CHANDIGARH UNIVERSITY

Discover. Learn. Empower.

UNIVERSITY INSTITUTEOF ENGINEERING

Bachelor of Engineering (Computer Science & Engineering)

Operating System (CST-328)

Subject Coordinator: Er. Puneet kaur (E6913)

Chapter 4

(Memory Management)

Memory Management: Address binding, logical versus physical address space, dynamic loading, Swapping, contiguous memory allocation, Fragmentation, Paging, Segmentation, Segmentation with Paging, Virtual Memory Concept, Demand Paging, Page Replacement, Page Replacement Algorithms.

Address Binding

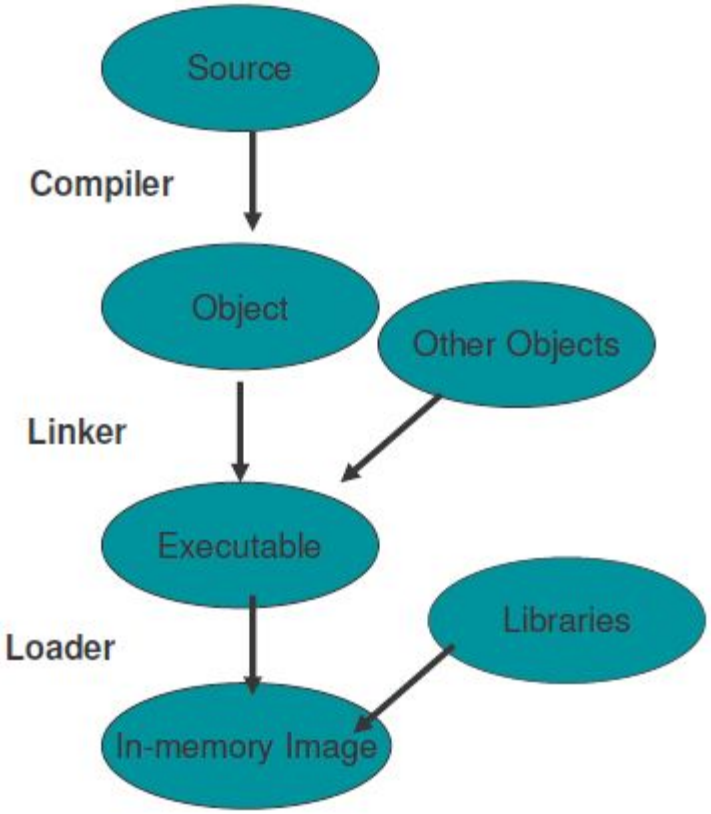
Address binding is the process of mapping the program's logical or virtual **addresses** to corresponding physical or main memory **addresses**

Compile time. The compiler translates symbolic addresses to absolute addresses. If you know at compile time where the process will reside in memory, then absolute code can be generated (Static).

Load time. The compiler translates symbolic addresses to relative (relocatable) addresses. The loader translates these to absolute addresses. If it is not known at compile time where the process will reside in memory, then the compiler must generate relocatable code (Static).

Relocatable means that the program image can reside anywhere in physical memory.

Execution time. If the process can be moved during its execution from one memory segment to another, then binding must be delayed until run time. The absolute addresses are generated by hardware. Most general-purpose OSs use this method (Dynamic).


Source
Compiler
Object
Other Objects
Linker
Executable
Libraries
Loader
In-memory Image

Memory Management

Main Memory refers to a physical memory that is the internal memory to the computer. The word main is used to distinguish it from external mass storage devices such as disk drives. Main memory is also known as RAM.

Memory Management

- At times one program is dependent on some other program. In such a case, rather than loading all the dependent programs, CPU links the dependent programs to the main executing program when its required. This mechanism is known as **Dynamic Linking**.

- Linking postponed until execution time.
- Small piece of code, stub, used to locate the appropriate memory-resident library routine.
- Stub replaces itself with the address of the routine, and executes the routine.
- Operating system needed to check if routine is in processes' memory address.
- Dynamic linking is particularly useful for libraries.

Memory Management

- All the programs are loaded in the main memory for execution. Sometimes complete program is loaded into the memory, but some times a certain part or routine of the program is loaded into the main memory only when it is called by the program, this mechanism is called **Dynamic Loading**, this enhance the performance.

  - Routine is not loaded until it is called
  - Better memory-space utilization; unused routine is never loaded.
  - Useful when large amounts of code are needed to handle infrequently occurring cases.
  - No special support from the OS is required - implemented through program design.

Process Address Space

- The process address space is the set of logical addresses that a process references in its code. For example, when 32-bit addressing is in use, addresses can range from 0 to 0x7fffffff; that is, 2^{31} possible numbers, for a total theoretical size of 2 gigabytes.

- The operating system takes care of mapping the logical addresses to physical addresses at the time of memory allocation to the program.

Process Address Space

There are three types of addresses used in a program before and after memory is allocated –

Symbolic addresses

The addresses used in a source code. The variable names, constants, and instruction labels are the basic elements of the symbolic address space.

Relative addresses

At the time of compilation, a compiler converts symbolic addresses into relative addresses.

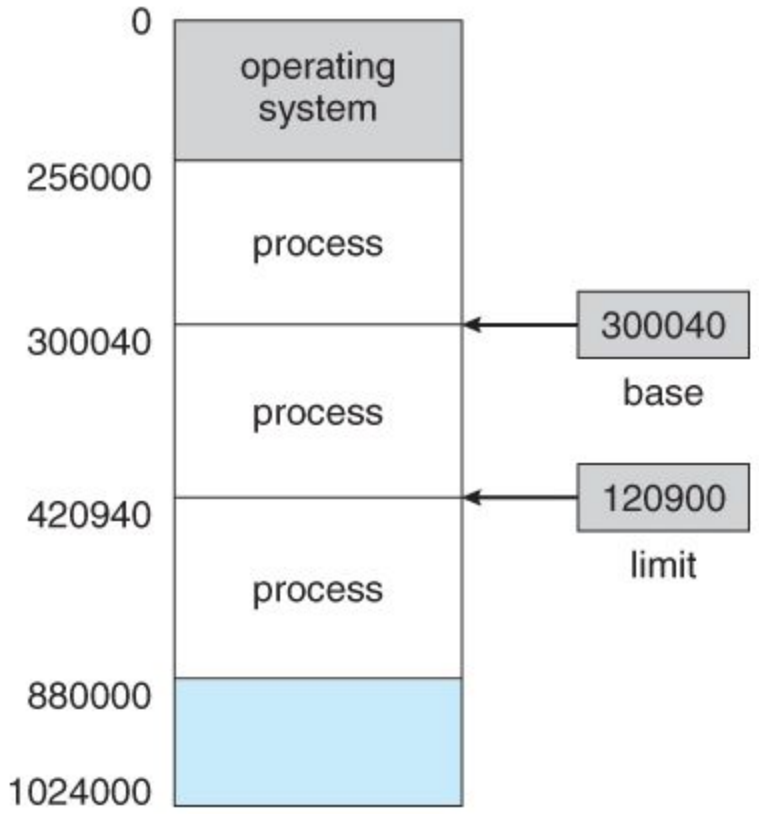
Physical addresses

The loader generates these addresses at the time when a program is loaded into main memory.

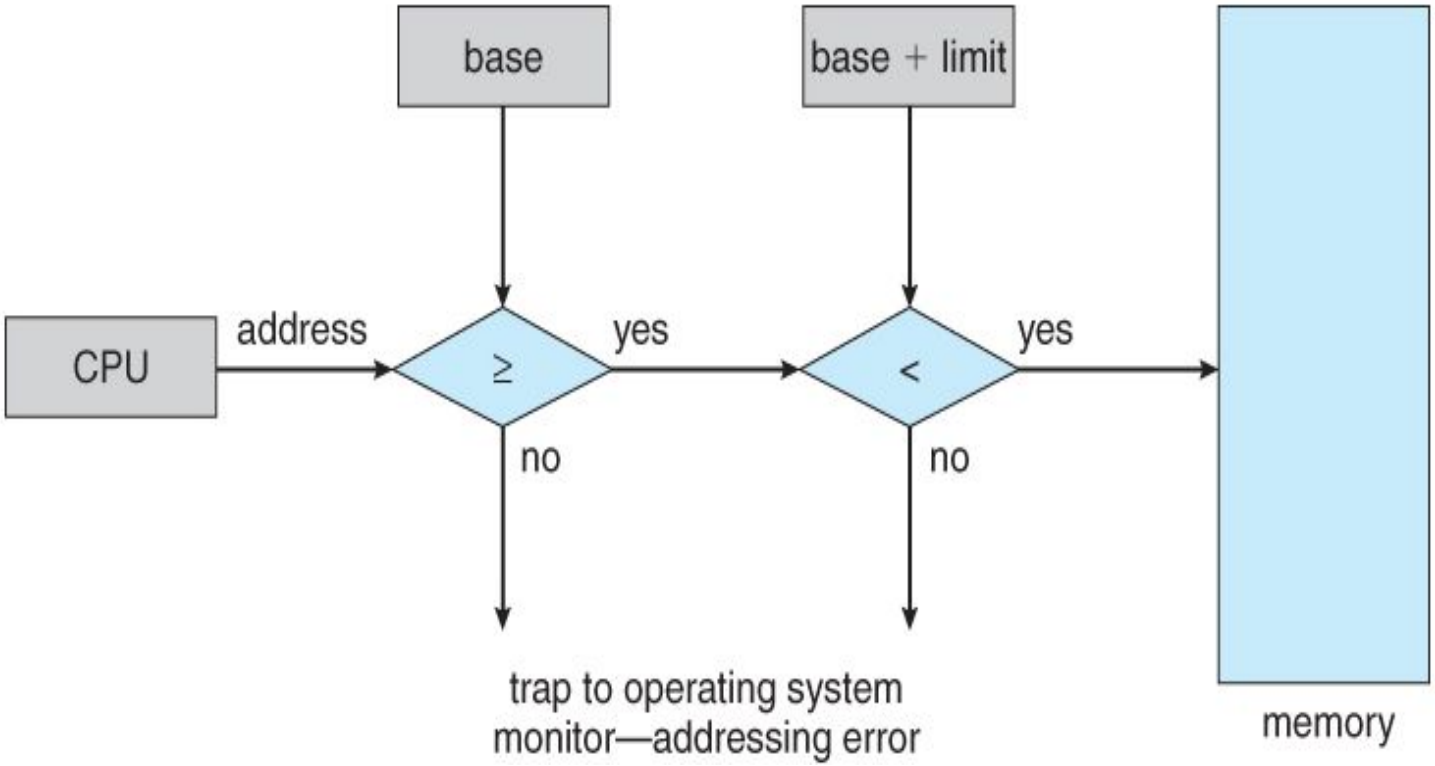
Note: Virtual and physical addresses are the same in compile-time and load-time address-binding schemes.

Base and Limit Registers

- A pair of **base** and **limit registers** define the logical address space
- CPU must check every memory access generated in user mode to be sure it is between base and limit for that user



Hardware Address Protection



Process Address Space

- **Logical address** – generated by the CPU; also referred to as **virtual address**
- **Physical address** – address seen by the memory unit

- Logical and physical addresses are the same in compile-time and load-time address-binding schemes; logical (virtual) and physical addresses differ in execution-time address-binding scheme

The set of all logical addresses generated by a program is referred to as a **logical address space.**

The set of all physical addresses corresponding to these logical addresses is referred to as a **physical address space.**

Physical Address Vs Physical Address Space

- Physical address space in a system can be defined as the size of the main memory. It is really important to compare the process size with the physical address space. The process size must be less than the physical address space.

- Physical Address Space = Size of the Main Memory
  If, physical address space = 64 KB = 2 ^ 6 KB = 2 ^ 6 X 2 ^ 10 Bytes = 2 ^ 16 bytes

  Let us consider,
  word size = 8 Bytes = 2 ^ 3 Bytes

  Hence,
  Physical address space (in words) = (2 ^ 16) / (2 ^ 3) = 2 ^ 13 Words

  Therefore,
  Physical Address = 13 bits

  In General,
  If, Physical Address Space = N Words

  then, Physical Address = $\log_2 N$

Logical Address vs Logical Address Space

- Logical address space can be defined as the size of the process. The size of the process should be less enough so that it can reside in the main memory.
- Logical Address Space = 128 MB = (2 ^ 7 X 2 ^ 20) Bytes = 2 ^ 27 Bytes
  Word size = 4 Bytes = 2 ^ 2 Bytes

  Logical Address Space (in words) = (2 ^ 27) / (2 ^ 2) = 2 ^ 25 Words
  Logical Address = 25 Bits

  In general,
  If, logical address space = L words
  Then, Logical Address = Log_2L bits

Conclusion

This lecture makes the student familiar with Memory management topics like logical memory, physical memory, virtual memory, dynamic linking, dynamic loading and process address space etc.

Video Links

https://www.youtube.com/watch?v=DdUeTN0qfuE

https://www.youtube.com/watch?v=Rnfu5qyysro

References

https://www.tutorialspoint.com/operating_system/os_memory_management.htm

https://www.studytonight.com/operating-system/memory-management

https://www.guru99.com/os-memory-management.html

https://www.geeksforgeeks.org/partition-allocation-methods-in-memory-management/

https://www.javatpoint.com/os-memory-management-introduction

http://www2.latech.edu/~box/os/ch08.pdf

https://www.cs.uic.edu/~jbell/CourseNotes/OperatingSystems/8_MainMemory.html#:~:text=8.3%20Contiguous%20Memory%20Allocation,allocated%20to%20processes%20as%20needed.

http://www.csdl.tamu.edu/~furuta/courses/99a_410/slides/chap08