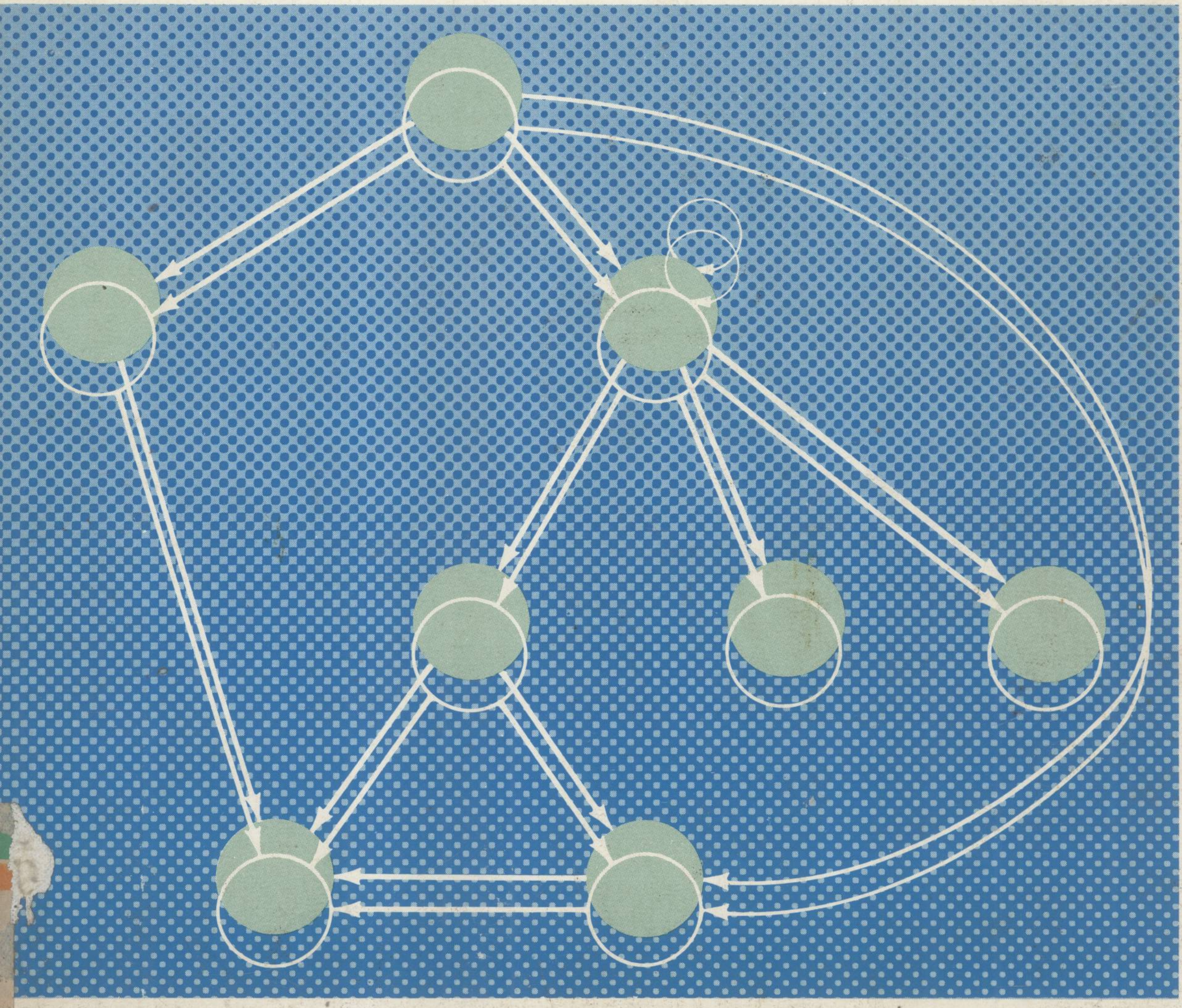
MACHINE AND ASSEMBLY LANGUAGE PROGRAMMING OF THE PDP-11

ARTHUR GILL



MACHINE AND ASSEMBLY LANGUAGE PROGRAMMING OF THE PDP-11

ARTHUR GILL

Department of Electrical Engineering and Computer Sciences
University of California, Berkeley

PRENTICE-HALL, INC., ENGLEWOOD CLIFFS, NEW JERSEY 07632

Library of Congress Cataloging in Publication Data

GILL, ARTHUR, (date)
Machine and assembly language programming of PDP-11.

Includes index.
1. PDP-11 (Computer)—Programming. 2. Assembler language (Computer program language) I. Title.

QA76.8.P2G54 001.6'42 78-9690
ISBN 0-13-541870-4

Printed in the United States of America

10 9 8 7 6 5 4 3 2

PRENTICE-HALL INTERNATIONAL, INC., *London*
PRENTICE-HALL OF AUSTRALIA PTY. LIMITED, *Sydney*
PRENTICE-HALL OF CANADA, LTD., *Toronto*
PRENTICE-HALL OF INDIA PRIVATE LIMITED, *New Delhi*
PRENTICE-HALL OF JAPAN, INC., *Tokyo*
PRENTICE-HALL OF SOUTHEAST ASIA PTE. LTD., *Singapore*
WHITEHALL BOOKS LIMITED, *Wellington, New Zealand*

MACHINE AND ASSEMBLY LANGUAGE PROGRAMMING OF THE PDP-11

To Velta

PREFACE

This book evolved from notes written for a course in machine structures offered at the University of California at Berkeley. It is a second course in computer science (the first being a course in a high-level language programming) where students are exposed to the basic concept of computer operation (registers, instruction set, addressing modes, etc.) and learn assembly language programming techniques.

The course offers the students "hands-on" experience with a stand-alone computer, one which they can program and manipulate with no operating system standing between them and important machine features. The availability of a "bare" machine not only eliminates much of the "magic" associated with computers and computation, but also permits experimentation with I/O programming, interrupt handling, and other essential techniques not possible with simulated or "protected" machines. In addition to the stand-alone machine, students have access to a large time-shared computer where they can perform editing, assembling, linking, filing, and other operations which are essentially of a bookkeeping nature.

For these purposes we selected the Digital Equipment Corporation's PDP-11 family of computers. (Specifically, we used the PDP-11/10 as the stand-alone machine, and the PDP-11/70, equipped with the UNIX operating system, as the time-shared computer.) The PDP-11 machines were chosen because of their versatility, popularity, and the wide range of available software systems. We found the PDP-11's highly suitable as vehicles for presenting the fundamental concepts of the course, and at the same time exposing the students to the ever-widening world of minicomputers.

The objective of this book, then, is to familiarize the reader with the basic organizational and operational features of the PDP-11 and to present machine and assembly language techniques for this class of computers. It is not, per se, a general text on machine structures, and does not attempt to provide a comprehensive treatment of available computer organizations and assemblers. However, to the extent that the concepts and methods governing the operation and programming of the PDP-11 are used in many other minicomputers, the material in this book should serve as good preparation for the operation and programming of other machines.

Chapter 1 outlines algorithms for converting numbers from one system (binary, octal, decimal) to another. The algorithms are not provided with proof and are intended to serve only for reference. Chapter 2 describes the organizational structure of the PDP-11 (the central memory, central processor, and peripheral devices). Chapter 3 explains how numbers (integers and floating point), characters, and strings are represented in the PDP-11. Chapter 4 describes the PDP-11's instruction formats and addressing modes, and Chapter 5 introduces the reader to assembly language programming.

The first five chapters should provide the reader with sufficient background to write simple programs for the PDP-11. The remaining chapters delve deeper into operational details and describe further techniques. Chapter 6 introduces stacks and subroutines (including recursion). Chapter 7 looks closer at the PDP-11's arithmetic (including double-precision) and other operations, such as the test, comparison, branch, and shift operations. Chapter 8 explains the trap and interrupt mechanisms. Chapter 9 describes the workings of the assembler and linkage editor and the notion of relocation. (Although the MACRO-11 assembler and LINKR-11 linkage editor are used for illustration, the concepts discussed are quite general.) Chapter 10 introduces some advanced assembler facilities, such as macros, repeated assembly, and conditional assembly.

The book ends with a number of appendixes, which consist of reference lists and tables (character code, summary of addressing modes, list of operation codes, etc.). There is also an appendix on programming style, which should be carefully read by the beginner.

Each chapter concludes with a set of exercises that serve to illustrate and sometimes complement the material in the text. The reader is encouraged to solve the problems and run the programs included in these exercises. True assimilation of the material in this book can come about only through practice — by the actual writing and execution of programs.

The only prerequisite to this book is some experience with high-level language programming. No particular language is assumed, but it is taken for granted that the reader is familiar with the notions of an algorithm, a flowchart, and a stored program.

It is not intended that this book stand alone as a course text. Since it does not describe all the fine details of the PDP-11 instructions and assembler directives, students should be in possession of the PDP-11 processor handbook and the assembler manual appropriate to their particular installation, where these details can be found when needed. Little is said in the book regarding peripheral equipment (only the teletype and line clock are treated in any detail), and students doing I/O programming may wish to refer to the PDP-11 peripherals handbook and local manuals for assistance.

The author is indebted to Mr. R. S. Epstein of the University of California at Berkeley for reviewing the manuscript, for offering useful advice, and for contributing most of Section 5.5 as well as some exercises. Thanks are also due to Professors R. S. Fabry and M. R. Stonebraker (also of U.C., Berkeley) for helpful comments and suggestions.

ARTHUR GILL

CONTENTS

2

5

ASSEMBLY LANGUAGE PROGRAMMING 45

6

STACKS AND SUBROUTINES 60

7

ARITHMETIC OPERATIONS 85

NUMBER SYSTEMS

1

In working with the PDP-11, we shall make extensive use of the binary and octal number systems, as well as the decimal system. It is important that the student acquire, as soon as possible, the facility to convert from one system to another. In this chapter we shall outline, without proof, some algorithms for carrying out these conversions. Students familiar with these algorithms may proceed directly to Chapter 2.

A "number" in this chapter will mean a non-negative integer (0, 1, 2, . . .). The number N will be denoted by N_{10}, N_8, or N_2 if it is in the decimal, octal, or binary system, respectively. However, the subscript may be dropped if it is understood from the context.

An m-digit number will be written symbolically as $D_{m-1} \cdots D_1 D_0$ [D_i being the (i + 1)st digit from the right].

The flowcharts in Figures 1.1 and 1.2 describe algorithms for converting a decimal number N into its binary equivalent M.

Example (Subtraction-of-powers method)

$$N = 217_{10}$$

$$217 - 2^7 = 217 - 128 = 89 \quad (D_7 = 1)$$
$$89 - 2^6 = 89 - 64 = 25 \quad (D_6 = 1)$$
$$25 - 2^4 = 25 - 16 = 9 \quad (D_4 = 1)$$
$$9 - 2^3 = 9 - 8 = 1 \quad (D_3 = 1)$$
$$1 - 2^0 = 1 - 1 = 0 \quad (D_0 = 1)$$
$$M = 11011001_2$$

□

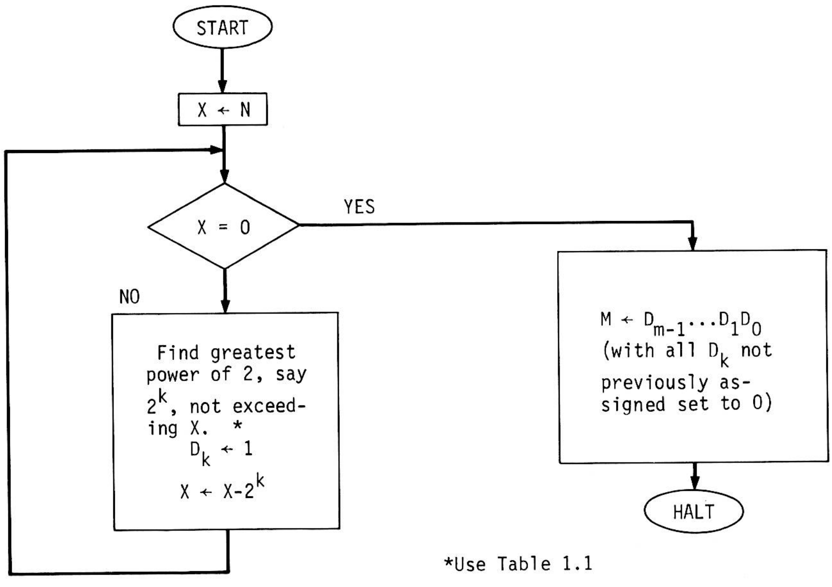


Figure 1.1 Decimal-to-binary conversion by subtraction of powers.

Example (Division method)

$$N = 217_{10}$$

$$217 \text{ is odd} \quad (D_0 = 1)$$
$$217/2 = 108 \text{ is even} \quad (D_1 = 0)$$
$$108/2 = 54 \text{ is even} \quad (D_2 = 0)$$
$$54/2 = 27 \text{ is odd} \quad (D_3 = 1)$$

$$
\begin{array}{lll}
27/2 & = 13 \text{ is odd} & (D_4 = 1) \\
13/2 & = 6 \text{ is even} & (D_5 = 0) \\
6/2 & = 3 \text{ is odd} & (D_6 = 1) \\
3/2 & = 1 \text{ is odd} & (D_7 = 1) \\
1/2 & = 0 & \\
 & M = 11011001_2 &
\end{array}
$$

□

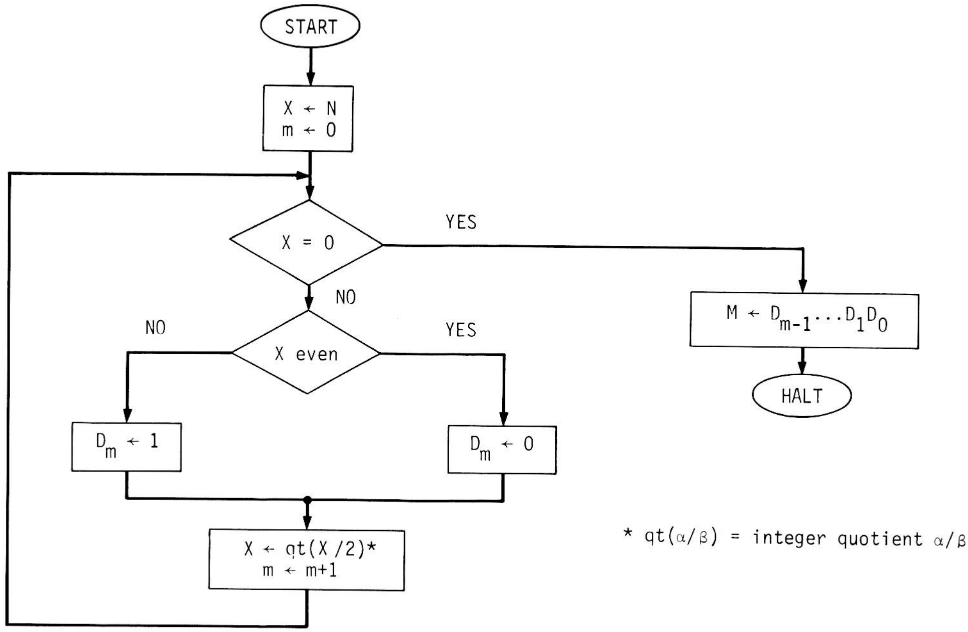


Figure 1.2 Decimal-to-binary conversion by division.

1.2 DECIMAL-TO-OCTAL CONVERSION

The flowcharts in Figures 1.3 and 1.4 describe algorithms for converting a decimal number N into its octal equivalent M.

Example (Subtraction-of-powers method)

$$
\begin{array}{rl}
 & N = 2591_{10} \\
2591 - 5 \cdot 8^3 = 2591 - 2560 = 31 & (D_3 = 5) \\
31 - 3 \cdot 8^1 = 31 - 24 = 7 & (D_1 = 3) \\
7 - 7 \cdot 8^0 = 0 & (D_0 = 7) \\
M = 5037_8 &
\end{array}
$$

□

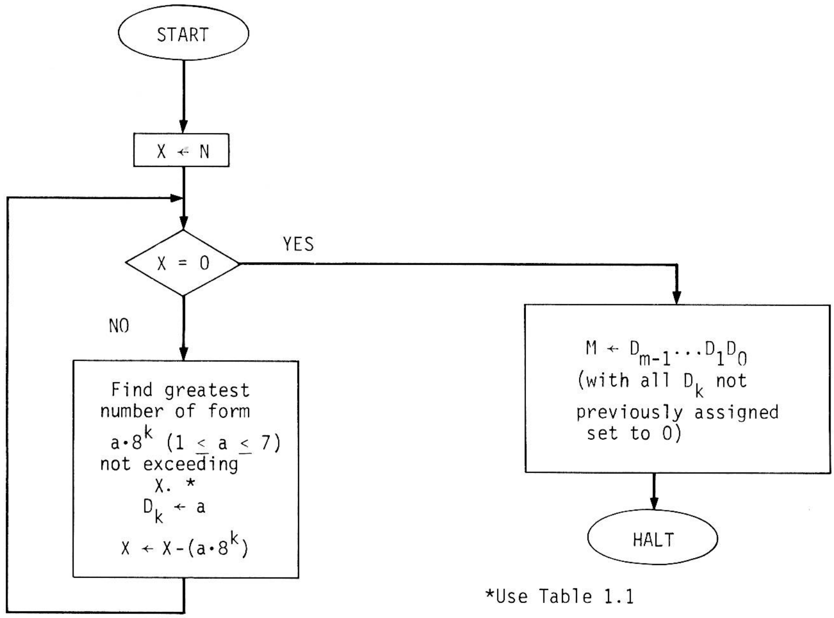


Figure 1.3 Decimal-to-octal conversion by subtraction of powers.

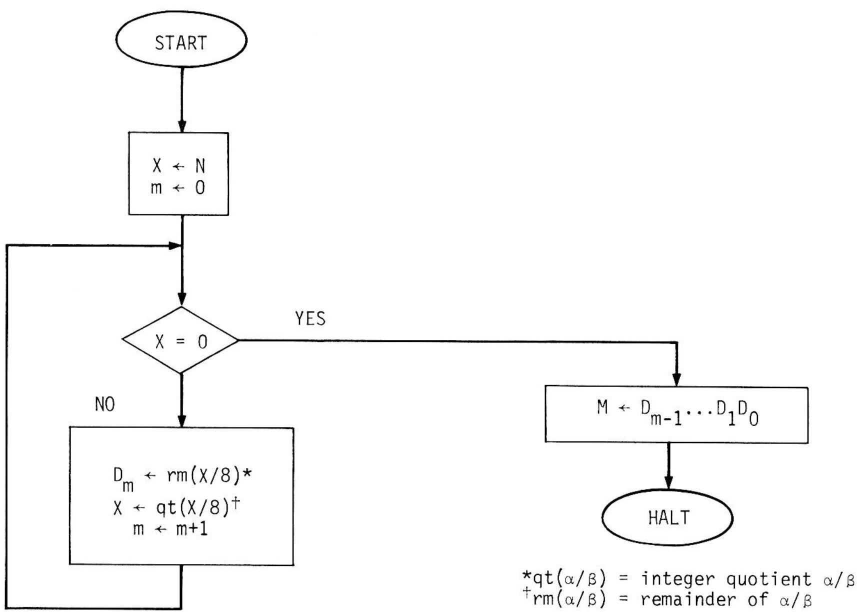


Figure 1.4 Decimal-to-octal conversion by division.