Memory (Part II):
Mechanisms for Memory Management: Paging & Segmentation

Professor Travis Peters
CSCI 460 Operating Systems
Fall 2019

Some slides & figures adapted from Stallings instructor resources.
Some slides adapted from Adam Bates’s F’18 CS423 course @ UIUC
https://courses.engr.illinois.edu/cs423/sp2018/schedule.html
Goals for Today

Learning Objectives

• Understand basics of memory management, including
  • memory partitioning and common techniques
  • paging and segmentation — what they are, and their relative advantages and disadvantages
• Understand basics of loading and linking
Announcements

- Use Google Sheet to share info about your project:
  https://docs.google.com/spreadsheets/d/1uMk0pcho_B2v8_7t_E3S-lpsdUfhjBKsT5XdBcfPOBI/edit?usp=sharing
- PA2 posted later this week…
Relocation of Processes into Partitions

• Logical Address
  a reference to a memory location independent of the current assignment of data to memory → need translation

• Relative Address
  An example of a logical address. Address = relative location to some known point (e.g., value in register)

• Physical Address
  The actual location in main memory
Paging

Basic Idea
• Partition main memory into small fixed-sized chunks of the same size
• Assign chunks of processes (pages) into available chunks of main memory (frames)
• Small processes need fewer pages; larger processes need more pages
• No more external fragmentation
• Minimal internal fragmentation → only part of the last page of a process
Paging — Example: Assigning Process Pages to Free Frames

(a) Fifteen Available Frames
(b) Load Process A
(c) Load Process B
(d) Load Process C
(e) Swap out B
(f) Load Process D
**Paging — Logical Addressing**

HW assists with logical addressing when using paging — HW must know how to access page table

\( n + m \) bit addresses where
- \( n = \# \) bits for page number (leftmost bits)
- \( m = \# \) bits for offset within page (rightmost bits)
  \( \rightarrow \) PAGESIZE = \( 2^m \)

**NOTE:** In general, we set page/frame size to be a power of 2
\( \rightarrow \) relative address == logical address

**Example:**
- 16-bit addresses
- Page Size = 1K (1024 Bytes)

**Q:** How many bits are needed to accommodate pages/frames of size 1K?
\( \rightarrow \) 10 bits needed for offset field \( \rightarrow 1K = 2^{10} \)

**Q:** How many pages are possible with 6-bits available for page numbers?
\( \rightarrow \) 6 bits left over for page # field \( \rightarrow 64 \) pages

---

![Diagram](https://www.traviswpeters.com/cs460/)

**Figure 7.11   Logical Addresses**

- Relative address = 1502
  \( \overline{0000010111011110} \)

- Logical address = Page\# = 1, Offset = 478
  \( \overline{0000010111011110} \)

<table>
<thead>
<tr>
<th>Page</th>
<th>Offset</th>
<th>Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>750</td>
<td>750</td>
</tr>
<tr>
<td>1</td>
<td>1950</td>
<td>1950</td>
</tr>
<tr>
<td>2</td>
<td>478</td>
<td>internal fragmentation</td>
</tr>
</tbody>
</table>

**Example:**
- 16-bit addresses
- Page Size = 1K (1024 Bytes)

**Q:** How many bits are needed to accommodate pages/frames of size 1K?
\( \rightarrow \) 10 bits needed for offset field \( \rightarrow 1K = 2^{10} \)

**Q:** How many pages are possible with 6-bits available for page numbers?
\( \rightarrow \) 6 bits left over for page # field \( \rightarrow 64 \) pages

\( \rightarrow \)
Paging — Example of Logical-to-Physical Address Translation

Figure 7.12  Examples of Logical-to-Physical Address Translation
### Segmentation

#### Basic Idea

- Programs can be broken up into *segments* that need not be in contiguous memory; may occupy more than one segment
- Partition main memory into *unequally-sized segments*
  - Similar to dynamic partitioning... but not the same
- Assign segments of processes into chunks of main memory allocated on demand
- No internal fragmentation
- Potential for external fragmentation
Segmentation — Logical Addressing

- Each segment needs to provide
  - starting address of segment
  - segment length
- Load address of segment table into register when process starts running

- \( n + m \) bit addresses where
  - \( n \) = # bits for segment number (leftmost bits)
  - \( m \) = # bits for offset within segment (rightmost bits)

Example:
- 16-bit addresses
  - \( n = 4 \) bits
  - \( m = 12 \) bits

Q: What is the maximum size of a segment?
\( 4K = 2^{12} \)

Q: How many segments are possible?
\( 4 \) bits used for segment # → 16 pages

Logical address = Segment# = 1, Offset = 752
**Segmentation** — Example of Logical-to-Physical Address Translation

![Diagram](image-url)

- **16-bit logical address**: 0001001011100000
- **4-bit segment #**: 0
- **12-bit offset**: 011110011110
- **Process segment table**:
  - Length: 0010111011100000010000001000000100000
  - Base: 0010111011100000010000001000000100000

**16-bit physical address**: 00100011000000100000