ΕΛΛΗΝΙΚΟ ΑΝΟΙΚΤΟ ΠΑΝΕΠΙΣΤΗΜΙΟ
ΣΧΟΛΗ ΘΕΤΙΚΩΝ ΕΠΙΣΤΗΜΩΝ ΚΑΙ ΤΕΧΝΟΛΟΓΙΑΣ
ΠΡΟΓΡΑΜΜΑ ΣΠΟΥΔΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ
ΘΕΜΑΤΙΚΗ ΕΝΟΤΗΤΑ: ΠΛΗ-21

ΑΡΧΙΤΕΚΤΟΝΙΚΗ ΥΠΟΛΟΓΙΣΤΩΝ

ΑΣΚΗΣΕΙΣ
ΓΡΑΠΤΩΝ ΕΡΓΑΣΙΩΝ
&
ΘΕΜΑΤΩΝ ΕΞΕΤΑΣΕΩΝ

ΣΥΝΤΕΛΕΣΤΕΣ (ΚΑΤ' ΑΛΦΑΒΗΤΙΚΗ ΣΕΙΡΑ):
Γ. ΑΛΕΞΙΟΥ
Χ. ΒΕΡΓΟΣ
Κ. ΕΥΣΤΑΘΙΟΥ
Γ. ΘΕΟΔΩΡΙΔΗΣ
Χ. ΚΑΒΟΥΣΙΑΝΟΣ
Ο. ΚΟΥΦΟΠΑΥΛΟΥ
Κ. ΛΑΜΠΡΙΝΟΥΔΑΚΗΣ
Φ. ΛΙΟΤΟΠΟΥΛΟΣ
Α. ΜΟΣΧΟΒΟΣ
Δ. ΜΠΑΚΑΛΗΣ
Σ. ΝΙΚΟΛΑΪΔΗΣ
Δ. ΝΙΚΟΛΟΣ
Β. ΠΑΛΙΟΥΡΑΣ
Ι. ΠΑΠΑΕΥΣΤΑΘΙΟΥ
Δ. ΠΑΠΑΚΩΣΤΑΣ
Α. ΣΤΟΥΡΑΙΤΗΣ
Α. ΣΚΟΔΡΑΣ
Β. ΦΩΤΟΠΟΥΛΟΣ
Α. ΧΑΤΖΟΠΟΥΛΟΣ

ΕΠΙΜΕΛΕΙΑ ΕΚΔΟΣΗΣ: Φ. ΛΙΟΤΟΠΟΥΛΟΣ – Δ. ΜΠΑΚΑΛΗΣ – Χ. ΚΑΒΟΥΣΙΑΝΟΣ

ΠΑΤΡΑ 2008

Το παρόν υλικό αποτελεί το κύριο τμήμα των ασκήσεων που δόθηκαν προς επίλυση στους φοιτητές του Ελληνικού Ανοικτού Πανεπιστημίου στα πλαίσια της Θεματικής Ενότητας **ΠΛΗ-21: Ψηφιακά Συστήματα** του Προγράμματος Σπουδών της Πληροφορικής κατά τα ακαδημαϊκά έτη 2001 - 2008.

Περιεχόμενα

Κωδικοποιήσεις & Αναπαραστάσεις

ΑΣΚΗΣΗ 1

Να μετατραπεί σε κώδικα ASCII (ΕΛΟΤ-928) και να γραφεί σε 16δική μορφή ο τίτλος «ΠΛΗ-21».

Λύση:
Ανατρέχοντας στον πίνακα του κώδικα ASCII (ΕΛΟΤ-928) (σελ.59, τόμος Β'), βρίσκουμε ότι οι δεκαεξαδικές αναπαραστάσεις των χαρακτήρων «Π», «Λ», «Η», «-», «2», «1» είναι: D0, CB, C7, AF, 32, 31, αντίστοιχα. Η δυαδική αναπαράσταση των χαρακτήρων είναι:
11010000 11001011 11000111 10101111 00110010 00110001

ΑΣΚΗΣΗ 2

Να μετατραπεί σε κώδικα ASCII (ΕΛΟΤ-928) και να γραφεί σε 16δική μορφή ο τίτλος «ΕΑΠ © 2002» (=10 ASCII χαρακτήρες).

Λύση:
Ανατρέχοντας στον πίνακα του κώδικα ASCII (ΕΛΟΤ-928) (σελ.59, τόμος Β'), βρίσκουμε ότι οι δεκαεξαδικές αναπαραστάσεις των χαρακτήρων «Ε», «Α», «Π», «κενό», «©»,«κενό», «2», «0», «0», «2» είναι: C5, C1, D0, 20, A9, 20, 32, 30, 30, 32 αντίστοιχα. Η δυαδική αναπαράσταση των χαρακτήρων είναι: 1100 0101, 1100 0001, 1101 0000, 0010 0000, 1010 1001, 0010 0000, 0011 0010, 0011 0000, 0011 0000, 0011 0010.

ΑΣΚΗΣΗ 3

α) Υπολογίστε (σε δεκαδική μορφή και εφαρμόζοντας την τεχνική της περικοπής (truncation)), το μέγιστο σφάλμα αναπαράστασης *E,* που προκαλείται από την αναπαράσταση ενός δυαδικού αριθμού σταθερής υποδιαστολής με 5 ψηφία μετά την υποδιαστολή.
β) Πόσα δυαδικά ψηφία μετά την υποδιαστολή απαιτούνται, ώστε το σφάλμα αναπαράστασης να είναι: $E<10^{-1}$, $E<10^{-2}$, $E<10^{-3}$, $E<10^{-6}$;

Λύση:

α) Με τη διαδικασία της περικοπής, ένας αριθμός x απεικονίζεται στον αριθμό $\hat{x}$, ο οποίος περιλαμβάνει πέντε κλασματικά δυαδικά ψηφία. Το μέγιστο σφάλμα είναι η μεγαλύτερη δυνατή διαφορά του x από τον $\hat{x}$, δηλ.

$$x-\hat{x}=0.0000011111....=2^{-6}+2^{-7}+\ldots=2^{-6}\left(1+\frac{1}{2}+\ldots\right)=2^{-6}\frac{1}{1-\frac{1}{2}}=2^{-6}2=2^{-5}$$

Άρα: $E<2^{-5}=0{,}03125$.

β) Ζητάμε την ελάχιστη τιμή του ακέραιου $m(x)$, για την οποία:
$E<2^{-m}\le 10^{-x}$, για $x=\{1, 2, 3, 6\}$
όπου $m(x)$ ο αριθμός των δυαδικών ψηφίων μετά την υποδιαστολή, για καθεμιά από τις 4 ζητούμενες περιπτώσεις (x={1,2,3,6}).

Επειδή η συνάρτηση $\log_2$ είναι γνησίως αύξουσα η παραπάνω ανισότητα δίνει ισοδύναμα:
$m\ge x\cdot\log_2 10 = x\cdot 3{,}321928095$.

Άρα, για $x=\{1, 2, 3, 6\}$ έχουμε αντίστοιχα: m = {4, 7, 10, 20} bits.

Επαλήθευση:

Bits	Σφάλμα	Bits	Σφάλμα
4	6,2500E-02	3	1,2500E-01
7	7,8125E-03	6	1,5625E-02
10	9,7656E-04	9	1,9531E-03
20	9,5367E-07	19	1,9073E-06

ΑΣΚΗΣΗ 4

α) Υπολογίστε (σε δεκαδική μορφή και εφαρμόζοντας την τεχνική της στρογγυλοποίησης (rounding)), το μέγιστο σφάλμα αναπαράστασης *E,* που προκαλείται από την αναπαράσταση ενός οκταδικού αριθμού σταθερής υποδιαστολής με 4 ψηφία μετά την υποδιαστολή.
β) Πόσα οκταδικά ψηφία μετά την υποδιαστολή απαιτούνται, ώστε το σφάλμα αναπαράστασης να είναι: $E<10^{-1}$, $E<10^{-2}$, $E<10^{-3}$, $E<10^{-6}$;

Λύση:

α) Με τη διαδικασία της στρογγυλοποίησης, ένας αριθμός *x* απεικονίζεται στον αριθμό $\hat{x}$, ο οποίος περιλαμβάνει τέσσερα κλασματικά οκταδικά ψηφία. Το μέγιστο σφάλμα είναι η μεγαλύτερη δυνατή διαφορά του *x* από τον $\hat{x}$, δηλ.

$$|x-\hat{x}| = 0.0000377\ldots_{[8]} = 3\cdot 8^{-5} + 7\cdot 8^{-6} + 7\cdot 8^{-7} + \ldots = 3\cdot 8^{-5} + 7\cdot 8^{-6}\left(1 + 8^{-1} + \ldots\right) =$$

$$= 3\cdot 8^{-5} + 7\cdot 8^{-6}\,\frac{1}{1-8^{-1}} = 3\cdot 8^{-5} + 7\cdot 8^{-6}\,\frac{8}{7} = 3\cdot 8^{-5} + 8^{-5} = 4\cdot 8^{-5} = \frac{1}{2}\cdot 8^{-4}.$$

Άρα: $E < 0{,}5\cdot 8^{-4} = 0{,}0001220703125$.

β) Ζητάμε την ελάχιστη τιμή του ακέραιου $m(x)$, για την οποία:
$E < 0{,}5\cdot 8^{-m} \le 10^{-x}$, για $x = \{1, 2, 3, 6\}$
όπου *m(x)* ο αριθμός των οκταδικών ψηφίων μετά την υποδιαστολή, για καθεμιά από τις 4 ζητούμενες περιπτώσεις ($x=\{1,2,3,6\}$).

Επειδή η συνάρτηση $\log_8$ είναι γνησίως αύξουσα, η παραπάνω ανισότητα δίνει ισοδύναμα:
$-m \le \log_8(2\cdot 10^{-x}) = \log_8 2 - x\cdot\log_8 10$

Άρα:
$m \ge x\cdot\log_8 10 - \log_8 2 = (\log_2 8)^{-1}\cdot(x\cdot\log_2 10 - \log_2 2) = 3^{-1}(x\cdot 3{,}321928 - 1) = x\cdot 1{,}107309 - 3^{-1}$.

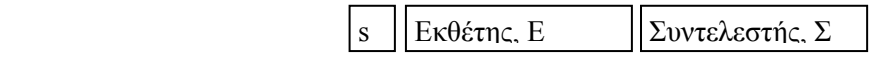
Άρα, για $x = \{1, 2, 3, 6\}$ έχουμε αντίστοιχα: m = {1, 2, 3, 7} bits.

Επαλήθευση:

Οκταδικά ψηφία μετά την υποδιαστολή	Σφάλμα (> ζητούμενου)	Ζητούμενο Σφάλμα $E_{[10]}$	Οκταδικά ψηφία μετά την υποδιαστολή	Σφάλμα (< ζητούμενου)
0	5,0000E-01	1E-01	1	0,6250E-01
1	6,2500E-02	1E-02	2	0,7813E-02
2	7,8125E-03	1E-03	3	0,9766E-03
6	1,9073E-06	1E-06	7	0,2384E-06

ΑΣΚΗΣΗ 5

Θεωρείστε το ακόλουθο δυαδικό format αναπαράστασης αριθμών κινητής υποδιαστολής:



(s=πρόσημο, «0»=θετικός, «1»=αρνητικός)

όπου $N = (-1)^s \cdot (1+\Sigma) \cdot 2^{E-16}$, (αντίστοιχα με τη σχέση 2.1, σελ.56, τόμος Β')

Να αναπαρασταθούν στο παραπάνω format οι παρακάτω δεκαδικοί αριθμοί και να συμπληρωθεί με τις κατάλληλες δυαδικές τιμές ο παρακάτω πίνακας (θεωρώντας ότι εφαρμόζεται η τεχνική της περικοπής (truncation)):

Αριθμός$_{(10)}$	Πρόσημο$_{(2)}$	Εκθέτης$_{(2)}$	Συντελεστής$_{(2)}$
0			
-1			
0,5			
10			
4096			
-0,000125			

Στη συνέχεια, να συμπληρωθεί ο επόμενος πίνακας με τις δεκαδικές τιμές που αντιστοιχούν στις δυαδικές αναπαραστάσεις που δίνονται (σύμφωνα με το ίδιο format):

Πρόσημο$_{(2)}$	Εκθέτης$_{(2)}$	Συντελεστής$_{(2)}$	Αριθμός$_{(10)}$
0	11000	111000	
1	10011	100000	
1	11100	000000	
1	10000	100000	
0	11111	111111	
0	00000	000001	

Λύση:

Δυαδική αναπαράσταση:

α) Συμβατικά, θεωρούμε ότι ο αριθμός μηδέν παριστάνεται με: π = Ε = Σ = 0.
β) $-1 = (-1)^1 \cdot (1+0) \cdot 2^0$, (π=1, Ε=16, Σ=0).
γ) $0,5 = (-1)^0 \cdot (1+0) \cdot 2^{-1}$, (π=0, Ε=15, Σ=0).
δ) $10 = (-1)^0 \cdot (1+0,25) \cdot 2^3$, (π=0, Ε=19, $\Sigma=0,01_2$).
ε) $4096 = (-1)^0 \cdot (1+0) \cdot 2^{12}$, (π=0, Ε=28, Σ=0).
στ) $-0,000125 \approx (-1)^1 \cdot (1+0,000001) \cdot 2^{-13}$, (π=1, Ε=3, $\Sigma=0,000001_2$).

Αριθμός$_{(10)}$	Πρόσημο$_{(2)}$	Εκθέτης$_{(2)}$	Συντελεστής$_{(2)}$
0	0	00000	000000
-1	1	10000	000000
0,5	0	01111	000000
10	0	10011	010000
4096	0	11100	000000
-0,000125	1	00011	000001

Δεκαδική αναπαράσταση: (με εφαρμογή του τύπου)

Πρόσημο$_{(2)}$	Εκθέτης$_{(2)}$	Συντελεστής$_{(2)}$	Αριθμός$_{(10)}$
0	11000	111000	480
1	10011	100000	-12
1	11100	000000	-4096
1	10000	100000	-1,5
0	11111	111111	65024
0	00000	000001	1,55E-05

ΑΣΚΗΣΗ 6

Θεωρείστε το ακόλουθο δυαδικό format αναπαράστασης αριθμών κινητής υποδιαστολής:

(s=πρόσημο, «0»=θετικός, «1»=αρνητικός)

όπου $N = (-1)^s \cdot (1+\Sigma) \cdot 2^{E-8}$, (αντίστοιχα με τη σχέση 2.1, σελ.56, τόμος Β')

Να αναπαρασταθούν στο παραπάνω format οι παρακάτω δεκαδικοί αριθμοί και να συμπληρωθεί με τις κατάλληλες δυαδικές τιμές ο παρακάτω πίνακας (θεωρώντας ότι εφαρμόζεται η τεχνική της περικοπής (truncation)):

Αριθμός$_{(10)}$	Πρόσημο$_{(2)}$	Εκθέτης$_{(2)}$	Συντελεστής$_{(2)}$
0			
-82			
122			
0,0123			
-20			
-0,1			

Στη συνέχεια, να συμπληρωθεί ο επόμενος πίνακας με τις δεκαδικές τιμές που αντιστοιχούν στις δυαδικές αναπαραστάσεις που δίνονται (σύμφωνα με το ίδιο format):

Πρόσημο$_{(2)}$	Εκθέτης$_{(2)}$	Συντελεστής$_{(2)}$	Αριθμός$_{(10)}$
0	1100	11100	
1	1011	10000	
0	1100	00000	
1	1000	10000	
1	1111	11111	
1	0000	00001	

Λύση:
Δυαδική αναπαράσταση:

α) Συμβατικά, θεωρούμε ότι ο αριθμός μηδέν παριστάνεται με: π = Ε = Σ = 0.
β) -82 $= (-1)^1 \cdot (1+ 9 \cdot 2^{-5}) \cdot 2^6$, (π=1, Ε=14, Σ=0,01001).
γ) 122 $= (-1)^0 \cdot (1+29 \cdot 2^{-5}) \cdot 2^6$, (π=0, Ε=14, Σ=0,11101).
δ) 0,0123 $= (-1)^0 \cdot (1+18 \cdot 2^{-5}) \cdot 2^{-7}$, (π=0, Ε= 1, Σ=0,10010).
ε) -20 $= (-1)^1 \cdot (1+ 8 \cdot 2^{-5}) \cdot 2^4$, (π=1, Ε=12, Σ=0,01000).
στ) -0,1 $= (-1)^1 \cdot (1+19 \cdot 2^{-5}) \cdot 2^{-4}$, (π=1, Ε= 4, Σ=0,10011).

Αριθμός$_{(10)}$	Πρόσημο$_{(2)}$	Εκθέτης$_{(2)}$	Συντελεστής$_{(2)}$	Ακριβής Αριθμός$_{(10)}$
0	0	0 = 0000	0 = 00000	0
-82	1	14 = 1110	9 = 01001	-82
122	0	14 = 1110	29 = 11101	122
0,0123	0	1 = 0001	18 = 10010	0,01221
-20	1	12 = 1100	8 = 01000	-20
-0,1	1	4 = 0100	19 = 10011	-0,09961

Δεκαδική αναπαράσταση: (με εφαρμογή του τύπου)

Πρόσημο$_{(2)}$	Εκθέτης$_{(2)}$	Συντελεστής$_{(2)}$	Αριθμός$_{(10)}$
0	12 = 1100	28 = 11100	30
1	11 = 1011	16 = 10000	-12
0	12 = 1100	0 = 00000	16
1	8 = 1000	16 = 10000	-1,5
1	15 = 1111	31 = 11111	-252
1	0 = 0000	1 = 00001	-0,00402832

ΑΣΚΗΣΗ 7

Ένας χρήστης ηλεκτρονικού υπολογιστή πληκτρολογεί έναν δεκαδικό πραγματικό αριθμό (χρησιμοποιώντας σαν διαχωριστή ακεραίου – πραγματικού μέρους την τελεία). Η ASCII αναπαράσταση των χαρακτήρων που πληκτρολόγησε είναι η ακόλουθη (με την σειρά που πληκτρολόγησε τα ψηφία από αριστερά προς τα δεξιά):
«0110010 0110001 0110101 0110110 1110010 0110010 0110101»
α) Ποιον αριθμό εισήγαγε ο χρήστης;
β) Αναπαραστήστε τον αριθμό αυτό στο πρότυπο IEEE 754 με 32 δυαδικά ψηφία.

Λύση:
Σύμφωνα με την ASCII κωδικοποίηση (Πίνακας 2.3 του βιβλίου Αρχιτεκτονικής Υπολογιστών) κάθε δεκαδικό σύμβολο κωδικοποιείται με 7 bits οπότε ο χρήστης εισήγαγε τον παρακάτω δεκαδικό πραγματικό αριθμό:

ASCII κωδικός	0110010	0110001	0110101	0110110	1110010	0110010	0110101
Δεκαδικό Ψηφίο/Σύμβολο	2	1	5	6	.	2	5

Αρχικά θα μετατρέψουμε στο δυαδικό σύστημα τον δεκαδικό αριθμό 2156.25 με τη γνωστή μέθοδο των διαδοχικών διαιρέσεων (πολλαπλασιασμών) με τη βάση για το ακέραιο (δεκαδικό) μέρος.

Έτσι παίρνουμε τη δυαδική αναπαράσταση «100001101100.01»

Για να μπορέσουμε να εκφράσουμε τον παραπάνω αριθμό στην μορφή του προτύπου IEEE 754 με 32 ψηφία θα πρέπει αρχικά να τον κανονικοποιήσουμε. Έτσι ολισθαίνουμε τον παραπάνω αριθμό προς τα δεξιά έως ότου αριστερά της υποδιαστολής να μείνει ένα μόνο δυαδικό ψηφίο ίσο με την μονάδα. Για κάθε μία ολίσθηση (που ισοδυναμεί με διαίρεση του αριθμού με το 2) πολλαπλασιάζουμε τον αριθμό με το 2 ώστε να διατηρηθεί σταθερός. Έτσι μετά από 11 ολισθήσεις παίρνουμε τελικά τον αριθμό «1.0000110110001» ο οποίος πολλαπλασιαζόμενος με τον δεκαδικό αριθμό 2^{11} θα δώσει τον αρχικό αριθμό. Άρα έχουμε:

Πρόσημο	+	(Θετικός αριθμός)
Εκθέτης (δεκαδικός)	Μη πολωμένος = 11	Πολωμένος = 127+11=138
Συντελεστής	$(1.0000110110001)_2 = (1+0{,}052734375)_{10}$	

Συνεπώς ο αριθμός είναι ο $(-1)^0$ x (1+0,052734375) x $2^{138-127}$ και σε δυαδική αναπαράσταση

0	10001010	00001101100010000000000

ΑΣΚΗΣΗ 8

1. Στους αριθμούς που ακολουθούν να εφαρμόσετε τη μέθοδο
α. της περικοπής και
β. της στρογγυλοποίησης,
ώστε το συνολικό πλήθος των ψηφίων τους να περιοριστούν σε πέντε (η υποδιαστολή δεν λαμβάνεται υπόψη σαν ψηφίο).
$0.3726579_{(10)}$
$0.372531_{(10)}$
$0.10111111_{(2)}$
$0.11000111_{(2)}$
$0.5372637_{(8)}$
$0.3271007_{(8)}$

2. Σε κάθε περίπτωση να υπολογίσετε το σφάλμα περικοπής και το σφάλμα στρογγυλοποίησης και να τα συγκρίνετε.

Λύση:

Εφαρμόζοντας την περικοπή περιορίζουμε τον κάθε αριθμό στα πέντε ψηφία, ενώ για να τον στρογγυλοποιήσουμε, πριν την περικοπή προσθέτουμε στον αριθμό το $\beta^{\lambda}/2$ (όπου β η βάση του συστήματος αρίθμησης και β^{λ} το βάρος του λιγότερο σημαντικού ψηφίου που διατηρείται) και κατόπιν κάνουμε την περικοπή.

Έτσι έχουμε.

α.α	ΑΡΙΘΜΟΣ	ΠΕΡΙΚΟΠΗ	$\beta^{\lambda}/2$	ΣΤΡΟΓΓΥΛΟΠΟΙΗΣΗ
1	$0.3726579_{(10)}$	$0.3726_{(10)}$	$0.00005_{(10)}$	$0.3727_{(10)}$
2	$0.372531_{(10)}$	$0.3725_{(10)}$	$0.00005_{(10)}$	$0.3725_{(10)}$
3	$0.10111111_{(2)}$	$0.1011_{(2)}$	$0.00001_{(2)}$	$0.1100_{(2)}$
4	$0.11000111_{(2)}$	$0.1100_{(2)}$	$0.00001_{(2)}$	$0.1100_{(2)}$
5	$0.5372637_{(8)}$	$0.5372_{(8)}$	$0.00004_{(8)}$	$0.5373_{(8)}$
6	$0.3271007_{(8)}$	$0.3271_{(8)}$	$0.00004_{(8)}$	$0.3271_{(8)}$

2. Σε κάθε περίπτωση να υπολογίσετε το σφάλμα περικοπής και το σφάλμα στρογγυλοποίησης και να τα συγκρίνετε.

ΣΦΑΛΜΑΤΑ:

α.α	ΠΕΡΙΚΟΠΗΣ	ΣΤΡΟΓ/ΗΣΗΣ	ΣΥΓΚΡΙΣΗ
1	$0.0000579_{(10)}$	$0.0000421_{(10)}$	Σφάλμα περικοπής μεγαλύτερο.
2	$0.000031_{(10)}$	$0.000031_{(10)}$	Σφάλματα ίδια.
3	$0.00001111_{(2)}$	$0.00000001_{(2)}$	Σφάλμα περικοπής μεγαλύτερο.
4	$0.00000111_{(2)}$	$0.00000111_{(2)}$	Σφάλματα ίδια.
5	$0.0000637_{(8)}$	$0.0000141_{(8)}$	Σφάλμα περικοπής μεγαλύτερο.
6	$0.0000007_{(8)}$	$0.0000007_{(8)}$	Σφάλματα ίδια.

Παρατηρούμε ότι το σφάλμα στρογγυλοποίησης είναι πάντα μικρότερο ή ίσο του σφάλματος που προκύπτει εφαρμόζοντας την τεχνική της περικοπής.

ΑΣΚΗΣΗ 9

1. Η αρτηρία διευθύνσεων ενός υπολογιστικού συστήματος έχει εύρος 32 δυαδικών ψηφίων. Αν σε κάθε θέση μνήμης μπορούμε να αποθηκεύουμε πληροφορία 16 δυαδικών ψηφίων, ποια είναι η μέγιστη φυσική μνήμη που μπορεί να έχει αυτό το σύστημα ?
2. Ένα υπολογιστικό σύστημα διαθέτει κ διαφορετικές εντολές (π.χ. ADD, LOAD, STORE, …). Σε κάθε εντολή, ένα τελούμενο μπορεί να διευθυνσιοδοτείται με λ διαφορετικούς τρόπους.
   - Ποιος είναι ο ελάχιστος αριθμός δυαδικών ψηφίων της εντολής που θα πρέπει να αφιερωθεί για κωδικό λειτουργίας και ποιος για το τρόπο διευθυνσιοδότησης ?
   - Προτείνετε εναλλακτική κωδικοποίηση αυτών των δύο πεδίων που να ελαχιστοποιεί τα χρησιμοποιούμενα δυαδικά ψηφία της εντολής.

Λύση :

1. Σε κάθε θέση μνήμης μπορούμε να αποθηκεύουμε 16 δυαδικά ψηφία ή αλλιώς 2 bytes. Οι διαφορετικές θέσεις φυσικής μνήμης που μπορούμε να έχουμε είναι όλοι οι πιθανοί συνδυασμοί των ψηφίων της αρτηρίας διευθύνσεων δηλαδή 2^{32}. Αρα η μέγιστη φυσική μνήμη στο σύστημά μας είναι 2×2^{32} bytes = $2^3 \times 2^{30}$ bytes = 8 GB.
2. Αν διαθέταμε ξεχωριστά πεδία στην εντολή μας για κωδικό λειτουργίας και τρόπο διευθυνσιοδότησης θα χρειαζόμασταν $x = \lceil \log_2 \kappa \rceil$ και $y = \lceil \log_2 \lambda \rceil$ δυαδικά ψηφία αντίστοιχα. Ο συνολικός αριθμός αυτών είναι $A = x + y = \lceil \log_2 \kappa \rceil + \lceil \log_2 \lambda \rceil$. Για παράδειγμα, αν υποθέσουμε ότι κ=25 και λ=5, θα χρειαζόμασταν 5+3=8 δυαδικά ψηφία.
   Αφού ισχύει ότι $\kappa \le 2^x$ και $\lambda \le 2^y$, πολλαπλασιάζοντας κατά μέλη παίρνουμε ότι $\kappa\lambda \le 2^{x+y}$ και συνεπώς $\lceil \log_2 (\kappa\lambda) \rceil \le x+y \Leftrightarrow \lceil \log_2 (\kappa\lambda) \rceil \le \lceil \log_2 \kappa \rceil + \lceil \log_2 \lambda \rceil$. Συνεπώς ένας τρόπος ελαχιστοποίησης των δυαδικών ψηφίων που χρησιμοποιούνται είναι η συνένωση αυτών των δύο πεδίων σε ένα, το οποίο καθορίζει τόσο τον κωδικό λειτουργίας όσο και το τρόπο διευθυνσιοδότησης. Οι διαφορετικές καταστάσεις αυτού του πεδίου είναι κλ και συνεπώς ο ελάχιστος αριθμός δυαδικών ψηφίων που θα πρέπει να αφιερωθεί γι' αυτό το πεδίο είναι $B = \lceil \log_2(\kappa\lambda) \rceil$. Για το παράδειγμά μας είναι B = 7.

ΑΣΚΗΣΗ 10

Ένας υπολογιστής χρησιμοποιεί μια δική του αναπαράσταση για τους αριθμούς κινητής υποδιαστολής με τις εξής παραδοχές :

- Προκαθορισμένη βάση αναπαράστασης το 16.
- Συνολικό εύρος αναπαράστασης 16 δυαδικά ψηφία.
- Το αριστερότερο δυαδικό ψηφίο παριστάνει το πρόσημο του αριθμού. 1 σημαίνει αρνητικός αριθμός.
- Για τον εκθέτη αφιερώνονται τα 3 επόμενα δυαδικά ψηφία σε παράσταση πόλωσης κατά 4.
- Για το μέτρο του αριθμού αφιερώνονται τα υπόλοιπα 12 δυαδικά ψηφία. Χρησιμοποιείται γι' αυτά η δυαδική αναπαράσταση τριών δεκαεξαδικών ψηφίων. Το δεκαδικό σημείο (υποδιαστολή) θεωρήστε ότι βρίσκεται ακριβώς πριν από αυτά τα 12 δυαδικά ψηφία.
- Οι αριθμοί αποθηκεύονται κανονικοποιημένοι.

α) Δώστε την αναπαράσταση του $-0{,}046875_{10}$ σε αυτόν τον υπολογιστή.
β) Ποιος αριθμός του δεκαδικού συστήματος αναπαρίσταται ως 0 111 1000 0011 0111 ?

Λύση :

α) Είναι $0{,}046875_{10} = 0{,}0C_{16}$ όπως μπορούμε να βρούμε με διαδοχικούς πολλαπλασιασμούς με το 16. Μιας και $0{,}0C_{16} = 0{,}0C_{16} * 16^0 = 0{,}C_{16} * 16^{-1}$, έχουμε ότι σε κανονικοποιημένη μορφή με βάση το 16 ισχύει ότι : $-0{,}046875_{10} = -0{,}C_{16} * 16^{-1} = -0{,}C00_{16} * 16^{-1}$. Για τον εκθέτη που έχει πόλωση κατά 4, η αναπαράστασή του θα έχει τιμή x, έτσι ώστε $x - 4 = -1 \Leftrightarrow x = 3$. Συνεπώς η ζητούμενη αναπαράσταση για τον εκθέτη είναι $3_{10} = 011_2$. Στα ψηφία μέτρου θα υπάρχει η δυαδική αναπαράσταση του $C00_{16}$, δηλαδή η 1100 0000 0000. Στο ψηφίο προσήμου θα υπάρχει

το 1, αφού ο αριθμός είναι αρνητικός. Συνεπώς η ζητούμενη αναπαράσταση είναι συνολικά : 1 011 1100 0000 0000.

β) Το δυαδικό ψηφίο προσήμου ισούται με 0, άρα πρόκειται για θετικό αριθμό. Στα ψηφία εκθέτη έχουμε $111_2 = 7_{10}$. Αφαιρώντας τη πόλωση (4) βρίσκουμε ότι ο εκθέτης της κανονικοποιημένης παράστασης είναι 7 – 4 = 3. Τέλος στα ψηφία μέτρου έχουμε 1000 0011 $0111_2 = 837_{16}$. Λαμβάνοντας υπόψη την θέση της υποδιαστολής συμπεραίνουμε ότι η παραπάνω παράσταση αντιπροσωπεύει τον αριθμό με μέτρο $0{,}837_{16}$. Άρα συνολικά η παράσταση αντιπροσωπεύει τον αριθμό $+0{,}837_{16}*16^3 = 837_{16}*16^0 = 8*16^2 + 3*16^1 + 7*16^0 = 2103_{10}$.

ΑΣΚΗΣΗ 11

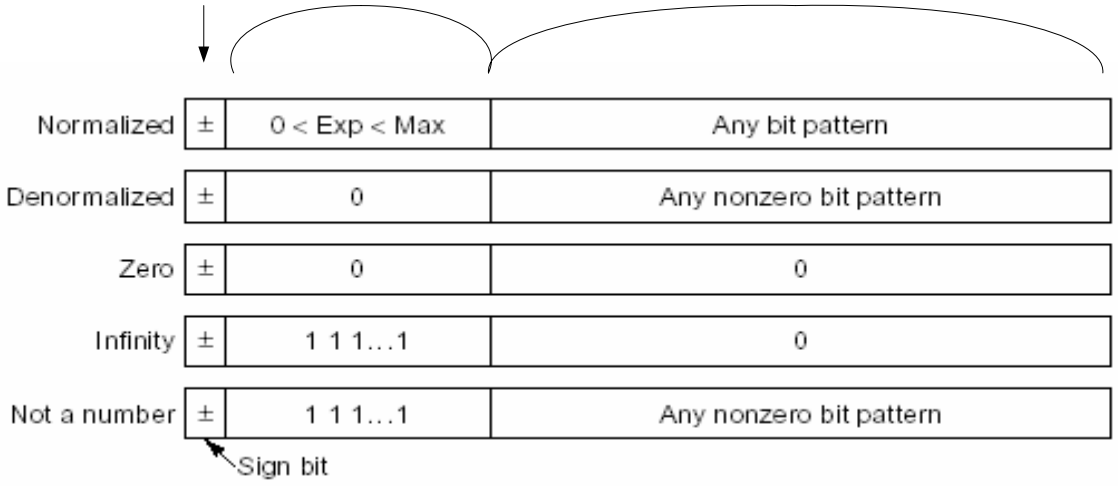
Στην παράσταση αριθμών κινητής υποδιαστολής σύμφωνα με το πρότυπο IEEE 754 εκτός των κανονικοποιημένων (normalized) αριθμών αναπαρίστανται άλλοι 4 τύποι αριθμών (denormalized, zero, infinity, not a number (NaN)).

1. Δώστε υπό μορφή πίνακα όλους τους τύπους των αριθμών του προτύπου IEEE 754 για παραστάσεις απλής ακριβείας (single precision).
2. Για κάθε έναν από του επόμενους αριθμούς κινητής υποδιαστολής απλής ακρίβειας να βρεθεί ποιον αριθμό του δεκαδικού συστήματος παριστάνουν.
   1011 1101 0100 0000 0000 0000 0000 0000
   0101 0101 0110 0000 0000 0000 0000 0000
   0111 1111 1000 1111 0000 1111 0000 0000
   1111 1111 1000 1111 0000 1111 0000 0000
3. Να χρησιμοποιηθούν αριθμοί κινητής υποδιαστολής απλής ακριβείας για να εκτελεστούν οι πράξεις
   a. 32 x 16
   b. 147,5 + 0,25

(***Υπόδειξη*** : Το πρότυπο IEEE 754 περιγράφεται επαρκώς στην βιβλιογραφία και το διαδίκτυο).

Λύση

1.



2. Στη παράσταση 1011 1101 0100 0000 0000 0000 0000 0000 το bit πρόσημου είναι 1 άρα αντιπροσωπεύει αρνητικό αριθμό, τα δυαδικά ψηφία του εκθέτη είναι 01111010 = 122 άρα ο εκθέτης είναι 122–127 =–5 και το κλασματικό μέρος 100 0000 0000 0000 0000 0000. Άρα πρόκειται για τη παράσταση του $-1{,}1_2 \times 2^{-5} = -0{,}046875$.
   Στη παράσταση 0101 0101 0110 0000 0000 0000 0000 0000 το bit πρόσημου είναι 0 άρα αντιπροσωπεύει θετικό αριθμό, τα δυαδικά ψηφία του εκθέτη είναι 10101010 = 170 άρα ο

εκθέτης είναι 170 – 127 = 43 και το κλασματικό μέρος 110 0000 0000 0000 0000 0000. Άρα ο αριθμός είναι $1,11_2 \times 2^{43} = +1,539 \times 10^{13}$.
Για τις παραστάσεις 0111 1111 1000 1111 0000 1111 0000 0000, και 1111 1111 1000 1111 0000 1111 0000 0000 ο εκθέτης είναι 11111111 με κλασματικό μέρος διαφορετικό του μηδενός, άρα αντιπροσωπεύουν το NaN (not a number).

3.

a. $32 = 2^5$, $16 = 2^4$ και οι αναπαραστάσεις τους είναι 0100 0010 0000 0000 0000 0000 0000 0000 και 0100 0001 1000 0000 0000 0000 0000 0000 αντίστοιχα. Για να πολλαπλασιασθούν, προσθέτουμε τους εκθέτες και αφαιρούμε μία φορά τη πόλωση και πολλαπλασιάζουμε τα κλασματικά μέρη. Το αποτέλεσμα είναι 0100 0100 0000 0000 0000 0000 0000 0000 = 2^9 = 512.

b. $147,5_{10} = 1,00100111_2 \times 2^7$ = 0100 0011 0001 0011 1000 0000 0000 0000 και
$0,25_{10} = 1,00000000_2 \times 2^{-2}$ = 0011 1110 1000 0000 0000 0000 0000 0000
Για να τους προσθέσουμε, ολισθαίνουμε τον αριθμό με τον μικρότερο εκθέτη ώστε να εξισωθούν οι εκθέτες δηλ. $0,25_{10} = 0,000000001_2 \times 2^7$. Προσθέτουμε τα κλασματικά μέρη και παίρνουμε σαν αποτέλεσμα τον αριθμό
$147,75_{10} = 1,001001111_2 \times 2^7$ = 0100 0011 0001 0011 1100 0000 0000 0000

ΑΣΚΗΣΗ 12

Έστω δύο υπολογιστικά συστήματα Α και Β οι λέξεις των οποίων είναι αντίστοιχα **8** και **16** δυαδικά ψηφία. Τα Α και Β χρησιμοποιούν αριθμητική **συμπληρώματος ως προς 2 για τους ακεραίους** και IEEE κινητής υποδιαστολής απλής ακρίβειας (single precision floating point standard) για τους **πραγματικούς** αριθμούς.

1. Δώστε την αναπαράσταση των αριθμών **$X=-95_{10}$**, **$Y= 67_{10}$** και **$Z = -67,625_{10}$** στα Α και Β.
2. Εκτελέστε τις πράξεις **X+Y** και **X-Y** στα Α και Β. Δώστε τα αποτελέσματα στο δυαδικό και στο δεκαδικό. Επίσης δώστε τη τιμή των σημαιών κατάστασης αρνητικού (**N**), υπερχείλισης (**V**) και κρατουμένου (**C**) μετά από κάθε πράξη.

Λύση

1.

a. Είναι $95_{10} = 0101\ 1111_2$ όπως μπορούμε να βρούμε με διαδοχικές διαιρέσεις. Σε 16 δυαδικά ψηφία θα είναι προφανώς 0000 0000 0101 1111_2. Το -95_{10} θα αναπαρίσταται σα το συμπλήρωμα ως προς 2 του 95_{10}, δηλαδή στο σύστημα Α ως 1010 0001, ενώ στο σύστημα Β ως 1111 1111 1010 0001.
b. Το 67_{10} έχει παράσταση στο δυαδικό 0100 0011_2 και σε 16 ψηφία 0000 0000 0100 0011_2. Αφού ο Υ είναι θετικός θα έχει τις ίδιες αναπαραστάσεις και σε αριθμητική συμπληρώματος ως προς 2.
c. $67_{10} = 0100\ 0011_2$. Με τη μέθοδο των διαδοχικών πολλαπλασιασμών μπορούμε να βρούμε ότι $0,625_{10}$ = **0,10100000** … Συνεπώς είναι $-67,625_{10}$ = -0100 0011,101000… x 2^0 = -1, 000011101000… x 2^6. Άρα για την IEEE single precision αναπαράσταση θα έχουμε :
    i. Bit προσήμου = 1 αφού ο αριθμός είναι αρνητικός.
    ii. Εκθέτης = 6 σε πόλωση κατά 127= 133 = 1000 0101.
    iii. Σημαντικό μέρος 1,000011101000… Αποκόπτουμε το κρυμμένο δυαδικό ψηφίο και έχουμε σε 23 bit 0000 1110 1000 …

2. Η πράξη X+Y είναι πρόσθεση ετερόσημων άρα δε μπορεί να υπάρξει υπερχείλιση. Συνεπώς έχουμε:
   a. Στο σύστημα A : X+Y = 1010 0001 + 0100 0011 = 1110 0100. Το αποτέλεσμα είναι σε παράσταση συμπληρώματος του 2. Το πρώτο δυαδικό ψηφίο είναι 1 άρα είναι αρνητικός και πρόκειται για τον αριθμό -28_{10}. Στην πρόσθεση αυτή δε προέκυψε κρατούμενο εξόδου (C=0), το αποτέλεσμα είναι αρνητικό (N=1) και δεν υπήρξε υπερχείλιση (V=0).
   Στο σύστημα B είναι X+Y = 1111 1111 1010 0001 + 0000 0000 0100 0011 = 1111 1111 1110 0100 = -28_{10}. Οι σημαίες κατάστασης είναι ίδιες με τη προηγούμενη περίπτωση.

b. Η αφαίρεση X-Y μπορεί να προκαλέσει υπερχείλιση. Στο υπολογιστικό σύστημα A που διαθέτει 8 δυαδικά ψηφία, ο μικρότερος αρνητικός που μπορούμε να παραστήσουμε είναι ο -128, ενώ από το X-Y περιμένουμε αποτέλεσμα -95-67 = -162. Αντίθετα λόγω των 16 δυαδικών ψηφίων του υπολογιστικού συστήματος B η πράξη αναμένουμε να εκτελεστεί σωστά. Όταν χρησιμοποιούμε συμπλήρωμα ως προς 2 η X-Y θα εκτελεστεί ως X+(-Y) όπου το –Y είναι το συμπλήρωμα ως προς 2 του Y, δηλαδή στο σύστημα A το 1011 1101 και στο υπολογιστικό σύστημα B το 1111 1111 1011 1101.
   Στο υπολογιστικό σύστημα A, X-Y = 1010 0001 + 1011 1101 = 0101 1110 = +1011110 !!! λανθασμένο αποτέλεσμα λόγω υπερχείλισης. Επίσης είναι N=0, C=1 και V=1.
   Στο υπολογιστικό σύστημα B, X-Y = 1111 1111 1010 0001 + 1111 1111 1011 1101 = 1111 1111 0101 1110, δηλαδή ο – $0000\ 0000\ 1010\ 0010_2 = -162_{10}$. Επίσης είναι C=1, N=1 και V=0.

ΑΣΚΗΣΗ 13

Ένας υπολογιστής χρησιμοποιεί την ακόλουθη αναπαράσταση αριθμών κινητής υποδιαστολής:

$$(-1)^{\pi} \times (1,yyy\ldots y) \times 2^{(\text{εκθέτης-πόλωση})}$$

- Συνολικό εύρος αναπαράστασης 20 δυαδικά ψηφία.
- Το αριστερότερο δυαδικό ψηφίο (π) παριστάνει το πρόσημο του αριθμού. Η τιμή 1 δηλώνει ότι ο αριθμός είναι αρνητικός.
- Για τον εκθέτη αφιερώνονται τα 6 επόμενα δυαδικά ψηφία σε παράσταση πόλωσης 31_{10}.
- Για το συντελεστή yyy…y αφιερώνονται τα υπόλοιπα 13 δυαδικά ψηφία. Προσέξτε ότι οι αριθμοί είναι κανονικοποιημένοι (το πιο σημαντικό δυαδικό ψηφίο του αριθμού 1,yyy…y δηλαδή το 1 δεν καταγράφεται στην παράσταση του συντελεστή).

α) Δώστε την αναπαράσταση (κανονικοποιημένη) του $-0,046875_{10}$.
β) Ποιος αριθμός του δεκαδικού συστήματος αναπαρίσταται ως
0 100001 1010000000000
γ) Να βρεθεί το γινόμενο (σε κανονικοποιημένη μορφή) των αριθμών (οι υπολογισμοί να γίνουν στο δυαδικό σύστημα και να επιβεβαιωθούν με μετατροπές στο δεκαδικό).
0 100001 1010000000000
1 100010 1011000000000

Λύση

α)
$-0,046875_{10} = -0,00001100_2 = -1,100 \times 2^{-5}$
άρα
πρόσημο =1
εκθέτης= -5+31=26=011010
μέτρο = 1000000000000
$-0,046875_{10}$ = 1 011010 1000000000000

β)
0 100001 1010000000000
πρόσημο +
100001 = 33 Άρα εκθέτης = 33-31 =2
Ο αριθμός είναι $(-1)^0$x1,101x2^2 =+110,1=+$6,5_{10}$

γ)
Πρόσημο 1⊕0=1
Εκθέτης 100001+100010-011111=100100 (εκθέτης1+εκθέτης2-πόλωση)
Μέτρο

```
  1,1011
 x 1,101
 ---------
   11011
  00000
 11011
11011
--------------
```

$10,1011111 = 1,01011111x2^1$

Μέτρο 0101111100000 και ο εκθέτης αυξάνεται κατά 1, δηλαδή εκθέτης =100101
Άρα ο ζητούμενος αριθμός είναι ο 1 100101 0101111100000
Εάν επαληθεύσουμε την παραπάνω πράξη στο δεκαδικό σύστημα θα δούμε ότι
0 100001 1010000000000 = $+6,5_{10}$
1 100010 1011000000000 = $-13,5_{10}$
1 100101 0101111100000 = $-87,75_{10}$
ενώ έχουμε +6,5 x (-13,5) = $-87,75_{10}$.

ΑΣΚΗΣΗ 14

Δίνονται οι παρακάτω αριθμοί κινητής υποδιαστολής σύμφωνα με το πρότυπο (standard) IEEE 754:

α : 0 10000111 01001110000000000000000
β : 0 10010000 00110000000000000000000
γ : 1 10011101 11110000000000000000000

α) Να μετατρέψετε τους αριθμούς σε δεκαδική μορφή.
β) Να εκτελέσετε τις πράξεις α+β, α · β και β – γ χρησιμοποιώντας τη μορφή της κινητής υποδιαστολής για τους αριθμούς α, β και γ. Να δοθεί η εκτέλεση των πράξεων βήμα προς βήμα.
γ) Μετατρέψτε τα αποτελέσματα του ερωτήματος β σε δεκαδική μορφή και επαληθεύστε τα αποτελέσματα κάνοντας τις πράξεις στο δεκαδικό σύστημα χρησιμοποιώντας τους δεκαδικούς που υπολογίσατε στο ερώτημα α.

Λύση:

Ερώτημα α

Για τον αριθμό α ισχύει ότι: 0 10000111 01001110000000000000000
άρα
πρόσημο =0
εκθέτης (πολωμένος) =10000111 = 135
συντελεστής =01001110000000000000000 = $\frac{1}{4}+\frac{1}{32}+\frac{1}{64}+\frac{1}{128}$ = =0.3046875
Συνεπώς α = $(-1)^0$x $(1+0.3046875)x2^{135-127}$ = **334**

Για τον αριθμό β ισχύει ότι: 0 10010000 00110000000000000000000
άρα
πρόσημο =0
εκθέτης (πολωμένος) =10010000 = 144
συντελεστής =00110000000000000000000 = $\frac{1}{8}+\frac{1}{16}$ = 0.1875
Συνεπώς β = $(-1)^0$x (1+0.1875)x$2^{144-127}$ = **155648**

Για τον αριθμό γ ισχύει ότι: 1 10011101 11110000000000000000000
άρα
πρόσημο =1
εκθέτης (πολωμένος) =10011101 = 157
συντελεστής =11110000000000000000000 = $\frac{1}{2}+\frac{1}{4}+\frac{1}{8}+\frac{1}{16}$ = 0.9375
Συνεπώς γ = $(-1)^1$x (1+0.9375)x$2^{157-127}$ = **-2080374784**

Ερώτημα β

Η πράξη α + β:
Οι αριθμοί α και β είναι ομόσημοι και θετικοί άρα το πρόσημο του αποτελέσματος θα είναι 0. Παρατηρούμε ότι ο β έχει μεγαλύτερο εκθέτη από τον α, συνεπώς αποκανονικοποιούμε τον α έτσι ώστε να έχει τον ίδιο εκθέτη με τον β. Αυτό επιτυγχάνεται με συνεχείς ολισθήσεις του συντελεστή του α προς τα δεξιά, κάθε μία από τις οποίες συνοδεύεται με αύξηση του εκθέτη του α κατά 1.

Η διαφορά των δύο εκθετών είναι:
εκθέτης (β) – εκθέτης (α): 10010000 = 144
10000111 = 135
00001001 = 9
Άρα πρέπει να ολισθήσουμε τον συντελεστή του α προς τα δεξιά κατά 9 θέσεις και γίνεται 00000000101001110000000 ενώ ο εκθέτης του γίνεται 10010000. Τώρα μπορούμε να προσθέσουμε τους συντελεστές χρησιμοποιώντας και το ψηφίο πριν την υποδιαστολή αφού ο α είναι πλέον αποκανονικοποιημένος,

0.00000000101001110000000
\+ 1.00110000000000000000000
1.00110000101001110000000

Παρατηρούμε ότι το αποτέλεσμα είναι κανονικοποιημένο οπότε μαζί με το πρόσημο και τον κοινό εκθέτη των α, β μας δίνει τον παρακάτω αριθμό στη μορφή του προτύπου:

0 10010000 00110000101001110000000

Η πράξη α β:
Πρόσημο αποτελέσματος είναι: 0 XOR 0 = 0
Υπολογισμός εκθέτη αποτελέσματος ως εξής: Πρόσθεση των εκθετών και διόρθωση αφαιρώντας την πόλωση

10000111 = 135
\+ 10010000 = 144
1)00010111 = 279

και διορθώνουμε

100010111 = 279
\- 01111111 = 127
10011000 = 152

Στη συνέχεια υπολογίζουμε τον συντελεστή του αποτελέσματος, πολλαπλασιάζοντας τους αυξημένους κατά ένα συντελεστές των όρων του γινομένου:

```
      1.0100111(0000000000000000
×     1.0011(0000000000000000000
       10100111
      10100111
     00000000
    00000000
1 0100111
1.10001100101000000000000000000000000000000000000
```

Παρατηρούμε ότι είναι κανονικοποιημένος ο συντελεστής του αποτελέσματος, οπότε για να πάρουμε τη μορφή του στο πρότυπο απορρίπτουμε το περισσότερο σημαντικό ψηφίο και από τα υπόλοιπα δυαδικά ψηφία κρατάμε τα 23 πλέον σημαντικά (στρογγύλευση ή περικοπή) και παίρνουμε 10001100101000000000000.

Άρα κατά το πρότυπο IEEE 754 το αποτέλεσμα γράφεται ως:
0 10011000 10001100101000000000000.

Η πράξη β – γ:

Αρχικά παρατηρούμε ότι έχουμε να αφαιρέσουμε έναν αρνητικό από έναν θετικό, άρα θα προσθέσουμε στο β το – γ το οποίο είναι θετικός. Συνεπώς, το αποτέλεσμα θα είναι θετικός και το πρόσημο του αποτελέσματος θα είναι 0. Παρατηρούμε ότι ο γ έχει μεγαλύτερο εκθέτη από τον β, συνεπώς αποκανονικοποιούμε τον β έτσι ώστε να έχει τον ίδιο εκθέτη με τον γ. Αυτό επιτυγχάνεται με συνεχείς ολισθήσεις του συντελεστή του β προς τα δεξιά, κάθε μία από τις οποίες συνοδεύεται με αύξηση του εκθέτη του β κατά 1.

Η διαφορά των δύο εκθετών είναι:
εκθέτης (γ) – εκθέτης (β): 10011101 = 157
10010000 = 144

00001101 = 13

Άρα πρέπει να ολισθήσουμε τον συντελεστή του β κατά 13 θέσεις προς τα δεξιά οπότε γίνεται 00000000000010011000000 ενώ ο εκθέτης του γίνεται 10011101. Τώρα μπορούμε να προσθέσουμε τους συντελεστές των β και (-γ) χρησιμοποιώντας και το ψηφίο πριν την υποδιαστολή αφού ο β είναι πλέον αποκανονικοποιημένος,

```
          0.00000000000010011000000
    +     1.11110000000000000000000
          1.11110000000010011000000
```

Παρατηρούμε ότι το αποτέλεσμα είναι κανονικοποιημένο οπότε μαζί με το πρόσημο και τον κοινό εκθέτη των β, (-γ) μας δίνει τον παρακάτω αριθμό στη μορφή του προτύπου:

0 10011101 11110000000010011000000

Ερώτημα γ

Αρχικά θα μετατρέψουμε τα αποτελέσματα του ερωτήματος β σε δεκαδική μορφή ακολουθώντας την μέθοδο που περιγράψαμε στο ερώτημα α.
α+β = 0 10010000 00110000101001110000000= 1,1900482177734375 x 2^{17}=**155982**
α.β = 0 10011000 10001100101000000000000= 1,54931640625 x 2^{25}=**51986432**.
β-γ = 0 10011101 11110000000010011000000= 1,93764495849609375 x 2^{30}=
= 2080530432

Τώρα θα κάνουμε τις πράξεις στο δεκαδικό με τα αποτελέσματα του ερωτήματος α για να επαληθεύσουμε τα αποτελέσματα του ερωτήματος β.
α+β=334 + 155648=155982
α.β =334 x 155648= 51986432
β-γ =155648-(-2080374784)= 2080530432.

Παρατηρούμε ότι πράγματι είναι τα ίδια.

ΑΣΚΗΣΗ 15

Δίνονται οι παρακάτω αριθμοί κινητής υποδιαστολής σύμφωνα με το πρότυπο IEEE 754:
α. 01000001011000000000000000000000
β. 01000011110010000000000000000000
γ. 11100110011000000000000000000000
Να εκτελεστούν οι πράξεις α+β, α · β και β – γ και να δοθούν τα αποτελέσματα τόσο σε μορφή κινητής υποδιαστολής σύμφωνα με το στάνταρτ IEEE 754 όσο και στη δεκαδική μορφή που χρησιμοποιούμε στην καθημερινή μας ζωή (π.χ. 9.85×10^{-3}).
Να δοθεί η εκτέλεση των πράξεων βήμα προς βήμα.

Λύση:
Για τον αριθμό α ισχύει ότι: 0 10000010 11000000000000000000000
άρα
πρόσημο = 0
εκθέτης = 10000010 = 130
συντελεστής = 11000000000000000000000 = 0.5 + 0.25 = 0.75
Συνεπώς α = $(-1)^0 \times (1+0.75) \times 2^{130-127} = 1 \times 1.75 \times 2^3 = 14 = 1.4 \times 10^1$

Για τον αριθμό β ισχύει ότι: 0 10000111 10010000000000000000000
άρα
Πρόσημο = 0
Εκθέτης = 10000111 = 135
Συντελεστής = 10010000000000000000000 = 0.5 + 0.0625 = 0.5625
Συνεπώς η τιμή του β είναι
$(-1)^0 \times (1+0.5625) \times 2^{135-127} = 1 \times 1.5625 \times 2^8 = 400 = 4.0 \times 10^2$

Για τον αριθμό γ ισχύει ότι 1 11001100 11000000000000000000000
Πρόσημο = 1
Εκθέτης = 11001100 = 204
Συντελεστής = 11000000000000000000000 = 0.5 + 0.25 = 0.75
Άρα η τιμή του γ είναι
$(-1)^1 \times (1+0.75) \times 2^{204-127} = -1 \times 1.75 \times 2^{77} = -264452523040700131966976$
$= -264.453 \times 10^{21}$

Η πράξη α + β:
Οι αριθμοί α και β είναι ομόσημοι και θετικοί άρα το πρόσημο του αποτελέσματος θα είναι 0.

Παρατηρούμε ότι ο *β* έχει μεγαλύτερο εκθέτη από τον *α*, συνεπώς αποκανονικοποιούμε τον α έτσι ώστε να έχει τον ίδιο εκθέτη με τον β,

$\alpha = 1.75 \times 2^3 = 1.75 \times 2^{8-5} = 1.75 \times 2^{-5} \times 2^8 = 0.0546875 \times 2^8$
Στη συνέχεια εκτελούμε την πράξη
0.0546875 + 1.5625 = 1.6171875 = 1.10011110000000000000000 (σε βάση 2),
το οποίο είναι σε κανονικοποιημένη μορφή γιατί είναι μεγαλύτερο του 1 και μικρότερο από 2. Άρα το αποτέλεσμα είναι $1.6171875 \times 2^8 = 414$. Αυτό σε μορφή IEEE 754 γράφεται 0 10000111 10011110000000000000000

Η πράξη α β:
Πρόσημο αποτελέσματος είναι: 0 XOR 0 = 0

Υπολογισμός εκθέτη αποτελέσματος ως εξής: Πρόσθεση των εκθετών και διόρθωση αφαιρώντας την πόλωση
130 + 135 – 127 = 138
Στη συνέχεια υπολογίζουμε το συντελεστή του αποτελέσματος, πολλαπλασιάζοντας τους αυξημένους κατά ένα συντελεστές των όρων του γινομένου:
1.1001(0000000000000000000
× 1.11(00000000000000000000
11001
1 1001
11 001
10.10111100000000000000000000000000000000000000

Για να προκύψει ο κανονικοποιημένος συντελεστής του αποτελέσματος, ολισθαίνουμε κατά μία θέση δεξιά το αποτέλεσμα αυτό, απορρίπτουμε το περισσότερο σημαντικό ψηφίο και κβαντίζουμε τα υπόλοιπα σε αριθμό με 23 δυαδικά ψηφία (στρογγύλευση ή περικοπή) και παίρνουμε 01011110000000000000000.

Παρατηρούμε ότι ο συντελεστής του αποτελέσματος είναι
$p = 1.75 \times 1.5625 = 2.734375$

Επειδή είναι μεγαλύτερος του 2, κανονικοποιούμε το συντελεστή στη μορφή 1 + Σ, $p = 1{,}3671875 \times 2$, άρα ο εκθέτης πρέπει να αυξηθεί κατά ένα, και γίνεται 138 + 1 = 139.

Συνεπώς το αποτέλεσμα του πολλαπλασιασμού γράφεται στη μορφή $(-1)^0 \times (1 + 0.3671875) \times 2^{139-127} = 5600 = 5.6 \times 10^3$, η οποία σε δυαδική μορφή και κατά το πρότυπο IEEE 754, γράφεται ως: 0 10001011 01011110000000000000000.

Η πράξη β – γ
Αρχικά παρατηρούμε ότι έχουμε να αφαιρέσουμε έναν αρνητικό από έναν θετικό, άρα θα προσθέσουμε στο β το – γ το οποίο είναι θετικός. Συνεπώς, το αποτέλεσμα θα είναι θετικός και το πρόσημο του αποτελέσματος θα είναι 0.

Παρατηρούμε ότι ο γ έχει μεγαλύτερο εκθέτη (204) από τον β (135)
Άρα θα πρέπει να αποκανονικοποιηθεί ο β ως εξής:
$\beta = 1.5625 \times 2^8 = 1.5625 \times 2^{77-69} = 1.5625 \times 2^{-69} \times 2^{77}$

Παρατηρούμε ότι το 1.5625 θα πρέπει να ολισθήσει προς τη λιγότερο σημαντική θέση κατά 69 θέσεις, το οποίο και υπερβαίνει τις διαθέσιμες θέσεις άρα ο σχετικός συντελεστής μηδενίζεται και στην περίπτωση αυτή το αποτέλεσμα είναι το – γ .

Αν ήταν διαθέσιμα όλα τα απαραίτητα δυαδικά ψηφία για την αποκανονικοποίηση, θα προσθέταμε τους αριθμούς
$1.75 + 1.5625 \times 2^{-69} = 1.75000000000000000000002646977960 17$,
το οποίο άθροισμα αν στρογγυλοποιηθεί στα 23 δυαδικά ψηφία δίνει 1.75, άρα το αποτέλεσμα θα είναι πάλι – γ ή σε δυαδική μορφή
0 11001100 11000000000000000000000

Υπολογιστική Απόδοση

ΑΣΚΗΣΗ 16

Δίνονται δύο υπολογιστικά συστήματα PC1 (333 MHz) και PC2 (267 MHz), που εκτελούν ένα μίγμα τριών προγραμμάτων (εφαρμογών λογισμικού) Α, Β, και Γ. Να συγκριθούν τα δύο συστήματα, ως προς την ταχύτητα εκτέλεσης του μίγματος, στις εξής περιπτώσεις:

α) όταν η αναλογία Α:Β:Γ = 3:4:3
β) όταν η αναλογία Α:Β:Γ = 1:1:8
γ) όταν η αναλογία Α:Β:Γ = 8:1:1

και να συμπληρωθούν οι παρακάτω πίνακες για τα PC1 και PC2:

Σύστημα PC1 στα 333 MHz:

Πρόγραμμα	Εντολές	Κύκλοι ανά Εντολή	nsec ανά Κύκλο	Χρόνος Εκτέλεσης
Α	50.000	8,4	?	?
Β	75.000	13,8	?	?
Γ	100.000	14,5	?	?
Μίγμα (α)	?	?	?	?
Μίγμα (β)	?	?	?	?
Μίγμα (γ)	?	?	?	?

Σύστημα PC2 στα 267 MHz:

Πρόγραμμα	Εντολές	Κύκλοι ανά Εντολή	nsec ανά Κύκλο	Χρόνος Εκτέλεσης
Α	50.000	7,4	?	?
Β	75.000	12,8	?	?
Γ	100.000	10,5	?	?
Μίγμα (α)	?	?	?	?
Μίγμα (β)	?	?	?	?
Μίγμα (γ)	?	?	?	?

Διατυπώστε και ερμηνεύστε τις παρατηρήσεις σας αναφορικά με τις σχετικές ταχύτητες των δύο συστημάτων.

Λύση:

Παρατηρούμε ότι:

α) Η διάρκεια του βασικού «κύκλου» λειτουργίας ισούται με το αντίστροφο της συχνότητας λειτουργίας του κάθε συστήματος.
β) Χρόνος εκτέλεσης = (αριθμός εντολών) * (κύκλοι ανά εντολή) * (διάρκεια κύκλου).

Άρα:
Ο συμπληρωμένος πίνακας για το σύστημα PC1 στα 333 MHz είναι:

Ο

Πρόγραμμα	Εντολές	Κύκλοι ανά Εντολή	nsec ανά Κύκλο	Χρόνος Εκτέλεσης (sec)
Α	50.000	8,4	3,003	0,001261
Β	75.000	13,8	3,003	0,003108
Γ	100.000	14,5	3,003	0,004354
Μίγμα (α)	750.000	13,000	3,003	0,029279
Μίγμα (β)	925.000	14,114	3,003	0,039204
Μίγμα (γ)	575.000	10,165	3,003	0,017553

συμπληρωμένος πίνακας για το σύστημα PC2 στα 267 MHz είναι:

Πρόγραμμα	Εντολές	Κύκλοι ανά Εντολή	nsec ανά Κύκλο	Χρόνος Εκτέλεσης (sec)
Α	50.000	7,4	3,745	0,001386
Β	75.000	12,8	3,745	0,003596
Γ	100.000	10,5	3,745	0,003933
Μίγμα (α)	750.000	10,800	3,745	0,030337
Μίγμα (β)	925.000	10,519	3,745	0,036442
Μίγμα (γ)	575.000	8,643	3,745	0,018614

Παρατηρούμε ότι, αν και το PC1 έχει μεγαλύτερη συχνότητα λειτουργίας, είναι πιο αργό από το PC2 στην εκτέλεση του μίγματος (β). Αυτό φανερώνει ότι η συχνότητα λειτουργίας δεν αρκεί από μόνη της για να συγκριθεί η απόδοση δύο συστημάτων. Ο σχεδιασμός του ρεπερτορίου εντολών (instruction set) ενός συστήματος μπορεί να ευνοεί την εκτέλεση συγκεκριμένων τύπων προγραμμάτων, για τα οποία έχει βελτιστοποιηθεί.

ΑΣΚΗΣΗ 17

Θεωρήστε ότι πρόκειται να βελτιώσουμε έναν υπολογιστή και ότι υπάρχουν δύο πιθανοί τρόποι: είτε να κάνουμε τις εντολές πολλαπλασιασμού να τρέχουν τέσσερις φορές γρηγορότερα από πριν είτε να κάνουμε τις εντολές προσπέλασης της μνήμης να τρέχουν δύο φορές γρηγορότερα. Τρέχουμε ένα πρόγραμμα που χρειάζεται εκατό δευτερόλεπτα για να εκτελεστεί. Από αυτό το χρόνο 20% χρησιμοποιείται για πολλαπλασιασμούς, 50% για την εκτέλεση εντολών που προσπελαύνουν την μνήμη και 30% για τις υπόλοιπες. Ποια θα είναι η επιτάχυνση εάν βελτιώσουμε
α. μόνο το πολλαπλασιασμό;
β. μόνο την προσπέλασή της μνήμης και
γ. εάν κάνουμε και τις δύο βελτιώσεις;

Λύση:
Έστω ότι T_w είναι ο χρόνος εκτέλεσης των εντολών τύπου w. Τότε ο συνολικός χρόνος εκτέλεσης του προγράμματος θα είναι:

$T = T_{\text{πολλαπλασιασμών}} + T_{\text{προσπέλασης μνήμης}} + T_{\text{άλλων εντολών}} = 100$ δευτερόλεπτα
Με:
$T_{\text{πολλαπλασιασμών}} = 20$
$T_{\text{προσπέλασης μνήμης}} = 50$
$T_{\text{άλλων εντολών}} = 30$

Η επιτάχυνση θα ισούται με τον λόγο της απόδοσης, $A_{\text{νέα}}$, μετά την βελτίωση προς την αρχική απόδοση $A_{\text{αρχική}}$. Ως γνωστόν η απόδοση είναι ανάλογη του αντίστροφου του χρόνου εκτέλεσης.
α.
$A_{\text{νέα}} / A_{\text{αρχική}} = (1/ T_{\text{νέα}}) / (1/ T_{\text{αρχική}}) = T_{\text{αρχική}} / T_{\text{νέα}} =$
$=$ (100 δευτερόλεπτα) / (20/4 +50+30 δευτερόλεπτα) = 100 / (5+50+30) = 1.17
β.

$A_{νέα} / A_{αρχική} = (1/ T_{νέα}) / (1/ T_{αρχική}) = T_{αρχική} / T_{νέα} =$
$= (100$ δευτερόλεπτα$) / (20 +50/2+30$ δευτερόλεπτα$) = 100 / (20+25+30) = 1.33$
γ.
$A_{νέα} / A_{αρχική} = (1/ T_{νέα}) / (1/ T_{αρχική}) = T_{αρχική} / T_{νέα} =$
$= (100$ δευτερόλεπτα$) / (20/4 +50/2+30$ δευτερόλεπτα$) = 100 / (5+25+30) = 1.67$

ΑΣΚΗΣΗ 18

Έστω ότι σχεδιάζετε μια νέα και βελτιωμένη έκδοση μιας αρχιτεκτονικής και έχουν προταθεί τρείς διαφορετικές βελτιστοποιήσεις που θα βελτιώσουν την απόδοση (χρόνο εκτέλεσης) κατά τους παρακάτω συντελεστές :

Βελτίωση1 = 30
Βελτίωση2 = 20
Βελτίωση3 = 15

Κάθε μια από τις βελτιστοποιήσεις είναι ανεξάρτητη των άλλων αλλά κάθε χρονική στιγμή μόνο μία από τις τρεις μπορεί να είναι εν ενεργεία.

Σημείωση: Βελτίωση 20 σημαίνει ότι Χρόνος Εκτέλεσης Χωρίς Βελτίωση / Χρόνος Εκτέλεσης Με Βελτίωση = 20.

α) Αν οι βελτιστοποιήσεις 1 και 2 μπορούν να χρησιμοποιηθούν η κάθε μία για 25% του αρχικού χρόνου εκτέλεσης για τι ποσοστό του αρχικού χρόνου εκτέλεσης πρέπει να χρησιμοποιηθεί η βελτιστοποίηση 3 ώστε τελικά να έχουμε βελτίωση της συνολικής απόδοσης κατά 10 (Βελτίωση = 10). Σε τι ποσοστό του χρόνου δε χρησιμοποιείται καμμία βελτίωση ? Εξηγήστε τους υπολογισμούς σας.

β) Έστω ότι για κάποιο σημαντικό πρόγραμμα είναι δυνατό να χρησιμοποιήσουμε :

- Τη Βελτίωση1 για 15% του αρχικού χρόνου εκτέλεσης,
- Τη Βελτίωση2 για 15% του αρχικού χρόνου εκτέλεσης,
- Τη Βελτίωση3 για 70% του αρχικού χρόνου εκτέλεσης.

Θέλουμε να βελτιστοποιήσουμε την απόδοση του συστήματος. Αν μπορούμε τελικά να χρησιμοποιήσουμε μόνο μία από τις τρεις βελτιστοποιήσεις ποια θα πρέπει να διαλέξουμε;

Λύση :

α) Έστω ν το ποσοστό του αρχικού χρόνου κατά το οποίο πρέπει να εφαρμοστεί η Βελτίωση3. Ο νέος χρόνος εκτέλεσης τότε μπορεί να εκφραστεί ως συνάρτηση του αρχικού χρόνου εκτέλεσης (χωρίς καμία βελτίωση) ως εξής:

*Χρόνος Εκτέλεσης Με Βελτιώσεις = Χρόνος Εκτέλεσης Χωρίς Βελτιώσεις **
*(0,25 * 1/30 + 0,25 * 1/20 + ν * 1/15 + (1 – 0,25 - 0,25 - ν) * 1)*

Και αυτό γιατί :

1. Για 25% του αρχικού χρόνο θα εφαρμοστεί η Βελτίωση1 που μειώνει το χρόνο αυτό 30 φορές,
2. Για ένα άλλο 25% του αρχικού χρόνου θα εφαρμοστεί η Βελτίωση2 που μειώνει το χρόνο αυτό 20 φορές,
3. Για ένα άλλο ν% του αρχικού χρόνου θα εφαρμοστεί η Βελτίωση3 που μειώνει το χρόνο αυτό 15 φορές και
4. Για το υπόλοιπο του αρχικού χρόνου, δηλαδή για ένα ποσοστό (100%-25%-25%-ν%) δεν θα εφαρμόζεται καμία βελτίωση οπότε ο χρόνος αυτός θα παραμείνει ο ίδιος.

Για να πετύχουμε συνολική βελτίωση κατά 10 φορές θα πρέπει:

Χρόνος Εκτέλεσης_Χωρίς_Βελτίωσεις / Χρόνος Εκτέλεσης_Με_Βελτίωσεις = 10

Οπότε προκύπτει:

Χρόνος Εκτέλεσης_Χωρίς_Βελτίωσεις / (Χρόνος Εκτέλεσης Χωρίς Βελτιώσεις x
*(0,25 * 1/30 + 0,25 * 1/20 + ν * 1/15 + (1 – 0,25 - 0,25 - ν) * 1)) = 10 ⇔*
*1/(0,25 * 1/30 + 0,25 * 1/20 + ν * 1/15 + (1 – 0,25 - 0,25 - ν) * 1) = 10 ⇔*
*(0,25 * 1/30 + 0,25 * 1/20 + ν * 1/15 + (1 – 0,25 - 0,25 - ν) * 1) = 1/10 ⇔*
ν = (1/10 – 0,25/30 – 0,25/20 – 0,5) / (1/15 – 1) ⇔
ν ~ 0,4508 ⇔ ν = 45,08%

Το ποσοστό του χρόνου κατά το οποίο δε χρησιμοποιείται καμμία βελτίωση είναι :

1-0,25-0,25-ν = 0,5-0.4508 = 0.0492 ή 4,92%

β) Έστω Ε, E_1, E_2, E_3 αντίστοιχα ο αρχικός χρόνος εκτέλεσης και οι χρόνοι εκτέλεσης με τις βελτιστοποιήσεις 1, 2 και 3. Έχουμε πως:

E_1 = E * (0,15 * 1/30 + 0,85 * 1) = E * 0,8550, οπότε η βελτίωση είναι E / E_1 = 1/0,8550 ~ **1,1696**

E_2 = E * (0,15 * 1/20 + 0,85 * 1) = E * 0,8575, οπότε η βελτίωση είναι E / E_2 = 1/0,8575 ~ **1,1662**

E_3 = E * (0,70 * 1/15 + 0,30 * 1) ~ E * 0,3466, οπότε η βελτίωση είναι E / E_3 = 1/0,3466 ~ **2,8850**

Συνεπώς θα πρέπει να προτιμήσουμε τη χρήση της τρίτης βελτιστοποίησης.

ΑΣΚΗΣΗ 19

α) Ας υποθέσουμε ότι μία δεδομένη αρχιτεκτονική επεξεργαστή δεν περιλαμβάνει κυκλώματα που να υλοποιούν απ' ευθείας μία πράξη κινητής υποδιαστολής και ότι απαιτούνται 200 κύκλοι για να εκτελεσθεί η πράξη αυτή με εντολές γλώσσας μηχανής. Έστω ότι απαιτούνται 4 κύκλοι για την εκτέλεση της πράξης αυτής με κάποιο εξειδικευμένο κύκλωμα. Ποια θα είναι η συνολική βελτίωση την οποία θα επιφέρει στην εκτέλεση ενός προγράμματος του οποίου το 15% του χρόνου εκτέλεσης δαπανάται για την εκτέλεση της πράξης αυτής, η ενσωμάτωση στον επεξεργαστή του κυκλώματος που υλοποιεί την πράξη κινητής υποδιαστολής ?

β) Ένας μηχανικός σχεδιάζει το σύστημα μνήμης της επόμενης γενιάς ενός υπολογιστή. Εάν η υπάρχουσα έκδοση του υπολογιστή δαπανά το 40% του χρόνου για την εκτέλεση προσπελάσεων στην μνήμη πόσο πρέπει να βελτιωθεί το σύστημα της μνήμης για να έχουμε συνολική βελτίωση 1.2?

Σημείωση: Βελτίωση ν σημαίνει ότι : Χρόνος Εκτέλεσης $_{\text{πριν την Βελτίωση}}$ / Χρόνος Εκτέλεσης $_{\text{Μετά την Βελτίωση}}$ = ν.

Λύση

α) Έστω ότι ο χρόνος εκτέλεσης του προγράμματος στην περίπτωση που η πράξη κινητής υποδιαστολής εκτελείται με πρόγραμμα είναι Τ. Τότε το 15% του χρόνου, δηλαδή, 0,15*Τ, δαπανάται για την εκτέλεση των πράξεων κινητής υποδιαστολής. Όταν ο υπολογιστής μας διαθέτει μονάδα εκτέλεσης της πράξης κινητής υποδιαστολής τότε αυτή εκτελείται 200/4 = 50 φορές γρηγορότερα. Επομένως ο χρόνος εκτέλεσης των πράξεων κινητής υποδιαστολής θα είναι ίσος με 0,15Τ/50 και ο συνολικός χρόνος εκτέλεσης του προγράμματος θα είναι Τ' = 0,15Τ/50 + (1-0,15)Τ.
Η βελτίωση Β ισούται με

Β = Τ/Τ' = Τ/[0,15Τ/50 + (1-0,15)Τ] = 1/[0,15/50 + (1-0,15)] = 1/(0,003+1-0,15) = 1/0,853 = 1,17

β) Έστω ότι ο χρόνος εκτέλεσης του προγράμματος στην υπάρχουσα έκδοση του υπολογιστή είναι Τ. Τότε το 40% του χρόνου, δηλαδή, 0,4*Τ, δαπανάται για την προσπέλαση του συστήματος μνήμης, οπότε ο συνολικός χρόνος προσπέλασης μνήμης $T_{\pi\mu}$ = 0,4*Τ. Έστω ότι στο νέο σύστημα ο συνολικός χρόνος προσπέλασης της μνήμης είναι χ φορές μικρότερος από ότι στην υπάρχουσα έκδοση του συστήματος. Τότε ο συνολικός χρόνος προσπέλασης της μνήμης θα είναι $T'_{\pi\mu}$ = 0,4Τ/χ. Εφόσον μόνο το σύστημα της μνήμης ξανασχεδιάζεται ο υπόλοιπος χρόνος για την εκτέλεση του προγράμματος παραμένει όσος ήταν και στην υπάρχουσα έκδοση, δηλαδή Τ -0,4Τ, και ο συνολικός χρόνος εκτέλεσης του προγράμματος στον νέο υπολογιστή θα είναι Τ' = 0,4Τ/χ + (1-0,4)Τ. Η βελτίωση Β δίνεται από την σχέση: Β = Τ/Τ' = Τ/[0,4Τ/χ + (1-0,4)Τ]. Λαμβάνοντας υπόψη ότι η επιθυμητή βελτίωση είναι 1,2 έχουμε:

1,2 = Τ/[0,4Τ/χ + (1-0,4)Τ] ή ισοδύναμα 1,2 = 1/[0,4/χ + (1-0,4)] ή
0,4/χ+0,6 =1/1,2 ή 0,4/χ = 1/1,2-0,6 ή 0,4/χ = 0,2333 ή χ = 1,71

Κεντρική Μονάδα Ελέγχου

ΑΣΚΗΣΗ 20

α) Να σχεδιάσετε μονάδα τριών επιπέδων λογικής (NOT-AND-OR) για την εκτέλεση λογικών πράξεων AND, NAND, NOR και OR μεταξύ λέξεων των τεσσάρων δυαδικών ψηφίων.
β) Επιλέγοντας κατάλληλα τις εισόδους ελέγχου να δώσετε το κύκλωμα που υλοποι-είται με τις λιγότερες πύλες.

Λύση:

α) Η λογική μονάδα θα αποτελείται από τέσσερις όμοιες λογικές μονάδες του ενός δυαδικού ψηφίου. Στη συνέχεια μελετούμε την κατασκευή της λογικής μονάδας του ενός ψηφίου.
Θεωρώντας την ακόλουθη ανάθεση λέξεων ελέγχου σε λογικές πράξεις:

E_1E_0	Λειτουργία
00	AND
01	OR
10	NAND
11	NOR

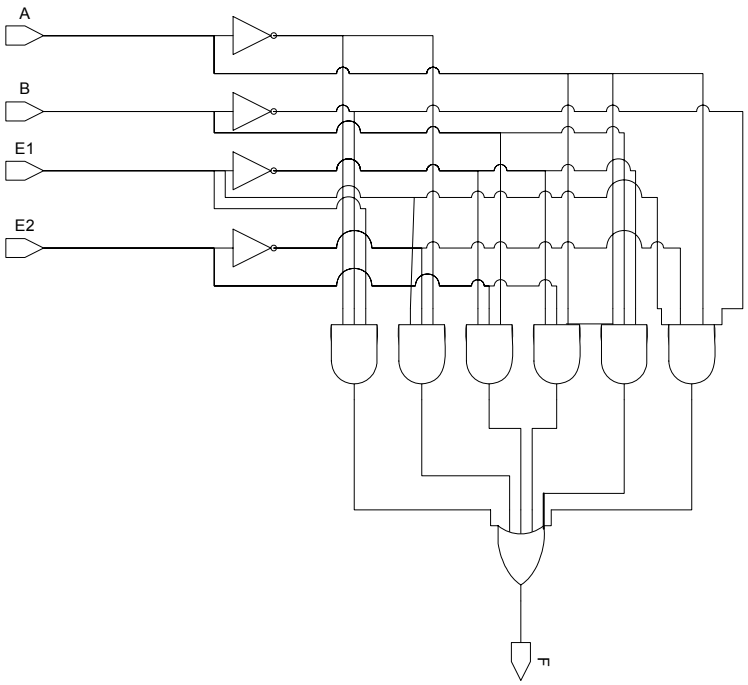
κατασκευάζουμε τον πίνακα αληθείας:

E_2E_1	A B	F	Λειτουργία
0 0	0 0	0	AND
0 0	0 1	0	
0 0	1 0	0	
0 0	1 1	1	
0 1	0 0	0	OR
0 1	0 1	1	
0 1	1 0	1	
0 1	1 1	1	
1 0	0 0	1	NAND
1 0	0 1	1	
1 0	1 0	1	
1 0	1 1	0	
1 1	0 0	1	NOR
1 1	0 1	0	
1 1	1 0	0	
1 1	1 1	0	

Με βάση τον πίνακα αληθείας, καταστρώνουμε τον ακόλουθο χάρτη Karnaugh:

E_1E_2\ AB	00	01	11	10
00			1	
01		1	1	1
11	1			
10	1	1		1

Άρα η προκύπτουσα λογική συνάρτηση είναι
$F = A'B'E1 + E1E2'A' + E1'E2B + E1'E2A + ABE1' + E1E2'AB'$
και το αντίστοιχο λογικό διάγραμμα είναι το ακόλουθο:

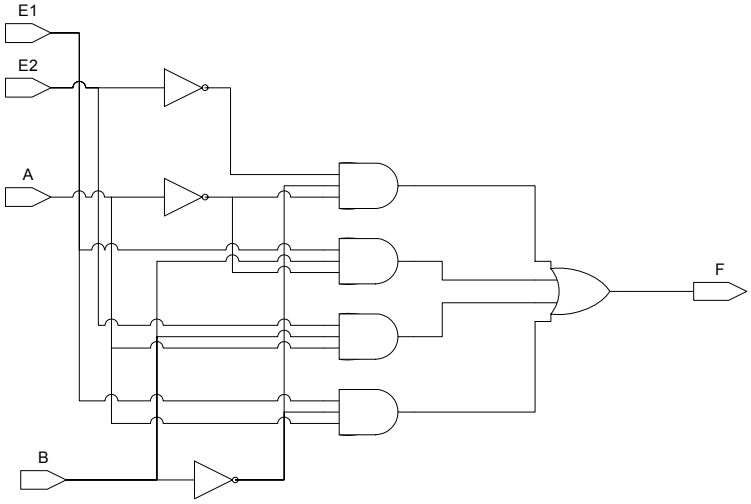


β) Παρατηρούμε ότι κάθε γραμμή του προηγούμενου χάρτη Karnaugh αντιστοιχεί σε μία λογική λειτουργία. Άρα οι διάφοροι δυνατοί τρόποι ανάθεσης λέξεων ελέγχου E_1E_0 σε λογικές λειτουργίες μπορούν να σχηματιστούν αναδιατάσσοντας τις γραμμές του Karnaugh. Μία ελάχιστη λογική συνάρτηση θα προκύψει για την ανάθεση όπου θα προκύψουν οι περισσότερες ομαδοποιήσεις πχ.

E_1E_2\ AB	00	01	11	10
00	1			
01			1	
11		1	1	1
10	1	1		1

Η ανάθεση αυτή δίνει τέσσερις όρους των τριών μεταβλητών
F=A' B' E2'+E1 A' B+E2 A B + E1 A B'

Το λογικό κύκλωμα είναι το ακόλουθο:



ΑΣΚΗΣΗ 21

α. Να αναφέρετε ποια είναι η διαφορά μεταξύ μικροπρογραμματισμένης και μικροπρογραμματιζόμενης μονάδας ελέγχου.
β. Να αναφέρετε πλεονεκτήματα και μειονεκτήματα της μικροπρογραμματισμένης μονάδας ελέγχου με οργάνωση δύο επιπέδων έναντι της μικροπρογραμματισμένης μονάδας ελέγχου με οργάνωση ενός επιπέδου.

Λύση:
α. Στη μικροπρογραμματιζόμενη μονάδα ελέγχου, ο χρήστης μπορεί να ορίσει το δικό του σύνολο εντολών σε επίπεδο γλώσσας μηχανής ή να προσθέσει νέες εντολές σε υπάρχον σύνολο εντολών, να γράψει τα μικροπρογράμματα τα οποία υλοποιούν τις νέες εντολές και να τα αποθηκεύσει στη μνήμη ελέγχου. Αντίθετα, στις μικροπρογραμματισμένες μονάδες ελέγχου, τα περιεχόμενα της μνήμης ελέγχου είναι προκαθορισμένα, από τον κατασκευαστή, και ο χρήστης δεν έχει τη δυνατότητα να αλλάξει τα περιεχόμενά της.

β. Η συνολική μνήμη η οποία απαιτείται όταν οργανώνουμε τη μονάδα ελέγχου σε δύο επίπεδα είναι σημαντικά μικρότερη από την απαιτούμενη μνήμη όταν οργανώνουμε τη μνήμη σε ένα επίπεδο.
Είναι δυνατόν η οργάνωση δύο επιπέδων να οδηγήσει σε μεγαλύτερη καθυστέρηση, εξαιτίας της προσπέλασης δύο μνημών αντί για μία.

ΑΣΚΗΣΗ 22

Θεωρείστε ότι το σύνολο των μικροεντολών που είναι αποθηκευμένο στη μνήμη ελέγχου μίας μικροπρογραμματισμένης μονάδας ελέγχου είναι:

C_0	C_1	C_2	C_3	C_4	C_5	C_6	C_7	C_8	C_9	C_{10}
1	1	0	0	1	0	0	0	0	0	0
0	1	0	0	0	0	1	0	0	0	0
0	0	1	0	0	1	0	0	0	0	0
0	0	0	0	1	1	0	0	0	1	0
0	0	0	0	1	0	0	0	1	0	0
1	0	1	1	0	0	0	0	0	0	0
0	0	0	1	0	0	0	1	0	0	0
0	0	0	0	0	0	0	0	0	1	1

(Προφανώς το σύνολο των μικροπρογραμμάτων μίας μικροπρογραμματισμένης μονάδας ελέγχου δεν αποτελείται ποτέ από τόσο μικρό αριθμό μικροεντολών ούτε και έχει τόσο μικρό αριθμό σημάτων ελέγχου. Στην άσκηση επιλέξαμε τόσο μικρό πλήθος αριθμών για να μπορούμε εύκολα να τα χειριστούμε.)
Θεωρείστε ότι κάθε σήμα ελέγχου ενεργοποιεί μία μικρολειτουργία. Να εφαρμόσετε την τεχνική της κωδικοποίησης των σημάτων ελέγχου για να ελαττωθεί η χωρητικότητα της απαιτούμενης μνήμης ελέγχου χωρίς να μειωθεί το πλήθος των μικρολειτουργιών που μπορούν να εκτελεστούν κάθε χρονική περίοδο παράλληλα.

α. Να υπολογίσετε την απαιτούμενη χωρητικότητα της μνήμης ελέγχου που προκύπτει με την εφαρμογή της κωδικοποίησης των σημάτων ελέγχου και το ποσοστό μείωσης της απαιτούμενης χωρητικότητας.
β. Να δώσετε τα νέα περιεχόμενα της μνήμης ελέγχου.

Λύση:
Αφού έχουμε θεωρήσει ότι κάθε σήμα ελέγχου ενεργοποιεί μία μικρολειτουργία, σε κάθε χρονική περίοδο θα εκτελούνται τόσες μικρολειτουργίες όσες μονάδες υπάρχουν στην μικροεντολή που

εκτελείται αυτή την περίοδο. Σύμφωνα με την τεχνική της κωδικοποίησης των σημάτων ελέγχου για να μην ελαττωθεί το πλήθος των μικρολειτουργιών που εκτελούνται σε κάθε χρονική περίοδο θα πρέπει να μην κωδικοποιηθούν μαζί σήματα ελέγχου που είναι ενεργά (έχουν την τιμή 1) στην ίδια μικροεντολή.

Μία από τις επιτρεπτές διαμερίσεις σε ομάδες είναι η ακόλουθη:
$\{C_0, C_5, C_{10}\}$, $\{C_1, C_2, C_7, C_8, C_9\}$, $\{C_3, C_4, C_6\}$

α. Επομένως η χωρητικότητα της μνήμης ελέγχου θα είναι 8 μικροεντολές X (2+3+2)δυαδικά ψηφία ανά μικροεντολή = 56 δυαδικά ψηφία. Η μείωση είναι 36%.

β.Τότε για την ομάδα σημάτων $\{C_0, C_5, C_{10}\}$ απαιτείται η αποθήκευση δύο τιμών (M_0 και M_1) σε κάθε μικροεντολή, για την ομάδα $\{C_1, C_2, C_7, C_8, C_9\}$ η αποθήκευση τριών τιμών(M_2, M_3 και M_4) σε κάθε μικροεντολή και για την ομάδα $\{C_3, C_4, C_6\}$ απαιτείται η αποθήκευση δύο τιμών (M_5 και M_6) σε κάθε μικροεντολή.

Έστω:

M_0 M_1
0 → δεν ενεργοποιείται κάποιο από τα σήματα C_0, C_5 ή C_{10}
0 → ενεργοποίηση του σήματος C_0
1 → ενεργοποίηση του σήματος C_5
1 → ενεργοποίηση του σήματος C_{10}

M_2 M_3 M_4
0 0 0 → δεν ενεργοποιείται κάποιο από τα σήματα C_1, C_2, C_7, C_8 ή C_9
1 0 0 → ενεργοποίηση του σήματος C_1
0 1 0 → ενεργοποίηση του σήματος C_2
1 1 0 → ενεργοποίηση του σήματος C_7
0 0 1 → ενεργοποίηση του σήματος C_8
1 0 1 → ενεργοποίηση του σήματος C_9
0 1 1 → δεν ενεργοποιείται κάποιο από τα σήματα C_1, C_2, C_7, C_8 ή C_9
1 1 1 → δεν ενεργοποιείται κάποιο από τα σήματα C_1, C_2, C_7, C_8 ή C_9

M_5 M_6
0 0 → δεν ενεργοποιείται κάποιο από τα σήματα C_3, C_4 ή C_6
1 0 → ενεργοποίηση του σήματος C_3
0 1 → ενεργοποίηση του σήματος C_4
1 1 → ενεργοποίηση του σήματος C_6

Τότε στη μνήμη ελέγχου θα είναι αποθηκευμένη η ακόλουθη πληροφορία:

M_0	M_1	M_2	M_3	M_4	M_5	M_6
1	0	1	0	0	0	1
0	0	1	0	0	1	1
0	1	0	1	0	0	0
0	1	1	0	1	0	1
0	0	0	0	1	0	1
1	0	0	1	0	1	0
0	0	1	1	0	1	0
1	1	1	0	1	0	0

ΑΣΚΗΣΗ 23

Θεωρήστε έναν μικροεπεξεργαστή με μονάδα ελέγχου που έχει σχεδιαστεί με την τεχνική του μικροπρογραμματισμού ενός επιπέδου. Το σύνολο των μικροπρογραμμάτων καταλαμβάνει 10.000 μικροεντολές της μνήμης ελέγχου και υπάρχουν 500 διαφορετικές μικροεντολές.

α) Ποιο είναι το πλεονέκτημα και ποιο το μειονέκτημα χρήσης και δεύτερου επιπέδου μικροπρογραμματισμού; Μπορείτε να χρησιμοποιήσετε αυτή την τεχνική στον παραπάνω μικροεπεξεργαστή; (Η απάντησή σας να μην ξεπερνάει τις 15 γραμμές).

β) Καθορίστε τον αριθμό των δυαδικών ψηφίων που πρέπει να έχει η μικροεντολή προκειμένου να επιτυγχάνουμε μείωση του απαιτούμενου αποθηκευτικού χώρου με τη χρήση του $2^{ου}$ επιπέδου κατά τουλάχιστον i) 50%, ii) 75% και iii) 90%.

Λύση:
Ερώτημα (α)
Εάν χρησιμοποιήσουμε και δεύτερο επίπεδο μικροπρογραμματισμού τότε μπορούμε να επιτύχουμε μείωση της χωρητικότητας που απαιτείται για τη μνήμη ελέγχου ενός επιπέδου. Για να μπορεί να χρησιμοποιηθεί αυτή η τεχνική θα πρέπει να εμφανίζονται στα μικροπρογράμματα συχνά όμοιες μικροεντολές. Τότε στη μνήμη ελέγχου δεύτερου επιπέδου (νανομνήμη) αποθηκεύονται όλες οι διαφορετικές μικροεντολές (νανοεντολές) ενώ στην μνήμη ελέγχου πρώτου επιπέδου αποθηκεύονται τα μικροπρογράμματα που τώρα αποτελούνται από διευθύνσεις της νανομνήμης.
Αυτό το χαρακτηριστικό ισχύει στην περίπτωση του παραπάνω μικροεπεξεργαστή αφού τα μικροπρογράμματα καταλαμβάνουν χώρο 10.000 μικροεντολών και απαρτίζονται μόνο από 500 διαφορετικές μικροεντολές. Άρα κατά μέσο όρο κάθε μικροεντολή επαναλαμβάνεται 10.000/500= 20 φορές, αριθμός αρκετά μεγάλος για να χρησιμοποιηθεί η τεχνική του δεύτερου επιπέδου.
Το μειονέκτημα της χρήσης και δεύτερου επιπέδου είναι ότι απαιτούνται δύο προσπελάσεις, μία στη μνήμη του πρώτου επιπέδου και μία στην μνήμη του δεύτερου. Αντίθετα με τη χρήση ενός επιπέδου απαιτείται μόνο μία προσπέλαση οπότε είναι ταχύτερη η εκτέλεση κάθε μικροεντολής.

Ερώτημα (β)
Έστω ότι το μήκος της κάθε μικροεντολής είναι Μ. Τότε για την υλοποίηση της μνήμης ελέγχου ενός επιπέδου απαιτούνται 10.000xΜ δυαδικά ψηφία.

Όταν η μνήμη ελέγχου υλοποιείται με την τεχνική του νανοπρογραμματισμού, τότε το δεύτερο επίπεδο αποτελείται από 500xΜ δυαδικά ψηφία, όσα ακριβώς απαιτούνται για την αποθήκευση των 500 διαφορετικών μικροεντολών. Το πρώτο επίπεδο έχει 10.000 θέσεις, όσες απαιτούνται για τα μικροπρόγραμματα στην περίπτωση του ενός επιπέδου, μόνο που τώρα το μήκος κάθε θέσης είναι διαφορετικό αφού είναι δείκτης στην νανομνήμη. Αφού λοιπόν η νανομνήμη έχει 500 θέσεις, άρα κάθε δείκτης θα έχει μήκος $\lceil \log_2(500) \rceil = 9$ δυαδικά ψηφία. Επομένως στην περίπτωση του νανοπρογραμματισμού απαιτούνται 10.000x9 + 500xΜ δυαδικά ψηφία.

Ο λόγος μείωσης του απαιτούμενου αποθηκευτικού χώρου για τις δύο τεχνικές είναι: $\Lambda = \frac{10.000M - 90.000 - 500M}{10.000M}$.
Άρα έχουμε:
i) Πρέπει $\Lambda \geq 50\%$ ή ισοδύναμα
$$\frac{10.000M - 90.000 - 500M}{10.000M} \geq 0.5$$
ή ισοδύναμα
10.000Μ - 90.000 - 500Μ ≥ 5000Μ ή ισοδύναμα
4.500Μ ≥ 90.000 ή ισοδύναμα
Μ ≥ 20.
ii) Ομοίως βρίσκω ότι για $\Lambda \geq 75\%$, πρέπει Μ ≥ 45.
iii) Ομοίως βρίσκω ότι για $\Lambda \geq 90\%$, πρέπει Μ ≥ 180.

ΑΣΚΗΣΗ 24

Σε ένα μικροπρογραμματιζόμενο υπολογιστικό σύστημα σκοπεύουμε να τρέξουμε το ακόλουθο πρόγραμμα:

Εντολή Α
Εντολή Β
Εντολή Γ
Εντολή Α
Εντολή Δ
Εντολή Ε

Τα μικροπρογράμματα που απαιτούνται για την εκτέλεση αυτών των εντολών συνοψίζονται στον παρακάτω πίνακα :

Εντολή Α	Μικροεντολή 1 Μικροεντολή 2 Μικροεντολή 3 Μικροεντολή 4 Μικροεντολή 2 Μικροεντολή 5	Εντολή Β	Μικροεντολή 1 Μικροεντολή 2 Μικροεντολή 6 Μικροεντολή 7 Μικροεντολή 8	Εντολή Γ	Μικροεντολή 1 Μικροεντολή 2 Μικροεντολή 9 Μικροεντολή 10
Εντολή Δ	Μικροεντολή 4 Μικροεντολή 7 Μικροεντολή 3 Μικροεντολή 11 Μικροεντολή 12 Μικροεντολή 13	Εντολή Ε	Μικροεντολή 1 Μικροεντολή 2 Μικροεντολή 9 Μικροεντολή 14		

Θεωρείστε ότι :

- Κάθε μικροεντολή έχει μήκος 64 δυαδικών ψηφίων.
- Ο χρόνος εκτέλεσης μιας μικροεντολής ταυτίζεται με το χρόνο προσπέλασής της.
- Ο χρόνος προσπέλασης μιας μικροεντολής, εξαρτάται μόνο από το συνολικό μέγεθος της μνήμης της μονάδας ελέγχου, δηλαδή, το μέγεθος της μνήμης ελέγχου προκειμένου για μικροπρογραμματιζόμενη μονάδα ελέγχου ή το άθροισμα των μεγεθών της μνήμης ελέγχου και της νανομνήμης στην περίπτωση νανοπρογραμματιζόμενης μονάδας ελέγχου. Για συνολικό μέγεθος μνήμης Μ ο χρόνος προσπέλασης μιας μικροεντολής ισούται με (M/1024) $*10^{-8}$sec.

Εξετάστε αν για την εκτέλεση του προγράμματος είναι προτιμότερη μια υλοποίηση της μονάδας ελέγχου με ένα ή δύο επίπεδα (απλά μικροπρογραμματιζόμενη ή νανοπρογραμματιζόμενη δηλαδή).

Λύση :

Στη μικροπρογραμματιζόμενη μονάδα ελέγχου ενός επιπέδου, θα αποθηκεύσουμε όλα τα μικροπρογράμματα για τις διαφορετικές εντολές που χρησιμοποιούμε στο πρόγραμμά μας. Αρα θα αποθηκεύσουμε τα 5 διαφορετικά μικροπρογράμματα ή ισοδύναμα τις 25 μικροεντολές του πίνακα. Το ότι η εντολή Α εκτελείται δύο φορές δε συνεπάγεται ότι χρειάζεται να αποθηκευτεί 2 φορές το μικροπρόγραμμα που της αντιστοιχεί. Το μέγεθος της μνήμης ελέγχου σε αυτή τη προσέγγιση θα είναι 25 x 64 = 1600 δυαδικά ψηφία. Αρα ο χρόνος ανάκλησης μιας μικροεντολής είναι (1600/1024) 10^{-8} sec = 15,625ns. Η εκτέλεση του προγράμματος απαιτεί την ανάκληση 6 μικροεντολών για την εντολή Α, 5 για τη Β, 4 για τη Γ, 6 για την Α, 6 για τη Δ και 4 για την Ε, δηλαδή συνολικά 31 μικροεντολών. Ο συνολικός χρόνος εκτέλεσης του προγράμματος συνεπώς θα είναι 31 x 15,625 = 484,375 ns.

Για μονάδα ελέγχου δύο επιπέδων, στη νανομνήμη θα αποθηκεύσουμε μόνο τις διαφορετικές μικροεντολές. Σύμφωνα με το πίνακα που μας έχει δοθεί, αυτές είναι 14 και συνεπώς το μέγεθος της νανομνήμης θα είναι S_1 = 14 x 64 = 896 bit. Τα περιεχόμενα της μνήμης ελέγχου είναι διευθύνσεις της

νανομνήμης. Για τη προσπέλαση των 14 διαφορετικών θέσεων της νανομνήμης χρειαζόμαστε να αποθηκεύουμε στη μνήμη ελέγχου πληροφορία μήκους $\lceil \log_2 14 \rceil = 4$ δυαδικών ψηφίων σε κάθε θέση της μνήμης ελέγχου. Προφανώς αυτό πρέπει να γίνει για τις 25 μικροεντολές του πίνακα. Αρα η μνήμη ελέγχου έχει μέγεθος $S_2 = 25 \times 4 = 100$ δυαδικά ψηφία. Αρα το συνολικό μέγεθος μνήμης για τη νανοπρογραμματιζόμενη μονάδα ελέγχου είναι $S = S_1+S_2 = 996$ bit. Συνεπώς μια μικροεντολή ανακαλείται (ισοδύναμα εκτελείται) σε χρόνο $(S/1024) * 10^{-8}$ sec = 9,7266 ns. (Ο χρόνος αυτός περιλαμβάνει και τις δύο προσπελάσεις των μνημών αφού στο S έχουμε λάβει υπ' όψιν μας το άθροισμα των μεγεθών των δύο μνημών). Ο χρόνος εκτέλεσης του προγράμματος ταυτίζεται σύμφωνα με την εκφώνηση της άσκησης με την ανάκληση 31 μικροεντολών και συνεπώς θα είναι 31 x 9,7266 = 301,5234 ns.

Συμπεραίνουμε λοιπόν ότι η οργάνωση δύο επιπέδων στο συγκεκριμένο παράδειγμα όχι μόνο μειώνει την απαιτούμενη μνήμη (996 bits έναντι 1600) για την υλοποίηση της μονάδας ελέγχου, αλλά οδηγεί και σε σημαντικά καλύτερο χρόνο εκτέλεσης για το δεδομένο πρόγραμμα (37,7% ταχύτερη εκτέλεση).

ΑΣΚΗΣΗ 25

1. Αναφέρετε εν συντομία τους τρόπους υλοποίησης μιας μονάδας ελέγχου και τα πλεονεκτήματα – μειονεκτήματα καθενός.
2. Εξηγείστε εν συντομία τι είναι η μνήμη ελέγχου και τι η νανομνήμη.
3. Εστω ότι το μήκος κάθε μικροεντολής σε ένα υπολογιστικό σύστημα είναι 32 δυαδικά ψηφία. Το σύνολο εντολών του υπολογιστικού συστήματος αποτελείται από 25 διαφορετικές εντολές και κάθε μία προσφέρεται με 2 εναλλακτικούς τρόπους διευθυνσιοδότησης. Όταν χρησιμοποιείται ο πρώτος τρόπος διευθυνσιοδότησης η διερμήνευση της εντολής απαιτεί 5 μικροεντολές ενώ όταν χρησιμοποιείται ο δεύτερος 7 μικροεντολές. Θεωρείστε ότι συνολικά υπάρχουν 109 διαφορετικές μικροεντολές.
   - Υπολογίστε σε KBytes το μέγεθος της απαιτούμενης μνήμης ελέγχου, για οργάνωση ενός επιπέδου.
   - Δώστε σε KBytes τη συνολική μνήμη που θα απαιτηθεί όταν χρησιμοποιηθεί οργάνωση δύο επιπέδων για τη μνήμη ελέγχου.

Λύση

1. Η υλοποίηση της μονάδας ελέγχου μπορεί να γίνει είτε ακολουθώντας τη γνωστή διαδικασία σχεδίασης ενός ακολουθιακού κυκλώματος είτε με τη τεχνική του μικροπρογραμματισμού. *Στη δεύτερη περίπτωση η ακολουθία σημάτων που πρέπει να παραχθεί αποθηκεύεται σαν μια ακολουθία μικροεντολών, σε μία μνήμη που ονομάζεται μνήμη ελέγχου.*

   Τα πλεονεκτήματα της σχεδίασης ως ακολουθιακού κυκλώματος είναι η μεγάλη ταχύτητα και το μικρό μέγεθος. Μειονεκτήματα είναι η αυξημένη πολυπλοκότητα, το κόστος ανάπτυξης και η αδυναμία τροποποιήσεων.

   Ο μικροπρογραμματισμός προσφέρει ευελιξία και συστηματικό τρόπο σχεδιασμού. Ωστόσο μια μικροπρογραμματισμένη μονάδα ελέγχου είναι πιο αργή και πιο μεγάλη ως κύκλωμα από αυτήν που προκύπτει από το $1^ο$ τρόπο σχεδιασμού.
2. Η μνήμη ελέγχου ορίστηκε στο προηγούμενο ερώτημα. Για τη μείωση του μεγέθους της μνήμης ελέγχου, μπορούμε να την υλοποιήσουμε ως μνήμη 2 επιπέδων, όπου το δεύτερο επίπεδο ονομάζεται νανομνήμη. Στη νανομνήμη αποθηκεύουμε *μόνο* τις διαφορετικές μικροεντολές. Στο $1^ο$ επίπεδο αποθηκεύουμε τα μικροπρογράμματα με τη μορφή διευθύνσεων της νανομνήμης.
3.
   - Κάθε τρόπος διευθυνσιοδότησης μας οδηγεί σε διαφορετικό μικροπρόγραμμα. Συνεπώς υπάρχουν 2 x 25 = 50 διαφορετικά μικροπρογράμματα. Τα μισά έχουν 5 μικροεντολές ενώ τα υπόλοιπα 7, άρα συνολικά στη μνήμη ελέγχου πρέπει να αποθηκευτούν 25 x 5 + 25 x 7 = 300 μικροεντολές. Στην περίπτωση οργάνωσης ενός επιπέδου, η μνήμη ελέγχου θα έχει ελάχιστο μέγεθος 300 x 32 δυαδικά ψηφία = 300 x 4 Bytes = 1200 Bytes = 1200 / 1024 KBytes ≈ 1,17 KBytes.

- Όπως εξηγήθηκε παραπάνω, στη νανομνήμη αποθηκεύουμε μόνο τις διαφορετικές μικροεντολές. Συνεπώς αυτή θα έχει μέγεθος 109 x 32 δυαδικά ψηφία = 109 x 4 Bytes = 436 Bytes = 436 / 1024 KBytes ≈ 0,425 KBytes. Στο πρώτο επίπεδο αποθηκεύουμε διευθύνσεις της νανομνήμης. Αφού αυτή έχει 109 θέσεις κάθε διεύθυνσή της θα έχει μέγεθος $\lceil \log_2 109 \rceil$ = 7 δυαδικά ψηφία. Αρα το μέγεθος της μνήμης ελέγχου είναι 300 x 7 = 2100 δυαδικά ψηφία = 262,5 Bytes = 0,256 KBytes. Το συνολικό μέγεθος μνήμης που χρειαζόμαστε είναι συνεπώς 0,425 + 0,256 ≈ 0,68 KBytes.

ΑΣΚΗΣΗ 26

Θέλουμε να αναπτύξουμε μια μικροπρογραμματιζόμενη μονάδα ελέγχου για τον επεξεργαστή 8085, ικανή να διερμηνεύει τις εντολές :

- **LXI H, $X_3X_2X_1X_0$**
  όπου το $X_3X_2X_1X_0$ είναι η δεκαεξαδική τιμή που ακολουθεί στις επόμενες θέσεις μνήμης. Θυμηθείτε ότι στην αμέσως επόμενη από το opcode ακολουθεί η λέξη X_1X_0 και στη μεθεπόμενη η λέξη X_3X_2.
- **MOV A, M**

Το υλικό του 8085 είναι ικανό να εκτελεί τις μικρολειτουργίες του παρακάτω πίνακα:

Μικρολειτουργία	Επεξήγηση	Σήμα Ελέγχου
PC ← PC+1;	Αύξηση του μετρητή προγράμματος PC	C_1
Address Bus ← PC; Read Memory;	Ανάγνωση από τη κύρια μνήμη από τη διεύθυνση που δείχνει ο PC	C_2
Address Bus ← HL; Read Memory;	Ανάγνωση από τη κύρια μνήμη από τη διεύθυνση που δείχνει το ζεύγος HL	C_3
L ← Data Bus;	Το δεδομένο που αναγνώστηκε από τη μνήμη αποθηκεύεται στον καταχωρητή L	C_4
H ← Data Bus;	Το δεδομένο που αναγνώστηκε από τη μνήμη αποθηκεύεται στον καταχωρητή H	C_5
A ← Data Bus;	Το δεδομένο που αναγνώστηκε από τη μνήμη αποθηκεύεται στο συσσωρευτή A	C_6

Η εκτέλεση κάθε μικρολειτουργίας ελέγχεται από το αντίστοιχο σήμα ελέγχου που αναφέρεται στον πίνακα. Για παράδειγμα όταν το C_1 είναι 1 ο μετρητής προγράμματος αυξάνεται κατά 1, ενώ όταν είναι 0 δεν εκτελείται αυτή η μικρολειτουργία. Υποθέστε ότι:

- δεν επιτρέπεται παράλληλη εκτέλεση μικρολειτουργιών,
- κάθε μικροεντολή σας είναι της μορφής $C_6\ C_5\ C_4\ C_3\ C_2\ C_1$,
- ο PC στην αρχή κάθε μικροπρογράμματος περιέχει τη διεύθυνση του κωδικού (opcode) κάθε εντολής.

- Ζητείται να δώσετε σε μορφή πίνακα τα μικροπρογράμματα για τις δύο παραπάνω εντολές, δηλαδή για την εντολή **LXI H, $X_3X_2X_1X_0$** και την εντολή **MOV A, M**.

Μικροεντολή	$C_6C_5C_4C_3C_2C_1$	Επεξήγηση
1	…	…
2	…	…
…	…	…
…	…	…
…	…	…
…	…	…

Λύση

LXI H, $X_3X_2X_1X_0$

Μικροεντολή	**$C_6C_5C_4C_3C_2C_1$**	**Επεξήγηση**
1	0 0 0 0 0 1	Αυξάνω το PC. Το περιεχόμενό του δείχνει στη λέξη X_1X_0
2	0 0 0 0 1 0	Διάβασμα του X_1X_0 από τη μνήμη
3	0 0 1 0 0 0	$L \leftarrow X_1X_0$
4	0 0 0 0 0 1	Αυξάνω το PC. Το περιεχόμενό του δείχνει στη λέξη X_3X_2
5	0 0 0 0 1 0	Διάβασμα του X_3X_2 από τη μνήμη.
6	0 1 0 0 0 0	$H \leftarrow X_3X_2$
7	0 0 0 0 0 1	Αυξάνω το PC. Το περιεχόμενό του δείχνει στον κωδικό της επόμενης εντολής

MOV A, M

Μικροεντολή	**$C_6C_5C_4C_3C_2C_1$**	**Επεξήγηση**
1	0 0 0 1 0 0	Θα προσπελάσουμε τη θέση μνήμης που μας δείχνει ο HL
2	1 0 0 0 0 0	Αποθήκευση του δεδομένου στο συσσωρευτή
3	0 0 0 0 0 1	Αυξάνω το PC. Το περιεχόμενό του δείχνει στον κωδικό της επόμενης εντολής

Αρχιτεκτονικές Επεξεργαστικών Μονάδων

ΑΣΚΗΣΗ 27

Ποιές είναι οι συναρτήσεις που υλοποιούν οι παρακάτω κώδικες γλώσσας μηχανής;

α) Αρχιτεκτονική μηχανισμού σωρού:

```
PUSH A
PUSH B
PUSH C
ADD
PUSH D
SUB
MUL
PUSH A
ADD
POP E
```

β) Αρχιτεκτονική συσσωρευτή:

```
LOAD B
ADD C
MUL A
STORE E
LOAD D
SUB E
STORE E
```

γ) Αρχιτεκτονική καταχωρητή-μνήμης:

```
LOAD R1, A
MUL  R1, D
SUB  R1, C
STORE R1, E
LOAD R2, B
ADD R2, A
ADD R2, E
STORE E, R2
```

δ) Αρχιτεκτονική καταχωρητή-καταχωρητή (Λάβετε υπόψη σας ότι SUB R4, R2, R3 σημαίνει R4 <- R2 - R3. Το ίδιο ισχύει και για τις άλλες πράξεις.):

```
LOAD R1, A
LOAD R2, B
LOAD R3, C
ADD R4, R2, R3
SUB R4, R4, R1
LOAD R5, D
MUL R5, R4, R5
ADD R5, R5, R1
STORE E, R5
```

Λύση:

α) Αρχιτεκτονική μηχανισμού σωρού:

ΕΝΤΟΛΗ	ΣΩΡΟΣ ΜΕΤΑ ΤΗΝ ΕΚΤΕΛΕΣΗ (ΚΟΡΥΦΗ ΣΤΑ ΑΡΙΣΤΕΡΑ)
PUSH A	A

PUSH B	B, A
PUSH C	C, B, A
ADD	(C + B), A
PUSH D	D, (C + B), A
SUB	(D – (C + B)), A
MUL	((D – (C + B)) * A)
PUSH A	A, ((D – (C + B)) * A)
ADD	(A + ((D – (C + B)) * A))
POP E	ΚΕΝΟ

E = A + (A * (D - (B + C)))

β) Αρχιτεκτονική συσσωρευτή:

ΕΝΤΟΛΗ	ΑΠΟΤΕΛΕΣΜΑ ΜΕΤΑ ΤΗΝ ΕΚΤΕΛΕΣΗ (Σ = ΣΥΣΣΩΡΕΥΤΗΣ)
LOAD B	Σ = B
ADD C	Σ = (B + C)
MUL A	Σ = A * (B + C)
STORE E	E = A * (B + C)
LOAD D	Σ = D
SUB E	Σ = D – E = D – (A * (B + C))
STORE E	E = D – (A * (B + C))

E = D – ((B + C) * A)

γ) Αρχιτεκτονική καταχωρητή-μνήμης:

ΕΝΤΟΛΗ	ΑΠΟΤΕΛΕΣΜΑ ΜΕΤΑ ΤΗΝ ΕΚΤΕΛΕΣΗ ΤΗΣ
LOAD R1, A	R1 = A
MUL R1, D	R1 = R1 * D = A * D
SUB R1, C	R1 = R1 – C = (A * D) – C
STORE R1, E	E = R1 = (A * D) – C
LOAD R2, B	R2 = B
ADD R2, A	R2 = R2 + A = B + A
ADD R2, E	R2 = R2 + E = B + A + ((A * D) – C)
STORE E, R2	E = B + A + ((A * D) – C)

E = B + A + ((A * D) – C)

δ) Αρχιτεκτονική καταχωρητή-καταχωρητή:

ΕΝΤΟΛΗ	ΑΠΟΤΕΛΕΣΜΑ ΜΕΤΑ ΤΗΝ ΕΚΤΕΛΕΣΗ ΤΗΣ
LOAD R1, A	R1 = A
LOAD R2, B	R2 = B
LOAD R3, C	R3 = C
ADD R4, R2, R3	R4 = R2 + R3 = B + C
SUB R4, R4, R1	R4 = R4 – R1 = (B + C) – A
LOAD R5, D	R5 = D
MUL R5, R4, R5	R5 = R4 * R5 = ((B + C) – A) * D
ADD R5, R5, R1	R5 = R5 + R1 = (((B + C) – A) *D) +A
STORE E, R5	E = R5 = (((B + C) – A) * D) + A

E = A + (D * ((B + C) – A))

ΑΣΚΗΣΗ 28

Να γραφεί πρόγραμμα σε επίπεδο γλώσσας μηχανής για τον υπολογισμό της παράστασης:
X = A + (B + C•D) •A - E•Z
σε υπολογιστή που βασίζεται:

α) στη χρήση μηχανισμού σωρού και
β) στη χρήση συσσωρευτή.

Λύση:
Ερώτημα (α)
Θα πρέπει να μετατρέψουμε κατ' αρχήν την παράσταση σε "postfix notation", δηλαδή για κάθε πράξη πρώτα να εμφανίζονται τα τελούμενα και μετά οι τελεστές. Στις παρακάτω πράξεις κάθε υπογραμμισμένη έκφραση είναι σε "postfix notation".

X = A + (B + C•D) •A - E•Z=
= A + (B + CD•)•A - E•Z=
= A + (BCD•+) •A - E•Z=
= A + BCD•+A• - E•Z=
= ABCD•+A•+ - E•Z=
= ABCD•+A•+ - EZ•=
= EZ•ABCD•+A•+- (Αλλαγή ορισμάτων λόγω της αφαίρεσης)

Τώρα διατρέχοντας τη σχέση από τα αριστερά προς τα δεξιά κάθε φορά που θα συναντάω τελούμενο θα γράφω την εντολή Push ενώ κάθε φορά που θα συναντάω τελεστή θα γράφω την εντολή του τελεστή.

Επομένως το πρόγραμμα είναι:

Push E	Τοποθετώ στον σωρό το E
Push Z	Τοποθετώ στον σωρό το Z
Multiply	Αντικαθιστώ την κορυφή του σωρού με το Z επί E
Push A	Τοποθετώ στον σωρό το A
Push B	Τοποθετώ στον σωρό το B
Push C	Τοποθετώ στον σωρό το C
Push D	Τοποθετώ στον σωρό το D
Multiply	Αντικαθιστώ την κορυφή του σωρού με το D επί C
Add	Αντικαθιστώ την κορυφή του σωρού με το D•C+B
Push A	Τοποθετώ στον σωρό το A
Multiply	Αντικαθιστώ την κορυφή του σωρού με το A•(D•C+B)
Add	Αντικαθιστώ την κορυφή του σωρού με A•(D•C+B)+A
Substract	Αντικαθιστώ την κορυφή του σωρού με A•(D•C+B)+A-Z•E
Pop X	Αποθηκεύω την κορυφή του σωρού στο X

Αν εκτελεστεί το παραπάνω πρόγραμμα τότε θα υπολογιστεί η έκφραση A•(D•C+B)+A-Z•E η οποία είναι ίδια με τη ζητούμενη εφόσον οι πράξεις πρόσθεση και πολλαπλασιασμός είναι αντιμεταθετικές.

Ερώτημα (β)
Θα ξεκινήσω από την υλοποίηση των υπο-εκφράσεων προκειμένου να πετύχω το μικρότερο δυνατό σε έκταση πρόγραμμα. Θα χρησιμοποιήσουμε και τη βοηθητική θέση μνήμης, Y.

Έτσι έχουμε το ακόλουθο πρόγραμμα:

Load E	Φορτώνουμε στο συσσωρευτή το E
Multiply Z	Πολλαπλασιάζουμε το συσσωρευτή με το Z

Store Y Αποθηκεύουμε το συσσωρευτή στο Y
Load C Φορτώνουμε στο συσσωρευτή το C
Multiply D Πολλαπλασιάζουμε το συσσωρευτή με το D
Add B Προσθέτουμε στο συσσωρευτή το B
Multiply A Πολλαπλασιάζουμε το συσσωρευτή με το A
Add A Προσθέτουμε στο συσσωρευτή το A
Substract Y Αφαιρούμε από το συσσωρευτή το Y
Store X Αποθηκεύουμε το συσσωρευτή στο X

ΑΣΚΗΣΗ 29

Ποιές είναι οι συναρτήσεις που υλοποιούν οι παρακάτω κώδικες γλώσσας μηχανής;

α) Αρχιτεκτονική μηχανισμού σωρού:

```
PUSH   A
PUSH   D
ADD
PUSH   B
PUSH   D
MUL
POP    E
PUSH   A
MUL
PUSH   C
ADD
PUSH   E
ADD
POP    E
```

β) Αρχιτεκτονική συσσωρευτή:

```
LOAD   A
MUL    C
ADD    B
STORE  E
LOAD   D
ADD    E
STORE  E
```

γ) Αρχιτεκτονική καταχωρητή-μνήμης:

```
LOAD   R1, A
ADD    R1, D
LOAD   R2, B
ADD    R2, C
STORE  R2, E
ADD    R1, E
MUL    R1, A
STORE  R1, E
```

δ) Αρχιτεκτονική καταχωρητή-καταχωρητή:

```
LOAD   R1, C
LOAD   R2, A
ADD    R2, R2, R1
LOAD   R1, B
ADD    R1, R2, R1
LOAD   R2, D
MUL    R2, R2, R1
ADD    R1, R2, R1
STORE  R1, E
```

Λύση :

α) Αρχιτεκτονική μηχανισμού σωρού:

Εντολή		**Σωρός μετά την εκτέλεση**
		(Κορυφή στα αριστερά)
PUSH	A	A
PUSH	D	D, A
ADD		D+A
PUSH	B	B, D+A
PUSH	D	D, B, D+A
MUL		D*B, D+A
POP	E	D+A
		Σημείωση : E=D*B
PUSH	A	A, D+A
MUL		A*(D+A)
PUSH	C	C, A*(D+A)
ADD		C+A*(D+A)
PUSH	E	D*B, C+A*(D+A)
ADD		D*B+C+A*(D+A)
POP	E	KENO

<u>E = D*B+C+A*(A+D)</u>

β) Αρχιτεκτονική συσσωρευτή:

Εντολή		**Αποτέλεσμα μετά την εκτέλεση**
		(Σ = συσσωρευτής)
LOAD	A	Σ = A
MUL	C	Σ = A*C
ADD	B	Σ = B+A*C
STORE	E	E = B+A*C
LOAD	D	Σ = D
ADD	E	Σ = D+B+A*C
STORE	E	E = D+B+A*C

<u>E = D+B+A*C</u>

γ) Αρχιτεκτονική καταχωρητή-μνήμης :

Εντολή		**Αποτέλεσμα μετά την εκτέλεση**
LOAD	R1, A	R1 = A
ADD	R1, D	R1 = A + D
LOAD	R2, B	R2 = B
ADD	R2, C	R2 = B + C
STORE	R2, E	E = B + C
ADD	R1, E	R1 = A + D + B + C
MUL	R1, A	R1 = A * (A + D + B + C)
STORE	R1, E	E = A * (A + D + B + C)

<u>E = A * (A + D + B + C)</u>

δ) Αρχιτεκτονική καταχωρητή-καταχωρητή :

Εντολή		**Αποτέλεσμα μετά την εκτέλεση**
LOAD	R1, C	R1 = C
LOAD	R2, A	R2 = A
ADD	R2, R2, R1	R2 = A+C

```
LOAD  R1, B        R1 = B
ADD   R1, R2, R1   R1 = A+C+B
LOAD  R2, D        R2 = D
MUL   R2, R2, R1   R2 = D*(A+C+B)
ADD   R1, R2, R1   R1 = D*(A+C+B)+(A+C+B)
STORE R1, E        E = D*(A+C+B)+(A+C+B)
```

E = D * (A + C + B) + (A + C + B)

ΑΣΚΗΣΗ 30

i. Ποιές είναι οι συναρτήσεις που υλοποιούν οι παρακάτω κώδικες γλώσσας μηχανής ;

Αρχιτεκτονική μηχανισμού σωρού:

```
PUSH  A
PUSH  B
MUL
PUSH  C
PUSH  D
DIV
SUB
POP   Z
PUSH  A
PUSH  C
ADD
PUSH  Z
ADD
POP   Z
```

Αρχιτεκτονική συσσωρευτή:

```
LOAD  A
SUB   B
MUL   C
STORE F
LOAD  D
DIV   F
STORE F
```

Αρχιτεκτονική καταχωρητή-καταχωρητή:

```
LOAD  R1, C
LOAD  R2, A
ADD   R2, R2, R1
LOAD  R1, B
ADD   R1, R2, R1
LOAD  R2, D
MUL   R2, R2, R1
ADD   R1, R2, R1
STORE R1, E
```

(Σημείωση : Στην αρχιτεκτονική μηχανισμού σωρού, οι μη αντιμεταθετικές πράξεις χρησιμοποιούν σαν πρώτο έντελο –διαιρετέο ή αφαιρετέο δηλαδή- το κορυφαίο στοιχείο του σωρού).

ii. Μια μηχανή υποστηρίζει εντολές καταχωρητή – μνήμης και καταχωρητή - καταχωρητή 2 εντέλων. Το αποτέλεσμα της εντολής αποθηκεύεται στον καταχωρητή για εντολές καταχωρητή – μνήμης και στον αριστερά αναγραφόμενο καταχωρητή για εντολές καταχωρητή – καταχωρητή, με εξαίρεση την εντολή STORE, που αντιγράφει τα περιεχόμενα του καταχωρητή στη μνήμη. Για τον υπολογισμό της ίδιας έκφρασης, δύο διαφορετικοί προγραμματιστές έγραψαν τα προγράμματα Α και Β παρακάτω.

[1] Ποιος είναι ο κοινός υπολογισμός ?

[2] Οι εντολές καταχωρητή – καταχωρητή εκτελούνται σε μικρότερο χρόνο από τις εντολές καταχωρητή μνήμης μιας και αποφεύγεται η χρονοβόρα προσπέλαση της μνήμης.. Με βάση τα παραπάνω συγκρίνετε τα προγράμματα Α και Β.

[3] Γράψτε ένα πρόγραμμα, Γ, που να κάνει τον ίδιο υπολογισμό στον μικρότερο δυνατό χρόνο λαμβάνοντας υπόψη ότι ο υπολογιστής στον οποίο θα εκτελεστεί το πρόγραμμά σας έχει μόνο τέσσερις καταχωρητές γενικού σκοπού.

Πρόγραμμα Α
LOAD R1, A
LOAD R2, B
ADD R1, R2
LOAD R2, C
LOAD R3, D
MUL R2, R3
SUB R1, R2
LOAD R2, A
LOAD R3, C
SUB R2, R3
DIV R1, R2
ADD R1, D
STORE R1, Z

Πρόγραμμα Β
LOAD R1, A
LOAD R2, B
LOAD R3, C
LOAD R4, D
LOAD R5, R1
LOAD R6, R3
ADD R5, R2
MUL R6, R4
SUB R1, R3
SUB R5, R6
DIV R5, R1
ADD R5, R4
STORE R5, Z

Λύση :

i.

Εκτελούμενη Εντολή		Κατάσταση Σωρού (Κορυφαίο στοιχείο αριστερά)
PUSH	A	A
PUSH	B	B, A
MUL		A*B
PUSH	C	C, A*B
PUSH	D	D, C, A*B
DIV		D/C, A*B
SUB		D/C – A*B
POP	Z	Garbage (σκουπίδι)[†].
		Στη θέση μνήμης Z αποθηκεύεται το D/C – A*B
PUSH	A	A
PUSH	C	C, A
ADD		C+A
PUSH	Z	D/C – A*B, C+A
ADD		D/C – A*B+ C+A
POP	Z	Σκουπίδι
		Στη θέση μνήμης Z αποθηκεύεται το D/C – A*B+ C+A

Αρα υλοποιείται η συνάρτηση **Z= D/C – A*B+ C+A**

Για την αρχιτεκτονική συσσωρευτή έστω ότι Σ συμβολίζει τον συσσωρευτή. Το αποτέλεσμα κάθε

[†] Garbage (σκουπίδι) = δεδομένο άσχετο με τον τρέχοντα υπολογισμό.

εντολής αναγράφεται στα δεξιά της.

LOAD	A	Σ=A
SUB	B	Σ=A-B
MUL	C	Σ=(A-B)*C
STORE	F	Σ=(A-B)*C, F=(A-B)*C
LOAD	D	Σ=D
DIV	F	Σ=D / ((A-B)*C)
STORE	F	Σ= D/((A-B)*C), F= D/((A-B)*C)

Αρα υλοποιείται η συνάρτηση **F= D/((A-B)*C)**

Για την αρχιτεκτονική καταχωρητή-καταχωρητή έχουμε :

LOAD	R1, C	R1=C
LOAD	R2, A	R2=A
ADD	R2, R2, R1	R2=R2+R1=A+C
LOAD	R1, B	R1=B
ADD	R1, R2, R1	R1=R2+R1=A+C+B
LOAD	R2, D	R2=D
MUL	R2, R2, R1	R2=R2*R1=D*(A+C+B)
ADD	R1, R2, R1	R1=R2+R1= D*(A+C+B)+ A+C+B=(D+1)* (A+C+B)
STORE	R1, E	E=(D+1)* (A+C+B)

Αρα υλοποιείται η συνάρτηση **E= (D+1)* (A+C+B)**

ii.

[1]

Πρόγραμμα Α

LOAD	R1, A	R1=A
LOAD	R2, B	R2=B
ADD	R1, R2	R1=A+B
LOAD	R2, C	R2=C
LOAD	R3, D	R3=D
MUL	R2, R3	R2=C*D
SUB	R1, R2	R1=A+B-C*D
LOAD	R2, A	R2=A
LOAD	R3, C	R3=C
SUB	R2, R3	R2=A-C
DIV	R1, R2	R1=(A+B-C*D)/(A-C)
ADD	R1, D	R1=(A+B-C*D)/(A-C)+D
STORE	R1, Z	**Z=(A+B-C*D)/(A-C)+D**

Πρόγραμμα Β

LOAD	R1, A	R1=A
LOAD	R2, B	R2=B
LOAD	R3, C	R3=C
LOAD	R4, D	R4=D
LOAD	R5, R1	R5=A
LOAD	R6, R3	R6=C
ADD	R5, R2	R5=A+B
MUL	R6, R4	R6=C*D
SUB	R1, R3	R1=A-C
SUB	R5, R6	R5=A+B-C*D
DIV	R5, R1	R5=(A+B-C*D)/(A-C)
ADD	R5, R4	R5=(A+B-C*D)/(A-C)+D
STORE	R5, Z	**Z=(A+B-C*D)/(A-C)+D**

Από τα αποτελέσματα των 2 προγραμμάτων, βλέπουμε ότι επιτελούν τον ίδιο υπολογισμό.

[2] Παρατηρούμε ότι το πρόγραμμα Α, χρησιμοποιεί μόνο 3 διαφορετικούς καταχωρητές και απαιτεί 8 εντολές καταχωρητή-μνήμης. Το πρόγραμμα Β, από την άλλη πλευρά, χρησιμοποιεί 6

διαφορετικούς καταχωρητές και απαιτεί μόνο 5 εντολές καταχωρητή-μνήμης. Συνεπώς το πρόγραμμα Β εκτελείται σε μικρότερο χρονικό διάστημα διότι χρησιμοποιεί περισσότερες εντολές καταχωρητή-καταχωρητή που είναι πιο γρήγορες.

[3]

Πρόγραμμα Γ

LOAD	R1, A	R1=A
LOAD	R2, R1	R2=A
ADD	R2, B	R2=A+B
LOAD	R3, C	R3=C
SUB	R1, R3	R1=A-C
LOAD	R4, D	R4=D
MUL	R3, R4	R3=C*D
SUB	R2, R3	R2=A+B-C*D
DIV	R2, R1	R2= (A+B-C*D)/(A-C)
ADD	R2, R4	R2=(A+B-C*D)/(A-C)+D
STORE	R2, Z	**Z=(A+B-C*D)/(A-C)+D**

Το πρόγραμμα Γ χρησιμοποιεί τους τέσσερις καταχωρητές γενικού σκοπού που διαθέτει ο υπολογιστής μας και 5 εντολές καταχωρητή – μνήμης. Επίσης έχει συνολικά 2 λιγότερες εντολές τόσο από το πρόγραμμα Α όσο και από το Β.

ΑΣΚΗΣΗ 31

Σε μια αρχιτεκτονική βασιζόμενη σε μηχανισμό σωρού, γράψτε πρόγραμμα για τον παρακάτω υπολογισμό :

$$E = ((A \cdot B - C \cdot D) / A) - A \cdot D$$

Οι διαθέσιμες εντολές είναι οι PUSH, POP, MUL, SUB και DIV. Ο πρώτος τελεστέος κάθε πράξης είναι αυτός που βρίσκεται στην κορυφή του σωρού. Το πρόγραμμά σας δε θα πρέπει να χρησιμοποιεί ενδιάμεσες αποθηκεύσεις.

Λύση

Αρχικά θα μετατρέψουμε την ζητούμενη έκφραση σε postfix notation. Προσοχή χρειάζεται στις μη-αντιμεταθετικές πράξεις, όπου ο δεύτερος τελεστέος στην αρχική έκφραση τοποθετείται πρώτος στην postfix notation, καθώς σύμφωνα με την εκφώνηση ο πρώτος τελεστέος κάθε πράξης είναι αυτός που βρίσκεται στην κορυφή του σωρού[*].

$E = ((A \cdot B - C \cdot D) / A) - A \cdot D = ((\underline{AB\cdot} - \underline{CD\cdot}) / A) - \underline{AD\cdot} = (\underline{CD\cdot AB\cdot -} / A) - \underline{AD\cdot} = \underline{ACD\cdot AB\cdot -/} - \underline{AD\cdot}$ = **$\underline{AD\cdot ACD\cdot AB\cdot -/-}$**

	Περιεχόμενα σωρού (Αριστερά αναγράφεται το στοιχείο που βρίσκεται στην κορυφή)
PUSH A	A
PUSH D	D, A
MUL	DxA
PUSH A	A, DxA
PUSH C	C, A, DxA
PUSH D	D, C, A, DxA
MUL	DxC, A, DxA
PUSH A	A, DxC, A, DxA
PUSH B	B, A, DxC, A, DxA

[*] Η μετατροπή μπορεί να επεκταθεί και στις αντιμεταθετικές πράξεις όπου δεν έχει σημασία η σειρά των τελεστών

MUL		BxA, DxC, A, DxA
SUB		(BxA)-(DxC), A, DxA
DIV		[(BxA)-(DxC)]/A, DxA
SUB		[(BxA)-(DxC)]/A- DxA
POP	E	-

E = [(BxA)-(DxC)]/A- DxA

ΑΣΚΗΣΗ 32

Για μια μηχανή μίας διεύθυνσης (**μηχανή συσσωρευτή**) ένας προγραμματιστής έγραψε το ακόλουθο πρόγραμμα :

LOAD	A
ADD	B
MUL	C
DIV	D
STORE	E
LOAD	B
SUB	D
SUB	E
STORE	F

α. Το αποτέλεσμα ποιου υπολογισμού που επιτελείται από το παραπάνω πρόγραμμα αποθηκεύεται στη θέση μνήμης **F**;

β. Δώστε πρόγραμμα που να επιτελεί τον ίδιο υπολογισμό σε μια μηχανή μηδενικών διευθύνσεων (**μηχανή σωρού**). Υποθέστε ότι έχετε στη διάθεσή σας τις εντολές PUSH, POP, ADD, SUB, MUL και DIV. Οι αριθμητικές εντολές εκτελούνται στα δύο κορυφαία στοιχεία του σωρού και το αποτέλεσμά τους επανατοποθετείται στο σωρό. Οι μη αντιμεταθετικοί τελεστές χρησιμοποιούν ως πρώτο τελούμενο (μειωτέο ή διαιρετέο) το κορυφαίο στοιχείο του σωρού.

Λύση

α. Τα περιεχόμενα του συσσωρευτή (Σ) και των επηρεαζόμενων θέσεων μνήμης μετά την εκτέλεση κάθε εντολής του προγράμματος φαίνεται παρακάτω :

LOAD	A	Σ = A
ADD	B	Σ = Σ + B = A + B
MUL	C	Σ = Σ * C = (A + B) * C
DIV	D	Σ = Σ / D = (A + B) * C/D
STORE	E	E = Σ = (A + B) * C / D
LOAD	B	Σ = B
SUB	D	Σ = Σ – D = B - D
SUB	E	Σ = Σ - E = (B - D) - (A + B) * C / D
STORE	F	F = Σ = (B - D) - (A + B) * C / D

Άρα επιτελείται ο υπολογισμός **F = (B - D) - (A + B) * C / D** (1)

β. Για να γράψουμε το αντίστοιχο πρόγραμμα για μηχανή σωρού θα μετασχηματίσουμε την έκφραση στην postfix μορφή της. Πριν όμως αντιμεταθέτουμε τα δεξιά και τα αριστερά μέρη των μη αντιμεταθετικών τελεστών οπότε παίρνουμε την έκφραση :

D / (A + B) * C – (D – B)

Ο μετασχηματισμός ακολουθεί ως εξής :

D / (A + B) * C – (D – B) ↔ D / (A B +) * C – (D B –) ↔ D A B + C * / D B – –

Το ζητούμενο πρόγραμμα και τα περιεχόμενα του σωρού μετά την εκτέλεση κάθε εντολής φαίνονται παρακάτω, με τη κορυφή του σωρού να αναγράφεται στα αριστερά :

PUSH	D	D
PUSH	A	A, D
PUSH	B	B, A, D
ADD		(B + A), D
PUSH	C	C, (B + A), D
MUL		C * (B+A), D

DIV		C* (B+A) / D
PUSH	D	D, C * (B + A) / D
PUSH	B	B, D, C * (B + A) / D
SUB		B - D, C * (B + A) / D
SUB		B – D – C * (B + A) / D
POP	F	άδειος
		F = B – D – C * (B + A) / D (2)

Οι σχέσεις (1) και (2) δείχνουν ότι τα δύο προγράμματα επιτελούν τον ίδιο υπολογισμό.

ΑΣΚΗΣΗ 33

Για μια μηχανή που βασίζεται στη χρήση συσσωρευτή κάποιος προγραμματιστής έγραψε το ακόλουθο πρόγραμμα :

```
LOAD   A
ADD    B
MUL    C
DIV    D
STORE  E
LOAD   B
SUB    D
SUB    E
STORE  F
```

1. Ποια είναι η έκφραση που υπολογίζεται και αποθηκεύεται στη θέση μνήμης F ;
2. Δώστε πρόγραμμα που να επιτελεί τον ίδιο υπολογισμό σε μια μηχανή που βασίζεται στη χρήση μηχανισμού σωρού. Υποθέστε ότι έχετε στη διάθεσή σας τις εντολές PUSH, POP, ADD, SUB, MUL και DIV. Οι αριθμητικές εντολές εκτελούνται στα δύο κορυφαία στοιχεία του σωρού και το αποτέλεσμά τους επανατοποθετείται στο σωρό. Κάθε τελεστής χρησιμοποιεί ως πρώτο έντελο το κορυφαίο στοιχείο του σωρού.

Λύση

1. Τα περιεχόμενα του συσσωρευτή (Σ) και των επηρεαζόμενων θέσεων μνήμης μετά την εκτέλεση κάθε εντολής του προγράμματος φαίνεται παρακάτω :

LOAD	A	Σ = A
ADD	B	Σ = Σ + B = A + B
MUL	C	Σ = Σ * C = (A + B) * C
DIV	D	Σ = Σ / D = [(A + B) * C]/D
STORE	E	E = Σ = [(A + B) * C] / D
LOAD	B	Σ = B
SUB	D	Σ = Σ – D = B – D
SUB	E	Σ = Σ – E = (B – D) – {[(A + B) * C] / D}
STORE	F	F = Σ = (B – D) – {[(A + B) * C] / D}

Αρα επιτελείται ο υπολογισμός **F = (B – D) – {[(A + B) * C] / D}** (1)

2. Για να γράψουμε το αντίστοιχο πρόγραμμα για μηχανή σωρού θα μετασχηματίσουμε την infix μορφή της έκφρασης σε postfix μορφή. Πριν όμως, αντιμεταθέτουμε τα δεξιά και τα αριστερά μέρη των μη αντιμεταθετικών τελεστών, οπότε παίρνουμε την έκφραση :

$$\{ D / [(A + B) * C] \} – (D – B)$$

Ο μετασχηματισμός σε postfix μορφή ακολουθεί ως εξής (υπογραμμίζονται οι υποεκφράσεις που έχουν μετατραπεί σε postfix) :

$$\{ D / [(A + B) * C] \} – (D – B) \leftrightarrow$$
$$\{ D / [\underline{(A\ B +)} * C] \} – \underline{(D\ B –)} \leftrightarrow$$
$$[D / \underline{(A\ B + C *)}] – \underline{(D\ B –)} \leftrightarrow$$
$$\underline{(D\ A\ B + C * /)} – \underline{(D\ B –)} \leftrightarrow$$
$$\underline{D\ A\ B + C * / D\ B – –}$$

Το ζητούμενο πρόγραμμα και τα περιεχόμενα του σωρού μετά την εκτέλεση κάθε εντολής φαίνονται παρακάτω, με τη κορυφή του σωρού να αναγράφεται στα αριστερά :

PUSH	D	D
PUSH	A	A, D
PUSH	B	B, A, D
ADD		(B + A), D
PUSH	C	C, (B + A), D
MUL		[C * (B+A)], D
DIV		[C* (B+A)] / D
PUSH	D	D, { [C * (B + A)] / D }
PUSH	B	B, D, { [C * (B + A)] / D }
SUB		(B – D), { [C * (B + A)] / D }
SUB		(B – D) – { [C * (B + A)] / D }
POP	F	Σωρός άδειος
		F = (B – D) – { [C * (B + A)] / D } (2)

Οι σχέσεις (1) και (2) δείχνουν ότι τα δύο προγράμματα επιτελούν τον ίδιο υπολογισμό.

ΑΣΚΗΣΗ 34

Δίδεται το επόμενο πρόγραμμα, γραμμένο σε συμβολική γλώσσα μιας μηχανής 2 διευθύνσεων, όπου A, B, C και F, διευθύνσεις της κύριας μνήμης και R_x, διευθύνσεις καταχωρητών. Το πρώτο αναγραφόμενο έντελο είναι και το έντελο αποθήκευσης του αποτελέσματος πλην της εντολής STORE.

MOV	R_1,	A
MOV	R_2,	B
MOV	R_3,	C
ADD	R_2,	R_1
MUL	R_2,	R_3
SUB	R_1,	R_3
DIV	R_1,	R_2
STORE	R_1,	F

Ζητούνται :

1. Να βρείτε τι αποθηκεύει το παραπάνω πρόγραμμα στη θέση μνήμης F.
2. Να δώσετε ισοδύναμο πρόγραμμα σε συμβολική γλώσσα μιας μηχανής συσσωρευτή. Χρησιμοποιείστε τις εντολές ADD, SUB, MUL, DIV, LOAD και STORE.
3. Να δώσετε ισοδύναμο πρόγραμμα σε συμβολική γλώσσα για μια μηχανή μηδενικών διευθύνσεων (μηχανή σωρού). Οι αριθμητικές εντολές εκτελούνται στα δύο κορυφαία στοιχεία του σωρού και το αποτέλεσμά τους επανατοποθετείται στο σωρό. Κάθε τελεστής χρησιμοποιεί ως πρώτο έντελο το κορυφαίο στοιχείο του σωρού.

Λύση

1. Ας παρακολουθήσουμε την εκτέλεση του προγράμματος εντολή προς εντολή :

Εντολή			**Αποτέλεσμα**
MOV	R_1,	A	$R_1 = A$
MOV	R_2,	B	$R_2 = B$
MOV	R_3,	C	$R_3 = C$
ADD	R_2,	R_1	$R_2 = R_2 + R_1 = B + A$
MUL	R_2,	R_3	$R_2 = R_2 \cdot R_3 = (B+A) \cdot C$
SUB	R_1,	R_3	$R_1 = R_1 – R_3 = A – C$
DIV	R_1,	R_2	$R_1 = R_1 / R_2 = (A – C) / [(B+A) \cdot C]$
STORE	R_1,	F	$F = (A – C) / (B+A) \cdot C$

2. Το ζητούμενο πρόγραμμα καθώς και τα αποτελέσματά του κατά την εκτέλεση κάθε εντολής έχουν ως εξής (Σ συμβολίζει το συσσωρευτή):

Εντολή		**Αποτέλεσμα**
LOAD	A	Σ = A
ADD	B	Σ = Σ + B = A + B
MUL	C	Σ = Σ · C = (A + B) · C
STORE	F	F = Σ = (A + B) · C
LOAD	A	Σ = A
SUB	C	Σ = Σ – C = A – C
DIV	F	Σ = Σ / F = (A – C) / [(A + B) · C]
STORE	F	F = Σ = (A – C) / [(A + B) · C]

3. Ξεκινώντας από την αρχική έκφραση αντιμεταθέτουμε τα έντελα των μη αντιμεταθετικών τελεστών, οπότε παίρνουμε την έκφραση [(B+A) · C] / (C – A). Τη μετατρέπουμε με διαδοχικούς μετασχηματισμούς σε postfix ως εξής :

   [(B+A) · C] / (C – A) ↔ [(BA+)· C] / (CA–) ↔ (BA+C·) / (CA–) ↔ BA+C· CA–/

   Το ζητούμενο πρόγραμμα και τα περιεχόμενα του σωρού κατά την εκτέλεσή του φαίνονται παρακάτω (η κορυφή του σωρού αναγράφεται αριστερά και υποθέτουμε ότι αρχικά ο σωρός είναι άδειος).

Πρόγραμμα		**Περιεχόμενα Σωρού**
PUSH	B	B
PUSH	A	A, B
ADD		A + B
PUSH	C	C, A + B
MUL		C · (A + B)
PUSH	C	C, [C · (A + B)]
PUSH	A	A, C, [C · (A + B)]
SUB		(A – C), [C · (A + B)]
DIV		(A – C) / [C · (A + B)]
POP	F	Σωρός χωρίς στοιχεία & F =(A–C)/[C· (A+B)]

ΑΣΚΗΣΗ 35

Δώστε πρόγραμμα σε αρχιτεκτονική σωρού, συσσωρευτή, καταχωρητή-μνήμης, καταχωρητή-καταχωρητή για την παρακάτω έκφραση

X=A+(B-C)/(D·E)-C

Το πρόγραμμα σε κάθε μία από τις παραπάνω μηχανές πρέπει να είναι το μικρότερο δυνατό. Θεωρείστε ότι κάθε δυαδικός τελεστής στη μηχανή σωρού εκτελεί την πράξη (κορυφαίο στοιχείο – τελεστής – επόμενο στοιχείο στο σωρό). Επιπλέον στην αρχιτεκτονική καταχωρητή-καταχωρητή έχετε στην διάθεση σας μόνο 4 καταχωρητές.

Λύση

α) Αρχιτεκτονική σωρού

Μετατρέπουμε την έκφραση διαδοχικά σε postfix με τους ακόλουθους μετασχηματισμούς

A+(B-C)/(D·E)-C =
A+(CB-)/(DE·)-C =
A+ DE·CB-/-C =
ADE·CB-/+-C =
CADE·CB-/+-

Προσέξτε ότι αντιμεταθέτουμε τα έντελα των μη αντιμεταθετικών τελεστών, ώστε να υπολογιστεί σωστά η πράξη

Το ζητούμενο πρόγραμμα και τα περιεχόμενα του σωρού κατά την εκτέλεσή του φαίνονται παρακάτω (η κορυφή του σωρού αναγράφεται αριστερά και υποθέτουμε ότι αρχικά ο σωρός είναι άδειος).

Πρόγραμμα	**Περιεχόμενα Σωρού**
PUSH C	C
PUSH A	A, C
PUSH D	D, A, C
PUSH E	E, D, A, C
MUL	E·D, A, C
PUSH C	C, E·D, A, C
PUSH B	B, C, E·D, A, C
SUB	(B – C), E·D, A, C
DIV	(B – C)/(E·D), A, C
ADD	(B – C)/(E·D)+A, C
SUB	(B – C)/(E·D)+A-C
POP X	Σωρός χωρίς στοιχεία & X = (B – C)/E·D+A-C

β) Αρχιτεκτονική Συσσωρευτή
Το ζητούμενο πρόγραμμα καθώς και τα αποτελέσματά του κατά την εκτέλεση κάθε εντολής έχουν ως εξής (Σ συμβολίζει το συσσωρευτή):

Εντολή		**Αποτέλεσμα**
LOAD	B	Σ = B
SUB	C	Σ = Σ - C = B - C
DIV	D	Σ = Σ/D = (B - C) / D
DIV	E	Σ = Σ/E = (B - C) / D·E
SUB	C	Σ = Σ – C = (B - C) / D·E – C
ADD	A	Σ = Σ + A = (B - C) / D·E – C +A
STORE	X	X = Σ = (B - C) / D·E – C +A

γ) Αρχιτεκτονική Καταχωρητή - μνήμης
Το ζητούμενο πρόγραμμα καθώς και τα αποτελέσματά του κατά την εκτέλεση κάθε εντολής έχουν ως εξής:

Εντολή		**Αποτέλεσμα**
LOAD	R1,B	R1 = B
SUB	R1,C	R1 = R1 - C = B – C
DIV	R1,D	R1 = (B – C)/D
DIV	R1,E	R1 = (B – C)/(D·E)
SUB	R1,C	R1 = R1 – C = (B - C) / D·E – C
ADD	R1,A	R1 = R1 + A = (B - C) / D·E – C +A
STORE	R1,X	X = R1 = (B - C) / D·E – C +A

δ) Αρχιτεκτονική Καταχωρητή - Καταχωρητή
Το ζητούμενο πρόγραμμα καθώς και τα αποτελέσματά του κατά την εκτέλεση κάθε εντολής έχουν ως εξής:

Εντολή		**Αποτέλεσμα**
LOAD	R1,B	R1 = B
LOAD	R2,C	R2 = C
SUB	R1,R1,R2	R1 = R1 – R2 = B – C
LOAD	R3,D	R3 = D
LOAD	R4,E	R4 = E
MUL	R3,R3,R4	R3 = R3·R4 = D·E
DIV	R1,R1,R3	R1 = R1/R3 = (B - C) / D·E
SUB	R1,R1,R2	R1 = R1 – R2 = (B - C) / D·E – C
LOAD	R3,A	R3 = A
ADD	R1,R1,R3	R1 = R1 + R3 = (B - C) / D·E – C +A
STORE	X,R1	X = R1 = (B - C) / D·E – C +A

Εκτέλεση Αριθμητικών Πράξεων σε Επεξεργαστικές Μονάδες

ΑΣΚΗΣΗ 36

Οι αριθμοί Α = $01101010_{(2)}$, Β = $10011110_{(2)}$ και Γ = $10000001_{(2)}$ είναι σε παράσταση συμπληρώματος ως προς 2. Να δείξετε πως θα εκτελεστούν οι πράξεις Α+Β και Β+Γ σε κάθε μία από τις ακόλουθες περιπτώσεις και να σχολιάσετε τα αποτελέσματα που θα προκύψουν.
α. Οι πράξεις γίνονται σε ένα αθροιστή των 8 δυαδικών ψηφίων.
β. Οι πράξεις γίνονται σε ένα αθροιστή των 16 δυαδικών ψηφίων.

Λύση:
α. Οι πράξεις γίνονται σε ένα αθροιστή των 8 δυαδικών ψηφίων.

A+B

```
  0 1 1 0 1 0 1 0
 +1 0 0 1 1 1 1 0
--------------------
1 0 0 0 0 1 0 0 0
```

Υπερχείλιση έχουμε όταν $\kappa_7 \oplus \kappa_8 = 1$, όπου κ_7 και κ_8 είναι τα κρατούμενα εξόδου της προτελευταίας και της τελευταίας βαθμίδας του αθροιστή αντίστοιχα. Στην προκειμένη περίπτωση έχουμε $\kappa_7 = \kappa_8 = 1$, άρα $\kappa_7 \oplus \kappa_8 = 0$. Επομένως δεν έχουμε υπερχείλιση.

Όταν προσθέτουμε αριθμούς σε παράσταση συμπληρώματος ως προς 2 το κρατούμενο εξόδου αγνοείται, επομένως το αποτέλεσμα είναι Α+Β = $0\,0\,0\,0\,1\,0\,0\,0_{(2)}$.

B + Γ

```
  1 0 0 1 1 1 1 0
+ 1 0 0 0 0 0 0 1
--------------------
1 0 0 0 1 1 1 1 1
```

Στην προκειμένη περίπτωση έχουμε $\kappa_7 = 0$ και $\kappa_8 = 1$, άρα $\kappa_7 \oplus \kappa_8 = 1$. Επομένως έχουμε υπερχείλιση.

β. Οι πράξεις γίνονται σε ένα αθροιστή των 16 δυαδικών ψηφίων.

Επειδή οι αριθμοί μας είναι των 8 δυαδικών ψηφίων εφαρμόζουμε την τεχνική της επέκτασης προσήμου, δηλαδή επαναλαμβάνουμε το πιο σημαντικό ψηφίο του αριθμού στις θέσεις από εννέα μέχρι και δεκαέξι.

A+B

```
  0 0 0 0 0 0 0 0 0 1 1 0 1 0 1 0
+ 1 1 1 1 1 1 1 1 1 0 0 1 1 1 1 0
---------------------------------------
1 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0
```

Στην προκειμένη περίπτωση έχουμε $\kappa_{15} = \kappa_{16} = 1$, όπου κ_{15} και κ_{16} είναι τα κρατούμενα εξόδου της προτελευταίας και της τελευταίας βαθμίδας του αθροιστή αντίστοιχα, άρα $\kappa_{15} \oplus \kappa_{16} = 0$. Επομένως δεν έχουμε υπερχείλιση. Όταν προσθέτουμε αριθμούς σε παράσταση συμπληρώματος ως προς 2 το κρατούμενο εξόδου αγνοείται, επομένως το αποτέλεσμα είναι:
Α+Β = $0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,1\,0\,0\,0_{(2)}$.

B + Γ

```
  1 1 1 1 1 1 1 1 1 0 0 1 1 1 1 0
+ 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 1
-----------------------------------
1 1 1 1 1 1 1 1 1 0 0 0 1 1 1 1 1
```

Στην προκειμένη περίπτωση έχουμε $\kappa_{15} = \kappa_{16} = 1$, άρα $\kappa_{15} \oplus \kappa_{16} = 0$. Επομένως δεν έχουμε υπερχείλιση. Όταν προσθέτουμε αριθμούς σε παράσταση συμπληρώματος ως προς 2 το κρατούμενο εξόδου αγνοείται, επομένως το αποτέλεσμα είναι:
$B + \Gamma = 1\,1\,1\,1\,1\,1\,1\,1\,0\,0\,0\,1\,1\,1\,1\,1_{(2)}$.

ΑΣΚΗΣΗ 37

Εστω οι αριθμοί :

- $A = -56_{10}$
- $B = -114_{10}$
- $\Gamma = 3F_{16}$

Χρησιμοποιώντας αναπαράσταση συμπληρώματος ως προς 2 και ακρίβεια 8 δυαδικών ψηφίων εκτελέστε τις πράξεις A+B, A-Γ, B+Γ.

1. Σχολιάστε και δικαιολογείστε την ορθότητα ή μη των αποτελεσμάτων.
2. Ποιά είναι τα ελάχιστα δυαδικά ψηφία που απαιτούνται για την ορθή εκτέλεση των παραπάνω πράξεων ?

Λύση

- Εχουμε ότι $A=-56_{10}$. Σε 8 δυαδικά ψηφία ο 56_{10} έχει αναπαράσταση 00111000_2 και συνεπώς η αναπαράσταση του -56_{10} σε συμπλήρωμα ως προς 2 είναι $11000111_2 + 1 = 11001000_{2's}$.
- Ομοια βρίσκουμε ότι $B = 10001110_{2's}$.
- Τέλος $\Gamma = 3F_{16} = 00111111_2 = 00111111_{2's} = 63_{10}$. Επίσης $-\Gamma = 11000001_{2's}$

➢ Για την πράξη A+B παίρνουμε : 11001000 + 10001110 = 01010110 και κρατούμενο εξόδου. Το αποτέλεσμα είανι προφανώς λανθασμένο αφού με την πρόσθεση δύο αρνητικών αριθμών (το πιο σημαντικό ψηφίο των προσθετέων είναι 1) πήραμε ως αποτέλεσμα θετικό αριθμό (το πιο σημαντικό ψηφίο του αποτελέσματος είναι 0). Ενώ δηλαδή αναμέναμε το $A+B = -56_{10} - 114_{10} = -170_{10}$, βρήκαμε $01010110_2 = 86_{10}$. Το λάθος οφείλεται σε υπερχείλιση. Πιο συγκεκριμένα, το κρατούμενο προς τη τελευταία βαθμίδα (κ_6) είναι 0 ενώ το κρατούμενο εξόδου (κ_7) είναι 1. Ο ενδείκτης υπερχείλισης (Y) –βλέπε σχήμα 3.3 της σελίδας 81 του Τόμου Β'– ενός αθροιστή των 8 δυαδικών ψηφίων θα μας δώσει ένδειξη υπερχείλισης αφού $Y = \kappa_7 \oplus \kappa_6 = 1$.

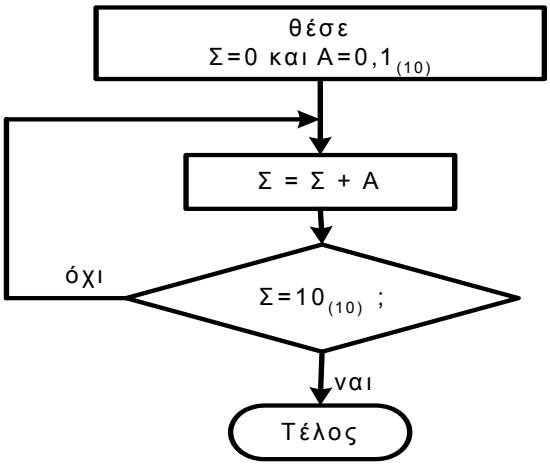
Για τη σωστή εκτέλεση της παραπάνω πράξης απαιτούνται τόσα ψηφία ώστε να μπορεί να αναπαρασταθεί ασφαλώς το αποτέλεσμα. Σύμφωνα με τη σελίδα 56 του Ψηφιακή Σχεδίαση Ι, με n ψηφία μπορώ στο συμπλήρωμα ως προς 2 να αναπαραστήσω το διάστημα των ακεραίων $[-2^{n-1}, 2^{n-1})$. Αρα για την αναπαράσταση του -170_{10} θα χρειαζόταν ακρίβεια 9 δυαδικών ψηφίων.

➢ $A - \Gamma = 11001000_{2's} + 11000001_{2's} = 10001001_{2's} = -119_{10}$ και $\kappa_7 = 1$. Το αποτέλεσμα είναι σωστό αφού $A - \Gamma = -56_{10} - 63_{10} = -119_{10}$. Επίσης $Y=\kappa_6 \oplus \kappa_7 = 1 \oplus 1 = 0$ και συνεπώς δεν υπάρχει ένδειξη υπερχείλισης.

➢ $B + \Gamma = 10001110_{2's} + 00111111_{2's} = 11001101_{2's} = -51_{10}$ και $\kappa_7=0$. Το αποτέλεσμα είναι σωστό αφού $B + \Gamma = -114_{10} + 63_{10} = -51_{10}$. Επίσης $Y=\kappa_6 \oplus \kappa_7 = 0 \oplus 0 = 0$ και δεν υπάρχει ένδειξη υπερχείλισης. Επίσης μπορούμε να πούμε ότι στην περίπτωση αυτή δεν μπορεί να υπάρξει πρόβλημα υπερχείλισης αφού προστίθενται ετερόσημοι αριθμοί.

ΑΣΚΗΣΗ 38

Να περιγράψετε τι θα γίνει κατά την εκτέλεση κάποιου προγράμματος που υλοποιεί το κάτωθι λογικό διάγραμμα. Θεωρείστε ότι ο υπολογιστής στον οποίο θα εκτελεστεί το πρόγραμμα υποστηρίζει μόνο το δυαδικό σύστημα αναπαράστασης αριθμών. Να δικαιολογήσετε την απάντησή σας.



Λύση :

Αφού ο υπολογισής μας υποστηρίζει μόνο το δυαδικό σύστημα αναπαράστασης, η αναπαράσταση του $A=0,1_{(10)}$ θα γίνει χρησιμοποιώντας το δυαδικό σύστημα. Επομένως πρέπει να βρούμε την δυαδική παράσταση του δεκαδικού αριθμού 0,1. Αυτό θα γίνει με διαδοχικούς πολλαπλασιασμούς με την βάση του δυαδικού συστήματος που είναι το 2. Για να πάρουμε μία ακριβή παράσταση του δεκαδικού αριθμού 0,1 στο δυαδικό θα πρέπει να κάνουμε τόσους πολλαπλασιασμούς μέχρι το κλασματικό μέρος που θα προκύψει να είναι μηδέν.

Βήμα 1. α_{-1} = ακέραιο μέρος του $2\cdot0,1 = 0$ και κλασματικό =0,2
Βήμα 2. α_{-2} = ακέραιο μέρος του $2\cdot0,2 = 0$ και κλασματικό =0,4
Βήμα 3. α_{-3} = ακέραιο μέρος του $2\cdot0,4 = 0$ και κλασματικό =0,8
Βήμα 4. α_{-4} = ακέραιο μέρος του $2\cdot0,8 = 1$ και κλασματικό =0,6
Βήμα 5. α_{-5} = ακέραιο μέρος του $2\cdot0,6 = 1$ και κλασματικό =0,2
Βήμα 6. α_{-6} = ακέραιο μέρος του $2\cdot0,2 = 0$ και κλασματικό =0,4
Βήμα 7. α_{-7} = ακέραιο μέρος του $2\cdot0,4 = 0$ και κλασματικό =0,8
Βήμα 8. α_{-8} = ακέραιο μέρος του $2\cdot0,8 = 1$ και κλασματικό =0,6

Παρατηρούμε μία περιοδικότητα στα διαδοχικά κλασματικά μέρη που προκύπτουν: 0,2 0,4 0,8 0,6 και πάλι 0,2 0,4 0,8 0,6 κλπ. Επομένως, δε θα πάρουμε ποτέ κλασματικό μέρος ίσο με μηδέν. Αρα δεν είναι δυνατόν να παραστήσουμε τον δεκαδικό αριθμό 0,1 ακριβώς στο δυαδικό σύστημα με πεπερασμένο αριθμό δυαδικών ψηφίων. Αυτό θα έχει ως αποτέλεσμα το άθροισμα Σ που προκύπτει μετά από 100 διαδοχικές προσθέσεις του Α να είναι πάντα μικρότερο του $10_{(10)}$, επομένως δεν θα σταματήσει η εκτέλεση του προγράμματος μετά από 100 επαναλήψεις όπως θα περιμέναμε συγκρίνοντας τους δεκαδικούς αριθμούς. Υπάρχει βέβαια η περίπτωση η εκτέλεση του προγράμματος να σταματήσει μετά από περισσότερες επαναλήψεις εάν τύχει και το πλήθος των δυαδικών ψηφίων που χρησιμοποιούνται για την παράσταση του 0,1 στο δυαδικό είναι τόσα ώστε η τιμή που πραγματικά απεικονίζουν είναι ένα ακέραιο υποπολλαπλάσιο του δεκαδικού αριθμού δέκα. Αυτό συμβαίνει για παράδειγμα μετά από 160 επαναλήψεις όταν για την παράσταση του 0,1 στο δυαδικό χρησιμοποιούνται 4 δυαδικά ψηφία.

Η άσκηση αυτή ας προβληματίσει σοβαρά όσους ισχυρίζονται ότι μπορούν να γίνουν καλοί προγραμματιστές αν αγνοούν τη δομή και οργάνωση του υπολογιστή ή τον τρόπο αναπαράστασης της πληροφορίας στον υπολογιστή.

ΑΣΚΗΣΗ 39

Βρείτε τις τιμές των καταχωρητών και των σημαιών του καταχωρητή κατάστασης (status register) μετά την εκτέλεση κάθε εντολής του ακόλουθου **προγράμματος**, όπου:

α. ADD X, Y, Z σημαίνει X←Y+Z χρησιμοποιώντας τον κατάλληλο τρόπο διευθυνσιοδότησης
β. όλοι οι αριθμοί στον υπολογιστή μας εμφανίζονται σε παράσταση συμπληρώματος ως προς 2 και
γ. η Αριθμητική Λογική Μονάδα εκτελεί πράξεις μεταξύ αριθμών των 16 δυαδικών ψηφίων σε παράσταση συμπληρώματος ως προς 2.

Πρόγραμμα

ADD R_2, R_2, # $-15_{(10)}$ ***το # δηλώνει άμεσο (immediate) τρόπο διευθυνσιοδότησης
ADD R_1, R_1, R_3
ADD R_3, R_3, R_4
ADD R_5, R_3, R_4

Αρχικές τιμές των καταχωρητών

R_1	$0014_{(16)}$
R_2	$000F_{(16)}$
R_3	$0FFF_{(16)}$
R_4	$ABCD_{(16)}$
R_5	$0000_{(16)}$

Καταχωρητής Κατάστασης

C	Y	Z	N
Σημαία κρατουμένου εξόδου	Σημαία υπερχείλισης	Σημαία μηδενικού αποτελέσματος	Σημαία αρνητικού αποτελέσματος

Λύση

- *Εκτέλεση της εντολής ADD R_2, R_2, # $-15_{(10)}$*

Η εκτέλεση της πρώτης εντολής του προγράμματος θα έχει ως συνέπεια την πρόσθεση του περιεχομένου του καταχωρητή R_2 και του αριθμού $-15_{(10)}$ και την αποθήκευση του αποτελέσματος στον καταχωρητή R_2. Επειδή η Αριθμητική Λογική Μονάδα εκτελεί πράξεις μεταξύ αριθμών των 16 δυαδικών ψηφίων σε παράσταση συμπληρώματος ως προς 2 θα πρέπει ο αριθμός $-15_{(10)}$ να παρασταθεί σε παράσταση συμπληρώματος ως προς 2 με 16 δυαδικά ψηφία. Στο δυαδικό ο αριθμός $15_{(10)}$ χρησιμοποιώντας 16 δυαδικά ψηφία έχει την παράσταση 0000000000001111 άρα το συμπλήρωμά του ως προς δύο που θα παριστάνει τον αριθμό $-15_{(10)}$ είναι 1111111111110001. Επομένως:

$000F_{(16)}$	0000000000001111
- $15_{(10)}$	+1111111111110001
	10000000000000000

Παρατηρούμε ότι υπάρχει κρατούμενο εξόδου επομένως η σημαία του κρατούμενου εξόδου θα τεθεί, δηλαδή το αντίστοιχο δυαδικό ψηφίο στον καταχωρητή κατάστασης θα λάβει την τιμή 1. Επειδή υπάρχει τόσο κρατούμενο εξόδου όσο και κρατούμενο από την προτελευταία βαθμίδα του αθροιστή προς την τελευταία δεν υπάρχει υπερχείλιση επομένως η σημαία υπερχείλισης τίθεται στο 0. Το αποτέλεσμα της πράξης του εκτελέστηκε είναι μηδενικό γι' αυτό η σημαία μηδενικού αποτελέσματος τίθεται στο 1. Το πιο σημαντικό δυαδικό ψηφίο του αποτελέσματος μετά το κρατούμενο εξόδου έχει την τιμή 0 οπότε το αποτέλεσμα δεν είναι αρνητικός αριθμός γι' αυτό η σημαία αρνητικού αποτελέσματος τίθεται στο 0.

R_1	$0014_{(16)}$
R_2	$0000_{(16)}$
R_3	$0FFF_{(16)}$
R_4	$ABCD_{(16)}$
R_5	$0000_{(16)}$

Καταχωρητής Κατάστασης

1	0	1	0
C	Y	Z	N
Σημαία κρατουμένου εξόδου	Σημαία υπερχείλισης	Σημαία μηδενικού αποτελέσματος	Σημαία αρνητικού αποτελέσματος

- *Εκτέλεση της εντολής ADD R1, R1, R3*

$0014_{(16)}$	0000000000010100
$+ 0FFF_{(16)}$	+ 0000111111111111
	0001000000010011

Παρατηρούμε ότι δεν υπάρχει κρατούμενο εξόδου επομένως η σημαία του κρατούμενου εξόδου δεν θα τεθεί, δηλαδή το αντίστοιχο δυαδικό ψηφίο στον καταχωρητή κατάστασης θα λάβει την τιμή 0. Επειδή δεν υπάρχει ούτε κρατούμενο εξόδου ούτε και κρατούμενο από την προτελευταία βαθμίδα του αθροιστή προς την τελευταία δεν υπάρχει υπερχείλιση επομένως η σημαία υπερχείλισης τίθεται στο 0. Το αποτέλεσμα της πράξης του εκτελέστηκε δεν είναι μηδενικό γι' αυτό η σημαία μηδενικού αποτελέσματος τίθεται στο 0. Το πιο σημαντικό δυαδικό ψηφίο του αποτελέσματος έχει την τιμή 0 οπότε το αποτέλεσμα δεν είναι αρνητικός αριθμός γι' αυτό η σημαία αρνητικού αποτελέσματος τίθεται στο 0.

R_1	$1013_{(16)}$
R_2	$0000_{(16)}$
R_3	$0FFF_{(16)}$
R_4	$ABCD_{(16)}$
R_5	$0000_{(16)}$

Καταχωρητής Κατάστασης

0	0	0	0
C	Y	Z	N
Σημαία κρατουμένου εξόδου	Σημαία υπερχείλισης	Σημαία μηδενικού αποτελέσματος	Σημαία αρνητικού αποτελέσματος

- *Εκτέλεση της εντολής ADD R3, R3, R4*

$0\ F\ F\ F_{(16)}$	0000111111111111
$+ ABCD_{(16)}$	+ 1010101111001101
	1011101111001100

Παρατηρούμε ότι δεν υπάρχει κρατούμενο εξόδου επομένως η σημαία του κρατούμενου εξόδου δεν θα τεθεί, δηλαδή το αντίστοιχο δυαδικό ψηφίο στον καταχωρητή κατάστασης θα λάβει την τιμή 0. Επειδή δεν υπάρχει ούτε κρατούμενο εξόδου ούτε και κρατούμενο από την προτελευταία βαθμίδα του αθροιστή προς την τελευταία δεν υπάρχει υπερχείλιση επομένως η σημαία υπερχείλισης τίθεται στο 0. Το αποτέλεσμα της πράξης του εκτελέστηκε δεν είναι μηδενικό γι' αυτό η σημαία μηδενικού αποτελέσματος τίθεται στο 0. Το πιο σημαντικό δυαδικό ψηφίο του αποτελέσματος έχει την τιμή 1 οπότε το αποτέλεσμα είναι αρνητικός αριθμός γι' αυτό η σημαία αρνητικού αποτελέσματος τίθεται στο 1.

R_1	$1013_{(16)}$
R_2	$0000_{(16)}$
R_3	$BBCC_{(16)}$
R_4	$ABCD_{(16)}$
R_5	$0000_{(16)}$

Καταχωρητής Κατάστασης

0	0	0	1
C	Y	Z	N
Σημαία κρατουμένου εξόδου	Σημαία υπερχείλισης	Σημαία μηδενικού αποτελέσματος	Σημαία αρνητικού αποτελέσματος

- *Εκτέλεση της εντολής ADD R5, R3, R4*

$BBCC_{(16)}$	1011101111001100
+ $ABCD_{(16)}$	+ 1010101111001101
	10110011110011001

Παρατηρούμε ότι υπάρχει κρατούμενο εξόδου επομένως η σημαία του κρατούμενου εξόδου θα τεθεί, δηλαδή το αντίστοιχο δυαδικό ψηφίο στον καταχωρητή κατάστασης θα λάβει την τιμή 1. Επειδή υπάρχει κρατούμενο εξόδου αλλά δεν υπάρχει κρατούμενο από την προτελευταία βαθμίδα το αθροιστή προς την τελευταία υπάρχει υπερχείλιση επομένως η σημαία υπερχείλισης τίθεται στο 1. Το αποτέλεσμα της πράξης του εκτελέστηκε δεν είναι μηδενικό γι' αυτό η σημαία μηδενικού αποτελέσματος τίθεται στο 0. Το πιο σημαντικό δυαδικό ψηφίο του αποτελέσματος μετά το κρατούμενο εξόδου έχει την τιμή 0 οπότε το αποτέλεσμα δεν είναι αρνητικός αριθμός γι' αυτό η σημαία αρνητικού αποτελέσματος τίθεται στο 0.

R_1	$1013_{(16)}$
R_2	$0000_{(16)}$
R_3	$BBCC_{(16)}$
R_4	$ABCD_{(16)}$
R_5	$6799_{(16)}$

Καταχωρητής Κατάστασης

1	1	0	0
C	Y	Z	N
Σημαία κρατουμένου εξόδου	Σημαία υπερχείλισης	Σημαία μηδενικού αποτελέσματος	Σημαία αρνητικού αποτελέσματος

ΑΣΚΗΣΗ 40

Θεωρείστε επεξεργαστή με καταχωρητές A (είναι ο συσσωρευτής), B, C και D των 8 δυαδικών ψηφίων, αριθμητική μονάδα των 8 δυαδικών ψηφίων που εκτελεί πράξεις μεταξύ αριθμών των 8 δυαδικών ψηφίων σε παράσταση συμπληρώματος ως προς 2 και τον καταχωρητή κατάστασης SR με τις ακόλουθες σημαίες:

b7	b6	b5	b4	b3	b2	b1	b0
P	-	Y	-	N	Z	-	C

Οι σημαίες κατάστασης έχουν την ακόλουθη σημασία :

P = 1 υποδεικνύει άρτια ισοτιμία (parity), 0 περιτή ισοτιμία
Y = 1 υποδεικνύει υπερχείλιση (overflow), 0 όχι υπερχείλιση
N = 1 υποδεικνύει αρνητικό αποτέλεσμα, 0 θετικό ή μηδενικό
Z = 1 υποδεικνύει μηδενικό αποτέλεσμα, 0 μη μηδενικό.
C = 1 υποδεικνύει ύπαρξη κρατουμένου εξόδου, 0 όχι κρατούμενο.

Θεωρείστε ότι αρχικά P=Y=N=Z=C=0 και ότι τα περιεχόμενα των καταχωρητών είναι:
A= 00100000, B=11100000, C= 10000001 και D= 11100001.
Να δώσετε το περιεχόμενο του καταχωρητή A και την τιμή κάθε σημαίας του καταχωρητή κατάστασης μετά την διαδοχική εκτέλεση κάθε εντολής του παρακάτω προγράμματος. Δικαιολογήσετε την τιμή κάθε σημαίας.

```
ADDA B      ;  A←A+B
ADDA D      ;  A←A+D
ADDA C      ;  A←A+C
```

Λύση

Εκτέλεση πρώτης εντολής:

A	=		00100000
B	=		11100000 +
A	=	1	00000000

Μετά την εκτέλεση της πρώτης εντολής: P=1, Y=0, N=0, Z=1, C=1

Εκτέλεση δεύτερης εντολής:

A	=		00000000
D	=		11100001 +
A	=	0	11100001

Μετά την εκτέλεση της τρίτης εντολής: P=1, Y=0, N=1, Z=0, C=0

Εκτέλεση τρίτης εντολής:

A	=		11100001
C	=		10000001 +
A	=	1	01100010

Μετά την εκτέλεση της δεύτερης εντολής: P=0, Y=1, N=0, Z=0, C=1

ΑΣΚΗΣΗ 41

Θεωρείστε υπολογιστικό σύστημα, το οποίο αναπαριστά ακέραιους αριθμούς σε μορφή **συμπληρώματος ως προς 2.**. Για κάθε ακέραιο ο υπολογιστής διαθέτει **8 δυαδικά** ψηφία. Παρακάτω φαίνεται ο καταχωρητής σημαιών κατάστασης του υπολογιστή:

7	6	5	4	3	2	1	0
N	Z	C	-	V	-	-	-

όπου:

N = σημαία αρνητικού αριθμού. (N=1 υποδεικνύει αρνητικό αποτέλεσμα).
Z = σημαία μηδενικού αποτελέσματος. (Z=1 υποδεικνύει αποτέλεσμα 0).
C = σημαία τελικού κρατουμένου.
V = σημαία υπερχείλισης. (V=1 υποδεικνύει υπερχείλιση από τη τελευταία πράξη).

α. Δώστε την αναπαράσταση των **Α=+37_{16}**, **Β=-43_8** και **Γ=+114_{10}** στο υπολογιστικό αυτό σύστημα. Ο δείκτης υποδεικνύει τη βάση του αριθμητικού συστήματος αναπαράστασης.

β. Δείξτε τη διαδικασία εκτέλεσης των πράξεων **Δ = Α + Β**, **Ε = Β - Γ** στο υπολογιστικό αυτό σύστημα. Καταγράψτε τα αποτελέσματα και την τιμή του καταχωρητή σημαιών κατάστασης μετά από κάθε μία από τις παραπάνω πράξεις. Τέλος, εκτελώντας τις παραπάνω πράξεις στο δεκαδικό, σχολιάστε την ορθότητα κάθε αποτελέσματος.

Λύση

α. Για τον αριθμό Α έχουμε Α = 00110111_2. Για τον αριθμό Β έχουμε $43_8 = 00100011_2$. Επειδή ο Β είναι αρνητικός η αναπαράστασή του προκύπτει προσθέτοντας 1 στην παράσταση που προκύπτει από την αντιστροφή κάθε ψηφίου της δυαδικής του αναπαράστασης, δηλαδή $B_{2'ς} = 11011100 + 1 = 11011101 = -35_{10}$. Τέλος, για τον αριθμό Γ έχουμε $Γ=114_{10}=01110010_2$. Ας σημειωθεί ότι με 8 δυαδικά ψηφία στον κώδικα συμπληρώματος ως προς 2 μπορούν να παρασταθούν όλοι οι ακέραιοι στο διάστημα [-128, 127].

β. Δ = Α + Β = $00110111 + 11011101 = 00010100_{2'ς}$. Ο καταχωρητής κατάστασης παίρνει τη τιμή 001-0---. Επειδή V=0, το αποτελεσμά μας θα είναι σωστό. Πράγματι $00010100_{2'ς} = +20_{10} = 55_{10} +$

(-35_{10}). Υπενθυμίζεται ότι σε συμπλήρωμα ως προς 2, η αφαίρεση εκτελείται με τη πρόσθεση του αντιθέτου. Ο αντίθετος ενός αριθμού προκύπτει από τη πρόσθεση του 1 στη παράσταση που προκύπτει από την αντιστροφή της αρχικής παράστασης του αριθμού. Ε = Β – Γ = Β + (-Γ). Αφού $Γ_{2'ς}$ = 01110010, είναι –Γ = 10001101 + 1 = $10001110_{2'ς}$. Συνεπώς Ε = Β + (-Γ) = 11011101 + 10001110 = $01101011_{2'ς}$ και ο καταχωρητής κατάστασης παίρνει τη τιμή 001-1---. Αφού V=1, το αποτέλεσμά μας είναι λανθασμένο. Πράγματι το αποτέλεσμά μας είναι το $01101011_{2'ς}$ = 107_{10}, ενώ το αναμενόμενο αποτέλεσμα είναι $-35_{10} – 114_{10} = - 149_{10}$. Το ότι θα παίρναμε λανθασμένο αποτέλεσμα ήταν κάτι αναμενόμενο, αφού το -149_{10} δεν ανήκει στο διάστημα έγκυρων αναπαραστάσεων.

ΑΣΚΗΣΗ 42

Οι αριθμοί που ακολουθούν είναι σε παράσταση συμπληρώματος ως προς 2.

β. 11000000
γ. 01111100
δ. 01000001

Να εκτελεστούν οι πράξεις **γ+δ** και **β+δ** σε κάθε μία από τις επόμενες περιπτώσεις, να δοθεί το αποτέλεσμα σε δυαδική και δεκαδική μορφή και να σχολιαστεί αν το αποτέλεσμα είναι σωστό ή όχι και γιατί:

1. Για την πράξη χρησιμοποιείται αθροιστής των 8 δυαδικών ψηφίων
2. Για την πράξη χρησιμοποιείται αθροιστής των 16 δυαδικών ψηφίων.

Λύση

1. Πράξεις με αθροιστή των 8 δυαδικών ψηφίων.

γ + δ

```
    0 1 1 1 1 1 0 0   = 124 (10)
+   0 1 0 0 0 0 0 1   =  65 (10)
  ---------------
    1 0 1 1 1 1 0 1
```

Παρατηρούμε ότι ενώ προσθέτουμε δύο θετικούς αριθμούς, το περισσότερο σημαντικό ψηφίο του αποτελέσματος δηλώνει ότι το αποτέλεσμα είναι αρνητικό, άρα έχει συμβεί υπερχείλιση.
Ο αθροιστής θα δηλώσει την ύπαρξη υπερχείλισης ενεργοποιώντας το σήμα Υ,
Υ=0 ⊕ 1 = 1.

Το πρόβλημα στις ανωτέρω πράξεις προκύπτει διότι τα διαθέσιμα δυαδικά ψηφία (8) δεν είναι αρκετά για να αναπαρασταθεί το αποτέλεσμα. Αφού το αποτέλεσμα είναι λάθος δεν έχει νόημα να το δώσουμε σε δεκαδική μορφή.

Στην πράξη β+δ οι προστιθέμενοι αριθμοί είναι ετερόσημοι, άρα δεν είναι δυνατόν να προκύψει υπερχείλιση. Αυτό συμβαίνει γιατί το αναπαραστάσιμο αποτέλεσμα θα είναι κατά απόλυτη τιμή μικρότερο από την απόλυτη τιμή του μεγαλύτερου ορίσματος, το οποίο και είναι εκφρασμένο με τα διαθέσιμα δυαδικά ψηφία. Συνεπώς το αποτέλεσμα θα απαιτεί το πολύ ίσο αριθμό δυαδικών ψηφίων με το μεγαλύτερο από τα ορίσματα.

β + δ

```
    1 1 0 0 0 0 0 0   = - 64 (10)
+   0 1 0 0 0 0 0 1   =  65 (10)
  ---------------------------
  1 0 0 0 0 0 0 0 1   =   1 (10)
```

Όταν κάνουμε προσθέσεις σε παράσταση συμπληρώματος ως προς 2 το κρατούμενο εξόδου αγνοείται. Και στις δύο ανωτέρω πράξεις παρατηρούμε ότι η έξοδος του αθροιστή Υ θα έχει την τιμή Υ =1 ⊕ 1 = 0.

2. Πράξεις με αθροιστή των 16 δυαδικών ψηφίων.

Για να εκτελέσουμε τις πράξεις με αθροιστές των 16 δυαδικών ψηφίων, αρχικά εκτελούμε *επέκταση προσήμου* (sign extension) των ορισμάτων των πράξεων, έτσι ώστε το μήκος λέξης τους να αυξηθεί από 8 σε 16 δυαδικά ψηφία. Η επέκταση προσήμου σε αριθμό εκφρασμένο σε αναπαράσταση συμπληρώματος βάσης (π.χ. συμπλήρωμα του δύο) είναι η επανάληψη του περισσότερο σημαντικού ψηφίου μιας αρχικής λέξης τόσες φορές ώστε το μήκος της να αυξηθεί κατά το ζητούμενο πλήθος ψηφίων χωρίς να αλλοιωθεί η πληροφορία του προσήμου.

γ + δ

0 0 0 0 0 0 0 0 0 1 1 1 1 1 0 0 = 124 (10)
\+ 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 1 = 65 (10)

0 0 0 0 0 0 0 0 1 0 1 1 1 1 0 1 = 189 (10)

Παρατηρούμε ότι Υ =0 ⊕ 0 = 0, άρα δεν υπάρχει υπερχείλιση. Τα 16 ψηφία είναι αρκετά για την αναπαράσταση του αποτελέσματος.

β + δ

1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 = - 64 (10)
\+ 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 1 = 65 (10)

1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 = 1 (10)

Προφανώς η πράξη β+δ που εκτελούνταν σωστά στον αθροιστή των 8 δυαδικών ψηφίων εκτελείται σωστά και σ' αυτόν των 16 δυαδικών ψηφίων.

ΑΣΚΗΣΗ 43

Θεωρείστε ένα υπολογιστικό σύστημα, το οποίο χρησιμοποιεί συμπλήρωμα ως προς 2 για την αναπαράσταση των ακεραίων αριθμών. Για κάθε ακέραιο ο υπολογιστής διαθέτει 8 δυαδικά ψηφία. Παρακάτω φαίνεται ο καταχωρητής σημαιών κατάστασης του υπολογιστή, 8 δυαδικών ψηφίων :

7	6	5	4	3	2	1	0
N	Z	C	P	V	AC	-	-

όπου :

N = σημαία αρνητικού αριθμού. N=1 υποδεικνύει αρνητικό αποτέλεσμα.
Z = σημαία μηδενικού αποτελέσματος. Z=1 υποδεικνύει αποτέλεσμα 0.
C = σημαία τελικού κρατουμένου.
P = σημαία περιττής ισοτιμίας (τα δυαδικά ψηφία του αποτελέσματος μαζί με το P πρέπει να έχουν περιττή ισοτιμία).
V = σημαία υπερχείλισης. V=1 υποδεικνύει υπερχείλιση από τη τελευταία πράξη.
AC = σημαία εσωτερικού κρατουμένου. Αποθηκεύει το κρατούμενο που προκύπτει από τα τέσσερα λιγότερο σημαντικά δυαδικά ψηφία.

α) Δώστε την αναπαράσταση των $A = +37_{16}$, $B = -43_8$, $\Gamma = +114_{10}$ και $X=+150_{10}$ στο υπολογιστικό αυτό σύστημα. Ο δείκτης υποδεικνύει τη βάση του αριθμητικού συστήματος αναπαράστασης.
β) Δείξτε τη διαδικασία εκτέλεσης των πράξεων Δ = Α + Β, Ε = Β - Γ και Ζ = Α + Γ στο υπολογιστικό αυτό σύστημα. Καταγράψτε τα αποτελέσματα και την τιμή του καταχωρητή

σημαιών κατάστασης μετά από κάθε παραπάνω πράξη. Τέλος εκτελώντας τις παραπάνω πράξεις στο δεκαδικό, σχολιάστε την ορθότητα κάθε αποτελέσματος.

Λύση

α)
Για τον αριθμό Α έχουμε: $A = 37_{16} = 3*16 + 7 = 55_{10} = 00110111_2$. Αφού ο αριθμός είναι θετικός η αναπαράστασή του σε συμπλήρωμα ως προς 2 είναι ίδια με τη δυαδική του αναπαράσταση, δηλαδή $A_{2'ς} = 00110111$.

Για τον αριθμό Β έχουμε: $B = –43_8 = –(4*8+3) = –35_{10} = –00100011_2$. Επειδή ο Β είναι αρνητικός η αναπαράστασή του προκύπτει προσθέτοντας 1 στην παράσταση που προκύπτει από την αντιστροφή κάθε ψηφίου της δυαδικής του αναπαράστασης, δηλαδή $B_{2'ς} = 11011100 + 1 = 11011101 = –35_{10}$.

Για τον αριθμό Γ έχουμε: όπως προκύπτει με διαδοχικές διαιρέσεις με το 2 $Γ=114_{10}=01110010_2$. Και σε αυτή τη περίπτωση ο αριθμός είναι θετικός η αναπαράστασή του σε συμπλήρωμα ως προς 2 είναι ίδια με τη δυαδική του αναπαράσταση, δηλαδή $Γ_{2'ς} = 01110010$.

Για τον αριθμό Χ έχουμε: ο αριθμός Χ δεν μπορεί να αναπαρασταθεί με 8 δυαδικά ψηφία στον κώδικα συμπληρώματος ως προς 2. Ας σημειωθεί ότι με 8 δυαδικά ψηφία στον κώδικα συμπληρώματος ως προς 2 μπορούν να παρασταθούν όλοι οι ακέραιοι στο διάστημα [-128, 127].

β)
Η πράξη Δ = Α + Β παρουσιάζεται ακόλουθα:

0111111 *(κρατούμενα)*
00110111
+11011101
1)00010100

Ο καταχωρητής κατάστασης παίρνει τη τιμή που φαίνεται στον ακόλουθο πίνακα.

7	6	5	4	3	2	1	0
N	Z	C	P	V	AC	-	-
0	0	1	1	0	1	-	-

Επειδή V=0, το αποτέλεσμά μας θα είναι σωστό. Πράγματι $00010100_{2'ς} = +20_{10} = 55_{10} + (–35_{10})$.

Υπενθυμίζεται ότι σε συμπλήρωμα ως προς 2, η αφαίρεση εκτελείται με τη πρόσθεση του αντιθέτου. Ο αντίθετος ενός αριθμού προκύπτει από τη πρόσθεση του 1 στη παράσταση που προκύπτει από την αντιστροφή της αρχικής παράστασης του αριθμού. Η πράξη Ε = Β – Γ = Β + (–Γ) παρουσιάζεται ακόλουθα:
Αφού $Γ_{2'ς} = 01110010$, έχουμε $–Γ = 10001101 + 1 = 10001110_{2'ς}$. Συνεπώς Ε = Β + (–Γ) =

0111111 *(κρατούμενα)*
11011101
+10001110
1)01101011

Ο καταχωρητής κατάστασης παίρνει τη τιμή που φαίνεται στον ακόλουθο πίνακα.

7	6	5	4	3	2	1	0
N	Z	C	P	V	AC	-	-
0	0	1	0	1	1	-	-

Αφού V=1, το αποτέλεσμά μας είναι λανθασμένο. Πράγματι το αποτέλεσμά μας είναι το $01101011_{2'ς}$ = 107_{10}, ενώ το αναμενόμενο αποτέλεσμα είναι $-35_{10} - 114_{10} = -149_{10}$. Το ότι θα παίρναμε λανθασμένο αποτέλεσμα ήταν κάτι αναμενόμενο, αφού το -149_{10} δεν ανήκει στο διάστημα έγκυρων αναπαραστάσεων.

Η πράξη Z = A + Γ παρουσιάζεται ακόλουθα:

1110110 *(κρατούμενα)*
00110111
+01110010
10101001

Το αποτέλεσμα αναπαριστά έναν αρνητικό αριθμό ο οποίος είναι ο $10101001_{2'ς} = -(01010110 + 1) = -01010111_2 = -87_{10}$.

Ο καταχωρητής κατάστασης παίρνει τη τιμή που φαίνεται στον ακόλουθο πίνακα.

7	6	5	4	3	2	1	0
N	Z	C	P	V	AC	-	-
1	0	0	1	1	0	-	-

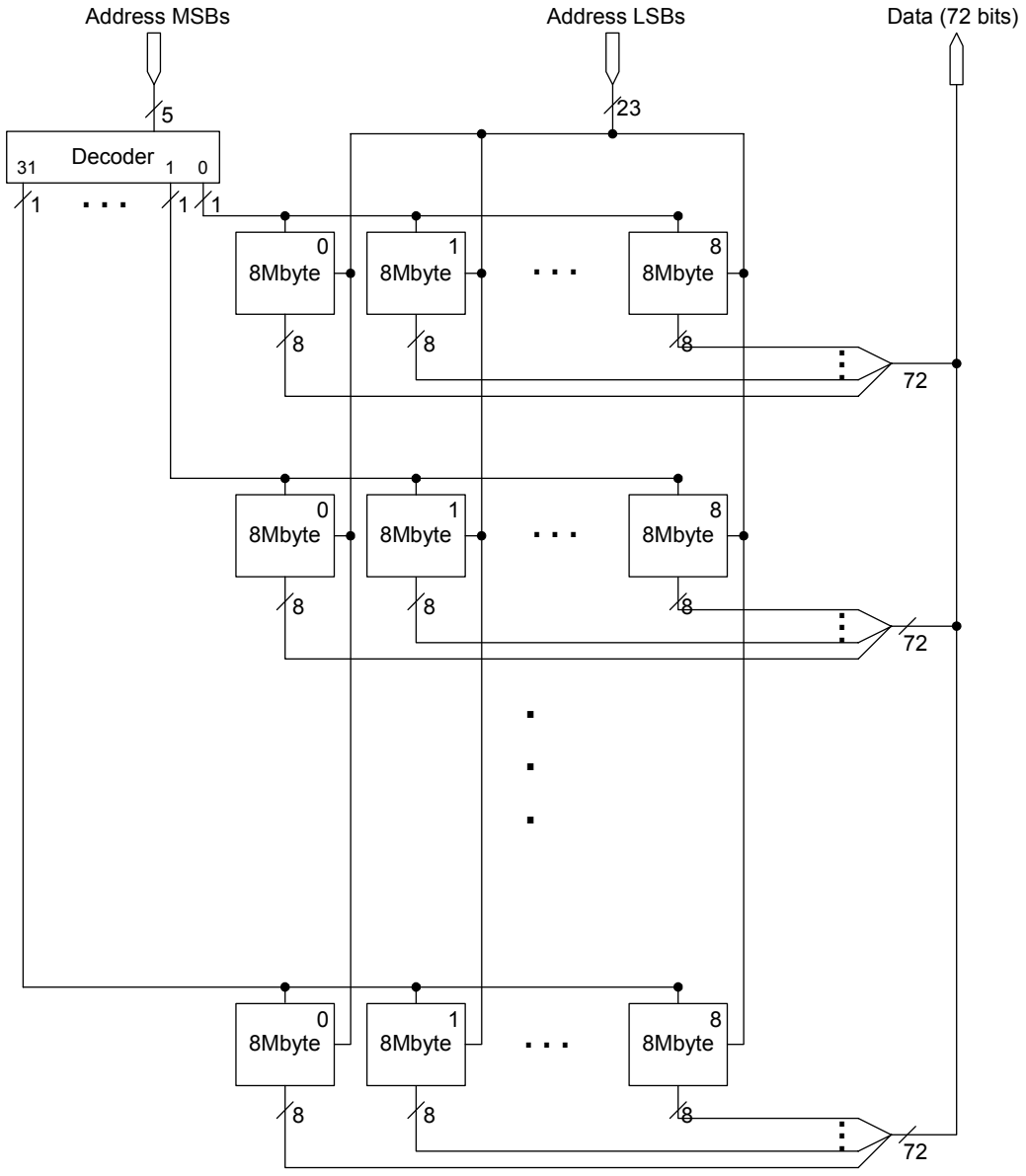
Αφού V=1, το αποτέλεσμά μας είναι λανθασμένο. Πράγματι το αποτέλεσμά μας είναι το -87_{10}, ενώ το αναμενόμενο αποτέλεσμα είναι $55_{10} + 114_{10} = 169_{10}$. Το ότι θα παίρναμε λανθασμένο αποτέλεσμα ήταν κάτι αναμενόμενο, αφού το 169_{10} δεν ανήκει στο διάστημα έγκυρων αναπαραστάσεων.

Οργάνωση Μνήμης

ΑΣΚΗΣΗ 44

Να σχεδιαστεί σύστημα κύριας μνήμης χωρητικότητας 2^{28} θέσεων με οργάνωση 64 δυαδικών ψηφίων ανά θέση μνήμης και ένα δυαδικό ψηφίο ισοτιμίας που θα αντιστοιχεί σε 8 δυαδικά ψηφία πληροφορίας. Έχετε στη διάθεσή σας ολοκληρωμένα κυκλώματα (chips) 8 Mbytes με οργάνωση 8 δυαδικών ψηφίων ανά θέση μνήμης.

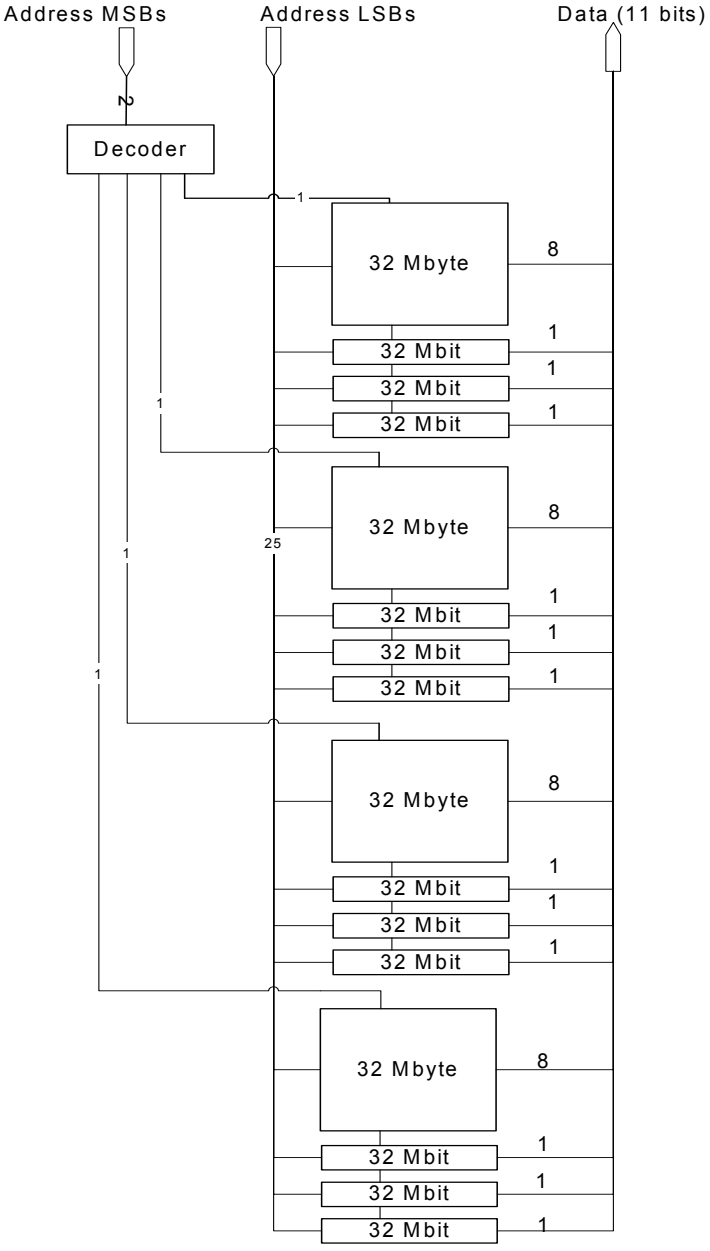
Λύση:
Για κάθε θέση μνήμης απαιτούνται 64 δυαδικά ψηφία συν 8 ψηφία ισοτιμίας, σύνολο 72 δυαδικά ψηφία. Άρα κάθε θέση μνήμης απαιτεί 9 bytes. Έτσι με μία δομή που αποτελείται από εννέα ολοκληρωμένα των 8Mbytes μπορούμε να έχουμε 2^{23} θέσεις x 72 bits. Αφού η ζητούμενη μνήμη απαιτεί 2^{28} θέσεις των 72 bits, πρέπει να χρησιμοποιήσουμε 2^5=32 επαναλήψεις αυτής της δομής και έναν αποκωδικοποιητή 5 σε 32 ως εξής:



ΑΣΚΗΣΗ 45

Να σχεδιαστεί σύστημα κύριας μνήμης με χωρητικότητα 128×2^{20} θέσεις με οργάνωση 10 δυαδικά ψηφία πληροφορίας ανά θέση μνήμης και το ψηφίο της ισοτιμίας. Έχετε στη διάθεσή σας ολοκληρωμένα κυκλώματα 32Mbytes με οργάνωση 8 δυαδικά ψηφία ανά θέση μνήμης και ολοκληρωμένα κυκλώματα 32 Mbits με οργάνωση 1 δυαδικό ψηφίο ανά θέση μνήμης.

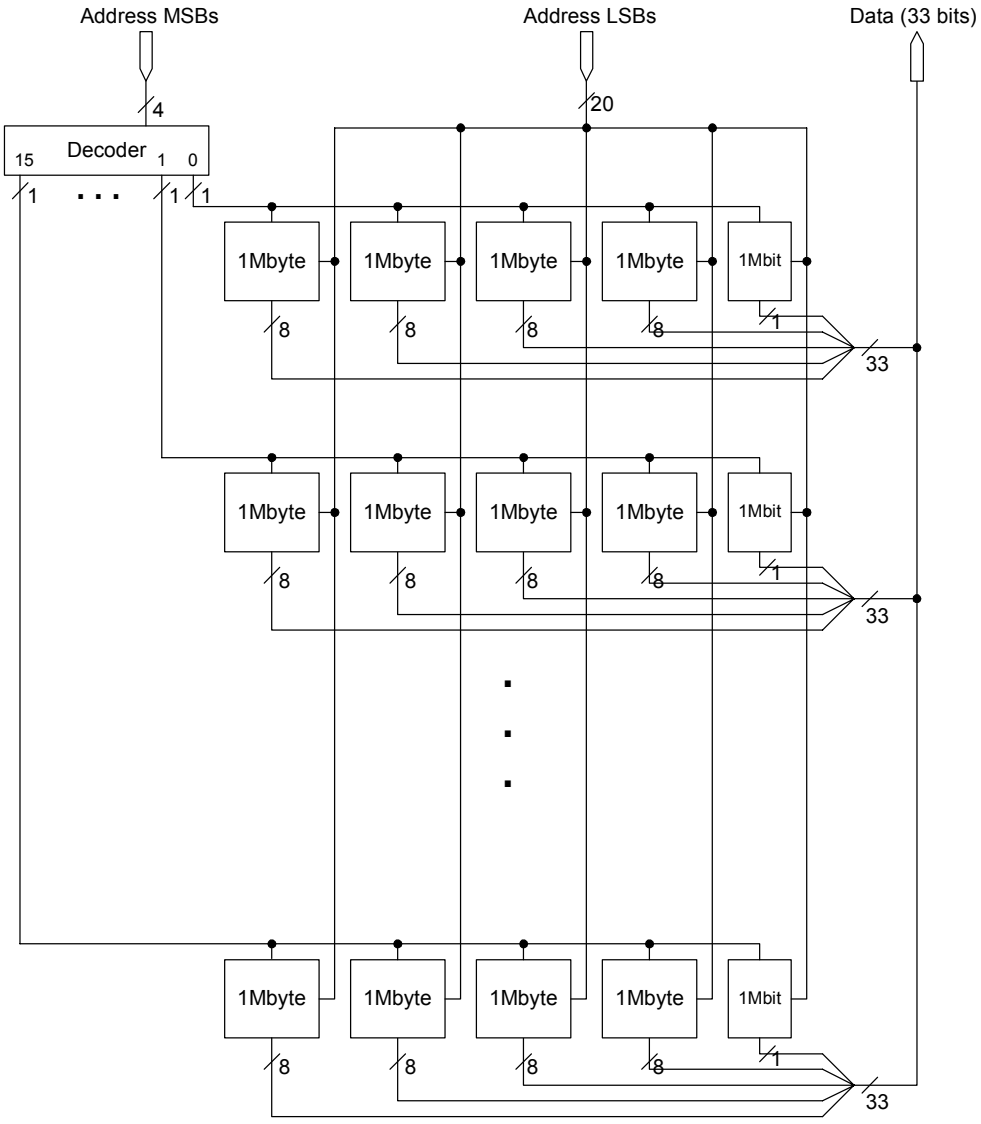
Λύση:
Για κάθε θέση μνήμης απαιτούνται 10 δυαδικά ψηφία συν το ψηφίο ισοτιμίας, σύνολο 11 δυαδικά ψηφία. Άρα κάθε θέση μνήμης απαιτεί 1 byte και 3 bits, γιατί $1 \times 8 + 3 = 10$.
Έτσι μπορεί να σχηματιστεί μια μνήμη 32×2^{20} θέσεων των 11 ψηφίων, με ένα ολοκληρωμένο των 32Mbyte και τρία ολοκληρωμένα των 32Mbit.
Για να σχηματιστούν οι ζητούμενες 128×2^{20} θέσεις απαιτούνται 4 επαναλήψεις αυτής της δομής και ένας αποκωδικοποιητής, διασυνδεδεμένα ως εξής:



ΑΣΚΗΣΗ 46

Να σχεδιαστεί σύστημα κύριας μνήμης χωρητικότητας 2^{24} θέσεων με οργάνωση 32 δυαδικών ψηφίων και ένα bit ισοτιμίας ανά θέση μνήμης. Έχετε στη διάθεσή σας ολοκληρωμένα κυκλώματα (chips) 1Mbyte με οργάνωση 8 δυαδικών ψηφίων ανά θέση μνήμης και ολοκληρωμένα κυκλώματα 1 Mbit με οργάνωση ενός δυαδικού ψηφίου ανά θέση μνήμης.

Λύση:
Για κάθε θέση μνήμης απαιτούνται 32 δυαδικά ψηφία συν το ψηφίο ισοτιμίας, σύνολο 33 δυαδικά ψηφία. Άρα κάθε θέση μνήμης απαιτεί 4 byte και 1 bit, γιατί $4 \times 8 + 1 = 33$.
Έτσι με μία δομή που αποτελείται από τέσσερα ολοκληρωμένα του 1Mbyte (2^{20} θέσεις x 8 bits) και ένα ολοκληρωμένο του 1Mbit (2^{20} θέσεις x 1 bit) μπορούμε να έχουμε 2^{20} θέσεις x 33 bits. Αφού η ζητούμενη μνήμη απαιτεί 2^{24} θέσεις των 33 bits, πρέπει να χρησιμοποιήσουμε 2^4=16 επαναλήψεις αυτής της δομής και έναν αποκωδικοποιητή 4 σε 16 ως εξής:



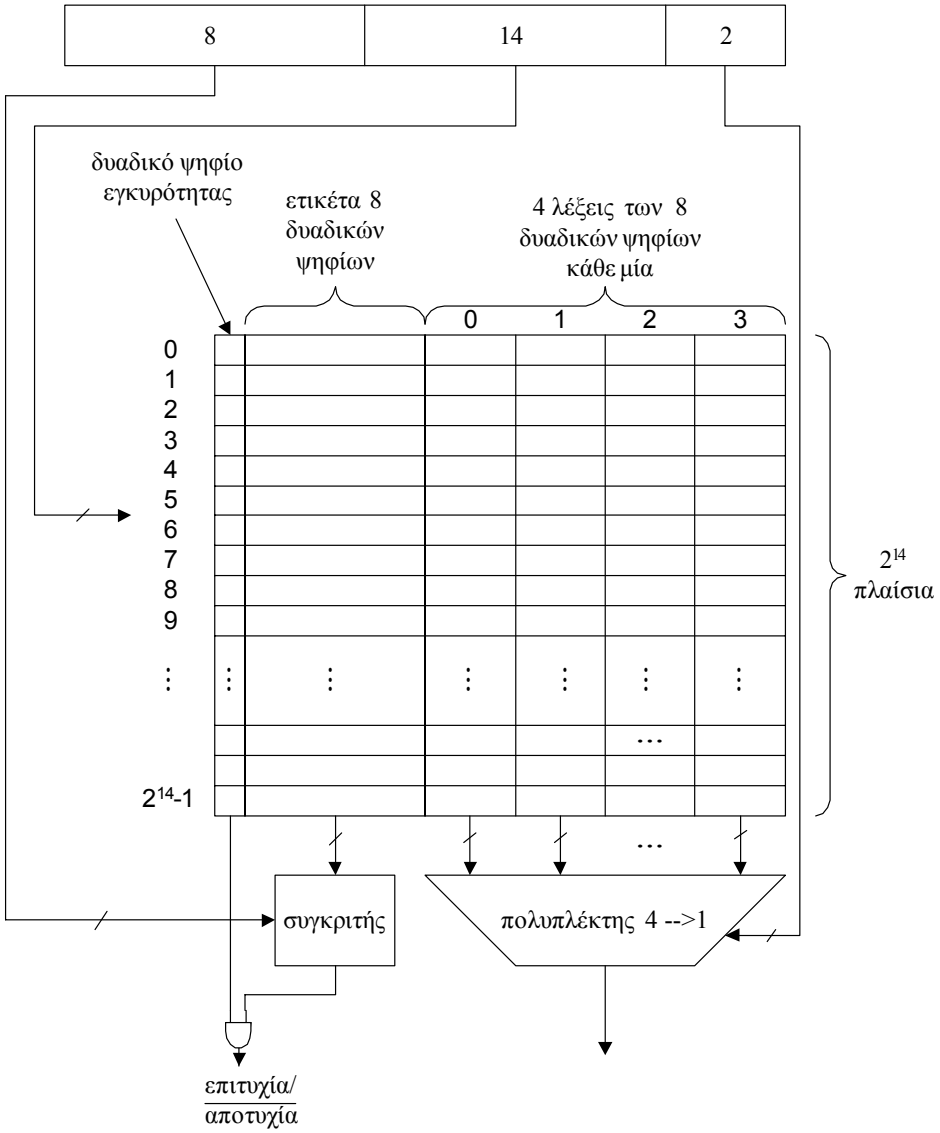
ΑΣΚΗΣΗ 47

Έχετε στη διάθεσή σας ολοκληρωμένα κυκλώματα μνήμης μεγέθους 8KBytes και 1KByte με οργάνωση 1 byte και 1 bit ανά θέση μνήμης αντίστοιχα, μία πύλη AND, μία πύλη NOT και οποιοδήποτε κύκλωμα συγκριτή και πολυπλέκτη χρειαστείτε. Να σχεδιάσετε κρυφή μνήμη άμεσης

οργάνωσης με χωρητικότητα 64 Κλέξεων και τέσσερις λέξεις των 8 δυαδικών ψηφίων η κάθε μία ανά πλαίσιο. Η διεύθυνση που παράγει ο επεξεργαστής είναι των 24 δυαδικών ψηφίων. Πρέπει να δώσετε οπωσδήποτε σχήμα.

Λύση :

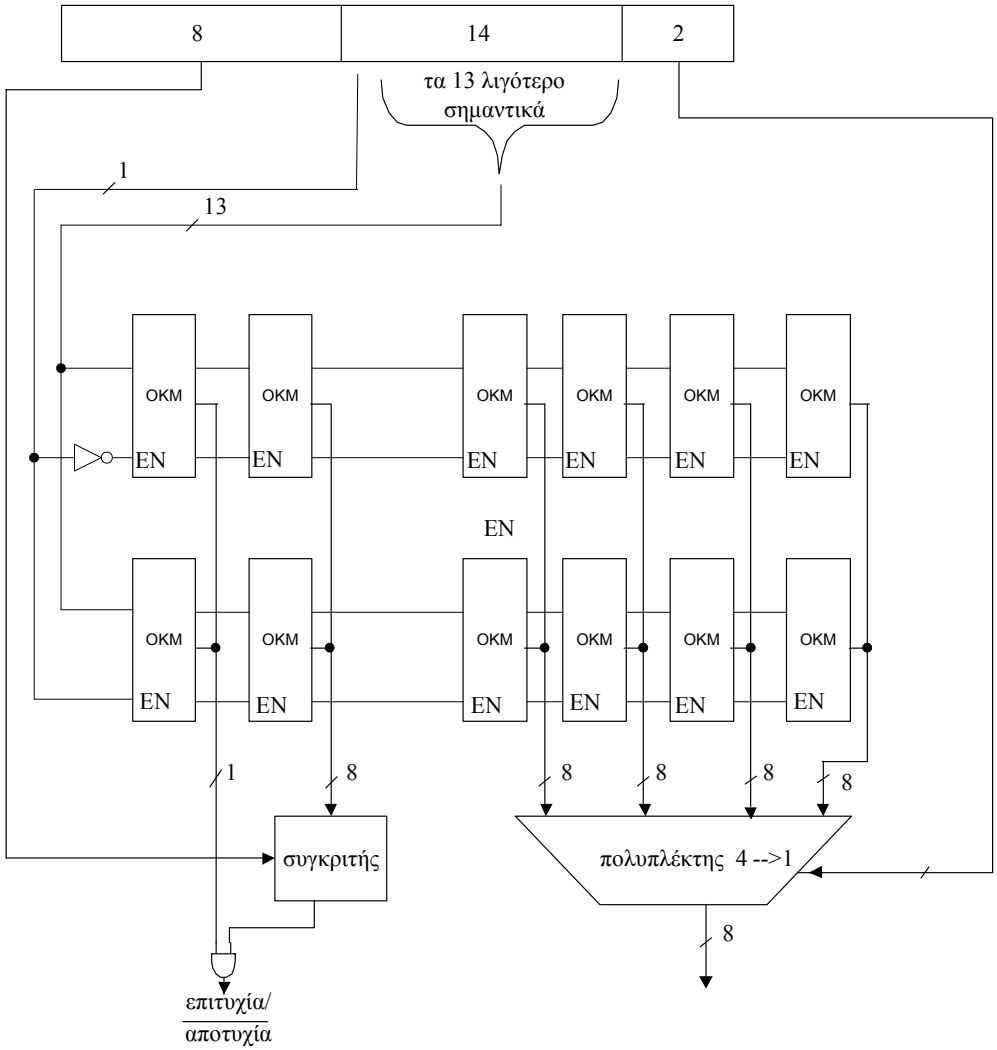
Αφού η χωρητικότητα της κρυφής μνήμης είναι 64 Κλέξεις και κάθε πλαίσιο είναι των τεσσάρων λέξεων η κρυφή μνήμη θα αποτελείται από 16K πλαίσια. Αφού κάθε πλαίσιο είναι των τεσσάρων λέξεων τα δύο λιγότερο σημαντικά δυαδικά ψηφία της διεύθυνσης που παράγει ο επεξεργαστής θα δηλώνουν την διεύθυνση της λέξης μέσα στο πλαίσιο. Επειδή η κρυφή μνήμη έχει 16K πλαίσια και είναι άμεσης οργάνωσης τα επόμενα 14 δυαδικά ψηφία της διεύθυνσης θα δηλώνουν την διεύθυνση του πλαισίου στην κρυφή μνήμη και τα υπόλοιπα 8 πιο σημαντικά δυαδικά ψηφία θα δηλώνουν την ετικέτα. Η λειτουργική εμφάνιση της μνήμης φαίνεται στο σχήμα 9.1.



Σχήμα 9.1.

Λαμβάνοντας υπόψη ότι έχουμε στη διάθεσή μας ολοκληρωμένα κυκλώματα μνήμης των 8KBytes και του 1KByte με οργάνωση 1 byte και 1 bit ανά θέση μνήμης αντίστοιχα συμπεραίνουμε ότι για να έχουμε ανά θέση της κρυφής μνήμης 5x8 + 1 δυαδικά ψηφία θα πρέπει να χρησιμοποιήσουμε 5 ολοκληρωμένα κυκλώματα μνήμης των 8KBytes και ένα ολοκληρωμένο κύκλωμα μνήμης του 1KByte που θα διευθυσιοδοτούνται ταυτόχρονα. Με αυτά τα ολοκληρωμένα

κυκλώματα υλοποιούμε κρυφή μνήμη των 8K πλαισίων (προσέξτε ότι το ολοκληρωμένο του 1Kbyte προσφέρει 8K θέσεις μνήμης του ενός bit η κάθε μία), επομένως για να υλοποιήσουμε κρυφή μνήμη 16K πλαισίων πρέπει να χρησιμοποιήσουμε μία ακόμη ομάδα από 5 ολοκληρωμένα κυκλώματα μνήμης των 8KBytes και ένα ολοκληρωμένο κύκλωμα μνήμης του 1KByte. Τα 13 λιγότερο σημαντικά δυαδικά ψηφία του πεδίου διεύθυνσης πλαισίου θα οδηγούν όλες τις εισόδους διευθύνσεων των ολοκληρωμένων μνήμης ενώ το πιο σημαντικό δυαδικό ψηφίο του πεδίου χρησιμοποιείται για ενεργοποίηση των ολοκληρωμένων μνήμης της πρώτης ομάδας ή της δεύτερης όπως φαίνεται στο σχήμα 9.2.



ΟΚΜ = Ολοκληρωμένο Κύκλωμα Μνήμης

Σχήμα 9.2.

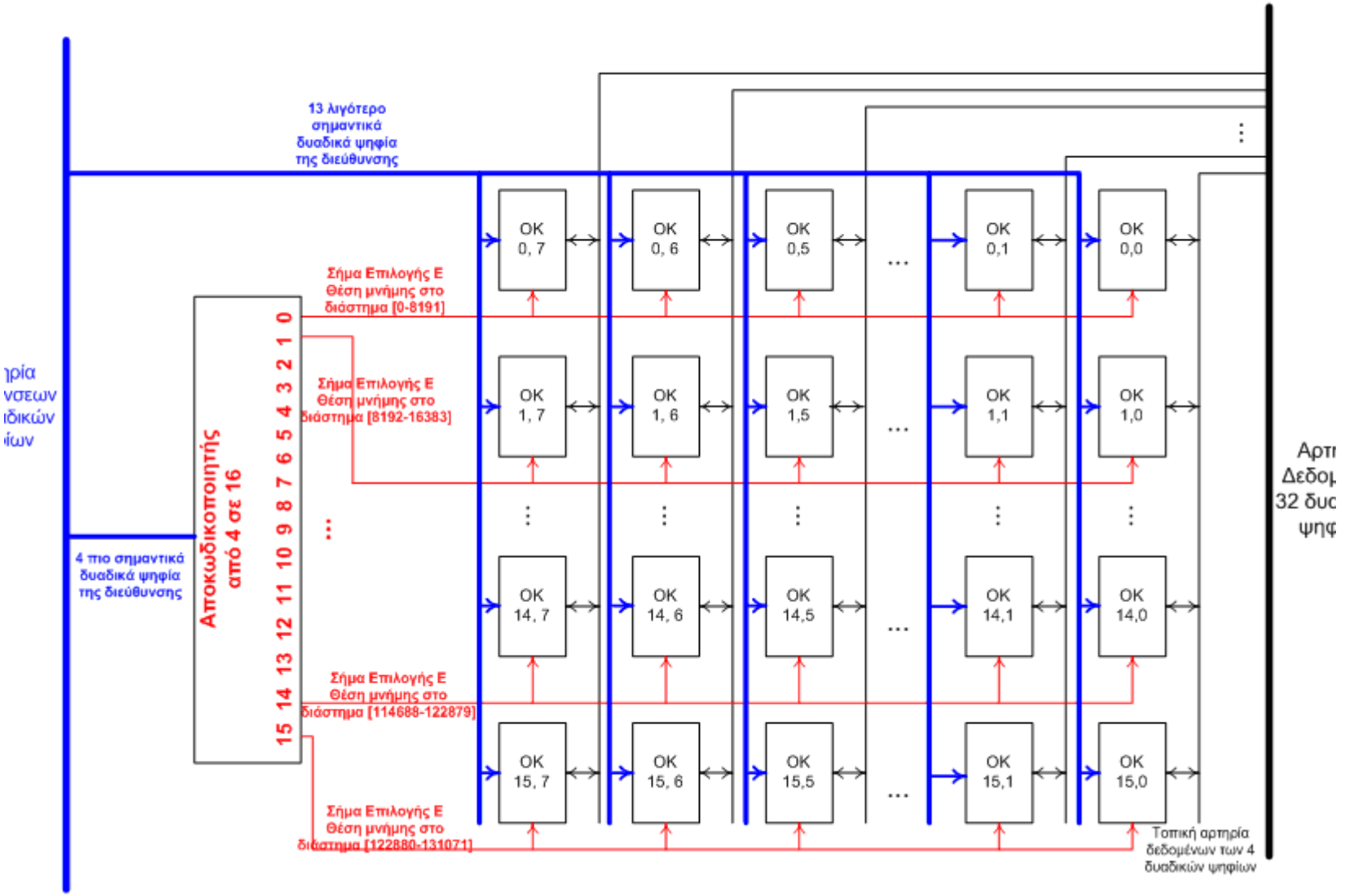
ΑΣΚΗΣΗ 48

Εχετε διαθέσιμα ολοκληρωμένα κυκλώματα μνήμης (ΟΚ) μεγέθους 4 KBytes, με οργάνωση 8192 θέσεων μνήμης των 4 δυαδικών ψηφίων ανά θέση μνήμης και σήμα επιλογής (Ε). Υποδείξτε πως μπορείτε διασυνδέοντας τέτοια ΟΚ να φτιάξετε ημιαγωγική μνήμη συνολικού μεγέθους 512 KBytes, με οργάνωση 131072 θέσεων των 32 δυαδικών ψηφίων ανά θέση μνήμης.

Λύση

- Σε κάθε ΟΚ υπάρχουν 8192 διαφορετικές θέσεις μνήμης. Για τη προσπέλαση κάποιας θέσης, θα πρέπει να χρησιμοποιηθεί τοπική αρτηρία διευθύνσεων εύρους $\lceil \log_2 8192 \rceil$ = 13 δυαδικών ψηφίων. Μέσω της τοπικής του αρτηρίας δεδομένων, κάθε ΟΚ διαχειρίζεται ποσότητα πληροφορίας 4 δυαδικών ψηφίων.
- Για τη κατασκευή της στοχευόμενης μνήμης απαιτούνται 512 KBytes / 4 (KBytes/ΟΚ) = 128 ΟΚ. Επειδή η στοχευόμενη μνήμη θα διαθέτει 131072 διαφορετικές θέσεις μνήμης, για τη προσπέλαση κάποιας από αυτές απαιτείται αρτηρία διευθύνσεων εύρους $\lceil \log_2 131072 \rceil$ = 17 δυαδικών ψηφίων.
- Η διασύνδεση των 128 ΟΚ προκύπτει ως ακολούθως :
  - Αφού κάθε θέση της στοχευόμενης μνήμης θα διαχειρίζεται 32 δυαδικά ψηφία, ενώ κάθε ΟΚ μπορεί να διαχειρίζεται μόνο 4, θα πρέπει να συνδεθούν παράλληλα 32 / 4 = 8 ΟΚ ώστε να επιτευχθεί το απαιτούμενο εύρος ανά θέση μνήμης.
  - Για τη δημιουργία των 131072 διαφορετικών θέσεων απαιτούνται 131072 / 8192 = 16 ΟΚ ώστε να επιτευχθεί το απαιτούμενο μήκος θέσεων μνήμης.

Αρα χρειάζεται μια οργάνωση πίνακα 16 γραμμών και 8 στηλών, όπως αυτή του σχήματος της επόμενης σελίδας. Τα 4 πιο σημαντικά ψηφία της διεύθυνσης των 17 δυαδικών ψηφίων αποτελούν είσοδο ενός αποκωδικοποιητή 4 σε 16. Κάθε έξοδός του επιλέγει (ενεργοποιεί) μια γραμμή των 8 ΟΚ και υποδεικνύει σε ποιό από τα 16 διαστήματα των 8192 διευθύνσεων ανήκει η διεύθυνση. Τα υπόλοιπα 13 δυαδικά ψηφία της διεύθυνσης οδηγούνται σε όλα τα ΟΚ και επιλέγουν μια θέση μνήμης σε κάθε ολοκληρωμένο. Κάθε στήλη ΟΚ διαχειρίζεται πληροφορία των 4 δυαδικών ψηφίων μέσω των τοπικών αρτηριών δεδομένων. Τα απαιτούμενα 32 δυαδικά ψηφία προκύπτουν από τη συνένωση και παράλληλη χρήση και των 8 τοπικών αρτηριών δεδομένων.


13 λιγότερο σημαντικά δυαδικά ψηφία της διεύθυνσης
4 πιο σημαντικά δυαδικά ψηφία της διεύθυνσης
Αποκωδικοποιητής από 4 σε 16
15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
Σήμα Επιλογής Ε
Θέση μνήμης στο διάστημα [0-8191]
Σήμα Επιλογής Ε
Θέση μνήμης στο διάστημα [8192-16383]
Σήμα Επιλογής Ε
Θέση μνήμης στο διάστημα [114688-122879]
Σήμα Επιλογής Ε
Θέση μνήμης στο διάστημα [122880-131071]
ΟΚ 0, 7
ΟΚ 0, 6
ΟΚ 0,5
ΟΚ 0,1
ΟΚ 0,0
ΟΚ 1, 7
ΟΚ 1, 6
ΟΚ 1,5
ΟΚ 1,1
ΟΚ 1,0
ΟΚ 14, 7
ΟΚ 14, 6
ΟΚ 14,5
ΟΚ 14,1
ΟΚ 14,0
ΟΚ 15, 7
ΟΚ 15, 6
ΟΚ 15,5
ΟΚ 15,1
ΟΚ 15,0
Τοπική αρτηρία δεδομένων των 4 δυαδικών ψηφίων

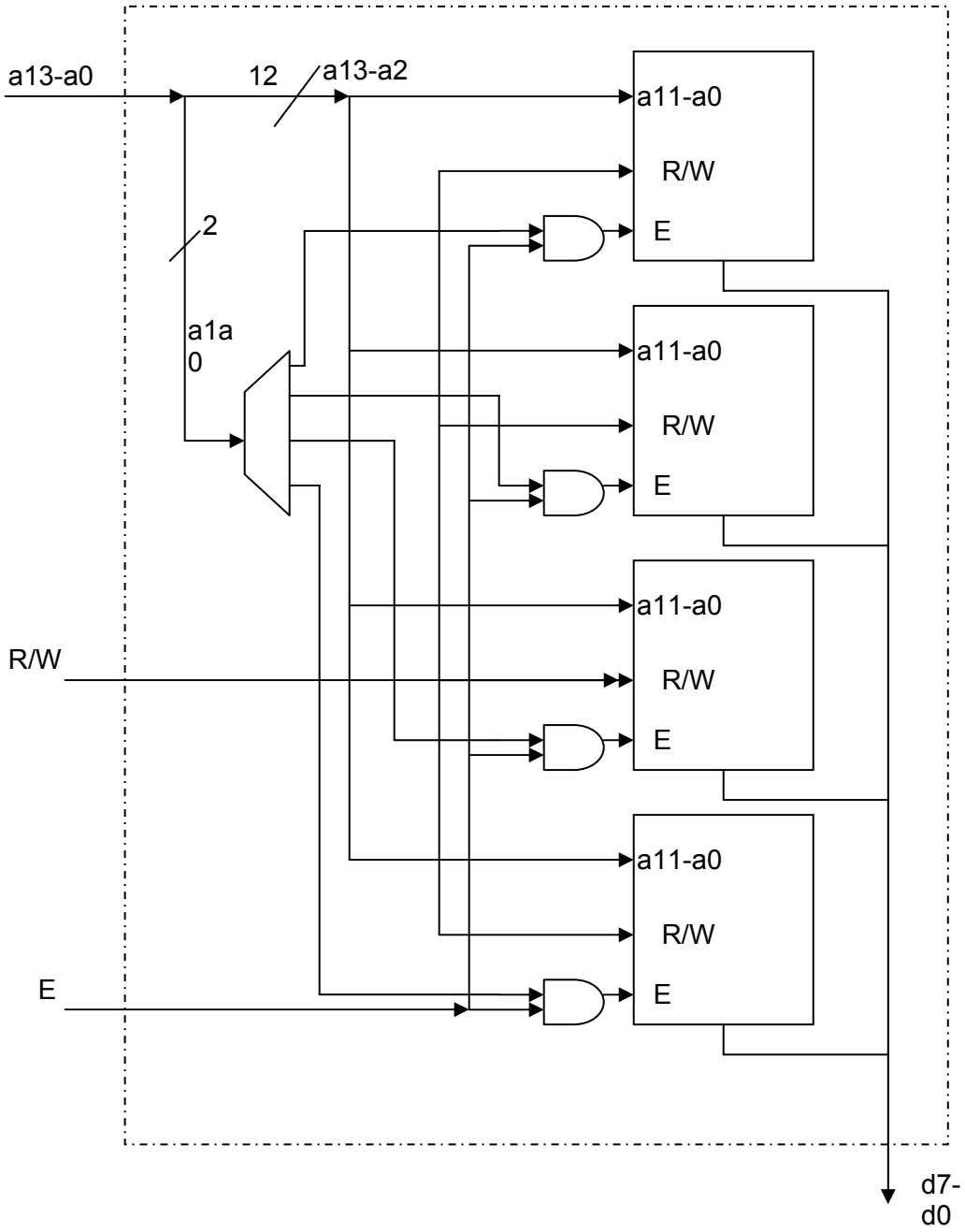
ΑΣΚΗΣΗ 49

Έστω πως έχουμε στη διάθεσή μας ολοκληρωμένα κυκλώματα μνήμης των 4K x 8 bit. Ο χρόνος προσπέλασης είναι 50ns και ο χρόνος κύκλου είναι 200ns δηλαδή πρέπει να περιμένουμε άλλα 150ns πριν να προσπελάσουμε το ίδιο ολοκληρωμένο. Σχεδιάστε σύστημα μνήμης 16K x 8 έτσι ώστε να ελαχιστοποιείται ο συνολικός χρόνος που θα χρειαστεί για να ολοκληρωθεί η παρακάτω ακολουθία αναγνώσεων:

0, 1, 2, 3, …, 16383

Λύση

Για να συνθέσουμε την μνήμη των 16Kx8bit χρειαζόμαστε 4 ολοκληρωμένα των 4Kx8bit. Στην ιδανική περίπτωση θα θέλαμε να ξεκινήσουμε την επόμενη προσπέλαση αμέσως μόλις ολοκληρωθεί η προηγούμενή της έτσι ώστε κάθε προσπέλαση να ολοκληρώνεται σε 50ns. Δηλαδή, αφού προσπελάσουμε τη διεύθυνση 0 και πάρουμε τα αποτελέσματα μετά από 50ns θα θέλαμε να μπορούμε να προσπελάσουμε τη διεύθυνση 1. Όμως το ολοκληρωμένο που περιέχει τη διεύθυνση 0 δε μπορούμε να το προσπελάσουμε για τα επόμενα 150ns (ή για 3x50ns). Οπότε η διεύθυνση 1 πρέπει να αντιστοιχηθεί σε διαφορετικό ολοκληρωμένο. Για το ίδιο λόγο σε διαφορετικά ολοκληρωμένα πρέπει να αντιστοιχήσουμε τις διευθύνσεις 2 και 3. Γενικότερα, εφόσον η ακολουθία προσπέλασης είναι γραμμική μπορούμε να αντιστοιχήσουμε τις διευθύνσεις βάσει του υπόλοιπου που προκύπτει όταν διαιρέσουμε τη διεύθυνση με το 4 (πλήθος των ολοκληρωμένων). Δηλαδή, αυτές που είναι διαιρέσιμες με το 4 (πλήθος ολοκληρωμένων) τις αντιστοιχούμε στο πρώτο ολοκληρωμένο, αυτές που όταν διαιρεθούν με το 4 προκύπτει υπόλοιπο 1 στο δεύτερο ολοκληρωμένο, κ.ο.κ. Δηλαδή έχουμε κατανομή διευθύνσεων σύμφωνα με την 4-δρόμων χαμηλής τάξης διαφύλλωση μνήμης. Αυτό μπορούμε να το επιτύχουμε χρησιμοποιώντας τα bit 1 και 0 της διεύθυνσης μέσω ένος αποκωδικοποιητή 2-σε-4 για να διαλέξουμε ένα από τα 4 ολοκληρωμένα και στη συνέχεια τα bit 2 ως και 13 για να διαλέξουμε τη γραμμή μέσα σε αυτό το ολοκληρωμένο. Το σχετικό κύκλωμα φαίνεται στο παρακάτω διάγραμμα:


a13-a0
12
a13-a2
a11-a0
R/W
E
2
a1a
0
a11-a0
R/W
E
a11-a0
R/W
E
R/W
a11-a0
R/W
E
E
d7-
d0

Κρυφή μνήμη

ΑΣΚΗΣΗ 50

Θεωρείστε ότι η ΚΜΕ ενός υπολογιστή, που παράγει διευθύνσεις των 8 δυαδικών ψηφίων, παράγει την επόμενη ακολουθία διευθύνσεων: 5, 140, 20, 29, 3, 8, 65, 141, 83, 152, 23, 155, 22, 71, 137, 80, 25, 153. Προτείνετε την φθηνότερη και γρηγορότερη οργάνωση κρυφής μνήμης που δίνει το μεγαλύτερο ποσοστό επιτυχίας λαμβάνοντας υπόψη ότι:

α. Η κρυφή μνήμη έχει χωρητικότητα 64 λέξεων.

β. Κάθε πλαίσιο της κρυφής μνήμης αποτελείται από 8 λέξεις.

Αποδείξτε ότι η οργάνωση που προτείνετε έχει πράγματι τα ζητούμενα χαρακτηριστικά. Για κάθε μία οργάνωση που θα εξετάσετε να γράψετε δίπλα σε κάθε διεύθυνση εάν έχουμε "hit" ή "miss" και στη συνέχεια να δώσετε το τελικό περιεχόμενο της κρυφής μνήμης. Στο ξεκίνημα η κρυφή μνήμη δεν έχει έγκυρα περιεχόμενα. Η στρατηγική αντικατάστασης πλαισίου, όπου χρειάζεται είναι η FIFO.

Λύση:

Αρχικά θα εξετάσουμε την φθηνότερη και γρηγορότερη οργάνωση που είναι η άμεση. Εφόσον η κρυφή μνήμη αποτελείται από 64 λέξεις και κάθε πλαίσιο αποτελείται από 8 λέξεις, υπάρχουν συνολικά 8 πλαίσια. Συνεπώς μία διεύθυνση των 8 ψηφίων απαιτεί τρία ψηφία για τον καθορισμό της θέσης λέξης μέσα στο πλαίσιο (μ=3), τρία ψηφία για τον καθορισμό του πλαισίου (κ=3) και 8 – 3 – 3 = 2 ψηφία για την ετικέτα.

Διεύθυνση μνήμης	Διεύθυνση σε δυαδική μορφή (ετικέτα – πλαίσιο – λέξη)		Π(διευθύνσεις μνήμης) => πλαίσιο με διεύθυνση*
5	00-000-101	Miss	Π(0…7) => 0
140	10-001-100	Miss	Π(136…143) => 1
20	00-010-100	Miss	Π(16…23) => 2
29	00-011-101	Miss	Π(24…31) => 3
3	00-000-011	Hit	
8	00-001-000	Miss	Π(8…15) => 1
65	01-000-001	Miss	Π(64…71) => 0
141	10-001-101	Miss	Π(136…143) => 1
83	01-010-011	Miss	Π(80…87) => 2
152	10-011-000	Miss	Π(152…159) => 3
23	00-010-111	Miss	Π(16…23) => 2
155	10-011-011	Hit	
22	00-010-110	Hit	
71	01-000-111	Hit	
137	10-001-001	Hit	
80	01-010-000	Miss	Π(80…87) => 2
25	00-011-001	Miss	Π(24…31) => 3
153	10-011-001	Miss	Π(152…159) => 3

* Π(διευθύνσεις μνήμης) => πλαίσιο με διεύθυνση : Ο συμβολισμός αυτός σημαίνει ότι η πληροφορία που είναι αποθηκευμένη στις θέσεις μνήμης που αναφέρονται μέσα στις παρενθέσεις θα μεταφερθεί στο πλαίσιο με τη διεύθυνση που δηλώνεται.

Μετά από τη συγκεκριμένη ακολουθία διευθύνσεων, τα περιεχόμενα της κρυφής μνήμης είναι τα ακόλουθα:

Διεύθυνση πλαισίου	Διευθύνσεις θέσεων της κύριας μνήμης τα περιεχόμενα των οποίων βρίσκονται στο πλαίσιο
000	Π(64 … 71)
001	Π(136 … 143)
010	Π(80 … 87)
011	Π(152 … 159)
100	-
101	-
110	-
111	-

Η παύλα σημαίνει ότι το συγκεκριμένο πλαίσιο δεν έχει έγκυρα περιεχόμενα. Παρατηρούμε ότι από τις 18 αναφορές οι 13 ήταν αποτυχημένες και οι υπόλοιπες επιτυχημένες. Άρα το ποσοστό επιτυχίας ήταν 5/18 = 27,78%.

Η αμέσως επόμενη οργάνωση σε κόστος και ταχύτητα είναι η οργάνωση 2-τρόπων συνόλου συσχέτισης. Σε αυτή την περίπτωση η κρυφή μνήμη αποτελείται από 4 σύνολα και 2 πλαίσια ανά σύνολο. Άρα λ=2 bits γιατί έχουμε τέσσερα σύνολα, μ=3, γιατί έχουμε 8 λέξεις ανά πλαίσιο. Τα υπόλοιπα τρία bits είναι η ετικέτα.

Διεύθυνση μνήμης	Διεύθυνση σε δυαδική μορφή (ετικέτα – σύνολο - λέξη)		Π(διευθύνσεις μνήμης) => πλαίσιο 0 του συνόλου με διεύθυνση	Π(διευθύνσεις μνήμης) => πλαίσιο 1 του συνόλου με διεύθυνση
5	000-00-101	Miss	Π(0…7) => 0	
140	100-01-100	Miss	Π(136…143) => 1	
20	000-10-100	Miss	Π(16…23) => 2	
29	000-11-101	Miss	Π(24…31) => 3	
3	000-00-011	Hit		
8	000-01-000	Miss		Π(8…15) => 1
65	010-00-001	Miss		Π(64…71) => 0
141	100-01-101	Hit		
83	010-10-011	Miss		Π(80…87) => 2
152	100-11-000	Miss		Π(152…159) => 3
23	000-10-111	Hit		
155	100-11-011	Hit		
22	000-10-110	Hit		
71	010-00-111	Hit		
137	100-01-001	Hit		
80	010-10-000	Hit		
25	000-11-001	Hit		
153	100-11-001	Hit		

Τα τελικά περιεχόμενα της κρυφής μνήμης είναι τα ακόλουθα:

Σύνολο	Διευθύνσεις θέσεων της κύριας μνήμης τα περιεχόμενα των οποίων βρίσκονται στο Πλαίσιο	Διευθύνσεις θέσεων της κύριας μνήμης τα περιεχόμενα των οποίων βρίσκονται στο Πλαίσιο

	0	1
00	Π(0…7)	Π(64…71)
01	Π(136…143)	Π(8…15)
10	Π(16…23)	Π(80…87)
11	Π(24…31)	Π(152…159)

Παρατηρούμε ότι από τις 18 αναφορές οι 8 ήταν αποτυχημένες και οι υπόλοιπες επιτυχημένες. Άρα το ποσοστό επιτυχίας ήταν 10/18 = 55,55%. Επίσης παρατηρούμε ότι όλες οι αποτυχημένες αναφορές οφείλονται στην αρχική μετακίνηση των αντίστοιχων μπλοκ της κύριας μνήμης στην κρυφή μνήμη.

Λαμβάνοντας υπόψη ότι:
α. Τα μπλοκ της κύριας μνήμης στα οποία ανήκουν οι ζητούμενες διευθύνσεις είναι τα ίδια για κάθε οργάνωση, αφού το μέγεθος των πλαισίων της κρυφής μνήμης είναι σταθερό.
β. Για κάθε διεύθυνση το αντίστοιχο μπλοκ θα δημιουργήσει τουλάχιστον μία αποτυχημένη αναφορά αφού δεν θα βρίσκεται αρχικά στην κρυφή μνήμη.
γ. Οι ζητούμενες διευθύνσεις αντιστοιχούν σε 8 μπλοκ τα οποία χωρούν ακριβώς στα 8 πλαίσια της κρυφής μνήμης.
Άρα συμπεραίνουμε ότι οι ελάχιστες αποτυχημένες αναφορές είναι 8 και άρα το μέγιστο ποσοστό επιτυχίας που μπορούμε να έχουμε είναι 55,55%. Άρα η οργάνωση 2-τρόπων συνόλου συσχέτισης πετυχαίνει το μεγαλύτερο δυνατό ποσοστό επιτυχίας οπότε:

Η άμεση οργάνωση απορρίπτεται καθώς έχει μικρότερο ποσοστό επιτυχίας.
Κάθε άλλη οργάνωση απορρίπτεται καθώς έχει ίσο (ή μικρότερο) ποσοστό επιτυχίας από την οργάνωση 2-τρόπων συνόλου συσχέτισης, αλλά μεγαλύτερο κόστος και μικρότερη ταχύτητα από αυτήν.

Άρα η ζητούμενη οργάνωση είναι η οργάνωση 2-τρόπων συνόλου συσχέτισης.

ΑΣΚΗΣΗ 51

Θεωρείστε ότι η ΚΜΕ ενός υπολογιστή παράγει την επόμενη ακολουθία διευθύνσεων: 1, 4, 8, 5, 20, 17, 19, 50, 9, 11, 4, 43, 5, 6, 9, 17. Για κάθε μία από τις επόμενες περιπτώσεις να γράψετε δίπλα σε κάθε διεύθυνση εάν έχουμε “hit” ή “miss” και στη συνέχεια να δώσετε το τελικό περιεχόμενο της κρυφής μνήμης. Στο ξεκίνημα η κρυφή μνήμη είναι άδεια.
α. Η κρυφή μνήμη είναι άμεσης οργάνωσης με χωρητικότητα 16 λέξεων και 4 λέξεις ανά πλαίσιο.
β. Η κρυφή μνήμη έχει οργάνωση 2-τρόπων συνόλου συσχέτισης, χωρητικότητα 16 λέξεων και δύο λέξεις ανά πλαίσιο. Υποθέστε ότι χρησιμοποιείται η “LRU” στρατηγική απελευθέρωσης πλαισίων της κρυφής μνήμης.
γ. Η κρυφή μνήμη έχει οργάνωση πλήρους συσχέτισης, χωρητικότητα 16 λέξεων και τέσσερις λέξεις ανά πλαίσιο. Υποθέστε ότι χρησιμοποιείται η “LRU” στρατηγική απελευθέρωσης πλαισίων της κρυφής μνήμης.

Λύση:
α) Η κρυφή μνήμη είναι άμεσης οργάνωσης 16 λέξεων και έχει 4 λέξεις ανά πλαίσιο. Άρα περιλαμβάνει τέσσερα πλαίσια. Συνεπώς μία διεύθυνση των 6 ψηφίων απαιτεί δύο ψηφία για τον καθορισμό της θέσης λέξης μέσα στο πλαίσιο (μ=2), δύο ψηφία για τον καθορισμό του πλαισίου (κ=2) και 6 – 2 – 2 = 2 ψηφία για την ετικέτα, η οποία είναι και η διεύθυνση του μπλοκ στην κύρια μνήμη.

Διεύθυνση μνήμης	Διεύθυνση σε δυαδική μορφή		Π(διευθύνσεις μνήμης) => πλαίσιο με διεύθυνση*

	(ετικέτα – πλαίσιο - λέξη)		
1	00 – 00 – 01	Miss	Π(0, 1 ,2 ,3) => 0
4	00 – 01 – 00	Miss	Π(4 , 5, 6, 7) => 1
8	00 – 10 – 00	Miss	Π(8, 9, 10, 11) => 2
5	00 – 01 – 01	Hit	
20	01 – 01 – 00	Miss	Π(20,21,22,23) => 1
17	01 – 00 – 01	Miss	Π(16,17,18,19) => 0
19	01 – 00 – 11	Hit	
50	11 – 00 – 10	Miss	Π(48,49,50,51) => 0
9	00 – 10 – 01	Hit	
11	00 – 10 – 11	Hit	
4	00 – 01 – 00	Miss	Π(4, 5, 6, 7) => 1
43	10 – 10 – 11	Miss	Π(40,41,42,43) => 2
5	00 – 01 – 01	Hit	
6	00 – 01 – 10	Hit	
9	00 – 10 – 01	Miss	Π(8,9,10,11) => 2
17	01 – 00 – 01	Miss	Π(16,17,18,19) => 0

* Π(διευθύνσεις μνήμης) => πλαίσιο με διεύθυνση : Ο συμβολισμός αυτός σημαίνει ότι η πληροφορία που είναι αποθηκευμένη στις θέσεις μνήμης που αναφέρονται μέσα στις παρενθέσεις θα μεταφερθεί στο πλαίσιο με τη διεύθυνση που δηλώνεται.

Μετά από την συγκεκριμένη ακολουθία διευθύνσεων, τα περιεχόμενα της κρυφής μνήμης είναι τα ακόλουθα:

Διεύθυνση πλαισίου	Διευθύνσεις θέσεων της κύριας μνήμης τα περιεχόμενα των οποίων βρίσκονται στο πλαίσιο
00	Π(16,17,18,19)
01	Π(4, 5, 6, 7)
10	Π(8, 9, 10, 11)
11	-

(Η παύλα σημαίνει ότι το συγκεκριμένο πλαίσιο δεν έχει έγκυρα περιεχόμενα.)

β) Η κρυφή μνήμη έχει οργάνωση 2-τρόπων συνόλου συσχέτισης, χωρητικότητα 16 λέξεων και δύο λέξεις ανά πλαίσιο. Άρα κάθε σύνολο έχει δύο πλαίσια των δύο λέξεων, συνεπώς η κρυφή μνήμη περιλαμβάνει τέσσερα σύνολα.

Άρα λ=2 bits γιατί έχουμε τέσσερα σύνολα, μ=1, γιατί έχουμε δύο λέξεις ανά πλαίσιο. Τα υπόλοιπα τρία bits είναι η ετικέτα.

Διεύθυνση μνήμης	Διεύθυνση σε δυαδική μορφή (ετικέτα – σύνολο - λέξη)		Π(διευθύνσεις μνήμης) => πλαίσιο 0 του συνόλου με διεύθυνση	Π(διευθύνσεις μνήμης) => πλαίσιο 1 του συνόλου με διεύθυνση
1	000 – 00 – 1	Miss	Π(0,1) => 0	
4	000 – 10 – 0	Miss	Π(4,5) => 2	

8	001 – 00 – 0	Miss		Π(8,9) => 0
5	000 – 10 – 1	Hit		
20	010 – 10 – 0	Miss		Π(20,21) => 2
17	010 – 00 – 1	Miss	Π(16,17) => 0 (LRU)	
19	010 – 01 – 1	Miss	Π(18,19) => 1	
50	110 – 01 – 0	Miss		Π(50,51) => 1
9	001 – 00 – 1	Hit		
11	001 – 01 – 1	Miss	Π(10,11) => 1 (LRU)	
4	000 – 10 – 0	Hit		
43	101 – 01 – 1	Miss		Π(42,43) => 1 (LRU)
5	000 – 10 – 1	Hit		
6	000 – 11 – 0	Miss	Π(6, 7) => 3	
9	001 – 00 – 1	Hit		
17	010 – 00 – 1	Hit		

Τα τελικά περιεχόμενα της κρυφής μνήμης είναι τα ακόλουθα:

Σύνολο	Διευθύνσεις θέσεων της κύριας μνήμης τα περιεχόμενα των οποίων βρίσκονται στο Πλαίσιο 0	Διευθύνσεις θέσεων της κύριας μνήμης τα περιεχόμενα των οποίων βρίσκονται στο Πλαίσιο 1
00	Π(16, 17)	Π(8, 9)
01	Π(10, 11	Π(42, 43)
10	Π(4, 5)	Π(20, 21)
11	Π(6 ,7)	-

γ) Η κρυφή μνήμη έχει οργάνωση πλήρους συσχέτισης, χωρητικότητα 16 λέξεων και τέσσερις λέξεις ανά πλαίσιο. Άρα υπάρχουν τέσσερα πλαίσια των τεσσάρων λέξεων. Από τα έξι ψηφία της διεύθυνσης απαιτούνται δύο δυαδικά ψηφία για να καταδείξουν τη θέση της λέξης μέσα σε ένα πλαίσιο και τέσσερα δυαδικά ψηφία ως ετικέτα, η οποία δηλώνει τη διεύθυνση του μπλοκ που έχει αποθηκευτεί σε συγκεκριμένο πλαίσιο.

Διεύθυνση μνήμης	Διεύθυνση σε δυαδική μορφή (ετικέτα –λέξη)		Π(διευθύνσεις μνήμης) => πλαίσιο με διεύθυνση και (ετικέτα πλαισίου)
1	0000 – 01	Miss	Π(0,1,2,3) => 0 (0000)
4	0001 – 00	Miss	Π(4,5,6,7) => 1 (0001)
8	0010 – 00	Miss	Π(8,9,10,11) => 2 (0010)
5	0001 – 01	Hit	
20	0101 – 00	Miss	Π(20,21,22,23) => 3 (0101)
17	0100 – 01	Miss	Π(16,17,18,19) => 0(0100) LRU
19	0100 – 11	Hit	
50	1100 – 10	Miss	Π(48,49,50,51) => 2 (1100) LRU
9	0010 – 01	Miss	Π(8,9,10,11) => 1 (0010) LRU
11	0010 – 11	Hit	
4	0001 – 00	Miss	Π(4,5,6,7) => 3 (0001) LRU
43	1010 – 11	Miss	Π(40,41,42,43) => 0 (1010) LRU
5	0001 – 01	Hit	
6	0001 – 10	Hit	
9	0010 – 01	Hit	
17	0100 – 01	Miss	Π16,17,18,19) => 2 (0100) LRU

Τα τελικά περιεχόμενα της κρυφής μνήμης είναι τα ακόλουθα.

Διεύθυνση πλαισίου	Διευθύνσεις θέσεων της κύριας μνήμης τα περιεχόμενα των οποίων βρίσκονται στο συγκεκριμένο πλαίσιο
00	40,41,42,43
01	8,9,10,11
10	16,17,18,19
11	4,5,6,7

ΑΣΚΗΣΗ 52

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 "hit" ή "miss" και στη συνέχεια να δώσετε το τελικό περιεχόμενο της κρυφής μνήμης. Στο ξεκίνημα η κρυφή μνήμη δεν έχει έγκυρα περιεχόμενα.
α. Η κρυφή μνήμη είναι άμεσης οργάνωσης 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 "FIFO" στρατηγική απελευθέρωσης πλαισίων της κρυφής μνήμης.
γ. Η κρυφή μνήμη έχει οργάνωση πλήρους συσχέτισης, χωρητικότητα 32 λέξεων και 4 λέξεις ανά πλαίσιο. Υποθέστε ότι χρησιμοποιείται η "LRU" στρατηγική απελευθέρωσης πλαισίων της κρυφής μνήμης.

Λύση:
Ερώτημα (α)
Η κρυφή μνήμη είναι άμεσης οργάνωσης 32 λέξεων και έχει 4 λέξεις ανά πλαίσιο. Άρα περιλαμβάνει 8 πλαίσια των τεσσάρων λέξεων. Συνεπώς μία διεύθυνση των 7 ψηφίων απαιτεί δύο ψηφία για τον καθορισμό της θέσης λέξης μέσα στο πλαίσιο (μ=2), τρία ψηφία για τον καθορισμό του πλαισίου (κ=3) και 7 – 3 – 2 = 2 ψηφία για την ετικέτα.

Διεύθυνση μνήμης	Διεύθυνση σε δυαδική μορφή (ετικέτα – πλαίσιο – λέξη)		Π(διευθύνσεις μνήμης) => πλαίσιο με διεύθυνση*
33	01-000-01	Miss	Π(32, 33, 34, 35) => 0
17	00-100-01	Miss	Π(16, 17, 18, 19) => 4
30	00-111-10	Miss	Π(28, 29, 30, 31) => 7
47	01-011-11	Miss	Π(44, 45, 46, 47) => 3
50	01-100-10	Miss	Π(48, 49, 50, 51) => 4
35	01-000-11	Hit	
6	00-001-10	Miss	Π(4, 5, 6, 7) => 1
26	00-110-10	Miss	Π(24, 25, 26, 27) => 6
50	01-100-10	Hit	
42	01-010-10	Miss	Π(40, 41, 42, 43) => 2

* Π(διευθύνσεις μνήμης) => πλαίσιο με διεύθυνση : Ο συμβολισμός αυτός σημαίνει ότι η πληροφορία που είναι αποθηκευμένη στις θέσεις μνήμης που αναφέρονται μέσα στις παρενθέσεις θα μεταφερθεί στο πλαίσιο με τη διεύθυνση που δηλώνεται.

58	01-110-10	Miss	Π(56, 57, 58, 59) => 6
50	01-100-10	Hit	
13	00-011-01	Miss	Π(12, 13, 14, 15) => 3
22	00-101-10	Miss	Π(20, 21, 22, 23) => 5
15	00-011-11	Hit	
4	00-001-00	Hit	
0	00-000-00	Miss	Π(0, 1, 2, 3) => 0
70	10-001-10	Miss	Π(68, 69, 70, 71) => 1

Μετά από τη συγκεκριμένη ακολουθία διευθύνσεων, τα περιεχόμενα της κρυφής μνήμης είναι τα ακόλουθα:

Διεύθυνση πλαισίου	Διευθύνσεις θέσεων της κύριας μνήμης τα περιεχόμενα των οποίων βρίσκονται στο πλαίσιο
000	Π(0, 1, 2, 3)
001	Π(68, 69, 70, 71)
010	Π(40, 41, 42, 43)
011	Π(12, 13, 14, 15)
100	Π(48, 49, 50, 51)
101	Π(20, 21, 22, 23)
110	Π(56, 57, 58, 59)
111	Π(28, 29, 30, 31)

Ερώτημα (β)
Η κρυφή μνήμη έχει οργάνωση 2-τρόπων συνόλου συσχέτισης, χωρητικότητα 32 λέξεων και δύο λέξεις ανά πλαίσιο. Άρα κάθε σύνολο έχει δύο πλαίσια των δύο λέξεων, συνεπώς η κρυφή μνήμη περιλαμβάνει οκτώ σύνολα.

Άρα λ=3 bits γιατί έχουμε οκτώ σύνολα, μ=1, γιατί έχουμε δύο λέξεις ανά πλαίσιο. Τα υπόλοιπα τρία bits είναι η ετικέτα.

Διεύθυνση μνήμης	Διεύθυνση σε δυαδική μορφή (ετικέτα – σύνολο – λέξη)		Π(διευθύνσεις μνήμης) => πλαίσιο 0 του συνόλου με διεύθυνση	Π(διευθύνσεις μνήμης) => πλαίσιο 1 του συνόλου με διεύθυνση
33	010-000-1	Miss	Π(32, 33) => 0	
17	001-000-1	Miss		Π(16, 17) => 0
30	001-111-0	Miss	Π(30, 31) => 7	
47	010-111-1	Miss		Π(46, 47) => 7
50	011-001-0	Miss	Π(50, 51) => 1	
35	010-001-1	Miss		Π(34, 35) => 1
6	000-011-0	Miss	Π(6, 7) => 3	
26	001-101-0	Miss	Π(26, 27) => 5	
50	011-001-0	Hit		
42	010-101-0	Miss		Π(42, 43) => 5
58	011-101-0	Miss	Π(58, 59) => 5 (FIFO)	
50	011-001-0	Hit		
13	000-110-1	Miss	Π(12, 13) => 6	
22	001-011-0	Miss		Π(22, 23) => 3
15	000-111-1	Miss	Π(14, 15) => 7 (FIFO)	

4	000-010-0	Miss	Π(4, 5) => 2	
0	000-000-0	Miss	Π(0, 1) => 0 (FIFO)	
70	100-011-0	Miss	Π(70, 71) => 3 (FIFO)	

Τα τελικά περιεχόμενα της κρυφής μνήμης είναι τα ακόλουθα:

Σύνολο	Διευθύνσεις θέσεων της κύριας μνήμης τα περιεχόμενα των οποίων βρίσκονται στο Πλαίσιο 0	Διευθύνσεις θέσεων της κύριας μνήμης τα περιεχόμενα των οποίων βρίσκονται στο Πλαίσιο 1
000	Π(0, 1)	Π(16, 17)
001	Π(50, 51)	Π(34, 35)
010	Π(4, 5)	-
011	Π(70, 71)	Π(22, 23)
100	-	-
101	Π(58, 59)	Π(42, 43)
110	Π(12, 13)	-
111	Π(14, 15)	Π(46, 47)

(Η παύλα σημαίνει ότι το συγκεκριμένο πλαίσιο δεν έχει έγκυρα περιεχόμενα.)

Ερώτημα (γ)

Η κρυφή μνήμη έχει οργάνωση πλήρους συσχέτισης, χωρητικότητα 32 λέξεων και τέσσερις λέξεις ανά πλαίσιο. Άρα υπάρχουν οκτώ πλαίσια των τεσσάρων λέξεων. Από τα επτά ψηφία της διεύθυνσης απαιτούνται δύο δυαδικά ψηφία για να καταδείξουν τη θέση της λέξης μέσα σε ένα πλαίσιο και πέντε δυαδικά ψηφία ως ετικέτα, η οποία δηλώνει τη διεύθυνση του μπλοκ που έχει αποθηκευτεί σε συγκεκριμένο πλαίσιο.

Διεύθυνση μνήμης	Διεύθυνση σε δυαδική μορφή (ετικέτα –λέξη)		Π(διευθύνσεις μνήμης) => πλαίσιο με διεύθυνση και (ετικέτα πλαισίου)
33	01000-01	Miss	Π(32, 33, 34, 35) => 0
17	00100-01	Miss	Π(16, 17, 18, 19) => 1
30	00111-10	Miss	Π(28, 29, 30, 31) => 2
47	01011-11	Miss	Π(44, 45, 46, 47) => 3
50	01100-10	Miss	Π(48, 49, 50, 51) => 4
35	01000-11	Hit	
6	00001-10	Miss	Π(4, 5, 6, 7) => 5
26	00110-10	Miss	Π(24, 25, 26, 27) => 6
50	01100-10	Hit	
42	01010-10	Miss	Π(40, 41, 42, 43) => 7
58	01110-10	Miss	Π(56, 57, 58, 59) => 1 (LRU)
50	01100-10	Hit	
13	00011-01	Miss	Π(12, 13, 14, 15) => 2 (LRU)
22	00101-10	Miss	Π(20, 21, 22, 23) => 3 (LRU)
15	00011-11	Hit	
4	00001-00	Hit	
0	00000-00	Miss	Π(0, 1, 2, 3) => 0 (LRU)
70	10001-10	Miss	Π(68, 69, 70, 71) => 6 (LRU)

Τα τελικά περιεχόμενα της κρυφής μνήμης είναι τα ακόλουθα.

Διεύθυνση πλαισίου	Διευθύνσεις θέσεων της κύριας μνήμης τα περιεχόμενα των οποίων βρίσκονται στο συγκεκριμένο πλαίσιο
000	0, 1, 2, 3
001	56, 57, 58, 59
010	12, 13, 14, 15
011	20, 21, 22, 23
100	48, 49, 50, 51
101	4, 5, 6, 7
110	68, 69, 70, 71
111	40, 41, 42, 43

ΑΣΚΗΣΗ 53

Έστω κρυφή μνήμη άμεσης οργάνωσης συνολικής χωρητικότητας 16K λέξεων (bytes), με πλαίσια των 8 λέξεων και που χρησιμοποιεί τις τακτικές «τελικής ενημέρωσης» (write-back) και «προσκόμισης κατά την εγγραφή» (write-allocate). Έστω ότι απαιτείται 1 κύκλος για την προσπέλαση της κρυφής μνήμης. Αν τα δεδομένα δε βρεθούν στην κρυφή μνήμη απαιτούνται επιπλέον 16 κύκλοι για την προσπέλαση της κύριας μνήμης και την ανάκτηση ολόκληρου του μπλοκ μνήμης στο οποίο ανήκει η ζητούμενη διεύθυνση. Οπότε μια προσπέλαση που δε βρίσκει τα δεδομένα που ζητά στην κρυφή μνήμη χρειάζεται τουλάχιστον (16 + 1) κύκλους για να ολοκληρωθεί είτε πρόκειται για ανάγνωση είτε για εγγραφή. Στην περίπτωση αποβολής (αντικατάστασης) ενός μπλοκ από τη κρυφή μνήμη και εφόσον χρειάζεται απαιτούνται επιπλέον 16 κύκλοι για την ενημέρωση της κύριας μνήμης. Υπολογίστε τον συνολικό αριθμό κύκλων που απαιτούνται για την ολοκλήρωση των παρακάτω ακολουθιών προσπέλασης διευθύνσεων λέξεων μνήμης. Υποθέστε πως οι προσπελάσεις εξυπηρετούνται από το σύστημα μνήμης ξεχωριστά η μία μετά την άλλη. Σε καθένα από τα υποερωτήματα θεωρήστε πως αρχικά η κρυφή μνήμη είναι άδεια. Εξηγείστε αναλυτικά τους υπολογισμούς σας.

α) Ακολουθία διευθύνσεων και τύπου προσπέλασης:
Ανάγνωση: 0,1,2,3…,16382,16383. Σημείωση: 16383 = 16K - 1

β) Η ακολουθία του υποερωτήματος (α) επαναλαμβανόμενη 10 φορές, δηλαδή:
0,1,2,...,16383,0,1,2,...,16383,...,0,1,2...,16383.

γ) Ανάγνωση 0,1,2,...,65535.

δ) Εγγραφή 0,1,2,...,32767, Ανάγνωση 32768,32769,...,65535.

Λύση:

α) Ανάγνωση: 0,1,2,3…,16382,16383.

Εφόσον αρχικά η κρυφή μνήμη είναι άδεια, η προσπέλαση της διεύθυνσης 0 θα χρειαστεί 1 + 16 κύκλους. Εφόσον τα πλαίσια είναι των 8 λέξεων μαζί με την τιμή της διεύθυνσης 1 θα διαβαστούν και οι τιμές των διευθύνσεων 1 ως και 7. Οι προσπελάσεις 1 ως και 7 θα ολοκληρωθούν σε ένα κύκλο η κάθε μία. Οπότε συνολικά χρειαζόμαστε (1 + 16) + 7 = 24 κύκλους για τις προσπελάσεις 0,1,..,7.
Ανάλογα θα χρειαστούν 24 κύκλοι για τις προσπελάσεις 8, 9,..., 15, άλλοι 24 για τις προσπελάσεις 16, 17, ..., 23 κ.ο.κ. Οπότε για την ανάγνωση καθενός μπλοκ των 8 λέξεων χρειαζόμαστε 24 κύκλους. Εφόσον διαβάζουμε 16K λέξεις και επειδή δε γίνεται καμία εγγραφή θα χρειαστούμε συνολικά 16K/8 * 24 = 49152 κύκλοι.

β) Η ακολουθία του υποερωτήματος (α) επαναλαμβανόμενη 10 φορές, δηλαδή:
0,1,2,...,16383,0,1,2,...,16383,...,0,1,2...,16383.

Για την πρώτη ακολουθία 0,1,...,16383 θα χρειαστούν 49152 κύκλοι όπως υπολογίσαμε προηγουμένως. Μετά την ολοκλήρωση της πρώτης ακολουθίας και επειδή η χωρητικότητα της κρυφής μνήμης είναι 16K οι τιμές όλων των διευθύνσεων 0 ως και 16383 θα βρίσκονται αποθηκευμένες στη κρυφή μνήμη. Οπότε όλες οι υπόλοιπες προσπελάσεις θα βρίσκουν το ζητούμενο τους μέσα στην κρυφή μνήμη. Έτσι θα χρειαστούν 16384 για κάθε μία από τις υπόλοιπες 9 ακολουθίες 0,1,...,16383. Οπότε συνολικά απαιτούνται 49152 + 9 * 16384 = 196608 κύκλοι.

γ) Ανάγνωση 0,1,2,...,65535.

Η ακολουθία αυτή αποτελείται από 4 συνεχόμενα μπλοκ των 16K λέξεων που είναι τα: 0-16383, 16384-32767, 32768-49151, 49152-65535 (32768 = 32K, 49152 = 48K και 65536= 64K). Επειδή η χωρητικότητα της κρυφής μνήμης είναι μόνο 16K κάθε μπλοκ των 16K θα αποβάλει το προηγούμενο του. Για το πρώτο μπλοκ θα χρειαστούν 49152 κύκλοι όπως υπολογίσαμε στο υποερώτημα (α). Ο ίδιος αριθμός κύκλων θα χρειαστεί και για καθένα από τα υπόλοιπα 3 μπλοκ μια και δεν έχουν καμία κοινή διεύθυνση. Τέλος επειδή δεν κάνουμε καμία εγγραφή δεν απαιτούνται επιπλέον κύκλοι για την ενημέρωση της κύριας μνήμης κατά την αποβολή ενός μπλοκ από τη κρυφή μνήμη. Έτσι συνολικά θα χρειαστούν 4 * 49152 = 196608 κύκλοι.

δ) Εγγραφή 0,1,2,...,32767, Ανάγνωση 32768,32769,...,65535.

Όπως είδαμε στο υποερώτημα (γ) η ακολουθία αυτή αποτελείται από 4 συνεχόμενα μπλοκ των 16K λέξεων μόνο που τώρα για τα πρώτα δύο μπλοκ κάνουμε εγγραφή.

Για το πρώτο μπλοκ 16K (0-16383): Επειδή χρησιμοποιούμε την τακτική της «προσκόμισης κατά την εγγραφή» μόλις προσπαθήσουμε να γράψουμε την πρώτη λέξη ενός μπλοκ των 8 λέξεων θα φέρουμε από τη κύρια μνήμη και τις 8 λέξεις του (π.χ. μόλις προσπαθήσουμε να γράψουμε στη διεύθυνση 0 θα φέρουμε και τις τιμές των διευθύνσεων 1 ως και 7). Οπότε για την εγγραφή στην πρώτη διεύθυνση ενός μπλοκ των 8 λέξεων χρειαζόμαστε 1 + 16 = 17 κύκλους ενώ για την εγγραφή στις υπόλοιπες 7 λέξεις του μπλοκ θα χρειαστούν επιπλέον 7 κύκλοι. Συνολικά για κάθε μπλοκ των 8 λέξεων έχουμε 24 κύκλους. Οπότε για όλο το πρώτο μπλοκ των 16K λέξεων απαιτούνται 16K/8*24 = 49152 κύκλοι. Στο σημείο αυτό όλα τα πλαίσια της κρυφής μνήμης περιέχουν δεδομένα που πρέπει κατά την αποβολή τους να γραφούν στη μνήμη.

Για το δεύτερο μπλοκ των 16K (16384-32763): Όπως και με το πρώτο θα απαιτηθούν 49152 κύκλοι για τις εγγραφές. Επειδή όμως η κρυφή μνήμη είναι γεμάτη στην αρχή, κάθε προσπέλαση ενός νέου μπλοκ των 8 λέξεων θα έχει σαν αποτέλεσμα και την αποβολή ενός άλλου από τη κρυφή μνήμη. Επειδή τα υπάρχοντα μπλοκ περιέχουν δεδομένα που δεν έχουν ακόμα γραφτεί στη κύρια μνήμα (πολιτική «εγγραφή κατά την αντικατάσταση») θα απαιτηθούν και επιπλέον 16 κύκλοι ανά μπλοκ που αποβάλλεται (αντικαθίστανται). Άρα για κάθε μπλοκ των 8 λέξεων έχουμε αρχικά 16 κύκλους για την ενημέρωση της κύρια μνήμης με τις τιμής του μπλοκ που αποβάλλεται από τη κρυφή μνήμη (1+16) για την ανάγνωση του νέου μπλοκ και την ολοκλήρωση της πρώτης εγγραφής και 7 κύκλους για τις υπόλοιπες 7 εγγραφές. Συνολικά λοιπόν θα απαιτηθούν 16K/8*(16 + (1 + 16) + 7) = 81920 κύκλοι για το δεύτερο μπλοκ των 16K.

Το τρίτο μπλοκ των 16K (32768-49151) θα αντικαταστήσει διαδοχικά όλες τις διευθύνσεις του δεύτερου μπλοκ (16384-32763) που βρίσκονται αρχικά στη κρυφή μνήμη. Έτσι θα χρειαστούν 32768 κύκλοι για την ενημέρωση της κύριας μνήμης με τις τιμές για τις διευθύνσεις (16384-32763) και 49152 κύκλοι (βάση του υποερωτήματος (α)) για την ανάγνωση των νέων διευθύνσεων οπότε έχουμε 81920 κύκλους.

Το τέταρτο μπλοκ των 16K (49152-65535) θα αντικαταστήσει διαδοχικά το τρίτο αλλά δεν απαιτείται ενημέρωση της κύριας μνήμης μια και για τα δύο τελευταία μπλοκ των 16K κάνουμε μόνο ανάγνωση. Άρα βάση των υποερωτήματων (α) και (β) θα χρειαστούν 49152 κύκλοι.

Συνολικά θα χρειαστούν 49152 + 81920 + 81920 + 49152 = 262144 κύκλους.

ΑΣΚΗΣΗ 54

Εστω ένα υπολογιστικό σύστημα, η κύρια μνήμη του οποίου αποτελείται από 32 μπλοκ, εφοδιασμένο με μια κρυφή μνήμη με 8 πλαίσια. Υποθέστε ότι κάθε μπλοκ και πλαίσιο έχουν μέγεθος μιας λέξης. Ξεκινώντας με μια εντελώς άδεια κρυφή μνήμη υποθέστε την ακολουθία αναφορών 0, 1, 28, 5, 31, 12, 11, 12, 13, 12, 1, 28, 5, 3, 4, 5, 6, 7, 3, 4, 5, 6, 7. Ποιος είναι ο λόγος επιτυχίας της κρυφής μνήμης όταν

1. Η κρυφή μνήμη είναι άμεσης συσχέτισης ;
2. Η κρυφή μνήμη είναι συνόλου συσχέτισης με δύο πλαίσια ανά σύνολο και χρησιμοποιεί την τεχνική FIFO για την απομάκρυνση των πλαισίων ;

Λύση :

1. Στο παρακάτω πίνακα φαίνονται τα περιεχόμενα του κάθε πλαισίου της κρυφής μνήμης μετά από κάθε αναφορά. Για παράδειγμα στην τελευταία στήλη της 6^{ης} γραμμής υπονοείται ότι στο πλαίσιο 7 της κρυφής μνήμης βρίσκεται αποθηκευμένο μετά την 5^{η} αναφορά το μπλοκ 31 της κύριας μνήμης. Με * σημειώνονται οι αναφορές που μπορούν να εξυπηρετηθούν από τη κρυφή μνήμη.

Αναφορά	***0***	***1***	***2***	***3***	***4***	***5***	***6***	***7***
0	0							
1	0	1						
28	0	1			28			
5	0	1			28	5		
31	0	1			28	5		31
12	0	1			12	5		31
11	0	1		11	12	5		31
12	0	1		11	12*	5		31
13	0	1		11	12	13		31
12	0	1		11	12*	13		31
1	0	1*		11	12	13		31
28	0	1		11	28	13		31
5	0	1		11	28	5		31
3	0	1		3	28	5		31
4	0	1		3	4	5		31
5	0	1		3	4	5*		31
6	0	1		3	4	5	6	31
7	0	1		3	4	5	6	7
3	0	1		3*	4	5	6	7
4	0	1		3	4*	5	6	7
5	0	1		3	4	5*	6	7
6	0	1		3	4	5	6*	7
7	0	1		3	4	5	6	7*

Ο λόγος επιτυχίας είναι συνεπώς 9/23 ή 39.1%.

2. Αφού η κρυφή μας μνήμη έχει 8 πλαίσια και δύο πλαίσια σε κάθε σύνολο συμπεραίνουμε ότι έχει 4 σύνολα. Στο παρακάτω πίνακα που χρησιμοποιεί την ίδια σημειολογία με τον προηγούμενο φαίνονται τα περιεχόμενα κάθε πλαισίου της κρυφής μνήμης μετά από κάθε αναφορά.

	Σύνολο 0		***Σύνολο 1***		***Σύνολο 2***		***Σύνολο 3***	
Αναφορά	***Πλαίσιο 0***	***Πλαίσιο 1***	***Πλαίσιο 0***	***Πλαίσιο 1***	***Πλαίσιο 0***	***Πλαίσιο 1***	***Πλαίσιο 0***	***Πλαίσιο 1***
0	0							
1	0		1					
28	0	28	1					
5	0	28	1	5				
31	0	28	1	5			31	
12	12	28	1	5			31	
11	12	28	1	5			31	11
12	12*	28	1	5			31	11
13	12	28	13	5			31	11
12	12*	28	13	5			31	11
1	12	28	13	1			31	11
28	12	28*	13	1			31	11
5	12	28	5	1			31	11
3	12	28	5	1			3	11
4	12	4	5	1			3	11
5	12	4	5*	1			3	11
6	12	4	5	1	6		3	11
7	12	4	5	1	6		3	7
3	12	4	5	1	6		3*	7
4	12	4*	5	1	6		3	7
5	12	4	5*	1	6		3	7
6	12	4	5	1	6*		3	7
7	12	4	5	1	6		3	7*

Ο λόγος επιτυχίας είναι και πάλι 9/23 ή 39.1%.

ΑΣΚΗΣΗ 55

Εστω ένα υπολογιστικό σύστημα, η κύρια μνήμη του οποίου αποτελείται από 32 μπλοκ, εφοδιασμένο με μια κρυφή μνήμη με 8 πλαίσια. Υποθέστε ότι κάθε μπλοκ και πλαίσιο έχουν μέγεθος δύο λέξεων. Ξεκινώντας με μια εντελώς άδεια κρυφή μνήμη υποθέστε την ακολουθία αναφορών 1, 3, 42, 23, 7, 22, 17, 0, 16, 25, 2, 31, 15, 7, 8, 11, 12, 14, 6, 17, 9, 11, 13, 15, 16, 18 (διευθύνσεις κύριας μνήμης). Ποιος είναι ο λόγος επιτυχίας της κρυφής μνήμης όταν

3. Η κρυφή μνήμη είναι άμεσης οργάνωσης ;
4. Η κρυφή μνήμη είναι συνόλου συσχέτισης με δύο πλαίσια ανά σύνολο και χρησιμοποιεί την τεχνική FIFO για την απομάκρυνση των πλαισίων ;

Λύση :
Μια αναφορά στη κύρια μνήμη αποτελείται από :

- Το πεδίο διεύθυνσης του μπλοκ στη κύρια μνήμη. Αφού έχουμε 32 μπλοκ, συμπεραίνουμε ότι το πεδίο αυτό έχει μέγεθος **5** δυαδικά ψηφία.
- Το πεδίο λέξης εντός του μπλοκ. Αφού σε κάθε μπλοκ έχουμε 2 λέξεις, συμπεραίνουμε ότι το πεδίο αυτό έχει μήκος **1** δυαδικό ψηφίο.

Αρα η διεύθυνσή μας έχει συνολικό μήκος **6** δυαδικά ψηφία.

1. Οταν η κρυφή μνήμη έχει άμεση οργάνωση, η διεύθυνση διαχωρίζεται στα εξής πεδία :
   - Ετικέτα, μήκους **2** δυαδικών ψηφίων,
   - Διεύθυνση πλαισίου, μήκους **3** δυαδικών ψηφίων για το καθορισμό του ενός από τα 8 πλαίσια της κρυφής μνήμης που μπορεί να αποθηκεύει τα ζητούμενα δεδομένα και
   - Διεύθυνση λέξης, μήκους **1** δυαδικού ψηφίου.

Η επιτυχία ή αποτυχία μιας προσπέλασης δεν εξαρτάται από τη ζητούμενη λέξη, παρά μόνο από το εάν η ετικέτα της αναφοράς είναι ίδια με την ετικέτα που υπάρχει στο συγκεκριμένο πλαίσιο που προσπελαύνεται στη μνήμη ετικετών. Στον ακόλουθο πίνακα φαίνονται οι ετικέτες που αποθηκεύονται σε κάθε πλαίσιο της κρυφής μετά από κάθε αναφορά. Για παράδειγμα στην τελευταία στήλη της $14^{ης}$ γραμμής υπονοείται ότι στο πλαίσιο 7 της μνήμης ετικετών βρίσκεται αποθηκευμένο μετά την $12^{η}$ αναφορά η ετικέτα 01. Με * σημειώνονται οι αναφορές που μπορούν να εξυπηρετηθούν από τη κρυφή μνήμη.

	Διαχωρισμός της διεύθυνσης			***Περιεχόμενα των πλαισίων της μνήμης ετικετών***							
Αναφορά	***Ετικέτα***	***Πλαίσιο***	***Λέξη***	***0***	***1***	***2***	***3***	***4***	***5***	***6***	***7***
1	00	000	1	00							
3	00	001	1	00	00						
42	10	101	0	00	00				10		
23	01	011	1	00	00		01		10		
7	00	011	1	00	00		00		10		
22	01	011	0	00	00		01		10		
17	01	000	1	01	00		01		10		
0	00	000	0	00	00		01		10		
16	01	000	0	01	00		01		10		
25	01	100	1	01	00		01	01	10		
2	00	001	0	01	00*		01	01	10		
31	01	111	1	01	00		01	01	10		01
15	00	111	1	01	00		01	01	10		00
7	00	011	1	01	00		00	01	10		00
8	00	100	0	01	00		00	00	10		00
11	00	101	1	01	00		00	00	00		00
12	00	110	0	01	00		00	00	00	00	00
14	00	111	0	01	00		00	00	00	00	00*
6	00	011	0	01	00		00*	00	00	00	00
17	01	000	1	01*	00		00	00	00	00	00
9	00	100	1	01	00		00	00*	00	00	00
11	00	101	1	01	00		00	00	00*	00	00
13	00	110	1	01	00		00	00	00	00*	00
15	00	111	1	01	00		00	00	00	00	00*
16	01	000	0	01*	00		00	00	00	00	00
18	01	001	0	01	01		00	00	00	00	00

Ο λόγος επιτυχίας είναι συνεπώς 9/26 ή 34,6%.

2. Η κρυφή μνήμη έχει 8 πλαίσια και δύο πλαίσια σε κάθε σύνολο. Συνεπώς υπάρχουν 4

σύνολα. Σε αυτή τη περίπτωση, η διεύθυνση διαχωρίζεται στα εξής πεδία :

- Ετικέτα, μήκους **3** δυαδικών ψηφίων,
- Διεύθυνση συνόλου, μήκους **2** δυαδικών ψηφίων για το καθορισμό του ενός από τα 4 σύνολα της κρυφής μνήμης που μπορεί να αποθηκεύει τα ζητούμενα δεδομένα και
- Διεύθυνση λέξης, μήκους **1** δυαδικού ψηφίου.

Στο παρακάτω πίνακα που χρησιμοποιεί την ίδια σημειολογία με τον προηγούμενο, φαίνονται τα περιεχόμενα κάθε πλαισίου της μνήμης ετικετών μετά από κάθε αναφορά. Σε κάθε σύνολο, τα δύο πλαίσια, υλοποιούν μια FIFO, στην οποία δεδομένα εισάγονται στο πλαίσιο 1 και εξάγονται από το πλαίσιο 0. Επομένως, το αριστερό πλαίσιο (πλαίσιο 0) του κάθε συνόλου αποθηκεύει ανά πάσα χρονική στιγμή την ετικέτα που αναφέρθηκε πρώτη (first-in) και είναι αυτό που επιλέγεται σε περίπτωση αντικατάστασης.

				Περιεχόμενα μνήμης ετικετών							
	Διαχωρισμός της Διεύθυνσης			***Σύνολο 0***		***Σύνολο 1***		***Σύνολο 2***		***Σύνολο 3***	
Αναφορά	***Ετικέτα***	***Σύνολο***	***Λέξη***	***Πλαίσιο 0***	***Πλαίσιο 1***	***Πλαίσιο 0***	***Πλαίσιο 1***	***Πλαίσιο 0***	***Πλαίσιο 1***	***Πλαίσιο 0***	***Πλαίσιο 1***
1	000	00	1		000						
3	000	01	1		000		000				
42	101	01	0		000	000	101				
23	010	11	1		000	000	101				010
7	000	11	1		000	000	101			010	000
22	010	11	0		000	000	101			010*	000
17	010	00	1	000	010	000	101			010	000
0	000	00	0	000*	010	000	101			010	000
16	010	00	0	000	010*	000	101			010	000
25	011	00	1	010	011	000	101			010	000
2	000	01	0	010	011	000*	101			010	000
31	011	11	1	010	011	000	101			000	011
15	001	11	1	010	011	000	101			011	001
7	000	11	1	010	011	000	101			001	000
8	001	00	0	011	001	000	101			001	000
11	001	01	1	011	001	101	001			001	000
12	001	10	0	011	001	101	001		001	001	000
14	001	11	0	011	001	101	001		001	001*	000
6	000	11	0	011	001	101	001		001	001	000*
17	010	00	1	001	010	101	001		001	001	000
9	001	00	1	001*	010	101	001		001	001	000
11	001	01	1	001	010	101	001*		001	001	000
13	001	10	1	001	010	101	001		001*	001	000
15	001	11	1	001	010	101	001		001	001*	000
16	010	00	0	001	010*	101	001		001	001	000
18	010	01	0	001	010	001	010		001	001	000

Ο λόγος επιτυχίας στην περίπτωση αυτή είναι 11/26 ή 42,3%.

ΑΣΚΗΣΗ 56

Θεωρείστε ένα σύστημα με αρτηρία διευθύνσεων των 16 δυαδικών ψηφίων. Κάθε μπλοκ της κύριας μνήμης είναι των 4 ψηφιολέξεων (bytes). Στο σύστημα αυτό προσαρτάται μια κρυφή μνήμη συνόλου συσχέτισης με τα εξής χαρακτηριστικά :

- Μέγεθος 128 bytes.
- 4 πλαίσια ανά σύνολο.
- 4 λέξεις ανά πλαίσιο.
- LRU (Least Recently Used) στρατηγική απελευθέρωσης πλαισίων.
- Προσκόμιση κατά την εγγραφή (fetch-on-write).

α) Αριθμώντας τα σύνολα της κρυφής μνήμης ως **0**, 1, 2, 3, 4, ..., συμπληρώστε στον παρακάτω πίνακα σε ποιο σύνολο της κρυφής μνήμης μπορούν να αποθηκευτούν οι πληροφορίες με τις διευθύνσεις της πρώτης στήλης (οι διευθύνσεις είναι στο δεκαεξαδικό).

Διεύθυνση της πληροφορίας (hex)	Ζητούμενο σύνολο
F3A2	
DDCD	
0B21	
3333	
2170	
CCCC	
2005	
0001	

β) Δώστε τη γενική μορφή των διευθύνσεων, οι πληροφορίες των οποίων μπορούν να αποθηκευτούν στο σύνολο 5 της κρυφής μνήμης.

γ) Ποιο είναι το συνολικό μέγεθος (σε bits) της μνήμης στην οποία η κρυφή μνήμη αποθηκεύει τις ετικέτες της ?

δ) Δώστε τα περιεχόμενα των μνημών ετικέτας του συνόλου 6 της κρυφής μνήμης, μετά την εξής ακολουθία διευθύνσεων (οι διευθύνσεις είναι στο δυαδικό) :

```
1111 0010 1101 1011
0001 1101 0000 1011
0001 1101 0001 1000
1111 0010 1101 1011
0000 0000 0001 1010
1101 1111 0001 1111
0010 1100 0111 1011
1010 1010 1011 1010
0000 0001 1111 1110
1001 1001 1001 1001
```

Λύση

α) Αφού υπάρχουν 4 bytes ανά πλαίσιο και 4 πλαίσια ανά σύνολο, συμπεραίνουμε ότι σε κάθε σύνολο έχουμε 16 bytes. Συνεπώς η κρυφή μας μνήμη έχει 128 / 16 = 8 σύνολα. Συνεπώς η φυσική διεύθυνση χωρίζεται στα εξής 3 πεδία :

<- 11 bits ->	<- 3 bits ->	<- 2 bits ->
Ετικέτα	Επιλογή συνόλου	Επιλογή λέξης

Το σύνολο στο οποίο αναφέρεται μια διεύθυνση καθορίζεται από τα 3 δυαδικά ψηφία του πεδίου "Επιλογή συνόλου". Ετσι παίρνουμε τον παρακάτω πίνακα :

Διεύθυνση της πληροφορίας (hex)		Ζητούμενο σύνολο
F3A2	(1111 0011 101 **0 00** 10)	0
DDCD	(1101 1101 110 **0 11** 01)	3
0B21	(0000 1011 001 **0 00** 01)	0
3333	(0011 0011 001 **1 00** 11)	4
2170	(0010 0001 011 **1 00** 00)	4
CCCC	(1100 1100 110 **0 11** 00)	3
2005	(0010 0000 000 **0 01** 01)	1
0001	(0000 0000 000 **0 00** 01)	0

β) XXXX XXXX XXX1 01XX.

γ) Το μέγεθος της μνήμης ετικετών είναι :
11 (δυαδικά ψηφία ετικέτας) * 8 (σύνολα) * 4 (πλαίσια ανά σύνολο) = 352 bits.

δ) Οι ετικέτες ενός συνόλου υποδεικνύουν ποιες από τις πληροφορίες της κύριας μνήμης που μπορεί να αποθηκευτούν στο σύνολο αυτό, βρίσκονται αποθηκευμένες τη τρέχουσα χρονική στιγμή στη κρυφή μνήμη. Αρα πρώτα πρέπει να βρούμε ποιές από τις δοθείσες αναφορές επηρρεάζουν τις ετικέτες του συνόλου 6 ή ισοδύναμα ποιες από τις δοθείσες αναφορές είναι αναφορές του συνόλου 6. Σύμφωνα με το προηγούμενο ερώτημα μας μένουν οι ακόλουθες αναφορές :

1111 0010 110 **1 10** 11
~~0001 1101 000 **0 10** 11~~
0001 1101 000 **1 10** 00
1111 0010 110 **1 10** 11
0000 0000 000 **1 10** 10
~~1101 1111 000 **1 11** 11~~
0010 1100 011 **1 10** 11
1010 1010 101 **1 10** 10
~~0000 0001 111 **1 11** 10~~
1001 1001 100 **1 10** 01

Αφού σε κάθε σύνολο εφαρμόζεται στατηγική LRU (Least Recently Used) για την απελευθέρωση πλαισίων, στο σύνολο 6 μετά το τέλος των παραπάνω αναφορών θα αποθηκεύονται οι πληροφορίες και οι αντίστοιχες ετικέτες των πλέον προσφάτως χρησιμοποιηθέντων διευθύνσεων. Αρα στη μνήμη ετικετών για το σύνολο 6 θα βρίσκονται αποθηκευμένες οι εξής ετικέτες :

0000 0000 000
0010 1100 011
1010 1010 101
1001 1001 100

ΑΣΚΗΣΗ 57

Θεωρείστε ότι η CPU ενός υπολογιστή, που παράγει διευθύνσεις των 8 δυαδικών ψηφίων, παράγει την επόμενη ακολουθία διευθύνσεων (οι διευθύνσεις είναι γραμμένες στο δεκαεξαδικό σύστημα αρίθμησης): 41, 76, B3, 00, 43, 70, 04, B1, 72, 02, 45, B7 . Η CPU διαθέτει κρυφή μνήμη με χωρητικότητα 64 λέξεις και 8 λέξεις ανά πλαίσιο. Στο ξεκίνημα η κρυφή μνήμη δεν έχει έγκυρα περιεχόμενα.

α. Αν η κρυφή μνήμη είναι άμεσης οργάνωσης ποιο είναι το ποσοστό επιτυχίας της για την παραπάνω ακολουθία διευθύνσεων; Αποδείξτε την απάντηση σας.

β. Ποιο είναι το μεγαλύτερο ποσοστό επιτυχίας που μπορεί να πετύχει οποιαδήποτε οργάνωση κρυφής μνήμης με τα παραπάνω χαρακτηριστικά (χωρητικότητα 64 λέξεις και 8 λέξεις ανά πλαίσιο), για την δεδομένη ακολουθία διευθύνσεων; Γιατί;

γ. Υποδείξτε την πιο φθηνή οργάνωση που δίνει το μεγαλύτερο ποσοστό επιτυχίας, για την δεδομένη ακολουθία διευθύνσεων.

Λύση

α. Η κρυφή μνήμη έχει 8 πλαίσια των 8 λέξεων το κάθε ένα οπότε οι αναφορές των διευθύνσεων είναι οι ακόλουθες:

Διεύθυνση	Ετικέτα	Πλαίσιο	Λέξη	
41	01	000	001	miss
76	01	110	110	miss

B3	10	110	011	miss
00	00	000	000	miss
43	01	000	011	miss
70	01	110	000	miss
04	00	000	100	miss
B1	10	110	001	miss
72	01	110	010	miss
02	00	000	010	hit
45	01	000	101	miss
B7	10	110	111	miss

Άρα το ποσοστό επιτυχίας σε αυτή την περίπτωση είναι 1/12 = 8,33%

β. *1ος Τρόπος*

Αφού κάθε πλαίσιο της κρυφής μνήμης αποτελείται από 8 λέξεις, μπορούμε να θεωρήσουμε ότι η κύρια μνήμη χωρίζεται (νοητά) σε μπλοκ μεγέθους 8 λέξεων το κάθε ένα. Τα τρία τελευταία δυαδικά ψηφία της διεύθυνσης καθορίζουν την λέξη μέσα στο μπλοκ και τα πέντε περισσότερο σημαντικά αποτελούν την διεύθυνση (αύξων αριθμό) του μπλοκ. Κάθε ένα μπλοκ της κύριας μνήμης αντιστοιχίζεται σε ένα ή περισσότερα πλαίσια της κρυφής μνήμης, ανάλογα με την οργάνωση της κρυφής μνήμης. Οι διευθύνσεις και τα μπλοκ στα οποία αντιστοιχούν φαίνονται ακόλουθα:

Διευθύνσεις Κύριας Μνήμης (Δεκαεξ/κό)	**Διευθύνσεις Κύριας Μνήμης (Δυαδικό)**	**Διεύθυνση Μπλόκ Κύριας Μνήμης (Δυαδικό)**	**Ζητούμενες Διευθύνσεις Κύριας Μνήμης ανά μπλοκ(Δεκαεξ/κό)**
00 … 07	**00000** 000 … **00000** 111	**00000**	00, 02, 04
08 … 0F	**00001** 000 … **00001** 111	**00001**	-
10 … 17	**00010** 000 … **00010** 111	**00010**	-
…	…	…	…
40 … 47	**01000** 000 … **01000** 111	**01000**	41, 43, 45
…	…	…	…
70 … 77	**01110** 000 … **01110** 111	**01110**	70, 72, 76
…	…	…	…
B0 … B7	**10110** 000 … **10110** 111	**10110**	B1, B3, B7
…	…	…	…

Άρα οι ζητούμενες διευθύνσεις ανήκουν σε 4 διαφορετικά μπλοκ (00000, 01000, 01110, 10110). Μία οργάνωση που θα πετυχαίνει το μεγαλύτερο ποσοστό επιτυχίας θα είναι εκείνη στην οποία θα μπορούν να συνυπάρχουν ταυτόχρονα και τα 4 αυτά μπλοκ. Τέτοια οργάνωση

κρυφής μνήμης για παράδειγμα είναι η οργάνωση πλήρους συσχέτισης, που στην περίπτωση μας θα αποτελείται από 8 πλαίσια (οπότε μπορούν να συνυπάρχουν ταυτόχρονα τα 4 μπλοκ). Έτσι θα έχει επιτυχία σε όλες τις αναφορές (για την δεδομένη ακολουθία διευθύνσεων) εκτός φυσικά από τις 4 αποτυχίες που οφείλονται στην αρχική προσκόμιση των ζητούμενων μπλοκ (η κρυφή μνήμη είναι άδεια αρχικά). Άρα το μεγαλύτερο ποσοστό επιτυχίας μπορεί να είναι 8 / 12 = 66,67%

2ος Τρόπος

β. Στη κρυφή μνήμη υπάρχουν 8 πλαίσια συνολικά. Αφού σε κάθε πλαίσιο της κρυφής μνήμης υπάρχουν 8 λέξεις, συμπεραίνουμε ότι ανεξάρτητα από την οργάνωση της κύριας μνήμης, τα 3 λιγότερο σημαντικά ψηφία της διεύθυνσης θα χρειάζονται για να επιλέξουν τη λέξη εντός του πλαισίου. Συνεπώς οι αναφορές που μας δίδονται μπορούν να χωριστούν σε δύο μέρη :

	1ο μέρος	2ο μέρος
41 :	01000	001
76 :	01110	110
B3 :	10110	011
00 :	00000	000
43 :	01000	011
70 :	01110	000
04 :	00000	100
B1 :	10110	001
72 :	01110	010
02 :	00000	010
45 :	01000	101
B7 :	10110	111

Ανάλογα με το βαθμό συσχέτισης, το 1ο μέρος κατανέμεται ως εξής:

πλήρως συσχετιστική οργάνωση :	5 bit ετικέτα	0 bit επιλογή πλαισίου / συνόλου
4-τρόπων συνόλου συσχέτισης :	4 bit ετικέτα	1 bit επιλογή συνόλου
2-τρόπων συνόλου συσχέτισης :	3 bit ετικέτα	2 bit επιλογή συνόλου
άμεση οργάνωση :	2 bit ετικέτα	3 bit επιλογή πλαισίου.

Η κάθε οργάνωση της κρυφής μνήμης με οδηγεί στο να χωρίσω το 1ο μέρος των παραπάνω διευθύνσεων σε "αντικρουόμενες" ομάδες, σε ομάδες δηλαδή που "μάχονται" για το ίδιο κομμάτι πλαισίων. Για την άμεση οργάνωση αυτό έχει γίνει στο (α) υποερώτημα.

Πλήρως συσχετιστική :
Ομάδα με ετικέτα 01000,
Ομάδα με ετικέτα 01110,
Ομάδα με ετικέτα 10110 και
Ομάδα με ετικέτα 00000.
όπου οι 4 ομάδες μάχονται για όλα τα (8 δηλαδή) πλαίσια της κρυφής μνήμης.

4-τρόπων συνόλου συσχέτισης :
Ομάδα με ετικέτα 0100,
Ομάδα με ετικέτα 0111,
Ομάδα με ετικέτα 1011 και
Ομάδα με ετικέτα 0000.
όπου οι 4 ομάδες μάχονται για τα (4) πλαίσια του σύνολου 0.

2-τρόπων συνόλου συσχέτισης :
Ομάδα με ετικέτα 010 που μάχεται για πλαίσια του συνόλου 0.
Ομάδα με ετικέτα 000 που μάχεται για πλαίσια του συνόλου 0.

Ομάδα με ετικέτα 011 που μάχεται για πλαίσια του συνόλου 2.
Ομάδα με ετικέτα 101 που μάχεται για πλαίσια του συνόλου 2.

Σε όλες τις παραπάνω οργανώσεις παρατηρούμε ότι ο αριθμός των αντικρουόμενων ομάδων είναι μικρότερος ή ίσος του διαθέσιμου αριθμού πλαισίων ανά σύνολο. Άρα οι αντικρουόμενες ομάδες μπορούν να συνυπάρχουν στη κρυφή μνήμη και, για τη δεδομένη ακολουθία διευθύνσεων και οι τρεις παραπάνω οργανώσεις θα προσφέρουν το ίδιο ποσοστό επιτυχίας. Δηλαδή μετά από 4 αποτυχημένες αναφορές, κάθε μία προερχόμενη από διαφορετική ομάδα, όλες οι υπόλοιπες θα δίνουν επιτυχία. Το μέγιστο ποσοστό επιτυχίας είναι συνεπώς 8/12 = 66,67%.

γ. Η επόμενη φθηνότερη οργάνωση (μετά την άμεση που είδαμε στο ερώτημα α) είναι η οργάνωση συνόλου συσχέτισης με 2 πλαίσια ανά σύνολο:

Διεύθυνση	Εττικέτα	Πλαίσιο	Λέξη	
41	010	00	001	miss
76	011	10	110	miss
B3	101	10	011	miss
00	000	00	000	miss
43	010	00	011	hit
70	011	10	000	hit
04	000	00	100	hit
B1	101	10	001	hit
72	011	10	010	hit
02	000	00	010	hit
45	010	00	101	hit
B7	101	10	111	hit

Άρα το ποσοστό επιτυχίας σε αυτή την περίπτωση είναι 8/12 = 66,67% και είναι πράγματι το υψηλότερο δυνατό.
Οι υπόλοιπες οργανώσεις είναι πιο ακριβές από την οργάνωση συνόλου συσχέτισης με 2 πλαίσια ανά σύνολο οπότε δεν τις μελετάμε. Άρα η ζητούμενη οργάνωση είναι η οργάνωση συνόλου συσχέτισης με 2 πλαίσια ανά σύνολο.

ΑΣΚΗΣΗ 58

Έστω ένα υπολογιστικό σύστημα, με κύρια μνήμη μεγέθους 128 bytes, εφοδιασμένο με μια κρυφή μνήμη μεγέθους 32 bytes. Υποθέστε ότι κάθε μπλοκ και πλαίσιο έχουν μέγεθος 4 λέξεων (bytes). Ξεκινώντας με μια εντελώς άδεια κρυφή μνήμη υποθέστε την ακολουθία αναφορών 32, 68, 01, 108, 33, 02, 76, 101, 34, 70, 3, 108 (διευθύνσεις κύριας μνήμης). Ποιος είναι ο λόγος επιτυχίας της κρυφής μνήμης όταν

5. Η κρυφή μνήμη είναι άμεσης οργάνωσης;
6. Ποια είναι η φθηνότερη οργάνωση της κρυφής μνήμης (με μέγεθος πλαισίου 4 λέξεων) που μεγιστοποιεί τον λόγο επιτυχίας της και γιατί; Θεωρήστε ότι η τεχνική αντικατάστασης πλαισίων στην κρυφή μνήμη για τις οργανώσεις συνόλου συσχέτισης είναι η FIFO.

Λύση :
Η διεύθυνσή μας έχει συνολικό μήκος $\log_2 128 = 7$ δυαδικά ψηφία. Η κρυφή μνήμη αποτελείται από 32/4=8 πλαίσια.

1. Όταν η κρυφή μνήμη έχει άμεση οργάνωση, η διεύθυνση διαχωρίζεται στα εξής πεδία :
   - Ετικέτα, μήκους **2** δυαδικών ψηφίων,

- Διεύθυνση πλαισίου, μήκους **3** δυαδικών ψηφίων για το καθορισμό του ενός από τα 8 πλαίσια της κρυφής μνήμης που μπορεί να αποθηκεύει τα ζητούμενα δεδομένα και
- Διεύθυνση λέξης, μήκους **2** δυαδικών ψηφίων.

Η επιτυχία ή αποτυχία μιας προσπέλασης δεν εξαρτάται από τη ζητούμενη λέξη, παρά μόνο από το εάν η ετικέτα της αναφοράς είναι ίδια με την ετικέτα που υπάρχει στο συγκεκριμένο πλαίσιο που προσπελαύνεται στη μνήμη ετικετών. Στον ακόλουθο πίνακα φαίνονται οι ετικέτες που αποθηκεύονται σε κάθε πλαίσιο της κρυφής μνήμης μετά από κάθε αναφορά.

	Διαχωρισμός της διεύθυνσης			***Ετικέτες των πλαισίων της μνήμης***								
Αναφορά	***Ετικέτα***	***Πλαίσιο***	***Λέξη***	***0***	***1***	***2***	***3***	***4***	***5***	***6***	***7***	***Επιτ./ Αποτ.***
32	01	000	00	01								Αποτ.
68	10	001	00	01	10							Αποτ.
01	00	000	01	00	10							Αποτ.
108	11	011	00	00	10		11					Αποτ.
33	01	000	01	01	10		11					Αποτ.
02	00	000	10	00	10		11					Αποτ.
76	10	011	00	00	10		10					Αποτ.
101	11	001	01	00	11		10					Αποτ.
34	01	000	10	01	11		10					Αποτ.
70	10	001	10	01	10		10					Αποτ.
03	00	000	11	00	10		10					Αποτ.
108	11	011	00	00	10		11					Αποτ.

Ο λόγος επιτυχίας είναι συνεπώς 0/12 ή 0%.

3. Η επόμενη φθηνότερη οργάνωση είναι η οργάνωση συνόλου συσχέτισης 2 τρόπων. Σε αυτή την περίπτωση η κρυφή μνήμη έχει 4 σύνολα με δύο πλαίσια σε κάθε σύνολο και η διεύθυνση διαχωρίζεται στα εξής πεδία :
   - Ετικέτα, μήκους **3** δυαδικών ψηφίων,
   - Διεύθυνση συνόλου, μήκους **2** δυαδικών ψηφίων για το καθορισμό του ενός από τα 4 σύνολα της κρυφής μνήμης που μπορεί να αποθηκεύει τα ζητούμενα δεδομένα και
   - Διεύθυνση λέξης, μήκους **2** δυαδικών ψηφίων.

Στον παρακάτω πίνακα που χρησιμοποιεί την ίδια σημειολογία με τον προηγούμενο, φαίνονται οι ετικέτες των διευθύνσεων τα περιεχόμενα των οποίων βρίσκονται σε κάθε πλαίσιο της κρυφής μνήμης μετά από κάθε αναφορά. Σε κάθε σύνολο, τα δύο πλαίσια (Π0, Π1), υλοποιούν μια FIFO, στην οποία δεδομένα εισάγονται στο πλαίσιο 1 (Π1) και εξάγονται από το πλαίσιο 0 (Π0). Επομένως, το αριστερό πλαίσιο (Π0) του κάθε συνόλου αποθηκεύει ανά πάσα χρονική στιγμή την ετικέτα που αναφέρθηκε πρώτη (first-in) και είναι αυτό που επιλέγεται σε περίπτωση αντικατάστασης.

				Ετικέτες των πλαισίων της μνήμης								***Επιτ./ Αποτ.***
	Διαχωρισμός της Διεύθυνσης			***Σύνολο 0***		***Σύνολο 1***		***Σύνολο 2***		***Σύνολο 3***		
Αναφορά	***Ετικέτα***	***Σύνολο***	***Λέξη***	***Π0***	***Π1***	***Π0***	***Π1***	***Π0***	***Π1***	***Π0***	***Π1***	
32	010	00	00		010							Αποτ.
68	100	01	00				100					Αποτ.
01	000	00	01	010	000							Αποτ.
108	110	11	00								110	Αποτ.
33	010	00	01	010	000							Επιτ.
02	000	00	10	010	000							Επιτ.
76	100	11	00							110	100	Αποτ.

101	110	01	01			100	110					Αποτ.
34	010	00	10	010	000							Επίτ.
70	100	01	10			100	110					Επίτ.
03	000	00	11	010	000							Επίτ.
108	110	11	00							110	100	Επίτ.

Ο λόγος επιτυχίας στην περίπτωση αυτή είναι 6/12 ή 50%.

Παρατηρούμε ότι οι 6 αποτυχημένες αναφορές οφείλονται στην αρχική προσκόμιση των αντίστοιχων μπλοκ στην κρυφή μνήμη. Το κόστος αυτό σε αποτυχημένες αναφορές δεν είναι δυνατόν να αποφευχθεί, οποιαδήποτε οργάνωση κρυφής μνήμης και αν επιλέξουμε. Άρα η κρυφή μνήμη 2 τρόπων συνόλου συσχέτισης είναι οργάνωση με μέγιστο λόγο επιτυχίας. Επιπλέον είναι και η φθηνότερη οργάνωση από κάθε άλλη οργάνωση συνόλου συσχέτισης περισσότερων των 2 τρόπων, άρα αποτελεί την ζητούμενη οργάνωση.

ΑΣΚΗΣΗ 59

Σε κάθε μπλοκ της κύριας μνήμης ενός υπολογιστικού συστήματος αποθηκεύονται 8 λέξεις. Θεωρήστε τρεις κρυφές μνήμες των 512 πλαισίων με 8 λέξεις ανά πλαίσιο έκαστη και οργανώσεις αντίστοιχα:

a. άμεση
b. πλήρους συσχέτισης
c. 8–τρόπων συνόλου συσχέτισης

1. Να δώσετε τις διευθύνσεις των πλαισίων ή των συνόλων (στο οκταδικό) της κρυφής μνήμης στα οποία μπορούν να αποθηκευτούν τα περιεχόμενα των θέσεων της κύριας μνήμης με διευθύνσεις $117165_{(8)}$, $117167_{(8)}$, $445231_{(8)}$, $575232_{(8)}$, $675253_{(8)}$ και $677335_{(8)}$ –όλες οι προηγούμενες διευθύνσεις δίνονται στο οκταδικό σύστημα αρίθμησης– για κάθε μία από τις περιπτώσεις a, b, και c.
2. Ποια από τα περιεχόμενα των ανωτέρω θέσεων μνήμης είναι δυνατόν να βρίσκονται ταυτόχρονα στην κρυφή μνήμη σε κάθε μία από τις περιπτώσεις a, b, και c, και γιατί;

Λύση

1. Αφού κάθε πλαίσιο έχει $8 = 2^3$ λέξεις τα 3 λιγότερο σημαντικά δυαδικά ψηφία (ισοδύναμα, το λιγότερο σημαντικό οκταδικό ψηφίο) της διεύθυνσης θα καθορίζουν την θέση της λέξης μέσα στο μπλοκ της κύριας μνήμης καθώς και μέσα στο πλαίσιο της κρυφής μνήμης.

   a. <u>Άμεση οργάνωση</u>

   Αφού η κρυφή μνήμη έχει $512 = 2^9$ πλαίσια, τα αμέσως επόμενα 9 λιγότερο σημαντικά δυαδικά ψηφία της διεύθυνσης θα καθορίζουν την διεύθυνση του πλαισίου στην κρυφή μνήμη. Λαμβάνοντας υπόψη ότι κάθε ψηφίο στο οκταδικό είναι των 3 δυαδικών ψηφίων, κάθε διεύθυνση θα έχει τις εξής ερμηνείες για τη κύρια και τη κρυφή μνήμη

Διεύθυνση	6 οκταδικά ψηφία		
Ερμηνεία για τη κύρια μνήμη	5 οκταδικά ψηφία που καθορίζουν το μπλοκ		1 οκταδικό ψηφίο που καθορίζει τη λέξη του μπλοκ
Ερμηνεία για τη κρυφή μνήμη	2 οκταδικά ψηφία ετικέτας	3 οκταδικά ψηφία που καθορίζουν το πλαίσιο	1 οκταδικό ψηφίο που καθορίζει τη λέξη του πλαισίου

Διεύθυνση μνήμης	**Διεύθυνση πλαισίου της κρυφής μνήμης**
$117165_{(8)}$	$716_{(8)}$
$117167_{(8)}$	$716_{(8)}$
$445231_{(8)}$	$523_{(8)}$
$575232_{(8)}$	$523_{(8)}$
$675253_{(8)}$	$525_{(8)}$
$677335_{(8)}$	$733_{(8)}$

b. Οργάνωση πλήρους συσχέτισης.

Στην οργάνωση πλήρους συσχέτισης οποιοδήποτε μπλοκ της κύριας μνήμης μπορεί να μεταφερθεί σε οποιοδήποτε πλαίσιο της κρυφής μνήμης. Οι διευθύνσεις $117165_{(8)}$ και $117167_{(8)}$ ανήκουν στο ίδιο μπλοκ. Επομένως, μπορούν να μεταφερθούν σε οποιοδήποτε αλλά και οι δύο πάντα στο ίδιο πλαίσιο της κρυφής μνήμης. Κάθε μία από τις υπόλοιπες διευθύνσεις μπορεί να μεταφερθεί σε οποιοδήποτε πλαίσιο της κρυφής μνήμης.

c. 8–τρόπων συνόλου συσχέτισης.

Σε οργάνωση συνόλου συσχέτισης, δε μπορούμε να δώσουμε διεύθυνση πλαισίου, αφού αυτή εξαρτάται και από τον αλγόριθμο αντικατάστασης. Μπορούμε ωστόσο να καθορίσουμε τη διεύθυνση συνόλου. Σ' αυτή την περίπτωση έχουμε 8 πλαίσια ανά σύνολο και, επομένως, έχουμε 512 / 8 = 64 σύνολα. Αφού κάθε πλαίσιο έχει $8 = 2^3$ λέξεις τα 3 λιγότερο σημαντικά δυαδικά ψηφία της διεύθυνσης θα καθορίζουν και πάλι τη θέση της λέξης μέσα στο μπλοκ της κύριας μνήμης καθώς και μέσα στο πλαίσιο της κρυφής μνήμης. Αφού η κρυφή μνήμη έχει $64 = 2^6$ σύνολα τα αμέσως επόμενα 6 λιγότερο σημαντικά δυαδικά ψηφία της διεύθυνσης θα καθορίζουν την διεύθυνση του πλαισίου στην κρυφή μνήμη. Λαμβάνοντας υπόψη ότι κάθε ψηφίο στο οκταδικό είναι των 3 δυαδικών ψηφίων, κάθε διεύθυνση θα έχει τις εξής ερμηνείες για τη κύρια και τη κρυφή μνήμη

Διεύθυνση	6 οκταδικά ψηφία		
Ερμηνεία για τη κύρια μνήμη	5 οκταδικά ψηφία που καθορίζουν το μπλοκ		1 οκταδικό ψηφίο που καθορίζει τη λέξη του μπλοκ
Ερμηνεία για τη κρυφή μνήμη	3 οκταδικά ψηφία ετικέτας	2 οκταδικά ψηφία που καθορίζουν το σύνολο	1 οκταδικό ψηφίο που καθορίζει τη λέξη στα 2 πλαίσια του συνόλου

Διεύθυνση μνήμης	**Διεύθυνση συνόλου της κρυφής μνήμης**
$117165_{(8)}$	$16_{(8)}$
$117167_{(8)}$	$16_{(8)}$
$445231_{(8)}$	$23_{(8)}$

$575232_{(8)}$	$23_{(8)}$
$675253_{(8)}$	$25_{(8)}$
$677335_{(8)}$	$33_{(8)}$

2.

a. Άμεση οργάνωση

Οι διευθύνσεις $11–716–5_{(8)}$ και $11–716–7_{(8)}$ ανήκουν στο ίδιο μπλοκ. Επομένως, όταν βρίσκονται τα περιεχόμενα της μίας διεύθυνσης στην κρυφή μνήμη θα βρίσκονται και της άλλης.

Οι διευθύνσεις $44–523–1_{(8)}$ και $57–523–2_{(8)}$ ανήκουν σε διαφορετικά μπλοκ της κύριας μνήμης που αντιστοιχούν στο ίδιο πλαίσιο της κρυφής μνήμης (με διεύθυνση 523). Επομένως, τα περιεχόμενά τους δεν μπορούν να βρίσκονται ταυτόχρονα στην κρυφή μνήμη.

Οι διευθύνσεις $67–525–3_{(8)}$ και $67–733–5_{(8)}$ μεταξύ τους και σε σχέση με τις προηγούμενες 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.

b. Οργάνωση πλήρους συσχέτισης.

Αφού έχουμε μόνο 6 διευθύνσεις κύριας μνήμης και 512 πλαίσια, τα περιεχόμενα των διευθύνσεων αυτών μπορούν να βρίσκονται ταυτόχρονα στην κρυφή μνήμη.

c. 8–τρόπων συνόλου συσχέτισης.

Αφού έχουμε μόνο 6 διευθύνσεις κύριας μνήμης και 8 πλαίσια ανά σύνολο, τα περιεχόμενα των διευθύνσεων αυτών μπορούν να βρίσκονται ταυτόχρονα στην κρυφή μνήμη, ακόμη και αν όλα αντιστοιχούσαν στο ίδιο σύνολο (πράγμα που βέβαια δεν συμβαίνει).

ΑΣΚΗΣΗ 60

Θεωρείστε ότι η ΚΜΕ ενός υπολογιστή, που παράγει διευθύνσεις των 8 δυαδικών ψηφίων, παράγει την επόμενη ακολουθία διευθύνσεων: 5, 140, 20, 29, 3, 8, 65, 141, 83, 152. Ποιος είναι ο λόγος επιτυχίας της κρυφής μνήμης όταν η κρυφή μνήμη είναι άμεσης οργάνωσης και όταν η κρυφή μνήμη είναι 2-τρόπων συνόλου συσχέτισης λαμβάνοντας υπόψη ότι:

α. Η κρυφή μνήμη έχει χωρητικότητα 64 λέξεων.

β. Κάθε πλαίσιο της κρυφής μνήμης αποτελείται από 8 λέξεις.

Για κάθε μία οργάνωση που θα εξετάσετε να γράψετε δίπλα σε κάθε διεύθυνση εάν έχουμε "hit" ή "miss" και στη συνέχεια να δώσετε το τελικό περιεχόμενο της κρυφής μνήμης. Στο ξεκίνημα η κρυφή μνήμη δεν έχει έγκυρα περιεχόμενα. Η στρατηγική αντικατάστασης πλαισίου, όπου χρειάζεται, είναι η FIFO.

Λύση

Κρυφή μνήμη άμεσης οργάνωσης

Εφόσον η κρυφή μνήμη αποτελείται από 64 λέξεις και κάθε πλαίσιο αποτελείται από 8 λέξεις, υπάρχουν συνολικά 8 πλαίσια. Συνεπώς μία διεύθυνση των 8 ψηφίων απαιτεί τρία ψηφία για τον καθορισμό της θέσης λέξης μέσα στο πλαίσιο (μ=3), τρία ψηφία για τον καθορισμό του πλαισίου (κ=3) και 8 – 3 – 3 = 2 ψηφία για την ετικέτα.

Διεύθυνση μνήμης	Διεύθυνση σε δυαδική μορφή (ετικέτα – πλαίσιο – λέξη)		Π(διευθύνσεις μνήμης) => πλαίσιο με διεύθυνση*
5	00-000-101	Miss	Π(0…7) => 0
140	10-001-100	Miss	Π(136…143) => 1
20	00-010-100	Miss	Π(16…23) => 2
29	00-011-101	Miss	Π(24…31) => 3
3	00-000-011	Hit	
8	00-001-000	Miss	Π(8…15) => 1
65	01-000-001	Miss	Π(64…71) => 0
141	10-001-101	Miss	Π(136…143) => 1
83	01-010-011	Miss	Π(80…87) => 2
152	10-011-000	Miss	Π(152…159) => 3

Μετά από τη συγκεκριμένη ακολουθία διευθύνσεων, τα περιεχόμενα της κρυφής μνήμης είναι τα ακόλουθα:

Διεύθυνση πλαισίου	Διευθύνσεις θέσεων της κύριας μνήμης τα περιεχόμενα των οποίων βρίσκονται στο πλαίσιο
000	Π(64 … 71)
001	Π(136 … 143)
010	Π(80 … 87)
011	Π(152 … 159)
100	-
101	-
110	-
111	-

Η παύλα σημαίνει ότι το συγκεκριμένο πλαίσιο δεν έχει έγκυρα περιεχόμενα. Παρατηρούμε ότι από τις 10 αναφορές η 1 ήταν επιτυχημένη και οι υπόλοιπες αποτυχημένες. Άρα το ποσοστό επιτυχίας ήταν 1/10 =10%.

Οργάνωση 2-τρόπων συνόλου συσχέτισης

Σε αυτή την περίπτωση η κρυφή μνήμη αποτελείται από 4 σύνολα και 2 πλαίσια ανά σύνολο. Άρα λ=2 bits γιατί έχουμε τέσσερα σύνολα, μ=3, γιατί έχουμε 8 λέξεις ανά πλαίσιο. Τα υπόλοιπα τρία bits είναι η ετικέτα.

Διεύθυνση	Διεύθυνση σε		Π(διευθύνσεις	Π(διευθύνσεις μνήμης

* Π(διευθύνσεις μνήμης) => πλαίσιο με διεύθυνση : Ο συμβολισμός αυτός σημαίνει ότι η πληροφορία που είναι αποθηκευμένη στις θέσεις μνήμης που αναφέρονται μέσα στις παρενθέσεις θα μεταφερθεί στο πλαίσιο με τη διεύθυνση που δηλώνεται.

μνήμης	δυαδική μορφή (ετικέτα – σύνολο - λέξη)		μνήμης) => πλαίσιο 0 του συνόλου με διεύθυνση	) => πλαίσιο 1 του συνόλου με διεύθυνση
5	000-00-101	Miss	Π(0…7) => 0	
140	100-01-100	Miss	Π(136…143) => 1	
20	000-10-100	Miss	Π(16…23) => 2	
29	000-11-101	Miss	Π(24…31) => 3	
3	000-00-011	Hit		
8	000-01-000	Miss		Π(8…15) => 1
65	010-00-001	Miss		Π(64…71) => 0
141	100-01-101	Hit		
83	010-10-011	Miss		Π(80…87) => 2
152	100-11-000	Miss		Π(152…159) => 3

Τα τελικά περιεχόμενα της κρυφής μνήμης είναι τα ακόλουθα:

Σύνολο	Διευθύνσεις θέσεων της κύριας μνήμης τα περιεχόμενα των οποίων βρίσκονται στο Πλαίσιο 0	Διευθύνσεις θέσεων της κύριας μνήμης τα περιεχόμενα των οποίων βρίσκονται στο Πλαίσιο 1
00	Π(0…7)	Π(64…71)
01	Π(136…143)	Π(8…15)
10	Π(16…23)	Π(80…87)
11	Π(24…31)	Π(152…159)

Παρατηρούμε ότι από τις 10 αναφορές οι 2 ήταν επιτυχημένες και οι υπόλοιπες αποτυχημένες. Άρα το ποσοστό επιτυχίας ήταν 2/10 = 20%.

ΑΣΚΗΣΗ 61

Έστω δύο κρυφές μνήμες που καθεμία έχει συνολικά 8 πλαίσια. Η πρώτη κρυφή μνήμη έχει άμεση οργάνωση ενώ η δεύτερη δύο τρόπων συσχέτισης με First-In First-out στρατηγική απελευθέρωσης πλαισίων. Κάθε πλαίσιο μπορεί να αποθηκεύσει δύο bytes. Η κύρια μνήμη έχει 256 θέσεις του ενός byte.

α. Πως ερμηνεύεται η κάθε διεύθυνση που παράγεται από την κεντρική μονάδα επεξεργασίας στις δύο παραπάνω κρυφές μνήμες;
β. Ποιά από τις δύο κρυφές μνήμες αναμένεται να έχει τον μεγαλύτερο ρυθμό επιτυχιών (hit rate) και γιατί;
γ. Ποιό είναι το ποσοστό επιτυχίας της ακολουθίας διευθύνσεων 48, 39, 13, 138, 250, 139, 49, 167, 76, 251, 12, 38, 49, 77, 166, 48 και στις δύο κρυφές μνήμες; Δώστε τα περιεχόμενα των κρυφών μνημών μετά το πέρας της ακολουθίας. Οι διευθύνσεις είναι στο δεκαδικό σύστημα αρίθμησης.
δ. Δώστε μία δική σας ακολουθία διευθύνσεων (trace) για την οποία η κρυφή μνήμη με οργάνωση δύο τρόπων συσχέτισης έχει μικρότερο ρυθμό επιτυχίας.

Λύση

α.
Στην κρυφή μνήμη άμεσης οργάνωσης η μορφή των διευθύνσεων φαίνεται ακόλουθα:

Διεύθυνση	8 δυαδικά ψηφία		
Ερμηνεία για τη κύρια μνήμη	7 δυαδικά ψηφία που καθορίζουν το μπλοκ		1 δυαδικό ψηφίο που καθορίζει τη λέξη του μπλοκ
Ερμηνεία για τη κρυφή μνήμη	4 δυαδικά ψηφία ετικέτας	3 δυαδικά ψηφία που καθορίζουν το πλαίσιο	1 δυαδικό ψηφίο που καθορίζει τη λέξη του πλαισίου

Στην κρυφή μνήμη συνόλου συσχέτισης η μορφή των διευθύνσεων φαίνεται ακόλουθα:

Διεύθυνση	8 δυαδικά ψηφία		
Ερμηνεία για τη κύρια μνήμη	7 δυαδικά ψηφία που καθορίζουν το μπλοκ		1 δυαδικό ψηφίο που καθορίζει τη λέξη του μπλοκ
Ερμηνεία για τη κρυφή μνήμη	5 δυαδικά ψηφία ετικέτας	2 δυαδικά ψηφία που καθορίζουν το σύνολο	1 δυαδικό ψηφίο που καθορίζει τη λέξη του πλαισίου

β. Συνήθως όσο αυξάνονται οι τρόποι συσχέτισης τόσο αυξάνεται και ο ρυθμός επιτυχιών μιας κρυφής μνήμης. Και αυτό γιατί αυξάνοντας τους τρόπους συσχέτισης μειώνουμε τους περιορισμούς που αφορούν τα μπλοκ της κύριας που μπορούν να συνυπάρχουν στην κρυφή μνήμη. Π.χ. σε μια κρυφή μνήμη πλήρους συσχέτισης και Ν πλαισίων οποιαδήποτε Ν διαφορετικά μπλοκ μπορούν να συνυπάρχουν. Αντίθετα, σε μια κρυφή μνήμη άμεσης οργάνωσης, δύο μπλοκ που αναφέρονται στο ίδιο πλαίσιο δε μπορούν να συνυπάρχουν. Ωστόσο, όπως αποδεικνύεται και από το υποερώτημα (δ), ο ρυθμός επιτυχιών είναι συνάρτηση της συγκεκριμένης ακολουθίας προσπελάσεων και της μεθόδου αντικατάστασης. Οπότε είναι δυνατό μια κρυφή μνήμη με λιγότερους τρόπους συσχέτισης να οδηγήσει σε περισσότερες επιτυχίες από μια κρυφή μνήμη με περισσότερους τρόπους συσχέτισης. Και αυτό γιατί αυξάνοντας τους τρόπους συσχέτισης με σταθερό μέγεθος κρυφής μνήμης, μειώνεται ο αριθμός των συνόλων, οπότε αυξάνεται η πιθανότητα κάποια μπλοκ να ανταγωνίζονται για το ίδιο σύνολο.

γ. Στον ακόλουθο πίνακα φαίνονται οι ετικέτες που αποθηκεύονται σε κάθε πλαίσιο της κρυφής μνήμης άμεσης οργάνωσης μετά από κάθε αναφορά. Για παράδειγμα στην στήλη 0 της $1^{ης}$ γραμμής αναφορών (48) υπονοείται ότι στο πλαίσιο 0 της μνήμης ετικετών βρίσκεται αποθηκευμένη μετά την $1^{η}$ αναφορά η ετικέτα 0011. Οι σκιασμένες γραμμές δείχνουν τις επιτυχείς αναφορές στην κρυφή μνήμη.

	Διαχωρισμός της διεύθυνσης			***Περιεχόμενα των πλαισίων της μνήμης ετικετών***							
Αναφορά	***Ετικέτα***	***Πλαίσιο***	***Λέξη***	***0 000***	***1 001***	***2 010***	***3 011***	***4 100***	***5 101***	***6 110***	***7 111***
48	0011	000	0	0011							
39	0010	011	1	0011			0010				
13	0000	110	1	0011			0010			0000	
138	1000	101	0	0011			0010		1000	0000	
250	1111	101	0	0011			0010		1111	0000	
139	1000	101	1	0011			0010		1000	0000	
49	0011	000	1	0011			0010		1000	0000	
167	1010	011	1	0011			1010		1000	0000	
76	0100	110	0	0011			1010		1000	0100	
251	1111	101	1	0011			1010		1111	0100	
12	0000	110	0	0011			1010		1111	0000	
38	0010	011	0	0011			0010		1111	0000	

49	0011	000	1	0011			0010		1111	0000	
77	0100	110	1	0011			0010		1111	0100	
166	1010	011	0	0011			1010		1111	0100	
48	0011	000	0	0011			1010		1111	0100	

Το ποσοστό επιτυχίας των παραπάνω αναφορών είναι 3/16. Μετά το πέρας της ακολουθίας τα περιεχόμενα της κρυφής μνήμης ανά πλαίσιο είναι τα ακόλουθα

Πλαίσιο 0: διευθύνσεις 48, 49
Πλαίσιο 3: διευθύνσεις 166, 167
Πλαίσιο 5: διευθύνσεις 250, 251
Πλαίσιο 6: διευθύνσεις 76, 77
Λοιπά πλαίσια: κενά

Στον ακόλουθο πίνακα φαίνονται οι ετικέτες που αποθηκεύονται σε κάθε πλαίσιο της κρυφής μνήμης 2 τρόπων συσχέτισης μετά από κάθε αναφορά. Οι σκιασμένες γραμμές και πάλι δείχνουν τις επιτυχείς αναφορές στην κρυφή μνήμη.

				Περιεχόμενα μνήμης ετικετών							
	Διαχωρισμός της Διεύθυνσης			***Σύνολο 0 (00)***		***Σύνολο 1 (01)***		***Σύνολο 2 (10)***		***Σύνολο 3 (11)***	
Αναφορά	***Ετικέτα***	***Σύνολο***	***Λέξη***	***Πλαίσιο 0***	***Πλαίσιο 1***	***Πλαίσιο 0***	***Πλαίσιο 1***	***Πλαίσιο 0***	***Πλαίσιο 1***	***Πλαίσιο 0***	***Πλαίσιο 1***
48	00110	00	0	00110							
39	00100	11	1	00110						00100	
13	00001	10	1	00110				00001		00100	
138	10001	01	0	00110		10001		00001		00100	
250	11111	01	0	00110		10001	11111	00001		00100	
139	10001	01	1	00110		10001	11111	00001		00100	
49	00110	00	1	00110		10001	11111	00001		00100	
167	10100	11	1	00110		10001	11111	00001		00100	10100
76	01001	10	0	00110		10001	11111	00001	01001	00100	10100
251	11111	01	1	00110		10001	11111	00001	01001	00100	10100
12	00001	10	0	00110		10001	11111	00001	01001	00100	10100
38	00100	11	0	00110		10001	11111	00001	01001	00100	10100
49	00110	00	1	00110		10001	11111	00001	01001	00100	10100
77	01001	10	1	00110		10001	11111	00001	01001	00100	10100
166	10100	11	0	00110		10001	11111	00001	01001	00100	10100
48	00110	00	0	00110		10001	11111	00001	01001	00100	10100

Το ποσοστό επιτυχίας των παραπάνω αναφορών είναι 9/16. Μετά το πέρας της ακολουθίας τα περιεχόμενα της κρυφής μνήμης ανά πλαίσιο είναι τα ακόλουθα

Σύνολο 0, Πλαίσιο 0: διευθύνσεις 48, 49
Σύνολο 0, Πλαίσιο 1: κενό
Σύνολο 1, Πλαίσιο 0: διευθύνσεις 138, 139
Σύνολο 1, Πλαίσιο 1: διευθύνσεις 250, 251
Σύνολο 2, Πλαίσιο 0: διευθύνσεις 12, 13
Σύνολο 2, Πλαίσιο 1: διευθύνσεις 76, 77
Σύνολο 3, Πλαίσιο 0: διευθύνσεις 38, 39
Σύνολο 3, Πλαίσιο 1: διευθύνσεις 166, 167

δ. Έστω η ακολουθία διευθύνσεων 0, 8, 16, 0, 8, 16, ... Στην κρυφή μνήμη άμεσης οργάνωσης οι διευθύνσεις 0, 8 και 16 αντιστοιχούν στα πλαίσια 0, 4 και 0 αντίστοιχα. Άρα οι

0 και 16 δε θα μπορέσουν να συνυπάρχουν και οι σχετικές προσπελάσεις θα είναι πάντα ανεπιτυχείς. Μετά την πρώτη προσπέλαση της διεύθυνσης 8, οι υπόλοιπες προσπελάσεις της διεύθυνσης 8 θα είναι επιτυχείς. Οπότε, αγνοώντας τις πρώτες τρεις προσπελάσεις ο ρυθμός επιτυχίας θα είναι 1/3. Στην κρυφή μνήμη δύο τρόπων συσχέτισης οι διευθύνσεις αντιστοιχούν όλες στο σύνολο 0. Επειδή υπάρχει χώρος μόνο για δύο πλαίσια, οι διευθύνσεις εμφανίζονται η μια μετά την άλλη και η μέθοδος αντικατάστασης είναι FIFO, δεν θα έχουμε ποτέ επιτυχία (ρυθμός επιτυχίας 0).

Είσοδος / Έξοδος

ΑΣΚΗΣΗ 62

Υποθέστε ότι έχουμε ένα σύστημα μνήμης που χρησιμοποιεί τη συχνότητα 50MHz και λέξεις των 4 ψηφιολέξεων (bytes).
Κατά την ανάγνωση της μνήμης η προσπέλαση γίνεται ως εξής:
1 κύκλος για να πάρει τη διεύθυνση
3 κύκλοι αναμονής και
8 κύκλοι για να στείλει 8 λέξεις, μία κάθε κύκλο.
Κατά την εγγραφή στη μνήμη η προσπέλαση γίνεται ως εξής:
1 κύκλος για να πάρει τη διεύθυνση
2 κύκλοι αναμονής,
8 κύκλοι για να λάβει 8 λέξεις, μία κάθε κύκλο, και
3 κύκλους για να υπολογίσει και να γράψει τα δυαδικά ψηφία ενός κώδικα ανίχνευσης λαθών.

Να υπολογίσετε τον μέγιστο ρυθμό μεταφοράς δεδομένων σε Mbytes ανά δευτερόλεπτο για μία σειρά προσπελάσεων της κύριας μνήμης που αποτελείται:

α. Μόνο από απαιτήσεις ανάγνωσης της μνήμης.
β. Μόνο από απαιτήσεις εγγραφής στη μνήμη.
γ. Ένα μείγμα αποτελούμενο κατά το 60% από απαιτήσεις ανάγνωσης και κατά το 40% από απαιτήσεις εγγραφής στη μνήμη.

Λύση:
α. Για την ανάγνωση 8 λέξεων = 4 * 8 bytes = 32 bytes απαιτούνται 1 + 3 + 8 = 12 κύκλοι. Κάθε κύκλος έχει διάρκεια $1/(50\ 10^6)$ = 20 nsec, άρα για την ανάγνωση 32 bytes απαιτούνται 240 nsec.
Άρα $32 * 10^{-6}$ Mbytes διαβάζονται σε 240 nsec = $240*10^{-9}$ sec, σε 1 sec θα διαβάζονται $(32 * 10^{-6}$ Mbytes$)/ 240*10^{-9} = 0.13333 * 10^3$ Mbytes = 133.33 Mbytes
Συνεπώς ο ρυθμός ανάγνωσης είναι Ραν = 133.33 Mbytes/sec.

β. Για την εγγραφή 8 λέξεων = 4 * 8 bytes = 32 bytes απαιτούνται 1 + 2 + 8 + 3 = 14 κύκλοι. Κάθε κύκλος έχει διάρκεια 20 nsec, άρα για την εγγραφή 32 bytes απαιτούνται 280 nsec.
Άρα $32 * 10^{-6}$ Mbytes γράφονται σε 280 nsec = $280*10^{-9}$ sec, σε 1 sec θα γράφονται $(32 * 10^{-6}$ Mbytes$)/ 280*10^{-9} = 0.11429 * 10^3$ Mbytes = 114.29 Mbytes
Συνεπώς ο ρυθμός εγγραφής να είναι Ρεγ = 114.29 Mbytes/sec.

γ. Έστω Χ αιτήσεις συνολικά. Εξαιτίας του μείγματος 60%-40% αιτήσεων ανάγνωσης και εγγραφής, θα έχουμε 0.6 Χ αιτήσεις ανάγνωσης και 0.4 Χ αιτήσεις εγγραφής.

Ο συνολικός αριθμός των bytes είναι 0.6 Χ 32 + 0.4 Χ 32 = 32 Χ.
Ο συνολικός χρόνος είναι 0.6 * Χ *12 * 20 + 0.4 * Χ * 14 * 20 = 256 * Χ nsec.
Άρα ο ρυθμός μεταφοράς είναι 32Χ/(256Χ)=1/8= 0.125 bytes/nsec= 125 Mbytes/sec

Προσοχή:
Η σχέση
Ρμετ = 0.6 *133.33 Mbytes/sec + 0.4 *114.29 Mbytes/sec = 125.71 Mbytes/sec

δεν ισχύει διότι τα ποσοστά 60% και 40% αναφέρονται σε αιτήσεις ανάγνωσης και εγγραφής αντίστοιχα και όχι σε ποσοστά χρόνου.

ΑΣΚΗΣΗ 63

Υποθέστε ότι έχουμε ένα σύστημα μνήμης που χρησιμοποιεί τη συχνότητα 100MHz και λέξεις των 2 ψηφιολέξεων (bytes).
Κατά την ανάγνωση της μνήμης η προσπέλαση γίνεται ως εξής:
1 κύκλος για να πάρει τη διεύθυνση
1 κύκλος αναμονής και
8 κύκλοι για να στείλει 4 λέξεις.
Κατά την εγγραφή στη μνήμη η προσπέλαση γίνεται ως εξής:
1 κύκλος για να πάρει τη διεύθυνση
2 κύκλοι αναμονής,
8 κύκλοι για να λάβει 4 λέξεις, και
1 κύκλο για να υπολογίσει και να γράψει τα δυαδικά ψηφία ενός κώδικα ανίχνευσης λαθών.

α. Να υπολογίσετε τον μέγιστο ρυθμό μεταφοράς δεδομένων σε Mbytes ανά δευτερόλεπτο για μία σειρά προσπελάσεων της κύριας μνήμης που αποτελείται:
Μόνο από απαιτήσεις ανάγνωσης της μνήμης.
Μόνο από απαιτήσεις εγγραφής στη μνήμη.
Ένα μείγμα αποτελούμενο κατά το 80% από απαιτήσεις ανάγνωσης και κατά το 20% από απαιτήσεις εγγραφής στη μνήμη.
β. Να υπολογίσετε τον μέγιστο ρυθμό μετακίνησης δεδομένων που βρίσκονται αποθηκευμένα σε διαδοχικές θέσεις από μία περιοχή της μνήμης σε μία άλλη. Εκφράστε το αποτέλεσμα σε Mbytes/sec.

Παρατήρηση: Όταν αναφερόμαστε στην χωρητικότητα μνημών 1Mbyte=2^{20} bytes ενώ όταν αναφερόμαστε στην μεταφορά πληροφορίας 1Mbyte=10^6 bytes. Προσοχή οι δύο τιμές είναι διαφορετικές.

Λύση:
Ερώτημα (α)
i. Για την ανάγνωση 4 λέξεων = 4 * 2 bytes = 8 bytes απαιτούνται 1 + 1 + 8 = 10 κύκλοι. Κάθε κύκλος έχει διάρκεια 1/(100 10^6) = 10 nsec, άρα για την ανάγνωση 8 bytes απαιτούνται 100 nsec.
Άρα 8 * 10^{-6} Mbytes διαβάζονται σε 100 nsec = 100*10^{-9} sec, σε 1 sec θα διαβάζονται (8 * 10^{-6} Mbytes)/ (100*10^{-9})=8 * 10^1 Mbytes = 80 Mbytes
Συνεπώς ο ρυθμός ανάγνωσης είναι $P_{αν}$ = 80 Mbytes/sec.

ii. Για την εγγραφή 4 λέξεων = 4 * 2 bytes = 8 bytes απαιτούνται 1 + 2 + 8 + 1 = 12 κύκλοι. Κάθε κύκλος έχει διάρκεια 10 nsec, άρα για την εγγραφή 8 bytes απαιτούνται 120 nsec.
Άρα 8 * 10^{-6} Mbytes γράφονται σε 120 nsec = 120*10^{-9} sec, σε 1 sec θα γράφονται (8 * 10^{-6} Mbytes)/ 120*10^{-9} =0.06666 * 10^3 Mbytes = 66.66 Mbytes
Συνεπώς ο ρυθμός εγγραφής να είναι $P_{εγ}$ = 66.66 Mbytes/sec.

iii.Έστω X αιτήσεις συνολικά. Εξαιτίας του μείγματος 80%-20% αιτήσεων ανάγνωσης και εγγραφής, θα έχουμε 0.8 X αιτήσεις ανάγνωσης και 0.2 X αιτήσεις εγγραφής.

Ο συνολικός αριθμός των bytes είναι 0.8 X 8 + 0.2 X 8 = 8 X.
Ο συνολικός χρόνος είναι 0.8 * X *10 * 10 + 0.2 * X * 12 * 10 = 104 * X nsec.
Άρα ο ρυθμός μεταφοράς είναι 8 X / (104 X) = 1/8 = 0.076923 bytes/nsec= 76,923 Mbytes/sec

Προσοχή.
Η σχέση
$P_{μετ}$ = 0.8 *80 Mbytes/sec + 0.2 *66.66 Mbytes/sec = 77.332 Mbytes/sec

δεν ισχύει διότι τα ποσοστά 80% και 20% αναφέρονται σε αιτήσεις ανάγνωσης και εγγραφής αντίστοιχα και όχι σε ποσοστά χρόνου.

Ερώτημα (β)
Κάθε μετακίνηση μίας λέξης απαιτεί μία ανάγνωση από την θέση στην οποία βρίσκεται στη μνήμη και κατόπιν εγγραφή στη θέση στην οποία πρέπει να τοποθετηθεί. Το σύστημά μας μεταφέρει κάθε φορά 4 λέξεις των 2 bytes η κάθε μία και απαιτούνται 10 κύκλοι για την ανάγνωση τους και άλλοι 12 κύκλοι για την εγγραφή τους στην νέα θέση τους. Άρα για την μετακίνηση 8 bytes απαιτούνται 22 κύκλοι δηλαδή χρόνος ίσος με 220nsec. Σε 1 sec μπορούν να μετακινηθούν $8/(220 \times 10^{-9})$ bytes ή ισοδύναμα 36.36 Mbytes. Άρα ο ρυθμός μετακίνησης δεδομένων είναι ίσος με:
$P_{μετακ}$ = 36.36 Mbytes/sec

ΑΣΚΗΣΗ 64

Μία μονάδα εισόδου/εξόδου μεταφέρει δεδομένα στην μνήμη του επεξεργαστή με ρυθμό 10 Mbytes / sec μέσω του διαύλου εισόδου/εξόδου που έχει συνολικό bandwidth 100 Mbytes / sec. Τα 10 MBytes μεταφέρονται σαν 2500 ανεξάρτητες σελίδες των 4 KByte η κάθε μία. Ο επεξεργαστής λειτουργεί με συχνότητα 200 MHz. Για την μεταφορά των δεδομένων με την τεχνική DMA απαιτούνται 1000 κύκλοι για την έναρξη της μεταφοράς και 1500 κύκλοι για την απόκριση στο σήμα διακοπής του τέλους της μεταφοράς. Ποιος είναι ο χρόνος που απασχολείται η CPU για την μεταφορά των δεδομένων χωρίς DMA και με την χρήση DMA;

Λύση

- Χωρίς την τεχνική DMA, ο επεξεργαστής πρέπει να αποθηκεύει τα δεδομένα στη μνήμη καθώς η μονάδα εισόδου/εξόδου τα τοποθετεί στο δίαυλο. Καθώς η μονάδα εισόδου εξόδου στέλνει δεδομένα με ρυθμό 10 MBytes / sec μέσω του διαύλου, ο οποίος έχει ρυθμό μεταφοράς 100 MBytes / sec, 10% του χρόνου δαπανάται για την μεταφορά δεδομένων στο δίαυλο. Αρα ο επεξεργαστής είναι απασχολημένος με τη μεταφορά των δεδομένων στο 10% του χρόνου του.
- Με τη χρήση DMA, η μεταφορά 1 σελίδας των 4KB, επιβαρύνει τον επεξεργαστή αφενός για να αρχικοποιήσει τον ελεγκτή DMA και αφετέρου για να εξυπηρετήσει τη διακοπή στο τέλος της μεταφοράς. Συνολικά κάθε μία τέτοια μεταφορά "κλέβει" από τον επεξεργαστή 1000 + 1500 κύκλους = 2500 κύκλους, που αλλιώς θα χρησιμοποιούνταν για υπολογισμούς. Η μεταφορά συνεπώς 10 MBytes σε διάστημα 1 sec, θα στερήσει από τον επεξεργαστή 2500 x 2500 = 6250000 υπολογιστικούς κύκλους. Αφού η συχνότητα του επεξεργαστή είναι 200 MHz, στο 1 sec, μπορούν να υπάρχουν 200 x 10^6 κύκλοι. Αρα η επιβάρυνση είναι 6250000 / 200 x 10^6 = 0,03125 = 3,125%.

Προσπέλαση Μονάδων Δίσκου

ΑΣΚΗΣΗ 65

Να σχολιάσετε το χρόνο αναζήτησης και το χρόνο αναμονής σε κάθε μία από τις επόμενες περιπτώσεις σε σχέση με τον αντίστοιχο χρόνο σε μία μονάδα μαγνητικών δίσκων με μία κεφαλή ανάγνωσης/εγγραφής ανά επιφάνεια.
α. Υπάρχουν δύο κεφαλές ανάγνωσης/εγγραφής ανά επιφάνεια, μία στον ομόκεντρο κύκλο 1 και μία στον ν/2, όπου ν είναι το πλήθος των ομόκεντρων κύκλων ανά επιφάνεια.
β. Υπάρχουν ν κεφαλές ανάγνωσης/εγγραφής ανά επιφάνεια, μία για κάθε ομόκεντρο κύκλο.

Λύση:
Ο χρόνος αναμονής είναι σε κάθε περίπτωση ο ίδιος, γιατί εξαρτάται από το σημείο επί του ομόκεντρου κύκλου από το οποίο αρχίζουν τα δεδομένα.

Στην περίπτωση (α) ο χρόνος αναζήτησης μειώνεται ως εξής.
Ενώ στην περίπτωση της μίας κεφαλής ο μέγιστος χρόνος αναζήτησης απαιτεί τη μετακίνηση της κεφαλής από τον κύκλο 1 στον ν, στην περίπτωση των δύο κεφαλών, η μετακίνηση περιορίζεται από τον κύκλο 1 στον ν/2 – 1 και από τον ν/2 – 1 στον ν, για κάθε μία κεφαλή. Άρα ο μέγιστος χρόνος αναζήτησης γίνεται περίπου μισός.

Στην (β) περίπτωση ο χρόνος αναζήτησης μηδενίζεται γιατί υπάρχει κεφαλή στον ομόκεντρο κύκλο, ο οποίος περιλαμβάνει τα δεδομένα.

ΑΣΚΗΣΗ 66

α. Υπολογίστε τον μέσο χρόνο ανάγνωσης ή εγγραφής ενός τμήματος 512 ψηφιολέξεων (bytes) ενός σκληρού δίσκου που περιστρέφεται με 5400 στροφές το λεπτό. Ο μέσος χρόνος αναζήτησης είναι 12 ms, ο ρυθμός μεταφοράς είναι 5 MBytes/sec και η καθυστέρηση που βάζει ο ελεγκτής του δίσκου είναι 2 ms. Οταν αναφερόμαστε σε ρυθμό μεταφοράς δεδομένων, ισχύει 1MByte = 10^6 bytes.
β. Πως θα μπορούσαμε να ελαττώσουμε στο μισό τον χρόνο αναμονής;

Λύση :
α. *Μέσος χρόνος ανάγνωσης = καθυστέρηση που βάζει ο ελεγκτής του δίσκου +*

μέσος χρόνος αναζήτησης + μέσος χρόνος αναμονής + χρόνος μεταφοράς.

Ο μέσος χρόνος αναμονής ισούται με τον χρόνο που απαιτείται για μισή περιστροφή του δίσκου, επομένως

μέσος χρόνος αναμονής = (0,5 στροφή) / (5400 στροφές ανά λεπτό) =
(0,5 x 60 x 10^3 ms) / 5400 = 5,6 ms.

χρόνος μεταφοράς = (512 bytes) / (5 MBytes/sec) =
= (10^3 ms x 512 bytes) / (5 x 10^6 bytes) = 0,1 ms

Επομένως
Μέσος χρόνος ανάγνωσης = 2 ms + 12 ms + 5,6 ms + 0,1 ms = 19,7 ms.
β. Χρησιμοποιώντας δύο κεφαλές τοποθετημένες αντιδιαμετρικά.

ΑΣΚΗΣΗ 67

Ένας σκληρός δίσκος διαθέτει μία κεφαλή ανάγνωσης / εγγραφής και περιστρέφεται με 15000 στροφές / λεπτό. Ο δίσκος έχει 1024 ομόκεντρους κύκλους (tracks) που ο καθένας έχει 2048 τομείς (sectors). Ο χρόνος αναζήτησης είναι 2 ms για κάθε 100 ομόκεντρους κύκλους που πρέπει να περάσει η κεφαλή. Ας υποθέσουμε ότι η κεφαλή του δίσκου βρίσκεται στον ομόκεντρο κύκλο 0 ο οποίος χρησιμοποιείται για την τοποθέτηση της κεφαλής και όχι για την εγγραφή δεδομένων. Έστω ότι ο δίσκος παίρνει εντολή να προσπελάσει δεδομένα σε έναν τυχαίο τομέα και σε ένα τυχαίο ομόκεντρο κύκλο. Σε αυτή την περίπτωση:

α) Ποιος είναι ο μέσος χρόνος αναζήτησης ?
β) Ποιος είναι ο μέσος χρόνος αναμονής ?
γ) Ποιος είναι ο χρόνος ανάγνωσης από τη κεφαλή των δεδομένων ενός τομέα ?

Λύση

α) Αφού η κεφαλή είναι στον ομόκεντρο κύκλο 0, πρέπει να διέλθει 1 ομόκεντρο κύκλο για να πάει στον 1, 2 ομόκεντρους κύκλους για να πάει στον 2, …. και 1023 ομόκεντρους κύκλους για να πάει στον 1023. Άρα ο μέσος αριθμός των ομόκεντρων κύκλων που πρέπει να διέλθει η κεφαλή για να προσπελάσει έναν τυχαίο ομόκεντρο κύκλο είναι

$$\frac{1+2+...+1023}{1023} = 512$$

Εφ' όσον ο χρόνος αναζήτησης είναι 2 ms για 100 ομόκεντρους κύκλους ο ζητούμενος μέσος χρόνος αναζήτησης είναι (512/100)* 2 ms = 10.24 ms.

β) Εφ' όσον ο δίσκος περιστρέφεται με 15000 στροφές/min κάθε περιστροφή διαρκεί 4ms. Ο μέσος χρόνος αναμονής θα είναι το ήμισυ του χρόνου περιστροφής, δηλαδή 2ms.

γ) Κάθε περιστροφή διαρκεί 4 ms. Κάθε ομόκεντρος κύκλος περιλαμβάνει 2048 τομείς. Η διέλευση κάθε τομέα κάτω από την κεφαλή εγγραφής / ανάγνωσης διαρκεί 4 ms / 2048=1.95 μs. Άρα ο χρόνος ανάγνωσης από τη κεφαλή των δεδομένων ενός τομέα είναι 1.95 μs.

ΑΣΚΗΣΗ 68

Ένας δίσκος χωρητικότητας 40 GBytes, έχει 131072 ομόκεντρους κύκλους (tracks), οι τομείς (sectors) του έχουν μέγεθος 512 Byte, περιστρέφεται με 3600 rpm και έχει μέσο χρόνο αναζήτησης 10 ms. Για τη βελτίωση του μέσου χρόνου προσπέλασης, ο δίσκος διαθέτει κρυφή μνήμη, που αποθηκεύει ολόκληρα tracks του δίσκου. Υπολογίστε τη βελτίωση του μέσου χρόνου προσπέλασης όταν ο λόγος επιτυχίας της κρυφής μνήμης είναι h = 0,9.
Δίνεται ότι :

Βελτίωση = Χρόνος προσπέλασης $_{\text{Πριν την Βελτίωση}}$ / Χρόνος προσπέλασης $_{\text{Μετά την Βελτίωση}}$.

Λύση

- Οταν δε χρησιμοποιείται κρυφή μνήμη, ο μέσος χρόνος προσπέλασης ισούται με $t = t_1 + t_2$, όπου :
  - t_1 είναι ο μέσος χρόνος αναζήτησης, δηλαδή τοποθέτησης της κεφαλής στο track, που μας δίνεται ίσος με 10 ms και
  - t_2 είναι ο χρόνος αναμονής, δηλαδή ο χρόνος εύρεσης της αρχής του συγκεκριμένου sector που περιέχει τη ζητούμενη πληροφορία, που είναι ίσος με το χρόνο μισής περιστροφής. Αυτή ολοκληρώνεται σε $t_2 = (60 \times 1000) / (3600 \times 2) = 8,33$ ms.

  Αρα ο μέσος χρόνος προσπέλασης εάν δεν υπήρχε η κρυφή μνήμη είναι $t = t_1 + t_2 = 18,33$ ms.

- Με την κρυφή μνήμη ο μέσος χρόνος προσπέλασης είναι $t' = h*t_{cache} + (1–h)*t_{track}$, όπου :
  - t_{cache} είναι ο χρόνος προσπέλασης της κρυφής μνήμης. Επειδή ο χρόνος προσπέλασης της κρυφής μνήμης, t_{cache}, είναι τάξεις μεγέθους μικρότερος από αυτόν της λειτουργίας του δίσκου ο παράγοντας $h*t_{cache}$ μπορεί να αγνοηθεί στον υπολογισμό του t', και
  - t_{track} είναι ο χρόνος μεταφοράς της πληροφορίας ενός ομόκεντρου κύκλου (track) στην κρυφή μνήμη. $t_{track}= t_1 + t_3 + t_4$, όπου
    - t_3 είναι ο χρόνος ανάγνωσης της πληροφορίας ενός track και ισούται με τον χρόνο μίας πλήρους περιστροφής του δίσκου, δηλαδή $t_3 = (60 \times 1000) / 3600 = 16,67$ ms και
    - t_4 είναι ο μέσος χρόνος εύρεσης της αρχής κάποιου τυχαίου sector, που ισούται με το χρόνο περιστροφής κατά μισό sector. Αφού σε κάθε track υπάρχουν $40 * 2^{30}$ Bytes / (131072 * 512 Bytes / sector) = 640 sectors, για τη περιστροφή κατά μισό sector χρειάζονται $(60 * 1000) / (3600 * 640 * 2) \approx 0,013$ ms

  Αρα ο μέσος χρόνος προσπέλασης στη περίπτωση που χρησιμοποιείται η κρυφή μνήμη είναι περίπου $t' \approx (1–h) \times (t_1 + t_3 + t_4) \approx 2,668$ ms.

Συνεπώς η βελτίωση που επιτυγχάνεται με τη χρήση της κρυφής μνήμης είναι $t / t' = 18,33 / 2,668 \approx 6,87$.

ΑΣΚΗΣΗ 69

Ένας σκληρός δίσκος διαθέτει 2 κεφαλές ανάγνωσης/εγγραφής (μία για κάθε επιφάνεια του δίσκου) και περιστρέφεται με r=10000 στροφές/λεπτό. Ο μέσος χρόνος αναζήτησης είναι ts=10 ms. Ο δίσκος έχει Nt=1024 ομόκεντρους κύκλους ή ίχνη (tracks) που ο καθένας έχει Ns = 1024 τομείς (sectors). Κάθε τομέας έχει NB=512 bytes.

α) Ποια είναι η χωρητικότητα του σκληρού δίσκου;

β) Ποιος είναι ο μέσος χρόνος αναμονής;

γ) Ποιος είναι ο μέσος χρόνος προσπέλασης;

δ) Ποιος είναι ο χρόνος ανάγνωσης των δεδομένων ενός τομέα; Θεωρήστε ότι είναι ίσος με το χρόνο διέλευσης του τομέα κάτω από την κεφαλή.

Λύση

α) Η χωρητικότητα υπολογίζεται εύκολα ως εξής

$C = 2 \times N_t \times N_s \times N_b = 2 \times 1024 \times 1024 \times 512 = 1GB$

β) Ο μέσος χρόνος αναμονής είναι ο χρόνος που απαιτείται για μισή περιστροφή

$t_w = 1/(2r) = 3ms$

γ) $t_{acc} = t_s + t_w = 13ms$

δ) Ο χρόνος περιστροφής είναι $t_r = 1/r = 6ms$ και ο χρόνος διέλευσης ενός τομέα

$t_r/N_s = 6ms/1024 = 5,8\ \mu s$