Patrick Trinkle
CMSC611
Homework
Fall 2008

5.20:

I ran through the simulations with two different specs. a-d on both are below separated by SPEC, part e falls at the bottom.

Each TLB miss requires 20 instructions.
Number of extra overhead instructions = (Number of Accesses * Miss Rate)
% Time Wasted with Overhead =
(Overhead Inst / (Original Instructions + Overhead Inst))

This means that now we have extra instructions to execute; so of the new total instructions, how many of these are purely overhead.

```
~olano/public/ss/simplesim-3.0/sim-cache -tlb:dtlb dtlb:512:4096:2:l
~olano/public/ss/SPEC/mcf00.outorderO0.gcc.100M.ss
```

I used the cache replacement algorithm of least recently used for all simulations. The SPEC I ran was small enough that there were few misses. This might explain the strange results from various simulations.

```
sim_num_insn          6703
dtlb.accesses         3767
```

a) 128 Entries, 2 way, 4-64KB LRU
4KB: sim-cache -tlb:dtlb dtlb:64:4096:2:l
8KB: sim-cache -tlb:dtlb dtlb:64:8192:2:l
16KB: sim-cache -tlb:dtlb dtlb:64:16384:2:l
32KB: sim-cache -tlb:dtlb dtlb:64:32768:2:l
64KB: sim-cache -tlb:dtlb dtlb:64:65536:2:l

Page Size	Data TLB Miss Rate	Data TLB Miss %	TLB Overhead Inst	% Time Wasted
4096	0.0021	0.21	792	10.567
8192	0.0016	0.16	603	8.253
16384	0.0011	0.11	414	5.817
32768	0.0008	0.08	301	4.298
65536	0.0005	0.05	188	2.728

b) 256 Entries, 2 way, 4-64KB
4KB: sim-cache -tlb:dtlb dtlb:128:4096:2:l
8KB: sim-cache -tlb:dtlb dtlb:128:8192:2:l

```
16KB: sim-cache -tlb:dtlb dtlb:128:16384:2:l
32KB: sim-cache -tlb:dtlb dtlb:128:32768:2:l
64KB: sim-cache -tlb:dtlb dtlb:128:65536:2:l
```

Page Size	Data TLB Miss Rate	Data TLB Miss %	TLB Overhead Inst	% Time Wasted
4096	0.0021	0.21	792	10.567
8192	0.0016	0.16	603	8.253
16384	0.0011	0.11	414	5.817
32768	0.0008	0.08	301	4.298
65536	0.0005	0.05	188	2.728

c) 512 Entries, 2 way, 4-64KB

```
4KB: sim-cache -tlb:dtlb dtlb:256:4096:2:l
8KB: sim-cache -tlb:dtlb dtlb:256:8192:2:l
16KB: sim-cache -tlb:dtlb dtlb:256:16384:2:l
32KB: sim-cache -tlb:dtlb dtlb:256:32768:2:l
64KB: sim-cache -tlb:dtlb dtlb:256:65536:2:l
```

Page Size	Data TLB Miss Rate	Data TLB Miss %	TLB Overhead Inst	% Time Wasted
4096	0.0021	0.21	792	10.567
8192	0.0016	0.16	603	8.253
16384	0.0011	0.11	414	5.817
32768	0.0008	0.08	301	4.298
65536	0.0005	0.05	188	2.728

d) 1024 entries, 2 way, 4-64KB

```
4KB: sim-cache -tlb:dtlb dtlb:512:4096:2:l
8KB: sim-cache -tlb:dtlb dtlb:512:8192:2:l
16KB: sim-cache -tlb:dtlb dtlb:512:16384:2:l
32KB: sim-cache -tlb:dtlb dtlb:512:32768:2:l
64KB: sim-cache -tlb:dtlb dtlb:512:65536:2:l
```

Page Size	Data TLB Miss Rate	Data TLB Miss %	TLB Overhead Inst	% Time Wasted
4096	0.0021	0.21	792	10.567
8192	0.0016	0.16	603	8.253
16384	0.0011	0.11	414	5.817
32768	0.0008	0.08	301	4.298
65536	0.0005	0.05	188	2.728

```
~olano/public/ss/simplesim-3.0/sim-cache -max:inst 100000000 -tlb:dtlb
dtlb:64:4096:2:l ~olano/public/ss/SPEC/gzip00.outorderO0.gcc.100M.ss
~tri1/sim_configs.txt
sim_num_insn           100000000
dtlb.accesses            46876116
```

a) 128 Entries, 2 way, 4-64KB LRU

Page Size	Data TLB Miss Rate	Data TLB Miss %	TLB Overhead Inst	% Time Wasted
4096	0.0021	0.21	9843984	8.962
8192	0.0011	0.11	5156372	4.904
16384	0.0005	0.05	2343805	2.290
32768	0.0003	0.03	1406283	1.387
65536	0.0001	0.01	468761	0.467

b) 256 Entries, 2 way, 4-64KB

Page Size	Data TLB Miss Rate	Data TLB Miss %	TLB Overhead Inst	% Time Wasted
4096	0.0021	0.21	9843984	8.962
8192	0.0010	0.10	4687612	4.478
16384	0.0005	0.05	2343805	2.290
32768	0.0003	0.03	1406283	1.387
65536	0.0001	0.01	468761	0.467

c) 512 Entries, 2 way, 4-64KB

Page Size	Data TLB Miss Rate	Data TLB Miss %	TLB Overhead Inst	% Time Wasted
4096	0.0021	0.21	9843984	8.962
8192	0.0010	0.10	4687612	4.478
16384	0.0005	0.05	2343805	2.290
32768	0.0002	0.02	937522	0.929
65536	0.0001	0.01	468761	0.467

d) 1024 entries, 2 way, 4-64KB

Page Size	Data TLB Miss Rate	Data TLB Miss %	TLB Overhead Inst	% Time Wasted
4096	0.0021	0.21	9843984	8.962
8192	0.0010	0.10	4687612	4.478
16384	0.0005	0.05	2343805	2.290
32768	0.0002	0.02	937522	0.929
65536	0.0001	0.01	468761	0.467

Amusingly enough it appears that the amount of entries doesn't seem to impact the amount of misses, only the increase in page size.

e) What would be the effect on TLB miss rate and overhead for a multitasking environment? How would the context switch frequency affect the overhead?

In a multitasking environment, two situations can arise depending on implementation. If the task switch causes the caches to flush then there will be a lot of overhead flushing the caches. If the task switch doesn't then there will be all initial misses (as would happen either way). Therefore it makes more sense in a multitasking environment to not flush the cache. To a certain degree the following case could occur. Many small programs in a multitasking environment might incidentally share the cache and their blocks might not be replaced when they switch out. Of course, if you're using Least Recently Used, then those smaller programs are guaranteed to be flushed out when the next program begins execution. Therefore to arrive at a strange sharing a Random replacement algorithm might provide the best results.