Page Tables Revisited

Two-level Translation

R3000 TLB Refill

Virtual Linear Array page table

Virtual Linear Array Operation
Virtual Linear Array Page Table
- Use Context register to simply load PTE by indexing a PTE array in virtual memory
- Occasionally, will get double faults
  - A TLB miss, while servicing a TLB miss
  - Handled by general exception handler
- Protected from user access
- Use Context register to simply load PTE by indexing a PTE array in virtual memory

c0 Context Register
- c0_Context = PTEBase + 4 * PageNumber
  - PTEs are 4 bytes
  - PTEBase is the base local of the page table array (note: aligned on 4 MB boundary)
  - PTEBase is (re)initialised by the OS whenever the page table array is changed
  - E.g. on a context switch
- Occasionally, will get double faults
  - A TLB miss, while servicing a TLB miss
  - Handled by general exception handler

Code for VLA TLB refill handler
- Load PTE address from context register
- Move the PTE into EntryLo
- Write EntryLo into random TLB entry
- Load address of instruction to return to
- Load the PTE, note: this load can cause a TLB refill miss itself, but this miss is handled by the general exception vector.
- The general exception vector has to understand this situation and deal with it appropriately
- Return from the exception

Software-loaded TLB
- Pros
  - Can simplify hardware design
  - Provide greater flexibility in page table structure
- Cons
  - Typically have slower refill times than hardware managed TLBs.

Trends
- Operating systems
  - Moving functionality into user processes
  - Making greater use of virtual memory for mapping data structures held within the kernel.
- RAM is increasing
  - TLB capacity is relatively static
- Statement:
  - Trends place greater stress upon the TLB by increasing miss rates and hence, decreasing overall system performance.
  - True/False? How to evaluate?
Figure 1: Tapeworm

The Tapeworm TLB simulator is built into the operating system and is invoked whenever there is a real TLB miss. The simulator uses the real TLB misses to simulate its own TLB configuration(s), because the simulator resides in the operating system, Tapeworm captures the dynamic nature of the system and avoids the problems associated with simulators driven by static traces.

<table>
<thead>
<tr>
<th>TLB Miss Type</th>
<th>Unix</th>
<th>OSF/1</th>
<th>Mach 3.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>L1U</td>
<td>50</td>
<td>32</td>
<td>20</td>
</tr>
<tr>
<td>L1K</td>
<td>504</td>
<td>516</td>
<td>467</td>
</tr>
<tr>
<td>L2</td>
<td>554</td>
<td>554</td>
<td>554</td>
</tr>
<tr>
<td>L3</td>
<td>555</td>
<td>555</td>
<td>555</td>
</tr>
<tr>
<td>Misc</td>
<td>546</td>
<td>456</td>
<td>456</td>
</tr>
</tbody>
</table>

Table 3: Costs for Different TLB Miss Types

This table shows the number of cycles of the various TLB misses required to service different types of TLB misses. The number of cycles varies between 0 (if TLB hit) and 1000 (if TLB miss). We separate TLB miss types into the 4 categories described below, noting that Unix does not have L3 misses because it implements a 4-level page table.

- L1U: TLB miss on level 1 user PTE.
- L1K: TLB miss on level 4 kernel PTE.
- L2: TLB miss on level 2 PTE. This can only occur after a miss on a level 1 user PTE.
- L3: TLB miss on level 3 PTE. This can occur after a level 2 miss in a level 1 line.

Note the TLB miss costs

- What is expected to be the common case?
Specialising the L2/L1K miss vector

<table>
<thead>
<tr>
<th>Type of L2 Miss</th>
<th>Counts</th>
<th>Previous Total Cost (ms)</th>
<th>New Total Cost (ms)</th>
<th>Time Saved (ms)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mach 3.0/AFS0f</td>
<td>39,123,272</td>
<td>30.3</td>
<td>30.15</td>
<td>0.00</td>
</tr>
<tr>
<td>L2</td>
<td>359,085</td>
<td>9.99</td>
<td>0.79</td>
<td>7.20</td>
</tr>
<tr>
<td>L3</td>
<td>2,491,263</td>
<td>43.09</td>
<td>2.29</td>
<td>40.80</td>
</tr>
<tr>
<td>L4</td>
<td>690,441</td>
<td>11.85</td>
<td>11.85</td>
<td>0.00</td>
</tr>
<tr>
<td>Identify</td>
<td>177,248</td>
<td>3.81</td>
<td>3.81</td>
<td>0.00</td>
</tr>
<tr>
<td>Invalid</td>
<td>186,439</td>
<td>2.70</td>
<td>2.70</td>
<td>0.00</td>
</tr>
<tr>
<td>Total</td>
<td>53,832,410</td>
<td>106.56</td>
<td>68.20</td>
<td>38.36</td>
</tr>
</tbody>
</table>

Table 7: Recomputed Cost of TLB Misses Given Additional Miss Vectors (Mach 3.0)

Other performance improvements?

- In Paper
  - Pinned slots
  - Increased TLB size
  - TLB associativity
- Other options
  - Bigger page sizes
  - Multiple page sizes