3. Report
• Cover page, including team members, Team member contribution.

• Explain your implementation as requested in Part II-2.

First, we find out that the initial NachOS code provided by TAs being lack of memory management in our kernel. Second, we create an array to record which places have been used in memory space to prevent one's memory content from being overlapped by other processes. After that, we put this array into the function "Addrspace" constructor, since it can find spare memory space for itself with the memory usage array. The "Addspace" constructor will record the address of the array, so the kernel will know which places have been used by a thread.

We have modified the "read" and "write" function by changing virtual address to physical address to get the code and data segment of a process. We use a function called Translate to accomplish the above task.

• Explain how NachOS creates a thread(process), load it into memory and place it into the 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:

1Q. How Nachos allocates the memory space for new thread(process)?
1A. Through Kernel::Exec function, we can see that it creates a new thread and allocates new space. In this version, we allow a thread to occupy the whole physical memory space.

2Q. How Nachos initializes the memory content of a thread(process), including loading the user binary code in the memory?
2A. We initialize the entire memory space through Addrspace() function and load the user binary code in the memory by calling AddrSpace::Load() function in the kernel.

3Q. How Nachos creates and manages the page table?
3A. NachOS creates a page table in Addrspace::Addrspace(), whose size is "NumPhysPages". When doing context switch(AddrSpace::RestoreState()), "kernel->machine" will get the new page table and page table size from the new thread.

4Q. How Nachos translates address?
4A. It uses virtual page number(vpn) and offset to get the corresponding physical address. In this function, we will check if virtual page number or page frame number is valid. Finally, we set the status of the page table entry(use, dirty).

5Q. How Nachos initializes the machine status (registers, etc) before running a thread(process)?
5A. In AddrSpace::Execute() function, it set the initial register values and load page table register.

6Q. Which object in Nachos acts the role of process control block?
6A. Thread::(void)*machinestate will act the role of PCB.

7Q. When and how does a thread get added into the ReadyToRun queue of Nachos CPU scheduler?
7A.When calling Thread::Fork(), Thread::Yield(), Semaphore::V(), they will call Scheduler::ReadyToRun (Thread *thread) function, which makes a thread get added into the ReadyToRun queue by this line: readyList->Append(thread);