ADDRESS TRANSLATION AND TLB

Mahdi Nazm Bojnordi
Assistant Professor
School of Computing
University of Utah
Overview

- Announcement
  - Homework 4 submission deadline: Mar. 27th

- This lecture
  - Virtual memory
  - Page tables and address translation
  - Translation look-aside buffer (TLB)
Recall: Memory Hierarchy

- Lower levels provide greater capacity longer time
  - Does the program fit in main memory?
  - What if running multiple programs?

Greater Capacity

- **Cache**
  - Capacity: 8MB
  - Time: ~20 ns

- **Main Memory**
  - Capacity: 8GB
  - Time: ~250 ns

- **Secondary Memory**
  - Capacity: 500GB
  - Time: ~10 ms
Virtual Memory

- Use the main memory as a "cache" for secondary memory
  - Placement policy?

```plaintext
for(i=0; i<100;++i) {
    a[i]++;
}
```
Virtual Memory

- Use the main memory as a "cache" for secondary memory
  - Placement policy?
- Allow efficient and safe sharing the physical main memory among multiple programs
  - Replacement policy?
Virtual Memory Systems

- Provides illusion of very large memory
  - Address space of each program larger than the physical main memory
- Memory management unit (MMU)
  - Between main and secondary mem.
  - Address translation
    - Virtual address space used by the program
    - Physical address space is provided by the physical main memory
Virtual Address

- Every virtual address is translated to a physical address with the help of hardware.
- Data granularity

What is the table size?
Address Translation Issues

- Where to store the table?
  - Too big for on-chip cache
  - Should be maintained in the main memory

- What to do on a page table miss (page fault)?
  - No valid frame assigned to the virtual page
  - OS copies the page from disk to page frame

- What is the cost of address translation?
  - Additional accesses to main memory per every access
  - Optimizations?
Address Translation Cost

- Page walk: look up the physical address in the page table

- How many pages to store the page table?

Diagram:

- Virtual Address: 20 12
- Page Table
- Page frame No: 12
- Physical Address: 12
- base → Page Table
Multi-Level Page Table

- The virtual (logical) address space is broken down into multiple pages

  - Example: 4KB pages

```
<table>
<thead>
<tr>
<th>Virtual Address</th>
<th>Page frame No</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 10 12</td>
<td>12</td>
</tr>
</tbody>
</table>
```

Diagram:
- Virtual Address: 10 10 12
- Page frame No: 12
- Physical Address:
  - base
  - Physical Address: 12
Translation Lookaside Buffer

- Exploit locality to reduce address translation time
- Keep the translation in a buffer for future references

Diagram:
- Virtual Address
  - Virtual Page No
  - Offset
- Page Table
- Physical Memory
  - Page Frame No
  - Offset
- TLB
  - Page
  - Frame 0
  - Frame 1
  - Frame 2
  - ...
  - Frame N-1
Translation Lookaside Buffer

- Just like any other cache, the TLB can be organized as fully associative, set associative, or direct.
- TLB access is typically faster than cache access:
  - Because TLBs are much smaller than caches.
  - TLBs are typically not more than 128 to 256 entries even on high-end machines.

<table>
<thead>
<tr>
<th>V</th>
<th>Virtual Page #</th>
<th>Physical Page #</th>
<th>Dirty</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
CAM Based TLB

- Content addressable memory (CAM)
  - Unlike RAM, data in address out

RAM: Read Operation
- Address in
- Decoder
- Data in
- Data out

CAM: Search Operation
- Data in
- Address
- Match

match
CAM Based TLB

- Content addressable memory (CAM)
  - Unlike RAM, data in address out

- CAM based TLB
  - Both CAM and RAM are used

What if multiple rows match?
On a TLB miss, is the page loaded in memory?

- Yes: takes 10’s cycles to update the TLB
- No: page fault
  - Takes 1,000,000’s cycles to load the page and update TLB

Physically indexed, physically tagged: TLB on critical path!
Physically Indexed Caches

- **Problem:** increased critical path due to sequential access to TLB and cache

![Diagram showing the interaction between Virtual Page No, Page Offset, TLB, Physical Frame, and Data Array.]

**Observation:** lower address bits (page offset) are not translated.
Virtually Indexed Caches

- **Idea:** Index into cache in parallel with page number translation in TLB

![Diagram]

- Virtual Page No
- Page Offset
- TLB
- Index
- Byte
- Tag Array
- Data Array
- Data Block
- Hit/miss*

*what if the page offset is not equal to index + byte?