PROJECT – Virtual Memory Simulator Using Java Script

Documentation

And

Test Cases

Prepared by Ganesh Sonwale

PART – 1

DOCUMENTATION

- Introduction
- Overall Description
- Working Description

1. INTRODUCTION

The Virtual Memory Simulator, is JavaScript version of the simulator.

It helps in better understanding of working of virtual memory, pages replacement policies.

Languages Used:

HTML

CSS

JavaScript

2.Overall Description

2.1) Parameters:

It contains all parameters which should be chosen required for working of simulator, options for main memory and virtual memory size under PARAMETERS varies depending upon page size selected.

2.2) Page replacement policy:

As name suggests here page replacement policy can be selected, which can be used to decide the next victim page in main memory in case of page fault and if main memory is full.

2.3) Page level:

It is used to select page level in virtual memory.

2.4) Submit Button:

After choosing all above parameters submit button is used to record the choices made by user, and reflect them in the respective areas made for them to be shown for reference of user while running simulator.

2.5) Generate and map Button (for address):

It generates the hexadecimal address randomly in range of address of size of virtual memory, it can be considered as request is generated for that address. Map button maps the generated address in main memory.

2.6) Selection of READ/WRITE:

Here which operation should be performed is selected by user on generated address.

2.7) Generate and map button (for page number):

It generates the page numbers randomly in range of size of virtual memory, it can be considered as request is generated for that page. Map button maps the generated page number in main memory.

2.8) Restart button:

Restart button restarts the simulator to run it again from beginning.

2.9) Textbox below OUTER, INNER, VPO:

These textboxes indicates the outer page number (only in case of page level 2), inner page number (simply page number in case of page level 1), and offset respectively. Number of page miss and page hit reflected each time when address/page number is generated or mapped according to situation.

2.10) section under BASIC:

1. Text area under virtual memory indicates virtual page number
2. Text area under main memory indicates which physical memory page is mapped to which virtual page and in time of page into main memory.
3. Text area under dirty indicates dirty bit corresponding to physical page, to decide is it necessary to update page in main memory to disk before removing it from main memory
4. Text area under explanation gives explanation about process done in simulator after generating/mapping done.

5. VPN indicates virtual page number, PPN physical page number, Valid shows valid bit (is virtual page present in main memory or not).

2.11) Recent history:

In this text area recent history like generated address and was it page hit or page fault. Text boxes above it reflects the choices made by user (shows parameters selected).

3. Working Description

Here functionalities of all the functions defined inside the program is discussed.

Global variables- pg (stores value of page size), mm (stores main memory size), vm (stores virtual memory size), size (indicates number of pages present in virtual memory), sz (indicates number of pages in main memory), mm_ctr(used as counter till main memory is full),ctr(counter used to get in time of page), pg_hit(stores number of page hits), pg_miss (stores number of page miss), rep_policy(stores the replacement policy), pg_lvl(stores page level), mem_map(it is 2-d array used to store all mapping info.)

Functions:

get_parameters

map

generate

map_page_no

generate_page_no

add_vm

add_vpn

add_vpn_1

add_vpn_2

add_vm_1

add_mm

add_to_explanation

add_to_recent_history

FIFO

LRU

LIFO

Fill_array

dynamicdropdown

3.1) get_parameters:

This function is on-click listener, it runs when SUBMIT button is clicked. This function saves the choices made by the user for further use in simulation. Parameters are saved in Global variables assigned to them.

This functions takes choices made by users to store them in by processing and converting them from string to integer, reflects them to their respective fields for reference while running simulator, initializes the array mem_map, add text to text areas under VPN and VIRTUAL MEMORY also initializes the global variables ctr, mem_ctr, pg_hit, pg_miss to zero.

3.2) generate:

This function randomly generates a hex-address in range supporting to size of virtual memory and shows it in textbox at right side of button. It also breaks address in two or three parts (depending upon page level) as page number and offset, further it updates the entries of mem_map array depending upon generated address and maps virtual address to physical address in main memory. It checks for page hit or page fault also and in case of page fault another function is called depending upon selection of page replacement policy, also updates text areas under BASIC button as well as text boxes under INNER, OUTER, VPO button.

3.3) map:

This function maps the address present in textbox in which hex address was generated (It can be user entered in range of size of

virtual memory or randomly generated). It also breaks address in two or three parts (depending upon page level) as page number and offset, further it updates the entries of mem_map array depending upon generated address and maps virtual address to physical address in main memory. It checks for page hit or page fault also and in case of page fault another function is called depending upon selection of page replacement policy, also updates text areas under BASIC button as well as text boxes under INNER, OUTER, VPO button.

3.4) generate_page_no:

This function randomly generates a page number in range supporting to size of virtual memory and shows it in textbox at right side of button. Further it updates the entries of mem_map array depending upon generated address and maps virtual address to physical address in main memory. It checks for page hit or page fault also and in case of page fault another function is called depending upon selection of page replacement policy, also updates text areas under BASIC button as well as text boxes under INNER, OUTER, VPO button.

3.5) map_page_no:

This function maps the page number present in textbox in which page number was generated (It can be user entered in range of size of virtual memory or randomly generated). Further it updates the entries of mem_map array depending upon generated address and maps virtual address to physical address in main memory. It checks for page hit or page fault also and in case of page fault another function is called depending upon selection of page replacement

policy, also updates text areas under BASIC button as well as text boxes under INNER, OUTER, VPO button.

3.6) add_vm:

This function adds the text which can be shown when running simulator for keeping track of which virtual page is used recently, also adds the text in text areas under PPN and VALID button for same purpose. It is called from get_paramerters function.

3.7) add_vpn_1:

This function is called from all four functions (generate, map, generate_page_no, map_page_no) when page level is ONE, to update the text areas under buttons VPN, PPN, VALID depending upon address/page number passed, for better understanding it also highlights the current virtual page, main memory page, its valid bit.

3.8) add_vpn_2:

This function is called from all four functions (generate, map, generate_page_no, map_page_no) when page level is TWO, to update the text areas under buttons VPN, PPN, VALID depending upon address/page number passed, for better understanding it also highlights the current virtual page, main memory page, its valid bit.

3.9) add_vm_1:

This function is called from all four functions (generate, map, generate_page_no, map_page_no) to update text area under

VIRTUAL MEMORY button every time when a generate/map button is pressed, it removes previous data and add updated data, also highlights current virtual page for better understanding.

3.10) add_mm:

This function is called from all four functions (generate, map, generate_page_no, map_page_no) to update text areas under MAIN MEMORY and DIRTY buttons every time when a generate/map button is pressed based on , it also highlights the current page which has been mapped for better understanding.

3.11) add_to _explanation:

This function adds the explanation about what is happened, it gives more information about which virtual page is mapped to which main memory page, was it a page hit or page miss, which virtual page was replaced depending on replacement policy. Adds this information in text area in EXPLANATION area under BASIC button. It is called whenever generate/map button is pressed.

3.12) add_to_recent_history:

This function adds the limited information about what is happened, it provides information about latest request as well as all previous ones like, was it a page hit or page miss, which virtual page was accessed and address/page number. Adds this information in text area in RECENT HISTORY. It is called whenever generate/map button is pressed.

3.13) FIFO:

This function is called whenever generate/map button is clicked if and only if all main memory pages are mapped already and a page fault is occurred then for replacement of page we need to find victim page so, in-order to find that victim page FIFO is called if selected page replacement policy is FIFO.

3.14) LIFO:

This function is called whenever generate/map button is clicked if and only if all main memory pages are mapped already and a page fault is occurred then for replacement of page we need to find victim page so, in-order to find that victim page LIFO is called if selected page replacement policy is LIFO.

3.15) LRU:

This function is called whenever generate/map button is clicked if and only if all main memory pages are mapped already and a page fault is occurred then for replacement of page we need to find victim page so, in-order to find that victim page FIFO is called if selected page replacement policy is FIFO.

3.16) fill_array:

This function is called when submit button is clicked through get_parameters function to initialize the 2-D array to store all mapping which is going to happen when simulator is run, it has same number of rows as there are pages in virtual memory, and 7-columns.

Column-0: virtual page number

Column-1: physical page number

Column-2: valid bit

Column-3: in time in PPG

Column-4: entering sequence in PPG of VPG

Column-5: dirty bit

Column-6: recently used time

All columns except column 0,2 are initialized to -1, col-0 is initialized to respective row number and col-2 initialized to 0.

3.17) dynamicdropdown:

This function is invoked at start when user selects page size, based on that virtual memory and main memory sizes are updated.

Prepared By:-

Ganesh Sonwale

Project Details:-

Project Domain – CSO

Issue Number – 206

Project Name – Virtual memory simulator

Submitted TO:-

SRIP@IIITH