COMPUTER ARCHITECTURE AND DESIGN

A. J. van de Goor

Cover designed by Crayon Design of Henley-on-Thames
and printed by The Riverside Printing Co (Reading) Ltd
Illustrations by Chartwell Illustrators.
Typeset by Columns of Reading.
Printed in Great Britain by The Bath Press, Avon

First printed 1989.

British Library Cataloguing in Publication Data
Goor, A. J. van de
Computer architecture and design.
1. Microprocessor systems. Design
I. Title
004.2'56

ISBN 0-201-18241-6

Library of Congress Cataloguing in Publication Data
Goor, A. J. van de.
Computer architecture and design.

Bibliography: p.
Includes index.
1. Computer architecture. 2. Microprocessors.
I. Title
QA76.9.A73G66 1989 004.2'2 88-16615
ISBN 0-201-18241-6

ELECTRONIC SYSTEMS ENGINEERING SERIES

Consulting Editor **E L Dagless**
University of Bristol

Special Consultant **D W Barron**
University of Southampton

OTHER TITLES IN THE SERIES

Advanced Microprocessor Architectures *L Ciminiera and A Valenzano*
Optical Pattern Recognition Using Holographic Techniques *N Collings*
Modern Logic Design *D Green*
Data Communications, Computer Networks and OSI (2nd Edn) *F Halsall*
Microwave Components and Systems *K F Sander*
Tolerance Design of Electronic Circuits *R Spence and R S Soin*

FOREWORD

Computer Architecture and Design is a fine introduction to today's computers. The book is based on the microprocessor, a component built on a single, very-large-scale integrated circuit with several hundred thousand transistors, but this is in no way limiting and in fact is likely to turn out very forward looking. In the next few years all but a few supercomputers and the ageing, traditional mainframes and mini-computers which serve as 'code museums' for the last three decades of programs and data will be built on the microprocessor.

According to the Preface, Professor van de Goor's book is aimed at introductory courses in computer architecture and design. I feel it goes well beyond this level given its approach, insight, breadth and extensive references. It should be useful to anyone engaged in the design of systems using microprocessors. While the book uses the Motorola 68000 microprocessor as the core, with extensions using the VAX architecture, an overall taxonomy for the various architectural dimensions is always provided to give the reader a full view of architecture.

The 68000 is a justifiable model to use since it is the core of three revolutionary machines: the workstation (Apollo and SUN), the finest personal computer (Apple Mac) and a fully fault-tolerant computer (Stratus). In addition, the structure of nearly all microprocessor-based systems today is around Motorola's VME bus, which is described along with the more elegant, but less used, Nubus. Justifiably absent is the I/O processor as embodied in the Intel and IBM 360 . . . 3090 Series channels – these structures, although complex, are not especially interesting or useful to anyone except their designers.

In five years, I suspect that virtually all important new computers will utilize a simple pipelined architecture – the RISC or reduced instruction set computer. Several new companies (for example, MIPS Computer) and nearly all semicomputer companies (for example, Motorola 88000) are introducing microprocessors using this approach. Chaper 15 provides the basis for understanding this shift.

I heartily recommend this book. All the important basic principles are introduced and described in a well-developed context by an experienced teacher and fine engineer who understands how to build computer systems.

C. Gordon Bell
Vice President R & D,
Ardent Computer

Trademark notice

AMD29000™ is a trademark of Advanced Micro Devices
ARM™ is a trademark of Acorn Computers
ARPANET
B1700™, B5000™, B5500™, B6700™ and B7700™ are trademarks of Burroughs
Clipper™ is a trademark of Fairchild
Cray-1™ and Cray Y-MP™ are trademarks of Cray
Cyber-205™ is a trademark of Control Data Corporation
DAP™ is a trademark of ICL
Futurebus™ is a trademark of IEEE
iAPX432™, iAPX286™, iAPX80386™, iPSC™, 80286™, 80287™, 80386™, 80387™, 80X86™ and Multibus II™ are trademarks of Intel Corporation
IBM 3033™, 3081™, 360™, 360/91™, 360/195™, 370™, 370/168™, 704™, 709™, 7090™, MS-DOS™, PC-RT™ and VM/370™ are trademarks of International Business Machines Corporation
MC68000™, 68010™, 68020™, 68030™, 68851™, 68881™, 68882™, 88000™ and DSP56000™ are trademarks of Motorola Corporation
Mostek 6502™ is a trademark of Mostek
NS32000™ and 32100™ are trademarks of National Semiconductor Corporation
Nubus™ and TMS32010™ are trademarks of Texas Instruments
PDP-8™, PDP-10™, PDP-11™, PDP-11/70™, System-10™, VAX-11™, VAX-11/780™ and Unibus™ are trademarks of Digital Equipment Corporation
Pyramid 90X™ is a trademark of Pyramid Company
Sequent™ is a trademark of Sequent Computer Systems Incorporated
Spectrum™ is a trademark of Hewlett-Packard
SPARC™ is a trademark of Sun Microsystems Inc.
transputer™ is a trademark of INMOS Group of Companies
UNIVAC I™ and 1100/80™ are trademarks of UNIVAC Systems
UNIX™ is a trademark of AT&T
uPD7720™ and uPD77230™ are trademarks of NEC
Z8000™ and Z8002™ are trademarks of Zilog Corporation

PREFACE

The advances achieved in computer systems have been very remarkable. Since the introduction of the first electronic computers in the 1940s, they have now reached a point of general acceptance to the extent that high school curricula include courses to familiarize students with the use of computers for applications such as text processing and database management. Courses in problem solving are also being offered, using programming languages such as BASIC or Pascal. It is safe to predict that many introductory college level computer science subjects will be included in the curricula of high schools.

This trend, however, causes an upward pressure on the material to be covered at the college level. The low price and wide acceptance of microprocessors creates a need for their architecture and organization to be covered in greater depth. The aim of this book is to allow computer science, computer engineering and electrical engineering students, and practising professionals, to master the terminology and concepts used to describe computers, and to provide them with a frame of reference for evaluation and comparison.

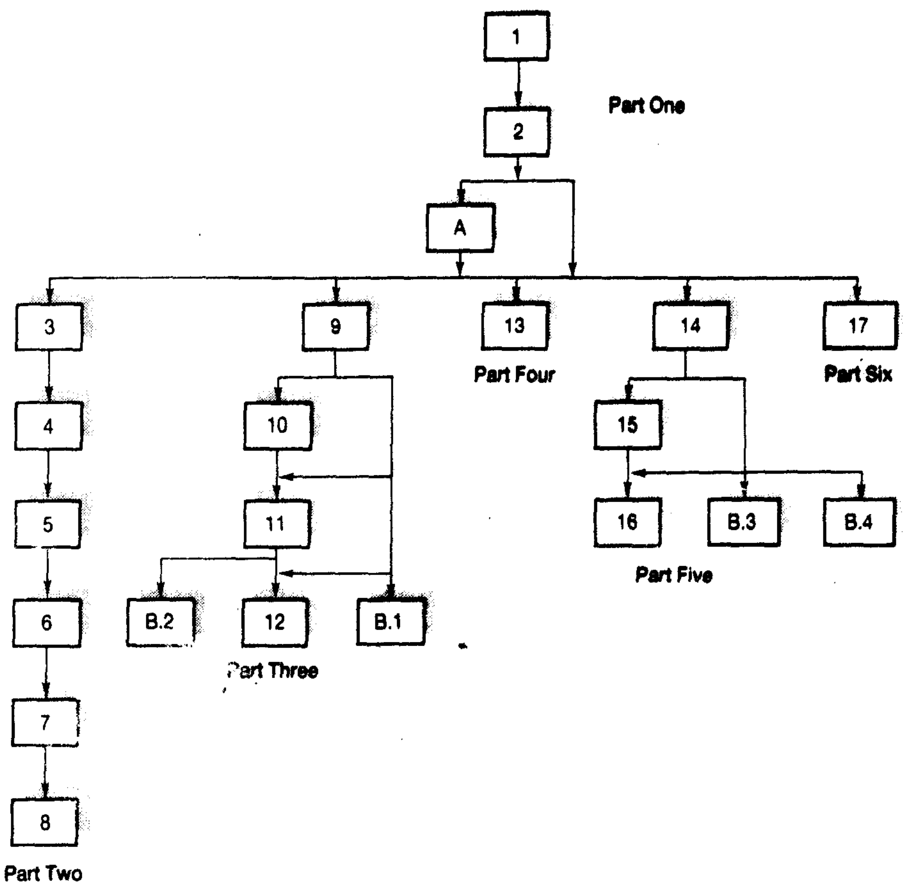
Aims and audience

The progress in technology has enabled microprocessor systems to have a level of functionality and performance only achievable by large computers of a previous generation. Packaging density has reached the point where a single microprocessor chip includes, in addition to the CPU, caches, memory management facilities, multiprocessor support and support for extended functionality (for example, floating-point operations) via a co-processor interface. In order to achieve a high execution speed, acceleration mechanisms such as pipelining and prefetching are used.

It is the aim of this book to provide a foundation for understanding, evaluating and comparing alternative design principles used by state-of-the-art microprocessors and minicomputers. Studying the manufacturer's manual on a new microprocessor then becomes a process of recognition and classification.

Rather than basing the text on a theoretical computer, the architectures of current microprocessors and minicomputers with a widespread acceptance have been used. Examples are taken primarily from the Motorola MC68000 family, abstracted to the relevant level of detail. The DEC VAX-11 architecture has been used as a second source, while examples from other architectures are used only to illustrate important concepts or alternatives. Although the Intel 80X86 architecture is being used widely in many personal computers, it is less suited to illustrate important architectural concepts because of its irregularities and inconsistent structure. Programming examples are mainly given using the language Pascal.

The material is presented at a level and depth suitable for



intermediate and advanced undergraduate courses in computer science, computer engineering and electrical engineering curricula. Currently, the book is being used as a text for two courses: Computer Architecture (covering Part One, Part Two and the first chapter of Part Three) and Computer Organization (covering the remainder of the book). The material in Part Three (operating system interface), Part Four (buses), Part Five (acceleration) and Part Six (trends) has been presented in such a way that each part can be taken in isolation as an optional subject for a course. The first chapters of Parts Three and Five serve as an introduction/overview for the remaining chapters of those parts. These remaining chapters are also organized so that they can be taken in isolation. The flowchart opposite shows some possible paths through the book.

The material in this book is complementary to that covered in courses on compilers and operating systems. When a bottom-up approach is taken, these courses can be given after the courses covering the compiler and operating system interface. When a top-down approach is preferred, compilers and operating systems should be covered first.

The reader should have some knowledge of programming languages and preferably some understanding of computer hardware at the introductory level.

Practising professionals can use the book to keep up to date with the latest developments in computer architecture and organization, and to become aware of how these developments interrelate.

Organization

The book is organized into several parts. Part One provides an introductory overview. Part Two discusses the compiler interface. Part Three covers the operating system interface. Part Four discusses buses and can be seen as the hardware interface. Part Five covers acceleration mechanisms, while Part Six covers modern trends, with the accent on non-conventional architectures.

A special language, such as IPL or ISP, could have been used for describing the operation of instructions. Instead, a notation similar to that used in processor manuals has been chosen, because it is closer to the real world and easier to learn. Rather than including a special chapter on the history of computers, the historical evolution of the different concepts is presented together with those concepts. Each chapter starts with an overview of the topics to be discussed and ends with a summary and a set of exercises. The solutions are available at the back of the book to aid self-study. A Glossary of terms has been added for the reader's convenience. An overview of the MC68020 has been included in Appendix A to give the reader a frame of reference for a complete architecture. This appendix is not intended to be comprehensive. Special topics are covered in Appendix B.

Part One Introductory

- **Chapter 1**, **Introduction**, defines computer architecture and introduces a set of quality measures, followed by a classification of computer architectures.
- **Chapter 2**, **Machine Levels**, addresses the levels of abstraction in computer systems (machine levels), together with an analysis of the proper choice of the level to be supported by the architecture. It also addresses software portability.

Part Two The Compiler Interface

- **Chapter 3**, **Data Representation**, illustrates the way in which data, as defined in high-level languages, is represented at the computer architectural level. It also covers data structures, such as stacks and queues, which support the run-time system.
- **Chapter 4**, **Machine Languages**, introduces the basics of instruction set design without going into detail into the different classes of instructions. Instruction formats, the number of operands, and the specification alternatives for opcode, data type, data location and data value are presented.
- **Chapter 5**, **Addressing**, introduces the notion of addressing and addressing modes, together with mechanisms for compact specification of operands (which includes addresses). Addressing modes are used for accessing data elements in arrays, records and stacks. In addition to the MC68000, the VAX-11, NS32000 and Nebula architectures are used to illustrate implementations of addressing mechanisms.
- **Chapter 6**, **Operations on Data**, covers operations on data. It describes the classes of operations, such as move, logical and arithmetic, together with the properties of each class. The effect of multiple-precision arithmetic on data representation is shown, and the handling of range and domain errors is treated.
- **Chapter 7**, **Program Flow Control**, covers the sequence of instructions, condition codes, conditional operations, such as branches, and architectural support for iteration mechanisms.
- **Chapter 8**, **High-Level Program Structures**, deals with groups of instructions – that is, procedures and modules. Binding mechanisms, run-time models and subroutines, and function and module mechanisms are presented conceptually and illustrated with examples from the MC68020 architecture.

Part Three The Operating System Interface

- **Chapter 9, Operating Systems**, presents the architectural support for operating systems: process protection, synchronization and switching. It also covers interrupt handling and virtual machine monitors.
- **Chapter 10, Memory Management**, covers input and output device properties and different types of I/O operations – that is, programmed, overlapped or autonomous I/O. Issues such as I/O in virtual or physical space and properties of I/O processes are addressed.
- **Chapter 11, Input/Output**, introduces virtual memory and discusses the mapping of virtual addresses on to physical addresses. Relocation, paging and segmentation are presented as mapping mechanisms. Memory protection alternatives are also presented. The introduced concepts are illustrated with some current implementations.
- **Chapter 12, Exception Handling,** covers error and exception handling, together with program debugging support (which typically uses a similar mechanism).

Part Four The Hardware Interface

- **Chapter 13, Bus Systems**, introduces the basics of bus systems – concepts such as arbitration, addressing, data transport, interrupts and error handling are presented. Examples are taken from the Multibus II, the Nubus and the IEEE-488 bus, while the VME bus is covered in more detail to show how the different concepts fit together.

Part Five Acceleration

- **Chapter 14, Acceleration Mechanisms**, starts with an overview of how program execution can be accelerated and is followed by a discussion of the acceleration mechanisms that can be found at the implementation level of a computer system. It includes caches, pipelining, interleaving and co-processors. The chapter ends with a discussion of the acceleration mechanisms used in the MC68000 family. Some of the concepts introduced in this chapter are detailed in Chapters 15 and 16.
- **Chapter 15, Reduced Instruction Set Computers**, presents the basics behind reduced instruction set computers (RISC). It shows how compiler technology can be used to simplify the hardware and to achieve faster instructions. Register windows and delayed branches are presented, together with the RISC II architecture used to show the synergism of the RISC concepts.
- **Chapter 16, Multiprocessor Systems**, is devoted to multiprocessor systems, with special attention being paid to interconnection

networks and tightly coupled multiprocessor systems with shared memory. Data consistency issues are also discussed.

Part Six Trends

- **Chapter 17**, **Architectural Trends**, covers trends. It starts with an analysis of the Von Neumann model, indicating its shortcomings. A new taxonomy is presented, based on the work of Treleaven, which allows for the classification of newer computational models. A short description of each of the eight models is given.

Acknowledgements

The first nine chapters of this book are essentially an updated version of the book *Computerarchitectuur* by A. J. van de Goor and H. A. Spanjersberg, written in the Dutch language. I would like to thank H. A. Spanjersberg for his contribution to this Dutch version, especially for his skilful editing, which often involved rewriting parts of the text.

The basic research for the remainder of the book has been done by many students, of whom I would like to mention J. Karbaat and C. van Reeuwijk, who undertook the first attempt to compile the large amount of available information, and M. J. Druzdzel who collected the information on non-conventional architectures.

I owe special acknowledgements to M. Korsloot and M. P. de Kluyver, who translated the original Dutch version and were instrumental in getting the remainder of the book into its current form. Jointly we went through many iterations of restructuring and of incorporating newer and better examples. They also made the writing of this book fun.

I would also like to express my appreciation to Addison-Wesley for the pleasant way we have been able to complete this book.

The author and publishers would like to thank: the Association for Computing Machinery Inc. for permission to reproduce Figures 14.15, 14.16 and 14.22 from A. J. Smith (1982) Cache memories, *Computing Surveys*, **14**(3), 473–530 and Figures 16.20 and 16.21 from J. Archibald and J. L. Baer (1986) Cache coherence protocols: evaluation using a multiprocessor simulation model, *Trans. on Computer Systems*, **4**(4), 273–98; and the Institute of Electrical and Electronics Engineers Inc. for permission to reproduce Tables 15.2 and 15.3 from D. MacGregor and J. Rubinstein (1985) A performance analysis of MC68020-based systems, *IEEE Micro*, **5**(6), 50–70 and Figure 10.14 from W. W. Chu and H. Opderbeck (1974) Performance of replacement algorithms with different page sizes, *Computer*, **7**(11), 14–21.

Finally, I would like to thank my wife Anny, and our children, for their patience and understanding.

A. J. van de Goor

April 1989

For reasons of simplicity, the pronoun 'he' is used to relate to both male and female throughout the book.

CONTENTS

PART ONE

INTRODUCTORY

CHAPTER 1 INTRODUCTION
CHAPTER 2 MACHINE LEVELS

The introductory part of this book defines the notion of computer architecture and distinguishes the three design levels (architecture, implementation and realization) involved in the process of evolving from idea to finished product. It describes the characteristics a good architecture should have and outlines a classification of *traditional architectures*.

The different levels of abstraction in a computer system, which are also called virtual machines, are presented. Computer architecture is defined to be the highest level of abstraction that is implemented in hardware. Conversion mechanisms to go from higher to lower levels of abstraction are described in relationship with the *problems of software portability to a new architecture*.