ΕΠΛ 605: ΑΡΧΙΤΕΚΤΟΝΙΚΗ ΥΠΟΛΟΓΙΣΤΩΝ

ΧΕΙΜΕΡΙΝΟ ΕΞΑΜΗΝΟ 2018

ΕΡΓΑΣΙΑ 3 (13/10/2018)

Ημερομηνία Παράδοσης δεύτερου μέρους: 18/10/2018

Ημερομηνία Παράδοσης πρώτου μέρους: 25/10/2018

Θα πρέπει να παραδώσετε τυπωμένη την αναφορά σας για το κάθε μέρος στα αντίστοιχα εργαστήρια αλλά και να την αποστείλετε στο email: panagiota.nikolaou@cs.ucy.ac.cy με το ακόλουθο subject: EPL605-HW3.

Πρώτο Μέρος

A. Υλοποίηση προσομοιωτή κρυφής μνήμης (D cache)

Σκοπός της άσκησης αυτής είναι να υλοποιήσετε ένα προσομοιωτή (simulator) κρυφής μνήμης δεδομένων (D cache).

Ο κώδικας του προσομοιωτή σας θα πρέπει να εισαχθεί μέσα σε ένα από τα tools του PIN, το pinatrace.so, όπου παράγει διευθύνσεις μνήμης. Ο προσομοιωτής θα δέχεται σαν είσοδο τις διευθύνσεις μνήμης (load-R και store-W) που παράγονται από το PIN εργαλείο και αναλόγως θα τις αναπαριστά μέσα στον cache προσομοιωτή.

Ο cache προσομοιωτής θα ακολουθεί τις πολιτικές **write-back cache** και **no-write-allocate cache** και πρέπει να υποστηρίζει **direct-mapped και N-way set-associative cache**. Για το N-way set-associative, να χρησιμοποιήσετε την πολιτική αντικατάστασης block (block replacement policy) **Least-Recently Used (LRU).**

Για την ανάλυση σας θα χρησιμοποιήσετε τα δύο benchmarks που σας ανατέθηκαν στην 2η εργασία.

Για να μπορέσετε να κάνετε debug τα περιεχόμενα της cache και να επιβεβαιώσετε την ορθότητα τους είναι καλό στο τέλος να κρατάτε ένα αρχείο με τις διευθύνσεις μνήμης που υπάρχουν μέσα στην cache.

Στο τέλος κάθε εκτέλεσης, θα πρέπει παράγετε τα ακόλουθα στατιστικά:

a. Total Cache Accesses and Accesses per kilo Instructions (APKI).
b. Total Number of Cache Misses and Misses per kilo Instructions (MPKI).
c. Total Number of Cache Hits and Hits per kilo Instruction (HPKI).
d. Total number of replacements and Replacements per kilo Instruction (RPKI).
e. Total number of writebacks and Writebacks per kilo Instruction (RPKI).
f. Number of dirty blocks at the end of the execution.

Ενδεικτικό παράδειγμα με την αναπαράσταση πληροφοριών κατά το τέλος της εκτέλεσης της παρακάτω ακολουθίας διευθύνσεων μνήμης μιας cache χωρητικότητας 32KB, όπου το κάθε block έχει χωρητικότητα 64Bytes.

****Direct Map Cache**

Addresses:

	Tag	Index	Offset	
R 0x7f321977f6a8→	11111110011001000011001011101111	11011010	101000	–Miss
R 0x7ffe182d93b8→	11111111111111000011000001011011	00100111	011000	–Miss
R 0x7ffe182d93c0→	11111111111111000011000001011011	00100111	000000	–Miss
W 0x7ffe182d93e8→	11111111111111000011000001011011	00100111	101000	–Hit
R 0x7f321977de70→	11111110011001000011001011101111	01111001	110000	–Miss

Output:

Set	Block Address
0	…
78	FFFC305B
79	FFFC305B
…	…
179	FE6432EF
…	
474	FE6432EF
…	…
511	

Total Cache Accesses: 5
Total Number of Cache Misses: 4
Total Number of Cache Hits: 1
Total Number of replacements: 0
Total Number of writebacks:0
Number of dirty blocks:1

****4-way set associativity Cache**

	Tag	Index	Offset	
R 0x7f321977f6a8→	11111110011001000011001011101111	11011010	101000	–Miss
R 0x7ffe182d93b8→	11111111111111000011000001011011	00100111	011000	–Miss
R 0x7ffe182d93c0→	11111111111111000011000001011011	00100111	000000	–Miss
W 0x7ffe182d93e8→	11111111111111000011000001011011	00100111	101000	–Hit
R 0x7f321977de70→	11111110011001000011001011101111	11110011	100000	–Miss
R 0x7f321977de70→	11111110011001000011001011101111	11110011	100000	–Miss

Output:

Set	Cache Way				
	MRU	**MRU–1**	**MRU–2**	**MRU–3**	**LRU**
0	…				
…					
78	3FFF0C16C				
79	3FFF0C16C				
…					
90	3F990CBBF				
…					
121	3F990CBBE	3F990CBBF			
…					
127					

Total Cache Accesses: 6
Total Number of Cache Misses: 5
Total Number of Cache Hits: 1
Total Number of replacements: 0
Total Number of writebacks:0
Number of dirty blocks:1

Σημειώστε πως στον προσομοιωτή σας το **κάθε block έχει μέγεθος 64Bytes** και για την N-way cache, σημειώστε πως ο προσομοιωτής πρέπει να συμπληρώνει το κάθε set από **το Most-Recently-Used (MRU) way προς το Least-Recently-Used block (LRU) way.**

Παραδοτέα αυτής της άσκησης:

1. Σε αυτή την εργαστηριακή άσκηση σας ζητείτε να παραδώσετε ένα output file με την τελική κατάσταση της cache για το σενάριο **32KB, 8-way cache MONO**, όπως το παράδειγμα που δόθηκε πιο πάνω και να περιγράψετε τι παρατηρείτε.

Συγκεκριμένα, το output σας, να είναι στην παρακάτω μορφή:
Set-Number MRU Block(valid, dirty, tag) MRU-1(valid, dirty, tag)… LRU(valid, dirty, tag)
Η κάθε στήλη να ξεχωρίζει από την άλλη μόνο με ένα κενό και τα tag addresses να είναι σε δεκαεξαδική μορφή .

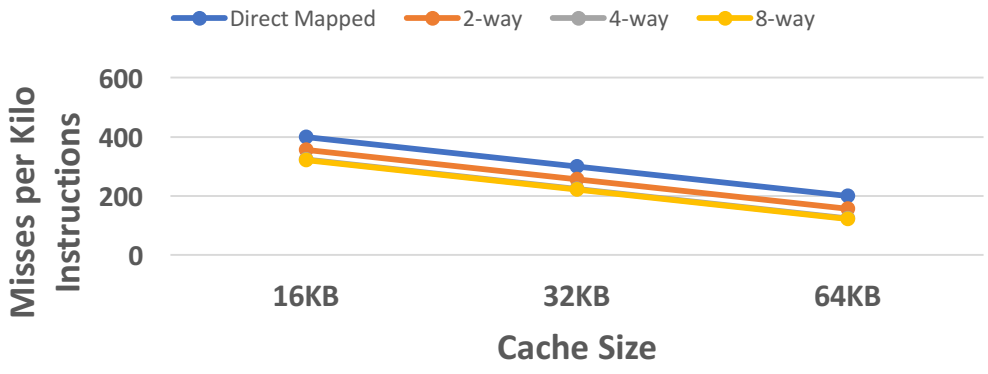
0 1,1,3FFF0C16C 1,0,3FFF0C17C ..
…
63 1,1,3FFF0C14C 1,0,3FFF0C17F ..

Αυτό το output μαζί με το report σας θα πρέπει να τα στείλετε στο email που αναγράφετε στην αρχή της άσκησης. **Το output θα έχει σαν όνομα μόνο την ταυτότητα σας.**

2. Να ετοιμάσετε μια γραφική παράσταση για το κάθε στατιστικό παρουσιάζοντας τις per Kilo τιμές για κάθε διαφορετικό μέγεθος και διαφορετικό αριθμό ways της cache.
   Τα **Cache Sizes που θα εξερευνήσετε είναι: 16KB, 32KB, 64KB** και οι τύποι cache:
   a. **Direct-Map Cache**
   b. **N-way Associativity:**
      i. 2-way
      ii. 4-way
      iii. 8-way

   Να σχολιάσετέ εκτενώς τα αποτελέσματα και τις παρατηρήσεις σας.

   Παράδειγμα γραφικής παράστασης για το στατιστικό Misses per Kilo Instructions:



B. Δεδομένου πως η L1D cache των μηχανών που τρέχατε τα προγράμματα σας στο δεύτερο μέρος είναι 32KB και είναι 8-way set associativity. Συγκρίνεται τις τιμές των στατιστικών του πρώτου μέρους (Α ερώτημα) με αυτά του δεύτερου. Τι παρατηρείται?

Δεύτερο Μέρος

Α. Εξοικείωση με το εργαλείο Perf

Σε αυτή την άσκηση σας ζητείτε να χαρακτηρίσετε τα benchmarks που σας έχουν δοθεί από την εργασία 2 με την βοήθεια του εργαλείου **perf**. §
Συγκεκριμένα, για το κάθε ένα benchmark να παράγετε τα ακόλουθα στατιστικά ανά 1second:

1. cache-misses
2. cache-references
3. branch-instructions
4. instructions
5. cycles
6. L1-dcache-load-misses
7. L1-icache-load-misses
8. branch-misses
9. LLC-load-misses
10. LLC-loads

Θα χρειαστείτε να τρέξετε την ακόλουθη εντολή:
perf stat -o {output file} -e {statistics} -I {time in milli-seconds} {executable}
eg. perf stat -o tmp -e instructions -I 1000 ./namd_r_base.EPL221SingleCore-gcc4.8.5-static-linux3.10-bits-64 --input apoa1.input --output apoa1.ref.output --iterations 7

Έπειτα, για το κάθε benchmark να δείξετε τον συσχετισμό που έχει το κάθε στατιστικό με το IPC χρησιμοποιώντας το correlation analysis tool που διδαχτήκατε στο Lab 4 **και να σχολιάσετε τα αποτελέσματα και τις παρατηρήσεις σας**.
Ακολούθως, να παράξετε μια γραφική δείχνοντας όλα τα ζητούμενα στατιστικά, παρουσιάζοντας τα ανά 1000 εντολές (per Kilo Instructions).

Παραδοτέα:

Ενδεικτικό παράδειγμα εξόδου αυτής της άσκησης για το Namd Benchmark:

Παράδειγμα γραφικής παράστασης για κάθε 1000 εντολές:

Παράδειγμα Correlation Analysis:

	IPC
IPC	1
Cache Misses	-0.3634235
Cache-References	-0.0945813
Branch Instructions	-0.1565165
Instructions	-0.175515
Cycles	-0.9807642
L1-dcache-load-misses	-0.2126848
L1-icache-load-misses	-0.6941617
branch-misses	0.02032164
LLC-load-misses	-0.8022018
LLC-loads	-0.2631457

Β. Ένα σύστημα έχει ιδεώδες CPI ίσο με 2. Αυτό το σύστημα έχει L1 Dcache miss rate 7%, L1 Icache miss rate 6% και L2 cache hit rate 98%. Οι κύκλοι ποινής για την DL1 και IL1 cache είναι 10 κύκλοι (miss penalty), ενώ οι κύκλοι ποινής για την L2 είναι 50 κύκλοι. Ποιο είναι το πραγματικό CPI για ένα πρόγραμμα που εκτελεί 0.5 προσβάσεις στην μνήμη ανά εντολή?

Γ. Να διαβάσετέ για την πολιτική αντικατάστασης Pseudo Least-Recently Used (PLRU) με tree - base υλοποίηση και να τη συγκρίνετε με την πολιτική LRU. Ποια πολιτική πιστεύετε θα έχει γενικά την καλύτερη απόδοση?

Δ.

1.7 [20/20/20/20] <1.6, 1.9> Your company's internal studies show that a single-core system is sufficient for the demand on your processing power; however, you are exploring whether you could save power by using two cores.

a. [20] <1.9> Assume your application is 80% parallelizable. By how much could you decrease the frequency and get the same performance?

b. [20] <1.6> Assume that the voltage may be decreased linearly with the frequency. Using the equation in Section 1.5, how much dynamic power would the dual-core system require as compared to the single-core system?

c. [20] <1.6, 1.9> Now assume the voltage may not decrease below 25% of the original voltage. This voltage is referred to as the *voltage floor*, and any voltage lower than that will lose the state. What percent of parallelization gives you a voltage at the voltage floor?

d. [20] <1.6, 1.9> Using the equation in Section 1.5, how much dynamic power would the dual-core system require as compared to the single-core system when taking into account the voltage floor?

	Sun Fire T2000	IBM x346
Power (watts)	298	438
SPECjbb (operations/sec)	63,378	39,985
Power (watts)	330	438
SPECWeb (composite)	14,001	4348

Figure 1.24 Sun power/performance comparison as selectively reported by Sun.