BOOK REVIEWS

Recently published books and new periodicals may be submitted for review to the Book Reviews Editor:

Dr. Francis P. Mathur
Professor and Computer Science Coordinator
Mathematics Department
California State Polytechnic University
3801 West Temple Avenue
Pomona, CA 91768
Telephone: (714) 598-4421

Note: Publications reviewed in this section are not available from the IEEE Computer Society; they must be ordered directly from the publisher. To request ordering information, circle the appropriate number on the Reader Service Card.

B79-8 *Computer Architecture and Organization*—John P. Hayes (New York: McGraw-Hill, 1978, $22.00, 498 pp.)

This is a textbook for electrical engineering and computer science courses at the advanced undergraduate and beginning graduate levels. The author presents a comprehensive treatment of computer architecture that is generally consistent with the recommendations of the IEEE Computer Society Task Force on Computer Architecture.[1] The book contains six chapters with examples, many figures, and extensive homework exercises. Course instructors can obtain a solutions manual from the publishers.

Chapter 1 is one of the most interesting introductory chapters I have seen in a computer architecture text. Hayes stimulates the reader by touching briefly on the "nature of computers," i.e., by discussing such automata theory topics as simple abstract models of computation and Turing machines, computable functions, unsolvable problems, halting problems, finite-state machines, time and space complexity, tractable and intractable problems, and computer limitations. He continues with a comprehensive history of computers, progressing through the mechanical era (1623 to 1945), the first generation (1946 to 1954), the second generation (1955 to 1964), and the third generation, which includes modern processors and microprocessors. The section on third-generation computers examines the minicomputer, microprocessor, microcomputer, microprogramming, virtual memory, the supervisor state vs. problem state issue, and the concept of a hardware/firmware family architecture such as the S/360.

Also included is a history of computer architecture, particularly excellent because the author identifies basic concepts and charts their evolution. (He even traces register-transfer language to its use by Babbage in the mid 1800's.) A special section is devoted to the development of the stored-program and single-address instruction concepts. The chapter ends with a summary of the major architectures of the four computer generations.

Chapter 2 introduces the reader to design methodology at the gate, register, and processor levels. At the gate level, the author reviews some basics, including the relationships between mathematical graphs and block diagrams, structure and behavior, and state tables and state diagrams. He provides good formal definitions of "control section" and "processing section" and a brief review of combination logic, sequential logic, gates, and flip-flops. There is no discussion of Karnaugh maps.

The register-level discussion introduces MSI functions, register-transfer language, and sequential components such as registers, counters, and buses. Included is a good discussion of the processing section, the control section, and control points. The processor-level discussion covers processors, memories, I/O, switching networks, handshaking and asynchronous control, introduction to performance evaluation, simulation, queuing models, and Bell and Newell's PMS notation.

Chapter 3 deals with processor design. It starts with processor organization, including the basic CPU and its extensions, and goes on to information representation and number formats. Instruction types are discussed along with typical addressing options. The coverage of fixed-point arithmetic operations includes basic algorithms as well as high-speed and special-purpose algorithms. The author also treats floating-point arithmetic, and addresses ALU design by comparing the structure of a typical fixed-point ALU with the architecture of the Intel 3002 2-bit slice. The chapter concludes with an examination of parallel processing and such multi-unit processors as the CDC 6600 and Illiac IV.

Chapter 4 discusses control design, with emphasis on two approaches to implementation—hardware and microprogramming. The hardware discussion is brief; the microprogramming discussion is in detail. The section on hardwired control discusses and applies three design methods: state table (the classical switching-theory

method); delay element (often called "programmed logic design" and easy to implement from flow charts); and sequence counter (utilizing MSI components). The author concludes by applying all three techniques to a hardwired control for a 2's-complement multiplier.

Most of Chapter 4 is devoted to microprogrammed control. Microprogram interpretation and emulation, microinstruction width, parallelism, encoded and decoded fields, horizontal and vertical classification, address specification, and conditional branching are discussed. Algorithms for minimizing microinstruction size are also covered. The author applies microprogramming to the same multiplier control unit used in the hardware design example.

The last part of the chapter is devoted to microprogrammed computers. The author describes a simple microprogrammed CPU and shows how easy it is to add a new macroinstruction, which requires new control lines in addition to the microprogram. He introduces microprogram sequencers, treats the AM2901 sequencer in detail, and points out that both CPU and control sequencers can be bit-sliced. The chapter concludes with a discussion of two commercially available microprogrammed minicomputers: the Hewlett-Packard 21MX and the Nanodata QM-1. There is a good treatment of the HP21MX architecture and an emulation of the simple control unit presented earlier. The author contrasts the "conventional" HP21MX with the extremes of microprogramming in the QM-1. (The latter has two levels of microprogramming in which a microprogram is interpreted by a nanoprogram.) The microprogramming is vertical (an 18-bit instruction containing an opcode and two addresses) whereas the nanoprogramming is horizontal (a 360-bit instruction). The firmware thus has a great deal of parallelism and control over the buses.

Chapter 5 treats memory. The section on memory organization covers the architecture of memory systems, hierarchical memory systems, memory-device characteristics, and storage-device technology. The chapter describes random-access memory technology, including magnetic cores, and serial-access devices, including magnetic bubbles and CCDs. The section on virtual memory takes up software architecture, with a good general treatment of memory hierarchies covering locality of reference, cost and performance, address mapping, relocation mechanisms, main-memory allocation, segments and pages, replacement policies, and file organization. The section on high-speed memory includes interleaved, cache, and associative memory. The performance of interleaved memories is also evaluated.

Chapter 6 treats system organization, communication, input-output systems, and multiple-CPU systems. The section on system organization discusses I/O implementation, multiprocessors, and how computers are interconnected and communicate at the processor level. The section on communication describes local and long-distance communication, buses, synchronous and asynchronous bus communication, priority, and polling. The PDP-11 Unibus is examined in detail, but the IEEE-488 is omitted.

The section on input-output covers programmed I/O thoroughly, including memory-mapped I/O, isolated I/O, and I/O instructions. It compares the range of I/O from microprocessors to big machines, including I/O processors, 360-370 channel programming, I/O scheduling, and performance evaluation of I/O-driven systems. The section on multiple-CPU systems defines multiprocessors, computer networks, and fault-tolerant computers. The examples of redundancy in fault-tolerant computers are well chosen. The discussion of computer networks includes the Arpanet as well as the history of Sage and Sabre.

This book covers the computer architecture spectrum. Its historical treatment, comparative discussions, and performance evaluation analyses are extremely interesting. Chapter 4's architecture material compares well with the sort of material usually presented in separate, digital-system design courses emphasizing MSI, LSI, and microprocessor design. Some instructors may want more detail about microprogramming and control-hardware design than they will find here. In that case, they can supplement the text with the details of a specific local, or favorite, architecture.

Javin M. Taylor
New Mexico State University,
Las Cruces

1. G. E. Rossman, et al., "A Course Study in Computer Hardware Architecture," *Computer*, Vol. 8, No. 12, Dec. 1975, pp. 44-63.

Reader Service Number 46