



3η ΑΣΚΗΣΗ ΠΡΟΗΓΜΕΝΑ ΘΕΜΑΤΑ ΑΡΧΙΤΕΚΤΟΝΙΚΗΣ ΥΠΟΛΟΓΙΣΤΩΝ

Ακ. έτος 2019-2020, 8ο Εξάμηνο, Σχολή ΗΜ&ΜΥ

Τελική Ημερομηνία Παράδοσης: **7/6/2020**

1. Εισαγωγή

Στην άσκηση αυτή θα χρησιμοποιήσετε τον προσομοιωτή “Sniper Multicore Simulator”, ο οποίος αξιοποιεί το εργαλείο “PIN” που χρησιμοποιήσατε στις προηγούμενες ασκήσεις. Στόχος της άσκησης είναι η μελέτη των χαρακτηριστικών των σύγχρονων superscalar, out-of-order επεξεργαστών και ο τρόπος με τον οποίο επηρεάζουν την απόδοση του συστήματος, την κατανάλωση ενέργειας καθώς και το μέγεθος του chip του επεξεργαστή. Λεπτομέρειες και υλικό (παρουσιάσεις, κώδικα, manual κτλ) σχετικά με τον προσομοιωτή μπορείτε να βρείτε εδώ:

http://snipersim.org/w/The_Sniper_Multi-Core_Simulator

2. Λήψη και εγκατάσταση του *sniper*

Στην ιστοσελίδα του μαθήματος θα βρείτε την έκδοση 7.3 του *sniper*:

```
$ wget http://www.cslab.ece.ntua.gr/courses/advcomparch/files/askiseis/sniper-7.3.tgz
$ tar xvfz sniper-7.3.tgz
$ cd sniper-7.3
```

Για να μπορέσετε να χρησιμοποιήσετε τα pinballs των benchmarks, όπως αναλύουμε παρακάτω, θα πρέπει να χρησιμοποιήσετε τον *sniper* με την έκδοση του pin που περιλαμβάνει και το εργαλείο *pinplay*, η οποία σας δίνεται στην ιστοσελίδα του μαθήματος:

```
$ wget
http://software.intel.com/content/dam/develop/external/us/en/protected/pinplay-dcfg-3.11-pin-3.11-97998-g7ecce2dac-gcc-linux.tar.bz2
$ tar xvfj pinplay-dcfg-3.11-pin-3.11-97998-g7ecce2dac-gcc-linux.tar.bz2
```

Μετά την λήψη του *sniper* και του pin προχωρήστε στην μεταγλώττιση του *sniper* μέσω των παρακάτω εντολών:

```
$ sudo apt-get update
$ sudo apt-get install python make g++ zlib1g-dev libbz2-dev libboost-dev
libsqlite3-dev
$ export PIN_HOME=/path/to/pinplay-dcfg-3.11-pin-3.11-97998-g7ecce2dac-gcc-linux
$ make
```

Όπου `/path/to/pin/pinplay-dcfg-3.11-pin-3.11-97998-g7ecce2dac-gcc-linux/` είναι το `path` στο οποίο βρίσκονται τα αρχεία του PIN.

Οι παραπάνω οδηγίες για την μεταγλώττιση του *sniper* έχουν δοκιμαστεί με επιτυχία σε:

- Ubuntu 18.04 με gcc-7
- Ubuntu 16.04 με gcc-5.4

Σε Ubuntu 20.04 η προεπιλεγμένη έκδοση του gcc είναι η έκδοση 9, με την οποία η μεταγλώττιση του sniper αποτυγχάνει. Σε αυτήν την περίπτωση μπορεί να χρησιμοποιηθεί η έκδοση 7 του gcc η οποία είναι διαθέσιμη στα αποθετήρια της διανομής.

```
$ sudo apt-get install gcc-7 g++-7
$ CC=gcc-7 CXX=g++-7 make
```

3. Χρήση του sniper

Αφού ολοκληρωθεί η μεταγλώττιση μπορείτε να τρέξετε τον sniper μέσω του αρχείου run-sniper:

```
$ ./run-sniper
Run program under the Sniper simulator
Usage:
  ./run-sniper [-n <ncores (1)>] [-d <outputdir (.)>] [-c <sniper-config>] [-c
c [objname:]<name[.cfg]>,<name2[.cfg]>,...] [-c <sniper-options:
section/key=value>] [-s <script>] [--roi] [--roi-script] [--viz] [--viz-aso]
[--profile] [--memory-profile] [--cheetah] [--perf] [--gdb] [--gdb-wait] [-
-gdb-quit] [--appdebug] [--appdebug-manual] [--appdebug-enable] [--follow-
execv=1] [--power] [--cache-only] [--fast-forward] [--no-cache-warming] [--
save-output] [--save-patch] [--pin-stats] [--wrap-sim=] [--mpi [--mpi-
ranks=<ranks>] [--mpi-exec="<mpiexec -mpiarg...>"] ] [--sim-end=<first|last|last-restart (default:
first)>] | --pinballs=<pinball-basename>,* | --pid=<process-pid> | [--sift]
-- <cmdline> }
```

Example: \$./run-sniper -- /bin/ls

Από τις παραμέτρους που δέχεται αυτό το script μας ενδιαφέρουν κυρίως οι εξής:

- -d <outputdir>: ο φάκελος στον οποίο θα αποθηκευτούν τα στατιστικά της προσομοίωσης
- -c <config-file>: το αρχείο με τις παραμέτρους του συστήματος που προσομοιώνουμε
- -g <options> : ορισμός παραμέτρων της προσομοίωσης
- --viz : ενεργοποιεί τα visualizations για τα αποτελέσματα της προσομοίωσης

5. Παραμετροποίηση του sniper

Η παραμετροποίηση των προσομοιωμένων συστημάτων γίνεται με τη χρήση των config files, τα οποία βρίσκονται στο φάκελο **sniper-7.3/config**. Εκεί βρίσκονται τα configurations για διάφορους επεξεργαστές. Για τους σκοπούς της άσκησης θα χρησιμοποιήσετε το configuration για τον επεξεργαστή gainestown, **gainestown.cfg**. Μέσα στο config file ορίζονται διάφορες παράμετροι του επεξεργαστή όπως η συχνότητα, το issue width, ο branch predictor και η ιεραρχία μνήμης. Για παράδειγμα, το παρακάτω κομμάτι ορίζει την L1 data cache:

```
[perf_model/l1_dcache]
perfect = false
cache_block_size = 64
cache_size = 32
associativity = 8
replacement_policy = lru
...
```

Όσες παράμετροι δεν ορίζονται στο αρχείο αυτό, ορίζονται στο είτε στο **base.cfg**, το οποίο χρησιμοποιείται σε κάθε προσομοίωση από το run-sniper είτε στο **nehalem.cfg**, το οποίο γίνεται include από το **gainestown.cfg**. Επίσης, μπορείτε να ορίσετε παραμέτρους της προσομοίωσης μέσω της γραμμής εντολών με τη χρήση της επιλογής -g του run-sniper.

Οι παράμετροι με τις οποίες καλείστε να πειραματιστείτε στην παρούσα άσκηση είναι οι **dispatch_width** και **window_size**. Η πρώτη είναι ο αριθμός των εντολών που μπορούν να γίνουν issue ταυτόχρονα (δηλαδή το “πόσο” superscalar είναι ο επεξεργαστής μας), ενώ η δεύτερη είναι το μέγεθος του ROB. Οι παράμετροι αυτοί βρίσκονται στο `[perf_model/core/interval_timer]` τμήμα του config file. Έτσι, για παράδειγμα με την παρακάτω εντολή ορίζετε τα **dispatch_width** και **window_size** ίσα με 8 και 256 αντίστοιχα:

```
./run-sniper <other options> -g --perf_model/core/interval_timer/dispatch_width=8
-g --perf_model/core/interval_timer/window_size=256
```

Σε κάθε εκτέλεση μιας προσομοίωσης δημιουργείται στο φάκελο των αποτελεσμάτων το αρχείο **sim.cfg**, το οποίο περιέχει αναλυτικά όλες τις παραμέτρους με τις τιμές που χρησιμοποιήθηκαν.

6. McPAT

Το McPAT (Multi-core Power, Area, Timing) μοντελοποιεί χαρακτηριστικά ενός επεξεργαστή, όπως η κατανάλωση ενέργειας και το μέγεθος που καταλαμβάνουν στο τσιπ οι διαφορετικές δομικές μονάδες του επεξεργαστή. Ο sniper περιλαμβάνει το McPAT, το οποίο χρησιμοποιείται για την εξαγωγή των στατιστικών μέσα από τον φάκελο μίας προσομοίωσης που έχει ολοκληρωθεί. Στο βοηθητικό κώδικα της άσκησης, δίνεται επιπλέον το script **advcomparch_mcpat.py** το οποίο είναι μία τροποποιημένη έκδοση του **mcpat.py** που περιέχεται στο sniper. Το **advcomparch_mcpat.py** χρησιμοποιεί το εργαλείο gnuplot για την απεικόνιση αποτελεσμάτων με τη μορφή γραφημάτων. Αν το gnuplot δεν είναι ήδη εγκατεστημένο στον υπολογιστή σας, μπορείτε να το εγκαταστήσετε με την παρακάτω εντολή:

```
$ sudo apt-get install gnuplot
```

Αφού αντιγράψετε το `advcomparch_mcpat.py` στον φάκελο `sniper-7.3/tools` μπορείτε να το χρησιμοποιήσετε ως εξής:

```
$ /path/to/sniper-7.3/tools/advcomparch_mcpat.py -h
Usage: /home/user/advcomparch/sniper/tools/advcomparch_mcpat.py [-h (help)] [-j
<jobid> | -d <resultsdirectory (default: .)>] [-t <type:
total|dynamic|static|peak|peakdynamic|area>] [-c <override-config>] [-o <output-
file (power{.png, .txt, .py})>]
```

Από τις παραμέτρους που δέχεται αυτό το script μας ενδιαφέρουν κυρίως οι εξής:

- `-d <resultsdirectory>` : ο φάκελος της προσομοίωσης.
- `-t <type>` : το είδος των στατιστικών που θέλουμε να εξάγουμε.
- `-o <output-file>` : το όνομα που θα έχουν τα αρχεία που θα παραχθούν.

Η εκτέλεση του script παράγει 4 αρχεία, τα **power.png**, **power.py**, **power.txt** και **power.xml**. Το πρώτο περιέχει τη γραφική αναπαράσταση των διαφόρων στατιστικών. Το δεύτερο περιλαμβάνει όλα τα στατιστικά που έχουν παραχθεί σε μορφή κατάλληλη ώστε να μπορεί να χρησιμοποιηθεί απευθείας σε python script, ενώ το τρίτο περιλαμβάνει τα ίδια στατιστικά σε μορφή κειμένου. Το `.xml` αρχείο είναι το αρχείο που παράγεται από τον sniper και δίνεται ως είσοδος στο McPAT. Το `advcomparch_mcpat.py` τυπώνει επίσης στην οθόνη (ή όπου κάνετε redirect την έξοδο) συνοπτικά τα στατιστικά που ορίσατε με το όρισμα `-t`. Ένα παράδειγμα χρήσης του δίνεται παρακάτω:

```

$ advcomparch_mcpat.py -d bench_dir -t total -o bench_dir/power >
bench_dir/power.total.out
$ ll bench_dir
total 172
drwxrwxr-x      2 user user  4096 Μάι  11 13:24 ./
drwxrwxr-x 458 user user 36864 Μάι  11 13:08 ../
-rw-rw-r--      1 user user  6602 Μάι  11 13:13 power.png
-rw-rw-r--      1 user user 18287 Μάι  11 13:13 power.py
-rw-rw-r--      1 user user 12572 Μάι  11 13:13 power.total.out
-rw-rw-r--      1 user user 12572 Μάι  11 13:13 power.txt
-rw-rw-r--      1 user user 19221 Μάι  11 13:13 power.xml
-rw-rw-r--      1 user user   6121 Μάι   7 21:02 sim.cfg
-rw-rw-r--      1 user user   2481 Μάι   7 21:11 sim.info
-rw-rw-r--      1 user user   1900 Μάι   7 21:11 sim.out
-rw-r--r--      1 user user 45056 Μάι   7 21:11 sim.stats.sqlite3

```

```
$ cat bench_dir/power.total.out
```

	Power	Energy	Energy %
core-core	10.15 W	0.92 J	41.45%
core-ifetch	1.78 W	0.16 J	7.26%
core-alu	0.40 W	0.04 J	1.65%
core-int	1.07 W	0.10 J	4.37%
core-fp	1.05 W	0.10 J	4.30%
core-mem	0.63 W	0.06 J	2.59%
core-other	0.95 W	0.09 J	3.89%
icache	0.82 W	0.08 J	3.37%
dcache	1.36 W	0.12 J	5.56%
l2	1.98 W	0.18 J	8.09%
l3	0.00 W	0.00 J	0.00%
nuca	0.00 W	0.00 J	0.00%
noc	0.02 W	2.26 mJ	0.10%
dram	4.25 W	0.39 J	17.37%
core	16.04 W	1.46 J	65.52%
cache	4.17 W	0.38 J	17.02%
total	24.49 W	2.23 J	100.00%

Σε κάποιες περιπτώσεις όταν τρέξετε το `advcomparch_mcpat.py` θα σας επιστρέψει ένα μήνυμα σφάλματος της μορφής:

```
"ValueError: No valid McPAT output found"
```

Αυτό συμβαίνει συγκεκριμένα σε περιπτώσεις όπου το window size είναι αρκετά μικρό (≤ 8). Το πρόβλημα δεν σχετίζεται με το `advcomparch_mcpat.py` που σας παρέχεται αλλά με τη λειτουργία του `mcpat` και τις ελάχιστες προδιαγραφές που θέτει στον επεξεργαστή που μοντελοποιεί. Μπορείτε να παραλείψετε τις προσομοιώσεις με window size ≤ 8 στις περιπτώσεις που μελετάτε την ενεργειακή κατανάλωση του επεξεργαστή.

7. Energy-Delay Product

Παραδοσιακά, για την αξιολόγηση της κατανάλωσης ενός επεξεργαστή χρησιμοποιείται ως μετρική η συνολική κατανάλωση ενέργειας σε Joules. Ωστόσο, πολλές φορές απαιτείται η μελέτη της επίδρασης διαφόρων χαρακτηριστικών του επεξεργαστή όχι μόνο στην κατανάλωση αλλά ταυτόχρονα και στην επίδοσή του. Για το σκοπό αυτό έχει προταθεί ως μετρική το energy-delay product (EDP). Το EDP για την εκτέλεση ενός benchmark ορίζεται ως το γινόμενο της ενέργειας επί τον χρόνο εκτέλεσης του benchmark:

$$EDP = Energy(J) * runtime(sec)$$

Αντίστοιχα, αν θέλουμε να δώσουμε περισσότερο βάρος στον χρόνο εκτέλεσης μπορούμε να υψώσουμε το *runtime* στο τετράγωνο, στον κύβο κ.λ.π. Έτσι προκύπτουν τα ED^2P , ED^3P και ούτω καθεξής:

$$ED^2P = Energy(J) * runtime^2(sec)$$

$$ED^3P = Energy(J) * runtime^3(sec)$$

8. Μετροπρογράμματα

Ο *sniper* μπορεί να χρησιμοποιηθεί για την εκτέλεση οποιασδήποτε εφαρμογής. Στα πλαίσια της παρούσας άσκησης θα χρησιμοποιήσετε τα SPEC_CPU2006 benchmarks, όπως και στην προηγούμενη άσκηση. Συγκεκριμένα, θα χρησιμοποιήσετε τα παρακάτω 12 benchmarks:

- | | | |
|------------------|---------------|-------------------|
| 1. 403.gcc | 5. 445.gobmk | 9. 459.GemsFDTD |
| 2. 429.mcf | 6. 450.soplex | 10. 471.omnetpp |
| 3. 434.zeusmp | 7. 456.hmmer | 11. 473.astar |
| 4. 436.cactusADM | 8. 458.sjeng | 12. 483.xalancbmk |

Ωστόσο, επειδή η εκτέλεση με χρήση του *sniper* είναι πολύ αργή, δε θα εκτελέσετε ολόκληρα τα benchmarks αλλά θα χρησιμοποιήσετε τα *pinballs* (*simpoints* που είχαμε συζητήσει στο μάθημα για τις προσομοιώσεις) που παρέχονται στη σελίδα του *sniper*:

```
$ wget http://snipersim.org/documents/pinballs/cpu2006-pinpoints-w0-d1B-m1.tar
$ tar xvf cpu2006-pinpoints-w0-d1B-m1.tar
$ ls cpu2006_pinballs
```

Για την εκτέλεση ενός benchmark χρησιμοποιώντας το κατάλληλο *pinball* δίνετε:

```
$ ./run-sniper -c gainestown -d sim.out --
pinballs=/path/to/cpu2006_pinballs/gcc/pinball_short.pp/pinball_t0r1_warmup3000_p
rolog0_region1000000002_epilog0_001_1-00000.0.address
```

9. Πειραματική Αξιολόγηση

Εκτελέστε όλα τα benchmarks για κάθε διαφορετικό επεξεργαστή που προκύπτει από το συνδυασμό των παρακάτω τιμών για τις παραμέτρους *dispatch_width* και *window_size*:

<i>dispatch_width</i>	1	2	4	8	16	32						
<i>window_size</i>	1	2	4	8	16	32	64	96	128	192	256	384

- i) Χρειάζεται πραγματικά να προσομοιώσετε και τους 72 διαφορετικούς επεξεργαστές που προκύπτουν με βάση τις παραπάνω τιμές; Αν όχι, εξηγήστε ποιές περιπτώσεις μπορείτε να παραλείψετε και γιατί. Δικαιολογήστε την απάντησή σας όχι μόνο θεωρητικά αλλά και αποδεικνύοντας για ένα μικρό αριθμό αυτών των περιπτώσεων ότι καλώς τις παραλείψατε.
- ii) Πώς επηρεάζει η κάθε παράμετρος την απόδοση του επεξεργαστή; Σε ποιά συμπεράσματα μπορείτε να καταλήξετε ως προς το σχεδιασμό ενός *superscalar*, *out-of-order* επεξεργαστή;
- iii) Πώς επηρεάζει η κάθε παράμετρος την κατανάλωση ενέργειας και το μέγεθος του τσιπ;
- iv) Βρείτε τα αντίστοιχα χαρακτηριστικά (*dispatch_width*, *window_size*) για τον επεξεργαστή του προσωπικού σας υπολογιστή ή για κάποιον από τους σύγχρονους επεξεργαστές (π.χ. Broadwell, Skylake, Kabylake). Δικαιολογούνται οι τιμές που επιλέξαν σε αυτά τα συστήματα οι αρχιτέκτονες με βάση τις προσομοιώσεις που εκτελέσατε και τα συμπεράσματα στα οποία

καταλήξατε; Θα είχε νόημα να ήταν διαφορετικές (π.χ. μεγαλύτερο `window_size`); Για ποιό λόγο πιστεύετε δεν κάνανε κάποια άλλη επιλογή;

Παραδοτέο της άσκησης θα είναι ένα ηλεκτρονικό κείμενο (**pdf**, **docx** ή **odt**). Στο ηλεκτρονικό κείμενο να αναφέρετε στην αρχή τα στοιχεία σας (Όνομα, Επώνυμο, ΑΜ).

Η άσκηση θα παραδοθεί ηλεκτρονικά στην ιστοσελίδα:

<http://www.cslab.ece.ntua.gr/courses/advcomparch/submit>

Δουλέψτε ατομικά. Έχει ιδιαίτερη αξία για την κατανόηση του μαθήματος να κάνετε μόνοι σας την εργασία. Μην προσπαθήσετε να την αντιγράψετε από άλλους συμφοιτητές σας.

Μην αφήσετε την εργασία για το τελευταίο Σαββατοκύριακο, απαιτεί αρκετό χρόνο για την εκτέλεση όλων των προσομοιώσεων, ξεκινήστε αμέσως!