PROCESS UTILIZATION

Read one record from file	15µs
Execute 100 instructions	1µs
Write one record to file	15µs
Total=	31µs

Percent CPU utilization = 1/31= 0.032 = 3.2%

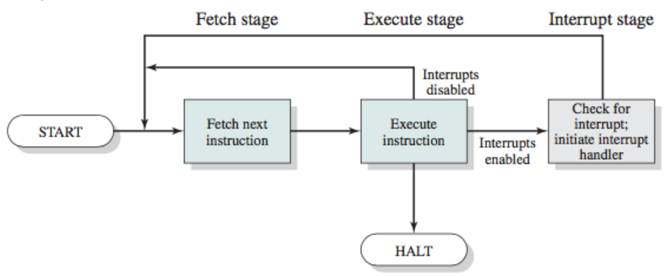
TROUGHPUT
This is a measure of how much work is being performed. It depends on the average length of a process but is also influenced by scheduling policy.

RESPONSE TIME
This is the time from the submission of a request until the response begins to be received. It is a measure for the user's perspective. Turnaround time is the machines perspective. You want low response time to maximize the number of I/O you can do. Uniprogramming has a longer mean response time than multiprogramming. This is because everything has to be done sequentially.

FIVE STATE PROCESS MODEL
Running: the process is being executed
Ready: a process is prepared to execute
Blocked/Waiting: A process that can't execute until some event occurs, ex. I/O
New: A new process take a second to get from main memory to the process pool
Exit: A released process, halted or aborted.

SEVEN STATE PROCESS MODEL
Blocked/Suspend: Process is in secondary memory
Ready/Suspend: In secondary memory but is available for execution

INTERUPTS
Program, Timer, I/O, Hardware failure



MEMORY HIERARCHY
-Inboard memory
- Registers
- Cache
- Main memory

-Outboard storage
- magnetic disk
- CDs,DVDs,BR

-Offline storage
- Magnetic tape

If accessed word is found faster in memory that's a hit, a miss occurs if the accessed word is not found in faster memory.

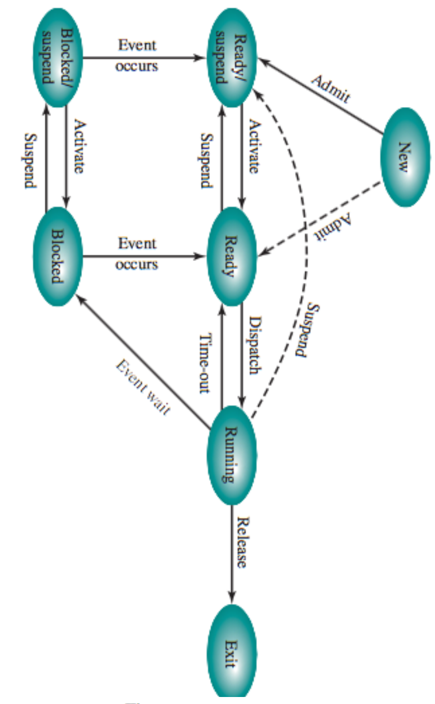
DMA, INTERUPT DRIVEN I/O vs PROGRAMMED I/O
DMA is sent by the processor to go grab blocks of memory and let it know when it has them.
The overall effect is to cause the processor to execute more slowly during a DMA transfer when processor access to the bus is required. Still a multiple-word I/O transfer DMA is more efficient than interrupt of programmed I/O

Table 2.3 Batch Multiprogramming versus Time Sharing

	Batch Multiprogramming	**Time Sharing**
Principal objective	Maximize processor use	Minimize response time
Source of directives to operating system	Job control language commands provided with the job	Commands entered at the terminal

Table 5.3 Common Concurrency Mechanisms

Semaphore	An integer value used for signaling among processes. Only three operations may be performed on a semaphore, all of which are atomic: initialize, decrement, and increment. The decrement operation may result in the blocking of a process, and the increment operation may result in the unblocking of a process. Also known as a **counting semaphore** or a **general semaphore.**
Binary Semaphore	A semaphore that takes on only the values 0 and 1.
Mutex	Similar to a binary semaphore. A key difference between the two is that the process that locks the mutex (sets the value to zero) must be the one to unlock it (sets the value to 1).
Condition Variable	A data type that is used to block a process or thread until a particular condition is true.
Monitor	A programming language construct that encapsulates variables, access procedures, and initialization code within an abstract data type. The monitor's variable may only be accessed via its access procedures and only one process may be actively accessing the monitor at any one time. The access procedures are *critical sections*. A monitor may have a queue of processes that are waiting to access it.
Event Flags	A memory word used as a synchronization mechanism. Application code may associate a different event with each bit in a flag. A thread can wait for either a single event or a combination of events by checking one or multiple bits in the corresponding flag. The thread is blocked until all of the required bits are set (AND) or until at least one of the bits is set (OR).
Mailboxes/Messages	A means for two processes to exchange information and that may be used for synchronization.
Spinlocks	Mutual exclusion mechanism in which a process executes in an infinite loop waiting for the value of a lock variable to indicate availability.



(b) With two suspend states

Figure 3.9 Process State Transition Diagram with Suspend States

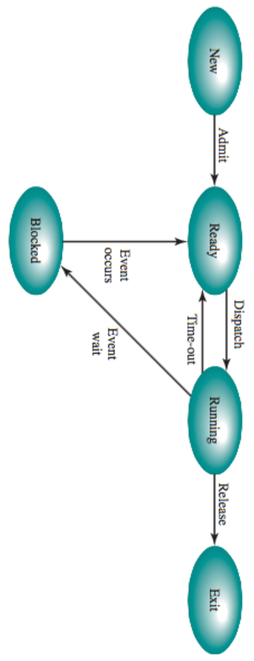


Figure 3.6 Five-State Process Model

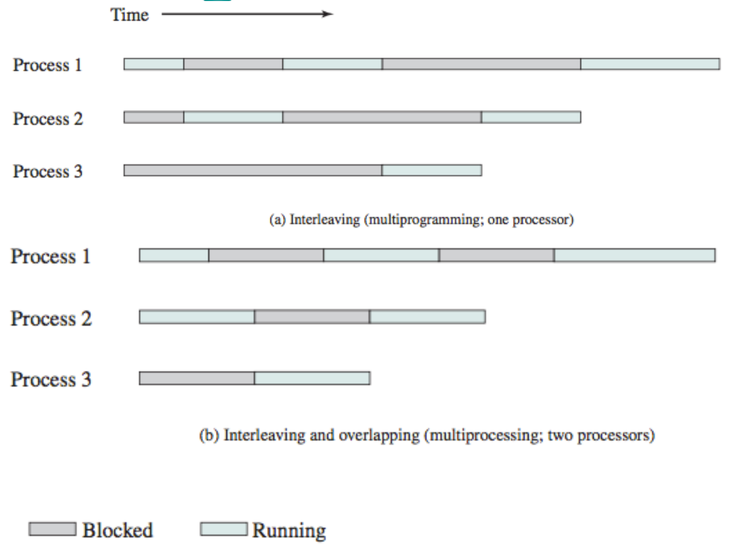


Figure 2.12 Multiprogramming and Multiprocessing

FIRST FIT is the most likely the best placement algo

PROCESS vs THREAD
Process: A collection of one or more threads and associated system *resources*, ex. A program in execution
Thread: A dispatchable unit of work. Includes processor context(PC and Stack pointer). They are executed sequentially and can be interrupted.

PROCESS IMAGE
Contains user data(program data), User program, Stack, and a Process control block

EXECUTION CONTEXT
Or processor state is the internal data by which the OS is able to supervise and control the process.

KERNEL MODE vs USER MODE
Kernel mode- Privileged instructions may be executed and protected areas of memory can be accessed
User mode- less access

MODE SWITCHING
Sets program counter to the starting address of an interrupt handler program
Switches from user mode to kernel mode

SWAPPING
Involves moving part or all of a process from main memory to disk. When none of the processes in main memory is in the Ready state, the OS swaps one of the blocked processes out on to disk into a suspended queue. This is a queue of existing processes. Swapping is an I/O operation.

DISPATCHER vs SCHEDULER
Dispatcher- a program that switches the processor from one processes to another
Scheduler- schedules processes

ULT vs KLT vs LWP
User Level Threads- akk if the work of thread management is done by the application and the kernel is not aware of the existence
Kernel Level Threads- all of the work of thread management is done by the kernel
Light Weight Process- are a mapping between ULTs and kernel threads. Each LWP supports ULT and maps to one kernel thread. They can be executed in parallel on multiprocessors.

JACKETING
A way to overcome the problem of blocking threads. It converts a blocking system call into a non blocking system call.

HIT RATIO
(Hit Rate)(Hit Time) + (Miss Rate)(Miss Time) = AVG
ex. (0.9)(20ns)+(0.1)((0.6)(60ns+20ns)+(0.4)(12,000,080ns)

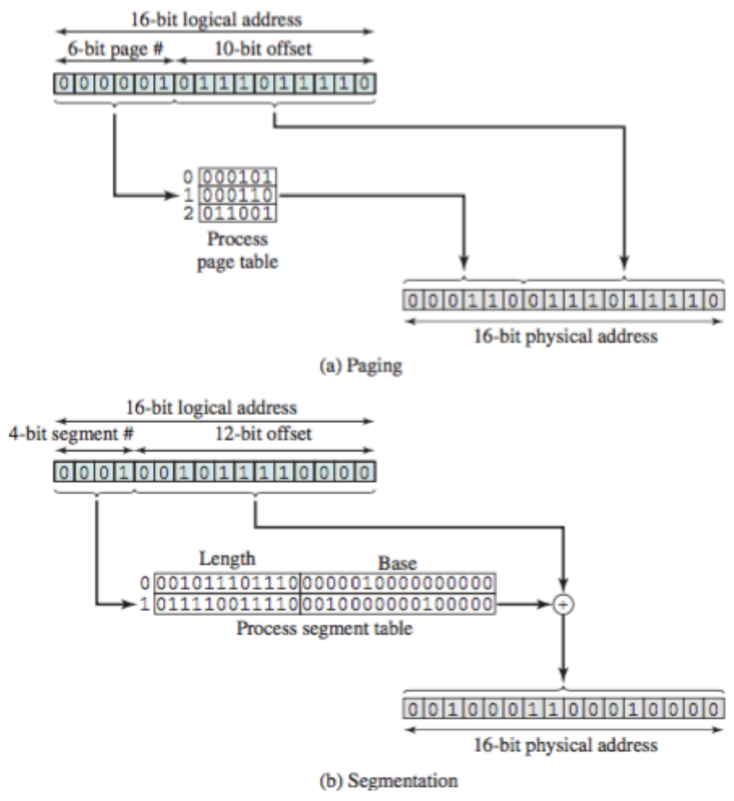


Figure 7.12 Examples of Logical-to-Physical Address Translation

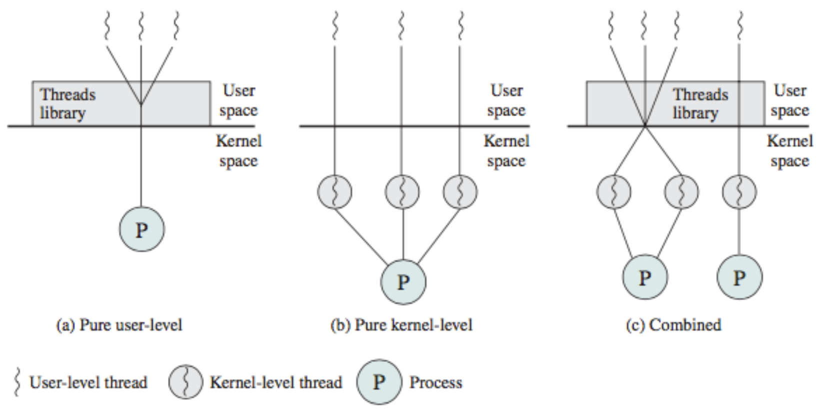


Figure 4.5 User-Level and Kernel-Level Threads

Table 3.9 UNIX Process States

User Running	Executing in user mode.
Kernel Running	Executing in kernel mode.
Ready to Run, in Memory	Ready to run as soon as the kernel schedules it.
Asleep in Memory	Unable to execute until an event occurs; process is in main memory (a blocked state).
Ready to Run, Swapped	Process is ready to run, but the swapper must swap the process into main memory before the kernel can schedule it to execute.
Sleeping, Swapped	The process is awaiting an event and has been swapped to secondary storage (a blocked state).
Preempted	Process is returning from kernel to user mode, but the kernel preempts it and does a process switch to schedule another process.
Created	Process is newly created and not yet ready to run.
Zombie	Process no longer exists, but it leaves a record for its parent process to collect.

Table 7.1 Memory Management Terms

Frame	A fixed-length block of main memory.
Page	A fixed-length block of data that resides in secondary memory (such as disk). A page of data may temporarily be copied into a frame of main memory.
Segment	A variable-length block of data that resides in secondary memory. An entire segment may temporarily be copied into an available region of main memory (segmentation) or the segment may be divided into pages which can be individually copied into main memory (combined segmentation and paging).