec} + 1 \text{ nsec} = 20 \text{ nsec}$

β) Ο **ρυθμός ολοκλήρωσης** (P.O.) διαδικασιών (διεργασιών) είναι 1 διεργασία κάθε $\Pi\Sigma X$, δηλ. **P.O. = 1 διεργασία κάθε 20 nsec**.

Ο **χρόνος ολοκλήρωσης διεργασιών** (ή χρόνος εκτέλεσης) είναι: **X.O. (ή X.E.) = πλήθος βαθμίδων * $\Pi\Sigma X = 5 * 20 \text{ nsec} = 100 \text{ nsec} < 105 \text{ nsec}$** .

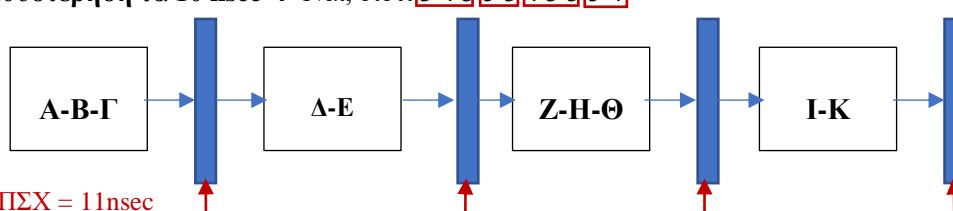
Άσκηση 4.4 από λυσάρι με ΜΕΛ

Για την εκτέλεση μιας διαδικασίας N απαιτείται η εκτέλεση της ακολουθίας βημάτων α, β, γ, δ, ε, ζ, η, θ, κ και λ. Για την εκτέλεση των βημάτων διατίθενται οι αντίστοιχες μονάδες A, B, Γ, Δ, E, Z, H, Θ, K και Λ με καθυστέρηση, 3 nsec, 4 nsec, 3 nsec, 5 nsec, 3 nsec, 4 nsec, 3 nsec, 3 nsec, 5 nsec και 4 nsec αντίστοιχα. Έχετε στη διάθεσή σας καταχωρητές που η καθυστέρησή τους είναι 1 nsec. Να σχεδιάσετε το μηχανισμό μερικώς επικαλυπτόμενων λειτουργιών (pipelining) που επιτυγχάνει το μέγιστο δυνατό ρυθμό ολοκλήρωσης διαδικασιών υπό την προϋπόθεση ότι η **συνολική καθυστέρηση** (latency) του μηχανισμού μερικώς επικαλυπτόμενων λειτουργιών είναι μικρότερη ή ίση των 44 nsec. Να δικαιολογήσετε τη σχεδίαση

Λύση

Στην περίπτωση αυτή έχουμε ως δεδομένο ότι η **συνολική καθυστέρηση είναι μικρότερη ή ίση των 44 nsec**. Αυτό σημαίνει ότι αν αθροίσουμε τις καθυστέρησεις των μονάδων που δίνονται, **μαζί** με αυτές των καταχωρητών ανάμεσά τους, τότε το συνολικό άθροισμα δεν θα πρέπει να ξεπερνά τα 44 nsec. Δηλαδή $3 + 4 + 3 + 5 + 3 + 4 + 3 + 3 + 5 + 4 + \lambda * 1 \text{ nsec} \leq 44 \text{ nsec} \Rightarrow \lambda \leq 7$ καταχωρητές, επομένως και 7 υπομονάδες, αφού για κάθε υπομονάδα υπάρχει (αντίστοιχεί) και ένας καταχωρητής. Επειδή $X.O. = \text{πλήθος βαθμίδων} * \Pi\Sigma X \leq 44 \text{ nsec} \Rightarrow \Pi\Sigma X \leq \left\lfloor \frac{44}{7} \right\rfloor = [6.28] = 6 \text{ nsec} \Rightarrow -1 \text{ nsec}$ (καθυστέρηση των καταχωρητών) = 5 nsec.

Επομένως, **το ερώτημα είναι αν μπορούμε να έχουμε 7 βαθμίδες, με μέγιστη καθυστέρηση τα 5 nsec** → Όχι. Για το λόγο αυτό θα πρέπει να μειώσουμε τον αριθμό των μονάδων, από 7 σε 6, οπότε $\Pi\Sigma X \leq \left\lfloor \frac{44}{6} \right\rfloor = [7.33] = 7 \text{ nsec} \Rightarrow -1 \text{ nsec}$ (καθυστέρηση των καταχωρητών) = 6 nsec. Επομένως, **το ερώτημα είναι αν μπορούμε να έχουμε 6 βαθμίδες, με μέγιστη καθυστέρηση τα 6 nsec** → Όχι. Συνεχίζουμε να μειώνουμε περαιτέρω τον αριθμό των μονάδων, μέχρι να φτάσουμε στις 4 μονάδες, οπότε έχουμε: $\Pi\Sigma X \leq \left\lfloor \frac{44}{4} \right\rfloor = [11] = 11 \text{ nsec} \Rightarrow -1 \text{ nsec} = 10 \text{ nsec}$. Επομένως, **το ερώτημα είναι αν μπορούμε να έχουμε 4 βαθμίδες, με μέγιστη καθυστέρηση τα 10 nsec** → Ναι, διότι **3 | 4 | 3 | 5 | 3 | 4 | 3 | 3 | 5 | 4**



$\Pi\Sigma X = \piο αργή μονάδα (\muονάδα με τη μεγαλύτερη καθυστέρηση) + χρόνος καταχωρητών = 10 \text{ nsec} + 1 \text{ nsec} = 11 \text{ nsec}$

Αν ζητηθεί και ο **ρυθμός ολοκλήρωσης** (P.O.) διαδικασιών (διεργασιών), θα έχουμε ότι ο **P.O. = 1 διεργασία κάθε 11 nsec**. Ο **χρόνος ολοκλήρωσης διεργασιών** είναι: **X.O. (ή X.E.) = πλήθος βαθμίδων * $\Pi\Sigma X = 4 * 11 \text{ nsec} = 44 \text{ nsec} \leq 44 \text{ nsec}$** .

Άσκηση 4.5 από λυσάρι με ΜΕΛ

Θεωρείστε δύο επεξεργαστές M1 και M2 με το ίδιο σύνολο εντολών. Οι εντολές διακρίνονται σε 4 κατηγορίες X₁, X₂, X₃ και X₄. Κάθε μία εντολή αποτελείται από μια σειρά υπολειτουργιών που μπορούν να ομαδοποιηθούν σε 5 ομάδες και κάθε ομάδα εκτελείται σε μία από τις υπομονάδες ΠΕ, ΑΕ, ΕΕ (ΕΠ), ΠΜ και ΑΑ. Ο χρόνος απασχόλησης κάθε υπομονάδας εξαρτάται από το είδος της εντολής και δίνεται στον Πίνακα 4.5.1.

Πίνακας 4.5.1

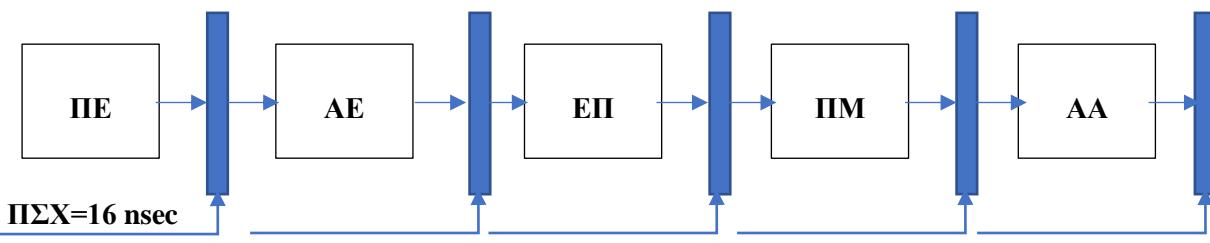
Κατηγορίες εντολών	ΠΕ ns	ΑΕ ns	ΕΕ (ΕΠ) ns	ΠΜ ns	ΑΑ ns
X ₁	15	5	5	10	5
X ₂	10	5	5	10	5
X ₃	10	5	5	5	5
X ₄	10	5	5	0	5

Ο επεξεργαστής M1 έχει ένα μηχανισμό μερικώς επικαλυπτόμενων λειτουργιών (pipeline) με 5 βαθμίδες ΠΕ, ΑΕ, ΕΕ, ΠΜ και ΑΑ, που χρησιμοποιείται για την εκτέλεση όλων των εντολών. Ένας καταχωρητής μεταξύ δύο μονάδων του M1 βάζει καθυστέρηση 1 ns. Ο επεξεργαστής M2 δεν υλοποιεί την τεχνική μερικώς επικαλυπτόμενων λειτουργιών, χρησιμοποιεί όμως τις ίδιες υπομονάδες, η διάρκεια του κύκλου ρολογιού του είναι 5 ns και ολοκληρώνει κάθε εντολή σε όσους κύκλους ρολογιού απαιτούνται.

- α. Να σχεδιάσετε το μηχανισμό μερικώς επικαλυπτόμενων λειτουργιών του M1 δίνοντας στο σχήμα όλη την πληροφορία.
- β. Να υπολογίσετε το χρόνο εκτέλεσης (latency) κάθε είδους εντολής στους επεξεργαστές M1 και M2.
- γ. Να υπολογίσετε την επιτάχυνση του ρυθμού ολοκλήρωσης εντολών στην περίπτωση που
 1. εκτελούνται 1000 εντολές του ίδιου είδους X_i με i = 1, 2, 3 και 4.
 2. εκτελούνται 1000 εντολές εκ των οποίων το 30% ανήκει στην κατηγορία X₁, το 40% στην κατηγορία X₂, το 20% στην κατηγορία X₃ και το 10% στην κατηγορία X₄.

Λύση

α) Έχουμε δύο επεξεργαστές, τον M1 που υποστηρίζει το μηχανισμό των μερικώς επικαλυπτόμενων λειτουργιών (ΜΕΛ) και τον M2 που δεν χρησιμοποιεί το συγκεκριμένο μηχανισμό. Για τον επεξεργαστή M1 ο χρόνος ολοκλήρωσης μιας διεργασίας (ΧΟ ή ΧΕ) υπολογίζεται με τον κλασικό τρόπο, δηλ. **X.O. = πλήθος βαθμίδων * ΠΣΧ = 5 * 16 nsec = 80 nsec**. Η ΠΣΧ (Περίοδος Σήματος Χρονισμού) υπολογίζεται ως εξής: **ΠΣΧ = μεγαλύτερη καθυστέρηση βαθμίδας + καθυστέρηση καταχωρητή = 15nsec + 1nsec = 16 nsec**.



β) Για τον επεξεργαστή M2, που δεν υποστηρίζει το μηχανισμό ΜΕΛ, δεν θα υπολογιστεί ένας X.O., αλλά θα υπολογιστούν συνολικά 4 χρόνοι ολοκλήρωσης, όσα δηλαδή είναι και τα είδη των εντολών. Πιο συγκεκριμένα, αθροίζουμε για κάθε εντολή (από X₁ έως X₄) τους χρόνους σε κάθε μονάδα:

$$X_1(M2) = 15 + 5 + 5 + 10 + 5 = 40 \text{ nsec.}$$

$$X_2(M2) = 10 + 5 + 5 + 10 + 5 = 35 \text{ nsec}$$

$$X_3(M2) = 10 + 5 + 5 + 5 + 5 = 30 \text{ nsec}$$

$$X_4(M2) = 10 + 5 + 5 + 0 + 5 = 25 \text{ nsec}$$

γ) Για τον υπολογισμό του EPO (Επιτάχυνσης Ρυθμού Ολοκλήρωσης) όταν εκτελούνται 1000 διεργασίες του είδους X₁ έχουμε: $EPO = \frac{K * χρόνος εκτέλεσης χωρίς pipeline}{(N+K-1) * χρόνος εκτέλεσης με pipeline} = \frac{1000 * 40 \text{ nsec}}{(5+1000-1) * \text{ΠΣΧ}} = \frac{1000 * 40 \text{ nsec}}{1004 * 16 \text{ nsec}} = 2.49$. Αν θεωρήσουμε ότι εκτελούνται 1000 διεργασίες του είδους X₂ έχουμε: $EPO = \frac{K * χρόνος εκτέλεσης χωρίς pipeline}{(N+K-1) * χρόνος εκτέλεσης με pipeline} = \frac{1000 * 35 \text{ nsec}}{(5+1000-1) * \text{ΠΣΧ}} = \frac{1000 * 35 \text{ nsec}}{1004 * 16 \text{ nsec}} = 2.17$.

Ομοίως αν εκτελούσαμε 1000 εντολές του είδους X₃ ή X₄.

Κανόνας: ο χρόνος εκτέλεσης με pipeline στον παρανομαστή του EPO είναι το ΠΣΧ.

γ2) Για τον υπολογισμό του EPO (Επιτάχυνσης Ρυθμού Ολοκλήρωσης) όταν εκτελούνται 1000 διεργασίες και των τεσσάρων ειδών διεργασιών (εντολών) με κάποιο ποσοστό, δηλαδή 30% του είδους X_1 , 40% του είδους X_2 , 20% του είδους X_3 και 10% του είδους X_4 , το EPO = $\frac{K * \text{χρόνος εκτέλεσης χωρίς pipeline}}{(N+K-1) * \text{χρόνος εκτέλεσης με pipeline}}$ = $\frac{0.3 * 1000 * 40 \text{ nsec} + 0.4 * 1000 * 35 \text{ nsec} + 0.2 * 1000 * 30 \text{ nsec} + 0.1 * 1000 * 25 \text{ nsec}}{(5+1000-1) * \text{ΠΣΧ}} = 2.15$. Σε όλες τις περιπτώσεις υπολογισμού του EPO, ο παρονομαστής δεν αλλάζει και επίσης ο χρόνος εκτέλεσης με pipeline που υπάρχει στον παρονομαστή, είναι ουσιαστικά το ΠΣΧ. Το EPO δεν έχει μονάδες και είναι καθαρός αριθμός.

Θέμα Ιοννίου 2014 με ΜΕΛ

Για την εκτέλεση μιας διαδικασίας απαιτείται η εκτέλεση της ακολουθίας βημάτων $\alpha, \beta, \gamma, \delta, \varepsilon, \zeta, \eta, \theta, \kappa$ και λ . Για την εκτέλεση των βημάτων διατίθενται οι αντίστοιχες μονάδες $A, B, \Gamma, \Delta, E, Z, H, \Theta, K$ και Λ με καθυστέρηση, 3 nsec, 4 nsec, 3 nsec, 5 nsec, 3 nsec, 4 nsec, 3 nsec, 5 nsec και 4 nsec αντίστοιχα. Έχετε στη διάθεσή σας καταχωρητές που η καθυστέρησή τους είναι 1 nsec. Να σχεδιάσετε το μηχανισμό μερικώς επικαλυπτόμενων λειτουργιών (pipelining) που επιτυγχάνει την ολοκλήρωση μιας διαδικασίας, το πολύ κάθε 10 nsec με την ελάχιστη συνολική καθυστέρηση (latency) του μηχανισμού μερικώς επικαλυπτόμενων λειτουργιών. Να δικαιολογήσετε πως οδηγηθήκατε στη σχεδίαση που προτείνατε.

Λύση

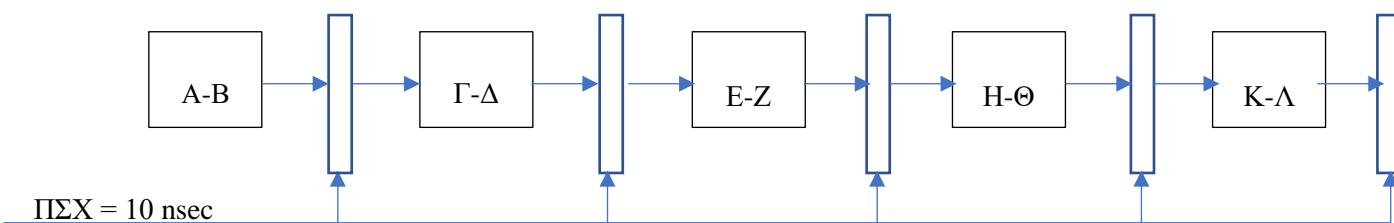
Από την εκφώνηση της άσκησης δίνονται τα εξής δεδομένα:

$\alpha, \beta, \gamma, \delta, \varepsilon, \zeta, \eta, \theta, \kappa, \lambda$: βήματα

$A, B, \Gamma, \Delta, E, Z, H, \Theta, K, \Lambda$: μονάδες

$3, 4, 3, 5, 3, 4, 3, 3, 5, 4$: καθυστέρηση (nsec)

Από τη στιγμή που θέλουμε ο σχεδιαζόμενος μηχανισμός επικαλυπτόμενων λειτουργιών (ΜΕΛ) να επιτυγχάνει την ολοκλήρωση μιας διαδικασίας κάθε 10 nsec, καταλαβαίνουμε ότι αναφερόμαστε στο ρυθμό ολοκλήρωσης διεργασίων (P.O.), ο οποίος θέλουμε να είναι 1 διεργασία κάθε 10 nsec (ΠΣΧ). Γενικότερα ισχύει ότι $P.O. = 1 \text{ διεργασία κάθε ΠΣΧ}$. Άρα καταλαβαίνουμε ότι το ΠΣΧ = 10 nsec και από αυτό το μέγεθος αφαιρούμε την καθυστέρηση των καταχωρητών, για να υπολογίσουμε τη μέγιστη καθυστέρηση που θέλουμε να έχουμε, προκειμένου να ενοποιήσουμε τις μονάδες σε βαθμίδες. Αυτό το κάνουμε διότι η εκφώνηση αναφέρει τον όρο ελάχιστη συνολική καθυστέρηση του μηχανισμού επικαλυπτόμενων λειτουργιών. Με βάση αυτή τη μέγιστη καθυστέρηση των 9 nsec, προκύπτει από την ενοποίηση των μονάδων σε βαθμίδες, το ακόλουθο σχήμα:

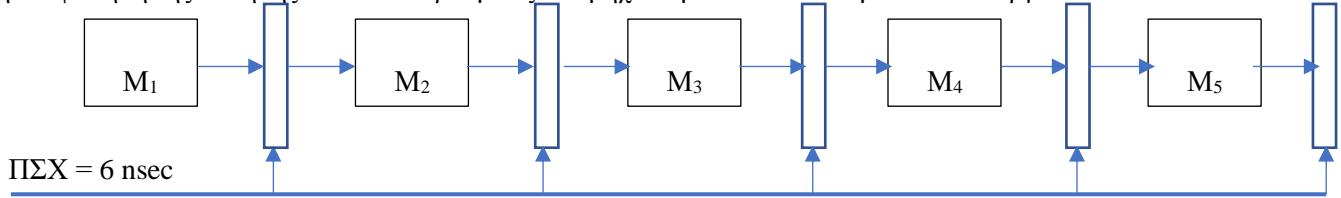


Θέμα Ιοννίου 2012 με ΜΕΛ

Θεωρείστε μηχανισμό μερικώς επικαλυπτόμενων λειτουργιών (pipelining) των πέντε βαθμίδων, M_1 (καθυστέρηση 4 nsec), M_2 (καθυστέρηση 5 nsec), M_3 (καθυστέρηση 2 nsec), M_4 (καθυστέρηση 3 nsec) και M_5 (καθυστέρηση 4 nsec) με ενδιάμεσους καταχωρητές που κάθε ένας βάζει καθυστέρηση 1 nsec. Να βρείτε το α) ρυθμό ολοκλήρωσης διεργασιών β) το χρόνο που απαιτείται για την ολοκλήρωση μίας διεργασίας και γ) το χρόνο που απαιτείται για ολοκλήρωση 100 διεργασιών.

Λύση

Από την εκφώνηση της άσκησης δίνονται οι βαθμίδες του μηχανισμού επικαλυπτόμενων λειτουργιών:



α) Ισχύει ότι ο ρυθμός ολοκλήρωσης διεργασιών (P.O.) = 1 διεργασία κάθε ΠΣΧ, όπου $\Pi\Sigma\chi = \text{μεγαλύτερη καθυστέρηση} - \text{καθυστέρηση των καταχωρητών} = 5 \text{ nsec} + 1 \text{ nsec} = 6 \text{ nsec}$.

β) $X.O. = \text{πλήθος βαθμίδων} * \Pi\Sigma\chi = 5 * 6 \text{ nsec} = 30 \text{ nsec}$

γ) Ο χρόνος ολοκλήρωσης που **απαιτείται για την ολοκλήρωση 100 διεργασιών** υπολογίζεται πάντα από τον παρανομαστή του EPO και είναι X.O. = (N - 1 + K) x ΠΣΧ = (5 - 1 + 100) x 6 nsec = 104 x 6 nsec = 624 nsec, όπου N = πλήθος βαθμίδων και K = σύνολο διεργασιών.

Παρατήρηση 1: Αν η εκφώνηση ζητούσε τον υπολογισμό του **μέγιστου ρυθμού ολοκλήρωσης διεργασιών/ελάχιστος χρόνος ολοκλήρωσης μιας διεργασίας**, τότε θα έπρεπε να **ενοποιήσουμε** τις μονάδες σε βαθμίδες, εφαρμόζοντας τα δύο κριτήρια ενοποίησης: οι βαθμίδες να είναι διαδοχικές και να μην ξεπερνάμε από την ενοποίηση τη μέγιστη καθυστέρηση (χωρίς την καθυστέρηση των καταχωρητών). Σε μια τέτοια περίπτωση, θα είχαμε 4 βαθμίδες, αφού θα ενοποιούσαμε μαζί τις βαθμίδες με καθυστέρηση 2nsec και 3 nsec.

Παρατήρηση 2: Για τον υπολογισμό του χρόνου ολοκλήρωσης διεργασιών που είναι στο **πλήθος > 1**, χρησιμοποιείται ο παρονομαστής του EPO.

Θέμα Σεπτεμβρίου 2014 με ιδανικό ΜΕΛ

Για την εκτέλεση μιας διαδικασίας N απαιτείται η εκτέλεση της ακολουθίας βημάτων α, β, γ, δ, ε, ζ, η και θ. Για την εκτέλεση των βημάτων διατίθενται οι αντίστοιχες μονάδες A, B, Γ, Δ, E, Z, H και Θ με καθυστέρηση 160 psec, 50 psec, 100 psec, 150 psec, 90 psec, 50 psec, 40 psec και 50 psec αντίστοιχα. Η καθυστέρηση των καταχωρητών είναι 10 psec.

a. Να σχεδιάσετε **μηχανισμό μερικώς επικαλυπτόμενων λειτουργιών των πέντε βαθμίδων που να επιτυγχάνει το μέγιστο δυνατό ρυθμό ολοκλήρωσης** και να τον συγκρίνεται με το μέγιστο ρυθμό που θα μπορούσε να επιτευχθεί με **ιδανικό μηχανισμό μερικώς επικαλυπτόμενων λειτουργιών των πέντε βαθμίδων**. Να δικαιολογήσετε τυχόν διαφορές.

β. Να υπολογίσετε το χρόνο εκτέλεσης (latency) μιας διαδικασίας.

Αύση

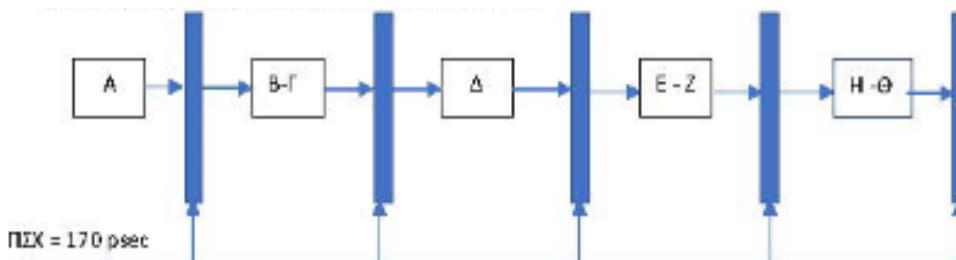
α. Όταν αναφέρει η εκφώνηση τον όρο **μέγιστο δυνατό ρυθμό ολοκλήρωσης**, εννοεί να ενοποιήσουμε τις μονάδες σε βαθμίδες, σύμφωνα με τα δύο κριτήρια ενοποίησης. Ένας **ιδανικός ΜΕΛ** διαθέτει τρία χαρακτηριστικά:

- Δεν έχει εξαρτήσεις.

- Όλες οι βαθμίδες έχουν την **ίδια** καθυστέρηση.

- Οι καταχωρητές μεταξύ των βαθμίδων **δεν** προσθέτουν κάποια καθυστέρηση.

Η ΠΣΧ = μεγαλύτερη καθυστέρηση + καθυστέρηση καταχωρητών = 160 psec + 10 = 170 psec. Ο P.O. είναι 1 διεργασία κάθε 170 psec. Για ένα **ιδανικό ΜΕΛ** η ΠΣΧ = 160 psec. Ο P.O. είναι 1 διεργασία κάθε 160 psec. Η διαφορά οφείλεται στο ότι ένας ιδανικός ΜΕΛ δεν έχει καθυστέρηση καταχωρητών. Στη συνέχεια φαίνεται ο μηχανισμός επικαλυπτόμενων λειτουργιών που επιτυγχάνει το μέγιστο ρυθμό ολοκλήρωσης.



β. Ο χρόνος εκτέλεσης (latency) μιας διαδικασίας είναι: X.O. = πλήθος βαθμίδων x ΠΣΧ = 5 x 170 psec = 850 psec. Αντίστοιχα, στον ιδανικό ΜΕΛ, ο χρόνος εκτέλεσης (latency) μιας διαδικασίας είναι: X.O. = πλήθος βαθμίδων x ΠΣΧ = 5 x 160 psec = 800 psec.

Θέμα Σεπτεμβρίου 2020 με ΜΕΛ

Για την εκτέλεση μιας διαδικασίας απαιτείται η εκτέλεση της ακολουθίας βημάτων α, β, γ, δ, ε, ζ, η, θ και ι. Τα βήματα αυτά μπορούν να εκτελεστούν από τις αντίστοιχες μονάδες A, B, Γ, Δ, E, Z, H, Θ και Ι με καθυστέρηση 2, 4, 3, 5, 7, 2, 9, 4, 5 nsec αντίστοιχα. Έχετε στη διάθεση σας καταχωρητές με μέγιστη καθυστέρηση διάδοσης 1 nsec. Να σχεδιάσετε το μηχανισμό μερικώς επικαλυπτόμενων λειτουργιών (pipelining) που επιτυγχάνει το **μέγιστο ρυθμό ολοκλήρωσης διαδικασιών** και για αυτό το ρυθμό **τον ελάχιστο χρόνο εκτέλεσης μιας διαδικασίας** (latency). Από πόσες βαθμίδες αποτελείται ο μηχανισμός μερικώς επικαλυπτόμενων λειτουργιών που σχεδιάσατε; Να υπολογίσετε την περίοδο του σήματος χρονισμού (clock). Να υπολογίσετε σε κάθε πόσα nsec ολοκληρώνεται μία νέα διαδικασία (ρυθμός ολοκλήρωσης διαδικασιών). Αγνοήστε τις αρχικές συνθήκες. Να υπολογίσετε πόσος χρόνος απαιτείται για την εκτέλεση μιας διαδικασίας (latency). Ποια είναι η μέγιστη επιτάχυνση του ρυθμού ολοκλήρωσης διεργασιών στο μηχανισμό μερικώς επικαλυπτόμενων λειτουργιών που σχεδιάσατε σε σχέση με ένα μηχανισμό που

δεν υλοποιεί την τεχνική των μερικώς επικαλυπτόμενων λειτουργιών; Σε όλες τις περιπτώσεις η απάντησή σας να είναι ένας τριψήφιος ακέραιος αριθμός, πχ. [002].

Από πόσες βαθμίδες αποτελείται ο μηχανισμός μερικώς επικαλυπτόμενων λειτουργιών που σχεδιάσατε; Ποια είναι η περίοδο του σήματος χρονισμού (clock) σε nsec; Σε κάθε πόσα nsec ολοκληρώνεται και μία νέα διαδικασία; Πόσος χρόνος απαιτείται για την εκτέλεση μιας διαδικασίας (latency); Ποια είναι η μέγιστη επιτάχυνση του ρυθμού ολοκλήρωσης διεργασιών στο μηχανισμό μερικώς επικαλυπτόμενων λειτουργιών που σχεδιάσατε σε σχέση με ένα μηχανισμό που δεν υλοποιεί την τεχνική των μερικώς επικαλυπτόμενων λειτουργιών; .

Λύση

- Θα χρησιμοποιήσουμε **5 βαθμίδες** που είναι: A – B – Γ, Δ, E – Z, H, Θ – I → 005
- Η περίοδος του σήματος χρονισμού (ΠΣΧ): $9 + 1 = 10$ nsec → 010. Αυτή προκύπτει από τη μεγαλύτερη καθυστέρηση + την καθυστέρηση των καταχωρητών.
- Μια νέα διαδικασία θα ολοκληρώνεται κάθε 10 nsec. Αυτός είναι ο ρυθμός ολοκλήρωσης (P.O.) διεργασιών → 010. (Ισχύει ότι ολοκληρώνεται και μια νέα διεργασία κάθε ΠΣΧ).
- X.O. = πλήθος βαθμίδων x ΠΣΧ = 5×10 nsec = 50 nsec → 050
- Μέγιστη επιτάχυνση του ρυθμού ολοκλήρωσης διεργασιών = 5 → 005 (θεωρητικά η μέγιστη τιμή του EPO = N, όπου N = πλήθος βαθμίδων).

Θέμα Σεπτεμβρίου 2020 με ΜΕΛ

Για την εκτέλεση μιας διαδικασίας απαιτείται η εκτέλεση της ακολουθίας βημάτων α, β, γ, δ, ε, ζ, η, θ και i. Τα βήματα αυτά μπορούν να εκτελεστούν από τις αντίστοιχες μονάδες A, B, Γ, Δ, E, Z, H, Θ και I με καθυστέρηση 19, 19, 11, 14, 43, 17, 40, 14, 17 nsec αντίστοιχα. Έχετε στη διάθεση σας καταχωρητές με μέγιστη καθυστέρηση διάδοσης 6 nsec. Να σχεδιάσετε το μηχανισμό μερικώς επικαλυπτόμενων λειτουργιών (pipelining) που επιτυγχάνει το **μέγιστο ρυθμό ολοκλήρωσης διαδικασιών** και για αυτό το ρυθμό τον **ελάχιστο χρόνο εκτέλεσης μιας διαδικασίας (latency)**. Από πόσες βαθμίδες αποτελείται ο μηχανισμός μερικώς επικαλυπτόμενων λειτουργιών που σχεδιάσατε; Να υπολογίσετε την περίοδο του σήματος χρονισμού (clock). Να υπολογίσετε σε κάθε πόσα nsec ολοκληρώνεται μία νέα διαδικασία (ρυθμός ολοκλήρωσης διαδικασιών). Αγνοήστε τις αρχικές συνθήκες. Να υπολογίσετε πόσος χρόνος απαιτείται για την εκτέλεση μιας διαδικασίας (latency). Ποια είναι η μέγιστη επιτάχυνση του ρυθμού ολοκλήρωσης διεργασιών στο μηχανισμό μερικώς επικαλυπτόμενων λειτουργιών που σχεδιάσατε σε σχέση με ένα μηχανισμό που δεν υλοποιεί την τεχνική των μερικώς επικαλυπτόμενων λειτουργιών; Σε όλες τις περιπτώσεις η απάντηση να είναι ένας τριψήφιος ακέραιος αριθμός, πχ. [002].

Από πόσες βαθμίδες αποτελείται ο μηχανισμός μερικώς επικαλυπτόμενων λειτουργιών που σχεδιάσατε; Ποια είναι η περίοδο του σήματος χρονισμού (clock) σε nsec; Σε κάθε πόσα nsec ολοκληρώνεται και μία νέα διαδικασία; Πόσος χρόνος απαιτείται για την εκτέλεση μιας διαδικασίας (latency); Ποια είναι η μέγιστη επιτάχυνση του ρυθμού ολοκλήρωσης διεργασιών στο μηχανισμό μερικώς επικαλυπτόμενων λειτουργιών που σχεδιάσατε σε σχέση με ένα μηχανισμό που δεν υλοποιεί την τεχνική των μερικώς επικαλυπτόμενων λειτουργιών; .

Λύση

- Θα χρησιμοποιήσουμε **6 βαθμίδες** που είναι: A – B, Γ – Δ, E, Z, H, Θ – I → 006
- Η περίοδος του σήματος χρονισμού (ΠΣΧ): $43 + 6 = 49$ nsec → 049. Αυτή προκύπτει από τη μεγαλύτερη καθυστέρηση + την καθυστέρηση των καταχωρητών.
- Μια νέα διαδικασία θα ολοκληρώνεται κάθε 49 nsec. Αυτός είναι ο ρυθμός ολοκλήρωσης (P.O.) διεργασιών → 049 (Ισχύει ότι ολοκληρώνεται και μια νέα διεργασία κάθε ΠΣΧ).
- X.O. = πλήθος βαθμίδων x ΠΣΧ = 6×49 nsec = 294 nsec → 294
- Μέγιστη επιτάχυνση του ρυθμού ολοκλήρωσης διεργασιών = 6 → 006 (θεωρητικά η μέγιστη τιμή του EPO = N, όπου N = πλήθος βαθμίδων).

Θέμα Ιουνίου 2016 με εξαρτήσεις

Δίνεται επεξεργαστής μερικώς επικαλυπτόμενων λειτουργιών με πέντε βαθμίδες (ΠΕ: Προσκόμιση εντολής, ΑΕ: Αποκωδικοποίηση εντολών και ανάγνωση καταχωρητών, ΕΠ: Εκτέλεση πράξης, ΠΜ: Προσπέλαση μνήμης, ΑΑ: Αποθήκευση αποτελεσμάτων) και **κοινή** κρυφή μνήμη εντολών και δεδομένων. Θεωρήστε ότι ο μηχανισμός **δεν** υλοποιεί την τεχνική παροχέτευσης.

πρόγραμμα
LOAD r1, (r2)

//r1 ← M(r2)

```

ADD r3, r4, r5          // r3+r4 → r5
SUB r1, r3, r6          // r1-r3 → r6
ADD r8, r6, r9          // r8+r6 → r9
AND r3, r4, r10         // r3&r4 → r10

```

α.1. Συμπληρώστε τον επόμενο πίνακα και αναγνωρίστε τις εξαρτήσεις που υπάρχουν (που εμφανίζονται και τι είδους είναι) για το πρόγραμμα που δίνεται στον διπλανό πίνακα.

εντολή	περίοδος σήματος χρονισμού												
	λ	$\lambda+1$	$\lambda+2$	$\lambda+3$	$\lambda+4$	$\lambda+5$	$\lambda+6$	$\lambda+7$	$\lambda+8$	$\lambda+9$	$\lambda+10$	$\lambda+11$	$\lambda+12$

α.2. Να αναφέρετε τις τεχνικές που θα χρησιμοποιήσετε για να επιλύσετε το πρόβλημα των εξαρτήσεων και να το εφαρμόσετε συμπληρώνοντας τον επόμενο πίνακα.

εντολή	περίοδος σήματος χρονισμού												
	λ	$\lambda+1$	$\lambda+2$	$\lambda+3$	$\lambda+4$	$\lambda+5$	$\lambda+6$	$\lambda+7$	$\lambda+8$	$\lambda+9$	$\lambda+10$	$\lambda+11$	$\lambda+12$

εντολή	περίοδος σήματος χρονισμού								
	λ	$\lambda+1$	$\lambda+2$	$\lambda+3$	$\lambda+4$	$\lambda+5$	$\lambda+6$	$\lambda+7$	$\lambda+8$
LOAD r1, (r2)	ΠΕ	ΑΕ	ΕΠ	ΠΜ	AA				
ADD r3, r4, r5	ΠΕ	ΑΕ	1	ΕΠ	ΠΜ	AA			
SUB r1, r3, r6	ΠΕ		ΑΕ	ΕΠ	ΠΜ	AA			
ADD r8, r6, r9			ΠΕ	ΑΕ	ΕΠ	ΠΜ	AA		
AND r3, r4, r10				ΠΕ	ΑΕ	ΕΠ	ΠΜ	AA	

Άνση

Η υπ' αριθμόν 1 εξάρτηση είναι δομική. Η υπ' αριθμόν 2 εξάρτηση είναι AME. Η υπ' αριθμόν 3 εξάρτηση είναι AME.a.2.

Το πρόβλημα της δομικής εξάρτησης μπορεί να λυθεί μόνο με την τεχνική του κλειδώματος βαθμίδων.

Οι εξαρτήσεις τύπου AME μπορούν να επιλυθούν είτε με την τεχνική κλειδώματος είτε με τη χρησιμοποίηση εντολών NOP.

Στη συνέχεια θα δώσουμε και τις δύο λύσεις. Ως απάντηση στην εξέταση η μία εκ των δύο ήταν αρκετή.

1. Λύση με κλείδωμα για την επίλυση του προβλήματος της δομικής εξάρτησης και χρήση εντολών NOP για τις εξαρτήσεις τύπου AME.

εντολή	περίοδος σήματος χρονισμού												
	λ	$\lambda+1$	$\lambda+2$	$\lambda+3$	$\lambda+4$	$\lambda+5$	$\lambda+6$	$\lambda+7$	$\lambda+8$	$\lambda+9$	$\lambda+10$	$\lambda+11$	$\lambda+12$
LOAD r1, (r2)	ΠΕ	ΑΕ	ΕΠ	ΠΜ	AA								
ADD r3, r4, r5	ΠΕ	ΑΕ	ΕΠ	ΠΜ	AA								
NOP		ΠΕ	ΑΕ	ΕΠ	ΠΜ	AA							
SUB r1, r3, r6			X	ΠΕ	ΑΕ	ΕΠ	ΠΜ	AA					
NOP				ΠΕ	ΑΕ	ΕΠ	ΠΜ	AA					
NOP					ΠΕ	ΑΕ	ΕΠ	ΠΜ	AA				
ADD r8, r6, r9					ΠΕ	ΑΕ	ΕΠ	ΠΜ	AA				
AND r3, r4, r10						ΠΕ	ΑΕ	ΕΠ	ΠΜ	AA			

2. Λύση με κλείδωμα τόσο για την επίλυση του προβλήματος της δομικής εξάρτησης όσο και των εξαρτήσεων τύπου AME.

εντολή	περίοδος σήματος χρονισμού											
	λ	$\lambda+1$	$\lambda+2$	$\lambda+3$	$\lambda+4$	$\lambda+5$	$\lambda+6$	$\lambda+7$	$\lambda+8$	$\lambda+9$	$\lambda+10$	$\lambda+11$
LOAD r1, (r2)	ΠΕ	ΑΕ	ΕΠ	ΠΜ	AA							
ADD r3, r4, r5		ΠΕ	ΑΕ	ΕΠ	ΠΜ	AA						
SUB r1, r3, r6			ΠΕ	AE'	AE	ΕΠ	ΠΜ	AA				
ADD r8, r6, r9				X	ΠΕ	AE'	X	AE	ΕΠ	ΠΜ	AA	
AND r3, r4, r10									ΠΕ	ΑΕ	ΕΠ	ΠΜ
										AA		

Είναι απαραίτητη η χρήση κλειδώματος X για την επίλυση της δομικής εξάρτησης, ακόμη και αν το βήμα ΠΕ της εντολής ADD ξεκινά τη χρονική περίοδο $\lambda + 4$, για να φανεί ότι επιλύεται η δομική εξάρτηση του ΜΕΛ με πάγωμα.

Άσκηση με όλα τα είδη εξαρτήσεων

Θεωρείστε επεξεργαστή μερικώς επικαλυπτόμενων λειτουργιών (pipeline) με πέντε βαθμίδες ΠΕ, ΑΕ, ΕΠ, ΠΜ και AA ο οποίος διαθέτει κοινή κρυφή μνήμη εντολών και δεδομένων με μια πόρτα ανάγνωσης/εγγραφής. Θεωρήστε ότι είναι δυνατή η εγγραφή και ανάγνωση του εγγραφέντος δεδομένου στον ίδιο κύκλο ρολογιού. Δίνεται το παρακάτω πρόγραμμα.

```

LOAD r1, (r2)      // r1 ← M(r2)
ADD r2, r4          // r2 ← r2 + r4
ADD r1, r3          // r1 ← r1 + r3
LOAD r8, (r1)        // r8 ← M(r1)
SUB r5, r7          // r5 ← r5 - r7
SUB r1, r8          // r1 ← r1 - r8
ADD r1, r9          // r1 ← r1 + r9
STORE r1, (r5)       // r1 → M(r5)

```

Για κάθε μία των περιπτώσεων i και ii να απαντήσετε στα ερωτήματα α και β.

- Στο μηχανισμό μερικώς επικαλυπτόμενων λειτουργιών **δεν έχει υλοποιηθεί** η τεχνική παροχέτευσης (bypassing).
- Στο μηχανισμό μερικώς επικαλυπτόμενων λειτουργιών **έχει υλοποιηθεί** η τεχνική παροχέτευσης (bypassing).
 - Να φτιάξετε το χρονικό διάγραμμα χρήσης των βαθμίδων, να σημειώσετε τις εξαρτήσεις και να αναφέρετε το είδος κάθε εξάρτησης (απλά να αναφέρετε το είδος, δεν χρειάζεται να αιτιολόγηση).
 - Χωρίς να αλλάξετε τη σειρά των εντολών να κάνετε τις απαιτούμενες ενέργειες ώστε να μην υπάρχει πρόβλημα στην ορθή εκτέλεση του τμήματος προγράμματος. Να ξαναφτιάξετε το χρονικό διάγραμμα χρήση των βαθμίδων.

Λύση

α) Επίλυση χωρίς την τεχνική παροχέτευσης – Εντοπισμός εξαρτήσεων

	1	$\lambda+1$	$\lambda+2$	$\lambda+3$	$\lambda+4$	$\lambda+5$	$\lambda+6$	$\lambda+7$	$\lambda+8$	$\lambda+9$	$\lambda+10$	$\lambda+11$
LOAD r1, (r2)	ΠΕ	ΑΕ	ΕΠ	ΠΜ	AA							
ADD r5, r4	ΠΕ	ΑΕ	ΕΠ	ΠΜ	AA							
ADD r1, r3	ΠΕ	ΑΕ	ΕΠ	ΠΜ	AA							
LOAD r8, (r1)	ΠΕ	ΑΕ	ΕΠ	ΠΜ	AA							
SUB r5, r7	ΠΕ	ΑΕ	ΕΠ	ΠΜ	AA							
SUB r1, r8	ΠΕ	ΑΕ	ΕΠ	ΠΜ	AA							
ADD r1, r9	ΠΕ	ΑΕ	ΕΠ	ΠΜ	AA							
STORE r1, (r5)	ΠΕ	ΑΕ	ΕΠ	ΠΜ	AA							

Επίλυση εξαρτήσεων χωρίς την τεχνική παροχέτευσης με εντολές NOP

Εξαρτήσεις		Επίλυση χωρίς την τεχνική παροχέτευσης																						
		λ	λ+1	λ+2	λ+3	λ+4	λ+5	λ+6	λ+7	λ+8	λ+9	λ+10	λ+11	λ+12	λ+13	λ+14	λ+15	λ+16	λ+17	λ+18	λ+19	λ+20	λ+21	
1. Δομική	LOAD r1, (r2)	ΠΙΕ	ΑΕ	ΕΠ	ΠΜ	ΑΑ																		
2. Δομική	ADD r2, r4		ΠΙΕ	ΑΕ	ΕΠ	ΠΜ	ΑΑ																	
3. ΑΜΕ	ADD r1, r3			ΠΙΕ	ΑΕ	ΕΠ	ΠΜ	ΑΑ																
4. ΑΜΕ	LOAD r8, (r1)				ΠΙΕ	ΑΕ	ΕΠ	ΠΜ	ΑΑ															
5. ΑΜΕ	SUB r5, r7					ΠΙΕ	ΑΕ	ΕΠ	ΠΜ	ΑΑ														
6. ΑΜΕ	SUB r1, r8						ΠΙΕ	ΑΕ	ΕΠ	ΠΜ	ΑΑ													
7. ΑΜΕ	ADD r1, r9							ΠΙΕ	ΑΕ	ΕΠ	ΠΜ	ΑΑ												
	STORE r1, (r5)								ΠΙΕ	ΑΕ	ΕΠ	ΠΜ	ΑΑ											

Σημείωση: Χρησιμεύουν αποφθέγματα 2 παραγόμενα από την επίλυση διαφορετικών σειράς εργασιών, διανομένων σε διάφορα κριτήρια, ποιες σαν τη παροχή εξέλιξης ή το ποσού καθημερινής

Επίλυση εξαρτήσεων χωρίς την τεχνική παροχέτευσης μόνο με παγώματα

	λ	λ+1	λ+2	λ+3	λ+4	λ+5	λ+6	λ+7	λ+8	λ+9	λ+10	λ+11	λ+12	λ+13	λ+14	λ+15	λ+16	λ+17	λ+18	λ+19	λ+20	λ+21
LOAD r1, (r2)	ΠΙΕ	ΑΕ	ΕΠ	ΠΜ	ΑΑ																	
ADD r2, r4		ΠΙΕ	ΑΕ	ΕΠ	ΠΜ	ΑΑ																
ADD r1, r3			ΠΙΕ	ΑΕ	ΕΠ	ΠΜ	ΑΑ															
LOAD r8, (r1)				ΠΙΕ	ΑΕ	ΕΠ	ΠΜ	ΑΑ														
SUB r5, r7					ΠΙΕ	ΑΕ	ΕΠ	ΠΜ	ΑΑ													
SUB r1, r8						ΠΙΕ	ΑΕ	ΕΠ	ΠΜ	ΑΑ												
ADD r1, r9							ΠΙΕ	ΑΕ	ΕΠ	ΠΜ	ΑΑ											
STORE r1, (r5)								ΠΙΕ	ΑΕ	ΕΠ	ΠΜ	ΑΑ										

8 παραγόμενα + 2 = 10 παραγόμενα

Σημείωση: με το κόκκινο X σημειώνονται τα παγώματα για επίλυση δομικών εξαρτήσεων.

β) Επίλυση με την τεχνική παροχέτευσης – Εντοπισμός εξαρτήσεων

	λ	λ+1	λ+2	λ+3	λ+4	λ+5	λ+6	λ+7	λ+8	λ+9	λ+10	λ+11
LOAD r1, (r2)	ΠΙΕ	ΑΕ	ΕΠ	ΠΜ	ΑΑ							
ADD r2, r4		ΠΙΕ	ΑΕ	ΕΠ	ΠΜ	ΑΑ						
ADD r1, r3			ΠΙΕ	ΑΕ	ΕΠ	ΠΜ	ΑΑ					
LOAD r8, (r1)				ΠΙΕ	ΑΕ	ΕΠ	ΠΜ	ΑΑ				
SUB r5, r7					ΠΙΕ	ΑΕ	ΕΠ	ΠΜ	ΑΑ			
SUB r1, r8						ΠΙΕ	ΑΕ	ΕΠ	ΠΜ	ΑΑ		
ADD r1, r9							ΠΙΕ	ΑΕ	ΕΠ	ΠΜ	ΑΑ	
STORE r1, (r5)								ΠΙΕ	ΑΕ	ΕΠ	ΠΜ	ΑΑ

1: Δομική εξάρτηση

2: Δομική εξάρτηση

Επίλυση εξαρτήσεων με την τεχνική παροχέτευσης

	λ	λ+1	λ+2	λ+3	λ+4	λ+5	λ+6	λ+7	λ+8	λ+9	λ+10	λ+11	λ+12	λ+13
LOAD r1, (r2)	ΠΙΕ	ΑΕ	ΕΠ	ΠΜ	ΑΑ									
ADD r2, r4		ΠΙΕ	ΑΕ	ΕΠ	ΠΜ	ΑΑ								
ADD r1, r3			ΠΙΕ	ΑΕ	ΕΠ	ΠΜ	ΑΑ							
LOAD r8, (r1)				X	ΠΙΕ	ΑΕ	ΕΠ	ΠΜ	ΑΑ					
SUB r5, r7						ΠΙΕ	ΑΕ	ΕΠ	ΠΜ	ΑΑ				
SUB r1, r8							ΠΙΕ	ΑΕ	ΕΠ	ΠΜ	ΑΑ			
ADD r1, r9								X	ΠΙΕ	ΑΕ	ΕΠ	ΠΜ	ΑΑ	
STORE r1, (r5)										ΠΙΕ	ΑΕ	ΕΠ	ΠΜ	ΑΑ

Χρειάστηκαν **μόνο δύο παγώματα**. Οι εξαρτήσεις από δεδομένα τύπου AME επιλέχθηκαν απευθείας μέσω του μηχανισμού παροχέτευσης, είτε γιατί ήταν μη-γειτονικές, είτε γιατί η εντολή από την οποία ξεκινάμε δεν είναι εντολή LOAD ή STORE.

Ασκηση από προφορική εξέταση πάνω στις εξαρτήσεις

Χωρίς παροχέτευση, Κοινή μνήμη

AND R1, R12, R13

DIV R10, R12, R3

STORE R10, (R13)

AND R4, R10, R13

SUB R3, R5, R13

MUL R9, R8, R8

Ερώτηση1: Για τον κώδικα να βρεθούν οι εξαρτήσεις που υπάρχουν. **Δεν υπήρχε παροχέτευση, υπήρχε κοινή μνήμη.**

Απάντηση: Δομικές → 3^η – 6^η και AME → 2^η – 3^η και 2^η – 4^η και 3^η – 4^η

Ερώτηση2: Αν έχω στην διάθεσή μου **παροχέτευση και κοινή κρυφή μνήμη** με 1 είσοδο εγγραφής και 1 πόρτα ανάγνωσης ποιες εξαρτήσεις παραμένουν.

Απάντηση: Παραμένουν η δομική και η AME μεταξύ 3^η – 4^η.

Θέμα Ιουνίου 2019 με εξαρτήσεις

Θεωρήστε επεξεργαστή μερικώς επικαλυπτόμενων λειτουργιών (pipeline) με πέντε βαθμίδες ΠΕ, ΑΕ, ΕΠ, ΠΜ και ΑΑ ο οποίος διαθέτει **κοινή κρυφή μνήμη εντολών** και δεδομένων με μια πόρτα ανάγνωσης/εγγραφής και στον οποίο **έχει υλοποιηθεί η τεχνική παροχέτευσης (bypassing)**. Τα προβλήματα από εξαρτήσεις επιλύονται κυκλωματικά (όχι από τον μεταγλωττιστή). Δίνεται το παρακάτω τμήμα προγράμματος.

LOAD r1, (r2)	/ r1 ← M(r2)
ADD r2, r3	/ r2 + r3 → r3
STORE r1, (r5)	/ r1 → M(r5)
LOAD r6, (r7)	/ r6 ← M(r7)
SUB r6, r2	/ r6-r2 → r6
SUB r4, r5	/ r4-r5 → r4

α. Να φτιάξετε το χρονικό διάγραμμα χρήσης των βαθμίδων, να σημειώσετε τις εξαρτήσεις και να αναφέρετε το είδος κάθε εξάρτησης (απλά να αναφέρετε το είδος, δεν χρειάζεται να αιτιολόγηση).

β. Χωρίς να αλλάξετε τη σειρά των εντολών να αναφέρετε την τεχνική που πρέπει να χρησιμοποιηθούν ώστε να μην υπάρχει πρόβλημα στην ορθή εκτέλεση του τμήματος προγράμματος

Λύση

α. Επειδή υποστηρίζεται εκ' των προτέρων η τεχνική της παροχέτευσης, i) ορισμένες εξαρτήσεις που υπάρχουν μεταξύ μη γειτονικών εντολών, επιλύονται απευθείας, όπως π.χ. η εξάρτηση μεταξύ 1^{ης} και 3^{ης} εντολής λόγω του καταχωρητή r1, ii) πρέπει να ελέγχουμε και να σημειώνουμε τις εξαρτήσεις όχι μεταξύ των βημάτων ΑΑ και ΑΕ (είναι λάθος στην περίπτωση αυτή), αλλά μεταξύ των βημάτων **ΠΜ και ΕΠ** (αφού η εξάρτηση μεταξύ των βημάτων ΕΠ και ΕΠ επιλύεται απευθείας). Επίσης, η επίλυση μιας εξάρτησης, μπορεί να οδηγήσει σε άλλη. Χρειάζεται ιδιαίτερη προσοχή στην άσκηση αυτή, διότι η ύπαρξη εντολών Load, Store, σε συνδυασμό με την κοινή κρυφή μνήμη εντολών και δεδομένων, δημιουργεί επιπλέον δομικές εξαρτήσεις.

LOAD r1, (r2)	ΠΕ	ΑΕ	ΕΠ	ΠΜ	ΑΑ	
ADD r2,r3	ΠΕ	ΑΕ	ΕΠ	ΠΜ	ΑΑ	
STORE r1,(r5)	ΠΕ	1	ΑΕ	ΕΠ	ΠΜ	ΑΑ
LOAD r6, (r7)	ΠΕ	2	ΑΕ	ΕΠ	ΠΜ	ΑΑ
SUB r6, r2	ΠΕ	3	ΑΕ	ΕΠ	ΠΜ	ΑΑ
SUB r4, r5	ΠΕ	4	ΑΕ	ΕΠ	ΠΜ	ΑΑ

1, 2: δομικές

3: AME

β. Θα λύσουμε τις προαναφερόμενες εξαρτήσεις **με παγώματα**

LOAD r1, (r2)	ΠΕ	ΑΕ	ΕΠ	ΠΜ	ΑΑ
ADD r2,r3	ΠΕ	ΑΕ	ΕΠ	ΠΜ	ΑΑ
STORE r1,(r5)	ΠΕ	ΑΕ	ΕΠ	ΠΜ	ΑΑ
LOAD r6, (r7)	X	ΠΕ	ΑΕ	ΕΠ	ΠΜ AA

SUB r6, r2

X PE

(AE EP PM AA)

SUB r4, r5

(X PE AE EP PM AA)

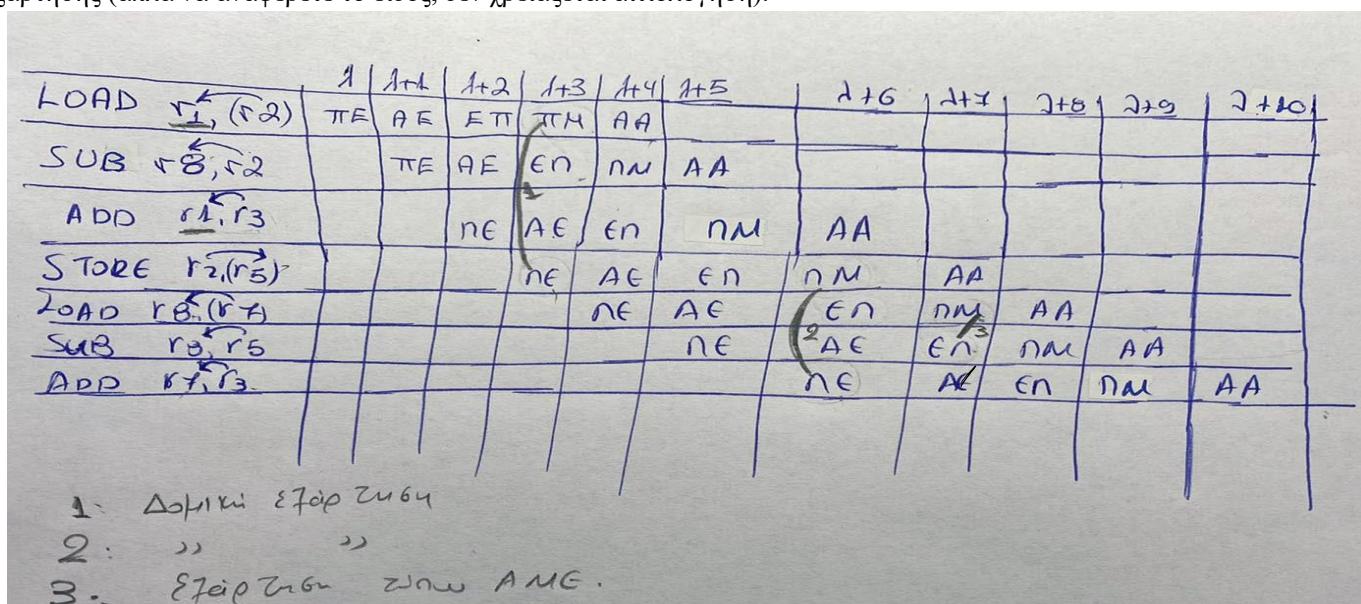
Χρειάστηκαν τρία παγώματα, για να επιλύσουν τις δομικές εξαρτήσεις.

Θέμα Σεπτεμβρίου 2019 με εξαρτήσεις

Θεωρείστε επεξεργαστή μερικώς επικαλυπτόμενων λειτουργιών (pipeline) με πέντε βαθμίδες PE, AE, EP, PM και AA ο οποίος διαθέτει κοινή κρυφή μνήμη εντολών και δεδομένων με μια πόρτα ανάγνωσης/εγγραφής και στον οποίο έχει υλοποιηθεί η τεχνική παροχέτευσης (bypassing). Δίνεται το παρακάτω τμήμα προγράμματος.

LOAD r1, (r2)	/ r1 \leftarrow M(r2)
SUM r8, r2	/ r8 + r2 \rightarrow r8
ADD r1, r3	/ r1 + r3 \rightarrow r1
STORE r2, (r5)	/ r2 \rightarrow M(r5)
LOAD r8, (r7)	/ r8 \leftarrow M(r7)
SUB r8, r5	/ r8 - r5 \rightarrow r8
ADD r7, r3	/ r7 + r3 \rightarrow r7

α. Να φτιάξετε το χρονικό διάγραμμα χρήσης των βαθμίδων, να σημειώσετε τις εξαρτήσεις και να αναφέρετε το είδος κάθε εξάρτησης (απλά να αναφέρετε το είδος, δεν χρειάζεται αιτιολόγηση).



β. Χωρίς να αλλάξετε τη σειρά των εντολών να αναφέρετε την τεχνική ή τις τεχνικές που πρέπει να χρησιμοποιηθούν ώστε να μην υπάρχει πρόβλημα στην ορθή εκτέλεση του τμήματος προγράμματος. Να ξαναφτιάξετε το χρονικό διάγραμμα χρήσης των βαθμίδων εφαρμόζοντας την τεχνική ή τις τεχνικές που προτείνατε.

Λύση

LOAD r1, (r2)	PE	AE	EP	PM	AA							
SUB r8, r2	PE	AE	EP	PM	AA							
ADD r1, r3	PE	AE	EP	PM	AA							
STORE r2, (r5)	X	PE	AE	EP	PM	AA						
LOAD r8, (r7)		PE	AE	EP	PM	AA						
SUB r8, r5		PE	AE'	EP	PM	AA						
ADD r7, r3		X	PE	AE	EP	PM	AA					

Το κάθε πάγωμα (ή κλείδωμα) που έχει χρησιμοποιηθεί, επιλύει μια δομική εξάρτηση. Το AE' επιλύει την εξάρτηση από δεδομένα τύπου AME που υπάρχει μεταξύ των εντολών LOAD κι SUB εξαιτίας του καταχωρητή R8.

Θέμα Σεπτεμβρίου 2020 με εξαρτήσεις

Θεωρείστε επεξεργαστή μερικώς επικαλυπτόμενων εντολών ο οποίος διαθέτει **κοινή κρυφή μνήμη εντολών και δεδομένων** με μια κοινή πόρτα ανάγνωσης και εγγραφής και **υλοποιεί την τεχνική της παροχέτευσης (bypassing)**. Θεωρήστε επίσης ότι σε ένα κύκλο μπορεί να γίνει εγγραφή σε κάποιο καταχωρητή και στον ίδιο κύκλο ανάγνωση αυτού που γράφτηκε. Δίνεται το τμήμα προγράμματος που ακολουθεί. Για κάθε είδος εξάρτησης να δηλώσετε τα ζεύγη των εξαρτημένων εντολών για τα οποία εάν δεν υπάρξει πρόνοια θα προκύψει πρόβλημα (hazard). Οι απαντήσεις σας να έχουν την ακόλουθη μορφή πχ. διαδικασιακές εξαρτήσεις: E2-E4+E3-E6 το οποίο σημαίνει ότι υπάρχουν δύο διαδικασιακές εξαρτήσεις που πρέπει να αντιμετωπιστούν για να μην υπάρξει πρόβλημα, μια μεταξύ των εντολών E2 και E4 και μια μεταξύ των εντολών E3 και E6. Εάν δεν υπάρχει εξάρτηση που να δημιουργεί πρόβλημα τότε γράφουμε το μηδέν πχ. 0. Προσοχή: στις αριθμητικές και τις λογικές εντολές το αποτέλεσμα γράφεται στο πρώτο τελούμενο, π.χ. ADD R4, R2, R3 σημαίνει $R4 \leftarrow R2 + R3$.

**E1 AND R1, R2, R3, E2 MUL R12, R14, R11, E3 ADD R12, R1, R12, E4 STORE R10, (R12), E5 AND R4, R12, R7
E6 SUB R12, R4, R13, E7 LOAD R1 (R10), E8 MUL R9, R1, R8**

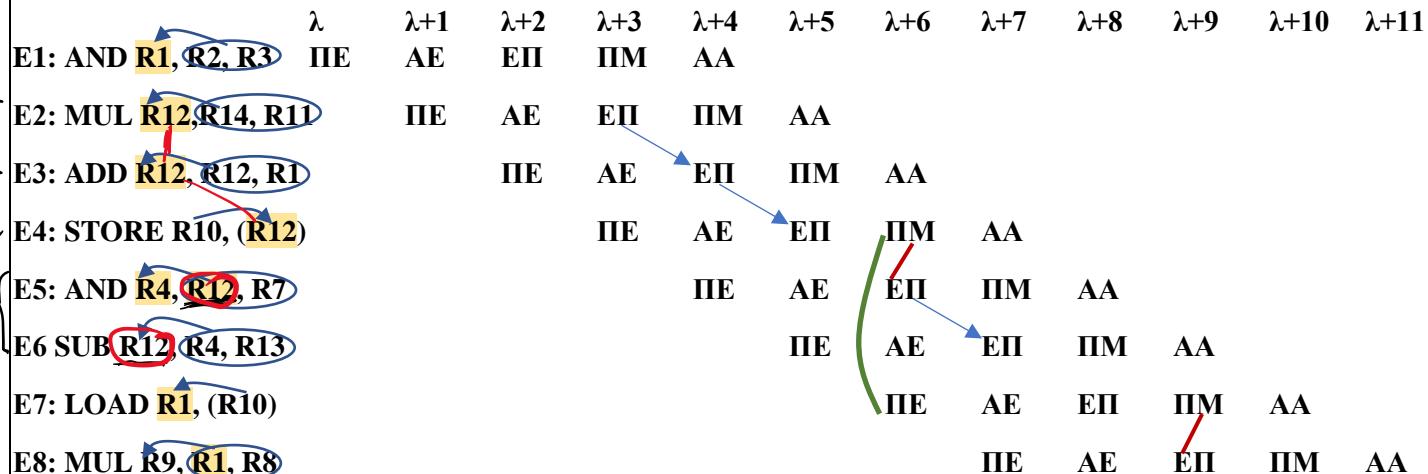
ΔΟΜΙΚΕΣ ΕΞΑΡΤΗΣΕΙΣ

ΕΞΑΡΤΗΣΕΙΣ ΑΝΑΓΝΩΣΗΣ ΜΕΤΑ ΑΠΟ ΕΓΓΡΑΦΗ

ΕΞΑΡΤΗΣΕΙΣ ΕΓΓΡΑΦΗΣ ΜΕΤΑ ΑΠΟ ΑΝΑΓΝΩΣΗ

ΕΞΑΡΤΗΣΕΙΣ ΕΓΓΡΑΦΗΣ ΜΕΤΑ ΑΠΟ ΕΓΓΡΑΦΗ

Προσοχή! Επειδή η εκφώνηση αναφέρει ότι ο επεξεργαστής υλοποιεί την **τεχνική της παροχέτευσης** από την αρχή, αυτό θα πρέπει να το λάβουμε υπόψη μας στον εντοπισμό των εξαρτήσεων



Δομικές εξαρτήσεις: 1

E4-E7

Εξαρτήσεις AME: 2

E4-E5+E7-E8 (οι υπόλοιπες έχουν επιλυθεί)

Εξαρτήσεις EME: εμφανίζονται μεταξύ των εντολών E2 και E3. Όμως, λόγω της τεχνικής της παροχέτευσης, επιλύονται αυτόματα (η τεχνική της παροχέτευσης επιλύει αυτόματα όλες τις εξαρτήσεις δεδομένων).

Εξαρτήσεις EMA: εμφανίζονται μεταξύ των εντολών E5 και E6. Όμως, λόγω της τεχνικής της παροχέτευσης, επιλύεται.

Παρατήρηση: όταν χρησιμοποιείται η τεχνική της παροχέτευσης, τότε οι εξαρτήσεις θα υπάρχουν (αν υπάρχουν) μόνο μεταξύ γειτονικών εντολών και πιο συγκεκριμένα μεταξύ των βημάτων PM (LOAD, STORE) και EP (οποιαδήποτε εντολής). Θα πρέπει πάντα το EP να είναι πιο δεξιά μια τουλάχιστον θέση σε σχέση με το PM. Όταν δεν χρησιμοποιείται η τεχνική της παροχέτευσης, τότε οι εξαρτήσεις μπορούν να υπάρχουν και μεταξύ μη – γειτονικών εντολών και πιο συγκεκριμένα θα υπάρχουν (αν υπάρχουν) μεταξύ των βημάτων AA (οποιαδήποτε εντολής) και AE (οποιαδήποτε εντολής). Αναφορικά με τις δομικές εξαρτήσεις, κοιτάμε τα βήματα PM (LOAD, STORE) και PE (οποιαδήποτε εντολής)

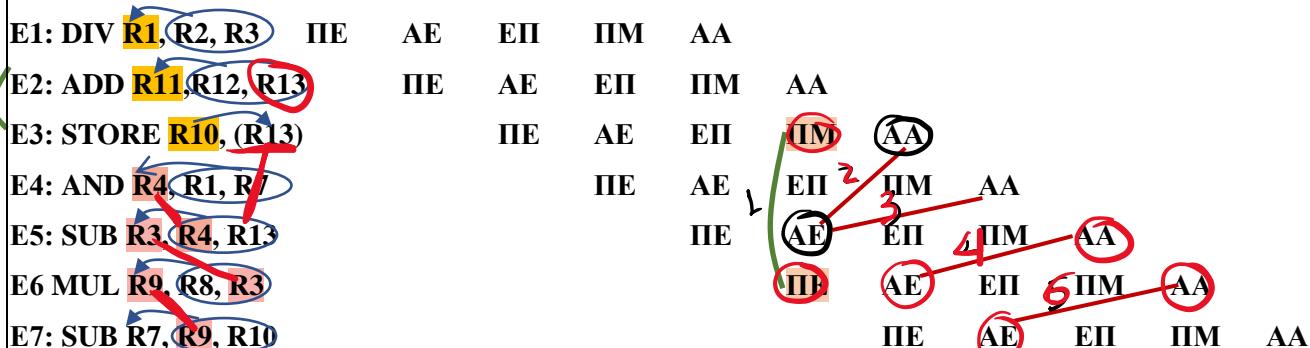
Θέμα Σεπτεμβρίου 2020 με εξαρτήσεις

Θεωρήστε επεξεργαστή μερικώς επικαλυπτόμενων εντολών ο οποίος διαθέτει **κοινή κρυφή μνήμη εντολών και δεδομένων** με μια πόρτα ανάγνωσης και μια άλλη πόρτα εγγραφής και **δεν υλοποιεί την τεχνική της παροχέτευσης (bypassing)**. Θεωρήστε επίσης ότι σε ένα κύκλο μπορεί να γίνει εγγραφή σε κάποιο καταχωρητή και στον ίδιο κύκλο ανάγνωση αυτού που γράφτηκε. Δίνεται το τμήμα προγράμματος που ακολουθεί. Για κάθε είδος εξάρτησης να δηλώσετε τα ζεύγη των εξαρτημένων εντολών για

τα οποία εάν δεν υπάρξει πρόνοια θα προκύψει πρόβλημα (hazard). Οι απαντήσεις σας να έχουν την ακόλουθη μορφή πχ. διαδικασιακή εξάρτηση [E2-E4+E3-E6].

E1 DIV R1, R2, R3 E2 ADD R11, R12, R13 E3 STORE R10, (R13) E4 AND R4, R1 R7 E5 SUB R3, R4, R13 E6 MUL R9, R8, R3 E7 SUB R7, R9, R10. Ζητάμε: δομικές εξαρτήσεις, εξαρτήσεις ανάγνωσης μετά από εγγραφή, εξαρτήσεις εγγραφής μετά από ανάγνωση, εξαρτήσεις εγγραφής μετά από εγγραφή

Λύση



Δομικές εξαρτήσεις: 1 (όταν έχουμε κοινή κρυφή μνήμη εντολών και δεδομένων και ταυτίζεται το βήμα ΠΜ εντολής LOAD ή STORE με το ΠΕ οποιασδήποτε άλλης εντολής) **E3 – E6**

Εξαρτήσεις AME: 4, E3 – E5 + E4 – E5 + E5 – E6 + E6 – E7

Σημείωση: αν χρησιμοποιούσαμε την τεχνική της παροχέτευσης, τότε θα είχαμε μόνο **μια δομική εξάρτηση**, επειδή οι εξαρτήσεις δεδομένων τύπου AME θα επιλύονταν αυτόματα στην περίπτωση αυτή.

Εξαρτήσεις EME: 0

Εξαρτήσεις EMA: E2-E3 (λόγω καταχωρητή R13)

ΣΜΣ
ΕΜΑ

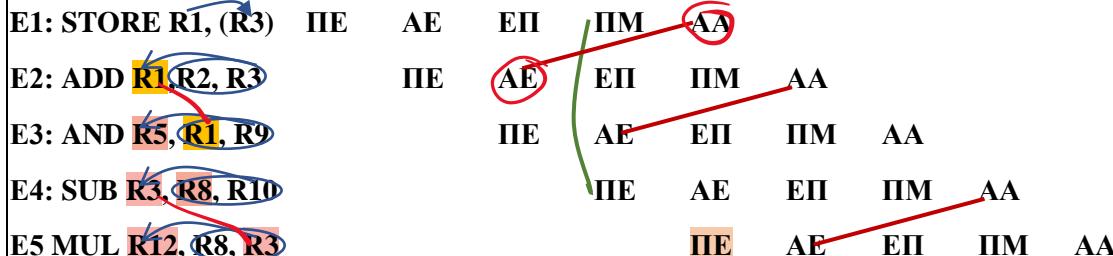
Θέμα Σεπτεμβρίου 2020 με εξαρτήσεις

Θεωρήστε επεξεργαστή μερικώς επικαλυπτόμενων εντολών ο οποίος διαθέτει **κοινή κρυφή μνήμη εντολών και δεδομένων** με μια κοινή πόρτα ανάγνωσης και εγγραφής και **δεν υλοποιεί την τεχνική της παροχέτευσης** (bypassing). Θεωρήστε επίσης ότι σε ένα κύκλο μπορεί να γίνει εγγραφή σε κάποιο καταχωρητή και στον ίδιο κύκλο ανάγνωση αυτού που γράφτηκε. Δίνεται το τμήμα προγράμματος που ακολουθεί. Για κάθε είδος εξάρτησης να δηλώσετε τα ζεύγη των εξαρτημένων εντολών για τα οποία εάν δεν υπάρξει πρόνοια θα προκύψει πρόβλημα (hazard). Οι απαντήσεις σας να έχουν την ακόλουθη μορφή πχ. διαδικασιακή εξάρτηση [E2-E4+E3-E6] το οποίο σημαίνει ότι υπάρχουν δύο διαδικασιακές εξαρτήσεις που πρέπει να αντιμετωπιστούν για να μην υπάρξει πρόβλημα, μια μεταξύ των εντολών E2 και E4 και μια μεταξύ των εντολών E3 και E6. Εάν δεν υπάρχει εξάρτηση που να δημιουργεί πρόβλημα τότε ανάμεσα στις αγκύλες γράφουμε το μηδέν πχ [0].

E1 STORE R1, (R3) E2 ADD R1, R2, R3 E3 AND R5, R1 R9 E4 SUB R3, R8, R10 E5 MUL R12, R8, R3 ΔΟΜΙΚΕΣ ΕΞΑΡΤΗΣΕΙΣ **E1-E4+E2-E5**

ΕΞΑΡΤΗΣΕΙΣ ΑΝΑΓΝΩΣΗΣ ΜΕΤΑ ΑΠΟ ΕΓΓΡΑΦΗ **E2-E4** ΕΞΑΡΤΗΣΕΙΣ ΕΓΓΡΑΦΗΣ ΜΕΤΑ ΑΠΟ ΑΝΑΓΝΩΣΗ **0** ΕΞΑΡΤΗΣΕΙΣ ΕΓΓΡΑΦΗΣ ΜΕΤΑ ΑΠΟ ΕΓΓΡΑΦΗ **0**.

Λύση



Δομικές εξαρτήσεις: 1 (όταν έχουμε κοινή κρυφή μνήμη εντολών και δεδομένων και ταυτίζεται το βήμα ΠΜ εντολής LOAD ή STORE με το ΠΕ οποιασδήποτε άλλης εντολής), μεταξύ εντολών E1 – E4

Εξαρτήσεις AME: 3, μεταξύ εντολών E1-E2 + E2-E3 + E4-E5

EME: 0, EMA: E1-E2 (λόγω καταχωρητή R1)

Θέμα Σεπτεμβρίου 2020 με εξαρτήσεις

Θεωρείστε επεξεργαστή μερικώς επικαλυπτόμενων εντολών ο οποίος διαθέτει **κοινή κρυφή μνήμη εντολών και δεδομένων** με μια πόρτα ανάγνωσης και μια άλλη πόρτα εγγραφής και **δεν υλοποιεί την τεχνική της παροχέτευσης** (bypassing). Θεωρήστε επίσης ότι σε ένα κύκλο μπορεί να γίνει εγγραφή σε κάποιο καταχωρητή και στον ίδιο κύκλο ανάγνωση αυτού που γράφτηκε. Δίνεται το τμήμα προγράμματος που ακολουθεί. Για κάθε είδος εξάρτησης να δηλώσετε τα ζεύγη των εξαρτημένων εντολών για τα οποία εάν δεν υπάρξει πρόνοια θα προκύψει πρόβλημα (hazard). Οι απαντήσεις σας να έχουν την ακόλουθη μορφή πχ. διαδικασιακές εξαρτήσεις E2-E4+E3-E6 το οποίο σημαίνει ότι υπάρχουν δύο διαδικασιακές εξαρτήσεις που πρέπει να αντιμετωπιστούν για να μην υπάρξει πρόβλημα, μια μεταξύ των εντολών E2 και E4 και μια μεταξύ των εντολών E3 και E6. Εάν δεν υπάρχει εξάρτηση που να δημιουργεί πρόβλημα τότε γράφουμε το μηδέν πχ. 0. Προσοχή: στις αριθμητικές και τις λογικές εντολές το αποτέλεσμα γράφεται στο πρώτο τελούμενο, πχ ADD R4, R2, R3 σημαίνει R4 < - R2+R3.

E1 STORE R1, (R3)	ΠΕ	ΑΕ	ΕΠ	ΠΜ	ΑΑ							
E2 ADD R1,(R2, R3)	ΠΕ	ΑΕ	ΕΠ	ΠΜ	ΑΑ							
E3 AND R5,(R1 R8)	ΠΕ	ΑΕ	ΕΠ	ΠΜ	ΑΑ							
E4 SUB R5,(R8, R10)		ΠΕ	ΑΕ	ΕΠ	ΠΜ	ΑΑ						
E5 MUL R9, R7, R3		ΠΕ	ΑΕ	ΕΠ	ΠΜ	ΑΑ						
E6 XOR R12,(R11, R1)		ΠΕ	ΑΕ	ΕΠ	ΠΜ	ΑΑ						
E7 NOR R11, R8, R2		ΠΕ	ΑΕ	ΕΠ	ΠΜ	ΑΑ						

ΔΟΜΙΚΕΣ ΕΞΑΡΤΗΣΕΙΣ [E1-E4] καθώς το ΠΜ της store συμπίπτει με το ΠΕ της sub

ΕΞΑΡΤΗΣΕΙΣ ΑΝΑΓΝΩΣΗΣ ΜΕΤΑ ΑΠΟ ΕΓΓΡΑΦΗ [E1-E2 + E2-E3]

ΕΞΑΡΤΗΣΕΙΣ ΕΓΓΡΑΦΗΣ ΜΕΤΑ ΑΠΟ ΑΝΑΓΝΩΣΗ [0].

ΕΞΑΡΤΗΣΕΙΣ ΕΓΓΡΑΦΗΣ ΜΕΤΑ ΑΠΟ ΕΓΓΡΑΦΗ [0].

Λύση

Εκτός από τις ήδη σημειωμένες εξαρτήσεις, υπάρχει άλλη μια εξάρτηση **EME μεταξύ των εντολών E3-E4**, (επειδή πρόκειται για διαδοχικές εντολές οι οποίες αποθηκεύονται στον ίδιο καταχωρητή (R5) και εξάρτηση **EMA μεταξύ των εντολών E1-E3 + E6-E7** (επειδή πρόκειται για διαδοχικές εντολές εκ' των οποίων η δεύτερη αποθηκεύεται στον R11, ενώ η πρώτη έχει διαβάσει τον R11). Αν είχαμε **ξεχωριστή** κρυφή μνήμη και όχι κοινή, τότε **δεν** θα υπήρχαν εντολές EME και EMA.

Θέματα Ιανουαρίου 2020

Θέμα 1

Θεωρείστε επεξεργαστή Μ μερικώς επικαλυπτόμενων λειτουργιών με **μ** βαθμίδες και **επεξεργαστή N** που **δεν** χρησιμοποιεί την τεχνική των μερικώς επικαλυπτόμενων λειτουργιών. Να σημειώσετε ποιες από τις επόμενες προτάσεις είναι σωστές. Για κάθε λάθος επιλογή θα ακυρώνεται μια σωστή.

1. Ο χρόνος εκτέλεσης μιας εντολής στον επεξεργαστή M είναι μικρότερος από το χρόνο εκτέλεσης στον N.
2. Ο χρόνος εκτέλεσης μιας εντολής στον επεξεργαστή M είναι **μ** φορές μικρότερος από το χρόνο εκτέλεσης στον N.
3. Ο χρόνος εκτέλεσης μιας εντολής στον επεξεργαστή N είναι **μικρότερος** από το χρόνο εκτέλεσης στον M.
4. Ο ρυθμός ολοκλήρωσης εντολών στον επεξεργαστή M είναι μικρότερος από το ρυθμό ολοκλήρωσης εντολών στον επεξεργαστή N.
5. Ο χρόνος εκτέλεσης μιας εντολής και στους δύο επεξεργαστές είναι ο ίδιος.
6. Ο ρυθμός ολοκλήρωσης εντολών στους επεξεργαστές M και N είναι ο ίδιος.
7. Ο ρυθμός ολοκλήρωσης εντολών στον επεξεργαστή N είναι μικρότερος από το ρυθμό ολοκλήρωσης εντολών στον επεξεργαστή M.
8. Ο ρυθμός ολοκλήρωσης εντολών στον επεξεργαστή N είναι μι φορές μικρότερος από το ρυθμό ολοκλήρωσης εντολών στον επεξεργαστή M. (θα ήταν σωστό όταν είχαμε ιδανικό ΜΕΛ)

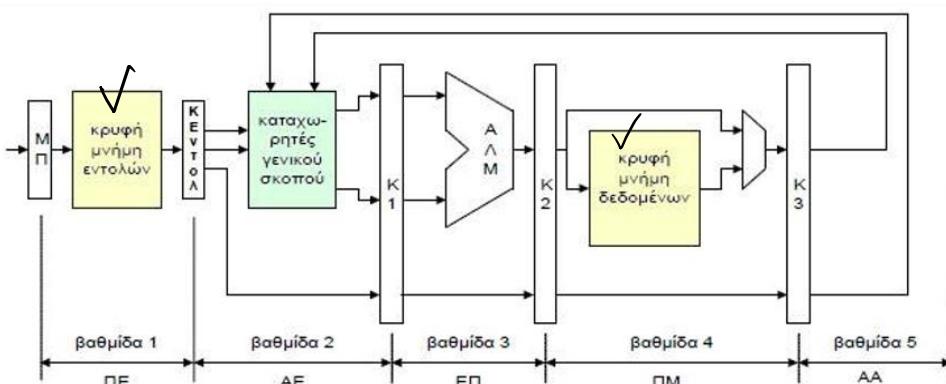
Λύση

Σωστά 3, 7

Αυτό που ισχύει είναι ότι στον επεξεργαστή που χρησιμοποιεί την τεχνική των επικαλυπτόμενων λειτουργιών, ο χρόνος εκτέλεσης μιας εντολής είναι μεγαλύτερος σε σύγκριση με ένα επεξεργαστή που δεν χρησιμοποιεί την τεχνική αυτή (μειονέκτημα), ενώ ο ρυθμός ολοκλήρωσης εντολών είναι και αυτός μεγαλύτερος σε σύγκριση με ένα επεξεργαστή που δεν χρησιμοποιεί την τεχνική αυτή (πλεονέκτημα).

Θέμα 2

Ο επόμενος επεξεργαστής μερικώς επικαλυπτόμενων λειτουργιών έχει σχεδιαστεί για την εκτέλεση των εντολών: LOAD r1, (r2) [$r1 \leftarrow M(r2)$], STORE r1, (r2) [$r1 \rightarrow M(r2)$], ADD r1, r2, r3 [$r1+r2 \rightarrow r3$], AND r1, r2, r3 [$r1 \cdot r2 \rightarrow r3$], SUB r1, r2, r3 [$r1-r2 \rightarrow r3$] και BRE r1, r2, d [εάν $r1-r2 = 0$ τότε $M_P = M_P + d$], στις αγκύλες είναι η επεξήγηση της εντολής.



Να βάλετε σε κύκλο τις σωστές προτάσεις. Για κάθε λάθος επιλογή θα ακυρώνεται μια σωστή.

1. Δεν είναι δυνατόν να προκύψει πρόβλημα λόγω εξάρτησης τύπου Ανάγνωση μετά από Εγγραφή.
2. Σε περίπτωση που δεν λάβουμε τα αναγκαία μέτρα είναι δυνατόν να προκύψει πρόβλημα λόγω εξάρτησης τύπου Εγγραφή μετά από Ανάγνωση.
3. Δεν είναι δυνατόν να προκύψει πρόβλημα λόγω εξάρτησης τύπου Εγγραφή μετά από Ανάγνωση.
4. Σε περίπτωση που δεν λάβουμε τα αναγκαία μέτρα είναι δυνατόν να προκύψει πρόβλημα λόγω εξάρτησης τύπου Ανάγνωση μετά από Εγγραφή.
5. Σε περίπτωση που δεν λάβουμε τα αναγκαία μέτρα είναι δυνατόν να προκύψει πρόβλημα λόγω εξάρτησης τύπου Εγγραφή μετά από Εγγραφή.
6. Δεν είναι δυνατόν να προκύψει πρόβλημα λόγω διαδικασιακών εξαρτήσεων. (Θα ίσχυε μόνο αν η εκφώνηση ανέφερε ότι δεν υπάρχουν εντολές διακλάδωσης, ενώ στην προκειμένη περίπτωση υπάρχει μεταξύ άλλων και η εντολή BRE).
7. Σε περίπτωση που δεν λάβουμε τα αναγκαία μέτρα είναι δυνατόν να προκύψει πρόβλημα λόγω δομικών εξαρτήσεων (δεν μπορεί να συμβεί δομική εξάρτηση, λόγω των δύο κρυφών μνημών που υπάρχουν).
8. Δεν είναι δυνατόν να προκύψει πρόβλημα λόγω εξάρτησης τύπου Εγγραφή μετά από Εγγραφή.
9. Σε περίπτωση που δεν λάβουμε τα αναγκαία μέτρα είναι δυνατόν να προκύψει πρόβλημα λόγω διαδικασιακών εξαρτήσεων.
10. Δεν είναι δυνατόν να προκύψει πρόβλημα λόγω δομικών εξαρτήσεων.

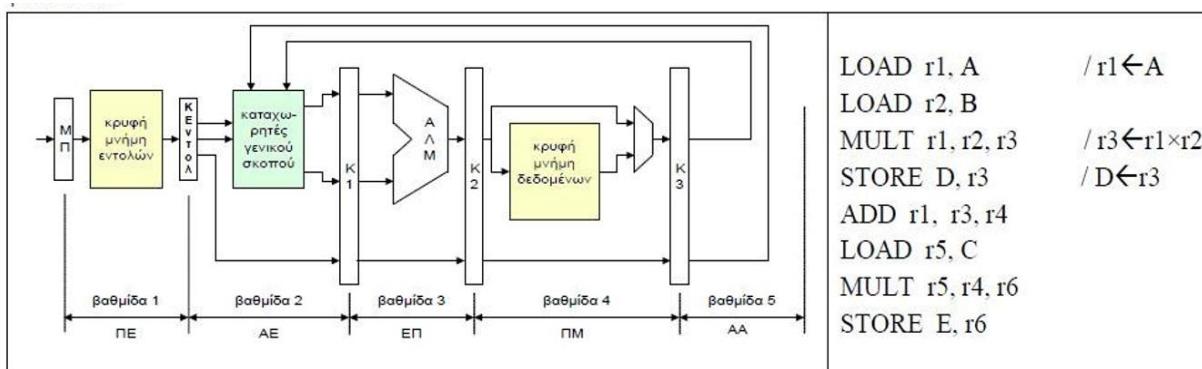
Λύση

Σωστά 3, 4, 8, 9, 10

Στον επεξεργαστή του σχήματος δεν μπορούν να προκύψουν δομικές εξαρτήσεις, λόγω της ύπαρξης δύο κρυφών μνημών. Επίσης, μπορούν να προκύψουν διαδικασιακές εξαρτήσεις, λόγω ύπαρξης της εντολής διακλάδωσης BRE. Τέλος, στο συγκεκριμένο επεξεργαστή δεν μπορεί να προκύψει εξάρτηση από δεδομένα τύπου EME και EMA, παρά μόνο AME.

Θέμα 3

Θεωρήστε το μηχανισμό μερικώς επικαλυπτόμενων λειτουργιών του επόμενου σχήματος ο οποίος **θεωρούμε ότι υλοποιεί την τεχνική παροχέτευσης** (bypassing ή forwarding), τα κυκλώματα για το σκοπό αυτό δεν φαίνονται.



Εάν το ανωτέρω τμήμα προγράμματος εκτελείται στον ανωτέρω μηχανισμό

- Να φτιάξετε το χρονικό διάγραμμα εκτέλεσης των εντολών, να σημειώσετε τις εξαρτήσεις και να αναφέρετε το είδος τους.
- Να λύσετε τις εξαρτήσεις χρησιμοποιώντας εντολές NOP.

Λύση

εντολή

Περίοδος Σήματος Χρονισμού

LOAD r1, A

	λ	$\lambda+1$	$\lambda+2$	$\lambda+3$	$\lambda+4$	$\lambda+5$	$\lambda+6$	$\lambda+7$	$\lambda+8$	$\lambda+9$	$\lambda+10$	$\lambda+11$	
LOAD r1, A	ΠΕ	ΑΕ	ΕΠ	ΠΜ	ΑΑ								LOAD r2, B
LOAD r2, B	ΠΕ	ΑΕ	ΕΠ	ΠΜ	ΑΑ								NOP
MULT r1, r2, r3		ΠΕ	ΑΕ	ΕΠ	ΠΜ	ΑΑ							MULT r1, r2, r3
STORE D, r3		ΠΕ	ΑΕ	ΕΠ	ΠΜ	ΑΑ							STORE D, r3
ADD r1, r3, r4		ΠΕ	ΑΕ	ΕΠ	ΠΜ	ΑΑ							ADD r1, r3, r4
LOAD r5, C		ΠΕ	ΑΕ	ΕΠ	ΠΜ	ΑΑ							LOAD r5, C
MULT r5, r4, r6		ΠΕ	ΑΕ	ΕΠ	ΠΜ	ΑΑ							NOP
STORE E, r6		ΠΕ	ΑΕ	ΕΠ	ΠΜ	ΑΑ							MULT r5, r4, r6
													STORE E, r6

Λόγω ύπαρξης των **δύο κρυφών μνημών**, αποκλείεται να υπάρχουν **δομικές εξαρτήσεις** και εξαρτήσεις από δεδομένα τύπου **EME και EMA**. Οι υπόλοιπες εξαρτήσεις από δεδομένα τύπου AME που υπάρχουν επιλύνονται απευθείας, διότι από την αρχή χρησιμοποιείται η τεχνική της παροχέτευσης. Για παράδειγμα, η εξάρτηση μεταξύ της πρώτης LOAD και της MULT r1, r2, r3 παύει να υπάρχει διότι έχουμε εντολές μη-διαδοχικές. Επίσης, η εξάρτηση μεταξύ της MULT r1, r2, r3 και της STORE D, r3 επιλύνεται επίσης αυτόματα, διότι ξεκινάμε από το βήμα ΕΠ (λόγω εντολής MULT) και καταλήγουμε στο ΕΠ της STORE D, r3, που είναι πιο δεξιά κ.ο.κ.

Θέμα με εξαρτήσεις

Θεωρείστε το μηχανισμό μερικώς επικαλυπτόμενων λειτουργιών του σχήματος 4.4 του βιβλίου και ότι έχει υλοποιηθεί η τεχνική παροχέτευσης για την επίλυση των εξαρτήσεων δεδομένων. Θέλουμε να εκτελέσουμε τις επόμενες κάποιας γλώσσας προγραμματισμού υψηλού επιπέδου:

```

LOAD r1, A
LOAD r2, B
MULT r1, r2, r3 // r1 x r2 → r3
STORE D, r3
ADD r1, r3, r4
LOAD r5, C
MULT r5, r4, r6
STORE E, r6

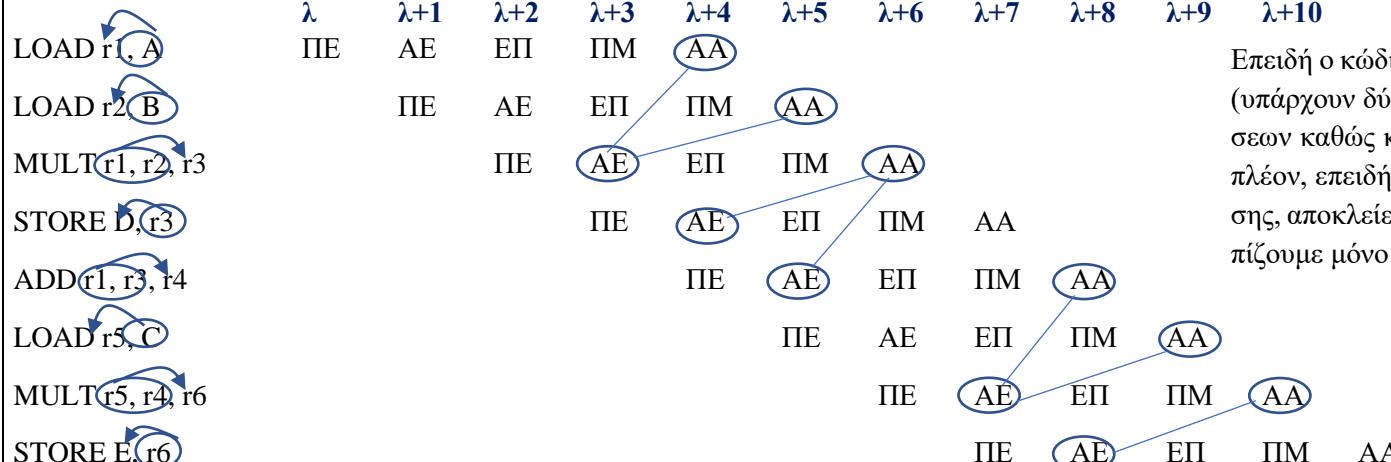
```

Να εντοπιστούν και να επιλυθούν οι εξαρτήσεις

- Χωρίς παροχέτευση
- Με παροχέτευση

Λύση

a) Εντοπισμός εξαρτήσεων (χωρίς παροχέτευση)



Επειδή ο κώδικας της εκφύνησης εκτελείται στο ΜΕΛ του σχήματος 4.4 (υπάρχουν δύο κρυφές μνήμες), αποκλείεται η ύπαρξη δομικών εξαρτήσεων καθώς και εξαρτήσεων από δεδομένα τύπου EME και EMA. Επιπλέον, επειδή στο δοθέντα κώδικα δεν εμφανίζονται εντολές διακλάδωσης, αποκλείεται η ύπαρξη διαδικασιακών εξαρτήσεων. Επομένως, εντοπίζουμε μόνο εξαρτήσεις από δεδομένα τύπου AME.

Υπάρχουν 7 εξαρτήσεις από δεδομένα, τύπου AME. Δεν υπάρχουν δομικές εξαρτήσεις.

a1) Επίλυση εξαρτήσεων με εντολές NOP

	λ	$\lambda+1$	$\lambda+2$	$\lambda+3$	$\lambda+4$	$\lambda+5$	$\lambda+6$	$\lambda+7$	$\lambda+8$	$\lambda+9$	$\lambda+10$	$\lambda+11$	$\lambda+12$	$\lambda+13$	$\lambda+14$	$\lambda+15$	$\lambda+16$	$\lambda+17$	$\lambda+18$
LOAD r1, A	ΠΕ	AE	EΠ	ΠΜ	AA														
LOAD r2, B	ΠΕ	AE	EΠ	ΠΜ	AA														
NOP	ΠΕ	AE	EΠ	ΠΜ	AA														
NOP	ΠΕ	AE	EΠ	ΠΜ	AA														
MULT r1, r2, r3		ΠΕ	AE	EΠ	ΠΜ	AA													
NOP		ΠΕ	AE	EΠ	ΠΜ	AA													
NOP		ΠΕ	AE	EΠ	ΠΜ	AA													
STORE D, r3		ΠΕ	AE	EΠ	ΠΜ	AA													
ADD r1, r3, r4		ΠΕ	AE	EΠ	ΠΜ	AA													
LOAD r5, C		ΠΕ	AE	EΠ	ΠΜ	AA													
NOP		ΠΕ	AE	EΠ	ΠΜ	AA													
NOP		ΠΕ	AE	EΠ	ΠΜ	AA													
MULT r5, r4, r6		ΠΕ	AE	EΠ	ΠΜ	AA													
NOP		ΠΕ	AE	EΠ	ΠΜ	AA													
NOP		ΠΕ	AE	EΠ	ΠΜ	AA													
STORE E, r6		ΠΕ	AE	EΠ	ΠΜ	AA													

a2) Επίλυση εξαρτήσεων με παγώματα

	λ	$\lambda+1$	$\lambda+2$	$\lambda+3$	$\lambda+4$	$\lambda+5$	$\lambda+6$	$\lambda+7$	$\lambda+8$	$\lambda+9$	$\lambda+10$	$\lambda+11$	$\lambda+12$	$\lambda+13$	$\lambda+14$	$\lambda+15$	$\lambda+16$	$\lambda+17$	$\lambda+18$
LOAD r1, A	ΠΕ	ΑΕ	ΕΠ	ΠΜ	AA														
LOAD r2, B		ΠΕ	ΑΕ	ΕΠ	ΠΜ	AA													
MULT r1, r2, r3		ΠΕ	AE'	X	AE	ΕΠ	ΠΜ	AA											
STORE D, r3						ΠΕ	AE'	X	AE	ΕΠ	ΠΜ	AA							
ADD r1, r3, r4									ΠΕ	ΑΕ	ΕΠ	ΠΜ	AA						
LOAD r5, C										ΠΕ	ΑΕ	ΕΠ	ΠΜ	AA					
MULT r5, r4, r6										ΠΕ	AE'	X	AE	ΕΠ	ΠΜ	AA			
STORE E, r6											ΠΕ	AE'	X	AE	ΕΠ	ΠΜ	AA		

Σημειώσεις:

- 1) Κάθε φορά η νέα εντολή ξεκινά κάτω από το AE της προηγούμενης.
- 2) Αν οι εντολές που έχουν εξάρτηση από δεδομένα τύπου AME είναι γειτονικές, τότε θα χρειαστούμε δύο εντολές NOP ανάμεσά τους ή εναλλακτικά δύο παγώματα, που συμβολίζονται με AE' και X.
- 3) Αν οι εντολές που έχουν εξάρτηση από δεδομένα τύπου AME είναι μη – γειτονικές (στην περίπτωση αυτή μεταξύ των εντολών που έχουν εξάρτηση μεσολαβεί μια εντολή), τότε θα χρειαστούμε μια εντολή NOP ανάμεσά τους ή εναλλακτικά ένα πάγωμα, που συμβολίζονται με AE'.
- 4) Στη συγκεκριμένη άσκηση, επειδή υπάρχουν δύο είδη κρυφής μνήμης, δεν υπάρχουν δομικές εξαρτήσεις.

b) Αν λάβουμε υπόψη την εκφώνηση, ότι δηλαδή υποστηρίζεται εξαρχής η τεχνική της παροχέτευσης, τότε κάποιες εξαρτήσεις επιλύονται αυτόματα από μόνες τους. Επιπλέον δεν υπάρχουν εξαρτήσεις μεταξύ εντολών μη – γειτονικών, διότι και αυτές επιλύονται αυτόματα.

Εντοπισμός εξαρτήσεων (με παροχέτευση)

	λ	$\lambda+1$	$\lambda+2$	$\lambda+3$	$\lambda+4$	$\lambda+5$	$\lambda+6$	$\lambda+7$	$\lambda+8$	$\lambda+9$	$\lambda+10$
LOAD r1, A	ΠΕ	ΑΕ	ΕΠ	ΠΜ	AA						
LOAD r2, B		ΠΕ	ΑΕ	ΕΠ	ΠΜ	AA					
MULT r1, r2, r3		ΠΕ	ΑΕ	ΕΠ	ΠΜ	AA					
STORE D, r3			ΠΕ	ΑΕ	ΕΠ	ΠΜ	AA				
ADD r1, r3, r4				ΠΕ	ΑΕ	ΕΠ	ΠΜ	AA			
LOAD r5, C					ΠΕ	ΑΕ	ΕΠ	ΠΜ	AA		
MULT r5, r4, r6						ΠΕ	ΑΕ	ΕΠ	ΠΜ	AA	
STORE E, r6							ΠΕ	ΑΕ	ΕΠ	ΠΜ	AA

Υπάρχουν 2 εξαρτήσεις από δεδομένα, τύπου AME. Όλες οι υπόλοιπες επιλύθηκαν αυτόματα επειδή χρησιμοποιείται από την αρχή η τεχνική επίλυσης εξαρτήσεων από δεδομένα τύπου AME με όνομα παροχέτευση.

b1) Επίλυση εξαρτήσεων με εντολές NOP

	λ	$\lambda+1$	$\lambda+2$	$\lambda+3$	$\lambda+4$	$\lambda+5$	$\lambda+6$	$\lambda+7$	$\lambda+8$	$\lambda+9$	$\lambda+10$
LOAD r1, A	ΠΕ	ΑΕ	ΕΠ	ΠΜ	ΑΑ						
LOAD r2, B	ΠΕ	ΑΕ	ΕΠ	ΠΜ	ΑΑ						
NOP		ΠΕ	ΑΕ	ΕΠ	ΠΜ	ΑΑ					
MULT r1, r2, r3		ΠΕ	ΑΕ	ΕΠ	ΠΜ	ΑΑ					
STORE D, r3		ΠΕ	ΑΕ	ΕΠ	ΠΜ	ΑΑ					
ADD r1, r3, r4		ΠΕ	ΑΕ	ΕΠ	ΠΜ	ΑΑ					
LOAD r5, C		ΠΕ	ΑΕ	ΕΠ	ΠΜ	ΑΑ					
NOP		ΠΕ	ΑΕ	ΕΠ	ΠΜ	ΑΑ					
MULT r5, r4, r6		ΠΕ	ΑΕ	ΕΠ	ΠΜ	ΑΑ					
STORE E, r6		ΠΕ	ΑΕ	ΕΠ	ΠΜ	ΑΑ					

B2) Επίλυση εξαρτήσεων με παγώματα

	λ	$\lambda+1$	$\lambda+2$	$\lambda+3$	$\lambda+4$	$\lambda+5$	$\lambda+6$	$\lambda+7$	$\lambda+8$	$\lambda+9$	$\lambda+10$
LOAD r1, A	ΠΕ	ΑΕ	ΕΠ	ΠΜ	ΑΑ						
LOAD r2, B	ΠΕ	ΑΕ	ΕΠ	ΠΜ	ΑΑ						
MULT r1, r2, r3	ΠΕ	ΑΕ'	ΑΕ	ΕΠ	ΠΜ	ΑΑ					
STORE D, r3	ΠΕ	ΑΕ	ΕΠ	ΠΜ	ΑΑ						
ADD r1, r3, r4	ΠΕ	ΑΕ	ΕΠ	ΠΜ	ΑΑ						
LOAD r5, C	ΠΕ	ΑΕ	ΕΠ	ΠΜ	ΑΑ						
MULT r5, r4, r6	ΠΕ	ΑΕ'	ΑΕ	ΕΠ	ΠΜ	ΑΑ					
STORE E, r6	ΠΕ	ΑΕ	ΕΠ	ΠΜ	ΑΑ						

Θέμα με εξαρτήσεις και επίλυση εξαρτήσεων

Θεωρείστε επεξεργαστή μερικώς επικαλυπτόμενων λειτουργιών (pipeline) με πέντε βαθμίδες ΠΕ, ΑΕ, ΕΠ, ΠΜ και ΑΑ ο οποίος διαθέτει **ζεχωριστή** κρυφή μνήμη εντολών και δεδομένων. Δίνεται το παρακάτω τμήμα προγράμματος.

```

LOAD r1, (r7)      // r1←M(r7)
ADD r1, r2, r3     // r1+r2→r3
LOAD r4, (r3)
SUB r1, r2, r5     // r1-r2→r5
ADD r4, r1, r6

```

α. Να φτιάξετε το χρονικό διάγραμμα χρήσης των βαθμίδων, να σημειώσετε τις εξαρτήσεις και να αναφέρετε το είδος κάθε εξαρτησης (απλά να αναφέρετε το είδος, δεν χρειάζεται αιτιολόγηση).

β. Για κάθε μία των κάτωθι περιπτώσεων, χωρίς να αλλάξετε τη σειρά των εντολών να κάνετε τις απαιτούμενες ενέργειες ώστε να μην υπάρχει πρόβλημα στην ορθή εκτέλεση του τμήματος προγράμματος. Ανάλογα με την τεχνική που θα προτείνεται θα πρέπει να ξαναγράψετε το τμήμα προγράμματος ή να ξαναφτιάξετε το χρονικό διάγραμμα χρήσης των βαθμίδων.

β1. Στο μηχανισμό μερικώς επικαλυπτόμενων λειτουργιών δεν έχει υλοποιηθεί η τεχνική παράκαμψης (bypassing).

β2. Στο μηχανισμό μερικώς επικαλυπτόμενων λειτουργιών έχει υλοποιηθεί η τεχνική παράκαμψης (bypassing).

Λύση

α.

Εντολή	t	t+1	t+2	t+3	t+4	t+5	t+6	t+7	t+8	t+9	t+10	t+11	t+12	t+13
LOAD r1, (r7)	ΠΕ	ΑΕ	ΕΠ	ΠΜ	ΑΑ									
ADD r1, r2, r3		ΠΕ	ΑΕ	ΕΠ	ΠΜ	ΑΑ								
LOAD r4, (r3)			ΠΕ	ΑΕ	ΕΠ	ΠΜ	ΑΑ							
SUB r1, r2, r5				ΠΕ	ΑΕ	ΕΠ	ΠΜ	ΑΑ						
ADD r4, r1, r6					ΠΕ	ΑΕ	ΕΠ	ΠΜ	ΑΑ					

Όλες οι εξαρτήσεις είναι τύπου AME.

β1. Αντιμετώπιση των εξαρτήσεων με την **τεχνική κλειδώματος βαθμίδων**.

Εντολή	t	t+1	t+2	t+3	t+4	t+5	t+6	t+7	t+8	t+9	t+10	t+11	t+12	t+13
LOAD r1, (r7)	ΠΕ	ΑΕ	ΕΠ	ΠΜ	ΑΑ									
ADD r1, r2, r3		ΠΕ	ΑΕ'	X	ΑΕ	ΕΠ	ΠΜ	ΑΑ						
LOAD r4, (r3)				ΠΕ	ΑΕ'	X	ΑΕ	ΕΠ	ΠΜ	ΑΑ				
SUB r1, r2, r5					ΠΕ	ΑΕ'	X	ΑΕ	ΕΠ	ΠΜ	ΑΑ			
ADD r4, r1, r6						ΠΕ	ΑΕ'	ΑΕ	ΕΠ	ΠΜ	ΑΑ			

Εναλλακτικά αντιμετώπιση των εξαρτήσεων με τη χρήση εντολών NOP (μία από τις δύο λύσεις είναι αρκετή).

LOAD r1, (r7)

NOP

NOP

ADD r1, r2, r3

NOP

NOP

LOAD r4, (r3)

SUB r1, r2, r5

NOP

ADD r4, r1, r6

Εντολή	t	t+1	t+2	t+3	t+4	t+5	t+6	t+7	t+8	t+9	t+10	t+11	t+12	t+13
LOAD r1, (r7)	ΠΕ	ΑΕ	ΕΠ	ΠΜ	ΑΑ									
ADD r1, r2, r3		ΠΕ	ΑΕ	ΕΠ	ΠΜ	ΑΑ								
LOAD r4, (r3)			ΠΕ	ΑΕ	ΕΠ	ΠΜ	ΑΑ							
SUB r1, r2, r5				ΠΕ	ΑΕ	ΕΠ	ΠΜ	ΑΑ						
ADD r4, r1, r6					ΠΕ	ΑΕ	ΕΠ	ΠΜ	ΑΑ					

	t	t+1	t+2	t+3	t+4	t+5	t+6	t+7	t+8	t+9	t+10	t+11	t+12	t+13
Εντολή														
LOAD r1, (r7)	ΠΙΕ	ΑΕ	ΕΠ	ΠΜ	ΑΑ									
ADD r1, r2, r3	ΠΙΕ	ΑΕ'		ΑΕ	ΕΠ	ΠΜ	ΑΑ							
LOAD r4, (r3)				ΠΙΕ	ΑΕ	ΕΠ	ΠΜ	ΑΑ						
SUB r1, r2, r5					ΠΙΕ	ΑΕ	ΕΠ	ΠΜ	ΑΑ					
ADD r4, r1, r6						ΠΙΕ	ΑΕ	ΕΠ	ΠΜ	ΑΑ				

Εναλλακτικά αντιμετώπιση των εξαρτήσεων με τη χρήση εντολών NOP (μία από τις δύο λύσεις είναι αρκετή).

LOAD r1, (r7)

NOP

ADD r1, r2, r3

LOAD r4, (r3)

SUB r1, r2, r5

ADD r4, r1, r6

Θέμα 4 με ΜΕΛ

Θεωρείστε επεξεργαστή M μερικώς επικαλυπτόμενων λειτουργιών και επεξεργαστή N που δεν χρησιμοποιεί την τεχνική των μερικώς επικαλυπτόμενων λειτουργιών. Να σημειώσετε ποιες από τις επόμενες προτάσεις είναι σωστές (δεν χρειάζεται αιτιολόγηση).

- α. Ο χρόνος εκτέλεσης μιας εντολής στον επεξεργαστή M είναι μικρότερος από το χρόνο εκτέλεσης στον N.
- β. Ο ρυθμός ολοκλήρωσης εντολών στον επεξεργαστή N είναι μικρότερος από το ρυθμό ολοκλήρωσης εντολών στον επεξεργαστή M.
- γ. Ο ρυθμός ολοκλήρωσης εντολών στον επεξεργαστή M είναι μικρότερος από το ρυθμό ολοκλήρωσης εντολών στον επεξεργαστή N.
- δ. Ο χρόνος εκτέλεσης μιας εντολής στον επεξεργαστή N είναι μικρότερος από το χρόνο εκτέλεσης στον M.
- ε. Ο χρόνος εκτέλεσης μιας εντολής και στους δύο επεξεργαστές είναι ο ίδιος.
- ζ. Ο ρυθμός ολοκλήρωσης εντολών στους επεξεργαστές M και N είναι ο ίδιος.

Λύση

Σωστές είναι οι β και δ.

Παράδειγμα 4.12 από λυσάρι με χρήση διαδικασιακών εξαρτήσεων

Θεωρήστε υπολογιστή απλού συνόλου εντολών με **ξεχωριστή κρυφή μνήμη εντολών και δεδομένων**, με μια εντολή των 32 δυαδικών ψηφίων ή ενός δεδομένου των 32 δυαδικών ψηφίων ανά θέση μνήμης και μηχανισμό μερικώς επικαλυπτόμενων λειτουργιών των πέντε βαθμίδων προσπέλασης εντολής (ΠΙΕ), αποκωδικοποίησης εντολής (ΑΕ), εκτέλεση πράξης (ΕΠ), προσπέλασης μνήμης (ΠΜ) και αποθήκευσης αποτελέσματος (ΑΑ). Θεωρήστε επίσης ότι στη βαθμίδα ΑΕ ταυτόχρονα με την αποκωδικοποίηση της εντολής γίνεται και ανάγνωση καταχωρητών και ότι στον ίδιο κύκλο ρολογιού είναι δυνατόν να γίνει εγγραφή σε ένα καταχωρητή και ανάγνωση του περιεχομένου το οποίο γράφτηκε. Να υπολογίστε το πλήθος των κύκλων ρολογιού που απατούνται για την εκτέλεση του τμήματος προγράμματος 4.12.1 σε συμβολική γλώσσα σε καθεμία των περιπτώσεων α και β. Και στις δύο περιπτώσεις να μην αλλάξετε τη σειρά εκτέλεσης των εντολών. Να δικαιολογήσετε τις απαντήσεις σας.

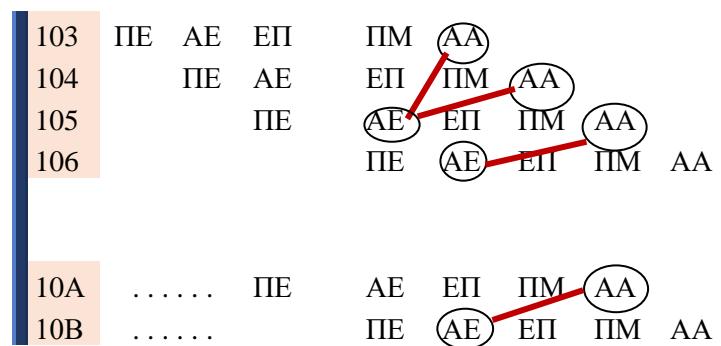
Λύση

Πρόγραμμα 4.12.1

```

100 LOAD r0, #0      // r0 ← 0
101 LOAD r1, #9
102 LOAD r2, #1
103 LOAD r3, (r7)    // r3 ← M(r7)
104 LOAD r4, (r8)    // r4 ← M(r8)
105 ADD r3, r4, r5   // r3+r4 → r5
106 STORE r5, (r9)   // r5← M(r9)
107 ADD r7, r2, r7   // r7+r2 → r7
108 ADD r8, r2, r8
109 ADD r9, r2, r9
10A SUB r1, r2, r1   // r1-r2 → r1
10B BRN r1, r0, -9H  // εάν r1-r0≠0 τότε ΜΠ = ΜΠ-9(16)
10C END

```



- α. Για την επίλυση κάθε είδους εξάρτησης χρησιμοποιείται η εντολή NOP.
- β. Για την επίλυση εξαρτήσεων τύπου "ανάγνωση μετά από εγγραφή", AME, χρησιμοποιείται η τεχνική της **παροχέτευσης**. Για την επίλυση διαδικασιακών εξαρτήσεων χρησιμοποιείται η **πρόβλεψη** η οποία βασίζεται στην τιμή του αριθμού μετατόπισης (όταν ο αριθμός μετατόπισης είναι αρνητικός γίνεται η πρόβλεψη ότι θα εκτελεστεί η εντολή BRN).

α. Πρόγραμμα 4.12.2

			λ	4 πρώτοι κύκλοι				3 επόμενοι κύκλοι				
				$\lambda+1$	$\lambda+2$	$\lambda+3$	$\lambda+4$		$\lambda+5$	$\lambda+6$	$\lambda+7$	$\lambda+8$
100	LOAD r0, #0	100	ΠΕ	ΑΕ	ΕΠ	ΠΜ	ΑΑ					
101	LOAD r1, #9	101		ΠΕ	ΑΕ	ΕΠ	ΠΜ	ΑΑ				
102	LOAD r2, #1	102			ΠΕ	ΑΕ	ΕΠ		ΠΜ	ΑΑ		
103	LOAD r3, (r7)	103				ΠΕ	ΑΕ		ΕΠ	ΠΜ		
104	LOAD r4, (r8)	104					ΠΕ		ΑΕ	ΕΠ	ΠΜ	ΑΑ
105	NOP							Ολοκλήρωση	100	Ολοκλήρωση	Ολοκλήρωση	
106	NOP								101		102	
107	ADD r3, r4, r5											
108	NOP											
109	NOP											
10A	STORE r5, (r9)											
10B	ADD r7, r2, r7											
10C	ADD r8, r2, r8											
10D	ADD r9, r2, r9											
10E	SUB r1, r2, r1											
10F	NOP											
110	NOP											
111	BRN r1, r0, -11H											
112	NOP											
113	NOP											
114	END											

Κάθε φορά μειώνουμε τον r1 κατά 1 μέχρι να μηδενιστεί και να γίνει ίσος με τον r0 οπότε τότε θα σταματήσει η εντολή BRN να εκτελείται

//11H = 00010001 = 17₁₀
 Έχουμε πλέον 17 εντολές μέσα στην επανάληψη (9 εντολές + 8 εντολές NOP), η οποία εκτελείται 9 φορές, άρα χρειαζόμαστε $17 \times 9 = 153$ κ.ρ. με δεδομένο ότι κάθε εντολή απαιτεί για την εκτέλεσή της 1 κ.ρ. Μετά την επανάληψη έχουμε την εντολή END (1 κ.ρ.) και πριν από αυτή έχουμε τρεις εντολές LOAD (3 κ.ρ.) και επιπλέον απαιτούνται 4 κ.ρ. στην αρχή του προγράμματος μέχρι να ολοκληρωθεί η πρώτη εντολή LOAD, συνολικά $153 + 1 + 3 + 4 = 161$ κ.ρ.

β. Πρόγραμμα 4.12.3

100	LOAD r0, #0		
101	LOAD r1, #9		
102	LOAD r2, #1		
103	LOAD r3, (r7)		
104	LOAD r4, (r8)		
105	NOP		
106	ADD r3, r4, r5		
107	STORE r5, (r9)		
108	ADD r7, r2, r7		
109	ADD r8, r2, r8		
10A	ADD r9, r2, r9		
10B	SUB r1, r2, r1		
10C	BRN r1, r0, -AH	τώρα το 9 έγινε 10(A) λόγω της NOP που παρεμβλήθηκε	
10D	END		

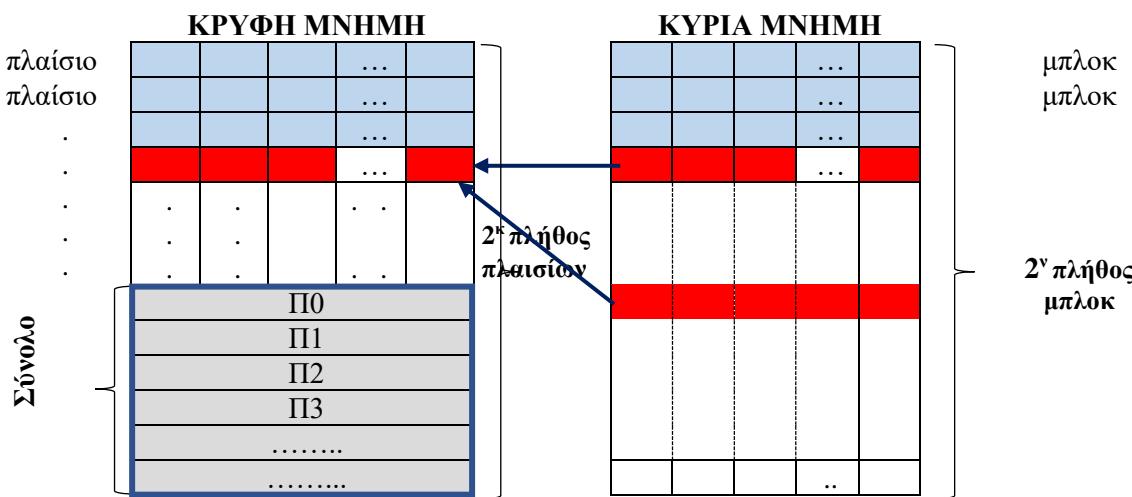
Επειδή ο βρόχος περιλαμβάνει 10 εντολές, για την ολοκλήρωση του θα απαιτηθούν $10 \times 9 = 90$ κύκλοι ρολογιού, αλλά θα απαιτηθούν και δύο πρόσθετοι κύκλοι ρολογιού οι οποίοι χάνονται λόγω της **λανθασμένης πρόβλεψης** κατά την **ένατη (τελευταία) εκτέλεση της εντολής BRN**, όπου η συνθήκη που εξετάζει η εντολή BRN γίνεται ψευδής. Θα απαιτηθεί επίσης, ένας επιπλέον κύκλος ρολογιού για την εντολή END και 3 κ.ρ. για τις εντολές LOAD πριν την επανάληψη και LOAD (3 κ.ρ.) και επιπλέον απαιτούνται και 4 κ.ρ. στην αρχή του προγράμματος μέχρι να ολοκληρωθεί η πρώτη εντολή LOAD. Επομένως ο συνολικός αριθμός κύκλων ρολογιού που θα απαιτηθούν για την εκτέλεση του προγράμματος είναι: $4 + 3 + 10 \times 9 + 2 + 1 = 100$ κύκλοι ρολογιού.

Κεφάλαιο 5.3 – Κρυφή μνήμη

Επεξήγηση τρόπου λειτουργίας κρυφής μνήμης (cache memory)

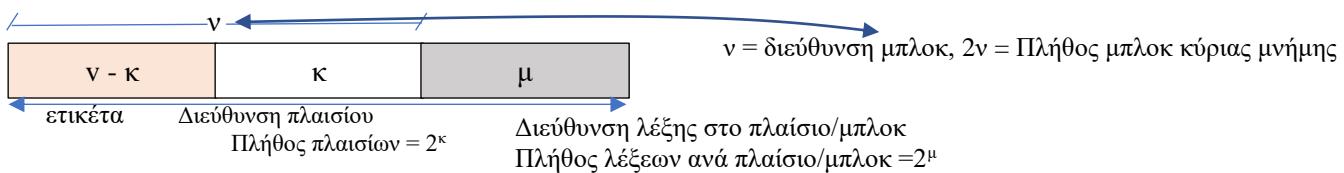
1. Βασικές έννοιες κρυφής μνήμης

- **Λέξη = θέση μνήμης.** Το μήκος (μέγεθος) της δεν είναι σταθερό και προσδιορίζεται κάθε φορά από την εκφώνηση της άσκησης. Ως έννοια, η λέξη (θ.μ.) είναι διαφορετική από την ψηφιολέξη. Όταν η λέξη περιέχει 8 δυαδικά ψηφία (bits) τότε και μόνο τότε ονομάζεται και ψηφιολέξη (byte). Ορισμένες φορές μπορεί το μήκος μιας λέξης να είναι των 16 bits ή και διαφορετικό.
- **Πλαισίο** = συλλογή (ομάδα) γειτονικών θέσεων μνήμης (λέξεων) που υπάρχει στην **κρυφή μνήμη**.
- **Μπλοκ** = συλλογή (ομάδα) γειτονικών θέσεων μνήμης (λέξεων) που υπάρχει στην **κύρια μνήμη**.
- **Σύνολο** = συλλογή (ομάδα) πλαισίων στην κρυφή μνήμη. Το πλήθος των πλαισίων που υπάρχει ανά σύνολο δεν είναι σταθερό και προσδιορίζεται κάθε φορά από την εκφώνηση της άσκησης. Πιο συγκεκριμένα, όταν έχουμε οργάνωση **τ - τρόπων συνόλου συσχέτισης**, τότε το πλήθος των πλαισίων ανά σύνολο είναι τ .
- Ισχύει ότι **μέγεθος πλαισίου = μέγεθος μπλοκ**.
- Ισχύει ότι πλήθος μπλοκ κύριας μνήμης (2^v) >> πλήθος πλαισίων κρυφής μνήμης (2^k) και αυτό οφείλεται στη χωρητικότητα της κύριας μνήμης που είναι μεγαλύτερη από την αντίστοιχη χωρητικότητα της κρυφής μνήμης.

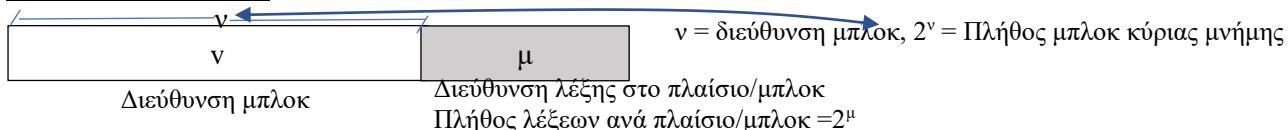


2. Βασικές οργανώσεις (είδη) κρυφής μνήμης:

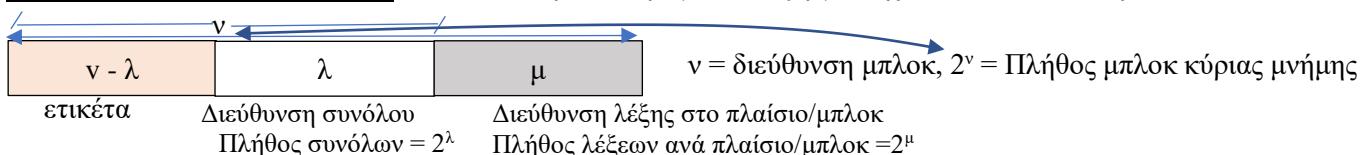
- **Μονοσήμαντη απεικόνιση.** Η διεύθυνση που παράγει ο επεξεργαστής αποτελείται από τρία πεδία:



- **Πλήρους συσχέτισης.** Η διεύθυνση που παράγει ο επεξεργαστής αποτελείται από δύο πεδία:



- **τ - τρόπων συνόλου συσχέτισης.** Η διεύθυνση που παράγει ο επεξεργαστής αποτελείται από τρία πεδία:



- Η **ετικέτα** που υπάρχει στην οργάνωση μονοσήμαντης απεικόνισης, αλλά και στην οργάνωση **τ - τρόπων συνόλου συσχέτισης**, προκύπτει πάντα με **αφαίρεση**, από το συνολικό μήκος της διεύθυνσης αφαιρούμε τα μήκη των υπολοίπων πεδίων.

- Μια κρυφή μνήμη με οργάνωση **τ-τρόπων συνόλου συσχέτισης**, όπου $\tau = 1$, έχει ένα πλαίσιο ανά σύνολο επομένως έχει **οργάνωση μονοσήμαντης απεικόνισης**. Επίσης σε μία κρυφή μνήμη με οργάνωση τ-τρόπων συνόλου συσχέτισης, διατηρώντας την χωρητικότητά της και το μέγεθος του πλαισίου σταθερά, κάθε φορά που διπλασιάζουμε την τιμή του τ υποδιπλασιάζεται ο αριθμός των συνόλων της κρυφής μνήμης. Επομένως ελαττώνεται κατά ένα δυαδικό ψηφίο το εύρος του πεδίου "διεύθυνση του συνόλου στην κρυφή μνήμη" της διεύθυνσης. Όταν ο αριθμός των πλαισίων ανά σύνολο γίνει ίσος με τον αριθμό των πλαισίων της κρυφής μνήμης, τότε η κρυφή μνήμη έχει ένα σύνολο και επομένως **οργάνωση πλήρους συσχέτισης**.

3. Σύγκριση οργανώσεων κρυφής μνήμης

misses κρ. μν. μονοσ. απεικ. > **misses** κρ. μν. τ-τρόπων. > **misses** κρ. μν. πλήρους συσχέτισης.

κόστος κρ. μν. μονοσ. απεικ. < **κόστος** κρ. μν. τ-τρόπων. < **κόστος** κρ. μν. πλήρους συσχέτισης.

t_{προσπ.} κρ. μν. μονοσ. απεικ. < **t_{προσπ.}** κρ. μν. τ-τρόπων. < **t_{προσπ.}** κρ. μν. πλήρους συσχέτισης.

Θέμα σχετικά με τα πλεονεκτήματα/μειονεκτήματα κάθε οργάνωσης κρυφής μνήμης

Να αναφέρετε τους τρόπους απεικόνισης μπλοκ της κύριας μνήμης σε πλαίσια της κρυφής. Ποια τα πλεονεκτήματα και μειονεκτήματα κάθε τρόπου;

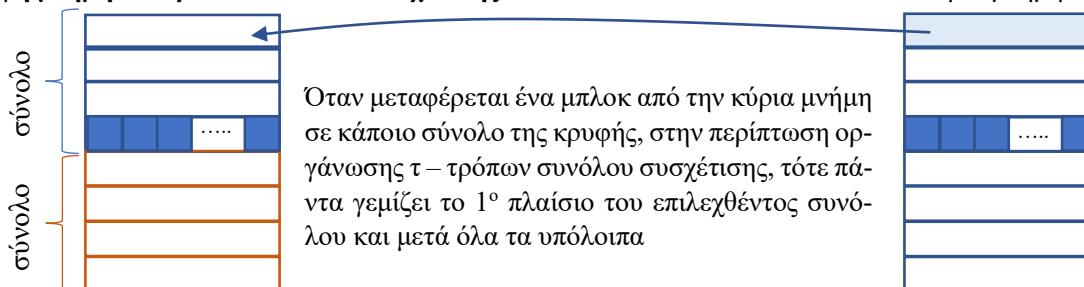
Λύση

a) **Οργάνωση μονοσήμαντης απεικόνισης:** κάθε μπλοκ της κύριας μνήμης μπορεί να τοποθετηθεί σε ένα συγκεκριμένο πλαίσιο της κρυφής. Η κρυφή μνήμη με οργάνωση μονοσήμαντης απεικόνισης παρουσιάζει το **μικρότερο χρόνο προσπέλασης** και επίσης έχει το **μικρότερο κόστος**, ενώ ταυτόχρονα παρουσιάζει το **μεγαλύτερο λόγο αποτυχιών**.

b) **Οργάνωση πλήρους συσχέτισης:** κάθε μπλοκ της κύριας μνήμης μπορεί να τοποθετηθεί σε **οποιοδήποτε** πλαίσιο της κρυφής. Η κρυφή μνήμη με οργάνωση πλήρους συσχέτισης έχει το **μικρότερο αριθμό αποτυχιών σε σχέση με τις υπόλοιπες δύο οργάνωσεις**, όμως τόσο **ο χρόνος προσπέλασης** και **το κόστος υλοποίησης είναι μεγαλύτερα** από ότι στις υπόλοιπες οργανώσεις.

c) **Οργάνωση τ-τρόπων συνόλου συσχέτισης:** η κρυφή μνήμη ότι αποτελείται από ομάδες των πλαισίων που καλούνται σύνολα και κάθε μπλοκ της κύριας μνήμης μπορεί να τοποθετηθεί σε οποιοδήποτε πλαίσιο ενός συγκεκριμένου συνόλου.

Κρυφή μνήμη 4 – τρόπων συνόλου συσχέτισης



Τόσο ο **χρόνος προσπέλασης** όσο και ο **αριθμός αποτυχιών** και το **κόστος υλοποίησης** είναι μεταξύ των αντίστοιχων μεγεθών των άλλων δύο οργανώσεων.

4. Θεωρία - Χαρακτηριστικά μεγέθη κρυφής μνήμης

Για να υπολογίσουμε το **πλήθος των πλαισίων μιας κρυφής μνήμης** με οποιαδήποτε οργάνωση θα πρέπει -αν δεν δίνεται από την εκφόνηση- να χρησιμοποιήσουμε **τη χωρητικότητα της κρυφής μνήμης** και να εφαρμόσουμε μια απλή μέθοδο των τριών. Ιδιαίτερη προσοχή χρειάζεται όταν π.χ. το μέγεθος μιας λέξης είναι των 16 bits = 2 bytes και η χωρητικότητα της κρυφής μνήμης έχει δοθεί σε KB, MB κ.λ.π. Για να υπολογίσουμε το **πλήθος των συνόλων μιας κρυφής μνήμης** πρέπει νωρίτερα να υπολογίσουμε το πλήθος των πλαισίων.

Στην περίπτωση χρήσης της **στρατηγικής απελευθέρωσης πλαισίων FIFO**, όταν χρειαστεί, αντικαθίσταται εκείνο το πλαίσιο που μπήκε πρώτο, ενώ σε περίπτωση επιτυχίας, **το πλαίσιο αφήνεται κενό**, επειδή δεν πραγματοποιείται καμία μεταφορά από την κύρια μνήμη.

Στην περίπτωση χρήσης της **στρατηγικής απελευθέρωσης πλαισίων LRU**, όταν χρειαστεί, αντικαθίσταται εκείνο το πλαίσιο που έχει να χρησιμοποιηθεί την πιο πολλή ώρα (δηλαδή το πιο παλιό, δηλαδή το μη – χρησιμοποιηθέν πρόσφατα), ενώ σε

περίπτωση επιτυχίας, αυτή σημειώνεται με – (κάτι που σημαίνει ότι το συγκεκριμένο πλαίσιο ενημερώνεται/ανανεώνεται, οπότε αυτό το γεγονός λαμβάνεται υπόψη στην επόμενη αντικατάσταση που θα απαιτηθεί).

Για να υπολογίσουμε **τη συνολική χωρητικότητα μιας κρυφής μνήμης**, θα πρέπει εκτός από τη χωρητικότητα των πλαισίων να λάβουμε υπόψη μας και τη χωρητικότητα των ετικετών, του δυαδικού ψηφίου εγκυρότητας και της πολιτικής (στρατηγικής) απελευθέρωσης πλαισίων, εφόσον δίνεται (στην περίπτωση αυτή έχουμε πλήθος συνόλων $x \frac{\text{bit}}{\text{σύνολο}}$). Πιο συγκεκριμένα:

- Χωρητικότητα κρυφής μνήμης οργάνωσης **μονοσήμαντης απεικόνισης** = αριθμός πλαισίων x (μέγεθος κάθε πλαισίου + μέγεθος ετικέτας + μέγεθος bit εγκυρότητας)

- Χωρητικότητα κρυφής μνήμης **$\tau - \tau_{\text{ρόπων}}$** συνόλου συσχέτισης = αριθμός συνόλων $x \frac{\text{αριθμός πλαισίων}}{\text{σύνολο}} x$ (μέγεθος κάθε πλαισίου + μέγεθος ετικέτας + μέγεθος bit εγκυρότητας).

- Χωρητικότητα μνήμης ετικετών = αριθμός πλαισίων x μέγεθος ετικέτας ή αριθμός συνόλων $x \frac{\text{αριθμός πλαισίων}}{\text{σύνολο}} x$ μέγεθος ετικέτας.

Όταν στη **μονοσήμαντη απεικόνιση** έχουμε ίδιο πλαίσιο (κ) και ίδια ετικέτα ($v - \kappa$), τότε οι διευθύνσεις **μπορούν και πρέπει να είναι ταυτόχρονα στην κρυφή μνήμη**. Όταν όμως στην ίδια οργάνωση έχουμε ίδιο πλαίσιο (κ) και διαφορετική ετικέτα ($v - \kappa$), τότε οι διευθύνσεις **δεν μπορούν να είναι ταυτόχρονα στην κρυφή μνήμη**, διότι η μία αντικαθιστά την άλλη.

Όταν στην **$\tau - \tau_{\text{ρόπων}}$** συνόλου συσχέτισης έχουμε **ίδιο σύνολο (λ) και ίδια ετικέτα ($v - \lambda$)**, τότε οι διευθύνσεις μπορούν και πρέπει να είναι ταυτόχρονα στην κρυφή μνήμη και μάλιστα πηγαίνουν όχι μόνο στο ίδιο σύνολο, αλλά και στο ίδιο πλαίσιο του συνόλου αυτού. Όταν όμως στην ίδια οργάνωση έχουμε **ίδιο σύνολο (λ) και διαφορετική ετικέτα ($v - \lambda$)**, τότε και πάλι οι διευθύνσεις μπορούν να είναι ταυτόχρονα στην κρυφή μνήμη, διότι η μία πηγαίνει στο Π0 του συνόλου αυτού και η άλλη στο Π1 του συνόλου αυτού. Αν όμως έρθει και **τρίτη διεύθυνση** στο ίδιο σύνολο και με δεδομένο ότι $\tau = 2$, αντικαθιστά μια από τις δύο προηγούμενες, ανάλογα με τη στρατηγική αντικατάστασης που χρησιμοποιείται.

Στην οργάνωση πλήρους συσχέτισης τα πλαίσια της κρυφής μνήμης γεμίζουν με τη σειρά από το πρώτο προς το τελευταίο. Αν έρθουν μπλοκ με ίδια διεύθυνση (v), μετακινούνται στο ίδιο πλαίσιο της κρυφής μνήμης.

Αν έχουμε **4 λέξεις/πλαίσιο**

00	01	10	11
17			23

$$2^{\mu} = 4 \Rightarrow \mu = 2$$

Αν για παράδειγμα έχουμε μια διεύθυνση $17_{10} = 001 - \mathbf{00} - \mathbf{01}$ ΜΠ(16, 17, 18, 19) \rightarrow Π0 ή ΜΠ(16 - 19) \rightarrow Π0
Αν για παράδειγμα έχουμε μια διεύθυνση $23_{10} = 001 - \mathbf{01} - \mathbf{11}$ ΜΠ(20, 21, 22, 23) \rightarrow Π1 ή ΜΠ(20 - 23) \rightarrow Π1

Θέμα σχετικά με τις στρατηγικές απελευθέρωσης πλαισίων από την κρυφή μνήμη

Ποιες οι στρατηγικές απελευθέρωσης πλαισίων;

Λύση

Όταν η ΚΜΕ ζητήσει κάποια πληροφορία από την κρυφή μνήμη και δεν υπάρχει αυτή διαθέσιμη, θα πρέπει ένα μπλοκ της κύριας μνήμης που περιέχει τα απαιτούμενα δεδομένα να προσκομισθεί στην κρυφή μνήμη. Αν η κρυφή μνήμη είναι γεμάτη, τότε το μπλοκ που θα προσκομισθεί θα πρέπει να αντικαταστήσει κάποιο άλλο. Στην περίπτωση κρυφής μνήμης με οργάνωση μονοσήμαντης απεικόνισης αφού ένα μπλοκ μπορεί να τοποθετηθεί μόνο σε ένα συγκεκριμένο πλαίσιο, η στρατηγική απελευθέρωσης είναι τετριμένη. Στην περίπτωση των άλλων δυο οργανώσεων όπου ένα μπλοκ μπορεί να τοποθετηθεί σε περισσότερα από ένα πλαίσιο, πρέπει να επιλέγει ένα από αυτά. Υπάρχουν 3 στρατηγικές που χρησιμοποιούνται:

1. Τυχαία επιλογή
2. Επιλογή LRU
3. Επιλογή FIFO

Θέμα σχετικά με τακτικές ενημέρωσης του επόμενου επίπεδου της ιεραρχικής μνήμης

Ποιες είναι οι τακτικές ενημέρωσης του επόμενου επίπεδου της ιεραρχικής μνήμης;

Λύση

Για την περίπτωση όπου το μπλοκ στο οποίο βρίσκεται η πληροφορία που θα εγγραφεί από την ΚΜΕ στην μνήμη βρίσκεται στην κρυφή μνήμη, ο σχεδιαστής της κρυφής μνήμης μπορεί να επιλέξει μεταξύ των παρακάτω 2 τακτικών:

- Τακτική της **άμεσης ενημέρωσης**: η πληροφορία γράφεται ταυτόχρονα τόσο από πλαίσιο της κρυφής μνήμης όσο και στο μπλοκ του επόμενου επιπέδου της iεραρχικής μνήμης.
- Τακτική της **τελικής ενημέρωσης**: η αντίστοιχη λέξη της κύριας μνήμης ενημερώνεται μόνο όταν το μπλοκ που περιέχει την τροποποιημένη λέξη πρόκειται να αντικατασταθεί στην κρυφή μνήμη από ένα άλλο μπλοκ της κύριας μνήμης.

Για την περίπτωση όπου το μπλοκ στο οποίο βρίσκεται η πληροφορία που θα εγγραφεί από την ΚΜΕ στην μνήμη δεν βρίσκεται στην κρυφή μνήμη, ο σχεδιαστής της κρυφής μνήμης μπορεί να επιλέξει μεταξύ των παρακάτω 2 τακτικών:

- Τακτική προσκόμισης **κατά την εγγραφή**: το μπλοκ προσκομίζεται από την κύρια μνήμη στην κρυφή και ακολουθείται μια από τις 2 παραπάνω τακτικές.
- Τακτική **μη προσκόμισης κατά την εγγραφή**: η πληροφορία αποθηκεύεται κατευθείαν στην κύρια μνήμη στο μπλοκ που αντιστοιχεί.

Θέμα Ιουνίου 2016 με κρυφή μνήμη

Θεωρείστε κρυφή μνήμη των 512 πλαισίων με 32 λέξεις ανά πλαίσιο και οργάνωση:

α. Μονοσήμαντης απεικόνισης

β. 2-τρόπων συνόλου συσχέτισης

Θεωρήστε ότι το εύρος της θέσης μνήμης είναι ίσο με το εύρος της λέξης και ο επεξεργαστής παράγει διευθύνσεις των 24 δυαδικών ψηφίων.

- Για κάθε μία από τις περιπτώσεις α, και β να δώσετε πως χρησιμοποιείται η διεύθυνση που παράγει ο επεξεργαστής για την προσπέλαση της κρυφής μνήμης (δηλαδή τα πεδία της διεύθυνσης, το εύρος κάθε πεδίου και τι δηλώνει κάθε πεδίο).
- Να δώσετε τις διευθύνσεις των πλαισίων της κρυφής μνήμης στα οποία μπορούν να αποθηκευτούν τα περιεχόμενα των θέσεων της κύριας μνήμης με διευθύνσεις στο δεκαεξαδικό F80CD3, F83CD8, F84CC5 και F80CC0 σε κάθε μία από τις περιπτώσεις α, και β.
- Ποια από τα περιεχόμενα των ανωτέρω θέσεων της κύριας μνήμης **δεν** είναι δυνατόν να βρίσκονται ταυτόχρονα στην κρυφή μνήμη σε κάθε μία από τις περιπτώσεις α, και β και γιατί;

Λύση

α. Οργάνωση μονοσήμαντης απεικόνισης

- Αφού έχουμε 512 πλαισία η διεύθυνση πλαισίου θα είναι των 9 δυαδικών ψηφίων, αφού $2^k = 512 \Rightarrow k = 9$ bits και αφού έχουμε 32 λέξεις ανά πλαίσιο, όπου σε κάθε λέξη αντιστοιχεί μια διεύθυνση, η διεύθυνση της λέξης μέσα στο πλαίσιο θα είναι των 5 δυαδικών ψηφίων, αφού $2^\mu = 32 \Rightarrow \mu = 5$ bits. Επομένως η διεύθυνση που παράγεται από τον επεξεργαστή χρησιμοποιείται για την προσπέλαση της κρυφής μνήμης ως εξής:

10 δυαδικά ψηφία	9 δυαδικά ψηφία	5 δυαδικά ψηφία
Ετικέτα (v-k)	Διεύθυνση πλαισίου (κ)	Διεύθυνση της λέξης μέσα στο πλαίσιο/μπλοκ (μ)

2.

διευθύνσεις	10 δυαδικά ψηφία	9 δυαδικά ψηφία	5 δυαδικά ψηφία
Ετικέτα v - κ	Διεύθυνση πλαισίου - κ	Διεύθυνση της λέξης μέσα στο πλαίσιο - μ	
F80CD3	1111100000	001100110	10011
F83CD8	1111100000	111100110	11000
F84CC5	1111100001	001100110	00101
F80CC0	1111100000	001100110	00000

- Δεν μπορούν να βρίσκονται ταυτόχρονα τα περιεχόμενα της τρίτης διεύθυνσης F84CC5 με τα περιεχόμενα των διευθύνσεων F80CD3 και F80CC0 (αυτές οι διευθύνσεις αφορούν το ίδιο μπλοκ που μεταφέρεται στο ίδιο πλαίσιο, επομένως αυτές μπορούν και πρέπει να είναι ταυτόχρονα στην κρυφή μνήμη), διότι έχουν την ίδια διεύθυνση πλαισίου και διαφορετική ετικέτα, **δηλαδή ανήκουν σε διαφορετικά μπλοκ που αντιστοιχούν στο ίδιο πλαίσιο**. Η δεύτερη διεύθυνση μπορεί να είναι ταυτόχρονα στην κρυφή μνήμη με οποιαδήποτε από τις υπόλοιπες, επειδή πηγαίνει σε διαφορετικό πλαίσιο σε σχέση με αυτές.

β. Οργάνωση 2-τρόπων συνόλου συσχέτισης

Αφού έχουμε κρυφή μνήμη 2-τρόπων συνόλου συσχέτισης με 512 πλαίσια συνεπάγεται ότι η κρυφή μνήμη θα έχει $512/2 = 256$ σύνολα, άρα η διεύθυνση συνόλου θα είναι των 8 δυαδικών ψηφίων, $2^8 = 256 \Rightarrow \lambda = 8$ bits. Αφού έχουμε 32 λέξεις ανά πλαίσιο όπου σε κάθε λέξη αντιστοιχεί μια διεύθυνση, η διεύθυνση της λέξης μέσα στο πλαίσιο θα είναι των 5 δυαδικών ψηφίων, αφού $2^5 = 32 \Rightarrow \mu = 5$ bits. Επομένως η διεύθυνση που παράγεται από τον επεξεργαστή χρησιμοποιείται για την προσπέλαση της κρυφής μνήμης ως εξής:

24

11 δυαδικά ψηφία	8 δυαδικά ψηφία	5 δυαδικά ψηφία
Ετικέτα ($v-\lambda$)	Διεύθυνση συνόλου (λ)	Διεύθυνση της λέξης μέσα στο πλαίσιο (μ)
διευθύνσεις	11 δυαδικά ψηφία	8 δυαδικά ψηφία
F80CD3	ετικέτα	Διεύθυνση συνόλου
F83CD8	11111000000	01100110
F84CC5	11111000001	11100110
F80CC0	11111000010	01100110
	11111000000	01100110

Δεν μπορούμε να απαντήσουμε σε ποιο πλαίσιο της κρυφής μνήμης, θα αποθηκευτεί ένα μπλοκ, αλλά σε ποιο σύνολο (σε ένα από τα δύο πλαίσια του συνόλου).

3. Δεν υπάρχουν διεύθυνσεις των οποίων τα περιεχόμενα δεν μπορούν να βρίσκονται ταυτόχρονα στην κρυφή μνήμη. Τα περιεχόμενα της διεύθυνσης F84CC5 με τα περιεχόμενα των διεύθυνσεων F80CC0 και F80CD3 (οι δύο τελευταίες διεύθυνσεις ανήκουν στο ίδιο μπλοκ) ανήκουν σε δύο διαφορετικά μπλοκ που αντιστοιχούν στο ίδιο σύνολο. Ωστόσο το σύνολο έχει δύο πλαίσια οπότε και τα δύο μπλοκ μπορούν να βρίσκονται ταυτόχρονα στην κρυφή μνήμη. Η διεύθυνση F83CD8 ανήκει σε μπλοκ που αντιστοιχεί σε άλλο σύνολο, οπότε μπορούν τα περιεχόμενά της να βρίσκονται ταυτόχρονα στην κρυφή μνήμη με αυτά των προηγούμενων.

Θέμα Ιουνίου 2019 με κρυφή μνήμη

Θεωρείστε τους υπολογιστές A και B με τα ακόλουθα κοινά χαρακτηριστικά: Αρτηρία διεύθυνσεων των 26 δυαδικών ψηφίων και αρτηρία δεδομένων των 8 δυαδικών ψηφίων. Η κύρια μνήμη έχει οργάνωση μιας ψηφιολέξης ανά θέση μνήμης. Κρυφή μνήμη επεξεργαστή με χωρητικότητα 16 Κψηφιολέξεις και πλαίσιο των 32 ψηφιολέξεων. Η κρυφή μνήμη επεξεργαστή του υπολογιστή A έχει οργάνωση μονοσήμαντης απεικόνισης, ενώ του υπολογιστή B 2-τρόπων συνόλου συσχέτισης. Για κάθε ένα εκ των υπολογιστών A και B:

1. Να δώσετε τα πεδία (εύρος και σημασία) από τα οποία θεωρούμε ότι αποτελείται η διεύθυνση που παράγεται από τον επεξεργαστή κατά την προσπέλαση της κρυφής μνήμης.
2. Να αναφέρετε και να αιτιολογήσετε ποια από τα δεδομένα που είναι αποθηκευμένα στις δεκαεξαδικές διεύθυνσεις α. 3E91794 β. 3E93798 γ. 3E9D7B3 και δ. 3E937A2 μπορούν να βρίσκονται ταυτόχρονα στην κρυφή μνήμη.
3. Να υπολογίσετε το συνολικό πλήθος των δυαδικών ψηφίων που αποθηκεύονται στην κρυφή μνήμη επεξεργαστή.

Λύση

i) **Μονοσήμαντη απεικόνιση:** Η ψηφιολέξη είναι ταυτόχρονα και λέξη, λόγω της οργάνωσης που δίνεται.

Διεύθυνση μπλοκ, $v = 21$ bits		
$v - \kappa = 12$	$\kappa = 9$	$\mu = 5$
ετικέτα	Διεύθυνση πλαισίου	Διεύθυνση λέξης μέσα στο πλαίσιο/μπλοκ
Για υπολογισμό πλαισίων έχουμε: 1πλ. x πλ;	32 ψηφιολέξεις = 32 bytes = 32 θ.μ. 16K ψηφιολέξεις	$2^{\mu} = 32$ λέξεις ($\theta.\mu.$) $\Rightarrow \mu = 5$ bits.

$x = 2^4 * 2^{10}/2^5 \pi\lambda. = 2^9 = 2^{\kappa} \Rightarrow \kappa = 9$ bits.

$v - \kappa$ (12 bits)	κ (9 bits)	μ (5 bits)
a. 3E91794 = 00 11 1110 1001 00 01 0111 100 1 0100	11 1110 1001 00 01 0111 100	1 0100
β. 3E93798= 00 11 1110 1001 00 11 0111 111 1 1010	11 1110 1001 00 11 0111 100	1 1010
γ. 3E9D7B3 = 00 11 1110 1001 11 01 0111 101 1 0011	11 1110 1001 11 01 0111 101	1 0011
δ. 3E937A2 = 00 11 1110 1001 00 11 0111 101 0 0010	11 1110 1001 00 11 0111 101	0 0010

Επειδή όλες οι διεύθυνσεις μεταφέρονται σε **διαφορετικά πλαισία** της κρυφής μνήμης (διαφορετικό κ), μπορούν να είναι **ταυτόχρονα** στην κρυφή μνήμη.

ii) 2 – τρόπων συνόλου συσχέτισης

Διεύθυνση μπλοκ, $v = 21$ bits

$v - \lambda = 13$ bits	$\lambda = 8$ bits = 2 hex ψηφία	$\mu = 5$ bits
ετικέτα	Διεύθυνση συνόλου	Διεύθυνση λέξης μέσα στο πλαίσιο και το μπλοκ
1 σύνολο	2 πλ.	

x; 512 πλ.

$$x = 2^9/2^1 = 2^8 = 2^\lambda \Rightarrow \lambda = 8 \text{ bits.}$$

	$v - \lambda$ (13 bits)	λ (8 bits)	μ (5 bits)
a. 3E91794 = 00	11 1110 1001 000 1 0111 100 1 0100	1 0111 100	1 0100
β. 3E93798 = 00	11 1110 1001 001 1 0111 1001 1010	1 0111 1001	1 1010
γ. 3E9D7B3 = 00	11 1110 1001 110 1 0111 101 1 0011	1 0111 101	1 0011
δ. 3E937A2 = 00	11 1110 1001 001 1 0111 101 0 0010	1 0111 101	0 0010

Οι διευθύνσεις (γ) και (δ) αναφέρονται σε διαφορετικό μπλοκ, τα οποία μετακινούνται στο ίδιο σύνολο (με διεύθυνση 10111101)

και μπορούν να βρίσκονται ταυτόχρονα στην κρυφή μνήμη, από τη στιγμή που καταλαμβάνουν διαφορετικά πλαίσια του ίδιου συνόλου. Οι διευθύνσεις (α) και (β) μετακινούνται στο ίδιο σύνολο που έχει διεύθυνση 1 0111 100, αλλά αυτό είναι 2-τρόπων, οπότε οι διευθύνσεις αυτές μπορούν να είναι ταυτόχρονα στην κρυφή μνήμη, τόσο μεταξύ τους, όσο και με τις υπόλοιπες διευθύνσεις. Επομένως, όλες οι διευθύνσεις μπορούν να **είναι ταυτόχρονα στην κρυφή μνήμη** στη συγκεκριμένη οργάνωση.

3. Συνολικό πλήθος δυαδικών ψηφίων που αποθηκεύονται στην κρυφή μνήμη για οργάνωση μονοσήμαντης απεικόνισης = αριθμός πλαισίων x (μέγεθος κάθε πλαισίου + μέγεθος ετικέτας + μέγεθος bit εγκυρότητας) = 512 πλ. x ($32 \frac{\text{ψηφιολέξεις}}{\text{πλ.}} + 12 \frac{\text{bits}}{\text{πλ.}} + 1 \frac{\text{bits}}{\text{πλ.}}$) = $2^9 \pi. x (2^5 \frac{\text{ψηφιολέξεις}}{\text{πλ.}} x 2^3 \frac{\text{bits}}{\text{ψηφιολέξη}} + 12 \frac{\text{bits}}{\text{πλ.}} + 1 \frac{\text{bits}}{\text{πλ.}}) = 2^9 x (256 + 13)$ δυαδικά ψηφία = $2^9 x 269$ bits.

Συνολικό πλήθος δυαδικών ψηφίων που αποθηκεύονται στην κρυφή μνήμη για οργάνωση 2-τρόπων συνόλου συσχέτισης = αριθμός συνόλων x $\frac{2 \text{ πλαίσια}}{\text{σύνολο}} x$ (μέγεθος κάθε πλαισίου + μέγεθος ετικέτας + μέγεθος bit εγκυρότητας) = 256 σύνολα x $2 \frac{\text{πλαίσια}}{\text{σύνολο}} x (32 \frac{\text{ψηφιολέξεις}}{\text{πλ.}} x 2^3 \frac{\text{bits}}{\text{ψηφιολέξη}} + 13 \frac{\text{bits}}{\text{πλ.}} + 1 \frac{\text{bits}}{\text{πλ.}}) = 2^9 \pi. x (2^5 \frac{\text{ψηφιολέξεις}}{\text{πλ.}} x 2^3 \frac{\text{bits}}{\text{ψηφιολέξη}} + 13 \frac{\text{bits}}{\text{πλ.}} + 1 \frac{\text{bits}}{\text{πλ.}}) = 2^9 x (256 + 14)$ δυαδικά ψηφία = $2^9 x 270$ bits.

Θέμα Σεπτεμβρίου 2019 με κρυφή μνήμη

Θεωρείστε επεξεργαστή με αρτηρία διεύθυνσεων των 26 δυαδικών ψηφίων, κύρια μνήμη με οργάνωση μιας ψηφιολέξης (byte) ανά θέση και κρυφή μνήμη με χωρητικότητα 16 Κψηφιολέξεων και πλαίσιο των 16 ψηφιολέξεων. Θεωρείστε τις ακόλουθες δύο οργανώσεις της κρυφής μνήμης:

- α. Οργάνωση μονοσήμαντης απεικόνισης (direct mapped)
- β. 2-τρόπων συνόλου συσχέτισης (2-way set associative)

Για κάθε μία από τις ανωτέρω οργανώσεις:

- i. Να δώσετε την ανάλυση της διεύθυνσης που παράγει ο επεξεργαστής έτσι ώστε χρησιμοποιείται από την κρυφή μνήμη (δηλαδή να δώσετε τα πεδία, το εύρος κάθε ενός και τι δηλώνει το κάθε ένα πεδίο).
- ii. Να αναφέρετε και να αιτιολογήσετε εάν τα δεδομένα που είναι αποθηκευμένα στις ακόλουθες τρεις δεκαεξαδικές διευθύνσεις: α: 3FF438A, β: 3FF2381, γ: 3FF6384 μπορούν να βρίσκονται ταυτόχρονα στην κρυφή μνήμη.

Λύση

i) Μονοσήμαντης απεικόνισης: Η ψηφιολέξη είναι ταυτόχρονα και λέξη, λόγω της οργάνωσης που δίνεται.

Διεύθυνση μπλοκ, $v = 22$ bits	26	
$v - \kappa = 12$	$\kappa = 10$	$\mu = 4$
ετικέτα	Διεύθυνση πλαισίου	Διεύθυνση λέξης μέσα στο πλαίσιο/μπλοκ
$2^\mu = 16$ λέξεις (θ.μ.) $\Rightarrow \mu = 4$ bits.		
Για υπολογισμό πλαισίων έχουμε: 1πλ.	16 ψηφιολέξεις = 16 bytes = 16 θ.μ.	
$\times \pi\lambda;$	16 K ψηφιολέξεις	$x = 2^{14}/2^4 = 2^{10} \pi\lambda. = 2^\kappa \Rightarrow \kappa = 10$ bits.
$v - \kappa$ (12 bits)	κ (10 bits)	μ (4 bits)

α. 3FF438A = 0011111111101	0000111000	1010
β. 3FF2381 = 0011111111100	1000111000	0001
γ. 3FF6384 = 0011111111101	1000111000	0100

Μεθοδολογία: Ξεκινάμε ελέγχοντας ποιες από τις παραπάνω διευθύνσεις έχουν το **ίδιο κ (διεύθυνση πλαισίου)**.

Αντές είναι οι διευθύνσεις (β), (γ). Στη συνέχεια ελέγχουμε **αν αυτές έχουν ίδια ετικέτα (ν – κ)**. Δεν έχουν ίδια ετικέτα. Από τη στιγμή που έχουν διαφορετική ετικέτα, σημαίνει ότι αναφέρονται σε διαφορετικά μπλοκ, τα οποία μετακινούνται στο ίδιο πλαίσιο, με αποτέλεσμα να αντικαθιστά το ένα το άλλο. **Επομένως, οι διευθύνσεις (β) και (γ) δεν μπορούν να είναι ταυτόχρονα στην κρυφή μνήμη.** Όσον αφορά τη διεύθυνση (α), αυτή πηγαίνει σε διαφορετικό πλαίσιο σε σχέση με τις (β) και (γ), επομένως μπορεί να είναι ταυτόχρονα στην κρυφή μνήμη, με οποιαδήποτε από αυτές.

ii) 2 – τρόπων συνόλου συσχέτισης:

Διεύθυνση μπλοκ, ν = 22 bits		
$\nu - \lambda = 13$ bits	$\lambda = 9$ bits	$\mu = 4$ bits = 1 hex ψηφίο
ετικέτα	Διεύθυνση συνόλου	Διεύθυνση λέξης μέσα στο πλαίσιο/μπλοκ

$\nu - \lambda$ (13 bits)	λ (9 bits)	μ (4 bits)
α. 3FF438A = 00111111111010	000111000	1010
β. 3FF2381 = 00111111111001	000111000	0001
γ. 3FF6384 = 00111111111011	000111000	0100

Μεθοδολογία: Ξεκινάμε ελέγχοντας **ποιες από τις παραπάνω διευθύνσεις έχουν το ίδιο λ (διεύθυνση συνόλου)**.

Αντές είναι οι διευθύνσεις (α), (β), (γ). Στη συνέχεια ελέγχουμε **αν αυτές έχουν ίδια ετικέτα (ν – λ)**. Καμία από αυτές δεν έχει την ίδια ετικέτα. Από τη στιγμή που έχουν διαφορετική ετικέτα, σημαίνει ότι αναφέρονται σε διαφορετικά μπλοκ, τα οποία μετακινούνται στο ίδιο σύνολο. Επειδή όμως το σύνολο αυτό περιέχει μόνο δύο πλαίσια, σημαίνει ότι κάθε φορά μπορούν να βρίσκονται στην κρυφή μνήμη μόνο δύο από τις τρεις συνολικά διευθύνσεις.

Παρατήρηση 1: Αν και οι τρείς διευθύνσεις **είχαν και ίδια ετικέτα ν-λ**, εκτός από το ίδιο λ που έχουν, τότε θα πήγαιναν και οι τρεις μαζί στο ίδιο πλαίσιο του συνόλου με διεύθυνση 000111000 και συνεπώς και οι τρεις θα μπορούσαν να είναι ταυτόχρονα στην κρυφή μνήμη.

Παρατήρηση 2: Αν και οι δύο από τις τρεις διευθύνσεις **είχαν και ίδια ετικέτα ν-λ**, εκτός από το ίδιο λ που έχουν, τότε θα πήγαιναν οι δύο από αυτές μαζί στο ίδιο πλαίσιο του συνόλου με διεύθυνση 000111000 και στο άλλο πλαίσιο του ίδιου συνόλου θα μεταφερόταν η τρίτη διεύθυνση, συνεπώς και οι τρεις θα μπορούσαν να είναι ταυτόχρονα στην κρυφή μνήμη.

Θέμα προόδου 2018 με κρυφή μνήμη

Θεωρείστε επεξεργαστή με αρτηρία διευθύνσεων και δεδομένων των 16 δυαδικών ψηφίων η κάθε μία και με κρυφή μνήμη εντολών και ξεχωριστή κρυφή μνήμη δεδομένων. Να συμπληρώσετε στον ακόλουθο πίνακα πόσες φορές γίνεται προσπέλαση της κρυφής μνήμης εντολών και πόσες φορές γίνεται προσπέλαση της κρυφής μνήμης δεδομένων κατά την προσκόμιση και την εκτέλεση κάθε εντολής. Κάθε θέση μνήμης είναι των 16 δυαδικών ψηφίων και οι καταχωρητές έχουν μήκος 16 δυαδικά ψηφία.

Εντολή	επεξήγηση	Πλήθος προσπελάσεων κρυφής μνήμης εντολών	Πλήθος προσπελάσεων κρυφής μνήμης δεδομένων
LOAD r1, #FD00 μήκος εντολής 4 ψηφιολέξεις	άμεσος τρόπος διευθυνσιοδότησης $r1 \leftarrow FD00$		
LOAD r1, (r2) μήκος εντολής 2 ψηφιολέξεις	Έμμεσος τρόπος διευθυνσιοδότησης με χρήση καταχωρητή $r1 \leftarrow M(r2)$		
LOAD r1, \$F30A μήκος εντολής 4 ψηφιολέξεις	Κατ' ευθείαν τρόπος διευθυνσιοδότησης $r1 \leftarrow (F30A)$		
ADD r1, r2 μήκος εντολής 2 ψηφιολέξεις	$r1 \leftarrow r1 + r2$		

Λύση

Εντολή	επεξήγηση	Πλήθος προσπελάσεων κρυφής μνήμης εντολών	Πλήθος προσπελάσεων κρυφής μνήμης δεδομένων
LOAD r1, #FD00 μήκος εντολής 4 ψηφιολέξεις	άμεσος τρόπος διευθυνσιοδότησης $r1 \leftarrow FD00$	2	0
LOAD r1, (r2) μήκος εντολής 2 ψηφιολέξεις	Έμμεσος τρόπος διευθυνσιοδότησης με χρήση καταχωρητή $r1 \leftarrow M(r2)$	1	1
LOAD r1, \$F30A μήκος εντολής 4 ψηφιολέξεις	Κατ' ευθείαν τρόπος διευθυνσιοδότησης $r1 \leftarrow (F30A)$	2	1
ADD r1, r2 μήκος εντολής 2 ψηφιολέξεις	$r1 \leftarrow r1 + r2$	1	0

Θέμα προόδου 2018 με κρυφή μνήμη

Θεωρείστε σύστημα μνήμης στο οποίο σε κάθε ψηφιολέξη (byte) αντιστοιχεί μια διεύθυνση. Θεωρήστε επίσης ότι το μέγεθος της λέξης είναι ίσο με μία ψηφιολέξη. Το σύστημα μνήμης περιλαμβάνει κρυφή μνήμη πρώτου επιπέδου των 32 Κψηφιολέξεων (Kbytes) και πλαίσια των 64 ψηφιολέξεων που προσπελαύνεται με φυσικές διεύθυνσεις. Όσον αφορά την οργάνωση εξετάστε δύο περιπτώσεις:

i. Μονοσήμαντης απεικόνισης

ii. 2-τρόπων συνόλου συσχέτισης

α. Για κάθε μία των ανωτέρω περιπτώσεων να δώσετε τα πεδία από τα οποία θεωρούμε ότι αποτελείται η διεύθυνση του επεξεργαστή για την προσπέλαση της κρυφής μνήμη και να δηλώσετε το ρόλο κάθε πεδίου.

β. Ποια από τα περιεχόμενα των θέσεων μνήμης με φυσικές διεύθυνσεις A: 00000077070(8), B: 00000037701(8),

C: 00000377053(8), D: 00000037707(8), E: 00000240137(8), F: 00000177052(8), G: 00000240063(8) είναι δυνατόν να βρίσκονται ταυτόχρονα στην κρυφή μνήμη σε κάθε μία από τις περιπτώσεις i. ii; Δικαιολογήστε την απάντησή σας.

Λύση

Αφού κάθε πλαίσιο έχει $64 = 2^6$ λέξεις, τα 6 λιγότερο σημαντικά δυαδικά ψηφία της διεύθυνσης θα καθορίζουν τη διεύθυνση της λέξης μέσα στο μπλοκ της κύριας μνήμης και στο πλαίσιο της κρυφής μνήμης, δημΠ. Αφού η κρυφή μνήμη έχει 32 Κλέξεις $= 25210$ λέξεις $= 2^{15}$ λέξεις και κάθε πλαίσιο $64 = 2^6$ λέξεις, η κρυφή μνήμη θα έχει $2^{15}/2^6$ πλαίσια $= 2^9$ πλαίσια.

a.i. Οργάνωση μονοσήμαντης απεικόνισης

Αφού η κρυφή μνήμη έχει 2^9 πλαίσια, τα αμέσως επόμενα (μετά το πεδίο δημΠ) 9 λιγότερο σημαντικά δυαδικά ψηφία της διεύθυνσης θα καθορίζουν τη διεύθυνση του πλαισίου στην κρυφή μνήμη. Τα υπόλοιπα δυαδικά ψηφία της διεύθυνσης αποτελούν την ετικέτα. 17 δυαδικά ψηφία

17 δυαδικά ψηφία	9 δυαδικά ψηφία	6 δυαδικά ψηφία
6 ψηφία στο οκταδικό	3 ψηφία στο οκταδικό	2 ψηφία στο οκταδικό
Ετικέτα	Διεύθυνση πλαισίου	Διεύθυνση της λέξης μέσα στο πλαίσιο
Διεύθυνση μπλοκ στην κύρια μνήμη		Διεύθυνση της λέξης μέσα στο μπλοκ

Αριθμός αναφοράς	Διεύθυνση θέσης μνήμης στο οκταδικό	Διεύθυνση μπλοκ στην κύρια μνήμη		Διεύθυνση της λέξης μέσα στο μπλοκ
		Ετικέτα	Διεύθυνση πλαισίου	
		17 δυαδικά ψηφία	9 δυαδικά ψηφία	6 δυαδικά ψηφία
A	00000037707	00-000-000-000-000	011-111-111	000-111
B	00000037701	00-000-000-000-000	011-111-111	000-001
C	00000077070	00-000-000-000-000	111-111-000	111-000
D	00000177052	00-000-000-000-001	111-111-000	101-010
E	00000377053	00-000-000-000-011	111-111-000	101-011
F	00000240063	00-000-000-000-010	100-000-000	110-011
G	00000240137	00-000-000-000-010	100-000-001	011-111



Εναλλακτικά

		6 ψηφία στο οκταδικό	3 ψηφία στο οκταδικό	2 ψηφίο στο οκταδικό
A	00000037707	000000	377	07
B	00000037701	000000	377	01
C	00000077070	000000	770	70
D	00000177052	000001	770	52
E	00000377053	000003	770	53
F	00000240063	000002	400	63
G	00000240137	000002	401	37

Οι διευθύνσεις Α και Β ανήκουν στο ίδιο μπλοκ (έχουν την ίδια διεύθυνση μπλοκ ή ισοδύναμα έχουν ίδια ετικέτα και ίδια διεύθυνση πλαισίου), επομένως όταν βρίσκονται τα περιεχόμενα της μίας διεύθυνσης στην κρυφή μνήμη θα βρίσκονται και της άλλης.

Οι διευθύνσεις C, D και E ανήκουν σε διαφορετικά μπλοκ της κύριας μνήμης που αντιστοιχούν στο ίδιο πλαίσιο της κρυφής μνήμης με διεύθυνση 770(8), επομένως τα περιεχόμενά τους δεν μπορούν να βρίσκονται ταυτόχρονα στην κρυφή μνήμη.

Οι διευθύνσεις F και G μεταξύ τους και σε σχέση με όλες τις άλλες διευθύνσεις ανήκουν σε διαφορετικά μπλοκ της κύριας μνήμης που αντιστοιχούν σε διαφορετικά πλαίσια της κρυφής μνήμης επομένως τα περιεχόμενά τους μπορούν να βρίσκονται ταυτόχρονα στην κρυφή μνήμη. Άρα στην κρυφή μνήμη μπορούν να βρίσκονται ταυτόχρονα τα περιεχόμενα των θέσεων της κύριας μνήμης με διευθύνσεις A, B, F, G και τα περιεχόμενα μίας εκ των C, D, E.

a.ii. Οργάνωση 2-τρόπων συνόλου συσχέτισης.

Η κρυφή μνήμη έχει 29 πλαίσια και 2 πλαίσια ανά σύνολο, επομένως έχουμε $29 / 2 = 28$ σύνολα, οπότε τα αμέσως επόμενα (μετά το πεδίο δμΠ) 8 λιγότερο σημαντικά δυαδικά ψηφία της διεύθυνσης θα καθορίζουν τη διεύθυνση του συνόλου στην κρυφή μνήμη. Τα υπόλοιπα δυαδικά ψηφία της διεύθυνσης αποτελούν την ετικέτα.

18 δυαδικά ψηφία	8 δυαδικά ψηφία	6 δυαδικά ψηφία
Ετικέτα	Διεύθυνση πλαισίου	Διεύθυνση της λέξης μέσα στο πλαίσιο
Διεύθυνση μπλοκ στην κύρια μνήμη		Διεύθυνση της λέξης μέσα στο μπλοκ

Αριθμός αναφοράς	Διεύθυνση θέσης μνήμης	Διεύθυνση μπλοκ στην κύρια μνήμη		Διεύθυνση της λέξης μέσα στο μπλοκ
		Ετικέτα	Διεύθυνση πλαισίου	Διεύθυνση της λέξης μέσα στο πλαίσιο
	18 δυαδικά ψηφία	8 δυαδικά ψηφία	6 δυαδικά ψηφία	
A	00000037707	000000000000000000-0	11-111-111	000-111
B	00000037701	000000000000000000-0	11-111-111	000-001
C	00000077070	000000000000000000-1	11-111-000	111-000
D	00000177052	000000000000000001-1	11-111-000	101-010
E	00000377053	000000000000000011-1	11-111-000	101-011
F	00000240063	000000000000000010-1	00-000-000	110-011
G	00000240137	000000000000000010-1	00-000-001	011-111

Οι διευθύνσεις Α και Β ανήκουν στο ίδιο μπλοκ (έχουν την ίδια διεύθυνση μπλοκ ή ισοδύναμα έχουν ίδια ετικέτα και ίδια διεύθυνση συνόλου), επομένως όταν βρίσκονται τα περιεχόμενα της μίας διεύθυνσης στην κρυφή μνήμη θα βρίσκονται και της άλλης.

Οι διευθύνσεις C, D και E ανήκουν σε διαφορετικά μπλοκ της κύριας μνήμης που αντιστοιχούν στο ίδιο σύνολο της κρυφής μνήμης με διεύθυνση 11-111-000(2), επομένως τα περιεχόμενα μόνο δύο εξ' αυτών μπορούν να βρίσκονται ταυτόχρονα στην



κρυφή μνήμη. Οι διευθύνσεις F και G μεταξύ τους και σε σχέση με όλες τις άλλες διευθύνσεις ανήκουν σε διαφορετικά μπλοκ της κύριας μνήμης που αντιστοιχούν σε διαφορετικά σύνολα της κρυφής μνήμης επομένως τα περιεχόμενά τους μπορούν να βρίσκονται ταυτόχρονα στην κρυφή μνήμη. Άρα στην κρυφή μνήμη μπορούν να βρίσκονται ταυτόχρονα τα περιεχόμενα των θέσεων της κύριας μνήμης με διευθύνσεις A, B, F, G και τα περιεχόμενα δύο εκ των C, D, E.

Θέμα Σεπτεμβρίου 2021 με κρυφή μνήμη

Θεωρήστε κρυφή μνήμη με χωρητικότητα 512 Bytes προσπελάσιμη με φυσικές διευθύνσεις των 32 δυαδικών ψηφίων και πλαίσιο των 16 bytes. Σε κάθε byte του συστήματος μνήμης αντιστοιχεί μία φυσική διεύθυνση. Δίνονται οι διευθύνσεις

- α. FF697269
- β. FF697263
- γ. FF697E68
- δ. FF697A68

α. Οργάνωση μονοσήμαντης απεικόνισης

Να δώσετε το μέγεθος του πεδίου "διεύθυνση μέσα στο Πλαίσιο δμπ" σε δυαδικά ψηφία (καθαρός αριθμός): .

Να δώσετε το μέγεθος του πεδίου "διεύθυνση πλαισίου" σε δυαδικά ψηφία (καθαρός αριθμός): .

Εάν τα περιεχόμενα των διευθύνσεων α και β είναι δυνατόν να βρίσκονται ταυτόχρονα στην κρυφή μνήμη γράψτε YES διαφορετικά γράψτε NO: .

Εάν τα περιεχόμενα των διευθύνσεων γ και δ είναι δυνατόν να βρίσκονται ταυτόχρονα στην κρυφή μνήμη γράψτε YES διαφορετικά γράψτε NO: .

Εάν τα περιεχόμενα των διευθύνσεων α, γ και δ είναι δυνατόν να βρίσκονται ταυτόχρονα στην κρυφή μνήμη γράψτε YES διαφορετικά γράψτε NO: .

β. Οργάνωση 2-τρόπων συνόλου συσχέτισης

Να δώσετε το μέγεθος του πεδίου "διεύθυνση μέσα στο Πλαίσιο δμπ" σε δυαδικά ψηφία (καθαρός αριθμός): .

Να δώσετε το μέγεθος του πεδίου "διεύθυνση συνόλου" σε δυαδικά ψηφία (καθαρός αριθμός): .

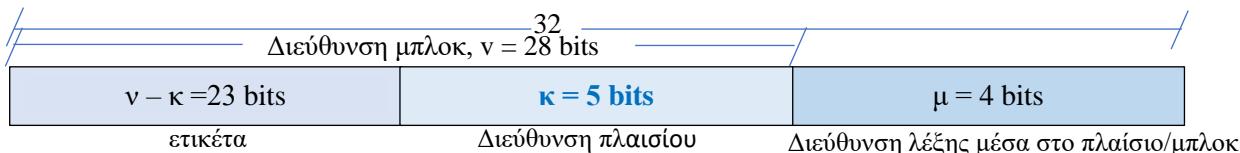
Εάν τα περιεχόμενα των διευθύνσεων α, β και δ είναι δυνατόν να βρίσκονται ταυτόχρονα στην κρυφή μνήμη γράψτε YES διαφορετικά γράψτε NO: .

Εάν τα περιεχόμενα των διευθύνσεων α, β, γ και δ είναι δυνατόν να βρίσκονται ταυτόχρονα στην κρυφή μνήμη γράψτε YES διαφορετικά γράψτε NO: .

Λύση

Κάθε διεύθυνση αποτελείται από 8 δεκαεξαδικά ψηφία που σημαίνει ότι έχουμε $8 \times 4\text{ bits} = 32\text{ bits}$ ως μήκος διεύθυνσης. Αυτό επαληθεύεται και από την εκφώνηση που αναφέρει φυσικές διευθύνσεις των 32 δυαδικών ψηφίων.

α) Οργάνωση μονοσήμαντης απεικόνισης



1 πλαίσιο = 16 bytes = 16 λέξεις ($\theta \cdot \mu$), επομένως το $2^\mu = 16 \Rightarrow \mu = 4\text{ bits}$.

Η διεύθυνση πλαισίου υπολογίζεται από τη χωρητικότητα της κρυφής μνήμης που δίνεται, ως εξής:

$$1 \text{ πλ. } 16 \text{ λέξεις} = 16 \text{ bytes}$$

$$x; \quad 512 \text{ bytes} \quad x = 2^9 / 2^4 = 2^5 \text{ πλ.} = 2^\kappa \Rightarrow \kappa = 5$$

Άρα η διεύθυνση μέσα στο πλαίσιο = $\mu = 4$ και η διεύθυνση πλαισίου = $\kappa = 5$.

Η διεύθυνση $\alpha = \text{FF697269} = 1111\ 1111\ 0110\ 1001\ 0111\ 001\ 00110\ 1001$

Η διεύθυνση $\beta = \text{FF697263} = 1111\ 1111\ 0110\ 1001\ 0111\ 001\ 00110\ 0011$

Πρόκειται για **ίδια μπλοκ** (έχουν ίδια ετικέτα $v - \kappa$ και ίδια διεύθυνση πλαισίου κ , άρα έχουν ίδιο v), τα οποία μεταφέρονται στο **ίδιο πλαίσιο** (έχουν ίδιο κ) → μπορούν **καιχρέπει** να είναι ταυτόχρονα στην κρυφή μνήμη → YES.

Η διεύθυνση $\gamma = \text{FF697E68} = 1111\ 1111\ 0110\ 1001\ 0111\ 111\ 00110\ 1000$

Η διεύθυνση $\delta = \text{FF697A68} = 1111\ 1111\ 0110\ 1001\ 0111\ 101\ 00110\ 1000$

Πρόκειται για **διαφορετικά μπλοκ** (έχουν διαφορετική ετικέτα $v - \kappa$, άρα και διαφορετική διεύθυνση μπλοκ v), τα οποία μεταφέρονται στο **ίδιο πλαίσιο** (έχουν ίδιο κ) → δεν μπορούν να είναι ταυτόχρονα στην κρυφή μνήμη → NO.



Η διεύθυνση $\alpha = \text{FF697269} = 1111\ 1111\ 0110\ 1001\ 0111\ 001\ 0\ 0110\ 1001$

Η διεύθυνση $\gamma = \text{FF697E68} = 1111\ 1111\ 0110\ 1001\ 0111\ 111\ 0\ 0110\ 1000$

Η διεύθυνση $\delta = \text{FF697A68} = 1111\ 1111\ 0110\ 1001\ 0111\ 101\ 0\ 0110\ 1000$

Πρόκειται για **διαφορετικά μπλοκ** (έχουν διαφορετική ετικέτα $v - \kappa$, άρα και διαφορετική διεύθυνση μπλοκ v), τα οποία μεταφέρονται στο **ίδιο πλαίσιο** (έχουν ίδιο κ) → δεν μπορούν να είναι ταυτόχρονα στην κρυφή μνήμη → **NO**.

β) Οργάνωση 2-τρόπων συνόλου συσχέτισης

32		
ετικέτα	Διεύθυνση μπλοκ, $v = 28$ bits	Διεύθυνση λέξης μέσα στο πλαίσιο/μπλοκ
$v - \lambda = 24$ bits = 6 Hex	$\lambda = 4$ bits = 1 Hex	$\mu = 4$ bits = 1 Hex

1 πλαίσιο = 16 bytes = 16 λέξεις ($\theta.\mu.$), επομένως το $2^\mu = 16 \Rightarrow \mu = 4$ bits.

1 σύνολο 2 πλαίσια

$$x; \quad 2^5 \text{ πλαίσια} \quad x = 2^5/2 = 2^4 \text{ σύνολα} = 2^\lambda \Rightarrow \lambda = 4$$

Παρατηρούμε ότι τώρα όλα τα μεγέθη των πεδίων της διεύθυνσης είναι πολλαπλάσια του 4

Η διεύθυνση $\alpha = \text{FF6972}\ 6\ 9$

Η διεύθυνση $\beta = \text{FF6972}\ 6\ 3$

Η διεύθυνση $\delta = \text{FF697A}\ 6\ 8$

Παρατηρούμε ότι και οι τρεις διευθύνσεις μεταφέρονται στο ίδιο σύνολο που είναι το Σ6, και οι δύο πρώτες διευθύνσεις έχουν ίδια ετικέτα $v - \lambda$, αλλά και ίδιο λ , επομένως και ίδια διεύθυνση μπλοκ v . Άρα, πρόκειται για δύο διευθύνσεις που αναφέρονται στο ίδιο μπλοκ που μεταφέρεται στο ίδιο σύνολο, επομένως οι δύο πρώτες διευθύνσεις μπορούν και πρέπει να είναι ταυτόχρονα στην κρυφή μνήμη και μάλιστα πηγαίνουν και οι δύο στο Π0 του Σ6. Η τρίτη διεύθυνση αναφέρεται σε διαφορετικό μπλοκ (έχει διαφορετική ετικέτα $v - \lambda$) που μεταφέρεται επίσης στο Σ6 και καταλαμβάνει το Π1 του Σ6. Άρα μπορούν και οι τρεις να είναι ταυτόχρονα στην κρυφή μνήμη → **YES**.

Η διεύθυνση $\alpha = \text{FF6972}\ 6\ 9$

Η διεύθυνση $\beta = \text{FF6972}\ 6\ 3$

Η διεύθυνση $\gamma = \text{FF697E}\ 6\ 8$

Η διεύθυνση $\delta = \text{FF697A}\ 6\ 8$

Παρατηρούμε ότι και οι τέσσερις διευθύνσεις μεταφέρονται στο ίδιο σύνολο που είναι το Σ6, και μάλιστα οι δύο πρώτες διευθύνσεις έχουν ίδια ετικέτα $v - \lambda$, αλλά και ίδιο λ , επομένως και ίδια διεύθυνση μπλοκ v . Άρα, πρόκειται για δύο διευθύνσεις που αναφέρονται στο ίδιο μπλοκ που μεταφέρεται στο ίδιο σύνολο, επομένως οι δύο πρώτες διευθύνσεις μπορούν και πρέπει να είναι ταυτόχρονα στην κρυφή μνήμη και μάλιστα πηγαίνουν και οι δύο στο Π0 του Σ6. Η τρίτη διεύθυνση αναφέρεται σε διαφορετικό μπλοκ (έχει διαφορετική ετικέτα $v - \lambda$) που μεταφέρεται επίσης στο Σ6 και καταλαμβάνει το Π1 του Σ6. Η τέταρτη διεύθυνση αναφέρεται επίσης σε διαφορετικό μπλοκ (έχει διαφορετική ετικέτα $v - \lambda$) που μεταφέρεται επίσης στο Σ6 και θα πρέπει να αντικαταστήσει ένα από τα πλαίσια του Σ6. Άρα δεν μπορούν και οι τέσσερις να είναι ταυτόχρονα στην κρυφή μνήμη → **NO**.

Σημείωση: κάθε φορά μπορούν να είναι ταυτόχρονα στην κρυφή μνήμη οι $\alpha - \beta$ με μια εκ' των γ, δ ή εναλλακτικά οι γ, δ χωρίς το ζεύγος $\alpha - \beta$.

Θέμα Σεπτεμβρίου 2020 με κρυφή μνήμη

Θεωρείστε επεξεργαστή που διαθέτει κρυφή μνήμη εντολών. Σε κάθε ψηφιολέξη (byte) αντιστοιχεί μια διεύθυνση. Για την προσπέλαση της κρυφής μνήμης εντολών θεωρούμε ότι η διεύθυνση που παράγει ο επεξεργαστής αποτελείται από τρία πεδία. Το πεδίο1 αποτελείται από τα λιγότερο σημαντικά δυαδικά ψηφία της διεύθυνσης. Το πεδίο2 αποτελείται από τα αμέσως σημαντικότερα δυαδικά ψηφία της διεύθυνσης και το πεδίο3 αποτελείται από τα πιο σημαντικά δυαδικά ψηφία της διεύθυνσης που παράγει ο επεξεργαστής. Να υπολογίσετε: α. το μέγεθος του πλαισίου σε bytes χρησιμοποιώντας τρία ψηφία, πχ. 022. β. τη χωρητικότητα της κρυφής μνήμης σε Kbytes ή Mbytes δίνοντας ένα τετραψήφιο ακέραιο αριθμό και το K ή το M ανάλογα. πχ. 0012K ή 0133M. Εάν η χωρητικότητα μπορεί να δοθεί σε Mbytes, πρέπει να δοθεί υποχρεωτικά σε Mbytes, πχ. το 1024K πρέπει να δοθεί ως 1M. γ. Το πλήθος των bits που απαιτούνται για την αποθήκευση των ετικετών χρησιμοποιώντας οκτώ ψηφία, πχ. 00293940.

Δίνονται:

Οργάνωση κρυφής μνήμης εντολών: 2-τρόπων συνόλου συσχέτισης.

και εύρος πεδίων

πεδίο1: 7 bits

πεδίο2: 11 bits

πεδίο3: 14 bits

Μέγεθος του πλαισίου σε bytes = .

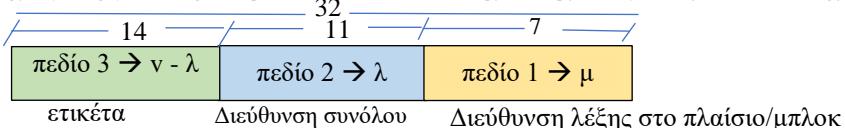
Χωρητικότητα της κρυφής μνήμης σε Kbytes ή Mbytes = .

Πλήθος των bits που απαιτούνται για την αποθήκευση των ετικετών = .

Αύση

Γνωρίζουμε από την εκφώνηση της άσκησης ότι το «πεδίο 1» περιέχει τα λιγότερο σημαντικά δυαδικά ψηφία της διεύθυνσης, άρα είναι το **τελευταίο πεδίο** της διεύθυνσης, δηλαδή το μ . Το «πεδίο 2» αποτελείται από τα αμέσως επόμενα πιο σημαντικά bits της διεύθυνσης, επομένως είναι το **μεσαίο πεδίο** της διεύθυνσης, δηλαδή το λ . Τέλος, το «πεδίο 3» αποτελείται από τα πιο σημαντικά δυαδικά ψηφία της διεύθυνσης, άρα είναι το πιο αριστερό πεδίο της διεύθυνσης, δηλ. είναι η ετικέτα $v - \lambda$.

Έχουμε οργάνωση **2- τρόπων συνόλου συσχέτισης**, επομένως τα πεδία της διεύθυνσης είναι τα εξής:



- Μέγεθος πλαισίου = 2^7 λέξεις ($\theta \cdot \mu$) = 2^7 bytes (λόγω της οργάνωσης 1byte/ $\theta \cdot \mu$) = 128 bytes → 128

- Χωρητικότητα κρυφής μνήμης = 2^{11} σύνολα $\times 2^1 \frac{\text{πλαισία}}{\text{σύνολο}} \times 2^7 \frac{\text{λέξεις}}{\text{πλαισίο}} = 2^{19}$ λέξεις ($\theta \cdot \mu$) = 2^{19} ψηφιολέξεις (bytes) = $2^{10} 2^9 = 512K$ → 0512K

- Χωρητικότητα αποθήκευσης ετικετών: 2^{11} σύνολα $\times 2^1 \frac{\text{πλ.}}{\text{σύνολο}} \times 14 \frac{\text{bits}}{\text{πλ.}} = 2^{11} \times 2^1 \times 14 = 2^{12} \times 14 \text{ bits} = 4096 \times 14 \text{ bits} = 57344 \text{ bits} = \rightarrow 00057344$.

Παρατήρηση 1: όταν η εκφώνηση ζητά τον υπολογισμό της χωρητικότητας της κρυφής μνήμης, θα πρέπει στην περίπτωση της **μονοσήμαντης απεικόνισης** να πολλαπλασιάσουμε τον αριθμό των πλαισίων με το μέγεθος του κάθε πλαισίου, δηλ. αριθμός πλαισίων \times μέγεθος πλαισίου. Στην περίπτωση της **τ - τρόπων συνόλου συσχέτισης** κρυφής μνήμης, θα πρέπει να πολλαπλασιάσουμε τον αριθμό των συνόλων με τον αριθμό των πλαισίων/σύνολο και με το μέγεθος του κάθε πλαισίου, δηλ. αριθμός συνόλων \times αριθμός $\frac{\text{πλαισίων}}{\text{σύνολο}} \times$ μέγεθος πλαισίου.

Παρατήρηση 2: όταν η εκφώνηση ζητά το συνολικό πλήθος των δυαδικών ψηφίων που αποθηκεύονται στην κρυφή μνήμη, θα πρέπει στην περίπτωση της **μονοσήμαντης απεικόνισης** να πολλαπλασιάσουμε τον αριθμό των πλαισίων με το μέγεθος του κάθε πλαισίου + μέγεθος ετικέτας + μέγεθος bit εγκυρότητας, δηλ. αριθμός πλαισίων \times (μέγεθος πλαισίου + μέγεθος ετικέτας + μέγεθος bit εγκυρότητας). Στην περίπτωση της **τ - τρόπων συνόλου συσχέτισης** κρυφής μνήμης θα πρέπει να πολλαπλασιάσουμε τον αριθμό των συνόλων με τον αριθμό των πλαισίων/σύνολο και το μέγεθος του κάθε πλαισίου + μέγεθος ετικέτας + μέγεθος bit εγκυρότητας, δηλ. αριθμός συνόλων \times αριθμός $\frac{\text{πλαισίων}}{\text{σύνολο}} \times$ (μέγεθος πλαισίου + μέγεθος ετικέτας + μέγεθος bit εγκυρότητας)

Παρατήρηση 3: όταν η εκφώνηση ζητά το πλήθος bits που απαιτούνται για την αποθήκευση των ετικετών, θα πρέπει στην περίπτωση της **μονοσήμαντης απεικόνισης** να πολλαπλασιάσουμε τον αριθμό των πλαισίων με το μέγεθος της ετικέτας, δηλαδή αριθμός πλαισίων \times μέγεθος ετικέτας ενώ στην περίπτωση της **τ - τρόπων συνόλου συσχέτισης** κρυφής μνήμης θα πρέπει να πολλαπλασιάσουμε τον αριθμό των συνόλων με τον αριθμό των πλαισίων/σύνολο και το μέγεθος ετικέτας, δηλαδή αριθμός συνόλων \times αριθμός $\frac{\text{πλαισίων}}{\text{σύνολο}} \times$ μέγεθος ετικέτας.

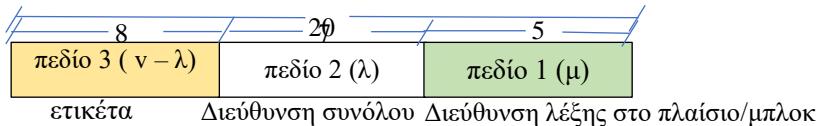
Θέμα Φεβρουαρίου 2021 με κρυφή μνήμη

Θεωρείστε επεξεργαστή που διαθέτει κρυφή μνήμη εντολών. **Σε κάθε ψηφιολέξη (byte) αντιστοιχεί μια διεύθυνση**. Για την προσπέλαση της κρυφής μνήμης εντολών θεωρούμε ότι η διεύθυνση που παράγει ο επεξεργαστής αποτελείται από τρία πεδία. Το πεδίο1 αποτελείται από τα λιγότερο σημαντικά δυαδικά ψηφία της διεύθυνσης. Το πεδίο2 αποτελείται από τα αμέσως σημαντικότερα δυαδικά ψηφία της διεύθυνσης και το πεδίο 3 αποτελείται από τα πιο σημαντικά δυαδικά ψηφία της διεύθυνσης που παράγει ο επεξεργαστής. Να υπολογίσετε: A, το μέγεθος του πλαισίου σε bytes χρησιμοποιώντας τρία ψηφία, πχ. [022] B. τη χωρητικότητα της κρυφής μνήμης σε Kbytes ή Mbytes δίνοντας ένα τετραψήφιο ακέραιο αριθμό και το K ή το M ανάλογα, πχ. [0012K] ή [0133M]. Εάν η χωρητικότητα μπορεί να δοθεί σε Mbytes, πρέπει να δοθεί υποχρεωτικά σε Mbytes, πχ. το 1024K πρέπει να δοθεί ως 1M.G. Το πλήθος των bits που απαιτούνται για την αποθήκευση των ετικετών χρησιμοποιώντας οκτώ ψηφία, πχ. [00293940].

Δίνονται οργάνωση κρυφής μνήμης εντολών: 4-τρόπων συνόλου συσχέτισης και εύρος πεδίων πεδίο1: 5 bits πεδίο2: 7 bits πεδίο3: 8 bits Μέγεθος του πλαισίου σε bytes = . Χωρητικότητα της κρυφής μνήμης σε KBytes ή Mbytes = . Πλήθος των bits που απαιτούνται για την αποθήκευση των ετικετών = .

Λύση

Έχουμε οργάνωση 4- τρόπων συνόλου συσχέτισης, επομένως τα πεδία της διεύθυνσης είναι τα εξής:



- Μέγεθος πλαισίου = 2^5 λέξεις (θ.μ.) = 2^5 bytes (λόγω της οργάνωσης 1 byte/θ.μ.) = 32 bytes → 032
- Χωρητικότητα κρυφής μνήμης = 2^7 σύνολα $\times 2^2 \frac{\text{πλαίσια}}{\text{σύνολο}} \times 2^5 \frac{\text{λέξεις}}{\text{πλαίσιο}} = 2^{14}$ λέξεις (θ.μ.) = 2^{14} ψηφιολέξεις (bytes) = $2^{10}2^4 = 16K$ → 0016K
- Χωρητικότητα αποθήκευσης ετικετών: 2^7 σύνολα $\times 4 \frac{\text{πλ.}}{\text{σύνολο}} \times 8 \frac{\text{bits}}{\text{πλ.}} = 2^7 \times 2^2 \times 2^3$ bits = 2^{12} bits = $2^2 \times 2^{10}$ bits = 4Kbits = 4 x 1024 bits = 4096 bits → 00004096.

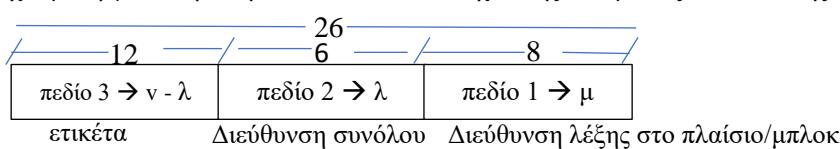
Παρατήρηση: όταν αναφέρεται στην εκφώνηση 1 – τρόπων συνόλου συσχέτισης, εννοεί μονοσήμαντη απεικόνιση.

Θέμα Σεπτεμβρίου 2020 με κρυφή μνήμη

Θεωρήστε επεξεργαστή που διαθέτει κρυφή μνήμη εντολών. Σε κάθε ψηφιολέξη (byte) αντιστοιχεί μια διεύθυνση. Για την προσπέλαση της κρυφής μνήμης εντολών θεωρούμε ότι η διεύθυνση που παράγει ο επεξεργαστής αποτελείται από τρία πεδία. Το πεδίο1 αποτελείται από τα λιγότερο σημαντικά δυαδικά ψηφία της διεύθυνσης. Το πεδίο2 αποτελείται από τα αμέσως σημαντικότερα δυαδικά ψηφία της διεύθυνσης και το πεδίο3 αποτελείται από τα πιο σημαντικά δυαδικά ψηφία της διεύθυνσης που παράγει ο επεξεργαστής. Να υπολογίσετε: Α. το μέγεθος του πλαισίου σε bytes χρησιμοποιώντας τρία ψηφία, πχ. [022] Β. τη χωρητικότητα της κρυφής μνήμης σε Kbytes ή Mbytes δίνοντας ένα τετραψήφιο ακέραιο αριθμό και το K ή το M ανάλογα, πχ. [0012K] ή [0133M]. Εάν η χωρητικότητα μπορεί να δοθεί σε Mbytes, πρέπει να δοθεί υποχρεωτικά σε Mbytes, πχ. το 1024K πρέπει να δοθεί ως 1M.G. Το πλήθος των bits που απαιτούνται για την αποθήκευση των ετικετών χρησιμοποιώντας οκτώ ψηφία, πχ. [00293940]. Δίνονται οργάνωση κρυφής μνήμης εντολών: 8-τρόπων συνόλου συσχέτισης και εύρος πεδίων πεδίο1: 8 bits πεδίο2: 6 bits πεδίο3: 12 bits Μέγεθος του πλαισίου σε bytes = . Χωρητικότητα της κρυφής μνήμης σε KBytes ή Mbytes = . Πλήθος των bits που απαιτούνται για την αποθήκευση των ετικετών = .

Λύση

Έχουμε οργάνωση 8- τρόπων συνόλου συσχέτισης, επομένως τα πεδία της διεύθυνσης είναι τα εξής:



- Μέγεθος πλαισίου = 2^8 λέξεις (θ.μ.) = 2^8 bytes (λόγω της οργάνωσης 1 byte/θ.μ.) = 256 bytes → 256
- Χωρητικότητα κρ. μν. = 2^6 σύνολα $\times 2^3 \frac{\text{πλαίσια}}{\text{σύνολο}} \times 2^8 \frac{\text{λέξεις}}{\text{πλαίσιο}} = 2^{17}$ λέξεις (θ.μ.) = 2^{17} ψηφιολέξεις (bytes) = $2^{10}2^7 = 128K$ → 0128K
- Χωρητικότητα αποθήκευσης ετικετών: 2^6 σύνολα $\times 8 \frac{\text{πλ.}}{\text{σύνολο}} \times 12 \frac{\text{bits}}{\text{πλ.}} = 2^6 \times 2^3 \times 12$ bits = $2^9 \times 12$ bits = $2^9 \times 2 \times 6$ bits = 6Kbits = 6 x 1024 bits = 6144 bits → 00006144.

Θέμα Σεπτεμβρίου 2020 με κρυφή μνήμη

Θεωρείστε επεξεργαστή που διαθέτει κρυφή μνήμη εντολών. Σε κάθε ψηφιολέξη (byte) αντιστοιχεί μια διεύθυνση. Για την προσπέλαση της κρυφής μνήμης εντολών θεωρούμε ότι η διεύθυνση που παράγει ο επεξεργαστής αποτελείται από τρία πεδία. Το πεδίο1 αποτελείται από τα λιγότερο σημαντικά δυαδικά ψηφία της διεύθυνσης. Το πεδίο2 αποτελείται από τα αμέσως σημαντικότερα δυαδικά ψηφία της διεύθυνσης και το πεδίο3 αποτελείται από τα πιο σημαντικά δυαδικά ψηφία της διεύθυνσης που παράγει ο επεξεργαστής. Να υπολογίσετε: α. το μέγεθος του πλαισίου σε bytes β. το πλήθος των πλαισίων και να δώσετε ένα καθαρό αριθμό χωρίς τελεία για να δηλώσετε χιλιάδες ή εκατομμύρια πχ. 49928466. γ. Το πλήθος των bits που απαιτούνται για την αποθήκευση των ετικετών και να δώσετε ένα καθαρό αριθμό χωρίς τελεία για να δηλώσετε χιλιάδες ή εκατομμύρια πχ. 98293940.



Δίνονται: Οργάνωση κρυφής μνήμης εντολών: 5-τρόπων συνόλου συσχέτισης και εύρος πεδίων

πεδίο1: 6 bits

πεδίο2: 12 bits

πεδίο3: 12 bits

Μέγεθος του πλαισίου σε bytes = .

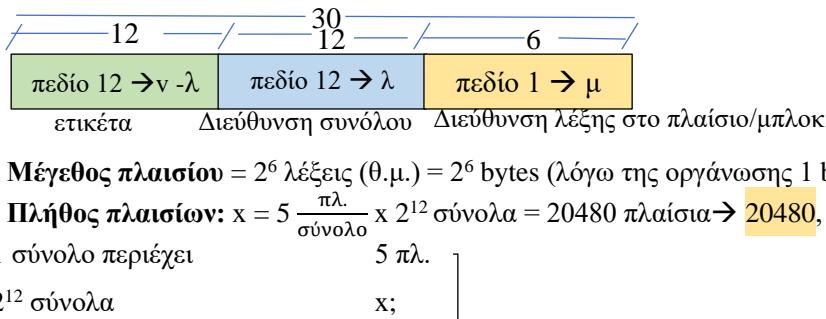
Πλήθος πλαισίων = .

Πλήθος των bits που απαιτούνται για την αποθήκευση των ετικετών = .

Λύση

Γνωρίζουμε από την εκφώνηση της άσκησης ότι το «πεδίο 1» περιέχει τα λιγότερο σημαντικά δυαδικά ψηφία της διεύθυνσης, άρα είναι το **τελευταίο πεδίο** της διεύθυνσης, δηλαδή το μ . Το «πεδίο 2» αποτελείται από τα αμέσως επόμενα πιο σημαντικά bits της διεύθυνσης, επομένως είναι το **μεσαίο πεδίο** της διεύθυνσης, δηλαδή το λ . Τέλος, το «πεδίο 3» αποτελείται από τα πιο σημαντικά δυαδικά ψηφία της διεύθυνσης, άρα είναι το πιο αριστερό πεδίο της διεύθυνσης, δηλ. είναι η ετικέτα $v - \lambda$.

Έχουμε οργάνωση **5- τρόπων συνόλου συσχέτισης**, επομένως τα πεδία της διεύθυνσης είναι τα εξής:



- **Μέγεθος πλαισίου** = 2^6 λέξεις ($\theta.\mu.$) = 2^6 bytes (λόγω της οργάνωσης 1 byte/ $\theta.\mu.$) = 64 bytes \rightarrow **64**

- **Πλήθος πλαισίων:** $x = 5 \frac{\pi\lambda}{\sigmaύνολο} \times 2^{12}$ σύνολα = 20480 πλαίσια \rightarrow **20480**, διότι:

$$\begin{aligned} &1 \text{ σύνολο περιέχει} && 5 \pi\lambda. \\ &2^{12} \text{ σύνολα} && x; \end{aligned}$$

- **Χωρητικότητα αποθήκευσης ετικετών:** 2^{12} σύνολα $\times 5 \frac{\pi\lambda}{\sigmaύνολο} \times 12 \frac{\text{bits}}{\pi\lambda} = 2^{12} \times 60 \text{ bits} = 245760 \text{ bits}$

Παρατήρηση 1: Αν η εκφώνηση ζητούσε και τη χωρητικότητα της κρυφής μνήμης, θα είχαμε:

- Χωρητικότητα κρυφής μνήμης = 2^{12} σύνολα $\times 5 \frac{\pi\lambda\text{σια}}{\sigmaύνολο} \times 2^6 \frac{\lambdaέξεις}{\pi\lambda\text{σια}} = 327680 \lambdaέξεις (\theta.\mu.) = 327680 \text{ bytes}$

Παρατήρηση 2: Αν η εκφώνηση ζητούσε και το **συνολικό πλήθος των δυαδικών ψηφίων που αποθηκεύονται στην κρυφή μνήμη**, θα είχαμε: Συνολικό πλήθος των δυαδικών ψηφίων κρυφής μνήμης = 2^{12} σύνολα $\times 5 \frac{\pi\lambda\text{σια}}{\sigmaύνολο} \times (64 \frac{\lambdaέξεις}{\pi\lambda\text{σια}} + 12 \frac{\text{bits}}{\pi\lambda\text{σια}} + \frac{\text{bit}}{\pi\lambda\text{σια}}) = 2^{12} \text{ σύνολα} \times 5 \frac{\pi\lambda\text{σια}}{\sigmaύνολο} \times (64 \frac{\text{bytes}}{\pi\lambda\text{σια}} + 12 \frac{\text{bits}}{\pi\lambda\text{σια}} + \frac{\text{bit}}{\pi\lambda\text{σια}}) = 2^{12} \text{ σύνολα} \times 5 \frac{\pi\lambda\text{σια}}{\sigmaύνολο} \times (64 \times 8 \frac{\text{bits}}{\pi\lambda\text{σια}} + 12 \frac{\text{bits}}{\pi\lambda\text{σια}} + \frac{\text{bit}}{\pi\lambda\text{σια}}) = 2^{12} \times 5 \times (64 \times 8 + 13) \text{ bits} = 2^{12} \times 5 \times 525 \text{ bits} = 10.752.000 \text{ bits.}$

Θέμα Σεπτεμβρίου 2021 με κρυφή μνήμη

Θεωρείστε κρυφή μνήμη με χωρητικότητα 512 Bytes προσπελάσιμη με φυσικές διευθύνσεις των 32 δυαδικών ψηφίων και πλαίσιο των 16 bytes. Σε κάθε byte του συστήματος μνήμης αντιστοιχεί μία φυσική διεύθυνση

Δίνονται οι διευθύνσεις

α. A379362A

β. A3793623

γ. A3793B26

δ. A3793926

α. Οργάνωση μονοσήμαντης απεικόνισης

Να δώσετε το μέγεθος του πεδίου "διεύθυνση μέσα στο Πλαίσιο δμπ" σε δυαδικά ψηφία (καθαρός αριθμός): .

Να δώσετε το μέγεθος του πεδίου "διεύθυνση πλαισίου" σε δυαδικά ψηφία (καθαρός αριθμός): .

Εάν τα περιεχόμενα των διευθύνσεων α και β είναι δυνατόν να βρίσκονται ταυτόχρονα στην κρυφή μνήμη γράψτε YES διαφορετικά γράψτε NO: .

Εάν τα περιεχόμενα των διευθύνσεων γ και δ είναι δυνατόν να βρίσκονται ταυτόχρονα στην κρυφή μνήμη γράψτε YES διαφορετικά γράψτε NO: .

Εάν τα περιεχόμενα των διευθύνσεων α, γ και δ είναι δυνατόν να βρίσκονται ταυτόχρονα στην κρυφή μνήμη γράψτε YES διαφορετικά γράψτε NO: .

β. Οργάνωση 2-τρόπων συνόλου συσχέτισης

Να δώσετε το μέγεθος του πεδίου "διεύθυνση μέσα στο Πλαισιο δμπ" σε δυαδικά ψηφία (καθαρός αριθμός):

Να δώσετε το μέγεθος του πεδίου "διεύθυνση συνόλου" σε δυαδικά ψηφία (καθαρός αριθμός):

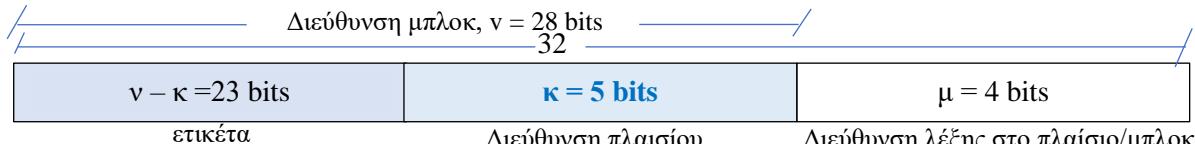
Εάν τα περιεχόμενα των διευθύνσεων α, β και δ είναι δυνατόν να βρίσκονται ταυτόχρονα στην κρυφή μνήμη γράψτε YES διαφορετικά γράψτε NO:

Εάν τα περιεχόμενα των διευθύνσεων α, β, γ και δ είναι δυνατόν να βρίσκονται ταυτόχρονα στην κρυφή μνήμη γράψτε YES διαφορετικά γράψτε NO:

Λύση

Κάθε διεύθυνση αποτελείται από 8 δεκαεξαδικά ψηφία που σημαίνει ότι έχουμε $8 \times 4\text{bits} = 32\text{ bits}$ ως μήκος διεύθυνσης. Αυτό εποληθεύεται και από την εκφώνηση που αναφέρει φυσικές διευθύνσεις των 32 δυαδικών ψηφίων.

α) Οργάνωση μονοσήμαντης απεικόνισης



1 πλαίσιο = 16 bytes = 16 λέξεις (θ.μ.), επομένως το $2^\mu = 16 \Rightarrow \mu = 4\text{ bits}$.

Η διεύθυνση πλαισίου υπολογίζεται από τη χωρητικότητα της κρυφής μνήμης που δίνεται, ως εξής:

$$\begin{array}{l} 1\text{ πλ. } 16\text{ λέξεις }= 16\text{ bytes} \\ x; \quad \quad \quad 512\text{ bytes} \end{array} \quad x = 2^9/2^4 = 2^5 \text{ πλ. } = 2^\kappa \Rightarrow \kappa = 5$$

Άρα η διεύθυνση μέσα στο πλαίσιο = $\mu = 4$ και η διεύθυνση πλαισίου = $\kappa = 5$.

Η διεύθυνση $\alpha = A379362A = 1010\ 0011\ 0111\ 1001\ 0011\ 011\ 0\ 0010\ 1010$

Η διεύθυνση $\beta = A3793623 = 1010\ 0011\ 0111\ 1001\ 0011\ 011\ 0\ 0010\ 0011$

Πρόκειται για **ίδια μπλοκ** (έχουν ίδια ετικέτα $v - \kappa$ και ίδια διεύθυνση πλαισίου κ , άρα έχουν ίδιο v), τα οποία μεταφέρονται στο **ίδιο πλαίσιο** (έχουν ίδιο κ) → μπορούν **και πρέπει** να είναι ταυτόχρονα στην κρυφή μνήμη → YES.

Η διεύθυνση $\gamma = A3793826 = 1010\ 0011\ 0111\ 1001\ 0011\ 100\ 00010\ 0110$

Η διεύθυνση $\delta = A3793926 = 1010\ 0011\ 0111\ 1001\ 0011\ 100\ 10010\ 0110$

Πρόκειται για **διαφορετικά μπλοκ** (έχουν ίδια ετικέτα $v - \kappa$, αλλά διαφορετική διεύθυνση μπλοκ v), τα οποία μεταφέρονται σε **διαφορετικά πλαίσια** (έχουν διαφορετικό κ) → μπορούν να είναι ταυτόχρονα στην κρυφή μνήμη → YES.

Η διεύθυνση $\alpha = A379362A = 1010\ 0011\ 0111\ 1001\ 0011\ 011\ 00010\ 1010$

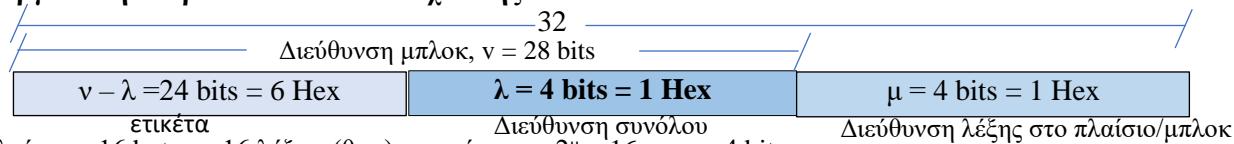
Η διεύθυνση $\gamma = A3793826 = 1010\ 0011\ 0111\ 1001\ 0011\ 100\ 00010\ 0110$

Η διεύθυνση $\delta = A3793926 = 1010\ 0011\ 0111\ 1001\ 0011\ 100\ 10010\ 0110$

Για α-γ: πρόκειται για **διαφορετικά μπλοκ** (έχουν διαφορετική ετικέτα $v - \kappa$, άρα και διαφορετική διεύθυνση μπλοκ v), τα οποία μεταφέρονται στο **ίδιο πλαίσιο** (έχουν ίδια κ) → δεν μπορούν να είναι ταυτόχρονα στην κρυφή μνήμη.

Για α-δ: πρόκειται για **διαφορετικά μπλοκ** (έχουν διαφορετική ετικέτα $v - \kappa$, άρα και διαφορετική διεύθυνση μπλοκ v), τα οποία μεταφέρονται σε **διαφορετικά πλαίσια** (έχουν διαφορετικό κ) → μπορούν να είναι ταυτόχρονα στην κρυφή μνήμη. Άρα, στην κρυφή μνήμη μπορούν να είναι ταυτόχρονα μόνο τα α, δ → NO.

β) Οργάνωση 2- τρόπων συνόλου συσχέτισης



1 πλαίσιο = 16 bytes = 16 λέξεις (θ.μ.), επομένως το $2^\mu = 16 \Rightarrow \mu = 4\text{ bits}$.

1 σύνολο 2 πλαίσια
x; 2^5 πλαίσια

$$x = 2^5/2 = 2^4 \text{ σύνολα} = 2^\lambda \Rightarrow \lambda = 4$$

Παρατηρούμε ότι τώρα όλα τα μεγέθη των πεδίων της διεύθυνσης είναι πολλαπλάσια του 4

Η διεύθυνση α = A37936 2 A

Η διεύθυνση β = A37936 2 3

Η διεύθυνση δ = A37939 2 6

Παρατηρούμε ότι και οι τρεις διευθύνσεις μεταφέρονται στο ίδιο σύνολο που είναι το Σ2, και οι δύο πρώτες διευθύνσεις έχουν ίδια ετικέτα $v - \lambda$, αλλά και ίδιο λ, επομένως και ίδια διεύθυνση μπλοκ ν. Άρα, πρόκειται για δύο διευθύνσεις που αναφέρονται στο ίδιο μπλοκ που μεταφέρεται στο ίδιο σύνολο, επομένως οι δύο πρώτες διευθύνσεις μπορούν και πρέπει να είναι ταυτόχρονα στην κρυφή μνήμη και μάλιστα πηγαίνουν και οι δύο στο Π0 του Σ2. Η τρίτη διεύθυνση αναφέρεται σε διαφορετικό μπλοκ (έχει διαφορετική ετικέτα $v - \lambda$) που μεταφέρεται επίσης στο Σ2 και καταλαμβάνει το Π1 του Σ2. Άρα μπορούν και οι τρεις να είναι ταυτόχρονα στην κρυφή μνήμη \rightarrow YES.

Η διεύθυνση α = A37936 2 A

Η διεύθυνση β = A37936 2 3

Η διεύθυνση γ = A37938 2 6

Η διεύθυνση δ = A37939 2 6

Παρατηρούμε ότι και οι τέσσερις διευθύνσεις μεταφέρονται στο ίδιο σύνολο που είναι το Σ2, και μάλιστα οι δύο πρώτες διευθύνσεις έχουν ίδια ετικέτα $v - \lambda$, αλλά και ίδιο λ, επομένως και ίδια διεύθυνση μπλοκ ν. Άρα, πρόκειται για δύο διευθύνσεις που αναφέρονται στο ίδιο μπλοκ που μεταφέρεται στο ίδιο σύνολο, επομένως οι δύο πρώτες διευθύνσεις μπορούν και πρέπει να είναι ταυτόχρονα στην κρυφή μνήμη και μάλιστα πηγαίνουν και οι δύο στο Π0 του Σ2. Η τρίτη διεύθυνση αναφέρεται σε διαφορετικό μπλοκ (έχει διαφορετική ετικέτα $v - \lambda$) που μεταφέρεται επίσης στο Σ2 και καταλαμβάνει το Π1 του Σ2. Η τέταρτη διεύθυνση αναφέρεται επίσης σε διαφορετικό μπλοκ (έχει διαφορετική ετικέτα $v - \lambda$) που μεταφέρεται επίσης στο Σ2 και θα πρέπει να αντικαταστήσει ένα από τα πλαίσια του Σ2. Άρα δεν μπορούν και οι τέσσερις να είναι ταυτόχρονα στην κρυφή μνήμη \rightarrow NO.

Σημείωση: κάθε φορά μπορούν να είναι ταυτόχρονα στην κρυφή μνήμη οι $\alpha - \beta$ με μια εκ' των γ, δ ή εναλλακτικά οι γ, δ χωρίς το ζεύγος $\alpha - \beta$.

Θέμα με κρυφή μνήμη

Θεωρήστε τρεις επεξεργαστές E1, E2 και E3 κάθε ένας με κρυφή μνήμη των 1024 θέσεων με μία ψηφιολέξη (byte) ανά θέση μνήμης και 8 ψηφιολέξεις ανά πλαίσιο και οργάνωση αντίστοιχα:

α. Μονοσήμαντης απεικόνισης (direct mapped)

β. 2 - τρόπων συνόλου συσχέτισης (2-way associative)

γ. 4 - τρόπων συνόλου συσχέτισης (4-way associative)

Θεωρείστε ότι και οι τρεις επεξεργαστές παράγουν διευθύνσεις των 20 δυαδικών ψηφίων και ότι κάθε θέση της κύριας μνήμης είναι της μιας ψηφιολέξης.

Για κάθε μία οργάνωση της κρυφής μνήμης να δώσετε:

i. Να δώσετε τα πεδία από τα οποία θεωρούμε ότι αποτελείται η διεύθυνση του επεξεργαστή για την προσπέλαση κάθε μιας των ανωτέρω κρυφών μνημάτων και τι δηλώνει κάθε πεδίο.

ii. Να αναφέρετε και να αιτιολογήσετε τα περιεχόμενα ποιων από τις διευθύνσεις FD504₍₁₆₎, FFD07₍₁₆₎, FD506₍₁₆₎, FD7FA₍₁₆₎ και EC107₍₁₆₎ μπορούν να βρίσκονται ταυτόχρονα στην κρυφή μνήμη για κάθε μια από τις περιπτώσεις α, β και γ.

Λύση

Μία κρυφή μνήμη έχει 1024 ψηφιολέξεις = 2^{10} ψηφιολέξεις

1 πλαίσιο έχει 8 ψηφιολέξεις = 2^3 ψηφιολέξεις

Επομένως κάθε κρυφή μνήμη έχει $2^{10}/2^3$ πλαίσια = 2^7 πλαίσια

Επεξεργαστής E1:

διεύθυνση μπλοκ				
10 δυαδικά ψηφία ετικέτα	7 δυαδικά ψηφία διεύθυνση πλαισίου		3 δυαδικά ψηφία διεύθυνση μέσα στο πλαίσιο, (δμΠ)	
α FD504	1111 1101 01		01 0000 0	100
β FFD07	1111 1111 11		01 0000 0	111
γ FD506	1111 1101 01		01 0000 0	110
δ FD7FA	1111 1101 01		11 1111 1	010

ε	EC107	1110 1100 00	01 0000 0	111
---	-------	--------------	-----------	-----

Οι διευθύνσεις α και γ έχουν την ίδια ετικέτα και την ίδια διεύθυνση πλαισίου, δηλαδή έχουν την ίδια διεύθυνση μπλοκ, επομένως όταν βρίσκονται τα περιεχόμενα της μίας εκ των α και γ θα βρίσκονται και της άλλης. Στη συνέχεια το ζεύγος α και γ θα συμβολίζουμε ως α-γ.

Οι διευθύνσεις των α-γ, β και ε έχουν διαφορετική ετικέτα, άρα ανήκουν σε διαφορετικά μπλοκ, αλλά την ίδια διεύθυνση πλαισίου, επομένως **δεν** μπορούν να βρίσκονται ταυτόχρονα στην κρυφή μνήμη. Κάθε χρονική περίοδο τα περιεχόμενα μόνο μίας εξ αυτών μπορεί να βρίσκεται στην κρυφή μνήμη.

Η διεύθυνση δ έχει διαφορετική ετικέτα και διεύθυνση πλαισίου με όλες τις άλλες διευθύνσεις α-γ, β και ε, **άρα τα περιεχόμενά της μπορούν να βρίσκονται ταυτόχρονα στην κρυφή μνήμη με τα περιεχόμενα οποιαδήποτε από τις α-γ, β και ε.**

Επεξεργαστής 2:

Έχουμε 2^7 πλαίσια και 2 πλαίσια ανά σύνολο, άρα έχουμε $2^7 / 2$ σύνολα = 2^6 σύνολα = 2^λ . Επομένως

διεύθυνση μπλοκ				
11 δυαδικά ψηφία		6 δυαδικά ψηφία	3 δυαδικά ψηφία	
ετικέτα		διεύθυνση συνόλου	διεύθυνση μέσα στο πλαίσιο, (δμΠ)	
	Διεύθυνση στο δεκαεξαδικό	Ετικέτα ($v - \lambda$)	Διεύθυνση συνόλου (λ)	Διεύθυνση μέσα στο πλαίσιο (μ)
α	FD504	1111 1101 010	1 0000 0	100
β	FFD07	1111 1111 110	1 0000 0	111
γ	FD506	1111 1101 010	1 0000 0	110
δ	FD7FA	1111 1101 011	1 1111 1	010
ε	EC107	1110 1100 000	1 0000 0	111

Οι διευθύνσεις α και γ έχουν την ίδια ετικέτα και την ίδια διεύθυνση συνόλου, δηλαδή έχουν την ίδια διεύθυνση μπλοκ, επομένως όταν βρίσκονται τα περιεχόμενα της μίας εκ των α και γ θα βρίσκονται και της άλλης. Στη συνέχεια το ζεύγος α και γ θα συμβολίζουμε ως α-γ.

Οι διευθύνσεις των α-γ, β και ε έχουν διαφορετική ετικέτα, άρα ανήκουν σε διαφορετικά μπλοκ, αλλά την ίδια διεύθυνση συνόλου. Λαμβάνοντας υπόψη ότι κάθε σύνολο έχει μόνο δύο πλαίσια, συμπεραίνουμε ότι τα περιεχόμενα **μόνο δύο εξ αυτών μπορούν να βρίσκονται ταυτόχρονα στην κρυφή μνήμη**.

Η διεύθυνση δ έχει διαφορετική ετικέτα και διεύθυνση πλαισίου με όλες τις άλλες διευθύνσεις α-γ, β και ε, **επομένως τα περιεχόμενά της μπορούν να είναι ταυτόχρονα στην κρυφή μνήμη με τα περιεχόμενα οποιαδήποτε δύο από τις α-γ, β και ε.**

Επεξεργαστής 3:

Έχουμε 2^7 πλαίσια και 42 πλαίσια ανά σύνολο, άρα έχουμε $2^7 / 2^2$ σύνολα = 2^5 σύνολα. Επομένως:

διεύθυνση μπλοκ				
12 δυαδικά ψηφία		5 δυαδικά ψηφία	3 δυαδικά ψηφία	
ετικέτα		διεύθυνση συνόλου	διεύθυνση μέσα στο πλαίσιο	
	Διεύθυνση στο δεκαεξαδικό	Ετικέτα ($v - \lambda$)	Διεύθυνση συνόλου (λ)	Διεύθυνση μέσα στο πλαίσιο (μ)
α	FD504	1111 1101 0101	0000 0	100
β	FFD07	1111 1111 1101	0000 0	111
γ	FD506	1111 1101 0101	0000 0	110
δ	FD7FA	1111 1101 0111	1111 1	010
ε	EC107	1110 1100 0001	0000 0	111

Οι διευθύνσεις α και γ έχουν την ίδια ετικέτα και την ίδια διεύθυνση πλαισίου, δηλαδή έχουν την ίδια διεύθυνση μπλοκ, επομένως όταν βρίσκονται τα περιεχόμενα της μίας εκ των α και γ θα βρίσκονται και της άλλης. Στη συνέχεια το ζεύγος α και γ θα συμβολίζουμε ως α-γ.

Οι διευθύνσεις των α-γ, β και ε έχουν διαφορετική ετικέτα, άρα ανήκουν σε διαφορετικά μπλοκ, αλλά την ίδια διεύθυνση συνόλου. Λαμβάνοντας υπόψη ότι κάθε σύνολο έχει τέσσερα πλαίσια, συμπεραίνουμε ότι τα περιεχόμενα και των τριών διεύθυνσεων μπορούν να βρίσκονται ταυτόχρονα στην κρυφή μνήμη.

Η διεύθυνση δ έχει διαφορετική ετικέτα και διεύθυνση πλαισίου με όλες τις άλλες διευθύνσεις α-γ, β και ε, επομένως τα περιεχόμενά της μπορούν να είναι ταυτόχρονα στην κρυφή μνήμη με τα περιεχόμενα ακόμη και των τριών α-γ, β και ε.



Θέμα Σεπτεμβρίου 2021 με κρυφή μνήμη και ιδεατή μνήμη

Θεωρείστε επεξεργαστή στον οποίο το μήκος των λογικών διευθύνσεων είναι 48 δυαδικά ψηφία ενώ το μήκος των φυσικών διευθύνσεων είναι 30 δυαδικά ψηφία και το μέγεθος της σελίδας είναι 2 Kbytes. Ο επεξεργαστής διαθέτει υβριδική κρυφή μνήμη επεξεργαστή με οργάνωση 4-τρόπων συνόλου συσχέτισης, με τη μέγιστη δυνατή χωρητικότητα και πλαίσιο των 32bytes, και υλοποιεί την τακτική της τελικής ενημέρωσης (write back με ένα dirty bit ανά πλαίσιο). Σε κάθε byte του συστήματος μνήμης αντιστοιχεί μία διεύθυνση.<p>

- α. Να υπολογίσετε τη χωρητικότητα της κρυφής μνήμης σε Kbytes (να δώσετε ένα καθαρό αριθμό χωρίς τελείες): .
β. Να υπολογίσετε το πλήθος των κυψελίδων που απαιτούνται για την υλοποίηση της κρυφής μνήμης (σε κάθε κυψελίδα αποθηκεύεται ένα δυαδικό ψηφίο). Αγνοήστε τα δυαδικά ψηφία που απαιτούνται για την υλοποίηση του μηχανισμού απελευθέρωσης πλαισίων (replacement policy).

Πλήθος κυψελίδων (δώστε ένα καθαρό αριθμό χωρίς τελείες): .

Θέμα Σεπτεμβρίου 2021 με κρυφή μνήμη και ιδεατή μνήμη

Θεωρείστε επεξεργαστή στον οποίο το μήκος των λογικών διευθύνσεων είναι 52 δυαδικά ψηφία ενώ το μήκος των φυσικών διευθύνσεων είναι 30 δυαδικά ψηφία και το μέγεθος της σελίδας είναι 16 Kbytes. Ο επεξεργαστής διαθέτει υβριδική κρυφή μνήμη επεξεργαστή με οργάνωση 2-τρόπων συνόλου συσχέτισης, με τη μέγιστη δυνατή χωρητικότητα και πλαίσιο των 16 bytes, και υλοποιεί την τακτική της τελικής ενημέρωσης (write back με ένα dirty bit ανά πλαίσιο). Σε κάθε byte του συστήματος μνήμης αντιστοιχεί μία διεύθυνση.<p>

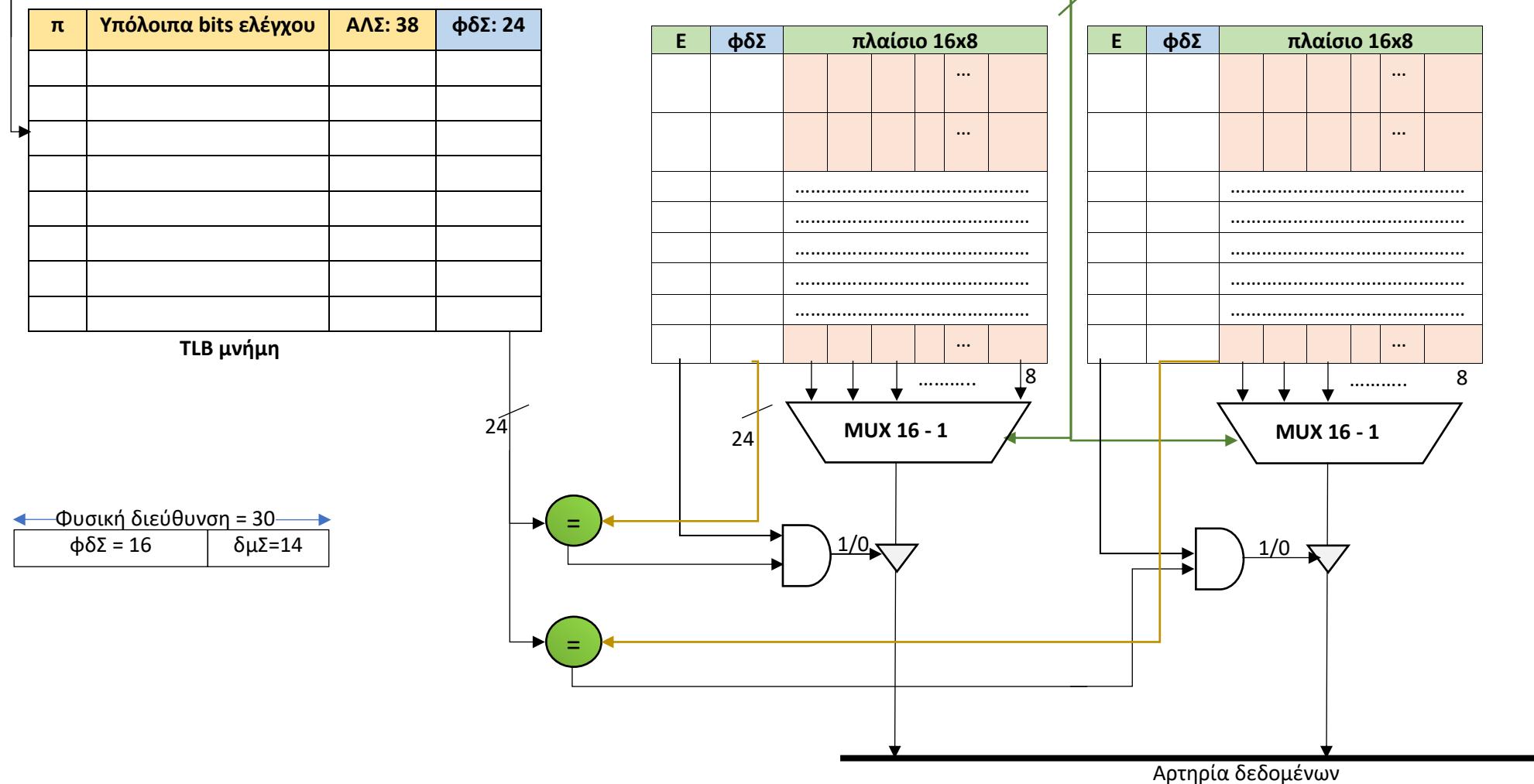
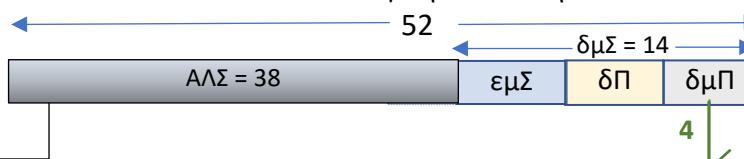
- α. Να υπολογίσετε τη χωρητικότητα της κρυφής μνήμης σε Kbytes (να δώσετε ένα καθαρό αριθμό χωρίς τελείες):
β. Να υπολογίσετε το πλήθος των κυψελίδων που απαιτούνται για την υλοποίηση της κρυφής μνήμης (σε κάθε κυψελίδα αποθηκεύεται ένα δυαδικό ψηφίο). Αγνοήστε τα δυαδικά ψηφία που απαιτούνται για την υλοποίηση του μηχανισμού απελευθέρωσης πλαισίων (replacement policy).

Πλήθος κυψελίδων (δώστε ένα καθαρό αριθμό χωρίς τελείες): .

Λύση

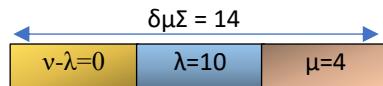
a)

π	Υπόλοιπα bits ελέγχου	ΑΛΣ: 38	φδΣ: 24



Η κρυφή μνήμη επεξεργαστή έχει **οργάνωση 2-τρόπων συνόλου συσχέτισης** και περιέχει 16 ψηφιολέξεις ανά πλαίσιο.

Επομένως, $2^{\mu} \geq 16 \Rightarrow \mu = 4$ bits. Επειδή αναφέρει η εκφώνηση να χρησιμοποιούμε τη μέγιστη δυνατή χωρητικότητα, θα θεωρήσουμε ότι στα πεδία της διεύθυνσης που παράγει ο επεξεργαστής για την οργάνωση 2-τρόπων συνόλου συσχέτισης, **η ετικέτα $v - \lambda = 0$** , οπότε $\lambda = 10$ bits.



2-τρόπων συνόλου συσχέτισης

Στη συνέχεια υπολογίζουμε τον αριθμό των πλαισίων της κρυφής μνήμης, ως εξής:

1 σύνολο περιέχει 2 πλαισία

$$2^{10} \quad x; \quad x = 2^{10} \times 2 = 2^{11} \text{ πλαίσια}$$

Η χωρητικότητα της κρυφής μνήμης είναι $2^{11} \text{ πλαίσια} \times \frac{16 \text{ bytes}}{\text{πλαίσιο}} = 2^{11} \times 2^4 = 2^{15} \text{ bytes} = 2^5 \text{ KB} = 32 \text{ KB}$

Σε αυτή τη χωρητικότητα πρέπει να προσθέσουμε τη χωρητικότητα των **bits εγκυρότητας** και τη χωρητικότητα των **dirty bits**, που σε κάθε περίπτωση είναι 1 bit/πλαίσιο. Επομένως, χωρητικότητα bits εγκυρότητας + χωρητικότητα των dirty bits = $2^{11} \text{ πλαίσια} \times \frac{1 \text{ bit}}{\text{πλαίσιο}} + 2^{11} \text{ πλαίσια} \times \frac{1 \text{ bit}}{\text{πλαίσιο}} = 2^{11} \text{ bits} + 2^{11} \text{ bits} = 2 \times 2^{11} \text{ bits} = 2^{12} \text{ bits} = 2^9 \text{ bytes} = 0.5 \text{ KB}$. Αθροιστικά έχουμε 32.5 KB → 32

b) Από τη στιγμή που η ακριβής χωρητικότητα της κρυφής μνήμης επεξεργαστή είναι $2^{15} \text{ bytes} + 2^9 \text{ bytes} = 33280 \text{ bytes} = 226240 \text{ bits} = 226240 \text{ κυψελίδες}$, αφού σε κάθε κυψελίδα αποθηκεύεται ένα δυαδικό ψηφίο.

Θέμα με κρυφή μνήμη

Θεωρείστε επεξεργαστή με αρτηρία διευθύνσεων των 30 δυαδικών ψηφίων κύρια μνήμη με οργάνωση μιας ψηφιολέξης (byte) ανά θέση και κρυφή μνήμη με χωρητικότητα 8 Κψηφιολέξεων και πλαίσιο των 32 ψηφιολέξεων. Θεωρήστε τις ακόλουθες δύο οργανώσεις της κρυφής μνήμης:

1. Οργάνωση μονοσήμαντης απεικόνισης (direct mapped)
2. 2-τρόπων συνόλου συσχέτισης (2-way set associative)

Για κάθε μία από τις ανωτέρω οργανώσεις:

- i. Να δώσετε την ανάλυση της διεύθυνσης που παράγει ο επεξεργαστής έτσι ώστε χρησιμοποιείται από την κρυφή μνήμη (δηλαδή να δώσετε τα πεδία, το εύρος κάθε ενός και τι δηλώνει το κάθε ένα πεδίο).
- ii. Να αναφέρετε και να αιτιολογήσετε εάν τα δεδομένα που είναι αποθηκευμένα στις ακόλουθες τρεις δεκαεξαδικές διεύθυνσεις: α: 36FF2384, β: 36FF438A, γ: 36FF2381 μπορούν να βρίσκονται ταυτόχρονα στην κρυφή μνήμη.

Λύση

A) Οργάνωση μονοσήμαντης απεικόνισης

$32 \text{ ψηφιολέξεις} = 2^5 \rightarrow$ Τα 5 τελευταία bit διεύθυνση λέξης στο πλαίσιο κύρια μνήμη 8 KBytes = $2^3 \times 2^{10} = 2^{13} \text{ bytes}$.

$2^{13}/2^5 = 2^8 \text{ πλ. } 2^{\kappa} \rightarrow \kappa = 8 \text{ bits}$, είναι το μέγεθος της διεύθυνσης πλαισίου. Άρα $30 - 8 - 5 = 17 \text{ bits}$.

Επομένως

Διεύθυνση	Ετικέτα $v - \kappa$ (17 bits)	Διεύθυνση πλαισίου κ (8 bits)	Διεύθυνση λέξης στο πλαίσιο μ (5 bits)
a) 36FF2384	0011 0110 1111 1111 001	0 0011 100	0 0100
β) 36FF438A	0011 0110 1111 1111 010	0 0011 100	0 1010
γ) 36FF2381	0011 0110 1111 1111 001	0 0011 100	0 0001

Η α και γ ανήκουν στο ίδιο μπλοκ αφού έχουν την ίδια ετικέτα και την ίδια διεύθυνση πλαισίου. Επομένως όταν θα βρίσκεται της μιας θα βρίσκεται και της άλλης σαν κρυφή μνήμη. Θα την συμβολίσουμε α – γ.

Η α - γ και η β έχουν ίδια διεύθυνση πλαισίου (κ) και διαφορετική ετικέτα (ν - κ). Επομένως, δεν μπορούν να βρίσκονται ταυτόχρονα στην κρυφή μνήμη. Μόνο μία από τις δύο μπορεί.

B) Οργάνωση 2 – τρόπων συνόλου συσχέτισης

Από πριν έχουμε βρει 2^8 πλαίσια, οπότε έχουμε $2^8/2 = 2^7 = 2^\lambda$ σύνολα. Το $\mu = 5$, αφού δεν αλλάζει ο αριθμός των λέξεων ($\theta.\mu.$) ανά πλαίσιο/μπλοκ. Επομένως:

Διεύθυνση	Ετικέτα $v - \lambda$ (18 bits)	Διεύθυνση συνόλου λ (7 bits)	Διεύθυνση λέξης στο πλαίσιο μ (5 bits)
α) 36FF2384	0011 0110 1111 1111 0010	0011 100	0 0100
β) 36FF438A	0011 0110 1111 1111 0100	0011 100	0 1010
γ) 36FF2381	0011 0110 1111 1111 0010	0011 100	0 0001

Η α και γ ανήκουν στο ίδιο μπλοκ αφού έχουν την ίδια ετικέτα και την ίδια διεύθυνση συνόλου. Επομένως όταν θα βρίσκεται της μιας θα βρίσκεται και της άλλης σαν κρυφή μνήμη. Θα την συμβολίσουμε α - γ. Αυτές οι δύο (που συμπεριφέρονται ως μία) θα καταλάβουν το Π0, του συνόλου με διεύθυνση 0011 100, και η διεύθυνση β πηγαίνει στο Π1. **Άρα μπορούν και οι τρεις να είναι ταυτόχρονα στην κρυφή μνήμη.**

Θέμα Σεπτεμβρίου 2016 με κρυφή μνήμη

Θεωρείστε δύο επεξεργαστές E1 και E2 κάθε ένας με ενοποιημένη κρυφή μνήμη (κοινή κρυφή μνήμη εντολών και δεδομένων) των 8192 θέσεων με μία ψηφιολέξη (byte) ανά θέση μνήμης και 16 ψηφιολέξεις ανά πλαίσιο και οργάνωση αντίστοιχα:

- α. Μονοσήμαντης απεικόνισης (direct mapped)
- β. 2-τρόπων συνόλου συσχέτισης (2-way associative)

Θεωρείστε ότι και οι δύο επεξεργαστές παράγουν διευθύνσεις των 24 δυαδικών ψηφίων και ότι **κάθε θέση της κύριας μνήμης είναι της μιας ψηφιολέξης.**

Για κάθε μία οργάνωση της κρυφής μνήμης να δώσετε:

- Να δώσετε τα πεδία από τα οποία θεωρούμε ότι αποτελείται η διεύθυνση του επεξεργαστή για την προσπέλαση κάθε μιας των ανωτέρω κρυφών μνημών και τι δηλώνει κάθε πεδίο.
- Να αναφέρετε και να αιτιολογήσετε τα περιεχόμενα ποιων από τις διευθύνσεις AED507₍₁₆₎ AFD7FA₍₁₆₎, AFD504₍₁₆₎, AFF7FF₍₁₆₎ AFF507₍₁₆₎ και AFD506₍₁₆₎ μπορούν να βρίσκονται ταυτόχρονα στην κρυφή μνήμη για κάθε μια από τις περιπτώσεις α και β.

Λύση

i) E1 - Μονοσήμαντης απεικόνισης

Διεύθυνση μπλοκ, $v = 24$ bits

$v - \kappa = 11$ ετικέτα	$\kappa = 9$ Διεύθυνση πλαισίου	$\mu = 4$ Διεύθυνση λέξης μέσα στο πλαίσιο και το μπλοκ
------------------------------	------------------------------------	--

$2^\mu = 16$ λέξεις ($\theta.\mu.$) $\Rightarrow \mu = 4$ bits.

Για υπολογισμό πλαισίων έχουμε: 1πλ.

16 ψηφιολέξεις = 16 bytes = 16 $\theta.\mu.$

x πλ;

8192 $\theta.\mu.$

$x = 2^{13}/2^4 = 2^9 = 2^\kappa \Rightarrow \kappa = 9$ bits.

$\nu - \kappa$

κ

μ

α. AED507 = 10101110110

101010000

0111

β. AFD7FA = 10101111110

101111111

1010

γ. AFD504 = **10101111110**

101010000

0100

δ. AFF7FF = 10101111111

101111111

1111

ε. AFF507 = 10101111111

101010000

0111

ζ. AFD506 = **10101111110**

101010000

0110

Μεθοδολογία: Ξεκινάμε ελέγχοντας ποιες από τις παραπάνω διευθύνσεις έχουν το **ίδιο κ (διεύθυνση πλαισίου)**.

Αυτές είναι οι διευθύνσεις (α), (γ), (ε), (ζ). Στη συνέχεια ελέγχουμε ποιες από αυτές έχουν ίδια ετικέτα ($v - \kappa$). Αυτές είναι οι διευθύνσεις (γ) και (ζ). Επομένως αυτές οι δύο διευθύνσεις μπορούν και πρέπει να είναι ταυτόχρονα στην κρυφή μνήμη, διότι έχουμε ίδιο μπλοκ (ίδιο v), το οποίο μεταφέρεται στο ίδιο πλαίσιο (ίδιο κ).

Οι διευθύνσεις (α) και (ε) δεν μπορούν να είναι ταυτόχρονα στην κρυφή μνήμη, τόσο μεταξύ τους όσο και με τις (γ) και (ζ), γιατί αναφέρονται σε διαφορετικά μπλοκ, που μετακινούνται στο ίδιο πλαίσιο και αναγκαστικά αντικαθιστούν η μία την άλλη. Τέλος Οι διευθύνσεις (β) και (δ) δεν μπορούν επίσης να είναι ταυτόχρονα μεταξύ τους στην κρυφή μνήμη, διότι αναφέρονται σε διαφορετικά μπλοκ που μετακινούνται στο ίδιο πλαίσιο (10111111) και συνεπώς αντικαθιστά και πάλι η μία την άλλη. Όμως, οποιαδήποτε από τις (β) και (δ) μπορεί να βρίσκεται ταυτόχρονα στην κρυφή μνήμη με οποιαδήποτε από τις υπόλοιπες, διότι μεταφέρεται σε διαφορετικό πλαίσιο σε σχέση με αυτές.

ii) 2 – τρόπων συνόλου συσχέτισης

Διεύθυνση μπλοκ, $v = 24$ bits

$v - \lambda = 12$ bits = 3 hex ψηφία	λ = 8 bits = 2 hex ψηφία	$\mu = 4$ bits = 1 hex ψηφίο
ετικέτα	Διεύθυνση συνόλου	Διεύθυνση λέξης μέσα στο πλαίσιο/μπλοκ

Από πριν έχουμε 512 πλαίσια επομένως διοθέντος ότι έχουμε 2 – τρόπων συνόλου συσχέτισης μνήμη, ισχύει ότι:

$$\begin{array}{ll} 1 \text{ σύνολο} & 2 \pi\lambda. \\ x; & 512 \pi\lambda. \end{array}$$

$x = 2^9 / 2^1 = 2^8 = 2^\lambda \Rightarrow \lambda = 8$ bits. Επειδή τα μεγέθη πεδίων είναι πολλαπλάσια του 4, μπορούμε για λόγους ευκολίας να τα μετατρέψουμε σε δεκαεξαδικά ψηφία.

v - λ	λ	μ
a. AED507 = AED	50	7
β. AFD7FA = AFD	7F	A
γ. AFD504 = AFD	50	4
δ. AFF7FF = AFF	7F	F
ε. AFF507 = AFF	50	7
ζ. AFD506 = AFD	50	6

Μεθοδολογία: Ξεκινάμε ελέγχοντας ποιες από τις παραπάνω διευθύνσεις έχουν το **ίδιο λ (διεύθυνση συνόλου)**.

Αυτές είναι οι διευθύνσεις (α), (γ), (ε), (ζ). Στη συνέχεια ελέγχουμε ποιες από αυτές έχουν ίδια ετικέτα ($v - \lambda$). Αυτές είναι οι διευθύνσεις (γ) και (ζ). Επομένως αυτές οι δύο διευθύνσεις μπορούν και πρέπει να είναι ταυτόχρονα στην κρυφή μνήμη, διότι έχουμε ίδιο μπλοκ (ίδιο v), το οποίο μεταφέρεται στο ίδιο σύνολο (ίδιο λ) και μάλιστα στο ίδιο πλαίσιο του συνόλου αυτού. Επίσης, επειδή αυτές οι 2 διευθύνσεις καταλαμβάνουν -όπως αναφέρθηκε- το ίδιο πλαίσιο του συνόλου στο οποίο μεταφέρονται, είναι δυνατό, οποιαδήποτε από τις διευθύνσεις (α) ή (ε) να βρίσκεται ταυτόχρονα στην κρυφή μνήμη με οποιαδήποτε από τις (γ) και (ζ).

Τέλος, οι διευθύνσεις (β) και (δ) μπορούν επίσης να είναι ταυτόχρονα μεταξύ τους στην κρυφή μνήμη, διότι παρά το ότι αναφέρονται σε διαφορετικά μπλοκ, μετακινούνται στο ίδιο σύνολο (7F) σε διαφορετικά πλαίσια του συνόλου αυτού και συνεπώς μπορούν να βρίσκονται ταυτόχρονα στην κρυφή μνήμη, τόσο μεταξύ τους, όσο και με τις υπόλοιπες.



Θέμα με κρυφή μνήμη και γέμισμα πίνακα διευθύνσεων με στρατηγική LRU

Θεωρούμε κρυφή μνήμη με οργάνωση 2-τρόπων συνόλου συσχέτισης αποτελούμενη από 8 πλαίσια και **δύο λέξεις ανά πλαίσιο**, η οποία υλοποιεί τη στρατηγική αντικατάστασης **LRU**. Υποθέστε ότι η KME παράγει διεύθυνσεις των 8 διαδικών ψηφίων. Θεωρούμε ότι αρχικά η κρυφή μνήμη ήταν κενή και ότι η KME παρήγαγε την επόμενη ακολουθία δεκαεξαδικών διευθύνσεων 80, 42, F9, CA, 80, 3F, 59, 97, 58, 1B, 58, C3, F8, 1A, 0B, C7, 08, 0A, CC, CF, C7. Να συμπληρώστε τους επόμενους δύο πίνακες. Θεωρήστε ότι πρώτα μεταφέρεται πληροφορία στο πλαίσιο 0 και μετά στο 1.

Διεύθυνση	Διεύθυνση	Μεταφορά δεδομένων στην κρυφή μνήμη
-----------	-----------	-------------------------------------



(δεκαεξαδικό)	$v-\lambda = 5$	$\lambda = 2$	$\mu = 1$	επιτυχία ή αποτυχία (επ ή απ)	Πλαίσιο-0	Πλαίσιο-1
80						
42						
F9						
CA						
80						
3F						
59						
97						
58						
1B						
58						
C3						
F8						
1A						
0B						
C7						
08						
0A						
CC						
CF						
C7						

Περιεχόμενο της κρυφής μνήμης μετά την προσπέλαση της ακολουθίας των διευθύνσεων

	Πλαίσιο 0	Πλαίσιο 1
Σύνολο 0		
Σύνολο 1		
Σύνολο 2		
Σύνολο 3		

Λύση

Διεύθυνση (δεκαεξαδικό)	Διεύθυνση ($v-\lambda=5, \lambda=2, \mu=1$)	επιτυχία ή αποτυχία (επ ή απ)	Μεταφορά δεδομένων στην κρυφή μνήμη	
			Πλαίσιο-0	Πλαίσιο-1
80	10000 – 00 – 0	απ	ΜΠ(80-81) → Σ0	
42	00100 – 01 – 0	απ	ΜΠ(42-43) → Σ1	
F9	11111 – 00 – 1	απ		ΜΠ(F8-F9) → Σ0
CA	11001 – 01 – 0	απ (miss)		ΜΠ(CA-CB) → Σ1
80	10000 – 00 – 0	επ (hit)	ΜΠ(80-81) – Σ0	Στην περίπτωση της LRU, οι επιτυχίες σημειώνονται με – για να δείξουμε ότι υπάρχει ενημέρωση/ανανέωση του πλαισίου, στο οποίο βρίσκεται ήδη η διεύθυνση. Στην περίπτωση της FIFO το αφήνουμε κενό.
3F	00111 – 11 – 1	απ	ΜΠ(3E-3F) → Σ3	
59	01011 – 00 – 1	απ		ΜΠ(58-59) → Σ0 LRU
97	10010 – 11 – 1	απ		ΜΠ(96-97) → Σ3
58	01011 – 00 – 0	επ		ΜΠ(58-59) – Σ0
1B	00011 – 01 – 1	απ	ΜΠ(1A-1B) → Σ1 LRU	
58	01011 – 00 – 0	επ		ΜΠ(58-59) – Σ0
C3	11000 – 01 – 1	απ		ΜΠ(C2-C3) → Σ1 LRU
F8	11111 – 00 – 0	απ	ΜΠ(F8-F9) → Σ0 LRU	
1A	00011 – 01 – 0	επ	ΜΠ(1A-1B) – Σ1	
0B	00001 – 01 - 1	απ		ΜΠ(0A-0B) → Σ1 LRU
C7	11000 – 11 - 1	απ	ΜΠ(C6-C7) → Σ3 LRU	
08	00001 - 00 - 0	απ		ΜΠ(08-09) → Σ0 LRU
0A	00001 – 01 - 0	επ		ΜΠ(0A-0B) - Σ1
CC	11001 – 10 - 0	απ	ΜΠ(CC-CD) → Σ2	
CF	11001 – 11 - 1	απ		ΜΠ(CE-CF) → Σ3 LRU
C7	11000 – 11 - 1	επ	ΜΠ(C6-C7) – Σ3	

Περιεχόμενο της κρυφής μνήμης μετά την προσπέλαση της ακολουθίας των διευθύνσεων

	Πλαίσιο 0	Πλαίσιο 1
Σύνολο 0	ΜΠ(F8-F9)	ΜΠ(08-09)
Σύνολο 1	ΜΠ(1A-1B)	ΜΠ(0A-0B)
Σύνολο 2	ΜΠ(CC-CD)	
Σύνολο 3	ΜΠ(C6-C7)	ΜΠ(CE-CF)

Θέμα με κρυφή μνήμη και στρατηγικές απελευθέρωσης πλαισίων

Θεωρείστε ότι η ΚΜΕ ενός υπολογιστή, που παράγει διευθύνσεις των 7 δυαδικών ψηφίων, παράγει την επόμενη ακολουθία διευθύνσεων: 33, 17, 30, 47, 50, 35, 6, 26, 50, 42, 58, 50, 13, 22, 15, 4, 0, 70. Για κάθε μία από τις επόμενες περιπτώσεις να γράψετε δίπλα σε κάθε διεύθυνση εάν έχουμε επιτυχία ή αποτυχία και στη συνέχεια να δώσετε το τελικό περιεχόμενο της κρυφής μνήμης. Στο ξεκίνημα η κρυφή μνήμη δεν έχει έγκυρα περιεχόμενα.

- α. Η κρυφή μνήμη έχει οργάνωση **μονοσήμαντης απεικόνισης** με χωρητικότητα 32 λέξεων και **4 λέξεις ανά πλαίσιο**.
- β. Η κρυφή μνήμη έχει οργάνωση **2-τρόπων συνόλου συσχέτισης**, χωρητικότητα 32 λέξεων και **2 λέξεις ανά πλαίσιο**. Υποθέστε ότι χρησιμοποιείται η στρατηγική απελευθέρωσης πλαισίων της κρυφής μνήμης "FIFO".
- γ. Η κρυφή μνήμη έχει οργάνωση **πλήρους συνόλου συσχέτισης**, χωρητικότητα 32 λέξεων και **4 λέξεις ανά πλαίσιο**. Υποθέστε ότι χρησιμοποιείται η στρατηγική απελευθέρωσης πλαισίων της κρυφής μνήμης "LRU".

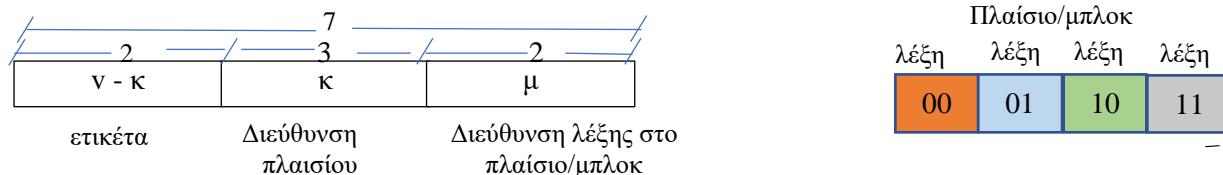
Λύση

α. Όταν δεν δίνεται από την εκφώνηση ο ακριβής αριθμός πλαισίων της κρυφής μνήμης, αυτός θα υπολογίζεται από τη χωρητικότητα της κρυφής μνήμης. Πιο συγκεκριμένα, εφαρμόζουμε μια απλή μέθοδο των τριών:

$$\begin{array}{ll} \text{1 πλ.} & 4 \text{ λέξεις} \\ x; & 32 \text{ λέξεις} \end{array} \quad x = 32/4 = 8 \text{ πλ. Άρα } 2^k = 8 \text{ πλαισία } \Rightarrow k=3 \text{ bits}$$

Ο συνολικός αριθμός των λέξεων είναι $2^\mu = 4$ λέξεις $\Rightarrow \mu = 2$ bits

Το πεδίο $v - \kappa$ της ετικέτας υπολογίζεται πάντα με αφαίρεση.

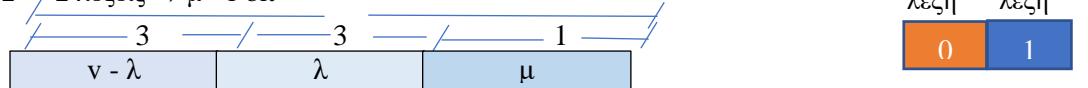


Το **v = 5** (διεύθυνση μπλοκ) \rightarrow επομένως στην κύρια μνήμη έχουμε συνολικά 2^5 μπλοκ (blocks). Επειδή το κάθε μπλοκ/πλαίσιο αποτελείται από τέσσερις λέξεις ($\theta.\mu.$), θα μεταφέρονται κάθε φορά από την κύρια στην κρυφή μνήμη, τέσσερις διαδοχικές διευθύνσεις που είναι οι λέξεις «00», «01», «10» και «11».

Διεύθυνση (Δεκαδική-Δυαδική.)	Διεύθυνση (ετικέτα – πλαίσιο – λέξη)	επ / απ (hit/miss)	Μεταφορά στην κρυφή μνήμη
33 = 0100001	01 - 000 - 01	απ (miss)	ΜΠ(32,33,34,35) \rightarrow ΙΙ0
17 = 0010001	00 - 100 - 01	απ	ΜΠ(16,17,18,19) \rightarrow ΙΙ4
30 = 0011110	00 - 111 - 10	απ	ΜΠ(28,29,30,31) \rightarrow ΙΙ7
47 = 0101111	01 - 011 - 11	απ (miss)	ΜΠ(44,45,46,47) \rightarrow ΙΙ3
50 = 0110010	01 - 100 - 10	απ (miss)	ΜΠ(48,49,50,51) \rightarrow ΙΙ4
35 = 0100011	01 - 000 - 11	επ (hit)	
06 = 0000110	00 - 001 - 10	απ	ΜΠ(4,5,6,7) \rightarrow ΙΙ1
26 = 0011010	00 - 110 - 10	απ	ΜΠ(24,25,26,27) \rightarrow ΙΙ6
50 = 0110010	01 - 100 - 10	επ	-
42 = 0101010	01 - 010 - 10	απ	ΜΠ(40,41,42,43) \rightarrow ΙΙ2
Κ.Ο.Κ			

β. 2 - τρόπων συνόλου συσχέτισης

$$2^\mu = 2 \text{ λέξεις} \Rightarrow \mu = 1 \text{ bit}$$



Πρώτα θα υπολογίσουμε τον αριθμό των πλαισίων και στη συνέχεια τον αριθμό των συνόλων. Ισχύει ότι

1 πλ. περιέχει 2 λέξεις

x; x = 32 λέξεις x= 16 πλαίσια. Από αυτά θα υπολογίσουμε τα σύνολα.

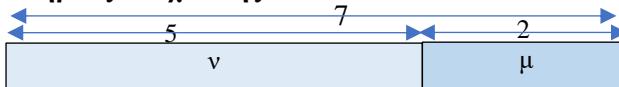
1 σύνολο 2 πλ.

x; 16 πλ x = 8 σύνολα = $2^\lambda \Rightarrow \lambda = 3$ bits

Όταν γεμίζει ένα σύνολο, πρώτα γεμίζει το πλαίσιο 0 του συνόλου αυτού, μετά το πλαίσιο 1, μετά το πλαίσιο 2 κ.ο.κ. Στον πίνακα που ακολουθεί, εφαρμόζουμε στρατηγική FIFO.

Διεύθυνση (Δεκαδική-Δυαδική.)	Διεύθυνση (ετικέτα - σύνολο - λέξη)	επ/απ (hit/miss)	Μεταφορά στην κρυφή μνήμη	
			Π0	Π1
33 = 0100001	010 - 000 - 1	απ (miss)	ΜΠ(32,33) → Σ0	
17 = 0010001	001 - 000 - 1	απ		ΜΠ(16,17) → Σ0
30 = 0011110	001 - 111 - 0	απ	ΜΠ(30,31) → Σ7	
47 = 0101111	010 - 111 - 1	απ		ΜΠ(46,47) → Σ7
50 = 0110010	011 - 001 - 0	απ	ΜΠ(50,51) → Σ1	
35 = 0100011	010 - 001 - 1	απ		ΜΠ(34,35) → Σ1
06 = 0000110	000 - 011 - 0	απ	ΜΠ(6,7) → Σ3	
26 = 0011010	001 - 101 - 0	απ	ΜΠ(26, 27) → Σ5	
50 = 0110010	011 - 001 - 0	επ		
42 = 0101010	010 - 101 - 0	απ		ΜΠ(42, 43) → Σ5
58 = 0111010	011 - 101 - 0	απ	ΜΠ(58, 59) → Σ5 (FIFO)	
50 = 0110010	011 - 001 - 0	επ		

γ) πλήρους συσχέτισης



Διεύθυνση μπλοκ

Διεύθυνση λέξης στο πλαίσιο/μπλοκ

Στην περίπτωση αυτή, **τα πλαίσια της κρυφής μνήμης γεμίζουν πάντα με τη σειρά από το πρώτο Π0 έως το τελευταίο (Π7).** Μια βασική διαφοροποίηση της στρατηγικής LRU σε σχέση με τη FIFO είναι ότι στην πρώτη σημειώνουμε τις επιτυχίες με -, ενώ στη δεύτερη αφήνουμε τα πλαίσια κενά. Ο λόγος είναι ότι μόλις ένα πλαίσιο έχει επιτυχία, αυτόματα **ανανεώνεται**, επομένως σταματά να είναι το πιο παλιό. Η στρατηγική LRU εφαρμόζεται από τη στιγμή που θα γεμίσουν όλα τα πλαίσια της κρυφής μνήμης και όχι νωρίτερα. Δηλαδή για όσο διάστημα γεμίζει η κρυφή μνήμη με μπλοκ που έρχονται από την κύρια μνήμη, η στρατηγική LRU δεν χρησιμοποιείται. Θα εφαρμοστεί **μόνο όταν έχουν γεμίσει και τα 8 πλαίσια της κρυφής** και στη συνέχεια έλθει ένα νέο μπλοκ από την κύρια μνήμη, το οποίο δεν υπάρχει στην κρυφή μνήμη. Στη στρατηγική LRU = Least Recently Used αντικαθίσταται εκείνο το πλαίσιο που έχει να χρησιμοποιηθεί την πιο πολύ ώρα.

διεύθυνση μνήμης (δεκαδικό)	Διεύθυνση (ετικέτα - λέξη)	επιτυχία ή αποτυχία (επ ή απ)	μεταφορά δεδομένων στην κρυφή μνήμη
33	01000-01	απ	
17	00100-01	απ	ΜΠ(32, 33, 34, 35) => Π0*
30	00111-10	απ	ΜΠ(16, 17, 18, 19) => Π1
47	01011-11	απ	ΜΠ(28, 29, 30, 31) => Π2
50	01100-10	απ	ΜΠ(44, 45, 46, 47) => Π3
35	01000-11	επ	ΜΠ(48, 49, 50, 51) => Π4
6	00001-10	απ	<u>ΜΠ(32, 33, 34, 35) - ΠΦ</u> ΜΠ(4, 5, 6, 7) => Π5
26	00110-10	απ	ΜΠ(24, 25, 26, 27) => Π6
50	01100-10	επ	<u>ΜΠ(48, 49, 50, 51) - Π4</u> ΜΠ(40, 41, 42, 43) => Π7
42	01010-10	απ	

58	01110-10	απ	ΜΠ(56, 57, 58, 59) => Π1 (LRU)
50	01100-10	επ	ΗΠ(48, 49, 50, 51) - Π4
13	00011-01	απ	ΜΠ(12, 13, 14, 15) => Π2 (LRU)
22	00101-10	απ	ΜΠ(20, 21, 22, 23) => Π3 (LRU)
15	00011-11	επ	ΗΠ(12, 13, 14, 15) - Π2
4	00001-00	επ	ΗΠ(4, 5, 6, 7) - Π5
0	00000-00	απ	ΜΠ(0, 1, 2, 3) => Π0 (LRU)
70	10001-10	απ	ΜΠ(68, 69, 70, 71) => Π6 (LRU)

Θέμα με κρυφή μνήμη και κώδικα Assembly

Θεωρήστε ότι σε ένα υπολογιστικό σύστημα, το οποίο δεν έχει ιδεατή μνήμη, υπάρχει μία κρυφή μνήμη δεδομένων με οργάνωση τ-τρόπων συνόλου συσχέτισης. Η κρυφή μνήμη αποτελείται από 64 πλαίσια και κάθε πλαίσιο της κρυφής μνήμης είναι των τεσσάρων λέξεων (στην περίπτωση μας μία λέξη = 16 δυαδικά ψηφία).

α. Προσδιορίστε τη μικρότερη τιμή του τ για την οποία έχουμε το μικρότερο αριθμό αποτυχιών (misses) στην κρυφή μνήμη κατά την εκτέλεση του προγράμματος που ακολουθεί. Σημειώστε ότι για $\tau=1$ έχουμε κρυφή μνήμη με οργάνωση μονοσήμαντης απεικόνισης.

Πρόγραμμα

LOAD D, # 05FF	/εντολή 1
LOAD H, # 02FF	/εντολή 2
LOOP INCR D	/εντολή 3
INCR H	/εντολή 4
LOAD A, [D]	/εντολή 5
ADD A, [H]	/εντολή 6
STORE A, [H]	/εντολή 7
MOVE A, H	/εντολή 8
COMP A, #03FF	/εντολή 9
JNZ LOOF	/εντολή 10
END	/εντολή 11

β. Στην κρυφή μνήμη που επιλέξατε ποιος είναι ο αριθμός των αποτυχιών (misses) κατά την εκτέλεση του προηγούμενου προγράμματος;

Αύστη

α) και β) Η μικρότερη τιμή του τ που θα μπορούσαμε να χρησιμοποιήσουμε είναι $\tau = 1$, οπότε στην περίπτωση αυτή μιλάμε για μια κρυφή μνήμη με οργάνωση μονοσήμαντης απεικόνισης. Γνωρίζουμε ότι σε μια τέτοια περίπτωση έχουμε το μεγαλύτερο αριθμό αποτυχιών. Θα δοκιμάσουμε για αυτή η συγκεκριμένη οργάνωση και θα ελέγξουμε τον αριθμό των αποτυχιών της.

Διεύθυνση μπλοκ, $v = 14$ bits

$v - \kappa = 8$	$\kappa = 6$	$\mu = 2$
------------------	--------------	-----------

ετικέτα Διεύθυνση πλαισίου Διεύθυνση λέξης μέσα στο
 $2^\mu = 4$ λέξεις ($\theta \cdot \mu$) $\Rightarrow \mu = 2$ bits. πλαίσιο και το μπλοκ

LOAD A, [D]	//A \leftarrow [D], δηλ. φορτώνουμε στον καταχωρητή A το περιεχόμενο της θ.μ. που //υποδεικνύει το [D]
ADD A, [H]	//A \leftarrow A + [H], δηλ. φορτώνουμε στον καταχωρητή A το άθροισμα του προηγούμενου //περιεχόμενου του καθώς και αυτού που υποδεικνύει το [H].
STORE A, [H]	//[H] \leftarrow A φορτώνουμε το περιεχόμενο του καταχωρητή A στη θέση μνήμης που υποδεικνύει το [H].
MOVE A, H	//A \leftarrow H δηλ. φορτώνουμε το περιεχόμενο του καταχωρητή H στον καταχωρητή A.
COMP A, #03FF	//A - 03FF δηλ. συγκρίνουμε το περιεχόμενο του καταχωρητή A με την τιμή 3FF. //Όταν αυτή η διαφορά γίνει ίση με «0», τότε ενεργοποιείται το zero flag, δηλ. το Z = 1. //Αρχικά, η σύγκριση αυτή γίνεται μεταξύ των τιμών 0300 και 03FF. Από αυτή τη //διαφορά καταλαβαίνουμε ότι οι επαναλήψεις θα γίνουν 03FF -

στο Σ0 (και συγκεκριμένα στο Π1 του συνόλου αυτού) υπάρχει επίσης αποτυχία. Στην επόμενη επανάληψη (από το LOOP) η πάνω τετράδα (λόγω της διεύθυνσης 0601 που έχει παραχθεί) υπάρχει ήδη στο Π0 του Σ0 και επίσης και η κάτω τετράδα (λόγω της διεύθυνσης 0301 που έχει παραχθεί) υπάρχει ήδη στο Π1 του Σ0 και επομένως έχουμε επιτυχία (hit). Αυτό συμβαίνει και με όλες τις υπόλοιπες διεύθυνσεις. Δηλαδή, κάθε νέα τετράδα που μεταφέρεται από την κύρια στην κρυφή μνήμη, έχει την πρώτη φορά αποτυχία και όλες τις υπόλοιπες φορές επιτυχία. Επομένως, από τις 256 (0300 – 03FF) και 256 (0600 – 06FF) προσπελάσεις μνήμης έχουμε για κάθε 4 προσπελάσεις μια αποτυχία και για τις 512 συνολικά προσπελάσεις έχουμε $\frac{1}{4} * 512$ προσπελάσεις = 128 αποτυχίες (misses). Αυτός είναι ο ιδανικός (μικρότερος) αριθμός αποτυχιών που μπορούμε να έχουμε, διότι ακόμη και να ανέβουμε σε οργάνωση και να χρησιμοποιήσουμε μια 4 – τρόπων συνόλου συσχέτισης κρυφή μνήμη, δεν θα κερδίσουμε κάτι (δηλ. δεν θα μειωθεί ο αριθμός των αποτυχιών), αφού σε μια τέτοια περίπτωση, μόνο τα δύο από τα τέσσερα πλαίσια του κάθε συνόλου της κρυφής μνήμης θα χρησιμοποιηθούν, γιατί σε κάθε επανάληψη, θα συνεχίσουν να παράγονται (μέσω των εντολών INCR D και INCR H) δύο τετράδες διεύθυνσεων που θα μεταφέρονται στα πλαίσια Π0 και Π1 του κάθε συνόλου. Όταν ολοκληρωθούν οι δύο πρώτες τετράδες από 0600H – 0603H και από 0300H – 0303H, τότε όταν παραχθούν οι επόμενες δύο τετράδες 0604H – 0607H = 0000 0110 0000 0100 – 0000 0110 0000 0111 και από 0304H – 0307H, αλλάζει το σύνολο και γίνεται **0001**. Επομένως η μικρότερη τιμή του τ που μπορεί να χρησιμοποιηθεί είναι $\tau = 2$.

Θέμα με χωρητικότητα κρυφής μνήμης

Έχετε κύρια μνήμη με οργάνωση **μιας ψηφιολέξης (byte)** ανά θέση μνήμης. Για την προσπέλαση της κρυφής μνήμης θεωρήστε ότι η διεύθυνση αποτελείται από τρία πεδία:

13 δυαδικά ψηφία	14 δυαδικά ψηφία	5 δυαδικά ψηφία
------------------	------------------	-----------------

Το μέγεθος της κρυφής μνήμης είναι μικρότερο ή ίσο με 2 Μψηφιολέξεις (2 Mbytes).

- α. Ποιες είναι οι πιθανές οργανώσεις της κρυφής μνήμης και ποια είναι η χωρητικότητα της σε κάθε περίπτωση;
- β. Να υπολογίσετε σε κάθε περίπτωση ποιος είναι ο αριθμός των μπλοκ της κύριας μνήμης με χωρητικότητα 2^{32} ψηφιολέξεις που μπορούν να μεταφερθούν (όχι ταυτόχρονα φυσικά) στο ίδιο πλαίσιο ή σύνολο, ανάλογα της οργάνωσης, της κρυφής.

Λύση

α. Από τα τρία πεδία στα οποία χωρίζεται η διεύθυνση μπορούμε να διαπιστώσουμε ότι οι πιθανές οργανώσεις κρυφής μνήμης είναι αυτή της **μονοσήμαντης απεικόνισης** και της **τ – τρόπων συνόλου συσχέτισης**.

β. Αν υποθέσουμε ότι έχουμε **οργάνωση μονοσήμαντης απεικόνισης**, τότε:

Διεύθυνση μπλοκ, $v = 27$ bits		
$v - \kappa = 13$	$\kappa = 14$	$\mu = 5$
ετικέτα	Διεύθυνση πλαισίου	Διεύθυνση λέξης μέσα στο πλαίσιο και το μπλοκ

$v = 27$ bits $\rightarrow 2^v$ μπλοκ 2^{27} μπλοκ. **Προσοχή!** το 2^{10} ψηφιολέξεις = 1KB.

Χωρητικότητα κρυφής μνήμης = πλήθος πλαισίων x μέγεθος κάθε πλαισίου = 2^{14} πλαίσια $x 2^5 \frac{\lambdaέξεις}{\piλαίσιο} = 2^{19}$ λέξεις ($\theta.\mu.$) =

2^{19} ψηφιολέξεις (bytes)¹ = $2^9 x 2^{10}$ ψηφιολέξεις (bytes) 512 KB < 2 MB, άρα **αποδεκτή** οργάνωση κρυφής μνήμης.

Αν υποθέσουμε ότι έχουμε **οργάνωση τ – τρόπων συνόλου συσχέτισης**, τότε:

Διεύθυνση μπλοκ, $v = 27$ bits		
$v - \lambda = 13$	$\lambda = 14$	$\mu = 5$
ετικέτα	Διεύθυνση συνόλου	Διεύθυνση λέξης μέσα στο πλαίσιο/μπλοκ

Έστω 2 – τρόπων συνόλου συσχέτισης: Χωρητικότητα κρυφής μνήμης = πλήθος συνόλων x αριθμός πλαισίων κάθε συνόλου x μέγεθος κάθε πλαισίου = 2^{14} σύνολα $x 2 \frac{\piλαίσια}{\sigmaύνολο} x 2^5 \frac{\lambdaέξεις}{\piλαίσιο} = 2^{14} x 2 x 2^5$ λέξεις ($\theta.\mu.$) = 2^{20} λέξεις ($\theta.\mu.$) = (λόγω της οργάνωσης) 2^{20} ψηφιολέξεις (bytes) = 1 MB < 2 MB, άρα αποδεκτή οργάνωση κρυφής μνήμης.

Έστω οργάνωση 3 – τρόπων συνόλου συσχέτισης: Χωρητικότητα κρυφής μνήμης = πλήθος συνόλων x αριθμός πλαισίων κάθε συνόλου x μέγεθος κάθε πλαισίου = 2^{14} σύνολα $x 3 \frac{\piλαίσια}{\sigmaύνολο} x 2^5 \frac{\lambdaέξεις}{\piλαίσιο} = 2^{14} x 3 x 2^5$ λέξεις = $3 x 2^{19}$ λέξεις ($\theta.\mu.$) = $3 x 2^9$ x 2^{10} λέξεις ($\theta.\mu.$) = $3 x 512$ Κψηφιολέξεις (bytes) = 1.5 MB < 2 MB, άρα αποδεκτή οργάνωση κρυφής μνήμης.

¹ Δεν ταυτίζονται πάντα οι λέξεις με τις ψηφιολέξεις. Αυτό ισχύει στην προκειμένη περίπτωση, διότι αναφέρει η εκφώνηση ότι έχουμε οργάνωση μιας ψηφιολέξης ανά θέση μνήμης.

Έστω οργάνωση 4 – τρόπων συνόλου συσχέτισης: Χωρητικότητα κρυφής μνήμης = πλήθος συνόλων x αριθμός πλαισίων

κάθε συνόλου x μέγεθος κάθε πλαισίου = 2^{14} σύνολα x $2^2 \frac{\text{πλαισια}}{\text{σύνολο}} x 2^5 \frac{\lambda\epsilon\xi\epsilon\varsigma}{\text{πλαισιο}} = 2^{14} x 2^2 x 2^5 = 2^{21} \lambda\epsilon\xi\epsilon\varsigma$ (θ.μ.) = (λόγω της οργάνωσης) 2^{21} ψηφιολέξεις (bytes) = $2^1 x 2^{20}$ bytes = 2 MB, άρα αποδεκτή οργάνωση κρυφής μνήμης. Δεν μπορούμε να πάμε σε οργάνωση μεγαλύτερης τάξης, διότι τότε θα ξεπεράσουμε την επιτρεπτή χωρητικότητα των 2 MB.

β. Στην περίπτωση της οργάνωσης μονοσήμαντης απεικόνισης, θα υπολογίσουμε από τη μια των αριθμό των πλαισίων της κρυφής μνήμης και από την άλλη τον αριθμό των μπλοκ της κύριας μνήμης. Αφού το $\kappa = 14$ bits \rightarrow έχουμε 2^{14} πλαισία. Από την άλλη, αφού έχουμε $v = 27$ bits \rightarrow έχουμε 2^{27} μπλοκ. Επομένως:

$$\begin{array}{l} \text{Σε } 2^{14} \text{ πλαισία μεταφέρονται } 2^{27} \text{ μπλοκ} \\ | \\ \text{1 πλ.} \quad \text{x:} \quad x = 2^{27} \text{ μπλοκ}/2^{14} \text{ πλαισία} = 2^{13} \text{ μπλοκ/πλαισίο.} \end{array}$$

Άρα μπορούν να μεταφερθούν 2^{13} μπλοκ/πλαισίο, προφανώς όχι ταυτόχρονα. Δηλαδή, κάθε φορά μόνο ένα από τα 2^{13} μπλοκ μπορούν να είναι ταυτόχρονα σε ένα αντίστοιχο πλαισίο στην κρυφή μνήμη.

Στην περίπτωση της οργάνωσης τ – τρόπων συνόλου συσχέτισης, θα πρέπει να υπολογίσουμε από τη μια των αριθμό των συνόλων της κρυφής μνήμης και από την άλλη τον αριθμό των μπλοκ της κύριας μνήμης. Αφού το $\lambda = 14$ bits \rightarrow έχουμε 2^{14} σύνολα. Από την άλλη, αφού έχουμε $v = 27$ bits \rightarrow έχουμε 2^{27} μπλοκ. Επομένως ισχύει:

$$\begin{array}{l} \text{Σε } 2^{14} \text{ σύνολα μεταφέρονται } 2^{27} \text{ μπλοκ} \\ | \\ \text{1 συν.} \quad \text{x:} \end{array}$$

$x = 2^{27} \text{ μπλοκ}/2^{14} \text{ σύνολα} = 2^{13} \text{ μπλοκ/σύνολο.}$ Άρα μπορούν να μεταφερθούν 2^{13} μπλοκ/σύνολο, προφανώς όχι ταυτόχρονα. Αν το σύνολο περιέχει 2 πλαισία, αυτό σημαίνει από τα 2^{27} διαφορετικά μπλοκ, μεταφέρονται κάθε φορά δύο από αυτά σε ένα σύνολο. Αν το σύνολο περιέχει 3 πλαισία, αυτό σημαίνει από τα 2^{27} διαφορετικά μπλοκ, μεταφέρονται κάθε φορά τρία από αυτά σε ένα σύνολο. Αν το σύνολο περιέχει 4 πλαισία, αυτό σημαίνει από τα 2^{27} διαφορετικά μπλοκ, μεταφέρονται κάθε φορά τέσσερα από αυτά σε ένα σύνολο.

Παράδειγμα κρυφής μνήμης με οργάνωση μονοσήμαντης απεικόνισης

Έστω κρυφή μνήμη με οργάνωση μονοσήμαντης απεικόνισης με 2 λέξεις ανά πλαισίο και μπλοκ \rightarrow άρα $2^\mu = 2 \Rightarrow \mu = 1$ και 8 συνολικά πλαισία $2^\kappa = 8 \Rightarrow \kappa = 3 \rightarrow$. Η διεύθυνση που παράγει ο επεξεργαστής αποτελείται συνολικά από 8 ψηφία.

$$2^\kappa = 8 \text{ πλαισία} \Rightarrow \kappa = 3 \text{ bits και } 2^\mu = 2 \text{ λέξεις} \Rightarrow \mu = 1 \text{ bit}$$



Πλαισίο/μπλοκ

λέξη λέξη

0 1

To $v = 7 \rightarrow$ επομένως στην κύρια μνήμη έχουμε συνολικά 2^7 μπλοκ (blocks).

Επειδή το κάθε μπλοκ/πλαισίο αποτελείται από δύο λέξεις (θ.μ.), θα μεταφέρονται κάθε φορά στην κρυφή μνήμη από την κύρια, δύο διαδοχικές διευθύνσεις που είναι οι λέξεις που τελειώνουν σε «0» και «1». Αν η διεύθυνση που μεταφέρεται (block \rightarrow πλαισίο) τελειώνει σε «0» (δηλ. το μ τελειώνει σε «0»), τότε θα μεταφέρεται αυτή η διεύθυνση και η επόμενη της, ενώ αν τελειώνει σε «1», τότε θα μεταφέρεται αυτή η διεύθυνση και η προηγούμενη της.

0 80	1 81
42	43
3E	3F

Διεύθυνση (Hex)	Διεύθυνση (ετικέτα – πλαισίο – λέξη)	επ / απ (hit/miss)	Μεταφορά στην κρυφή μνήμη
80 = 1000 0000	1000 - 000 - 0	απ (miss)	MΠ(80 – 81) \rightarrow Π0
42 = 0100 0010	0100 - 001 - 0	απ	MΠ(42 – 43) \rightarrow Π1
F9 = 1111 1001	1111 - 100 - 1	απ	MΠ(F8 – F9) \rightarrow Π4
CA = 1100 1010	1100 - 101 - 0	απ	MΠ(CA – CB) \rightarrow Π5
80 = 10000000	1000 - 000 - 0	επ (hit)	-
3F = 00111111	0011 - 111 - 1	απ	MΠ(3E – 3F) \rightarrow Π7
59 = 01011001	0101 - 100 - 1	απ	MΠ(58 – 59) \rightarrow Π4
97 = 10010111	1001 - 011 - 1	απ	MΠ(96 – 97) \rightarrow Π3



58 = 01011000	0101 - 100 - 0	επ	-
1B = 00011011	0001 - 101 - 1	απ	MΠ(1A – 1B) → Π5
58 = 01011000	0101 - 100 - 0	επ	-
C3 = 1100 001 1	1100 - 001 - 1	απ (miss)	MΠ(C2 – C3) → Π1
F8 = 11111000	1111 - 100 - 0	απ (miss)	MΠ(F8 – F9) → Π4
1A = 00011010	0001 - 101 - 0	επ	-
0B = 00001011	0000 - 101 - 1	απ	MΠ(0A – 0B) → Π5

Για να βρούμε τα τελικά περιεχόμενα της κρυφής μνήμης, μετά την παραγωγή των παραπάνω διευθύνσεων, διατρέχουμε τον προηγούμενο πίνακα από το τέλος προς την αρχή του.

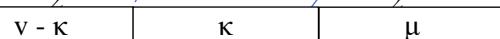
Τελικά περιεχόμενα κρυφής μνήμης

Πλαισίο 0 (Π0)	ΜΠ(80 – 81)
Πλαισίο 1	ΜΠ(C2 – C3)
Πλαισίο 2	-
Πλαισίο 3 (Π3)	ΜΠ(96 – 97)
Πλαισίο 4	ΜΠ(F8 - F9)
Πλαισίο 5	ΜΠ(0A – 0B)
Πλαισίο 6	ΜΠ(CC – CD)
Πλαισίο 7	ΜΠ(CE – CF)

Παρατήρηση: ΑΝ είχαμε 4 λέξεις/πλ. τότε $2^{\mu} = 4 \Rightarrow \mu = 2$ και κρυφή μνήμη αποτελούμενη από 32 λέξεις, θα είχαμε;

1 πλ. 4 λέξεις
x; 32 λέξεις

x = 8 πλ. Άρα το $2^k = 8 \Rightarrow k = 3$ bits. Έστω ότι το συνολικό μήκος της κάθε διεύθυνσης είναι 7 bits.



Έστω ότι δίνονται διευθύνσεις στο δεκαδικό, αποτελούμενες από 7 bits:

$33_{10} \rightarrow 01$	000 - 01	miss	ΜΠ(32-35) → Π0
$19 \rightarrow 00$	100 - 11	miss	ΜΠ(16-19) → Π4 ή ΜΠ(16, 17, 18, 19) → Π4.
$26 \rightarrow 00$	110 - 10	miss	ΜΠ(24-27) → Π6
00	01	10	11
			19

Θέμα με κρυφή μνήμη και χωρητικότητα μνήμης ετικετών

Ένα υπολογιστικό σύστημα έχει δίαυλο διευθύνσεων 20 δυαδικών ψηφίων ($A_{19} \dots A_0$).

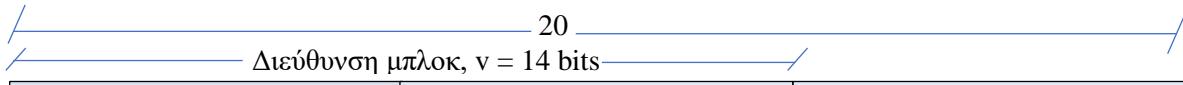
a. Το σύστημα διαθέτει κρυφή μνήμη άμεσης οργάνωσης (direct mapped) με χωρητικότητα 16 Kbytes και μέγεθος πλαισίου 64 λέξεις (1 λέξη = 1 byte). Ποια δυαδικά ψηφία της διεύθυνσης χρησιμοποιούνται για τον προσδιορισμό της λέξης του πλαισίου στο οποίο γίνεται η αναφορά, ποια για τον προσδιορισμό του πλαισίου και ποια ως ετικέτα;

β. Ένας πιο εξελιγμένο τύπος του ίδιου υπολογιστή διαθέτει κρυφή μνήμη 2-τρόπων συνόλου συσχέτισης (2-way set – associative) με χωρητικότητα 32 Kbytes και μέγεθος πλαισίου 32 λέξεις. Ποια δυαδικά ψηφία της διεύθυνσης χρησιμοποιούνται για τον προσδιορισμό της λέξης του πλαισίου στο οποίο γίνεται η αναφορά, ποια για τον προσδροιρισμό του συνόλου στο οποίο τοποθετείται το block και ποια ως ετικέτα;

γ. Ποια είναι η χωρητικότητα της μνήμης ετικετών των δύο αυτών συστημάτων;

Λύση

a. **Μονοσήμαντης απεικόνισης** Η ψηφιολέξη είναι ταυτόχρονα και λέξη, λόγω της οργάνωσης που δίνεται.



ετικέτα

Διεύθυνση πλαισίου

Διεύθυνση λέξης μέσα στο πλαισίο/μπλοκ

$2^{\mu} = 64$ λέξεις ($\theta \cdot \mu$) $\Rightarrow \mu = 6$ bits.



Για υπολογισμό πλαισίου: 1 πλ.

$$64 \text{ ψηφιολέξεις} = 64 \text{ bytes} = 64 \text{ θ.μ.}$$

$x \pi\lambda;$

16 KB

$$x = 2^{14}/2^6 = 2^8 \pi\lambda. = 2^\kappa \Rightarrow \kappa = 8 \text{ bits.}$$

β. 2 – τρόπων συνόλου συσχέτισης

Τώρα αλλάζει ο αριθμός των λέξεων ανά πλαίσιο/μπλοκ και γίνεται 32 από 64 που ήταν. Επομένως, το $2^\mu \geq 32 \Rightarrow \mu = 5 \text{ bits}$



$v - \lambda = 6 \text{ bits}$	$\lambda = 9 \text{ bits}$	$\mu = 5 \text{ bits}$
ετικέτα	Διεύθυνση συνόλου	Διεύθυνση λέξης μέσα στο πλαίσιο και το μπλοκ

Σημείωση: Από τη χωρητικότητα της κρυφής μνήμης – όταν δίνεται- υπολογίζουμε **πάντα** τον αριθμό των **πλαισίων**. Στη συνέχεια, από τα πλαισία, θα υπολογίσουμε τα σύνολα.

$$1 \text{ πλ.} \quad 32 \text{ λέξεις} = 32 \text{ bytes} \text{ (λόγω της οργάνωσης που δίνεται)}$$

$$x \pi\lambda; \quad 32 \text{ KB} \quad x = 2^{15}/2^5 = 2^{10} \text{ πλ.}$$

$$1 \text{ σύνολο} \quad 2 \text{ πλ.} \quad x = 2^{10}/2 = 2^9 \text{ σύνολα} = 2^\lambda \Rightarrow \lambda = 9 \text{ bits.}$$

γ. Μονοσήμαντη απεικόνιση: **Χωρητικότητα μνήμης ετικετών:** 6 bits/πλ. $x 2^8 \pi\lambda. = \dots$

2-τρόπων συνόλου συσχέτισης: **Χωρητικότητα μνήμης ετικετών:** 6 bits/πλ. $x 2^9 \text{ σύνολα} x 2 \text{ πλ./σύνολο} = 6 \text{ bits} x 2^{10}$

Θέμα με κρυφή μνήμη και στρατηγική αντικατάστασης LRU

Θεωρείστε ένα υπολογιστικό σύστημα με λέξεις των 16 δυαδικών ψηφίων και αρτηρία διευθύνσεων των 24 δυαδικών ψηφίων. Εφοδιάσαμε αυτό το σύστημα με κρυφή μνήμη 16 Kbytes, με οργάνωση συνόλου συσχέτισης με 2 πλαισία ανά σύνολο και μέγεθος πλαισίου ίσο με 4 λέξεις. Στην κρυφή μνήμη χρησιμοποιείται η τεχνική Least Recently Used για την αντικατάσταση πλαισίων.

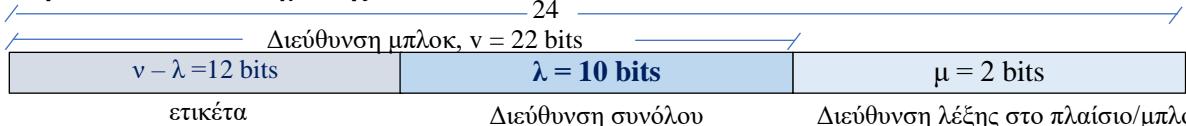
α) Ζητείται να υπολογίσετε το **ακριβές μέγεθος που καταλαμβάνει η κρυφή μνήμη σε Kbytes**. Θα πρέπει στον υπολογισμό να λάβετε υπ' όψη σας τη μνήμη ετικετών, ψηφία εγκυρότητας, κλπ. Θεωρείστε το μέγεθος κάθε συνδυαστικού κυκλώματος αμελητέο.

β) Έστω η ακολουθία 300 αναφορών στο σύστημα μνήμης που παράγει την ακολουθία διευθύνσεων F432B4₁₆, B892B7₁₆, AAC2B6₁₆, F432B4₁₆, B892B7₁₆, AAC2B6₁₆, ..., (100 δηλαδή επαναλήψεις των τριών πρώτων διευθύνσεων). Υποθέτοντας ότι ξεκινάμε με μια άδεια κρυφή μνήμη, υπολογίστε το ποσοστό επιτυχίας και εξηγείστε συνοπτικά.

γ) Υπολογίστε το ποσοστό επιτυχίας της **ίδιας ακολουθίας αναφορών** στη μνήμη αν η κρυφή μνήμη ήταν **άμεσης οργάνωσης** με όλα τα υπόλοιπα χαρακτηριστικά ίδια.

Λύση

(α) 2 – τρόπων συνόλου συσχέτισης



Σημείωση: Για να υπολογίσουμε το μ , δεν θα λάβουμε υπόψη μας το μέγεθος της κάθε λέξης, γιατί δεν επηρεάζει τον αριθμό των λέξεων που περιέχει κάθε πλαίσιο. Ο αριθμός αυτός είναι 4 λέξεις/πλαίσιο σύμφωνα με την εκφώνηση, επομένως έχουμε ότι $2^\mu = 4 \Rightarrow \mu = 2 \text{ bits}$.

$$1 \text{ πλ.} \quad 4 \text{ λέξεις} (\theta.μ.) = 4 \times 2 = 8 \text{ bytes}$$

$$x; \quad 16 \text{ KB}$$

Προσοχή! Στην άσκηση αυτή **δεν** ισχύει η συνήθης οργάνωση ότι κάθε θέση μνήμης (λέξη) είναι τους ενός byte. Ισχύει ότι κάθε θ.μ. (λέξη) είναι των 16 bits = 2 bytes. Το

$$x = 2^{14}/2^3 = 2^{11} = 2^\kappa \Rightarrow \kappa = 11 \text{ bits}$$

$$16 \text{ KB} = 2^4 2^{10} = 2^{14}$$

$$1 \text{ σύνολο} \quad 2 \text{ πλ.}$$

$$x = 2^{11}/2 = 2^{10} \text{ σύνολα} = 2^\lambda \Rightarrow \lambda = 10 \text{ bits.}$$

Θεωρούμε ότι ζητάμε τον υπολογισμό i) μνήμης ετικετών, ii) μνήμης των bit εγκυρότητας και iii) χωρητικότητας κρυφής μνήμης.

i) **Μόνο για τη μνήμη ετικετών** η χωρητικότητα είναι: $2^{10} \text{ σύνολα} \times 2 \frac{\pi\lambda.}{\text{σύνολο}} \times 12 \frac{\text{bits}}{\pi\lambda.} = 2^{11} \times 12 \text{ bits.}$

ii) **Μόνο για το bit εγκυρότητας** η χωρητικότητα είναι $2^{10} \text{ σύνολα} \times 2 \frac{\pi\lambda.}{\text{σύνολο}} \times 1 \frac{\text{bit}}{\pi\lambda.} = 2^{11} \times 1 \text{ bits.}$

iii) **Χωρητικότητα κρυφής μνήμης:** $2^{10} \text{ σύνολα} \times 2 \frac{\pi\lambda.}{\text{σύνολο}} \times 4 \frac{\lambdaέξεις}{\pi\lambda.} \times \frac{16 \text{ bits}}{\lambdaέξη} = 2^{17} \text{ bits} = 2^{14} \text{ bytes} = 16 \text{ KB}$

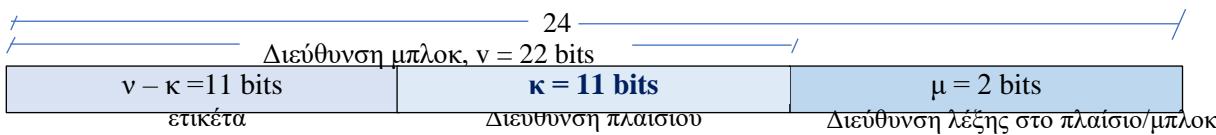
(β) 2 – τρόπων συνόλου συσχέτισης

Γράφουμε τις διευθύνσεις που μας δίνονται σε μορφή δυαδική:

	$v - \lambda$	λ	μ
F432B4	1111 0100 0011	0010 1011 01	00
B892B7	1011 1000 1001	0010 1011 01	11
AACB26	1010 1010 1100	0010 1011 01	10

Παρατηρούμε ότι και οι **τρείς διευθύνσεις** έχουν **την ίδια διεύθυνση συνόλου** (αλλά διαφορετική ετικέτα $v - \lambda$), επομένως έχουμε τρία **διαφορετικά** μπλοκ της κύριας μνήμης να μεταφέρονται στο **ίδιο σύνολο** της κρυφής μνήμης. Έχουμε ως αλγόριθμο αντικατάστασης τον LRU και αυτό σημαίνει όταν μεταφερθούν οι δύο πρώτες (διαφορετικές) διευθύνσεις στην κρυφή μνήμη, στο ίδιο σύνολο, τότε το σύνολο αυτό (που έχει διεύθυνση 0010101101 θα γεμίσει) και όταν έρθει η τρίτη διεύθυνση θα αντικαταστήσει την πρώτη και στη συνέχεια, σε κάθε επανάληψη η επόμενη κάθε φορά διεύθυνση της κύριας μνήμης θα αντικαθιστά μια από τις δύο διευθύνσεις που βρίσκονται ήδη μέσα στο σύνολο με διεύθυνση 0010101101. Συνεπώς θα έχουμε **συνεχώς αποτυχίες και το ποσοστό επιτυχίας θα είναι 0%**.

(γ) Μονοσήμαντη απεικόνιση



Γράφουμε τις διευθύνσεις που μας δίνονται σε μορφή δυαδική:

	$v - \kappa$	κ	μ
F432B4	1111 0100 001	10010 1011 01	00
B892B7	1011 1000 100	10010 1011 01	11
AACB26	1010 1010 110	00010 1011 01	10

Παρατηρούμε ότι μόνο τα δύο πρώτα μπλοκ της κύριας μνήμης μεταφέρονται στο **ίδιο πλαίσιο της κρυφής μνήμης** και για το λόγο αυτό **δεν** μπορούν να συνυπάρχουν στην κρυφή μνήμη, διότι έχουν διαφορετική ετικέτα $v - \kappa$, επομένως αναφέρονται σε διαφορετικά μπλοκ που μεταφέρονται στο ίδιο πλαίσιο της κρυφής μνήμης και η μια αντικαθιστά την άλλη. Αντίθετα το μπλοκ της κύριας μνήμης με διεύθυνση AACB26 μεταφέρεται σε **διαφορετικό πλαίσιο** της κρυφής μνήμης και μπορεί να συνυπάρχει με οποιοδήποτε από τις άλλες δύο διευθύνσεις. Άρα στις 100 επαναλήψεις (300 αναφορές) θα έχουμε 99 επιτυχίες, διότι κατά την προσπέλαση της διεύθυνσης AACB26 έχουμε αποτυχία την πρώτη φορά. Το **ποσοστό επιτυχίας** είναι $99/300 = 33\%$.

Θέμα 2019 με κρυφή μνήμη και δύο επεξεργαστές

Θεωρήστε δύο επεξεργαστές E1 και E2 κάθε ένας με ενοποιημένη κρυφή μνήμη (κοινή κρυφή μνήμη εντολών και δεδομένων) των 8192 θέσεων με μία ψηφιολέξη (byte) ανά θέση μνήμης και 16 ψηφιολέξεις ανά πλαίσιο και οργάνωση αντίστοιχα:

a. Μονοσήμαντης απεικόνισης (direct mapped)

b. 2-τρόπων συνόλου συσχέτισης (2-way associative)

Θεωρείστε ότι και οι δύο επεξεργαστές παράγουν διευθύνσεις των 24 δυαδικών ψηφίων και ότι κάθε θέση της κύριας μνήμης είναι της μιας ψηφιολέξης.

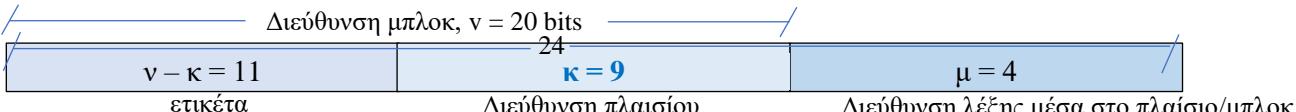
Για κάθε μία οργάνωση της κρυφής μνήμης να δώσετε:

i. Να δώσετε τα πεδία από τα οποία θεωρούμε ότι αποτελείται η διεύθυνση του επεξεργαστή για την προσπέλαση κάθε μιας των ανωτέρω κρυφών μνημών και τι δηλώνει κάθε πεδίο.

ii. Να αναφέρετε και να αιτιολογήστε τα περιεχόμενα ποιων από τις διευθύνσεις AED507₍₁₆₎ AFD7FA₍₁₆₎, AFD504₍₁₆₎, AFF7FF₍₁₆₎ AFF507₍₁₆₎ και AFD506₍₁₆₎ μπορούν να βρίσκονται ταυτόχρονα στην κρυφή μνήμη για κάθε μια από τις περιπτώσεις α και β.

Λύση

i) **E1 - Μονοσήμαντης απεικόνισης** Η ψηφιολέξη είναι ταυτόχρονα και λέξη, λόγω της οργάνωσης που δίνεται.



$2^\mu = 16$ ψηφιολέξεις (bytes) = 16 λέξεις ($\theta \cdot \mu$) $\Rightarrow \mu = 4$ bits. Για να υπολογίσουμε το μ , θα πρέπει να γνωρίζουμε τον αριθμό των λέξεων ($\theta \cdot \mu$) που περιέχει κάθε πλαίσιο/μπλοκ. Άρα, πρέπει να μετατρέψουμε τις ψηφιολέξεις που δίνονται, σε λέξεις.



Για υπολογισμό πλαισίων έχουμε: $1\pi\lambda$.

$16 \text{ bytes} = 16 \theta.\mu. (\text{λέξεις})$

$\times \pi\lambda;$

8192 θ.μ.

$$x = 2^{13}/2^4 = 2^9 \pi\lambda. = 2^\kappa \Rightarrow \kappa = 9 \text{ bits.}$$

$v - \kappa$ (11 bits)	κ (9 bits)	μ (4 bits)
a. AED507 = 10101110110	101010000	0111
β. AFD7FA = 1010111110	101111111	1010
γ. AFD504 = 10101111110	101010000	0100
δ. AFF7FF = 10101111111	101111111	1111
ε. AFF507 = 10101111111	101010000	0111
ζ. AFD506 = 10101111110	101010000	0110

Μεθοδολογία: Ξεκινάμε ελέγχοντας ποιες από τις παραπάνω διευθύνσεις έχουν το ίδιο κ (διεύθυνση πλαισίου).

Αυτές είναι οι διευθύνσεις (α), (γ), (ε), (ζ). Στη συνέχεια ελέγχουμε ποιες από αυτές έχουν ίδια ετικέτα ($v - \kappa$). Αυτές είναι οι διευθύνσεις (γ) και (ζ). Επομένως αυτές οι δύο διευθύνσεις μπορούν και πρέπει να είναι ταυτόχρονα στην κρυφή μνήμη, διότι έχουμε ίδιο μπλοκ (ίδιο v), το οποίο μεταφέρεται στο ίδιο πλαίσιο (ίδιο κ). Επομένως συμβολίζονται ως γ – ζ.

Οι διευθύνσεις (α) και (ε) δεν μπορούν να είναι ταυτόχρονα στην κρυφή μνήμη, τόσο μεταξύ τους όσο και με τις (γ) και (ζ), γιατί αναφέρονται σε διαφορετικά μπλοκ, που μετακινούνται στο ίδιο πλαίσιο και αναγκαστικά αντικαθιστούν η μία την άλλη. Τέλος Οι διευθύνσεις (β) και (δ) δεν μπορούν επίσης να είναι ταυτόχρονα μεταξύ τους στην κρυφή μνήμη, διότι αναφέρονται σε διαφορετικά μπλοκ που μετακινούνται στο ίδιο πλαίσιο (101111111) και συνεπώς αντικαθιστά και πάλι η μία την άλλη. Όμως, οποιαδήποτε από τις (β) ή (δ) μπορεί να βρίσκεται ταυτόχρονα στην κρυφή μνήμη με οποιαδήποτε από τις υπόλοιπες, διότι μεταφέρεται σε διαφορετικό πλαίσιο σε σχέση με αυτές.

ii) 2 – τρόπων συνόλου συσχέτισης

Διεύθυνση μπλοκ, $v = 20$ bits

$v - \lambda = 12$ bits = 3 hex ψηφία	$\lambda = 8$ bits = 2 hex ψηφία	$\mu = 4$ bits = 1 hex υποώρού
ετικέτα	Διεύθυνση συνόλου	Διεύθυνση λέξης μέσα στο πλαίσιο/μπλοκ

Από πριν έχουμε 512 πλαίσια, επομένως δοθέντος ότι έχουμε 2 – τρόπων συνόλου συσχέτισης μνήμη, ισχύει ότι:

1 σύνολο	2 πλ.
x;	512 πλ.

$$x = 2^9/2^4 = 2^8 = 2^\lambda \Rightarrow \lambda = 8 \text{ bits.}$$

Τώρα, επειδή τα μεγέθη πεδίων προέκυψαν πολλαπλάσια του 4, μπορούμε για λόγους ευκολίας να τα μετατρέψουμε σε δεκαεξαδικά ψηφία.

$v - \lambda$	λ	μ
a. AED507 = AED	50	7
β. AFD7FA = AFD	7F	A
γ. AFD504 = AFD	50	4
δ. AFF7FF = AFF	7F	F
ε. AFF507 = AFF	50	7
ζ. AFD506 = AFD	50	6

Μεθοδολογία: Ξεκινάμε ελέγχοντας ποιες από τις παραπάνω διευθύνσεις έχουν το ίδιο λ (διεύθυνση συνόλου).

Αυτές είναι οι διευθύνσεις (α), (γ), (ε), (ζ). Στη συνέχεια ελέγχουμε ποιες από αυτές έχουν ίδια ετικέτα ($v - \lambda$). Αυτές είναι οι διευθύνσεις (γ) και (ζ). Επομένως, αυτές οι δύο διευθύνσεις μπορούν και πρέπει να είναι ταυτόχρονα στην κρυφή μνήμη, διότι έχουμε ίδιο μπλοκ (ίδιο v), το οποίο μεταφέρεται στο ίδιο σύνολο (ίδιο λ) και μάλιστα στο ίδιο πλαίσιο του συνόλου αυτού. Επίσης, επειδή αυτές οι δύο διευθύνσεις καταλαμβάνουν -όπως αναφέρθηκε- το ίδιο πλαίσιο του συνόλου στο οποίο μεταφέρονται (50), είναι δυνατό, οποιαδήποτε από τις διευθύνσεις (α) ή (ε) να βρίσκεται ταυτόχρονα στην κρυφή μνήμη με τις (γ) και (ζ). Εναλλακτικά, μπορούν να συνυπάρχουν οι διευθύνσεις (α) και (ε) στο Σ50, χωρίς τις (γ) – (ζ), (α), (ε) μπορούν να συνυπάρχουν στην κρυφή μνήμη.



Τέλος οι διευθύνσεις **(β)** και **(δ)** μπορούν επίσης να είναι ταυτόχρονα μεταξύ τους στην κρυφή μνήμη, διότι παρά το ότι αναφέρονται σε διαφορετικά μπλοκ, μετακινούνται στο ίδιο σύνολο (7F) σε διαφορετικά πλαίσια του συνόλου αυτού και συνεπώς μπορούν να βρίσκονται ταυτόχρονα στην κρυφή μνήμη, τόσο μεταξύ τους, όσο και με τις υπόλοιπες.

Θέμα Ιοννίου 2015 με κρυφή μνήμη

Δίνεται επεξεργαστής που δεν υποστηρίζει ιδεατή μνήμη (virtual memory), οργάνωση κύριας μνήμης μιας ψηφιολέξης ανά θέση μνήμης (σε κάθε ψηφιολέξη αντιστοιχεί μια διεύθυνση), **αρτηρία διευθύνσεων των 32 δυαδικών ψηφίων** και κρυφή μνήμη χωρητικότητας **16 Kbytes** (16 Κυψηφιολέξεις) με **16 ψηφιολέξεις ανά πλαίσιο**. Για κάθε μία των κάτωθι περιπτώσεων:

- a. οργάνωση κρυφής μνήμης **μονοσήμαντης απεικόνισης** (direct mapped)
 - β. οργάνωση κρυφής μνήμης **2-τρόπων συσχέτισης** (2-way set associative)
 - γ. οργάνωση κρυφής μνήμης **4-τρόπων συσχέτισης** (4-way set associative)
 - δ. **πλήρους συσχέτισης** (full associative).
- i. Να δώσετε τον τρόπο με τον οποίο χρησιμοποιείται η διεύθυνση που παράγει ο επεξεργαστής.
 - ii. Να απαντήσετε και να δικαιολογήσετε ποια από τα δεδομένα των διευθύνσεων (στο δεκαεξαδικό) Δ1: AB34852A, Δ2: 034588529, Δ3: AB348527, Δ4: A671852A, Δ5: B6718ACA, Δ6: DE71889A, Δ7: A67184DA είναι δυνατόν να βρίσκονται ταυτόχρονα στην κρυφή μνήμη.

Λύση

Κανόνας: Για να υπολογίσουμε τα μεγέθη των πεδίων των διευθύνσεων, όχι μόνο της μονοσήμαντης απεικόνισης αλλά και των υπολοπών οργανώσεων, θα πρέπει πάντα να **μετατρέπουμε τις χωρητικότητες σε θ.μ. (λέξεις). Το πλήθος των πλαισίων της κρυφής μνήμης, αν δίνεται απευθείας, το βρίσκουμε από τη χωρητικότητα της κρυφής μνήμης. Αφού βρούμε το πλήθος των πλαισίων, στη συνέχεια θα υπολογίσουμε το πλήθος των συνόλων, για οργανώσεις τ – τρόπων συνόλου συσχέτισης.**

i) Μονοσήμαντη απεικόνιση Η ψηφιολέξη είναι ταυτόχρονα και λέξη (θ.μ.), λόγω της οργάνωσης που δίνεται.

Διεύθυνση μπλοκ, $v = 28 \text{ bits}$		
$v - k = 18$	$k = 10$	$\mu = 4$
ετικέτα	Διεύθυνση πλαισίου	Διεύθυνση λέξης μέσα στο πλαίσιο/μπλοκ
$x \lambda;$		16 KB
$x = 2^{14}/2^4 = 2^{10} \text{ πλ.} = 2^k \Rightarrow k = 10 \text{ bits.}$		
Για υπολογισμό πλαισίων έχουμε: $1\pi\lambda \cdot 16 \text{ ψηφιολέξεις} = 16 \text{ bytes} = 16 \text{ θ.μ. (λέξεις)}$		

$x = 2^{14}/2^4 = 2^{10} \text{ πλ.} = 2^k \Rightarrow k = 10 \text{ bits.}$ Οι διευθύνσεις που δίνονται είναι εκφρασμένες στο δεκαεξαδικό σύστημα. Αν το επιτρέπει το μέγεθος των επιμέρους πεδίων της διεύθυνσης, μετατρέπουμε τα πεδία αυτά σε πολλαπλάσια του «4», αν όχι, αναλόουμε τις διευθύνσεις σε μορφή δυαδική.

	$v - k$ (18 bits)	k (10 bits)	μ (4 bits)
Δ1. AB34852A =	01010110011010010	0001010010	1010
Δ2. 34588529 =	001101000101100010	0001010010	1001
Δ3. AB348527 =	101010110011010010	0001010010	0111
Δ4. A671852A =	101001100111000110	0001010010	1010
Δ5. B6718ACA =	101101100111000110	0010101100	1010
Δ6. DE71889A =	110111100111000110	0010001001	1010
Δ7. A67184DA =	101001100111000110	0001001101	1010

Μεθοδολογία: Ξεκινάμε ελέγχοντας ποιες από τις παραπάνω διευθύνσεις έχουν το **ίδιο κ (διεύθυνση πλαισίου)**.

Αντές είναι οι διευθύνσεις **(Δ1), (Δ2), (Δ3) και (Δ4)**. Στη συνέχεια ελέγχουμε ποιες από αυτές έχουν ίδια ετικέτα ($v - k$). Αυτές είναι οι διευθύνσεις **(Δ1) και (Δ3)**. Επομένως, αυτές οι δύο διευθύνσεις μπορούν και πρέπει να είναι ταυτόχρονα στην κρυφή μνήμη, διότι έχουμε **ίδιο μπλοκ (ίδιο v)**, το οποίο μεταφέρεται στο **ίδιο πλαίσιο (ίδιο κ)**. Εφεξής, αυτές οι δύο διευθύνσεις θα αναγράφονται ως $\Delta 1 - \Delta 3$. Οι διευθύνσεις **(Δ2) και (Δ4)** δεν μπορούν να είναι ταυτόχρονα στην κρυφή μνήμη, τόσο μεταξύ τους όσο και με τις **(Δ1) και (Δ3)**, γιατί αναφέρονται σε διαφορετικά μπλοκ, που μετακινούνται στο **ίδιο πλαίσιο** και αναγκαστικά αντικαθιστούν η μία την άλλη.

Τέλος, οι διευθύνσεις (Δ5) (Δ6) και (Δ7) μπορούν επίσης να είναι ταυτόχρονα μεταξύ τους στην κρυφή μνήμη, τόσο μεταξύ τους όσο και σε σχέση με τις υπόλοιπες, διότι αναφέρονται σε **διαφορετικά μπλοκ** που μετακινούνται σε διαφορετικά πλαίσια.

ii) 2 – τρόπων συνόλου συσχέτισης

Διεύθυνση μπλοκ, $v = 28$ bits		
$v - \lambda = 19$ bits	$\lambda = 9$ bits	$\mu = 4$ bits = 1 hex ψηφίο

ετικέτα Διεύθυνση συνόλου Διεύθυνση λέξης μέσα στο πλαίσιο και το μπλοκ

Από πριν έχουμε 2^{10} πλαίσια επομένως δοθέντος ότι έχουμε 2 – τρόπων συνόλου συσχέτισης μνήμη, ισχύει ότι:

$$\begin{array}{l} 1 \text{ σύνολο} \\ x; \\ x = 2^{10}/2^1 = 2^9 = 2^\lambda \Rightarrow \lambda = 9 \text{ bits.} \end{array}$$

	$v - \lambda$ (19 bits)	λ (9bits)	μ (4 bits)
Δ1. AB34852A =	1010101100110100100	001010010	1010
Δ2. 34588529 =	0011010001011000100	001010010	1001
Δ3. AB348527 =	1010101100110100100	001010010	0111
Δ4. A671852A =	1010011001110001100	001010010	1010
Δ5. B6718ACA =	1011011001110001100	010101100	1010
Δ6. DE71889A =	1101111001110001100	010001001	1010
Δ7. A67184DA =	1010011001110001100	001001101	1010

Μεθοδολογία: Ξεκινάμε ελέγχοντας ποιες από τις παραπάνω διευθύνσεις έχουν το **ίδιο λ** (διεύθυνση συνόλου).

Αντές είναι οι διευθύνσεις (Δ1), (Δ2), (Δ3) και (Δ4). Στη συνέχεια ελέγχουμε ποιες από αυτές έχουν ίδια ετικέτα ($v - \lambda$). Αυτές είναι οι διευθύνσεις (Δ1) και (Δ3). Επομένως, αυτές οι δύο διευθύνσεις μπορούν και πρέπει να είναι ταυτόχρονα στην κρυφή μνήμη, διότι έχουμε **ίδιο μπλοκ (ίδιο v)**, το οποίο μεταφέρεται στο **ίδιο σύνολο (ίδιο λ)** και **μάλιστα στο ίδιο πλαίσιο του συνόλου αυτού, π.χ. στο Π0**. Εφεξής, αυτές οι δύο διευθύνσεις θα αναγράφονται ως Δ1 – Δ3.

Οι διευθύνσεις (Δ2) και (Δ4) αναφέρονται σε **διαφορετικά μπλοκ**, που μετακινούνται στο **ίδιο σύνολο** σε σχέση με τις Δ1 – Δ3.

Από αυτές τις τρεις διευθύνσεις, **μόνο δύο κάθε φορά μπορούν** να είναι **ταυτόχρονα** στην κρυφή μνήμη.

Τέλος, οι διευθύνσεις (Δ5) (Δ6) και (Δ7) μπορούν επίσης να είναι ταυτόχρονα μεταξύ τους στην κρυφή μνήμη, τόσο μεταξύ τους όσο και σε σχέση με τις υπόλοιπες, διότι αναφέρονται σε **διαφορετικά μπλοκ** που μετακινούνται σε διαφορετικά πλαίσια.

γ) 4 – τρόπων συνόλου συσχέτισης

Διεύθυνση μπλοκ, $v = 28$ bits = 7 hex ψηφία		
$v - \lambda = 20$ bits = 5 hex ψηφία	$\lambda = 8$ bits = 2 hex ψηφία	$\mu = 4$ bits = 1 hex ψηφίο

ετικέτα Διεύθυνση συνόλου Διεύθυνση λέξης στο πλαίσιο/μπλοκ

Από πριν έχουμε 2^{10} πλαίσια επομένως δοθέντος ότι έχουμε 4 – τρόπων συνόλου συσχέτισης μνήμη, ισχύει ότι:

$$\begin{array}{l} 1 \text{ σύνολο} \\ x; \\ x = 2^{10}/2^2 = 2^8 = 2^\lambda \Rightarrow \lambda = 8 \text{ bits.} \end{array}$$

	$v - \lambda$ (20 bits)	λ (8bits)	μ (4 bits)
Δ1.	AB348	52	A
Δ2.	345885	52	9
Δ3.	AB348	52	7
Δ4.	A6718	52	A
Δ5.	B6718	AC	A
Δ6.	DE718	89	A
Δ7.	A6718	4D	A

Μεθοδολογία: Ξεκινάμε ελέγχοντας ποιες από τις παραπάνω διευθύνσεις έχουν το **ίδιο λ** (διεύθυνση συνόλου).

Μπορούν όλες οι διευθύνσεις να είναι ταυτόχρονα στην κρυφή μνήμη, διότι κάθε σύνολο αποτελείται στην περίπτωση αυτή από 4 πλαίσια. Μάλιστα επειδή οι Δ1 – Δ3 πηγαίνουν (μεταφέρονται) στο ίδιο πλαίσιο του συνόλου, αυτό σημαίνει ότι απομένει ένα ελεύθερο πλαίσιο στο σύνολο 52. Οι διευθύνσεις Δ4, Δ5, Δ6 μεταφέρονται ούτως ή άλλως σε διαφορετικά σύνολα.

δ) Πλήρους συσχέτισης



Διεύθυνση μπλοκ

ν = 28 bits = 7 hex ψηφία

μ = 4 bits

ετικέτα

Διεύθυνση λέξης μέσα στο πλαίσιο και το μπλοκ

ν (28bits)

μ (4 bits)

Δ1.	AB34852
Δ2.	3458852
Δ3.	AB34852
Δ4.	A671852
Δ5.	B6718AC
Δ6.	DE71889
Δ7.	A67184D

A

9

7

A

A

A

A

Μπορούν όλες οι διευθύνσεις να είναι ταυτόχρονα στην κρυφή μνήμη, στην περίπτωση της πλήρους οργάνωσης οποιοδήποτε μπλοκ μπορεί να μετακινηθεί σε οποιοδήποτε πλαίσιο και μάλιστα επειδή και πάλι οι διευθύνσεις Δ1 – Δ3 αναφέρονται στο ίδιο πλαίσιο, από τα 2^{10} πλαίσια της κρυφής μνήμης γεμίζουν μόνο τα έξι πρώτα.

Θέμα Φεβρουαρίου 2012 με κρυφή μνήμη

Θεωρείστε σύστημα υπολογιστή με αρτηρία διευθύνσεων των 32 δυαδικών ψηφίων, σύστημα μνήμης με μια ψηφιολέξη ανά θέση μνήμης (δηλαδή μια διεύθυνση αντιστοιχεί σε κάθε ψηφιολέξη) και κρυφή μνήμη με χωρητικότητα 1 Mψηφιολέξεις (1Mbytes) και πλαίσιο των 64 ψηφιολέξεων. Να σχεδιάσετε σε μπλοκ διάγραμμα την κρυφή μνήμη για καθεμία από τις ακόλουθες οργανώσεις και να υπολογίσετε το υλικό (μνήμη, πύλες, πολυπλέκτες κ.λπ.) που απαιτείται για κάθε υλοποίηση. Όλη η απαιτούμενη πληροφορία να φαίνεται στο σχήμα.

a. Οργάνωση **μονοσήμαντης απεικόνισης** (direct mapped).

β. Οργάνωση 2-τρόπων συνόλου συσχέτισης (2-way set associative) (Μονάδα: 1) Για κάθε μία από τις ανωτέρω οργανώσεις να εξετάσετε ποιες από τις πληροφορίες που βρίσκονται αποθηκευμένες στις διευθύνσεις 00601F4C, 00681F4E και 00601F4F μπορούν να βρίσκονται ταυτόχρονα στην κρυφή μνήμη.

Λύση

Αφού η χωρητικότητα της κρυφής μνήμης είναι $1 \text{ Mbytes} = 2^{20} \text{ bytes}$ και κάθε πλαίσιο είναι των $64 \text{ bytes} = 2^6 \text{ bytes}$ 2^μ , συνεπάγεται ότι η κρυφή μνήμη έχει $2^{20}/2^6 = 2^{14} = 2^k$ πλαίσια. Αφού το σύστημα μνήμης έχει μία ψηφιολέξη (byte) ανά θέση μνήμης και το πλαίσιο είναι των $64=2^6$ ψηφιολέξεων, τα 6 λιγότερο σημαντικά δυαδικά ψηφία της διεύθυνσης χρησιμοποιούνται για να δηλώσουν τη διεύθυνση της λέξης μέσα στο πλαίσιο.

a. Οργάνωση **μονοσήμαντης απεικόνισης** (direct mapped)

Αφού η κρυφή μνήμη έχει 2^{14} πλαίσια, τα επόμενα 14 δυαδικά ψηφία χρησιμοποιούνται για να δηλώσουν τη διεύθυνση του πλαισίου. Τα υπόλοιπα $32 - 14 - 6 = 12$ δυαδικά ψηφία αποτελούν την ετικέτα.

Για την υλοποίηση αυτής της κρυφής μνήμης απαιτείται μνήμη χωρητικότητας 2^{14} πλαίσια \times (64 δυαδικά ψηφία ανά πλαίσιο) για την αποθήκευση των δεδομένων + 12 δυαδικά ψηφία ανά πλαίσιο για την αποθήκευση των ετικετών + 1 δυαδικό ψηφίο ανά πλαίσιο για την αποθήκευση του δυαδικού ψηφίου εγκυρότητας) $= 2^{14}$ πλαίσια \times 77 δυαδικά ψηφία ανά πλαίσιο. $= 2^{14}$ πλαίσια \times 77 bits. Επίσης απαιτείται ένας συγκριτής των 12 δυαδικών ψηφίων, μία πύλη AND, 8 πολυπλέκτες 64 \rightarrow 1 και 8 στοιχεία τριών καταστάσεων.

00601F4C(16): διεύθυνση **α**

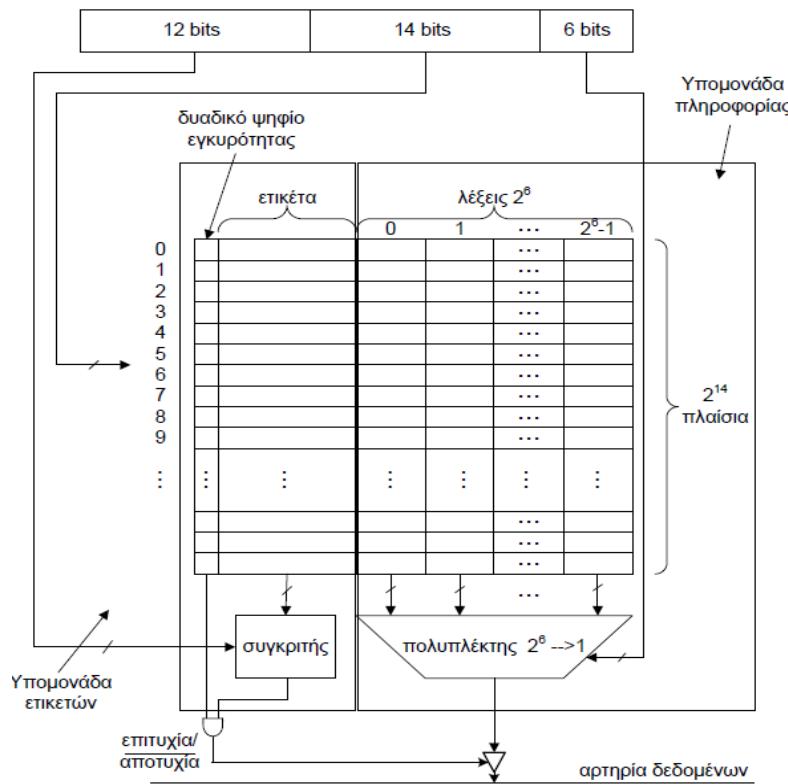
0000 0000 0110 0000 0001 1111 0100 1100

00681F4E(16): διεύθυνση **β**

0000 0000 0110 1000 0001 1111 0100 1110

00601F4F(16): διεύθυνση **γ**

0000 0000 0110 0000 0001 1111 0100 1111



Οι διευθύνσεις α και γ ανήκουν στο ίδιο μπλοκ της κύριας μνήμης (έχουν την ίδια διεύθυνση πλαισίου και την ίδια ετικέτα), άρα όταν μεταφέρεται το περιεχόμενο του μπλοκ στην κρυφή μνήμη θα βρίσκονται σ' αυτή τα περιεχόμενα και των δύο διευθύνσεων. Η διεύθυνση β έχει διαφορετική διεύθυνση πλαισίου από τις προηγούμενες, άρα και αυτής τα περιεχόμενο μπορεί να βρίσκεται ταυτόχρονα στην κρυφή μνήμη με τα περιεχόμενα των άλλων δύο.

β. Οργάνωση 2-τρόπων συνόλου συσχέτισης (2-way set associative)

Αφού η κρυφή μνήμη έχει 2^{14} πλαισία και 2 πλαισία ανά σύνολο συνεπάγεται ότι έχει 2^{13} σύνολα. Επομένως μετά τα 6 λιγότερο σημαντικά δυαδικά ψηφία της διεύθυνσης τα οποία δηλώνουν τη διεύθυνση της λέξης μέσα στο πλαίσιο, τα 13 δυαδικά ψηφία θα δηλώνουν τη διεύθυνση συνόλου και τα υπόλοιπα $32-13-6=13$ δυαδικά ψηφία θα αποτελούν την ετικέτα. Για την υλοποίηση αυτής της κρυφής μνήμης απαιτείται μνήμη χωρητικότητας: 2^{13} πλαισία $\times 2 \times (64 \text{ δυαδικά ψηφία ανά πλαίσιο} + 13 \text{ δυαδικά ψηφία ανά πλαίσιο για την αποθήκευση των ετικετών} + 1 \text{ δυαδικό ψηφίο ανά πλαίσιο για την αποθήκευση του δυαδικού ψηφίου εγκυρότητας} + \text{το υλικό που απαιτείται για τη υλοποίηση της τεχνικής αντικατάστασης πλαισίων της κρυφής μνήμης (το υλικό αυτό δεν φαίνεται στο σχήμα})$. Επίσης απαιτούνται δύο συγκριτές των 13 δυαδικών ψηφίων ο κάθε ένας, δύο πύλες AND, 2×8 πολυπλέκτες $64 \rightarrow 1$ και 2×8 στοιχεία τριών καταστάσεων.

00601F4C(16): διεύθυνση α

0000 0000 0110 0000 0001 1111 0100 1100

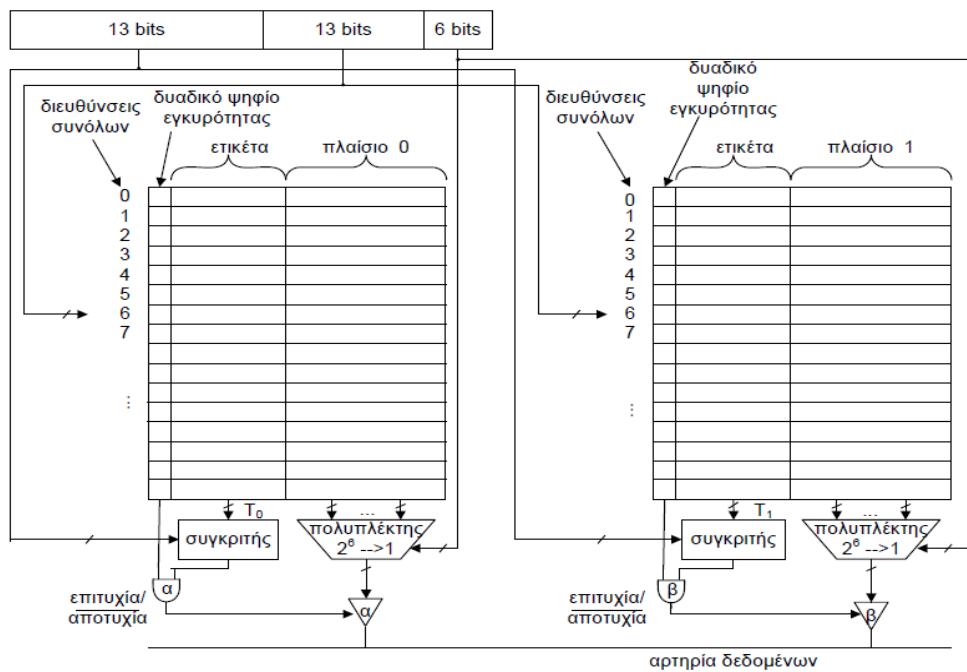
00681F4E(16): διεύθυνση β

0000 0000 0110 1000 0001 1111 0100 1110

00601F4F(16): διεύθυνση γ

0000 0000 0110 0000 0001 1111 0100 1111

Οι διευθύνσεις α και γ ανήκουν στο ίδιο μπλοκ της κύριας μνήμης (έχουν την ίδια διεύθυνση συνόλου και την ίδια ετικέτα), άρα όταν μεταφέρεται το περιεχόμενο του μπλοκ στην κρυφή μνήμη θα βρίσκονται σ' αυτή τα περιεχόμενα και των δύο διευθύνσεων. Η διεύθυνση β έχει διαφορετική διεύθυνση μπλοκ από τις προηγούμενες αλλά έχει την ίδια διεύθυνση συνόλου και διαφορετική ετικέτα από τις προηγούμενες. Επομένως αντιστοιχεί στο ίδιο σύνολο, αλλά αφού κάθε σύνολο έχει δύο πλαισία μπορεί να βρίσκεται το περιεχόμενό της ταυτόχρονα με τα περιεχόμενα των άλλων διευθύνσεων στην κρυφή μνήμη.



Άσκηση 5.8 με κρυφή μνήμη

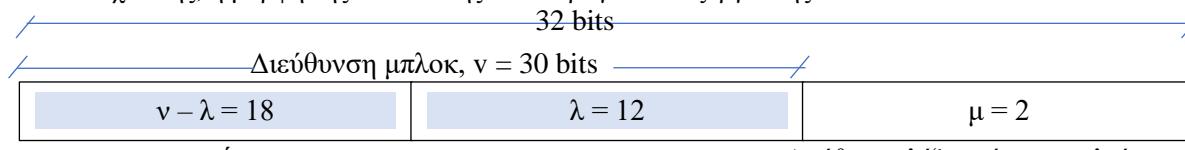
Θεωρήστε ένα υπολογιστή με εύρος αρτηρίας διευθύνσεων των 32 δυαδικών ψηφίων και ότι η κύρια μνήμη είναι οργανωμένη σε λέξεις των 16 δυαδικών ψηφίων ανά θέση μνήμης, θεωρήστε επίσης ότι η κρυφή μνήμη έχει οργάνωση 4-τρόπων συνόλου συσχέτισης με χωρητικότητα 64 Κλέξεις και τέσσερις λέξεις των 16 δυαδικών ψηφίων ανά πλαίσιο.

- Πόσα μπλοκ της κύριας μνήμης μπορούν να αποθηκευτούν στο ίδιο σύνολο της κρυφής μνήμης (προφανώς όχι ταυτόχρονα);
- Προσδιορίστε το σύνολο της κρυφής μνήμης στο οποίο μπορεί να αποθηκευτεί η λέξη που είναι αποθηκευμένη στη θέση με διεύθυνση 960F8.

γ. Προσδιορίστε τις διευθύνσεις όλων των μπλοκ της κύριας μνήμης που μπορούν να αποθηκευτούν στο σύνολο με διεύθυνση 60₍₆₎.

Λύση

α) Για να απαντήσουμε στο πρώτο ερώτημα, θα πρέπει να υπολογίσουμε από τη μια τον αριθμό των συνόλων της κρυφής μνήμης και από την άλλη τον αριθμό των μπλοκ της κύριας μνήμης. Όσον αφορά την κρυφή μνήμη, επειδή έχει οργάνωση 4 – τρόπων συνόλου συσχέτισης, η μορφή της διεύθυνσης που παράγει ο επεξεργαστής είναι:



$$v = 30 \text{ bits} \rightarrow 2^v \text{ μπλοκ} = 2^{30} \text{ μπλοκ (κύρια μνήμη)}$$

$2^\mu = \text{αριθμός λέξεων ανά πλαίσιο/μπλοκ} = 4 \Rightarrow \mu = 2$. Προσοχή! Η πληροφορία ότι κάθε λέξη (θ.μ.) είναι των 16 bits δεν έχει καμία απολύτως χρήση. Εμάς μας ενδιαφέρει το πλήθος των λέξεων που υπάρχουν ανά πλαίσιο ή μπλοκ και όχι το μέγεθος κάθε λέξης. Ακόμη και αν το μέγεθος κάθε λέξης ήταν 4, 8 ή κάποιος άλλος αριθμός με bits, το γεγονός ότι περιέχονται 4 λέξεις ανά πλαίσιο/μπλοκ, σημαίνει ότι σε κάθε περίπτωση το $\mu = 2$. Για να υπολογίσουμε στη συνέχεια τον αριθμό των συνόλων, δηλ. το 2^λ , πρέπει πρώτα να υπολογίσουμε τον αριθμό των πλαισίων, και αυτό το μέγεθος υπολογίζεται από τη χωρητικότητα της κρυφής μνήμης που δίνεται, ως εξής:

$$\begin{array}{l|l} 1 \text{ πλ. } 4 \text{ λέξεις } & \\ \hline x; \quad 64K \text{ λέξεις } & x = (2^6 * 2^{10})/2^2 = 2^{16}/2^2 = 2^{14} \text{ πλαισια} = 2^\kappa. \end{array}$$

Υπενθύμιση: $1 K = 2^{10}$, $1 M = 2^{20}$, $1 G = 2^{30}$. Από τη στιγμή που έχουμε 4 – τρόπων συνόλου συσχέτισης κρυφή μνήμη, το κάθε σύνολο αποτελείται από 4 πλαισία. Γιαυτό έχουμε ότι:

$$\begin{array}{l|l} 1 \text{ σύνολο } & 4 \text{ πλ. } \\ \hline x; \quad 2^{14} \text{ πλ. } & x = (2^{14})/2^2 = 2^{12} \text{ σύνολα} = 2^\lambda \text{ σύνολα} \end{array}$$

Σε 2^{12} σύνολα μεταφέρονται 2^{30} μπλοκ

1 σύνολο

x:

$$x = 2^{30}/2^{12} = 2^{18} \text{ μπλοκ/σύνολο, αυτά μεταφέρονται (προφανώς όχι ταυτόχρονα).}$$

β) Αναλύουμε τη διεύθυνση που δίνεται σε μορφή δυαδική: $960F8 = 1001\ 01\ 10\ 0000\ 1111\ 10\ 00$. Άρα το $\lambda = 1000\ 0011\ 1110 = 83E$ H = $0x83E = 83E_{16}$.

γ) Η διεύθυνση συνόλου είναι $\lambda = 12 \text{ bits} = 3 \text{ δεκαεξαδικά ψηφία}$ και επειδή δίνεται το σύνολο **60** H = **060** H. Η εκφώνηση μας ζητά να υπολογίσουμε **διευθύνσεις μπλοκ**, όπου κάθε διεύθυνση μπλοκ αποτελείται από 30 bits, που ως μέγεθος **δεν** είναι πολ/σιο του 4. Για το λόγο αυτό, θα θεωρήσουμε ότι οι διευθύνσεις είναι της μορφής **AXYZW060**, όπου X, Y, Z, W = τυχαία δεκαεξαδικά ψηφία και A = τυχαίο ψηφίο των 2 bits. **Εναλλακτικά**, μπορούμε να γράψουμε την μορφή των διευθύνσεων σε δυαδική μορφή: xxxxxxxx.....xxxx000001100000 (30 bits).

Ασκηση 5.10 με κρυφή μνήμη

Θεωρείστε τρεις κρυφές μνήμες των 512 πλαισίων με 8 λέξεις ανά πλαισίο και οργανώσεις:

i. Μονοσήμαντης απεικόνισης

ii. Πλήρους συσχέτισης

iii. 8 – τρόπων συνόλου συσχέτισης

α. Να δώσετε στο οκταδικό τις διευθύνσεις των πλαισίων της κρυφής μνήμης στα οποία μπορούν να αποθηκευτούν τα περιεχόμενα των θέσεων της κύριας μνήμης με διευθύνσεις $117165_8, 117167_8, 445231_8, 575232_8, 675253_8, 677335_8$ σε καθεμία από τις περιπτώσεις i, ii, iii.

β. Ποια από τα μπορούν περιεχόμενα των ανωτέρω θέσεων μνήμης είναι δυνατό να βρίσκονται ταυτόχρονα στην κρυφή μνήμη σε καθεμία από τις περιπτώσεις i, ii, iii και γιατί

Λύση

i) Μονοσήμαντης απεικόνισης

Διεύθυνση μπλοκ, v = 15 bits

$v - \kappa = 6 \text{ ή } 2$ οκταδικά ψηφία ετικέτα	$\kappa = 9 \text{ ή } 3$ οκταδικά ψηφία Διεύθυνση πλαισίου	$\mu = 3 \text{ bits ή } 1$ οκταδικό ψηφίο Διεύθυνση λέξης μέσα στο πλαισίο/μπλοκ
---	--	--

$$2^{\kappa} = 512 \text{ πλαισία} \Rightarrow \kappa = 9 \text{ bits} \text{ και } 2^{\mu} = 8 \text{ λέξεις/πλαισίο} \Rightarrow \mu = 3 \text{ bits}$$

Οταν η εκφώνηση δεν αναφέρει το ακριβές μέγεθος (μήκος) μιας διεύθυνσης, τότε αυτό το παίρνουμε από τις διευθύνσεις που δίνονται. Για παράδειγμα, $117165_8 = 18$ bits κ.ο.κ. Επομένως, αφού το συνολικό μήκος της διεύθυνσης είναι 18 δυαδικά ψηφία, το $v - \kappa = 6$ ($18 - 12$). Επειδή όλα τα πεδία της διεύθυνσης έχουν μήκος που είναι ακέραιο πολ/σιο του 3, αυτό σημαίνει ότι $v - \kappa = 6$ bits = 2 οκταδικά ψηφία, το $\kappa = 9$ bits = 3 οκταδικά ψηφία και το $\mu = 3$ bits = 1 οκταδικό ψηφίο.

$$117165 = [11\ 716] 5 \rightarrow \text{Πλ. } 716 \quad \text{Μπορούν και πρέπει να είναι ταυτόχρονα στην κρυφή μνήμη, αφού πρόκειται για ίδιο μπλοκ (με διεύθυνση 11716) που μεταφέρεται στο ίδιο πλαισίο (με διεύθυνση 716).}$$

$$117167 = [11\ 716] 7 \rightarrow \text{Πλ. } 716 \quad \text{Δεν μπορούν να είναι ταυτόχρονα στην κρυφή μνήμη, αφού πρόκειται για διαφορετικά μπλοκ}$$

$$445231 = [44\ 523] 1 \rightarrow \text{Πλ. } 523 \quad \text{που μεταφέρονται στο ίδιο πλαισίο (με διεύθυνση 523).}$$

$$575232 = [57\ 523] 2 \rightarrow \text{Πλ. } 523 \quad \text{που μεταφέρονται στο ίδιο πλαισίο (με διεύθυνση 523).}$$

$$675253 = [67\ 525] 3 \rightarrow \text{Πλ. } 525 \quad \text{Μπορούν να είναι ταυτόχρονα στην κρυφή μνήμη τόσο μεταξύ τους, αφού πρόκειται για διαφορετικά μπλοκ που μεταφέρονται σε διαφορετικά πλαισία, όσο και με τα προηγούμενα μπλοκ.}$$

$$677335 = [67\ 733] 5 \rightarrow \text{Πλ. } 733 \quad \text{Μπορούν να είναι ταυτόχρονα στην κρυφή μνήμη τόσο μεταξύ τους, αφού πρόκειται για διαφορετικά μπλοκ που μεταφέρονται σε διαφορετικά πλαισία, όσο και με τα προηγούμενα μπλοκ.}$$

ii) Πλήρους συσχέτισης

$v = 15 \text{ bits ή } 5$ οκταδικά ψηφία Διεύθυνση μπλοκ	$\mu = 3 \text{ bits ή } 1$ οκταδικό ψηφίο Διεύθυνση λέξης μέσα στο πλαισίο/μπλοκ
--	--

Τώρα που έχουμε **οργάνωση πλήρους συσχέτισης, τα πλαισία της κρυφής μνήμης γεμίζουν με τη σειρά από το πρώτο προς το τελευταίο**. Απλώς επειδή οι δύο πρώτες διευθύνσεις αναφέρονται στο ίδιο μπλοκ, με διεύθυνση 11716, μεταφέρονται στο ίδιο πλαισίο της κρυφής μνήμης. Επειδή από τη μία έχουμε στη διάθεσή μας 512 πλαισία και από την άλλη μόνο 6 διευθύνσεις που μεταφέρονται σε αυτά, σημαίνει ότι μπορούν όλες να είναι ταυτόχρονα στην κρυφή μνήμη. Από τα 512 πλαισία της κρυφής μνήμης γεμίζουν μόνο τα 5 πλαισία.

$$117165 = 11716\ 5 \rightarrow \text{Πλ. } 0$$

$$117167 = 11716\ 7 \rightarrow \text{Πλ. } 0$$

$$445231 = 44523\ 1 \rightarrow \text{Πλ. } 1$$

575232 = **57523** 2 → Πλ. 2

675253 = **67525** 3 → Πλ. 3

677335 = **67733** 5 → Πλ. 4

iii) 8-τρόπων συνόλου συσχέτισης

Διεύθυνση μπλοκ, $v = 15$ bits

$v - \lambda = 9$ bits ή 3 οκταδικά ψηφία	$\lambda = 6$ ή 2 οκταδικά ψηφία	$\mu = 3$ bits ή 1 οκταδικό ψηφίο
ετικέτα	Διεύθυνση συνόλου	Διεύθυνση λέξης μέσα στο πλαίσιο/μπλοκ

Από πριν έχουμε **512 πλαίσια**, επομένως δοθέντος ότι έχουμε **8 – τρόπων συνόλου συσχέτισης μνήμη**, ισχύει ότι:

1 σύνολο 8 πλ.
x; 512 πλ.

$$x = 2^9/2^3 = 2^6 = 2^\lambda \text{ σύνολα} \Rightarrow \lambda = 6 \text{ bits.}$$

Έχουμε ότι $v - \lambda = 9$ bits = **3 οκταδικά ψηφία**, το $\lambda = 6$ bits = **2 οκταδικά ψηφία** και το $\mu = 3$ bits = **1 οκταδικό ψηφίο**.

$$117165 = [117 \ 16] 5 \rightarrow \Sigma. 16$$

$$117167 = [117 \ 16] 7 \rightarrow \Sigma. 16$$

$$445231 = [445 \ 23] 1 \rightarrow \Sigma. 23$$

$$575232 = [575 \ 23] 2 \rightarrow \Sigma. 23$$

$$675253 = [675 \ 25] 3 \rightarrow \Sigma. 25$$

$$677335 = [677 \ 33] 5 \rightarrow \Sigma. 33$$

Μπορούν να είναι ταυτόχρονα όλες αυτές οι διευθύνσεις στην κρυφή μνήμη, αφού πρόκειται για διαφορετικά μπλοκ που μεταφέρονται σε διαφορετικά σύνολα. Αναφορικά με τις δύο πρώτες, θα πρέπει να αναφερθεί ότι πρόκειται για το ίδιο μπλοκ (με διεύθυνση 11716) που μεταφέρεται στο ίδιο σύνολο της κρυφής μνήμης (Σ. 16) και μάλιστα στο πλαίσιο Π0 του συνόλου αυτού, ενώ για τις δύο επόμενες θα πρέπει να αναφερθεί ότι πρόκειται για διαφορετικά μπλοκ (με διεύθυνσεις 44523 και 57523) που μεταφέρονται στο ίδιο σύνολο της κρυφής μνήμης (Σ. 23), επειδή όμως κάθε σύνολο αποτελείται από **8 πλαίσια**, μπορούν να είναι ταυτόχρονα στην κρυφή μνήμη. Επίσης, όσον αφορά τις δύο τελευταίες διευθύνσεις, επειδή αυτές πάνε σε διαφορετικά σύνολα, μπορούν επίσης να είναι ταυτόχρονα στην κρυφή μνήμη, τόσο μεταξύ τους όσο και με τις υπόλοιπες διευθύνσεις.

Αρκούσε ως απάντηση και η εξής: Επειδή υπάρχουν μόνο 6 διευθύνσεις μπλοκ, και από την άλλη επειδή κάθε σύνολο έχει 8 πλαίσια, όλες αυτές οι διευθύνσεις μπορούν να είναι ταυτόχρονα στην κρυφή μνήμη, ακόμη και αν όλες πήγαιναν στο ίδιο σύνολο.

Ασκηση από Φροντιστήριο με κρυφή μνήμη

Σε επεξεργαστή με αρτηρία διευθύνσεων των 28 δυαδικών ψηφίων υπάρχει κρυφή μνήμη των 16 Kbytes, με 8 bytes ανά πλαίσιο και οργάνωση **μονοσήμαντης απεικόνισης**. Θεωρήστε ότι το σύστημα της μνήμης έχει οργάνωση ενός byte ανά θέση μνήμης, δηλαδή σε κάθε byte αντιστοιχεί μια διεύθυνση, και η προσπέλαση γίνεται ανά byte. Ποια από τα περιεχόμενα των διευθύνσεων Α, Β, Γ και Δ μπορούν να βρίσκονται **ταυτόχρονα** στην κρυφή μνήμη;

$$A = AAABFF1$$

$$B = AAABFF5$$

$$\Gamma = FAABFF6$$

$$\Delta = FAA8FFF$$

Λύση

Μονοσήμαντη απεικόνιση. Η ψηφιολέξη είναι ταυτόχρονα και λέξη (θ.μ.), λόγω της οργάνωσης που δίνεται.



2^μ = αριθμός λέξεων (θ.μ.) ανά μπλοκ/πλαίσιο = 8 ψηφιολέξεις = 8 λέξεις (θ.μ.) $\Rightarrow \mu = 3$ bits.

Για **υπολογισμό πλαισίων** έχουμε: 1 πλ. 8 ψηφιολέξεις = 8 θ.μ. (λέξεις) = 8 bytes

$$x \pi\lambda; \quad 16 \text{ KB}$$

$x = 2^{14}/2^3 = 2^{11} \pi\lambda. = 2^{11} \text{ bits.}$ Οι διευθύνσεις που δίνονται είναι εκφρασμένες στο δεκαεξαδικό σύστημα. Αν το επιτρέπει το μέγεθος των επιμέρους πεδίων της διεύθυνσης, μετατρέπουμε τα πεδία αυτά σε πολλαπλάσια του «4», αν όχι, αναλύουμε τις διευθύνσεις σε μορφή δυαδική.

v-κ	κ	μ
A = AAABFF1 = 1010 1010 101010	11 1111 1111 0	001
B = AAABFF5 = 1010 1010 101010	11 1111 1111 0	101
Γ = FAABFF6 = 1111 1010 101011	01 1111 1111 0	110
Δ = FAA8FFF = 1111 1010 101010	00 1111 1111 1	111



Οι διευθύνσεις Α, Β μπορούν και πρέπει να είναι ταυτόχρονα στην κρυφή μνήμη καθώς αναφέρονται σε ίδια μπλοκ (ίδιο ν) που μεταφέρονται στο ίδιο πλαίσιο (ίδιο κ). Αυτό σημαίνει ότι όταν βρίσκονται στην κρυφή μνήμη τα περιεχόμενα της μιας διεύθυνσης πρέπει να βρίσκονται και αυτά της άλλης διεύθυνσης. Οι διευθύνσεις Γ, Δ μπορούν να είναι ταυτόχρονα στην κρυφή μνήμη, διότι μεταφέρονται σε διαφορετικά πλαίσια της κρυφής μνήμης. Συμπερασματικά, όλες οι διευθύνσεις Α, Β, Γ, Δ μπορούν να είναι ταυτόχρονα στην κρυφή μνήμη.

Άσκηση από Φροντιστήριο με κρυφή μνήμη

Θεωρήστε τρεις κρυφές μνήμες των 512 πλαισίων με 8 λέξεις ανά πλαίσιο και οργανώσεις αντίστοιχα:

i. Μονοσήμαντης απεικόνισης

ii. Πλήρους συσχέτισης

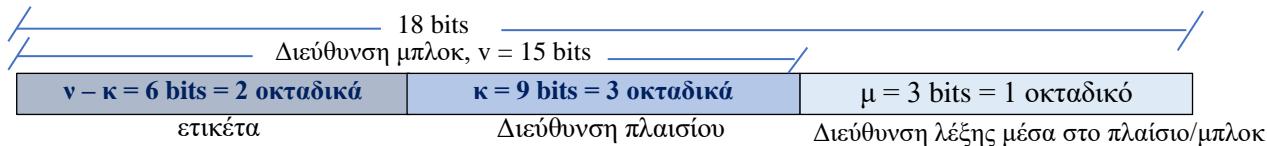
iii. 8-τρόπων συνόλου συσχέτισης.

α. Να δώσετε στο οκταδικό τις διεύθυνσεις των πλαισίων της κρυφής μνήμης στα οποία μπορούν να αποθηκευτούν τα περιεχόμενα των θέσεων της κύριας μνήμης με διεύθυνσεις 117165(8), 117167(8), 445231(8), 575232(8), 675253(8) και 677335(8) σε κάθε μία από τις περιπτώσεις i. ii. iii.

β. Ποια από τα περιεχόμενα των ανωτέρω θέσεων μνήμης είναι δυνατόν να βρίσκονται **ταυτόχρονα** στην κρυφή μνήμη σε κάθε μία από τις περιπτώσεις i. ii. iii και γιατί

Λύση

i. Μονοσήμαντη απεικόνιση. Η ψηφιολέξη είναι ταυτόχρονα και λέξη (θ.μ.), λόγω της οργάνωσης που δίνεται. Από τον αριθμό των πλαισίων υπολογίζουμε το κ, διότι ισχύει ότι $2^k = 512 \Rightarrow k = 9$ bits. Από τον αριθμό των λέξεων ανά πλαίσιο υπολογίζουμε το μ, διότι ισχύει ότι $2^\mu = 8 \Rightarrow \mu = 3$ bits.



$$2^\mu = \text{αριθμός λέξεων ανά πλαίσιο/μπλοκ} = 8 \text{ ψηφιολέξεις} = 8 \text{ λέξεις (θ.μ.)} \Rightarrow \mu = 3 \text{ bits.}$$

Οι διευθύνσεις που δίνονται είναι εκφρασμένες στο οκταδικό σύστημα. Αν το επιτρέπει το μέγεθος των επιμέρους πεδίων της διεύθυνσης, μετατρέπουμε τα πεδία αυτά σε πολλαπλάσια του «3», αν όχι, αναλόνουμε τις διεύθυνσεις σε μορφή δυαδική. Στην προκειμένη περίπτωση τα μεγέθη όλων των επιμέρους πεδίων είναι πολλαπλάσια του 3 και αυτό μπορούμε να το αξιοποιήσουμε στην ανάλυση των επιμέρους πεδίων, όπως φαίνεται παρακάτω:

Διεύθυνση	$n - k$	k	μ
117165(8),	11	716	5
117167(8),	11	716	7
445231(8),	44	523	1
575232(8),	57	523	2
675253(8)	67	525	3
677335(8)	67	733	5

Οι δύο πρώτες διεύθυνσεις **μπορούν** και πρέπει να είναι ταυτόχρονα στην κρυφή μνήμη καθώς αναφέρονται σε ίδια μπλοκ (ίδιο ν) που μεταφέρονται στο ίδιο πλαίσιο (ίδιο κ). Αυτό σημαίνει ότι όταν βρίσκονται στην κρυφή μνήμη τα περιεχόμενα της μιας διεύθυνσης πρέπει να βρίσκονται και αυτά της άλλης διεύθυνσης. Οι δύο επόμενες διεύθυνσεις (3^η και 4^η) δεν μπορούν μνήμη καθώς αναφέρονται σε διαφορετικά μπλοκ (διαφορετικό ν – κ) που μεταφέρονται στο ίδιο πλαίσιο (ίδιο κ). Αυτό σημαίνει ότι όταν βρίσκονται στην κρυφή μνήμη τα περιεχόμενα της μιας διεύθυνσης, π.χ. της 3^{ης} ή της 4^{ης}, δεν μπορούν να βρίσκονται στην κρυφή μνήμη και αυτά της άλλης διεύθυνσης. Τέλος αναφορικά με τις δύο τελευταίες διεύθυνσεις, μπορούν να είναι ταυτόχρονα στην κρυφή μνήμη, διότι μεταφέρονται σε διαφορετικά πλαίσια της κρυφής μνήμης, **Συμπερασματικά**, μπορούν να είναι ταυτόχρονα στην κρυφή μνήμη οι δύο πρώτες μαζί με τις δύο τελευταίες διεύθυνσεις και μαζί με μία εκ' των δύο ενδιάμεσων διεύθυνσεων.

ii. Πλήρους συσχέτισης. Σε αυτή την περίπτωση γεμίζουν **τα πλαίσια της κρυφής μνήμης γεμίζουν με τη σειρά από το πρώτο προς το τελευταίο**. Απλώς επειδή οι δύο πρώτες διεύθυνσεις αναφέρονται στο ίδιο μπλοκ, με διεύθυνση 11716,

μεταφέρονται στο ίδιο πλαίσιο της κρυφής μνήμης. Επειδή από τη μία έχουμε στη διάθεσή μας 512 πλαίσια και από την άλλη μόνο 6 διευθύνσεις που μεταφέρονται σε αυτά, σημαίνει ότι μπορούν όλες να είναι ταυτόχρονα στην κρυφή μνήμη. Από τα 512 πλαίσια της κρυφής μνήμης γεμίζουν μόνο τα 5 πλαίσια.

Διεύθυνση μπλοκ

$v = 15 \text{ bits} \wedge 5 \text{ οκταδικά ψηφία}$

Διεύθυνση λέξης μέσα στο πλαίσιο/μπλοκ

$\mu = 3 \text{ bits} \wedge 1 \text{ οκταδικό ψηφίο}$

$$117165 = \boxed{117 \ 16} \ 5 \rightarrow \text{Πλ. } \boxed{0}$$

$$117167 = \boxed{117 \ 16} \ 7 \rightarrow \text{Πλ. } \boxed{0}$$

$$445231 = \boxed{445 \ 23} \ 1 \rightarrow \text{Πλ. } \boxed{1}$$

$$575232 = \boxed{575 \ 23} \ 2 \rightarrow \text{Πλ. } \boxed{2}$$

$$675253 = \boxed{675 \ 25} \ 3 \rightarrow \text{Πλ. } \boxed{3}$$

$$677335 = \boxed{677 \ 33} \ 5 \rightarrow \text{Πλ. } \boxed{4}$$

iii 8-τρόπων συνόλου συσχέτισης

Διεύθυνση μπλοκ, $v = 15 \text{ bits}$

$v - \lambda = 9 \text{ bits} \wedge 3 \text{ οκταδικά ψηφία}$

ετικέτα

$\lambda = 6 \wedge 2 \text{ οκταδικά ψηφία}$

Διεύθυνση συνόλου

$\mu = 3 \text{ bits} \wedge 1 \text{ οκταδικό ψηφίο}$

Διεύθυνση λέξης μέσα στο πλαίσιο/μπλοκ

Από πριν έχουμε 512 πλαίσια, επομένως δοθέντος ότι έχουμε 8 – τρόπων συνόλου συσχέτισης μνήμη, ισχύει ότι:

$$1 \text{ σύνολο} \quad 8 \text{ πλ.}$$

$$x; \quad 512 \text{ πλ.}$$

$$x = 2^9 / 2^3 = 2^6 = 2^\lambda \text{ σύνολα} \Rightarrow \lambda = 6 \text{ bits.}$$

Έχουμε ότι $v - \lambda = 9 \text{ bits} = 3 \text{ οκταδικά ψηφία}$, το $\lambda = 6 \text{ bits} = 2 \text{ οκταδικά ψηφία}$ και το $\mu = 3 \text{ bits} = 1 \text{ οκταδικό ψηφίο.}$

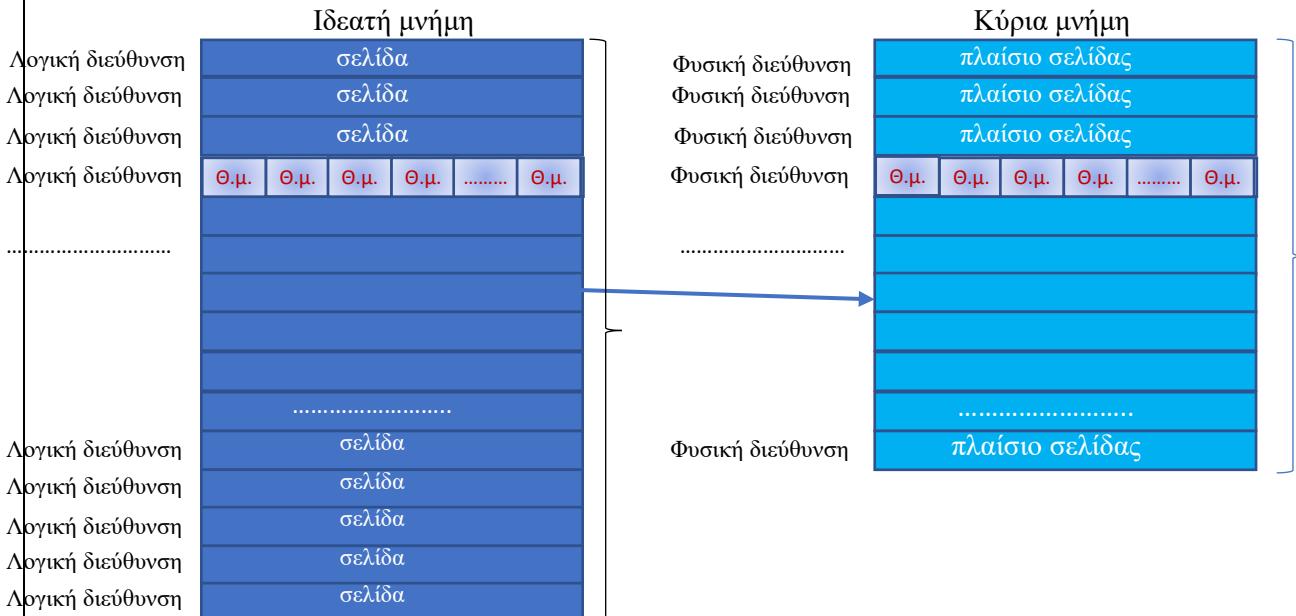
$$117165 = \boxed{117 \ 16} \ 5 \rightarrow \Sigma. \boxed{16}$$

Μπορούν να είναι ταυτόχρονα όλες αυτές οι διευθύνσεις στην κρυφή μνήμη, αφού πρόκειται για διαφορετικά μπλοκ που μεταφέρονται σε διαφορετικά σύνολα. Αναφορικά με τις δύο πρώτες, θα πρέπει να αναφερθεί ότι πρόκειται για το ίδιο μπλοκ (με διεύθυνση 11716) που μεταφέρεται στο ίδιο σύνολο της κρυφής μνήμης (Σ. 16) και μάλιστα στο πλαίσιο Π0 του συνόλου αυτού, ενώ για τις δύο επόμενες θα πρέπει να αναφερθεί ότι πρόκειται για διαφορετικά μπλοκ (με διευθύνσεις 44523 και 57523) που μεταφέρονται στο ίδιο σύνολο της κρυφής μνήμης (Σ. 23), επειδή όμως κάθε σύνολο αποτελείται από 8 πλαίσια, μπορούν να είναι ταυτόχρονα στην κρυφή μνήμη. Επίσης, όσον αφορά τις δύο τελευταίες διευθύνσεις, επειδή αυτές πάνε σε διαφορετικά σύνολα, μπορούν επίσης να είναι ταυτόχρονα στην κρυφή μνήμη, τόσο μεταξύ τους όσο και με τις υπόλοιπες διευθύνσεις.

Αρκούσε ως απάντηση και η εξής: Επειδή υπάρχουν μόνο 6 διευθύνσεις μπλοκ, και από την άλλη κάθε σύνολο έχει 8 πλαίσια, όλες αυτές οι διευθύνσεις μπορούν να είναι ταυτόχρονα στην κρυφή μνήμη, ακόμη και αν όλες πήγαιναν στο ίδιο σύνολο.

Κεφάλαιο 5.5 – Ιδεατή μνήμη

5.5.1 Λειτουργία ιδεατής μνήμης



Με τον όρο **ιδεατή μνήμη** εννοούμε την νοητή (και όχι τη φυσική) επέκταση της κύριας μνήμης (Κ.Μ.) έτσι ώστε ο επεξεργαστής να βλέπει **την κύρια μνήμη** και ένα μέρος του δίσκου ως μια μεγάλη, ενιαία και άμεσα προσπελάσιμη κύρια μνήμη. **Στην ιδεατή μνήμη υπάρχουν οι σελίδες**, όπου **κάθε σελίδα είναι μια συλλογή γειτονικών θέσεων μνήμης** και έχει μια λογική διεύθυνση μέσω της οποίας **προσπελαύνεται**. Αντίστοιχα στην Κύρια Μνήμη υπάρχουν τα πλαίσια σελίδας, όπου **κάθε πλαίσιο σελίδας είναι μια συλλογή γειτονικών θέσεων μνήμης** και έχει μια φυσική διεύθυνση μέσω της οποίας **προσπελαύνεται**. Μια σελίδα μεταφέρεται σε ένα πλαίσιο σελίδας. Ο χώρος της ιδεατής μνήμης, επειδή αποτελείται από σελίδες με τις λογικές τους διευθύνσεις, ονομάζεται και **χώρος λογικών διευθύνσεων** και αντίστοιχα ο χώρος της κύριας μνήμης, επειδή αποτελείται από πλαίσια σελίδων με τις φυσικές τους διευθύνσεις, ονομάζεται **και χώρος φυσικών διευθύνσεων** Ισχύουν οι ακόλουθες αντιστοιχίες:

Ιδεατή μνήμη

- Μέγεθος σελίδας = Μέγεθος πλαισίου σελίδας
- Σελίδα → Πλαίσιο σελίδας
- Πλήθος σελίδων >> Πλήθος πλαισίων σελίδας

Κρυφή μνήμη

- Μέγεθος πμλοκ = μέγεθος πλαισίου
- Μπλοκ → πλαίσιο
- Πλήθος μπλοκ >> Πλήθος πλαισίων

Κατά τη μεταφορά μιας σελίδας σε ένα πλαίσιο σελίδας, θα πρέπει να μεταφραστεί η λογική διεύθυνση σε φυσική. Για το λόγο αυτό, η λογική διεύθυνση εισέρχεται σε ένα πίνακα σελίδων όπου και μεταφράζεται σε φυσική διεύθυνση. Το μήκος της λογικής διεύθυνσης είναι συνήθως μεγαλύτερο από το μήκος της φυσικής διεύθυνσης. Η **λογική διεύθυνση** αποτελείται από δύο μέρη, όπως φαίνεται στη συνέχεια. Το συνολικό μήκος της λογικής διεύθυνσης δίνεται πάντα από την εκφώνηση.

ΑΛΣ	δμΣ
Αριθμός λογικής σελίδας	Διεύθυνση μέσα στη σελίδα

Η **φυσική διεύθυνση** αποτελείται από δύο μέρη, όπως φαίνεται στη συνέχεια. Το συνολικό μήκος της φυσικής διεύθυνσης άλλοτε δίνεται πάντα από την εκφώνηση και άλλοτε προκύπτει από το μέγεθος της Κ.Μ.

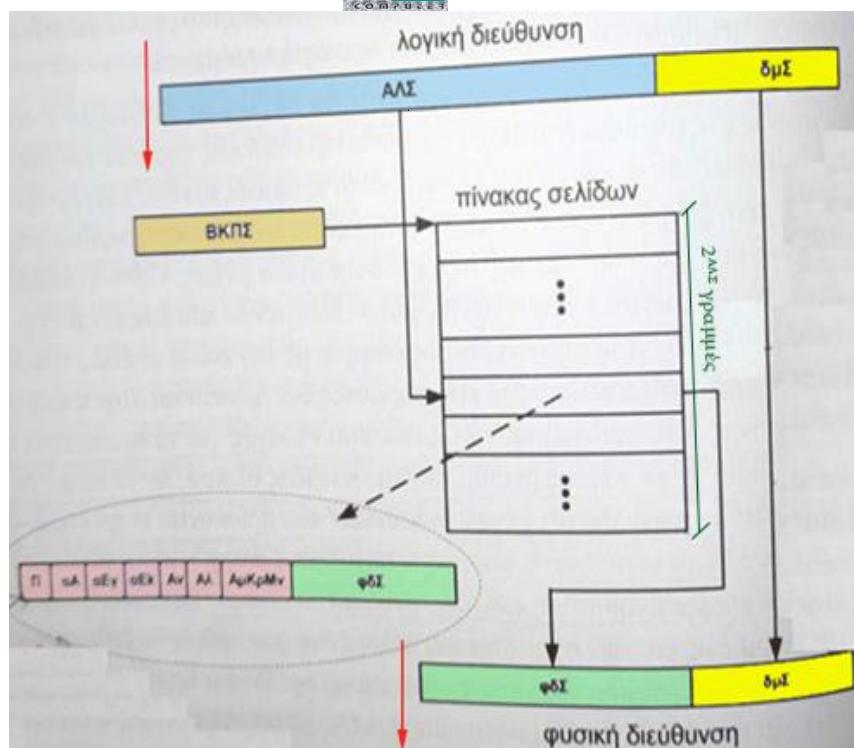
φδΣ	δμΣ
Φυσική διεύθυνση σελίδας	Διεύθυνση μέσα στη σελίδα

Παρατηρούμε ότι το **κοινό πεδίο** και στις δύο διευθύνσεις είναι το πεδίο δμΣ, το οποίο προκύπτει πάντα από το μέγεθος σελίδας = μέγεθος πλαισίου σελίδας και δίνεται από την εκφώνηση.

Τεχνικές υλοποίησης ιδεατής μνήμης

Σελιδοποίηση Τμηματοποίηση Σελιδοποιημένη τμηματοποίηση

Στις ασκήσεις αυτού του Κεφαλαίου χρησιμοποιείται η τεχνική της **σελιδοποίησης**, ενώ οι άλλες δύο τεχνικές υλοποίησης της ιδεατής μνήμης χρησιμοποιούνται κυρίως σε θεωρητική βάση.



Στην τεχνική αυτή εισέρχεται στο πάνω μέρος η λογική διεύθυνση. Το πεδίο ΑΛΣ της λογικής διεύθυνσης εισέρχεται σε έναν πίνακα σελίδων (Π.Σ.), ο οποίος περιέχει γραμμές και κάθε γραμμή του αποτελείται από 7 bits ελέγχου καθώς και το πεδίο φδΣ. Με βάση το συγκεκριμένο πεδίο ΑΛΣ που εισάγεται, εντοπίζεται μια συγκεκριμένη γραμμή του Π.Σ. από την οποία εξάγεται το αντίστοιχο φδΣ, το οποίο ενώνεται στο κάτω μέρος με το πεδίο δμΣ (το οποίο νωρίτερα έχει μεταφερθεί αυτούσιο από το λογική διεύθυνση) και παράγεται στην έξοδο η φυσική διεύθυνση. Ο καταχωρητής ΒΚΠΣ = Βασικός Καταχωρητής Πίνακα Σελίδων δείχνει τη διεύθυνση έναρξης της αποθήκευσης του Π.Σ. στην κύρια μνήμη.

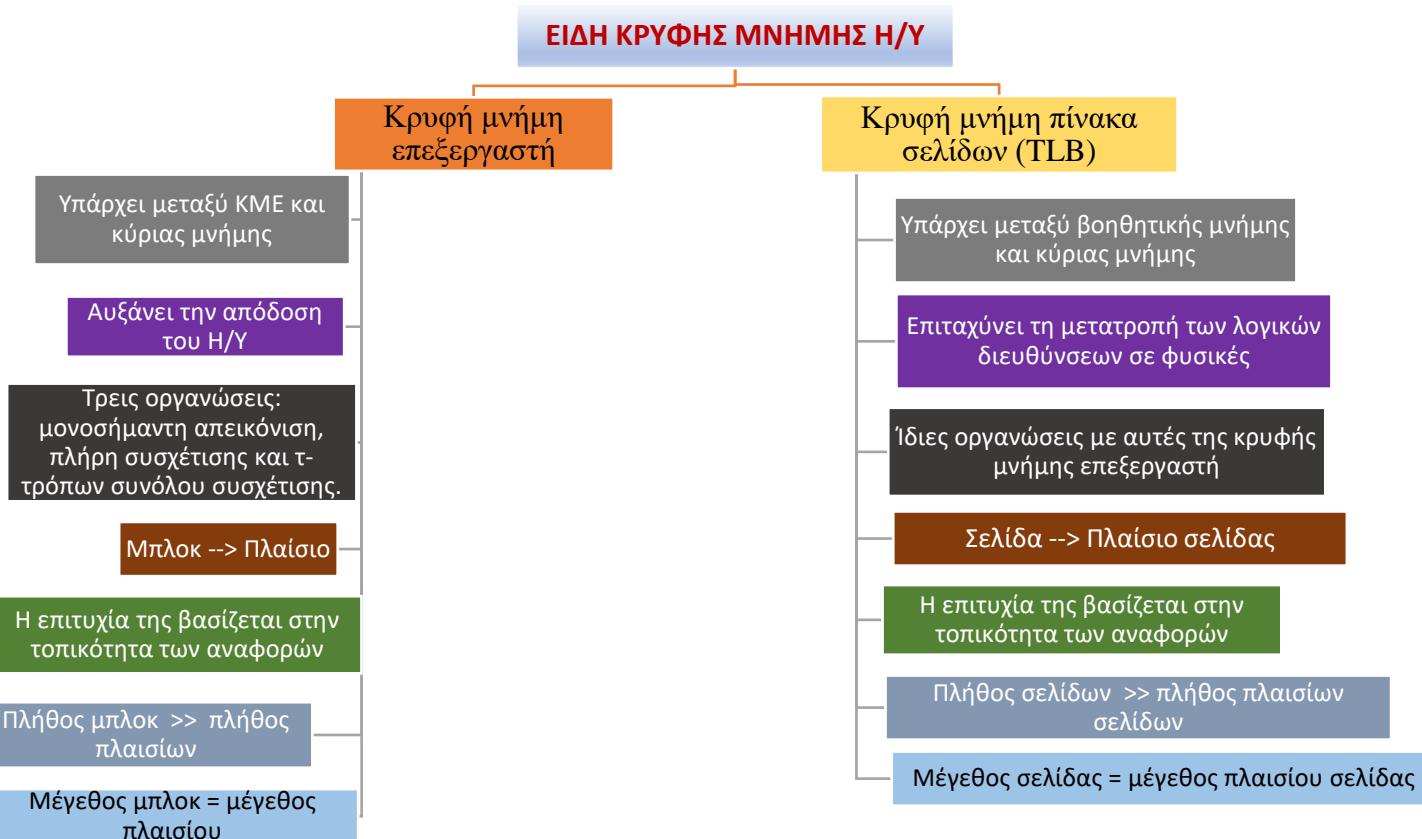
$$\text{Μέγεθος Π.Σ.} = \text{πλήθος γραμμών Π.Σ.} \times \text{μέγεθος κάθε γραμμής} = 2^{\text{ΑΛΣ}} \text{γρ.} \times (7 + \text{φδΣ}) \text{ bits/γρ.}$$

Η συνήθης **τεχνική της σελιδοποίησης** έχει το μειονέκτημα ότι οδηγεί σε έναν τεράστιο πίνακα σελίδων, αφού για κάθε σελίδα του χώρου των λογικών διευθύνσεων υπάρχει και μια γραμμή στον πίνακα σελίδων. Αυτό συνεπάγεται ότι ο Π.Σ. δεν χωρά να αποθηκευτεί μέσα στην Κ.Μ. αφού ξεπερνά τη χωρητικότητά της. Για το λόγο αυτό χρησιμοποιούνται **τεχνικές μείωσης της χωρητικότητας του πίνακα σελίδων (Π.Σ.)**, όπως είναι η **χρήση επιπέδων** και η **χρήση του αντιστραμμένου πίνακα σελίδων (Α.Π.Σ.)**, που αναφέρονται στη συνέχεια.

Στην τεχνική των επιπέδων υπό τη μορφή δέντρου, υπάρχουν δύο περιπτώσεις: το δέντρο δεν χρειάζεται να είναι πλήρες και στη δεύτερη περίπτωση υποπίνακες του Π.Σ. αποθηκεύονται στην ιδεατή μνήμη. Στην τεχνική αυτή μπορούν να χρησιμοποιηθούν από δύο επίπεδα και πάνω και τότε το πεδίο ΑΛΣ υποδιαιρείται σε τόσα τμήματα όσα είναι ο αριθμός των επιπέδων. Κάθε επιμέρους επίπεδο -εξαιρουμένου του τελευταίου- περιέχει σε κάθε γραμμή του μόνο ένα από τα 7 bits ελέγχου (για μείωση του μεγέθους των γραμμών) και το πεδίο φδΠΣ 2^{2^0} επιπέδου, 3⁰ επιπέδου κ.λ.π. Επιπλέον υπάρχει και πάλι ο ΒΚΠΣ, που δείχνει τη διεύθυνση έναρξης του Π.Σ του 1⁰ επιπέδου στην Κ.Μ. Για να βρούμε το συνολικό μέγεθος του Π.Σ., θα πρέπει να αθροίσουμε όλα τα επιμέρους μεγέθη των Π.Σ. που υπάρχουν στα επιμέρους επίπεδα. Το πεδίο φδΣ εξάγεται πάντα από τον Π.Σ. του **τελευταίου επιπέδου** και ενώνεται με το πεδίο δμΣ που μεταφέρεται αυτούσιο από τη λογική διεύθυνση) για να παραχθεί στην έξοδο η φυσική διεύθυνση.

5.5.2 Σύγκριση κρυφής μνήμης επεξεργαστή και κρυφής μνήμης Πίνακα Σελίδων

Η προσπέλαση του πίνακα μετάφρασης λογικών διευθύνσεων σε φυσικές διευθύνσεις, οδηγεί σε διπλασιασμό του χρόνου που απαιτείται για να διαβαστεί η ζητούμενη πληροφορία (είτε πρόκειται για εντολή είτε για δεδομένο) από την κύρια μνήμη. Αυτή η καθυστέρηση μεγαλώνει ακόμη περισσότερο όταν ο πίνακας μετάφρασης λογικών διευθύνσεων σε φυσικές διευθύνσεις υλοποιείται με τη μορφή δένδρου με k επίπεδα καθώς σε μια τέτοια περίπτωση μπορεί να απαιτηθούν μέχρι και k προσπελάσεις της κύριας μνήμης. Για αποφυγή της συγκεκριμένης καθυστέρησης που μειώνει την απόδοση της ιδεατής μνήμης, εκμεταλλευόμαστε την τοπικότητα των αναφορών που παρουσιάζουν τα προγράμματα και τα δεδομένα και καταφεύγουμε στην ιδέα της κρυφής μνήμης, όπου χρησιμοποιούμε μια κρυφή μνήμη ειδικού σκοπού που ονομάζεται κρυφή μνήμη πίνακα σελίδων (TLB). Σε αυτή αποθηκεύουμε τις πιο πρόσφατες αντιστοιχίσεις λογικών διευθύνσεων σε φυσικές. Οι μικρότερες μνήμες αυτού του είδους έχουν οργάνωση πλήρους συσχέτισης και οι μεγαλύτερες έχουν οργάνωση τ – τρόπων συνόλου συσχέτισης.



5.5.3 Τεχνική σελιδοποίησης

Περιγράψτε την τεχνική της σελιδοποίησης

Λύση

Για να υλοποιήσουμε την ιδεατή μνήμη με την τεχνική της σελιδοποίησης διαιρούμε το χώρο λογικών διευθύνσεων σε ομάδες διαδοχικών διευθύνσεων ίσου μεγέθους που καλούμε **σελίδες (pages)** και το χώρο φυσικών διευθύνσεων, δηλαδή την κύρια μνήμη, σε ομάδες διαδοχικών διευθύνσεων ίσου μεγέθους που καλούμε **πλαίσια σελίδων (page frames)**. Το μέγεθος μίας σελίδας ισούται με το μέγεθος ενός πλαισίου σελίδας και για λόγους εύκολης υλοποίησης το πλήθος των διευθύνσεων σε μία σελίδα ή πλαισίο σελίδας είναι μια δύναμη του δύο, έστω κ. Επομένως κάθε λογική διεύθυνση καθώς και κάθε φυσική διεύθυνση θεωρούμε ότι αποτελείται από δύο τμήματα. Τα κ λιγότερο σημαντικά δυαδικά ψηφία της λογικής καθώς και της φυσικής διεύθυνσης δμΣ, δίνουν τη διεύθυνση μέσα σε μία σελίδα ή πλαισίο σελίδας αντίστοιχα. Τα υπόλοιπα δυαδικά ψηφία της λογικής διεύθυνσης δίνουν το αριθμό της λογικής σελίδας ΑΛΣ. Η μετάφραση μίας λογικής διεύθυνσης σε φυσική διεύθυνση γίνεται με τη βοήθεια ενός πίνακα που καλείται **πίνακας σελίδων (page table)**.

5.5.4 Τρόποι (τεχνικές) υλοποίησης του πίνακα σελίδων

Αναφέρετε τους τρόπους υλοποίησης του πίνακα σελίδων.

Λύση

Υπάρχουν τρεις τεχνικές για την κατασκευή του Πίνακα Σελίδων (Π.Σ.) που μετατρέπει λογική διεύθυνση → φυσική:

- Τεχνική σελιδοποίησης
- Τεχνική τμηματοποίησης
- Τεχνική σελιδοποιημένης τμηματοποίησης

5.5.5 Τεχνική του αντιστραμμένου πίνακα σελίδων (ΑΠΣ)

Ποια είναι τα πλεονεκτήματα της τεχνικής του **αντιστραμμένου** πίνακα σελίδων σε σχέση με την κλασική υλοποίηση με πίνακα σελίδων ενός επιπέδου.

Λύση

Η τεχνική του αντιστραμμένου πίνακα σελίδων είναι ένας τρόπος μείωσης του μεγέθους του πίνακα σελίδων. Κάτι τέτοιο αποτελεί και το κύριο πλεονέκτημα της τεχνικής. Συγκεκριμένα ο αντεστραμμένος πίνακας έχει μια θέση για κάθε πλαισίο σελίδας σε αντίθεση με την υλοποίηση σε ένα επίπεδο όπου ο πίνακας σελίδων διαθέτει μια γραμμή για κάθε σελίδα. Μια συνάρτηση διασποράς μετασχηματίζει το τμήμα της λογικής διεύθυνσης που δηλώνει τον ΑΛΣ και το αποτέλεσμα του μετασχηματισμού δηλώνει την θέση του αντεστραμμένου πίνακα σελίδων που θα προσπελαστεί. Κάθε θέση έχει πολλές αντιστοιχίσεις ΑΛΣ σε φδΣ και κάθε αντιστοιχίσης έχει 7 bit ελέγχου το πεδίο ΑΛΣ και το πεδίο φδΣ.

5.5.6 Σημασία κρυφής μνήμης πίνακα σελίδων (TLB)

Να αναφέρετε τι αποθηκεύουμε στην κρυφή μνήμη πίνακα σελίδων (TLB) και γιατί;

Λύση

Η προσπέλαση του πίνακα σελίδων για τη μετάφραση λογικών διευθύνσεων σε φυσικές οδηγεί σε αύξηση του χρόνου που απαιτείται για να διαβαστεί η ζητούμενη πληροφορία. Για να αποφύγουμε αυτή την καθυστέρηση εκμεταλλευόμαστε την τοπικότητα των αναφορών και χρησιμοποιούμε μια κρυφή μνήμη ειδικού σκοπού που την ονομάζουμε κρυφή μνήμη πίνακα σελίδων (TLB). Σε αυτήν αποθηκεύουμε τις πιο πρόσφατες αντιστοιχίσεις λογικών διευθύνσεων σε φυσικές. Επειδή επιθυμούσε η TLB να έχει το μεγαλύτερο δυνατό λόγο επιτυχίας επιλεγούμε να έχει οργάνωση πλήρους συσχέτισης η τα-τρόπων συνόλου συσχέτισης.

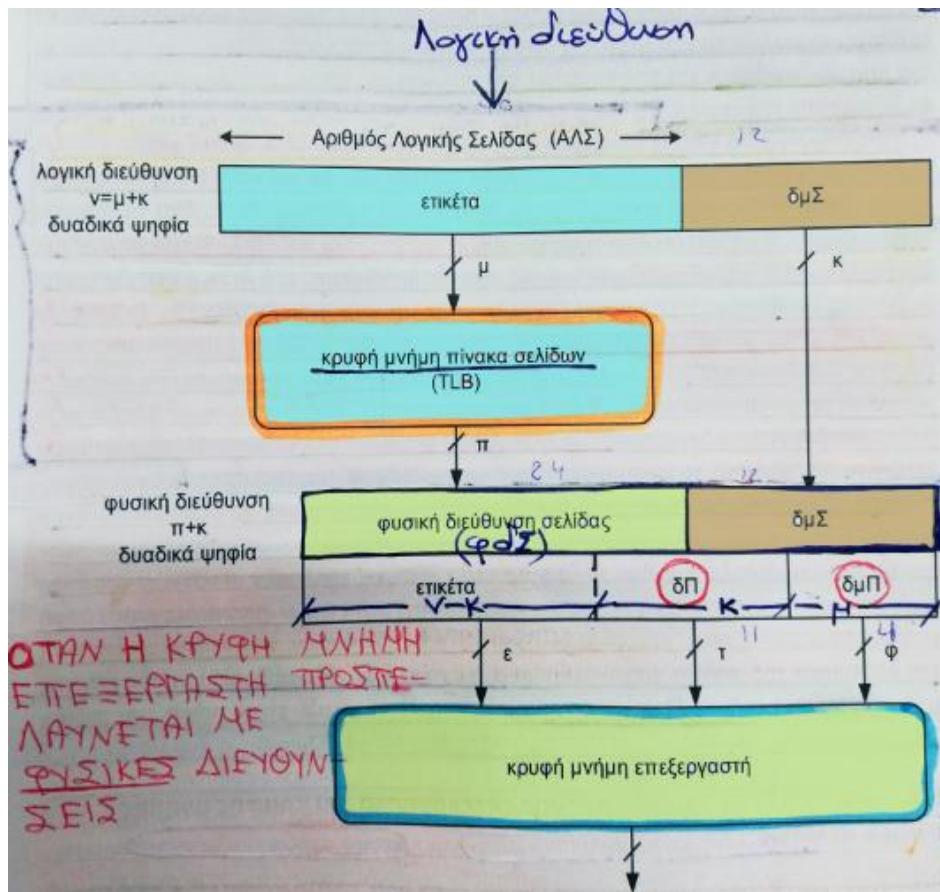
5.5.7 Τρόποι προσπέλασης κρυφής μνήμης επεξεργαστή όταν υπάρχει και η TLB

Υπάρχουν δύο τρόποι για να προσπελαστεί η κρυφή μνήμη επεξεργαστή, όταν υπάρχει και η μνήμη TLB:

- a) **Με φυσικές διευθύνσεις:** το μειονέκτημα στην περίπτωση αυτή είναι η καθυστέρηση που προκύπτει καθώς πρέπει πρώτα να προσπελαστεί η κρυφή μνήμη πίνακα σελίδων για να μεταφραστεί η λογική διεύθυνση σε φυσική και στη συνέχεια να χρησιμοποιηθεί αυτή για να προσπελαστεί η κρυφή μνήμη επεξεργαστή.
- b) **Με λογικές διευθύνσεις:** σε αυτή την περίπτωση έχουμε μείωση της προηγούμενης καθυστέρησης και μάλιστα πρέπει η προσπέλαση της κρυφής μνήμης πίνακα σελίδων να γίνεται παράλληλα με την προσπέλαση της κρυφής μνήμης επεξεργαστή. Το μειονέκτημα που υπάρχει στην περίπτωση αυτή είναι η ύπαρξη πολλαπλών αντιγράφων των ίδιων δεδομένων στην κρυφή μνήμη επεξεργαστή, τα οποία μπορεί να είναι και μη – ταυτοτικά (φαινόμενο aliasing).

5.5.8 Προσπέλασης κρυφής μνήμης επεξεργαστή (TLB) με φυσικές διευθύνσεις

Όταν η κρυφή μνήμη του επεξεργαστή προσπελαύνεται με φυσικές διευθύνσεις, τότε πρώτα θα πρέπει να προσπελαστεί η κρυφή μνήμη πίνακα σελίδων ώστε να μεταφραστεί η λογική διεύθυνση σε φυσική και στη συνέχεια η κρυφή μνήμη του επεξεργαστή. Εάν η αντιστοίχιση δεν βρίσκεται στην κρυφή μνήμη πίνακα σελίδων θα πρέπει να προσπελαστεί ο πίνακας σελίδων που βρίσκεται στην κύρια μνήμη. Σε περίπτωση αποτυχίας κατά την προσπέλαση της κρυφής μνήμης του επεξεργαστή θα προσπελαστεί και πάλι η κύρια μνήμη. Εφόσον απαιτείται πρώτα η προσπέλαση της κρυφής μνήμης πίνακα σελίδων, συνεπάγεται ότι ο χρόνος ανάγνωσης εντολής ή δεδομένων από την κρυφή μνήμη του επεξεργαστή θα είναι στην καλύτερη περίπτωση περίπου διπλάσιος του χρόνου προσπέλασης της κρυφής μνήμης του επεξεργαστή. Για να μειώσουμε τις συνέπειες θα πρέπει η προσπέλαση της κρυφής μνήμης πίνακα σελίδων να γίνεται σε διαφορετική βαθμίδα του μηχανισμού μερικώς επικαλυπτόμενων λειτουργιών (pipeline).



5.5.9 Προσπέλασης κρυφής μνήμης επεξεργαστή (TLB) με λογικές διευθύνσεις

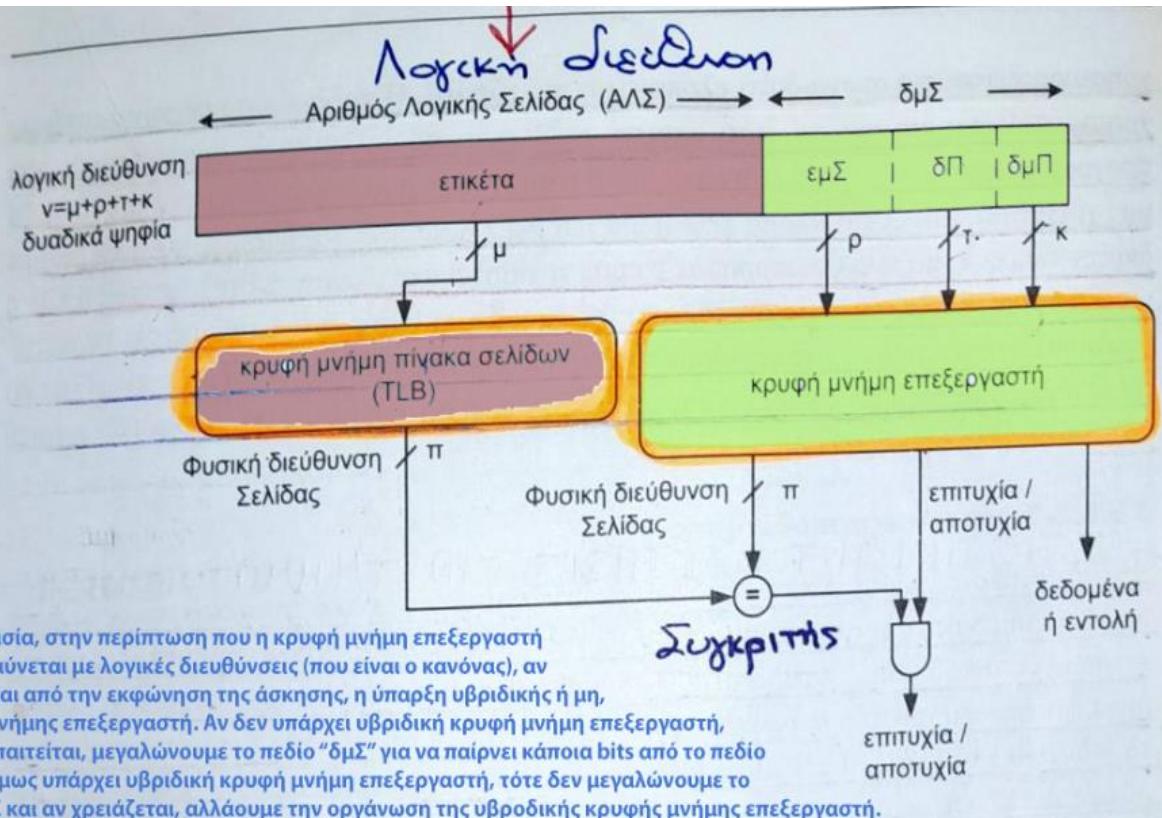
Περιγράψτε τον τρόπο προσπέλασης της κρυφής μνήμης επεξεργαστή χρησιμοποιώντας λογικές διευθύνσεις.

Λύση

Όταν η κρυφή μνήμη του επεξεργαστή προσπελαύνεται με λογικές διευθύνσεις τότε εάν έχουμε επιτυχία διαβάζονται τα απαιτούμενα δεδομένα ή η εντολή από την κρυφή μνήμη επεξεργαστή. Για λόγους μείωσης της καθυστέρησης η προσπέλαση της κρυφής μνήμης πίνακα σελίδων γίνεται **παράλληλα** με την προσπέλαση της κρυφής μνήμης επεξεργαστή. Όμως μόνο στην περίπτωση αποτυχίας της κρυφής μνήμης επεξεργαστή, χρησιμοποιείται το αποτέλεσμα της προσπέλασης της κρυφής μνήμης πίνακα σελίδων για να πάρουμε τη φυσική διεύθυνση. Αν η πληροφορία αντιστοίχισης δεν υπάρχει στην κρυφή μνήμη πίνακα σελίδων, απαιτείται προσπέλαση του πίνακα σελίδων στην κύρια μνήμη για να πάρουμε την φυσική διεύθυνση και στη συνέχεια προσπέλαση της κύριας μνήμης για να πάρουμε τα δεδομένα ή την εντολή.

Παρατηρούμε ότι όταν η ζητούμενη πληροφορία βρίσκεται στην κρυφή μνήμη επεξεργαστή, τότε η προσπέλασή της είναι **σημαντικά πιο γρήγορη** από ότι στην περίπτωση που η κρυφή μνήμη επεξεργαστή προσπελαύνεται με φυσικές διευθύνσεις. Θα πρέπει να σημειώσουμε σε αυτό το σημείο ότι όταν η κρυφή μνήμη επεξεργαστή προσπελαύνεται με φυσικές διευθύνσεις, τότε η ετικέτα είναι τμήμα της φυσικής διεύθυνσης, ενώ όταν προσπελαύνεται με λογικές διευθύνσεις, τότε η ετικέτα είναι ένα τμήμα

της λογικής διεύθυνσης. Το πρόβλημα με την κρυφή μνήμη επεξεργαστή που προσπελαύνεται με λογικές διευθύνσεις, είναι ότι όταν δεδομένα πρέπει να μοιραστούν μεταξύ διάφορων διαδικασιών, τα ίδια δεδομένα μπορεί να έχουν διαφορετικές λογικές διευθύνσεις στους χώρους διευθύνσεων των διαδικασιών, οδηγώντας στην ύπαρξη πολλών αντιγράφων των ίδιων δεδομένων στην κρυφή μνήμη επεξεργαστή (aliasing) τα οποία μπορεί να είναι και ασυνεπή (μη ταυτοτικά). Είναι δυνατόν να χρησιμοποιήσουμε μία υβριδική, όσον αφορά τις διευθύνσεις, κρυφή μνήμη για να πετύχουμε την ταχύτητα της κρυφής μνήμης επεξεργαστή που προσπελαύνεται με λογικές διευθύνσεις αλλά και να μην περιέχει πολλαπλά αντίγραφα της ίδιας πληροφορίας. Αυτό επιτυγχάνεται με το να εξαρτάται η θέση της πληροφορίας στην κρυφή μνήμη επεξεργαστή μόνο από το τμήμα της λογικής διεύθυνσης που καθορίζει τη θέση της λέξης μέσα στην σελίδα.



5.5.10 Φαινόμενο thrashing

Περιγράψτε το φαινόμενο thrashing .

Λύση

Αυτό έχει ως συνέπεια το πρόγραμμα να διακόπτεται συνεχώς για να μεταφερθεί τμήμα του ή δεδομένα μεταξύ της κύριας μνήμης και της μονάδας δίσκου. Αν τα τμήματα όλων των προγραμμάτων και των σχετιζόμενων δεδομένων που βρίσκονται στην κύρια μνήμη είναι πολύ μικρά, τότε το υπολογιστικό σύστημα ασχολείται συνεχώς με τη μεταφορά πληροφορίας μεταξύ κύριας μνήμης και μονάδας δίσκου (thrashing).

5.5.11 Σημασία αντιστραμμένου πίνακα σελίδων (ΑΠΣ)

Να αναφέρετε το λόγο για τον οποίο χρησιμοποιείται ο αντιστραμμένος πίνακας σελίδων

Λύση

Ένας τρόπος να μειώσουμε το μέγεθος του πίνακα σελίδων ώστε να διατηρείται ολόκληρος στην κύρια μνήμη και να μην καταλαμβάνει απαγορευτικά μεγάλο τμήμα της, είναι να χρησιμοποιήσουμε την τεχνική του αντιστραμμένου πίνακα σελίδων, ο οποίος έχει μια θέση για κάθε πλαίσιο σελίδας της κύριας μνήμης, δηλαδή του χώρου φυσικών διευθύνσεων και όχι μια θέση για κάθε σελίδα του χώρου λογικών διευθύνσεων όπως ο πίνακας σελίδων. Το γεγονός αυτό οδηγεί, όπως αναφέρθηκε, στη μείωση της χωρητικότητάς του.

5.5.12. Σημασία κρυφής μνήμης πίνακα σελίδων (TLB)

Να αναφέρετε τι αποθηκεύουμε στην κρυφή μνήμη πίνακα σελίδων (TLB) τον λόγο για τον οποίο χρησιμοποιείται και που οφείλεται η επιτυχία του.

Λύση

Όταν ο πίνακας μετάφρασης λογικών διευθύνσεων σε φυσικές διευθύνσεις υλοποιείται υπό την μορφή δέντρου με κ επίπεδα, οπότε για να γίνει η μετάφραση της λογικής διεύθυνσης σε φυσική μπορεί να απαιτηθούν μέχρι και η προσπελάσεις της κύριας μνήμης. Για να αποφύγουμε αυτή την καθυστέρηση, η οποία μειώνει την απόδοση της ιδεατής μνήμης εκμεταλλευόμαστε και πάλι την τοπικότητα των αναφορών που παρουσιάζουν τα προγράμματα και τα δεδομένα και καταφεύγουμε στην χρησιμοποίηση της ιδέας της κρυφής μνήμης. Χρησιμοποιούμε μια μικρή κρυφή μνήμη ειδικού σκοπού, που την ονομάζουμε **κρυφή μνήμη πίνακα σελίδων** και σε αυτή αποθηκεύουμε τις πιο πρόσφατες αντιστοιχίες λογικών διευθύνσεων σε φυσικές διευθύνσεις. Το μέγεθος της κρυφής μνήμης πίνακα σελίδων κυμαίνεται από μερικές δεκάδες μέχρι μερικές χιλιάδες θέσεις. Επειδή επιθυμούμε να έχει το μεγαλύτερο δυνατό λόγο επιτυχίας οι μικρότερες έχουν οργάνωση πλήρους συσχέτισης ενώ οι μεγαλύτερες τ-τρόπων συνόλου συσχέτισης.

5.5.13. Επιτυχία ιδεατής μνήμης

Να αναφέρετε που βασίζεται η επιτυχία της ιδεατής μνήμης και να σχολιάσετε το πολύ με 5 γραμμές ποιος είναι ο στόχος της.

Λύση

Η επιτυχία της ιδεατής μνήμης βασίζεται στην τοπικότητα των αναφορών. Η ιδεατή μνήμη αναπτύχθηκε για να αυτοματοποιήσει τη μετακίνηση προγραμμάτων και δεδομένων μεταξύ της κύριας μνήμης και της βοηθητικής μνήμης και να δώσει στον προγραμματιστή εφαρμογών την ψευδαίσθηση μιας μεγάλης, ενιαίας και άμεσα προσπελάσιμης μνήμης.

5.5.14. Θέμα σχετικά με μειονεκτήματα σελιδοποίησης

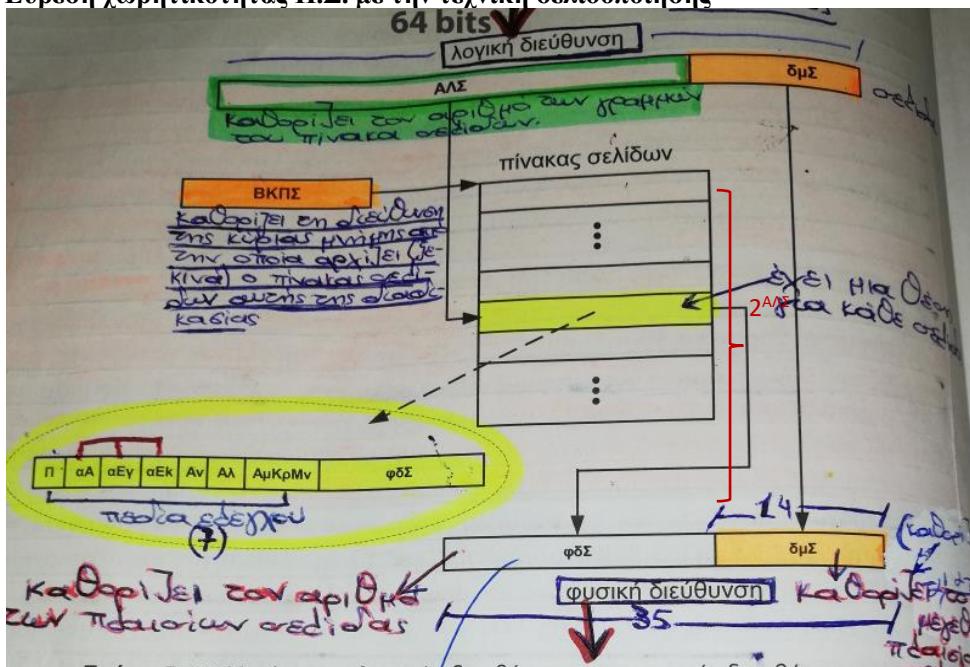
Να αναφέρετε τα κύρια μειονεκτήματα του βασικού τρόπου υλοποίησης της τεχνικής της **σελιδοποίησης** που χρησιμοποιείται για την υλοποίηση της ιδεατής μνήμης και τρόπους υλοποίησης που έχουν προταθεί για να ξεπεραστεί το μειονέκτημα αυτό.

Λύση

Σύμφωνα με το βασικό τρόπο υλοποίησης του πίνακα σελίδων, **για κάθε σελίδα του χώρου λογικών διευθύνσεων πρέπει να έχουμε και μια γραμμή μέσα στον Π.Σ. Οι διαδικασίες δεν χρησιμοποιούν συνήθως ολόκληρο τον χώρο λογικών διευθύνσεων, επομένως πολλές από τις γραμμές του πίνακα σελίδων δεν χρησιμοποιούνται**. Μια λογική σελίδα που δεν έχει κατανεύμηθεί σε μια διαδικασία δεν καταλαμβάνει χώρο ούτε στον μαγνητικό δίσκο. **Παρόλα αυτά ο βασικός τρόπος υλοποίησης του πίνακα σελίδων απαιτεί ο πίνακας σελίδων της διαδικασίας που εκτελείται, να βρίσκεται στην κύρια μνήμη και διαθέτει μια γραμμή για κάθε σελίδα του χώρου λογικών διευθύνσεων, γεγονός που οδηγεί σε σπατάλη μεγάλου χώρου της κύριας μνήμης**. Για να λυθεί το πρόβλημα μπορούμε να υλοποιήσουμε τον πίνακα σελίδων:

1. Σε πολλά επίπεδα υπό την μορφή δέντρου. Το δέντρο δεν χρειάζεται να είναι πλήρες.
2. Σε πολλά επίπεδα υπό την μορφή δέντρου και υποπίνακες του πίνακα σελίδων να αποθηκεύονται στην ιδεατή μνήμη.
3. Με την τεχνική του αντιστραμμένου πίνακα σελίδων.

Παράδειγμα 5.10 – Εύρεση χωρητικότητας Π.Σ. με την τεχνική σελιδοποίησης



Δεδομένα:

- KM: $32 \text{ GB} = 2^5 \times 2^{30} \text{ bytes} = 2^{35} \text{ bytes} = (\lambda\text{όγω οργάνωσης 1 ψηφιολέξη/θ.μ.}) 2^{35} \text{ θ.μ.} \rightarrow \text{εύρεση μεγέθους φυσικής διεύθυνσης}$
- Μήκος λογικής διεύθυνσης = 64 bits (πάντα δίνεται)
- Μέγεθος σελίδων = μέγεθος πλαισίων σελίδων = 16 KB (από αυτό το μέγεθος υπολογίζεται πάντα το πεδίο δμΣ).

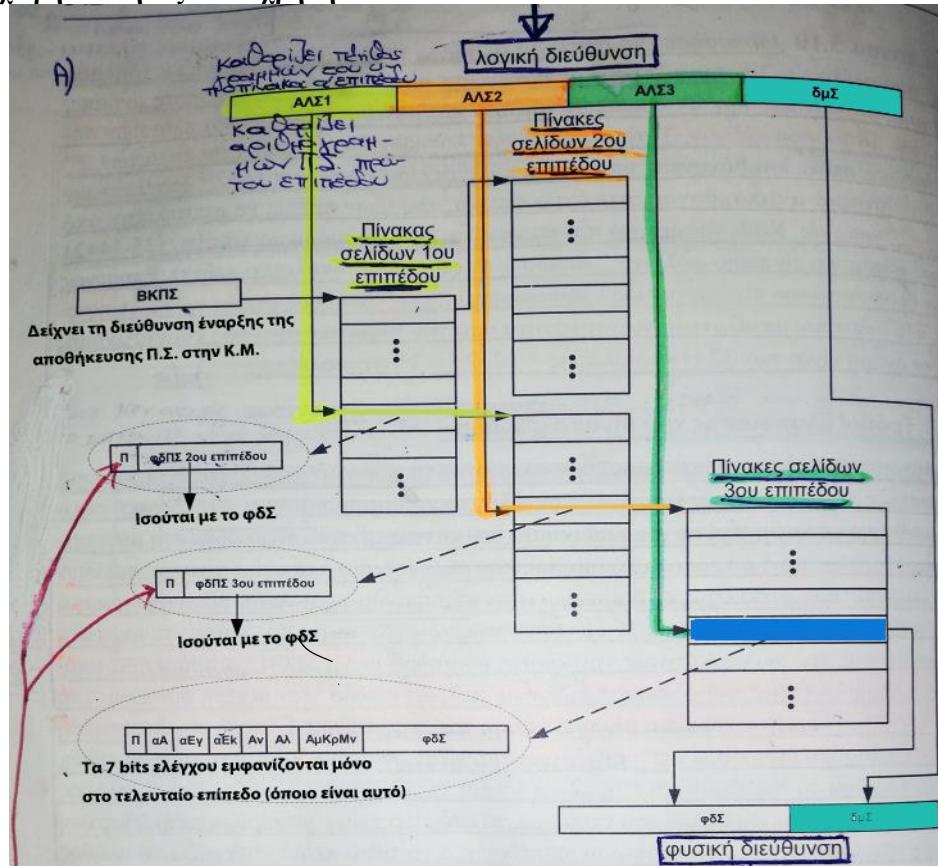
Ζητούμενο: εύρεση της χωρητικότητας του Π.Σ.

Λύση

Οι σελίδες βρίσκονται στην ιδεατή μνήμη και τα πλαισία σελίδων βρίσκονται στην κύρια μνήμη. Ισχύει ότι το πλήθος σελίδων > πλήθος πλαισίων σελίδων. Πάντα από το μέγεθος σελίδων = μέγεθος πλαισίων σελίδων υπολογίζουμε το πεδίο δμΣ (διεύθυνση μέσα στη Σελίδα). Δηλαδή $16 \text{ KB} = 2^4 \times 2^{10} \text{ bytes} = 2^{14} \text{ bytes} = (\lambda\text{όγω οργάνωσης 1 ψηφιολέξη/θ.μ.}) 2^{14} \text{ θ.μ.} \rightarrow \text{δμΣ} = 14 \text{ bits}$ (είναι πάντα ο εκθέτης του μεγέθους των σελίδων). Προσοχή! Πρέπει πρώτα η χωρητικότητα των σελίδων/πλαισίων σελίδων να μετατραπεί σε θ.μ. και μετά να υπολογίζουμε το μέγεθος του πεδίου δμΣ (είναι πάντα ο εκθέτης του πλήθους των σελίδων/πλαισίων σελίδων). Πάντα το πεδίο ΑΛΣ το υπολογίζουμε αφαιρώντας από το συνολικό μήκος της λογικής διεύθυνσης που είναι 64 bits το δμΣ που είναι 14 bits και έχουμε: $\text{ΑΛΣ} = 64 \text{ bits} - 14 \text{ bits} = 50 \text{ bits}$.

Ομοίως το πεδίο φδΣ (φυσική διεύθυνση μέσα στη σελίδα) το υπολογίζουμε αφαιρώντας από το συνολικό μήκος της φυσικής διεύθυνσης που είναι 35 bits, το δμΣ που είναι 14 bits και έχουμε: $\text{φδΣ} = \text{φυσική διεύθυνση} - \text{δμΣ} = 35 - 14 = 21 \text{ bits}$. Για να υπολογίζουμε το μέγεθος του Π.Σ. (Πίνακα Σελίδων) πολλαπλασιάζουμε το πλήθος των γραμμών του με το μέγεθος κάθε γραμμής του. Καταρχήν, **το μέγεθος κάθε γραμμής του Π.Σ. = 7 bits ελέγχου + φδΣ = 7 + 21 = 28 bits**. Άρα μέγεθος Π.Σ. = πλήθος γραμμών x μέγεθος κάθε γραμμής = $2^{\text{ΑΛΣ}} \text{ γραμμές} \times \text{μέγεθος κάθε γραμμής} = 2^{50} \text{ γραμμές} \times 28 \frac{\text{bits}}{\text{γραμμή}} = 2^{30} \times 2^{20} \times 28 \text{ bits} = 2^{30} \times 2^{17} \times 28 \text{ bytes} = 2^{17} \times 28 \text{ GB} > 32 \text{ GB (KM)} \rightarrow \text{o Π.Σ. δεν χωρά να αποθηκευτεί στην κύρια μνήμη. Δηλαδή με τη συγκεκριμένη τεχνική της σελιδοποίησης, ο Π.Σ δεν χωρά να αποθηκευτεί στην κύρια μνήμη (KM).}$

A' Τρόπος μείωσης χωρητικότητας Π.Σ.: χρήση πολλών επιπέδων



Άσκηση 5.16 – Ιδεατή μνήμη με την τεχνική των πολλών επιπέδων

Θεωρείστε υπολογιστή με κύρια μνήμη 32M ψηφιολέξεις και οργάνωση μιας ψηφιολέξης ανά θέση μνήμης. Η ιδεατή μνήμη χρησιμοποιεί λογικές διευθύνσεις των 39 δυαδικών ψηφίων και μέγεθος σελίδας 4K ψηφιολέξεις. Υποθέστε ότι η ιδεατή μνήμη

υλοποιείται με την τεχνική της σελιδοποίησης. Ο Πίνακας σελίδων υλοποιείται σε τρία επίπεδα υπό μορφή δένδρου και οι υποπίνακες του πίνακα σελίδων αποθηκεύονται στην ιδεατή μνήμη. Για την προσπέλαση **οποιουδήποτε υποπίνακα του πίνακα σελίδων χρησιμοποιείται το ίδιο πλήθος δυαδικών ψηφίων**. Θεωρήστε ότι οι πρώτες λογικές διευθύνσεις που παράγονται από τον επεξεργαστή είναι οι ακόλουθες $7235672237753_{(8)}$, $7235672237754_{(8)}$, $7235672264340_{(8)}$ $7235632214354_{(8)}$ και $7721232216754_{(8)}$. Να σχεδιάσετε τη δομή του πίνακα σελίδων που βρίσκεται στην κύρια μνήμη μετά από καθεμία λογική διεύθυνση που παράγει ο επεξεργαστής. Να δικαιολογήσετε τις απαντήσεις σας.

Λύση

Πάντα από το μέγεθος σελίδων = μέγεθος πλαισίων σελίδων υπολογίζουμε το πεδίο **δμΣ** (**διεύθυνση μέσα στη Σελίδα**). Δηλαδή $4 \text{ KB} = 2^2 \times 2^{10} \text{ bytes} = 2^{12} \text{ bytes} = (\lambda\text{γώ οργάνωσης 1 ψηφιολέξη}/\theta.\mu.) 2^{12} \theta.\mu. \rightarrow \delta\mu\Sigma = 12 \text{ bits}$ (είναι πάντα ο εκθέτης του μεγέθους των σελίδων). Προσοχή! Πρέπει πρώτα η χωρητικότητα των σελίδων/πλαισίων σελίδων να μετατραπεί σε θ.μ. και μετά να υπολογίζουμε το μέγεθος του πεδίου δμΣ (είναι πάντα ο εκθέτης του πλήθους των σελίδων/πλαισίων σελίδων). Πάντα το πεδίο ΑΛΣ το υπολογίζουμε αφαιρώντας από το συνολικό μήκος της λογικής διεύθυνσης που είναι 39 bits το δμΣ που είναι 12 bits και έχουμε: **ΑΛΣ = 39 bits – 12 bits = 27 bits = 9 ψηφία στο οκταδικό**. Από το μέγεθος της K.M. υπολογίζουμε το μήκος της φυσικής διεύθυνσης, που είναι $32M$ ψηφιολέξεις $= 2^5 \times 2^{20} \text{ bytes} = 2^{25} \text{ bytes} = 2^{25} \theta.\mu.$ επομένως η φυσική διεύθυνση έχει μέγεθος 25 bits. Ομοίως το πεδίο φδΣ (φυσική διεύθυνση μέσα στη σελίδα) το υπολογίζουμε αφαιρώντας από το συνολικό μήκος της φυσικής διεύθυνσης που είναι 25 bits, το δμΣ που είναι 12 bits και έχουμε: **φδΣ = φυσική διεύθυνση – δμΣ = 25 – 12 = 13 bits**. Για να υπολογίζουμε το **μέγεθος του Π.Σ.** (Πίνακα Σελίδων) πολλαπλασιάζουμε το πλήθος των γραμμών του με το μέγεθος κάθε γραμμής του. Καταρχήν, **το μέγεθος κάθε γραμμής του Π.Σ. = 7 bits ελέγχου + φδΣ = 7 + 13 = 20 bits**. Άρα μέγεθος Π.Σ. = πλήθος γραμμών \times μέγεθος κάθε γραμμής $= 2^{\text{ΑΛΣ}} \text{ γραμμές} \times \text{μέγεθος κάθε γραμμής} = 2^{27} \text{ γραμμές} \times 20 \frac{\text{bits}}{\text{γραμμή}} = 2^7 \times 2^{20} \times 20 \text{ bits} = 2^{20} \times 2^4 \times 20 \text{ bytes} = 320 \text{ MB} >> 32 \text{ MB (KM)} \rightarrow$ ο Π.Σ. δεν χωρά να αποθηκευτεί στην κύρια μνήμη.

Η λογική διεύθυνση αποτελείται από δύο μέρη:

ΑΛΣ = 27 bits

δμΣ = 12 bits

Αριθμός λογικής σελίδας

Διεύθυνση μέσα στη σελίδα

Η φυσική διεύθυνση αποτελείται από δύο μέρη, όπως φαίνεται στη συνέχεια:

φδΣ = 13 bits

δμΣ = 12 bits

Φυσική διεύθυνση σελίδας

Διεύθυνση μέσα στη σελίδα

Επειδή ο Πίνακας σελίδων υλοποιείται σε τρία επίπεδα υπό μορφή δένδρου και επειδή για την προσπέλαση οποιουδήποτε υποπίνακα του πίνακα σελίδων χρησιμοποιείται το **ίδιο πλήθος δυαδικών ψηφίων**, θα πρέπει να διαιρέσουμε το πεδίο ΑΛΣ σε τρία επιμέρους πεδία ίσου μεγέθους το καθένα, οπότε θα έχουμε:

Η λογική διεύθυνση αποτελείται από δύο μέρη:

ΑΛΣ1 = 9 bits = 3 οκταδικά

ΑΛΣ2 = 9 bits = 3 οκταδικά

ΑΛΣ3 = 9 bits = 3 οκταδικά

δμΣ = 12 bits = 4 οκταδικά

Η φυσική διεύθυνση αποτελείται από δύο μέρη, όπως φαίνεται στη συνέχειαQ

Διεύθυνση μέσα στη σελίδα

φδΣ = 13 bits

δμΣ = 12 bits

Φυσική διεύθυνση σελίδας

Διεύθυνση μέσα στη σελίδα

Οι λογικές διευθύνσεις που παράγονται από την KME είναι:

ΑΛΣ1 – ΑΛΣ2 – ΑΛΣ3 – δμΣ

723 567 223 7753₍₈₎

723 567 223 7754₍₈₎

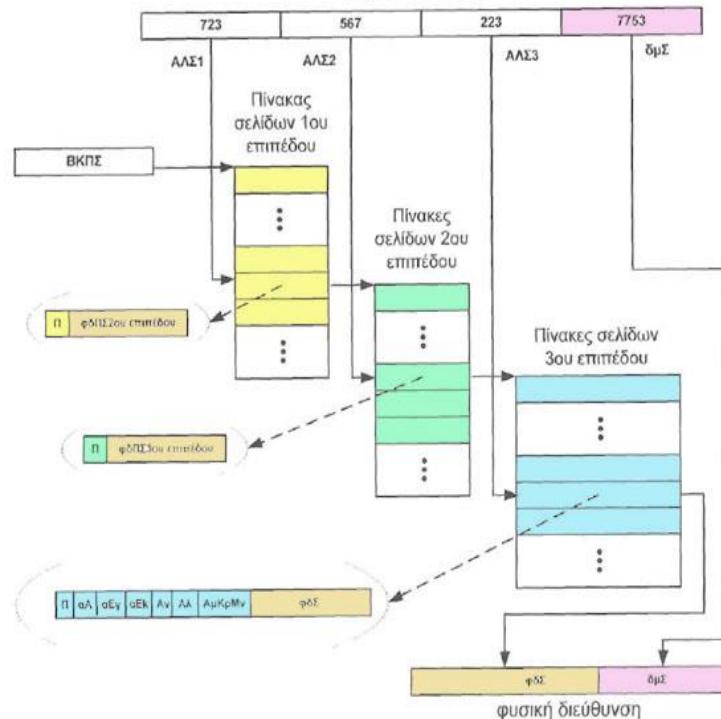
723 567 226 4340₍₈₎

723 563 221 4354₍₈₎

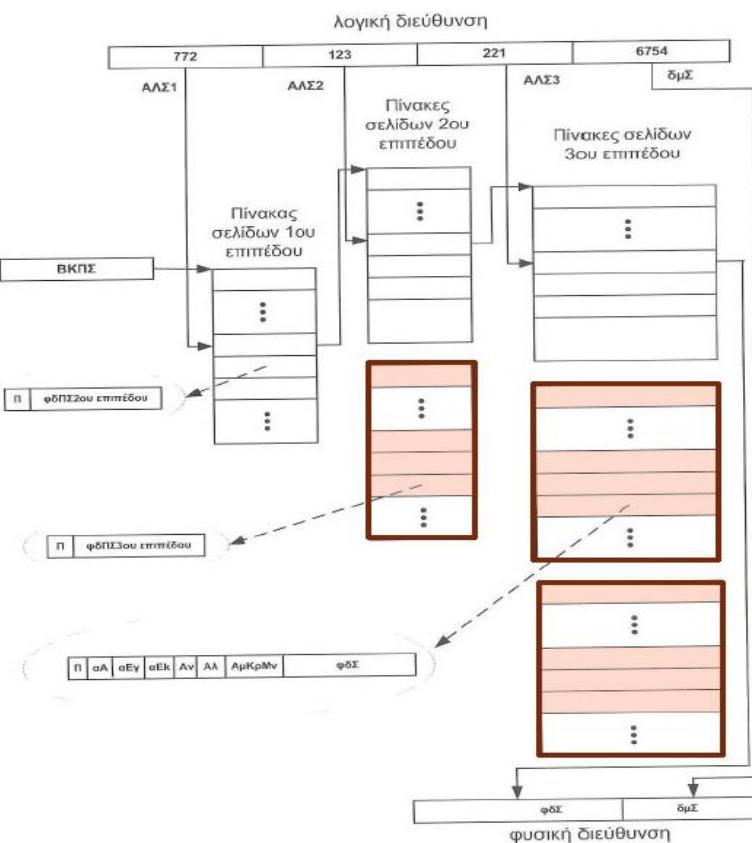
772 123 221 6754₍₈₎

Κανόνας δημιουργίας νέων Π.Σ.: κάθε φορά που παράγεται μια νέα λογική διεύθυνση, συγκρίνεται η νέα λογική διεύθυνση που παράγει ο επεξεργαστής με την αμέσως προηγούμενή της και αν διαφέρουν στο πεδίο δμΣ ή στο τελευταίο ΑΛΣ (όποιο και αν είναι αυτό), τότε στο σχήμα δεν επέρχεται καμία διαφορά. Αν όμως διαφέρουν σε άλλα ΑΛΣ **εκτός** του τελευταίου, τότε προστίθενται νέοι Π.Σ. σε **κάθε επόμενο** επίπεδο από αυτό που διαφέρουν..

Αρχικά υπάρχει σε κάθε επίπεδο ένας Π.Σ. όπως φαίνεται στο επόμενο σχήμα, το οποίο παρουσιάζει την αρχική κατάσταση.



Στη συγκρίνουμε τη 2^η λογική διεύθυνση με την 1^η και παρατηρούμε ότι η διαφορά τους έγκειται στο πεδίο δμΣ, άρα δεν προστίθεται νέος Π.Σ. Κατόπιν συγκρίνουμε τη 3^η λογική διεύθυνση με την 2^η και παρατηρούμε ότι η διαφορά τους έγκειται στο πεδίο ΑΛΣ3 (τελευταίο ΑΛΣ), άρα και πάλι δεν προστίθεται κανένας νέος Π.Σ. Κατόπιν συγκρίνουμε τη 4^η λογική διεύθυνση με την 3^η και παρατηρούμε ότι η διαφορά τους έγκειται στο πεδίο ΑΛΣ2 (που δεν είναι το τελευταίο ΑΛΣ), κάτι που σημαίνει ότι τώρα προστίθεται ένας νέος Π.Σ. στο 3^ο επίπεδο (στο επόμενο από αυτό που διαφέρουν). Τέλος, συγκρίνουμε τη 5^η λογική διεύθυνση με την 4^η και παρατηρούμε ότι η διαφορά τους έγκειται στο πεδίο ΑΛΣ1 (που και πάλι δεν είναι το τελευταίο ΑΛΣ), κάτι σημαίνει ότι τώρα προστίθενται νέοι Π.Σ. τόσο στο 2^ο όσο στο 3^ο επίπεδο (σε κάθε επόμενο επίπεδο από αυτό που διαφέρουν). Συνολικά, θα παραχθούν 3 νέοι Π.Σ., δύο στο τελευταίο επίπεδο και ένας στο προτελευταίο, όπως φαίνεται στο επόμενο σχήμα, που δείχνει την τελική μορφή των Π.Σ.



Άσκηση 5.17 – Ιδεατή μνήμη με την τεχνική των πολλών επιπέδων

Θεωρήστε υπολογιστή με κύρια μνήμη 32M ψηφιολέξεις και οργάνωση μιας ψηφιολέξης ανά θέση μνήμης. Η ιδεατή μνήμη χρησιμοποιεί λογικές διευθύνσεις των 36 δυαδικών ψηφίων και μέγεθος σελίδας 4 Κψηφιολέξεις. Υποθέστε ότι η ιδεατή μνήμη υλοποιείται με την τεχνική της σελιδοποίησης. Να δικαιολογήσετε τις απαντήσεις σας.

a. Να υπολογίσετε το μέγεθος της κυρίας μνήμης που απαιτείται για την αποθήκευση του πίνακα σελίδων εάν αυτός υλοποιείται σε ένα επίπεδο.

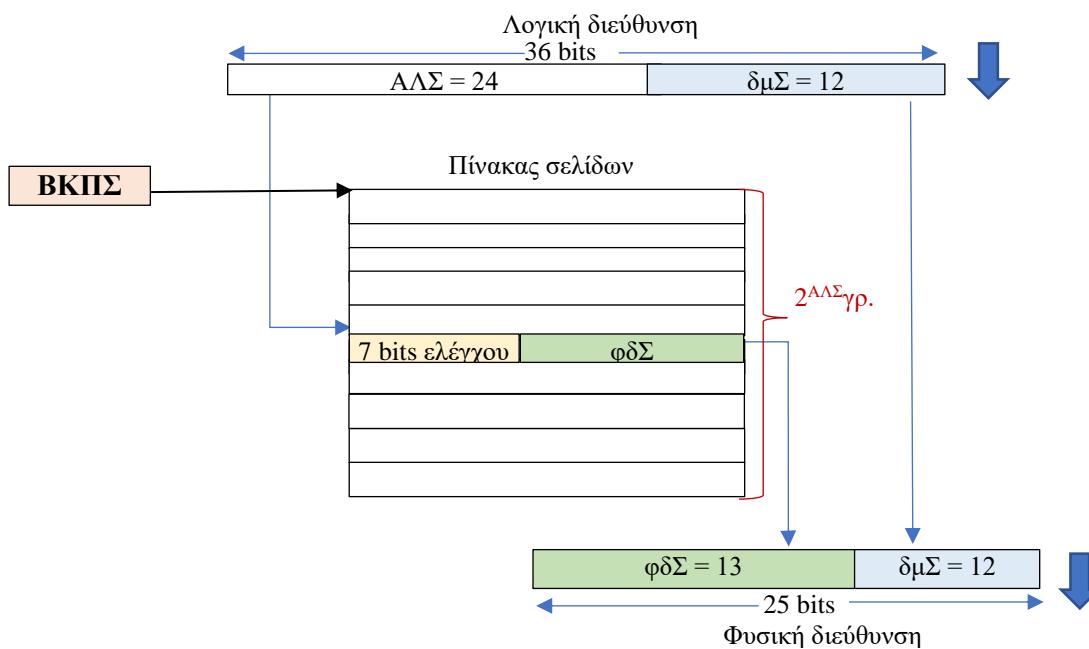
β. Θεωρήστε μια διαδικασία στην οποία ο χώρος των λογικών διευθύνσεων καταλαμβάνει 256 σελίδες. Να υπολογίσετε το μέγεθος της κυρίας μνήμης που απαιτείται για την αποθήκευση του πίνακα σελίδων στη χειρότερη και στην καλύτερη των περιπτώσεων εάν αυτός υλοποιείται σε **δύο** επίπεδα.

Δεδομένα:

- KM: $32 \text{ MB} = 2^5 \times 2^{20} \text{ bytes} = 2^{25} \text{ bytes}$ = (λόγω οργάνωσης 1 byte/θ.μ.) $2^{25} \text{ θ.μ.} \rightarrow$ μήκος φυσικής διεύθυνσης = 25 bits
- Μήκος λογικής διεύθυνσης = 36 bits (πάντα δίνεται),
- Μέγεθος σελίδων = μέγεθος πλαισίων σελίδων = 4 KB

Λύση

a. Οι σελίδες βρίσκονται στην ιδεατή μνήμη και τα πλαίσια σελίδων βρίσκονται στην κύρια μνήμη. Ισχύει ότι το **πλήθος σελίδων >> πλήθος πλαισίων σελίδων**. Πάντα από το μέγεθος σελίδων = μέγεθος πλαισίων σελίδων υπολογίζουμε το **δμΣ (διεύθυνση μέσα στη Σελίδα)**. Δηλαδή $4 \text{ KB} = 2^2 \times 2^{10} \text{ bytes} = 2^{12} \text{ bytes}$ = (λόγω οργάνωσης 1 ψηφιολέξη/θ.μ.) $2^{12} \text{ θ.μ.} \rightarrow \delta\mu\Sigma = 12 \text{ bits}$ (είναι πάντα ο εκθέτης του μεγέθους των σελίδων). Επομένως το πεδίο ΑΛΣ = 36 bits – 12 bits = 24 bits. Πάντα το πεδίο ΑΛΣ υπολογίζεται με αφαίρεση. Επίσης, πάντα το πεδίο φδΣ (φυσική διεύθυνση μέσα στη σελίδα) το υπολογίζουμε αφαιρώντας από το συνολικό μήκος της KM που είναι 25 bits, το δμΣ που είναι 12 bits και έχουμε: φδΣ = φυσική διεύθυνση – δμΣ = $25 - 12 = 13 \text{ bits}$. Για να υπολογίζουμε το μέγεθος του Π.Σ. (Πίνακα Σελίδων) πολλάζουμε το πλήθος των γραμμών του με το μέγεθος κάθε γραμμής του. Καταρχήν, το μέγεθος κάθε γραμμής του Π.Σ. = 7 bits ελέγχου + φδΣ = $7 + 13 = 20 \text{ bits}$. Άρα μέγεθος Π.Σ. = πλήθος γραμμών x μέγεθος κάθε γραμμής = $2^{\text{ΑΛΣ}} \times \text{μέγεθος κάθε γραμμής} = 2^{24} \text{ γραμμές} \times 20 \frac{\text{bits}}{\text{γραμμή}} = 2^4 \times 2^{20} \times 20 \text{ bits} = 2 \times 2^{20} \times 20 \text{ bytes} = 40 \text{ MB} > 32 \text{ MB(KM)} \rightarrow$ ο Π.Σ. **δεν** χωρά να αποθηκευτεί στην κύρια μνήμη. Δηλαδή με τη συγκεκριμένη τεχνική της **σελιδοποίησης**, ο Π.Σ δεν χωρά να αποθηκευτεί στην κύρια μνήμη (KM). Στη συνέχεια ακολουθεί ο σχεδιασμός της ιδεατής μνήμης με την τεχνική της σελιδοποίησης με ένα επίπεδο.



b. **Κανόνας πολλαπλών επιπέδων:** όταν ο επεξεργαστής παράγει λογικές διευθύνσεις, συγκρίνουμε κάθε φορά την επόμενη λογική διεύθυνση με την προηγούμενή της και αν αυτές διαφέρουν στο δμΣ ή στο τελευταίο ΑΛΣ, στο σχήμα δεν επέρχεται καμία διαφορά. Όταν όμως διαφέρουν σε άλλα ΑΛΣ εκτός του τελευταίου, τότε προστίθενται νέοι Π.Σ. **σε κάθε επόμενο επίπεδο** από αυτό που διαφέρουν. Για παράδειγμα, έστω ότι ο επεξεργαστής παράγει τις διευθύνσεις 7235672237753₈, 7235672237754₈,

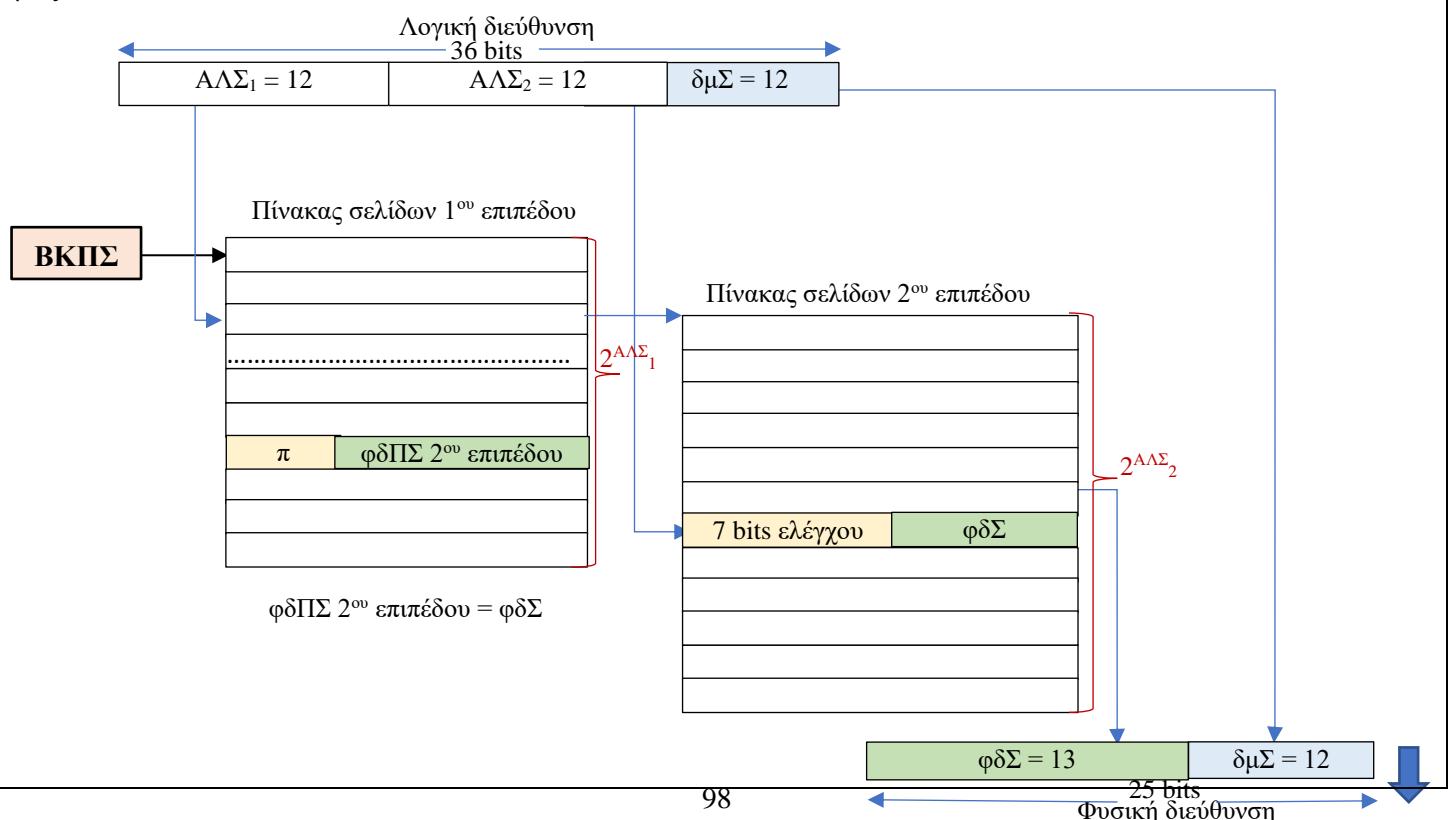
7235672264340₈, 7235632214354₈, 7721232216754₈ και έστω ότι έχουμε 3 επίπεδα με μεγέθη $\text{ΑΛΣ}_1 = \text{ΑΛΣ}_2 = \text{ΑΛΣ}_3 = 9 \text{ bits} = 3 \text{ οκταδικά ψηφία}$ και $\delta\mu\Sigma = 12 \text{ bits} = 4 \text{ οκταδικά ψηφία}$. Επομένως, αν χωρίσουμε τις προαναφερόμενες λογικές διεύθυνσεις σε τέσσερα τμήματα, θα έχουμε:

ΑΛΣ_1	ΑΛΣ_2	ΑΛΣ_3	$\delta\mu\Sigma$
723	567	223	7753
723	567	223	7754
723	567	226	4340
723	563	221	4354
772	123	221	6754

Συγκρίνοντας την πρώτη με τη δεύτερη διεύθυνση, παρατηρούμε ότι διαφέρουν στο $\delta\mu\Sigma$, οπότε δεν προστίθεται κανένας νέος Π.Σ. Συγκρίνοντας τη δεύτερη με την τρίτη διεύθυνση, παρατηρούμε ότι διαφέρουν στο ΑΛΣ_3 (τελευταίο Α.Λ.Σ.), οπότε δεν προστίθεται κανένας νέος Π.Σ. Συγκρίνοντας την τρίτη με την τέταρτη διεύθυνση, παρατηρούμε ότι διαφέρουν στο ΑΛΣ_2 (που δεν είναι το τελευταίο Α.Λ.Σ.), οπότε προστίθεται ένας νέος Π.Σ. στο επόμενο επίπεδο από αυτό που διαφέρουν, δηλαδή αφού διαφέρουν στο δεύτερο επίπεδο, θα προστεθεί ένας νέος Π.Σ. στο τρίτο επίπεδο. Συγκρίνοντας την τέταρτη με την πέμπτη διεύθυνση, παρατηρούμε ότι διαφέρουν στο ΑΛΣ_1 , οπότε προστίθενται νέοι Π.Σ. σε κάθε επόμενο επίπεδο από αυτό που διαφέρουν στο πρώτο επίπεδο, θα προστεθεί ένας νέος Π.Σ. στο δεύτερο επίπεδο και ακόμη ένας νέος Π.Σ. στο τρίτο επίπεδο. Συνολικά, θα έχουν προστεθεί 3 νέοι Π.Σ.

Στην προκειμένη περίπτωση οι 256 σελίδες έχουν 256 αντίστοιχες λογικές διεύθυνσεις. Η **καλύτερη περίπτωση** είναι όταν και οι 256 σελίδες έχουν **ίδιο** ΑΛΣ_1 , οπότε τότε δεν χρειάζεται να προστεθούν νέοι Π.Σ. Η **χειρότερη περίπτωση** είναι όταν και οι 256 σελίδες έχουν **διαφορετικό** ΑΛΣ_1 , οπότε τότε χρειάζεται να προστεθούν νέοι Π.Σ. σε κάθε επόμενο επίπεδο από αυτό που διαφέρουν, δηλαδή στην προκειμένη περίπτωση στο δεύτερο επίπεδο. Αν έχουν διαφορετικό ΑΛΣ_2 , επειδή αυτό είναι το τελευταίο Α.Λ.Σ., δεν προστίθενται νέοι Π.Σ. στην περίπτωση αυτή.

Για να μειώσουμε τη χωρητικότητα του Π.Σ. χρησιμοποιούμε την τεχνική των δύο επιπέδων. Όταν χρησιμοποιούμε τη συγκεκριμένη τεχνική, θα πρέπει το **πεδίο Α.Λ.Σ να υποδιαιρεθεί σε τόσα επιμέρους πεδία ώστα είναι το πλήθος των επιπέδων**, στην προκειμένη περίπτωση σε δύο πεδία ($\text{ΑΛΣ}_1 - \text{ΑΛΣ}_2$). Το αν θα είναι τα δύο πεδία ίσα μεταξύ τους, το προσδιορίζει η εκφώνηση. Στην προκειμένη περίπτωση, επειδή δεν αναφέρεται κάτι σχετικό, θα λάβουμε υπόψη μας τρεις διαφορετικές περιπτώσεις: $\text{ΑΛΣ}_1 = \text{ΑΛΣ}_2$, $\text{ΑΛΣ}_1 > \text{ΑΛΣ}_2$, $\text{ΑΛΣ}_1 < \text{ΑΛΣ}_2$. Σε κάθε περίπτωση το άθροισμα $\text{ΑΛΣ}_1 + \text{ΑΛΣ}_2$ δεν θα πρέπει να ξεπερνά το συνολικό μήκος του Α.Λ.Σ που είναι 24.



A) Καλύτερη περίπτωση: Χωρητικότητα Π.Σ. = χωρητικότητα Π.Σ. 1^{ον} επιπέδου + χωρητικότητα Π.Σ. 2^{ον} επιπέδου = $2^{\text{ΑΛΣ}_1}$ γραμμές x ($\pi + \phi\Delta\Sigma 2^{\text{ον}} \text{ επιπέδου}$) $\frac{\text{bits}}{\text{γραμμή}}$ + $2^{\text{ΑΛΣ}_2}$ γραμμές x ($7 + \phi\Delta\Sigma$) $\frac{\text{bits}}{\text{γραμμή}}$ = 2^{12} γραμμές x (1 + 13) $\frac{\text{bits}}{\text{γραμμή}}$ + 2^{12} γραμμές x (7 + 13) $\frac{\text{bits}}{\text{γραμμή}}$ = 2^{12} γραμμές x (14 + 20) $\frac{\text{bits}}{\text{γραμμή}}$ = 2^{12} γραμμές x 34 $\frac{\text{bits}}{\text{γραμμή}}$ = $2^{12} \times 34 \text{ bits} = 2^9 \times 34 \text{ bytes} = 17408 \text{ bytes} = 17 \text{ KB}$.

B) Χειρότερη περίπτωση: Χωρητικότητα Π.Σ. = χωρητικότητα Π.Σ. 1^{ον} επιπέδου + χωρητικότητα Π.Σ. 2^{ον} επιπέδου = $2^{\text{ΑΛΣ}_1}$ γραμμές x ($\pi + \phi\Delta\Sigma 2^{\text{ον}} \text{ επιπέδου}$) $\frac{\text{bits}}{\text{γραμμή}}$ + $2^{\text{ΑΛΣ}_2}$ γραμμές x ($7 + \phi\Delta\Sigma$) $\frac{\text{bits}}{\text{γραμμή}}$ x **256** = 2^{12} γραμμές x (1 + 13) $\frac{\text{bits}}{\text{γραμμή}}$ + 2^{12} γραμμές x (7 + 13) $\frac{\text{bits}}{\text{γραμμή}}$ x **256** = 2^{12} γραμμές x (14 + 20 x 256) $\frac{\text{bits}}{\text{γραμμή}}$ = 2^{12} γραμμές x (14 + 5120) $\frac{\text{bits}}{\text{γραμμή}}$ = 2^{12} γραμμές x 5134 $\frac{\text{bits}}{\text{γραμμή}}$ = $2^{12} \times 5134 \text{ bits} = 2^9 \times 5134 \text{ bytes} = 2.628.608 \text{ bytes} = 2567 \text{ KB}$.

Αν υποθέσουμε ότι $\text{ΑΛΣ}_1 > \text{ΑΛΣ}_2$, τότε για $\text{ΑΛΣ}_1 = 14$ και $\text{ΑΛΣ}_2 = 10$ (πρέπει το συνολικό άθροισμα των επιμέρους ΑΛΣ να συνεχίσει να είναι 24 bits), τότε θα έχουμε ότι $2^{\text{ΑΛΣ}_1}$ γραμμές = 2^{14} γραμμές και $2^{\text{ΑΛΣ}_2}$ γραμμές = 2^{10} γραμμές. Αν υποθέσουμε ότι $\text{ΑΛΣ}_1 < \text{ΑΛΣ}_2$, τότε για $\text{ΑΛΣ}_1 = 10$ και $\text{ΑΛΣ}_2 = 14$ (πρέπει το συνολικό άθροισμα των επιμέρους ΑΛΣ να συνεχίσει να είναι 24 bits), τότε θα έχουμε ότι $2^{\text{ΑΛΣ}_1}$ γραμμές = 2^{10} γραμμές και $2^{\text{ΑΛΣ}_2}$ γραμμές = 2^{14} γραμμές.

Θέμα Ιονίου 2019 - – Ιδεατή μνήμη με την τεχνική των πολλών επιπέδων και ΑΠΣ

Θεωρείστε υπολογιστή με κύρια μνήμη 4GBytes και οργάνωση ενός byte ανά θέση μνήμης. Η ιδεατή μνήμη χρησιμοποιεί λογικές διευθύνσεις των 42 δυαδικών ψηφίων και μέγεθος σελίδας 4 KB.

1. Να υπολογίσετε το χώρο που καταλαμβάνει ο Π.Σ. στην κύρια μνήμη όταν υλοποιηθεί σε ένα επίπεδο.
2. Θεωρείστε μια διαδικασία στην οποία ο χώρος των λογικών διευθύνσεων καταλαμβάνει 200 σελίδες. Ο πίνακας σελίδων έχει υλοποιηθεί με τον κλασσικό τρόπο σε δύο επίπεδα υπό μορφή δέντρου. Τα πεδία που χρησιμοποιούνται για τη διεύθυνσιο δύτηση του πρώτου και του δεύτερου επιπέδου έχουν το ίδιο μήκος. Να υπολογίσετε το μέγεθος της κυρίας μνήμης που απαιτείται για την αποθήκευση του πίνακα σελίδων στη χειρότερη και στην καλύτερη των περιπτώσεων.
3. Να υπολογίσετε το χώρο που καταλαμβάνει ο πίνακας σελίδων στην κύρια μνήμη όταν για την υλοποίηση του πίνακα σελίδων έχει χρησιμοποιηθεί η τεχνική του αντεστραμμένου πίνακα σελίδων με 4 αντιστοιχίσεις ανά θέση.

Υπενθύμιση: Το πλήθος των δυαδικών ψηφίων ελέγχου είναι 7.

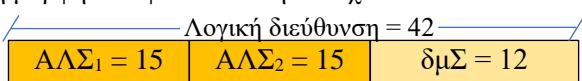
Λύση

1) Από τη **χωρητικότητα της κύριας μνήμης που είναι 4 GB** θα υπολογίσουμε το μήκος της φυσικής διεύθυνσης που είναι 4 GB = $2^2 \times 2^{30}$ bytes = 2^{32} bytes = (λόγω της οργάνωσης ενός byte ανά θέση μνήμης) $2^{32} \theta.\mu..$ Άρα το μήκος της φυσικής διεύθυνσης είναι 32 bits. **Στη συνέχεια από το μέγεθος της σελίδας = μέγεθος πλαισίου σελίδας** θα υπολογίσουμε το μέγεθος του πεδίου **δμΣ**. Πιο συγκεκριμένα, θα έχουμε 4KB = $2^2 \times 2^{10}$ bytes = 2^{12} bytes = (λόγω της οργάνωσης ενός byte ανά θέση μνήμης) $2^{12} \theta.\mu..$ Άρα το **μήκος του πεδίου δμΣ = 12**. Το **μήκος της λογικής διεύθυνσης δίνεται ίσο με 42 bits**. Άρα το $\text{ΑΛΣ} = \text{λογική διεύθυνση} - \delta\text{μΣ} = 42 - 12 = 30$ bits και το $\phi\Delta\Sigma = \text{φυσική διεύθυνση} 32 - 12 = 20$ bits. Δηλαδή:



Μέγεθος (χωρητικότητα) **Π.Σ.** = πλήθος γραμμών x μέγεθος κάθε γραμμής = $2^{\text{ΑΛΣ}}$ γρ. x μέγεθος κάθε γραμμής = 2^{30} γρ. x (7 bits ελέγχου + $\phi\Delta\Sigma$) = 2^{30} γρ. x $27 \frac{\text{bits}}{\text{γρ.}}$ = 2^{30} γρ. x 27 bits = 2^{27} γρ. x 27 bytes = $2^{20} \times 2^7 \text{ γρ.} \times 27 \text{ bytes} = 2^7 \times 27 \text{ MB}$.

2) Εχουμε **200 σελίδες άρα και 200 λογικές διευθύνσεις** και επειδή έχουμε **2 επίπεδα ίδιου μήκους**, κάθε λογική διεύθυνση έχει τη μορφή που φαίνεται στη συνέχεια:

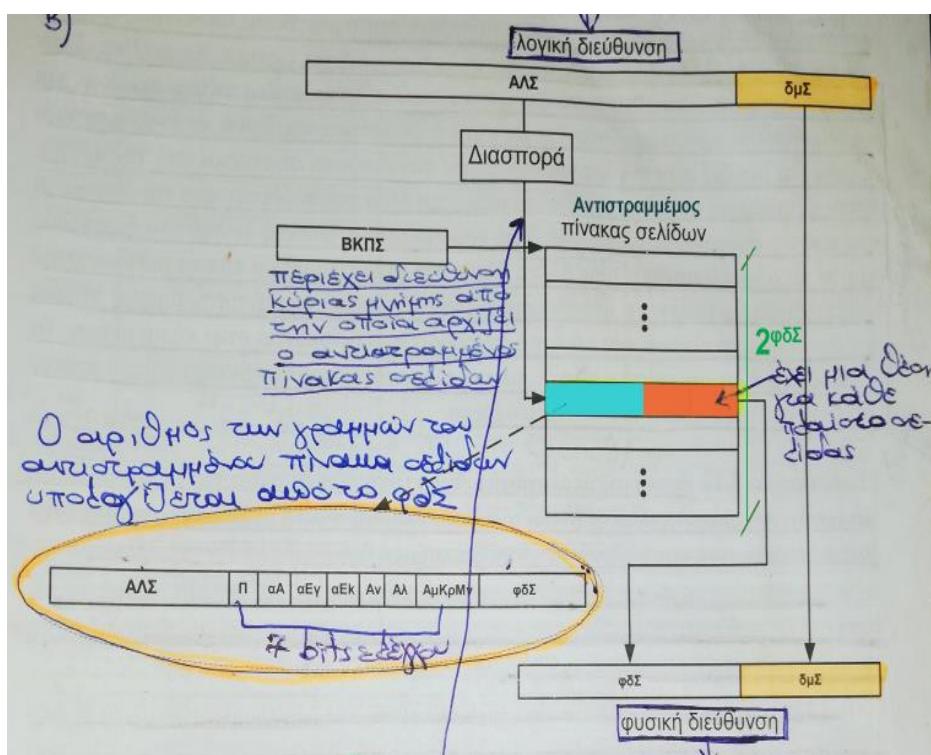


Η **καλύτερη περίπτωση** προκύπτει όταν **όλες** οι σελίδες έχουν το **ίδιο ΑΛΣ₁** και τότε το συνολικό μέγεθος του Π.Σ. των δύο επιπέδων είναι: Μέγεθος = Μέγεθος Π.Σ. 1^{ον} επιπέδου + Μέγεθος Π.Σ. 2^{ον} επιπέδου = $2^{15} \text{ γρ.} \times (1 + 20) \frac{\text{bits}}{\text{γρ.}}$ + $2^{15} \text{ γρ.} \times (7 + 20) \frac{\text{bits}}{\text{γρ.}}$ = $2^{15} \text{ γρ.} \times 48 \frac{\text{bits}}{\text{γρ.}} = 2^{12} \times 48 \text{ bytes} = 192 \text{ KB}$

Η **χειρότερη περίπτωση** προκύπτει όταν όλες οι σελίδες έχουν **διαφορετικό ΑΛΣ1**, οπότε τότε προστίθενται 200 νέοι Π.Σ. στο δεύτερο επίπεδο. Τότε το συνολικό μέγεθος του Π.Σ. των δύο επιπέδων είναι: Μέγεθος = Μέγεθος Π.Σ. 1^ο επιπέδου + Μέγεθος Π.Σ. 2^ο επιπέδου = $2^{15} \text{ γρ. x } (1 + 20) \frac{\text{bits}}{\text{γρ.}} + 2^{15} \text{ γρ. x } (7 + 20) \frac{\text{bits}}{\text{γρ.}} \times 200 = 2^{15} \text{ γρ. x } 5421 \frac{\text{bits}}{\text{γρ.}} = 2^{12} \times 5421 \text{ bytes} = 21684 \text{ KB} = 21.17 \text{ MB}$

21.17 MB

3) Ένας Α.Π.Σ. (inverted page table) περιέχει **μια γραμμή για κάθε πλαίσιο σελίδας του χώρου των φυσικών διευθύνσεων** και όχι για κάθε σελίδα του χώρου των λογικών διευθύνσεων. Αυτό σημαίνει ότι το συνολικό πλήθος του γραμμών του Α.Π.Σ. είναι $2^{\Phi\Delta\Sigma}$ (και όχι $2^{\Lambda\Lambda\Sigma}$), επομένως πολύ λιγότερων από πριν. Κάθε γραμμή του inverted page table (ΑΠΣ) υποδιαιρείται σε δύο ή περισσότερες αντιστοιχίσεις (υποδιαιρέσεις) και κάθε **αντιστοιχίση** με τη σειρά της αποτελείται από τρία πεδία: **7 bits ελέγχου + φδΣ + ΑΛΣ**. Το **Μέγεθος Α.Π.Σ. = μέγεθος κάθε γραμμής x πλήθος γραμμών Α.Π.Σ. = πλήθος αντιστοιχίσεων κάθε γραμμής x μέγεθος κάθε αντιστοιχίσης x πλήθος γραμμών Α.Π.Σ.**, όπου πλήθος γραμμών Α.Π.Σ. = $2^{\Phi\Delta\Σ}$. Δηλαδή μπορεί στη συγκεκριμένη περίπτωση να μεγαλώνει το μέγεθος κάθε γραμμής του ΑΠΣ, από την άλλη όμως μειώνεται σε μεγάλο βαθμό το πλήθος των γραμμών του ΑΠΣ. Επίσης, υπάρχει μια συνάρτηση διασποράς (Διασπορά), η οποία ονομάζεται **hash function** και η οποία αναγνωρίζεται με το όνομα: **σδμΑΠΣ = σχετική διεύθυνση μέσα στον Α.Π.Σ. και εντοπίζει μια διεύθυνση μέσα στον Α.Π.Σ.** Ο καταχωρητής ΒΚΠΣ = Βασικός Καταχωρητής Πίνακα Σελίδων δείχνει τη διεύθυνση έναρξης της αποθήκευσης του Α.Π.Σ. στην κύρια μνήμη.



Έχουμε την **τεχνική του ΑΠΣ (Αντιστραμμένου Πίνακα Σελίδων)** με **4 αντιστοιχίσεις** ανά γραμμή του πίνακα. Κάθε αντιστοιχίση αποτελεί το άθροισμα τριών πεδίων: **7 bits ελέγχου + φδΣ + ΑΛΣ** = $7 + 20 \text{ bits} + 30 \text{ bits} = 57 \text{ bits}$ και επειδή -όπως αναφέραμε- έχουμε 4 αντιστοιχίσεις ανά γραμμή του πίνακα, το συνολικό μήκος κάθε γραμμής είναι $57 \text{ bits} \times 4 = 228 \text{ bits}$. Το συνολικό μέγεθος του ΑΠΣ είναι: **πλήθος γραμμών x μέγεθος κάθε γραμμής = $2^{\Phi\Delta\Σ} \text{ γρ.} \times \text{μέγεθος κάθε γραμμής} = 2^{20} \text{ γρ.} \times (57 \times 4 \frac{\text{bits}}{\text{γρ.}}) = 2^{19} \times 57 \text{ bytes}$**

Θέμα Ιουνίου 2020 με αντιστραμμένο πίνακα σελίδων

Θεωρείστε υπολογιστή με ιδεατή μνήμη που υλοποιείται με τη τεχνική της σελιδοποίησης. Το μέγεθος της σελίδας είναι 2^{18} bytes. Σε κάθε byte αντιστοιχεί μια διεύθυνση. Υποθέστε ότι η ιδεατή μνήμη υλοποιείται με την τεχνική του **αντεστραμμένου πίνακα σελίδων, με 2 αντιστοιχίσεις ανά θέση** του πίνακα. Το πλήθος των δυαδικών ψηφίων στον πίνακα σελίδων είναι 7. Η λογική διεύθυνση είναι των 48 bits ενώ η φυσική διεύθυνση των 50 bits. Να υπολογίσετε τη χωρητικότητα του αντεστραμμένου

πίνακα σελίδων σε bits. Να δώσετε τη χωρητικότητα χρησιμοποιώντας 6 ψηφία και το K, M ή G ανάλογα, π.χ. 003472K, 000321M ή 348848G. Εάν η χωρητικότητα μπορεί να δοθεί ως ακέραιος αριθμός σε Gbits πρέπει υποχρεωτικά να δοθεί σε Gbits, εάν δεν μπορεί να δοθεί ως ακέραιος αριθμός σε Gbits και μπορεί να δοθεί ως ακέραιος σε Mbits να δοθεί σε Mbits, διαφορετικά να δοθεί σε Kbits p.x. το 5120K πρέπει να δοθεί ως 5M.

Χωρητικότητα του αντεστραμμένου πίνακα σελίδων σε bits

Λύση

Χρησιμοποιούμε την τεχνική του αντιστραμμένου πίνακα σελίδων (Α.Π.Σ.). Με βάση την τεχνική αυτή, κάθε γραμμή του ΑΠΣ χωρίζεται σε δύο η περισσότερες αντιστοιχίσεις. **Το μέγεθος κάθε αντιστοιχίσης αποτελείται από τρία πεδία: 7 bits ελέγχου + φδΣ + ΑΛΣ.** Το μέγεθος κάθε γραμμής είναι το μέγεθος μιας αντιστοιχίσης x πλήθος αντιστοιχίσεων. Το συνολικό μέγεθος του ΑΠΣ = πλήθος γραμμών x μέγεθος κάθε γραμμής = $2^{\phi\Delta\Sigma}$ γραμμές x μέγεθος κάθε γραμμής. Παρατηρούμε ότι στον Α.Π.Σ. μεγαλώνει πολύ το μέγεθος κάθε γραμμής, αλλά από την άλλη μειώνεται δραστικά το πλήθος των γραμμών.

Από το μέγεθος της σελίδας = μέγεθος πλαισίου σελίδας = 2^{18} bytes = 2^{18} θέσεις μνήμης προκύπτει το μέγεθος του πεδίου **δμΣ = 18 bits**. Για να υπολογίζουμε το μέγεθος του πεδίου δμΣ τόσο της λογικής όσο και της φυσικής διεύθυνσης, πρέπει πάντα να μετατρέπουμε τις χωρητικότητες σε θέσεις μνήμης και μετά να υπολογίζουμε το πεδίο δμΣ. Στην προκειμένη περίπτωση δίνονται τα μεγέθη της λογικής (δίνεται **πάντα**) και της φυσικής διεύθυνσης (αν δεν δινόταν, θα το υπολογίζαμε από τη χωρητικότητα της κύριας μνήμης που θα δινόταν για το λόγο αυτό). Τα πεδία ΑΛΣ και φδΣ υπολογίζονται πάντα με αφαίρεση. Πιο συγκεκριμένα: **ΑΛΣ = λογική διεύθυνση – δμΣ = 48 – 18 = 30 bits** και το πεδίο **φδΣ = φυσική διεύθυνση – δμΣ = 50 – 18 = 32 bits**. Επίσης, το μέγεθος κάθε γραμμής του Α.Π.Σ., όπως προαναφέρθηκε είναι: μέγεθος μιας αντιστοιχίσης x πλήθος αντιστοιχίσεων = $(7 + 32 + 30)$ bits x 2 = 138 bits. Επομένως, το μέγεθος Α.Π.Σ. = πλήθος γραμμών x μέγεθος κάθε γραμμής = $2^{\phi\Delta\Sigma}$ γρ. x $\frac{138 \text{ bits}}{\text{γρ.}} = 2^{32} \times 138 \text{ bits} = 2^2 \times 2^{30} \times 138 \text{ bits} = 552 \text{ Gbits} \rightarrow 000552G$.

Σημείωση: Αν δεν δίνεται από την εκφώνηση η φυσική διεύθυνση άμεσα, θα δίνεται μέσω του μεγέθους της κύριας μνήμης (έμμεσα). Συγκεκριμένα, αν η K.M. έχει χωρητικότητα 2 GB = 2×2^{30} bytes = 2^{32} bytes = 2^{32} θ.μ. Επομένως το μέγεθος της φυσικής διεύθυνσης είναι 32 bits.

Θέμα Σεπτεμβρίου 2020 – Ιδεατή μνήμη με αντιστραμμένο πίνακα σελίδων

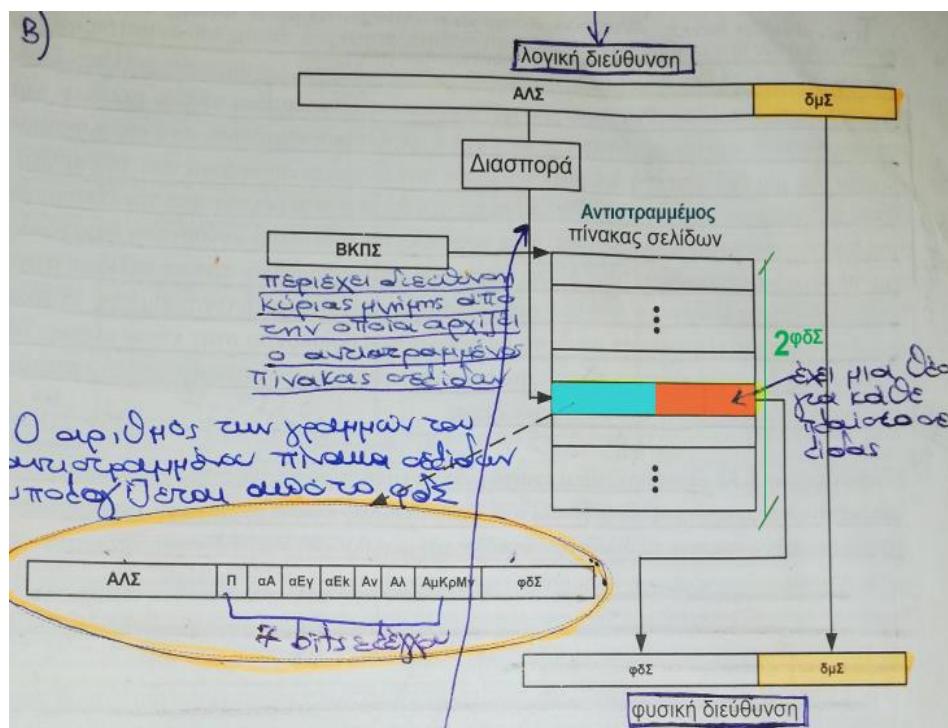
Θεωρείστε υπολογιστή με ιδεατή μνήμη που υλοποιείται με τη τεχνική της σελιδοποίησης. Το μέγεθος της σελίδας είναι 2^{11} bytes. Σε κάθε byte αντιστοιχεί μια διεύθυνση. Υποθέστε ότι η ιδεατή μνήμη υλοποιείται με την τεχνική του αντεστραμμένου πίνακα σελίδων, με 8 αντιστοιχίσεις ανά θέση του πίνακα. Το πλήθος των δυαδικών ψηφίων στον πίνακα σελίδων είναι 7. Η λογική διεύθυνση είναι των 65 bits ενώ η φυσική διεύθυνση των 34 bits. Να υπολογίσετε τη χωρητικότητα του αντεστραμμένου πίνακα σελίδων σε bits. Να δώσετε τη χωρητικότητα χρησιμοποιώντας 6 ψηφία και το K, M ή G ανάλογα, π.χ. 003472K, 000321M ή 348848G. Εάν η χωρητικότητα μπορεί να δοθεί ως ακέραιος αριθμός σε Gbits πρέπει υποχρεωτικά να δοθεί σε Gbits, εάν δεν μπορεί να δοθεί ως ακέραιος αριθμός σε Gbits και μπορεί να δοθεί ως ακέραιος σε Mbits να δοθεί σε Mbits, διαφορετικά να δοθεί σε Kbits p.x. το 5120K πρέπει να δοθεί ως 5M.

Χωρητικότητα του αντεστραμμένου πίνακα σελίδων σε bits

Λύση

Χρησιμοποιούμε την τεχνική του αντιστραμμένου πίνακα σελίδων (Α.Π.Σ.). Με βάση την τεχνική αυτή, κάθε γραμμή του ΑΠΣ χωρίζεται σε δύο η περισσότερες αντιστοιχίσεις. **Το μέγεθος κάθε αντιστοιχίσης αποτελείται από τρία πεδία: 7 bits ελέγχου + φδΣ + ΑΛΣ.** Το μέγεθος κάθε γραμμής είναι το μέγεθος μιας αντιστοιχίσης x πλήθος αντιστοιχίσεων. Το συνολικό μέγεθος του ΑΠΣ = πλήθος γραμμών x μέγεθος κάθε γραμμής = $2^{\phi\Delta\Sigma}$ x μέγεθος κάθε γραμμής. Παρατηρούμε ότι στον Α.Π.Σ. μεγαλώνει πολύ το μέγεθος κάθε γραμμής, αλλά από την άλλη μειώνεται δραστικά το πλήθος των γραμμών. Στη συνέχεια φαίνεται το σχήμα του ΑΠΣ (μία από τις δύο τεχνικές μείωσης της χωρητικότητας του Α.Π.Σ.).

Από το μέγεθος της σελίδας = μέγεθος πλαισίου σελίδας = 2^{11} bytes = 2^{11} θέσεις μνήμης προκύπτει το μέγεθος του πεδίου δμΣ = 11 bits. Για να υπολογίσουμε το μέγεθος του πεδίου δμΣ τόσο της λογικής όσο και της φυσικής διεύθυνσης, πρέπει πάντα να μετατρέπουμε τις χωρητικότητες σε θέσεις μνήμης και μετά να υπολογίζουμε το πεδίο δμΣ. Στην προκειμένη περίπτωση δίνονται τα μεγέθη της λογικής (δίνεται πάντα) και της φυσικής διεύθυνσης (αν δεν δινόταν, θα το υπολογίζαμε από τη χωρητικότητα της κύριας μνήμης που θα δινόταν για το λόγο αυτό). Τα πεδία ΑΛΣ και φδΣ υπολογίζονται πάντα με αφαίρεση. Πιο συγκεκριμένα: $\text{ΑΛΣ} = \text{λογική διεύθυνση} - \delta\mu\Sigma = 65 - 11 = 54$ bits και το πεδίο φδΣ = φυσική διεύθυνση - δμΣ = $34 - 11 = 23$ bits. Επίσης, το μέγεθος κάθε γραμμής του Α.Π.Σ., όπως προαναφέρθηκε είναι: μέγεθος μιας αντιστοίχισης x πλήθος αντιστοιχίσεων = $(7 + 54 + 23) \times 8 = 672$ bits. Επομένως, το μέγεθος Α.Π.Σ. = πλήθος γραμμών x μέγεθος κάθε γραμμής = $2^{\varphi\delta\Sigma}$ γρ. $\times \frac{672 \text{ bits}}{\text{γρ.}} = 2^{23} \times 672$ bits = $2^3 \times 2^{20} \times 672$ bits = 5376 Mbits \rightarrow 005376M.



Μέγεθος (χωρητικότητα) **Π.Σ.** = πλήθος γραμμών x μέγεθος κάθε γραμμής = $2^{\text{ΑΛΣ}}$ γρ. x μέγεθος κάθε γραμμής = 2^{30} γρ. x (7 bits ελέγχου + φδΣ) = 2^{30} γρ. x $25 \frac{\text{bits}}{\text{γρ.}} = 2^{30} \times 25 \text{ bits} = 2^{27}$ γρ. x $25 \frac{\text{bytes}}{\text{γρ.}} = 2^{20} \times 2^7 \times 25 \text{ bytes} = 2^7 \times 25 \text{ MB} = 3.125 \text{ GB}$

> 1GB. Άρα ο Π.Σ. δεν χωρά να αποθηκευτεί στην ΚΜ.

β. Έχουμε την **τεχνική του ΑΠΣ (Αντιστραμμένου Πίνακα Σελίδων)** με 4 αντιστοιχίσεις ανά γραμμή του πίνακα. Κάθε αντιστοιχίση αποτελεί το άθροισμα τριών πεδίων: **7 bits ελέγχου + φδΣ + ΑΛΣ** = $7 + 18 \text{ bits} + 30 \text{ bits} = 55 \text{ bits}$ και επειδή -όπως αναφέραμε- έχουμε 4 αντιστοιχίσεις ανά γραμμή του πίνακα, **το συνολικό μήκος (μέγεθος) κάθε γραμμής** είναι $55 \text{ bits} \times 4 = 220 \text{ bits}$. Το συνολικό μέγεθος του ΑΠΣ είναι: **πλήθος γραμμών x μέγεθος κάθε γραμμής = $2^{\phi\delta\Sigma}$ γρ. x μέγεθος κάθε γραμμής = 2^{18} γρ. x $(220 \frac{\text{bits}}{\text{γρ.}}) = 2^{18} \times 220 \text{ bits} = 2^{15} \times 220 \text{ bytes} = 2^{10} \times 2^5 \times 220 \text{ bytes} = 2^5 \times 220 \text{ KB}$**

Παράδειγμα 5.11 με ιδεατή μνήμη και ΑΠΣ

Θεωρήστε υπολογιστή με κύρια μνήμη χωρητικότητας 2 Γγηφιολέξεις και οργάνωση μίας ψηφιολέξης (byte) ανά θέση μνήμης, ιδεατή μνήμη που χρησιμοποιεί λογικές διευθύνσεις των 52 δυαδικών ψηφίων και μέγεθος σελίδας 4 Κγηφιολέξεις.

α. Υποθέστε ότι η ιδεατή μνήμη υλοποιείται με την τεχνική της σελιδοποίησης με πίνακα σελίδων ενός επιπέδου. Να υπολογίσετε το μέγεθος της κύριας μνήμη που απαιτείται για την αποθήκευση του πίνακα σελίδων.

Σημείωση: Για να βρούμε τη χωρητικότητα του Π.Σ. πρέπει να υπολογίσουμε πόσες θέσεις έχει και πόση χωρητικότητα έχει η κάθε θέση. Οπότε χωρητικότητα ΠΣ = θέσεις x μέγεθος θέσεων. Οι θέσεις του Π.Σ. είναι ο αριθμός των σελίδων επειδή έχουμε οργάνωση ενός επιπέδου. Το μέγεθος κάθε θέσης θα προκύψει από το φδΣ, το οποίο θα βρεθεί από $2^{31}/2^{12}$. Το δμΣ=12. Επειδή το μέγεθος σε είδος = μέγεθος πλαισίου σελίδας, το μέγεθος σελίδας καθορίζει το δμΣ.

β. Υποθέστε ότι η ιδεατή μνήμη υλοποιείται με την τεχνική του αντιστραμμένου πίνακα σελίδων με τρεις αντιστοιχίσεις ανά θέση του πίνακα και η σχετική διεύθυνση μέσα στον αντιστραμμένο πίνακα σελίδων, σδμΑΠΣ, δίνεται από τη συνάρτηση διασποράς σδμΑΠΣ_i = ΑΛΣ_i ⊕ ΑΛΣ_{i+20} για i από 0 έως και 18.

β.1 Να υπολογίσετε το μέγεθος της κύριας μνήμης που καταλαμβάνει ο αντιστραμμένος πίνακας σελίδων.

β.2 Θεωρήστε ότι τα περιεχόμενα του αντιστραμμένου πίνακα σελίδων είναι αυτά που φαίνονται στον Πίνακα 5.12 (στο δεκαεξαδικό) και ότι τέσσερις διαδοχικές από τις διευθύνσεις που παρήγαγε η ΚΜΕ είναι οι: EC5306FB0B345, 7C3B621CB2000, EE402EF9FE436, και DD30280C06347. Θεωρήστε ότι το περιεχόμενο του βασικού καταχωρητή πίνακα σελίδων είναι μηδέν. Περιγράψτε τι θα συμβεί.

Τρέχουσα κατάσταση του αντιστραμμένου πίνακα σελίδων (από τα πεδία ελέγχου δίνεται μόνο το π)

Διεύθυνση	π	ΑΛΣ1	φδΣ1	π	ΑΛΣ2	φδΣ2	π	ΑΛΣ3	φδΣ3
01DFC	1	E6472EF98E	235DA	0			0		
...
5DF04	1	741B029EB4	A0034	1	7C3B621CB2	B340D	1	FD3B6A0CB2	345CD
...
03E3B	1	EC5306FB0B	1532A	0			0		
...

Άνση

α. Λαμβάνοντας υπόψη ότι έχουμε λογικές διευθύνσεις των 52 δυαδικών ψηφίων, η κύρια μνήμη έχει οργάνωση μίας ψηφιολέξης (byte) ανά θέση μνήμης και ότι το μέγεθος σελίδας είναι 4 Κγηφιολέξεις, συμπεραίνουμε ότι ο χώρος λογικών διευθύνσεων αποτελείται από $2^{52}/2^{12} = 2^{40}$ σελίδες, επομένως **το ΑΛΣ αποτελείται από 40 δυαδικά ψηφία** και ο πίνακας σελίδων θα έχει 2^{40} θέσεις. Η κύρια μνήμη θα αποτελείται από 2 Γγηφιολέξεις / 4 Κγηφιολέξεις = $2 \times 2^{30} / 4 \times 2^{10} = 2^{31}/2^{12} = 2^{31-12} = 2^{19}$ πλαίσια σελίδων, άρα η φυσική διεύθυνση σελίδας αποτελείται από 19 δυαδικά ψηφία. Κάθε θέση του πίνακα σελίδων θα έχει $7 + 19 = 26$ δυαδικά ψηφία. Ο συνολικός χώρος της κύριας μνήμης που απαιτείται για την αποθήκευση του πίνακα σελίδων είναι 26×2^{40} δυαδικά ψηφία, δηλαδή 3.25×2^{10} Γγηφιολέξεις >> 2 Γγηφιολέξεις (κύρια μνήμη).

β.1 Η κύρια μνήμη αποτελείται από 2^{19} πλαίσια σελίδων. Ο ΑΠΣ έχει μία θέση για κάθε πλαίσιο σελίδας της κύριας μνήμης, επομένως θα αποτελείται από 2^{19} θέσεις. Ο χώρος λογικών διευθύνσεων αποτελείται από $2^{52}/2^{12}=2^{40}$ σελίδες, επομένως ο ΑΛΣ

αποτελείται από 40 δυαδικά ψηφία. Για κάθε αντιστοίχιση απαιτούνται $7+40+19$ δυαδικά ψηφία. Κάθε θέση περιέχει πληροφορία για τρεις αντιστοιχίσεις, άρα απαιτούνται $3 \times (7 + 40 + 19) = 198$ δυαδικά ψηφία ανά θέση του αντιστραμμένου πίνακα σελίδων. Ο συνολικός χώρος της κύριας μνήμης που καταλαμβάνεται από τον αντιστραμμένο πίνακα σελίδων είναι 198×2^{19} δυαδικά ψηφία δηλαδή 99×2^{20} δυαδικά ψηφία ή 12.375 ψηφιολέξεις.

β.2. Λαμβάνοντας υπόψη ότι το περιεχόμενο του βασικού καταχωρητή πίνακα σελίδων είναι μηδέν, συμπεραίνουμε ότι εφαρμόζοντας τη δοθείσα συνάρτηση διασποράς στο ΑΛΣ τμήμα κάθε διεύθυνσης την οποία παράγει ο επεξεργαστής, λαμβάνουμε τη φυσική διεύθυνση της θέσης του αντιστραμμένου πίνακα σελίδων, Πίνακας 5.13. Για παράδειγμα, αν θεωρήσουμε τη λογική διεύθυνση **EC5306FB0B**³⁴⁵, η συνάρτηση διασποράς $\text{σδμΑΠΣ}_i = \text{ΑΛΣ}_i \oplus \text{ΑΛΣ}_{i+20}$ για i από 0 έως και 18, εφαρμόζεται μόνο στο ΑΛΣ τμήμα της, που είναι το $\text{EC5306FB0B} = 1110\ 1100\ 0101\ 0011\ 0000\ 0110\ 1111\ 1011\ 0000\ 1011$ και δίνει ως αποτέλεσμα το εξής: για $i = 0$, έχουμε $\text{ΑΛΣ}_0 \oplus \text{ΑΛΣ}_{0+20} = 1 \oplus 0 = 1$, για $i = 1$ έχουμε $\text{ΑΛΣ}_1 \oplus \text{ΑΛΣ}_{1+20} = 1 \oplus 0 = 1$, για $i = 2$ έχουμε $\text{ΑΛΣ}_2 \oplus \text{ΑΛΣ}_{2+20} = 0 \oplus 0 = 0$, για $i = 3$ έχουμε $\text{ΑΛΣ}_3 \oplus \text{ΑΛΣ}_{3+20} = 1 \oplus 0 = 0$ κ.ο.κ. οπότε στο τέλος έχουμε: $\text{EC5306FB0B} \rightarrow 03E3B$

Πίνακας 5.13

ΑΛΣ	Έξοδος της συνάρτησης διασποράς, δηλαδή φυσική διεύθυνση θέσης στον ΑΠΣ
EC5306FB0B	03E3B
7C3B621CB2	5DF04
EE402EF9FE	01DFC
DD30280C06	5DF04

Από τον αριθμό λογικής σελίδας, ΑΛΣ, EC5306FB0B με την εφαρμογή της συνάρτησης διασποράς παίρνουμε τη διεύθυνση 03E3B. Στη θέση του αντιστραμμένου πίνακα σελίδων με διεύθυνση 03E3B. Στη θέση του αντιστραμμένου πίνακα σελίδων με διεύθυνση 03E3B υπάρχει η αντιστοίχιση ΑΛΣ σε φυσική διεύθυνση σελίδας, EC5306FB0B \rightarrow 1532A με δυαδικό ψηφίο παρουσίας $\pi = 1$. Επομένως η φυσική διεύθυνση σελίδας 1532A θα χρησιμοποιηθεί για την προσπέλαση της πληροφορίας.

Από τον αριθμό λογικής σελίδας, ΑΛΣ, 7C3B621CB2 με την εφαρμογή της συνάρτησης διασποράς παίρνουμε τη διεύθυνση 5DF04. Στη θέση του αντιστραμμένου πίνακα σελίδων με διεύθυνση 5DF04 υπάρχει η αντιστοίχιση ΑΛΣ σε φυσική διεύθυνση σελίδας 7C3B621CB2 \rightarrow B340D με δυαδικό ψηφίο παρουσίας $\pi = 1$. Επομένως η φυσική διεύθυνση σελίδας B340D θα χρησιμοποιηθεί για την προσπέλαση της πληροφορίας.

Από τον αριθμό λογικής σελίδας, ΑΛΣ, EE402EF9FE με την εφαρμογή της συνάρτησης διασποράς παίρνουμε τη διεύθυνση 01DFC. Στη θέση του ΑΠΣ με διεύθυνση 01DFC δεν υπάρχει η αντιστοίχιση του ζητούμενου ΑΛΣ σε φυσική διεύθυνση σελίδας. Το λειτουργικό σύστημα θα μεταφέρει τη ζητούμενη σελίδα στην κύρια μνήμη και θα ενημερώσει κατάλληλα τον αντιστραμμένο πίνακα σελίδων. Στον πίνακα 5.14 δίνεται το περιεχόμενο του ΑΠΣ μετά την ενημέρωση.

Πίνακας 5.14

Διεύθυνση	Π	ΑΛΣ1	φδΣ1	Π	ΑΛΣ2	φδΣ2	Π	ΑΛΣ3	φδΣ3
01DFC	1	E6472EF98E	235DA	1	ΕΕ402EF9FE	EED12	0		
...
5DF04	1	741B029EB4	A0034	1	7C3B621CB2	B340D	1	DD30280C06	111CD
...
03E3B	1	EC5306FB0B	1532A	0			0		
...

Άσκηση 5.18 – Ιδεατή μνήμη με ΑΠΣ

Θεωρείστε υπολογιστή με κύρια μνήμη χωρητικότητας 16 Μψηφιολέξεις (16 Mbytes) και οργάνωση μίας ψηφιολέξης (byte) ανά θέση μνήμης, ιδεατή μνήμη που χρησιμοποιεί λογικές διευθύνσεις των 36 δυαδικών ψηφίων και μέγεθος σελίδας 4 Κψηφιολέξεις. α. Υποθέστε ότι η ιδεατή μνήμη υλοποιείται με την τεχνική της σελιδοποίησης με πίνακα σελίδων ενός επιπέδου. Να υπολογίσετε το μέγεθος της κύριας μνήμης που απαιτείται για την αποθήκευση του πίνακα σελίδων.

β. Υποθέστε ότι η ιδεατή μνήμη υλοποιείται με την τεχνική του αντιστραμμένου πίνακα σελίδων (inverted page table) με δύο αντιστοιχίσεις ανά θέση του πίνακα. Η σχετική διεύθυνση μέσα στον αντιστραμμένο πίνακα σελίδων, σδμΑΠΣ, δίνεται από τη

συνάρτηση διασποράς (hash function) $\text{σδμΑΠΣ}_i = \text{ΑΛΣ}_i + \text{ΑΛΣ}_{i+4} \bmod 8$, για i από 1 έως και 4. Το ΑΛΣ_i είναι το i ψηφίο, στο οκταδικό, του Αριθμού Λογικής Σελίδας, αρχίζοντας τη μέτρηση από το λιγότερο σημαντικό οκταδικό ψηφίο.

i. Να υπολογίσετε το μέγεθος της κύριας μνήμης που καταλαμβάνει ο αντιστραμμένος πίνακας σελίδων.

ii Θεωρήστε ότι αρχικά ο αντιστραμμένος πίνακας σελίδων είναι κενός και δώστε το περιεχόμενό του μετά την παραγωγή από τον επεξεργαστή της επόμενης ακολουθίας λογικών διευθύνσεων στο οκταδικό: 156032010000, 370667171234, 043732174552 και 320115600022. Θεωρήστε ότι αυτές οι λογικές διευθύνσεις, μετά την προσκόμιση των απαιτούμενων σελίδων στην κύρια μνήμη, αντιστοιχούν στις φυσικές διευθύνσεις: 17620000, 63551234, 46004552 και 06650022.

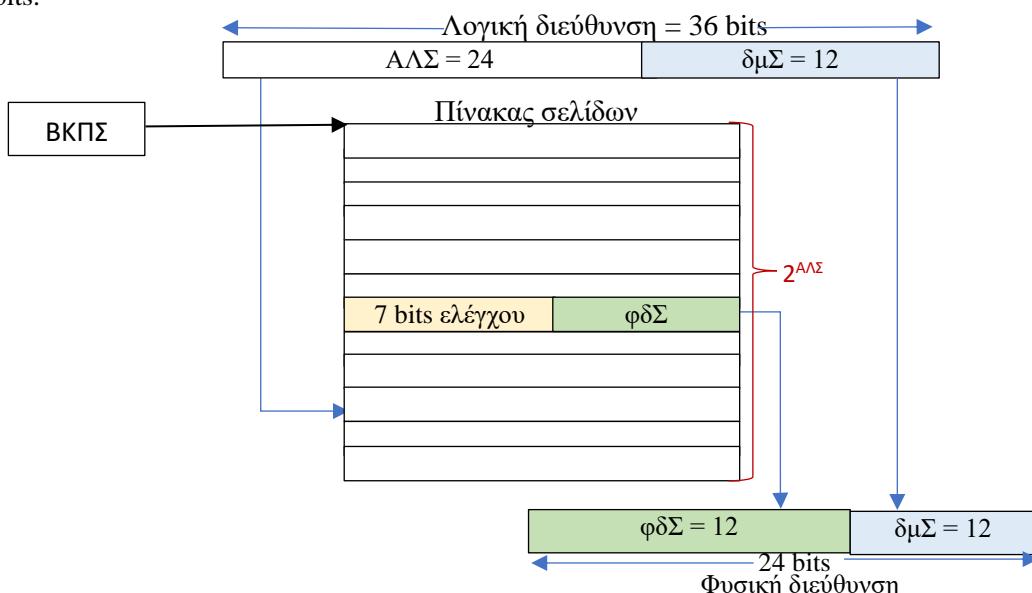
Θεωρήστε ότι το περιεχόμενο του βασικού καταχωρητή πίνακα σελίδων είναι μηδέν.

Δεδομένα:

- KM: $16 \text{ MB} = 2^4 \times 2^{20} \text{ bytes} = 2^{24} \text{ bytes} = (1 \text{ ψηφιολέξη}/\theta.\mu.) 2^{24} \theta.\mu.$ → Μήκος φυσικής διεύθυνσης = 24 bits
 - Μήκος λογικής διεύθυνσης = 36 bits (πάντα δίνεται)
 - Μέγεθος σελίδων = μέγεθος πλαισίων σελίδων = 4 KB

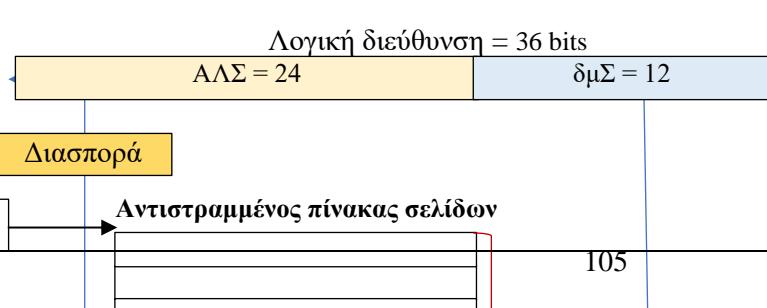
Αύση

Οι σελίδες βρίσκονται στην ιδεατή μνήμη και τα πλαίσια σελίδων βρίσκονται στην κύρια μνήμη. Ισχύει ότι το πλήθος σελίδων > πλήθος πλαισίων σελίδων. Πάντα από το μέγεθος σελίδων = μέγεθος πλαισίων σελίδων υπολογίζουμε το **δμΣ** (**διεύθυνση μέσα στη Σελίδα**). Δηλαδή $4 \text{ KB} = 2^2 \times 2^{10} \text{ bytes} = 2^{12} \text{ bytes} = (\lambda\circ\gamma\omega \text{ οργάνωσης } 1 \text{ ψηφιολέξη}/\theta.\mu.) 2^{12} \theta.\mu. \rightarrow \delta\mu\Sigma = 12 \text{ bits}$ (είναι πάντα ο εκθέτης του μεγέθους των σελίδων). Επομένως το πεδίο **ΑΛΣ** = λογική διεύθυνση – $\delta\mu\Sigma = 36 \text{ bits} - 12 \text{ bits} = 24 \text{ bits}$.

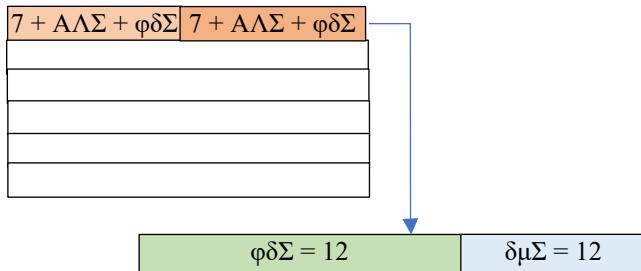


Πάντα το πεδίο ΑΛΣ υπολογίζεται με αφαίρεση. Επίσης, το πεδίο φδΣ (φυσική διεύθυνση μέσα στη σελίδα) πάντα το υπολογίζουμε αφαιρώντας από το συνολικό μήκος της KM που είναι 24 bits, το δμΣ που είναι 12 bits και έχουμε: $\phi\delta\Sigma = \text{φυσική διεύθυνση} - \delta\Sigma = 24 - 12 = 12$ bits. Για να υπολογίσουμε το **μέγεθος του Π.Σ.** (Πίνακα Σελίδων) πολ/ζουμε το πλήθος των γραμμών του με το μέγεθος κάθε γραμμής του. Καταρχήν, το **μέγεθος κάθε γραμμής του Π.Σ.** = 7 bits ελέγχου + φδΣ = $7 + 12 = 19$ bits. Άρα **μέγεθος Π.Σ. = πλήθος γραμμών x μέγεθος κάθε γραμμής = $2^{\text{ΑΛΣ}}$ γρ. x μέγεθος κάθε γραμμής = 2^{24} γραμμές x $19 \frac{\text{bits}}{\text{γραμμή}}$ = $2^4 \times 2^{20} \times 19 \text{ bits} = 2 \times 2^{20} \times 19 \text{ bytes} = 38 \text{ MB} > 16 \text{ MB(KM)}$** → ο Π.Σ. δεν χωρά να αποθηκευτεί στην κύρια μνήμη.

b.



$2^{\phi\delta\Sigma}$



i. Για να υπολογίσουμε το μέγεθος του ΑΠΣ Φυσική διεύθυνση = 24 bits θα πρέπει απλαστά σύμμετρο το πλήθος των γραμμών του με το μέγεθος κάθε γραμμής. Κάθε γραμμή υποδιαιρείται σε μια σειρά αντιστοιχίσεων, όπου κάθε αντιστοιχίση είναι το άθροισμα τριών πεδίων ($7 + \text{AL}\Sigma + \phi\delta\Sigma$). Επομένως: μέγεθος Α.Π.Σ. = πλήθος γραμμών x μέγεθος κάθε γραμμής = $2^{\phi\delta\Sigma}$ γρ. x **2 x μέγεθος αντιστοιχισης** = 2^{12} γρ. x **2 x (7 + 12 + 24) bits/γρ.** = $2^{12} \times 86 \text{ bits} = 2^9 \times 86 \text{ bytes} = 2^9 \times 2 \times 43 \text{ bytes} = 2^{10} \times 43 \text{ bytes} = 43 \text{ KB}$.

ii. Αρχικά ο ΑΠΣ είναι **κενός**, όπως φαίνεται στη συνέχεια:

Φυσική διεύθυνση	π	$\text{AL}\Sigma 1$	$\phi\delta\Sigma 1$	π	$\text{AL}\Sigma 2$	$\phi\delta\Sigma 2$

Ο επεξεργαστής παράγει την επόμενη ακολουθία λογικών διεύθυνσεων, οι οποίες μεταφράζονται στις αντίστοιχες φυσικές διεύθυνσεις που φαίνονται στη συνέχεια.

Λογική διεύθυνση **Φυσική διεύθυνση**

156032010000	17620000
370667171234	63551234
043732174552	46004552
320115600022	06650022

Η **συνάρτηση διασποράς** εφαρμόζεται στο ΑΛΣ τμήμα της κάθε λογικής διεύθυνσης και είναι -σύμφωνα με την εκφώνηση- η εξής: $\text{σδμΑΠΣ} = (\text{ΑΛΣ}_i + \text{ΑΛΣ}_{i+4}) \text{ mod } 8$, για $i = 1, 2, 3, 4$, όπου το i αναφέρεται στο λιγότερο σημαντικό οκταδικό ψηφίου του ΑΛΣ τμήματος.

15603201 → $i=1, (\text{ΑΛΣ}_1 + \text{ΑΛΣ}_5) \text{ mod } 8 = (1 + 0) \text{ mod } 8 = 1$
 $i=2, (\text{ΑΛΣ}_2 + \text{ΑΛΣ}_6) \text{ mod } 8 = (0 + 6) \text{ mod } 8 = 6$
 $i=3, (\text{ΑΛΣ}_3 + \text{ΑΛΣ}_7) \text{ mod } 8 = (2 + 5) \text{ mod } 8 = 7$
 $i=4, (\text{ΑΛΣ}_4 + \text{ΑΛΣ}_8) \text{ mod } 8 = (3 + 1) \text{ mod } 8 = 4$

37066717 → $i=1, (\text{ΑΛΣ}_1 + \text{ΑΛΣ}_5) \text{ mod } 8 = (7 + 6) \text{ mod } 8 = 5$
 $i=2, (\text{ΑΛΣ}_2 + \text{ΑΛΣ}_6) \text{ mod } 8 = (1 + 0) \text{ mod } 8 = 1$
 $i=3, (\text{ΑΛΣ}_3 + \text{ΑΛΣ}_7) \text{ mod } 8 = (7 + 7) \text{ mod } 8 = 6$
 $i=4, (\text{ΑΛΣ}_4 + \text{ΑΛΣ}_8) \text{ mod } 8 = (6 + 3) \text{ mod } 8 = 1$

04373217 → $i=1, (\text{ΑΛΣ}_1 + \text{ΑΛΣ}_5) \text{ mod } 8 = (7 + 7) \text{ mod } 8 = 6$
 $i=2, (\text{ΑΛΣ}_2 + \text{ΑΛΣ}_6) \text{ mod } 8 = (1 + 3) \text{ mod } 8 = 4$
 $i=3, (\text{ΑΛΣ}_3 + \text{ΑΛΣ}_7) \text{ mod } 8 = (2 + 4) \text{ mod } 8 = 6$
 $i=4, (\text{ΑΛΣ}_4 + \text{ΑΛΣ}_8) \text{ mod } 8 = (3 + 0) \text{ mod } 8 = 3$

32011560 → $i=1, (\text{ΑΛΣ}_1 + \text{ΑΛΣ}_5) \text{ mod } 8 = (1 + 0) \text{ mod } 8 = 1$
 $i=2, (\text{ΑΛΣ}_2 + \text{ΑΛΣ}_6) \text{ mod } 8 = (6 + 0) \text{ mod } 8 = 6$
 $i=3, (\text{ΑΛΣ}_3 + \text{ΑΛΣ}_7) \text{ mod } 8 = (5 + 2) \text{ mod } 8 = 7$
 $i=4, (\text{ΑΛΣ}_4 + \text{ΑΛΣ}_8) \text{ mod } 8 = (1 + 3) \text{ mod } 8 = 4$

Στη συνέχεια τοποθετούμε τα αποτελέσματα της συνάρτησης διασποράς μέσα στον ΑΠΣ με σειρά αύξουσα:

Φυσική διεύθυνση	π	$\text{AL}\Sigma 1$	$\phi\delta\Sigma 1$	π	$\text{AL}\Sigma 2$	$\phi\delta\Sigma 2$

.....	0			0		
.....	0			0		
1615	1	37066717	6355	0		
.....	0			0		
.....	0			0		
3646	1	04373217	4600	0		
.....	0			0		
.....	0			0		
4761	1	15603201	1762	1	32011560	0665
.....	0			0		

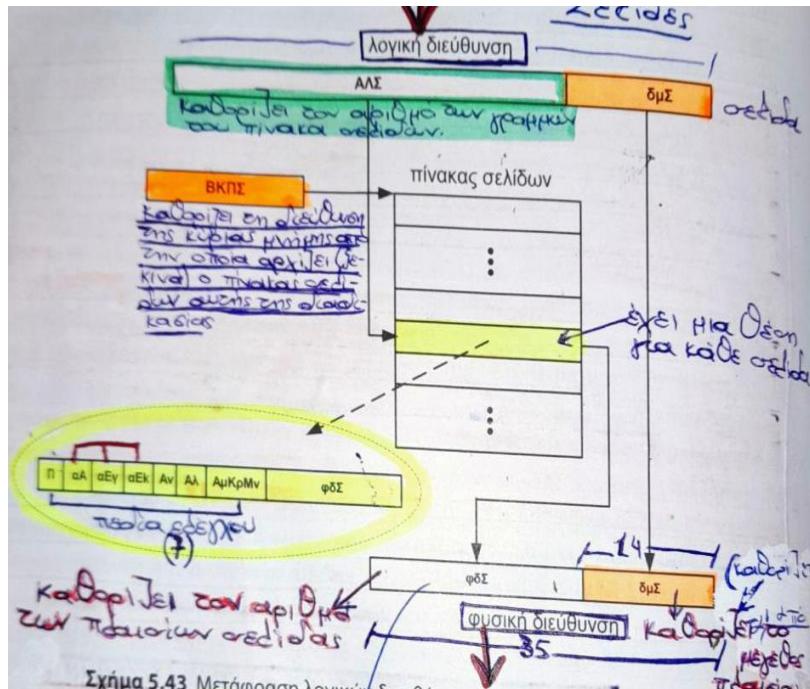
Θέμα Ιανουαρίου 2020 – Ιδεατή μνήμη με ΑΠΣ και τεχνική πολλών επιπέδων

Θεωρείστε υπολογιστή με κύρια μνήμη 16 GBytes και οργάνωση ενός byte ανά θέση μνήμης. Η ιδεατή μνήμη χρησιμοποιεί λογικές διεύθυνσεις των 64 δυαδικών ψηφίων και μέγεθος σελίδας 8 KBytes.

1. Να υπολογίσετε το χώρο που καταλαμβάνει ο Π.Σ. στην κύρια μνήμη όταν υλοποιηθεί με τον κλασσικό τρόπο **σε ένα επίπεδο**.
2. Να υπολογίσετε το χώρο που καταλαμβάνει ο πίνακας σελίδων στην κύρια μνήμη όταν για την υλοποίηση του πίνακα σελίδων έχει χρησιμοποιηθεί η τεχνική του **αντιστραμμένου πίνακα σελίδων** με 2 αντιστοιχίσεις ανά θέση.

Υπενθύμιση: Το πλήθος των δυαδικών ψηφίων ελέγχου είναι 7.

Λύση



1. Σε αυτήν την περίπτωση το **μέγεθος του πίνακα σελίδων** προκύπτει από το γινόμενο του πλήθους των γραμμών x το μέγεθος κάθε γραμμής. Από τα δεδομένα της εκφώνησης έχουμε το μέγεθος σελίδας 8 KBytes = $2^3 \times 2^{10}$ bytes = 2^{13} bytes = 2^{13} θ.μ. Επομένως το **δμΣ = 13 bits**. Από τη **χωρητικότητα της κύριας μνήμης** θα βρεθεί το μέγεθος της φυσικής διεύθυνσης που είναι: 16 GBytes = $2^4 \times 2^{30}$ bytes = 2^{34} bytes = 2^{34} θ.μ. **Άρα, το μέγεθος της φυσικής διεύθυνσης είναι 34 bits**.

Η λογική και η φυσική διεύθυνση είναι της μορφής:

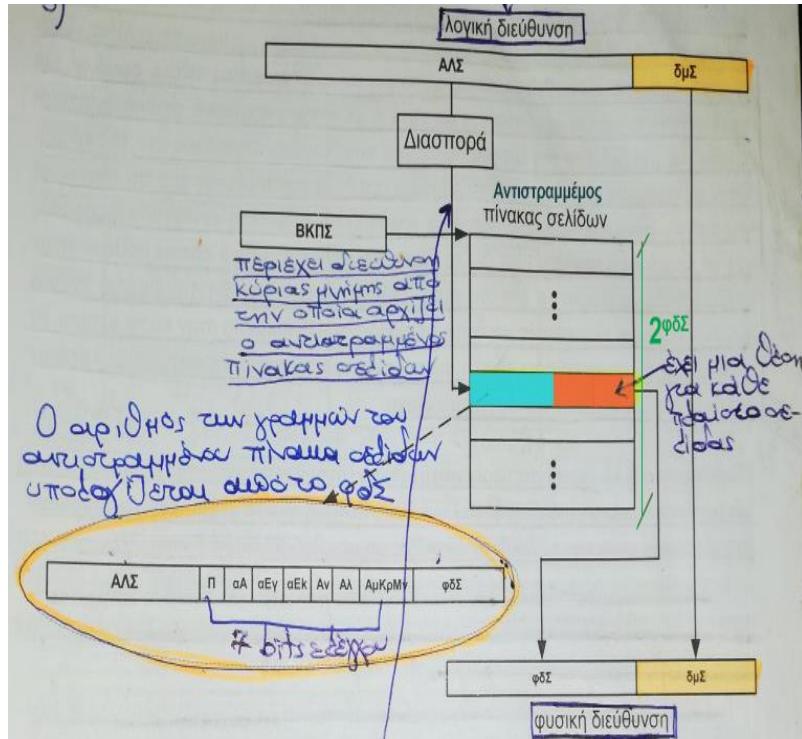
$$\text{Λογική διεύθυνση} = 64 \quad / \quad \text{Φυσική διεύθυνση} = 34$$

ΑΛΣ = 51	δμΣ = 13	φδΣ = 21	δμΣ = 13
----------	----------	----------	----------

Επομένως το μέγεθος του Π.Σ. = πλήθος γραμμών x μέγεθος κάθε γραμμής = $2^{\text{ΑΛΣ}}$ γραμμές x μέγεθος κάθε γραμμής = 2^{51} γραμμές x (7 bits ελέγχου + φδΣ) = 2^{51} γραμμές x (7 bits ελέγχου + 21 bits)/γρ. = 2^{51} γραμμές x $28 \frac{\text{bits}}{\text{γρ.}}$ = $2^{51} \times 28$ bits = $2^{48} \times 28$ bytes = $2^{30} \times 2^{18} \times 28$ bytes = $2^{18} \times 28$ GBytes >> 16 GBytes. Επομένως ο Π.Σ. δεν χωρά να αποθηκευτεί στην Κ.Μ.

2. Στην περίπτωση του **Α.Π.Σ. (Αντιστραμμένου Πίνακα Σελίδων)** γνωρίζουμε ότι για να υπολογίσουμε το μέγεθος του θα χρησιμοποιήσουμε τον τύπο: Μέγεθος Α.Π.Σ. = πλήθος γραμμών x μέγεθος κάθε γραμμής = $2^{\phi\delta\Sigma}$ γραμμές x μέγεθος κάθε γραμμής = $2^{\phi\delta\Sigma}$ γραμμές x (7 bits ελέγχου + ΑΛΣ + φδΣ) x 2 αντιστοιχίσεις = 2^{21} γραμμές x (7 bits ελέγχου + 51 + 21) x 2

$$\frac{\text{bits}}{\text{γραμμή}} = 2^{21} \text{ γραμμές} \times 79 \times 2 \frac{\text{bits}}{\text{γραμμή}} = 2^{21} \times 158 \text{ bits} = 2^{18} \times 158 \text{ bytes} = 2^{10} \times 2^8 \times 158 \text{ bytes} = 2^8 \times 158 \text{ Kbytes.}$$



Θέμα προόδου Μαΐου 2022 με ΑΠΣ

Θεωρήστε υπολογιστή με κύρια μνήμη χωρητικότητα 16 Mbytes και οργάνωση μίας ψηφιολέξης (byte) ανά θέση μνήμης, ιδεατή μνήμη που χρησιμοποιεί λογικές διευθύνσεις των 32 δυαδικών ψηφίων και μέγεθος σελίδας 256 Bytes. Υποθέστε ότι η ιδεατή μνήμη υλοποιείται με την **τεχνική του αντεστραμμένου πίνακα σελίδων με τρεις αντιστοιχίσεις ανά θέση του πίνακα** και η σχετική διεύθυνση μέσα στον αντεστραμμένο πίνακα σελίδων, σδμΑΠΣ, δίνεται από τη συνάρτηση διασποράς **σδμΑΠΣ = υπόλοιπο της διαιρέσης του ΑΛΣ δια του 2^{16}** . Θεωρήστε ότι τα περιεχόμενα του αντεστραμμένου πίνακα σελίδων είναι αντά που φαίνονται στον πίνακα (στο δεκαεξαδικό) και το περιεχόμενο του βασικού καταχωρητή πίνακα σελίδων είναι μηδέν. Θεωρήστε ότι ο επεξεργαστής παράγει τη λογική διεύθυνση **0E7CDFC3**. Να βρεθεί η αντίστοιχη φυσική διεύθυνση που αντιστοιχεί στην ανωτέρω λογική διεύθυνση.

Πίνακας

Τρέχουσα κατάσταση του αντεστραμμένου πίνακα σελίδων (από τα πεδία ελέγχου δίνουμε την τιμή μόνο του πεδίου Παρουσίας Π της αντιστοιχίσης). Δεν δίνεται το περιεχόμενο όλων των θέσεων του Πίνακα

Διεύθυνση	Π	ΑΛΣ1	φδΣ1	Π	ΑΛΣ2	φδΣ2	Π	ΑΛΣ3	φδΣ3
0494	1	130494	E3A9	1	BF0494	8101	0	BF0494	342A
16EA	0	1716EA	7E7A	1	6116EA	290D	1	1716EA	7EF6
2563	1	EF2563	0EE3	1	FE2563	4309	0	232563	3E45
2CFE	1	AB2CFE	5AEF	1	542CFE	EA46	1	592CFE	5AEF
68DF	1	1568DF	CB00	1	8968DF	CB00	0	5768DF	0945
7077	1	007077	23EA	1	057077	B697	0	787077	B698
78CC	1	3278CC	465B	0	2378CC	AD45	1	AC78CC	8923
7CDF	0	237CDF	3543	1	247CDF	A342	1	0E7CDF	9C44
8C55	0	A48C55	57A3	1	D48C55	2443	1	A38C55	45B2
8E8E	1	D08E8E	D6A1	1	988E8E	1987	1	AD8E8E	AD45
9F45	1	279F45	3498	1	859F45	CBEE	0	399F45	27AC
B72A	1	99B72A	DF64	1	45B72A	035B	1	ADB72A	AD45
BF04	1	94BF04	AE45	0	23BF04	4512	1	49BF04	F567
C508	0	00C508	2300	1	11C508	4311	1	10C508	9DFE
D08E	1	A3D08E	B546	1	8ED08E	2567	1	56D08E	8909
D364	1	66D364	69FB	0	96D364	AD45	1	A4D364	C678
EA0D	1	1AEA0D	9EEE	1	1BEA0D	6578	1	A4EA0D	6543

Άνση

Η κύρια μνήμη έχει χωρητικότητα $16 \text{ Mbytes} = 2^4 \times 2^{20} \text{ bytes}$ (ψηφιολέξεις) $= 2^{24}$ ψηφιολέξεις $= 2^{24} \theta.\mu.$ λόγω της οργάνωσης μίας ψηφιολέξης (byte) ανά θέση μνήμης. Αυτό σημαίνει ότι το μήκος της φυσικής διεύθυνσης είναι 24 bits. Στη συνέχεια, από το μέγεθος σελίδας = μέγεθος πλαισίου σελίδας $= 256 \text{ bytes} = 2^8 \theta.\mu.$ προκύπτει ότι $\delta\mu\Sigma = 8$. Επομένως,:

$$\text{Λογική διεύθυνση} = 32 \text{ bits}$$

$$\text{ΑΛΣ} = 24$$

$$\delta\mu\Sigma = 8$$

$$\text{Φυσική διεύθυνση} = 24 \text{ bits}$$

$$\phi\delta\Sigma = 16$$

$$\delta\mu\Sigma = 8$$

Ο επεξεργαστής παράγει τη λογική διεύθυνση **0E7CDFC3**, η οποία αποτελείται από το ΑΛΣ τμήμα της μεγέθους 24 bits = 6 δεκαεξαδικά ψηφία και το δμΣ τμήμα μεγέθους = 8 bits, δηλ. **0E7CDF | C3**. Το ΑΛΣ τμήμα (**0E7CDF**) της διεύθυνσης αυτής υπάρχει μέσα στο δοθέντα ΑΠΣ στην 3^η αντιστοίχιση της γραμμής με φυσική διεύθυνση 7CDF, και το φδΣ που αντιστοιχεί σε αυτό είναι το **9C44**. Παίρνοντας αυτό το φδΣ τμήμα και ενώνοντάς το με το πεδίο δμΣ, που είναι κοινό πεδίο και στις δύο διεύθυνσεις, έχουμε την τελική φυσική διεύθυνση: **9C44 | C3 → 9C44C3**.

Άλλος τρόπος: Εφαρμόζοντας τη συνάρτηση διασποράς **σδμΑΠΣ = υπόλοιπο της διαιρέσης του ΑΛΣ δια του 2^{16}** στο ΑΛΣ τμήμα της λογικής διεύθυνσης **0E7CDF | C3**, θα έχουμε: **0E7CDF** = 0000 1110 0111 1100 1101 11112 = 94947110 και στη συνέχεια $94947110 \bmod 2^{16} = 3196710 = 7CDF$ που είναι η διεύθυνση γραμμής του ΑΠΣ όπου και ελέγχουμε αν σε οποιαδήποτε από τις τρεις αντιστοιχίσεις υπάρχει το ζητούμενο ΑΛΣ: **0E7CDF**. Αυτό υπάρχει στην 3^η αντιστοίχιση της συγκεκριμένης γραμμής, και από εκεί παίρνουμε το φδΣ = 9C44, το οποίο ενώνουμε στη συνέχεια με το πεδίο δμΣ για να δημιουργήσουμε τη φυσική διεύθυνση **9C44C3**.



Παρατήρηση: Σε περίπτωση που ζητηθεί η διεύθυνση **45B72A | 64**, το ΑΛΣ τμήμα της διεύθυνσης αυτής υπάρχει μέσα στο δοθέντα ΑΠΣ στην 2^η αντιστοίχιση της γραμμής με φυσική διεύθυνση B72A, και το φδΣ που αντιστοιχεί σε αυτό είναι το 035B. Παίρνοντας αυτό το φδΣ τμήμα και ενώνοντάς το με το πεδίο δμΣ, που είναι κοινό πεδίο και στις δύο διευθύνσεις, έχουμε την τελική φυσική διεύθυνση: 035B | **64 → 035B64**.

Εναλλακτικά, εφαρμόζοντας τη συνάρτηση διασποράς σδμΑΠΣ = υπόλοιπο της διαίρεσης του ΑΛΣ δια του 2^{16} στο ΑΛΣ τμήμα της λογικής διεύθυνσης **45B72A | 64**, θα έχουμε: **45B72A** = 0010 0101 1011 0111 0010 1010 = 4568874 και στη συνέχεια $94947110 \bmod 2^{16} = 46890 = B72A$ που υπάρχει στην 2^η αντιστοίχιση της συγκεκριμένης γραμμής, και από εκεί παίρνουμε το φδΣ = 035B και στο τέλος έχουμε 035B | **64 = 035β64**.



Θέμα Ιουνίου 2016 – Ιδεατή μνήμη με χρήση ΑΠΜ

Θεωρείστε υπολογιστή με κύρια μνήμη χωρητικότητας 4 Γγηφιολέξεις (1 γγηφιολέξη = 1 byte) και οργάνωση μιας γγηφιολέξης ανά θέση μνήμης, ιδεατή μνήμη που χρησιμοποιεί διευθύνσεις των 54 δυαδικών γγηφίων και μέγεθος σελίδας 8 KBytes. Α. Εάν η ιδεατή μνήμη υλοποιείται με πίνακα σελίδων ενός επιπέδου να υπολογίσετε το μέγεθος της κύριας μνήμης που απαιτείται για την αποθήκευση του πίνακα σελίδων.

β. Εάν η ιδεατή μνήμη υλοποιείται με την τεχνική του αντιστραμμένου πίνακα σελίδων (inverted page table) με 5 αντιστοιχίσεις ανά θέση του πίνακα να υπολογίσετε το μέγεθος της κύριας μνήμης που καταλαμβάνει ο αντιστραμμένος πίνακας σελίδων. Θεωρήστε ότι τα πεδία ελέγχου καταλαμβάνουν 7 δυαδικά ψηφία.

Αύση

Η κύρια μνήμη έχει χωρητικότητα $4 \text{ Γψηφιολέξεις} = 2^2 \times 2^{30}$ ψηφιολέξεις $= 2^{32}$ ψηφιολέξεις, άρα η φυσική διεύθυνση είναι των 32 δυαδικών ψηφίων. Η ιδεατή μνήμη χρησιμοποιεί διευθύνσεις των 54 δυαδικών ψηφίων άρα η λογική διεύθυνση είναι των 54 δυαδικών ψηφίων. Αφού το μέγεθος σελίδας 8 Κψηφιολέξεις $= 2^3 \times 2^{10}$ ψηφιολέξεις $= 2^{13}$ ψηφιολέξεις, δημΣ: 13 δυαδικά ψηφία, φδΣ: $32-13=19$ δυαδικά ψηφία και ΑΛΣ: $54-13=41$ δυαδικά ψηφία.

α. Χωρητικότητα πίνακα σελίδων = $(\text{δυαδικά ψηφία ελέγχου} + \phi\delta\Sigma)x(\text{πλήθος λογικών σελίδων}) = (7 \text{ δυαδικά ψηφία} + 19 \text{ δυαδικά ψηφία})x^{241} = 26x^{241} \text{ δυαδικά ψηφία} = 13x2x4x2^{39} \text{ δυαδικά ψηφία} = 13x2^{39} \text{ ψηφιολέξεις} = 13x2^9 \text{ Γψηφιολέξεις.}$

β. Χωρητικότητα πίνακα σελίδων = (πλήθος αντιστοιχίσεων ανά θέση του πίνακα) x (δυαδικά ψηφία ελέγχου + ΑΛΣ + φδΣ) x (πλήθος φυσικών σελίδων) = 5 x (7 δυαδικά ψηφία + 41 δυαδικά ψηφία + 19 δυαδικά ψηφία)x2¹⁹ = 5x67x2¹⁹ δυαδικά ψηφία = 335x2¹⁹ δυαδικά ψηφία = 335 x 2¹⁶ ψηφιολέξεις = 335 x 2⁶ Κψηφιολέξεις.

Θέμα Σεπτεμβρίου 2017 – Ιδεατή μνήμη με χρήση κρυφής μνήμης πίνακα σελίδων (TLB)

Θεωρείστε υπολογιστή με ιδεατή μνήμη που υλοποιείται με την τεχνική της σελιδοποίησης και έχει **TLB 4 – τρόπων συνόλου συσχέτισης** με 32 σύνολα. Η λογική διεύθυνση αποτελείται από 52 bits και φυσική διεύθυνση που αποτελείται από 32 bits και μέγεθος σελίδας 4 KB με οργάνωση 1 byte/θ.μ. Να σχεδιαστεί η TLB και να βρεθεί η χωρητικότητά της

Αύση

Όταν έχουμε οργάνωση πλήρους συσχέτισης στην TLB τότε τα πεδία της είναι:

Στην οργάνωση πλήρους συσχέτισης, η στήλη με το ΑΛΣ αντιστοιχεί στο πεδίο «ν» της διεύθυνσης της κρυφής μνήμης.

v μ

Τώρα όμως που έχουμε οργάνωση 4 – τρόπων συνόλου συσχέτισης, στη θέση του ΑΛΣ βάζουμε μόνο το πεδίο «ν - λ» που είναι 35 bits. Το μέγεθος σελίδας = 4 KB = $2^2 \times 2^{10} = 2^{12}$ (λόγω της οργάνωσης 1 byte/θ.μ.) = 2^{12} θ.μ. $\rightarrow \delta\mu\Sigma = 12$ bits. Άρα το $\Delta\Sigma = 52 - 12 = 40$ bits και το $\omega\delta\Sigma = 32 - 12 = 20$ bits.

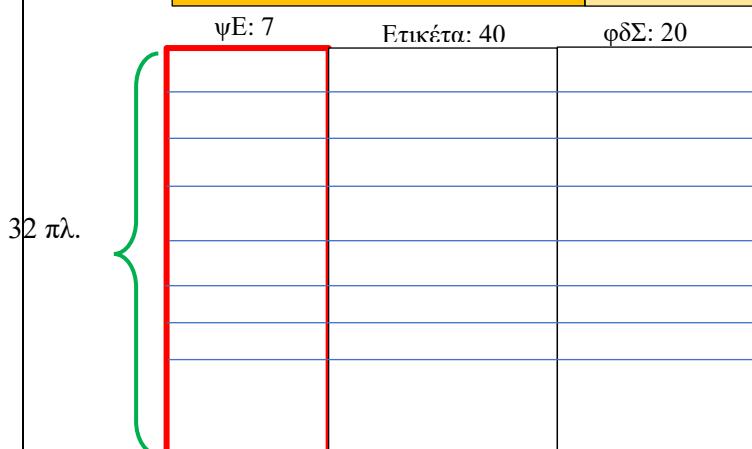
Η λογική και η φυσική διεύθυνση είναι της μορφής:



Στην TLB που είναι 4 – τρόπων δεν χρησιμοποιούμε το πεδίο δμΠ (μ) για πολυπλέκτη, οπότε $\mu = 0$ και το $v - \lambda = \text{ΑΛΣ} - \lambda = 40 - 5 = 35$ bits και αυτή χρησιμοποιούμε αντί για το ΑΛΣ στην 3^η στήλη της TLB. Έτσι και αλλιώς το πεδίο δμΣ της λογικής διεύθυνσης χρησιμοποιείται για την προσπέλαση της κρυφής μνήμης επεξεργαστή, οπότε επειδή έχουμε προσπέλαση και των δύο κρυφών μνημάτων με λογικές διεύθυνσεις, **το ΑΛΣ χρησιμοποιείται για την προσπέλαση της κρυφής μνήμης Πίνακα Σελίδων (TLB)** και πιο συγκεκριμένα το πεδίο ετικέτας « $v - \lambda$ », διότι το πεδίο « λ » χρησιμοποιείται για την επιλογή του συνόλου. Επειδή έχουμε 32 σύνολα, ισχύει ότι $2^\lambda = 32 \Rightarrow \lambda = 5$ bits και το $v - \lambda = \text{ΑΛΣ} - \lambda = 40$ bits - 5 bits = 35 bits.

Σημείωση: Στην περίπτωση της **πλήρους συσχέτισης**, επειδή υπάρχει μόνο το πεδίο « v » ως ετικέτα, χρησιμοποιείται εξ' ολοκλήρου το ΑΛΣ για την προσπέλαση της TLB.

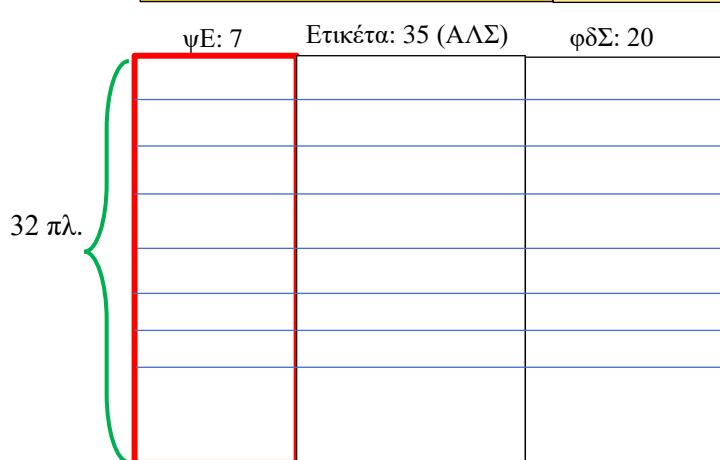
Παρατήρηση 1: Αν η TLB περιείχε 32 πλαίσια (αντί για σύνολα) και ήταν **πλήρους συσχέτισης** τότε θα ήταν της μορφής:



Τώρα **δεν** υπάρχει το πεδίο « v » ή « λ » οπότε η ετικέτα έχει το μήκος του ΑΛΣ.

Χωρητικότητα TLB = $32 \text{ πλ.} \times (7 + 40 + 20) \frac{\text{bits}}{\text{πλ.}} = 32 \times 67 \text{ bits} = 4 \times 67 \text{ bytes.}$

Παρατήρηση 2: Αν η TLB περιείχε 32 πλαίσια (αντί για σύνολα) και ήταν **μονοσήμαντης απεικόνισης** τότε θα ήταν:



Τώρα υπάρχουν δύο πεδία « $v - \kappa$ » ως ετικέτα και « κ » ως διεύθυνση πλαισίου, οπότε η ετικέτα έχει « $v - \kappa$ » μήκους $\text{ΑΛΣ} - \kappa = 40 - 5 = 35$ bits δίνει το μήκος αυτό στην ετικέτα της TLB

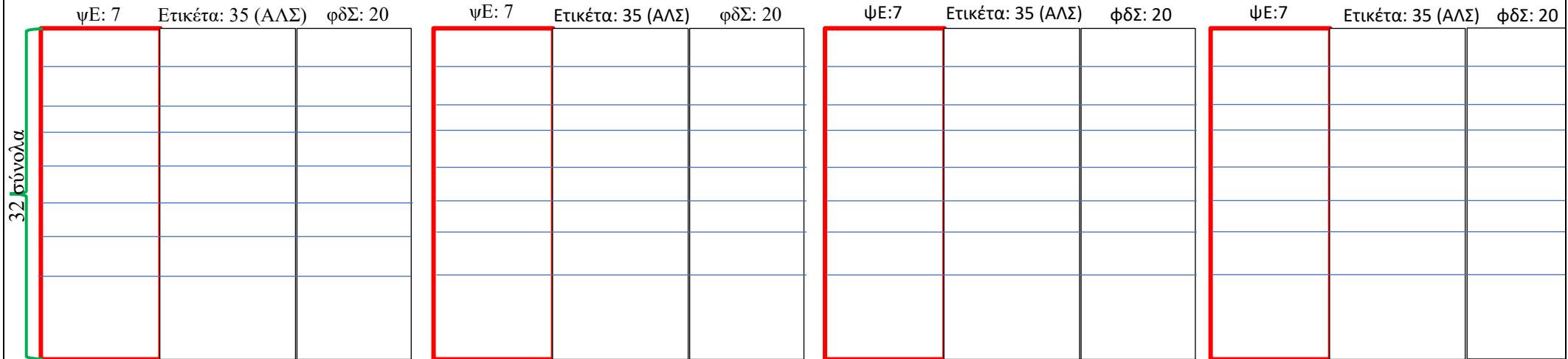
Χωρητικότητα TLB = $32 \text{ πλ.} \times (7 + 35 + 20) \frac{\text{bits}}{\text{πλ.}} = 32 \times 62 \text{ bits} = 4 \times 62 \text{ bytes.}$



$$v - \lambda = 35 \quad \lambda = 5 \quad \mu = 12$$

Μορφή διεύθυνσης

Τώρα υπάρχουν δύο πεδία « $v-\lambda$ » ως ετικέτα και « λ » ως διεύθυνση συνόλου, οπότε η ετικέτα « $v-\lambda$ »
έχει μήκος ΑΛΣ – $\lambda = 40 - 5 = 35$ bits και δίνει το μήκος αυτό στην ετικέτα της TLB



$$\text{Χωρητικότητα TLB} = 32 \text{ σύνολα} \times \frac{4 \pi \lambda}{\text{σύνολο}} \times (7 + 35 + 20) \frac{\text{bits}}{\pi \lambda} = 32 \times 4 \times 62 \text{ bits} = 4 \times 4 \times 62 \text{ bytes} = 16 \times 62 \text{ bytes}.$$

Θέμα Σεπτεμβρίου 2015 με υβριδική κρυφή μνήμη επεξεργαστή

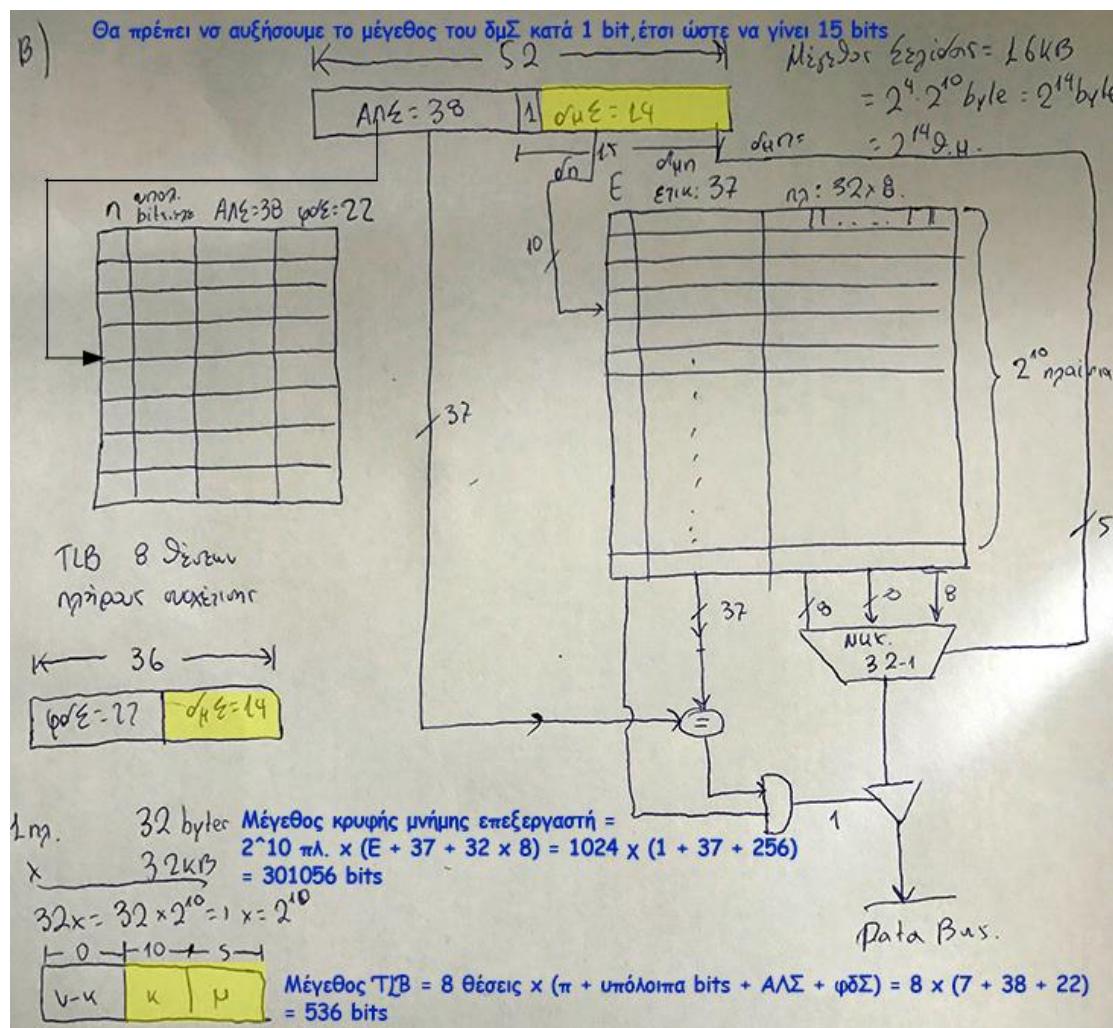
Θεωρούμε υπολογιστή με ιδεατή μνήμη (virtual memory) που υλοποιείται με την τεχνική της σελιδοποίησης. Η λογική διεύθυνση είναι των 32 δυαδικών ψηφίων, ενώ η φυσική διεύθυνση είναι των 36 δυαδικών ψηφίων και το μέγεθος της σελίδας είναι ίσο με 16 Kbytes. Η κύρια μνήμη έχει οργάνωση μιας ψηφιολέξης ανά θέση. Θεωρούμε ότι υπάρχει κρυφή μνήμη πίνακα σελίδων (TLB) των 8 θέσεων με οργάνωση πλήρους συσχέτισης και κρυφή μνήμη επεξεργαστή (CPU cache memory) οργάνωσης μονοσήμαντης απεικόνισης με 32 ψηφιολέξεις ανά πλαίσιο και χωρητικότητα 32 Kbytes. Για κάθε μία των κάτωθι περιπτώσεων να σχεδιάσετε το κατάλληλο σχήμα το οποίο να δείχνει πως γίνεται η προσπέλαση της κρυφής μνήμης πίνακα σελίδων της κρυφής μνήμης του επεξεργαστή. Σε κάθε σχήμα πρέπει να φαίνεται όλη η πληροφορία (πχ. εύρος πεδίων, πλήθος γραμμών διασύνδεσης).

α. Η κρυφή μνήμη επεξεργαστή προσπελαύνεται με φυσικές διευθύνσεις.

β. Η κρυφή μνήμη επεξεργαστή προσπελαύνεται με λογικές διευθύνσεις.

Λύση

Επειδή δεν έχουμε υβριδική κρυφή μνήμη επεξεργαστή, μπορούμε να αυξήσουμε το μέγεθος του πεδίου δμΣ, έτσι ώστε να γίνει ίσο με 15 bits, αφού τόσο είναι αθροιστικά το πεδίο «κ» και το πεδίο «μ» της διεύθυνσης της μονοσήμαντης απεικόνισης. Αναφορικά με τον υπολογισμό των μεγεθών των δύο κρυφών μνημών, χρησιμοποιούμε παρόμοια μεθοδολογία, δηλαδή πολλαπλασιάζουμε το πλήθος των γραμμών με το μέγεθος κάθε γραμμής σε κάθε περίπτωση. Το σχήμα βοηθά στον υπολογισμό αυτών των μεγεθών.



Θέμα Ιουνίου 2014 – Ιδεατή μνήμη με χρήση TLB

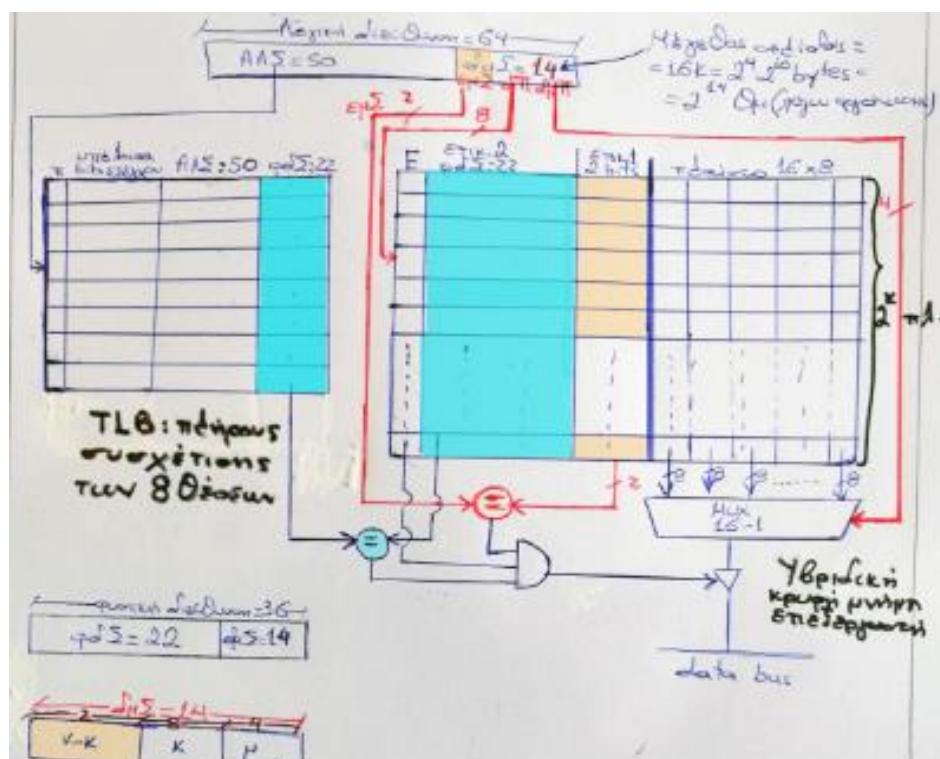
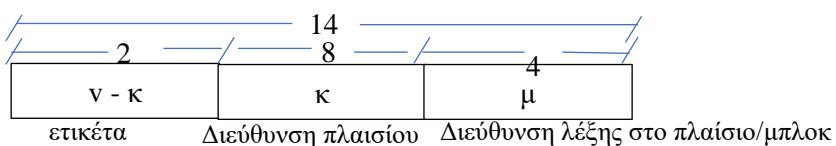
Θεωρούμε υπολογιστή με ιδεατή μνήμη (virtual memory) που υλοποιείται με την τεχνική της σελιδοποίησης. Θεωρούμε ότι υπάρχει κρυφή μνήμη πίνακα σελίδων (TLB) των 8 θέσεων με οργάνωση πλήρους συσχέτισης και υβριδική κρυφή μνήμη επεξεργαστή (CPU cache memory) με χωρητικότητα 4 Κψηφιολέξεις, 16 ψηφιολέξεις ανά πλαίσιο και οργάνωση μονοσήμαντης απεικόνισης (direct map). Η λογική διεύθυνση είναι των 64 δυαδικών ψηφίων ενώ η φυσική διεύθυνση των 36 δυαδικών ψηφίων και το μέγεθος της σελίδας ίσο με 16 Κψηφιολέξεις (Kbytes). Η κύρια μνήμη έχει οργάνωση μίας ψηφιολέξης ανά θέση. Να σχεδιάσετε το κατάλληλο σχήμα (κρυφή μνήμη πίνακα σελίδων και κρυφή μνήμη επεξεργαστή) που να δείχνει πως χρησιμοποιείται η λογική διεύθυνση για την προσπέλαση της κρυφής μνήμης πίνακα σελίδων και επεξεργαστή. Στο σχήμα να δίνονται όλα τα μεγέθη.

Αύση

A) Από τη χωρητικότητα της σελίδας που είναι 16 KB θα βρούμε το μήκος του πεδίου δμΣ τόσο της λογικής όσο και της φυσικής διεύθυνσης. Έχουμε ότι $16KB = 2^4 \times 2^{10} \text{ bytes} = 2^{14} \text{ bytes} = 2^{14} \theta.\mu.$ $\rightarrow \delta\mu\Sigma = 14$. Το **μήκος της λογικής διεύθυνσης δίνεται ίσο με 64 bits**. Άρα το $\Delta\Lambda\Sigma = \text{λογική διεύθυνση} - \delta\mu\Sigma = 64 - 14 = 50 \text{ bits}$ και το $\varphi\delta\Sigma = \text{φυσική διεύθυνση} 36 - 14 = 22 \text{ bits}$. Δηλαδή θα έχουμε:

ΑΛΣ = 50	δμΣ = 14	φδΣ = 22	δμΣ = 14
----------	----------	----------	----------

Αναφορικά με την **υβριδική κρυφή μνήμη επεξεργαστή** (CPU cache memory) που έχει χωρητικότητα 4 Κυψηφιολέξεις, 16 ψηφιολέξεις ανά πλαίσιο και οργάνωση **μονοσήμαντης απεικόνισης**, θα έχουμε τα πεδία που περιγράφονται στη συνέχεια. Πιο συγκεκριμένα, από το πλήθος των λέξεων ανά πλαίσιο και μπλοκ θα υπολογίσουμε το “ μ ”, οπότε θα έχουμε: $2^{\mu} = 16$ λέξεις $\Rightarrow \mu = 4$ bits. Από τη χωρητικότητα της κρυφής μνήμης επεξεργαστή που είναι 4KB, θα υπολογίσουμε τον αριθμό των πλαισίων που είναι $(2^2 \times 2^{10})/2^4 = 2^8 = 2^{\kappa} \Rightarrow \kappa = 8$ bits. Άρα τώρα θα υπάρχει και ετικέτα $\langle v - \kappa \rangle = 2$ bits.



Επειδή υπάρχει ετικέτα «ν – κ», σημαίνει ότι στην υβριδική κρυφή μνήμη επεξεργαστή εμφανίζεται **μια επιπλέον στήλη μεγέθους 2 bits**, μετά από τη στήλη φδΣ. Για την επιπλέον αυτή στήλη χρησιμοποιείται ο δεξιός συγκριτής του σχήματος που ακολουθεί, που συγκρίνει τα περιεχόμενά της με τα περιεχόμενα του πεδίου “εμΣ” του δμΣ τημάτως της λογικής διεύθυνσης. Επιπλέον, υπάρχει και ο αριστερός συγκριτής, για τη σύγκριση των πεδίων φδΣ των δύο κρυφών μνημάτων. Οι έξοδοι των δύο συγκριτών μαζί με το δυαδικό ψηφία εγκυρότητας “Ε” εισέρχονται σε μια πύλη AND, η έξοδος της οποίας μπαίνει στο στοιχείο τριών καταστάσεων που βρίσκεται στην έξοδο του MUX.

Άσκηση 5.21 από λυσάρι – Ιδεατή μνήμη με χρήση TLB

Θεωρούμε υπολογιστή με ιδεατή μνήμη (virtual memory) που υλοποιείται με την τεχνική της σελιδοποίησης. Θεωρούμε ότι υπάρχει κρυφή μνήμη πίνακα σελίδων (TLB) των 8 θέσεων με οργάνωση πλήρους συσχέτισης και υβριδική κρυφή μνήμη επεξεργαστή (CPU cache memory). Η λογική διεύθυνση είναι των 52 δυαδικών ψηφίων, ενώ η φυσική διεύθυνση των 36 δυαδικών ψηφίων και το μέγεθος της σελίδας ίσο με 4 KBytes. Η κύρια μνήμη έχει οργάνωση μίας ψηφιολέξης ανά θέση. Να σχεδιάσετε το κατάλληλο σχήμα που να δείχνει πως χρησιμοποιείται η λογική διεύθυνση για την προσπέλαση της κρυφής μνήμης πίνακα σελίδων και επεξεργαστή για κάθε μία των κάτωθι περιπτώσεων. Η κρυφή μνήμη επεξεργαστή σε κάθε περίπτωση να έχει το ελάχιστο κόστος υλοποίησης.

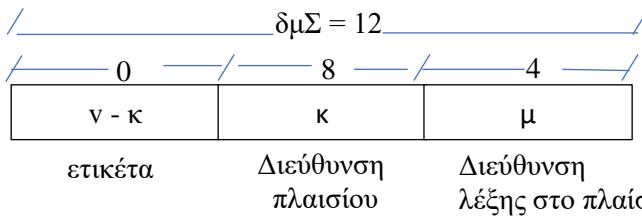
- α. Η κρυφή μνήμη επεξεργαστή έχει χωρητικότητα 4 KBytes με 16 ψηφιολέξεις ανά πλαίσιο.
- β. Η κρυφή μνήμη επεξεργαστή έχει χωρητικότητα 8 KBytes με 16 ψηφιολέξεις ανά πλαίσιο.

Λύση

Στην περίπτωση αυτή υπάρχει **υβριδική κρυφή μνήμη επεξεργαστή**, οπότε υπάρχουν οι εξής δύο διαφοροποιήσεις σε σχέση με πριν:

- i) Τώρα το δμΣ **δεν** μπορεί να αλλάξει όπως πριν.
- ii) Στην **υβριδική** κρυφή μνήμη επεξεργαστή η 2^n στήλη δεν είναι το ΑΛΣ, αλλά το φδΣ.

a) Για να έχουμε το **ελάχιστο κόστος υλοποίησης**, θα πρέπει να αναθέσουμε **οργάνωση μονοσήμαντης απεικόνισης στην κρυφή μνήμη επεξεργαστή**. Τότε θα έχουμε τα πεδία:



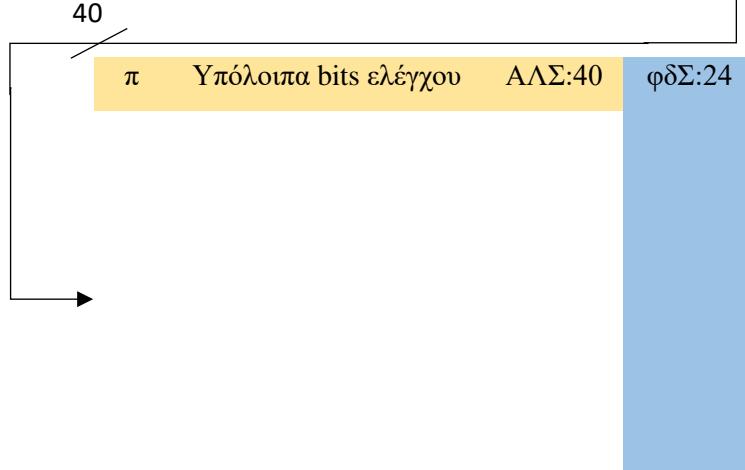
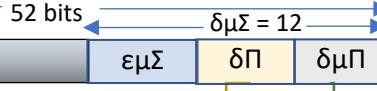
Από το πλήθος των λέξεων ανά πλαίσιο και μπλοκ θα υπολογίσουμε το μ, οπότε θα έχουμε: $2^{\mu} = 16$ λέξεις $\Rightarrow \mu = 4$ bits. Από τη χωρητικότητα της κρυφής μνήμης επεξεργαστή που είναι 4KB, θα υπολογίσουμε τον αριθμό των πλαισίων που είναι $(2^2 \times 2^{10})/2^4 = 2^8 = 2^{\kappa} \Rightarrow \kappa = 8$ bits. Επίσης, από το μέγεθος της σελίδας = μέγεθος πλαισίου σελίδας = 4KB = $2^2 \times 2^{10} = 2^{12} \Rightarrow \delta\mu\Sigma = 12$. Παρατηρούμε ότι $\delta\mu\Sigma = \kappa + \mu$ (αποδεκτό). Άρα το $v - \kappa = 0$. Αυτό γίνεται, διότι **δεν** μπορούμε να ξεπεράσουμε το δμΣ.

Το **μήκος της λογικής διεύθυνσης δίνεται ίσο με 52 bits**. Άρα το ΑΛΣ = λογική διεύθυνση – δμΣ = $52 - 12 = 40$ bits και το φδΣ = φυσική διεύθυνση – φδΣ = $36 - 12 = 24$ bits. Δηλαδή θα έχουμε:



Λογική διεύθυνση

52 bits



TLB: 8 θέσεων πλήρους συσχέτισης

Η κρυφή μνήμη επεξεργαστή έχει **οργάνωση μονοσήμαντης απεικόνισης** και περιέχει 16 ψηφιολέξεις ανά πλαίσιο και χωρητικότητα 4KB.

Ισχύει ότι το $2^\mu \geq 16 \Rightarrow \mu = 4$ bits. Το $\delta\mu\Pi \leftrightarrow \mu$

1 πλ. περιέχει 16 ψηφιολέξεις

$x; 4 \text{ KB}$

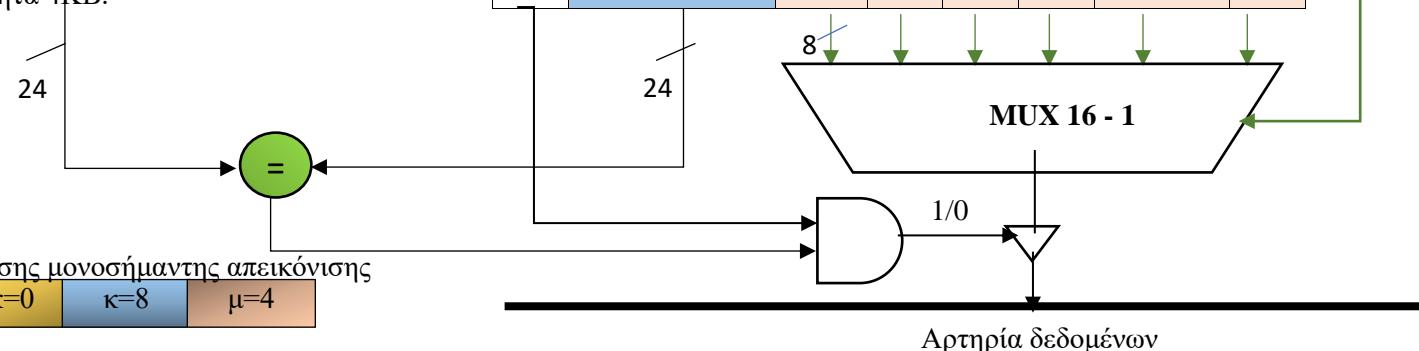
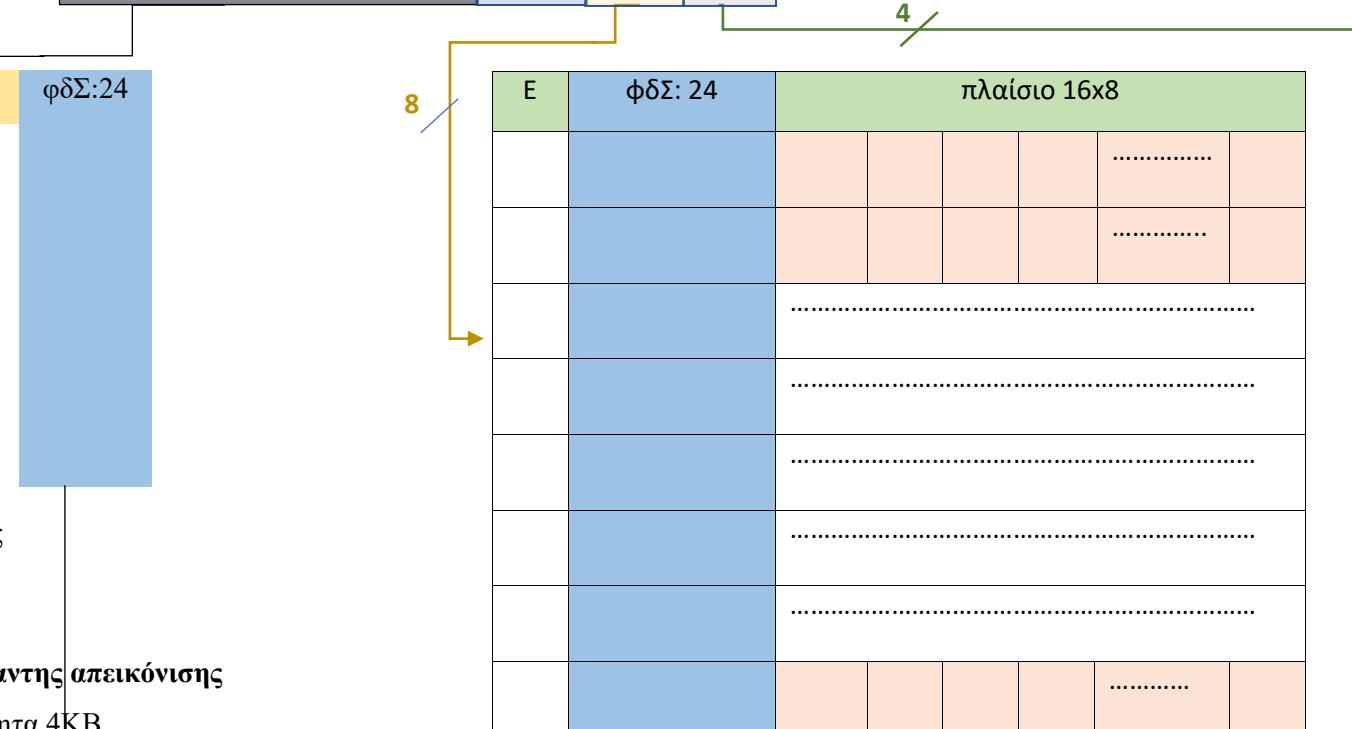
$x = 2^{12}/2^4 = 2^8$, άρα $\kappa = 8$. Το $\delta\Pi \leftrightarrow \kappa$

Φυσική διεύθυνση = 36

φδΣ = 24	δμΣ = 12
----------	----------

Πεδία διεύθυνσης μονοσήμαντης απεικόνισης

$\nu - \kappa = 0$	$\kappa = 8$	$\mu = 4$
--------------------	--------------	-----------





b) Στην περίπτωση αυτή το μόνο που αλλάζει είναι το μέγεθος της κρυφής μνήμης επεξεργαστή, όπου από 4 KB γίνεται 8KB. Αν υποθέσουμε ότι έχουμε και πάλι οργάνωση μονοσήμαντης απεικόνισης, τότε τα τρία πεδία της διεύθυνσης που παράγει ο επεξεργαστής θα έχουμε τα μεγέθη που φαίνονται στη συνέχεια:

$$\delta\mu\Sigma = 12$$

0 9 4

V - K	K	μ
-------	---	-------

ετικέτα Διεύθυνση Διεύθυνση
πλαισίου λέξης στο πλαίσιο/μπλοκ

Από το πλήθος των λέξεων ανά πλαίσιο και μπλοκ θα υπολογίσουμε το μ , οπότε θα έχουμε: $2^\mu = 16$ λέξεις $\Rightarrow \mu = 4$ bits. Από τη χωρητικότητα της κρυφής μνήμης επεξεργαστή που είναι 8 KB, θα υπολογίσουμε τον αριθμό των πλαισίων που είναι $(2^3 \times 2^{10})/2^4 = 2^9 = 2^\kappa = 2^9$ $\Rightarrow \kappa = 9$ bits. Επίσης, από το μέγεθος της σελίδας = μέγεθος πλαισίου σελίδας = 4KB = $2^2 \times 2^{10} = 2^{12}$ $\rightarrow \delta\mu\Sigma = 12$. **Παρατηρούμε ότι $\delta\mu\Sigma = 12 < \kappa + \mu = 13$ (μη αποδεκτό).** Αυτό γίνεται, διότι λόγω της υβριδικής κρυφής μνήμης επεξεργαστή, **δεν** μπορούμε να ξεπεράσουμε το $\delta\mu\Sigma$. Επομένως θα πρέπει να χρησιμοποιήσουμε οργάνωση 2-τρόπων συνόλου συσχέτισης. Στην περίπτωση αυτή η διεύθυνση που παράγει ο επεξεργαστής θα αποτελείται από τα πεδία που φαίνονται στη συνέχεια:

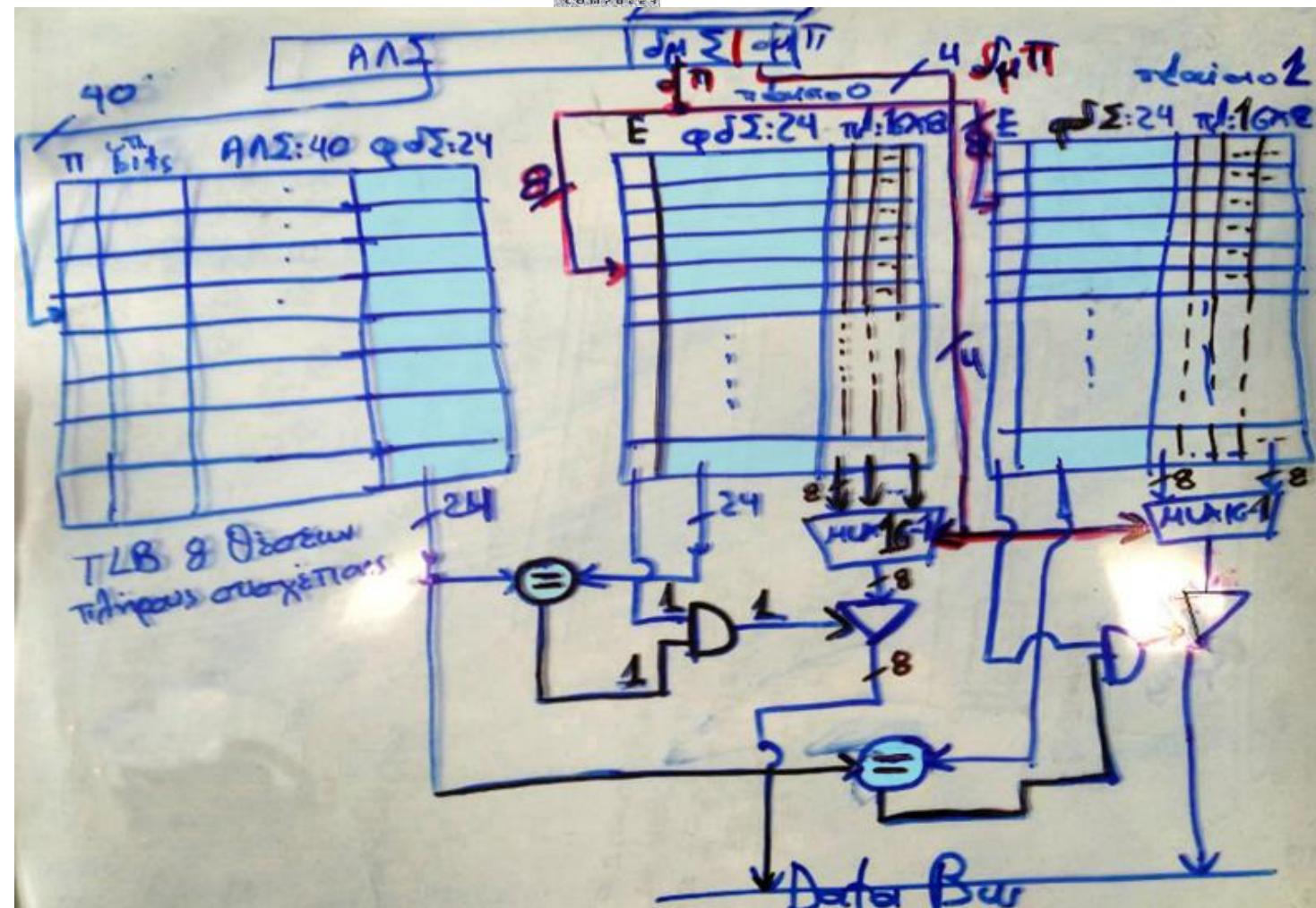
$$\delta\mu\Sigma = 12$$

A horizontal line with tick marks at 0, 8, and 4.

$$v - \lambda \quad \lambda \quad \mu$$

ετικέτα Διεύθυνση συνόλου Διεύθυνση λέξης στο πλαίσιο/μπλοκ

Από το πλήθος των λέξεων ανά πλαίσιο και μπλοκ θα υπολογίσουμε το μ , οπότε θα έχουμε: $2^\mu = 16$ λέξεις $\Rightarrow \mu = 4$ bits. Από τη χωρητικότητα της κρυφής μνήμης επεξεργαστή που είναι 8 KB, θα υπολογίσουμε αρχικά τον αριθμό των πλαισίων που είναι $(2^3 \times 2^{10})/2^4 = 2^9 = 2^k$ πλαίσια και στη συνέχεια θα υπολογίσουμε τον αριθμό των συνόλων. Επειδή έχουμε 2^9 πλαίσια και οργάνωση 2-τρόπων συνόλου συσχέτισης, θα έχουμε $2^9/2 = 2^8$ σύνολα $= 2^\lambda \Rightarrow \lambda = 8$ bits Επίσης, από το μέγεθος της σελίδας = μέγεθος πλαισίου σελίδας = 4 KB = $2^2 \times 2^{10} = 2^{12} \Rightarrow \delta\mu\Sigma = 12$. Παρατηρούμε ότι $\delta\mu\Sigma = 12 = \lambda + \mu$ (αποδεκτό). Προφανώς, το πεδίο $v - \lambda = 0$. Στη συνέχεια, ακολουθεί ο σχεδιασμός της κρυφής μνήμης επεξεργαστή με οργάνωση 2 – τρόπων συνόλου συσχέτισης και της κρυφής μνήμης πίνακα σελίδων (TLB) με οργάνωση πλήρους συσχέτισης.



Άσκηση 5.22 – Ιδεατή μνήμη με χρήση TLB

Θεωρούμε υπολογιστή με ιδεατή μνήμη (virtual memory) που υλοποιείται με την τεχνική της σελιδοποίησης. Θεωρούμε ότι υπάρχει κρυφή μνήμη πίνακα σελίδων (TLB) των 8 θέσεων με οργάνωση πλήρους συσχέτισης και υβριδική κρυφή μνήμη επεξεργαστή (CPU cache memory) με οργάνωση μονοσήμαντης απεικόνισης. Η λογική διεύθυνση είναι των 52 δυαδικών ψηφίων ενώ η φυσική διεύθυνση των 36 δυαδικών ψηφίων και το μέγεθος της σελίδας ίσο με 4 Kbytes. Η κρυφή μνήμη επεξεργαστή αποτελείται από 16 πλαίσια των 16 ψηφιολέξεων το κάθε ένα. Η κύρια μνήμη έχει οργάνωση μίας ψηφιολέξης ανά θέση. Τα τρέχοντα περιεχόμενα της κρυφής μνήμης πίνακα σελίδων και επεξεργαστή δίνονται στο σχήμα 5.21.1.

α. Να σχεδιάσετε το κατάλληλο σχήμα το οποίο να δείχνει πως χρησιμοποιείται η λογική διεύθυνση για την προσπέλαση της κρυφής μνήμης πίνακα σελίδων και επεξεργαστή.

β. Υποθέστε ότι ο επεξεργαστής παράγει τις ακόλουθες λογικές διευθύνσεις στο δεκαεξαδικό: A000BC124C83A, A000BC124C83A, A000BC124C341, B000000011201.

Να περιγράψετε τι θα συμβεί σε κάθε περίπτωση και στην περίπτωση αποτυχίας (miss) να δώσετε τα περιεχόμενα της κρυφής μνήμης πίνακα σελίδων και επεξεργαστή μετά την ενημέρωση.



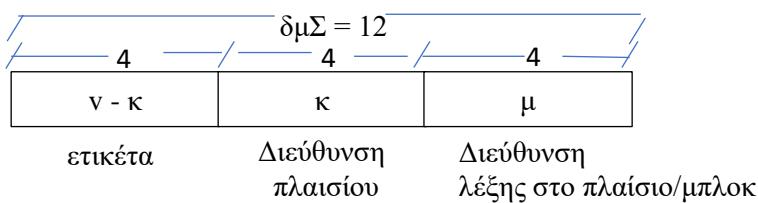
Λύση

α. Στην περίπτωση αυτή υπάρχει και πάλι υβριδική κρυφή μνήμη επεξεργαστή, οπότε ισχύουν και πάλι οι εξής δύο διαφοροποιήσεις σε σχέση με πριν:

i) Το δμΣ δεν μπορεί να αλλάξει όπως πριν.

ii) Στην υβριδική κρυφή μνήμη επεξεργαστή η 2^η στήλη δεν είναι το ΑΛΣ, αλλά το φδΣ.

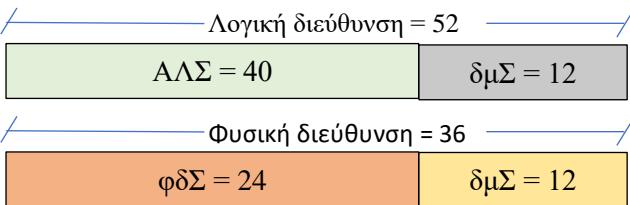
Έχουμε οργάνωση μονοσήμαντης απεικόνισης στην κρυφή μνήμη επεξεργαστή. Τότε θα έχουμε τα πεδία:



Από τον αριθμό των λέξεων ανά πλαίσιο/μπλοκ έχουμε $2^{\mu} = 16$ λέξεις $\Rightarrow \mu = 4 \text{ bits}$. Από τον αριθμό των πλαισίων της κρυφής μνήμης επεξεργαστή που είναι 16, θα υπολογίσουμε το κ, που είναι $2^{\kappa} = 16 \Rightarrow \kappa = 4 \text{ bits}$. Από το μέγεθος της σελίδας = μέγεθος πλαισίου σελίδας = 4KB = $2^2 \times 2^{10} = 2^{12} \Rightarrow \delta\mu\Sigma = 12$. Θα πρέπει το $\delta\mu\Sigma = (ν - κ) + κ + \mu$ και επειδή $\delta\mu\Sigma = 12$ και το $\kappa = \mu = 4$, θα πρέπει το $\nu - \kappa = 4 \text{ bits}$. Άρα τώρα υπάρχει ετικέτα στη διεύθυνση της υβριδικής κρυφής μνήμης επεξεργαστή. Σε αυτήν την ετικέτα αντιστοιχεί το πεδίο εμΣ του δμΣ. Δηλαδή:

εμΣ	δΠ	δμΠ
-----	----	-----

Το μήκος της λογικής διεύθυνσης δίνεται ίσο με 52 bits. Άρα το $\Lambda\Sigma =$ λογική διεύθυνση – $\delta\mu\Sigma = 52 - 12 = 40$ bits και το $\varphi\delta\Sigma =$ φυσική διεύθυνση $36 - 12 = 24$ bits. Δηλαδή, θα έχουμε:



b. Ο επεξεργαστής παράγει τις ακόλουθες διευθύνσεις:

A000BC124C – 83A → $\varepsilon\mu\Sigma = 8, \delta\Pi = 3, \delta\mu\Pi = \Lambda$

A000BC124C – 341 → $\varepsilon\mu\Sigma = 3, \delta\Pi = 4, \delta\mu\Pi = 1$

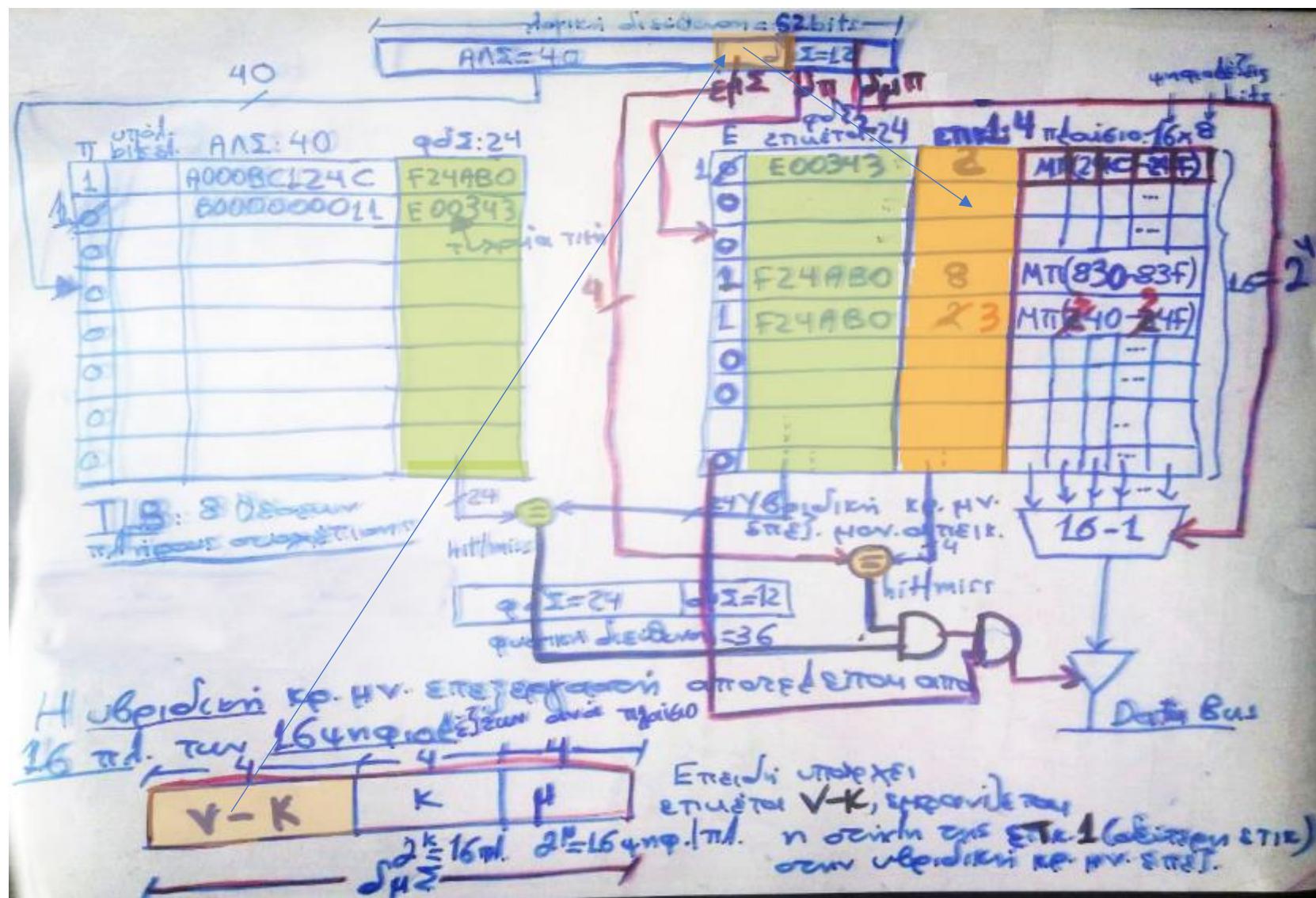
B000000011 – 201 → $\varepsilon\mu\Sigma = 2, \delta\Pi = 0, \delta\mu\Pi = 1$

Θα ελέγξουμε κάθε μία από αυτές τις τρεις διευθύνσεις αν υπάρχει το $\Lambda\Sigma$ τμήμα μέσα στην TLB.

Αναφορικά με την πρώτη διεύθυνση, παρατηρούμε ότι το $\Lambda\Sigma$ τμήμα αυτής υπάρχει ήδη στην TLB. Αυτό σημαίνει ότι γίνεται αυτόματη μετάφραση στο $\varphi\delta\Sigma = F24AB0$ που υπάρχει δίπλα της. Παρατηρούμε στη συνέχεια ότι το συγκεκριμένο $\varphi\delta\Sigma$ υπάρχει σε δύο διαφορετικές θέσεις της υβριδικής κρυφής μνήμης επεξεργαστή στα δεξιά. Αν παρατηρήσουμε, το **$\delta\Pi = 3$** , οπότε θα μεταβούμε στην **πρώτη** από αυτές, όπου η ετικέτα είναι «8» και ταυτίζεται με το πεδίο $\varepsilon\mu\Sigma$ που είναι επίσης «8». Αυτό σημαίνει ότι ο δεξιός συγκριτής θα δώσει «1» (επιτυχία). Επίσης, και ο αριστερός συγκριτής, που συγκρίνει τα πεδία $\varphi\delta\Sigma$ τω δύο κρυφών μνημών, θα δώσει «1» (επιτυχία), λόγω της τιμής **F24AB0** που εμφανίζεται και στα δύο πεδία. Τέλος, επειδή και το ψηφίο εγκυρότητας είναι «1», σημαίνει ότι στο στοιχείο τριών καταστάσεων που υπάρχει στην έξοδο του MUX, θα φτάσει η τιμή «1», με αποτέλεσμα η έξοδος του πολυπλέκτη να περάσει στην αρτηρία δεδομένων. Δηλαδή έχουμε επιτυχία και τα δεδομένα/εντολή που εμφανίζονται στην έξοδο του πολυπλέκτη είναι τα ζητούμενα.

Για τη δεύτερη διεύθυνση έχουμε και πάλι το **ίδιο $\Lambda\Sigma \rightarrow \varphi\delta\Sigma$** , αλλά τώρα θα πάμε στο πλαίσιο 4 της υβριδικής κρυφής μνήμης επεξεργασίας (διότι $\delta\Pi = 4$), όπου η ετικέτα είναι $2 \neq 3$, άρα ο δεξιός συγκριτής θα δώσει αποτυχία. Τότε θα πρέπει να μεταφερθεί το περιεχόμενο των φυσικών διευθύνσεων **F24AB0340 – F24AB034F** από την κύρια μνήμη στην κρυφή, στο πλαίσιο με διεύθυνση 4 και να ενημερωθεί κατάλληλα η ετικέτα, δηλαδή να γίνει από 2 → 3. Ο αριστερός συγκριτής θα δώσει και πάλι «1», αλλά στο στοιχείο τριών καταστάσεων που είναι στην έξοδο του MUX, θα φτάσει το «0», με αποτέλεσμα να μην υπάρχει έξοδος.

Για την τρίτη διεύθυνση, επειδή **δεν** υπάρχει στην TLB, θα πρέπει να μεταφερθεί εκεί (στη δεύτερη θέση λόγω πλήρους συσχέτισης). Έχουμε μια διαφορετική λογική διεύθυνση (**B000000011 – 201**), οπότε θα προστεθεί στην TLB το $\Lambda\Sigma$ τμήμα αυτής της λογικής διεύθυνσης, όπως φαίνεται στην Εικόνα. Επειδή δεν αναφέρεται κάτι σχετικό στην εκφώνηση, θα πρέπει να γράψουμε κάποιο αυθαίρετο $\varphi\delta\Sigma$, στο οποίο θα μεταφραστεί το $\Lambda\Sigma$. Υποθέτουμε ότι το **$\Lambda\Sigma: B000000011 \rightarrow \varphi\delta\Sigma: E00343$** . Δηλαδή υποθέσαμε ότι το λειτουργικό σύστημα τοποθέτησε τη σελίδα, στο πλαίσιο με φυσική διεύθυνση **E00343**. Λόγω του $\delta\mu\Sigma$ (201) στο πλαίσιο 0 της υβριδικής κρυφής μνήμης επεξεργασίας μεταφέρονται οι φυσικές διευθύνσεις **E00343240 – E0034324F**.





Θέμα προόδου 2018 με κρυφή – ιδεατή μνήμη

Θεωρούμε υπολογιστή με ιδεατή μνήμη (virtual memory) και υβριδική κρυφή μνήμη επεξεργαστή (CPU cache memory) με οργάνωση 2-τρόπων συνόλου συσχέτισης και πλαίσιο των 64 ψηφιολέξεων (bytes). Τόσο η λογική διεύθυνση όσο και η φυσική διεύθυνση είναι των 32 δυαδικών ψηφίων, και το μέγεθος της σελίδας ίσο με 8 Κψηφιολέξεις (Kbytes). Η κύρια μνήμη έχει οργάνωση μίας ψηφιολέξης ανά θέση, δηλαδή σε κάθε διεύθυνση αντιστοιχεί μια ψηφιολέξη. Να προσδιορίσετε το **μέγιστο** μέγεθος της κρυφής μνήμης επεξεργαστή.

Λύση

Αφού το μέγεθος της σελίδας είναι ίσο με 8 Κψηφιολέξεις = $2^3 \times 2^{10}$ ψηφιολέξεις = 2^{13} ψηφιολέξεις = 2^{13} θ.μ. και τα 13 λιγότερο σημαντικά δυαδικά ψηφία της διεύθυνσης θα χρησιμοποιούνται για την επιλογή της ψηφιολέξης μέσα στη σελίδα και έχουν την ίδια τιμή τόσο στην λογική διεύθυνση όσο και την φυσική διεύθυνση. Αφού η άσκηση μας ζητάει το μέγιστο μέγεθος κρυφής μνήμης, αυτά τα 13 δυαδικά ψηφία θα αποτελούν το πεδίο διεύθυνσης μέσα στο πλαίσιο και τη διεύθυνση συνόλου. Σύμφωνα με την άσκηση το πλαίσιο είναι των 64 ψηφιολέξεων = 2^6 ψηφιολέξεων, επομένως από τα 13 δυαδικά ψηφία τα 6 λιγότερο σημαντικά δυαδικά ψηφία δίνουν τη διεύθυνση της ψηφιολέξης μέσα στο πλαίσιο και επομένως, αφού θέλουμε την μέγιστη χωρητικότητας κρυφής μνήμης, τα υπόλοιπα $13 - 6 = 7$ τη διεύθυνση του συνόλου. Οπότε η κρυφή μνήμη έχει 2^7 σύνολα και αφού είναι 2-τρόπων συνόλου συσχέτισης έχει 2×2^7 πλαίσια = 2^8 πλαίσια. Λαμβάνοντας υπόψη ότι κάθε πλαίσιο είναι των 64 ψηφιολέξεων = 2^6 ψηφιολέξεων, η κρυφή μνήμη έχει χωρητικότητα $2^8 \times 2^6$ ψηφιολέξεις = 2^{14} ψηφιολέξεις = $2^4 \times 2^{10}$ ψηφιολέξεις = 16 Κψηφιολέξεις.

Κεφάλαιο 7 – Υπερβαθμωτοί επεξεργαστές και ΠΜΜΕ επεξεργαστές

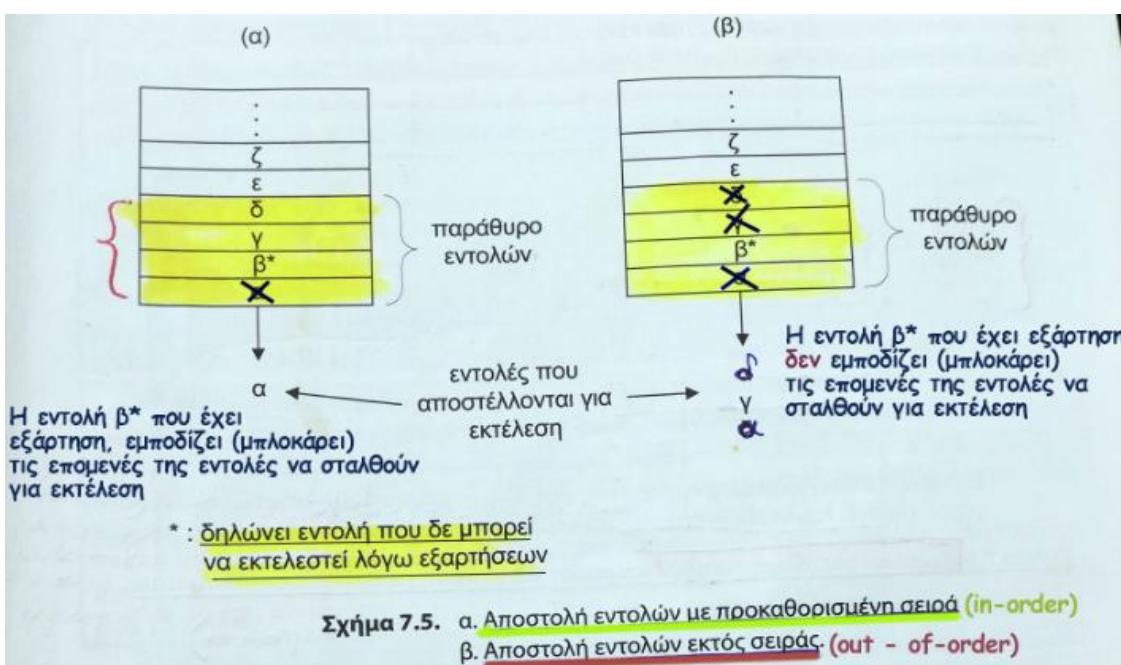
7.1 Υπερβαθμωτοί επεξεργαστές

7.1.1. Βασικά σημεία θεωρίας

1. Ένας υπερβαθμωτός επεξεργαστής διαθέτει **ξεχωριστή κρυφή μνήμη εντολών και δεδομένων**. Επιπλέον διαθέτει τη ΜΠΑΕ (Μονάδα Προσωρινής Αποθήκευσης Εντολών) στην οποία προσκομίζονται σε κάθε χρονική περίοδο μια ομάδα από μ γειτονικές εντολές της κρυφής μνήμης εντολών. Η τιμή του μ είναι τουλάχιστον ίση με το μέγιστο αριθμό κ των εντολών που μπορούν να αποκαθιστούν και να ξεκινήσει η εκτέλεσή τους ανά χρονική περίοδο. Σκοπός της ΜΠΑΕ είναι η **εξομάλυνση των διαφοροποιήσεων του πλήθους των εντολών** που προσκομίζονται σε κάθε χρονική μονάδα. Σε κάποιες περιπτώσεις το πλήθος των εντολών που προσκομίζονται στη ΜΠΑΕ είναι μικρότερο του κ. Για παράδειγμα, αν οι εντολές που πρέπει να προσκομιστούν σε κάποια χρονική περίοδο δεν βρίσκονται στην κρυφή μνήμη, τότε καμία εντολή δεν προσκομίζεται. Αν ο **ρυθμός προσκόμισης εντολών μ είναι μεγαλύτερος του ρυθμού αποκαθιστούμενης εντολών κ**, τότε στη ΜΠΑΕ υπάρχουν πάντα εντολές προς αποκαθιστούμενης εκτέλεσης και εκτέλεσης, με αποτέλεσμα οι μονάδες του υπερβαθμωτού να μην μένουν ανεκμετάλλευτες. Μετά τη ΜΠΑΕ ακολουθεί η μονάδα ΜΑΕΕΑ (Μονάδα Αποκαθιστούμενης Ελέγχου Εξαρτήσεων και Αποστολής εντολών) που ψάχνει για τυχόν εξαρτήσεις.

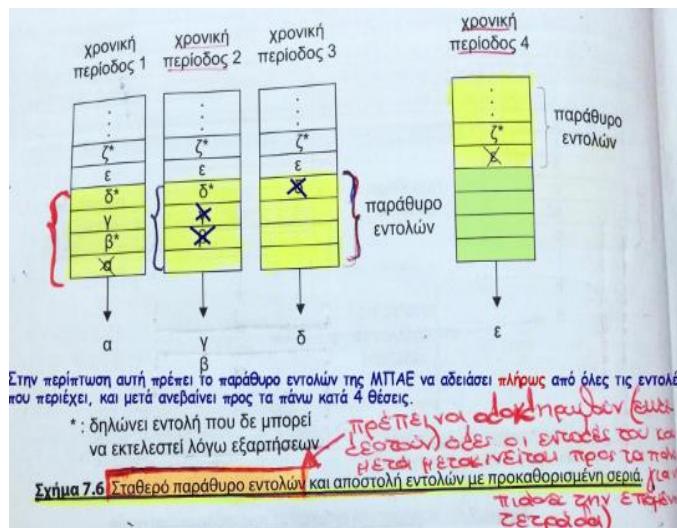
2. Στους υπερβαθμωτούς υπάρχουν **δύο ειδών εξαρτήσεις**: δομικές και εξαρτήσεις δεδομένων.

3. Η αποστολή εντολών από τη ΜΠΑΕ προς τη ΜΑΕΕΑ μπορεί να γίνει με δύο τρόπους: **με προκαθορισμένη σειρά (in-order)** και **εκτός σειράς (out-of-order)**. Στην πρώτη περίπτωση αν μια εντολή έχει εξάρτηση εμποδίζει τις επόμενες από αυτή που βρίσκονται μέσα στο παράθυρο εντολών να σταλθούν για εκτέλεση στις λειτουργικές μονάδες του υπερβαθμωτού επεξεργαστή για να εκτελεστούν, αν και θα μπορούσαν γιατί δεν έχουν κάποια εξάρτηση. Στη δεύτερη περίπτωση αν υπάρχουν εντολές στο παράθυρο εντολών που μπορούν να εκτελεστούν διότι δεν έχουν κάποιο πρόβλημα εξαρτήσεων και έπονται της εντολής που δεν μπορεί να εκτελεστεί, αποστέλλονται στις λειτουργικές μονάδες προς εκτέλεση. Η διαφορά των δύο τρόπων διευκρινίζεται με το ακόλουθο σχήμα:



4. Το παράθυρο εντολών της ΜΠΑΕ μπορεί να είναι σταθερό ή μετατοπιζόμενο. Στην πρώτη περίπτωση πρέπει να αδειάσει πλήρως από όλες τις εντολές που περιέχει για να ανέβει προς τα πάνω τόσες θέσεις όσο είναι το μέγεθός του. Στη δεύτερη περίπτωση μετατοπίζεται αυτόματα ανάλογα με το πλήθος των εντολών που αποστέλλονται προς εκτέλεση. Όταν οι υπερβαθμωτοί επεξεργαστές **δεν** διαθέτουν Μονάδα Αναμονής Αποστολής (MAA) χρησιμοποιούν μετατοπιζόμενο παράθυρο,

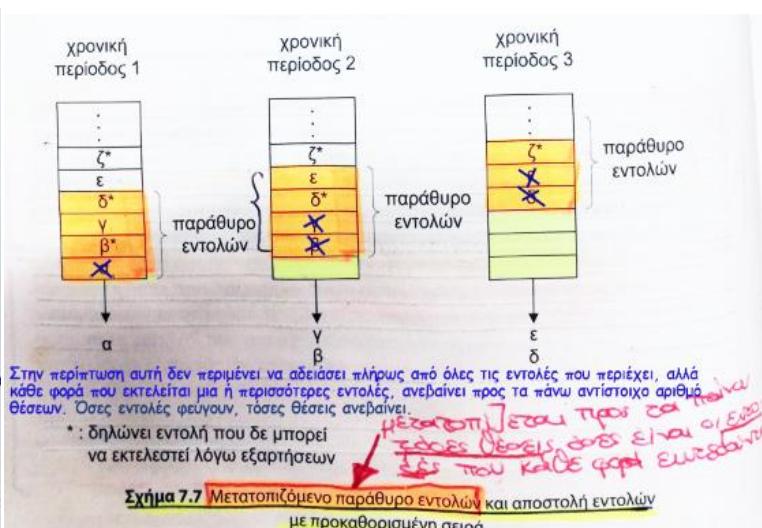
ενώ οι πιο πρόσφατοι που διαθέτουν Μονάδα Αναμονής Αποστολής (MAA), χρησιμοποιούν σταθερό παράθυρο εντολών. Η διαφορά των δύο τρόπων διευκρινίζεται με το ακόλουθο σχήμα:



Στην περίπτωση αυτή πρέπει το παράθυρο εντολών της ΜΤΑΕ να αδειάσει πλήρως από όλες τις εντολές που περιέχει, και μετά ανεβαίνει προς τα πάνω κατά 4 θέσεις.

* : δηλώνει εντολή που δε μπορεί να εκτελεστεί λόγω εξαρτήσεων
ΠΡΕΠΕΙ ΝΑ ΕΔΩ ΚΛΗΡΟΥΘΕΙΝ (Εδώ οι εντολές του κατεύθυνσης προς τα πάνω μέχρι η επόμενη θέση που καθέ έχει επεξεργαστεί)

Σχήμα 7.6 Σταθερό παράθυρο εντολών και αποστολή εντολών με προκαθορισμένη σειρά



Στην περίπτωση αυτή δεν περιμένει να αδειάσει πλήρως από όλες τις εντολές που περιέχει, αλλά φορά που εκτελείται μια ή περισσότερες εντολές, ανεβαίνει προς τα πάνω αντίστοιχα αριθμό θέσεων. Όσες εντολές φεύγουν, τόσες θέσεις ανεβαίνει.

* : δηλώνει εντολή που δε μπορεί να εκτελεστεί λόγω εξαρτήσεων
ΜΕΤΑΤΟΠΙΖΟΜΕΝΟ ΠΑΡΑΘΥΡΟ ΕΝΤΟΛΩΝ ΚΑΙ ΑΠΟΣΤΟΛΗ ΕΝΤΟΛΩΝ
ΜΕ ΠΡΟΚΑΘΟΡΙΣΜΕΝΗ ΣΕΙΡΑ

Σχήμα 7.7 Μετατοπιζόμενο παράθυρο εντολών και αποστολή εντολών με προκαθορισμένη σειρά

5. Όταν ο υπερβαθμωτός επεξεργαστής διαθέτει Μονάδα Αναμονής Αποστολής (MAA) θα διαθέτει και Μονάδα Ελέγχου Εξαρτήσεων και Αποστολής εντολών (MEEA) και σε κάποιους υπερβαθμωτούς για κάθε λειτουργική μονάδα υπάρχει μια MAA και μια MEEA και σε κάποιους άλλους για κάθε ομάδα λειτουργικών μονάδων με κάποια κοινά χαρακτηριστικά υπάρχει μια κοινή MAA και MEEA, ενώ σε κάποιους άλλους υπάρχει μια MAA και MEEA για όλες τις λειτουργικές μονάδες.

6. Όταν χρησιμοποιείται η έκφραση "η εντολή εκτελέστηκε" σημαίνει ότι η εντολή εκτελέστηκε αλλά το αποτέλεσμα αποθηκεύτηκε σε ένα προσωρινό καταχωρητή, δηλαδή δεν έχει αποθηκευτεί ακόμη στον καταχωρητή ή τη θέση μνήμης που δηλώνεται στην εντολή που εμφανίζεται στο πρόγραμμα. Αντίθετα όταν χρησιμοποιείται η έκφραση "η εντολή ολοκληρώθηκε" εννοείται ότι η εντολή εκτελέστηκε και το αποτέλεσμα αποθηκεύτηκε στον καταχωρητή ή τη θέση μνήμης που δηλώνεται στην εντολή που εμφανίζεται στο πρόγραμμα.

7. Όσον αφορά τη συνέπεια του επεξεργαστή διακρίνουμε την **ασθενή** και την **ισχυρή** συνέπεια. Όταν ένας επεξεργαστής ακολουθεί το μοντέλο της **ασθενούς συνέπειας** επιτρέπεται εντολές να ολοκληρωθούν εκτός σειράς, αρκεί να μην παραβιάζουν τις εξαρτήσεις δεδομένων. Οι περισσότεροι από τους πρώτους υπερβαθμωτούς επεξεργαστές ακολουθούσαν αυτό το μοντέλο συνέπειας. Ως παραδείγματα αναφέρουμε τους Power1, Power2, PowerPC 601 και R8000. Στους επεξεργαστές που ακολουθούν το μοντέλο **ισχυρής συνέπειας** οι εντολές πρέπει να ολοκληρώνονται με τη σειρά που εμφανίζονται στο πρόγραμμα. Συνήθως αυτό επιτυγχάνεται με τη χρήση του μηχανισμού επαναδιάταξης αποτελεσμάτων (Recorder Buffer). Πάρα πολλοί από τους σύγχρονους υπερβαθμωτούς επεξεργαστές χρησιμοποιούν το μηχανισμό επαναδιάταξης αποτελεσμάτων.

Θέμα με υπερβαθμωτούς και το μηχανισμό επαναδιάταξης αποτελεσμάτων

Να εξηγήσετε ποιο πρόβλημα επιλύουμε με το **μηχανισμό επαναδιάταξης αποτελεσμάτων** σε έναν υπερβαθμωτό επεξεργαστή. Να αναφέρετε ποιος είναι ο ρόλος του μηχανισμού επαναδιάταξης αποτελεσμάτων σε έναν υπερβαθμωτό επεξεργαστή, δηλαδή το πρόβλημα το οποίο λύνει καθώς και την πηγή ή τις πηγές του προβλήματος.

Λύση

Ο μηχανισμός επαναδιάταξης αποτελεσμάτων εξασφαλίζει ότι οι εντολές ολοκληρώνονται **αυστηρά** με την σειρά που εμφανίζονται στο πρόγραμμα. Οι εντολές εκτελούνται εκτός σειράς αλλά ακόμα και αν τις εκτελέσουμε με τη σειρά εμφάνισης τους, λόγω των διαφορετικών καθυστερήσεων των διαφορετικών μονάδων (π.χ. πολλαπλασιαστές, αθροιστές) θα τελείωναν και πάλι εκτός σειράς.



Θέμα Θεωρίας Σεπτεμβρίου 2021 με υπερβαθμωτούς επεξεργαστές

Στον υπερβαθμωτό επεξεργαστή (superscalar processor) η επιλογή των εντολών που θα σταλούν για εκτέλεση σε ένα κύκλο ρολογιού γίνεται κατά τη μεταγλώττιση του πηγαίου προγράμματος.

α) Σωστό

β) Λάθος

Λύση

Λάθος, γίνεται κατά την εκτέλεση. Στον επεξεργαστή **ΠΙΜΜΕ** η επιλογή των εντολών που θα σταλούν για εκτέλεση σε ένα κύκλο ρολογιού γίνεται κατά την **μεταγλώττιση** του πηγαίου προγράμματος, αντίθετα στον **υπερβαθμωτό επεξεργαστή** γίνεται κατά την **εκτέλεση** του πηγαίου προγράμματος.

Θέμα Θεωρίας Σεπτεμβρίου 2021 με υπερβαθμωτούς επεξεργαστές

Στους υπερβαθμωτούς επεξεργαστές οι εξαρτήσεις Ανάγνωση Μετά από Εγγραφή και Εγγραφή Μετά από Ανάγνωση είναι πραγματικές εξαρτήσεις, ενώ η εξάρτηση Εγγραφή Μετά από Εγγραφή είναι ψευδοεξάρτηση.

α) Σωστό

β) Λάθος

Λύση

Λάθος, στον υπερβαθμωτό επεξεργαστή οι εξαρτήσεις **ΑΜΕ είναι πραγματικές**, ενώ οι **ΕΜΕ και ΕΜΑ είναι ψευδοεξαρτήσεις**, και μπορούν πάντα να επιλυθούν. Η χρήση **συνώνυμων καταχωρητών**, θα βοηθήσει στην επίλυση των ψευδοεξαρτήσεων.

Θέμα Θεωρίας Σεπτεμβρίου 2021 με υπερβαθμωτούς επεξεργαστές

Στους υπερβαθμωτούς επεξεργαστές με αποστολή εντολών προς εκτέλεση με προκαθορισμένη σειρά (in-order) είναι δυνατόν να εμφανιστούν μόνο εξαρτήσεις Ανάγνωση Μετά από Εγγραφή και Εγγραφή Μετά από Ανάγνωση.

α) Σωστό

β) Λάθος

Λύση

Λάθος, εμφανίζονται τρία συνολικά είδη εξαρτήσεων.

Θέμα Θεωρίας Ιουνίου 2015 με μηχανισμό επαναδιάταξης αποτελεσμάτων σε υπερβαθμωτούς

Ο μηχανισμός επαναδιάταξης αποτελεσμάτων (reorder buffer) είναι απαραίτητος:

α. στους υπερβαθμωτούς επεξεργαστές που η αποστολή εντολών προς εκτέλεση γίνεται με προκαθορισμένη σειρά (in order).

β. στους επεξεργαστές πολύ μεγάλου μήκους εντολών (VLIW)

γ. στους υπερβαθμωτούς επεξεργαστές είτε η αποστολή εντολών προς εκτέλεση γίνεται με προκαθορισμένη σειρά είτε εκτός σειράς.

δ. στους υπερβαθμωτούς επεξεργαστές που η αποστολή εντολών προς εκτέλεση γίνεται με εκτός σειράς (out-of-order).

Να δικαιολογήσετε την απάντησή σας.

Λύση

Η σωστή απάντηση είναι η (γ), διότι ο μηχανισμός επαναδιάταξης αποτελεσμάτων χρησιμοποιείται αφενός στην ισχυρή συνέπεια αναφορικά με εντολές επεξεργασίας, όταν η αποστολή εντολών προς εκτέλεση γίνεται με προκαθορισμένη σειρά (in order) και αφετέρου χρησιμοποιείται και όταν η αποστολή εντολών προς εκτέλεση γίνεται εκτός σειράς, κάτι που προκαλείται λόγω των διαφορετικών καθυστερήσεων των διαφορετικών μονάδων.

Θέμα για μηχανισμό επαναδιάταξης αποτελεσμάτων

Να αναφέρετε ποιος είναι ο ρόλος του μηχανισμού επαναδιάταξης αποτελεσμάτων σε ένα υπερβαθμωτό επεξεργαστή.

Λύση

Ο μηχανισμός επαναδιάταξης αποτελεσμάτων εξασφαλίζει ότι οι εντολές ολοκληρώνονται αυστηρά με τη σειρά που εμφανίζονται στο πρόγραμμα. Οι εντολές εκτελούνται εκτός σειράς (out-of-order) αλλά ακόμα και αν τις εκτελούσαμε με τη



σειρά εμφάνισης, λόγω των διαφορετικών καθυστερήσεων των διαφόρων μονάδων (π.χ. πολλαπλασιαστής, αθροιστής) θα τελειώνουν και πάλι εκτός σειράς.

Θέμα Θεωρίας Σεπτεμβρίου 2019 με υπερβαθμωτούς επεξεργαστές

Θεωρείστε υπερβαθμωτό επεξεργαστή (superscalar processor). Η εντολή E2 που στο εκτελέσιμο πρόγραμμα έπεται της εντολής E1, μπορεί να σταλεί προς εκτέλεση **νωρίτερα** από την εντολή E1 εάν δεν υπάρχει εξάρτηση μεταξύ της E2 και της E1 ή της E2 και κάποιας/ων εκτελούμενων εντολών.

- α) Σωστό
- β) Λάθος

Λύση

Λάθος, από τη στιγμή που δεν έχει αποσαφηνιστεί ότι έχουμε σειρά αποστολής out – of – order (εκτός σειράς). Δηλαδή για να σταλθεί η E2 νωρίτερα από την E1, θα πρέπει **αφενός να μην υπάρχει εξάρτηση ανάμεσά τους** και αφετέρου να υποστηρίζεται η αποστολή εντολών εκτός σειράς (out – of – order).

Θέματα Θεωρίας Σεπτεμβρίου 2019 με υπερβαθμωτούς επεξεργαστές

Θέμα 1

Όταν ο υπερβαθμωτός επεξεργαστής ακολουθεί το μοντέλο της **ισχυρής συνέπειας** επιτρέπεται οι εντολές να ολοκληρώνονται εκτός σειράς.

- α) Σωστό
- β) Λάθος

Λύση

Στην ισχυρή συνέπεια επεξεργαστή, οι εντολές πρέπει να ολοκληρώνονται ακριβώς με τη σειρά που εμφανίζονται στο πρόγραμμα. Αυτό που αναφέρει η ερώτηση θα ήταν σωστό για την περίπτωση της **ασθενούς συνέπειας, όπου τότε οι εντολές επιτρέπεται να ολοκληρώνονται εκτός σειράς.** Αν η ερώτηση ανέφερε τη λέξη «**εκτελούνται**» θα ήταν σωστή.

Θέμα 2

Στους υπερβαθμωτούς επεξεργαστές με αποστολή εντολών προς εκτέλεση με **προκαθορισμένη** σειρά, είναι δυνατόν να εμφανιστούν **μόνο εξαρτήσεις** Εγγραφή Μετά από Ανάγνωση και Εγγραφή Μετά από Εγγραφή

- α) Σωστό
- β) Λάθος

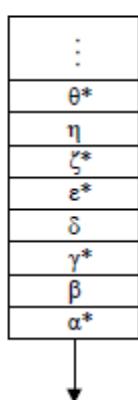
Λύση

Λάθος, διότι εμφανίζονται και AME

7.1.2. Άσκηση σχετικά με υπερβαθμωτούς επεξεργαστές

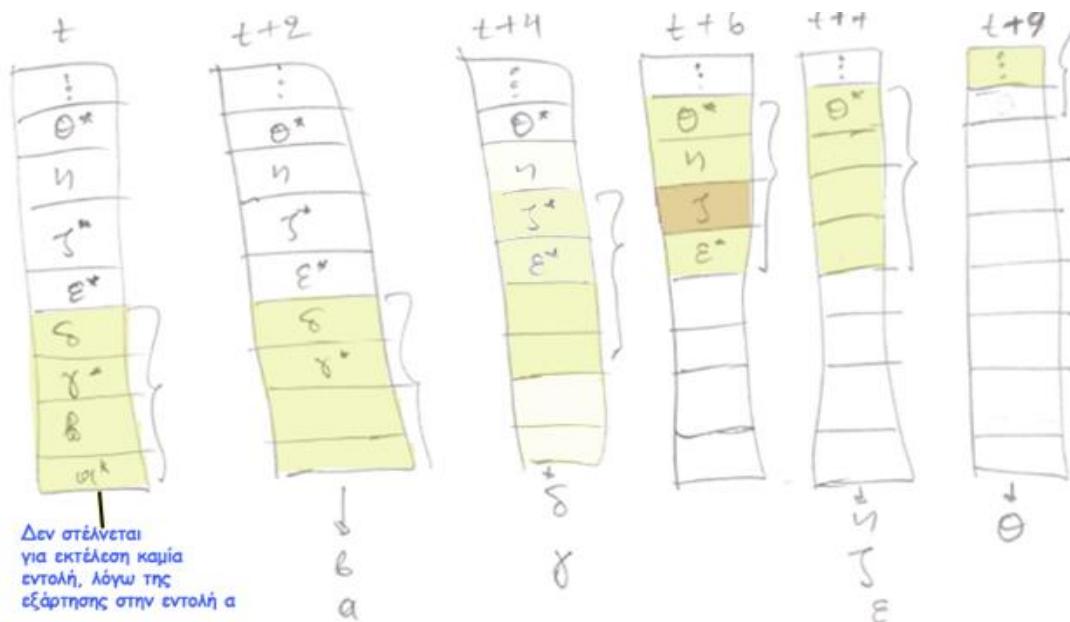
Στο παρακάτω σχήμα δίνεται το περιεχόμενο της Μονάδας Προσωρινής Αποθήκευσης Εντολών, ΜΠΑΕ, ενός υπερβαθμωτού επεξεργαστή, ο οποίος δεν διαθέτει μονάδα αναμονής αποστολής. Θεωρήστε παράθυρο τεσσάρων εντολών και ότι το σύμβολο * δηλώνει εντολή που δεν μπορεί να εκτελεστεί λόγω εξαρτήσεων.

Θεωρώντας ότι η αποστολή εντολών γίνεται με τη σειρά που εμφανίζονται οι εντολές στο πρόγραμμα (**in-order issue**), να δηλώσετε το περιεχόμενο της ΜΠΑΕ και τις εντολές που θα αποσταλούν κάθε φορά για εκτέλεση μέχρι να αδειάσει η ΜΠΑΕ, λαμβάνοντας υπόψη ότι η εντολή α παύει να είναι εξαρτημένη τη χρονική περίοδο t+2, η εντολή γ παύει να είναι εξηρτημένη 2 χρονικές περιόδους μετά την αποστολή προς εκτέλεση της εντολής β, η εντολή ζ παύει να είναι εξηρτημένη 3 χρονικές περιόδους μετά την αποστολή προς εκτέλεση της εντολής γ, η εντολή ζ παύει να είναι εξηρτημένη 2 χρονικές περιόδους μετά την αποστολή προς εκτέλεση της εντολής δ και η εντολή θ παύει να είναι εξηρτημένη 2 χρονικές περιόδους μετά την αποστολή προς εκτέλεση της εντολής ζ. Θεωρήστε ότι το παράθυρο εντολών είναι **μετατοπιζόμενο**.



Σχήμα 5.1 Τρέχουσα κατάσταση της ΜΠΑΕ.

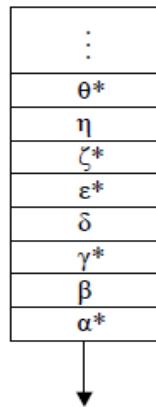
Λύση



7.1.3. Ασκηση σχετικά με υπερβαθμωτούς επεξεργαστές

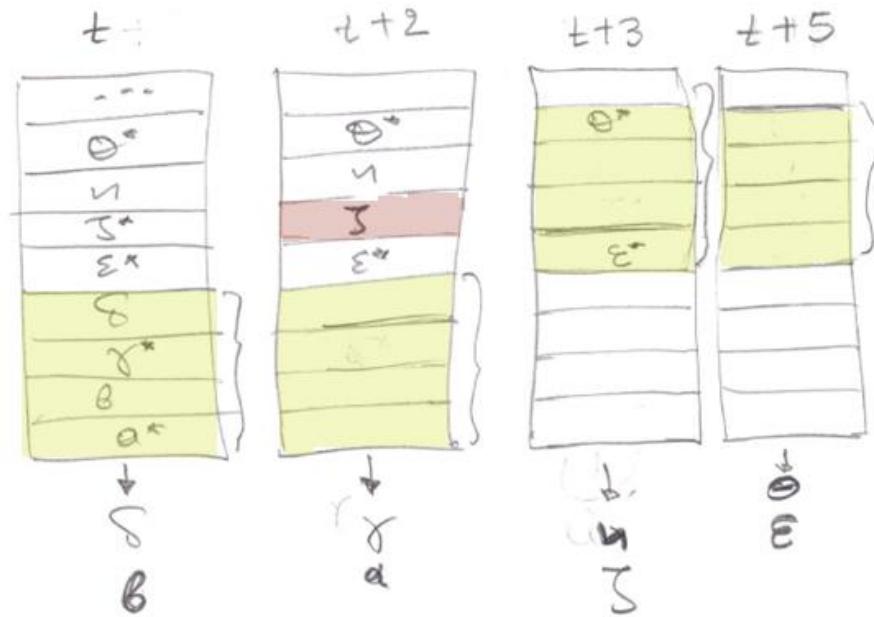
Στο σχήμα 5.1 δίνεται το περιεχόμενο της Μονάδας Προσωρινής Αποθήκευσης Εντολών, ΜΠΑΕ, ενός υπερβαθμωτού επεξεργαστή, ο οποίος δεν διαθέτει μονάδα αναμονής αποστολής. Θεωρήστε παραθύρο τεσσάρων εντολών και ότι το σύμβολο * δηλώνει εντολή που δεν μπορεί να εκτελεστεί λόγω εξαρτήσεων.

Θεωρώντας ότι η αποστολή εντολών γίνεται εκτός σειρά (**out-of-order issue**), να δηλώσετε το περιεχόμενο της ΜΠΑΕ και τις εντολές που θα αποσταλούν κάθε φορά για εκτέλεση μέχρι να αδειάσει η ΜΠΑΕ, λαμβάνοντας υπόψη ότι η εντολή α παύει να είναι εξαρτημένη τη χρονική περίοδο t+2, η εντολή γ παύει να είναι εξηρτημένη 2 χρονικές περιόδους μετά την αποστολή προς εκτέλεση της εντολής β, η εντολή ζ παύει να είναι εξηρτημένη 3 χρονικές περιόδους μετά την αποστολή προς εκτέλεση της εντολής γ, η εντολή θ παύει να είναι εξηρτημένη 2 χρονικές περιόδους μετά την αποστολή προς εκτέλεση της εντολής ζ. Θεωρήστε ότι το παράθυρο εντολών είναι **σταθερό**.



Σχήμα 5.1 Τρέχουσα κατάσταση της ΜΠΑΕ.

Λύση



7.1.4 Θέμα με υπερβαθμωτούς

Θεωρήστε ότι το τμήμα προγράμματος που ακολουθεί βρίσκεται στη Μονάδα Προσωρινής Αποθήκευσης Εντολών ενός υπερβαθμωτού (superscalar) επεξεργαστή με ρυθμό αποστολής εντολών 4, ο οποίος διαθέτει μεταξύ άλλων μία μονάδα Προσπέλασης Δεδομένων/Αποθήκευσης Αποτελεσμάτων (ΠΔ/ΑΑ) με ικανότητα χειρισμού μιας απαίτησης ανά κύκλο ρολογιού, μία Αριθμητική Λογική Μονάδα (ΑΛΜ) με καθυστέρηση ενός κύκλου ρολογιού και μία μονάδα πολλαπλασιασμού με καθυστέρηση 4 κύκλων ρολογιού. Για κάθε μία από τις ακόλουθες περιπτώσεις να συμπληρώσετε ένα πίνακα (όπως ο πίνακας 7.3.1) ώστε να φαίνεται σε κάθε κύκλο ρολογιού ποιες εντολές αποστέλλονται για εκτέλεση και σε ποια λειτουργική μονάδα.

- Η αποστολή των εντολών στις λειτουργικές μονάδες προς εκτέλεση γίνεται με προκαθορισμένη σειρά (in-order) και έχουμε μετατοπιζόμενο παράθυρο εντολών.
- Η αποστολή των εντολών στις λειτουργικές μονάδες προς εκτέλεση γίνεται εκτός σειράς (out-of-order) και έχουμε μετατοπιζόμενο παράθυρο εντολών.

Τμήμα προγράμματος

```

Load r1, a0
Load r2, b0
Mul r3, r1, r2      //r3←r1xr2
Load r4, c0
Add r5, r3, r4      //r5←r3+r4
Store r5, d0
Load r6, a1
Load r7, b1
Mul r8, r6, r7
Load r9, c1
Add r10, r8, r9
Store r10, d1
Load r11, a2
Load r12, b2
Mul r13, r11, r12
Load r14, c2
Add r15, r13, r14
Store r15, d2

```

Κύκλος ρολογιού	ΠΔ/ΑΑ	ΑΛΜ	πολλαπλασιαστής
1			
2			
.	.	.	.
.	.	.	.
.	.	.	.

Λύση

Επειδή έχουμε **μόνο μια μονάδα ΠΔ/ΑΑ** (εκτελεί αποκλειστικά εντολές LOAD/STORE), σημαίνει ότι αν στην ίδια χρονική περίοδο βρεθούν μέσα στο παράθυρο της ΜΠΑΕ για **εκτέλεση περισσότερες από μια εντολές LOAD ή STORE**, τότε θα προκληθεί **δομική εξάρτηση**. Στην **ΑΛΜ θα γίνουν μόνο οι προσθέσεις των κώδικα** (διαρκούν μια χρονική περίοδο) και όχι και οι πολλαπλασιασμοί (διαρκούν τέσσερις χρονικές περιόδους), διότι οι τελευταίοι εκτελούνται στη **μονάδα πολλαπλασιασμού**. Επίσης, εκτός από δομικές εξαρτήσεις κατά την εκτέλεση του προγράμματος, μπορούν να προκύψουν και εξαρτήσεις δεδομένων, δηλαδή μια εντολή να χρησιμοποιεί το αποτέλεσμα μιας προηγούμενης εντολής. Το ζητούμενο είναι η συμπλήρωση του πίνακα.

α. Στη συνέχεια φαίνεται η εκτέλεση των εντολών για το συνδυασμό **in – order και μετατοπιζόμενο παράθυρο εντολών**. Το * συμβολίζει εξάρτηση από δεδομένα τύπου AME, και το ♦ συμβολίζει δομική εξάρτηση. Η αποστολή για εκτέλεση των εντολών γίνεται προς τα πάνω.

κύκλος ρολογιού 1	κύκλος ρολογιού 2	κύκλος ρολογιού 3	κύκλος ρολογιού 4
Load r1, a0	Load r2, b0	Mul r3, r1, r2	Add r5, r3, r4*
Load r2, b0*	Mul r3, r1, r2*	Load r4, c0	Store r5, d0
Mul r3, r1, r2	Load r4, c0	Add r5, r3, r4*	Load r6, a1
Load r4, c0	Add r5, r3, r4	Store r5, d0	Load r7, b1

κύκλος ρολογιού 5	κύκλος ρολογιού 6	κύκλος ρολογιού 7	κύκλος ρολογιού 8
Add r5, r3, r4*	Add r5, r3, r4*	Add r5, r3, r4	Store r5, d0
Store r5, d0	Store r5, d0	Store r5, d0*	Load r6, a1*
Load r6, a1	Load r6, a1	Load r6, a1	Load r7, b1
Load r7, b1	Load r7, b1	Load r7, b1	Mul r8, r6, r7

κύκλος ρολογιού 9	κύκλος ρολογιού 10	κύκλος ρολογιού 11	κύκλος ρολογιού 12
Load r6, a1*	Load r7, b1	Mul r8, r6, r7	Add r10, r8, r9*
Load r7, b1*	Mul r8, r6, r7*	Load r9, c1	Store r10, d1
Mul r8, r6, r7	Load r9, c1	Add r10, r8, r9*	Load r11, a2
Load r9, c1	Add r10, r8, r9	Store r10, d1	Load r12, b2

κύκλος ρολογιού 13	κύκλος ρολογιού 14	κύκλος ρολογιού 15	κύκλος ρολογιού 16
Add r10, r8, r9*	Add r10, r8, r9*	Add r10, r8, r9	Store r10, d1
Store r10, d1	Store r10, d1	Store r10, d1*	Load r11, a2*
Load r11, a2	Load r11, a2	Load r11, a2	Load r12, b2
Load r12, b2	Load r12, b2	Load r12, b2	Mul r13, r11, r12

κύκλος ρολογιού 17	κύκλος ρολογιού 18	κύκλος ρολογιού 19	κύκλος ρολογιού 20
Load r11, a2	Load r12, b2	Mul r13, r11, r12	Add r15, r13, r14*
Load r12, b2*	Mul r13, r11, r12*	Load r14, c2	Store r15, d2
Mul r13, r11, r12	Load r14, c2	Add r15, r13, r14*	
Load r14, c2	Add r15, r13, r14	Store r15, d2	

κύκλος ρολογιού 21	κύκλος ρολογιού 22	Κύκλος ρολογιού 23	κύκλος ρολογιού 24
Add r15, r13, r14*	Add r15, r13, r14*	Add r15, r13, r14	Store r15, d2
Store r15, d2	Store r15, d2	Store r15, d2*	

Κάθε φορά που εκτελείται μια εντολή πολλαπλασιασμού, επειδή διαρκεί 4 χρονικές περιόδους, αυτό σημαίνει εφόσον υπάρχει εξάρτηση από δεδομένα από την εντολή πρόσθεσης που ακολουθεί, θα πρέπει να περάσουν υποχρεωτικά 4 χρονικές περίοδοι, **πριν** εκτελεστεί η κάθε εντολή πρόσθεσης. Η τελική απάντηση της άσκησης είναι ο πίνακας που ακολουθεί.

Κύκλος ρολογιού	ΠΔ/ΑΑ	ΑΛΜ	πολλαπλασιαστής
1	Load r1, a0		
2	Load r2, b0		
3	Load r4, c0		Mul r3, r1, r2
4			
5			
6			
7		Add r5, r3, r4	
8	Store r5, d0		
9	Load r6, a1		
10	Load r7, b1		
11	Load r9, c1		Mul r8, r6, r7
12			
13			
14			
15		Add r10, r8, r9	
16	Store r10, d1		
17	Load r11, a2		
18	Load r12, b2		
19	Load r14, c2		Mul r13, r11, r12
20			
21			
22			
23		Add r15, r13, r14	
24	Store r15, d2		

β. Στη συνέχεια φαίνεται η εκτέλεση των εντολών για το συνδυασμό **out - of - order** και **μετατοπιζόμενο παράθυρο εντολών**. Το * συμβολίζει και πάλι εξάρτηση από δεδομένα τύπου AME, και το ♦ συμβολίζει δομική εξάρτηση. Η αποστολή για εκτέλεση των εντολών γίνεται προς τα πάνω.

↑ κύκλος ρολογιού 1	κύκλος ρολογιού 2	κύκλος ρολογιού 3	κύκλος ρολογιού 4
Load r1, a0	Load r2, b0	Mul r3, r1, r2	Add r5, r3, r4*
Load r2, b0*	Mul r3, r1, r2*	Load r4, c0	Store r5, d0*
Mul r3, r1, r2*	Load r4, c0*	Add r5, r3, r4*	Load r6, a1
Load r4, c0*	Add r5, r3, r4*	Store r5, d0**	Load r7, b1*

κύκλος ρολογιού 5	κύκλος ρολογιού 6	κύκλος ρολογιού 7	κύκλος ρολογιού 8
Add r5, r3, r4*	Add r5, r3, r4*	Add r5, r3, r4	Store r5, d0
Store r5, d0*	Store r5, d0*	Store r5, d0*	
Load r7, b1			Mul r8, r6, r7

κύκλος ρολογιού 9	κύκλος ρολογιού 10	κύκλος ρολογιού 11	κύκλος ρολογιού 12
Load r9, c1	Add r10, r8, r9*	Add r10, r8, r9*	Add r10, r8, r9
Add r10, r8, r9*	Store r10, d1*	Store r10, d1*	Store r10, d1*
Store r10, d1**	Load r11, a2		
Load r11, a2*	Load r12, b2*	Load r12, b2	

κύκλος ρολογιού 13	κύκλος ρολογιού 14	κύκλος ρολογιού 15	κύκλος ρολογιού 16
Store r10, d1	Load r14, c2	Add r15, r13, r14*	Add r15, r13, r14*
	Add r15, r13, r14*	Store r15, d2*	Store r15, d2*
	Store r15, d2**		
Mul r13, r11, r12			

κύκλος ρολογιού 17	κύκλος ρολογιού 18
Add r15, r13, r14	Store r15, d2
Store r15, d2*	

Κύκλος ρολογιού	ΠΔ/ΑΑ	ΑΛΜ	πολλαπλασιαστής
1	Load r1, a0		
2	Load r2, b0		
3	Load r4, c0		Mul r3, r1, r2
4	Load r6, a1		
5	Load r7, b1		
6			
7		Add r5, r3, r4	
8	Store r5, d0		Mul r8, r6, r7
9	Load r9, c1		
10	Load r11, a2		
11	Load r12, b2		
12		Add r10, r8, r9	
13	Store r10, d1		Mul r13, r11, r12
14	Load r14, c2		
15			
16			
17		Add r15, r13, r14	
18	Store r15, d2		

Παρατηρούμε ότι στη δεύτερη περίπτωση χρειάζονται λιγότερες χρονικές περίοδοι.

Θέμα Σεπτεμβρίου 2021 με υπερβαθμωτούς επεξεργαστές

Θεωρείστε ότι το τμήμα προγράμματος που ακολουθεί, ανήκει σε πρόγραμμα που εκτελείται σε υπερβαθμωτό (superscalar) επεξεργαστή με **ρυθμό αποστολής εντολών προς εκτέλεση 4**.

1. NOR r1, r7, r8 // $r1 \leftarrow \text{NOR}(r7, r8)$
2. LOAD r0, (r9) // $r0 \leftarrow M(r9)$
3. ADD r3, r1, r19 // $r3 \leftarrow r1+r10$
4. SUB r4, r1, r2 // $r4 \leftarrow r1-r2$
5. ADD r6, r4, r5 // $r6 \leftarrow r4+r5$
6. ADD r3, r8, r2 // $r3 \leftarrow r8+r2$
7. MUL r7, r5, r2 // $r7 \leftarrow r5 * r2$
8. STORE r1, (r6) // $r1 \rightarrow M(r6)$

Να απαντήσετε εάν η ακόλουθη πρόταση είναι σωστή ή λάθος: Σε επεξεργαστή που ισχύει το μοντέλο της **ασθενούς** σειριακής συνέπειας στην εκτέλεση των εντολών, η υπ' αριθμόν 7 εντολή **δεν** μπορεί να **ολοκληρωθεί** πριν την υπ' αριθμόν 6 εντολή.

- α) Σωστό
- β) Λάθος

Λύση

Η πρόταση είναι λάθος, αφού ο ρυθμός αποστολής εντολών προς εκτέλεση που είναι ίσος με 4, σημαίνει ότι κάθε φορά μπορούν να σταλθούν για εκτέλεση 4 εντολές συγχρόνως ή με άλλα λόγια το παράθυρο εντολών της ΜΠΑΕ αποτελείται από «4» εντολές, εφόσον δεν υπάρχουν εξαρτήσεις ανάμεσά τους. Το παράθυρο των τεσσάρων πρώτων εντολών στέλνεται για εκτέλεση, αφού δεν υπάρχει καμία εξάρτηση μεταξύ των εντολών του. Στο δεύτερο παράθυρο των 4 επόμενων εντολών, υπάρχει εξάρτηση μεταξύ των εντολών «4» και «5» λόγω του καταχωρητή R4 και έτσι η εντολή «5» δεν μπορεί να σταλθεί



για εκτέλεση στις λειτουργικές μονάδες του υπερβαθμωτού. Το γεγονός όμως ότι η σειρά αποστολής εντολών είναι εκτός σειράς (out of order), σημαίνει ότι οι εντολές «6», «7» και «8» στέλνονται κανονικά για εκτέλεση και η εντολή «7» μπορεί να ολοκληρωθεί πριν την «6», αφού δεν υπάρχει κάποια εξάρτηση ανάμεσά τους. Επομένως η πρόταση είναι **λάθος**.

Θέμα Σεπτεμβρίου 2021 με υπερβαθμωτούς επεξεργαστές

Θεωρήστε ότι το τμήμα προγράμματος που ακολουθεί ανήκει σε πρόγραμμα που εκτελείται σε υπερβαθμωτό (superscalar) επεξεργαστή με ρυθμό αποστολής εντολών προς εκτέλεση 5.

1. ADD r3, r1, r6 // r3 \leftarrow r1+r6
2. SUB r4, r1, r2 // r4 \leftarrow r1-r2
3. XOR r6, r4, r5 // r6 \leftarrow XOR(r4, r5)
4. ADD r3, r8, r2 // r3 \leftarrow r8+r2
5. LOAD r0, (r9) // r0 \leftarrow M(r9)
6. DIV r7, r5, r2 // r7 \leftarrow r5/r2
7. STORE r1, (r0) // r1 \rightarrow M(r0)

Να απαντήσετε εάν η ακόλουθη πρόταση είναι σωστή ή λάθος: Σε επεξεργαστή που ισχύει το μοντέλο της ισχυρής σειριακής συνέπειας στην εκτέλεση των εντολών, η υπ' αριθμόν 4 εντολή **μπορεί** να ολοκληρωθεί πριν την υπ' αριθμόν 3 εντολή.

α) Σωστό

β) Λάθος

Λύση

Η πρόταση εξ' ορισμού είναι **λάθος** αφού στην ισχυρή σειριακή συνέπεια **καμία εντολή δεν μπορεί να ολοκληρωθεί** πριν την προηγούμενή της (μπορεί όμως να εκτελεστεί, αν δεν υπάρχει κάποια εξάρτηση ανάμεσά τους).

Θέματα Σεπτεμβρίου 2021 με υπερβαθμωτούς επεξεργαστές

Θέμα 1

Θεωρείστε ότι το τμήμα του προγράμματος που ακολουθεί ανήκει σε πρόγραμμα που εκτελείται σε υπερβαθμωτό επεξεργαστή με ρυθμό αποστολής εντολών προς εκτέλεση 3.

1. ADD r3, r1, r2 // r3 \leftarrow r1+r2
2. ADD r4, r1, r2 // r4 \leftarrow r1+r2
3. MUL r6, r4, r5 // r6 \leftarrow r4*r5
4. DIV r7, r5, r2 // r7 \leftarrow r5/r2
5. LOAD r0, (r7) // r0 \leftarrow M(r7)
6. STORE r0, (r9) // r0 \rightarrow M(r9)

Να απαντήσετε εάν η ακόλουθη πρόταση είναι σωστή ή λάθος.

Σε επεξεργαστή που ισχύει το μοντέλο της **ασθενούς σειριακής συνέπειας** στην εκτέλεση των εντολών, η υπ' αριθμόν 6 εντολή **δεν μπορεί να ολοκληρωθεί** πριν την υπ' αριθμόν 4 εντολή.

Λύση

Σωστό. Οι εντολές 4, 6 βρίσκονται στο ίδιο παράθυρο, επομένως μπορούν να συγκριθούν μεταξύ τους. Η εντολή υπ' αριθμόν 6 έχει εξάρτηση από την εντολή υπ' αριθμόν 5, λόγω του r0, και η 5^η εντολή έχει εξάρτηση από την 4^η εντολή, λόγω του r7. Επομένως, δεν μπορεί η 6^η εντολή να ολοκληρωθεί πριν την 4^η εντολή.

Θέμα 2

Θεωρείστε ότι το τμήμα προγράμματος που ακολουθεί ανήκει σε πρόγραμμα που εκτελείται σε υπερβαθμωτό (superscalar) επεξεργαστή με ρυθμό αποστολής εντολών προς εκτέλεση 3.

1. ADD r3, r1, r2 // r3 \leftarrow r1+r2
2. SUB r4, r1, r2 // r4 \leftarrow r1-r2
3. NOR r6, r4, r5 // r6 \leftarrow NOR(r4, r5)
4. ADD r3, r8, r2 // r3 \leftarrow r8+r2
5. DIV r7, r5, r2 // r7 \leftarrow r5/r2
6. LOAD r0, (r7) // r0 \leftarrow M(r7)

7. STORE r0, (r9) // r0 → M(r9)

Να απαντήσετε εάν η ακόλουθη πρόταση είναι σωστή ή λάθος: Σε επεξεργαστή που ισχύει το μοντέλο της ασθενούς σειριακής συνέπειας στην εκτέλεση των εντολών, η υπ' αριθμόν 3 εντολή **δεν** μπορεί να ολοκληρωθεί πριν την υπ' αριθμόν 2 εντολή.

- α) Σωστό
- β) Λάθος

Λύση

Η πρόταση είναι **σωστή**, γιατί παρά το γεγονός ότι υπάρχει ασθενής σειριακή συνέπεια, υπάρχει εξάρτηση ανάμεσα στις εντολές «2» και «3», λόγω του καταχωρητή r4, και ως εκ' τούτου η «3» δεν μπορεί να ολοκληρωθεί (αλλά και ούτε να εκτελεστεί) πριν τη «2».

Θέμα 3

Θεωρείστε ότι το τμήμα προγράμματος που ακολουθεί ανήκει σε πρόγραμμα που εκτελείται σε υπερβαθμωτό (superscalar) επεξεργαστή με ρυθμό αποστολής εντολών προς εκτέλεση 6.

1. SUB r4, r1, r2 // $r4 \leftarrow r1 - r2$
2. NOR r6, r4, r5 // $r6 \leftarrow \text{NOR}(r4, r5)$
3. ADD r3, r8, r2 // $r3 \leftarrow r8 + r2$
4. DIV r7, r5, r3 // $r7 \leftarrow r5/r3$
5. LOAD r0, (r7) // $r0 \leftarrow M(r7)$
6. STORE r0, (r9) // $r0 \rightarrow M(r9)$
7. ADD r3, r1, r0 // $r3 \leftarrow r1 + r0$

Να απαντήσετε εάν η ακόλουθη πρόταση είναι σωστή ή λάθος.

Σε επεξεργαστή που ισχύει το μοντέλο της ισχυρής σειριακής συνέπειας στην εκτέλεση των εντολών, η υπ' αριθμόν 7 εντολή δεν μπορεί να εκτελεστεί πριν την υπ' αριθμόν 6 εντολή.

- α) Σωστό
- β) Λάθος

Λύση

Η πρόταση είναι **σωστή**, γιατί έχουμε παράθυρο 6 εντολών και η εντολή υπ' αριθμόν 7 δεν μπορεί να εκτελεστεί πριν την «6», αφού βρίσκεται εκτός παραθύρου.

Θέμα 4

Θεωρείστε ότι το τμήμα προγράμματος που ακολουθεί ανήκει σε πρόγραμμα που εκτελείται σε υπερβαθμωτό (superscalar) επεξεργαστή με ρυθμό αποστολής εντολών προς εκτέλεση 5.

1. SUB r3, r1, r6 // $r3 \leftarrow r1 - r6$
2. ADD r4, r1, r2 // $r4 \leftarrow r1 + r2$
3. MUL r6, r4, r5 // $r6 \leftarrow r4 * r5$
4. ADD r3, r8, r2 // $r3 \leftarrow r8 + r2$
5. LOAD r0, (r9) // $r0 \leftarrow M(r9)$
6. MUL r7, r5, r2 // $r7 \leftarrow r5 * r2$
7. STORE r1, (r0) // $r1 \rightarrow M(r0)$

Να απαντήσετε εάν η ακόλουθη πρόταση είναι σωστή ή λάθος.

Σε επεξεργαστή που ισχύει το μοντέλο της ασθενούς σειριακής συνέπειας στην εκτέλεση των εντολών, η υπ' αριθμόν 5 εντολή **δεν** μπορεί να ολοκληρωθεί πριν την υπ' αριθμόν 3 εντολή.

- α) Σωστό
- β) Λάθος

Λύση

Λάθος, διότι καταρχήν οι δύο εντολές υπ' αριθμόν 3 και 5 βρίσκονται στο ίδιο παράθυρο, επομένως μπορούν να συγκριθούν μεταξύ τους και από τη στιγμή που έχουμε ασθενή σειριακή συνέπεια, η εντολή υπ' αριθμόν 5, μπορεί να ολοκληρωθεί

πριν την εντολή υπ' αριθμόν 3, λόγω του ότι δεν υπάρχει εξάρτηση ανάμεσά τους και επιπλέον λόγω της ασθενούς σειριακής συνέπειας στην εκτέλεση εντολών.

Θέμα 5

Θεωρείστε ότι το τμήμα προγράμματος που ακολουθεί ανήκει σε πρόγραμμα που εκτελείται σε υπερβαθμωτό (superscalar) επεξεργαστή με ρυθμό αποστολής εντολών προς εκτέλεση 3.

1. ADD r3, r1, r2 // $r3 \leftarrow r1+r2$
2. SUB r4, r1, r2 // $r4 \leftarrow r1-r2$
3. NOR r6, r4, r5 // $r6 \leftarrow \text{NOR}(r4, r5)$
4. ADD r3, r8, r2 // $r3 \leftarrow r8+r2$
5. DIV r7, r5, r2 // $r7 \leftarrow r5/r2$
6. LOAD r0, (r7) // $r0 \rightarrow M(r7)$
7. STORE r0, (r9) // $r0 \leftarrow M(r9)$

Να απαντήσετε εάν η ακόλουθη πρόταση είναι σωστή ή λάθος.

Σε επεξεργαστή που ισχύει το μοντέλο της ασθενούς σειριακής συνέπειας στην εκτέλεση των εντολών, η υπ' αριθμόν 6 εντολή δεν μπορεί να ολοκληρωθεί πριν την υπ' αριθμόν 4 εντολή.

α) Σωστό

β) Λάθος

Λύση

Η πρόταση είναι λάθος, διότι στο δεύτερο παράθυρο αποστολής εντολών η εντολή 5 μπορεί λόγω ασθενούς σειριακής συνέπειας στην εκτέλεση των εντολών, να εκτελεστεί πριν την εντολή 4 (αφού δεν έχουν εξάρτηση μεταξύ τους) και κατά συνέπεια και η εντολή 6 που εξαρτάται από την 5, να εκτελεστεί και αυτή πριν την 4.

Θέμα 6

Θεωρείστε ότι το τμήμα προγράμματος που ακολουθεί ανήκει σε πρόγραμμα που εκτελείται σε υπερβαθμωτό (superscalar) επεξεργαστή με ρυθμό αποστολής εντολών προς εκτέλεση 3.

1. ADD r3, r1, r2 // $r3 \leftarrow r1+r2$
2. ADD r4, r1, r2 // $r4 \leftarrow r1+r2$
3. MUL r6, r4, r5 // $r6 \leftarrow r4 * r5$
4. MUL r3, r8, r2 // $r3 \leftarrow r8 * r2$
5. DIV r7, r5, r2 // $r7 \leftarrow r5/r2$
6. LOAD r0, (r7) // $r0 \leftarrow M(r7)$
7. STORE r0, (r9) // $r0 \rightarrow M(r9)$

Να απαντήσετε εάν η ακόλουθη πρόταση είναι σωστή ή λάθος: Σε επεξεργαστή που ισχύει το μοντέλο της ασθενούς σειριακής συνέπειας στην εκτέλεση των εντολών, η υπ. αριθμόν 6 εντολή δεν μπορεί να ολοκληρωθεί πριν την υπ. αριθμόν 4 εντολή.

α. Σωστό

β. Λάθος

Λύση

Λάθος. Οι εντολές 4, 6 βρίσκονται στο ίδιο παράθυρο, επομένως μπορούν να συγκριθούν μεταξύ τους. Η εντολή υπ' αριθμόν 6 έχει εξάρτηση από την εντολή υπ' αριθμόν 5, όμως η 5^η εντολή μπορεί να ολοκληρωθεί πριν την 4^η εντολή λόγω ασθενούς σειριακής συνέπειας. Επομένως και η 6^η εντολή που εξαρτάται από την 5^η, μπορεί να εκτελεστεί πριν την 4^η εντολή.

Θέμα 7

Θεωρείστε ότι το τμήμα προγράμματος που ακολουθεί ανήκει σε πρόγραμμα που εκτελείται σε υπερβαθμωτό (superscalar) επεξεργαστή με ρυθμό αποστολής εντολών προς εκτέλεση 5.

1. SUB r3, r1, r6 / r3 <- r1-r6
2. ADD r4, r1, r2 / r4 <- r1+r2
3. MUL r6, r4, r5 / r6 <- r4*r5
4. ADD r3, r8, r2 / r3 <- r8+r2
5. LOAD r0, (r9) / r0 <- M(r9)
6. MUL r7, r5, r2 / r7 <- r5*r2
7. STORE r1, (r0) / r1 --> M(r0)

Να απαντήσετε εάν η ακόλουθη πρόταση είναι σωστή ή λάθος.

Σε επεξεργαστή που ισχύει το μοντέλο της ασθενούς σειριακής συνέπειας στην εκτέλεση των εντολών, η υπ' αριθμόν 5 εντολή δεν μπορεί να ολοκληρωθεί πριν την υπ' αριθμόν 3 εντολή.

α. Σωστό

β. Λάθος

Λύση

Λάθος, διότι καταρχήν οι δύο εντολές υπ' αριθμόν 3 και 5 βρίσκονται στο ίδιο παράθυρο, επομένως μπορούν να συγκριθούν μεταξύ τους και από τη στιγμή που έχουμε ασθενή σειριακή συνέπεια, η εντολή υπ' αριθμόν 5, μπορεί να ολοκληρωθεί πριν την εντολή υπ' αριθμόν 3, λόγω του ότι δεν υπάρχει εξάρτηση ανάμεσά τους και επιπλέον λόγω της ασθενούς σειριακής συνέπειας στην εκτέλεση εντολών.

Θέμα 8

Θεωρείστε ότι το τμήμα προγράμματος που ακολουθεί ανήκει σε πρόγραμμα που εκτελείται σε υπερβαθμωτό (superscalar) επεξεργαστή με ρυθμό αποστολής εντολών προς εκτέλεση 6.

1. SUB r4, r1, r2 // r4 <- r1-r2
2. NOR r6, r4, r5 // r6 <- NOR(r4,r5)
3. ADD r3, r8, r2 // r3 <- r8+r2
4. DIV r7, r5, r3 // r7 <- r5/r3
5. LOAD r0, (r7) // r0 <- M(r7)
6. STORE r0, (r9) // r0 --> M(r9)
7. ADD r3, r1, r0 // r1 <- r1+r0

Να απαντήσετε εάν η ακόλουθη πρόταση είναι σωστή ή λάθος: Σε επεξεργαστή που ισχύει το μοντέλο της ισχυρής σειριακής συνέπειας στην εκτέλεση των εντολών, η υπ' αριθμόν 6 εντολή δεν μπορεί να εκτελεστεί πριν την υπ' αριθμόν 4 εντολή.

α. Σωστό

β. Λάθος

Λύση

Σωστό, διότι καταρχήν η εντολή υπ' αριθμόν 6 έχει εξάρτηση από την εντολή υπ' αριθμόν 5 και αυτή με τη σειρά της έχει εξάρτηση από την εντολή υπ' αριθμόν 4, ενώ όλες περιέχονται στο ίδιο παράθυρο, επομένως μπορούν να συγκριθούν μεταξύ τους. Ακόμη και αν υπήρχε υπάρχει ασθενής σειριακής συνέπεια στην εκτέλεση εντολών, μια εντολή δεν μπορεί να εκτελεστεί πριν από μια άλλη από τη στιγμή που υπάρχει εξάρτηση ανάμεσά τους.

Θέματα Σεπτεμβρίου 2020 με υπερβαθμωτούς επεξεργαστές

Θέμα 1

Θεωρείστε ότι το τμήμα προγράμματος που ακολουθεί ανήκει σε πρόγραμμα που εκτελείται σε υπερβαθμωτό (superscalar) επεξεργαστή με ρυθμό αποστολής εντολών προς εκτέλεση 6.

1. SUB r4, r1, r2 // $r4 \leftarrow r1 - r2$
2. NOR r6, r4, r5 // $r6 \leftarrow \text{NOR}(r4, r5)$
3. ADD r3, r8, r2 // $r3 \leftarrow r8 + r2$
4. DIV r7, r5, r3 // $r7 \leftarrow r5 / r3$
5. LOAD r0, (r7) // $r0 \leftarrow M(r7)$
6. STORE r0, (r9) // $r0 \rightarrow M(r9)$
7. ADD r3, r1, r0 // $r3 \leftarrow r1 + r0$

Να απαντήσετε εάν η ακόλουθη πρόταση είναι σωστή ή λάθος:

Σε επεξεργαστή που ισχύει το μοντέλο της ισχυρής σειριακής συνέπειας στην εκτέλεση των εντολών, η υπ' αριθμόν 7 εντολή δεν μπορεί να εκτελεστεί πριν την υπ' αριθμόν 6 εντολή.

α) Σωστό

β) Λάθος

Λύση

Σωστό. Από τη στιγμή που είναι εκτός παραθύρου, δεν μπορεί να εκτελεστεί πριν την 6η

Θέμα 2

Θεωρείστε ότι το τμήμα προγράμματος που ακολουθεί ανήκει σε πρόγραμμα που εκτελείται σε υπερβαθμωτό (superscalar) επεξεργαστή με ρυθμό αποστολής εντολών προς εκτέλεση 3.

1. ADD r3, r1, r2 // $r3 \leftarrow r1 + r2$
2. SUB r4, r1, r2 // $r4 \leftarrow r1 - r2$
3. NOR r6, r4, r5 // $r6 \leftarrow \text{NOR}(r4, r5)$
4. ADD r3, r8, r2 // $r3 \leftarrow r8 + r2$
5. DIV r7, r5, r2 // $r7 \leftarrow r5 / r2$
6. LOAD r0, (r7) // $r0 \leftarrow M(r7)$
7. STORE r0, (r9) // $r0 \rightarrow M(r9)$

Να απαντήσετε εάν η ακόλουθη πρόταση είναι σωστή ή λάθος: Σε επεξεργαστή που ισχύει το μοντέλο της ασθενούς σειριακής συνέπειας στην εκτέλεση των εντολών, η υπ' αριθμόν 4 εντολή μπορεί να ολοκληρωθεί πριν την υπ' αριθμόν 3 εντολή.

α) Σωστό

β) Λάθος

Λύση

Λάθος. Από τη στιγμή που είναι εκτός παραθύρου, δεν μπορεί η υπ' αριθμόν 4 εντολή να ολοκληρωθεί αλλά ούτε και να εκτελεστεί πριν την υπ' αριθμόν 3 εντολή.

Θέματα Φεβρουαρίου 2021 με υπερβαθμωτούς επεξεργαστές

Θέμα 1

Θεωρείστε ότι το τμήμα προγράμματος που ακολουθεί ανήκει σε πρόγραμμα που εκτελείται σε υπερβαθμωτό (superscalar) επεξεργαστή με ρυθμό αποστολής εντολών προς εκτέλεση 4.

1. NOR r1, r7, r8 // $r1 \leftarrow \text{NOR}(r7, r8)$
2. LOAD r0, (r9) // $r0 \leftarrow M(r9)$
3. ADD r3, r1, r10 // $r3 \leftarrow r1 + r10$
4. SUB r4, r1, r2 // $r4 \leftarrow r1 - r2$
5. ADD r6, r4, r5 // $r6 \leftarrow r4 + r5$
6. ADD r3, r8, r2 // $r3 \leftarrow r8 + r2$
7. MUL r7, r5, r2 // $r7 \leftarrow r5 * r2$
8. STORE r1, (r6) // $r1 \rightarrow M(r6)$

Να απαντήσετε εάν η ακόλουθη πρόταση είναι σωστή ή λάθος: Σε επεξεργαστή που ισχύει το μοντέλο της ασθενούς σειριακής συνέπειας στην εκτέλεση των εντολών, η υπ' αριθμόν 4 εντολή δεν μπορεί να εκτελεστεί πριν την υπ' αριθμόν 3 εντολή.

α) Σωστό

β) Λάθος

Λύση

Λάθος, διότι από τη στιγμή που έχουμε **ασθενή σειριακή συνέπεια** και η εντολή 4 δεν έχει κάποια εξάρτηση από την εντολή 3, και παράλληλα βρίσκεται μέσα στο παράθυρο των 4 εντολών μαζί με την 3, μπορεί να σταλθεί για εκτέλεση στις λειτουργικές μονάδες του υπερβαθμωτού.

Θέμα 2

Θεωρείστε ότι το τμήμα προγράμματος που ακολουθεί ανήκει σε πρόγραμμα που εκτελείται σε υπερβαθμωτό (superscalar) επεξεργαστή με ρυθμό αποστολής εντολών προς εκτέλεση 6.

1. SUB r4, r1, r2 // $r4 \leftarrow r1 - r2$
2. NOR r6, r4, r5 // $r6 \leftarrow \text{NOR}(r4, r5)$
3. ADD r3, r8, r2 // $r3 \leftarrow r8 + r2$
4. DIV r7, r5, r3 // $r7 \leftarrow r5 / r3$
5. LOAD r0, (r7) // $r0 \leftarrow M(r7)$
6. STORE r0, (r9) // $r0 \rightarrow M(r9)$
7. ADD r3, r1, r0 // $r3 \leftarrow r1 + r0$

Να απαντήσετε εάν η ακόλουθη πρόταση είναι σωστή ή λάθος.

Σε επεξεργαστή που ισχύει το μοντέλο της **ισχυρής σειριακής συνέπειας** στην εκτέλεση των εντολών, η υπ' αριθμόν 4 εντολή δεν μπορεί να εκτελεστεί πριν την υπ' αριθμόν 3 εντολή.

α) Σωστό

β) Λάθος

Αύση

Σωστό, θα μπορούσε να συμβεί αυτό που αναφέρει η εκφώνηση μόνο αν οι εντολές 3 και 4 **δεν** είχαν εξάρτηση.

Θέμα 3

Θεωρείστε ότι το τμήμα προγράμματος που ακολουθεί ανήκει σε πρόγραμμα που εκτελείται σε υπερβαθμωτό (superscalar) επεξεργαστή με ρυθμό αποστολής εντολών προς εκτέλεση 4.

1. NOR r1, r7, r8 // $r1 \leftarrow \text{NOR}(r7, r8)$
2. LOAD r0, (r9) // $r0 \leftarrow M(r9)$
3. ADD r3, r1, r10 // $r3 \leftarrow r1 + r10$
4. SUB r4, r1, r2 // $r4 \leftarrow r1 - r2$
5. ADD r6, r4, r5 // $r6 \leftarrow r4 + r5$
6. ADD r3, r8, r2 // $r3 \leftarrow r8 + r2$
7. MUL r7, r5, r2 // $r7 \leftarrow r5 * r2$
8. STORE r1, (r6) // $r1 \rightarrow M(r6)$

Να απαντήσετε εάν η ακόλουθη πρόταση είναι σωστή ή λάθος: Σε επεξεργαστή που ισχύει το μοντέλο της ισχυρής σειριακής συνέπειας στην εκτέλεση των εντολών, η υπ' αριθμόν 4 εντολή μπορεί να **ολοκληρωθεί** πριν την υπ' αριθμόν 3 εντολή.

α) Σωστό

β) Λάθος

Αύση

Λάθος, λόγω ισχυρής σειριακής συνέπειας, όπου μεταξύ άλλων αναφέρεται και ο όρος «ολοκληρωθεί». Αν η εκφώνηση ανέφερε τη λέξη «εκτελεστεί» θα ήταν **σωστή**, από τη στιγμή που δεν υπάρχει εξάρτηση μεταξύ των εντολών «3» και «4».

Θέμα Σεπτεμβρίου 2020 με υπερβαθμωτούς επεξεργαστές

Θεωρήστε ότι το τμήμα προγράμματος που ακολουθεί ανήκει σε πρόγραμμα που εκτελείται σε υπερβαθμωτό (superscalar) επεξεργαστή με ρυθμό αποστολής εντολών προς εκτέλεση 5.

1. ADD r3, r1, r6 // $r3 \leftarrow r1 + r6$
2. SUB r4, r1, r2 // $r4 \leftarrow r1 - r2$
3. XOR r6, r4, r5 // $r6 \leftarrow \text{XOR}(r4, r5)$
4. ADD r3, r8, r2 // $r3 \leftarrow r8 + r2$
5. LOAD r0, (r9) // $r0 \leftarrow M(r9)$
6. DIV r8, r5, r2 // $r7 \leftarrow r5 / r2$
7. STORE r1, (r0) // $r1 \rightarrow M(r0)$

Να απαντήσετε εάν η ακόλουθη πρόταση είναι σωστή ή λάθος: Σε επεξεργαστή που ισχύει το μοντέλο της ισχυρής σειριακής συνέπειας στην εκτέλεση των εντολών, η υπ' αριθμόν 4 εντολή μπορεί να ολοκληρωθεί πριν την υπ' αριθμόν 3 εντολή.

α) Σωστό

β) Λάθος

Λύση

Λάθος. Η εντολή «4» δεν μπορεί να ολοκληρωθεί πριν την «3», διότι ισχύει το μοντέλο ισχυρής **σειριακής** συνέπειας.

Σημείωση: Ο ρυθμός αποστολής εντολών προς εκτέλεση που είναι ίσος με 5, σημαίνει ότι κάθε φορά μπορούν να σταλθούν για εκτέλεση 5 εντολές συγχρόνως ή με άλλα λόγια το παράθυρο εντολών της ΜΠΑΕ αποτελείται από «5» εντολές

Σημείωση: Αν έλεγε την έκφραση «μπορεί να εκτελεστεί πριν την υπ' αριθμόν 3 εντολή», τότε η πρόταση θα ήταν **σωστή**.

Θέμα με ασθενή/ισχυρή συνέπεια επεξεργαστή

Θεωρείστε ότι το διπλανό τμήμα προγράμματος ανήκει σε πρόγραμμα που εκτελείται σε υπερβαθμωτό (superscalar) επεξεργαστή με ρυθμό αποστολής εντολών προς εκτέλεση 4.

Να κυκλώσετε τις σωστές προτάσεις. Για κάθε λάθος επιλογή θα μηδενίζεται μια σωστή. Δεν χρειάζεται αιτιολόγηση.

1. LOAD r1, (r7) //r1←M(r7)
2. LOAD r0, (r9) //r0←M(r9)
3. ADD r3, r1, r2 //r3←r1+r2
4. SUB r4, r1, r2 //r4←r1-r2
5. ADD r6, r4, r5 //r6←r4+r5
6. ADD r3, r8, r2 //r3←r8+r2
7. MUL r7, r5, r2 //r7←r5*r2

1. Σε επεξεργαστή που ισχύει το μοντέλο της **ισχυρής σειριακής συνέπειας** στην εκτέλεση των εντολών η υπ' αριθμόν 5 εντολή μπορεί να εκτελεστεί πριν την υπ' αριθμόν 4 εντολή. Λ

2. Σε επεξεργαστή που ισχύει το μοντέλο της **ασθενούς σειριακής συνέπειας** στην εκτέλεση των εντολών η υπ' αριθμόν 4 εντολή δεν μπορεί να ολοκληρωθεί πριν την υπ' αριθμόν 3 εντολή. Λ

3. Σε επεξεργαστή που ισχύει το μοντέλο της **ασθενούς σειριακής συνέπειας** στην εκτέλεση των εντολών, η υπ' αριθμόν 5 εντολή δεν μπορεί να εκτελεστεί πριν την υπ' αριθμόν 4 εντολή. Σ

4. Σε επεξεργαστή που ισχύει το μοντέλο της **ισχυρής σειριακής συνέπειας** στην εκτέλεση των εντολών η υπ' αριθμόν 4 εντολή μπορεί να ολοκληρωθεί πριν την υπ' αριθμόν 3 εντολή. Λ (Αν έγραψε εκτελεστεί, θα ήταν σωστό)

5. Σε επεξεργαστή που ισχύει το μοντέλο της **ισχυρής σειριακής συνέπειας** στην εκτέλεση των εντολών η υπ' αριθμόν 5 εντολή δεν μπορεί να ολοκληρωθεί πριν την υπ' αριθμόν 4 εντολή. Σ (λόγω εξάρτησης δεν μπορεί να εκτελεστεί)

6. Σε επεξεργαστή που ισχύει το μοντέλο της **ασθενούς σειριακής συνέπειας** στην εκτέλεση των εντολών η υπ' αριθμόν 5 εντολή μπορεί να ολοκληρωθεί πριν την υπ' αριθμόν 4 εντολή. Λ

7. Σε επεξεργαστή που ισχύει το μοντέλο της **ισχυρής σειριακής συνέπειας** στην εκτέλεση των εντολών η υπ' αριθμόν 5 εντολή μπορεί να ολοκληρωθεί πριν την υπ' αριθμόν 4 εντολή. Λ

8. Σε επεξεργαστή που ισχύει το μοντέλο της **ισχυρής σειριακής συνέπειας** στην εκτέλεση των εντολών η υπ' αριθμόν 5 εντολή δεν μπορεί να εκτελεστεί πριν την υπ' αριθμόν 4 εντολή. Σ (λόγω εξάρτησης δεν μπορεί να εκτελεστεί)

9. Σε επεξεργαστή που ισχύει το μοντέλο της **ασθενούς σειριακής συνέπειας** στην εκτέλεση των εντολών η υπ' αριθμόν 5 εντολή μπορεί να εκτελεστεί πριν την υπ' αριθμόν 4 εντολή. Λ (έξω από το μπλοκ).

10. Σε επεξεργαστή που ισχύει το μοντέλο της **ασθενούς σειριακής συνέπειας** στην εκτέλεση των εντολών η υπ' αριθμόν 4 εντολή **μπορεί να ολοκληρωθεί** πριν την υπ' αριθμόν 3 εντολή. Σ (επίσης μπορεί και να εκτελεστεί πριν την 3).

11. Σε επεξεργαστή που ισχύει το μοντέλο της **ασθενούς σειριακής συνέπειας** στην εκτέλεση των εντολών η υπ' αριθμόν 4 εντολή μπορεί να εκτελεστεί πριν την υπ' αριθμόν 3 εντολή. Σ

12. Σε επεξεργαστή που ισχύει το μοντέλο της **ασθενούς σειριακής συνέπειας** στην εκτέλεση των εντολών η υπ' αριθμόν 5 εντολή δεν μπορεί να ολοκληρωθεί πριν την υπ' αριθμόν 4 εντολή. Σ



13. Σε επεξεργαστή που ισχύει το μοντέλο της **ασθενούς σειριακής συνέπειας** στην εκτέλεση των εντολών η υπ' αριθμόν 4 εντολή δεν μπορεί να εκτελεστεί πριν την υπ' αριθμόν 3 εντολή. Λ
14. Σε επεξεργαστή που ισχύει το μοντέλο της **ισχυρής σειριακής συνέπειας** στην εκτέλεση των εντολών η υπ' αριθμόν 4 εντολή δεν μπορεί να ολοκληρωθεί πριν την υπ' αριθμόν 3 εντολή. Σ (μπορεί όμως να εκτελεστεί)
15. Σε επεξεργαστή που ισχύει το μοντέλο της **ισχυρής σειριακής συνέπειας** στην εκτέλεση των εντολών η υπ' αριθμόν 4 εντολή μπορεί να εκτελεστεί πριν την υπ' αριθμόν 3 εντολή. Σ
16. Σε επεξεργαστή που ισχύει το μοντέλο της **ισχυρής σειριακής συνέπειας** στην εκτέλεση των εντολών η υπ' αριθμόν 4 εντολή δεν μπορεί να εκτελεστεί πριν την υπ' αριθμόν 3 εντολή. Λ

Λύση

Σωστά 3, 5, 8, 10, 11, 12, 14, 15

Συμπέρασμα: όταν ισχύει το μοντέλο της ασθενούς σειριακής συνέπειας στην εκτέλεση των εντολών, εφόσον δεν υπάρχουν εξαρτήσεις, μια εντολή μπορεί και να εκτελεστεί αλλά και να ολοκληρωθεί πριν μια άλλη. **Όταν ισχύει το μοντέλο της ισχυρής σειριακής συνέπειας** στην εκτέλεση των εντολών, εφόσον δεν υπάρχουν εξαρτήσεις, μια εντολή μπορεί μόνο να εκτελεστεί αλλά όχι να ολοκληρωθεί πριν μια άλλη

7.2 ΠΜΜΕ επεξεργαστές

7.2.1. Βασικά σημεία θεωρίας

1. Χαρακτηριστικά επεξεργαστή ΠΜΜΕ: Οι επεξεργαστές πολύ μεγάλου μήκους εντολών, ΠΜΜΕ, (Very Long Instruction Word, VLIW, processors), όπως και οι υπερβαθμωτοί επεξεργαστές, εκμεταλλεύονται την **παραλληλία** που μπορεί να υπάρχει σε ένα πρόγραμμα σε επίπεδο εντολών για να πετύχουν ταχύτερη εκτέλεση των προγραμμάτων.

2. Σύγκριση Υπερβαθμωτών και ΠΜΜΕ Επεξεργαστών

- i) Οι εντολές του επεξεργαστή **ΠΜΜΕ** **έχουν σταθερή μορφή** και οι λειτουργίες που περιγράφονται σε μία εντολή δεν ξεπερνούν τις λειτουργικές δυνατότητες του επεξεργαστή. Επομένως η αποκωδικοποίηση των εντολών ενός επεξεργαστή ΠΜΜΕ είναι εύκολη. Σ' ένα υπερβαθμωτό επεξεργαστή θα πρέπει να ελεγχθεί μία ακολουθία από εντολές και η απόφαση για το ποια θα σταλεί για εκτέλεση θα βασιστεί σε παράγοντες όπως πόσες εντολές από κάθε τύπο εντολών ήδη εκτελούνται και τι εξαρτήσεις υπάρχουν.
- ii) Σε ένα επεξεργαστή ΠΜΜΕ η επιλογή των λειτουργιών που θα σταλούν για εκτέλεση σε ένα κύκλο γίνεται κατά τη **μεταγλωττιση του πηγαίου προγράμματος**. Αυτό καλείται **στατικός χρονοπρογραμματισμός (static scheduling)**. Αντίθετα σε ένα υπερβαθμωτό επεξεργαστή, ο χρονοπρογραμματισμός γίνεται από **κυκλώματα κατά την εκτέλεση του εκτελέσιμου προγράμματος** και για αυτό καλείται **δυναμικός χρονοπρογραμματισμός (dynamic scheduling)**. Αυτός είναι ο λόγος που το υλικό υλοποίησης ενός επεξεργαστή ΠΜΜΕ με αντίστοιχο αριθμό λειτουργικών μονάδων είναι **λιγότερο πολύπλοκο**. Το πλεονέκτημα ενός επεξεργαστή ΠΜΜΕ είναι ότι η **μικρότερη πολυπλοκότητα του υλικού που απαιτείται για την υλοποίησή του** οδηγεί συνήθως σε **μεγαλύτερες συχνότητες του σήματος χρονισμού** από αυτές που μπορούν να χρησιμοποιηθούν σε αντίστοιχους υπερβαθμωτούς επεξεργαστές.

7.2.2. Παράδειγμα χρήσης ΠΜΜΕ

Θεωρείστε ένα επεξεργαστή ΠΜΜΕ με ένα αθροιστή καθυστέρησης, μίας χρονικής περιόδου, ένα πολλαπλασιαστή καθυστέρησης τριών χρονικών περιόδων, μια μονάδα προσκόμισης δεδομένων και μία μονάδα εγγραφής αποτελεσμάτων στη μνήμη με καθυστέρηση δύο χρονικών περιόδων κάθε μία. Θεωρείστε, επίσης το ακόλουθο πρόγραμμα σε ψευδοκώδικα.

```
R5 ← R2 + R6
R7 ← R5 + R8
R9 ← R3 + R4
LOAD R4 ← M[X]
R3 ← R2 * R1
R10 ← R3 + R7
STORE M[Y] ← R10
```

Τότε ο μεταγλωττιστής θα παράγει το πρόγραμμα που δίνεται στον παρακάτω Πίνακα.

εντολές	KΠ	KΔ1	KΔ2	KΠ	KΔ1	KΔ2	KΔ	ΘΜ	ΘΜ	KΔ
I ₁	R5	R2	R6	R3	R2	R1				
I ₂	R7	R5	R8			R4	X			
I ₃	R9	R3	R4							
I ₄	R10	R3	R7							
I ₅								Y	R10	

Επεξήγηση

Όταν εκτελεστεί η πρώτη άθροιση (I₁) τότε ο πολλαπλασιασμός αρχίζει να εκτελείται. Στη συνέχεια γίνεται η δεύτερη άθροιση (I₂) και ο πολλαπλασιασμός διανύει τη δεύτερη χρονική περίοδο και όταν τελειώσει και η τρίτη άθροιση (η οποία γίνεται με το παλιό περιεχόμενο του R3 που είναι και το ζητούμενο) ολοκληρώνεται ο πολλαπλασιασμός, οπότε ο R3 αλλάζει περιεχόμενο. Στη συνέχεια όταν εκτελεστεί η τέταρτη άθροιση ο R3 έχει πάρει το νέο περιεχόμενο (που είναι και το ζητούμενο) και γίνεται σωστά η άθροιση.

Χρήση Τδιου Μεταγλωττιστή σε Υπερβαθμωτό

Ένα σύνολο επεξεργαστών μπορεί να έχει το ίδιο σύνολο εντολών σε επίπεδο γλώσσας μηχανής άσχετα αν κάποιοι από αυτούς είναι υλοποιημένοι να εκτελούν τις εντολές καθαρά σειριακά ή χρησιμοποιούν την τεχνική των μερικώς επικαλυπτόμενων λειτουργιών ή είναι υπερβαθμωτοί. Όχι μόνο δε συμβαίνει το ίδιο και με τους επεξεργαστές ΠΙΜΜΕ, αλλά οι επεξεργαστές ΠΙΜΜΕ με διαφορετικούς αριθμούς από λειτουργικές μονάδες απαιτούν διαφορετικά σύνολα εντολών. Αυτό οφείλεται στο γεγονός ότι η εντολή ενός επεξεργαστή ΠΙΜΜΕ αποτελείται από τόσα πεδία όσες είναι και οι λειτουργικές του μονάδες. Επομένως **το πλήθος των λειτουργικών μονάδων πρέπει να είναι γνωστό στο μεταγλωττιστή**. Δυστυχώς όμως χρειάζεται ο μεταγλωττιστής να γνωρίζει και άλλες τεχνολογικές παραμέτρους που αφορούν την υλοποίηση του επεξεργαστή ΠΙΜΜΕ, όπως το πλήθος των χρονικών περιόδων που απαιτούνται για την ολοκλήρωση μίας λειτουργίας σε κάθε λειτουργική μονάδα του επεξεργαστή. Τα ανωτέρω εμποδίζουν τη χρησιμοποίηση του ίδιου μεταγλωττιστή για μια οικογένεια από επεξεργαστές ΠΙΜΜΕ με διαφορετικές υλοποιήσεις.

7.2.3. Θέμα με ΠΙΜΜΕ

α. Θεωρείστε ένα υπερβαθμωτό (superscalar) επεξεργαστή και ένα επεξεργαστή ΠΙΜΜΕ (VLIW) που έχουν το ίδιο πλήθος και είδος λειτουργικών μονάδων. Ποιου επεξεργαστή το υλικό υλοποίησης είναι πιο **πολύπλοκο**; Να δικαιολογήσετε την απάντησή σας.

β. Θεωρείστε ένα επεξεργαστή ΠΙΜΜΕ με ένα αθροιστή καθυστέρησης μιας χρονικής περιόδου, δύο πολλαπλασιαστές καθυστέρησης τριών χρονικών περιόδων, μία μονάδα προσκόμισης δεδομένων και μία μονάδα εγγραφής αποτελεσμάτων στην μνήμη με καθυστέρηση δύο χρονικών περιόδων κάθε μία. Θεωρείστε επίσης το ακόλουθο πρόγραμμα σε ψευδοκώδικα.

LOAD R4 ← M[X]

R1 ← R2 * R6

R7 ← R5 + R8

R10 ← R3 * R7

R9 ← R6 + R4

R3 ← R2 + R5

STORE M[Y] ← R3

Να δώσετε την μορφή της εντολής στον ανωτέρω επεξεργαστή ΠΙΜΜΕ.

Επίσης να δώσετε ένα πρόγραμμα με το ελάχιστο πλήθος εντολών που μπορεί να παράγει ένας μεταγλωττιστής για τον ανωτέρω επεξεργαστή.

Λύση

α. Πιο **πολύπλοκο** είναι το υλικό του υπερβαθμωτού επεξεργαστή. Σε ένα επεξεργαστή ΠΙΜΜΕ η επιλογή των λειτουργιών που θα σταλθούν για εκτέλεση σε ένα κύκλο γίνεται κατά τη **μεταγλώττιση του πηγαίου προγράμματος**. Αυτό καλείται **στατικός χρονοπρογραμματισμός** (static scheduling). Αντίθετα σε ένα υπερβαθμωτό επεξεργαστή, ο χρονοπρογραμματισμός γίνεται από κυκλώματα κατά την εκτέλεση των εκτελέσιμων προγράμματος και για αυτό καλείται **δυναμικός χρονοπρογραμματισμός** (dynamic scheduling). Αυτός είναι ο λόγος που το υλικό υλοποίησης ενός επεξεργαστή ΠΙΜΜΕ με αντίστοιχο αριθμό λειτουργικών μονάδων είναι **λιγότερο πολύπλοκο**

β. Κατασκευάζουμε τον ακόλουθο πίνακα:

	Αθροιστής			Πολλαπλασιαστής 1			Πολλαπλασιαστής 2			Μονάδα προσκόμισης		Μονάδα αποθήκευσης	
εντολές	KΠ	KΔ1	KΔ2	KΠ	KΔ1	KΔ2	KΠ	KΔ1	KΔ2	KΔ	ΘΜ	ΘΜ	KΔ
I ₁	R7	R5	R8	R1	R2	R6				R4	X		
I ₂	R3	R2	R5				R10	R3	R7				
I ₃	R9	R6	R4									Y	R3

7.2.3 Δεύτερο Θέμα με ΠΜΜΕ

Θεωρείστε ένα επεξεργαστή ΠΜΜΕ (VLIW) με ένα αθροιστή καθυστέρησης μιας χρονικής περιόδου, δύο πολλαπλασιαστές καθυστέρησης δύο χρονικών περιόδων, μία μονάδα προσκόμισης δεδομένων και μία μονάδα εγγραφής αποτελεσμάτων στην μνήμη με καθυστέρηση δύο χρονικών περιόδων κάθε μία. Θεωρείστε επίσης το ακόλουθο πρόγραμμα σε ψευδοκώδικα.

```

LOAD R4 ← M[X]
LOAD R9 ← M[W]
R1 ← R2 * R6
R1 ← R4 * R1
R7 ← R5 + R8
R3 ← R2 + R5
R10 ← R3 * R7
R4 ← R5 + R4
R9 ← R6 + R4
STORE M[Y] ← R4
STORE M[Z] ← R10

```

Να δώσετε την μορφή της εντολής στον ανωτέρω επεξεργαστή ΠΜΜΕ. Επίσης να δώσετε ένα πρόγραμμα με το ελάχιστο πλήθος εντολών που μπορεί να παράγει ένας μεταγλωττιστής για τον ανωτέρω επεξεργαστή.

Λύση

Λειτουργικές μονάδες	Αθροιστής			Πολλαπλασιαστής 1			Πολλαπλασιαστής 2			Μονάδα προσκόμισης δεδομένων		Μονάδα αποθήκευσης δεδομένων	
εντολές	KΠ	KΔ1	KΔ2	KΠ	KΔ1	KΔ2	KΠ	KΔ1	KΔ2	KΔ	ΘΜ	ΘΜ	KΔ
R7	R5	R8	R1	R2	R6					R4	X		
R3	R2	R5											
R4	R5	R4	R1	R4	R1	R10	R3	R7	R9	W			
R9	R6	R4									Y	R4	
											Z	R10	

Θέμα Θεωρίας Σεπτεμβρίου 2019 με ΠΜΜΕ

Θεωρήστε επεξεργαστή Πολύ Μεγάλου Μήκους Εντολών, ΠΜΜΕ (Very Long Instruction Processor). Ο μεταγλωττιστής (compiler) πρέπει να γνωρίζει το πλήθος κάθε είδους λειτουργικών μονάδων του επεξεργαστή.

- α) Σωστό
- β) Λάθος

Λύση

Σωστό, διότι η εντολή σε ένα επεξεργαστή ΠΜΜΕ αποτελείται από τόσα πεδία, όσα είναι οι λειτουργικές μονάδες του επεξεργαστή (Κεφάλαιο 7.2).

Θέμα Θεωρίας Σεπτεμβρίου 2019 με ΠΜΜΕ

Θεωρήστε επεξεργαστή Πολύ Μεγάλου Μήκους Εντολών, ΠΜΜΕ (Very Long Instruction Processor). Ένα σύνολο επεξεργαστών μπορεί να έχει το ίδιο σύνολο εντολών σε επίπεδο εντολών γλώσσας μηχανής άσχετα αν κάποιοι από αυτούς είναι υλοποιημένοι να εκτελούν τις εντολές καθαρά σειριακά ή είναι Πολύ Μεγάλου Μήκους Εντολών, ΠΜΜΕ.

- α) Σωστό
- β) Λάθος

Λύση

Λάθος, διότι οι ΠΜΜΕ με διαφορετικούς αριθμούς από λειτουργικές μονάδες απαιτούν διαφορετικά σύνολα εντολών.

Αυτό που αναφέρει η εκφώνηση (δηλαδή το ίδιο σύνολο εντολών) συμβαίνει στους **σειριακούς επεξεργαστές**, σε αυτούς που χρησιμοποιούν την τεχνική των **μερικώς επικαλυπτόμενων λειτουργιών (ΜΕΛ)** ή τους **υπερβαθμωτούς**. Στους ΠΜΜΕ το σύνολο των εντολών σε επίπεδο γλώσσας μηχανής διαφοροποιείται ανάλογα με τον αριθμό των λειτουργικών μονάδων του ΠΜΜΕ.

Θέμα αναφορικά με την κύρια διαφορά υπερβαθμωτών/ΠΜΜΕ

Να αναφέρετε την **κύρια διαφορά** μεταξύ των υπερβαθμωτών επεξεργαστών και των **επεξεργαστών πολύ μεγάλου μήκους εντολών** (very long instruction word, WLIW).

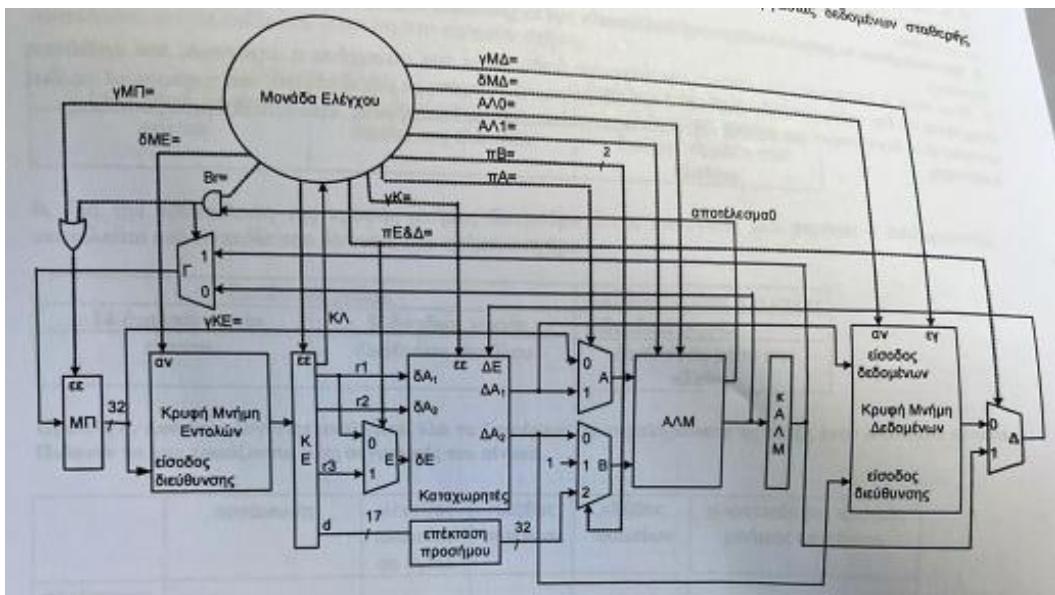
Λύση

Σε έναν επεξεργαστή ΠΙΜΜΕ η επιλογή των λειτουργιών που θα σταλούν για εκτέλεση σε ένα κύκλο γίνεται κατά τη μεταγλώττιση του πηγαίου προγράμματος. Αυτό καλείται στατικός χρονοπρογραμματισμός (static scheduling). Αντίθετα σε ένα υπερβαθμωτό επεξεργαστή ο χρονοπρογραμματισμός γίνεται από κυκλώματα κατά την εκτέλεση του εκτελέσιμου προγράμματος και για αυτό καλείται δυναμικός χρονοπρογραμματισμός (dynamic scheduling). Αυτός είναι ο λόγος που το υλικό υλοποίησης ενός επεξεργαστή ΠΙΜΜΕ με αντίστοιχο αριθμό λειτουργικών μονάδων είναι λιγότερο πολύπλοκο. Το πλεονέκτημα ενός επεξεργαστή ΠΙΜΜΕ είναι ότι η μικρότερη πολυπλοκότητα του υλικού που απαιτείται για την υλοποίησή του οδηγεί συνήθως σε μεγαλύτερες συχνότητες του σήματος χρονισμού από αυτές που μπορούν να χρησιμοποιηθούν σε αντίστοιχους υπερβαθμωτούς επεξεργαστές.

Θέματα Φεβρουαρίου 2023

Θέμα 1

Θεωρήστε επεξεργαστή που διαθέτει τη μονάδα επεξεργασίας δεδομένων σταθερής υποδιαστολής του επόμενου σχήματος.



α. Όλες οι εντολές εκτελούνται σε ένα κύκλο ρολογιού ή ανάλογα της πολυπλοκότητας της εντολής κάποιες εντολές απαιτούν περισσότερους κύκλους από άλλες;

Απαντήστε ναι ή όχι και αιτιολογήστε την απάντησή σας.

β. Μεταξύ των εντολών που μπορούν να εκτελεστούν περιλαμβάνονται και οι κάτωθι εντολές:

STORE r1, (r2) // r1 → M(r2), ADD r1, r2, r3 // r1+r2 → r3, SUB R1, R2, R3 // r1-r2 → r3,

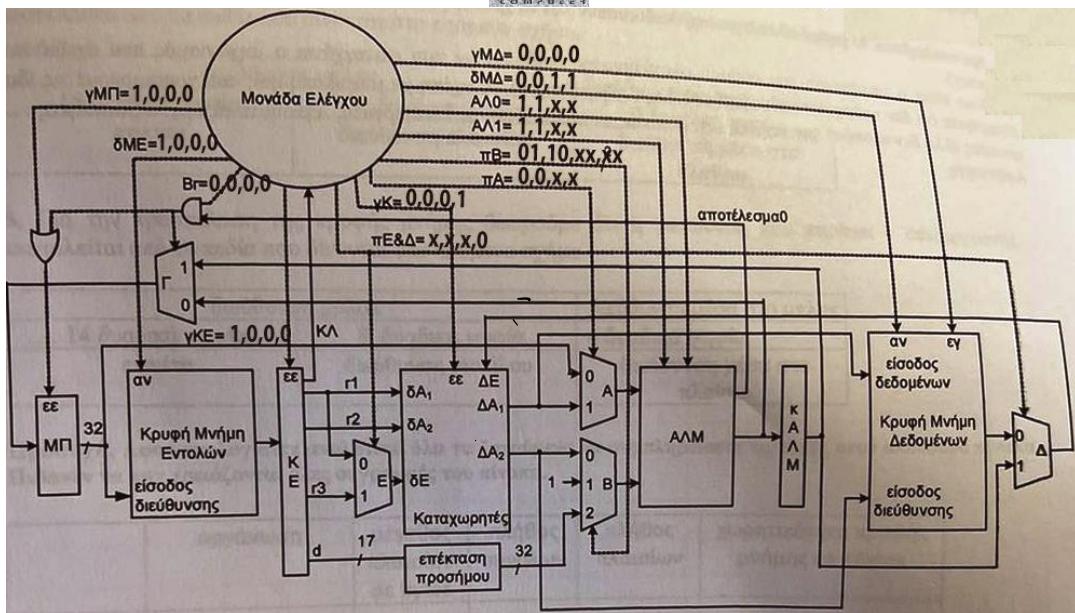
LOAD r1, (r2) // r1 ← M(r2) και BRE r1, r2, d // εάν r1-r2=0 τότε MΠ=MΠ+d

Να βάλετε πάνω στο σχήμα τις τιμές των σημάτων που απαιτούνται για την προσπέλαση εντολής και την εκτέλεση της εντολής **LOAD r1, (r2)**. Για **πρόσθεση** να χρησιμοποιηθούν οι τιμές **ΑΛ1=1, ΑΛ0=1** και για **αφαίρεση** οι τιμές **ΑΛ1=0, ΑΛ0=0**. Δεν χρειάζεται αιτιολόγηση.

Λύση

α. Όχι. (Το "όχι" αφορά το ότι οι εντολές δεν εκτελούνται σε ένα κ.ρ.) Οι εντολές εκτελούνται σε περισσότερους κ.ρ. Αυτό προκύπτει από την ύπαρξη **των καταχωρητών ειδικού σκοπού ΚΕ και κΑΛΜ**, αλλά και από την **απουσία** των αθροιστών A, B που υπάρχουν **μόνο** στην περίπτωση του ενός κ.ρ.

β. Επειδή η συγκεκριμένη εντολή αποθηκεύει το αποτέλεσμά της σε καταχωρητή, θα χρειαστούν 4 κ.ρ. Επομένως για κάθε σήμα ελέγχου θα απαιτηθούν 4 αντίστοιχες τιμές



Θέμα 2

Για την εκτέλεση μιας διαδικασίας απαιτείται η εκτέλεση της ακολουθίας βημάτων $\alpha, \beta, \gamma, \delta, \varepsilon, \zeta, \eta, \theta$ και ι . Τα βήματα αυτά μπορούν να εκτελεστούν από αντίστοιχες μονάδες $A, B, \Gamma, \Delta, E, Z, H, \Theta$ και I με καθυστέρηση 9ns, 7ns, 6ns, 13 ns, 8 ns, 2 ns, 10 ns, 2 ns και 3 ns αντίστοιχα.

Έχετε στη διάθεσή σας καταχωρητές που η καθυστέρησή τους είναι 0,5 nsec.

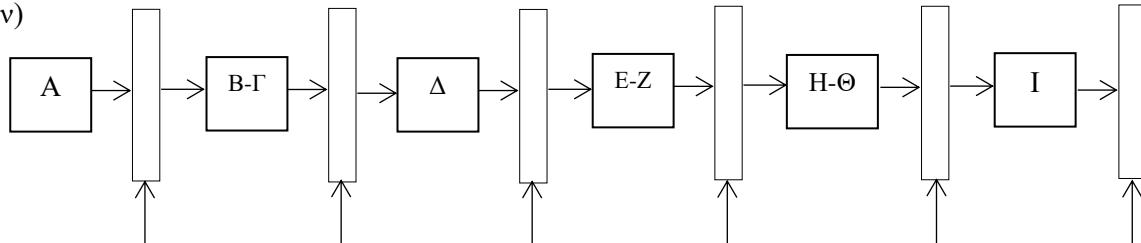
α. Να σχεδιάσετε το μηχανισμό μερικώς επικαλυπτόμενων λειτουργιών (pipelining) που επιτυγχάνει το μέγιστο ρυθμό ολοκλήρωσης διαδικασιών και για αυτό το ρυθμό, τον ελάχιστο χρόνο εκτέλεσης μιας διαδικασίας (latency). Δεν χρειάζεται αιτιολόγηση.

β. Να υπολογίσετε το ρυθμό ολοκλήρωσης διαδικασιών και το χρόνο που απαιτείται για την εκτέλεση μιας διαδικασίας (latency).

γ. Ποια είναι η επιτάχυνση του ρυθμού ολοκλήρωσης διαδικασιών που επιτυγχάνει ο μηχανισμός που σχεδιάστε (θεωρήστε ότι δεν υπάρχουν καθυστερήσεις λόγω εξαρτήσεων) σε σχέση με μια υλοποίηση που χρησιμοποιεί τις ίδιες μονάδες αλλά δεν υλοποιεί την τεχνική των μερικώς επικαλυπτόμενων λειτουργιών; Αγνοήστε τις αρχικές συνθήκες.

Λύση

α. Θα ενοποιήσουμε τις μονάδες σε βαθμίδες, χωρίς να ξεπεράσουμε τη μέγιστη καθυστέρηση (χωρίς την καθυστέρηση των καταχωρητών)



$$\text{ΠΣΧ} = 13 + 0.5 = 13.5 \text{ nsec.}$$

$$\text{X.O.} = \text{πλήθος βαθμίδων} * \text{ΠΣΧ} = 6 * 13.5 \text{ nsec} = 81 \text{ nsec.}$$

$$\beta) \text{P.O.} = 1 \text{ διεργασία κάθε } \text{ΠΣΧ} = 1 \text{ διεργασία κάθε } \text{ΠΣΧ} = 1 \text{ διεργασία κάθε } 13.5 \text{ nsec}$$

$$\gamma) \text{EPO} = \frac{\text{Κ*χρόνος εκτέλεσης χωρίς pipeline}}{(N+K-1)*\text{χρόνος εκτέλεσης με pipeline}} = \frac{\text{Κ*χρόνος εκτέλεσης χωρίς pipeline}}{(6+k-1)*\text{ΠΣΧ}} = \frac{\text{Κ*χρόνος εκτέλεσης χωρίς pipeline}}{(5+K)*13.5 \text{ nsec}}$$

Η μέγιστη θεωρητική τιμή του EPO είναι ίση με N, όπου N = αριθμός βαθμίδων = 6

Θέμα 3

Θεωρήστε επεξεργαστή που παράγει φυσικές διευθύνσεις των 28 δυαδικών ψηφίων και διαθέτει **κρυφή μνήμη** ενός επιπέδου στο χώρο των φυσικών διευθύνσεων με χωρητικότητα μικρότερη των 66 Kbytes. Σε κάθε διεύθυνση αντιστοιχεί μια

ψηφιολέξη (byte). Για κάθε μία των περιπτώσεων Α και Β να δώσετε την οργάνωση, το μέγεθος του πλαισίου σε ψηφιολέξεις, το πλήθος των πλαισίων, το πλήθος των συνόλων (αν η οργάνωση συνεπάγεται την ύπαρξη συνόλων) και τη χωρητικότητα που μπορεί να έχει η κρυφή μνήμη.

Α. Για την προσπέλαση της κρυφής μνήμης θεωρούμε ότι η διεύθυνση που παράγει ο επεξεργαστής αποτελείται από τα πεδία που δίνονται στο επόμενο σχήμα.

διεύθυνση μπλοκ		διεύθυνση μέσα στο μπλοκ
16 δυαδικά ψηφία	9 δυαδικά ψηφία	3 δυαδικά ψηφία
ετικέτα	διεύθυνση πλαισίου	διεύθυνση μέσα στο πλαίσιο

Β. Για την προσπέλαση της κρυφής μνήμης θεωρούμε ότι η διεύθυνση που παράγει ο επεξεργαστής αποτελείται από τα πεδία που δίνονται στο επόμενο σχήμα.

διεύθυνση μπλοκ		διεύθυνση μέσα στο μπλοκ
14 δυαδικά ψηφία	8 δυαδικά ψηφία	6 δυαδικά ψηφία
ετικέτα	διεύθυνση συνόλου	διεύθυνση μέσα στο πλαίσιο

Προσοχή. Αφού υπολογίζετε αναλυτικά όλα τα ζητούμενα να συμπληρώσετε τις τιμές στον ακόλουθο πίνακα. Πιθανόν να μην χρειάζονται όλες οι γραμμές του πίνακα.

	Οργάνωση	Μέγεθος πλαισίου σε bytes	Πλήθος συνόλων	Πλήθος πλαισίων	Χωρητικότητα κρυφής μνήμης σε KBytes
περίπτωση Α					
περίπτωση Α					
περίπτωση Α					
περίπτωση Β					
περίπτωση Β					
περίπτωση Β					
περίπτωση Β					

Λύση

	Οργάνωση	Μέγεθος πλαισίου σε bytes	πλήθος συνόλων	πλήθος πλαισίων	Χωρητικότητα κρυφής μνήμης σε KBytes
περίπτωση Α	μονοσήμαντη απεικόνιση	$2^3 = 8$	-	$2^9 = 512$	$2^9 \pi\lambda \cdot 2^3 \lambda \varepsilon \xi / \pi\lambda = 2^{12} \text{ bytes} = 2^2 2^{10} = 4 \text{ KB}$
περίπτωση Α					
περίπτωση Α					
περίπτωση Β	2-τρόπων συνόλου συσχέτισης	$2^6 = 64$	$2^8 = 256$	$2^8 \cdot 2 = 2^9$	$2^8 \sigma\nu\nu \cdot 2\pi\lambda/\sigma\nu\nu \cdot 2^6 \lambda \varepsilon \xi / \pi\lambda = 2^{15} \text{ bytes} = 32 \text{ KB}$
περίπτωση Β	3-τρόπων συνόλου συσχέτισης	$2^6 = 64$	$2^8 = 256$	$2^8 \cdot 3 = 768$	$2^8 \sigma\nu\nu \cdot 3 \pi\lambda/\sigma\nu\nu \cdot 2^6 \lambda \varepsilon \xi / \pi\lambda = 3 \cdot 2^{14} \text{ bytes} = 3 \cdot 16 \cdot 2^{10} \text{ bytes} = 48 \text{ KB}$
περίπτωση Β	4-τρόπων συνόλου συσχέτισης	$2^6 = 64$	$2^8 = 256$	$2^8 \cdot 2^2 = 2^{10}$	$2^8 \sigma\nu\nu \cdot 2^2 \pi\lambda/\sigma\nu\nu \cdot 2^6 \lambda \varepsilon \xi / \pi\lambda = 2^{16} \text{ bytes} = 64 \text{ KB}$
περίπτωση Β	5-τρόπων συνόλου συσχέτισης	$2^6 = 64$	$2^8 = 256$	$2^8 \cdot 5 = 1280$	$2^8 \sigma\nu\nu \cdot 5 \pi\lambda/\sigma\nu\nu \cdot 2^6 \lambda \varepsilon \xi / \pi\lambda = 5 \cdot 2^{14} \text{ bytes} = 5 \cdot 16 \cdot 2^{10} \text{ bytes} = 80 \text{ KB} > 66 \text{ KB}$

Θέματα Προόδου 2023

Θέμα 1

Θεωρήστε κρυφή μνήμη με χωρητικότητα 8 Kbytes προσπελάσιμη με φυσικές διευθύνσεις των 32 δυαδικών ψηφίων και πλαίσιο των 16 Bytes. Σε κάθε byte του συστήματος μνήμης αντιστοιχεί μια φυσική διεύθυνση. Δίνονται οι διευθύνσεις:

- α. A379632A
- β. A3796323
- γ. A379B326
- δ. A3799326

α. Οργάνωση μονοσήμαντης απεικόνισης

Να δώσετε το μέγεθος του πλαισίου «Διεύθυνση μέσα στο πλαίσιο δμΠ» σε δυαδικά ψηφία (καθαρός αριθμός)

Να δώσετε το μέγεθος του πλαισίου «Διεύθυνση πλαισίου» σε δυαδικά ψηφία (καθαρός αριθμός)



Εάν τα περιεχόμενα των διευθύνσεων α και β είναι δυνατόν να βρίσκονται ταυτόχρονα στην κρυφή μνήμη γράψτε YES, διαφορετικά γράψτε NO.

Εάν τα περιεχόμενα των διευθύνσεων α και β είναι δυνατόν να βρίσκονται ταυτόχρονα στην κρυφή μνήμη γράψτε YES, διαφορετικά γράψτε NO.

Εάν τα περιεχόμενα των διευθύνσεων α, γ και δ είναι δυνατόν να βρίσκονται ταυτόχρονα στην κρυφή μνήμη γράψτε YES, διαφορετικά γράψτε NO.

β. Οργάνωση 2-τρόπων συνόλου συσχέτισης

Να δώσετε το μέγεθος του πλαισίου «Διεύθυνση μέσα στο πλαίσιο δμΠ» σε δυαδικά ψηφία (καθαρός αριθμός)

Να δώσετε το μέγεθος του πλαισίου «Διεύθυνση συνόλου» σε δυαδικά ψηφία (καθαρός αριθμός)

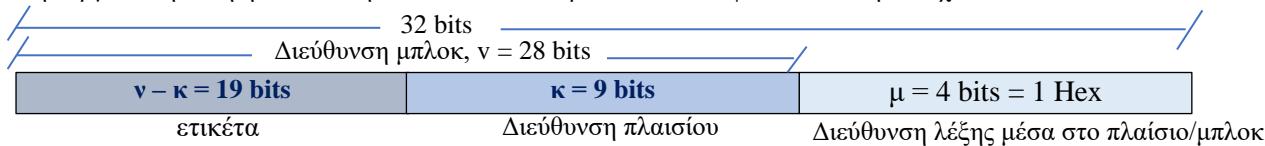
Εάν τα περιεχόμενα των διευθύνσεων α, β και δ είναι δυνατόν να βρίσκονται ταυτόχρονα στην κρυφή μνήμη γράψτε YES, διαφορετικά γράψτε NO.

Εάν τα περιεχόμενα των διευθύνσεων α, β, γ και δ είναι δυνατόν να βρίσκονται ταυτόχρονα στην κρυφή μνήμη γράψτε YES, διαφορετικά γράψτε NO.

Λύση

α. Οργάνωση μονοσήμαντης απεικόνισης

Στην οργάνωση αυτή η διεύθυνση αποτελείται από τρία πεδία, που φαίνονται στη συνέχεια



Πλαίσιο 16 bytes (ψηφιολέξεων) = $16 \cdot \theta \cdot \mu = 2^4 \cdot \theta \cdot \mu = 2^\mu \Rightarrow \mu = 4$ bits, πάντα μετατρέπουμε τις χωρητικότητες σε θέσεις μνήμης και στην προκειμένη περίπτωση λόγω της οργάνωσης που έχουμε **σε κάθε byte του συστήματος μνήμης αντιστοιχεί μια φυσική διεύθυνση, ισχύει ότι τα bytes είναι αυτόματα και θέσεις μνήμης.**

Από τη χωρητικότητα της κρυφής μνήμης, όταν δίνεται, υπολογίζουμε τον αριθμό των πλαισίων.

1 πλ. 16 bytes

$$x; 8 \text{ KB} \quad x = 2^{13}/2^4 = 2^9 \text{ πλ.} = 2^\kappa \Rightarrow \kappa = 9$$

Άρα η διεύθυνση μέσα στο πλαίσιο = $\mu = 4$ και η διεύθυνση πλαισίου = $\kappa = 9$.

Το συνολικό μήκος της διεύθυνσης που παράγει ο επεξεργαστής είναι 32 bits, κάτι που σημαίνει ότι το πεδίο της ετικέτας $v - \kappa$ θα το υπολογίζουμε με αφαίρεση και θα είναι ίσο με $32 - 13 = 19$ bits. Άρα το πεδίο «Διεύθυνση μέσα στο πλαίσιο δμΠ» = $\mu = 4$ bits και το πεδίο «Διεύθυνση πλαισίου» = $\kappa = 9$ bits.

Σημείωση: Επειδή οι διευθύνσεις που δίνονται είναι στο δεκαεξαδικό σύστημα και από την άλλη τα μεγέθη των πεδίων που υπολογίζαμε δεν είναι πολλαπλάσια του «4», δεν μπορούμε να τα μετατρέψουμε σε δεκαεξαδικά. Αν όμως ήταν πολλαπλάσια του «4», τότε θα συνέφερε να τα μετατρέψουμε σε δεκαεξαδικά.

$v - \kappa$	κ
a. A379632A = 1010 0011 0111 1001 011	0 0011 0010
β. A3796323 = 1010 0011 0111 1001 011	0 0011 0010
γ. A379B326 = 1010 0011 0111 1001 101	1 0011 0010
δ. A3799326 = 1010 0011 0111 1001 100	1 0011 0010

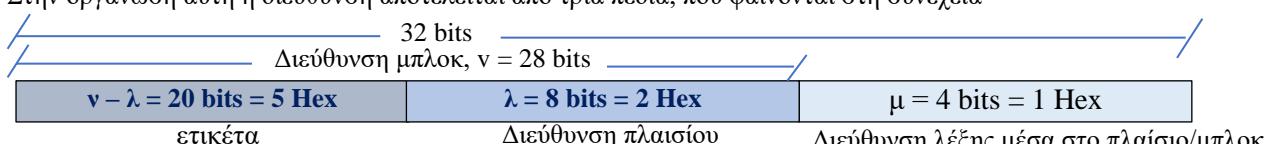
Οι διευθύνσεις α, β έχουν **ίδια διεύθυνση πλαισίου «κ»** και **ίδια ετικέτα «v-κ»**, άρα αναφέρονται στο **ίδιο μπλοκ** με διεύθυνση «v» που μεταφέρεται **στο ίδιο πλαίσιο** με διεύθυνση «κ», αυτό σημαίνει ότι **μπορούν και πρέπει** να βρίσκονται ταυτόχρονα στην κρυφή μνήμη, επομένως η απάντηση είναι Yes.

Οι διευθύνσεις γ, δ έχουν **ίδια διεύθυνση πλαισίου «κ»** και **διαφορετική ετικέτα «v-κ»**, άρα αναφέρονται στο **ίδιο μπλοκ** με διεύθυνση «v» που μεταφέρεται **σε διαφορετικό πλαίσιο** με διεύθυνση «κ», αυτό σημαίνει ότι δεν μπορούν να βρίσκονται ταυτόχρονα στην κρυφή μνήμη, επομένως η απάντηση είναι No.

Οι διευθύνσεις α, γ, δ **δεν** μπορούν να βρίσκονται ταυτόχρονα στην κρυφή μνήμη, επομένως η απάντηση είναι No.

β. Οργάνωση 2-τρόπων απεικόνισης

Στην οργάνωση αυτή η διεύθυνση αποτελείται από τρία πεδία, που φαίνονται στη συνέχεια



Από τον **αριθμό των πλαισίων υπολογίζουμε πάντα τον αριθμό των συνόλων**, ως εξής:

1 σύνολο 2 πλαίσια

$$x; 2^9 \text{ πλαίσια} \quad x = 2^9/2 = 2^8 \text{ σύνολα} = 2^\lambda \Rightarrow \lambda = 8$$

Άρα η διεύθυνση μέσα στο πλαίσιο = $\mu = 4$ και η διεύθυνση συνόλου = $\lambda = 8$.

Το συνολικό μήκος της διεύθυνσης που παράγει ο επεξεργαστής είναι 32 bits, κάτι που σημαίνει ότι το πεδίο της ετικέτας $v - \lambda$ θα το υπολογίσουμε με αφαίρεση και θα είναι ίσο με $32 - 12 = 20$ bits. Άρα το πεδίο «Διεύθυνση μέσα στο πλαίσιο δμΠ» = $\mu = 4$ bits και το πεδίο «Διεύθυνση συνόλου» = $\lambda = 9$ bits.

Σημείωση: επειδή στην περίπτωση αυτή **τα μεγέθη και των τριών πεδίων είναι πολλαπλάσια του «4»**, συμφέρει, επειδή οι διευθύνσεις είναι δοσμένες στο δεκαεξαδικό, να τα ομαδοποιήσουμε ανά τέσσερα, για να μη χρειαστεί να τα μετατρέψουμε στο δυαδικό σε περίπτωση που τα ελέγχουμε για να δούμε αν μπορούν κάποιες από αυτές τις διευθύνσεις να είναι ταυτόχρονα στην κρυφή μνήμη. Αι οι διευθύνσεις είχαν δοθεί στο οκταδικό σύστημα, τότε αν το επέτρεπαν τα μεγέθη των επιμέρους πεδίων (πολλαπλάσια του «8»), θα τις ομαδοποιούσαμε ανά τρία ψηφία.

Στη συνέχεια χωρίζουμε τις διευθύνσεις σε επιμέρους πεδία, όπως φαίνεται στη συνέχεια:

a. A379632A = A3796 $\begin{array}{|c|c|} \hline 32 & A \\ \hline v-\lambda & \lambda \\ \hline \end{array}$

β. A3796323 = A3796 $\begin{array}{|c|c|} \hline 32 & 3 \\ \hline v-\lambda & \lambda \\ \hline \end{array}$

γ. A379B326 = A379B $\begin{array}{|c|c|} \hline 32 & 6 \\ \hline v-\lambda & \lambda \\ \hline \end{array}$

δ. A3799326 = A3799 $\begin{array}{|c|c|} \hline 32 & 6 \\ \hline v-\lambda & \lambda \\ \hline \end{array}$



Οι διευθύνσεις α, β, δ έχουν **ίδια διεύθυνση συνόλου** «λ» και οι δύο πρώτες έχουν και **ίδια ετικέτα «v-λ»**, άρα αναφέρονται στο **ίδιο μπλοκ** με διεύθυνση «ν» που μεταφέρεται στο **ίδιο σύνολο** με διεύθυνση «λ» και μάλιστα μεταφέρονται και στο **ίδιο πλαίσιο** του συνόλου με διεύθυνση 32, π.χ. στο πλαίσιο Π0. Η διεύθυνση δ μεταφέρεται και αυτή στο ίδιο σύνολο, το 32 και μεταφέρεται στο πλαίσιο Π1. Αυτό σημαίνει ότι **μπορούν και οι τρεις** να βρίσκονται ταυτόχρονα στην κρυφή μνήμη, επομένως η απάντηση είναι Yes.

Οι διευθύνσεις α, β, γ και δ έχουν **ίδια διεύθυνση συνόλου** «λ» και οι δύο τελευταίες **διαφορετική ετικέτα «v-λ»**, άρα αναφέρονται σε **διαφορετικό μπλοκ** με διεύθυνση «ν». Επομένως, υπάρχουν τέσσερις διευθύνσεις που μεταφέρονται στο ίδιο σύνολο (αυτό με διεύθυνση 32) και από αυτές, οι α, β, στο ίδιο πλαίσιο του Σ32, ενώ οι γ, δ σε διαφορετικά πλαίσια του Σ32. Αυτό σημαίνει ότι δεν μπορούν να βρίσκονται ταυτόχρονα στην κρυφή μνήμη, επομένως η απάντηση είναι No.

Θέμα 2

Θεωρήστε κρυφή μνήμη με χωρητικότητα 128 Kbytes προσπελάσιμη με φυσικές διευθύνσεις των 32 δυαδικών ψηφίων και πλαίσιο των 16 Bytes. **Σε κάθε byte του συστήματος μνήμης αντιστοιχεί μια φυσική διεύθυνση.** Δίνονται οι διευθύνσεις:

a. FF629769

β. FF629763

γ. FF6E9768

δ. FF697A68

α. Οργάνωση μονοσήμαντης απεικόνισης

Να δώσετε το μέγεθος του πλαισίου «Διεύθυνση μέσα στο πλαίσιο δμΠ» σε δυαδικά ψηφία (καθαρός αριθμός)

Να δώσετε το μέγεθος του πλαισίου «Διεύθυνση πλαισίου» σε δυαδικά ψηφία (καθαρός αριθμός)

Εάν τα περιεχόμενα των διευθύνσεων α και β είναι δυνατόν να βρίσκονται ταυτόχρονα στην κρυφή μνήμη γράψτε YES, διαφορετικά γράψτε NO.

Εάν τα περιεχόμενα των διευθύνσεων α και β είναι δυνατόν να βρίσκονται ταυτόχρονα στην κρυφή μνήμη γράψτε YES, διαφορετικά γράψτε NO.

Εάν τα περιεχόμενα των διευθύνσεων α, γ και δ είναι δυνατόν να βρίσκονται ταυτόχρονα στην κρυφή μνήμη γράψτε YES, διαφορετικά γράψτε NO.

β. Οργάνωση 2-τρόπων συνόλου συσχέτισης

Να δώσετε το μέγεθος του πλαισίου «Διεύθυνση μέσα στο πλαίσιο δμΠ» σε δυαδικά ψηφία (καθαρός αριθμός)

Να δώσετε το μέγεθος του πλαισίου «Διεύθυνση συνόλου» σε δυαδικά ψηφία (καθαρός αριθμός)

Εάν τα περιεχόμενα των διευθύνσεων α, β και δ είναι δυνατόν να βρίσκονται ταυτόχρονα στην κρυφή μνήμη γράψτε YES, διαφορετικά γράψτε NO.

Εάν τα περιεχόμενα των διευθύνσεων α, β, γ και δ είναι δυνατόν να βρίσκονται ταυτόχρονα στην κρυφή μνήμη γράψτε YES, διαφορετικά γράψτε NO.

Αύστη

α. Οργάνωση μονοσήμαντης απεικόνισης

Στην οργάνωση αυτή η διεύθυνση αποτελείται από τρία πεδία, που φαίνονται στη συνέχεια



$v - \kappa = 15$ bits

$\kappa = 13$ bits

$\mu = 4$ bits = 1 Hex

ετικέτα

Διεύθυνση πλαισίου

Διεύθυνση λέξης μέσα στο πλαίσιο/μπλοκ

Πλαίσιο 16 bytes ($\psi_{\text{φιολέζεων}} = 16 \theta_{\mu} = 2^4 \theta_{\mu} = 2^{\mu}$ ⇒ $\mu = 4$ bits, πάντα μετατρέπουμε τις χωρητικότητες σε θέσεις μνήμης και στην προκειμένη περίπτωση λόγω της οργάνωσης που έχουμε **σε κάθε byte του συστήματος μνήμης αντιστοιχεί μια φυσική διεύθυνση, ισχύει ότι τα bytes είναι αυτόματα και θέσεις μνήμης.**

Από τη χωρητικότητα της κρυφής μνήμης, όταν δίνεται, υπολογίζουμε τον αριθμό των πλαισίων.

1 πλ. 16 bytes

x; 128 KB

$$x = 2^{17}/2^4 = 2^{13} \text{ πλ.} = 2^{\kappa} \Rightarrow \kappa = 13$$

Άρα η διεύθυνση μέσα στο πλαίσιο = $\mu = 4$ και η διεύθυνση πλαισίου = $\kappa = 9$.

Το συνολικό μήκος της διεύθυνσης που παράγει ο επεξεργαστής είναι 32 bits, κάτι που σημαίνει ότι το πεδίο της ετικέτας $v - \lambda$ θα το υπολογίζουμε με αφαίρεση και θα είναι ίσο με $32 - 17 = 15$ bits. Άρα το πεδίο «Διεύθυνση μέσα στο πλαίσιο δμΠ» = $\mu = 4$ bits και το πεδίο «Διεύθυνση πλαισίου» = $\kappa = 15$ bits.

Σημείωση: Επειδή οι διευθύνσεις που δίνονται είναι στο δεκαεξαδικό σύστημα και από την άλλη τα μεγέθη των πεδίων που υπολογίσαμε δεν είναι πολλαπλάσια του «4», δεν μπορούμε να τα μετατρέψουμε σε δεκαεξαδικά. Αν όμως ήταν πολλαπλάσια του «4», τότε θα συνέφερε να τα μετατρέψουμε σε δεκαεξαδικά.

	$v - \lambda$	κ
a. FF629769	1111 1111 0110 001	0 1001 0111 0110
β. FF629763	1111 1111 0110 001	0 1001 0111 0110
γ. FF6E9768	1111 1111 0110 111	0 1001 0111 0110
δ. FF697A68	1111 1111 0110 100	1 0111 1010 0110

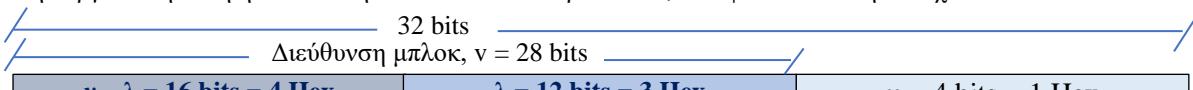
Οι διευθύνσεις α, β έχουν **ίδια διεύθυνση πλαισίου «κ»** και **ίδια ετικέτα «v-κ»**, άρα αναφέρονται στο **ίδιο μπλοκ** με διεύθυνση «v» που μεταφέρεται **στο ίδιο πλαίσιο** με διεύθυνση «κ», αυτό σημαίνει ότι **μπορούν και πρέπει** να βρίσκονται ταυτόχρονα στην κρυφή μνήμη, επομένως η απάντηση είναι Yes.

Οι διευθύνσεις γ, δ έχουν **διαφορετική διεύθυνση πλαισίου «κ» μπορούν να βρίσκονται ταυτόχρονα** στην κρυφή μνήμη, επομένως η απάντηση είναι Yes.

Οι διευθύνσεις α, γ, δ **δεν μπορούν να βρίσκονται ταυτόχρονα** στην κρυφή μνήμη, επομένως η απάντηση είναι No.

β. Οργάνωση 2-τρόπων απεικόνισης

Στην οργάνωση αυτή η διεύθυνση αποτελείται από τρία πεδία, που φαίνονται στη συνέχεια



Από τον **αριθμό των πλαισίων υπολογίζουμε πάντα τον αριθμό των συνόλων**, ως εξής:

1 σύνολο 2 πλαίσια

x; 2¹³ πλαίσια

$$x = 2^{13}/2 = 2^{12} \text{ σύνολα} = 2^{\lambda} \Rightarrow \lambda = 12$$

Άρα η διεύθυνση μέσα στο πλαίσιο = $\mu = 4$ και η διεύθυνση συνόλου = $\lambda = 8$.

Το συνολικό μήκος της διεύθυνσης που παράγει ο επεξεργαστής είναι 32 bits, κάτι που σημαίνει ότι το πεδίο της ετικέτας $v - \lambda$ θα το υπολογίζουμε με αφαίρεση και θα είναι ίσο με $32 - 16 = 16$ bits. Άρα το πεδίο «Διεύθυνση μέσα στο πλαίσιο δμΠ» = $\mu = 4$ bits και το πεδίο «Διεύθυνση συνόλου» = $\lambda = 16$ bits.

Σημείωση: επειδή στην περίπτωση αυτή **τα μεγέθη και των τριών πεδίων είναι πολλαπλάσια του «4»**, συμφέρει, επειδή οι διευθύνσεις είναι δοσμένες στο δεκαεξαδικό, να τα ομαδοποιήσουμε ανά τέσσερα, για να μη χρειαστεί να τα μετατρέψουμε στο δυαδικό σε περίπτωση που τα ελέγχουμε για να δούμε αν μπορούν κάποιες από αυτές τις διευθύνσεις να είναι ταυτόχρονα στην κρυφή μνήμη. Αι οι διευθύνσεις είχαν δοθεί στο οκταδικό σύστημα, τότε αν το επέτρεπαν τα μεγέθη των επιμέρους πεδίων (πολλαπλάσια του «8»), θα τις ομαδοποιούσαμε ανά τρία ψηφία.

Στη συνέχεια χωρίζουμε τις διευθύνσεις σε επιμέρους πεδία, όπως φαίνεται στη συνέχεια:

α. FF629769 = FF62 | 976 | 9

β. FF629763 = FF62 | 976 | 3

γ. FF6E9768 = FF6E | 976 | 8

δ. FF697A68 = FF69 | 7A6 | 8

Οι διευθύνσεις α, β, δ έχουν **ίδια διεύθυνση συνόλου «λ»** και οι δύο πρώτες έχουν και **ίδια ετικέτα «v-λ»**, άρα αναφέρονται στο **ίδιο μπλοκ** με διεύθυνση «v» που μεταφέρεται **στο ίδιο σύνολο** με διεύθυνση «λ» και μάλιστα μεταφέρονται και στο **ίδιο πλαίσιο** του συνόλου με διεύθυνση 32, π.χ. στο πλαίσιο Π0. Η διεύθυνση δ μεταφέρεται σε άλλο σύνολο. Αυτό σημαίνει ότι **μπορούν και οι τρεις** να βρίσκονται ταυτόχρονα στην κρυφή μνήμη, επομένως η απάντηση είναι Yes.

Οι διευθύνσεις α, β, γ και δ μπορούν να βρίσκονται ταυτόχρονα στην κρυφή μνήμη, διότι οι α, β μεταφέρονται στο Π0 του Σ976, η γ μεταφέρεται στο Π1 του Σ976 και η δ στο Π0 του συνόλου Σ7A6. Επομένως η απάντηση είναι Yes.

Θέμα 3

Θεωρήστε υπολογιστή με κύρια μνήμη χωρητικότητας 16 MBytes και οργάνωση μίας ψηφιολέξης (byte) ανά θέση μνήμης, ιδεατή μνήμη που χρησιμοποιεί λογικές διευθύνσεις των 32 δυαδικών ψηφίων και μέγεθος σελίδας 256 Bytes.

Υποθέστε ότι η ιδεατή μνήμη υλοποιείται με την τεχνική του αντεστραμμένου πίνακα σελίδων με τρεις αντιστοιχίσεις ανά θέση του πίνακα και η σχετική διεύθυνση μέσα στον αντεστραμμένο πίνακα σελίδων, σδμΑΠΣ, δίνεται από τη συνάρτηση διασποράς

σδμΑΠΣ = υπόλοιπο της διαίρεσης του ΑΛΣ δια του 2^{16} .

Θεωρήστε ότι τα περιεχόμενα του αντεστραμμένου πίνακα σελίδων είναι αυτά που φαίνονται στον Πίνακα (στο δεκαεξαδικό) και το περιεχόμενο του βασικού καταχωρητή πίνακα σελίδων είναι μηδέν.

Θεωρήστε ότι ο επεξεργαστής παράγει τη λογική διεύθυνση 45B72A64.

Πίνακας

Τρέχουσα κατάσταση του αντεστραμμένου πίνακα σελίδων (από τα πεδια ελέγχου δίνουμε την τιμή μόνο των πεδίων Παρουσίας Π της αντιστοίχισης). Δεν δίνεται το περιεχόμενο όλων των θέσεων του Πίνακα.

Διεύθυνση	ΠΙ	ΑΛΣ1	φδΣ1	ΠΙ	ΑΛΣ2	φδΣ2	ΠΙ	ΑΛΣ3	φδΣ3
0494	I	130494	E3A9	I	BF0494	8101	O	BF0494	342A
16EA	O	1716EA	7E7A	I	6116EA	290D	I	1716EA	7EF6
2563	I	EF2563	0EE3	I	FE2563	4309	O	232563	3E45
2CFE	I	AB2CFE	5AEF	I	542CFE	EA46	I	592CFE	5AEF
68DF	I	1568DF	CB00	I	8968DF	CB00	O	5768DF	0945
7077	I	007077	23EA	I	057077	B697	O	787077	B698
78CC	I	3278CC	465B	O	2378CC	AD45	I	AC78CC	8923
7CDF	O	237CDF	3543	I	247CDF	A342	I	0E7CDF	9C44
8C55	O	A48C55	57A3	I	D48C55	2443	I	A38C55	45B2
8E8E	I	D08E8E	D6A1	I	988E8E	1987	I	AD8E8E	AD45
9F45	I	279F45	3498	I	859F45	CBEE	O	399F45	27AC
B72A	I	99B72A	DF64	I	45B72A	035B	I	ADB72A	AD45
BF04	I	94BF04	AE45	O	23BF04	4512	I	49BF04	F567
C508	O	00C508	2300	I	11C508	4311	I	10C508	9DFE
D08E	I	A3D08E	B546	I	8ED08E	2567	I	56D08E	8909
D364	I	66D364	69FB	O	96D364	AD45	I	A4D364	C678
EA0D	I	JAEA0D	9EEE	I	IBEA0D	6578	I	A4EA0D	6543

Να δώσετε τη φυσική διεύθυνση που αντιστοιχεί στην ανωτέρω λογική διεύθυνση.

Θέμα 4

Θεωρήστε υπολογιστή με κύρια μνήμη χωρητικότητας 256 Mbytes και οργάνωση μίας ψηφιολέξης (byte) ανά θέση μνήμης, ιδεατή μνήμη που χρησιμοποιεί λογικές διευθύνσεις των 36 δυαδικών ψηφίων και μέγεθος σελίδας 4 KBytes.

Υποθέστε ότι η ιδεατή μνήμη υλοποιείται με την τεχνική του αντεστραμμένου πίνακα σελίδων με τρεις αντιστοιχίσεις ανά θέση του πίνακα και η σχετική διεύθυνση μέσα στον αντεστραμμένο πίνακα σελίδων, σδμΑΠΣ, δίνεται από τη συνάρτηση διασποράς

σδμΑΠΣ = υπόλοιπο της διαίρεσης του ΑΛΣ δια του 2^{16} .

Θεωρήστε ότι τα περιεχόμενα του αντεστραμμένου πίνακα σελίδων είναι αυτά που φαίνονται στον Πίνακα (στο δεκαεξαδικό) και το περιεχόμενο του βασικού καταχωρητή πίνακα σελίδων είναι μηδέν.

Θεωρήστε ότι ο επεξεργαστής παράγει τη λογική διεύθυνση BF04940E4.

Πίνακας

Τρέζουσα κατάσταση του αντεστραμμένου πίνακα σελίδων (από τα πεδία έλξης δύνομας την τιμή μόνο του πεδίου Παρουσίας Π της αντιστοίχησης). Δινείται το περιεχόμενο όλων των θέσιων του Πίνακα.



Διεύθυνση	Π	ΑΛΣ1	φδΣ1	Π	ΑΛΣ2	φδΣ2	Π	ΑΛΣ3	φδΣ3
0494	1	130494	E3A9	1	BF0494	8101	0	BF0494	342A
16EA	0	1716EA	7E7A	1	6116EA	290D	1	1716EA	7EF6
2563	1	EF2563	0EE3	1	FE2563	4309	0	232563	3E45
2CFE	1	AB2CFE	5AEF	1	542CFE	EA46	1	592CFE	5AEF
68DF	1	1568DF	CB00	1	8968DF	CB00	0	5768DF	0945
7077	1	007077	23EA	1	057077	B697	0	787077	B698
78CC	1	3278CC	465B	0	2378CC	AD45	1	AC78CC	8923
7CDF	0	237CDF	3543	1	247CDF	A342	1	0E7CDF	9C44
8C55	0	A48C55	57A3	1	D48C55	2443	1	A38C55	45B2
8E8E	1	D08E8E	D6A1	1	988E8E	1987	1	AD8E8E	AD45
9F45	1	279F45	3498	1	859F45	CBEE	0	399F45	27AC
B72A	1	99B72A	DF64	1	45B72A	035B	1	ADB72A	AD45
BF04	1	94BF04	AE45	0	23BF04	4512	1	49BF04	F567
C508	0	00C508	2300	1	11C508	4311	1	10C508	9DFE
D08E	1	A3D08E	B546	1	8ED08E	2567	1	56D08E	8909
D364	1	66D364	69FB	0	96D364	AD45	1	A4D364	C678
EA0D	1	1AEA0D	9EEE	1	1BEA0D	6578	1	A4EA0D	6543

Να δώσετε τη φυσική διεύθυνση

που αντιστοιχεί στην ανωτέρω λογική διεύθυνση.



Θέμα 5

Θεωρήστε επεξεργαστή στον οποίο το μήκος των λογικών διευθύνσεων είναι 48 δυαδικά ψηφία ενώ το μήκος των φυσικών διευθύνσεων είναι 30 δυαδικά ψηφία και το μέγεθος της σελίδας είναι 8 KBytes. Ο επεξεργαστής διαθέτει υβριδική κρυψή μνήμη επεξεργαστή με οργάνωση 2-τρόπων συνόλου συσχέτισης, με τη μέγιστη δυνατή χωρητικότητα και πλαίσιο των 32 bytes, και υλοποιεί την τακτική της τελικής ενημέρωσης (write back με ένα dirty bit ανά πλαίσιο). Σε κάθε byte του συστήματος μνήμης αντιστοιχεί μία διεύθυνση.<p>

- a. Να υπολογίσετε τη χωρητικότητα της κρυψής μνήμης σε KBytes (να δώσετε ένα καθαρό αριθμό χωρίς τελείες):
- β. Να υπολογίσετε το πλήθος των κυψελίδων που απαιτούνται για την υλοποίηση της κρυψής μνήμης (σε κάθε κυψελίδα αποθηκεύεται ένα δυαδικό ψηφίο). Αγνοήστε τα δυαδικά ψηφία που απαιτούνται για την υλοποίηση του μηχανισμού απελευθέρωσης πλαισίων (replacement policy).

Πλήθος κυψελίδων (δώστε ένα καθαρό αριθμό χωρίς τελείες):

Θέμα

Θεωρήστε επεξεργαστή στον οποίο το μήκος των λογικών διευθύνσεων είναι 42 δυαδικά ψηφία ενώ το μήκος των φυσικών διευθύνσεων είναι 30 δυαδικά ψηφία και το μέγεθος της σελίδας είναι 1 KBytes. Ο επεξεργαστής διαθέτει υβριδική κρυφή μνήμη επεξεργαστή με οργάνωση 3-τρόπων συνόλου συσχέτισης, με τη μέγιστη δυνατή χωρητικότητα και πλαίσιο των 16 bytes, και υλοποιεί την τακτική της τελικής ενημέρωσης (write back με ένα dirty bit ανά πλαίσιο). Σε κάθε byte του συστήματος μνήμης αντιστοιχεί μία διεύθυνση.<p>

a. Να υπολογίσετε τη χωρητικότητα της κρυφής μνήμης σε KBytes (να δώσετε ένα καθαρό αριθμό χωρίς τελείες):

β. Να υπολογίσετε το πλήθος των κυψελίδων που απαιτούνται για την υλοποίηση της κρυφής μνήμης (σε κάθε κυψελίδα αποθηκεύεται ένα δυαδικό ψηφίο). Αγνοήστε τα δυαδικά ψηφία που απαιτούνται για την υλοποίηση του μηχανισμού απελευθέρωσης πλαισίων (replacement policy).

Πλήθος κυψελίδων (δώστε ένα καθαρό αριθμό χωρίς τελείες):