1. Introduction

The Standard Performance Evaluation Corporation (SPEC) is a nonprofit organization that consists of various members such as hardware and software vendors, universities, customers, and consultants. The main goal of this organization is development of a set of standardized computer benchmarks that can be used for performance evaluation of components and system in presence of different operating systems and/or environments. A computer benchmark suite may be helpful in characterizing different aspects of a system overall performance.

A benchmark is defined as a standard way of measurement and a computer benchmark is a program that runs specific operations for specific purposes and returns certain results that show the system behavior. The results can be mainly related to speed (of completed operations) and throughput (i.e. number of completed operations per unit time). Running a benchmark on multiple computer systems provide some information that can be utilized for comparison.

Any benchmark might not be useful in determination of a system performance characteristics. In fact, a benchmark with having a suitable correlation with the desired performance parameters needs to be leveraged in this regard. The SPEC benchmark suite targets computation-intensive performance with emphasizing on the performance of processor, memory hierarchy, and compilers. This suite can be divided to two parts that are: (a) CINT, which focuses on integer based operations; and (b) CFP, which focuses on floating point based operations. For running these benchmarks, a Unix system with at least 256MB of memory, 1GB of disk, and a set of compilers is required.

CINT contains eleven applications written in C (except one of them that is in C++) and is shown in Table 01, and CFP contains 14 applications (six Fortran77, four Fortran90 and four C) and is shown in Table 02. The rest of this report is organized as follows. In Section 2, the memory footprint of SPEC benchmark suite is discussed. Section 3 describes the benchmark under analysis that is MCF. The leveraged tool for running the SPEC benchmark suite is explained in Section 4. The benchmark is fully analyzed and investigated in Section 5. The original and modified commands for running simulation are mentioned in the sixth section. The achieved results from using original and modified commands are compared in Section 7. The possible arguments in the Super Seven commands are cited in Section 8. The achieved results from using the original and modified commands and running the MCF benchmark are provided in Section 9 and 10.

Name	Reference Time (Second)	Remarks
164.gzip	1400	Data Compression Utility
175.vpr	1400	FPGA Circuit Placement and Routing
176.gcc	1100	C Compiler
181.mcf	**1800**	**Minimum Cost Network Flow Solver**
186.crafty	1000	Chess Program
197.parser	1800	Natural Language Processing
252.eon	1300	Ray Tracing
253.perlbmk	1800	Perl
254.gap	1100	Computational Group Theory
255.vortex	1900	Object Oriented Database
256.bzip2	1500	Data Compression Utility
300.twolf	3000	Place and Route Simulator

Table 01. The SPEC integer-based benchmarks

Name	Reference Time (Second)	Remarks
168.wupwise	1600	Quantum Chromodynamics
171.swim	3100	Shallow Water Modeling
172.mgrid	1800	Multi-Grid Solver in 3D Potential Field
173.applu	2100	Parabolic/Elliptic Partial Differential Equations
177.mesa	1400	3D Graphics Library
178.galgel	2900	Fluid Dynamics: Analysis of Oscillatory Instability
179.art	2600	Neural Network Simulation; Adaptive Resonance Theory
183.equake	1300	Finite Element Simulation; Earthquake Modeling
187.facerec	1900	Computer Vision: Recognizes Faces
188.ammp	2200	Computational Chemistry

189.lucas	2000	Number Theory: Primality Testing
191.fma3d	2100	Finite Element Crash Simulation
200.sixtrack	1100	Particle Accelerator Model
301.apsi	2600	Solves Problems Regarding Temperature, Wind, Velocity and Distribution of Pollutants

Table 02. The SPEC floating point-based benchmarks

2. SPEC Benchmark Suite Memory Footprint

As it was mentioned before, the SPEC benchmarks are used for examination of the processor, memory hierarchy, and compiler efficiency. Most of these benchmarks consume data space larger than common cache size (~ 512KB – 16MB), many of them larger than 100MB, and none of them larger than 200MB. In this way, a sufficient margin is considered on a 256MB computer system. Table 03 shows the memory footprint of these benchmarks. Resident size and virtual size were obtained using the respective command in a loop with a sleep of 5 seconds between each observation. The third column indicates the total number of observations. The fourth column shows the number of observations that reported the same value prior to the termination of the benchmark. Thus a zero in that column indicates that the last observation was different from the one before that. The final column contains the word "Yes" if the produced value is the same in at least 90% of the number of observations. In other words, a benchmark like that has a rapid operating process and then remains in a stable state.

	Resident Size (Max)	Virtual Size (Max)	Number of Observation	Number of Produced Same Value	Stable Behavior
gzip	180.0	199.0	181	68	No
vpr	50.0	53.6	151	6	Yes
gcc	154.0	156.0	134	0	Yes
mcf	**190.0**	**190.0**	**232**	**230**	**Yes**
crafty	2.0	2.6	107	106	No
parser	37.0	66.8	263	254	No
eon	0.6	1.5	130	0	Yes
perlbmk	146.0	158.0	186	0	No

gap	192.0	158.0	149	148	No
vortex	72.0	79.4	162	0	No
bzip2	185.0	199.0	153	6	Yes
twolf	3.4	4.0	273	0	Yes
wupwise	176.0	177.0	185	181	Yes
swim	191.0	192.0	322	320	Yes
mgrid	56.0	56.7	281	279	Yes
applu	181.0	191.0	371	369	No
mesa	9.4	23.1	132	131	No
galgel	63.0	155.0	287	59	Yes
art	3.7	4.3	157	37	Yes
equake	49.0	49.4	218	216	Yes
facerec	16.0	18.5	182	173	Yes
ammp	26.0	28.4	277	269	Yes
lucas	142.0	143.0	181	179	Yes
fma3d	103.0	105.0	268	249	Yes
sixtrack	26.0	59.8	148	141	Yes
apsi	191.0	192.0	271	270	Yes

Table 03. The SPEC suite memory footprint

3. MCF Benchmark

MCF benchmark is included in the SPEC integer-based benchmarks, and was derived from a ANSI C language based program for single-depot vehicle scheduling problem in public mass transportation that is a “combinatorial optimization” problem. The arithmetic operations in this benchmark are mostly integer-based. During the process of this problem solving, the program considers one single depot and a homogenous vehicle fleet. Consequently, the timetabled trips with fixed departure or arrival locations and times are derived according to a line plan and frequented service. Each timetabled trip has the responsibility of serving only one vehicle. These trips are linked to each other using dead-head trips.

There are also pull-out and pull-in trips for leaving and entering the depots in addition to the aforementioned ones. The goal is to have a cost efficient and scheduled state for all these trips to blocks in order to have the least number of vehicles. The core of MCF benchmark is the network complex algorithm that is defined as a specialized version of the well-known simplex algorithm for network flow problems. For increasing the execution speed, the linear algebra of the general algorithm is replaced by simple network operations (e.g. modifying the spanning tree).

The executable file of this benchmark takes an input file and generates an output file. The input fie consists of: (a) the number of timetabled and dead-head trips; (b) the starting time and ending time for each timetabled; (c) the cost and the starting and ending timetabled for each dead-head trip. The worst execution time is pseudo-polynomial in the number of timetabled and dead-head trips and in the amount of the maximal cost coefficient, as opposed to the expected execution time that is in the order of a low-order polynomial. The output file includes check output values describing an optimal schedule computed by the program. Meanwhile, the benchmark execution requires about 100 and 190 megabytes for a 32-bit and a 64-bit architecture.

4. Benchmark Analysis by Architectural Simulators

In order to perform fast, flexible, and accurate simulation of modern microprocessors, architectural simulators such as GEM5 or SimpleScalar tool set can be used. An architectural simulator is defined as a software program that models computer systems (and their components) for output prediction and performance evaluation for a delivered input. It can model only the processor or the whole system that includes processor, memory hierarchy, compiler, and input/output devices. Another application of an architectural simulator is analyzing a benchmark according to the achieved results from running it on multiple processors. For analyzing the MCF, the SimpleScalar tool set is used in this project.

The built architecture inside this tool set is a derivative of the MIPS architecture, which takes the architecture-specific compiled binaries for examination of seven different architectural aspects of the processor architecture (a.k.a. Super Seven). The examination is done through compiling a user developed application or a benchmark (e.g. from a SPEC's benchmark suite) by the architecture-specific GNU GCC compiler and running it on the processor.

The seven architectural aspects include branch prediction behavior, cache behavior, external I/O trace generator, fast functional simulator implementation, out of order execution, sample functional simulator implementation with profiling, sample functional simulator implementation, and provide high flexibility, portability, extensibility, and performance. The main software components inside the tool set are: `bpred`, `cache`, `config`, `dlite`, `eio`, `endian`, `eval`, `eventq`, loader, machine, main, memory, misc, options, `ptrace`, `range`, `regs`, `resource`, `sim-bpred`, `sim-cache`, `sim-eio`, `sim-fast`, `sim-outorder`, `sim-profile`, `sim-safe`, `symbol`, `syscall`, `sysprobe`.

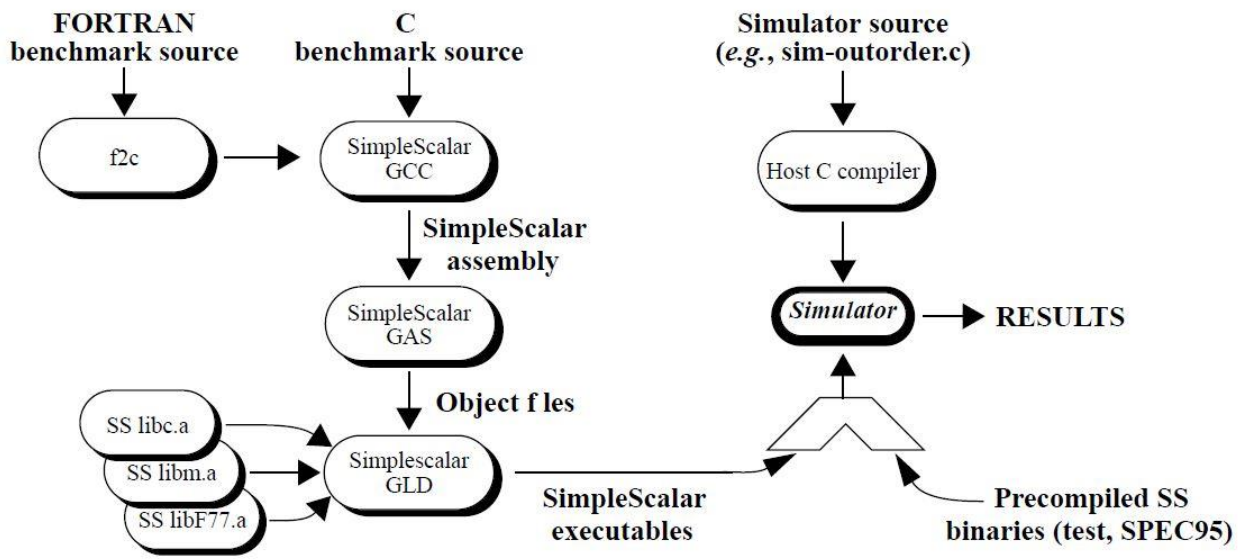

Figure 01. SimpleScalar tool set overview

The SimpleScalar tool set includes both little-endian and big-endian versions of the derivation of the MIPS-IV ISA for improvement of the portability. The differences between the semantics of the SimpleScalar ISA and the MIPS ISA can be described as: (1) having no architected delay slots: loads, stores, and control transfers don't execute the succeeding instruction; (2) loads and stores support two addressing modes that are indexed (Register + Register) and auto increment/decrement; (3) inclusion of a square-root instruction for floating point numbers; and (4) an extended 64-bit instruction encoding.

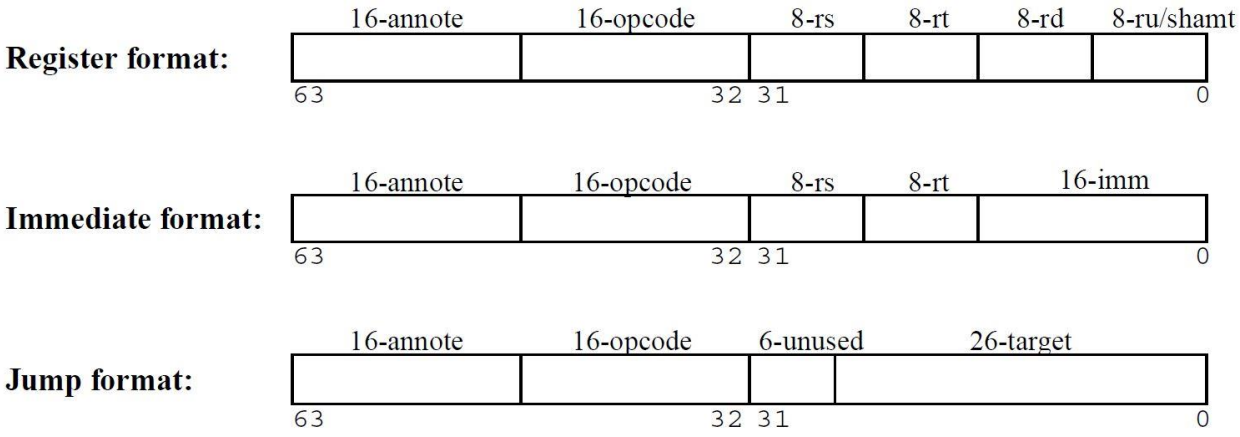

Figure 02. SimpleScalar architecture instruction formats

5. Analysis of MCF Benchmark

A) General Analysis

According to Table 01-03, the MCF has the third position among the integer-based benchmarks in having a high execution time. This indicates that it is one of the benchmarks that has many number of time consuming operations, which causes it to be considered as a (relatively) complex benchmark. With respect to memory footprint, this benchmark has the third and fourth positions in having the maximum resident size and the maximum virtual size respectively. So, it can be represented as a data intensive or highly memory consuming program. Although it displays a stable behavior in interaction with the memory hierarchy after a certain period of time.

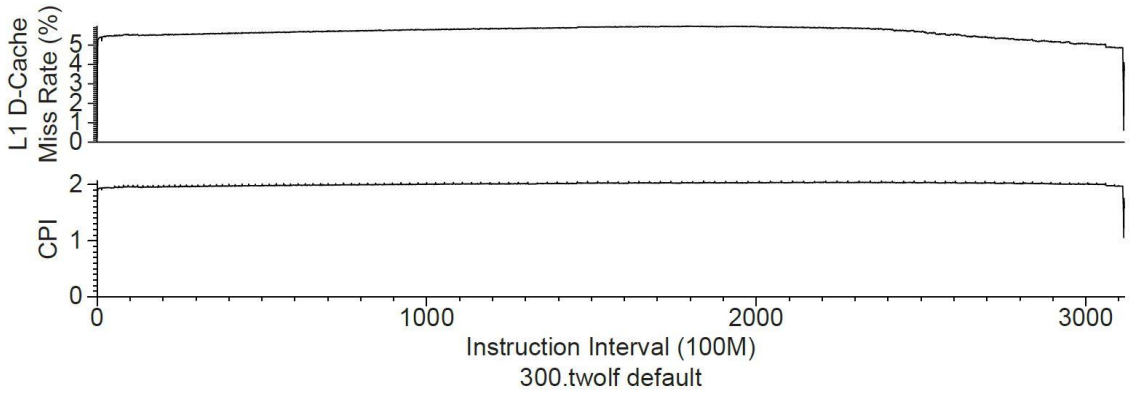

Figure 03. SimPoint trend of TWOLF benchmark

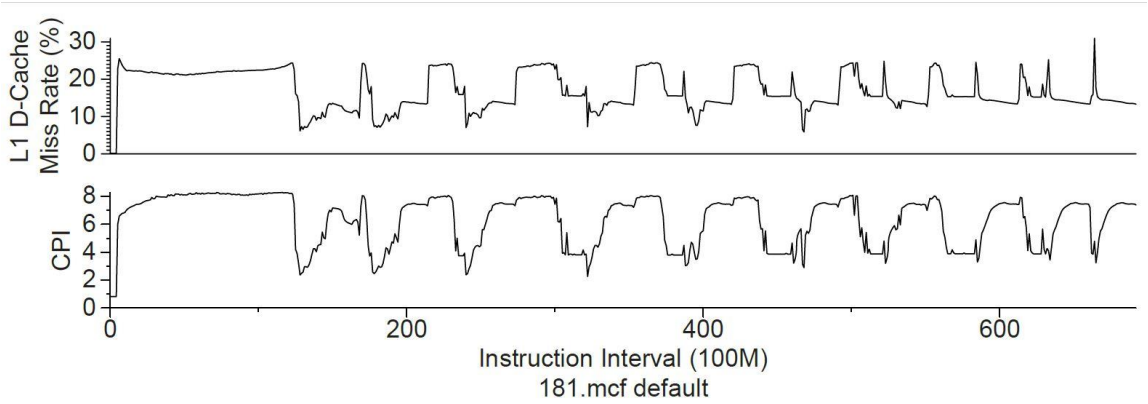

Figure 04. SimPoint trend of MCF benchmark

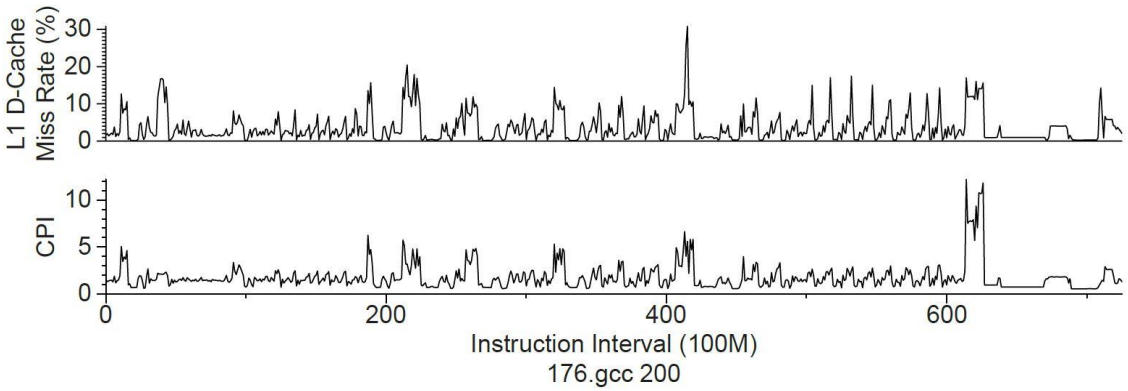

Figure 05. SimPoint trend of GCC benchmark

Vincent M. Weaver and Sally A. McKee (in "Using Dynamic Binary Instrumentation to Generate Multi-platform SimPoints: Methodology and Accuracy") perused phase behavior of the SPEC benchmarks using SimPoint modeling. In fact, many software programs express cyclic behavior (i.e. having a similar behavior in every time period). Therefore, a time interval of such a program can be utilized to represent its whole behavior. This way of program behavior representation is called SimPoints modeling. The program phase behavior of three SPEC benchmarks namely, TWOLF, MCF, and GCC, with granularity of 100M instructions are demonstrated in Figures 03, 04, and 05. Each of these figures shows two performance related metrics that are Level-1 Data Cache Miss Rate (Top) and Cycles Per Instruction (CPI). The TWOLF benchmark shows a uniform trend throughout the time interval. More complex behavior can be seen in the MCF benchmark trend. The GCC benchmark has the most complex behavior among these three benchmarks. Besides that, it can be observed from the figure that the MCF benchmark among them has the highest L-1 data cache miss rate and the highest CPI that represent this benchmark as a very “memory consuming” and “time consuming” benchmark. This theoretical interpretation is in accordance with the other theoretical interpretation from the tables.

The simulation results show that the total number of executed instructions for all of the functionalities equals to 49,073,291,786. The simulation speed in terms of total number of instructions per second for “`sim-outorder`” functionality is 300906.2255. With considering this amount as the reference value, the simulation speed for “`sim-cache`”, “`sim-bpred`”, “`sim-eio`”, “`sim-profile`”, “`sim-safe`”, and “`sim-fast`” are 12.716, 28.112, 43.184, 43.28, 43.45, 50.905 times higher respectively. As it can be understood from the results, the “`sim-fast`” has the highest speed since more number of instructions are executed in less duration time.

The miss rate for L-1 instruction cache, L-1 data cache, and data TLB (i.e. translation lookaside buffer) in the processor in-order mode are 0.6, 0.1, and 0.07 times higher than the out-of-order mode, the miss rate for L-2 cache in the out-of-order mode

is 0.025 times higher than the in-order mode, and the miss rate for instruction TLB in both of the in-order and out-of-order modes are ~0.0. Regarding the total size of allocated memory pages, we have the same value for all of the seven architectural aspects that is 97740 KB. Finally, the instruction per cycle (IPC) parameter in the out-of-order mode that is one of the very important parameters for performance evaluation of a processor architecture is equal to 0.3425. The ideal case is having execution of one instruction per clock cycle.

B) Detailed Analysis

The total number of executed branches in both original and modified versions of the **BPRED** functionality is 12,052,458,319, which causes having instruction per branch (IPB) of 4.0716 or branch per instruction of 0.2456. This shows that approximately 25% of the instructions are branch-related. By reducing the bimodal predictor table size from 2048 to 1024, the simulation time is reduced from 5602 to 5442 and the number of misses is reduced from 1,161,245,779 to 1,161,478,917.

The total number of executed loads and stores in both original and modified versions of the **CACHE** functionality is 20,768,099,818, which causes having loads and stores per instruction of 0.4232. This shows that approximately 43% of the instructions are memory-related and the benchmark can somehow be considered as memory-intensive. By increasing the number of associativity from 1 to 4, the simulation time is reduced from 11890 to 12688, the level-1 instruction cache miss rate is reduced from 0.0008 to 0.0000, the level-1 data cache miss rate is reduced from 0.3501 to 0.3341, the level-2 miss rate is increased from 0.6005 to 0.6334, and the total memory page table accesses is increased from 237,836,123,985 to 237.836,123,987.

For **EIO** functionality, the experiment is fast forwarding the 1000 number of instructions. In both versions, the program text size is 113136 (bytes). The effect of experiment is increase of simulation time from 3691 to 3929 and the total memory page table accesses from 237836123977 to 237836123979. The verbose operation was enabled in the modified version of the **FAST** functionality that cause increase of the simulation time and total memory page table accesses from 3142 to 3373 and 237,836,123,981 to 237,836,123,983 respectively.

In the **OUT-OF-ORDER** functionality experiment, the instruction issue width is increased from 4 to 8. This causes increase of simulation time from 163085 to 318597, growth of the number executed load and store instructions from 25,978,903,722 to 25,992,887,011, improvement of instruction per cycle from 0.3425 to 0.3433. The level-1 instruction cache miss rate and level-2 cache miss rate stay the same. Also, we have reduction of level-1 data cache and total memory page table accesses from 0.3172 to

0.3171 and from 426,790,608,940 to 426,568,014,126 respectively. The miss rate for branch prediction lookups remains the same and is ~0.0755.

For the **PROFILE** functionality, the instruction profiling is enabled in the modified version. Consequently, the profile of program instructions is extracted. This enabling causes increase of simulation time from 3683 to 6434 and total memory page table accesses from 237836123993 to 237836123995. In modification of the **SAFE** functionality, we have 10 million number of instructions for execution as the limitation. This causes reduction of total number of executed loads and stores and total memory page table accesses from 20768099818 to 6663806 and 237836123981 to 54038872.

C) Conclusion

In this section, the MCF benchmark from the SPEC benchmark suite was scrutinized. In the beginning, the reference time and the reference memory footprints of this benchmark were disscused based on the provided information in Tables 01-03 and Figures 03-05. The information tagged this benchmark as a data intensive program with having many number of complex and time-hungry operations. Referring to the achieved results from running the MCF benchmark on the SimpleScalar processor architecture, there is 49,073,291,786 number of executed instructions. Also, the level-1 instruction cache miss rate of 0.0008, the level-1 data cache miss rate of 0.3501, and the level-2 miss rate of 0.6005 represent this benchmark as a data intensive and memory consuming program. Besides that, the delivered branch per instruction of 0.2456 remarks it as a relatively control-intensive program. All of these results indicate that 43% of the programs instructions are memory-related and 25% of them are branch-related. Meanwhile, the instruction per cycle (IPC) parameter in the out-of-order mode is equal to 0.3425 that is a sign of its time necessity. In overall, this benchmark can be labeled as Data-Intensive, Computation-Intensive (or Time-Intensive), and Control-Intensive in sequence.

6. Modified Commands Vs. Original Commands

********** **BPRED** **********

Original Command = `../sim-bpred -dumpconfig config_file.config /home/sh861201/SimpleScalar/simplesim-3.0/benchmark/mcf/mcf/Mcf /home/sh861201/SimpleScalar/simplesim-3.0/benchmark/mcf/mcf/inp.in`

Modified Command = `../sim-bpred -bpred:bimod 1024 -dumpconfig config_file.config /home/sh861201/SimpleScalar/simplesim-3.0/benchmark/mcf/mcf/Mcf /home/sh861201/SimpleScalar/simplesim-3.0/benchmark/mcf/mcf/inp.in`

Description = The "`sim-bpred`" command is used to simulate the branch prediction behavior. In this command, "`Mcf`" shows the executable file, "`inp.in`" shows the input file, and the argument "-dumpconfig" is leveraged for dumping the processor configuration to the "`config_file.config`" file.

********** **CACHE** **********

Original Command = `../sim-cache -dumpconfig config_file.config /home/sh861201/SimpleScalar/simplesim-3.0/benchmark/mcf/mcf/Mcf /home/sh861201/SimpleScalar/simplesim-3.0/benchmark/mcf/mcf/inp.in`

Modified Command = `../sim-cache -cache:dl1 dl1:256:32:4:l -cache:il1 il1:256:32:4:l -dumpconfig config_file.config /home/sh861201/SimpleScalar/simplesim-3.0/benchmark/mcf/mcf/Mcf /home/sh861201/SimpleScalar/simplesim-3.0/benchmark/mcf/mcf/inp.in`

Description = The "`sim-cache`" command is utilized for examining the cache behavior.

********** **EIO** **********

Original Command = `../sim-eio -dumpconfig config_file.config /home/sh861201/SimpleScalar/simplesim-3.0/benchmark/mcf/mcf/Mcf /home/sh861201/SimpleScalar/simplesim-3.0/benchmark/mcf/mcf/inp.in`

Modified Command = `../sim-eio -fastfwd 1000 -dumpconfig config_file.config /home/sh861201/SimpleScalar/simplesim-3.0/benchmark/mcf/mcf/Mcf /home/sh861201/SimpleScalar/simplesim-3.0/benchmark/mcf/mcf/inp.in`

Description = The "`sim-eio`" command is used to simulate external I/O trace generator.

********** **FAST** **********

Original Command = `../sim-fast -dumpconfig config_file.config /home/sh861201/SimpleScalar/simplesim-3.0/benchmark/mcf/mcf/Mcf /home/sh861201/SimpleScalar/simplesim-3.0/benchmark/mcf/mcf/inp.in`

Modified Command = `../sim-fast -v true -dumpconfig config_file.config /home/sh861201/SimpleScalar/simplesim-3.0/benchmark/mcf/mcf/Mcf /home/sh861201/SimpleScalar/simplesim-3.0/benchmark/mcf/mcf/inp.in`

Description = This command has usage for sample fast functional simulator implementation.

********** **OUT-OF-ORDER** **********

Original Command = `../sim-outorder -dumpconfig config_file.config /home/sh861201/SimpleScalar/simplesim-3.0/benchmark/mcf/mcf/Mcf /home/sh861201/SimpleScalar/simplesim-3.0/benchmark/mcf/mcf/inp.in`

Modified Command = `../sim-outorder -issue:width 8 -dumpconfig config_file.config /home/sh861201/SimpleScalar/simplesim-3.0/benchmark/mcf/mcf/Mcf /home/sh861201/SimpleScalar/simplesim-3.0/benchmark/mcf/mcf/inp.in`

Description = In order to simulate the out-of-order version of the SimpleScalar processor, this command can be used.

********** **PROFILE** **********

Original Command = `../sim-profile -dumpconfig config_file.config /home/sh861201/SimpleScalar/simplesim-3.0/benchmark/mcf/mcf/Mcf /home/sh861201/SimpleScalar/simplesim-3.0/benchmark/mcf/mcf/inp.in`

Modified Command = `../sim-profile -iprof true -dumpconfig config_file.config /home/sh861201/SimpleScalar/simplesim-3.0/benchmark/mcf/mcf/Mcf /home/sh861201/SimpleScalar/simplesim-3.0/benchmark/mcf/mcf/inp.in`

Description = This command is used to create a profile along with the sample functional simulator implementation.

********** **SAFE** **********

Original Command = `../sim-safe -dumpconfig config_file.config /home/sh861201/SimpleScalar/simplesim-3.0/benchmark/mcf/mcf/Mcf /home/sh861201/SimpleScalar/simplesim-3.0/benchmark/mcf/mcf/inp.in`

Modified Command = `../sim-safe -max:inst 10000000 -dumpconfig config_file.config /home/sh861201/SimpleScalar/simplesim-3.0/benchmark/mcf/mcf/Mcf /home/sh861201/SimpleScalar/simplesim-3.0/benchmark/mcf/mcf/inp.in`

Description = Lack of profile creation is the only difference between this command and the previous one.

7. Comparison of Important Results

********** **BPRED** **********

	Original	Modified
Total Simulation Time (Sec)	5602	5442
Total Number of Branch Prediction Lookups	12052458319	12052458319
Total Number of Branch Prediction Lookup Misses	1161245779	1161478917

********** **CACHE** **********

	Original	Modified
Total Simulation Time (Sec)	11890	12688
Total Number of Hits for Instruction Level 1 Cache	49031987632	49073285919
Total Number of Misses Instruction Level 1 Cache	41304154	5867
Instruction Level 1 Cache Miss Rate	0.0008	0.0000
Total Number of Hits for Data Level 1 Cache	13498624696	13498624696
Total Number of Misses for Data Level 1 Cache	7272502493	6938623810
Data Level 1 Cache Miss Rate	0.3501	0.3341
Total Number of Hits for Level 2 Cache	3468565682	3022866073
Total Number of Misses for Level 2 Cache	5213009270	5222161556
Level 2 Cache Miss Rate	0.6005	0.6334
Total Memory Page Table Accesses	237836123985	237836123987

********** **EIO** **********

	Original	Modified
Total Simulation Time (Sec)	3691	3929
Total Memory Page Table Accesses	237836123977	237836123979

********** **FAST** **********

	Original	Modified
Total Simulation Time (Sec)	3142	3373
Total Memory Page Table Accesses	237836123981	237836123983

********** **OUT-OF-ORDER** **********

	Original	Modified
Total Simulation Time (Sec)	163085	318597
Total Number of Loads and Stores Executed	25978903722	25992887011
Instructions Per Cycle	0.3425	0.3433
Total Number of Branch Prediction Lookups	15290928081	15286663565
Total Number of Branch Prediction Lookup Misses	1154505380	1154402538
Total Number of Hits for Instruction Level 1 Cache	63229170607	63176996182
Total Number of Misses for Instruction Level 1 Cache	29786749	29786748
Instruction Level 1 Cache Miss Rate	0.0005	0.0005
Total Number of Hits for Data Level 1 Cache	15296018191	15298616499
Total Number of Misses for Data Level 1 Cache	7104499699	7104501266
Data Level 1 Cache Miss Rate	0.3172	0.3171
Total Number of Hits for Level 2 Cache	3251946297	3251951709
Total Number of Misses for Level 2 Cache	5216741499	5216745069

Level 2 Cache Miss Rate	0.6160	0.6160
Total Memory Page Table Accesses	426790608940	426568014126

********** **PROFILE** **********

	Original	Modified
Total Memory Page Table Accesses	237836123993	237836123995
Total Simulation Time (Sec)	3683	6434

********** **SAFE** **********

	Original	Modified
Total Number of Instructions Executed	49073291786	10000000
Total Number of Loads and Stores Executed	20768099818	6663806
Total Memory Page Table Accesses	237836123981	54038872

8. Analysis of Arguments in Commands

1) Load Configuration from a File

Argument = `-config`

2) Dump Configuration to a File

Argument = `-dumpconfig`

3) Random Number Generator Seed (0 for timer seed)

Argument = `-seed 1`

4) Restore EIO Trace Execution from `<fname>`

Argument = `-chkpt`

5) Simulator Scheduling Priority

Argument = `-nice 0`

6) Maximum Number of Instructions to Execute

Argument = `-max:inst 0`

7) Branch Predictor Type {`nottaken`|`taken`|`bimod`|`2lev`|`comb`}

Argument = `-bpred bimod`

8) Bimodal Predictor Configuration (<table size>)

Argument = `-bpred:bimod = 2048`

9) 2-Level Predictor Configuration (`<l1size> <l2size> <hist_size> <xor>`)

Argument = `-bpred:2lev 1 1024 8 0`

10) Combining Predictor Configuration (`<meta_table_size>`)

Argument = `-bpred:comb 1024`

11) Return Address Stack Size (0 for no return stack)

Argument = `-bpred:ras = 8`

12) BTB config (`<num_sets> <associativity>`)

Argument = `-bpred:btb 512 4`

13) L-1 Data Cache Configuration, i.e., {`<config>`|none}

Argument = `-cache:dl1 = dl1:256:32:1:l`

14) L-2 Data Cache Configuration, i.e., {`<config>`|none}

Argument = `-cache:dl2 = ul2:1024:64:4:l`

15) L-1 Instruction Cache Configuration, i.e., {`<config>|dl1|dl2|none`}

Argument = `-cache:il1 = il1:256:32:1:l`

16) L-2 Instruction Cache Configuration, i.e., {`<config>|dl2|none`}

Argument = `-cache:il2 = dl2`

17) Instruction TLB Configuration, i.e., {`<config>|none`}

Argument = `-tlb:itlb = itlb:16:4096:4:l`

18) Data TLB Configuration, i.e., {`<config>|none`}

Argument = `-tlb:dtlb = dtlb:32:4096:4:l`

19) Flush Caches on System Calls

Argument = `-flush = false`

20) Convert 64-bit Instruction Addresses to 32-bit Instruction Equivalents

Argument = `-cache:icompress false`

21) Number of Instructions Skipped Before Tracing Starts

Argument = `-fastfwd 0`

22) EIO Trace File Output File Name

Argument = `-trace <null>`

23) Periodic Checkpoint Every “N” Instructions: `<base fname> <interval>`

Argument = `-perdump = <null>`

24) Specify Checkpoint File and Trigger: `<fname> <range>`

Argument = `-dump <null>`

25) Generate Pipetrace, i.e., `<fname|stdout|stderr> <range>`

Argument = `-ptrace <null>`

26) Instruction Fetch Queue Size (in Instructions)

Argument = `-fetch:ifqsize 4`

27) Extra Branch Mis-Prediction Latency

Argument = `-fetch:mplat 3`

28) Speed of Front-End of Machine Relative to Execution Core

Argument = `-fetch:speed 1`

29) Speculative Predictors Update in {`ID`|`WB`} (default non-spec)

Argument = `-bpred:spec_update = <null>`

30) Instruction Decode B/W (Instructions/Cycle)

Argument = `-decode:width 4`

31) Instruction Issue B/W (Instructions/Cycle)

Argument = `-issue:width 4`

32) Run Pipeline with In-Order Issue

Argument = `-issue:inorder false`

33) Issue Instructions Down Wrong Execution Paths

Argument = `-issue:wrongpath true`

34) Instruction Commit B/W (Instructions/Cycle)

Argument = `-commit:width = 4`

35) Register Update Unit (RUU) Size

Argument = `-ruu:size 16`

36) Load/Store Queue (LSQ) Size

Argument = `-lsq:size 8`

37) L-1 Data Cache Hit Latency (in Cycles)

Argument = `-cache:dl1lat 1`

38) L-2 Data Cache Hit Latency (in Cycles)

Argument = `-cache:dl2lat 6`

39) L-1 Instruction Cache Hit Latency (in Cycles)

Argument = `-cache:il1lat 1`

40) L-2 Instruction Cache Hit Latency (in Cycles)

Argument = `-cache:il2lat 6`

41) Flush Caches on System Calls

Argument = `-cache:flush false`

42) Memory access latency (`<first_chunk> <inter_chunk>`)

Argument = `-mem:lat 18 2`

43) Memory Access Bus Width (in Bytes)

Argument = `-mem:width 8`

44) Instruction/Data TLB Miss Latency (in Cycles)

Argument = `-tlb:lat 30`

45) Total Number of Integer ALU's Available

Argument = `-res:ialu 4`

46) Total Number of Integer Multiplier/Dividers Available

Argument = `-res:imult 1`

47) Total Number of Memory System Ports Available (to CPU)

Argument = `-res:memport 2`

48) Total Number of Floating Point ALU's Available

Argument = `-res:fpalu 4`

49) Total Number of Floating Point Multiplier/Dividers Available

Argument = `-res:fpmult 1`

50) Operate in Backward-Compatible Bugs Mode (for Testing Only)

Argument = `-bugcompat false`

51) Enable All Profile Options

Argument = `-all false`

52) Enable Instruction Class Profiling

Argument = `-iclass false`

53) Enable Instruction Profiling

Argument = `-iprof false`

54) Enable Branch Instruction Profiling

Argument = `-brprof false`

55) Enable Address Mode Profiling

Argument = `-amprof false`

56) Enable Load/Store Address Segment Profiling

Argument = `-segprof false`

57) Enable Text Symbol Profiling

Argument = `-tsymprof false`

58) Enable Text Address Profiling

Argument = `-taddrprof false`

59) Enable Data Symbol Profiling

Argument = `-dsymprof false`

60) Include Compiler-Internal Symbols During Symbol Profiling

Argument = `-internal false`

61) Profile Stat(s) Against Text Addresses (Mult uses ok)

Argument = `-pcstat = <null>`

9. Simulation Results of Original Commands

(1) sim-bpred

```
sh861201@eustis:~/SimpleScalar/simplesim-3.0/bpred_Dir$ ../sim-bpred -
dumpconfig config_file.config /home/sh861201/SimpleScalar/simplesim-
3.0/benchmark/mcf/mcf/Mcf /home/sh861201/SimpleScalar/simplesim-
3.0/benchmark/mcf/mcf/inp.in

sim-bpred: SimpleScalar/PISA Tool Set version 3.0 of August, 2003.

Copyright (c) 1994-2003 by Todd M. Austin, Ph.D. and SimpleScalar,
LLC.

All Rights Reserved. This version of SimpleScalar is licensed for
academic

non-commercial use.  No portion of this work may be used by any
commercial

entity, or for any commercial purpose, without the prior written
permission

of SimpleScalar, LLC (info@simplescalar.com).


sim: command line: ../sim-bpred -dumpconfig config_file.config
/home/sh861201/SimpleScalar/simplesim-3.0/benchmark/mcf/mcf/Mcf
/home/sh861201/SimpleScalar/simplesim-3.0/benchmark/mcf/mcf/inp.in


sim: simulation started @ Wed Feb  3 12:27:37 2016, options follow:


sim-bpred: This simulator implements a branch predictor analyzer.


# -config                     # load configuration from a file

# -dumpconfig                 # dump configuration to a file

# -h                    false # print help message
```

```
# -v                    false # verbose operation

# -d                    false # enable debug message

# -i                    false # start in Dlite debugger

-seed                       1 # random number generator seed (0 for
timer seed)

# -q                    false # initialize and terminate immediately

# -chkpt               <null> # restore EIO trace execution from
<fname>

# -redir:sim           <null> # redirect simulator output to file
(non-interactive only)

# -redir:prog          <null> # redirect simulated program output to
file

-nice                       0 # simulator scheduling priority

-max:inst                   0 # maximum number of inst's to execute

-bpred                  bimod # branch predictor type
{nottaken|taken|bimod|2lev|comb}

-bpred:bimod     2048 # bimodal predictor config (<table size>)

-bpred:2lev      1 1024 8 0 # 2-level predictor config (<l1size>
<l2size> <hist_size> <xor>)

-bpred:comb      1024 # combining predictor config (<meta_table_size>)

-bpred:ras                  8 # return address stack size (0 for no
return stack)

-bpred:btb       512 4 # BTB config (<num_sets> <associativity>)


  Branch predictor configuration examples for 2-level predictor:

    Configurations:   N, M, W, X

      N   # entries in first level (# of shift register(s))

      W   width of shift register(s)

      M   # entries in 2nd level (# of counters, or other FSM)

      X   (yes-1/no-0) xor history and address for 2nd level index
```

```
    Sample predictors:
      GAg     : 1, W, 2^W, 0
      GAp     : 1, W, M (M > 2^W), 0
      PAg     : N, W, 2^W, 0
      PAp     : N, W, M (M == 2^(N+W)), 0
      gshare  : 1, W, 2^W, 1
  Predictor `comb' combines a bimodal and a 2-level predictor.

sim: ** starting functional simulation w/ predictors **

MCF SPEC version 1.6.I
by  Andreas Loebel
Copyright (c) 1998,1999   ZIB Berlin
All Rights Reserved.

nodes                      : 16555
active arcs                : 244246
simplex iterations         : 182415
flow value                 : 8980173901
new implicit arcs          : 300000
active arcs                : 544246
simplex iterations         : 189170
flow value                 : 8910169940
new implicit arcs          : 300000
active arcs                : 844246
simplex iterations         : 216493
flow value                 : 8650168945
new implicit arcs          : 300000
```

```
active arcs               : 1144246
simplex iterations        : 261464
flow value                : 8570161464
new implicit arcs         : 300000
active arcs               : 1444246
simplex iterations        : 290615
flow value                : 8570159306
new implicit arcs         : 300000
active arcs               : 1744246
simplex iterations        : 318729
flow value                : 8570157650
new implicit arcs         : 300000
active arcs               : 2044246
simplex iterations        : 340078
flow value                : 8570156531
new implicit arcs         : 300000
active arcs               : 2344246
simplex iterations        : 354548
flow value                : 8570156010
new implicit arcs         : 77333
active arcs               : 2421579
simplex iterations        : 361819
flow value                : 8570155949
new implicit arcs         : 1100
active arcs               : 2422679
simplex iterations        : 361826
flow value                : 8570155949
checksum                  : 258659426
```

```
optimal


sim: ** simulation statistics **
sim_num_insn            49073291786 # total number of instructions
executed
sim_num_refs            20768099818 # total number of loads and stores
executed
sim_elapsed_time               5602 # total simulation time in seconds
sim_inst_rate          8759959.2620 # simulation speed (in insts/sec)
sim_num_branches        12052458319 # total number of branches
executed
sim_IPB                      4.0716 # instruction per branch
bpred_bimod.lookups     12052458319 # total number of bpred lookups
bpred_bimod.updates     12052458319 # total number of updates
bpred_bimod.addr_hits   10890361653 # total number of address-
predicted hits
bpred_bimod.dir_hits    10891212540 # total number of direction-
predicted hits (includes addr-hits)
bpred_bimod.misses       1161245779 # total number of misses
bpred_bimod.jr_hits        40089720 # total number of address-
predicted hits for JR's
bpred_bimod.jr_seen        40956662 # total number of JR's seen
bpred_bimod.jr_non_ras_hits.PP      1250351 # total number of address-
predicted hits for non-RAS JR's
bpred_bimod.jr_non_ras_seen.PP      1266947 # total number of non-RAS
JR's seen
bpred_bimod.bpred_addr_rate    0.9036 # branch address-prediction rate
(i.e., addr-hits/updates)
bpred_bimod.bpred_dir_rate    0.9037 # branch direction-prediction
rate (i.e., all-hits/updates)
bpred_bimod.bpred_jr_rate    0.9788 # JR address-prediction rate
(i.e., JR addr-hits/JRs seen)
```

```
bpred_bimod.bpred_jr_non_ras_rate.PP    0.9869 # non-RAS JR addr-pred
rate (ie, non-RAS JR hits/JRs seen)

bpred_bimod.retstack_pushes     39689717 # total number of address
pushed onto ret-addr stack

bpred_bimod.retstack_pops     39689715 # total number of address
popped off of ret-addr stack

bpred_bimod.used_ras.PP     39689715 # total number of RAS predictions
used

bpred_bimod.ras_hits.PP     38839369 # total number of RAS hits

bpred_bimod.ras_rate.PP    0.9786 # RAS prediction rate (i.e., RAS
hits/used RAS)
```

(2) sim-cache

```
sh861201@eustis:~/SimpleScalar/simplesim-3.0/cache_Dir$ ../sim-cache -
dumpconfig config_file.config /home/sh861201/SimpleScalar/simplesim-
3.0/benchmark/mcf/mcf/Mcf /home/sh861201/SimpleScalar/simplesim-
3.0/benchmark/mcf/mcf/inp.in

sim-cache: SimpleScalar/PISA Tool Set version 3.0 of August, 2003.

Copyright (c) 1994-2003 by Todd M. Austin, Ph.D. and SimpleScalar,
LLC.

All Rights Reserved. This version of SimpleScalar is licensed for
academic

non-commercial use.  No portion of this work may be used by any
commercial

entity, or for any commercial purpose, without the prior written
permission

of SimpleScalar, LLC (info@simplescalar.com).


sim: command line: ../sim-cache -dumpconfig config_file.config
/home/sh861201/SimpleScalar/simplesim-3.0/benchmark/mcf/mcf/Mcf
/home/sh861201/SimpleScalar/simplesim-3.0/benchmark/mcf/mcf/inp.in


sim: simulation started @ Tue Feb  2 16:32:36 2016, options follow:


sim-cache: This simulator implements a functional cache simulator.
Cache

statistics are generated for a user-selected cache and TLB
configuration,

which may include up to two levels of instruction and data cache (with
any

levels unified), and one level of instruction and data TLBs.  No
timing

information is generated.
```

```
# -config                     # load configuration from a file
# -dumpconfig                 # dump configuration to a file
# -h                    false # print help message
# -v                    false # verbose operation
# -d                    false # enable debug message
# -i                    false # start in Dlite debugger
-seed                       1 # random number generator seed (0 for
timer seed)
# -q                    false # initialize and terminate immediately
# -chkpt                <null> # restore EIO trace execution from
<fname>
# -redir:sim            <null> # redirect simulator output to file
(non-interactive only)
# -redir:prog           <null> # redirect simulated program output to
file
-nice                       0 # simulator scheduling priority
-max:inst                   0 # maximum number of inst's to execute
-cache:dl1       dl1:256:32:1:l # l1 data cache config, i.e.,
{<config>|none}
-cache:dl2       ul2:1024:64:4:l # l2 data cache config, i.e.,
{<config>|none}
-cache:il1       il1:256:32:1:l # l1 inst cache config, i.e.,
{<config>|dl1|dl2|none}
-cache:il2                 dl2 # l2 instruction cache config, i.e.,
{<config>|dl2|none}
-tlb:itlb        itlb:16:4096:4:l # instruction TLB config, i.e.,
{<config>|none}
-tlb:dtlb        dtlb:32:4096:4:l # data TLB config, i.e.,
{<config>|none}
-flush                false # flush caches on system calls
```

```
-cache:icompress        false # convert 64-bit inst addresses to 32-
bit inst equivalents
# -pcstat              <null> # profile stat(s) against text addr's
(mult uses ok)

  The cache config parameter <config> has the following format:

    <name>:<nsets>:<bsize>:<assoc>:<repl>

    <name>   - name of the cache being defined
    <nsets>  - number of sets in the cache
    <bsize>  - block size of the cache
    <assoc>  - associativity of the cache
    <repl>   - block replacement strategy, 'l'-LRU, 'f'-FIFO, 'r'-
random

    Examples:   -cache:dl1 dl1:4096:32:1:l
                -dtlb dtlb:128:4096:32:r

  Cache levels can be unified by pointing a level of the instruction
cache
  hierarchy at the data cache hiearchy using the "dl1" and "dl2" cache
  configuration arguments.  Most sensible combinations are supported,
e.g.,

    A unified l2 cache (il2 is pointed at dl2):
      -cache:il1 il1:128:64:1:l -cache:il2 dl2
      -cache:dl1 dl1:256:32:1:l -cache:dl2 ul2:1024:64:2:l

    Or, a fully unified cache hierarchy (il1 pointed at dl1):
```

```
      -cache:il1 dl1
      -cache:dl1 ul1:256:32:1:l -cache:dl2 ul2:1024:64:2:l

sim: ** starting functional simulation w/ caches **

MCF SPEC version 1.6.I
by  Andreas Loebel
Copyright (c) 1998,1999   ZIB Berlin
All Rights Reserved.

nodes                      : 16555
active arcs                : 244246
simplex iterations         : 182415
flow value                 : 8980173901
new implicit arcs          : 300000
active arcs                : 544246
simplex iterations         : 189170
flow value                 : 8910169940
new implicit arcs          : 300000
active arcs                : 844246
simplex iterations         : 216493
flow value                 : 8650168945
new implicit arcs          : 300000
active arcs                : 1144246
simplex iterations         : 261464
flow value                 : 8570161464
new implicit arcs          : 300000
active arcs                : 1444246
```

```
simplex iterations         : 290615
flow value                 : 8570159306
new implicit arcs          : 300000
active arcs                : 1744246
simplex iterations         : 318729
flow value                 : 8570157650
new implicit arcs          : 300000
active arcs                : 2044246
simplex iterations         : 340078
flow value                 : 8570156531
new implicit arcs          : 300000
active arcs                : 2344246
simplex iterations         : 354548
flow value                 : 8570156010
new implicit arcs          : 77333
active arcs                : 2421579
simplex iterations         : 361819
flow value                 : 8570155949
new implicit arcs          : 1100
active arcs                : 2422679
simplex iterations         : 361826
flow value                 : 8570155949
checksum                   : 258659426
optimal


sim: ** simulation statistics **
sim_num_insn            49073291786 # total number of instructions
executed
```

```
sim_num_refs            20768099818 # total number of loads and stores
executed
sim_elapsed_time              11890 # total simulation time in seconds
sim_inst_rate          4127274.3302 # simulation speed (in insts/sec)
il1.accesses            49073291786 # total number of accesses
il1.hits                49031987632 # total number of hits
il1.misses                 41304154 # total number of misses
il1.replacements           41303898 # total number of replacements
il1.writebacks                    0 # total number of writebacks
il1.invalidations                 0 # total number of invalidations
il1.miss_rate                0.0008 # miss rate (i.e., misses/ref)
il1.repl_rate                0.0008 # replacement rate (i.e.,
repls/ref)
il1.wb_rate                  0.0000 # writeback rate (i.e., wrbks/ref)
il1.inv_rate                 0.0000 # invalidation rate (i.e.,
invs/ref)
dl1.accesses            20771127189 # total number of accesses
dl1.hits                13498624696 # total number of hits
dl1.misses               7272502493 # total number of misses
dl1.replacements         7272502237 # total number of replacements
dl1.writebacks           1367768305 # total number of writebacks
dl1.invalidations                 0 # total number of invalidations
dl1.miss_rate                0.3501 # miss rate (i.e., misses/ref)
dl1.repl_rate                0.3501 # replacement rate (i.e.,
repls/ref)
dl1.wb_rate                  0.0658 # writeback rate (i.e., wrbks/ref)
dl1.inv_rate                 0.0000 # invalidation rate (i.e.,
invs/ref)
ul2.accesses             8681574952 # total number of accesses
ul2.hits                 3468565682 # total number of hits
```

```
ul2.misses               5213009270 # total number of misses
ul2.replacements         5213005174 # total number of replacements
ul2.writebacks           1207279599 # total number of writebacks
ul2.invalidations                 0 # total number of invalidations
ul2.miss_rate                0.6005 # miss rate (i.e., misses/ref)
ul2.repl_rate                0.6005 # replacement rate (i.e.,
repls/ref)
ul2.wb_rate                  0.1391 # writeback rate (i.e., wrbks/ref)
ul2.inv_rate                 0.0000 # invalidation rate (i.e.,
invs/ref)
itlb.accesses           49073291786 # total number of accesses
itlb.hits               49073291759 # total number of hits
itlb.misses                      27 # total number of misses
itlb.replacements                 0 # total number of replacements
itlb.writebacks                   0 # total number of writebacks
itlb.invalidations                0 # total number of invalidations
itlb.miss_rate               0.0000 # miss rate (i.e., misses/ref)
itlb.repl_rate               0.0000 # replacement rate (i.e.,
repls/ref)
itlb.wb_rate                 0.0000 # writeback rate (i.e., wrbks/ref)
itlb.inv_rate                0.0000 # invalidation rate (i.e.,
invs/ref)
dtlb.accesses           20771127189 # total number of accesses
dtlb.hits               19090713234 # total number of hits
dtlb.misses              1680413955 # total number of misses
dtlb.replacements        1680413827 # total number of replacements
dtlb.writebacks           365182120 # total number of writebacks
dtlb.invalidations                0 # total number of invalidations
dtlb.miss_rate               0.0809 # miss rate (i.e., misses/ref)
```

```
dtlb.repl_rate                0.0809 # replacement rate (i.e.,
repls/ref)
dtlb.wb_rate                  0.0176 # writeback rate (i.e., wrbks/ref)
dtlb.inv_rate                 0.0000 # invalidation rate (i.e.,
invs/ref)
ld_text_base             0x00400000 # program text (code) segment base
ld_text_size                  113136 # program text (code) size in
bytes
ld_data_base             0x10000000 # program initialized data segment
base
ld_data_size                   19060 # program init'ed `.data' and
uninit'ed `.bss' size in bytes
ld_stack_base            0x7fffc000 # program stack segment base
(highest address in stack)
ld_stack_size                  16384 # program initial stack size
ld_prog_entry            0x00400140 # program entry point (initial PC)
ld_environ_base          0x7fff8000 # program environment base address
address
ld_target_big_endian               0 # target executable endian-ness,
non-zero if big endian
mem.page_count                 24435 # total number of pages allocated
mem.page_mem                   97740k # total size of memory pages
allocated
mem.ptab_misses            10414410 # total first level page table
misses
mem.ptab_accesses      237836123985 # total page table accesses
mem.ptab_miss_rate           0.0000 # first level page table miss rate
```

(3) sim-eio

```
sh861201@eustis:~/SimpleScalar/simplesim-3.0/eio_Dir$ ../sim-eio -
dumpconfig config_file.config /home/sh861201/SimpleScalar/simplesim-
3.0/benchmark/mcf/mcf/Mcf /home/sh861201/SimpleScalar/simplesim-
3.0/benchmark/mcf/mcf/inp.in

sim-eio: SimpleScalar/PISA Tool Set version 3.0 of August, 2003.

Copyright (c) 1994-2003 by Todd M. Austin, Ph.D. and SimpleScalar,
LLC.

All Rights Reserved. This version of SimpleScalar is licensed for
academic

non-commercial use.  No portion of this work may be used by any
commercial

entity, or for any commercial purpose, without the prior written
permission

of SimpleScalar, LLC (info@simplescalar.com).


sim: command line: ../sim-eio -dumpconfig config_file.config
/home/sh861201/SimpleScalar/simplesim-3.0/benchmark/mcf/mcf/Mcf
/home/sh861201/SimpleScalar/simplesim-3.0/benchmark/mcf/mcf/inp.in


sim: simulation started @ Wed Feb  3 14:20:30 2016, options follow:


sim-eio: This simulator implements simulator support for generating

external event traces (EIO traces) and checkpoint files.  External

event traces capture one execution of a program, and allow it to be

packaged into a single file for later re-execution.  EIO trace
executions

are 100% reproducible between subsequent executions (on the same
platform.

This simulator also provides functionality to generate checkpoints at
```

arbitrary points within an external event trace (EIO) execution. The checkpoint file (along with the EIO trace) can be used to start any SimpleScalar simulator in the middle of a program execution.

```
# -config                     # load configuration from a file
# -dumpconfig                 # dump configuration to a file
# -h                    false # print help message
# -v                    false # verbose operation
# -d                    false # enable debug message
# -i                    false # start in Dlite debugger
-seed                       1 # random number generator seed (0 for
timer seed)
# -q                    false # initialize and terminate immediately
# -chkpt               <null> # restore EIO trace execution from
<fname>
# -redir:sim           <null> # redirect simulator output to file
(non-interactive only)
# -redir:prog          <null> # redirect simulated program output to
file
-nice                       0 # simulator scheduling priority
-max:inst                   0 # maximum number of inst's to execute
-fastfwd                    0 # number of insts skipped before tracing
starts
# -trace               <null> # EIO trace file output file name
# -perdump             <null> # periodic checkpoint every n
instructions: <base fname> <interval>
# -dump                <null> # specify checkpoint file and trigger:
<fname> <range>
```

```
  Checkpoint range triggers are formatted as follows:

    {{@|#}<start>}:{{@|#|+}<end>}

  Both ends of the range are optional, if neither are specified, the
range

  triggers immediately.  Ranges that start with a `@' designate an
address

  range to trigger on, those that start with an `#' designate a cycle
count

  trigger.  All other ranges represent an instruction count range.
The

  second argument, if specified with a `+', indicates a value relative

  to the first argument, e.g., 1000:+100 == 1000:1100.

    Examples:   -ptrace FOO.trc #0:#1000

                -ptrace BAR.trc @2000:

                -ptrace BLAH.trc :1500

                -ptrace UXXE.trc :

sim: ** starting functional simulation **

MCF SPEC version 1.6.I

by  Andreas Loebel

Copyright (c) 1998,1999   ZIB Berlin

All Rights Reserved.

nodes                       : 16555
```

```
active arcs               : 244246
simplex iterations        : 182415
flow value                : 8980173901
new implicit arcs         : 300000
active arcs               : 544246
simplex iterations        : 189170
flow value                : 8910169940
new implicit arcs         : 300000
active arcs               : 844246
simplex iterations        : 216493
flow value                : 8650168945
new implicit arcs         : 300000
active arcs               : 1144246
simplex iterations        : 261464
flow value                : 8570161464
new implicit arcs         : 300000
active arcs               : 1444246
simplex iterations        : 290615
flow value                : 8570159306
new implicit arcs         : 300000
active arcs               : 1744246
simplex iterations        : 318729
flow value                : 8570157650
new implicit arcs         : 300000
active arcs               : 2044246
simplex iterations        : 340078
flow value                : 8570156531
new implicit arcs         : 300000
```

```
active arcs               : 2344246
simplex iterations        : 354548
flow value                : 8570156010
new implicit arcs         : 77333
active arcs               : 2421579
simplex iterations        : 361819
flow value                : 8570155949
new implicit arcs         : 1100
active arcs               : 2422679
simplex iterations        : 361826
flow value                : 8570155949
checksum                  : 258659426
optimal


sim: ** simulation statistics **
sim_num_insn            49073291786 # total number of instructions
executed
sim_num_refs            20768099818 # total number of loads and stores
executed
sim_elapsed_time               3691 # total simulation time in seconds
sim_inst_rate          13295391.9767 # simulation speed (in insts/sec)
ld_text_base             0x00400000 # program text (code) segment base
ld_text_size                 113136 # program text (code) size in
bytes
ld_data_base             0x10000000 # program initialized data segment
base
ld_data_size                  19060 # program init'ed `.data' and
uninit'ed `.bss' size in bytes
ld_stack_base            0x7fffc000 # program stack segment base
(highest address in stack)
```

```
ld_stack_size                16384 # program initial stack size
ld_prog_entry           0x00400140 # program entry point (initial PC)
ld_environ_base         0x7fff8000 # program environment base address
address
ld_target_big_endian             0 # target executable endian-ness,
non-zero if big endian
mem.page_count               24435 # total number of pages allocated
mem.page_mem                 97740k # total size of memory pages
allocated
mem.ptab_misses              10414410 # total first level page table
misses
mem.ptab_accesses      237836123977 # total page table accesses
mem.ptab_miss_rate           0.0000 # first level page table miss rate
```

(4) sim-fast

```
sh861201@eustis:~/SimpleScalar/simplesim-3.0/fast_Dir$ ../sim-fast -
dumpconfig config_file.config /home/sh861201/SimpleScalar/simplesim-
3.0/benchmark/mcf/mcf/Mcf /home/sh861201/SimpleScalar/simplesim-
3.0/benchmark/mcf/mcf/inp.in

sim-fast: SimpleScalar/PISA Tool Set version 3.0 of August, 2003.

Copyright (c) 1994-2003 by Todd M. Austin, Ph.D. and SimpleScalar,
LLC.

All Rights Reserved. This version of SimpleScalar is licensed for
academic

non-commercial use.  No portion of this work may be used by any
commercial

entity, or for any commercial purpose, without the prior written
permission

of SimpleScalar, LLC (info@simplescalar.com).


sim: command line: ../sim-fast -dumpconfig config_file.config
/home/sh861201/SimpleScalar/simplesim-3.0/benchmark/mcf/mcf/Mcf
/home/sh861201/SimpleScalar/simplesim-3.0/benchmark/mcf/mcf/inp.in


sim: simulation started @ Wed Feb  3 15:52:24 2016, options follow:


sim-fast: This simulator implements a very fast functional simulator.
This

functional simulator implementation is much more difficult to digest
than

the simpler, cleaner sim-safe functional simulator.  By default, this

simulator performs no instruction error checking, as a result, any

instruction errors will manifest as simulator execution errors,
possibly

causing sim-fast to execute incorrectly or dump core.  Such is the
```

```
price we pay for speed!!!!


# -config                     # load configuration from a file
# -dumpconfig                 # dump configuration to a file
# -h                    false # print help message
# -v                    false # verbose operation
# -d                    false # enable debug message
# -i                    false # start in Dlite debugger
-seed                       1 # random number generator seed (0 for
timer seed)
# -q                    false # initialize and terminate immediately
# -chkpt               <null> # restore EIO trace execution from
<fname>
# -redir:sim           <null> # redirect simulator output to file
(non-interactive only)
# -redir:prog          <null> # redirect simulated program output to
file
-nice                       0 # simulator scheduling priority


sim: ** starting *fast* functional simulation **


MCF SPEC version 1.6.I
by  Andreas Loebel
Copyright (c) 1998,1999   ZIB Berlin
All Rights Reserved.


nodes                     : 16555
active arcs               : 244246
simplex iterations        : 182415
```

```
flow value                : 8980173901
new implicit arcs         : 300000
active arcs               : 544246
simplex iterations        : 189170
flow value                : 8910169940
new implicit arcs         : 300000
active arcs               : 844246
simplex iterations        : 216493
flow value                : 8650168945
new implicit arcs         : 300000
active arcs               : 1144246
simplex iterations        : 261464
flow value                : 8570161464
new implicit arcs         : 300000
active arcs               : 1444246
simplex iterations        : 290615
flow value                : 8570159306
new implicit arcs         : 300000
active arcs               : 1744246
simplex iterations        : 318729
flow value                : 8570157650
new implicit arcs         : 300000
active arcs               : 2044246
simplex iterations        : 340078
flow value                : 8570156531
new implicit arcs         : 300000
active arcs               : 2344246
simplex iterations        : 354548
```

```
flow value                  : 8570156010
new implicit arcs           : 77333
active arcs                 : 2421579
simplex iterations          : 361819
flow value                  : 8570155949
new implicit arcs           : 1100
active arcs                 : 2422679
simplex iterations          : 361826
flow value                  : 8570155949
checksum                    : 258659426
optimal


sim: ** simulation statistics **
sim_num_insn            49073291786 # total number of instructions
executed
sim_elapsed_time               3142 # total simulation time in seconds
sim_inst_rate          15618488.7925 # simulation speed (in insts/sec)
ld_text_base             0x00400000 # program text (code) segment base
ld_text_size                 113136 # program text (code) size in
bytes
ld_data_base             0x10000000 # program initialized data segment
base
ld_data_size                  19060 # program init'ed `.data' and
uninit'ed `.bss' size in bytes
ld_stack_base            0x7fffc000 # program stack segment base
(highest address in stack)
ld_stack_size                 16384 # program initial stack size
ld_prog_entry            0x00400140 # program entry point (initial PC)
ld_environ_base          0x7fff8000 # program environment base address
address
```

```
ld_target_big_endian              0 # target executable endian-ness,
non-zero if big endian
mem.page_count                24435 # total number of pages allocated
mem.page_mem                  97740k # total size of memory pages
allocated
mem.ptab_misses            10414410 # total first level page table
misses
mem.ptab_accesses      237836123981 # total page table accesses
mem.ptab_miss_rate           0.0000 # first level page table miss rate
```

(5) sim-outorder

```
sh861201@eustis:~/SimpleScalar/simplesim-3.0/outorder_Dir$ ../sim-
outorder -dumpconfig config_file.config
/home/sh861201/SimpleScalar/simplesim-3.0/benchmark/mcf/mcf/Mcf
/home/sh861201/SimpleScalar/simplesim-3.0/benchmark/mcf/mcf/inp.in

sim-outorder: SimpleScalar/PISA Tool Set version 3.0 of August, 2003.

Copyright (c) 1994-2003 by Todd M. Austin, Ph.D. and SimpleScalar,
LLC.

All Rights Reserved. This version of SimpleScalar is licensed for
academic

non-commercial use.  No portion of this work may be used by any
commercial

entity, or for any commercial purpose, without the prior written
permission

of SimpleScalar, LLC (info@simplescalar.com).


sim: command line: ../sim-outorder -dumpconfig config_file.config
/home/sh861201/SimpleScalar/simplesim-3.0/benchmark/mcf/mcf/Mcf
/home/sh861201/SimpleScalar/simplesim-3.0/benchmark/mcf/mcf/inp.in


sim: simulation started @ Wed Feb  3 16:48:37 2016, options follow:


sim-outorder: This simulator implements a very detailed out-of-order
issue

superscalar processor with a two-level memory system and speculative

execution support.  This simulator is a performance simulator,
tracking the

latency of all pipeline operations.


# -config                     # load configuration from a file
```

```
# -dumpconfig                # dump configuration to a file
# -h                    false # print help message
# -v                    false # verbose operation
# -d                    false # enable debug message
# -i                    false # start in Dlite debugger
-seed                       1 # random number generator seed (0 for
timer seed)
# -q                    false # initialize and terminate immediately
# -chkpt               <null> # restore EIO trace execution from
<fname>
# -redir:sim           <null> # redirect simulator output to file
(non-interactive only)
# -redir:prog          <null> # redirect simulated program output to
file
-nice                       0 # simulator scheduling priority
-max:inst                   0 # maximum number of inst's to execute
-fastfwd                    0 # number of insts skipped before timing
starts
# -ptrace              <null> # generate pipetrace, i.e.,
<fname|stdout|stderr> <range>
-fetch:ifqsize              4 # instruction fetch queue size (in
insts)
-fetch:mplat                3 # extra branch mis-prediction latency
-fetch:speed                1 # speed of front-end of machine relative
to execution core
-bpred                  bimod # branch predictor type
{nottaken|taken|perfect|bimod|2lev|comb}
-bpred:bimod     2048 # bimodal predictor config (<table size>)
-bpred:2lev      1 1024 8 0 # 2-level predictor config (<l1size>
<l2size> <hist_size> <xor>)
-bpred:comb      1024 # combining predictor config (<meta_table_size>)
```

```
-bpred:ras                  8 # return address stack size (0 for no
return stack)

-bpred:btb       512 4 # BTB config (<num_sets> <associativity>)

# -bpred:spec_update       <null> # speculative predictors update in
{ID|WB} (default non-spec)

-decode:width               4 # instruction decode B/W (insts/cycle)

-issue:width                4 # instruction issue B/W (insts/cycle)

-issue:inorder          false # run pipeline with in-order issue

-issue:wrongpath         true # issue instructions down wrong
execution paths

-commit:width               4 # instruction commit B/W (insts/cycle)

-ruu:size                  16 # register update unit (RUU) size

-lsq:size                   8 # load/store queue (LSQ) size

-cache:dl1       dl1:128:32:4:l # l1 data cache config, i.e.,
{<config>|none}

-cache:dl1lat               1 # l1 data cache hit latency (in cycles)

-cache:dl2       ul2:1024:64:4:l # l2 data cache config, i.e.,
{<config>|none}

-cache:dl2lat               6 # l2 data cache hit latency (in cycles)

-cache:il1       il1:512:32:1:l # l1 inst cache config, i.e.,
{<config>|dl1|dl2|none}

-cache:il1lat               1 # l1 instruction cache hit latency (in
cycles)

-cache:il2                 dl2 # l2 instruction cache config, i.e.,
{<config>|dl2|none}

-cache:il2lat               6 # l2 instruction cache hit latency (in
cycles)

-cache:flush            false # flush caches on system calls

-cache:icompress        false # convert 64-bit inst addresses to 32-
bit inst equivalents

-mem:lat         18 2 # memory access latency (<first_chunk>
<inter_chunk>)
```

```
-mem:width                  8 # memory access bus width (in bytes)

-tlb:itlb        itlb:16:4096:4:l # instruction TLB config, i.e.,
{<config>|none}

-tlb:dtlb        dtlb:32:4096:4:l # data TLB config, i.e.,
{<config>|none}

-tlb:lat                   30 # inst/data TLB miss latency (in cycles)

-res:ialu                   4 # total number of integer ALU's
available

-res:imult                  1 # total number of integer
multiplier/dividers available

-res:memport                2 # total number of memory system ports
available (to CPU)

-res:fpalu                  4 # total number of floating point ALU's
available

-res:fpmult                 1 # total number of floating point
multiplier/dividers available

# -pcstat              <null> # profile stat(s) against text addr's
(mult uses ok)

-bugcompat               false # operate in backward-compatible bugs
mode (for testing only)


  Pipetrace range arguments are formatted as follows:


    {{@|#}<start>}:{{@|#|+}<end>}


  Both ends of the range are optional, if neither are specified, the
entire

  execution is traced.  Ranges that start with a `@' designate an
address

  range to be traced, those that start with an `#' designate a cycle
count
```

```
  range.  All other range values represent an instruction count range.
The

  second argument, if specified with a `+', indicates a value relative

  to the first argument, e.g., 1000:+100 == 1000:1100.  Program
symbols may

  be used in all contexts.


    Examples:   -ptrace FOO.trc #0:#1000

                -ptrace BAR.trc @2000:

                -ptrace BLAH.trc :1500

                -ptrace UXXE.trc :

                -ptrace FOOBAR.trc @main:+278


  Branch predictor configuration examples for 2-level predictor:

    Configurations:   N, M, W, X

      N   # entries in first level (# of shift register(s))

      W   width of shift register(s)

      M   # entries in 2nd level (# of counters, or other FSM)

      X   (yes-1/no-0) xor history and address for 2nd level index

    Sample predictors:

      GAg     : 1, W, 2^W, 0

      GAp     : 1, W, M (M > 2^W), 0

      PAg     : N, W, 2^W, 0

      PAp     : N, W, M (M == 2^(N+W)), 0

      gshare  : 1, W, 2^W, 1

  Predictor `comb' combines a bimodal and a 2-level predictor.


  The cache config parameter <config> has the following format:
```

```
    <name>:<nsets>:<bsize>:<assoc>:<repl>

    <name>   - name of the cache being defined
    <nsets>  - number of sets in the cache
    <bsize>  - block size of the cache
    <assoc>  - associativity of the cache
    <repl>   - block replacement strategy, 'l'-LRU, 'f'-FIFO, 'r'-
random

    Examples:   -cache:dl1 dl1:4096:32:1:l
                -dtlb dtlb:128:4096:32:r

  Cache levels can be unified by pointing a level of the instruction
cache
  hierarchy at the data cache hiearchy using the "dl1" and "dl2" cache
  configuration arguments.  Most sensible combinations are supported,
e.g.,

    A unified l2 cache (il2 is pointed at dl2):
      -cache:il1 il1:128:64:1:l -cache:il2 dl2
      -cache:dl1 dl1:256:32:1:l -cache:dl2 ul2:1024:64:2:l

    Or, a fully unified cache hierarchy (il1 pointed at dl1):
      -cache:il1 dl1
      -cache:dl1 ul1:256:32:1:l -cache:dl2 ul2:1024:64:2:l

sim: ** starting performance simulation **

MCF SPEC version 1.6.I
```

by Andreas Loebel

Copyright (c) 1998,1999   ZIB Berlin

All Rights Reserved.

```
nodes                      : 16555
active arcs                : 244246
simplex iterations         : 182415
flow value                 : 8980173901
new implicit arcs          : 300000
active arcs                : 544246
simplex iterations         : 189170
flow value                 : 8910169940
new implicit arcs          : 300000
active arcs                : 844246
simplex iterations         : 216493
flow value                 : 8650168945
new implicit arcs          : 300000
active arcs                : 1144246
simplex iterations         : 261464
flow value                 : 8570161464
new implicit arcs          : 300000
active arcs                : 1444246
simplex iterations         : 290615
flow value                 : 8570159306
new implicit arcs          : 300000
active arcs                : 1744246
simplex iterations         : 318729
flow value                 : 8570157650
```

```
new implicit arcs          : 300000
active arcs                : 2044246
simplex iterations         : 340078
flow value                 : 8570156531
new implicit arcs          : 300000
active arcs                : 2344246
simplex iterations         : 354548
flow value                 : 8570156010
new implicit arcs          : 77333
active arcs                : 2421579
simplex iterations         : 361819
flow value                 : 8570155949
new implicit arcs          : 1100
active arcs                : 2422679
simplex iterations         : 361826
flow value                 : 8570155949
checksum                   : 258659426
optimal


sim: ** simulation statistics **
sim_num_insn            49073291786 # total number of instructions
committed
sim_num_refs            20768099818 # total number of loads and stores
committed
sim_num_loads           18041672686 # total number of loads committed
sim_num_stores         2726427132.0000 # total number of stores
committed
sim_num_branches        12052458319 # total number of branches
committed
```

```
sim_elapsed_time                163085 # total simulation time in seconds
sim_inst_rate                300906.2255 # simulation speed (in insts/sec)
sim_total_insn                58786702113 # total number of instructions
executed
sim_total_refs                25992887011 # total number of loads and stores
executed
sim_total_loads               22737122025 # total number of loads executed
sim_total_stores           3255764986.0000 # total number of stores
executed
sim_total_branches            14110463801 # total number of branches
executed
sim_cycle                  143296637320 # total simulation time in cycles
sim_IPC                           0.3425 # instructions per cycle
sim_CPI                           2.9201 # cycles per instruction
sim_exec_BW                       0.4102 # total instructions (mis-spec +
committed) per cycle
sim_IPB                           4.0716 # instruction per branch
IFQ_count                  543163217883 # cumulative IFQ occupancy
IFQ_fcount                 132402474202 # cumulative IFQ full count
ifq_occupancy                     3.7905 # avg IFQ occupancy (insn's)
ifq_rate                          0.4102 # avg IFQ dispatch rate
(insn/cycle)
ifq_latency                       9.2396 # avg IFQ occupant latency
(cycle's)
ifq_full                          0.9240 # fraction of time (cycle's) IFQ
was full
RUU_count                  2109935503690 # cumulative RUU occupancy
RUU_fcount                 100780909996 # cumulative RUU full count
ruu_occupancy                    14.7242 # avg RUU occupancy (insn's)
ruu_rate                          0.4102 # avg RUU dispatch rate
(insn/cycle)
```

```
ruu_latency                35.8914 # avg RUU occupant latency
(cycle's)
ruu_full                    0.7033 # fraction of time (cycle's) RUU
was full
LSQ_count              971656506743 # cumulative LSQ occupancy
LSQ_fcount              66817724062 # cumulative LSQ full count
lsq_occupancy               6.7807 # avg LSQ occupancy (insn's)
lsq_rate                    0.4102 # avg LSQ dispatch rate
(insn/cycle)
lsq_latency                16.5285 # avg LSQ occupant latency
(cycle's)
lsq_full                    0.4663 # fraction of time (cycle's) LSQ
was full
sim_slip               6747556112725567395 # total number of slip
cycles
avg_sim_slip           137499561.7199 # the average slip between issue
and retirement
bpred_bimod.lookups     15290928081 # total number of bpred lookups
bpred_bimod.updates     12052458319 # total number of updates
bpred_bimod.addr_hits   10896784509 # total number of address-
predicted hits
bpred_bimod.dir_hits    10897952939 # total number of direction-
predicted hits (includes addr-hits)
bpred_bimod.misses       1154505380 # total number of misses
bpred_bimod.jr_hits        39772187 # total number of address-
predicted hits for JR's
bpred_bimod.jr_seen        40956662 # total number of JR's seen
bpred_bimod.jr_non_ras_hits.PP      1250351 # total number of address-
predicted hits for non-RAS JR's
bpred_bimod.jr_non_ras_seen.PP      1266947 # total number of non-RAS
JR's seen
```

```
bpred_bimod.bpred_addr_rate    0.9041 # branch address-prediction rate
(i.e., addr-hits/updates)

bpred_bimod.bpred_dir_rate    0.9042 # branch direction-prediction
rate (i.e., all-hits/updates)

bpred_bimod.bpred_jr_rate    0.9711 # JR address-prediction rate
(i.e., JR addr-hits/JRs seen)

bpred_bimod.bpred_jr_non_ras_rate.PP    0.9869 # non-RAS JR addr-pred
rate (ie, non-RAS JR hits/JRs seen)

bpred_bimod.retstack_pushes     73535045 # total number of address
pushed onto ret-addr stack

bpred_bimod.retstack_pops     58199448 # total number of address
popped off of ret-addr stack

bpred_bimod.used_ras.PP     39689715 # total number of RAS predictions
used

bpred_bimod.ras_hits.PP     38521836 # total number of RAS hits

bpred_bimod.ras_rate.PP    0.9706 # RAS prediction rate (i.e., RAS
hits/used RAS)

il1.accesses            63258957356 # total number of accesses

il1.hits                63229170607 # total number of hits

il1.misses                 29786749 # total number of misses

il1.replacements           29786242 # total number of replacements

il1.writebacks                    0 # total number of writebacks

il1.invalidations                 0 # total number of invalidations

il1.miss_rate                0.0005 # miss rate (i.e., misses/ref)

il1.repl_rate                0.0005 # replacement rate (i.e.,
repls/ref)

il1.wb_rate                  0.0000 # writeback rate (i.e., wrbks/ref)

il1.inv_rate                 0.0000 # invalidation rate (i.e.,
invs/ref)

dl1.accesses            22403117765 # total number of accesses

dl1.hits                15298616499 # total number of hits
```

```
dl1.misses               7104501266 # total number of misses
dl1.replacements         7104500754 # total number of replacements
dl1.writebacks           1334405193 # total number of writebacks
dl1.invalidations                 0 # total number of invalidations
dl1.miss_rate                0.3171 # miss rate (i.e., misses/ref)
dl1.repl_rate                0.3171 # replacement rate (i.e.,
repls/ref)
dl1.wb_rate                  0.0596 # writeback rate (i.e., wrbks/ref)
dl1.inv_rate                 0.0000 # invalidation rate (i.e.,
invs/ref)
ul2.accesses             8468693208 # total number of accesses
ul2.hits                 3251951709 # total number of hits
ul2.misses               5216741499 # total number of misses
ul2.replacements         5216737403 # total number of replacements
ul2.writebacks           1207172414 # total number of writebacks
ul2.invalidations                 0 # total number of invalidations
ul2.miss_rate                0.6160 # miss rate (i.e., misses/ref)
ul2.repl_rate                0.6160 # replacement rate (i.e.,
repls/ref)
ul2.wb_rate                  0.1425 # writeback rate (i.e., wrbks/ref)
ul2.inv_rate                 0.0000 # invalidation rate (i.e.,
invs/ref)
itlb.accesses           63258957356 # total number of accesses
itlb.hits               63258957329 # total number of hits
itlb.misses                      27 # total number of misses
itlb.replacements                 0 # total number of replacements
itlb.writebacks                   0 # total number of writebacks
itlb.invalidations                0 # total number of invalidations
itlb.miss_rate               0.0000 # miss rate (i.e., misses/ref)
```

```
itlb.repl_rate                0.0000 # replacement rate (i.e.,
repls/ref)
itlb.wb_rate                  0.0000 # writeback rate (i.e., wrbks/ref)
itlb.inv_rate                 0.0000 # invalidation rate (i.e.,
invs/ref)
dtlb.accesses           22426137324 # total number of accesses
dtlb.hits               20739675328 # total number of hits
dtlb.misses              1686461996 # total number of misses
dtlb.replacements        1686461868 # total number of replacements
dtlb.writebacks                 0 # total number of writebacks
dtlb.invalidations              0 # total number of invalidations
dtlb.miss_rate                0.0752 # miss rate (i.e., misses/ref)
dtlb.repl_rate                0.0752 # replacement rate (i.e.,
repls/ref)
dtlb.wb_rate                  0.0000 # writeback rate (i.e., wrbks/ref)
dtlb.inv_rate                 0.0000 # invalidation rate (i.e.,
invs/ref)
sim_invalid_addrs               0 # total non-speculative bogus
addresses seen (debug var)
ld_text_base             0x00400000 # program text (code) segment base
ld_text_size                113136 # program text (code) size in
bytes
ld_data_base             0x10000000 # program initialized data segment
base
ld_data_size                 19060 # program init'ed `.data' and
uninit'ed `.bss' size in bytes
ld_stack_base            0x7fffc000 # program stack segment base
(highest address in stack)
ld_stack_size                16384 # program initial stack size
ld_prog_entry            0x00400140 # program entry point (initial PC)
```

```
ld_environ_base          0x7fff8000 # program environment base address
address
ld_target_big_endian             0 # target executable endian-ness,
non-zero if big endian
mem.page_count               24435 # total number of pages allocated
mem.page_mem                 97740k # total size of memory pages
allocated
mem.ptab_misses             6774601 # total first level page table
misses
mem.ptab_accesses      426790608940 # total page table accesses
mem.ptab_miss_rate           0.0000 # first level page table miss rate
```

(6) sim-profile

```
sh861201@eustis:~/SimpleScalar/simplesim-3.0/profile_Dir$ ../sim-
profile -dumpconfig config_file.config
/home/sh861201/SimpleScalar/simplesim-3.0/benchmark/mcf/mcf/Mcf
/home/sh861201/SimpleScalar/simplesim-3.0/benchmark/mcf/mcf/inp.in

sim-profile: SimpleScalar/PISA Tool Set version 3.0 of August, 2003.

Copyright (c) 1994-2003 by Todd M. Austin, Ph.D. and SimpleScalar,
LLC.

All Rights Reserved. This version of SimpleScalar is licensed for
academic

non-commercial use.  No portion of this work may be used by any
commercial

entity, or for any commercial purpose, without the prior written
permission

of SimpleScalar, LLC (info@simplescalar.com).


sim: command line: ../sim-profile -dumpconfig config_file.config
/home/sh861201/SimpleScalar/simplesim-3.0/benchmark/mcf/mcf/Mcf
/home/sh861201/SimpleScalar/simplesim-3.0/benchmark/mcf/mcf/inp.in


sim: simulation started @ Fri Feb  5 14:48:23 2016, options follow:


sim-profile: This simulator implements a functional simulator with

profiling support.  Run with the `-h' flag to see profiling options

available.


# -config                     # load configuration from a file

# -dumpconfig                 # dump configuration to a file

# -h                    false # print help message

# -v                    false # verbose operation
```

```
# -d                    false # enable debug message

# -i                    false # start in Dlite debugger

-seed                       1 # random number generator seed (0 for
timer seed)

# -q                    false # initialize and terminate immediately

# -chkpt               <null> # restore EIO trace execution from
<fname>

# -redir:sim           <null> # redirect simulator output to file
(non-interactive only)

# -redir:prog          <null> # redirect simulated program output to
file

-nice                       0 # simulator scheduling priority

-max:inst                   0 # maximum number of inst's to execute

-all                  false # enable all profile options

-iclass               false # enable instruction class profiling

-iprof                false # enable instruction profiling

-brprof               false # enable branch instruction profiling

-amprof               false # enable address mode profiling

-segprof              false # enable load/store address segment
profiling

-tsymprof             false # enable text symbol profiling

-taddrprof            false # enable text address profiling

-dsymprof             false # enable data symbol profiling

-internal             false # include compiler-internal symbols
during symbol profiling

# -pcstat              <null> # profile stat(s) against text addr's
(mult uses ok)


sim: ** starting functional simulation **
```

```
MCF SPEC version 1.6.I
by  Andreas Loebel
Copyright (c) 1998,1999   ZIB Berlin
All Rights Reserved.

nodes                      : 16555
active arcs                : 244246
simplex iterations         : 182415
flow value                 : 8980173901
new implicit arcs          : 300000
active arcs                : 544246
simplex iterations         : 189170
flow value                 : 8910169940
new implicit arcs          : 300000
active arcs                : 844246
simplex iterations         : 216493
flow value                 : 8650168945
new implicit arcs          : 300000
active arcs                : 1144246
simplex iterations         : 261464
flow value                 : 8570161464
new implicit arcs          : 300000
active arcs                : 1444246
simplex iterations         : 290615
flow value                 : 8570159306
new implicit arcs          : 300000
active arcs                : 1744246
simplex iterations         : 318729
```

```
flow value                : 8570157650
new implicit arcs         : 300000
active arcs               : 2044246
simplex iterations        : 340078
flow value                : 8570156531
new implicit arcs         : 300000
active arcs               : 2344246
simplex iterations        : 354548
flow value                : 8570156010
new implicit arcs         : 77333
active arcs               : 2421579
simplex iterations        : 361819
flow value                : 8570155949
new implicit arcs         : 1100
active arcs               : 2422679
simplex iterations        : 361826
flow value                : 8570155949
checksum                  : 258659426
optimal


sim: ** simulation statistics **
sim_num_insn            49073291786 # total number of instructions
executed
sim_num_refs            20768099818 # total number of loads and stores
executed
sim_elapsed_time               3683 # total simulation time in seconds
sim_inst_rate          13324271.4597 # simulation speed (in insts/sec)
ld_text_base             0x00400000 # program text (code) segment base
```

```
ld_text_size                113136 # program text (code) size in
bytes
ld_data_base             0x10000000 # program initialized data segment
base
ld_data_size                 19060 # program init'ed `.data' and
uninit'ed `.bss' size in bytes
ld_stack_base            0x7fffc000 # program stack segment base
(highest address in stack)
ld_stack_size                16384 # program initial stack size
ld_prog_entry            0x00400140 # program entry point (initial PC)
ld_environ_base          0x7fff8000 # program environment base address
address
ld_target_big_endian             0 # target executable endian-ness,
non-zero if big endian
mem.page_count               24435 # total number of pages allocated
mem.page_mem                 97740k # total size of memory pages
allocated
mem.ptab_misses            10414410 # total first level page table
misses
mem.ptab_accesses      237836123993 # total page table accesses
mem.ptab_miss_rate           0.0000 # first level page table miss rate
```

(7) sim-safe

```
sh861201@eustis:~/SimpleScalar/simplesim-3.0/safe_Dir$ ../sim-safe -
dumpconfig config_file.config /home/sh861201/SimpleScalar/simplesim-
3.0/benchmark/mcf/mcf/Mcf /home/sh861201/SimpleScalar/simplesim-
3.0/benchmark/mcf/mcf/inp.in

sim-safe: SimpleScalar/PISA Tool Set version 3.0 of August, 2003.

Copyright (c) 1994-2003 by Todd M. Austin, Ph.D. and SimpleScalar,
LLC.

All Rights Reserved. This version of SimpleScalar is licensed for
academic

non-commercial use.  No portion of this work may be used by any
commercial

entity, or for any commercial purpose, without the prior written
permission

of SimpleScalar, LLC (info@simplescalar.com).


sim: command line: ../sim-safe -dumpconfig config_file.config
/home/sh861201/SimpleScalar/simplesim-3.0/benchmark/mcf/mcf/Mcf
/home/sh861201/SimpleScalar/simplesim-3.0/benchmark/mcf/mcf/inp.in


sim: simulation started @ Fri Feb  5 17:10:52 2016, options follow:


sim-safe: This simulator implements a functional simulator.  This

functional simulator is the simplest, most user-friendly simulator in
the

simplescalar tool set.  Unlike sim-fast, this functional simulator
checks

for all instruction errors, and the implementation is crafted for
clarity

rather than speed.
```

```
# -config                     # load configuration from a file
# -dumpconfig                 # dump configuration to a file
# -h                    false # print help message
# -v                    false # verbose operation
# -d                    false # enable debug message
# -i                    false # start in Dlite debugger
-seed                       1 # random number generator seed (0 for
timer seed)
# -q                    false # initialize and terminate immediately
# -chkpt               <null> # restore EIO trace execution from
<fname>
# -redir:sim           <null> # redirect simulator output to file
(non-interactive only)
# -redir:prog          <null> # redirect simulated program output to
file
-nice                       0 # simulator scheduling priority
-max:inst                   0 # maximum number of inst's to execute

sim: ** starting functional simulation **

MCF SPEC version 1.6.I
by  Andreas Loebel
Copyright (c) 1998,1999   ZIB Berlin
All Rights Reserved.

nodes                     : 16555
active arcs               : 244246
simplex iterations        : 182415
flow value                : 8980173901
```

```
new implicit arcs          : 300000
active arcs                : 544246
simplex iterations         : 189170
flow value                 : 8910169940
new implicit arcs          : 300000
active arcs                : 844246
simplex iterations         : 216493
flow value                 : 8650168945
new implicit arcs          : 300000
active arcs                : 1144246
simplex iterations         : 261464
flow value                 : 8570161464
new implicit arcs          : 300000
active arcs                : 1444246
simplex iterations         : 290615
flow value                 : 8570159306
new implicit arcs          : 300000
active arcs                : 1744246
simplex iterations         : 318729
flow value                 : 8570157650
new implicit arcs          : 300000
active arcs                : 2044246
simplex iterations         : 340078
flow value                 : 8570156531
new implicit arcs          : 300000
active arcs                : 2344246
simplex iterations         : 354548
flow value                 : 8570156010
```

```
new implicit arcs          : 77333
active arcs                : 2421579
simplex iterations         : 361819
flow value                 : 8570155949
new implicit arcs          : 1100
active arcs                : 2422679
simplex iterations         : 361826
flow value                 : 8570155949
checksum                   : 258659426
optimal


sim: ** simulation statistics **
sim_num_insn            49073291786 # total number of instructions
executed
sim_num_refs            20768099818 # total number of loads and stores
executed
sim_elapsed_time               3669 # total simulation time in seconds
sim_inst_rate          13375113.5966 # simulation speed (in insts/sec)
ld_text_base             0x00400000 # program text (code) segment base
ld_text_size                 113136 # program text (code) size in
bytes
ld_data_base             0x10000000 # program initialized data segment
base
ld_data_size                  19060 # program init'ed `.data' and
uninit'ed `.bss' size in bytes
ld_stack_base            0x7fffc000 # program stack segment base
(highest address in stack)
ld_stack_size                 16384 # program initial stack size
ld_prog_entry            0x00400140 # program entry point (initial PC)
```

```
ld_environ_base          0x7fff8000 # program environment base address
address
ld_target_big_endian             0 # target executable endian-ness,
non-zero if big endian
mem.page_count               24435 # total number of pages allocated
mem.page_mem                 97740k # total size of memory pages
allocated
mem.ptab_misses            10414410 # total first level page table
misses
mem.ptab_accesses      237836123981 # total page table accesses
mem.ptab_miss_rate           0.0000 # first level page table miss rate
```

10. Simulation Results of Modified Commands

(1) sim-bpred

```
sh861201@eustis:~/SimpleScalar/simplesim-3.0/bpred_Dir$ ../sim-bpred -
bpred:bimod 1024 -dumpconfig config_file.config
/home/sh861201/SimpleScalar/simplesim-3.0/benchmark/mcf/mcf/Mcf
/home/sh861201/SimpleScalar/simplesim-3.0/benchmark/mcf/mcf/inp.in

sim-bpred: SimpleScalar/PISA Tool Set version 3.0 of August, 2003.

Copyright (c) 1994-2003 by Todd M. Austin, Ph.D. and SimpleScalar,
LLC.

All Rights Reserved. This version of SimpleScalar is licensed for
academic

non-commercial use.  No portion of this work may be used by any
commercial

entity, or for any commercial purpose, without the prior written
permission

of SimpleScalar, LLC (info@simplescalar.com).


sim: command line: ../sim-bpred -bpred:bimod 1024 -dumpconfig
config_file.config /home/sh861201/SimpleScalar/simplesim-
3.0/benchmark/mcf/mcf/Mcf /home/sh861201/SimpleScalar/simplesim-
3.0/benchmark/mcf/mcf/inp.in


sim: simulation started @ Sun Feb 21 14:14:05 2016, options follow:


sim-bpred: This simulator implements a branch predictor analyzer.


# -config                     # load configuration from a file

# -dumpconfig                 # dump configuration to a file

# -h                    false # print help message
```

```
# -v                    false # verbose operation

# -d                    false # enable debug message

# -i                    false # start in Dlite debugger

-seed                       1 # random number generator seed (0 for
timer seed)

# -q                    false # initialize and terminate immediately

# -chkpt               <null> # restore EIO trace execution from
<fname>

# -redir:sim           <null> # redirect simulator output to file
(non-interactive only)

# -redir:prog          <null> # redirect simulated program output to
file

-nice                       0 # simulator scheduling priority

-max:inst                   0 # maximum number of inst's to execute

-bpred                  bimod # branch predictor type
{nottaken|taken|bimod|2lev|comb}

-bpred:bimod     1024 # bimodal predictor config (<table size>)

-bpred:2lev      1 1024 8 0 # 2-level predictor config (<l1size>
<l2size> <hist_size> <xor>)

-bpred:comb      1024 # combining predictor config (<meta_table_size>)

-bpred:ras                  8 # return address stack size (0 for no
return stack)

-bpred:btb       512 4 # BTB config (<num_sets> <associativity>)


  Branch predictor configuration examples for 2-level predictor:

    Configurations:   N, M, W, X

      N   # entries in first level (# of shift register(s))

      W   width of shift register(s)

      M   # entries in 2nd level (# of counters, or other FSM)

      X   (yes-1/no-0) xor history and address for 2nd level index
```

```
    Sample predictors:
      GAg     : 1, W, 2^W, 0
      GAp     : 1, W, M (M > 2^W), 0
      PAg     : N, W, 2^W, 0
      PAp     : N, W, M (M == 2^(N+W)), 0
      gshare  : 1, W, 2^W, 1
  Predictor `comb' combines a bimodal and a 2-level predictor.

sim: ** starting functional simulation w/ predictors **

MCF SPEC version 1.6.I
by  Andreas Loebel
Copyright (c) 1998,1999   ZIB Berlin
All Rights Reserved.

nodes                      : 16555
active arcs                : 244246
simplex iterations         : 182415
flow value                 : 8980173901
new implicit arcs          : 300000
active arcs                : 544246
simplex iterations         : 189170
flow value                 : 8910169940
new implicit arcs          : 300000
active arcs                : 844246
simplex iterations         : 216493
flow value                 : 8650168945
new implicit arcs          : 300000
```

```
active arcs               : 1144246
simplex iterations        : 261464
flow value                : 8570161464
new implicit arcs         : 300000
active arcs               : 1444246
simplex iterations        : 290615
flow value                : 8570159306
new implicit arcs         : 300000
active arcs               : 1744246
simplex iterations        : 318729
flow value                : 8570157650
new implicit arcs         : 300000
active arcs               : 2044246
simplex iterations        : 340078
flow value                : 8570156531
new implicit arcs         : 300000
active arcs               : 2344246
simplex iterations        : 354548
flow value                : 8570156010
new implicit arcs         : 77333
active arcs               : 2421579
simplex iterations        : 361819
flow value                : 8570155949
new implicit arcs         : 1100
active arcs               : 2422679
simplex iterations        : 361826
flow value                : 8570155949
checksum                  : 258659426
```

```
optimal


sim: ** simulation statistics **
sim_num_insn            49073291786 # total number of instructions
executed
sim_num_refs            20768099818 # total number of loads and stores
executed
sim_elapsed_time               5442 # total simulation time in seconds
sim_inst_rate          9017510.4348 # simulation speed (in insts/sec)
sim_num_branches        12052458319 # total number of branches
executed
sim_IPB                      4.0716 # instruction per branch
bpred_bimod.lookups     12052458319 # total number of bpred lookups
bpred_bimod.updates     12052458319 # total number of updates
bpred_bimod.addr_hits   10890128516 # total number of address-
predicted hits
bpred_bimod.dir_hits    10890979402 # total number of direction-
predicted hits (includes addr-hits)
bpred_bimod.misses       1161478917 # total number of misses
bpred_bimod.jr_hits        40089720 # total number of address-
predicted hits for JR's
bpred_bimod.jr_seen        40956662 # total number of JR's seen
bpred_bimod.jr_non_ras_hits.PP      1250351 # total number of address-
predicted hits for non-RAS JR's
bpred_bimod.jr_non_ras_seen.PP      1266947 # total number of non-RAS
JR's seen
bpred_bimod.bpred_addr_rate    0.9036 # branch address-prediction rate
(i.e., addr-hits/updates)
bpred_bimod.bpred_dir_rate    0.9036 # branch direction-prediction
rate (i.e., all-hits/updates)
bpred_bimod.bpred_jr_rate    0.9788 # JR address-prediction rate
(i.e., JR addr-hits/JRs seen)
```

```
bpred_bimod.bpred_jr_non_ras_rate.PP    0.9869 # non-RAS JR addr-pred
rate (ie, non-RAS JR hits/JRs seen)

bpred_bimod.retstack_pushes     39689717 # total number of address
pushed onto ret-addr stack

bpred_bimod.retstack_pops     39689715 # total number of address
popped off of ret-addr stack

bpred_bimod.used_ras.PP     39689715 # total number of RAS predictions
used

bpred_bimod.ras_hits.PP     38839369 # total number of RAS hits

bpred_bimod.ras_rate.PP    0.9786 # RAS prediction rate (i.e., RAS
hits/used RAS)
```

(2) sim-cache

```
sh861201@eustis:~/SimpleScalar/simplesim-3.0/cache_Dir$ ../sim-cache -
cache:dl1 dl1:256:32:4:l -cache:il1 il1:256:32:4:l -dumpconfig
config_file.config /home/sh861201/SimpleScalar/simplesim-
3.0/benchmark/mcf/mcf/Mcf /home/sh861201/SimpleScalar/simplesim-
3.0/benchmark/mcf/mcf/inp.in

sim-cache: SimpleScalar/PISA Tool Set version 3.0 of August, 2003.

Copyright (c) 1994-2003 by Todd M. Austin, Ph.D. and SimpleScalar,
LLC.

All Rights Reserved. This version of SimpleScalar is licensed for
academic

non-commercial use.  No portion of this work may be used by any
commercial

entity, or for any commercial purpose, without the prior written
permission

of SimpleScalar, LLC (info@simplescalar.com).


sim: command line: ../sim-cache -cache:dl1 dl1:256:32:4:l -cache:il1
il1:256:32:4:l -dumpconfig config_file.config
/home/sh861201/SimpleScalar/simplesim-3.0/benchmark/mcf/mcf/Mcf
/home/sh861201/SimpleScalar/simplesim-3.0/benchmark/mcf/mcf/inp.in


sim: simulation started @ Sun Feb 21 16:04:09 2016, options follow:


sim-cache: This simulator implements a functional cache simulator.
Cache

statistics are generated for a user-selected cache and TLB
configuration,

which may include up to two levels of instruction and data cache (with
any

levels unified), and one level of instruction and data TLBs.  No
timing
```

information is generated.

```
# -config                     # load configuration from a file
# -dumpconfig                 # dump configuration to a file
# -h                    false # print help message
# -v                    false # verbose operation
# -d                    false # enable debug message
# -i                    false # start in Dlite debugger
-seed                       1 # random number generator seed (0 for
timer seed)
# -q                    false # initialize and terminate immediately
# -chkpt                <null> # restore EIO trace execution from
<fname>
# -redir:sim            <null> # redirect simulator output to file
(non-interactive only)
# -redir:prog           <null> # redirect simulated program output to
file
-nice                       0 # simulator scheduling priority
-max:inst                   0 # maximum number of inst's to execute
-cache:dl1       dl1:256:32:4:l # l1 data cache config, i.e.,
{<config>|none}
-cache:dl2       ul2:1024:64:4:l # l2 data cache config, i.e.,
{<config>|none}
-cache:il1       il1:256:32:4:l # l1 inst cache config, i.e.,
{<config>|dl1|dl2|none}
-cache:il2                dl2 # l2 instruction cache config, i.e.,
{<config>|dl2|none}
-tlb:itlb        itlb:16:4096:4:l # instruction TLB config, i.e.,
{<config>|none}
-tlb:dtlb        dtlb:32:4096:4:l # data TLB config, i.e.,
{<config>|none}
```

```
-flush                  false # flush caches on system calls
-cache:icompress        false # convert 64-bit inst addresses to 32-
bit inst equivalents
# -pcstat              <null> # profile stat(s) against text addr's
(mult uses ok)

  The cache config parameter <config> has the following format:

    <name>:<nsets>:<bsize>:<assoc>:<repl>

    <name>   - name of the cache being defined
    <nsets>  - number of sets in the cache
    <bsize>  - block size of the cache
    <assoc>  - associativity of the cache
    <repl>   - block replacement strategy, 'l'-LRU, 'f'-FIFO, 'r'-
random

    Examples:   -cache:dl1 dl1:4096:32:1:l
                -dtlb dtlb:128:4096:32:r

  Cache levels can be unified by pointing a level of the instruction
cache
  hierarchy at the data cache hiearchy using the "dl1" and "dl2" cache
  configuration arguments.  Most sensible combinations are supported,
e.g.,

    A unified l2 cache (il2 is pointed at dl2):
      -cache:il1 il1:128:64:1:l -cache:il2 dl2
      -cache:dl1 dl1:256:32:1:l -cache:dl2 ul2:1024:64:2:l
```

```
    Or, a fully unified cache hierarchy (il1 pointed at dl1):
      -cache:il1 dl1
      -cache:dl1 ul1:256:32:1:l -cache:dl2 ul2:1024:64:2:l

sim: ** starting functional simulation w/ caches **

MCF SPEC version 1.6.I
by  Andreas Loebel
Copyright (c) 1998,1999   ZIB Berlin
All Rights Reserved.

nodes                     : 16555
active arcs               : 244246
simplex iterations        : 182415
flow value                : 8980173901
new implicit arcs         : 300000
active arcs               : 544246
simplex iterations        : 189170
flow value                : 8910169940
new implicit arcs         : 300000
active arcs               : 844246
simplex iterations        : 216493
flow value                : 8650168945
new implicit arcs         : 300000
active arcs               : 1144246
simplex iterations        : 261464
flow value                : 8570161464
new implicit arcs         : 300000
```

```
active arcs                : 1444246
simplex iterations         : 290615
flow value                 : 8570159306
new implicit arcs          : 300000
active arcs                : 1744246
simplex iterations         : 318729
flow value                 : 8570157650
new implicit arcs          : 300000
active arcs                : 2044246
simplex iterations         : 340078
flow value                 : 8570156531
new implicit arcs          : 300000
active arcs                : 2344246
simplex iterations         : 354548
flow value                 : 8570156010
new implicit arcs          : 77333
active arcs                : 2421579
simplex iterations         : 361819
flow value                 : 8570155949
new implicit arcs          : 1100
active arcs                : 2422679
simplex iterations         : 361826
flow value                 : 8570155949
checksum                   : 258659426
optimal

sim: ** simulation statistics **
```

```
sim_num_insn              49073291786 # total number of instructions
executed
sim_num_refs              20768099818 # total number of loads and stores
executed
sim_elapsed_time              12688 # total simulation time in seconds
sim_inst_rate           3867693.2366 # simulation speed (in insts/sec)
il1.accesses              49073291786 # total number of accesses
il1.hits                  49073285919 # total number of hits
il1.misses                     5867 # total number of misses
il1.replacements               4845 # total number of replacements
il1.writebacks                    0 # total number of writebacks
il1.invalidations                 0 # total number of invalidations
il1.miss_rate                0.0000 # miss rate (i.e., misses/ref)
il1.repl_rate                0.0000 # replacement rate (i.e.,
repls/ref)
il1.wb_rate                  0.0000 # writeback rate (i.e., wrbks/ref)
il1.inv_rate                 0.0000 # invalidation rate (i.e.,
invs/ref)
dl1.accesses              20771127189 # total number of accesses
dl1.hits                  13832503379 # total number of hits
dl1.misses                 6938623810 # total number of misses
dl1.replacements           6938622786 # total number of replacements
dl1.writebacks             1306397952 # total number of writebacks
dl1.invalidations                 0 # total number of invalidations
dl1.miss_rate                0.3341 # miss rate (i.e., misses/ref)
dl1.repl_rate                0.3341 # replacement rate (i.e.,
repls/ref)
dl1.wb_rate                  0.0629 # writeback rate (i.e., wrbks/ref)
dl1.inv_rate                 0.0000 # invalidation rate (i.e.,
invs/ref)
```

```
ul2.accesses              8245027629 # total number of accesses
ul2.hits                  3022866073 # total number of hits
ul2.misses                5222161556 # total number of misses
ul2.replacements          5222157460 # total number of replacements
ul2.writebacks            1207162269 # total number of writebacks
ul2.invalidations                  0 # total number of invalidations
ul2.miss_rate                 0.6334 # miss rate (i.e., misses/ref)
ul2.repl_rate                 0.6334 # replacement rate (i.e.,
repls/ref)
ul2.wb_rate                   0.1464 # writeback rate (i.e., wrbks/ref)
ul2.inv_rate                  0.0000 # invalidation rate (i.e.,
invs/ref)
itlb.accesses            49073291786 # total number of accesses
itlb.hits                49073291759 # total number of hits
itlb.misses                       27 # total number of misses
itlb.replacements                  0 # total number of replacements
itlb.writebacks                    0 # total number of writebacks
itlb.invalidations                 0 # total number of invalidations
itlb.miss_rate                0.0000 # miss rate (i.e., misses/ref)
itlb.repl_rate                0.0000 # replacement rate (i.e.,
repls/ref)
itlb.wb_rate                  0.0000 # writeback rate (i.e., wrbks/ref)
itlb.inv_rate                 0.0000 # invalidation rate (i.e.,
invs/ref)
dtlb.accesses            20771127189 # total number of accesses
dtlb.hits                19090713234 # total number of hits
dtlb.misses               1680413955 # total number of misses
dtlb.replacements         1680413827 # total number of replacements
dtlb.writebacks            365182120 # total number of writebacks
```

```
dtlb.invalidations               0 # total number of invalidations
dtlb.miss_rate              0.0809 # miss rate (i.e., misses/ref)
dtlb.repl_rate              0.0809 # replacement rate (i.e.,
repls/ref)
dtlb.wb_rate                0.0176 # writeback rate (i.e., wrbks/ref)
dtlb.inv_rate               0.0000 # invalidation rate (i.e.,
invs/ref)
ld_text_base             0x00400000 # program text (code) segment base
ld_text_size                113136 # program text (code) size in
bytes
ld_data_base             0x10000000 # program initialized data segment
base
ld_data_size                 19060 # program init'ed `.data' and
uninit'ed `.bss' size in bytes
ld_stack_base            0x7fffc000 # program stack segment base
(highest address in stack)
ld_stack_size                16384 # program initial stack size
ld_prog_entry            0x00400140 # program entry point (initial PC)
ld_environ_base          0x7fff8000 # program environment base address
address
ld_target_big_endian             0 # target executable endian-ness,
non-zero if big endian
mem.page_count               24435 # total number of pages allocated
mem.page_mem                 97740k # total size of memory pages
allocated
mem.ptab_misses           10414410 # total first level page table
misses
mem.ptab_accesses      237836123987 # total page table accesses
mem.ptab_miss_rate          0.0000 # first level page table miss rate
```

(3) sim-eio

```
sh861201@eustis:~/SimpleScalar/simplesim-3.0/eio_Dir$ ../sim-eio -
fastfwd 1000 -dumpconfig config_file.config
/home/sh861201/SimpleScalar/simplesim-3.0/benchmark/mcf/mcf/Mcf
/home/sh861201/SimpleScalar/simplesim-3.0/benchmark/mcf/mcf/inp.in

sim-eio: SimpleScalar/PISA Tool Set version 3.0 of August, 2003.

Copyright (c) 1994-2003 by Todd M. Austin, Ph.D. and SimpleScalar,
LLC.

All Rights Reserved. This version of SimpleScalar is licensed for
academic

non-commercial use.  No portion of this work may be used by any
commercial

entity, or for any commercial purpose, without the prior written
permission

of SimpleScalar, LLC (info@simplescalar.com).


sim: command line: ../sim-eio -fastfwd 1000 -dumpconfig
config_file.config /home/sh861201/SimpleScalar/simplesim-
3.0/benchmark/mcf/mcf/Mcf /home/sh861201/SimpleScalar/simplesim-
3.0/benchmark/mcf/mcf/inp.in


sim: simulation started @ Mon Feb 22 11:46:22 2016, options follow:


sim-eio: This simulator implements simulator support for generating

external event traces (EIO traces) and checkpoint files.  External

event traces capture one execution of a program, and allow it to be

packaged into a single file for later re-execution.  EIO trace
executions

are 100% reproducible between subsequent executions (on the same
platform.

This simulator also provides functionality to generate checkpoints at
```

arbitrary points within an external event trace (EIO) execution. The checkpoint file (along with the EIO trace) can be used to start any SimpleScalar simulator in the middle of a program execution.

```
# -config                     # load configuration from a file
# -dumpconfig                 # dump configuration to a file
# -h                    false # print help message
# -v                    false # verbose operation
# -d                    false # enable debug message
# -i                    false # start in Dlite debugger
-seed                       1 # random number generator seed (0 for
timer seed)
# -q                    false # initialize and terminate immediately
# -chkpt               <null> # restore EIO trace execution from
<fname>
# -redir:sim           <null> # redirect simulator output to file
(non-interactive only)
# -redir:prog          <null> # redirect simulated program output to
file
-nice                       0 # simulator scheduling priority
-max:inst                   0 # maximum number of inst's to execute
-fastfwd                 1000 # number of insts skipped before tracing
starts
# -trace               <null> # EIO trace file output file name
# -perdump             <null> # periodic checkpoint every n
instructions: <base fname> <interval>
# -dump                <null> # specify checkpoint file and trigger:
<fname> <range>
```

Checkpoint range triggers are formatted as follows:

```
    {{@|#}<start>}:{{@|#|+}<end>}

  Both ends of the range are optional, if neither are specified, the
range

  triggers immediately.  Ranges that start with a `@' designate an
address

  range to trigger on, those that start with an `#' designate a cycle
count

  trigger.  All other ranges represent an instruction count range.
The

  second argument, if specified with a `+', indicates a value relative

  to the first argument, e.g., 1000:+100 == 1000:1100.

    Examples:   -ptrace FOO.trc #0:#1000

                -ptrace BAR.trc @2000:

                -ptrace BLAH.trc :1500

                -ptrace UXXE.trc :

sim: ** fast forwarding 1000 insts **

sim: ** starting functional simulation **

MCF SPEC version 1.6.I

by  Andreas Loebel

Copyright (c) 1998,1999   ZIB Berlin

All Rights Reserved.

nodes                       : 16555

active arcs                 : 244246
```

```
simplex iterations         : 182415
flow value                 : 8980173901
new implicit arcs          : 300000
active arcs                : 544246
simplex iterations         : 189170
flow value                 : 8910169940
new implicit arcs          : 300000
active arcs                : 844246
simplex iterations         : 216493
flow value                 : 8650168945
new implicit arcs          : 300000
active arcs                : 1144246
simplex iterations         : 261464
flow value                 : 8570161464
new implicit arcs          : 300000
active arcs                : 1444246
simplex iterations         : 290615
flow value                 : 8570159306
new implicit arcs          : 300000
active arcs                : 1744246
simplex iterations         : 318729
flow value                 : 8570157650
new implicit arcs          : 300000
active arcs                : 2044246
simplex iterations         : 340078
flow value                 : 8570156531
new implicit arcs          : 300000
active arcs                : 2344246
```

```
simplex iterations        : 354548
flow value                : 8570156010
new implicit arcs         : 77333
active arcs               : 2421579
simplex iterations        : 361819
flow value                : 8570155949
new implicit arcs         : 1100
active arcs               : 2422679
simplex iterations        : 361826
flow value                : 8570155949
checksum                  : 258659426
optimal


sim: ** simulation statistics **
sim_num_insn            49073290786 # total number of instructions
executed
sim_num_refs            20768099582 # total number of loads and stores
executed
sim_elapsed_time               3929 # total simulation time in seconds
sim_inst_rate          12490020.5615 # simulation speed (in insts/sec)
ld_text_base             0x00400000 # program text (code) segment base
ld_text_size                113136 # program text (code) size in
bytes
ld_data_base             0x10000000 # program initialized data segment
base
ld_data_size                 19060 # program init'ed `.data' and
uninit'ed `.bss' size in bytes
ld_stack_base            0x7fffc000 # program stack segment base
(highest address in stack)
ld_stack_size                16384 # program initial stack size
```

```
ld_prog_entry            0x00400140 # program entry point (initial PC)
ld_environ_base          0x7fff8000 # program environment base address
address
ld_target_big_endian              0 # target executable endian-ness,
non-zero if big endian
mem.page_count                24435 # total number of pages allocated
mem.page_mem                  97740k # total size of memory pages
allocated
mem.ptab_misses            10414410 # total first level page table
misses
mem.ptab_accesses      237836123979 # total page table accesses
mem.ptab_miss_rate           0.0000 # first level page table miss rate
```

(4) sim-fast

```
sh861201@eustis:~/SimpleScalar/simplesim-3.0/fast_Dir$ ../sim-fast -v
true -dumpconfig config_file.config
/home/sh861201/SimpleScalar/simplesim-3.0/benchmark/mcf/mcf/Mcf
/home/sh861201/SimpleScalar/simplesim-3.0/benchmark/mcf/mcf/inp.in

sim-fast: SimpleScalar/PISA Tool Set version 3.0 of August, 2003.

Copyright (c) 1994-2003 by Todd M. Austin, Ph.D. and SimpleScalar,
LLC.

All Rights Reserved. This version of SimpleScalar is licensed for
academic

non-commercial use.  No portion of this work may be used by any
commercial

entity, or for any commercial purpose, without the prior written
permission

of SimpleScalar, LLC (info@simplescalar.com).


sim: command line: ../sim-fast -v true -dumpconfig config_file.config
/home/sh861201/SimpleScalar/simplesim-3.0/benchmark/mcf/mcf/Mcf
/home/sh861201/SimpleScalar/simplesim-3.0/benchmark/mcf/mcf/inp.in


sim: simulation started @ Mon Feb 22 13:30:59 2016, options follow:


sim-fast: This simulator implements a very fast functional simulator.
This

functional simulator implementation is much more difficult to digest
than

the simpler, cleaner sim-safe functional simulator.  By default, this

simulator performs no instruction error checking, as a result, any

instruction errors will manifest as simulator execution errors,
possibly

causing sim-fast to execute incorrectly or dump core.  Such is the
```

```
price we pay for speed!!!!

# -config                     # load configuration from a file
# -dumpconfig                 # dump configuration to a file
# -h                    false # print help message
# -v                     true # verbose operation
# -d                    false # enable debug message
# -i                    false # start in Dlite debugger
-seed                       1 # random number generator seed (0 for
timer seed)
# -q                    false # initialize and terminate immediately
# -chkpt               <null> # restore EIO trace execution from
<fname>
# -redir:sim           <null> # redirect simulator output to file
(non-interactive only)
# -redir:prog          <null> # redirect simulated program output to
file
-nice                       0 # simulator scheduling priority

sim: ** starting *fast* functional simulation **

MCF SPEC version 1.6.I
by  Andreas Loebel
Copyright (c) 1998,1999   ZIB Berlin
All Rights Reserved.

nodes                     : 16555
active arcs               : 244246
simplex iterations        : 182415
```

```
flow value                : 8980173901
new implicit arcs         : 300000
active arcs               : 544246
simplex iterations        : 189170
flow value                : 8910169940
new implicit arcs         : 300000
active arcs               : 844246
simplex iterations        : 216493
flow value                : 8650168945
new implicit arcs         : 300000
active arcs               : 1144246
simplex iterations        : 261464
flow value                : 8570161464
new implicit arcs         : 300000
active arcs               : 1444246
simplex iterations        : 290615
flow value                : 8570159306
new implicit arcs         : 300000
active arcs               : 1744246
simplex iterations        : 318729
flow value                : 8570157650
new implicit arcs         : 300000
active arcs               : 2044246
simplex iterations        : 340078
flow value                : 8570156531
new implicit arcs         : 300000
active arcs               : 2344246
simplex iterations        : 354548
```

```
flow value                    : 8570156010
new implicit arcs             : 77333
active arcs                   : 2421579
simplex iterations            : 361819
flow value                    : 8570155949
new implicit arcs             : 1100
active arcs                   : 2422679
simplex iterations            : 361826
flow value                    : 8570155949
checksum                      : 258659426
optimal


sim: ** simulation statistics **
sim_num_insn            49073291786 # total number of instructions
executed
sim_elapsed_time               3373 # total simulation time in seconds
sim_inst_rate          14548856.1476 # simulation speed (in insts/sec)
ld_text_base             0x00400000 # program text (code) segment base
ld_text_size                 113136 # program text (code) size in
bytes
ld_data_base             0x10000000 # program initialized data segment
base
ld_data_size                  19060 # program init'ed `.data' and
uninit'ed `.bss' size in bytes
ld_stack_base            0x7fffc000 # program stack segment base
(highest address in stack)
ld_stack_size                 16384 # program initial stack size
ld_prog_entry            0x00400140 # program entry point (initial PC)
ld_environ_base          0x7fff8000 # program environment base address
address
```

```
ld_target_big_endian              0 # target executable endian-ness,
non-zero if big endian

mem.page_count                24435 # total number of pages allocated

mem.page_mem                  97740k # total size of memory pages
allocated

mem.ptab_misses            10414410 # total first level page table
misses

mem.ptab_accesses      237836123983 # total page table accesses

mem.ptab_miss_rate           0.0000 # first level page table miss rate
```

(5) sim-outorder

```
sh861201@eustis:~/SimpleScalar/simplesim-3.0/outorder_Dir$ ../sim-
outorder -issue:width 8 -dumpconfig config_file.config
/home/sh861201/SimpleScalar/simplesim-3.0/benchmark/mcf/mcf/Mcf
/home/sh861201/SimpleScalar/simplesim-3.0/benchmark/mcf/mcf/inp.in

sim-outorder: SimpleScalar/PISA Tool Set version 3.0 of August, 2003.

Copyright (c) 1994-2003 by Todd M. Austin, Ph.D. and SimpleScalar,
LLC.

All Rights Reserved. This version of SimpleScalar is licensed for
academic

non-commercial use.  No portion of this work may be used by any
commercial

entity, or for any commercial purpose, without the prior written
permission

of SimpleScalar, LLC (info@simplescalar.com).


sim: command line: ../sim-outorder -issue:width 8 -dumpconfig
config_file.config /home/sh861201/SimpleScalar/simplesim-
3.0/benchmark/mcf/mcf/Mcf /home/sh861201/SimpleScalar/simplesim-
3.0/benchmark/mcf/mcf/inp.in


sim: simulation started @ Tue Feb 23 12:35:56 2016, options follow:


sim-outorder: This simulator implements a very detailed out-of-order
issue

superscalar processor with a two-level memory system and speculative

execution support.  This simulator is a performance simulator,
tracking the

latency of all pipeline operations.


# -config                     # load configuration from a file
```

```
# -dumpconfig                # dump configuration to a file
# -h                    false # print help message
# -v                    false # verbose operation
# -d                    false # enable debug message
# -i                    false # start in Dlite debugger
-seed                       1 # random number generator seed (0 for
timer seed)
# -q                    false # initialize and terminate immediately
# -chkpt               <null> # restore EIO trace execution from
<fname>
# -redir:sim           <null> # redirect simulator output to file
(non-interactive only)
# -redir:prog          <null> # redirect simulated program output to
file
-nice                       0 # simulator scheduling priority
-max:inst                   0 # maximum number of inst's to execute
-fastfwd                    0 # number of insts skipped before timing
starts
# -ptrace              <null> # generate pipetrace, i.e.,
<fname|stdout|stderr> <range>
-fetch:ifqsize              4 # instruction fetch queue size (in
insts)
-fetch:mplat                3 # extra branch mis-prediction latency
-fetch:speed                1 # speed of front-end of machine relative
to execution core
-bpred                  bimod # branch predictor type
{nottaken|taken|perfect|bimod|2lev|comb}
-bpred:bimod     2048 # bimodal predictor config (<table size>)
-bpred:2lev      1 1024 8 0 # 2-level predictor config (<l1size>
<l2size> <hist_size> <xor>)
-bpred:comb      1024 # combining predictor config (<meta_table_size>)
```

```
-bpred:ras                  8 # return address stack size (0 for no
return stack)

-bpred:btb       512 4 # BTB config (<num_sets> <associativity>)

# -bpred:spec_update       <null> # speculative predictors update in
{ID|WB} (default non-spec)

-decode:width               4 # instruction decode B/W (insts/cycle)

-issue:width                8 # instruction issue B/W (insts/cycle)

-issue:inorder          false # run pipeline with in-order issue

-issue:wrongpath         true # issue instructions down wrong
execution paths

-commit:width               4 # instruction commit B/W (insts/cycle)

-ruu:size                  16 # register update unit (RUU) size

-lsq:size                   8 # load/store queue (LSQ) size

-cache:dl1       dl1:128:32:4:l # l1 data cache config, i.e.,
{<config>|none}

-cache:dl1lat               1 # l1 data cache hit latency (in cycles)

-cache:dl2       ul2:1024:64:4:l # l2 data cache config, i.e.,
{<config>|none}

-cache:dl2lat               6 # l2 data cache hit latency (in cycles)

-cache:il1       il1:512:32:1:l # l1 inst cache config, i.e.,
{<config>|dl1|dl2|none}

-cache:il1lat               1 # l1 instruction cache hit latency (in
cycles)

-cache:il2                dl2 # l2 instruction cache config, i.e.,
{<config>|dl2|none}

-cache:il2lat               6 # l2 instruction cache hit latency (in
cycles)

-cache:flush            false # flush caches on system calls

-cache:icompress        false # convert 64-bit inst addresses to 32-
bit inst equivalents

-mem:lat         18 2 # memory access latency (<first_chunk>
<inter_chunk>)
```

```
-mem:width                  8 # memory access bus width (in bytes)

-tlb:itlb        itlb:16:4096:4:l # instruction TLB config, i.e.,
{<config>|none}

-tlb:dtlb        dtlb:32:4096:4:l # data TLB config, i.e.,
{<config>|none}

-tlb:lat                   30 # inst/data TLB miss latency (in cycles)

-res:ialu                   4 # total number of integer ALU's
available

-res:imult                  1 # total number of integer
multiplier/dividers available

-res:memport                2 # total number of memory system ports
available (to CPU)

-res:fpalu                  4 # total number of floating point ALU's
available

-res:fpmult                 1 # total number of floating point
multiplier/dividers available

# -pcstat              <null> # profile stat(s) against text addr's
(mult uses ok)

-bugcompat               false # operate in backward-compatible bugs
mode (for testing only)


  Pipetrace range arguments are formatted as follows:


    {{@|#}<start>}:{{@|#|+}<end>}


  Both ends of the range are optional, if neither are specified, the
entire

  execution is traced.  Ranges that start with a `@' designate an
address

  range to be traced, those that start with an `#' designate a cycle
count
```

```
  range.  All other range values represent an instruction count range.
The

  second argument, if specified with a `+', indicates a value relative

  to the first argument, e.g., 1000:+100 == 1000:1100.  Program
symbols may

  be used in all contexts.


    Examples:   -ptrace FOO.trc #0:#1000

                -ptrace BAR.trc @2000:

                -ptrace BLAH.trc :1500

                -ptrace UXXE.trc :

                -ptrace FOOBAR.trc @main:+278


  Branch predictor configuration examples for 2-level predictor:

    Configurations:   N, M, W, X

      N   # entries in first level (# of shift register(s))

      W   width of shift register(s)

      M   # entries in 2nd level (# of counters, or other FSM)

      X   (yes-1/no-0) xor history and address for 2nd level index

    Sample predictors:

      GAg     : 1, W, 2^W, 0

      GAp     : 1, W, M (M > 2^W), 0

      PAg     : N, W, 2^W, 0

      PAp     : N, W, M (M == 2^(N+W)), 0

      gshare  : 1, W, 2^W, 1

  Predictor `comb' combines a bimodal and a 2-level predictor.


  The cache config parameter <config> has the following format:
```

```
    <name>:<nsets>:<bsize>:<assoc>:<repl>

    <name>   - name of the cache being defined
    <nsets>  - number of sets in the cache
    <bsize>  - block size of the cache
    <assoc>  - associativity of the cache
    <repl>   - block replacement strategy, 'l'-LRU, 'f'-FIFO, 'r'-
random

    Examples:   -cache:dl1 dl1:4096:32:1:l
                -dtlb dtlb:128:4096:32:r

  Cache levels can be unified by pointing a level of the instruction
cache
  hierarchy at the data cache hiearchy using the "dl1" and "dl2" cache
  configuration arguments.  Most sensible combinations are supported,
e.g.,

    A unified l2 cache (il2 is pointed at dl2):
      -cache:il1 il1:128:64:1:l -cache:il2 dl2
      -cache:dl1 dl1:256:32:1:l -cache:dl2 ul2:1024:64:2:l

    Or, a fully unified cache hierarchy (il1 pointed at dl1):
      -cache:il1 dl1
      -cache:dl1 ul1:256:32:1:l -cache:dl2 ul2:1024:64:2:l
sim: ** starting performance simulation **

MCF SPEC version 1.6.I
by  Andreas Loebel
```

Copyright (c) 1998,1999   ZIB Berlin
All Rights Reserved.

nodes   : 16555
active arcs   : 244246
simplex iterations   : 182415
flow value   : 8980173901
new implicit arcs   : 300000
active arcs   : 544246
simplex iterations   : 189170
flow value   : 8910169940
new implicit arcs   : 300000
active arcs   : 844246
simplex iterations   : 216493
flow value   : 8650168945
new implicit arcs   : 300000
active arcs   : 1144246
simplex iterations   : 261464
flow value   : 8570161464
new implicit arcs   : 300000
active arcs   : 1444246
simplex iterations   : 290615
flow value   : 8570159306
new implicit arcs   : 300000
active arcs   : 1744246
simplex iterations   : 318729
flow value   : 8570157650
new implicit arcs   : 300000

```
active arcs               : 2044246
simplex iterations        : 340078
flow value                : 8570156531
new implicit arcs         : 300000
active arcs               : 2344246
simplex iterations        : 354548
flow value                : 8570156010
new implicit arcs         : 77333
active arcs               : 2421579
simplex iterations        : 361819
flow value                : 8570155949
new implicit arcs         : 1100
active arcs               : 2422679
simplex iterations        : 361826
flow value                : 8570155949
checksum                  : 258659426
optimal


sim: ** simulation statistics **
sim_num_insn            49073291786 # total number of instructions
committed
sim_num_refs            20768099818 # total number of loads and stores
committed
sim_num_loads           18041672686 # total number of loads committed
sim_num_stores         2726427132.0000 # total number of stores
committed
sim_num_branches        12052458319 # total number of branches
committed
sim_elapsed_time             318597 # total simulation time in seconds
```

```
sim_inst_rate           154029.3593 # simulation speed (in insts/sec)
sim_total_insn          58743007456 # total number of instructions
executed
sim_total_refs          25978903722 # total number of loads and stores
executed
sim_total_loads         22727689445 # total number of loads executed
sim_total_stores      3251214277.0000 # total number of stores
executed
sim_total_branches      14101744419 # total number of branches
executed
sim_cycle             142946543617 # total simulation time in cycles
sim_IPC                     0.3433 # instructions per cycle
sim_CPI                     2.9129 # cycles per instruction
sim_exec_BW                 0.4109 # total instructions (mis-spec +
committed) per cycle
sim_IPB                     4.0716 # instruction per branch
IFQ_count             542371002468 # cumulative IFQ occupancy
IFQ_fcount            132264310386 # cumulative IFQ full count
ifq_occupancy               3.7942 # avg IFQ occupancy (insn's)
ifq_rate                    0.4109 # avg IFQ dispatch rate
(insn/cycle)
ifq_latency                 9.2329 # avg IFQ occupant latency
(cycle's)
ifq_full                    0.9253 # fraction of time (cycle's) IFQ
was full
RUU_count             2103464815921 # cumulative RUU occupancy
RUU_fcount            100233353457 # cumulative RUU full count
ruu_occupancy              14.7150 # avg RUU occupancy (insn's)
ruu_rate                    0.4109 # avg RUU dispatch rate
(insn/cycle)
```

```
ruu_latency                35.8079 # avg RUU occupant latency
(cycle's)
ruu_full                    0.7012 # fraction of time (cycle's) RUU
was full
LSQ_count              969632851523 # cumulative LSQ occupancy
LSQ_fcount              66608943771 # cumulative LSQ full count
lsq_occupancy               6.7832 # avg LSQ occupancy (insn's)
lsq_rate                    0.4109 # avg LSQ dispatch rate
(insn/cycle)
lsq_latency                16.5064 # avg LSQ occupant latency
(cycle's)
lsq_full                    0.4660 # fraction of time (cycle's) LSQ
was full
sim_slip               9088993776478224952 # total number of slip
cycles
avg_sim_slip           185212636.9699 # the average slip between issue
and retirement
bpred_bimod.lookups     15286663565 # total number of bpred lookups
bpred_bimod.updates     12052458319 # total number of updates
bpred_bimod.addr_hits   10896887376 # total number of address-
predicted hits
bpred_bimod.dir_hits    10898055781 # total number of direction-
predicted hits (includes addr-hits)
bpred_bimod.misses       1154402538 # total number of misses
bpred_bimod.jr_hits        39772212 # total number of address-
predicted hits for JR's
bpred_bimod.jr_seen        40956662 # total number of JR's seen
bpred_bimod.jr_non_ras_hits.PP      1250351 # total number of address-
predicted hits for non-RAS JR's
bpred_bimod.jr_non_ras_seen.PP      1266947 # total number of non-RAS
JR's seen
```

```
bpred_bimod.bpred_addr_rate    0.9041 # branch address-prediction rate
(i.e., addr-hits/updates)

bpred_bimod.bpred_dir_rate    0.9042 # branch direction-prediction
rate (i.e., all-hits/updates)

bpred_bimod.bpred_jr_rate    0.9711 # JR address-prediction rate
(i.e., JR addr-hits/JRs seen)

bpred_bimod.bpred_jr_non_ras_rate.PP    0.9869 # non-RAS JR addr-pred
rate (ie, non-RAS JR hits/JRs seen)

bpred_bimod.retstack_pushes     72911441 # total number of address
pushed onto ret-addr stack

bpred_bimod.retstack_pops     57318927 # total number of address
popped off of ret-addr stack

bpred_bimod.used_ras.PP     39689715 # total number of RAS predictions
used

bpred_bimod.ras_hits.PP     38521861 # total number of RAS hits

bpred_bimod.ras_rate.PP    0.9706 # RAS prediction rate (i.e., RAS
hits/used RAS)

il1.accesses            63206782930 # total number of accesses

il1.hits                63176996182 # total number of hits

il1.misses                 29786748 # total number of misses

il1.replacements           29786241 # total number of replacements

il1.writebacks                    0 # total number of writebacks

il1.invalidations                 0 # total number of invalidations

il1.miss_rate                0.0005 # miss rate (i.e., misses/ref)

il1.repl_rate                0.0005 # replacement rate (i.e.,
repls/ref)

il1.wb_rate                  0.0000 # writeback rate (i.e., wrbks/ref)

il1.inv_rate                 0.0000 # invalidation rate (i.e.,
invs/ref)

dl1.accesses            22400517890 # total number of accesses

dl1.hits                15296018191 # total number of hits
```

```
dl1.misses               7104499699 # total number of misses
dl1.replacements         7104499187 # total number of replacements
dl1.writebacks           1334404919 # total number of writebacks
dl1.invalidations                0 # total number of invalidations
dl1.miss_rate               0.3172 # miss rate (i.e., misses/ref)
dl1.repl_rate               0.3172 # replacement rate (i.e.,
repls/ref)
dl1.wb_rate                 0.0596 # writeback rate (i.e., wrbks/ref)
dl1.inv_rate                0.0000 # invalidation rate (i.e.,
invs/ref)
ul2.accesses             8468691366 # total number of accesses
ul2.hits                 3251946297 # total number of hits
ul2.misses               5216745069 # total number of misses
ul2.replacements         5216740973 # total number of replacements
ul2.writebacks           1207172485 # total number of writebacks
ul2.invalidations                0 # total number of invalidations
ul2.miss_rate               0.6160 # miss rate (i.e., misses/ref)
ul2.repl_rate               0.6160 # replacement rate (i.e.,
repls/ref)
ul2.wb_rate                 0.1425 # writeback rate (i.e., wrbks/ref)
ul2.inv_rate                0.0000 # invalidation rate (i.e.,
invs/ref)
itlb.accesses           63206782930 # total number of accesses
itlb.hits               63206782903 # total number of hits
itlb.misses                    27 # total number of misses
itlb.replacements               0 # total number of replacements
itlb.writebacks                 0 # total number of writebacks
itlb.invalidations              0 # total number of invalidations
itlb.miss_rate              0.0000 # miss rate (i.e., misses/ref)
```

```
itlb.repl_rate                0.0000 # replacement rate (i.e.,
repls/ref)
itlb.wb_rate                  0.0000 # writeback rate (i.e., wrbks/ref)
itlb.inv_rate                 0.0000 # invalidation rate (i.e.,
invs/ref)
dtlb.accesses           22423679096 # total number of accesses
dtlb.hits               20737230192 # total number of hits
dtlb.misses              1686448904 # total number of misses
dtlb.replacements        1686448776 # total number of replacements
dtlb.writebacks                 0 # total number of writebacks
dtlb.invalidations              0 # total number of invalidations
dtlb.miss_rate                0.0752 # miss rate (i.e., misses/ref)
dtlb.repl_rate                0.0752 # replacement rate (i.e.,
repls/ref)
dtlb.wb_rate                  0.0000 # writeback rate (i.e., wrbks/ref)
dtlb.inv_rate                 0.0000 # invalidation rate (i.e.,
invs/ref)
sim_invalid_addrs               0 # total non-speculative bogus
addresses seen (debug var)
ld_text_base             0x00400000 # program text (code) segment base
ld_text_size                113136 # program text (code) size in
bytes
ld_data_base             0x10000000 # program initialized data segment
base
ld_data_size                 19060 # program init'ed `.data' and
uninit'ed `.bss' size in bytes
ld_stack_base            0x7fffc000 # program stack segment base
(highest address in stack)
ld_stack_size                16384 # program initial stack size
ld_prog_entry            0x00400140 # program entry point (initial PC)
```

```
ld_environ_base          0x7fff8000 # program environment base address
address
ld_target_big_endian             0 # target executable endian-ness,
non-zero if big endian
mem.page_count               24435 # total number of pages allocated
mem.page_mem                 97740k # total size of memory pages
allocated
mem.ptab_misses             6773789 # total first level page table
misses
mem.ptab_accesses      426568014126 # total page table accesses
mem.ptab_miss_rate           0.0000 # first level page table miss rate
```

(6) sim-profile

```
sh861201@eustis:~/SimpleScalar/simplesim-3.0/profile_Dir$ ../sim-
profile -iprof true -dumpconfig config_file.config
/home/sh861201/SimpleScalar/simplesim-3.0/benchmark/mcf/mcf/Mcf
/home/sh861201/SimpleScalar/simplesim-3.0/benchmark/mcf/mcf/inp.in

sim-profile: SimpleScalar/PISA Tool Set version 3.0 of August, 2003.

Copyright (c) 1994-2003 by Todd M. Austin, Ph.D. and SimpleScalar,
LLC.

All Rights Reserved. This version of SimpleScalar is licensed for
academic

non-commercial use.  No portion of this work may be used by any
commercial

entity, or for any commercial purpose, without the prior written
permission

of SimpleScalar, LLC (info@simplescalar.com).


sim: command line: ../sim-profile -iprof true -dumpconfig
config_file.config /home/sh861201/SimpleScalar/simplesim-
3.0/benchmark/mcf/mcf/Mcf /home/sh861201/SimpleScalar/simplesim-
3.0/benchmark/mcf/mcf/inp.in


sim: simulation started @ Sat Feb 27 14:25:42 2016, options follow:


sim-profile: This simulator implements a functional simulator with

profiling support.  Run with the `-h' flag to see profiling options

available.


# -config                     # load configuration from a file

# -dumpconfig                 # dump configuration to a file

# -h                    false # print help message
```

```
# -v                    false # verbose operation
# -d                    false # enable debug message
# -i                    false # start in Dlite debugger
-seed                     1 # random number generator seed (0 for
timer seed)
# -q                    false # initialize and terminate immediately
# -chkpt               <null> # restore EIO trace execution from
<fname>
# -redir:sim           <null> # redirect simulator output to file
(non-interactive only)
# -redir:prog          <null> # redirect simulated program output to
file
-nice                     0 # simulator scheduling priority
-max:inst                 0 # maximum number of inst's to execute
-all                  false # enable all profile options
-iclass               false # enable instruction class profiling
-iprof                 true # enable instruction profiling
-brprof               false # enable branch instruction profiling
-amprof               false # enable address mode profiling
-segprof              false # enable load/store address segment
profiling
-tsymprof             false # enable text symbol profiling
-taddrprof            false # enable text address profiling
-dsymprof             false # enable data symbol profiling
-internal             false # include compiler-internal symbols
during symbol profiling
# -pcstat              <null> # profile stat(s) against text addr's
(mult uses ok)


sim: ** starting functional simulation **
```

```
MCF SPEC version 1.6.I
by  Andreas Loebel
Copyright (c) 1998,1999   ZIB Berlin
All Rights Reserved.

nodes                      : 16555
active arcs                : 244246
simplex iterations         : 182415
flow value                 : 8980173901
new implicit arcs          : 300000
active arcs                : 544246
simplex iterations         : 189170
flow value                 : 8910169940
new implicit arcs          : 300000
active arcs                : 844246
simplex iterations         : 216493
flow value                 : 8650168945
new implicit arcs          : 300000
active arcs                : 1144246
simplex iterations         : 261464
flow value                 : 8570161464
new implicit arcs          : 300000
active arcs                : 1444246
simplex iterations         : 290615
flow value                 : 8570159306
new implicit arcs          : 300000
active arcs                : 1744246
```

```
simplex iterations         : 318729
flow value                 : 8570157650
new implicit arcs          : 300000
active arcs                : 2044246
simplex iterations         : 340078
flow value                 : 8570156531
new implicit arcs          : 300000
active arcs                : 2344246
simplex iterations         : 354548
flow value                 : 8570156010
new implicit arcs          : 77333
active arcs                : 2421579
simplex iterations         : 361819
flow value                 : 8570155949
new implicit arcs          : 1100
active arcs                : 2422679
simplex iterations         : 361826
flow value                 : 8570155949
checksum                   : 258659426
optimal


sim: ** simulation statistics **
sim_num_insn            49073291786 # total number of instructions
executed
sim_num_refs            20768099818 # total number of loads and stores
executed
sim_elapsed_time               6434 # total simulation time in seconds
sim_inst_rate          7627182.4349 # simulation speed (in insts/sec)
```

```
sim_inst_prof          # instruction profile
sim_inst_prof.array_size = 119
sim_inst_prof.bucket_size = 1
sim_inst_prof.count = 119
sim_inst_prof.total = 23303488009
sim_inst_prof.imin = 0
sim_inst_prof.imax = 119
sim_inst_prof.average = 195827630.3277
sim_inst_prof.std_dev = 626293071.4131
sim_inst_prof.overflows = 0
# pdf == prob dist fn, cdf == cumulative dist fn
#          index      count    pdf
sim_inst_prof.start_dist
nop                     764   0.00
j        J        651618221   2.80
jal      J         39688181   0.17
jr       s         40956662   0.18
jalr     d,s            1536   0.00
beq      s,t,j   2455310771  10.54
bne      s,t,j   2132627839   9.15
blez     s,j      762830491   3.27
bgtz     s,j         393358   0.00
bltz     s,j         622677   0.00
bgez     s,j     1673441287   7.18
bc1f     j                0   0.00
bc1t     j                0   0.00
lb       t,o(b)     7216069   0.03
lbu      t,o(b)    15114749   0.06
```

lh	t,o(b)	25	0.00
lhu	t,o(b)	8043447	0.03
lw	t,o(b)	831357301	3.57
dlw	t,o(b)	71861	0.00
l.s	T,o(b)	50	0.00
l.d	T,o(b)	0	0.00
lwl	t,o(b)	0	0.00
lwr	t,o(b)	0	0.00
sb	t,o(b)	8734370	0.04
sh	t,o(b)	0	0.00
sw	t,o(b)	2717620861	11.66
dsw	t,o(b)	71861	0.00
dsz	o(b)	0	0.00
s.s	T,o(b)	40	0.00
s.d	T,o(b)	0	0.00
swl	t,o(b)	0	0.00
swr	t,o(b)	0	0.00
lb	t,(b+d)	0	0.00
lbu	t,(b+d)	0	0.00
lh	t,(b+d)	0	0.00
lhu	t,(b+d)	0	0.00
lw	t,(b+d)	0	0.00
dlw	t,(b+d)	0	0.00
l.s	T,(b+d)	0	0.00
l.d	T,(b+d)	0	0.00
sb	t,(b+d)	0	0.00
sh	t,(b+d)	0	0.00
sw	t,(b+d)	0	0.00

dsw	t,(b+d)	0	0.00
dsz	(b+d)	0	0.00
s.s	T,(b+d)	0	0.00
s.d	T,(b+d)	0	0.00
l.s.r2	T,(b+d)	0	0.00
s.s.r2	T,(b+d)	0	0.00
lw.r2	t,(b+d)	0	0.00
sw.r2	t,(b+d)	0	0.00
add	d,s,t	0	0.00
addi	t,s,i	0	0.00
addu	d,s,t	2175186010	9.33
addiu	t,s,i	3446779664	14.79
sub	d,s,t	0	0.00
subu	d,s,t	1969282580	8.45
mult	s,t	2820831	0.01
multu	s,t	1012	0.00
div	s,t	0	0.00
divu	s,t	904713	0.00
mfhi	d	1267507	0.01
mthi	s	0	0.00
mflo	d	3726556	0.02
mtlo	s	0	0.00
and	d,s,t	2147816	0.01
andi	t,s,u	19291912	0.08
or	d,s,t	1067609	0.00
ori	t,s,u	14642976	0.06
xor	d,s,t	398040	0.00
xori	t,s,u	160422	0.00

nor	d,s,t	70919	0.00
sll	d,t,H	518726247	2.23
sllv	d,t,s	432515	0.00
srl	d,t,H	33282197	0.14
srlv	d,t,s	360773	0.00
sra	d,t,H	38644117	0.17
srav	d,t,s	6	0.00
slt	d,s,t	2562709004	11.00
slti	t,s,i	67183788	0.29
sltu	d,s,t	1022147939	4.39
sltiu	t,s,i	3083867	0.01
add.s	D,S,T	0	0.00
add.d	D,S,T	10	0.00
sub.s	D,S,T	0	0.00
sub.d	D,S,T	0	0.00
mul.s	D,S,T	0	0.00
mul.d	D,S,T	10	0.00
div.s	D,S,T	0	0.00
div.d	D,S,T	0	0.00
abs.s	D,S	0	0.00
abs.d	D,S	0	0.00
mov.s	D,S	0	0.00
mov.d	D,S	20	0.00
neg.s	D,S	0	0.00
neg.d	D,S	0	0.00
cvt.s.d	D,S	0	0.00
cvt.s.w	D,S	0	0.00
cvt.d.s	D,S	0	0.00

```
cvt.d.w  D,S             30   0.00
cvt.w.s  D,S              0   0.00
cvt.w.d  D,S              0   0.00
c.eq.s   S,T              0   0.00
c.eq.d   S,T              0   0.00
c.lt.s   S,T              0   0.00
c.lt.d   S,T              0   0.00
c.le.s   S,T              0   0.00
c.le.d   S,T              0   0.00
sqrt.s   D,S              0   0.00
sqrt.d   D,S              0   0.00
syscall                 773   0.00
break    B                0   0.00
lui      t,U       73445655   0.32
mfc1     t,S             40   0.00
dmfc1    t,S             10   0.00
cfc1     t,S              0   0.00
mtc1     t,S             20   0.00
dmtc1    t,S              0   0.00
ctc1     t,S              0   0.00
sim_inst_prof.end_dist


ld_text_base             0x00400000 # program text (code) segment base
ld_text_size                 113136 # program text (code) size in
bytes
ld_data_base             0x10000000 # program initialized data segment
base
ld_data_size                  19060 # program init'ed `.data' and
uninit'ed `.bss' size in bytes
```

```
ld_stack_base            0x7fffc000 # program stack segment base
(highest address in stack)
ld_stack_size                 16384 # program initial stack size
ld_prog_entry            0x00400140 # program entry point (initial PC)
ld_environ_base          0x7fff8000 # program environment base address
address
ld_target_big_endian              0 # target executable endian-ness,
non-zero if big endian
mem.page_count                24435 # total number of pages allocated
mem.page_mem                  97740k # total size of memory pages
allocated
mem.ptab_misses            10414410 # total first level page table
misses
mem.ptab_accesses      237836123995 # total page table accesses
mem.ptab_miss_rate           0.0000 # first level page table miss rate
```

(7) sim-safe

```
sh861201@eustis:~/SimpleScalar/simplesim-3.0/safe_Dir$ ../sim-safe -
max:inst 10000000 -dumpconfig config_file.config
/home/sh861201/SimpleScalar/simplesim-3.0/benchmark/mcf/mcf/Mcf
/home/sh861201/SimpleScalar/simplesim-3.0/benchmark/mcf/mcf/inp.in

sim-safe: SimpleScalar/PISA Tool Set version 3.0 of August, 2003.

Copyright (c) 1994-2003 by Todd M. Austin, Ph.D. and SimpleScalar,
LLC.

All Rights Reserved. This version of SimpleScalar is licensed for
academic

non-commercial use.  No portion of this work may be used by any
commercial

entity, or for any commercial purpose, without the prior written
permission

of SimpleScalar, LLC (info@simplescalar.com).


sim: command line: ../sim-safe -max:inst 10000000 -dumpconfig
config_file.config /home/sh861201/SimpleScalar/simplesim-
3.0/benchmark/mcf/mcf/Mcf /home/sh861201/SimpleScalar/simplesim-
3.0/benchmark/mcf/mcf/inp.in


sim: simulation started @ Sat Feb 27 17:10:24 2016, options follow:


sim-safe: This simulator implements a functional simulator.  This

functional simulator is the simplest, most user-friendly simulator in
the

simplescalar tool set.  Unlike sim-fast, this functional simulator
checks

for all instruction errors, and the implementation is crafted for
clarity

rather than speed.
```

```
# -config                     # load configuration from a file
# -dumpconfig                 # dump configuration to a file
# -h                    false # print help message
# -v                    false # verbose operation
# -d                    false # enable debug message
# -i                    false # start in Dlite debugger
-seed                       1 # random number generator seed (0 for
timer seed)
# -q                    false # initialize and terminate immediately
# -chkpt               <null> # restore EIO trace execution from
<fname>
# -redir:sim           <null> # redirect simulator output to file
(non-interactive only)
# -redir:prog          <null> # redirect simulated program output to
file
-nice                       0 # simulator scheduling priority
-max:inst            10000000 # maximum number of inst's to execute

sim: ** starting functional simulation **

MCF SPEC version 1.6.I
by  Andreas Loebel
Copyright (c) 1998,1999   ZIB Berlin
All Rights Reserved.

sim: ** simulation statistics **
sim_num_insn               10000000 # total number of instructions
executed
```

```
sim_num_refs                6663806 # total number of loads and stores
executed
sim_elapsed_time                  1 # total simulation time in seconds
sim_inst_rate          10000000.0000 # simulation speed (in insts/sec)
ld_text_base             0x00400000 # program text (code) segment base
ld_text_size                113136 # program text (code) size in
bytes
ld_data_base             0x10000000 # program initialized data segment
base
ld_data_size                 19060 # program init'ed `.data' and
uninit'ed `.bss' size in bytes
ld_stack_base            0x7fffc000 # program stack segment base
(highest address in stack)
ld_stack_size                16384 # program initial stack size
ld_prog_entry            0x00400140 # program entry point (initial PC)
ld_environ_base          0x7fff8000 # program environment base address
address
ld_target_big_endian             0 # target executable endian-ness,
non-zero if big endian
mem.page_count                6540 # total number of pages allocated
mem.page_mem                 26160k # total size of memory pages
allocated
mem.ptab_misses               8589 # total first level page table
misses
mem.ptab_accesses          54038872 # total page table accesses
mem.ptab_miss_rate           0.0002 # first level page table miss rate
```