HW 2: Memory

Deadline 2021/1/3 23:55

- **Notice**
  - **Please make sure your program can run on Linux.**
  - 10% score penalty for homework submitted in wrong formats.
  - 30% score penalty for late submissions.

- **Part I - Shared Memory**
  - Shared memory segment is an important ipc mechanism for communication between multiple programs. We create a shared memory segment and attach the programs to the memory segment. After that, the programs are able to read and modify the memory content.
  - **[Part I-1 40pts]** We provide server.c, client.c, and Makefile. Implement the TODOs in server.c and client.c. Create a shared memory segment in server.c and attach the shared memory segment in client.c. If you set up the shared memory properly, writes to the shared memory segment by one program will be seen by the other program.
  - Compilation and Usage
    You could simply compile them by
    ```
    $ make
    ```
    launch the server program
    ```
    $ ./server
    ```
    and launch the client programs in another terminal
    ```
    $ ./client
    ```
  - Expected result

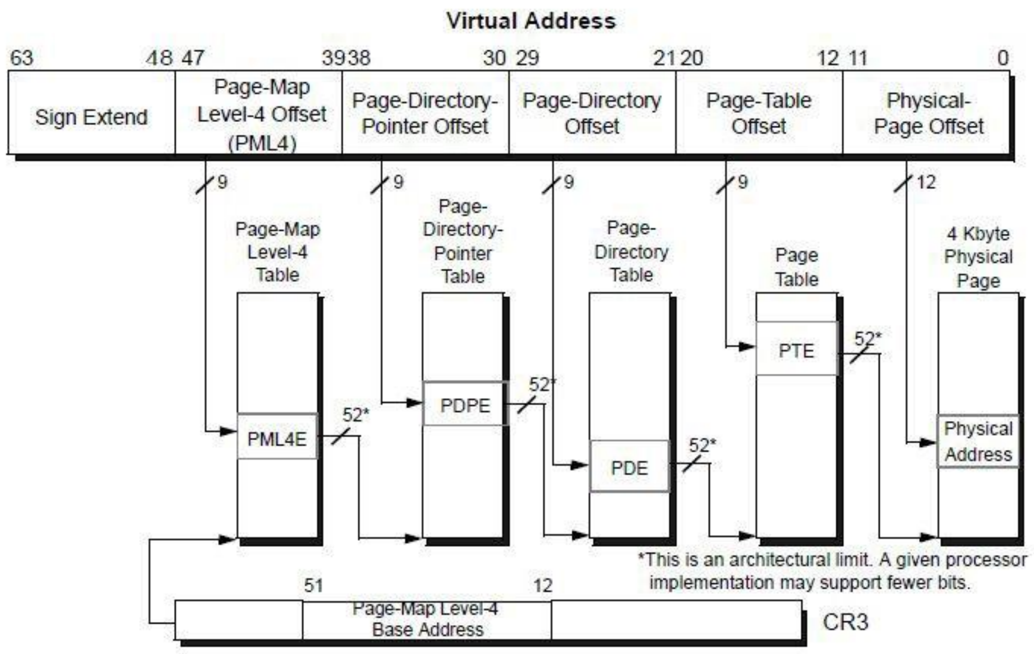
Server	Client

[server] The value is 0	[client] The value is 0
1: Show the value	1: Show the value
2: Modify the value	2: Modify the value
3: Exit	3: Exit
Enter commands: 1	Enter commands: 1
[server] The value is 0	[client] The value is 0
1: Show the value	
2: Modify the value	
3: Exit	
Enter commands: 2	
Input new value: 231	
	1: Show the value
	2: Modify the value
	3: Exit
	Enter commands: 1
	[client] The value is 231
	1: Show the value
	2: Modify the value
	3: Exit
	Enter commands: 2
	Input new value: -97
1: Show the value	
2: Modify the value	
3: Exit	
Enter commands: 1	
[server] The value is -97	

- **[Part I-2 10pts]** There are some potential issues with the programs. Please identify at least two issues and describe the corresponding solutions that would fix the issues. Upload your answer as **{Student_id}.pdf**. (Hint: what if several modifications happened at the same time or the modified values were invalid)

• Part II - Page Table

- Page table is the primary technique to implement virtual memory. The CPU uses the value of the CR3 register to locate the page table. In Part II, you are asked to modify the page table directly to map two different virtual addresses to the same physical address.



- For example,
  - cr3 : 0x70300e000
  - virtual address : 0x563229f05456

	PML4	PDPT	PD	PT	offset
index	0xac	0xc8	0x14f	0x105	0x456
index*8	0x560	0x640	0xa78	0x828	

  - Address of PML4 entry = 0x70300e560
    Content of PML4 entry = 0x80000007131b8067
  - Address of PDPT entry = 0x7131b8640
    Content of PDPT entry = 0x6b84bf067
  - Address of PD entry = 0x6b84bfa78
    Content of PD entry = ...

- Address of PT entry = ...
  Content of PT entry = …

- In the program, there are two strings

  char x[] = "This is a simple HW"

  char y[] = "You have to modify my page table"

  Please map the different virtual addresses to the same physical address
- **[Part II-1 25pts]** Add code to the program to modify the page table entries so that the printf("%s", y) at Line 72 shows the content of x on the screen and the printf statements at Line 78 and 79 show "When you modify y, x is modified actually".
- **[Part II-2 25pts]** Add code to the program to modify the page table entries so that the printf("%s", y) at Line 92 shows the content of y on the screen
- We provide three functions **get_cr3_value()**, **read_physical_address()** and **write_physical_address()** that allow your program to access the cr3 register value and to perform read/write at a chosen physical address
- Make sure all your modifications and code are self-contained in **page_table.c** since this is the only file you will hand in for this part of the homework.
- **Notice:** Just simulating the expected results (e.g., using strcpy() etc) will not get any points for the homework. You have to set up the page table entries properly.
- Compilation and Usage
  We provide page_table.c, module-related files, and Makefile. You could simply compile them by

  ```
  $ make
  ```
  and install the kernel module by
  ```
  $ sudo make install
  ```
  If you want to remove the module, please type
  ```
  $ sudo make remove
  ```

to execute the program

```
$ ./page_table
```

- Expected result

```
Before
x : This is a simple HW
y : You have to modify my page table.

After modifying page table
x : This is a simple HW
y : This is a simple HW

After modifying string y
x : When you modify y, x is modified actually.
y : When you modify y, x is modified actually.

After recovering page table of y
x : When you modify y, x is modified actually.
y : You have to modify my page table.
```

Submission Rules

- Put **server.c**, **client.c**, and **page_table.c** in directory **{Student_id}** and compress as **{Student_id}.zip**
- Submit the {Student_id}.pdf (partI-2) and the zip file to **e3**.