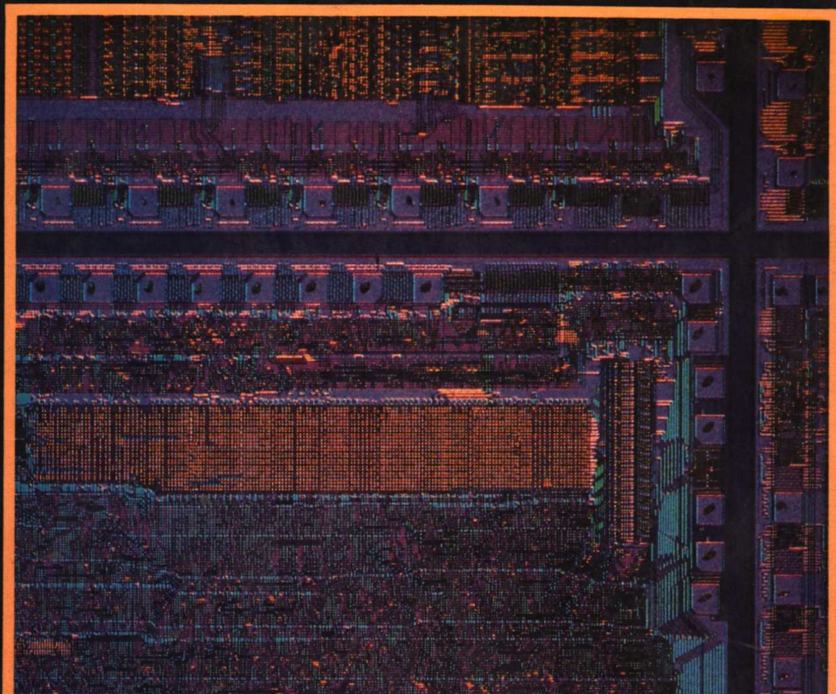
MICRO-PROCESSOR APPLICATIONS HANDBOOK



DAVID F. STOUT

Microprocessor Applications Handbook

David F. Stout *Editor-in-Chief*

Vice President of Engineering
Dataface, Inc.
Santa Clara, California

McGraw-Hill Book Company

New York St. Louis San Francisco Auckland
Bogotá Hamburg Johannesburg London Madrid Mexico
Montreal New Delhi Panama Paris São Paulo
Singapore Sydney Tokyo Toronto

Library of Congress Cataloging in Publication Data

Main entry under title:
Microprocessor applications handbook.

Includes index.
1. Microprocessors. I. Stout, D. F.
TK7895.M5M463 621.3819'58 81-11787
AACR2

34567890 KPKP 898765432

ISBN 0-07-061798-8

The editors for this book were Harold B. Crawford and Ruth L. Weine, the designer was Naomi Auerbach (with binding design by Mark E. Safran), and the production supervisor was Teresa F. Leaden. It was set in Melior by University Graphics. Printed and bound by The Kingsport Press

Contributors

ABRAMS, C. W. *International Business Machines Corp.* (CHAPTER 15)

CANNON, L. E. *Washington State University* (CHAPTER 6)

CARTER, C. W. *Corning Glass Works* (CHAPTER 17)

CASTRO, C. *Casheab* (CHAPTER 10)

GRAPPEL, R. D. *Hemenway Associates* (CHAPTER 11)

HEABERLIN, A. *Casheab* (CHAPTER 10)

KARSTAD, K. *RCA Corporation* (CHAPTER 4)

KOPERDA, F. *International Business Machines Corp.* (CHAPTER 14)

KREAGER, P. S. *Washington State University* (CHAPTER 6)

LANDAU, J. V. *Teradyne, Inc.* (CHAPTER 16)

LE VASSEUR, D. *NASA Ames Research Center* (CHAPTER 2)

LI, K. *Signetics Corporation* (CHAPTER 7)

MELVIN, D. K. *Intel Corporation* (CHAPTER 9)

NEGRO, V. C. *U.S. Department of Energy* (CHAPTER 5)

STOUT, D. F. *Dataface, Inc.* (CHAPTERS 1, 8, 12, AND 13)

WALL, E. L. *ITT Telecommunications* (CHAPTER 3)

Preface

We are in the midst of an era in electronics that requires intelligent instruments, machines, and systems in all technological and scientific fields. These systems obtain their intelligence from microprocessors, microcomputers, and other large-scale integrated circuits operating under control of programs stored in memory. Intelligent systems are designed by a specialized class of individuals who must be talented in both hardware and software. This book is primarily addressed to such a group of people, and to those who are aspiring to become members of the group. These individuals must be familiar with many aspects of both digital and analog electronics; they must be able to design software programs that fully exploit the capabilities of the hardware and understand the many tradeoffs required to achieve the proper balance between hardware and software in each particular system.

During the initial design phase of an intelligent system, designers need to draw upon proven ideas from many reference sources because they cannot afford to "re-invent the wheel" for each new task. The design process often can be expedited or simplified if the designer has access to finished plans of similar applications. Some people may argue the ethics of using other people's designs, but this is the only way the human race makes progress. We must build upon previous designs to expand the state of the art. Typically, the designer may be able to borrow 10 to 20 percent of the design from each of several references. The remaining 60 to 80 percent must be original work. However, even in the remaining "original" design the designer can shorten work time by using ideas from proven concepts.

This handbook clearly presents a wide variety of microprocessor applications, based upon contributions from specialists in many diversified fields. Most chapters contain an ample treatment of both hardware and software aspects of microprocessor system design and discuss specific design information gathered during the development of actual microprocessor applications.

There are literally millions of potential applications for microprocessors, microcomputers, and LSI/VLSI devices. A single volume dedicated to applications of these programmable instruments must, of necessity, be limited to a finite set of devices, technologies, and hardware. This handbook discusses applications using the following types of microprocessors: 1802, 2650, 6800, and the 8080; the types of microcomputer chips included are the 2920, 3870, 3872, 6801, 6802, and 8088. The handbook also covers the following LSI devices operating from a microprocessor data and address bus: parallel ports 6820, 6821, 68488, and 8255; serial ports 6850, 6852, and 8251; LSI instruments discussed in various chapters include analog-to-digital and digital-to-analog converters, communications devices, timers, clocks, and video interface chips.

In years past, real-time processing tasks were difficult to handle using microprocessors because of the machines' slow speed and small data-word size. Recently, many breakthroughs have been achieved using faster microprocessors, better instruction sets, wider buses, and improved algorithms. Several of these real-time tasks are included here. For example, Chapter 3 discusses real-time error correction, which is becoming more important as more compact memory systems are developed. Chapter 6 discusses a smart lumber-grading machine that analyzes each board as it momentarily pauses on a conveyor belt. Chapter 9 covers many concepts in telephony, which requires fast microcomputers and well-designed software to speed up telephone traffic and reduce cost. Chapter 10 presents design guidelines for a digitally programmable waveform synthesizer used in music synthesis. Although the high-speed portions of this circuit use MSI, the functions requiring decisions and input-output in real time are performed with a 6800 microprocessor. Chapter 11 has design guidelines for real-time digital filters and describes low-, high-, and bandpass filters implemented with an A/D converter, a microprocessor board, and a D/A converter. Chapter 14 describes techniques for real-time voice recognition using multiple microprocessors to increase processing speed.

Applications using several standard interfacing circuits are described in Chapters 2, 5, 8, 12, and 13. These include the IEEE-488 bus standard in Chapter 2; the bus is used to interface a programmable calculator to a microprocessor-controlled airborne distance-measuring device. Serial and parallel interfaces using a single-chip microcomputer are thoroughly examined in Chapter 5. A number of A/D converter interfaces to microprocessor systems are explained in Chapter 7. Parallel and serial interfaces are further described in Chapter 12, using both synchronous and asynchronous data transfer. Human interfaces through keyboards and thumbwheel switches are presented in Chapter 13.

Video systems have innumerable application possibilities for microprocessor control. Chapter 4 describes in detail several features of a color TV receiver that may be placed under microprocessor control. Design techniques for various types of video games using a microprocessor to control a specialized game chip are described in Chapter 7. Chapter 15 describes how this technology is used for radio facsimile transmission and in scientific and technical fields because it processes video information at rates compatible with standard microprocessors.

Designing an intelligent system is a form of art. It is difficult to state generally those things that should first be put down on paper. Some designers start with a familiar microprocessor circuit and design the software around it; others work on flowcharts and algorithms long before the logic devices are chosen. There are many advocates for either approach. Chapter 16 is a well-presented example of the "algorithm first" approach, wherein an industrial sewing machine controller is designed using state variable description techniques. Many complex systems can be reduced to more manageable algorithms using the ideas presented in this chapter.

Because of wide usage, microprocessors and microcomputers are no more expensive than most of the other MSI/LSI components on a circuit board, which is why many designers use two or more microprocessors/microcomputers to implement the desired function. In all cases, however, one device must be programmed as the master and the others must be slaves. The master contains the EXECUTIVE or MAIN program, and the other devices process INPUT/OUTPUT routines and other dedicated tasks. Chapter 17 clearly describes one approach to the multiple-microcomputer design philosophy.

It has been a mentally stimulating task to work with the contributors of this handbook for the past two years. Although I have been working with microprocessors for many years, these contributors gave me many new insights concerning the design process. I wish to express my appreciation to these contributors, who spent numerous hours working with me to help produce a useful and widely diversified handbook.

I also wish to express my appreciation to my wife Mildred, and to my two sons Michael and Matthew, who did much of the manuscript typing and proofreading. Many of my colleagues at Ford Aerospace and Communications Corp. and Dataface, Inc., also deserve thanks for their helpful suggestions and moral support.

David F. Stout
Cupertino, California

Contents

Contributors ix
Preface xi

1. Survey of Microprocessor Technology 1-1
The Evolution of the Microprocessor 1-1
1.1 Elements of a Microprocessor 1-3
1.2 Microprocessor and Microcomputer Software 1-11
1.3 Fundamentals of Microprocessor System Design 1-17
References 1-19

2. A Microprocessor-Based Interface for the IEEE-488 Bus 2-1
Introduction 2-1
2.1 The IEEE-488 Standard 2-1
2.2 Implementing a Microprocessor-Based 488 Interface 2-4
2.3 IEEE-488 Application—Distance-Measuring Equipment (DME) 2-8
2.4 DME Principles of Operation 2-17
2.5 Application Summary 2-18
References 2-19

3. Hamming Code Error Correction for Microcomputers 3-1
Introduction 3-1
3.1 Error Detection by Means of Parity 3-2
3.2 Hamming Code Theory 3-4
3.3 A Hamming Equation Calculator/Corrector 3-7
3.4 Real-Time Error Correction (8 Bits) 3-9
3.5 Real-Time Error Correction (16 Bits) 3-13
3.6 A Hardware/Firmware Approach to Hamming Code Correction 3-16
3.7 An 8-Bit Hardware/Firmware Hybrid Circuit 3-17
3.8 Summary 3-22
References 3-22

4. A Microprocessor-Controlled Color TV Receiver 4-1
Introduction 4-1
4.1 Overall Functional Operation 4-2
4.2 Basic Software System Approach 4-7

4.3 Descriptions of Major Functions 4-9
4.4 Task Handling 4-29
4.5 Tuning and Program Selection 4-38
4.6 Battery Backup 4-45
4.7 Summary of Implemented Functions and Commands 4-47
References 4-50

5. Microcomputer Data Acquisition Module 5-1

Introduction 5-1
5.1 Signal Handling 5-2
5.2 System Overview 5-2
5.3 Detailed System Description 5-5
5.4 Major LSI Integrated Circuits 5-8
5.5 Interfacing Between Major Parts of the System 5-15
5.6 Software Overview 5-23
5.7 Memory Considerations 5-26
5.8 Program Routines 5-28
5.9 Hardware-Oriented Routines 5-30
References 5-40

6. A Microprocessor-Controlled Lumber Grader 6-1

Background Information 6-1
6.1 Review 6-2
6.2 Basic System Design 6-3
6.3 Hardware Configuration 6-4
6.4 Software 6-6
6.5 Summary 6-11
References 6-12

7. Programmable Video Games 7-1

Introduction 7-1
7.1 Components of a Video Game 7-2
7.2 Microprocessor Video Game Architecture 7-6
7.3 Examples of Microprocessor-Controlled Video Games 7-9
7.4 Game Examples Using the Object-Oriented Approach 7-28
Reference 7-29

8. Microprocessor–A/D Converter Interfaces 8-1

8.1 Basic Definition of an Analog-to-Digital Converter (ADC) 8-1
8.2 ADC Parameter Definitions 8-8
8.3 Two Examples: Interfacing an ADC with a Microprocessor 8-11
8.4 Free-Running Averaging ADC 8-11
8.5 Software-Initiated ADC 8-16
References 8-18

9. Microcomputer Applications in Telephony 9-1

9.1 Digital Switching Systems 9-1
9.2 Time-Shared Transmission Paths and Control Functions 9-7

9.3 Per-Line Codecs and Channel Filters 9-11
9.4 Voice Digitization 9-20
9.5 Distributed Switching System Architectures 9-22
9.6 Line-Group Architectures 9-24
9.7 A Digital PABX 9-32
9.8 Analog-Signal Processing 9-47
9.9 Digital Subscriber Lines 9-50
9.10 Voice/Data Services 9-54
References 9-57

10. A 32-Channel Digital Waveform Synthesizer 10-1

Introduction 10-1
10.1 Synthesizer Design Philosophy 10-1
10.2 Synthesis Techniques 10-3
10.3 Synthesizer Hardware Design 10-6
10.4 Software 10-12
References 10-24

11. Digital Filters Utilizing Microprocessors 11-1

11.1 Real-Time Analog Processing with Computers 11-1
11.2 Simulating Analog Components Digitally 11-4
11.3 Single-Pole Digital Low-Pass Filter 11-6
11.4 Single-Pole Digital High-Pass Filter 11-9
11.5 Nonrecursive Filters 11-10
11.6 Recursive Filters 11-10
11.7 Hardware Considerations in Digital Filters 11-11
11.8 Software Considerations in Digital Filters 11-13
11.9 Special Computing Devices for Digital Filtering 11-18
11.10 A Practical Digital Resonant Filter 11-19
References 11-23

12. Parallel and Serial Microprocessor Data Interfaces 12-1

Data Transfer Overview 12-1
12.1 The Buffer-Gate Array 12-1
12.2 Latch—D Flip-Flop 12-3
12.3 Parallel Peripheral Interface (PPI) 12-4
12.4 Serial Data Interfaces 12-11
12.5 Universal Asynchronous Receiver-Transmitter 12-13
12.6 UART Application: A Cassette Modem 12-15
12.7 Universal Synchronous Receiver-Transmitter 12-19
References 12-25

13. Keyboard Data Input Techniques 13-1

Overview of Scanning Techniques 13-1
13.1 X-Y Matrix Switch Array 13-2
13.2 A Musical Keyboard Scanner Using a PIA 13-3
13.3 A BCD Thumbwheel Switch Scanner 13-10
References 13-14

14. Voice Recognition 14-1

Historical Overview 14-1
14.1 System Goals 14-2
14.2 System Overview 14-3
14.3 Input-Signal Processing 14-4
14.4 Data Compression 14-6
14.5 Discrete Word-Boundary Determination 14-8
14.6 Definitions of Linguistic Terms 14-12
14.7 Pattern Detection Within a Word 14-13
14.8 Pattern Parameters and Analysis 14-15
14.9 Word-Identification Techniques 14-24
14.10 Hardware Implementation 14-25
14.11 Status 14-26
References 14-26

15. A Slow-Scan Television System Using a Microprocessor 15-1

Introduction 15-1
15.1 SSTV Specifications 15-2
15.2 Analog SSTV Systems 15-5
15.3 Digital SSTV Systems 15-8
15.4 Microprocessor SSTV Systems 15-9
15.5 Microprocessor Hardware Design 15-13
15.6 Microprocessor Software Design 15-20
References 15-29

16. Hardware-Oriented State-Description Techniques 16-1

An Overview of the Technique 16-1
16.1 State-Description Techniques 16-2
16.2 The State Language 16-5
16.3 State Language Translation 16-7
16.4 Application to an Industrial Sewing Machine 16-9
16.5 Implementation of a State Processor 16-13
16.6 Benefits of State-Description Techniques 16-16
References 16-18

17. Multiple Microcomputers in Small Systems 17-1

Single-Chip Microcomputers and Small Measurement Systems 17-1
17.1 Small Measurement Systems Defined 17-2
17.2 Classical Multichip μP Architecture 17-4
17.3 A Dual μC Design 17-4
17.4 Extension to Multiple Slaves 17-10
17.5 Partitioning Tasks between μCs 17-13
17.6 A Task-Scheduling Executive 17-13
17.7 Interfacing Techniques 17-16
17.8 Development Hints 17-22
References 17-22

Index follows Chapter 17

CHAPTER ONE

Survey of Microprocessor Technology

David F. Stout

Dataface Incorporated, Santa Clara, California

THE EVOLUTION OF THE MICROPROCESSOR

Few areas of electronics have experienced the rapid progress now occurring in the field of microprocessors, microcomputers, and associated integrated circuits. The complexity of integrated circuit (IC) devices has doubled approximately every year since the first device was developed in the early 1960s. Projections indicate that, if the present trend continues, devices containing hundreds of millions of transistors per chip will be available in the 1990s. Figure 1 shows the maximum number of components (transistors, diodes, capacitors, and resistors) on state-of-the-art IC chips for the past 20 years. The dashed line indicates the expected chip density for the next 10 years, assuming certain present technical barriers are surmounted.[1] Many difficult barriers have been overcome to bring ICs up to their present, mature state of technology, and humankind will continue to make substantial progress in this area to achieve the dashed line shown in the figure because new processes are being developed continually as each process reaches the end of its capabilities.

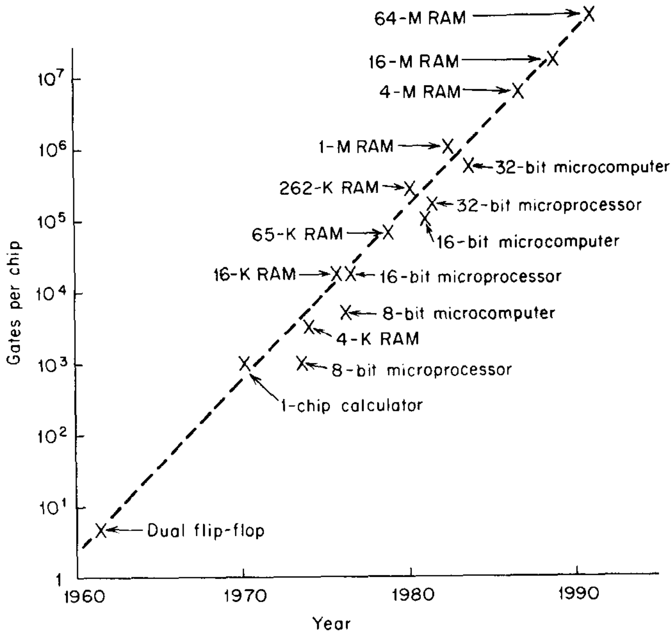


FIG. 1 *With IC chip density doubling every year we may expect to see chips containing hundreds of millions of devices in the 1990s. (From Ref. 1 with permission)*

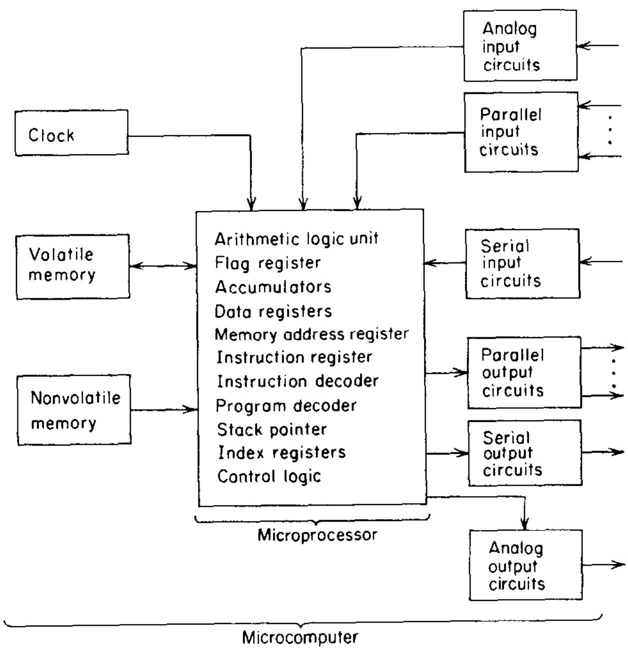


FIG. 2 *Simplified block diagram showing the microprocessor as the major component in a microcomputer.*

In the early 1960s, the first gate was fabricated by using a chip of silicon. As more gates were added to ICs over the years, the functions performed by these devices grew more complex. In just a few years after the first chip was developed, nearly all the logic and arithmetic functions performed by today's microcomputer devices were available to a user on a large assortment of IC devices. The arithmetic logic unit (ALU) chip developed in the 1960s could add, subtract, rotate, and shift, just as a microprocessor or microcomputer device can today. However, not until the early 1970s was the ALU combined (on a single chip) with a sequential circuit (flip-flops, shift registers, and so on) to make a rudimentary microprocessor. This had not been feasible until the IC developers were able to place at least 1000 transistors on a single chip. It took only several more years until input-output (I/O) circuits, read-only memory (ROM), random access memory (RAM), and clock circuits were also placed on the chip; the device was then called a microcomputer.

The line of demarcation between microprocessors and microcomputers is not well-defined. Generally, a microprocessor sequentially executes instructions, whereas a microcomputer executes instructions and provides (1) I/O circuitry for a user in serial and/or parallel modes, (2) ROM or programmable ROM (PROM) on which users may permanently store their program, (3) large quantities of RAM for a user to store temporary data and programs, and (4) a clock circuit requiring only an external crystal or RC network.

A microcomputer chip is sometimes referred to as a "computer on a chip." This term is really a misnomer because no matter how much logic is put on the chip, large devices such as a keyboard, display, and power supply are required. Many microcomputer systems still require several dozen other chips to interface the system with some outside process. And the limited number of pins available on a microcomputer package is a major drawback. Although chips containing millions of gates are predicted for the 1980s, the real problem with using these devices will be interfacing them to the outside world.

Figure 2 illustrates that a microprocessor is the central device in a microcomputer. Some microcomputer chips contain less than all the blocks shown; some contain more. Many special-purpose microcomputers contain on-chip I/O interfaces applicable to only a small class of applications. These interfaces may be keyboard scanners, video monitor circuitry, alphanumeric driver circuits, relay or lamp drivers, and so forth. The potential list of specialized microcomputers is endless.

1.1 Elements of a Microprocessor

***The Arithmetic Logic Unit* (ALU)** As mentioned, the microprocessor was an outgrowth of the ALU. For example, the 74181 is a 4-bit ALU with 75-gate complexity. It performs 16 separate operations on two 4-bit input words, with results appearing on a 4-bit output word plus four miscellaneous outputs. The function SELECT is made on four other input lines. Figure 3 shows the logic representation of the 74181. This device is cascadable up to any I/O word size. The 16 functions performed by the 74181 are listed in Table 1. If control

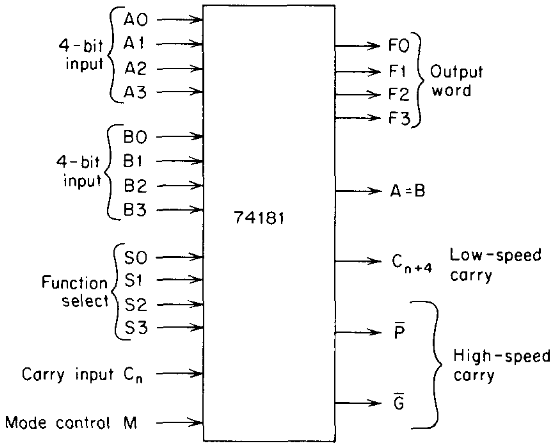


FIG. 3 *Logic representation of the 74181 ALU. (From Ref. 1 with permission)*

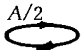
TABLE 1 *Arithmetic operations of the 74181* ALU*

Function SELECT				Arithmetic output function	Logic output function
S3	S2	S1	S0	$M = C_n = 0$	$M = 1$
0	0	0	0	$F = A$	$F = \overline{A}$
0	0	0	1	$F = A + B$	$F = \overline{A + B}$
0	0	1	0	$F = A + \overline{B}$	$F = \overline{A} \cdot B$
0	0	1	1	$F = -1$ (2s complement)	$F = 0000$
0	1	0	0	$F = A$ plus $A \cdot \overline{B}$	$F = \overline{A \cdot B}$
0	1	0	1	$F = (A + B)$ plus $A \cdot \overline{B}$	$F = \overline{B}$
0	1	1	0	$F = A - B - 1$	$F = A \oplus B$
0	1	1	1	$F = A \cdot \overline{B} - 1$	$F = A \cdot \overline{B}$
1	0	0	0	$F = A$ plus $A \cdot B$	$F = \overline{A} + B$
1	0	0	1	$F = A$ plus B	$F = \overline{A \oplus B}$
1	0	1	0	$F = (A + \overline{B})$ plus $A \cdot B$	$F = B$
1	0	1	1	$F = A \cdot B - 1$	$F = A \cdot B$
1	1	0	0	$F = A$ plus $A = 2A$	$F = 1111$
1	1	0	1	$F = (A + B)$ plus A	$F = A + \overline{B}$
1	1	1	0	$F = (A + \overline{B})$ plus A	$F = A + B$
1	1	1	1	$F = A - 1$	$F = A$

*+ = logical OR, · = logical AND, ⊕ = logical XOR.
SOURCE: See Ref. 1.

input $M = 0$, then arithmetic and logic operations are implemented. With $M = 1$, the output word F represents only logical combinations of words A and B. When utilizing a device that performs both "plus" and OR, we must be careful not to use the + symbol for both. In the following discussion we use + for OR and "plus" for addition. The basic operations of a general-purpose ALU are summarized in Table 2.

TABLE 2 Basic operations of a general-purpose ALU*

Function	Description (in any operation A and B can be interchanged)
$F = A$ plus **1**	Increment A (or B)
$F = A - 1$	Decrement A (or B)
$F = A$ plus B	Add A and B
$F = A - B$	Subtract B from A
$F = A \cdot B$	Compute logical AND of inputs
$F = A + B$	Compute logical OR of inputs
$F = A \oplus B$	Compute logical XOR of inputs
$F = 2A$	Shift A left
$F = A/2$	Shift A right
$F =$ 	Rotate A left
$F =$ 	Rotate A right
$F = \overline{A}$	Complement A
$F = 0 \cdot A$	Clear A

*Assume two input buses A and B and an output bus F.
SOURCE: See Ref. 2.

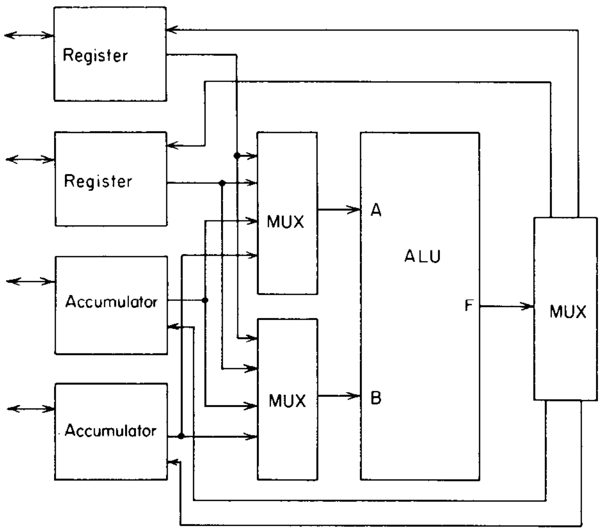


FIG. 4 *By adding registers, accumulators, and multiplexers to the ALU a simple microprocessor begins to formulate. Each line shown represents a data bus of 4, 8, or 16 lines. (From Ref. 1 with permission)*

Registers The ALU in a microprocessor is capable of little more than those operations listed in Table 2. A microprocessor gains its additional power by attaching several storage *registers* to the ALU, as shown in Fig. 4. The ALU can operate on data from only one or two of the registers at a time. Some of these registers are more versatile than the rest and are therefore labeled "accumulators."

Devices called *multiplexers* transfer selected register contents into either ALU input port. Another multiplexer moves the ALU output to a destination register, where it is held until needed by some outside circuit. All these operations are controlled by a sequential circuit that is attached to all registers and multiplexers. For example, for an 8-bit microprocessor, the registers, accumulators, and multiplexers are all 8-bit-wide parallel-in/parallel-out devices.

The next step in the development of a full-scale microprocessor is to add a program counter, condition code register, instruction register, instruction decoder, and memory. Because these devices are all interrelated, they must be added as a group. Figure 5 shows the interrelationship of these circuits. For a microprocessor the memory is typically external to the chip. However, microcomputer devices contain much of the memory on the chip.

Program Counter A typical 8-bit microprocessor has a 16-bit *program counter*. This counter keeps track of the sequence of operations for all microprocessor circuits; that is, the number in the program counter is the line number being executed in the program. This is the memory address which is active at that time. A 16-bit program counter allows programs to be written to a maximum length of 65,535 steps. Most programs are only a few hundred or a few thousand steps long, so the 65,535 limit is seldom reached.

Memory and Instruction Register As indicated by Fig. 5, the program counter does not directly sequence the ALU through various operations. As the program counter sequences through the numbers associated with a program stored in memory, the data word at each corresponding memory location is sequentially presented to the instruction register. Here the data word is temporarily stored while the instruction decoder determines which operation the ALU and/or the program counter must perform. If the instruction decoder discerns a Branch or Jump instruction, the program counter is changed to a new value and the memory is read at that location. The ALU is not accessed in this case. However, if the instruction decoder finds an instruction like one in Table 1 or 2, the ALU and its associated registers are placed into operation. Data moves in and out of the ALU on a data bus having 4, 8, 16, or more parallel lines. This is the same data bus used by memory. Information can flow in either direction on this bus as controlled by sequential logic. In most systems this bus is tristate. All receivers and transmitters connected to the bus are normally deactivated. When a particular instruction is being executed, the instruction decoder decides which transmitter and receiver are activated. During that instruction execution, the bus carries information only between those two devices.

Condition Code Register **(CCR)** While the ALU is performing its task, the special-purpose register called the *condition code register* (CCR) becomes activated. Each bit in this register represents a terse summary of various types of results possible during ALU operations. For example, if an ALU register goes to an all-zero state, then a CCR bit called the zero bit (O or Z bit) is set to the HIGH state. If a carry was generated, then C = **1**. If no carry (NC) was gen-