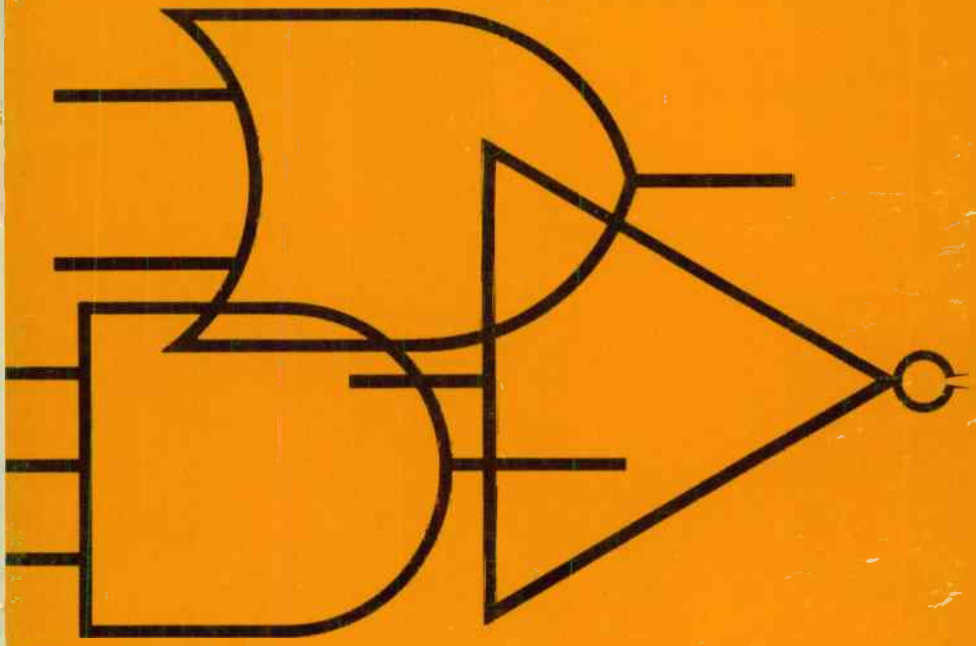
Howard W. Sams PUBLICATION 20808

COMPUTER DATA HANDLING CIRCUITS

by Alfred Corbin



COMPUTER DATA HANDLING CIRCUITS

by

Alfred Corbin

HOWARD W. SAMS & CO., INC.
THE BOBBS-MERRILL CO., INC.
INDIANAPOLIS • KANSAS CITY • NEW YORK

FIRST EDITION

SECOND PRINTING—1974

International Standard Book Number: 0-672-20808-3
Library of Congress Catalog Card Number: 73-135993

PREFACE

The technology of digital data processing systems is certainly the fastest growing segment of the electronics industry. In no other technical field have we witnessed comparable growth and development, all in the relatively short period since the early 1940's.

Predictably, with this frantic growth of technology, even those engineers and technicians in daily association with digital electronics can barely stay abreast of modern techniques, packaging methods, and hardware developments. For the student or novice in the field, starting essentially from scratch is more difficult than ever. Those who make attempts at self-education find a great variety of textbooks available, but most are either too highly theoretical for the average journeyman or too narrow in their scope of subject matter. This seems to be especially true in the subject of semiconductor circuits, which naturally is an important part of any digital circuit study.

In technical schools and institutes, a great amount of time and effort must be expended in the classical approach to electronics—through magnetic circuits, ac circuits, linear circuits, vacuum-tube theory, and semiconductor physics—before the student is introduced to the new and relatively simple study of digital logic circuits. Of course, a thorough knowledge of feedback stability criteria is essential to a servo-system technician, but what of the beginner who wishes to make an immediately productive career of digital circuit work?

This book has been prepared with the idea of satisfying the need for an introductory course in practical data systems analysis.

Very little coverage is given to computers *per se*, although all of the circuits, theory, and techniques described are used directly in the design and construction of computers. In data processing, the computer usually is a relatively small part of the total system hardware. Most of the equipment is involved in the auxiliary process of collecting, shaping, formatting, and transmitting data for use by the computer, and in the conversion of the computer output to usable information or control signals. The material presented here is intended to furnish a good working knowledge of the details and overall functioning of the modern digital data system.

Perhaps the material is not covered in enough theoretical depth for the average electronics engineer; it may serve, however, as an introduction or starting point in a highly complex technology. Hopefully, the material will serve as a framework of theory on which to build for the experienced technician who is not acquainted with digital logic or semiconductor circuit analysis. Finally, the material may present an opportunity for the mathematician, programmer, or software specialist who must be in constant contact with data hardware engineers to become at least conversant in the operational theory of data handling circuits.

ALFRED CORBIN

CONTENTS

CHAPTER 1

CHAPTER 2

CHAPTER 3

CHAPTER 4

CHAPTER 5

CHAPTER 6

CHAPTER 7

1

SEMICONDUCTORS

Athough the concept of large-scale data processing and virtually all of the data processing circuits presently used were in existence in the early 1920's, this major segment of the electronics industry did not have a chance to develop practically until the invention of the transistor. From the initial announcement of the "semiconductor triode" in 1948 until fairly recently, the transistor has evolved from a poor and costly substitute for the vacuum tube to the present stage in which semiconductors have left the vacuum tube far behind, except in a few special applications. Early in the development of transistors, some solid-state data systems were designed and constructed in spite of their relative shortcomings and high cost, because vacuum tubes just could not be used practically in systems involving thousands of stages. Some large vacuum-tube computers had been built prior to the availability of good transistors, but were plagued by tube failures and the monstrous air-conditioning and heat-dissipation problems associated with thousands of tubes.

In this chapter, two fundamental semiconductors will be treated —the transistor in its various forms, and the semiconductor diode. These two classes of semiconductors comprise all of the active and logical circuitry of modern digital computers and data handling systems. With a thorough understanding of only the outward characteristics and properties of the two basic semiconductors, all of the functional circuitry of a large-scale data processor can be developed and analyzed.

SEMICONDUCTOR DIODES

The switching properties of semiconductor, or solid-state, diodes have been understood and applied for nearly a century. Originally used only in power rectifying applications, semiconductor diodes have evolved from bulky metal-and-oxide sandwich stacks to the highly efficient and compact signal and power switching devices available today.

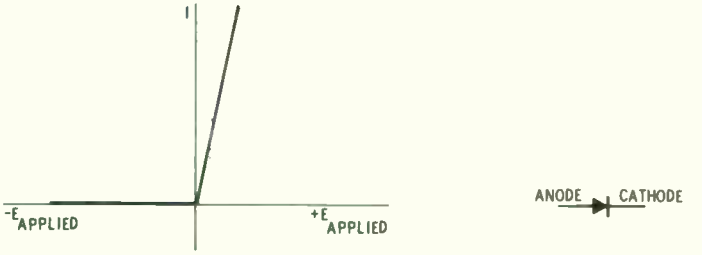
In the applications of interest to the data systems engineer and technician, low-power signal diodes are of most importance. The silicon and germanium diodes in wholesale use in all data systems are an evolutionary outgrowth of the crystal detectors which were popular in the early days of broadcast radio. Using this relatively crude device, the operator would delicately probe a block of crystalline galena with a stiff wire to find a spot with satisfactory rectifying properties as indicated by the strength of the detected signal. With improvements in material selection and control, the germanium point-contact diode developed. In this type of diode, which is still used to some extent, the probing wire is in permanent fixed contact with a crystal of germanium, and the entire assembly is encased in a protective capsule of glass or plastic.

The final major step in the development of signal diodes has been the use of junction construction, in which the diode is formed of two grown layers of the base material, of slightly different chemical and electrical properties. This positive-negative (pn) junction is commonly formed from germanium or silicon crystal. As a result of the rugged nature of the junction, welded leads, and encapsulated construction, modern diodes are extremely durable, stable, and reliable when used within their design ratings.

General Features of Semiconductor Diodes

The most fundamental and useful characteristic of the diode is its tendency to conduct a dc current readily in one direction and block the flow in the opposite direction. Fig. 1-1A illustrates the current versus applied voltage characteristic of an *ideal* diode.

The ideal diode described by Fig. 1-1A has an extremely low resistance in the "forward" direction, in which the anode is positive with respect to the cathode (see Fig. 1-1B). This is indicated by the very high slope of the I/E characteristic on the right side of the diagram. When a reverse voltage is applied, the diode appears to be an open circuit, permitting no current. The action of a diode in a circuit is exactly analogous to that of a check valve in a pipeline, in that it permits current in one direction and blocks current in the opposite direction.



(A) Current/voltage characteristic. (B) Schematic symbol.

Fig. 1-1. Ideal diode.

Although actual diodes do not quite match the performance of the ideal diode illustrated, units are commercially available with forward resistance equal to a few ohms and reverse resistance exceeding 100 megohms. Sometimes, the ratio of back-to-forward resistance is quoted as an indication of the quality of a diode.

A more accurate illustration of the E/I characteristic of an actual diode is shown by Fig. 1-2. In the reverse direction, a slight leakage current, I_R, is shown. As indicated, this current does not vary much with the applied reverse voltage, but is fairly constant at a value of a few microamperes. The forward characteristic indicates that the diode resistance is extremely high until the applied voltage exceeds E_f, above which the diode assumes its low forward resistance. Voltage E_f is often referred to as the *knee* of the diode curve.

Reverse Voltage Limitation—As the reverse voltage applied to a diode is increased, a region is reached at which the diode reverse or back resistance effectively breaks down to a low value. This voltage range is called the *zener* region. If the reverse current at this point is not limited by the external circuit resistance, the diode can be destroyed by internal heating. The value of the reverse voltage above which breakdown may occur is always speci-

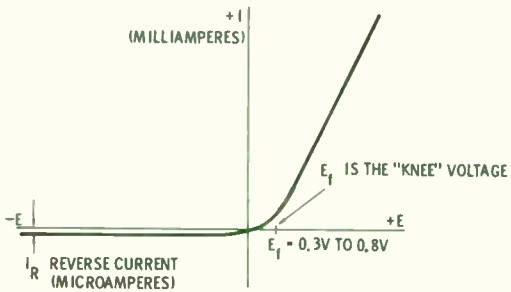
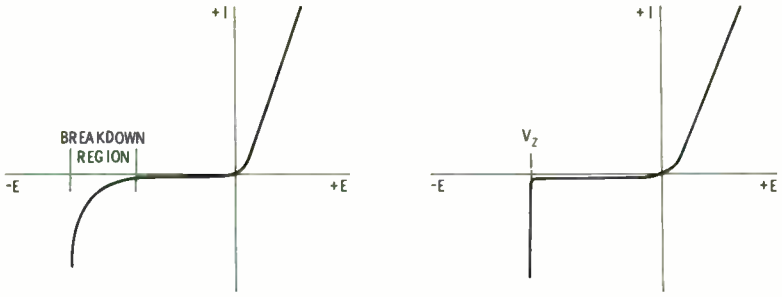


Fig. 1-2. Actual diode characteristic.



(A) Reverse breakdown characteristic. (B) Zener diode characteristic.

Fig. 1-3. Diode reverse breakdown.

fied as a characteristic of any particular diode, and is known as the *peak inverse voltage,* or PIV. As shown by Fig. 1-3A, when the applied reverse voltage approaches the breakdown point, the back resistance appears to decrease gradually to a low value. Diodes have been developed with a very sharp breakdown point and a closely predictable breakdown voltage, and are commonly operated in this region as voltage regulators, with the reverse current limited to a safe value. A diode which is manufactured for this particular application is called a *zener diode*. Fig. 1-3B shows the forward and reverse voltage and current characteristics of a typical zener diode.

Silicon Versus Germanium Diodes

Aside from a cost consideration, in which germanium diodes have some advantage, silicon diodes conform more closely to the ideal than do germanium diodes. Silicon diodes have an extremely high front-to-back impedance ratio. Silicon semiconductors also offer the greater advantage when operation at high temperature is required.

The reverse leakage current, I_R, increases rapidly with temperature. At room temperature, the reverse leakage current of a typical germanium diode might be in the neighborhood of one microampere. An otherwise similar silicon diode may have a reverse leakage current of 0.05 microampere. When maintained at low ambient temperatures, the leakage of the germanium unit is tolerable; at temperatures approaching 100°C, however, the leakage of a germanium diode increases to a value at which it is all but useless, while a typical silicon diode may be safely operated at a junction temperature of 150°C or greater. Fig. 1-4 illustrates the exponential manner in which the reverse leakage current (I_R) of typical diodes varies with temperature.

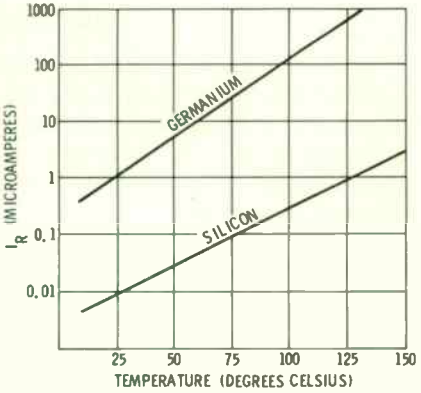


Fig. 1-4. Diode leakage variation with temperature.

Another very significant difference between silicon and germanium diodes is the value of the forward knee voltage, E_f. This voltage drop, which remains quite constant, is typically 0.25 to 0.4 volt for germanium diodes and 0.6 to 0.8 volt for similar silicon units. Many interesting circuit applications, such as low-voltage regulators and signal clippers, make use of the stability of E_f.

Power Diodes

The same principles of construction and operation apply to large high-power diodes as to signal switching diodes. Power diodes, more commonly referred to as *rectifiers,* are available at low cost with current capacities up to 1000 amperes or more and PIV's in excess of 1000 volts. Diodes of this type are seldom, if ever, found in data circuitry, but are universally used in the dc power supplies which must be a part of any practical data system.

Most of the solid-state rectifiers in common use are constructed with one of the elements, anode or cathode, connected physically to a mounting stud, providing a good thermal path to carry internally generated heat to the chassis or mounting plate. This "heat-sinking" permits rectifiers to be used at higher current levels than would be possible with only air convection cooling.

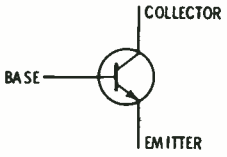
As an outgrowth of the development of silicon power rectifiers, a new class of semiconductor has evolved—the three-element rectifier. This device, known most commonly as the *silicon controlled rectifier* (SCR), exhibits the properties of a diode in the forward direction only when turned on by a third element. SCR's are used extensively in ac motor speed control and some power-supply regulator applications.

THE TRANSISTOR

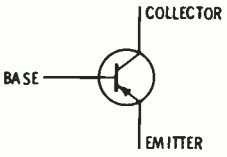
Much has been written describing and analyzing the complex internal physics of semiconductor action. Valuable as this science may be to a designer of new types of transistors, it is really not essential to an understanding of practical circuits. Actually, a clear concept of how a transistor performs outwardly is sufficient for circuit analysis.

Internal Structure

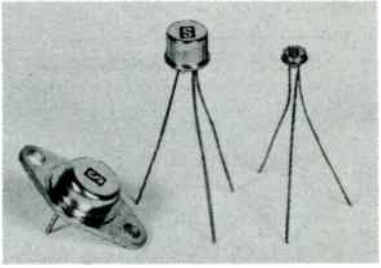
All conventional transistors are three-terminal devices schematically represented as shown by Fig. 1-5. Transistors, with very



(A) Npn: usually silicon.



(B) Pnp: usually germanium.



(C) Various transistor cases.

Courtesy Transistor Div., Solitron Devices, Inc.

Fig. 1-5. Transistor types.

few exceptions, are formed from crystals of germanium or silicon. The working element of a transistor is usually made in the form of a sandwich consisting of three layers, or regions, of the basic material formed in positive-negative-positive (pnp) or negative-positive-negative (npn) sequence. Various other configurations have been developed to improve the physical and electrical properties of the devices, but the fundamental structure always consists of a *base* region in contact with an *emitter* region on one side and a *collector* region on the other (Fig. 1-6). Typically, the whole active portion of a transistor is a chip or cube less than 1/16 inch in its maximum dimension. Most of the overall volume of transistors is devoted to leads, headers, and case, and is usually designed for optimum heat dissipation and mechanical stability.

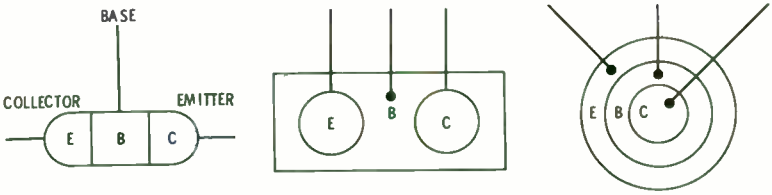


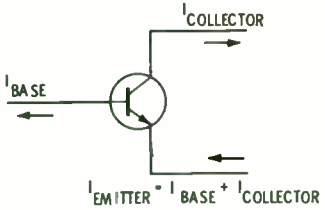
Fig. 1-6. Examples of junction transistor structure.

Current Amplification

In all of its practical applications, the transistor is used as a current-controlling valve. A relatively small current into one terminal, usually the base, proportionally controls a larger current through the collector circuit. This action may be very exactly and rigorously analyzed and defined in terms of an effective internal voltage or current generator and output resistance, or by other theoretical parameters. However, without sacrificing much accuracy, the transistor can be adequately analyzed simply by consideration of its current gain.

Of all the measurable characteristics of a transistor, the most directly useful is its current amplification. Fig. 1-7 illustrates the relationships of the three dc currents in an npn transistor, and the direction of the positive currents.

Fig. 1-7. Transistor current components.



When a transistor is operated within its design limits of current and voltage, and in the correct polarity, the sum of the dc currents *out* of the base and collector terminal is equal to the current *into* the emitter terminal. The current gain of a transistor is defined as the ratio of the collector current to the base current. In most applications, the input signal is applied to the base, and the amplified collector circuit current is used as the output.

The ratio of collector current to base current, known as *beta* (β), or H_{FE}, typically has a value in the range of about 10 to 200. Using 50 as a representative value of β, the following measurements could be made on an npn transistor:

$$I_B = 1 \text{ mA}$$
$$I_B = 51 \text{ mA}$$
$$I_C = 50 \text{ mA}$$

Thus, checking,

$$\beta = \frac{I_C}{I_B} = 50$$

See Fig. 1-8.

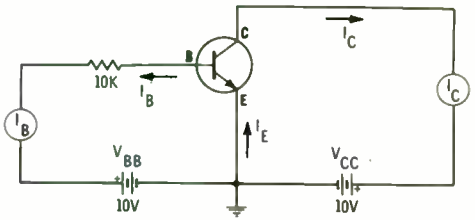


Fig. 1-8. Transistor current relations.

Collector and Base Voltage Connections

If we were to examine an npn transistor with an ohmmeter, the transistor would outwardly seem to consist of two diodes and two resistors as shown in Fig. 1-9. With respect to the emitter terminal, the base terminal has a very low resistance to a positive applied voltage and a very high resistance to a negative applied voltage. The collector-to-emitter path has a high resistance in either direction.

When used in *any* circuit application, the collector diode is always back-biased by the supply voltage, and any current is what would ordinarily be considered "reverse" current in the collector diode. The collector current is directly controlled by a small base-to-emitter current which is in the "forward" direction as seen by the emitter diode.

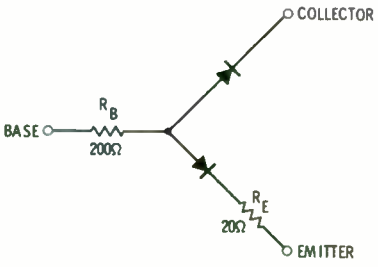


Fig. 1-9. Equivalent circuit of npn transistor.

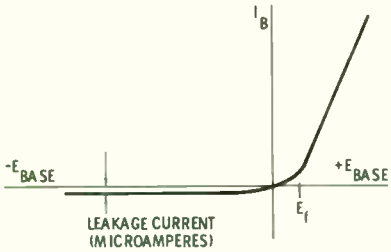


Fig. 1-10. Input resistance of npn transistor.

In the case of the npn transistor, the collector and base are always driven by a voltage which is positive with respect to the emitter. The pnp transistor uses negative supply voltages.

Input Resistance

Considering the base-emitter terminals to be the input and the collector-emitter terminals to be the output, the input resistance appears to be very much like that of a diode across the terminals (Fig. 1-10). When a back voltage is applied to the base, the current is zero, except for the very slight diode-type leakage.

As the base voltage is increased in the forward direction, the current remains negligible until the "knee" voltage, E_f, is reached Above E_f, the input current rises rapidly with applied voltage. Because of this low input resistance above E_f, the base circuit of a transistor must be driven by a source which has a high enough resistance to prevent excessive input current.

The value of E_f is typically 0.2 to 0.4 volt for a germanium transistor and 0.6 to 0.8 volt for a silicon transistor, just as is true of germanium and silicon diodes.

Output Resistance

The effective output, or collector, resistance is extremely high compared to the input resistance. This is indicated by a very small change in collector current as the collector voltage is varied. Fig. 1-11 illustrates this.

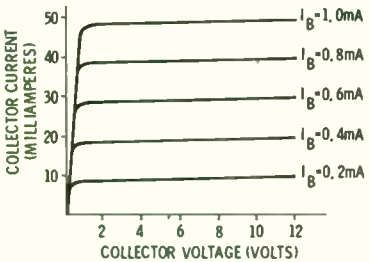


Fig. 1-11. Npn collector characteristic.

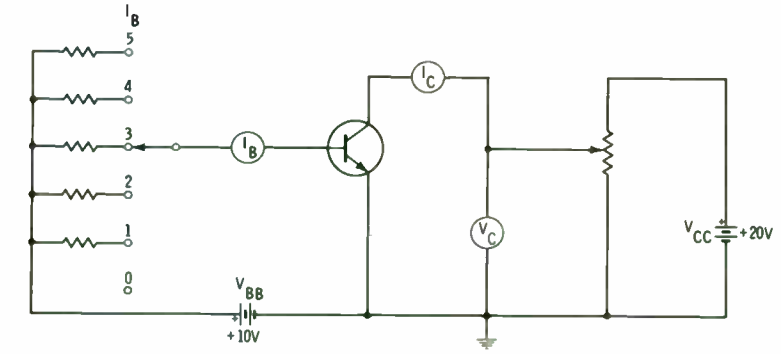
Input/Output Characteristics

Since the transistor is always used to control an output current in accordance with an input current or voltage, the relation of output to input must be known for circuit design or analysis. Consider an ideal npn transistor with a β of 50, in the circuit of Fig. 1-12A.

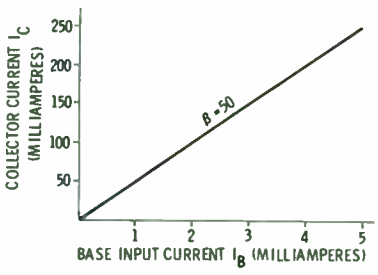
As the base input current is increased from 0 to 5 milliamperes, the collector current varies between 0 and 250 milliamperes, essentially unaffected by collector voltage (Fig. 1-12B). The uniform spacing of the V_C/I_C curves at various I_B values in Fig. 1-12C indicates the direct proportionality of collector current to base current. This is an indication that the value of current gain, β, is constant in the normal operating range.

Grounded-Base and Grounded-Collector Operation

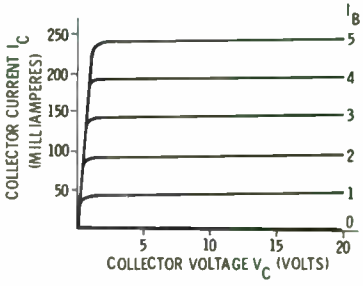
In the preceding descriptions, it has been assumed that the input signal is applied between the base terminal and the grounded



(A) Basic circuit.



(B) Base/collector current characteristic.



(C) Collector characteristic.

Fig. 1-12. Current gain measurement.

Table 1-1. Characteristics of Transistor Circuits

Collection	Voltage Gain	Power Gain	Input Impedance	Output Impedance	Signal Inversion
Grounded-Emitter	High	High	Medium	High	Yes
Grounded-Base	High	Medium	Low	High	No
Grounded-Collector (Emitter Follower)	Less than 1	Low	High	Low	No

emitter, and the output is taken from the collector and emitter circuit. That configuration is the most commonly used in amplifiers; however, two other configurations are possible. In the grounded-base connection, the signal is applied to the emitter and the output is taken at the collector. In the grounded-collector configuration, otherwise known as the *emitter follower,* the signal is applied at the base and taken from the emitter terminal. Each of the three connections offers certain advantages and disadvantages, as shown in Table 1-1.

Because of its obvious advantages, the grounded-emitter connection is most often used when maximum voltage and power gain are needed. In some unusual applications, where signal inversion is undesirable, or where a very low input impedance is desirable, the grounded-base configuration is used.

The emitter-follower circuit is very useful as an impedance matching device. Its high input and low output impedances make it ideal for isolation of input and output lines and as a load driver.

In the circuit shown in Fig. 1-13, the input resistance is equal to approximately $\beta \times R_L$. Thus, if R_L is equal to 1000 ohms and the current gain of the transistor is 50, the impedance looking into the base terminal would appear to be approximately 50 kilohms.

Linear and Saturated Circuits

All active electronic circuits may be classified as operating in either the linear or the saturated mode. In the case of an ampli-

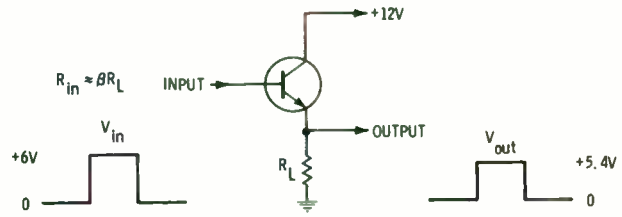


Fig. 1-13. Emitter follower.

fier circuit designed to operate in the linear mode, the output voltage (or current) amplitude is caused to vary in exact proportion to the input signal amplitude, and an effort is made to maintain the output signal within some design boundaries. For example, a linear amplifier which is fed by a +12-volt power supply is always operated at an instantaneous output voltage somewhat greater than 0 and less than +12 volts, and the output signal waveform ideally is an exact, amplified representation of the input waveform. This type of design is necessary, for example, in a high-fidelity amplifier to retain the exact nature of the input signal. A saturated or switching type of amplifier under similar conditions produces an output voltage which is either +12 or 0 volts, depending on the input level, and it never dwells at any midway level. The two conditions of the saturated amplifier are *cutoff* and *saturation*. Cutoff is the state in which the collector current is cut off by absence of base current, and saturation is the condition in which so much base current is provided that the collector current is limited only by the power supply voltage and load resistance. In Fig. 1-14, with the transistor using a +12-volt supply and a 1000-ohm collector load resistor, it is obvious that the maximum possible collector current under any condition is 12 milliamperes. With a transistor β of 50, approximately 12/50 milliampere will produce saturation, and any base current in excess of this value will have no effect.

The saturated mode of operation is used exclusively in digital data handling circuits. Fortunately for the digital electronics specialist, analysis of saturated circuits is much more easily understood than that of linear circuits.

Well-designed digital circuits operate with generous design margins which allow for wide variations in component quality and environmental conditions. For example, a saturated inverting amplifier which nominally operates at cutoff or saturation is usually biased a volt or more beyond cutoff and is driven with several times more base input current than is required for satura-

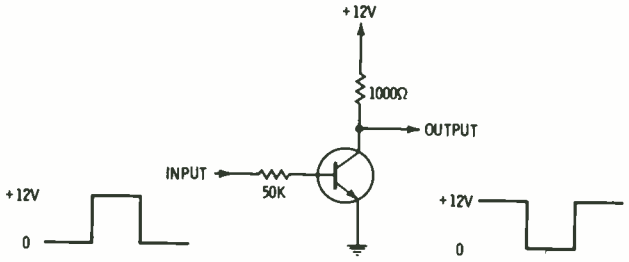


Fig. 1-14. Saturated operation.

tion. To ensure that such an overdesigned circuit will respond to average input signals, the system operating signal excursions are made much greater than would otherwise be required. This brute-force design approach is very commonly used, and is extremely effective from a reliability standpoint. Variations in supply levels, temperature, component quality, and electrical noise will have essentially no effect on a circuit designed with a good margin.

Operating Levels and Polarity

In all of the following circuit descriptions, for the sake of uniformity and coherence, an operating supply level of +12 volts and bias supply of −12 volts will be illustrated. These values are more or less typical, although data systems may be designed to operate at supply levels as low as 2 or 3 volts and as high as 40 volts or more. Selection of power supply levels is generally a matter of practical convenience and economy. Voltages in excess of about 20 volts require the use of relatively expensive semiconductors, while extremely low voltage levels may involve impractically high signal currents and noise problems in many types of transistor circuits.

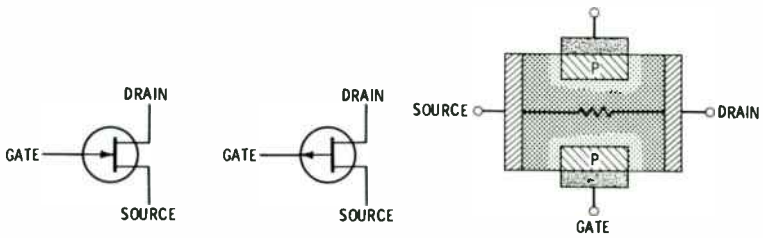
The choice of transistor polarity has been an economic consideration which has changed over the years. All of the early transistors were formed from germanium, which lends itself to pnp polarity. As semiconductor technology improved and developed, low-cost germanium pnp switching transistors of excellent quality became the standard of the industry. Means were also developed whereby good npn germanium units could be produced, but these transistors were much more costly than their npn counterparts, and consequently were used only in the rare cases where complementary circuits were needed. During the "germanium" years, negative operating voltages and pnp devices were the industry standard.

In the early 1960's, better methods of producing silicon npn transistors were developed, making available all of the well-known advantages of silicon at costs as low as the germanium transistors. This has led to a general changeover in design philosophy toward the use of all-silicon npn devices and positive supply voltages. For this reason, all of the following circuit descriptions are based on npn silicon transistors using positive operating supplies and signal voltages. It should be remembered, however, that without exception, all of the npn circuits could be replaced by functionally identical circuits using pnp transistors. The only necessary changes would be in the supply polarities and the diode connections.

Junction FET

In some specialized switching circuits field-effect transistors are used in place of the more common transistors just discussed. Figs. 1-15A and 1-15B show p-type and n-type FET's. The *drain* of an FET corresponds to the plate of a vacuum tube and the *source* corresponds to the cathode. The *gate* of the FET is the controlling element. Action in the FET is essentially the same as in the previously discussed transistors except that an input voltage rather than a current is used to turn on and off the output current. The most commonly used circuit has the source terminal common to both input and output circuits. The input is applied between gate and source, while the output is taken off between drain and source.

Fig. 1-15C shows how an FET works. Without signal and with proper bias, there is a high resistance between the source and drain. A signal of the proper polarity applied to the gates draws charge carriers available for conducting current into the channel between the source and the drain, decreasing the resistance and increasing the output current. This semiconductor is referred to as a *junction FET* (JFET) since the gate makes direct contact with the channel.



(A) N-type schematic symbol. (B) P-type schematic symbol. (C) Operation of FET.

Fig. 1-15. Field-effect transistor.

INTEGRATED CIRCUITS

Although the technology and application of integrated circuits (IC's) do not represent a significant departure from the established methods and theory, the overall impact of IC packaging has been revolutionary. Within a few years of its introduction to industry, the IC, or microcircuitry, has affected electronic packaging to about the same degree as the introduction of transistors to the vacuum-tube world.

Integrated circuits are produced in a variety of forms, both digital and linear. In our area of interest, digital circuits, an IC may

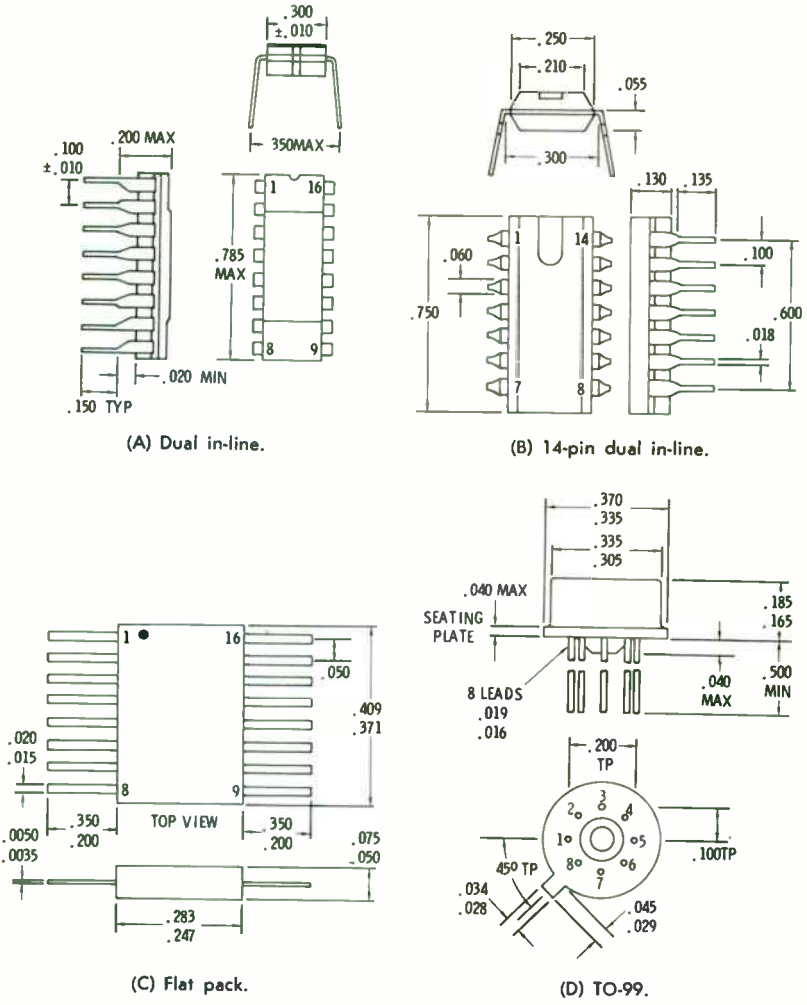
consist of a single logical element such as gate, or it may contain several complex elements in one miniature package. In the extreme, large-scale integration (LSI) produces modules which contain all of the interrelated components and connections of an entire functional block of a computer.

As a matter of definition, an IC is characterized not so much by its miniature size as the method by which it is constructed. Integrated circuits are produced by several processes including chemical, photographic, and various etching and deposition techniques. In the manufacturing process, the entire circuit and all of its internal connections are produced as a single monolithic unit, and not as an assembly of separate components. Resistors, capacitors, semiconductors, and wires, all microscopically tiny, are formed on a chip of base material called a *substrate*. This functional element is then assembled to a header with external leads, and encapsulated. A typical and popular package is the "dual–in–line" style which is roughly ¼ × ¾ × ⅛ inch with 14 or 16 external connecting pins. Several IC package styles are shown in Fig. 1-16.

Obviously, this technique, which enables the designer to package the equivalent of an entire chassis of circuitry into a matchbox-size assembly, has had a profound effect on the outward appearance and size of modern data handling equipment. Philosophically, the use of the IC has made an even greater change in the electronics industry.

Designers and users of IC-based systems have become much less concerned about circuit details and individual components. Since the components of an IC flip-flop, for example, are not accessible, and in fact do not even exist as individual parts, the system designer has no control over, and little interest in, the internal workings of the IC. As a result, entire systems may be designed and serviced by engineers and technicians who have a thorough knowledge of the outward functional and logical aspects of the IC modules, but are required to know only a minimum of electronic theory. From a standpoint of cost, standard integrated circuits are much less costly and more reliable than similar discrete component circuits.

In the theory and logical circuit analysis which follow, no distinction is made between integrated and conventional discrete component circuitry, since no *functional* difference exists. All conventional IC's are of silicon construction, usually operated from a +5-volt power supply, and their internal workings and outward behavior may be considered as identical to that of discrete silicon circuits. Because of the ease with which very complex circuits can be produced by IC processes, the trend in IC is toward the



(A) Dual in-line.

(B) 14-pin dual in-line.

(C) Flat pack.

(D) TO-99.

Fig. 1-16. Common IC package styles.

use of more sophisticated circuit elements than were previously used. For example, the basic computing storage element, the flip-flop, which is described later, is a relatively simple device. Most IC flip-flops, however, operate on a "master-slave" principle which is in many respects a great improvement, but is somewhat more difficult to analyze.

Fortunately, as a general rule, it can be assumed that all of the special techniques used in any commercial IC are very well described in the manufacturer's data.

2

DIGITAL DATA LOGIC

Every electronic control or computing system can be classified as an analog or a digital system, or a combination of the two. In an analog computation, a variable quantity or term is represented by an exact electrical analog which typically is a variable voltage or current level. This analog signal is manipulated and controlled by potentiometers, variable-gain amplifiers, summing networks, and many other devices which simulate or perform arithmetic operations. The final accuracy of the computation is directly dependent on the fidelity of the analog representation and the precision of the circuit components.

An example of a simple electrical analog instrument is an automobile fuel gage, in which the fuel in the tank is represented by a variable resistance which controls a proportional meter. The accuracy of the fuel quantity indication is dependent on the exact proportionality between fuel level and resistance as determined by the float potentiometer and by the quality of the meter.

A similar example is the slide rule. This mechanical analog computer uses length and position as analogs to represent numeric quantities. Again, the quality of the computation is determined and limited by the manufactured precision of the rule and by the operator's skill in positioning the movable elements.

On the other hand, a digital computation is carried out in terms of discrete bits of data, and the significant quality of each data bit is only its presence or absence, not the precise size or quality of the bit. In a digital system, a bit of data may be represented by a pulse, a voltage level, or any other electrical or physical signal whose *presence* or *absence* can be easily detected. An important

distinction between the digital and analog methods is that the overall accuracy of a digital computation is determined not by the precision of each operation but the number of bits and operations used to carry out the computation. In the extreme case, the accuracy of an analog computing system is definitely limited by the available quality of components, while the only accuracy limitation on a digital system is a practical consideration of size and complexity. To illustrate this significant difference, consider a mathematical problem to be solved by the use of a slide rule (analog) or a desk calculator (digital). Assuming that the practical achievable accuracy of the slide rule computation is about 0.1 percent, one cannot improve the accuracy of his computation by using two or three slide rules. However, in the case of the digital operation, an answer which is correct to 20 decimal places can be produced by the use of two 10-place desk calculators, or by using one 10-place calculator in a repeated operation.

Generalizing on the design of typical digital data systems, it may be said that digital systems are composed of a large number of very simple, almost crude, circuit elements. The simple functions performed by the various elements, and the manner in which the elements are interconnected to solve complex problems, are referred to as *digital data logic.*

This chapter is devoted to the theory and analysis of digital logic as applied to data systems, with very little consideration of the electrical aspects of the problems. In subsequent chapters, the hardware implementation, or mechanization, of digital logic will be covered in detail.

BINARY LEVELS

In a digital computation, or in any digital data process, information is usually represented and transmitted as a signal level which is, at any instant, in one of two states. These "binary" states are generally referred to as 1 and 0 levels; however, the terms "on" and "off," or "high" and "low," or "true" or "false" are often used. The output of a battery and switch is an example of a piece of binary signal information—the voltage is either all there or not at all, with no in-between level or condition. A single unit of binary information such as this example is called a *bit,* a short form of the expression *b*inary digit.

It is important to realize that binary digital data is not necessarily restricted to voltage or current, but may be in the form of light or sound, or position, or any other medium which can assume *two distinct states.* Many of the most reliable older forms of communication have used binary signals; for example, the teletype,

flashing signal lights, and even smoke signals. All of these share the common feature of the information being conveyed by the *presence* or *absence* of signal, rather than the precise amount or quality of the signal.

CODES

When binary signals are used to represent numerical or physical quantities, each one-bit signal channel must be assigned a specific value to define its contribution to the overall value of the data in a group of several bits. This assignment of bit weight is known as co*ding*. Without the use of coding, representation of a number varying from 0 to 1000 would require up to 1000 binary signal channels, or bits. Consider, for example, the problem of describing the number 537 with binary information. With uncoded data, this would require 537 wires, or possibly 537 sequential pulses on one wire. However, using a decimal code, this number could be represented by only 30 bits as

00000XXXXX	0000000XXX	000XXXXXXX
5	3	7

We are able to recognize the number 537 in this example because we have the prior knowledge that decimal characters express a number in terms of units, tens, hundreds, and so on. Instinctively, since we are familiar with the decimal system, we know that the number is equal to the sum of the units, tens, and hundreds digits.

In every data system of any practical magnitude, the data must be coded in some manner to keep the size, complexity, and cost within reason. Binary data signals, those with which we will be most concerned, may be coded to represent any numeric value; however, a number of especially useful codes have been developed as standards for instrumentation and data processing. All of the codes which are in common use have been designed or selected to offer specific advantages.

Numeric Codes

Any number can be expressed as the sum of a series of digits in which each digit represents graded exponential powers of some common base number. Complicated as this may sound, it is very easily illustrated. The numeric code most familiar to all of us is the decimal code, in which a number is expressed as a sum of exponential powers of ten. The least significant digit in a decimal whole number gives the number of units, or 10^0. The next digit gives the number of tens, or 10^1, followed by the number of hundreds, thousands, and so on. Since the decimal number system

is built on powers of 10, it is called the *base 10* numeric code. A code based on powers of 2 is called a *base 2* (or binary) code; the base 3 yields a ternary code. Any number may be used as the base of a numeric system.

In all numeric coding systems, the base number also defines the number of states in which each digit can exist. In the decimal code, each digit can have one of 10 states, 0 through 9. In an octal (base 8) code, each digit can have one of eight states.

The only codes which have found extensive practical use in data systems are the decimal and the binary codes and variations or combinations of the two. Binary coding is the most practical from a hardware point of view, since each bit can be very conveniently represented by an electrical device with two states, such as a switch, relay contact, or lamp. The most attractive feature of a decimal system is its simplicity for the designer or operator, because of our familiarity with the handling of decimal numbers.

Natural Binary Code

Among the several commonly used binary coding systems, the simplest and mathematically most efficient is known as *straight binary*, or *natural binary*. In this notation, the number to be coded is described as a sum of powers of 2 representing the coded quantity. One line, or bit, of data is used to represent each of the terms of the sum. An eight-bit natural binary code word, for example, consists of eight separate bits, the first of which represents 2^0, the next 2^1, and so forth to the eighth bit, which represents 2^7. This means that the first bit is worth 1, the next bit 2, the next bit 4, and so on to the eighth bit, which is worth 128.

A binary coded number is written as a series of 1's and 0's, such as 11011001. It is impossible to tell here, without additional information, which end is the 2^0 and which is the 2^7 bit, since no standard arrangement has ever been adopted; however, if we have the prior knowledge that the left-hand bit is the most significant bit (MSB) and the right-hand bit is the least significant (LSB), the number 11011001 can be evaluated as:

$$
\begin{aligned}
1 \times 2^7 &= 128 \\
+\,1 \times 2^6 &= 64 \\
+\,0 \times 2^5 &= 0 \\
+\,1 \times 2^4 &= 16 \\
+\,1 \times 2_3 &= 8 \\
+\,0 \times 2^2 &= 0 \\
+\,0 \times 2^1 &= 0 \\
+\,1 \times 2^0 &= 1 \\
\hline
&\quad 217
\end{aligned}
$$

Expressed in decimal form, the binary number 11011001 is seen as 217. From this example, it should be evident that an eight-bit binary word can be used to express any number from 0 (00000000) to 255, which is 11111111, or (128 + 64 + 32 + 16 + 8 + 4 + 2 + 1).

Reading a binary number at first appears to be an impossibly cumbersome task, and it would be if it were necessary to expand each binary number into its powers of two as illustrated. With practice and constant exposure to binary data, one can become surprisingly adept at reading binary data visually, in much the same way that an experienced Morse code operator instinctively translates a message without consciously recognizing the dots and dashes.

One might question the practicality of using a binary code when it is demonstrated that it requires eight lines of information to represent what can be handled by only three decimal digits. The answer to this is that representation of each decimal digit requires 10 distinctly recognizable states, or signal levels, while the binary bit requires only two states, 1 and 0. Decimal electronic systems have been devised in which the state of each line of information is represented by levels of dc voltage, say from 0 to +10 volts, but this approach requires very complex and critical circuitry, and in fact is a hybrid of analog and digital computation.

The advantages of using binary levels of information in an electronic system are evident in the variety of simple and reliable devices which are inherently of a binary nature. Switches, relays, lamps, diodes, saturated amplifiers, and many other basic electronic and electromechanical components produce outputs with two discrete and distinct states which may easily and conveniently be defined as "on" and "off," or 1 and 0.

Some attempts have been made in the direction of using ternary (base 3) logic for computation, in which positive, negative, and zero levels are the vehicle for data, and the digits are coded in powers of 3. Except in very specialized applications where there is a distinct numerical advantage in a three-level signal, ternary logic is impractical because of its overcomplex circuit requirements.

Combination Codes

It is possible, by the use of special binary codes, to combine the technical advantages of binary coding with some of the practical advantages of other coding methods. One of the most commonly used combinations is the binary-coded-decimal (BCD) notation, in which each digit of a decimal number is expressed as a four-bit binary code. The number 1572, for example, in BCD is:

MSB LSB

0001 1001 0111 0010

The first four bits, 0001, represent thousands, the next group, 1001, represents hundreds, etc. To convert the BCD number 0001 1001 0111 0010, the binary groups are examined and converted separately:

MSB

0001 = 1 thousand
1001 = 5 hundreds
0111 = 2 tens
0010 = 2 units

1572

BCD codes, other than the natural 1-2-4-8/10-20-40-80/100-200-400-800 type, are also in popular use in data processing systems. All, however, share the characteristic of representing each decimal digit by *some* form of binary coding. As will be pointed out later, certain types of unusual binary-decimal coding lend themselves to hardware applications more readily than natural 1-2-4-8 BCD.

PARALLEL AND SERIAL DATA

Notwithstanding the type of coding or the signal characteristics used in a binary data process, each variable data word* must be made up of a fixed number of bits with sufficient numerical capacity to express the maximum value of the data word. At the instant of time in which the data word is to be used or sampled, each of the bits must be recognizable as a 1 or a 0. The data words are transmitted, received, and processed in one of two fundamental formats—serial or parallel.

When data is handled in parallel fashion, one channel is allocated to each bit in the data. Thus, for example, 12 lamps might be used to display a 12-bit *parallel* word, or the output of 12 independent switches could produce a *parallel* dc binary data word.

When binary data is handled in serial format, the data is carried on a single channel, and the bits are presented in a time sequence. Morse code communication is an example of a serial data process. Fig. 2-1 gives an example of a six-bit parallel binary word and a six-bit serial word, each representing the value 110001.

*A *data word* is an ordered set of two or more bits which occupies one storage location and is treated as a unit.

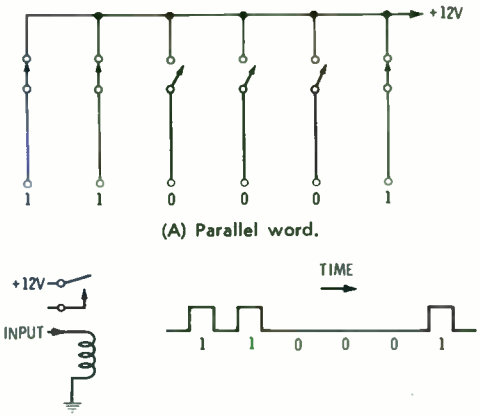


Fig. 2-1. Serial and Parallel data.

Both serial and parallel data handling offer specific advantages. Transmission of data over long distances is one of the best applications of a serial data process because of the obvious economy of a single data channel. In most high-speed data systems, the parallel method is used for the sake of conserving time in transferring and processing data. Most data systems include some serial and some parallel data manipulation, and often use a combination of the two, in which serial data is handled in several parallel channels simultaneously.

In the processing, especially in the reception, of serial data, the receiver must be provided synchronizing information in addition to the actual data. Since each data bit, or pulse, in a serial stream is electrically identical to all other bits, the auxiliary information serves to identify the data bits in the time sequence. Just as with parallel data processing, the user of the data must also have *prior* knowledge of the data coding, the bit weight, and other details of the format.

Without auxiliary information, the serial data word 110001 shown in Fig. 2-1 would be impossible to recognize. However, if a train of identifying pulses is transmitted with the data (on a separate channel) as shown in Fig. 2-2, the location of each bit is established at the receiver.

In this example, the receiver or user of the data must have the prior information that, in this particular format, each word is six bits long, and the MSB, when present, is coincident with the first identifier pulse, and so forth. Many other methods of providing serial bit identification are used in data systems, and are described in Chapter 4.

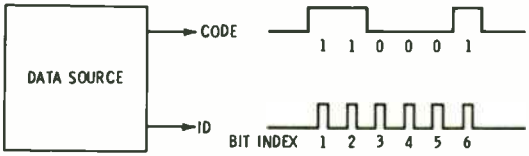


Fig. 2-2. Serial data with identifier pulses.

MACHINE OR SYSTEM LOGIC

The aggregate of all the simple decision-making elements and the manner in which they are interconnected is called the *logic* of a data system. Every data system is designed to carry out a specific pattern or sequence of operations, determined by the logic of the system. Depending on the nature of the problems to be handled, a system may be designed to carry out one specific set of instructions, or it may be designed with sufficient flexibility that its processes are controlled or modified by the operator.

Many levels of control and logic exist in a complex data system. With a large, general-purpose digital computer at his disposal, an operator can determine the overall nature of the computation to be performed; for example, ballistic calculations, business arithmetic, chemical processes, and so forth. Going one step further, he can usually dictate the flow of computation, arranging the major mathematical steps of the process, calling out the flow of data from the input devices or from internal storage, and mapping out the solution in terms of trigonometric manipulations, calculus operations, etc. In some data processing machines, the programming flexibilty ends at this major computational level, while in others the programmer has to synthesize all of the complex operations in terms of simpler functions. In one computer, for example, it might be necessary only to press a button to compute the sine of an angle, while in another, the programmer/operator must call out each arithmetical step of the series which describes the sine function. Generally speaking, most general-purpose computing machines are preprogrammed to the extent that the most common arithmetic operations are performed automatically.

If a computer were designed to offer the absolute maximum in flexibility, practical or not, it would consist of thousands of logical *elements* and a master patchboard through which all of the elements could be interconnected. Each problem to be solved would dictate the overall electrical and logical design of the computer. With such a computer in mind, let us consider the problems involved in setting the computer up to do some useful work. Assuming our problem is a very simple one, say, the addition of

two two-bit binary numbers, we would start with an expression like "the least bit of the sum is 1 if only the least bit of word *A* or only the least bit of word *B* is 1, and the second bit of the sum is 1 if only the second bit of word *A* is 1 and there is no carry, or only the second bit of word *B* is one and there is no carry, or if the second bit of word *A* and the second bit of word *B* are 1 and there is a carry, and so on and so forth. . . ." With a statement like that required to describe a very trivial operation, it would be practically impossible to put into words the details of the logical operations involved in a major problem.

Fortunately, two systems of symbology have evolved by which even the most complicated system can be described and analyzed rigorously without the use of cumbersome language. The first is a special type of logical mathematics known as *Boolean algebra,* with a framework of rules roughly analogous to ordinary algebra, in which binary logical operations are described by manipulation of alphabetical symbols. The second is a system of functional symbolic diagrams which are similar to electrical schematic diagrams.

One of the main advantages of Boolean algebra is its usefulness as a means of simplifying a logical design. Often, when a logical network is first designed (on paper), it contains many more elements than are absolutely necessary. Through algebraic manipulation, other logic configurations can be worked out which accomplish *exactly* the same result as before although using fewer components.

The two main advantages of schematic logic are its simplicity—a logical diagram is really no more than a special type of electrical schematic—and the fact that, if properly understood, a logic diagram can actually be used as a wiring diagram. For a good understanding of data systems in general, an engineer must be thoroughly familiar with logical schematic symbology, and have at least a good working acquaintance with the principles of Boolean algebra.

ELEMENTS OF DIGITAL LOGIC

There are four elemental logic functions which, when used in sufficient quantity, and in proper combination, can mechanize virtually any digital computation. These are the *and, or, negation,* and *memory* functions. Each represents a unique logical relationship between input and output.

The AND function, sometimes called *conjunction* or *coincidence,* yields a 1 output when *all* of its inputs are simultaneously in the 1 state. This can be illustrated by a series connection of switches, as in Fig. 2-3.

Fig. 2-3. Illustrating AND function.

The output is 1 (or "on") only when all three switches are on. Under any other condition, the output of this three-element AND circuit is zero.

The OR function, also known as *disjunction* or *logical mixing,* yields a 1 output when one or more of its inputs are in the 1 state. The OR function may be illustrated by three switches in parallel, as in Fig. 2-4.

Logical negation, or *inversion,* is the property of an element to produce a 1 output when its input is 0 and to produce a 0 output when its input is 1.

The memory function may be described as an element's property of remaining in a given binary state until forced into the opposite state. The mechanical action of a toggle switch has this property. When the switch is turned on, it remains on even after the actuating force is removed. When turned off, it remains off. The toggle switch also has the useful property of being bistable. Its mechanism is so arranged that it is always fully on or fully off, and cannot remain in an in-between condition.

Using only the four basic logic functions, all of the active parts of any large computer can be assembled. Practical data processors, in fact, are composed simply of thousands of elements of this type, usually packaged as special combinations, plus a small amount of incidental auxiliary circuitry.

Boolean Algebra

In the symbology of Boolean algebra, which is named for its creator, George Boole, variables are represented by letter symbols as in ordinary algebra. In much the same manner as manipulations and simplifications of expressions are performed in conventional algebra, Boolean terms and equations may be manipulated by the use of a simple framework of rules.

As a starting point from which the commonly used Boolean theorems are derived, one must accept a set of basic logical postulates, or definitions. Since we are concerned only with binary functions, the principle of duality comes first. This principle,

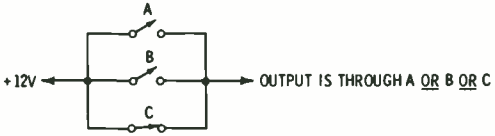


Fig. 2-4. Illustrating OR function.

in simple language, is the understanding that if a term is not 0, it must be 1, and vice versa. Algebraically,

$$A = 0 \text{ if } A \neq 1$$

and

$$A = 1 \text{ if } A \neq 0$$

The AND function is symbolized as multiplication, although this is not to be confused with arithmetic multiplication.

$$A \cdot B = A \text{ AND } B$$

The operational symbol for AND may be the dot ($\cdot$) as shown, or the times sign ($\times$), or the absence of any symbol between terms, as:

$$A \times B$$
$$A \cdot B$$
$$AB$$

Certain postulates relating to the AND function must be accepted for Boolean analysis:

$$A \times 0 = 0$$
$$A \times 1 = A$$
$$A \times A = A$$

Substituting the values 0 and 1 for A gives the binary multiplication table:

$$0 \times 0 = 0$$
$$1 \times 0 = 0$$
$$0 \times 1 = 0$$
$$1 \times 1 = 1$$

The OR function is symbolized by a plus sign. This, too, is not to be confused with the conventional arithmetic symbol:

$$A + B = A \text{ OR } B$$

The basic OR postulates are as follows:

$$A + 0 = A$$
$$A + 1 = 1$$
$$A + A = A$$

Substituting the values 1 and 0 for A gives the binary addition table:

$$0 + 0 = 0$$
$$1 + 0 = 1$$
$$0 + 1 = 1$$
$$1 + 1 = 1$$

Negation, or logical inversion, is symbolized by a bar above the term to be negated, as $\overline{X}$, or by the prime (′) symbol, as X' or verbally as "NOT X." Thus,

$$1' = \text{NOT } 1 = \overline{1}$$
$$1 = \overline{0}$$
$$0 = \overline{1}$$

Much of the usefulness of Boolean algebra is in its application as a means of manipulating logical expressions into various other forms, without changing the meaning, or value, of the expressions. This is extremely valuable in the design of data systems when used to "minimize" terms for the sake of simplifying or reducing the amount of hardware required to make a computation. In a similar application, Boolean manipulation is widely used in fitting a logical expression to an available type of hardware.

One of the most helpful Boolean manipulations is common factoring, which is identical with the corresponding operation in conventional algebra:

$$AB + AC + AD = A(B + C + D)$$

Also identical with the familiar algebraic law is expansion of terms:

$$(A + B)(C + D) = AC + AD + BC + BD$$

Several of the general theorems of Boolean algebra illustrated below may be easily proven or demonstrated by the basic postulates:

$$A + \overline{A} = 1$$
$$A \cdot \overline{A} = 0$$
$$A + AB = A \text{ (factor it out and see)}$$
$$A + \overline{A}B = A + B$$
$$AB + BC + C\overline{A} = AB + C\overline{A}$$
$$(A + B)(B + C)(C + \overline{A}) = (A + B)(C + \overline{A})$$

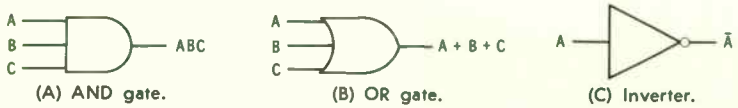
Probably the most often used of all of the Boolean general theorems is the principle of inversion of expressions, known as *De Morgan's theorem*:

$$\overline{AB} = \overline{A} + \overline{B}$$

and

$$\overline{A + B} = \overline{A}\,\overline{B}$$

De Morgan's theorem states, in effect: "To find the inverse of an expression of several terms, invert each of the terms and change the AND's to OR's and the OR's to AND's."



(A) AND gate. (B) OR gate. (C) Inverter.

Fig. 2-5. Element symbols.

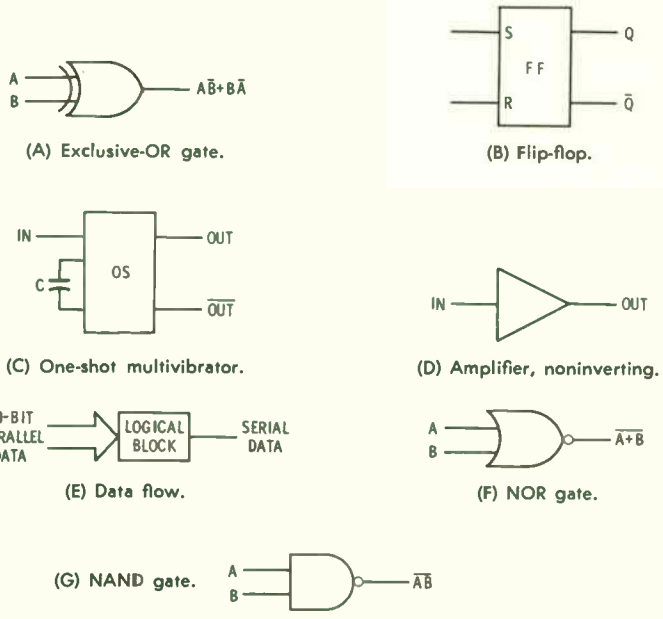
Extreme care must be used when applying De Morgan's theorem to complex expressions. For example,

$$\begin{aligned}(\overline{X+Y})(\overline{W+BZ}) &= (\overline{X+Y}) + (\overline{W+BZ}) \\ &= \overline{X}\,\overline{Y} + \overline{W}(\overline{BZ}) \\ &= \overline{X}\,\overline{Y} + \overline{W}(\overline{B}+\overline{Z}) \\ &= \overline{X}\,\overline{Y} + \overline{W}\overline{B} + \overline{W}\overline{Z}\end{aligned}$$

Note, in applying this theorem, that $\overline{AB}$ is not the same as $\overline{A}\,\overline{B}$.

Logical Symbology

The functional workings and interconnections of a digital system can be described by the use of Boolean representations or by a special form of schematic diagram. While the Boolean symbology is concise and very valuable in system analysis and design, the graphic method is more easily analyzed and serves more readily as a working tool.



(A) Exclusive-OR gate. (B) Flip-flop.

(C) One-shot multivibrator. (D) Amplifier, noninverting.

(E) Data flow. (F) NOR gate.

(G) NAND gate.

Fig. 2-6. Logical symbols.

In the relatively new science of digital data handling, no system of diagramming has been universally adopted, but several major steps have been made in this direction. The most frequently used logic elements are usually represented by specific schematic symbols as shown in Fig. 2-5, while the more complex elements and logical function blocks are most often represented simply by blocks with descriptive names or initial designations, several of which are shown by Fig. 2-6.

(A) With AND gate. (B) With OR gate.

Fig. 2-7. Term inversion.

The O symbol is often used to designate inversion of a single term, implying that a specific input or output includes an inverter which is not otherwise indicated. Fig. 2-7 shows two examples of term inversion.

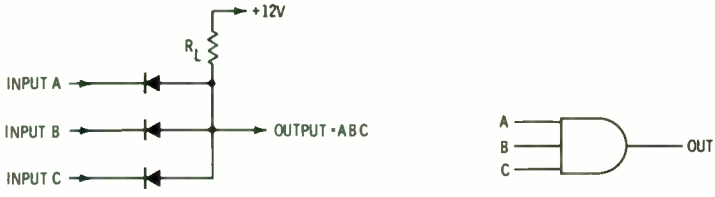
3

ELEMENTARY LOGIC CIRCUITS

Every digital computation or data manipulation, however complex, can be expressed as a sequence or combination of very simple steps defined by the basic rules of digital logic. In the hardware of any digital system, there are a few elemental circuits. Each circuit is designed to perform one of the basic functions, and the overall system is an interconnection of however many elements are required to solve the problem at hand. The main difference between a very large computer and a system designed to perform a minor data manipulation is only the number of interconnected logic elements.

BASIC FUNCTIONS AND LEVELS

Each of the basic logical functions, AND, OR, etc., may be electrically expressed, or mechanized, by the use of a variety of circuits. Through a process of evolution, however, certain circuit forms of such magnificent simplicity have been developed that they are almost universally used in the design of digital data processing machines. In this chapter, many of the most commonly used basic logic circuits are described and functionally analyzed. For the sake of simplicity and standardization, all of the illustrated circuits are assumed to operate on a 12-volt "positive true" level. This means that a "1" is represented by +12 volts dc and a "0" is represented by 0 volts. Most of the circuits could be operated "negative true" or at a different signal amplitude by some minor



(A) Schematic diagram. (B) Logic diagram.

Fig. 3-1. AND circuit.

circuit changes, such as reversing the diode polarities and, in some cases, substituting pnp for npn transistors.

AND GATE

This very basic logic element is used to detect coincidence of logical 1 inputs. An AND gate may have any number of inputs, but, by definition, the output is logical 1 only when *all* inputs are 1.

Electrically, AND gates can be designed in many forms. The simplest and most commonly used type is the diode gate shown in Fig. 3-1.

If all of the inputs are logical 1 (+12 volts), the diodes have no effect, and the output terminal voltage is pulled up to logical 1 by the load resistor. If any one of the inputs is at zero, its diode clamps the output to zero, preventing the output voltage from rising. Fig. 3-2 shows the action of a three-input AND gate under dynamic conditions.

In the type of AND circuit illustrated, since the output is pulled up by the load resistor and down by the inputs, the output impedance for a 1 output is equal to the value of the load resistor, and for a 0 it is equal to that of the input signal. This is sometimes evident in the rise and fall times of the AND gate output (Fig. 3-3). If the input signal has a low 0-level impedance, which is characteristic of most inverters and flip-flops, the AND gate output will exhibit a much slower rise time than fall time.

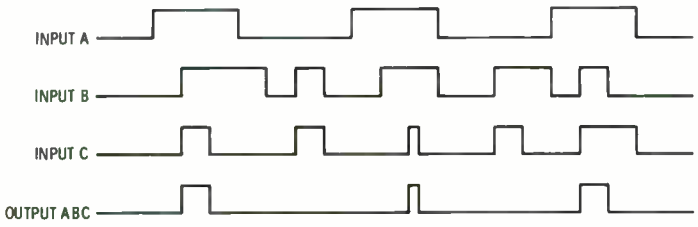


Fig. 3-2. AND gate signals.

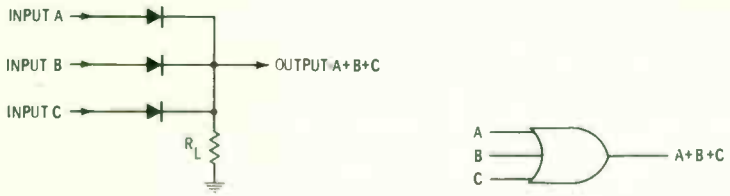
(A) Circuit. (B) Output waveform.

Fig. 3-3. Effect of output resistance.

OR GATE

The OR gate is sometimes called a *logical mixer*. An OR gate may have any number of inputs, and by definition produces a 1 output when one or more of its inputs are a logical 1.

The simplest and most often encountered OR gate circuit is shown in Fig. 3-4. A +12-volt input level is passed through the input diode, pulling the output up to +12 volts. Two or more +12-volt inputs produce exactly the same output effect as only one. Fig. 3-5 illustrates the action of a three-input OR gate under dynamic signal conditions.



(A) Schematic diagram. B) Logic diagram.

Fig. 3-4. OR circuit.

PULSE GATE

In connection with logic circuits, there is often a need for a controlled circuit element which will pass or block the passage of a pulse. This function can be satisfied by the diode pulse gate shown in Fig. 3-6.

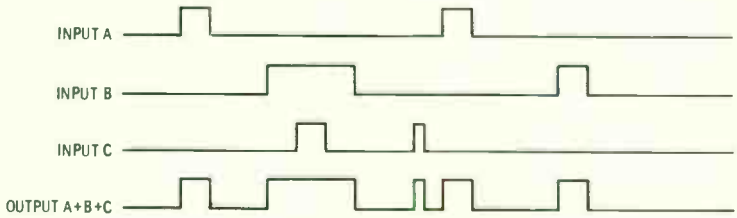


Fig. 3-5. OR gate signals.

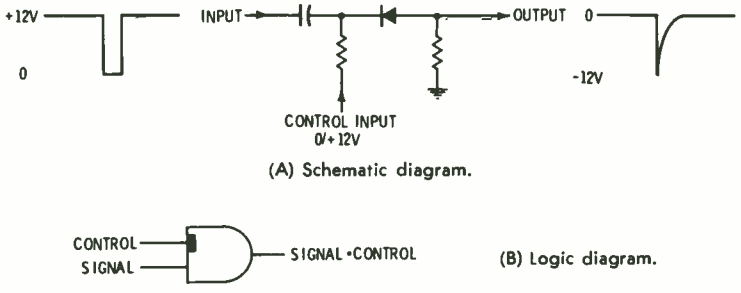


Fig. 3-6. Pulse gate circuit.

If the control input is at zero, the diode is not back-biased and the negative pulse is passed through the diode. If the dc control input is +12 volts, however, the diode is back-biased and the negative pulse is not of sufficient amplitude to overcome the bias. It should be evident that in this circuit, operation depends on the control level being somewhat greater than the pulse amplitude. For example, if the pulse amplitude were −15 volts, an output pulse of approximately −3 volts would pass even when the gate is blocked. In the circuit shown in Fig. 3-6, the pulse is blocked, or inhibited, by the presence of a logical 1 at the control input. The functional expression for this particular circuit, therefore, is

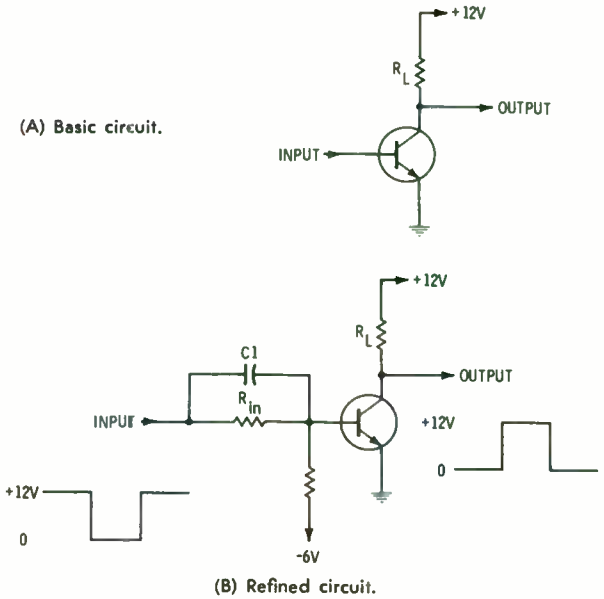
$$\text{Output} = \text{Input} \cdot \overline{\text{Control}}$$

The pulse gate is really not a unique logic element in that it is a special form of the AND gate, sometimes known as the ac AND gate. There are many variations of this circuit, often with several dc inputs AND'ed to control one pulse line. Most pulse gates, however complex, operate on the same general principle of blocking a signal by the use of a back-biased diode in the signal path.

INVERTER

The basic element around which almost all *active* circuits are designed is the inverting circuit. Shown in its simplest form in Fig. 3-7A, the inverter consists only of a transistor and load connected in the grounded-emitter configuration. Small refinements may be added, such as an input cutoff biasing resistor, a current-limiting resistor, and perhaps a speed-up capacitor, as shown in Fig. 3-7B, but the circuit remains essentially the same.

This circuit has several applications, two of which are logical inversion (changing a 1 level to a 0 level) and signal isolation. However, the inverter finds its greatest application as a part of more complex circuits such as flip-flops.



(B) Refined circuit.

Fig. 3-7. Inverter.

In this saturated circuit, the transistor is always either completely cut off or conducting so hard that all of the supply voltage is dropped across the load resistor.

Fig. 3-8 shows typical voltage and current levels for the on and off states of an inverter.

When in the on condition, enough base input current must be supplied to cause the collector to drive to saturation, or essentially zero output voltage. This is determined by the design values of supply voltage, input and load resistance, and the transistor characteristics. In the illustrated case, with a 12-volt power

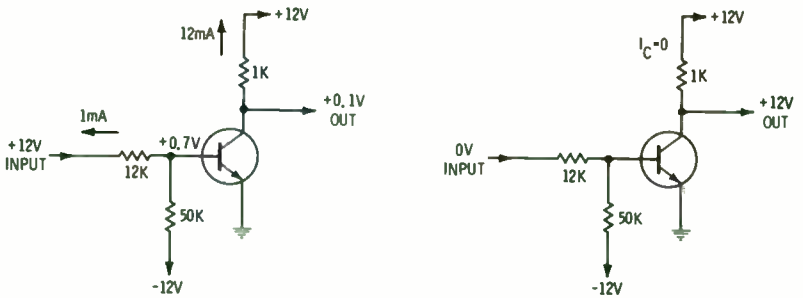


Fig. 3-8. Inverter on and off states.

supply and a 1-kilohm load resistor, the collector current must be equal to 12 V/1 Ω or 12 milliamperes, to pull the output level to zero. A collector current of 12 milliamperes requires a base input current of 12 mA/β. Using a value of 50 as a typical β, this calls for a base current of 12/50 or 0.24 milliampere, as an absolute minimum for the saturation condition.

In the design of a saturated circuit such as the inverter, it must be remembered that transistor parameters such as β will vary considerably between transistors, and may also vary with age, temperature, and other factors. For this reason, when computing the required input current for saturation, "worst-case" figures are used, and a safety factor is applied after computation. In the illustrated circuit, although a base current of only 0.24 milliampere is theoretically required for saturation, the actual base current shown is approximately 1 milliampere. The excess base current simply provides a design safety factor for a transistor with extremely low β.

When the inverter is off, the negative bias voltage holds the base at slightly below zero level, ensuring that the transistor will have no tendency to conduct, even in the presence of noise on the input signal. This small negative bias also is a design safety factor, since theoretically the transistor will not conduct unless the base voltage rises to at least about 0.6 volt above the emitter voltage.

The input resistance of the inverter illustrated is determined almost entirely by the value of the input resistor, since the forward base-to-emitter internal resistance is very low compared to the value of the input resistor.

The output impedance of the inverter is equal to the load resistance when the transistor is cut off, and it is quite low—in the neighborhood of 50 ohms—when the transistor is saturated. This large variation in impedance from a 1 to a 0 is noticeable when fast switching between states is attempted. As shown by Fig. 3-9, when the inverter has to drive a capacitive load or high wiring capacitance, the switching time from +12 volts to 0 is much faster

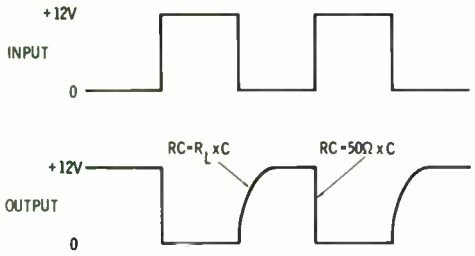


Fig. 3-9. Effect of load capacitance.

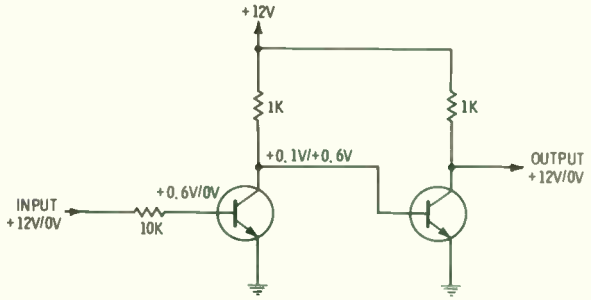


Fig. 3-10. Direct-coupled cascade inverters.

than in the opposite direction because of the great difference in RC time constants. Swinging upward, the time constant is $R_{load} \times C_{load}$; swinging down to zero, the time constant is $R_{saturation} \times C_{load}$.

In some circuit applications, usually for the sake of simplicity or economy, inverters are directly connected in cascade as shown in Fig. 3-10. This direct connection is possible and safe because the collector voltage of the first transistor, when saturated, is considerably less than the conduction threshold voltage at the base of the second transistor. With the first transistor saturated, its collector voltage, about +0.1 volt, cannot cause the second stage to conduct. When the first stage is cut off, the second stage is driven to saturation by the current through the first collector load resistor.

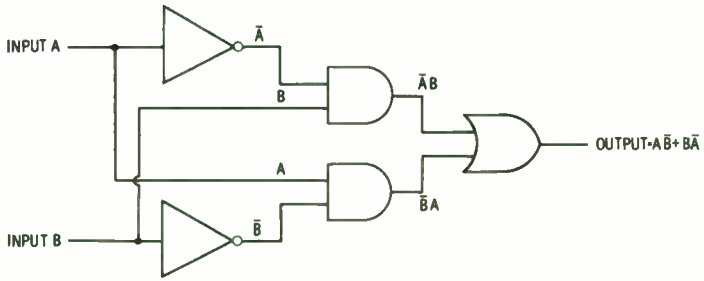
Obviously, the circuit shown in Fig. 3-10 is not used for signal inversion, since the output is doubly inverted, which makes it logically equal to the input. Cascaded inverters, as shown, are most often used for buffering or signal-level restoration.

EXCLUSIVE-OR GATE

Although the exclusive-OR (XOR) gate is sometimes treated as a logical element, it is actually a block of several basic elements. The XOR is a special OR gate which excludes the AND function.

Given two inputs, *A* and *B*, the XOR function is defined as "*A* AND NOT *B*, OR *B* AND NOT *A*." This logic is performed straightforwardly, as shown by Fig. 3-11A.

Depending on practical considerations such as the number of XOR's required in a data system, the XOR circuit is sometimes wired as a combination of individual AND, OR, and inverter elements, or may be fabricated specifically as an exclusive-OR gate. As will be shown later, the XOR gate finds varied applications in comparators and arithmetic circuits.



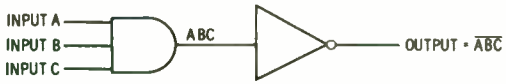
(A) Logic diagram.

(B) Logic symbol.

Fig. 3-11. Exclusive-OR circuit.

NAND AND OR CIRCUITS

Especially in data systems which employ integrated circuit logic, the use of NAND or NOR gates is extremely popular. These circuit elements are essentially equal in logical values to their AND and OR counterparts, but offer certain distinct electrical and logical advantages.



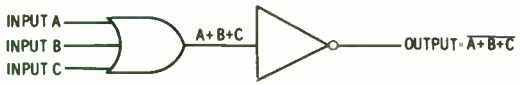
(A) Logic diagram.

(B) Logic symbol.

Fig. 3-12. NAND gate.

The NAND (NOT-AND) gate is simply an AND gate followed by an inverter, as shown in Fig. 3-12A. Fig. 3-12B shows the NAND logic symbol.

Likewise, the NOR (NOT-OR) gate is an assembly of an OR gate followed by an inverter, as shown in Fig. 3-13A. Fig. 3-13B shows the NOR logic symbol.



(A) Logic diagram.

(B) Logic symbol.

Fig. 3-13. NOR gate.

Although the use of NAND or NOR logic complicates the problem of system analysis and requires that the user have a very firm understanding of De Morgan's theorem of inversion of expressions, these elements offer two extremely useful features. The first, and probably most valuable, is signal restoration at each stage. In a conventional diode gate, the logic level is diminished somewhat by the forward voltage drop in the diodes. The effect of this voltage drop usually results in a slightly reduced 1 level from an OR gate, and a slightly elevated 0 level from an AND gate. When several gates must be used in cascade, the individual voltage errors may build up to the point where the difference between a 1 and 0 is not sufficient for reliable operation. In this situation, an inverter or two cascaded inverters must be used to restore the original logic levels. In data systems which use NAND and NOR logic, this problem is never encountered since an inverter is included as a part of each gate.

The second, and less apparent, advantage of NAND/NOR logic is often appreciated only by the system design engineer. An entire digital system may be assembled using only one standard circuit element. NAND gates or NOR gates may be interconnected, using an occasional external component, to form flip-flops, one-shots, Schmitt triggers, and other more complex circuits. From a standpoint of system design economy, this can be a very useful feature.

WIRED-AND AND WIRED-OR GATES

In applications which require gates with a large number of inputs, usually smaller gates are ganged in parallel to satisfy the requirements. Consider a situation which calls for a ten-input AND gate to be made up from three or four input standard elements. The circuit of Fig. 3-14 satisfies this requirement.

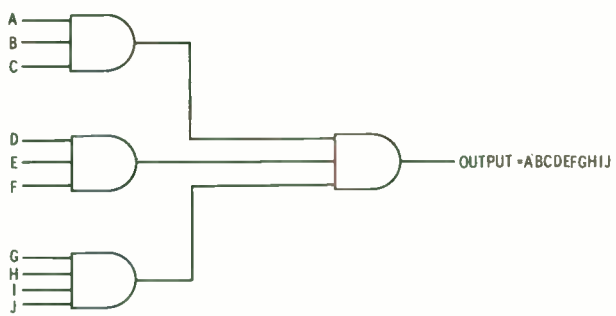


Fig. 3-14. Two-level AND gate.

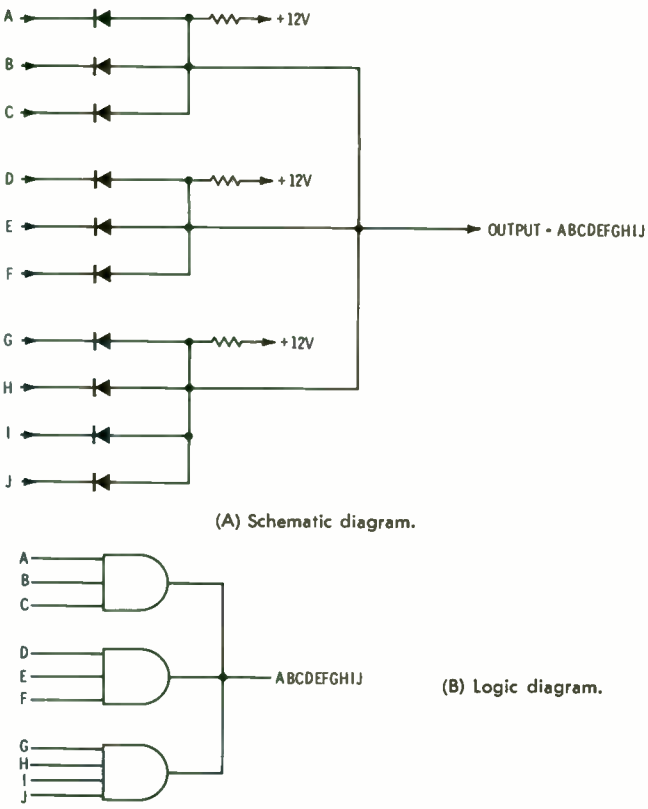
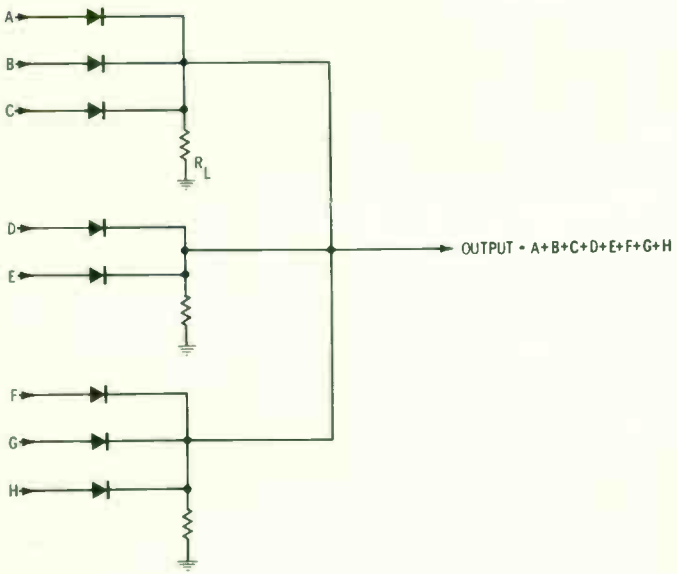


Fig. 3-15. Wired-AND configuration.

This configuration is logically satisfactory, although economically inefficient. If we consider the circuit of the AND gate previously described, (Fig. 3-1), it is apparent that if the outputs of two or more AND gates are directly connected, the output line will represent the AND function of all of the inputs (Fig. 3-15). This characteristic, known as the "wired-AND" configuration is a property of the particular circuit elements being used in the design of a system.

The OR gates described in Fig. 3-4 have a similar property; that is, if two or more OR gate outputs are directly connected, the output line will be the OR function of all the inputs (Fig. 3-16).

It must be kept in mind that the wired-AND and -OR functions are a peculiarity of the electrical design of the circuit elements. Depending on the detail design and manufacturer's specifications of the circuit elements, simply connecting gate outputs together does not necessarily produce wired-AND or wired-OR performance.



(A) Schematic diagram.

(B) Logic diagram.

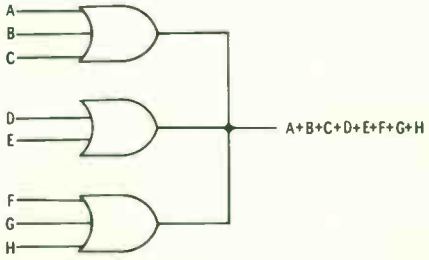


Fig. 3-16. Wired-OR configuration.

FLIP-FLOP

Originally known as the Eccles-Jordan bistable multivibrator, the flip-flop is the most widely used active element in any data system. It finds its most important applications as a one-bit memory element or as one stage of a counter, scaler, or shift register. In the simplest form, the flip-flop consists of two inverters in a positive feedback loop as shown in Fig. 3-17.

Except when acted on by an outside influence, the circuit will assume one of two possible stable states: inverter No. 1 saturated and inverter No. 2 cut off, or vice versa. When inverter No. 1 is saturated, its output is zero, which provides no drive to inverter

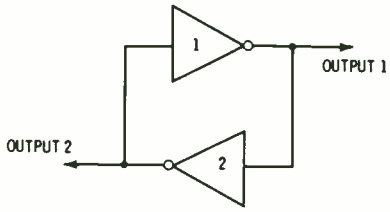


Fig. 3-17. Bistable inverter loop, or latch.

No. 2. The output of inverter No. 2, in the cutoff state, is high, which provides the saturation drive for No. 1.

Since the feedback loop is positive, that is, any change tends to produce more change in the same direction, the circuit cannot remain in an in-between state, with both inverters partially conducting. The transition from one state to the other is extremely fast, determined by the gain and frequency response of the inverters. Switching time for a medium-speed flip-flop is typically a very small fraction of one microsecond.

With the addition of external inputs to the two inverters, the basic flip-flop becomes a more useful logic element. The circuit of Fig. 3-18 now has all of the essential components of a one-bit storage or memory device. For illustration, let us define output No. 1 as the set or "1" output, and output No. 2 as the *reset* or "0" output. By definition, if the 1 output terminal is high (+12 volts), the flip-flop is in the set state. In the set state, the No. 2 output must be zero. When the No. 2 output is high, the flip-flop is in the reset, or cleared, condition.

If a positive voltage, or momentary positive pulse, is applied to the No. 1 input, flip-flop will be forced into the set state and will remain set indefinitely. If the flip-flop had already been in the set state, it would simply remain set. Likewise, if a positive voltage or pulse is received at the No. 2 input, the flip-flop goes to the reset state and remains there. In effect, the circuit remembers and indicates the last input which has been received. In the circuit shown in Fig. 3-18, the flip-flop responds only to a 1 (positive) input because of the action of the input diodes. If the diodes were reversed, the flip-flop would respond only to 0-level signals.

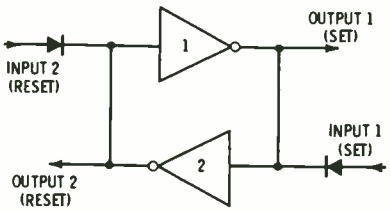


Fig. 3-18. Elementary flip-flop.

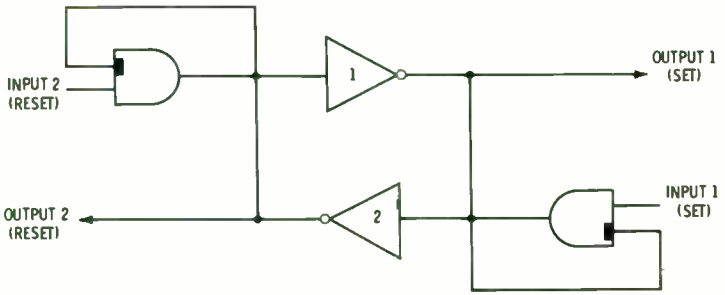


Fig. 3-19. Pulse-gated toggle flip-flop.

Toggling

The simplest flip-flop described thus far will respond predictably to a set or a reset input, but might be confused by two simultaneous inputs. The addition of a pair of input pulse steering gates, as shown in Fig. 3-19, causes the flip-flop to reliably change to the opposite state when acted upon by simultaneous set and reset inputs. This very important function is known as *toggling*.

Assume that the gates are pulse gates which will pass a *negative* input transient when enabled by a dc 1 level. If the set input gate is enabled by the set output of the flip-flop and the reset input gate is enabled by the reset output, a transient applied to both inputs simultaneously will be steered to only one of the inverters. By the nature of the gating connection, if the flip-flop has been in the set state, the simultaneous inputs will cause it to reset, and vice versa. Fig. 3-20 illustrates ideal input and output waveforms for this flip-flop.

Note that each time a negative transient is applied to the set or reset input, the corresponding output goes to the 1 level, and in the case of simultaneous set and reset inputs, as at $t = 14$, the output changes state, regardless of its previous state. Also, because of the negative pulse input gates, positive input transients are ignored, as at $t = 3$.

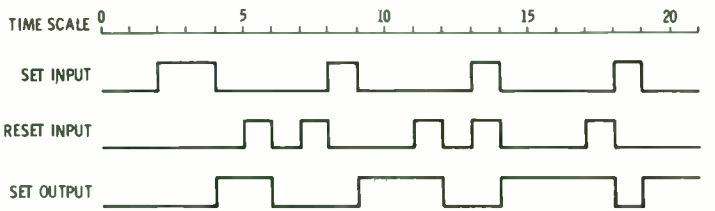


Fig. 3-20. Flip-flop response.

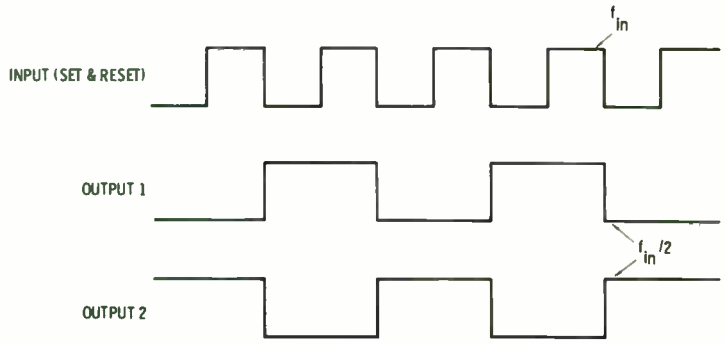
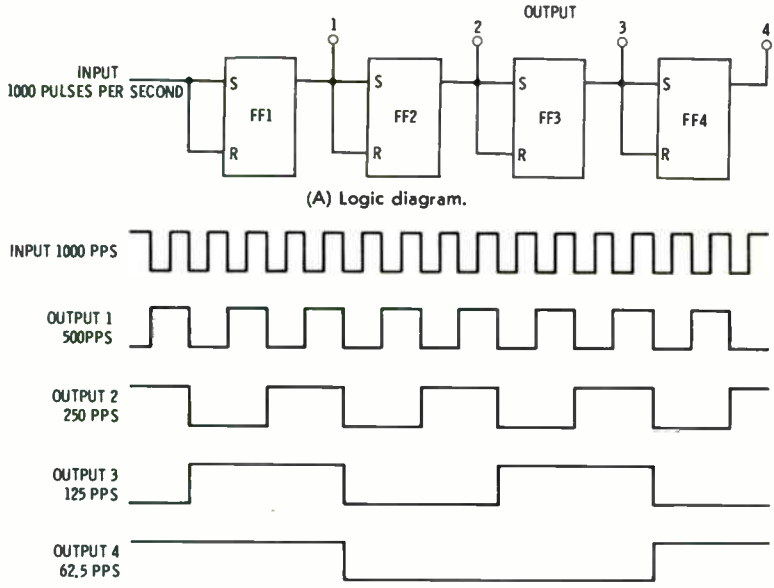


Fig. 3-21. Frequency division by toggling.

Binary Frequency Division

Using the pulse-gated, toggling flip-flop of Fig. 3-19 and applying a square wave to the inputs results in the waveform shown in Fig. 3-21.

The output square-wave frequency is exactly one-half of the input frequency. This binary division of frequency or pulse rate can be cascaded through several stages to produce "binarily" related output frequencies as shown by Fig. 3-22.



(A) Logic diagram.

(B) Waveforms.

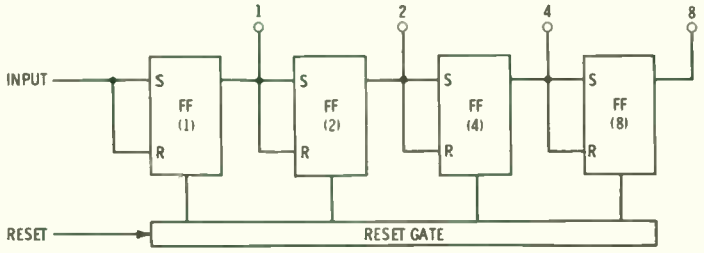
Fig. 3-22. Binary frequency division.

Binary Counting

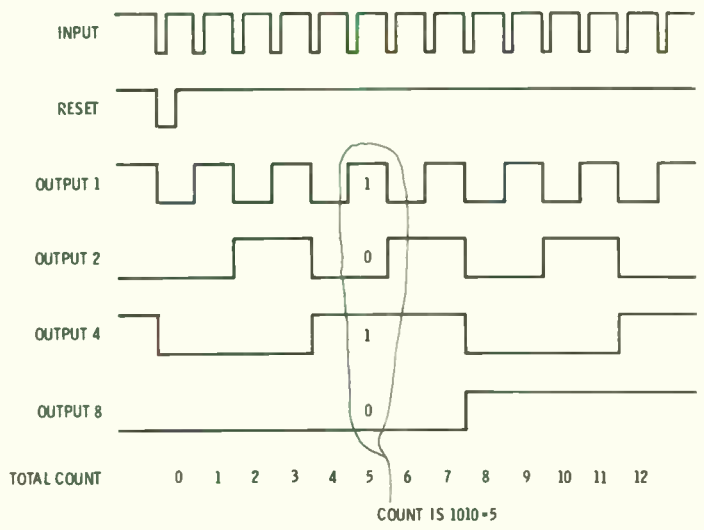
The circuit of Fig. 3-22 becomes a binary counter, or *accumulator* by the inclusion of a common resetting signal (Fig. 3-23A). By the action of the reset gate, all of the toggle flip-flops of the counter are started in the zero, or reset, state. As input pulses are accumulated, the condition, or state, of all of the output lines represents a binary number which is the total of all of the input pulses received since the initial resetting pulse (Fig. 3-23B).

Data Shifting

Another extremely useful application of the flip-flop is in the *shift register*, in which data bits are transferred from one flip-flop to another under control of a "clocking" pulse train. If the set and

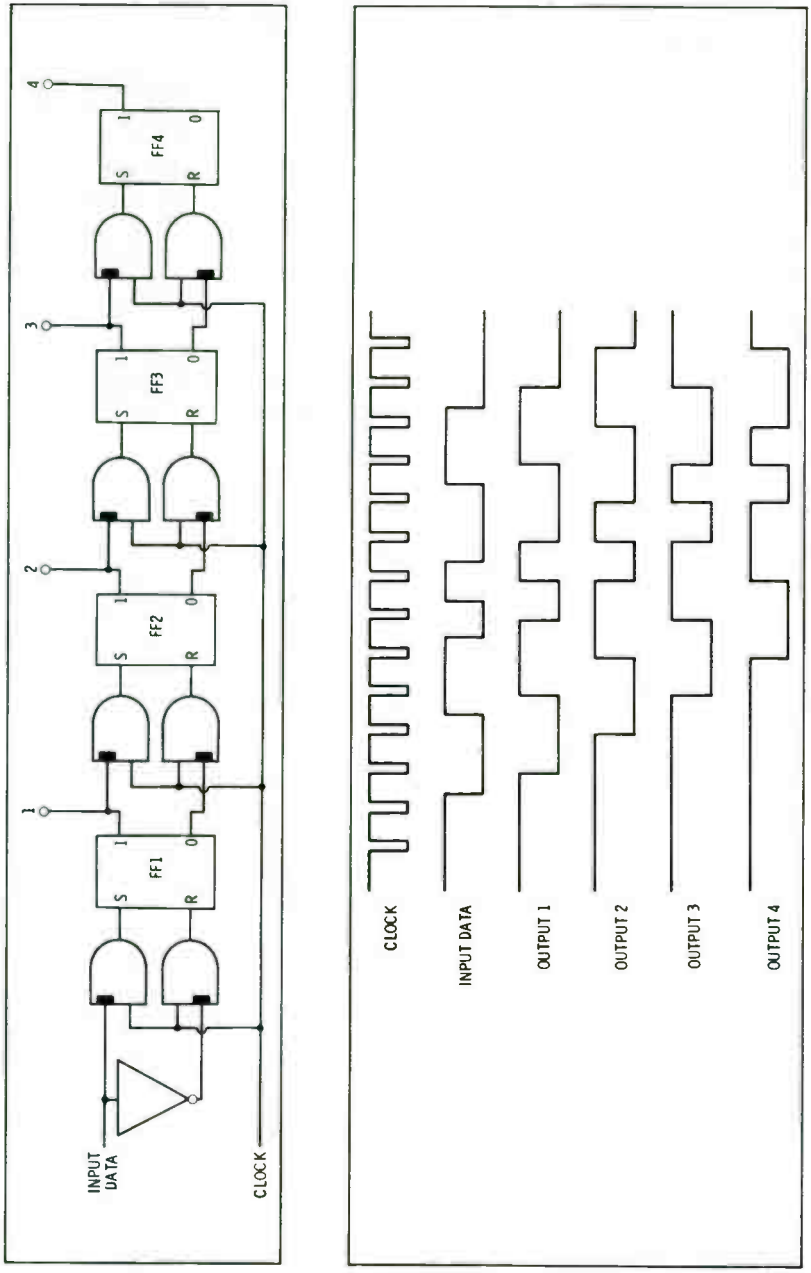


(A) Logic diagram.



(B) Waveforms.

Fig. 3-23. Binary accumulator.



(A) Logic diagram.

(B) Data flow: waveforms.

Fig. 3-24. Four-stage shift register.

reset inputs to a flip-flop are controlled by pulse gates, several flip-flops may be cascaded to form a shift register as shown in Fig. 3-24A.

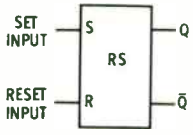
The clock pulse train is applied simultaneously to all of the set and reset input gates, and each gate is controlled by one output of the preceding flip-flop. Since it is possible for only one of the outputs of each flip-flop to be in the 1 state at any time, only one of the input gates on each flip-flop is enabled at any time. The stages are connected so that the set input of each flip-flop is enabled by the 1 output of the preceding stage; likewise for the reset input to each stage. At the instant of the clock-pulse trailing edge (using negative-pulse gates), each flip-flop assumes the previous state of the stage on its left. Fig. 3-24B illustrates the flow of a data pattern through a shift register. Each of the output waveforms is seen to be similar to the input waveform but delayed one clock period per stage.

Standard Flip-Flops

Because of their tremendous utility and range of applications, flip-flops, often numbering in the thousands, are used in all data systems. Depending on the type of system and the nature of the data processing to be performed, various standard types of flip-flops are usually employed. Ordinarily, in the fabrication of a large data system when modular design is used, the standard flip-flop package includes not only the basic bistable circuit but also the associated pulse-steering and logical gating circuits. If a great number of flip-flops are required for a very simple application such as one-bit storage, it is practical to use a standard flip-flop which includes only the essential input gates for that application, or, if a system requires hundreds of stages of shift register flip-flops, it becomes practical to use a standard unit with a more complex gating structure designed to suit the application.

The most universally used standard types of flip-flops have come to be identified by letter designations. The letter designators are generally standardized in industry; however, slight logical differences sometimes exist between units of the same letter type produced by different manufacturers. All of the commonly used flip-flops are described by five classes: types RS, T, RST, D, and J-K.

RS Flip-Flop—The RS (reset-set) flip-flop is the simplest, and may consist only of a pair of NAND gates connected in a loop, or a pair of inverters with a very simple input structure as shown in Fig. 3-18. This flip-flop is sometimes called a *latch*, and is used for the simplest applications, usually as a one-bit storage. The RS flip-flop may be of the asynchronous type, as shown in Fig. 3-25A,



(A) Logic diagram.

R	S	Q_{t+1}
0	0	Q_t
1	0	0
0	1	1
1	1	?

(B) Logic table.

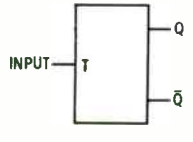
Fig. 3-25. RS flip-flop.

in which the output responds immediately to input signals, or it may be synchronous, in which case the inputs control periodic clock input pulses and the output action occurs slightly delayed. Shown with the RS flip-flop in Fig. 3-26 is the truth table for that element, indicating the output condition, Q, corresponding to all possible input conditions. It can usually be assumed that an RS flip-flop (other than the simplest dc latch) includes pulse-forming gates at the R and S inputs, so that the output responds only to input transients and not to the input dc levels. The most significant feature of the RS flip-flop is that it will respond predictably to a set or reset input, but will not predictably respond to *simultaneous* R and S inputs. For this reason, the RS flip-flop cannot be used as a toggle.

T Flip-Flop—Shown in Fig. 3-26 is the *T*, or *toggling*, flip-flop. This is a fairly simple, specialized configuration used frequently in binary counters. The single input is internally coupled to both sections of the flip-flop through steering gates. T flip-flops often include a dc reset input for convenience in setting up initial starting conditions.

RST Flip-Flop—An RST flip-flop (Fig. 3-27) combines the functions of the RS and the T flip-flops and may be used as a logical RS element or as a counting stage.

D Flip-Flop—The Type D, or delay, flip-flop (Fig. 3-28) is ideally suited for application as one stage of a shift register. It

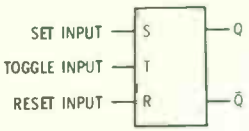


(A) Logic diagram.

T	Q_{t+1}
0	Q_t
1	$\bar{Q}_t$

(B) Logic table.

Fig. 3-26. T flip-flop.



(A) Logic diagram.

R	S	T	Q_{t+1}
0	0	0	Q_t
1	0	0	0
0	1	0	1
0	0	1	$\bar{Q}_t$
1	1	0	?

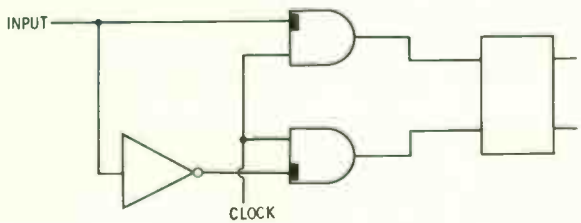
(B) Logic table.

Fig. 3-27. RST flip-flop.

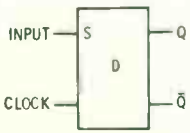
requires only one dc logic input line and a clock pulse. The output state is always equal to the state which existed on the input line just prior to the clock pulse, effectively delaying the input data one clock period. Fig. 3-28 also shows the internal configuration and the truth table for the D flip-flop.

J-K Flip-Flop—The J-K configuration is the most complete of all the standard units. Fig. 3-29 shows the basic J-K unit with one set (J) and one reset (K) input and a clock line. Logically, the J-K flip-flop can replace any of the other standard units. J-K and RST flip-flops are very similar, with the exception that the J-K will predictably change state when both the J and K conditions are satisfied.

All of the standard flip-flops may, and usually do, have additional inputs such as direct reset and expanded input gates. A very popular design is the RST flip-flop with two- or three-term



(B) Equivalent circuit.

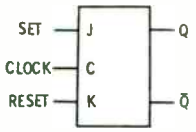


(A) Logic diagram.

S	Q_{t+1}
0	0
1	1

(C) Logic table.

Fig. 3-28. Type D flip-flop.



(A) Logic diagram.

J	K	Q_{t+1}
0	0	Q_t
1	0	1
0	1	0
1	1	$\bar{Q}_t$

(B) Logic table.

Fig. 3-29. J-K flip-flop.

dc AND gates on the R and S inputs. A flip-flop of this type can be converted to J-K operation merely by connecting the set output to one reset input and the reset output to one set input.

TRIGGERING CIRCUITS

There are four popular circuit elements known as trigger circuits (including the basic flip-flop), which are very nearly identical in circuit form, but are designed to perform specialized logical functions. In addition to the flip-flop, these are the one-shot multivibrator, the free-running multivibrator, and the Schmitt trigger.

Triggering circuits, in their most general form, consist of two inverters in a positive-feedback loop. All trigger circuits are of a bistable nature; that is, they are always in one of two stable or semistable dc states, and cannot steadily assume an in-between condition. The "triggered" classification derives from their common characteristic of being tripped irreversibly from one of the states to the other by a momentary input signal. Once the internal process of changing states has been initiated, a trigger circuit will flip all the way to the opposite state regardless of other external conditions.

The four triggering circuits in common use differ primarily in the way that the two component inverters are coupled into the loop. Fig. 3-30 illustrates the fundamental similarity of the four circuits and the small electrical differences.

One-Shot Multivibrator

This circuit, electrically almost identical with the flip-flop, is known as the monostable or one-shot multivibrator (commonly called the *one-shot*), and is used in most digital systems for the generation of adjustable, semiprecise time delays. Logically, its action can be considered as a temporary memory function, in that it retains the effect of an impulse for a fixed period.

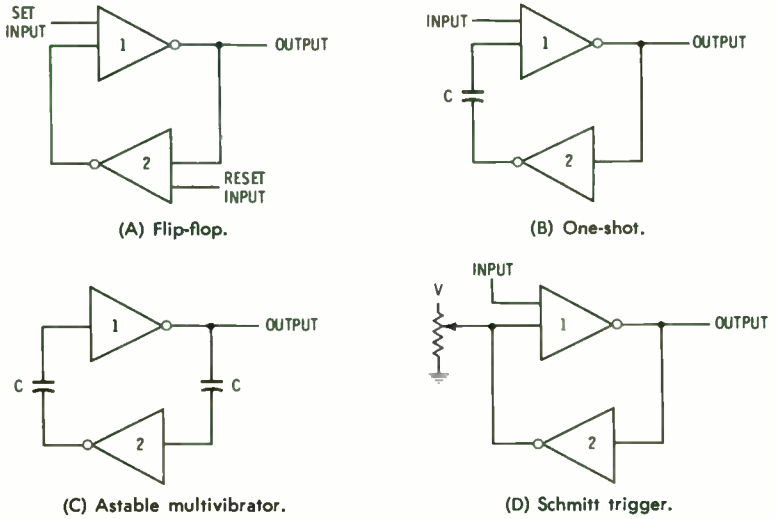


Fig. 3-30. Triggering circuits.

While the flip-flop consists of two inverters connected in a dc positive-feedback loop, the one-shot consists of two dc-coupled inverters with an ac feedback path. Fig. 3-30B shows this configuration. Because of its dc symmetry, the flip-flop is equally stable in either of two states: inverter 1 saturated and inverter 2 cut off, or vice versa. In the case of the one-shot, however, no external dc drive is supplied to inverter 1, therefore inverter 1 will always *tend* to be cut off, which in turn tends to drive inverter 2 to saturation. The one-shot, as illustrated, will always assume this stable state from the time that power is first applied.

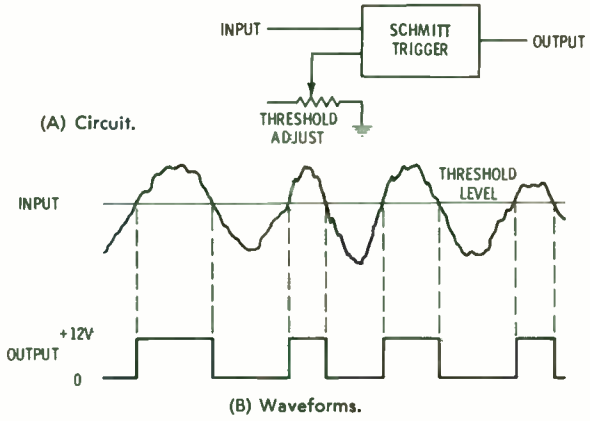


Fig. 3-31. Schmitt trigger operation.

Starting from this stable condition, if a positive pulse is momentarily applied to the input of inverter 1, driving it to saturation, inverter 2 will momentarily be cut off, coupling a signal through C to the input of inverter 1. This signal fed back to inverter 1 is in the same positive direction as the input pulse, and will hold inverter 1 in saturation as long as C retains its charge. During this period, the one-shot is said to be in the quasi-stable, or active, state. Although the circuit cannot remain in this state indefinitely, both outputs will temporarily be solid dc logic levels, the opposite of the quiescent, or stable, levels.

During the active state, while inverter 1 is being driven by the charge on C, that charge is bleeding off in the circuit resistance and into the input of inverter 1. When the input voltage, or current, applied to inverter 1 has diminished to the point where inverter 1 begins to come out of saturation, the output of inverter 1 rises from zero and drives inverter 2 to saturation. The output transient of inverter 2 is coupled back through C, regenerating the turnoff effect, causing the entire circuit to suddenly flip back to its quiescent state.

Since the time duration of the active state is determined almost entirely by the value of C and the circuit resistance with the discharge of C, the delay time is reasonably stable. With good circuit design, delay stability of a few percent is easily achieved. The delay time is usually determined by selection of an appropriate value of C, and may be finely adjusted by means of a variable resistance in the discharge circuit.

One-shots find their main applications in delaying digital events. For instance, if a flip-flop is required to be set approximately one millisecond after receipt of an input pulse, the pulse may be fed to a one-millisecond one-shot, and the delayed output (the end of the active state) transient of the one-shot may be used to trigger the flip-flop. One-shots are also commonly used as "pulse stretchers," in which a narrow pulse is used to initiate the generation of a pulse of any desired width.

The circuit of Fig. 3-30B is intended only as an illustration of the functional aspect of the one-shot. There are several basic variations of the illustrated design, and a huge variety of detail designs in popular use. All, however, embody the illustrated concept of two inverters, connected in a positive-feedback loop which includes both dc and ac coupling.

Free-Running Multivibrator

If both the forward path and the feedback path in an inverter loop are capacitively coupled as shown by Fig. 3-30C, and proper adjustments are made, the loop will oscillate continuously,

producing a square wave or binary-level pulse train. This configuration of the two-inverter loop is called an *astable*, or *free-running, multivibrator.*

Just as the value of the coupling capacitor determines the time duration of the quasi-stable state of a one-shot, each of the coupling capacitors in the free-running multivibrator determines the duration of one-half of a cycle of oscillation. When the circuit is symmetrical, the output waveform is a symmetrical square wave. If an unsymmetrical pulse waveform is required, often the case in digital systems, the on-time/off-time ratio is determined by the ratio of the values of the two coupling capacitors.

When two inverters are coupled exactly as shown in Fig. 3-30C, the resulting multivibrator will oscillate, but it is not self-starting, because a stable condition is possible with both inverters in the nonconducting state. A practical multivibrator circuit always includes a dc biasing arrangement which furnishes a small quiescent base current to each inverter, providing the necessary instability or active gain required for oscillation. As a circuit refinement, a selectable or variable resistance value is usually designed into one of the RC coupling circuits, permitting a fine adjustment of the oscillating frequency. The operating frequency and waveform symmetry are coarsely determined by the value of the two coupling capacitors.

Schmitt Trigger

The Schmitt trigger, shown in a simplified form in Fig. 3-30D, uses a dc-coupled inverter loop which is internally designed very much like a flip-flop. This circuit, however, is not used as a logic element, but for signal conditioning.

All of the internal signals of a digital data processing system must be binary dc levels or binary pulse trains; however, the signals presented to the input terminals of a system are often of a poorly defined shape and level. The Schmitt trigger is used as a threshold detector, producing a "0" output when the signal level is below a precisely defined value and flipping to a "1" output when the level exceeds that value. Most practical Schmitt triggers have an adjustable input bias network so that the exact input threshold level can be adjusted to suit the signal and system requirements (Fig. 3-31A). Fig. 3-31B illustrates the amplitude and time relationships of a typical Schmitt trigger.

OPERATIONAL AMPLIFIER

Seemingly out-of-place in a text on digital data circuits, the operational amplifier is usually used as the fundamental building

block in an analog computer. Before the advent of integrated circuits, operational amplifiers were so costly and complex that they were used only in very critical, high-grade, computing applications. With recent developments in monolithic construction, however, operational amplifiers have been reduced in size and cost to a practical degree, allowing a large scale of general-purpose applications.

The term "operational" stems from its original use only in the performance of a mathematical operation—integration, differentiation, multiplication, and division. In digital circuits the operational amplifier is almost never used as a mathematical operator, but finds wide application in various signal-conditioning circuits.

By the simplest definition, an operational amplifier is a voltage amplifier with extremely high (sometimes assumed to be infinite) gain, wide frequency response, excellent stability, and relatively high input resistance. Operational amplifiers may be ac or dc, depending on the application requirements.

In most typical applications, a circuit which includes an operational amplifier appears to have low gain, or precisely a 1-to-1 ratio of output to input voltage, in spite of the inherently high gain of the operational amplifier. This low gain, or *transfer*, as it is more correctly called, is the result of a negative feedback loop around the operational amplifier. Consider the circuit of Fig. 3-32 in which the operational amplifier output is returned through a feedback resistor, R_f, to the input point.

It must be understood, or assumed, in all operational amplifier circuits, that the sense of the feedback connection is negative; that is, a signal swing at the input terminal will produce an amplified output swing in the opposite direction. If we assume that the amplifier has a voltage gain of a million or more, the signal required at the input terminal to produce full-scale output, say 10 volts, will be very low, a few microvolts at most. Since the output voltage at all times tends to reduce the net input signal because of the feedback connection, under all normal conditions the input signal will be practically zero. For most purposes of analysis, the input signal may be considered to be zero, although we understand that this condition is not actually possible.

Assuming the input signal to be zero in Fig. 3-32, implies that zero signal current flows into the operational amplifier input terminal. In other words, the signal current produced by V_{in}/R_{in} is equal to the current drawn by V_{out}/R_f. Expressed as an equation,

$$\frac{V_{in}}{R_{in}} = \frac{V_{out}}{R_f}$$

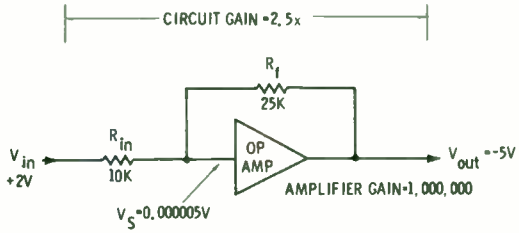


Fig. 3-32. Operational amplifier.

which is the fundamental expression that defines the operating condition of all operational amplifier feedback circuits. Slightly rearranged, this expression gives the value of the output voltage as the input voltage times the ratio of R_f to R_{in}, or,

$$V_{out} = V_{in} \times \frac{R_f}{R_{in}}$$

Using this relation, one can determine at a glance the transfer of an operational amplifier circuit. For example, in the circuit of Fig. 3-32, if R_{in} is 10 kilohms and R_f is 25 kilohms, the transfer, or V_{out}/V_{in}, is 25/10, or 2.5. With a very high gain amplifier, as is assumed, the transfer value of 2.5 is as precise and stable as the ratio of the resistor values, and is essentially unaffected by variations in the exact value of the amplifier gain.

4

LOGICAL BLOCKS

Just as an electronic circuit consists of individual small components, a large data handling system is made up of functional blocks, each block performing a specific logical task. Ordinarily, the logical blocks are composed of a number of elemental logic circuits interconnected in some standard configuration.

LOGIC SYSTEM CONVENTIONS

In describing a data system of any significant complexity, it would be nearly impossible to draw a usable diagram showing all of the individual components. Likewise, it is unnecessarily cumbersome in most instances even to show the individual logic elements when they are used in standard applications. For example, a system might include several binary counters, all identical, or perhaps of different lengths. Knowing the internal details of the flip-flops used in the system, and knowing the standard method of using the flip-flops as binary counter stages, the circuit analyst need only be shown a block symbol with input and output terminals and the label "eight-bit binary counter" to know exactly what is in the block and how it affects the overall system.

In this chapter, some of the most frequently used logic blocks are described and analyzed. Although the blocks can be mechanized by the use of a wide variety of particular circuit element designs, for the sake of clarity the use of a typical family of elements will be assumed. This is essentially the same as the philosophy used in the design of a commercial data processing system, in which the system is first blocked out to satisfy the user's overall

performance requirements, then adapted to whatever specific hardware elements may be in current use or most readily available.

STANDARD CIRCUIT ELEMENTS

The family of typical logic elements to be used for circuit descriptions consists of a flip-flop, one-shot, inverter, and various AND, OR, and special-purpose gates, designed for mutual compatibility. Their maximum operating speed of 1 MHz is adequate for most general-purpose applications. The operating levels are +12 and 0 volts dc, positive-true logic.

Flip-Flop

The flip-flop shown schematically and logically in Fig. 4-1 can be described as a J-K type with three-input set and clear (reset) gates, and direct set and clear access. Trailing-edge triggering is used.

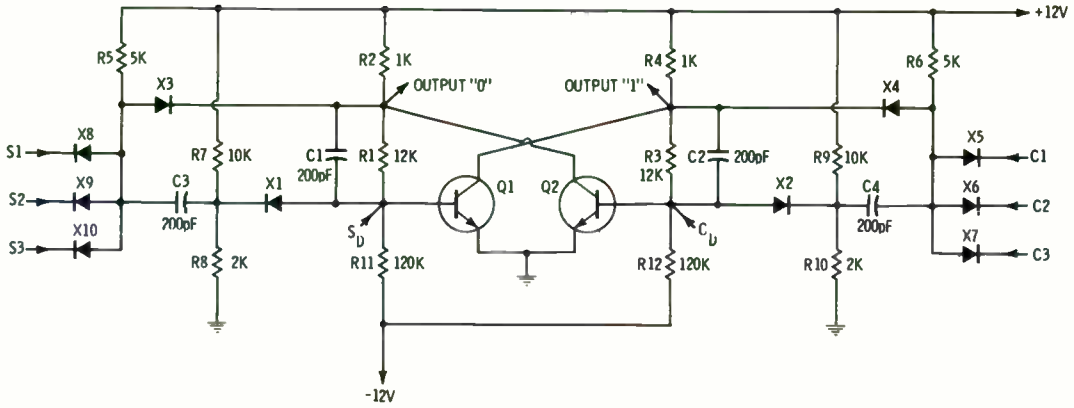
The heart of the circuit is the familiar bistable loop consisting of Q1, Q2, R1, R2, R3, and R4. When Q1 is saturated, its collector voltage, which is essentially zero, holds Q2 cut off. With Q2 cut off, the base of Q1 is held sufficiently positive to maintain saturation of Q1. The circuit is symmetrical so that if Q2 is conducting, Q1 is held in the cutoff condition.

When the circuit is in a given state, it will remain in that state until an externally supplied pulse momentarily cuts off the conducting transistor, at which time the circuit flips rapidly to the opposite state. Capacitors C1 and C2 speed up the transfer between states.

The negative pulses which cause the bistable circuit to trigger may be brought in through the direct set and clear inputs, or they may be generated by the set and clear gating circuits.

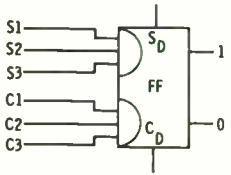
On the set input side, X8, X9, and X10 with R5 make up an AND gate. The voltage at the output of the gate is positive only when all inputs are positive (1 or true). When the gate output switches from +12 volts to zero, this negative transient is coupled through C3 and X1 to the base of Q1, cutting it off and starting the trigger action. The charging time constant of the trigger capacitors, C3 and C4, is such that the gate is required to dwell in the logical 1 condition for at least 0.5 microsecond to generate a usable trailing edge pulse. The clear gate is identical to the set gate in all respects, working into the base of Q2.

Diodes X3 and X4, which may be seen to be a fourth input to the gates, are used as steering diodes. If identical signals are applied simultaneously to the set and clear gates, a negative pulse is



(A) Circuit.

Fig. 4-1. Standard J-K flip-flop.



(B) Logic diagram.

S_D	C_D	Q_{t+1}
0	0	Q_t
1	0	1
0	1	0
1	1	?

(C) Asynchronous (direct) logic table.

S	C	Q_{t+1}
0	0	Q_t
1	0	1
0	1	0
1	1	$\bar{Q}_t$

(D) Synchronous logic table.

delivered only to the saturated transistor, because of the logical action of the diodes. This ensures that the flip-flop has the "J-K" or toggling property. The flip-flop may be used reliably as a toggle, or binary frequency divider, simply by connecting an input signal to one set input and one clear input.

A flip-flop of the type shown by Fig. 4-1 exhibits excellent rejection of input noise as a result of two details in the design. The principle of trailing-edge triggering, or of causing the action to occur after the input gate has been satisfied for at least a minimum specified time (0.5 microsecond in this example), makes the flip-flop insensitive to extremely short, accidental input signals or narrow spikes of noise. Also, the back bias which is applied to the pulse diodes, X1 and X2, by the R7/R8 and R9/R10 voltage dividers causes the diodes to reject pulses which are not of sufficient amplitude to overcome the bias.

Summarizing the input/output conditions of the standard flip-flop, the flip-flop will go to the set condition and remain set when any of the following conditions are met:

1. The direct set input is momentarily held at 0 or at a small negative level.
2. All set logic inputs have been 1 for at least 0.5 microsecond, and one or all go to 0. If only one logic input is connected and the others are open, the open inputs are effectively 1, and the flip-flop will set if the one active input switches from logical 1 to 0.

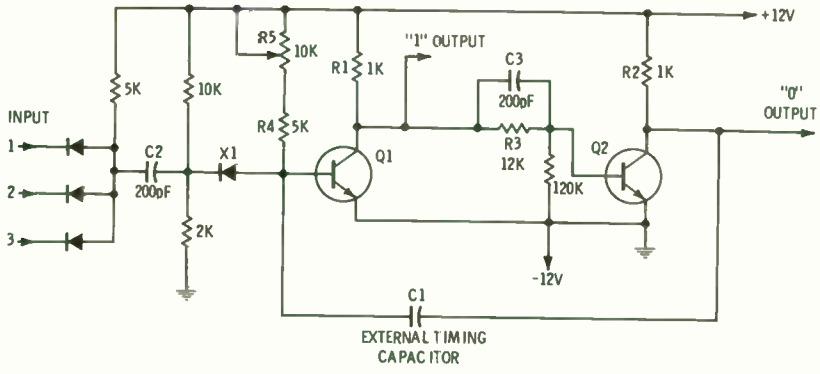
The flip-flop will go to the cleared condition and remain, if the corresponding conditions are met at the clear inputs.

If both *direct* set and clear inputs are held at 0 simultaneously, the state of the flip-flop when the signals are removed is indeterminate; however, if the *logical* set and clear inputs are both satisfied simultaneously, the flip-flop will reliably *change* states.

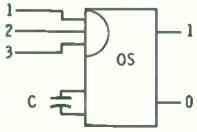
One-Shot Multivibrator

As shown by Fig. 4-2, the one-shot to be used as a standard example consists of a straightforward monostable loop comprised primarily of Q1, Q2, and the feedback path, which includes C1.

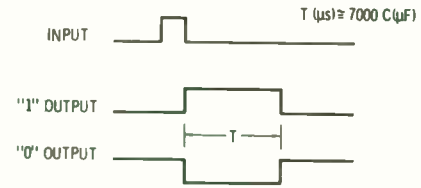
Because of the bias current into Q1 provided by R4 and variable resistor R5, Q1 is held in saturation in the quiescent, or steady, state. Q1, in saturation, prevents Q2 conduction. When Q1 is momentarily cut off by an external influence, Q2 is immediately driven into saturation and its collector voltage drops from +12 volts to zero. This large negative-going transient is coupled through C1 to the base of Q1, maintaining Q1 in the cutoff state. As long as C1 holds Q1 base at a negative or zero voltage, the



(A) Circuit.



(B) Logic diagram.



(C) Waveforms.

Fig. 4-2. Standard one-shot multivibrator.

circuit remains in the quasi-stable state. When C1 has discharged sufficiently through R4 and R5 for Q1 base to be slightly positive, Q1 begins to conduct, its collector voltage goes toward zero, and Q2 conduction stops. In a small fraction of a microsecond, the loop flips back to the original state, with transistor Q1 saturated and transistor Q2 cut off.

The triggering pulse which initially cut off Q1 is generated by a logical input circuit which is identical to the set and reset gates in the standard flip-flop. At the trailing edge of a "true" or logical 1 condition out of the three-input gate, a negative-going transient is coupled through C2 and X1 to the base of normally conducting Q1.

Capacitor C1 and R4 and R5 determine the time duration of the quasi-stable, or active, state of the one-shot. The active time is roughly determined by the value selected for C1, and may be adjusted by the setting of the variable resistor, R5.

The input circuits of the standard one-shot include the same safeguards against input noise as those of the flip-flop, previously described.

DC Gates

Fig. 4-3 shows the very simple configuration of the standard AND and OR gates. Five-input gates are shown as a typical example; however, both types of gates may have as few as two inputs and may be practically expanded to as many as twenty or more inputs without changing the operating characteristics.

Inverter

As is often the case in families of logic elements, the inverter is electrically identical to one-half of the flip-flop and one-shot, providing exactly the same driving characteristics as those units. Fig. 4-3 illustrates the simplicity of the standard inverter.

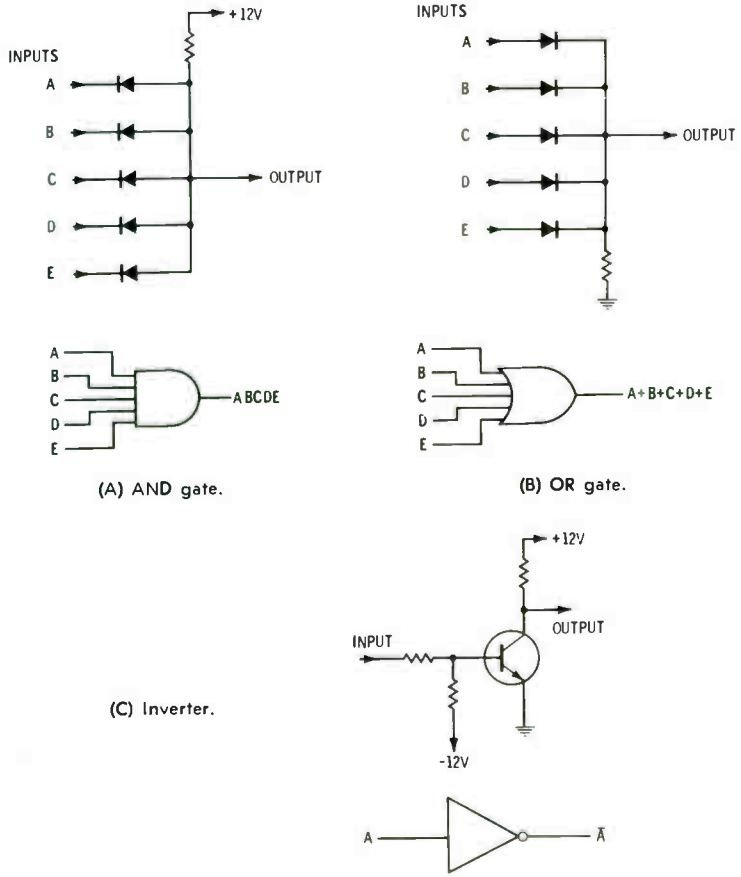


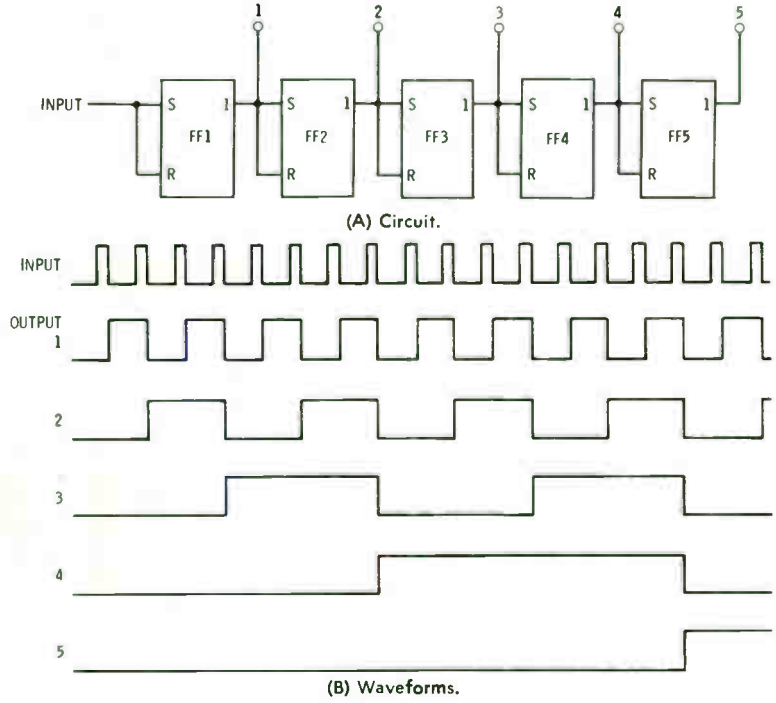
Fig. 4-3. Standard logic elements.

In this chapter, many of the functional logic blocks used frequently in data systems and computers will be described and analyzed. All of the blocks may be assumed to be designed around the standard circuit components.

COUNTERS AND SCALERS

The terms *counter* and *scaler* are often used synonymously, in spite of a fundamental difference. Although counters and scalers appear to be nearly identical, the difference is in the application. A scaler is a connection of logical elements, usually flip-flops, used to scale down or divide an input frequency by an integral factor. A counter is a similar logical block which is used to accumulate the total of a train of input pulses. Most counters can be used as frequency scalers, but not vice versa.

There are as many possible configurations of counters and scalers as there are numeric codes. As is true of the numeric codes, the simplest and most widely used is the natural binary counter. Fig. 4-4 describes a five-stage binary scaler, consisting simply of five toggling flip-flops in cascade.



(A) Circuit.

(B) Waveforms.

Fig. 4-4. Five-stage binary scaler.

From the waveforms, and with an understanding of the flip-flop toggling action, it is seen that each flip-flop divides its own input frequency by 2. A five-stage binary scaler, as shown, divides the input frequency by 2^5, or 32. Likewise, a six-stage scaler divides by 2^6, or 64, and a seven-stage scaler divides by 128, and so on.

An interesting and often very useful feature of the binary scaler is the precise phase relationship of all of the intermediate outputs. If, in some application, a family of binary related pulse rates such as 10,000; 5000; 2500; 1250; and 625 pulses per second, is required, the scaler produces outputs, one from each stage, at the required frequencies, and all outputs are precisely aligned in time with one another.

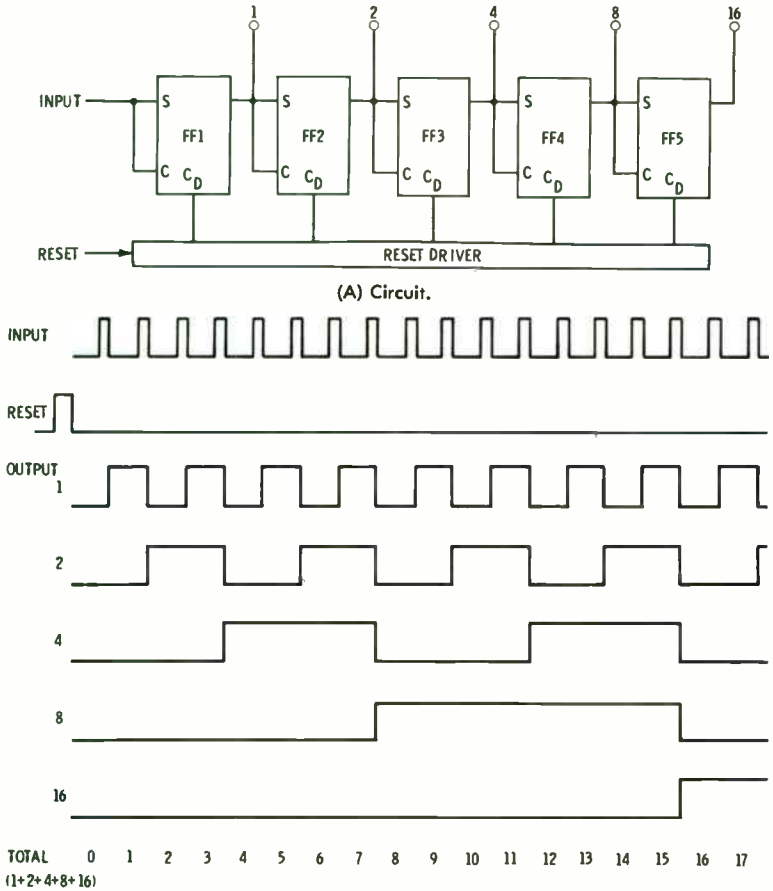
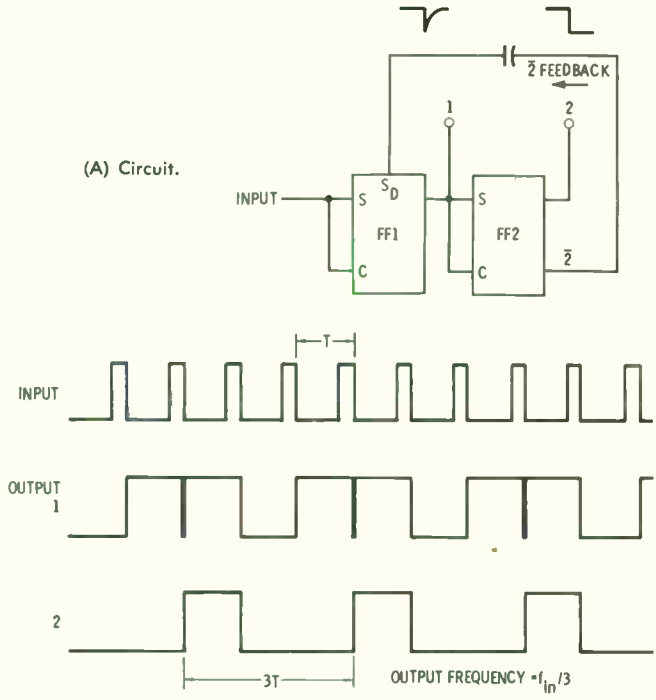


Fig. 4-5. Binary counter, or accumulator.

When a common reset signal is added to the scaler circuit as shown by Fig. 4-5, the circuit may be used as a binary accumulator, or counter. In this application, all of the intermediate stages drive output lines. At any instant, the five output lines indicate, in parallel binary code, the total number of input pulses received since the reset signal. It must be kept in mind that the useful capacity of the counter is limited by the number of stages. A five-stage counter is capable of registering 32 states, reading out from 00000 to 11111, or 0 to 31. If the counter is allowed to accumulate to its "full-house" count of 31, the next input pulse will cause the output to recycle to 00000, and the counter again will accumulate toward 11111.

Feedback Counter

Division ratios other than the pure binary numbers can be obtained from flip-flop counters by the application of feedback signals. Consider the circuit of Fig. 4-6, in which a feedback pulse is added from the 2 output of an $N/4$ scaler to the direct set input of the first stage. Assume, as shown, that both stages are initially

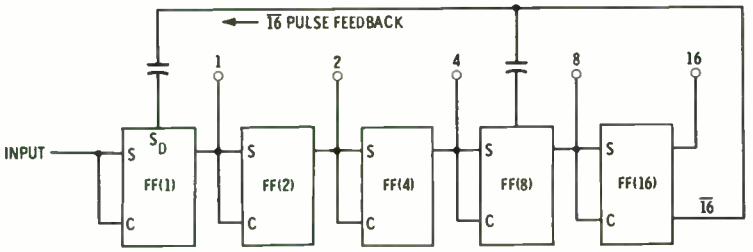


(B) Waveforms.

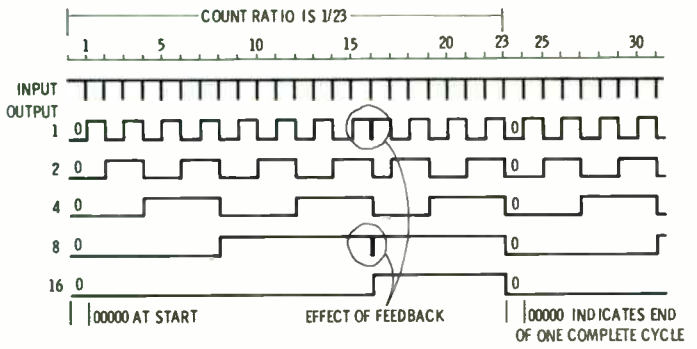
Fig. 4-6. *N*/3 feedback scaler.

cleared. The first pulse sets the 1 flip-flop. The second pulse clears the 1 flip-flop and sets the 2 flip-flop. As the 2 flip-flop is set, the 2 feedback line goes from +12 volts to zero, coupling a negative pulse into the direct set input of flip-flop 1, causing flip-flop 1 to set. The third pulse clears flip-flop 1, which in turn clears flip-flop 2. With both flip-flops again cleared, we have returned to the initial conditions, and the overall cycle is repeated. It may be seen that the $N/4$ scaler has been changed to an $N/3$ scaler by the action of the feedback line.

Pulse feedback, as described, may be used to convert a binary scaler of any length to a scaler of effectively lower ratio. A generalization may be made regarding the numerical effect of pulse feedback when applied from the *last* stage of a binary counter to the set input of a prior stage, or stages: *The resulting count ratio of a feedback scaler is equal to the natural binary ratio minus the sum of the weights of the stages receiving the feedback pulse.*



(A) Circuit.



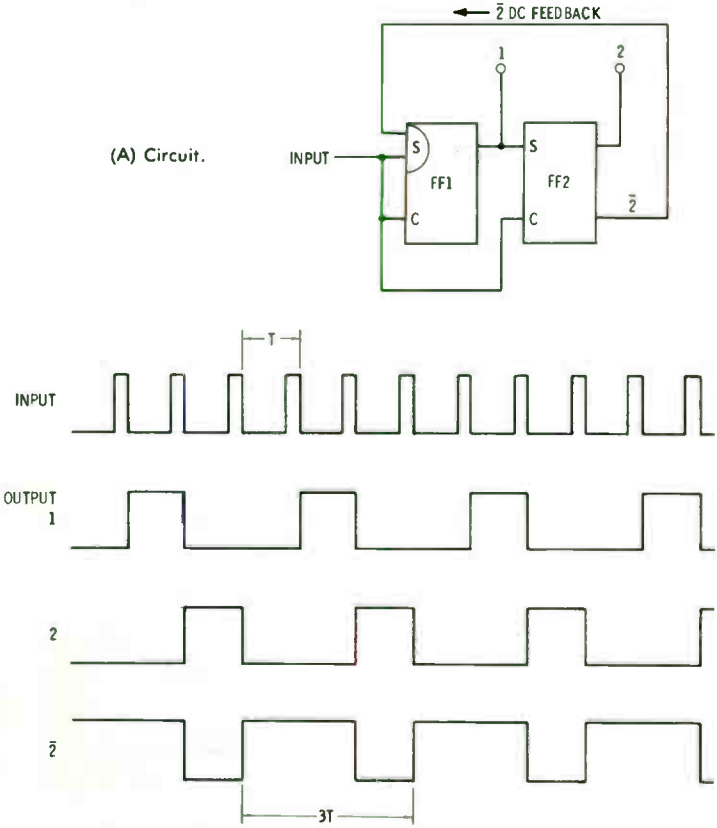
(B) Waveforms.

Fig. 4-7. *N*/23 feedback scaler.

Fig. 4-7 illustrates this rule. The five-stage scaler has a "natural" ratio of 1/32. With feedback applied to the 1 and 8 set inputs, the count ratio becomes 32 minus (1 + 8), or 23.

Although pulse feedback is very simple and straightforward in principle, it is not the most reliable method of modifying a binary count ratio. To guarantee that the feedback signal will be effective, the feedback pulse width must be carefully matched to the internal characteristics and delays in the flip-flops. Using logically gated flip-flops such as our standard unit, very reliable feedback scaling may be accomplished by the use of dc logical manipulation which eliminates the critical timing considerations of pulse feedback.

Fig. 4-8 shows the connection of an $N/3$ scaler using dc feedback gating to effect the conversion from the natural $N/4$ ratio. With

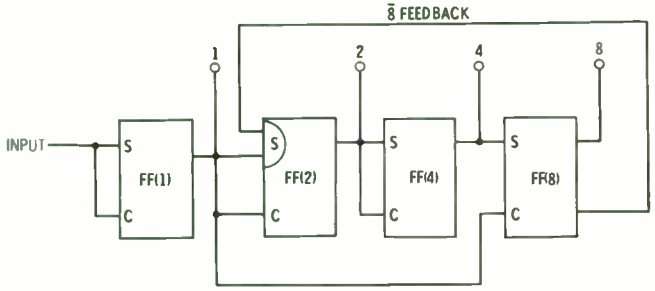


(B) Waveforms.

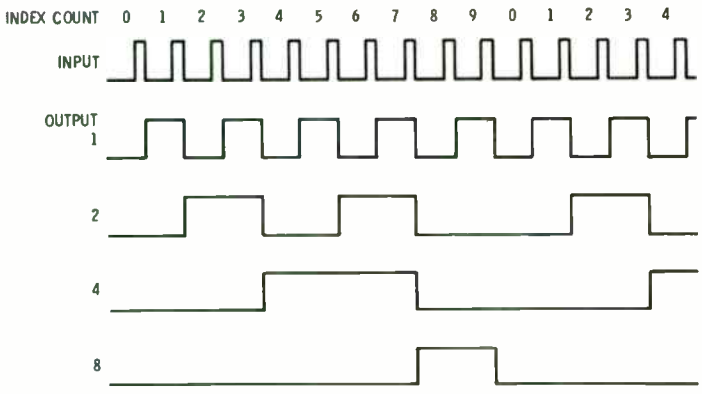
Fig. 4-8. *N*/3 scaler: dc feedback method.

both flip-flops initially cleared, the first input pulse sets FF1. The input is also fed to FF2 clear gate, but has no effect since FF2 is already cleared. The second pulse clears FF1. At the trailing edge of the second pulse, FF2 receives both a set and a clear signal—the set from FF1 and the clear from the input line. Receiving two simultaneous inputs, FF2 changes state, going to the set condition. The third input pulse is blocked at the set input of FF1 by the action of the 2 gating line on one of the FF1 set lines. FF1 remains clear and FF2 is cleared by the input signal. At this time, after receiving three input pulses, the scaler is back to its initial 00 condition, indicating that an overall $N/3$ cycle has been completed.

Various combinations of forward and feedback paths can be combined to make a scaler of any ratio, but are always *less* than the natural binary ratio of the unmodified counter. A very familiar and commonly used modified binary scaler is shown in Fig. 4-9. Here, an $N/2$ and $N/5$ scaler are cascaded to operate as an $N/10$ BCD scaler, or counter.



(A) Circuit.



(B) Waveforms.

Fig. 4-9. BCD decade feedback counter.

Up-and-Down Counter

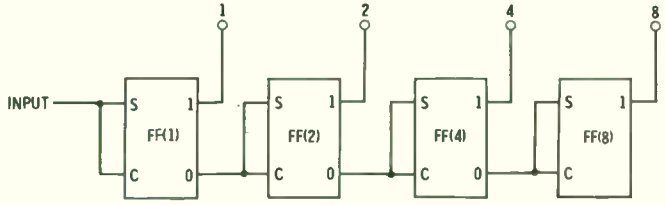
As a binary counter accumulates upward starting from a reset condition, the set outputs of all the flip-flops describe the binary total of the input pulses received. Likewise, the reset outputs of the same counter describe a binary word; this word, however, starts at 11111 and counts downward to zero.

A binary counter may be used for downward counting simply by coupling from each reset output to the next flip-flop as shown by Fig. 4-10.

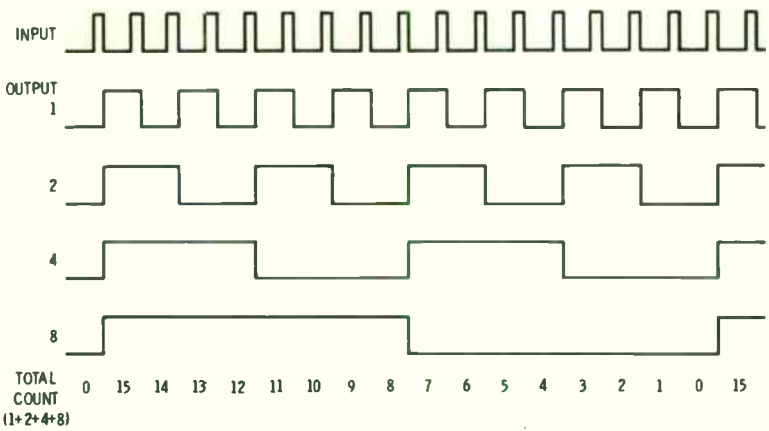
If a counter is required to count upward or downward, externally controlled, the use of two pulse gates at each stage effectively selects the set or reset output of each flip-flop to drive the next stage. In Fig. 4-11, the operation of this connection is shown by the waveform timing.

DECODER

An array of AND gates and inverters connected to recognize one particular data word, or sense a fixed combination of ones and



(A) Circuit.



(B) Waveforms.

Fig. 4-10. Downward binary counter.

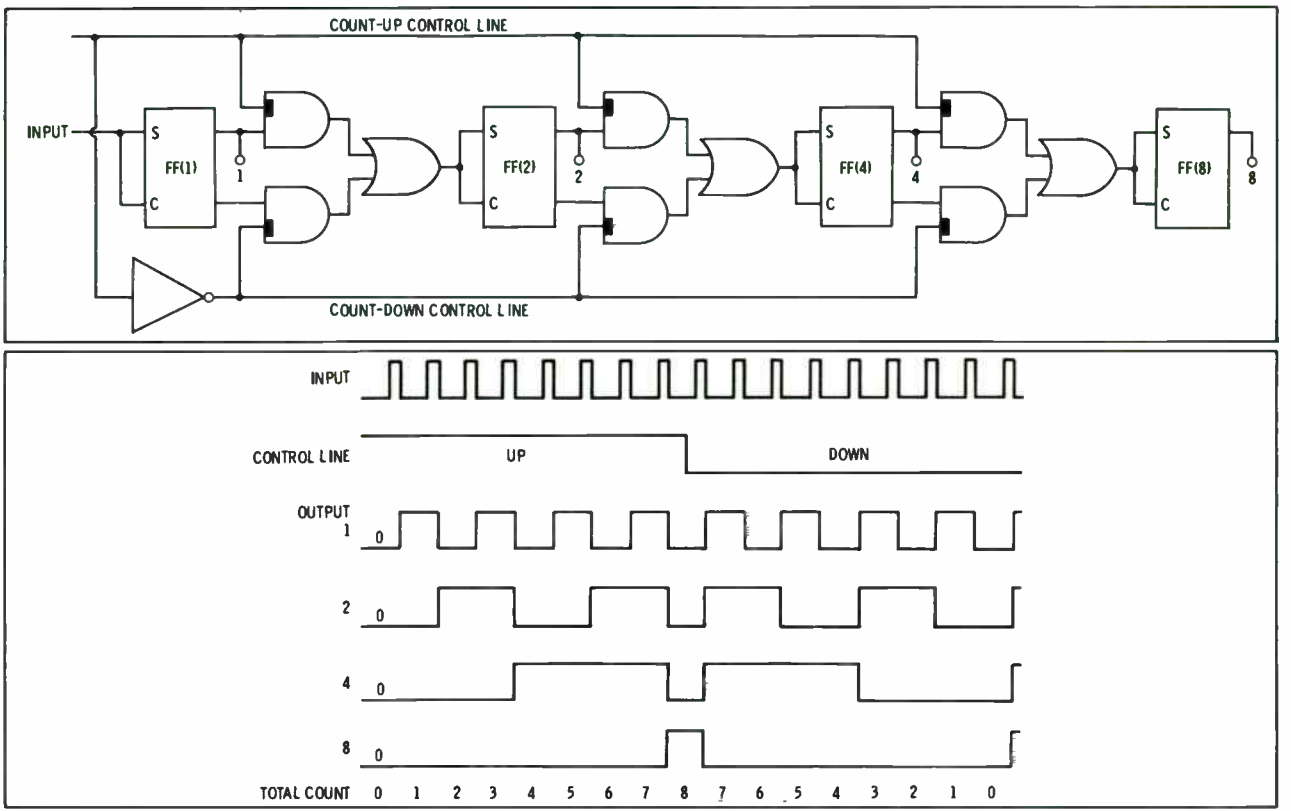
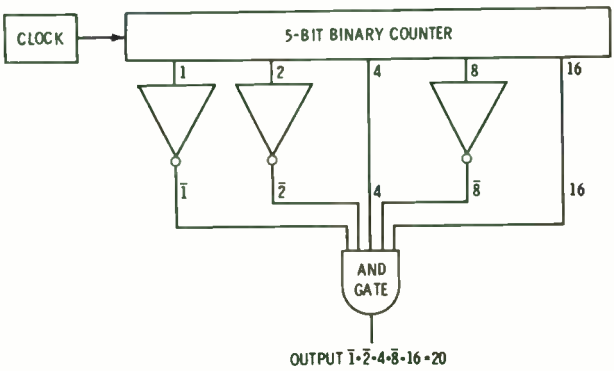


Fig. 4-11. Up/down controlled counter.

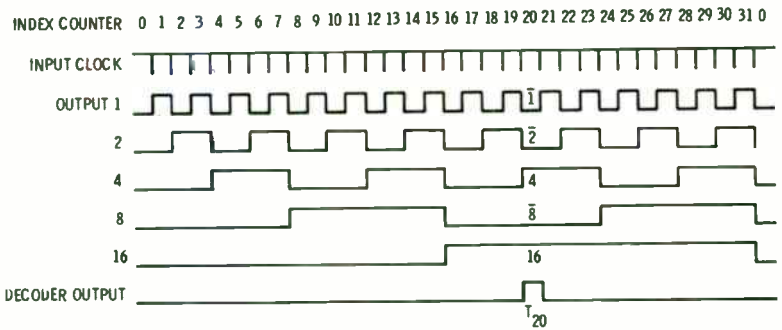
zeros is called a *decoder*. Commonly used for recognition of sync patterns, selecting time slots, or data identification addresses, a decoder may be one multiple-input gate or a connection of several smaller gates.

Consider a typical application as shown by Fig. 4-12. In this example, control pulses, or gating signals, are to be decoded from the output of a clock-counter. The five-bit binary counter runs continuously from 0 to 31, automatically recycling on receipt of every 32nd pulse. Let us assume that some external function is to be activated in the 20th time slot, that is, between the 20th and 21st input pulse. The decoder is connected to produce a logical 1 output when the 16 and 4 lines are 1 and the 1, 2, and 8 lines are 0. If several different time slots are needed, naturally a separate decoder gate is required for each slot.

Frequently, in a counter-decoder connection, as described in Fig. 4-12, a decoder output is used as a master resetting or clearing signal to all of the stages of the counter. This is, in effect, another



(A) Circuit.



(B) Waveforms.

Fig. 4-12. Decoder.

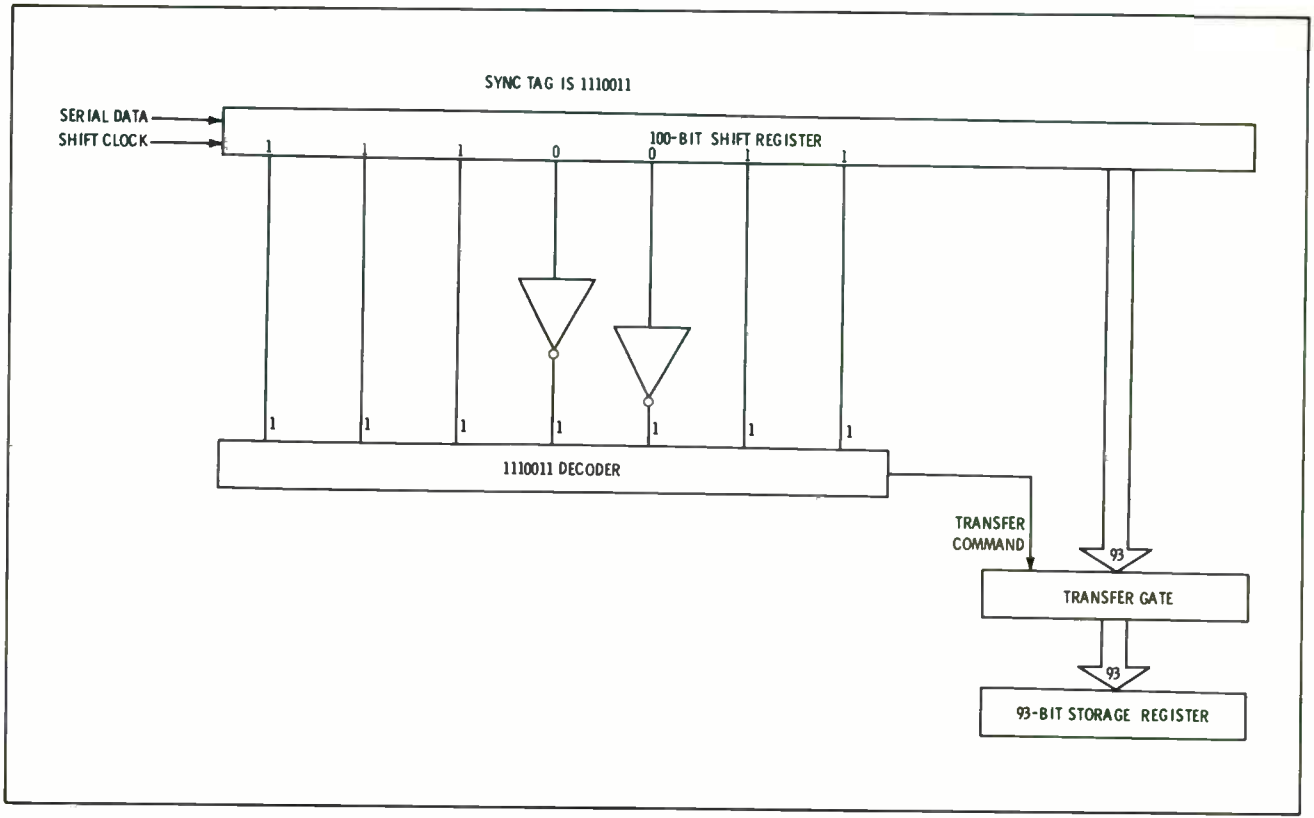


Fig. 4-13. Decoder as sync detector.

configuration of the feedback counter. If the decoded 20th time slot in Fig. 4-12 were fed back to all of the flip-flop direct clear inputs through a suitable driver, the counter would divide the clock frequency by 20.

Another common application of the decoder is recognition of a sync pattern or a unique code word in a serial data stream. For a synchronizing system to be effective, the sync code or pattern must be a combination of 1's and 0's which will never, or very seldom, occur in the data message except as the sync signal. Assume that the word 1110011 is used as the sync code in a data system. As shown in Fig. 4-13, a data stream is shifted through a shift register which is monitored by a seven-bit decoder. Only at the time that the sync word is in place in the register, the decoder delivers a 1 level output. This output could be used to control the transfer of the shifting data out of the register as a parallel dump.

SHIFT REGISTERS

An assembly of clocked flip-flops connected in series for direct transfer of serial data from each flip-flop to the next following flip-flop is called a *shift register*. Fig. 4-14A illustrates a five-bit shift register.

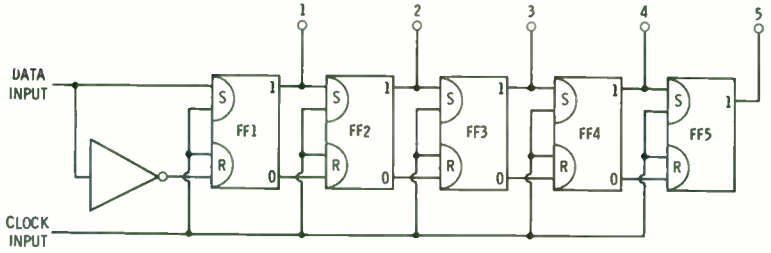
Serial data is entered continuously into the first flip-flop. Each clock pulse updates the condition of the first flip-flop to correspond to the condition of the data line at the instant of the clock-pulse trailing edge. The second flip-flop uses the output of the first flip-flop as its data input. At the trailing edge of each clock pulse, the second flip-flop assumes the logical state which existed in the first flip-flop prior to the clock pulse, and so on down the line.

The input data pattern appears to flow through the shift register, with identical, but time-staggered waveforms at each successive stage. Fig. 4-14B shows the relationships between the waveforms at each of the stages of the register.

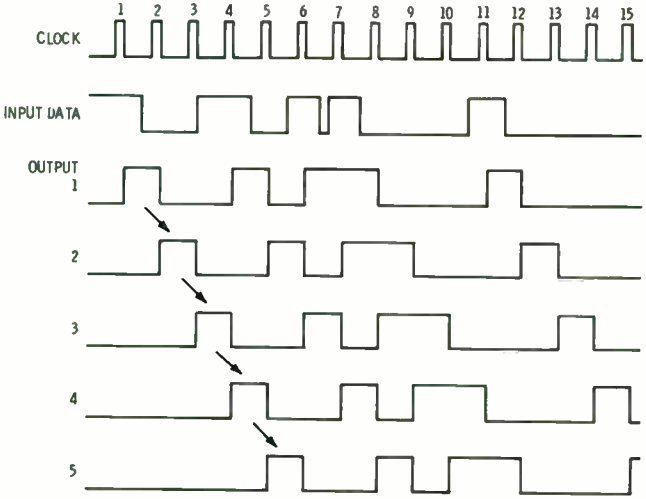
Examination of Fig. 4-14 indicates several useful properties of the shift register. The first stage of the register effectively samples and synchronizes the input data to the clock. Note that, although the input data changes state at random times, the data at the output of the first (and successive) stage changes only at clock time.

Also, by virtue of the clock action, the input data is effective only during the very short clock sampling interval. Although the input data switches from 1 to 0 and back to 1 between clock pulses number 6 and number 7, this data excursion is not recognized, since it did not affect the data value at the sampling time.

Data may be transferred into a shift register in parallel form as shown in Fig. 4-15. In the design of a parallel-entry shift register,



(A) Circuit.



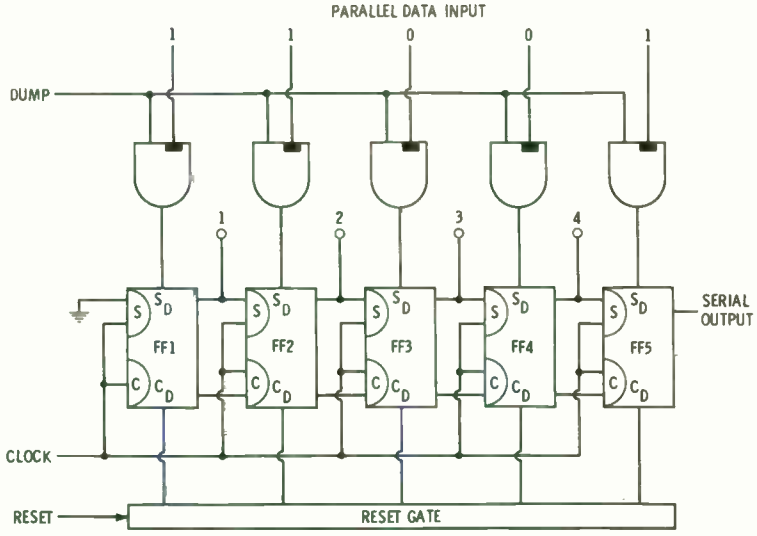
(B) Waveforms.

Fig. 4-14. Shift register data flow.

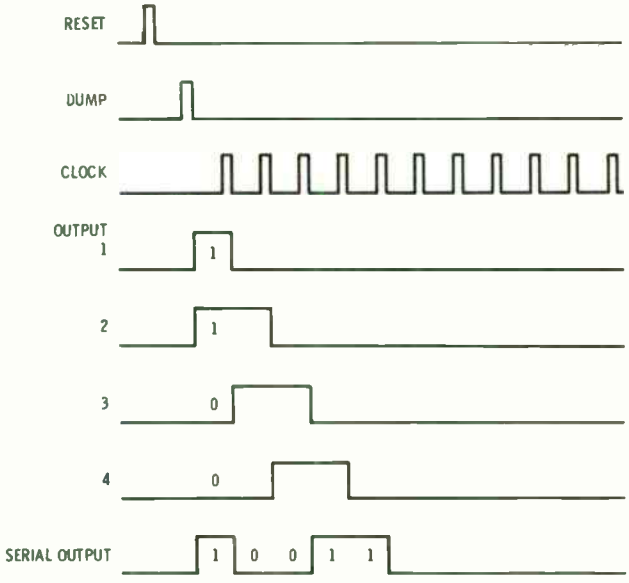
the input "dump" or transfer pulse must be timed so as not to occur simultaneously with the shift clock.

In the circuit of Fig. 4-15A, the register is initially cleared to a condition of all 0's by a reset pulse which may occur at any time prior to the input dump pulse. The parallel data dump causes each flip-flop to assume the condition of its input data line at the instant of the dump pulse. In this instance, as in the previous example, the logical condition of the data lines is meaningful only at the sampling instant. Assuming the input data word in Fig. 4-15 is 11001, this pattern is initially dumped into the stages of the register, then carried serially to the right, as shown by the waveforms.

In the waveforms, the flip-flops are shown to return to a 0 state after the data has shifted through. This, naturally, would depend on the connection at the front (left) end of the register. If the first



(A) Circuit.



(B) Waveforms.

Fig. 4-15. Parallel-to-serial conversion.

flip-flop input is tied to a 0 level, the register will fill with zeros after the data passes through, or fill with ones if the front is tied to a 1 level. As a practical matter, in many applications it is not necessary to clear the register before dumping in new data. If the front end is connected to a 0 logic level and the time between data dumps is longer than the register length (number of stages × clock period), the register will clear itself between data dumps.

For use as a shift register element, a flip-flop requires set and clear (reset) input gates to steer the clock pulses. When used with serial input data, as in Fig. 4-14, only a single set of gates per flip-flop is required. These may be dc or pulse gates, internal or external. Usually, in practical logic systems, the input gates are built in as a part of the flip-flop circuit as in our standard flip-flop. For the parallel-input shift register, an additional set of gates, one per flip-flop, is required. Ordinarily, this gating function is accomplished by the use of external gates working into the direct set of inputs of the flip-flops.

Although individual gates for each parallel data line satisfy the input requirement, the fact that all gates share a common pulse input leads to the use of a special gate assembly known as a *transfer gate*, sometimes called a *strobe gate* or *dump gate*. The transfer gate is an assembly of several independent AND gates, each controlled by one dc data line and a common pulse line (Fig. 4-16).

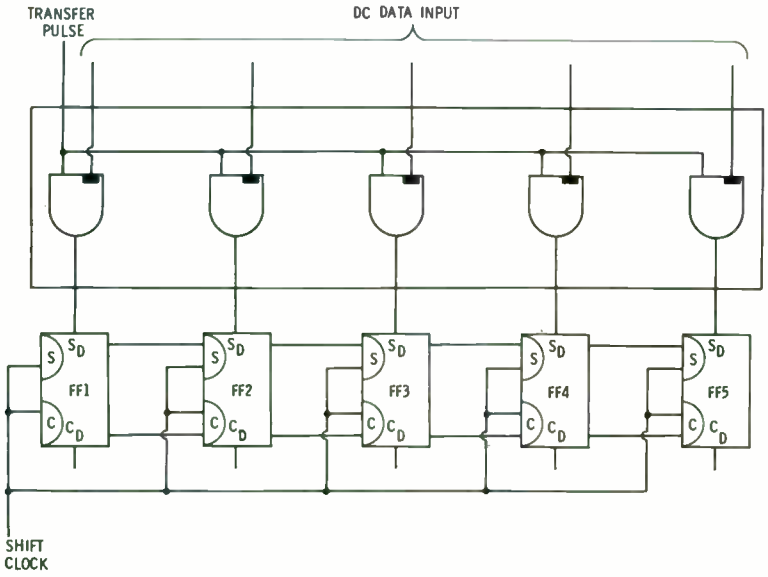


Fig. 4-16. Shift register with transfer gate.

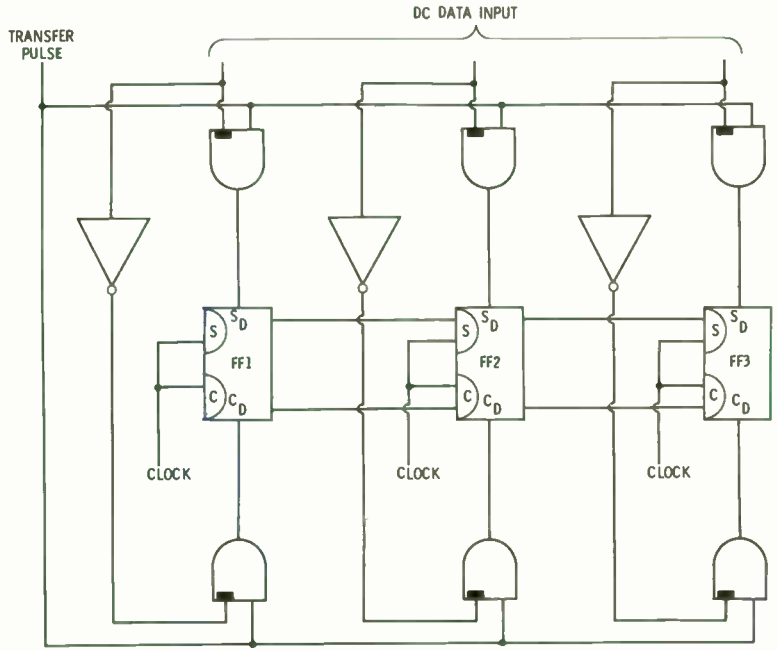


Fig. 4-17. Shift register with full input transfer.

There are situations in which it is impractical to clear the register before dumping in new data. A commonly used method of clearing and parallel loading simultaneously makes use of two transfer gates, one working into the set inputs and the other into the clear inputs (Fig. 4-17). The input data lines control the set gates, and the same data, logically inverted, controls the clear lines. At the instant of data transfer to the shift register, each flip-flop receives *either* a set *or* a clear input command, and the register assumes the pattern of the data.

Parallel-to-Serial Conversion

Shift registers find their most common use in serial-to-parallel and parallel-to-serial data conversion. Fig. 4-15 illustrates the parallel-to-serial conversion. The input data is transferred into the register in a parallel dump and shifted out as a serial transmission by the system clock. The clock may be continuous or consist of a burst of pulses initiated by the data transfer pulse.

Serial-to-Parallel Conversion

The serial-to-parallel converter illustrated by Fig. 4-18 receives serial data continuously. The data may be read out as a parallel

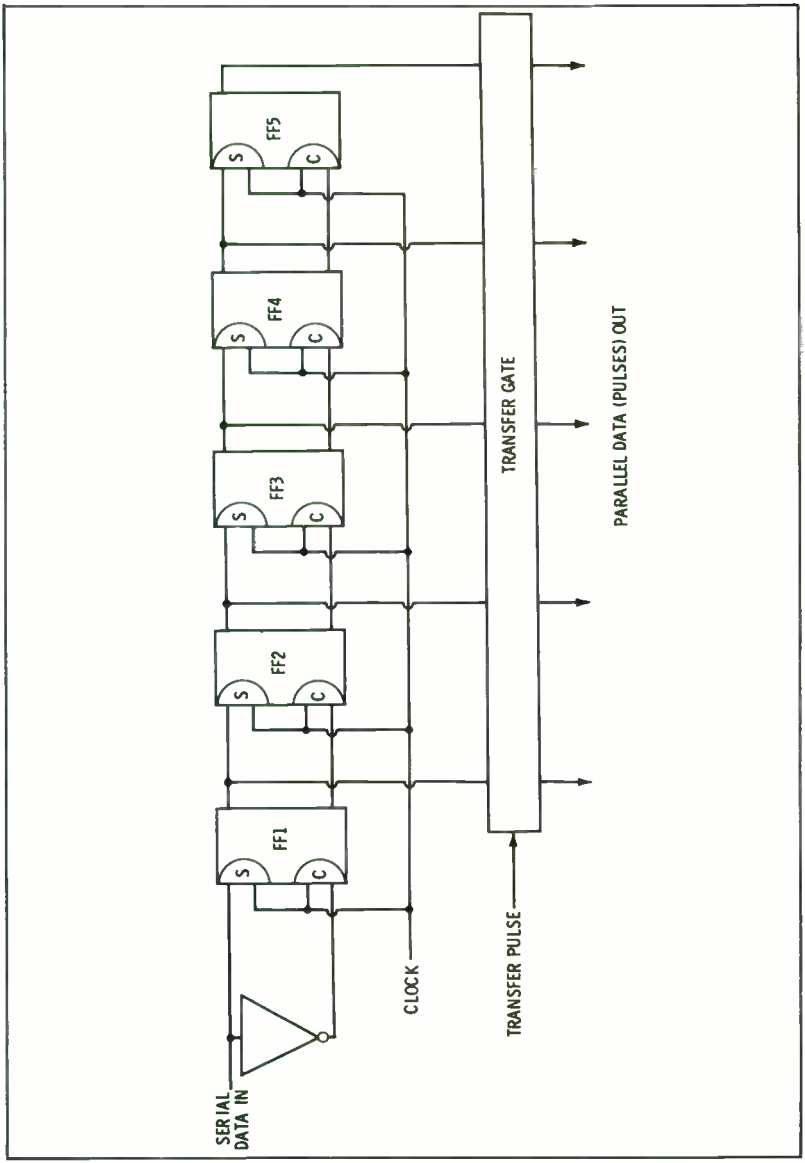


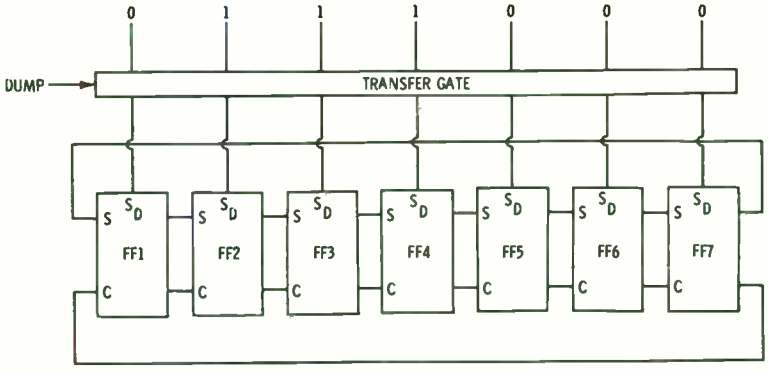
Fig. 4-18. Serial-to-parallel conversion.

dump only at the time when each data bit is known to be in its proper position in the shift register. This must be determined by external timing or by a synchronizing code word in the data stream. An instantaneous, pulsed sample of parallel data may be

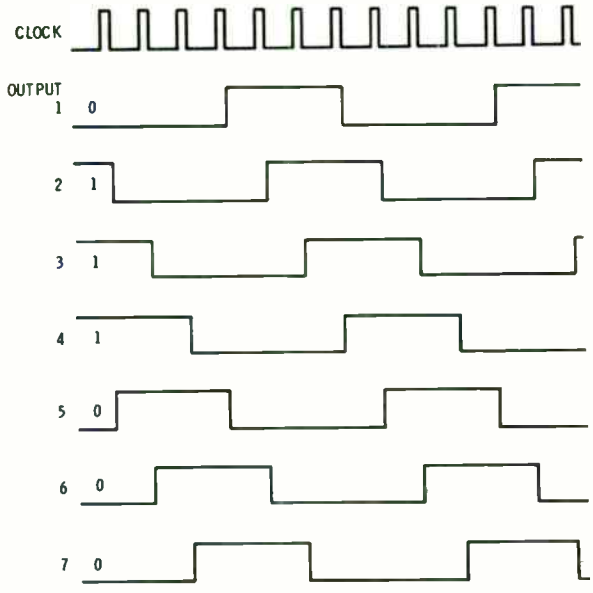
taken by the use of a transfer gate, or, if the parallel data is to be held for an indefinite period of time, the pulsed output of the transfer gate may be used as the input to a storage register.

Serial Pattern Generation

Fixed serial code patterns, such as those which are used for synchronization tags on serial data transmissions can be generated by a shift register. The code pattern is wired to the input transfer



(A) Circuit.



(B) Waveforms.

Fig. 4-19. Recirculating shift register.

gate as dc 1's and 0's dumped into the register, and shifted out with the data.

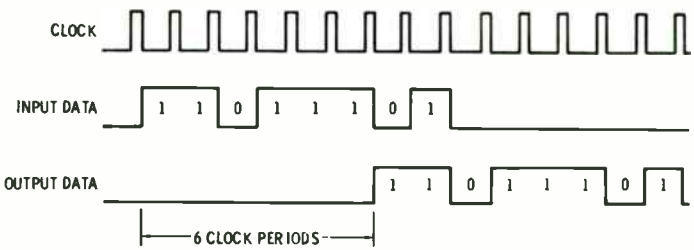
When the output of a shift register is connected back to the input to form a loop, a recirculating pattern is generated. Fig. 4-19 illustrates this application. If, for example, 0111000 is dumped into the recirculating register one time, the same code will circulate indefinitely until interrupted by a new message. Waveforms at all stages are identical and time-staggered one clock period per stage, as shown in Fig. 4-19B.

Delayed Data

The requirement for temporary storage of a stream of data often arises in a complex data processing system. In a continuous process, data might be fed serially to one part of the system and be needed in exactly the same condition several clock times later in another part of the system. This requirement can be handled by the use of a high-speed memory, but is usually accomplished much more simply by the use of a shift register of the necessary length. Data is fed serially to the front end of a shift register, as shown in Fig. 4-20, and is taken out of a downstream stage, delayed by one clock pulse per stage from input to output.

The same technique is used in delaying parallel or serial-parallel data by an integral number of clock periods. In delaying parallel data, care must be taken to ensure that the output data transferred at exactly the time that the data bits are in the selected position of the shift register. In Fig. 4-21, a separate shift register is used to delay the output transfer pulse by the same number of

(A) Circuit.



(B) Waveforms.

Fig. 4-20. Serial data delay.

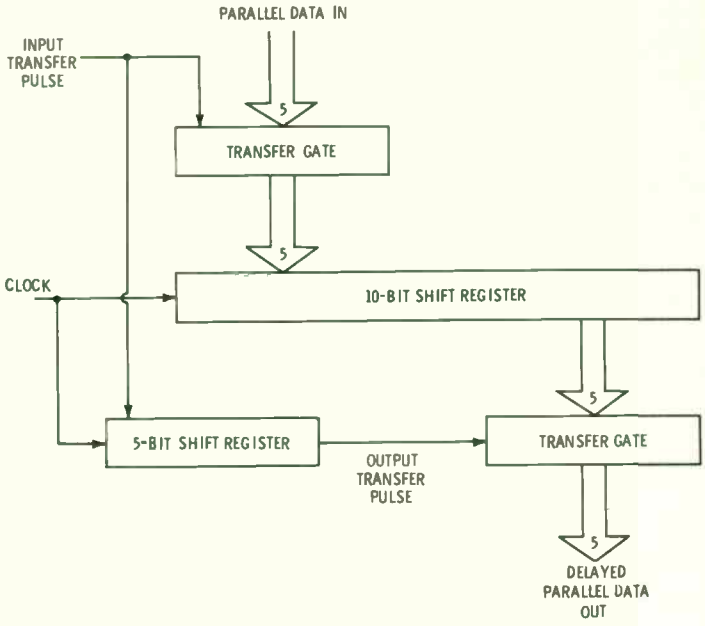


Fig. 4-21. Parallel data delay.

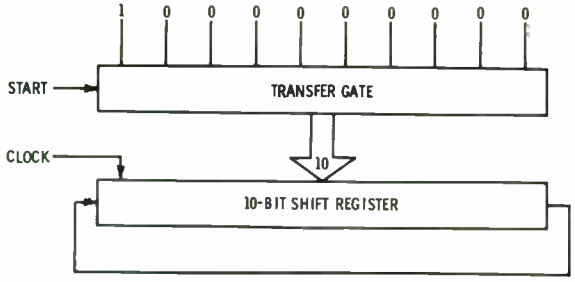
clock periods as the data. A binary counter and decoder could serve the same purpose as the auxiliary shift register.

Linear Counter

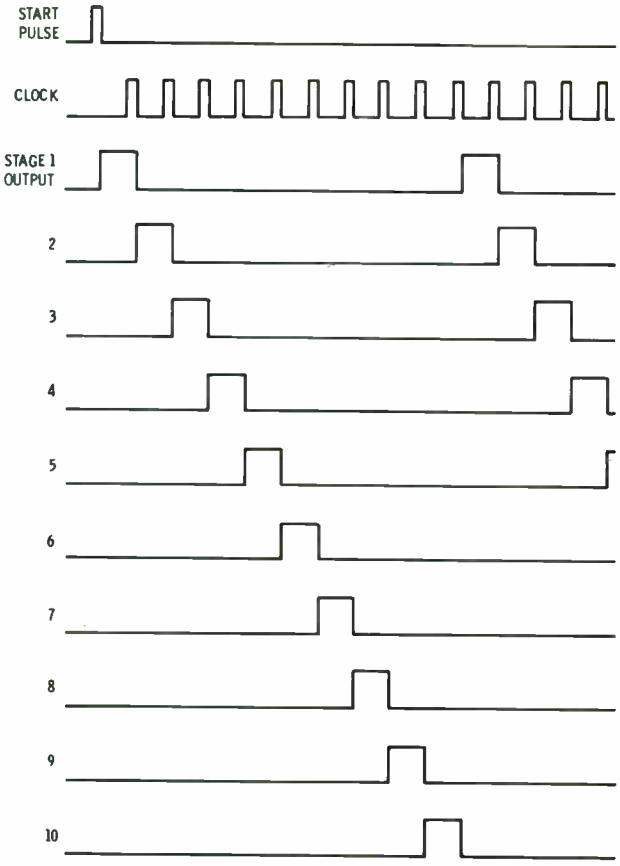
A recirculating shift register with only a single 1 circulating may be used as a linear counter or frequency scaler. Although the linear counter is much more complex and costly than an equivalent binary counter, it offers some unique features.

Fig. 4-22 shows a scale-of-ten linear counter and its waveforms. When power is first applied to this circuit, the flip-flops can assume any combination of 1's and 0's. To ensure that only one 1 level is propagated, and also to synchronize the counter to an external time, the 1 000 000 000 condition is dumped in as the starting condition.

As indicated by the waveforms, the linear counter performs frequency division. Each of the output waveforms has a fundamental frequency of input/*N*. The linear counter is especially useful in that each of its outputs is a separate time slot in the overall output time period. In this respect, the 10-element linear counter is *exactly* equivalent to a four-bit BCD binary counter driven by 10 decoder gates.



(A) Circuit.



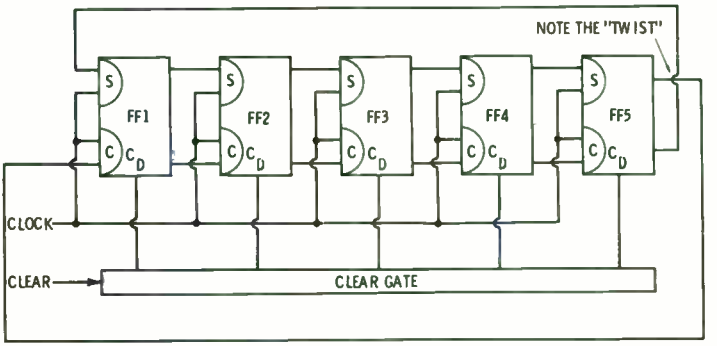
(B) Waveforms.

Fig. 4-22. 1/10 linear counter.

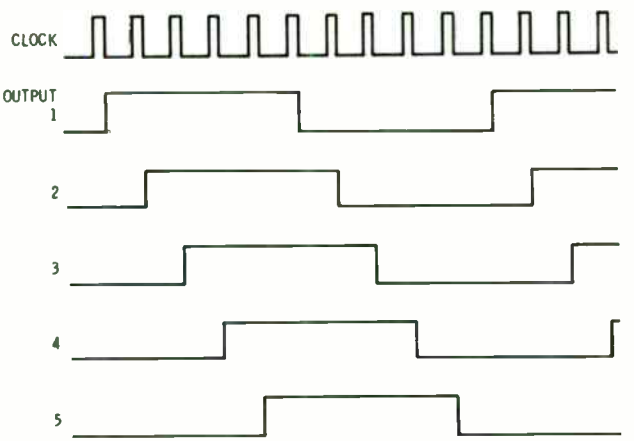
Johnson Ring Counter

Fig. 4-23 illustrates a special linear counter configuration with some unusual properties. At first glance, the circuit appears to be a simple five-stage linear counter with the output of the fifth stage returned as the input to the first. Notice, however, that there is a twist in the loop and that the inverse of the fifth stage output is returned to the first stage. Because of this twist, and the resemblance of the circuit to the paper-and-scissors figure known as the Moebius loop, this circuit is often referred to as the *Moebius loop counter*.

As indicated by the waveforms, when started from a 00000 clear condition, each stage produces a symmetrical square wave at one-tenth the input clock frequency, with each stage output displaced



(A) Circuit.



(B) Waveforms.

Fig. 4-23. Johnson ring counter.

one clock period from the preceding stage. For any number of stages, the frequency division ratio is $2N$.

Although this configuration is not commonly used, it exhibits one very valuable characteristic: When used at the front end of a counter or frequency divider, this circuit may be used to count pulses at a frequency considerably higher than the maximum rated frequency of its flip-flops. Assume that a frequency divider must be capable of dividing an input rate up to 10 megahertz, but the fastest flip-flop available is rated at a maximum output frequency of 2 megahertz. When used in the Johnson ring counter, the input flip-flop receives the 10-megahertz maximum clock frequency, but is required to toggle at a maximum of only 1 megahertz (in the case of the five-stage ring).

If the ring is required to operate as a counter as well as a frequency divider, the 10 distinct states can be decoded by the use of five input decoder gates.

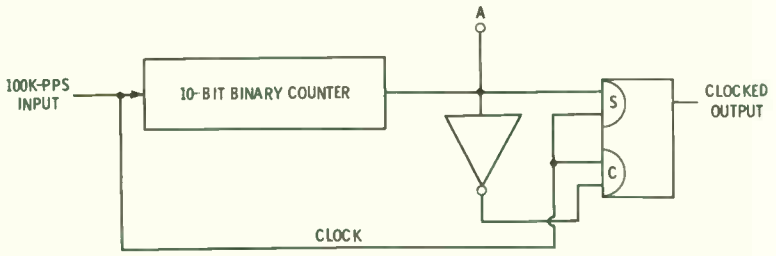
CLOCKING

More a system design concept than a specific circuit, the technique known as *clocking* permits an orderly, controlled operation of all components of a data system, and in certain applications, greatly improves the apparent performance of the components.

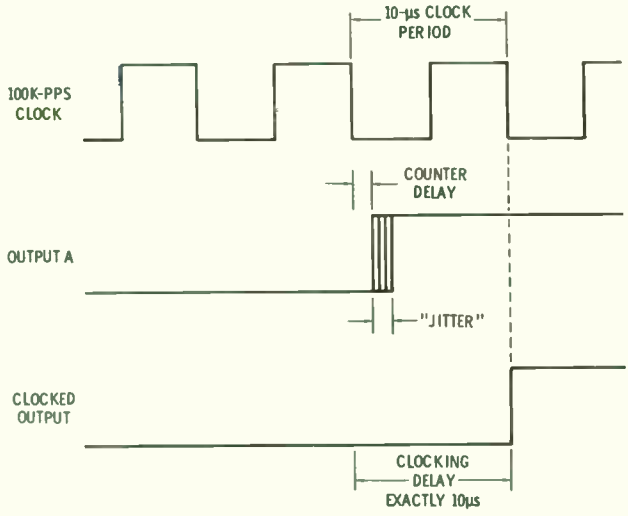
Specifically, clocking is the use of gated pulses from a central oscillator to trigger the active elements such as flip-flops. In an unclocked system, flip-flops and one-shots are usually controlled by the leading or trailing edges of the dc logic signals. In a clocked operation, all of the logical operations are performed by conventional dc gates as far as the set or reset inputs, and the final AND gate (usually part of the flip-flop) includes a clock pulse train. The flip-flop is triggered by the first clock pulse after the dc gate inputs are all true. Several benefits are derived from this technique:

1. *Delay Standardization*—Every element, active or passive, in a data system imparts some small time delay to the signal which it handles. Usually, the average value of this propagation delay is known for each circuit element. A typical flip-flop rated for operation up to 200 kilohertz might produce a fixed delay from input to output of perhaps 1.5 to 2.5 microseconds. Although the delay is fairly stable for a given flip-flop, two units of identical design might differ in a delay by as much as 1 microsecond. Included in this delay is a component of "jitter," or delay variation, equal to perhaps 200 nanoseconds.

If 20 flip-flops as described are connected as a binary counter, the overall fixed delay would have a steady value of 30 to 50 microseconds with an output jitter as great as 3 or 4 microseconds. In some applications, this performance would be completely satisfactory; however, if the counter output is required for precise timing or control, it might not be stable enough. The problem is exaggerated in an application such as a counter with several output decoders, in which even the "fixed" delay is not constant, since the decoded outputs do not necessarily include propagation delay of *all* stages. Without the use of some form of clocking, this type of problem can simply be reduced by the use of much higher speed flip-flops. The propagation delay and jitter values are closely



(A) Circuit.

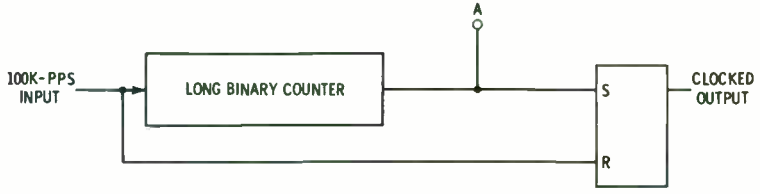


(B) Waveforms.

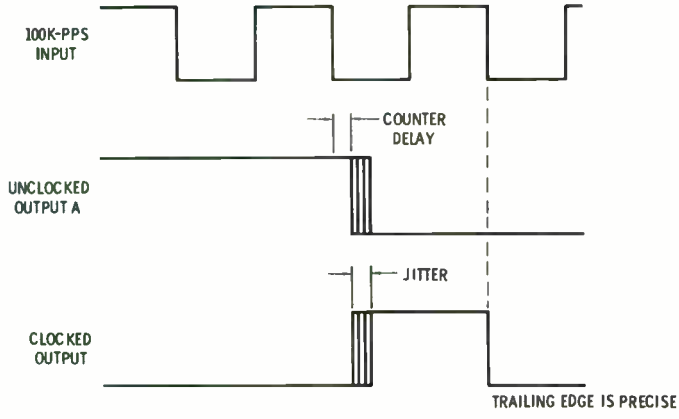
Fig. 4-24. Use of clocking to reduce counter delay and jitter.

related to the maximum rated speed. A much more practical solution, however, is the use of a clocked gate or clocked flip-flop at each output. Fig. 4-24 shows the time relations involved in a counter followed by a clocked flip-flop. In the case of the counter with several decoders, clocking may consist only of including the system clock signal as an additional input term on each decoder gate.

A somewhat different method of clocking the output of a counter is shown by Fig. 4-25. In this case, the flip-flop is set



(A) Circuit.



(B) Waveforms.

Fig. 4-25. Set-reset clocking.

by the sloppy output of the counter and is reset by the clock pulse. Consequently, the leading edge of the flip-flop output signal is no better, in fact slightly worse, than the unclocked signal, but the trailing-edge timing is as good as the clock plus the random effect of only one flip-flop.

Some caution must be applied with the use of simple clocking as described. As a general rule, in the case of a counter, the clock pulse-to-pulse period must be somewhat greater than the maximum unclocked delay through the counter. This ensures that the output is *always* delayed ex-

actly one clock period from the input signal, if the input is also timed by the clock. If this general rule is not followed, and several outputs are derived from a clock and decoders, all of the outputs will be coincident with a clock pulse, but not necessarily *the same* clock pulse. The requirement for making the master clock period greater than the overall delay through a unit is by no means a system design limitation. A 20-bit counter with a 40-microsecond propagation delay, for example, may be used with a 100-kilohertz clock by the use of intermediate clocking. The counter, or any other long logical chain, may be separated into smaller segments with a clocking stage inserted between the segments. Keeping in mind that exactly one clock period is lost at each clocking stage, the overall delay is easily managed and compensated.

2. *Sampling and Noise Considerations*—Where data is introduced into a system, the data bits may be presented as pulses, or as binary dc levels. In the first case, if the input circuit is designed to recognize a pulse of some arbitrary amplitude as a data 1, every noise spike of the proper polarity and acceptable amplitude will enter the system as an erroneous data bit. Using the second method, the input data level is used as a gating signal for the master clock, or for a sampling pulse derived from the clock. For example, a line carrying serial data at a 100-pps bit rate need only be sampled once every 10 milliseconds. Using a two-element gate with the data line on one input and a 2-microsecond, 100-pps clock on the other input provides a noise improvement factor of 5000, since the gate is only open to the data line 2 microseconds in the 10,000-microsecond period. Where noise is a problem, as is often the case, it is wise to use as narrow a sampling pulse, or aperture time, as the circuits will tolerate. This design philosophy is usually applied not only to input circuits, but throughout most data systems. Because of the extremely fast transients, tight packaging, and clock wiring in typical digital systems, high-frequency spike noise is an ever-present nuisance, and all practical safeguards, such as very narrow clock pulses, must be employed against its effects.

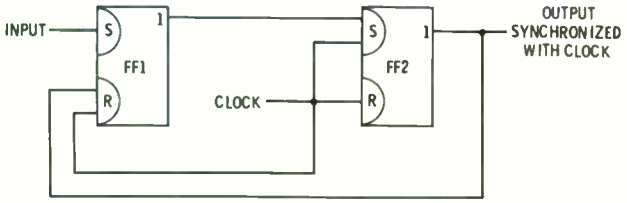
PULSE SYNCHRONIZATION

At the interface, or interconnection point, of two independently timed data systems, the problem of pulse coincidence often arises. Suppose that a flip-flop in the receiving system is to be set by a 1k-pps external input signal and reset by a locally generated 100k-pps clock. Let us also assume that the flip-flop requires at

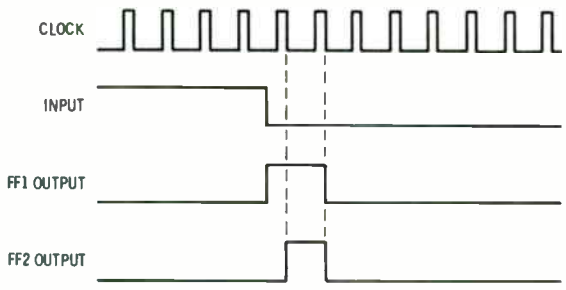
least 2 microseconds stabilizing time between inputs for reliable and predictable operation. If, at some instant, the input pulse occurs one microsecond before the following clock pulse, the flip-flop may or may not respond to the input, and, if it does respond, it may or may not respond to the reset pulse. Even worse, if the flip-flop responds to both signals, its output pulse may be one microsecond or less in width, and possibly be lost at the next stage. This condition, known as "slivering," must be prevented by some method which guarantees that the output of the interface flip-flop is always at least the minimum allowable width of 2 microseconds.

There are two basic methods of pulse synchronization which are usually applied to correct this condition. The first, which is straightforward but sometimes impractical, is to control the phase of the local clock to maintain a safe time relation with the incoming signal. If more than one input signal is to be handled, or if the input signal rate is not steady, this method cannot be used. Most data systems use a digital technique of synchronization in which the input pulse is delayed one whole clock period to avoid an interference or sliver condition. Fig. 4-26A shows one of the many possible variations of this type of circuit.

In the example, FF1 is set by the incoming pulse, opening the gate. If the gate opens in sufficient time to pass the next clock pulse, that clock pulse sets FF2, producing the leading edge of the



(A) Circuit.



(B) Waveforms.

Fig. 4-26. Pulse synchronization.

output pulse. The next clock pulse resets both FF1 and FF2. If the first clock pulse arrives so soon after the starting signal that the gate passes only a sliver, FF2 may or may not be set, but this is no problem since the gate remains open until FF2 has been set and reset. If FF2 is set by a sliver, the next clock pulse will be applied to both inputs of FF2, causing it to reset. In either case, FF2 will be set only once and will remain set for exactly one clock period.

CLOCKS

Many data systems of moderate size operate on a continuous basis, constantly computing and processing changes from input to output. If, at certain points along the computing path, small time delays are required, triggered delay elements such as one-shots or delay lines may be inserted where needed. In large data system, however, an impractical number of discrete delay elements would be required, and often there is a need for sequential, timed control of operations. This need is satisfied by the use of what is called a system clock.

A system clock may be as simple as a single astable multivibrator, or it may include several oscillators, counters, shift registers, and decoders, to satisfy all of the sequentially timed steps in a large data processor.

In the usual application, a common central clock signal is developed by a suitable oscillator or from an externally supplied continuous frequency source. The choice of clock frequency is dictated by the speed required in the data process, and must be limited to a value which is safely below the limitation of the circuit components or logic elements.

As a rule, the clock signals which are distributed throughout a system are in the form of very narrow true-level pulses. Using input-gated elements such as the standard flip-flop and one-shot, the input gates are enabled by the incoming logic levels *and* the clock, and the triggering action occurs at the trailing edge of the clock pulse. The use of a very narrow clock pulse provides a significant improvement in the overall noise immunity of a system, since a noise pulse on a data line can be effective only during the time when an input gate is enabled by the true level of the clock. For example, a clock signal consisting of one-microsecond pulses at a 1k-pps rate would reject the effect of 99.9 percent of any noise present.

Fig. 4-27 illustrates the generation and distribution of a clock signal in a very simple application. The clock frequency is generated by a 100k-pps astable multivibrator and shaped into a train of 0.5-microsecond pulses by a one-shot. As shown by the wave-

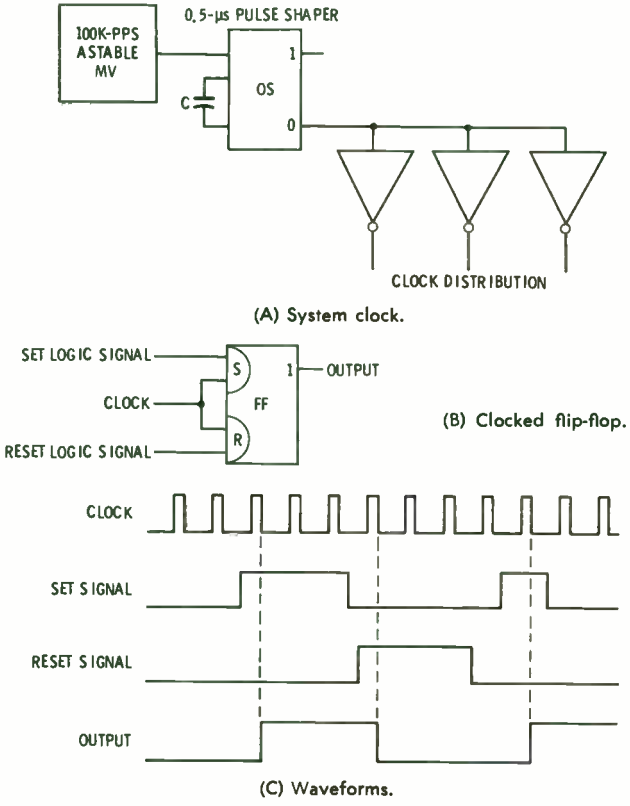


Fig. 4-27. Typical system clock.

forms, the clocked flip-flop (Fig. 4-27B) responds to its set and reset logic signals only at clock time.

The use of a clocking scheme such as that illustrated by Fig. 4-27 is usually carried through the entire system, with one oscillator supplying buffered clock pulses to all sections as required. Sometimes, a secondary clock signal derived from the same oscillator, but of opposite phase to the master clock, is used to prevent the condition known as *interference*. This condition occurs when the propagation time through a portion of the system is equal or approximately equal to the clock period, so that the output of that circuit is in a transient state at the instant of the next clock pulse after the input clock pulse. The use of an alternate clock phase or a delayed clock for the output sampling ensures that the output is sampled when the signal, or data, is stable.

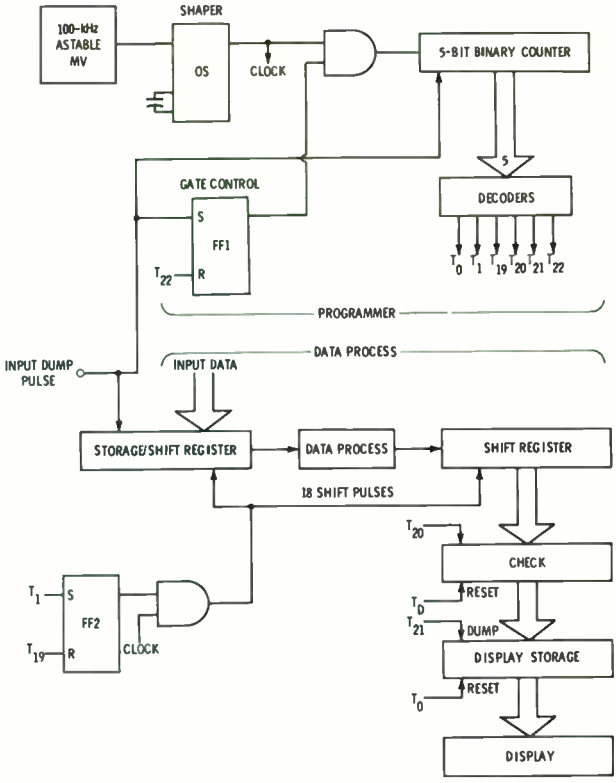
The use of clocked logical operations reduces the overall speed of a process somewhat, since operations cannot sequentially occur

less than one clock period apart. This slight reduction in speed is more than compensated for by the orderly and smooth flow of a clocked process.

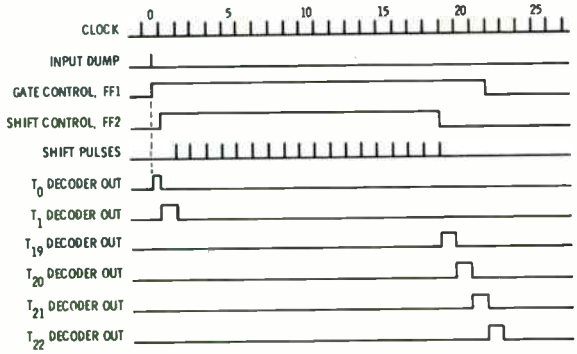
A more complex clock, sometimes known as a *sequencer* or *programmer*, is shown by Fig. 4-28. This type of circuit is used to control a data process which is performed as a repetitive sequence of events, with each system operational cycle covering a span of many clock periods. Suppose that the illustrated clock circuit is designed to control a system which includes several data channels, each channel receiving parallel data which is sampled at the start of each system cycle. Simply for illustration, let us assume that all channels are processed independently and simultaneously, and that the data must be dumped into an input storage register, be shifted through an 18-step arithmetic process, be checked for parity errors, and then dumped into an output display storage register. The system shown in Fig. 4-28 satisfies all of the requirements for controlling this system. The basic clock source is a free-running 100-kilohertz oscillator or multivibrator which is used throughout the system. This signal is gated into a five-bit binary counter which, in combination with several decoders, serves as the system programmer. The data sampling (dump) pulse which is supplied by the data source, resets the counter to 00000 and sets the gate control flip-flop, permitting the clock pulses to flow into the counter.

From the instant of reset until *after* the counter has received the next clock pulse, the counter remains in the 00000 condition. This condition is recognized by the 0 decoder gate which furnishes a reset signal to various elements in the system. During the interval T_1, which is the time between the first and second pulses after reset, the T_1 decoder gate is on, setting FF2. FF2 remains set until it is reset by T_{19}. From T_1 through T_{19}, the control gate (FF1) is open, allowing a train of 18 clock pulses to pass. At T_{20}, the parity check control pulse is generated, followed by the display dump pulse at T_{21}. This completes the required system cycle, and the programmer is turned off by T_{22}, which resets the gate control flip-flop, FF1. The flip-flop remains in the T_{22} state until the cycle is again initiated by an input sampling pulse.

Although the circuit of Fig. 4-28 appears to satisfy the clocking requirements of the system as it was described, this circuit has one serious weakness. The 100-kilohertz clock oscillator is in no way related or connected to the externally supplied input sampling pulse which initiates the clock sequence. If, as illustrated, the sampling pulse would always appear midway between clock pulses, the clock and sequencer would work as described. Actually, the sampling pulse will occur at random with respect to



(A) Circuit.

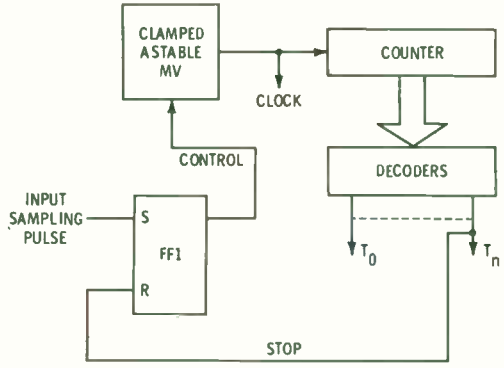


(B) Waveforms.

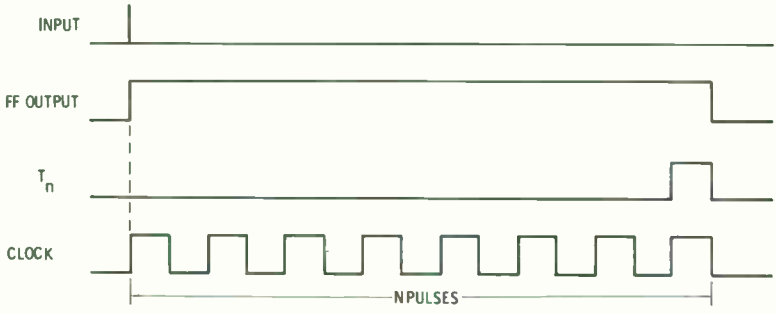
Fig. 4-28. Use of system programmer.

the local clock, occasionally arriving in coincidence. In this case, the first stage of the counter may receive a reset signal and an input pulse at the same time, or the input control gate may be opened in the middle of a clock pulse, passing only a narrow sliver. At any rate, the overall functioning of the programmer, and most likely the entire system, will be erratic in this interference condition.

The problem of synchronization to the external signal may be handled in one of several ways. If the sampling pulse rate is constant and relatively high, it is sometimes practical to exercise frequency and phase control over the local clock oscillator through a phase comparator loop, effectively tying the local oscillator to the clock at the source of the data. A simpler method, shown by Fig. 4-29A, makes use of a clamped local clock which is turned on by the input sampling pulse. Properly designed, a clamped oscillator or multivibrator will start up in a precisely predictable



(A) Circuit.



(B) Waveforms.

Fig. 4-29. Clamped multivibrator clock.

manner, guaranteeing that the start of each system cycle is in step with the local clock. This method is used frequently in systems which do not require a continuously running system clock between major system cycles.

In a system of such magnitude and complexity as to require a continuous common local clock and be capable of handling several unsynchronized input channels, it is usually most practical to synchronize each of the input channels to the local clock. This method of pulse synchronization, previously described, generates a synthetic input pulse, timed by the first clock pulse following each real input pulse. The net effect of this type of synchronization is to delay all internal action for a variable period up to a maximum of one clock cycle.

Where serial data is to be carried over long communication channels, the data signal is usually used as a keying waveform or modulation envelope for a higher-frequency carrier signal. Most data transmitters of this type are designed to produce the carrier signal in a synchronous or coherent relation with the data stream. The data receiver and processing circuits, receiving a signal of this nature, may be designed to extract the carrier, if it is continuous, and use the squared carrier waveform as the local clock signal. In a situation where the incoming carrier frequency is not sufficiently high for all of the local requirements, the carrier clock may be used to control the frequency and phase of a higher-frequency local clock. Fig. 4-30 illustrates a clock synchronizer of this type.

The input data is received on a 1-kHz carrier which is converted to a 1k-pps square wave by a Schmitt trigger or inverter circuit.

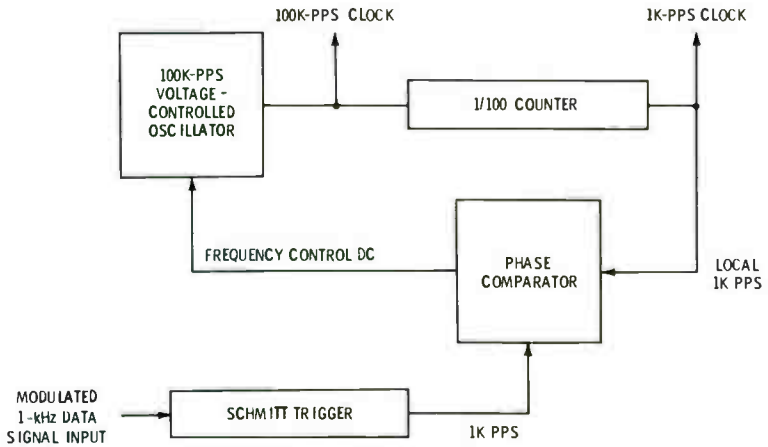


Fig. 4-30. Clock synchronization loop.

The incoming 1k-pps is phase-compared with a local 1k-pps signal derived by counting down the local 100k-pps clock. The phase comparator produces a dc control signal which is proportional in amplitude and polarity to the phase relationship of the two 1k-pps signals. This signal is used as a frequency control into the 100k-pps oscillator, steering the oscillator frequency as required to maintain exact coincidence of the two 1k-pps signals. This action ensures that not only the local 1k-pps clock but the 100k-pps and all other rates derived from the 100k-pps oscillator have a solid, predictable, relationship with the incoming data. An ancillary benefit is derived from the indirect use of the incoming carrier frequency in that the effect of carrier jitter is eliminated. If the input carrier signal is noisy or if an occasional carrier cycle drops out, the local clock operation is not affected, since the 100k-pps local oscillator runs continuously and uses the incoming signal only for average frequency steering.

ADDERS AND SUBTRACTORS

From a practical point of view, no basic difference exists between the mechanization of the addition and the subtraction processes. Generally, both circuits are similar; the distinction appears mainly in the manner in which the input data is presented to the circuits.

To understand the operation of a binary circuit, one must be familiar with the logical rules which govern the addition of two binary numbers. These rules are analogous to the rules of decimal addition. Multibit binary addition is performed one bit at a time, starting at the least significant bit, progressing to the most significant bit. As with the decimal process, certain combinations require that a bit be carried from one step to the next.

Any number of binary words may be added simultaneously. However, most arithmetic circuits perform single addition of only two words at a time, and use consecutive operations for additional words. The rules for adding two multibit binary words are as follows:

1. Starting the LSB, examine both bits. If both are 0, the sum bit and the carry bit are 0. If one bit is 1 and one bit is 0, the sum bit is 1 and the carry bit is 0. If both bits are 1, the sum bit is 0 and the carry bit is 1. The carry bit is used in the next step.
2. Advance one step from the LSB and examine both bits and the carry bit from the preceding operation. Considering the three, if all are 0, the sum bit and the carry bits are 0. If one of the three is 1, the sum bit is 1 and the carry is 0. If two are

1, the sum is 0 and the carry is 1. If all three are 1, the sum is 1 and the carry is 1.

3. Advance one more step toward the MSB and repeat step 2.
4. Repeat step 3 until all bits have been processed. If, after the last step, there is a carry bit, this becomes the MSB of the sum. In other words, if two 10-bit words are added and the last step produces a carry 1, the sum will be an 11-bit word.

This process may be carried out serially, using one adder circuit and two shift registers which feed the inputs to the adder, one bit at a time, or by a parallel addition which requires one adder circuit for each input bit. When a parallel process is used, the section which handles the LSB is simpler than the others, since only two inputs, A and B, must be processed, while the others must handle three input signals: A, B, and Carry. The simpler type is called a *half adder*; the three-term circuit is called a *full adder*.

Logically, the adder circuits are an exact mechanization of the rules of binary addition. In the case of the half adder, the Boolean expressions for the sum and carry outputs are:

$$\text{Sum} = AB + BA$$

$$\text{Carry} = AB$$

The expression for the sum term will be recognized as the exclusive-OR function; the carry circuit is one AND gate.

For the full adder, the sum and carry expressions are:

$$\text{Sum} = A\overline{B}\overline{C} + B\overline{A}\overline{C} + C\overline{A}\overline{B} + ABC$$

$$\text{Carry} = AB + BC + AC$$

Interpretation of the full-adder expression shows that the sum is 1 if only one, or all three inputs, are 1. The carry is 1, if any two, or all three, inputs are 1. Using the sum and carry expressions, the full-adder circuit could be realized as shown by Fig. 4-31, although there are many other circuit variations which can accomplish the same logical function.

In a system application where two binary numbers are to be added serially, the overall block could be laid out as shown by Fig. 4-32. This requires three shift registers (unless the inputs are already in serial format): one for each of the inputs and one for the sum. The shift clock may be a burst of exactly the required number of pulses, or it may be a continuous clock. If the clock is continuous, the output sum word is only valid during one clock period, after the last input bits have been processed. In this case, the output must be dumped into a storage register if it is required as a parallel word. Usually, for the sake of simplification, the

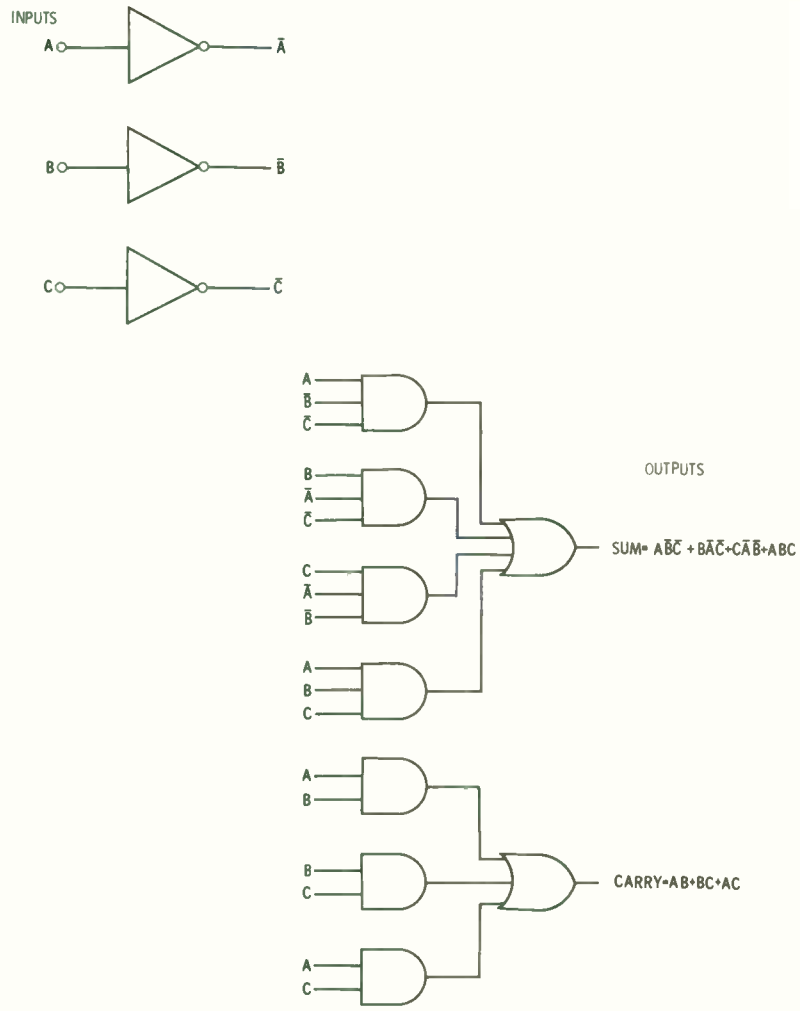


Fig. 4-31. Full-adder logic.

block is arranged as shown by Fig. 4-33 in which only two registers are required, since one of the registers serves a dual purpose, acting as an input data register and also receiving the output data.

Subtraction is usually performed by taking the negative of the number to be subtracted (the subtrahend) and adding it to the minuend. The result is the correct difference.

In a natural binary system, the arithmetic *negative* of a number is produced by complementing the number and adding 1. A binary

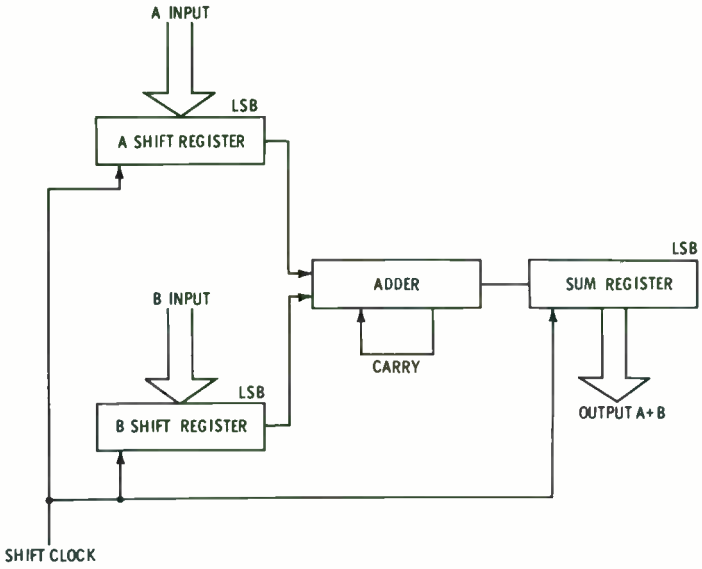


Fig. 4-32. Serial word addition.

number is *complemented* simply by inverting each of the bits. The complement of 01101, for example, is 10010.

To illustrate this process, suppose we wish to subtract binary 5 (00101) from binary 23 (10111). To the binary 23 we add the negative of binary 5, which is the complement plus 1:

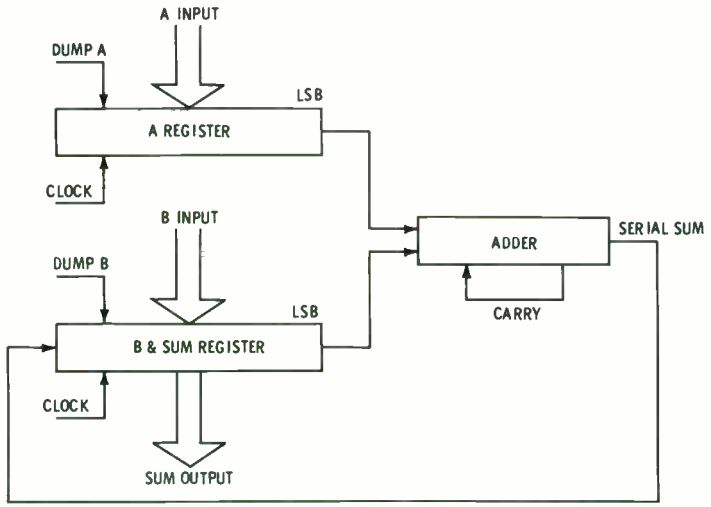


Fig. 4-33. Addition with two registers.

	10111	Binary 23
	11011	(Complement 5) + 1
1	10010	Difference and overflow

The difference appears as a six-bit word. The new MSB, or overflow, will always appear in the difference when the subtraction produces a *positive* difference. Using the overflow bit as an indicator, the actual difference is read as 10010, or binary 18, the correct answer.

When a large number is subtracted from a small number, the overflow bit is zero, indicating that the difference is negative and must be complemented and corrected. Consider the problem of subtracting binary 23 from binary 5:

	00101	Binary 5
	01001	(Complement 23) + 1
×	01110	Difference: no overflow
	10001	Complement the difference
	1	Add 1
	10010	Negative difference is 18

The first addition step produces no overflow, indicating a negative difference. The difference is complemented and corrected by +1, and the answer is seen to be negative 10010, or −18.

A serial subtractor is shown by Fig. 4-34, in which the minuend and subtrahend are shifted into a full-adder circuit. The subtrahend is complemented by the action of the inverter, and the +1 correction pulse is set into the adder as an initial condition. The

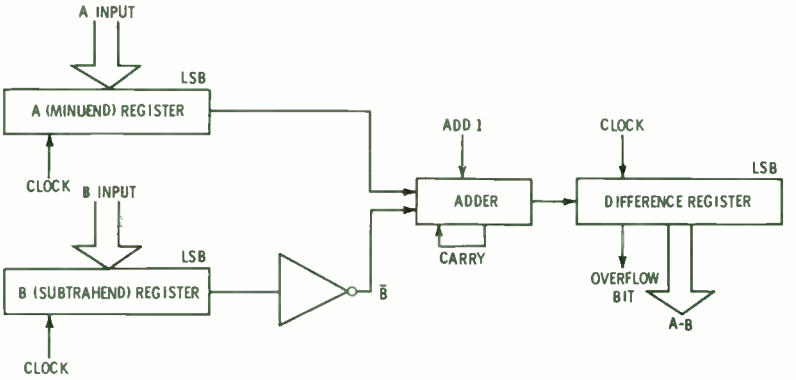


Fig. 4-34. Serial binary subtraction.

difference is shifted from the adder into an output register of one bit more length than the maximum positive difference, providing the overflow bit. If the subtractor were required to handle a negative difference, the output complementing and correcting would be controlled or selected by the absence of the overflow bit.

Using the same logical rules, binary subtraction may be mechanized as a parallel process. This method involves considerably more hardware, but requires only a fraction of the computing time used by a serial process.

Addition and subtraction of BCD and other special binary codes are carried out by a generally similar method. The specific rules of machine arithmetic are designed to satisfy each code, and some special numeric codes have been designed to facilitate complementing and processing.

DELAY GENERATION

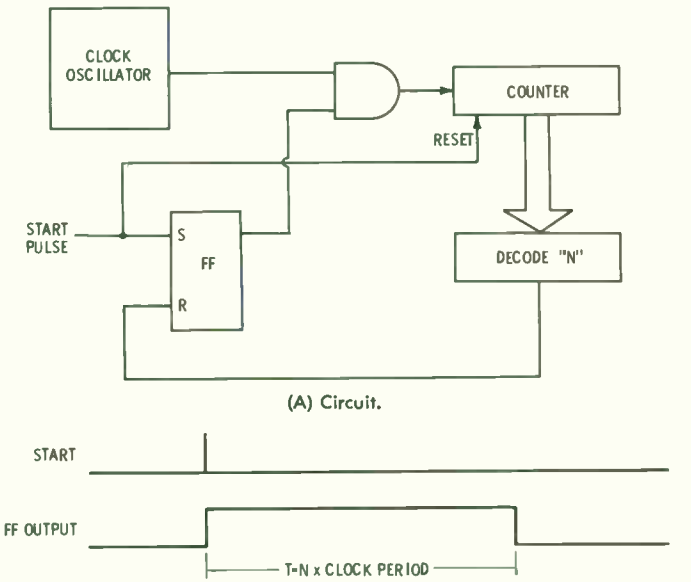
Certain fields of instrumentation, particularly radar and its related branches, require precisely controlled variable time-delay circuits. Before the use of digital techniques became practical, generation of time delay, both fixed and variable, was accomplished by the use of reactive circuits. Short-term fixed delays of good precision and stability were usually controlled by the use of artificial transmission lines in the form of lumped-constant LC delay lines. Longer delays, particularly those which were required to be variable or easily adjusted, were generated almost exclusively by the use of RC circuits.

For applications requiring only moderate accuracy, RC analog delay circuits are still quite commonly used in digital systems, as indicated by the popular use of one-shot multivibrators. A good one-shot, when fed by a regulated power supply, can be expected to maintain about 5-percent delay stability. Greater accuracy and stability can be achieved by the use of more sophisticated active RC circuits, such as the bootstrap integrator and other Miller effect circuits, but the best of these circuits, as always with any analog computation, are limited by the stability and quality of their components. Without a laboratory environment and hand-picked components, analog delay circuits become impractical if accuracy better than about 0.1 percent is required.

On the other hand, digital delay generation and control circuits, while generally more complex than analog delay generation and control circuits, are practically unlimited in their potential accuracy. Just as the accuracy of a digital arithmetic computation can be improved by handling more bits, the accuracy of a digital time delay is improved by adding more logical hardware.

The essential part of any digital delay generating system is a counter of some form accumulating pulses at a fixed and precise rate. If, for example, an input pulse starts the operation of a 1k-pps counter from reset, and the counter output is decoded at T_{150}, it is easily seen that the time delay from start to output is 150 pulses—in this case, exactly 150 milliseconds. It should also be clear that the only significant source of error is the inaccuracy of the 1k-pps clock oscillator, and this can be maintained without difficulty to a part in a million, or many times better if such accuracy is necessary.

Consider the circuit of Fig. 4-35, in which a free-running oscillator is used as the delay clock, whose output is gated on by an input pulse and gated off by the decoded output of a counter. The accuracy and precision of this generalized delay system are defined by three factors: frequency error, quantizing error, and granularity. The *frequency error* is determined entirely by the quality of the clock oscillator, and the magnitude of its effect is proportional to the oscillator frequency inaccuracy and the selected length of delay. If, for instance, the oscillator frequency is one percent above its nominal value, a one-millisecond delay be in error by 10 microseconds, while a one-second delay would be would be in error by 10 milliseconds.



(A) Circuit.

(B) Waveforms.

Fig. 4-35. Digital time-delay generation.

The component usually called *quantizing error* occurs only in a delay generator in which the counting clock is free-running and independent of the starting pulse. In this case, the error is caused by the random time lost between the start pulse and the first clock pulse which is counted. The magnitude of this error is random, but is limited to no more than one clock period. Unless a compensating provision is made, this error is always in the positive direction, producing a slightly higher delay than is desired. Within practical limits, the quantizing error can be reduced to an insignificant value simply by extending the length, or number of stages, of the counter and increasing the clock frequency accordingly. For instance, if a delay generator with one-second maximum delay is required, a three-decade counter and a 1k-pps clock could be used, producing a maximum delay of 0.999 second with a 0 to 1-millisecond quantizing error. If this error were not tolerable, a four-decade counter and a 10k-pps clock could be used for the job, with a maximum quantizing error of 100 microseconds.

The third consideration in the performance of a digital delay control is *granularity*, or *resolution*. Granularity has a direct bearing on delay precision rather than on accuracy, and it is determined solely by the length of the delay counter. A three-digit BCD counter is capable of a maximum delay of 999 clock cycles, and has a granularity of 1/1000. A six-digit BCD counter has a granularity of 1/1,000,000. Assuming no other sources of error or imprecision, generation of a delay of 0.5632 second obviously requires at least a four-digit counter, while a delay of 0.56 second may be controlled by a two-digit counter. Both would be equally accurate, but would differ significantly in precision.

Considering the three basic sources of error, it should be apparent that there is almost no practical limit to the realizable performance of a digital time-delay generator, if the application warrants the cost and complexity.

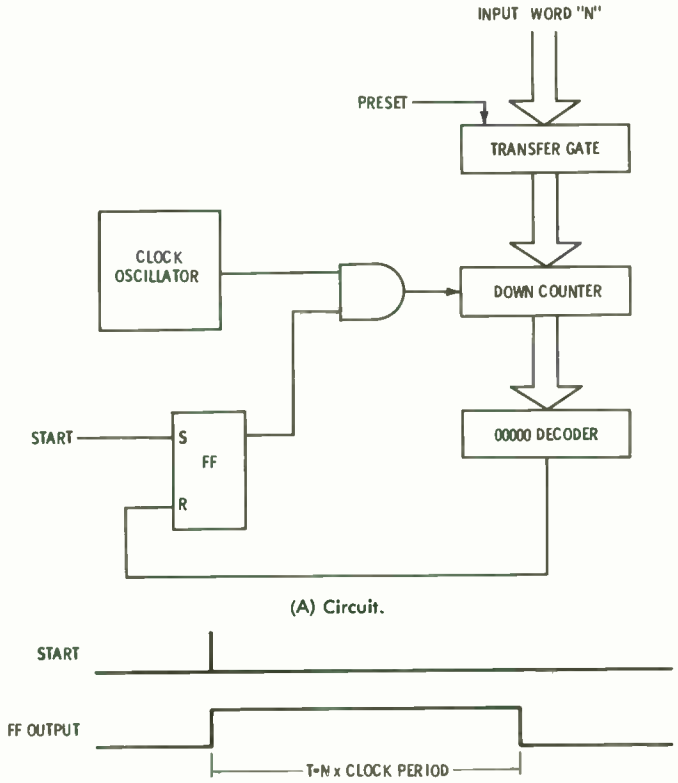
The primary source of error, clock accuracy, may be reduced to an insignificant amount by the use of a crystal-controlled clock oscillator. An inexpensive, good-quality crystal clock will maintain one part in a million, while a more complex oven-stabilized crystal oscillator may yield accuracy and stability as good as one part in a billion (1×10^{-9}). For the rare applications where even greater accuracy is required, oscillators stabilized by molecular or atomic resonance effects are commercially available, specified to an accuracy as fine as 1×10^{-12}.

The quantizing error, as previously shown, cannot be completely eliminated in a high-precision delay system, but can usually be reduced to any tolerable level. A clock oscillator which is not free-running, but is keyed on by the start signal, can be used

to completely eliminate the quantizing error; however, this type of oscillator ordinarily does not exhibit the best frequency accuracy. The use of a longer counter and a correspondingly higher clock frequency is usually the practical solution to both the quantizing and granularity problems. The only practical limitation to this solution is the maximum frequency rating of the counter flip-flops.

There are two counter configurations commonly used in delay control: reset, count-up, and decode; or preset and count-down. Several factors dictate which of the configurations is most suitable for a given job. These include considerations such as fixed or variable delay generation, the required method of control, the nature of the input, and the required speed of operation.

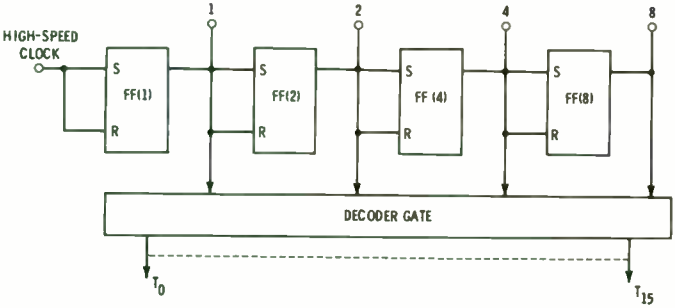
Assuming that variable or instantly selectable delay is required, and that the delay control signal is in digital form, the simplest solution is the binary preset–count-down method, shown by Fig. 4-36.



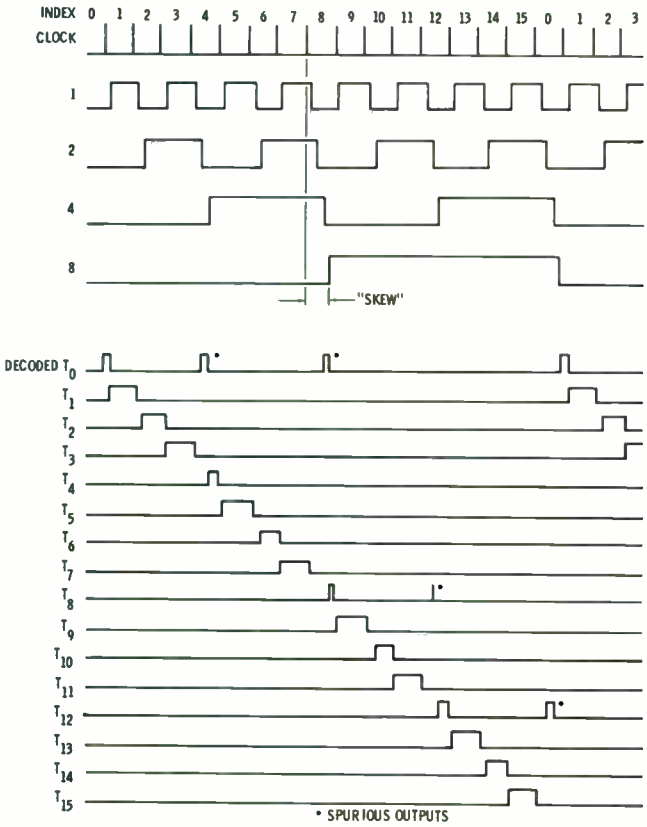
(A) Circuit.

(B) Waveforms.

Fig. 4-36. Preset/count-down delay control.



(A) Circuit.



(B) Waveforms.

Fig. 4-37. Effect of propagation delay.

Two main inputs control the system: the binary dc data which defines the desired length of delay, and the start pulse, which initiates the delay cycle. The control flip-flop is quiescently in the reset state, blocking the flow of clock pulses to the counter. At some time prior to the start pulse, the binary control word is dumped into the counter, establishing the initial state of each flip-flop in the counter. After the counter has stabilized from the presetting transient, it is ready to perform. The start pulse sets the flip-flop, opening the gate, and the counter proceeds to count downward from its initial state. After sufficient clock pulses have been received to bring the counter to the 00000 state, the flip-flop is reset, ending the cycle. In the circuit shown by Fig. 4-36, either the decoded 00000 pulse or the trailing edge of the flip-flop output signal may be used as the delayed output.

Although the starting pulse need not be synchronous or time-related with the data and its transfer pulse, some means must be included to ensure that a new data word is not dumped into the counter while the counting cycle is in process. This may be arranged externally, or accomplished by an inhibiting gate in the data transfer pulse line, preventing transfer during the interval in which the control flip-flop is set.

The circuit may be simplified slightly by eliminating the zero decoder gate and using the 1 transient of the last stage of the counter as the resetting signal. Just as an up-counting binary counter recycles to 0000 one pulse after 1111, a down-counter recycles to 1111 immediately after the 0000 condition. This method of control is extremely useful as a way of avoiding one of the serious effects of propagation delay.

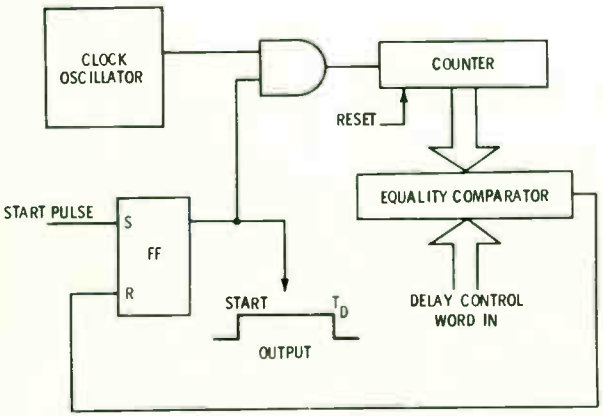
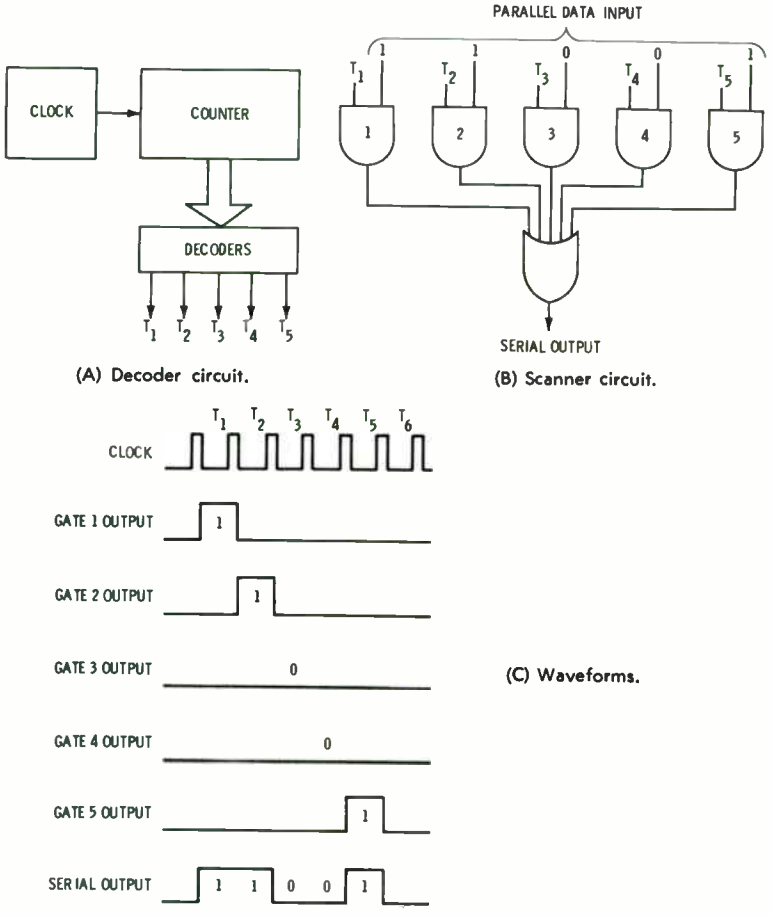


Fig. 4-38. Reset/count-up delay control.

When a counter is operated at a clock rate approaching its maximum rated frequency, the total propagation time, or "ripple-down" time through the counter, may equal or exceed one whole clock period. In this situation, decoding the counter output becomes very unreliable, and in extreme cases, impossible. As shown by Fig. 4-37, the decoded time slots are of varying widths and spurious outputs may be generated.

Because of this limitation of a decoder at extremely high frequencies, it is always advisable to use the recycling transient of the last stage of the down-counter to control the end of the delay cycle. The net effect of this method is an increase in the overall delay equal to exactly one clock period, plus the propagation de-



(A) Decoder circuit.

(B) Scanner circuit.

(C) Waveforms.

Fig. 4-39. Five-bit scanner logic.

fact, the scanner can be designed to operate directly from the clock-counter if the gates are sufficiently complex to perform the necessary time slot decoding.

When designed for production in quantity, the actual hardware in a scanning array, or matrix, is extremely simple. Using diode logic, the scanner of Fig. 4-39 consists of no more than fifteen diodes and six resistors, as indicated by Fig. 4-40.

A serial data word may be converted to parallel by the use of a circuit very much like a scanner in reverse, called a *decommutator*. In the five-bit decommutator shown in Fig. 4-41, the serial data is applied to all of the gates, and is distributed sequentially to lay. The total propagation delay is equal to the delay per stage times the number of stages, since the output transient is caused by the domino effect of all of the stages in cascade.

Exactly the same results can be produced by the use of an up-counting binary counter if the complement of the delay command is preset into the stages. When this configuration is used, the action must be stopped when the 11111 condition is decoded, or by using the *reset* transient of the last stage output.

The second basic arrangement of a delay counter is shown by Fig. 4-38. Starting from reset, the counter accumulates upward until the selected time value is detected by controlled decoder gates, or until an equality comparison between the counter output and the input control data is satisfied. This configuration is straightforward, but is also subject to the problems of high-speed decoding.

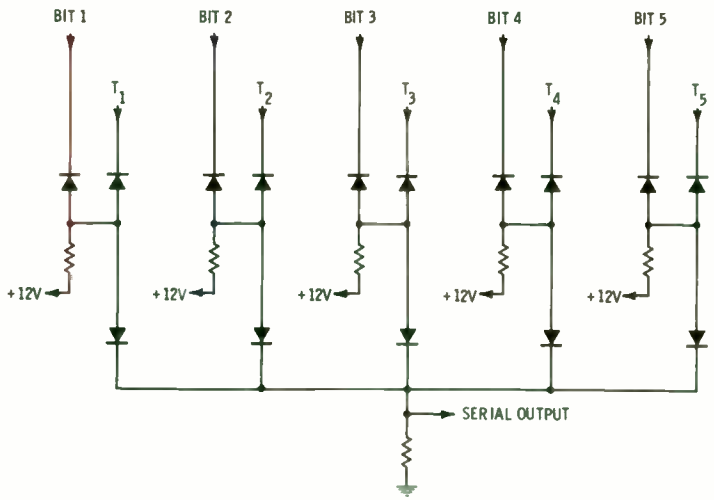
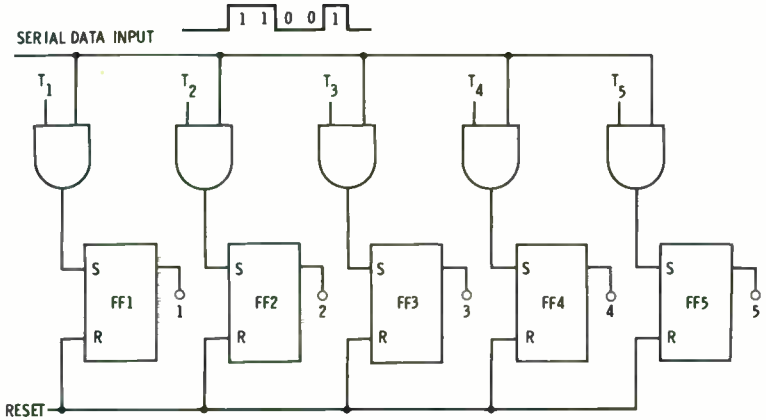


Fig. 4-40. Five-bit scanner circuit.

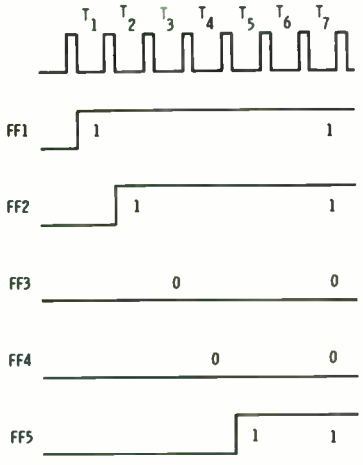
SCANNERS AND DECOMMUTATORS

It is often practical to use a set of sequentially energized AND/OR gates to convert a parallel data word to serial, rather than by the more conventional method of using a shift register. An array of gates used in this fashion is called a *scanner*.

Fig. 4-39 shows the logical arrangement of a five-bit scanner. Each of the AND gates is connected to one of the dc data-bit lines, and each gate is enabled from the decoder during one of the sequential time slots. Although the circuit as shown appears to be more complex than a simple five-stage shift register, consider a



(A) Storage register.



(B) Waveforms.

Fig. 4-41. Serial-to-parallel decommutator.

data system in which several words must be simultaneously converted from parallel to serial form. In this application, only one clock-decoder circuit would be required for controlling all of the scanners, and the individual scanners are relatively simple.

The detail design of the gating array is usually determined by the available standard logic hardware. It should be evident that the individual AND gates are not necessarily two-input gates—in fact, the scanner can be designed to operate directly from the clock-counter if the gates are sufficiently complex to perform the necessary time slot decoding.

When designed for production in quantity, the actual hardware in a scanning array, or matrix, is extremely simple. Using diode logic, the scanner of Fig. 4-39 consists of no more than fifteen diodes and six resistors, as indicated by Fig. 4-40.

A serial data word may be converted to parallel by the use of a circuit very much like a scanner in reverse, called a *decommutator*. In the five-bit decommutator shown in Fig. 4-41, the serial data is applied to all of the gates, and is distributed sequentially to each of the five flip-flops which serve as a storage register. The time slots which control the gates may be generated in any one of several ways, as with the scanner, but in the case of the decommutator a special restriction must be considered. For the decommutator to be meaningful, each time slot must turn on its gate at the instant in which the selected bit is on the serial data line. This, naturally, is controlled by the timing of the overall system, and requires that the clock-counter which generates the time slots be operated in some form of synchronism with the serial data cycle.

5

DISPLAYS

At the output of most data systems, and often at intermediate stages, some form of visual or permanent data display is usually provided. Although the terms are not firmly defined, the word "display" ordinarily implies an indication which may be visually monitored, such as a bank of data lights, while a "readout" is a more permanent graphic record of data.

DATA DISPLAYS

The most commonly used data displays are incandescent or neon lamps, usually front-panel mounted. In a parallel display of data, each lamp monitors one bit. This form of display is ideally suited to binary or BCD data presentation. Selection of the best type of lamp for use in a system is often a very difficult initial decision for the design engineer, since each of the choices presents advantages and limitations. Of the many available indicators, a few are listed along with their significant features.

Neon Indicators

A small plug-in or wired-in-place neon indicator is a reliable and effective one-bit display. Because of the extremely long life expectancy of a neon bulb, this type of indicator can be permanently wired into a suitable mounting on or behind a panel, for a simple and economical installation. Drawing only a few milliwatts, neon displays generate very little heat.

Neon indicators are sometimes unsuitable for use in areas which are sensitive to radio-frequency interference. Although most well-

engineered data systems are not especially susceptible to rf interferences, it is not unusual for a digital system to be installed in the vicinity of a communication or radar set, in which case the radiation generated by the neon switching action can be a serious nuisance.

The most severe disadvantage of neon indicators is the inherently high operating voltage required, in the neighborhood of 100 volts for reliable action. This demands that a system includes a power supply for the indicators only, since it would be impractical to attempt to operate logic circuits at that level. The high-voltage characteristics of the neon lamps, in turn, require that high-voltage driver amplifiers be included, using the system logic levels as inputs. With good circuit design, driver amplifiers which are capable of about 35 or 40 volts output swing will satisfactorily control neon indicators.

Usually the size of the data system and the number of indicators dictate whether or not the use of neon indicators is practical. Corollary to the high supply-voltage requirement, neons draw very little current, typically one milliampere or less, so that one very simple, unregulated dc supply may be sufficient for all indicators in a large system. Likewise, if a great number of indicators are needed, the driving amplifiers are usually packaged in the same manner as standard logic elements, reducing the cost and complexity.

For applications where only one or a few indicators are required, there are units commercially available which include in a single package a neon bulb, driver amplifier, and high-voltage power supply. In spite of the relatively high cost per unit of neon indicators they are an expedient solution to a limited display requirement.

The physical characteristics of neon indicators make them more suitable than incandescent lamps for high-speed applications. It is possible to visually display on neon an event lasting as short as 100 microseconds. Because of the characteristic color of the neon discharge, displays of this type may be filtered to produce shades of red or amber, but never blue or green.

Incandescent Lamps

Miniature incandescent data lamps are well matched to typical digital logic circuits, since the operating voltage of most data circuits is in the same general range as that of low-voltage lamps. The advantage of low-voltage operation is somewhat outweighed by the relatively high current required and the heat generated by incandescent bulbs. A typical bulb dissipates about one-half watt. When panel displays of 100 bits or more are required, the prob-

lems created by lamp heat might become so severe as to make the use of incandescent bulbs impractical.

Although "extended-life" incandescent lamps are available, the inherent probability of burn-out usually requires that the bulbs be mounted in receptacles which permit easy replacement from the front. This is a more complex arrangement than a wired-in-place indicator.

Incandescent indicators have one feature which is unique among data lights—they may be filtered to produce any color of display. In some data systems, this is a very practical consideration.

Glow Tubes

A very popular display device which combines some of the advantages of neon and incandescent indicators is the cathode-ray glow triode. This device is a miniature triode vacuum tube, about 1 inch long and 1/4 inch in diameter, with a phosphor-coated anode element. When driven to saturation, at about one milliampere, the anode emits a green light of moderate intensity, over an area of about 1/2 by 1/16 inch.

The glow triode can be used in relatively high speed applications. It requires moderately high (approximately +50 volts) supply voltage, and generates very little heat. When used in large quantities, the glow triode is conveniently and attractively mounted on a strip assembly behind a front panel.

Reliability of the glow triode is high, although not as good as the neon indicator. The main disadvantages of this type of indicator are the need for a low-voltage, medium-current (about 30 milliwatts per tube) filament supply and a somewhat higher cost than the simpler indicators. The cost disadvantage is balanced by the nearly zero-power control signal. The control grid draws microamperes of current in the "on" state, and may be operated directly by signal levels of 3 or 4 volts, without the need for amplifiers.

NUMERICAL DISPLAYS

For "quick-look" quantitative evaluation of data, many types of numeric and alphanumeric indicating devices are used. Depending on the nature of the requirements, each of the commonly used types has its own special advantages and limitations. One of the main considerations determining the selection of the method of display to be used in a system is the significance of the display in the overall system function. For example, in a count-down system used to coordinate a space-vehicle launch, the numerical displays are the *prime* output of the system. In such an applica-

tion, the use of a large, bright, highly legible readout may be justified since the display is the only essential reason for the existence of the system. On the other hand, if a numerical display is included in a computing system primarily for use as a servicing aid, and need not be legible at a great distance, a small, less elegant display may be the practical answer.

In general, the most commonly used numerical display devices are designed for presentation of a decimal readout, although certain types include provisions for special symbols in addition to the ten decimal characters. Multidigit displays are assembled in groups of one-digit modules. Each digit is driven by an associated decoder, amplifier or other suitable control element.

In a system which handles BCD data, the problem of generating control signals for a decimal display is relatively simple. One four-bit decoder-driver is required for each of the 10 states of a decimal digit. In other words, a single-digit presentation capable of displaying 0 through 9 is ordinarily backed up by 10 four-bit decoder gates and 10 output amplifiers. Although this appears to be very complicated and costly, development of simple and compact driver circuits has kept pace with the industry, especially in integrated circuits. BCD data ordinarily is fed in parallel format to a decoder/display, with each of the four-bit BCD characters controlling one display digit.

Where a system uses straight binary data throughout its internal processes, decimal readouts are seldom used, because of the complexity of making the translation from the binary to decimal code. In critical applications where the cost of such a conversion is justified, the binary is translated into BCD and displayed conventionally. In many applications, a compromise solution is used, in the form of an octal display. This method is relatively simple, yet offers *some* of the advantages inherent in a decimal readout. The octal code is very closely related to the straight binary code because of the pure relation of the base 8 to the base 2 $(8 = 2 \times 2 \times 2)$. A binary number can be translated into octal simply by separating the binary into groups, or characters, of three bits and evaluating the three-bit characters, starting with the three least significant bits. Consider the binary word

MSB 1 1 0 1 1 1 0 1 1 1 0 1 LSB

Separate the word into three-bit characters:

1 1 0 1 1 1 0 1 1 1 0 1

Evaluate the three-bit characters:

6 7 3 5

The octal equivalent of the binary word 1 1 0 1 1 1 0 1 1 1 0 1 is 6735, which can be displayed on a four-digit decimal-type readout. It may be apparent that, by the fundamental nature of the octal code, the highest value each digit can attain is 7, so that the 8 and 9 states of the decimal display devices need not be driven or energized in an octal application. When using an octal display or readout, one must be careful not to allow the numbers to be misread or confused with decimal numbers. In the example given, the octal number is "six-seven-three-five" and *not* "six-thousand seven-hundred thirty-five."

Electrically, decoding and driving an octal display from binary data is similar to driving a decimal display from BCD data. For each of the seven possible states of each digit, one three-bit gate and driver are required—a total of seven three-bit gates and seven drivers per digit. Standard BCD decoder-drivers are usually used, and the extra inputs and outputs are left unconnnected.

The most widely used numeric display devices may be grouped in several categories, listed in the approximate order of their present popularity:

1. Gas-discharge glow tube.
2. Projection display.
3. Bar segment.
4. Electroluminescent.
5. Edge-lighted.
6. Electromechanical.
7. Dot-matrix.
8. Cathode-ray.

The categories are not exclusive; that is, some display devices are combinations of more than one type—for example, an electroluminescent bar-segment display. Some of the most significant features of each of the important types of display will be described.

Gas-Discharge Devices

The oldest and most commonly used of all of the electronic numerical display devices is the shaped-cathode neon glow tube. Produced for many years by the Burroughs Corporation under the trade name NIXIE, these indicators are generally referred to as NIXIE-type displays, although they are now produced by other manufacturers. The NIXIE tube effectively consists of 10 neon bulbs inside a single glass envelope, with each of the wire cathodes shaped in the form of one decimal digit from 0 through 9. Only one of the 10 elements is energized at any given time, and the energized cathode (negative element) emits the characteristic red glow of a neon discharge. The glow surrounding the shaped

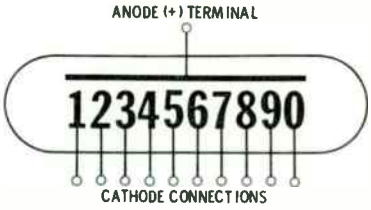
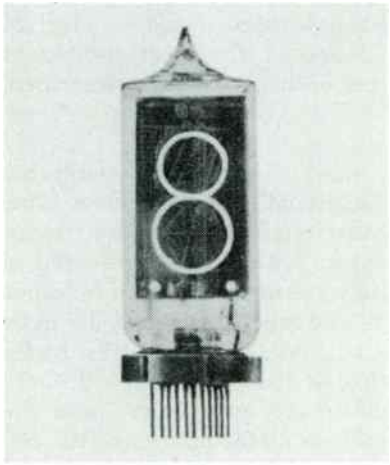


Fig. 5-1. Schematic layout of NIXIE tube.

cathode is visible through the glass faceplate of the tube. Schematically, the NIXIE tube is constructed as shown by Fig. 5-1. Each of the cathodes is connected to ground through a switch or saturated amplifier, and only one is permitted to conduct at any time. Fig. 5-2 is a photograph of a NIXIE tube.

NIXIE-type tubes are available in a wide range of sizes and shapes, from about 3/8- to 2-inch character height. Their reliability and life expectancy are extremely high, and in most electrical respects they are very similar to standard neon bulbs. Physically, the construction of a NIXIE-type indicator leads to a slight defect in display quality, since the 10 shaped-wire cathodes must be slightly separated fore and aft from one another. In an average size tube, the digits are in separate planes with about 1/2-inch overall spread. Consequently, as the numbers change, the digits appear to jump in and out over a fraction of an inch. This effect is not especially noticeable except when the tubes are viewed from an angle.

Among the operational features that NIXIE tubes share with ordinary neon indicators are low power dissipation and cool oper-



Courtesy Burroughs Corp.

Fig. 5-2. Photograph of NIXIE tube.

ation, high supply and driver voltage (about 200-volt supply and 40-volt drivers), a tendency to radiate electrical noise, and a limited color range. NIXIE tubes and NIXIE-type displays emit a light red color with a tinge of violet, which to some observers is slightly irritating. An amber, circular-polarized filter is often used with NIXIE tubes to reduce reflective glare and tone down the color to a very legible shade of red. The use of an integrated-circuit decoder-driver with NIXIE tubes simplifies the packaging problem, allowing the entire BCD-to-decimal circuit to be mounted on or very close to the NIXIE socket.

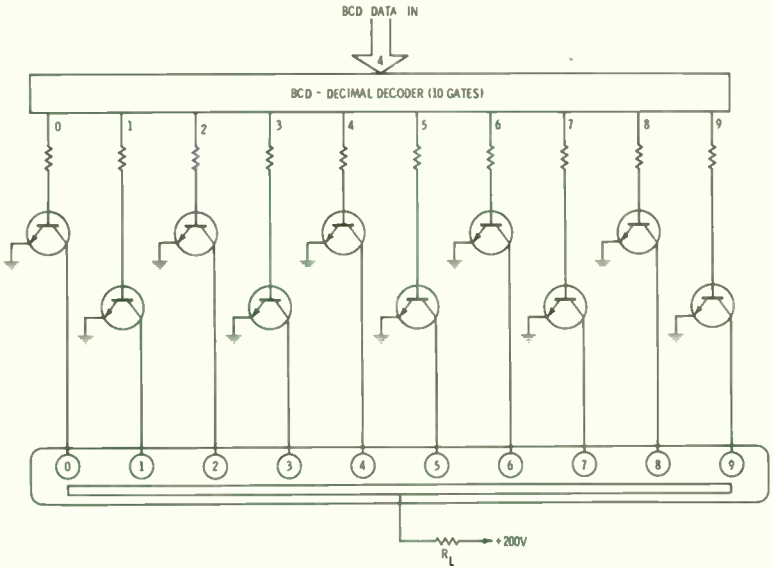


Fig. 5-3. NIXIE control circuit.

Fig. 5-3 shows the logic and circuitry required to drive one NIXIE character display. When implemented by IC's, the entire circuit, except for the NIXIE load resistor, is in one IC package.

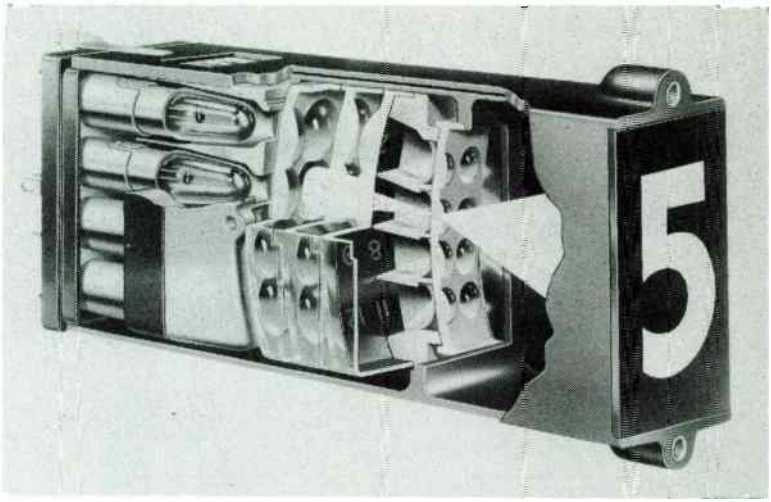
NIXIE-type indicators are ordinarily used for straight decimal numeric indication, although a few special symbols such as plus (+) and minus (—) are included in standard units. The manner in which NIXIE tubes are constructed does not lend itself to creation of special characters on a limited production basis.

Projection Displays

Running a close second to NIXIE tubes in the standard display field is the projection display, which includes an incandescent bulb and an optional projection system for each digit or symbol. Typical

projection units contain 12 independent channels, providing for the 10 decimal states and two special characters. All characters are projected onto a frosted glass faceplate which is part of the digit assembly. Fig. 5-4 shows a modern projection type display.

Projection displays are available in a range of character sizes from about 1/2- to 2-inches. Overall, a projection display is considerably larger and much heavier than a NIXIE display of comparable character size. Because of the anticipated maintenance problem of replacing burnt-out lamps, the design of a projection display is often complicated by the need for front-replaceable bulbs.



Courtesy Industrial Electronic Engineers, Inc.

Fig. 5-4. Projection display.

Considering the relatively large size of a projection display assembly, heat is not ordinarily a serious problem. The lamps may be driven by ac or dc, permitting the use of a variety of driver amplifiers, SCR's, or relays.

The construction of a typical projection unit lends itself to the display of special characters as well as numerals and letters. The shadow mask through which the light passes is usually made of plastic photographic film, so that any figure, photograph, or color filter which will transmit light may be used as one character in a display. An additional advantage is the fact that more than one of the channels may be energized simultaneously, if it is desirable to superimpose messages, characters, or colors. The optical arrangement demands that the bulbs have rigid and precisely located filaments, since a slight misplacement or vibration of the source of

light results in very noticeable defocusing and movement of the projected image.

A decimal projection display driven by BCD data requires a four-bit decoder and a high-current driver for each of the 10 (or more) states. Ordinarily, the decoder-driver units for a projection display are larger than those required for a NIXIE.

Bar-Segment Displays

All of the decimal characters, several letters, and a few special symbols can be generated by selectively energizing combinations of seven bars of light arranged as shown by Fig. 5-5.

If all bars are on, the character 8 is formed. If all except bar 5 are energized, the character 9 may be recognized, and so forth. Produced by several manufacturers in various detail forms, bar segment displays offer some advantages over the NIXIE and projection displays in applications requiring moderate flexibility. Bar-segment displays may be constructed in several ways, using incandescent or neon-lighted bars, direct or projected light, shaped cathode-glow, or electroluminescence. Fig. 5-6 shows multidigit bar-segment displays.

It should be evident that each state of the display requires the energizing of a unique combination of segments. This is done by the use of 10 encoders, one for each decimal character. Using the bar numbering system illustrated by Fig. 5-5, each of the 10 characters must be encoded as follows:

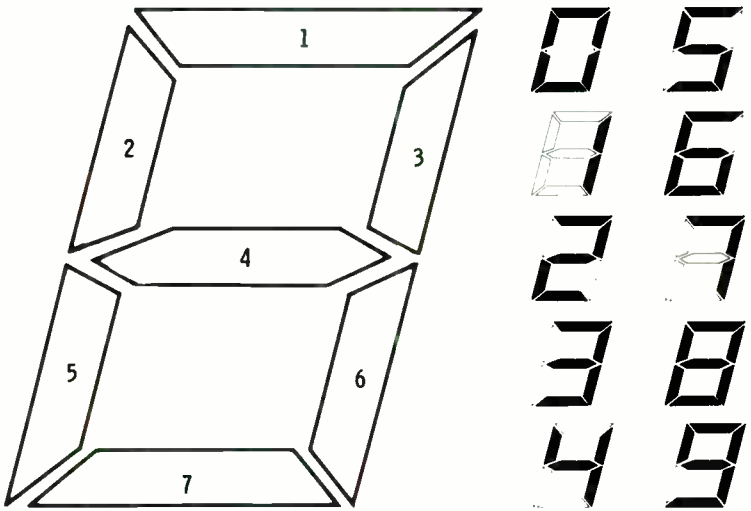
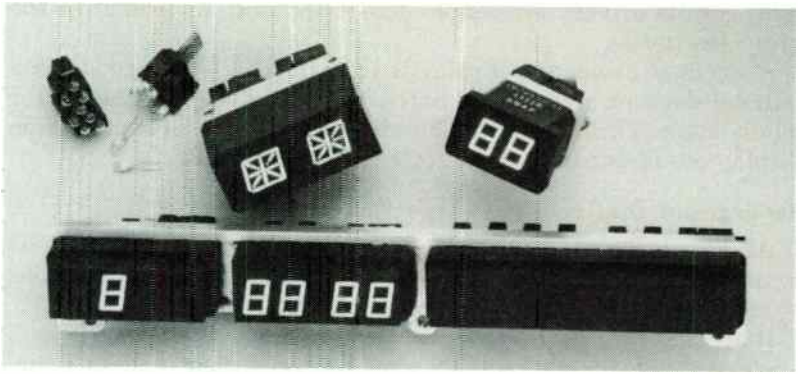


Fig. 5-5. Bar-segment character display.



Courtesy Tung-Sol Div., Wagner Electric Corp.

Fig. 5-6. Bar-segment displays.

State	*Energize Bars*
0	123567
1	36
2	13457
3	13467
4	2346
5	12467
6	124567
7	136
8	1234567
9	123467

If dc is used for the lights, encoding may be accomplished simply by a matrix of diodes, much like a decoder in reverse.

Fig. 5-7 shows a set of encoders for one digit. The use of an encoder such as shown by Fig. 5-7 presupposes that the input data is already in decimal form; that is, on ten input lines. If the data is in BCD, each four-bit character must first be decoded into 10 decimal lines, and the 10 lines must be used as the inputs to the encoder.

Fig. 5-8 illustrates the overall arrangement. In a practical decoder-encoder driver circuit, many of the internal terms and connections can be eliminated by combining the decoding and encoding functions.

The electrical characteristics of the bar light sources dictate the driver circuit required for a particular bar display and, to a great extent, govern the physical characteristics of the display. Whether neon, incandescent, or electroluminescent light sources are used with a bar display, the controlling logic is the same.

For display of decimal numbers only, the seven-segment character is adequate, and provides an easily recognized symbol for each

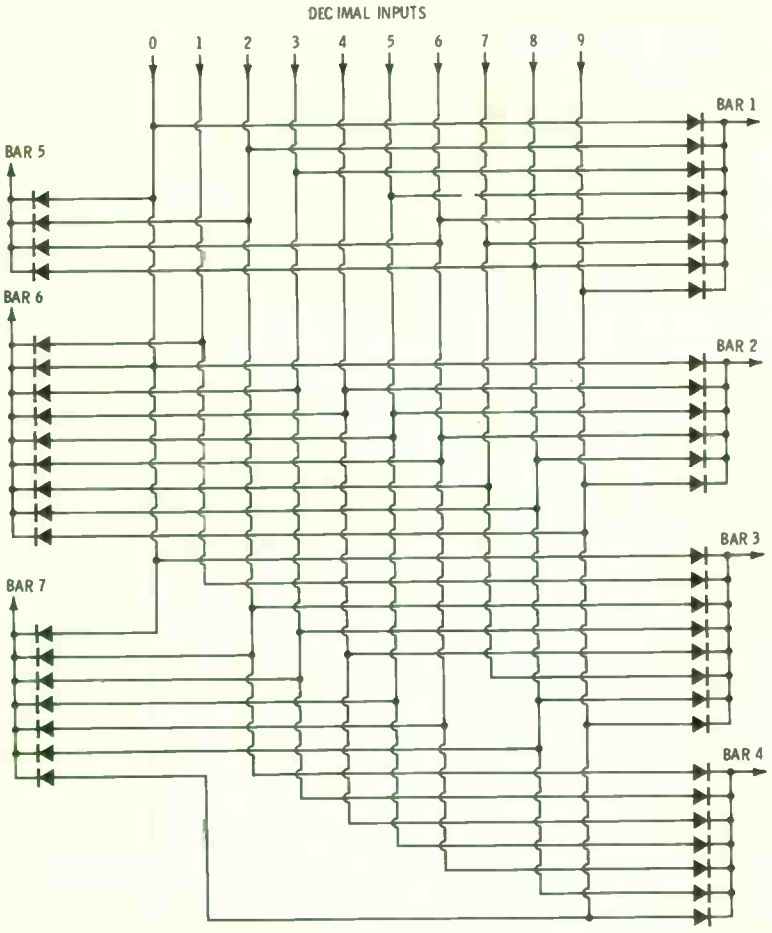


Fig. 5-7. Decimal-to-bar-segment decoder matrix.

digit. When alphabetic characters or other symbols must be formed, units with more segments are used. Standard bar segment devices with as many as 16 segments per character are commonly used.

Electroluminescent Displays

A relative newcomer in the character display market is the electroluminescent (EL), or dielectric-glow, device. The active element in an EL display is a capacitor formed of two transparent conductive films separated by a phosphor-impregnated dielectric plate. The entire assembly is typically less than 0.01 inch in thick-

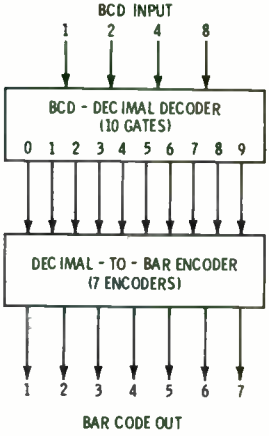


Fig. 5-8. BCD-to-bar-segment translation.

ness, and may be fabricated in virtually any size and shape. Ordinarily, for mechanical strength, the film sandwich is bonded to a glass or plastic substrate, with an overall assembly thickness of about 1/4 inch.

When an ac voltage is impressed across the EL capacitor, the phosphor dielectric layer glows visibly. The intensity of the glow is proportional to both the amplitude and the frequency of the applied ac, with optimum practical values in the neighborhood of 250 volts rms and 400 Hz.

Although the voltage required for EL display is high, a typical 2-inch EL bar segment character draws less than 50 milliwatts of power with all bars energized, and produces very little heat. The EL display is mechanically the most rugged of all existing devices, and has practically unlimited life expectancy.

Electroluminescent devices may be produced in several phosphor colors; the brightest and most commonly used is pale green.

Since EL displays can be fabricated in nearly any form, either by controlling the shape of the plates or by the use of a stencil-mask, they may be used for various special symbol or message displays. The most common application of EL is in the form of bar segment characters with one common plate and several individually switched elements.

In spite of all of the extremely attractive physical characteristics of EL displays, applications are somewhat limited by the difficulty of logically switching high-voltage ac. In a system in which relays or stepping switches can be used as the display controlling elements, EL displays are practical and economical. Without the use of relays, however, EL driver and control circuits are prohibitively complex.

Edge-Lighted Displays

Using the peculiar light-transmitting properties of Lucite or similar material, each digit of an edge-lighted display consists of a stack of 10 transparent wafers, with one character etched or engraved on each wafer. A character is displayed by energizing a lamp which illuminates the edge of only one selected wafer, causing the engraved character to glow. Although the legibility and clarity of an edge-lighted display are excellent, this type of display device is somewhat cumbersome.

Since incandescent bulbs, one for each character or symbol, are ordinarily used in edge-lighted displays, the decoding and driving requirements are similar to those of projection display units.

Dot-Matrix Displays

In the dot-matrix character display, a character is formed by a group of individual light dots in much the same manner as the letters in a movie theater marquee or football scoreboard presentation (Fig. 5-9). Each character is generated by a matrix of individual lamps consisting of enough lamps, say 6×7, to form legible numbers or letters when the appropriate lamps are energized. The coding, or selection of the proper lamps for each character, is accomplished through a diode or relay encoder as for the bar-segment display. The light sources for the elements of a dot-matrix display can be small incandescent bulbs, neon bulbs, or the more recently developed light-emitting diodes.

CRT Character Displays

A cathode-ray character display is a miniature cathode-ray tube with ten separate control grids, each grid controlling the electron stream from one common cathode (Fig. 5-10). Each grid directs the stream through a shaped aperture in a target electrode, producing the selected display on the front screen. The target

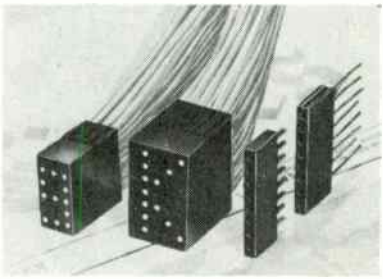


Fig. 5-9. Dot-matrix character displays.

Courtesy Pinlites, Inc.

Courtesy Industrial Electronic Engineers, Inc.

Fig. 5-10. CRT character display.

apertures can be designed to produce any desired shape of character, number, or pictogram, but are ordinarily produced only in the form of alphabet or numeric characters.

The displays produced by CRT devices have all of the characteristics of an oscilloscope display, and requires an extremely high voltage supply, typically about 2000 volts dc, in addition to the

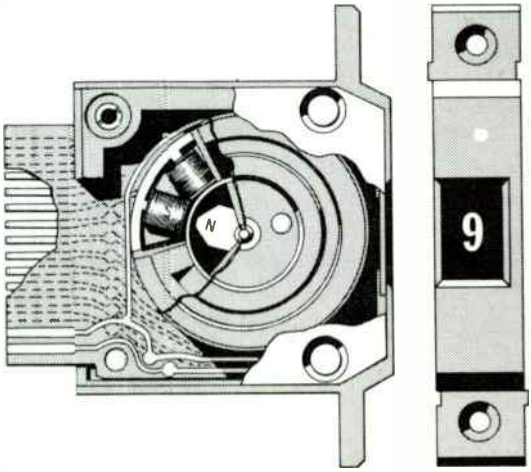


Fig. 5-11. Electromechanical display.

control and filament voltages. Manufacturers of CRT displays, however, provide compact power supplies designed specifically for their displays at a very reasonable price and in conveniently small packages.

Electromechanical Displays

A very straightforward, although somewhat slow and bulky, display is the electromechanical type, in which a belt or wheel imprinted with characters is selectively positioned before a viewing window (Fig. 5-11). Each coded digit fed to the display device actuates a simple servomechanism which controls the motion and position of the character wheel. Other forms of electromechanical character displays use a group of character flags which are electromagnetically selected by solenoids driven by the input data.

6

DATA CONVERSION

In almost every large data processing system, there are areas in which the basic format or language of the data must be translated without altering the value of the data. This requirement arises most commonly in the input and output interfaces of a system, where the system must accept the incoming data in whatever form is available and deliver an output as demanded by the user. When the original data input is already in the form of a digital signal, the problem of converting to a standard format is relatively simple. However, if a digital system is initially provided with analog or synchro information, special and sometimes complicated means must be used to accurately and faithfully translate the data into a digital representation.

Varied techniques are used to perform digital-to-analog and analog-to-digital conversion. Several of the most straightforward and most commonly encountered methods will be discussed.

DIGITAL-TO-ANALOG CONVERSION

A simple method of directly converting a parallel digital word into an analog voltage makes use of a "weighted-current" summation. By this technique, each data bit controls a precise current component in a resistive summing network. With each current component scaled exactly in proportion to the data value of its control bit, the total current or voltage summation is proportional to the value of the data word. Fig. 6-1 illustrates a simple digital-to-analog conversion of this type.

In the example, the total current indicated by the milliammeter is an exact measure of the value of the binary input word. If all

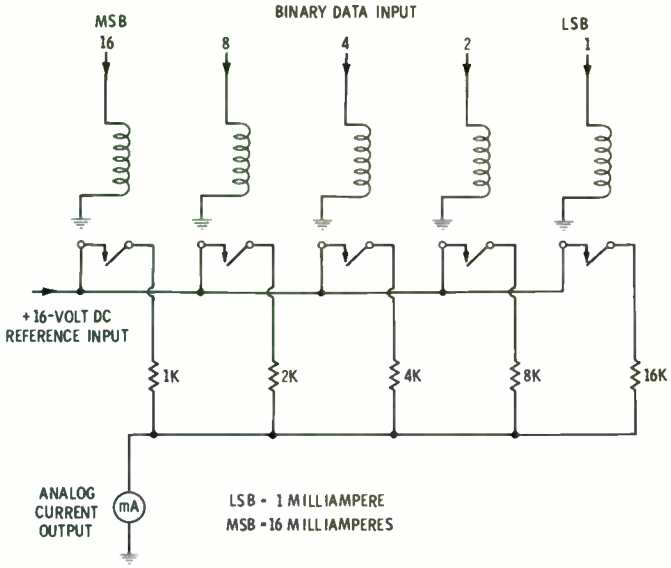


Fig. 6-1. Simple digital-to-analog converter.

bits are 0, the indicated current is zero, since all the relays are open. If only the LSB is 1, a current of one milliampere will flow, as determined by the 16-volt reference and the 16-kilohm LSB input resistor. If the next higher bit is energized and all other bits are 0, the current is 16 V/8 kΩ, or 2 milliamperes, and so on. This proportionality is equally true when combinations of bits are energized; for example, the LSB and the MSB together will yield a total current of 17 milliamperes. The convenient scale factor of one milliampere per bit is used simply for illustrative purposes, and is governed by the values of the resistors and the reference voltage level. It must be understood, however, that the relative scaling of the resistor values must be binary; that is, each resistor in the network is equal to exactly twice the value of the adjacent resistor for the more significant bit.

The overall accuracy of this simple digital-to-analog converter is determined entirely by the accuracy and stability of the reference voltage and the resistor values.

Although the example illustrates a conversion from straight binary data to an analog current, a similar circuit can be used for conversion of BCD or, for that matter, any binary code to analog. The BCD-to-analog current converter shown by Fig. 6-2 operates in exactly the same manner as the binary converter, except for the resistance scaling factors. Here, the resistors are chosen with

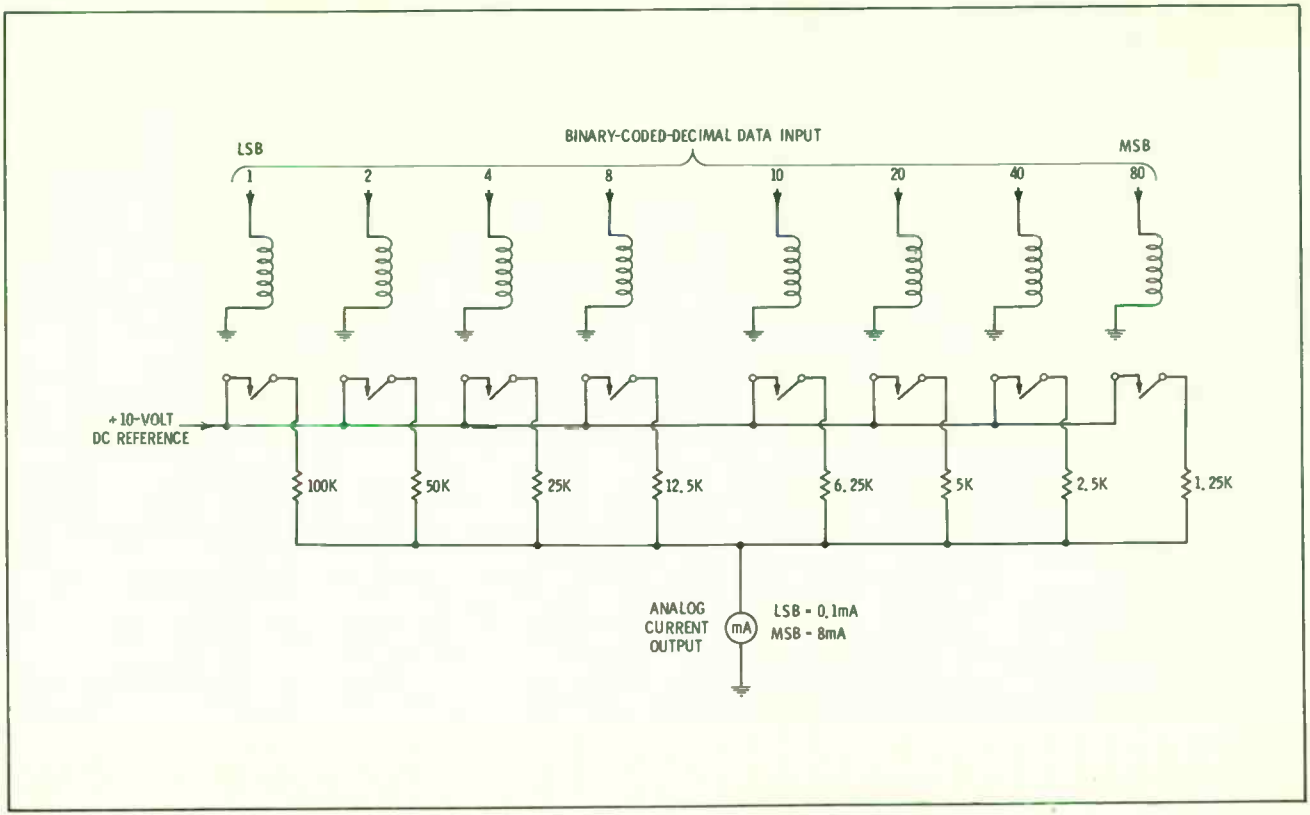


Fig. 6-2. BCD digital-to-analog converter.

values in the ratio of 1, 2, 4, 8, and 10, 20, 40, 80, corresponding to the values of the BCD data bits.

In most systems, an analog voltage, rather than current, is required as the output of a conversion network. This imposes a special constraint on the network design. Although it might appear that the milliammeter in the circuit of Fig. 6-1 could simply be replaced by a resistor to develop a voltage output, this would produce a serious conversion error. A basic requirement of the current summation network is that each input current component must be independent of all others. Inclusion of a common resistance from the summation point to ground would have a different degree of effect on each of the inputs, tending to depress the effect of the high-order bits. The result of this would be a nonlinearity of output vs input roughly proportional to the value of the common resistor.

The problem of producing an output voltage from the current summation can be solved in several ways. One method, shown by Fig. 6-3, develops the output voltage across a resistance which is equal to all of the "off" input resistors in parallel, through the normally closed contacts of the data relays. This method is mathematically correct, and is quite commonly used in spite of the requirement for double-pole switching relays and the increased number of precision resistors.

The easiest solution to the problem of developing an output voltage is the use of a very small resistance from the summing point to ground. If the common resistance is extremely small com-

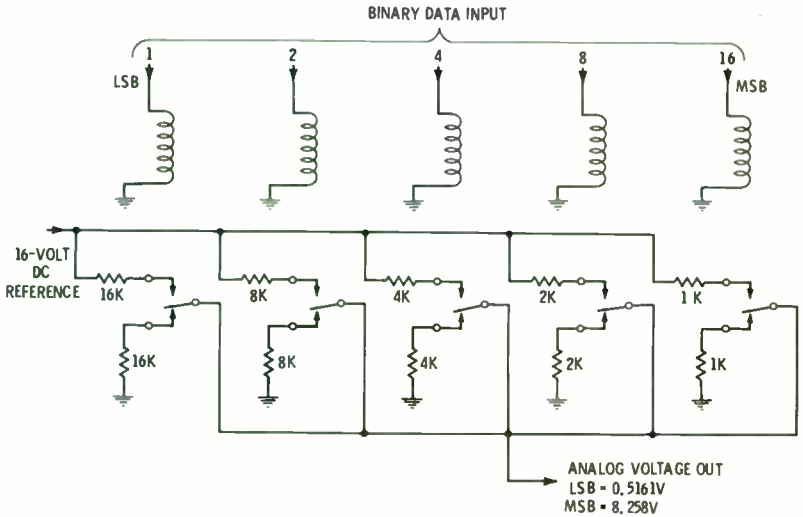


Fig. 6-3. Digital-to-analog converter: voltage out.

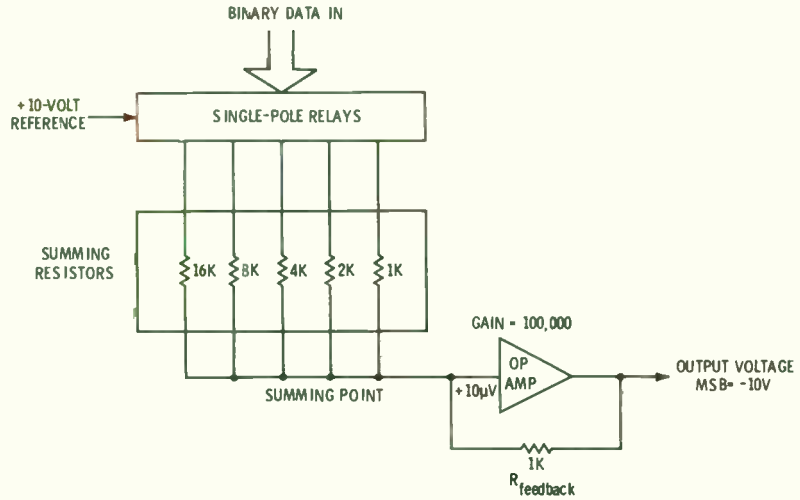


Fig. 6-4. Digital-to-analog converter using operational amplifier.

pared to the summing resistors, the error caused by the common resistor can be kept within tolerable limits. As a result of the very low output voltage developed in this manner, the network must be followed by a stable, high-gain voltage amplifier.

Most digital-to-analog conversion circuits include an operational amplifier connected as shown in Fig. 6-4. Although the summing point does not appear to be returned to ground through a low resistance, the summing point effectively looks into nearly zero impedance because of the effect of the feedback signal. This may be illustrated by a specific example.

Let us assume that the amplifier has a voltage gain of 100,000 and that the dc output of the amplifier is —10 volts as determined by the input data word. To maintain a —10-volt output level requires only 10/100,000 or 100 microvolts across the amplifier input terminals. Consequently, the summing network output voltage is held at a level of 100 microvolts, which is very nearly the same as short-circuiting the summing point to ground.

The use of an operational amplifier in the conversion network provides a great degree of flexibility of design and application. Operational amplifiers may be connected to yield polarity inversion as shown, or used in a noninverting configuration. By selecting or adjusting the value of the feedback resistor, the precise scaling of the converter may be selected or trimmed. In certain applications, it is desirable to produce an output which varies both positive and negative about a zero center. This is easily accomplished, using a "sign" input bit to control the output polarity.

The circuit of Fig. 6-4, exactly as illustrated, is a practical and commonly used configuration. Without undue design effort, a circuit of this type can be used in applications requiring accuracy on the order of 0.01 percent. In controlling the overall accuracy of this converter, three main sources of error must be considered: the resistance network, the relay contact quality, and the reference voltage accuracy. Despite the fact that the operational amplifier plays a critical part in the circuit function, the error contributed by a run-of-the-mill operational amplifier of modest cost is usually insignificant.

The great majority of digital-to-analog conversions produce a dc analog output voltage. Naturally, the output voltage range and polarity are determined by the requirements of the load. There are certain applications, often in the case of a digital system feeding inputs to a servo system, in which it is convenient to produce an ac analog signal. All of the circuits and techniques described can be applied to an ac system simply by the use of an ac reference voltage. Provided that the reference frequency is low enough not to create problems related to pickup, cross talk, and stray capacitance, operation and analysis of an ac converter are identical with those of the dc circuits.

R/2R Ladder Network

One of the advantages of the weighted binary resistor network is the simplicity of the required switching elements. The nature of the current summation is such that only single-pole switches are necessary. In a conversion circuit of high precision, however, certain disadvantages of the binary resistor network become apparent. The major drawback is the necessary range of resistor values in the network. A 14-bit binary conversion, for example, requires a network in which the highest resistor value is 8192 times the value of the lowest. If we assume a value of 1000 ohms for the lowest, this dictates a value in excess of 8 megohms for the highest. Not only is this resistance value high, the external circuit at such an impedance level must be absolutely clean, and very special precautions must be observed to guard against the shunting effect of moisture. From the resistor-network manufacturer's standpoint, the problem of fabricating an assembly of resistors ranging from 1 kilohm to 8 megohms with identical stability characteristics is extremely difficult. In short, a precision network of binary scaled resistance values presents three major problems: the lowest value is limited by excessive current, the highest value is limited by external shunt leakage effects, and the high/low ratio is somewhat limited by stability considerations. These drawbacks have led to the popular use of the R/2R ladder network,

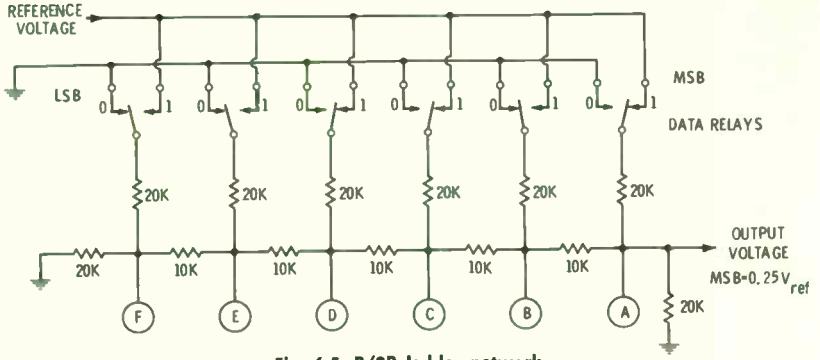


Fig. 6-5. R/2R ladder network.

in which a multibit network using only two resistor values is fabricated. Fig. 6-5 illustrates the configuration and application of this network.

The network, as illustrated, consists of pairs of 10- and 20-kilohm resistors. The switching relays are double-throw, connecting each input either to ground or to the reference voltage in accordance with the input data word.

To analyze the operation of the network, assume that the MSB is 1 and that all other inputs are 0. The voltage at the output, point A, is the result of V_{ref} and the divider action of the network. The upper leg of the divider is the MSB 20-kilohm input resistor; the lower leg value may be computed starting at the left end. The resistance at point F, looking left, is 20 kilohms in parallel with 20 kilohms, or 10 kilohms. This makes the resistance at point E, looking left, 10 kilohms plus 10 kilohms, or 20 kilohms, in parallel with 20 kilohms to the switch, or 10 kilohms. The resistance looking left from point D is 10 kilohms plus the 10 kilohms computed for point E in parallel with 20 kilohms to the switch, or 10 kilohms. At each point, it may be seen that the resistance looking left (with all zeros) is 10 kilohms to ground. Accordingly, the voltage at point A is

$$\frac{V_{ref} \times (20\ \text{k}\Omega \text{ in parallel with } 10\ \text{k}\Omega)}{20\ \text{k}\Omega + (20\ \text{k}\Omega \text{ in parallel with } 10\ \text{k}\Omega)}$$

or

$$\frac{6.666}{26.666} \times V_{ref} = 0.25V_{ref}$$

To illustrate the weighting of the effect of lower-order bits, consider the case in which only the LSB is 1. In this situation, the resistance at points B, C, D, E, and F, looking right, is 10 kilohms.

The voltage at point F is the result of V_{ref} through 20 kilohms to the parallel combination of 10 kilohms and 20 kilohms, or 6.666 kilohms to ground. This produces a voltage of $0.25V_{ref}$ at point F. The voltage at point E is exactly one-half of the voltage at point F, and so on toward the right. Thus, the output voltage for the LSB only is $0.5 \times 0.5 \times 0.5 \times 0.5 \times 0.5 \times 0.25V_{ref}$, or $0.25V_{ref}/32$, indicating that the binary weight of the LSB is 1/32 of the MSB.

Among the many desirable features of the R/2R network is the obvious advantage of requiring only two resistance values if a network is to be made up from discrete components. Any convenient and practical value of R may be used, and, as in the case of the binary resistance network, only the ratio, rather than the absolute value, of R is critical. Where extreme precision is required, the R/2R offers the additional advantage of the resistance values being so close that identical materials and processes can be used in the manufacture of both the R and 2R resistors. Manufactured under identical conditions, the resistors are very likely to have identical or similar temperature and aging characteristics, and tend to maintain much better network accuracy than resistors of widely separated values.

Values in the range of 5 kilohms to about 50 kilohms are usually used in R/2R networks, making the circuits relatively insensitive to the effects of series contact resistance and the shunting effect of moisture and contamination.

The main disadvantage of the R/2R network is the requirement for double-throw input switching. In low-speed digital-to-analog conversion, where relays may be used as the switching elements, the double-throw contact requirement is no hardship. However, in applications where solid-state switches are necessary, the switching circuits become somewhat more complicated by the R/2R network.

Switching Techniques

The most straightforward method of switching components of voltage or current into a ladder network is by way of relays. As illustrated by the foregoing examples, one relay coil is controlled by each data bit and single- or double-pole contacts are used to control the analog summation.

Because of their excellent characteristics of low series resistance and high leakage (open) resistance, good-quality relays may be used in digital-to-analog converters of 0.01 percent accuracy or better. Magnetic-reed relays with mercury-wetted contacts are generally considered to offer the best combination of contact quality, speed of operation, and reliability. Reed relays with a life expectancy of 100 million contact cycles or more are available.

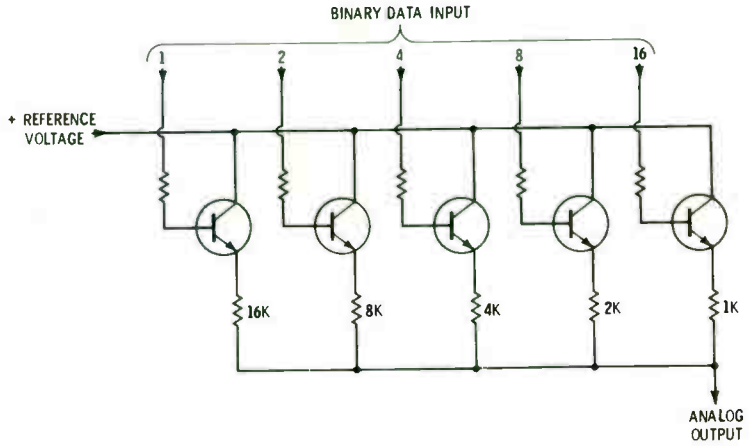


Fig. 6-6. Single-pole transistor switching.

The maximum usable switching rate of conventional reed relays is about 300 hertz. Conversion applications requiring higher rates use various types of solid-state switching devices which are not limited by mechanical parts. With careful and sometimes complex circuit design, the accuracy of a solid-state switching circuit can be made to approach that of a high-quality relay, and may be operated at conversion rates in the megahertz.

The simplest electronic data switching technique consists of transistors connected in series with each of the inputs to a binary scaled ladder network, as shown by Fig. 6-6. This configuration, or other similar single-pole switching circuits, provides a good solution to a high-speed, low-accuracy conversion problem. The accuracy limitation is the result of the large dynamic range of input resistors when more than six or eight bits are to be controlled. The series transistors, instead of acting as a pure short circuit or open circuit when turned on and off, exhibit a measurable forward series resistance and slight shunt leakage. These effects can be compensated to some extent, but may become unmanageable where accuracy of one percent or better is required.

High-speed, high-precision conversion circuits ordinarily employ the R/2R resistor ladder with a pair of transistors connected as a double-pole switch in each input leg as shown by Fig. 6-7. The control voltages to the switching transistors in Fig. 6-7 must swing from a voltage greater than V_{ref} for a "1" to a slightly negative voltage for "0." This ensures that, in either condition, one of the transistors in each pair is in hard saturation and the other is absolutely cut off. Using the R/2R network, the switch circuit design can be optimized to one specific impedance and current

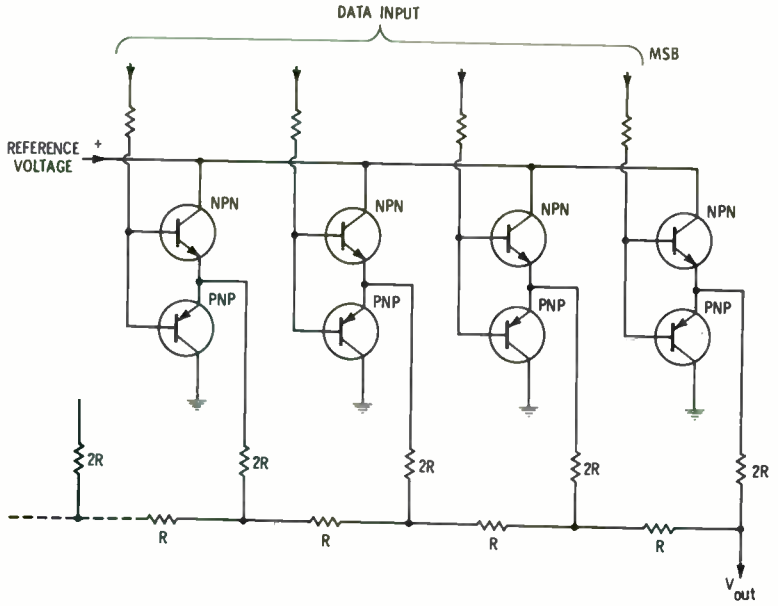


Fig. 6-7. Double-pole transistor switching.

value. Although the collector-to-emitter saturation voltage has a direct effect on the overall accuracy of this circuit, this error can usually be compensated. As a rule, the limiting factor on the performance of the circuit of Fig. 6-7 is not the absolute quality of the individual switching circuits, but the closeness to which all of the switches are matched and the stability of the circuits with age and temperature.

High-quality conversion networks often use field-effect transistors (FET) as the switching elements, making use of the extremely high "off" resistance of the FET. In this application, the FET operates much the same as a high-gain conventional transistor; however, when cut off, the input control circuit and output circuit exhibit impedances up to hundreds of megohms. When turned on, FET's present a forward resistance slightly higher than conventional transistors.

ANALOG-TO-DIGITAL CONVERSION

As a result of a century or more of development of electrical technology in the analog direction, most of our instrumentation, or measuring equipment, operates on an analog principle. Typically, devices for measuring temperature, force, pressure, or other

physical quantities deliver an output in the form of a variable voltage or current, scaled in relation to the measured quantity. When data such as is gathered by analog sensing devices must be processed in a digital computer, the first step is to convert the analog information to a digital form.

Sometimes the conversion of analog data is performed simultaneously at each of the inputs to a digital system, in which case several converters are used. In other systems, for reasons of economy or simplicity, a single converter is used, sampling each of the inputs in time sequence and delivering the converted data in the same sequence to digital storage or into the data processor. The use of a single time-shared analog-to-digital converter is known as *time multiplexing*.

The technique ordinarily used in analog-to-digital conversion makes use of a digital-to-analog converter in a feedback-comparison loop. Fig. 6-8 shows how this may be accomplished in its most basic form.

The digital output word is controlled by a counter which is driven by a local clock. Controlling the flow of clock pulses into the counter is a pulse gate which is controlled by a comparator. The comparator, a very high gain saturating amplifier, compares two dc voltages. One is the input dc analog signal; the other is a dc analog representation of the output digital word. When the two are equal, the comparator produces a logical 0 output, and the pulse gate blocks the clock signal. When the input analog signal is greater than the converted output word, the comparator output goes to logical 1, opening the pulse gate to permit the flow of clock pulses into the counter. When the converter counter output word is equal to the input signal, the comparator again closes the pulse gate.

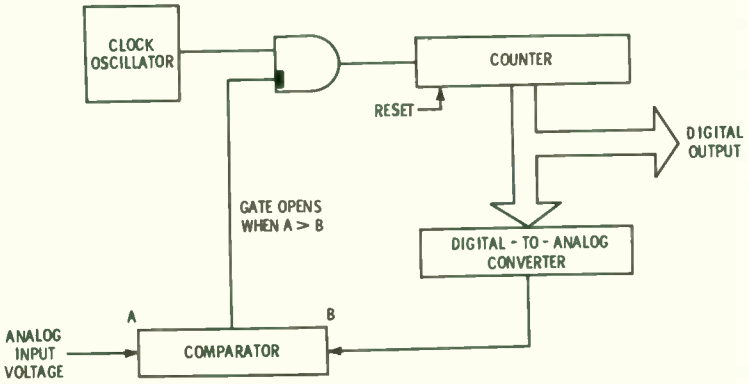


Fig. 6-8. Analog-to-digital conversion loop.

The simplified circuit of Fig. 6-8 is intended only to illustrate the basic counter/comparator method of analog-to-digital conversion. With a few minor changes, however, the circuit can be used in a practical application. As shown, the counter operates in only one direction, and if the output word needs to be diminished, the counter must go through an entire cycle to satisfy the comparator. For example, in a converter designed for 10 bits, or a maximum reading of 1023, if the output is required to change from a count of 300 to 299, the counter must accumulate from 300 all the way to 1023, recycle to 0000, and count up to 299. This problem can be avoided by the use of an up/down counter and two comparator circuits. The dual comparator produces a three-level output indicating if the count is too low, too high, or correct. The output of the comparator in this case not only controls the flow of clock pulses, but also controls the direction in which the counter accumulates. A circuit of this type can maintain a *continuously* correct relationship between the output digital word and the input analog.

Other techniques of comparison and correction of the output digital word are often used. In a very popular commercial circuit, the digital output word is sampled one bit at a time and is compared with the analog input. Starting with the output register reset, a comparison is made to determine if the input is greater than the MSB. If so, the MSB is set to 1. If not, the MSB remains 0, and the system steps to the next bit, repeating until all of the output bits have been processed. The advantage of this method is speed of operation. If the output is a 15-bit digital word, for example, exactly 15 steps at the system clock frequency are required to perform the conversion. The only disadvantage of this method is the relatively complex programmer and control circuit.

The clock frequency and the method of controlling the output register or counter determine the maximum input rate of change which can be faithfully followed by an analog-to-digital converter. Using the up/down counter and a 1-megahertz clock would permit a maximum rate of change equivalent to about one bit per microsecond. The unidirectional counter with a 1-megahertz clock can follow a one bit per microsecond change in the increasing direction, but requires much more time to follow a slight downward change in input.

To standardize the conversion time, and also to avoid the effects caused by the input changing while the conversion cycle is in process, most practical analog-to-digital converters include a sampling circuit at the input. Upon command from an external source, such as the digital data processor which receives the output, a very narrow pulse is generated which gates the input level

to a holding circuit, maintaining the sampled level for as long as is required for the conversion. This technique, known as *sample and hold,* is especially useful in time-shared conversion systems where a significant error might be caused by sequentially converting several fast-changing inputs. With sample and hold circuits, all inputs may be sampled simultaneously during an extremely short time slot, or aperture, and may be converted in sequence after the samples have been taken.

In a multichannel system, the sample and hold circuit usually consists of a set of identical input circuits, one for each channel, gated sequentially to a common input line into the analog-to-digital converter. In each input circuit, the storage or holding component is a high-quality capacitor driven by a low-impedance gated amplifier. During the sampling aperture, the amplifier is turned on, charging the capacitor to the input level. At the end of the time aperture, the charging amplifier is gated out of the cir-

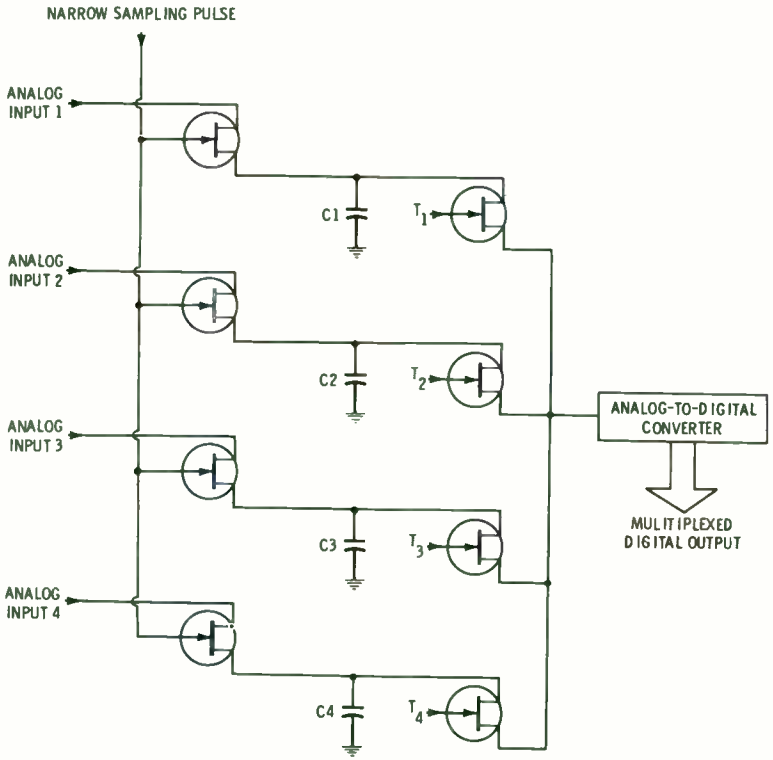


Fig. 6-9. Four-channel sample-and-hold multiplexed analog-to-digital conversion, simplified.

cuit. While gated off, the amplifier must present an extremely high impedance to the capacitor so that the charge does not decay through the shunt leakage. Because of the need for very high "off" resistance, field-effect transistors are commonly used in the amplifier and gate circuits. A four-channel sample and hold circuit is shown in Fig. 6-9. The main sampling pulse saturates all input transistors, charging each of the holding capacitors to the instantaneous voltage of its input line. After the sampling pulse, all capacitors retain their voltage, since the input and output transistors are cut off, presenting a very high shunt impedance. When the conversion is to be performed, each channel in turn is connected to the converter input by a control signal. The input stage of the converter must have high enough input impedance to avoid discharging the capacitor during the conversion time.

NUMERICAL CONVERSIONS

Because of its natural adaptability to hardware and circuits, straight binary code is used in most data processing circuits. In general applications, data is displayed or monitored by binary indicators such as banks of panel lamps. Unless it is necessary for an operator to make extremely fast observations and evaluation of displayed data, binary data displays are an adequate and simple solution. With data words more than four or five bits in length, however, evaluation of a data word is usually a pencil-and-paper operation except for a very experienced operator.

In systems requiring quick-look data display, one of three methods is ordinarily used to present the data in an instantly readable form:

1. The overall data processing system can be designed to function using a BCD or decimal code, permitting the use of a relatively simple decimal data display.
2. The system may be designed to function with binary data and use an octal display, which is as easily read as decimal.
3. The output of a binary system may be converted into its decimal equivalent and displayed.

Generally, the first solution can be applied to a small-scale system in which the somewhat greater complexity of BCD data manipulation is offset by the reduced complexity of the display system.

The second method, using an octal display, is quite satisfactory except for the ever-present chance for human error. Outwardly, an octal display appears identical to a decimal display and may be misinterpreted. For example, a bank of NIXIE indicators display-

ing a radar range of 12,750 yards would represent 12,750 yards, if decimal, but only 5608 yards if octal.

As described earlier, the circuit hardware required for displaying a BCD or octal character on a numerical indicator is very simple, consisting only of a set of decoder gates plus the necessary amplifiers. Preparing a multibit binary word for decimal readout, however, can be complicated.

The simplest method of making a data conversion between binary and decimal codes involves two counters and a gate as shown by Fig. 6-10. On command, the binary input word is transferred into a binary downward counter, so that the input data is the starting condition of the counter. Simultaneous with the input transfer, the gate control flip-flop is set, opening the clock control gate. The clock, at any convenient frequency, causes the counter word to decrease until the counter reaches the 0000 condition. This condition is detected by a decoder or other means, and resets the control flip-flop, turning off the clock gate.

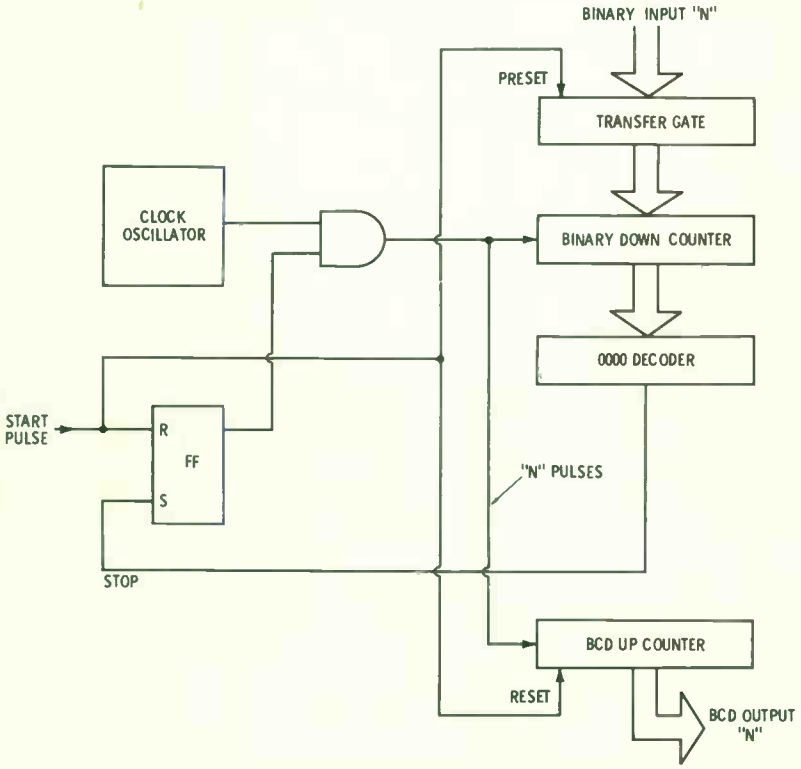


Fig. 6-10. Binary-to-BCD conversion.

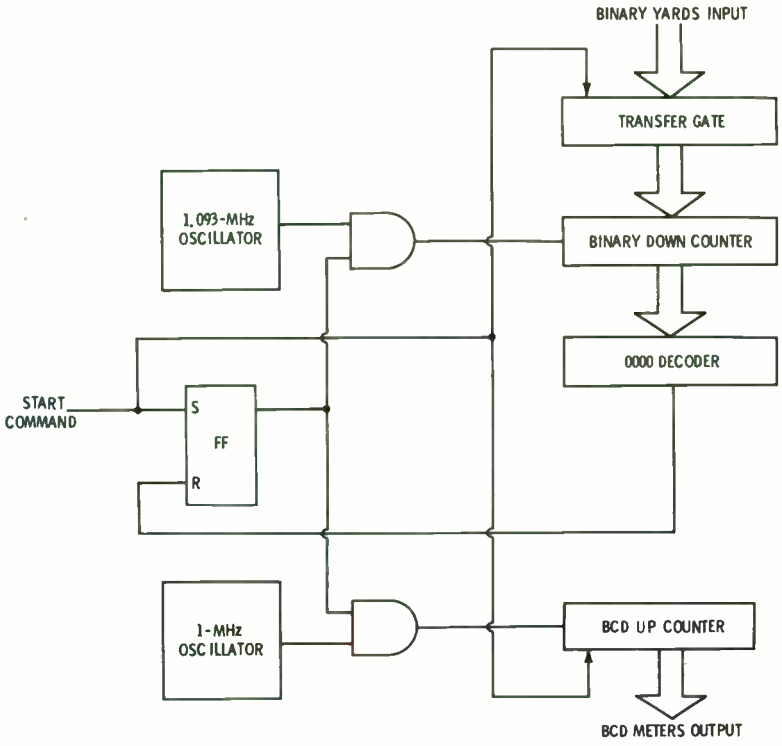


Fig. 6-11. Dual conversion: binary yards to BCD meters.

During the interval in which the clock gate was open, a number of pulses exactly equal to the input binary word N passed through the gate. This is evident since the binary counter counted down from condition N to zero. During the same interval, the gated train of N pulses was fed to the output BCD counter which started from a reset condition. As a result, the final condition of the BCD counter is the value of N.

This general configuration can be applied to a conversion between any two codes or languages for which a counter can be built, and produces no conversion error at all.

An interesting innovation to the dual counter conversion is shown by Fig. 6-11. In this example, the input data represents binary yards, while the output is expressed in decimal meters. The binary-to-BCD conversion is performed as described before; however, two separate oscillators are used for the counters. The frequencies of the oscillators are chosen in accordance with the ratio of the units of measurement. In the illustrated case, since the ratio of one meter to one yard is 1.093 to 1, the oscillator fre-

quencies are scaled in exactly the same ratio. This conversion is subject to a small error, since there is no fixed relation between the two oscillators. The maximum digital error is one bit at the output plus the analog error which is caused by the frequency inaccuracy of the two oscillators. By the use of higher clock frequencies and more bits than necessary, the digital error can be reduced to whatever value may be tolerated, and the analog error can be controlled by the use of high-quality crystal oscillators if high precision is required.

Conversion by the use of two counters is limited in speed by the maximum allowable clock frequency and the number of bits in the words to be processed. With a 1-MHz clock, a 10-bit conversion would require up to 1024 microseconds, depending on the value of the word to be converted. In some applications, this speed limitation may be prohibitive, and other methods must be employed.

There are various logical ways in which conversion speed can be increased, usually involving a compromise between the count-up/count-down method and a straight arithmetic solution.

Conversion from binary to decimal, or vice versa, using arithmetic logic can be performed exactly as manual computation, although several very clever short-cut methods have been devised for the process. Before integrated circuits were commercially available and practical, the pure arithmetic solution was seldom used; however, the tremendous logical capacity and the low cost of modern IC's has made the arithmetic solution a practical and reasonably simple method of conversion. Using a dc gating structure comprised primarily of full adders, a conversion can be performed in a microsecond or less.

7

SUBSYSTEMS

It is difficult to define precisely a data system, subsystem, or logical block. A variable delay generator, for instance, can be a part of a data system, or if used independently it *is* a data system of sorts. Several of the blocks described in Chapter 4 could be included in this category.

INTRODUCTION

Some component parts of large data systems are so independently complex and relatively self-sufficient that they should be considered as subsystems, or systems within an interconnection of other subsystems. As a practical matter, large systems are often made up of several subunits, each designed and fabricated by a manufacturer who specializes in one particular branch of the industry. This specialization is very apparent in scientific data systems in which a room full of hardware may represent the coordinated effort of specialists in recorders, input-output devices, signal conditioners, and general-purpose computers. A well-planned design program of this nature is usually coordinated by a systems engineer, through whose effort and talent the performance characteristics of each of the subsystems are specified, leading hopefully to an overall system whose components are harmonious and efficiently matched.

In this chapter, the intention is to whet the reader's appetite with not much more than a glance at several of the functional subsystems found in modern data processors. Beyond this, the reader is invited to refer to the ample volume of reference ma-

terial available on the bookshelves and from the customer service departments of most digital equipment manufacturers.

DATA TRANSMISSION SYSTEMS

A very important part of the data processing industry is devoted to the problems of transmitting data over a long distance. In a sense, every communication system may be considered to be a data transmission system, since all communication consists of conveying intelligence, or data.

Of special interest to the data systems engineer is the handling of digital information, since practically all long-distance data transmission links operate with digital signals. The reason for this is quite evident. To transmit analog instrumentation data requires a transmission medium which is as precise, accurate, and stable as the data itself. In the very simplest system, in which the signal amplitude represents an analog quantity, a 5-percent change in transmission characteristics because of line losses or atmospheric changes would introduce a corresponding 5-percent error in the received signal. If a digital transmission system is used, and the input data is first converted to binary words of whatever length is required for accuracy, the system is immune to the effects of noise and variations in signal quality to a very great extent. All that is required of the communication link in a digital applicaton is that it be capable of reliably distinguishing the *presence* or *absence* of each bit at the receiving end. Whether a logical 1 is weak or strong, square or ragged, it needs only to be recognized as a 1 by the receiving device.

Transmission of data is most simply accomplished by the use of parallel channels; one channel is allocated to each bit position. For reasons of economy, however, serial transmission is used in most practical systems. Through the use of a serial format, a data stream of any complexity can be carried on a single wire or through one radio channel, if adequate means of synchronization are provided in the format. One disadvantage of a purely serial transmission is the requirement for a higher data transmission rate and, consequently, more transmission channel bandwidth. For instance, if a continuous stream of 10-bit data words is carried in a parallel format with a word repetition rate of 1k pps, 10 separate lines or channels are needed, and each channel must be capable of handling a 1k-pps signal. If, however, the data is scanned into a single serial format, only one line is required, but it must carry a 10k-pps (or slightly higher) bit rate. In most practical situations, a tradeoff is made to match the necessary data rates to the available transmission channel bandwidth, usually

in the form of a serial-parallel transmission format. In the previous example, if several 3-kHz telephone channels were available for signal transmission, a practical compromise could possibly be the use of five channels, each channel handling one scanned data line at 2k pps.

In digital data communication, provisions are always included in the transmission format to establish a firm time-frame relation between the transmitted and the received signals. This part of the process is called *synchronization*, and is roughly analogous to the use of punctuation in written communication. Just as spaces, capitals, and punctuation marks enable us to make sense out of a written message, special data characters and timed spaces are used to give data bits and words their proper meaning in data communication.

The synchronization problem exists at several levels in a complex transmission format. Typically, each data bit is in some way locked in to a radio- or audio-frequency carrier, and the groups of bits are usually transmitted at an exact submultiple of the carrier. In most formats, a fixed number of grouped data bits constitute a data character, or subword, and a fixed number of characters are called a *word*. Although the terminology of aggregate groups of data words is not well standardized, in general, an organized quantity of grouped characters is called a *record*, and several records constitute a *file*. For each degree of aggregation of data, a unique form of recognizable punctuation must be provided along with the data, either interlaced with the data stream or, in some cases, as a separate control signal. Each of these synchronizing tags must be unique and recognizable.

Providing immunity against the effects of noise is a major consideration in the design of a data link. *Noise* and *errors* may be loosely defined as the signal difference which appears between the received data and the transmitted data, and can be introduced by malfunctions in the transmitting and receiving digital circuits, or by propagation path problems such as changing atmospheric conditions and radar interference.

As a general rule, digital data is not presented directly to the communications medium, but is used as a modulating signal which operates on an audio- or radio-frequency carrier. The nature of the modulation, whether it be amplitude, phase, or frequency, is an important factor in determining the noise immunity and overall reliability of a data transmission system. A great deal of research and development effort has gone into the specialized field of data modulators and demodulators; in fact, has given rise to a distinct class of hardware called *modems*, a combination of the words *mo*dulator and *dem*odulator.

The design and internal workings of modems will not be covered in any detail, as the modem is ordinarily an independent and nondigital component of a data transmission system. A well-designed, properly operating modem acts simply as a vehicle with no *logical* effect in the system. It may be assumed that the output of a modem (at the data receiving end) is identical to the original input, usually consisting of a data line, or data lines, and a clock signal.

Error Checks

Reliability and freedom from communication errors may be enhanced not only by the use of hardware designed to optimize the electrical quality of transmission and reception, but also by the use of logical checks and error traps in the data format. Depending on the degree of confidence required, the system designer might include as little as one extra bit for checking a 1000-bit message, or, at the other extreme, he might set aside as great a portion of the format for checking functions as is used for the actual data.

The simplest and most often used data quality check is the *parity count*. Transmitted with the data, whether serial or parallel, is one extra bit at regular intervals which indicates whether the sum of the data 1's in the interval is an odd or an even number. In the serial data train shown in Fig. 7-1, each six-bit data word is followed by what appears to be a seventh data bit. This parity bit, as shown, is 1 when the sum of the 1's in the corresponding word is even, and is 0 when the sum is odd. At the transmitting end of the data link, each word is examined by a 1's counter and the parity bit is set or reset accordingly.

At the receiving end, the data is operated on by an identical counter, and the result (odd or even) is compared with the received parity bit. If the receiving parity check does not agree with the transmitted parity bit, it is evident that the received data is not identical to the data as it was transmitted. Fig. 7-2 illustrates the overall logic of a simple parity checking system.

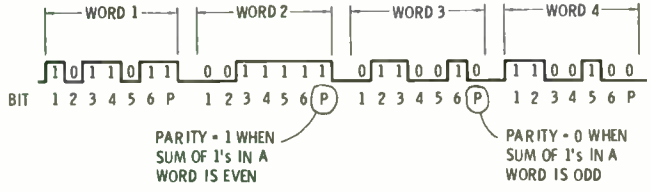


Fig. 7-1. Serial parity check.

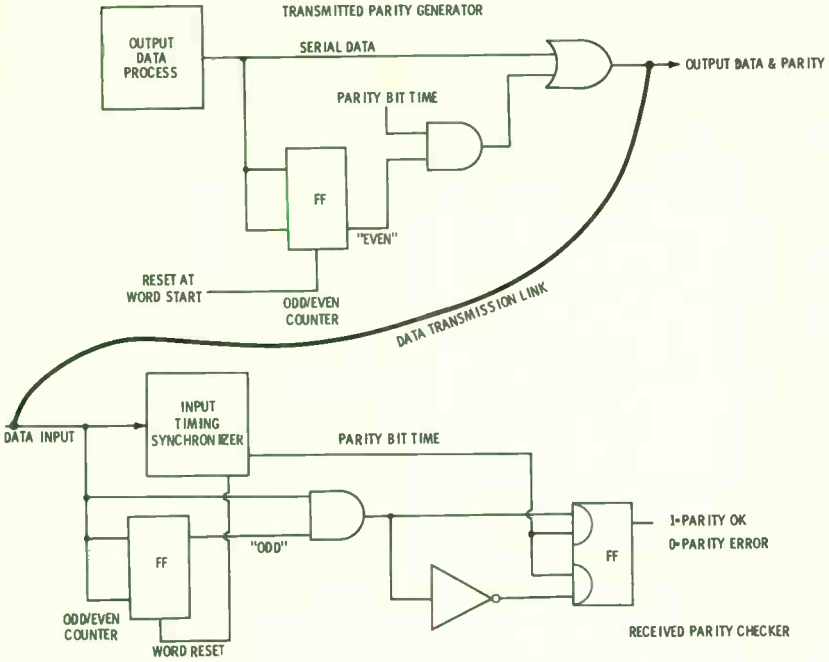


Fig. 7-2. Parity checking.

This one-dimensional parity check suffers a very obvious drawback. If two errors, or any *even* number of errors, occur in the word or character covered by the parity check, the check is ineffective. This limitation can be minimized, but not entirely eliminated, by the use of more frequent parity checking, to whatever degree is practical.

A two-dimensional parity checking method is illustrated by Fig. 7-3, in which the data in a serial-parallel format is checked both laterally and longitudinally. Each of the parallel characters includes an extra bit to indicate parity count, and serial parity bits are impressed at regularly spaced intervals in the format. This method is almost universally used in magnetic tape recording of digital data, as well as in data transmission, providing an excellent indication of the presence of an error whether the error is an extra 1 or a dropped bit. Equally significant is the fact that the exact location of a bad bit can be determined by the intersection of the lateral and longitudinal parity error indications. Through this analysis, errors are not only detected and located, but may be corrected by substituting a 0 for 1 or a 1 for 0 at the indicated point of error.

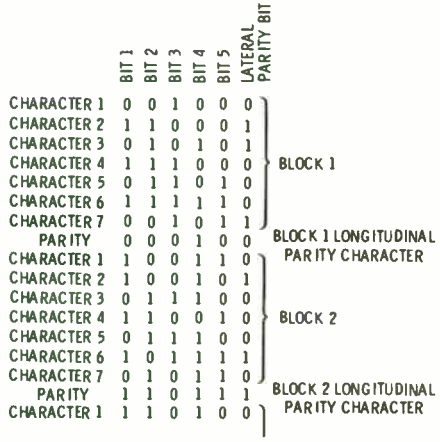


Fig. 7-3. Two-dimensional parity check.

By narrowing down the coverage of each parity check bit, the probability of undetected errors can be made as low as may be desired. The data carrying efficiency of the format, however, becomes reduced, since a relatively large portion of the format might be used for the "overhead" bits, leaving correspondingly less space for active data. A lot of research has gone into the development of mathematical and logical techniques other than parity counting for the trapping of errors in transmission, recording, and processing. It is beyond the scope of this book to attempt to analyze the complex mathematics of the methods in popular use. The application, however, may be very simply generalized. An error check word, or polynomial, is developed by subjecting the outgoing data to *some form of* logical manipulation, and this check word is transmitted with the data. At the receiving end, the data is subjected to an *identical* logical process, locally producing the error check word. Comparison of the two error check words, which should be identical, gives an indication of the presence and, in some systems, the location of the errors in the data. One check of this type consists of adding up the data 1's in the stream and transmitting a sum check word at regular intervals. Other, much more sophisticated manipulations are usually used in systems which demand an unusually high degree of reliability. These methods, although extremely complex in mathematical principle, as a rule involve a network of conventional logic elements for the generation of the error check word, and may be analyzed electrically and logically without the need for understanding the underlying mathematical theory.

Data Formatting

When continuous data is to be transmitted over a single communication channel without the use of an auxiliary channel for control and synchronization, the data transmission system must make provisions for synchronizing the receiver and transmitter. These provisions are included as a part of the structure of the data message, or data format.

The overall problem of designing an efficient format is analogous to that of designing the frame of a television video signal. Without horizontal and vertical line and frame synchronization signals, a received TV picture would be a meaningless blur of white, black, and gray dots. When synchronized, the horizontal and vertical oscillators in the receiver are effectively locked in the proper relationship with those in the transmitter, and each bit of video is placed where it contributes intelligence to the picture.

In a transmitted data stream, some means of synchronization or identification must be included for each of the levels of data aggregation. If, for example, data is transmitted in a sequential frame which contains several unrelated words, and each word is made up of several characters, the data format must make it possible for the receiver to recognize and sort the structure of frame, characters, words, and bits. At each level, either a distinctive code tag or a unique timing relation must be provided.

Let us assume that a continuous sampling of engineering test data is to be transmitted over some distance and the data consists of three words, representing thrust, rpm, temperature, and a fourth word indicating that the time of measurement is included with the data message. Fig. 7-4 illustrates a format which might be used to satisfy such a requirement.

The entire format contains exactly 100 bits, or bit positions, although there are only 65 actual data bits. Of the 35 "overhead" bit positions, 12 are used for sync code (1) and parity (P) bits. The remaining 23 slots are not a necessary part of the format, and serve only to round out the overall format to a convenient length of 100 slots.

In the design of a transmission format and, for that matter, an entire system, the rate at which the data values can vary must be adequate to accommodate the problem. In this example, we will assume that a rate of one sample per second is sufficient for the three data words. This would indicate a frame rate of one per second and a bit rate of 100 bits per second, which is easily within the capacity of a telephone-line modem.

In the example of Fig. 7-4, a very simple sync code of 111111 is used to indicate the start of the frame. This code, although subject

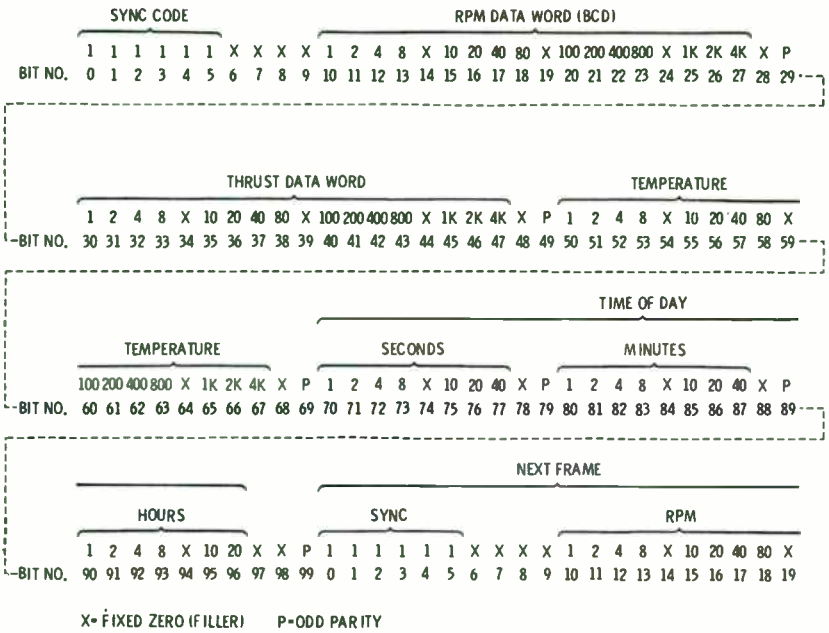


Fig. 7-4. Serial data transmission format.

to errors in the presence of noise, is unique, since under no conditions can the format contain six consecutive 1's except in the sync code. Examination of the frame format shows that the data words and character are separated into groups of no more than four or five consecutive bits with zero "filler" bits (×) interspersed between characters.

In an arrangement typical of working data formats, the rpm, thrust, temperature, and time words are positioned at regular intervals as BCD characters. Although no sync codes are provided other than for the overall data frame, the words, characters, and bits in the frame may be recognized by their time position in the format.

Sometimes, for more reliable synchronization, the sync code is inverted, or complemented, in each alternate data frame. In the example of Fig. 7-4, one frame would include the 111111 sync code, the next a 000000 code. At the receiver, the sync decoder or detector would have to be programmed to seek the alternating code.

Fig. 7-5 gives, in block diagram form, the overall logic of a system which would generate a format as described in Fig. 7-4. The corresponding data receiving system for this format might be blocked out as shown by Fig. 7-6.

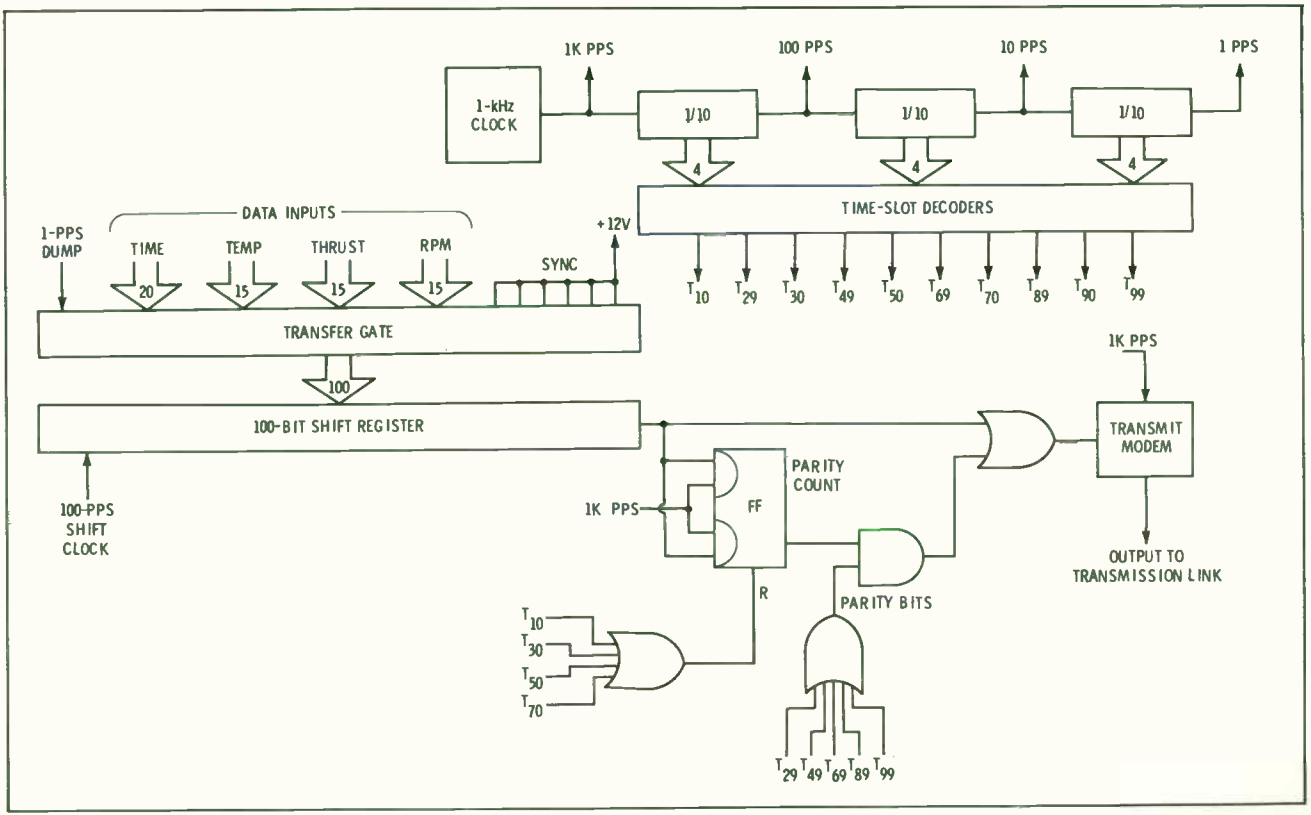


Fig. 7-5. Serial data transmitter.

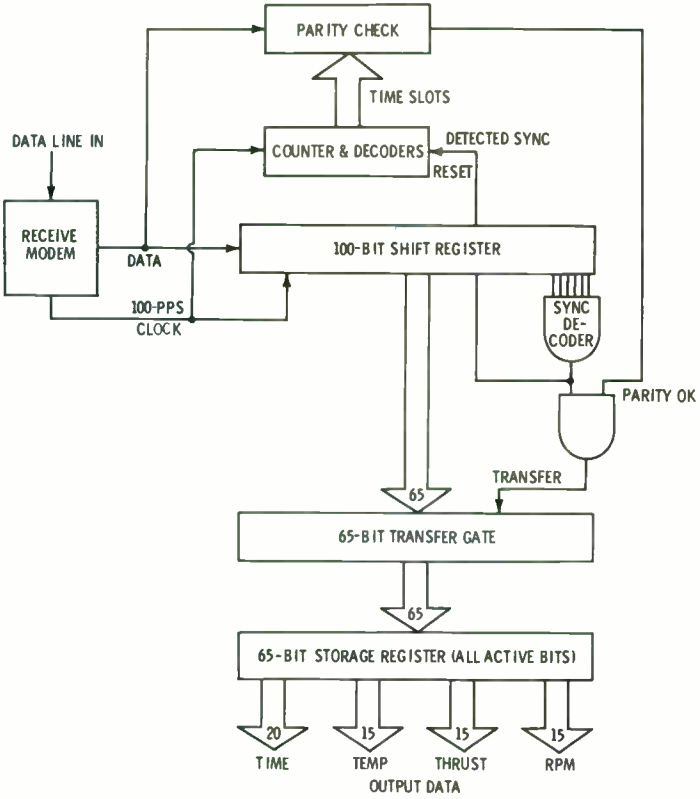


Fig. 7-6. Serial data receiver.

In Fig. 7-5, a 1000-Hz master clock drives a three-decade counter which, through time decoders, produces all of the necessary control signals and pulse rates down to 1 pps. Once per second, a sample of all of the input data is dumped into the 100-bit shift register, which is clocked at a continuous 100-pps rate. A single toggling flip-flop counts the data 1's in the shifting data, generating the parity tags. Before each word-parity count begins, at T_{10}, T_{30}, T_{50}, and T_{70}, the parity counter is reset. At the instant of sampling the parity count, the parity gate is enabled by decoded time slots T_{29}, T_{49}, T_{69}, and T_{99}, and the sequential parity bit is OR'ed into the data stream. The composite bit stream of data, parity, and sync code is fed to the transmitting modem and is prepared for transmission. In most systems, a clock signal is also delivered to the modem, or else the modem provides the clock to the data system to ensure a stable relationship between the data and the modem carrier signal.

The data receiver shown by Fig. 7-6 consists mainly of a 100-bit shift register and data storage register. The data frame is continuously shifted through the register, and, once per second, when the sync code is recognized, a transfer is made delivering the data from the shift register into the storage register. In a system as simple as the one described by Fig. 7-6, all of the necessary data transfer control is accomplished by the action of the detected sync pulse and no other timing control need be provided. However, to perform the parity count in the same manner as in the transmitter requires a counter/decoder to provide the reset and sampling signals for the parity flip-flop. For this purpose only, a 1/100 counter is driven continuously by the incoming 100-pps clock, and is kept in step with the transmitter clock/counter by the action of the detected sync code.

The manner in which the result of the parity check is used at the receiver is a matter of the designer's choice. In some systems, the presence of a parity error is simply used as a warning indication; sometimes, as illustrated, the presence of a received parity error may be used to inhibit transfer of the incoming data frame into storage, if it is considered better to lose one frame of data than to deliver a frame which is known to contain an error.

ARITHMETIC PROCESSING

All of the arithmetic functions performed by a general-purpose computer are made in very elemental steps by the combined operation of an adder, shift registers, and storage registers. As described in Chapter 4, addition and subtraction are both handled by the full-adder circuit, plus some storage and control logic. Multiplication is performed as a series of steps of binary scaled addition, making use of the fact that each time a data word is shifted one step "upward" in a binary register, its numerical value is doubled. Using this property of a shift register, a binary word is multiplied by another binary word by a process of adding the word itself as many times as called for by the value of the multiplier word. Under control of the multiplier word, the data is shifted, added, shifted, and added to produce the final accumulated product. Division may be accomplished by a similar process in reverse.

Each of the arithmetic or other mathematical processes could be mechanized by calling out every detail of the operation in the instructions to a computer, and in some simple computers this must be done by the programmer. Most computers, however, are designed for automatic performance of the basic arithmetic functions, and require only a simple program command for execution. In general, the more complicated functions must be performed as

a series of simple arithmetic steps called out by a programming specialist. The solution, or method of computing a problem, is usually stored as a program on punched tape or similar input control medium, and played into the control section of a computer memory when that problem is required to be processed. Each time a solution program is produced for a specific type of computer, it is kept on file for future use with a minimum of additional effort.

STORAGE AND MEMORY

A very fundamental part of any computer and an important part of most data processing systems is that portion which performs the function of storage. The terms "storage" and "memory" are synonymous, although the latter usually connotes high-volume, long-term storage. In essence, storage refers to the retention of data, from the simplest one-bit momentary storage such as the one-shot, to the complex magnetic core arrays and disc memories which are used for permanent record keeping by accounting and scientific data processing computers.

Storage units are divided into several classes—large and small, temporary and permanent—depending on the uses for which they are specified. A typical computer includes several types. The main working memory, in which data is more or less permanently stored and updated, usually consists of a high-speed array of thousands of ferrite cores or similar one-bit elements, and can be instantaneously addressed for retrieval of data or insertion of new data. All of the data which might be required in the solution or processing of the problem at hand is placed in the main memory before the start of computation and recalled from the memory when needed during the steps of the computing process.

Data which must be held permanently and recalled only occasionally without the requirement for microsecond access is often stored in a magnetic disc file. This form of storage consists of a large disc, or several discs in a stack, coated with a magnetic material similar to the oxide coating on conventional magnetic home-recording tapes. The discs rotate continuously at high speed, in contact with multiple recording and pickup heads which are positioned at fixed, close intervals from the center out to the periphery of the discs. The data is stored in concentric rings, somewhat like the acoustic signals on a phonograph record, serially in each ring. If, for example, a disc were arranged as 100 rings, each ring capable of resolving 20,000 bits of data, the disc could be loaded and unloaded with 2,000,000 bits by a single head with variable radial positioning, or by 100 heads in fixed

locations. This type of memory is always addressed, or interrogated, in the *sequential* mode; that is, every data location around the periphery is scanned in turn as the disc rotates. If one specific data location is to be read, the system must wait until that location comes under the pickup head. As a result of the restriction of sequential operation, access to a disc file typically requires several milliseconds, as compared to one or two microseconds or less cycle time for a *random access* core memory. In the random mode of access, any desired memory location may be interrogated at any given time.

Even more permanent forms of storage are used in applications where extremely large quantities of data can be filed and manually retrieved when needed. The most useful forms of long-term data storage are magnetic tape, punched paper tape, and punched card files. These forms of storage are usually used indirectly, delivering the data in total or in groups of characters, to a high-speed memory unit in the data processor, prior to the start of computation.

Once the data to be processed by a system has been delivered from permanent storage to a high-speed memory, it is usually read out from memory, a word at a time, as the problem requires, and delivered to a holding register, which is a small, temporary storage unit. A register typically consists of a group of flip-flops and whatever gates may be needed to accommodate input and output data transfer. In the course of a problem, data words may be held in one of several registers, shifted through arithmetic or other logical processing blocks, and finally returned to the main memory or read out to the output displays, printers, or transmission system.

Here, another distinction between classes of storage should be noted—volatility. A storage device such as a flip-flop which requires uninterrupted supply power, and loses its data if the power fails momentarily is said to be *volatile*. All of the forms of magnetic storage in well-designed systems must retain their storage data reliably with or without power applied, and are classed as *nonvolatile*. Needless to say, data processors which must reliably maintain permanent records, as is required of most business and accounting computers, use nonvolatile forms of memory for the main storage.

COMPUTER PERIPHERAL UNITS

In a typical large data gathering and processing system, the computer represents a relatively small part of the overall hardware. Among the many components which may comprise the system, those subsystems which operate directly in conjunction

with the computer are called *peripheral units*. In most cases, operation of the peripherals is directly controlled by the computer. Some of the most commonly encountered standard peripheral units are:

Data transmitters and receivers
Input keyboards
Magnetic tape transports
Line, page, and strip printers
Card and paper tape punches and readers
Plotting boards
Disc memory files
Pictorial display units
Time code generators
Time-sharing terminals

Each of these accessory units may be essential in some way to the operation of a system, depending naturally on the overall function which the system is required to perform. In a very general sense, the peripheral units provide an adequate, or optimum, means of introducing input data to the computer and presenting the computer output data to the user. Selection of the most suitable peripherals for a system is a matter of very great economic importance in the design of a system, considering the complexity and cost of the more sophisticated units. Required speed of operation is often a major consideration in the selection; for instance, the choice between an output typewriter and a high-speed line printer might represent a difference of $25,000.00 in the initial cost of a system.

Many manufacturers concentrate their efforts on only one, or perhaps a few, of the very specialized types of peripheral equipment, often supplying their specialty hardware directly to the manufacturers of computers, to be integrated with the computer package.

Magnetic Tape Units

Almost all large computer installations include one or more magnetic tape handling units used primarily as a data input device. The tape system is ordinarily used for one of two basic functions: as a permanent form of data storage and as a source of data when the computer is used in an "off-line" process.

Off-line operation may be defined as the processing of data which has been gathered and formatted at some previous time, and recorded for later use by the computer. By contrast, an "on-line" or "real-time" process is one in which live data is fed directly to the computer. Economic and practical scheduling of a com-

puting facility dictates that real-time data processing be performed only when absolutely demanded by the nature of the problem. Through the use of off-line computation, a large computer can be shared by many users without the need for the complications of instantaneous timesharing, or multiplexing.

Although any suitable format might be used in preparing a data tape, the industry has adopted, almost universally, the use of one standard format for the tape storage of large quantities of data. This format, developed by IBM, stores data as six-bit parallel *characters* transversely at very close intervals on ½-inch tape. With a parity bit accompanying each character, the data is recorded and played back through a precision tape head which includes seven independent elements. For convenience in organizing and identifying separate groups in a large quantity of taped data, an arbitrary number of characters, usually about 1000, is called a *record*, and is identified with a block of characters called a *header*. The header usually includes useful information about the record itself, such as the time or date at which the data was gathered, or the location of the data source. At regular intervals, a longitudinal parity character is recorded, providing the two-dimensional error check. Several *records* on a tape constitute one *file*, the largest standard aggregate of taped data.

In the standard IBM format, the data is recorded at intervals of 556 characters per lengthwise inch of tape. The end of each record is indicated by a blank, or unrecorded, gap which is 0.75 inch long. At the end of each file is a similar gap 3 inches long. In the usual playback process, through a very high-quality tape motion control, the tape motion is started by the computer request for a record of data and is brought up to full speed within a few milliseconds. Data is read into the computer memory or into an auxiliary memory unit continuously until the end-of-record gap is sensed by the absence of data or parity bits. The tape is brought to a very sudden stop while the read head is still in the end-of-record gap, and remains stationary until the next data request from the computer.

Details of a typical IBM-formatted magnetic recording are shown by Fig. 7-7. As a rule, when data representing several different parameters is recorded, the data is grouped into *samples*. Each sample or block contains one full word of each parameter.

The quality and complexity of a digital tape transport are usually much greater than that required of analog or high-fidelity tape decks because of the extremely precise tape motion control. In general, the quality of the electronic portion of the tape unit is not so critical since, as in all digital electronics, the playback process is only required to sense the *presence* or *absence* of the

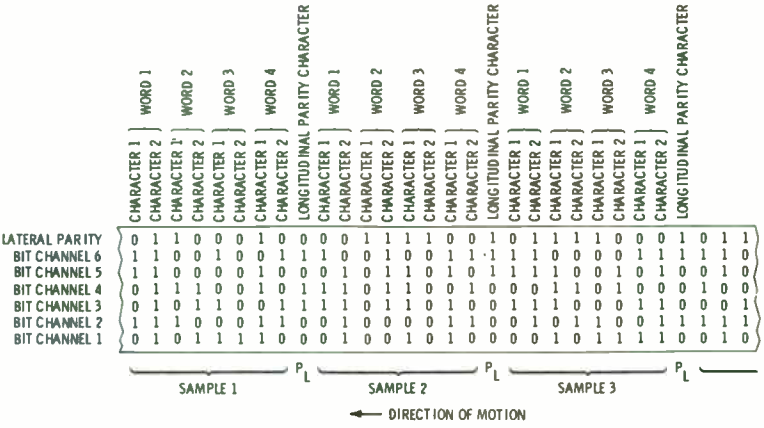


Fig. 7-7. Magnetic tape data format.

recorded bits, and is not concerned with the exact amplitude or quality of the bits.

A relatively new concept in digital tape control makes use of an incremental drive system in which the recorder does not run continuously when making a record, but uses a stepping motor which is energized each time a character is to be recorded. In this arrangement, the coupling between the stepping motor and the tape is designed to move the tape exactly $\frac{1}{556}$ inch to produce the standard recording density. This method of control is considerably simpler and less costly and produces tapes which are identical to those recorded by the conventional method.

With the development of improved magnetic tapes and equipment, a higher-density standard format is becoming increasingly popular. This format, also identified with IBM, is very similar to that of Fig. 7-7 using eight-bit-plus-parity characters and a packing density of 800 characters per inch.

Data Input Devices

Data and control signals are brought into data processors by a variety of devices, and often by the simultaneous use of several. Typically, a large computing system includes several direct data channels, each of which receives a data stream from a local data acquisition peripheral, plus a means of reading in data which has been recorded and stored.

In addition to adequate accommodations for data input channels, most general-purpose computers use a separate input channel through which program control instructions are introduced. Naturally, the size, complexity, and intended application play an

important part in determining the speed, capacity, and flexibility of the computer input channels. If a computer is intended for use as an industrial process controller, for example, and will most likely devote its entire career to the solution of a single type of problem, one input channel is practical and sufficient for both program and data inputs. At the other extreme, in an installation requiring flexibility of operation and rapid turnaround between different types of jobs, the input processing equipment must be more extensive.

If a generalization can be made regarding the great variety of computer designs in use today, the main working memory of a computer is designed, or selected, with sufficient capacity to store all of the data received for the data process, and also the detailed program and control instructions. For simplicity of programming and efficient organization, one section, or block, of continuous memory locations is usually set aside for the instructions, and the balance of the locations, or addresses, is made available for the data in process. All of the information which is fed to the computer is initially placed in the working memory or temporarily held in the memory during computation.

The devices which present the input data and the program instructions are designed to deliver characters, or *bytes,* of data to the computer memory at a rate and sequence which are paced by the computer. Input devices of four basic types are used in practically all computer systems—magnetic tape, paper tape, punched cards, and manual keyboards.

Magnetic Tape—Large masses of data are conveniently handled at high speed by the use of digital magnetic tape input units. Usually formatted in one of the standard IBM character, record, and file frameworks, taped data can be transferred to a computer on demand at rates up to 100,000 characters per second. The attractive features of speed and high storage capacity inherent in tape data handling are accompanied usually by a high degree of mechanical and electronic complexity dictated by the very stringent performance requirements of the tape motion control system.

Paper Tape—A slower, simpler, and less costly vehicle for recorded data entry is the paper tape reader. This system, which has evolved over more than a century from the original applications in telegraph, textile loom control, and the player piano, makes use of a reel of paper, foil, or plastic tape with data bits represented by small holes punched in a coded pattern. The paper tape most commonly used is about one inch wide, with five- to eight-bit characters impressed laterally. Conventional paper tape handlers employ a sprocket drive for controlling the motion and position of the tape, and sense the presence or absence of bits by

an array of miniature pin contacts, brushes, or switches. Because of the limitations of mechanical positioning, contact inertia and bounce, and tape strength and stability, mechanical tape readers are limited to speeds up to about 25 or 30 characters per second. Higher-speed paper tape units, using photoelectric or pneumatic data sensing are often used up to speeds as high as 1000 characters per second.

The relative simplicity and low cost of a paper tape reader make it a very practical method of inserting low- or medium-speed data or program instructions into a computer. Many off-the-shelf computers include a paper tape input unit as the standard input device, with provisions built in for the addition of a magnetic tape handler if an application demands higher data rates. Paper tape data inputs are used almost exclusively in the controllers of automated machine tools. Their rugged construction and moderate speed are ideally matched to most tool control problems.

Punched Cards—Punched cards have been used in various forms and sizes for many years in mechanical and electronic calculators, control systems, and record-keeping systems. The punched card is fundamentally similar to paper tape, presenting a parallel data readout in the form of holes punched in a cardboard, plastic, or metal card. Cards of any convenient size and format can be devised; however, the most universally accepted standard card is the familiar IBM card used by many government and business agencies. This card, about 3 by 7 inches, accommodates data punched as rectangular holes, arranged in 80 columns and 12 rows.

Punched card data is used in more varied applications than the other popular forms of recorded data. The reading devices which translate data from the cards to electrical signals range in complexity from a static array of switches in a manually operated reader, to the very elaborate consoles which automatically feed, sort, and read from a stack of hundreds of cards.

When used in the simplest static application, a punched card can serve very much the same purpose as a patch panel, controlling a network of interconnections. In this sense, the punched card may also be classified as a simple "read-only" memory in which the instructions for, or solution to, one specific problem are stored. In high-speed processing of punched card data into a system, the data may be read in parallel, a card at a time, but is usually scanned out in characters. As with paper tape, card data may be read by contact, brush, or switch sensing, or by photoelectric scanning. Depending on the method of handling and sensing, punched cards are automatically processed at a rate of a few hundred to about 2000 cards per minute.

A useful feature of punched card data is the convenience of printing the data in visually readable form in combination with the punched holes. We are all familiar with billings from telephone companies, most credit card agencies, and many large department stores, which include a printout of the account number, name, address, and transaction records with the punched data.

Keyboard Input—The fourth category of input devices used with most computing systems is the manually operated keyboard. Of all of the methods used for data entry, the keyboard is unique in that it provides the operator direct or on-line access to the computer input circuits, without an intermediate record/playback step. Most applications make use of a combination keyboard and printer, similar in appearance to a typewriter, for use as both an input and an output access to the computer. The printout feature serves an equally important function in the input process by providing the operator with a means of monitoring the data as it is written into the computer, reducing the probability of operator error. In some systems an input-only keyboard is used, with a separate visual display or other form of readout furnishing a "quick-look" means of monitoring the input data. A keyboard of this type consists simply of an array of numbered and lettered push-button switches arranged in a convenient, usually standard, layout.

Most of the standard keyboards available are produced by a few specialty manufacturers. Several of the popular designs consist simply of a modified electric typewriter with input and output decoders and encoders. Data to and from a typewriter-style keyboard is usually in the form of six-bit-plus-parity parallel characters, providing a capacity of 64 unique combinations, ample for the alphabet, a numeric scale, and special control characters.

Although write-out keyboards are gaining popularity in data systems because of their relative mechanical simplicity, the most commonly encountered keyboard device in the industry is still the teletypewriter. The teletypewriter is very much like a desk typewriter in outward appearance, but its internal construction is completely unlike that of a typewriter or other devices of its kind. In spite of the marvelous mechanical complexity of its coding, scanning, and printing mechanisms, the teletypewriter has developed through many decades in the communications industry to an extremely reliable and relatively inexpensive instrument.

All of the keyboard data input controllers are naturally limited by the operator's own dexterity and speed. Some, such as the teletypewriter, because of a mechanical timing cycle, are limited to a speed which is slightly less than a very skilled operator's own physical limit.

Direct Access—In a discussion of methods of entering data into a computer, it should be indicated that most, if not all, computers include a bank of switches through which data can be introduced without any auxiliary input device. Data entered in this manner must be placed in the switch register *bit-by-bit* and manually directed through switches into the proper memory locations. This process is so tedious and time-consuming that it is ordinarily used only in troubleshooting a malfunction or in making small corrections to a program which has been stored.

Data Output Devices

The application for which a data processor is designed is the major governing factor in the selection of a suitable display or readout system. If a system is to be used for air traffic control, for example, the most important part of the system may be an active pictorial display of traffic around an airport, and the bulk of the system may be devoted to the output display problem. An equally complex data system designed to manage the accounts of a bank or control the inventory of a department store chain might require no more than a simple page printer for its entire output presentation. Just as is true of input devices, the speed, complexity, and overall elegance of an output system are dictated by the nature of the user's requirements.

Output displays and readouts, as defined previously, are generally distinguished as being momentary or permanent. A *display* is ordinarily considered to be a visual presentation of lights or numbers, or a pictorial representation of data. The term *readout* usually implies hard copy such as a printed page or punched card.

Each of the four basic types of data input device has as a counterpart an output processing device. The magnetic tape playback unit must be associated with a similar magnetic tape recorder; the paper tape reader must be associated with a tape punch. In each example, the characteristics which apply to the data input process generally apply to the output process of the same medium, in terms of operational speed, data handling capacity, cost, and extent of usage. All of the popular forms of input/output data handlers are designed as dual-purpose units, but each type may be purchased for operation in only one of the two modes. A magnetic tape unit uses the same tape motion control mechanism for both recording and playback, but, as a rule, the recording driver electronics and the playback amplifiers are supplied as optional accessories by the manufacturers. Paper tape punches and tape readers are usually mounted as separate units on a common assembly, providing input or output or a dual function.

Output Printers—To satisfy the wide variation in hard-copy printout requirements, many different forms of data printers are in use. They vary in size from small strip printers, producing a single line of typed or printed data on a ¼-inch paper tape, to the huge high-speed line printers which continuously bang out as many as 200 columns of print per line at speeds as high as 100 lines per second. Three methods of printing are commonly used: impact, photosensitive, and electrosensitive.

The impact method of printing used in a typewriter mechanism is the most common, and it is employed by most medium-speed printout systems. This method of producing readable characters is performed in many different detailed ways, but in all of its variations it involves fast-moving mechanical components, and it is limited in speed by the mechanical action. Impact printers of all types tend to be somewhat noisy, especially those which produce an entire line in one stroke.

Photosensitive and electrosensitive printers outperform impact devices in speed and silence of operation, but require the use of special paper which is sensitized or impregnated with a chemical solution designed for each specific process. In some systems employing sensitized paper or internal developers, the printout is damp when produced and requires special handling until dry.

Output systems other than those which produce hard copy are often designed to suit users' requirements. Two which deserve mentioning as general types, or classes, are plotting boards and cathode-ray-tube displays.

Plotting Boards—In principle, plotting boards are very uncomplicated two-dimensional readout systems, although precision and accuracy requirements usually dictate somewhat complex design and construction. A general-purpose plotting board receives data representing two variable position coordinates such as the computed latitude and longitude of a moving vehicle, and continuously positions a pen on a sheet of paper to plot the vehicle's track. Often, two independent pen movements are assembled on a single plotter, to produce an approximation of a three-dimensional track, superimposing an *X*-*Y* and *X*-*Z*, or *Y*-*Z* plot.

When used in conjunction with a source of digital data, the positioning mechanism of a plotting board may use one of two fundamental methods of following the data. In the older, analog style, the input data is initially converted to an analog voltage, ac or dc, and used to control a conventional follow-up servo which positions the pen. A digital servo plotting board makes a direct comparison between the input digital data and a digital word which is generated by a shaft encoder coupled to the pen, driving the pen to a position which makes the two words exactly equal.

Another type of digital servo makes use of a stepping motor, positioning the pen in a series of tiny steps in accordance with changes in the input data.

CRT Displays—Cathode-ray-tube displays may be used to generate a presentation similar to that of a plotting board, but with a much greater degree of flexibility. Without the mechanical limitations of the plotting board a CRT display unit can be designed or programmed for virtually any type of graphic or symbolic presentation, instantaneously selecting, combining, and superimposing lines, characters, special symbols, and printed messages. The major limitation of a CRT output data display is the lack of permanent copy, although this is sometimes offset by the use of an auxiliary optical system and a camera or photosensitive paper.

INDEX

COMPUTER DATA HANDLING CIRCUITS

by Alfred Corbin

Up to now the difficulties usually faced by the beginner in acquiring an education in digital data circuits were formidable—long courses in many areas of electronics, each course containing much that is unnecessary or only remotely related to a practical understanding of the subject. This book has eliminated the nonessentials from those courses. It offers the beginner a handy one-volume course in digital circuit analysis.

First to be covered are semiconductors and the semiconductor circuits used in digital equipment. This treatment will be of special interest to the technician not conversant with semiconductor circuit theory. Digital data logic, which deals with data in the form of signals whose presence or absence need only be detected, is explained next, along with codes and Boolean algebra, the mathematics behind the design of digital systems. The basic logic circuits, such as AND and OR gates, inverters, and flip-flops, which produce the data signals, are then analyzed. Functional blocks of these logic circuits, such as counters and scalers, decoders, and shift registers, are covered simply and clearly. There is a full chapter on digital display devices. Forms of data conversion are also discussed, including the several ways in which conversions are made from digital to analog data, and vice versa. Computer subsystems, including peripheral units, are surveyed in the final chapter.

This book will be of value to technicians and engineers wanting information on a field outside their own; to students looking for a quick, straightforward course in digital data circuits; and to software personnel seeking to understand better the equipment they are associated with.

ABOUT THE AUTHOR

Alfred Corbin received the B.S.E.E. from the University of Maryland and did postgraduate work at Miami University. He has worked in missiles as a Project Engineer in Central Timing and as a Range Planning Engineer. He also designed analog computing systems in electronic flight simulation. He has held the positions of Chief Digital Systems Engineer, Chief Engineer, and Vice-President of Engineering. Currently, Mr. Corbin is President of Metric Systems Corporation. He has written several technical papers, and he holds one patent and has three patents pending, all on digital and computer equipment.

HOWARD W. SAMS & CO., INC.
THE BOBBS-MERRILL CO., INC.

20808

$5.50 (In Canada $6.60)