SigmaSoft and Systems

SOUND EFFECTS BOARD USERS' MANUAL

I. Introduction

The Sound Effects Board is a peripheral device which was designed especially for the Heathkit H89 computer. It is an implementation of the General Instruments AY-3-8910 Programmable Sound Generator (PSG), and this board supports the full capabillities of the PSG, as well as some additional functions, such as game paddles, and two parallel ports. The Programmable Sound Generator Data Manual, which is supplied in addition to this manual, is a complete description of the PSG itself, and of how to program the device. Therefore, it is important that the user of this Sound Effects Board read the PSG Data Manual carefully, before attempting to operate the device by the instructions given in this manual. Several demonstration programs, as well as programming instructions are given in this manual, for the purpose of giving specific examples for operating the board on the H89 computer, and how it implements the use of the PSG device

II. Installation

A. Board Installation

The Sound Effects Board can be connected to either P504, or P505 on the H89 CPU board, and uses one of the three serial ports However, all three serial ports can not be operated with the sound effects board simultaneously. Therefore, if you have a three port serial I/O board it is extremely important that you read the section below entitled "Port Configuration" before you attempt to operate the sound effects board.

After you have installed the sound effects board, simply connect the audio cable to the audio connector (The two pin connector nearest the top of the board), and then connect the other end of the cable to an audio amplifier. A tape deck or auxillury input on a stero receiver is ideal, however, a microphone or turntable input will also work fine

B. Paddle Installation

If you purchased a game paddle it can be connected to either of the two pin paddle connectors, which are located on the top right corner of the board The top paddle connector is for the left hand game paddle, and the lower paddle connector is for the right hand game paddle. If you purchased two paddles, they can be operated simultaneously

C. Port Configuration

Unless you specified otherwise in your order, your board has been configured to operate at port 208-209D, which is the Alternate Terminal port. This configuration can easily be changed so that the board will operate at any of the three serial ports. However, the port that your board is configured for can not be used by any other device at the same time. So, if you have a serial I/O board the port you wish to use must be disconnected from it so that it will not interfere with the communication between the sound board and the CPU of your H89 computer.

To disconnect the serial I/O board from the port you wish to use for the sound board, you must first remove the corresponding UART from the serial I/O board. The UARTs are the large 40 pin Integrated Curcuits on the serial I/O board, and there may be either two or three of them, depending upon whether you have a two or three port serial I/O board There is one UART for each of the serial ports, so by removing one, you will free that corresponding port for use by the sound effects board (See chart below). If you have the two port serial I/O board, and if your sound effects board is configured to operate on the Alternate Terminal port, then no modification is nesassary since the UART for that port is not present anyway. You only need to remove a UART if you have a three port serial I/O board, or if you wish to operate the sound board on the line printer, or modem port.

To configure the sound effects board to operate at another port, simply remove the small jumper between the foil and the 25 pin edge card socket with a soldering iron, and then resolder the jumper between the same foil and whichever of the three pins you want. Each pin is a different port, and the chart below shows how the pins of the edge card socket correspond to the various port addresses. The chart also lists the IC numbers for the UARTs that must be removed in order to use that port.

25 pin edge card socket

```
   +---+
   | 1 |
   | 2 |
   | 3 |
   | 4 |
   | 5 |
   | 6 |
   | 7 |                  Intended Use          Port Address         IC Number
   | 8 |
   | 9 - 1st Port   Alternate Terminal    320-321Q (208-209D)    Remove U603
   |10 - 2nd Port   Modem                 330-331Q (216-217D)    Remove U604
   |11 - 3rd Port   Line Printer          340-341Q (224-225D)    Remove U602
   |12 |
   |13 |
     -
   |24 |
   |25 |
   +---+
```

III. Operation

A. Programming the PSG for Sound Effects

Throughout the rest of this manual it will be assumed that your board is configured for the Alternate Terminal Port (208-209 Decimal), so if you have changed the configuration of your board to operate at another port, your sound board will not respond to data sent out the Alternate Terminal Port. Refer to the chart below to determine the correct port addresses for your board.

Serial Port	Data Port (Even)		Address Port (Odd)
Alternate Terminal Port	320 Octal (208 Decimal)	-	321 Octal (209 Decimal)
Modem Port	330 Octal (216 Decimal)	-	331 Octal (217 Decimal)
Line Printer Port	340 Octal (224 Decimal)	-	341 Octal (225 Decimal)

Whichever of the ports your board is configured for, there will be a set of two addresses, one even and one odd. You can read and write to both addresses, which means there are four possible operations, and the chart below summarizes them.

Data I/O	Read	Write
Port 208D (Even) 320Q	Data is read from one of the PSG's 16 registers.	Data is loaded into one of the PSG's 16 registers.
Port 209D (Odd) 321Q	Data is read from the game paddles.	Specifies to the PSG, which register you want.

Once the PSG is enabled, it will continuously generate an audio output based on the data that is currently present in its 16 internal registers To change the contents of these registers is a two step process. First, you must specify which of the 16 registers you want to access by writing the number of the register (0-15) out the odd numbered port (209D) Then, you can either read, or write data from/to that register by either reading, or writing from/to the even numbered port (208D)

To enable sound effects out the audio output requires that you load a value other than zero into one or more of the three volume control registers, which are 8, 9, and 10. You must also set certain bits of the enable register, which is register number 7 (See pages 11, 18-29 in the PSG Data Manual).

1. Sound Effects in BASIC

BASIC has two commands which provide direct port I/O capability. These are the OUT A,B command and the PIN(A) function The OUT command can be used to write a value into one of the PSG's 16 internal registers, and the PIN function can be used to either read the data from one of the PSG's registers, or to read the status of the game paddles The example below shows how a register of the PSG can be programmed in BASIC.

```
10 OUT 209,8
20 OUT 208,15            See DEMONSTRATION PROGRAM II
30 A=PIN(208)
```

Line 10 in the above program, selects the volume control register for channel A (8), by OUTing the number 8 to the odd numbered port. Then, line 20 selects the maximum volume (15) by OUTing 15 to the even numbered port. Line 30 then reads the data (15) back from the volume control register for channel A (8), and stores it in variable A.

2. Sound Effects in Assembly Language

Assembly Language has two mnemonics which provide port I/O capability. These are OUT ADDRESS and IN ADDRESS. The OUT instruction can be used to write a value into one of the PSG's 16 internal registers, and the IN instruction can be used to either read the data from one of the PSG's registers, or to read the status of the game paddles. The example below shows how a register of the PSG can be programmed with Assembly Language

```
VOLUME  MVI     A,8
        OUT     209     See DEMONSTRATION PROGRAM IV
        MVI     A,15
        OUT     208
        IN      208
```

The first two lines in the above program select the volume control register for channel A (8) by OUTing the value 8 to the odd numbered port Then, the next two lines select the maximum volume (15) by OUTing 15 to the even numbered port The last line then reads the data (15) back from the volume control register for channel A (8).

B. Reading the Status of the Game Paddles

The current status of both of the game paddles is continuously present at the odd numbered port (209D) and can be read as often as your program requires. The game paddles do not generate interrupts, so they will be ignored by the computer, unless your program is monitoring them frequently When the game paddle status is read, the data from both paddles is coded into a single byte so that a single read operation is all that is required. The least significant four bits contain the data from the right hand paddle, and the most significant four bits contain the data from the left hand paddle. Each of the four bits represents one of the four possible directions, and when the red button is pushed all four of the bits for that paddle are set to logical "1" to indicate that the button is being pressed. The chart below is a sample listing of the data that might be read from the game paddles.

	Left Hand Paddle	Right Hand Paddle			
	Most Significant 4 Bits	Least Significant 4 Bits		Decimal Equivilents	Indicated Status
Sample data bytes read from game paddles.	0000	0000	=	(000D)	Neither paddle is being used
	1000	0000	=	(128D)	Left paddle is moving left
	0100	0000	=	(064D)	Left paddle is moving down
	0010	0000	=	(032D)	Left paddle is moving right.
	0001	0000	=	(016D)	Left paddle is moving up.
	0000	1000	=	(008D)	Right paddle is moving left.
	0000	0100	=	(004D)	Right paddle is moving down.
	0000	0010	=	(002D)	Right paddle is moving right.
	0000	0001	=	(001D)	Right paddle is moving up.
	1100	0000	=	(192D)	Left paddle moving left and down.
	0000	1100	=	(012D)	Right paddle moving left and down.
	0100	0001	=	(065D)	Left down and Right up
	1111	0000	=	(240D)	Left red button is depressed
	0000	1111	=	(015D)	Right red button is depressed.
	1111	1111	=	(255D)	Both red buttons are depressed
	1111	0001	=	(241D)	Left red button is depressed and right paddle is moving up.

1. Using the Paddles in BASIC

The PIN(A) function of BASIC will allow you to read the status of the game paddles. The following example shows how to read the game paddle status and store it in a numerical variable, such as "A".

```
10 A=PIN(209)
```

Variable A would now be equal to the value from the game paddle status port, and you can selectively mask out the particular bits you are interested in by using the AND operator (See DEMONSTRATION PROGRAM III)

2. Using the Paddles in Assembly Language

The IN ADDRESS instruction of Assembly Language will allow you to monitor the game paddle status. The following example shows how to read the game paddle status into the accumulator.

```
READ    IN      209     *Odd numbered port.
```

Register A would now be equal to the value from the game paddle status port, and you can selectively mask out the particular bits you are interested in by using logical AND instructions (See DEMONSTRATION PROGRAM V)

C. Accessing the Parallel I/O Ports

Data I/O with the parallel ports is done through the PSG. First, you must enable the port you want for either read, or write in register 7 of the PSG. Then, any data that is written to one of the two registers (14, or 15) will be transferred to the data buss connectors, and latched until a new value is sent to take its place. If the port is enabled for reading, then the data that is present on the data buss connector will be transferred to the CPU when your program reads from that port (See DEMONSTRATION PROGRAM I).

IV. Hardware Expansion

A. Interfacing to the Parallel I/O Ports

The sound effects board has two parallel I/O ports, which can be accessed through the PSG. There are four connectors that make up the interface of these two ports, two seperate data busses and two sets of control lines (See Schematic). The two connectors just below the PSG are the data buss connectors The left connector is Port B and the right connector is Port A The two connectors nearest the right edge of the board are the control connectors for these two ports (These two connectors are identical).

When your software outputs data to one of the two ports that data will be latched, and continually displayed at one of the two data buss connectors, until new data is sent to take its place. For reading data your interface can hold a value on one of the port busses, and when your software reads from that port, the data will be transferred onto the CPU data buss by the PSG.

Port A and Port B Data Connectors

1	Bit 1 of data buss (Least Significant Bit)
2	Bit 2 of data buss
3	Bit 3 of data buss
4	Bit 4 of data buss
5	Bit 5 of data buss
6	Bit 6 of data buss
7	Bit 7 of data buss
8	Bit 8 of data buss (Most Significant Bit)

Port A and Port B Control Connectors

1	+5 Volt Supply
2	Ground
3	Master Reset - Active Low
4	Interrupt 3 - Active Low
5	Interrupt 4 - Active Low
6	+12 Volt Supply
7	Processor Wait - Active Low
8	Write to PSG - Active High
9	Read from PSG - Active High
10	0.8948863 MHz Clock

B. The Expansion Connector

This board was designed with future expansion in mind. The eight pin connector nearest the bottom of the board is the Expansion Connector, which provides a number of important control lines that are crucial to future expansion of the H89 computer. This connector is not used in the operation of the sound effects board itself

Expansion Connector

1	Processor Read - Active Low
2	Processor Write - Active Low
3	Processor Wait - Active Low
4	0.8948863 MHz Clock
5	Master Reset
6	Interrupt 3 - Active Low
7	Interrupt 4 - Active Low
8	Interrupt 5 - Active Low

DEMONSTRATION PROGRAM I

```
00005 REM Data Input/Output Program.
00010 INPUT "Would you like to input, or output data?";A$
00015 IF LEFT$(A$,1)="O" GOTO 40
00020 OUT 209,7:OUT 208,191:REM Enable port A for input (191D=10111111B).
00025 OUT 209,14:REM            Port A is register 14.
00030 PRINT PIN(208):REM        Read, and print data.
00035 GOTO 30
00040 OUT 209,7:OUT 208,255:REM Enable port A for read (255D=11111111B).
00045 OUT 209,14:REM            Send address for port A.
00050 INPUT "Data to be output?";A
00055 OUT 208,A:REM             Write data to port A.
00060 GOTO 50
```

DEMONSTRATION PROGRAM II

```
00005 REM Frequency Sweep Program.
00010 OUT 209,7:OUT 208,254:REM Enable tone on channel A only (254D=11111110B).
00015 OUT 209,8:OUT 208,15:REM  Select maximum amplitude on channel A.
00020 FOR A=0 TO 15:REM         Sweep through the complete spectrum.
00025 OUT 209,1:REM             Data will be going to register 1
00030 OUT 208,A
00035 OUT 209,0:REM             Data will be going to register 0.
00040 FOR B=0 TO 255:REM        Fine tuning.
00045 OUT 208,B
00050 NEXT B:NEXT A
00055 GOTO 20
```

DEMONSTRATION PROGRAM III

```
00005 REM Paddle monitoring program.
00010 A=PIN(209):IF A=0 GOTO 10
00015 B=A AND 15
00020 IF B=15 THEN PRINT "Right Blast":GOTO 10
00025 B=A AND 240
00030 IF B=240 THEN PRINT "Left Blast":GOTO 10
00035 B=A AND 1
00040 IF B=1 THEN PRINT "Right Up"
00045 B=A AND 2
00050 IF B=2 THEN PRINT "Right Right"
00055 B=A AND 4
00060 IF B=4 THEN PRINT "Right Down"
00065 B=A AND 8
00070 IF B=8 THEN PRINT "Right Left"
00075 B=A AND 16
00080 IF B=16 THEN PRINT "Left Up"
00085 B=A AND 32
00090 IF B=32 THEN PRINT "Left Right"
00095 B=A AND 64
00100 IF B=64 THEN PRINT "Left Down"
00105 B=A AND 128
00110 IF B=128 THEN PRINT "Left Left"
00115 IF A=0 THEN PRINT "Right, and Left Stop"
00120 GOTO 10
```

DEMONSTRATION PROGRAM IV

```
SWEEP                                                       HEATH ASM #104.05.00
                                                            07-May-81   Page   1

   042.200                 00001           XTEXT   HDOSDEF
   042 200                 00034           ORG     USERFWA
   042 200  076 010        00035  BEGIN    MVI     A,8      *Maximum Volume
   042.202  323 321        00036           OUT     321Q
   042.204  076 017        00037           MVI     A,15
   042.206  323 320        00038           OUT     320Q
   042.210  076 007        00039           MVI     A,7      *Enable tone on chan.
   042.212  323 321        00040           OUT     321Q
   042.214  076 376        00041           MVI     A,254
   042.216  323 320        00042           OUT     320Q
   042.220  001 000 000    00043  SWEEP    LXI     B,0      *Clear the BC
   042 223  076 001        00044  LOOPA    MVI     A,1
   042 225  323 321        00045           OUT     321Q
   042.227  170            00046           MOV     A,B      *B=Coarse frequency
   042.230  323 320        00047           OUT     320Q
   042.232  074            00048           INR     A
   042.233  376 020        00049           CPI     16       *15 is maximum freq.
   042.235  312 220 042    00050           JE      SWEEP
   042 240  107            00051           MOV     B,A
   042.241  257            00052  LOOPB    XRA     A        *Dead waitting loop
   042.242  074            00053  WAIT     INR     A
   042.243  376 000        00054           CPI     0
   042.245  302 242 042    00055           JNE     WAIT
   042.250  323 321        00056           OUT     321Q
   042.252  171            00057           MOV     A,C      *C=Fine frequency
   042.253  323 320        00058           OUT     320Q
   042.255  074            00059           INR     A
   042.256  376 000        00060           CPI     0        *255 is maximum freq.
   042 260  312 223 042    00061           JE      LOOPA
   042.263  117            00062           MOV     C,A
   042.264  303 241 042    00063           JMP     LOOPB
   042.267  000            00064           END     BEGIN

00064 Statements Assembled
30377 Bytes Free
 No Errors Detected
```

DEMONSTRATION PROGRAM V

```
042 200                 00001           XTEXT   HDOSDEF
042.200                 00034           ORG     USERFWA
042.200  333 321        00035  READ     IN      209      *Input data from paddle
042.202  376 000        00036           CPI     0
042.204  312 200 042    00037           JE      READ
042.207  062 046 043    00038           STA     DATA     *Store it in memory
042.212  346 017        00039  BLASTR   ANI     15       *Right button pushed?
042.214  376 017        00040           CPI     15
042.216  302 231 042    00041           JNE     BLASTL
042.221  041 171 043    00042           LXI     H,RBLAST
042 224  377 003        00043           SCALL   .PRINT
042.226  303 200 042    00044           JMP     READ
042.231  072 046 043    00045  BLASTL   LDA     DATA
042.234  346 360        00046           ANI     240      *Left button pushed?
042.236  376 360        00047           CPI     240
042 240  302 253 042    00048           JNE     DIREC1
042.243  041 205 043    00049           LXI     H,LBLAST
042 246  377 003        00050           SCALL   .PRINT
042.250  303 200 042    00051           JMP     READ
042.253  072 046 043    00052  DIREC1   LDA     DATA
042.256  346 001        00053           ANI     1        *Mask all unwanted bits
042.260  376 001        00054           CPI     1        *Is that bit set?
042.262  302 272 042    00055           JNE     DIREC2   *If not, try second bit
042.265  041 047 043    00056           LXI     H,MSG1
042.270  377 003        00057           SCALL   .PRINT
042.272  072 046 043    00058  DIREC2   LDA     DATA     *Reload data into A
042.275  346 002        00059           ANI     2        *Mask all unwanted bits
042.277  376 002        00060           CPI     2        *Is that bit set?
042.301  302 311 042    00061           JNE     DIREC3   *If not, try third bit
042.304  041 060 043    00062           LXI     H,MSG2
042.307  377 003        00063           SCALL   .PRINT
042.311  072 046 043    00064  DIREC3   LDA     DATA     *Repeat procedure
042.314  346 004        00065           ANI     4
042 316  376 004        00066           CPI     4
042.320  302 330 042    00067           JNE     DIREC4
042.323  041 074 043    00068           LXI     H,MSG3
042.326  377 003        00069           SCALL   .PRINT
042.330  072 046 043    00070  DIREC4   LDA     DATA
042.333  346 010        00071           ANI     8
042.335  376 010        00072           CPI     8
042.337  302 347 042    00073           JNE     DIREC5
042 342  041 107 043    00074           LXI     H,MSG4
042.345  377 003        00075           SCALL   .PRINT
042.347  072 046 043    00076  DIREC5   LDA     DATA
042.352  346 020        00077           ANI     16
042.354  376 020        00078           CPI     16
042.356  302 366 042    00079           JNE     DIREC6
042.361  041 122 043    00080           LXI     H,MSG5
042.364  377 003        00081           SCALL   .PRINT
042.366  072 046 043    00082  DIREC6   LDA     DATA
042.371  346 040        00083           ANI     32
042.373  376 040        00084           CPI     32
042.375  302 005 043    00085           JNE     DIREC7
043.000  041 132 043    00086           LXI     H,MSG6
043.003  377 003        00087           SCALL   .PRINT
```

```
   043.005  072 046 043 00088   DIREC7  LDA     DATA
   043.010  346 100     00089           ANI     64
   043 012  376 100     00090           CPI     64
   043 014  302 024 043 00091           JNE     DIREC8
   043 017  041 145 043 00092           LXI     H,MSG7
   043.022  377 003     00093           SCALL   .PRINT
   043.024  072 046 043 00094   DIREC8  LDA     DATA
   043.027  346 200     00095           ANI     128
   043.031  376 200     00096           CPI     128
   043.033  302 200 042 00097           JNE     READ
   043.036  041 157 043 00098           LXI     H,MSG8
   043.041  377 003     00099           SCALL    PRINT
   043.043  303 200 042 00100           JMP     READ     *Read data again
   043.046              00101   DATA    DS      1        *Paddle status data
   043.047  122 151 147 00102   MSG1    DB      'Right Up',10+128
   043.060  122 151 147 00103   MSG2    DB      'Right Right',10+128
   043.074  122 151 147 00104   MSG3    DB      'Right Down',10+128
   043.107  122 151 147 00105   MSG4    DB      'Right Left',10+128
   043.122  114 145 146 00106   MSG5    DB      'Left Up',10+128
   043.132  114 145 146 00107   MSG6    DB      'Left Right',10+128
   043.145  114 145 146 00108   MSG7    DB      'Left Down',10+128
   043.157  114 145 146 00109   MSG8    DB      'Left Left',10+128
   043.171  122 151 147 00110   RBLAST  DB      'Right Blast',10+128
   043.205  114 145 146 00111   LBLAST  DB      'Left Blast',10+128
   043.220  000         00112           END     READ

00112 Statements Assembled
30238 Bytes Free
 No Errors Detected
```

SOUND EFFECTS FOR THE H89!

Now your H89 computer can have sound effects that rival the sounds of the best arcade video games! This new board expands the capability of your H89 computer by not only providing sound effects, but music, game paddles, and two parallel ports!

FEATURES

- * Full eight octaves of notes, plus cords
- * Sounds are totally controlled by software.
- * Envelope Generator, for complex sound effects.
- * Almost unlimited number of possible sounds.
- * Two game paddles, for program control (Sold separately)
- * Can be programmed in almost any language, including BASIC.
- * Two general purpose, parallel ports.
- * More than 50 pages of documentation is provided.
- * Fully Assembled and tested.

All sounds are totally controlled by software, and the number of possible sounds that can be generated, is almost unlimited. This board can produce the full eight octaves of the piano, and even finer frequencies, between the notes You can also combine any two, or three notes, to produce cords.

Then there's the envelope generator! The envelope generator enables you to create complex sounds, like whistling bombs and gun shots, very easily, and it can also be used to control a musical rythum.

This board employs the General Instruments sound effects chip (AY-3-8910), which I believe to be the best sound chip available, and the game paddles (Sold seperately) can dramatically improve the quality of fast-action type games on the H89.

The PSG (Programmable Sound Generator) contains 16 internal registers which can be read and written to. Once the PSG is enabled, it will continuously generate an audio output based on the data that is currently present in its 16 internal registers. The chart below represents these registers and summarizes their functions.

PSG Register Set	Bit 7 Bit 6 Bit 5 Bit 4 Bit 3 Bit 2 Bit 1 Bit 0
Registers 0-1	12 Bit Tone Frequency Control for Audio Channel A
Registers 2-3	12 Bit Tone Frequency Control for Audio Channel B
Registers 4-5	12 Bit Tone Frequency Control for Audio Channel C
Register 6	5 Bit Digital Noise Frequency Control
Register 7	Mixer Control for All Three Audio Channels
Registers 8-10	Individual Volume Control for All Three Audio Channels
Registers 11-12	16 Bit Envelope Frequency Control
Register 13	Envelope Shape Control
Registers 14-15	Two 8 Bit Parallel I/O Ports

The game paddles consist of a joy stick device which can be pushed into nine different positions. These positions are up, right, down, left, center, and then there are four more corner positions which are a combination of these. The paddles also have a red button which can be used as a fire button, etc.

When the game paddle status is read, the data from both paddles is coded into a single byte so that a single read operation is all that is required   The chart below shows the coding system used.

	Left Hand Paddle	Right Hand Paddle			
	Most Significant 4 Bits	Least Significant 4 Bits		Decimal Equivalents	Indicated Status
Sample data bytes read from game paddle status port.	0000	0000	=	(000D)	Neither paddle is being used
	1000	0000	=	(128D)	Left paddle is moving left.
	0100	0000	=	(064D)	Left paddle is moving down
	0010	0000	=	(032D)	Left paddle is moving right
	0001	0000	=	(016D)	Left paddle is moving up
	0000	1000	=	(008D)	Right paddle is moving left
	0000	0100	=	(004D)	Right paddle is moving down
	0000	0010	=	(002D)	Right paddle is moving right
	0000	0001	=	(001D)	Right paddle is moving up.
	1100	0000	=	(192D)	Left paddle moving left and down.
	0000	1100	=	(012D)	Right paddle moving left and down
	0100	0001	=	(065D)	Left down and Right up.
	1111	0000	=	(240D)	Left red button is depressed.
	0000	1111	=	(015D)	Right red button is depressed.
	1111	1111	=	(255D)	Both red buttons are depressed
	1111	0001	=	(241D)	Left red button is depressed and right paddle is moving up.

This board can be connected to either P504, or P505 on the CPU board of your H89 computer, and it can be configured to operate on any of the three serial ports, so long as the port is not being used for anything else. The Sound Effects Board will operate on an H88, however it is not physically compatible with the H8 buss.

The Sound Effects Board comes complete with everything you need to create sound effects, including an audio cable for connection to an audio amplifier, two complete manuals, demonstration programs, and a schematic

Imagine how this device could expand the power of your H89, and improve your programs, by adding this new dimension, sound!

Send To: C. D. Montgomery

[ ] Enclosed is $125.00 as payment for the sound effects board.
[ ] Enclosed is an additional $15.0 each for ____ game paddles.
[ ] Please send more information.

Name ____

Address _____________________________________

Texas residents please add 5% sales tax.

City ______________ State _______ Zip _________

The price for the board is $125.00 postage paid. Game paddles are an additional $15.00 each, and the board can operate a set of two paddles.

THE H89/Z89 SOUND EFFECTS BOARD SOFTWARE PACKAGE VOLUME I

It's amazing how the addition of sound can improve the quality of software on the H89/Z89 Once you have seen and heard real-time fast action games that make use of both the game paddles and sound effects, it will be the only type of computer game you will ever want to play! Of course, the sound effects have many applications besides games. Music and many types of sound imitations are also possible.

This software package includes two top quality Assembly Language games, that really show off the capabilities of the Sound Effects Board, and are a lot of fun to play Also included in the package are several Basic programs which produce sound imitations and some unusual effects.

The first of these games is CROSS-UP version 2 0. CROSS-UP is a fast action game with real time competition between two players. You'll feel like your at the Arcade! Two players attempt to trap each other in their paths, and fire laser blasts to hit the other player. The game board perpetually wraps around in all directions, so that there are no edges, and when you fire a shot that hits something you'll really hear the explosion!

Also included is REBOUND version 2.0. REBOUND is another real time, fast moving game in which one player tries to keep a ball bouncing, and knocking out as many bricks in the wall as he can This game also makes full use of the Sound Effects Board, game paddles, and the graphics capabilities of the H89/Z89 computer.

This package also contains a number of sound imitation programs written in Basic. You'll hear the Gunfight at the OK Corral, Pearl Harbor, Frankenstein's Laboratory, Sirens, Helicopters, etc.

This software requires an H89/Z89 computer with 48K memory, HDOS, the Sound Effects Board, and some of the programs in the package require one, or both game paddles. The Software Package will be sent to you on a single 5.25 inch disk, and the price of the package is $30.00 postage paid.

Send To: SigmaSoft and Systems
C. D. Montgomery

Name ______________________________

Address ____________________________

City ____________ State ______ Zip ________

Texas residents please add 5% sales tax.

Please send Check or Money Order for the amount of $30.00 postage paid.