Chapter 10
Figure Rebound Effects

The classic table-tennis and squash video games rely on a bouncing, or rebounding, effect. Whenever the ball strikes a paddle or one of the fixed barriers, it instantly changes its direction of motion.

Such effects are appropriate for a lot of other kinds of video games as well, and the purpose of this chapter is to provide the background necessary for working the rebound effect into any desired custom game.

Most of the examples cited here assume the rebounding figure works from a slipping counter. As pointed out near the end of the chapter, however, it is just as easy to achieve the same effects with a figure generated by one of the position programming circuits in Chapter 8.

The basic idea behind the rebound effect is to sense the moment a moving object makes contact with a second object, generally a stationary one, then reverse the horizontal or vertical direction of the moving object.

Suppose, for example, a ball figure is moving horizontally to the right. It then strikes a fixed figure on the right-hand side of the screen and immediately switches direction so that it is moving horizontally to the left. That is one particular rebound effect. The same idea can be applied to vertical figure motion, with the object's direction of motion switching from up to down as it hits a figure near the top of the screen.

Figure 10-1 shows an up/down rebound scheme. It includes figures for generating fixed top and bottom figures as well as a ball

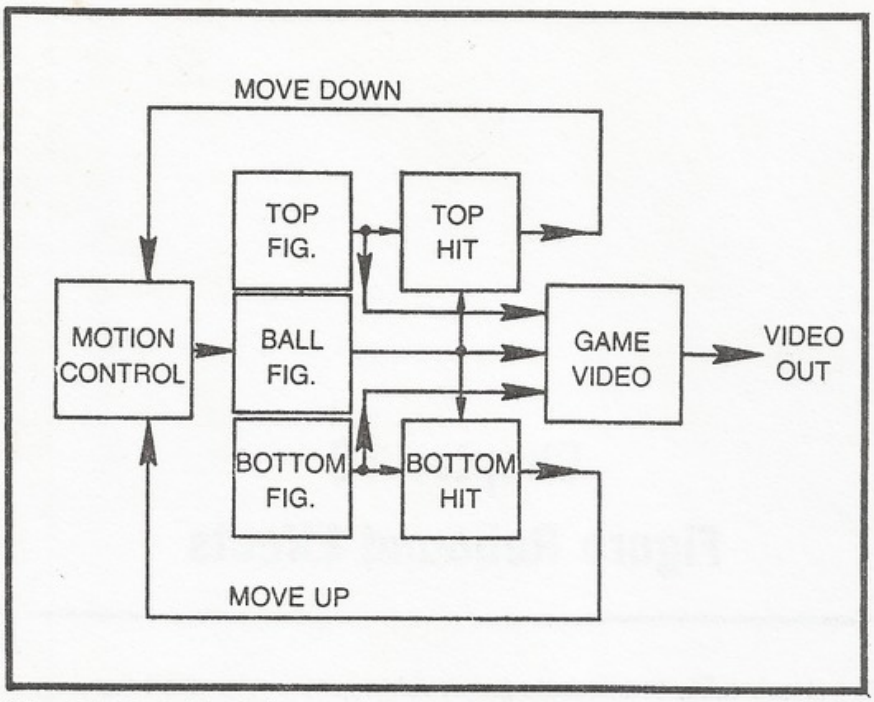


Fig. 10-1. Basic block diagram for rebound effects.

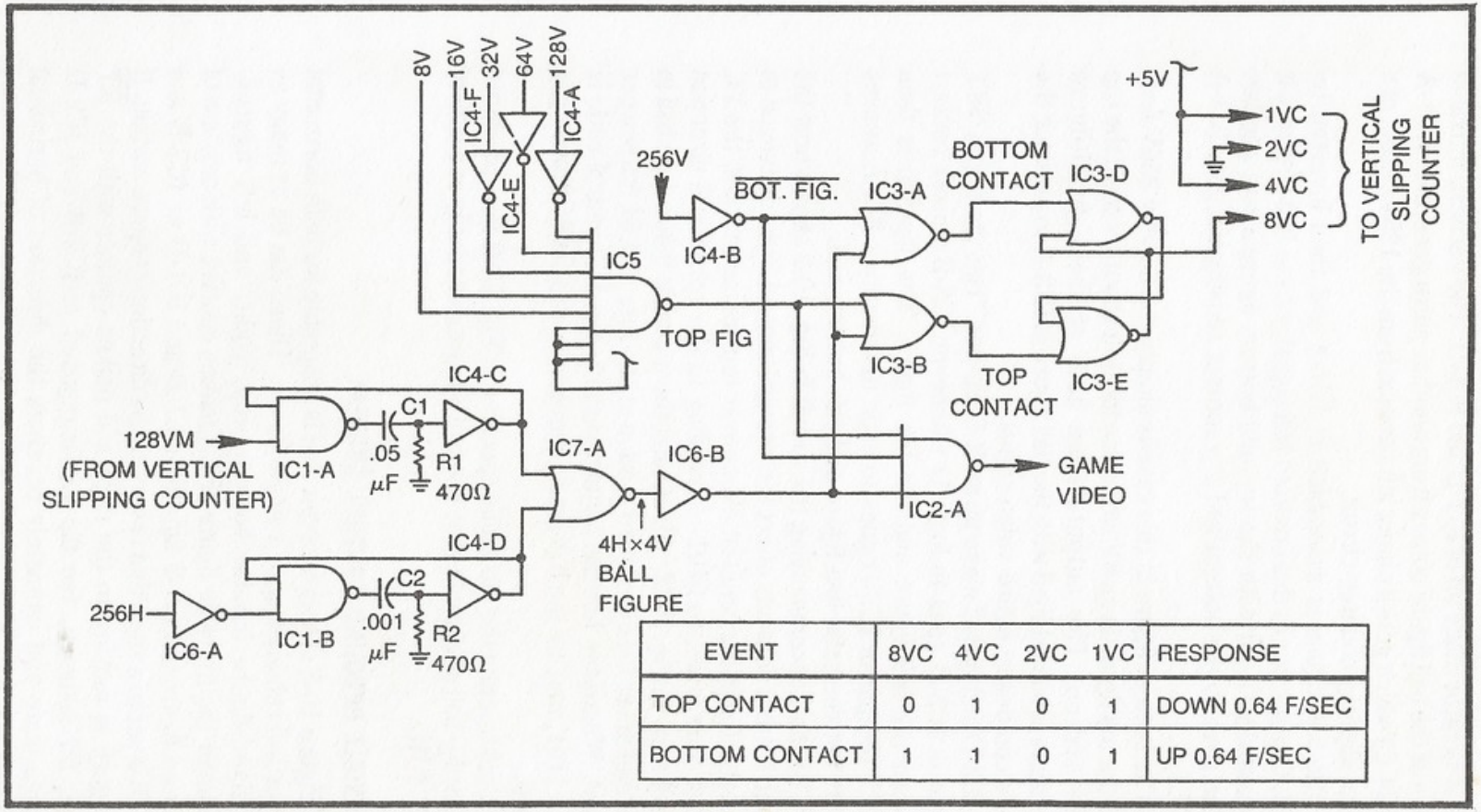
figure that is made movable from a motion-control circuit. The outputs of the three figure generators are combined in the game video circuit to produce a video output for displaying them on the TV screen.

Whenever the ball figure and top figure meet, however, the TOP HIT block generates a logic level that sets the ball's motion-control circuit for downward motion. Presumably the ball then moves in a downward direction until it contacts the bottom figure. At that moment, the BOTTOM HIT block generates a logic level that makes the motion control circuit move the ball figure upward.

The ball can thus bounce up and down between the top and bottom figures at a rate determined by the ball's velocity and the spacing between the fixed lines.

Of course the same general idea applies to horizontal ball motion, substituting left and right fixed figures for the top and bottom ones, and using a horizontal-motion-control circuit. Figure 10-2, however, shows the circuitry for the vertical ball-bouncing circuit.

The ball figure in this instance is a simple 4H × 4V square, and the fixed figures are white lines near the top and bottom of the screen. The horizontal position of the ball figure is fixed near the center of the screen by the $\overline{128H}$ signal being fed to a negative-edge



EVENT	8VC	4VC	2VC	1VC	RESPONSE
TOP CONTACT	0	1	0	1	DOWN 0.64 F/SEC
BOTTOM CONTACT	1	1	0	1	UP 0.64 F/SEC

Fig. 10-2. A vertical rebound demonstration circuit.

pulse generator made up of IC1-B and IC4-D. The vertical, movable portion of the ball figure comes from another pulse generator (IC1-A and IC4-C) which gets its input information from the 128V output of a vertical-slipping counter circuit.

The ball figure is assembled at IC7-A and then inverted by IC6-B. One portion of this inverted ball signal goes to IC2-A where it is essentially ORed with the top and bottom figures, and another portion goes to the bottom and top contact sensing circuits, IC3-A and IC3-B.

The bottom figure in this case is simply the narrow 256V line, while the top figure is an 8V white line located about 8V from the top of the screen. The outputs from both of these fixed-figure generators are combined with the ball figure in IC2-A to produce the game's composite figure video signal.

IC3-D and IC3-C make up an R-S flip-flop. This flip-flop is SET (output of IC3-D goes to logic 1) whenever IC3-B senses contact between the ball figure and the top figure. The flip-flop is then RESET (output of IC3-D goes to logic 0) whenever IC3-A senses contact between the ball figure and the bottom one.

The table accompanying the circuit in Fig. 10-2 shows how the top and bottom contacts affect the vertical control word delivered to the vertical-slipping counter. Whenever top contact occurs, the VC control code is set to 0101. According to the speed and direction control table in Fig. 7-22, this means the movable figure (the ball in this case) is set for downward motion at the rate of 0.64 frames per second. Whenever bottom contact occurs, the VC control code is set to 1101, making the ball figure move upward at 0.64 frames per second.

Constructing the circuit as specified in Fig. 10-2 thus produces a bouncing-ball effect whereby the ball bounces vertically at a rate of about 1 Hz.

A FLEXIBLE REBOUND CONTROL SYSTEM

Figure 10-3 shows a circuit that is adaptable for both horizontal and vertical rebounding of a single figure. There can be as many as four fixed-figure inputs: bottom, top, right, and left figures. Whenever the movable figure ($\overline{\text{FIG}}$) makes contact with any one of the fixed figures, an R-S flip-flop (IC1-B and IC1-D or IC2-B and IC2-D) is set to a state that reverses the direction of figure motion. If the figure is moving to the right and makes contact with the $\overline{\text{RT}}$ figure, for instance, the flip-flop composed of IC2-B and IC2-D changes state and ultimately reverses the direction of horizontal motion.

The horizontal- and vertical-motion codes are generated by a pair of quad 2:1 data multiplexers, IC3 and IC4. The experimenter can program the directions and rates as desired, and the R-S flip-flops automatically select the codes whenever a contact takes place.

The UP PROGRAM inputs to IC3 must be connected to a combination of 1s and 0s that will set the figure motion in an upward direction and at a rate determined by the experimenter. Likewise, the DOWN PROGRAM inputs fix the rate of motion in the downward direction. The same general idea applies to the RIGHT and LEFT PROGRAM inputs to the horizontal-motion selector, IC4.

The table in Fig. 10-4 shows the recommended pairs of UP and DOWN or RIGHT and LEFT program inputs. If the experimenter programs downward motion using 0101 for example, the UP PROGRAM should be 1101 after contact is made. Compare the data in this table with the master-control table in Fig. 7-22.

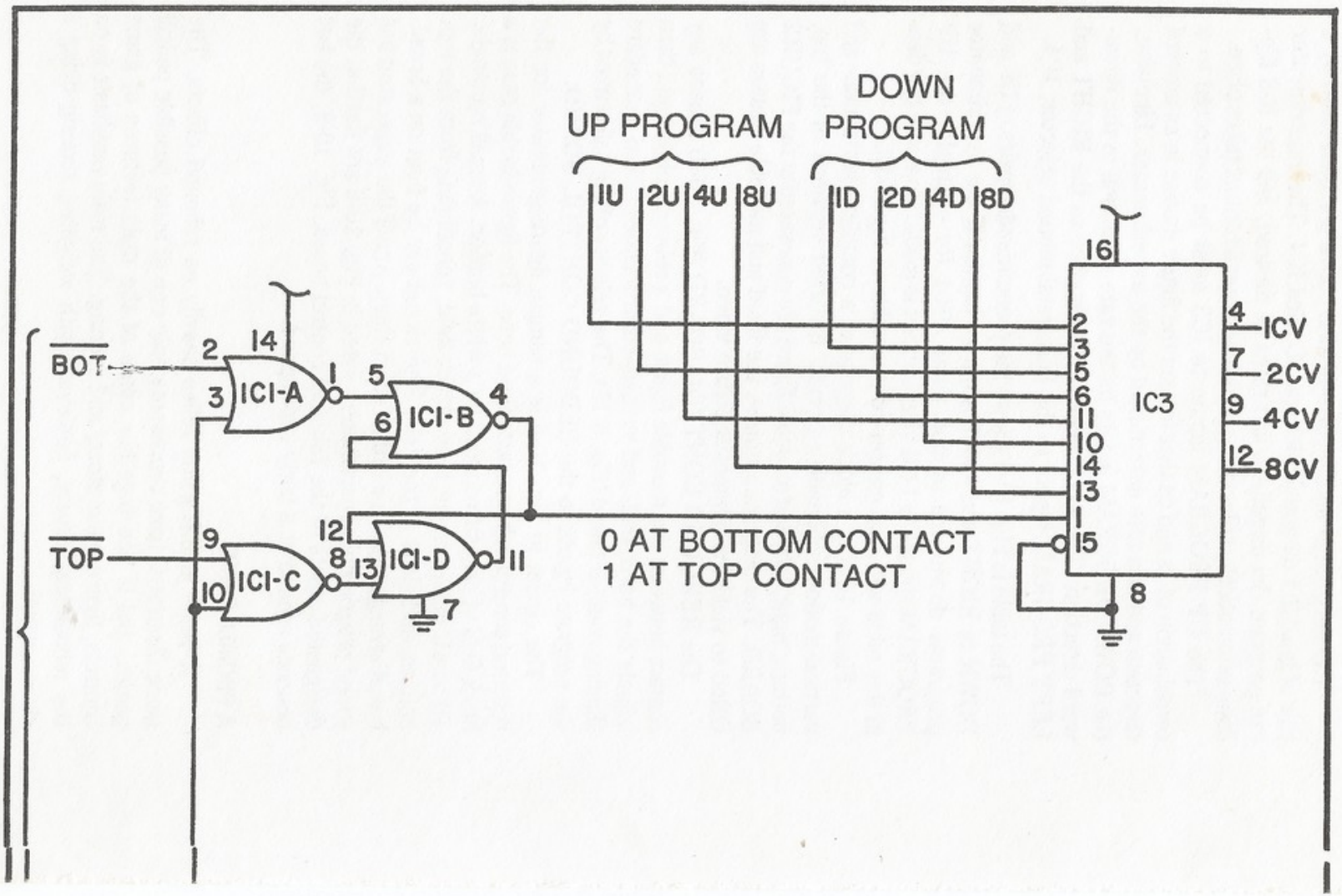
Figure 10-5 is a block diagram of a complete horizontal- and vertical-motion rebounding circuit. Inverted versions of the top, bottom, right, left, and movable figure are generated at the FIGURE BOARD. The diagram assumes the fixed and movable figures are ORed to yield the composite-figure video.

The REBOUND CONTROL BOARD (Fig. 10-3) senses any contact between the movable figure and a rebound object and then adjusts the horizontal- and vertical-control codes fed to a standard slipping-counter board (Fig. 5-15). The velocity of the ball is fixed by the program inputs to the REBOUND CONTROL BOARD.

The circuit in Fig. 10-6 is a sample figure-generator for the rebound control scheme described here. The figure in this case is a black field of play surrounded by a white border. A small rectangle, 4H × 4V moves about in the black field, rebounding from the top, bottom, and sides of the field. The circuit can be built on a breadboard arrangement in a rather short time. And if the suggested ball speed programming parameters listed in Fig. 10-6 are fixed at the designated inputs of the rebound control board, Fig. 10-3, the ball bounces around at a fairly high speed.

A PINBALL GAME

A video pinball game relies heavily on rebound effects. The game illustrated here represents only one of many possible pinball games, and it ties together some of the main features of game controls, figure generators, and scoring. The main emphasis is on the rebounding effects, however, with weighted scoring being a close second.


UP PROGRAM
DOWN PROGRAM
1U
2U
4U
8U
1D
2D
4D
8D
IC3
1CV
2CV
4CV
8CV
BOT
TOP
IC1-A
IC1-B
IC1-C
IC1-D
0 AT BOTTOM CONTACT
1 AT TOP CONTACT

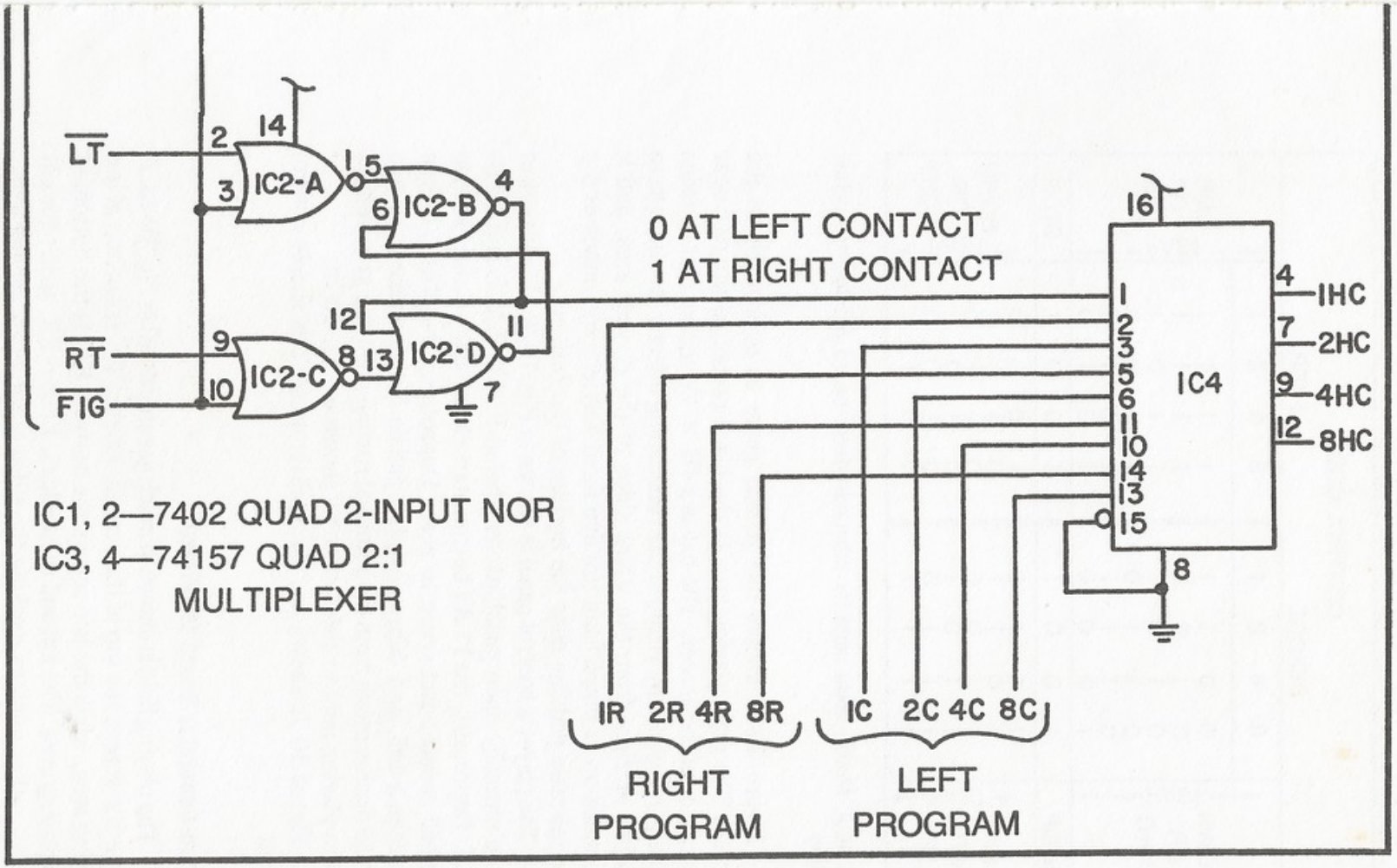


Fig. 10-3. An universal horizontal and vertical rebound control circuit.

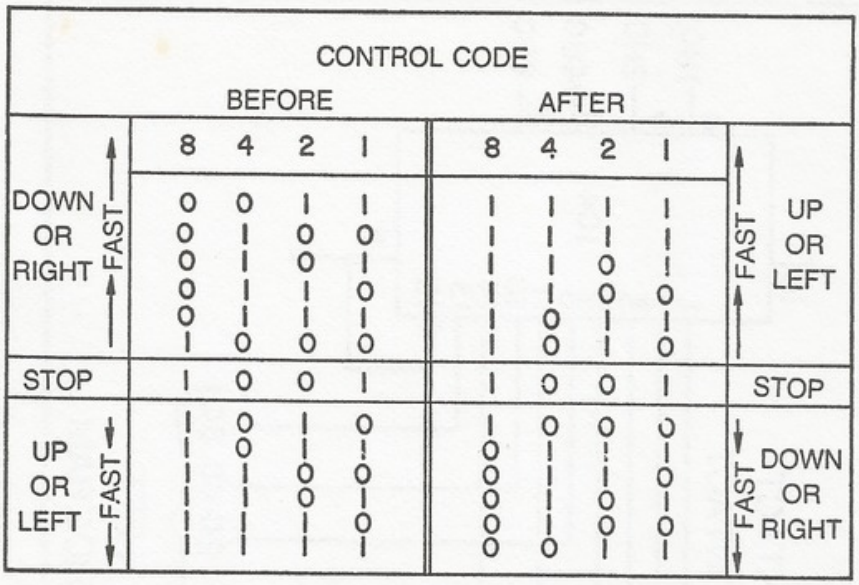


Fig. 10-4. Motion code table for rebound effects based on slipping-counter motion.

Figure 10-7a shows this pinball game as it appears on the screen. The main playing area is the black rectangle situated near the center of the screen. The ball is a 4H × 4V square that remains in the playing area by virtue of the rebounding effects. It rebounds to the left or right from the white sides of the playing area, and it rebounds up or down from the five fixed barriers and either of the two movable paddles near the bottom of the screen.

The player's control panel is shown in Fig. 10-7b. It consists of three normally-open pushbuttons labeled PLAY, BALL and PADDLE. Depressing the PLAY button starts the playing action, setting the ball counter and score to zero. Depressing the BALL button launches a ball, and depressing the paddle button makes the two paddle figures move from their normal resting positions at the edges of the playing field to the center as shown in Fig. 10-7a.

Figure 10-7c shows the wiring diagram for this simple control panel.

Figure Generator Board for Pinball

The playing field is shown in much greater detail in Fig. 10-8a. A white line across the top of the screen, labeled TE, is the top of the playing area, while the line across the bottom, BE, is the bottom of the playing area. The left and right edges, LE and RE, aren't lines at all, but rather borders marking the sides of the black playing field.

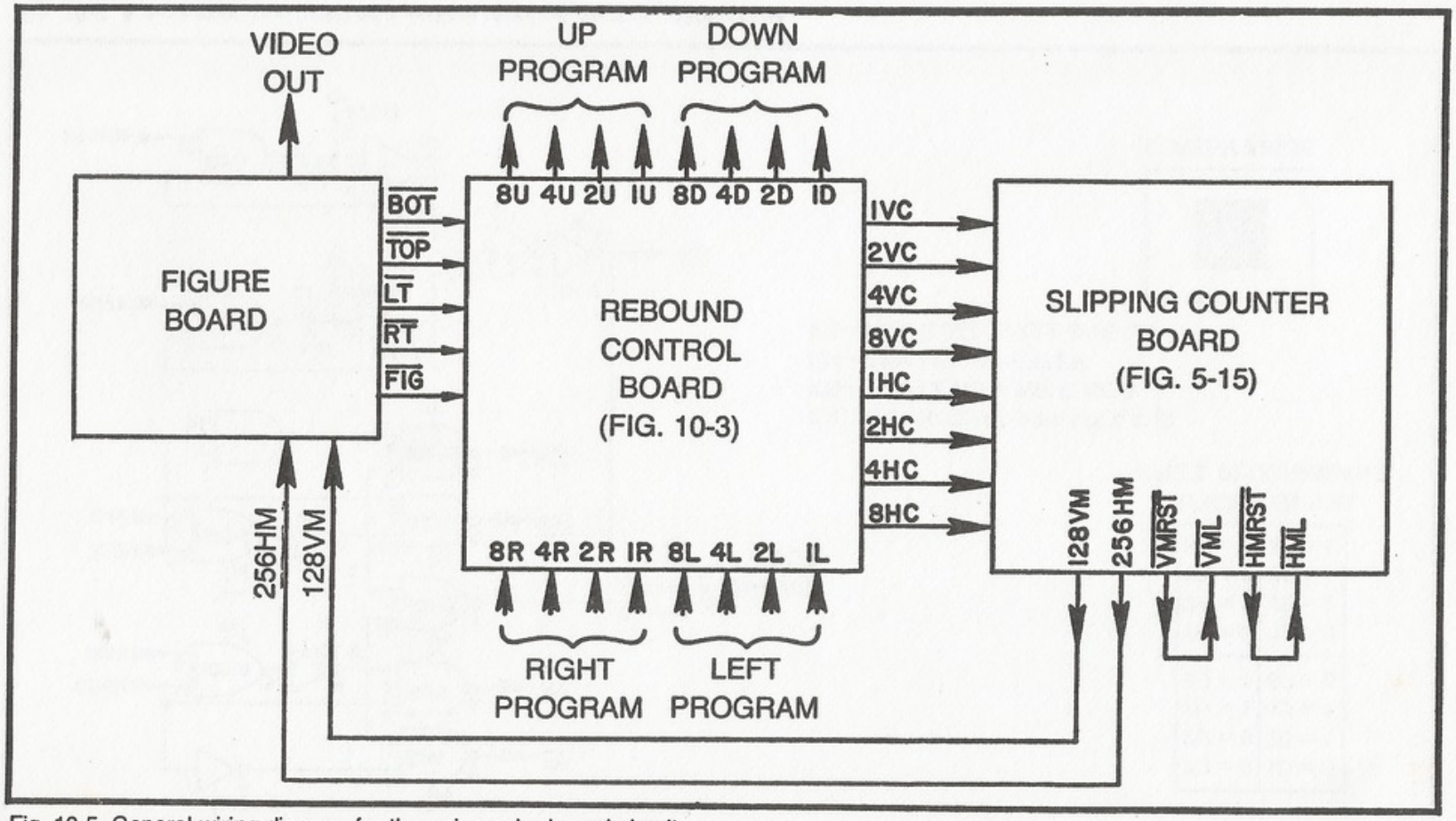


Fig. 10-5. General wiring diagram for the universal rebound circuit.

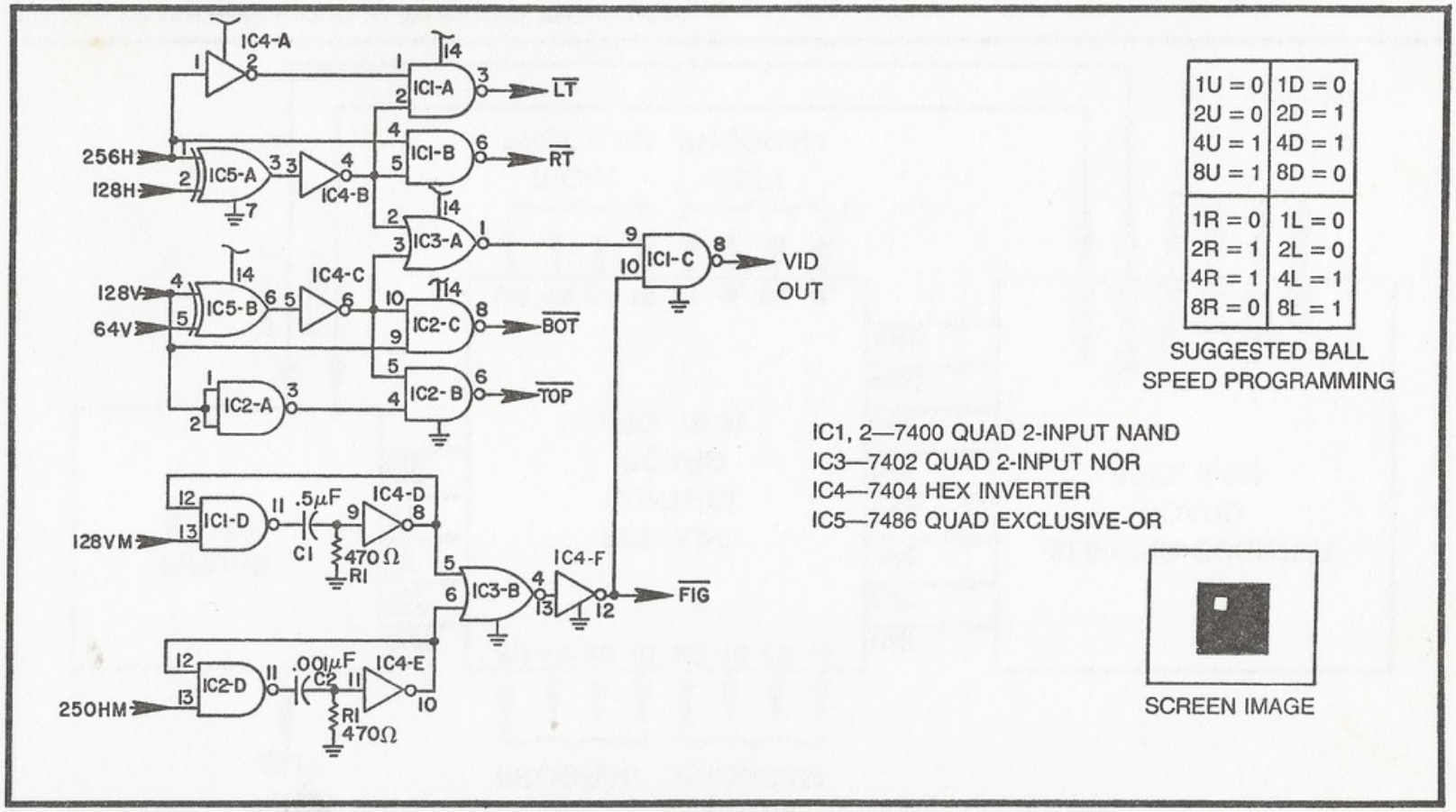


Fig. 10-6. A 4-counter, horizontal and vertical rebound demonstration circuit.

The fixed rebound barriers are labeled A through F, and are arranged in a symmetrical pattern about the vertical axis of the playing area.

PL and PR are the player's left and right paddles as they appear in their normal resting positions. The dashed figures labeled PL' and PR' indicate the same paddles as they appear when the player depresses the PADDLE pushbutton on the control panel.

This entire figure—the four edges, fixed barriers, and paddles—is generated by an 8 × 8 extended foldover-matrix generator of the type described in Chapter 4. The extension in this case is from top to bottom, while the foldover is around the vertical axis. Note, for instance, that the right-hand half of the figure is a mirror image of the left-hand half. The data programming for the matrix generator refers to the upper left-hand quadrant, and the remainder of the figure comes about by extension and foldover.

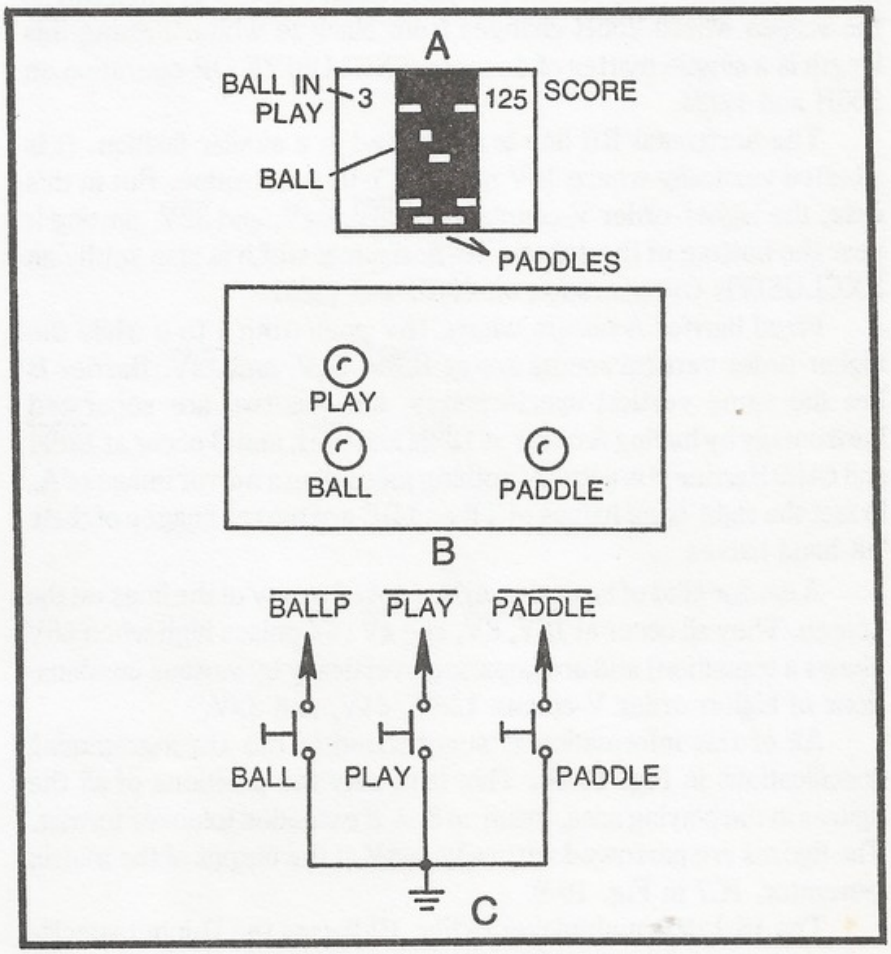


Fig. 10-7. A pinball game. (a) Figures appearing on the screen. (b) Player's control panel layout. (c) Control panel schematic diagram.

The waveforms shown in Fig. 10-8a and the matrix-programming parameters in Fig. 10-8c are vital to this figure-generating scheme. The truly unique feature, however, is the technique used for making the top and bottom edges, fixed barriers, and paddle figures appear as lines instead of the usual matrix squares and rectangles. This will be explained in connection with the actual figure-generating circuit in Fig. 10-9.

Note from the image and waveforms in Fig. 10-8a that a white line appears only where 16V makes a transition from 1 to 0. A line does not appear each time this sort of transition occurs, but it occurs only when 16V shows a 1-to-0 change.

The horizontal TE line, for instance, begins as 16V goes from 1 to 0. This line, like all the others, is 4V wide; so its position and vertical size are fixed by the vertical-count specifications, $\overline{128V}$, $\overline{64V}$, 32V, 16V, $\overline{8V}$, and $\overline{4V}$. The horizontal length of the TE line, and indeed the field of play, is one complete 128H cycle long, centered on the screen where 256H changes from black to white. Setting this length is a simple matter of doing an EXCLUSIVE OR operation on 256H and 128H.

The horizontal BE line is generated in a similar fashion. It is situated vertically where 16V makes a 1-to-0 transition. But in this case, the higher-order V-counts are $\overline{128V}$, $\overline{64V}$, and $\overline{32V}$, putting it near the bottom of the screen. Its horizontal width is also set by an EXCLUSIVE OR operation on 256H and 128H.

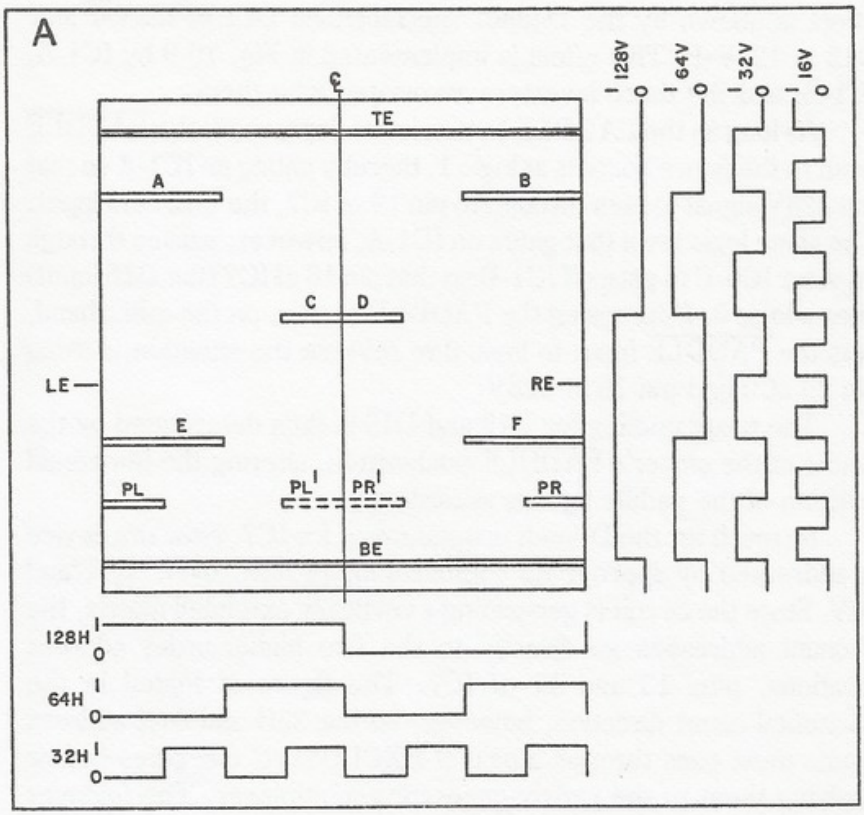
Fixed barrier A occurs where 16V goes from 1 to 0 while the higher-order vertical counts are at $\overline{128V}$, 64V, and $\overline{32V}$. Barrier B has the same vertical specifications, but the two are separated horizontally by having A occur at 128H and $\overline{64H}$, and B occur at $\overline{128H}$ and 64H. Barrier B is actually nothing more than a mirror image of A. In fact the right-hand halves of TE and BE are mirror images of their left-hand halves.

A similar kind of analysis can be applied to any of the lines on the screen. They all occur at $\overline{16V}$, $\overline{8V}$, and $\overline{4V}$ (4V pulses high when 16V shows a transition) and are separated vertically by various combinations of higher order V-counts 128V, 64V, and 32V.

All of this information is summarized in the D-programming specifications in Fig. 10-8c. This data sets the positions of all the figures in the playing area, using an 8 × 8 extended foldover format. The figures are narrowed vertically to 4V at the output of the matrix generator, IC7 in Fig. 10-9.

The 16:1 data multiplexer in Fig. 10-9 uses the D-input specifications derived from the figure and waveforms in Fig. 10-8a. As indicated by the presence of 128V and $\overline{128V}$ in the data program-

A



B

TE = TOP EDGE BE = BOTTOM EDGE LE = LEFT EDGE RE = RIGHT EDGE	A, B, C, D, E, F = FIXED PADDLES PL = LEFT PADDLE (NORMAL POSITION) PL^1 = LEFT PADDLE (HIT POSITION) PR = RIGHT PADDLE (NORMAL POSITION) PR^1 = RIGHT PADDLE (HIT POSITION)

C

D0 = $\overline{128V}$	D4 = $\overline{128V}$	D8 = 1	D12 = 128V•$\overline{P}$
D1 = $\overline{128V}$	D5 = $\overline{128V}$	D9 = 1	D 13 = 0
D2 = $\overline{128V}$	D6 = $\overline{128V}$	D10 = 0	D14 = 0
D3 = 1	D7 = $\overline{128V}$	D11 = 0	D15 = 128V•P

Fig. 10-8. Pinball playing area figure. (a) Basic figure and relevant H- and V-count waveforms. (b) Nomenclature for the Pinball figure. (c) D-input programming for the playing-area matrix generator.

ming, the figure is extended vertically by 128V: the upper half of the figure is generated while 128V is low, and the lower half is generated while 128V is high. This feature is implemented in Fig. 10-9 by the $\overline{128V}$ connection to a number of D inputs to IC7.

The paddle figures are to be in their resting position as long as the PADDLE button is not depressed, but then they should move to their center-screen positions when that button is depressed. This

effect is shown by the D-input specifications D12 = 128V•$\overline{P}$ and D15 = 128V•P. This effect is implemented in Fig. 10-9 by IC1-A, IC1-B, and the three inverters associated with them.

As long as the PADDLE button is *not* depressed, the $\overline{\text{PADDLE}}$ input to the figure board is at logic 1, thereby gating on IC1-A so that the 128V signal passes through to pin 19 of IC7, the D12 data input. The same logic level that gates on IC1-A, however, passes through inverter IC6-C to gate off IC1-B so that pin 16 of IC7 (the D15 input) sees a logic 0. Depressing the PADDLE button, on the other hand, sets the $\overline{\text{PADDLE}}$ input to logic 0 to reverse the situation, setting pin 19 at 0 and pin 16 at 128V.

The programming for D12 and D15 is thus determined by the status of the player's PADDLE pushbutton, altering the horizontal position of the paddle figures accordingly.

So much for the D-input programming for IC7. Now the device is addressed by appropriate combinations of 32H, 64H, 32V, and 64V. Since the circuit is generating a vertically extended matrix, the V-count addresses go directly to the two higher-order address locations, pins 13 and 11 of IC7. The figure is folded in the horizontal-count direction, however, so the 32H and 64H address inputs must pass through a pair of EXCLUSIVE OR gates before applying them to the matrix-generating multiplexer. The foldover effect makes the horizontal addressing run in the normal up-counting fashion as long as 128H is low. But when this input goes to logic 1 halfway across the playing area, it reverses the direction of horizontal addressing to create the mirrored left-hand half of the figure.

The EXCLUSIVE OR gates for creating the foldover effect are IC5-B and IC5-C. The "gear shift" in this instance is the 256H connection that is common to both of them.

You might find it necessary to study the extension and foldover effects in greater detail in Chapter 4. Without understanding the basic principles involved here, there is little hope of understanding this particular circuit or, more importantly, you will find it virtually impossible to modify the game or design any of your own.

The pinball figure is not windowed at all in the vertical direction. It occupies the entire height of the screen. It must be windowed horizontally, however. And in this case it is windowed by means of another EXCLUSIVE OR operation on 256H and 128H. These specifications put the playing area near the center of the screen, spanning one complete 128H cycle.

The windowing is implemented in Fig. 10-9 by the EXCLUSIVE OR gate, IC5-A . After the output of this gate is inverted by IC6-D, it is applied to the enabling input of IC7 at pin 9.

Even after going through all these D-input, addressing, and windowing steps, the figure information coming from the pin-10 output of IC7 only vaguely resembles that shown in Fig. 10-8a. Aside from being inverted (blacks and whites reversed) this output shows groups of 32H × 32V squares instead of 4V lines. What remains to be done to complete the figure-generating process is to invert the logic and narrow all the figures to 4V.

IC2-B in Fig. 10-9 uprights the contrast between blacks and whites, while the inputs to IC3-A narrow the figures to 4V. The figure information from IC2-B is thus NANDed with $\overline{16V}$, $\overline{8V}$, and $\overline{4V}$ to trim down the height of each element in the playing area to a 4V level whenever 16V shows a change from 1 to 0.

The white sides of the figure are added at IC3-B to create the basic pinball game figure. Narrowing the white top and bottom edges, fixed barriers, and paddles to 4V is the unique part of the operation. The rest of it comes from material already outlined in Chapter 4.

The little ball figure is generated by the pulse circuits composed of IC1-C, IC1-D, IC4-A, and IC4-C. It is assembled into a square by IC4-B. The original information for creating this movable ball figure comes from a slipping-counter board generating the necessary 256HM and 128VM signals.

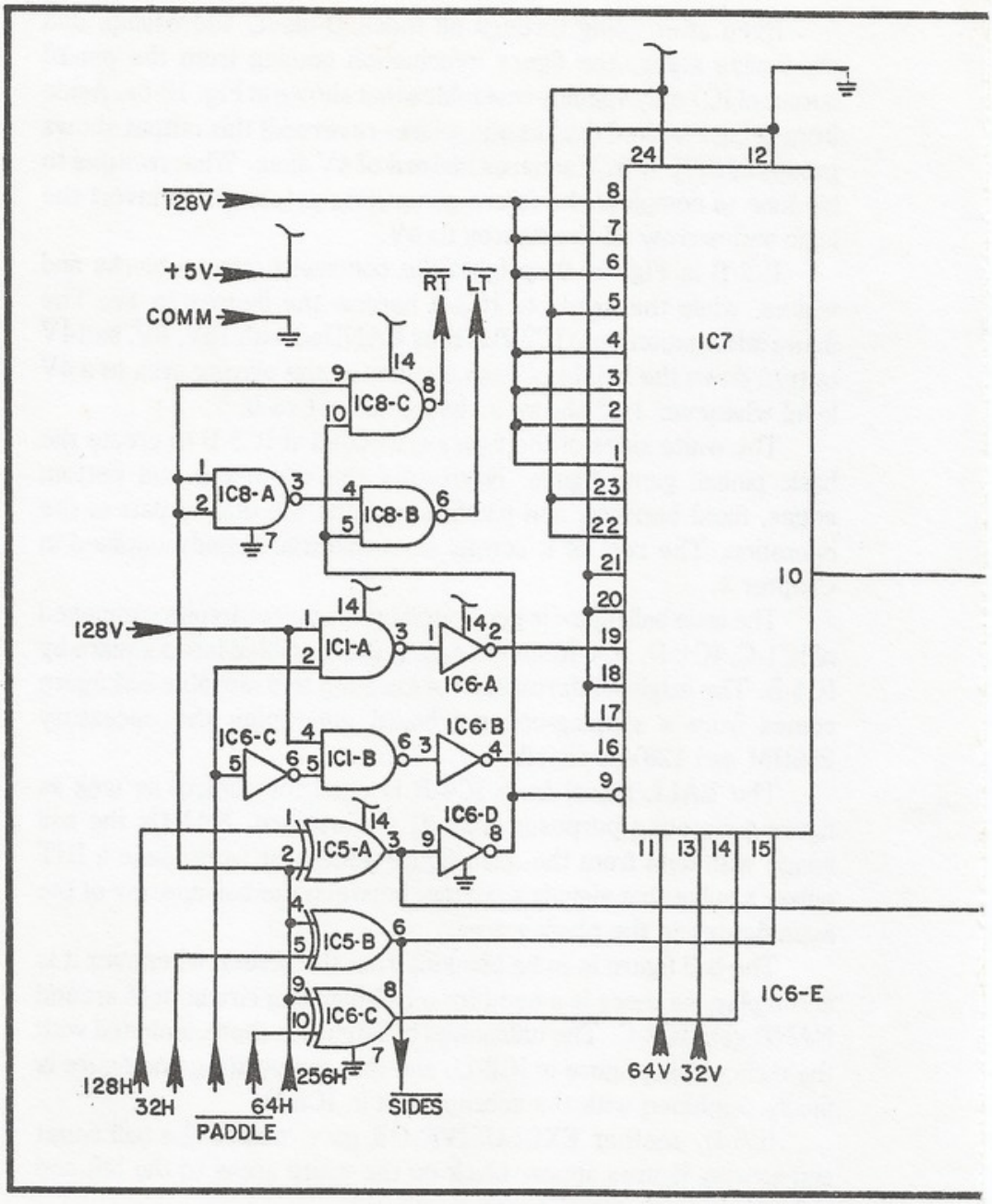
The BALL signal from IC4-B is used for control as well as figure-generating purposes. IC2-A, for instance, NANDs the ball image with data from the main figure generator to produce a HIT pulse, a pulse that signals a contact between the ball and any of the main figures in the playing area.

The ball figure is to be blanked from the screen whenever it is not in play, so there is a need for a ball-blanking circuit built around NAND gate IC2-C. The unblanked ball figure is then combined with the main playing figure in IC2-D, and this composite game figure is finally combined with the scoring data in IC5-D.

IC5-D, another EXCLUSIVE OR gate, makes the ball count and scoring figures appear black on the white areas to the left and right of the main playing area.

The purpose of the ball's control signals and the origin of the SCO (scoring data) and BBLANK (ball blanking) signals will be described later.

Before leaving this discussion of the figure board, however, we must point out the origin of the $\overline{LT}$ and $\overline{RT}$ signals from IC8-B and IC8-C respectively. These two signals represent the two sides of the main playing area, and are ultimately used for setting the horizontal-rebounding directions for the ball figure.



The left and right edges of the playing area are generated as a single unit at IC5-A and IC6-D. They must be separated for rebounding purposes, however. This is accomplished by gating IC8-C with 256H and IC8-B with a version of 256H that is first inverted by IC8-A. It makes some sense that the system is generating the left-hand side of the playing area when 256H is at logic 0 and it is generating the right-hand half when 256H = 1.

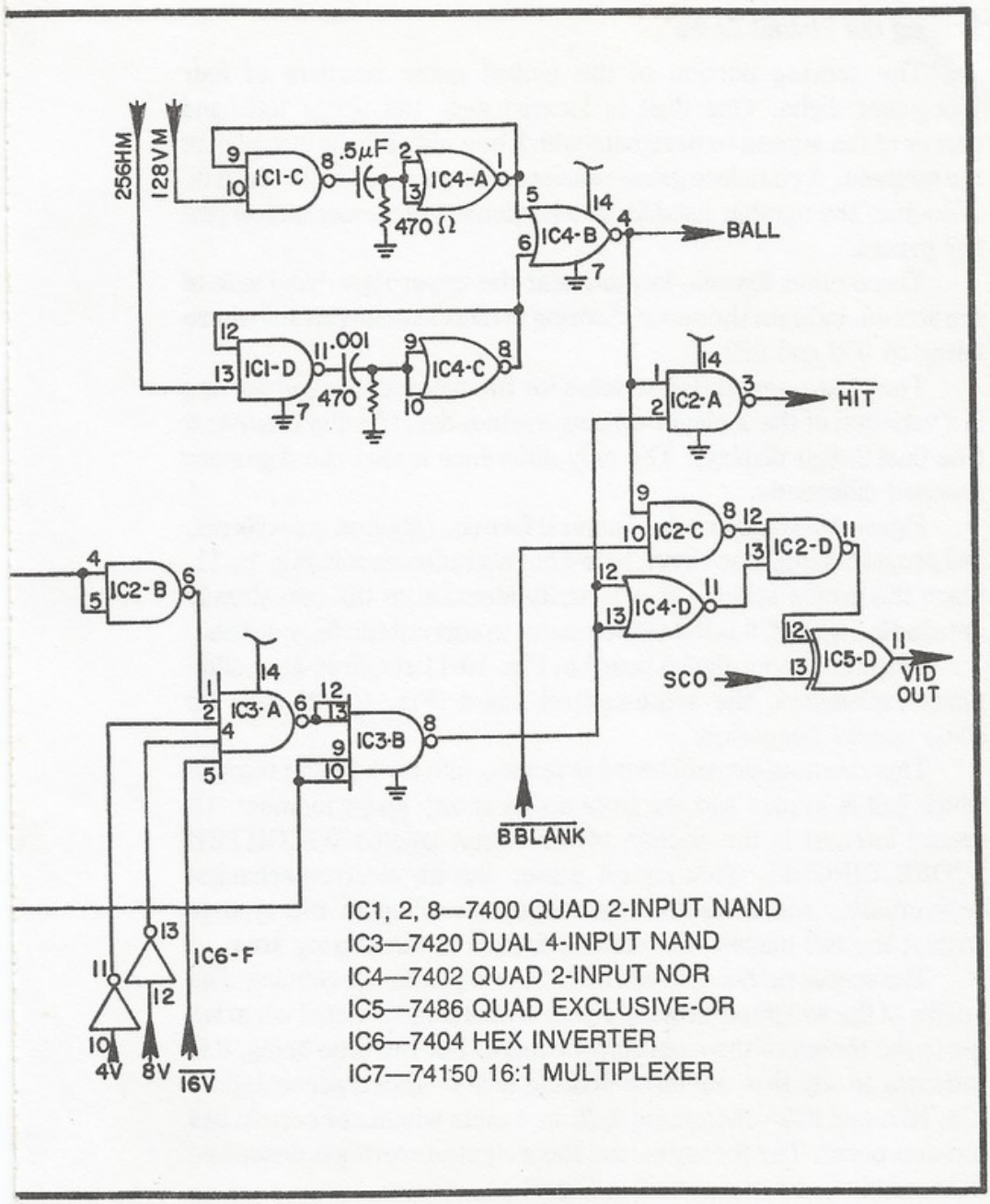


Fig. 10-9. Pinball figure board schematic diagram.

So when 256H is at logic 0, IC8-C is gated off, and the inversion of IC8-A inverts 256H to gate on IC8-B. The $\overline{LT}$ output is thus enabled in its active-low format. When 256H goes to logic 1, signifying the right-hand half of the screen is being serviced, IC8-C is enabled and IC8-B is effectively switched off.

That completes the theory of operation of the figure-generator portion of this pinball game.

Scoring the Pinball Game

The scoring portion of the pinball game consists of four 7-segment digits. One digit is located near the upper left-hand corner of the screen to designate which one of five balls is in play at the moment. A complete game consists of playing five balls which is, of course, the number used for conventional electromechanical pinball games.

Three other figures, located near the upper right-hand side of the screen, indicate the score. Scoring in this case can run anywhere between 000 and 999.

The figure-generating scheme for the ball counter and scoring is a variation of the 2-player scoring system described in Chapter 9 (the dual 2-digit display). The only difference is that the digits are grouped differently.

Figure 10-10 shows the numeral format, required waveforms, and programming. The circuit based on this rationale is in Fig. 10-11. Since this whole scheme is practically identical to the one already detailed in Fig. 9-7, it is left to the reader to sort out the finer details.

While the score-figure board in Fig. 10-11 requires little additional explanation, the score-control board (Fig. 10-12) calls for some special discussion.

This counting-control board is responsible for keeping track of which ball is in play and the total score at any given moment. Of special interest is the section of the circuit labeled WEIGHTED SCORE CIRCUIT. This pinball game, like its electromechanical counterparts, scores different amounts according to the type of contact the ball makes with various figures in the playing area.

The scores in this case can be 1, 2, or 4 points per contact. The origins of the weighted-scoring inputs is described in detail when we get to the theory of the main control board. For the time being, it is sufficient to say that the three scoring digits—those generated by IC6, IC7, and IC8—increment 1, 2, or 4 units whenever certain ball contacts occur. The theory behind the weighted scoring is described in connection with the circuit in Fig. 9-11.

IC5 in Fig. 10-12 is the ball counter. This counter is incremented by a BCOUNT pulse, which occurs each time the player depresses the BALL pushbutton. When the count reaches binary 5, IC9-A generates a logic-0 level the signals the end of the game (the $\overline{\text{END}}$ output).

The ball and score counters are all reset to zero by a pulse from IC9-D, a pulse that occurs whenever the player depresses the PLAY pushbutton.

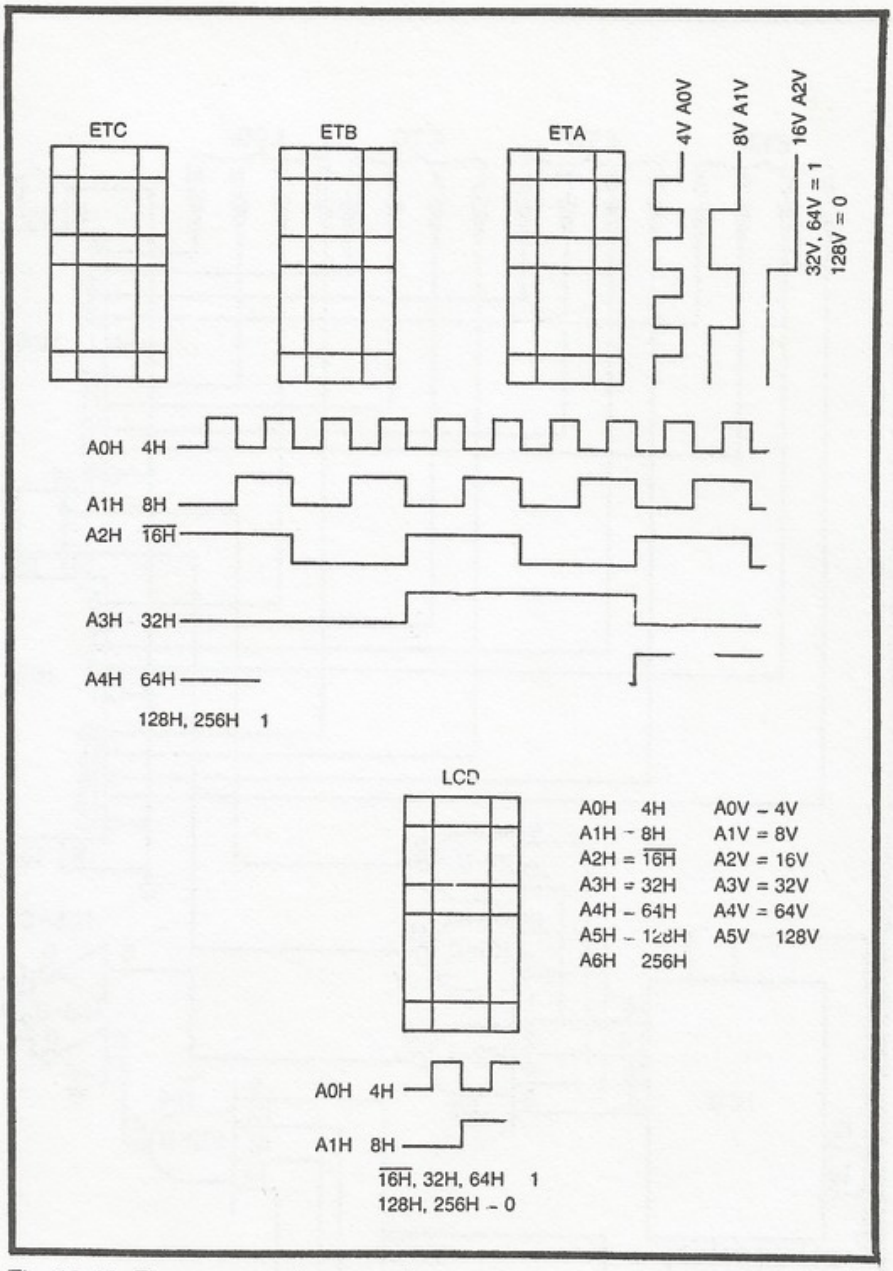
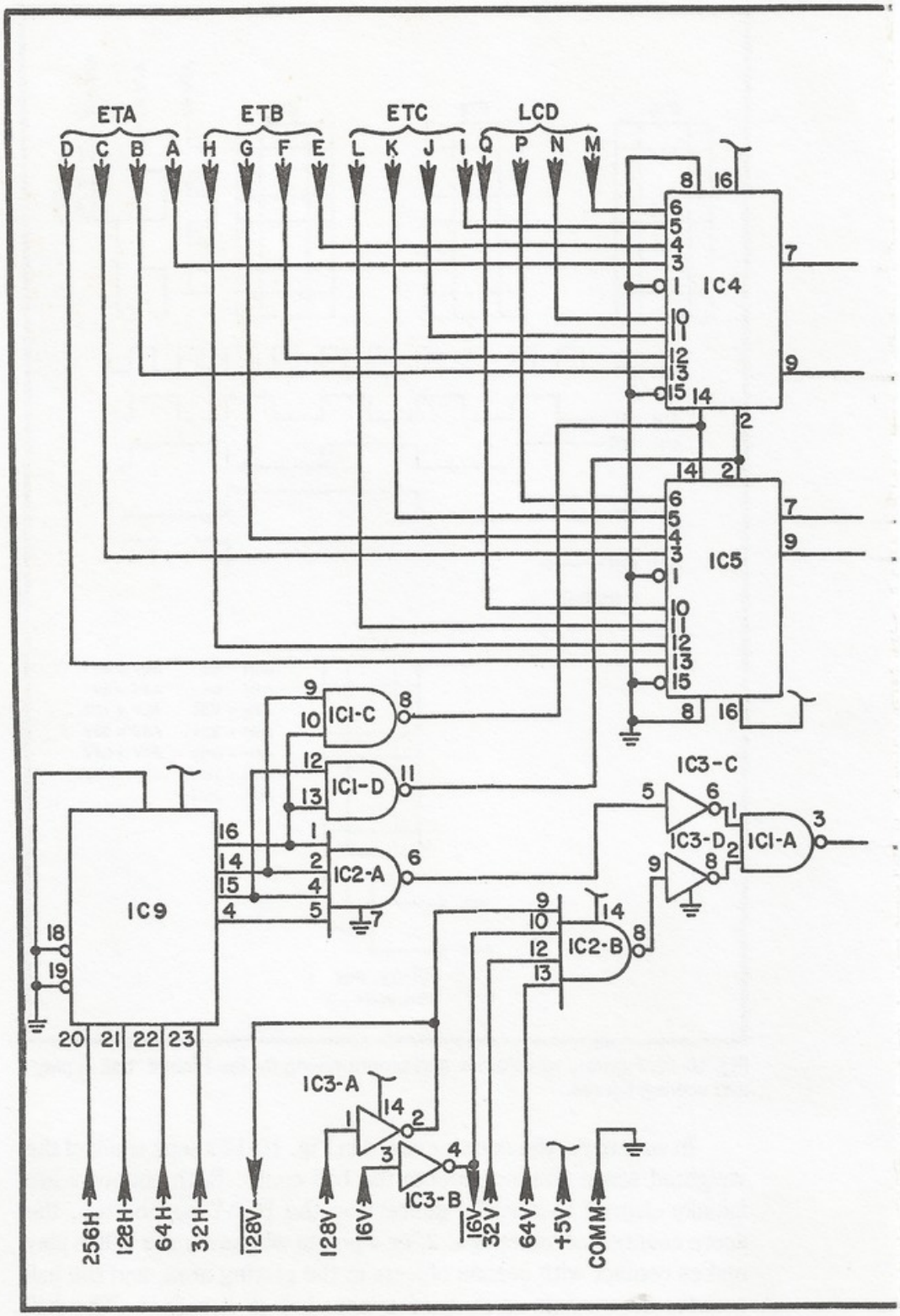


Fig. 10-10. Figures, waveforms, and programming for the Pinball "ball in play" and scoring figures.

In summary, the counter board in Fig. 10-12 keeps track of the weighted score and increments the ball count. Both counters are initially cleared to zero by depressing the PLAY pushbutton, the score counter increments 1, 2, or 4 points whenever the ball in play makes contact with certain objects in the playing area, and the ball counter increments each time a new ball is launched. The ball counter automatically stops the game when the fifth ball is played.


ETA
ETB
ETC
LCD
D C B A H G F E L K J I Q P N M
IC4
IC5
IC1-C
IC1-D
IC2-A
IC2-B
IC1-A
IC3-A
IC3-B
IC3-C
IC3-D
IC9
256H
128H
64H
32H
128V
128V
16V
16V
32V
64V
+5V
COMM

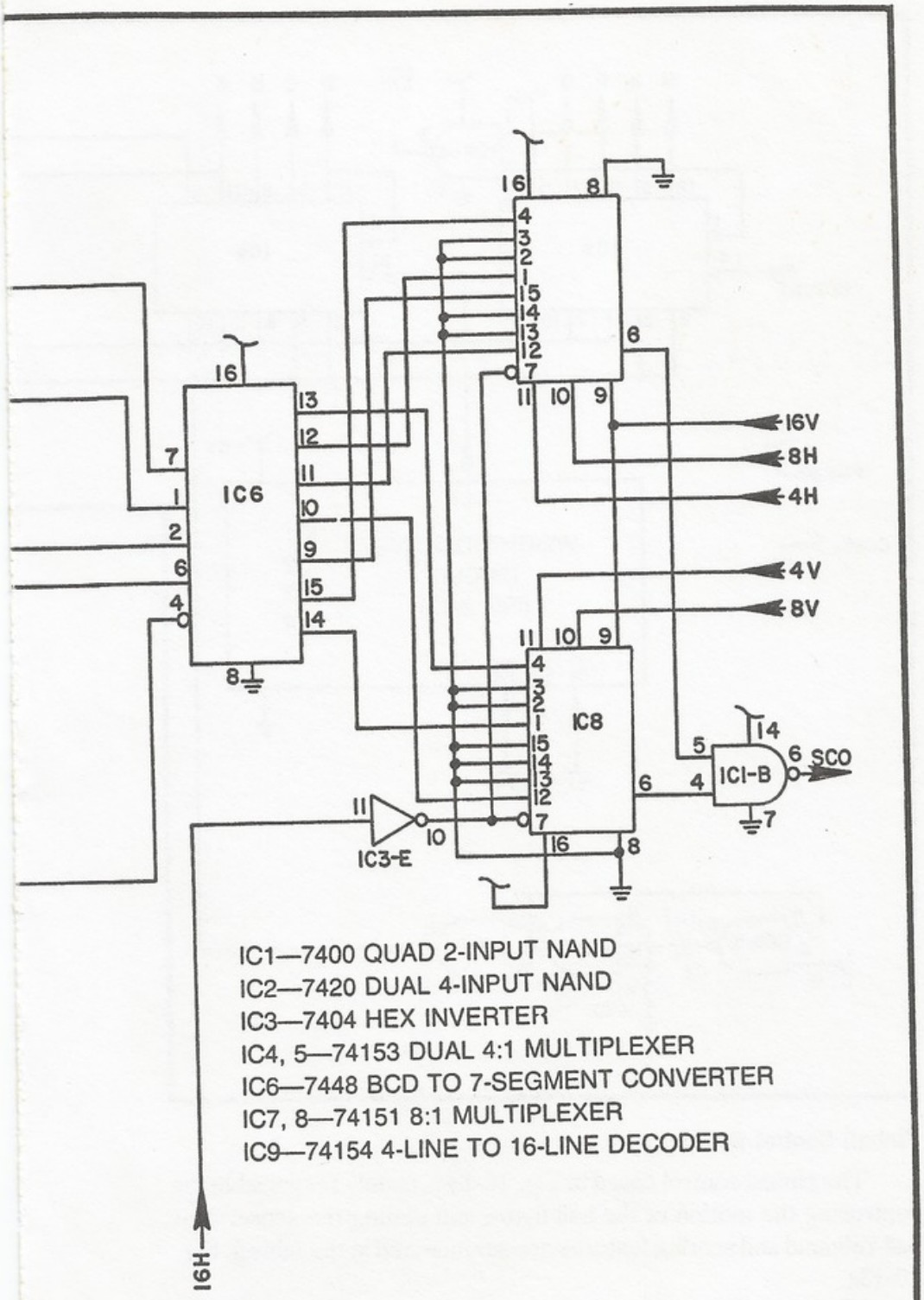
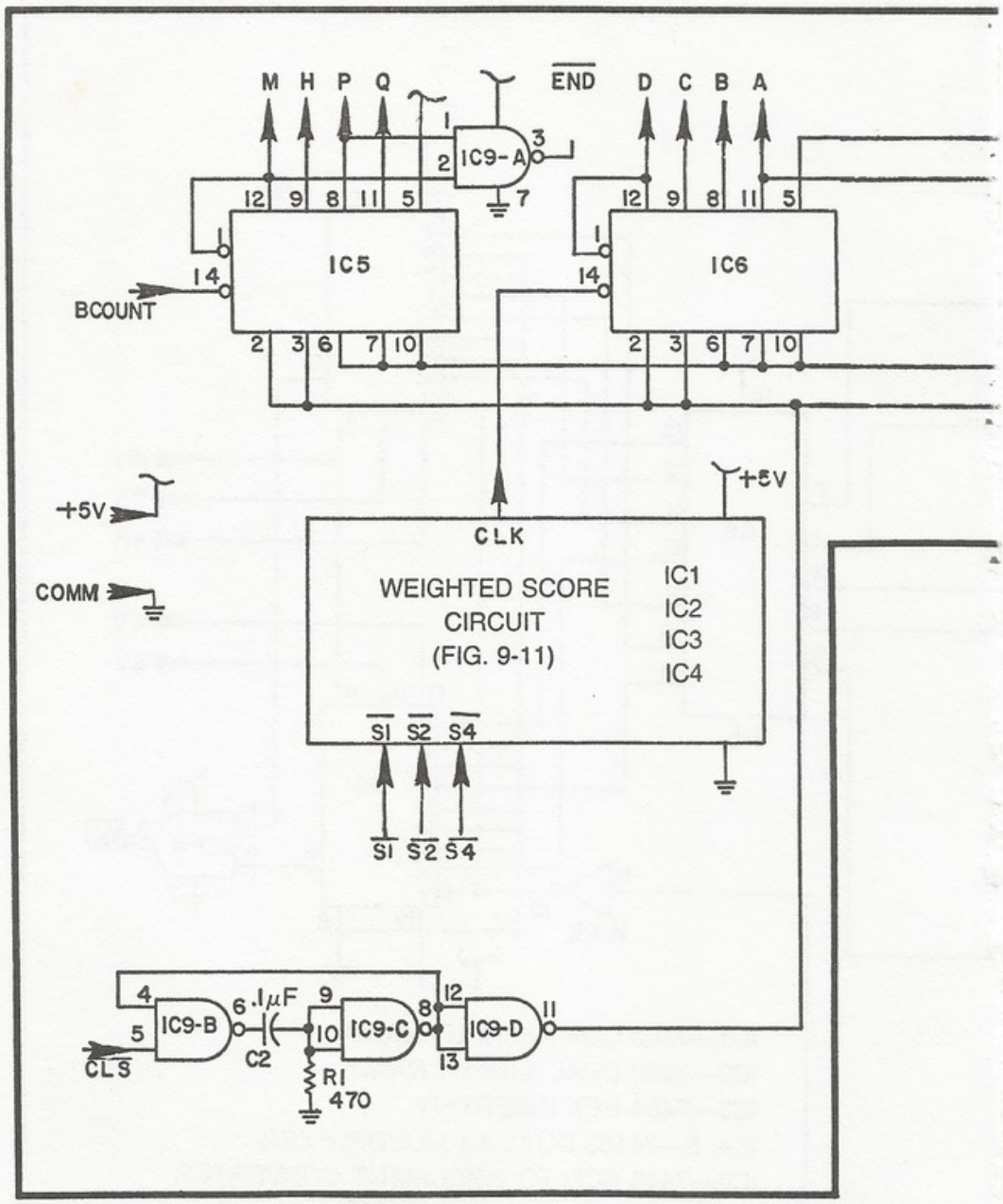


Fig. 10-11. Pinball score board schematic diagram.



Pinball Control Board

The pinball control board in Fig. 10-14 is mainly responsible for controlling the motion of the ball figure and setting the score. The ball-rebound and scoring features are summarized in the table in Fig. 10-13a.

The table shows that the vertical motion of the ball does not change at all if it is moving downward when it makes contact with TE, the top edge of the playing area. When the ball touches TE while

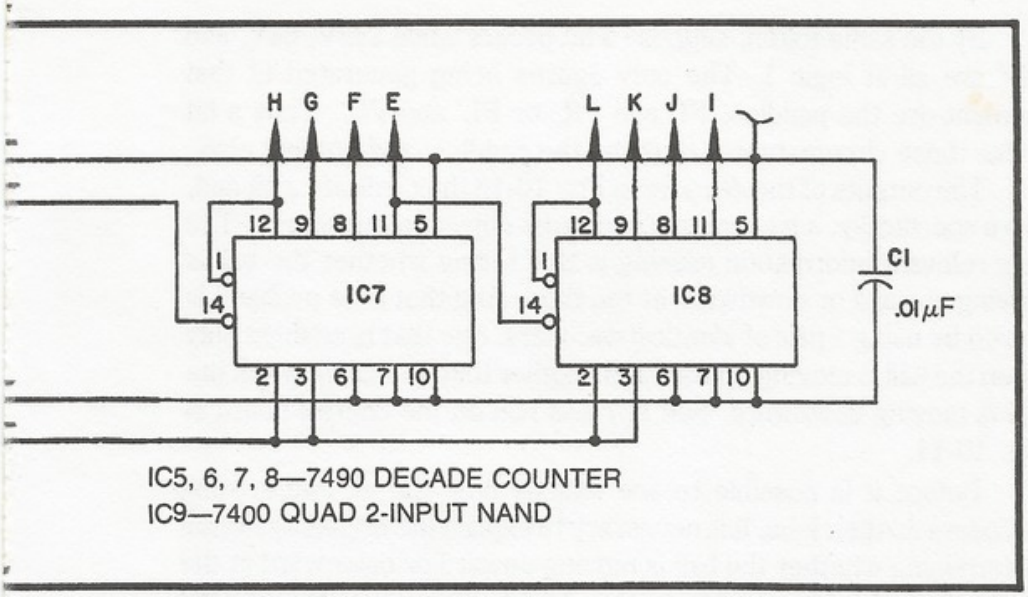


Fig. 10-12. Pinball counter board schematic diagram

moving upward, however, its direction is reversed. Neither kind of contact with TE causes a score.

The second line in the table in Fig. 10-13a shows a more interesting set of effects whenever the ball makes contact with fixed barriers A or B (designed A + B). If the ball is moving downward at the time, making contact with either A or B causes it to change to an upward direction and score 2 points. Further down the truth table, it can be seen that the ball traveling upward can contact either of the same two barriers, change its direction to down and score 4 points.

A brief study of the table in Fig. 10-13a can show the entire ball-motion and scoring rationale. The technical problem in this case is to know which barriers, edges, or paddles the ball touches. This can be done rather easily by means of the 3-line-to-8-line decoder shown in Fig. 10-13b.

Whenever a hit occurs, but not one of the two sides, this IC is enabled. (Horizontal rebounding from the left and right sides of the playing area is handled separately.)

The decoder circuit is addressed from the same three V-counts used for addressing the vertical portion of the figure matrix generator in Fig. 10-9. If a hit thus occurs when 128V, 64V, and 32V are all at logic 0, for example, the ball must be hitting the bottom edge of the playing area (BE), simply because that is the only figure being generated when 128V, 64V, and 32V are all at logic 0 at the same time.

By the same token, suppose a hit occurs while 128V, 64V, and 32V are all at logic 1. The only figures being generated at that moment are the paddles, PL and PR, or PL' and PR'. Thus a hit under those circumstances must be the paddles, and nothing else.

The outputs of the decoder in Fig. 10-13 thus indicate a hit and, more specifically, a hit against a particular object on the screen. The only relevant information missing is that telling whether the ball is moving upward or downward at the time. And that little problem is solved by using a pair of identical decoders, one that is enabled only when the ball is moving upward and another that is enabled when the ball is moving downward. See IC7 and IC8 on the control board in Fig. 10-14.

Before it is possible to see exactly how the hit and scoring decoders do their jobs, it is necessary to explain the origins of signals determining whether the ball is moving upward or downward at the time. Note the VC outputs in Fig. 10-14. These are the vertical speed and direction codes for the ball's slipping-counter circuit.

1VC and 4VC are fixed at a logic-1 level, while 2VC is fixed at logic 1. According to the vertical-speed control table in Fig. 7-22, this means the vertical section of the slipping counter will see either 0101 or 1101, where the most-significant bit (8VC) is the only one allowed to change. It turns out that the ball moves downward at a rate of about 0.64 frames per second when 8VC=0, and upward at the same rate when 8VC=1.

The 8VC output of the control board—or more specifically, the output of IC5-B—determines the ball's direction of vertical motion: 0 yields down, and 1 yields up.

Now suppose the ball happens to be moving upward. The output of IC5-B is at logic 1, and the logic-1 level is fed back to the pin-5 input of NAND gate IC4-A. This particular NAND gate responds with a logic-0 output only when three conditions are satisfied at the same time: the ball is moving downward, a hit pulse is taking place, and the hit is NOT against one of the sides of the playing area. IC8 is thus the down-motion hit detector and score decoder.

IC7, on the other hand, is the up-motion detector/decoder which is enabled only when: the ball is moving upward (a logic 1 from IC5-A), a hit occurs, and the hit is NOT against one of the two sides of the playing area. The output of IC5-A, incidentally, is always the complement of that from IC5-B; therefore, it is virtually impossible to enable IC7 and IC8 at the same time.

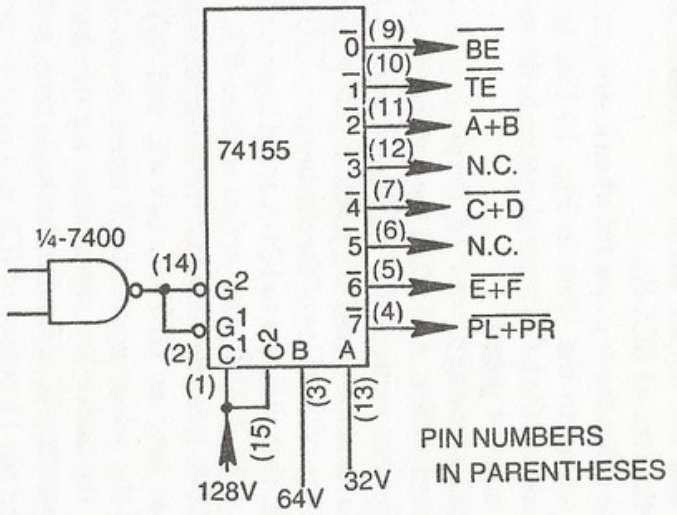
The next step in explaining the operation of this system is to work through the logic standing between the detector/decoders

DIRECTION BEFORE CONTACT	CONTACT	DIRECTION AFTER CONTACT	SCORE
DN	TE	DN	0
DN	A+B	UP	2
DN	C+D	UP	1
DN	E+F	UP	2
DN	PL+PR	UP	0
DN	PL^1+PR^1	UP	0
DN	BE	DN	0
UP	TE	DN	0
UP	A+B	DN	4
UP	C+D	DN	2
UP	E+F	DN	4
UP	PL+PR	UP	0
UP	PL^1+PR^1	UP	0
UP	BE	UP	0

HIT

$\overline{\text{SIDES}}$

A



B

Fig. 10-13. Pinball hit detector. (a) Function table. (b) Hit dejector circuit.

(IC7 and IC8) and the output circuit that determines the ball's direction of vertical travel (IC5-B).

Suppose the ball is moving upward when it strikes the bottom of barrier A. According to the table in Fig. 10-13a, this particular situation should switch the ball's vertical motion to downward and, at the same time, score 4 points.

When the ball is moving downward, the pin-5 input of IC4-A is at logic 1 as explained earlier. And when the ball hits the A barrier (or any other one for that matter), the HIT input goes to logic 0, and inverter IC9-A inverts the signal to a positive-going pulse. Since the hit is not against one of the sides of the playing area, $\overline{\text{SIDES}}$ is at logic 1, and it turns out that all inputs to IC4-A are at logic 1 as long as the $\overline{\text{HIT}}$ pulse lasts. This NAND-gate action enables IC8.

From Fig. 10-8, it can be seen that the A barrier is being drawn on the screen as long as 128V=0, 64V=1, and 32V=0. These V-count lines to the select inputs of IC8 cause its pin-11 output to drop to logic 0, the active-low signal state for the decoders. The pin-5 output of that same IC is fixed at that same time, so NAND gate IC2-D passes the pin-11 output of IC8 as a positive-going pulse.

This pulse is inverted again by inverter IC9-F, and then goes to the $\overline{\text{S4}}$ connection on the weighted-scoring circuit. This accounts for an additional 4 points on the score readout.

That same negative-going pulse from IC9-F also passes through NAND gate IC3-D, emerging as a positive-going pulse to the pin-6 input of IC5-B. IC5-B is one-half of an R-S flip-flop circuit that also includes IC5-A. The positive pulse at IC5-B in this case resets the flip-flop so that two things happen simultaneously: the output of IC5-B is switched to logic 0, thereby reversing the direction of motion of the ball, and the output of IC5-B is switched to logic 0, thereby reversing the direction of motion of the ball, and the output of IC5-A, is switched to logic 1 to enable the downward-motion decoder, IC7.

Recall that this entire sequence of activity began when an upward-moving ball hit barrier A. The final results of this action is 4 additional points in the score and changing the direction of ball motion from up to down.

A similar kind of analysis shows that the decoders and logic circuits perform all the direction-changing and scoring operations specified in the table on Fig. 10-13a.

All the operations described for the control board to this point have concerned the VC outputs and changes in the vertical direction. It can be seen that changes in the horizontal direction of ball motion is a bit more straightforward.

The outputs controlling the ball's horizontal motion are 1HC through 8HC in Fig. 10-14. 1HC and 4HC are fixed at logic 1, while 2HC is fixed at logic 0. Only the 8HC bit is concerned with the ball's changes in horizontal motion.

The horizontal motion codes to the horizontal portion of the slipping-counter board are thus 0101 or 1101, depending on whether the ball is to be moving to the right or left respectively. In either case, the horizontal velocity is 0.98 screens per second, and whether the ball is moving to the right or left is determined by the status of an R-S flip-flop composed of IC5-C and IC5-D.

The inputs to this little horizontal-direction control circuit are BALL (the ball figure), $\overline{LT}$, and $\overline{RT}$ (signals indicating the two sides of the playing area.

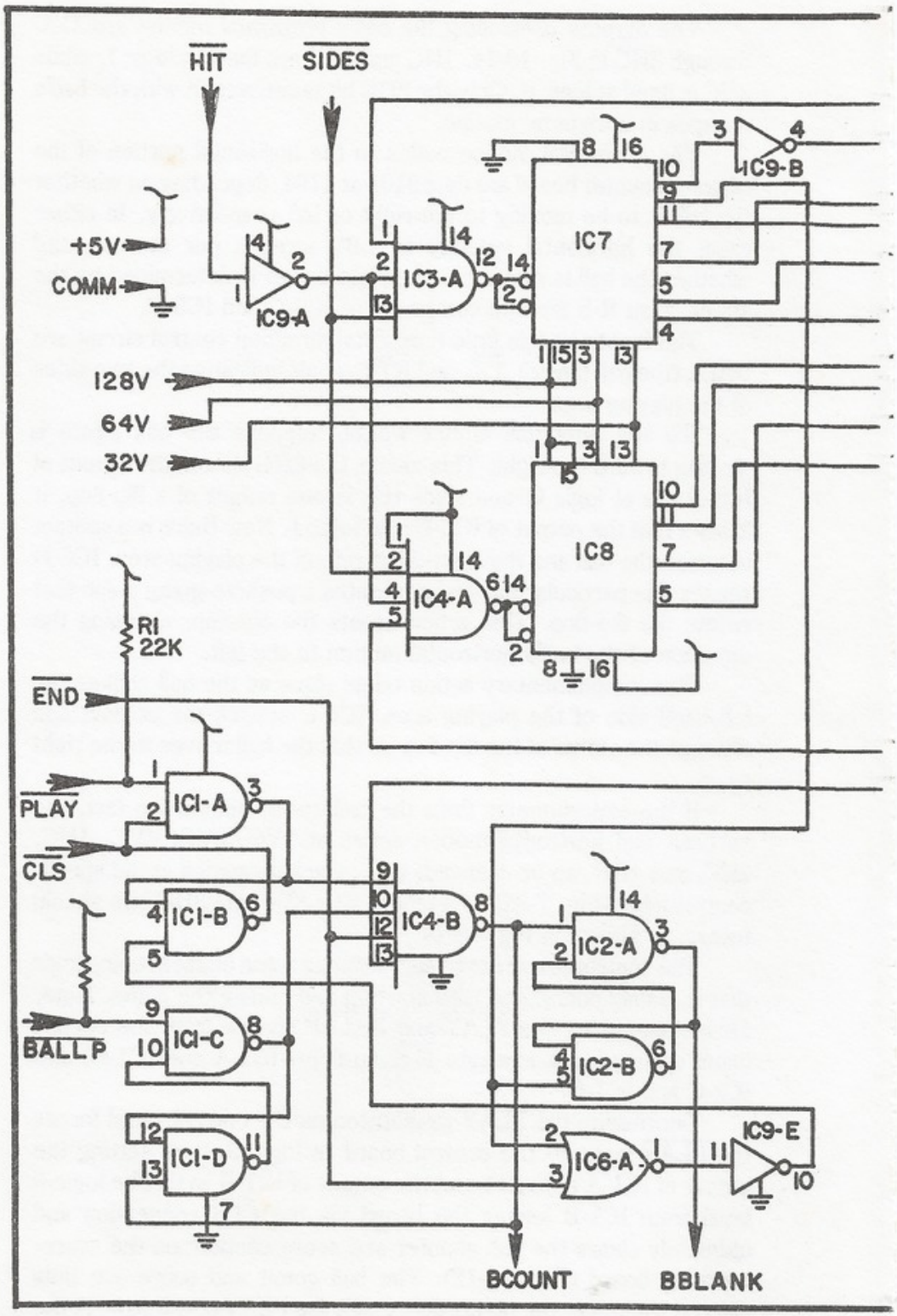
To see how this circuit works, suppose the ball figure is moving toward the right. This means the 8HC bit and the output of IC5-D are at logic 0; and since this is one output of a flip-flop, it follows that the output of IC5-C is at logic 1. Now there is a contact between the ball and the right-hand side of the playing area. IC6-D senses this particular hit, and generates a positive-going pulse that resets the flip-flop. This action resets the flip-flop, changing the direction of the ball's horizontal motion to the left.

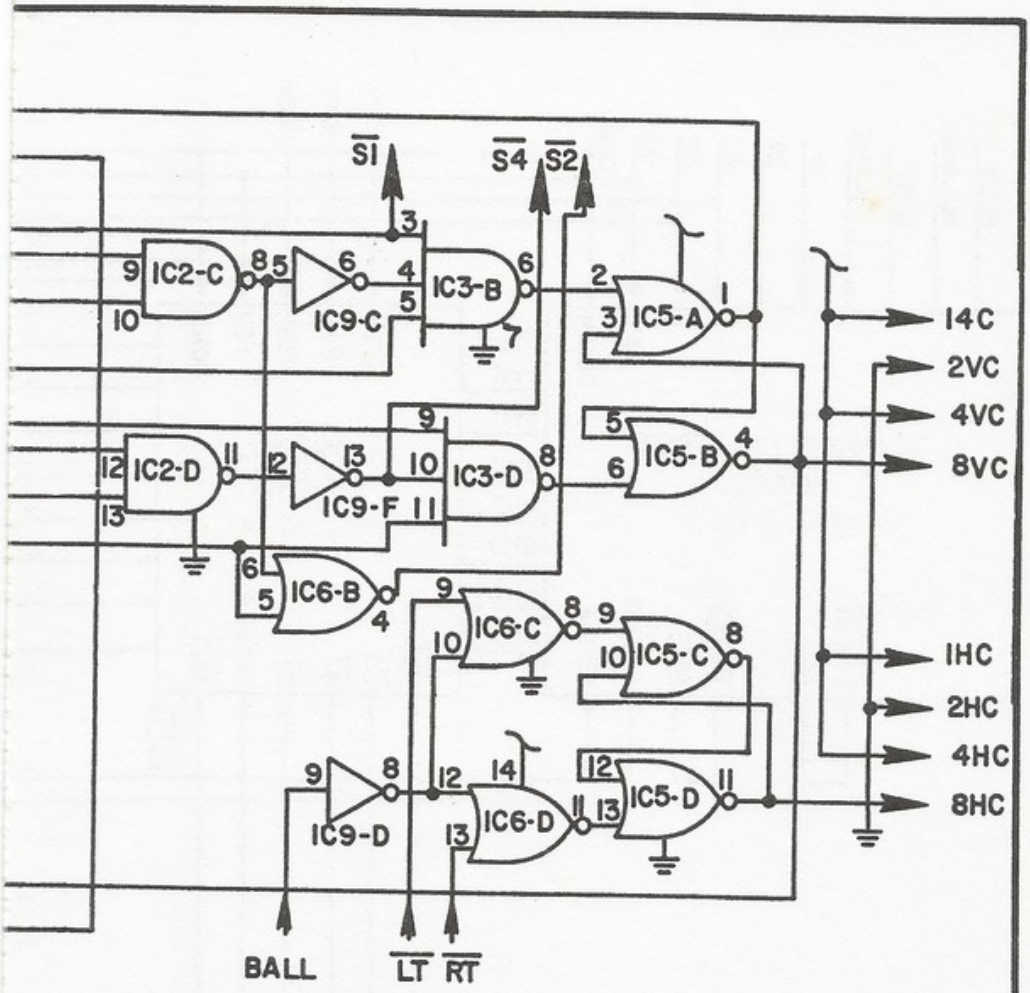
The complementary action takes place as the ball strikes the left-hand side of the playing area. IC6-C senses the contact and changes the status of the flip-flop so that the ball moves to the right again.

If the experimenter finds the ball speed seems too fast, the vertical- and horizontal-motion codes at 1VC, 2VC, 4VC, 1HC, 2HC, and 4HC can be changed, using the information in the speed-control table (Fig. 7-22) as a guide. The 8VC and 8HC bits should remain as shown in Fig. 10-14.

The control circuit board also includes some housekeeping logic that is mainly concerned with starting and ending the game. Note, for instance, that the $\overline{PLAY}$ and $\overline{BALLP}$ signals from the control panel each go to a separate $\overline{R}$-$\overline{S}$ lip-flop—IC1-A and IC1-B, and IC1-C and IC1-D.

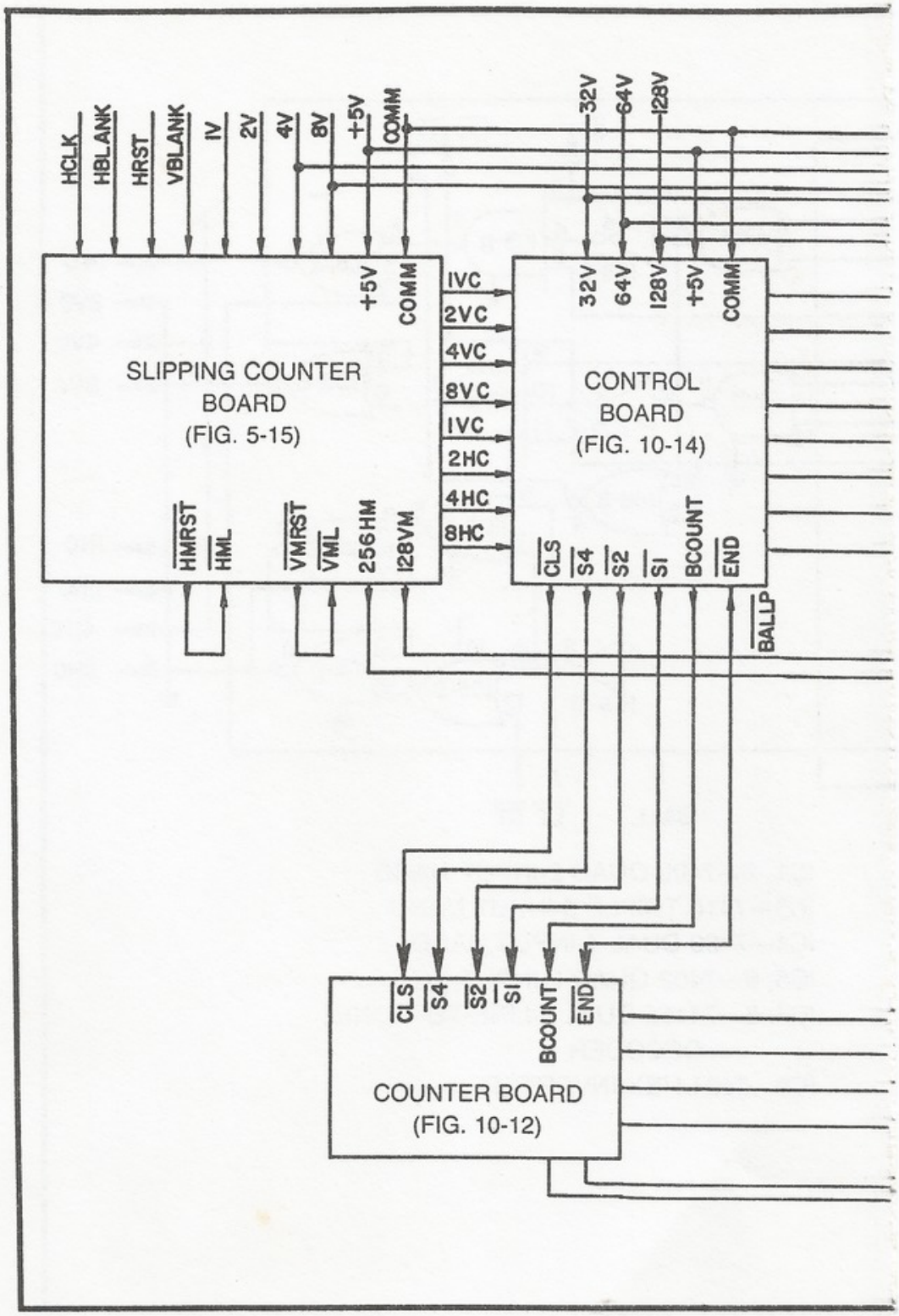
Depressing the PLAY pushbutton on the control panel forces the $\overline{PLAY}$ input to the control board to logic 0, thus setting the output of IC1-A to logic 1 and the output of IC1-B to 0. The logic-0 level from IC1-B leaves the board via the $\overline{CLS}$ connection and ultimately clears the ball counter and score counter on the score-counting board (Fig. 10-12). The ball count and score are thus cleared whenever the player depresses the PLAY pushbutton at the beginning of a new game sequence.

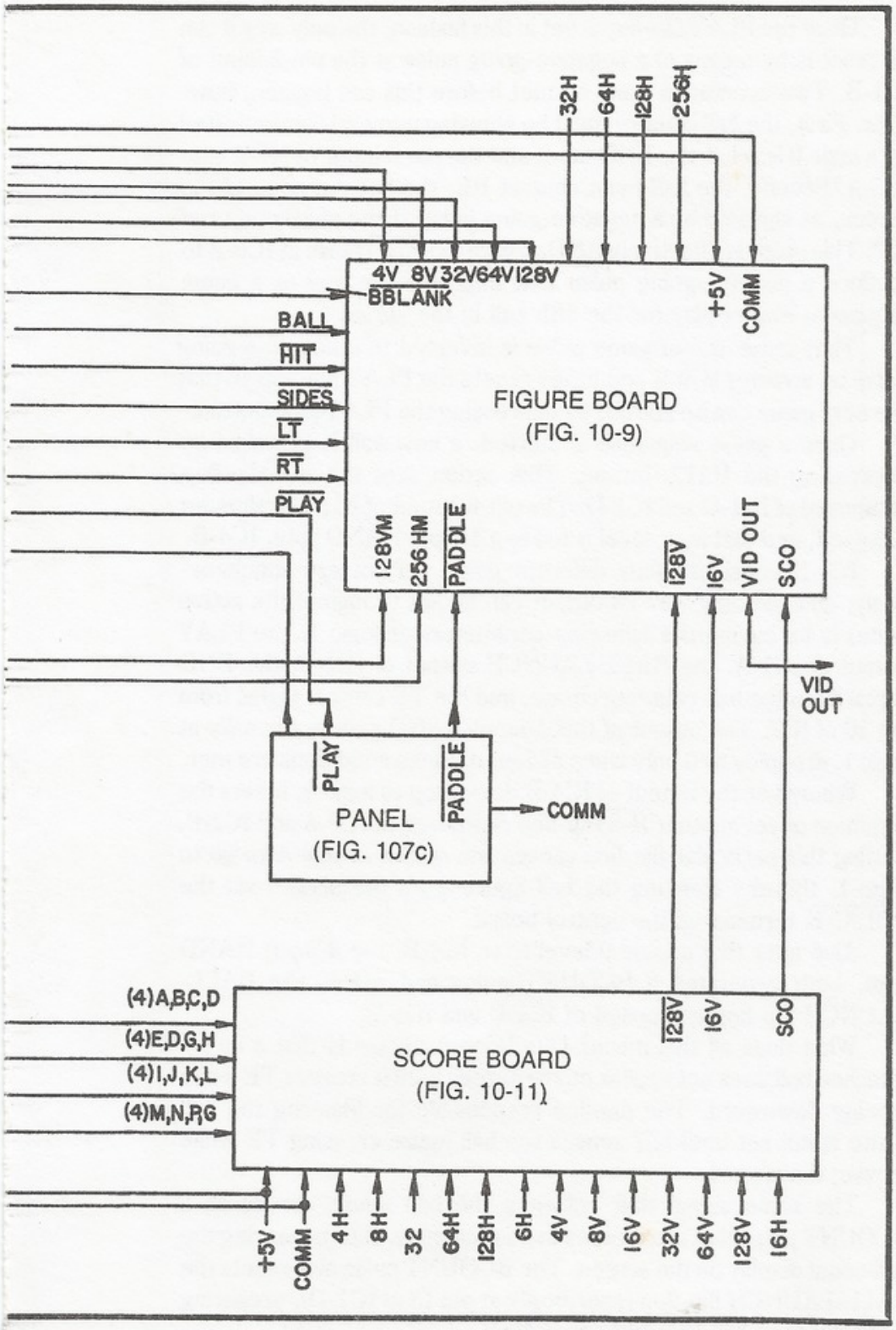

HIT
SIDES
+5V
COMM
IC9-A
IC3-A
IC7
IC9-B
128V
64V
32V
IC8
IC4-A
R1
22K
END
PLAY
IC1-A
CLS
IC1-B
IC4-B
IC2-A
BALLP
IC1-C
IC2-B
IC1-D
IC6-A
IC9-E
BCOUNT
BBLANK



IC1, 2—7400 QUAD 2-INPUT NAND
IC3—7410 TRIPLE 3-INPUT NAND
IC4—7420 DUAL 4-INPUT NAND
IC5, 6—7402 QUAD 2-INPUT NOR
IC7, 8—74155 DUAL 2-LINE-TO-4-LINE DECODER
IC9—7404 HEX INVERTER

Fig. 10-14. Pinball control board schematic diagram.


HCLK
HBLANK
HRST
VBLANK
1V
2V
4V
8V
+5V
COMM
32V
64V
128V
SLIPPING COUNTER BOARD (FIG. 5-15)
CONTROL BOARD (FIG. 10-14)
1VC
2VC
4VC
8VC
1VC
2HC
4HC
8HC
HMRST
HML
VMRST
VML
256HM
128VM
CLS
S4
S2
S1
BCOUNT
END
BALLP
COUNTER BOARD (FIG. 10-12)



F 10-15. Pinball wiring block diagram.

Once the PLAY flip-flop is set in this fashion, the only way it can be reset is by means of a negative-going pulse at the pin-5 input of IC1-B. Two conditions must be met before this can happen, however. First, the ball counter must be showing numeral 5 as indicated by a logic 0 level at the END input and the pin-3 input of NOR gate IC6-A. Second, the ball must contact BE, the bottom edge of the screen, as signaled by a negative-going pulse at the pin-9 output of IC7. This pulse is effectively ANDed with the $\overline{\text{END}}$ pulse at IC6-A to produce a positive-going pulse that indicates the end of a game sequence: end of play for the fifth ball in the series.

That same end-of-game pulse is inverted to a negative-going pulse by inverter IC9-E and finally resets the PLAY flip-flop so that the next game can be started by depressing the PLAY pushbutton.

Once a game sequence is started, a new ball is launched by depressing the BALL button. This action sets the $\overline{\text{R-S}}$ flip-flop composed of IC1-C and IC1-D. The pin-8 output of IC1-C is thus set to logic 1, and that logic level is fed to a 4-input NAND gate, IC4-B.

IC4-B considers four different game parameters simultaneously, and the only way its output can be set to logic 0 (its active state) is by having the following parameters at logic 1: the PLAY output of IC1-A, the BALL-LAUNCH output of IC1-C, the END signal from the ball-counting circuit, and the TE contact signal from pin 10 of IC7. The output of this 4-input NAND gate is normally at logic 1, dropping to 0 only when all four of these conditions are met.

Whenever the output of IC4-B does drop to logic 0, it sets the condition of yet another $\overline{\text{R-S}}$ flip-flop composed of IC2-A and IC2-B. Setting this particular flip-flop causes the output of IC2-A to go to logic 1, thereby blanking the ball figure from the screen via the $\overline{\text{BBLANK}}$ terminal of the control board.

Also note that a logic-0 level from IC4-B, the 4-input NAND gate, both generates a BCOUNT pulse and resets the BALL-LAUNCH flip-flop composed of IC1-C and IC1-D.

What does all this mean? One thing it means is that a newly launched ball does not appear on the screen until it crosses TE while moving downward. The flip-flop responsible for blanking the ball figure is not set until IC7 senses the ball figure crossing TE while moving downward.

The same signal that unblanks the ball figure serves as a BCOUNT pulse that increments the ball counter, thus advancing the ball-count display on the screen. The BCOUNT pulse also resets the BALL-LAUNCH flip-flop (specifically at pin 13 of IC1-D), preparing it for the next ball-launching operation.

The ball-blanking flip-flop is reset only when the ball crosses BE (the bottom edge of the playing area) while traveling downward. This effect is detected by the pin-9 output of IC7, the same one responsible for resetting the entire game at the end of play for the fifth ball.

Overall Pinball Block Diagram

A final wiring diagram for the pinball game appears in Fig. 10-15. You will note that the game requires five circuit boards and a special control panel. Of course you should already have the slipping-counter board available from previous experiments and games.

The slipping counter and control boards can be powered from one of the +5V sources, while the remaining boards ought to be connected to a second source to avoid overloading any one of the power supplies.