Amauri Lopez
Lab 9
Address Space and Paging
Chapter 1 and 2

Each operating system has a responsibility to ensure progress for each process, so that each process can eventually approach completion, even if there may not be enough processors or cores to support all the processing being executed. Also, if a process is to fail, then it should not affect the other processes in such a way that inter-process communication is lost, and these processes should be strictly isolated. The xv6 operating system takes these circumstances into account by using page tables, and assigning each process an address space - a private memory space. The purpose of the page table is to map the virtual addresses to physical addresses on the chip. Each process page table holds the kernel's instructions, data, and program memory, and each process has its own page table.

When the xv6 operating system starts, the system loads the kernel from the disk into memory at a physical address. Some addresses are occupied by I/O devices and peripherals. A page table is then established that maps the virtual addresses to the physical ones, and as a result, the kernel instructions and data are limited to 4MB. With a page table, the first process (the "init" or *userinit)* parent process begins making space to the page table to hold processes. This processes which allocates space to the page table is called *allocproc*. This process is called for every other process besides the init. Each "slot" in the page table that is empty is marked with an "unused", and when the slot is used then it is marked at an "embryo". When an embryo is established, then it is assigned an process ID, or PID, and a kernel stack. If nothing fails, then *userinit* will finalize the build procedure and set the process state to "runnable". Afterwards, "scheduler" is started to search for processes that are runnable, and switched the process state to "Running". Because "init" is the first process that satisfies this, this process alerts the kernel to begin using the page table.

For the information, each page table has a physical page number that is 20-bits in size. The 12 low bits from the virtual address are copied into a physical address, becoming a unit of 4096 bits, or a page size. Each page table entry also contains flags identifying the state of the entry which include present, writable, etc.