11 DIGITAL CIRCUITS AND DEVICES

The digital world is a world of ones and zeros. It utilizes circuits that are on or off, logic conditions that are true or false, and voltages that are high or low. By recognizing only two possible conditions, errors due to component tolerance and temperature drift are all but completely eliminated. The real world is an analog world. The speed of a motor, the temperature of a process, and the intensity of a laser beam are all examples of analog measures. There are an infinite number of actual values in the analog world. It is possible to use digital circuits to measure, control, and generally interact with the analog world. The current trend is to use more digital and less analog circuitry in industrial electronics. The industrial technician must understand both types of circuits and how they are interfaced.

11-1 CHARACTERISTICS OF DIGITAL CIRCUITS

Early digital circuits used vacuum tubes. Then, in the 1960s, integrated digital circuits were developed. The first integrated digital circuits were based on resistors, diodes, and transistors. They were *resistor-transistor logic* (RTL) *circuits*. *Diode-transistor logic* (DTL) was the next step. By the 1980s, RTL and DTL devices were obsolete for modern designs. They have been replaced by the *transistor-transistor logic* (TTL) and the *complementary metallic oxide semiconductor* (CMOS) families. This section will cover the important electrical parameters of TTL family devices; a later section will cover the CMOS family.

Figure 11-1 shows the input signal requirements for a TTL circuit. Any signal voltage from 2 to 5 V is interpreted as a logic 1. These levels are called $V_{INH(MIN)}$ and $V_{INH(MAX)}$, respectively. Any voltage from 0 to 0.8 V is interpreted as a logic 0. Notice that the region from 0.8 to 2 V is forbidden. Any signal in that region would produce unpredictable output results. Obviously, a signal that is changing from 0 to 1 or from 1 to 0 must go through the forbidden region. This is acceptable. A signal that remains in the forbidden region for any length of time is not acceptable. An input that is floating (not connected to anything) will measure between 1.1 and 1.5 V, which is in the forbidden region. The device will interpret this as a high input (logic 1), but a floating input invites noise pickup. Therefore, an input that is to be fixed at logic 1 should be tied to the positive supply. The TTL systems run on a 5-V power supply. Most signals will be between 0 and 5 V. Signals more positive than 5 V should not be applied to a TTL input because damage may result. A signal that is negative with respect to ground can also damage a TTL device. Most devices have input clamp diodes to protect against negative signals, but a signal more negative than 1.5 V may damage the input.

Most often TTL inputs are driven by TTL outputs. When the expected output levels are compared to the required input levels, the margin for error can be evaluated. The rectangular waveforms shown in Fig. 11-1 are for the worst-case and typical levels that can be expected at the output of a device. *Worst case* means that the manufacturer guarantees the logic 1 level from the device supplying the signal to be at least 2.4 V and the logic 0 level to be no more than 0.4 V when the device is fully loaded. The difference between the worst-case driving signal levels and the input thresholds at the driven device is seen to be 400 mV. This is called the *noise margin*. In practice, the actual noise margin is better. The typical TTL signal source will swing from 0.2 V (logic 0) to 3.4 V (logic 1) when fully loaded. A full load is also called a *full fanout* and is shown in Fig. 11-2. The signal source, called the *driving gate*, must supply logic 0 or a logic 1 to 10 driven gates. By the nature of the totem pole output stage (discussed in Chapter 2) in the driving gate, the logic 0 sink current capability is greater than the logic 1 source current capability. The totem pole pair uses a load resistor of

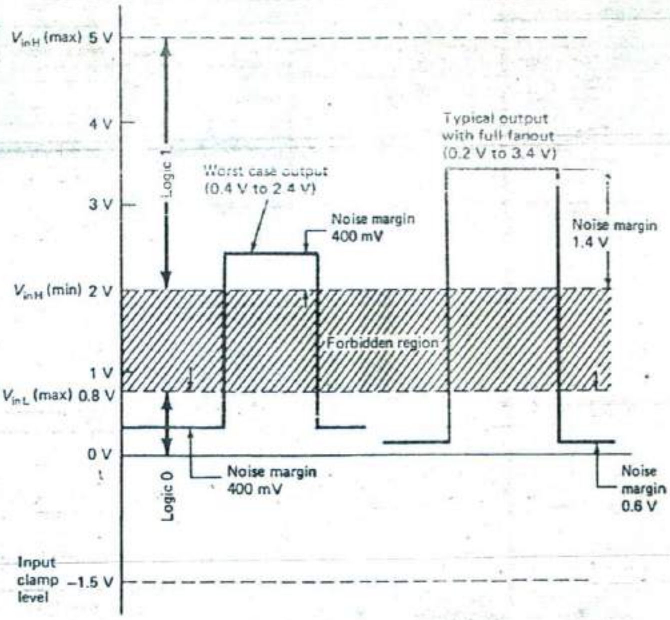


Fig. 11-1 Transistor-transistor logic (TTL) input requirements.

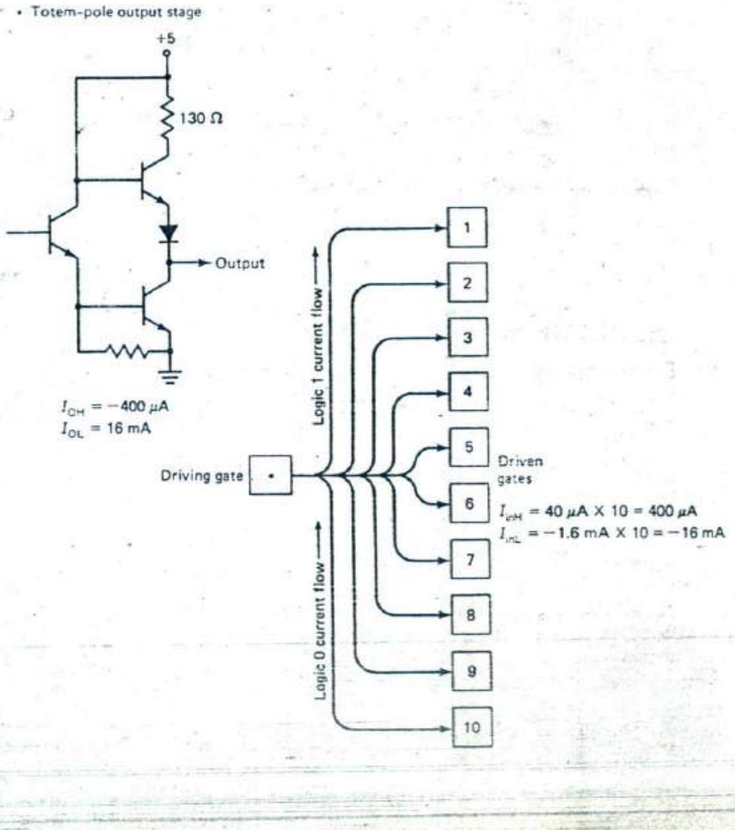


Fig. 11-2 Transistor-transistor logic (TTL) fanout.

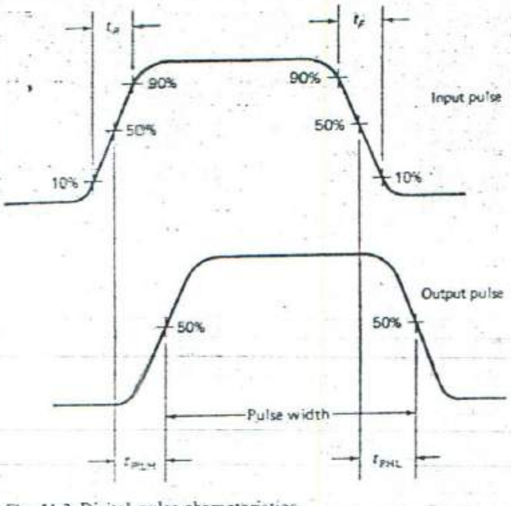


Fig. 11-3. Digital pulse characteristics.

approximately 130 Ω between the supply and the top transistor. This resistor limits the current that the output can source when it is at logic 1. The sink current, I_{OL}, is rated at 16 mA, and the source current I_{OH} is rated at only -400 μA. The negative sign indicates that the current is flowing away from the driving gate when it is sourcing current. The input requirements for each driven gate are $I_{INH} = 40$ μA and $I_{INL} = -1.6$ mA. This means that a TTL output can drive no more than 10 TTL inputs, or the guaranteed noise margin will be lost. If the maximum fanout is exceeded, the output voltage may fall in the forbidden region. This can happen when the driving gate is sourcing or sinking current.

Figure 11-3 shows the characteristics of digital pulses. *Rise time* (t_R) is the time required for the pulse to change from its 10 to its 90 percent level. *Fall time* (t_F) is the time required for the pulse to change from the 90 to the 10 percent level. When the pulse is applied to the input of a TTL device, it takes time before the output changes. The propagation delay from low to high (t_{PLH}) is measured from the 50 percent point on the input waveform to the 50 percent point on the output waveform. The propagation delay from high to low (t_{PHL}) is also measured from 50 percent point to 50 percent point. Digital circuits with short propagation delays are faster and are capable of high-frequency operation. The pulse width is measured from the 50 percent point on the rising edge of a pulse to the 50 percent point on the falling edge of the same pulse.

Table 11-1 lists a few important characteristics for the TTL logic family. This family has four subfamily members. It is important to note that the part numbers clearly identify the subfamily members. For example, a 7400 gate is a TTL device, and a 74LS00 is a *low-power Schottky* (LS) clamped device, which is a TTL subfamily. The part numbers 7490 and 74LS90 represent another example: the 7490 is a TTL device, and the 74LS90 is an LS TTL subfamily device. You may also encounter 5400 series numbers, which are military versions of the 7400 TTL family. The military devices are rated from −55°C to +125°C; commercial and industrial devices are rated from 0°C to +70°C. Different manufacturers may deviate from the 7400 numbering system, and 8000 numbers are also popular. The TTL and LS TTL devices are the most widely applied. Table 11-1 shows that the LS devices are faster (less propagation delay) than the TTL devices and use less current. The LS devices used to cost more but now are about as expensive as TTL parts, and designers choose them most often. Since they are more power-efficient and faster devices, there is usually no reason to specify TTL devices in new designs unless the LS version of the required device is not available.

All TTL devices and TTL subfamily devices are rated at a fanout of 10, except when subfamilies are mixed or when subfamily devices are mixed with TTL devices. A low-power Schottky device will drive 10 gates in its own subfamily but will drive only 5 TTL gates. There are also speed differences. Some LS devices can approach twice the speed of their TTL counterparts. If a circuit is operating near its top speed, a substitution may not work or, worse yet, may work intermittently. The current drain is also different. Multiple substitutions of standard TTL for LS TTL could overload a power supply or add to the heat build-up in a circuit. Substitutions may

TABLE 11-1 THE TTL LOGIC FAMILY

Family Name	Part Number	t_{PLH} ns	t_{PHL} ns	I_{max} mA	Comments
TTL (transistor-transistor logic)	7400	11	7	22	Being replaced by LS subfamily devices
High-speed TTL	74H00	5.9	6.2	40	High power consumption; not popular
Low-power TTL	74L00	35	31	2.04	Being replaced by CMOS family devices
Low-power Schottky clamped TTL	74LS00	5	5	4.4	Very popular; used heavily in modern designs
Schottky clamped TTL	74S00	3	3	36	Very fast; used in high-speed circuits

TABLE 11-2 COUNTING WITH SEVERAL NUMBER SYSTEMS

Decimal, Base 10	Binary, Base 2	Hexadecimal, Base 16	Octal, Base 8
0	0	0	0
1	1	1	1
2	10*	2	2
3	11	3	3
4	100	4	4
5	101	5	5
6	110	6	6
7	111	7	7
8	1000	8	10*
9	1001	9	11
10	1010	A	12
11	1011	B	13
12	1100	C	14
13	1101	D	15
14	1110	E	16
15	1111	F	17
16	10000	10*	20†
17	10001	11	21
18	10010	12	22

* Read as "one-zero," not "ten."
† Read as "two-zero," not "twenty."

be acceptable, but be sure to investigate fanout, speed, current demand, and heat. The best replacement usually has exactly the same part number.

Since digital circuits recognize only two conditions, the binary number system is used for operations involving counting, arithmetic, and for representation of analog quantities. Table 11-2 compares several number systems. The decimal number system has 10 symbols, 0 through 9. The quantity of symbols in a number system is referred to as its *base* or *radix*. Therefore, the radix of our familiar decimal system is 10. When a quantity larger than 9 must be represented, more than one symbol is used at a time. The same technique applies to *binary*, which uses only two symbols, 0 and 1. Follow the count in Table 11-2. You should determine that it is possible to represent any quantity in binary that can be represented in decimal, provided there is provision for enough 0s and 1s. Table 11-2 also shows the hexadecimal number system, which has a radix of 16. It adds the characters A through F to the familiar decimal set to provide a total of 16 symbols. Hexadecimal is a convenient shortcut when working with binary systems as we shall see. The *octal* system is also shown in the table. It has a base of 8 and is also used as a shortcut for working with binary, although hexadecimal is more popular.

Binary, hexadecimal, and octal are all weighted codes, making converting them to decimal straightforward. Table 11-3 shows how weighted códes work. Starting to the immediate left of the radix point (we call it the *decimal point* when working with decimal numbers) it is seen that the weight is the base raised to the 0 power. Any number raised to the zero power has a value of 1. Therefore, the weight of this position is always 1. Moving to the left, the next position is weighted equal to the base of the number system raised to the first power. Any number raised to the first power is equal to itself. Therefore, the weight of this position is equal to 2 in binary, 16 in hexadecimal, and 8 in octal. Moving to the left again we find the weight equal to the base raised to the second power. Therefore, the weight of this position is equal to 4 in binary, 256 in hexadecimal, and 64 in octal. The fractional parts of a number are represented by positions to the right of the radix point.

To convert a binary number to decimal, it is necessary to add up the decimal weights for each binary digit. The word *bit* is a contraction for *binary digit* and will be used from now on. Suppose we wish to convert binary 10101101 to decimal. The process begins at the far right, which is called the *least significant bit* (LSB), and progresses to the leftmost bit, which is called the *most significant bit* (MSB).

EXAMPLE 1

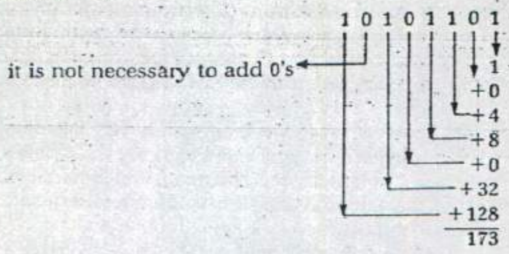
Convert binary 10101101 to decimal.

TABLE 11-3 WEIGHTED CODES

	Whole Part								Fractional Part	
Binary	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0	2^{-1}	2^{-2}
Decimal weight	128	64	32	16	8	4	2	1	$\frac{1}{2}$	$\frac{1}{4}$
Hexadecimal	16^7	16^6	16^5	16^4	16^3	16^2	16^1	16^0	16^{-1}	16^{-2}
Decimal weight	2.68×10^8	1.68×10^7	1,048,576	65,536	4096	256	16	1	$\frac{1}{16}$	$\frac{1}{256}$
Octal	8^7	8^6	8^5	8^4	8^3	8^2	8^1	8^0	8^{-1}	8^{-2}
Decimal weight	2.10×10^6	262,144	32,768	4096	512	64	8	1	$\frac{1}{8}$	$\frac{1}{64}$

↑ *Radix Point*

SOLUTION



The LSB is weighted 1 (2^0), so we begin by adding 1. The next bit position is weighted at decimal 2 (2^1), but there is a 0 there, so we add 0. The next bit is weighted at 4 (2^2), and there is a 1 there, so we add 4. The next bit is weighted at 8 (2^3), and there is a 1 in this position, so we add 8. The next bit is weighted at 16 (2^4), and there is a 0 there, so we add 0. The next position weight is 32 (2^5), and there is a 1 there, so we add 32. The next position weight is 64 (2^6), and there is a 0 there, so we add 0 (it is not necessary to add the 0s). The MSB position is weighted 128 (2^7), and there is a 1 there, so we add 128. The total is 173, which is the base 10 equivalent of binary 10101101.

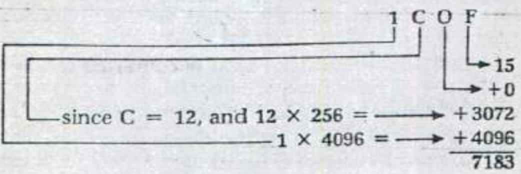
Because there may be a possibility of confusion, the base of a number may be specified with a subscript. For example, $10101101_2 = 173_{10}$.

The same general technique is used to convert hexadecimal numbers to decimal.

EXAMPLE 2

Convert hexadecimal 1COF to decimal.

SOLUTION



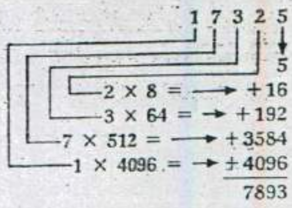
The weight of the least significant position is 1 (16^0), and there is an F there. In decimal F is equal to 15 so we add 15. The next weight is 16 (16^1), but there is a 0 there, so we can add 0 or not. The weight of the next position is 256 (16^2), and there is a C there. Since C is equal to decimal 12 we add 12 × 256, or 3072. The last position is weighted 4096 (16^3), and there is a 1 there, so we add 4096. The total is 7183. Therefore, $1COF_{16} = 7183_{10}$.

Converting from octal to decimal uses the same process and is demonstrated in Example 3:

EXAMPLE 3

Convert octal 17325 to decimal.

SOLUTION

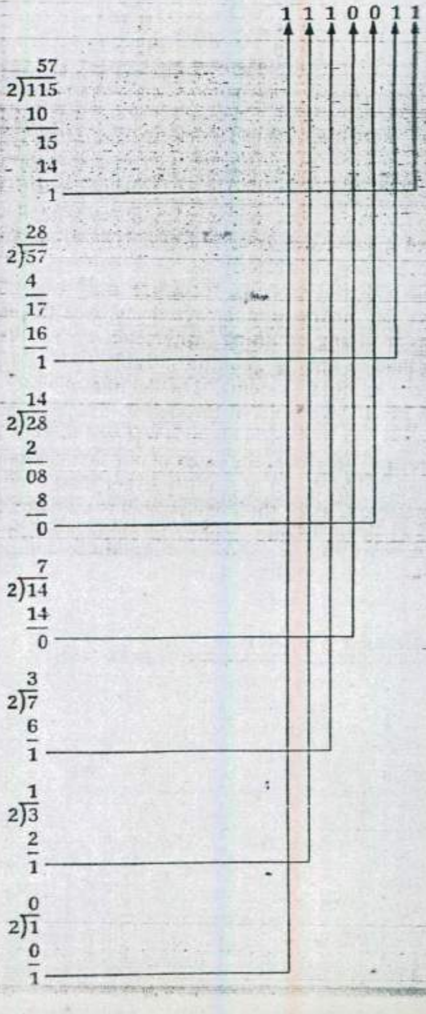


We must also be able to convert from decimal to other number systems. This process involves dividing the decimal number by the base of the given number system while keeping a record of all remainders. The division continues until the decimal number is exhausted. The list of remainders is the number in the given number system.

EXAMPLE 4

Convert decimal 115 to binary.

SOLUTION



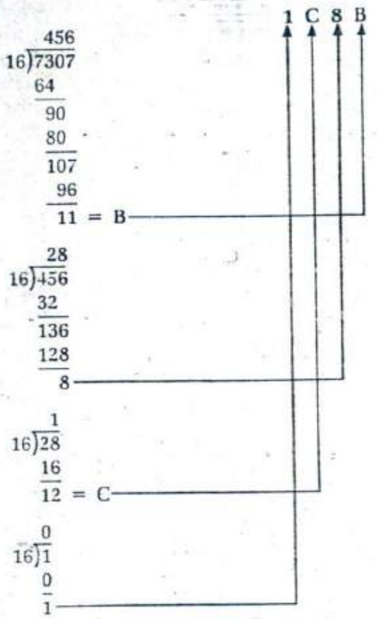
When 2 is divided into 115, the quotient is 57 with a remainder of 1. This first remainder becomes the LSB of the answer. Next 57 is divided by 2 with a quotient of 28 and a remainder of 1. This remainder becomes the next bit of the answer. Then 28 is divided by 2 with a quotient of 14 and a remainder of 0. This 0 remainder becomes the next bit of the answer. The process continues until the quotient is 0, and the last remainder has been recorded as the MSB of the answer. Therefore, $115_{10} = 1110011_2$.

Converting from decimal to hexadecimal involves repeated division by 16 until the number is exhausted. All remainders are converted to hexadecimal characters and recorded. The process ends when the quotient is 0 and the last remainder has been placed in the leftmost position.

EXAMPLE 5

Convert decimal 7307 to hexadecimal.

SOLUTION



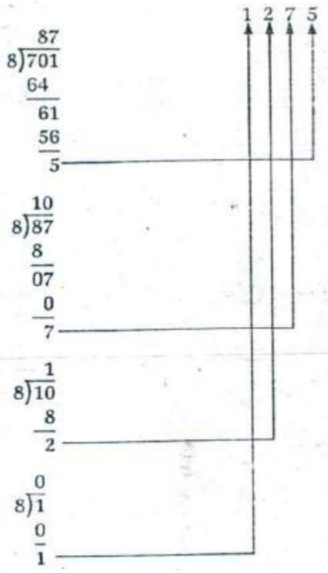
Note that the first remainder is decimal 11, which is equal to B in hexadecimal.

Hexadecimal (and occasionally octal) is used to enter information into and extract information from digital systems quickly. The trouble with binary is that too many bits are required to represent numbers of moderate size. It takes too long to enter or read a lot of 0s and 1s, and the process is error-prone. A digital system does not understand hexadecimal any better than it understands decimal. Digital systems are strictly binary in nature. However, there is a natural relationship between binary and hexadecimal that makes conversion very easy. Decimal 16 is a power of 2, and each hex character can be represented by 4 bits.

EXAMPLE 6

Convert decimal 701 to octal.

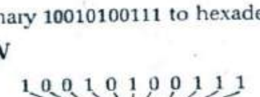
SOLUTION



EXAMPLE 7

Convert hexadecimal A4E to binary.

SOLUTION



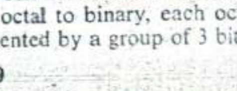
Each digit of $A4E_{16}$ is converted to a binary number as shown.

It is just as easy to convert from binary to hex.

EXAMPLE 8

Convert binary 1001010011 to hexadecimal.

SOLUTION



Group the bits into 4s, starting with the rightmost bits. Do not be concerned if the leftmost group does not have 4 bits. Convert whatever is in the leftmost group to the appropriate hex character.

Octal is also convenient since 8 is also a power of 2. To convert octal to binary, each octal character must be represented by a group of 3 bits.

EXAMPLE 9

Convert octal 725 to binary.

TABLE 11-4 BINARY CODED DECIMAL

Decimal	Binary	BCD
0	0	0000
1	1	0001
2	10	0010
3	11	0011
4	100	0100
5	101	0101
6	110	0110
7	111	0111
8	1000	1000
9	1001	1001
10	1010	0001 0000
11	1011	0001 0001
12	1100	0001 0010
13	1101	0001 0011
14	1110	0001 0100
15	1111	0001 0101
16	10000	0001 0110
17	10001	0001 0111
18	10010	0001 1000

SOLUTION

7 2 5

1 1 1 0 1 0 1 0 1

Binary to octal is the reverse process.

EXAMPLE 10

Convert binary 11010100 to octal.

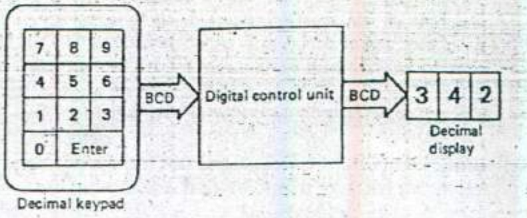


Fig. 11-4 Binary-coded decimal applications.

SOLUTION

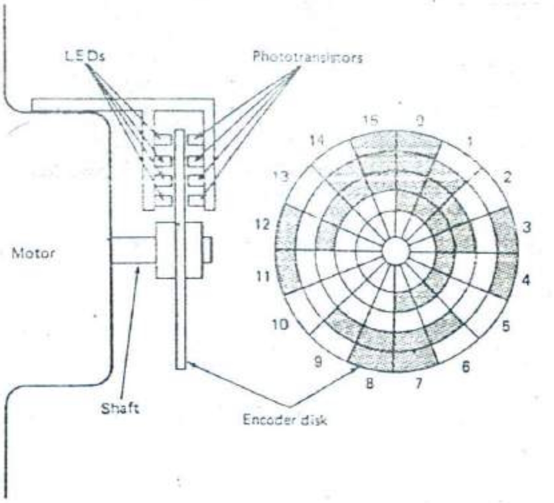
1 1 0 1 0 1 0 0

3 2 4

Hex and octal are fine for engineers, technicians, and programmers who have taken the time to learn other number systems. Operators are often not familiar with these number systems. They simply use digital equipment and do not need an understanding of the digital circuits. Figure 11-4 shows a decimal keypad connected to the input of a digital control unit. The output goes to a decimal display. This environment is far more comfortable for most people than a hexadecimal keypad or a binary display. Another code has been developed to interface between the decimal operator and the binary machine conveniently. It is called *binary-coded decimal* (BCD) and is shown in Table 11-4. It is the same as ordinary binary until numbers greater than decimal 9 are represented. These numbers require an additional group of 4 bits for each decimal digit. The decimal number 309 would require 12 bits, 1234 would require 16 bits,

TABLE 11-5 AMERICAN STANDARD CODE FOR INFORMATION INTERCHANGE

Row	Column Bits 4321 ↓ 765 →	0	1	2	3	4	5	6	7
		000	001	010	011	100	101	110	111
0	0000	NUL	DLE	SP	0	@	P	\	p
1	0001	SOH	DC1	!	1	A	Q	a	q
2	0010	STX	DC2	"	2	B	R	b	r
3	0011	ETX	DC3	#	3	C	S	c	s
4	0100	EOT	DC4	$	4	D	T	d	t
5	0101	ENQ	NAK	%	5	E	U	e	u
6	0110	ACK	SYN	&	6	F	V	f	v
7	0111	BEL	ETB	'	7	G	W	g	w
8	1000	BS	CAN	(	8	H	X	h	x
9	1001	HT	EM	)	9	I	Y	i	y
10	1010	LF	SUB	*	:	J	Z	j	z
11	1011	VT	ESC	+	;	K	[	k	{
12	1100	FF	FS	,	<	L	\	l	\
13	1101	CR	GS	—	=	M	]	m	}
14	1110	SO	RS	.	>	N	∩	n	~
15	1111	SI	US	/	?	O	—	o	DEL



Decimal	Gray
0	0000
1	0001
2	0011
3	0010
4	0110
5	0111
6	0101
7	0100
8	1100
9	1101
10	1111
11	1110
12	1010
13	1011
14	1001
15	1000

Fig. 11-5 Gray code shaft encoder.

and so on. Binary-coded decimal is not a convenient code for arithmetic operations, and it usually requires more bits than binary. For these reasons, BCD may be used for input and output operations only, and binary-to-BCD and BCD-to-binary code converters will be required.

Other special codes are required for special applications. Some are *nonweighted*, meaning that each bit position does not have a definite weight. The Gray code is an example. Figure 11-5 shows how the Gray code can be used to convert the angular position of a motor shaft, which is analog, into a group of nonweighted 0s and 1s. As the shaft turns, the encoder disk turns with it and presents a clear or an opaque sector to each pair of light-emitting diodes and phototransistors. The 4-bit code that is read from the phototransistor outputs represents one of 16 shaft positions. A five-level code would resolve 32 shaft positions, a six-level code 64 shaft positions, and so on. The Gray code is unique in that no more than 1 bit changes at a time as the shaft rotates. Examine the table in Fig. 11-5 to verify this point. A binary-encoded disk would be error-prone because more than one bit would change at a time as the shaft turned. The Gray code does not lend itself to arithmetic operations, and so code conversion to binary will be required.

An *alphanumeric code* is one that represents letters and numbers. The most popular one is the *American Standard Code for Information Interchange* (ASCII). Table 11-5 shows an ASCII code chart. The bit pattern for each row and column is given. For example, the letter *A* is in row 1. All characters in this row end with the bit pattern 0001. *A* is in column 4, and all characters in this column begin with the bit pattern 100. Therefore, the ASCII bit pattern for *A* is 1000001. Note that ASCII is a 7-bit code. Sometimes an eighth bit is added in the leftmost position to act as a parity bit. *Parity* is an error-checking technique. If the parity is supposed to be even, the letter *A* will be represented by 01000001, which has an even number of 1s in the group. If the parity is odd, the letter *A* will be represented by 11000001, which contains an odd number of 1s.

REVIEW QUESTIONS

1. A digital signal that remains at a level between 0.2 and 2.0 V is in the ______ region.

2. The worst-case noise margin in TTL logic circuits is ______.

3. Do worst-case noise margins occur with small fanouts or with full fanouts?

4. Can a totem pole output stage source or sink more current?

5. A digital pulse ranges from 0.2 to 3.4 V. What two voltage points will be used when measuring the rise time of the pulse?

6. What voltage point will be used for the pulse of question 5 when measuring propagation delay?

7. Is it always acceptable to substitute a TTL device for an LS TTL device?

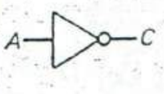
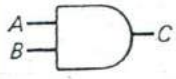
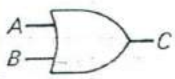
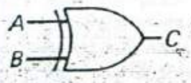
8. Is it always acceptable to substitute an LS TTL device for a TTL device?

9. Use Table 11-5 and determine the ASCII code for the letter Z in an odd parity system.

11-2 GATES AND COMBINATIONAL LOGIC

A *gate* is a decision-making element. It produces an output that is high or low, depending on its input conditions. Table 11-6 shows the basic logic gates.

TABLE 11-6 LOGIC GATES

Gate	Symbol	Truth Table			Boolean Expression
NOT		*A*	*C*		$C = \overline{A}$
		0	1		
		1	0		
AND		*A*	*B*	*C*	$C = A \cdot B$
		0	0	0	
		1	0	0	
		0	1	0	
		1	1	1	
INCLUSIVE OR		*A*	*B*	*C*	$C = A + B$
		0	0	0	
		1	0	1	
		0	1	1	
		1	1	1	
EXCLUSIVE OR		*A*	*B*	*C*	$C = A\overline{B} + B\overline{A}$ $C = A \oplus B$
		0	0	0	
		1	0	1	
		0	1	1	
		1	1	0	

The *NOT gate,* shown at the top of the table, is also called an *inverter.* The truth table for the NOT gate shows that the input (*A*) can be low (0) or high (1). When the input is low, the output (*C*) is high. When the input is high, the output is low. This function is known as *logical inversion.* The *NOT symbol* is a triangle with a circle at the output. The circle is important because it tells us that the output is inverted. *Boolean algebra* is a special branch of mathematics used to describe and design binary systems. The boolean expression for the NOT gate is read as *C is equal to NOT A*. The bar over the *A* is an inversion bar. The NOT gate is useful in many situations. Suppose, for example, that a limit switch produces a logic 0 when activated, but it would be more convenient if it produced a logic 1. A NOT gate can be used to invert the switch logic.

Now look at the AND gate in Table 11-6. It has two inputs and its truth table has four rows, one for each possible input combination. The output is high when input A and input B are high. The boolean expression is read *C is equal to A AND B*. The dot between the A and the B is the symbol for the AND operator. The dot is optional; thus C = AB is an equivalent expression, and so is C = A(B). The AND gate is an "all or nothing" circuit and is useful for activating a load when all of the inputs are high. For example, it may be desirable to activate an alarm circuit when both the pressure and the temperature of some process go high.

Table 11-6 shows two types of OR gates. The *inclusive OR gate* is often simply referred to as an *OR gate*. However, to avoid confusion, the other version must be referred to by its full name, *exclusive OR.* The inclusive OR is an "any or all" gate and produces a high output when either or both of its inputs are high. The *exclusive OR gate* produces a high output when either A or B is high but excludes the condition where both are high. The inclusive OR gate includes a high output for the condition where both inputs are high. The boolean expression for the inclusive OR gate is read *C is equal to A OR B*. The plus (+) sign is the OR operator, but you should ***not*** read the expression as *C is equal to A plus B*. The inclusive OR is useful for detecting those situations in which any or all of its inputs are high. For example, it may be desirable to activate an alarm circuit when the pressure is high, or the temperature is high, or both conditions are true. The exclusive OR gate is useful for detecting inequality. Note that it produces a high output only when its inputs are not equal. It may also be used as a comparator because its output is low when its inputs are equal. The boolean expression for the exclusive OR gate is read *C is equal to A AND NOT B OR B AND NOT A*. This is often shortened as Table 11-6 shows by using the

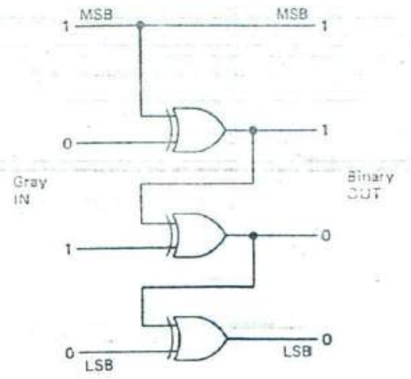
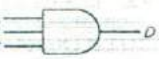


Fig. 11-6 Gray-to-binary code converter.



$D = A \cdot B \cdot C$

A	*B*	*C*	*D*
0	0	0	0
1	0	0	0
0	1	0	0
1	1	0	0
0	0	1	0
1	0	1	0
0	1	1	0
1	1	1	1

Fig. 11-7 Three-input AND gate.

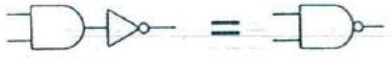


Fig. 11-8 NAND gate.

ring sum symbol, which is the exclusive OR operator. The expression is read *C is equal to A exclusive OR B*. It can also be read *C is equal to A ring sum B*.

The last section introduced the Gray code. It is a nonweighted code that is often used to encode shaft position. Unfortunately, since it is nonweighted, it is not useful when arithmetic operations are required. Figure 11-6 shows a Gray-to-binary code converter that uses exclusive OR gates. Suppose that Gray code 1010 is presented to the converter. The MSB is a 1 and connects straight through to the output. The next bit is exclusively ORed with the MSB and produces an output of 1. This 1 is exclusively ORed with the next Gray bit, which is also a 1, producing the next output of 0. This 0 is exclusively ORed with the last Gray bit, which is also a 0; the LSB of the output is 0. By following the decision of each gate, you should be able to verify that the output is equal to binary 1100 or decimal 12. Now, refer again to Fig. 11-5. Try a couple of other shaft positions to test your understanding of the converter circuit.

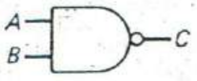
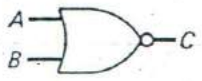
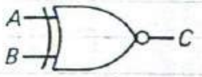
Except for the inverter, logic gates can have more than two inputs. Figure 11-7 shows a three-input AND gate. There are eight possible input combinations, and the truth table has a row for each. The number of combinations is predicted by 2^N, where N is equal to the number of inputs. A four-input gate has 16 input combinations, a five-input gate has 32 input combinations, and so on. Note that only one input combination produces a high output in Fig. 11-7. This occurs when A, B, and C are high.

Gate outputs are often inverted (refer to Fig. 11-8). This can be accomplished by following a gate with an inverter. More commonly, it is obtained in one package. The symbol on the right is for a *NOT-AND gate* or simply *NAND gate*. Note the inversion circle on the NAND output. Table 11-7 shows the symbols, truth tables, and boolean expressions for the inverted output gates. The truth tables show that the output conditions are inverted from those shown in Table 11-6. The boolean expressions show the inversion bar over the terms and their operators.

Figure 11-9 shows some of the laws of combination for gates. These laws are useful because they will help you to understand how any logic function can be synthesized from other logic functions. For example, the *law of association* shows us how to obtain a three-input AND function from two-input AND gates. It also shows that it is possible to obtain a three-input OR function from two-input OR gates. Simply stated, the associative law tells us that it makes no difference as to the order of how AND or OR expressions are combined. The *law of distribution* shows that there can be two forms of an expression and a circuit can be realized for each. In general, the circuit containing the fewest number of gates is the most desirable. The distributive law is useful when examining an expression to determine whether it can be simplified. For example, as Fig. 11-9 shows, a logic function may be implemented with two or three gates and have the same function. The *law of tautology* shows that when a variable is ORed or ANDed with itself the result is equal to the variable. Taking this one step further, if the gate is a NOR or a NAND, an inverter is realized by tying the inputs together. The *law of double complementation* shows that two inversions cancel and that the variable is restored.

The last two rows of Fig. 11-9 illustrate *DeMorgan's theorem*, which states that the complement of a sum is equal to the product of the complements. Likewise, the complement of a product is equal to the sum of the complements. The word *complement* means the inversion of a variable or expression. The complement of A means the same

TABLE 11-7 INVERTED OUTPUT GATES

Gate	Symbol	Truth Table			Boolean Expression
		A	B	C	
NAND		0	0	1	$C = \overline{A \cdot B}$
		1	0	1	
		0	1	1	
		1	1	0	
		A	B	C	
INCLUSIVE NOR		0	0	1	$C = \overline{A + B}$
		1	0	0	
		0	1	0	
		1	1	0	
		A	B	C	
EXCLUSIVE NOR		0	0	1	$C = \overline{A \oplus B}$
		1	0	0	
		0	1	0	
		1	1	1	

thing as NOT A. Simply stated, the theorem tells us we can break an inversion bar over an operator if we change the operator. For example, in the equation $\overline{A + B} = \overline{A} \cdot \overline{B}$, the bar is broken over the OR operator and the operator is changed to AND. The theorem shows that it is possible to synthesize a NOR gate from four NAND gates. Figure 11-9 shows that the first two NAND gates are used as inverters since their inputs are tied together. The next gate NANDs the complemented inputs. The theorem shows that the complements are to be ANDed, and the last NAND gate acts as an inverter to meet that requirement because when a NAND gate is followed by an inverter the AND operation results. An OR function can therefore be realized by leaving off the last inverter. The last row of Fig. 11-9 shows that the NAND function can be synthesized from four NOR gates. The complement of A is NORed with the complement of B and then inverted. The AND function is realized by leaving off the last inverter. NAND gates are often referred to as *universal logic elements* because it is possible to synthesize any logic function by combining them. The same thing is true of NOR gates.

DeMorgan's theorem has led to alternate symbols for NAND and NOR gates. Figure 11-10 shows these alternate symbols. The NAND gate is sometimes represented as an OR gate with inverted inputs. The NOR gate may be represented as an AND gate with inverted inputs. You may find both the standard and the alternate symbols used on the same logic diagram. The purpose of using both is to represent circuit operation more clearly by emphasizing high inputs or inverted (low) inputs.

The *laws of combination* are also useful to solve circuits. For example, Fig. 11-11 shows a combinational logic circuit based on four NAND gates. What is the output supposed to be? Follow the steps:

1. At this point we find variable A combined with the NAND of variables A and B.
2. It is desired to simplify this so the bar is broken, and the AND operator is changed to the OR operator (DeMorgan's theorem).
3. The law of double complementation allows us to remove the bars.

4–6. These steps are the same as the first three.

7. The simplified terms from steps 3 and 6 are NANDed.
8. The bar is broken in the middle, and the operator is changed to OR.
9. The bar is broken at the left and at the right, and the operators are changed.
10. The double bars are removed.
11. The term is multiplied (ANDed) with each expression in the parentheses. Two terms cancel, since any time a variable is ANDed with its complement the result is 0. You may verify this by looking at the AND gate truth table.

12–13. The result is the exclusive-OR operation.

Another basic understanding that you will need to work with digitial systems is in the area of dynamic

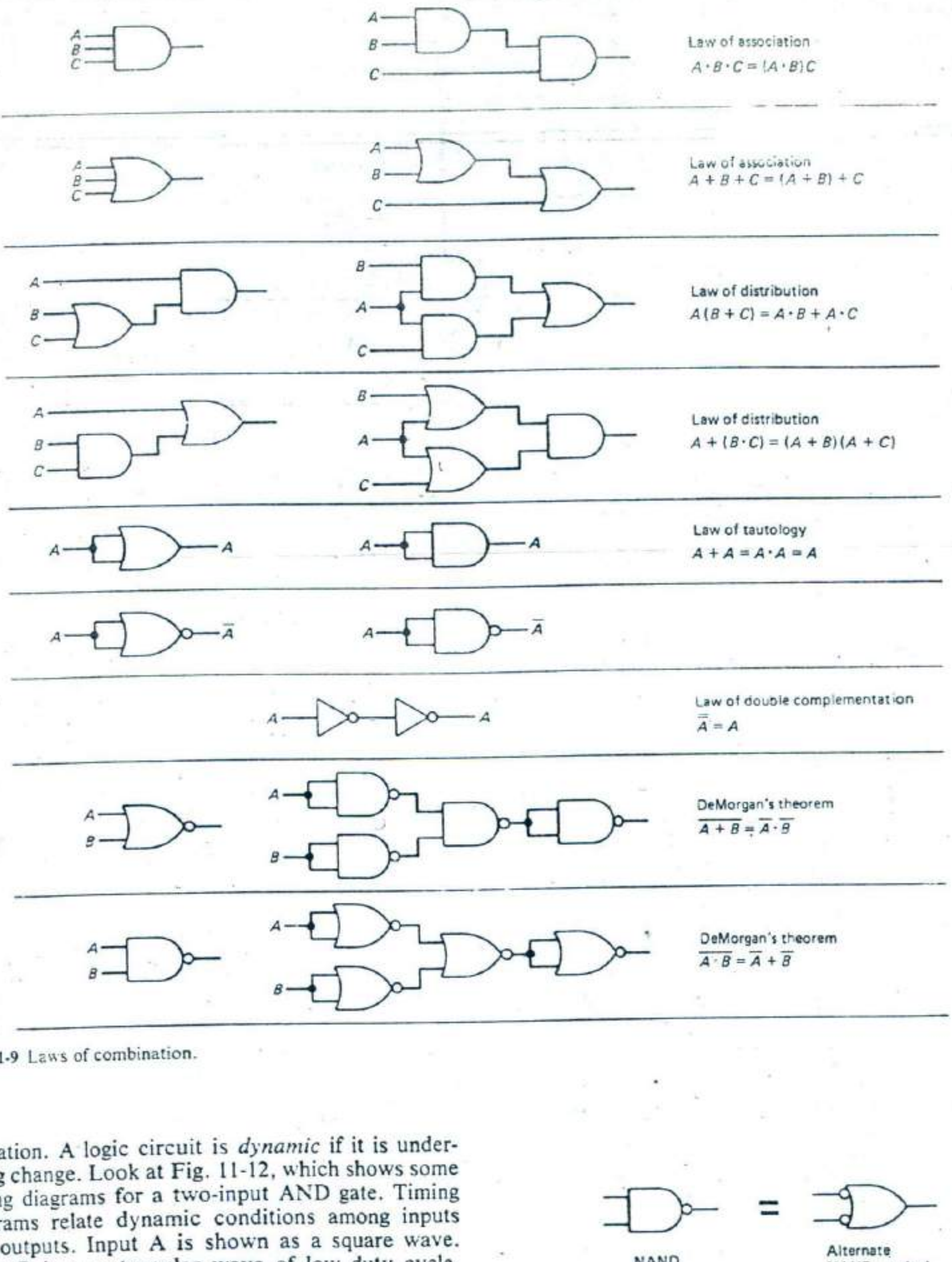

Law of association
$A \cdot B \cdot C = (A \cdot B)C$
Law of association
$A + B + C = (A + B) + C$
Law of distribution
$A(B + C) = A \cdot B + A \cdot C$
Law of distribution
$A + (B \cdot C) = (A + B)(A + C)$
Law of tautology
$A + A = A \cdot A = A$
Law of double complementation
$\overline{\overline{A}} = A$
DeMorgan's theorem
$\overline{A + B} = \overline{A} \cdot \overline{B}$
DeMorgan's theorem
$\overline{A \cdot B} = \overline{A} + \overline{B}$

Fig. 11-9 Laws of combination.

operation. A logic circuit is *dynamic* if it is undergoing change. Look at Fig. 11-12, which shows some timing diagrams for a two-input AND gate. Timing diagrams relate dynamic conditions among inputs and outputs. Input A is shown as a square wave. Input B is a rectangular wave of low duty cycle. Output C is high only during those periods when both A and B are high. Now look at Fig. 11-13; it shows the timing diagram for an exclusive OR gate. Note that the output waveform is the inverse of input A when input B is high. When B is low, the output is in phase with input A. Verify this operation by looking at the exclusive OR truth table if necessary. The exclusive OR gate is sometimes called a *program-*

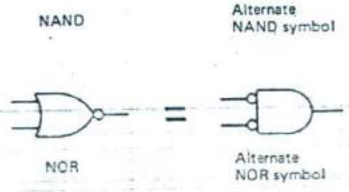


Fig. 11-10 Alternate symbols.

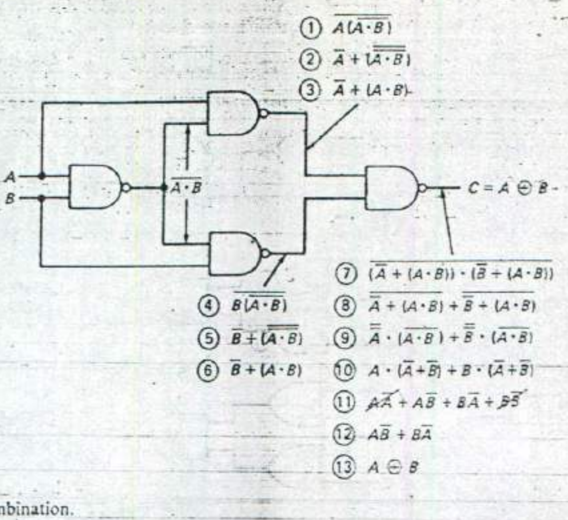

A
B
$\overline{A \cdot B}$
$C = A \ominus B$
① $\overline{A(\overline{A \cdot B})}$
② $\overline{A} + (\overline{\overline{A \cdot B}})$
③ $\overline{A} + (A \cdot B)$
④ $\overline{B(\overline{A \cdot B})}$
⑤ $\overline{B} + (\overline{\overline{A \cdot B}})$
⑥ $\overline{B} + (A \cdot B)$
⑦ $\overline{(\overline{A} + (A \cdot B)) \cdot (\overline{B} + (A \cdot B))}$
⑧ $\overline{\overline{A} + (A \cdot B)} + \overline{\overline{B} + (A \cdot B)}$
⑨ $\overline{\overline{A}} \cdot (\overline{A \cdot B}) + \overline{\overline{B}} \cdot (\overline{A \cdot B})$
⑩ $A \cdot (\overline{A} + \overline{B}) + B \cdot (\overline{A} + \overline{B})$
⑪ $\cancel{A\overline{A}} + A\overline{B} + B\overline{A} + \cancel{B\overline{B}}$
⑫ $A\overline{B} + B\overline{A}$
⑬ $A \ominus B$

Fig. 11-11 Using the laws of combination.

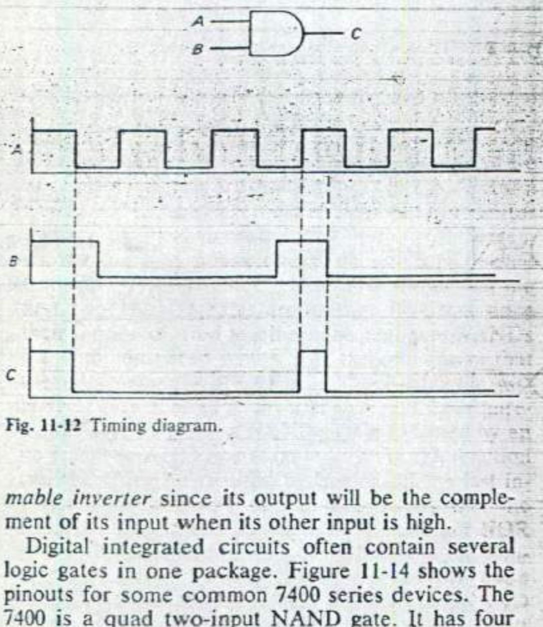


Fig. 11-12 Timing diagram.

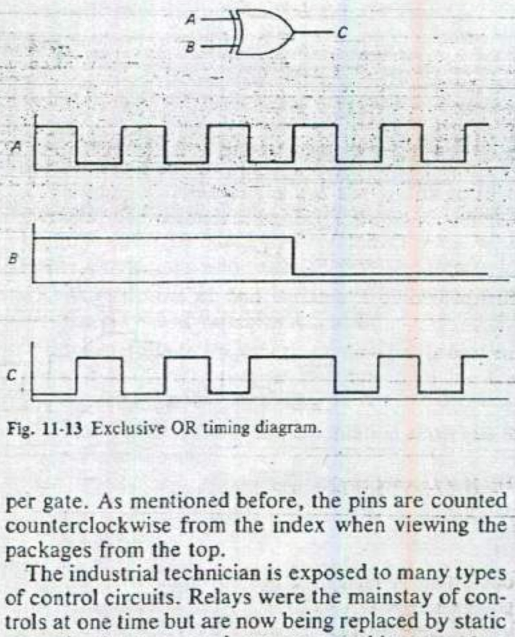
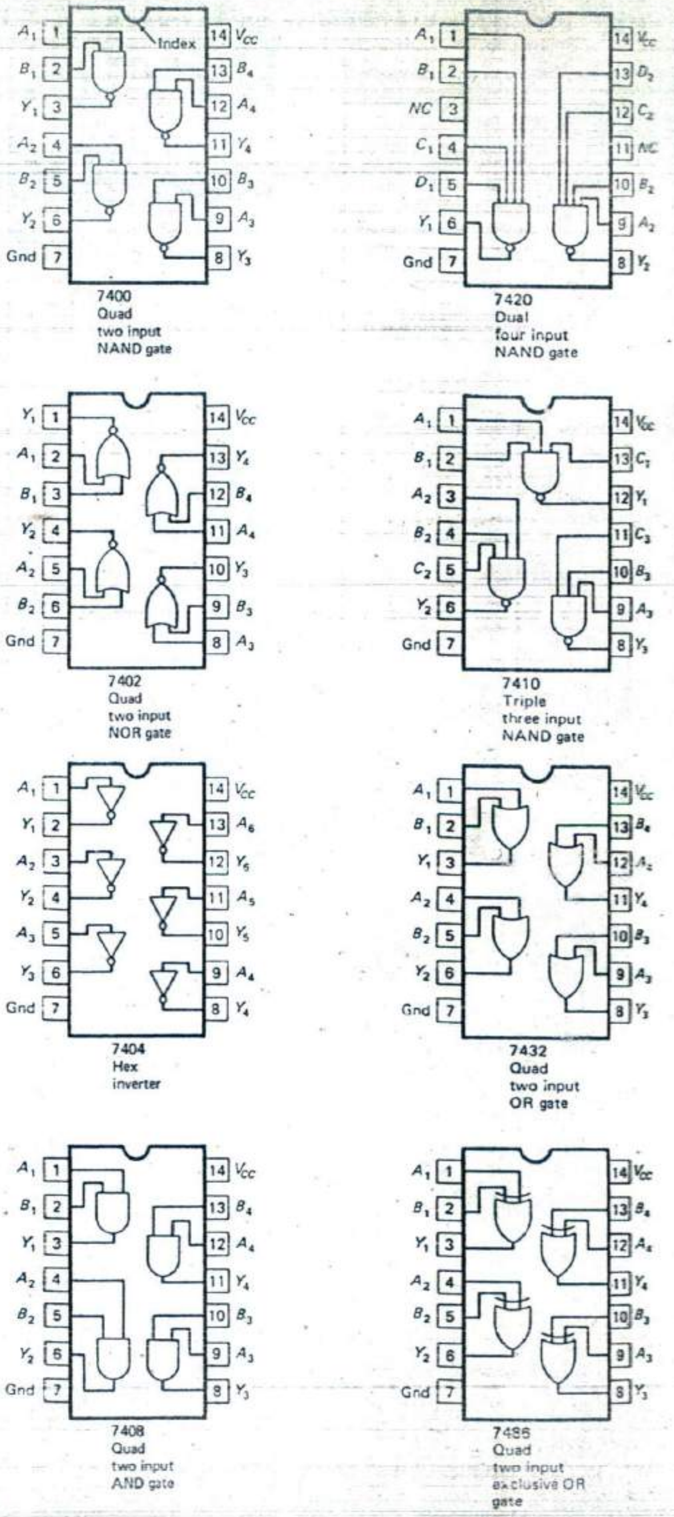


Fig. 11-13 Exclusive OR timing diagram.

mable inverter since its output will be the complement of its input when its other input is high.

Digital integrated circuits often contain several logic gates in one package. Figure 11-14 shows the pinouts for some common 7400 series devices. The 7400 is a quad two-input NAND gate. It has four gates per package. The 7404 is called a *hex inverter* since it has six gates per package. The 7420 contains only two NAND gates because each one has four inputs and there are only 14 pins available. In the TTL family and subfamilies V_{CC} is +5 V and is applied to pin 14 of each package. Pin 7 is grounded in each case. This is not a uniform standard, however. Some 7400 devices have different ground and supply pins, and some have more than 14 pins. The output pins are labeled *Y* so they are not confused with the input pins, which are labeled *A* through *D* and beyond, depending upon how many inputs there are per gate. As mentioned before, the pins are counted counterclockwise from the index when viewing the packages from the top.

The industrial technician is exposed to many types of control circuits. Relays were the mainstay of controls at one time but are now being replaced by static controls. *Static controls* are so named because they have no moving parts. Figure 11-15 compares static controls with relay controls. The American National Standard Institute (ANSI) logic symbols have gained widespread acceptance, but the National Electrical Manufacturer's Association (NEMA) logic symbols are found on many industrial schematics and wiring diagrams. The NOT function can be achieved with a gate or by using a relay with normally closed contacts. The ladder diagram for the AND function shows the control contacts in series so all will have to be closed to energize the relay contacts. The OR



Index
7400 Quad two input NAND gate
A_1 1, B_1 2, Y_1 3, A_2 4, B_2 5, Y_2 6, Gnd 7, 14 V_{CC}, 13 B_4, 12 A_4, 11 Y_4, 10 B_3, 9 A_3, 8 Y_3
7420 Dual four input NAND gate
A_1 1, B_1 2, NC 3, C_1 4, D_1 5, Y_1 6, Gnd 7, 14 V_{CC}, 13 D_2, 12 C_2, 11 NC, 10 B_2, 9 A_2, 8 Y_2
7402 Quad two input NOR gate
Y_1 1, A_1 2, B_1 3, Y_2 4, A_2 5, B_2 6, Gnd 7, 14 V_{CC}, 13 Y_4, 12 B_4, 11 A_4, 10 Y_3, 9 B_3, 8 A_3
7410 Triple three input NAND gate
A_1 1, B_1 2, A_2 3, B_2 4, C_2 5, Y_2 6, Gnd 7, 14 V_{CC}, 13 C_1, 12 Y_1, 11 C_3, 10 B_3, 9 A_3, 8 Y_3
7404 Hex inverter
A_1 1, Y_1 2, A_2 3, Y_2 4, A_3 5, Y_3 6, Gnd 7, 14 V_{CC}, 13 A_6, 12 Y_6, 11 A_5, 10 Y_5, 9 A_4, 8 Y_4
7432 Quad two input OR gate
A_1 1, B_1 2, Y_1 3, A_2 4, B_2 5, Y_2 6, Gnd 7, 14 V_{CC}, 13 B_4, 12 A_4, 11 Y_4, 10 B_3, 9 A_3, 8 Y_3
7408 Quad two input AND gate
A_1 1, B_1 2, Y_1 3, A_2 4, B_2 5, Y_2 6, Gnd 7, 14 V_{CC}, 13 B_4, 12 A_4, 11 Y_4, 10 B_3, 9 A_3, 8 Y_3
7486 Quad two input exclusive OR gate
A_1 1, B_1 2, Y_1 3, A_2 4, B_2 5, Y_2 6, Gnd 7, 14 V_{CC}, 13 B_4, 12 A_4, 11 Y_4, 10 B_3, 9 A_3, 8 Y_3

Fig. 11-14 Pinouts for some common logic packages.

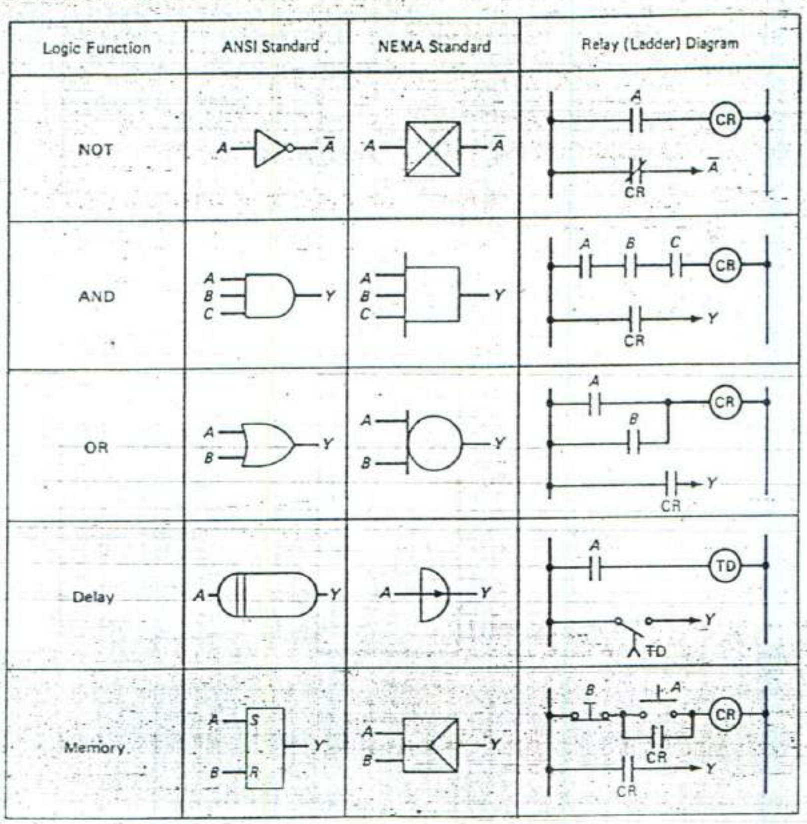


Fig. 11-15 A comparison of static and relay controls.

function is realized by wiring the control contacts in parallel. A time delay relay can be replaced with a static device such as an NE555 timer. The memory function is based on a static device such as a latch or uses a second set of relay contacts to hold the circuit on until it is reset by switch B. Latches are discussed in the next section of this chapter.

REVIEW QUESTIONS

10. Which gate acts as an all or nothing decision element?

11. If a gate is simply referred to as *OR*, which type is it?

12. Will the circuit of Fig. 11-6 produce the correct output instantaneously? Why?

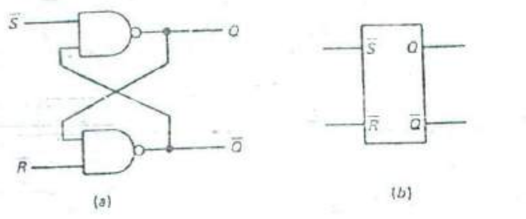
13. Calculate the total propagation delay for Fig. 11-6 if the circuit is built by using standard TTL components.

14. Refer to Fig. 11-12. What will waveform *C* look like if waveform *B* is constant at logic 1?

11-3 LATCHES AND FLIP-FLOPS

A *latch* is a sequential logic element. It has a memory characteristic that makes it useful for storing events and binary numbers. Figure 11-16(*a*) shows an *R-S* latch based on two NAND gates. The *R* is the reset input, and the *S* is the set input. The inputs are activated by logic low signals. This is why the standard symbol shown in Fig. 11-16(*b*) of the illustration indicates NOT S and NOT R. The latch has two outputs: *Q* and *NOT Q*. When Q = 0, NOT Q = 1, and this is the *reset condition* of the latch. When Q = 1, NOT Q = 0, and this is the *set condition* of the latch. Figure 11-16(*c*) of the illustration shows the timing diagram. Starting at the left, the latch is initially in the reset condition since Q is low. Moving to the right, the set input goes momentarily low. The Q output responds by going high, and the latch is now set. Moving more to the right, the reset input goes momentarily low, and the Q output goes low again. The latch has been reset.

The *R-S* latch works because of feedback. Look



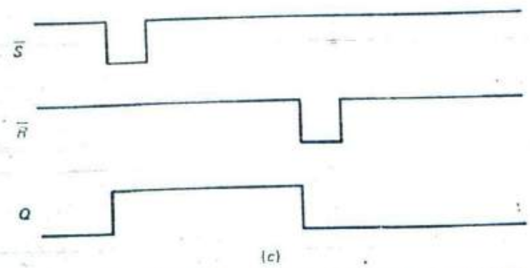


Fig. 11-16 Low activated *R-S* latch. (*a*) *R-S* latch from NAND gates. (*b*) Symbol. (*c*) Timing diagram.

at Fig 11-16(*a*). The output of each NAND gate feeds back to the input of the other gate. Recall that the truth table for the NAND gate shows a logic high at the output for all input conditions except the one where both inputs are high. Assume that the latch is in the reset conditioh (Q = 0 and NOT Q = 1) and that both inputs are high. This means that the top NAND gate has two high inputs and its output must be low. The bottom NAND gate will have one high input and one low input, and its output must be high. The truth tables are satisfied, and the circuit is stable in this condition. Now, suppose the set input goes low. The top gate now has one low input and one high input, and its output goes high. This high is fed back to the bottom gate, so that it now has both inputs high and its output goes low. Note that the latch is now in the set state: Q = 1 and NOT Q = 0. The low output from the bottom gate feeds back to the top gate, and it now has both inputs low. This changes nothing, and the top gate still has a high output. When the set pulse ends, the top gate has one high input and one low input and there is no change. The latch is now stable in the set condition. Because of the feedback, it remembers that it was set and will stay in the set mode until a reset pulse comes along or the circuit is powered down.

To analyze digital circuits that use feedback, begin with the circuit in some stable condition. Verify that the truth tables are correct for each gate. Then, introduce a change and follow the change through by using the truth table for each gate. Use this technique now with the circuit of Fig. 11-16(*a*). Start with the latch in the set condition and with both inputs high. Verify the truth tables. Then apply a reset pulse (logic 0) and trace the changes through the circuit. Prove to yourself that the outputs change as expected and that the circuit remains reset after the reset pulse is removed.

Simple *R-S* latches work very well, but they are susceptible to being forced to an illegal output state. They are also capable of settling to an unpredictable output state in some cases. For example, what would happen in Fig. 11-16(*a*) if logic 0 pulses were applied to the inputs at the same time? Both NAND gates would have high outputs. Both Q and NOT Q would be high at the same time. This is illegal, since a variable and its complement cannot be the same. Then, what would happen if both pulses returned to logic 1 at the same time? Both gates would "race" to have a low output. The gate with the shorter propagation delay would win the race and force the other gate to have a high output. The latch would resume a legal output state, in either the set mode or the reset mode. Such unpredictable behavior is unacceptable in digital circuits. This condition is known as *pulse race* or simply *race* and must be avoided.

Figure 11-17 shows an application for the *R-S* latch. It is being used to debounce a mechanical switch. When switch contacts close, they bounce. They make and break rapidly until the bouncing stops. This process lasts for several milliseconds and can create dozens of extra pulses in a circuit. The *bounceless switch circuit* avoids this problem because when the switch is thrown, it grounds either the set or the reset input of the latch, so the latch is either set or reset. As the contacts bounce open and closed, there is no change in the latch because once it is set it is not affected by subsequent set pulses. Likewise, once it is reset it is not affected by subsequent reset pulses. The *pull-up resistors* ensure a logic high at the inputs when not grounded by the switch. The circuit will work without the resistors if the latch is a TTL family device since the inputs float high. However, it is considered good practice to use the resistors.

Figure 11-18 shows an *R-S* latch configured from two NOR gates. Note that the Q output is taken from the bottom gate and the NOT Q output is taken from the top gate. The symbol, shown in Fig. 11-18(*b*), locates the Q output on top because this is its stan-

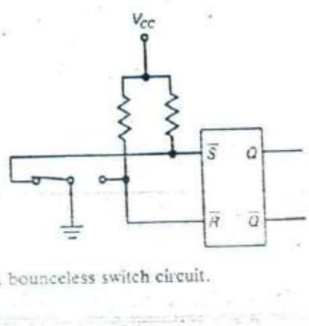


Fig. 11-17 A bounceless switch circuit.

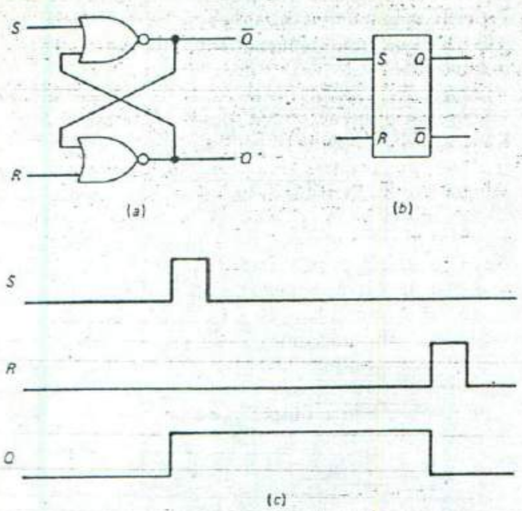


Fig. 11-18 High activated *R-S* latch. (*a*) *R-S* latch from NOR gate. (*b*) Symbol. (*c*) Timing diagram.

dard position. It is activated by logic high signals applied to the set and reset inputs. Remember that the output of a NOR gate is low if any or all of its inputs are high. Study the timing diagram and verify circuit operation. The NOR latch is susceptible to the same problems as the NAND latch. If both inputs are driven high, the outputs will assume an invalid condition: both Q and NOT Q will be at logic 0. Then, if both inputs go low at the same time, the outputs will race, and the latch will settle into the set or reset condition.

It is often desirable to enable a latch only during a specified period of time. Figure 11-19 illustrates a clocked *R-S* latch with asynchronous preset and clear inputs. Note that Fig. 11-19(*a*) shows two AND gates at the front end of the latch. These gates share a common input line called the *clock* or *enable input*. It does not matter what is happening at the set and reset inputs if the clock input is low, since all inputs to an AND gate must be high for the output to go high. Study the timing diagram in Fig. 11-19(*c*). Start at the left, where the latch is initially in the reset condition (Q is low). The clock signal goes high, enabling both inputs. Next, the set input goes high and the latch is set. Then the clock signal goes low. Now a reset pulse comes along, but the latch is not reset because the inputs are disabled. The next reset pulse does reset the latch since it occurs when the clock is high. That time when the clock is high is often referred to as a *window*. Input pulses must be synchronized with the clock window to have any effect on the output; the *S* and *R* inputs are considered to be synchronous inputs for this reason. The preset and clear inputs are asynchronous since they can be applied at any time to set or clear the latch. The asynchronous inputs are often used to initialize the circuits to a known condition after power-on.

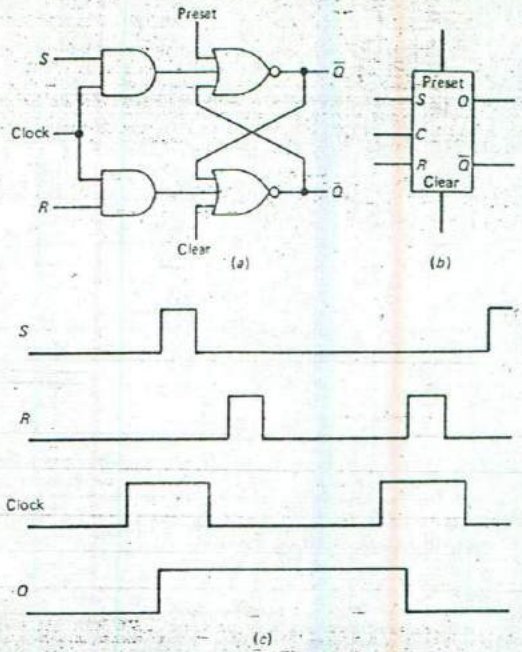


Fig. 11-19 Clocked *R-S* latch with asynchronous preset and clear. (*a*) Gate implementation. (*b*) Symbol. (*c*) Timing diagram.

Figure 11-20 shows a clocked *D* latch. The *D* input is the only synchronous input. An inverter has been added to supply inverted data to the bottom gate. If the *D* input is high during the clock window, a logic high is applied to the top NOR gate, and a logic low is applied to the bottom NOR gate. The latch is driven to the set condition if it hasn't already been set. If the *D* input is low during the window, a high will be applied to the bottom NOR gate; it will reset the latch if it wasn't previously in that mode. The latch is said to be *transparent* because the output follows the data input during the time that the clock is high. The advantage of the *D* latch is that race has been eliminated at the synchronous input. It is also no longer possible to force the outputs to an illegal mode since there is only one input. However, it is still possible to race the latch and force the outputs to an illegal mode with the asynchronous preset and clear inputs.

Now it is time to look at *flip-flops;* these are circuits with much in common with latches. In fact, many people use the term *flip-flop* to describe any bistable circuit, including the latches studied to this point. However, the digital IC manufacturers generally reserve the term for devices that trigger on a clock edge. Latches are level-sensitive, by comparison. Figure 11-21 shows the circuit, symbol, and the timing diagrams for a *D* flip-flop. The circuit uses a combination of input latches with the familiar NAND output latch. The *D* input is sampled only during the

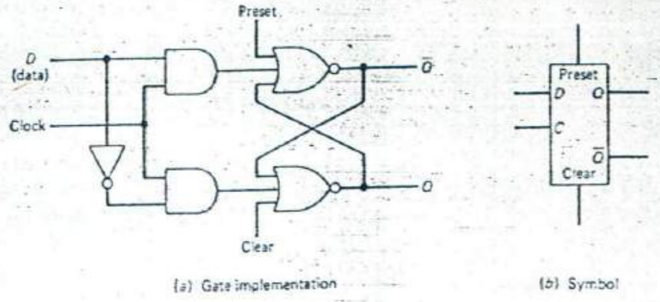


Fig. 11-20 Clocked *D* latch with asynchronous preset and clear.

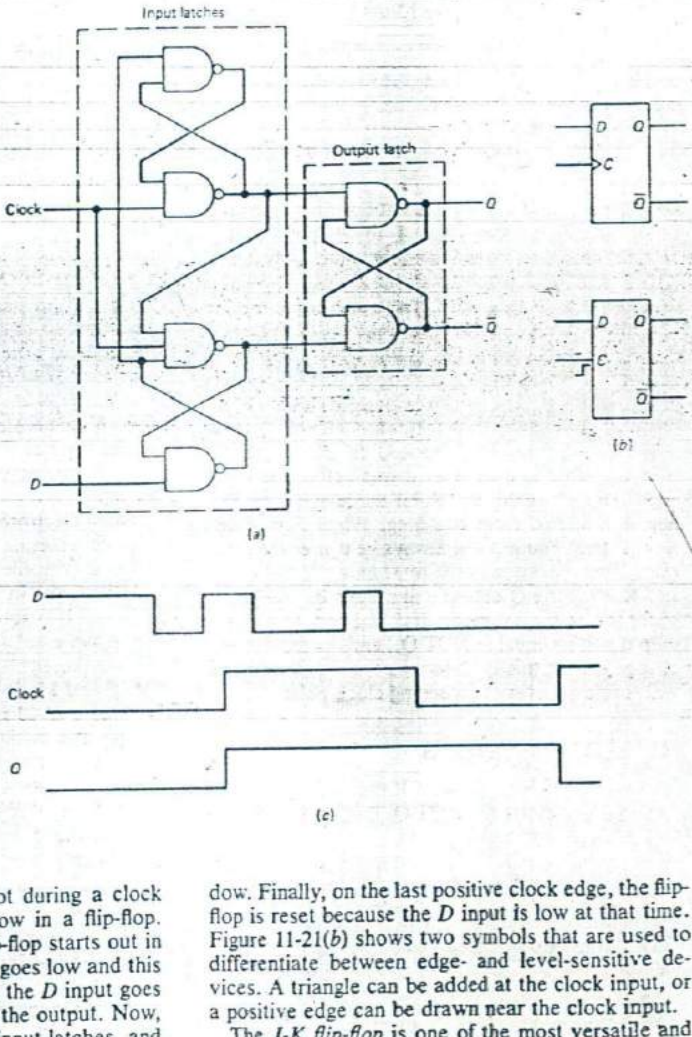
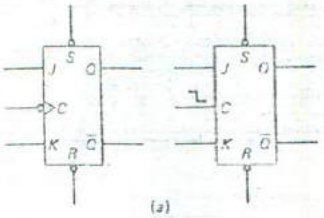


Fig. 11-21 The *D* flip-flop. (*a*) Gate implementation. (*b*) Symbols. (*c*) Timing diagram.

positive edge of the clock and not during a clock window. There is no clock window in a flip-flop. Study the timing diagram. The flip-flop starts out in the reset mode. Next, the *D* input goes low and this has no effect on the output. Next, the *D* input goes high, with no immediate effect on the output. Now, the clock goes high, enabling the input latches, and the *D* input is sampled at this time. Note that the *D* input goes low again while the clock is still high with no effect on the output; again there is no clock window. Finally, on the last positive clock edge, the flip-flop is reset because the *D* input is low at that time. Figure 11-21(*b*) shows two symbols that are used to differentiate between edge- and level-sensitive devices. A triangle can be added at the clock input, or a positive edge can be drawn near the clock input.

The *J-K flip-flop* is one of the most versatile and popular of all sequential logic circuits. Figure 11-22 depicts two symbols and a truth table for the circuit. Most J-K flip-flops are negative-edge devices. They



(a)

J	K	Q_{n+1}	Mode
0	0	Q_n	Inhibit
0	1	0	Reset
1	0	1	Set
1	1	$\overline{Q}_n$	Toggle

(b)

Fig. 11-22 Negative edge *J-K* flip-flop with asynchronous set and reset. (*a*) Symbols. (*b*) Truth table.

sample the J and the K inputs at the time when the clock is falling from logic 1 to logic 0. Figure 11-22(*a*) shows an inversion circle at the clock input, and Figure 11-22(*c*) shows the alternate symbol, which uses a negative edge. The truth table in Figure 11-22(*b*) requires some explanation. It uses a notation that refers to Q_n and to Q_{n+1}. The Q_n refers to the Q output just before the negative edge, and Q_{n+1} refers to the Q output just after the negative edge. When $J = 0$ and $K = 0$, the truth table shows that the output is Q_n. This simply means that Q did not change. It is in the same condition as it was before the clock edge. This is called the inhibit mode since the flip-flop is inhibited from changing. When $J = 0$ and $K = 1$, the Q output will always be 0 after the clock edge. This is known as the *reset mode*. When $J = 1$ and $K = 0$, the Q output will always be 1 after the edge; this is the *set mode*. If $J = 1$ and $K = 1$, the output will be equal to NOT Q_n, and the flip-flop will change states; this is known as the *toggle mode*. It will toggle on every succeeding clock pulse.

REVIEW QUESTIONS

15. Refer to Fig. 11-16(*c*). If the diagram included a waveform for NOT Q, how would it compare with the Q waveform?

16. Refer to Fig. 11-16(*c*). If a second set pulse followed the first set pulse and preceded the reset pulse, how would the Q waveform change?

17. Refer to Fig. 11-17. What are the resistors called?

18. Refer to Fig. 11-18. What output state results if both the set and reset inputs are high? What is the condition called if both inputs go low at the same time?

19. Refer to Fig. 11-19. What will happen if the set and reset inputs race when the clock is at logic 0?

20. Refer to Fig. 11-20. If the latch portion of the circuit were constructed with NAND gates instead of NOR gates, what gates should be substituted for the AND gates to ensure an identical timing diagram?

21. Refer to Fig. 11-20. Is it possible to "race" this latch? If so, at which input(s)?

22. Refer to Fig. 11-21(*c*). Why doesn't the Q output change when the data change in the clock window?

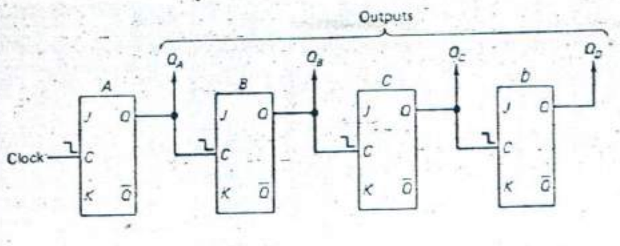
11-4 COUNTERS AND REGISTERS

A *counter* is a circuit based on the sequential logic elements studied in the last section. Look at the binary ripple counter in Fig. 11-23. It is an arrangement of four *J-K* flip-flops and is capable of counting from binary 0000 to binary 1111. The J and K inputs are all floating, so every flip-flop is in the toggle mode. The timing diagram starts at the left with all four outputs at 0. On the first negative clock edge, Q_A goes high. On the second negative clock edge, Q_A goes low, providing a negative edge for the second flip-flop so that Q_B goes high. Follow the timing diagram through to the sixteenth negative clock edge and note that the counter is reset at this time and all four outputs are once again at 0. Thus, the counter has 16 unique states. This is also called the *modulus* of the counter. The modulus is equal to 2^N, where N is the number of sequential logic elements in the counter.

Most electronic diagrams are drawn with the input signals entering at the left. Binary numbers are printed with the LSB at the right. These two conventions are in conflict when describing counters. Figure 11-23 solves this problem by rearranging the truth table columns. Transistor Q_A is the LSB and heads the right-hand column. Transistor Q_D is the MSB and heads the left-hand column.

Figure 11-23 is called a *binary ripple counter* because the count ripples from flip-flop to flip-flop. Check what happens on the eighth negative clock edge. Outputs *A*, *B*, and *C* all go low, and output *D* goes high. Will these four events occur simultaneously? No, the count will ripple from *A* to *B* to *C* and finally to *D*, and *D* will not go high until four propagation delays after the clock edge. The high-frequency performance of ripple counters is limited. Figure 11-24 illustrates a synchronous binary counter with better high-frequency performance. The clock is directly applied to each flip-flop. This means that every flip-flop can toggle at the same time (they are synchronous). Additional connections and some combinational logic are required to allow the synchronous counter to achieve the correct binary se-

Count	Q_D	Q_C	Q_B	Q_A
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1
10	1	0	1	0
11	1	0	1	1
12	1	1	0	0
13	1	1	0	1
14	1	1	1	0
15	1	1	1	1



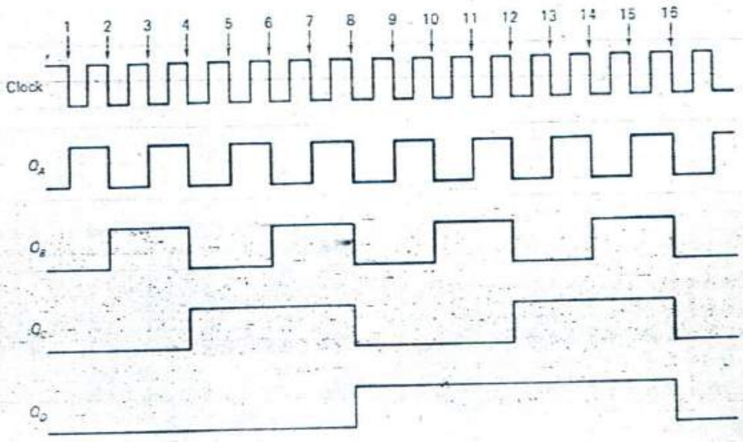


Fig. 11-23 Binary ripple counter.

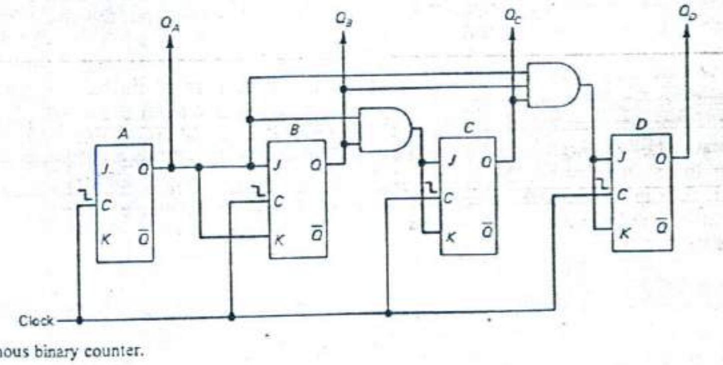


Fig. 11-24 Synchronous binary counter.

quence. The Q_A output also goes to the *J* and *K* inputs of flip-flop *B*. If you check the timing diagram from Fig. 11-23 you will see that Q_B is supposed to toggle only on those edges when Q_A is high before the edge. Also notice that Q_C toggles only on the edges where both *A* and *B* are high before the edge. Finally, note that Q_D toggles only on those edges where *A*, *B*, and *C* are high before the edges. It should now be clear how the connections and AND gates of Fig. 11-24 function.

The natural modulus of a counter is 2^N. It is possible to use feedback to reduce the natural modulus.

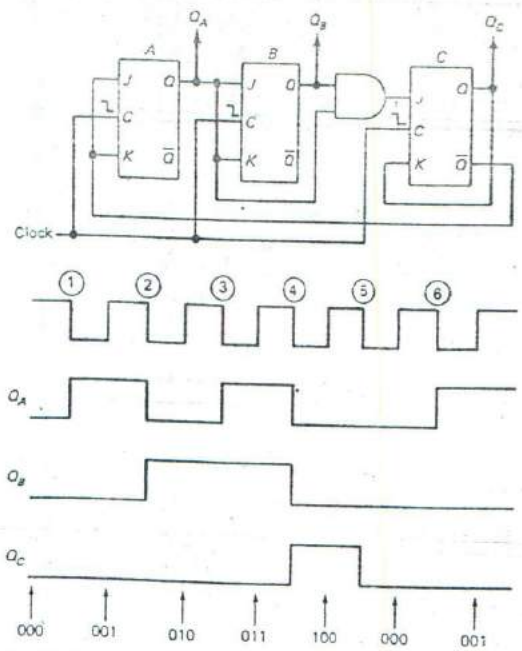


Fig. 11-25 A modulo 5 counter.

For example, Fig. 11-25 shows a modulo 5 counter. It uses three flip-flops, and the natural modulus is $2^3 = 8$. However, feedback from flip-flop *C* to flip-flop *A* and from Q to *K* at flip-flop *C* reduces the natural modulus, and the counter has only five unique states. They range from binary 000 to 100. The counter does not advance from binary 100 to binary 101; instead it resets to 000. As in the last circuit, Q_A feeds the *J* and *K* inputs of flip-flop *B* and Q_A is ANDed with Q_B to control the *J* input of flip-flop *C*. The timing diagram follows a natural pattern up to the fifth negative clock edge. At the fifth edge, Q_A does not toggle. This is because NOT Q_C was low before the edge and the feedback put flip-flop *A* into the inhibit mode. Also note that Q_C does toggle on the fifth edge because Q_C was high before the clock edge, and so was the *K* input of the flip-flop. With $J = 0$ and $K = 1$, the flip-flop is in the reset mode.

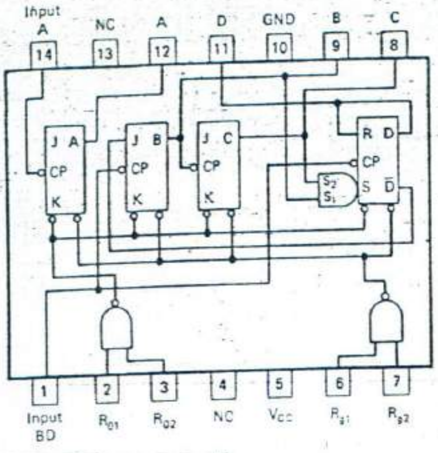
A modulo 5 counter can be combined with a modulo 2 counter to make a modulo 10 counter. Counters with 10 states are called *decade counters;* if they follow the BCD count sequence, they are called *BCD counters*. Figure 11-26 shows the pinout, count sequence, and reset truth table for a 74LS90 decade counter. The output of the modulo 2 counter is pin 12 and is connected to the input of the modulo 5 counter at pin 1 when the device is to be used as a BCD counter. The clock is applied to pin 14. You will recall that binary-coded decimal follows the standard binary sequence up to a count of 1001. The BCD count sequence is indicated in Fig. 11-26(*b*).

For some applications, the count sequence is not important, but the division and the output waveform are. The BCD count sequence shows that the frequency at output D (pin 11) will be equal to one-tenth the clock frequency, but the waveform will be a rectangular pulse of low duty cycle. It is possible to have the IC divide the input frequency by 10 and produce a square wave (50 percent duty cycle) at the output. This is accomplished by feeding the clock into pin 1, connecting pin 11 to pin 14, and taking the output signal from pin 12. The output from a modulo 2 counter is a square wave; therefore, dividing by 5 first and then by 2 produces the desired waveform. However, when this is done, the BCD count sequence is lost.

Figure 11-26 shows the truth table for the decade counter. It has four reset inputs. An X indicates a *don't care* condition, meaning that either a logic 0 or a logic 1 may be applied at that particular reset pin. There are three reset/preset combinations: two of them reset the counter to 0000 and one presets it to 1001. There are four count combinations, and any one of them may be used for counting.

Some digital counters are programmable; Fig. 11-27 is an example. The DM8555 is a decade counter that can be preloaded (or preset) with a BCD number. It has a binary counterpart with the part number DM8556 that can be preloaded with a 4-bit binary number. This allows the modulus of the devices to be adjusted (programmed) by changing the number that is preloaded. For example, if the counter is preset to BCD 0111, the next count will be 1000, then 1001; then it can be preset again to 0111. In this case, the modulus has been programmed to 3. Counters with this capability are known as *modulo-N counters*. To facilitate presetting, the device has four I/O pins for loading data into the counter. Follow the typical timing diagram shown in Fig. 11-27(*c*). First, the counter is cleared to zero by applying a reset pulse to pin 4. Second, the counter is preset to BCD 5 by applying 0101 to I/O pins 14, 13, 11, and 10 and by applying a negative load pulse to pin 7. Note that this sets outputs Q_D through Q_A (pins 2, 3, 5, and 6) to 0101 and occurs at the positive clock edge (this is a synchronous load). Third, the count progresses 6, 7, 8, 9, 0, 1, 2, 3, 4, and 5. Fourth, during the count, the I/O pins are disabled for a period of time by applying a disable pulse to pin 12 (more on this later). Fifth, the counter is disabled for a time by applying a high pulse to pin 1. Sixth, the count resumes to BCD 6.

The DM8555 counter is very flexible and can be used in many ways. It also lends itself to bus-structured digital systems. A *bus* is an arrangement in which devices are connected in parallel on several circuit paths for binary information transfer. Many devices are typically connected to the bus and must be isolated from each other at times. For example, if one logic package tries to pull a bus line to logic high and another tries to pull the same line to logic low, this process creates a problem called *bus con-*



(a)

Count	Output			
	D	C	B	A
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1

Pin 12 connected to pin 1 and Clock applied to pin 14

(b)

Reset inputs				Output			
R_{01}	R_{02}	R_{91}	R_{92}	D	C	B	A
1	1	0	X	0	0	0	0
1	1	X	0	0	0	0	0
X	X	1	1	1	0	0	1
X	0	X	0	COUNT			
0	X	0	X	COUNT			
0	X	X	0	COUNT			
X	0	0	X	COUNT			

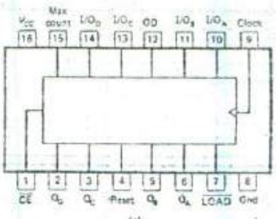
X indicates that either a logical 1 or a logical 0 may be present

(c)

Fig. 11-26 Decade counter. (*a*) Pinout. (*b*) Count sequence. (*c*) Reset/count.

tention. Bus contention can be avoided by using logic devices with tri-state outputs. A *tri-state output* can be at one of three valid conditions: logic high, logic low, or high impedance (tri-stated). The four I/O pins on the DM8555 are tri-state. Any time that pin 12 (output disable) is at logic high, the I/O pins are disabled and go into their high-impedance state. This is shown with Zs in the truth table of Fig. 11-27(*b*). Thus, the I/O pins can be effectively disconnected from the bus, allowing some other logic package to place data on the bus without the problem of bus contention. The four Q output pins of the DM8555 are standard totem pole outputs and cannot be tri-stated.

All of the counters discussed to this point have been *up counters*, which show an increasing count with each clock pulse. *Down counters*, which show a decreasing count with each clock pulse, are also available. Figure 11-28 represents a 74LS192 up/down BCD counter. It is also available in a 4-bit binary version with the part number 74LS193. Both versions are fully programmable and can be used as modulo-N counters. Each output can be set to a high or a low by entering the desired data at the inputs and then pulsing the load input low. This is an asynchronous load since it is independent of the clock inputs. Figure 11-28(*b*) is a typical timing diagram. First, the outputs are reset to 0 by applying a clear

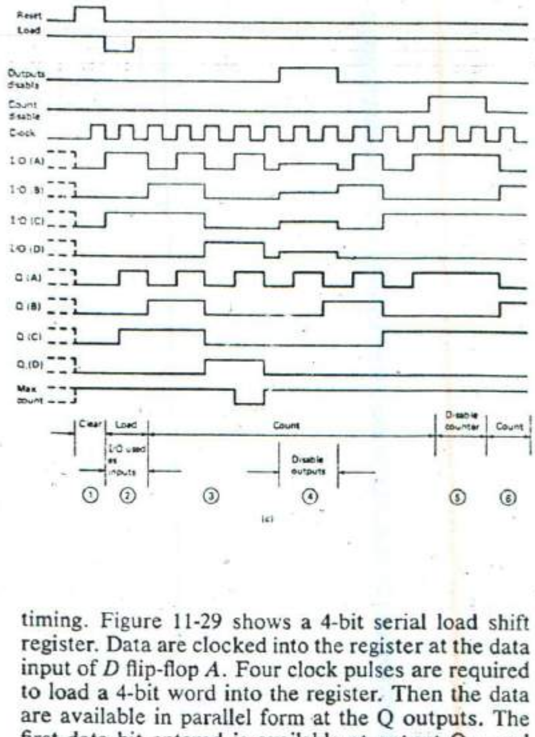


(a)

Control Inputs					I/O Ports				Active Outputs			
$\overline{LOAD}$	$\overline{CE}$	CLK	OD	Reset	I/O$_A$	I/O$_B$	I/O$_C$	I/O$_D$	Q_A	Q_B	Q_C	Q_D
H	X	X	L	H	L	L	L	L	L	L	L	L
H	X	X	H	H	Z	Z	Z	Z	L	L	L	L
H	X	L	L	L	Q_{AO}	Q_{BO}	Q_{CO}	Q_{DO}	Q_{AO}	Q_{BO}	Q_{CO}	Q_{DO}
H	X	L	H	L	Z	Z	Z	Z	Q_{AO}	Q_{BO}	Q_{CO}	Q_{DO}
L	H	↑	L	L	a	b	c	d	A	B	C	D
H	L	↑	L	L		COUNT				COUNT		
H	L	↑	H	L	Z	Z	Z	Z		COUNT		

The I/O pins are used as inputs when they are TRI-STATED, and $\overline{LOAD}$ input is Low. They are outputs and active when $\overline{LOAD}$ input is High and OD is Low.
H = High Level (Steady State)
L = Low Level (Steady State)
X = Don't Care including transitions
a, b, c, d = The level of the steady state input at inputs A, B, C, D respectively
Q_{AO}, Q_{BO}, Q_{CO}, Q_{DO} = The level of Q_A, Q_B, Q_C, Q_D respectively, before the indicated steady state input conditions were established.

(b)

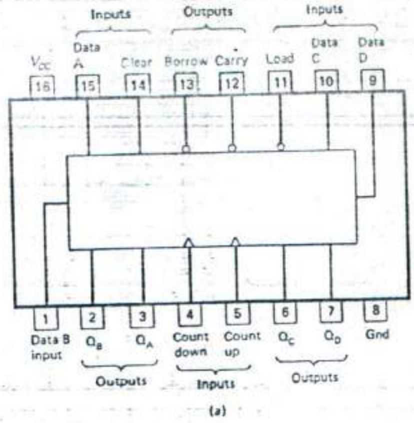


(c)

Fig. 11-27 DM8555 programmable decade counter. (*a*) Pinout. (*b*) Truth table. (*c*) Typical timing diagram.

pulse to pin 14. Second, the counter is preset to BCD 7 by applying 0111 to the data pins and pulsing pin 11 low. Third, the count progresses upward to 8, 9, carry, 0, 1, and 2 as count-up pin 5 is clocked. Fourth, the count progresses down to 1, 0, borrow, 9, 8, and 7 as count-down pin 4 is clocked. The carry and borrow pins are used to cascade more than one device. *Cascading* is accomplished by feeding the carry and borrow outputs to the count-up and count-down inputs, respectively, of the succeeding counter.

Shift registers constitute another category of digital circuits based on sequential logic elements. They are useful for temporary storage, change of data from one format to another, and sequence control and timing. Figure 11-29 shows a 4-bit serial load shift register. Data are clocked into the register at the data input of *D* flip-flop *A*. Four clock pulses are required to load a 4-bit word into the register. Then the data are available in parallel form at the Q outputs. The first data bit entered is available at output Q_D, and the last, or fourth data bit, is available at output Q_A. The shift register is useful for converting *serial data* (one bit at a time) to *parallel data* (all bits at a time). It is also useful for temporary storage of data. If the data are needed later, they can be retrieved in serial form at output Q_D. The first bit that was loaded is immediately available at the serial data output. Then, three clock pulses will be required to shift the other



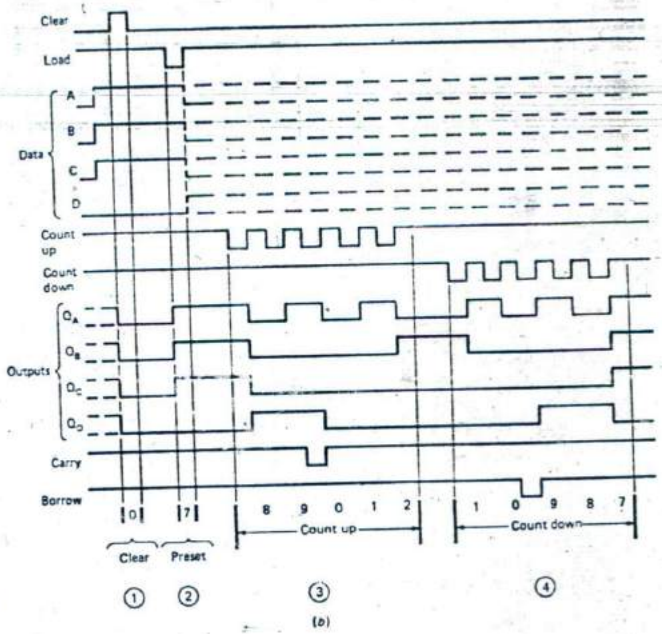


Fig. 11-28 74LS192 synchronous up/down counter. (a) Pinout. (b) Typical timing diagram.

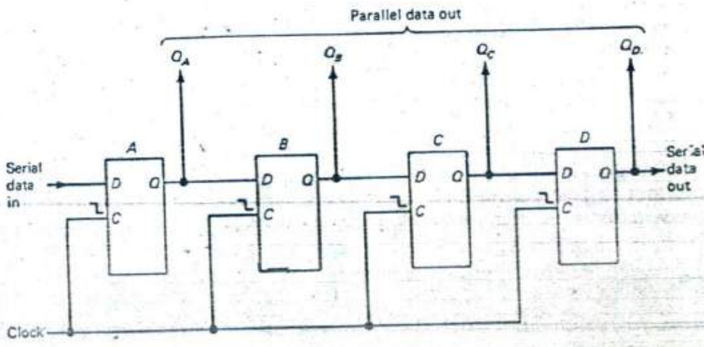
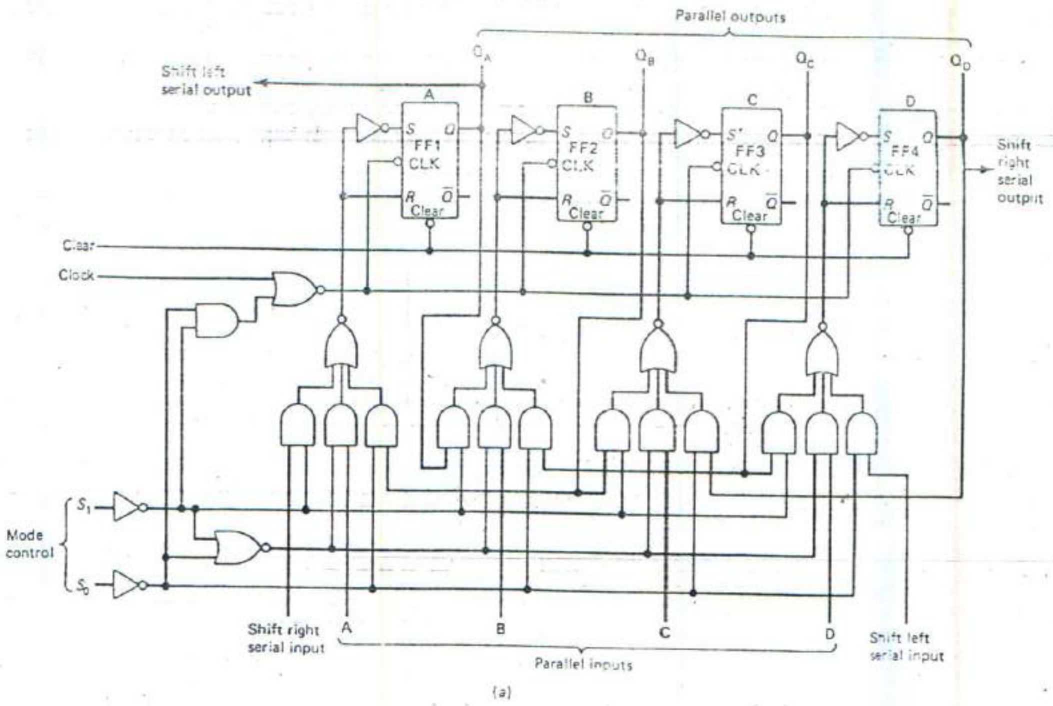


Fig. 11-29 Shift register.



Operation	Mode control	
	S_1	S_0
Parallel load	1	1
Shift right $Q_A \rightarrow Q_D$	0	1
Shift left $Q_D \rightarrow Q_A$	1	0
Inhibit clock (hold)	0	0

(b)

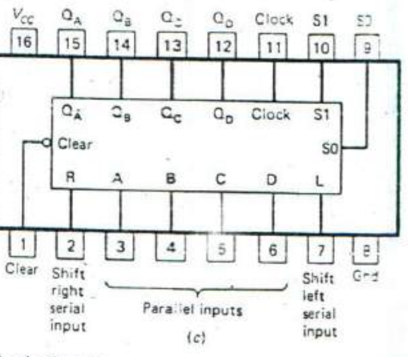
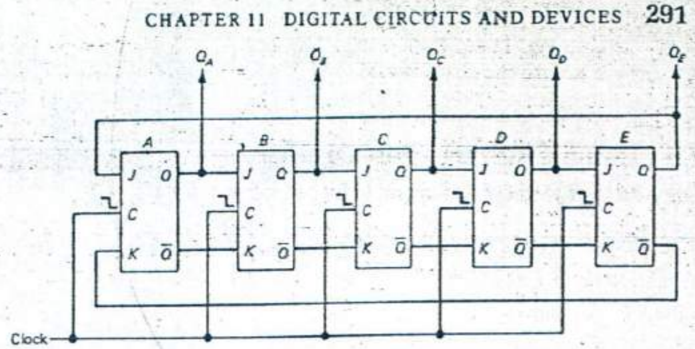


Fig. 11-30 74LS194 4-bit bidirectional universal shift register. (*a*) Logic diagram. (*b*) Mode control. (*c*) Pinout.

3 bits to the output. A register of this type is sometimes called a *first-in first-out* (FIFO) register.

A much more versatile device is shown in Fig. 11-30. It is a 74LS194 universal 4-bit shift register. It is capable of shift right, shift left, and parallel loading and has parallel outputs. Figure 11-30(*b*) shows the truth table for the two mode control pins S_1 and S_2. When both mode control pins are high, parallel data are loaded into the register. This occurs because a NOR gate senses the inverted mode signals. With both mode inputs high, both NOR gate inputs are low. The output of the NOR gate is therefore high, and it supplies a logic high to one lead of each of the center AND gates at the bottom of the logic diagram. The output of each AND gate will follow the data applied at the parallel inputs. The data reach each flip-flop through a NOR gate and an inverter and are loaded on the first positive clock edge. With the mode control in the shift right pattern ($S_1 = 0$ and $S_0 = 1$), a high is supplied to one input of every left-hand AND gate in the bottom group. The shift right serial input is now active, and data can be clocked into flip-flop *A*. The data from *A* will shift to *B* since its AND gate is also active, and so on. Reversing the



Q_A	Q_B	Q_C	Q_D	Q_E	Clock pulse
1	0	0	0	0	0
0	1	0	0	0	1
0	0	1	0	0	2
0	0	0	1	0	3
0	0	0	0	1	4
1	0	0	0	0	5

N unique states

Fig. 11-31 Circulating shift register (ring counter).

mode control pattern (S_1 = 1 and S_0 = 0) places a logic high at one input of each right-hand AND gate in the bottom group. This activates the shift left serial input, and data are clocked into flip-flop *D*. The output of *D* is clocked into *C* since its AND gate provides a data path to the left, and so on. With both mode control pins low, the shift register is inhibited because the clock cannot get through the NOR gate at the left of the diagram. Note that the other NOR gate input is from an AND gate which has both inputs high when the mode control pins are both low. The AND gate output will be high, holding the clock NOR gate output low.

The output from a shift register can be connected back to the input. The data will circulate endlessly in this case as the register is clocked. Refer to Fig. 11-31. This circuit is known as a *ring counter*. Assume the initial condition is that flip-flop *A* is set and all others are reset. On the first negative clock edge, the high bit will shift to the right by one position. It will shift to the right again on the second negative edge. When the fifth clock pulse comes along, the high bit shifts from flip-flop *E* to *A*. Note that ring counters have *N* unique states, where *N* is the number of flip-flops. Ring counters are sometimes used as *control sequencers*: one operation follows another, followed by another, and so on, until the process restarts, and the first operation is performed again. The circuit of Fig. 11-31 is not complete since some means of establishing the initial condition must be added. For example, some means will be required to preset one flip-flop and clear all the others. Also, the circuit may get into an illegal mode (more than one high bit) because of noise or a momentary power dip. For this reason, the circuit may also require extra logic to detect illegal modes and correct them.

REVIEW QUESTIONS

23. Refer to Fig. 11-23. If the clock frequency is 10 MHz, what is the output frequency at Q_D?

24. Do all of the flip-flops in Fig. 11-23 change state at the same time on the 16th negative clock edge? Why?

25. Refer to Fig. 11-26. Can this IC be used as a BCD divider and output a square wave of one-tenth the input frequency?

26. Refer to Fig. 11-27. What happens to the I/O pins when a logic 1 is applied to pin 12?

27. Refer to Fig. 11-27. Is the load function synchronous or asynchronous?

28. Refer to Fig. 11-27. Can Q_A through Q_D be tri-stated? Which device pins would be appropriate for connecting to a bus that has other device outputs connected to it?

29. Refer to Fig. 11-28. Is the load function synchronous or asynchronous?

30. Refer to Fig. 11-28. What number should the counter be preset with to use it as a modulo 8 device in the up-count mode?

31. Refer to Fig. 11-29. What will the parallel data output be if the data input is held low for four clock edges?

33. Refer to Fig. 11-30. Both mode control pins are high. When are the data loaded?

11-5 MULTIPLEXERS AND DECODERS

A *digital multiplexer* is a "several into one" circuit. Multiplexers are used to select data from one of several sources and output those data onto a single line. They are also called *data selectors*. The typical multiplexer IC uses select pins to determine which of the inputs will be connected to the output. For example, a four-line to one-line multiplexer will have two select pins. The binary pattern at these pins will range from 00 (input 0 is selected) to 11 (input 3 is selected).

It is possible to use a multiplexer as a parallel-to-serial data converter. For example, if a 4-bit word is applied to the data inputs of a four-line to one-line device and if the select pins are clocked by a two-stage counter from binary 00 to binary 11, the word will appear in serial form at the output. The first input will appear first when the counter is at 00, then the next input will appear at a count of 01, and so on.

A *demultiplexer* represents an opposite function; it is a "one into several" logic circuit. Demultiplexers take data from a single source and distribute them to one of several output lines. They are also known as *data distributors*. Figure 11-32 shows the pinout and the logic diagram for a 74LS138 decoder/demultiplexer. It is used for memory decoding and data-routing applications. (Another type of decoder is shown in Fig. 11-33.) The 74LS138 distributes to one of eight output lines, depending upon the logic conditions at its three select inputs. When all three select

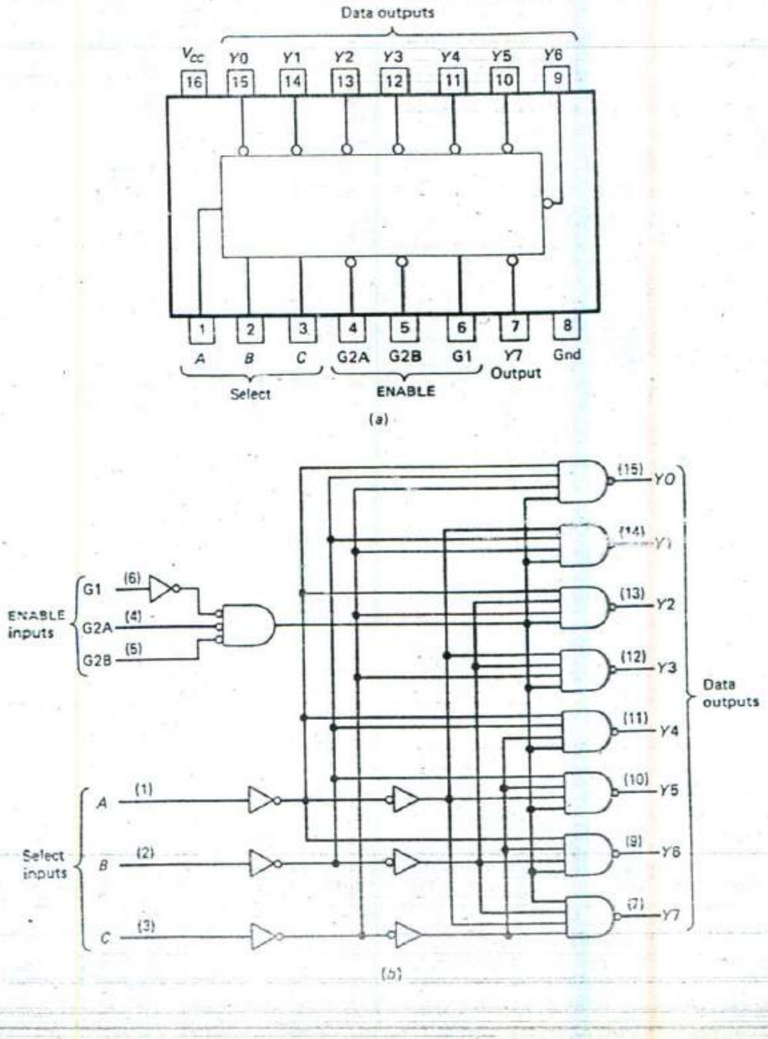


Fig. 11-32 74LS138 decoder/demultiplexer. (*a*) Pinout. (*b*) Logic diagram.

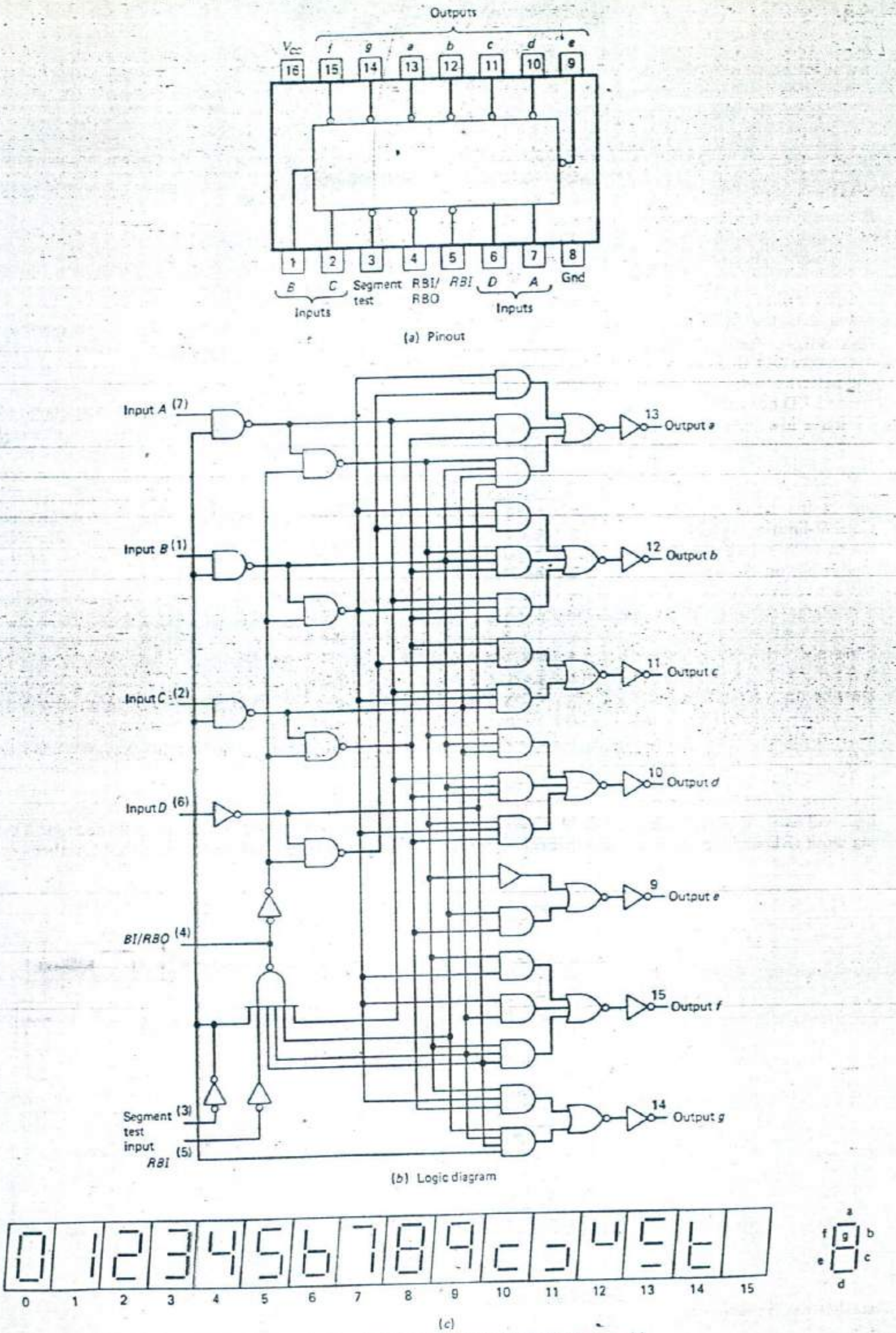


Fig. 11-33 74LS47 open collector BCD/7-segment decoder/driver. (*a*) Pinout. (*b*) Logic diagram. (*c*) Display and segment identification.

inputs are at logic 0, output Y0 is selected. If select *A* is high and *B* and *C* are low, output Y1 is selected. The LSB of the select word is *A*, and *C* is the MSB. Any one of the three enable inputs can be used for data input for demultiplexing operations. Two of the enable inputs are active low (G2A and G2B), and the other is active high (G1). The G1 input should be used if the data do not require inversion, and the other enable inputs should be grounded.

A *decoder* is a circuit or device that reacts to one or more conditions at its inputs. Decoders are commonly used to convert binary, BCD, ASCII, or some other common code into an uncoded form. A good example is the 74LS47 BCD-to-seven-segment decoder/driver shown in Fig. 11-33. Its application is to convert four BCD lines into the appropriate levels and patterns to display decimal numbers on common anode LED numeric displays. It is also a driver since its active low open collector outputs can sink up to 24 mA of current. A common anode display has all of its anodes tied to a single pin. In practice, this pin will connect to the positive supply. Grounding any one of the LED cathodes will light one segment. Current-limiting resistors are usually required between each segment cathode and its corresponding output pin on the decoder/driver. Since the 74LS47 has active low outputs, it will sink the cathode current. Refer to Fig. 11-33(*c*), which shows the possible displays and the segment identification. To display numeral 8, all segments must be on and all outputs will be low. To display numeral 1, segments *b* and *c* must be on, and output *b* and output *c* will be low. The binary numbers from 1010 (decimal 10) through 1111 (decimal 15) are *forbidden* in BCD. Note the display conditions for these numbers.

The 74LS47 shown in Fig. 11-33 also has a segment test input at pin 3. Grounding this pin when pin 4 is high will light all of the segments. Pin 4 is the blanking input (BI) and ripple blanking output (RBO) pin. This pin may be pulsed to control LED intensity. Pin 5 is for *ripple blanking input* (RBI). The ripple blanking pins are used in those applications in which it is desired to blank leading or trailing zeros in a numeric display. If the RBI is low, the character 0 will not be displayed; all segments will be turned off when the BCD input code is 0000. Also, the RBO will go low to extinguish character 0 in the next stage in those applications in which leading or trailing zeros are to be blanked. Figure 11-34 shows the arrangement for leading zero blanking. The RBI pin on the most significant decoder/driver is at logic zero. The BCD code at its inputs is 0000, and the output display is blanked. The RBO is low, placing the next display in the zero blanking mode. Its BCD input is also 0000 and is blanked, and its RBO pin is at logic 0. The next display is also in the zero blanking mode, but its BCD input is 0101, so some of its outputs go to logic zero, and the LED display shows the numeral 5. Note that its RBO pin is at logic 1 since its BCD input was not 0000. Therefore, the next decoder/driver displays a 0 even though its BCD input is 0000. *Trailing zero suppression* is accomplished by grounding the RBI pin on the least significant decoder/driver, and connecting its RBO to the RBI of the decoder/driver to the left, and so on.

The problems with multidigit displays, such as the one shown in Fig. 11-34, are the large number of components required and the large number of connections. For example, to have a six-digit display, 42 current-limiting resistors will be required, and 43 connections (add 1 connection for the positive supply) have to be run to the display unit. Multidigit displays are often multiplexed to save components and connections. Refer to Fig. 11-35. The appropriate segment select transistors are turned on to display the desired character. At the same time, one of the digit select transistors is also turned on. That particular display will be active. Later, another seg-

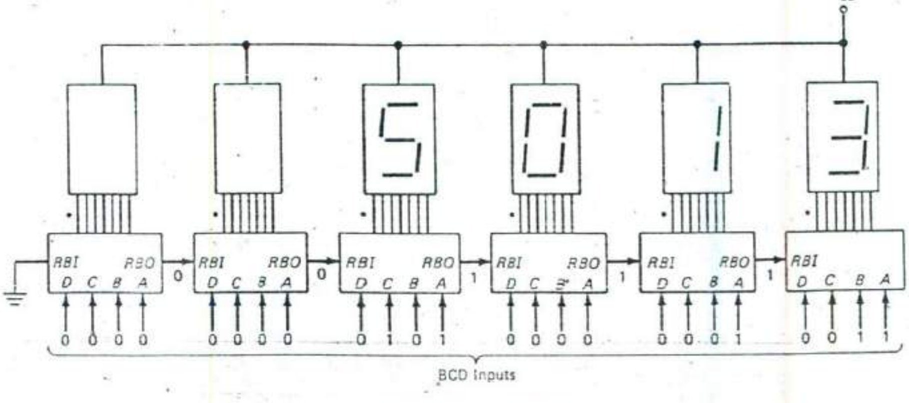


Fig. 11-34 Leading zero blanking.

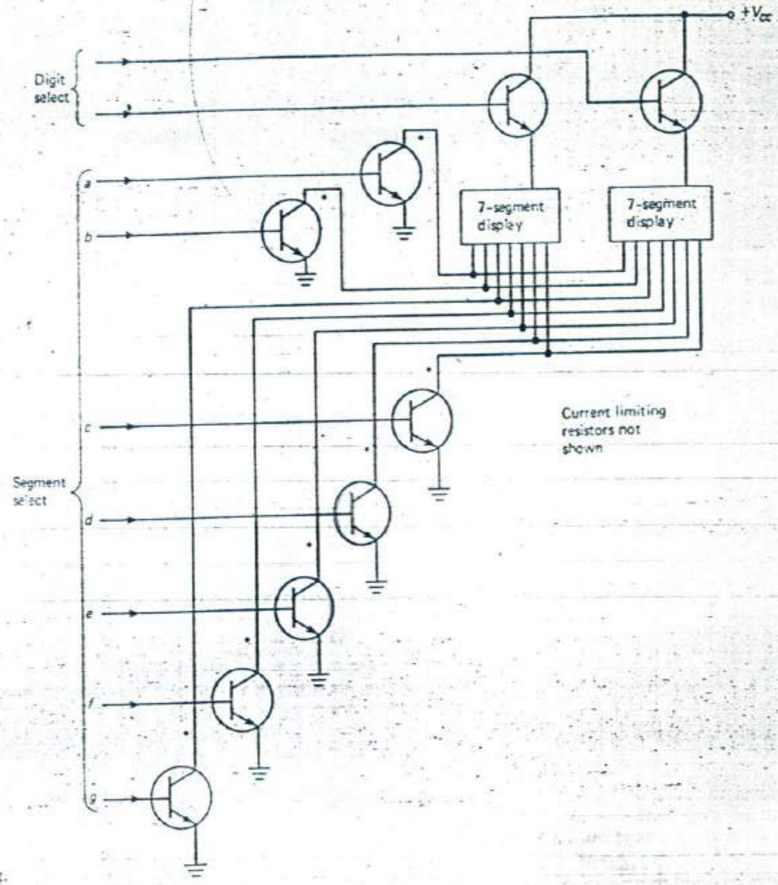


Fig. 11-35 Display multiplexing.

ment code is applied, and the next digit select transistor is turned on while the other is turned off. Now the second display is active. If this is done rapidly, the blinking is not perceptible. The savings in components and connections are substantial when a large number of digits must be displayed. For example, a multiplexed six-digit display requires only seven resistors and 13 connections to the display unit, which includes seven segment select lines and six digit select lines. Since the segments are in parallel, only seven lines are required regardless of the number of digits displayed. The clock frequency must be increased as the number of displays increases.

An *encoder* is a logic circuit that receives one or more signals in an uncoded form and outputs a code that can be used by another logic circuit. A *keyboard encoder* converts a keypress into some code; the keyboard may be octal, decimal, hexadecimal, alphanumeric, or some special type. The encoder can be a single IC or can be built up from several devices. For example, *single-chip encoders* convert a keypress into the equivalent ASCII code. Figure 11-36 depicts a simple decimal-to-BCD encoder circuit. When all of the decimal inputs are low, the output is BCD 0000. If the 1 input goes high, the BCD output is 0001. If the 9 input is high, the BCD output is 1001. Verify the BCD output code for several different input conditions.

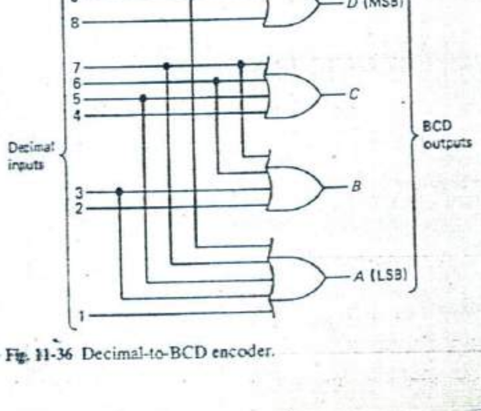
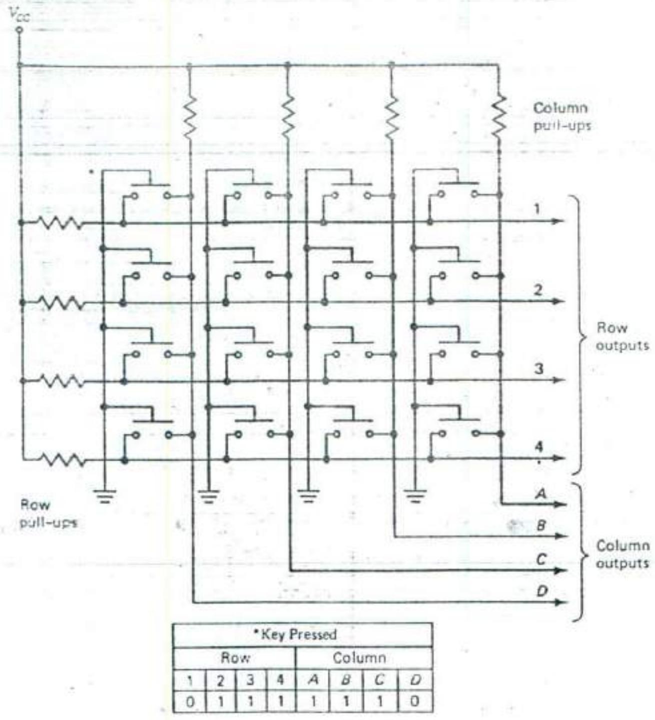


Fig. 11-36 Decimal-to-BCD encoder.



*Key Pressed							
Row				Column			
1	2	3	4	A	B	C	D
0	1	1	1	1	1	1	0

Fig. 11-37 A keypad matrix.

Single-chip keyboard encoders are available with row and column inputs. For example, there may be a four by four matrix of 16 key switches. Figure 11-37 shows a typical keyboard matrix based on DPST N.O. key switches. If any key switch is closed, one of the row outputs and one of the column outputs will go low. The encoder chip may also contain debouncing circuitry and a strobe generator. The *strobe* is an output used to alert the rest of the system that a key has been pressed. Another desirable feature is suppressed output for illegal entries (more than one key pressed at a time).

REVIEW QUESTIONS

33. Refer to Fig. 11-32. Select A = 0, B = 1, and C = 1. Which data output has been chosen?

34. Refer to Fig. 11-32. Suppose it is desired to use this device as a one-to-four data distributor. How can this be accomplished?

35. How can the device in Fig. 11-32 be used as an inverting demultiplexer?

36. Refer to Fig. 11-34. What should the display show if the RBI pin at the most significant digit loses its ground?

37. Refer to Fig. 11-34. What should the display show if the BCD code to the least significant digit changes to 0000?

11-6 ARITHMETIC ELEMENTS

Binary arithmetic is really quite simple. Since there are only two symbols to manipulate, there are only a few possible combinations. For example, in binary addition, there are only four cases:

$$\begin{aligned} 0 + 0 &= 0 \\ 1 + 0 &= 1 \\ 0 + 1 &= 1 \\ 1 + 1 &= 0 \text{ (carry 1)} \end{aligned}$$

The rules of *binary addition* match the truth table for the exclusive OR gate. Therefore, all that is needed to add 2 bits is an exclusive OR gate plus a carry generate circuit. Refer to Fig. 11-38. This circuit is called a *half-adder;* it produces the sum and carry outputs for 2 input bits. It is considered a half-adder because when binary words with more than 1 bit are added, carries from column to column occur which require a circuit with three inputs: one for the

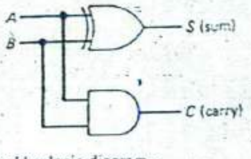


Fig. 11-38 Half-adder logic diagram.

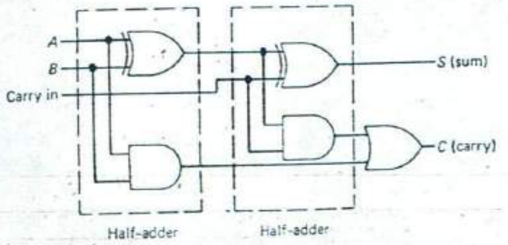


Fig. 11-39 Full-adder logic diagram.

augend, one for the addend, and one for the carry. For example, suppose we add binary 1001 to binary 0101:

```
CARRY  ──────→    1
AUGEND ──────→ 1001
ADDEND ──────→ 0101
               ────
SUM    ──────→ 1110
```

This example shows that a circuit that can handle three inputs is needed. Figure 11-39 shows the logic required for a *full-adder*, which is based on two half-adders plus an OR gate.

A 4-bit parallel adder with ripple carry is illustrated by Fig. 11-40. It is constructed from one half-adder and three full-adders. A half-adder is sufficient for adding the LSB of the augend to the LSB of the addend. However, all subsequent circuits must be full-adders because carries can be generated anywhere along the line. The carries ripple from stage to stage in this type of circuit. Under worst-case conditions, the correct sum will not be available until four propagation delays have occurred. For example:

```
CARRIES ──────→   111
AUGEND  ──────→  1111
ADDEND  ──────→  1111
                ─────
SUM     ──────→ 11110
```

Of course, as the size of the adder increases to 8, 16, or 32 bits, the ripple carries will create significant delays. Faster circuits using look-ahead carry logic are available. They eliminate the ripple carries and the cumulative delays.

Binary subtraction is also very simple. As in the case of addition, there are only a few combinations:

$$0 - 0 = 0$$
$$1 - 0 = 1$$
$$0 - 1 = 1 \text{ (borrow 1)}$$
$$1 - 1 = 0$$

Figure 11-41 shows the logic required for a *full-subtractor*. Note the similarity to the full-adder circuit. Also note that there are three inputs: one for the subtrahend, one for the minuend, and one for the borrow. Just as the full-adder had to add in carries from prior columns, the full-subtractor must subtract borrows from prior columns. As an example:

```
BORROWS    ──────→  111
MINUEND    ──────→ 1010
SUBTRAHEND ──────→ 0111
                   ────
DIFFERENCE ──────→ 0011
```

Many digital systems do not have subtraction circuits. They accomplish subtraction with adding circuits by converting the subtrahend into its 2s complement form and then adding it to the minuend. A number is converted into 2s complement form by inverting every bit (this operation puts it in 1s complement form) and then adding 1. As an example, let's convert the subtrahend from the previous example into its 2s complement form:

```
BINARY FORM    ──────→ 0111
1s COMPLEMENT  ──────→ 1000
ADD 1          ──────→    1
                       ────
2s COMPLEMENT  ──────→ 1001
```

Now, let's see what happens when we add the 2s complement form of the subtrahend to the minuend:

```
MINUEND        ──────→ 1010
2s COMPLEMENT  ──────→ 1001
                       ────
SUM            ──────→ 0011 (the last carry is
                             thrown away)
```

Ignoring the final carry, the sum obtained is the same as the difference when we subtracted. It should be clear that subtraction may be accomplished by adding the 2s complement of the subtrahend to the minuend. Don't forget to discard the final carry.

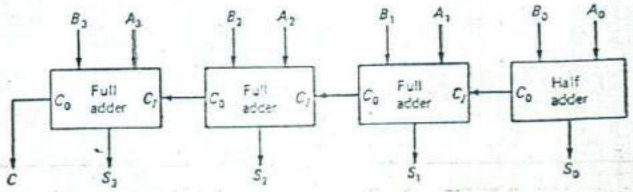


Fig. 11-40 Four-bit parallel adder with ripple carry.

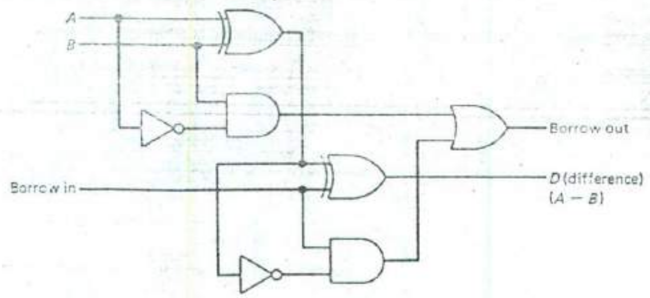


Fig. 11-41 Full-subtractor logic diagram.

The 74S381 IC is an arithmetic logic unit that can perform any of eight binary arithmetic and logic functions on two 4-bit words. The operations are selected by function select lines. A look-ahead carry circuit is provided, and fast cascade operation is accomplished by *propagate* and *generate* outputs. The device has two subtract modes, one add, three boolean, preset, and clear. Boolean operations can be performed on 4-bit words with this device. For example, if 1010 is ORed with 0101, the function output will be 1111, since there is 1 high bit in each position. However, if the AND function is selected with the same input words, the function output will be 0000 because there is no occurrence of 2 high bits in any position.

The rules of *binary multiplication* are also few in number:

$$\begin{aligned}0 \times 0 &= 0\\ 1 \times 0 &= 0\\ 0 \times 1 &= 0\\ 1 \times 1 &= 1\end{aligned}$$

A separate multiplier circuit can be built; however, multiplication is more likely to be performed in an adder circuit. It is possible to use a shift register and an adder to multiply. The process is based on the rules of binary multiplication and the formation of partial products which are summed to find the final product. The following example illustrates the procedure.

EXAMPLE

Multiply 1010 by 0101.

SOLUTION

```
MULTIPLICAND→  1010
MULTIPLIER   → 0101
               ----
               1010  (first partial product)
              0000   (second partial product)
             1010    (third partial product)
            0000     (final partial product)
            ------
PRODUCT ——→ 110010
```

The example shows the way that the process of shift and add can be used to multiply. Division is a similar process. It can be done by repeated subtraction. Since subtraction can be done in an adder, once again addition is seen to be the key process. In fact, all arithmetic and mathematical operations can be accomplished by an adder and some ancillary support circuitry.

Rate multipliers represent another approach to binary arithmetic. They are circuits that take one input pulse frequency and provide two output frequencies. One of the output frequencies is programmable. Rate multipliers can be used to control hydraulic and pneumatic valve actuation rates in industrial applications, and they may also be used to control stepper motor velocities. Figure 11-42 shows the pinout for a 7497 binary rate multiplier. It will provide from 0 to 63 output pulses on pin 5 for every 64 clock pulses applied to pin 9. With strobe, clear, and enable grounded, the enable output will provide 1 pulse for every 64 input clock pulses. The enable output is also called the *base rate output* and is fixed. The other output is variable and is at 0, 1, 2, 3, 4, 5, etc., times the base rate, giving it the name *rate multiplier.*

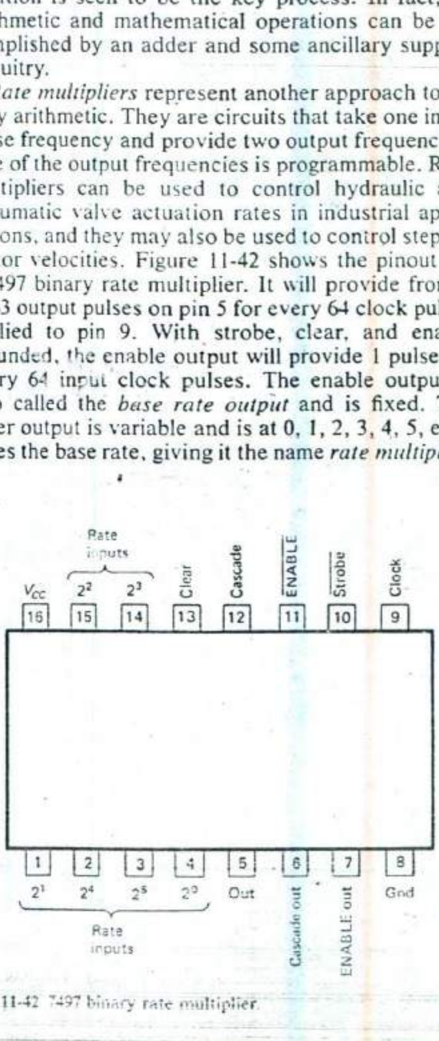


Fig. 11-42 7497 binary rate multiplier.

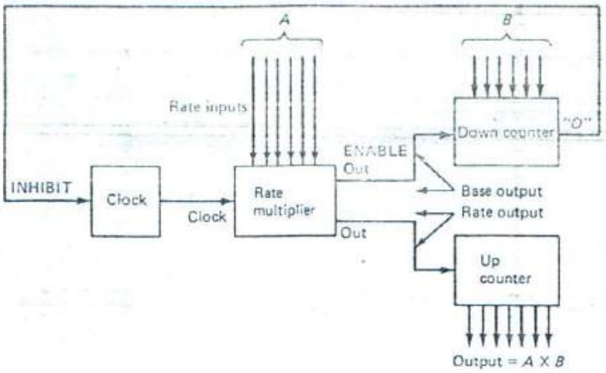


Fig. 11-43 Multiplication with a rate multiplier.

The variable output rate is determined by the binary code applied to the rate inputs.

Figure 11-43 shows a multiplication circuit based on a rate multiplier. The base rate is applied to a down counter, and the rate output is applied to an up counter. The multiplicand is applied to the rate inputs. The action begins with the multiplier loaded into the preset inputs of the down counter. The up counter is cleared, and then the clock is started. The circuit runs until the down counter reaches zero, and this inhibits the clock. The up counter outputs are now equal to the product because the rate output is N times the base rate, where N is the binary word applied to the rate inputs. For example, if N = 000101, the rate output will be five times the base output. The up counter will get five pulses for every one pulse supplied to the down counter. If the down counter was preset to 000010 (2), the up counter will end up with a total count of 0001010 ($5 \times 2 = 10$). The circuit can be converted to a divider by interchanging the outputs of the rate multiplier. The base output will be applied to the up counter, and the rate output will be sent to the down counter. The up counter will contain B/A when the down counter reaches zero. Rate multipliers can also produce squares, square roots, and other arithmetic operations.

The rate output of a rate multiplier is not always a periodic waveform. The output pulses are of constant width but often are not uniformly spaced. For example, if the multiplier is programmed to produce 9 out of 64 clock pulses, they cannot be evenly spaced because 64 cannot be divided evenly by 9. This effect is known as *pulse jitter*. Binary-coded decimal rate multipliers are also available and provide N out of 10 clock pulses.

Figure 11-44 shows the pinout for a 74LS85 4-bit magnitude comparator. It compares two 4-bit binary words presented to its A and B data inputs. It has three outputs: A is greater than B ($A > B$), $A = B$, and A is less than B ($A < B$). One of these outputs will be high for any given set of input conditions. The IC also has cascading inputs that make it easy to use several devices so that words larger than 4 bits can be compared.

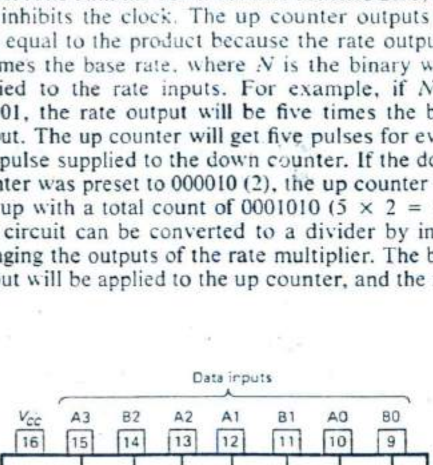


Fig. 11-44 74LS85 4-bit magnitude comparator.

REVIEW QUESTIONS

38. Refer to Fig. 11-40. Adding binary 1010 to 0101 will produce a correct sum output in less time than when adding binary 0111 to 1011. (true or false)

39. Refer to Fig. 11-42. The word applied to the rate inputs is 010111_2, and the clock input is 640 Hz. The strobe, clear, and enable pins are low. What is the base rate output at pin 7? What is the rate output at pin 5?

40. Will the output waveform in question 39 be periodic? Will it show pulse jitter?

41. What is the binary sum of 10011_2 and 10111_2?

42. What is the minimum number of full-adders required to add two 5-bit words? How many half-adders would also be required?

43. What is the binary result of subtracting 10101_2 from 11000_2?

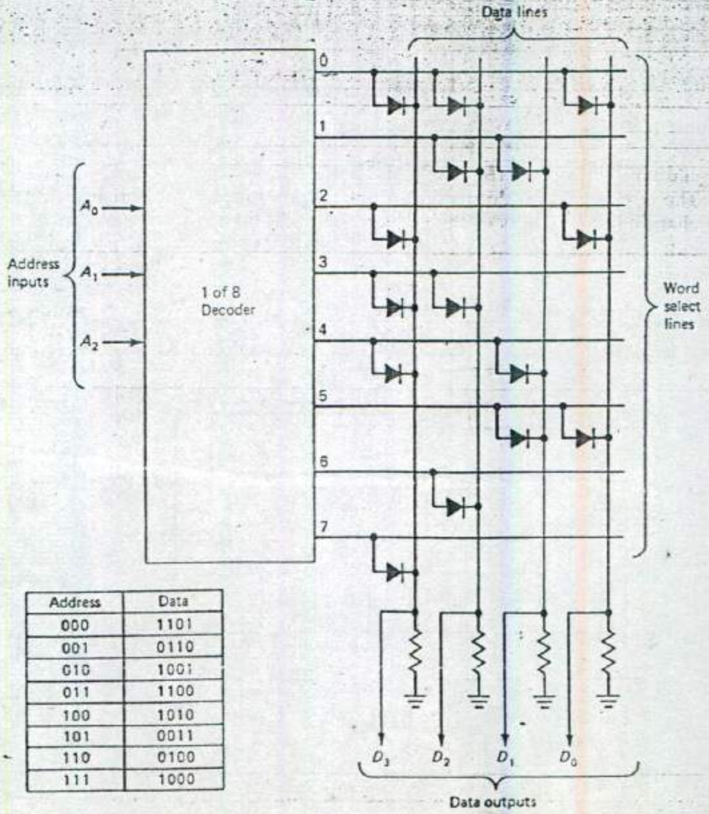
44. Change 10101_2 to 2s complement form and add it to 11000_2. How does this result compare with the results of question 43?

11-7 MEMORY

In the broadest sense, *memory* includes all circuits and devices that are capable of storing binary information. Therefore, the latches, flip-flops, and registers already covered in this chapter can be considered as memory devices. The memory devices discussed in this section are specifically designed for short- and long-term storage of significant amounts of binary information. The first memory device to be discussed is the read-only memory (ROM).

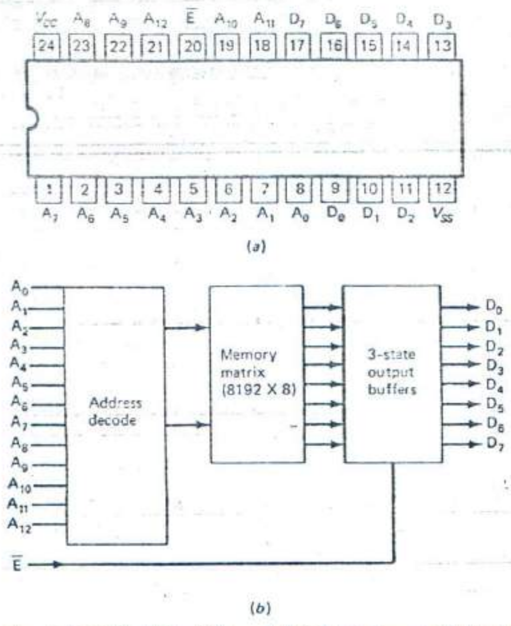
Figure 11-45 shows a simple diode ROM. It is arranged as a matrix of eight 4-bit words. A decoder circuit translates the bit pattern on the address input lines to one of eight word select lines. When the address inputs are all low, word select line 0 goes high. Note that there are three diode anodes connected to the 0 select line. Current flows from the decoder, through the three diodes, and into three of the load resistors. The data outputs are high at these three locations. The word stored at location 0 is 1101. If the address input lines are changed to 001, word 1 is selected. The data output is now 0110. Examine the diode matrix, the table of addresses, and data in Fig. 11-45 and verify each 4-bit data word.

A more practical ROM is shown by Fig. 11-46. It is housed in a 24-pin package and contains 8192 words of 8 bits each. Eight-bit words are very common in digital circuits and are called *bytes*. Therefore, the IC may be called an *8 K-byte ROM*. Since in this case K represents 1024, 8192 is rounded to 8K. Or, it could be called a *64 K-bit ROM* since it has 65,536 bits stored inside ($8192 \times 8 = 65{,}536$). The part number ends in 64, which is the rounded (to the nearest power of 2) number of K bits the device has stored. This is a common, but not universal, practice for memory ICs. The device has 13 address pins labeled A_0 through A_{12}. This number of address pins is required because $2^{13} = 8192$. It also has 8 data pins labeled D_0 through D_7, a +5-V supply pin (V_{CC}), a ground pin (V_{SS}), and an enable pin (NOT E). The block diagram, Fig. 11-46(*b*), shows that the NOT enable pin controls the tri-state output buffers. When the NOT enable input is high, the



Address	Data
000	1101
001	0110
010	1001
011	1100
100	1010
101	0011
110	0100
111	1000

Fig. 11-45 An 8 × 4-bit ROM.



V_{CC} A_8 A_9 A_{12} $\overline{E}$ A_{10} A_{11} D_7 D_6 D_5 D_4 D_3
24 23 22 21 20 19 18 17 16 15 14 13
1 2 3 4 5 6 7 8 9 10 11 12
A_7 A_6 A_5 A_4 A_3 A_2 A_1 A_0 D_0 D_1 D_2 V_{SS}
(a)
A_0–A_{12}
Address decode
Memory matrix (8192 X 8)
3-state output buffers
D_0–D_7
$\overline{E}$
(b)

Fig. 11-46 68364 8192×8-bit mask ROM. (*a*) Pinout. (*b*) Block diagram.

output buffers are in their high-impedance state. This state makes the ROM compatible with bus-structured digital systems. It is called a *mask ROM* because it is programmed with a photolithographic mask when it is manufactured.

Mask ROMs are feasible for high-production memory applications. Since they involve a custom mask design for information storage, they must be produced in reasonably large quantities to distribute the mask design costs over a number of units. They are typically used as code converters, character generators, trigonometric look-up tables, logarithmic look-up tables, and as storage for monitor programs. A *memory-type code converter* applies the code to be converted to the address inputs, and the data outputs show the converted code. A *character generator* takes a code such as ASCII at its inputs and outputs the bit pattern necessary to form the character on a display device such as a cathode-ray tube. Look-up tables eliminate the need for some arithmetic operations in digital circuits. It is possible to generate trigonometric and logarithmic functions with adders, but the process takes time. Memory look-up tables are much faster. A *monitor program* is a control program for a computer (this topic will be covered in Chapter 12).

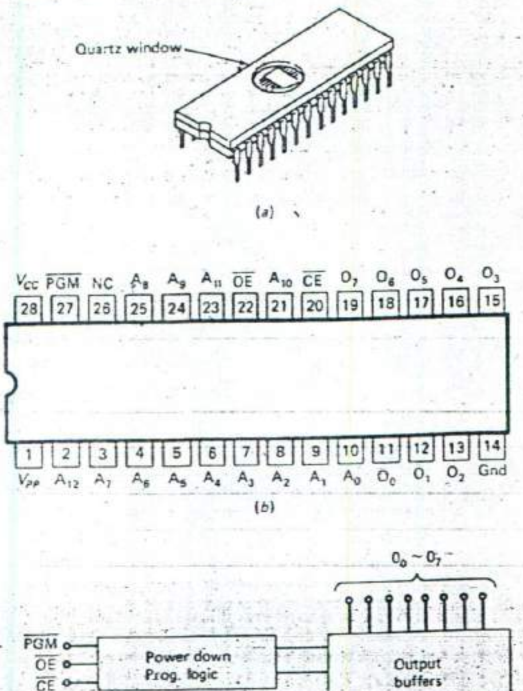
Mask ROMs are available in several arrangements. Some examples are 4096 × 8, 8192 × 8, 16,384 × 8, 32,768 × 4, and 65,536 × 4. Bytewide memory systems are used extensively. This brings up an important question: how can memory devices with word sizes of 4 bits be used in a byte-sized system? They are used by connecting all address pins and the enable pins in parallel. One IC will then provide the 4 lower bits, and the other will provide the 4 higher bits. Likewise, four parallel 4-bit ROMs can be used in a system with a word size of 16 bits. Mask ROMs have a typical access time of 350 ns. This means that valid data will be available at the outputs 350×10^{-9} s after stable address and enable signals are applied. Slower, but more power-efficient mask ROMs are also available. They have access times on the order of 6 μs but use only about 1 percent of the amount of power. They are based on the complementary metallic oxide semiconductor (CMOS) process, which will be covered in the next section of this chapter. The CMOS devices are particularly well suited to battery-operated equipment.

Mask ROMs are not cost-effective for small and moderate quantity designs. For this reason, *field-programmable ROMs* have been developed. These devices make producing any number of ROMs feasible. They are programmed by fusing diode jumpers inside the memory matrix. Refer again to Fig. 11-45. A field-programmable ROM can be based on a diode matrix. The device will be shipped with a full array of diodes. A momentary high programming current can be applied to those diodes that must be removed from the matrix. For example, word 5 shows two diodes. The diodes in the two most significant positions can be eliminated by passing a large current through them to fuse their connecting links and remove the diodes from the matrix. A second technique of field programming is the *blown diode system*. It also uses a high programming current but in pulse form. Selected junctions are short-circuited by aluminum migrating across the junction. The migration is caused by the high current pulses. Whichever programming process is used is performed in a special piece of equipment called a *programmable read only memory* (PROM) *burner* or *PROM programmer*.

The PROMs are one-shot devices. They cannot be reprogrammed. *Erasable programmable read only memory* (EPROM) devices are often a better choice and have become quite popular. Refer to Fig. 11-47, which shows a 2764 EPROM. It is field-programmable and also is field-erasable. Notice that the package has a quartz window over the chip. It is possible to erase this device by exposing it to ultraviolet light at or near a wavelength of 0.2537 μm. The total dose required for complete erasure is typically 15 watt-seconds per square centimeter ($W \cdot s/cm^2$). The erasure is accomplished by using an ultraviolet light source and exposing the IC for the proper length of time. For example, a source might develop 15 mW/cm^2. Dividing gives

$$\frac{15 \text{ W} \cdot \text{s/cm}^2}{0.015 \text{ W/cm}^2} = 1000 \text{ s} = 16.7 \text{ min}$$

All the stored bits go to logic 1 when the device is erased. The light enters the silicon crystal and releases the charges that were trapped there to represent logic 0s. Usually EPROMs can be programmed, erased, and reprogrammed 100 times or more.



Pins / Mode	$\overline{CE}$ (20)	$\overline{OE}$ (22)	$\overline{PGM}$ (27)	V_{PP} (1)	V_{CC} (28)	Outputs (11 ~ 13, 15 ~ 19)
Read	V_{IL}	V_{IL}	V_{IH}	V_{CC}	V_{CC}	D_{out}
Stand by	V_{IH}	X	X	V_{CC}	V_{CC}	High Z
Program	V_{IL}	X	V_{IL}	V_{PP}	V_{CC}	D_{in}
Program verify	V_{IL}	V_{IL}	V_{IH}	V_{PP}	V_{CC}	D_{out}
Program inhibit	V_{IH}	X	X	V_{PP}	V_{CC}	High Z

(d)

Fig. 11-47 2764 EPROM. (*a*) Pictorial. (*b*) Pinout. (*c*) Block diagram. (*d*) Mode selection.

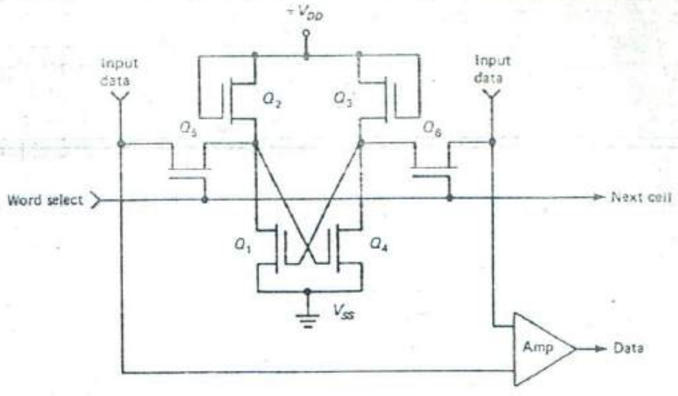
The 2764 EPROM shown in Fig. 11-47 is arranged as 8192 bytes. Therefore, it contains 64K bits, and its part number follows the pattern mentioned earlier. Other EPROMs include the 2708 (8K bits), the 2716 (16K bits), the 2732 (32K bits), and the 27128 (128K bits) and the 27256 (256K bits). The 2764 has 13 address pins and 8 output pins. It also has pins for the application of a programming voltage (V_{PP} = +21 V), a program control pin (NOT PGM), an output enable pin (NOT OE), and a chip enable pin (NOT CE). A high voltage is required during programming to trap charges in the silicon. Some EPROMs require +25 V for programming. In Fig. 11-47(*d*) the programming mode is illustrated. Programming is usually performed by placing the EPROM in a programmer or burner. The chip enable pin is held low, and +21 V is applied to the V_{PP} pin. The V_{CC} is +5 V. Stable address and data are supplied to the address and data pins, respectively. The PGM pin is pulsed low for 50 ms. This process is repeated for every address and takes about 7 min if all 8192 bytes are programmed. Once programming is complete, the programming voltage is reduced to V_{CC}, which is +5 V. Once programmed, the IC serves as an ordinary ROM. A piece of opaque tape should be placed over the window. The access time ranges from 200 to 450 ns. A suffix is usually added to the part number to indicate the access time.

Electrically erasable read only memories (EEROMs) are also available but are less popular than the EPROMs because they are more expensive. They are completely erased by applying a single pulse to the program pin while +25 V is applied to the V_{PP} pin and the chip select pin is low. The EEROMs can be erased very quickly and lend themselves to applications requiring that data be changed frequently. Equipment that is designed to use EEROMs often allows erasure and programming to occur without removing the ROM from the equipment.

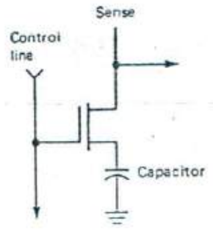
None of the memory devices discussed to this point lends itself to rapid storage of data; devices of this type are known as *random access memory* (RAM) devices. This is an unfortunate descriptor since all of the ROM devices are also random access memory. *Random access* simply means that the data can be read (or written) in any order. For example, if a RAM device stores 8K bytes, any one of the 8192 bytes can be read at any time. A better descriptor for the type of memory device to be discussed now would be *read/write memory*. This is not what it is popularly called, however. Please remember that RAM really refers to memory that can be written to as easily as it can be read. Also remember that the data stored in ROMs are also randomly accessible.

The RAM devices bring on a new problem: volatility. A memory is *volatile* if its contents will be lost on power down. All of the ROM devices are nonvolatile. They will hold their data forever unless they are intentionally erased. All RAM devices are volatile. If the power is lost, even for a short time, all contents will be lost. For this reason, some volatile memory circuits must be provided with battery backup power. This is never necessary with ROMs.

In spite of the volatility of RAMs, they are very widely applied devices in modern digital systems. In fact, there is no practical way to accomplish many operations without them. They provide fast and in-



(a) Static storage cell



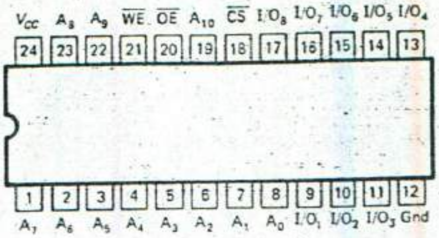
(b) Dynamic storage cell

Fig. 11-48 Metallic oxide semiconductor memory cell types. (a) Static storage cell. (b) Dynamic storage.

expensive storage of data, variables, and instructions. Most RAMs are based on one of two *metallic oxide semiconductor* (MOS) circuit structures. Figure 11-48 shows both memory cell structures; in Fig. 11-48(*a*) is the static RAM storage cell. It is based on six MOS N-channel enhancement mode transistors. These are normally off devices that can be turned on by applying a positive gate voltage. Transistors Q_1 and Q_4 are cross-coupled to form a bi-stable latch. Transistors Q_2 and Q_3 serve as load resistors. Transistors take up less space in the chip than resistors. Suppose Q_1 is on and Q_4 is off. This is one of the two stable states for the latch. With Q_1 on, it drops very little voltage from source to drain. Therefore, its drain voltage is near 0 V, and this voltage is applied to the gate of Q_4. With near zero gate voltage, Q_4 remains off and drops the entire supply from its source to drain. Thus, the drain voltage at Q_4 is equal to $+V_{DD}$ and is applied to Q_1, which keeps it on. It should be clear now that the latch is stable in this condition and can also be stable in the opposite condition. Transistors Q_5 and Q_6 are used to write to and read from the latch. When the select line goes positive, they are turned on. This connects the latch outputs to the sense amplifier for a read operation. It also allows input data to set or reset the latch.

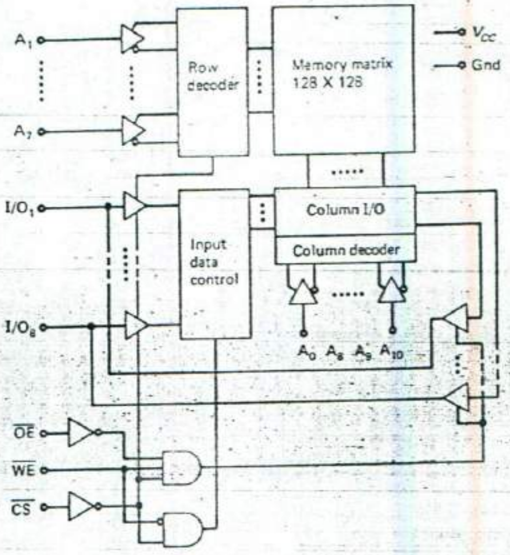
Now look at Fig. 11-48(*b*), which shows the simplicity of the dynamic storage cell. It is based on a single MOS transistor and a storage capacitor. A dynamic cell is read by sending the control line high and reading the sense voltage. If the capacitor is charged, the sense voltage will be high. If the capacitor is discharged, the sense voltage will be at ground potential. The cell can be written to by taking the control line high and applying a positive voltage or zero voltage to the sense line, which will charge or discharge the storage capacitor. Because the dynamic cell is so simple, dynamic RAMs are capable of much more storage in a given size chip of silicon. Unfortunately, the dynamic cells must be refreshed every several milliseconds. A cell is *refreshed* by reading its contents and recharging the capacitor if it was charged. If it is not, data will be lost as the charge in the storage capacitor leaks off. Refresh circuitry will be required when dynamic RAMs are used. This requirement complicates the design of a system, and static RAMs are easier to use when a moderate amount of memory is required. Static RAMs are also chosen when battery back-up is required since refresh circuits are not necessary.

A 6116 static RAM IC is shown in Fig. 11-49. It is a CMOS device arranged as 2048 × 8 bits. Other popular sizes include 1024 × 4, 4096 × 1, and 16,384 × 1. The 6116 has 11 address lines and 8 I/O lines. The access time ranges from 120 to 200 ns and is designated by a suffix after the part number. The standby power ranges from 10 to 100 μW and is also designated by a suffix. The 10-μW version is ideally



V_{CC} A_8 A_9 $\overline{WE}$ $\overline{OE}$ A_{10} $\overline{CS}$ I/O_8 I/O_7 I/O_6 I/O_5 I/O_4
24 23 22 21 20 19 18 17 16 15 14 13
1 2 3 4 5 6 7 8 9 10 11 12
A_7 A_6 A_5 A_4 A_3 A_2 A_1 A_0 I/O_1 I/O_2 I/O_3 Gnd

(a)



(b)

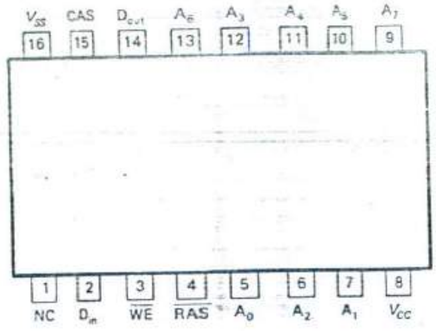
$\overline{CS}$	$\overline{OE}$	$\overline{WE}$	Mode	V_{CC} Current	I/O Pin	Reference Cycle
H	X	X	Not selected	I_{SB}, I_{SB1}	High Z	
L	L	H	Read	I_{CC}	D_{out}	Read cycle (1) ~ (3)
L	H	L	Write	I_{CC}	D_{in}	Write cycle (1)
L	L	L	Write	I_{CC}	D_{in}	Write cycle (2)

(c)

Fig. 11-49 6116 static RAM. (*a*) Pinout. (*b*) Functional block diagram. (*c*) Truth table.

suited for battery back-up service. When the supply voltage drops below 4.5 V, the device goes into the data retention mode. The supply must be maintained at 2 V or more for the data to be retained. Also, the NOT chip select pin must be maintained no less than 0.2 V below the supply voltage in the data retention mode. The truth table in Fig. 11-49(*c*) shows the modes of operation. When the NOT chip select is high, the IC is in standby, and the I/O pins are high-impedance. When the NOT chip select is low, the NOT output enable is low, and the NOT write enable is high, it is in the read mode, and the I/O pins become data outputs. Taking the NOT write enable pin low places the memory IC in the write mode. The I/O pins become data inputs. There are two write cycles, and the state of the NOT chip enable determines which will be selected. Write cycle (2) is used if the NOT chip enable is continuously low. In practice, either type of write cycle may be used in a system.

Figure 11-50 represents a 4164 dynamic RAM. It is an MOS device and is arranged as 65,536 × 1-bit. Other dynamic RAMs include 4096 × 1, 16,384 × 1, and 262,144 × 1 bit. Common access times range from 120 to 250 ns. Note that the 4164 has only 8 address pins; 16 address bits are required to decode



Pin(s)	Designation
A_0–A_7	Address inputs
$\overline{CAS}$	Column address strobe
D_{in}	Data in
D_{out}	Data out
$\overline{RAS}$	Row address strobe
$\overline{WE}$	Read/write input
V_{CC}	+5 V
V_{SS}	Ground

Fig. 11-50 4164 dynamic RAM.

1 of 65,536 locations. Eight-row address bits are set up on A_0 through A_7 and latched with the *row address strobe* (NOT RAS). Then eight column addresses are set up on pins A_0 through A_7 and latched with the *column address strobe* (NOT CAS). Read or write is selected with the *write enable* (NOT WE) input. A high selects the read mode, and a low selects the write mode. If NOT WE goes low prior to NOT CAS, the D_{out} pin will remain in the high-impedance state. A refresh operation must take place at least every 4 ms. Strobing each of the 256 row addresses refreshes every bit. The row address strobe must be low (active) during the refresh cycle. The column address strobe may remain high (inactive) during the refresh cycle.

Bipolar RAMs are also available but are less popular because they use many times more power per bit than MOS and CMOS devices. They are also more expensive because the bipolar process cannot achieve the high bit densities achieved by the other type. They are used in those applications in which speed is the major consideration. Some bipolar RAMs have access times less than 10 ns. Some common sizes of bipolar RAM ICs include 256 × 1, 256 × 4, 1024 × 1, 1024 × 4, and 4096 × 1.

REVIEW QUESTIONS

45. What would have to be done to the ROM circuit shown in Fig. 11-45 to change the last stored word to 1111_2?

46. A ROM is organized as 4096 × 8 bits. What is its capacity in k-bytes?

47. A ROM has a capacity of 1 k-byte. What is the actual number of bits that it has stored?

48. A ROM has 11 address pins and 4 data pins. What is the actual number of bits that it has stored?

49. Factory-programmed ROMs are also called ______ ROMs.

50. What is the minimum number of 65,536 × 4-bit ROMs required for a system that uses 16-bit words?

51. Refer to Fig. 11-50. How can 65,536 locations be accessed with only 8 address pins?

11-8 LOGIC FAMILIES AND INTERFACING

The LS TTL subfamily and CMOS have become the most popular logic families; however, a few others are worth mentioning. *Emitter coupled logic* (ECL) is a family of devices used mainly in large machines requiring very-high-speed operation. The ECL gates have propagation delays as short as 1 ns. It is a nonsaturated logic in which bipolar current-switching devices operate in a circuit configuration similar to that of a differential amplifier. Because the devices are never in saturation, the turn-off delays associated with stored carriers are mostly eliminated. Unfortunately, ECL devices are "power hogs" and typically consume 25 to 65 mW per gate. The power supply must provide −5.2 V with respect to ground. This characteristic is also a disadvantage because positive supply systems are more popular and because interfacing ECL to other families requires extra level shifting components.

There are also PMOS and NMOS digital ICs available. P-channel MOS (PMOS) is based on P-channel MOS transistors and is the older of the two technologies. It requires a +5- and a −12-V supply and is not very popular today. N-channel MOS (NMOS) devices are based on N-channel MOS transistors and operate from a single +5-V supply. They are usually capable of driving at least one LSTTL load and are directly compatible with CMOS. The current NMOS devices are usually LSI or VLSI devices such as microprocessors (discussed in Chapter 12).

The CMOS devices belong to an ideal logic family. It features very low power consumption, excellent noise immunity, almost unlimited fanout, noncritical power supply requirements, and a very wide selection of devices. It is not so fast as LS TTL, but improvements are being made. Several manufacturers have announced high-speed CMOS parts. Its low power consumption is probably its most outstanding characteristic and is only 10 nanowatts (nW) per gate under static conditions and 10 mW per gate when toggling at a 1-MHz rate. It offers a 50:1 to 100:1

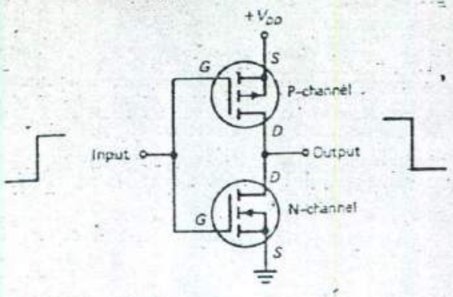


Fig. 11-51 Basic CMOS inverter.

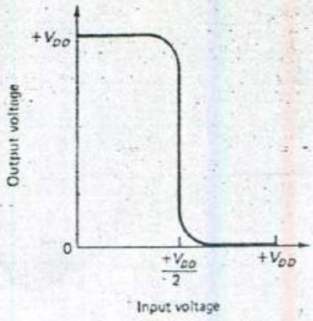


Fig. 11-52 Complementary metallic oxide semiconductor (CMOS) inverter transfer function.

power saving when compared to LS TTL. The noise margin is better than 2 V when operated from a +5-V supply, as compared to LS TTL, which is only guaranteed at 0.4 V. The fanout is almost unlimited since the gate inputs are MOS transistors with an input resistance of 10^{12} Ω. However, each transistor does represent a load capacitance of 5 pF; thus extremely large fanouts could cause switching delays due to capacitive loading (fanouts up to 50 are considered practical). The CMOS devices operate over a power supply range of 3 to 15 V with no need for tight voltage regulation. Speed does improve at the higher voltages with typical maximum toggle rates of 2.5 MHz at 5 V and 7 MHz at 10 V. A browse through CMOS reference manuals will show all of the standard logic devices, from gates through microprocessors. There are also many special devices, including phase-locked loops, communications devices, linear devices (such as BI-FET op-amps), and speed controllers.

The configuration for a basic CMOS inverter is illustrated by Fig. 11-51. It shows that a complementary pair of transistors is used: one P-channel and one N-channel. This complementary pairing is the source of the term *complementary metallic oxide semiconductor* (CMOS). The transistors are enhancement mode devices. You should recall that enhancement mode transistors are normally off. They are turned on by attracting carriers into the channel region. In the case of the P-channel unit, a negative gate voltage will attract holes and turn the device on. In the case of the N-channel unit, a positive gate voltage will be required to attract electrons into the channel and turn it on. Assume a +5-V V_{DD} supply in Fig. 11-51 and a logic swing at the input of 0 to +5 V. When the input is 0, the P-channel transistor is on because 0 V is 5 V negative with respect to the source of the transistor, which is at +5 V. With the P-channel transistor on, the output will be pulled high, to near +5 V. No current flows in the output when it drives other CMOS inputs. No current flows in the complementary pair since the N-channel transistor is off. Now, assume that the input goes to +5 V. This turns the top transistor off and the bottom transistor on. The output is pulled low, to near 0 V. No current flows in the output, and no current flows in the complementary pair because the top transistor is off. Now you can see why CMOS is so power-efficient: no current flows under either static condition. However, there is some current flow during logic transitions. There is a brief period when both transistors are partially on. There is also some momentary output current during transitions due to the charging or discharging of capacitance in the output circuit.

Figure 11-52 illustrates the transfer characteristic for the CMOS inverter. The output voltage switches at an input voltage equal to half the supply. This yields the best possible noise immunity. It also shows that the output voltage swings from $+V_{DD}$ down to 0 V. Older CMOS devices will show a gradually changing output for a gradually changing input. These are known as *A-series devices*. The newer devices have buffered outputs and are known as *B-series*. The added buffers give quicker rise and fall times. They have a constant output impedance of 500 Ω (logic 1 and 0) and are capable of driving one LS TTL load directly when both families are operated from +5 V. Finally, there are some unbuffered devices known as the *UB-series*. They have similar specifications to those of the B-series but are a little faster.

The CMOS input logic levels are shown in Fig. 11-53. Logic 0 extends from 0 to 30 percent of V_{DD}. Logic 1 extends from 70 to 100 percent of V_{DD}. With a 5-V supply, this means that a logic 0 must be 1.5 V or less and that a logic 1 must be 3.5 V or more. If you refer again to Fig. 11-1 you will see that a TTL or LS TTL gate output is not high enough to drive CMOS and meet the requirements of Fig. 11-53. A simple but effective solution is shown by Fig. 11-54. Both families are powered from a +5-V supply. A pull-up resistor is added to the circuit to bring the CMOS input up to near +5 V when the LS TTL gate switches high. This is necessary because of the diode and transistor drop in the totem pole output stage. The resistor should not be too small in value because it will hold the output up too high when it tries to go to logic 0.

Interfacing CMOS to TTL will also require a little

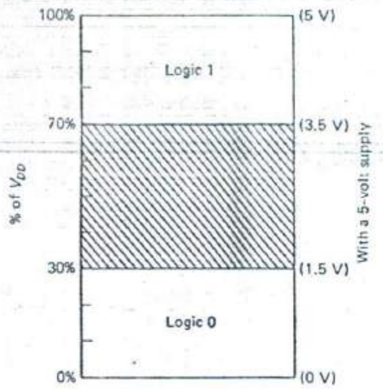


Fig. 11-53 Complementary metallic oxide semiconductor (CMOS) input logic levels.

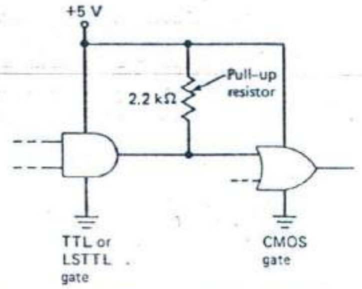


Fig. 11-54 Interfacing TTL (or LSTTL) to CMOS.

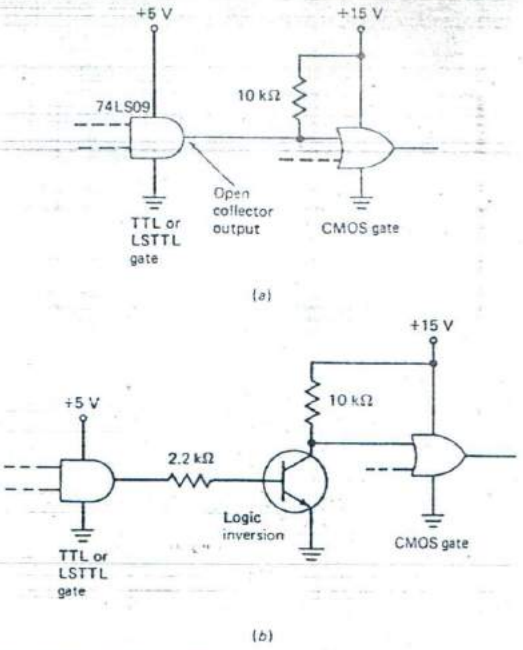


Fig. 11-56 Interfacing different voltage levels. (a) Using open collector output. (b) Using separate transistor.

assistance, as shown in Fig. 11-55. Driving a TTL gate input low requires a sink current of 1.6 mA. If the CMOS gate has an output impedance of 500 Ω, the sink current will cause the output to be $1.6 \times 10^{-3} \times 500 = 0.8$ V. This provides no noise margin for the TTL gate. A pull-down resistor is added to help sink the current. If the CMOS gate is a B-series or UB-series device, it will drive (sink) one LS TTL load without the need for a pull-down.

We have learned that there are speed advantages to powering CMOS from a voltage higher than 5 V. Figure 11-56 shows two ways to interface TTL or LS TTL to CMOS that is operating at a higher voltage. Figure 11-56(*a*) shows the use of an open collector gate. It also shows that an external pull-up resistor connected to the CMOS supply is required. Figure 11-56(*b*) shows a solution using a separate transistor and two resistors. Again, the collector resistor must connect to the CMOS supply. Notice that the transistor inverts the logic from the TTL or LS TTL gate output. Figure 11-57 shows a CMOS device designed to shift from one voltage level to another. It has two supply pins, V_{CC} and V_{DD}, one for each voltage level. It is designed for 3- to 18-V operation. It will shift CMOS at one voltage level to CMOS at another voltage level (mode 0) and will shift TTL to CMOS at another voltage level (mode 1). Other CMOS devices, called *buffers*, are available for driving TTL and LS TTL loads.

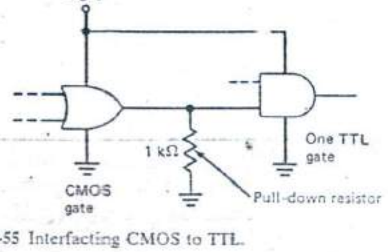


Fig. 11-55 Interfacing CMOS to TTL.

The CMOS part numbers traditionally are grouped in a 4000 series. Other systems closely align with this series, such as Motorola's MC14000 and MC14500 groups. There is also a CMOS group with pinouts identical to those of the TTL and LS TTL parts family; this is the 74C00 series.

Handling CMOS devices demands the static discharge precautions discussed earlier in this book. Almost all CMOS devices have diode protection circuits on their input terminals, but careful handling is still strongly recommended.

Last, but certainly not least, is the matter of floating inputs. Although this practice may work with TTL and LS TTL, it will absolutely not work with

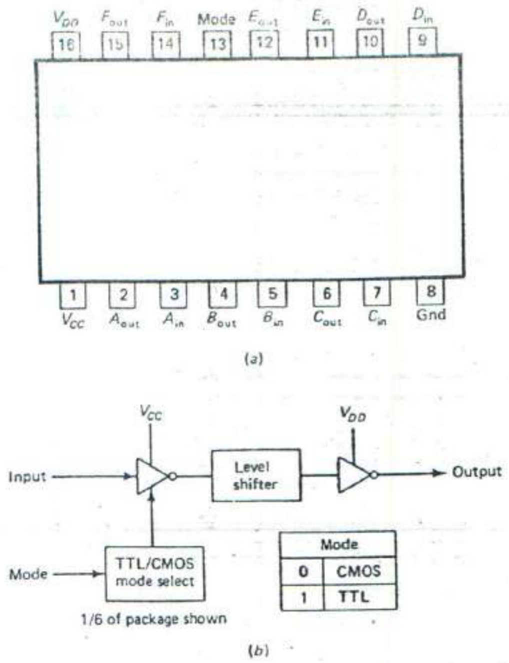


Fig. 11-57 MC14504B hex level shifter. (*a*) Pinout. (*b*) Logic diagram.

CMOS. All inputs that do not connect to gate outputs must be tied to the positive supply or ground. A floating CMOS input can cause insidious problems. The input impedance is so high that time constants of hours are likely. The circuit may work for some time and then fail as an input gradually drifts high or low. Or a floating input may track a driven input in an inconsistent manner. Finally, a floating input can cause a gate to oscillate and draw abnormally high currents from the supply.

REVIEW QUESTIONS

52. Which logic family is the fastest?

53. Refer to Fig. 11-51. The input voltage is equal to V_{DD}. Which transistor is on? What is the logic state of the output?

54. The transistors shown in Fig. 11-51 are normally _______ devices.

55. What is the transition voltage for the input of a CMOS gate operating from a 10-V supply?

11-9 TROUBLESHOOTING AND MAINTENANCE

Always check all the obvious points first. Inspect for improper switch settings, loose connectors, broken cables, and faulty output displays. Never start troubleshooting at the component level until the power supply voltages have been verified. The TTL family permits only a plus or minus 250-mV variation from the nominal 5 V. Power supply ripple should also be checked. A ripple reading of 250 mV or more can cause logic errors and erratic operation. Do not forget to check power supply voltages in several parts of the system. An industrial logic controller may use on-card voltage regulators, and the supply voltages may be good everywhere except on one printed circuit board. A card extender is often a must for troubleshooting in systems with plug-in cards. The cards are too close together for taking any readings. The cards must be removed and replaced with an extender supplied by the equipment manufacturer. The card under test is then plugged into the extender. Never break or make card connections when the power is on.

When troubleshooting robotic and other machine controls, always be wary of the possibility of sudden motion when troubleshooting. This is especially important when the troubleshooting process involves injecting pulses or control signals into circuits. Disconnect or disable high-energy components where feasible. Follow the manufacturer's recommendations for these procedures.

There are two broad categories of defects in digital systems: functional faults and overloads. A *functional fault* is a logic error due to an internal gate failure. For example, both inputs to an AND gate are high, but the output is low. Assuming the output is not overloaded, this is a functional fault, and the gate must be replaced. In an *overload* the gate output is incorrect as a result of a fault after that point. It could be caused by a short circuit in the wiring, the PC board, a connector, or a faulty device driven by the gate. Your knowledge of TTL, LS TTL, and CMOS voltage and current parameters will go a long way in helping you find such faults. For example, you know that a TTL gate can source up to 400 μA while maintaining a logic high of at least 2.4 V. You also know that it can sink up to 16 mA while maintaining a logic low of no more than 0.4 V (LS TTL will only sink up to 8 mA).

How are faults isolated? Usually, the gate output in question must be isolated electrically from the rest of the circuit. This procedure will confirm whether the gate output will respond correctly when unloaded. If the IC is in a socket, it can be removed, and the output pin can be carefully bent out a little. The IC is then replaced in the socket (with the bent pin not going into the socket), and the output is checked for proper response. Do not forget to restore all bent pins when you finish making your measurements. If the IC is soldered in, sometimes the output can be isolated at an edge connector. A sliver of thin tape is used to insulate one of the edge contacts and unload the output in question. Make the tape sliver long enough to wrap part way around the board edge. This will keep the tape from sliding off when the board is inserted into the edge connector. Be very sure to remove all such pieces of tape when you are finished troubleshooting. Another technique is to remove the solder from the output pin in question with a vacuum desoldering tool. Then, a short length of insulating tubing is inserted around the pin and through the circuit board. This procedure takes a little time, but it is better than desoldering and removing the entire device only to learn that there is nothing wrong with it. Be sure to resolder all pins when you are finished. One caution is in order when using any isolation technique: inspect the schematic to be sure that you will not leave a CMOS gate input floating when you isolate any output.

It is necessary to know whether voltages are normal when troubleshooting. In TTL or LS TTL, an output voltage of 0 to 0.07 V is too low. It indicates a defect in the gate or a short circuit to ground somewhere in the system. Likewise, an output voltage near V_{CC} is too high and indicates a short circuit to the positive supply rail. However, if there is an external pull-up, a reading near V_{CC} may be normal. Also, do not forget that an external pull-up resistor may be open and cause no output swing in an open collector device. An output voltage between 2.4 and 0.4 V indicates an overload. The gate may be sourcing or sinking excessive current. Check to see whether the device drives an LED indicator or some other high-current load. If it does, the voltage may be normal, but not if it also drives another gate input. Finally, a floating TTL or LS TTL input will measure between 1.1 and 1.5 V, which is normal.

Circuit board and socket defects are common. A solder splash on a board may not cause problems for months or years and then suddenly "make" an unwanted connection. Always be on the lookout for such problems. An IC may be inserted into a socket with one pin bent under instead of going into the socket. Surprisingly, this pin will sometimes "make" simply by touching the top of the socket pin, but the connection will eventually fail. You have to look very closely to see these kinds of difficulties. Edge connectors, especially on large boards that have a tendency to warp, are an ongoing source of problems. If a control is intermittent and responds to vibration, check the edge connectors. A trace on a PC board may be cracked. These can be impossible to locate by a visual inspection. If a problem comes and goes when a board is flexed, there may be a cracked trace. Try to localize the sensitive spot. A little fresh solder can be flowed over all suspect connections and traces to correct the fault.

Various pieces of special test equipment are available for troubleshooting digital circuits for logic faults. Of course, the standard items such as a multimeter and oscilloscope are very useful. The *logic clip* is a device that clips onto 14- or 16-pin dual-inline ICs. It contains 16 LEDs, 1 for each IC pin. It automatically locates the ground pin and the plus (+) supply pin to energize itself. Some models will work over a 4- to 18-V range and can be used with TTL, LS TTL, NMOS, and CMOS. Logic clips are easy to use and supply a lot of information with one quick and easy connection. A *logic pulser* is another valuable tool; it is a penlike structure with a sharp probe point. It is powered from the system under test by using two clip leads. Some models will work over the 3- to 18-V supply range and cover all the popular logic families. The output of a pulser is tri-state. When its switch is pressed, it automatically drives a gate output or input from high to low or from low to high. It has high source and sink current capability and can therefore override output points originally in the high or low states. It can be used in conjunction with the clip to test flip-flops, latches, registers, etc. For example, the reset or preset pin can be pulsed, and the clip will show whether the device outputs respond as expected. Or the clock input may be pulsed; the device should respond unless it is in an inhibited mode. Logic pulsers are also used in conjunction with another penlike instrument, the *logic probe*. A probe has a built-in light which glows brightly for logic high, dimly for invalid or floating; it goes out for logic low. The pulser is often placed on the gate input and the probe on the gate output; this technique is called *stimulus-response testing*. Or both pulser and probe can be placed at the same point to check for a short circuit to ground or to the supply. If there is a short circuit, the probe will not blink when the switch is pressed on the pulser. The low impedance of the short circuit will not allow the pulser to change the status at that point.

Component-level troubleshooting may be more difficult with LSI and VLSI ICs. They can develop subtle problems that affect one of many possible input conditions. They may also develop timing problems that can only be seen on a multitrace oscilloscope. This is where your general knowledge of the system block diagram is important. When you know the function of a component and the way it interacts with the system, you will be able to pinpoint the trouble to one device or a small group of devices. Substitution with a known good IC is a valid technique when the part is socketed. If the IC is soldered in, it usually pays to take the time to verify as many input conditions as possible before removing it. Verify all signals and power connections at the pins of the device itself, including the ground pin(s). Use a meter and measure from a circuit ground to the ground pin of the device. You should measure 0 V. Remember: circuit boards, sockets, and solder joints do fail.

If operation is erratic, investigate the possibility of noise in parts of the circuit. Check all grounds and shields on cables. Verify all connectors. A pulse of only 20 ns can "glitch" a flip-flop or a register. The oscilloscope is valuable when looking for glitches. Also, logic probes contain pulse-stretching circuitry and will produce a visible flash for pulses as narrow as 10 ns.

The process of CMOS troubleshooting is similar to TTL and LSTTL troubleshooting. Power supply regulation and ripple are less critical, however. Remember the 30 and 70 percent threshold points when taking readings. Also remember that the output impedance of most CMOS gates is 500 Ω whether high or low. A load current of 2 mA will cause a 1-V change (Ohm's law). For example, if the gate is powered from 5 V and sourcing 2 mA, its output will be 4 V. Careful handling is necessary when working with CMOS parts and circuit boards that contain CMOS parts. A board that is removed from the system must be handled with the same caution exercised for handling CMOS ICs (handling precautions were covered in Chapter 2). One additional precaution is that the power should be on when connecting low-impedance test equipment (such as a pulse generator) to CMOS circuitry. Also, remove the test equipment before turning the power off.

REVIEW QUESTIONS

56. What is the allowable supply range for LS TTL?

57. A 0-V reading on the output of an LS TTL gate may indicate that the output circuit is ______.

58. An open collector output measures 0 V when it should be high. The pull-up resistor could be ______.

CHAPTER REVIEW QUESTIONS

11-1. Convert binary 10101111 to decimal.

11-2. Convert hex 3F1D to decimal.

11-3. Convert decimal 98 to binary.

11-4. Convert decimal 3908 to hex.

11-5. Convert binary 1001111 to hex.

11-6. Convert hex 3C1 to binary.

11-7. Convert decimal 128 to BCD.

11-8. Name a nonweighted code used in shaft encoders.

11-9. Suppose that Gray 1000 is applied to the input of Fig. 11-6. What is the binary output? What is the decimal value of the output?

11-10. How many input combinations are there for a six-input OR gate? How many rows of its truth table will show a logic 0 output?

11-11. Two identical square waves are fed into an exclusive OR gate. What is the output wave form?

11-12. Assume a TTL *J-K* flip-flop with both *J* and *K* floating. What is the Q output if a 1-MHz square wave is fed into the clock input?

11-13. How many flip-flops would be required for a modulo 3 counter?

11-14. What is used to reduce the natural modulus of a counter? Can this same technique be used to increase the natural modulus?

11-15. Refer to Fig. 11-26. All four reset pins are at logic 0. What mode is the IC in?

11-16. Refer to Fig. 11-26. All four reset pins are at logic 1. What mode is the IC in?

11-17. How many flip-flops will be required to build a ring counter to control 16 events?

11-18. Refer to Fig. 11-34. How many connections to the display unit would be needed for an eight-digit display?

11-19. What kind of logic circuit is needed to interface a hexadecimal keypad to a digital control?

11-20. Refer to Fig. 11-37. What is the logic state of all outputs when no key is pressed?

11-21. Convert 10101_2 to its 2s complement form.

11-22. Multiply 10111_2 by 1001_2 and state your answer in binary.

11-23. Subtract 00001_2 from 10000_2 and report your answer in binary.

11-24. An EPROM is both field-programmable and field ______.

11-25. How many minutes should an EPROM be exposed to an ultraviolet source that produces 10 mW/cm^2, assuming the recommended dose is 15 W · s/cm^2?

11-26. What is the acronym for a nonvolatile memory device that can be erased by a single electrical pulse?

11-27. What are read/write memory devices popularly called?

11-28. Are ROMs volatile?

11-29. How may RAMs be protected from data loss during power outages?

11-30. What should be added so that an LS TTL gate can drive a CMOS gate operating from the same supply?

11-31. When CMOS drives TTL at the same supply voltage, a ______ resistor is required.

11-32. The CMOS inputs must never be allowed to ______.

11-33. A logic probe is applied to a gate output, and the light glows dimly. The output is ______.

11-34. A logic probe is applied to the ground pin on an IC, and a dim light results. The problem is a defective ______.

11-35. A logic probe is applied to the V_{CC} pin on an IC, and the light goes out. The problem is a defective ______.

11-36. A CMOS gate is sinking 4 mA. What output voltage can be expected?

ANSWERS TO REVIEW QUESTIONS

1. forbidden **2.** 400 mV **3.** with full fanouts **4.** sink **5.** 0.32 V and 2.88 V **6.** 1.8 V **7.** no **8.** no **9.** 11011010 **10.** AND **11.** inclusive **12.** no; propagation delay **13.** 30 ns **14.** waveform *A* **15.** inverted (180° out of phase) **16.** it would not change **17.** pull-up resistors **18.** Q = 0 and NOT Q = 0 (invalid); race **19.** nothing **20.** NAND **21.** yes; preset and clear **22.** there is no window (it samples on the clock edge) **23.** 0.625 MHz **24.** no; it is a ripple counter **25.** no (not in the BCD mode) **26.** they are tri-stated **27.** synchronous **28.** no; the I/O pins **29.** asynchronous **30.** two (0010) **31.** 0000 **32.** on the first positive clock edge **33.** Y6 (pin 9) **34.** ground pin 3 and use Y0 through Y3 as outputs **35.** use G2A or G2B as the input **36.** 005013 **37.** 5010 **38.** true **39.** 10 Hz; 230 Hz **40.** no; yes **41.** 101010 **42.** 4:1 **43.** 11 **44.** 11; same **45.** add three diodes **46.** 4 **47.** 8192 **48.** 8192 **49.** mask **50.** 4 **51.** by using the pins twice with CAS and RAS **52.** ECL **53.** the N-channel transistor; low **54.** off **55.** 5 V **56.** 4.75 to 5.25 V **57.** defective or short-circuited to ground **58.** open

12 MICROPROCESSORS

Microprocessors are among the most elaborate of all integrated circuits. They are very large scale integrated circuit (VLSI) devices, and some of the more sophisticated ones contain over 100,000 transistors. Most microprocessors are housed in the familiar dual-inline package, and pin counts of 28, 40, 48, and 64 are available. Microprocessors are often identified as 4-bit, 8-bit, 16-bit, or 32-bit, according to the size of the data bus. A 4-bit microprocessor has four data pins, an 8-bit microprocessor has eight data pins, and so on. The larger microprocessors are more costly but have the advantage of speed since they can handle more bits at one time. The 8-bit units are very popular because they are fast enough to handle the majority of industrial applications and are low in cost. This chapter will deal mainly with the Motorola MC6809, which is an 8-bit microprocessor.

12-1 OVERVIEW

Microprocessors are very powerful and flexible devices. They can be teamed up with support devices such as memory chips, a keyboard, decoders, video display devices, a cathode ray tube, a mass storage device (disk or tape), and a power supply to form a complete computer. Computers based on microprocessors are called *microcomputers*. Microcomputers are general purpose products. They can be configured to serve in a wide range of applications such as circuit analysis, word processing, games, data-base management, planning, drafting and design, and a host of others. Another way that microprocessors can be used is in a specific application such as motor control. When they are used this way they are called *dedicated microprocessors* or *microcontrollers*. This chapter will emphasize dedicated microprocessors since they are so prevalent in the industrial environment.

Figure 12-1 shows some of the internal details of the Motorola MC6809 microprocessor. It is considered to be one of the most powerful of all 8-bit microprocessors. However, any industrial setting will probably use several types. Much of what you learn about the Motorola MC6809 will transfer to other types of microprocessors. The *arithmetic logic unit* (ALU) is located near the center of the drawing in Fig. 12-1. It represents that section of the processor that is used to perform all arithmetic, logic, and boolean operations. Just above the ALU, we find the *accumulators*, which are each 8 bits wide and are used to hold the intermediate and final results of the various operations. Just below the ALU is the *condition code register* (also called the *flag register*). It is also 8 bits wide, and the various bits (flags) indicate the status of the processor and give information about carries, borrows, and related conditions created by ALU operations. The *data register* is a temporary storage area for the data bus and connects to various sections of the processor such as the decoder. The decoder looks at instructions that come into the processor on the data bus and sends information to the control section. The controller routes signals, times events, and sequences the various registers and circuits. Now, let's look at the *program counter*. It is 16 bits wide and connects to the address register and then on to the address bus, also 16 bits wide. This gives the microprocessor 2^{16} or 65,536 unique addresses, usually designated 64K, as discussed in Chapter 11.

Four other 16-bit registers are shown in Fig. 12-1. These are the index and stack registers. These registers serve mainly as pointers to the 64K address space. They are useful for setting up some sections of memory as data tables and stacks. They are also handy for moving sequential sections of memory around and as 16-bit counters or timers. The last section of Fig. 12-1 to be considered is the *direct page register* (DPR). This is an 8-bit register that makes up the upper byte of a 16-bit address for one of the addressing modes. The various addressing modes will be discussed in a later section.

Even though Fig. 12-1 is grossly simplified, it contains more detail than most people need. Figure 12-2 shows a programmer's model of the MC6809 microprocessor, which indicates those registers that are usable by a programmer. The other internal details are not necessary and are said to be *transparent* from a user's standpoint. The programmer's model shows

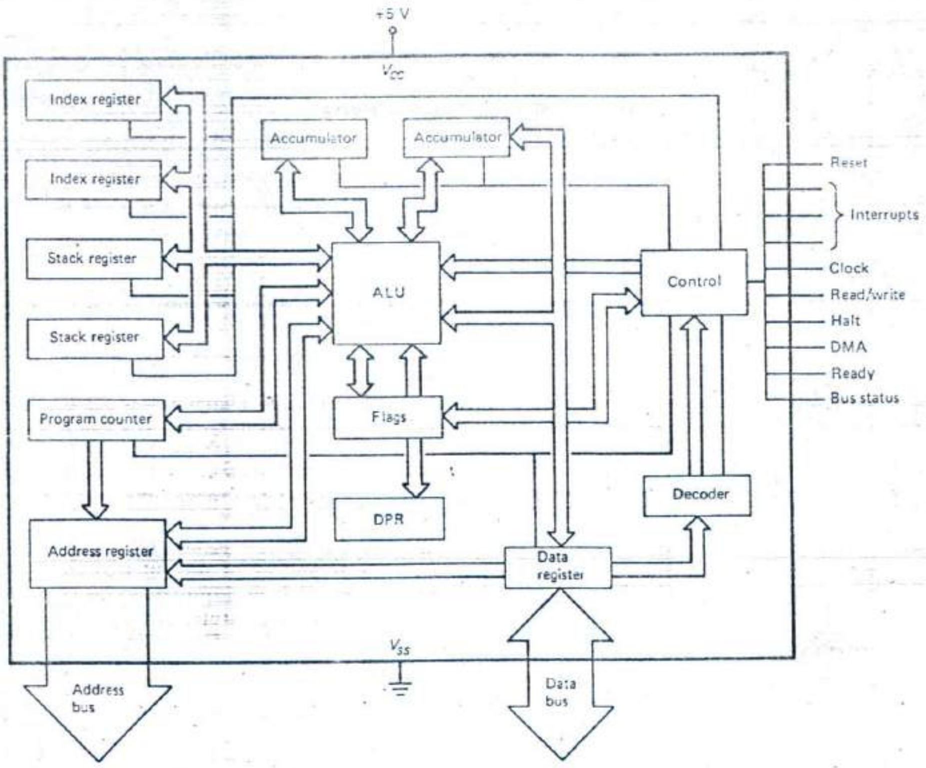


Fig. 12-1 Typical microprocessor architecture.

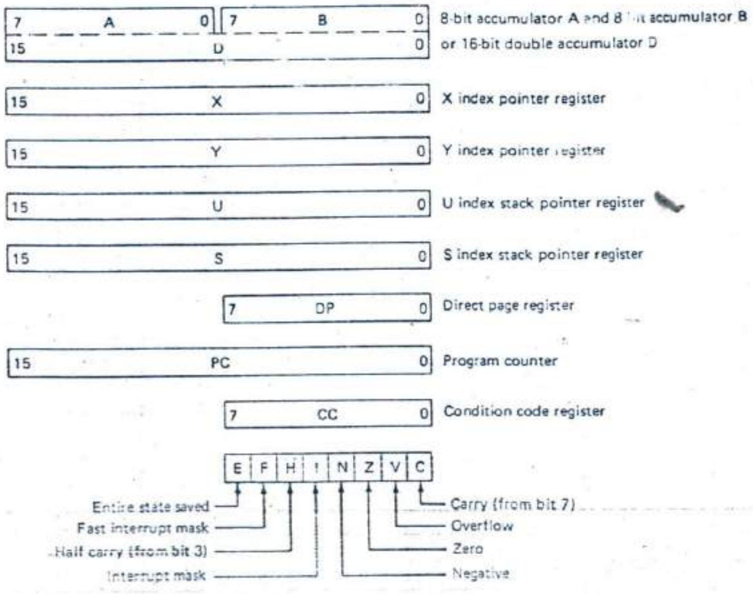


Fig. 12-2 Programmer's model.

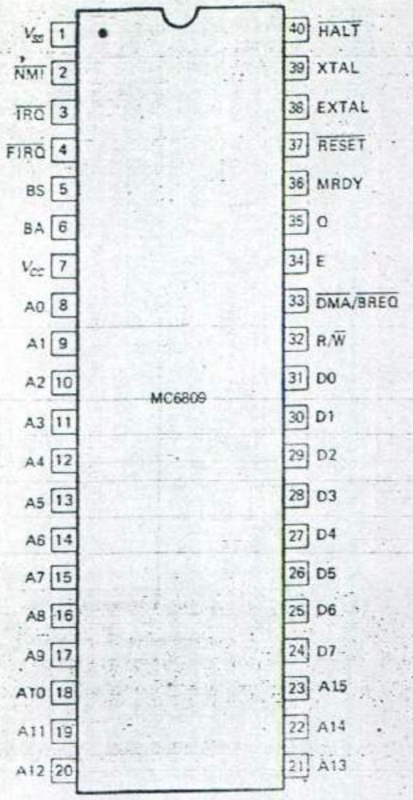


Fig. 12-3 Microprocessor pinout.

two 8-bit accumulators (accumulator A and accumulator B). The two accumulators can be used as a single 16-bit accumulator (accumulator D). It also shows four 16-bit pointer registers and a 16-bit program counter. The direct page register and the condition code register are shown, with each flag identified. The *carry flag* is set when some arithmetic or shift operation generates a ninth bit. It is also set when a larger number is subtracted from a smaller number, indicating a *borrow* in this case. The flags will be discussed in more detail in later sections of this chapter.

The pinout of the MC6809 is illustrated by Fig. 12-3. Some of the pins are straightforward, such as V_{SS}, which is connected to ground; V_{CC}, which is connected to +5 V, and pins 8 through 31, which make up the address bus and the data bus. The *address pins* are outputs only and are used by the microprocessor to select the device or memory location to write data to or to read data from. The *data pins* are bidirectional and are used to transfer data bytes out of or into the processor. Pins 2, 3, and 4 are interrupt inputs. They provide a way for an external hardware device (perhaps a limit switch on a robot arm) to signal to the processor that an important event which requires attention has occurred. Pins 5 and 6 are status outputs that reveal whether the processor is in the normal or running mode, whether it is servicing an interrupt, whether it is in the synchronization mode, or whether it is in the halt mode. Pin 32 is an output and indicates the direction of transfer on the data bus. When the processor puts information on the bus it is writing (pin 32 is low); when it takes information from the bus it is reading (pin 32 is high). Pin 33 is an input used to suspend processor operation and allow another device to take control of the buses for up to a maximum of 15 clock cycles at a time. Pins 34 and 35 are outputs which provide timing (clock) signals to the rest of the system. The Q clock signal phase leads the E clock signal by 90°. Generally, the E clock is used to synchronize the other parts of the system. Pin 36 is an input used by slow devices to stretch the E and Q clock signals to allow more time for data access. Pin 37 is an input used to reset the processor. Pins 38 and 39 are connected to an external crystal that controls the frequency of the internal clock oscillator. Finally, pin 40 is an input that can be used to halt the processor indefinitely. All pins will not be used in every application. For example, the halt pin is often tied to +5 V. Since it is active when low, this will preclude the halt condition.

Microprocessors can be used to control almost anything in industry. They are very flexible because they themselves are controlled by software. *Software* is a group of instructions and data that directs the step-by-step operation of the processor. All that is often required for a new application is new software. For example, suppose we want to use a microprocessor to control the temperature of some industrial process. Also assume that we want to store the highest and the lowest temperatures reached by this process. This is a trivial job for a microprocessor, but it will illustrate the software concept. Refer to Fig. 12-4, which shows a flowchart of the problem. Constructing flowcharts is usually the first step when designing microprocessor software. Enter the chart at the top where it is marked *start*. Rectangles are operation blocks in flowcharts. The first operation is to read the temperature. The next two operations store the temperature just read into the low and high readings. This step initializes both values. Next, the temperature is read again. Then it is compared to a low limit. Diamonds represent decision boxes in flowcharts. Note that there are two exits from the first decision box. If the temperature is less than $70 (the dollar sign is often used to signify a hexadecimal number), we exit the box on the right and encounter a *turn on valve* operation. If the temperature is equal to or greater than $70, we enter the next decision box. The next decision box checks whether the temperature is greater than $90. If it is, the gas valve must be turned off. At this point, you should see that one of three paths must be taken. One path turns the gas on, one path turns the gas off, and one path does nothing to the valves.

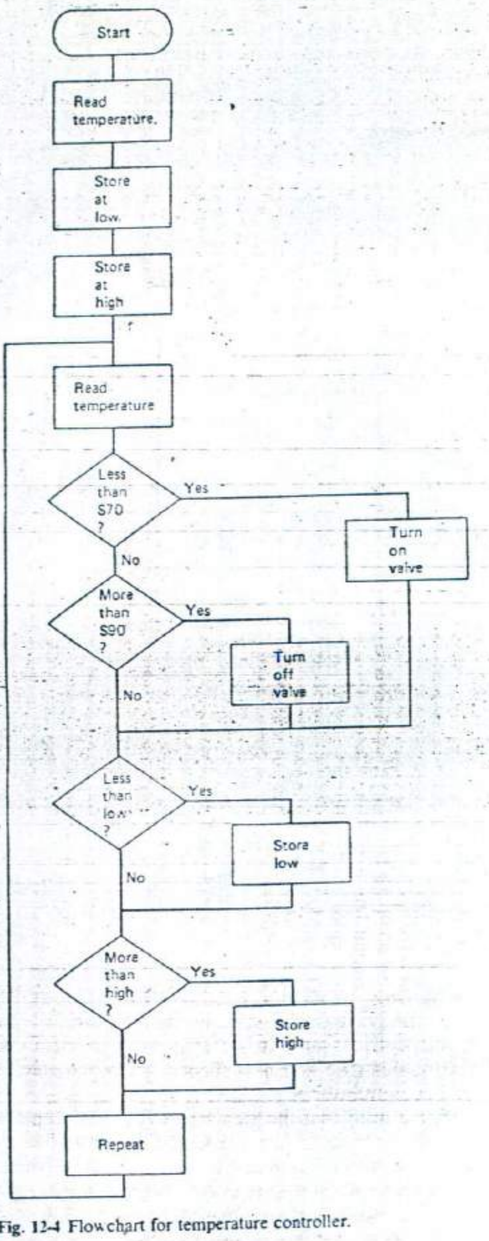


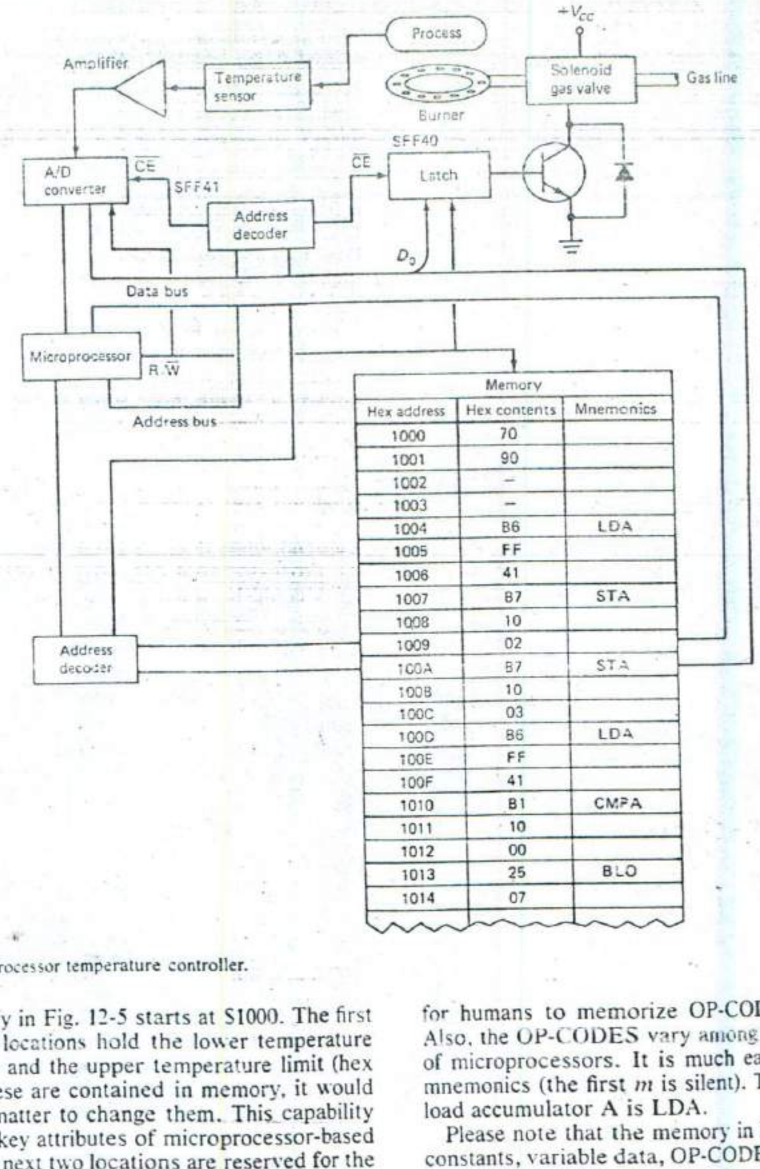
Fig. 12-4 Flowchart for temperature controller.

The next decision box in Fig. 12-4 checks whether the temperature is lower than the previously stored low reading. If it is, the reading is stored, because it represents a new low reading. Next, the temperature is compared to the previously stored high reading. If it is greater, then a new high has been reached, and it must be stored. The repeat box sends the program flow back to take a new reading and make the same decisions again. This program will repeat over and over as an endless loop until a halt signal, or perhaps a reset signal or an interrupt, comes along. There are many possible variations, even in a simple program such as this one. For example, it is really not necessary to check for a high reading if it has been decided that a new low has occurred. The flow from the operation box marked *store low* could go to the repeat box. Another variation is that the second *read temp.* box could be moved to the bottom of the flowchart and replace the *repeat* box. The flow would then leave this box and enter before the first decision box.

It was mentioned that the temperature control program shown in Fig. 12-4 is trivial. It is trivial in that it does not even begin to tap the power of a microprocessor. Much more sophisticated control is possible. For example, the processor could also handle the ignition of the gas burner. It could analyze the exhaust gas and control the air-fuel ratio for better efficiency and reduced pollution. Average temperatures could be calculated and stored, in addition to high and low values. A gas flow meter could supply the processor with consumption data, and cost reports could be generated. A proportional gas valve could be used for tighter control of the process temperature. Software "anticipation" routines could reduce temperature overshoot and undershoot. Safety procedures could be added for overtemperature conditions. There are many, many such features that could be added before the microprocessor would begin approaching its limits. The system would begin to "labor" only when the processor was so busy that its response time for critical actions suffered. The clock rate of the MC6809 is 1 MHz, and one clock cycle takes 1 μs. Therefore, the time required for most operations is very short. For example, reading the temperature will take 5 μs, and 8 μs will be required to execute a decision block. Also available are an MC68A09, which runs at 1.5 MHz, and an MC68B09, which runs at 2 MHz.

Figure 12-5 shows some of the hardware that would be required for the temperature controller. A *solenoid-driven gas valve* allows the processor to turn the gas off and on. By writing a 1 to the latch, the valve is turned on. Writing a 0 to the latch turns the gas off. Note that D0 (least significant bit of the data bus) is sent to the latch for this purpose. An *analog-to-digital* (A/D) *converter* is used to change the analog temperature signal to an 8-bit word. The digital output of the A/D converter is connected to the data bus.

Memory is a key part of any microcontroller. The memory holds the control program and also allows data to be stored. Address decoders are placed on the address bus so the processor can select memory, the A/D converter, or the latch. Each device has a unique address. For example, the address of the latch is $FF40, and the latch is selected (chip enable goes low) when that address appears on the bus. The A/D converter is decoded for $FF41 and is selected when its address appears on the bus.



Memory		
Hex address	Hex contents	Mnemonics
1000	70	
1001	90	
1002	–	
1003	–	
1004	B6	LDA
1005	FF	
1006	41	
1007	B7	STA
1008	10	
1009	02	
100A	B7	STA
100B	10	
100C	03	
100D	B6	LDA
100E	FF	
100F	41	
1010	B1	CMPA
1011	10	
1012	00	
1013	25	BLO
1014	07	

Fig. 12-5 Microprocessor temperature controller.

The memory in Fig. 12-5 starts at $1000. The first two memory locations hold the lower temperature limit (hex 70) and the upper temperature limit (hex 90). Since these are contained in memory, it would be a simple matter to change them. This capability is one of the key attributes of microprocessor-based systems. The next two locations are reserved for the low temperature reading and the high temperature reading. They are marked with a dash since the memory contents here will change as the program runs. The control program itself begins at hex $1004, and the contents here are hex B6 or binary 10110110. Remember, the microprocessor reads the binary contents, and hex is simply a shorthand notation for humans. B6 is an *operation code* (normally abbreviated OP-CODE). It tells the microprocessor to load accumulator A with the contents of the location specified by the next two bytes in memory. Note that the next two bytes are $FF and $41. This means that accumulator A will be loaded from address $FF41, the address of the A/D converter. It is very difficult for humans to memorize OP-CODES such as B6. Also, the OP-CODES vary among the various types of microprocessors. It is much easier to memorize mnemonics (the first *m* is silent). The mnemonic for load accumulator A is LDA.

Please note that the memory in Fig. 12-5 contains constants, variable data, OP-CODES, and addresses. How does the microprocessor know which is which? It doesn't. A microprocessor runs a *fetch-execute cycle*. It must start by fetching an OP-CODE from the correct address. In our example, the program counter must contain $1004 when the first fetch occurs. The contents of location $1004 are loaded into the microprocessor and go on to the decoder since the microprocessor is in a fetch phase. Then $B6 is decoded, and the controller inside the microprocessor advances to the execute phase. It reads the next two bytes in memory and loads them into the address register (refer to Fig. 12-1). The address register now contains $FF41, and the address bus shows that bit pattern. The microprocessor completes the execute phase of the cycle by loading the contents of the data bus (the temperature) into the data register and then into accumulator A. The program counter has, by now, been incremented to $1007, and this value is transferred to the address register. The next fetch begins, and $B7 is loaded into the processor. The OP-CODE for store accumulator A into the location specified by the next two bytes in memory is B7. Note that the next locations contain $10 and $02, so the contents of accumulator A are stored in memory location $1002.

The third fetch phase begins at address $100A. The OP-CODE there is the same as for the prior fetch, but the address information that follows is different, so the contents of accumulator *A* are stored in location $1003. The fourth fetch phase begins at $100D. When executed, the A/D converter is read again. The fifth fetch phase starts at $1010. Hex B1 is the OP-CODE for compare accumulator A with the contents of the location specified by the next two bytes, which are $10 and $00. Therefore, accumulator A is compared with the contents of memory location $1000, which contains $70, the lower temperature limit. The sixth fetch brings $25 into the decoder of the processor; it is the OP-CODE for *branch if lower*. If the temperature is less than the lower limit $70, the processor will branch to the instructions that store a $01 to address $FF40. This will turn the gas on. If the temperature is not less than $70, the program flow will continue on to the next compare instruction that checks whether the temperature is above the upper limit.

What happens if a microprocessor fetches from the wrong place? For example, suppose the program

LISTING 12-1 TEMPERATURE-CONTROL PROGRAM LISTING

```
                                 *TEMP CONTROL PROGRAM
1000                                          ORG     $1000
               FF41              SENSE        EQU     $FF41
               FF40              VALVE        EQU     $FF40
1000           70                LTL          FCB     $70
1001           90                UTL          FCB     $90
1002                             LOW          RMB     $01
1003                             HIGH         RMB     $01
1004  B6       FF41                           LDA     SENSE
1007  B7       1002                           STA     LOW
100A  B7       1003                           STA     HIGH
100D  B6       FF41              READ         LDA     SENSE
1010  B1       1000                           CMPA    LTL
1013  25       07                             BLO     ON
1015  B1       1001                           CMPA    UTL
1018  22       09                             BHI     OFF
101A  20       0A                             BRA     TEST
101C  C6       01                ON           LDB     #$01
101E  F7       FF40                           STB     VALVE
1021  20       03                             BRA     TEST
1023  7F       FF40              OFF          CLR     VALVE
1026  B1       1002              TEST         CMPA    LOW
1029  25       07                             BLO     STL
102B  B1       1003                           CMPA    HIGH
102E  22       07                             BHI     STH
1030  20       DB                REPEAT       BRA     READ
1032  B7       1002              STL          STA     LOW
1035  20       F9                             BRA     REPEAT
1037  B7       1003              STH          STA     HIGH
103A  20       F4                             BRA     REPEAT
               0000                           END
00000  TOTAL ERRORS
HIGH      1003
LOW       1002
LTL       1000
OFF       1023
ON        101C
READ      100D
REPEAT    1030
SENSE     FF41
STH       1037
STL       1032
TEST      1026
UTL       1001
VALVE     FF40
```

LISTING 12-2 BASIC TEMPERATURE CONTROL PROGRAM LISTING

```
10 CLS: REM CLEAR CATHODE RAY TUBE
20 PRINT@130,"LOW TEMP": REM PRINT LABEL ON C.R.T.
30 PRINT@162,"HIGH TEMP": REM PRINT LABEL ON C.R.T.
40 T = PEEK(65345): REM READ TEMPERATURE
50 TL = T: TH = T: REM INITIALIZE LOW AND HIGH VALUES
60 PRINT@140,STR$(TL): REM STORE TO C.R.T.
70 PRINT@172,STR$(TH): REM STORE TO C.R.T.
80 T = PEEK(65345): REM READ TEMPERATURE
90 IF T<112 THEN POKE 65344,1: GOTO 110: REM TURN VALVE ON
100 IF T>144 THEN POKE 65344,0: REM TURN VALVE OFF
110 IF T<TL THEN TL=T ELSE GOTO 130: REM CHECK FOR NEW LOW
120 PRINT@140,STR$(TL): REM PRINT NEW LOW ON C.R.T.
130 IF T>TH THEN TH=T ELSE GOTO 150: REM CHECK FOR NEW HIGH
140 PRINT@172,STR$(TH): REM PRINT NEW HIGH
150 GOTO 80: REM REPEAT
```

counter contains the address of the A/D converter, and the fetch phase is begun. The processor has no inherent intelligence and will attempt to decode whatever byte the converter happens to place on the data bus. Obviously, the results will be unpredictable. When this happens, the system "crashes." Control is usually lost, and a reset is required to restore operation. It is the responsibility of the programmer to arrange memory contents so that OP-CODES are fetched when they are supposed to be. The program must also begin operating at the correct place. If the program in Fig. 12-5 is executed starting at $1000, unpredictable results will occur and the system will crash. A crash will not damage the processor or the memory devices, but it could cause severe consequences in the industrial environment. For example, the gas valve could be turned on and stay on indefinitely.

It should be obvious by now that a microprocessor is worthless without a program to run. *Programs* are software, and software is required for any microprocessor application, even the most trivial. How is software written? The first step is a flowchart. The next step is a conversion to machine code (0s and 1s). This can be done by learning the instruction set of the microprocessor and then looking up all the OP-CODES. The OP-CODES are placed into memory along with the correct addresses and data; this is called *hand assembly*. It is error-prone and laborious. A better way to program in machine language is to use a computer program called an *editor/assembler*. Listing 12-1 shows the printout from an editor/assembler used to generate the code for the temperature control program. The editor allows labels that "make sense" to be assigned. The labels help document the program and make it more readable by people. The assembler "knows" all the mnemonics, so there is no need to look up OP-CODES. Use of an editor/assembler is a giant step above hand assembly. Another alternative is to use a high-level language such as BASIC to write the control program. This will work only if the control computer has high-level-language capabilities. Listing 12-2 shows a BASIC program that accomplishes the same temperature control as the machine language program. It is complete with remark (REM) statements that help to document the program and make it easy for a human to understand. Read the listing and see how it compares with the flowchart. Don't forget to convert the decimal values in the BASIC program to hexadecimal so that your comparisons make sense.

BASIC is much easier to learn than machine language. Several versions of BASIC that have been enhanced for industrial applications are available. Although it is easier to read than machine language, it cannot be run unless the microcomputer has this capability. Most dedicated industrial controllers do not. It also adds overhead to the system. BASIC itself is a program and takes up a significant amount of memory space. Last but not least, BASIC is very slow when compared to machine language. The machine language temperature control program runs over 100 times faster than the BASIC control program. This characteristic is not important in the simple temperature control program, but speed is very important in many other industrial applications.

REVIEW QUESTIONS

1. Motorola also manufactures a 16-bit microprocessor with 24 address pins. How many memory locations can it directly access?

2. Another name for the condition code register is the ________ register.

3. The MC6809 index and stack registers have ________ bits.

4. The MC6809 has two ________ bit accumulators.

5. Refer to Fig. 12-5. If accumulator B contains $01, what happens when it is stored to address $FF40?

6. In Fig. 12-5, B6 at memory location $1004 is an example of a(n) ______.

7. Could the program of Fig. 12-5 be run out of ROM? Why?

12-2 ADDRESSING MODES

The MC6809 microprocessor uses six basic addressing modes: inherent, immediate, extended, direct, indexed, and relative. This section describes each of these modes. In each description, the term *effective address* indicates the actual address in memory where data will be fetched or stored or where instruction processing will proceed. Some of the addressing modes require an extra byte after the OP-CODE to provide the required addressing information. This byte is called a *postbyte*.

Inherent addressing is also called *implied addressing* because the effective address is implicit in the instruction itself. For example, one of the MC6809 instruction mnemonics is ABX, which adds accumulator B to the X register and places the sum in the X register. No additional address information is required since it is inherent in the instruction itself. Another example is DAA, which is a decimal addition adjustment to accumulator A. It is used to correct the sum in the accumulator after binary-coded decimal numbers have been added. Some instructions have an inherent mode in addition to other addressing modes. The mnemonic CLR stands for *clear*; when it is used to clear accumulator A or B, the addressing mode is inherent. However, when it is used to clear some memory location, one of the other addressing modes must be used.

The *immediate addressing mode* places the operand (data) in one or two memory locations immediately following the OP-CODE. This mode is used to provide constant data that do not change during operation of the program. As stated, LDA is the mnemonic for load accumulator A. If the immediate addressing mode is used, the next byte after the OP-CODE for LDA must contain the data that are to be loaded into accumulator A. Let's look at a short program segment that uses immediate addressing to load accumulator A with hex F1, accumulator B with hex 7E, and the X register with hex A09E:

Memory Location	Hex Contents	Mnemonic
2000	86	LDA
2001	F1	
2002	C6	LDB
2003	7E	
2004	8E	LDX
2005	A0	
2006	9E	

Note that two bytes follow the OP-CODE for LDX, the mnemonic for load index register X. Since this is a 16-bit register, two data bytes must immediately follow the OP-CODE in memory. The preceding program segment could be shortened by 1 byte by using a load accumulator D (LDD) instruction in place of the LDA and LDB instructions. Recall that accumulator D is a combination of accumulators A and B. The OP-CODE for LDD immediate is CC. Therefore, $CC followed by $F1 and then by $7E would produce the same result.

One form of immediate addressing uses a postbyte to determine which two registers will be manipulated. Table 12-1 shows how the postbyte must be formed for the exchange and transfer instructions. Exchange is a swap operation. For example, if accumulators A and B are exchanged, each will contain what the other contained prior to execution of the instruction. A program segment to swap the accumulators and then the index registers follows:

Memory Location	Hex Contents	Mnemonic
2007	1E	EXG
2008	89	
2009	1E	EXG
200A	12	

Note that the same OP-CODE is used for both exchange instructions. It is the postbyte in each case that tells the processor which two registers are to be exchanged.

The *transfer operation* copies the source register into the destination register. Both registers will contain the same data after the instruction is executed. The following program segment transfers accumulator A into the direct page register and the S stack pointer into the U stack pointer:

Memory Location	Hex Contents	Mnemonic
200B	1F	TFR
200C	8B	
200D	1F	TFR
200E	43	

The source and destination must be in the proper order. Table 12-1 shows that the source code makes up the upper half of the postbyte and that the destination code makes up the lower half of the postbyte. The code for the S pointer register is binary 0100 (hex 4), and the code for the U pointer register is binary 0011 (hex 3). Therefore, the correct postbyte to transfer S to U is hex 43. Only registers of the same size should be manipulated by the exchange and transfer functions.

Table 12-2 shows how the postbyte is formed for push and pull instructions. These instructions are for saving (or recalling) the contents of one or more of the microprocessor registers in a special area of memory called the *stack*. A *push* operation will save the register(s) to memory, and a *pull* will load the register(s) from memory. There are two 16-bit stack pointers in the MC6809. Let's look at a program segment that will store all of the internal registers, except the program counter, to the memory stack

TABLE 12-1 POSTBYTE FORMATION FOR EXCHANGE AND TRANSFER INSTRUCTIONS

b7	b6	b5	b4	b3	b2	b1	b0
SOURCE (R1)				DESTINATION (R2)			

Code	Register	Code	Register
0000	D (A:B)	0101	Program counter
0001	X index	1000	A accumulator
0010	Y index	1001	B accumulator
0011	U stack pointer	1010	Condition code
0100	S stack pointer	1011	Direct page

TABLE 12-2 POSTBYTE FORMATION FOR PUSH AND PULL INSTRUCTIONS

b7	b6	b5	b4	b3	b2	b1	b0
PC	S/U	Y	X	DP	B	A	CC

PC	=	Program counter
S/U	=	Hardware/User stack pointer
Y	=	Y index register
X	=	U index register
DP	=	Direct page register
B	=	B accumulator
A	=	A accumulator
CC	=	Condition code register

pointed to by the U register and then load the registers from the memory stack pointed to by the S register:

Memory Location	Hex Contents	Mnemonic
200F	36	PSHU
2010	7F	
2011	35	PULS
2012	7F	

Table 12-2 shows that bit 6 of the postbyte, if set, will cause either S or U to be pushed or pulled. If U is the pointer, then S will be pushed or pulled. If S is the pointer, then U will be pushed or pulled. The program segment shows a postbyte of $7F; therefore bit 6 is set. The *PSHU instruction* will store the S register on the memory stack. The *PULS instruction* will load the U register from the memory stack.

Extended addressing locates the effective address in the two bytes following the OP-CODE. Suppose we need a program segment that loads accumulator A from memory location $45E1, loads accumulator B from location $5021, and then stores the X register at locations $8000 and $8001:

Memory Location	Hex Contents	Mnemonics
2013	B6	LDA
2014	45	
2015	E1	
2016	F6	LDB
2017	50	
2018	21	
2019	BF	STX
201A	80	
201B	00	

Even though the X register is a 16-bit register, only the first address byte is specified. The microprocessor will store the upper byte in location $8000 and the lower byte in location $8001.

Direct addressing uses a combination of the direct page register and a single byte following the OP-CODE to form the effective address of the operand. For example, if the direct page register contains $90 and the byte following the OP-CODE is $45, the effective address is $9045. The total memory range of the microprocessor can be thought of as 256 pages of 256 bytes each. The direct page register always points to one page of memory. When quite a bit of data must be accessed from one page, the direct mode provides faster access to the locations in that page if the direct page register is pointing there. The following program segment uses immediate addressing to load accumulator A with $90, transfers it to the direct page register, and then uses direct addressing to load both accumulators from that page:

Memory Location	Hex Contents	Mnemonic
201C	86	LDA
201D	90	
201E	1F	TFR
201F	8B	
2020	96	LDA
2021	45	
2022	D6	LDB
2023	99	

Accumulator A is loaded from location $9045, and accumulator B is loaded from location $9099.

The *indexed addressing* mode uses one of the 16-bit pointer registers (X, Y, S, U, and sometimes the program counter) in the calculation of the effective address. There are several variations within the indexed mode, which include constant offset, accumulator offset, autoincrement, autodecrement, indirect, extended indirect, and program counter relative. The indexed addressing modes are the most powerful and are among the key features of the MC6809 microprocessor.

The *constant offset indexed mode* uses a postbyte to identify the pointer register and the offset size. The offset sizes available are zero offset, 5-bit offset, 8-bit offset, and 16-bit offset. Table 12-3 shows the postbyte formation for the indexed addressing modes. A postbyte of binary 11100100 ($E4) means that the offset is zero and the *S* register will point to

TABLE 12-3 POSTBYTE FORMATION FOR INDEXED ADDRESSING MODES

Mode Type	Variation	Direct	Indirect
Constant offset from register (twos complement offset)	No offset.	1RR00100	1RR10100
	5-Bit offset	0RRnnnnn	Defaults to 8-bit
	8-Bit offset	1RR01100	1RR11000
	16-Bit offset	1RR01001	1RR11001
Accumulator offset from register (twos complement offset)	A accumulator offset	1RR00110	1RR10110
	B accumulator offset	1RR00101	1RR10101
	D accumulator offset	1RR01011	1RR11011
Auto increment/decrement from register	Increment by 1	1RR00000	Not allowed
	Increment by 2	1RR00001	1RR10001
	Decrement by 1	1RR00010	Not allowed
	Decrement by 2	1RR00011	1RR10011
Constant offset from program counter	8-Bit offset	1XX01100	1XX11100
	16-Bit offset	1XX01101	1XX11101
Extended indirect	16-Bit address	--------	10011111

R = X, Y, U, or S; X = 00; Y = 01; X = don't care; U = 10; S = 11.

the effective address. A 5-bit offset can be contained in the postbyte in 2s complement form. The total range of a 5-bit 2s complement offset is −16 to +15. The effective address will be anywhere from 16 less to 15 more than the contents of the pointer register. For example, a postbyte of binary 00001111 ($0F) means that the effective address will be decimal 15 greater than the contents of the X register. If the most significant bit (bit 4) of the offset is 1, the offset is negative and the effective address will be less than the contents of the pointer register.

Eight-bit offsets cannot fit into the postbyte and are contained in an offset byte that follows the postbyte. These are also in 2s complement form for a total decimal range of −128 to +127 added to the contents of the pointer register. As an example, the postbyte binary 10101100 ($AC) followed by $38 means that the effective address is $38 (decimal 56) greater than the contents of the Y register. If the most significant bit of an 8-bit offset is high, the offset is negative and the effective address will be less than the contents of the pointer register. Sixteen-bit offsets are also in 2s complement form and provide a decimal range of −32,768 to +32,767 added to the contents of the pointer register. They are contained in 2 bytes following the postbyte.

Accumulator offset uses the contents of accumulator A, B, or D added to the pointer register to calculate the effective address. The number in the accumulator is treated as a 2s complement number; if the most significant bit is high, the number is negative. In this case, the effective address will be less than the contents of the pointer register. Neither the contents of the pointer register nor the accumulator is affected by the calculation. Table 12-3 shows how to form the postbyte for accumulator offset indexed addressing.

The *autoincrement indexed mode* works by determining the effective address from the desired pointer register and then incrementing the pointer by 1 or 2. The *autodecrement mode* first subtracts one or two from the desired pointer register and then produces the effective address. These modes are known as *postincrementing* and *predecremeting*. They are extremely valuable modes for moving lists or tables of data from one area of memory to another. Table 12-3 shows how the postbytes are formed for these addressing modes.

The *indirect addressing mode* points to two memory locations which contain the address of the operand. For example, suppose the postbyte is binary 10010100, and the X register contains $78CE. Table 12-3 shows that this postbyte is for zero offset, indirect with register X serving as the pointer. The microprocessor will fetch the contents of $78CE and $78CF, not as the operand but as the effective address of the operand. If memory location $78CE contains $01 and location $78CF contains $A4, then the operand will be fetched from memory location $01A4. In extended indirect, the effective address is located at the address specified by the two bytes following the postbyte. Suppose the postbyte is $9F, which specifies extended indirect and is followed by $23 and then $12. The contents of memory location $2312 and $2313 will form the effective address.

Program counter relative addressing uses the program counter as the pointer with either an 8-bit or a 16-bit 2s complement offset. The offset is added to the program counter to form the effective address. Table 12-3 shows the postbytes for program counter relative addressing. Either one or two offset bytes must follow the postbyte.

The last addressing mode is the *relative addressing mode*, which is used when branches from the current instruction location to some other location are desired. The branches are relative to the program counter. When the test of a branch condition is true, either a 1- or a 2-byte relative address is added to the program counter. The relative address is in 2s complement form, allowing both forward and backward program branches. A 1-byte relative address is called a *short branch* and allows a total range of −128 to +127. A 2-byte relative address is called a *long branch* and provides a total range of −32,768 to +32,767. The following program segment illustrates how relative addressing can be used to make the program branch backward a number of times until some condition is met. In this application, it is used to send 64 pulses to an output port:

Memory Location	Hex Contents	Mnemonics
2024	8E	LDX
2025	FF	
2026	40	
2027	86	LDA
2028	01	
2029	C6	LDB
202A	40	
202B	A7	STA
202C	84	
202D	6F	CLR
202E	84	
202F	5A	DECB
2030	26	BNE
2031	F9	

This program segment uses the immediate addressing mode to load the index register with the address of the output port and to load accumulator A with $01. It also uses the immediate mode to load accumulator B with $40 (decimal 64), which is the number of output pulses. Next, it stores accumulator A by using the zero offset indexed mode (note the postbyte of $84). Then it clears the same location again, by using the zero offset indexed mode. The output is pulsed by storing 1 and then clearing it. Accumulator B is then *decremented* (DECB), and a *branch if not equal to zero* (BNE) instruction follows. Note that $F9 follows the BNE instruction: it is the relative address. It will cause a backward branch to the STA instruction every time the BNE test is true. Thus, the program will loop back and continue pulsing the output port until accumulator B is decremented to zero. When it does equal zero, the processor will fetch the next OP-CODE from address $2032.

How does the relative address cause a backward branch to $202B? The program counter is pointing to $2032, which is the address for the next fetch. When the branch test is true, the ALU of the microprocessor adds the relative address to the program counter to form the effective address. Let's subtract the destination address from the source address to determine the backward branch:

```
 $2032 ←— SOURCE ADDRESS
 $202B ←— DESTINATION ADDRESS
 -----
   $07 ←— DIFFERENCE
```

Since $B is greater than $2, we must borrow from the next column. Since we are working in hexadecimal, the borrow adds decimal 16 to the first column. Because 16 + 2 = 18 and hex B = 11, the difference is $07. Now we can see that the ALU must subtract 7 from the program counter. This in the range of a short branch, and the relative address will be 1 byte. We learned in the previous chapter that subtraction may be accomplished by changing the subtrahend to a 2s complement number:

```
      07   ←— HEX RELATIVE ADDRESS
00000111   ←— BINARY VALUE
11111000   ←— ONE'S COMPLEMENT
      +1
--------
11111001   ←— TWO'S COMPLEMENT
      F9   ←— HEX VALUE
```

Relative addresses are always in 2s complement form. When the most significant bit is high, a backward branch will occur. When the most significant bit is low, a forward branch will occur.

REVIEW QUESTIONS

8. One of the MC6809 instructions is MUL. It multiplies accumulator A times accumulator B and places the result in accumulator D. What addressing mode does this instruction use?

9. The immediate addressing mode is used to load accumulator B. How many operand bytes must follow the OP-CODE?

10. Refer to Table 12-1. What hex postbyte must follow the EXG OP-CODE to swap the S and U registers?

11. Refer to Table 12-1. What hex postbyte must follow the TFR instruction to transfer the DP register to accumulator B?

12. Refer to Table 12-2. Determine the hex postbyte required to pull the CC register from the stack.

13. Refer to Table 12-2. Assuming a postbyte of $40, which register(s) will be saved to the stack by a PSHU operation?

14. Refer to Table 12-3. What hex postbyte is required to use the 16-bit offset indexed direct addressing mode with the Y register serving as the pointer? What must follow the postbyte in this case?

15. Refer to Table 12-3. What hex postbyte will be required to select the autoincrement by one direct indexed addressing mode with the X register serving as the pointer? When will the X register be incremented?

12-3 INSTRUCTION SET

The Motorola MC6809 microprocessor has 59 different instructions. When these are combined with all of the available addressing modes, well over 1000 different operations are possible. The instruction set can be functionally divided into five categories: 8-bit accumulator and memory instructions, 16-bit accumulator and memory instructions, index register and stack pointer instructions, branch instructions, and miscellaneous instructions.

Before the instructions are examined, a short discussion of signed and unsigned numbers is appropriate. Suppose the bit pattern in an accumulator or some memory location is 11010011. What decimal number does this pattern represent? It is a matter of interpretation. If the program deals only with *positive (unsigned) numbers*, it represents 211 decimal. However, if the program deals with *signed numbers*, it represents −45 decimal. Negative numbers are represented in 2s complement form, and the most significant bit must be high. Please note that the microprocessor treats all numbers in the same way. It is the programmer's responsibility to decide what a particular bit pattern means.

Table 12-4 shows the 8-bit accumulator and memory instructions. There are two types of addition instructions. The first type shown in the table is *add with carry*. This instruction adds a byte from memory to the contents of one of the accumulators plus the contents of the carry flag. If a prior operation has set the carry flag, then the sum will be 1 greater than the memory contents plus the accumulator contents. The second type of add instruction in the table does not add the carry flag. If microprocessors were limited to 8-bit arithmetic, they would not be adequate for many applications. The add with carry instruction allows multiple precision arithmetic. *Multiple precision addition* allows both the augend and addend to be represented by multiple bytes. The least significant bytes of the augend and addend are added first by using the add instruction. The carry flag may or may not be set by this operation. Then the two next most significant bytes are added, using the add with carry instruction, and so must all subsequent bytes. Thus, the precision is limited by available memory and time. It should be obvious that multiple precision operations take more time than single-byte operations.

The ANDA and ANDB instructions shown in Table 12-4 perform a logical AND with the accumulator contents and some memory location and store the result in the accumulator. Logical ANDing can be used to strip bits off a byte: i.e., the lower 4 bits of the ASCII codes for decimal numbers 0 through 9 are weighted to correspond to the numeric value. By stripping off the upper 4 bits, the correct binary value results. To convert ASCII 0 through 9 to binary, the

TABLE 12-4 8-BIT ACCUMULATOR AND MEMORY INSTRUCTIONS

Instruction	Description
ADCA, ADCB	Add memory to accumulator with carry
ADDA, ADDB	Add memory to accumulator
ANDA, ANDB	Add memory with accumulator
ASL, ASLA, ASLB	Arithmetic shift of accumulator or memory left
ASR, ASRA, ASRB	Arithmetic shift of accumulator or memory right
BITA, BITB	Bit test memory with accumulator
CLR, CLRA, CLRB	Clear accumulator or memory location
CMPA, CMPB	Compare memory from accumulator
COM, COMA, COMB	Complement accumulator or memory location
DAA	Decimal adjust A accumulator
DEC, DECA, DECB	Decrement accumulator or memory location
EORA, EORB	Exclusive or memory with accumulator
EXG R1, R2	Exchange R1 with R2 (R1, R2 = A, B, CC, DP)
INC, INCA, INCB	Increment accumulator or memory location
LDA, LDB	Load accumulator from memory
LSL, LSLA, LSLB	Logical shift left accumulator or memory location
LSR, LSRA, LSRB	Logical shift right accumulator or memory location
MUL	Unsigned multiply (A × B → D)
NEG, NEGA, NEGB	Negate accumulator or memory
ORA, ORB	Or memory with accumulator
ROL, ROLA, ROLB	Rotate accumulator or memory left
ROR, RORA, RORB	Rotate accumulator or memory right
SBCA, SBCB	Subtract memory from accumulator with borrow
STA, STB	Store accumulator to memory
SUBA, SUBB	Subtract memory from accumulator
TST, TSTA, TSTB	Test accumulator or memory location
TFR R1, R2	Transfer R1 to R2 (R1, R2 = A, B, CC, DP)

ASCII code can be ANDed with $0F (binary 00001111). The $0F is called a *mask* and *strips off* (sets to 0) the upper 4 bits of the ASCII code. Conversely, the ORA and ORB instructions shown in the table can be used to convert binary to ASCII. Provided that the binary number in the accumulator is 00001001 (decimal 9) or less, ORing with $30 (binary 00110000) will set the 2 bits necessary to convert to ASCII.

The *arithmetic shift left* instructions in Table 12-4 shift the contents of an accumulator or some memory location left by one bit position. Bit 0 is cleared and bit 7, the MSB, shifts into the carry flag. Whatever was in the carry flag is lost. This operation has the effect of doubling the value of the accumulator or memory location up until the point where bits are shifted out and lost. For example, suppose the accumulator contains binary 00010110 before the shift left. This is equal to decimal 22. The accumulator will contain 00101100 after a shift left which is equal to decimal 44. Another shift left will produce 01011000, which is equal to decimal 88, and so on. The *arithmetic shift right* instructions shown in the table shift the number to the right. Bit 0 is shifted into the carry flag, and the prior content of the flag is lost. Bit 7 does not change. This preserves the sign of the number since the most significant bit is the sign bit when 2s complement interpretation is used. If it is set, the number is negative. If it is clear, the number is positive. For example:

```
10101000 ←— CONTENTS BEFORE ASR
11010100 ←— CONTENTS AFTER ASR
```

In 2s complement interpretation, both of the preceding numbers are negative. To find the magnitude of each number, invert every bit and add 1:

```
10101000 ←— TWO'S COMPLEMENT FORM
01010111 ←— INVERT
      -1 ←— ADD ONE
01011000 ←— MAGNITUDE (decimal 88)

11010100 ←— TWO'S COMPLEMENT FORM
00101011 ←— INVERT
      -1 ←— ADD ONE
00101100 ←— MAGNITUDE (decimal 44)
```

In 2s complement interpretation, the number is equal to negative 88 before the shift and is equal to negative 44 after the shift.

The *logic shift left operations* in Table 12-4 do exactly the same thing as the arithmetic shift left instructions already discussed. Motorola made this accommodation to make the MC6809 compatible with a mnemonic used for an earlier microprocessor, the MC6800. The *logic shift right* (LSR) *operation*, however, is different from the arithmetic shift right. The LSR does not preserve the sign bit. A 0 is shifted into bit 7 instead. Bit 0 is shifted into the carry flag as it is for the ASR operation.

Some microprocessor operations do not produce any results other than setting or clearing the appropriate flags. The BITA and BITB operations in Table 12-4 are examples. These operations perform the logical AND of the accumulator contents with some memory location but produce no change in accumulator or memory contents. The only result is that three flags are affected. First, the V flag is always cleared by this operation. Second, the N flag is set if the AND operation sets the most significant bit; otherwise it is cleared. The N flag is the negative flag. Third, the Z flag is set if the result of the AND operation is binary 00000000; otherwise it is cleared. The Z flag is the zero flag. The bit test instructions are often used to test 1 bit of some memory location to see whether it is high or low. For example, the accumulator can be loaded with $01 to test the least significant bit of a memory location. If the bit is high, the Z flag will be cleared. If the bit is low, the Z flag will be set.

The CMPA and CMPB instructions also produce no results other than setting or clearing the appropriate flags. The *compare* instructions subtract the contents of some memory location from one of the accumulators. The N flag is set if the result is negative. The Z flag is set if the result is zero. The C flag is set if a borrow is generated. Finally, the V flag is set if an overflow occurs. An *overflow* refers to 2s complement overflow and not to a borrow (or a carry). For example, if a negative number is subtracted from a positive number, the result should be positive. Let's look at an example of what can happen:

```
 01011111 ←— ACCUMULATOR
-10101010 ←— MEMORY
 10110101 ←— RESULT OF THE CMP
             OPERATION
```

Note that the accumulator contains a positive number (the most significant bit is 0) and that the memory location contains a negative number (the most significant bit is 1). However, the result is negative since its most significant bit is 1. In this case, the compare instruction will set the V flag since 2s complement overflow has occurred. It will also set the C flag since a borrow has also occurred. Two's complement overflow can occur in other operations as well. For example, two positive numbers can be added and produce a negative result because of a carry into the sign bit (bit 7). This will also set the V flag.

The *test instruction* (TST) shown in Table 12-4 also produces no results other than in the flag register. It subtracts zero from some memory location or one of the accumulators. It always clears the V flag. It sets or clears the N and Z flags according to the results. The *complement operation* (COM) does produce a result; it replaces the contents of one of the accumulators or a memory location with its 1s complement value. The *negate operation* (NEG) produces the 2s complement value. The *rotate right instruction* (ROR) rotates all of the bits (memory location or

TABLE 12-5 16-BIT ACCUMULATOR AND MEMORY INSTRUCTIONS

Instruction	Description
ADDD	Add memory to D accumulator
CMPD	Compare memory from D accumulator
EXG D, R	Exchange D with X, Y, S, U, or PC
LDD	Load D accumulator from memory
SEX	Sign Extend B accumulator into A accumulator
STD	Store D accumulator to memory
SUBD	Subtract memory from D accumulator
TFR D, R	Transfer D to X, Y, S, U, or PC
TFR R, D	Transfer X, Y, S, U, or PC to D

accumulator) right through the carry flag. Bit 0 is placed in the carry flag, and the carry flag is placed in bit 7. This instruction simulates a circulating shift register with 9 bits. The *rotate left instruction* (ROL) is similar, but the direction of rotation is reversed.

Table 12-5 shows the 16-bit accumulator and memory instructions. Accumulator D can be added, loaded, stored, subtracted, transferred, compared, and exchanged. When used in conjunction with external memory, two consecutive memory bytes will be affected. The *sign extend operation* (SEX) transforms a 2s complement 8-bit value in accumulator B into a 2s complement 16-bit value in the D accumulator. For example, suppose accumulator B contains binary 11001001 ($C9) before the SEX operation. This represents −55 decimal in 2s complement form. After the SEX operation, the D accumulator will contain 1111111111001001 ($FFC9), which is the 16-bit 2s complement representation of −55 decimal.

Table 12-6 lists the index and stack pointer instructions. The pointer registers can be loaded, stored, transferred, exchanged, and compared. The four *load effective address instructions* (LEA) calculate the effective address from the indexed addressing mode and place an address in one of the 16-bit pointer registers. The calculations that can be performed include adding or subtracting 5-, 8-, or 16-bit constants; adding or subtracting the contents of one of the 8-bit accumulators; and adding or subtracting the contents of the D accumulator.

The branch instructions are shown in Table 12-7. The simple branches test only one flag. For example, the *branch if equal zero instruction* (BEQ) tests the Z flag. If it is set, the branch is implemented by fetching the relative address and adding it to the program counter. The short branch instructions use a 1-byte relative address; the long branch instructions, such as LBEQ, use a 2-byte relative address. Some of the signed branches use a more involved test of the flag register. The *branch if greater than zero* (BGT) checks three flags and is implemented only when the N and V flags are equal and the Z flag is zero. The BGT branch is used after a subtract or compare operation to alter program flow when the signed register contents are greater than the signed memory operand. You should note that some of the

TABLE 12-6 INDEX AND STACK POINTER INSTRUCTIONS

Instruction	Description
CMPS, CMPU	Compare memory from stack pointer
CMPX, CMPY	Compare memory from index register
EXG R1, R2	Exchange D, X, Y, S, U or PC with D, X, Y, S, U or PC
LEAS, LEAU	Load effective address into stack pointer
LEAX, LEAY	Load effective address into index register
LDS, LDU	Load stack pointer from memory
LDX, LDY	Load index register from memory
PSHS	Push A, B, CC, DP, D, X, Y, U, or PC onto hardware stack
PSHU	Push A, B, CC, DP, D, X, Y, X, or PC onto user stack
PULS	Pull A, B, CC, DP, D, X, Y, U, or PC from hardware stack
PULU	Pull A, B, CC, DP, D, X, Y, S, or PC from hardware stack
STS, STU	Store stack pointer to memory
STX, STY	Store index register to memory
TFR R1, R2	Transfer D, X, Y, S, U, or PC to D, X, Y, S, U, or PC
ABX	Add B accumulator to X (unsigned)

TABLE **12-7** BRANCH INSTRUCTIONS

Instruction	Description
	Simple Branches
BEQ. LBEQ	Branch if equal
BNE. LBNE	Branch if not equal
BMI. LBMI	Branch if minus
BPL. LBPL	Branch if plus
BCS. LBCS	Branch if carry set
BCC. LBCC	Branch if carry clear
BVS. LBVS	Branch if overflow set
BVC. LBVC	Branch if overflow clear
	Signed Branches
BGT. LBGT	Branch if greater (signed)
BVS. LBVS	Branch if invalid twos complement result
BGE. LBGE	Branch if greater than or equal (signed)
BEQ. LBEQ	Branch if equal
BNE. LBNE	Branch if not equal
BLE. LBLE	Branch if less than or equal (signed)
BVC. LBVC	Branch if valid twos complement result
BLT. LBLT	Branch if less than (signed)
	Unsigned Branches
BHI. LBHI	Branch if higher (unsigned)
BCC. LBCC	Branch if higher or same (unsigned)
BHS. LBHS	Branch if higher or same (unsigned)
BEQ. LBEQ	Branch if equal
BNE. LBNE	Branch if not equal
BLS. LBLS	Branch if lower or same (unsigned)
BCS. LBCS	Branch if lower (unsigned)
BLO. LBLO	Branch if lower (unsigned)
	Other Branches
BSR. LBSR	Branch to subroutine
BRA. LBRA	Branch always
BRN. LBRN	Branch never

simple branches also appear in the signed branch section since they are also useful for testing signed numbers. The unsigned branches in the table are for testing positive numbers only. The most significant bit is not interpreted as a sign bit. The *branch if higher than instruction* (BHI) tests two flags: carry and zero. If a subtract or compare operation causes neither a carry (borrow) nor a zero result, the branch will be implemented. The other branches shown in Table 12-7 do not test any flags. The *branch to subroutine* (BSR) will be covered in the next section. *Branch always* (BRA) is an unconditional branch; the relative address will always be fetched and added to the program counter. *Branch never* (BRN) does nothing and acts as a 2-byte *no operation* (NOP). The LBRN acts as a 4-byte no operation because the OP-CODE is 2 bytes long and must be followed by a 2-byte relative address. In a program NOPs may be used to reserve memory space for future insertion of code. They also may be placed in a timing loop to consume a few clock cycles. Branch BRN consumes three clock cycles, and LBRN consumes five clock cycles.

The miscellaneous instructions are shown in Table 12-8. The first mnemonic is *ANDCC*, which stands for *logical* AND *the condition code register*. It gives the programmer a way to preserve or clear the flags and uses the immediate addressing mode only. If the CC register is ANDed with $00, all of the flags will be cleared. If it is ANDed with $F0, the upper four will be preserved, and the lower four will be cleared. The CWAI instruction ANDs an immediate byte with the condition code register, stacks all of the microprocessor registers on the S stack, and then waits for an interrupt. It will be covered in more detail in the next section. The no operation (NOP) consumes 1 memory byte and two clock cycles. The ORCC ORs an immediate byte with the condition code reg-

TABLE 12-8 MISCELLANEOUS INSTRUCTIONS

Instruction	Description
ANDCC	AND condition code register
CWAI	AND condition code register, then wait for interrupt
NOP	No operation
ORCC	OR condition code register
JMP	Jump
JSR	Jump to subroutine
RTI	Return from interrupt
RTS	Return from subroutine
SWI, SWI2, SWI3	Software interrupt (absolute indirect)
SYNC	Synchronize with interrupt line

ister and places the results in the CC register. It is used to set flags. The *jump instruction* (JMP) transfers program control to the effective address. It is similar to the BRA instruction but uses extended, direct, or indexed addressing rather than relative addressing. The *jump to subroutine* (JSR) is covered in the next section along with RTI, RTS, SWI, and SYNC.

REVIEW QUESTIONS

16. Suppose an accumulator contains binary 10011100. What unsigned decimal number does this represent? What signed decimal number does it represent?

17. If accumulator A contains $02, the effective address contains $03, and the carry flag is set, what hex number will result in accumulator *A* after an ADDA instruction? After an ADCA instruction?

18. Accumulator A contains $19 before an ASLA instruction. What hex values does it contain after the operation?

19. Suppose accumulator A is loaded with $80 and then the BITA operation is performed at the beginning of a list in memory. This is followed by the BEQ instruction, and then the next location in the list is tested in the same manner. The branch will occur when a ________ is found. Will execution of this segment change any of the list locations or change the $80 in the accumulator?

20. Hex 4F is added to hex 3E. Will this set the V flag?

21. Accumulator B contains $0F before the COMB instruction. What is the hex value in the accumulator after the operation?

12-4 SUBROUTINES AND INTERRUPTS

A *subroutine* is a program segment that is used over and over again. It might handle some arithmetic function, check the position of a motor shaft, read a sensor, or produce a delay. The main program can call a subroutine with the jump to subroutine instruction (JSR), as shown in Fig. 12-6. Notice that the main program calls the subroutine from two different locations. Any number of calls is possible. Also notice that the last instruction in the subroutine ends with the *return from subroutine instruction* (RTS). When the JSR instruction is executed, the contents of the program counter are pushed onto the hardware stack. The hardware stack is located somewhere in memory and is pointed to by the S register. When the RTS instruction is executed, the program counter is pulled from the hardware stack. In this way the main program can be reentered at the correct location. You might be wondering whether the same result can be accomplished with the jump (JMP) instruction. It cannot. As Fig. 12-6 shows, a subroutine may be called from several locations in the main program. The only way to reenter the main program at the proper location is to pull the program counter from the stack.

Subroutines may call other subroutines; this procedure is referred to as *nesting* and is shown in Fig. 12-7. The main program calls subroutine A with the JSR instruction. The address of the next instruction in the main program is $13C9. The low-order byte of this address is pushed on the hardware stack, followed by the high-order byte. At the time of the JSR instruction, the S register was pointing at memory location $1006. The JSR instruction first causes the stack pointer to be decremented by 1. Then the low-order byte of the return address is pushed on the stack. The pointer is decremented again, and the high-order byte is pushed onto the stack. Subroutine A begins executing, and later another JSR is encountered. The next instruction in subroutine A is at $2E10 and this address is pushed on the stack. The S pointer now contains $1002. Subroutine B is executed and it ends with the RTS instruction. This causes 2 bytes to be pulled from the stack and loaded into the program counter. Therefore, the contents of location $1002 is loaded into the high byte of the program counter. The S pointer is incremented, and the contents of stack location $1003 are loaded into

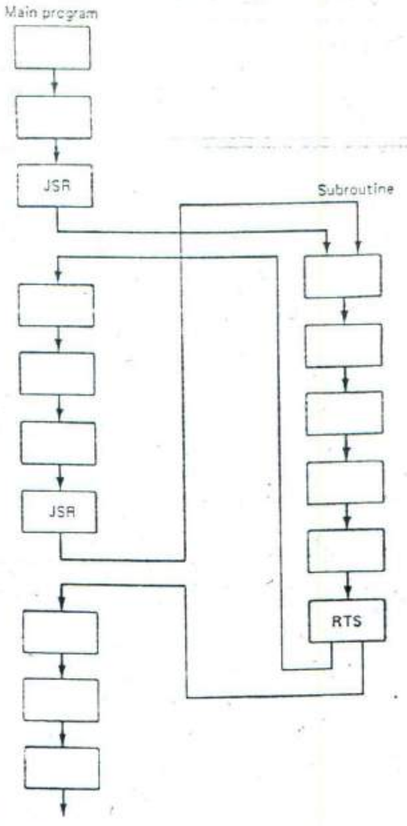


Fig. 12-6 Main program jumps to subroutine.

the low byte of the program counter. The instruction from location $2E10 is fetched, and the rest of subroutine A is executed. The RTS instruction at the end of subroutine A pulls $13C9 into the program counter, and the main program is reentered at the correct location.

Many levels of nesting are possible. For example, subroutine A calls subroutine B, which calls subroutine C, which calls subroutine D, and so on. The only limitation is the memory available for the stack. Figure 12-7 shows that the stack grows as each JSR instruction pushes the program counter. If this occurs too many times, the stack will grow until memory is exhausted or the stack overlaps with a section of memory that is being used for some other purpose. If this happens, the stack may be overwritten, and the processor will be unable to find its way back. The software will "crash."

Stepper motors are covered in Chapter 3. It is possible to use a microprocessor to control a stepper motor. Assume that the motor has been decoded for address $FF40. Also assume that the lower 4 bits of the data bus are latched when this address is written to. The motor windings can be turned on by writing a 1 to this address and turned off by writing a 0. Suppose the required bit pattern for clockwise rotation is as follows:

```
0011
1001
1100
0110
```

After 0110 is written to the motor, 0011 is written again, and the process repeats.

Let's look at a program that will run the stepper motor:

Memory Location	Hex Contents	Mnemonics
7D12	86	LDA
7D13	03	
7D14	C6	LDB
7D15	09	
7D16	B7	STA
7D17	FF	
7D18	40	
7D19	43	COMA
7D1A	BD	JSR
7D1B	7D	
7D1C	26	
7D1D	F7	STB
7D1E	FF	
7D1F	40	
7D20	53	COMB
7D21	BD	JSR
7D22	7D	
7D23	26	
7D24	20	BRA
7D25	F0	

The first two instructions load accumulators A and B with $03 and $09, respectively. These represent the first 2-bit patterns for the motor. Accumulator A is then written to the motor port. You should notice the address of the motor in the 2 bytes following the STA instruction. Accumulator A is then complemented. The 1s complement of $03 is 11111100. Note that the 4 lower bits are the pattern needed for the third motor step. The delay subroutine is called next. A delay is required before the motor can be stepped again. After the delay, the program flow continues with the STB operation. This writes the second bit pattern to the motor. Accumulator B is now completed to produce the bit pattern needed for the fourth motor step. The delay subroutine is called again, and when it is finished the main program flow resumes at the BRA instruction. The relative address of $F0 sends the program back to the STA instruction. The main program runs again and provides the third and fourth bit patterns to the motor. The complement instructions flip the motor bits again so the next run through will be identical to the first.

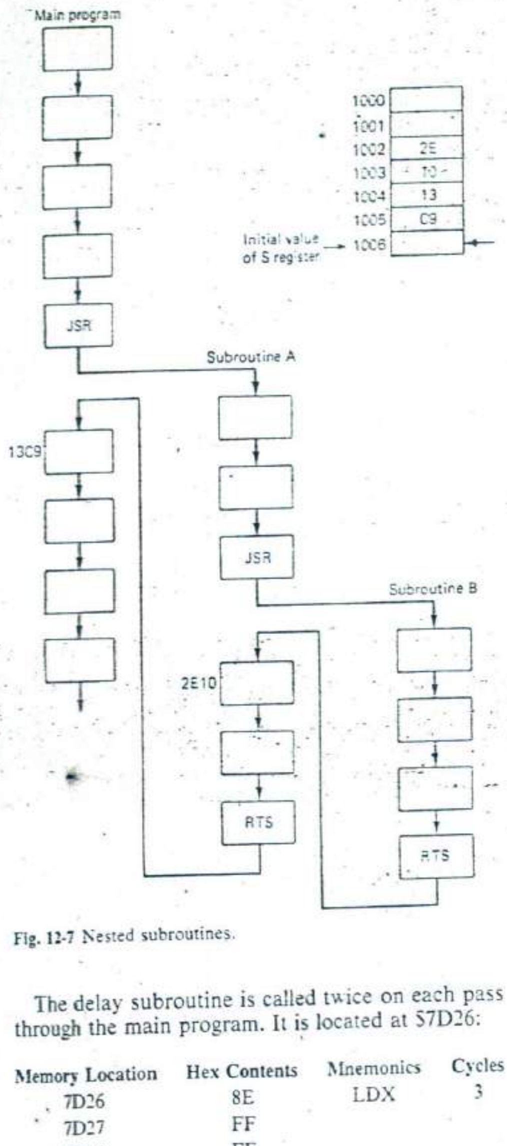


Fig. 12-7 Nested subroutines.

The delay subroutine is called twice on each pass through the main program. It is located at $7D26:

Memory Location	Hex Contents	Mnemonics	Cycles
7D26	8E	LDX	3
7D27	FF		
7D28	FF		
7D29	30	LEAX	4
7D2A	1F		
7D2B	12	NOP	2
7D2C	12	NOP	2
7D2D	12	NOP	2
7D2E	12	NOP	2
7D2F	26	BNE	3
7D30	F8		
7D31	39	RTS	5

The first instruction in the delay subroutine loads the X register immediately with $FFFF. The next instruction loads the X register with the effective address calculated from the 5-bit offset contained in the postbyte $1F. If you refer to Table 12-3, you will see that this provides an offset of binary 11111, which is in 2s complement form and is equal to −1. Thus, the index register is decremented by 1. The four no operations each consume two clock cycles. The branch if not equal zero instruction checks to see that the index register has been decremented to zero. If it has not, the relative address $F8 is fetched, and the subroutine branches back to $7D29. When the index register is decremented to zero, the RTS instruction is fetched, the program counter is pulled from the stack, and the main program is reentered.

How much time will the delay subroutine take? If the processor is running at 1 MHz, each clock cycle is 1 μs. The loop will run $FFFF (decimal 65,535) times. The LEAX instruction consumes 4 cycles, the NOPs 8 cycles, and the BNE 3 cycles for a total of 15 cycles: 65,535 × 15 = 983,025 cycles. The LDX instruction executes once, using another 3 cycles: the RTS executes once, using another five cycles. Thus, the total delay is 983,033 μs.

Because the delay program is a subroutine, it appears in memory only once. Otherwise, it would have to appear twice in the stepper motor program. It is obvious that subroutines save a lot of memory and make programs easier to write. Another advantage is that the delay constant is contained in two adjacent memory locations, $7D27 and $7D28 in our example. These memory locations can be easily changed by another part of the program. The subroutine approach makes changing the speed of the motor easier.

There is also a *branch to subroutine* (BSR) *instruction*. It does about the same thing as JSR but uses relative addressing. Its use is preferred if the program will have to be moved around in memory. In our example program, the JSR instruction refers to an absolute address of $7D26. If the main program and the subroutine are moved to another area of memory, all such absolute references will have to be changed. Programs that refer to absolute memory locations are said to be written in *position-dependent code*. If the two JSR instructions are replaced with BSR instructions, the program becomes *position-independent*. Since branch instructions use relative addressing, the subroutine would be properly called as long as it held the same relative position with reference to the main program. The address of the motor is also absolute, but this is not considered to be a problem since input-output (I/O) ports are usually fixed anyway.

The stepper motor program is trivial in that it barely taps the power of the microprocessor. Many additional features could be included: The motor could be reversed, or the pulses could be counted to allow precise positioning of some mechanism. Complex calculations could be performed on input data to position or control the speed of the motor according to other external conditions. Controlled acceleration could be achieved by shortening the delay time after each pulse. Deceleration would be another possibility. Several motors could be synchronized, and they could be synchronized with other hardware. Any or all of these features are easily within the capability of a microprocessor, and they can be achieved mainly with software.

Microprocessor control of motors provides an accurate and cost-effective solution to many industrial problems. However, something can always go wrong. For example, a motor may stall if the load is too large. A set screw may loosen, or a key may shear, allowing a gear to slip on a shaft. In cases such as these, the mechanism will not be positioned where the processor "thinks" it is. One way to handle potentially damaging and dangerous situations is to use the interrupt capability of the microprocessor. The MC6809 has three hardware interrupt pins: the *nonmaskable interrupt input* (NMI), the *interrupt request input* (IRQ), and the *fast interrupt request input* (FIRQ). A limit switch or an array of limit switches can be placed on the mechanism. If the mechanism exceeds one of these limits, a signal can be sent to one of the interrupt inputs of the processor to alert it to a potentially dangerous condition.

Except for a reset signal, the NMI signal has the highest priority. A negative logic signal applied to this pin demands that a nonmaskable interrupt sequence be generated. As its name indicates, this input cannot be masked by software. Figure 12-8 shows the reaction of the MC6809 to this interrupt signal. The current instruction cycle is finished first. Next, the E flag is set, indicating that the entire internal register set will be pushed to the hardware stack. Assuming that the S register contains \$100F at the time of the interrupt, the registers are pushed, starting at memory location \$100E and ending at location \$1003. Note that the S register itself is not stacked. Next the F and the I flags are set. The F flag masks a fast interrupt request, and the I flag masks an interrupt request. This means that a lower-priority interrupt cannot interfere with the processing of the nonmaskable interrupt. Next, the BA pin on the processor is set low, and the BS pin is set high. Through this logic combination at pins 5 and 6 the processor provides a hardware acknowledgment of the interrupt. Then the NMI vector (a *vector* is an address) is fetched from memory locations \$FFFC and \$FFFD. Suppose the contents at these two locations are \$45 and \$C1, respectively. The program counter will be loaded with \$45C1. Now the BS pin is set low, which signifies the normal or running mode, and the processor will begin executing the program that starts at \$45C1. The program must end with the *return from interrupt instruction* (RTI). Execution of the RTI instruction will pull all of the internal registers from the hardware stack. This will cause the main program to be reentered at the point where the interrupt occurred with all internal registers restored to their original condition.

The interrupt service routine that begins at \$45C1

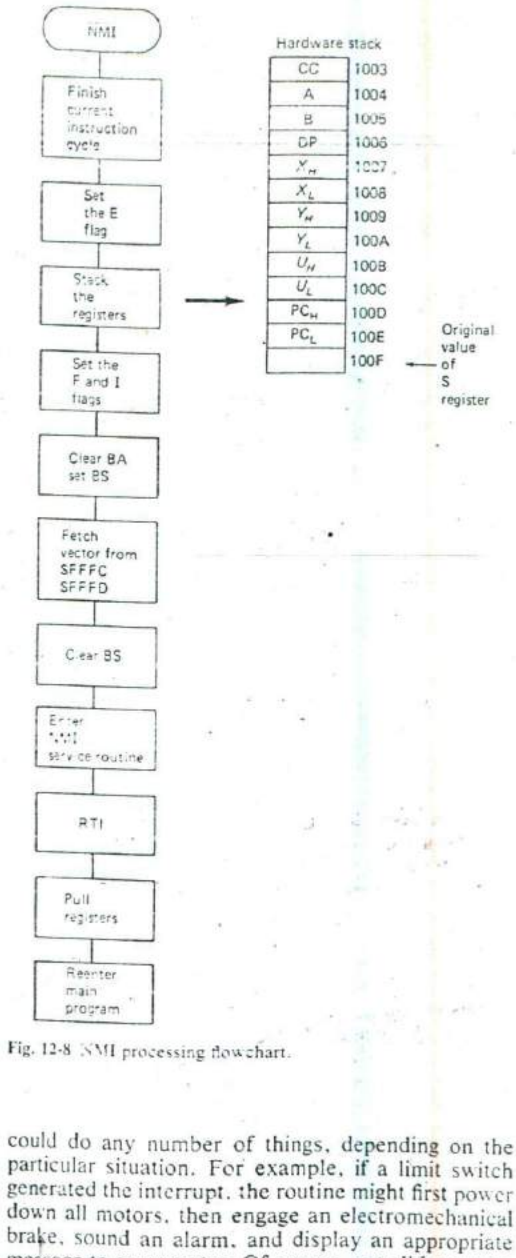


Fig. 12-8 NMI processing flowchart.

could do any number of things, depending on the particular situation. For example, if a limit switch generated the interrupt, the routine might first power down all motors, then engage an electromechanical brake, sound an alarm, and display an appropriate message to an operator. Of course, not all interrupts require such drastic action. An interrupt may be generated by a sensor circuit to alert the processor that data are available at some input port. The service routine may only be required to read the data and store it in memory in this case. Interrupt service routines can be located at almost any location in memory. All that is required is that the proper vectors be stored in high memory. The interrupt vectors for the MC6809 must be stored at the following locations:

$FFFE & $FFFF	←	RESTART VECTOR
$FFFC & $FFFD	←	NMI VECTOR
$FFFA & $FFFB	←	SWI VECTOR
$FFF8 & $FFF9	←	IRQ VECTOR
$FFF6 & $FFF7	←	FIRQ VECTOR
$FFF4 & $FFF5	←	SW12 VECTOR
$FFF2 & FFF3	←	SW13 VECTOR

Even though the vector locations are fixed, those of the various service routines are not. For example, the NMI routine can be located beginning at $01FF by storing this address at $FFFC and $FFFD. The decisions to locate which routines where in memory are normally made early in the design of a microprocessor system, and the interrupt vectors are stored permanently in read-only memory (ROM). The service routines themselves may be located in ROM or in RAM.

An interrupt request (IRQ) will be ignored if the I flag is set. This gives the programmer a way to ensure that some time-sensitive routine will not be interrupted by a low-priority event. If the I flag is clear, a logic zero applied to the IRQ pin will initiate the service routine. This routine is similar to the one shown in Fig. 12-8, except that the F flag is not set, and the vectors are fetched from $FFF8 and $FFF9. A *fast interrupt request* (FIRQ) has a higher priority than an IRQ, which means that an FIRQ signal can interrupt the processing of an IRQ. An FIRQ is also maskable by setting the F flag. If the flag is cleared, an FIRQ signal will initiate the FIRQ service routine. In this case, the E flag is cleared, and only the contents of the program counter and the condition code register are saved to the stack. This saves time and allows the processor to provide quicker interrupt service. Both the F and I flags are set to prevent an IRQ or a second FIRQ from interrupting. The IRQ and FIRQ routines also end with the RTI instruction. In the case of the FIRQ, only the condition code register and the program counter will be pulled from the stack since the E flag is low.

A *reset signal* is used to initialize a microprocessor system. A low-going signal on pin 37 will cause the processor to abort the current instruction cycle. The direct page register is cleared, and the restart vector is fetched from $FFFE & $FFFF.

The MC6809 also has three software interrupts: SWI, SWI2, and SWI3. These work in much the same way as the hardware interrupts but are generated by software. All of the processor registers are pushed on the hardware stack (with the exception of the S pointer itself), and control is transferred through the appropriate vector. Both the I and F flags are set so the processor will ignore IRQ and FIRQ signals while the software interrupt is being processed. Software interrupts are normally used during design and development of the microprocessor system. They can be used to simulate hardware interrupts and are also useful in software debugging.

A microprocessor may be idle. The MC6809 has an instruction with the mnemonic *CWAI*, which ANDS an immediate byte with the condition code register, stacks the entire machine state on the S stack, and waits for an interrupt. The immediate byte can be used to clear the I or the F flags to allow the processor to respond to selected interrupts (but not software interrupts). When an NMI occurs, no further stacking will be required before vectoring off to the service routine. An FIRQ will enter its interrupt routine with the entire processor state saved, and the RTI will automatically return the entire processor state after testing the E flag. The CWAI instruction is used to initiate an idle mode in anticipation of an interrupt. Since the stacking is already complete, the interrupt will be serviced faster. The *SYNC* instruction is similar to CWAI, except that the machine state is not stacked and no vector is fetched. When a SYNC instruction is executed, the processor stops and waits for an interrupt. When an interrupt occurs, the synchronizing state is cleared; processing continues, depending on the condition of the F and I flags.

REVIEW QUESTIONS

22. Is there any limit to the number of locations in a main program where a subroutine can be called?

23. Which internal processor register(s) are stacked in the execution of the JSR or BSR instructions?

24. Which processor register points to the hardware stack?

25. Subroutines called by other subroutines are said to be ______.

26. The S register is ______ every time a byte is pushed onto the hardware stack.

27. What will usually happen if the stack overflows or is overwritten?

28. Refer to the stepper motor program. What would happen if accumulator A were loaded with $09 and accumulator B were loaded with $03?

29. Refer to the delay subroutine. Calculate the total delay if the four NOPs are removed from the program. You may assume a 1-MHz clock.

30. Identify the mnemonic of the instruction that should be used to call subroutines if the program must run from several areas of memory.

12-5 SYSTEM DESIGN

A microprocessor must be teamed up with other devices in order that it may serve as a useful product. Figure 12-9 shows a simplified block diagram of a microcomputer or microcontroller. The *read-only memory* (ROM) is required to store permanently the program or programs that are necessary for system

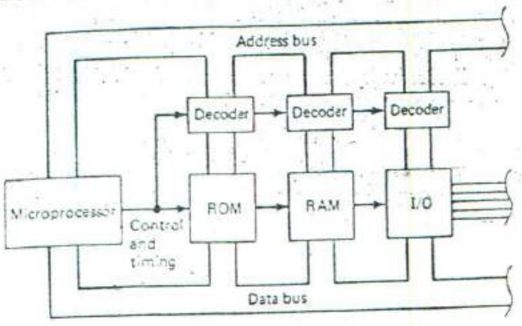


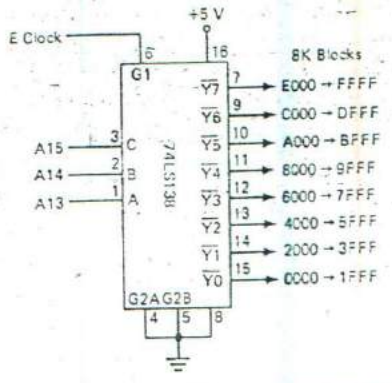
Fig. 12-9 A microcomputer or microcontroller.

operation. The *random access memory* (RAM) is used to store variables, data, and, in some cases, programs. Programs can be run from ROM or from RAM. In smaller, dedicated microcomputers or microcontrollers, the programs are usually stored in ROM. Some means for input and output are also required for the microprocessor to be functional. The I/O capability provides a way to enter programs, data, and control parameters, and it also allows for output to terminals, printers, motors, relays, and displays.

Figure 12-9 shows that decoders sit on the address bus to allow the microprocessor to access the major parts of the system selectively. It also shows that timing and control signals are required to synchronize the flow of data among the various devices. For example, the microprocessor read/NOT write line must connect to RAM and all other devices involved in bidirectional data transfer. All system components sit on the data bus, and the selective decoding and synchronizing signals allow for orderly data transfers.

A *memory map* is an important item when working with a microprocessor-based system. Figure 12-10 is an example; it shows that RAM extends from address $0000 to $1FFF for a total of 8K bytes. The I/O extends from $8000 to $9FFF, ROM ranges from $E000 to $FFFF, and each consumes another 8K bytes. This leaves 40K bytes unused out of the available 64K. Small, dedicated microcomputers and microcontrollers usually do not need 64K of address space. Empty sockets may be provided to expand ROM or RAM in some cases.

Address decoders may be used to break the memory map up into sections; one method of accomplishing this is shown in Fig. 12-11. A 74LS138 decodes the three most significant bits of the address bus: A13, A14, and A15. One of the eight outputs of this IC will go low when it is enabled; which one does, depends on the bit pattern at its inputs. The chart in Fig. 12-11 shows the most significant hex characters of the address bus from 0 through E. When any address from $0000 through and including $1FFF appears on the bus, the Y0 output of the decoder will go low if it is enabled. When any address from $2000 through and including $3FFF appears on the bus, the Y1 output of the decoder will go low if it is enabled. The decoder will enable one 8K byte section of memory at a time. This is a particularly effective



Hex	0	2	4	6	8	A	C	E
Binary	000X	001X	010X	011X	100X	101X	110X	111X
Output	Y0	Y1	Y2	Y3	Y4	Y5	Y6	Y7

X = Don't care

Fig. 12-11 Using a 74LS138 for address decoding.

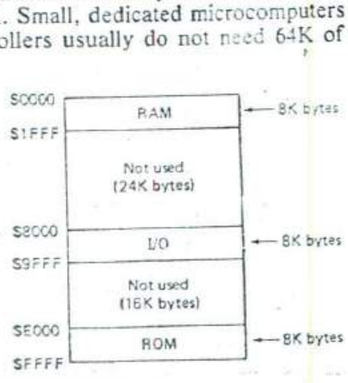


Fig. 12-10 Sample memory map.

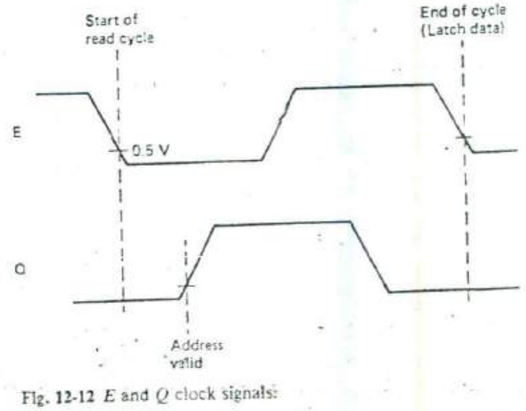


Fig. 12-12 *E* and *Q* clock signals.

Hex	8				1				3				F			
Binary	1	0	0	0	0	0	0	1	0	0	1	1	1	1	1	1
Address pin	A15	A14	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0

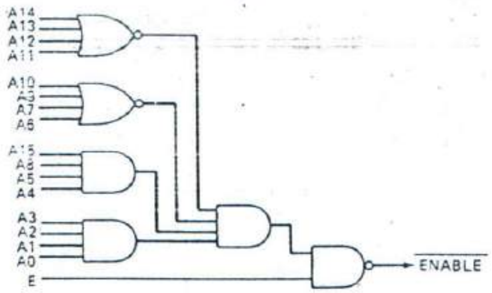


Fig. 12-13 An address decoder for $813F.

technique when used in conjunction with 8K memory devices such as the 68364 and the 2764, which are discussed in Chapter 11. For example, suppose an 8K 2764 EPROM is to be mapped into the top of memory. All that is required is to connect the Y7 output of the decoder shown in Fig. 12-11 to the output enable and chip enable pins of the EPROM. You may recall that these are active low, and the EPROM will be selected for any address from $E000 to $FFFF. The rest of the address bus (A0 through A12) will connect directly to the EPROM for internal decoding of its 8K byte space.

Figure 12-11 also shows that the E clock signal from the microprocessor is connected to the active high enable input of the 74LS138 decoder. This connection ensures that one of the 8K blocks will be selected only when the E clock is high. Figure 12-12 shows why this is necessary. A read cycle begins when the E clock drops to 0.5 V. The address bus takes time to set up and is not valid until the Q clock signal rises to 0.5 V. Then the E signal goes high. If the E clock is used to enable the decoder, no device will be selected during the time the address bus is setting up. The data are read (latched) into the microprocessor on the falling edge of E.

The timing diagram of Fig. 12-12 shows that only one-half clock cycle is available for a memory device to be read. This can present a probem in some systems. The MC68B09 microprocessor can be operated at up to 2 MHz. The period is found by taking the reciprocal of the frequency:

$$\begin{aligned} t &= \frac{1}{f} \\ &= \frac{1}{2 \times 10^6} \\ &= 500 \text{ ns} \end{aligned}$$

Since only half that time is available, memory must be read in 250 ns. One solution is to use fast memory devices, but this technique is expensive. Another solution is to form an extended clock window by ORing the E and Q clock signals. The address is valid with the rising edge of Q, and some systems use this technique to gain another quarter cycle of read time. It is also possible to use the MRDY input (pin 36) of the processor to allow slow memory devices to stretch the clock signals and provide extra time for data access.

Block decoding is not always adequate. It may be necessary to decode a single address to allow the microprocessor to access a data port, a relay, or an analog-to-digital converter. Examples of the way a single address may be decoded are given in Figs. 12-13 and 12-14. The address in Fig. 12-13 is $813F. Decoder design begins with the conversion from hex to binary. The table shows the state of the various address pins for $813F. Two four-input NOR gates are used for address lines that are low. The NOR outputs

Hex	8				X				X				F			
Binary	1	0	0	0	X	X	X	X	X	X	X	X	1	1	1	1
Address pin	A15	A14	A13	A12	X	X	X	X	X	X	X	X	A3	A2	A1	A0

X = Don't care

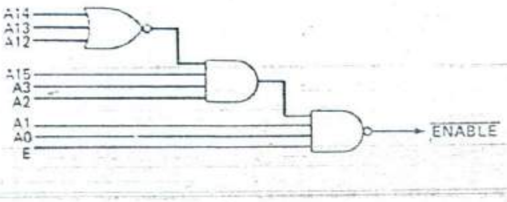


Fig. 12-14 A partial decoder for $8XXF.

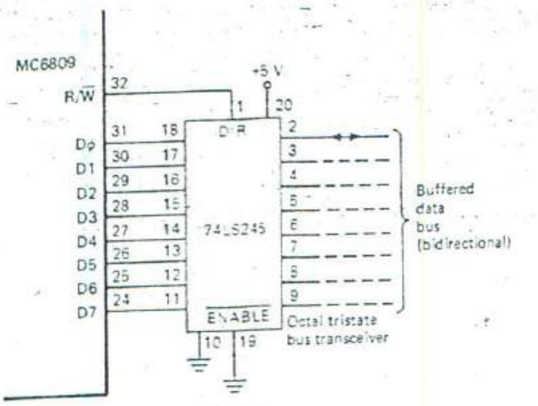

MC6809
R/W
D0
D1
D2
D3
D4
D5
D6
D7
+5 V
74LS245
DIR
ENABLE
Buffered data bus (bidirectional)
Octal tristate bus transceiver

Fig. 12-15 Buffering the data bus.

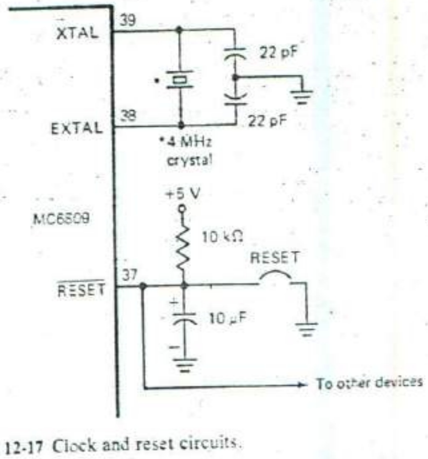


Fig. 12-17 Clock and reset circuits.

will be high only when all of the NOR inputs are low. Three four-input AND gates are used to combine the NOR outputs with all those address lines that are high for $813F. Finally, a two-input NAND gate is used to produce a low enable signal when the address pattern is correct and the E clock is high.

The address $813F is fully decoded by the circuit shown in Fig. 12-13. Of all 65,536 addresses that can occur, this is the only one that can produce an enable

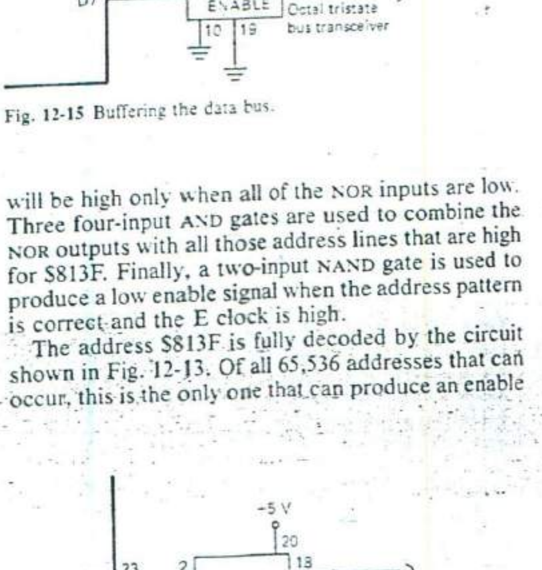


Fig. 12-16 Buffering the address bus.

signal. Full decoding is not necessary in many industrial systems. We have already learned that the memory map is seldom full. Figure 12-14 shows how partial decoding can simplify circuit design. The same address is decoded, but the circuit is designed with don't care bits on the address bus. This arrangement allows three gates to satisfy the design requirements. The same rules are used. The NOR gate handles those bits that are low, and the AND and NAND gates take care of the high bits and the E clock. The price that must be paid for partial decoding is that many addresses will produce an enable signal. The circuit of Fig. 12-14 will decode $812F, $800F, $823F, $8FFF, and many others.

The MC6809 microprocessor is rated to source −205 μA and to sink 2 mA at its data pins. This makes it necessary to buffer the data bus in some industrial applications. Figure 12-15 shows a 74LS245 octal tri-state bus transceiver IC. It is called a *bus transceiver* since it is bidirectional. The state of pin 1 determines whether certain other pins are inputs or outputs. When pin 1 is high, pins 2 through 9 are inputs, and pins 11 through 18 are outputs. This status reverses by taking pin 1 low. Note that pin 1 is controlled by the read/NOT write line of the microprocessor. The 74LS245 is housed in a 20-pin dual-inline package and can source −15 mA and sink 24 mA. It greatly improves the data bus capacity of the system.

The source current rating of the MC6809 microprocessor address pins is only −145 μA. Thus, it may also be necessary to buffer the address bus. Two 74LS244 octal tri-state bus driver ICs are pictured in Fig. 12-16. Transceivers are not required here because the microprocessor address pins are always outputs. The driver ICs greatly improve the address bus capacity. They also improve the noise margin since each input of the 74LS244 has 400 mV of hysteresis. The driver outputs are permanently enabled in Fig. 12-16 because pins 1 and 19 are

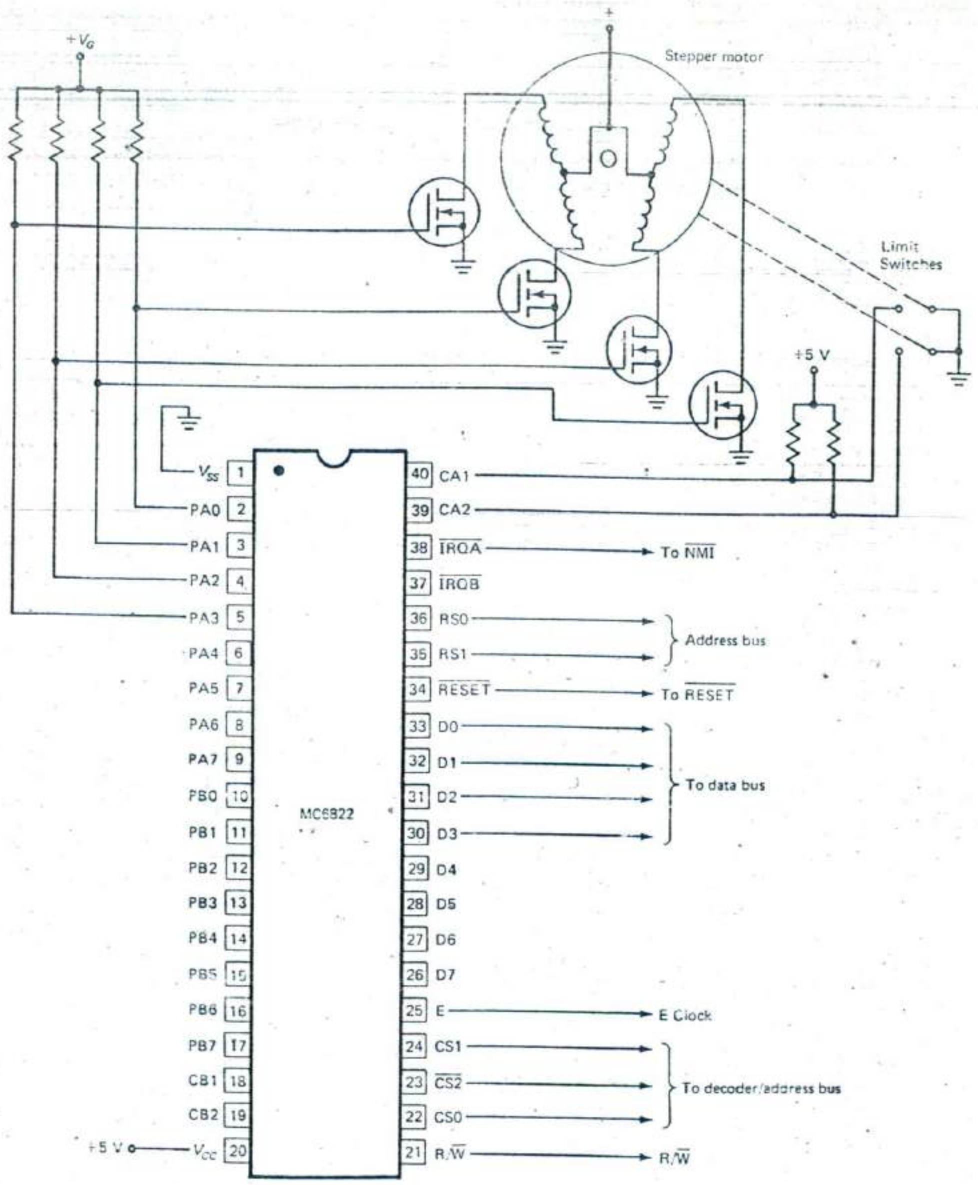
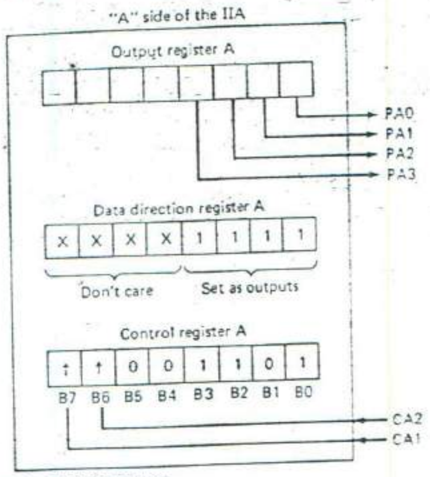


Fig. 12-18 Stepper motor interface.

grounded. However, it is possible to disable the outputs and tri-state the address bus. This would be done in those systems in which another device or circuit required direct memory access (DMA). Bus contention would be avoided by tri-stating the bus drivers during the DMA period.

Another aspect of microprocessor system design is the clock circuit. The MC6809 has an internal clock oscillator. An external crystal and two capacitors are all that is required for a complete clock circuit. The connections are indicated in Fig. 12-17. The crystal frequency must be four times the desired clock frequency. An external clock oscillator can be used by grounding pin 39 and feeding a TTL-compatible clock signal into pin 38. The crystal and capacitors will be eliminated from the wiring in this case.



CONTROL REGISTER BITS

B0 = 1:	Enables IRQA interrupt by CA1
B1 = 0:	Select neg. edge on CA1
B2 = 1:	Output register selected
B3 = 1:	Enables IRQA interrupt by CA2
B4 = 0:	Select neg. edge on CA2
B5 = 0:	Establish CA2 as an input
B6 ↑ :	Set high by neg. edge on CA2 (cleared by reading output register)
B7 ↑ :	Set high by neg. edge on CA1 (cleared by reading output register)

Fig. 12-19 The IIA registers.

Also shown by Fig. 12-17 is a typical reset circuit for the microprocessor. Pressing the reset switch will ground pin 37 and initiate the reset procedure. Pin 37 is a Schmitt-trigger input, and the reset signal must be present for more than one bus cycle. During initial power on, the reset input must be held low until the clock oscillator is fully operational. It may take as long as 20 ms for this to happen; it is provided for in Fig. 12-17 by an *RC* time delay network. The 10,000-Ω resistor and the 10-μF capacitor have a time constant of 100 ms. This ensures that reset will not be released before the clock is operational. Because of the Schmitt-trigger input, the reset state will not be released until pin 37 reaches 4.0 V. This is in contrast to other devices in the system, which will recognize 2.4 V as a logic high. If all devices are connected to the same reset bus, the microprocessor will leave the reset state last. This feature is important because it guarantees that the processor will never begin communicating with another part of the system that has not completed its own reset procedure.

Figure 12-18 shows an example of a device that must be connected to the system reset bus, the MC6822 industrial interface adapter. This 40-pin IC provides a universal means of interfacing various kinds of peripheral equipment to the microprocessor. It features two 8-bit bidirectional data ports and four control lines. The data ports are programmable; any of the 16 can be used as inputs or outputs. The functional configuration of the industrial interface adapter (IIA) is programmed by the microprocessor unit (MPU) during system initialization. System initialization begins immediately after a reset. For this reason the MPU must clear the reset state last.

The IIA features open drain outputs. The 16 port lines and 4 control lines can be pulled up externally to a maximum of 18 V. Level shifters are not required to interface directly with 15-V CMOS. Also, better noise margins are possible. Figure 12-18 shows four lines of port A (PA0 through PA3) configured as outputs. These four outputs are pulled up to $+V_G$, which is the gate supply for the four enhancement-mode VMOS transistors. The gate supply voltage must be high enough to saturate the transistors when any of the ports is at logic high. When the microprocessor writes a 0 to any of the ports, the gate voltage will drop to near 0 V, and the transistors will turn off. This allows the microprocessor to control the motor since the transistors provide the ground return for the motor coils.

The industrial interface adapter has two sides, A and B. Each side has three registers; the registers for the A side are in Fig. 12-19. The output register latches data written by the microprocessor and makes it available to the outputs PA0 through PA3. The stepper motor bit patterns will be written to output register A. The data direction register determines whether the ports will be inputs or outputs. Figure 12-19 shows that four logic 1s are stored in the lower half of the register. This programs PA0 through PA3 to serve as outputs. The upper 4 bits are "don't care" since these ports are not used. The contents of the control register enable the interrupts, select the negative edge for the interrupt inputs, select the output register, establish CA2 as an input, and reflect which input, if any, caused an interrupt. Initialization of the IIA after a system reset would involve writing binary XXXX1111 to the data direction register and then writing binary XX001101 to the control register. Subsequent writes would then go to the output register to step the motor. If an interrupt occurred, the interrupt service routine would read the contents of the control register and examine bits 7 and 6 to determine which limit switch tripped.

REVIEW QUESTIONS

31. Refer to Fig. 12-11. Suppose G2A of the decoder is not grounded and A15 of the address bus is connected to it. Also, assume that the A, B, and C inputs of the decoder are connected to A12, A13, and A14 of the bus, respectively. What is the lowest hex address that will enable decoder output Y0? The highest? What size blocks does this provide?

32. Refer to Fig. 12-12. When is the address bus invalid?

33. Refer to Fig. 12-12. Ignore propagation delay in the decoder. With a 1-MHz clock, how much time is available for memory access if the E clock is used for the enable signal? How much time is available if the enable signal is E OR Q?

34. Refer to Fig. 12-13. Suppose address lines A0 and A14 are interchanged. What address will be decoded? Is this the only address that will decode?

35. Refer to Fig. 12-14. What is the lowest hex address to which the decoder will respond? The highest? How many addresses will be decoded?

36. Refer to Fig. 12-18. Ignoring interrupt capabilities, how many stepper motors of the type shown could be controlled by one IIA?

37. Refer to Fig. 12-18. How would the micro-processor know which limit switch caused an interrupt?

38. Refer to Fig. 12-18. Ignoring interrupt capabilities, what simple device might be used to replace the IIA?

12-6 SUPPORT DEVICES

The industrial interface adapter presented in the last section is an example of a support device that makes interfacing a microprocessor to various other circuits and systems easier. A number of large-scale integrated circuits that are microprocessor-compatible have been developed. They connect directly to the data bus and control and timing lines and to some portion of the address bus. These support devices make it easier to apply microprocessors to industrial applications.

The *peripheral interface adapter* (PIA) is a popular support device. It is available in several styles from various IC manufacturers. The IIA already discussed is a variation of the PIA. Motorola's part number for their PIA is MC6821, and it has the same pin configuration as their IIA. The only major difference between the two is that the IIA has open drain outputs. Another PIA variation is the *versatile interface adapter* (VIA). Figure 12-20 shows the pinout for the R6522 VIA manufactured by Rockwell. This pinout is somewhat different from the PIA and IIA configurations. The VIA has only one interrupt output to the processor and only two chip select inputs. PIAs and IIAs have two interrupt outputs and three chip select inputs, making two more pins available for register selection. Note that the VIA has four: RS0 through RS3. Another minor difference is pin 25, which is labeled *phase 2* on the VIA. This input is the same as the E clock input on Motorola devices.

The VIA features two 8-bit bidirectional I/O ports, and each line can be programmed as an input or as an output. This is the same arrangement as in the PIA. However, PIAs can latch only output data, and the VIA is also capable of latching input data. This is an important feature when a microprocessor must be interfaced to a device which makes data available for only brief periods of time. The VIA can latch such data and hold them until the microprocessor is ready. The VIA also contains two 16-bit programmable counter/timers. Several I/O lines can be controlled directly from the interval timers to generate programmable-frequency square waves. External pulses can also be counted, and an interrupt can be generated to the processor when a predetermined count is reached. Finally, the VIA contains a shift register, which can be used to provide serial data communication.

Figure 12-21 shows the VIA block diagram. Starting at the right, note that input latches (IRA and IRB) are available for port A and for port B. We also find the familiar output registers and data direction registers for each port. A shift register is available and connects to CB1 and CB2. These pins can be used for serial data communication. The *handshake control section* allows data to be transferred in and out only when devices are ready. An analogy for handshaking would be asking people to slow down when they are giving you directions too quickly for you to assimilate them. Moving to the left, we find the two timers. The timers consist of counters and latches. The counters are divided into low bytes and high bytes. Timer 1 has both a high-byte latch and a low-byte latch; timer 2 has only a low-byte latch. Figure 12-21 also shows the function control and interrupt control sections of the VIA, which facilitate

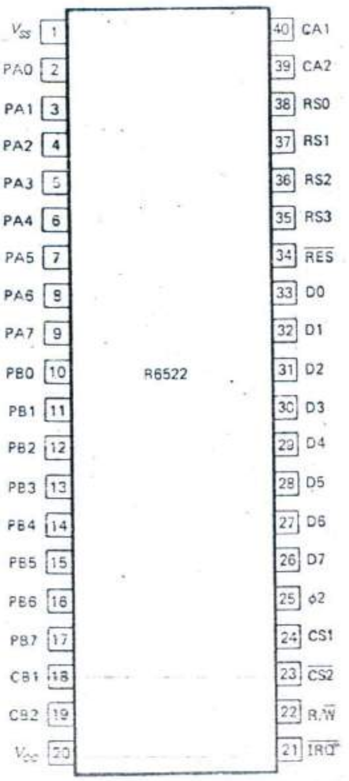


Fig. 12-20 VIA pinout.

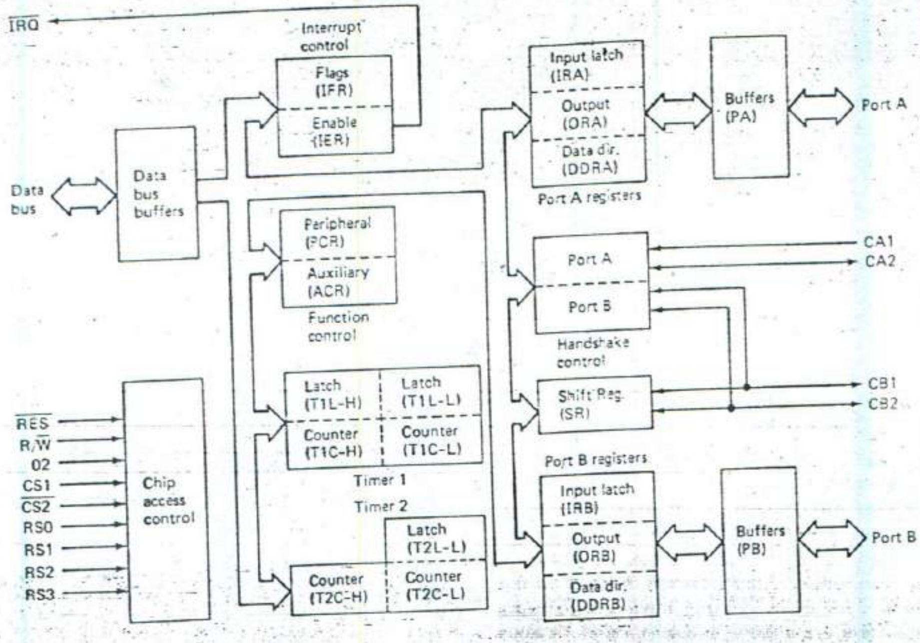
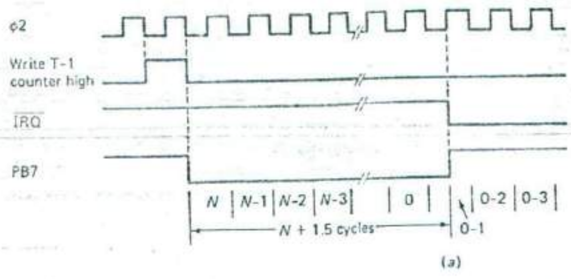


Fig. 12-21 VIA block diagram.

programming the many powerful features of this device.

The four register select lines of the VIA provide access to 16 internal registers. The register addressing is shown in Fig. 12-22. Normally RS0 through RS3 will connect to A0 through A3 of the address bus, respectively. The high-order bits of the address bus will normally connect to a separate decoder and to the two chip-select inputs of the VIA. The VIA can be mapped almost anywhere into memory. Timer 1 can be loaded by writing data to registers 6 and 7. After loading, the counter decrements at the phase

Register number	RS Coding				Register designation	Description	
	RS3	RS2	RS1	RS0		Write	Read
0	0	0	0	0	ORB/IRB	Output register B	Input register B
1	0	0	0	1	ORA/IRA	Output Register A	Input Register A
2	0	0	1	0	DDRB	Data direction register B	
3	0	0	1	1	DDRA	Data direction register A	
4	0	1	0	0	T1C-L	T1 Low-order latches	T1 Low-order latches
5	0	1	0	1	T1C-H	T1 High-order counter	
6	0	1	1	0	T1L-L	T1 Low-order latches	
7	0	1	1	1	T1L-H	T1 High-order latches	
8	1	0	0	0	T2C-L	T2 Low-order latches	T2 Low-order latches
9	1	0	0	1	T2C-H	T2 High-order counter	
10	1	0	1	0	SR	Shift Register	
11	1	0	1	1	ACR	Auxiliary control register	
12	1	1	0	0	PCR	Peripheral control register	
13	1	1	0	1	IFR	Interrupt flag register	
14	1	1	1	0	IER	Interrupt enable register	
15	1	1	1	1	ORA/IRA	Same as register 1 except no "handshake"	



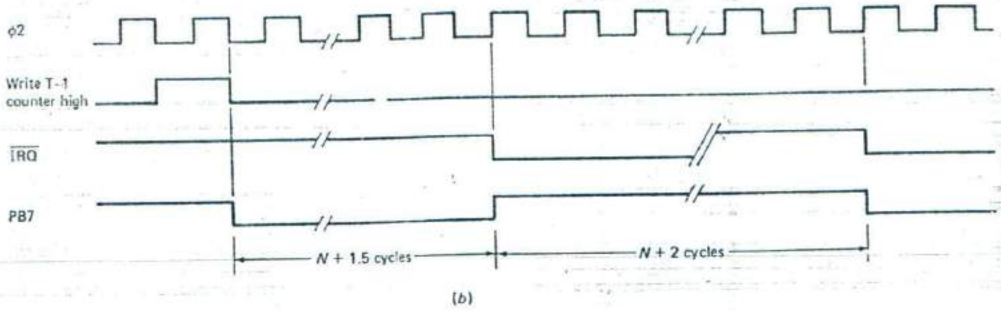


Fig. 12-23 Timer 1 operating modes. (*a*) One-shot operation. (*b*) Free-run operation.

2 clock rate. Upon reaching zero, an interrupt flag is set, and IRQ goes low if the T1 interrupt is enabled. Timer 1 then disables any further interrupts or automatically transfers the contents of the latches to the counter and begins decrementing from the loaded value again. The timer may also be programmed to invert the output signal on PB7 on every occasion that it times out.

Timer 1 can be used in a one-shot mode or in a free-running mode. The *one-shot mode* generates a single interrupt for every timer load operation. The delay between the write to the high byte of the counter and the generation of the processor interrupt is a function of the 16-bit data word loaded into the counter. Timer 1 can be programmed to produce a single negative pulse on the PB7 peripheral pin in addition to generating a single interrupt. If bit 7 in the auxiliary control register is set, a write to the high byte of counter 1 will cause PB7 to go low (see Fig. 12-23[*a*]). When timer 1 times out, PB7 returns high. This provides a single programmable-width output pulse for controlling some device or circuit interfaced to the microcomputer or microcontroller.

Time-out in the one-shot mode sets the timer 1 interrupt flag, and the IRQ pin goes low. The timer then continues to decrement from zero at the system clock rate, allowing the processor to read the counter and determine the time since the interrupt. When the processor again writes into the high-order counter, the T1 interrupt flag will be cleared, the contents of the low-order latch will be transferred into the low-order counter, and the timer will once again begin to decrement from the value loaded.

Suppose some industrial system requires single pulses of 35,768 μs in length. First, bits 6 and 7 of the auxiliary control register would have to be set to 0 and 1 respectively. This procedure will select the one-shot mode and enable PB7. Also, bit 7 of the data direction register for port B must be set to enable PB7 as an output. Assuming a 1-MHz clock, each clock cycle will be 1 μs in length. Therefore timer 1 must be loaded with the binary equivalent of 35,768 minus 1 or 2. This is because PB7 will go low for $N + 1.5$ cycles as shown in Fig. 12-23(*a*). The pulses produced will be 0.5 μs more or less than the required value. This error is of no consequence in most industrial applications. The hex equivalent of decimal 35,767 is $8BB7. Next, the processor will write $B7 to register 4. This operation stores the lower byte in timer 1's low-order latch. Finally, the processor will write $8B to register 5, which starts the timing pulse. Next PB7 will go low and then go high 35,768.5 μs later. Subsequently, one of these pulses can be produced every time the processor writes $8B to register 5. It is not necessary to rewrite the low byte because it has been latched.

The *free-run mode* takes more full advantage of the latches associated with timer 1. Every time the counter reaches zero, the contents of the latches are transferred into the counter, and the counter starts decrementing from *N* again. This is in contrast to the one-shot mode, in which the count decrements from 0 after a time-out. Figure 12-23(*b*) shows the free-run mode of operation for timer 1. Suppose register 6 (the low-order latch) contains $3C, and register 7 (the high-order latch) contains $1A. The decimal

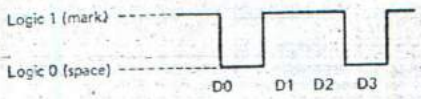


Fig. 12-24 Serial (mark-space) data format.

equivalent of $1A3C is 6716. The waveforms show one complete cycle of the output waveform to be $N + 1.5$ clock cycles $+ N + 2$ clock cycles. With a 1-MHz clock, the total time for one output cycle will be $6716 + 1.5 + 6716 + 2 = 13{,}435.5\ \mu s$. The reciprocal of this yields 74.43 Hz. Thus, a nearly square 74.43 Hz waveform will be available if PB7 is enabled.

The processor can access the latches during the down-counting operation by writing to registers 6 and 7. This process will not affect the time-out in process. Instead, the data written to the latches will determine the next time-out period. Since the interrupt flag is set with each time-out, the processor can respond with new data for the latches and set the period for the next half cycle. This characteristic enables very complex waveforms to be generated at the PB7 output.

Timer 2 in the VIA also has a one-shot mode similar to that discussed for timer 1. It will provide a single interrupt for each write to the high byte of counter 2. After it times out, the count will continue to decrement. The interrupt flag is disabled and will not be set again even if the counter decrements to zero again. The processor must write to the high byte to enable the setting of the interrupt flag. The flag is cleared by reading the low byte of the counter or writing the high byte.

Timer 2 also has a mode for counting the number of negative pulses applied to PB6. This is accomplished by loading a number into timer 2. Writing the high byte clears the interrupt flag and allows the counter to decrement on each negative pulse. When the counter decrements past zero, the interrupt flag is set. The counter then continues decrementing on every negative pulse. It is necessary to rewrite the high byte to allow the interrupt flag to be set on a subsequent time-out.

Many details of the VIA are not covered here. The reader is referred to the manufacturers' data manuals for additional information. Anyone who works with microprocessor-based systems must have access to these data manuals for hardware and programming information.

The support devices covered to this point are mainly concerned with parallel data transfers. *Serial data transfers* are also very important in industrial systems. Many support devices have been developed to facilitate serial I/O. The earliest among these was the *universal asynchronous receiver/transmitter* (UART). *Universal synchronous/asynchronous receiver/transmitters* (USART) are also available. We will look at a more modern serial support device, the *asynchronous communications interface adapter* (ACIA). First, however, some basics concerning serial I/O will be presented.

Parallel I/O is fast but requires a cable with many circuits. It is usually limited to distances of approximately 8 m (26.25 ft). Serial data communications circuits provide simplified wiring and the capacity for modulation. *Modulation* is a process of using the data to control the amplitude or the frequency of a high-frequency signal called a *carrier*. The carrier signal can be in the audio spectrum, allowing serial data transfers over ordinary telephone circuits. A *modulator* will be required to change the data into audio signals, and a *demodulator* will be used to change the audio signals back into data. Both circuits are usually contained in one unit called a modulator-demodulator, or *modem*.

Figure 12-24 shows the mark-space format used to transfer serial data. Logic 1 is called *mark* and represents some current level or some voltage level, depending on the standard used. Logic 0 is called *space* and represents 0 current or some voltage level, again depending upon which standard is in use. The serial line is held at mark when the equipment is turned on but no data are being transmitted. When

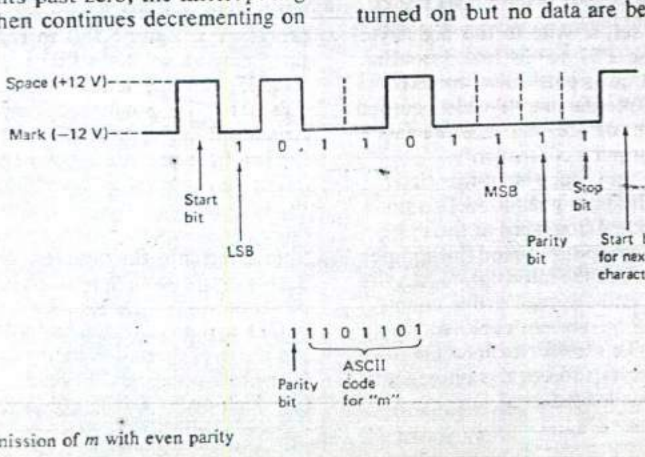


Fig. 12-25 RS-232C transmission of *m* with even parity and 1 stop bit.

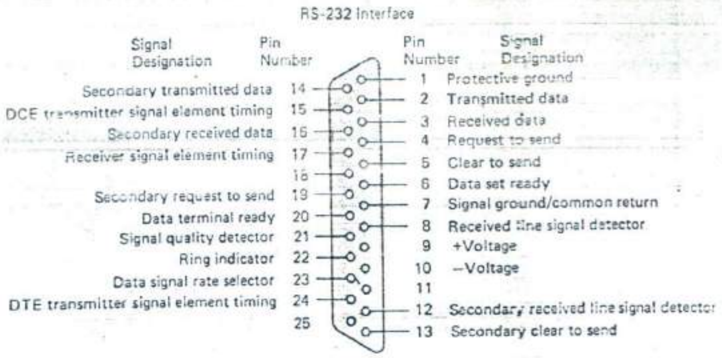

RS-232 Interface
Signal Designation
Pin Number
Pin Number
Signal Designation
Secondary transmitted data 14
DCE transmitter signal element timing 15
Secondary received data 16
Receiver signal element timing 17
18
Secondary request to send 19
Data terminal ready 20
Signal quality detector 21
Ring indicator 22
Data signal rate selector 23
DTE transmitter signal element timing 24
25
1 Protective ground
2 Transmitted data
3 Received data
4 Request to send
5 Clear to send
6 Data set ready
7 Signal ground/common return
8 Received line signal detector
9 +Voltage
10 −Voltage
11
12 Secondary received line signal detector
13 Secondary clear to send

Fig. 12-26 RS-232C connector wiring.

a serial circuit uses current rather than voltage, the mark current will be either 60 or 20 mA. Both of these are older standards and are less popular than they once were. Several voltage standards exist, and the EIA RS-232C system is the most popular. It allows any value from −3 to −25 V to represent mark and from +3 to +25 V to represent space. Figure 12-25 shows how the transmission of the ASCII code for *m* would look in a typical RS-232C system. The waveform is shown as it would appear on an oscilloscope (mark is negative and is at the bottom of the waveform). Note that the least significant bit is sent first. This is an example of asynchronous transmission, which is the most popular. Start and stop bits are required to frame the data word. An alternative is to use *synchronous data transmission*, which requires a common clock at both ends of the communication circuit. Special characters are sent during idle periods to keep the clocks synchronized. A synchronous data transmission begins with a *preamble* consisting of a fixed number of mark bits that allow the receiver to lock onto the characters. Since no framing bits are needed, synchronous transmission is approximately 20 percent faster.

Serial transmission speed is rated in baud (Bd); 1 Bd is equal to 1 bit/s. Common speeds are 300, 600, 1200, 2400, and 4800 Bd, and so on. If the rate is 4800 Bd, how many characters are sent per second? In asynchronous circuits, it depends on the format used. As an example, Fig. 12-25 shows that 10 bits are required to send one ASCII character. Therefore, the data rate will be 480 characters per second at 4800 Bd. If 2 stop bits are used, the data rate will be lower. If no parity bit is sent, the data rate will be higher.

Figure 12-26 shows the standard RS-232C connector wiring. It is common to find a female DB-25 connector on a data terminal and a male DB-25 connector on the data cable. Most installations use far fewer than 25 wires. In fact, it is possible to connect some data equipment with as few as three wires. The most important connections on the DB-25 connector are pin 2 (transmit data), pin 3 (receive data), and pin 7 (signal ground). Although it is common practice to do so, it is not a good idea to eliminate the protective ground.

Other important serial communications standards exist in addition to RS-232C. One example of an increasingly popular standard is RS-422. This standard uses a balanced transmission line. Two data wires are required for each circuit. The bandwidth is very high, and data transfers up to several million bytes per second are possible. It is a good choice when very large amounts of data must be transferred in a short period of time.

Most microprocessors are parallel devices. The *asynchronous communications interface adapter* (ACIA) is an important support device that makes interfacing to serial devices such as data terminals, printers, modems, industrial controls, and sensors easy. Figure 12-27 represents the pin diagram for the

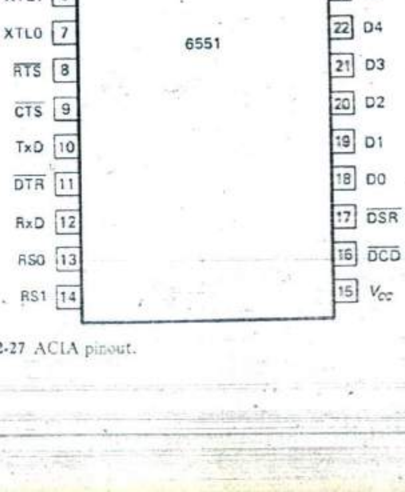


Fig. 12-27 ACIA pinout.

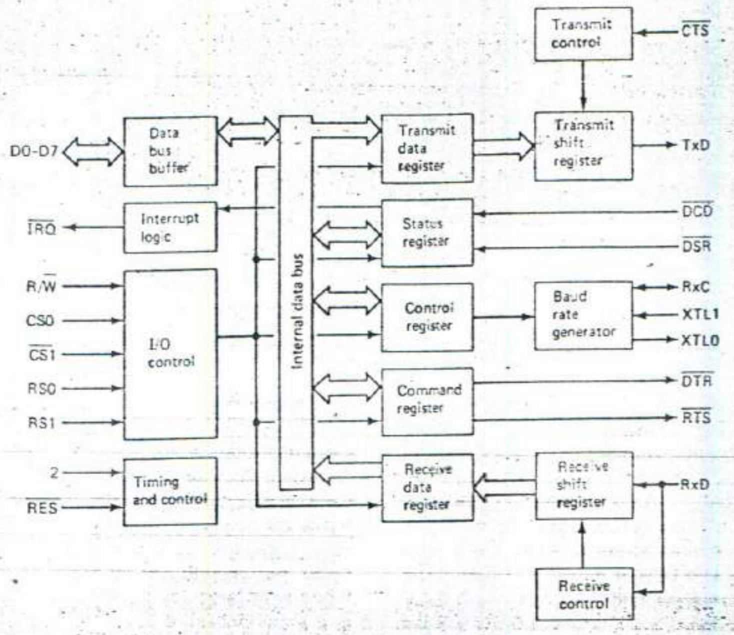
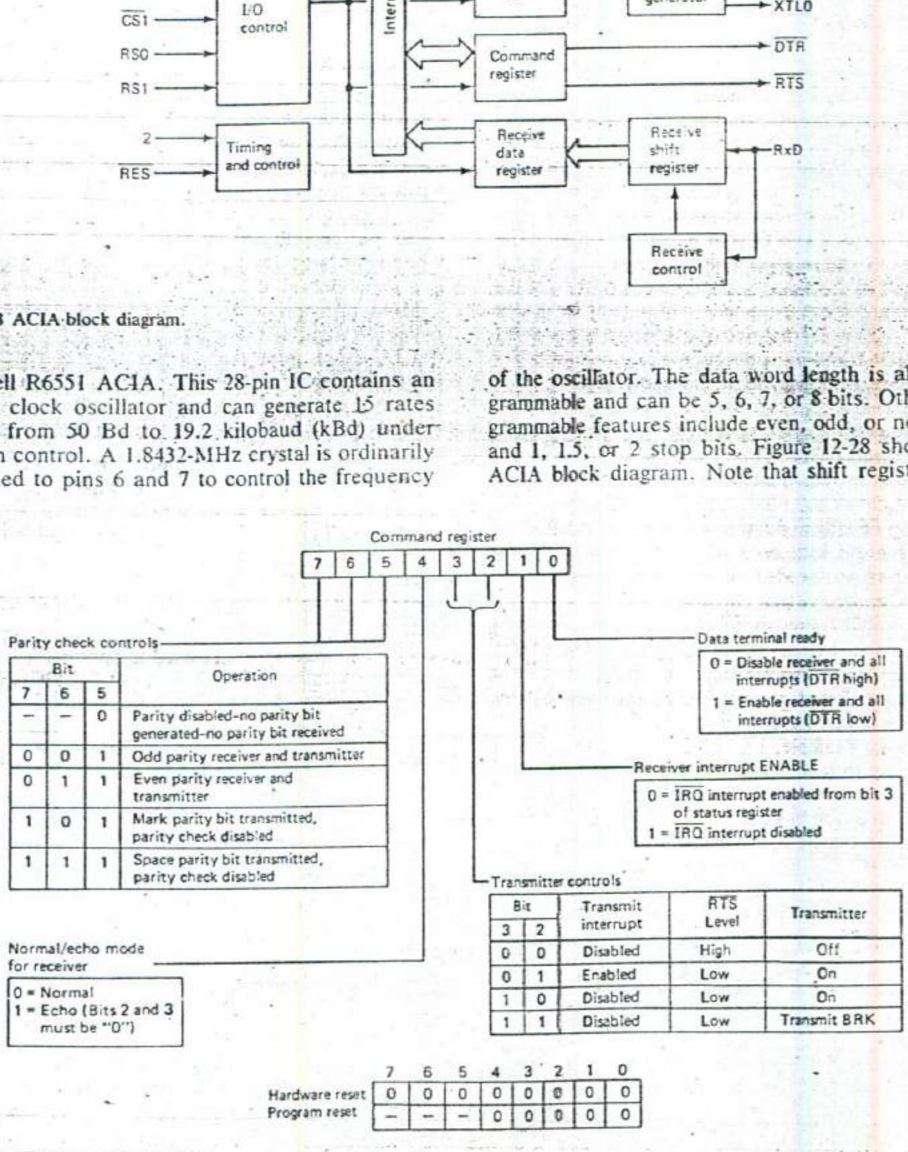


Fig. 12-28 ACIA block diagram.

Rockwell R6551 ACIA. This 28-pin IC contains an internal clock oscillator and can generate 15 rates ranging from 50 Bd to 19.2 kilobaud (kBd) under program control. A 1.8432-MHz crystal is ordinarily connected to pins 6 and 7 to control the frequency of the oscillator. The data word length is also programmable and can be 5, 6, 7, or 8 bits. Other programmable features include even, odd, or no parity and 1, 1.5, or 2 stop bits. Figure 12-28 shows the ACIA block diagram. Note that shift registers are



Bit 7	Bit 6	Bit 5	Operation
—	—	0	Parity disabled-no parity bit generated-no parity bit received
0	0	1	Odd parity receiver and transmitter
0	1	1	Even parity receiver and transmitter
1	0	1	Mark parity bit transmitted, parity check disabled
1	1	1	Space parity bit transmitted, parity check disabled

Data terminal ready
0 = Disable receiver and all interrupts ($\overline{DTR}$ high)
1 = Enable receiver and all interrupts ($\overline{DTR}$ low)

Receiver interrupt ENABLE
0 = $\overline{IRQ}$ interrupt enabled from bit 3 of status register
1 = $\overline{IRQ}$ interrupt disabled

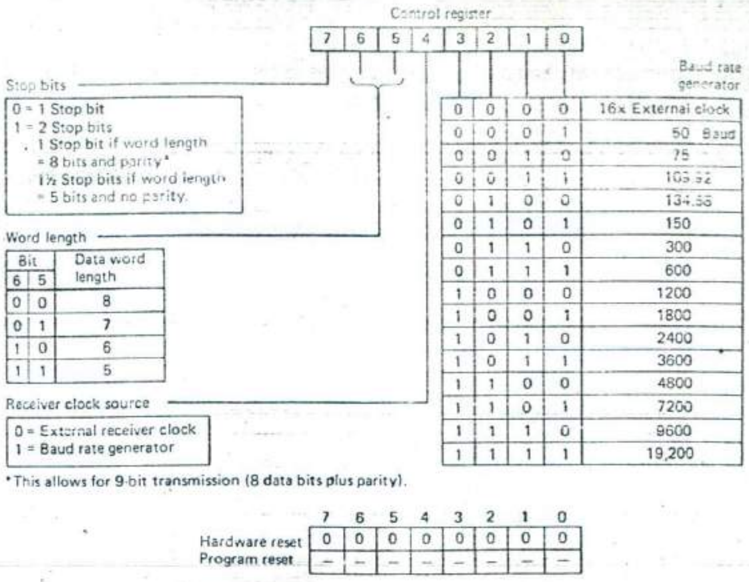
Transmitter controls

Bit 3	Bit 2	Transmit interrupt	$\overline{RTS}$ Level	Transmitter
0	0	Disabled	High	Off
0	1	Enabled	Low	On
1	0	Disabled	Low	On
1	1	Disabled	Low	Transmit BRK

Normal/echo mode for receiver
0 = Normal
1 = Echo (Bits 2 and 3 must be "0")

	7	6	5	4	3	2	1	0
Hardware reset	0	0	0	0	0	0	0	0
Program reset	—	—	—	0	0	0	0	0

Fig. 12-29 ACIA command register.



Stop bits
0 = 1 Stop bit
1 = 2 Stop bits, 1 Stop bit if word length = 8 bits and parity*, 1½ Stop bits if word length = 5 bits and no parity.

Bit 6	Bit 5	Data word length
0	0	8
0	1	7
1	0	6
1	1	5

Receiver clock source
0 = External receiver clock
1 = Baud rate generator

3	2	1	0	Baud rate generator
0	0	0	0	16x External clock
0	0	0	1	50 Baud
0	0	1	0	75
0	0	1	1	109.92
0	1	0	0	134.58
0	1	0	1	150
0	1	1	0	300
0	1	1	1	600
1	0	0	0	1200
1	0	0	1	1800
1	0	1	0	2400
1	0	1	1	3600
1	1	0	0	4800
1	1	0	1	7200
1	1	1	0	9600
1	1	1	1	19,200

*This allows for 9-bit transmission (8 data bits plus parity).

Fig. 12-30 ACIA control register.

used to convert parallel data to serial for transmission and to convert received serial data to parallel for use by the microprocessor.

The programmability of the ACIA lies in the structure of the command and control registers shown in Fig. 12-28. Figure 12-29 is a detailed look at the ACIA command register. Bit 0 sets pin 11 (data terminal ready) of the ACIA high or low, allowing the processor to inform other equipment (such as a modem) when the system is ready. Bit 1 enables or disables the receiver interrupt for incoming data. Bits 2 and 3 control the state of pin 8 (ready to send) and the transmit interrupt. Bit 4 controls the receiver echo mode. Bit 5 enables or disables parity generation and parity checking. Bits 6 and 7 select one of four parity modes: even, odd, mark, or space. *Mark parity* means that a parity bit is sent, but it is always mark (high). *Space parity* means the parity bit sent is always space (low). No parity checking is done on incoming data when mark or space parity is selected.

The control register is shown in Fig. 12-30. Bits 0 through 3 select the baud rate. Bit 4 selects the receiver clock source. If it is 0, the receiver will be clocked at a rate of one-sixteenth of an external clock applied to pin 5 of the ACIA. If it is 1, the receiver will operate at the same rate as the transmitter. Bits 5 and 6 select the word length (5, 6, 7, or 8 bits). Bit 7 selects the number of stop bits.

The ACIA also contains a status register that can be read by the processor. This register contains bits that signify whether an interrupt has occurred, the status of the signals applied to pins 16 and 17, whether or not the transmitter data register is empty, whether or not the receiver data register is full, whether there has been a parity error, or whether there has been a *framing error* (incorrect number of stop bits received).

As with the VIA, some details of the ACIA are not covered here. Once again the reader is urged to consult data manuals for additional information. Quite a few other microprocessor support devices are found in the industrial environment, including cathode-ray tube controllers (CRTC), floppy disk controllers, printer controllers, and communications controllers for other data transmission standards.

As mentioned before, the most popular standard for serial data communications is RS-232C. Therefore, some standard parts have been developed to translate between TTL voltages and RS-232C voltages. Motorola, for example, manufactures the MC1488 quad driver and the MC1489 quad receiver. The driver ICs are used to convert TTL to RS-232C, and the receivers convert RS-232C to TTL. These ICs are in 14-pin dual-inline packages.

REVIEW QUESTIONS

39. Assume that a VIA is programmed for the one-shot mode and that PB7 is enabled as a timer-controlled output. When will PB7 first go low? When will it return high?

40. Refer to Fig. 12-23(*a*). Assume a 1-MHz clock and the VIA one-shot timer mode. What hex value should be stored in the low byte of timer 1 for an output pulse of 500.5 μs? What hex high byte must the processor write to initiate the pulse?

41. Refer to Fig. 12-23(*b*). Assume a 1-MHz clock and the VIA free-running timer mode. Also assume that the high-byte latch of timer 1 contains $00 and the low-byte latch contains $3E. What is the output frequency at PB7 assuming it is enabled? The waveform?

42. Which timer of the VIA is used for counting externally generated pulses? To which VIA pin must the pulses be applied?

43. Refer to Fig. 12-25. How many characters will be sent per second at 19.2 kBd?

12-7 SOFTWARE DEVELOPMENT

It should now be clear that microprocessors and their support devices are useless without software. What is the role of the industrial electronics technician in software development? It varies from company to company; large companies tend to hire software specialists for writing, applying, and maintaining software. Smaller companies may expect the technician to do some programming. In any case, the technician needs at least a general knowledge of microprocessor software and how it is developed.

Some industrial software is written in a high-level language such as BASIC. BASIC is one of the easiest computer languages to learn and apply. It will solve many industrial control, data collection, and analysis needs with ease. However, BASIC is not available on every microcomputer. Even if it is available, some control applications cannot be implemented in BASIC because it is too slow and requires more memory than equivalent machine language programs. There are other high-level languages gaining popularity in the industrial environment. FORTH is one example. FORTH is not so easy to learn and apply as BASIC, but it offers the advantages of consuming less memory and running much faster.

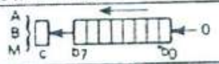
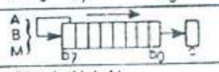
Machine language programs must be written in the processor's native code. When written by skilled programmers, they consume less memory than any equivalent high-level program and execute much faster. Sometimes, a compromise approach is used. An industrial control program for a robot may be written in BASIC with several calls to machine language subroutines. These subroutines will execute very quickly and provide the response speed required for time-sensitive robot actions and reactions.

How are machine language programs and subroutines written? It depends on the environment. Small programs and subroutines may be hand-coded, with the programmer looking up the OP-CODEs for each instruction. Hand coding places quite a bit of burden on the programmer, who must be familiar with the instruction set of the microprocessor and especially proficient with its addressing modes. Also, the programmer must calculate all relative addresses. Hand coding is considered adequate for small programs and subroutines. For large programs, it becomes tedious and is susceptible to errors.

Hand coding is made easier with programming aids such as the instruction set summary shown in Fig. 12-31. This summary is continued in Fig. 12-32, and the branch instructions are in a separate summary shown in Fig. 12-33. Look at the add with carry accumulator A instruction (ADCA) near the top of Fig. 12-31. This instruction may have any one of four different OP-CODEs, depending on the addressing mode to be used: $89 (immediate), $99 (direct), $A9 (indexed), and $B9 (extended). The ~ column shows the number of machine cycles required to fetch and execute each instruction. This information is vital for writing timing loops and for determining how fast software will execute. The # column tells how many memory bytes are needed for each instruction. This is useful for predicting how much memory a program will require and helps eliminate hand-coding errors. For example, if you wish to use ADCA with extended addressing, the 3 in the # column tells you that a total of 3 bytes will be required for the instruction. Two additional bytes must follow the OP-CODE $B9 in memory. These two bytes are the high-order address byte of the operand followed by the low-order address byte. The number of additional bytes and cycles required for the indexed addressing mode is indicated in Fig. 12-34 (page 348).

The instruction set summary also includes a description of what each instruction does and how the flags are affected. A dot under a flag means it is not changed, a 1 means the flag is set, a 0 indicates the flag is cleared, and a ↕ means the flag is toggled high or low depending on the results of the operation. Figure 12-32 lists the notes for those special flag results which require additional explanation. All of the information shown in Figs. 12-31 through 12-34 plus quite a bit more is contained on one fan-fold card. Motorola calls this a *reference card*, and the part number is M6809(AC3) for their MC6809 microprocessor. Motorola offers quite a few different microprocessors, and reference cards are available for most of them. Other manufacturers also have similar cards for their microprocessors; some call them *programmer's cards* or *programmer's aids*. You are urged to collect all relevant cards for your work. They provide a tremendous amount of information in a very compact form and are quite useful.

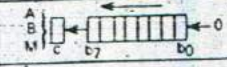
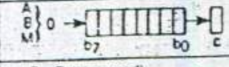
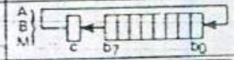
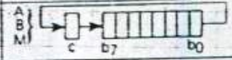
Reference cards go a long way toward making hand coding easier and error-free. However, if significant machine language development is required, then better tools are needed. One such tool is a program called an *editor*; it allows the programmer to write and correct a program written in assembly form easily. When this task is completed, a second program, the *assembler*, takes the assembly listing that was generated by the editor and converts it to machine language. Refer to Fig. 12-35 (page 348). *PAUSE* was typed into the editor by the programmer. So was everything to the right of it and below it up to the word *END*. This is usually called *source code*. Source code is typically divided into four columns: the *label* (for example, START), the *instruction mne-*

		Addressing Modes																5	3	2	1	0
		Immediate			Direct			Indexed			Extended			Inherent								
Instruction	Forms	Op	~	#	Op	~	#	Op	~	#	Op	~	#	Op	~	#	Description	H	N	Z	V	C
ABX														3A	3	1	B + X → X (Unsigned)	•	•	•	•	•
ADC	ADCA	89	2	2	99	4	2	A9	4+	2+	B9	5	3				A + M + C → A	↕	↕	↕	↕	↕
	ADCB	C9	2	2	D9	4	2	E9	4+	2+	F9	5	3				B + M + C → B	↕	↕	↕	↕	↕
ADD	ADDA	8B	2	2	9B	4	2	AB	4+	2+	BB	5	3				A + M → A	↕	↕	↕	↕	↕
	ADDB	CB	2	2	DB	4	2	EB	4+	2+	FB	5	3				B + M → B	↕	↕	↕	↕	↕
	ADDD	C3	4	3	D3	6	2	E3	6+	2+	F3	7	3				D + M:M + 1 → D	•	↕	↕	↕	↕
AND	ANDA	84	2	2	94	4	2	A4	4+	2+	B4	5	3				A ∧ M → A	•	↕	↕	0	•
	ANDB	C4	2	2	D4	4	2	E4	4+	2+	F4	5	3				B ∧ M → B	•	↕	↕	0	•
	ANDCC	1C	3	2													CC ∧ IMM → CC					7
ASL	ASLA													48	2	1		8	↕	↕	↕	↕
	ASLB													58	2	1		8	↕	↕	↕	↕
	ASL				08	6	2	68	6+	2+	78	7	3					8	↕	↕	↕	↕
ASR	ASRB													47	2	1		8	↕	↕	•	↕
	ASR													57	2	1		8	↕	↕	•	↕
	ASR				07	6	2	67	6+	2+	77	7	3					8	↕	↕	•	↕
BIT	BITA	85	2	2	95	4	2	A5	4+	2+	B5	5	3				Bit Test A (M ∧ A)	•	↕	↕	0	•
	BITB	C5	2	2	D5	4	2	E5	4+	2+	F5	5	3				Bit Test B (M ∧ B)	•	↕	↕	0	•
CLR	CLRA													4F	2	1	0 → A	•	0	1	0	0
	CLRB													5F	2	1	0 → B	•	0	1	0	0
	CLR				0F	6	2	6F	6+	2+	7F	7	3				0 → M	•	0	1	0	0
CMP	CMPA	81	2	2	91	4	2	A1	4+	2+	B1	5	3				Compare M from A	8	↕	↕	↕	↕
	CMPB	C1	2	2	D1	4	2	E1	4+	2+	F1	5	3				Compare M from B	8	↕	↕	↕	↕
	CMPD	10	5	4	10	7	3	10	7+	3+	10	8	4				Compare M:M + 1 from D	•	↕	↕	↕	↕
		83			93			A3			B3											
	CMPS	11	5	4	11	7	3	11	7+	3+	11	8	4				Compare M:M + 1 from S	•	↕	↕	↕	↕
		8C			9C			AC			BC											
	CMPU	11	5	4	11	7	3	11	7+	3+	11	8	4				Compare M:M + 1 from U	•	↕	↕	↕	↕
		83			93			A3			B3											
	CMPX	8C	4	3	9C	6	2	AC	6+	2+	BC	7	3				Compare M:M + 1 from X	•	↕	↕	↕	↕
	CMPY	10	5	4	10	7	3	10	7+	3+	10	8	4				Compare M:M + 1 from Y	•	↕	↕	↕	↕
		8C			9C			AC			BC											
COM	COMA													43	2	1	$\overline{A}$ → A	•	↕	↕	0	1
	COMB													53	2	1	$\overline{B}$ → B	•	↕	↕	0	1
	COM				03	6	2	63	6+	2+	73	7	3				$\overline{M}$ → M	•	↕	↕	0	1
CWAI		3C	≥20	2													CC ∧ IMM → CC Wait for Interrupt					7
DAA														19	2	1	Decimal Adjust A	•	↕	↕	0	↕
DEC	DECA													4A	2	1	A – 1 → A	•	↕	↕	↕	•
	DECB													5A	2	1	B – 1 → B	•	↕	↕	↕	•
	DEC				0A	6	2	6A	6+	2+	7A	7	3				M – 1 → M	•	↕	↕	↕	•
EOR	EORA	88	2	2	98	4	2	A8	4+	2+	B8	5	3				A ⊻ M → A	•	↕	↕	0	•
	EORB	C8	2	2	D8	4	2	E8	4+	2+	F8	5	3				B ⊻ M → B	•	↕	↕	0	•
EXG	R1, R2	1E	8	2													R1 → R2^2	•	•	•	•	•
INC	INCA													4C	2	1	A + 1 → A	•	↕	↕	↕	•
	INCB													5C	2	1	B + 1 → B	•	↕	↕	↕	•
	INC				0C	6	2	6C	6+	2+	7C	7	3				M + 1 → M	•	↕	↕	↕	•
JMP					0E	3	2	6E	3+	2+	7E	4	3				EA^3 → PC	•	•	•	•	•
JSR					9D	7	2	AD	7+	2+	BD	8	3				Jump to Subroutine	•	•	•	•	•
LD	LDA	86	2	2	96	4	2	A6	4+	2+	B6	5	3				M → A	•	↕	↕	0	•
	LDB	C6	2	2	D6	4	2	E6	4+	2+	F6	5	3				M → B	•	↕	↕	0	•
	LDD	CC	3	3	DC	5	2	EC	5+	2+	FC	6	3				M:M + 1 → D	•	↕	↕	0	•
	LDS	10	4	4	10	6	3	10	6+	3+	10	7	4				M:M + 1 → S	•	↕	↕	0	•
		CE			DE			EE			FE											
	LDU	CE	3	3	DE	5	2	EE	5+	2+	FE	6	3				M:M + 1 → U	•	↕	↕	0	•
	LDX	8E	3	3	9E	5	2	AE	5+	2+	BE	6	3				M:M + 1 → X	•	↕	↕	0	•
	LDY	10	4	4	10	6	3	10	6+	3+	10	7	4				M:M + 1 → Y	•	↕	↕	0	•
		8E			9E			AE			BE											
LEA	LEAS							32	4+	2+							EA^3 → S	•	•	•	•	•
	LEAU							33	4+	2+							EA^3 → U	•	•	•	•	•
	LEAX							30	4+	2+							EA^3 → X	•	•	↕	•	•
	LEAY							31	4+	2+							EA^3 → Y	•	•	↕	•	•

Legend:

OP	Operation Code (Hexadecimal)	$\overline{M}$	Complement of M	↕	Test and set if true, cleared otherwise
~	Number of MPU Cycles	→	Transfer Into	•	Not Affected
#	Number of Program Bytes	H	Half-carry (from bit 3)	CC	Condition Code Register
+	Arithmetic Plus	N	Negative (sign bit)	:	Concatenation
–	Arithmetic Minus	Z	Zero (Reset)	∨	Logical or
•	Multiply	V	Overflow, 2's complement	∧	Logical and
		C	Carry from ALU	⊻	Logical Exclusive or

Fig. 12-31 MC6809 instruction set summary.

		Addressing Modes																5	3	2	1	0
		Immediate			Direct			Indexed[1]			Extended			Inherent				H	N	Z	V	C
Instruction	Forms	Op	~	#	Op	~	#	Op	~	#	Op	~	#	Op	~	#	Description					
LSL	LSLA													48	2	1		•	↕	↕	↕	↕
	LSLB													58	2	1		•	↕	↕	↕	↕
	LSL				08	6	2	68	6+	2+	78	7	3					•	↕	↕	↕	↕
LSR	LSRA													44	2	1		•	0	↕	•	↕
	LSRB													54	2	1		•	0	↕	•	↕
	LSR				04	6	2	64	6+	2+	74		3					•	0	↕	•	↕
MUL														3D	11	1	A × B – D (Unsigned)	•	•	↕	•	9
NEG	NEGA													40	2	1	$\overline{A}$ + 1 – A	8	↕	↕	↕	↕
	NEGB													50	2	1	$\overline{B}$ + 1 – B	8	↕	↕	↕	↕
	NEG				00	6	2	60	6+	2+	70	7	3				$\overline{M}$ + 1 – M	8	↕	↕	↕	↕
NOP														12	2	1	No Operation	•	•	•	•	•
OR	ORA	8A	2	2	9A	4	2	AA	4+	2+	BA	5	3				A V M – A	•	↕	↕	0	•
	ORB	CA	2	2	DA	4	2	EA	4+	2+	FA	5	3				B V M – B	•	↕	↕	0	•
	ORCC	1A	3	2													CC V IMM – CC				7	
PSH	PSHS	34	5+[4]	2													Push Registers on S Stack	•	•	•	•	•
	PSHU	36	5+[4]	2													Push Registers on U Stack	•	•	•	•	•
PUL	PULS	35	5+[4]	2													Pull Registers from S Stack	•	•	•	•	•
	PULU	37	5+[4]	2													Pull Registers from U Stack	•	•	•	•	•
ROL	ROLA													49	2	1		•	↕	↕	↕	↕
	ROLB													59	2	1		•	↕	↕	↕	↕
	ROL				09	6	2	69	6+	2+	79	7	3					•	↕	↕	↕	↕
ROR	RORA													46	2	1		•	↕	↕	•	↕
	RORB													56	2	1		•	↕	↕	•	↕
	ROR				06	6	2	66	6+	2+	76	7	3					•	↕	↕	•	↕
RTI														3B	6/15	1	Return From Interrupt					7
RTS														39	5	1	Return from Subroutine	•	•	•	•	•
SBC	SBCA	82	2	2	92	4	2	A2	4+	2+	B2	5	3				A – M – C – A	8	↕	↕	↕	↕
	SBCB	C2	2	2	D2	4	2	E2	4+	2+	F2	5	3				B – M – C – B	8	↕	↕	↕	↕
SEX														1D	2	1	Sign Extend B into A	•	↕	↕	0	•
ST	STA				97	4	2	A7	4+	2+	B7	5	3				A – M	•	↕	↕	0	•
	STB				D7	4	2	E7	4+	2+	F7	5	3				B – M	•	↕	↕	0	•
	STD				DD	5	2	ED	5+	2+	FD	6	3				D – M:M + 1	•	↕	↕	0	•
	STS				10 DF	6	3	10 EF	6+	3+	10 FF	7	4				S – M:M + 1	•	↕	↕	0	•
	STU				DF	5	2	EF	5+	2+	FF	6	3				U – M:M + 1	•	↕	↕	0	•
	STX				9F	5	2	AF	5+	2+	BF	6	3				X – M:M + 1	•	↕	↕	0	•
	STY				10 9F	6	3	10 AF	6+	3+	10 BF	7	4				Y – M:M + 1	•	↕	↕	0	•
SUB	SUBA	80	2	2	90	4	2	A0	4+	2+	B0	5	3				A – M – A	8	↕	↕	↕	↕
	SUBB	C0	2	2	D0	4	2	E0	4+	2+	F0	5	3				B – M – B	8	↕	↕	↕	↕
	SUBD	83	4	3	93	6	2	A3	6+	2+	B3	7	3				D – M:M + 1 – D	•	↕	↕	↕	↕
SWI	SWI[6]													3F	19	1	Software Interrupt 1	•	•	•	•	•
	SWI2[6]													10 3F	20	2	Software Interrupt 2	•	•	•	•	•
	SWI3[6]													11 3F	20	1	Software Interrupt 3	•	•	•	•	•
SYNC														13	≥ 4	1	Synchronize to Interrupt	•	•	•	•	•
TFR	R1, R2	1F	6	2													R1 – R2[2]	•	•	•	•	•
TST	TSTA													4D	2	1	Test A	•	↕	↕	0	•
	TSTB													5D	2	1	Test B	•	↕	↕	0	•
	TST				0D	6	2	6D	6+	2+	7D	7	3				Test M	•	↕	↕	0	•

Notes:

1. This column gives a base cycle and byte count
2. R1 and R2 may be any pair of 8 bit or any pair of 16 bit registers.
   The 8 bit registers are: A, B, CC, DP
   The 16 bit registers are: X, Y, U, S, D, PC
3. EA is the effective address
4. The PSH and PUL instructions require 5 cycles plus 1 cycle for each byte pushed or pulled
5. 5(6) means: 5 cycles if branch not taken, 6 cycles if taken (Branch instructions).
6. SWI sets I and F bits. SWI2 and SWI3 do not affect I and F.
7. Conditions Codes set as a direct result of the instruction
8. Value of half-carry flag is undefined.
9. Special Case – Carry set if b7 is SET.

Fig. 12-32 MC6809 instruction set summary (*continued*).

Instruction	Forms	Addressing Mode Relative OP	–	/	Description	5 H	3 N	2 Z	1 V	0 C
BCC	BCC LBCC	24 10 24	3 5(6)	2 4	Branch C = 0 Long Branch C = 0	• •	• •	• •	• •	• •
BCS	BCS LBCS	25 10 25	3 5(6)	2 4	Branch C = 1 Long Branch C = 1	• •	• •	• •	• •	• •
BEQ	BEQ LBEQ	27 10 27	3 5(6)	2 4	Branch Z = 0 Long Branch Z = 0	• •	• •	• •	• •	• •
BGE	BGE LBGE	2C 10 2C	3 5(6)	2 4	Branch ≥ Zero Long Branch ≥ Zero	• •	• •	• •	• •	• •
BGT	BGT LBGT	2E 10 2E	3 5(6)	2 4	Branch > Zero Long Branch > Zero	• •	• •	• •	• •	• •
BHI	BHI LBHI	22 10 22	3 5(6)	2 4	Branch Higher Long Branch Higher	• •	• •	• •	• •	• •
BHS	BHS LBHS	24 10 24	3 5(6)	2 4	Branch Higher or Same Long Branch Higher or Same	• •	• •	• •	• •	• •
BLE	BLE LBLE	2F 10 2F	3 5(6)	2 4	Branch ≤ Zero Long Branch ≤ Zero	• •	• •	• •	• •	• •
BLO	BLO LBLO	25 10 25	3 5(6)	2 4	Branch lower Long Branch Lower	• •	• •	• •	• •	• •
BLS	BLS LBLS	23 10 23	3 5(6)	2 4	Branch Lower or Same Long Branch Lower or Same	• •	• •	• •	• •	• •
BLT	BLT LBLT	2D 10 2D	3 5(6)	2 4	Branch < Zero Long Branch < Zero	• •	• •	• •	• • •	• •
BMI	BMI LBMI	2B 10 2B	3 5(6)	2 4	Branch Minus Long Branch Minus	• •	• •	• •	• •	• •
BNE	BNE LBNE	26 10 26	3 5(6)	2 4	Branch Z ≠ 0 Long Branch Z ≠ 0	• •	• •	• •	• •	• •
BPL	BPL LBPL	2A 10 2A	3 5(6)	2 4	Branch Plus Long Branch Plus	• •	• •	• •	• •	• •
BRA	BRA LBRA	20 16	3 5	2 3	Branch Always Long Branch Always	• •	• •	• •	• •	• •
BRN	BRN LBRN	21 10 21	3 5	2 4	Branch Never Long Branch Never	• •	• •	• •	• •	• •
BSR	BSR LBSR	8D 17	7 9	2 3	Branch to Subroutine Long Branch to Subroutine	• •	• •	• •	• •	• •
BVC	BVC LBVC	28 10 28	3 5(6)	2 4	Branch V = 0 Long Branch V = 0	• •	• •	• •	• •	• •
BVS	BVS LBVS	29 10 29	3 5(6)	2 4	Branch V = 1 Long Branch V = 1	• •	• •	• •	• •	• •

Fig. 12-33 MC6809 branch instructions.

monic (STA), the *operand* (#$49FF), and the *comment* (SET CARRY FLAG). The assembler prints the equivalent machine code for each line of instruction on the left. The first column on the left represents the program counter, or the address of the first byte of machine code for that instruction. The next two columns contain the actual machine code that can be executed by the processor: the OP-CODEs and the operands. Some lines do not have any machine code since they contain assembler directives for the assembler program's information only. Note that the assembler also generates a symbol table which lists the addresses for all the labels and constants declared by the programmer.

An assembler makes the generation of machine language programs easier, faster, and more error-free. The programmer does not have to look up any OP-CODEs. The assembler "knows" the OP-CODEs for all the mnemonics and for all the various addressing modes. The programmer is free to assign label names that make sense and help keep track of program function and entry points. This also makes it easy to specify branches. For example, the *BCC clear* instruction in Fig. 12-35 specifies SCAN as the operand. *SCAN* is a label assigned earlier in the assembly listing. Look at the machine code columns to the left of BCC, and you will find $24 (the OP-CODE for BCC) followed by $E4, the relative address generated by the assembler. Assemblers also present some helpful messages if there are errors in the assembly program. A 0 error message indicates a good assembly. However, this is no guarantee that the machine code will execute as planned. There can still be logical errors, errors of omission, timing errors, and other types of errors in the code.

After assembly, another program, called a *debugger*, may be used to help find errors in the program. Debuggers allow the programmer to set breakpoints in a program. A *breakpoint* allows a part of the program to execute, before execution stops at that breakpoint. At this time the programmer can examine memory contents to determine whether the software is doing what was intended. Debuggers also allow single stepping, making it possible to check software operation on an instruction by instruction basis.

We have learned that many industrial computers are very small. They are dedicated to a special task or a group of tasks and often have limited memory. How can software be developed for these computers? Editors, assemblers, and debuggers often will not run on such small systems. Development is done on a larger computer in these cases. In fact, many microprocessor manufacturers offer a computer development system for these situations. After the programs are written, assembled, and debugged on the large system, they are downloaded to the small computers or may be burned into EPROMs and transferred to the small computer as ROM programs.

What can you do in the way of software development with a dedicated microcomputer or microcontroller if a development system is not available? Often, not very much; however, some small systems do have a monitor program stored in ROM to permit some minor software development and debugging. For example, Motorola offers ASSIST09 to support the MC6809 microprocessor. Monitors often only consume about 2K bytes of memory and can be included even in a small system. A serial I/O port and a terminal will normally be required to utilize a monitor program.

Figure 12-36 lists the commands available in the

Type	Forms	Non Indirect				Indirect			
		Assembler Form	Postbyte OP Code	+ ~	+ #	Assembler Form	Postbyte OP Code	+ ~	+ #
Constant Offset From R (twos complement offset)	No Offset	,R	1RR00100	0	0	[,R]	1RR10100	3	0
	5 Bit Offset	n, R	0RRnnnnn	1	0	defaults to 8-bit			
	8 Bit Offset	n, R	1RR01000	1	1	[n, R]	1RR11000	4	1
	16 Bit Offset	n, R	1RR01001	4	2	[n, R]	1RR11001	7	2
Accumulator Offset From R (twos complement offset)	A — Register Offset	A, R	1RR00110	1	0	[A, R]	1RR10110	4	0
	B — Register Offset	B, R	1RR00101	1	0	[B, R]	1RR10101	4	0
	D — Register Offset	D, R	1RR01011	4	0	[D, R]	1RR11011	7	0
Auto Increment/Decrement R	Increment By 1	,R+	1RR00000	2	0	not allowed			
	Increment By 2	,R++	1RR00001	3	0	[,R++]	1RR10001	6	0
	Decrement By 1	,-R	1RR00010	2	0	not allowed			
	Decrement By 2	,--R	1RR00011	3	0	[,--R]	1RR10011	6	0
Constant Offset From PC (twos complement offset)	8 Bit Offset	n, PCR	1XX01100	1	1	[n, PCR]	1XX11100	4	1
	16 Bit Offset	n, PCR	1XX01101	5	2	[n, PCR]	1XX11101	8	2
Extended Indirect	16 Bit Address	—	—	—	—	[n]	10011111	5	2

R = X, Y, U or S X = 00 Y = 01
X = Don't Care U = 10 S = 11

$\overset{+}{\sim}$ and $\overset{+}{\#}$ indicate the number of additional cycles and bytes for the particular variation

Fig. 12-34 Postbyte formation for the indexed addressing mode.

Motorola ASSIST09 monitor program. Note that breakpoint and trace commands are available for debugging. Commands are also available for examining and changing memory contents and the contents of the processor registers. There is even an offset command to calculate relative addresses for branch instructions. Monitor programs are easy to learn and use. They are useful for entering and debugging small machine language programs. As an example of how they work, refer to Fig. 12-37. This is an example of a memory dump as printed on the terminal. The dump is in response to the display command typed into the terminal:

D F880,F8FF

When this command is entered (by hitting RETURN on the terminal) ASSIST09 responds with a dump of memory locations $F880 to $F8FF. Note that all values are in hexadecimal. Also note that the ASCII contents of memory are printed to the right. If the contents of any memory location are not a printable

```
                        LABEL   MNEMONIC OPERAND          COMMENT COLUMN (OPTIONAL)
                        COLUMN  COLUMN   COLUMN
-------------------------------------------------------------------------------------
                  0008  PAUSE   EQU      11         ASSIST09 break+check service
                  F200  PORTB   EQU      $F200      VIA port B
                  F202  DDRB    EQU      $F202      port B data direction reg.
                  6000  DTIME   EQU      $6000      DELAY TIME
                        *
0000 CC    49FF         START   LDD      #$49FF     ROLA op code + DDRB
0003 A7    8C 08                STA      <SCAN,PCR
0006 F7    F202                 STB      DDRB       set VIA port B as outputs
0009 CC    0008                 LDD      #$0008
000C 1A    01                   ORCC     #1         set carry flag

000E 49                 SCAN    ROLA                shift to next bit
000F B7    F200                 STA      PORTB      light 1 LED
0012 8E    6000                 LDX      #DTIME
0015 30    1F           DELAY   LEAX     -1,X       delay for a while
0017 26    FC                   BNE      DELAY
0019 5A                         DECB                all 8 LEDs yet?
001A 26    0A                   BNE      CONT1      no
001C C6    0F                   LDB      #$F        yes
001E E8    8C ED                EORB     <SCAN,PCR switch rotate direction
0021 E7    8C EA                STB      <SCAN,PCR L-to-R-to-L
0024 C6    07                   LDB      #7         reset bit counter  .
0026 3F                 CONT1   SWI                 check for "FREEZE" or "CANCEL"
0027 08                         FCB      PAUSE
0028 24    E4                   BCC      SCAN       repeat if no CANCEL
002A 39                         RTS                 otherwise, return to  ASSIST09

                                END

0 ERROR(S) DETECTED

SYMBOL TABLE:

CONT1  0026   DDRB   F202   DELAY  0015   DTIME  6000   PAUSE  0008
PORTB  F200   SCAN   000E   START  0000
```

Fig. 12-35 Machine language program development example.

Command Name	Description	Command Entry
Breakpoint	Set, clear, display, or delete breakpoints	B
Call	Call Program as subroutine	C
Display	Display memory block in hex and ASCII	D
Encode	Return indexed postbyte value	E
Go	Start or resume program execution	G
Load	Load memory from tape	L
Memory	Examine or alter memory	M
	Memory change or examine last referenced	/
	Memory change or examine	hex/
Null	Set new character and new line padding	N
Offset	Compute branch offsets	O
Punch	Punch memory on tape	P
Registers	Display or alter registers	R
Stlevel	Alter stack trace level value	S
Trace	Trace number of instructions	T
	Trace one instruction	.
Verify	Verify tape to memory load	V
Window	Set a window value	W

Fig. 12-36 Assist 09 monitor commands.

```
      0  1  2  3  4  5  6  7  8  9  A  B  C  D  E  F
F880 01 A2 01 BF 01 D7 01 CF 01 B3 01 B1 01 CC 01 B7   ................
F890 00 5C 01 8B 02 68 01 DD 17 02 3B EE 6A 33 5F DD   ......h....;.j3_.
F8A0 FB 26 11 17 06 B3 50 5A 2B DA 11 A3 A1 26 FB EF   .&....PZ+....&..
F8B0 6A 16 02 34 0F FB 37 D6 C1 DB 10 22 02 25 EF 6A   j..4..7....".%.j
F8C0 58 33 8C BC EC C5 6E C8 54 43 32 20 41 53 53 49   X3....n.TC2 ASSI
F8D0 53 54 30 39 20 5B 56 31 2E 32 5D 04 10 DF 97 ED   ST09 [V1.2]....
F8E0 61 26 09 AD 9D E6 EF 30 8C DE 3F 03 9E F6 6F CB   a&.....0..?...o.
F8F0 86 C0 A7 DE 3F D6 35 01 34 01 17 06 5C 2A 0C 50   ....?.5.4....*.P
```

Fig. 12-37 Sample output for monitor display command.

ASCII character, a period (.) is printed. None of the memory locations from $F880 to $F8FF contains a printable character. If you examine Fig. 12-37 you can see that this particular dump was from an area of memory that contained the ASSIST09 monitor program.

There are other software development and debugging aids. A *disassembler* is a program that converts machine code into an assembly listing. It is useful for analyzing code to determine how it works (or why it doesn't) and how to modify it for a new application. Some systems offer linkers to allow smaller programs to be gathered into a single larger program. Diagnostic software may be available for testing various parts of the system. This will be covered in the next section.

REVIEW QUESTIONS

44. Refer to Fig. 12-31. What is the OP-CODE for the LDD extended instruction? How many memory bytes will it consume? How many machine cycles will it take?

45. Refer to Fig. 12-31. Which processor flags will be set or cleared by the LDD instruction?

46. Refer to Fig. 12-31. Which processor flags will not be affected by the LDD instruction?

47. Refer to Fig. 12-33. How many bytes are required for the BCC instruction? For the LBCC instruction? Why?

48. Refer to Fig. 12-33. Which flags are affected by the branch instructions?

49. Refer to Fig. 12-35. Explain the line of machine code at the left that begins at address $000F.

12-8 TROUBLESHOOTING AND MAINTENANCE

As always, the troubleshooting procedure should begin with the obvious and preliminary checks and proceed from there on an orderly and logical basis. Verify proper power supply operation, including any separate negative and positive supplies used for RS-232C data communication. Do not forget to include all parts of the system that may be connected to the processor. Sometimes, a microprocessor will "hang up" when attempting to communicate with another device that is powered down or not ready to respond for some other reason. Some industrial computers may contain battery back-up circuits. Test the battery voltage with the main power off.

Try to determine whether something has been changed since the last time the equipment was operational. For example, has new software been installed in the system, or has the software been modified? It may be necessary to revert to earlier software to determine where the problem is actually located. Software problems can appear to be hardware problems, and the opposite is also true. Has another part of the system been changed or reprogrammed? In some cases, what appears to be a defective computer is actually a terminal or other device that is not properly configured. Modern serial terminals are capable of quite a few baud rates, different word sizes, a few stop bit choices, and several parity options. These can be changed by setting switches on the back of the terminal or perhaps inside the terminal. They can also be changed from the keyboard on some terminals. Refer to the relevant manuals and make certain that the terminal is properly set up to communicate with the computer. When doing so, you may encounter half- and full-duplex options. The *half-duplex option* allows communication between the terminal and the computer in only one direction at a time. The *full-duplex option* allows communication in two directions independently at the same time. If the terminal displays or prints all typed characters double, try changing it to full-duplex. If nothing is displayed or printed, try changing it to half-duplex. Finally, check the cable between the terminal and the computer. Sometimes pins 2 and 3 on one of the DB-25 connectors must be reversed. This happens because 2 must connect to 2 for some equipment interfaces, and 2 must connect to 3 for others.

Some systems include diagnostic software. One example is a *RAM test program*. Such a program writes various bit patterns to RAM locations and then reads them back to verify that memory is working as it should. Another example is software that exercises various parts of the system such as display outputs, control outputs, and inputs. The diagnostic package may be menu-driven and sequentially structured. A menu will appear on the terminal, allowing the technician to select from several test options. When a test is selected, the program may ask the technician to verify that a display or some other output is responding as described on the screen or printout. As each question is answered, the next test is started. The program may also request that the technician actuate certain controls or apply test signals to various inputs. A "walk-through" of this type can be very effective in verifying system operation and diagnosing malfunctions. Some systems can even be diagnosed from many miles away. Modems have been used to connect a diagnostic computer at the factory to a processor in the field.

If the defective computer works at least partially, a monitor program may be a valuable troubleshooting aid. Examine several memory locations in every block. Consult the memory map. If 8K devices are used, check several addresses in each device. If the devices are RAM, verify that the data can be changed. Try writing $FF to a location and then read it. Then change it to $00 and read it again. Although this is less conclusive than a diagnostic program that checks every location, it still may be helpful. A monitor program can also be used to check support devices such as PIAs and ACIAs. You can use the monitor to set the data direction for output and then write $FF followed by $00 to the output registers. An oscilloscope, logic probe, or meter can be used to test the output pins for the proper response. A few cautions are in order here. First, don't forget that writing information to an ACIA could disable the ability to communicate via the terminal. If this happens, a system reset should reinitialize the ACIA and reestablish communications with the terminal. Second, be aware that devices such as motors or hydraulic valves could be activated by changing data at certain addresses. If at all possible, troubleshoot with high-energy systems disabled.

If BASIC is available on the system, it can also serve as a troubleshooting aid. Some versions have a *PEEK command* that can be used to examine memory locations and a *POKE command* that can be used to change them. BASIC can also be used to repeatedly strobe an address or a device to allow oscilloscope troubleshooting. For example, suppose that you wish to verify that an address decoder at $E000 (decimal 57,344) is working. If the system is operating as it should, the following program will produce a pulse train at the decoder output.

```
10 POKE 57344,0
20 GOTO 10
```

BASIC can also be used to strobe a device repeatedly to facilitate oscilloscope testing. For example, assume that a PIA is decoded, beginning at address $FF40 (decimal 65,344). The following program initializes the PIA (program lines 10 through 30) and then repeatedly toggles the outputs. The REM statements are for documentation purposes and are ignored by BASIC:

```
10 POKE 65346,0: POKE 65347,0: REM clears bit 2
to select DDRA and DDRB
20 POKE 65344,255: POKE 65345,255: REM sets all
bits for output
30 POKE 65346,4: POKE 65347,4: REM sets bit 2
to select ORA and ORB
40 POKE 65344,0: POKE 65345,0: REM toggle all
16 outputs low
50 POKE 65344,255: POKE 65345,255: REM toggle
all 16 high
60 GOTO 40: REM continue toggling outputs
```

If an oscilloscope is not available, a logic probe can also be used to check for toggling. If only a meter is available, the toggling can be slowed down by adding this delay line to the program:

```
45 FOR X=1 TO 200: NEXT X: REM change "200"
value to change delay
```

If a monitor program is available, similar diagnostic routines can be hand-assembled, entered, and executed in machine code.

The microcomputer may not work at all or may not work well enough to use the monitor or BASIC. Also, many industrial computers are small, dedicated units. They often do not have a monitor, BASIC, or even a terminal connected to them. In these cases, an oscilloscope may be used to examine the various address lines, data lines, clock lines, control lines, and status outputs of the microprocessor to determine whether there is proper activity. The clock signals must always be present. If the system uses a microprocessor with an on-chip oscillator, the lack of clock signals would indicate a defective microprocessor or crystal. It is normal to expect activity on the address bus and the data bus. If there is no activity or limited activity, the microprocessor may be in a halt mode, may be waiting for an interrupt, or possibly may be in a DMA or sync mode. If this is the case, the processor status pins should be checked with an oscilloscope to determine which mode the processor is in; these pins are BA and BS on the Motorola MC6809 processor. When BA and BS are both low, the processor is in the normal running mode. The three other status conditions established by the state of BA and BS are the *interrupt or reset acknowledge mode*, the *sync acknowledge mode*, and *the halt or bus grant mode*. It may also be necessary to investigate the halt, reset, MRDY, DMA/BREQ, and interrupt pins. Set the oscilloscope for dc coupling and establish a 0-V reference point on the screen when analyzing a microprocessor system. This will allow both signal activity and dc level to be readily determined.

Assuming that the oscilloscope shows bus activity, a check for chip select and device select signals will shed some light on what the processor is doing or not doing. It is normal to observe repeated ROM select signals as the processor executes the code stored there. Check for RAM select signals also and don't forget to check for I/O select signals. Support devices are initialized on power up and after a system reset. You may have to continue to reset the system while looking for select signals at these devices. These should occur immediately after the processor leaves the reset state, and there should be evidence that the processor is writing several bytes to each initialized device. If resets do not appear to initiate the expected actions, set the oscilloscope for a sweep speed of approximately 10 ms per division and connect the probe to the reset bus. An exponential ramp should be evident when the reset switch is released. You can also look at the read/write line and the input and output of buffer ICs to help localize difficulties.

A microprocessor is a very complex IC, with many functions and features. It is possible that a partial failure can occur. For example, a processor may do everything it should except provide the proper response to an interrupt. It may be possible to use a logic pulser to simulate an interrupt. An oscilloscope can be triggered on the pulser output, and the scope probe can be used to look for a ROM select signal. This procedure will determine whether the processor is fetching the interrupt vector as it should. Do not forget that the processor may mask an interrupt request.

Some advanced microprocessor troubleshooting equipment that is available provides quite a bit of information. A *logic analyzer* is an example; this piece of equipment has probes that are connected to the address bus, the data bus, the read write line, and the clock line. Qualifier switches can be set to make the instrument synchronize on a particular event. The logic analyzer stores bus events, which can be displayed as they occurred before and after the qualifying event. Some logic analyzers are very advanced and can present the data in binary form, hexadecimal form, and in the form of timing diagrams on a cathode-ray tube. Some even have *personality modules* for the popular microprocessors and can thereby produce *disassembly* listings on the screen or send the listings to a printer. *Signature analyzers* are also available for microprocessor troubleshooting and examine the data bus and the address bus of a microcomputer in real time. The pattern (or signature) is compared to a pattern previously stored in memory from a normally operating system. Any differences between the stored signature and the actual signature can be analyzed to provide clues as to what is wrong with the system.

The statistics of part failure predict that the most complex devices are the most failure-prone. This includes the microprocessor, support devices, and memory ICs. If a system uses sockets and if spare ICs are available, component swapping is a reasonable procedure. Make sure the power is off and observe proper handling procedures to prevent damage from static electricity. Also make sure that all devices are properly and firmly seated in their sockets.

REVIEW QUESTIONS

50. Refer back to the BASIC diagnostic program for testing the PIA. Could this program be modified to test the input performance of the PIA? How?

51. How will the system react in response to the modification referred to in the prior question?

52. The various lines of the address and data buses should show continuous activity when the processor is in the ________ mode.

CHAPTER REVIEW QUESTIONS

12-1. How many flags are there in the MC6809 microprocessor?

12-2. The address pins on a microprocessor are unidirectional, and the data pins are ______.

12-3. Which input pins are used to signal the microprocessor that an external hardware event that requires attention has occurred?

12-4. Which pin tells whether the microprocessor is writing or reading data?

12-5. A group of instructions and data in memory used to direct the operation of a microprocessor is called a ______.

12-6. A CMPA is an example of a(n) ______.

12-7. What often happens when a microprocessor begins execution at the wrong address?

12-8. The immediate addressing mode is used to load the U pointer register. How many operand bytes must follow the OP-CODE?

12-9. How many bytes must follow the OP-CODE for extended addressing?

12-10. The *S* register is to be loaded from memory locations $F011 and $F012 by using the extended addressing mode. What must follow the OP-CODE for LDS?

12-11. Assume that the DP register contains $2E. What is the effective address when the OP-CODE for LDA is followed by $34?

12-12. Which indexed addressing mode points to the address of the effective address?

12-13. A relative address is added to the contents of the ______ when the result of a branch test is true.

12-14. A branch test is located at $1114, and the relative address is located at $1115. What is the required hex value of the relative address to cause a backward branch to $1100? Don't forget that the program counter will be pointing to $1116 when the effective address is calculated.

12-15. Accumulator *A* contains $3E. What will it contain after nine consecutive RORA operations?

12-16. Which instruction always transfers program flow to the effective address by using extended, direct, or indexed addressing? Which instruction does the same thing but uses relative addressing?

12-17. What is the purpose of the F flag?

12-18. What is the purpose of the I flag?

12-19. Which flag is set to tell the processor to pull all the registers but one from the hardware stack? Which register is not pulled?

12-20. Where are the interrupt vectors stored in an MC6908 system?

12-21. Which interrupt stacks only the CC register and the program counter?

12-22. Assume that an MC6809 is in the SYNC state. What happens when an NMI signal is received?

12-23. In a microprocessor-based system, which device must leave the reset state last? Why?

12-24. Which MC6809 signal is equivalent to the phase 2 clock signal required by the VIA?

12-25. Assume that a VIA is mapped into memory beginning at $82E0. What address will the processor write to in order to change PB7 to a timer-controlled output?

12-26. Serial data communications can use voltage or current signals. Which is the most popular? Which specific voltage standard is the most popular?

12-27. Which serial mode requires start and stop bits?

12-28. The RS-422 serial standard uses balanced transmission for high bandwidth and ______ data rates.

12-29. For what type of data communication is an ACIA designed?

12-30. How can you determine which mode the MC6809 processor is in when troubleshooting?

12-31. What would repeated ROM select signals indicate?

12-32. What would repeated I/O select signals indicate?

ANSWERS TO REVIEW QUESTIONS

1. 16 M (approximately 16 million) **2.** flag **3.** 16 **4.** 8 **5.** the valve is turned on **6.** OP-CODE **7.** no; no readings could be stored **8.** inherent **9.** one **10.** \$34 or \$43 **11.** \$B9 **12.** \$01 **13.** the S register **14.** \$A9; a 2-byte offset **15.** \$80; after the effective address is determined **16.** 156; −100 **17.** \$05; \$06 **18.** \$32 **19.** positive number; no **20.** yes **21.** \$F0 **22.** no **23.** program counter **24.** the S register **25.** nested **26.** decremented **27.** the software will crash **28.** the motor will run counterclockwise **29.** 458,753 μs **30.** BSR or LBSR **31.** \$7000; \$7FFF; 4K (4096 bytes) **32.** from the falling edge of E to the rising edge of Q **33.** 500 ns; 750 ns **34.** \$C13E; yes **35.** \$800F; \$8FFF; \$FF1 (4081) **36.** four **37.** by reading control register *A* and testing bits 6 and 7 **38.** a latch **39.** when the counter high byte is written; when the counter times out **40.** \$F3; \$01 **41.** 7.843 kHz; rectangular (nearly square) **42.** timer 2; PB6 **43.** 1920 **44.** \$FC; 3; 6 **45.** N, Z, and V **46.** H and C **47.** 1; 4; 2-byte OP-CODE plus 2-byte relative address **48.** none **49.** \$B7 is the OP-CODE for store accumulator *A* extended, and \$F200 is the address where it will be stored **50.** yes; change line 20 to POKE 0 to both locations; replace line 40 with PRINT PEEK(65344): PRINT PEEK(65345); delete line 50 **51.** the decimal equivalent of port A input data will be printed followed by the port B data, repeatedly **52.** normal (running)

13

DATA CONVERSION, COMMUNICATION, AND STORAGE

Advances in computer electronics have produced a major impact on industry. Many industrial systems are connected to, monitored by, or in some way controlled by computers. Some of these systems are sophisticated and involve both digital and analog signals. This chapter focuses on conversions between the digital and analog worlds and examines digital communications in more detail. It also introduces mass storage devices for digital data and programs.

13-1 DIGITAL-TO-ANALOG CONVERSION

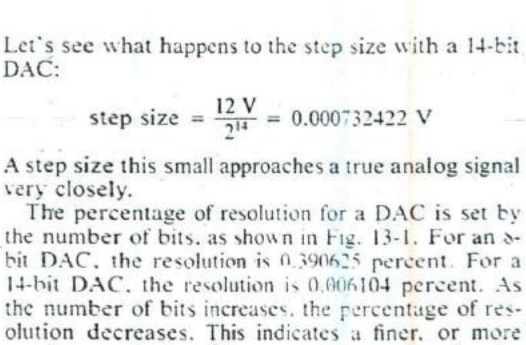
The *digital-to-analog converter* (DAC) accepts a binary input and produces a corresponding analog output, which is usually in the form of a voltage or a current. Figure 13-1 demonstrates the basic idea. A number of binary inputs are shown at the left, and a single analog output is shown at the right. In the strictest sense, a true analog signal is capable of an infinite number of measurement values. Can the DAC produce an infinite number of output levels? It cannot. A DAC produces discrete steps at its output: one for every possible binary input. With an adequate number of binary inputs, the step size will be small enough so that the output approaches a true analog signal.

The DAC shown in Fig. 13-1 has eight binary inputs. It is connected to a 12-V supply. Suppose that the output can span from 0 to 12 V. It is possible to determine the step size of the output voltage by dividing the full span by 2^N where N is the number of binary inputs; $2^8 = 256$ and $12/256 = 0.046875$ V. This step size is small enough to meet many industrial needs for an analog signal. In those cases where it is not, 10-, 12-, and 14-bit DACs are available. Let's see what happens to the step size with a 14-bit DAC:

$$\text{step size} = \frac{12\text{ V}}{2^{14}} = 0.000732422\text{ V}$$

A step size this small approaches a true analog signal very closely.

The percentage of resolution for a DAC is set by the number of bits, as shown in Fig. 13-1. For an 8-bit DAC, the resolution is 0.390625 percent. For a 14-bit DAC, the resolution is 0.006104 percent. As the number of bits increases, the percentage of resolution decreases. This indicates a finer, or more precise, resolution. Of course, the step size also decreases as the number of bits increases. The overall accuracy is a function of the number of bits and the accuracy of the DAC itself. Digital-to-analog con-



$$\text{Step size} = \frac{\text{Full value}}{2^N}$$

$$\text{\% Resolution} = \frac{1}{2^N} \times 100$$

$$\text{Max output} = (2^N - 1) \times \text{step size}$$

Fig. 13-1 Basic DAC.

verter manufacturers often rate *accuracy* as a percentage. Do not confuse this rating with the *resolution*, which is fixed by the number of bits. Manufacturers may also rate the accuracy as equal to plus or minus 1/2 LSB, or plus or minus 1/4 LSB, and so on. The least significant bit produces the smallest possible change in the output. An 8-bit DAC with a plus or minus 1/2 LSB rating has an accuracy of plus or minus approximately 0.19 percent. Note that this is half the percentage of resolution calculated by the equation shown in Fig. 13-1. An 8-bit DAC with a 1/4 LSB rating has an accuracy of plus or minus approximately 0.1 percent, which is one-fourth of its resolution.

Figure 13-1 shows that the maximum output from a DAC is found by multiplying $2^N - 1$ times the step size. We found the step size earlier for the 8-bit DAC with a 12-V supply to be 0.046875 V. The maximum output is therefore $(256 - 1) \times 0.046875\ \text{V} = 11.953125$ V. Note that it is not equal to 12 V. The minimum output is 0 V and occurs with a binary input of 00000000; the maximum output is one step size less than the supply and occurs with a binary input of 11111111 (decimal 255). To find the output at any intermediate value, simply convert the binary input to decimal and multiply it times the step size. For example, with the same step size and a binary input of 10101011 (decimal 171) the analog output would be nominally 8.01563 V. The actual output voltage could vary as much as half a step above or below nominal, depending on the accuracy rating of the DAC.

Figure 13-2 shows the circuit for a 5-bit binary weighted adder. It uses an op amp as an inverting adder. Assume that V_{ref} is -5 V. The output range will be from 0 to almost 5 V. The step size will be

$$\frac{5}{2^5} = 0.15625\ \text{V}$$

As the circuit is drawn, none of the switches is closed. This is equivalent to a binary input of 00000. No voltage is applied to the input of the op amp, and $V_{out} = 0$. What happens if binary 00001 is entered? This sets the LSB and closes the switch connecting V_{ref} to the input through the $32R$ resistor. The voltage gain of the op amp is given by

$$-\frac{R_F}{R_{in}}$$

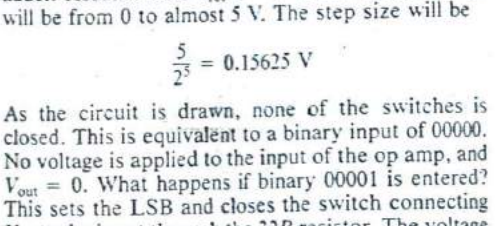


Fig. 13-2 Binary weighted D/A converter.

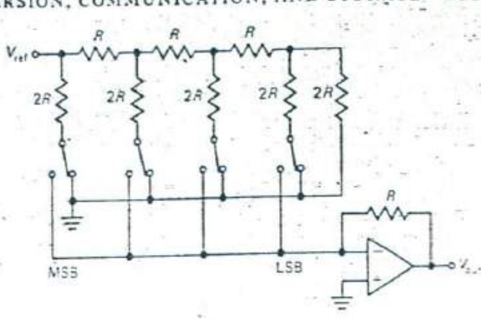


Fig. 13-3 *R-2R* ladder D/A converter.

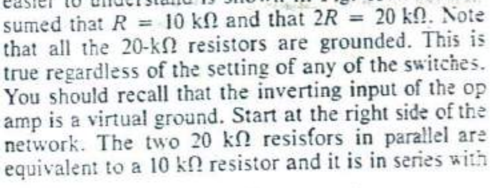
Since the input resistor is 32 times the value of the feedback resistor, the gain is $-1/32$. With a reference voltage of -5 V, the output will be

$$-\frac{1}{32} \times -5 = 0.15625\ \text{V}$$

This result agrees with the step size calculated previously.

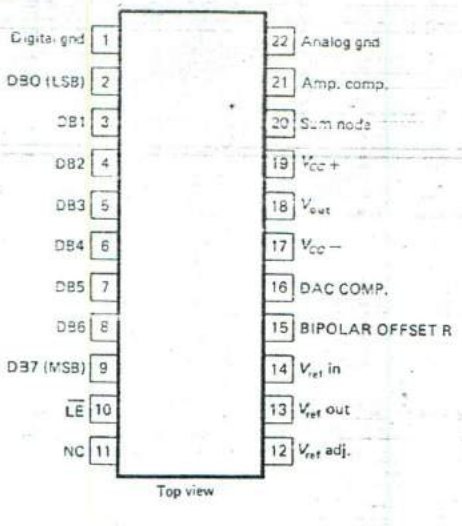
In practice, the switches shown in Fig. 13-2 would be replaced by transistor switches to allow logic signals to be directly applied to the D/A converter. Binary weighted DACs are limited to 5 bits because of the large range of resistor values required. It is very difficult to match resistors with ratios greater than 32:1. Therefore, any attempt to gain resolution by adding more bits will result in a loss of accuracy that negates the extra bits.

Figure 13-3 shows a D/A converter that is based on an *R-2R* ladder network. It requires only two resistor values and is therefore easier to implement. Each switch is a single-pole, double-throw type that connects the $2R$ elements to the input of the op amp or to ground. All switches are shown in the 0 position. An equivalent network that makes the circuit easier to understand is shown in Fig. 13-4. It is assumed that $R = 10\ \text{k}\Omega$ and that $2R = 20\ \text{k}\Omega$. Note that all the 20-kΩ resistors are grounded. This is true regardless of the setting of any of the switches. You should recall that the inverting input of the op amp is a virtual ground. Start at the right side of the network. The two 20 kΩ resisfors in parallel are equivalent to a 10 kΩ resistor and it is in series with

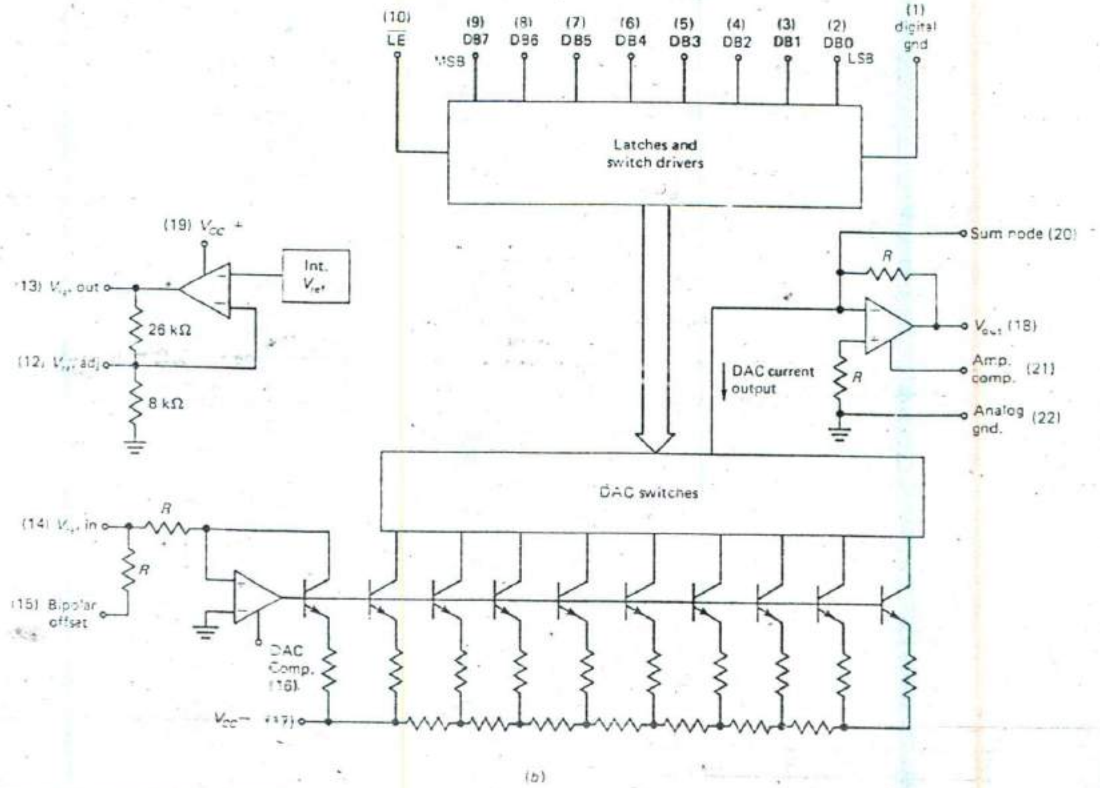


*These can represent actual ground or the virtual ground of the op amp, depending on the switch settings.

Fig. 13-4 Voltages around the *R-2R* network.



(a)



(b)

Fig. 13-5 Integrated circuit NE5018 D/A converter. (*a*) Pinouts. (*b*) Block diagram (Signetics).

a 10kΩ resistor for an equivalent resistance of 20 kΩ. This equivalent 20 kΩ is in parallel with the next 20 kΩ resistor for a resistance of 10 kΩ. It is in series with the next 10 kΩ for an equivalent resistance of 20 kΩ. You should see the pattern by now. No matter how many bits the *R*-2*R* network requires, the equivalent resistance values keep repeating.

Now look at the voltages shown in Fig. 13-4. The full −5 V appears at the first point indicated. Half that value appears at the next point, and then half that value appears at the next point, and so on. The *R*-2*R* network repeatedly divides the input voltage by 2. When any of the switches in Fig. 13-3 are set to binary 1, the full reference voltage or one of the divided values is applied through a 2*R* element to the input of the op amp. The feedback resistor has a value of *R*, and therefore half of the network voltage appears at the output. For example, if V_{ref} is −5 V and only the MSB is set, V_{out} = 2.5 V.

Most DACs are monolithic ICs such as the Signetics NE5018 shown in Fig. 13-5. This IC contains the *R*-2*R* ladder network, an adjustable reference supply, the DAC switches, input latches for the binary data, and an operational summing amplifier. The input latches make the device compatible with microprocessors. The binary inputs (pins 2 to 9) can be connected directly to the data bus. When not LE (pin 10) is low, the latches are transparent. When it goes high, the input data present at the moment of transition is latched and retained until not LE goes low again.

The NE5018 reference voltage is nominally 5 V. It can be trimmed with an external potentiometer to provide easy adjustment of full scale. It is temperature-compensated and therefore relatively stable. The maximum supply voltage is plus and minus 18 V with plus and minus 15 V being recommended. The settling time is rated at 2.3 μs. This is the time required for the output to stabilize (settle) after a binary input appears. Its rated accuracy is plus or minus 1/2 LSB. Signetics also manufactures the NE5019, which is almost identical but offers an improved accuracy of plus or minus 1/4 LSB. They also offer the NE5020, a 10-bit DAC rated at plus and minus 0.1 percent relative accuracy. It features two latch enable inputs, which allow an easy interface with 8-bit microprocessors. The processor can write data to the lower 8 bits and the upper 2 bits at two different addresses. Two write cycles are required to load a 10-bit word into the DAC when it is interfaced to an 8-bit bus.

The Signetics DAC can use its own internal reference supply or an external reference supply. Converters that use an external reference are sometimes referred to as *multiplying DACs*. The multiplying mode allows a signal applied to the reference input to be scaled according to the value of the binary input. Thus, a microprocessor could alter the level of an analog signal by sending a binary word to the DAC. Figure 13-6 shows another application. A microprocessor controls the binary input of the D/A converter. The resulting analog output is applied to the NE540 error amplifier. The other input to the error amplifier comes from the NE5520 LVDT signal conditioner. Any error between the command position from the microprocessor and the actual position will cause the motor to run in the direction necessary to eliminate the error.

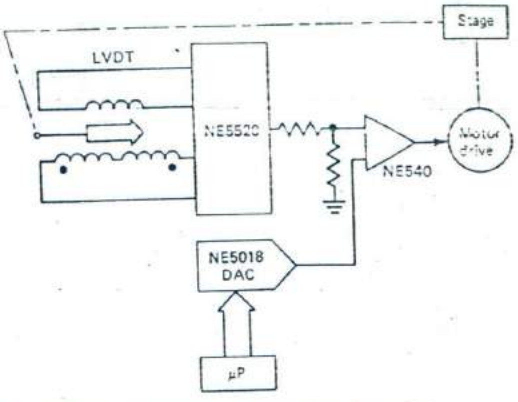


Fig. 13-6 Microprocessor control interface (Signetics).

REVIEW QUESTIONS

1. Calculate the step size for a 10-bit DAC that uses a 5-V supply. Calculate its maximum output voltage.

2. Calculate the percentage of resolution for a 10-bit digital-to-analog converter.

3. The manufacturer of a 10-bit DAC rates the accuracy at plus and minus 1/4 LSB. What is the accuracy expressed as a percentage?

4. Refer to Fig. 13-2. What is the output with a −10-V reference if the two left-most switches are closed and all other switches are open?

13-2 ANALOG-TO-DIGITAL CONVERSION

Analog-to-digital converters (ADCs) are used to convert analog voltages or currents to binary words. They are often based on digital-to-analog converters such as the ramp-type circuit shown in Fig. 13-7. Voltage V_{in} is an analog voltage that is to be converted to a binary value. It is applied to the noninverting input of a comparator. The inverting input of the comparator is connected to the output of a DAC. The microprocessor supplies the DAC with a binary input that counts up from zero. When V_{out} exceeds V_{in}, the comparator output switches and generates an interrupt to the microprocessor. The last binary value written to the DAC is proportional to the value of the analog input. The illustration shows that the

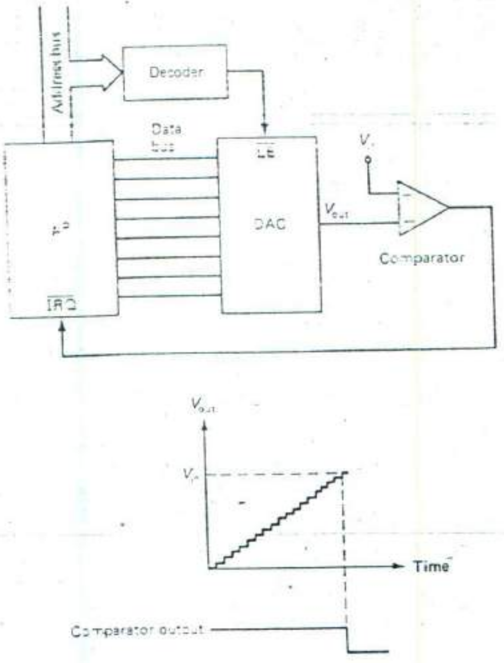


Fig. 13-7 Ramp A/D conversion.

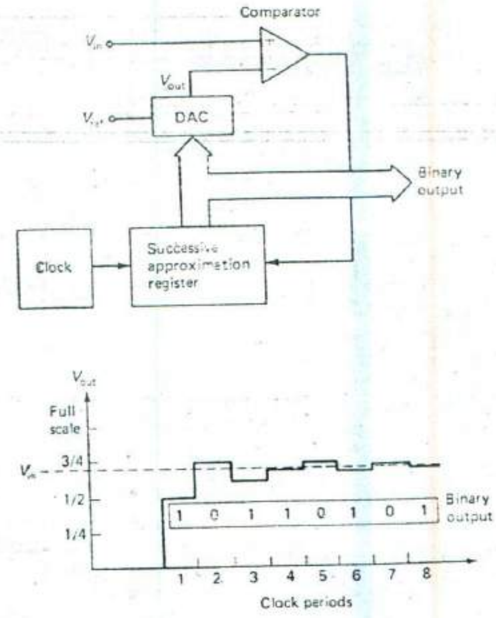


Fig. 13-8 Successive approximation converter.

DAC output is a series of voltage steps called a *staircase waveform*. The resolution of the ramp-type ADC is a function of the number of bits in the DAC.

A ramp converter can also be implemented by replacing the microprocessor with an up counter, a clock, and a logic control circuit. The conversion sequence will begin with the control circuit's resetting the counter to zero. The clock will then be enabled, and up counting will commence. When the comparator output switches, the clock will be disabled, and the counter will then contain the binary value of the analog input. An interrupt can be generated at the completion of the conversion cycle to alert a processor that a binary reading is available. This technique unburdens the processor during the conversion cycle and allows it to perform other functions.

Regardless of how the DAC is supplied with a binary up count, the ramp-type converter is slow. A large analog input will create a significant delay during up counting. It can be as great as 2^N times the DAC and comparator cumulative settling time, where N is the number of bits in the DAC. A tracking converter may provide better performance for some applications. This circuit replaces the up counter with an up/down counter. If the DAC output is less than V_{in}, the comparator causes the counter to count up on the next clock pulse; if the DAC output is more than V_{in}, the counter counts down. The binary value of the counter will track changes in the analog input. The counter output can be sampled at any time a digital value is required. Tracking converters can be used up to low audio rates.

Figure 13-8 shows the *successive approximation converter*, which is much faster. It produces a conversion delay of only N times the cumulative settling time. With an 8-bit DAC, the total delay will be eight clock periods regardless of the value of the analog input signal. The conversion process begins with all DAC inputs set to zero. The MSB bit is set at the first clock pulse. This produces a V_{out} to the comparator equal to one-half the full-scale value. If the comparator switches, the MSB is cleared, and the next most significant bit is set. If the comparator does not switch, the MSB remains set, and the next most significant bit is set. The process repeats N times as each bit is tested. A typical waveform at the DAC output as the conversion proceeds is illustrated by Fig. 13-8. Note that the binary output will show a 0 at every bit position where V_{in} was exceeded and the comparator switched.

The successive approximation technique can be implemented with a clock and a separate register, as shown in Fig. 13-8. It can also be realized without these circuits by interfacing a DAC to a microprocessor. The microprocessor can set each bit and test the output of the comparator. This software approach saves some hardware, but it burdens the processor during the conversion cycle. This is a typical compromise and is known as the *hardware/software tradeoff*.

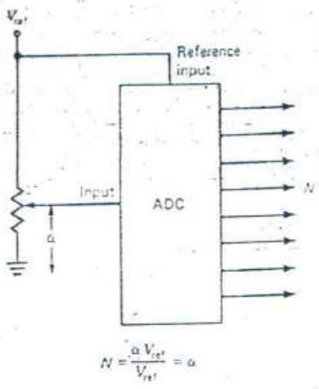
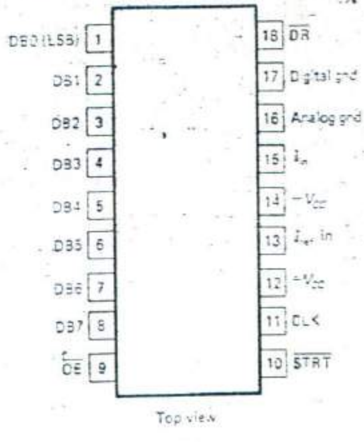


Fig. 13-9 Ratiometric measurement.



(a)

(b)

Fig. 13-10 Integrated circuit NE5034 A/D converter. (a) Pinouts. (b) Block diagram (Signetics).

The external reference voltage input shown in Fig. 13-8 may be used for *ratiometric conversions:* these types of conversions allow an A/D converter to produce an output that represents the ratio of the analog input to the reference input. Using ratiometric conversion, the circuit acts as an analog divider with a binary output. Precision measurements that are insensitive to the reference voltage are possible. Refer to Fig. 13-9: a potentiometer-type transducer is supplied by V_{ref}, which also supplies the reference input to the analog-to-digital converter. The equation shows that the binary count (N) is equal to the potentiometer ratio and is not sensitive to the reference voltage. Ratiometric conversion can also be used with resistive bridges by using an instrumentation amplifier between the bridge and the ADC.

Figure 13-10 shows the pin configuration and block diagram for the Signetics NE5034 monolithic A/D converter. It uses the successive approximation technique and contains a comparator, an 8-bit DAC, a register, a clock, a reference amplifier, and a three-state output buffer. The buffer simplifies interfacing the IC to microprocessors. An external capacitor controls the clock frequency, and conversion times as short as 17 μs are possible. An external clock can be used for even faster conversion times. The IC digitizes an analog input current applied to pin 15. It is capable of ratiometric conversion with an external reference current applied to pin 13. It is also capable of digitizing unipolar or bipolar input voltages, as shown in Fig. 13-11. This illustration also shows the curves for selecting the clock capacitor (C_{CL}).

Another integrated circuit converter, the monolithic CMOS ADC0808 by National Semiconductor, is shown by Fig. 13-12. It uses successive approximation as the conversion technique and provides 8 bits of resolution with a conversion time of 100 μs. It operates from a single 5-V supply and features a low power consumption of only 15 mW. It is microprocessor-compatible with a tri-state output latch/buffer and an address latch and decoder for the input multiplexer. The *input multiplexer* allows any one of 8 analog inputs to be switched to the comparator for conversion to binary. This provides a cost-effective solution for those industrial applications in which the output of several transducers must be digitized. Digitizing eight inputs will take approximately 1 ms, but this is an adequate response time for many applications in industry, such as temperature monitoring.

An A/D conversion can be accomplished without a DAC. Figure 13-13 (p. 362) shows a voltage-to-frequency converter. With a negative V_{in} applied, the integrator output will ramp in a positive direction until the V_{ref} threshold is crossed. At this time, the comparator output will switch and discharge the integrator. The next positive ramp will begin, and the cycle will repeat over and over. The magnitude of V_{in} will set the slope of the positive ramps. Larger

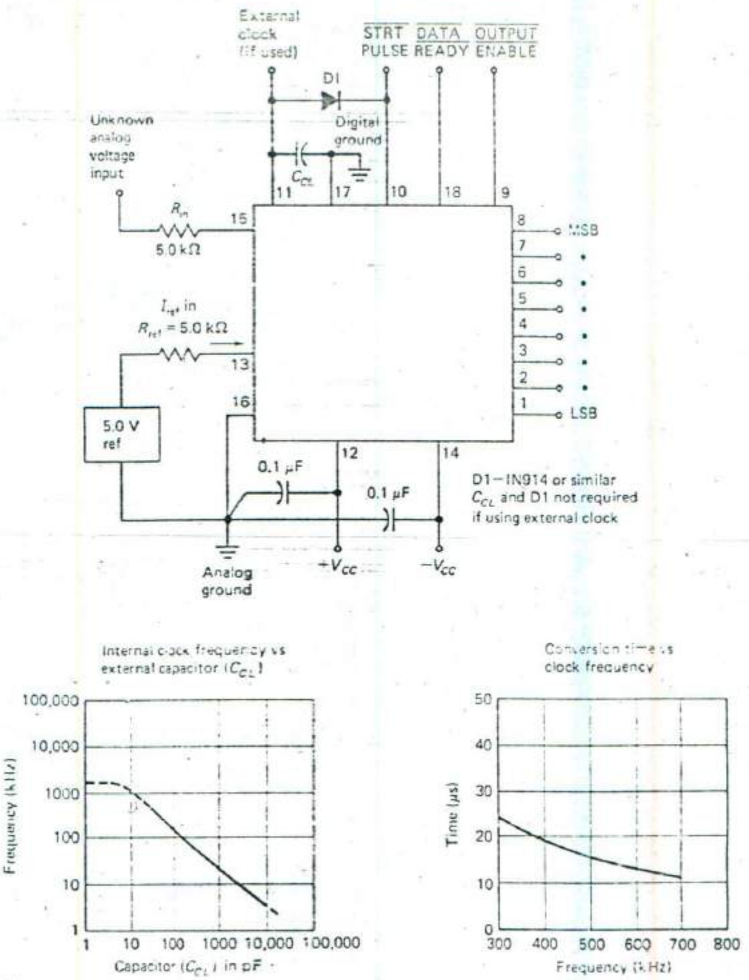
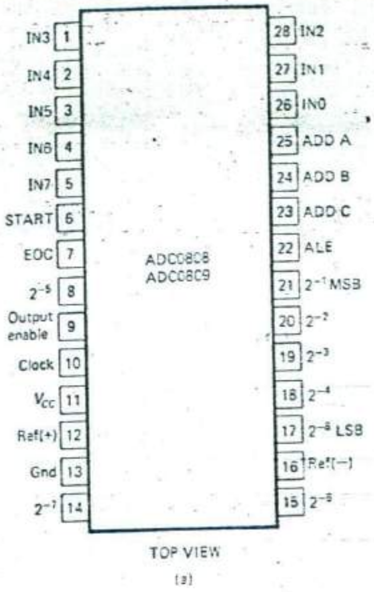


Fig. 13-11 Applying the NE5034 (Signetics).

input voltages will produce a steeper slope, and the comparator threshold will be reached in less time. The pulse frequency at the output of the comparator is proportional to the input voltage. Figure 13-13 adds a clock circuit, a gate, and a counter to convert the frequency to a binary count proportional to the analog input voltage. The performance of voltage-to-frequency A/D converters is relatively poor for small input voltages, as a result of offset drift in the integrator and comparator.

Integration is also used in the *dual-slope circuit* of Fig. 13-14. The conversion cycle begins with $-V_{in}$ applied to the integrator through the switch for a fixed time T_1. During T_1, the integrator output will rise to a point dependent on the input signal magnitude. Then the input switch is changed, and a reference voltage of opposite polarity is applied to the integrator. The counter will count clock pulses during the time (T_2) it takes the integrator to ramp back down to the comparator threshold. Since V_{ref} and T_1 are fixed known quantities, V_{in} is proportional to T_2. The dual-slope technique has the advantage of minimizing errors because the same circuit and components are used for summing both the reference and unknown input voltages. Any long-term component drift is canceled. However, the circuit is sensitive to zero offset and drift in the integrator and comparator. These effects can be compensated for by a more complex circuit that adds calibration phases to measure the analog error and eliminate it from the final output. Some error-correcting techniques allow as much as 14 bits of resolution, but integration is a slow technique, and the conversion time is too long for some applications. Dual-slope A/D converters are typically used in instruments such as digital voltmeters (DVMs) and digital multimeters (DMMs).

The A/D converter response time is critical for some applications. For example, it may be necessary



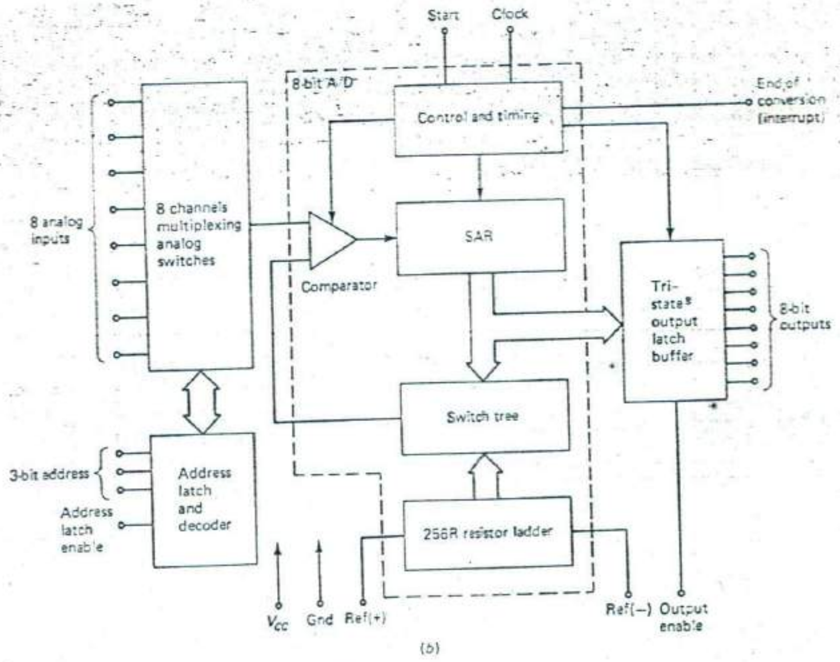


Fig. 13-12 Integrated circuit ADC0808 A/D converter. (a) Pinouts. (b) Block diagram (National Semiconductor).

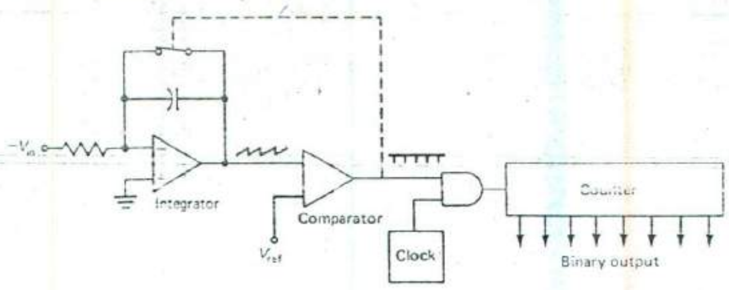


Fig. 13-13 Voltage-to-frequency converter.

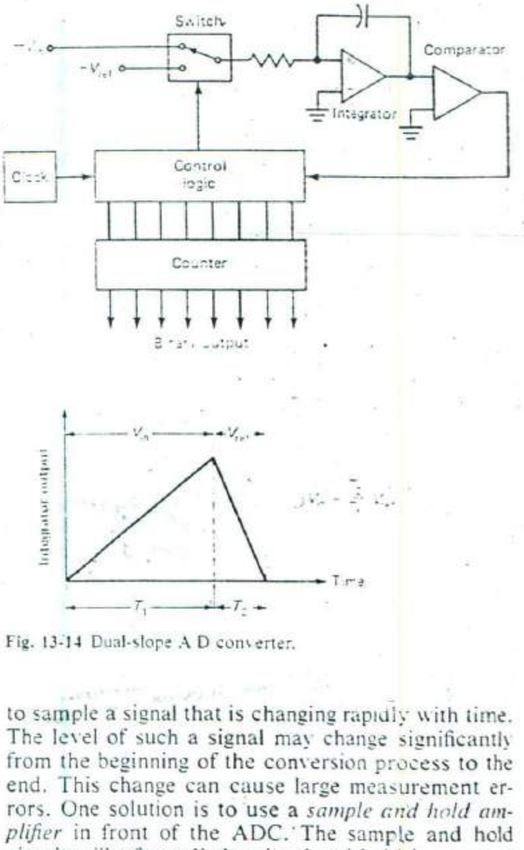


Fig. 13-14 Dual-slope A/D converter.

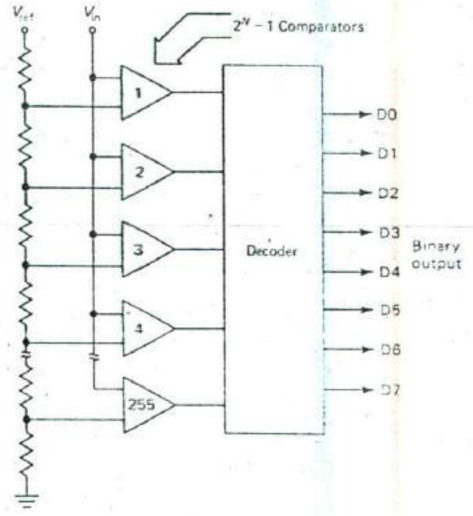


Fig. 13-15 Flash converter.

to sample a signal that is changing rapidly with time. The level of such a signal may change significantly from the beginning of the conversion process to the end. This change can cause large measurement errors. One solution is to use a *sample and hold amplifier* in front of the ADC. The sample and hold circuit will "freeze" the signal and hold it constant during the conversion period. Sample and hold amplifiers are discussed in Chapter 6. Another solution is to use a converter that is very fast. One example is the flash converter shown in Fig. 13-15. It requires $2^N - 1$ comparators for N bits of resolution. An 8-bit converter requires 255 comparators stacked at voltage levels of 1 part in 256 from each other. This stacking is accomplished by a resistive divider. The comparator array compares the input signal with each reference voltage to produce an N of 255 code sometimes referred to as the "thermometer" code since all comparators below the signal level will be on and all those above will be off. The decoder logic converts this code to a binary output.

Flash converters are very fast. A typical 8-bit device can respond in 50 ns. This allows up to 20 million conversions per second. The disadvantages are high cost and a low input impedance caused by the input signal's driving many comparators in parallel. Figure 13-16 shows a compromise circuit, the *dual-stage flash converter;* it may also be referred to as a *half-flash converter.* It requires only 30 comparators to achieve 8 bits of resolution. The tradeoff is speed; it is a little slower than the single-stage circuit. The first conversion is done with the 4-bit flash circuit at the left. Its output is simultaneously fed to the register and the input of a 4-bit DAC. The output of the DAC is applied to the inverting input of a subtracting amplifier, and V_{in} is applied to the noninverting input. Because V_{in} is delayed, the two amplifier inputs arrive in the proper time relationship. The difference is then converted in a second 4-bit flash circuit at the

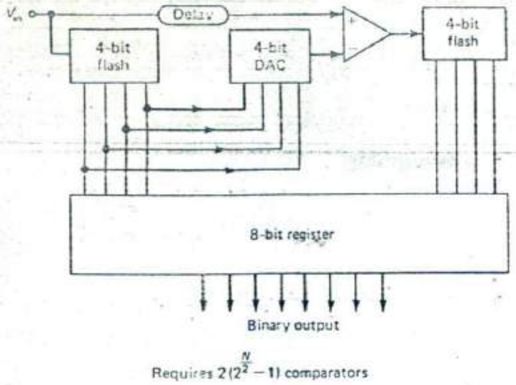


Fig. 13-16 Dual-stage flash converter.

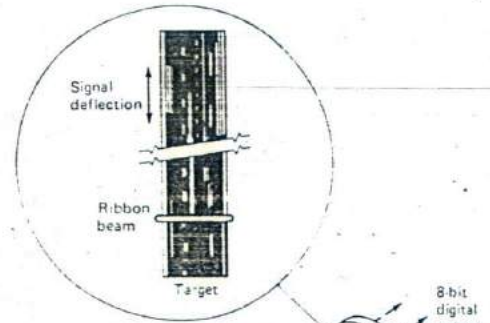


Fig. 13-17 EBS A/D converter.

right. The final result is now available in the 8-bit register as a complete binary word.

One of the fastest A/D converters is the *electron-bombarded semiconductor* (EBS) shown in Fig. 13-17. It uses an electron beam that is focused into a narrow ribbon and deflected across a masked diode target. The analog input signal is applied to the vertical deflection plates, where the beam is deflected vertically along the target in accordance with the magnitude of the input signal. The target is Gray-coded with eight columns of diodes and eight corresponding high-speed comparators. The effects of beam jitter are reduced by the Gray code since only one diode switches at a time from any target position to the next target position. The EBS converter can sample at rates up to 200 MHz (5 ns) but is very expensive.

REVIEW QUESTIONS

5. A ramp-type A/D converter generates a ________ waveform at its DAC output.

6. Assume a cumulative settling time of 2 μs for the DAC and comparator in an 8-bit ramp-type converter and calculate the worst-case conversion time.

7. Assume a cumulative settling time of 2 μs for the DAC and comparator in an 8-bit successive approximation converter and calculate the conversion time.

8. Refer to Fig. 13-9. What effect will a 5 percent change in the reference voltage have on the binary output?

9. What is selected by the 3-bit address applied to the IC shown in Fig. 13-12?

10. Refer to Fig. 13-13. What happens to the pulse frequency at the comparator output as the analog input voltage decreases? What happens at the binary output?

13-3 COMMUNICATION STANDARDS

One of the most popular digital communication standards is RS-232C, which is introduced in Chapter 12. It is an old standard and is being superseded in some cases by new standards. The RS-232C has some limitations that have been improved by these newer standards. One of the major ones is that it is limited to distances of 15.2 m (50 ft). In practice, much longer distances are actually covered. However, the chance for data errors does increase as the distance increases. Another limitation is a maximum rate of 20,000 Bd with 19,200 Bd being the fastest standard data rate. A combination of long distance and high data rate increases the chances for errors. The RS-232C standard sends data on a single wire. A single ground wire serves as the return or reference for one or more data circuits in the cable. This type of transmission is called *unbalanced*. Figure 13-18 shows how the recommended baud rate drops as the line length increases in unbalanced circuits. The RS-232C standards are somewhat conservative when compared to the graph.

The RS-232C uses ground as its reference. When the line receiver sees a voltage more negative than 3 V referenced to ground, *mark* is produced. When the receiver sees a voltage more positive than 3 V referenced to ground, *space* is produced. Various conditions can cause the ground potential to vary from point to point. A difference in ground potential between the line transmitter and the line receiver will

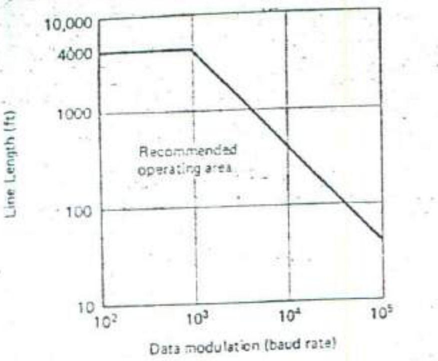


Fig. 13-18 Line length versus baud rate for unbalanced transmission lines.

cause ground loop currents to flow in the ground wire. On long runs, the wire resistance can allow significantly different ground potentials at each end of the circuit, and data errors are possible. The protective ground, if used, helps but does not eliminate this problem.

One obvious way to eliminate ground potential errors is to send the data on two wires and to ground neither of them. This is known as *balanced* or *differential data transmission* and provides several important advantages over unbalanced transmission. A balanced line is not nearly so susceptible to stray fields and noise pick-up as an unbalanced line. Balanced lines generate less noise, an advantage when several data circuits are bundled together in one cable. Balanced transmission also allows much higher data rates, as shown in Fig. 13-19. Note that data rates as high as 10 megabaud (MBd) are possible. Even at 1219 m (4000 ft), data rates as high as 100 kBd are recommended. This is a very significant speed improvement over unbalanced circuits. The Electronic Industries Association (EIA) has established RS-422 as a balanced data transmission standard. It specifies a bipolar driver output of 2 to 6 V and a receiver sensitivity of 200 mV. If the receiver sees a positive difference signal between the two wires of more than 200 mV, a mark is read; if the difference signal is negative and more than 200 mV, a space is read. This allows suitable transmitters and receivers to be designed with bipolar 5-V supplies; RS-232C, however, often requires bipolar 12-V supplies.

The EIA has also established RS-423 as a data transmission standard. This standard resembles RS-232C in that it too is an unbalanced standard. It is capable of higher data rates (up to 100 kBd at 12.2 m) and uses different voltage levels. You may recall that RS-232C transmitters develop from −5 to −15 V on mark and from +5 to +15 V on space. The RS-423 transmitters use −4 to −6 V for mark and +4 to +6 V for space; they also use a balanced receiver, which is referenced to the driver ground to permit ground potential differences to exist without causing data errors. The RS-423 can use one wire in a cable to serve as a common return for several data circuits and is therefore more "wire-efficient" than RS-422, although not nearly as fast.

Figure 13-20 shows the DS3691 line driver IC, which can be used for RS-422 or RS-423 transmission. Pin 4 is the mode select and determines which standard will be used. The RS-423 operation provides four communications circuits, but RS-422 operation provides only two. The truth table shows that pins 3 and 6 can be used to tri-state the outputs with RS-422 operation, whereas they serve as data inputs for RS-423 operation. The IC is powered with a positive voltage applied to pin 1 and a negative voltage applied to pin 8. The bipolar supply can be as much as 7 V, but 5 V is typical. Capacitors can be connected to the rise time control pins to limit the output slew rate. This connection is necessary in some applications to eliminate cross-talk in multicircuit cables. In *cross-talk* one communication circuit interferes with another; it is more likely to occur when the waveforms have very fast rise times.

Figure 13-21 shows a DS78LS120 *dual differential line receiver* that can be used to receive both RS-422 and RS-423 data transmissions. The response time of each receiver can be controlled with an external capacitor. This is useful for filtering out high-frequency noise that may be superimposed on the signal. Also note that the output gate has hysteresis. The combination of response control and hysteresis is very helpful for producing a clean output signal in noisy industrial environments. Each receiver includes a built-in terminating resistor that is enabled by strapping pins 2 and 3 together for one receiver and pins 13 and 14 for the other. It is advisable to terminate long transmission lines to prevent signal reflection. When a line is not properly terminated, all of the signal energy is not absorbed at the load end. Some is reflected back toward the source. It can be rereflected at the source and travel toward the load again. These reflected signals act as noise

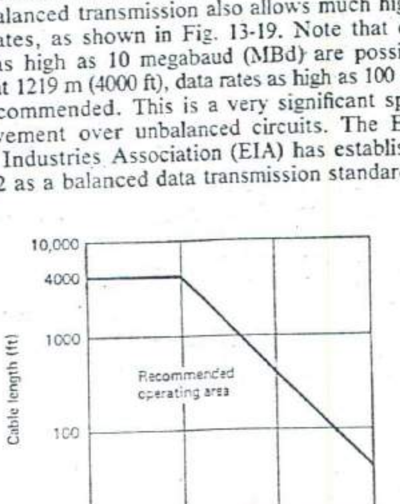
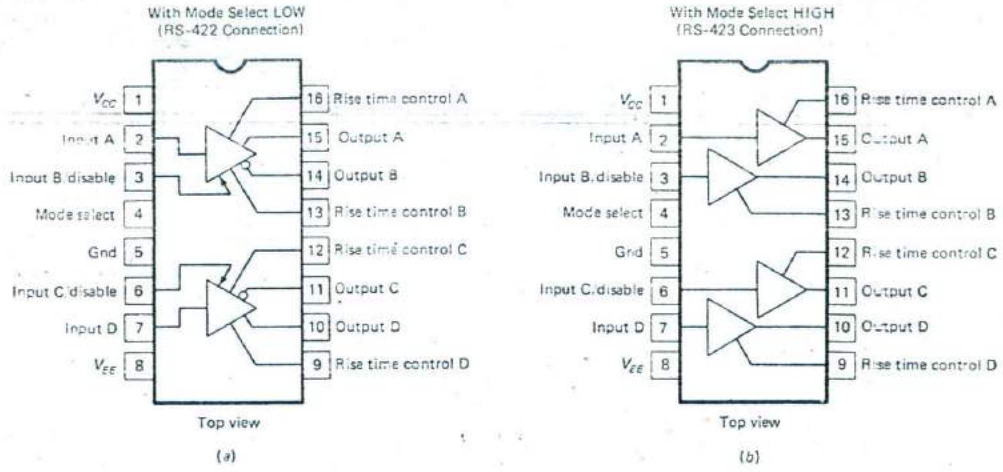


Fig. 13-19 Line length versus baud rate for balanced transmission lines.



Operation	Inputs			Outputs	
	Mode	A (D)	B (C)	A (D)	B (C)
RS-422	0	0	0	0	1
	0	0	1	Tri-state	Tri-state
	0	1	0	1	0
	0	1	1	Tri-state	Tri-state
RS-423	1	0	0	0	0
	1	0	1	0	1
	1	1	0	1	0
	1	1	1	1	1

(c)

Fig. 13-20 Integrated circuit DS3691 line driver. (*a*) With mode select LOW. (*b*) With mode select HIGH. (*c*) Truth table (National Semiconductor).

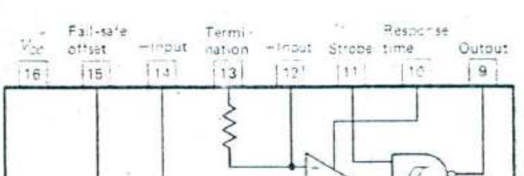


Fig. 13-21 Integrated circuit DS78LS120 dual differential line receiver (National Semiconductor).

and may cause data errors. Sometimes a capacitor is used in series with the terminating resistor to eliminate dc power loss in the resistor.

Figure 13-21 also shows that each line receiver contains a fail-safe offset input. *Fail-safe operation* involves forcing the output to a known condition in the event that the data circuit opens or short-circuits. When pins 1 and 15 are connected to +5 V, the input thresholds are offset from 200 to 700 mV, referred to the noninverting input, and from −200 to −700 mV when referred to the inverting input. If the input should open or short-circuit, the input will be greater than the input threshold, and the receiver output will remain in a specific logic state.

Figure 13-22 gives examples of both balanced and unbalanced data transmission circuits using the DS3691 line driver and the DS78LS120 line receiver. The balanced wires are shown as a twisted pair. This is standard practice for balanced transmission, and data cables are available with the wires twisted into pairs. The offset inputs of the line receivers are

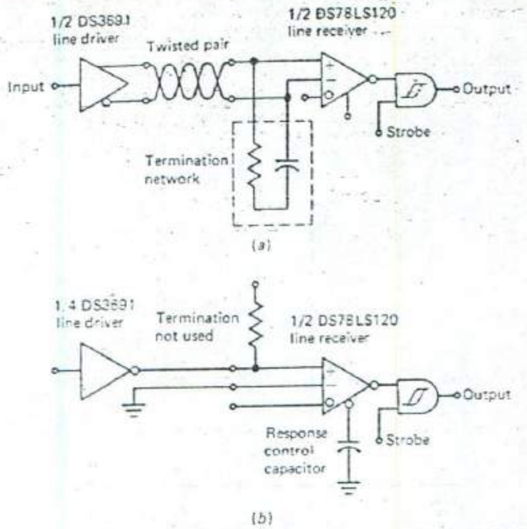


Fig. 13-22 Balanced and unbalanced data transmission. (*a*) RS-422A balanced data transmission. (*b*) RS-423A unbalanced data transmission.

shown floating as the fail-safe feature is not implemented in these examples.

The RS-422 and RS-423 standards involve electrical specifications only. The EIA has also introduced RS-449 as a possible successor to RS-232C. This standard includes a description of the signals needed for modem control and the mechanical specifications for the connectors. The RS-449 specifies a total of 46 wires and uses two separate connectors; one with 37 pins and the other with 9 pins. The 9-pin connector carries all signals associated with a secondary channel. Most applications do not have the secondary channel, and the 9-pin connector is eliminated in these cases.

All of the data communication standards discussed to this point are *bit serial*. Another technique is *bit parallel*, which allows 8, 16, or 32 bits to be transferred at a time. One example is the Institute of Electrical and Electronic Engineers (IEEE) 488 standard, which is a bit-parallel, byte-serial system. This standard was originally proposed by Hewlett-Packard, Inc., as a general purpose interface bus for instruments, calculators, and computers. It is still often referred to as the *Hewlett-Packard Interface Bus* (HPIB) or as the *general purpose interface bus* (GPIB). It allows building a complex system of devices to be as simple as connecting cables to the various parts. However, software compatibility is not guaranteed by the IEEE 488 standard. This standard specifies signal voltages, currents, and signal timing. The work of Hewlett-Packard goes a bit beyond the IEEE standard in that it does specify software communication techniques.

Figure 13-23 shows the basic configuration of the IEEE 488 bus. It can accommodate up to 15 devices, which might include items such as computers, digital multimeters, printers, plotters, calculators, signal generators, counters, and disk drives. It is designed for limited distances (up to 20 m) and data rates up to 1 megabyte per second. The signals are TTL-compatible, and 16 active lines transfer data and commands. Eight of the lines are used for data transfers, which are asynchronous and are coordinated by handshaking lines. *Handshaking signals* are used to prevent one device from sending data faster than another device can receive it. The handshaking is accomplished by the data byte transfer control lines shown in Fig. 13-23. The remaining five lines are for the control of bus activity and are shown as the general interface management group in the illustration.

The devices connected to the bus are identified as talkers, listeners, or controllers. A *talker* originates information, and a *listener* receives it. Some devices

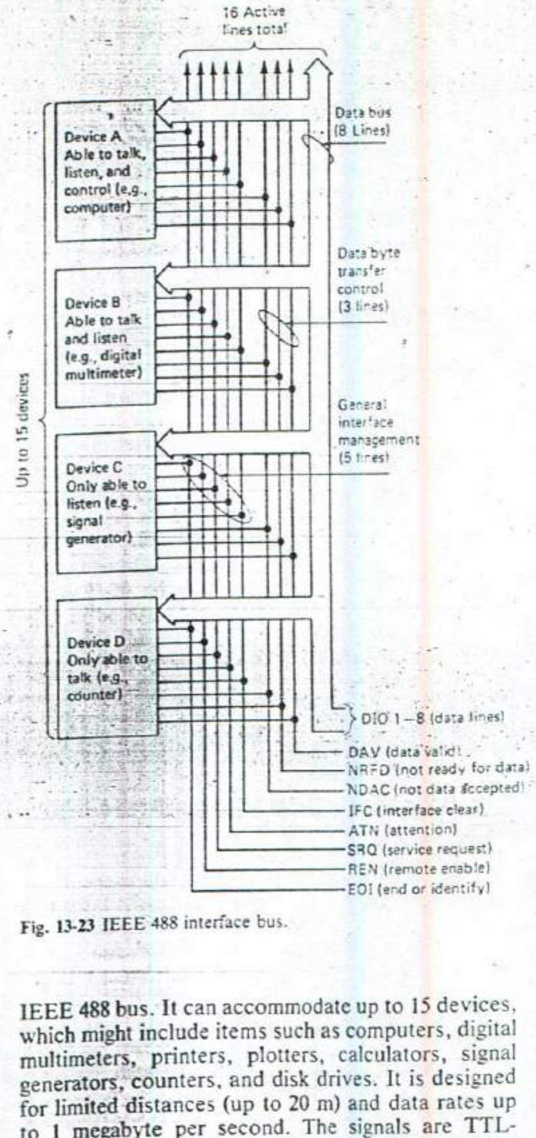

16 Active lines total
Data bus (8 Lines)
Device A Able to talk, listen, and control (e.g., computer)
Data byte transfer control (3 lines)
Device B Able to talk and listen (e.g., digital multimeter)
General interface management (5 lines)
Up to 15 devices
Device C Only able to listen (e.g., signal generator)
Device D Only able to talk (e.g., counter)
DIO 1–8 (data lines)
DAV (data valid)
NRFD (not ready for data)
NDAC (not data accepted)
IFC (interface clear)
ATN (attention)
SRQ (service request)
REN (remote enable)
EOI (end or identify)

Fig. 13-23 IEEE 488 interface bus.

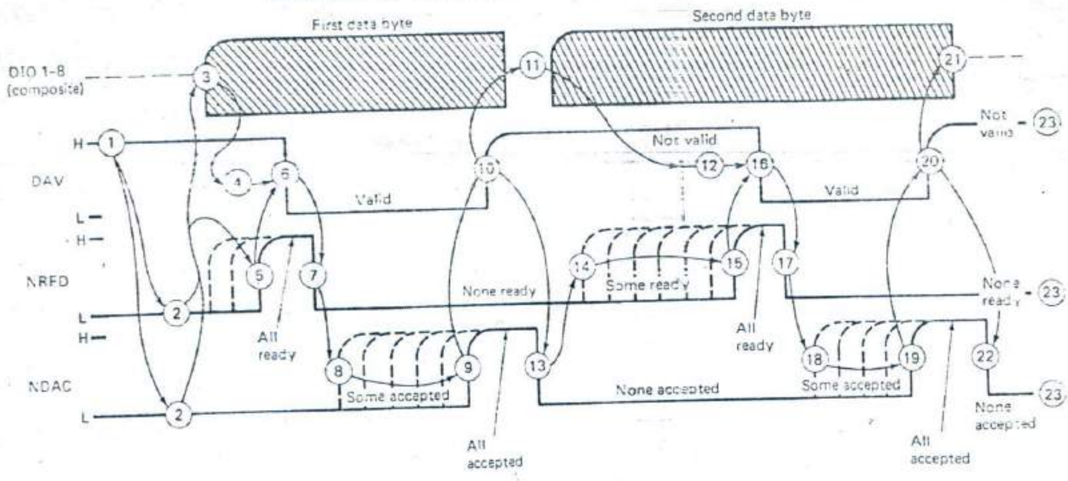


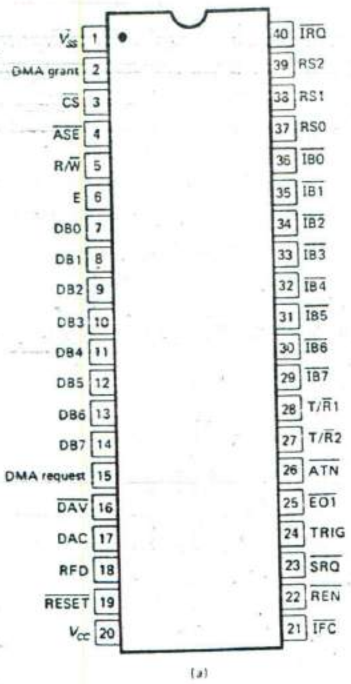
Fig. 13-24 HPIB three-wire handshaking.

may be both. A computer is usually a talker, a listener, and a controller. *Controllers* dictate the role of each device by asserting the *attention* (ATN) line and sending talk or listen addresses on the data lines. Addresses must be programmed into each device at the time of system configuration. This is usually done with switches on the back of the instrument or with jumpers on an internal PC board. When ATN is asserted, all devices must listen to the data lines. When ATN is false, only those devices that have been addressed will send or receive data, and all other devices ignore the data lines.

Several listeners can be active at a time, but only one talker may be active at any given time. When a talk address is placed on the data bus with ATN asserted, all other talkers are automatically deselected. The active talker controls the *data valid* (DAV) line. The active listener or listeners control *not ready for data* (NRFD) and *not data accepted* (NDAC). The various listeners may not all accept data at the same rate. The transfer speed is set by the slowest active listener on the bus. This is controlled by an open collector voting system. All the active listeners must agree before the transfer is allowed. The NRFD line is pulled to +5 V with a resistor. Any device that is not ready will ground this line through the collector of a transistor. All of the collectors must go high before NRFD can go high. When the talker sees NRFD go high, it places a byte on the data bus and waits for a 2-μs settling period. It then asserts DAV by pulling it low. This alerts the listeners to read the bus. They acknowledge by pulling NRFD low. A timing diagram that details the sequence involved in the handshaking process is given in Fig. 13-24.

When ATN is true, addresses and universal commands are transmitted on seven of the data lines. The ASCII code is used for commands. Any active talker will relinquish control of the bus when this happens by releasing the DAV line. The controller now acts as the talker, and all other devices accept the information. This allows the controller to configure the bus and assign the roles of talker and listener to the various devices on the bus. More than one controller is allowed, but only one may be active at a time. The main system controller is activated at the time of power-on. Any other controllers remain passive until control is passed to them. Once the controller has configured the bus, it takes no part in subsequent data transfers until reconfiguration is required.

In addition to ATN, there are four other control lines. *Interface clear* (IFC) places the interface in a quiescent state. *Remote enable* (REN) is used with other coded messages to select either local or remote control of each device. Any active device can assert the *service request* (SRQ) line, which tells the controller that some device on the bus needs attention. *End or identity* (EOI) is used by a device to indicate the end of a multiple byte transfer. A second use for EOI is when the controller asserts ATN and EOI at the same time to signal that a parallel poll is requested. This alerts any device capable of a parallel poll to indicate its current status on the data line assigned to it. Up to eight devices can respond. This is a fast way for the controller to determine which device has requested service. The controller can also use a *serial polling routine*, in which it addresses one device as a talker and then sends that device a serial-poll enable command. The serial poll allows the controller to obtain 8 bits of information from a single device. The controller then sends a serial poll disable command to return the device to the data mode. The advantage of serial polling is that much more infor-



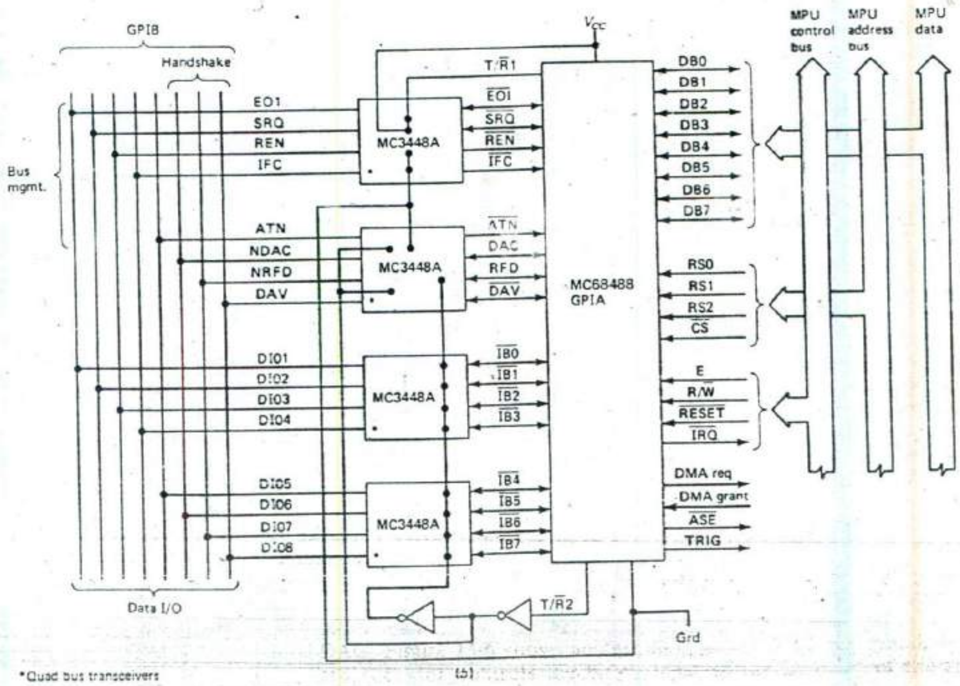


Fig. 13-25 Integrated circuit MC68488 general purpose interface adapter. (*a*) Pinouts. (*b*) IEEE interface (Motorola).

mation can be obtained from each device polled. The disadvantage is the time required to poll every device.

Figure 13-25 shows the Motorola MC68488 *general purpose interface adapter* (GPIA) designed to interface microprocessors such as the MC6809 to the IEEE 488 bus. The basic device is rated for a clock rate of 1 MHz. Motorola also makes a 1.5-MHz version (MC68A488) and a 2-MHz version (MC68B488). This IC greatly simplifies the interface. Many bus functions are handled automatically by the GPIA and require no action from the processor. Other functions require minimum processor attention because of the large number of internal registers that convey information on the state of the GPIA and of the bus. It features address recognition, full support of the handshake protocol, programmable interrupts, serial and parallel polling capability, talk-only or listen-only capability, and compatibility with the microprocessor bus.

REVIEW QUESTIONS

11. Data communications that occur on a single wire referenced to ground or to a return wire are a form of _______ transmission.

12. Differential communication circuits that use two wires are a form of _______ transmission.

13. RS-232C is an example of a(n) _______ transmission circuit.

14. Refer to Fig. 13-21. What does the symbol inside the NAND gates signify?

15. Refer to Fig. 13-23. What is different about the ground reference technique used in RS-423 as compared to that in RS-232C?

16. Refer to Fig. 13-23. What is the combined function of DAV, NRFD, and NDAC?

17. Refer to Fig. 13-24. From what time (circled number) to what other time is the first byte being read by the slowest listener on the bus?

13-4 DISK STORAGE

Magnetic disk storage has become a very important part of many computer systems. Disk drives are used on large mainframe computers, minicomputers, and microcomputers. This section will deal mainly with the types of disk systems used with minicomputers and microcomputers. The industrial technician is less likely to be directly involved with the high-capacity rigid disk systems used with large mainframe computers.

Magnetic disks provide a way to store programs and data. They also serve as an extension of computer memory. The ideal situation is to have all necessary information in computer memory. Since memory is random access, the best operating speed is provided by this situation. However, it is often not practical to provide enough memory to hold all of the programs and data that the computer will ever need. Disk memory is block access. It is slower than RAM, but it is fast enough for many applications.

Magnetic disks are made from aluminum or Mylar. They are coated with a magnetic material such as ferric oxide. Binary information is stored in the magnetic coating by orienting the domains one way or the other. This is done by passing a current through a write head, as shown in Fig. 13-26. The current is reversed to change the direction of magnetization. Figure 13-27 shows two of the ways that binary data can be recorded magnetically. Figure 13-27(*a*) shows the basic *nonreturn to zero* (NRZ) method. The cur-

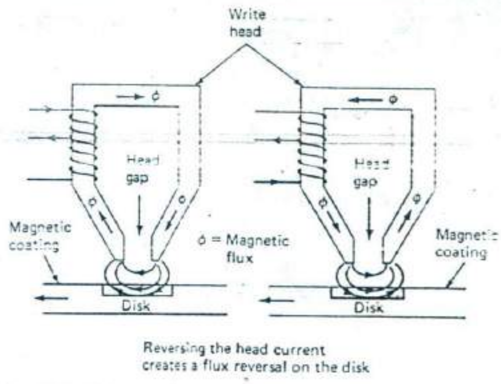


Fig. 13-26 Disk recording.

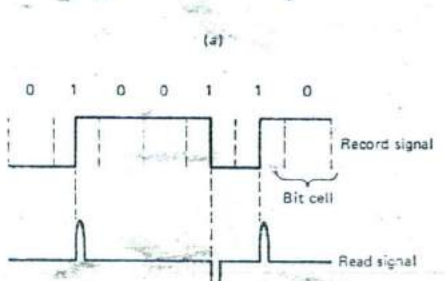


Fig. 13-27 Magnetic recording techniques. (*a*) Basic NRZ recording. (*b*) NRZI recording.

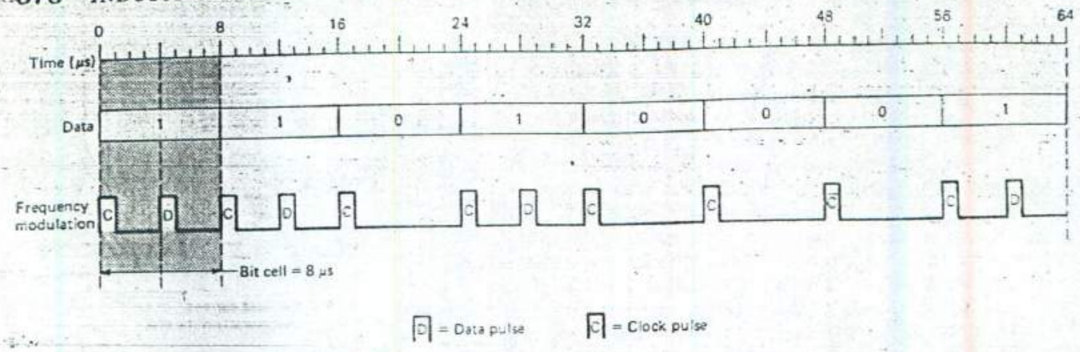


Fig. 13-28 Frequency-modulated (FM) encoding.

rent in the head remains constant for consecutive 1s or 0s. It reverses on transitions from 0 to 1 or 1 to 0. The magnetically stored information is read back by exposing a read head to the magnetic patterns on the disk. One head is usually used for both reading and writing and is called the *read/write head*. Note that the read pulses are positive for 0 to 1 transitions and are negative for 1 to 0 transitions for the NRZ recording technique. Figure 13-27(*b*) shows the *nonreturn to zero inverted* (NRZI) recording method. This method reverses the direction of the head current for every binary 1 that is recorded. The head current remains constant for 0s. Note that a series of 1s will show a flux change for each bit. A read pulse will occur only where a 1 has been recorded. Whether the read pulse is positive or negative is immaterial when the NRZI recording technique is used. Information is read back from the disk by interpreting any flux change as a 1 and the absence of change within a predefined interval called a *bit cell* as a 0. The NRZI is the standard recording technique used in all disk systems.

In addition to recording techniques, there are encoding techniques. Do not confuse the two. The *encoding* schemes affect the patterns of flux changes placed on the disk; the *recording* technique is always NRZI. A read circuit called a *data separator* converts the recorded patterns into meaningful information. Frequency modulation (FM) is an early encoding scheme used in disk systems and is illustrated in Fig. 13-28. A clock bit is written at the beginning of each bit cell, and data bits are written between clock bits. A typical bit cell might be 8 μs wide, and the data bit is written 4 μs after the clock bit. The data bits are 8 μs apart, and the data transfer rate can be found by taking the reciprocal of this period:

$$\text{Data rate} = \frac{1}{8 \times 10^{-6}} = 125{,}000 \text{ bits/s}$$

The constant bit cell reference makes encoding and decoding simple in FM systems. The data separator synchronizes on the clock bits and generates a 4-μs window beginning 2 μs after the clock pulse. Straightforward timing circuits are used to generate the window.

The problem with FM encoding is that the clock bits represent a significant overhead. They take half the available space on the disk and provide timing information only. *Modified FM* (MFM) doubles the available space by replacing clock bits with data bits: MFM is shown in Fig. 13-29. This encoding technique writes clock bits only if data bits have not been written into the preceding and present bit cells. The clock bits are placed at the beginning of the cells and the data bits in the center, just as in FM. This arrangement cuts the bit cell time to 4 μs. The capacity of the disk is doubled, as is the data transfer rate. However, timing tolerances are cut in half. The data windows are only 2 μs wide, necessitating a more sophisticated data separator. The data stream must be continuously analyzed to keep the windows synchronized with the correct bits. A *phase-locked loop circuit* is used to lock a local clock to the data stream for decoding purposes.

The ultimate encoding system would eliminate clock pulses altogether. The FM and MFM must have clock pulses. Otherwise, a string of 0s would

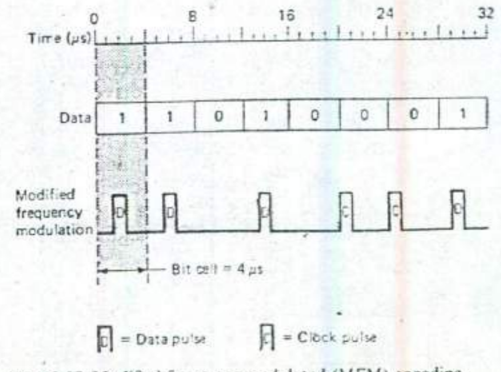
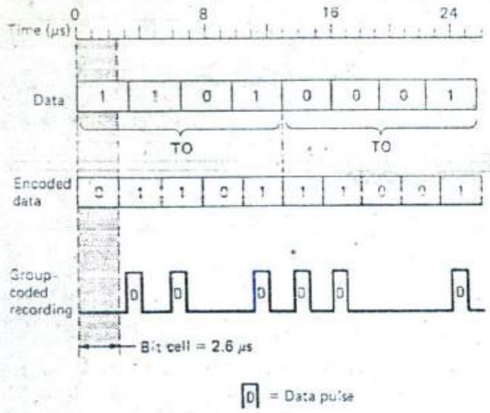


Fig. 13-29 Modified frequency-modulated (MFM) encoding.



Translating Chart for Group-Coded Recording		
Hexadecimal number	4-bit nibble	GCR code
0	0000	11001
1	0001	11011
2	0010	10010
3	0011	10011
4	0100	11101
5	0101	10101
6	0110	10110
7	0111	10111
8	1000	11010
9	1001	01001
A	1010	01010
B	1011	01011
C	1100	11110
D	1101	01101
E	1110	01110
F	1111	01111

Fig. 13-30 Group coded recording (GCR) encoding.

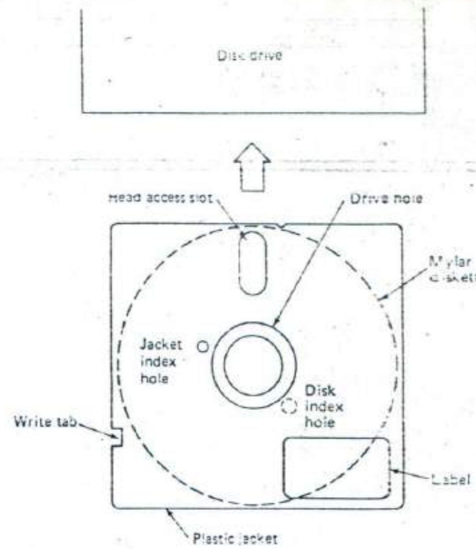


Fig. 13-31 Floppy diskette.

produce a section on the disk with no flux changes, and the data separator would not know the length of the string. *Group coded recording* (GCR) overcomes this limitation by dividing each incoming byte to be recorded into 2 nibbles (a *nibble* is a 4-bit word). The nibbles are then translated into 5-bit binary codes. Out of a total of 32 possible codes, only 16 are used, and they are chosen to meet two important requirements: (1) there can never be more than two consecutive 0s, and (2) no combination of codes can produce more than two consecutive 0s on the disk. Figure 13-30 shows the translation scheme. Inspect the column of GCR codes. You will find that no code contains more than two consecutive zeros, and neither does any combination of two codes. This allows the read clock to synchronize on data bits alone. The read circuit converts the 5-bit codes back into the original nibbles. The nibbles are rejoined to form a byte for return to the computer. Use of GCR encoding provides approximately a 150 percent improvement in disk capacity and data transfer rate when compared to MFM.

Figure 13-31 shows the appearance of a typical floppy diskette. *Floppy* describes the characteristic that the actual recording medium is coated Mylar and is not rigid as it is in the large mainframe drives. *Diskette* signifies that the size is 5.25 in., rather than 8 in. The original floppy drives used 8-in. disks; however the smaller drives and diskettes have become very popular. The prefix *mini* is another way to signify that the size is 5.25, rather than 8 in. A plastic jacket protects the Mylar disk. There are several holes in the jacket. One provides access for the head or heads, another allows the drive hub to contact the center hole of the Mylar so the disk can be turned in the jacket, and another allows the index hole to be sensed as the disk is turning. *Indexing* provides a starting point for read and write operations. There is also a write protect cutout along the side of the jacket. The cutout is covered with a gummed tab to write-enable 8-in. disks. Just the opposite is true with 5.25-in. diskettes: the tab is applied to write-protect the disk.

Floppies must be handled with care. They should be stored vertically in protective envelopes. They should not be exposed to high temperatures and direct sunlight. Their storage and operating range is 10 to 50°C. Use only felt tip pens when writing on the labels. Never touch the disk surface through the head access slot. Do not attempt to clean them. The inside of the plastic jacket is lined with a special pad material that cleans the disk surfaces as it rotates inside the jacket. Never bend or fold them and never expose them to magnetic fields, dust, dirt, or smoke.

Figure 13-32 shows how one type of head-positioning mechanism in floppy disk drives works. A stepper motor turns a lead screw. The lead screw nut

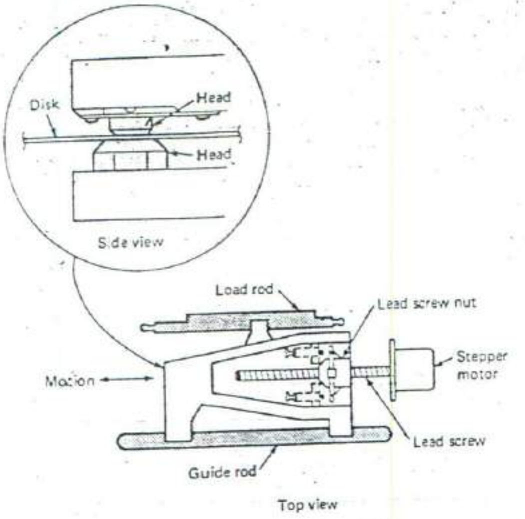


Fig. 13-32 Floppy disk head-positioning mechanism.

is mounted on the head mechanism, which can slide back and forth on the guide and load rods. The enlarged side view shows the disk pinched between the heads. This happens when the disk is being written or read and is accomplished with a head-load solenoid. Some drives are *single-sided*, indicating that they record on one side of the disk only. In these drives, the top head is replaced with a pressure pad. Although not shown in the illustration, a drive spindle and collett assembly engage the center hole of the Mylar disk. Another motor turns the disk at 300 rpm. The information is recorded in tracks as the disk turns.

Figure 13-33 shows a typical disk format. There are 40 concentric tracks in all. The stepper motor moves the head toward the center or away from the center to record or read the various tracks. The track-to-track access time is typically around 25 ms. The average access time to reach a given track is around 400 ms. Thus, you can see that this block access device is reasonably fast, but not nearly so fast as RAM. The tracks are divided into 18 sectors. Each sector can hold 256 bytes of data plus approximately 82 control bytes. The control bytes are used for synchronizing, identifying the track and sector number, and checking for errors. Since there are 18 sectors with 256 data bytes each, the track capacity is 4608 bytes. The total disk capacity is 184,320 bytes. However, not all of this is available as user storage. The center track is used to store the directory of the disk. The *directory* is a listing of all of the files stored on a disk, their size, their type, and their location (track and sector). Since the directory is frequently accessed, placing it at the center of the disk speeds operation.

The original 8-in. floppy disks used FM encoding and stored 250K bytes per side. *Double-density drives* using MFM encoding for a capacity of 500K bytes per side were then developed. The latest 8-in. drives store 800K bytes per side. The original 5.25-in. mini floppy diskettes used FM and provided 90K bytes per side. The change to MFM provided double density and 180K bytes per side. Recent developments, such as GCR encoding and higher track/sector densities, have produced capacities up to 670K bytes per side on 5.25-in. diskettes.

Another development is the microfloppy disk shown in Fig. 13-34. This floppy features smaller size and a more rugged structure. The plastic case is much more rigid than the plastic jackets used to protect the larger floppies. It uses a metal shutter

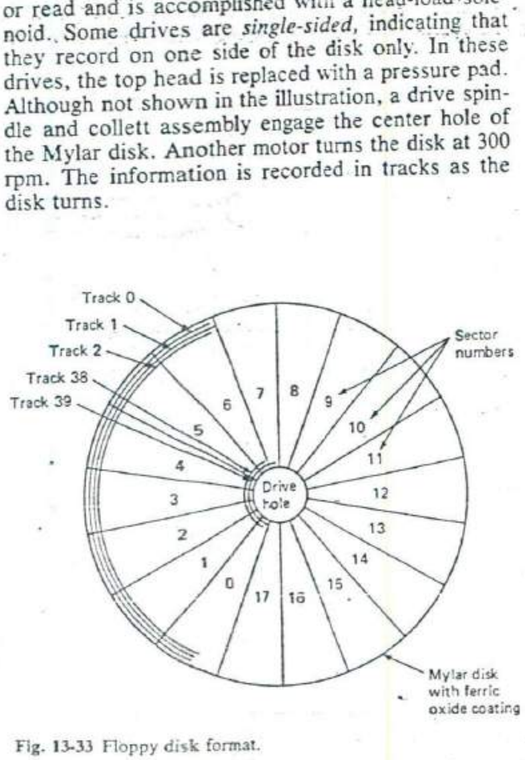


Fig. 13-33 Floppy disk format.

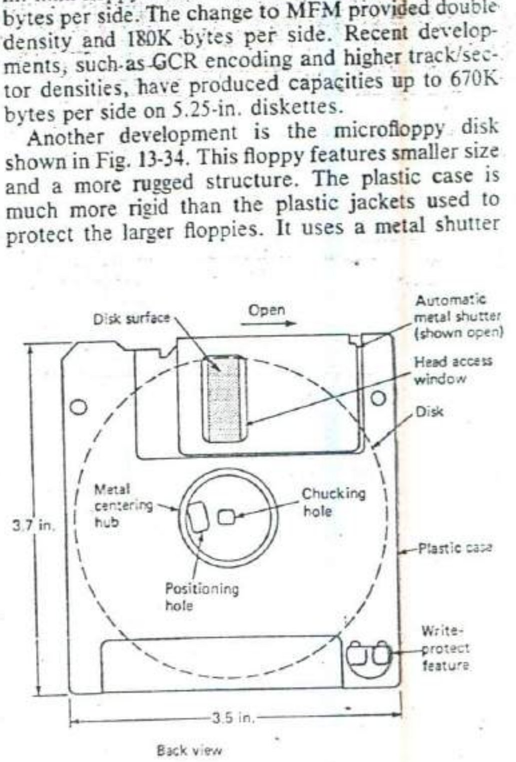


Fig. 13-34 Microfloppy.

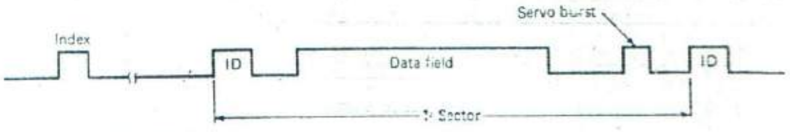


Fig. 13-35 Servo disk format. ID = track and sector identification bytes

that hides the head access slot to prevent the disk surface from being accidentally touched. The shutter is opened automatically when the disk is inserted in the drive. It is write-protected by rotating a small plastic piece located in the lower right-hand corner. Instead of an open center, it uses a metal drive hub for accurate centering and positive mechanical indexing. Microfloppies have capacities of around 400K bytes per side. They are one-fourth the size, one-half the weight, and consume one-half the power of minidrives (5.25 in.).

Disk technology improves rapidly. The designers are constantly finding ways to pack the data at higher densities. One of the limitations is the medium itself. The Mylar expands and contracts with heat and humidity variations, altering the concentricity of the tracks. All points on any track should be at the same distance from the center hole, but the variations produce track weave. With very high track density, the resulting positioning errors cause data errors. The clamping collett does not always center the disk accurately, thus also limiting track density. These problems have been solved by the newest minifloppies, which pack 3.3M bytes onto a double-sided disk. The drives are only half the size of a standard minifloppy. Greatly improved track density is made possible by a closed loop head-positioning system. Servo information is written to the diskette when it is manufactured. A servo writer places a burst of information in each sector, as shown in Fig. 13-35. A microprocessor-based controller in the drive continuously monitors the servo bursts and sends correction pulses to a fine stepper motor in the head-positioning mechanism. A coarse stepper motor is also available for ordinary track-to-track positioning. The result of this closed loop system is a head-positioning accuracy of within 2.5 μm of the exact data track. The allowable track density increases from 38 to 76 tracks per centimeter with this system. The disadvantages are higher drive and higher media costs resulting from the extra processing step added when the disks are made. Nonservo drives can use blank disks. A blank disk can be organized with track and sector information the first time it is used; this process is called *formatting the disk.*

Winchester drives using rigid aluminum disks are also available for small- and medium-sized computer systems. The original Winchester drive used a disk that was 14 in. in diameter. Subsequently, it was reduced to 8 in., and then to 5.25 in. The disks are fixed in Winchester drives. Today's small Winchester drives provide capacities from 5 to 100M bytes. They use heads that are permanently sealed with the disk to eliminate problems with smoke, dust, and dirt.

Removable cartridge drives are made with rigid disks that are 3.9, 5.25, or 8 in. in diameter. They are not true Winchester technology, but they are often referred to as *Winchester cartridges*. The cartridge drives offer high capacity (6.38M bytes for the 3.9-in. unit) and an average access time of 75 ms. Their data transfer rate is 5M bits/s, as compared to 250K bits/s for floppies. They also use servo-positioned heads under microprocessor control. Figure 13-36 shows a removable cartridge and drive. As the cartridge is inserted, the door slides open to permit access to the read/write heads, which were previously retracted to track zero. The disk is seated on a spindle and secured by a magnetic hub. The medium is shown in cross section and uses thin-film plating for the magnetic surface. This metal plating is 1000 times harder than ferric oxide, for decreased susceptibility to damage. It is also capable of supporting a magnetic density of over 3000 flux reversals per centimeter, which is 2.5 times better than that of oxide. A graphite lubricating coating is provided to shield against damage from dust and dirt. The cartridge drives eliminate the purge cycle associated

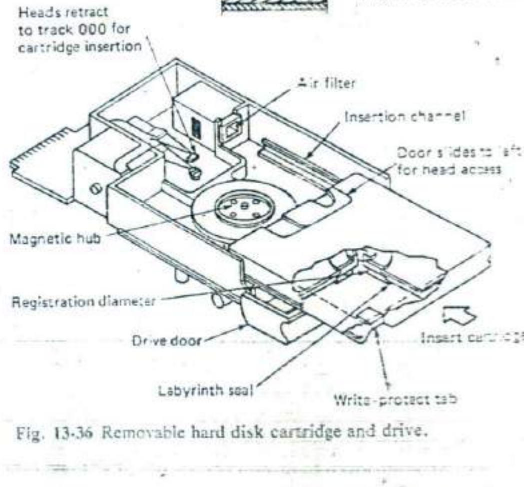


Fig. 13-36 Removable hard disk cartridge and drive.

with Winchester drives. A *purge* involves blowing filtered air over the disk to remove dust and debris. This can take several minutes and delays computer operation when the system is powered on.

Disk manufacturers are experimenting with a vertical recording format that may provide a 10-fold increase in disk capacity. The current recording format is horizontal (along the disk surface), as illustrated in Fig. 13-26. Vertical recording will allow a much higher bit density if the technical problems can be worked out.

The performance of disk drives, especially the floppies, can be "fussy." They may work only some of the time. They may refuse to read information that was written to the disk by a different drive. Speed errors greater than plus or minus 2 percent are one source of difficulty. Speed is adjustable on the older drives by turning an internal multiturn potentiometer. It may be set while watching a built-in strobe disk with the drive cover off and with the mechanism positioned under fluorescent lighting. The potentiometer is slowly turned until the strobe bars appear to be motionless. Or special diagnostic software may be available to assist in setting the speed. The newer drives use a servo speed-control circuit and are not adjustable. Clamping problems cause hub eccentricity because the disk does not properly center on the drive spindle. Remove the disk and try inserting it again when problems occur. *Head azimuth* is the angle at which the head intercepts the track's center line. Diagnostic software and special alignment disks may be available to aid in adjustment. Head azimuth is critical and will cause errors if it is out of tolerance. Drives that have seen quite a few hours of service may develop backlash in the positioning mechanism. This can be diagnosed with software that steps the head in one direction to a test track and then steps the head in the other direction to the same test track. Any backlash will show up as increased error for one of the trials. The heads may need cleaning from time to time. Special cleaning disks are available for this purpose. Finally, the medium does wear out; when occasional errors occur, try transferring the contents to a new disk.

REVIEW QUESTIONS

18. When reading a disk recorded with the NRZI technique, the _______ of the read pulses is immaterial.

19. When reading an NRZI disk, if no read pulse occurs within the bit cell interval, we know that a _______ was stored there.

20. Which disk encoding scheme writes a clock bit at the beginning of every bit cell?

21. The MFM encoding method writes a clock bit at the beginning of a bit cell only if that cell and the preceding cell contain a _______.

22. Which encoding method writes no clock bits to the disk?

23. What is the maximum number of consecutive 0s found on a disk using GCR encoding?

24. Floppy disks are made from Mylar, and rigid disks are made from _______.

25. What is the function of the servo burst shown in Fig. 13-35? When is it recorded onto the disk?

13-5 TAPE STORAGE

Magnetic tape storage was one of the earliest extensions of computer memory. It is a *serial access medium*. If the information is located at the other end of the tape, a significant delay will be experienced in loading the information back into computer memory. Disk storage systems have replaced tape as an extension of computer memory because they are block access devices with far less delay. However, magnetic tape is still an important part of many computer systems. Tape drives are commonly used to back up disk files. For example, suppose an industrial computer uses a 10M byte Winchester drive to store data and programs. If the disk fails, how will the information be restored? If it has been replicated on tape the system can be restored in minutes after the disk drive is repaired or replaced. Some computer operations have a daily back-up schedule to transfer the disk contents to tape. This practice ensures that no more than 1 day's work will be lost. It is possible to back up when using floppies, but the necessary disk swapping is tedious, time-consuming, and error-prone. Tape back-up is the method of choice for many applications.

Tape is easy to store. It is packaged in reels, cartridges, or cassettes. It is less vulnerable to loss of data than are floppies. The medium itself is 38-μm (0.0015-in.) thick Mylar with a ferric oxide coating on one side. The width ranges from 0.38 to 1.27 cm (0.15 to 0.50 in.). A reel that is 27 cm (10.5 in.) in diameter can hold as much as 1100 m (3600 ft) of tape. Modern tape drives use the same high-density recording techniques as the disk drives discussed in the last section. By using NRZI recording and GCR encoding, the storage capacity of a single reel of tape can be as high as 540M bytes. This is one of tape's strong features: vast quantities of data can be stored inexpensively.

The mechanical assembly that drives the tape and includes the heads is called a *transport mechanism*. High-capacity transports use reels to supply and take up the tape. They also use a capstan drive that rotates continuously. A solenoid-actuated pressure roller pinches the tape, which then comes up to reading or writing speed in several milliseconds. Reading and writing are performed at speeds up to 6.35 m/s, and even higher speeds are used for rewinding. The rapid acceleration saves tape because that part of it that passes over the heads during starting and stopping is wasted. No information can be stored at these locations, and they are known as *interrecord gaps*. The supply reel and the take-up reel are buffered from the capstan to avoid stretching or breaking the tape. The buffers are formed by tape loops that hang in vertical columns under vacuum tension. The reels are servo-driven to maintain a constant supply of tape in the buffers. Tension arms can also be used to form the tape buffers. Data are recorded in blocks with gaps in between. Special start and stop characters signal the beginning and the end of each block.

High-speed start/stop transports are expensive. Because tape is more often used today to back up disk storage, a less complicated transport is possible. Back-up tapes do not have to be designed to be searched for specific blocks of data. Streaming drives read or write all of the disk data in a continuous stream. Start/stop drives require fast mechanical performance and also require identifying header and trailer blocks to locate specific groups of data. These blocks are eliminated in streaming drives, and so are the interrecord gaps. Thus, streaming tape drives are less complicated and less expensive, and they pack more data on the tape.

Figure 13-37 shows a tape read/write head with separate gaps for writing and reading. This arrangement allows the data to be verified immediately after it is written. The head is actually a stack of nine sections to allow nine tracks or channels to be placed on the tape simultaneously. The nine-track format is convenient because it allows the data to be stored as bytes and allows an extra bit for parity. The parity channel can be used in conjunction with interspersed parity words to provide error correction, as shown in Fig. 13-38. A parity word follows a block of 8 data bytes: this arrangement provides a double parity check. The illustration shows an odd parity system. All the rows and columns should have an odd number of 1s. The circled bit is in error. It is found at the intersection of the row and column, where the parity is even in both cases. This information is used to locate and change the 0 bit that is in error to a 1. Multiple bit errors may not be detected by this method and cannot be corrected in any case. Actual instances of multiple errors are so low that more elaborate checking is not warranted. When no errors are detected, the parity channel and the parity words are stripped away, and the information is returned to the computer.

Cartridge tape systems provide convenient and compact mass storage for mini- and microcomputers. The tape drive takes up the same physical space as a minifloppy drive. It contains an *end of tape/beginning of tape* (EOT/BOT) sensor to prevent overrunning the tape. The cartridge is made from plastic and contains the tape, supply hub, take-up hub, and drive roller. The cartridge holds 137 m (450 ft) of 0.64-cm (0.25-in.) wide tape. The data capacity of one cartridge is as much as 17M bytes, with a transfer rate as high as 90K bytes/s. Back-up of a 10M byte Winchester can be accomplished in less than 3 min at this rate.

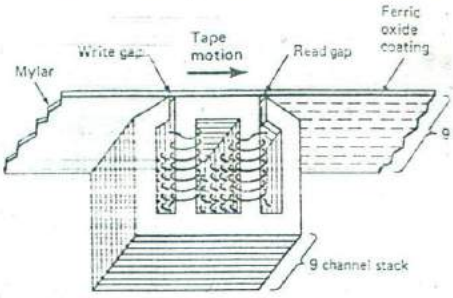


Fig. 13-37 Typical tape read/write head.

A *cassette* is a specific type of cartridge that was originally designed to record audio by Phillips of the Netherlands. The Phillips cassette is a well-established standard; other cartridges have not yet reached that same level of standardization. Special digital cassettes that are based on the Phillips standard and are certified for binary recording are manufactured. Binary recording is a saturated form of recording; audio is nonsaturated. The tape in digital cassettes uses a magnetic coating that is easier to saturate. The cassettes hold 137 m (450 ft) of tape. Some digital cassette tape drives use NRZI recording with GCR encoding and place four tracks on the tape for a 20M byte capacity. The data transfer rate is 30K bytes/s, and a 10M byte Winchester can be backed up in less than 6 min.

It is also possible to record binary data on an ordinary audio recorder. The binary data must be converted to audio tones that fit into the bandwidth of the recorder. Logic 0s are represented with four cycles of 1200-Hz sinusoidal audio and logic 1s with eight cycles of 2400-Hz sinusoidal audio. The bit time period is equal to 3.33 ms for both 0s and 1s using

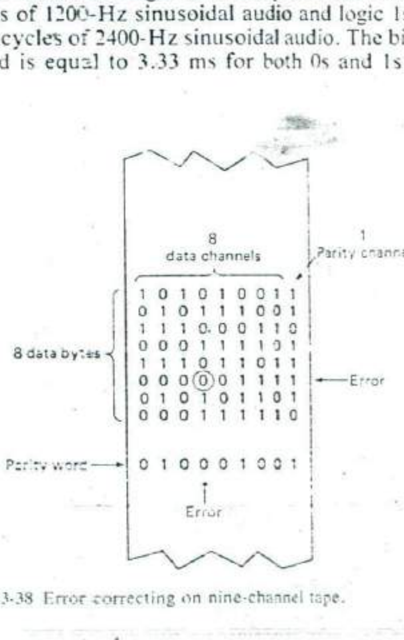


Fig. 13-38 Error correcting on nine-channel tape.

this technique. The result is a rather slow bit rate of 300 Bd. The data transfer rate is less, since the asynchronous format requires that 1 start bit and 2 stop bits be attached to each byte. Various techniques can be used to improve the bit rate, and 2400 Bd has been achieved. However, this still results in a relatively slow data transfer rate. Frequent errors are another problem, and because of these limitations the audio recording method has found acceptance only with hobby and personal computers.

Tape reels, cartridges, and cassettes should be stored away from magnetic fields. They should not be exposed to dust and dirt. Cartridges and cassettes may exhibit erratic behavior due to binding. It may be possible to free them temporarily by tapping the case on a hard surface. It is best to replace them at the first sign of any difficulty. The drive heads and other parts may require cleaning. Special cleaning cartridges and cassettes are available. Follow the manufacturer's recommended procedures.

REVIEW QUESTIONS

26. Computer memory can be divided into random access, block access, and serial access. Which type is the fastest? Which type includes magnetic tape? Disk?

27. A major application of tape storage is to back up ______ files.

28. Refer to Fig. 13-38. What would happen if the top bit in the parity column were read wrong? Assume that there would be no other errors.

29. Refer to Fig. 13-38. Suppose two of the 1s are read as 0s in the top data byte. Assume that there are no other errors. Can the 2 bits be corrected? Which of the 8 data bytes would have to be flagged with a possible error?

30. The Phillips cassette was originally designed to record ______.

13-6 BUBBLE MEMORY

The volatility of RAM is a serious drawback for some industrial computer applications. Power failures, brownouts, and power glitches can wipe out the contents of memory. Bubble memory is based on magnetic storage and is nonvolatile. It is used in applications such as numerical control, robotics, process control, remote terminals, and remote data acquisition.

Bubbles are based on domains. A *domain* is a group of atoms with parallel magnetic orientations. Each domain may be considered as a tiny unit magnet. When a material is unmagnetized, its domains are randomly oriented along three axes. Magnetizing the material orients a significant number of its domains along the same axis. If the material is magnetically saturated, most of its domains will have achieved the same orientation. It is possible to limit the axes of orientation to two by making a magnetic material very thin (less than 25 μm). Bubble memories are made by depositing a thin crystalline film of garnet on a substrate. Garnet is a material that occurs in nature; it can also be manufactured from iron oxide and yttrium. The domains in the garnet film are snakelike in shape and are perpendicularly oriented to the surface, as shown at the left in Fig. 13-39. The flux direction arrows and magnetic polarity signs show that the domain arrangement is random with no external field applied. When a weak external field is applied, the domains that are in opposition to the external field begin to become narrower. As the external field increases in strength, the opposing domains shrink in length and become cylindrical. They appear as circles when viewed from above the surface and are called *bubbles*. If the external field is strong enough, the opposing domains will vanish entirely. The external field is called the *bias field*.

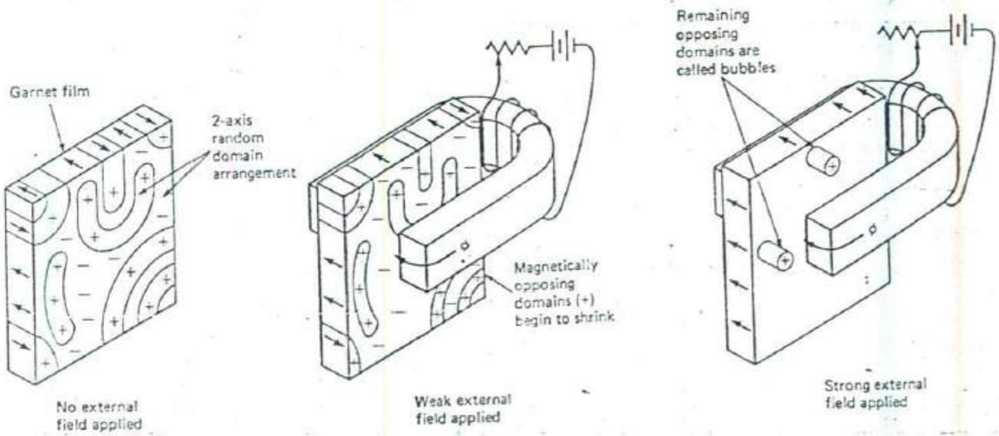

Garnet film
2-axis random domain arrangement
No external field applied
Magnetically opposing domains (+) begin to shrink
Weak external field applied
Remaining opposing domains are called bubbles
Strong external field applied

Fig. 13-39 Creating magnetic bubbles with a bias field.

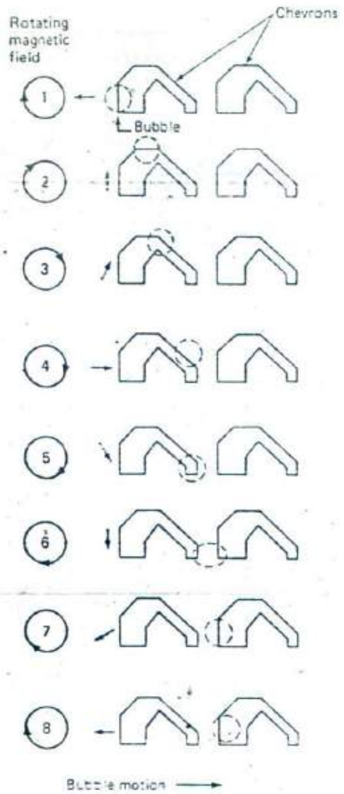


Fig. 13-40 Bubble propagation (Intel).

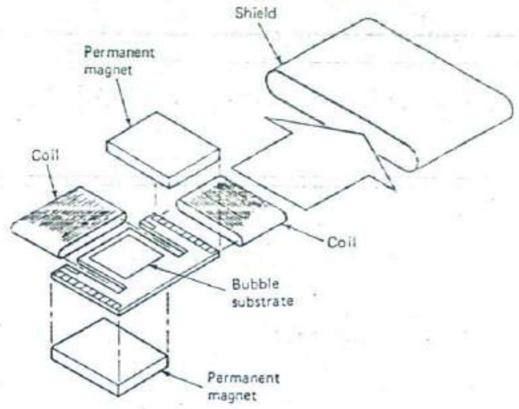


Fig. 13-41 Exploded view of magnetic bubble unit assembly (Intel).

The bubbles can be used to store binary information. The presence of a bubble indicates a 1, and the absence of a bubble indicates a 0. The bubbles are driven through the garnet film under a metallic pattern of chevrons. Figure 13-40 shows how the bubbles are made to propagate under the chevrons. An external rotating field interacts with the chevrons and provides the driving force. This rotating field is produced by exciting a pair of driver coils with two 50-kHz triangular waveforms that are 90° out of phase. The field rate is 50,000 rps. The data, in the form of bubbles, circulate in loops. It requires 82 ms for one bubble to make a complete loop for an average access time of 41 ms to get to the first data bit. The data transfer speed is on the order of tens of thousands of bits per second. Thus, bubble memories are serial access. They have no moving parts and offer very high reliability.

An exploded view of a *bubble memory assembly* is illustrated in Fig. 13-41. The bubble substrate is surrounded by two coils that provide the rotating field. It is also sandwiched between two permanent magnets that provide the bias field. The entire assembly is surrounded by a magnetic shield that provides a return path for the bias field and also prevents disturbances from outside magnetic forces.

The data storage is arranged in a block replicate architecture that is shown in Fig. 13-42. The bubble chip contains a number of endless loops. The bits of successive pages continuously circulate around these loops. A controller writes into and reads data from the storage loops by using the input and output tracks. A seed bubble provides a way for the controller to write a binary 1 to the input track, as shown in Fig. 13-43. The seed is generated when the bubble memory chip is manufactured by a current pulse in the hairpin loop. The pulse is strong enough to reverse the flux of the local bias field, and a bubble is

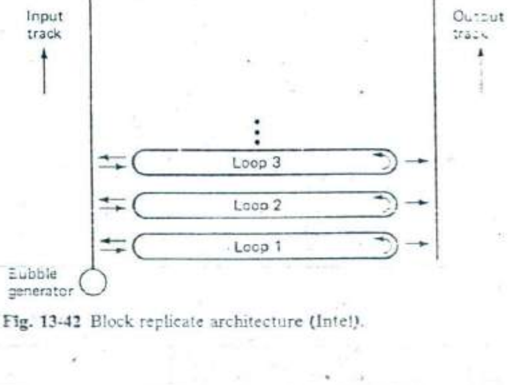


Fig. 13-42 Block replicate architecture (Intel).

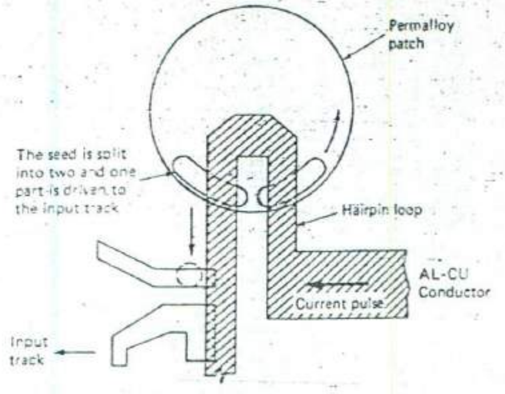

Permalloy patch
The seed is split into two and one part is driven to the input track
Hairpin loop
AL-CU Conductor
Current pulse
Input track

Fig. 13-43 Bubble generation (Intel).

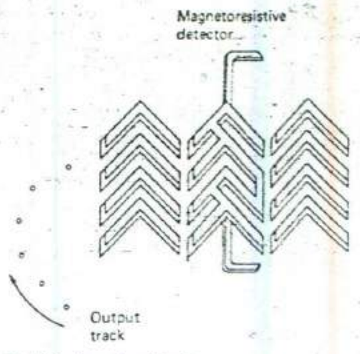


Fig. 13-45 Bubble detection (Intel).

created. The permalloy patch acts as a flux concentrater, and the seed assumes a kidney shape and rotates under the patch in reaction to the combined bias and rotating fields. When a 1 is to be written to the input track, a current pulse splits the seed in half. One half remains as the seed and quickly regains its original size. The other half, driven by the rotating field, propagates to the input track. When a 0 is to be written, no current pulse is applied.

A swapping procedure is used to transfer data (bubbles) from the input track to the storage tracks. The old data are brought back onto the input track for disposal at the end of the line, and the new data take their place in the storage loop. This occurs at the swap gate shown in Fig. 13-44 and is accomplished by sending a rectangular swap pulse through a conductor located under the chevrons. When data are to be read, the bubble is cut in two at the replicate gate by applying a current pulse with a steep leading edge. This process replicates the bubble for the output track, and the original data are retained in the storage loop. This is an example of *nondestructive readout*.

The bubble detector is near the end of the output track. It consists of a magnetoresistive bridge element and is shown in Fig. 13-45. Magnetoresistance is a change in electrical resistance due to the presence of a magnetic field. As the bubbles pass under the detector, an output of several millivolts is generated. Beyond the detector, the bubbles run into a rail and are destroyed.

Intel Corporation makes the 1M-bit 7110 bubble memory chip. Dummy detectors cancel the common mode noise induced by the rotating field. The chip is arranged as four identical quads. The entire device contains 320 storage loops, 272 of which are actually used. This leaves 48 spares. The decision about which to use is made after the unit is assembled and tested at the factory. The outcome of this decision is stored in an extra loop called the *boot loop* in the form of a 12-bit code for each active and inactive loop. Intel does this to increase the number of chips that will pass as acceptable. Nonworking loops will not prevent the memory from being fully functional unless more than 12 in any given quad are defective. Each active loop stores 4096 bits for a total capacity of 1,134,592 bits. The external appearance is 2048

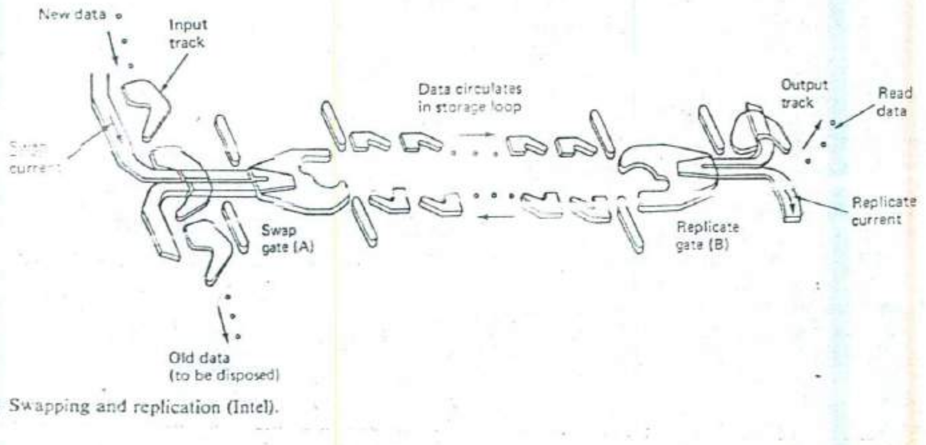


Fig. 13-44 Swapping and replication (Intel).

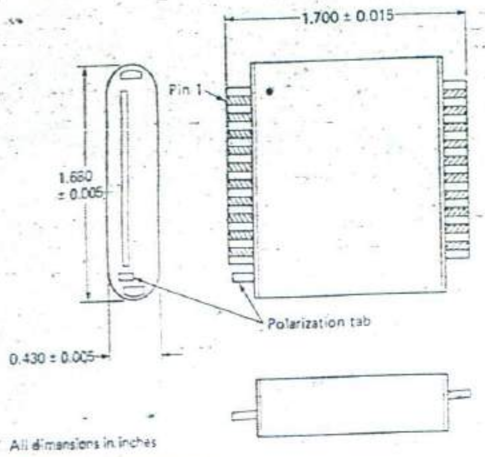


Fig. 13-46 7110 package (Intel).

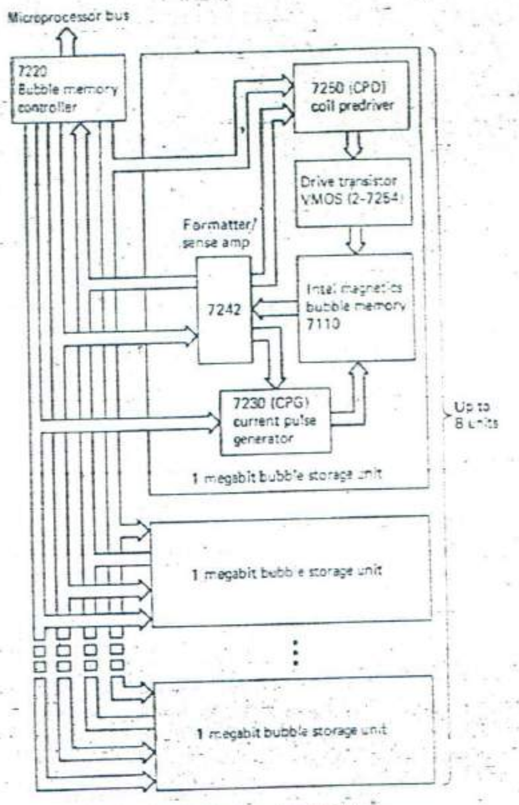

Microprocessor bus
7220 Bubble memory controller
7250 (CPD) coil predriver
Drive transistor VMOS (2-7254)
Formatter/sense amp
7242
Intel magnetics bubble memory 7110
7230 (CPG) current pulse generator
1 megabit bubble storage unit
Up to 8 units
1 megabit bubble storage unit
1 megabit bubble storage unit

Fig. 13-47 Bubble memory expansion (Intel).

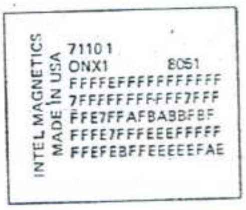
pages of 512 bits each. The physical appearance of the 7110 is shown in Fig. 13-46.

Intel also makes a group of support chips to make application of the bubble memory device easier. Fig. 13-47 shows the block arrangement of the bubble memory chip, along with the 7220 bubble memory controller, the 7230 current pulse generator, the 7242 formatter/sense amplifier, the 7250 coil predriver, and two 7254-VMOS coil drivers. One controller chip can handle up to eight memory units for a total of 1M byte of storage.

Bubble memories are extremely reliable. However, there are a few things that can go wrong. A memory failure may be due to the loss of one or more seed bubbles. This loss is usually caused by power supply problems. Be sure to verify proper operation of the power source for the bubble memory. A software verification procedure can then be used to check the status of the seeds. Intel recommends the following procedure to reseed the memory in the event that it is required. It assumes that certain software routines are available in computer RAM or ROM memory.

1. Power down.
2. Remove the 7230 pulse generator chip from its socket and place it into the special seed generator module.
3. Plug the seed generator module into the empty 7230 socket.
4. Power on.
5. Call ABORT (this is a software procedure that aborts the present command and resets the controller).
6. Call MBMPRG (a purge routine).
7. Call WRBUBL (write to bubble memory routine: the registers in the controller must contain the correct data).
8. Power down.
9. Restore the pulse generator chip.

The next step is to read the *boot loop code* inside the 7110; this code is used at every power on as a part of the bubble system initialization procedure. The code is also printed on the device label as in Fig. 13-48. The code is stored in a register located in the 7242 formatter/sense amplifier chip. From then on, this register defines the active loops for reading and writing. The formatter does not store any information to the inactive loops, and the sense amplifier ignores any data that come from them. In trouble-


INTEL MAGNETICS
MADE IN USA

```
71101
ONX1          8051
FFFFEFFFFFFFFFFF
7FFFFFFFFFFF7FFF
FFE7FFAFBABBFBF
FFFE7FFFEEEFFFFF
FFEFEBFFEEEEEFAE
```

Fig. 13-48 A 7110 bubble memory label showing the boot loop code.

shooting, the boot loop code should be matched byte for byte with the code found on the label of the 7110 bubble memory chip, as shown in Fig. 13-48.

REVIEW QUESTIONS

31. List the three types of nonvolatile memory covered in this chapter.

32. The permanent magnets in a bubble memory chip are used to provide the _______ field.

33. Bubble memories use a metallic pattern of _______ to guide the bubbles as they move.

34. The force that moves the bubbles is supplied by a _______ magnetic field.

35. A 1 is written to the input track with a current pulse that splits the _______ bubble into two parts.

36. With no swap pulses applied in Fig. 13-44, what happens to the data in the storage loop?

37. What happens when swap pulses are applied?

CHAPTER REVIEW QUESTIONS

13-1. Calculate the nominal output voltage for an 8-bit DAC with a 5-V supply and a binary input of 10010111.

13-2. Digital-to-analog converters that use an external signal as their reference may be referred to as _______ DACs.

13-3. Identify the A/D converter that uses an up/down counter, a DAC, a clock, and a comparator as the major circuit elements.

13-4. Can successive approximation A/D conversion be accomplished by using a microprocessor to write binary data to the DAC and thus eliminate the clock and the register?

13-5. The binary output from a dual-slope A/D converter is established by counting the time that the _______ signal is applied to the integrator.

13-6. What type of circuit can be used to "freeze" a time-varying analog signal for A/D conversion?

13-7. How many comparators would be required to build a 10-bit flash converter?

13-8. How many comparators would be required to build a dual-stage flash converter with 10 bits of resolution?

13-9. Much faster data rates are possible on _______ transmission circuits.

13-10. Differences in ground potentials can cause errors in circuits using the _______ standard.

13-11. The EIA has established the _______ standard for balanced transmission.

13-12. High-speed data circuits are subject to errors due to reflected signals, especially when the transmission lines are not properly _______.

13-13. The RS-232C, RS-422, RS-423, and RS-449 are all bit serial standards; the IEEE 488 is a bit _______ standard.

13-14. Which type of polling on the IEEE 488 bus provides the controller with most information? Which type is the fastest?

13-15. Polling is usually initiated by the controller after a talker or a listener asserts the _______ line on the IEEE 488 bus.

13-16. Should the write tab be opened or covered to write-protect a 5.25-in. diskette?

13-17. Floppy disks _______ be cleaned.

13-18. What is the function of the stepper motor in disk drives?

13-19. How is the average access time for the directory track minimized?

13-20. Which floppy disk uses a hard plastic case and a metal shutter to hide the head access slot?

13-21. Winchester drives use rigid disks that are _______ with the head assembly.

13-22. When a Winchester drive is first turned on, there is a start-up delay due to the _______ cycle.

13-23. A start/stop tape drive must start and stop on blank sections of tape known as _______ gaps.

13-24. Tape drives that are limited to back-up operation are known as _______ drives.

13-25. Digital cassettes contain tape that is easier to _______ than audio tape.

13-26. Digital information can be recorded on an ordinary audio recorder by converting the 0s and 1s to _______.

13-27. What happens when data are read from the storage loop in a bubble memory?

13-28. The bubble detector is based on the principle of _______.

13-29. Where is the code that identifies the active loops in a bubble memory chip stored?

13-30. How many bubble memory units can the Intel 7220 controller handle?

13-31. Under normal circumstances, when are the seed bubbles generated?

13-32. When is the boot loop code normally read into the 7242 formatter/sense amplifier chip?

13-33. Is the boot loop code ever read at any other time?

13-34. Is it possible to reseed a bubble memory?

13-35. What are the other two names used for the IEEE 488 standard?

ANSWERS TO REVIEW QUESTIONS

1. 0.00488 V; 4.995 V **2.** 0.098 percent **3.** 0.024 percent **4.** 7.5 V **5.** staircase **6.** 512 μs **7.** 16 μs **8.** none **9.** one of eight analog inputs **10.** it decreases; it will be less **11.** unbalanced **12.** balanced **13.** unbalanced **14.** hysteresis **15.** RS-423 uses the driver ground only; RS-232C grounds both ends **16.** they control handshaking **17.** from 6 to 9 **18.** polarity **19.** 0 **20.** FM **21.** 0 **22.** GCR **23.** 2 **24.** aluminum **25.** provides control information for accurate head positioning when the disk is manufactured **26.** random; serial; block **27.** disk **28.** the error intersection would occur at the parity bit, and all 8 data bytes would be accepted as being correct **29.** no; all of them **30.** audio **31.** disk, tape, and bubble **32.** bias **33.** chevrons **34.** rotating **35.** seed **36.** it circulates continuously **37.** new data enter the storage loop and old data leave it

14 OPTOELECTRONICS

The combination of optics and electronics, called optoelectronics, *has evolved into a major factor in industrial systems. Devices such as light-emitting diodes, lasers, photodiodes, and fiber optics, along with the older standard devices such as photo cells, will be covered in this chapter. The theory of operation of optoelectronic devices, their characteristics, and their applications will be presented.*

14-1 EMITTERS

There are many different light sources, such as tungsten lamps, fluorescent lamps, neon lamps, xenon lamps, and light-emitting diodes (LEDs). The characteristics and operation of the conventional light sources (lamps, flash tubes, sunlight) are familiar. This section will be devoted to a newer source, the LED.

Electroluminescence in solids is a phenomenon which has been well known and intensively studied for many years. Perhaps the most commony utilized application of electroluminescence is in the screen of an oscilloscope or a television set. Better known as *cathodeluminescence*, this type of electroluminescence is caused by the collision of high-energy electrons with the phosphor coating which lines the inside surface of a cathode-ray tube. One particular type of electroluminescence has given rise to an entirely new field of technology. This is the phenomenon of *PN junction electroluminescence*, which results from the application of direct current at a low voltage to a suitably doped PN junction. This is the basis of the LED, a PN junction diode that emits light when biased in a forward direction. The light emitted can be in either the invisible (infrared) or the visible spectrum.

Semiconducting light sources can be made in a wide range of wavelengths, ranging from the near-ultraviolet region of the electromagnetic spectrum to the far-infrared region. This wavelength limit is due to today's production constraints. The LEDs used in electronic applications (as emitters) are normally *infrared-emitting diodes* (IREDs).

Spectral radiation can be divided into two basic types: *coherent light* (in which the light waves are in phase and are usually produced at cryogenic temperatures) and *noncoherent light* (the waves are out of phase with each other). The following discussion will consider only noncoherent light emitters, invisible as in the case of IREDs or visible as with LEDs.

A PN junction can be formed in a semiconductor material by doping one region with donor atoms and an adjacent region with acceptors. If an external bias is applied across the junction, a bias current flow in the PN junction causes holes to be injected into the N-type material and electrons to be injected into the P-type material. This process is illustrated in Fig. 14-1. The radiation from a PN junction arises from the recombination of electrons with the minority carriers, and energy proportional to the band gap energy of the semiconductor material is released. Some of this energy is released as light, and the remainder is released as heat. The proportion is determined by the mixture of recombination processes taking place. The energy contained in a photon of light (shown in Fig. 14-2) is proportional to the frequency (that is, the color). The higher the band gap energy of the semiconductor material forming the junction, the higher the frequency of the light emitted. In industry, the wavelength of emitted light is commonly expressed in nanometers (nm). Photon energy can be converted to wavelength by the equation

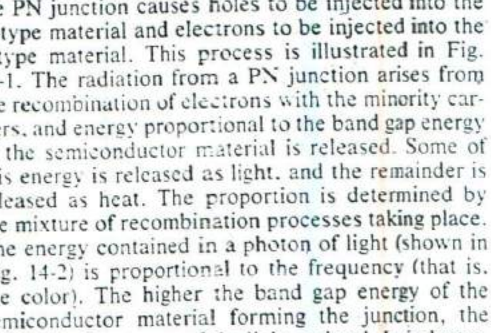


Fig. 14-1 PN junction forward-biased.

$$\lambda = \frac{1240}{E}$$

where E = energy transition, electronvolts (eV)
λ = wavelength, nm

It is possible to increase the wavelength by decreasing the band gap energy. Wavelengths of 1000 nm are possible but expensive to produce. However, the long wavelength emitters are useful in fiber-optic communications and will be covered later in this chapter. Commercially available LED devices are made from gallium arsenide (GaAs), gallium phosphide (GaP), or the (three-element) ternary compound gallium arsenide phosphide Ga (As,P).

There are two types of IREDs, and both use a low-band-gap, silicon-doped epitaxially grown material, *gallium arsenide* (GaAs). The GaAs diodes are efficient and very reliable and produce a peak wavelength of 940 nm. The second type is manufactured by replacing some of the gallium with aluminum. This increases the band gap energy, yielding an IRED with a wavelength of 880 nm. The gallium aluminum arsenide (GaAlAs) emitters are much more efficient than the GaAs emitters. Also, their wavelength is a better match for silicon detectors, thereby increasing system sensitivity.

Gallium phosphide (GaP) is used for visible-light-emitting diodes. The mechanism for visible light radiation is the same as for the infrared diodes. The transition of electrons from the conduction band to the acceptor level releases a photon. The wavelength of the photon is in the visible spectrum as shown in Fig. 14-3, and for the green LED, λ = 560 nm. The wavelength is dependent on the energy gap of the semiconductor. For GaP, E_g = 2.24 eV, and the wavelength is (560 nm). It should be noted that there is a single transition of electrons from the higher to lower bands. If an impurity such as oxygen is contained in the GaP semiconductor, an intermediate energy level will exist deep in the forbidden region close to the valence band. Figure 14-4 shows the oxygen level, E_{ox}, appearing 1.8 eV above the acceptance level. This oxygen trapping level will force the electrons to have double transitions, one from the conduction band to E_{ox}, and the other from E_{ox} to E_{ACC}. The photon release during the first transition period is of low energy deep in the infrared area. This transition is of no value to the operation of a visible LED. The second transition, from E_{ox} to

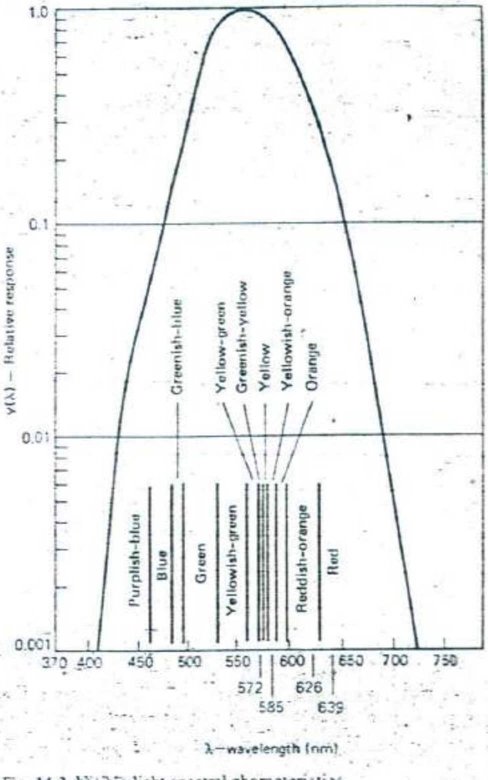


Fig. 14-3 Visible-light spectral characteristics.

E_{ACC}, will cause radiation in the red area of the spectrum at λ = 700 nm. The combination of both red and green emission, by double transitions due to increased current, will give a yellow or orange light. At high current levels, most electrons make the single transition from E_c to E_{ACC}, and green light of high intensity results. The red emission in this case is negligible compared to green emission.

The Ga(As,P) is produced as an epitaxial layer grown on a substrate of either GaAs or GaP. The grown wafer is then processed to produce PN junctions with photolithography used to shape the structure. Figure 14-5 shows the cross section of an LED. The photons generated at the junction of a PN electroluminescent diode are emitted in all directions. If

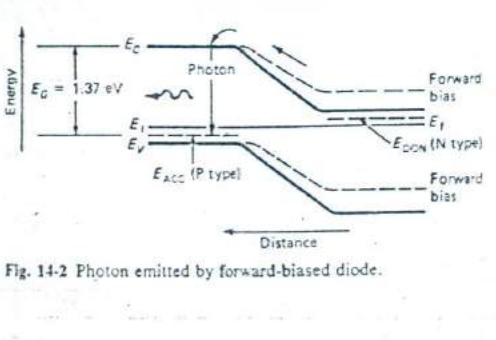


Fig. 14-2 Photon emitted by forward-biased diode.

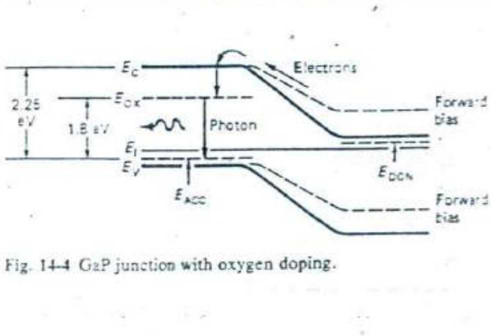


Fig. 14-4 GaP junction with oxygen doping.

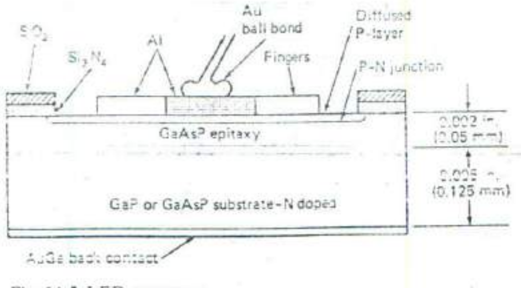


Fig. 14-5 LED structure.

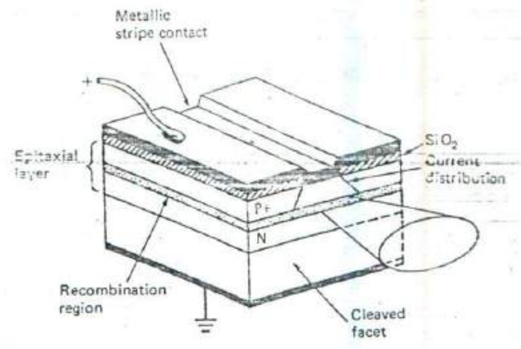


Fig. 14-7 Edge-emitting diode structure.

the diode substrate is opaque, as in the case of GaAs, only those photons which are emitted upward within a critical angle will be emitted as useful light. All other photons emitted into or reflected into the bulk crystal will be absorbed. This phenomenon is illustrated in Fig. 14-6(*a*). Gallium phosphide is nearly transparent compared to GaAs: diodes grown on the GaP substrate will exhibit improved efficiency caused by the emission of photons which would be absorbed in the GaAs substrate. This is illustrated in Fig. 14-6(*b*). The most common uses for visible LEDs are in displays; their implementation will be covered in the next section.

The surface-emitting infrared diodes are similar in structure to the visible LEDs. They are used where low cost and moderate performance are required. It is possible to construct an edge-emitting IRED. The physical structure of an edge emitter consists of a rectangle-shaped semiconductor die (PN junction) in which the radiant output is emitted from the edges of the diode in the recombination region of the junction, as shown in Fig. 14-7. The lateral size of the radiation area is usually defined by etching an opening in an oxide insulating layer and forming an ohmic contact by depositing a metal film into the open contact region. This type of construction is referred to as *edge emitters* utilizing *stripe geometry* (as opposed to the surface-emitting structures just covered). The edge-emission structure has an oxide metallization stripe constricting the current flow through the recombination region to that area of the junction directly below the stripe contact. It is possible by using this technique to confine the radiating portion of the junction to a spot approximately 50 micrometers (μm) in its largest dimension.

When the edge-emitting structure is combined with some other epitaxial processes, it is possible to restrict the angle of radiation to a relatively narrow angle. A comparison of this narrow edge radiation pattern to the parallel (surface) pattern is shown by Fig. 14-8. These devices exhibit excellent high-speed performance in excess of 100 MHz and are capable of a much greater output of radiant flux.

As mentioned earlier, the surface-emitting infrared diodes are used in modest-performance applications. They exhibit power outputs of 2 to 7 mW in continuous service and 26 to 200 mW in pulse service at 940 nm. Figure 14-9 shows the radiant flux output from a typical IRED emitter at a temperature of 27°C. It is crucial to maintain a safe operating temperature as with all semiconductors. There is an order of magnitude (factor of 10) difference between the radiant flux output for the steady-state and pulsed modes. These are *flux curves*, not to be confused with the characteristic volt-ampere curves of the diode, which are shown in Fig. 14-10.

The presence of infrared (IR) light is not obvious. It cannot be seen by the eye; special means of detection, which are discussed later in this chapter,

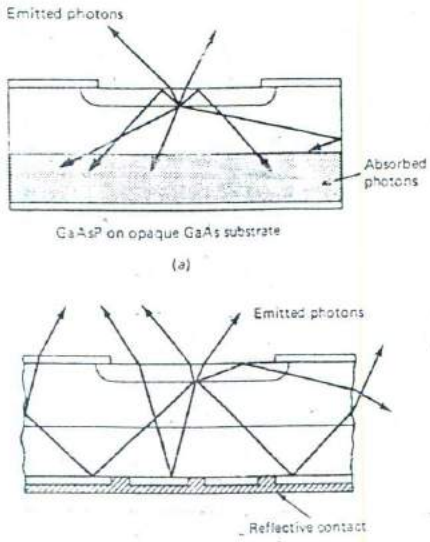


Fig. 14-6 Transparent and opaque substrates.

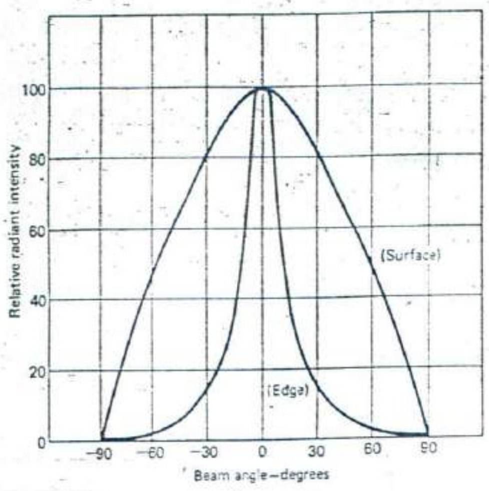


Fig. 14-8 Emitted patterns.

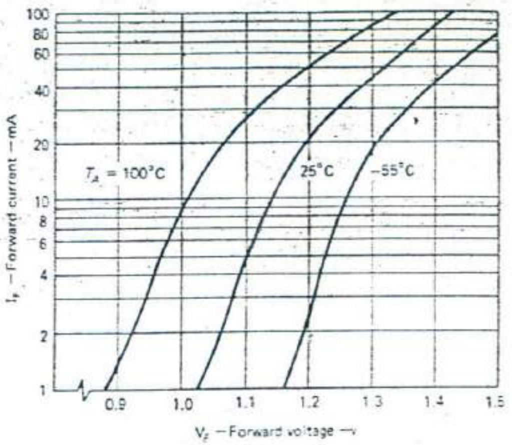


Fig. 14-10 Typical V-I characteristics of an IRED.

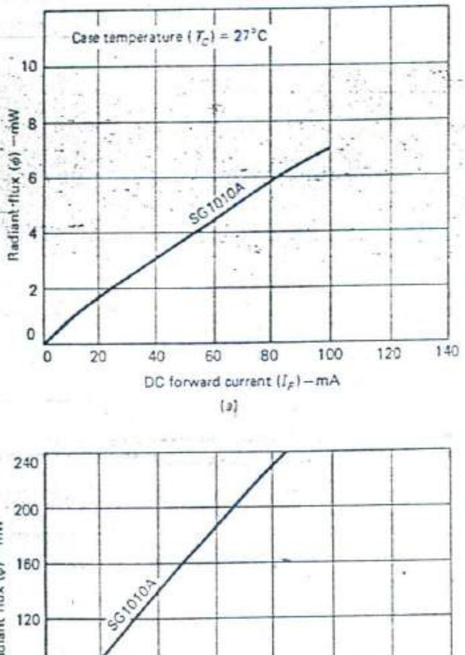


Fig. 14-9 Radiant flux versus dc and pulsed operations. (a) Steady-state current. (b) Pulsed current.

must be employed. The wavelength (long compared to visible radiation wavelength) would permit the transmitted flux to pass through the paper of this book as though it were transparent. Blocking or interrupting this wavelength requires a more dense material, and this point should be kept in mind. Suitable precautions should be taken to protect one's eyes from the radiant flux. This safety consideration will be covered in more detail when lasers are explored.

For best performance, emitters should be biased from a current source rather than a voltage source. A simple solution is to place a resistor in series with a voltage source to approximate a current source. A few examples of typical emitter circuits are shown in Fig. 14-11. These are building blocks for the more sophisticated and optimum circuits. Information can be impressed on the emitter output by modulating the bias input to the diode. The modulation is of the amplitude modulation (AM) type. Either voice or radio frequency (RF) signals can be used for modulation, but it is important to know the frequency limitations of the diode being modulated. Typical (simple) modulator circuits are shown in Fig. 14-12. As mentioned previously, frequencies in the 100-MHz region can be obtained with some emitters. Note that all circuits have a current-limiting resistor in series with the diode.

The IRED is often operated in the pulse mode for the transmission of digital information over short distances or to send signals in and out of various unfriendly environments such as gas, water, and high voltage. The circuit may be as simple as Fig. 14-13(*a*) or as complex as Fig. 14-13(*b*).

The IRED has a maximum reverse voltage rating of only 2 V, and this value should not be exceeded. Many IREDs are destroyed during testing. Check all data sheets thoroughly before performing any tests.

Some of the GaAs diodes are assembled in conjunction with a glass lens to produce a narrow-beam

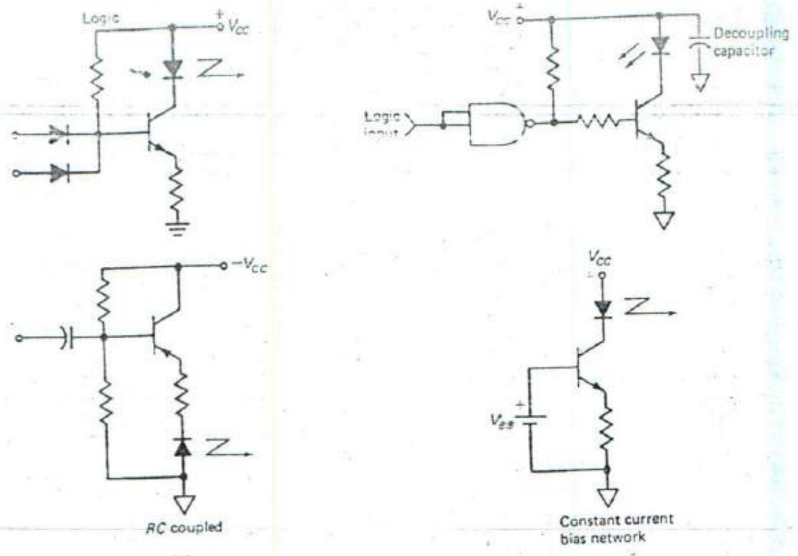


Fig. 14-11 Typical emitter circuits blocks.

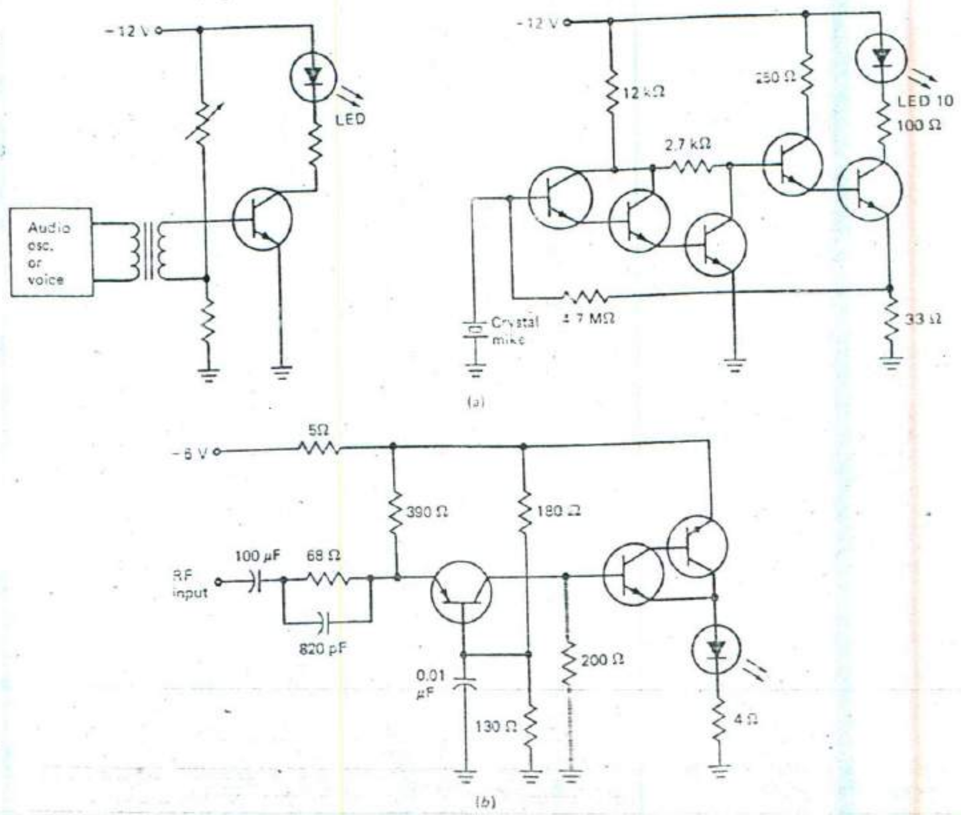
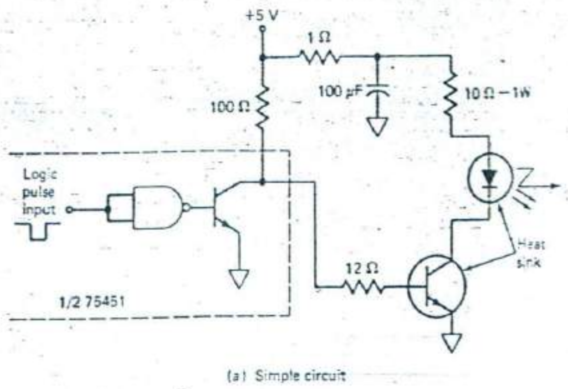


Fig. 14-12 Emitter-modulation circuits. (*a*) Audio frequency. (*b*) Radio frequency.



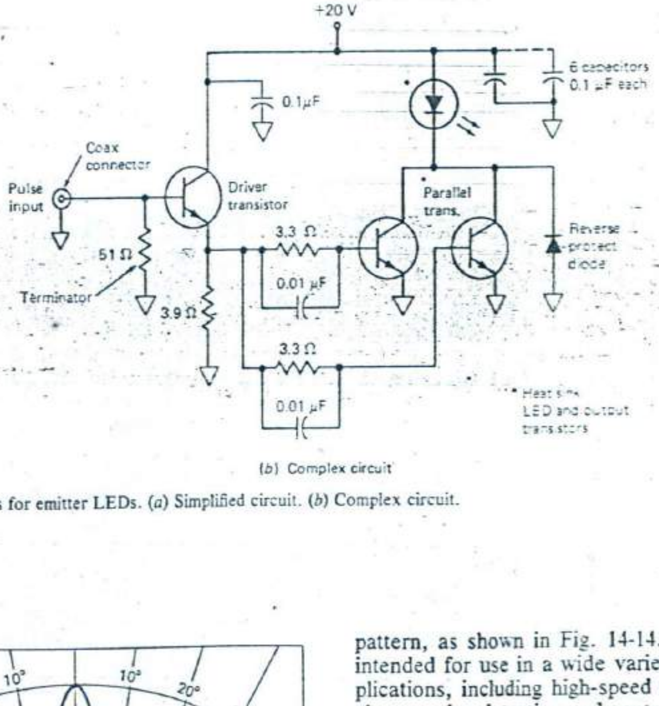


Fig. 14-13 Pulse circuits for emitter LEDs. (*a*) Simplified circuit. (*b*) Complex circuit.

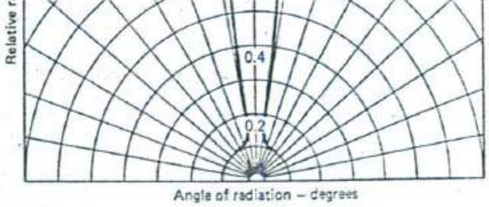


Fig. 14-14 Typical lens-corrected radiant flux pattern.

pattern, as shown in Fig. 14-14. These IREDs are intended for use in a wide variety of industrial applications, including high-speed counting, intrusion alarms, edge detection and control, collision protection, and data transmission.

REVIEW QUESTIONS

1. Light emission from a forward-biased PN junction is known as ______.

2. The lower the frequency of the light emission, the ______ the wavelength.

3. Infrared-emitting diodes emit coherent light waves. (true or false)

4. ______ are released as part of the energy of recombination in the PN junction.

5. Calculate the number of electronvolts required for a wavelength of 620 nm (near red).

6. The GaAlAs diode emits a _______ nm wavelength in the infrared range.

7. The GaP PN diode is used for _______ light radiation.

14.2 DETECTORS (SOLID-STATE)

The reverse-biased PN junction is basic to operation of photosensitive semiconductor devices. When a photon is absorbed in a semiconductor, a hole-electron pair is formed and swept across the junction by the electric field (ϵ) developed across the depletion region. A photocurrent due to the separated hole-electron pairs results. The electrons go to the N side, and the holes go to the P side. Separation of a photon-generated hole-electron pair is more likely to occur when the pair is formed in a region where there is an electric field (ϵ); see Fig. 14-15. The alternative to separation is for the hole-electron pair to recombine, with no contribution to photocurrent. The photocurrent flow in the external circuit is proportional to the illumination.

Electric field distribution in a semiconductor diode is not uniform, as indicated in Fig. 14-15. In the regions of the P-type diffusion (front) and N-type diffusion (back), the field is much weaker than it is in the center region, known as the *depletion region*. For best results, a photodiode should be made so as to allow the greatest number of photons to be absorbed in the depletion region. That is, the photons should not be absorbed until they have penetrated as far as the depletion region, and they should be absorbed before penetrating beyond the depletion region.

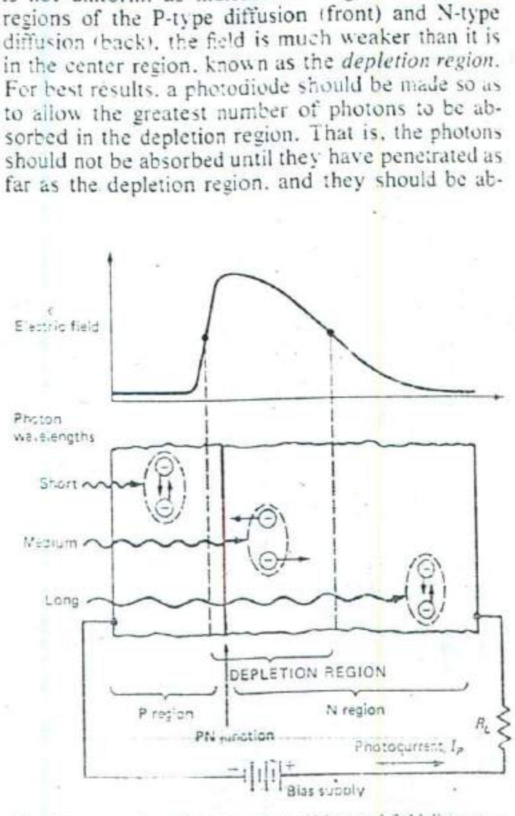


Fig. 14-15 PN photodiode junction and internal field diagram.

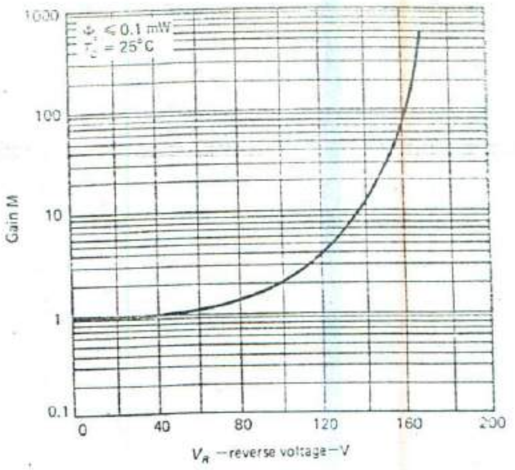


Fig. 14-16 Photodiode gain versus reverse voltage.

The depth to which a photon will penetrate before it is absorbed is a function of the photon wavelength. Short-wavelength photons are absorbed near the surface; those of longer wavelength may penetrate the entire thickness of the crystal (see Fig. 14-15). Therefore, if a photodiode is to have a broad spectral response (broad spectrum of wavelengths) it should have a very thin P-layer to allow short-wavelength photons to penetrate, as well as a thick depletion region to maximize photocurrent from the long-wavelength photons.

The thickness of the depletion region depends on the resistivity of the region to be depleted and on the reverse bias. A depletion region exists even if no reverse bias is applied because of the "built-in" field produced by diffusion of minority carriers across the junction (at room temperature). Reverse biasing aids this built-in field and expands the depletion region.

In PN photodiodes, a thin P diffusion allows good short-wavelength response. A deep P diffusion degrades the short-wavelength response but lowers the bias required for good response at longer wavelengths.

An avalanche photodiode (APD) uses avalanche multiplication to amplify the photocurrent created by hole-electron pairs. The PN junction is operated at a high reverse bias voltage needed for avalanche multiplication (refer to Fig. 14-16). Very high multiplication factors (M) can be achieved, but the process is very noisy. The photocurrent signal gain is magnitude-dependent on the reverse voltage, as illustrated in Fig. 14-16. These devices are usually optimized for detection of infrared-radiant energy but are useful from audio to microwave frequencies.

Because the avalanche photodiode's temperature

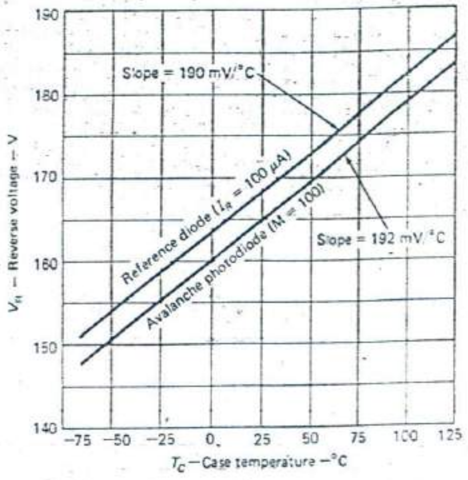


Fig. 14-17 Photodiode reverse voltage versus temperature.

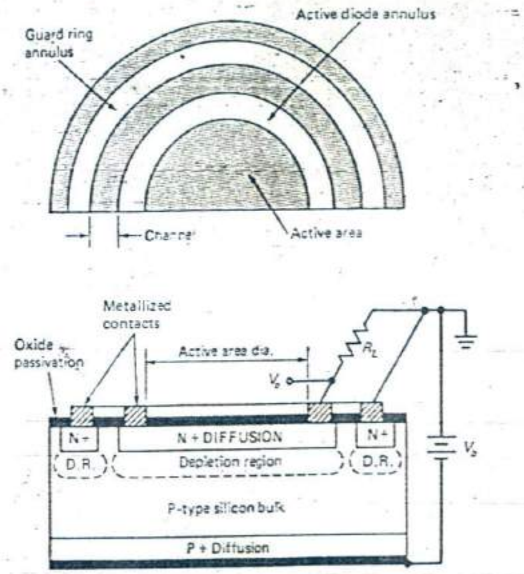


Fig. 14-19 Diffused guard ring construction.

stability is poor it must be carefully controlled (see Fig. 14-17). To compensate for this effect, diode pairs consisting of an avalanche photodiode (APD) and a small reference diode are manufactured together in the same package. This pairing makes it possible to build a temperature-compensating bias circuit that will hold the avalanche gain constant over wide temperature variations. The diode pair is in a hermetically sealed case with a window of borosilicate glass. Figure 14-18 shows a block diagram application of a temperature-compensating bias circuit for an avalanche diode.

Besides the photocurrent generated, a *dark (reverse leakage) current* exists in a photodiode. The two main contributors of dark current (diode leakage) are surface leakage and bulk leakage. The name *dark current* indicates what it implies: the current (leakage) that flows without light. The surface leakage of a photodiode is 100 times that of the bulk leakage. Some photodiodes employ a unique diffused guard ring construction to minimize this surface leakage. Figure 14-19 illustrates the guard ring diffused on the active area. In normal operation the guard ring (a PN junction of its own) and the active area are biased at the same potential. The surface leakage is now shunted around the load resistor and flows through the guard ring, as shown in Fig. 14-20. Now the main source of leakage will be the much lower bulk leakage. The shunting of surface leakage around the load by the guard ring also improves the diodes' noise figure since the noise current varies directly as the square root of the leakage current.

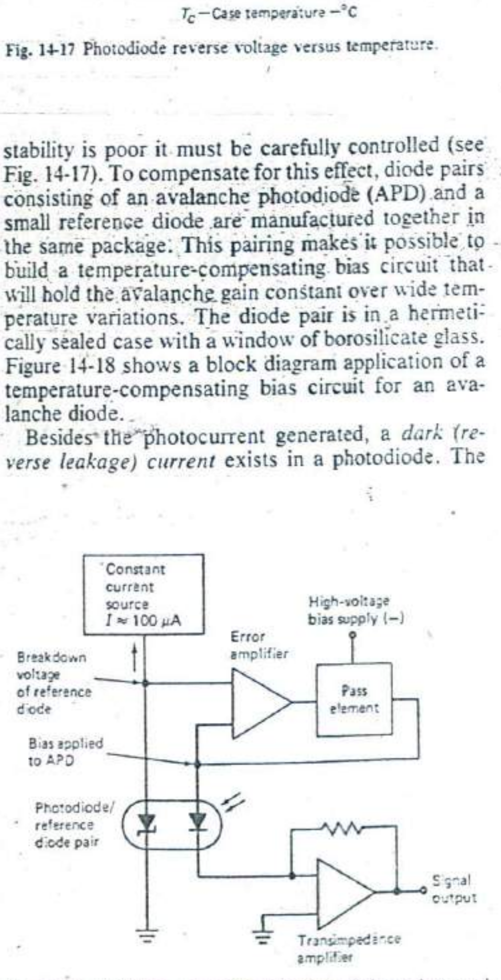

Constant current source $I \approx 100\ \mu A$
High-voltage bias supply (−)
Error amplifier
Pass element
Breakdown voltage of reference diode
Bias applied to APD
Photodiode/ reference diode pair
Signal output
Transimpedance amplifier

Fig. 14-18 Avalanche photodiode/reference diode package and block diagram.

The *light-sensitive transistor* is in reality one of the simplest photodiode-amplifier combinations. Directing light toward the reverse-biased PN junction (collector-base) generates base current, which is amplified by the current gain of the transistor. Figure 14-21 shows the schematic, equivalent circuit, and junction (with depletion region). External biasing of the base is possible, if that lead is available, so that the equation for emitter current is

$$I_e = (I_p + I_b)\cdot(h_{FE} + 1)$$

where I_b = base current
I_p = photon-generated base current
I_e = emitter current
h_{FE} = transistor dc current gain

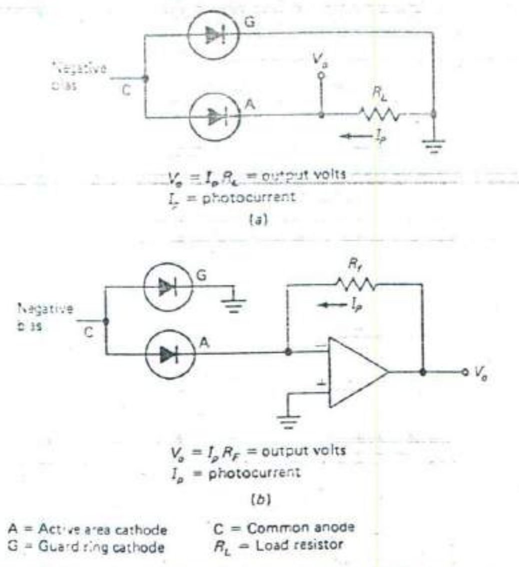


Fig. 14-20 Avalanche photodiode/guard circuit. (*a*) Basic operating circuit. (*b*) Photodiode/op amp circuit.

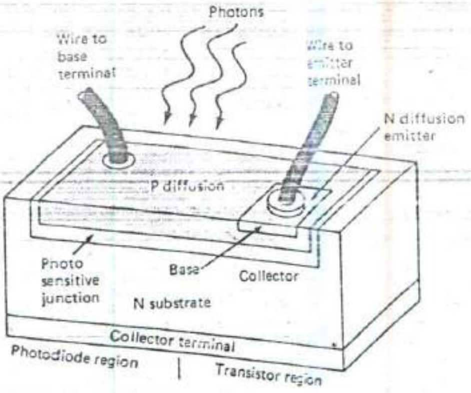


Fig. 14-22 Phototransistor.

The sensitivity of this type of transistor can be influenced by the bias levels at the base. The response of the phototransistor will vary as h_{FE} varies with current, bias voltage, and temperature. A high value of h_{FE} and a large collector-base junction area are required for high sensitivity but can also cause high dark current levels when the collector-base junction is reverse-biased, as indicated in Fig. 14-21. The phototransistor dark current is given by

$$I_{ceo}\,(\text{dark}) = h_{FE} \times I_{cbo}$$

where I_{cbo} = collector-base junction leakage current (the "o" denotes an open emitter lead)
I_{ceo} = collector-emitter leakage (the "o" denotes an open base lead)

This leakage is proportional to the junction area and the perimeter at the surface. Careful processing of the transistor chip, shown in Fig. 14-22, is required to minimize the dark current and maintain high light sensitivity. Typical phototransistor dark currents with a 10-V reverse bias are on the order of 10 nA at room temperature and double by a factor of 2 for every 10°C rise in temperature. This is a very important point and worth remembering; it could minimize troubleshooting time when temperatures are suspect.

Dark current effects may be minimized for low light applications by keeping the base-collector junction from being reverse biased, that is, having a V_{ceo} of less than one silicon diode forward voltage drop. This technique will allow light currents in the nanoampere range to be detected. The circuit shown in Fig. 14-23 illustrates this type of operation. The band gap effect of the doped base-emitter (B-E) junction of Q_1 sets the open base potential. thereby forcing $V_{be}(Q_1)$ to be equal to one diode drop. Since $V_{be}(Q_1)$

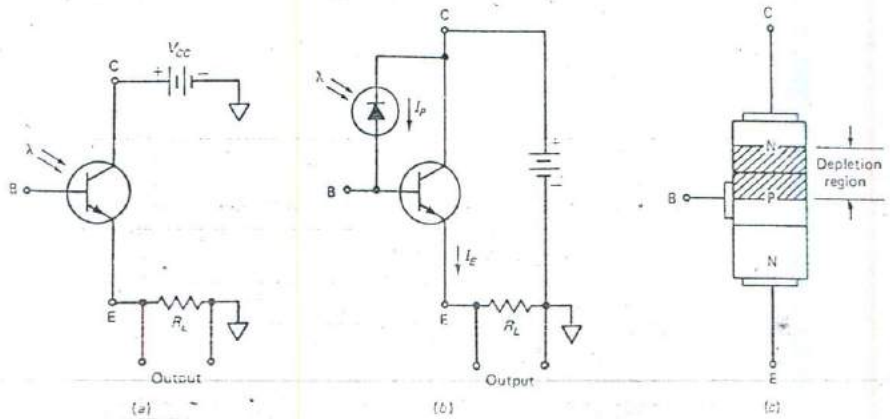


Fig. 14-21 Phototransistor circuit and junction representation. (*a*) Schematic diagram. (*b*) Equivalent circuit. (*c*) Junction representation.

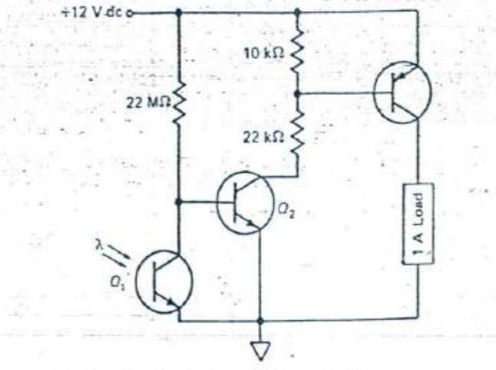


Fig. 14-23 Very-low-level phototransistor circuit.

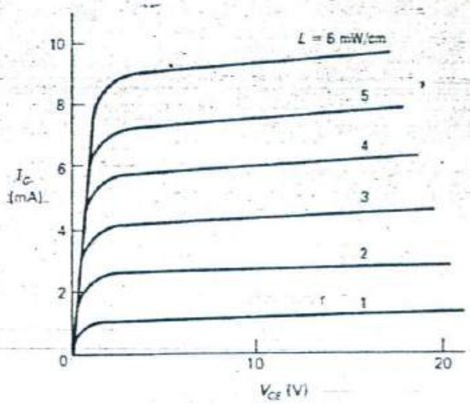


Fig. 14-24 Typical characteristic curves of a phototransistor.

closely approximates V_{be} (Q_2) (one diode drop each), V_{bc} (Q_1) is approximately equal to zero (0). This is now a minimum-leakage condition for the phototransistor.

The light falling on a given area varies inversely with the square of the distance from the source; the relationship between light intensity L (in lumens), and the strength of the source C (in candela) is

$$L = \frac{CA}{d^2}$$

where C = luminous intensity in candela
A = area illuminated in square meters
d = distance from source to area in meters

Figure 14-24 shows typical characteristic curves of a phototransistor. The usual base steps are now intensity steps in lumens.

The *photodarlington* basically is the same as the phototransistor, except for its much higher gain from two stages of amplification cascaded onto a single chip. Figure 14-25 shows the photodarlington symbol, its geometry, and its equivalent circuit. The formula for the photodarlington's emitter current is

$$I_{e2} = I_{p1} \cdot (h_{FE1}) \cdot (h_{FE2}) + I_{p2} \cdot (h_{FE2})$$

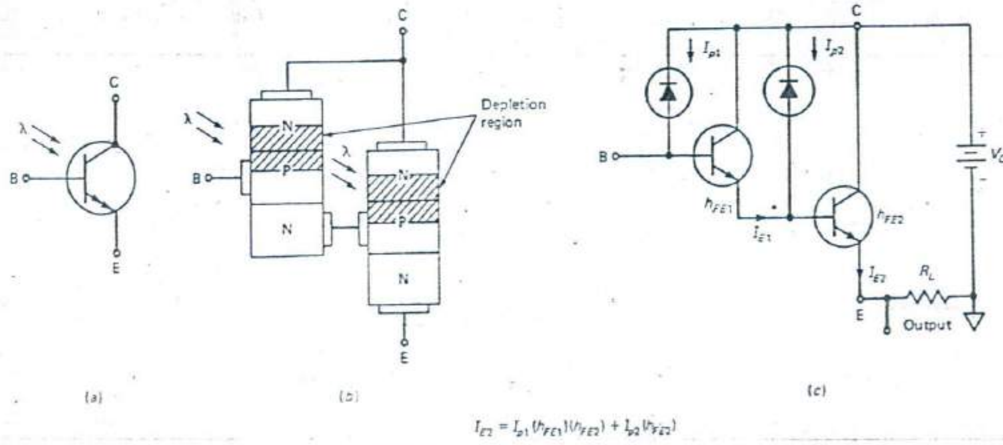
Because $I_{e1} >> I_{p2}$

$$I_{e2} = I_{p1} \cdot (h_{FE1}) \cdot (h_{FE2})$$

where I_e = emitter current, each device
I_p = photon-produced current, each region
h_{FE} = current gain of each transistor, dc

To maximize sensitivity, I_{p1} should contain a large portion of the photon-produced current. This is accomplished by expanding the base of the pellet so the photodiode dominates the topography, in the same way as the phototransistor does in Fig. 14-22.

Optimization of both short- and long-wavelength response (ultraviolet [UV] to infrared [IR]) at low reverse voltage requires a PIN, rather than a PN diode structure. A PIN diode has a thin P-type diffusion in the front and an N-type diffusion into the back of a wafer of very-high-resistivity silicon. The high-resistivity material between the P-type and N-



$I_{E2} = I_{p1}(h_{FE1})(h_{FE2}) + I_{p2}(h_{FE2})$

Fig. 14-25 Photo Darlington (*a*) Symbol. (*b*) Construction. (*c*) Equivalent circuit.

type diffusions is called the *intrinsic region*, or *I layer*.

The silicon planer PIN photodiodes are ultrafast detectors for visible and infrared radiation. Their response to blue and violet is exceptionally good. Speed of response of these detectors can be less than 1 ns. They are particularly well suited for laser pulses, and their bandwidth is from direct current to 1 GHz. Their low dark current or leakage current (usually in picoamperes) results in an almost negligible noise current. This is a great asset for fiber optics (discussed later in this chapter).

The electrical characteristics of the PIN photodiode are shown in Fig. 14-26. Close scrutiny will reveal that the PIN photodiode may be operated as a photovoltaic device (right-hand 2.5-MΩ load line), in which the light impinging upon the unit creates a voltage which varies in proportion to the input light. This mode of operation is shown in Fig. 14-27. Operation in this mode requires no external bias, and the photodiode generates the load voltage when illuminated. This photovoltaic operation can be either linear or logarithmic, depending upon the selected value of the load resistance. *Logarithmic operation* is obtained if the load resistance is very high (greater than 10 MΩ). With an input FET or a BI-FET op amp, this can easily be achieved, as shown in Fig. 14-28(*a*). If the amplifier has a very high input resistance, the loop gain is $(1 + R_2/R_1)$. The speed of response of this type of amplifier is very slow, with a time constant of approximately 0.1 s. If high speed is required, the logarithmic amplifier should be preceded by a linear amplifier.

The more common mode of operation for the PIN photodiode employs reverse bias (as with the PN photodiode), and this mode is called the *photocurrent* or *photoconductive mode*. The main drawback of the photocurrent mode is the flow of dark current, which is due to the reverse bias. As mentioned previously, *dark current* is that current which flows when no radiant flux is applied to the diode. For best linear operation, the photodiode should be operated with as small a load resistor as possible. Figure 14-28(*b*) shows the typical amplifier arrangement. The inverting input of the amplifier is at virtual ground; the dynamic resistance of the photodiode is R_1 divided by loop gain (greater than 10 k). As shown in Fig. 14-28(*b*), the output voltage will rise (go positive) in response to the optical input signal. Reversing both the photodiode and V_c will produce a drop (output voltage will go negative) in the amplifier output for an optical input. The output voltage from this circuit is

$$V_o = R_1 (I_p + I_{\text{dark}})$$

The speed of response is greatly improved and is now limited by the amplifier.

In most discrete-device literature, the photo SCR is often termed a *light-activated SCR* (LASCR). Figure 14-29 presents the schematic symbol, structure, and equivalent circuit of this device. The photon current (I_p) generated in the reverse-biased PN junction (depletion region) reaches the gate region to forward bias the NPN transistor and initiate switching. Since the photodiode current is of a low level, a LASCR must be constructed so that it can be triggered with a very low gate current. The high sensitivity of the LASCR causes it to be sensitive also to any effect that will produce an internal current. Therefore, it is very sensitive to temperature,

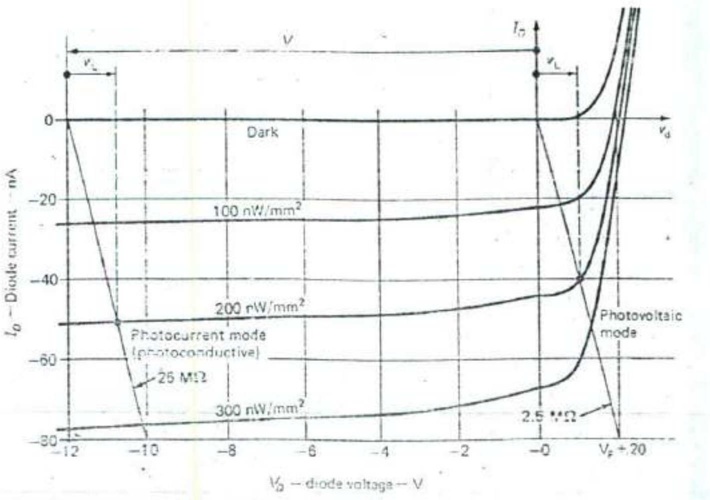


Fig. 14-[illegible] [illegible]ctrical characteristics of a PIN photodiode.

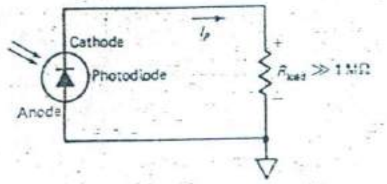


Fig. 14-27 PIN photodiode in photovoltaic mode.

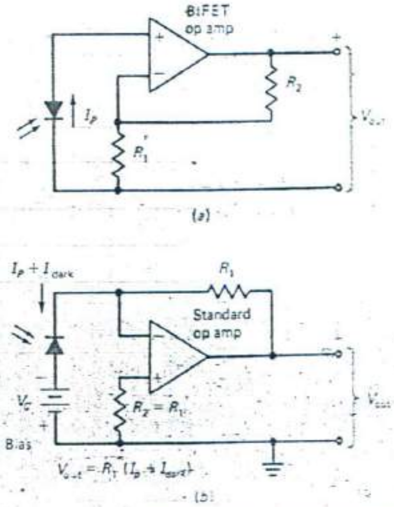


Fig. 14-28 Two modes of PIN photodiode operation using op amps

applied voltage and rate of change of applied voltage (dv/dt), and it has a longer turn-off time than a typical SCR. All other parameters of the LASCR are similar to those of an ordinary SCR, so that the LASCR can also be triggered with a positive gate signal. Most commercially available LASCR types are of comparatively low current rating (less than 2 A). In practical applications they are used to drive a higher-power device, usually another SCR.

There are many other photodetector combinations using integrated circuits that allow many combinations. Among the possibilities are FETs, triacs, logic devices, and MOSFETs. These will be examined in the optical coupling section.

Another basic type of photosensor uses the photoconductive bulk effect. The photoconductive bulk effect cells are normally made of cadmium sulfide (CdS) or cadmium selenide (CdSe). Unlike the previous types discussed, they have no junction. The entire layer of material changes resistance when illuminated. The photoconductive cell decreases in resistance as the light level increases and increases in resistance as the light level decreases. The response curve of a cell is shown in Fig. 14-30. The response curve of resistance versus intensity (lumens) is nonlinear; however, over a limited range of intensities, a linear approximation is valid. Figure 14-31 shows the electrical symbol along with a drawing of the usual configuration of the cell.

Although photoconductors require an external power supply, a sensitivity 1000 times greater than that of the photovoltaic types more than compensates in most applications. Because of their low cost, a pair can be used for temperature compensation in

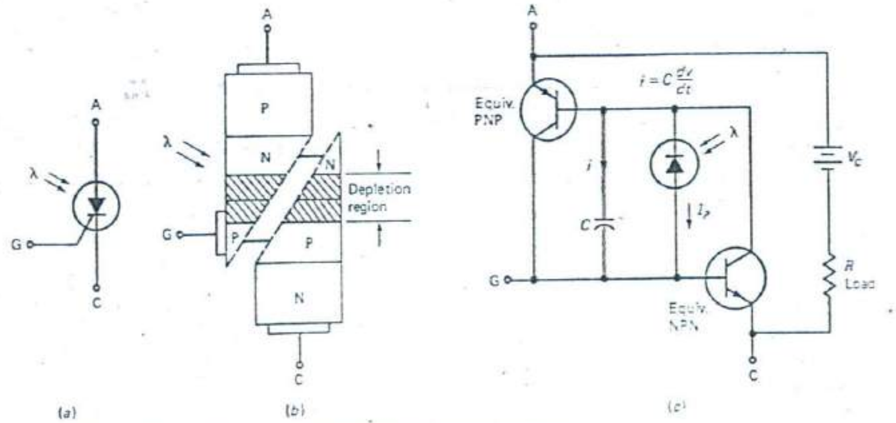


Fig. 14-29 Photo SCR (LASCR). (a) Symbol. (b) Construction. (c) Equivalent circuit.

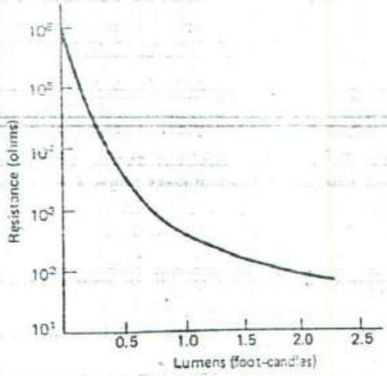


Fig. 14-30 Photoconductive cell's characteristics.

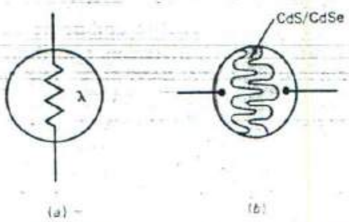


Fig. 14-31 Photoconductive cell. (a) Symbol. (b) Physical layout.

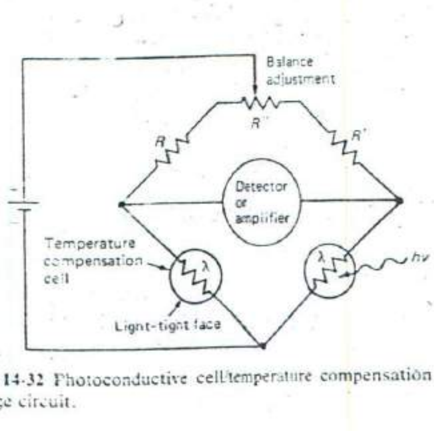


Fig. 14-32 Photoconductive cell/temperature compensation in a bridge circuit.

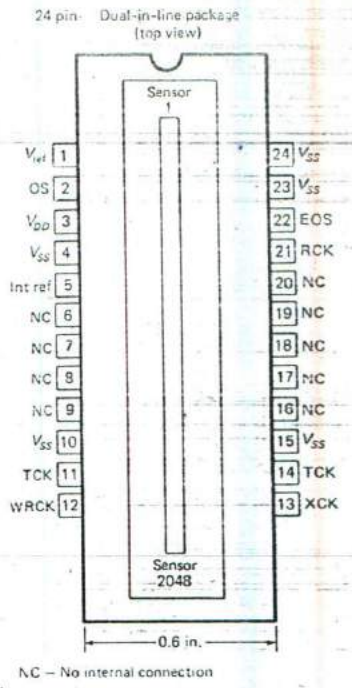


Fig. 14-33 Image sensor chip with 2048 elements.

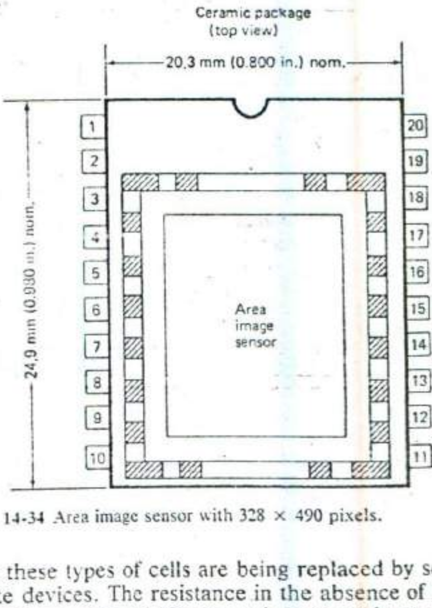


Fig. 14-34 Area image sensor with 328 × 490 pixels.

a Wheatstone bridge arrangement. One cell is shielded from light so that its resistance is only a function of the surrounding temperature; the other cell is placed in the opposite arm of the bridge to measure light intensity, as shown in Fig. 14-32.

The inherent nonlinearity of the photoconductive cell has led to its widespread use in chopping circuits, but these types of cells are being replaced by solid-state devices. The resistance in the absence of light usually exceeds 1 MΩ; the resistance under a given light condition can be as low as 100 Ω. The small-size cells find extensive use in light-meter circuits. The CdS cell has a drawback, however. If exposed to a very strong light (saturated), it displays a "memory" and may take hours to return to the original "dark" resistance.

Many industrial control systems use punched cards (paper, plastic, metal) or punched tape to input programs or data. To meet these requirements, many arrays usually containing 9 or 12 devices per unit are available. The device may include some logic, along with a biasing network. A precise matching emitter array (IRED) is usually available to allow proper alignment. These devices are being replaced by diskettes, magnetic tape, and cards.

The solid-state image array sensors (line scanners) include anywhere from 64 to 10,000 individual photodiodes, which are scanned to provide serial output on a single video line. Applications include pattern recognition, size and position monitoring, and inspection. A 2048 by 1 linear image sensor is shown in Fig. 14-33. The sensor elements are also called *pixels*, which stands for *picture elements*. Their operation is based on charge-coupled devices, transfer gates, and drive circuitry. The area image sensor of Fig. 14-34 has 328 horizontal and 245 vertical pixels for camera operation. High-resolution sensors with 1024 by 1024 pixels are being developed for robot vision systems and as replacements for vidicons and other tube-type image sensors.

REVIEW QUESTIONS

8. Photon absorption results in the formation of a hole-electron ______ in a PN junction.

9. Recombination will occur when the pair is formed in an ______ field.

10. The electric field is greatest in what region of the PN junction?

11. Short-wavelength photons will penetrate the entire crystal. (true or false)

12. An avalanche photodiode uses ______ bias.

13. The avalanche photodiode is used mainly for ______ detection.

14. Leakage currents are the main contributors to ______ current in a photodiode.

15. The phototransistor must have its base forward-biased. (true or false)

14-3 DISPLAYS

Incandescent, fluorescent, and neon lamps have been in use for a long period of time and in a wide variety of applications. In recent years, the solid-state LED has replaced these earlier lamps in many applications. Most new designs and applications use the LED indicators and displays. Therefore, it is important to know and understand the properties and characteristics of the LED-type devices.

Most commercial LEDs are manufactured by encapsulating the diode chip inside a plastic package with an immersion lens directly above the junction. The immersion lens arrangement results in a light output which is concentrated in a narrow beam and applies only to LEDs in undiffused plastic. This configuration is used for backlighting applications and those applications requiring a concentrated light source.

A front panel indicator requires a wide off-axis viewing angle. To obtain this wide viewing angle, a diffusant is added to the plastic to disperse the light rays being emitted. Dye coloring is also added to tint the plastic to enhance its on/off contrast.

As mentioned earlier, not every photon generated within the PN junction emerges to the surface. The GaAsP/GaP devices have an efficiency of approximately 80 percent. When light passes from a medium whose index of refraction is N_1 to a medium whose index of refraction is N_2, a portion of the light is reflected back at the medium interface. This loss of light is called *fresnel loss*. Without going into the physics of lens construction, it will suffice to say that coating the lens with an intermediate material with a suitable index of refraction (say, N_3) reduces the fresnel loss to permit almost 98 percent efficiency.

The LED packages take on many physical shapes to meet a wide variety of applications. They may be rectangular, round, dot-shaped, or the size and shape of typical incandescent lamps. Some units may have additional components integrated inside the LED package. An example is the *resistor LED indicator*. The integral resistor is a nominal 215 Ω, allowing the lamp to be driven directly from the 5.0-V supply for a logic gate. Otherwise, an external current-limiting resistor will be required. Some LEDs can produce red or green light from one package. They usually are equipped with a milk-white diffused lens to provide good on-off contrast. They have three leads and two anodes (one for red and one for green) with a common cathode. This device is used in applications that require an indicator for the three states of interest and provide more information than an on/off lamp. Some applications require that analog information be converted into a visual light display. This has typically been done with a panel meter. Many analog displays require only moderate accuracy and resolution. Light-emitting diode bar/graph modules can often be substituted for meters in these applications. The LED bar/graph displays consist of a linear array of LEDs that are driven by a device that decodes an analog or digital signal into a bar/graph indicator display code. Figure 14-35 is an example of

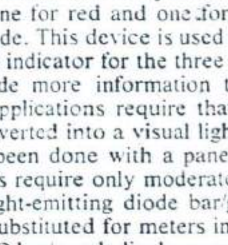
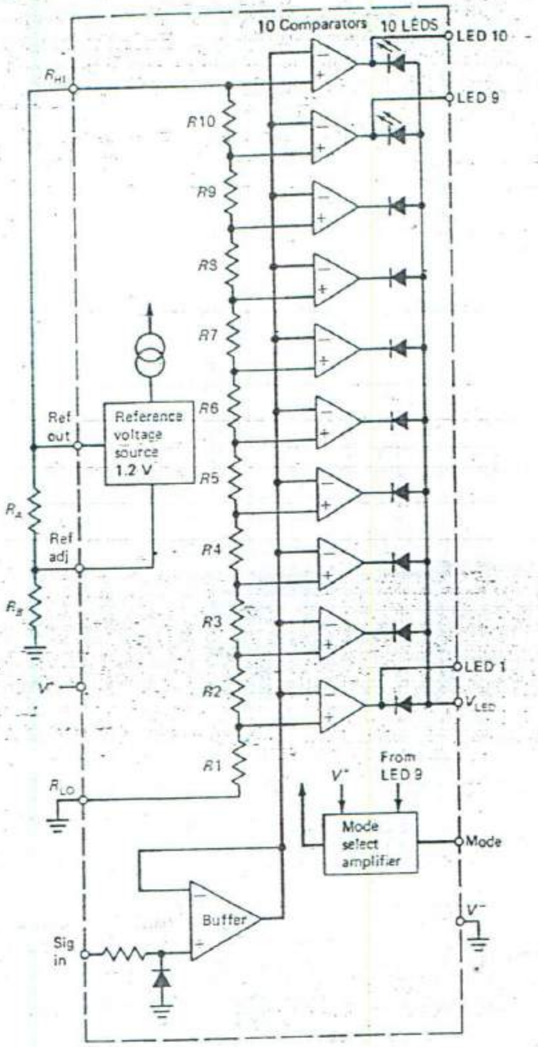


Fig. 14-35 DIP bar/graph packages.



Typical resistor string values (ohms)

Resistor	Linear	Log	VU
R1	1.00 kΩ	1.0 kΩ	0.708 kΩ
R2	1.00 kΩ	0.41 kΩ	1.531 kΩ
R3	1.00 kΩ	0.59 kΩ	0.923 kΩ
R4	1.00 kΩ	0.83 kΩ	0.819 kΩ
R5	1.00 kΩ	1.17 kΩ	1.031 kΩ
R6	1.00 kΩ	1.66 kΩ	1.298 kΩ
R7	1.00 kΩ	2.34 kΩ	0.769 kΩ
R8	1.00 kΩ	3.31 kΩ	0.864 kΩ
R9	1.00 kΩ	4.69 kΩ	0.970 kΩ
R10	1.00 kΩ	6.63 kΩ	1.087 kΩ
Total	10 kΩ	22.6 kΩ	10 kΩ

Fig. 14-36 Integrated bar/graph display.

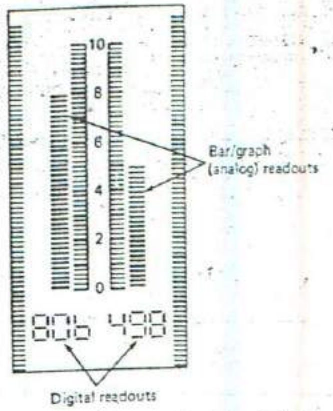


Fig. 14-37 Analog digital meter with bar/graph display.

a bar/graph display. Most units are fabricated to be stackable. In some cases, 10 arrays are cascaded to form a bar graph of 100 elements. The totally integrated units contain drivers, a reference, and threshold detectors and may be programmed to be linear or logarithmic or to display volume units (VU). Figure 14-36 illustrates the block and connection diagrams for an LED bar/graph display with driver. The device can produce a bar/graph display (histogram) or a moving-dot display, depending on the mode pin. Some units can produce an automatic color change (typically green to red) to indicate overrange, overload, or some unsafe condition. A combined analog and digital readout is very desirable in many instances, such as in process control. Such a dual-type readout display is shown in Fig. 14-37.

The LED display devices have expanded very rapidly and now include a wide variety of distinctly different products. Four generalized categories are the following:

1. Displays with on-board integrated circuits
2. Strobable seven-segment displays
3. Dot matrix alphanumeric displays
4. Magnified monolithic displays

All the displays use the GaAsP type of diode and can offer colors of red, green, or yellow. The most prominent feature of any display is the physical arrangement of the display elements. This arrangement, or *font* as it is usually called, is important for the type of information to be displayed but also dictates the type and complexity of the driving electronics required by the display. Some of the common display fonts are illustrated in Fig. 14-38.

The seven-segment display (font A) is the most commonly used LED technology and is also the easiest to implement electronically. However, it is limited to displaying numeric and a small range of al-

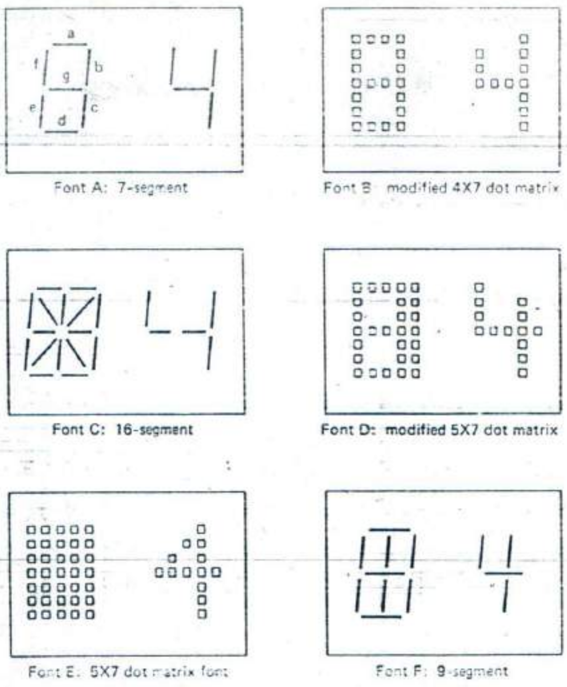

Font A: 7-segment
Font B: modified 4X7 dot matrix
Font C: 16-segment
Font D: modified 5X7 dot matrix
Font E: 5X7 dot matrix font
Font F: 9-segment

Fig. 14-38 Typical LED display fonts.

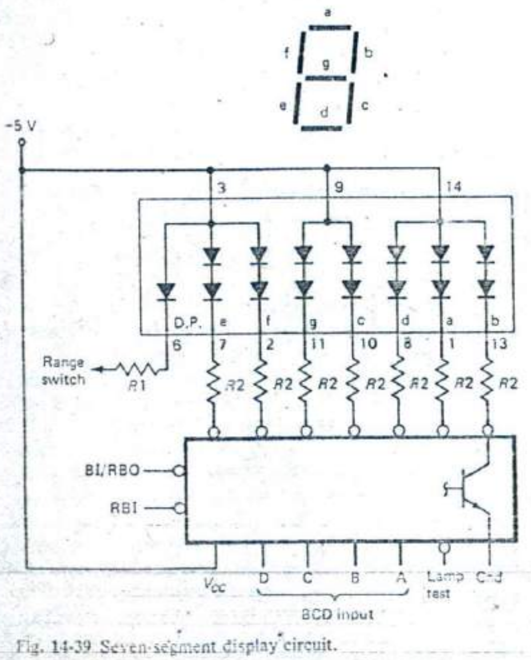


Fig. 14-39 Seven-segment display circuit.

phabetic information. The five by seven dot matrix (font E) can display a wide range of numeric, alphabetic, and other characters but requires rather extensive electronic circuitry for implementation. The sixteen-segment display (font C) has full alphanumeric capability but has found only limited acceptance. Font F illustrates a nine-segment display, which is somewhat more pleasing and readable than the seven-segment display of font A. It can display the same numeric information and has more alphabetic capability. The dot matrix fonts of B and D are abbreviated versions of the 35-dot matrix of font E and are used primarily for displaying numeric and hexadecimal information.

The interface to a seven-segment display is provided by the BCD to seven-segment decoder driver shown in Fig. 14-39. The input to the decoder is BCD code for the number to be displayed. The RBI and BI can be pulled low to turn off all segments. When BI is high, the lamp test (LT) input can be brought low to turn on all segments to perform a lamp test operation. The BI/RBO can serve as an output for *ripple blanking* (blanking nonsignificant zeros) to other decoders. When RBI is low, the RBO output will go low to ripple a blanking signal to other display decoders. The segment drivers a through g and D.P. (decimal point) are connected to the display to control which LEDs are to be turned on.

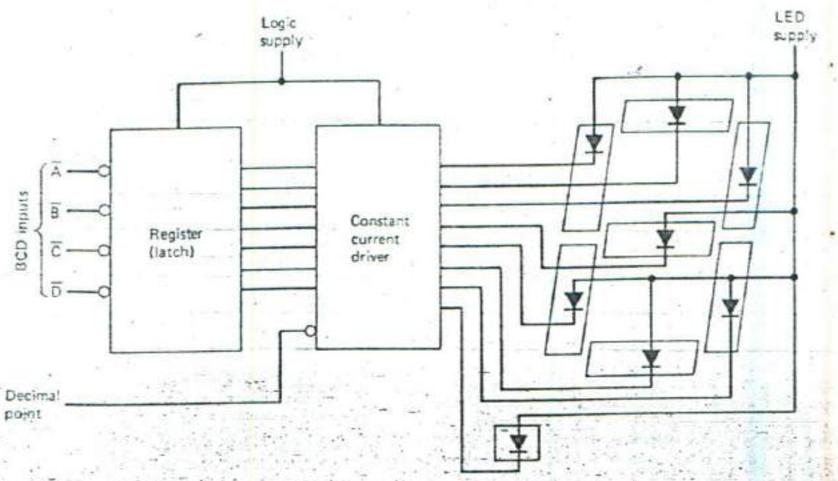


Fig. 14-40 Single-chip seven-segment displays.

The entire circuit and display are also available as a single device, as shown in Fig. 14-40. This device has a 4-bit BCD code input and decimal point input. Devices that include a storage register (latch) as well as a decoder/driver and display unit in the same unit also exist. The display will show the number whose code is latched. A latch simplifies the input/output (I/O) requirements of a microcomputer because the display can be treated as a memory location. The display with latch may be connected to the data bus or to an I/O bus.

The interface to a five by seven or other dot matrix display is handled in much the same manner as the seven-segment display. A device which displays hexadecimal characters using LEDs arranged on a four by seven dot matrix pattern is shown in Fig. 14-41. It includes a 4-bit data register with a latch strobe to store input data. As long as the strobe stays high, the information displayed (stored) will not change. There is a blanking input that, when high, causes the display to be blanked. There are also left and right decimal points available. In some cases, the LED power supply is different from the logic supply, which must exhibit low ripple and good regulation. These stringent requirements are not necessary for the LED display power.

The control of a five by seven dot matrix display usually requires a read only memory (ROM) or EPROM in which the display pattern for each character to be displayed is stored. The basic circuit structure is illustrated in Fig. 14-42 for an individual interface to a five by seven matrix LED. The ASCII code for the character to be displayed is applied to the seven inputs. The current drive capability is provided by the seven row sink drivers (on the O_1 to O_7 lines) and the four source drivers for the column lines of the matrix. At the time a column line is turned on, the column select CA through CE must simultaneously be applied to the column select lines of the EPROM. The EPROM outputs the appropriate row signals for each selected column character. Thus, the circuitry must scan through the columns at an appropriate rate with a ring counter or some other counter-decoder combination. Only one column of the EPROM is addressed at any time. The UJT clock is set to provide a clock rate of 1 kHz. A new column

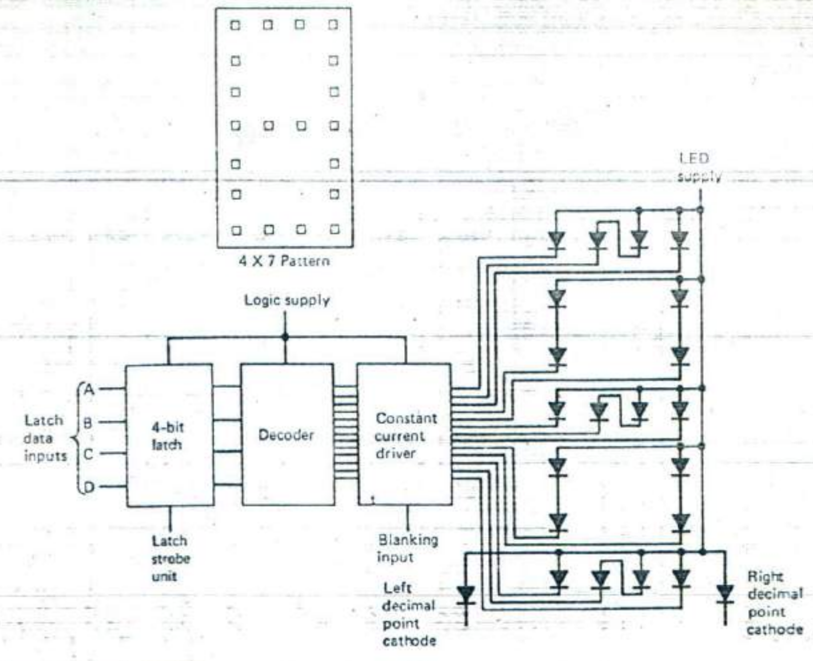


Fig. 14-41 4 × 7 matrix circuit structure.

is selected and turned on for about 1 ms, and any given column is on 20 percent of the time. The circuit of Fig. 14-42 displays a single character. If a multiple-character display is necessary, it is not reasonable to provide a separate EPROM for each display unit. A circuit that shares the EPROM must be used. A random access memory (RAM) can be used to provide storage of the character codes to be displayed, along with a controller that will sequence through the ASCII codes stored in the RAM while different matrix displays are activated.

As shown in Fig. 14-43, the *liquid crystal display* (LCD) has two glass plates, the insides of which are coated with a transparent pattern of conductive materials. The plates are mounted so that the conducting layers are facing each other. The spacing between the two plates is only about 10 to 30 μm (25 μm = 0.001 in.). Liquid crystal materials are retained between these plates by a peripheral seal of glass frit, or epoxy. Both outer surfaces of the front and rear glass require light-polarizing films. Ordinary light is composed of vertical and horizontal components; polarizing it removes one of the components. The polarizing film on the rear glass is covered with a reflective material (silver bead, silver foil, or gold foil) or a transreflective material (reflects ambient light and transmits back light). The reflective silver foil type is the most popular.

Liquid crystal displays are activated by applying voltage between the segment and the common electrodes shown in Fig. 14-43. Typical driving voltage is 5 V_{rms}, and the allowable ac frequency range of the driving voltage is from 30 to 100 Hz. Flicker may be seen if the drive frequency is below 30 Hz. The power consumption increases in direct proportion to the driving frequency, and a driving frequency of 100 Hz is typically used. The LCDs are driven with ac voltages to prevent plating of the conductive electrodes due to electrolysis. Direct current driving, or ac driving with a large dc offset, greatly shortens the life of the LCD. Strict attention is necessary in order not to exceed this specified dc offset, typically 25 mV. Usually LCDs are connected to logic circuits, so it is very common to drive them by using an ac symmetrical square wave. This ac symmetrical square wave features less dc offset and can be obtained in all LCD drivers by using an exclusive OR gate as shown in Fig. 14-44. Waveform C is the oscillator signal and is applied to the gate and to the common connection on the display. The D waveform is the output of exclusive OR, which has shifted the oscillator input 180°, when the control input is high. Waveform E is the resultant waveform of C and D and is the one that in effect drives the display. Many LCD drivers include exclusive OR gates. The 10 V between the segments and the common electrode is obtained from the 5-V logic power supply.

The LCD display mates very well with CMOS-type logic. Decoder, driver, and interface circuits are available to use with LCD devices. Typical seven-

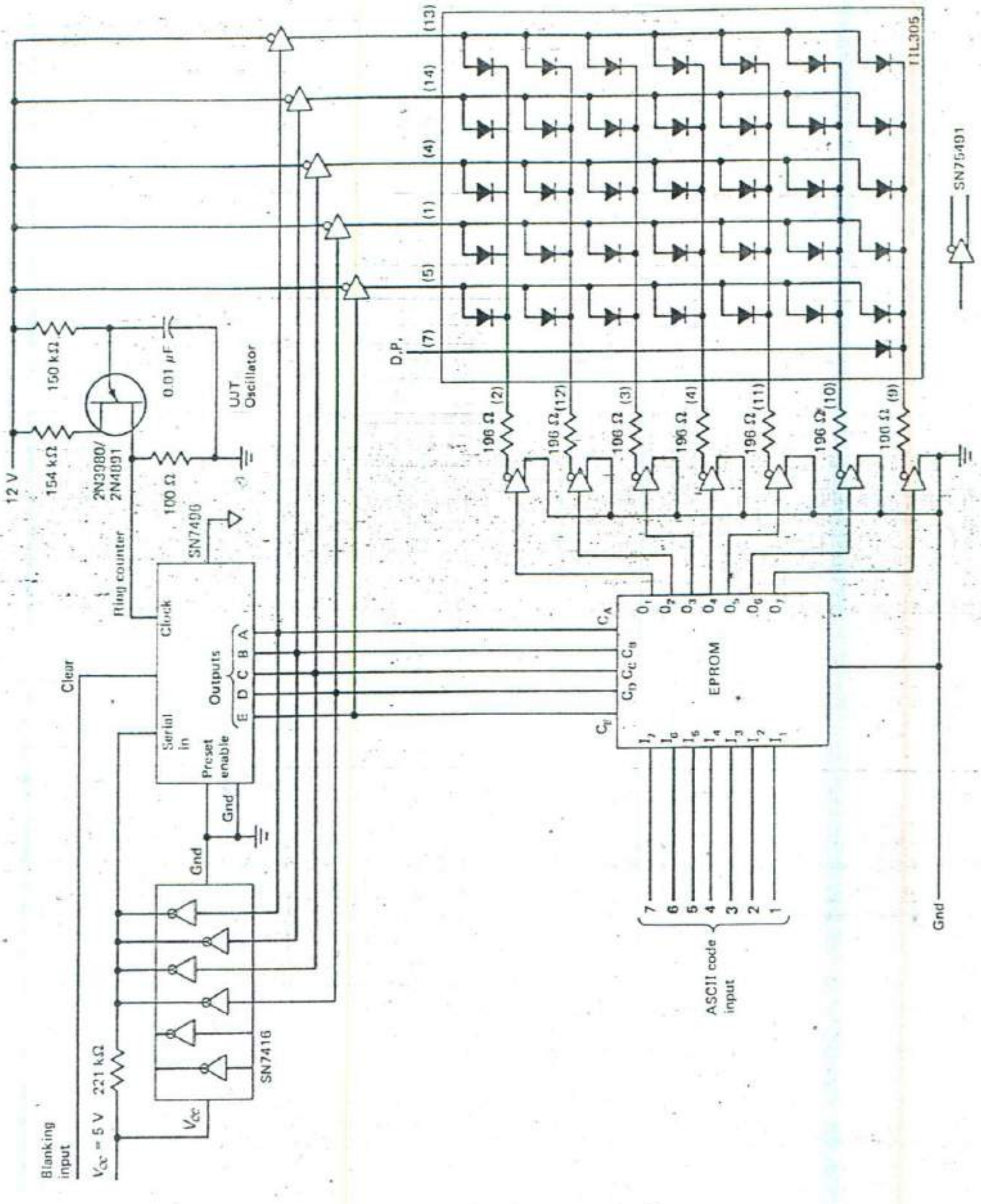


Fig. 14-42 5 × 7 dot matrix circuit structure.

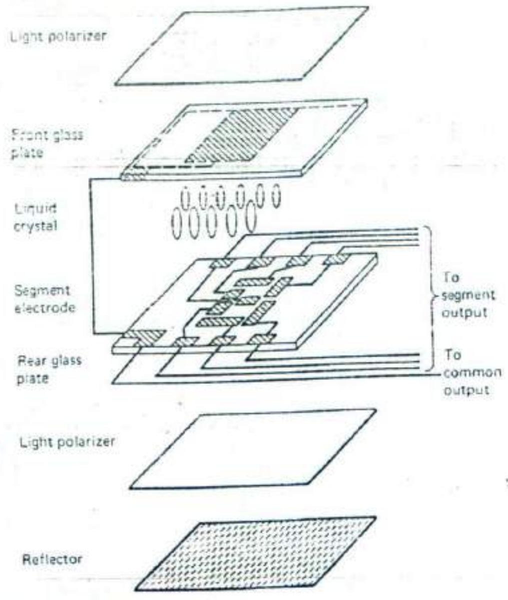


Fig. 14-43 LCD construction.

segment numeric display circuits are illustrated in Fig. 14-45.

Liquid crystal display devices require backlighting if they are to be used in a darkened area or at night. They are very sluggish at low temperatures, and in some applications heaters (back-plane type) are added or integrated into the display when it is fabricated. Flat-panel displays are available for viewing real-time information. This type of display need has previously been dominated by the cathode-ray tube, but three flat-panel technologies are providing alternative choices. They are the electroluminescent, vacuum-fluorescent, and plasma-gas discharge techniques.

Electroluminescent panels use certain solids which emit light when an electric field is across them. They can display 80 characters by 25 lines on a 4- by 8-in. active area. They must be refreshed (60-Hz rate) to retain the image. High-voltage drive (usually 200 V) is required for a bright display, and this requirement limits its popularity.

The *vacuum-fluorescent* (VF) display technology makes use of a low-temperature filament cathode to produce electrons that strike the phosphor-coated anode in a vacuum envelope with a wire mesh grid. The VF display competes with the other technologies for the display market. It is very durable (100,000 h) and less subject to contamination and dirt, and it produces a visible display in bright lighting or total darkness. As VF displays increase in size and capacity they become difficult to produce. Their cost goes up with the necessary drivers, and makes their price almost as high as that of a CRT. Therefore VF display applications have been limited to small-size readouts.

Figure 14-46(*a*) illustrates a VF four-digit display circuit driven from the TTL output of a seven-segment decoder. Usually V+ is 60 to 70 V; for this reason the high-voltage driver chips are employed as

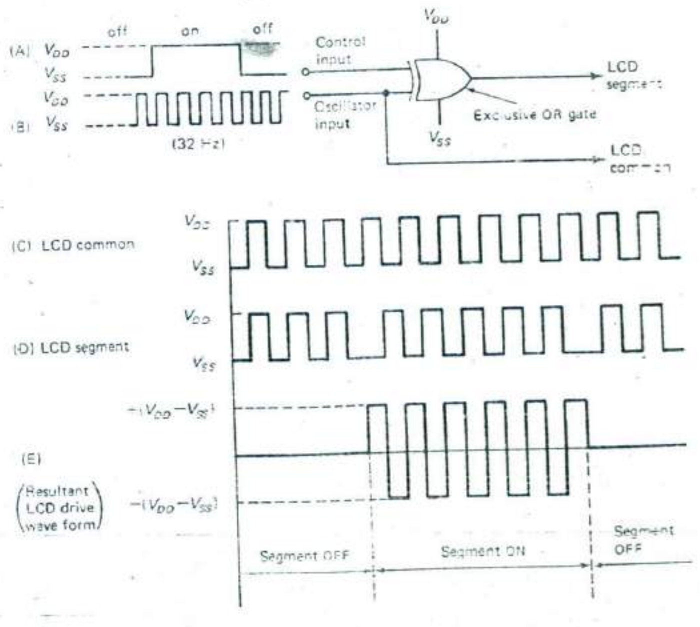


Fig. 14-44 LCD drive circuit and waveforms.

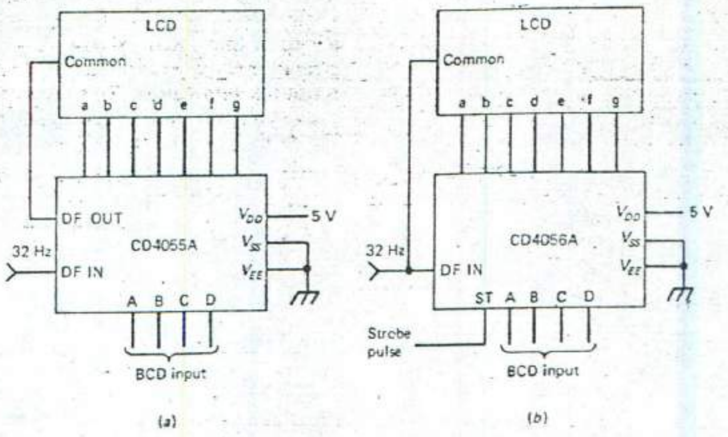


Fig. 14-45 LCD seven-segment decoder equivalents. *(a)* Non-latched type. *(b)* Latched type.

an interface between the display and the decoder outputs.

When both the *segment* (equivalent to the vacuum tube anode) and the *digit* (controlled by the grid) are switched sufficiently positive with respect to the cathode (filament), the electron cloud around the cathode will be accelerated by the grid and continue on to the anode (grid is meshed and the electrons pass through), where the digit/segment will become fluorescent. These displays are typically blue-green in color because of the phosphors employed in their construction. Filters can be added to display virtually any color. A CMOS driver that uses a bipolar 12-V supply is shown in Fig. 14-46(*b*). The driver is compatible with 6- to 15-V CMOS logic signals. In this type of drive the substrate must be connected to a voltage equal to or greater than V_{dd}, so while V_{dd} is 12 V the substrate and the output are tied to the most negative rail.

Plasma panel displays are a cousin to the gas discharge display tube of past years. A brief look at the gas discharge tube will aid in your understanding of the plasma panel type of displays that are in use today. The gas discharge display uses neon gas, which produces an orange glow when ionized. This type of tube is a *cold-cathode* (no heated filaments) device. It consists of a common anode and some number of individual metallic cathode elements to form the characters. Application of a negative voltage to the selected cathode elements with respect to the common anode creates a current flow. The selected cathodes glow and form the character. The equivalent circuit for the gas discharge tube is shown in Fig. 14-47. The tubes require a minimum cathode current density to assure complete glow of the entire element. A maximum current limit is established to provide a long life. Typical drive circuits are shown in Fig. 14-48. The anode resistor acts as a current limiter. A B+ voltage of 170 V is typical.

Plasma displays also use a neon gas mixture and emit an orange glow when switched at high frequencies. Their light output intensity is a function of frequency. They are sometimes called *ac plasma displays* because they operate from a toggled dc supply (usually around 20 kHz). The panel is basically a neon-filled capacitor. It has plates (electrodes) which are covered with a dielectric. The ac type has an inherent memory and therefore eliminates the need for refreshing. A cutaway view of a plasma display is given in Fig. 14-49. The dc plasma displays have no dielectric, and the gas is not separated from the electrodes. The gas therefore glows with application of a dc signal and goes off when that signal is removed. Although this operation requires simpler drive electronics than the ac version does, screen refreshing is required to keep the pixels lit. Alphanumeric plasma display panels are available in one- to four-line versions with 24 characters per line and characters that are about 0.5 in. high. Some panels utilize a self-scan scheme and significantly cut the complexity of the drive circuit. Displays with 120,000 addressable pixels arranged in a 250- by 480-pixel grid offer an alternative to the CRT for many display applications.

Although hot wire readouts are incandescent devices, their application in multidigit, multiplexed display systems closely resembles LED operation. Since hot wire displays will conduct current in either direction, isolation diodes are required to prevent "sneak" paths from partially turning on unaddressed segments. The hot wire readouts are available in both 7-segment and alphanumeric (16-segment) versions and are quite well suited for high-ambient-light applications. They do not wash out in sunlight. Multiplexed schemes can be cumbersome because of the large number of discrete diodes required.

Though flat panels have advanced rapidly, the CRT is at the forefront for high-resolution displays, both

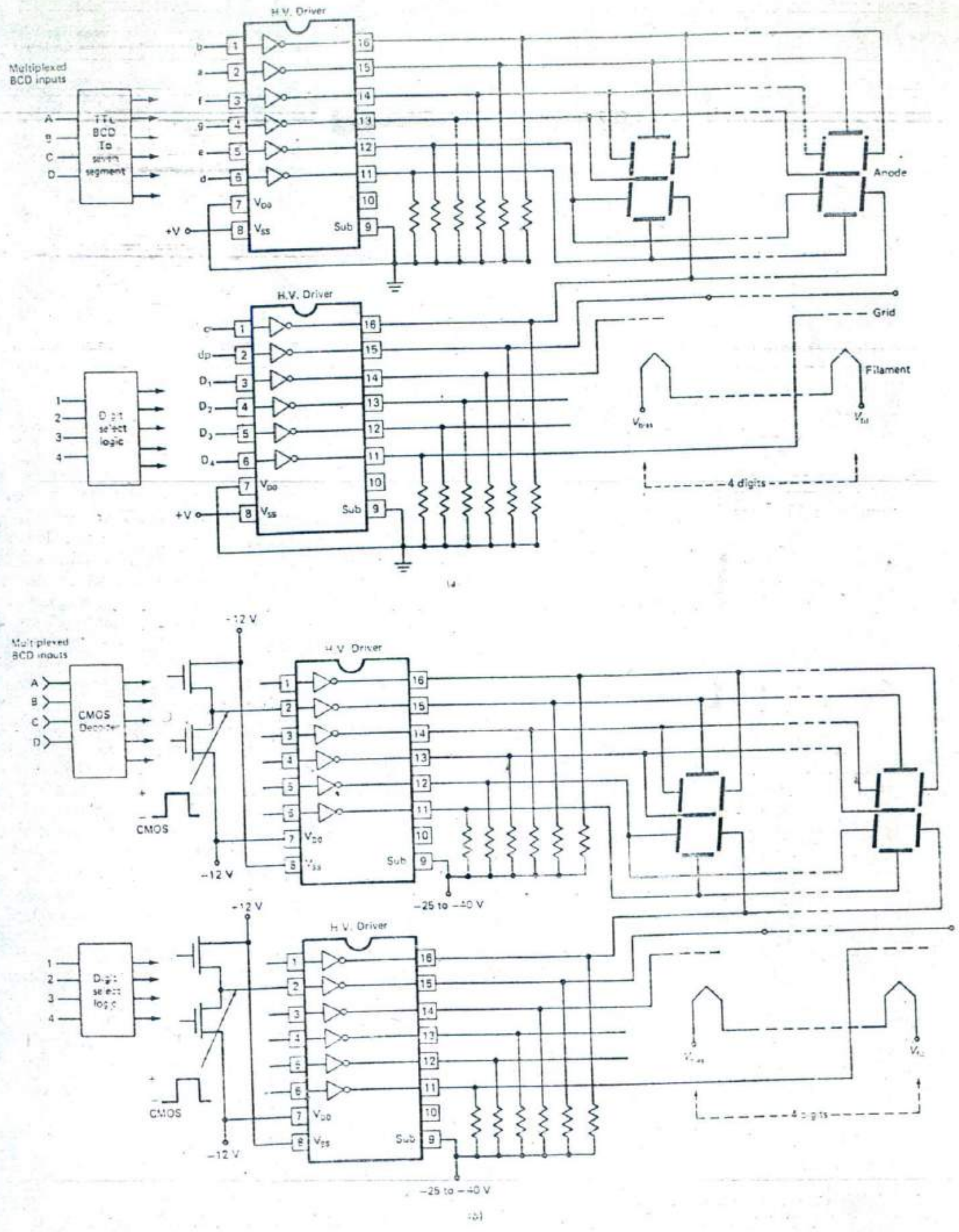


Fig. 14-46 Vacuum fluorescent display circuits.

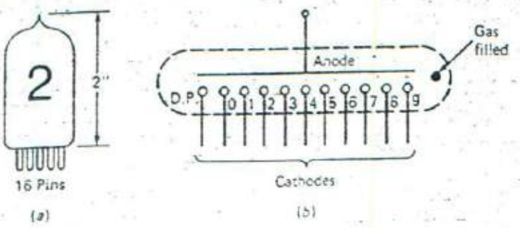


Fig. 14-47 Gas discharge tube. (*a*) Outline. (*b*) Schematic diagram.

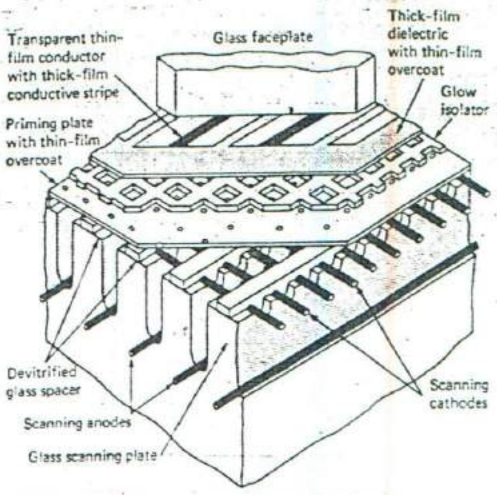

Transparent thin-film conductor with thick-film conductive stripe
Glass faceplate
Thick-film dielectric with thin-film overcoat
Glow isolator
Priming plate with thin-film overcoat
Devitrified glass spacer
Scanning anodes
Glass scanning plate
Scanning cathodes

Fig. 14-49 Cutaway view of plasma display panel.

monochrome and color. Computer graphics and engineering and image processing will be handled by CRTs for many years to come.

The *CRT* is a vacuum tube in which electrons are produced at a heated cathode, then directed to a phosphor coating on the tube face. The electrons striking this coating produce light. The several parts of the CRT are diagrammed in Fig. 14-50. The coated cathode is heated by the filament (heater). Electrons are boiled off and are directed toward the screen at the opposite end of the tube. The electrons are compacted into a narrow beam so that a point, rather than a diffuse spot, is obtained at the screen. The beam is electrostatically focused by succeeding stages and is acted upon by the deflection plates. These plates produce an electrostatic field that follows the input signal. With the correct signal, it is possible to deflect the beam anywhere on the face of the CRT to display characters, lines, or pictures. The accelerating section of the CRT is necessary to give sufficient energy to the beam that the electrons will produce phosphorescence when they strike the screen. The magnitude of the voltage in the accelerating section is of the order of kilovolts and must be uniform in space so as not to distort the beam. The beam may also be deflected magnetically instead of by using the deflection plates as illustrated.

The block diagrams of the principal parts of a CRT imaging system are shown in Fig. 14-51. Figure 14-51(*a*) shows the conventional CRT, where events must be viewed and refreshed constantly. Figure 14-51(*b*) illustrates a storage-type CRT that allows a display to be retained for long periods of time (up to an hour or more). This type of display (storage) was commonly used before inexpensive semiconductor memory was available. Oscilloscopes use CRTs to display electronic waveforms in real time. Computer terminals use them to display alphanumerics and graphic images. Television receivers use them to display pictures. All in all, the CRT is probably the most versatile of all display devices. There are two basic ways to produce images on the CRT screen. One is a *vector display*. To draw a box, the beam is deflected from corner to corner. This process is repeated over and over, and the box appears on the screen. Or if the CRT is of the storage type the box need be drawn only once. Vector displays are normally limited to high-resolution screens used in computer-aided design and drafting.

The other way to draw on the screen is through a *raster display*. The electron beam is continuously scanned from left to right and from top to bottom. When the bottom right is reached, the process repeats by returning to the upper left. If the beam is

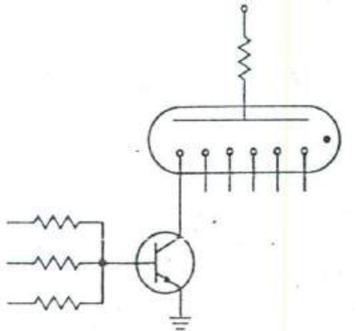


Fig. 14-48 Typical gas discharge display drive circuits.

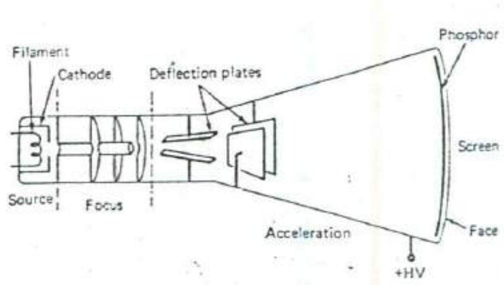
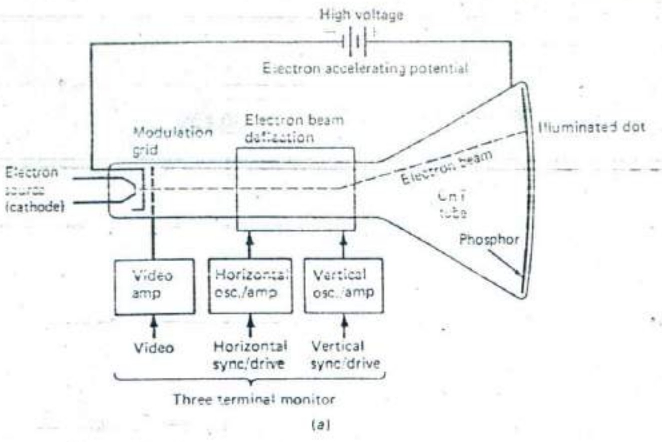


Fig. 14-50 Cathode-ray tube.



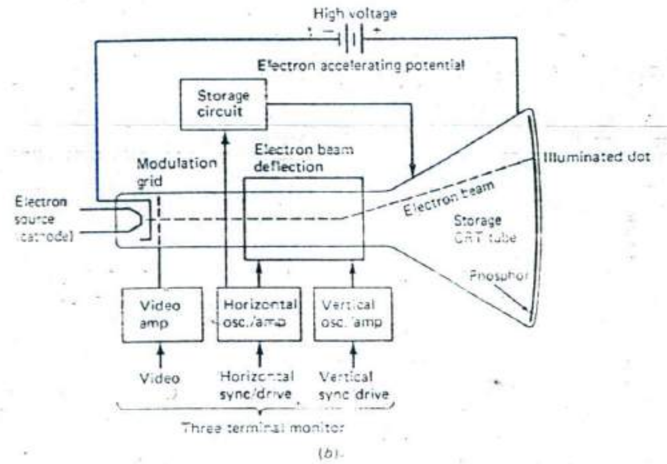


Fig. 14-51 CRT monitor block diagrams. *(a)* CRT monitor. *(b)* Storage CRT monitor.

allowed to remain on, the entire screen lights up. However, if the beam is turned off and on at the proper times, a box, or almost any other character or picture, can be drawn.

Industrial terminals and computer displays generally use a 525-line raster display. About 64 μs is necessary to scan one horizontal line and about 17 ms to get from top to bottom. Only half the picture is scanned at a time (262.5 lines). Every other vertical scan fits an additional 262.5 lines in between. This process is called *interlacing* and is done to minimize flicker on the screen. Noninterlaced displays, which draw all 525 lines in a single pass from top to bottom, are also used.

Figure 14-52 indicates what has to happen to the beam in order to display the letter *N* on the screen. With a seven- by nine-character font, the beam must be turned on to display dots in column 1 and column 7 when line 1 is being displayed. Line 3 must have dots appear at column positions 1, 2, and 7. The beam is turned on with information stored in a ROM called a *character generator.* Figure 14-53 shows a portion of a typical character set stored in a five- by seven-character-generator ROM.

In order to have the characters appear stable and in the correct screen position, the ROM information must arrive at the CRT at the correct time. It must be synchronized with the vertical and horizontal scanning circuits. A block diagram that accomplishes this synchronization is shown in Fig. 14-54. The clock circuits are locked to the raster waveforms mentioned earlier. The characters to be displayed are stored in RAM in ASCII form. Note that only 6-bit ASCII code is required to display the uppercase character set. The 6-bit ASCII code is latched and decoded and applied to the ROM. This code acts as

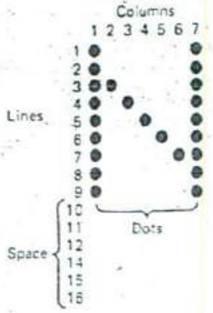


Fig. 14-52 7 × 9 font character example.

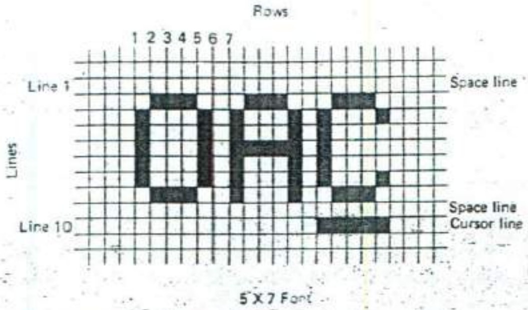


Fig. 14-53 Character font display.

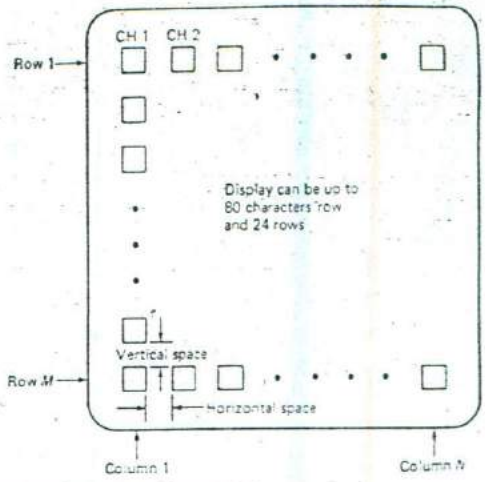


Fig. 14-55 CRT display example.

an address, and the ROM output is one of a possible 64 combinations. The ROM output is multiplexed into a 7-bit shift register and then shifted out to the CRT one dot (pixel) at a time to display one line of the character.

Figure 14-55 shows how the typical industrial CRT display is organized. Generally, there are 24 rows of

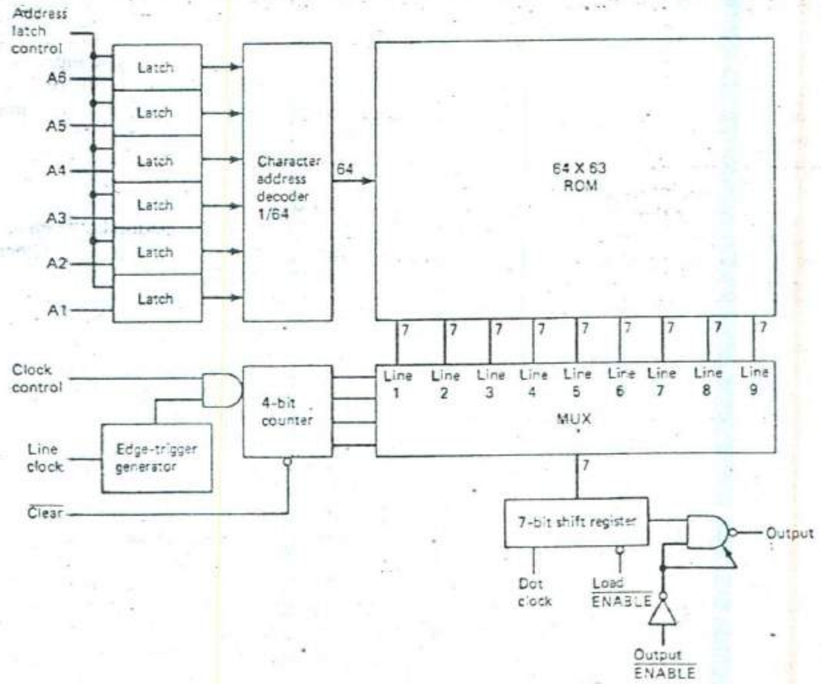


Fig. 14-54 Character generator block diagram.

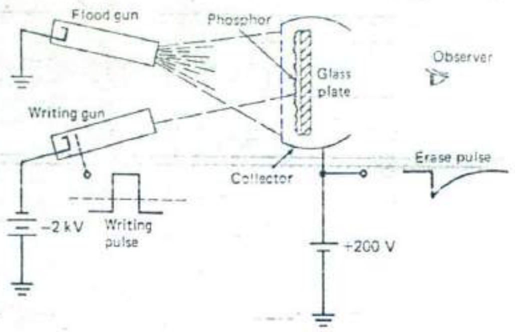


Fig. 14-56 Elementary bistable storage CRT.

80 characters each. Some terminals display 132 characters per row but tend to cause operator fatigue and are not very popular. Raster displays can also do a fair job with graphics. Instead of using character generators, RAM information is clocked out at the dot rate to draw lines, arcs, and irregular curves on the screen. As mentioned earlier, a storage CRT allows a display to be retained for long periods of time. Signals that occur at low repetition rates often cause flickering of the display. Storage allows these signals to be displayed at a constant light level. The typical storage CRT is called a *bistable storage tube*. Its parts are shown in Fig. 14-56, though they are not drawn to scale. Initially, the writing gun is biased off, and the flood-gun cathode is grounded. The collector is fixed at +200 V, and all the targets (phosphors) are at their lower stable point. The flood gun is able to hold each target independently at either of its two stable points (on or off), once they are written or erased to those points.

When the writing beam is gated on and bombards any target (point on the screen), this target charges positive and is now at a stable upper point. It is held at this upper stable point by the flood gun, while the other targets remain at their lower stable states. Any target or targets at the upper state will emit light due to excitation from the flood gun. Thus, once an image is written to the screen, it will remain there for about an hour.

When the erase pulse, Fig. 14-56, is applied to the collector (which acts as a capacitor with the targets), the target voltage drops (as a result of the capacitive coupling), and the screen is reset to its lower stable point. These tubes can store waveforms with nanosecond rise times, but they are very expensive (more than $1000). They are used mainly in storage oscilloscopes and in some computer graphic terminals.

REVIEW QUESTIONS

16. What type of solid-state display would you use for creating a histogram?

17. A BCD to _______ segment decoder is required to operate font A of Fig. 14-38.

18. Liquid crystal displays are driven with _______ voltage to eliminate plating of their electrodes.

19. Liquid crystal displays are used mainly with _______ type logic.

20. A fluorescent display is controlled by signals applied to its _______.

21. The individual dots/spots on a display are known as _______.

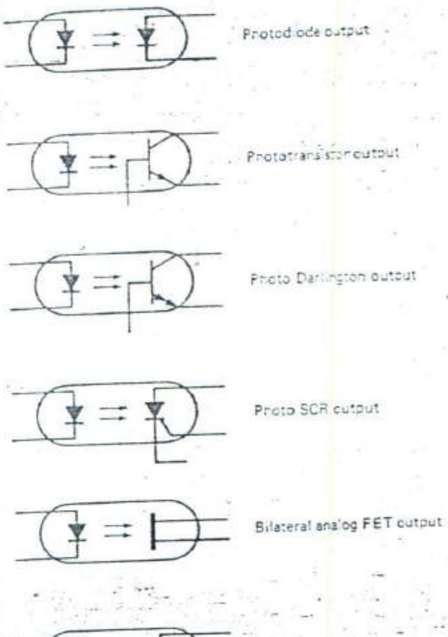
14-4 OPTICAL COUPLING

The *optocoupler*, also known as an *optoisolator*, consists of a photon-emitting device whose flux is coupled through a transparent insulation material to some sort of detector. The photon-emitting device may be an incandescent or neon lamp (earliest models used these) or an LED. The transparent insulation may be air, glass, plastic, or optic fiber. The detector may be a photoconductor, photodiode, phototransistor, photo FET, phototriac, photo SCR, or integrated photodiode/amplifier. Various combinations of these elements result in a wide variety of input characteristics, output characteristics, and coupled characteristics. This discussion will be concerned with optocouplers that use an IRED input with a variety of output detectors. Characteristics such as *coupling efficiency* (effect of IRED current on the output device), speed of response, voltage drops, current capability, and *V-I* curves vary from model to model. The characteristics must be considered especially when performing substitutions. The only common characteristic is that the input is dielectrically isolated from the output. Figure 14-57 shows some of the more common symbols for optoisolators.

The optocoupler was designed as a solid-state replacement for mechanical relays and pulse transformers. Functionally, the optocoupler is similar to its older mechanical counterpart because it offers a high degree of isolation between the input and output terminals. Some of the improvements offered by the solid-state devices are as follows:

Faster operating speeds
Positive (no-bounce) action
Small size
Insensitivity to vibration and shock
No moving parts to stick
Compatibility with many logic and microprocessor circuits
Frequency response from dc to 100 kHz

Isolation is a very important parameter of the optocoupler. The three critical isolation parameters are resistance, isolation capacitance, and dielectric withstand capability. *Isolation resistance* is the dc resistance from the input to the output of the coupler. A value of 10^{11} Ω isolation resistance is very typical;



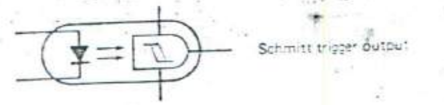


Fig. 14-57 Optoisolator/coupler symbols.

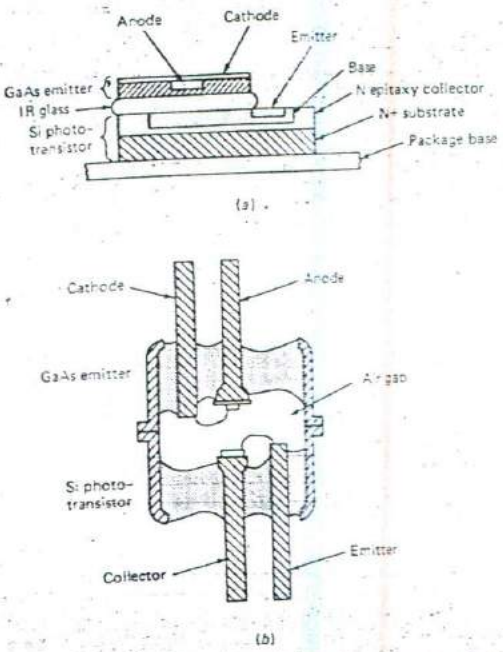


Fig. 14-58 Optocoupler configurations. (*a*) Glass isolated. (*b*) Air isolated.

this value may be higher than the resistance between the mounting pads on many of the printed circuit boards on which the coupler is mounted. Therefore, care in handling of the printed circuit board is required to avoid degrading this parameter. For example, the flux residue from soldering must be removed from the board. Isolation capacitance is a parasitic capacitance from input to output through the dielectric. Typical values range between 0.3 and 2.5 pF. *Isolation voltage* is the maximum voltage which the dielectric can be expected to withstand. Typical ratings are around 1500 V, and special units are available with voltage isolation as high as 50,000 V.

The *input* stage of an optocoupler consists of an efficient GaAs infrared-emitting diode (IRED). The *output* stage of the basic optocoupler is a phototransistor. Two methods of dielectric isolation are shown in Fig. 14-58. Figure 14-58(*a*) has a thin layer of infrared (IR)-transmitting glass between the input and output stages; Fig. 14-58(*b*) uses an air gap to attain greater electrical isolation. Regardless of whether an air gap or IR-transmitting glass is used to separate the input and output circuits, the operating characteristics are basically identical.

The *input* characteristics of the coupler are the same as for the IRED previously discussed in the section on emitters. Typical optocoupler input characteristics are shown in Fig. 14-59. Note that currents from 100 mA to 10 A are permitted only in the pulsed mode. The transfer curve for a typical photodiode coupler is illustrated in Fig. 14-60(*a*) and that of a photodarlington coupler in Fig. 14-60(*b*). The *current transfer ratio* is a common method of cataloging these devices. These ratios are given as percentages. They range from 10 to 80 percent for the phototransistors. For example, if the input is 10 mA,

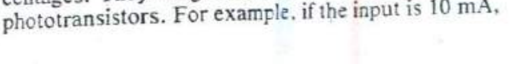


Fig. 14-59 Optocoupler input characteristics.

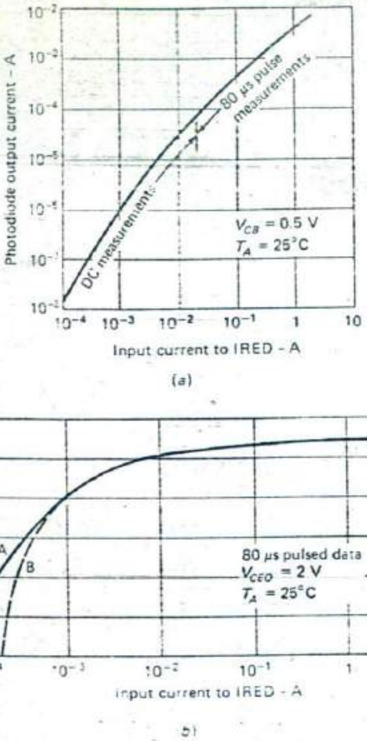


Fig. 14-60 Phototransistor/Darlington optocoupler transfer curves. *(a)* Transistor. *(b)* Darlington transistor.

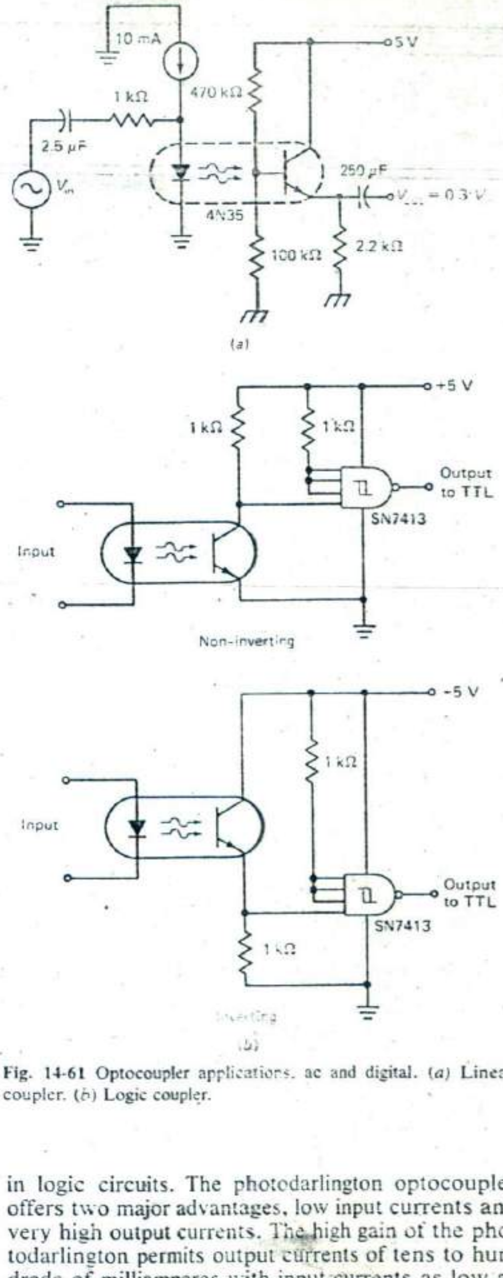


Fig. 14-61 Optocoupler applications, ac and digital. *(a)* Linear coupler. *(b)* Logic coupler.

and the collector current (output) is 2 mA, the transfer ratio is 20 percent. Photodarlingtons exhibit ratios of 100 to 1000 percent. With a 1.0-mA input, the output might be 10 mA for a transfer ratio of 1000 percent.

The dynamic response of the optocouplers is dominated by the capacitance of the photodiode, the input resistance of the transistor, and the voltage gain of the transistor. Through the *Miller effect*, the stray capacitance (C_s) is considered as a single capacitance across the input whose magnitude is approximately equal to the gain times C_s. The *RC* time constant becomes input resistance × capacitance × voltage gain. The penalty for high gain is slow response. Typical rise and fall times of the phototransistor are on the order of 2 to 10 μs, which is quite satisfactory for most analog types of applications. The addition of a resistor from the base terminal to ground will lower the gain but speed up the circuit's response. Figure 14-61(*a*) uses a 10-mA current source to bias the photodiode for linear response to the input signal (V_{in}), and the phototransistor is also biased for linear operation. Figure 14-61(*b*) shows noninverting and inverting digital coupling. The base lead is not available but is not required since linear bias is not used in logic circuits. The photodarlington optocoupler offers two major advantages, low input currents and very high output currents. The high gain of the photodarlington permits output currents of tens to hundreds of milliamperes with input currents as low as 0.5 mA. The switching speeds in the low-input-current region are quite slow but are acceptable for driving loads such as solenoids and lamps.

In the past, phototransistors were used to drive external SCRs, but today the photo SCR is packaged with the IRED. The *photo SCR optocoupler* differs from other SCRs in respect to the very-low-level gate

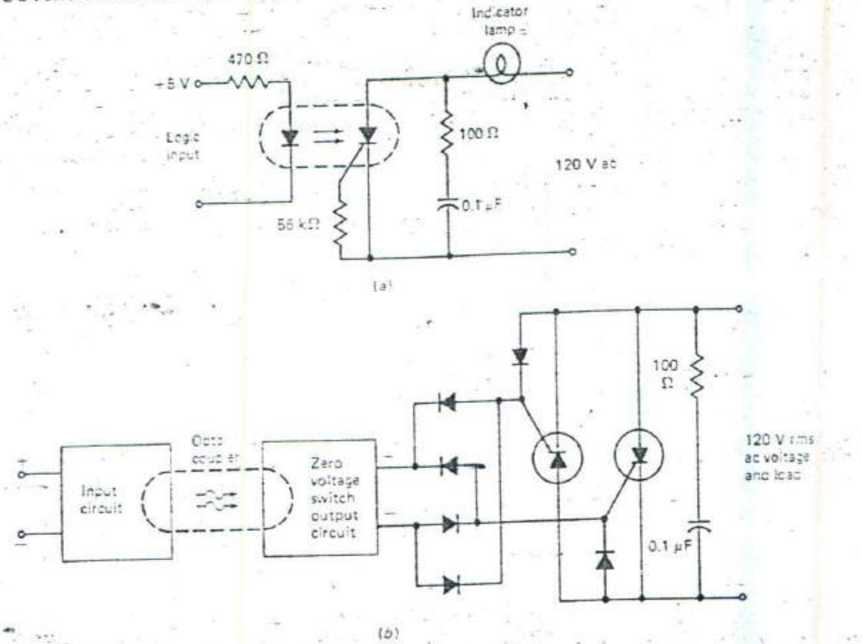


Fig. 14-62 Typical opto SCR applications.

drive available from its detector. This low-level gate drive requires a sensitive gate structure for the SCR. Applications may require the SCR to operate in 120- and 240-Vac circuits. Recent fabrication techniques have reduced some undesirable effects, such as rate effect (dv/dt). The pulse capability of the SCR makes it ideal for capacitor discharge and triggering applications. It is also applied in full-wave ac control, high-voltage SCR series string triggering, three-phase circuitry, and isolated power supplies. Figure 14-62 shows two applications of the opto SCR. In Fig. 14-62(*a*) a logic level output from a processor or computer is used to safely control a line-operated 120-V lamp or indicator. In Fig. 14-62(*b*) the device is applied as a solid-state relay (no moving parts) as to allow a logic input of 5 V at 15 mA to control a 220-V, 10-A load. The 100-Ω resistor and 0.1-μF capacitor are the suppression (snubber) network for any inductance in the load. The SCR is a half-wave device; so to obtain full-wave control, two have to be connected as configured in Fig. 14-62(*b*). The result is known as an *antiparallel* (or *inverse parallel*) *SCR*. Figure 14-63 shows a triac full-wave solid-state relay. For the most part, the optocoupled SCR can only switch milliamperes of load current, but that capability is more than sufficient to trigger larger SCRs and triacs.

As previously mentioned, zero voltage switching is necessary in many cases to reduce in-rush current and radio frequency interference (RFI). Solid-state relays with zero crossing give approximately a factor of 10 improvement in RFI, mainly because they do

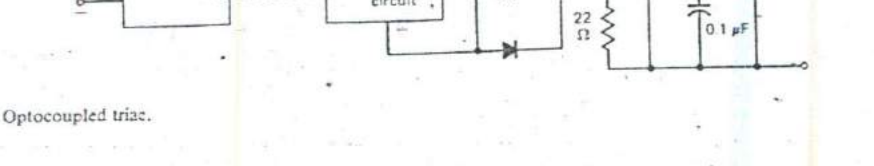
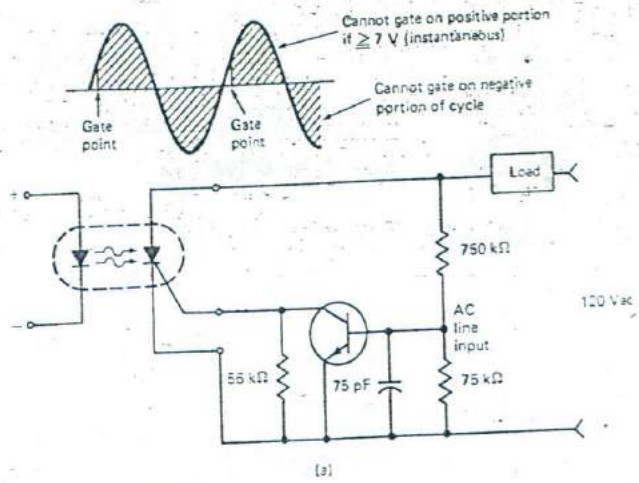

Input terminals
Input circuit
Opto coupler
Zero voltage switch output circuit
Triac
100 Ω
MOV
120 V rms ac voltage and load terminals
22 Ω
0.1 μF

Fig. 14-63 Optocoupled triac.


Cannot gate on positive portion if ≥ 7 V (instantaneous)
Cannot gate on negative portion of cycle
Gate point
Gate point
Load
750 kΩ
AC line input
120 Vac
55 kΩ
75 pF
75 kΩ
(a)

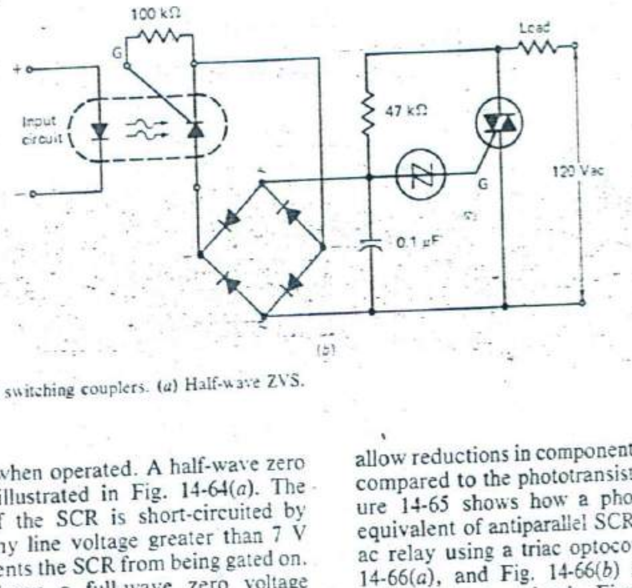


Fig. 14-64 Zero voltage switching couplers. (*a*) Half-wave ZVS. (*b*) Full-wave ZVS.

not arc or bounce when operated. A half-wave zero voltage switch is illustrated in Fig. 14-64(*a*). The gate to cathode of the SCR is short-circuited by the transistor at any line voltage greater than 7 V positive. This prevents the SCR from being gated on. Figure 14-64(*b*) shows a full-wave zero voltage switch.

Phototriacs are convenient devices when ac loads must be controlled from digital logic circuits. These devices are not designed to act as ac load switches, but as pilot devices for triggering power triacs. They allow reductions in components and circuit size when compared to the phototransistor or photo SCR. Figure 14-65 shows how a phototriac coupler is the equivalent of antiparallel SCRs. A simple solid-state ac relay using a triac optocoupler is shown in Fig. 14-66(*a*), and Fig. 14-66(*b*) shows a zero voltage solid-state relay circuit. Figure 14-67 indicates the way that the motor-starting contactors of Chapter 4 can be replaced by solid-state relays (SSRs) with no moving parts or arcing.

The bilateral analog FET optocoupler consists of

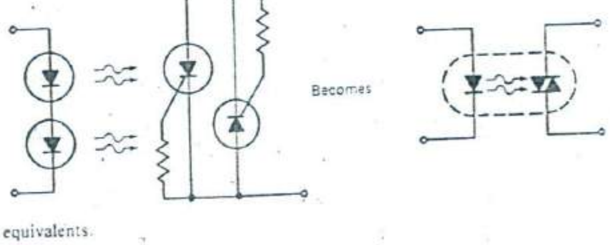
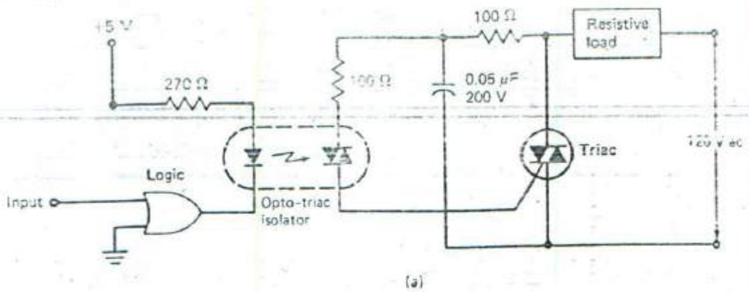


Fig. 14-65 Opto SCR/triac equivalents.



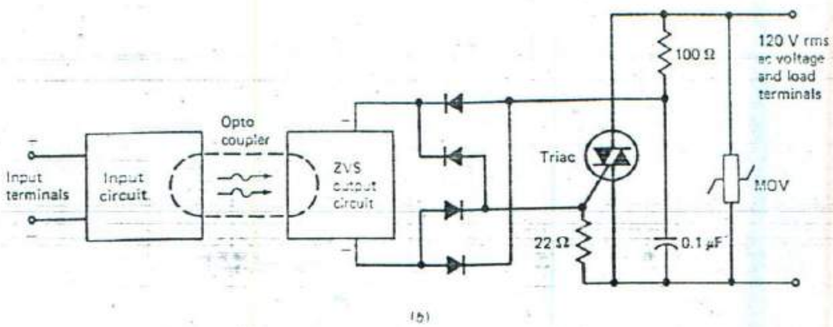


Fig. 14-66 Solid-state ac relays. (*a*) Solid-state relay (full-wave). (*b*) ZVC solid-state relay.

an IRED source coupled to a symmetrical bidirectional silicon detector chip. The characteristics are similar to those of a bidirectional FET. The output conductance is linear at low signal levels, and the bilateral analog FET optocoupler performs as a nearly ideal analog switch. A commutation circuit is illustrated by Fig. 14-68. When the control signal is high, the IRED in optocoupler (1) is on. This allows the signal to reach the op-amp because optocoupler (2) is off. When the control signal is low, optocoupler (2) is on, grounding the op-amp input. Switching the input alternately from the signal to ground is called *chopping* or *commutation* and is used to eliminate offset drift in the amplifier. Figure 14-68(*b*) shows a four-channel multiplexer circuit that selects one of four analog input signals.

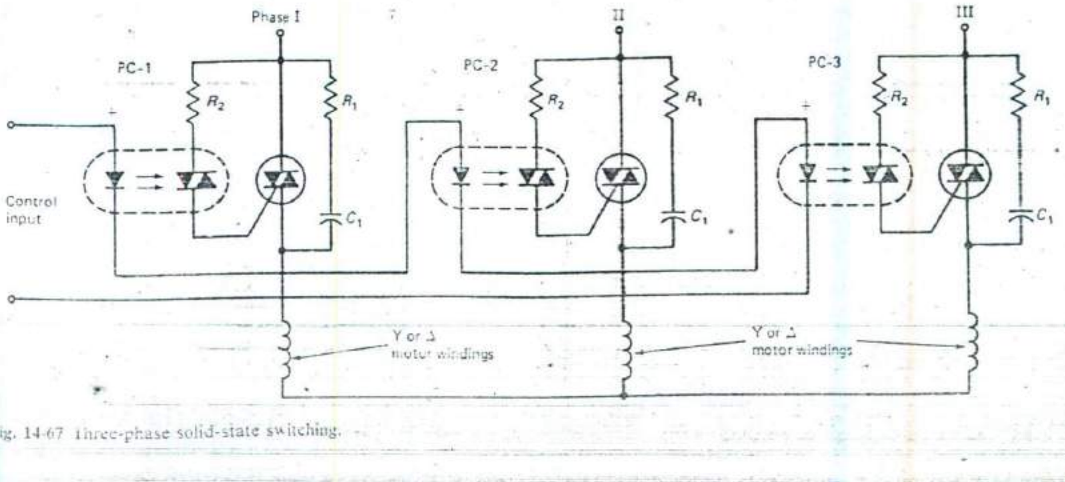


Fig. 14-67 Three-phase solid-state switching.

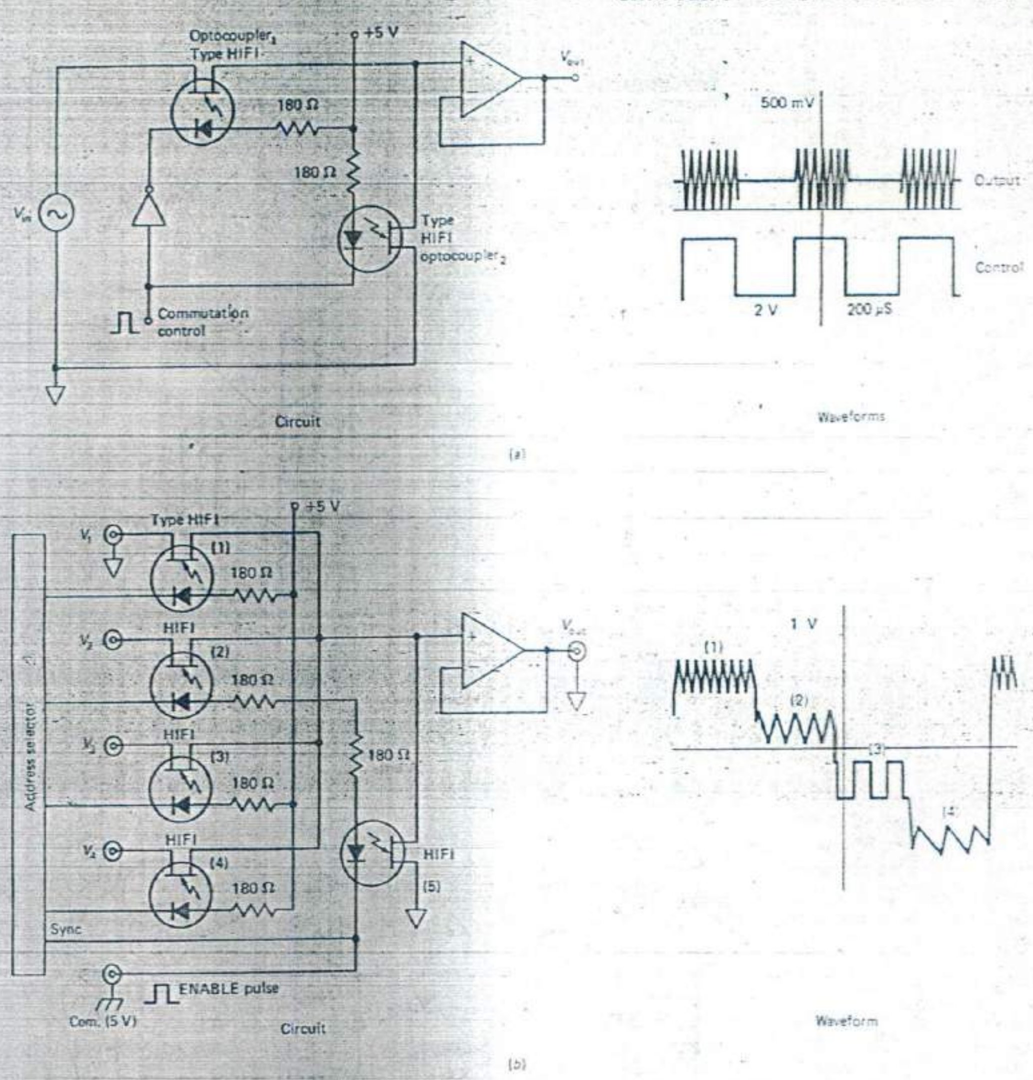
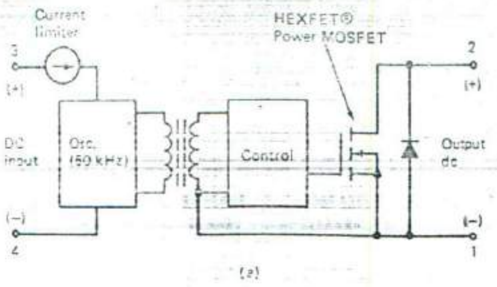


Fig. 14-68 Bilateral FET optocoupler applications. (*a*) Commutator. (*b*) Multiplexer.

Solid-state relays can be made to handle ac or dc loads. They provide isolation with optical or transformer coupling. These devices eliminate the output off-set voltages associated with a transistor SSR and are found in a wide variety of packages and sizes. Figure 14-69(*a*) illustrates a block diagram of a transformer-isolated SSR for dc. The input requires 1.6 mA at 5 Vdc, which makes it compatible with logic circuits and computers. Figure 14-69(*b*) shows a typical optically isolated solid-state relay, which lends itself to a wide variety of automatic and computer control applications.

The high-speed digital coupler, also known as a *Schmitt coupler* (because most include an internal Schmitt trigger), can interface directly with transistor-transistor logic (TTL), and LSTTL families. It provides ac and dc isolation to eliminate ground loops, allowing direct interfacing between computers and peripheral devices at data rates up to 1 M bits/s. Four common data isolators are pictured in Fig. 14-70. Both inverting and noninverting types are available, as well as totem pole and open collector output styles.

The *coupled interrupter module* is another member



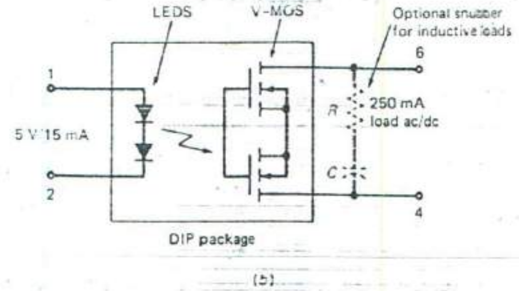
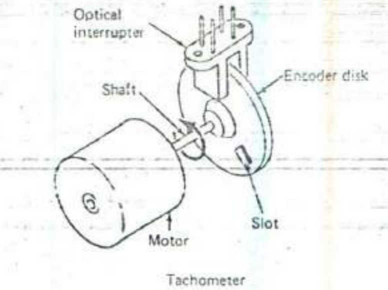


Fig. 14-69 Solid-state relays. (*a*) DC SSR block diagram. (*b*) AC/DC solid-state relay.



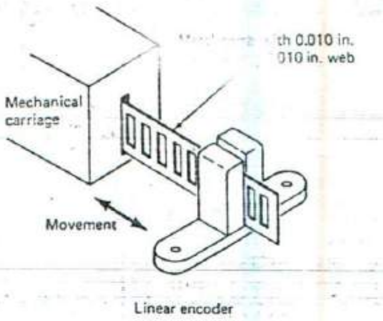


Fig. 14-71 Optical interrupter applications.

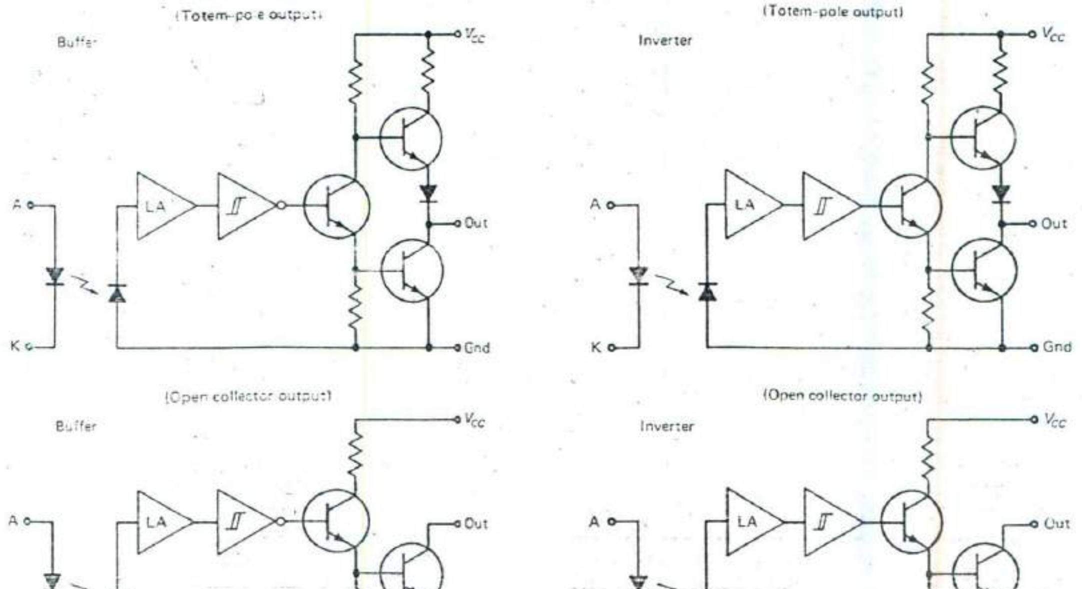


Fig. 14-70 Logic optocouplers.

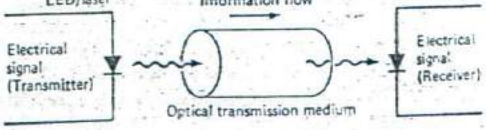

LED/laser
Information flow
PIN/APD
Electrical signal (Transmitter)
Electrical signal (Receiver)
Optical transmission medium

Fig. 14-72 Typical fiber-optic system.

of the optoisolator family. This device is also known as a *slotted switch* or *source/detector assembly*, but the name *optical interrupter* is most descriptive of its function. The device contains an IRED and a silicon photodarlington detector in a plastic housing. A gap in the housing provides a means of interrupting the light with tape, cards, shaft encoders, or any opaque material to switch off the internal transistor. Figure 14-71 shows two typical applications using the interrupter as a tachometer/speed monitor and as a linear encoder for relating distance to pulses.

The last member of the optocoupler/isolator family is the *reflective object sensor*. Each assembly consists of an infrared-emitting diode and an NPN silicon phototransistor or photodarlington mounted side by side on converging optical axes in a black housing. The photosensor responds to radiation from the IRED only when a reflective object passes within its field of view. These devices are sensitive only within a range of less than 0.3 in. (75 mm). A typical use is detection of the beginning-of-tape markers on magnetic tapes. A typical interfacing circuit uses a comparator with sensitivity and hysteresis controls.

Fiber-optic systems offer an alternative method for transmitting information and sensing physical events. Fiber optics offer a small, lightweight, durable, corrosion-resistant, nonconducting signal path that is virtually unaffected by and has no effect on the electrical environment through which the signal passes. Figure 14-72 shows that the electrical signal is changed to light, which enters the fiber cable. The light signal is changed back to an electrical signal at the other end of the cable. The system illustrated in Fig. 14-72 is called a *simplex system* (one-directional). Two-way communication (*duplex*) requires two such links. Figure 14-73 shows the basic data transmission systems. Distance limitations of fiber-optics communications arise mainly from the means of producing the optical flux and from path losses. Although power into a wire cable can easily be several watts, the flux into a fiber-optic cable is typically much less than a milliwatt. Wire cables may have several signal *taps;* however, multiple taps on fiber-optic cables are impractical at present.

The losses in a *point-to-point fiber-optic system* are insertion loss at the input and output, connector loss, and transmission loss, which is proportional to cable length. Fortunately, no noise is picked up by a fiber-optic cable so the receiver signal-to-noise ratio (SNR) is limited only by the noise produced within the receiver.

Light rays are confined to the core of the optical fiber by cladding the core with a transparent material having a lower index of refraction. This defines the critical angle of reflection at the core cladding interface, thereby confining rays at smaller angles to the core of the fiber. A typical optical fiber cable is shown in Fig. 14-74.

There are three common optical fiber types: the stepped index multimode, stepped index monomode,

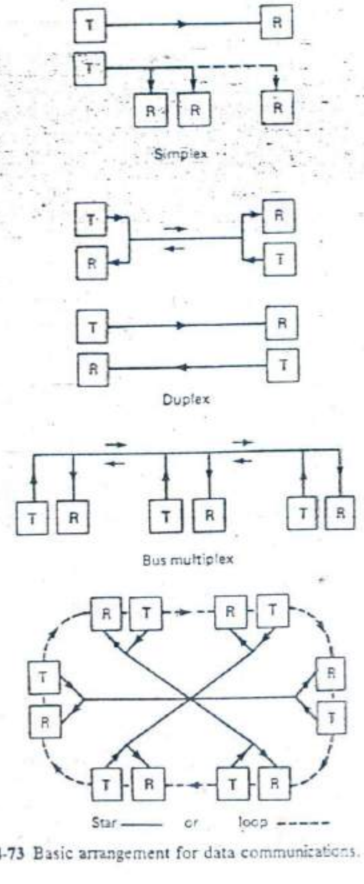


Fig. 14-73 Basic arrangement for data communications.

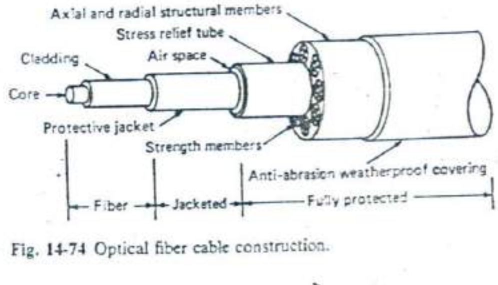


Fig. 14-74 Optical fiber cable construction.

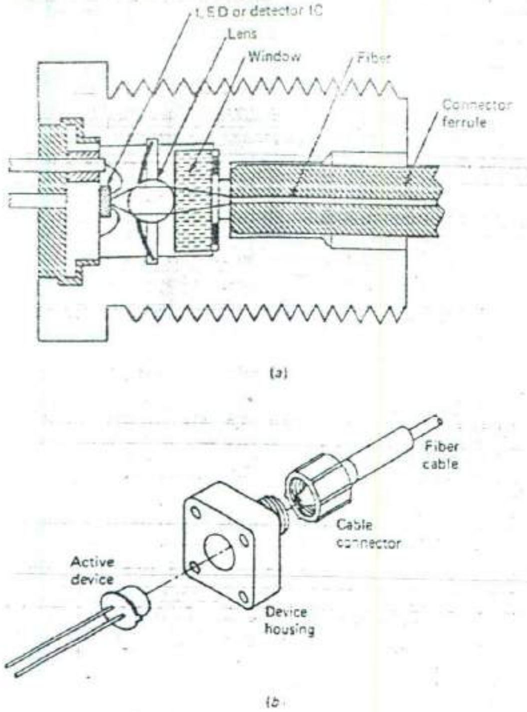


Fig. 14-75 Fiber cable connection to active device. (*a*) Cross section. (*b*) Mating fiber connector.

and graded index multimode. We will be concerned mainly with the stepped index multimode fiber, which is used for moderate bandwidth applications.

The term *stepped index fiber* refers to the abrupt change in the index of refraction between the core and the cladding. All transmission of light occurs within the core. The different index of refraction between the core and the cladding defines a critical angle such that any light ray entering the core at less than that critical angle will be completely reflected. The numerical aperture is the *sine* of this angle and defines a cone in which incident light may be launched into a cable. The smaller the numerical aperture, the harder it is to launch light into the fiber.

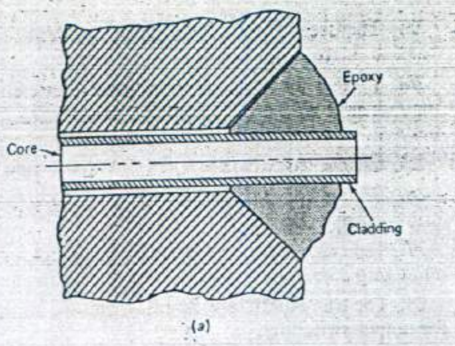
A wide variety of fibers exist, with little standardization, in three basic materials: *plastic clad-plastic core fiber* (plastic fiber); *plastic clad-glass (silica) core fiber* (PCC fiber); and *glass clad-glass core fiber* (glass fiber). Fiber attenuation varies greatly within the core material types and with the wavelength of the light used. So far, this discussion of optical fibers has treated only single fibers. In some applications, improved performance may be gained by utilizing a *fiber bundle*, which consists of a group of single fibers in a single jacket. A bundle is more flexible than a single fiber of equal area and continues operating even if a few strands break. A bundle is usually harder to terminate with a connector than a single fiber. It is usually more difficult to polish the ends of the fibers of a bundle. Higher-power insertion losses are caused by a poor finish. Poor fiber connections and polish have been observed to cause up to 10-dB (90 percent power loss) signal loss per end.

There is a wide variety of fiber-optic connectors because of a lack of standards. To provide a low-loss fiber-to-fiber splice, a connector must position the two optically polished fiber ends very closely together in axial concentric alignment. If the fiber ends touch, abrasion may spoil the end finish and cause power loss. Some connectors for plastic fibers maintain pressure between the fiber ends. The pressure deforms the plastic ends for a better fit. Coupling efficiency falls off rapidly as the distance between the fiber ends increases and also with angular error and errors off concentricity. In general, connectors for fibers of 200-μm core diameter and greater are easier to install and provide better consistency than connectors for the smaller-diameter cables.

Most active devices, such as emitters and detectors, are applied to fibers with adapter connectors or short lengths of fiber built into the active device and terminated within a connector. Figure 14-75 illustrates the mating of a fiber cable connector with a detector/emitter.

The preparation for repairing damaged cables is quite tedious. The 10- to 12-step process requires skill and practice. The cable connectors are ultrasonically cleaned and baked. The fiber in the cable is cleaved (cut with a blade), cleaned, and epoxied into a ferrule shoulder. The sealing of the fiber is shown in Fig. 14-76(*a*). Eighteen hours must elapse for the fiber to be cured before it can be polished. The cutaway view of the polished end is shown in Fig. 14-76(*b*). Caution must be taken to wear eye protection to avoid injury from any fragments of the fibers. Also avoid any skin punctures. In many cases, prepared cables with connectors are used or installed. They are available in 1- to 1000-m lengths.

The emitters are usually LED/IREDs or diode lasers (lasers will be covered later in this chapter). A majority of LED/IRED emitters operate at wavelengths of 550 to 1300 nm, with most falling in the 640- to 940-nm range to work more efficiently with the detectors. Figure 14-77 shows simple digital and analog fiber transmitter circuits. The detectors (sensors) can be photodiodes, phototransistors, PIN photodiodes, or avalanche photodiodes, all of which operate on the same basic principle. In some cases, the receiver circuitry is integrated in the same package with the detector. This assembly may be no larger than the typical metal case transistor (TO-5). Figure 14-78(*a*) shows a simple receiver (detector) circuit. Its response would be limited to about 150 kHz by



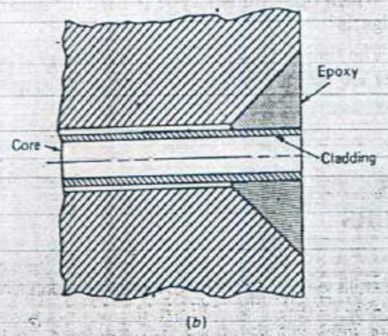
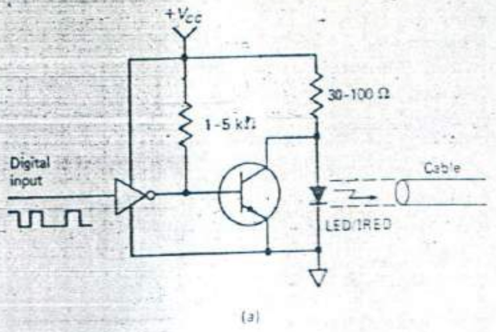


Fig. 14-76 Fiber cable sealing/termination. (*a*) Sealing fiber to ferrule. (*b*) After polishing.



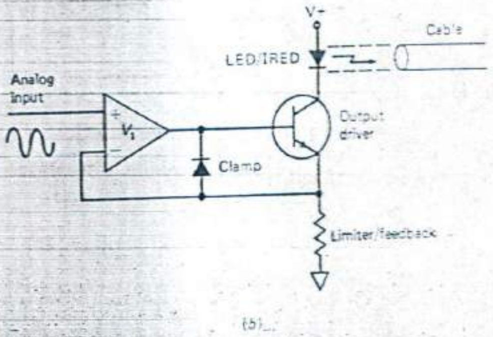
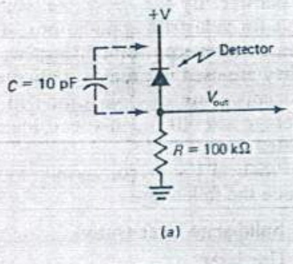
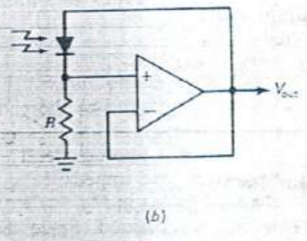
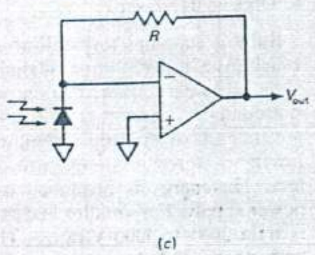


Fig. 14-77 Simple fiber-optic transmitters. (*a*) Digital transmitter. (*b*) Analog transmitter.







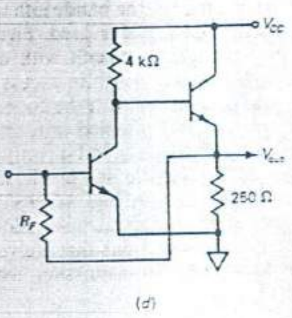


Fig. 14-78 Transimpedance configuration. (*a*) Simple receiver. (*b*) Bootstrap configuration. (*c*) Transimpedance. (*d*) Discrete transistor amplifier (transimpedance).

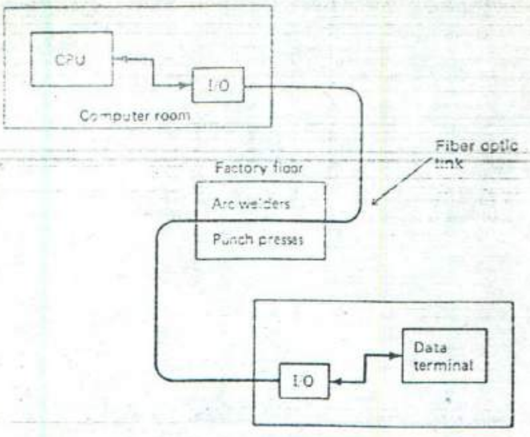


Fig. 14-79 Sample system demonstration.

the time constant of the 10-pF shunt capacitance along the 100-kΩ load resistor:

$$\begin{aligned} T_r &= 2.2RC \\ &= 2.2 \times 100 \times 10^3 \times 10 \times 10^{-12} \\ &= 2.2\ \mu\text{s} \\ F &= \frac{0.35}{T_r} \\ &= \frac{0.35}{2.2 \times 10^{-6}} \\ &= 159\ \text{kHz} \end{aligned}$$

where T_r = rise time (time from 10 to 90% of the output waveform)
F = upper cutoff frequency of an amplifier
$F \times T_r = 0.35$, a constant

Figure 14-78(*b*) is a *bootstrap configuration*. The amplifier follows the voltage developed by the photocurrent flowing through the resistor and applies this voltage to the opposite end of the photodiode. This configuration provides a lower load impedance and a faster response. By rearranging, as in Fig. 14-78(*c*), a transimpedance amplifier is obtained. Since the inverting input is a virtual ground, the bandwidth will be determined mainly by the amplifier used. Figure 14-78(*d*) shows a transimpedance circuit with discrete components that has a bandwidth in excess of 50 MHz. Many receivers are made with the connector, detector, and amplifier all contained in a small low-profile package that can be mounted directly to a printed circuit board. A fiber-optic link in a hostile industrial environment is shown in Fig. 14-79. The outstanding noise immunity of optical links allows them to operate flawlessly in situations that are very difficult for conventional data transmission techniques.

REVIEW QUESTIONS

22. Most optocouplers use a GaAs _______ as the internal emitter.

23. The use of a photodarlington coupler provides an increase in speed. (true or false)

24. If the collector current changes 10 mA for an input change of 2 mA the transfer ratio is _______ percent.

25. The photo SCR is a _______ wave switch.

26. The antiparallel photo SCR connection is for _______ wave control.

27. The zero voltage switch (ZVS) control turns on at zero current crossings. (true or false)

28. The true solid-state ac relay uses a _______ for output switching.

29. Which optocoupler type is used for linear analog signals?

30. The three main output parts of a digital optocoupler are the phototransistor, the linear amplifier, and the _______.

14-5 LASERS

Lasers were originally referred to as *optical masers*. *Maser* is an acronym for *microwave amplification by stimulated emission of radiation*. *Laser* is an acronym for *light amplification by stimulated emission of radiation*. The first laser was a ruby crystal device developed in 1960. It provided a single wavelength, which is called *monochromatic* (one-color) *light*. Lasers can also provide *coherent light*, with all of the waves in phase. Early lasers were fragile and expensive. By the 1970s, reliable lasers were available for industrial applications. They are a practical source of energy for cutting, welding, and drilling. They are used in precision alignment and measuring systems. Future applications in the areas of mass storage and ultra-high-speed logic circuits are anticipated.

Practical lasers commonly found in industry include the following:

1. Solid-state laser (ruby)
2. Gas laser
3. Semiconductor laser
4. Organic dye laser

Ruby is sapphire (crystalline aluminum oxide) in which a small percentage of the aluminum has been replaced by chromium. The chromium concentration is around 0.05 percent. Figure 14-80 shows the component parts of an optically pumped solid-state laser. Energy is stored in a capacitor or a bank of capacitors. The energy is produced by a dc high-voltage power supply. For smaller laser crystals, the voltage is in the 2000- to 5000-V range. The flash tube is filled with xenon and does not conduct until the gas is ionized by a high-voltage pulse from the trigger transformer. When it is pulsed by the trigger, the stored energy (in joules [J] = 1/2 (farads × volts^2) causes the xenon tube to emit very intense radiation, which

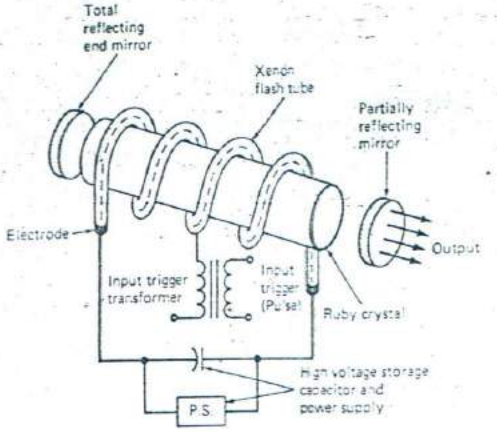

Total reflecting end mirror
Xenon flash tube
Partially reflecting mirror
Electrode
Output
Input trigger transformer
Input trigger (Pulse)
Ruby crystal
High voltage storage capacitor and power supply
P.S.

Fig. 14-80 Optically pumped solid-state laser.

then is absorbed in the laser crystal by a pumping action. This optical pumping action is necessary for all solid-state lasers to function. Light from the flash tube pumps the chromium atoms from ground state G to excited levels in band E or band F. The pumped atoms are unstable at this level and lose energy in a two-step process. The first step is to a metastable level M. No emission takes place at this level. The second step is from M to ground G state, with release of photons by simulated or spontaneous emissions. This is called a *three-level energy laser action*. There is also a four-level energy action in some lasers. The main laser action occurs in the optical cavity formed by the reflecting mirrors at the ends of the laser rod. The laser beam emerges through the partially transmitting mirror at the right end of the laser shown in Fig. 14-80.

To increase the effective optical coupling between the flash lamp and the laser rod, it is necessary to surround the complete assembly by reflecting walls. Figure 14-81 shows a system that uses a helical flash lamp with the ruby rod centered in the enclosure.

The flash lamp is similar to those used in photography. The flash lamps used in lasers are glass or quartz tubes filled with a gas at low pressure. Xenon is commonly used, although higher brightness is achieved by using krypton or helium. The lamps can come in various shapes: helical, linear, and U-shaped are the most common. Electrical input energy, pulse duration, and lamp size can be varied over a wide range. When an electric current is discharged through the lamp, a high-temperature plasma which emits radiant energy over a wide range of wavelengths is formed. The gas type and pressure can be adjusted to produce a wavelength peak that matches the absorption spectrum of the crystal material being used.

Discharge in a flash lamp is initiated by causing a spark streamer to form between electrodes (usually at the ends of the tube). This is done by pulsing the lamp with a high-voltage trigger pulse. A typical circuit using a pulse transformer to step up the externally applied trigger pulse is shown in Fig. 14-82. Once the flash lamp has been triggered, the main discharge capacitor C can discharge through the ionized gas. The series inductance is added to shape the current pulse through the lamp and to avoid ringing or reverse flow of current, which is harmful to the lamp life.

The lasers we are concerned with produce light in the near-ultraviolet, visible, and infrared portions of the electromagnetic spectrum. The wavelengths are in the approximate range of 10^{-3} to 10^{-5} cm, and frequencies are on the order of 10^{13} to 10^{15} Hz. Note: Light wavelengths are usually expressed in micrometers (μm) or nanometers (nm). The micrometer is equal to 10^{-4} cm and was often referred to as a *micron*. The unit angstrom (Å), equal to 10^{-8} cm, was also commonly used. For example, a green light of 5.5×10^{-5} cm = 0.55 μm = 550 nm = 5500 Å. Both the angstrom and the micron are considered obsolete terms. Their use as modern units of measurement is being discontinued.

Lasers can be constructed by using a variety of different materials, each of which produces a distinctive wavelength. Figure 14-83 shows where the output of some of these materials occurs in relation-

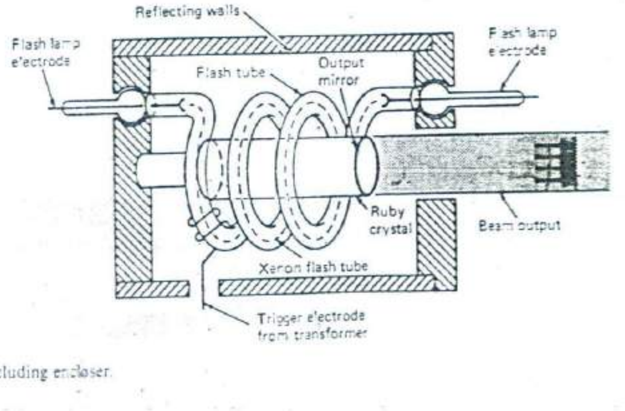


Fig. 14-81 Ruby laser, including encloser.

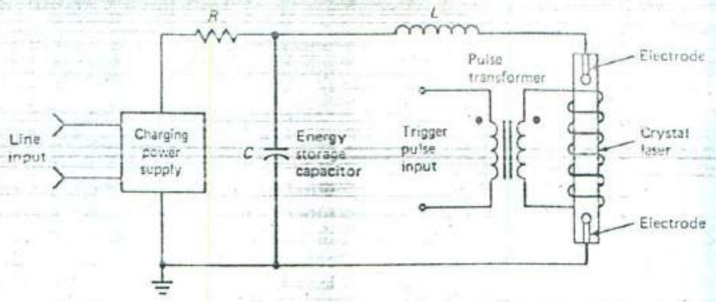


Fig. 14-82 Schematic of a typical flash lamp circuit.

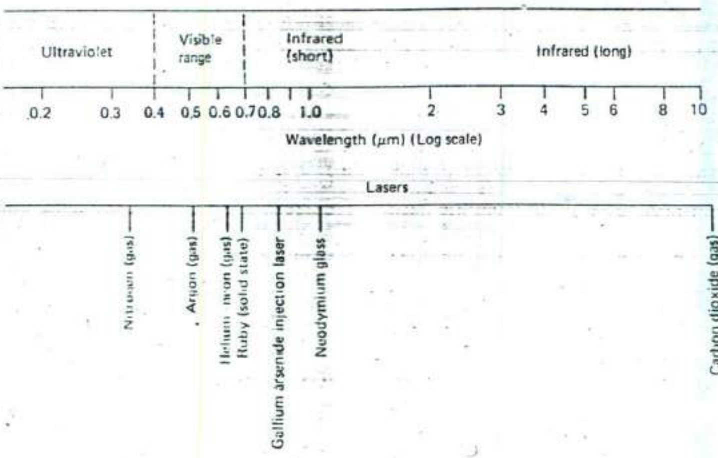


Fig. 14-83 Electromagnetic spectrum of wavelength of lasers.

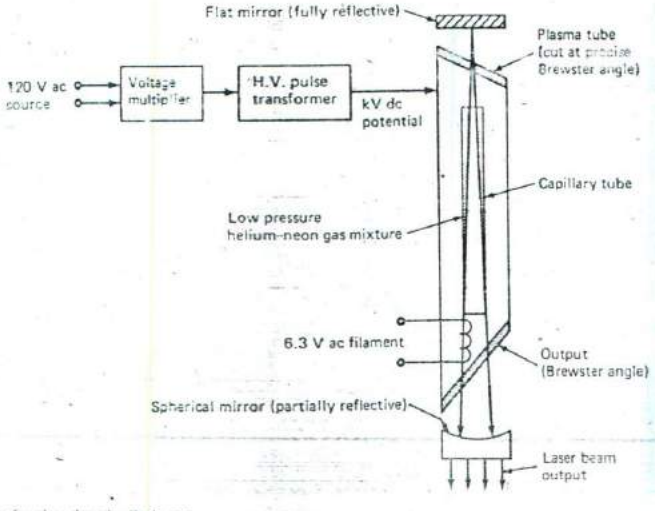
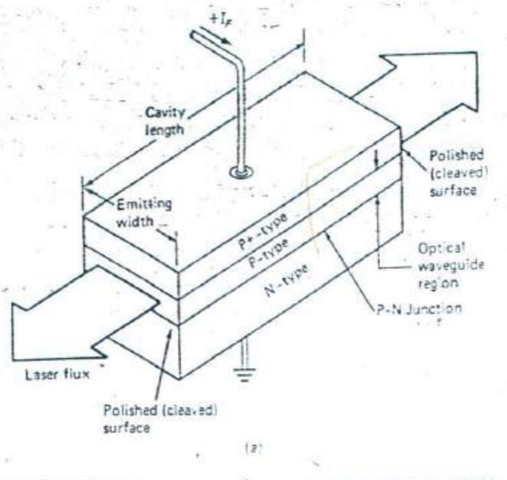

Flat mirror (fully reflective)
Plasma tube (cut at precise Brewster angle)
120 V ac source
Voltage multiplier
H.V. pulse transformer
kV dc potential
Capillary tube
Low pressure helium-neon gas mixture
6.3 V ac filament
Output (Brewster angle)
Spherical mirror (partially reflective)
Laser beam output

Fig. 14-84 Gas laser: excitation by electric discharge.



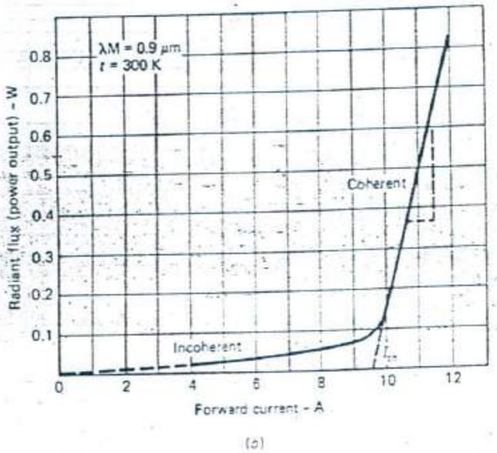


Fig. 14-85 Injection laser diode structure and output characteristics.

ship to each other and to the electromagnetic spectrum as a whole. The most useful lasers include the following:

1. The CO_2 laser, far infrared to 10.6 μm
2. The neodymium glass laser, near infrared at 1.06 μm
3. The gallium arsenide laser, near infrared, at wavelengths around 0.85 to 0.90 μm
4. The helium-neon laser, emitting 0.6328-μm radiation, reddish-orange in color
5. The argon laser, operating at several wavelengths in the blue and green portions of the spectrum
6. The nitrogen laser, an ultraviolet laser, operating at 0.3371 μm

This listing is not complete but will serve as a reference point for the significant types.

Gas lasers function quite differently from pulsed solid-state lasers. Gas lasers are not powerful. They operate mostly in the continuous-wave (CW) mode. Their steady beams will not burn holes or vaporize steel as the pulsed (crystal and semiconductor) laser can (this effect will be discussed shortly). There are many reasons for using gas. The volume of the material can be large, in contrast to those of crystals and semiconductors. Heat can be removed readily by transporting the heated gas out of the region and replacing it with cooler gas. A mixture of two gases is required for laser action. Atoms of the second gas are raised to higher levels with external excitation. For example, a laser might use 10 parts helium and 1 part neon. The mixture is pumped with radio frequency energy. A more modern approach is to use a high-voltage discharge from a voltage multiplier circuit, while a filament heats the gas as shown in Fig. 14-84. A further refinement of the electronics has eliminated the need for a filament to start the ionization of the gases. The circuit uses a voltage quadrupler to produce a starting boost of 7.5 kV to ionize the gas in the tube.

As discussed earlier, it is possible to generate light beams (emitter section) by using the principles of semiconductors. Laser diodes made from direct bandgap material differ from conventional light-emitting diodes in that they require an optical cavity and a high-injection carrier density. Thus the name *injection diode* usually refers to the semiconductor laser diode. The semiconductor laser is very efficient and small in physical size compared to other types of lasers. Figure 14-85(*a*) shows an injection laser diode structure, with a PN optical cavity formed by cleaving opposite ends of the diode and sawing the adjacent sides of the rectangular structure. In Fig. 14-85(*b*) the radiant output power as a function of the diode current is demonstrated. Note that a threshold current (I_{th}) of about 10 A is required for coherent output. Since the required current is so high, the device is operated in the pulsed mode only. The emission wavelength depends on the semiconductor material, doping level, and temperature of operation. In some injection lasers, a film of silver or gold is deposited over one end so that the other end will serve as the only output. The gallium arsenide material is reflective, so no internal mirrors are required to produce reflections. Typical lasers of this type emit 0.5 to 50 W of pulsed energy. Many injection lasers are operated at cryogenic temperatures (about 77 K). Two basic difficulties in achieving high peak power output from single laser diodes are that (1) high drive currents are required to drive the larger laser pellets and (2) the larger source size requires large, costly optics.

Laser diodes are also arrayed to provide increased power outputs at reasonable drive currents. Stacked diode lasers are currently available with minimum peak radiant flux levels ranging from 75 to 300 W with a drive current of 40 A. The increase in source radiant excitance (emittance) for these arrays com-

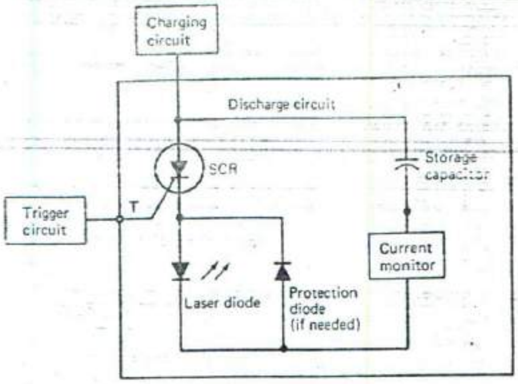


Fig. 14-86 Basic circuit blocks for solid-state pulser for injection laser diodes.

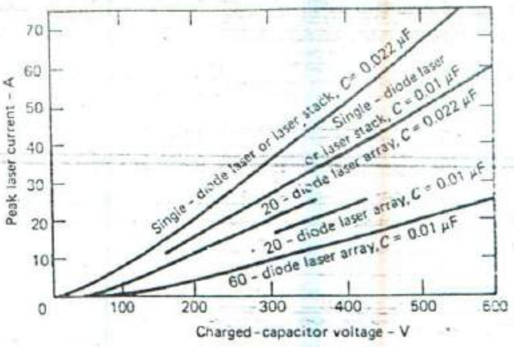


Fig. 14-87 Peak laser currents and capacitor voltage for pulsed lasers.

pared to single-diode arrays is significant; however, some sacrifice in duty cycle limit must be made.

The SCR serves as an off-to-on solid-state switch for many injection lasers and laser arrays. Figure 14-86 illustrates the basic blocks for a solid-state pulse power supply for injection lasers. The discharge circuit generates the current pulse in the laser and is the most important section of the pulser. The current pulse is generated by discharging storage capacitor C through the SCR and the laser diode, laser stack, or laser array. The rise time of the current pulse is usually determined by the SCR; the fall time is determined by the capacitor value. The peak laser current and the charged-capacitor voltage relationships for some typical injection lasers using the circuit of Fig. 14-86 are illustrated in Fig. 14-87.

The circuit shown in Fig. 14-88(*a*) can deliver a peak current of 30 A for both single-diode lasers and laser stacks. The circuit in Fig. 14-88(*b*) can drive single or stack laser diodes with currents up to 75 A. The circuit of Fig. 14-88(*c*) is capable of driving a laser array consisting of up to 60 diodes in series.

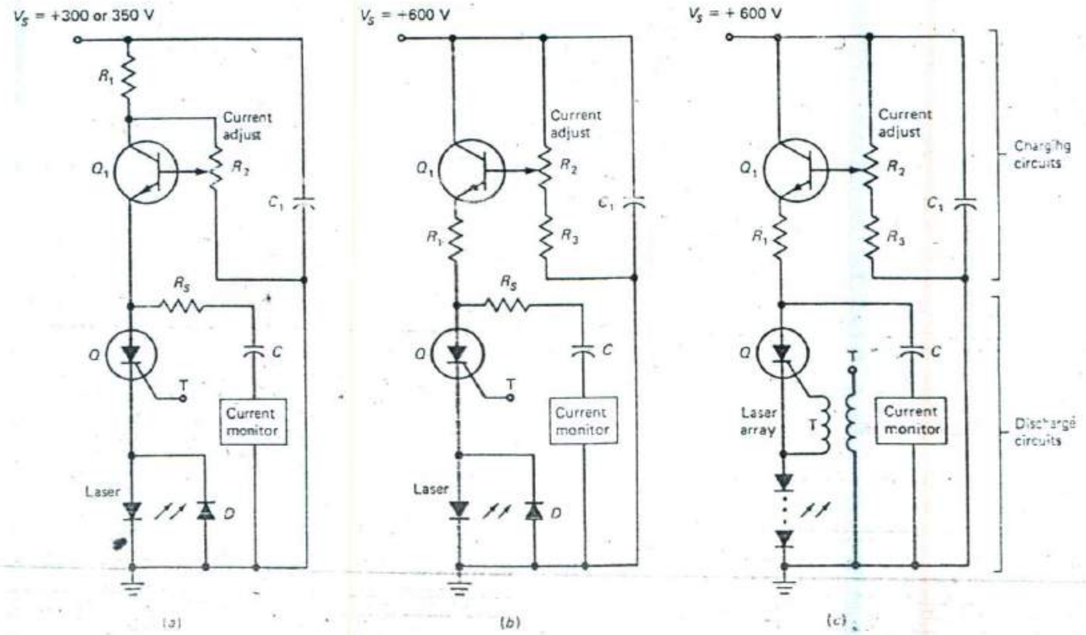


Fig. 14-88 Laser and laser-array pulse power supplies.

The repetition rates are limited to less than 500 Hz.

Liquids have useful advantages in relation to both solid and gas laser media. Several different liquid lasers have been developed, but the most important is the *dye laser,* which has the advantage that it can be tuned over a significant wavelength range. This capability is extremely useful in many applications such as spectroscopy and the study of chemical reactions. The materials that are used in dye lasers are similar to many familiar dyestuffs that are commonly employed as colorants in fabrics, plastics, soaps, and cosmetics. A pulsed laser is put through a block of the dye material then filtered (optically); this type of laser has almost no industrial applications at present.

Industrial applications of lasers include welding, hole drilling, cutting, trimming of electronic components, and heat treating. The most important property of laser energy is that it can be concentrated by a lens to achieve extremely high-power density at the focal spot. Let us compare a CO_2 (gas) laser and a large mercury arc lamp. The CO_2 laser emits 100 W of power in a beam that can be focused by a lens with a focal length of 1 cm to a spot 0.01 cm in diameter. Without going into the physics of beam divergence, the power density is 10^6 W/cm^2. In comparison, a mercury arc lamp of 1000 W has much more output; however, all the light from this lamp cannot be focused because the large divergence angle of the rays leads to a very large focal area. Therefore, the power density from this lamp is much lower. It can be shown (although we shall not go through the details here) that the same lens (focal length of 1 cm) will deliver a power density of only 100 W/cm^2 in the focal area. Thus, the total power emitted by the lamp is 10 times larger than for the laser, but the power density delivered to a workpiece by the laser is 10,000 times greater. Pulsed lasers can deliver much higher values of power density. A ruby laser can easily produce 1-ms-duration pulses with a power density of 10^9 W/cm^2 at a workpiece. When a high-power beam interacts with the workpiece surface, the material at the focal point is vaporized.

When laser radiation falls on a target surface, part of it is absorbed and part is reflected. The energy that is absorbed will heat the surface. Reradiation from the surface is usually insignificant. Heating from absorption can occur rapidly (losses due to thermal conduction are small if the pulse width is short but are important with long pulses). The surface quickly rises to its melting temperature. Melting is important in welding applications, and vaporization should not occur. Many workpiece factors such as heat flow, thermal conductivity, and material density determine the required power density for laser welding. The depth of penetration is important when sealing lids on difficult-to-weld containers housing delicate, heat-sensitive components, such as in a heart pacemaker container.

Metal removal (cutting or drilling) generally means operation at higher levels of laser power density than for welding. As in welding, material properties influence the amount of metal removed. Pulse lengths in the range of several hundred microseconds are generally used. Repeated pulses to the same target area form deeper holes. Hole drilling is possible in ceramic and similar materials which are hard, brittle, and easy to break. There are no drill bits to break or wear out. For example, a jet engine part with 400 holes can be drilled by laser machining (with computer control) in less than 30 min. Figure 14-89 illustrates laser pulse shapes used for welding and drilling (drilling waveforms are also used for cutting, by overlapping the holes). The marriage of laser machining with robotics provides a manufacturing capability able to handle some very intricate production tasks. The laser is also used for alignment and is able to obtain accuracies of 0.085 μm/m.

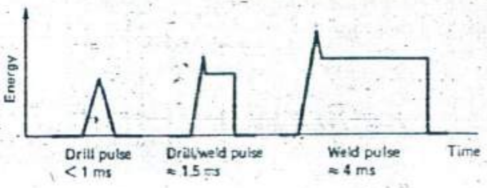


Fig. 14-89 Laser pulse shapes for welding and drilling.

Lasers have a wide range of potential applications in processing, storing, and transmitting information. They will have a large impact in these areas in the near future.

REVIEW QUESTIONS

31. Ruby lasers are usually excited by a ______ flash lamp.

32. This excitation action is also called optical ______.

33. The flash lamp is triggered by a high-current low-voltage pulse. (true or false)

34. The laser cavity is formed by ______ mirrors at each end.

35. Helium- ______ and CO_2 are common gas lasers.

14-6 PHOTOTUBES AND PHOTOMULTIPLIER TUBES

Light striking the large photocathode surface of a phototube will transfer energy to electrons in the metal, causing them to be ejected. This effect is called *photoelectric emission.* The photocathode surface is selected for a *low work function* (least amount of energy needed to remove electrons from the substance). Cesium is the most efficient photocathode used in the visible and infrared range. The phototube cathode emits a number of electrons which depends on the number of photons (intensity of the light)

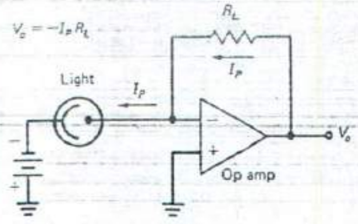


Fig. 14-90 Phototube coupled to op amp.

striking its cathode surface. The electrons are then attracted to the anode, which is positive with respect to the cathode. In a typical simple circuit for a phototube the photocurrent develops a signal across a load resistor. Increasing the value of the load resistor increases the sensitivity of the circuit (volts/lumen). This effect is desirable in some cases but can produce nonlinearity. Operating at a lower voltage (say, 100 V) will decrease the output voltage, but not the sensitivity.

Since the phototube current is a function of the anode-cathode voltage, if below 20 V or so, it is important that the voltage developed across the load resistor be as small as possible compared to the applied voltage. This places a limit on the value of the load resistor if a linear response is required (as it almost always is). This problem can be overcome by using an operational amplifier, as shown in Fig. 14-90. This configuration has been shown previously as either a current-to-voltage or transimpedance amplifier, where

$$V_o = -I_p \times R_L$$

In this case, the anode of the phototube is kept at virtual ground, and a negative bias is applied to the photocathode.

To improve the sensitivity, gas-filled tubes that produce larger currents for lower light input are used. The increase in photocurrent is due to secondary ionization of the gas molecules, but the characteristic curves are quite nonlinear. For accurate measurements, the vacuum phototube is employed. When less accuracy, but high sensitivity is required, the gas-filled phototube is selected. This device is similar to the photodiode previously covered. Most applications today use the solid-state device. As a result of certain environmental constraints, such as high temperature, nuclear radiation (which destroys most solid-state devices), and high humidity, the phototube may be used in some situations. The phototube is a rugged, long-life, durable device and will be employed in certain environments for many more years.

The phototube has very low gain at low intensities of light. The photomultiplier (PM) tube overcomes this deficiency. The basic operating principle of the PM tube is illustrated in Fig. 14-91. Light falling on a light-sensitive photocathode causes it to emit free

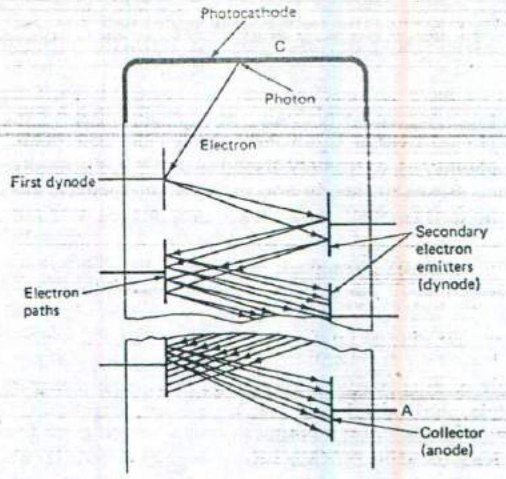


Fig. 14-91 Basic PM tube principle.

electrons, which are drawn away from the photocathode by an electrode having a more positive potential. These electrons accelerate and strike the first dynode, where they dislodge additional electrons. The next dynode is more positive, so the electrons are now accelerated to it. This process is repeated, with each stage having a higher positive potential than the previous one. The electrons emitted from the last secondary stage are collected at an anode (*A* in Fig. 14-91), and the resulting current is passed to the accompanying circuitry.

The number of secondary electrons emitted is dependent on the type of surface, the energy of the bombarding (primary) electrons, and the angle of incidence of the primary electrons. The ratio of the number of electrons leaving the surface to the number of incident electrons is called the ***secondary emission ratio***. The total current amplification factor (G) between the photocathode and the anode is given by

$$G \doteq \delta^N$$

where G = current gain
N = number of dynodes
δ = secondary emission ratio

Considerable gain is possible. If, for example, $\delta = 6$ and $N = 9$, we obtain a gain of 1×10^7.

The gain for a particular PM tube depends on the physical structure and the potential difference between each stage. A graph is prepared for each individual tube type to indicate the change in average overall amplification with the voltage per stage.

The dynode biasing chain in most cases is a linear resistor network. The negative high voltage ($-$HV) is applied across a resistor string which serves as a voltage divider and maintains the dynodes at increasingly higher positive potentials. The first dynode may be a grid used for focusing in some PM tubes.

If so, the cathode resistor will be two times (twice the potential of) the other elements. When the amplified signal current (or pulse) arrives at the anode, it flows through the anode load resistor or into the input of an op amp for a current-to-voltage transformation.

The current that flows in the PM tube when no light or radiation is falling on the photocathode is called the *dark current*. The dark current will limit the minimum detectable signal since it will be amplified along with any photocurrent due to light input. The dark current is temperature-dependent, and some applications cool the PM tube to improve its weak signal performance.

Current peaks in the dynodes can cause the voltage supplied by the divider network to sag. This sagging results in a type of degenerative feedback which may result in a nonlinear output from the PM tube. The last few dynode bleeder resistors are bypassed to stabilize the voltages. The PM tube is a very sensitive device, with gains of an order of magnitude greater than those of solid-state detectors. They are less sensitive to temperature and environmental effects. A great deal of development work has been done on photoemissive surfaces.

REVIEW QUESTIONS

36. Phototubes use cesium-coated anodes to increase their integrity.

37. Refer to Fig. 14-90. If I_P is $-10\ \mu A$ and R_L is 100 kΩ, V_O equals ______.

38. For higher sensitivity a ______ type phototube is used.

39. The PM tube relies on the ______ emission of the dynodes for multiplication.

40. The dynodes of PM tubes must be increasingly ______ to sustain secondary emission.

14-7 TROUBLESHOOTING AND MAINTENANCE

Possible hazards unique to optoelectronic devices are due to some of the materials employed. Although gallium arsenide and gallium aluminum arsenide are both arsenic compounds, under conditions of normal use they are considered relatively benign. Electrical or mechanical damage to devices containing these materials should not produce a toxic hazard, but thorough washing of one's hands before eating or touching any food is recommended.

The eye may be damaged by infrared light. Most present GaAs and GaAlAs devices do not approach the safe limit values set by governmental agencies, but all the manufacturer's safety recommendations must be observed. If any doubt exists, check first before servicing an infrared device.

The output from a laser can be a source of fascination, but one must always maintain caution while near these devices. Never look into the beam of any laser. Reflections from polished surfaces are as dangerous as the original beam itself. Always be aware of the beam path and of others in the area when performing any action that may cause movement of the beam. Also, the high voltages involved may be lethal, so read the manufacturer's manuals and become familiar with the unit before energizing or troubleshooting any high-voltage components.

All PN junction emitters are operated in the forward-bias region. They can be tested in the same way as a common diode; using a multimeter with a diode test mode will exhibit the similar (with different thresholds) characteristics in the forward direction. The reverse breakover voltage is very low, usually only 2 V. As with all electronic equipment, the power supply must first be verified. With optoelectronics, dust, dirt, oil, etc., on the light-sensitive emitter or detector can be a source of problems. In some cases, the build-up is slow, and adjustments are made until they are out of range. Input from an operator may provide useful clues. High ripple on a power supply can have a very adverse effect and can cause unstable operation, including relay chattering, overheating, and premature failure of solid-state devices. A good maintenance record provides valuable information and may indicate trends.

All photodetectors are heat-sensitive and exhibit increases in dark current and shifting of operating characteristics. Therefore, any changes in ventilation or the surrounding environment should be noted. Many photodetector circuits can be verified (go–no-go) with the use of a small test lamp, which can aid in localizing problems. The output of the first amplifier in a system can be observed by using a meter, probe, or scope. A blinking light source is preferred to avoid saturating any of the circuits.

Photocells can be checked easily with an ohmmeter by exposing them to light and dark conditions. If the device is used in a bridge configuration, the power supply along with the other legs of the bridge must be checked. The bridge and detector circuit can be verified by substituting a variable resistance for the cell. It can be varied to simulate the cell's response to check the response of amplifiers and detectors.

Most displays that incorporate LEDs have some sort of driver and limiting circuitry. Failures here are not uncommon. In the segment-type display, knowing the function codes can help verify whether the decoder or the display itself is defective. Many units have a test input for lighting all of the segments. Multiple failures may indicate a power supply problem. Overvoltage shortens LED lives exponentially, and excessive ripple may cause the displays to twinkle and lead to premature failure.

Liquid crystal displays require an oscillator drive, which is more critical than the supply voltage in most instances. Even though most units are modular in form, the failure of a decoder or driver should not be ruled out, especially when the digits are multiplexed.

Gas-type displays typically require a supply in excess of 150 V for illumination. Because overvoltage will cause metallization (blackening) of the glass surface it should be investigated. Some units have a built-in diagnostic test for the plasma display panels. If available, the test should be made before proceeding to other checks.

Any CRT display, whether storage or non-storage, must be serviced by using the manufacturer's maintenance manual, which will show how to perform any self-tests along with the other test points to be checked. The CRT may implode if it is scratched or struck severely. Do not handle the CRT by its neck and wear protective clothing and a face shield when handling it.

The optocoupler combines the emitter and detector in the same package. Therefore, the input drive must be certified first. The input device is an IRED or an LED and may be operated in a pulsed or steady-state mode. The voltage drop in the steady-state mode is more than that of a regular diode. If it is driven (sinking current) by a gate, no drop across the input diode indicates an open transistor in the gate's ouptut, an open limiting resistor, or a short-circuited diode. The pulsed mode requires the use of a scope or a probe. If the input diode is suspect, a simple diode test (with an ohmmeter) will verify whether it is functioning properly.

Repeated failures may indicate transients. Since these devices are often employed for isolation purposes they are subject to high transients. All circuit suppressors should be checked because they may be intended to protect the optocoupler from transients.

The detector side of the optocoupler may be any of the devices shown in Fig. 14-57. Many are hard to test except for a short circuit or an open condition. Substitution is the simplest course in some cases. Voltmeters are used to verify steady-state conditions, but scopes or logic probes are preferred for pulsed circuits. Triggering the scope from the input to the optocoupler should synchronize the output waveform. In a zero crossing circuit, the output should appear when the input is active and the line is near 0 V.

Fiber-optic transmitters may be LEDs or IREDs and are usually high-current pulsed types. The IRED output can be confirmed with an optical detector or a special IR-sensitive card made for detecting the beam output. The drivers, usually transistors, must be checked along with any current-limiting resistors. The oscilloscope is the most effective instrument to verify that the drive is normal. Heat sinking and cooling are very important, and any loss or reduction can cause repeated failures. The coupling to the cable must be firm and clean. The biggest loss in fiber optics is in the interface connections. Oil or other liquids may build up in loose connections. Cleaning procedures must follow the manufacturer's recommendations.

Receivers are usually sensitive to visible light, so they can be checked easily. Clean and tight connections are just as important at the receiver end as they are at the transmitter end of the cable. It is advantageous to have test units for each end of the cable. Lengths of test cable are also valuable for locating the source of problems. In most cases, a logic probe or scope will verify correct levels out of the detector modules. The transmitter and receiver constitute a system and should always be viewed as such. The idea is to isolate the difficulty to one end or the other or to the connecting cable.

Most lasers found in industry will be high-power and will involve high-voltage circuits. Along with the safety precautions for working around laser light, the high voltage supplies and circuitry demand cautious work. Measurements (other than *live*) must be made after all power is locked off and capacitors are discharged (as mentioned in other chapters). For live measurements, all test equipment must be rated in excess of any voltages to be encountered. Never work alone when high voltages are anticipated. Most units require an ignition pulse that is only present for a short period. This pulse can be of extremely high potential and can damage test equipment. Consult the manufacturer's recommendations. Helium will leach out through the glass in due time, and its loss is inevitable in helium-neon lasers. Metal-removal lasers have water cooling, vacuum, and oil pumps, along with a host of interlocks for safe operation. The use of the manufacturers' manuals is imperative before maintenance is attempted.

Phototube circuits are very dependent on bias voltage and load resistance (if used). These tubes are sensitive to visible light, and excess background light can disrupt normal operation. Cleanliness is also very important. Vapors can cause dirt and grime to collect on surfaces and desensitize the unit. When they are used with a current-to-voltage op amp, a simple current source (high-value resistor from the bias voltage) substitute for the phototube will verify the amplifier's integrity.

All PM tubes must have at least 100 V between dynodes to perform properly. A simple resistance check (with the power off, of course) will confirm the divider string. The high gain of these tubes prohibits exposure to ambient light with the dynode voltage applied and may destroy the tube if it is exposed. Never expose any PM tube to light with the supply voltage applied. A flashing LED or neon lamp produces more than enough light to test a PM tube. With 100 V per dynode, high voltages are required and breakdown can occur in wiring or sockets and must also be checked. All test instruments must be able to handle these potentials with safety.

REVIEW QUESTIONS

41. Excessive ______ can increase the dark current from a photodetector.

42. Segments of a seven-segment display may appear bad if the ______ is bad.

43. Blackening of a plasma display means the ______ is too high.

44. The best aid for troubleshooting pulsed fiber optics is an ______.

45. Laser power supplies use a/an ______ to start lasing.

CHAPTER REVIEW QUESTIONS

14-1. The combination of red and green light produces a ______ light.

14-2. The light produced by a forward-biased PN junction is due to ______ transitions in the energy gap.

14-3. ______ emitting diodes produce a very narrow spot.

14-4. Refer to Fig. 14-21(*b*). If I_P is 100 μA, h_{FE} is 100, and I_B is zero, calculate I_E.

14-5. Dark current in a phototransistor will increase ______ times for a 20° rise in temperature.

14-6. To double the light falling on a detector, its exposed area would have to be ______.

14-7. The PIN diode includes a ______ region between the P- and N-type material.

14-8. The PIN photodiode with no bias is in the ______ mode.

14-9. Refer to Fig. 14-21(*b*). A PIN diode produces 2.5 μA of photo current and zero dark current. With the op amp feedback equal to 1 MΩ, calculate the output voltage.

14-10. The light-sensitive SCR is also called a ______.

14-11. An increase in the light level produces an ______ in the resistance of a photoconductive bulk cell.

14-12. Photoconductive cells may not quickly return to their original resistance after exposure to a bright source because of their ______ effect.

14-13. A big drawback of hot-wire readouts is the large number of isolation ______ needed.

14-14. A CRT monitor may be either a storage or a ______ type.

14-15. CRT graphic displays may be of the vector or ______ type.

14-16. The information required to form characters on the screen of an industrial terminal is stored in a ROM called a ______.

14-17. The photo ______ is used for encoders and speed monitors.

14-18. A beginning of a tape sensor is typically a ______ object sensor.

14-19. Fiber optics that provide two-way communication form a ______ system.

14-20. Fiber optics operate in the tens of watts range. (true or false)

14-21. The ______ index fiber table type operates at moderate bandwidths.

14-22. Poor end polishing can cause losses up to ______ dB in fiber cable systems.

14-23. Identify an amplifier that provides a very low load impedance and thus a fast response time for a fiber-optic detector.

14-24. The gas laser operates in the far ______ region.

14-25. Injection lasers must be in a ______ array to achieve high power outputs.

14-26. The ______ is a good switch for injection laser pulses.

14-27. If δ = 5 and d = 5, the gain of a PM tube is ______.

14-28. A 10 percent change in dynode voltage results in a 10 percent change in the gain of a PM tube. (true or false)

14-29. Large pulse outputs require the last dynodes to be ______.

14-30. The PM tubes should not be exposed to ambient light with the high-voltage ______.

ANSWERS TO REVIEW QUESTIONS

1. electroluminescence 2. longer 3. F; incoherent 4. photons 5. 2.0 eV 6. 880 nm 7. visible 8. pair 9. electric 10. depletion 11. false 12. reverse 13. infrared 14. dark 15. false 16. bar/graph 17. seven 18. ac 19. MOS 20. grids 21. pixels 22. IRED 23. false; gain 24. 500 percent 25. half wave 26. full 27. false; zero voltage 28. triac 29. FET 30. Schmitt trigger 31. xenon 32. pumping 33. false 34. reflecting 35. neon 36. false 37. 1 V 38. gas-filled 39. secondary 40. positive 41. heat 42. decoder 43. voltage 44. oscilloscope 45. exciter

15 AUTOMATION AND ROBOTICS

This chapter presents programmable controllers and robots, which are key components for industrial automation. They are based on the devices, circuits, and the concepts presented earlier. This chapter will help you understand how all of the elements of electronics fit together to form powerful systems for industry. These systems have enabled our factories to increase productivity, decrease costs, and increase the quality of manufactured goods.

15-1 PROGRAMMABLE CONTROLLERS

Programmable controllers are well suited to the cyclical and repetitive operations found in sequential industrial processes. Originally, sequential control was accomplished with relays, stepping drums, timers, and counters. These systems were very difficult to reprogram for production changes and often had to be scrapped and completely redesigned from the ground up. The automobile industry was the first to change over to programmable controllers. They immediately saved money and time, and they gained reliability plus tremendous flexibility. Now, many industries use this technology to perform logic decisions, timing, up/down counting, record keeping, sequencing, arithmetic operations, report generation, information handling, debugging, and troubleshooting. Obviously, the modern programmable controller is capable of everything the old relay logic was plus a lot more.

Programmable controllers (PCs) are computers. They will be referred to in this chapter with the abbreviation PC; not all computers can be considered as PCs. To qualify as a PC, a computer must be designed to operate in the industrial environment, which can be rather harsh. It must have a wide temperature operating range (0 to 60°C) and a wide humidity range (0 to 90 percent). It must be packaged in rugged enclosures and be well shielded against electromagnetic interference, dust, dirt, and moisture. It must be capable of surviving power outages, brownouts, and line transients. Its memory circuits must be backed up with battery power. It must be capable of being programmed with logic commands, symbols, or mnemonics that correspond to relay ladder diagrams. Finally, it must be designed for scanning operation. A *scanning computer* solves logic from the beginning of memory to some specified stopping point. Once the end is reached, the operation repeats again.

Figure 15-1 shows a block diagram for a PC. The *central processing unit* (CPU) and the *input-output* (I/O) form the major core of any programmable controller. The programmer may or may not be connected to the CPU at any given time. It is required only when programs are being entered, changed, or debugged. It may also be connected during periods of system maintenance. When the CPU is performing normal scanning, the programmer may be disconnected and moved to another PC. In this way, one programmer can serve several systems. A typical programmer is shown in Fig. 15-2. Note the rugged case and the membrane keyboard, which prevents moisture and dirt entry. The keyboard folds up and protects the CRT when the terminal is not in use or is being transported. The printer shown connected to the programmer in Fig. 15-1 is an option. It provides hard copies of programs, data, reports, and ladder diagrams. The program storage unit is another option. It could be a floppy disk drive or a tape drive that connects to the programmer for saving programs and for loading in control programs or diagnostic software.

The CPU shown in Fig. 15-1 contains a processor, logic memory, storage memory, and an optional communications module. The *communications module*, if present, provides a data link to other PCs and possibly to a computer. Figure 15-3 shows the CPU of a programmable controller. The rugged cabinet has brackets for mounting the unit in a standard 48-cm (19-in.) equipment rack. The connector at the left is used to connect the programming terminal or some other RS-232C peripheral. The four diagnostic indicators tell whether the processor is running, the outputs are enabled, the CPU is in the program load mode, or there is an access in process. The 20-key

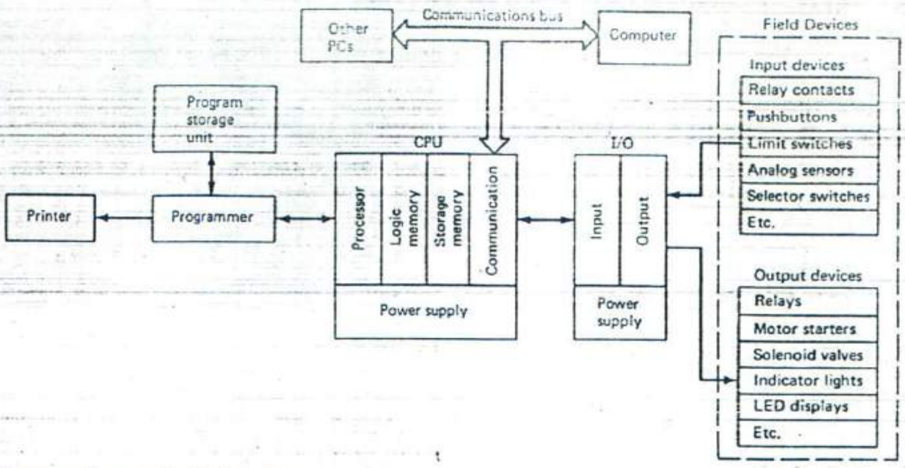


Fig. 15-1 Programmable controller block diagram.

pad and 16-character alphanumeric display can be used to read and change values such as the preset value of a timer. The key switch allows the CPU to be placed in one of three modes. One mode is *MEMORY PROTECT ON*, which allows an operator to read memory but not change it. *DATA CHANGE* allows an operator to read memory and change only selected portions. *MEMORY PROTECT OFF* allows read or write to all user memory locations.

The I/O unit of Fig. 15-1 is separated from the CPU and has its own power supply. The I/O unit buffers the CPU from the noise and transients associated with the field devices. It also prevents the control wiring, which can pick up quite a bit of noise, from interfering with the CPU. Note that the power supply shown in the I/O unit powers that unit only. It is not used to power output field devices. Programmable controller manufacturers make various I/O modules. The following list is for the Allen-Bradley PLC-3 system:

ac/dc (120-V) input
dc (12- to 24-V) input
dc (48-V) input
Isolated ac/dc (120-V) input
Analog (8-bit) input
Analog (12-bit) input
TTL input
dc (24- to 48-V) input
Encoder/counter (5-V)
Encoder/counter (12- to 24-V)
ac/dc (220- to 240-V) input

Fig. 15-2 8200 programming terminal. (*Courtesy of Allen-Bradley Co., Systems Division*)

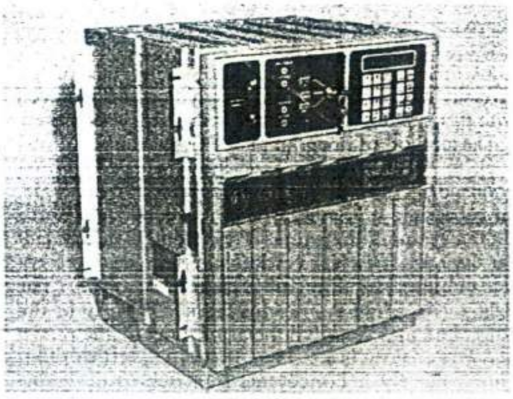
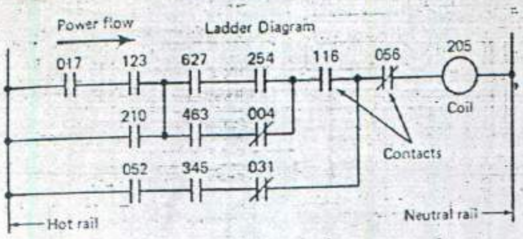


Fig. 15-3 PLC-3 programmable controller. (*Courtesy of Allen-Bradley Co., Systems Division*)



Program implementation

```
STR  017        AND   NOT   004        AND NOT  031
AND  123        OR          STR        OR       STR
OR   210        AND         STR        AND NOT  056
STR  627        AND         116        OUT      205
AND  254        STR         052
STR  463        AND         345
```

Fig. 15-4 Relay ladder-based boolean language.

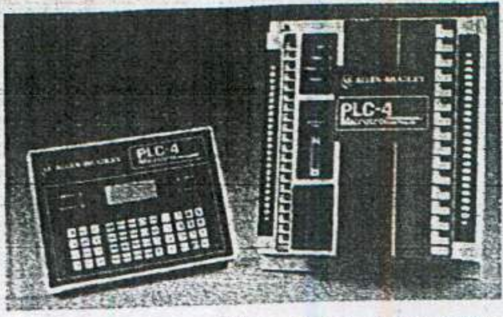


Fig. 15-5 PLC-4 programmable controller and programmer. (*Courtesy of Allen-Bradley Co., Systems Division*)

dc (5- to 30-V) selectable
dc (5-V) selectable input
Fast-response dc (12- to 24-V) input
dc (12- to 24-V) driver logic input
Thermocouple input
Thermocouple expander
ASCII
Absolute encoder input (8-bit)
PID module
ac (120-V) output
dc (12- to 24-V) output
dc (48-V) output
Isolated ac (120-V) output
Analog (12-bit) output
TTL output
ac (220- to 240-V) output
Protected ac
Contact output (4 N.O. plus 4 N.C.)
Contact output (8 N.O.)
Stepper positioning assembly
Servo positioning assembly

Look this list over. It will give you an idea of the diversity of control applications that can be achieved with this type of equipment.

The method of programming PCs is really a throwback to the old relay logic systems. *Ladder diagrams,* originally developed to represent relay logic, have become the universal language of sequential control systems. Figure 15-4 represents a typical ladder diagram. The left vertical line represents the hot side of the power line and is called the *hot rail*. The right vertical line represents the neutral side of the power line and is called the *neutral rail* or the *return rail*. Power flow is assumed to be from left to right. This is merely a conceptual approach to designing the logic. Its use helps the programmer avoid unexpected circuits in the diagram (and in the logic) called *sneak paths*. All contacts and coils are assigned addresses. Note that simple boolean statements may be used to describe the ladder diagram. Many other things can be done in a program as well. Coils may be latched so they are retained after a power failure. Latched outputs are restored to their previous ON or OFF states when the CPU resumes scanning. Coils may also be designated as *one-shots* to create references which are energized for short periods of time. Timer, counter, and arithmetic functions may also be assigned in the program. Many PCs also allow subroutines and other advanced programming techniques. Some allow more than one program to be resident in CPU memory at a time to permit production operations to change quickly from one product to another. The various features and programming details are brand- and model-dependent and cannot be covered here.

Programming in some models may be accomplished *online*. This mode allows existing programs to be changed in relation to reference numbers, timer/counter functions, contact types, latching, and so on, without disturbing the scanning operation. *Offline* programming centers around the programmer terminal unit itself. The logic program and all changes are entered into the terminal's memory with no intervention from the CPU. When the program is ready, CPU scanning is halted, and the program is transferred to the CPU memory. If the terminal's memory has battery back-up, or if disk or tape storage is available, offline programming can be done at some remote site.

The programmer terminal is also invaluable for troubleshooting. It can be used to display the status of field devices and the real-time power flow of any portion of the ladder diagram without disturbing scanning. Inputs and coils can be overridden so that their status can be forced by a human operator. The content of registers can be displayed. Some models have a special online mode in which the CPU scans, reads inputs, establishes all coil states internally, but holds all outputs in the OFF state. This mode allows the logic to be exercised with real inputs to check for proper operation without the danger of unexpected outputs.

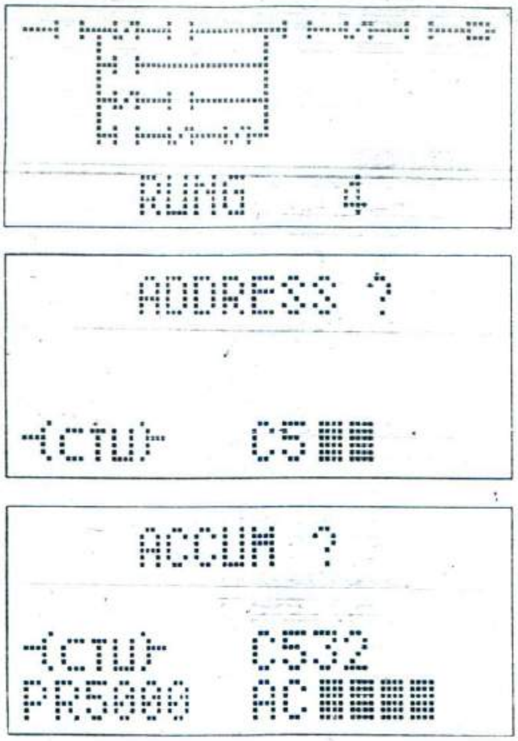


Fig. 15-6 Sample displays from PLC-4 programmer.

Large PCs have tremendous capacity. For example, the Allen-Bradley PLC-3 has 192K bytes of memory and can handle up to 8192 inputs and outputs. Not all industrial situations demand this much capacity. Small PCs are also available; Figure 15-5 shows the Allen-Bradley PLC-4 PC and programmer. This system is capable of handling 20 inputs and 12 outputs. Allen-Bradley also makes an expander module which allows up to eight PLC-4 controllers for a capacity of 256 I/O points and over 4K bytes of memory. Figure 15-6 contains some sample displays from the PLC-4 programmer. Note that rungs from the ladder diagram can be viewed one at a time on the liquid crystal display.

REVIEW QUESTIONS

1. Programmable controllers are flexible because they allow production changes by modifying _______ rather than modifying hardware.

2. What are the two main sections of a PC?

3. The programs for a PC are based on _______ diagrams.

4. Programmable controllers execute their programs with repetitive passes from the beginning of memory to the end of the program. This process is known as _______.

5. The CPU memory of a PC must be provided with _______ back-up.

6. The CPU and the I/O sections of a PC are separated to protect the CPU from _______.

15-2 ROBOT CLASSIFICATIONS AND TERMINOLOGY

There are several ways to classify industrial robots. One way is to categorize them as low-, medium-, or high-technology types. Another is to divide them into nonservo- and servo-controlled groups. They can also be divided according to their axes of movement or their system of coordinates. Yet another way is to divide them into nonintelligent and intelligent groups. All of these classification techniques have advantages and disadvantages. This section will explain the classification techniques and define the basic terms used in industry.

Four basic configurations for industrial robots are illustrated in Fig. 15-7. The rectangular robot moves along X, Y, and Z axes. As it extends and retracts to its maximum reach and its minimum reach along each of its axes, it describes a rectangle in space. This rectangle is also called its *work envelope*. The *cylindrical robot* rotates at the base and extends and retracts along Z and Y axes; its work envelope is a cylinder. The *spherical robot* rotates at its base, pivots or bends at its shoulder, and has arm extension and retraction; its work envelope is a portion of a sphere, as shown in Fig. 15-8. The *jointed spherical robot* adds an elbow joint; its work envelope is also basically spherical, as shown in Fig. 15-9. The addition of an elbow joint gives it greater flexibility and a larger work envelope than that of the spherical robot.

Each axis of motion adds another degree of freedom to a robot arm. Figure 15-10 shows a six-axis jointed spherical robot. Yaw, pitch, and roll have been added to a wrist joint at the end of the forearm. Robots with six or more axes can be classified as medium- or high-technology robots. Some robots must be capable of rather complicated combinations of movements. As an example, robot arms may be used to reach inside automobile bodies and perform welding operations. In addition to the axes shown in Fig. 15-10, horizontal base motion is achieved by mounting of the base on a moving track. This enables the robot to weld and follow the work piece down the assembly line at the same time. Also added is column extension to allow the shoulder to raise out of the base, the forearm to be extended, and the waist to be bent by means of a joint below the shoulder. Of course, it doesn't have to stop there. Almost any number of degrees of freedom can be used. However, the complexity and cost quickly get out of hand

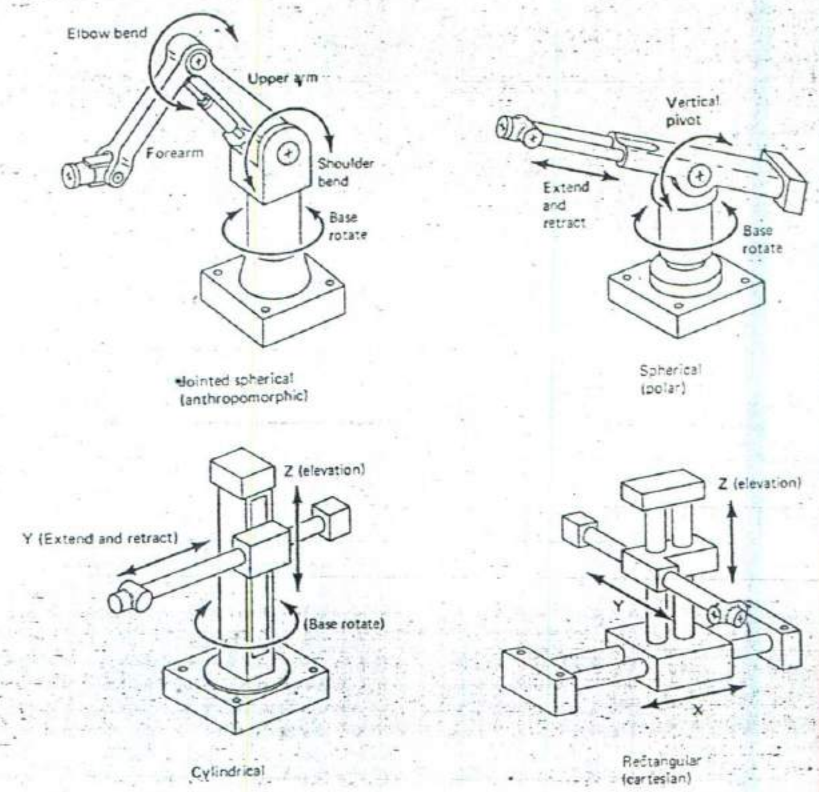


Fig. 15-7 Basic configurations for industrial robots.

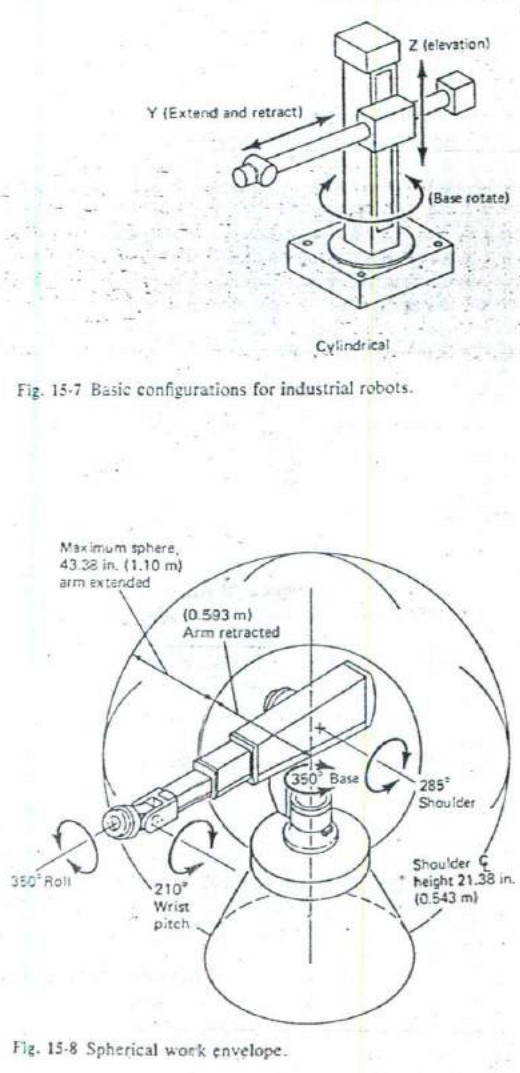

Maximum sphere, 43.38 in. (1.10 m) arm extended
(0.593 m) Arm retracted
350° Base
285° Shoulder
350° Roll
210° Wrist pitch
Shoulder ℄ height 21.38 in. (0.543 m)

Fig. 15-8 Spherical work envelope.

as the number of axes increases. Robots with more than 8 degrees of freedom are the exception rather than the rule. Most industrial tasks require from 3 to 6 degrees of freedom.

Simple robots do many routine jobs in industry. Some may have as little as two degrees of freedom. These robots are in the *nonservo-controlled* category. There is no feedback used in their control system. They are often called "bang-bang" robots because of the way they work. They bang from position to position, as shown in Fig. 15-11. They are set up for a task by adjusting fixed stops. They are excellent robots for pick-and-place operations. In fact, they are often called *pick-and-place robots*. They are also useful in simple assembly operations such as stuffing printed circuit boards and loading and unloading parts from machines.

Figure 15-12 shows a typical low-cost robot, and Fig. 15-13 shows some examples of its motion patterns. It uses pneumatic cylinders to activate the axes. The following list includes some of the ways that the robot of Fig. 15-12 can be classified:

1. *X*, *Y*, *Z* type
2. Non-servo-controlled type
3. "Bang-bang" robot

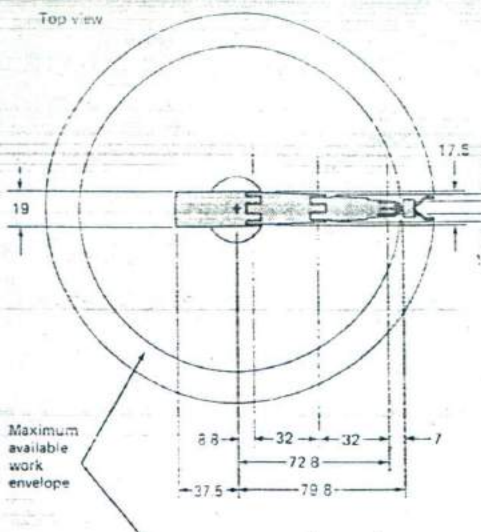

Top view
19
17.5
7
12
Maximum available work envelope
8.8
32
32
7
72.8
37.5
79.8
Side view
20
44.5
52
Floor line with standard base mounting
36.5
Min. retraction with axes limits
8.8
32
32
7
64
79.8
(All dimensions are in inches)

Fig. 15-9 Work envelope of a jointed spherical robot.

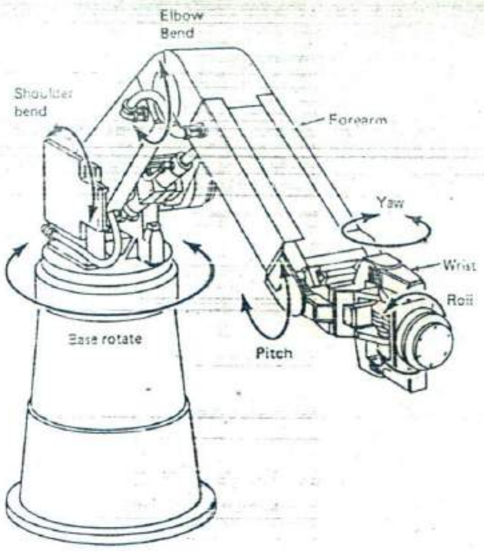


Fig. 15-10 Six-axis jointed spherical robot.

4. Low-technology type
5. Three-axis robot
6. Pneumatic robot
7. Pick-and-place robot
8. Rectangular-coordinate robot
9. Limited-sequence robot

There is a problem with using any single classification system. Look again at Fig. 15-11. This robot is also in the nonservo-controlled and low-technology categories. However, it is a jointed spherical arm and has some differences that make it suited to a different range of applications. It may use different actuators (other than pneumatic) and have very different speed and strength characteristics. You are urged to learn the terms and use them in combinations to describe any robot accurately. Another trap to avoid is the assumption that low-technology robots are limited in accuracy and usefulness. Actually, they tend to be the most accurate of all robots. They can be expanded with more degrees of freedom to perform fairly intricate tasks. Figure 15-14 shows the addition of roll, pitch, and yaw to the low-technology robot of Fig. 15-12 and Fig. 15-13. In a sense, the term *low technology* is unfortunate: it can cause people to reach erroneous conclusions about robot capabilities and has even caused some companies to install high-technology robots in applications in which less expensive robots would have done the job faster and better.

The general attributes assigned to *low-technology robots* include the following:

1. Limited sequence of movements
2. Only the endpoints of travel are controllable
3. A speed range of 80 to 160 cm/s
4. Low cost and low maintenance
5. A 0.03-mm resolution and 0.15-mm repeatability
6. Load capacities from 150 g to 15 kg
7. Controls ranging from electromechanical timers to microprocessors
8. Usually 2 to 4 degrees of freedom
9. Nonservo-controlled
10. Short cycle time (typically 30/min)
11. Used in parts transfer, assembly, loading, packaging, inspection, and automatic testing

Medium- and high-technology robots are servo-controlled. *Servo control* can be divided into two types: *point-to-point* and *continuous path. Point-to-*

point servos move from one work envelope position to the next in a straight-line segment. They exhibit jerky motions; however, more than one axis can be activated at a time to alleviate this somewhat. *Continuous path servos* produce true curves as the work envelope is traversed. They are useful in applications such as spray painting and welding. They require more memory to store all the positions needed to follow a smooth path.

The general attributes assigned to *medium-technology robots* include the following:

1. Point-to-point servo control
2. Up to 6 degrees of freedom
3. Payloads up to 70 kg
4. Longer cycle time than that of low-technology types
5. Less accurate than low-technology types
6. Electrohydraulic actuators typically used
7. Electronic controls (typically microprocessor-based)
8. Walk-through programming
9. Medium cost and maintenance requirements

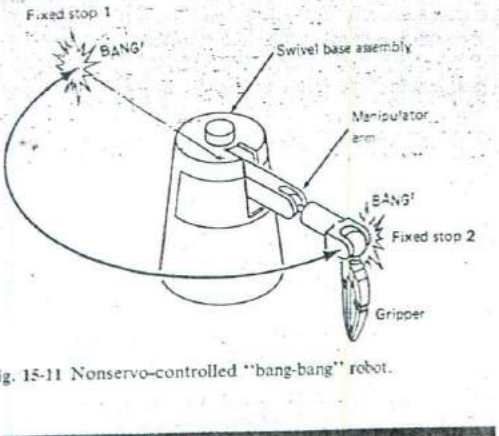


Fig. 15-11 Nonservo-controlled "bang-bang" robot.

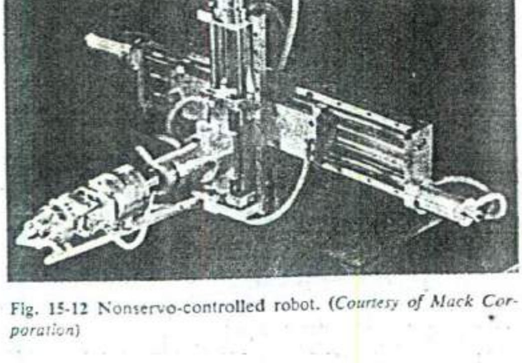


Fig. 15-12 Nonservo-controlled robot. (*Courtesy of Mack Corporation*)

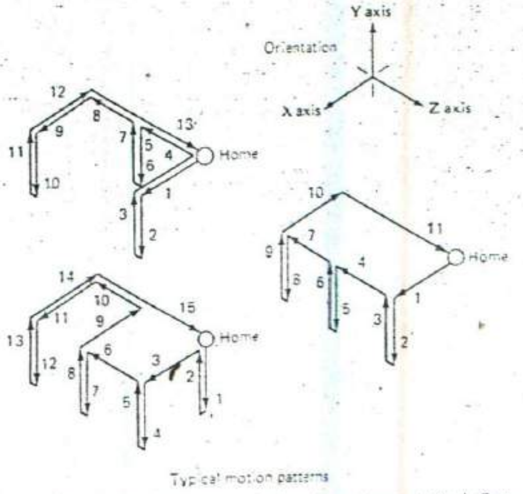


Fig. 15-13 Robot parts and typical motion patterns. (*Mack Corporation*)

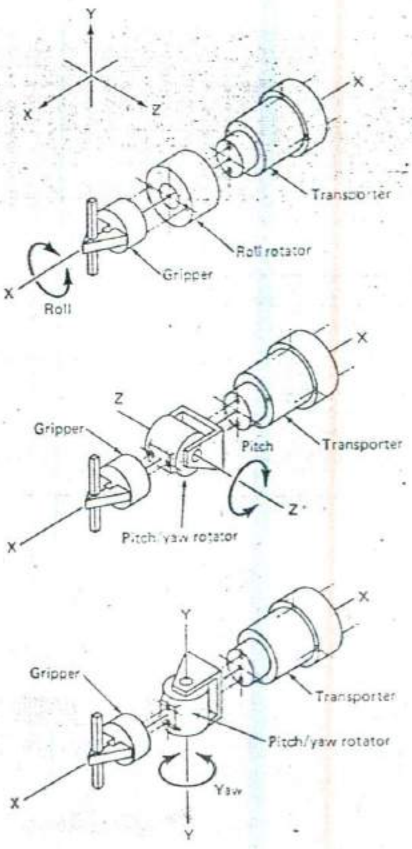


Fig. 15-14 Adding roll, pitch, and yaw.

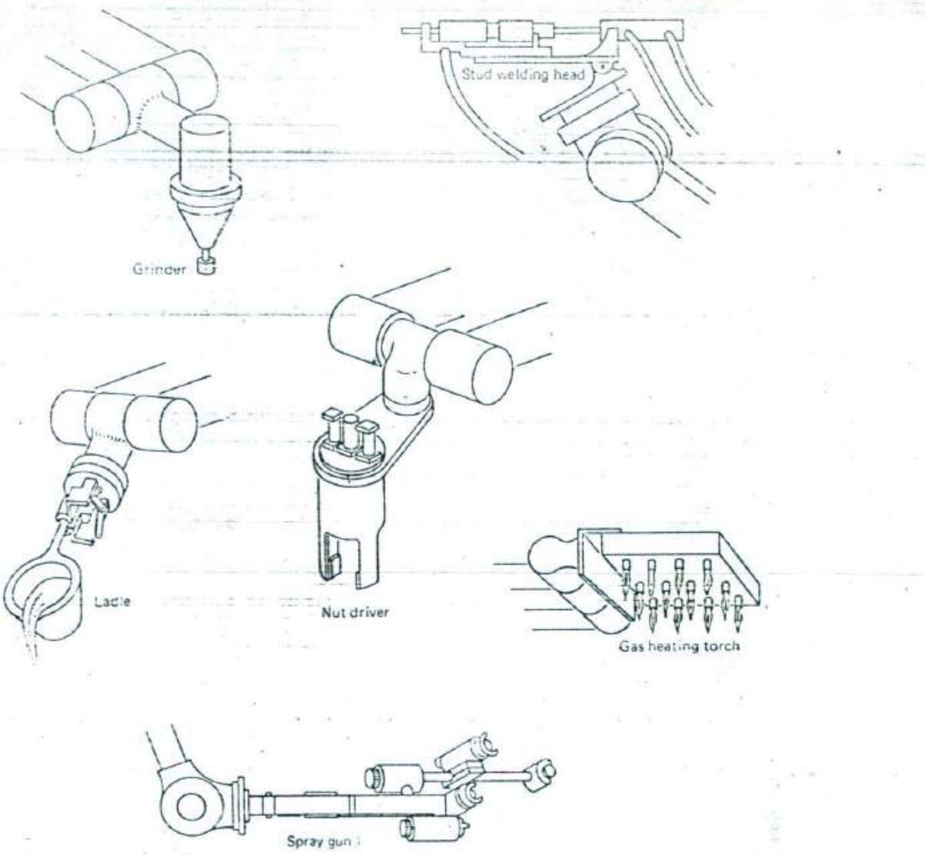


Fig. 15-15 Robot end effectors.

Walk-through programming involves a human operator leading the robot through the desired routine step by step. It will be covered in more detail in a later section.

High-technology robots are the most complicated. Their attributes include the following:

1. Continuous path servo control
2. 6 to 10 degrees of freedom
3. Microprocessor and minicomputer controls
4. Expensive and increased maintenance requirements
5. Most flexibility of all robots
6. Advanced programming, including high-level languages
7. Built-in editing and diagnostic software packages
8. Mass storage devices (disk and tape)
9. Looping, subroutine, interrupt, and branching capabilities
10. Control of some other machines around them
11. Sensor inputs such as vision, proximity, touch, and sound
12. Applications include sorting, inspection, welding, painting, and assembly

Robot actuators include pneumatic cylinders or motors, hydraulic cylinders or motors, and electric motors. The *pneumatic actuators* are low in cost, require little maintenance, and are well suited for high-speed operations with light payloads. *Hydraulic actuators* are more costly, require more maintenance, and are best suited to heavy loads. *Electric motors* are the easiest to control, have moderate cost and maintenance requirements, and lend themselves to applications that do not require high speed or heavy loads.

The robot arm may be called a *manipulator*. The part that handles the work piece or holds the tool is often referred to as an *end effector*. Figure 15-15 shows some examples of end effectors. These may be changed automatically, as shown in Fig. 15-16. The end effectors are stored in a tool holder within

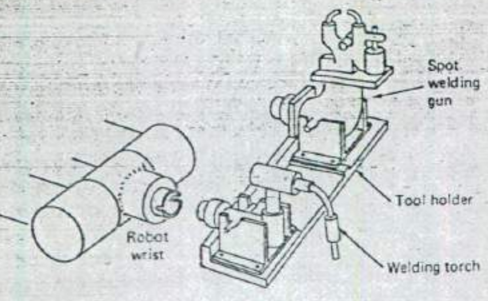


Fig. 15-16 Robot tool changing.

the manipulator's work envelope. One tool is placed in the holder, and the wrist disengages from it. The manipulator then moves to the next tool and engages the wrist. Programmed tool changing is usually limited to high-technology robots. Fingerlike grippers are also common end effectors. There are a myriad of gripper designs to handle the many tasks being assigned to robots today.

REVIEW QUESTION

7. The perimeter of a robot's reach describes a shape in space called its ______ ______.

15-3 PNEUMATICS AND MECHANICS

The industrial electronic technician must have a well-rounded understanding of several technologies. Automation involves more than electronics; fluidic and mechanical systems interact with the electronic controls. The term *fluidics* covers both hydraulics and pneumatics. *Mechanics* includes gears, pulleys, bearings, transmissions, chain drives, timing belts, and so on. The technician must have an overall understanding of what each component in a system accomplishes and how each component interacts with the other components.

Pneumatic systems are similar to hydraulic systems. The major difference is that pneumatics uses air instead of a liquid to transmit power. Air compresses, limiting the amount of force that can be transferred. For this reason hydraulic systems are preferred when large loads must be handled. However, there are many benefits to pneumatic systems. Compressed air, once it has delivered its potential energy to a cylinder or a motor, can be returned to the atmosphere. This is a key advantage since it prevents the build-up of contamination in the system. Air is also a coolant. Pneumatic motors, for example, are cooled by the air that drives them. Compressed air is easy to produce and can be distributed to various machines at low cost. The pressures used are often not more than 150 psi (approximately 1 million N/m^2), and distribution is inexpensive and relatively safe.

Figure 15-17 shows a *pneumatic actuator;* it consists of a cylinder with a port at each end and a piston to drive the output shaft or shafts. A *single-ended actuator* has an output shaft at one end; a *double-ended actuator* an output shaft at each end. When compressed air is applied to one port, the piston reacts accordingly. The *four-way spool valve* provides the control. With the spools in the position shown in the drawing, the compressed air inlet is blocked, and no air is delivered to the actuator. If the control handle is pushed down, the spools will lift; compressed air will flow to port *B*, and the piston will be forced to the left. At the same time, port *A* will be vented through the spool valve to the atmosphere. Raising the control handle produces the opposite reaction in the cylinder, and the piston moves to the right. The two motions make this a *double-acting cylinder.*

Reversing air motors are also available and are controlled in the same way. They may have eight or so vanes connected to an output shaft. There are two input ports. The direction of rotation depends upon which port is pressurized and which provides the exhaust.

The spool valve can be controlled by a solenoid; this solenoid can then be controlled by a microprocessor. Microprocessor control greatly improves the performance of air motors. This is the system used in the robot arm shown in Fig. 15-18. Air motors tend to give rough performance at low speeds. The microprocessor control smooths this out by digitally pulsing the air supply to the accumulator of each air motor. This technique is called *dithering* and provides performance characteristics once considered beyond the capabilities of air motors. There are five air motors in this particular manipulator, one for each major axis. Each is controlled by a dedicated 8-bit

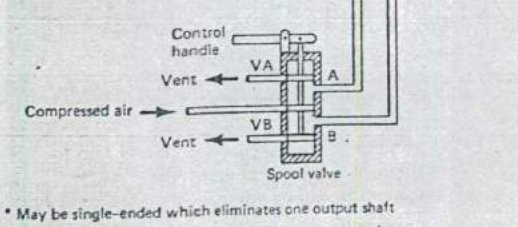


Fig. 15-17 Pneumatic actuator.

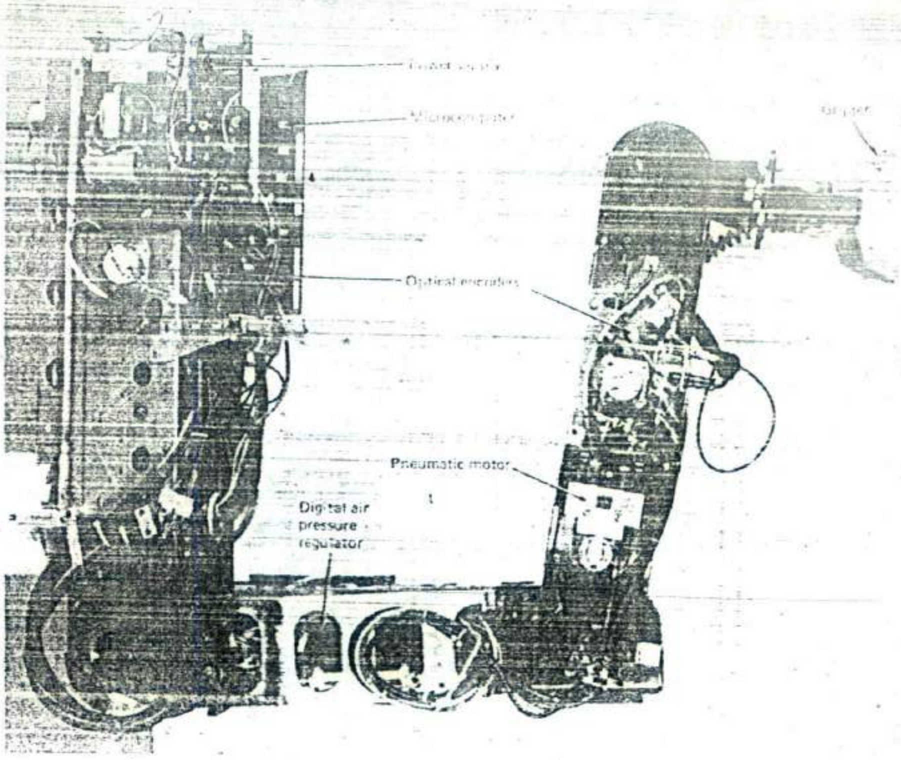


Fig. 15-18 IRI M50 robot arm. *(Courtesy of International Robotmation Intelligence)*

microprocessor which drives a spring-centered spool valve. The microprocessors dither the air supply to their respective motors by adjusting the frequency and duty cycle of the pulses supplied to the solenoid-activated valves. Optical encoders provide feedback information to allow the microprocessors to provide accurate positioning and to adjust the motor performance for varying conditions of friction and gravity under different load conditions.

Figure 15-19 is a diagram of the pneumatic system for a model M50 robot. In this robot, an equipment module contains a pressure control valve for each axis. Packing the air motors and the valves together eliminates transients due to the compressibility of air. A *control valve assembly* is composed of a solenoid-activated spool valve and a two-position valve that operates at pilot pressure when the solenoid is activated. A *volume chamber accumulator* is mounted on each pressure control valve assembly. The *pressure control valve* dithers the air supply to the volume chamber accumulator to maintain a constant pressure supply to the motor under varying conditions.

Air motors are high-speed, low-torque actuators. Some type of speed reduction system is required for robotic applications. Figure 15-20 shows the chain drive stages of the M50 robot. They step down the 8000-rpm motors to 10 rpm at the output shaft. The chain drives provide a bonus function. The pulsed output of the air motors appears smooth and continuous after translation through the reduction chains. Each chain must be adjusted to the proper tension to avoid backlash and to avoid stretching and excessive wear. A *chain tensioning adjustment* is provided for each chain-driven module. Two pneumatically operated *caliper brake assemblies* engage the first sprocket in each chain module. A *control valve* regulates the air pressure supplied to each brake. The modules also contain the *optical encoders* that provide axis position feedback to the microprocessors.

In addition to the five 8-bit axis microprocessors, the M50 robot also contains one 16-bit microprocessor. The 16-bit processor provides commands to the individual axis processors. The robot can be programmed by walk-through commands, assembly language, or high-level languages. Eight digital I/O ports are available on the main processor for interfacing the robot to other equipment in the factory.

Ordinary belt drives do not find much application in robotics and other positioning systems because of slippage. However, *timing belts* are used in some cases. A timing belt has teeth and drives or is driven

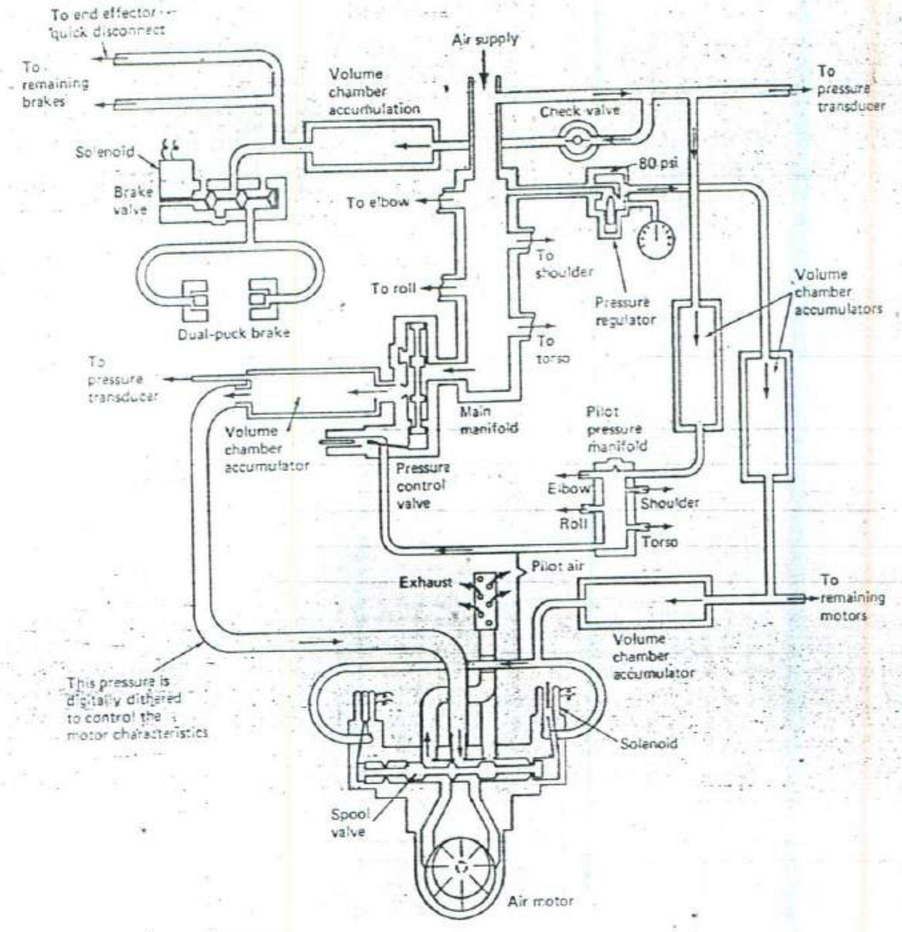


Fig. 15-19 Pneumatic system of the M50 robot.

by a pulley which also has teeth. The teeth eliminate the slippage problem. Timing belts are usually applied where the forces are small to moderate.

Robots and other automation devices often require a way to convert rotary motion to linear motion. Figure 15-21 shows a *rack and pinion* arrangement that accomplishes this task. The teeth on the pinion gear engage the teeth on the rack. If the pinion shaft is driven by a motor, the rack will move in a straight-line path. Rack motion can be reversed by reversing the motor. Figure 15-22 shows a *ball bearing screw drive* which accomplishes the same objective. The full name is seldom used, and they are usually referred to as *ball screws*. Ball screw drives are more expensive than rack and pinions, but they provide better positioning accuracy because backlash is controlled. Also, the balls exhibit rolling friction, which is less than sliding friction. This results in a more efficient drive. The return tubes shown in Fig. 15-22(a) recirculate the balls in the ball nut. Figure 15-22(b) and Fig. 15-22(c) show that the screw or the nut can be driven, respectively. Figure 15-23 shows some applications for ball screw drives. They are also widely applied in numerically controlled machines such as lathes and milling machines.

Chain drives and timing belts can be used to reduce speed and increase torque. They are subject to some elasticity and backlash effects, however. Gear boxes may be used to control the elasticity, but backlash is still a problem. The harmonic drive system shown in Fig. 15-24 provides large reductions in speed with essentially zero backlash. It is also more compact than multistage gear boxes. These advantages make the harmonic drive system very attractive for robotic and other automation applications. The drive is composed of a rigid circular spline, a flexible spline (flexspline), and an elliptical wave generator. The *reduction ratio* of a harmonic drive is determined by the number of teeth on the circular spline divided by the difference in the number of teeth between the

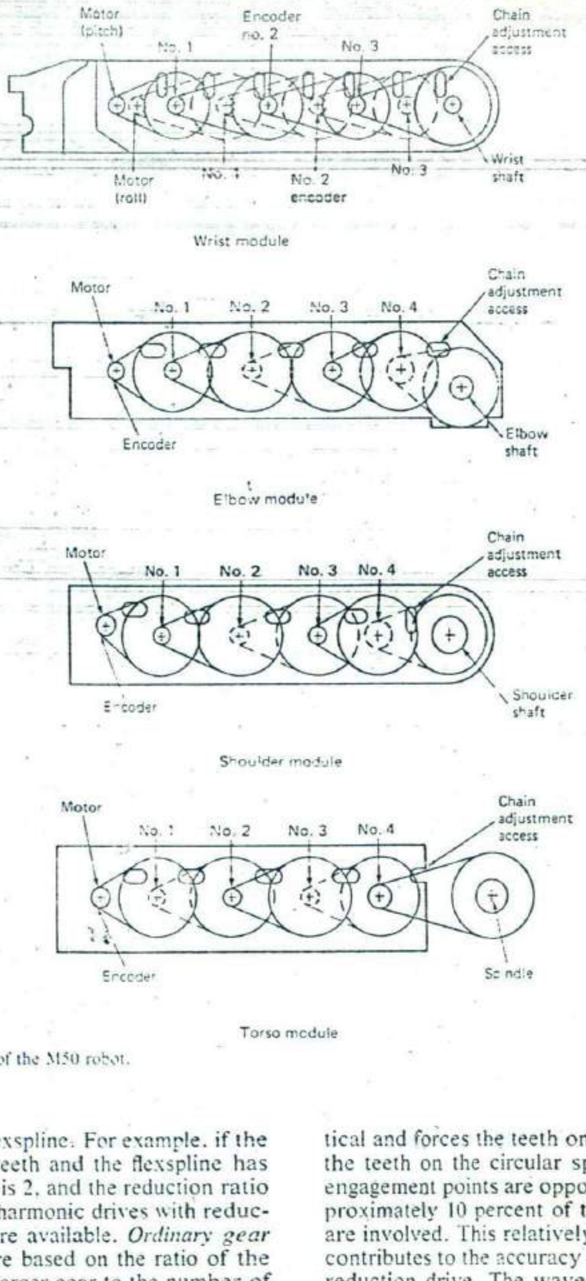


Fig. 15-20 Chain drive stages of the M50 robot.

circular spline and the flexspline. For example, if the circular spline has 300 teeth and the flexspline has 302 teeth, the difference is 2, and the reduction ratio is 150 to 1. Single-stage harmonic drives with reductions of over 300 to 1 are available. *Ordinary gear drive reduction ratios* are based on the ratio of the number of teeth on the larger gear to the number of teeth on the smaller gear. This means that large reduction ratios require multistage gear boxes. The backlash is additive in the multiple stages, and so are the friction losses. The harmonic drive, although costly, is used in some applications because of its marked advantages.

The wave generator, shown in Fig. 15-24, is elliptical and forces the teeth on the flexspline to engage the teeth on the circular spline in two places. The engagement points are opposite one another, and approximately 10 percent of the total number of teeth are involved. This relatively large tooth engagement contributes to the accuracy and zero backlash of this reduction drive. The wave generator is coupled to the input shaft of the transmission. As it turns, the points of engagement move. If the numbers of teeth on the two splines were the same, no relative motion would be produced between them. Because there is a slight difference in the number of teeth, a small relative motion is created between the two splines for every rotation of the wave generator. If the

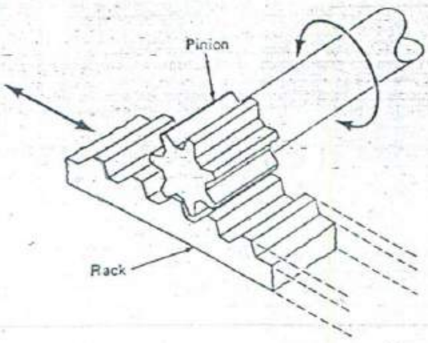
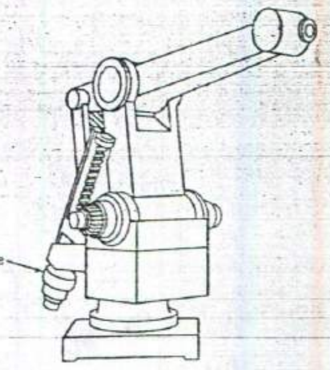


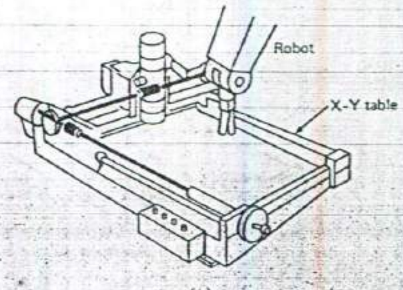
Fig. 15-21 Rack and pinion system.



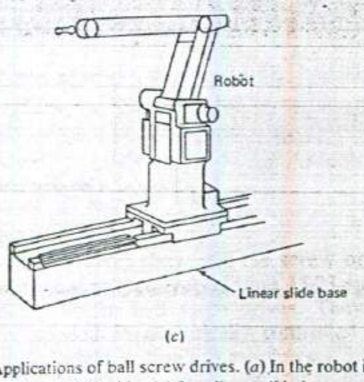
(a)

flexspline is fixed, this small relative motion appears at the circular spline and becomes the reduced speed output.

A moving robot arm has kinetic energy. If the combined weight of the arm and its payload is substantial, the energy is substantial. If the arm is moving fast, then even more kinetic energy is involved. Some robots use shock absorbers to provide controlled deceleration and to prevent damage due to hammering. The absorbers are usually cylindrical units with an input shaft. The input shaft is connected to a piston inside the cylinder. When the shaft is moved, the piston acts against hydraulic fluid inside the cylinder. The piston pumps the fluid through restricted passages. The restricted flow inhibits motion of the input shaft. The more rapid the motion, the more opposing force is developed by the piston working against the restricted fluid flow. The fluid will heat as the piston forces it to flow. This action converts the kinetic mechanical energy of the robot arm to heat energy.

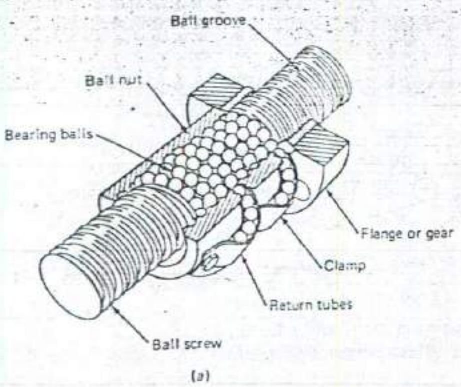


(b)

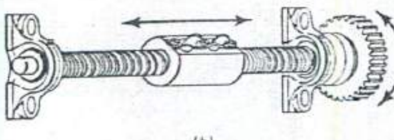


(c)

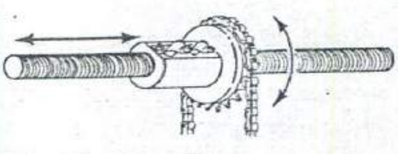
Fig. 15-23 Applications of ball screw drives. (*a*) In the robot itself. (*b*) In the axes of an *X-Y* table. (*c*) In a linear slide base.



(a)



(b)



(c)

Fig. 15-22 The ball bearing screw drive. (*a*) Screw and nut assembly. (*b*) Driving the ball screw. (*c*) Driving the ball nut.

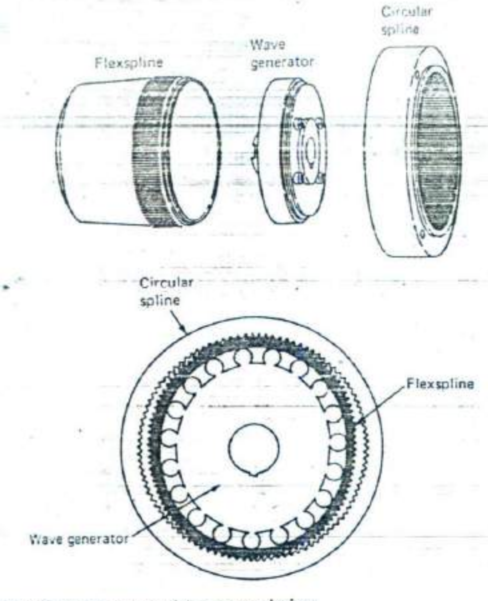


Fig. 15-24 Harmonic drive transmission.

REVIEW QUESTIONS

8. The term fluidics includes both _______ and _______.

9. Examine Fig. 15-17. Which way will the piston move when the control handle is lifted?

10. Is the arm shown in Fig. 15-18 an example of a servo-controlled or a nonservo-controlled robot? Why?

11. Digital pulsing of the supply to an air motor is called ___ ___.

12. Drive belts are smooth: timing belts have _______.

13. The chain drives shown in Fig. 15-20 produce a _______ reduction.

15-4 CONTROL AND PROGRAMMING

Robot controllers range from simple electromechanical systems to powerful minicomputers. Figure 15-25 shows an example of a simple control for a low-technology robot. An electromechanical timer has a series of cams that activate microswitches. The microswitches control four-way solenoid valves to activate pneumatic cylinders for the *X* and *Y* axes and the gripper. Mechanical stops are set to limit the travel along the axes. The gripper has no stops and is either full open or closed on the work piece. Setting up this robot for a task involves adjusting the timing and sequence of the cams and adjusting the mechanical stops. Thus, the robot is not programmed for a task: rather, it is adjusted.

Programmable controllers can be used with low-technology robots, as shown in Fig. 15-26. The *X*, *Y*, and *Z* axes are controlled along with the gripper. There is also an intermediate stop cylinder that works in conjunction with the *Z* axis. The total number of outputs is 5, which is easily within the capacity of a small programmable controller. Other outputs may be required to move the work piece out of position after the robot completes its cycle and to bring the next work piece into position. Inputs on the controller will normally be used to determine the proper position of the work piece before the cycle is initiated. The inputs may also include operator commands or commands from another computer controlling the overall production process. Another way to use inputs is to read limit switches that signal manipulator position. The following sequence is typical for a nonservo-controlled robot and will help you understand the programming steps:

1. When the program starts executing, the controller initiates signals to the valves on the manipulator's actuators.
2. The valves open and admit air or hydraulic fluid to the actuators. The affected members begin to move.
3. The valves remain open, and the members continue moving until the stops are contacted.
4. Limit switches signal the controller that the move or moves are complete. The controller closes the valve or valves.
5. The next sequence is initiated. The process is repeated until the entire program is complete.
6. A new scan is started by the controller, and the entire program runs again when the inputs signal that the next work piece is in position.

Programmable controllers may also be used with servo-controlled robots. The following sequence is typical for a medium-technology robot:

1. The controller addresses the memory location of the first command and reads the encoders to determine the actual position of the manipulator axes.
2. The two sets of data are compared, and any error is amplified and becomes a command signal to the axis motors or servo valves.
3. The actuators move the manipulator and the encoders (or other sensors) send data back to the controller. The feedback signals are constantly compared with the programmed position data; new error signals are developed and become command signals to the actuators.
4. The process continues until there are no errors. The manipulator comes to rest.
5. The controller now accesses the next memory location and responds accordingly to the data

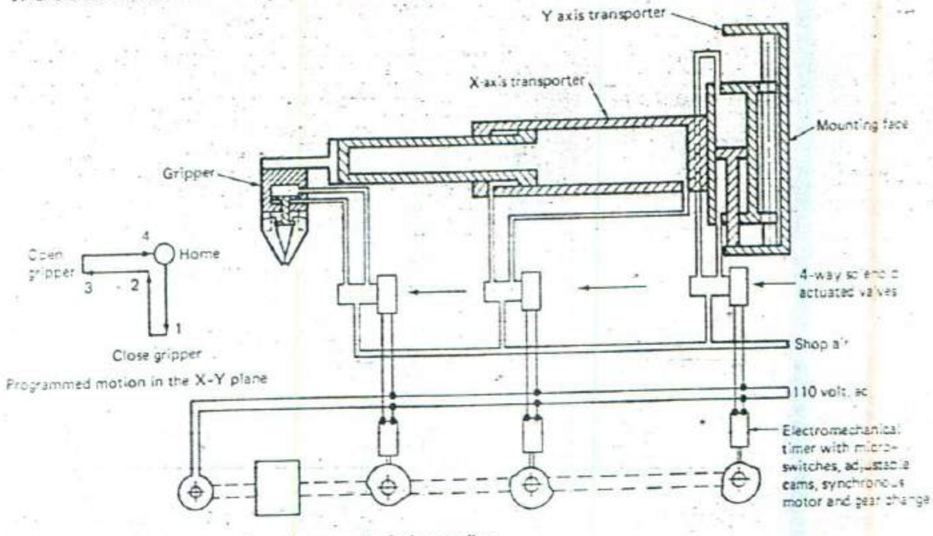

Y axis transporter
X-axis transporter
Mounting face
Gripper
Open gripper
Home
4
3
2
1
Close gripper
Programmed motion in the X-Y plane
4-way solenoid actuated valves
Shop air
110 volt, ac
Electromechanical timer with micro-switches, adjustable cams, synchronous motor and gear change

Fig. 15-25 Basic schematic diagram of an electromechanical controller.

stored there. It may be a new position command or a signal to another mechanism.

6. The process is repeated sequentially until the entire program has been executed.
7. Given the proper input conditions, the program will automatically execute again from the beginning.

Servo control is obviously more sophisticated and allows the manipulator to stop at any point in its

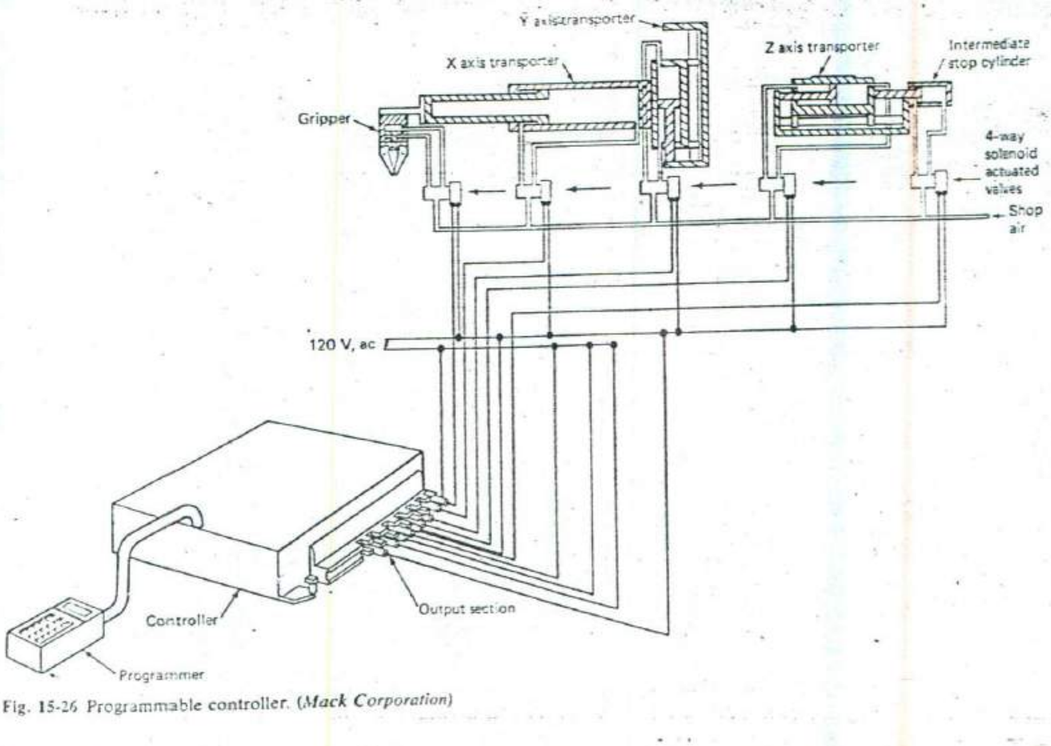


Fig. 15-26 Programmable controller. (*Mack Corporation*)

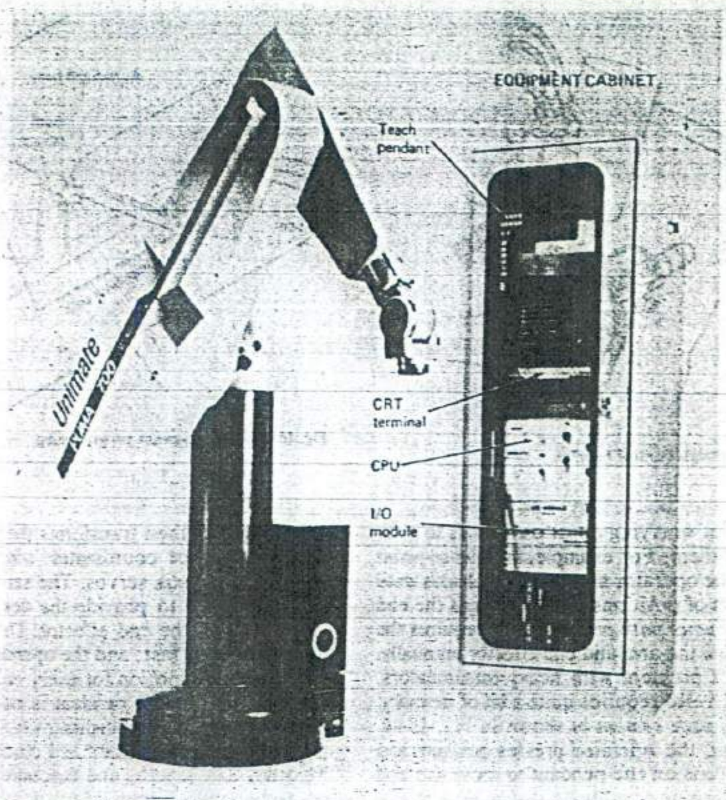


Fig. 15-27 PUMA 700. (*Courtesy of Unimation, Inc.*)

work envelope under software direction. It is also feasible to control velocity, acceleration, and deceleration with the software. This capacity is especially useful with heavy payloads. The memory capacity is usually large enough to store many positions with both point-to-point and continuous path operation as possibilities. It may also be large enough to store several programs, thus allowing for rapid production changes.

A high-technology robot and its minicomputer-based controller are shown by Fig. 15-27. The equipment cabinet stores the teach pendant when it is not in use and also houses a CRT terminal, the central processing unit, the I/O module, and the power supply for the control electronics. The *teach pendant* is one way of programming medium- and high-technology robots. A human operator holds the pendant and uses it as a remote control device to lead (usually at less than operating speeds) the manipulator through a series of moves. This kind of teaching or programming is called *lead-through*. A store button on the pendant is pressed at the end of each motion to store the position in memory. *Walk-through* is another robot programming technique. The human operator grasps the arm and the end effector and manually moves them. The operator signals the computer to record each motion or position. Walk-through is commonly applied in continuous path robots for teaching operations such as spray painting and welding. Figure 15-28 shows some typical walk-through control handles on a painting robot.

Lead-through programming must be done with some care, as indicated in Fig. 15-29. The spot welding end effector is shown in its start position. The arm must be programmed to move past the roof column and then into the window opening where the welding operation is to take place. If an operator fails to record the intermediate point in memory, the robot will take a direct path when the program runs. This will result in a collision and damage to the work piece and to the robot.

When minicomputers are interfaced to robots, another control method becomes available. It is called *controlled path* and combines the best features of point-to-point and continuous path programming. It takes advantage of the computational abilities of the computer to provide coordinated control of all axes. The operator uses a pendant to move the end effector to various end point positions. The computer then generates a controlled path at the desired velocity, acceleration, and deceleration. Teaching is more instinctive if the operator can specify tool or gripper

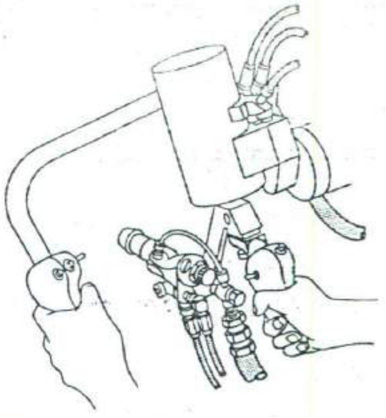


Fig. 15-28 Walk-through control handles.

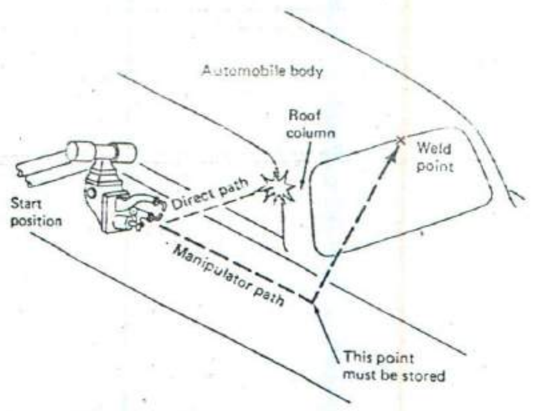


Fig. 15-29 Point-to-point programming.

positions without worrying about commands to each axis to achieve them. For example, in *point-to-point programming* the operator must move each axis until the combination of positions properly aligns the end effector. *Continuous path programming* requires the operator to move the arm and end effector manually. This becomes a problem with heavy manipulators. Continuous path also requires quite a bit of memory.

A *controlled path system* is shown in Fig. 15-30. During teaching, the operator presses position and orientation buttons on the pendant to move the end effector. The teaching coordinate system may be any of several types. Continuously changing position signals are generated as long as the buttons are pushed. The computer transforms the teaching coordinates to rectangular coordinates (if the teaching coordinates are rectangular, this operation is not required). The computer then transforms the rectangular coordinates to robot coordinates, which become commands to the axis servos. The servo loops then act simultaneously to provide the desired position and orientation of the end effector. The computer operations are very fast, and the operator sees an apparent immediate motion for every command. The program button on the pendant is pushed to store the current rectangular coordinates in memory. The pendant also has a keyboard and other buttons to enter velocity, tool length, and functions to be performed at each point. The CRT displays the information being programmed.

The CRT terminal shown in Fig. 15-30 also provides editing features. The operator may view the data stored at the previously programmed points. Data may be changed, deleted, or restored. Points

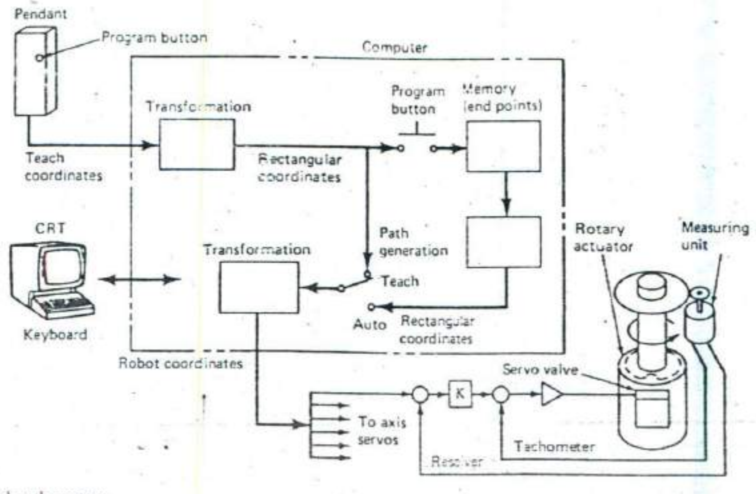


Fig. 15-30 Controlled path system.

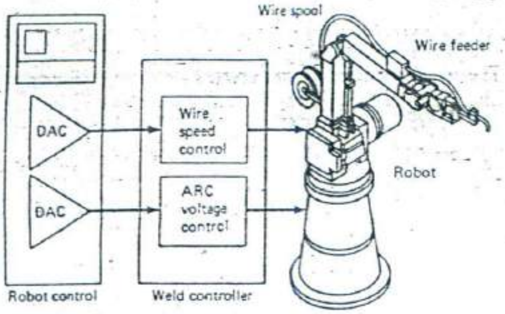


Fig. 15-31 Integrating robot control with other parts of the system.

between programmed points and program functions can be added. Some typical functions are the following:

1. DELAY: This function stops the robot for a defined period of time before moving to the next programmed point.
2. CONTINUE: The manipulator does not stop but moves to the next point. An output signal can be programmed at the continue point if desired.
3. WAIT: Motion is stopped until a signal or a conditional set of signals is received.
4. OUTPUT: Motion is stopped, and an output signal is produced before proceeding to the next point. The signal may be a pulse or a level.
5. TOOL: Motion is stopped until a tool operation is completed.
6. BRANCH: Motion is stopped, and the robot awaits input conditions. It jumps to different program segments or subroutines based on the combination of signals received.

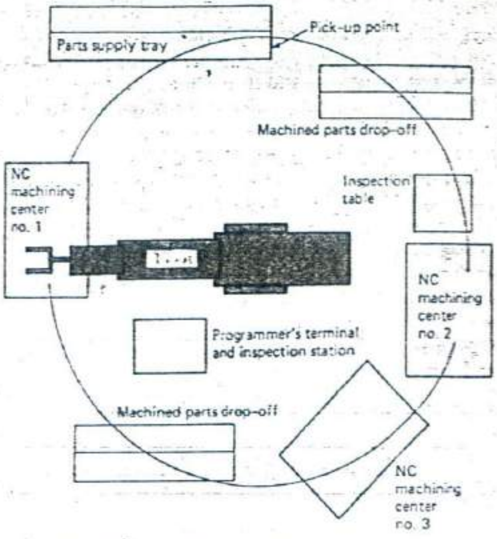

Parts supply tray
Pick-up point
Machined parts drop-off
NC machining center no. 1
Inspection table
NC machining center no. 2
Programmer's terminal and inspection station
Machined parts drop-off
NC machining center no. 3

Fig. 15-32 Flexible manufacturing cell.

Robots are one part of the automation puzzle. Figures 15-31, 15-32, and 15-33 provide examples of the robotic/automation relationship. Few, if any, industrial robots can work effectively without being interfaced to other equipment. Figure 15-31 is an example of a robot controller equipped with D/A converters to provide output signals to the welding controller, which in turn controls the wire feed and the arc voltage. Figure 15-32 shows a robot integrated into a flexible manufacturing cell. The robot's computer provides signals to, and accepts signals from, the

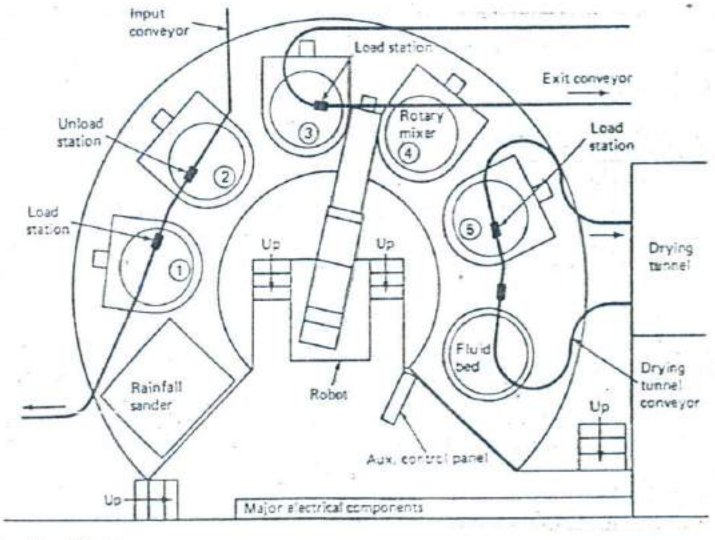


Fig. 15-33 Automated mold making.

three numerically controlled (NC) machining centers. All of the machines communicate through the computer to provide an automated flow of machining steps, parts transfer, and inspection.

A robot used in automated mold making is shown in Fig. 15-33. The process involves coating a wax pattern by dipping it into a ceramic slurry. Different slurries are available at the five rotary mixers shown. After dipping, the mold must be coated in the rainfall sander or in the fluid bed. Next, it goes into the drying tunnel. After multiple applications of slurry and sand, the mold is dried one last time and loaded onto the exit conveyor. The robot, through its computer, controls the entire process. This includes turning the mixers on and off, operating the conveyors, and controlling the sanders.

The capabilities of high-technology robots are as much due to their software as to their hardware. Looping, branching, conditional testing, and interrupts are examples of software features that make the robots "intelligent." For example, a subroutine that provides an appropriate response can allow a robot to recover from a difficult situation. Suppose the tips of a spot welding gun become stuck to the work piece. When the program detects the problem, it branches to a subroutine of twisting motions designed to free the tips. It will attempt this several times before shutting the robot down and alerting a control operator, as well as signaling the other machines around it. Robot decision making is implemented in other ways. When a robot is used for palletizing (stacking), deceleration is often programmed into the cycle. The manipulator slows down as the object nears the stacking point. Then, a sensor detects contact with the stack to stop the arm and release the payload. What happens as the stack grows? The deceleration does not begin soon enough, and damage may result. The software can take care of this possibility. The last stacking position is used to replace the destination point in memory on every cycle. In this way, deceleration will begin earlier and earlier as the stack grows.

REVIEW QUESTIONS

14. Refer to Fig. 15-25. How would the sequence of operations be changed with this control system?

15. How would the sequence be changed with the system shown in Fig. 15-26?

16. How many positions can the robot shown in Fig. 15-26 stop at along the Y axis? The Z axis? Why is this so?

17. What type of robot allows stopping at many positions along each axis?

18. Using a teach pendant to move a manipulator from point to point is known as ______ teaching.

19. Grasping the manipulator and moving it manually from position to position is called ______ teaching.

15-5 SENSORY SYSTEMS

The term *hard automation* refers to special purpose machines performing specific repetitive tasks in an automatic fashion. Many industrial activities fit this pattern and do not require sophisticated or intelligent machines to support them. However, there is an expanding effort to move toward "intelligent automation." There are two key components in this effort: sensory systems and artificial intelligence. Most industrial robots paint, weld, load and unload parts, grind, drill, or perform simple assembly operations. By the addition of sensors and intelligent software, the applications can expand to complicated assembly operations, inspection, part identification and sorting, and adaptability to a range of situations. Another important aspect is the ability to change rapidly from one production run to another. When this is realized, there is no longer a need to maintain an inventory of spare parts because they can be quickly manufactured on demand.

Sensors may be used to provide touch, tactile, proximity, vision, or sound information. Touch and tactile sensing are related but are not the same. *Touch sensing* involves contact detection at one or several points. *Tactile sensing* measures continuous and variable forces in an array; it is similar to the type of feedback people obtain from their skin. People also use *haptic sensing*, which comes from the muscles and joints. Joint angles and muscle forces provide the brain with quite an array of information during manipulative operations. Robots may crudely approximate haptic sensing by measuring motor current or fluidic back pressure as an indicator of the forces acting on an axis. This procedure gives an indirect measurement at best and is considerably influenced by manipulator position, friction, and inertia.

Figure 15-34 shows a wrist sensor for a robot manipulator that provides feedback signals proportional to the X, Y, and Z forces acting on it. It consists of

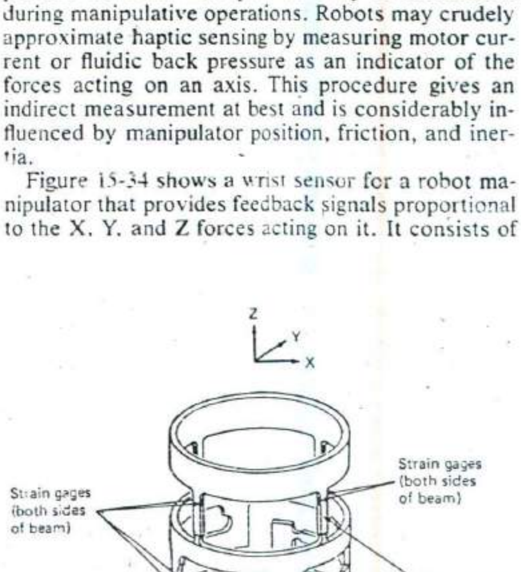

Z
Y
X
Strain gages (both sides of beam)
Strain gages (both sides of beam)
Elastic beams

Fig. 15-34 Wrist sensor.

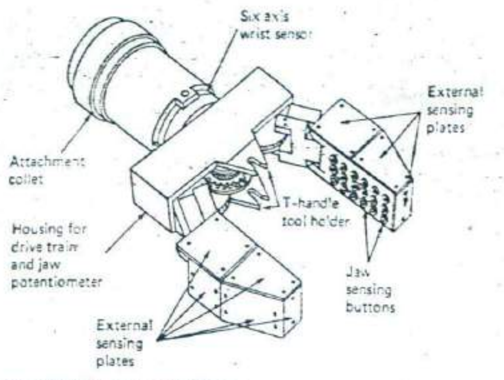


Fig. 15-35 Sensory end effector.

a metal tube with elastic beams. The beams flex in response to forces acting along the various axes. Strain gage sensors are placed on each side of the beam at the measurement points. Temperature compensation is achieved by using pairs of sensors. The wrist measures three components of force and torque between the manipulator and the end effector. The wrist sensor is illustrated in combination with a touch-sensitive gripper in Fig. 15-35. Each jaw face has a three by six matrix of touch buttons to sense the work piece. The outer jaw surfaces are also equipped with touch sensors to detect unexpected obstacles. The touch buttons provide on or off signals. They can supply only a limited amount of information; true tactile sensing requires more.

Various synthetic rubber and plastic materials are used to produce materials with elastic properties. These materials are called *elastomers* and can be used to make skin for a robot gripper. Conductive elastomers can provide feedback signals that are proportional to the forces acting on them. Another possibility is to use a matrix of touch cells, as shown in Fig. 15-36. The touch cells also produce signals that are proportional to force. The X and Y scan circuits connect the various cells to a multiplexer. The analog signal from the selected cell is sent to an analog-to-digital (A/D) converter. The processor stores the binary values from the various cells in a memory array. The contents of the array represent a *force contour* of the object, which is dictated by the shape of the object and its orientation in the gripper. The software then develops a "picture" of the object being held; this process is much the same as a human's grasping familiar objects in the dark and identifying them. It provides the robot with some ability to identify objects and determine their orientation by touching them. The current systems are crude by human performance standards, but improvements are expected.

Arrays of touch cells produce high-frequency signals when the gripper is moving. The information must be processed rapidly for the manipulator to respond in a reasonable fashion. Some touch cells are actually complete microcircuits containing an amplifier, an A/D converter, and a microprocessor. Another microprocessor receives parallel information from the individual microprocessors, rapidly assimilates it, and then sends its output to the main processor in the controller. It is interesting to note how difficult it is to simulate something humans do with such ease that they take it for granted. It is also

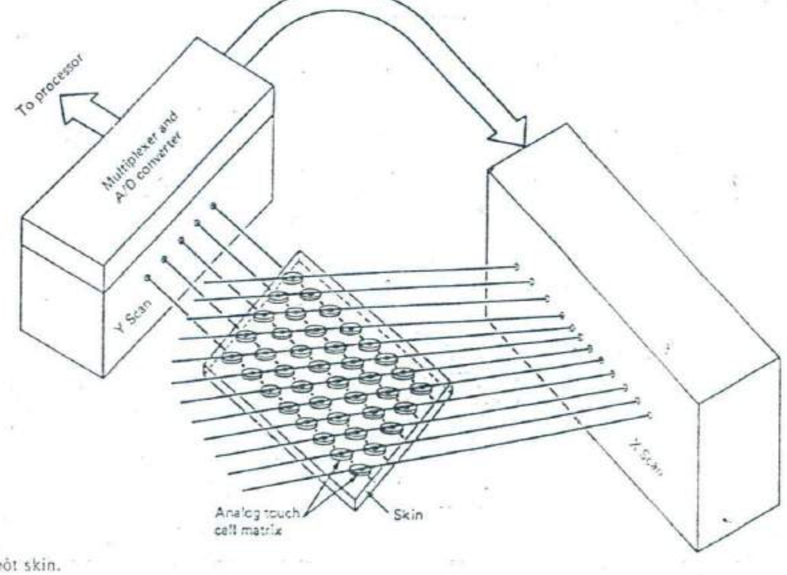


Fig. 15-36 Robot skin.

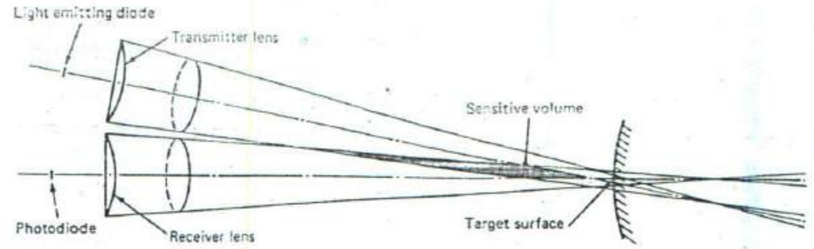


Fig. 15-37 Proximity sensing.

comforting to know that people are still the most amazing and sophisticated of all "machines" and will not become obsolete in the foreseeable future.

Arrays of conductive parallel strips may be used for tactile sensing. Scanning circuits reveal the contact points, and images can be created by the software. Piezoelectric polyvinyl fluoride materials provide signals that are proportional to changes in pressure. These materials offer promise for slip sensing, which is also very important in robotics. It is possible to design a single gripper that can pick up an egg without breaking it or a heavy metal machine part by sensing slip and applying only that amount of force required to eliminate it.

Proximity sensors are noncontact devices. They utilize magnetic fields, radio frequencies, ultrasound, or light for their operation. Some types were covered in Chapter 9. Figure 15-37 shows a simple system based on an infrared LED, a photodiode, and a lens system. Its sensitive volume is approximately the intersection of the two cones in space as shown. Such systems can be used for collision avoidance and motion detection within the work envelope. They work from centimeters to meters away. The intensity of the light received by the photodiode varies according to the distance to the target surface, its incidence angle, and its reflectivity. Therefore, this simple system cannot be used for *ranging* (distance measuring to the target) unless the angle of incidence and the reflectivity are constants. This is seldom the case. Figure 15-38 shows a more complicated laser scanning system that is capable of ranging as well as imaging. It utilizes an amplitude-modulated laser beam which is split into two paths. One path leads directly to a detector and the other to a two-axis scanning mirror. The mirror scans in two planes, and the laser beam is reflected toward the area of interest. The target surface reflects the beam back to the mirror and then on to the detector. The amplitude of the reflected beam provides surface information

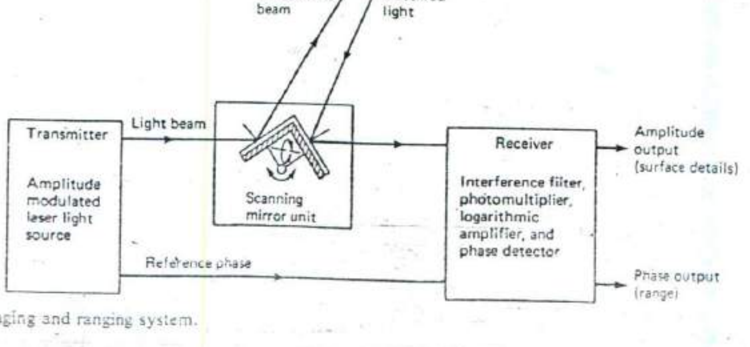

Target surface
Reflected light
Transmitted beam
Received light
Transmitter
Amplitude modulated laser light source
Light beam
Scanning mirror unit
Receiver
Interference filter, photomultiplier, logarithmic amplifier, and phase detector
Amplitude output (surface details)
Reference phase
Phase output (range)

Fig. 15-38 Laser imaging and ranging system.

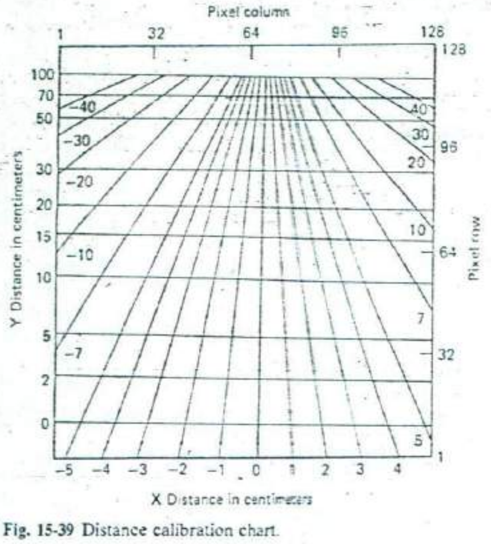


Fig. 15-39 Distance calibration chart.

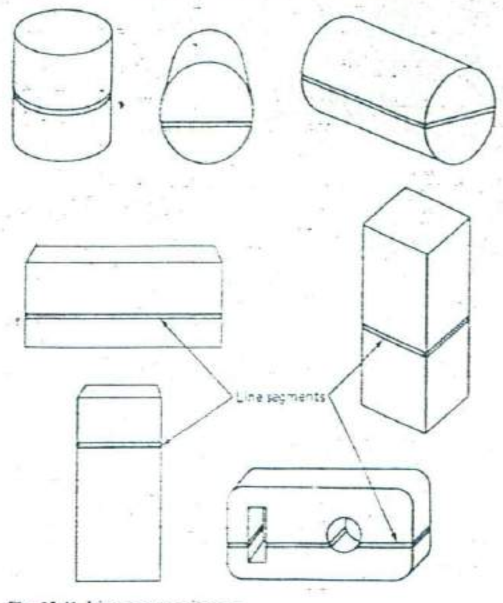


Fig. 15-41 Line segment images.

about the target. The phase of the reflected beam is compared to the reference phase to provide range information.

The U.S. National Bureau of Standards (NBS) has developed a robot vision system that uses triangulation to compute the distance to the target. This system is based on a 128 by 128 pixel image from a solid-state video camera and a stroboscopic source that emits a plane of light. The calibration chart for the NBS vision system is shown in Fig. 15-39. It is possible to compute the distance from the robot to each point in the image. When an object is in front of the robot, a line segment image is formed by the plane of light as shown in Fig. 15-40. The shape of the segment image indicates the orientation of the part. For example, look at the sample line segment images shown in Fig. 15-41. A cylinder can produce an image that appears as a curved segment, a straight line segment, or an angle segment, depending on the orientation of the cylinder. The rectangular solid can produce a long segment, a short segment, or an angle segment, depending on its orientation. Holes in an object may produce angles and broken segments as shown. By reducing images to line segments, the NBS system reduces memory requirements and speeds image processing. It emphasizes the acquisition of images that are easy to analyze. It has been used successfully to pick up parts from a random pile, identify them as to shape and size, and place them into a work holder. It has also allowed moving parts to be captured by the robot gripper.

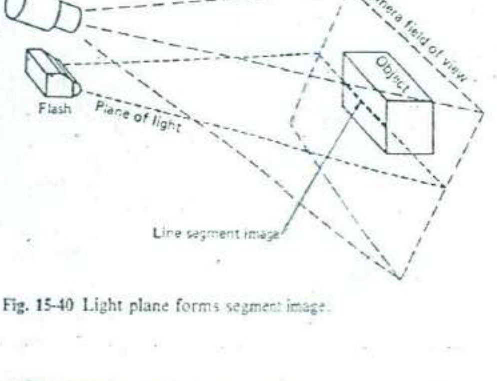


Fig. 15-40 Light plane forms segment image.

Not all robot vision systems use a plane of light. Other systems process the entire image, which may consist of 256 by 256 pixels. The image can be binary or Gray scale. A binary imaging system converts each point into black or white. A simple threshold circuit performs the conversion. Binary systems require less memory, and the images can be processed quickly. Gray scale systems provide more information, however. Some can even infer three-dimensional information by using nonuniform lighting and interpreting the reflective image. A binary image with 256 by 256 pixels requires 8K bytes of memory. The same image with 8 Gray bits per pixel requires 64K bytes. Some future systems will use 1024 by 1024 pixel images. 1M byte will be required to store one of these images in binary form; of course, even more memory will be required to store a Gray-scale image. Such systems are becoming more feasible as the price and space requirements of memory continue to drop and as processor speed continues to increase.

International Robotmation/Intelligence, the manufacturers of the M50 manipulator covered earlier in this chapter, offer a robot vision system that accepts

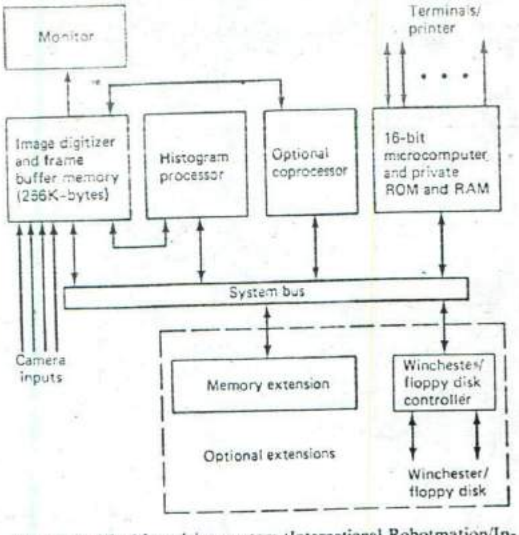


Fig. 15-42 Machine vision system (International Robotmation/Intelligence).

the input from up to four video cameras. The 256 by 256 pixel images are digitized in an 8-bit A/D converter and are stored in a 64K byte page of memory. The memory can store four images at one time for color processing or stereoscopic analysis. Figure 15-42 shows a block diagram of the M50 vision system. A 16-bit processor is used for image analysis and recognition. The system uses a feature vector to accomplish recognition. A *feature vector* consists of a set of measurements, or descriptors, that condense a Gray scale image into a small number of characteristic features. *Recognition* is the process in which the system compares the descriptors of an unknown object with those of model objects stored in memory. The models are developed during a training process by presenting objects to the system so the descriptors can be computed. The most useful descriptors are those that remain most constant regardless of position or orientation. Some of these are area, *elongation* (the ratio of the minimum radius to the maximum radius), number of holes, and length of vectors from the center of the object to points along its radius.

Descriptors are developed by analyzing the image for Gray scale gradients. A *gradient* is defined as a rate of change; for example, edges are recognized as sharp gradients in the Gray scale image. The edges define the basic geometry of the image so that measurements can be taken. Recognition follows with the *nearest neighbor algorithm*, which computes the straight line distance from the point representing the unknown object to each of the points representing the stored models. The model that most closely matches the unknown is selected. The software techniques are complex, and even with a fast 16-bit processor, processing and recognition require 0.1 s. For high-speed operations, an optional coprocessor can be added to speed this up to 0.01 s.

The block diagram of the vision system shown in Fig. 15-42 also shows a *histogram processor*. This section performs image enhancement operations at the same time the image is being digitized and can also process images that have been stored in the frame buffer. The enhancements improve contrast, subtract background information, window the image (select specific portions for further processing), and select various Gray scale thresholds to eliminate noise. The contrast is enhanced with histogram equalization. The histogram of an image provides the frequency of occurrence of each Gray level. Equalization expands the number of Gray levels near the maximum portion of the histogram and compresses the number of Gray levels near the minimum. This expands the brightest parts of the image and compresses the darkest for an apparent increase in contrast.

Machine vision systems are not confined to robotics. Figure 15-43 shows an arrangement for inspecting moving parts. Two diode arrays are used to develop *orthogonal views* (two right-angle images). The diodes are arranged to scan one line of the image at a time. Encoder signals are used to synchronize the scans with fixed increments of part movement. The video signals that result from each scan are sent to a computer for analysis in real time or are stored in memory until the entire part image is available for processing. Such a system may be used to identify parts, inspect them for defects, and verify their orientation on the conveyor. In some areas machine vision inspection is superior to human vision. Complex printed circuit boards are an example; it is unlikely that a person will detect a missing hole, but it is highly likely that this defect will be detected by the machine.

Sound sensors also find some application in automation. They can be used to detect abnormal situations. For example, a cutter bit makes a different sound when it is dull than when it is sharp. Digitized versions of the sharp and dull sounds can be stored in computer memory. The computer can then contin-

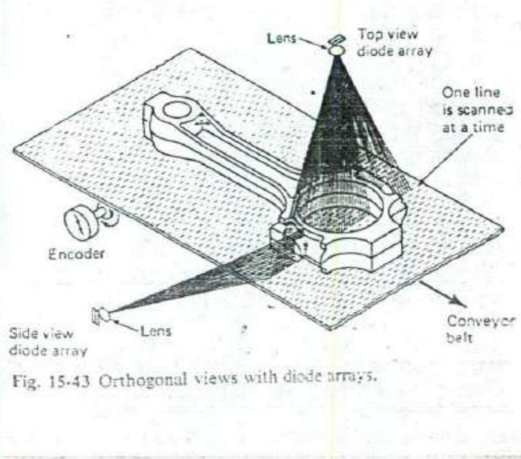


Fig. 15-43 Orthogonal views with diode arrays.

uously compare the output of a sound sensor with the memory standards and make a decision about when it is time to change tools.

REVIEW QUESTIONS

20. The two major parts of "intelligent automation" are the computer software and the ______ that provide information.

21. Touch sensing involves contact detection, and ______ sensing which measures continuous forces in an array.

22. Proximity devices are examples of non-______ sensors.

23. Refer to Fig. 15-34. Why are the strain gages placed on both sides of the beams?

24. Refer to Fig. 15-41. The segment images change in relation to part distance and part ______.

25. Machine vision systems may be divided into ______ or ______ ______ types.

26. Which type of vision system requires less memory?

15-6 TROUBLESHOOTING AND MAINTENANCE

Automation systems may be very easy to repair, but they can also be very difficult. The diagnostic methods used are as varied as the technicians who practice them. However, there are common practices that you must make a part of your routine. Safety is the most important aspect of your work. You must always be concerned about your own safety and that of others, and you must work to minimize equipment damage wherever possible. Please review the general safety practices in Chapter 1.

As you probably realize by now, knowing what a system is supposed to do and how it accomplishes what it does is over half the battle when troubleshooting. This is the "what and how" part. The remaining part deals with keen observation and a logical process of elimination. The symptoms must be filtered through the "what and how" framework to enable a skilled technician to focus attention on the correct area quickly. With experience, this technique will produce results in the least amount of time.

Do not forget the other rules of troubleshooting that have been discussed. Power supply malfunctions can create a myriad of symptoms and should always be investigated early in the troubleshooting process. Supply failures are common in programmable controllers and robotics. They are often caused by heat or line transients. Vents on equipment cabinets should not be obstructed. The cabinets themselves should not be used to store manuals, spare parts, or any items that could interfere with the flow of cooling air currents. You may have to educate other plant personnel. After all, to most people it seems innocent enough to place a blueprint on top of a power supply "temporarily." Repeated supply failures may indicate line transients. Many installations use line conditioners or uninterruptible power supplies to improve reliability. The I/O modules also fail with greater frequency than other parts of the automation system. They often receive transients of thousands of volts and are subject to damage from short circuits, bad connections, and wiring errors.

Controllers may have diagnostic indicators. Some of these are as follows:

1. Overvoltage
2. Undervoltage
3. Overcurrent
4. Overtemperature
5. Processor
6. Memory/parity error
7. I/O function

With modular designs, the indicators may suggest which module or modules should be changed. This indication can be misleading; for example, a memory error may be caused by another module that has access to the memory. Always power down when changing modules and follow the manufacturer's recommendations. Also, never change more than one module at a time. Power supply shutdown due to overcurrent may also require removal of the modules. If the overload persists, the chassis wiring may be defective, a connector may be short-circuited, or the supply itself may be defective. If the overload clears, the modules can be inserted one at a time until the problem shows up again. Again, consult the recommended procedures before using this technique and disable any high-energy circuit that can be activated during testing.

Timing problems in controllers can be difficult to diagnose. They may be caused by software faults, contact bounce, noise, or circuit failures. A common software fault in this area is the program end statement. Verify its format when timing problems occur. The scan time may be too long to provide adequate response to rapidly changing input conditions; this condition is known as *program racing*. Some controllers have diagnostic software to help locate difficulties. A contact histogram records the status of a bit in memory over a period of time. If the bit changes more times than its input device has toggled, then contact bounce or noise is indicated. The *exclusive* OR (XOR) *function* is another diagnostic tool that can be used to compare a number of inputs against their desired state:

```
10101001 ← ACTUAL BINARY INPUT
11101001 ← DESIRED BINARY STATE (MASK)
──────── ← XOR OPERATION
01000000 ← RESULT
```

Any differences between the inputs and the mask will produce a nonzero result. This technique is useful when the operating cycles are fixed and the input patterns are repeatable at each step. The XOR function is also useful for building a sequence table. The last entry in the table is continuously compared to the inputs. Each time a nonzero result is obtained, the new input status is added to the table. When the sample cycle has been recorded, it can be compared to the program cycle to determine whether they match.

When machines are being controlled, the cycle times may vary with factors such as friction. Delays may change the order in which input transitions occur during any given cycle. The diagnostic program may be able to construct timing windows to allow for these variations. The important test is to determine that each transition occurred within its own time window. The built-in diagnostic software may also provide "watchdog timers" to detect excessive scan times, machine cycle times, or motion times.

New installations or recently modified installations can present a special set of maintenance problems. Errors in wiring, poor connections, and short circuits are among the possibilities. The placement of wiring is also important because it can cause noise pick-up. Electrical noise is a prevalent problem in most industries. Wires are antennas; they radiate as well as receive signals and should be twisted in pairs to help cancel external fields and reduce this effect. Shielding may be required in some cases. Sensitive circuits, such as data communications lines, control lines, sensor lines, and feedback lines from resolvers, must be routed well away from motor supplies and other high-energy circuits. Some installers, in the interest of neatness, tie-wrap all possible wires together into a bundle. Although this bundle may be pleasing to the eye, it is not acceptable because of noise.

Ground loops are another contributor to noise and faulty operation. Verify that the manufacturer's grounding and shielding recommendations have been followed. Use an oscilloscope to check low-level signals for the presence of noise. Make sure that the oscilloscope is properly grounded to the signal circuit. Use safe floating measurement techniques as discussed in Chapter 1 when necessary. Ground wires may also have to be rerouted in some cases. Poor grounds or missing grounds can even cause oscillations in a robot manipulator.

The placement of equipment is also important. National Electrical Manufacturers Association (NEMA) enclosures are most often used to house controls. Some motor drives use large inductors. It may be tempting to house these inductors in the same cabinet with the control electronics. This is not recommended practice, but if it must be done, the physical separation between the inductors and the control electronics must be maximized.

When data communications circuits will not work, check the wiring. Pins 2 and 3 on RS-232C connectors tend to become reversed. Make sure that the grounds are intact and that all necessary signals and jumpers are provided. Verify that the baud rate, parity, and the number of stop bits are correctly set up at both ends of the circuit. Input-output modules are often wired incorrectly. Make sure that high-voltage circuits have not been mixed with low-voltage control circuits.

Servomechanisms should be checked with no load, if possible. If normal operation cannot be achieved under this condition, the defect is not an overload. The gain should be adjusted for good stiffness and no oscillations. The gain may be controlled by a hardware adjustment or by software, depending on the controller. Next, measure the drive current under no-load conditions. It should be far less than maximum. Or measure the drive voltage at the input of the power amplifier. It may range from -10 to $+10$ V. It is safe to assume that maximum current occurs when the drive voltage is at one end or the other of its range. Again, the drive voltage should be less than maximum under no-load conditions. If everything appears normal, the servomechanism can be reconnected to its mechanical load. In the case of a robot manipulator, a typical payload can be added to the gripper. Readjust the gain for the proper characteristics and measure current or drive voltage again. Try different operating speeds. Higher drive voltages and currents can be expected at the faster speeds.

A system will not work properly if it is overloaded. Overloads may be due to pinching or binding in a mechanism. This may show up at only one point of travel so exercise the entire range while monitoring drive. Oscillations may be caused by abnormal backlash. Check the ball screws, harmonic drives, rack and pinions, chain drives, and other parts of the system. Hammering or banging may indicate a defective shock absorber.

A fluidic mechanism can be checked by using an oscilloscope to monitor control signals to the valves. For example, if a pneumatic system uses dithering to control the air motors, the duty cycle of the control waveform will be an indication of how hard the servomechanism is working at various speed and load conditions. Once again, you must follow the manufacturer's recommended procedures. A technique that works on one machine may be misleading or perhaps dangerous on another.

REVIEW QUESTIONS

27. The prime consideration in troubleshooting is ______.

28. When modules are being removed or replaced, the power must be ______.

29. A contact histogram is a software diagnostic that can be used to detect contact ______.

30. When a mask is exclusively ORed with input signals, any discrepancies will produce a ______ result.

31. A built-in diagnostic to detect excessive scan times or cycle times is called a ______ timer.

32. It is possible to cancel external fields around conductors by ______ the wires into pairs.

CHAPTER REVIEW QUESTIONS

15-1. Another name for degree of freedom in robotics is ______.

15-2. Most industrial tasks can be accomplished with ______ or fewer degrees of freedom.

15-3. Robots that use no feedback may be called low-technology, pick-and-place, limited sequence, and other names. However, they are distinctly in the ______ category.

15-4. "Bang-bang" robots are set up for a task by adjusting ______ ______.

15-5. The three terms used to describe the wrist axes of a robot are ______, ______, and ______.

15-6. Medium- and high-technology robots use feedback and are therefore considered as ______ types.

15-7. Point-to-point servos require less memory than ______ ______ types.

15-8. Which robot actuator is best suited for fast operations with light payloads?

15-9. Which actuator is best suited to handling heavy loads?

15-10. A manipulator is another name for a robot ______.

15-11. End effectors are robot ______ or ______.

15-12. Rack and pinion assemblies are used to convert rotary motion to ______ motion. What other mechanism does the same thing with much less backlash?

15-13. The input energy to a harmonic drive transmission is applied to an elliptical ______ ______.

15-14. If the rigid spline of a harmonic drive has 400 teeth, and the flexspline has 402 teeth, calculate the speed reduction ratio.

15-15. A harmonic drive will exhibit less ______ and less ______ when compared to a gearbox with the same reduction ratio.

15-16. Shock absorbers convert mechanical energy into ______ energy.

15-17. A robot will take an unexpected path when executing a program if the operator has neglected to ______ a point in memory.

15-18. Using a computer to calculate path moves eliminates some of the problems associated with point-to-point and continuous path systems. This technique is based on the computational ability of the computer and is known as ______ path programming.

15-19. Some robot controllers allow the data and programs to be modified at a CRT terminal. This feature is called ______.

15-20. What software capability will most likely be used to stop a robot when a sensor detects an obstruction (such as a person) in the work envelope?

15-21. A robot is used to drill holes. It has sensors to detect a defective drill bit. It also has the ability to change the bit. What software feature would probably be used to control the bit change?

15-22. Which type of vision system requires the least processing time?

15-23. Which type of vision system provides the most information?

15-24. Object edges in a Gray scale picture can be found by looking for sharp ______ in the image.

15-25. Histogram equalization can be used to enhance the apparent ______ of a Gray scale picture.

15-26. Sensitive circuits must be routed well ______ ______ high-energy circuits.

15-27. Erratic operation at one point of travel in a manipulator may indicate ______ in some mechanism.

15-28. Oscillations in a manipulator may be due to ______ caused by a misadjusted mechanical system.

ANSWERS TO REVIEW QUESTIONS

1. software **2.** CPU; I/O **3.** ladder **4.** scanning **5.** battery **6.** noise (transients) **7.** work envelope **8.** hydraulics; pneumatics **9.** to the right **10.** servo; it uses feedback **11.** dithering **12.** teeth **13.** speed **14.** by adjusting the cams **15.** by entering a new program (or changing the existing one) **16.** 2; 3; the *Z* axis has an intermediate stop cylinder **17.** servo-controlled **18.** lead-through **19.** walk-through **20.** sensors **21.** tactile **22.** contact **23.** to provide temperature compensation **24.** orientation **25.** binary; Gray scale **26.** binary **27.** safety **28.** off **29.** bounce **30.** nonzero **31.** watchdog **32.** twisting