Overview

- Page Table
- Translation Lookaside Buffer (TLB)

Review: Memory Hierarchy

Upper Level
- Registers
- Instruction Operands
- Cache
- L2 Cache
- Memory

Lower Level
- Blocks
- Cache
- Memory
- Disk
- Files
- Tape

Faster
Larger

Review: Address Mapping: Page Table

Virtual Address:
- page no.
- offset

Page Table
- Base Reg
- V
- A.R.
- P. P. A.
- Val
- Access Rights
- Physical Page Number
- index into page table

Physical Memory Address

Page Table located in physical memory

Some of the slides are adopted from David Patterson (UCB)
Paging/Virtual Memory for Multiple Processes

User A:
Virtual Memory

User B:
Virtual Memory

Address Map, Mathematically Speaking
V = {0, 1, ..., n - 1} virtual page address space (n > m)
M = {0, 1, ..., m - 1} physical page address space
MAP: V --> M \cup \{\theta\} page address mapping function
MAP(a) = a' if data at virtual address a is present in physical address a'
and a' = \theta if data at virtual address a is not present in M

Comparing the 2 Levels of Hierarchy
° Cache Version Virtual Memory vers.
° Block or Line Page
° Miss Page Fault
° Block Size: 32-64B Page Size: 4K-8KB
° Placement: Fully Associative
  Direct Mapped, N-way Set Associative
° Replacement: Least Recently Used
  LRU or Random (LRU)
° Write Thru or Back Write Back

Analogy
° Book title like virtual address (ARM System On-Chip)
° Library of Congress call number like (QA76.5.F8643 2000) physical address
° Card (or online-page) catalogue like page table, indicating mapping from book title to call number
° On card (or online-page) info for book, indicating in local library vs. in another branch like valid bit indicating in main memory vs. on disk
° On card (or online-page), available for 2-hour in library use (vs. 2-week checkout) like access rights
Notes on Page Table

- Solves Fragmentation problem: all chunks same size, so all holes can be used
- OS must reserve “Swap Space” on disk for each process
- To grow a process, ask Operating System
  - If unused pages, OS uses them first
  - If not, OS swaps some old pages to disk
  - (Least Recently Used to pick pages to swap)
- Each process has its own Page Table
- Will add details, but Page Table is essence of Virtual Memory

Virtual Memory Problem #1

- Not enough physical memory!
  - Only, say, 64 MB of physical memory
  - N processes, each 4GB of virtual memory!
  - Could have 64 virtual pages/physical page!
- Spatial Locality to the rescue
  - Each page is 4 KB, lots of nearby references
  - No matter how big program is, at any time only accessing a few pages
  - “Working Set”: recently used pages

Virtual Address and a Cache (#1/2)

- Cache operates on Virtual addresses.
- ARM Strategy
  - The advantage: If in cache the translation is not required.
  - Disadvantage: Several copies of the the same physical memory location may be present in several cache blocks. (Synonyms problem). Gives rise to some complications!

Virtual Address and a Cache (#2/2)

- Cache typically operates on physical addresses on most other systems.
- Address Translation (Page Table access) is another memory access for each program memory access!
- Accessing memory for Page Table to get Physical address (Slow Operation)
- Need to fix this!
Virtual Memory Problem #2

- Map every address ⇒ 1 extra memory accesses for every memory access
- Observation: since locality in pages of data, must be locality in virtual address of those pages
- Why not use a cache of virtual to physical address translations to make translation fast? (small is fast)
- For historical reasons, this cache is called a Translation Lookaside Buffer, or TLB

Typical TLB Format

<table>
<thead>
<tr>
<th>Virtual Address</th>
<th>Physical Address</th>
<th>Dirty</th>
<th>Ref</th>
<th>Valid</th>
<th>Access Rights</th>
</tr>
</thead>
</table>

- TLB just a cache on the page table mappings
- TLB access time comparable to cache (much less than main memory access time)
  - Ref: Used to help calculate LRU on replacement
  - Dirty: since use write back, need to know whether or not to write page to disk when replaced

PROJECTS IN DIGITAL HARDWARE DESIGN FOR 4th Year
Digital Systems Laboratory Hardware

DSL Board is
• A Board for 21st Century
• The State-of-the-Art Development Board

DSL Board Contains:
• Designed by the University of Manchester with lots of collaboration from UNSW
• An ARM Microcontroller
• With 2 MB of Flash and up to 4 MB of SRAM Memory
• 2 Xilinx FPGAs for extended interfacing and specialised co-processors
• Optional Ethernet chip
• LCD Module
• Lots of uncommitted Switches and LEDs
• Terminal connector to FPGAs

Two PCBs

DSLMU Hardware Block Diagram

Projects with Digital Systems Lab Board

° Project 1: Design of a Vector Floating Point Co-Processor for ARM Core:
° Aim: The Aim of this project is to build Floating Point Vector Processor On the Virtex FPGA.
° Degree of Difficulty: Hard, and Challenging
° Ability: Number Representation, Digital System Design, DSP

Projects with Digital Systems Board

° Project 2: Design of a Fixed Point DSP for ARM Core:
° Aim: The Aim of this project is to build DSP Optimised Processor (Single Cycle MAC Processor/Distributed Arithmetic) On the Virtex FPGA.
° Degree of Difficulty: Hard, and Challenging
° Ability: Number Representation, Digital System Design, DSP
Projects with Digital Systems Board

- **Project 3**: Development of a Simple Multi tasking/Threading Operating System
  - **Aim**: The Aim of this project is Development of a Simple Multi tasking Operating System with Simple Virtual Memory Protection for on-board program monitoring and debugging.
  - **Degree of Difficulty**: Moderate, and some Challenges
  - **Ability**: Software Development, Basic Operating Systems
  - **Application**: Our Undergraduate/Post Graduate Teaching
  - **Advantage**: Make yourself Immortal!

Projects with Digital Systems Board

- **Project 4**: Interfacing GNU debugging tool with on-board emulator (Komodo)
  - **Aim**: The Aim of this project is to Interface GNU debugging tool with on-board emulator program to facilitate source-level debugging and monitoring.
  - **Degree of Difficulty**: Moderate to Hard with some Challenges
  - **Ability**: Software Development, Basic Operating Systems
  - **Application**: Our Undergraduate/Post Graduate Teaching
  - **Advantage**: Make yourself Immortal!

Projects with Digital Systems Board

- **Project 5**: PS/2 and USB Controller For DSL Board
  - **Aim**: The aim of this project is to design an PS/2 and USB Controller using the on-board FPGA chips.
  - **Degree of Difficulty**: Hard, and Challenging
  - **Ability**: Hardware Development, Basic Software development

Projects with Digital Systems Board

- **Project 6**: Frame Grabber Device for CMOS Digital Camera
  - **Aim**: The aim of this project is to build a high resolution web camera using a Kodak KAC-1310 CMOS image sensor. The pixel data would buffered in SRAM/SDRAM, compressed and uploaded through the Internet interface for display on a PC.
  - **Degree of Difficulty**: Very Hard, and Challenging
  - **Ability**: DSP, Hardware Development, Basic Software development
**Projects with Digital Systems Board**

- **Project 7: Audio Signal Processing**
  - **Aim:** The aim of this project is to interface a codec to a DSP Audio Signal Processor/Compressor built on the on-Board FPGAs.
  - **Degree of Difficulty:** Moderate to Hard, and Challenging.
  - **Ability:** DSP, Hardware Development, Basic Software development.

- **Project 8: Location Aware Sound System**
  - **Aim:** The aim of this project is to build a wireless location detection system to provide optimum sound from stereo speakers depending on the location of the listener.
  - **Degree of Difficulty:** Moderate to Hard, and Challenging.
  - **Ability:** DSP, Hardware Development, Basic Software development.

---

**Things to Remember (#1/2)**

- Apply Principle of Locality Recursively.
- Reduce Miss Penalty? Add a (L2) cache.
- Manage memory to disk? Treat as cache.
  - Included protection as bonus, now critical.
  - Use Page Table of mappings vs. tag/data in cache.
- Virtual memory to Physical Memory Translation too slow?
  - Add a cache of Virtual to Physical Address Translations, called a TLB.

---

**Things to Remember (#2/2)**

- Virtual Memory allows protected sharing of memory between processes with less swapping to disk, less fragmentation than always swap or base/bound.
- Spatial Locality means Working Set of Pages is all that must be in memory for process to run fairly well.
Things to Remember

° Spatial Locality means Working Set of Pages is all that must be in memory for process to run fairly well

° Virtual memory to Physical Memory Translation too slow?
  • Add a cache of Virtual to Physical Address Translations, called a TLB
  • TLB to reduce performance cost of VM