ΑΛΕΞΑΝΔΡΟΣ ΜΑΡΑΝΤΟΣ 3329

ΣΕΙΡΑ ΑΣΚΗΣΕΩΝ 12

Παραδειγμα (εξωπραγματικου) συστηματος εικονικης μνημης.

Μεγεθος εικονικων διευθυνσεων: 20 bits (5 ψηφια)

Χωρος λοιπον εικονικων διευθυνσεων: 1 Mbyte / Διεργασια

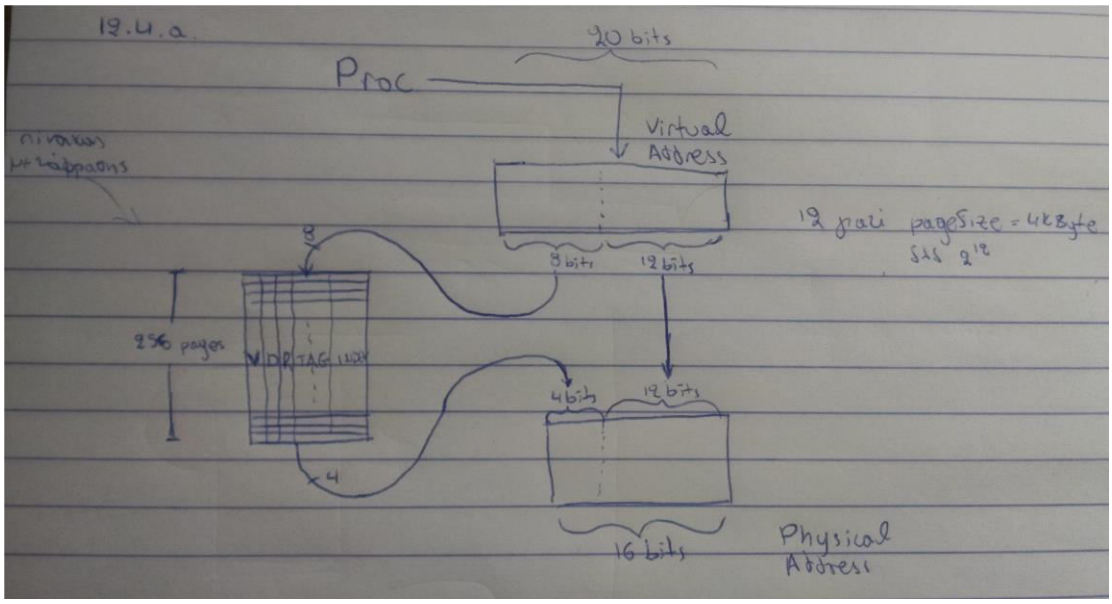
Μεγεθος σελιδας: 4 KBytes. Αρα 256 σελιδες / διεργασια

Φυσικη μνημη: 64 KBytes Αρα οι φυσικες διευθυνσεις αποτελουνται από 16 bits

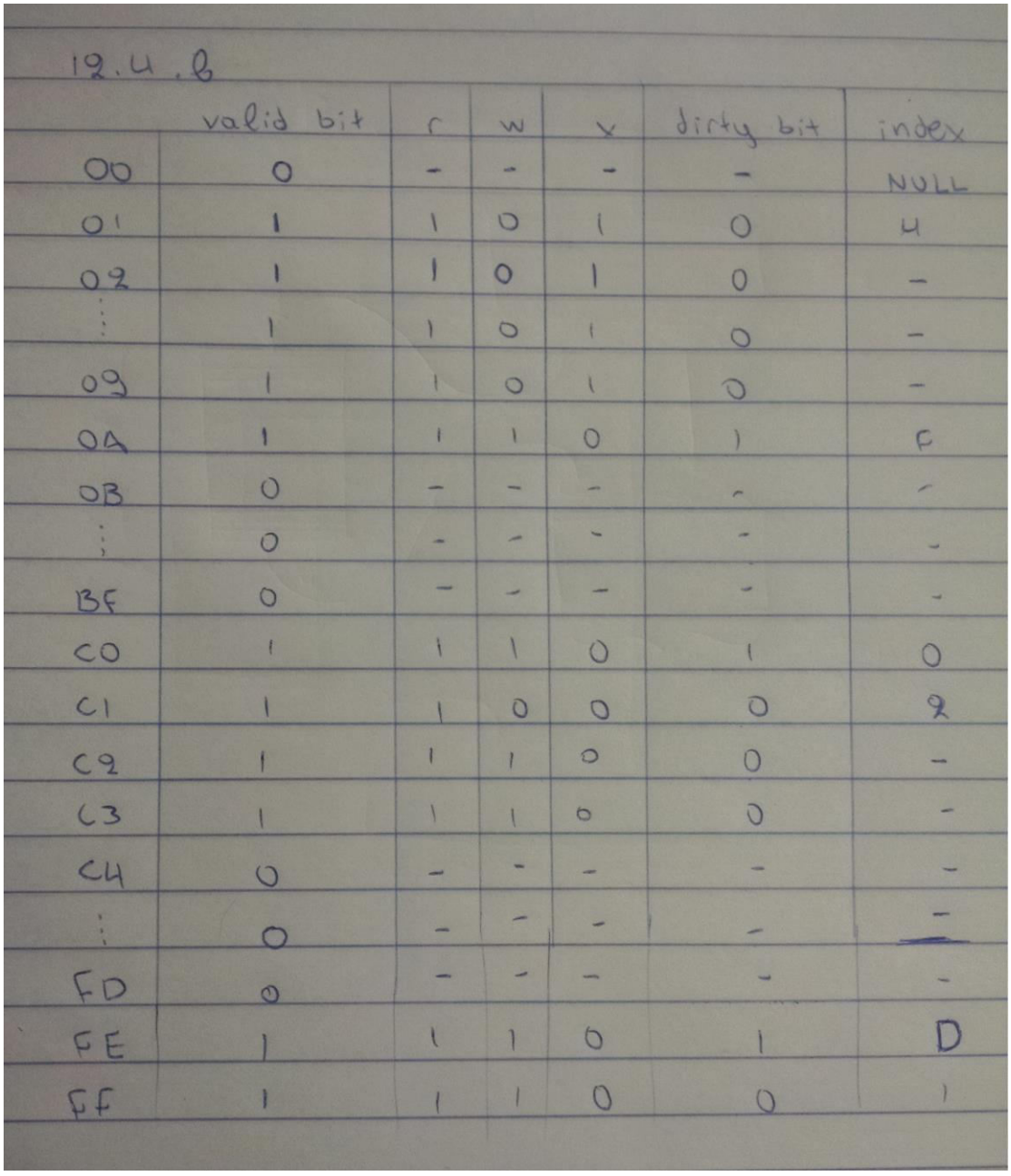
Και αρα αφου κάθε σελιδα 4 KBytes, υπαρχουν 16 φυσικες σελιδες σε κάθε φυσικη μνημη.

ΑΣΚΗΣΗ 12.4

Α)



Β)



12.4.β

	valid bit	r	w	x	dirty bit	index
00	0	-	-	-	-	NULL
01	1	1	0	1	0	4
02	1	1	0	1	0	-
⋮	1	1	0	1	0	-
09	1	1	0	1	0	-
0A	1	1	1	0	1	F
0B	0	-	-	-	-	-
⋮	0	-	-	-	-	-
BF	0	-	-	-	-	-
C0	1	1	1	0	1	0
C1	1	1	0	0	0	2
C2	1	1	1	0	0	-
C3	1	1	1	0	0	-
C4	0	-	-	-	-	-
⋮	0	-	-	-	-	-
FD	0	-	-	-	-	-
FE	1	1	1	0	1	D
FF	1	1	1	0	0	1

Γ)

Από τα 5 ψηφια της διευθυνσης που δινεται κάθε φορα, μας ενδιαφερουν 2 πρωτα καθως τα 3 τελευταια είναι τα 12 bits που δεν αλλαζουν.

Οποτε από τον παραπανω πινακα:

01038 (fetch) - Φυσικη διευθυνση 4038 (το 038 δεν αλλαζει όπως προειπα)

0B0F4 (read) - Παρανομη εντολη, προωρος τερματισμος

C001C (write) - Φυσικη διευθυνση 001C

0292C (fetch) - Απουσα σελιδα, επανεκκινηση

00000 (read) - Παρανομη εντολη, προωρος τερματισμος

99F88 (read) - Παρανομη εντολη, προωρος τερματισμος

FE5D8 (write) - Φυσικη διευθυνση D5D8

FF100 (fetch) - Παρανομη εντολη, για να γινει fetch πρεπει να εχει το x

C20CC (write) - Απουσα σελιδα, επανεκκινηση

CD0CC (write) - Παρανομη εντολη, προωρος τερματισμος

C0444 (read) - Φυσικη διευθυνση 0444

01FF4 (fetch) - Φυσικη διευθυνση 4FF4

C1FFC (write) - Παρανομη εντολη, προωρος τερματισμος (δεν επιτρεπεται το w)

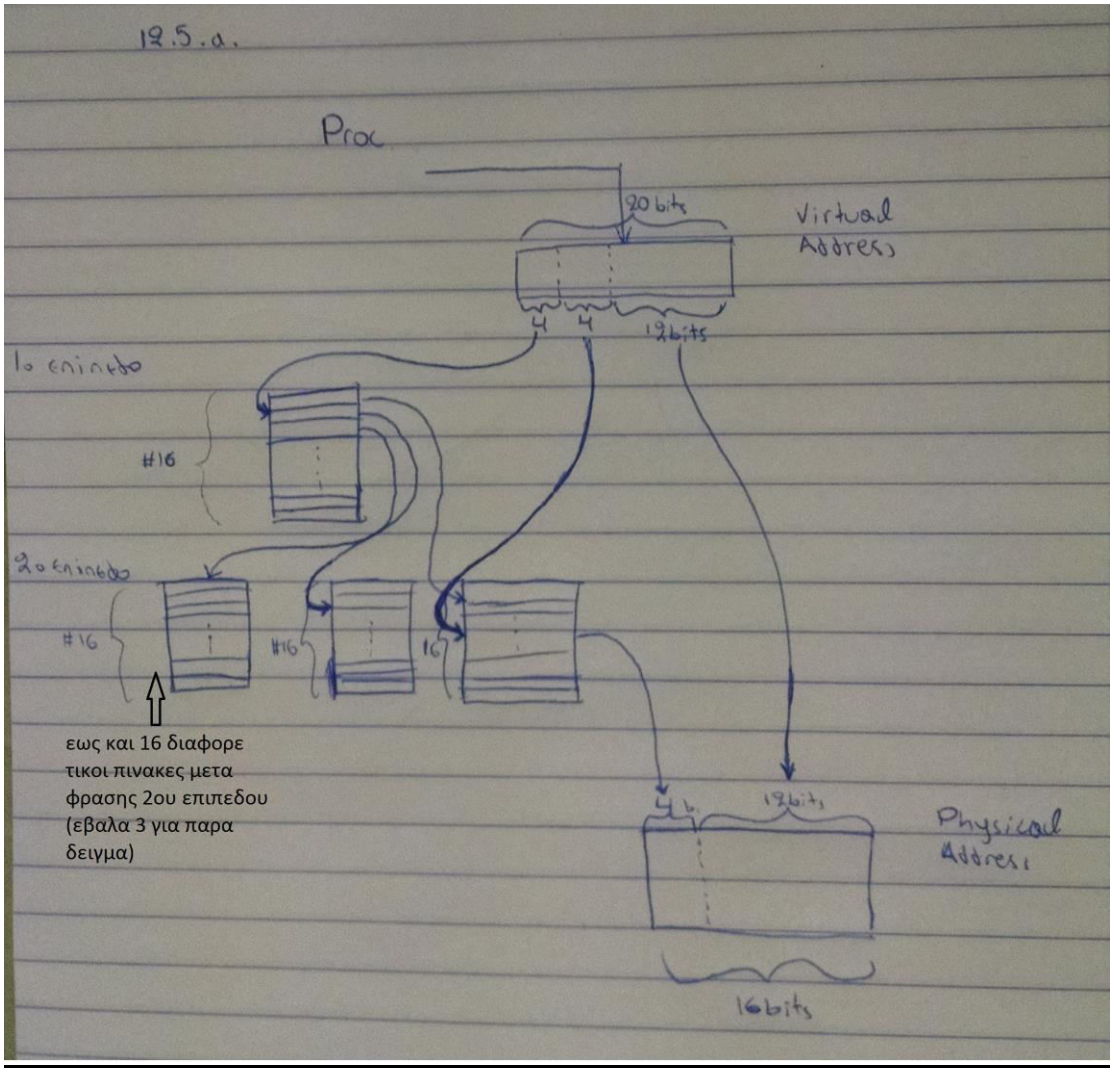
008E4 (write) - Παρανομη εντολη, προωρος τερματισμος

C7700 (read) - Παρανομη εντολη, προωρος τερματισμος

01E40 (write) - Παρανομη εντολη, προωρος τερματισμος (δεν επιτρεπεται το w)

ΑΣΚΗΣΗ 12.5

ΑΒΓ)



Δ)

Στην ασκηση 4 ειχαμε έναν πινακα μεταφρασης 256 σελιδων με κάθε σελιδα 4 Kb

Αρα χρειαζοταν 1Mb συνολικα (256 * 4)

Στην παρουσα κατασταση εχουμε έναν πινακα μεταφρασης 1^{ου} επιπεδου 16 σελιδων (με κάθε σελιδα 4 Kb) Αρα 16*4 = 64Kb

Ενώ εχουμε και πινακες 2^{ου} επιπεδου, στην παρουσα περιπτωση 5 αφου 5 διευθυνσεις περασαν εγκυρα στην φυσικη μνημη.

Αρα 5 (πινακες) * 16 (pages) * (4 Kb η καθεμια) = 320 Kb

ΑΡΑ 64 Kb + 320 Kb = 384 Kb.

Υπαρχει λοιπον σημαντικη μειωση (οικονομια) στις θεσεις μνημης που καταλαμβανονται στην παρουσα ασκηση σε σχεση με την προηγουμενη.

ΑΣΚΗΣΗ 12.6

Πλεον το συστημα εικονικης μνημης θα εχει τα εξης χαρακτηριστικα:

Μεγεθος εικονικων διευθυνσεων: 64 bits. Που όμως χρησιμοποιουνται τα 48 LS.
Χωρος εικονικων διευθυνσεων / διεργασια = 256 TeraByte

Μεγεθος κάθε σελιδας 4 Kbytes. Αρα τα 12 LS bits της εικονικης διευθυνσης δεν χρειαζονται μεταφραση.

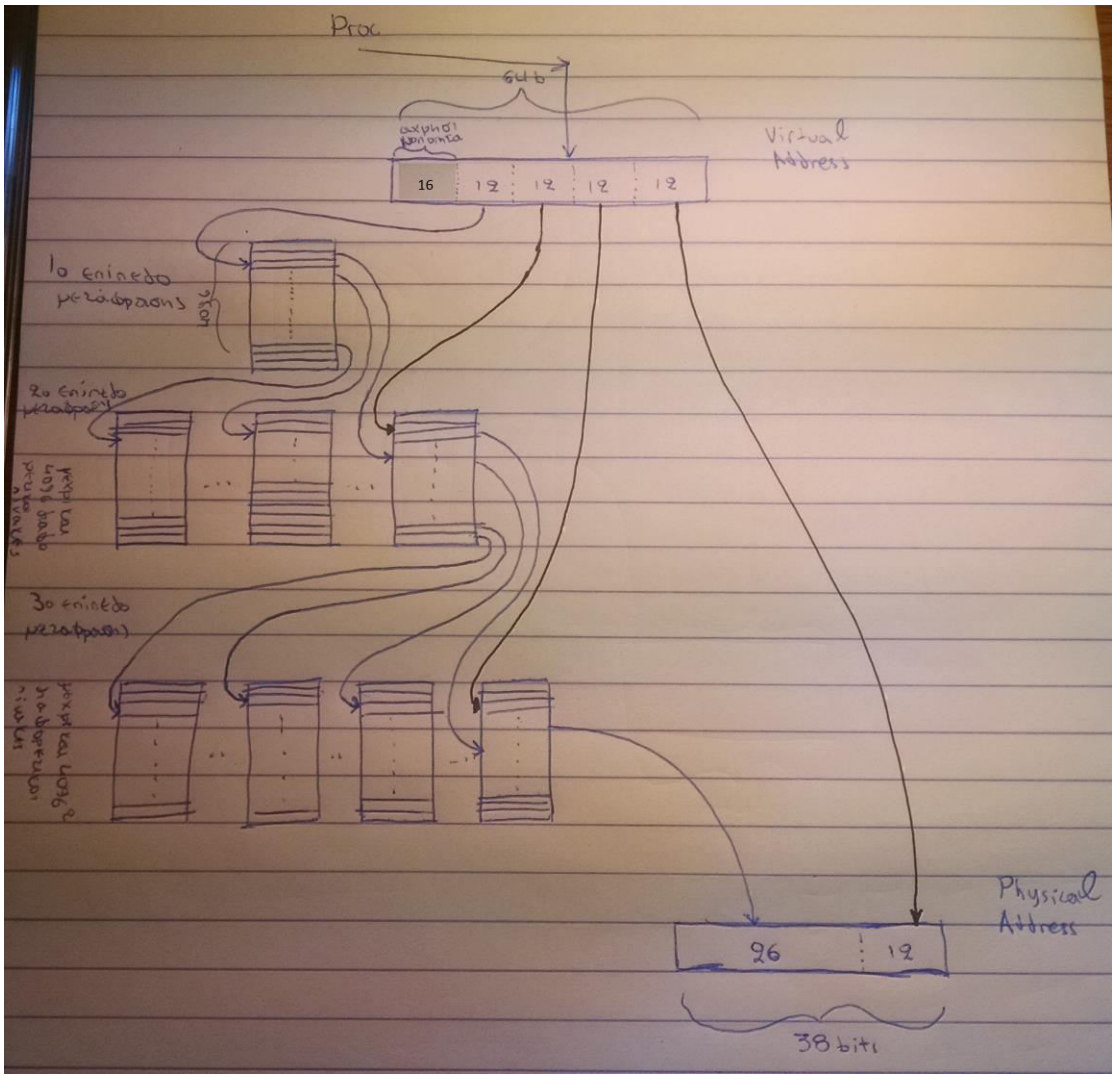
Τρια επιπεδα πινακων μεταφρασης ανα διεργασια

Κάθε ενας πινακας μεταφρασης εχει 4096 θεσεις και αρα βλεπει 12 από τα bits του αριθμου εικονικης σελιδας

Δυνατότητα επέκτασης έως 256 GBytes φυσικής μνήμης, δηλαδή 38 bits φυσική

διεύθυνση, ή 64 M φυσικές σελίδες (26-μπιτος αριθμός φυσικής σελίδας) επί 4

KBytes ανά σελίδα



ΑΣΚΗΣΗ 12.7

ΕΡΩΤΗΜΑ Α

Τα πεδια που θα υπαρχουν στο TLB είναι τα εξης:

Α) PID για να ξερω για ποια διεργασια προκειται. 8 bits

Β) Virtual Page 20 bits

C) Physical Page 16 bits

ΕΡΩΤΗΜΑ Β

	PID	VP	PP
0	3B	03	07
1	B4	03	11
2	3C	03	07
3	3B	FF	02
4	3C	FF	02
5	A2	C2	06
6	A3	E3	06

ΕΡΩΤΗΜΑ Γ

Υπαρχει ένα bit (το protection) οπου ελεγχει αν ο επεξεργαστης θα είναι σε καταστάση user ή kernel. Για ασφαλεια το μπιτ αυτό δεν μπορει να το αλλαξει ο χρηστης παραμονο το ιδιο το λειτουργικο συστημα. Ενω για παραπανω ελεγχο και προστασια υπαρχουν τα 3 μπιτ που καθοριζουν αν θα γινεται read/write/execute (r/w/x) ετσι ώστε ακομη κι όταν οι διεργασιες αλληλεπιδρουν μην χρειαζεται παραπανω χωρος στην φυσικη μνημη αλλα να προστατευονται από τα συγκεκριμενα μπιτς. Στην συγκεκριμενη περιπτωση όμως όλα αυτά δεν λειτουργουν σωστα αφου επιτρεπουν την αλληλεπιδραση μεταξυ διεργασιων.