RISC/CISC

DEVELOPMENT AND TEST SUPPORT

MARVIN HOBBS

RISC/CISC Development and Test Support

Marvin Hobbs

PRENTICE HALL
Englewood Cliffs, New Jersey 07632

Library of Congress Cataloging-in-Publiction Data

Hobbs, Marvin.
RISC/CISC development and test support / Marvin Hobbs.
p. cm.
Includes bibliographical references and index.
ISBN 0-13-388414-7
1. RISC microprocessors. 2. CISC microprocessors. I. Title.
TK7895.M5H62 1992
621.39'16—dc20 90-26110
CIP

Acquisition Editor: Karen Gettman
Editorial supervision: Cyndy Lyle Rymer
Cover Design: Karen Stephens
Prepress buyer: Kelly Behr
Manufacturing buyer: Susan Brunke

A Simon & Schuster Company
Englewood Cliffs, New Jersey 07632

Printed in the United States of America
10 9 8 7 6 5 4 3 2 1

ISBN 0-13-388414-7

Prentice-Hall International (UK) Limited, *London*
Prentice-Hall of Australia Pty. Limited, *Sydney*
Prentice-Hall Canada Inc., *Toronto*
Prentice-Hall Hispanoamericana, S.A., *Mexico*
Prentice-Hall of India Private Limited, *New Delhi*
Prentice-Hall of Japan, Inc., *Tokyo*
Simon & Schuster Asia Pte. Ltd., *Singapore*
Editora Prentice-Hall do Brasil, Ltda., *Rio de Janeiro*

RISC/CISC Development and Test Support

Preface

This book is about the development and test support tools and techniques for RISC and CISC 32-bit microprocessors. During the past few years the quest for higher-speed microprocessors has resulted in the development of several versions of RISC architecture. The CISC architecture has also seen considerable development as its advocates competed with the RISC processors to protect their software base. While RISC microprocessors are capable of higher-speed performance than CISC microprocessors, the latter are competing very well in some applications. There is little doubt that both types will be around for the foreseeable future. The development of both types has placed new demands on development and test support tools. In addition, built-in development and self-testing within the chips will impact development and test tools in the future. This book covers the role of in-circuit emulators, logic analyzers, and software simulators in the development phase and the role of in-circuit testers and emulators in the test phase. The approaches taken to built-in development support and built-in self-testing are covered.

Chapter 1 reviews the role of in-circuit emulation (ICE) in dedicated and universal development systems. In the universal role ICE can be used with a wide range of computer hosts, including personal types. The role of logic analyzers is seen to be rising for both RISC and CISC applications and

to be serving as an important supplement to either emulation or simulation tools. Design automation as realized by the combination of CASE, electronic CAE/CAD, and CAD/CAM is covered as one approach. It is seen that built-in development support can play a significant role in simplifying external tools. In-circuit and functional testing is shown to be on the threshold of change due to built-in chip test features, such as boundary scan.

Chapters 2 and 3 cover RISC and CISC microprocessors. In Chapter 2 the evolution of RISC architecture from the universities to industry is reviewed. RISC principles are detailed. The SPARC and RISC architectures are compared. The types comprising the Motorola 88000 family are covered. The features of Intel's 80860, AMD's 29000, and VLSI Technology's VL86C010 and VL86C020 are detailed. In Chapter 3 the evolution of Intel's 80X86 and Motorola's 68000 families is reviewed. The general characteristics of the Intel and Motorola 32-bit CISC microprocessors are described. These include Intel's 80386 and Motorola's 68020 and 68030. Early applications of higher clock frequencies are described. It is seen how Intel studied the RISC technology and applied what they learned to CISC designs, such as the 80486, to compete with RISC. The features of Motorola's 32-bit CISC microcontroller are covered in detail.

In Chapter 4 the development system used by Bell Laboratories to support AT&T's 32100 and 32200 CISC microprocessors is described in detail. Although these microprocessors have been used almost exclusively in AT&T systems, this development system is a good example of one that can be used to support the development of microprocessor-based products and systems to be implemented by CISC microprocessors operating at clocks up to at least 25 MHz. The use of in-circuit emulation and the hardware of the analysis pod are covered. The four different environments in which this development system can be used are described in detail.

In Chapter 5 logic analyzer operation and its use in hardware and software debugging are covered. The use of a logic analyzer with a device events monitor to support applications of AT&T microprocessors operating up to clock frequencies of 18 MHz is described. Those who used it say that it operated satisfactorily up to 25 MHz. The in-circuit debugger for Intel's 80486 is shown to be closely associated with high-speed logic analyzers. A C language source code debugger is described. Analyzers have distinct advantages over emulators in the development of high-speed systems. A logic analyzer application to Motorola's 88000 RISC microprocessor is described.

Chapter 6 shows how software simulation has been used to support the application of SPARC, MIPS, and Motorola's 88000 RISC microprocessors. It is demonstrated that the hardware development relies on logic analyzers.

Chapter 7 covers evaluation and development boards. A typical evalution board used by AT&T in the application of its 32-bit CISC microprocessors is described. Detailed coverage is given to a SPARC evaluation

board used by Cypress Semiconductor. A special feature, called stretched clock logic, is a part of this board. An evaluation board for the MIPS R3000 RISC microprocessor is also described in similar detail. The relationship between the CPU and memory is emphasized.

Chapter 8 deals with in-circuit emulators in detail. The Hewlett-Packard 647000 series is covered with a discussion of the problems posed by the operation of microprocessors at higher clock frequencies. An emulation system for ASIC verification is detailed. The methods of using in-circuit emulation in functional board testing are covered. They are CPU emulation, memory emulation, and bus cycle emulation.

Chapter 9 covers the software tools associated with the development of both RISC- and CISC-based products and systems. The development cycle environment for SPARC RISC microprocessors is covered first. Optimizing compilers for both SPARC and MIPS microprocessors are covered in some detail. The software tools used in the application of Intel's 80960 RISC microcontroller are described. The requirements and main features of debuggers follow. This chapter concludes with a listing of the support software tools for a 32-bit CISC microprocessor.

Chapter 10 reviews the built-in development support features of Intel's 80386 and 80486. The extensive trace features of their 80960KB microcontroller are detailed. The evolution of development support in Motorola's 68000 family is shown to lead to the rather extensive built-in support in their 68332 microcontroller. The built-in debugging support of Intel's RISC chip, the 80860, is discussed briefly, and it is indicated that built-in development support is not a feature of most RISC chips at this time.

Chapter 11 covers boundary scan, a design for testability (DFT) technique proposed by the Joint Test Action Group (JTAG). It has been adopted as IEEE Standard P1149.1. The material in this chapter is largely based on the JTAG proposal.

In Chapter 12 Texas Instruments is cited as the first manufacturer to offer a range of products compatible with boundary scan testing. More detail is given about the built-in self-test (BIST) features of Intel's 80486 microprocessor and Motorola's 68332 microcontroller. It is shown that they are not compatible with boundary scan but nevertheless represent rather extensive implementations of BIST techniques. The boundary scan mode of Intel's 80860 RISC microprocessor is described in detail.

This book is for the electronics engineer, whose responsibility is to combine the software and hardware to realize a working system or product. The engineer does not necessarily need to be trained in computer science, but an understanding of its basics would be helpful. Training in the theory and operation of microprocessors is necessary, and some experience would be helpful. This book should be of particular interest to electronics engineers who have decided to specialize in microprocessor applications at all levels, whether in college or industry.

Acknowledgments

This book has been made possible by the support of leading companies in the field of microprocessor development and test technology. These include the Intel Corporation, Hewlett-Packard Company, American Telephone and Telegraph, Cypress Semiconductor, Ross Technology, Motorola, Inc., Integrated Device Technology, Inc., MIPS Computer Systems, Sun Microsystems, Inc., Fujitsu, Talon Instruments, Advanced Micro Devices, Tektronics, Inc., Quickturn Systems, Inc., and John Fluke Mfg. Co.

Societies and publications in the field have also been very helpful. These include the Institute of Electrical and Electronics Engineers, BUSCON, MICRO, the IEEE *Spectrum, Personal Engineering and Instrumentation News, Electronic Engineering Times*, and Dempa Publications.

Contents

Chapter 1

Introduction

For the next decade the development of tools to support the design and testing of microprocessor-based products and systems will be driven mainly by two trends—one, the rise in clock frequencies, and two, the increase in the number of transistors on a chip. A projection of the increase in clock frequencies over the next decade from 33 MHz to 250 MHz is shown in Figure 1-1, and the projected increase of transistors per chip from about 1 million to possibly 50 million is shown in Figure 1-2. Such projections are not being made only by dreamers but by the R&D management of leading semiconductor manufacturers.

The rise in clock frequencies has already had a significant impact on the design of in-circuit devices such as emulators, debuggers, and interface pods. Simulation with logic analyzers has been favored by some as an alternate in many high-speed applications.

The increase in transistors per die will require the implementation of scan techniques, which will undoubtedly be necessary to successfully test so many transistors on a single chip. However, the placement of so many transistors on a chip will make it possible to have the equivalent of a "board-on-a-chip." Consequently, it will become much more difficult to access points within a chip and thus complicate development techniques as they are known today.

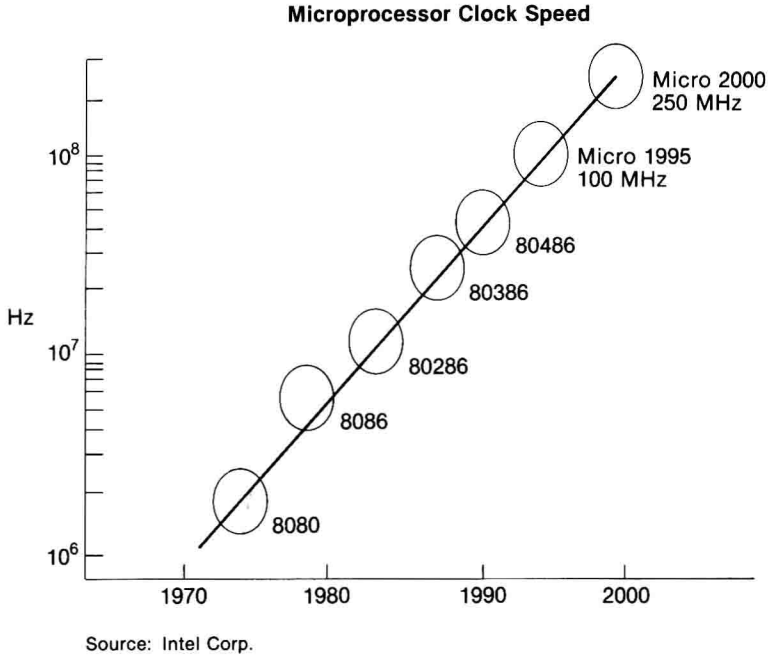

FIGURE 1-1
The projected rise of microprocessor clock speed. Reprinted courtesy of Intel Corporation.

In the meantime, tools and techniques that have been employed during the past decade will continue to be used in various forms as they evolve to keep pace with microprocessor development. These tools and techniques are covered in this book.

In its application, no microprocessor stands alone. It is either assembled on a circuit board or embedded into a product or system with other components essential to its function. Software and hardware are required to bring it into a viable entity. Software development may require the production of many thousands of words of code. Software and hardware development must be integrated. After the design has been finalized and put into production, it must be suitable for adequate testing to ensure the quality of the product in a competitive world.

Software development using high-level languages, such as C or Pascal, requires the basic tools shown in Figure 1-3. The editor is loaded, files are set up, and code is typed in. A compiler or cross-compiler delivers an object

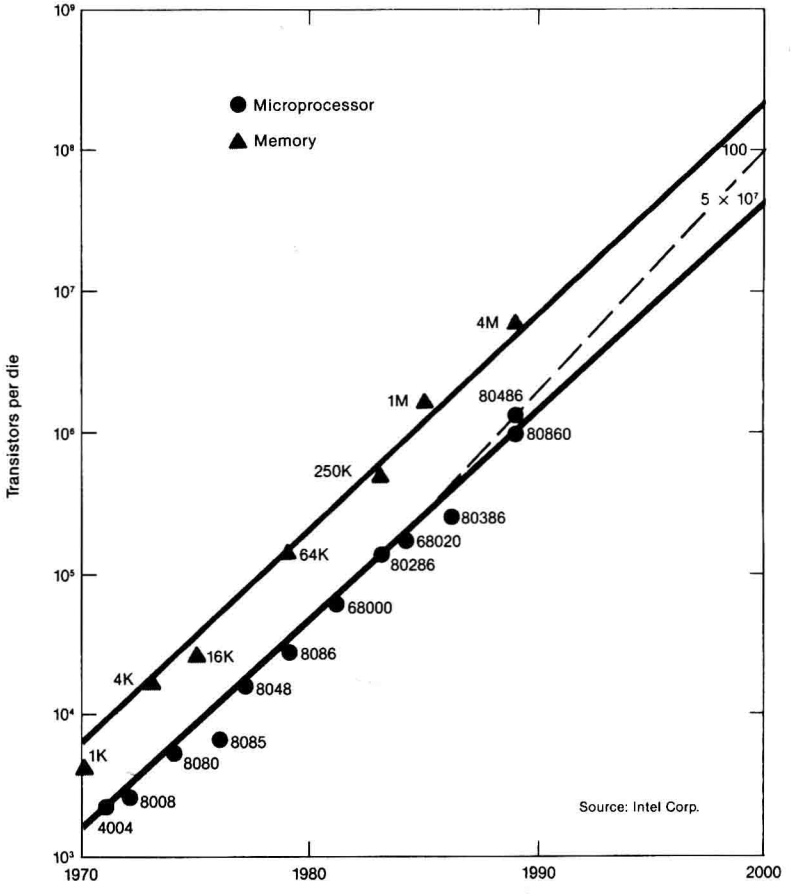

FIGURE 1-2
A projected increase of transistor density per chip. Patrick P. Gelsinger, Paolo A. Gargini, Gerard H. Parker and Albert Y.C. Yu, "Microprocessors circa 2000," *IEEE Spectrum*, Oct. 1989. © 1989 by IEEE.

file to a linker, which combines it with library files and the assembly code produced by an assembler, if any assembly language is used. Finally, the program is loaded into memory and run or downloaded into the target hardware. Such a program is not likely to run properly the first several times it is tried, and some form of debugger will be required to make it workable. All of these steps may be taken on a host computer (PC, workstation, or the like) before the hardware of the microprocessor-based product or system is developed or built.

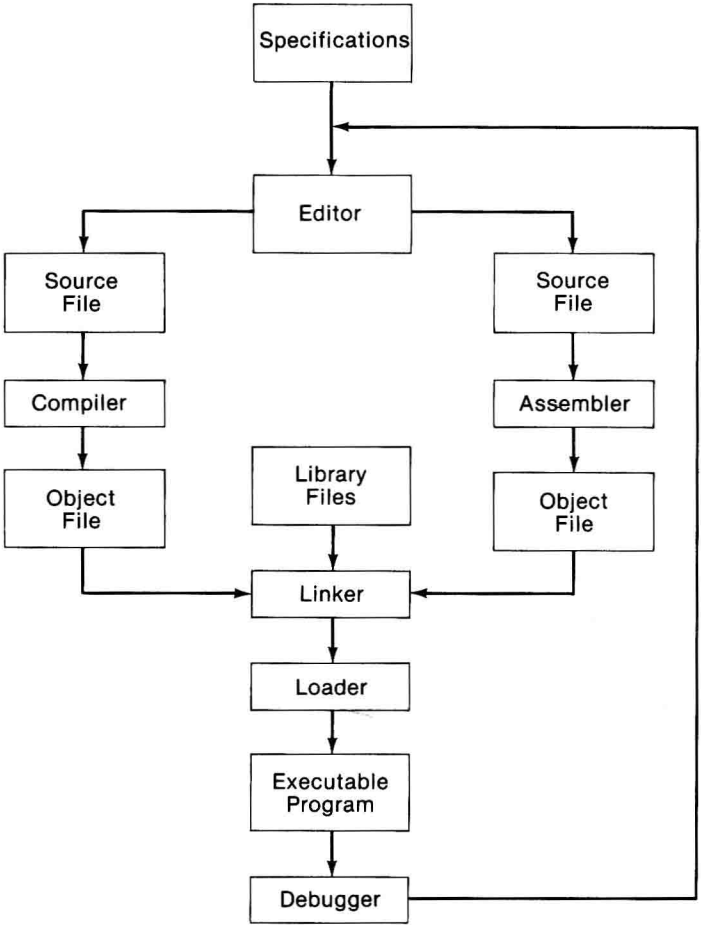

FIGURE 1-3
The basic software development process

The hardware design will almost always proceed in parallel with the system software development. It will be done using development tools such as CAE (computer-aided engineering) workstations, schematic capture packages, logic and circuit simulators, and PCB (printed circuit board) layout software. Once the hardware has been built, the first prototypes may be checked out with instruments such as waveform generators, scopes, and logic analyzers to ensure basic functionality. At this point hardware/software integration can take place. It is here that the designer has two basic approaches by which he or she can proceed. One of these might be called the hardware approach and the other the software approach.