

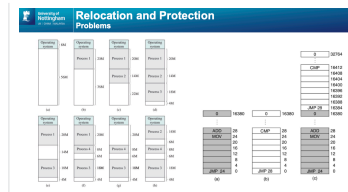
Paging

Code relocation and protection

Non-contiguous Approach : Paging

Why relocation (principle) 3

Relocation and Protection Principles

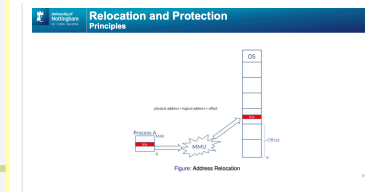


Relocation and Protection Principles

- **Relocation:** when a program is run, it does not know in advance which partition/address it will occupy
 - The program cannot easily generate static addresses (e.g. jump instructions) that are absolute
 - Addresses should be relative to where the program has been loaded
 - Relocation must be achieved in an operating system that allows processes to run at changing memory locations (on the fly)
- **Protection:** once you can have two programs in memory at the same time, protection must be enforced

What is relocation 5

Relocation and Protection Principles



Relocation and Protection Address Types

- A **logical address** is a memory address seen by the process.
 - It is independent of the current physical memory assignment
 - E.g. i.e., relative to the start of the program
- A **physical address** refers to an actual location in main memory
- The logical address space must be mapped onto the machine's physical address space

When to perform relocation 7

Relocation and Protection Principles

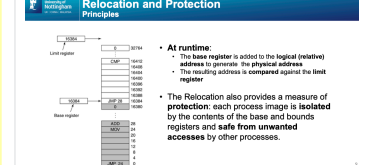
- **Static "relocation"** at compile time: a process has to be located at the same location every single time (impractical)
- **Dynamic relocation at load time**
 - An offset is added to every logical address to account for its physical location in memory
 - Slows down the loading of a process, does not account for swapping
- **Dynamic relocation at runtime**

How to perform relocation 8

Relocation and Protection Address Types

- Two special purpose registers are maintained in the CPU (the MMU), containing a **base address** and **limit**
 - The base register stores the start address of the partition
 - The limit register holds the size of the partition (end of program)
- This approach requires **hardware support** (was not always present in the early days)

Relocation and Protection Principles



- **At runtime:**
 - The base register is added to the logical (relative) address to generate the physical address
 - The resulting address is compared against the limit register
- The Relocation also provides a measure of **protection**: each process image is **isolated** by the contents of the base and bounds registers and **safe from unwanted accesses** by other processes.

Paging

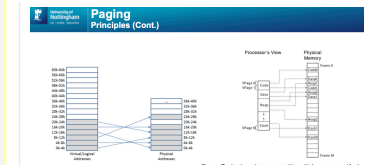
What is paging 12

Paging Principles

- **Paging** uses the principles of **fixed partitioning** and **code re-location** to devise a new **non-contiguous management scheme**
 - Memory is split into small **master blocks** and one or **multiple blocks** are allocated to a process
 - E.g. i.e. 110 process would take up 3 blocks of 4 kb
 - These blocks do not have to be contiguous in main memory, but the process still perceives them to be contiguous
- Benefits compared to contiguous schemes include:
 - **Internal fragmentation** is reduced to the last "block" only
 - There is **no external fragmentation**, since physical blocks are **stacked directly onto each other** in main memory

Paging demo 13

Paging Principles (Cont.)



Paging Principles (Cont.)

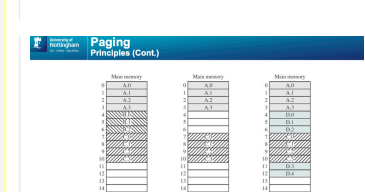


Figure: Concept of Paging (Shallings)

Concepts used in paging 15

Paging Principles Definitions

- A **page** is a small block of contiguous memory in the logical address space, i.e. as seen by the process
- A **page frame** is a small contiguous block in physical memory
- Pages and frames (usually) have the same size:
 - The size is usually a power of 2
 - Sizes range between 512 bytes and 1Gb

Page table

Address translation using page table => implementation 21

Paging Address Translation: Implementation

- A **logical (physical) address** is relative to the start of the program (memory) and consists of two parts:
 - The **right most n bits** that represent the offset within the page (frame)
 - E.g. within the offset, dividing up to start of 10 bytes per page (frame)
 - The **left most n bits** that represent the **page (frame) number**
 - E.g. within the page number, dividing up to start of 10 bytes per page (frame)
- The offset within the page and frame remains the same (they are the same size)
- The page number to frame number mapping is held in the **page table**

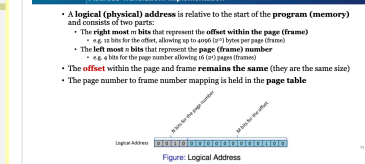


Figure: Logical Address

Address translation: example 22

Paging Address Translation: Implementation

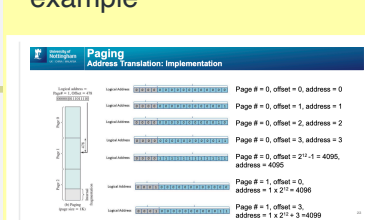


Figure: Address Translation

Address translation: Model 23

Paging Address Translation: Implementation

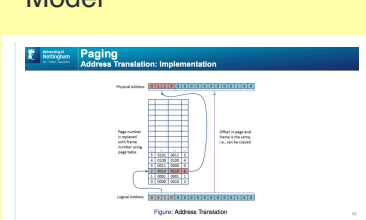


Figure: Address Translation

Address translation: Steps 24

Paging Address Translation

- **Steps in address translation:**
 1. Extract the **page number** from logical address
 2. Use page number as an index to retrieve the frame number in the page table
 3. Add the "logical offset within the page" to the start of the physical frame
- **Hardware implementation of address translation:**
 - 1. The CPU's memory management unit (MMU) interprets logical addresses
 - 2. MMU uses a page table as above
 - 3. The resulting physical address is put on the **memory bus**