United States Patent [19]

Miffitt et al.

[11] Patent Number:

4,957,291

[45] Date of Patent:

Sep. 18, 1990

[54]	ELECTRO	NIC PUZZLE			
[75]	Inventors:	Donald C. Miffitt, Chelmsford; Angelo Tortola, Lexington; Charles S. Sebor, Reading; Robert L. Halliday, Lexington, all of Mass.			
[73]	Assignee:	Venture Technologies, Inc., North Billerica, Mass.			
[21]	Appl. No.:	433,389			
[22]	Filed:	Nov. 6, 1989			
Related U.S. Application Data					
[63]	Continuatio doned.	n of Ser. No. 166,927, Mar. 11, 1988, aban-			
[51] [52] [58]	U.S. Cl				
[56]		References Cited			
U.S. PATENT DOCUMENTS					
	4,320,901 3/1	982 Hanson et al 273/153			

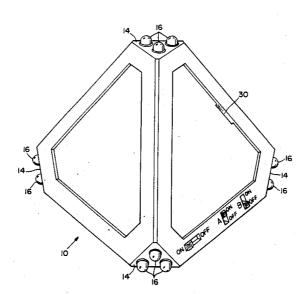
4,513,973	4/1985	Sinclair 273/237
4,575,087	3/1986	Sinclair 273/153
4,809,979	3/1989	Skowronski et al 273/153 R

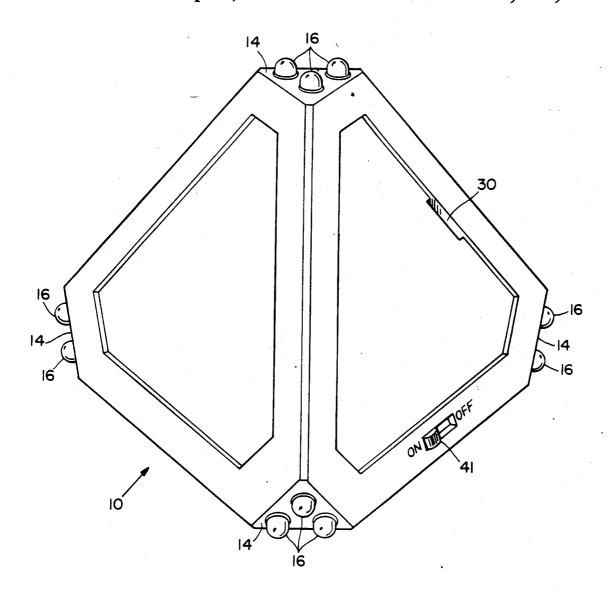
Primary Examiner—Edward M. Coven Assistant Examiner—Jessica J. Harrison Attorney, Agent, or Firm—Choate, Hall & Stewart

[57] ABSTRACT

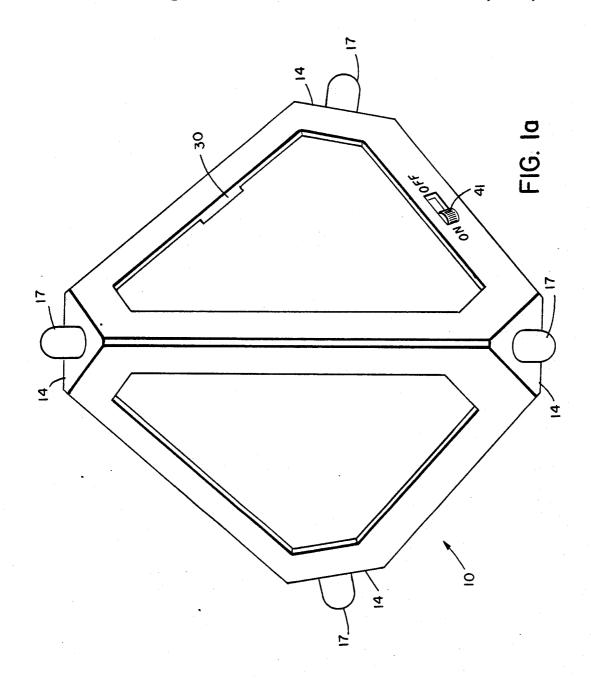
An electronic game that includes a main body that consists of four panels assembled to form a four-sided pyramid having four vertexes. Each vertex serves as a node for supporting a light which may illuminate a plurality of different colors. Digital circuitry defines a color state vector at each node by illuminating one or none of the colors. Control circuitry connected to the digital circuitry is used for altering the color state of at least one of the lights as the tetrahedron is rotated. In one game, the color of the node brought to the top position changes in the color change pattern established by the color state vector. A player rotates the tetrahedron in an effort to figure out the pattern of color changes. Once the pattern is deciphered, the player makes additional rotations to achieve an object of the game such as having each vertex of the tetrahedron lighted the same color.

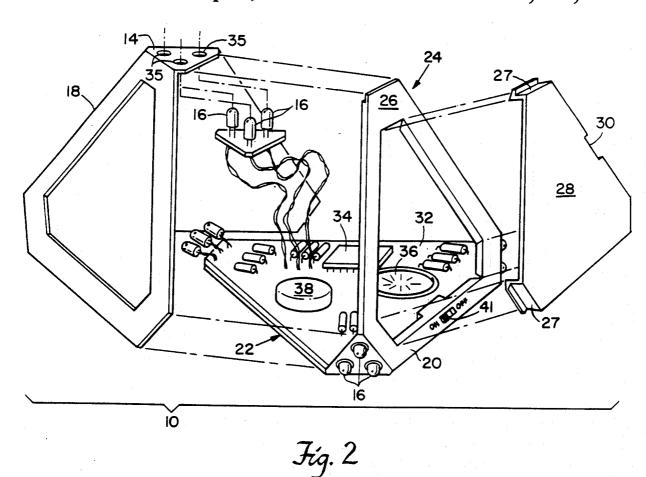
14 Claims, 8 Drawing Sheets

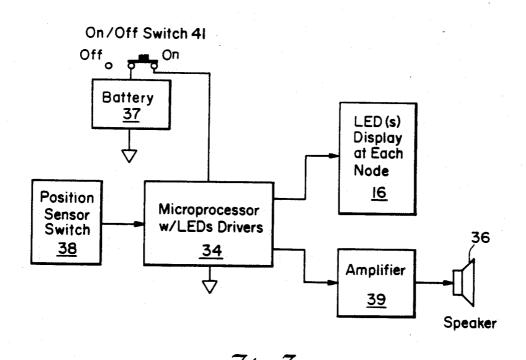


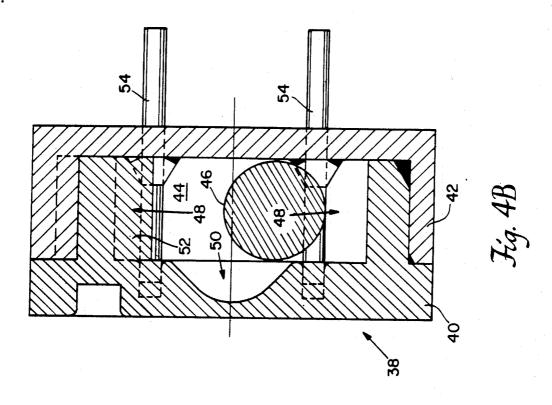


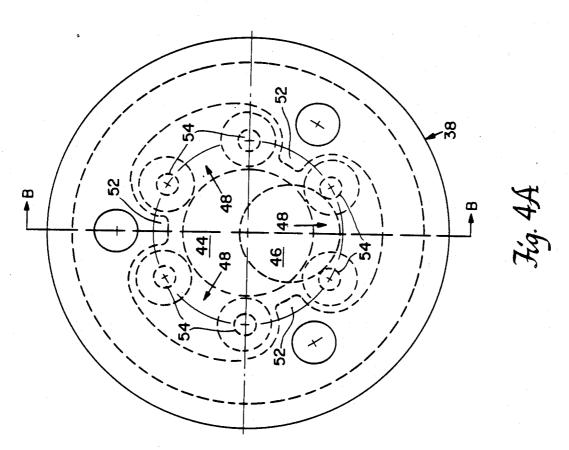
Fíg. 1



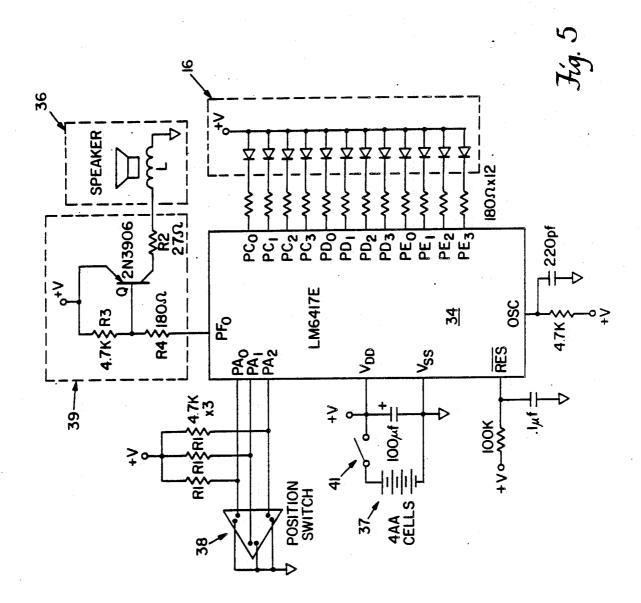


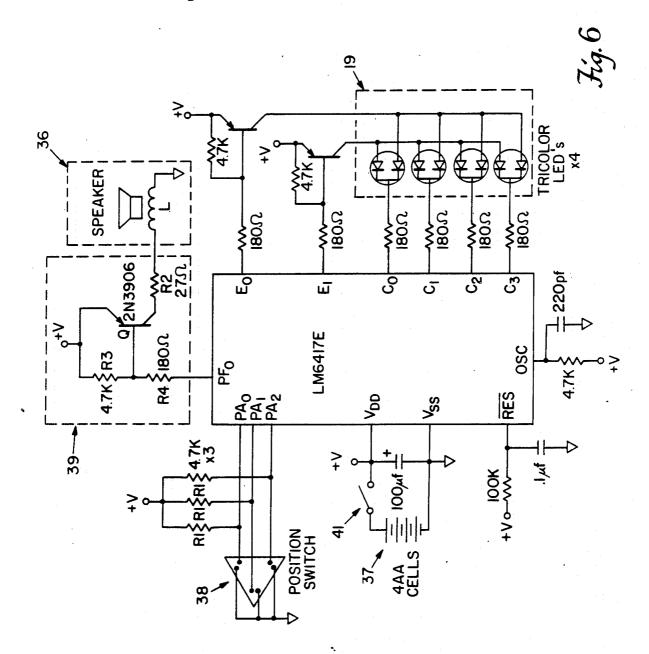


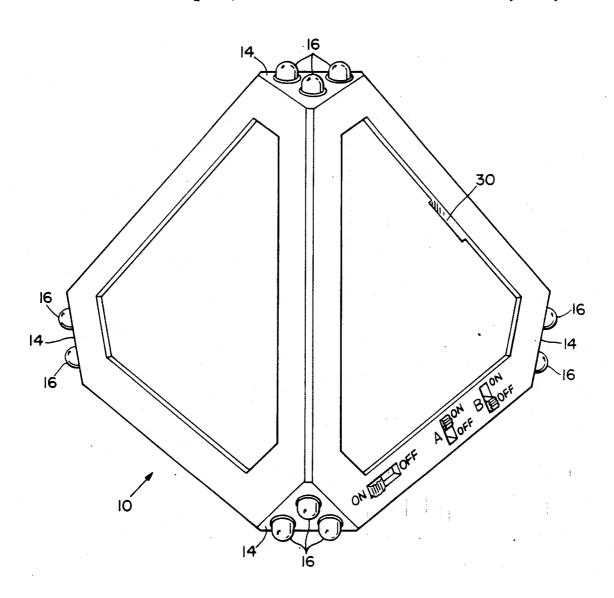












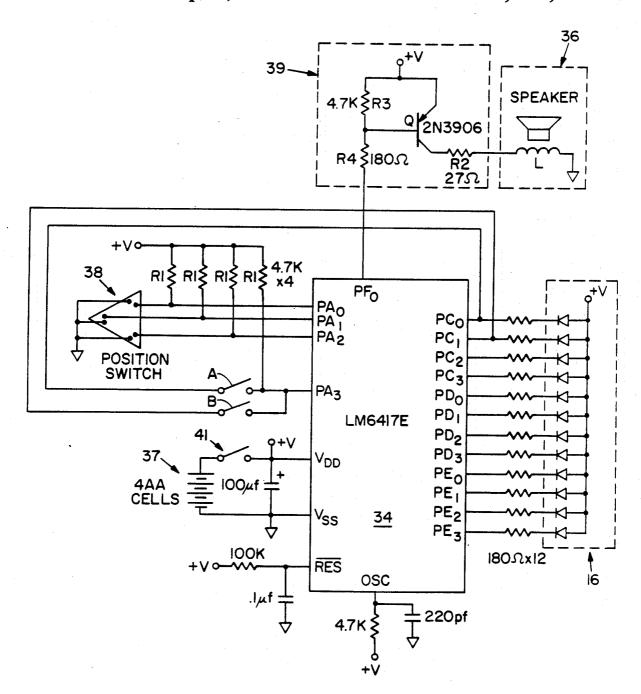


Fig. 8

ELECTRONIC PUZZLE

This is a continuation of copending application(s) Ser. No. 07/166,927 filed on Mar. 11, 1988 now abandoned. 5

BACKGROUND OF THE INVENTION

This invention relates to an electronic puzzle or

U.S. Pat. No. 4,575,087 discloses an electronic puzzle 10 configured as a cube. The puzzle stores a fixed, predetermined sequence of orientation changes which will result in all of the faces of the cube being illuminated. Only if a player makes the predetermined sequence of orientation changes in the correct order will the faces 15 ing the puzzle. Thus, a virtually infinite set of orientabe illuminated. In this patent, each face of the cube or other polyhedron can be in only one of two possible states such as ON or OFF.

Other electronic games are known which generally include a keyboard through which a player interacts 20 with the game. See, for example, U.S. Pat. Nos. 4,513,973; 4,240,638; 4,320,901 and 4,323,243. The puzzle such as that disclosed in U.S. Pat. No. 4,575,087 discussed above requires both three-dimensional orientation skills and memory skills to memorize sequences 25 of movements to be performed in order to achieve a solution.

SUMMARY OF THE INVENTION

The electronic puzzle according to the invention 30 zle having a single light at each vertex. includes a puzzle body having a plurality of lightable positions, each lightable position capable of being illuminated in one of a plurality of colors. Digital circuitry is provided to establish a color state vector defining a pattern of color changes for each lightable position. 35 Control circuitry interconnected with the digital circuitry changes the color of at least one of the lightable positions to the next color in the color state vector in response to a player's manipulation of the puzzle body. One object of the puzzle is to have each lightable posi- 40 puzzle. tion illuminated the same color.

Preferred embodiments of the invention include the following features. The main body consists of four panels assembled to form a tetrahedron. A light is supported at each of its four vertexes. Each light may con- 45 the number of games that may be played. sist of a group of differently colored lights, which may include a red, green, and yellow LED, for example, or a single light capable of shining in several colors. Each light is illuminated according to a color state vector stored in a microprocessor. The control circuitry in- 50 cludes a position sensor switch for informing the digital circuitry of the current position of the tetrahedron. The position sensor switch consists of a housing having a cavity that defines four positions, each corresponding to one of the vertexes. Conducting pins at three of the four 55 positions are used to inform the digital circuitry of the location of a ball that is free to roll to any of the positions within the cavity. The position sensor switch is also used to select different games stored in the microprocessor.

As the tetrahedron is rotated to bring a vertex to an upright position, a different color (or off) is illuminated. One of the games, which may be stored in the microprocessor, has as its object the lighting of the same color at each node. Many other games may be stored in the 65 microprocessor, as will be discussed to provide nearly limitless play. The puzzle utilizes flashing colored lights that will provide visual entertainment, especially in a

darkened room. The pyramid shape itself is another appealing feature. The electronic puzzle is also a very inexpensive product to manufacture.

According to the invention, each time a vertex is rotated to an upright position, the color of that vertex changes to the next color in the color state stored in a microprocessor. A player rotates the tetrahedron in an experimental fashion to try to figure out the pattern of color changes. Once the pattern is deciphered, the player continues manipulating the tetrahedron in an effort to achieve a solution such as having each vertex illuminated the same color. Successive plays by the same or different players will likely result in different patterns of rotations, all such patterns resulting in solvtion changes will solve the puzzle according to this invention. This is unlike the puzzle of U.S. Pat. No. 4,575,087 in which a predetermined sequence of orientations is required to solve the puzzle. The puzzle of the invention thus provides a much richer universe of situations resulting in a much more interesting puzzle than known in the prior art.

BRIEF DESCRIPTION OF THE DRAWINGS

The figures are briefly described as follows.

FIG. 1 is a perspective view of the electronic puzzle of the invention having different colored LEDs at each of its four vertexes or nodes.

FIG. 1A is a perspective view of the electronic puz-

FIG. 2 is an exploded perspective view of the electronic puzzle of FIG. 1.

FIG. 3 is a block diagram of the circuit driving the LED displays at each node.

FIG. 4A is a top view of a positional switch used to indicate which of the four nodes is in the top position.

FIG. 4B is a side sectional view along the line B-B of the position switch of FIG. 4A.

FIG. 5 is an electronic diagram of the electronic

FIG. 6 is an alternate electronic diagram of the electronic puzzle.

FIG. 7 is a perspective view of an alternate electronic puzzle having two additional switches for increasing

FIG. 8 is an electronic diagram of the electronic puzzle of FIG. 7.

DESCRIPTION OF THE PREFERRED **EMBODIMENT**

Referring to FIG. 1, an electronic puzzle in the form of a tetrahedron 10 is shown. Tetrahedron 10 has four vertexes, each serving as a node or lightable position 14 for positioning three different colored LEDs 6. These LEDs 16 are preferably red, green and yellow, but any color may be used. Each node 14 is numbered 1 to 4 (not shown) to allow a player to distinguish one from another. FIG. 1A shows an embodiment of the invention including a single light 17 at each of the nodes 14. Each of the lights 17 is capable of shining in multiple colors such as red, green, and yellow. The lights 17 may be LEDs including red and green elements disposed behind a common clear lens. Red and green colors are achieved by activating either the red or green dye portions of the light and a yellow-orange color is achieved when both red and green elements are activated.

Shown in FIG. 2, the tetrahedron 10 consists of two case halves 18, 20 that are made from rigid durable 3

material such as plastic. Lower case half 20 includes a bottom panel 22 and a side panel 24. Bottom panel 22 supports a circuit board 32, which electronically controls LEDs 16 inserted through holes 35 at each node 14. Side panel 24 consists of a frame 26 and a battery 5 door 28 that is secured to frame 26 by flanges 27. Battery door 28 includes a slot 30 for prying the door 28 from frame 26 with a flat object, such a coin, permitting access to circuit board 32. A power switch 41 recessed in frame 26 turns the game on and off.

Referring to FIG. 3, the power switch 41 activates a microprocesor 34 by connecting it to a battery 37, which consists of 4 AA cell batteries. Once activated, microprocessor 34 samples a position sensor switch 38 for information which is used to control the color state 15 of LEDs 16 at each node 14 or the single light 17 of FIG. IA. In the preferred embodiment microprocessor 34 controls the state of only one LED 16 at each node 14 at any time. A change in the color state of any node 14 is governed by the ordering of the "Color State 20 Vector". As an example, the Color State Vector may be defined as:

GREEN-OFF-YELLOW-OFF-RED

After the RED state, the color state of the node would return to GREEN. The color state may be advanced 25 one position from left to right in the Color State Vector or right to left. Many other Color State Vectors are also possible.

Microprocessor 34 also controls an audio device such as a speaker 36 via an amplifier 39. The speaker 36 30 provides action sounds, for example clicking or beeping sounds, indicating a change in the color state at one of the nodes 14.

The object of the game is to manipulate the tetrahedron 10 into a state where all nodes 14 are lit by the 35 same color, for example when all of the red LEDs are lit. To accomplish this, the top node becomes the reference node. Thereafter, the color state at each node is advanced by selectively rotating the tetrahedron so that different nodes become the top node. The microprocessor 34 is continuously informed as to which node 14 is currently the top node by position sensor switch 38.

The puzzle of the present invention is based on a player's deciphering the color state vector pattern, that is, the pattern of switching from one color to the next as 45 a node is brought to the upright position. A player will thus manipulate the tetrahedron 10 bringing successive nodes to the top in an effort to figure out the pattern of color changes. After the player has deciphered the code, he then makes further rotations of the tetrahedron 50 in hopes of solving the puzzle such as having each node lighted red. There is thus no unique set of rotations necessary for solving the puzzle.

Shown in FIGS. 4A and 4B, position sensor switch 38 includes a cylindrical base 40 and cover 42 assembly 55 that is approximately ½ inch in diameter and ¼ inch in height, and made from electronically insulating material such as plastic. The assembly defines a cavity 44 that permits a conductive ball 46 to roll to one of four possible positions as indicated by arrows 48 and 50. Ball 46 is 60 approximately ½ inch in diameter, and made of silver plated steel. Three of the positions (indicated by arrows 48) are located between walls 52 of base 40, which extend into cavity 44. A pair of contact pins 54 are disposed at each of these positions. When ball 46 is 65 placed in contact between the pins an electrical connection is made. Contact between pins from adjacent positions is prevented by wall 52. The fourth position (indi-

cated by arrow 50) is located at a depression 56 formed in base 40. When the ball 46 is in this position, it is electrically isolated from any of the contact pins 54. Each of these four positions corresponds to a node 14.

As shown in FIG. 5, position sensor switch 38 operates as a three-way switch to inform microprocessor 34 of the relative position of the tetrahedron 10. Microprocessor 34 detects the position of the ball 46 by simultaneously sampling the voltage at each pin 54 connected to the positive terminal of the battery 37 via resistors R1 and power switch 41. Depending on the position of ball 46, the microprocessor is programmed to drive LEDs 16 according to the Color State Vector at the nodes 14.

In the preferred embodiment, the rules for five games are stored in the microprocessor. The first four games are selected by chosing one of the nodes as the top node before turning the power switch on. For example, if the node labeled 1 is the top node when power is turned on, game 1 will be played. If the node labeled 2 is the top node, game 2 will be played and so on. Game 5 may only be played at the end of game 4; that is, when all 4 nodes are red the microprocessor will switch to a game 5 mode. Typically, game 1 would be the easiest of the games and game 5 would be the most difficult. As an example, rules for playing each of the five games stored in the microprocessor are as follows:

GAME 1 consists of advancing one color in the selected top node according to the Color State Vector;

GAME 2 consists of advancing one color in the selected top node and advancing one color in the previous top node;

GAME 3 consists of advancing one color in the selected top node if that node was not visited in the previous two turns;

GAME 4 consists of advancing one color in each of the three nodes that are not the selected top node; and

GAME 5 consists of advancing one color in the selected top node, advancing one color in the previous top node and backing up one color in each of the remaining two nodes.

As demonstrated by the above rules, the patterns of play can become intricate and involved.

The processor is also programmed to enable an amplifier 39 to drive speaker 36 whenever a color state changes. Amplifier 39 includes a transistor Q having its emitter tied to the positive terminal of the battery and its collector tied to a voice coil L of speaker via resistor R2. The base of transistor Q is connected between a resistor R3 tied to the positive terminal of the battery 37 and a resistor R4 tied to the microprocessor 34 at an output terminal PFo. When microprocesor 34 drives the output terminal PFo low, transistor Q is enabled, thereby activating speaker 36.

When the puzzle is first powered up, or at the end of a game, software stored in microprocessor 34 idles in a pre-execution mode waiting for a new switch closure to start the next game. During the waiting period, microprocessor 34 runs a "light show" to keep idle spectators amused. During this light show, four LEDs 16 are continuously lit, one at each node 14. Every 40 milliseconds, a different node is visited, the current LED is turned off, and the next LED is turned on.

As shown in FIG. 6, tricolored LEDs 19 may be substituted for the individual LEDS 16 shown in FIG. 5.

As an example of the software design for executing the games, six software modules are appended below.

-continu	ea

DISABLE INTERRUPTS NITIALIZE STACK POINTER. NITIALIZE STACK POINTER. NITIALIZE LO PORTS TO EITHER INPUTS OR DITPUTS CLEAR OUTPUT PORTS. CLEAR GOVERNORS. CLEA	games, six software modules are appended below.			SWITCHES				
DISABLE INTERRUPTS NITIALIZE STACK POINTER. NITIALIZE STACK POINTER. NITIALIZE LO PORTS TO EITHER INPUTS OR DITPUTS CLEAR OUTPUT PORTS. CLEAR GOVERNORS. CLEA		_	GAME	A-B	TOP NODE	RULES		
DISABLE INTERRUPTS NITIALIZE IS TACK POINTER. NITIALIZE IS TACK POINTER. NITIALIZE IS TACK POINTER. 10 CLEAR COUTPUT PORTS. CLEAR RAM TO ALL ZERGES. CALL SWITCH READING. SET "GAME" = 4 SET "GAME" = 5. UMP TO PRE-EXEC COND. THE ADD FALLE SWITCH READING. SET PESSED TO GAME SHOW. THE DO FAME CLECK ND EXEC LOOP. DO WHILE (NO TEND OF GAME) ALL SWITCH READING. The selected top node and the previous top node advances one color. The selected top node advances one color only if twas not visited in the previous two turns. Game 4 is the same as Game 4, with one additional rule: If the top node advances one color. The selected top node advances one color only if twas not visited in the previous two turns. Game 4 is the same as Game 4, with one additional rule: If the top node advances one color. The selected top node advances one color only if the previous two turns. Game 4 is the same as Game 4, with one additional rule: If the top node advances one color. The selected top node advances one color only if the previous two turns. Game 4 is the same as Game 4 with one additional rule: If the top node advances one color. The selected top node advances one color, the previous top devil advances one color. The selected top node advances one color, the previous top node will also advance one color. The selected top node advances one color, the previous top node will also advance one color. The selected top node advances one color. The selected top node	POWER ON	- 5				The selected top node		
NITIALIZE LO PORTS TO EITHER INPUTS OR DUTPUTS. LEAR OUTPUT PORTS. LEAR COUTPUT PORTS. 2LEAR RAM TO ALL ZEROES. 2ALL SWITCH READING. SALL SWITCH READING. NIT 40 MILLISECOND PRE-EXEC TIMER. UIMP TO PRE-EXEC. PRE-EXEC P	DISABLE INTERRUPTS.	,						
NITIALIZE LO PORTS TO EITHER INPUTS OR DUTPUTS. LEAR OUTPUT PORTS. LEAR COUTPUT PORTS. 2LEAR RAM TO ALL ZEROES. 2ALL SWITCH READING. SALL SWITCH READING. NIT 40 MILLISECOND PRE-EXEC TIMER. UIMP TO PRE-EXEC. PRE-EXEC P	INITIALIZE STACK POINTER.		2 .	0.0	2	The selected top node and		
DUTPUT PORTS. LEAR AUTPUT PORTS. LEAR RAM TO ALL ZEROES. LEAR RAM TO ALL ZEROES. ALL SWITCH READING. INT 49 MILLISECOND PRE-EXEC TIMER. IUMP TO PRE-EXEC PRE-EX								
LEAR AUTPUT PORTS CLEAR RAM TO ALL ZEROES. CALL SWITCH READING. CALL SWITCH READING. NIT 40 MILLISECOND PRE-EXEC TIMER. LIMP TO PRE-EXEC. PRE-EXEC LOOP. DO WHILE (NO NEW SWITCH LIOSURE). AVE 40 MS PASSED? FYES, THEN DO. RESTART 40 MS TIMER. VIDINITY OF TYTH LED AT N'TH NODE. TURN OFF X'TH LED AT N'TH NODE. TURN OFF X'TH LED AT N'TH NODE. VIDIN OF SWEE EXEC. SINCE LOOP. UMP TO PRE-EXEC. SINCE LOOP. ALL SWITCH READING. SIND PRE-EXEC LOOP. UMP TO PRE-EXEC. SINCE LOOP. UMP TO PRE-EXEC. SINCE LOOP. ALL SWITCH READING. SIND PRE-EXEC LOOP. UMP TO PRE-EXEC. SINCE LOOP. ALL SWITCH READING. SINCE LOOP. ALL SWITCH CLOSURE, THEN DO: CALL GAME LOOF. CALL GAME LOOF. CALL BOYNER TI-DO FGAME PLAG. LEAR END OF GAME FLAG. SIND PRE-EXEC. SIND EXERCE LOOP. UMP TO PRE-EXEC. SIND EXEC LOOP. UMP TO PRE-EXEC. SINCE LOOP. ALL SWITCH READING. SIND PRE-EXEC LOOP. UMP TO PRE-EXEC. SIND OF GAME CHECK SIND EXEC LOOP. SIND EXERCATION OF GAME PLAG. SIND OF GAME CHECK SIND EXERCATION OF GAME PLAG. SIND OF GAME CHECK SIND EXERCATION OF GAME PLAG. SIND OF GAME CHECK SIND EXERCATION OF GAME PLAG. SIND OF GAME FLAG. SIND OF								
ALL AR RAM TO ALL ZEROES. CALL SWITCH READING. SET "GAME" = CURRENT SWITCH READING. NIT 40 MILLISECOND PRE-EXEC TIMER. UMP TO PRE-EXEC. PRE-EXEC PRE-E			3	0.0	3			
CALL SWITCH READING. NIT 40 MILLISECOND PRE-EXEC TIMER. NIT 40 MILLISECOND PRE-EXEC TIMER. NIT 40 MILLISECOND PRE-EXEC TIMER. 15 CLOSURE) RAVE 40 MS PASSED? FYES, THEN DO. RESTART 40 MS TIMER. 15 LOSURE) RAVE 40 MS PASSED? FYES, THEN DO. RESTART 40 MS TIMER. 15 10 11 The selected top node advances one color, the previous node will also advance one color. The selected top node is node labeled #1. it will advance a color only if the previous top node was node #1. it will advance a color only if the previous top node was node #1. it will advance a color only if the previous top node was node #1. it will advance a color only if the previous top node was node #1. it will advance one color. ALL LISHICH READING SAME LOGIC NUTICH READING AND PRE-EXEC. NUTICH READING SAME LOGIC NUTICH READING NUT TO PRE-EXEC. NUTICH READING AND PRE-EXEC. NUTICH READING The selected top node and the 2 previous top node will read the 2 previous top node and the 2 previous top node and the 2 previous top node will read advance one color. The selected top node and the 2 previous top node will read advance one color. The selected top node and the 2 previous top node will read adva			5	00	J ,			
THE TYPICAME CURRENT SWITCH READING. IT TYPICAME CURRENT SWITCH READING. IN MILLISECOND PRE-EXEC TIMER. 4 00 4 Game 4 is the same as Game 3, with one additional rule: If the top node advances one color, the previous node will also advance one color. The same as Game 4 is the same as Game 5 is the same as Game 4 is the same as Game 4 is the same as Game 4 is the same as Game 5 is the same as Game 4 is the same as Game 5 is the same as Game 5 is the same as Game 6 is not alreaded and all onches as the top will advance one color. The same 4 is the same as Game 4 is the same as Game 5 is the same as Game 6 is not alreaded and all onches and the previous top node was different from the current top node. The same 4 is the same as Game 5 is the previous top node and the previous top node was firent from the current pount 6 is node labeled #1, it will advance one color. The prev		10						
INIT 40 MILLISECOND PRE-EXEC TIMER. UIMP TO PRE-EXEC. PRE-EXEC LOOP. DO WHILE (NO NEW SWITCH LOSURE) PRE-EXEC LOOP. DO WHILE (NO NEW SWITCH LOSURE) PAVE 40 MS PASSED? FYES, THEN DO: RESTART 40 MS TIMER. UIST N'TH 0 F4 NODES. UIRN ON X + 1'TH LED AT N'TH NODE. UIRN ON X + 1'TH LED AT N'TH NODE. UIRN ON X + 1'TH LED AT N'TH NODE. UIRN ON X + 1'TH LED AT N'TH NODE. ALL SWITCH READING. SIND PRE-EXEC LOOP. UMP TO EXEC. EXEC LOOP. DO WHILE (NOT END OF GAME) ALL SWITCH READING F NEW SWITCH CLOSURE, THEN DO: ALL LISTOR GAME FLAG. F "GAME" = 4, SET "GAME" = 5. UMP TO PRE-EXEC. DO WE HAVE A NEW SWITCH CLOSURE VALUE. EXED OWERAY SWITCH CLOSURE VALUE. EXED OWERAY SWITCH CLOSURE VALUE. EXED OWERAY SWITCH CLOSURE VALUE. EXED CURRENT SWITCH CLOSURE AKEAD CURRENT SWITCH CLOSURE OWE HAVE A NEW SWITCH CLOSURE AKEAD CURRENT SWITCH CLOSURE AKEAD CURRENT SWITCH CLOSURE OWE HAVE A NEW SWITCH CLOSURE AKEAD CURRENT SWITCH CLOSURE OWE HAVE A NEW SWITCH CLOSURE AKEAD CURRENT SWITCH CLOSURE AKE ALD CURRENT SWITCH CLOSURE AKEAD CURRENT SWITCH								
IUMP TO PRE-EXEC. IUMP TO EXEC. IUMP TO EXEC.				0.0	4			
PRE-EXEC PRE-EXEC LOOP: DO WHILE (NO NEW SWITCH LISURUS LOOP: DO WHILE (NO NEW SWITCH LISURUS LOOP: DO WHILE (NO NEW SWITCH LISURUS LOOP: DO WHILE (NO NEW SWITCH LESTART 40 MS TIMER. S 10 1 The selected top node remains unchanged and all other nodes advance one color. The selected top node is node labeled #1, it will advance one color. URIN ON X + 1'TH LED AT N'TH NODE. SIND PRE-EXEC LOOP. UMP TO EXEC. EXEC LOOP. UMP TO EXEC. EXEC LOOP. DO WHILE (NOT END OF GAME) ZALL SWITCH READING F NEW SWITCH CLOSURE, THEN DO: ZALL DISPLAY UPDATE ZALL END OF GAME CHECK ND EXEC LOOP. LEAR END OF GAME FLAG. F "GAME" = 4, SET "GAME" = 5. UMP TO PRE-EXEC. DO WE HAVE A NEW SWITCH CLOSURE VALUE. READ CURRENT SWITCH CLOSURE. SEAD CURRENT SWITCH CLOSURE. DO WE HAVE A NEW SWITCH CLOSURE AKEAD CURRENT SWITCH CLOSURE. DO WE HAVE A NEW SWITCH CLOSURE AKEAD CURRENT SWITCH CLOSURE. DO WE HAVE A NEW SWITCH CLOSURE. DO WE HAVE A NEW SWITCH CLOSURE. AKEAD CURRENT SWITCH CLOSURE. DO WE HAVE A NEW SWITCH CLOSURE. DO WE HAVE A NEW SWITCH CLOSURE. AKEAD CURRENT SWITCH CLOSURE. DO WE HAVE A NEW SWITCH CLOSURE. AKEAD CURRENT SWITCH CLOSURE. DO WE HAVE A NEW SWITCH CLOSURE. AKEAD CURRENT SWITCH CLOSURE. DO WE HAVE A NEW SWITCH CLOSURE. AKE KEY CLICK SOUND DONVERT I/O VALUE TO KEY ID ALL I SMODES = RED? F YES, THEN DO. VHICH GAME ARE WE IN? AMME LOGIC VHICH GAME ARE WE IN? AMME LOGIC FOR GAMES 1-5. 10 0 1 2 The selected top node and the 2 previous top node will advance one color. Only the previous top node will advance one color. Only the previous top node will advance one color. F "GAME" = 5, THEN DO. UN END OF GAME 5 SHOW STERNALLY HIER STATE OF ALL 4 NODES: AF JEN THEN TOO THE COLOR T			4	00	4			
RPE_EXEC LOOP: DO WHILE (NO NEW SWITCH 15 LOSURE) AVE 40 MS PASSED? F YES, THEN DO: F YES, THEN DO: FYES, THEN DO: The selected top node remains unchanged and all order one color. The selected top node remains unchanged and all order one color. The selected top node remains unchanged and all order one color. The selected top node remains unchanged and all order one color. The selected top node remains unchanged and all order nodes advance one color. The selected top node remains unchanged and all order nodes advance one color. The selected top node is node labeled #1, it will advance one color. Other nodes at the top will advance a color only if the previous top node was different from the current top node. SEC LOOP. DO WHILE (NOT END OF GAME) ZALL SWITCH READING F NEW SWITCH CLOSURE, THEN DO: ALL END OF GAME CHECK SIND EXEC LOOP. LEAR END OF GAME FLAG. F "GAME" = 4, SET "GAME" = 5. UMP TO PRE-EXEC LOOP. UMP TO PRE-EXEC. SET PREVIOUS SWITCH CLOSURE SET PREVIOUS SWITCH CLOSURE AKE KEY CLICK SOUND ONVERT I/O VALUE TO KEY ID AME LOGIC ALL END OF GAME SI-5. ID 0 1 2 The selected top node and the 2 previous top node will advance one color. The selected top node and the 2 previous top node will devance one color. The selected top node and the 2 previous top node will remain unchanged. ONLY END OF GAME SI-5. ID 0 1 2 The selected top node and the 2 previous top node will remain unchanged. Only the previous top node will advance one color. The selected top node and the 2 previous top node will remain unchanged. Only the previous top node and the 2 previous top node will advance one color. The selected top node and the 2 previous top node will remain unchanged. Only the previous top node will advance one color. The selected top node and the 2 previous top node will remain unchanged. Only the previous top node will remain unchanged. Only the previous top node and the previous top node will remain unchanged. Only the previous top node and the previous top node will remain unchange								
LOSURE) AVE 40 MS PASSED? F YES, THEN DO: RESTART 40 MS TIMER. TURN OFF X'TH LED AT N'TH NODE. TURN ON X + 1'TH LED AT N'TH NODE. TURN ON X + 1'TH LED AT N'TH SALL SWITCH READING. EXEC LOOP. UMP TO EXEC. EXEC LOOP. SACE LOOP. DO WHILE (NOT END OF GAME) ALL SWITCH CLOSURE, THEN DO: ALL DISPLAY UPDATE SALL DISPLAY UPDATE ALL DISPLAY UPDATE ALL DISPLAY UPDATE SALL SWITCH CLOSURE, THEN DO: LEAR END OF GAME FLAG. F "GAME" = 4, SET "GAME" = 5. UMP TO PRE-EXEC. SWITCH CLOSURE VALUE. SET PREVIOUS SWITCH CLOSURE VALUE. SET PREVIOUS SWITCH CLOSURE AWAR EA NEW SWITCH CLOSURE AWAR EA NEW SWITCH CLOSURE AMAE LOGIC SWITCH READING THE selected top node equals the selected top node plus the previous top nodes each advance one color. The selected top node equals the selected top node plus the previous top node and the 2 previous top nodes each advance one color. The selected top node equals the selected top node equal	PRE-EXEC							
CLOSURE) HAVE 40 MS PASSED? F YES, THEN DO: RESTART 40 MS TIMER. VISIT N'THO P6 4 NODES. UIRN OFF X'TH LED AT N'TH NODE. UIRN ON X 1 I'TH LED AT N'TH NODE. VIDRO NOX X 1 I'TH LED AT N'TH VIDRO NX X 1 I'TH LED AT N'TH VII A VIDRO NX VIDRO	PRE-EXEC LOOP: DO WHILE (NO NEW SWITCH	15						
color. F YES, THEN DO: RESTART 40 MS PASSED? F YES, THEN DO: RESTART 40 MS TIMER. 10 The selected top node remains unchanged and all other nodes advance one color. The selected top node remains unchanged and all other nodes advance one color. The selected top node remains unchanged and all other nodes advance one color. The selected top node is node labeled #1. it will advance one color. Other nodes at the top will advance one color only if the previous top node was node #1, and the second previous top node was node #1, and the current top node. ALL SWITCH READING F NEW SWITCH CLOSURE, THEN DO: ALL DISTRICT GAME LOGIC ALL DISTRICT GAME CHECK ALL DISTRICT GAME FLAG. F "GAME" = 4, SET "GAME" = 5. UMP TO PRE-EXEC LOOP. DIEAR END OF GAME CHECK SWITCH READING SET PREVIOUS SWITCH CLOSURE VALUE. SET PREVIOUS SWITCH CLOSURE AKE AD CURRENT SWITCH CLOSURE AKE CURRENT SWITCH CLOSURE AND WE HAVE A NEW SWITCH CLOSURE AKE KEY CLICK SOUND DOWE HAVE A NEW SWITCH CLOSURE AKE KEY CLICK SOUND DOWE HAVE A NEW SWITCH CLOSURE AKE KEY CLICK SOUND DOWE HAVE A NEW SWITCH CLOSURE AKE ALL 4 NODES = RED? F "GAME" = 5, THEN DO: WHICH GAME ARE WE IN? SAME LOGIC FOR GAMES 1-5. NEW CLICK SOUND The selected top node and the 2nd previous top node will advance one color. The selected top node and the previous top node will remain unchanged. Only the previous top node will advance one color. Game 9, with one additional rule: If the top node will remain unchanged. Only the previous top node will advance one color. The selected top node and the previous top node will advance one color. The selected top node and the previous top node will remain unchanged. Only the previous top node will advance one color. The selected top node and the previous top node will remain unchanged. Only the previous top node will not previous	CLOSURE)							
F YES, THEN DO: RESTART 40 MS TIMER. VISIT N'TH OF 4 NODES. UINTO OFF X'TH LED AT N'TH NODE. UINTO OFF X'TH LED AT N'TH OODE. ALL SWITCH READING. SEND PRE-EXEC LOOP. UIMP TO EXEC. SXEC SXE	HAVE 40 MS PASSED?							
RESTART 40 MS TIMER. VISIT N'TH OF 4 NODES. VIUNT OF Y NODES. VIUNT OF X'TH LED AT N'TH NODE. VIONDE. ALL SWITCH READING. EXEC LOOP. VIUMP TO EXEC. EXEC LOOP. DO WHILE (NOT END OF GAME) CALL SWITCH READING F NEW SWITCH CLOSURE, THEN DO: CALL BISPLAY UPDATE CALL END OF GAME CHECK SWITCH READING F "GAME" = 4, SET "GAME" = 5. UMP TO PRE-EXEC. SWITCH READING F "GAME" AND W SWITCH CLOSURE. SWITCH READING F "GAME" AND W SWITCH CLOSURE SEXPLOOP. OUNT HAVE A NEW SWITCH CLOSURE WHAVE A NEW SWITCH CLOSURE WHAVE A NEW SWITCH CLOSURE WHAVE A NEW SWITCH CLOSURE WHICH GAME ARE WE IN? SAME LOGIC VIUNE TO KEY ID VALUE SAME LOGIC VIUNE TO GAME S SHOW STEFRALLY VIUNE NO DO F GAME S SHOW STEFRALLY VIUNE NO DO F GAME S SHOW STEFRALLY VIUNE NO DO F GAME S SHOW STERNALLY VIUNE NO DO F GAME S SHOW VILSE NO. VIENT OF TO THE STATE OF ALL 4 NODES: VIUNE NO DO F GAME S SHOW VILSE NO. VIENT OF TO THE STATE OF ALL 4 NODES: VIUNE NO DO F GAME S SHOW VILSE NO. VIENT OF TO THE STATE OF ALL 4 NODES: VIENT OF TO THE STATE OF ALL 4 NODES: VIENT OF TO THE STATE OF ALL 4 NODES: VIENT OF TO THE STATE OF ALL 4 NODES: VIENT OF THE STATE OF ALL			_		_			
VISIT N'TH OF 4 NODES. TURN OF X'TH LED AT N'TH NODE. TURN OF X'TH LED AT N'TH NODE. TURN OF X'TH LED AT N'TH NODE. TURN OR X'TH LED AT N'TH NODE. 6 10 2 If the selected top node is node labeled #1, it will advance one color. Other nodes advance one color. Other nodes at the top will advance a color only if the previous top node was node #1, and the second previous top node was different from the current top node. ALL SWITCH READING F NEW SWITCH CLOSURE, THEN DO: ALL LEAD OF GAME CHECK AND DETAIL END OF GAME FLAG. F "GAME" = 4, SET "GAME" = 5. UMP TO PRE-EXEC. SET PREVIOUS SWITCH CLOSURE VALUE. UMP TO PRE-EXEC. SET PREVIOUS SWITCH CLOSURE VALUE. UEAD CURRENT SWITCH CLOSURE? F YES, THEN DO: DOWE HAVE A NEW SWITCH CLOSURE? F YES, THEN DO: DOWE HAVE A NEW SWITCH CLOSURE MAKE KEY CLICK SOUND OND HAVE A NEW SWITCH CLOSURE MAKE KEY CLICK SOUND ONVERT IN OVALUE TO KEY ID AME LOGIC VAILUE. AND OF GAME S 1-5. SOUND F GAME CHECK NO DO GAME ARE WE IN? SAME LOGIC FOR GAMES 1-5. SOUND F GAME S SHOW STERNALLY WISS DO: STEPPEVIOUS OF GAME 5 SHOW STERNALLY STEND DO: SOUND F GAME S SHOW STERNALLY STEND DO: SOUND F GAME S SHOW STERNALLY SOUND F GAME S SHOW STERN			5	10	1			
TURN OFF X'TH LED AT N'TH NODE. TURN ON X + 1'TH LED AT N'TH SALL SWITCH READING. EXEC LOOP. TOTAL SWITCH READING EXEC LOOP. DO WHILE (NOT END OF GAME) EXEC LOOP. EXEC LOOP. DO WHILE (NOT END OF GAME) EXEC LOOP. TO 10 EXEC LOOP. The selected top node exact advance one color. EXEC LOOP. EXEC LOOP. The selected top node exact advance one color. EXEC LOOP. EXEC LOOP. The selected top node exact advance one color. EXEC LOOP. EXEC LOOP. The selected top node EXEC LOOP. EXEC LOOP. The selected top node and the 2 previous top node and the 2 pre								
FURN ON X + 1°TH LED AT N°TH NODE. NODE. ALL SWITCH READING. SMP PRE-EXEC LOOP. UMP TO EXEC. SXEC SXEC SXEC SXEC SXEC SXEC SXEC SXEC		20						
NODE. CALL SWITCH READING. SIMD PRE-EXEC LOOP. UMP TO EXEC. EXEC LOOP: DO WHILE (NOT END OF GAME) CALL SWITCH READING EXEC LOOP: DO WHILE (NOT END OF GAME) CALL SWITCH READING EXEC LOOP: DO WHILE (NOT END OF GAME) CALL SWITCH READING EXEC LOOP: DO WHILE (NOT END OF GAME) CALL SWITCH READING EXEC LOOP: DO WHILE (NOT END OF GAME) CALL SWITCH READING EXEC LOOP: DO WHILE (NOT END OF GAME) CALL GAME LOGIC CALL DISPLAY UPDATE CALL END OF GAME CHECK SIND EXEC LOOP. CALL END OF GAME FLAG. F "GAME" = 4, SET "GAME" = 5. UMP TO PRE-EXEC. WITCH READING SHET PREVIOUS SWITCH CLOSURE VALUE. SEAD CURRENT SWITCH CLOSURE VALUE. SEAD CURRENT SWITCH CLOSURE F YES, THEN DO: COWE HAVE A NEW SWITCH CLOSURE AME KEYE CLICK SOUND CONVERT 1/0 VALUE TO KEY ID AME LOGIC VHICH GAME ARE WE IN? SAME LOGIC F "GAME" = 5, THEN DO: CONCEPT THE STATE OF ALL 4 NODES: READ OF GAME S SHOW STERNALLY WIS SE DO: 10 01 2 The selected top node and the previous top node will advance one color. If a player returns to a node that he had just previously visited, all four nodes go blank!		20						
EALL SWITCH READING. END PRE-EXEC LOOP. UMP TO EXEC. EXEC EXEC EXEC LOOP: DO WHILE (NOT END OF GAME) CALL SWITCH READING EXEC LOOP: DO WHILE (NOT END OF GAME) CALL SWITCH READING F NEW SWITCH CLOSURE, THEN DO: CALL DISPLAY UPDATE CALL DISPLAY UPDATE CALL END OF GAME CHECK END EXEC LOOP: CALL DISPLAY UPDATE AND EXEC LOOP: CALL BND OF GAME CHECK SOUTH TO GAME FLAG. F "GAME" = 4, SET "GAME" = 5. AND UMP TO PRE-EXEC. SOUTH TO PRE-EXEC. SOUTH TO PRE-EXEC. SOUTH TO PRE-EXEC. SOUND CONVERT I/O VALUE TO KEY ID AND ELOGIC WHICH GAME ARE WE IN? AND OF GAME CHECK NSPECT THE STATE OF ALL 4 NODES: AND OF GAME CHECK NSPECT THE STATE OF ALL 4 NODES: AND OF GAME CHECK NSPECT THE STATE OF ALL 4 NODES: AND OF GAME S SHOW THEN DO: WIN END OF GAME 5 SHOW THEN DO: WIN END OF GAME 5 SHOW THEN ALL 4 NODES AND CHECK AND OF GAME S SHOW THEN DO: WIN END OF GAME 5 SHOW THEN ALL 4 NODES AND CHECK AND THEN ALL 4 NODES A			6	10	2			
END PRE-EXEC LOOP. UMP TO EXEC. EXEC UMP TO EXEC. EXEC EXEC LOOP: DO WHILE (NOT END OF GAME) EXEL SWITCH READING EXEL LOOP: DO WHILE (NOT END OF GAME) EXALL END OF GAME CHECK EXEL END OF GAME CHECK EXEL END OF GAME FLAG. F "GAME" = 4, SET "GAME" = 5. EXEL END OF GAME FLAG. F "GAME" = 5. EXEL END OF GAME FLAG. EXEL END OF GAME SWITCH CLOSURE VALUE. EXEL EXEL EXEL EXEL EXEL EXEL EXEL EXE						is node labeled #1, it		
UMP TO EXEC. EXEC LOOP. DO WHILE (NOT END OF GAME) EXEL END OF GAME LOGIC EXALL DISPLAY UPDATE EXALL END OF GAME CHECK EXALL END OF GAME CHECK EXALL END OF GAME FLAG. F "GAME" = 5. 8 10 4 The selected top node equals the selected top node (Modulo 4). The selected top node (Modulo 4). The selected top node (Modulo 4). The selected top node and the 2 previous top nodes each advance one color. Game 9 is the same as Game 8, with one additional rule: If the top node and the 2m previous top node are the same, then the top node will advance one color. GAME LOGIC WHICH GAME ARE WE IN? GAME LOGIC FOR GAMES 1-5. EXECUTED STATE OF ALL 4 NODES: REF ALL 4 NODES = RED? F YES, THEN DO: UN END OF GAME 5 SHOW EXECUTED STATE OF ALL 4 NODES: REF STATE OF GALL 4 NODES: REF STATE OF GAME 5 SHOW EXECUTED STATE OF ALL 4 NODES: REF STATE OF GAME 5 SHOW EXECUTED STATE OF ALL 4 NODES: REF STATE OF GAME 5 SHOW EXECUTED STATE OF ALL 4 NODES: REF STATE OF GAME 5 SHOW EXECUTED STATE OF ALL 4 NODES: REF STATE OF GAME 5 SHOW EXECUTED STATE OF ALL 4 NODES: REF STATE OF GAME 5 SHOW EXECUTED STATE OF ALL 4 NODES: REF STATE OF GAME 5 SHOW EXECUTED STATE OF ALL 4 NODES: REF STATE OF GAME 5 SHOW EXECUTED STATE OF ALL 4 NODES: REF STATE OF GAME 5 SHOW EXECUTED STATE OF ALL 4 NODES: REF STATE OF GAME 5 SHOW EXECUTED STATE OF ALL 4 NODES: REF STATE OF GAME 5 SHOW EXECUTED STATE OF ALL 4 NODES: REF STATE OF GAME 5 SHOW EXECUTED STATE OF ALL 4 NODES: REF STATE OF GAME 5 SHOW EXECUTED STATE OF ALL 4 NODES: REF STATE OF GAME 5 SHOW EXECUTED STATE OF ALL 4 NODES: REF STATE OF GAME 5 SHOW EXECUTED STATE OF GAME 5 SHOW EXECU						will advance one color.		
EXEC EXEC LOOP: DO WHILE (NOT END OF GAME) EXEC LOOP: DO WHILE (NOT END OF GAME) EXEC LOOP: DO WHILE (NOT END OF GAME) EXEL SWITCH READING F NEW SWITCH CLOSURE, THEN DO: EXALL GAME LOGIC EXALL DED OF GAME CHECK EXEC LOOP. EXE EXEC LOOP. EXEC LOOP. EXEC LOOP. EXEC LOOP. EXEC LOOP. EXE EXEC LOOP. EXEC LOOP. EXEC LOOP. EXEC LOOP. EXEC LOOP. EXE LOOP. EXEL LOO						Other nodes at the top		
EXEC LOOP: DO WHILE (NOT END OF GAME) CALL SWITCH READING F NEW SWITCH CLOSURE, THEN DO: CALL DISPLAY UPDATE TO SAME FLAG. TO SAME FLAG. TO SAME FLAG. The selected top node equals the selected top node (Modulo 4). The selected top node and the 2 previous top node will end wincome one color. SET PREVIOUS SWITCH CLOSURE VALUE. SEAD CURRENT SWITCH CLOSURE. DO WE HAVE A NEW SWITCH CLOSURE. DO WE HAVE A NEW SWITCH CLOSURE. DO WE HAVE A NEW SWITCH CLOSURE MAKE KEY CLICK SOUND CONVERT I/O VALUE TO KEY ID ALUE. SAME LOGIC WIICH GAME ARE WE IN? GAME LOGIC WIICH GAME ARE WE IN? GAME LOGIC WIICH GAME ARE WE IN? SAME LOGIC FOR GAMES 1-5. ID 01 2 The selected top node and the previous top node will advance one color. Only the previous top node will advance one color. The selected top node and the 2nd previous top node will advance one color. Only the previous top node will advance one color. If a player returns to a node that he had just previously visited, all four nodes go blank!						will advance a color only		
ALL SWITCH READING F NEW SWITCH CLOSURE, THEN DO: CALL GAME LOGIC CALL END OF GAME CHECK CALL END OF GAME FLAG. F "GAME" = 4, SET "GAME" = 5. CALL DISPLAY UPDATE SET PREVIOUS SWITCH CLOSURE VALUE. READ CURRENT SWITCH CLOSURE. F YES, THEN DO: CALVE AKE KEY CLICK SOUND CONVERT I/O VALUE TO KEY ID CALUE. CAME LOGIC WHICH GAME ARE WE IN? GAME LOGIC WHICH GAME ARE WE IN? GAME LOGIC WHICH GAME ARE WE IN? GAME CHECK WAS node #1, and the second previous top node was different from the current top node. The selected top node equals the selected top node plus the previous top node of the 2 previous top node and the 2 previous top nodes each advance one color. GAME LOGIC WHICH GAME ARE WE IN? GAME LOGIC FOR GAMES 1-5. WHICH GAME ARE WE IN? GAME LOGIC FOR GAMES 1-5. WHICH GAME ARE WE IN? GAME LOGIC FOR GAMES 1-5. WHICH GAME ARE WE IN? GAME LOGIC FOR GAMES 1-5. WHICH GAME ARE WE IN? GAME LOGIC FOR GAMES 1-5. WHICH GAME ARE WE IN? GAME LOGIC FOR GAMES 1-5. WHICH GAME ARE WE IN? GAME LOGIC FOR GAMES 1-5. WHICH GAME ARE WE IN? GAME LOGIC FOR GAMES 1-5. WHICH GAME ARE WE IN? GAME LOGIC FOR GAMES 1-5. WHICH GAME ARE WE IN? GAME LOGIC FOR GAMES 1-5. WAS DOWN OF GAME OF THE STATE OF ALL 4 NODES: ARE ALL 4 NODES: ARE ALL 4 NODES = RED? F YES, THEN DO: WHICH GAME 5 SHOW WAS node #1, and the second previous top node was different from the current top node will advance one color. The selected top node and the 2 previous top node will advance one color. The selected top node and the previous top node will advance one color. The selected top node and the previous top node will advance one color. The selected top node and the previous top node will advance one color. The selected top node and the previous top node will advance one color. The selected top node and the 2 nd previous top node will advance one color. The selected top node and the 2 nd previous top node will advance one		25						
SALL SWITCH CLOSURE, THEN DO: CALL GAME LOGIC CALL DISPLAY UPDATE CALL DISPLAY UPDATE CALL END OF GAME CHECK CEND EXEC LOOP. CLEAR END OF GAME FLAG. F "GAME" = 5. CLEAR END OF GAME" = 5. CLEAR END OF GAME" = 5. CLEAR END OF REAECC. SWITCH READING SET PREVIOUS SWITCH CLOSURE VALUE. CLEAR END CURRENT SWITCH CLOSURE. CLEAR END OF GAME SWITCH CLOSURE. CLEAR END OF GAME SWITCH CLOSURE VALUE. CLEAR END OF GAME SWITCH CLOSURE VALUE. CLEAR END OF GAME STORM SWITCH CLOSURE VALUE. CLEAR END OF GAME STORM SWITCH CLOSURE VALUE. CLEAR END OF GAME SHOW THE SEPON SWITCH CLOSURE VALUE. AND WE HAVE A NEW SWITCH CLOSURE. COWE HAVE A NEW SWITCH CLOSURE AAKE KEY CLICK SOUND CONVERT I/O VALUE TO KEY ID VALUE. CALL END OF GAME SHOW THE SEPON SHOW THE STATE OF ALL 4 NODES: ARE ALL 4 NODES = RED? F "GAME" = 5, THEN DO: CUN END OF GAME 5 SHOW THE SEPON SWITCH CLOSURE VALUE. TO UT THE STATE OF ALL 4 NODES: ARE ALL 4 NODES = RED? F "GAME" = 5, THEN DO: CONTRIBUTION TO THE STATE OF ALL 4 NODES: ARE ALL 4 NODES = RED? F "GAME" = 5, THEN DO: CONTRIBUTION THE STATE OF ALL 4 NODES: ARE ALL 4 NODES = RED? F "GAME" = 5, THEN DO: CONTRIBUTION THE STATE OF ALL 4 NODES: AND OF GAME SHOW THE NO OF THE STATE OF ALL 4 NODES: AND OF GAME SHOW THE NO OF THE STATE OF ALL 4 NODES: AND OF GAME SHOW THE NO OF THE STATE OF ALL 4 NODES: AND OF THE STATE OF ALL								
Was different from the current top node. CALL DISPLAY UPDATE CALL END OF GAME CHECK CALL END OF GAME CHECK CALL END OF GAME FLAG. F "GAME" = 4, SET "GAME" = 5. WINTER READING CHERRENT SWITCH CLOSURE VALUE. SEAD OWE HAVE A NEW SWITCH CLOSURE? F YES, THEN DO: CONVERT I/O VALUE TO KEY ID VALUE. GAME LOGIC WHICH GAME ARE WE IN? GAME LOGIC WHICH GAME ARE WE IN? GAME LOGIC FOR GAMES 1-5. WAS different from the current top node. To 10 3 The selected top node equals the selected top node plus the previous top node plus the previous top node and the 2 previous top node and the 2 previous top node and the 2 previous top node are the same, with one additional rule: If the top node and the 2nd previous top node are the same, then the top node will remain unchanged. Only the previous top node are the same, then the top node will advance one color. GAME LOGIC FOR GAMES 1-5. WHICH GAME ARE WE IN? GAME LOGIC FOR GAMES 1-5. 10 01 2 The selected top node and the 2 previous top node will advance one color. The selected top node and the 2 previous top node and the previous top node will advance one color. The selected top node and the 2 previous top node and the previous top node will advance one color. The selected top node and the 2 previous top node and the 2 previous top node will advance one color. The selected top node and the 2 previous top node and the previous top node will advance one color. The selected top node and the 2 previous top node and the								
CALL DISPLAY UPDATE CALL DISPLAY UPDATE CALL END OF GAME CHECK END EXEC LOOP. CLEAR END OF GAME FLAG. F "GAME" = 4, SET "GAME" = 5. UMP TO PRE-EXEC. SWITCH READING ET PREVIOUS SWITCH CLOSURE VALUE. CEAR END OF GAME SWITCH CLOSURE. COW HAVE A NEW SWITCH CLOSURE. COW HAVE A NEW SWITCH CLOSURE MAKE KEY CLICK SOUND CONVERT I/0 VALUE TO KEY ID VALUE. GAME LOGIC WHICH GAME ARE WE IN? GAME LOGIC FOR GAMES 1-5. END OF GAME CHECK NSPECT THE STATE OF ALL 4 NODES: ARE ALL 4 NODES = RED? F "GAME" = 5, THEN DO: CRUN END OF GAME 5 SHOW ETERNALLY USES DO. CUITCH top node. The selected top node equals the selected top node photo equals the selected top node and the 2 previous top node and the 2 previous top node and the 2 previous top node and the 2 nd previous top node and the 2 nd previous top node and the 2 nd previous top node will advance one color. The selected top node and the 2 nd previous top node will advance one color. The selected top node and the 2 nd previous top node advance one color. The selected top node and the 2 nd previous top node advance one color. The selected top node and the 2 nd previous top node advance one color. If a player returns to a node that he had just previously visited, all four nodes go blankl								
CALL END OF GAME CHECK END EXEC LOOP. CLEAR END OF GAME FLAG. END EXEC LOOP. CLEAR END OF GAME FLAG. END EXEC LOOP. CLEAR END OF GAME FLAG. END EXEC LOOP. CLEAR END OF GAME FLAG. END EXEC LOOP. CLEAR END OF GAME FLAG. END EXEC LOOP. CLEAR END OF GAME FLAG. END EXEC LOOP. CLEAR END OF GAME FLAG. END EXEC LOOP. AUTHORITIES TO THE SELECTED TO THE SELECT	CALL GAME LOGIC							
EALL END OF GAME CHECK END EXEC LOOP. LEAR END OF GAME FLAG. F "GAME" = 4, SET "GAME" = 5. UMP TO PRE-EXEC. SET PREVIOUS SWITCH CLOSURE VALUE. READ CURRENT SWITCH CLOSURE. DO WE HAVE A NEW SWITCH CLOSURE? F YES, THEN DO: CONVERT 1/0 VALUE TO KEY ID ALUE. GAME LOGIC WHICH GAME ARE WE IN? GAME LOGIC WHICH GAME ARE WE IN? SAME LOGIC FOR GAMES 1-5. END OF GAME CHECK NSPECT THE STATE OF ALL 4 NODES: ARE ALL 4 NODES = RED? F "GAME" = 5, THEN DO: CRUN END OF GAME 5 SHOW ETERNALLY ETERNALLY ETERNALLY EQUALS He selected top node plus the previous top node and the 2 previous top node are the same, then the top node will remain unchanged. Only the previous top node will advance one color. The selected top node and the previous top node advance one color. If a player returns to a node that he had just previously visited, all four nodes go blank!	CALL DISPLAY UPDATE		7	1.0	3			
END EXEC LOOP. CLEAR END OF GAME FLAG. F "GAME" = 4, SET "GAME" = 5. UMP TO PRE-EXEC. SWITCH READING GET PREVIOUS SWITCH CLOSURE VALUE. READ CURRENT SWITCH CLOSURE. DO WE HAVE A NEW SWITCH CLOSURE? F YES, THEN DO: DEBOUNCE NEW SWITCH CLOSURE MAKE KEY CLICK SOUND CONVERT 1/0 VALUE TO KEY ID ALUE. GAME LOGIC WHICH GAME ARE WE IN? GAME LOGIC FOR GAMES 1-5. END OF GAME CHECK NSPECT THE STATE OF ALL 4 NODES: ARE ALL 4 NODES = RED? F YES, THEN DO: RUN END OF GAME 5 SHOW STERNALLY IN The selected top node and the 2nd previous top node will advance one color. In O 1 2 The selected top node and the previous top node and the previous top node and the previous top node will advance one color. In O 1 2 The selected top node and the previous top node advance one color. In O 1 2 The selected top node and the previous top node advance one color. In O 1 2 The selected top node and the previous top node advance one color. In O 1 2 The selected top node and the previous top node advance one color. In SERVICE STATE OF ALL 4 NODES: ARE ALL 4 NODES = RED? F YES, THEN DO: RUN END OF GAME 5 SHOW STERNALLY IN THE SERVICE A 10 4 The selected top node and the 2nd previous top node will remain unchanged. Only the previous top node advance one color. In O 1 2 The selected top node and the previous top node advance one color. If a player returns to a node that he had just previously visited, all four nodes go blank!	CALL END OF GAME CHECK	30	•	. 0	J .			
top node (Modulo 4). The selected top node and the 2 previous top nodes each advance one color. Game 1 top node and the 2 previous top nodes each advance one color. Game 9 is the same as Came 8, with one additional rule: If the top node and the 2nd previous top node are the same, then the top node are the same, then the top node will remain unchanged. Only the previous top node will advance one color. Game Logic For Games 1-5. WHICH GAME ARE WE IN? GAME LOGIC WHICH GAME ARE WE IN? GAME LOGIC FOR GAMES 1-5. WHICH GAME ARE WE IN? GAME LOGIC FOR GAMES 1-5. WHICH STATE OF ALL 4 NODES: ARE ALL 4 NODES = RED? F "GAME" = 5, THEN DO: RUN END OF GAME 5 SHOW STERNALLY WISE DO.	END EXEC LOOP.	50						
H "GAME" = 4, SE1 "GAME" = 5. IMP TO PRE-EXEC. INVITCH READING GET PREVIOUS SWITCH CLOSURE VALUE. READ CURRENT SWITCH CLOSURE. DO WE HAVE A NEW SWITCH CLOSURE? F YES, THEN DO: DEBOUNCE NEW SWITCH CLOSURE MAKE KEY CLICK SOUND CONVERT I/O VALUE TO KEY ID JALUE. GAME LOGIC WHICH GAME ARE WE IN? GAME LOGIC FOR GAMES 1-5. ARE ALL 4 NODES = RED? F YES, THEN DO: RUN END OF GAME 5 SHOW The selected top node and the 2 previous top node will remain unchanged. Only the previous top node and the previous top node and the previous top node and the previous top node advance one color. If a player returns to a node that he had just previously visited, all four nodes go blank!	CLEAR END OF GAME FLAG.							
UMP TO PRE-EXEC. SWITCH READING GET PREVIOUS SWITCH CLOSURE VALUE. READ CURRENT SWITCH CLOSURE. DO WE HAVE A NEW SWITCH CLOSURE? F YES, THEN DO: DEBOUNCE NEW SWITCH CLOSURE MAKE KEY CLICK SOUND CONVERT 1/0 VALUE TO KEY ID VALUE. GAME LOGIC WHICH GAME ARE WE IN? GAME LOGIC FOR GAMES 1-5. BAND OF GAME CHECK NSPECT THE STATE OF ALL 4 NODES: ARE ALL 4 NODES = RED? F YES, THEN DO: RUN END OF GAME 5 SHOW THE PREVIOUS top nodes each advance one color. GAME UGST 7 Game 9 is the same as Game 8, with one additional rule: If the top node and the 2nd previous top node will remain unchanged. Only the previous top node will advance one color. The selected top node and the previous top node advance one color. If a player returns to a node that he had just previously visited, all four nodes go blank!	F "GAME" = 4, SET "GAME" = 5.		Q	1.0	4			
PATT CH READING SET PREVIOUS SWITCH CLOSURE VALUE. READ CURRENT SWITCH CLOSURE. DO WE HAVE A NEW SWITCH CLOSURE? F YES, THEN DO: DEBOUNCE NEW SWITCH CLOSURE MAKE KEY CLICK SOUND CONVERT 1/0 VALUE TO KEY ID //ALUE. GAME LOGIC WHICH GAME ARE WE IN? GAME LOGIC FOR GAMES 1-5. END OF GAME CHECK NSPECT THE STATE OF ALL 4 NODES: ARE ALL 4 NODES = RED? F YES, THEN DO: RUN END OF GAME 5 SHOW TERNALLY each advance one color. Game 9 is the same as Game 9 is the same as Game 9, with one additional rule: If the top node and the 2nd previous top node will remain unchanged. Only the previous top node will advance one color. 10 0 1 2 The selected top node and the previous top node advance one color. 10 0 1 2 The selected top node and the previous top node advance one color. 10 0 1 2 The selected top node and the previous top node advance one color. 10 0 1 2 The selected top node and the previous top node advance one color. 10 0 1 2 The selected top node and the previous top node advance one color. 10 0 1 2 The selected top node and the previous top node advance one color. 10 0 1 2 The selected top node and the previous top node advance one color. 10 0 1 2 The selected top node and the previous top node advance one color. 10 0 1 2 The selected top node and the previous top node advance one color. 10 0 1 2 The selected top node advance one color. 10 0 1 2 The selected top node and the previous top node advance one color. 10 0 1 2 The selected top node and the previous top node advance one color. 10 0 1 2 The selected top node and the previous top node advance one color. 10 0 1 2 The selected top node and the previous top node advance one color. 10 0 1 2 The selected top node and the previous top node advance one color. 10 0 1 2 The selected top node and the previous top node and the prev	UMP TO PRE-EXEC.		0	10	. •			
GET PREVIOUS SWITCH CLOSURE VALUE. READ CURRENT SWITCH CLOSURE. BY YES, THEN DO: DEBOUNCE NEW SWITCH CLOSURE MAKE KEY CLICK SOUND CONVERT 1/0 VALUE TO KEY ID ARE ALL 4 NODES: ARE ALL 4 NODES: RUN END OF GAME '= 5, THEN DO: RUN END OF GAME 5 SHOW TISE POLY ASSOCIATION Additional rule: If the top node additional rule: If the top node and the previous top node will remain unchanged. Only the previous top node will advance one color. The selected top node and the previous top node will remain unchanged. AND OF CONVERT 1/0 VALUE. ASSOCIATION OF CONVERT 1/0 VALUE.	SWITCH READING_							
READ CURRENT SWITCH CLOSURE. DO WE HAVE A NEW SWITCH CLOSURE? F YES, THEN DO: DEBOUNCE NEW SWITCH CLOSURE MAKE KEY CLICK SOUND CONVERT I/O VALUE TO KEY ID VALUE. GAME LOGIC WHICH GAME ARE WE IN? GAME LOGIC FOR GAMES 1-5. GAME COOLOR. WHICH GAME CHECK NSPECT THE STATE OF ALL 4 NODES: ARE ALL 4 NODES = RED? F YES, THEN DO: RUN END OF GAME 5 SHOW TERNALLY RISE DOO. GAME S, with one additional rule: If the top node and the 2nd previous top node and the 2nd previous top node will remain unchanged. Only the previous top node will advance one color. The selected top node and the previous top node advance one color. If a player returns to a node that he had just previously visited, all four nodes go blank!	GET PREVIOUS SWITCH CLOSURE VALUE.		0	0.1	1			
DO WE HAVE A NEW SWITCH CLOSURE? F YES, THEN DO: AKE KEY CLICK SOUND CONVERT I/0 VALUE TO KEY ID AALUE. GAME LOGIC WHICH GAME ARE WE IN? GAME LOGIC FOR GAMES 1-5. BAND OF GAME CHECK NSPECT THE STATE OF ALL 4 NODES: ARE ALL 4 NODES = RED? F YES, THEN DO: RUN END OF GAME 5 SHOW Additional rule: If the top node and the 2nd previous top node will remain unchanged. Only the previous top node will advance one color. In the state of provious top node and the previous top no		35	9	U I	1 .			
F YES, THEN DO: DEBOUNCE NEW SWITCH CLOSURE MAKE KEY CLICK SOUND CONVERT I/O VALUE TO KEY ID /ALUE. GAME LOGIC WHICH GAME ARE WE IN? GAME LOGIC FOR GAMES 1-5. GAME LOGIC FOR GAMES 1-5. NSPECT THE STATE OF ALL 4 NODES: ARE ALL 4 NODES = RED? F YES, THEN DO: F "GAME" = 5, THEN DO: RUN END OF GAME 5 SHOW TO Provious top node and the previous top node will advance one color. 10 0 1 2 The selected top node and the previous top node will advance one color. The selected top node and the previous top node will advance one color. The selected top node and the previous top node will advance one color. The selected top node and the previous top node and								
DEBOUNCE NEW SWITCH CLOSURE MAKE KEY CLICK SOUND CONVERT 1/0 VALUE TO KEY ID ALUE. GAME LOGIC WHICH GAME ARE WE IN? GAME LOGIC FOR GAMES 1-5. GAME LOGIC FOR GAMES 1-5. AND OF GAME CHECK NSPECT THE STATE OF ALL 4 NODES: ARE ALL 4 NODES = RED? F "GAME" = 5, THEN DO: RUN END OF GAME 5 SHOW TERNALLY RUS E DO. A0 A0 A0 A0 A0 A0 A0 A0 A0 A								
MAKE KEY CLICK SOUND CONVERT I/O VALUE TO KEY ID /ALUE. /AL								
CONVERT I/0 VALUE TO KEY ID /ALUE. /ALUE. SAME LOGIC WHICH GAME ARE WE IN? GAME LOGIC FOR GAMES 1-5. GAME COGIC FOR GAMES 1-5. END OF GAME CHECK NSPECT THE STATE OF ALL 4 NODES: ARE ALL 4 NODES = RED? F YES, THEN DO: F "GAME" = 5, THEN DO: RUN END OF GAME 5 SHOW STERNALLY Will remain unchanged. Only the previous top node will advance one color. If a player returns to a node that he had just previously visited, all four nodes go blank!								
VALUE. GAME LOGIC WHICH GAME ARE WE IN? GAME LOGIC FOR GAMES 1-5. END OF GAME CHECK NSPECT THE STATE OF ALL 4 NODES: ARE ALL 4 NODES = RED? F ''GAME'' = 5, THEN DO: RUN END OF GAME 5 SHOW TERNALLY STERNALLY Only the previous top node will advance one color. 10 0 1 2 The selected top node and the previous top node advance one color. If a player returns to a node that he had just previously visited, all four nodes go blank!								
AGAME LOGIC WHICH GAME ARE WE IN? GAME LOGIC FOR GAMES 1-5. SPIND OF GAME CHECK NSPECT THE STATE OF ALL 4 NODES: ARE ALL 4 NODES = RED? F YES, THEN DO: F "GAME" = 5, THEN DO: RUN END OF GAME 5 SHOW THE STATE OF GAME 5 SHOW THE STATE OF ALL 4 NODES: 45 ASSOCIATED THE STATE OF ALL 4 NODES: Advance one color. If a player returns to a node that he had just previously visited, all four nodes go blank! TERNALLY RUS E DO: The selected top node and the previous top node and the previ		40						
WHICH GAME ARE WE IN? GAME LOGIC FOR GAMES 1-5. END OF GAME CHECK NSPECT THE STATE OF ALL 4 NODES: ARE ALL 4 NODES = RED? F 'YES, THEN DO: RUN END OF GAME 5 SHOW TERNALLY USE DO. 10 0 1 2 The selected top node and the previous top node advance one color. If a player returns to a node that he had just previously visited, all four nodes go blank!								
GAME LOGIC FOR GAMES 1-5. END OF GAME CHECK NSPECT THE STATE OF ALL 4 NODES: ARE ALL 4 NODES = RED? F YES, THEN DO: RUN END OF GAME 5 SHOW TERNALLY USE DO: 10 0 1 2 The selected top node and the previous top node advance one color. If a player returns to a node that he had just previously visited, all four nodes go blank!								
the previous top node advance one color. If a player returns to a node that he had just previously visited, all four nodes go blank!			4.0					
NSPECT THE STATE OF ALL 4 NODES: ARE ALL 4 NODES = RED? F YES, THEN DO: F "GAME" = 5, THEN DO: UN END OF GAME 5 SHOW ETERNALLY ETERNALLY STORY Advance one color. If a player returns to a node that he had just previously visited, all four nodes go blank!			10	0.1	2			
ARE ALL 4 NODES = RED? F YES, THEN DO: F "GAME" = 5, THEN DO: RUN END OF GAME 5 SHOW ETERNALLY FIRST POO.								
ARE ALL 4 NODES = RED? F YES, THEN DO: F "GAME" = 5, THEN DO: RUN END OF GAME 5 SHOW TERNALLY THEN ALLY	NSPECT THE STATE OF ALL 4 NODES:	15						
F "GAME" = 5, THEN DO: previously visited, all four nodes go blank! EVEN END OF GAME 5 SHOW four nodes go blank! EVEN END OF GAME 5 SHOW four nodes go blank!	ARE ALL 4 NODES = RED?	43						
F "GAME" = 5, THEN DO: previously visited, all four nodes go blank! ETERNALLY ET SE DO:	F YES, THEN DO:							
RUN END OF GAME 5 SHOW four nodes go blank! ETERNALLY ETERNALLY ETERNALLY								
ETERNALLY ELSE DO:	RUN END OF GAME 5 SHOW					four nodes go blank!		
ELSE DO:	ETERNALLY							
	ELSE DO:							
	RUN NORMAL END OF GAME SHOW	50	Refer	ring to	FIG. 8, the	circuit of FIG. 5 is modi		

Referring to FIG. 7, in an alternate embodiment two additional control switches labeled A and B are added for expanding the number of games (up to 16 games). A combination of switches A and B together with the position sensor switch 38, indicating which node is the top node, is used to inform the microprocessor 34 which games is to be played. As an example, rules for 10 games and how each of the games is selected when the power switch is turned on are explained in the following table;

FOR 5 SECONDS.

SET FLAG: GAME HAS ENDED.

	S	WITCHES	<u> </u>
GAME	A-B	TOP NODE	RULES
1	00	İ	(DEMONSTRATION GAME)

Referring to FIG. 8, the circuit of FIG. 5 is modified by connecting switches A and B to microprocessor 34 as shown.

Other embodiments are within the following claims. For example, the puzzle may be expanded to a pentahedron or more sided figures having different colored lights at each of its vertexes. It may also be reduced to a planar board having groups of different colored lights arbitrarily located on the face of the board. The number of different colored lights at each node may be increased to four or more differently colored LEDs, and the color state at each node may be defined by a different color state vector. More than one light may be illuminated at each node to increase the complexity of play. The puzzle may also be equipped with a synthesizer for producing words or music at the completion of a game.

What is claimed is:

1. An electronic puzzle comprising:

- (a) a puzzle body including a plurality of lightable positions, each lightable position capable of being illuminated in a plurality of colors;
- (b) a first digital circuit for establishing color state vectors defining patterns of color changes for each lightable position, a single color state vector being assigned to a game being played;
- (c) a second digital circuit for establishing one or more game rules, each game rule defining the application of one of said color state vectors, a single game rule being assigned to a game being played; and
- (d) a third circuit, interconnected with said first digital circuit and said second digital circuit, for changing the color, if required by the game rule, of at least one of said lightable positions;
- whereby in response to a manipulation of said puzzle body to bring one of said lightable positions into a reference plane said first digital circuit, said second 20 digital circuit, and said third circuit cooperate to change the color of at least one of said lightable positions to the next color, if required by the game rule, said next color being determined by the color state vector of the game being played and by the ²⁵ game rule of said game being played.
- 2. The electronic puzzle of claim 1 wherein each of said plurality of lightable positions comprises a group of differently colored lights.
- 3. The electronic puzzle of claim 2 wherein each of said groups of differently colored lights comprises three differently colored LEDs.
- 4. The electronic puzzle of claim 3 wherein the LEDs are red, green, and yellow.
- 5. The electronic puzzle of claim 1 wherein said puzzle body comprises four panels assembled to form a tetrahedron that defines one of said lightable positions at each of its four vertexes.
- 6. The electronic puzzle of claim 5 wherein said third 40 circuit comprises a position sensor switch for informing said first digital circuit and said second digital circuit of the current position of said tetrahedron.

- 7. The electronic puzzle of claim 6 wherein said position sensor switch comprises:
 - a housing having a cavity defining four positions, wherein each of the four positions corresponds to one of said vertexes;
 - a ball located within said cavity that is free to roll to any of said four positions; and
 - conducting pins at at least three of said positions for informing the third circuit of the location of said ball.
- 8. The electronic puzzle of claim 7 wherein said position sensor switch is used to select different games.
- 9. The electronic puzzle of claim 1 further comprising a speaker for emitting audio noises indicating a change in color at each of said lightable positions.
- 10. The electronic puzzle of claim 1 wherein one of said game rules comprises advancing one color in the lightable position brought into the reference plane, according to the color state vector.
- 11. The electronic puzzle of claim 1 wherein one of said game rules comprises advancing one color in the lighable position brought into the reference plane and advancing one color in the lightable position previously occupying the reference plane according to the color state vector.
- 12. The electronic puzzle of claim 1 wherein one of said game rules comprises advancing one color in the lightable position brought into the reference plane according to the color state vector if that lightable position was not brought into the reference plane in the previous two turns.
- 13. The electronic puzzle of claim 1 wherein one of said game rules comprises advancing one color in the color state vector for each of the three lightable positions that are not brought into the reference plane.
- 14. The electronic of puzzle of claim 1 wherein one of said game rules comprises advancing one color in the lightable position brought into the reference plane, advancing one color in the lightable position previously occupying the reference plane and backing up one color in each of the remaining two lightable positions, all in accordance with the color state vector.