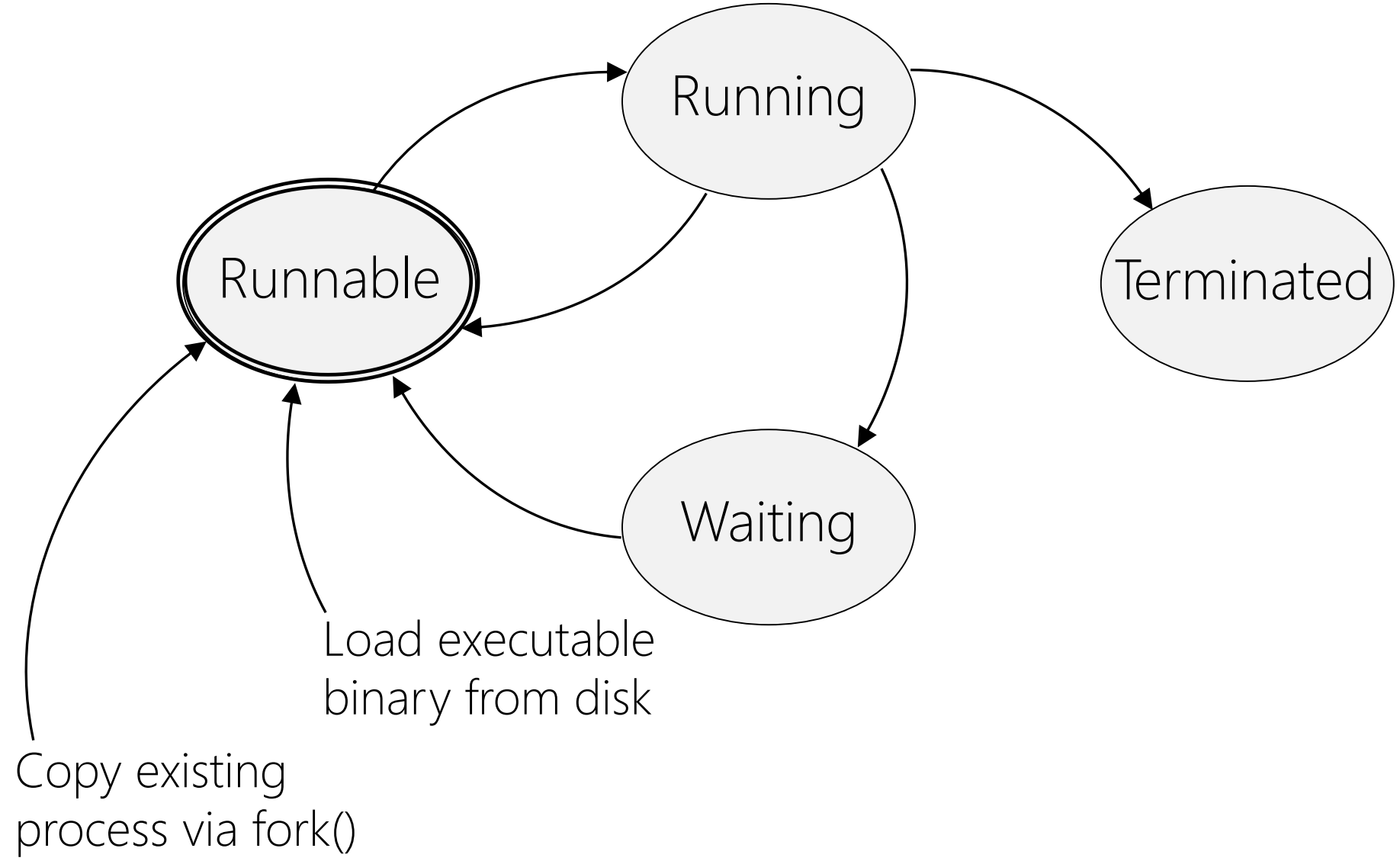
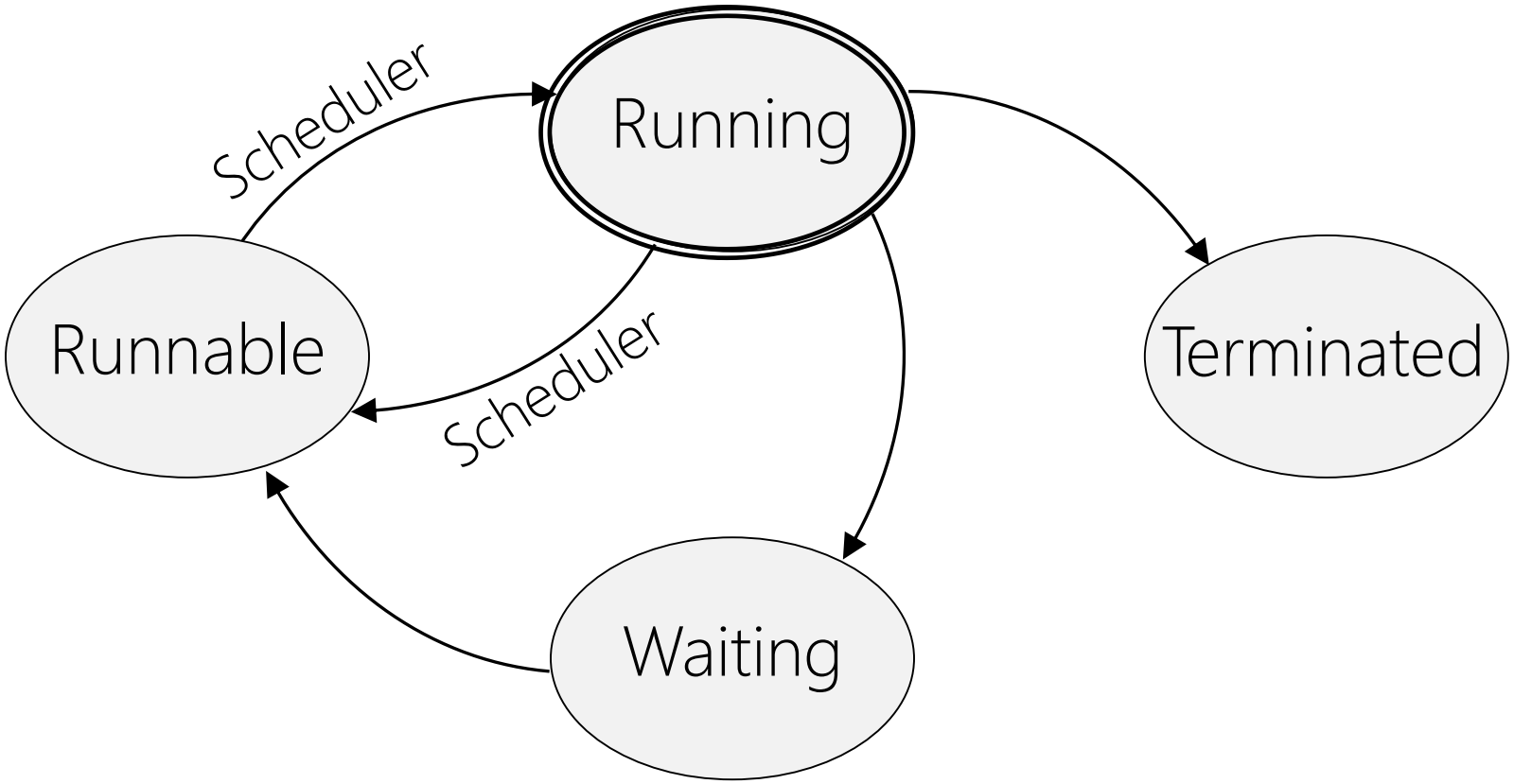
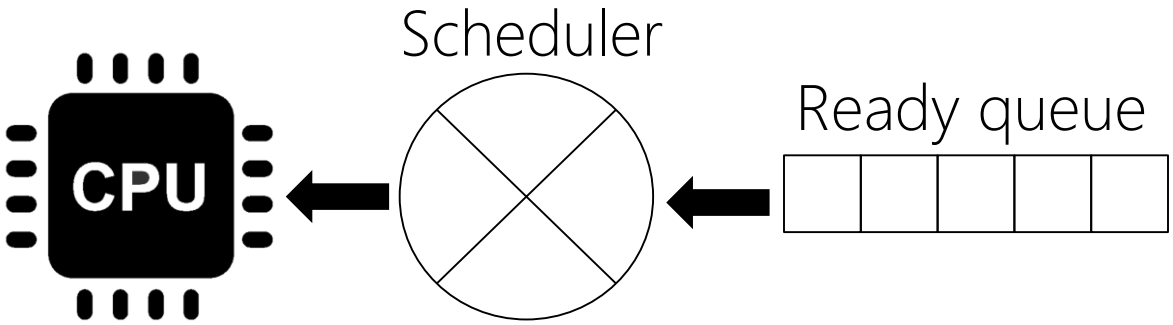
The Life Cycle for a Process

Process states



Process states

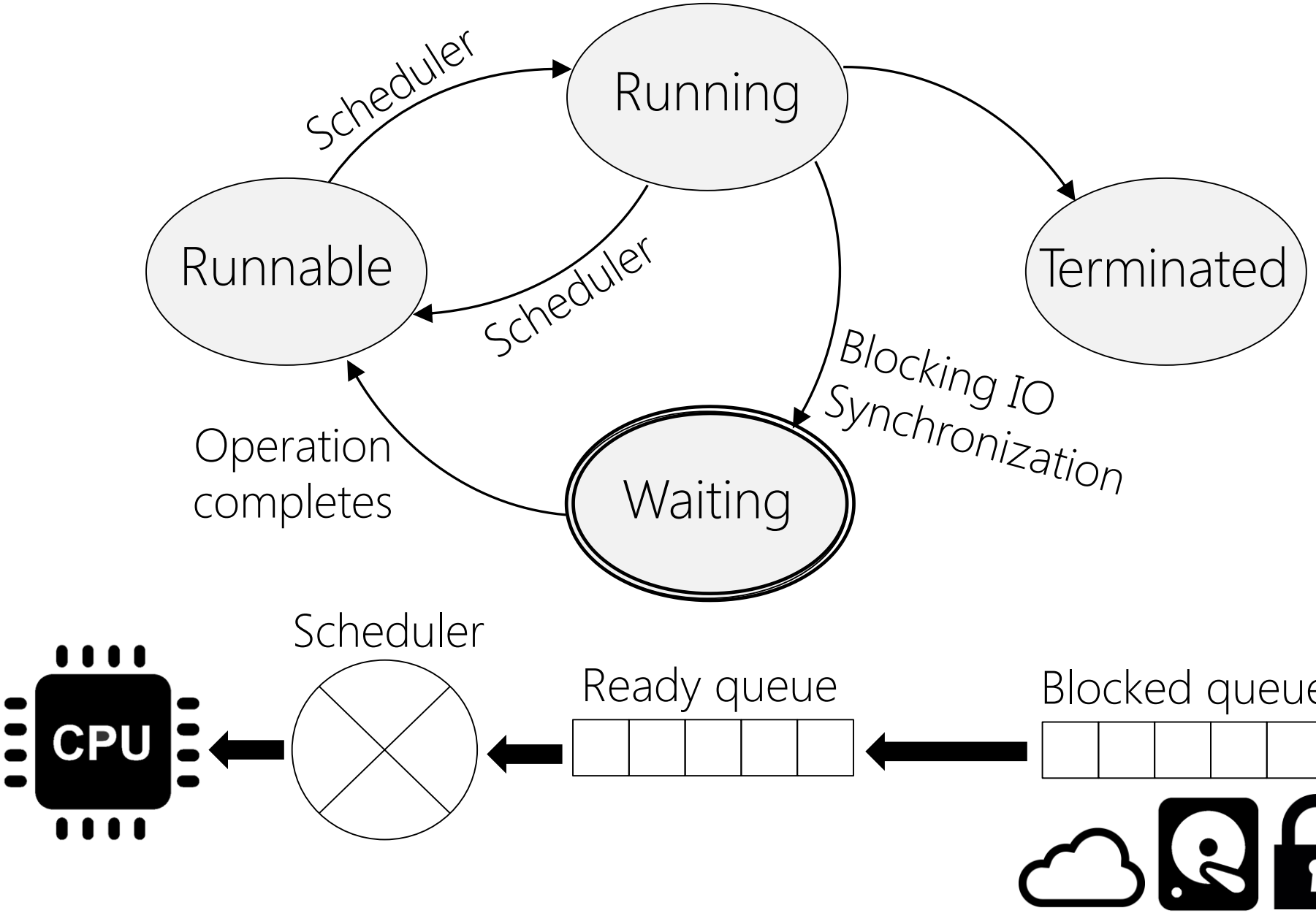




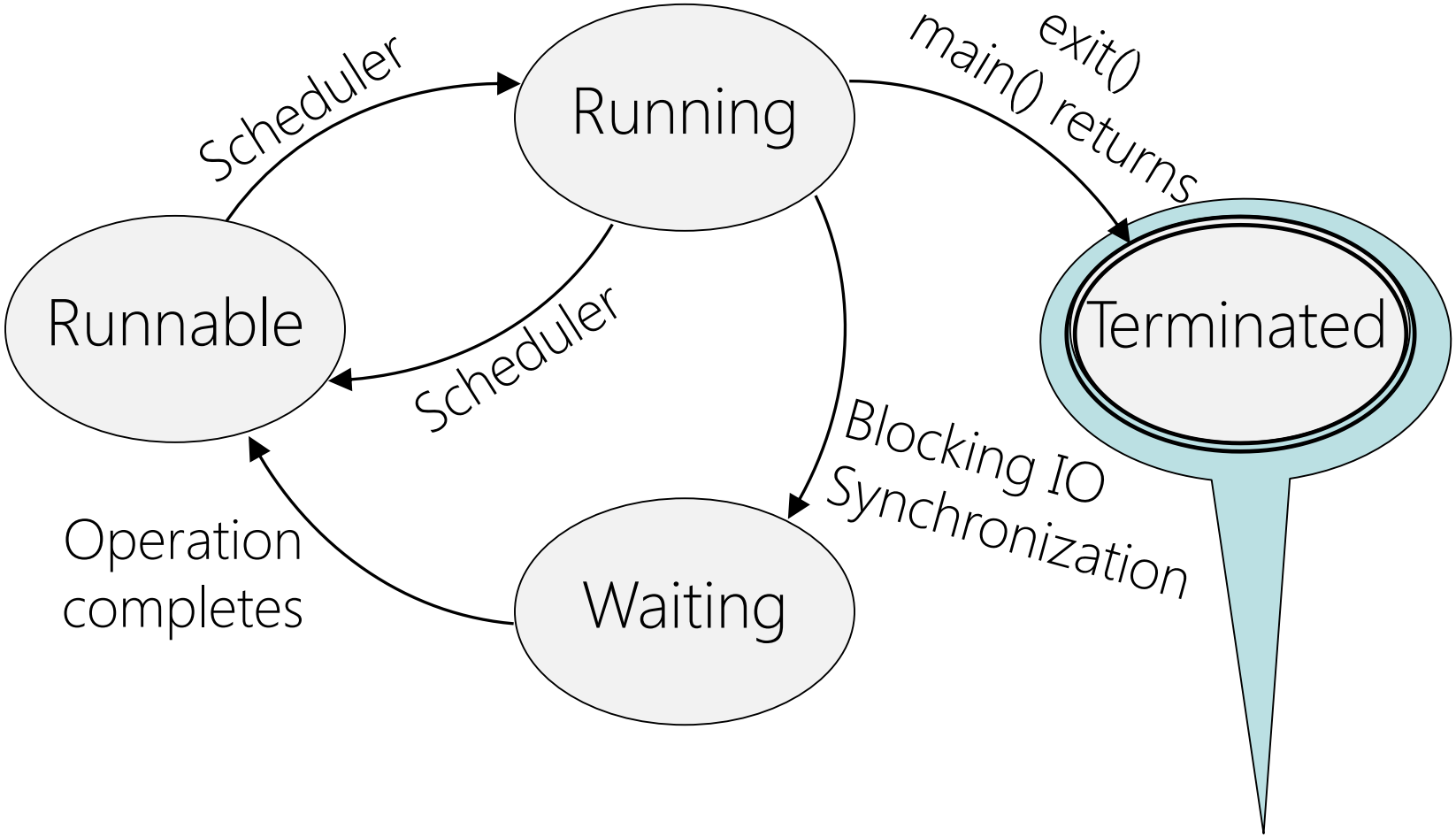
Scheduler Algorithms

- FIFO?
- Priorities?

Process states



Process states



OS keeps the PCB for the process, so that parent process can wait().

Process termination

Time flows down

Parent

```
child_pid = fork();
  //Parent does some
  //stuff, and then
  //does this . . .
int child_status;
waitpid(child_pid,
        &child_status);
```

```
  //waitpid() returns!
printf("%d", child_status);
  //Displays "42".
```

Child

```
//Child starts
//executing.
```

```
//Later…
```

```
exit(42);
```

In the state diagrams in this video, we assumed that a process can only reach the "Terminated" state from the "Running" state. Can you think a way that a process in the "Runnable" or "Waiting" state could transition to the "Terminated" state?