RV College of Engineering

Title: PAGED SEGMENTATION

VARSHA V P(1RV22CS225)
SRAVYA D(1RV22CS202)

Department of Computer Science and Engineering
RV College of Engineering, Bangalore - 560059, INDIA

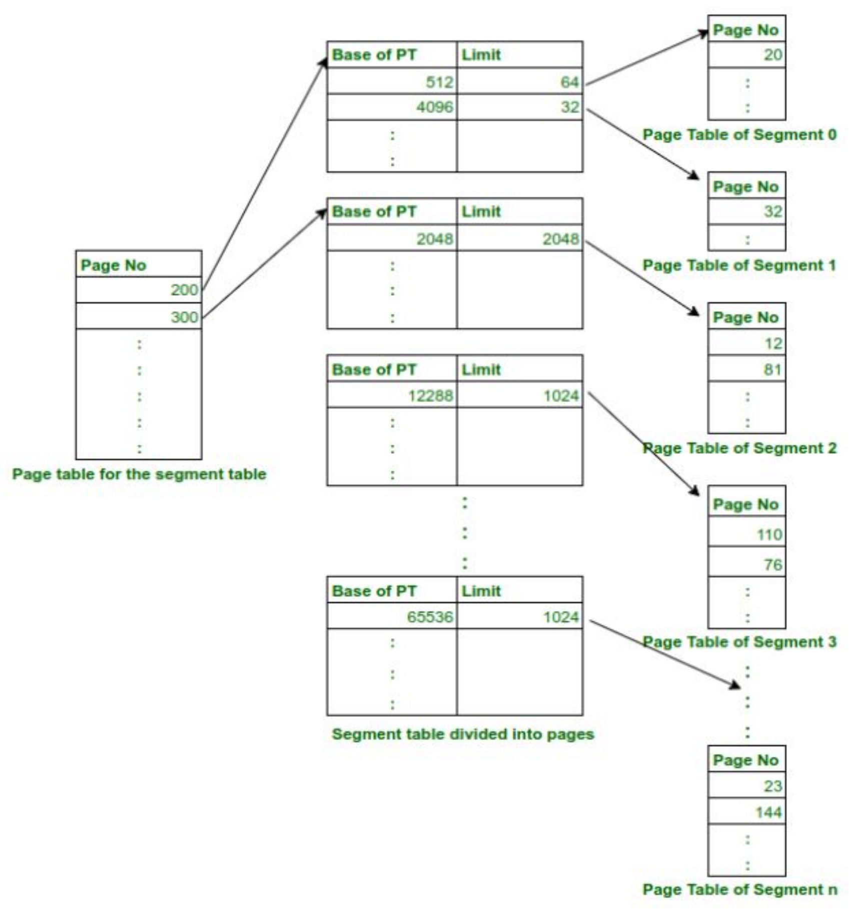
Go, change the world

Introduction

Paged segmentation is a memory management technique employed by operating systems to organize and allocate memory in a structured manner. In paged segmentation, the main memory is divided into fixed-size units called pages, typically smaller than the size of the entire program or process. Similarly, the logical address space of a process is divided into fixed-size units known as logical pages. This segmentation allows for efficient allocation and management of memory resources.

System Architecture

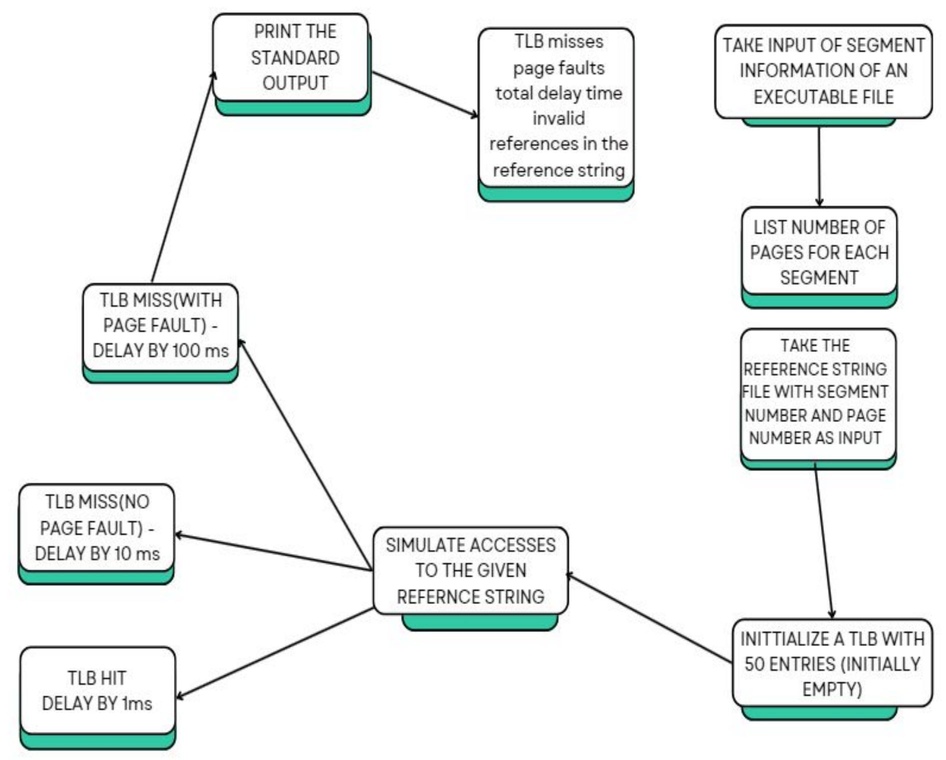
In paged segmentation architecture, the Memory Management Unit (MMU) translates CPU-generated logical addresses to physical addresses by consulting the process's page table. Entries in the page table map logical to physical pages, facilitating efficient address translation. If a page is present in main memory, the MMU translates the logical address. Otherwise, a page fault triggers the operating system to fetch the required page from secondary storage, a process called paging. Once loaded, the page table updates, enabling CPU access. Paged segmentation optimizes memory usage, dynamically allocating resources while minimizing overhead, enhancing system efficiency.



Methodology

The report presents a methodology for simulating Paging with Segmentation, a crucial memory management technique in operating systems. Initially, the program ingests segment details from an executable file, outlining segment sizes divided into fixed-size pages. Subsequently, it compiles a table displaying page counts per segment based on this information.

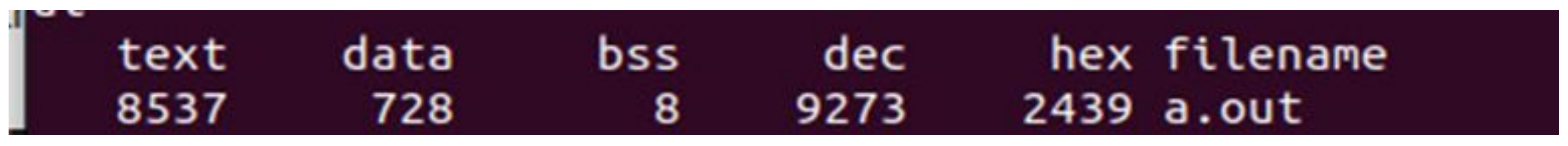
The simulation proceeds by accessing pages via a reference string file containing segment and page numbers. The program monitors TLB hits and misses, with a 1 millisecond delay for hits and 10 milliseconds for misses without page faults. Page faults trigger a 100-millisecond delay as necessary pages are fetched from virtual memory. TLB and virtual memory management follow specified policies, employing FIFO and LRU algorithms, respectively.


PRINT THE STANDARD OUTPUT
TLB misses page faults total delay time invalid references in the reference string
TAKE INPUT OF SEGMENT INFORMATION OF AN EXECUTABLE FILE
LIST NUMBER OF PAGES FOR EACH SEGMENT
TAKE THE REFERENCE STRING FILE WITH SEGMENT NUMBER AND PAGE NUMBER AS INPUT
INITTIALIZE A TLB WITH 50 ENTRIES (INITIALLY EMPTY)
SIMULATE ACCESSES TO THE GIVEN REFERNCE STRING
TLB MISS(WITH PAGE FAULT) - DELAY BY 100 ms
TLB MISS(NO PAGE FAULT) - DELAY BY 10 ms
TLB HIT DELAY BY 1ms

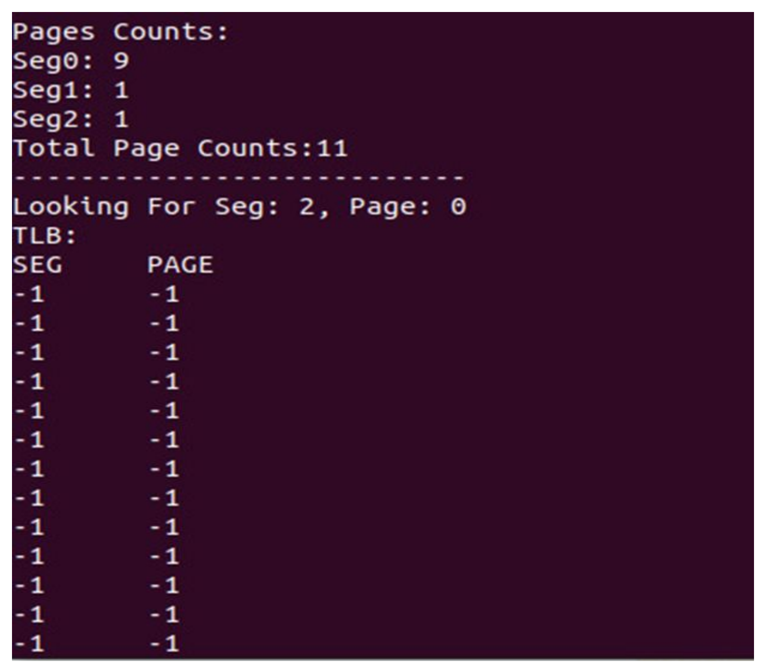
Performance metrics such as TLB misses, page faults, and total delay time are analyzed throughout the simulation. Results, including the count of invalid references encountered, are summarized at the report's conclusion. This approach offers a holistic understanding of Paging with Segmentation, shedding light on memory management strategies and their implications for system performance.

Output

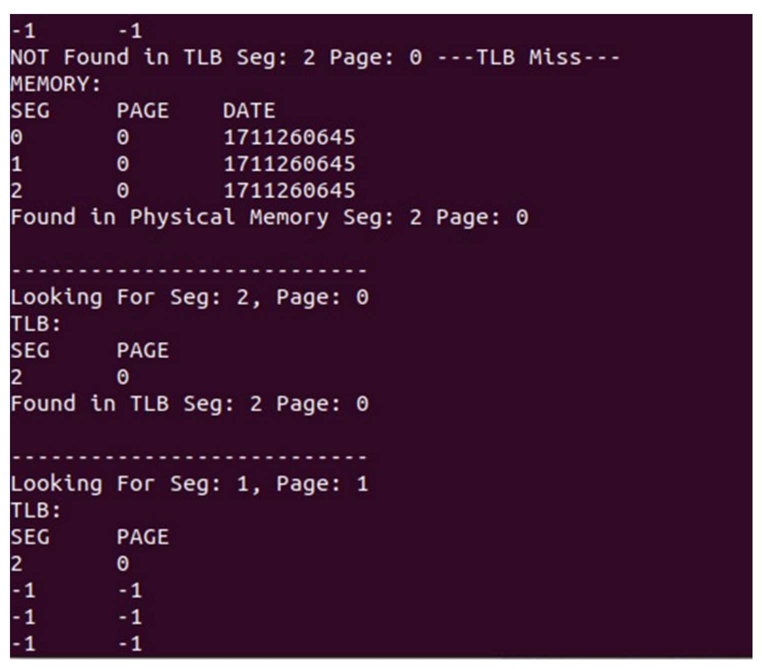
1. The number and ratio of TLB misses.
2. The number and ratio of page faults.
3. The total delay time.
4. There may be invalid references in the reference string (accessing a non-existing segment or a non-existing page of an existing segment). Treat such references as if they don't exist but print the number of invalid references at the end.
5. The first entry of the segment table is in the physical memory initially, keeping all the other entries in the virtual memory. Whenever a particular page is accessed, it is brought into the physical memory. This ensures efficient memory usage.
6. Integrating paging and Segmentaion helps in eliminating any kind of fragmentation and hence memory wastage.



```
   text    data     bss     dec     hex filename
   8537     728       8    9273    2439 a.out
```



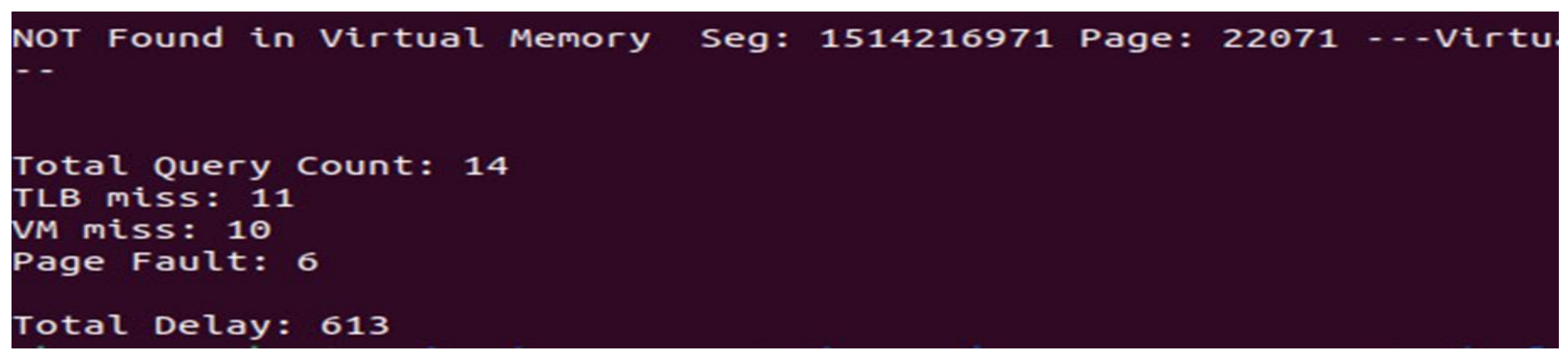
```
Pages Counts:
Seg0: 9
Seg1: 1
Seg2: 1
Total Page Counts:11
----------------------------
Looking For Seg: 2, Page: 0
TLB:
SEG     PAGE
-1      -1
-1      -1
-1      -1
-1      -1
-1      -1
-1      -1
-1      -1
-1      -1
-1      -1
-1      -1
-1      -1
-1      -1
-1      -1
```



```
-1      -1
NOT Found in TLB Seg: 2 Page: 0 ---TLB Miss---
MEMORY:
SEG     PAGE    DATE
0       0       1711260645
1       0       1711260645
2       0       1711260645
Found in Physical Memory Seg: 2 Page: 0

---------------------------
Looking For Seg: 2, Page: 0
TLB:
SEG     PAGE
2       0
Found in TLB Seg: 2 Page: 0

---------------------------
Looking For Seg: 1, Page: 1
TLB:
SEG     PAGE
2       0
-1      -1
-1      -1
-1      -1
```



```
NOT Found in Virtual Memory  Seg: 1514216971 Page: 22071 ---Virtu
--


Total Query Count: 14
TLB miss: 11
VM miss: 10
Page Fault: 6

Total Delay: 613
```

Applications

1. Operating Systems
2. Virtualization
3. Programming Languages
4. Graphics Processing
5. Networks Protocol

Advantages

1. It offers protection in specific segments
2. It uses less memory than paging
3. There is no external fragmentation
4. Easiness in adding or removing segments and pages
5. It also facilitates multi-programming

Conclusion:

This project explored memory management techniques: paging, segmentation, single/multi-level paging, and segmented paging. To overcome limitations of the traditional approaches, we proposed paged segmentation, combining paging and segmentation benefits. Paged segmentation divides logical memory into segments, further subdivided into pages, addressing fragmentation concerns. This hybrid approach optimizes memory utilization, address space management, and system performance. Through experiments, simulations, and analyses, we compared paged segmentation with traditional techniques. It combines the benefits of paging and segmentation while minimizing their drawbacks, paving the way for advancements in operating systems.

Acknowledgements

We express our sincere gratitude to the Principal,Dean,HOD and teacher for their kind support received for the completion of the project.

Faculty Guide:
Dr.Jyoti Shetty
Department of Computer Science and Engineering