```
sim-outorder: SimpleScalar/Alpha Tool Set version 3.0 of November, 2000.
Copyright (c) 1994-2000 by Todd M. Austin.  All Rights Reserved.
This version of SimpleScalar is licensed for academic non-commercial use only.

warning: section `.comment' ignored...
sim: command line: sim-outorder -cache:dl2 ul2:512:128:4:l -mem:lat 60 4
tests/bin/test-math

sim: simulation started @ Mon Nov 14 18:00:38 2005, 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       ul2:512:128:4: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         60 4 # 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 **
pow(12.0, 2.0) == 144.000000
pow(10.0, 3.0) == 1000.000000
pow(10.0, -3.0) == 0.001000
str: 123.456
x: 123.000000
str: 123.456
x: 123.456000
str: 123.456
x: 123.456000
123.456 123.456000 123 1000
sinh(2.0) = 3.62686
sinh(3.0) = 10.0179
h=3.60555
atan2(3,2) = 0.982794
pow(3.60555,4.0) = 169
169 / exp(0.982794 * 5) = 1.24102
3.93117 + 5*log(3.60555) = 10.3435
cos(10.3435) = -0.606798,  sin(10.3435) = -0.794856
x     0.5x
x0.5     x
x   0.5x
-1e-17 == -1e-17 Worked!
warning: partially supported sigprocmask() call...

sim: ** simulation statistics **
sim_num_insn                  46391 # total number of instructions committed
sim_num_refs                  12815 # total number of loads and stores committed
```

```
sim_num_loads                  7949 # total number of loads committed
sim_num_stores            4866.0000 # total number of stores committed
sim_num_branches               6768 # total number of branches committed
sim_elapsed_time                  1 # total simulation time in seconds
sim_inst_rate            46391.0000 # simulation speed (in insts/sec)
sim_total_insn                51740 # total number of instructions executed
sim_total_refs                14059 # total number of loads and stores executed
sim_total_loads                8872 # total number of loads executed
sim_total_stores          5187.0000 # total number of stores executed
sim_total_branches             7530 # total number of branches executed
sim_cycle                     75907 # total simulation time in cycles
sim_IPC                      0.6112 # instructions per cycle
sim_CPI                      1.6362 # cycles per instruction
sim_exec_BW                  0.6816 # total instructions (mis-spec + committed)
per cycle
sim_IPB                      6.8545 # instruction per branch
IFQ_count                     92227 # cumulative IFQ occupancy
IFQ_fcount                    20675 # cumulative IFQ full count
ifq_occupancy                1.2150 # avg IFQ occupancy (insn's)
ifq_rate                     0.6816 # avg IFQ dispatch rate (insn/cycle)
ifq_latency                  1.7825 # avg IFQ occupant latency (cycle's)
ifq_full                     0.2724 # fraction of time (cycle's) IFQ was full
RUU_count                    381279 # cumulative RUU occupancy
RUU_fcount                    12033 # cumulative RUU full count
ruu_occupancy                5.0230 # avg RUU occupancy (insn's)
ruu_rate                     0.6816 # avg RUU dispatch rate (insn/cycle)
ruu_latency                  7.3691 # avg RUU occupant latency (cycle's)
ruu_full                     0.1585 # fraction of time (cycle's) RUU was full
LSQ_count                    111657 # cumulative LSQ occupancy
LSQ_fcount                     2095 # cumulative LSQ full count
lsq_occupancy                1.4710 # avg LSQ occupancy (insn's)
lsq_rate                     0.6816 # avg LSQ dispatch rate (insn/cycle)
lsq_latency                  2.1580 # avg LSQ occupant latency (cycle's)
lsq_full                     0.0276 # fraction of time (cycle's) LSQ was full
sim_slip                     515159 # total number of slip cycles
avg_sim_slip                11.1047 # the average slip between issue and
retirement
bpred_bimod.lookups            7795 # total number of bpred lookups
bpred_bimod.updates            6768 # total number of updates
bpred_bimod.addr_hits          5498 # total number of address-predicted hits
bpred_bimod.dir_hits           5879 # total number of direction-predicted hits
(includes addr-hits)
bpred_bimod.misses              889 # total number of misses
bpred_bimod.jr_hits             646 # total number of address-predicted hits for
JR's
bpred_bimod.jr_seen             738 # total number of JR's seen
bpred_bimod.jr_non_ras_hits.PP           78 # total number of address-predicted
hits for non-RAS JR's
bpred_bimod.jr_non_ras_seen.PP          120 # total number of non-RAS JR's seen
bpred_bimod.bpred_addr_rate    0.8124 # branch address-prediction rate (i.e.,
addr-hits/updates)
bpred_bimod.bpred_dir_rate    0.8686 # branch direction-prediction rate (i.e.,
all-hits/updates)
bpred_bimod.bpred_jr_rate    0.8753 # JR address-prediction rate (i.e., JR addr-
hits/JRs seen)
bpred_bimod.bpred_jr_non_ras_rate.PP    0.6500 # non-RAS JR addr-pred rate (ie,
non-RAS JR hits/JRs seen)
```

```
bpred_bimod.retstack_pushes          685 # total number of address pushed onto
ret-addr stack
bpred_bimod.retstack_pops          684 # total number of address popped off of
ret-addr stack
bpred_bimod.used_ras.PP          618 # total number of RAS predictions used
bpred_bimod.ras_hits.PP          568 # total number of RAS hits
bpred_bimod.ras_rate.PP    0.9191 # RAS prediction rate (i.e., RAS hits/used RAS)
il1.accesses                  55838 # total number of accesses
il1.hits                      53323 # total number of hits
il1.misses                     2515 # total number of misses
il1.replacements               2077 # total number of replacements
il1.writebacks                    0 # total number of writebacks
il1.invalidations                 0 # total number of invalidations
il1.miss_rate                0.0450 # miss rate (i.e., misses/ref)
il1.repl_rate                0.0372 # 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                  12893 # total number of accesses
dl1.hits                      12511 # total number of hits
dl1.misses                      382 # total number of misses
dl1.replacements                 25 # total number of replacements
dl1.writebacks                   14 # total number of writebacks
dl1.invalidations                 0 # total number of invalidations
dl1.miss_rate                0.0296 # miss rate (i.e., misses/ref)
dl1.repl_rate                0.0019 # replacement rate (i.e., repls/ref)
dl1.wb_rate                  0.0011 # writeback rate (i.e., wrbks/ref)
dl1.inv_rate                 0.0000 # invalidation rate (i.e., invs/ref)
ul2.accesses                   2911 # total number of accesses
ul2.hits                       2456 # total number of hits
ul2.misses                      455 # total number of misses
ul2.replacements                  0 # total number of replacements
ul2.writebacks                    0 # total number of writebacks
ul2.invalidations                 0 # total number of invalidations
ul2.miss_rate                0.1563 # miss rate (i.e., misses/ref)
ul2.repl_rate                0.0000 # replacement rate (i.e., repls/ref)
ul2.wb_rate                  0.0000 # writeback rate (i.e., wrbks/ref)
ul2.inv_rate                 0.0000 # invalidation rate (i.e., invs/ref)
itlb.accesses                 55838 # total number of accesses
itlb.hits                     55814 # total number of hits
itlb.misses                      24 # 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               0.0004 # 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                 13117 # total number of accesses
dtlb.hits                     13095 # total number of hits
dtlb.misses                      22 # total number of misses
dtlb.replacements                 0 # total number of replacements
dtlb.writebacks                   0 # total number of writebacks
dtlb.invalidations                0 # total number of invalidations
dtlb.miss_rate               0.0017 # miss rate (i.e., misses/ref)
dtlb.repl_rate               0.0000 # 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           0x0120000000 # program text (code) segment base
ld_text_size                 188416 # program text (code) size in bytes
ld_data_base           0x0140000000 # program initialized data segment base
ld_data_size                  41984 # program init'ed `.data' and uninit'ed
`.bss' size in bytes
ld_stack_base          0x011ff9b000 # program stack segment base (highest
address in stack)
ld_stack_size                 16384 # program initial stack size
ld_prog_entry          0x012000f750 # program entry point (initial PC)
ld_environ_base        0x011ff97000 # program environment base address address
ld_target_big_endian              0 # target executable endian-ness, non-zero if
big endian
mem.page_count                   28 # total number of pages allocated
mem.page_mem                   224k # total size of memory pages allocated
mem.ptab_misses                  78 # total first level page table misses
mem.ptab_accesses            689922 # total page table accesses
mem.ptab_miss_rate           0.0001 # first level page table miss rate
```