ΕΠΛ 605: ΑΡΧΙΤΕΚΤΟΝΙΚΗ ΥΠΟΛΟΓΙΣΤΩΝ
ΧΕΙΜΕΡΙΝΟ ΕΞΑΜΗΝΟ 2017
ΕΡΓΑΣΙΑ 2 (13/02/2017)
ΗΜΕΡΟΜΗΝΙΑ ΠΑΡΑΔΟΣΗΣ 26/02/2017

Σας δίνονται **2 benchmark** από το **SPEC CPU2006** (http://www.spec.org/cpu2006/). Το πρώτο, για σκοπούς σύγκρισης είναι το ίδιο για όλους και είναι το **401.bzip2** και το δεύτερο ένα από τα πιο κάτω διαφορετικό για τον καθένα.
Οδηγίες για το πώς να τρέξετε τα benchmarks μπορείτε να βρείτε στο πιο κάτω σύνδεσμο:
http://boegel.kejo.be/ELIS/spec_cpu2006/spec_cpu2006_command_lines.html
(Σε περίπτωση που έχετε πολλαπλές εκτελέσεις να επιλέγετε την πρώτη)

Τα benchmarks και τα δεδομένα εισόδου τους θα τα βρείτε στον πιο κάτω φάκελο:
/home/students/cs/SPEC2006/
Στην Ιστοσελίδα των εργαστηρίων υπάρχει περιγραφή του πώς να τα χρησιμοποιήσετε π.χ. Compile, Run and PIN.

1) Περιγράψετε με απλά λόγια το τι κάνει (τι υπολογίζει) το καθένα από τα Benchmark που σας έχουν δοθεί και δώστε τον τρόπο και χρόνο (Wall Time) εκτέλεσης του, και τον αριθμό εντολών με time και perf . (Χρησιμοποιήστε τις μηχανές του εργαστηρίου των οποίων θα πρέπει να δώσετε τα χαρακτηριστική)
(Δες http://www.spec.org/cpu2006/publications/CPU2006benchmarks.pdf και http://www.spec.org/cpu2006/Docs/)

2) Με τη βοήθεια του **gprof** να εντοπίσετε το κώδικα (function, loop) στον οποίο αναλώνεται ο περισσότερος χρόνος εκτέλεσης του προγράμματος σας και να σχολιάσετε. Δώστε όλα τα στοιχεία για να τεκμηριώσετε την τοποθέτηση σας.

3) Με τη βοήθα του **gprof** και άλλων εργαλείων σχεδιάστε το Call Graph για το πιο πάνω benchmark.

Αφού τρέξετε τα benchmarks με τη βοήθεια του εργαλείου PIN να:

4) Να γίνει καταμέτρηση (1 μέτρηση = total number of instruction/100,000) πόσες φορές εκτελέστηκαν εντολές branch και ποιες από αυτές ήταν taken κατά την εκτέλεση των πιο πάνω benchmark. Δηλαδή, να φτιάξετε μία γραφική παράσταση με άξονα X με 100,000 μετρήσεις και άξονα Y_{left} πόσες εντολές branch εντοπίστηκαν και Y_{right} πόσες εντολές branch ήταν taken .

5) Με τη βοήθεια του PINTool iTrace και του objdump να δημιουργήσετε γραφική παράσταση με τη συχνότητα εμφάνισης των εντολών των προγραμμάτων σας ταξινομημένη με αύξουσα σειρά. Ο άξονας των X να δίνει την εντολή και ο άξονας Y την συχνότητα και τον αριθμό που η κάθε εντολή εκτελείτε. Επιπρόσθετα θα πρέπει να βρείτε και την συνάρτηση στην οποία ανήκει η εντολή που εκτελείτε τις περισσότερες φορές και να εξηγήσετε τι κάνει αυτή η συνάρτηση.

6) Σχολιάστε εκτενώς τα αποτελέσματα των πιο πάνω μετρήσεων σας.

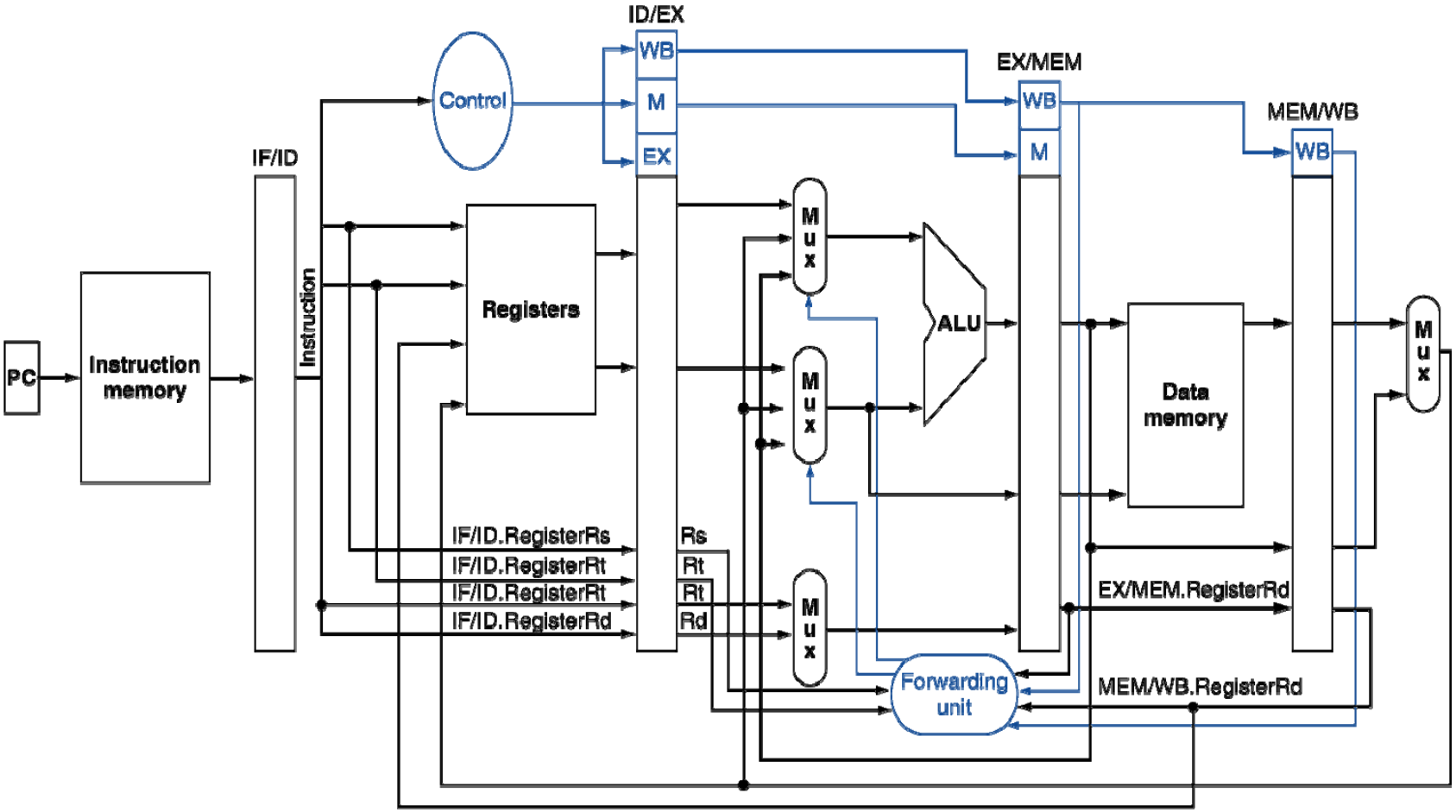
7) Προσπαθήστε να μειώσετε το χρόνο εκτέλεσης μεταγλωττίζοντας με τις πιο κάτω παραμέτρους
–O0, -O1, -O2, -O3, -Os -ftree-vectorize
Τι παρατηρείτε και γιατί; Τι αλλαγές βλέπετε στα αποτελέσματα των πιο πάνω πειραμάτων.

8) Με τη βοήθεια του –S (http://gcc.gnu.org/onlinedocs/gcc-4.4.1/gcc/Overall-Options.html#Overall-Options) και **–O3** πάρτε την assembly του κώδικα σας και αφού τη μελετήσετε προσεκτικά τροποποιήστε την εφαρμόζοντας τεχνικές που έχετε διδακτή στις διαλέξεις έτσι ώστε να εκτελείτε σε λιγότερο χρόνο. Δείξετε και αιτιολογήστε τις αλλαγές σας. (**Bonus +15%)**

9) Από πληροφορίες που θα βρείτε στην ιστοσελίδα (http://www.spec.org/cpu2006/results/) να απαντήσετε:

a. Ποιά μηχανή είχε το πιο ψηλό SPEC CPU 2006 base ratio για integer και ποιά για float κατά το 3^{o} τετράμηνο του 2014.
b. Περιγράψετε τα χαρακτηριστικά της μηχανής, το λειτουργικό, το μεταγλωττιστή κτλ.
c. Για την πιο πάνω μηχανή να δώστε το spec ratio του κάθε προγράμματος.
d. Ποιά η διαφορά του SPEC base από το SPEC peak performance μιας μηχανής.

ID	Benchmark	ID	Benchmark
1	400.perlbench	24	473.astar
2	410.bwaves	25	481.wrf
3	416.gamess	26	482.sphinx3
4	429.mcf	27	483.xalancbmk
5	433.milc	28	998.specrand
6	434.zeusmp	29	999.specrand
7	435.gromacs		
8	436.cactusADM		
9	437.leslie3d		
10	444.namd		
11	445.gobmk		
12	447.dealII		
13	450.soplex		
14	453.povray		
15	454.calculix		
16	456.hmmer		
17	458.sjeng		
18	459.GemsFDTD		
19	462.libquantum		
20	464.h264ref		
21	465.tonto		
22	470.lbm		
23	471.omnetpp		



10. Στο πιο πάνω σχήμα προσθέστε τα απαραίτητα συστατικά ώστε αν εντολές LW-SW εκτελούνται σε διαδοχικά (χρησιμοποιούν τον ίδιο καταχωρητή) να εκτελούνται πιο αποδοτικά (forwarding). (move from mem. Location A to mem. Location B)