MODEL FDG - $95.00

Price Includes Diskette

CROMEMCO DAZZLER GAMES

CROMEMCO INCORPORATED

2400 Charleston Road, Mountain View, California 94043

TABLE OF CONTENTS

CROMEMCO DAZZLER GAMES

Cromemco offers the following computer games on either a 5" diskette (model FDG-S) or 8" diskette (model FDG-L):

CHASE

DAZZLE-DOODLE

DAZZLE-MATION

FOUR-DIMENSIONAL TIC TAC TOE

DOGFIGHT

GOTCHA

KALEIDOSCOPE

LIFE

MAGENTA MARTINI

SPACEWAR

TANKWAR

TRACK

XLIFE

All of these games use the Cromemco Dazzler* interface for your color TV display. The diskettes are designed to be used with Cromemco disk computer systems configured with Z-80 CPU running at 4MHz and with 16K of RAM memory. Several of the games also make use of joystick controls (Cromemco model JS-1)

*Registered trademark of Cromemco

interfaced to the computer using the Cromemco model analog interface.

After loading the game diskette, you will receive the following response on your display console:

```
CROMEMCO DAZZLER GAMES
A.
```

In response to the CDOS prompt "A." you can type "DIR" to get a complete directory of the Dazzler games on your diskette. You can begin execution of any of these games by simply typing the name of the game exactly as it appears in the directory followed by a carriage return. Since directory names are limited to a maximum of eight characters, several of the names are abbreviated, as can be seen in the directory listing. For example, to run Kaleidoscope you will

```
A. KALEIDO
```

To terminate any game simply depress the computer reset switch then depress the carriage return key on your terminal three or four times until you again receive the prompt to select another game

```
CROMEMCO DAZZLER GAMES
A.
```

This manual contains operating instructions for each of the Dazzler games on your diskette. In addition, source code listings for two of the games (GOTCHA and DAZZLE-DOODLE) are given here as illustrative examples of Dazzler programming technique. The source code for GOTCHA was assembled using

the Cromemco CDOS Z-80 assembler while the source code for DAZZLE-DOODLE was assembled by hand.

Please also note that Appendix A details engineering modifications required for REV B and REV C series Dazzlers to assure compatibility with subsequent Cromemco products.

CHASE!

CHASE! is a two-person competitive game using two Cromemco JS-1 joystick consoles. The game display is generated on a color TV using the Cromemco Dazzler TV interface.

The game begins with a cross and a circle in opposite corners of the playing field. One joystick controls the cross, the other the circle. The object is for the cross to catch the circle. The game score is automatically kept as is the time remaining. The player controlling the cross gains a point every time he catches the circle. When the time for the first half of the game is exhausted, the second half can be entered by depressing button number two on the joystick console. During the second half of the game, the second player now controls the cross and gains points as he catches the circle (now controlled by the first player). At the end of the game, the score of both players is displayed on the screen.

The game is begun by pushing switch number one on the joystick console. The game can be restarted at any time by pushing button number four on the joystick console

DAZZLE DOODLE

The Cromemco Dazzle-Doodle software is designed to allow the user to draw full-color pictures on the screen of an ordinary color TV under joystick control. The hardware required is a Cromemco JS-1 joystick console, a Cromemco D+7A interface for the joystick console, and a Cromemco TV Dazzler for the TV display interface. When using the Cromemco Dazzler games diskette, simply type "DOODLE" to begin execution of this program.

To use the Dazzle-Doodle program simply depress either button 2, 3, or 4 on the joystick console and begin "drawing" with the joystick. Button 2 is for red, 3 gives green, and button 4 is for blue. More than one of these buttons may be depressed for a combination of colors. Button 1 is used to erase the picture. The screen may also be filled with color by depressing button 1 while at the same time depressing one or more of buttons 2, 3, or 4. A source listing of the program is given below:

Address	Contents	Comments
000 000	303 JMP	Jump to main program (optional
000 001	000	instruction for execution to begin
000 002	002	at zero in memory).
002 000	076 MVI , A	Main program begins here.
001	204	
002	323 OUT	Out to Dazzler to display picture
003	016	from 2K to 4K in memory.
004	076 MVI , A	
005	060	
006	323 OUT	Out to Dazzler for 64X64 mode
007	017	full color.
010	333 IN	Input from JS-1 console switches.
011	030	
012	057 CMA	
013	366 ORI	
014	020	
015	037 RAR	
016	107 MOV B , A	Save in B register state of switches.
017	332 JC	Jump if switch #1 is depressed.

```
002 020          146
    021          002
    022          333 IN          Input joystick x-axis.
    023          031
    024          306 ADI
    025          100
    026          362 JP          Jump if voltage within range.
    027          033
    030          002
    031          006 MVI B       Otherwise put zeros in B register
    032          000             to prevent screen write.
    033          037 RAR
    034          137 MOV E , A   Put X displacement in E.
    035          333 IN          Input joystick y-axis.
    036          032
    037          306 ADI
    040          100
    041          362 JP          Jump if voltage within range.
    042          046
    043          002
    044          006 MVI B       Otherwise put zeros in B register
    045          000             to prevent screen write.
    046          037 RAR
    047          057 CMA
    050          127 MOV D , A   Put Y displacement in D register.
    051          000 NOP
    052          000 NOP
    053          000 NOP
    054          346 ANI         The following instructions are used
    055          077             to generate a 64X64 Dazzler address
    056          147 MOV H , A   in HL given that the X,Y coordinates
    057          346 ANI         are in DE.
    060          040
    061          204 ADD H
    062          147 MOV H , A
    063          173 MOV A , E
    064          346 ANI
    065          040
    066          264 ORA H
    067          017 RRC
    070          017 RRC
    071          017 RRC
    072          017 RRC
    073          147 MOV H , A
    074          173 MOV A , E
    075          017 RRC
    076          346 ANI
    077          017
    100          157 MOV L , A
    101          174 MOV A , H
    102          346 ANI
    103          360
    104          265 ORA L
    105          157 MOV L , A
    106          174 MOV A , H
```

Address	Contents	Comments
002 107	346 ANI	
110	007	
111	366 ORI	This sets the addresses of picture
112	010	between 2K and 4K in memory.
113	147 MOV H , A	
114	116 MOV C , M	Fetch data byte from memory.
115	173 MOV A , E	
116	017 RRC	Put LSB of X in carry.
117	332 JC	Jump to write in upper nybble
120	132	of data byte.
121	002	
122	076 MVI A	
123	017	
124	240 ANA B	Strip color information from B.
125	261 ORA C	OR with present memory data.
126	167 MOV M , A	Replace with new memory data.
127	303 JMP	Jump back to the beginning.
130	004	
131	002	
132	076 MVI A	
133	017	
134	240 ANA B	Strip color information from B.
135	007 RLC	Shift into upper half of byte.
136	007 RLC	
137	007 RLC	
140	007 RLC	
141	261 ORA C	OR with present memory data.
142	167 MOV M , A	Replace with new memory data.
143	303 JMP	Jump back to the beginning.
144	004	
145	002	
146	041 LXI H	Start of memory clear routine.
147	000	Address of first byte
150	010	of Dazzler picture.
151	076 MVI A	
152	017	
153	240 ANA B	Strip color from B.
154	117 MOV C , A	
155	007 RLC	Copy in upper half of byte.
156	007 RLC	
157	007 RLC	
160	007 RLC	
161	261 ORA C	
162	117 MOV C , A	
163	161 MOV M , C	Store new data in memory.
164	043 INX H	Increment memory location.
165	174 MOV A , H	
166	376 CPI	Check to see if at 4K.
167	020	
170	322 JNC	Jump if through.
171	004	
172	002	
173	303 JMP	Otherwise loop for new location.
174	163	
175	002	

DAZZLE-MATION

General Description

The Dazzlemation program, written by Steve Dompier, is designed as an aid in the production of animated DAZZLER displays. The "Magenta Martini" animation is provided as one example of the type of animation possible using the Dazzlemation program.

Once the Dazzlemation program is entered into your computer, the animation sequence can be entered from your keyboard or paper tape reader. CONTROL R on your keyboard is the command to begin the display of the animated sequence.

Composing an Animation Sequence

Animation sequences are composed using your keyboard. First you should be familiar with these Dazzlemation Executive Commands:

CONTROL Q - Begin a new sequence.
CONTROL B - Stop cursor from flashing.
CONTROL C - Delete cursor.
CONTROL R - Run.
CONTROL X - Stop and return to executive.

After depressing CONTROL Q on your keyboard to begin a new sequence, the sequence is drawn on your TV screen using keyboard entries. As you deposit the sequence, the direction of cursor motion is first set by these commands:

N - Up
M - Down
COMMA - Left
PERIOD - Right

For diagonal moves, hold down the SHIFT key while depressing N or M and then COMMA or PERIOD.

To set the intensity of each point as you deposit it in sequence use one of these two commands:

H - High intensity
L - Low intensity

Now you are ready to enter the animation sequence. The color of each point entered in the sequence is determined by which key is used to deposit that element:

R - Red
G - Green
Y - Yellow
W - White
B - Blue
P - Purple
C - Cyan
RUBOUT - Black
\+ - Pause

The following commands may be inserted in a program sequence:

CONTROL Z Clear screen, maintain trace memory when the screen is rewritten.

ESCAPE Clear screen, inhibit trace memory.

CONTROL S Programmed stop point.

For teletypes CONTROL - SHIFT K provides the ESCAPE function. For most other terminals, it is CONTROL SEMI - COLON.

After the Dazzlemation sequence is deposited from the keyboard, the cursor should be positioned at the point relative to the original drawing where the original sequence should be redrawn in the animation. To start the animation depress CONTROL R. As long as there is no CONTROL S in the sequence, the sequence will be redrawn on the screen again and again, the speed of execution being set by the sense switches. Eash subsequent drawing in the animation will be displaced from the previous one precisely by the same amount the cursor was displaced from the original drawing at the time of execution. Note that CONTROL R clears CONTROL S, so should you want just a single execution of your sequence, CONTROL S must be set for each execution.

Once you have composed a Dazzlemation animation you may wish to save the animation on paper tape. The special Dazzlemation command SHIFT P can be used to punch your Dazzlemation sequence on paper tape. When using the SHIFT P command, be sure that a CONTROL S is used to terminate your sequence.

The game of "Dogfight" is a two-player game using two Cromemco JS-1 joystick modules. Each player uses a joystick and four joystick pushbuttons to control his fighter plane on the Dazzler display.

The airplane's throttle is controlled by buttons 3 and 4 on the joystick console. Push both buttons 3 and 4 for maximum thrust. The joystick is the airplane's elevator control. Once flying speed is attained (by holding down buttons 3 and 4) pull back on the joystick to become airborn. An aeleron roll can be achieved by depressing buttons 2 on the joystick console. Machine gun fire is initiated by pressing button number 1.

The purpose of the game is to shoot the opponent out of the sky. You gain a point every time that a hit is scored. The dynamics of flight are carefully simulated in this game so that you must maintain flying speed to stay aloft.

The first player to gain 21 points is the winner of the game. At this point the game can be restarted by pressing all four buttons on each joystick console.

FOUR DIMENSIONAL TIC-TAC-TOE

Four dimensional tic-tac-toe is a logical extension of the familiar two dimensional tic-tac-toe. Once the basic concept of converting four dimensions into two is grasped, the game is easy to play.

Imagine first a four square by four square (16 squares) playing board. Stack three identical boards on top of the first such that each square on a board is the bottom of a cube. This is a three dimensional tic-tac-toe board, a cube composed of 64 small cubes. Any sequence of four cubes that spans the cube from one surface to another in a straight line is a winning combination. To visualize these combinations more easily, imagine that instead of four horizontal boards, there were four vertical boards, or two boards (slightly stretched) placed between the edges of the cube, forming an "X". All sequences of squares that are winning combinations on the boards in two dimensions are also winning combinations in three dimensions. There are no other winning combinations in three dimensional tic-tac-toe. The same can be tried in two dimensions, using a one dimensional board to find the winning combination.

Before proceeding to four dimensions, the three dimensional playing board must be represented in two dimensions. This is simple to do by unstacking the four two dimensional boards which compose the three dimentional board, and lying them top to bottom in a column. Instead of trying to visualize a four dimensional tic-tac-toe board, it is much easier to convert it into three dimensions. A four dimensional board becomes four three dimensional boards side by side in a row. The three dimensional

representation can be compressed into two dimensions by unstacking the four two dimensional boards each three dimensional board is composed of, and placing them in columns. That leaves sixteen boards arranged four by four, each of which contains sixteen squares arranged four by four.

To find all the combinations use the two dimensional representation of a three dimensional board (four two dimensional boards in a column). Superimpose it on the four rows, four columns, and (with each board turned 45° the two diagonals of two dimensional boards which make up the four dimensional board. Each sequence of four squares that corresponds to a winning combination on the superimposed three dimensional board is also a winning combination in four dimensional tic-tac-toe. There are no other winning combinations.

The four dimensional board represented in two dimensions looks like a big two dimensional board whose squares are smaller two dimensional boards. The similarity is very useful. Each small board has ten winning combinations. Each combination can be represented by a sequence of four squares. The sequence can be given in two different directions. The same combinations can be used to specify a sequence of small boards within the big board. Two sequences, specifying a board and a square within that board are combined to specify a sequence of four squares in four dimensions.

Using the winning combinations of two dimensional tic-tac-toe, specified by sequences of squares in both directions, any two sequences may be combined to give a winning combination in four dimensional tic-tac-toe. In addition to these combinations, each board is a tic-tac-toe game in itself, and the square's position can remain constant while the boards follow some winning sequence.

How to Play the Game

The four dimensional tic-tac-toe program is a game in which one person plays against the computer. The player makes his move by selecting the number which corresponds to the square he wishes to occupy. The computer will show this move on a color television display controlled by a Cromemco Dazzler. The computer then makes its move, which is shown on the television display.

The first 1.75K of memory contains the program. The next .25K of memory is reserved for the program stack. The display is located from 2K to 2.5K (this must be static RAM). A .5K workspace fills the rest of the 3K. All of the non-program memory must be RAM.

The program starts from location 0000. When started, it disables the interrupt system, and turns on the Dazzler display. It then asks if it can play first. If a "Y" is typed, the computer will make the first move. If any other character is typed, then the player may make the first move. A playing board is then constructed on the display, and the computer is ready to accept the player's move. If the computer made the first move, that move will be displayed. The computer will then accept the player's move.

The player enters his move by entering from a ASCII keyboard the number of the square he wishes to take (see Figure 1). The computer will make sure the square has not been taken. The computer is then ready to accept another input. If the square is unoccupied, the square's number is output to the lights and is marked in yellow on the display. If the player is unsatisfied with his move, he may type a space. The computer will extinguish the yellow square, and wait for another move to be input. If the player is satisfied with his move, he may type a return and the

computer will change the yellow square to the player's color. The computer will then make its move. When that move appears on the screen and on the lights, the computer is ready to accept the player's next move. If the move appears in white, the game is a tie, and the program jumps to the beginning.

If the computer discovers that the player has four squares in a row, it will turn the winning squares green and jump to the beginning of the program. If the computer finds that it will have four in a row after it makes its move, it turns the possible winning squares yellow. If the player does not also have four squares in a row, the computer will turn the yellow squares white, output the winning square's number to the lights, and jump to the beginning of the program. If the player does have four in a row, the computer will turn those squares green and jump to the beginning of the program.

How the Program Operates

The computer makes its moves by examining each winning combination of four squares. The computer determines which of nine categories each combination fits into. The nine categories are: all the squares empty; one, two, three, or four squares occupied by the player and the rest empty; one, two, or three squares occupied by the player and the rest empty; or some squares occupied by the player and some by the computer. In the later case, the computer does nothing and continues on to the next combination. In the cases where zero, one, or two squares are occupied, the computer uses two words of memory corresponding to each square. This forms a sixteen bit word for each square. If all the squares are empty, then, for each square, the computer adds one to the

least significant word in memory corresponding to the empty square. If the computer or the player occupies one square, and the rest are empty, then for each square the computer adds one to the most significant word in memory which corresponds to the empty square. If the player has two squares, the computer adds 10H to the most significant words in memory corresponding to the two empty squares. The computer will not add more than 30H to any one square. If the computer has two squares, it adds 40H to the most significant words corresponding to the two empty squares. The computer will not add more than 0C0H to any one square.

In the cases where three squares are occupied, the computer remembers which square was empty. If the player has three squares, then the computer must block him by taking the empty square. If the computer has three squares, then it will remember the empty square it needs to win, and forget about blocking. It will turn all four squares yellow, and continue on to see if the player has won.

If the player has four squares in a row, then the computer stops looking at winning combinations. It will turn the winning squares green and jump to the beginning of the program.

After all winning combinations have been looked at, the computer must decide which square it wants to take. If the player has won, the computer will not reach this point. It first checks to see if it has already chosen a winning square. If it has, it will output that square on the lights, change the yellow squares to red, and jump to the beginning of the program. If it has chosen a blocking square, it will output that square to the lights and display and wait for the player's next move.

If no square has already been chosen, the computer must look at the words of memory which correspond to the squares. The computer will pick the square whose two words of memory contain the greatest sixteen bit value. The selected square is output on the lights and display, and the computer waits for the player's move. If the game turns out to be a tie, the computer will output its move in red and jump to the beginning of the program.

Note- In addition to a Cromemco Dazzler to generate the color TV display, a CRT terminal or Teletype is required to play 4D TIC TAC TOE. Messages from the computer appear on the teletype or CRT display while the keyboard is used for input.

THE BOARD

00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	
10	11	12	13	14	15	16	17	18	19	1A	1B	1C	1D	1E	1F
20	21	22	23	24	25	26	27	28	29	2A	2B	2C	2D	2E	2F
30	31	32	33	34	35	36	37	38	39	3A	3B	3C	3D	3E	3F
40	41	42	43	44	45	46	47	48	49	4A	4B	4C	4D	4E	4F
50	51	52	53	54	55	56	57	58	59	5A	5B	5C	5D	5E	5F
60	61	62	63	64	65	66	67	68	69	6A	6B	6C	6D	6E	6F
70	71	72	73	74	75	76	77	78	79	7A	7B	7C	7D	7E	7F
80	81	82	83	84	85	86	87	88	89	8A	8B	8C	8D	8E	8F
90	91	92	93	94	95	96	97	98	99	9A	9B	9C	9D	9E	9F
A0	A1	A2	A3	A4	A5	A6	A7	A8	A9	AA	AB	AC	AD	AE	AF
B0	B1	B2	B3	B4	B5	B6	B7	B8	B9	BA	BB	BC	BD	BE	BF
C0	C1	C2	C3	C4	C5	C6	C7	C8	C9	CA	CB	CC	CD	CE	CF
D0	D1	D2	D3	D4	D5	D6	D7	D8	D9	DA	DB	DC	DD	DE	DF
E0	E1	E2	E3	E4	E5	E6	E7	E8	E9	EA	EB	EC	ED	EE	EF
F0	F1	F2	F3	F4	F5	F6	F7	F8	F9	FA	FB	FC	FD	FE	

GOTCHA!

GOTCHA! pits you against your opponent on a colorful game in which you try to occupy the playing field while blocking your opponent's access.

RED starts in the upper left hand corner of the board; BLUE in the lower right. At START, RED begins travelling downward, leaving a solid red line, and BLUE travels upward, leaving a blue line. Direction is altered by the Cromemco Joystick controls. If a player hits a boundary, himself, or the other player's line, his opponent scores a point. The game continues until nine points have been scored by one player. Pushbutton 1 starts the game and pushbutton 3 resets it to zero.

Pushbutton 2 speeds up the progress of the lines and can be used strategically against your opponent.

A source listing of the GOTCHA! program is given on the following pages.

```
CROMEMCO CDOS Z80 ASSEMBLER V. 1. 4A
GOTCHA, GOTCHA, GOTCHA !!!!!!!!!!!!!!

                        0002 ;
                        0003 ;THIS IS THE EXCITING GAME OF GOTCHA !!!!1!!
                        0004 ;
0000                    0005            ORG     0
0000 F3                 0006 START:     DI
0001 315B03             0007            LD      SP,STACK
0004 CDBE00             0008            CALL    INIT                ;INIT PROGRAM
0007 CDFB02             0009            CALL    WBONE               ;WAIT FOR BUTTON ONE
000A 2A1403             0010 MAIN:      LD      HL,(POS1)           ;POSITION, PLAYER 1
000D EB                 0011            EX      DE,HL               ;PUT IN DE
000E 3E09               0012            LD      A,P1COLR            ;COLOR, PLAYER 1
0010 321A03             0013            LD      (NCOLOR),A
0013 3A1803             0014            LD      A,(DIR1)            ;DIRECTION, PLAYER 1
0016 CD1602             0015            CALL    MOVDOT              ;MAKE MOVE
0019 EB                 0016            EX      DE,HL
001A 221403             0017            LD      (POS1),HL           ;STORE NEW XY
001D EB                 0018            EX      DE,HL
001E 010000             0019            LD      BC,0                ;INIT FLAGS
0021 CA2D00             0020            JP      Z,MAN300            ;CAN MOVE
0024 C5                 0021            PUSH    BC                  ;SAVE BC
0025 3E0F               0022            LD      A,15                ;PAINT DOT WHITE
0027 CDCB01             0023            CALL    PUTCOL
002A C1                 0024            POP     BC                  ;RESTORE BC
002B 0601               0025            LD      B,1                 ;SAY CAN'T MOVE
002D C5                 0026 MAN300:    PUSH    BC                  ;SAVE FLAGS
002E 2A1603             0027            LD      HL,(POS2)           ;POSITION, PLAYER 2
0031 EB                 0028            EX      DE,HL               ;PUT IN DE
0032 3E0C               0029            LD      A,P2COLR            ;COLOR, PLAYER 2
0034 321A03             0030            LD      (NCOLOR),A
0037 3A1903             0031            LD      A,(DIR2)            ;DIRECTION, PLAYER 2
003A CD1602             0032            CALL    MOVDOT              ;MAKE MOVE
003D EB                 0033            EX      DE,HL
003E 221603             0034            LD      (POS2),HL           ;SAVE NEW XY
0041 EB                 0035            EX      DE,HL
0042 C1                 0036            POP     BC
0043 CA4F00             0037            JP      Z,MAN320            ;CAN MOVE
0046 C5                 0038            PUSH    BC                  ;SAVE BC
0047 3E0F               0039            LD      A,15                ;PAINT DOT WHITE
0049 CDCB01             0040            CALL    PUTCOL
004C C1                 0041            POP     BC                  ;RESTORE BC
004D 0E01               0042            LD      C,1                 ;SAY CAN'T MOVE
004F 78                 0043 MAN320:    LD      A,B                 ;GET 1ST FLAG
0050 A7                 0044            AND     A
0051 C26900             0045            JP      NZ,MAN400           ;PLAYER 1 HIT
0054 79                 0046            LD      A,C
0055 A7                 0047            AND     A
0056 CA9A00             0048            JP      Z,MAN600            ;NOBODY HIT
0059 CDA600             0049            CALL    CPOS                ;CHECK IF RAN INTO EACH OTH
005C CA7B00             0050            JP      Z,MAN450            ;YES
005F 211203             0051            LD      HL,NUM1             ;PT TO SCORE 1
0062 34                 0052            INC     (HL)
0063 CD0401             0053            CALL    DPLAY1              ;DISPLAY NEW SCORE
```

```
CROMEMCO CDOS Z80 ASSEMBLER V. 1. 4A
GOTCHA, GOTCHA, GOTCHA !!!!!!!!!!!!!!

0066 C37B00          0054            JP      MAN450
                     0055 ;
0069 79              0056 MAN400:    LD      A, C            ;GET 2ND FLAG
006A A7              0057            AND     A
006B C27B00          0058            JP      NZ, MAN450      ;BOTH HIT
006E CDA600          0059            CALL    CPOS            ;Q. RAN INTO EACH OTHER
0071 CA7B00          0060            JP      Z, MAN450       ;YES
0074 211303          0061            LD      HL, NUM2        ;PT TO SCORE 2
0077 34              0062            INC     (HL)
0078 CDF600          0063            CALL    DPLAY2          ;DISPLAY NEW SCORE
007B CDB300          0064 MAN450:    CALL    SONG            ;PLAY A SONG
007E 3A1203          0065 MAN500:    LD      A, (NUM1)       ;GET SCORE 1
0081 FE09            0066            CP      9
0083 CA8E00          0067            JP      Z, ENDRND       ;END OF ROUND
0086 3A1303          0068            LD      A, (NUM2)       ;GET SCORE 2
0089 FE09            0069            CP      9
008B C29400          0070            JP      NZ, MAN550      ;NOT END OF ROUND
008E CDFB02          0071 ENDRND:    CALL    WBONE           ;WAIT FOR BUTTON 1
0091 CDBE00          0072            CALL    INIT            ;RE-INIT PROGRAM
0094 CDD100          0073 MAN550:    CALL    DAZWRT          ;REWRITE SCREEN
0097 C30A00          0074            JP      MAIN            ;LOOP
                     0075 ;
009A CDD202          0076 MAN600:    CALL    WAIT            ;WAIT A WHILE
009D CD3C02          0077            CALL    GNEW1           ;GET NEW DIRECTIONS
00A0 CD5502          0078            CALL    GNEW2
00A3 C30A00          0079            JP      MAIN
                     0080 ;
                     0081 ;COMPARE BOTH POSITIONS
                     0082 ;OUTPUT - Z SET IF EQUAL
                     0083 ;
00A6 2A1403          0084 CPOS:      LD      HL, (POS1)      ;POSITION, PLAYER 1
00A9 EB              0085            EX      DE, HL
00AA 2A1603          0086            LD      HL, (POS2)      ;POSITION, PLAYER 2
00AD 7A              0087            LD      A, D
00AE BC              0088            CP      H
00AF C0              0089            RET     NZ              ;NOT EQUAL
00B0 7B              0090            LD      A, E
00B1 BD              0091            CP      L
00B2 C9              0092            RET
                     0093 ;
                     0094 ;PLAY A SONG
                     0095 ;
00B3 21B900          0096 SONG:      LD      HL, SON090
00B6 C3AE02          0097            JP      NOTES
                     0098 ;
00B9 40              0099 SON090:    DB      40H             ;VOLUME
00BA 78              0100            DB      120             ;FREQ. PARM
00BB F401            0101            DW      500             ;DURATION
00BD 00              0102            DB      0               ;END OF TABLE
                     0103 ;
                     0104 ;INITIALIZE PROGRAM
                     0105 ;
```

```
00BE 3E90       0106 INIT:   LD      A,090H
00C0 D30F       0107         OUT     15,A
00C2 3E82       0108         LD      A,[DISPLY SHR 9]+80H
00C4 D30E       0109         OUT     14,A
00C6 3E00       0110         LD      A,0
00C8 321203     0111         LD      (NUM1),A          ;INIT SCORE
00CB 321303     0112         LD      (NUM2),A
00CE CDFD01     0113         CALL    INTJOY            ;INIT JOY STICKS
                0114 ;
                0115 ;WRITE DAZZLER DISPLAY
                0116 ;
00D1 210903     0117 DAZWRT: LD      HL,309H
00D4 221403     0118         LD      (POS1),HL         ;INIT POSITION FOR PLAYER 1
00D7 211C1C     0119         LD      HL,1C1CH
00DA 221603     0120         LD      (POS2),HL         ;INIT POSITION FOR PLAYER 2
00DD 3E02       0121         LD      A,2               ;DIRECTION 1 = DOWN
00DF 321803     0122         LD      (DIR1),A
00E2 3E01       0123         LD      A,1               ;DIRECTION 2 = UP
00E4 321903     0124         LD      (DIR2),A
00E7 210004     0125         LD      HL,DISPLY         ;PT TO DISPLAY
00EA 010002     0126         LD      BC,200H           ;LENGTH
00ED CDEE01     0127         CALL    CLEAR             ;CLEAR DISPLAY AREA
00F0 CD6D01     0128         CALL    BOARD             ;DISPLAY BOARDER
00F3 CD0401     0129         CALL    DPLAY1            ;DISPLAY 1ST SCORE
                0130 ;
                0131 ;DISPLAY 2ND SCORE
                0132 ;
00F6 11001C     0133 DPLAY2: LD      DE,28*256                 ;PT TO POSITION
00F9 3E0C       0134         LD      A,P2COLR          ;GET COLOR
00FB 321A03     0135         LD      (NCOLOR),A
00FE 3A1303     0136         LD      A,(NUM2)          ;GET SCORE
0101 C30F01     0137         JP      DSPNUM            ;DISPLAY NUMBER
                0138 ;
                0139 ;DISPLAY 1ST SCORE
                0140 ;
0104 110001     0141 DPLAY1: LD      DE,100H           ;PT TO POSITION
0107 3E09       0142         LD      A,P1COLR          ;GET COLOR
0109 321A03     0143         LD      (NCOLOR),A
010C 3A1203     0144         LD      A,(NUM1)          ;GET SCORE
                0145 ;
                0146 ;DISPLAY 3X5 DIGIT ON DAZZLER
                0147 ;INPUT - DE CONTAINS TOP,LEFT X,Y FOR NUMBER
                0148 ;        A CONTAINS NUMBER
                0149 ;
010F 213D01     0150 DSPNUM: LD      HL,DNMTAB         ;PT TO DIGIT TABLE
0112 E60F       0151         AND     15
0114 47         0152         LD      B,A
0115 87         0153         ADD     A
0116 80         0154         ADD     B
0117 CDF801     0155         CALL    ADDAHL            ;PT TO CORRECT NUMBER
011A 0E03       0156         LD      C,3               ;COUNTER
011C 0605       0157 DNM300: LD      B,5               ;COUNTER
```

```
CROMEMCO CDOS Z80 ASSEMBLER V. 1. 4A
GOTCHA, GOTCHA, GOTCHA !!!!!!!!!!!!!!

011E 7E          0158           LD      A, (HL)          ;GET BYTE FROM TABLE
011F 17          0159 DNM320:   RLA                      ;GET 1ST BIT
0120 F5          0160           PUSH    AF               ;SAVE AF
0121 3A1A03      0161           LD      A, (NCOLOR)      ;GET COLOR FOR NUMBER
0124 DA2801      0162           JP      C, DNM350        ;DO PUT COLOR THERE
0127 97          0163           SUB     A                ;DO NOT PUT COLOR THERE
0128 CDCB01      0164 DNM350:   CALL    PUTCOL           ;PUT COLOR
012B F1          0165           POP     AF               ;RESTORE AF
012C 1C          0166           INC     E                ;INC Y POSITION
012D 05          0167           DEC     B                ;COUNT DOWN
012E C21F01      0168           JP      NZ, DNM320       ;LOOP
0131 1D          0169           DEC     E
0132 1D          0170           DEC     E
0133 1D          0171           DEC     E
0134 1D          0172           DEC     E
0135 1D          0173           DEC     E
0136 14          0174           INC     D                ;INC X POSITION
0137 23          0175           INC     HL               ;PT TO NEXT BYTE
0138 0D          0176           DEC     C                ;COUNT DOWN
0139 C21C01      0177           JP      NZ, DNM300       ;LOOP
013C C9          0178           RET
                 0179 ;
013D F888F8      0180 DNMTAB:   DB      0F8H, 088H, 0F8H ;ZERO
0140 0000F8      0181           DB      000H, 000H, 0F8H ;ONE
0143 B8A8E8      0182           DB      0B8H, 0A8H, 0E8H ;TWO
0146 A8A8F8      0183           DB      0A8H, 0A8H, 0F8H ;THREE
0149 E020F8      0184           DB      0E0H, 020H, 0F8H ;FOUR
014C E8A8B8      0185           DB      0E8H, 0A8H, 0B8H ;FIVE
014F F8A8B8      0186           DB      0F8H, 0A8H, 0B8H ;SIX
0152 8080F8      0187           DB      080H, 080H, 0F8H ;SEVEN
0155 F8A8F8      0188           DB      0F8H, 0A8H, 0F8H ;EIGTH
0158 E0A0F8      0189           DB      0E0H, 0A0H, 0F8H ;NINE
015B F8A0F8      0190           DB      0F8H, 0A0H, 0F8H ;A
015E F82838      0191           DB      0F8H, 028H, 038H ;B
0161 F88888      0192           DB      0F8H, 088H, 088H ;C
0164 3828F8      0193           DB      038H, 028H, 0F8H ;D
0167 F8A8A8      0194           DB      0F8H, 0A8H, 0A8H ;E
016A F8A0A0      0195           DB      0F8H, 0A0H, 0A0H ;F
                 0196 ;
                 0197 ;DISPLAY BOARDER
                 0198 ;
016D 110600      0199 BOARD:    LD      DE, 6            ;START OF BOARDER
0170 0620        0200           LD      B, 32            ;LENGTH OF BOARDER
0172 3E0A        0201 BRD300:   LD      A, BCOLOR        ;COLOR OF BOARDER
0174 CDCB01      0202           CALL    PUTCOL           ;PUT COLOR
0177 14          0203           INC     D                ;INC X PTR
0178 05          0204           DEC     B                ;COUNT DOWN
0179 C27201      0205           JP      NZ, BRD300       ;LOOP UNTIL DONE
017C 111F00      0206           LD      DE, 31
017F 0620        0207           LD      B, 32            ;LENGTH
0181 3E0A        0208 BRD320:   LD      A, BCOLOR        ;COLOR OF BOARDER
0183 CDCB01      0209           CALL    PUTCOL           ;PUT COLOR
```

```
0186 14          0210          INC     D               ;INC X PTR
0187 05          0211          DEC     B               ;COUNT DOWN
0188 C28101      0212          JP      NZ,BRD320       ;LOOP UNTIL DONE
018B 110700      0213          LD      DE,7
018E 0618        0214          LD      B,24            ;LENGTH
0190 3E0A        0215 BRD340:  LD      A,BCOLOR        ;COLOR OF BOARDER
0192 CDCB01      0216          CALL    PUTCOL          ;PUT COLOR
0195 1C          0217          INC     E               ;INC Y PTR
0196 05          0218          DEC     B               ;COUNT DOWN
0197 C29001      0219          JP      NZ,BRD340       ;LOOP UNTIL THRU
019A 11071F      0220          LD      DE,1F07H
019D 0618        0221          LD      B,24            ;LENGTH
019F 3E0A        0222 BRD360:  LD      A,BCOLOR        ;COLOR OF BOARDER
01A1 CDCB01      0223          CALL    PUTCOL          ;PUT COLOR
01A4 1C          0224          INC     E               ;INC Y PTR
01A5 05          0225          DEC     B               ;COUNT DOWN
01A6 C29F01      0226          JP      NZ,BRD360       ;LOOP
01A9 C9          0227          RET
                 0228 ;
                 0229 ;POINT TO DOT
                 0230 ;INPUT - DE CONTAINS XY
                 0231 ;OUTPUT - HL PTS TO NIBBLE
                 0232 ;           CARRY SET IF TOP NIBBLE
                 0233 ;
01AA 6B          0234 DOTPTR:  LD      L,E             ;GET Y POSITION
01AB 2600        0235          LD      H,0
01AD 29          0236          ADD     HL,HL           ;MULTIPLY BY 16
01AE 29          0237          ADD     HL,HL
01AF 29          0238          ADD     HL,HL
01B0 29          0239          ADD     HL,HL
01B1 7A          0240          LD      A,D             ;GET X POSITION
01B2 1F          0241          RRA                     ;DIVIDE BY 2
01B3 F5          0242          PUSH    AF              ;SAVE CARRY
01B4 CDF801      0243          CALL    ADDAHL          ;ADD TO HL
01B7 010004      0244          LD      BC,DISPLY       ;PT TO DISPLAY
01BA 09          0245          ADD     HL,BC           ;PT TO CORRECT DOT
01BB F1          0246          POP     AF              ;RESTORE CARRY
01BC C9          0247          RET
                 0248 ;
                 0249 ;GET DOT
                 0250 ;INPUT - DE CONTAINS XY
                 0251 ;OUTPUT - A CONTAINS COLOR
                 0252 ;
01BD CDAA01      0253 GETCOL:  CALL    DOTPTR          ;PT TO NIBBLE
01C0 7E          0254          LD      A,(HL)          ;GET BYTE
01C1 D2C801      0255          JP      NC,GCL300       ;BOTTOM NIBBLE
01C4 1F          0256          RRA                     ;GET TOP NIBBLE
01C5 1F          0257          RRA
01C6 1F          0258          RRA
01C7 1F          0259          RRA
01C8 E60F        0260 GCL300:  AND     15
01CA C9          0261          RET
```

```
                        0262 ;
                        0263 ;PUT COLOR
                        0264 ;INPUT - A CONTAINS COLOR
                        0265 ;         DE CONTAINS X,Y POSITION
                        0266 ;
01CB E5                 0267 PUTCOL:  PUSH    HL              ;SAVE REGISTERS
01CC C5                 0268          PUSH    BC
01CD F5                 0269          PUSH    AF              ;SAVE COLOR
01CE CDAA01             0270          CALL    DOTPTR          ;PT TO NIBBLE
01D1 C1                 0271          POP     BC              ;GET COLOR
01D2 78                 0272          LD      A,B             ;MOVE COLOR TO A
01D3 D2E301             0273          JP      NC,PTC400       ;BOTTOM NIBBLE
01D6 17                 0274          RLA                     ;TOP NIBBLE
01D7 17                 0275          RLA
01D8 17                 0276          RLA
01D9 17                 0277          RLA
01DA E6F0               0278          AND     0F0H            ;AND OFF BOTTOM NIBBLE
01DC 47                 0279          LD      B,A             ;SAVE
01DD 7E                 0280          LD      A,(HL)          ;GET DAZZLER BYTE
01DE E60F               0281          AND     15              ;AND OFF TOP NIBBLE
01E0 C3E901             0282          JP      PTC900
                        0283 ;
01E3 E60F               0284 PTC400:  AND     15              ;AND OFF TOP NIBBLE
01E5 47                 0285          LD      B,A             ;SAVE
01E6 7E                 0286          LD      A,(HL)          ;GET DAZZLER BYTE
01E7 E6F0               0287          AND     0F0H            ;AND OFF BOTTOM NIBBLE
01E9 B0                 0288 PTC900:  OR      B               ;COMBINE NIBBLES
01EA 77                 0289          LD      (HL),A          ;PUT IN DISPLAY
01EB C1                 0290          POP     BC              ;RESTORE REGISTERS
01EC E1                 0291          POP     HL
01ED C9                 0292          RET
                        0293 ;
                        0294 ;CLEAR AREA WITH ZERO'S
                        0295 ;INPUT - HL PT TO AREA
                        0296 ;        BC CONTAIN LENGTH
                        0297 ;
01EE 78                 0298 CLEAR:   LD      A,B
01EF B1                 0299          OR      C
01F0 C8                 0300          RET     Z               ;LENGTH = 0
01F1 97                 0301          SUB     A               ;CLEAR A
01F2 77                 0302          LD      (HL),A          ;CLEAR BYTE
01F3 23                 0303          INC     HL              ;PT TO NEXT BYTE
01F4 0B                 0304          DEC     BC              ;COUNT DOWN
01F5 C3EE01             0305          JP      CLEAR           ;LOOP
                        0306 ;
                        0307 ;ADD A TO HL
                        0308 ;
01F8 85                 0309 ADDAHL:  ADD     L
01F9 6F                 0310          LD      L,A
01FA D0                 0311          RET     NC
01FB 24                 0312          INC     H
01FC C9                 0313          RET
```

```
                        0314 ;
                        0315 ;INITIALIZE JOY STICKS
                        0316 ;
01FD DB1A               0317 INTJOY:  IN      A,JOY1UD        ;GET UP/DOWN JOY STICK 1
01FF 2F                 0318          CPL
0200 320E03             0319          LD      (AJ1UD),A       ;ADJUSTMENT
0203 DB19               0320          IN      A,JOY1RL        ;GET RIGHT/LEFT JOY STICK 1
0205 2F                 0321          CPL
0206 320F03             0322          LD      (AJ1RL),A       ;ADJUSTMENT
0209 DB1C               0323          IN      A,JOY2UD        ;GET UP/DOWN JOY STICK 2
020B 2F                 0324          CPL
020C 321003             0325          LD      (AJ2UD),A       ;ADJUSTMENT
020F DB1B               0326          IN      A,JOY2RL        ;GET RIGHT/LEFT JOY STICK
0211 2F                 0327          CPL
0212 321103             0328          LD      (AJ2RL),A       ;ADJUSTMENT
0215 C9                 0329          RET
                        0330 ;
                        0331 ;MOVE DOT FOR PLAYER
                        0332 ;INPUT - DE CONTAINS XY FOR CURRENT POSITION
                        0333 ;         A CONTAINS DIRECTION TO MOVE
                        0334 ;         NCOLOR CONTAINS PLAYER'S COLOR
                        0335 ;OUTPUT - DE CONTAINS NEW XY
                        0336 ;         Z SET IF CAN MOVE
                        0337 ;
0216 3D                 0338 MOVDOT:  DEC     A
0217 CA2602             0339          JP      Z,MDT300        ;MOVE UP
021A 3D                 0340          DEC     A
021B CA2A02             0341          JP      Z,MDT320        ;MOVE DOWN
021E 3D                 0342          DEC     A
021F CA2E02             0343          JP      Z,MDT340        ;MOVE RIGHT
0222                    0344                                  ;MOVE LEFT
0222 15                 0345          DEC     D               ;MOVE X POSITION LEFT
0223 C32F02             0346          JP      MDT400
                        0347 ;
0226 1D                 0348 MDT300:  DEC     E               ;MOVE Y POSITION UP
0227 C32F02             0349          JP      MDT400
                        0350 ;
022A 1C                 0351 MDT320:  INC     E               ;MOVE Y POSITION DOWN
022B C32F02             0352          JP      MDT400
                        0353 ;
022E 14                 0354 MDT340:  INC     D               ;MOVE X POSITION RIGHT
022F CDBD01             0355 MDT400:  CALL    GETCOL          ;GET COLOR
0232 A7                 0356          AND     A
0233 C0                 0357          RET     NZ              ;CAN'T MOVE
0234 3A1A03             0358          LD      A,(NCOLOR)      ;GET COLOR
0237 CDCB01             0359          CALL    PUTCOL          ;PUT COLOR
023A 97                 0360          SUB     A               ;SAY MOVED
023B C9                 0361          RET
                        0362 ;
                        0363 ;GET NEW DIRECTION FOR PLAYER 1
                        0364 ;
023C 3A0E03             0365 GNEW1:   LD      A,(AJ1UD)       ;GET ADJUSTMENT
```

```
:ROMEMCO CDOS Z80 ASSEMBLER V.1.4A
)OTCHA, GOTCHA, GOTCHA !!!!!!!!!!!!!

)23F 47          0366          LD      B,A
)240 DB1A        0367          IN      A,JOY1UD        ;READ JOY STICK UP/DOWN
)242 80          0368          ADD     B               ;ADJUST
)243 47          0369          LD      B,A
)244 3A0F03      0370          LD      A,(AJ1RL)       ;GET ADJUSTMENT
)247 4F          0371          LD      C,A
)248 DB19        0372          IN      A,JOY1RL        ;READ RIGHT/LEFT
)24A 81          0373          ADD     C               ;ADJUST
)24B 4F          0374          LD      C,A
)24C CD6E02      0375          CALL    FNDDIR          ;FIND DIRECTION
)24F A7          0376          AND     A
)250 C8          0377          RET     Z               ;NO CHANGE
)251 321803      0378          LD      (DIR1),A        ;CHANGE DIRECTION
)254 C9          0379          RET
                 0380 ;
                 0381 ;GET NEW DIRECTION FOR PLAYER 2
                 0382 ;
)255 3A1003      0383 GNEW2:   LD      A,(AJ2UD)       ;GET ADJUSTMENT
)258 47          0384          LD      B,A
)259 DB1C        0385          IN      A,JOY2UD        ;READ UP/DOWN
)25B 80          0386          ADD     B               ;ADJUST
)25C 47          0387          LD      B,A
)25D 3A1103      0388          LD      A,(AJ2RL)       ;GET ADJUSTMENT
)260 4F          0389          LD      C,A
)261 DB1B        0390          IN      A,JOY2RL        ;READ RIGHT/LEFT
)263 81          0391          ADD     C               ;ADJUST
)264 4F          0392          LD      C,A
)265 CD6E02      0393          CALL    FNDDIR          ;FIND DIRECTION
)268 A7          0394          AND     A
)269 C8          0395          RET     Z               ;NO CHANGE
)26A 321903      0396          LD      (DIR2),A        ;CHANGE DIRECTION
)26D C9          0397          RET
                 0398 ;
                 0399 ;FIND DIRECTION
                 0400 ;INPUT - B CONTAINS UP/DOWN
                 0401 ;          C CONTAINS RIGHT/LEFT
                 0402 ;OUTPUT - A CONTAINS DIRECTION
                 0403 ;
)26E 78          0404 FNDDIR:  LD      A,B             ;GET UP/DOWN
)26F A7          0405          AND     A
)270 FA7802      0406          JP      M,FDR300        ;DOWN
)273 1601        0407          LD      D,1             ;UPWARD
)275 C37C02      0408          JP      FDR320
                 0409 ;
)278 1602        0410 FDR300:  LD      D,2             ;DOWNWARD
)27A 2F          0411          CPL                     ;COMPLIMENT
)27B 47          0412          LD      B,A             ;SAVE COMPLIMENT
)27C FE40        0413 FDR320:  CP      40H
)27E D28302      0414          JP      NC,FDR330       ;LARGE MOVEMENT
)281 1600        0415          LD      D,0             ;NO CHANGE
)283 79          0416 FDR330:  LD      A,C             ;GET RIGHT/LEFT
)284 A7          0417          AND     A
```

```
;ROMEMCO CDOS Z80 ASSEMBLER V.1.4A
;OTCHA, GOTCHA, GOTCHA !!!!!!!!!!!!!!

)285 FA8D02          0418            JP      M,FDR400        ;LEFT
)288 1E03            0419            LD      E,3             ;RIGHT
)28A C39102          0420            JP      FDR420
                     0421 ;
)28D 1E04            0422 FDR400:    LD      E,4             ;LEFT
)28F 2F              0423            CPL                     ;COMPLIMENT
)290 4F              0424            LD      C,A             ;SAVE COMPLIMENT
)291 FE40            0425 FDR420:    CP      40H
)293 D29802          0426            JP      NC,FDR430       ;LARGE MOVEMENT
)296 1E00            0427            LD      E,0             ;NO CHANGE
)298 7A              0428 FDR430:    LD      A,D
)299 93              0429            SUB     E
)29A C8              0430            RET     Z               ;NO CHANGE
)29B 7A              0431            LD      A,D
)29C A7              0432            AND     A
)29D CAAA02          0433            JP      Z,FDR500        ;MUST BE RIGHT/LEFT
)2A0 7B              0434            LD      A,E
)2A1 A7              0435            AND     A
)2A2 CAAC02          0436            JP      Z,FDR550        ;MUST BE UP/DOWN
)2A5 78              0437            LD      A,B
)2A6 B9              0438            CP      C
)2A7 DAAA02          0439            JP      C,FDR500        ;MUST BE RIGHT/LEFT
)2AA 7B              0440 FDR500:    LD      A,E
)2AB C9              0441            RET
                     0442 ;
)2AC 7A              0443 FDR550:    LD      A,D
)2AD C9              0444            RET
                     0445 ;
                     0446 ;NOTES
                     0447 ;THIS ROUTINE PLAYS THE NOTES POINTED TO BY HL.
                     0448 ;1ST BYTE = VOLUME
                     0449 ;2ND BYTE = FREG. PARM
                     0450 ;3RD BYTE = LOW BYTE OF DURATION
                     0451 ;4TH BYTE = HIGH BYTE OF DURATION
                     0452 ;
)2AE 7E              0453 NOTES:     LD      A,(HL)          ;GET VOLUME
)2AF A7              0454            AND     A
)2B0 C8              0455            RET     Z               ;END OF NOTES
)2B1 47              0456            LD      B,A             ;MOVE VOLUME TO B
)2B2 23              0457            INC     HL
)2B3 4E              0458            LD      C,(HL)          ;GET FREG. PARM
)2B4 23              0459            INC     HL
)2B5 5E              0460            LD      E,(HL)          ;GET DURATION LOW
)2B6 23              0461            INC     HL
)2B7 56              0462            LD      D,(HL)          ;GET DURATION HIGH
)2B8 23              0463            INC     HL
)2B9 CDBF02          0464            CALL    TONE            ;OUTPUT TONE
)2BC C3AE02          0465            JP      NOTES
                     0466 ;
                     0467 ;TONE ROUTINE
                     0468 ;INPUT - B CONTAINS VOLUME
                     0469 ;          C CONTAINS FREG. PARM
```

```
CROMEMCO CDOS Z80 ASSEMBLER V. 1. 4A
GOTCHA, GOTCHA, GOTCHA !!!!!!!!!!!!!!

                        0470 ;          DE CONTAINS DURATION
                        0471 ;
02BF                    0472 TONE:
02BF 78                 0473 TON200:  LD      A, B              ;GET VOLUME
02C0 2F                 0474          CPL                       ;COMPLIMENT
02C1 47                 0475          LD      B, A              ;SAVE VOLUME
02C2 D319               0476          OUT     SPEEK1, A         ;OUTPUT TO SPEEKERS
02C4 D31B               0477          OUT     SPEEK2, A
02C6 79                 0478          LD      A, C              ;GET FREQ. PARM
02C7 3D                 0479 TON300:  DEC     A                 ;DEC COUNTER
02C8 C2C702             0480          JP      NZ, TON300        ;WAIT LOOP
02CB 1B                 0481          DEC     DE                ;COUNT DOWN
02CC 7A                 0482          LD      A, D
02CD B3                 0483          OR      E
02CE C2BF02             0484          JP      NZ, TON200        ;NOT THRU
02D1 C9                 0485          RET
                        0486 ;
                        0487 ;WAIT A WHILE
                        0488 ;
02D2 DB18               0489 WAIT:    IN      A, 18H            ;READ BUTTONS
02D4 E622               0490          AND     22H               ;LOOK AT BOTH 2'S
02D6 FE22               0491          CP      22H               ;Q, IS EITHER PUSHED
02D8 061E               0492          LD      B, 30             ;SLOW TIME
02DA CADF02             0493          JP      Z, DELAY          ;NONE PUSHED, SLOW TIME
02DD 060F               0494          LD      B, 15             ;FAST TIME
                        0495 ;
                        0496 ;DELAY
                        0497 ;INPUT - B CONTAINS TIME
                        0498 ;
02DF DB0E               0499 DELAY:   IN      A, 14             ;READ DAZZLER
02E1 FE3F               0500          CP      3FH
02E3 C2DF02             0501          JP      NZ, DELAY         ;WAIT FOR END OF FRAME
02E6 DB0E               0502 DLY300:  IN      A, 14             ;READ DAZZLER
02E8 FE3F               0503          CP      3FH
02EA CAE602             0504          JP      Z, DLY300         ;WAIT FOR START OF FRAME
02ED 05                 0505          DEC     B
02EE C2DF02             0506          JP      NZ, DELAY         ;DO IT AGAIN
02F1 DB18               0507          IN      A, 18H            ;READ BUTTONS
02F3 E644               0508          AND     44H
02F5 FE44               0509          CP      44H
02F7 C20B03             0510          JP      NZ, ABORT
02FA C9                 0511          RET
                        0512 ;
                        0513 ;WAIT FOR BUTTON 1
                        0514 ;*** ABORT IF BUTTON 3 ****
                        0515 ;
02FB DB18               0516 WBONE:   IN      A, 18H            ;READ BUTTON
02FD E611               0517          AND     11H
02FF FE11               0518          CP      11H
0301 C0                 0519          RET     NZ                ;BUTTON HIT
0302 DB18               0520          IN      A, 18H            ;READ BUTTONS
0304 E644               0521          AND     44H
```

```
306 FE44          0522            CP      44H
308 CAFB02        0523            JP      Z,WBONE
30B C30000        0524 ABORT:     JP      0               ;*** ABORT ***
                  0525 ;
    (001A)        0526 JOY1UD:    EQU     1AH             ;JOY STICK 1 UP/DOWN
    (0019)        0527 JOY1RL:    EQU     19H             ;JOY STICK 1 RIGHT/LEFT
    (001C)        0528 JOY2UD:    EQU     1CH             ;JOY STICK 2 UP/DOWN
    (001B)        0529 JOY2RL:    EQU     1BH             ;JOY STICK 2 RIGHT/LEFT
    (0019)        0530 SPEEK1:    EQU     19H             ;SPEEKER 1
    (001B)        0531 SPEEK2:    EQU     1BH             ;SPEEKER 2
                  0532 ;
30E (0001)        0533 AJ1UD:     DEFS    1               ;ADJUSTMENT FOR JOY1UD
30F (0001)        0534 AJ1RL:     DEFS    1               ;ADJUSTMENT FOR JOY1RL
310 (0001)        0535 AJ2UD:     DEFS    1               ;ADJUSTMENT FOR JOY2UD
311 (0001)        0536 AJ2RL:     DEFS    1               ;ADJUSTMENT FOR JOY2RL
                  0537 ;
312 (0001)        0538 NUM1:      DEFS    1               ;PLAYER 1 SCORE
313 (0001)        0539 NUM2:      DEFS    1               ;PLAYER 2 SCORE
314 (0002)        0540 POS1:      DEFS    2               ;PLAYER 1 POSITION
316 (0002)        0541 POS2:      DEFS    2               ;PLAYER 2 POSITION
318 (0001)        0542 DIR1:      DEFS    1               ;PLAYER 1 DIRECTION
319 (0001)        0543 DIR2:      DEFS    1               ;PLAYER 2 DIRECTION
                  0544 ;
    (000A)        0545 BCOLOR:    EQU     0AH             ;GREEN FOR BOARDER
31A (0001)        0546 NCOLOR:    DEFS    1               ;COLOR FOR NUMBER
    (0009)        0547 P1COLR:    EQU     09H             ;PLAYER 1 COLOR = RED
    (000C)        0548 P2COLR:    EQU     0CH             ;PLAYER 2 COLOR = BLUE
                  0549 ;
    (0040)        0550            DEFS    64
    (035B)        0551 STACK:     EQU     $               ;STACK
                  0552 ;
    (0400)        0553 DISPLY:    EQU     [$-1]/512*512+512
                  0554 ;
                  0555            END

)00 ERRORS
```

CROMEMCO CROSS REFERENCE LISTING V.1.0 FOR FILE GOTCHA

```
ABORT   0525  0511
ADDAHL  0308  0154 0242
AJ1RL   0535  0321 0369
AJ1UD   0534  0318 0364
AJ2RL   0537  0327 0387
AJ2UD   0536  0324 0382
BCOLOR  0546  0200 0207 0214 0221
BOARD   0198  0127
BRD300  0200  0204
BRD320  0207  0211
BRD340  0214  0218
BRD360  0221  0225
CLEAR   0297  0126 0304
CPOS    0083  0048 0058
DAZWRT  0116  0072
DELAY   0500  0494 0502 0507
DIR1    0543  0013 0121 0377
DIR2    0544  0030 0123 0395
DISPLY  0554  0107 0124 0243
DLY300  0503  0505
DNM300  0156  0176
DNM320  0158  0167
DNM350  0163  0161
DNMTAB  0179  0149
DOTPTR  0233  0252 0269
DPLAY1  0140  0052 0128
DPLAY2  0132  0062
DSPNUM  0149  0136
ENDRND  0070  0066
FDR300  0409  0405
FDR320  0412  0407
FDR330  0415  0413
FDR400  0421  0417
FDR420  0424  0419
FDR430  0427  0425
FDR500  0439  0432 0438
FDR550  0442  0435
FNDDIR  0403  0374 0392
GCL300  0259  0254
GETCOL  0252  0354
GNEW1   0364  0076
GNEW2   0382  0077
INIT    0105  0007 0071
INTJOY  0316  0112
JOY1RL  0528  0319 0371
JOY1UD  0527  0316 0366
JOY2RL  0530  0325 0389
JOY2UD  0529  0322 0384
MAIN    0009  0073 0078
MAN300  0025  0019
MAN320  0042  0036
MAN400  0055  0044
MAN450  0063  0049 0053 0057 0059
```

CROMEMCO CROSS REFERENCE LISTING V. 1.0 FOR FILE GOTCHA

```
MAN500   0064
MAN550   0072  0069
MAN600   0075  0047
MDT300   0347  0338
MDT320   0350  0340
MDT340   0353  0342
MDT400   0354  0345 0348 0351
MOVDOT   0337  0014 0031
NCOLOR   0547  0012 0029 0134 0142 0160 0357
NOTES    0452  0096 0464
NUM1     0539  0050 0064 0110 0143
NUM2     0540  0060 0067 0111 0135
P1COLR   0548  0011 0141
P2COLR   0549  0028 0133
POS1     0541  0009 0016 0083 0117
POS2     0542  0026 0033 0085 0119
PTC400   0283  0272
PTC900   0287  0281
PUTCOL   0266  0022 0039 0163 0201 0208 0215 0222 0358
SON090   0098  0095
SONG     0095  0063
SPEEK1   0531  0477
SPEEK2   0532  0478
STACK    0552  0006
START    0005
TON200   0474  0485
TON300   0480  0481
TONE     0473  0463
WAIT     0490  0075
WBONE    0517  0008 0070 0524
```

KALEIDOSCOPE

KALEIDOSCOPE, written by Li-Chen Wang, is surely one of the most colorful Dazzler programs. No keyboard is required, and there are no controls to operate. Just sit back and marvel at what a program only 127 bytes long can do.

KALEIDOSCOPE uses the first 2.5K of memory space. The upper 2K of this area is reserved for the Dazzler picture. The lower 127 bytes are used for the program.

When using KALEIDOSCOPE from the Cromemco Dazzler games diskette, simply type "KALEIDO" to begin program execution.

LIFE

The game of LIFE was first introduced in the October 1970 issue of Scientific American magazine. The game is described in the following issues of Scientific American: October 1970, p. 120; February 1971, p. 112; April 1971, p. 116. The Dazzler-Life program is a truly spectacular full-color interpretation of the interesting and varied game of LIFE.

Operation

Once the LIFE program is loaded into the computer, an initial colony of cells can be drawn on the TV screen using keyboard controls.

C	move cursor
D	deposit data and move cursor
E	erase data and move cursor

The motion of the cursor in the above functions continues in a given direction until that direction is changed by one of the following:

W	move cursor up
Z	move cursor down
A	move cursor to the left
S	move cursor to the right
RETURN	move cursor to the left edge
Q	move cursor home

Note that W, Z, A and S form a diamond-shaped pattern on the keyboard.

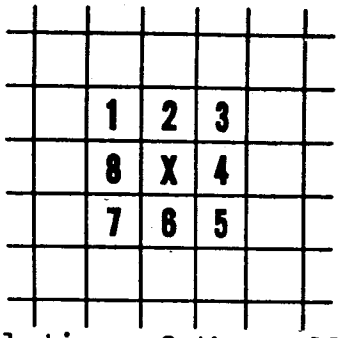
The cursor can be made to move diagonally by using the control key in conjunction with W, Z, A and S. For example, if control-W is pushed followed by control-S, the cursor will then move diagonally up to the right until changed by further keystrokes.

Once the initial colony is complete, the following keys can be used to control the evolution of the cells.

G	go (starts the evolution)
F	freeze (halts the evolution until the space-bar is pushed)
CONTROL-X	kill the entire colony and start over.

The Game of LIFE

Once the initial colony of cells is drawn on the TV screen using the keyboard commands described above, the keyboard command "G" begins the evolution of the life simulation. The evolution then proceeds according to a fixed set of rules. Each cell in the colony has 8 possible neighbors as shown below:



Each cell has 8 possible neighbors.

The evolution of the cells proceeds in a sequence of distinct generations. Every cell with two or three neighbors will survive to the next generation. Every cell with four or more neighbors dies from over-population. Every empty cell with exactly three neighbors is a birth cell - a new cell is born here in the subsequent generation.

In the Dazzler display of Life, blue cells are used to represent life itself. The birth of a new cell is shown in green, while the death of a cell is shown in red.

There are many surprises to be found in the game. Some colonies survive and prosper; others reach a stable state - neither growing nor lessening. Other colonies are doomed to extinction. Still other colonies, known as "gliders" sail across the screen and can be devoured by other colonies in the process.

XLIFE is a particularly attractive LIFE display that is supplied on your Dazzler game diskette.

SPACEWAR

SPACEWAR is a simulation game for two players. The simulation is performed by a Z-80 or 8Ø8Ø microprocessor equipped with at least the following:

1. 16K RAM Memory
2. A CROMEMCO DAZZLER TV interface (including a TV)
3. A CROMEMCO D+7A analog/digital board
4. A pair of CROMEMCO JS-1 joysticks

The program simulates a portion of an imaginary universe. Within this portion of space, the two combatants' spaceships travel around a central sun and are attracted to it by gravity. The spaceships have distinct profiles so that they can be distinguished.

Each joystick console controls one ship. The object of the is to blow up your opponent's ship with a torpedo, while your remains intact.

The joystick controls the acceleration and the aspect of the ship. Move the stick slightly forward to activate the ship's thruster. The ship will accelerate in the direction it is pointed as long as the stick is held forward (and as long as there is fuel remaining). Acceleration is indicated by exhaust leaving the rear of the ship.

The ship will rotate clockwise while the joystick is held to the right of center, and counter-clockwise while held to

the left of center. Ships can be rotated as much as desired without using up any resources

The actions may be combined. For example, holding the stick forward and to the right will cause the ship to accelerate while rotating clockwise

Note that it only takes a small motion forward or to the side to control the ship. Pushing the stick all the way forward or to the side produces no additional effect.

Pulling the stick to the rear will cause the ship to enter HYPERSPACE (see next page). It is necessary to pull the stick at least 3/4 of the way back in order to accomplish this (unlike the comparatively subtle motions required for thruster and rotator control).

Pushing switch 1 on the console causes a torpedo to be fired from the torpedo tube located in the nose of the ship The torpedo leaves the ship with a fixed forward velocity relative to the ship's velocity. To aim the torpedo you must aim the entire ship. A torpedo will destroy any ship or other torpedo which it may come very close to. Each ship has a limited number of torpedoes.

Torpedoes self-destruct after a short period. Their range is thus limited by their speed. Torpedoes are not affected by gravity so if they are fired in a forward direction by a ship near the sun (and thus going fast) they will fly away at great speed. If switch 1 is held down, torpedoes will be fired in a machine-gun-like fashion at the rate of about 2 per second at 2MHz CPU speed (or 4 per second at 4MHz)

When a player's ship is about to be blown up by a torpedo which can't be shot down, it is wise to enter HYPERSPACE as a last resort. This is done by pulling the joystick sharply to the rear. The ship will disappear, to reappear shortly thereafter in some random location disguised as a star. While in this state, it is vulnerable to torpedoes but cannot be controlled. Another second or two and it reappears as a spaceship, with a random velocity and attitude imparted to it by hyperspace. There is a small (1/8) chance that it will explode upon emergence from hyperspace - so hyperspace is indeed a last resort.

Special Environmental Details:

1. First one should note that space curves back upon itself in such a manner that the upper and lower boundaries of space coincide. Consequently, if a ship or a torpedo drifts off the top of the screen, it reappers on the bottom, and vice-versa. The same is true of the left and right boundaries. Experts will use this fact to "shoot around the screen" and a novice will find a torpedo attacking him from out of nowhere.
2. Since each opposite edge is identical in the simulated environment, all four corners are in reality one single spot in space (in fact, the spot furthest from the sun). If the sun attracts a ship too closely, rather than swallow it up, the ship is dumped "in the four corners". This spatial singularity adds interest to the game but a physics purist may suppress it (see OPERATING INSTRUCTIONS).

3. The stars in the background are part of a large star field which circulates about once per hour. These stars have no effect on the game, except for aesthetics and helping the players see the edges of the screen

SCORING:

1. If both players run out of torpedoes, the game is counted as a tie.

2. When either ship explodes, the simulation continues for a few seconds (to make sure the survivor evades any remaining torpedoes) and then the survivor (if any) is credited with a win.

3. If your microprocessor has an IMSAI or CROMEMCO front panel with 8 programmed output lights, the score is kept there. The rightmost 4 bits for the player on the right and the leftmost 4 bits for the player on the left, naturally.

4. It is customary to play until one player achieves a certain score (usually 16- overflowing his 4 bit counter) and then he plays the next opponent. Shorter series, such as best 5 out of 9, are quite rewarding. To reset the score, restart the program (see below).

OPERATING INSTRUCTIONS

The starting address of the program is Ø. The game may be restarted by simultaneously depressing switches 2 and 3 on either console. (However, if either ship has exploded, the program will not respond to a restart request for several seconds).

OPTIONS

Several options may be selected when the game is restarted. Hold down the switches corresponding to the desired options on one console and depress and release switches 2 and 3 on the other console

Switch	Option
2	Eliminate the sun (and its gravity
3	Cause the sun to be lethal.
4	Eliminate the starfield.

(Combinations of options which do not involve both switches 2 and 3 are allowed).

TANK WAR

Tank War is a two-player computer game using two Cromemco JS-1 joystick consoles and the Cromemco Dazzler TV interface. The Tank War program itself requires 3K bytes of memory (beginning at location zero in memory space). An additional 3K bytes of RAM memory (from 3K to 6K in memory space) are required for picture storage and stack area.

The game begins with the words "TANK WAR" boldly displayed in color on the TV screen. To start playing, depress button 4 on both joystick consoles. Each player can then control his tank using his joystick. Missiles can be fired from the tank by depressing button 1 on the joystick console. Two points are scored when the opponent's tank is successfully hit by a missile. The opponent gains a point if a mine is contacted in the mine field. Score is kept automatically for each player in the upper corners of the playing screen.

The game continues until one of the players wins by reaching the score of 90. To start the game again depress buttons 2 and 3 on both joystick consoles.

TRACK

TRACK is a full-color TV game designed to be used with the Cromemco TV DAZZLER interface. Track is a game of skill and coordination. The object is to manipulate a cursor, under joystick control, through a spiral path toward the center goal. If, however, the player contacts the sides of the spiral in the process, the game is over and must start again.

TRACK begins with a white spiral track displayed on a bright green background. A joystick (Cromemco model JS-1) is used to control the yellow cursor on this track. Towards the center of the spiral the track narrows, requiring increasingly precise control of the cursor to avoid contacting the sides of the spiral. If the side of the spiral is hit, an alarm (in the JS-1 console) sounds, and the point of contact is turned bright red.

APPENDIX A

DAZZLER ENGINEERING CHANGES

In order for your REV B or REV C series Dazzler to operate properly with the Cromemco games diskette, the following engineering changes must be made:

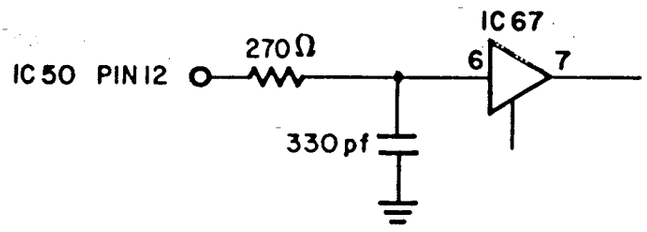
REV B DAZZLER ONLY

1 Remove (or bend out) pin 10 of Dazzler IC 29 (A 7400 IC)

2) Remove (or bend out pin 12 of Dazzler IC 66 (A 7405 IC).

REV B AND REV C DAZZLERS

1) Add a 270 ohm resistor and 330 picofarad capacitor to the Dazzler as shown below:



2) On Dazzler board 2 connect a jumper wire from finger 54 of the S-100 bus connection (EXT. CLEAR) to finger 75 of the S-100 bus (RESET).

3) There are 4 pads on board 2 just above IC57 in a triangle Cut the trace on the component side which runs between the two leftmost pads. This trace connects IC57P12 to IC49P1. Put a wire jumper so that IC49P1 connects to IC57P11 instead. This eliminates the bus float state at DMA transfer.