

Laboratory Experiments

11-0 INTRODUCTION TO EXPERIMENTS

This chapter presents 18 laboratory experiments in digital circuits and logic design. They provide hands-on experience for the student using this book. The digital circuits can be constructed by using standard integrated circuits (ICs) mounted on breadboards that are easily assembled in the laboratory. The experiments are ordered according to the material presented in the book.

A logic breadboard suitable for performing the experiments must have the following equipment:

1. LED (light-emitting diode) indicator lamps.
2. Toggle switches to provide logic-1 and -0 signals.
3. Pulsers with pushbuttons and debounce circuits to generate single pulses.
4. A clock-pulse generator with at least two frequencies, a low frequency of about one pulse per second to observe slow changes in digital signals and a higher frequency of about 10 kHz or higher for observing waveforms in an oscilloscope.
5. A power supply of 5 V for TTL ICs.
6. Socket strips for mounting the ICs.
7. Solid hookup wire and a pair of wire strippers for cutting the wires.

Digital logic trainers that include the required equipment are available from several manufacturers. A digital logic trainer contains LED lamps, toggle switches, pulsers, a variable clock, power supply, and IC socket strips. Some experiments may require ad-

ditional switches, lamps, or IC socket strips. Extended breadboards with more solderless sockets and plug-in switches and lamps may be needed.

Additional equipment required are a dual-trace oscilloscope (for Experiments 1, 2, 8, and 15), a logic probe to be used for debugging, and a number of ICs. The ICs required for the experiments are of the transistor-transistor logic (TTL) type, series 7400.

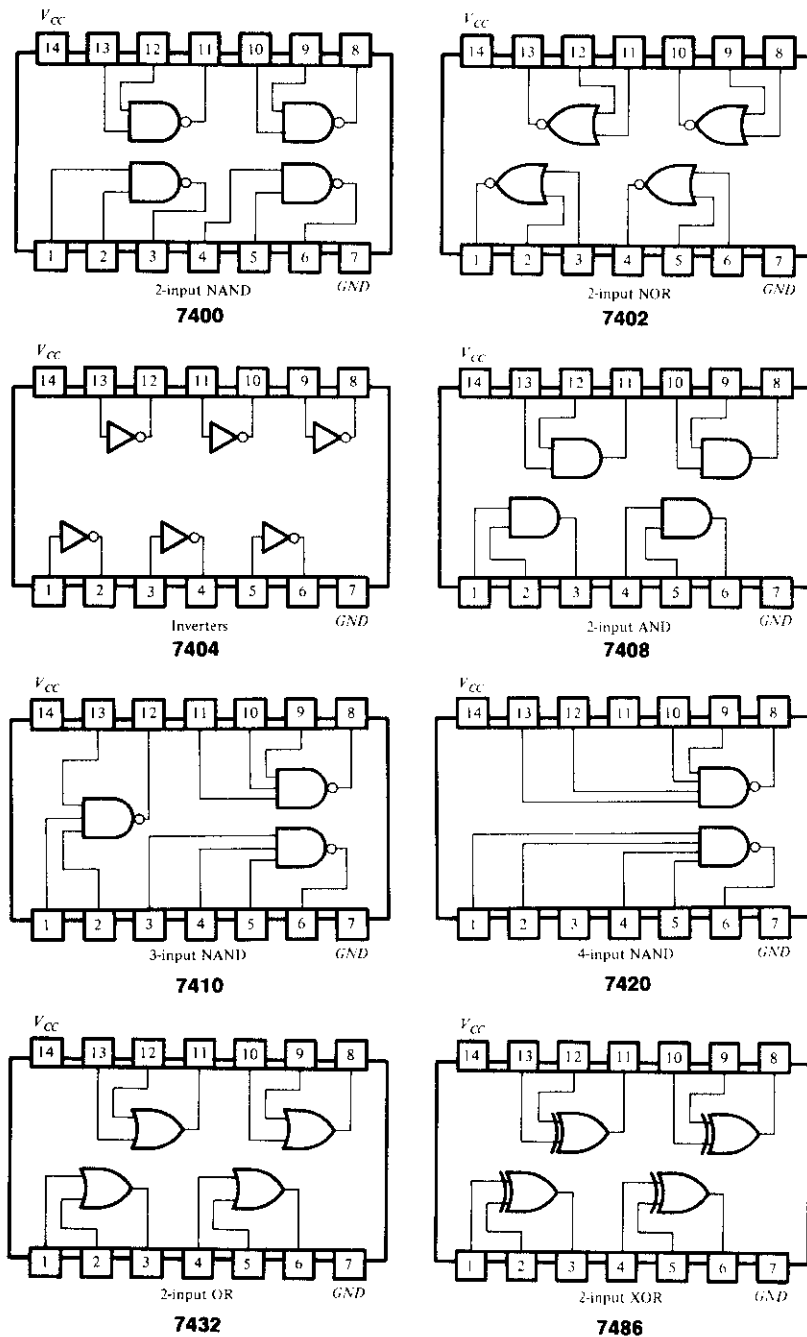
The integrated circuits to be used in the experiments can be classified as small-scale integration (SSI) or medium-scale integration (MSI) circuits. SSI circuits contain individual gates or flip-flops, and MSI circuits perform specific digital functions. The eight SSI gate ICs needed for the experiments are shown in Fig. 11-1. They include 2-input NAND, NOR, AND, OR, and XOR gates, inverters, and 3-input and 4-input NAND gates. The pin assignment for the gates is indicated in the diagram. The pins are numbered from 1 to 14. Pin number 14 is marked V_{CC} , and pin number 7 is marked GND (ground). These are the supply terminals, which must be connected to a power supply of 5 V for proper operation. Each IC is recognized by its identification number; for example, the 2-input NAND gates are found inside the IC whose number is 7400.

Detailed descriptions of the MSI circuits can be found in data books published by the manufacturers. The best way to acquire experience with commercial MSI circuits is to study their description in a data book that provides complete information on the internal, external, and electrical characteristics of the integrated circuits. Various semiconductor companies publish data books for the TTL 7400 series. Examples are *The TTL Data Book*, published by Texas Instruments, and the *Logic Databook*, published by National Semiconductor Corp.

The MSI circuits that are needed for the experiments are introduced and explained when they are used for the first time. The operation of the circuit is explained by referring to similar circuits in previous chapters. The information given in this chapter about the MSI circuits should be sufficient for performing the experiments adequately. Nevertheless, a reference to a data book will always be preferable, as it gives more detailed description of the circuits.

We will now demonstrate the method of presentation of MSI circuits adopted here. This will be done by means of a specific example that introduces the ripple counter IC, type 7493. This IC is used in Experiment 1 and in subsequent experiments to generate a sequence of binary numbers for verifying the operation of combinational circuits.

The information about the 7493 IC that is found in a data book is shown in Figs. 11-2(a) and (b). Part (a) shows a diagram of the internal logic circuit and its connection to external pins. All inputs and outputs are given symbolic letters and assigned to pin numbers. Part (b) shows the physical layout of the IC with its 14-pin assignment to signal names. Some of the pins are not used by the circuit and are marked as *NC* (no connection). The IC is inserted into a socket, and wires are connected to the various pins through the socket terminals. When drawing schematic diagrams in this chapter, we will show the IC in a block diagram form as in Fig. 11-2(c). The IC number 7493 is written inside the block. All input terminals are placed on the left of the block and all output terminals on the right. The letter symbols of the signals, such as *A*, *R1*, and *QA*, are written inside the block, and the corresponding pin numbers, such as 14, 2, and 12, are written along the external lines. V_{CC} and GND are the power terminals connected to

**FIGURE 11-1**

Digital gates in IC packages with identification numbers and pin assignments

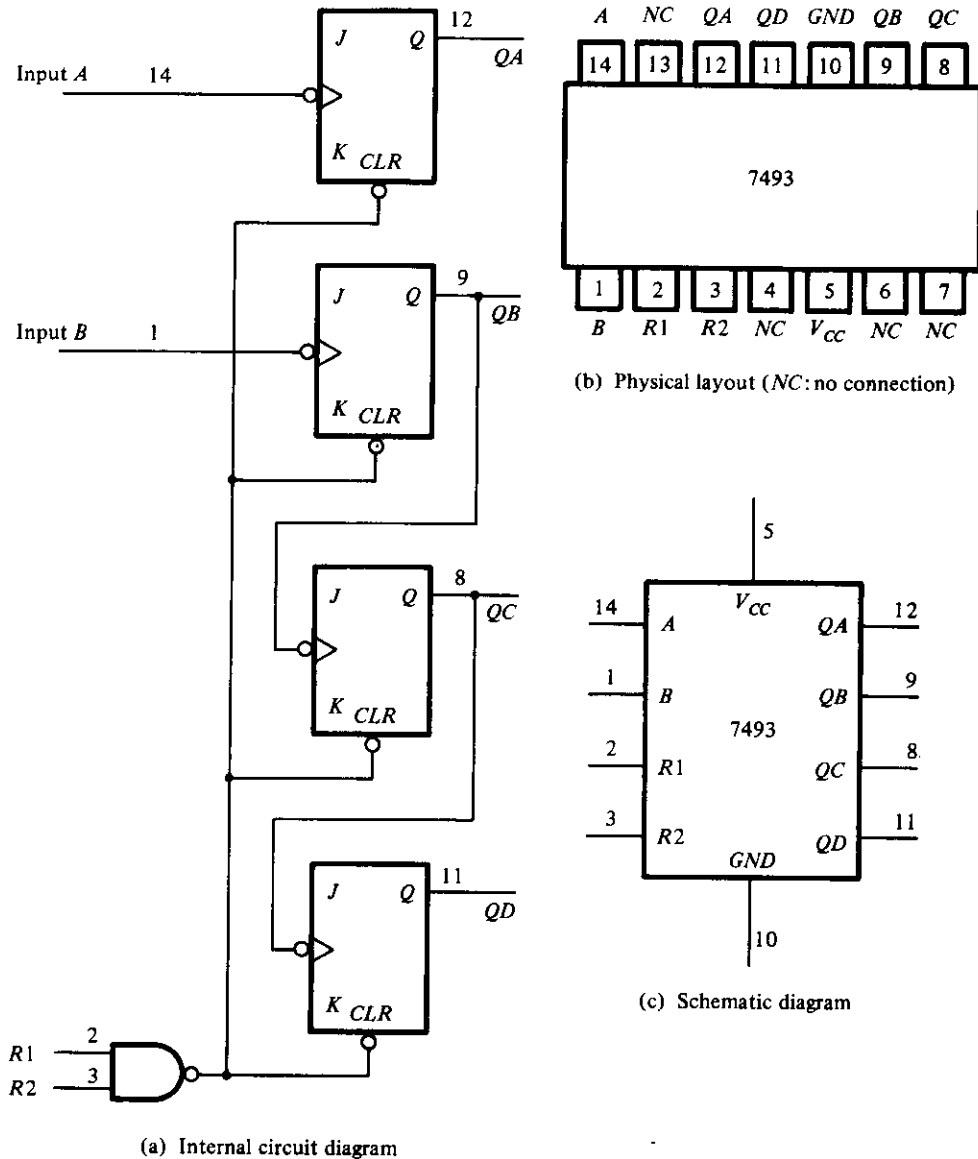


FIGURE 11-2
IC type 7493 ripple counter

pins 5 and 10. The size of the block may vary to accommodate all input and output terminals. Inputs or outputs may sometimes be placed on the top or the bottom of the block for convenience.

The operation of the circuit is similar to the ripple counter shown in Fig. 7-12 with an asynchronous clear to each flip-flop, as shown in Fig. 6-15. When inputs R1 or R2

or both are equal to logic 0 (ground for TTL circuits), all asynchronous clears are equal to 1 and are disabled. To clear all four flip-flops to 0, the output of the NAND gate must be equal to 0. This is accomplished by having both inputs $R1$ and $R2$ at logic-1 (about 3 to 5 V in TTL circuits). Note that the J and K inputs show no connections. It is characteristic of TTL circuits that an input terminal with no external connections has the effect of producing a signal equivalent to logic-1. Also note that output QA is not connected to input B internally.

The 7493 IC can operate as a 3-bit counter using input B and flip-flops QB , QC , and QD . It can operate as a 4-bit counter using input A if output QA is connected to input B . Therefore, to operate the circuit as a 4-bit counter, it is necessary to have an external connection between pin 12 and pin 1. The reset inputs, $R1$ and $R2$, at pins 2 and 3, respectively, must be grounded. Pins 5 and 10 must be connected to a 5-V power supply. The input pulses must be applied to input A at pin 14, and the four flip-flop outputs of the counter are taken from QA , QB , QC , and QD , at pins 12, 9, 8, and 11, respectively, with QA being the least significant bit.

Figure 11-2(c) demonstrates the way that all MSI circuits will be symbolized graphically in this chapter. Only a block diagram similar to the one shown in this figure will be shown for each IC. The letter symbols for the inputs and outputs in the IC block diagram will be according to the symbols used in the data book. The operation of the circuit will be explained with reference to logic diagrams from previous chapters. The operation of the circuit will be specified by means of a truth table or a function table.

Other possible graphic symbols for the ICs are presented in Chapter 12. These are

TABLE 11-1
Integrated Circuits Required for the Experiments

IC Number	Description	Graphic Symbol	
		In Chap. 11	In Chap. 12
	Various gates	Fig. 11-1	Fig. 12-1
7447	BCD-to-seven-segment decoder	Fig. 11-8	
7474	Dual D -type flip-flops	Fig. 11-13	Fig. 12-9(b)
7476	Dual JK -type flip-flops	Fig. 11-12	Fig. 12-9(a)
7483	4-bit binary adder	Fig. 11-10	Fig. 12-2
7489	16×4 random-access memory	Fig. 11-18	Fig. 12-15
7493	4-bit ripple counter	Fig. 11-2	Fig. 12-13
74151	8×1 multiplexer	Fig. 11-9	Fig. 12-7(a)
74155	3×8 decoder	Fig. 11-7	Fig. 12-6
74157	Quadruple 2×1 multiplexers	Fig. 11-17	Fig. 12-7(b)
74161	4-bit synchronous counter	Fig. 11-15	Fig. 12-14
74194	Bidirectional shift register	Fig. 11-19	Fig. 12-12
74195	4-bit shift register	Fig. 11-16	Fig. 12-11
7730	Seven-segment LED display	Fig. 11-8	
72555	Timer (same as 555)	Fig. 11-21	

standard graphic symbols approved by the Institute of Electrical and Electronics Engineers and are given in IEEE standard 91-1984. The standard graphic symbols for SSI gates have rectangular shapes, as shown in Fig. 12-1. The standard graphic symbol for the 7493 IC is shown in Fig. 12-13. This symbol can be substituted in place of the one shown in Fig. 11-2(c). The standard graphic symbols of the other ICs that are needed to run the experiments are presented in Chapter 12. They can be used for drawing schematic diagrams of the logic circuits if the standard symbols are preferred.

Table 11-1 lists the ICs that are needed for the experiments together with the figure numbers where they are presented in this chapter. In addition, the table lists the figure numbers in Chapter 12 where the equivalent standard graphic symbols are drawn.

The rest of this chapter is divided into 18 sections, with each section covering one experiment. The section number designates the experiment number. Each experiment should take about two to three hours of laboratory time, except for Experiments 14, 16, and 17, which may take longer.

11-1 BINARY AND DECIMAL NUMBERS

This experiment demonstrates the count sequence of binary numbers and the binary-coded decimal (BCD) representation. It serves as an introduction to the breadboard used in the laboratory and acquaints the student with the cathode-ray oscilloscope. Reference material from the text that may be useful to know while performing the experiment can be found in Section 1-2, on binary numbers, and Section 1-7, on BCD numbers.

Binary Count. IC type 7493 consists of four cells called flip-flops, as shown in Fig. 11-2. The cells can be connected to count in binary or in BCD. Connect the IC to operate as a 4-bit binary counter by wiring the external terminals, as shown in Fig. 11-3. This is done by connecting a wire from pin 12 (output *QA*) to pin 1 (input *B*). Input *A* at pin 14 is connected to a pulser that provides single pulses. The two reset inputs, *R1* and *R2*, are connected to ground. The four outputs go to four indicator lamps with the low-order bit of the counter from *QA* connected to the rightmost indicator lamp. Do not forget to supply 5 V and ground to the IC. All connections should be made with the power supply in the off position.

Turn the power on and observe the four indicator lamps. The 4-bit number in the output is incremented by one for every pulse generated in the push-button pulser. The count goes to binary 15 and then back to 0. Disconnect the input of the counter at pin 14 from the pulser and connect it to a clock generator that produces a train of pulses at a low frequency of about one pulse per second. This will provide an automatic binary count. Note that the binary counter will be used in subsequent experiments to provide the input binary signals for testing combinational circuits.

Oscilloscope Display. Increase the frequency of the clock to 10 kHz or higher and connect its output to an oscilloscope. Observe the clock output on the oscilloscope and

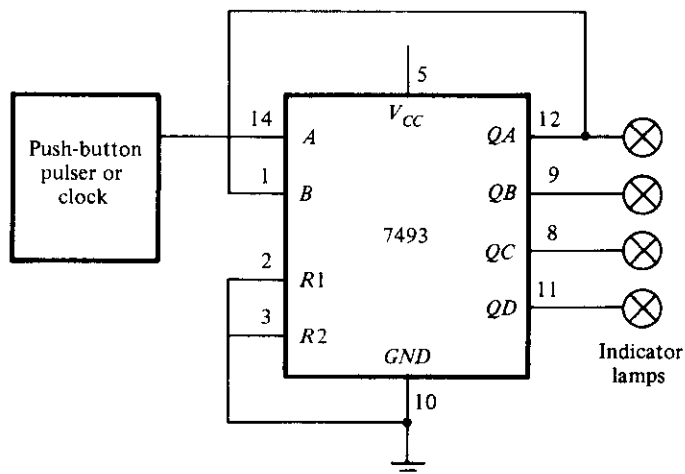
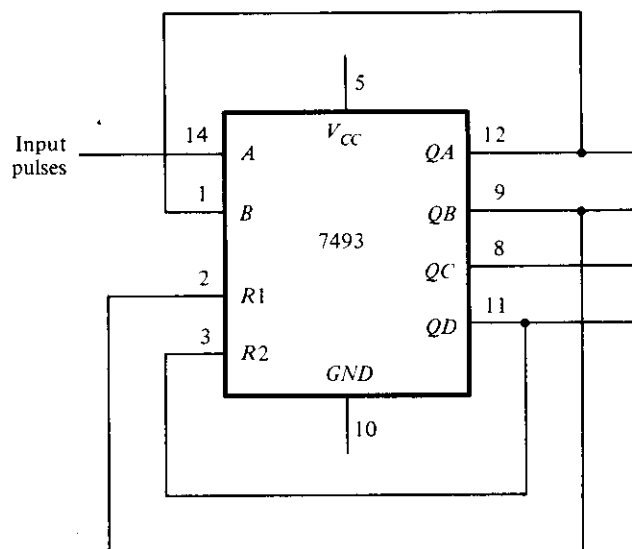


FIGURE 11-3
Binary counter

sketch its waveform. If a dual-trace oscilloscope is available, connect the output of QA to one channel and the output of the clock to the second channel. Note that the output of QA is complemented every time the clock pulse goes through a negative transition from 1 to 0. (The two waveforms should look similar to the timing diagram shown in Fig. 7-15 for the count pulses and Q_1 .)

When the count pulses into the counter occur at constant frequency, the frequency at the output of the first cell, QA , is one-half that of the input clock frequency. Each cell in turn divides its incoming frequency by 2. The 4-bit counter divides the incoming frequency by 16 at output QD . Obtain a timing diagram showing the time relationship of the clock and the four outputs of the counter. Make sure that you include at least 16 clock pulses. The way to proceed with a dual-trace oscilloscope is as follows. First, observe the clock pulses and QA and record their timing waveforms. Then repeat by observing and recording the waveforms of QA together with QB , followed by the waveforms of QB with QC and then QC with QD . Your final result should be a diagram showing the time relationship of the clock and the four outputs in one composite diagram having at least 16 clock pulses.

BCD Count. The BCD representation uses the binary numbers from 0000 to 1001 to represent the coded decimal digits from 0 to 9. IC type 7493 can be operated as a BCD counter by making the external connections shown in Fig. 11-4. Outputs QB and QD are connected to the two reset inputs, $R1$ and $R2$. When both $R1$ and $R2$ are equal to 1, all four cells in the counter clear to 0 irrespective of the input pulse. The counter starts from 0, and every input pulse increments it by 1 until it reaches the count of 1001. The next pulse changes the output to 1010, making QB and QD equal to 1. This momentary output cannot be sustained, because the four cells immediately clear to 0,

**FIGURE 11-4**

BCD counter

with the result that the output goes to 0000. Thus, the pulse after the count of 1001 changes the output to 0000, producing a BCD count.

Connect the IC to operate as a BCD counter. Connect the input to a pulser and the four outputs to indicator lamps. Verify that the count goes from 0000 to 1001.

Disconnect the input from the pulser and connect it to a clock generator with a frequency of 10 kHz or higher. Observe the clock waveform and the four outputs on the oscilloscope. Obtain an accurate timing diagram showing the time relationship between the clock and the four outputs. Make sure to include at least ten clock pulses in the oscilloscope display and in the composite timing diagram.

Output Pattern. When the count pulses into the BCD counter are continuous, the counter keeps repeating the sequence from 0000 to 1001 and back to 0000 over and over. This means that each bit in the four outputs produces a fixed pattern of 1's and 0's, which is repeated every ten pulses. These patterns can be predicted from the list of the binary numbers from 0000 to 1001. The list will show that output *QA*, being the least significant bit, produces a pattern of alternate 1's and 0's. Output *QD*, being the most significant bit, produces a pattern of eight 0's followed by two 1's. Obtain the pattern for the other two outputs and then check all four patterns on the oscilloscope. This is done with a dual-trace oscilloscope by displaying the clock pulses in one channel and one of the output waveforms in the other channel. The pattern of 1's and 0's for the corresponding output is obtained by observing the output levels at the vertical positions where the pulses change from 1 to 0.

Other Counts. IC type 7493 can be connected to count from 0 to a variety of final counts. This is done by connecting one or two outputs to the reset inputs, $R1$ and $R2$. Thus, if $R1$ is connected to QA instead of QB in Fig. 11-4, the resulting count will be from 0000 to 1000, which is 1 less than 1001 ($QD = 1$ and $QA = 1$).

Utilizing your knowledge of how $R1$ and $R2$ affect the final count, connect the 7493 IC to count from 0000 to the following final counts.

- (a) 0101
- (b) 0111
- (c) 1011

Connect each circuit and verify its count sequence by applying pulses from the pulser and observing the output count in the indicator lamps. If the initial count starts with a value greater than the final count, keep applying input pulses until the output clears to 0.

11-2 DIGITAL LOGIC GATES

In this experiment, you will investigate the logic behavior of various IC gates:

- 7400 Quadruple 2-input NAND gates
- 7402 Quadruple 2-input NOR gates
- 7404 Hex inverters
- 7408 Quadruple 2-input AND gates
- 7432 Quadruple 2-input OR gates
- 7486 Quadruple 2-input XOR gates

The pin assignments to the various gates are shown in Fig. 11-1. “Quadruple” means that there are four gates within the package. The TTL digital logic gates and their characteristics are discussed in Section 2-8. NAND implementation is discussed in Sections 3-6 and 4-7.

Truth Tables. Use one gate from each IC listed above and obtain the truth table of the gate. The truth table is obtained by connecting the inputs of the gate to switches and the output to an indicator lamp. Compare your results with the truth tables listed in Fig. 2-5.

Waveforms. For each gate listed above, obtain the input-output waveform relationship of the gate. The waveforms are to be observed in the oscilloscope. Use the two low-order outputs of a binary counter (Fig. 11-3) to provide the inputs to the gate. As an example, the circuit and waveforms for the NAND gate are illustrated in Fig. 11-5. The oscilloscope display will repeat this waveform, but you should record only the non-repetitive portion.

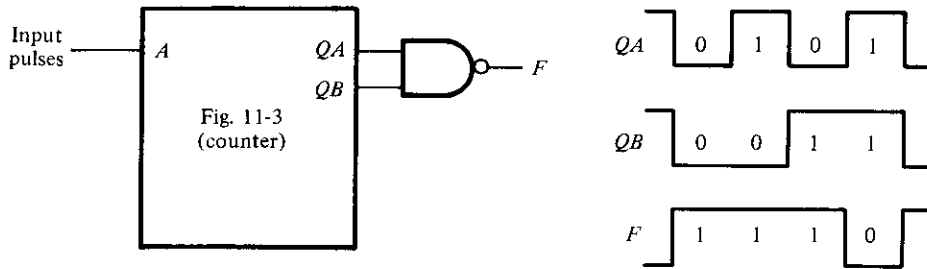


FIGURE 11-5
Waveforms for NAND gate

Propagation Delay. Connect all six inverters inside the 7404 IC in cascade. The output will be the same as the input except that it will be delayed by the time it takes the signal to propagate through all six inverters. Apply clock pulses to the input of the first inverter. Using the oscilloscope, determine the delay from the input to the output of the sixth inverter during the upswing and again during the downswing of the pulse. This is done with a dual-trace oscilloscope by applying the input clock pulses to one of the channels and the output of the sixth inverter to the second channel. Set the time-base knob to the lowest time-per-division setting. The rise or fall time of the two pulses should appear on the screen. Divide the total delay by 6 to obtain an average propagation delay per inverter.

Universal NAND Gate. Using a single 7400 IC, connect a circuit that produces:

- (a) an inverter
- (b) a 2-input AND
- (c) a 2-input OR
- (d) a 2-input NOR
- (e) a 2-input XOR (see Fig. 4-21)

In each case, verify your circuit by checking its truth table.

NAND Circuit. Using a single 7400 IC, construct a circuit with NAND gates that implements the Boolean function

$$F = AB + CD$$

1. Draw the circuit diagram.
2. Obtain the truth table for F as a function of the four inputs.
3. Connect the circuit and verify the truth table.
4. Record the patterns of 1's and 0's for F as inputs A , B , C , and D go from binary 0 to binary 15.
5. Connect the four outputs of the binary counter shown in Fig. 11-3 to the four in-

puts of the NAND circuit. Connect the input clock pulses from the counter to one channel and output F to the other channel of a dual-trace oscilloscope. Observe and record the 1's and 0's pattern of F after each clock pulse and compare it to the pattern recorded in Step 4.

11-3 SIMPLIFICATION OF BOOLEAN FUNCTIONS

This experiment demonstrates the relationship between a Boolean function and the corresponding logic diagram. The Boolean functions are simplified by using the map method, as discussed in Chapter 3. The logic diagrams are to be drawn using NAND gates, as explained in Section 3-6.

The gate ICs to be used for the logic diagrams must be those from Fig. 11-1 that contain NAND gates:

7400 2-input NAND

7404 Inverter (1-input NAND)

7410 3-input NAND

7420 4-input NAND

If an input to a NAND gate is not used, it should not be left open, but, instead, should be connected to another input that is used. For example, if the circuit needs an inverter and there is an extra 2-input gate available in a 7400 IC, then both inputs of the gate are to be connected together to form a single input for an inverter.

Logic Diagram. This part of the experiment starts with a given logic diagram from which we proceed to apply simplification procedures to reduce the number of gates and possibly the number of ICs. The logic diagram shown in Fig. 11-6 requires two ICs, a 7400 and a 7410. Note that the inverters for inputs x , y , and z are obtained from the remaining three gates in the 7400 IC. If the inverters were taken from a 7404 IC, the circuit would have required three ICs. Also note that in drawing SSI circuits, the gates are not enclosed in blocks as is done with MSI circuits.

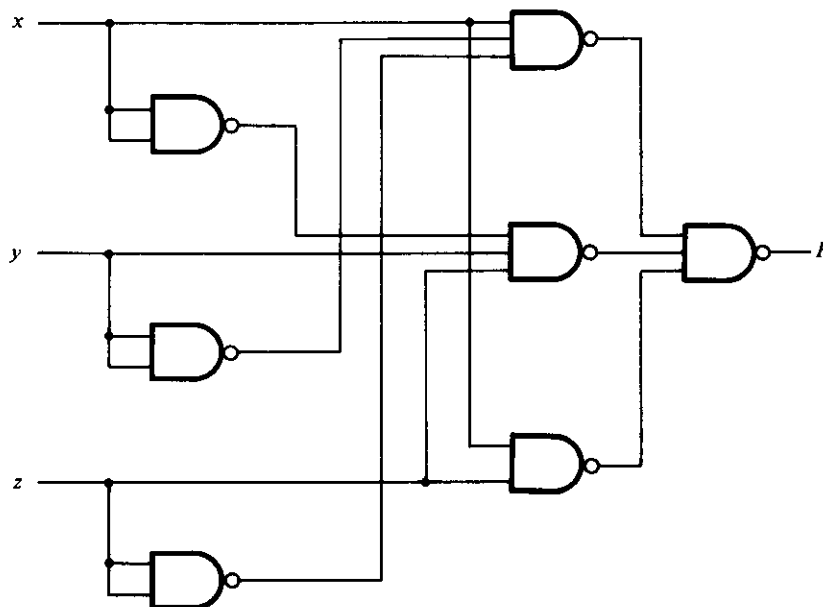
Assign pin numbers to all inputs and outputs of the gates and connect the circuit with the x , y , and z inputs going to three switches and the output F to an indicator lamp. Test the circuit by obtaining its truth table.

Obtain the Boolean function of the circuit and simplify it using the map method. Construct the simplified circuit without disconnecting the original circuit. Test both circuits by applying identical inputs to both and observing the separate outputs. Show that for each of the eight possible input combinations, the two circuits have identical outputs. This will prove that the simplified circuit behaves exactly as the original circuit.

Boolean Functions. Given the two Boolean functions in sum of minterms:

$$F_1(A, B, C, D) = (0, 1, 4, 5, 8, 9, 10, 12, 13)$$

$$F_2(A, B, C, D) = (3, 5, 7, 8, 10, 11, 13, 15)$$

**FIGURE 11-6**

Logic diagram for Experiment 3

Simplify the two functions by means of maps. Obtain a composite logic diagram with four inputs, A , B , C , and D , and two outputs, F_1 and F_2 . Implement the two functions together using a minimum number of NAND ICs. Do not duplicate the same gate if the corresponding term is needed for both functions. Use any extra gates in existing ICs for inverters when possible. Connect the circuit and check its operation. The truth table for F_1 and F_2 obtained from the circuit should conform with the minterms listed.

Complement. Plot the following Boolean function in a map:

$$F = A'D + BD + B'C + AB'D$$

Combine the 1's in the map to obtain the simplified function for F in sum of products. Then combine the 0's in the map to obtain the simplified function for F' also in sum of products. Implement both F and F' using NAND gates and connect the two circuits to the same input switches, but to separate output indicator lamps. Obtain the truth table of each circuit in the laboratory and show that they are the complements of each other.

11-4 COMBINATIONAL CIRCUITS

In this experiment, you will design, construct, and test four combinational logic circuits. The first two circuits are to be constructed with NAND gates, the third with XOR gates, and the fourth with a decoder and NAND gates. Reference to a parity gen-

erator can be found in Section 4-9. Implementation with a decoder is discussed in Section 5-5.

Design Example. Design a combinational circuit with four inputs, A , B , C , and D , and one output, F . F is to be equal to 1 when $A = 1$ provided that $B = 0$, or when $B = 1$ provided that either C or D is also equal to 1. Otherwise, the output is to be equal to 0.

1. Obtain the truth table of the circuit.
2. Simplify the output function.
3. Draw the logic diagram of the circuit using NAND gates with a minimum number of ICs.
4. Construct the circuit and test it for proper operation by verifying the conditions stated above.

Majority Logic. A majority logic is a digital circuit whose output is equal to 1 if the majority of the inputs are 1's. The output is 0 otherwise. Design and test a 3-input majority circuit using NAND gates with a minimum number of ICs.

Parity Generator. Design, construct, and test a circuit that generates an even parity bit from four message bits. Use XOR gates. Adding one more XOR gate, expand the circuit so it generates an odd parity bit also.

Decoder Implementation. A combinational circuit has three inputs, x , y , and z , and three outputs, F_1 , F_2 , and F_3 . The simplified Boolean functions for the circuit are as follows:

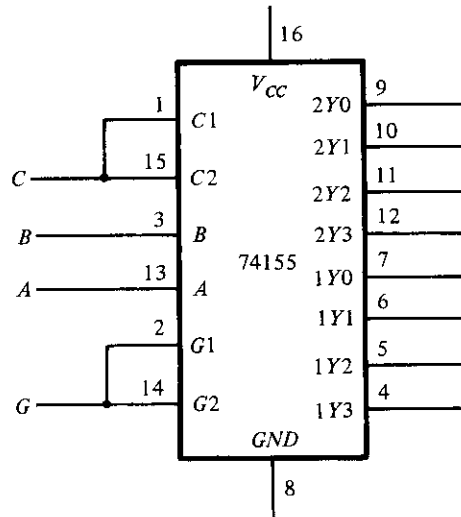
$$F_1 = xz + x'y'z'$$

$$F_2 = x'y + xy'z'$$

$$F_3 = xy + x'y'z$$

Implement and test the combinational circuit using a 74155 decoder IC and external NAND gates.

The block diagram of the decoder and its truth table are shown in Fig. 11-7. The 74155 can be connected as a dual 2×4 decoder or as a single 3×8 decoder. When a 3×8 decoder is desired, inputs $C1$ and $C2$ must be connected together as well as inputs $G1$ and $G2$, as shown in the block diagram. The function of the circuit is similar to the one shown in Fig. 5-10. G is the enable input and must be equal to 0 for proper operation. The eight outputs are labeled with symbols given in the data book. As in Fig. 5-10, the 74155 uses NAND gates, with the result that the selected output goes to 0 while all other outputs remain at 1. The implementation with the decoder is as shown in Fig. 5-9, except that the OR gates must be replaced with external NAND gates when the 74155 is used.



Truth table

Inputs				Outputs							
<i>G</i>	<i>C</i>	<i>B</i>	<i>A</i>	2Y0	2Y1	2Y2	2Y3	1Y0	1Y1	1Y2	1Y3
1	X	X	X	1	1	1	1	1	1	1	1
0	0	0	0	0	1	1	1	1	1	1	1
0	0	0	1	1	0	1	1	1	1	1	1
0	0	1	0	1	1	0	1	1	1	1	1
0	0	1	1	1	1	1	0	1	1	1	1
0	1	0	0	1	1	1	1	0	1	1	1
0	1	0	1	1	1	1	1	1	0	1	1
0	1	1	0	1	1	1	1	1	1	0	1
0	1	1	1	1	1	1	1	1	1	1	0

FIGURE 11-7

IC type 74155 connected as a 3 × 8 decoder

11-5 CODE CONVERTERS

The conversion from one binary code to another is common in digital systems. In this experiment, you will design and construct three combinational-circuit converters. Code conversion is discussed in Section 4-5.

Gray Code to Binary. Design a combinational circuit with four inputs and four outputs that converts a four-bit Gray code number (Table 1-4) into the equivalent four-bit binary number. Implement the circuit with exclusive-OR gates. (This can be done with one 7486 IC.) Connect the circuit to four switches and four indicator lamps and check for proper operation.

9's Complementer. Design a combinational circuit with four input lines that represent a decimal digit in BCD and four output lines that generate the 9's complement of the input digit. Provide a fifth output that detects an error in the input BCD number. This output should be equal to logic-1 when the four inputs have one of the unused combinations of the BCD code. Use any of the gates listed in Fig. 11-1, but minimize the total number of ICs used.

Seven-Segment Display. A seven-segment indicator is used for displaying any one of the decimal digits 0 through 9. Usually, the decimal digit is available in BCD. A BCD-to-seven-segment decoder accepts a decimal digit in BCD and generates the corresponding seven-segment code. This is shown pictorially in Problem 4-16.

Figure 11-8 shows the connections necessary between the decoder and the display. The 7447 IC is a BCD-to-seven-segment decoder/driver. It has four inputs for the BCD digit. Input *D* is the most significant and input *A* the least significant. The 4-bit BCD digit is converted to a seven-segment code with outputs *a* through *g*. The outputs of the 7447 are applied to the inputs of the 7730 (or equivalent) seven-segment display. This

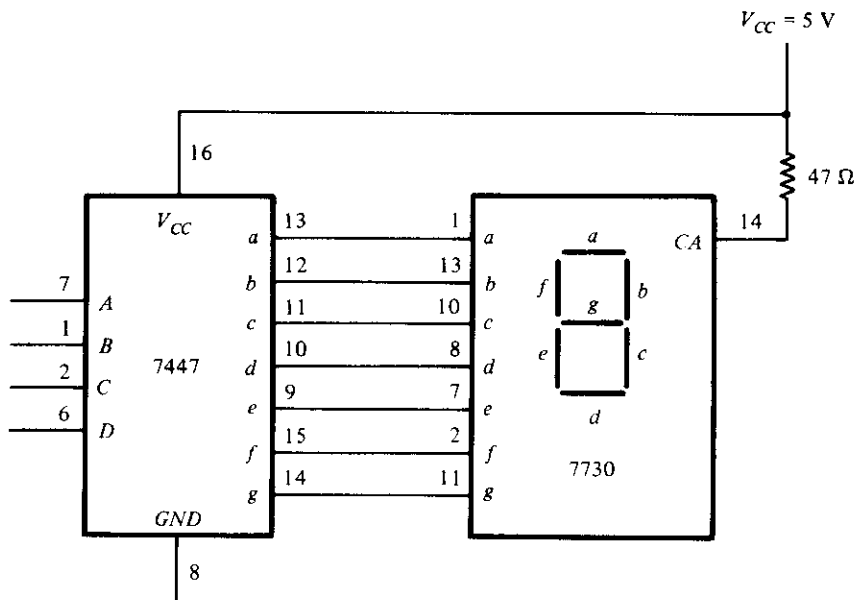


FIGURE 11-8

BCD-to-seven-segment decoder (7447) and seven-segment display (7730)

IC contains the seven LED (light-emitting diode) segments on top of the package. The input at pin 14 is the common anode (CA) for all the LEDs. A $47\text{-}\Omega$ resistor to V_{CC} is needed in order to supply the proper current to the selected LED segments. Other equivalent seven-segment display ICs may have additional anode terminals and may require different resistor values.

Construct the circuit shown in Fig. 11-8. Apply the 4-bit BCD digits through four switches and observe the decimal display from 0 to 9. Inputs 1010 through 1111 have no meaning in BCD. Depending on the decoder, these values may cause either a blank or a meaningless pattern to be displayed. Observe and record the output displayed patterns of the six unused input combinations.

11-6 DESIGN WITH MULTIPLEXERS

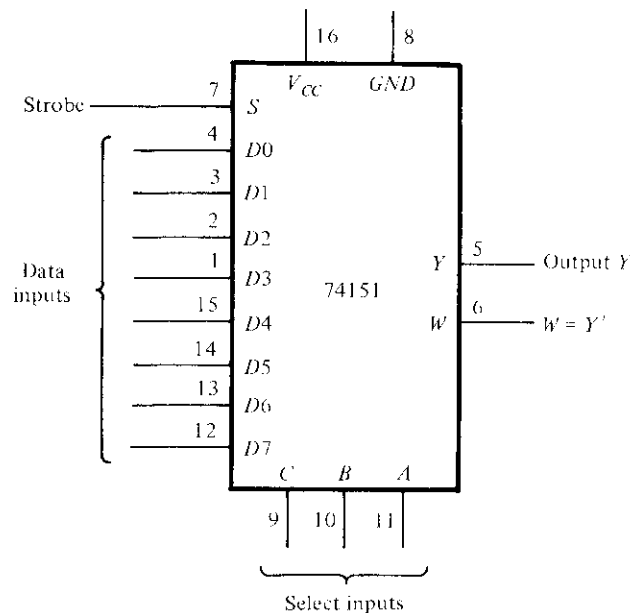
In this experiment, you will design a combinational circuit and implement it with multiplexers, as explained in Section 5-6. The multiplexer to be used is IC type 74151, shown in Fig. 11-9. The internal construction of the 74151 is similar to the diagram shown in Fig. 5-16 except that there are eight inputs instead of four. The eight inputs are designated $D0$ through $D7$. The three selection lines, C , B , and A , select the particular input to be multiplexed and applied to the output. A strobe control, S , acts as an enable signal. The function table specifies the value of output Y as a function of the selection lines. Output W is the complement of Y . For proper operation, the strobe input S must be connected to ground.

Design Specifications. A small corporation has 10 shares of stock, and each share entitles its owner to one vote at a stockholder's meeting. The 10 shares of stock are owned by four people as follows:

Mr. W: 1 share
Mr. X: 2 shares
Mr. Y: 3 shares
Mrs. Z: 4 shares

Each of these persons has a switch to close when voting yes and to open when voting no for his or her shares.

It is necessary to design a circuit that displays the total number of shares that vote yes for each measure. Use a seven-segment display and a decoder, as shown in Fig. 11-8, to display the required number. If all shares vote no for a measure, the display should be blank. (Note that binary input 15 into the 7447 blanks all seven segments.) If 10 shares vote yes for a measure, the display should show 0. Otherwise, the display shows a decimal number equal to the number of shares that vote yes. Use four 74151 multiplexers to design the combinational circuit that converts the inputs from the stock owners' switches into the BCD digit for the 7447. Do not use 5 V for logic-1. Use the output of an inverter whose input is grounded.



Function table

Strobe S	Select C B A			Output Y
1	X	X	X	0
0	0	0	0	$D0$
0	0	0	1	$D1$
0	0	1	0	$D2$
0	0	1	1	$D3$
0	1	0	0	$D4$
0	1	0	1	$D5$
0	1	1	0	$D6$
0	1	1	1	$D7$

FIGURE 11-9
IC type 74151 8×1 multiplexer

11-7 ADDERS AND SUBTRACTORS

In this experiment, you will construct and test various adder and subtractor circuits. The subtractor circuit is then used for comparing the relative magnitude of two numbers. Adders are discussed in Section 4-3. Subtraction with 2's complement is ex-

plained in Section 1-5. A 4-bit parallel adder is introduced in Section 5-2, and the comparison of two numbers is explained in Section 5-4.

Half-Adder. Design, construct, and test a half-adder circuit using one XOR gate and two NAND gates.

Full-Adder. Design, construct, and test a full-adder circuit using two ICs, 7486 and 7400.

Parallel Adder. IC type 7483 is a 4-bit binary parallel adder. Its internal construction is similar to Fig. 5-5. The pin assignment is shown in Fig. 11-10. The two 4-bit input binary numbers are A_1 through A_4 and B_1 through B_4 . The 4-bit sum is obtained from S_1 through S_4 . C_0 is the input carry and C_4 the output carry.

Test the 4-bit binary adder 7483 by connecting the power supply and ground terminals. Then connect the four A inputs to a fixed binary number such as 1001 and the B inputs and the input carry to five toggle switches. The five outputs are applied to indicator lamps. Perform the addition of a few binary numbers and check that the output sum and output carry give the proper values. Show that when the input carry is equal to 1, it adds 1 to the output sum.

Adder-Subtractor. The subtraction of two binary numbers can be done by taking the 2's complement of the subtrahend and adding it to the minuend. The 2's complement can be obtained by taking the 1's complement and adding 1. To perform $A - B$,

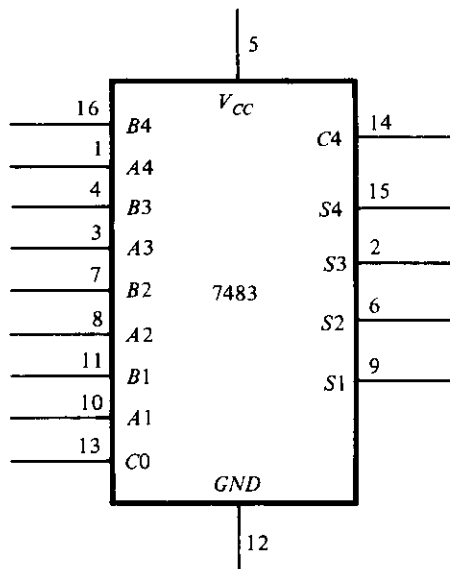
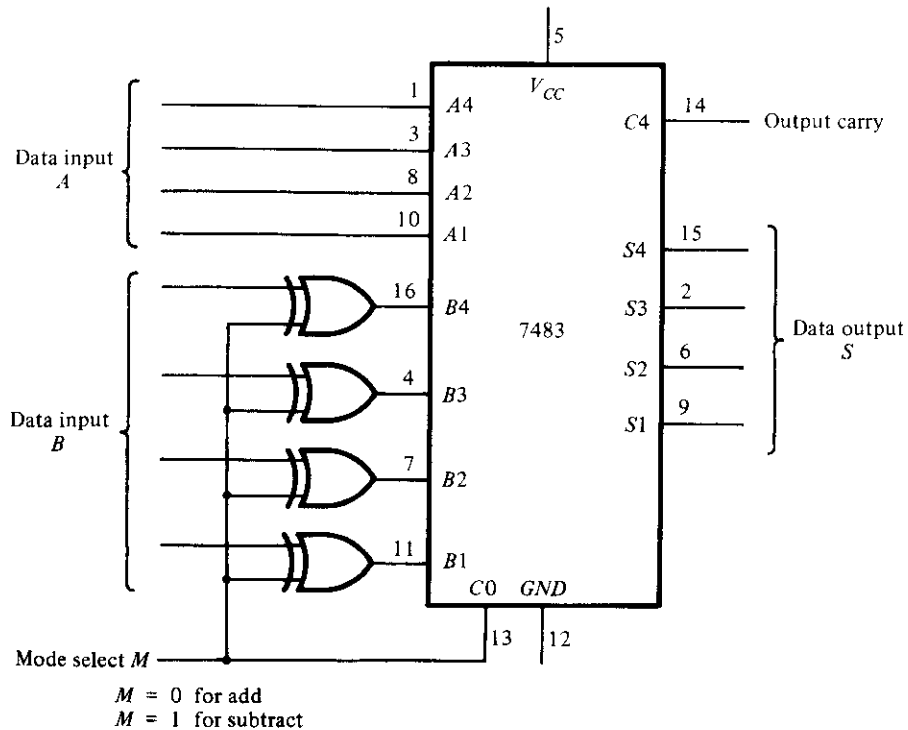


FIGURE 11-10
IC type 7483 4-bit binary adder

**FIGURE 11-11**

4-bit adder-subtractor

we complement the four bits of B , add them to the four bits of A , and add 1 through the input carry. This is done as shown in Fig. 11-11. The four XOR gates complement the bits of B when the mode select $M = 1$ (because $x \oplus 1 = x'$) and leave the bits of B unchanged when $M = 0$ (because $x \oplus 0 = x$). Thus, when the mode select M is equal to 1, the input carry $C0$ is equal to 1 and the sum output is A plus the 2's complement of B . When M is equal to 0, the input carry is equal to 0 and the sum generates $A + B$.

Connect the adder-subtractor circuit and test it for proper operation. Connect the four A inputs to a fixed binary number 1001 and the B inputs to switches. Perform the following operations and record the values of the output sum and the output carry $C4$.

$9 + 5$	$9 - 5$
$9 + 9$	$9 - 9$
$9 + 15$	$9 - 15$

Show that during addition, the output carry is equal to 1 when the sum exceeds 15. Also show that when $A \geq B$, the subtraction operation gives the correct answer, $A - B$, and the output carry $C4$ is equal to 1. But when $A < B$, the subtraction gives the 2's complement of $B - A$ and the output carry is equal to 0.

Magnitude Comparator. The comparison of two numbers is an operation that determines whether one number is greater than, equal to, or less than the other number. Two numbers, A and B , can be compared by first subtracting $A - B$ as done in Fig. 11-11. If the output in S is equal to zero, we know that $A = B$. The output carry from $C4$ determines the relative magnitude: when $C4 = 1$, we have $A \geq B$; when $C4 = 0$, we have $A < B$; and when $C4 = 1$ and $S \neq 0$, we have $A > B$.

It is necessary to supplement the subtractor circuit of Fig. 11-11 to provide the comparison logic. This is done with a combinational circuit that has five inputs, $S1$ through $S4$ and $C4$, and three outputs designated by x , y , and z , so that

$$x = 1 \quad \text{if } A = B \quad (S = 0000)$$

$$y = 1 \quad \text{if } A < B \quad (C4 = 0)$$

$$z = 1 \quad \text{if } A > B \quad (C4 = 1 \text{ and } S \neq 0000)$$

The combinational circuit can be implemented with the two ICs, 7404 and 7408.

Construct the comparator circuit and test its operation. Use at least two sets of numbers for A and B to check each of the outputs x , y , and z .

11-8 FLIP-FLOPS

In this experiment, you will construct, test, and investigate the operation of various flip-flop circuits. The internal construction of the flip-flops can be found in Sections 6-2 and 6-3.

SR Latch. The *SR* latch is a basic flip-flop made with two cross-coupled NAND gates. Construct a basic flip-flop circuit and connect the two inputs to switches and the two outputs to indicator lamps. Set the two switches to logic-1, then momentarily turn each switch separately to the logic-0 position and back to 1. Obtain the truth table of the circuit.

RS Flip-Flop. Construct a clocked *RS* flip-flop with four NAND gates. Connect the *S* and *R* inputs to two switches and the clock input to a pulser. Verify the characteristic table of the flip-flop.

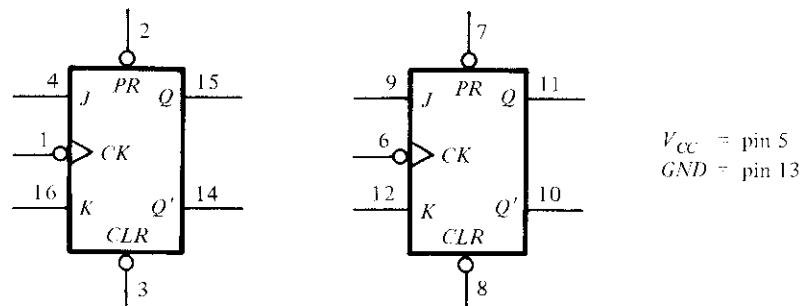
D Flip-Flop. Construct a clocked *D* flip-flop with four NAND gates. This can be done by modifying the circuit shown in Fig. 6-5 as follows. Remove gate number 5. Connect the output of gate 3 to the input of gate 4, where gate 5 was originally connected. Show that the modified circuit is identical to the original circuit by deriving the Boolean function for the output of gate 4 in each case. Connect the modified *D* flip-flop circuit and verify its characteristic table.

Master-Slave Flip-Flop. Construct a clocked master-slave *JK* flip-flop with one 7410 and two 7400 ICs. Connect the *J* and *K* inputs to logic-1 and the clock input to a

pulser. Connect the normal output of the master flip-flop to one indicator lamp and the normal output of the slave flip-flop to another indicator lamp. Press the push button in the pulser and then release it to produce a single positive pulse. Observe that the master flip-flop changes when the pulse goes positive and the slave flip-flop follows the change when the pulse goes negative. Repeat a few times while observing the two indicator lamps. Explain the transfer sequence from input to master and from master to slave.

Disconnect the clock input from the pulser and connect it to a clock generator with a frequency of 10 kHz or higher. Using a dual-trace oscilloscope, observe the waveforms of the clock and the master and slave outputs. Verify that the delay between the master and slave outputs is equal to the pulse width. Obtain a timing diagram showing the relationship between the clock waveform and the master and slave flip-flop outputs.

Edge-Triggered *D* Flip-Flop. Construct a *D*-type positive-edge-triggered flip-flop using six NAND gates. Connect the clock input to a pulser, the *D* input to a toggle switch, and the output *Q* to an indicator lamp. Set the value of *D* to the complement value of *Q*. Show that the flip-flop output changes only in response to a positive transi-



Function table





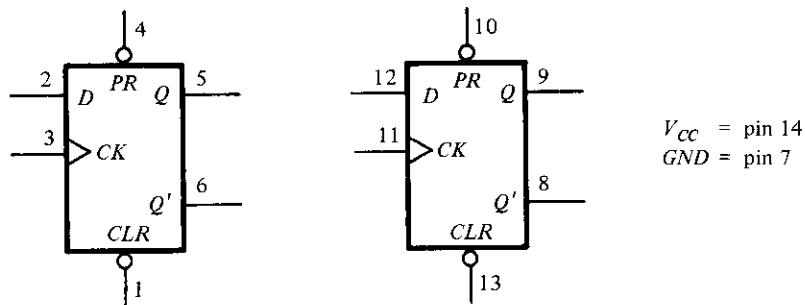
Inputs					Outputs	
Preset	Clear	Clock	<i>J</i>	<i>K</i>	<i>Q</i>	<i>Q'</i>
0	1	<i>X</i>	<i>X</i>	<i>X</i>	1	0
1	0	<i>X</i>	<i>X</i>	<i>X</i>	0	1
0	0	<i>X</i>	<i>X</i>	<i>X</i>	1	1
1	1		0	0	No change	
1	1		0	1	0	1
1	1		1	0	1	0
1	1		1	1	Toggle	

FIGURE 11-12
IC type 7476 dual *JK* master–slave flip-flops

tion of the clock pulse. Verify that the output does not change when the clock input is logic-1, or when the clock goes through a negative transition, or when it is logic-0. Continue changing the D input to correspond to the complement of the Q output at all times.

Disconnect the input from the pulser and connect it to the clock generator. Connect the complement output Q' to the D input. This causes the output to complement with each positive transition of the clock pulse. Using a dual-trace oscilloscope, observe and record the timing relationship between the input clock and output Q . Show that the output changes in response to a positive edge transition.

IC Flip-Flops. IC type 7476 consists of two JK master–slave flip-flops with preset and clear. The pin assignment for each flip-flop is shown in Fig. 11-12. The function table specifies the circuit operation. The first three entries in the table specify the operation of the asynchronous preset and clear inputs. These inputs behave like a NAND SR latch and are independent of the clock or the J and K inputs (the X 's indicate don't-care conditions). The last four entries in the function table specify the clock operation with both the preset and clear inputs maintained at logic-1. The clock value is shown as a



Function table

Inputs				Outputs	
Preset	Clear	Clock	D	Q	Q'
0	1	X	X	1	0
1	0	X	X	0	1
0	0	X	X	1	1
1	1	\uparrow	0	0	1
1	1	\uparrow	1	1	0
1	1	0	X	No change	

FIGURE 11-13

IC type 7474 dual D positive-edge-triggered flip-flops

single pulse. The positive transition of the pulse changes the master flip-flop, and the negative transition changes the slave flip-flop as well as the output of the circuit. With $J = K = 0$, the output does not change. The flip-flop toggles or complements when $J = K = 1$. Investigate the operation of one 7476 flip-flop and verify its function table.

IC type 7474 consists of two D positive-edge-triggered flip-flops with preset and clear. The pin assignment is shown in Fig. 11-13. The function table specifies the preset and clear operations and the clock operation. The clock is shown with an upward arrow to indicate that it is a positive-edge-triggered flip-flop. Investigate the operation of one of the flip-flops and verify its function table.

11-9 SEQUENTIAL CIRCUITS

In this experiment, you will design, construct, and test three synchronous sequential circuits. Use IC type 7476 JK flip-flops (Fig. 11-12) in all three designs. Choose any gate type that will minimize the total number of ICs. The design of synchronous sequential circuits is covered in Sections 6-7 and 6-8.

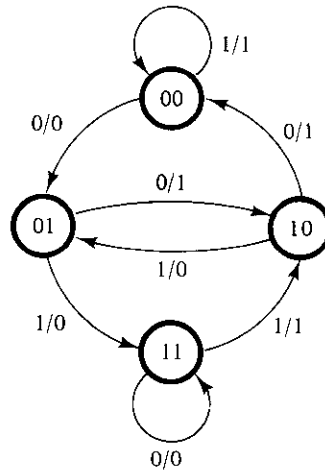
Up-Down Counter with Enable. Design, construct, and test a 2-bit counter that counts up or down. An enable input E determines whether the counter is on or off. If $E = 0$, the counter is disabled and remains at its present count even though clock pulses are applied to the flip-flops. If $E = 1$, the counter is enabled and a second input, x , determines the count direction. If $x = 1$, the circuit counts up with the sequence 00, 01, 10, 11, and the count repeats. If $x = 0$, the circuit counts down with the sequence 11, 10, 01, 00, and the count repeats. Do not use E to disable the clock. Design the sequential circuit with E and x as inputs.

State Diagram. Design, construct and test a sequential circuit whose state diagram is shown in Fig. 11-14. Designate the two flip-flops as A and B , the input as x , and the output as y .

Connect the output of the least significant flip-flop B to the input x and predict the sequence of states and output that will occur with the application of clock pulses. Verify the state transition and output by testing the circuit.

Design of Counter. Design, construct, and test a counter that goes through the following sequence of binary states: 0, 1, 2, 3, 6, 7, 10, 11, 12, 13, 14, 15, and back to 0 to repeat. Note that binary states 4, 5, 8, and 9 are not used. The counter must be self-starting; that is, if the circuit starts from any one of the four invalid states, the count pulses must transfer the circuit to one of the valid states to continue the count correctly.

Check the circuit operation for the required count sequence. Verify that the counter is self-starting. This is done by initializing the circuit to each unused state by means of the preset and clear inputs and then applying pulses to see whether the counter reaches one of the valid states.

**FIGURE 11-14**

State diagram for Experiment 9

11-10 COUNTERS

In this experiment, you will construct and test various ripple and synchronous counter circuits. Ripple counters are discussed in Section 7-4, and synchronous counters are covered in Section 7-5.

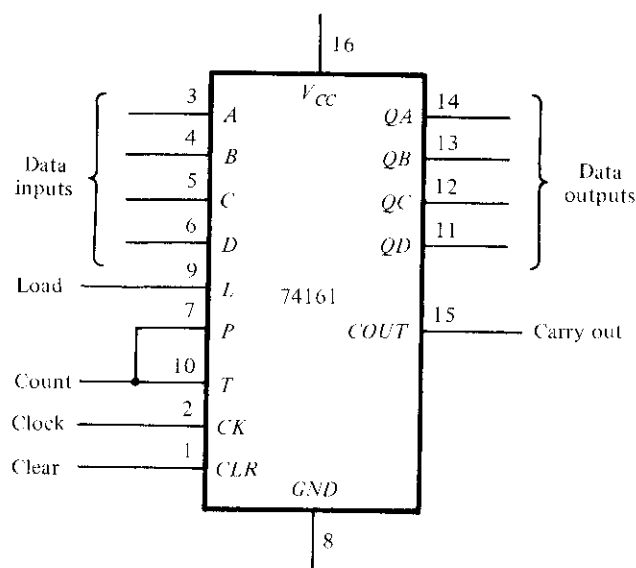
Ripple Counter. Construct a 4-bit binary ripple counter using two 7476 ICs (Fig. 11-12). Connect all asynchronous clear and preset inputs to logic-1. Connect the count-pulse input to a pulser and check the counter for proper operation.

Modify the counter so it will count down instead of up. Check that each input pulse decrements the counter by 1.

Synchronous Counter. Construct a synchronous 4-bit binary counter and check its operation. Use two 7476 ICs and one 7408 IC.

Decimal Counter. Design a synchronous BCD counter that counts from 0000 to 1001. Use two 7476 ICs and one 7408 IC. Test the counter for the proper sequence. Determine whether it is self-starting. This is done by initializing the counter to each of the six unused states by means of the preset and clear inputs. The application of pulses must transfer the counter to one of the valid states if the counter is self-starting.

Binary Counter with Parallel Load. IC type 74161 is a 4-bit synchronous binary counter with parallel load and asynchronous clear. The internal logic is similar to the circuit shown in Fig. 7-19. The pin assignment to the inputs and outputs is shown in Fig. 11-15. When the load signal is enabled, the four data inputs are transferred into four internal flip-flops, QA through QD , with QD being the most significant bit. There



Function table

Clear	Clock	Load	Count	Function
0	X	X	X	Clear outputs to 0
1	↑	0	X	Load input data
1	↑	1	1	Count to next binary value
1	↑	1	0	No change in output

FIGURE 11-15
IC type 74161 binary counter with parallel load

are two count-enable inputs called *P* and *T*. Both must be equal to 1 for the counter to operate. The function table is similar to Table 7-6 with one exception: the load input in the 74161 is enabled when equal to 0. To load the input data, the clear input must be equal to 1 and the load input must be equal to 0. The two count inputs have don't-care conditions and may be equal to either 1 or 0. The internal flip-flops trigger on the positive transition of the clock pulse. The circuit functions as a counter when the load input is equal to 1 and both count inputs *P* and *T* are equal to 1. If either *P* or *T* goes to 0, the output does not change. The carry-out output is equal to 1 when all four data outputs are equal to 1. Perform an experiment to verify the operation of the 74161 IC according to the function table.

Show how the 74161 IC together with a 2-input NAND gate can be made to operate as a synchronous BCD counter that counts from 0000 to 1001. Do not use the clear input. Use the NAND gate to detect the count of 1001, which then causes all 0's to be loaded into the counter.

Connect each of the four circuits shown in Fig. 7-20 to achieve a mod-6 counter. Remember that the load input is enabled when equal to 0 and therefore a NAND gate will be needed in each case.

11-11 SHIFT REGISTERS

In this experiment, you will investigate the operation of shift registers. The IC to be used is the 74195 shift register with parallel load. Shift registers are explained in Section 7-3. Some applications are introduced in Section 7-6.

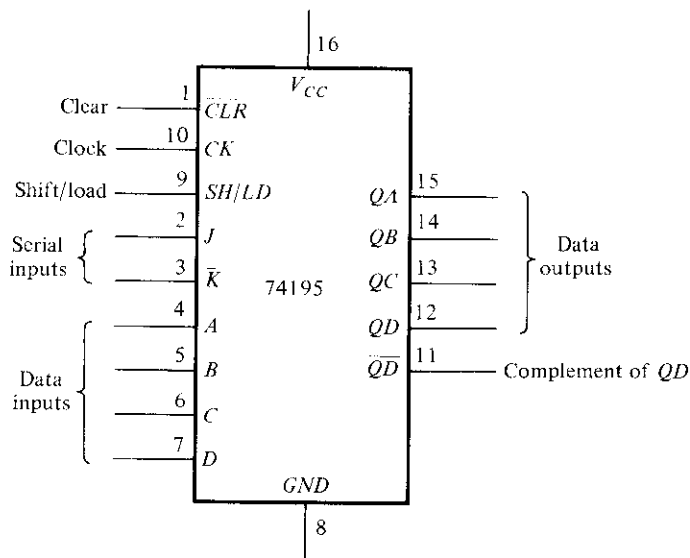
IC Shift Register. IC type 74195 is a 4-bit shift register with parallel load and asynchronous clear. The internal logic is similar to Fig. 7-9 except that the 74195 is unidirectional, that is, it is capable of shifting only in one direction. The pin assignment to the inputs and outputs is shown in Fig. 11-16. The single control line labeled SH/LD (shift/load) determines the synchronous operation of the register. When $SH/LD = 0$, the control input is in the load mode and the four data inputs are transferred into the four internal flip-flops, QA through QD . When $SH/LD = 1$, the control input is in the shift mode and the information in the register is shifted right from QA toward QD . The serial input into QA during the shift is determined from the J and \bar{K} inputs. The two inputs behave like the J and the complement of K of a JK flip-flop. When both J and \bar{K} are equal to 0, flip-flop QA is cleared to 0 after the shift. If both inputs are equal to 1, QA is set to 1 after the shift. The other two conditions for the J and \bar{K} inputs (not shown in the function table) provide a complement or no change in the output of flip-flop QA after the shift.

The function table for the 74195 shows the mode of operation of the register. When the clear input goes to 0, the four flip-flops clear to 0 asynchronously, that is, without the need of a clock. Synchronous operations are affected by a positive transition of the clock. To load the input data, the SH/LD must be equal to 0 and a positive clock-pulse transition must occur. To shift right, the SH/LD must be equal to 1. The J and \bar{K} inputs must be connected together to form the serial input.

Perform an experiment that will verify the operation of the 74195 IC. Show that it performs all the operations listed in the function table. Include in your function table the two conditions for $J\bar{K} = 01$ and 10.

Ring Counter. A ring counter is a circular shift register with the signal from the serial output QD going into the serial input. Connect the J and \bar{K} input together to form the serial input. Use the load condition to preset the ring counter to an initial value of 1000. Rotate the single bit with the shift condition and check the state of the register after each clock pulse.

A switch-tail ring counter uses the complement output of QD for the serial input. Preset the switch-tail ring counter to 0000 and predict the sequence of states that will result from shifting. Verify your prediction by observing the state sequence after each shift.



Function table

Clear	Shift/ load	Clock	J	\bar{K}	Serial input	Function
0	X	X	X	X	X	Asynchronous clear
1	X	0	X	X	X	No change in output
1	0	\uparrow	X	X	X	Load input data
1	1	\uparrow	0	0	0	Shift from QA toward QD, QA = 0
1	1	\uparrow	1	1	1	Shift from QA toward QD, QA = 1

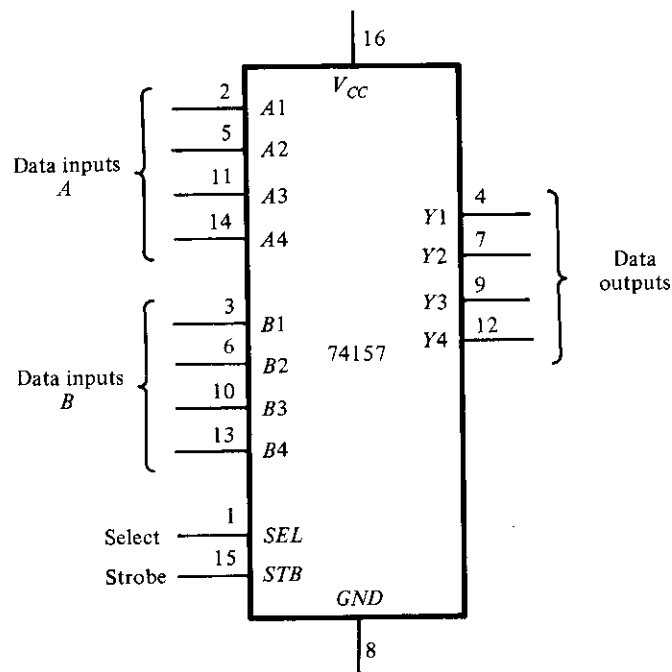
FIGURE 11-16
IC type 74195 shift register with parallel load

Feedback Shift Register. A feedback shift register is a shift register whose serial input is connected to some function of selected register outputs. Connect a feedback shift register whose serial input is the exclusive-OR of outputs *QC* and *QD*. Predict the sequence of states of the register starting from state 1000. Verify your prediction by observing the state sequence after each clock pulse.

Bidirectional Shift Register. The 74195 IC can shift only right from *QA* toward *QD*. It is possible to convert the register to a bidirectional shift register by using the load mode to obtain a shift left operation (from *QD* toward *QA*). This is accomplished by connecting the output of each flip-flop to the input of the flip-flop on its left and using the load mode of the *SH/LD* input as a shift-left control. Input *D* becomes the serial input for the shift-left operation.

Connect the 74195 as a bidirectional shift register (without parallel load). Connect the serial input for shift right to a toggle switch. Construct the shift left as a ring counter by connecting the serial output QA to the serial input D . Clear the register and then check its operation by shifting a single 1 from the serial input switch. Shift right three more times and insert 0's from the serial input switch. Then rotate left with the shift-left (load) control. The single 1 should remain visible while shifting.

Bidirectional Shift Register with Parallel Load. The 74195 IC can be converted to a bidirectional shift register with parallel load in conjunction with a multiplexer circuit. We will use IC type 74157 for this purpose. This is a quadruple 2-to-1-line multiplexers



Function table

Strobe	Select	Data outputs Y
1	X	All 0's
0	0	Select data inputs A
0	1	Select data inputs B

FIGURE 11-17
IC type 74157 quadruple 2×1 multiplexers

whose internal logic is shown in Fig. 5-17. The pin assignment to the inputs and outputs of the 74157 is shown in Fig. 11-17. Note that the enable input is called a strobe in the 74157.

Construct a bidirectional shift register with parallel load using the 74195 register and the 74157 multiplexer. The circuit should be able to perform the following operations:

1. Asynchronous clear
2. Shift right
3. Shift left
4. Parallel load
5. Synchronous clear.

Derive a table for the five operations as a function of the clear, clock, and SH/LD inputs of the 74195 and the strobe and select inputs of the 74157. Connect the circuit and verify your function table. Use the parallel-load condition to provide an initial value into the register and connect the serial outputs to the serial inputs of both shifts in order not to lose the binary information while shifting.

11-12 SERIAL ADDITION

In this experiment, you will construct and test a serial adder-subtractor circuit. Serial addition of two binary numbers can be done by means of shift registers and a full adder, as explained in Section 7-3. Example 7-3 shows that the number of gates for the full-adder can be reduced if a JK flip-flop is used for storing the carry.

Serial Adder. Starting from the diagram of Fig. 7-11, design and construct a 4-bit serial adder using the following ICs: 74195 (two), 7408, 7486, and 7476. Note that 74195 has a complement output for QD that is equivalent to the variables x' and y' in Fig. 7-11. Provide a facility for register B to accept parallel data from four toggle switches and connect its serial input to ground so that 0's are shifted into register B during the addition. Provide a toggle switch to clear the registers and the flip-flop. Another switch will be needed to specify whether register B is to accept parallel data or is to be shifted during the addition.

Testing the Adder. To test your serial adder, perform the binary addition $5 + 6 + 15 = 26$. This is done by first clearing the registers and the carry flip-flop. Parallel load the binary value 0101 into register B . Apply four pulses to add B to A serially and check that the result in A is 0101. (Note that clock pulses for the 7476 must be as shown in Fig. 11-12.) Parallel load 0110 into B and add it to A serially. Check that A has the proper sum. Parallel load 1111 into B and add to A . Check that the value in A is 1010 and that the carry flip-flop is set.

Clear the registers and flip-flop and try a few other numbers to verify that your serial adder is functioning properly.

Serial Adder-Subtractor. If we follow the procedure used in Example 7-3 to design a serial subtractor that subtracts $A - B$, we will find that the output difference is the same as the output sum, but that the input to the J and K of the borrow flip-flop needs the complement of x (see Problem 7-11 and its answer in the Appendix). Using the other two XOR gates from the 7486, convert the serial adder to a serial adder-subtractor with a mode control M . When $M = 0$, the circuit adds $A + B$. When $M = 1$, the circuit subtracts $A - B$ and the flip-flop holds the borrow instead of the carry.

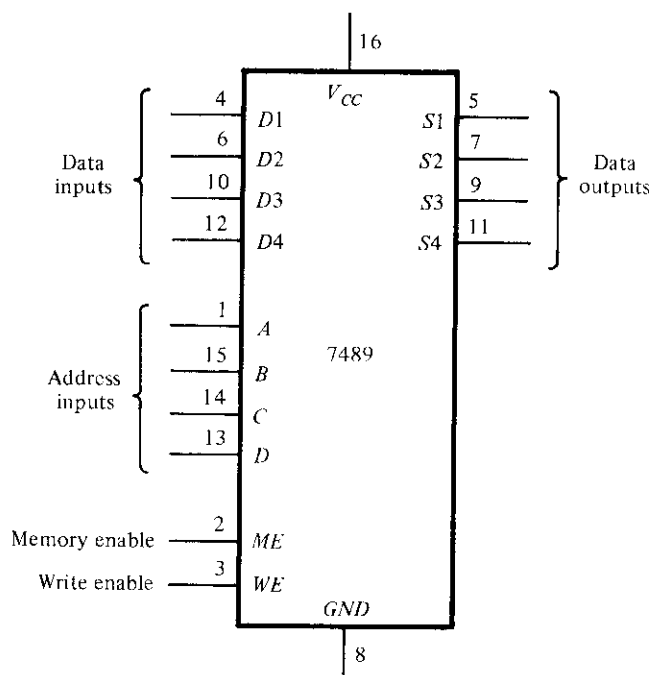
Test the adder part of the circuit by repeating the operations recommended above to ensure that the modification did not change the operation. Test the serial subtractor part by performing the operations $15 - 4 - 5 - 13 = -7$. Binary 15 can be transferred to register A by first clearing it to 0 and adding 15 from B . Check the intermediate results during the subtraction. Note that -7 will appear as the 2's complement of 7 with a borrow of 1 in the flip-flop.

11-13 MEMORY UNIT

In this experiment, you will investigate the behavior of a random-access memory (RAM) unit and its storage capability. The RAM will be used to simulate a read-only memory (ROM). The ROM simulator will then be used to implement combinational circuits, as explained in Section 5-7. The memory unit is discussed in Sections 7-7 and 7-8.

IC RAM. IC type 7489 is a 16×4 random-access memory. The internal logic is similar to the circuit shown in Fig. 7-27 for a 4×3 RAM. The pin assignment to the inputs and outputs is shown in Fig. 11-18. The four address inputs select one of 16 words in the memory. The least significant bit of the address is A , and the most significant is D . The memory enable (ME) input must be equal to 0 to enable the memory. If ME is equal to 1, the memory is disabled and all four outputs are at a logic-1 level. The write enable (WE) input determines the type of operation as indicated in the function table. The write operation is performed when $WE = 0$. This is a transfer of the binary number from the data inputs into the selected word in memory. The read operation is performed when $WE = 1$. This transfers the complement value stored in the selected word into the output data lines. The outputs are open-collector to allow external wired logic for memory expansion.

Testing the RAM. An open-collector gate requires an external resistor for proper operation. However, an open-collector gate can be operated without an external resistor if its output is connected to the input of another gate. Since the outputs of the 7489 produce the complement values, we might as well insert four inverters to change the outputs to their normal value and, at the same time, avoid the need for external resistors. The RAM can be tested after making the following connections. Connect the address



Function table

<i>ME</i>	<i>WE</i>	Operation	Data outputs
0	0	Write	Complement of data inputs
0	1	Read	Complement of selected word
1	X	Disable	All 1's

FIGURE 11-18

IC type 7489 16 × 4 RAM

inputs to a binary counter using the 7493 IC, as shown in Fig. 11-3. Connect the four data inputs to toggle switches and the data outputs to four 7404 inverters. Provide four indicator lamps for the address and four more for the outputs of the inverters. Connect input *ME* to ground and *WE* to a toggle switch (or a pulser that provides a negative pulse). Store a few words into the memory and then read them to verify that the write and read operations are functioning properly. You must be careful when using the *WE* switch. Always leave the *WE* input in the read mode, unless you want to write into memory. The proper way to write is first to set the address in the counter and the inputs in the four toggle switches. To store the word in memory, flip the *WE* switch to the write position and then return it to the read position. Be careful not to change the address or the inputs when *WE* is in the write mode.

ROM Simulator. A ROM simulator is obtained from a RAM when operated in the read mode only. The pattern of 1's and 0's is first entered into the simulating RAM by placing the unit momentarily in the write mode. Simulation is achieved by placing the unit in the read mode and taking the address lines as inputs for the ROM. The ROM can then be used to implement any combinational circuit.

Implement a combinational circuit using the ROM simulator that converts a 4-bit binary number to its equivalent Gray code as defined in Table 1-4. This is done as follows. Obtain the truth table of the code converter. Store the truth table into the 7489 memory by setting the address inputs to the binary value and the data inputs to the corresponding Gray code value. After all 16 entries of the table are written in memory, the ROM simulator is set by connecting the *WE* line to logic-1 permanently. Check the code converter by applying the inputs to the address lines and verifying the correct outputs in the data output lines.

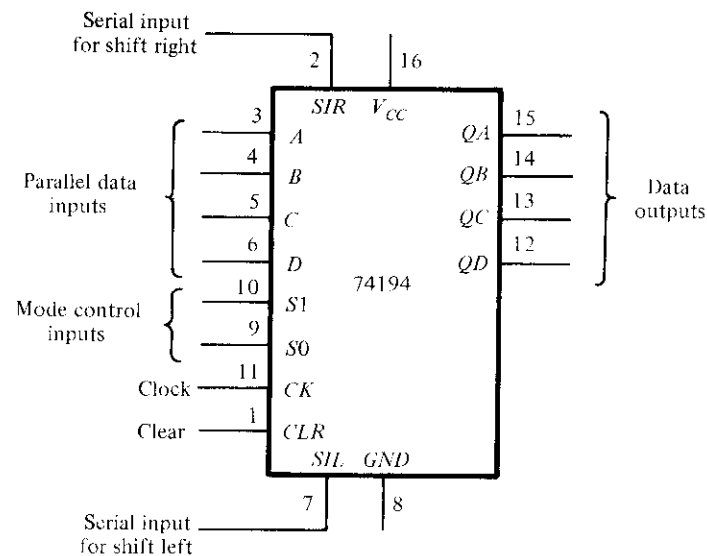
Memory Expansion. Expand the memory unit to a 32×4 RAM using two 7489 ICs. Use the *ME* inputs to select between the two ICs. Note that since the data outputs are open-collector, you can tie pairs of terminals together to obtain a logic wired-OR operation in conjunction with the output inverter. Test your circuit by using it as a ROM simulator that adds a 3-bit number to a 2-bit number to produce a 4-bit sum. For example, if the input of the ROM is 10110, then the output is calculated to be $101 + 10 = 0111$. (The first three bits of the input represent 5, the last two bits represent 2, and the output sum is binary 7.) Use the counter to provide four bits of the address and a switch for the fifth bit of the address.

11-14 LAMP HANDBALL

In this experiment, you will construct an electronic game of handball using a single light to simulate the moving ball. This project demonstrates the application of a bi-directional shift register with parallel load. It also shows the operation of the asynchronous inputs of flip-flops. We will first introduce an IC that is needed for this experiment and then present the logic diagram of the simulated lamp handball game.

IC Type 74194. This is a 4-bit bidirectional shift register with parallel load. The internal logic is similar to Fig. 7-9. The pin assignment to the inputs and outputs is shown in Fig. 11-19. The two mode-control inputs determine the type of operation as specified in the function table. The operation of the circuit is described in detail in Section 7-3 in conjunction with Fig. 7-9.

Logic Diagram. The logic diagram of the electronic lamp handball is shown in Fig. 11-20. It consists of two 74194 ICs, a dual *D* flip-flop 7474 IC, and three gate ICs: 7400, 7404, and 7408. The ball is simulated by a moving light that is shifted left or right through the bidirectional shift register. The rate at which the light moves is determined by the frequency of the clock. The circuit is first initialized with the *reset*

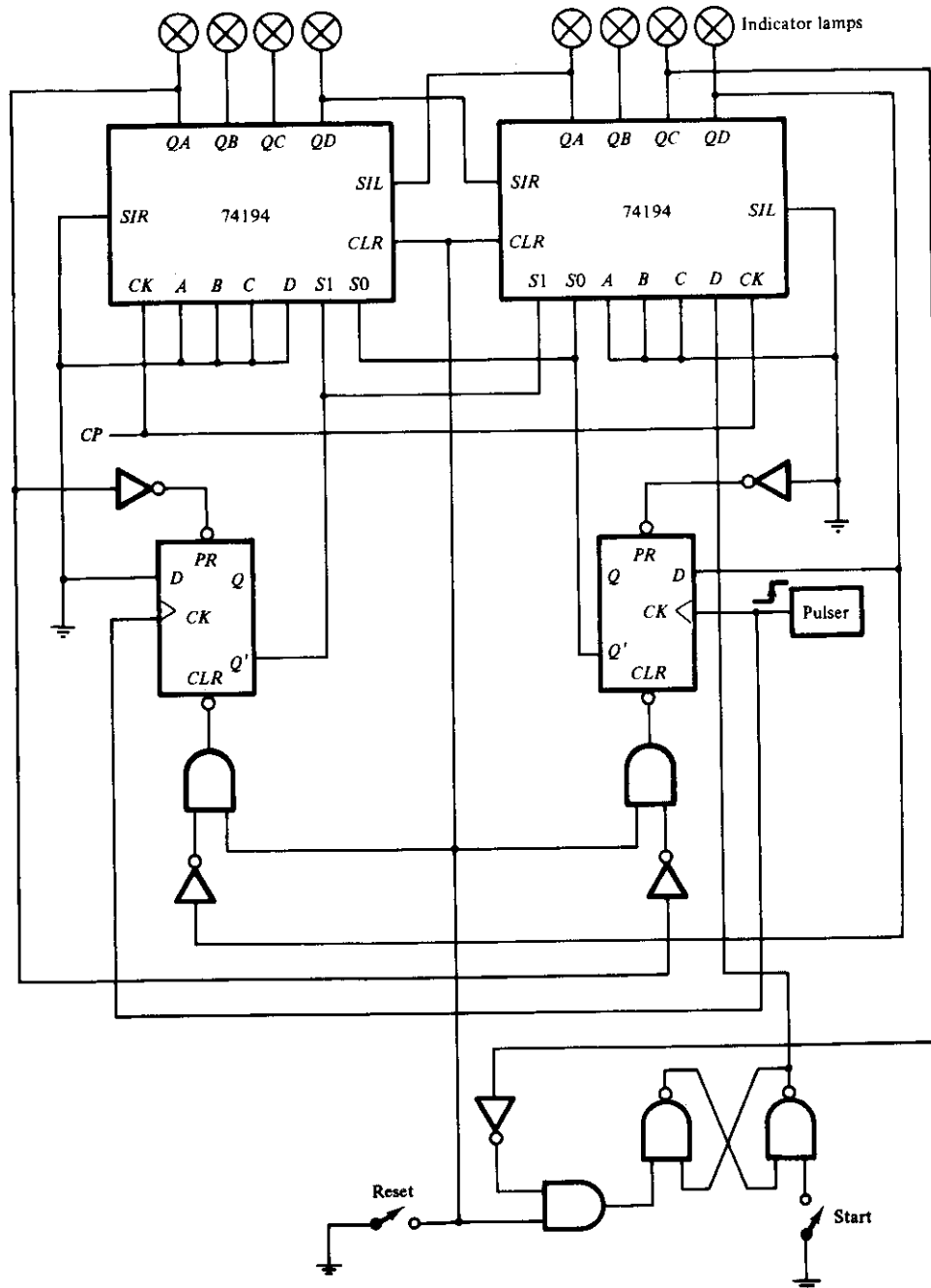


Function table

		Mode		Function
Clear	Clock	$S1$	$S0$	
0	X	X	X	Clear outputs to 0
1	\uparrow	0	0	No change in output
1	\uparrow	0	1	Shift right in the direction from QA to QD . SIR to QA
1	\uparrow	1	0	Shift left in the direction from QD to QA . SIL to QD
1	\uparrow	1	1	Parallel-load input data

FIGURE 11-19
IC type 74194 bidirectional shift register with parallel load

switch. The *start* switch starts the game by placing the ball (an indicator lamp) at the extreme right. The player must press the pulser push button to start the ball moving to the left. The single light shifts to the left until it reaches the leftmost position (the wall), at which time the ball returns to the player by reversing the direction of shift of the moving light. When the light is again at the rightmost position, the player must press the pulser again to reverse the direction of shift. If the player presses the pulser too soon or too late, the ball disappears and the light goes off. The game can be restarted by turning the start switch on and then off. The start switch must be open (logic-1) during the game.

**FIGURE 11-20**

Lamp handball logic diagram

Circuit Analysis. Prior to connecting the circuit, analyze the logic diagram to ensure that you understand how the circuit operates. In particular, try to answer the following questions:

1. What is the function of the reset switch?
2. Explain how the light in the rightmost position comes on when the start switch is grounded. Why is it necessary to place the start switch in the logic-1 position before the game starts?
3. What happens to the two mode-control inputs, $S1$ and $S0$, once the ball is set in motion?
4. What happens to the mode-control inputs and to the ball if the pulser is pressed while the ball is moving to the left? What happens if it is moving to the right but has not reached the rightmost position yet?
5. Suppose that the ball returned to the rightmost position, but the pulser has not been pressed yet; what is the state of the mode-control inputs if the pulser is pressed? What happens if it is not pressed?

Playing the Game. Wire the circuit of Fig. 11-20. Test the circuit for proper operation by playing the game. Note that the pulser must provide a positive-edge transition and that both the reset and start switches must be open (be in the logic-1 state) during the game. Start with a low clock rate and increase the clock frequency to make the handball game more challenging.

Counting the Number of Losses. Design a circuit that keeps score of the number of times the player loses while playing the game. Use a BCD-to-seven-segment decoder and a seven-segment display as in Fig. 11-8 to display the count from 0 through 9. Counting is done with a decimal counter using either the 7493 as a ripple decimal counter or the 74161 and a NAND gate as a synchronous decimal counter. The display should show 0 when the circuit is reset. Every time the ball disappears and the light goes off, the display should increase by 1. If the light stays on during the play, the number in the display should not change. The final design should be an automatic scoring circuit, with the decimal display incremented automatically each time the player loses when the light disappears.

Lamp Ping-Pong. Modify the circuit of Fig. 11-20 so as to obtain a lamp Ping-Pong game. Two players can participate in this game, with each player having his own pulser. The player with the right pulser returns the ball when in the extreme right position, and the player with the left pulser returns the ball when in the extreme left position. The only modification required for the Ping-Pong game is a second pulser and a change of few wires.

With a second start circuit, the game can be made to start (serve) by either one of the two players. This addition is optional.

11-15 CLOCK-PULSE GENERATOR

In this experiment, you will use an IC timer unit and connect it to produce clock pulses at a given frequency. The circuit requires the connection of two external resistors and two external capacitors. The cathode-ray oscilloscope is used to observe the waveforms and measure the frequency.

IC Timer. IC type 72555 (or 555) is a precision timer circuit whose internal logic is shown in Fig. 11-21. (The resistors, R_A and R_B , and the two capacitors are not part of the IC.) It consists of two voltage comparators, a flip-flop, and an internal transistor.

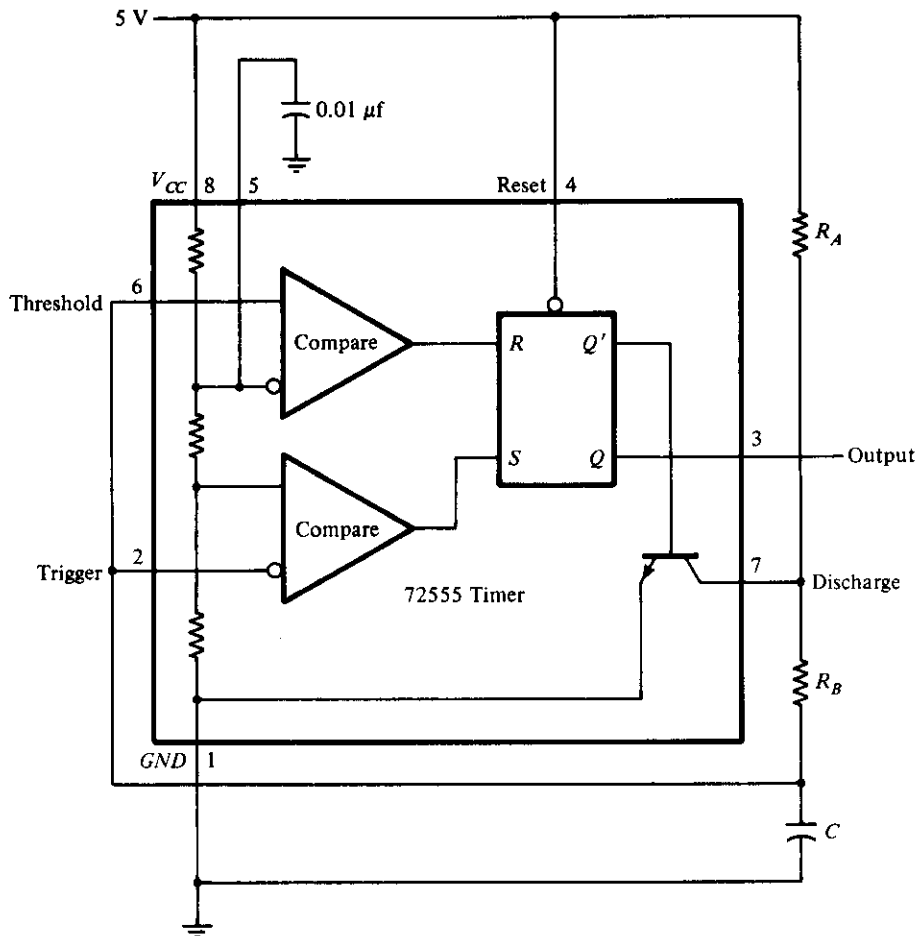


FIGURE 11-21

IC type 72555 timer connected as a clock-pulse generator

The voltage division from $V_{CC} = 5\text{ V}$ through the three internal resistors to ground produce $\frac{2}{3}$ and $\frac{1}{3}$ of V_{CC} (3.3 V and 1.7 V) into the fixed inputs of the comparators. When the threshold input at pin 6 goes above 3.3 V, the upper comparator resets the flip-flop and the output goes low to about 0 V. When the trigger input at pin 2 goes below 1.7 V, the lower comparator sets the flip-flop and the output goes high to about 5 V. When the output is low, Q' is high and the base-emitter junction of the transistor is forward-biased. When the output is high, Q' is low and the transistor is cut off (see Section 10-2). The timer circuit is capable of producing accurate time delays controlled by an external RC circuit. In this experiment the IC timer will be operated in the astable mode to produce clock pulses.

Circuit Operation. Figure 11-21 shows the external connections for the astable operation. The capacitor C charges through resistors R_A and R_B when the transistor is cut off and discharges through R_B when the transistor is forward-biased and conducting. When the charging voltage across capacitor C reaches 3.3 V, the threshold input at pin 6 causes the flip-flop to reset and the transistor turns on. When the discharging voltage reaches 1.7 V, the trigger input at pin 2 causes the flip-flop to set and the transistor turns off. Thus, the output continually alternates between two voltage levels at the output of the flip-flop. The output remains high for a duration equal to the charge time. This duration is determined from the equation

$$t_H = 0.693(R_A + R_B)C$$

The output remains low for a duration equal to the discharge time. This duration is determined from the equation

$$t_L = 0.693R_B C$$

Clock-Pulse Generator. Starting with a capacitor C of $0.001\text{ }\mu\text{F}$, calculate values for R_A and R_B to produce clock pulses, as shown in Fig. 11-22. The pulse width is $1\text{ }\mu\text{s}$ in the low level, and it is repeating at a frequency rate of 100 kHz (every $10\text{ }\mu\text{s}$). Connect the circuit and check the output in the oscilloscope.

Observe the output across the capacitor C and record its two levels to verify that they are between the trigger and threshold values.

Observe the waveform in the collector of the transistor at pin 7 and record all pertinent information. Explain the waveform by analyzing the circuit action.

Connect a variable resistor (potentiometer) in series with R_A to produce a variable-

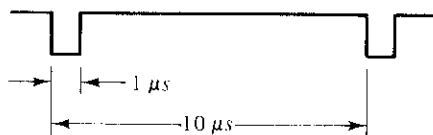


FIGURE 11-22

Output waveform for clock generator

frequency pulse generator. The low-level duration remains at 1 μ s. The frequency should range from 20 to 100 kHz.

Change the low-level pulses to high-level pulses with a 7404 inverter. This will produce positive pulses of 1 μ s with a variable-frequency range.

11-16 PARALLEL ADDER

In this experiment, you will construct a 4-bit parallel adder whose sum can be loaded into a register. The numbers to be added will be stored in a random-access memory. A set of binary numbers will be selected from memory and their sum will be accumulated in the register.

Block Diagram. Use the RAM circuit from the memory experiment of Section 11-13, a 4-bit parallel adder, a 4-bit shift register with parallel load, a carry flip-flop, and a multiplexer to construct the circuit. The block diagram and the ICs to be used are shown in Fig. 11-23. Information can be written into RAM from data in four switches or from the 4-bit data available in the outputs of the register. The selection is done by means of a multiplexer. The data in RAM can be added to the contents of the register and the sum transferred back to the register.

Control of Register. Provide toggle switches to control the 74194 register and the 7476 carry flip-flop as follows:

- (a) A LOAD condition to transfer the sum to the register and the output carry to the flip-flop upon application of a clock pulse.
- (b) A SHIFT condition to shift the register right with the carry from the carry flip-flop transferred into the leftmost position of the register upon application of a clock pulse. The value in the carry flip-flop should not change during the shift.
- (c) A NO-CHANGE condition that leaves the contents of the register and flip-flop unchanged even when clock pulses are applied.

Carry Circuit. In order to conform with the above specifications, it is necessary to provide a circuit between the output carry from the adder and the *J* and *K* inputs of the 7476 flip-flop so that the output carry is transferred into the flip-flop (whether it is equal to 0 or 1) only when the LOAD condition is activated and a pulse is applied to the clock input of the flip-flop. The carry flip-flop should not change if the LOAD condition is disabled or the SHIFT condition is enabled.

Detailed Circuit. Draw a detailed diagram showing all the wiring between the ICs. Connect the circuit and provide indicator lamps for the outputs of the register and carry flip-flop and for the address and output data of the RAM.

Checking the Circuit. Store the following numbers in RAM and then add them to

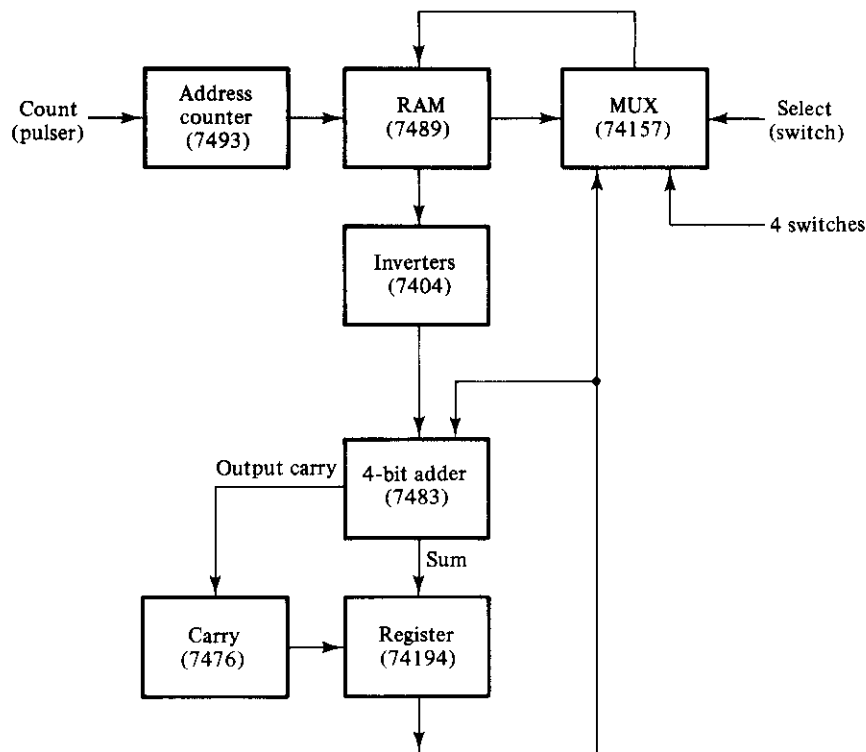


FIGURE 11-23
Block diagram of a parallel adder for Experiment 16

the register one at a time. Start with a cleared register and flip-flop. Predict the values in the output of the register and carry after each addition and verify your results.

$0110 + 1110 + 1101 + 0101 + 0011$

Circuit Operation. Clear the register and the carry flip-flop to zero and store the following 4-bit numbers in RAM in the indicated addresses.

Address	Content
0	0110
3	1110
6	1101
9	0101
12	0011

Now perform the following four operations:

1. Add the contents of address 0 to the contents of the register using the LOAD condition.
2. Store the sum from the register into RAM at address 1.
3. Shift right the contents of the register and carry with the SHIFT condition.
4. Store the shifted contents of the register at address 2 of RAM.

Check that the contents of the first three locations in RAM are as follows:

<u>Address</u>	<u>Content</u>
0	0110
1	0110
2	0011

Repeat the above four operations for each of the other four binary numbers stored in RAM. Use addresses 4, 7, 10, and 13 to store the sum from the register in step 2. Use addresses 5, 8, 11, and 14 to store the shifted value from the register in step 4. Predict what the contents of RAM at addresses 0 through 14 would be and check to verify your results.

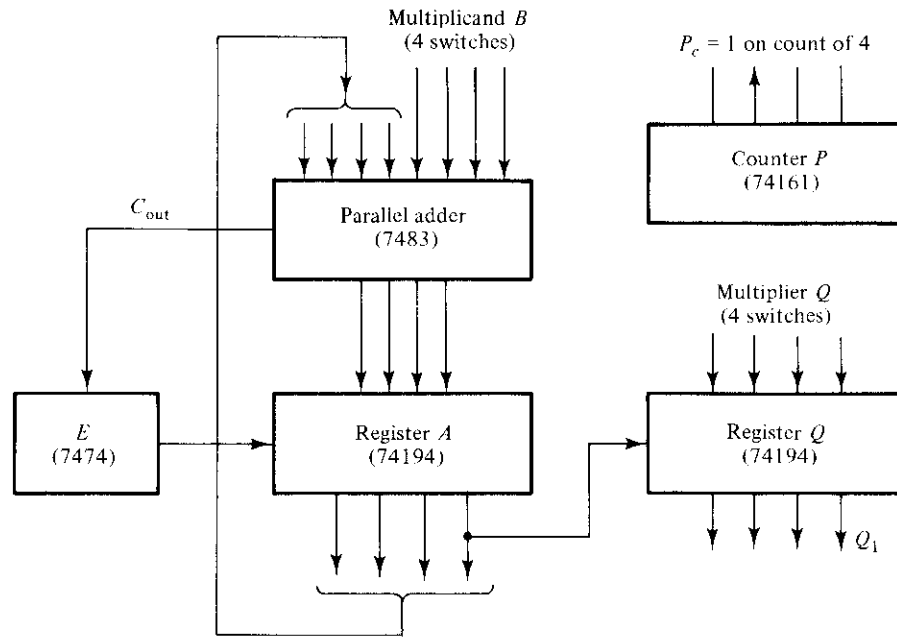
11-17 BINARY MULTIPLIER

In this experiment, you will design and construct a circuit that multiplies two 4-bit unsigned numbers to produce an 8-bit product. An algorithm for multiplying two binary numbers is presented in Section 8-6.

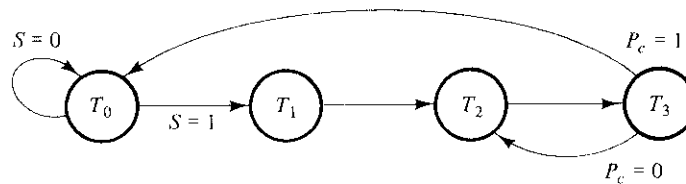
Block Diagram. The block diagram of the binary multiplier with the recommended ICs to be used is shown in Fig. 11-24(a). This is similar to the block diagram of Fig. 8-20 with some minor variations. The multiplicand B is available from four switches instead of a register. The multiplier Q is obtained from another set of four switches. The product is displayed with eight indicator lamps. Counter P is initialized to 0 and then incremented after each partial product is formed. When the counter reaches the count of four, output P_c becomes 1 and the multiplication operation terminates.

Control of Registers. The ASM chart for the binary multiplier in Fig. 8-21 shows that the three registers and the carry flip-flop are controlled with signals T_1 , T_2 , and T_3 . An additional control signal that depends on Q_1 loads the sum into register A and the output carry into flip-flop E . Q_1 is the least significant bit of register Q . The control-state diagram and the operations to be performed in each state are listed in Fig. 11-24(b). T_2Q_1 is generated with an AND gate whose inputs are T_2 and Q_1 . Note that carry flip-flop E can be cleared to 0 with every clock pulse except when the output carry is transferred to it.

Multiplication Example. Before connecting the circuit, make sure that you under-



(a) Data-processor block diagram


$$T_1: \quad A \leftarrow 0, E \leftarrow 0, P \leftarrow 0, Q \leftarrow \text{Multiplier}$$
$$T_7: \quad P \leftarrow P + 1$$
$$T_2 Q_1: A \leftarrow A + B, E \leftarrow C_{\text{out}}$$

T_3 : Shift right EAQ , $E \leftarrow 0$

(b) Control state diagram

FIGURE 11-24

Binary multiplier circuit

stand the operation of the multiplier. To do this, construct a table similar to the one shown in Fig. 8-22, but with $B = 1111$ for the multiplicand and $Q = 1011$ for the multiplier. Along each comment listed on the left side of the table, specify which one of the state variables, T_1 or T_2 or T_3 , is enabled in each case. (The states should start with T_1 and then repeat T_2 and T_3 four times.)

Data-Processor Design. Draw a detailed diagram of the data-processor part of the multiplier, showing all IC pin connections. Generate the control signals, T_1 , T_2 , and T_3 , with three switches and use them to provide the required control operations for the various registers. Connect the circuit and check that each component is functioning properly. With the three control variables at 0, set the multiplicand switches to 1111 and the multiplier switches to 1011. Sequence the control variables manually by means of the control switches as specified by the state diagram of Fig. 11-24(b). Apply a single pulse while in each control state and observe the outputs of registers A and Q and the values in E and P_c . Compare with the numbers in your numerical example to verify that the circuit is functioning properly. Note that IC type 74161 has master-slave flip-flops. To operate it manually, it is necessary that the single clock pulse be a negative pulse.

Design of Control. Design the control circuit specified by the state diagram. You can use any method of control implementation discussed in Section 8-4 (see also Problem 8-18). Choose the method that minimizes the number of ICs. Verify the operation of the control circuit prior to its connection to the data processor.

Checking the Multiplier. Connect the outputs of the control circuit to the data processor and verify the total circuit operation by repeating the steps of multiplying 1111 by 1011. The single clock pulses should now sequence the control states as well (remove the manual switches). The start signal S can be generated with a switch that is on while the control is in state T_0 .

Generate the start signal S with a pulser or any other short pulse and operate the multiplier with continuous clock pulses from a clock generator. Pressing the S pulser should initiate the multiplication operation and upon completion, the product should be displayed in the A and Q registers. Note that the multiplication will be repeated as long as signal S is enabled. Make sure that S goes back to 0, then set the switches to two other 4-bit numbers and press S again. The new product should appear at the outputs. Repeat the multiplication of a few numbers to verify the operation of the circuit.

11-18 ASYNCHRONOUS SEQUENTIAL CIRCUITS

In this experiment, you will analyze and design asynchronous sequential circuits. These type of circuits are presented in Chapter 9.

Analysis Example. The analysis of asynchronous sequential circuits with SR latches is outlined in Section 9-3. Analyze the circuit of Fig. P9-9 (shown with Problem 9-9) by deriving the transition table and output map of the circuit. From the transition table and output map, determine: (a) what happens to output Q when input x_1 is a 1 irrespective of the value of input x_2 ; (b) what happens to output Q when input x_2 is a 1 and x_1 is equal to 0; and (c) what happens to output Q when both inputs go back to 0?

Connect the circuit and show that it operates according to the way you analyzed it.

Design Example. The circuit of a positive-edge-triggered D -type flip-flop is shown

in Fig. 6-12. The circuit of a negative-edge T -type flip-flop is shown in Fig. 9-46. Using the six-step procedure recommended in Section 9-8, design, construct, and test a D -type flip-flop that triggers on both the positive and negative transitions of the clock. The circuit has two inputs, D and C , and a single output, Q . The value of D at the time C changes from 0 to 1 becomes the flip-flop output, Q . The output remains unchanged irrespective of the value of D as long as C remains at 1. On the next clock transition, the output is again updated to the value of D when C changes from 1 to 0. The output then remains unchanged as long as C remains at 0.