THE MICROCOMPUTER IN CELL AND NEUROBIOLOGY RESEARCH

Edited by

R. Ranney Mize, Ph.D.

THE MICROCOMPUTER IN CELL AND NEUROBIOLOGY RESEARCH

Edited by

R. Ranney Mize, Ph.D.

Associate Professor, Department of Anatomy
and Division of Neuroscience
College of Medicine, University of Tennessee
Center for the Health Sciences, Memphis

ELSEVIER
New York · Amsterdam · Oxford

Preface

The computer revolution has arrived in cell and neurobiology research. The advent of the microcomputer makes measurement, control, and analysis of data both simpler and far faster than seemed imaginable a decade ago. In the mid-70s, computer analysis of cell and neurobiology experiments was generally limited to a few "wealthy" laboratories that could afford high-priced mainframe or minicomputers. Today, inexpensive microprocessors and microcomputers make computer analysis accessible to virtually every scientist. Spurred by the popularity of the personal computer and the availability of low cost computers like the Apple and IBM-XT, many cell and neurobiologists have "computerized" their laboratories. In the near future, the microcomputer will become an indispensable general purpose research tool in the laboratory.

In a recent review of microcomputer applications in cell and neurobiology (Mize, 1984), I became aware of a number of microcomputer systems that had been developed for use in those disciplines. Descriptions of these systems, however, either had not been published or were scattered in a wide variety of journals. Those that were published varied greatly in technical detail and research use. Often they were too technical to be understood by the computer novice. Just as frequently they included too little detail to be of use to those with a background in computers. The articles often lacked a comprehensive description of the research application for which they were designed. Most importantly, I found very few books that collected such information in a single volume. This book is an effort in that direction.

Although the book is designed primarily for cell biologists, neurobiologists, and others who analyze the structure and function of tissue, scientists in related fields will also find it useful for understanding microcomputer architecture and the potential of microcomputers in a laboratory setting. Graduate and advanced undergraduate students should be able to use it as a supplementary text in biologically oriented computer courses.

The book is divided into six sections: the first section includes chapters describing the components of a microcomputer, programming languages, and hardware and software selection. The remaining sections describe microcomputer systems used in Light and Electron Microscopy, Morphometry, Serial Section Reconstruction, Imaging and Densitometry, and Electrophysiological Recording. These sections cover a broad range of topics of interest to most scientists in the rapidly growing fields of cell and neurobiology. The chapters are written for those with experience in research but without computer expertise. Each chapter includes sufficient technical detail, however, to be useful to those with extensive backgrounds in computer science.

I have chosen an introductory chapter for each section, which will introduce the reader to principles of hardware and software design for that type of research. The introductory chapter is followed by chapters that describe specific systems used for particular research applications. I have chosen systems that should interest a large group of biologists. Several criteria were used in choosing these chapters. First, I looked for systems that used commercially available desk-top or microcomputers. (A microcomputer was defined as a portable self-contained unit with 8- or 16-bit microprocessors, memory, and standard peripherals such as printers, plotters, and a CRT.) Second, I tried to choose systems that were just that—systems. Programs developed to collect or analyze data without the necessary design of appropriate interfaces to laboratory instruments or other hardware were not considered. The computer systems also had to include specially designed software of special value to the biological community. Third, I tried to choose systems in which the software was written at least partially in a high-level programming language such as BASIC, FORTRAN, or PASCAL. This should allow the software to be transported to a variety of microcomputers. Finally, I chose computer systems that have been used extensively to collect data in a laboratory environment. Most of the application chapters in the book include sections that describe actual applications in cell and neurobiology research.

There are a number of people I wish to thank for helping put this book together. The late John Lawrence of Elsevier was extremely cooperative in all phases of the publication. His enthusiasm for the idea and his expert publishing skills were indispensable to the completion of the book. Louise Calabro Gruendel, Dorita James, and Jane Licht of Elsevier were of great assistance in preparing and editing the book. Their pleasant and cooperative spirit is gratefully acknowledged. Mary King Givens and Ellen McDonell of the Medical Library of the University of Tennessee Center for the Health Sciences assisted considerably in the library searches that brought some of the contributors to my attention. My research assistant, Linda Horner, was most helpful in proofing the text. I also want to thank several colleagues who first exposed me to computer science. They include Dr. John Stevens, Dr. Larry Palmer, and Cameron Street, who taught me to appreciate structured programming. Finally, I want to thank my wife, Dr. Emel Songu-Mize, who has tolerated and often encouraged my obsession, both with computers and the completion of this book.

The computer revolution has arrived. I hope this book helps the reader join that revolution more easily. Happy computing.

Reference

Mize, R. R. (1984) Computer applications in cell and neurobiology: A review. Int. Rev. Cytol. 90:83–124.

Contributors

Steven Akeson
Department of Biochemistry, University of California, San Francisco, California

Franklin R. Amthor
Department of Physiological Optics, School of Optometry, University of Alabama in Birmingham, Birmingham, Alabama

Michael Breton
Scheie Eye Institute, University of Pennsylvania, Philadelphia, Pennsylvania

J. J. Capowski
Department of Physiology, University of North Carolina Medical School, Chapel Hill, North Carolina

Vivien A. Casagrande
Department of Anatomy, Vanderbilt University, Nashville, Tennessee

Thomas L. Davis
Department of Anatomy, University of Pennsylvania School of Medicine, Philadelphia, Pennsylvania

Bruce Drum
Department of Ophthalmology, George Washington University, Washington, D.C.

Daniel DuVarney
Emory University, Atlanta, Georgia

Raymond C. DuVarney
Department of Physics, Emory University, Atlanta, Georgia

Lennart Enerbäck
Department of Pathology II, University of Goteborg, Goteborg, Sweden

Dale G. Flaming
Department of Physiology, University of California, San Francisco, California

C. R. Gallistel
Department of Psychology, University of Pennsylvania, Philadelphia, Pennsylvania

Ellen M. Johnson
Department of Physiology, University of North Carolina Medical School, Chapel Hill, North Carolina

David C. Joy
AT&T Bell Laboratories, 600 Mountain Avenue, Murray Hill, New Jersey

W. Daniel Kornegay III
Department of Anatomy, East Carolina University School of Medicine, Greenville, North Carolina

Noel Kropf
Department of Biological Sciences, Columbia University, New York, New York

Jeffrey H. Kulick
Department of Computing and Information Sciences, Queen's University, Kingston, Ontario, Canada

Cyrus Levinthal
Department of Biological Sciences, Columbia University, New York, New York

James A. McKanna
Department of Anatomy, Vanderbilt University, Nashville, Tennessee

Cary N. Mariash
Division of Endocrinology and Metabolism, Department of Medicine, University of Minnesota, Minneapolis, Minnesota

Robert Massof
Wilmer Eye Institute, Johns Hopkins University, Baltimore, Maryland

R. Ranney Mize
Department of Anatomy and Division of Neuroscience, University of Tennessee Center for the Health Sciences, Memphis, Tennessee

Walter H. Mullikin
Department of Anatomy, University of Pennsylvania School of Medicine, Philadelphia, Pennsylvania

Melburn R. Park
Department of Anatomy and Division of Neuroscience, University of Tennessee Center for the Health Sciences, Memphis, Tennessee

S. Mark Poler
Departments of Anesthesia and Medicine, University of California, San Francisco, California
Present address: Washington University, St. Louis, Missouri

Max C. Poole
Department of Anatomy, East Carolina University School of Medicine, Greenville, North Carolina

Peter Ramm
Department of Psychology, Queen's University, Kingston, Ontario, Canada

Ingemar Rundquist
Department of Pathology II, University of Linköping, Linköping, Sweden

Eric D. Salin
Department of Chemistry, McGill University, Montreal, Quebec, Canada

Steven Seelig
Division of Endocrinology and Metabolism, Department of Pediatrics, University of Minnesota, Minneapolis, Minnesota

R. L. A. Sing
Department of Chemistry, McGill University, Montreal, Quebec, Canada

Irwin Sobel
Department of Biological Sciences, Columbia University, Fairchild Center, New York, New York

Cameron H. Street
Department of Anatomy, University of Pennsylvania School of Medicine, Philadelphia, Pennsylvania

Oleh Tretiak
Department of Computer Engineering, Drexel University, Philadelphia, Pennsylvania

Contents

Preface xi

Contributors xiii

THE MICROCOMPUTER AS A RESEARCH TOOL 1

1 **The Components and Architecture of the Microcomputer** 3
Raymond C. DuVarney

2 **Programming Languages for the Laboratory** 25
R. L. A. Sing and E. D. Salin

3 **Selection of Hardware and Software for Laboratory Microcomputers** 47
S. Mark Poler, Steven Akeson, and Dale G. Flaming

MICROCOMPUTER USES IN LIGHT AND ELECTRON MICROSCOPY 83

4 **The Reconstruction, Display, and Analysis of Neuronal Structures Using a Computer Microscope** 85
J. J. Capowski

5 **A Microcomputer Plotter for Use with Light and Electron Microscopes** 111
R. Ranney Mize

6 **Quantitative Analysis and Reconstruction of Retinal Ganglion Cells Using a Color Graphics Computer** 135
Franklin R. Amthor

7 **Microcomputer Control for Electron Microscope Systems** 155
David C. Joy

MICROCOMPUTER USES IN MORPHOMETRY 175

8 **Morphometric Measurement Using a Computerized Digitizing System** 177
R. Ranney Mize

9 **A Microcomputer Approach to Point-Counting Stereology** 217
Max C. Poole and W. Daniel Kornegay III

10 **A Computer-Based Video Microscope for Cell Measurement** 233
Daniel DuVarney and Raymond C. DuVarney

SERIAL SECTION RECONSTRUCTION 247

11 **Principles of Reconstruction and Three-Dimensional Display of Serial Sections Using a Computer** 249
Ellen M. Johnson and J. J. Capowski

12 **Serial Section Reconstruction Using CARTOS** 265
Noel Kropf, Irwin Sobel, and Cyrus Levinthal

13 **An Algorithm for Removing Hidden Lines in Serial Section Reconstructions Using MICROS** 293
Cameron H. Street and R. Ranney Mize

MICROCOMPUTER USES IN IMAGING AND DENSITOMETRY 309

14 **Principles of Computer-Assisted Imaging in Autoradiographic Densitometry** 311
Peter Ramm and Jeffrey H. Kulick

15 **A Simple Microcomputer System for Microscope Fluorometry** 335
Ingemar Rundquist and Lennart Enerbäck

16 **Computerized Radioautographic Grain Counting** 355
James A. McKanna and Vivien A. Casagrande

17 **Quantitation of Two-Dimensional Gel Electroretograms by Computerized Analysis** 375
Cary N. Mariash and Steven Seelig

18 **Microcomputer Systems for Analyzing 2-Deoxyglucose Autoradiographs** 389
C. R. Gallistel and Oleh Tretiak

MICROCOMPUTER USES IN ELECTROPHYSIOLOGY 409

19 **A Completely Digital Neurophysiological Recording Laboratory** 411
Melburn R. Park

20 **Real-Time Analysis of Visual Receptive Fields Using an IBM XT Personal Computer** 435
Walter H. Mullikin and Thomas L. Davis

21 **Ophthalmic Electrophysiology with a Microcomputer-Based System** 447
Robert Massof, Bruce Drum, and Michael Breton

Index 467

THE MICROCOMPUTER AS A RESEARCH TOOL

1

The Components and Architecture of the Microcomputer

Raymond C. DuVarney

Introduction

Microcomputer Components

Information Storage

Binary, Octal, and Hexadecimal Numbers

Microcomputer Architecture
Memory Registers
The Bus Structure
The Memory and I/O Map

Microprocessor Architecture
The Internal Registers
The Fetch and Execute Cycle
Machine Language Instructions

Read Only Memory

The I/O Interface
The Parallel Interface
The Serial Interface
The Use of Interrupts by I/O Devices

Microcomputer Software
Interpreted Versus Compiled Code
Assemblers

Interfacing the Microcomputer in the Laboratory
The General Purpose Interface Bus
Nonstandardized Binary I/O
A-to-D and D-to-A Conversion

Conclusions

Introduction

It is difficult these days to define what we mean by the term *microcomputer*. I shall, however, take as a working definition those computers whose arithmetic functions and logical decision-making processes are performed using a microprocessor, which is a single, very large-scale integrated (VLSI) circuit. Dramatic improvements in very large-scale integration technology, particularly the ability to mass-produce circuits containing about 50,000 transistors on a silicon chip less than a square centimeter in area, have revolutionized the computer industry. This miniaturization is largely responsible for the increases in microprocessor operating speed and complexity and decreases in cost that we have witnessed during the past decade. As the development of integrated circuits continues, there is every indication that today's microcomputers will be considered toys in comparison to what will become available before the end of this decade.

These small wonders have proliferated into our homes, offices, and laboratories. They provide us, at affordable prices, tools for research that will continue to revolutionize the way in which science is conducted. In this chapter I want to introduce the reader to the way in which the microcomputer functions. In particular, I want to emphasize what is happening at the hardware or electronic level, not with regard to electric currents and charges, but rather the flow, processing, and storage of information in its most fundamental form within the microcomputer.

Microcomputer Components

Figure 1 shows a block diagram of the basic architecture of a microcomputer. Depicted are the central processing unit (CPU) or microprocessor, a block of random access memory (RAM, here labeled read/write memory), a section of read only memory (ROM), and two input/output interfaces (parallel and serial ports are illustrated in the figure). Each of these components of the microcomputer is interconnected by three types of bus: the address, data, and control buses. The term *bus* refers to a parallel set of conducting lines used collectively to transmit data or control signals. An electronic oscillator, called the *clock*, supplies timing pulses to the microprocessor, triggering each step in the operating sequence of the microcomputer. The microprocessor performs the arithmetic and logic functions that one usually associates with the term *computing*. In addition, it controls the three buses that interconnect all the components of the microcomputer. Microcomputers usually contain two types of electronic memory. RAM may be written to as well as read by the microprocessor. The information it contains can be changed under the control of the microprocessor. On the other hand, ROM may only be read by the CPU. The information it contains never changes. The input/output (I/O) interfaces interconnect the internal components of the microcomputer with external peripheral components such as the keyboard or the printer. Mass storage devices such as floppy disk drives, fixed hard disk drives, or cassette tape recorders also receive and transmit information through an I/O interface.

Information Storage

To understand the operation of a microcomputer (or any computer, for that matter), it is important to have a clear idea of the way in which information is coded and stored within the computer. A computer memory consists of a vast collection

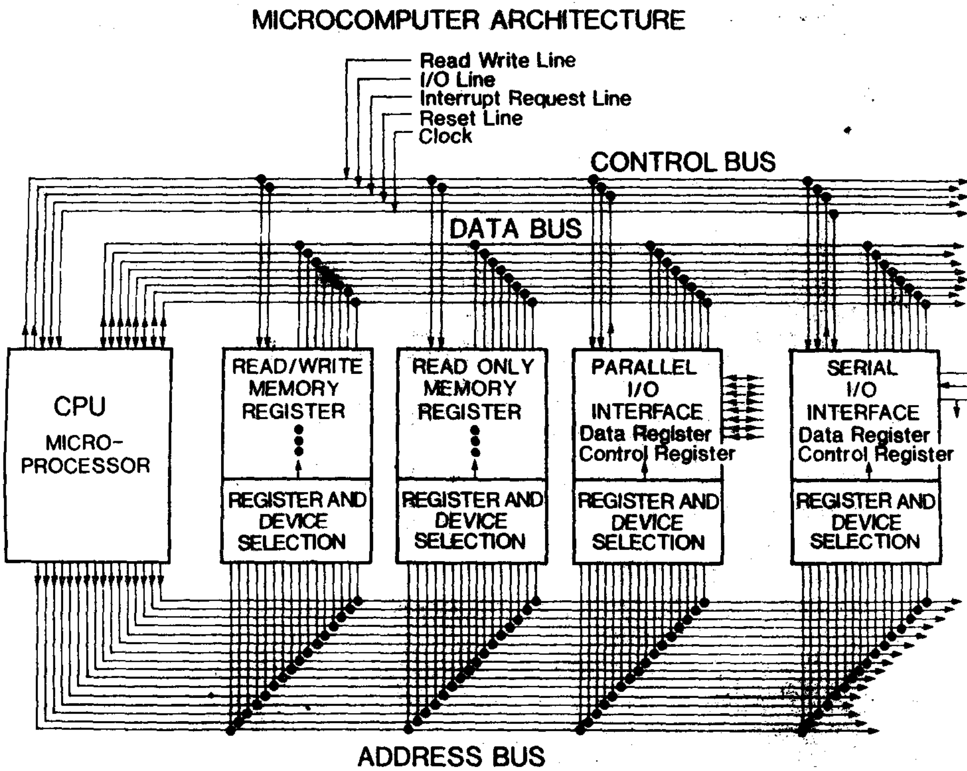

Figure 1. Block diagram illustrating the internal architecture of a typical 8-bit microcomputer. The microprocessor is interconnected to the other system components via the address, data, and control buses. The system components shown include read/write memory (RAM), read only memory (ROM), and a parallel and serial I/O interface. Each component is connected to the address bus through an address decoder, which selects and activates the addressed register and connects it to the data bus so that the microprocessor may read the register's contents or change the register's contents.

of electronic switches that can be opened or closed (written to) by the central processing unit, or read by the central processing unit to "see" which are open and which are closed. Each switch can store one piece of information, which is called a *bit*. There are only two possible states for a switch. The information stored in one bit could therefore represent Yes or No, True or False, On or Off, Plus or Minus, the numbers 0 or 1, or any other quantity that can be represented as one of two possible states.

To be able to represent more complex sets of things, such as letters of the alphabet (26 states) or the decimal integers 0–9 (10 states), the switches are organized into groups called *registers*. Registers contain multiple bits. Most microcomputers available today contain either 8- or 16-bit registers. An 8-bit register is a group of eight switches that functions as if it were one 256-position switch. (Eight two-position switches have 2^8 or 256 possible configurations.) These can range from all eight switches open to all eight switches closed. An 8-bit register holds one *byte* of information. This means it contains one out of 256 possible codes. The types of data that can be represented by one byte of information are practically unlimited; they must only be members of a set of 256 elements or less. Here are some examples.

We shall use 1s and 0s to indicate the state of each bit or switch in the register.

A 1 will represent the on-state and a 0 the off-state. Let us suppose that we stored the pattern 10101010 at some memory register. The 1s and 0s indicate that a bit is set or not set, respectively. This pattern could represent the decimal number 170, which is $(1 \times 2^7) + (0 \times 2^6) + (1 \times 2^5) + (0 \times 2^4) + (1 \times 2^3) + (0 \times 2^2) + (1 \times 2^1) + (0 \times 2^0)$ in *binary code.* Here each bit shows the presence or absence of the successive powers of 2 that comprise a number, and in this fashion all the integers from 0 to 255 may be represented. Of course, the information stored in a register can also represent things other than numbers. For example, 01010000 (binary for 80) represents the letter P using the American Standard Code for Information Interchange (ASCII) or could represent one of 256 possible instructions for the microprocessor. The meaning lies in the interpretation of the code.

Binary, Octal, and Hexadecimal Numbers

All numbering or counting systems have a set of symbols to represent the number of items in a group. The number of symbols one uses is called the *base.* In the familiar decimal or base 10 system there are ten symbols (0–9). If we wish to express a number greater than the number of symbols in our counting system, we do so by counting the number of times each power of the base appears as a constituent of that number. For example, in base 10 the number 111 means $(1 \times 10^2) + (1 \times 10^1) + (1 \times 10^0)$. The same 111 in base 2 or binary means $(1 \times 2^2) + (1 \times 2^1) + (1 \times 2^0) = 7$. In *octal* (base 8) the same 111 would mean $(1 \times 8^2) + (1 \times 8^1) + (1 \times 8^0) = 73$, and in *hexadecimal* (base 16) it means $(1 \times 16^2) + (1 \times 16^1) + (1 \times 16^0) = 273$. The hexadecimal system has a base of 16 and thus requires 16 symbols to express all possible integers. Any 16 symbols would suffice but the convention is to use the ten decimal symbols 0–9 plus the first six letters of the alphabet. As an instructive example we will convert a decimal representation of a number into each of the other representations mentioned above. We begin with the decimal number 205 and first convert this into binary. The first question we must ask is what is the largest power of 2 that is less than or equal to 205? The answer is 128, which is 2^7. In binary this is 10000000. Subtracting 128 from 205 leaves 77. Now we find the largest power of 2 that is less than or equal to 77. The answer this time is 64, which is 2^6 (1000000 in binary). Subtracting 64 from 77 leaves 13. The greatest power of 2 contained in 13 is 8 or 2^3 (1000 in binary). Subtracting 8 from 13 leaves 5. The greatest power of 2 contained in 5 is 4 or 2^2 (100 in binary). Subtracting 4 from 5 leaves 1. The greatest power of 2 less than or equal to 1 is 2^0. (Any number to the power of 0 equals 1; thus, in binary this is 1.) Combining all of the binary constituents of the decimal number 205 yields the following:

Binary Constituents

Decimal Representation	Binary Representation
1 × 128 = 128	10000000
1 × 64 = 64	1000000
1 × 8 = 8	1000
1 × 4 = 4	100
1 × 1 = 1	1
205	11001101

Had we converted this decimal number 205 into an octal or base 8 representation, we would have obtained the following octal constituents.

Octal Constituents

Decimal Representation	Octal Representation
$3 \times 64 = 192$	300
$1 \times 8 = 8$	10
$5 \times 1 = 5$	5
205	315

Similarly the hexadecimal conversion yields the following constituent table.

Hexadecimal Constituents

Decimal Representation	Hexadecimal Constituents
$12 \times 16 = 192$	C0
$13 \times 1 = 13$	D
205	CD

Examining this hexadecimal conversion we see that the greatest power of 16 that is less than 205 is 16^1. Sixteen may be subtracted from 205 twelve times leaving a remainder of 13. The hexadecimal symbol for 12 is C, which appears in the "16s" column of the hexadecimal representation. The remainder of 13 is represented by a D in the units (16^0) column.

It should be clear from our discussion of two-state information storage that the binary representation of a number is the most natural way of storing that number within the microcomputer. However, it is not the easiest way for us mortals to express a number. Base 10 numbers are easy to use but are difficult to convert into binary. Base 8 or 16 numbers, however, may be converted to and from binary representation quite simply. This is because 8 and 16 are powers of 2 whereas 10 is not. We can think of a base 8 or base 16 number as a binary number expressed as groups of 3 or 4 bits, respectively. Let us return briefly to the previous example. The decimal number 205 has the binary representation.

11001101

Let us write this in groups of 3 bits (starting from the right),

11 001 101

and then convert each group into one of the eight octal symbols. The leftmost or most significant group becomes a 3, the middle group a 1, and the rightmost or least significant group a 5. The octal representation is thus

315

Let us now write the binary number in groups of four bits,

1100 1101

and then convert each group into one of the 16 possible hexadecimal symbols. The most significant group (on the left) is a decimal 12 or a hexadecimal C; the other group of four bits is a decimal 13, which is a hexadecimal D. The hexadecimal representation for this number is thus

CD