CHAPTER 11

Introduction to Multiprocessors

Having covered the essential issues in the design and analysis of uniprocessors and pointing out the main limitations of a single-stream machine, we begin in this chapter to pursue the issue of multiple processors. Here a number of processors (two or more) are connected in a manner that allows them to share the simultaneous execution of a single task. The main argument for using multiprocessors is to create powerful computers by simply connecting many existing smaller ones. A multiprocessor is expected to reach a faster speed than the fastest uniprocessor. In addition, a multiprocessor consisting of a number of single uniprocessors is expected to be more cost-effective than building a high-performance single processor. An additional advantage of a multiprocessor consisting of n processors is that if a single processor fails, the remaining fault-free $n - 1$ processors should be able to provide continued service, albeit with degraded performance. Our coverage in this chapter starts with a section on the general concepts and terminology used. We then point to the different topologies used for interconnecting multiple processors. Different classification schemes for computer architectures are then introduced and analyzed. We then introduce a topology-based taxonomy for interconnection networks. Two memory-organization schemes for MIMD (multiple instruction multiple data) multiprocessors are also introduced. Our coverage in this chapter ends with a touch on the analysis and performance metrics for multiprocessors. It should be noted that interested readers are referred to more elaborate discussions on multiprocessors in Chapters 2 and 3 of our book on Advanced Computer Architecture and Parallel Processing (see reference list).

11.1. INTRODUCTION

A multiple processor system consists of two or more processors that are connected in a manner that allows them to share the simultaneous (parallel) execution of a given computational task. Parallel processing has been advocated as a promising approach for building high-performance computer systems. Two basic requirements are inevitable for the efficient use of the employed processors. These requirements

Fundamentals of Computer Organization and Architecture, by M. Abd-El-Barr and H. El-Rewini
ISBN 0-471-46741-3

are (1) low communication overhead among processors while executing a given task and (2) a degree of inherent parallelism in the task.

A number of communication styles exist for multiple processor networks. These can be broadly classified according to (1) the communication model (CM) or (2) the physical connection (PC). According to the CM, networks can be further classified as (1) multiple processors (single address space or shared memory computation) or (2) multiple computers (multiple address space or message passing computation). According to PC, networks can be further classified as (1) bus-based or (2) network-based multiple processors. Typical sizes of such systems are summarized in Table 11.1.

The organization and performance of a multiple processor system are greatly influenced by the interconnection network used to connect them. On the one hand, a single shared bus can be used as the interconnection network for multiple processors. On the other hand, a crossbar switch can be used as the interconnection network. While the first technique represents a simple easy-to-expand topology, it is, however, limited in performance since it does not allow more than one processor/memory transfer at any given time. The crossbar provides full processor/memory distinct connections but it is expensive. *Multistage interconnection networks* (MINs) strike a balance between the limitation of the single, shared bus system and the expense of a crossbar-based system. In a MIN more than one processor/memory connection can be established at the same time. The cost of a MIN can be considerably less than that of a crossbar, particularly for a large number of processors and/or memories. The use of multiple buses to connect multiple processors to multiple memory modules has also been suggested as a compromise between the limited single bus and the expensive crossbar. Figure 11.1 illustrates the four types of interconnection networks mentioned above. Interested readers are referred to our book on Advanced Computer Architecture and Parallel Processing (see reference list).

11.2. CLASSIFICATION OF COMPUTER ARCHITECTURES

A classification means to order a number of objects into categories, each having common features, among which certain relationship(s) exist(s). In this regard, a classification scheme for computer architectures aims at categorizing them such that those architectures that have common features fall into one category and such that different categories represent distinct groups of architectures. In addition,

TABLE 11.1 Typical Sizes of Some Multiprocessor Systems

Category	Subcategories	Number of processors
Communication model	Multiple processors	2–256
	Multiple computers	8–256
Physical connection	Bus-based	2–32
	Network-based	8–256

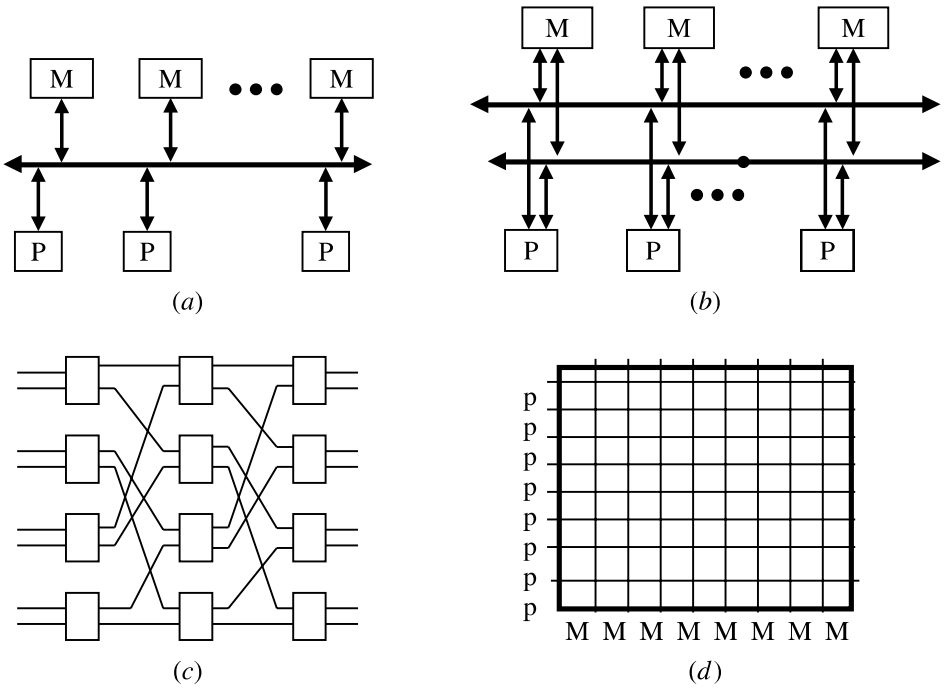

Figure 11.1 Four types of multiprocessor interconnection networks. (*a*) Single-bus system, (*b*) multi-bus system, (*c*) multi-stage interconnection network, (*d*) crossbar system

a classification scheme for computer architecture should provide a basis for information ordering and a basis for predicting the features of a given architecture.

Two broad schemes exist for computer architecture classification. The first is based on external (morphological) features of architectures and the second is based on the evolutionary features of architectures. The first scheme emphasizes the finished form of architectures, while the second scheme emphasizes the way an architecture has been derived from its predecessor and suggests speculative views on its successor. Morphological classification provides a basis for predictive power, while evolutionary classification provides a basis for better understanding of architectures. Examining the extent to which a classification scheme is satisfying its stated objective(s) could assess the pros and cons of that scheme.

A number of classification schemes have been proposed over the last three decades. These include the Flynn's classification (1966), the Kuck (1978), the Hwang and Briggs (1984), the Erlangen (1981), the Giloi (1983), the Skillicorn (1988), and the Bell (1992). A number of these are briefly discussed below.

11.2.1. Flynn's Classification

Flynn's classification scheme is based on identifying two orthogonal streams in a computer. These are the instruction and the data streams. The instruction stream is defined as the sequence of instructions performed by the computer. The data stream is defined as the data traffic exchanged between the memory and the processing unit. According to Flynn's classification, either of the instruction or data streams can be single or multiple. This leads to four distinct categories of

computer architectures:

1. Single-instruction single-data streams (SISD)
2. Single-instruction multiple-data streams (SIMD)
3. Multiple-instruction single-data streams (MISD)
4. Multiple-instruction multiple-data streams (MIMD)

Figure 11.2 shows the orthogonal organization of the streams according to Flynn's classification.

Schematics for the four categories of architectures resulting from Flynn's classification are shown in Figure 11.3. Table 11.2 lists some of the commercial machines belonging to each of the four categories.

Observations on Flynn's Classification

1. Flynn's classification is among the first of its kind to be introduced and as such it must have inspired subsequent classifications.
2. The classification helped in categorizing architectures that were available and those that have been introduced later. For example, the introduction of the SIMD and MIMD machine models in the classification must have inspired architects to introduce these new machine models.
3. The classification stresses the architectural relationship at the memory-processor level. Other architectural levels are totally overlooked.
4. The classification stresses the external (morphological) features of architectures. No information is included on the revolutionary relationship of architectures that belong to the same category.
5. Owing to its pure abstractness, no practically viable machine has exemplified the MISD model introduced by the classification (at least so far). It should, however, be noted that some architects have considered pipelined machines (and perhaps systolic-array computers) as examples for MISD.
6. A very important aspect that is lacking in Flynn's classification is the issue of machine performance. Although the classification gives the impression that machines in the SIMD and the MIMD are superior to their SISD and MISD counterparts, it gives no information on the relative performance of SIMD and MIMD machines.

		Data	Stream
		Single	Multiple
Instruction	Single	SISD	SIMD
Stream	Multiple	MISD	MIMD

Figure 11.2 Flynn's classification

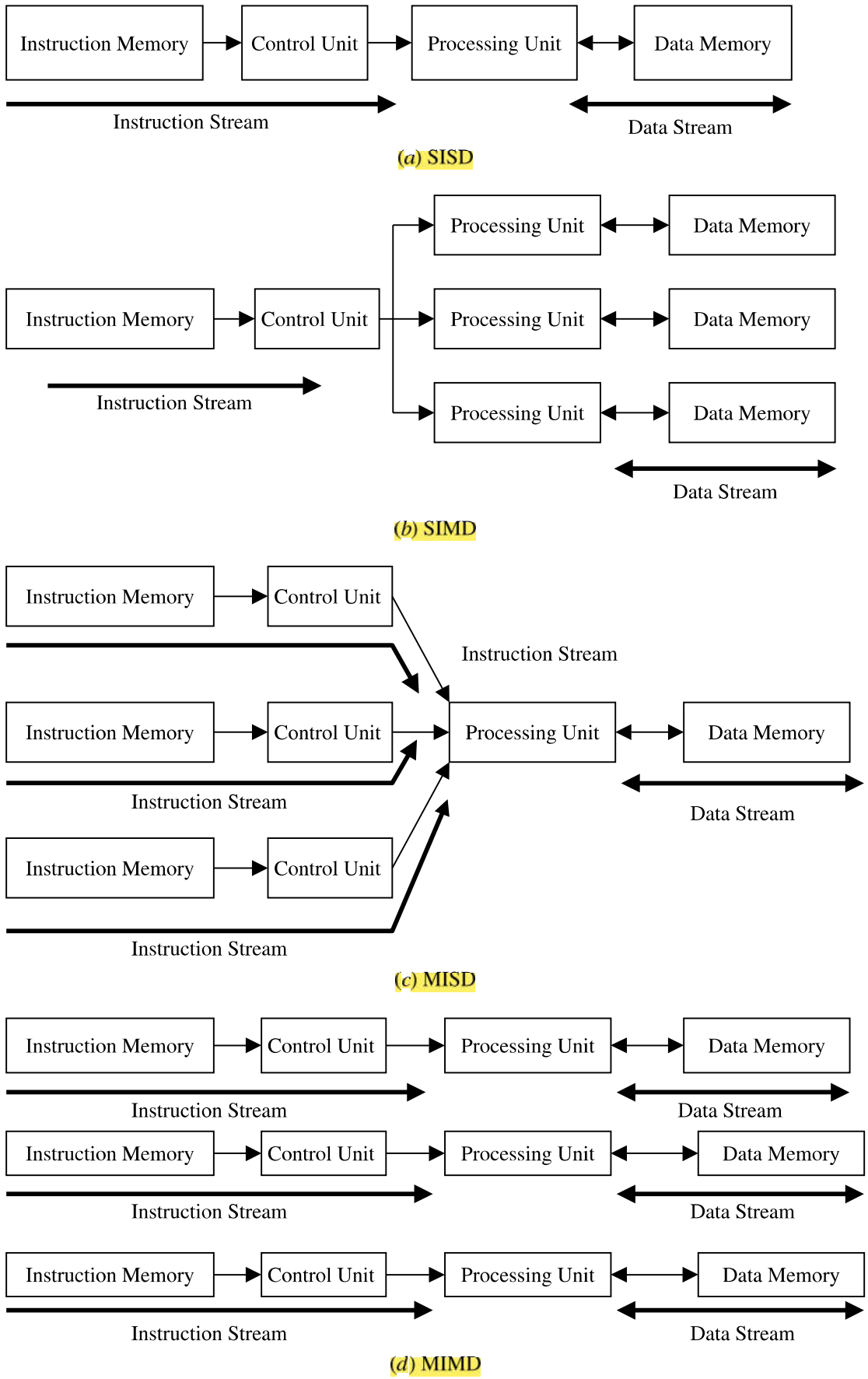

Figure 11.3 The four architecture classes resulting from Flynn's taxonomy. (*a*) SISD, (*b*) SIMD, (*c*) MISD, (*d*) MIMD

11.4.1. Shared Memory Organization

There has been recent growing interest in distributed shared memory systems. This is because shared memory provides an attractive conceptual model for interprocess interaction even when the underlying hardware provides no direct support. A shared memory model is one in which processors communicate by reading and writing locations in a shared memory that is equally accessible by all processors. Each processor may have registers, buffers, caches, and local memory banks as additional memory resources.

A number of basic issues in the design of shared memory systems have to be taken into consideration. These include access control, synchronization, protection, and security. Access control determines which process accesses are possible to which resources. Access control models make the required check for every access request issued by the processors to the shared memory, against the contents of the access control table. The latter contains flags that determine the legality of each access attempt. If there are access attempts to resources, then until the desired access is completed, all disallowed access attempts and illegal processes are blocked. Requests from sharing processes may change the contents of the access control table during execution. The flags of the access control with the synchronization rules determine the system's functionality. Synchronization constraints limit the time of accesses from sharing processes to shared resources. Appropriate synchronization ensures that the information flows properly and ensures system functionality. Protection is a system feature that prevents processes from making arbitrary access to resources belonging to other processes. Sharing and protection are incompatible; sharing allows access, whereas protection restricts it.

Running two copies of the same program on two processors will decrease the performance relative to that of a single processor, due to contention for shared memory. The performance degrades further as three, four, or more copies of the program execute at the same time.

A shared memory computer system consists of (1) a set of independent processors, (2) a set of memory modules, and (3) an interconnection network. The simplest shared memory system consists of one memory module (M) that can be accessed from two processors P_a and P_b (Fig. 11.8). Requests arrive at the memory module through its two ports. An arbitration unit within the memory module passes requests through to a memory controller. If the memory module is not busy and a single request arrives, then the arbitration unit passes that request to the memory controller and the request is satisfied. The module is placed in the busy state while a request is being serviced.

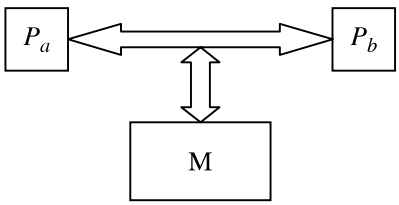

Figure 11.8 A simple shared memory scheme