Q1.9

a. To initiate a DMA transfer, the CPU first sets up the DMA registers, which contain a pointer to the source of a transfer, a pointer to the destination of the transfer, and a counter of the number of bytes to be transferred. Then the DMA controller proceeds to place addresses on the bus to perform transfers, while the CPU is available to accomplish other work.
b. When the transfer is finished, the DMA controller interrupts the CPU.
c. Yes. Both the CPU and the DMA controller are bus masters. A problem would be created if they access the memory at the same time. Accordingly, the CPU should be momentarily prevented from accessing main memory when the DMA controller seizes the memory bus.
However, if the CPU is still allowed to access data in its primary and secondary caches, a coherency issue may be created if both the CPU and the DMA controller update the same memory locations.

Q2.7

1. Shared memory model
Pros:
    - Faster when the processes are on the same machine.
Cons:
    - Different processes need to endure that they are not writing to the same location at the same time.
    - Need to be aware of the memory protection and synchronization problems.
2. Message passing model
Pros:
    - Easier to implement
Cons:
    - slower than shared memory model because of the time involved in connection setup

Q2.10

1. advantages:
    a. adding a new service does not require modifying the kernel (easier to extend)
    b. more secure as more operations are done in user mode
    c. simpler kernel design and functionality results in a more reliable operating system
2. user programs and system services interact in a microkernel architecture by using interprocess communication mechanisms such as messaging. These messages are conveyed by the OS.
3. disadvantages:
the overheads associated with the interprocess communication and the frequent use of the operating system's messaging functions to enable the user process and the system service to interact with each other

Q3.1

Chart for comparing differences among short-term, medium-term, and long-term scheduling.

	Long Term	Short Term	Medium Term
type	job scheduler	CPU scheduler	job swapping

Speed	slower than the short term scheduler	very fast	between both
control of the degree of multiprogramming	controls the degree of multiprogramming	less control over the degree of multiprogramming	reduce the degree of multiprogramming
situation in time-sharing system	absent or minimal	minimal	time-sharing system uses a medium-term scheduler
location to select	It selects processes from the pool and loads them into memory for execution.	It selects from among the processes that are ready to execute.	Processes can be reintroduced into the meat and its execution can be continued.
process state	New to Ready	Ready to Running	– / Ready to wait
function	Select a good mix of I/O bound and CPU bound	Select a new process for a CPU quite frequently	Improve process mix, free up memory

Q3.11

a. Synchronous and asynchronous communication
   A benefit of synchronous communication is that it allows a rendezvous between the sender and receiver. A disadvantage of a blocking send is that a rendezvous may not be required and the message could be delivered asynchronously. As a result, message-passing systems often provide both forms of synchronization.
b. Automatic and explicit buffering
   Automatic buffering provides a queue with indefinite length, thus ensuring the sender will never have to block while waiting to copy a message. There are no specifications on how automatic buffering will be provided; one scheme may reserve sufficiently large memory where much of the memory is wasted.
   Explicit buffering specifies how large the buffer is. In this situation, the sender may be blocked while waiting for available space in the queue. However, it is less likely that memory will be wasted with explicit buffering.
c. Send by copy and send by reference
   Send by copy does not allow the receiver to alter the state of the parameter; send by reference does allow it. A benefit of send by reference is that it allows the programmer to write a distributed version of a centralized application.
   Java's RMI provides both; however, passing a parameter by reference requires declaring the parameter as a remote object as well.
d. Fixed-sized and variable-sized messages
   The implications of this are mostly related to buffering issues; with fixed-size messages, a buffer with a specific size can hold a known number of messages. The number of variable-sized messages that can be held by such a buffer is unknown.
   Consider how Windows 2000 handles this situation: with fixed-sized messages (anything < 256 bytes), the messages are copied from the address space of the sender to the address space of the receiving process. Larger messages (i.e. variable-sized messages) use shared memory to pass the message.