

**CS342301: Operating System**  
**MP2: Multi-Programming**  
**Deadline: 2019/11/11 23:59**

## I. Goal

1. Understand how memory management works in NachOS
2. Understand how to implement page table mechanism

## II. Assignment

1. Trace code
  - Starting from “threads/kernel.cc **Kernel::ExecAll()**”, “threads/thread.cc **thread::Sleep**” until “machine/mipsim.cc **Machine::Run()**” is called for executing the first instruction from the user program.

2. Implement page table in NachOS

- Working item: Modify its memory management code to make NachOS support multi-programming.
- Verification:

☐ Wrong results without multi-programming

```
[ta@lsalab ~/ryan/NachOS-4.0_MPL/code/test]$ ../build.linux/nachos -e consoleIO_test1 -e consoleIO_test2
consoleIO_test1
consoleIO_test2
9
16
15
18
19
1return value:0
7
return value:0
█
```

☐ Correct results with multi-programming

```
[ta@lsalab ~/ryan/MP2_sol/code/test]$ ../build.linux/nachos -e consoleIO_test1 -e consoleIO_test2
consoleIO_test1
consoleIO_test2
9
8
7
6
1return value:0
5
16
17
18
19
return value:0
█
```

- **Note:**
  - Be careful that program size might exceed a pagesize
  - You must put the data structure recording used physical memory in kernel.h / kernel.cc
  - You must set up “valid, readOnly, use, and dirty” field for your page table, which is defined under “translate.h TranslationEntry class”
  - The size of your page table must be able to contain the whole physical memory
- Hint: The following files “may” be modified...
  - ☐ userprog/addrspace.\*
  - ☐ threads/kernel.\*

### 3. Report

- Cover page, including team members, Team member contribution.
- Explain your implementation as requested in Part II-2.
- Explain how NachOS creates a thread(process), load it into memory and place it into scheduling queue as requested in Part II-1. Your explanation on the functions along the code path should **at least** cover answer for the questions below:
  - ☐ How Nachos allocates the memory space for new thread(process)?
  - ☐ How Nachos initializes the memory content of a thread(process), including loading the user binary code in the memory?
  - ☐ How Nachos creates and manages the page table?
  - ☐ How Nachos translates address?
  - ☐ How Nachos initializes the machine status (registers, etc) before running a thread(process)
  - ☐ Which **object** in Nachos acts the role of **process control block**
  - ☐ When and how does a thread get added into the ReadyToRun queue of Nachos CPU scheduler?

### III. Instruction

1. Copy your code for MP1 to a new folder
  - `cp -r NachOS-4.0_MP1 NachOS-4.0_MP2`
2. Copy test file
  - `cp /home/os2019/share/consoleIO_test* NachOS-4.0_MP2/code/test/`
3. Test your program
  - ☐ `cd NachOS-4.0_MP2/code/test`
  - ☐ `../build.linux/nachos -e consoleIO_test1 -e consoleIO_test2`

### IV. Grading

1. Implementation correctness – 60%
  - Execute “`../build.linux/nachos -e consoleIO_test1 -e consoleIO_test2`” correctly
2. Report – 20%
  - Upload it to iLMS with the Filename:  
**MP2\_report\_[GroupNumber].pdf.**
3. Demo– 20%
  - ☐ Answer questions during demo.
  - ☐ Demo will take place on our server, so you are responsible to make sure your code works on our server.

**\*Refer to syllabus for late submission penalty.**