Text, logo

Description automatically generated **San Francisco Bay University**

**CE521 - Real-time Systems and Programming**

**Homework Assignment #6**

**Due day: 4/17/2022**

**Instructions:**

1. **Push the answer sheet to Github**
2. **Overdue homework submission could not be accepted.**
3. **Takes academic honesty and integrity seriously (Zero Tolerance of Cheating & Plagiarism)**
4. Explain the concepts in the steps of the processing of programs: compile time/load time/execution time.

Answer: A program is a piece of code which may be a single line or millions of lines. A computer program is usually written by a computer programmer in a programming language. A computer program is a collection of instructions that performs a specific task when executed by a computer. When we compare a program with a process, we can conclude that a process is a dynamic instance of a computer program. A Program, in simple words, can be considered as a system activity. In batch processing system these are called executing jobs while in a real-time operating system it is called tasks or programs. A user can run multiple programs where the operating system facilitates its own internal programmed activities such as memory management using some techniques. A program is a passive entity, for example, a file accommodating a group of instructions to be executed (executable file). It is so called because it does not perform any action by itself, it has to be executed to realize the actions specified in it. The address space of a program is composed of the instruction, data and stack. Assume P is the program we are writing, to realize execution of P, the operating system allocates memory to accommodate P’s address space. A part of a computer program that performs a well-defined task is known as an algorithm. A collection of computer programs, libraries and related data are referred to as a software. Classically, the binding of instructions and data to memory addresses can be done at any step along the way:

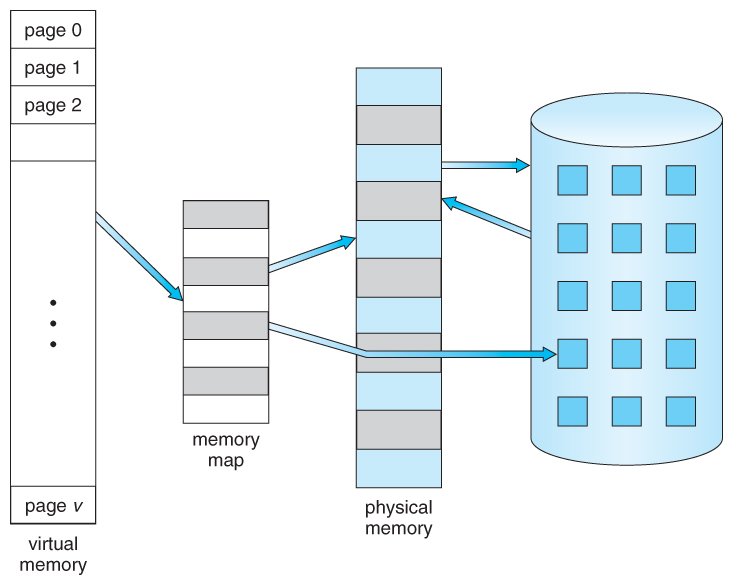
* **Compile time**. The compiler translates symbolic addresses to absolute addresses. If you know at compile time where the process will reside in memory, then absolute code can be generated (Static). If you know that during compile time, where process will reside in memory, then an absolute address is generated. i.e the physical address is embedded to the executable of the program during compilation. Loading the executable as a process in memory is very fast. But if the generated address space is preoccupied by other processes, then the program crashes and it becomes necessary to recompile the program to change the address space.
* **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). If it is not known at the compile time where the process will reside, then a relocatable address will be generated. The loader translates the relocatable address to an absolute address. The base address of the process in main memory is added to all logical addresses by the loader to generate an absolute address. In this, if the base address of the process changes, then we need to reload the process again.
* **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). The instructions are in memory and are being processed by the CPU. Additional memory may be allocated and/or deallocated at this time. This is used if a process can be moved from one memory to another during execution(dynamic linking-Linking that is done during load or run time). e.g. – Compaction

1. Explain why the virtual memory is commonly built in the modern computing system

Answer :

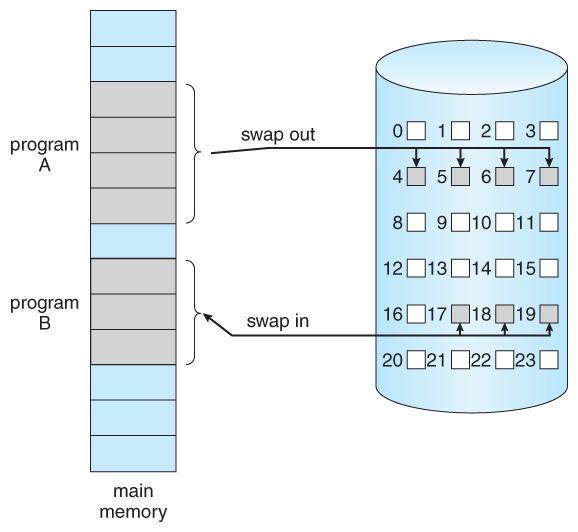
**Virtual memory is implemented in an operating system with the help of Demand Paging or Segmentation. It is nothing but the method of loading the page into memory on demand that is whenever a page fault occurs.**

**VM is nothing but an extra memory that a computer requires rather than the amount physically installed on the system.** Virtual-memory is a feature of an operating system it will enables a computer and transferring pages of data from random access memory to disk storage, but it is not faster than ram, it will serve as two purposes.



First one, it allows us to extend the use of physical memory by using disk. Second one, it allows us to have memory protection and each virtual address is translated to a physical address.it has two types to handled, first paged second segmented, the paging divide memory into sections and segment will not divide memory into sections.it is a storage mechanism which offers user an illusion of having a very big main memory, it does not work without RAM, it will make a computer run slower and the process takes much time. It is also known as virtual storage and is a memory management technique virtual memory executes the big size process.

The virtual memory used on most modern operating systems because the memory has no fixed physical address and can be moved between locations and even swapped to disk. Virtual Memory is a storage allocation scheme in which secondary memory can be addressed as though it were part of main memory, its size is limited by the addressing scheme of the computer system, it is a technique it will implemented using both hardware and software, all memory references within a process are logical addresses that are dynamically translated into physical addresses at run time, the process may be broken into many number of pieces and these pieces are not useful to located in the main memory during execution, it is a feature of operating system, Operating system brings into main memory a few pieces of the program.



In the virtual memory ,we provide required pages to the main memory through swap in and swap out ,we use the virtual memory in operating system, the CPU provides the process first and also demands the data if in case the demanded data not present in main memory is known as data fault or page fault and the demanded data present in main memory the memory management will give this data to CPU, the user handles the whole process and in the time of CPU demands the data that will not present in main memory it goes to the trapping and operating system comes to main memory and do the authentication for security purpose, the data will read in hard disk therefore the operating system goes to hard disk or logical address space first and search for required process after successful searching it will send that data or process to main memory in an empty place after that updates in memory management and finally CPU gets the demanded data ,like this the virtual memory helps to find the data and completes the user or CPU demands. **When the memory is larger than the process then there is a chance for internal fragmentation to occur. So, the process will be larger than the main memory of the computer system.**

1. Describe how many methods can be taken to map virtual address to physical address, and compare the pros/cons for each.

Answer-

To store the data and to manage the processes, we need a large-sized memory and, at the same time, we need to access the data as fast as possible. But if we increase the size of memory, the access time will also increase and, as we know, the CPU always generates addresses for secondary memory, i.e. logical addresses. But we want to access the main memory, so we need Address translation of logical address into physical address.  
The main memory interacts with both the user processes and the operating system. So we need to efficiently use the main memory. Main memory is divided into non-overlapping memory regions called partitions.

The main memory can be broadly allocated in two ways –

1. Contiguous memory allocation

2.Non-Contiguous memory allocation

Contiguous memory allocation can be categorized into two ways:

1.Fixed partition scheme

2.Variable partition scheme.

Different Partition Allocation methods are used in Contiguous memory allocations –

* First Fit
* Best Fit
* Worst Fit
* Next Fit

Non-Contiguous memory allocation can be categorized into many ways:

* Base and Bound
* Paging
* Multilevel paging
* Inverted paging
* Segmented paging

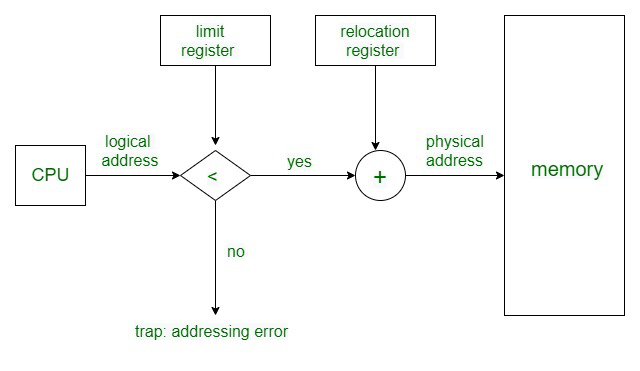
Mapping Virtual Addresses to Physical Addresses: In Contiguous memory allocation mapping from virtual addresses to physical addresses is not a difficult task, because if we take a process from secondary memory and copy it to the main memory, the addresses will be stored in a contiguous manner, so if we know the base address of the process, we can find out the next addresses.  In non-contiguous memory allocation, different parts of a process are allocated different places in Main Memory. in Non-Contiguous allocation, process can be divided into different parts and hence filling the space in main memory.

* Base and bound –

Base Register (Relocation Register)

Bound or Limit Register.

Base Register – contains the starting physical address of the process.  
Limit Register -mentions the limit relative to the base address on the region occupied by the process.

The logical address generated by the CPU is first checked by the limit register, If the value of the logical address generated is less than the value of the limit register, the base address stored in the relocation register is added to the logical address to get the physical address of the memory location.  
If the logical address value is greater than the limit register, then the CPU traps to the OS, and the OS terminates the program by giving fatal error.

In Non-Contiguous Memory allocation, processes can be allocated anywhere in available space. The address translation in non-contiguous memory allocation is difficult.  
There are several techniques used for address translation in noncontiguous memory allocation like Paging, Multilevel paging, Inverted paging, Segmentation, Segmented paging. Different data structures and hardware support like TLB are required in these techniques.

Pro and cons of base and bound-

Pros-  
• OS protection and program isolation  
• Low overhead address translation  
Cons-  
• Expandable heap Issue   
• Expandable stack Issue  
• Memory sharing between processes -overwriting  
• non-relative addresses – hard to move memory around  
• Memory fragmentation

* Paging –

Paging is a memory management scheme that eliminates the need for contiguous allocation of physical memory. This scheme permits the physical address space of a process to be non – contiguous.

Logical Address or Virtual Address (represented in bits): An address generated by the CPU

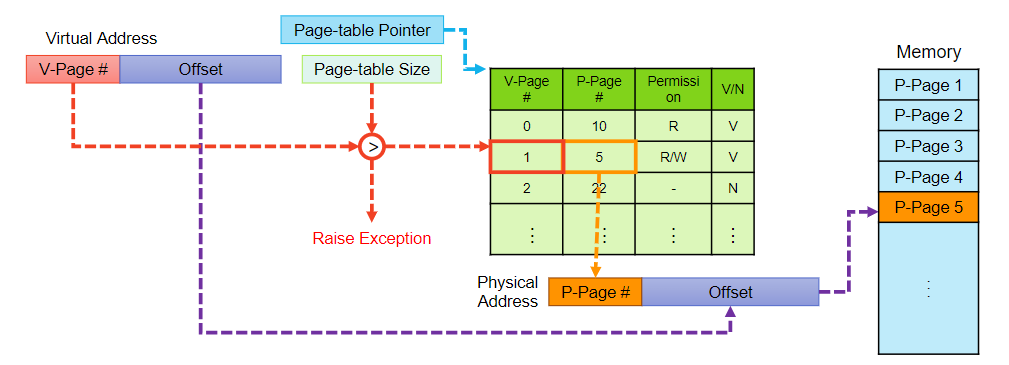
Logical Address Space or Virtual Address Space (represented in words or bytes): The set of all logical addresses generated by a program

Physical Address (represented in bits): An address actually available on memory unit

Physical Address Space (represented in words or bytes): The set of all physical addresses corresponding to the logical addresses

Pros and cons of paging –

* Paging reduces external fragmentation, but still suffer from internal fragmentation.
* Paging is simple to implement and assumed as an efficient memory management technique.
* Due to equal size of the pages and frames, swapping becomes very easy.
* Page table requires extra memory space, so may not be good for a system having small RAM.
* Page table –

A data structure called page map table is used to keep track of the relation between a page of a process to a frame in physical memory. Page table has page table entries where each page table entry stores a frame number and optional status (like protection) bits. Many of status bits used in the virtual memory system. The most important thing in PTE is frame Number. When the system allocates a frame to any page, it translates this logical address into a physical address and create entry into the page table to be used throughout execution of the program.

When a process is to be executed, its corresponding pages are loaded into any available memory frames.

Pros and Cons-

Pros  
• Simple memory allocation  
• Easy to share  
Cons  
• Inefficient for sparse address spaces  
• There are too many unused page-table entries  
• What if page size is very small?  
• With 1KiB pages, we need 222 (~4 million) table entries!  
• What if page size is too big?  
• Wastes space inside of page (internal fragmentation)

* Segmentation-

Segmentation is a memory management technique in which each job is divided into several segments of different sizes, one for each module that contains pieces that perform related functions. Each segment is actually a different logical address space of the program.

When a process is to be executed, its corresponding segmentation are loaded into non-contiguous memory though every segment is loaded into a contiguous block of available memory.

Segmentation memory management works very similar to paging but here segments are of variable-length where as in paging pages are of fixed size.

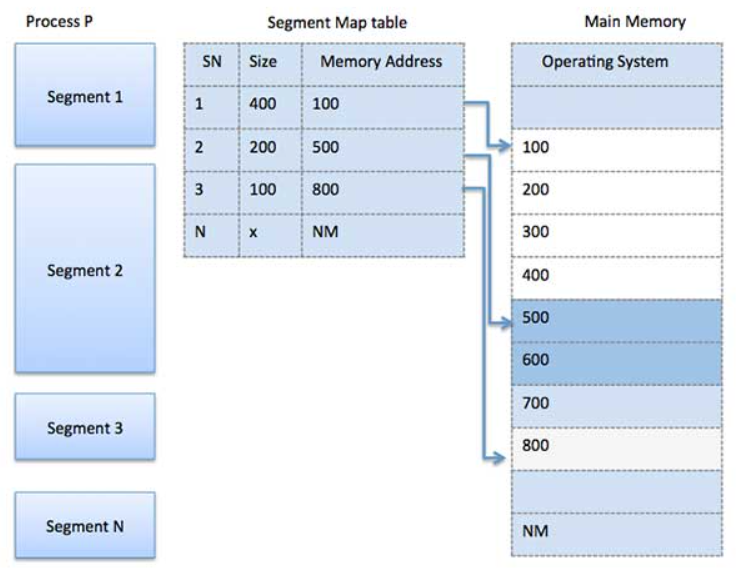
A program segment contains the program's main function, utility functions, data structures, and so on. The operating system maintains a segment map table for every process and a list of free memory blocks along with segment numbers, their size and corresponding memory locations in main memory. For each segment, the table stores the starting address of the segment and the length of the segment. A reference to a memory location includes a value that identifies a segment and an offset.

Pro and cons of segmentation:

Pros of Segmentation –

* No Internal fragmentation.
* Segment Table consumes less space in comparison to Page table in paging.

Cons of Segmentation –

* As processes are loaded and removed from the memory, the free memory space is broken into little pieces, causing External fragmentation.

Inverted Page table-

Most of the Operating Systems implement a separate pagetable for each process, i.e. for ‘n’ number of processes running on a Multiprocessing/ Timesharing operating system, there are ‘n’ number of pagetables stored in the memory. Sometimes when a process is very large in size and it occupies virtual memory then with the size of the process, it’s pagetable size also increases substantially. An alternate approach is to use the Inverted Page Table structure that consists of one-page table entry for every frame of the main memory. So the number of page table entries in the Inverted Page Table reduces to the number of frames in physical memory and a single page table is used to represent the paging information of all the processes.

Through the inverted page table, the overhead of storing an individual page table for every process gets eliminated and only a fixed portion of memory is required to store the paging information of all the processes together. This technique is called as inverted paging as the indexing is done with respect to the frame number instead of the logical page number. Each entry in the page table contains the following fields.

Pros and cons:

* Reduced memory space –Inverted Pagetables typically reduces the amount of memory required to store the page tables to a size bound of physical memory. The maximum number of entries could be the number of page frames in the physical memory.
* Longer lookup time –Inverted Page tables are sorted in order of frame number but the memory look-up takes place with respect to the virtual address, so, it usually takes a longer time to find the appropriate entry but often these page tables are implemented using hash data structures for a faster lookup.
* Difficult shared memory implementation –As the Inverted Page Table stores a single entry for each frame, it becomes difficult to implement the shared memory in the page tables. Chaining techniques are used to map more than one virtual address to the entry specified in order of frame number.