

COMPUTER PROGRAMMING AND ARCHITECTURE

The VAX-11

Henry M. Levy
Richard H. Eckhouse, Jr.

TP31
L4

8360125

COMPUTER PROGRAMMING AND ARCHITECTURE

The VAX-11

Henry M. Levy
Richard H. Eckhouse, Jr.



E8360125

digital
Digital Press

Copyright ©1980 by Digital Equipment Corporation

All rights reserved. Reproduction of this book, in part or in whole, is strictly prohibited. For copy information, contact Digital Press, Educational Services, Digital Equipment Corporation, Bedford, Mass. 01730.

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, sub-routines, 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

Foreword	xv
Preface	xvii
Acknowledgements	xxi

PART ONE THE USER ARCHITECTURE 1

Chapter 1 Architecture and Implementation 3

Organization of the Book	5
Review of Number Systems	6
Number Systems	6
Binary and Hexadecimal Representation	8
Negative Numbers	9
References	11
Exercises for Chapter 1	11

Chapter 2 Computer Structures and Elementary VAX-11 Programming 13

Computer Structures	13
The Memory	15
The Central Processing Unit	17

Instruction Execution	18
The Input/Output System	20
Introduction to the VAX-11	21
VAX-11 Information Units and Data-Types	24
Integers	26
Floating-Point Numbers	27
Alphanumeric Characters	29
Decimal Strings	29
Data-Type Summary	32
Using Assembly Language	33
Assembler Statement Types	34
VAX-11 Instruction Format	35
The Functions of a Symbolic Assembler	39
The Location Counter	39
Symbols	40
Constants	41
Storage Allocation	42
Expressions	42
Control Statements	44
The Listing	45
The Assembly Process	46
Conventions for Writing Programs	48
Summary	51
References	52
Exercises for Chapter 2	52
Chapter 3 Instruction and Addressing Fundamentals	55
VAX-11 Instruction Characteristics	55
Generic Operations	56
Control Flow	58
Operand Addressing Techniques	62
Simple Addressing	63
Immediate Mode	66
General Purpose Registers	68
Indirect Mode	70
Autoincrement and Autodecrement Modes	73
Operand Context	76

Displacement Mode	77
Index Mode	80
Instruction Formats	82
General Instruction Format	83
Encoding an Instruction	83
Program Counter Relative Addressing	87
Immediate Addressing	89
Absolute Addressing	91
Branch Addressing	92
Summary	93
References	98
Exercises for Chapter 3	98

Chapter 4 More Advanced Programming Techniques 103

The Jump Instruction	103
Case Statements	105
Loops	107
The Stack	110
Subroutines and Procedures	116
Argument Lists and Call Instructions	118
The Argument Pointer	120
Saving Registers	121
Sample Procedure	122
The Call Frame	123
Local Variables	126
Fast Linkages	127
Recursion	129
Re-Entrant Routines	132
Macros	133
Creating Local Labels	135
Macro Calls Within Macro Definitions	136
Argument Concatenation	137
Repeat Blocks	138
Conditional Assembly	141
Summary	143
References	143
Exercises for Chapter 4	144

Chapter 5 Data-Types and Data Structures 147

- Bits and Bit Fields 148
 - Logical Bit Instructions 148
 - Single-Bit Instructions 149
 - Variable-Length Bit Fields 153
- Converting Integer Data-Types 155
- Character Strings 158
- Packed Decimal String Instructions 160
- Multiple Precision Integer Arithmetic 165
- Floating-Point Arithmetic 167
- Multi-Element Structures and Records 167
 - Arrays 170
 - Circular Lists 171
 - Linked Lists 174
 - Doubly Linked Lists 178
 - Trees 188
- Summary 192
- References 193
- Exercises for Chapter 5 193

Chapter 6 Comparative Architectures 197

- The IBM System 360/370 Instruction Set Architecture 197
- The Control Data Cyber Series Architecture 204
- The IBM Series 1 Instruction Set Architecture 211
- Summary 217
- References 217
- Exercises for Chapter 6 218

PART TWO THE SYSTEM ARCHITECTURE 219

Chapter 7 Physical Input and Output 221

- I/O Processing 222
- Control and Status Registers and I/O Space 224
- Low Speed Devices 225
 - The Line Printer 225
 - Terminal Multiplexing 228
- High Speed Devices 230
 - Magnetic Disks 231

Simplified Disk Control	232
Magnetic Tape	238
I/O Adapters	239
The Massbus Adapter	240
The Unibus Adapter	241
The Initial Bootstrap Problem	241
Console and Floppy	242
Summary	243
References	244
Exercises for Chapter 7	244

Chapter 8 The Support of an Operating System 245

Sharing the Processor	246
Sharing the Memory	247
Processes	253
Processor Access Modes	255
Process Access Mode Stacks	256
Changing Modes	257
Checking for Accessibility	259
Process Context Switching	259
Summary of Process Concepts	261
VAX-11 Memory Management	263
VAX-11 Memory Structure	263
VAX-11 Page Tables	265
VAX-11 Address Space Regions	268
System Space	269
Process Space	271
Privileged Registers	273
Summary of Memory Management Concepts	275
I/O Condition Handling	276
Interrupts and Exceptions	276
Interrupts	277
Exceptions	277
Vectors	280
Software Interrupts	280
Summary of I/O Condition Handling Concepts	282
Summary	283
References	283

Exercises for Chapter 8 283

Chapter 9 The Structure of a VAX-11 Operating System 287

Process Scheduling	287
VMS Process Scheduling	288
VMS Scheduler Context Switch Example	293
Process Paging	296
VMS Memory Management	300
Paging Under VMS	300
Swapping Under VMS	302
Input and Output Processing	303
The VMS I/O System	304
VMS I/O Data Base	305
VMS I/O System Components	308
I/O Control Flow	309
The Use of Interrupt Priority Levels	311
System Service Implementation	315
Summary	317
References	317
Exercises for Chapter 9	317

Chapter 10 The Operating System Interface 319

User-Level Interface: The Command Language	319
Communicating with the System	321
File Conventions	322
File Manipulation Commands	325
Informational Commands	326
Extending the Command Language	328
Program Development Software	330
The Editor	330
The Production of the Object File	332
The Linker	333
Debugging	335
The System Service Level Interface	336
VMS System Service Interface	336
System Service Macros	337
Status Return Codes	338
VMS System Services	339

Information Services	340
Memory Management Services	340
Process Control Services	341
Interprocess Communication	342
Event Flags	342
Mailboxes	343
Shared Memory	344
Combining Interprocess Communication Mechanisms	344
I/O Services	346
Handling Asynchronous Events	347
Summary	348
References	349
Exercises for Chapter 10	349
Chapter 11 The Efficient Implementation of an Architecture	351
Choice of Memory Technology and Structure	351
The Fastest Technology Approach	353
Cache Memory Approach	353
Associative Memories	354
Cache Memory on the VAX-11/780	356
The Translation Buffer	358
The Instruction Buffer	360
Instruction Accelerators	362
I/O Implementation	362
The Evolution of the VAX-11 Architecture	364
The Family Concept	365
Implementation Differences in the VAX-11/750	366
Summary	367
References	368
Exercises for Chapter 11	368
APPENDIXES	
Appendix A VAX-11 Instruction Set Description	371
Appendix B Abbreviations Used in the Text	384
Appendix C VAX/VMS Terminal Input/Output Routines	386
Bibliography	393
Index	397

PART ONE

THE USER ARCHITECTURE