CH1：

1.9

The CPU can initiate a DMA operation by writing values into special registers that can be independently accessed by the device. The device initiates the corresponding operation once it receives a command from the CPU. When the device is finished with its operation, it interrupts the CPU to indicate the completion of the operation. Both the device and the CPU can be accessing memory simultaneously. The memory controller provides access to the memory bus in a fair manner to these two entities. A CPU might therefore be unable to issue memory operations at peak speeds since it has to compete with the device in order to obtain access to the memory bus.

CH2：

2.7

|  |  |  |
| --- | --- | --- |
|  | Message-passing model | Shared-memory model |
| Pros | 1. useful for exchanging smaller amounts of data  2.easier to implement | 1.Allow maximum speed and convenience of communication |
| Cons | 1.slower than shared shared-memory because of the time involved in connection setup | 1.Compromise on protection and synchronization |

2.10

Pros：

1. Easier to extend the operation system
2. Easier to port the OS to new architectures
3. More security and reliability

Cons：

1. Increased system-function overhead
2. Operation system may have to switch from one process to the next to exchange the messages

User programs and system services interact in a microkernel architecture by using interprocess communication mechanisms such as messaging.

CH3：

3.1

Short-term：Selects from jobs in memory that are ready to execute and allocates the CPU to them.

Mid-term：Used specially with time-sharing systems as an intermediate scheduling level. A swapping scheme is implemented to remove partially run programs from memory and reinstate them later to continue where they left off.

Long-term：Determines which jobs are brought into memory for

processing. The primary difference is in the frequency of their execution. The

short-term must select a new process quite often. Long-term is used much less often

since it handles placing jobs in the system and may wait a while for a job to finish before it admits another one.

3.11

a) 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 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 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) 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.