COMPUTER PROGRAMMING AND ARCHITECTURE

The VAX-11

Henry M. Levy
Richard H. Eckhouse, Jr.

COMPUTER PROGRAMMING AND ARCHITECTURE

The VAX-11

Henry M. Levy
Richard H. Eckhouse, Jr.

digital
Digital Press

Printed in U.S.A.

1st Printing, April 1980

Documentation Number EY-AX008-DP-001

LIBRARY OF CONGRESS
CATALOGING IN PUBLICATION DATA

Levy, Henry M. 1952-
Computer Programming and Architecture - - The VAX-11

Bibliography: p.
Includes index
1. VAX-11 (Computer) - - Programming. 2. Assembler Language (Computer program language). 3. Computer architecture. 1. Eckhouse, Richard H., 1940- joint author. 2. Title.
QA 76.8.V37 001.64.'2 80-14409

ISBN 0-932376-07-X

Trademarks

Digital Equipment Corporation: DEC, DECUS, PDP, UNIBUS, VAX, DECnet, DECsystem-10, DECSYSTEM 20, DECwriter, DIBOL, EduSystem, IAS, MASSBUS, PDT, RSTS, RSX, VMS, VT.

IBM: System 360/370, Series 1.

Control Data Corporation: CYBER.

In memory of Samuel J. Levy

Foreword

Understanding today's computers requires a systems viewpoint. Relentless advances in semiconductor components open the opportunity to integrate higher levels of the systems architecture into the basic hardware. Thus, we should not approach computer structures without examining software needs.

This text is unique in addressing hardware structures and assembly language programming, while also describing the interfaces and mechanics of an operating system; it introduces the full range of fundamental hardware and software structures. Another noteworthy aspect of this book is its use of a practical, modern computer system that contains process support as well as virtual memory. Using the VAX-11, the authors have raised a host of topics and problems encountered by programmers and operating systems.

I highly recommend this book to those looking for a readable, practical introduction to the fundamentals of hardware and software computer structures.

Samuel H. Fuller
Technical Director
Digital Equipment Corporation

Preface

This book is for those who wish to understand the architecture and operation of computer systems. We believe that the best way to understand a computer's architecture is to use it, and the best way to use the architecture is to program at the assembly level. Once the basic assembly language concepts of addressing and instruction execution are mastered, one can begin to consider more advanced concepts such as data structures, Input/Output programming, and features for operating system resource management. The operating system support features, however, are a part of the architecture seen by the operating system, as opposed to the programmer-visible interface.

Therefore, this book is divided into two parts. The first half of the book, Chapters 1 through 6, is concerned with the architecture of a computer as seen by the assembly language programmer. The reader is first presented with the basics of computer organization and arithmetic. More complex concepts, including data-types and data structures, are then developed along with their manipulation by assembly language programming. The computer used to illustrate the concepts discussed is the VAX-11 manufactured by Digital Equipment Corporation.

Chapter 1 begins with a discussion of the differences between architecture and implementation. It also presents a brief review of number systems.

Chapter 2 introduces basic computer structures: memories, processors, and I/O devices. It first covers the machine-independent concepts of memory addressing, instruction execution cycles, and data representation, proceeding to describe the VAX-11 and its data-types and instructions. By the end of the chapter, the reader should be able to code simple machine language instructions using simple addressing.

Chapter 3 presents more advanced addressing and instruction techniques. The reader is shown how VAX-11 instructions are represented in memory. At this stage, the reader should then be able to code small routines using more complex instructions and varied addressing modes.

Chapter 4 develops more advanced control structures such as loops, subroutines, and stacks for storage and linkage. The use of macros to simplify assembly programming is also described.

Chapter 5 concludes the VAX-11 assembly language section with more sophisticated instructions and the manipulation of complex data structures such as lists, queues, and trees. By this point, the reader should thoroughly understand how to manipulate data types and data structures to solve problems.

Chapter 6 allows the reader to contrast his or her general knowledge of the VAX-11 architecture to that of three other architectures: the IBM System 370, the CDC Cyber, and the IBM Series 1. The material in this chapter focuses on the instruction encoding and memory addressing of these machines. In addition, the material provides some insight into the architectural tradeoffs made in designing each one.

The second half of the book, Chapters 7 through 11, considers the more sophisticated architectural support of an operating system and the strategies used by an operating system to manage hardware resources. These chapters examine that part of the architecture and implementation not usually seen by the applications programmer.

The VAX-11 physical I/O system is introduced in Chapter 7. The nature of I/O devices is explained, and simple examples of programmed I/O device control are presented.

Chapter 8 examines the architecture that supports the operating system. It develops the need for sharing of resources and deals with the VAX-11 process structure, the use of access modes, the implementation of virtual memory, and the handling of interrupts and exceptions.

In Chapter 9 we examine how an operating system uses the architectural support described in Chapter 8. The VAX/VMS operating system is used as an example, along with discussions of general operating system strategies.

Chapter 10 describes the interfaces and utilities provided for the user by an operating system.

Chapter 11 concludes with an examination of the implementation of a particular member of an architectural family. The discussion covers features transparent to the programmer, such as the cache, translation buffer, and instruction buffer. It shows the tradeoffs available to the hardware designer in producing a cost-effective implementation while still meeting the architectural constraints.

The book is intended for the programmer with some experience in a high level language, such as Pascal or FORTRAN. Of course, the language is not nearly as important as the ability to construct algorithms that yield computer programs. An understanding of basic data structures is also expected.

We do not intend to make the reader a sophisticated assembly language programmer. Rather, we expect the reader to emerge with a sound understanding of computer organization, memory addressing, program execution, and the fundamentals of one particular architecture. An awareness of the purpose of an operating system and the support it requires from the underlying hardware should also crystallize.

Therefore, while the story is told with the VAX-11 as the main character, we believe the book is generally applicable to the understanding of any computer system. The techniques developed should enable a programmer to quickly master any new machine encountered. It should also aid the programmer in assessing the strengths and weaknesses of a particular architecture relative to the VAX-11.

Acknowledgements

We would like to thank the reviewers for their valuable comments. In particular, Steve Beckhardt, Tom Dopirak, Sandy Kaplan, Art Karshmer, Larry Kenah, Ed Lazowska, Victor Lessor, and Carol Peters (along with a number of reviewers who remain anonymous) provided us with much valuable feedback. We also appreciate the support of the staff of Digital Press who were instrumental in starting us on the book. We would also like to thank Digital Equipment Corporation for its support, especially Gordon Bell, Jim Bell, Sam Fuller, and Bill Strecker. Finally, we would like to thank the VAX-11 architects, VAX-11/780 implementors, and VAX/VMS operating system group for making the last few years so interesting.

Hank Levy
Dick Eckhouse

Contents

PART ONE

THE USER ARCHITECTURE