1.2

1. The resource utilization. We need to make sure that all available CPU time, memory, and I/O share are used efficiently and that no individual user takes more than his fair share.
2. We need to compromise between individual usability and resource utilization.
3. Mostly individual usability is important, but performance per amount of battery life is important, too.

1.5

Asymmetric multiprocessing makes each processor do specific tasks, while symmetric multiprocessing makes each processor do all tasks in the system. An asymmetric multiprocessing system has a master processor, which controls the system and the other processors, while the processors in a symmetric multiprocessing system are equal.

The advantages:

1. The throughput is increased. We can get more work done in less time.
2. It can take advantage of the economy of scale, which means as the size of an industry increases, the lowest possible cost is often reduced.
3. The reliability is increased. It can still maintain limited functionality when any one of its processors halts or fails.

The disadvantage:

It’s harder to program for a multiprocessing system than a single-processor one.

1.10

The purpose of interrupts is to signal the occurrence of an event.

A trap can only be generated by software, while an interrupt can be generated by either hardware or software.

Yes.

A user program generates a trap to request some operating-system service be performed.

1.11

1. The CPU sends a command to initialize the device controller. When the controller received the command, it send a request to take control of the system bus. After the CPU finished present instruction, it response to the request. When the controller takes control of the system bus, DMA starts.
2. After a block of data has been transferred, one interrupt is generated to tell the CPU that DMA has finished.
3. When DMA finishes, the DMA controller sends a interrupt, which can make a user program stop.

1.13

1. First, with proper cache-management, we can put a copy of the data we use more frequently in cache to reduce the time we will use to access them again. Second, most systems have an instruction cache to hold the next instructions expected to be executed, making the CPU wait for less time.
2. Cache makes communication between components of different speeds more efficient.
3. If a variant A is used by various processes or CPUs at the same time, its value must be the same in different caches, which is called cache coherency.
4. It can be too expensive.