Virtual Memory

• It's just another level in the cache/memory hierarchy
• *Virtual memory* is the name of the technique that allows us to view main memory as a cache of a larger memory space (on disk).

Virtual Memory

• is just cacheing, but uses different terminology
  
<table>
<thead>
<tr>
<th>cache</th>
<th>VM</th>
</tr>
</thead>
<tbody>
<tr>
<td>block</td>
<td>page</td>
</tr>
<tr>
<td>cache miss</td>
<td>page fault</td>
</tr>
<tr>
<td>address</td>
<td>virtual address</td>
</tr>
<tr>
<td>index</td>
<td>physical address (sort of)</td>
</tr>
</tbody>
</table>

Virtual Memory

• What happens if another program in the processor uses the same addresses that yours does?
• What happens if your program uses addresses that don’t exist in the machine?
• What happens to “holes” in the address space your program uses?

• So, virtual memory provides
  - performance (through the caching effect)
  - protection
  - ease of programming/compilation
  - efficient use of memory
Virtual Memory

• is just a mapping function from virtual memory addresses to physical memory locations, which allows caching of virtual pages in physical memory.

What makes VM different than memory caches

• **MUCH** higher miss penalty (millions of cycles)!
• Therefore
  – large pages [equivalent of cache line] (4 KB to MBs)
  – associative mapping of pages (typically fully associative)
  – software handling of misses (but not hits!!)
  – write-through not an option, only write-back

Virtual Memory mapping

Address translation via the page table

• all page mappings are in the page table, so hit/miss is determined solely by the valid bit (i.e., no tag)
• so why is this fully associative???
Making Address Translation Fast

• A cache for address translations: translation lookaside buffer

ASN == address space number (basically, Process ID)

Virtual Memory Key Points

• How does virtual memory provide:
  – protection?
  – sharing?
  – performance?
  – illusion of large main memory?

• Virtual Memory requires twice as many memory accesses, so we cache page table entries in the TLB.

• Three things can go wrong on a memory access: cache miss, TLB miss, page fault.