Virtual Memory Manager

A program that translates logical addresses to physical addresses.

By:

Stanley Lalanne
Nusrath Ahmed
Rasheed A.
Gregory Ayo

Some important info:

TLB_SIZE = 16
PAGES = 256
PAGE MASK = 255
PAGE_SIZE = 256
OFFSET_MASK = 255
MEMORY_SIZE = PAGES * PAGE_SIZE = 65,536

THESE SPECIFICS ARE GIVEN IN THE PROJECT HANDOUT

HOW THE PROGRAM IS SETUP

Struct tlb_entry:

Holds the logical addresses and the physical addresses

An array of structs type tlb_entry named “tlb” has the size of 16 which is specified by the project.

A pagetable array to hold the different pages

pagetable[pagenumber] holds the page number for the physical addresses. Its values are -1 if the page for logical addresses isn't yet in the table.

A TLBsearch Function:

-Takes a logical address page number as argument

-Searches the tlb array

- Return the physical address if present in tlb
- Return -1 if not present

A tlbInsert Function:

-Inserts the logical and physical addresses in the tlb array after they have been determined

MAIN PROGRAM

Program requires two command line arguments:

1) the BACKING_STORE binaries
2) The address.txt file

Program does not continue running if arguments are not provided.

How OFFSET is determined

The logical addresses are given in the address.txt file

BITWISE AND(&) OPERATION

```
int offset = logical_addresses & OFFSET_MASK;
```

Ex:

5 & 3 = 1

5 -> 0101
and
3 -> 0011
1 -> 0001

How Page Numbers for Logical Addresses are determined

2 STEPS:

1)

BIT SHIFTING OPERATIONS: Right shift on the logical addresses by the offset bits

2)

BitWise AND operation on Step 1 and the page_mask

FRAME NUMBER

Using the tlbSearch function and a page number,
The function returns a frame number if found in the tlb.

If page_number != -1

We have a tlb hit. Otherwise we have a tlb miss and the frame number is retrieved from the pagetable.

Each time the frame number == -1:
-It's considered a page fault, and we increase the count of page faults.

The Physical Addresses

```
int physical_addresses = (frame_number << OFFSET_BITS) | offset;
```

2 steps:

1- left shift the frame number by the offset bits
2- bitwise operation OR on 1 and the actual offset

THE VALUES

```
signed char value = main_memory[frame_number * PAGE_SIZE + offset];
```

The values are located in the main_memory array
At index = frame_number * page_size + the offset