MICROCOMPUTER EXPERIMENTATION WITH THE INTEL SDK-85

LANCE A. LEVENTHAL
COLIN WALSH

MICROCOMPUTER EXPERIMENTATION WITH THE INTEL SDK-85

LANCE A. LEVENTHAL
COLIN WALSH

Emulative Systems Company
San Diego, California

PRENTICE-HALL, INC., *Englewood Cliffs, New Jersey 07632*

Library of Congress Cataloging in Publication Data

Leventhal, Lance A date–
Microcomputer experimentation with the Intel SDK-85.

Bibliography: p.
Includes index.
1. INTEL SDK-85 (Computer) I. Walsh, William Colin, joint author. II. Title.
QA76.8.I293L48 1980. 001.6'4'05 79-22052
ISBN 0-13-580860-X

Editorial/production supervision
and interior design by Virginia Huebner
Cover design by Edsal Enterprises
Manufacturing buyer: Gordon Osbourne

Printed in the United States of America

10 9 8 7 6 5 4 3 2 1

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*

Preface

This manual provides an introductory laboratory for those who plan to use microcomputers as controllers in instruments, communications systems, computer peripherals, test equipment, business equipment, industrial control, process control, signal processing, and consumer products. Our emphasis is on users who plan to purchase assembled microcomputers and use them in engineering applications. Our aims are: 1) to introduce the basic methods of assembly language programming, 2) to show how to perform simple functions that are required in controller applications, 3) to provide examples of hardware/software tradeoffs, 4) to show how programs should be designed, debugged, and documented, 5) to describe the interfacing of microcomputers with mechanical devices and with human users, 6) to demonstrate alternative approaches to input/output and timing, 7) to present the advantages and uses of programmable LSI devices, and 8) to describe methods by which microcomputers can communicate with external systems. We have provided examples that are drawn from actual applications, but are simplified so as not to require extensive background, special equipment (beyond the basic microcomputer), or long setup times. We have made the manual self-contained, so that it can be used in a variety of disciplines at differing levels.

We have based the manual on the Intel SDK-85 microcomputer because of its low cost, wide availability, completeness, and ease of use. The SDK-85 does not require additional peripherals, uses a popular

microprocessor, has adequate documentation, and is produced by a leading semiconductor manufacturer. It is also easy to assemble, includes a prototyping area and expansion facilities, and provides all the components of typical microcomputer systems. The SDK-85 has a ROM-based monitor for simple interaction and has enough memory and input/output lines for a variety of useful examples.

Our emphasis throughout this manual is on the control of external systems through software. We have illustrated this control with the simplest possible examples using switches, single displays, and the on-board peripherals. We intend to cover more advanced interfacing topics in a later continuation. Our intent here has been to provide exercises that require little additional hardware and can be performed in short time periods. We have often included initial programs as starting points for students.

We have followed a standard format throughout the manual so as to conform with other references. We have used the notation from the Intel 8085 assembler. We have tried to make all programs clear, simple, well-structured, and well-documented. We have avoided programming tricks even when they would make programs somewhat shorter and faster. Good programming practices are essential for those who plan to work with microcomputers and we have tried to provide sound examples for students to follow.

This manual does not describe the Intel 8085 microprocessor in detail. We have provided references to the standard 8085 manuals and to appropriate textbooks. Nor does this manual provide a complete discussion of 8085 assembly language programming. We have therefore provided extensive references to programming books. However, we have tried to make the manual self-contained so that it can be used independently of any of the reference material.

A manual like this obviously requires inputs from many people. We would like to particularly thank the faculty and students of the Mechanical Engineering Department of San Diego State University for giving us an opportunity to test this material in a class environment. Dr. George Craig and Mr. George Mansfield arranged the course. Among those who pointed out errors and provided helpful suggestions were Mr. Richard Chiang, Mr. Chuck Chandler, Mr. Douglas Dahl, Mr. Augustin Gonzalez, Mr. James Housman, Mr. Martin Lee, Mr. Sergio Lopez, Mr. Chan Nguyen, Mr. James Scheidler, Mr. Shiv Singh, Mr. Kunvarji Thakor, and Mr. Miroslaw Czaljkowski. Our special thanks go to Mr. Mihran LeVon Jr. of NCR Corporation who tried many of the experiments and suggested numerous improvements and corrections.

Others who helped include Mr. Gary Hankins and Mr. Winthrop Saville of Sorrento Valley Associates. The reviewers, all of them anonymous except for Mr. Sol Libes of Union County Technical Institute (New

Jersey), provided many useful suggestions. Our editor, Mr. Paul Becker, encouraged this project as did Mr. Walter Welch, the local representative of Prentice-Hall. Ms. Marielle Brand-Carter and Ms. Jacqueline Roberge did most of the typing. Obviously, all remaining errors, omissions, and inconsistencies are the responsibility of the authors.

LANCE A. LEVENTHAL
COLIN WALSH

San Diego, California

Contents

LABORATORY 6—PROCESSING DATA ARRAYS

LABORATORY 7—FORMING DATA ARRAYS

LABORATORY 8—DESIGNING AND DEBUGGING PROGRAMS

LABORATORY 9—ARITHMETIC

LABORATORY A—SUBROUTINES AND THE STACK

LABORATORY B—INPUT/OUTPUT USING HANDSHAKES

LABORATORY C—INTERRUPTS

LABORATORY D—TIMING METHODS

LABORATORY E—SERIAL INPUT/OUTPUT

LABORATORY F—EXAMINING PROCESSOR SIGNALS

APPENDICES

Laboratory 0

Introduction to the SDK-85 Microcomputer

PURPOSE

To learn how to use the basic functions of the SDK-85 microcomputer.

PARTS REQUIRED

An assembled SDK-85 microcomputer with a 5-Volt power supply.

REFERENCE MATERIALS

SDK-85 User's Manual, Intel Corp., Santa Clara, CA, December 1977, Chapters 2 (assembly) and 3 (checkout), pp. 4-1 through 4-3 (monitor commands).

WHAT YOU SHOULD LEARN

1) How to reset the computer.
2) How to examine the contents of a memory location.
3) How to change the contents of a memory location.
4) How to enter and execute a simple program.

TERMS

Central processing unit (CPU)—the control section of the computer; the part that controls its operations, fetches and executes instructions, and performs arithmetic and logical functions.

Hexadecimal (or Hex)—number system with base 16. The digits are the decimal numbers 0 through 9, followed by the letters A through F.

Microcomputer—a computer that has a microprocessor as its central processing unit.

Microprocessor—a complete central processing unit for a computer constructed on one or a few chips of silicon.

Monitor—a program that allows the computer user to enter programs and data, run programs, examine the contents of the computer's memory and registers, and utilize the computer's peripherals.

Nonvolatile memory—a memory that retains its contents when power is removed.

Random-access memory (RAM)—a memory that can be both read and altered (written) in normal operation.

Read-only memory (ROM)—a memory that can be read but not altered in normal operation.

Register—a storage location inside the CPU.

Reset—a control signal that causes the computer to enter a known initial (or startup) state.

Volatile memory—a memory that loses its contents when power is removed.

8085 INSTRUCTIONS

RST 1 (CF hex)—RESTART 1; on the SDK-85 microcomputer, this instruction causes the microprocessor to return control to the monitor program.

OVERVIEW

The Intel SDK-85 (or System Design Kit-85) is an inexpensive microcomputer based on the widely used Intel 8085 microprocessor. Assembly instructions for the kit are given in Chapters 2 and 3 of the *SDK-85 User's Manual.* The kit consists of (see Fig. 0-1) the following items:

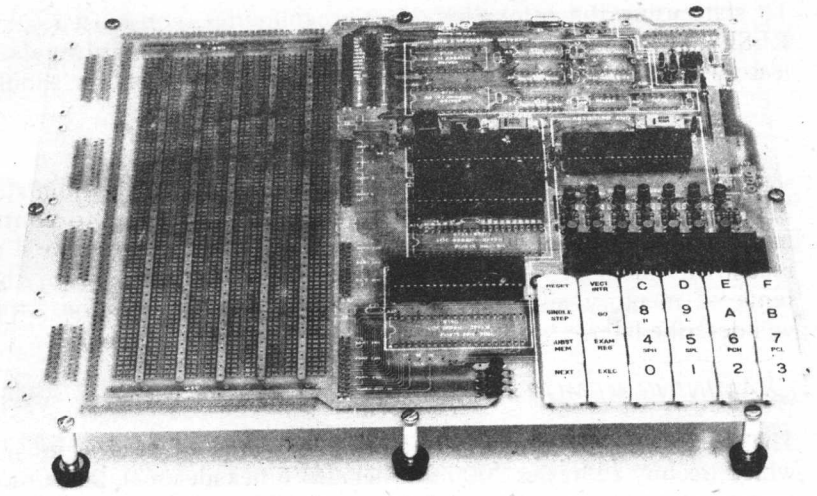

FIGURE 0-1. The assembled SDK-85 microcomputer. (Photo, courtesy Intel Corp., Santa Clara, CA.)

- An 8085 microprocessor which serves as the central processing unit or "brain."
- Read-only memory or ROM (an 8355 device which contains a monitor program). Each 8355 ROM contains 2K 8-bit words ($1K = 2^{10} = 1024$).
- Read/write memory or RAM (an 8155 device into which the user can enter data and programs). Each 8155 RAM contains 256 8-bit words.
- Keyboard/display interface (an 8279 device).
- 24-key keyboard.
- Six-digit seven-segment LED display.
- Expansion area (upper right-hand side).
- Prototyping area (left-hand side).

You can find more complete descriptions of the various devices in Chapter 5 of the *SDK-85 User's Manual* and in the *MCS-85 User's Manual.* Appendix 3 contains partial reproductions of those descriptions.

RESETTING THE COMPUTER

To start using the Intel SDK-85 microcomputer, you must reset it. The RESET key is in the top left-hand corner of the keyboard; press and release it. If everything is operating properly, the displays should read

– 80 85

The microcomputer is now executing a monitor program stored in the 8355 read-only memory. This program allows you to control the microcomputer from the keyboard. You can place programs and data in read/write memory, execute programs, examine and change the contents of memory and registers, and perform other functions which we will describe later.

EXAMINING MEMORY

The basic SDK-85 system contains 256 words of read/write memory which occupy addresses 2000 through 20FF hexadecimal. Since the monitor uses the addresses above 20C2, we will not use those locations.

Note that each memory location has a 16-bit address (four hexadecimal digits) and contains 8 bits of data (two hexadecimal digits). Table 0-1 is a list of the hexadecimal digits and their binary and decimal equivalents. Refer to this table if you need help converting numbers to and from the hexadecimal representation.

Table 0-1

HEXADECIMAL-TO-DECIMAL CONVERSION TABLE

HEXADECIMAL DIGIT	DECIMAL VALUE	BINARY VALUE
0	0	0000
1	1	0001
2	2	0010
3	3	0011
4	4	0100
5	5	0101
6	6	0110
7	7	0111
8	8	1000
9	9	1001
A	10	1010
B or b	11	1011
C	12	1100
D or d	13	1101
E	14	1110
F	15	1111

To examine the contents of memory, press the SUBST MEM or SUBSTITUTE MEMORY key. If you get an error (the display shows -Err) or some other unexpected display, press the RESET key and then the SUBST MEM key. The displays should show only a decimal point after the four leftmost digits.

Now enter a four-digit address: try 2, 0, 0, 0. Note that as soon as you enter the first digit, the four leftmost displays read

0 0 0 2

Each subsequent entry causes the "2" to move one place to the left. The two rightmost displays remain blank. Remember that all the displays are in hexadecimal and that addresses (shown on the leftmost displays) are four digits long whereas data entries (shown on the rightmost displays) are two digits long.

Now press the NEXT key in the lower left-hand corner. The two rightmost displays show what is in memory location 2000 (hex). The result is arbitrary, since the 8155 RAM loses its contents when power is removed and could start in any state whatsoever. Such a memory is said to be *volatile.* If you want to see the effects of this volatility, simply unplug the SDK-85's power supply and repeat the examination procedure.

The following procedure allows you to examine the contents of a memory location:

1) (if necessary) Reset the computer with the RESET key.
2) Press the SUBST MEM key.
3) Enter the address as four hexadecimal digits starting with the most significant digit.
4) Press the NEXT key.

Before you press the NEXT key, be sure that you have entered the address correctly. If not, enter the correct address. If you make a mistake or get confused, just press RESET and get back on the right track.

PROBLEM 0-1

Examine the contents of memory location 2038 (hex).

PROBLEM 0-2

Examine the contents of memory location 005B (hex). Its value should be 21. Try disconnecting the power supply and examining this location again. The result will be the same, since this memory location is in the *nonvolatile* read-only memory.

Once you have examined the contents of a memory location, you can examine the contents of the next location by pressing the NEXT key