CHAPTER 6

Synchronization Tools

A **cooperating process** is one that can affect or be affected by other processes executing in the system. Cooperating processes can either directly share a logical address space (that is, both code and data) or be allowed to share data only through shared memory or message passing. Concurrent access to shared data may result in data inconsistency, however. In this chapter, we discuss various mechanisms to ensure the orderly execution of cooperating processes that share a logical address space, so that data consistency is maintained.

Bibliographical Notes

The mutual-exclusion problem was first discussed in a classic paper by [Dijkstra (1965)]. The critical-region concept was suggested by [Hoare (1972)] and by [Brinch-Hansen (1972)]. Dekker's algorithm (Exercise 6.13)—the first correct software solution to the two-process mutual-exclusion problem—was developed by the Dutch mathematician T. Dekker. This algorithm also was discussed by [Dijkstra (1965)]. A simpler solution to the two-process mutual-exclusion problem has since been presented by [Peterson (1981)] (Figure 6.3) A thorough discussion of memory barriers and cache memory is presented in [Mckenney (2010)]. [Herlihy and Shavit (2012)] presents details on several issues related to multiprocessor programming, including memory models and compare-and-swap instructions. [Bahra (2013)] examines nonblocking algorithms on modern multicore systems. The semaphore concept was suggested by [Dijkstra (1965)]. The monitor concept was developed by [Brinch-Hansen (1973)]. [Hoare (1974)] gave a complete description of the monitor. [Lu et al. (2008)] provide a study of concurrency bugs in real-world applications. The algorithm proposed in Exercise 6.10 is derived from [Treiber (1986)].

Bibliography

[Bahra (2013)] S. A. Bahra, "Nonblocking Algorithms and Scalable Multicore Programming", Volume 11, Number 5 (2013).

[Brinch-Hansen (1972)] P. Brinch-Hansen, “Structured Multiprogramming”, *Communications of the ACM*, Volume 15, Number 7 (1972), pages 574–578.

[Brinch-Hansen (1973)] P. Brinch-Hansen, *Operating System Principles*, Prentice Hall (1973).

[Dijkstra (1965)] E. W. Dijkstra, “Cooperating Sequential Processes”, Technical report, Technological University, Eindhoven, the Netherlands (1965).

[Herlihy and Shavit (2012)] M. Herlihy and N. Shavit, *The Art of Multiprocessor Programming* Revised First Edition, Morgan Kaufmann Publishers Inc. (2012).

[Hoare (1972)] C. A. R. Hoare, “Towards a Theory of Parallel Programming”, in **[Hoare and Perrott 1972]** (1972), pages 61–71.

[Hoare (1974)] C. A. R. Hoare, “Monitors: An Operating System Structuring Concept”, *Communications of the ACM*, Volume 17, Number 10 (1974), pages 549–557.

[Lu et al. (2008)] S. Lu, S. Park, E. Seo, and Y. Zhou, “Learning from mistakes: a comprehensive study on real world concurrency bug characteristics”, *SIGPLAN Notices*, Volume 43, Number 3 (2008), pages 329–339.

[Mckenney (2010)] P. E. Mckenney, “Memory Barriers: a Hardware View for Software Hackers” (2010).

[Peterson (1981)] G. L. Peterson, “Myths About the Mutual Exclusion Problem”, *Information Processing Letters*, Volume 12, Number 3 (1981).

[Treiber (1986)] R. K. Treiber, “Systems Programming: Coping with Parallelism” (1986).