KOMPUTER KORNER

The first in a series of columns that will illustrate the essential principles of computers.

by TIM BARRY

THE MICROPROCESSOR IS THE HOTTEST device in electronics today. If you doubt this you need only examine any of the current electronics trade magazines. "Microprocessor" has become the "buzz word" of 1975, and you will find it at the base of designs from TV games to home computers. Yet in spite of the features of all of this available hardware, few engineers and even fewer hobbyists are in a position to make effective use of these devices. The reason for this is that most users today lack even a rudimentary background in programming techniques and the design of hardware/software systems. As more and more standard microcomputers become available, the user emphasis will shift from design to being able to use software to get the most out of a given hardware design.

The columns in this series will illustrate the principles that are essential to all types of computer systems, but we will approach them from the hobbyist rather than the industrial point of view. We will discuss and software. If there is sufficient interest, we will talk about what to look for in a hobbyist computer. This field is new, so we will want to hear what *you* want to talk about.

The need to be constantly aware of microcomputers as a hardware/software system is absolutely essential, since unlike most other hardware, a microprocessor and all the peripheral hardware in the world will not accomplish anything unless provided with the programs to tell it what to do. Similarly, a program that is not matched to the available hardware is just a useless abstraction. What we will aim for is to help you get maximum use from any system you can afford. The first several columns in this series will be a mini-course in how to program. If we all have a common level of general programming experience it will make later application-oriented columns easier to understand.

To present these concepts in a realistic manner which can actually be used on existing hardware, all programming in

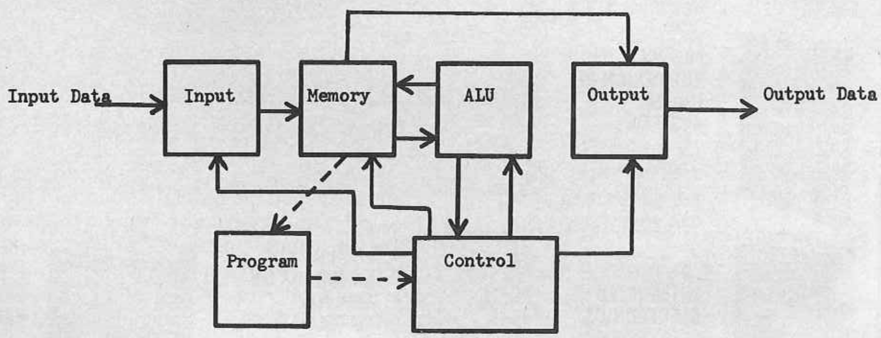


Fig. 1

control, I/O (Input/Output), arithmetic, subroutines, and all the other elements that are combined together to form programs. We will discuss how programs interact with the hardware to form systems and we will look at ways to implement useful features in both hardware

this series will be demonstrated using the 8080-type microprocessor. The primary reason for this choice is that for the non-industrial user, the 8080 is presently the most available second-generation microprocessor. It is available from several sources in both component and finished computer form and support from the various manufacturers is good. Also, the instruction set and architecture of the 8080 are general enough so that the techniques and examples presented here should prove applicable to other microcomputers and minicomputers. The 8080, its instruction set, and other basic information required to program systems using it will be the subject of the second column in this series.

Since this is the first in a series of columns on microprocessor systems, it would be illustrative if we devoted some time to a brief presentation on the basics of computers, software, and the microprocessor itself.

Elements of computers

All digital computers*, from the $9.95 pocket calculator to the largest computer

(continued on page 16)

*Computers of all types are generally divided into two broad groups: digital and analog. Digital computers recognize only two data levels, conveniently represented in binary as "1's" and "0's". Analog computers, on the other hand, deal with continuously-variable data levels. While much of the discussion to follow applies to both digital and analog computers, this basic difference in data representation makes the hardware and software required to implement the two types of computers markedly different. The design of analog computer systems is a complete subject in its own right, and a parallel treatment of digital and analog computation would only detract from both. The microprocessor is part of a digital computer, and henceforth the analog world will be considered as something we convert digital data to or from, never lingering longer than absolutely necessary to perform the conversion.

KOMPUTER KORNER

(continued from page 14)

networks, perform one basic function: they transform data from one form to another using precisely defined procedures. This basic definition immediately makes clear the two divisions of a computer system: hardware and software. Transformations of data are performed by hardware under the control of procedures defined by software. Any computer can now be divided into six blocks which accomplish these basic functions: input, memory, arithmetic logic unit (ALU); control, output, and program. Graphically, this scheme is shown in Fig. 1.

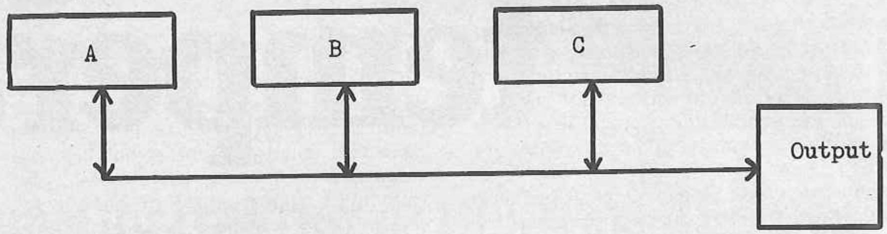
Briefly, the functions of these blocks are as follows:

INPUT: Input devices are used to collect data for processing. These devices can be as simple as toggle switches or as complex as high-speed card readers. Examples would include keyboards, paper tape readers, and various types of magnetic tape units.

MEMORY: Memory devices are used to store input data, programs, intermediate results, and output data. These devices are generally subdivided into serial-access devices such as disks, drums, and tapes and random access devices such as read only memory (ROM) and read/write memory (RAM). (The terms RAM and ROM as they are commonly used can lead to confusion. ROM stands for Read Only Memory, while RAM stands for Random Access Memory. One refers to the storage characteristics of the memory (read only) and the other refers to the way data in the memory is accessed (random access). Most ROMs are also RAMs. To avoid any confusion on this basis the term RAM, if used, will refer to read/write memory. Generally, memory will be referred to as either read-only or read/write, with the implication being that both are semiconductor memories with random access organization.)

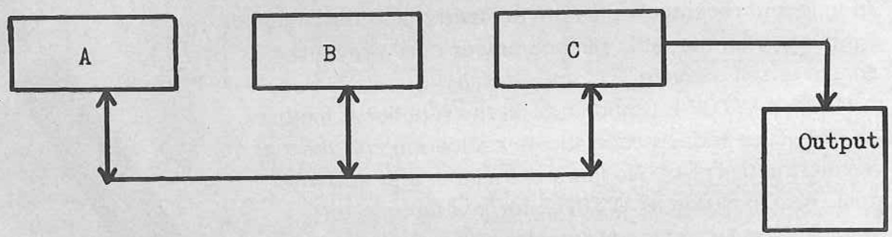
ARITHMETIC/LOGIC UNIT: This is the portion of the computer which actually

(continued on page 88)



Code	Operation
01	Transfer A to B
02	" A to C
03	" A to Output
04	" B to A
05	" B to C
06	" B to Output
07	" C to A
08	" C to B
09	" C to Output

Architecture A



Code	Operation
01	Transfer A to B
02	" A to C
03	" B to A
04	" B to C
05	" C to A
06	" C to B
07	" C to Output

Architecture B

Fig. 2

STATE OF SOLID STATE

(continued from page 80)

the added device.

The value of capacitor C was not given for the circuit but a quick frequency calculation seems to take the form of $f = \frac{1}{2RC} \ln 2$. The fractional logarithm ($\ln 2 = .693$) term accounts for the fact that the oscillator frequency is lowered because the capacitor voltage is exponential and not linear. Exponential charging waveforms are the rule when capacitors are charged through resistors from fixed voltages. The value of C calculates to about 69 pF.

The current sources are not used in this application. Constant current into a capacitor generates an increasing ramp voltage at a uniform rate. Linear sawtooth or triangle generators are designed around this principle. They can be found on the ERG data sheet available from Electronics Research Group, 22 Mill Street, Arlington, MA 02174.

KOMPUTER KORNER

(continued from page 16)

performs the data translations. It consists of registers and arithmetic/logic hardware which can be controlled and tested by the control unit.

CONTROL: The control unit is a logic network which sequentially decodes the program being executed into sequences of signals which control the computer's operation.

OUTPUT: Output devices transmit the processed data from the computer to the outside world. Output devices would include printers, CRT displays, paper tape punches, and magnetic tape devices.

PROGRAM: The program is the means by which the user defines the procedures which he wants the computer system to perform.

The dotted lines linking memory, program, and control are to show that the program controlling the computer is often stored in the same memory as the data being operated on. Indeed, it is possible for a program to treat itself as data and modify its function. Unintentional use of this feature can result in some spectacular program failures and a good deal of debugging time. A microprocessor combines the control and arithmetic/logic portions of the computer on a single integrated circuit. There is some confusion as to the use of the terms "microprocessor" and "microcomputer," so we will adhere to the following definitions: A microprocessor is the integrated circuit which is used to perform the control and arithmetic/logic functions in a microcomputer. A microcomputer is a system consisting of a microprocessor, memory, and I/O interfacing. The 8080, 6800, and F-8 are examples of microprocessors while the Altair 8800, Logical Services Servant-8, and the Intel Intellec 8 Mod 80™ are microcomputers. Generally we will be talking about programming microcomputers, since the microprocessor itself requires considerable support harware to be made into a useful system.

The way a specific computer implements these general computer features is known as the computer's architecture. This includes the computer's register resources, its memory organization, input/output structure, interrupt structure, arithmetic and logic functions, and the data paths by which data transfers and operations using these features are performed. There are many types of computer architecture, and comparisons between machines is a difficult task, even for experienced computer scientists. A detailed discussion of architecture is not appropriate to these columns, although we will discuss the 8080 architecture in some detail in the next article.

As computer users, the most important thing to recognize about architecture is that it has a direct effect on the ease with which you will be able to use the computer. This is because the architecture of the computer determines the functions that you will be able to perform directly with it. The functions that can be performed directly by the computer comprise its instruction set or its machine language. Machine language is simply a pattern of "1's" and "0's" in memory which the control unit decodes to determine the functions to be performed by the computer.

Ultimately all programs result in sequences of machine language codes which are translated and executed by the processor. Thus programming in machine language allows you direct access to all the features provided by the computer's architecture. Now while it is true that cunning programming can overcome the deficiencies of a bad architecture, a good architecture can make your life a lot easier. Consider the two simple register architectures and partial machine languages shown in Fig. 2. If it is desired to move the contents of register A to the output device, a single machine instruction, 03 (move A to Output), would suffice for architecture A. This is because all three registers and the output device are connected directly to the same data path. With architecture B it is not so simple. The three registers can all be freely interchanged, but only the C register can communicate with the output device. Thus two commands are now required to perform the transfer:

02 (move A to C)
09 (move C to Output)

Now if C happened to contain something we needed to save, we would have to add two more instructions to save and restore C (in B, for example).

08 (move B to C)
02 (move A to C)
09 (move C to Output)
04 (move B to C)

Thus, this one simple change in the data paths connecting the registers caused a significant change in the programming required to do output operations. Other small architectural changes can have comparably large effects on ease of computer operations.

There is no generally accepted "best" architecture, so all computers will exhibit characteristics that the designers felt were most important for their application. The architecture of all computers is affected by many compromises which affect speed, I/O, memory size and access, and functions provided. This is even more true of the single-chip microprocessor. Process constraints forced extreme architectural compromises on the earlier microprocessors. More recent processors (8080, M-6800, F-8, etc.) have fewer compromises, but they still exhibit more compromises than computers composed of SSI/MSI devices. The most important thing to remember about architecture is that it is something you have to live with, since it is not possible to alter the internal workings of a microprocessor. Understanding the architecture of the processor you have chosen enables you to use programming to compensate for most of the deficiencies and take advantages of the strengths to gain maximum effective use from your own system. **R-E**

KOMPUTER KORNER 2

by JOHN TITUS, DAVID LARSEN, and PETER RONY*

*This article is reprinted courtesy American Laboratories.

THIS MONTH WE WILL DISCUSS WHAT A MICROprocessor is and how it fits into the general scheme of the controllers and computers that exist today. In the book entitled *Introduction to the Basic Computer*, by D. Eadie, the term data processor is defined as "a digital device that processes data. It may be a computer, but in a larger sense it may gather, distribute, digest, analyze, and perform other organization or smoothing operations on data. These operations, then, are not necessarily computational. Data processor is a more inclusive term than computer."[1]

A microprocessor is a single integrated circuit that contains at least 75% of the power of a computer. It usually cannot do anything without the aid of additional IC's and memory and therefore can be distinguished from a *microcomputer*.

A microcomputer is a full operational system based upon a microprocessor chip that contains memory, latches, counters, input/output devices, buffers, and a power supply in addition to the microprocessor chip. A microcomputer may be a 'black box' with only a single switch: OPERATE/RESET. The 8080 microprocessor chip, for example, is a 40-pin LSI chip (see Fig. 1). A typical system based upon

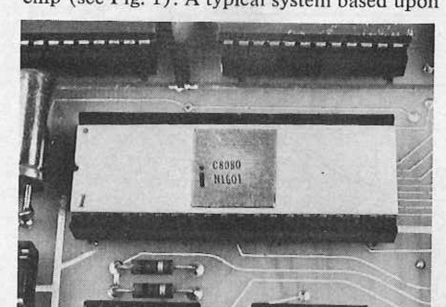


Fig. 1

this chip is shown in Fig. 2 with the 8080 chip located on the CPU board on the left.

A microcomputer has all of the minimum requirements of a computer. For example:

(continued on page 91)