1-IC PROGRAMMABLE

Need to control music or sound effects from software without overtaxing your computer? Here are two new IC's that make it possible.

BONAVENTURA ANTONY PATURZO

REALISM IS ONE OF THE MAIN ATTRACtions of the microprocessor-based games (handheld and video) that have become so popular in recent years. That realism is enhanced by the ingenious sound effects that are an important part of most of those games.

Hobbyists and experimenters also want realism in their projects, but duplicating those sound effects is not that easy. A microprocessor can create sounds with just a speaker, but considering the processor time required to sustain even a single note, it would be incapable of doing much else.

General Instruments has developed two LSI circuits that have solved that problem. The AY-3-8910 and AY-3-8912 PSG's (*P*rogrammable *S*ound *G*enerator) are designed to work with a microprocessor/microcomputer to produce complex sounds under software control. The AY-3-8910 has two 8-bit parallel ports and comes in a 40-lead DIP package. The AY-3-8912 has one port and 28 leads.

The PSG's have a wide range of applications. Those devices can create a simple noise such as a gunshot or explosion for use with an electronic game, or they can create complex musical notes and chords for use with even high-end electronic instruments. You

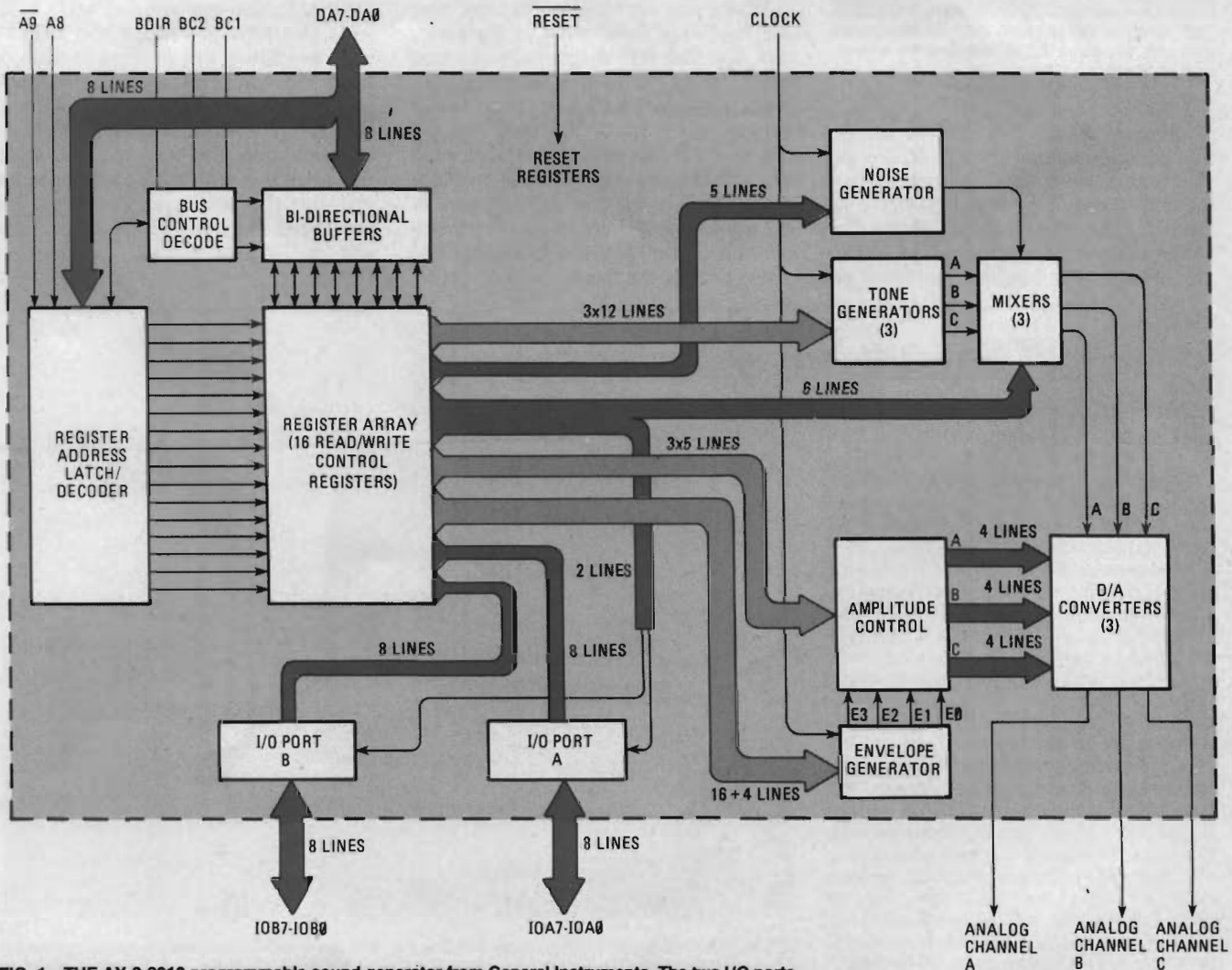

FIG. 1—THE AY-3-8910 programmable sound generator from General Instruments. The two I/O ports connect the host computer with the outside world.

SOUND GENERATOR

may even want to use one of the PSG's to generate sound effects for the Unicorn-1 robot. (*Note: If you come up with a sound effects circuit for the Unicorn-1, send it to us at: Editorial Department, **Radio-Electronics**, 200 Park Ave. S., New York, NY 10003. We'll pass the better ones along to our readers*—Editor.)

To generate sounds, the host processor gives the PSG instructions as to what sound is to be produced by writing to specific registers within the PSG. Once done, the processor is free to do other chores while the PSG generates the desired sounds.

The PSG

The AY-3-8910/AY-3-8912 contain the following sound generating blocks (Fig. 1):

Tone generators — three square-wave generators which, depending on the clock frequency used, can produce tones from sub-audible to supra-audible in frequency.

Noise generator — produces a frequency-modulated pseudo-random pulse-width squarewave output.

Mixers — combine the outputs of the tone generators and the noise generator (there's one mixer for each channel).

Amplitude control — provides the internal D/A (*D*igital to *A*nalog) converters with either a fixed or variable amplitude pattern. The fixed amplitude pattern is under control of the CPU, while the variable amplitude pattern comes from use of the envelope generator.

Envelope generator — provides amplitude modulation of the output from each mixer. Both shape and cycle of the envelope can be set by the user.

D/A converters — the three converters each produce an output signal of up to 16 levels as determined by the amplitude control.

Additionally, the I/O ports can be used to interface the host processor and the outside world. (Refer to Fig. 2 for a pin-out diagram of the PSG).

Sixteen registers control the sound-generating blocks and I/O ports of the PSG. Those registers are memory-mapped and appear to the processor as a 16-word block out of 1024 possible addresses. The four low-order data/address bits on the PSG (DØ-DA3) select one of the 16 registers. Data/address bits DA4-DA7 must be low, pin A8 must be high, and pin $\overline{A9}$ must be low when selecting a register. After the desired register has been selected, DAØ-DA7 are used for either a read-from, or write-to, PSG operation. The bus direction input (BDIR-pin 18), BC1 (pin 20), and BC2 (pin 19) are used to select whether to latch an address into the PSG, perform a write-to or read-from PSG, or go into the inactive state. Table 1 summarizes the operations.

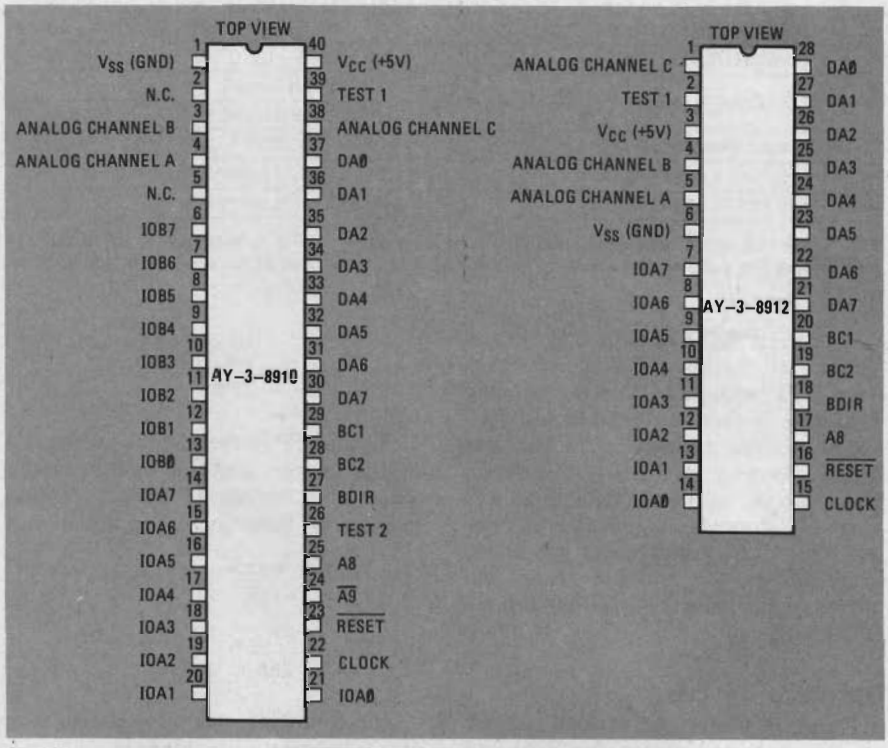

FIG. 2—PINOUT for the AY-3-8910 and the AY- 3-8912 programmable sound generators.

TABLE 1

BDIR	BC1	BC2	PSG FUNCTION
Ø	1	Ø	INACTIVE — the PSG DA7-DAØ are in a high impedance state.
Ø	1	1	READ FROM PSG — causes contents of currently addressed register to appear on DA7-DAØ.
1	1	Ø	WRITE TO PSG — latch data on DA7-DAØ into currently addressed register.
1	1	1	LATCH ADDRESS — the PSG DA7-DAØ contain a register address (select the register whose address appears on DA7-DAØ).

Before going into the different operations of the PSG, it is necessary to understand the proper sequence of signals to and from the devices. During a register-addressing sequence, the bus controls (see Table 1) should assume an "inactive" state and then a "latch address" state; the address is then placed on the bus, followed by an "inactive" state. A write-to-PSG sequence would involve sending "inactive", placing the data on the bus, sending a

REGISTER \ BIT		B7	B6	B5	B4	B3	B2	B1	B0
R0	CHANNEL A TONE PERIOD	8-BIT FINE TUNE A							
R1						4-BIT COARSE TUNE A			
R2	CHANNEL B TONE PERIOD	8-BIT FINE TUNE B							
R3						4-BIT COARSE TUNE B			
R4	CHANNEL C TONE PERIOD	8-BIT FINE TUNE C							
R5						4-BIT COARSE TUNE C			
R6	NOISE PERIOD				5-BIT PERIOD CONTROL				
R7	ENABLE	IN/OUT		NOISE			TONE		
		IOB	IOA	C	B	A	C	B	A
R10	CHANNEL A AMPLITUDE				M	L3	L2	L1	L0
R11	CHANNEL B AMPLITUDE				M	L3	L2	L1	L0
R12	CHANNEL C AMPLITUDE				M	L3	L2	L1	L0
R13	ENVELOPE PERIOD	8-BIT FINE TUNE E							
R14		8-BIT COARSE TUNE E							
R15	ENVELOPE SHAPE/CYCLE					CONT	ATT	ALT	HOLD
R16	I/O PORT A DATA STORE	8-BIT PARALLEL I/O ON PORT A							
R17	I/O PORT B DATA STORE	8-BIT PARALLEL I/O PORT B							

FIG. 3—DETAIL of the read/write control resister array shown in Fig. 1. Note registers R16 and R17 that restore the parallel port taken up by the PSG and, in the case of the AY-3-8910, add another.

"write-to PSG" signal, and finally sending an "inactive" signal. A read-from-PSG sequence involves sending "inactive", then a "read-from-PSG" signal, reading the data on the bus, and finally sending an "inactive" signal. Note that the read/write data sequences normally (though not necessarily) follow the register-addressing operation. That allows for multiple reads and writes of the same register without re-addressing.

Operation

Figure 3 shows the various register arrays contained within the PSG and their corresponding functions. Bits B7-B0 correspond to DA7-DA0 once the specific register has been selected.

As seen in Fig. 3, registers R0-R5 control the frequencies of the three tone generators. Note that each channel (the PSG has 3 independently-programmable analog output channels) has an eight-bit fine-tune and four-bit coarse-tune register. The equations for the tone frequency output are:

$$1.\ f_T = \frac{f_{CLOCK}}{16TP_{10}}$$

$$2.\ TP_{10} = 256CT_{10} + FT_{10}$$

Where: f_T = desired tone frequency

f_{CLOCK} = input clock frequency

TP_{10} = decimal equivalent of the tone period bits TP11—TP0.

CT_{10} = decimal equivalent of the coarse-tune register bits B3—B0 (TP11—TP8)

FT_{10} = decimal equivalent of the fine-tune register bits B7—B0 (TP7—TP0)

Note that TP is the 12-bit combination of the coarse- and fine-tune registers. Equation 2 breaks TP into its component parts. Rearranging the equations:

$$a.\ TP_{10} = \frac{f_{CLOCK}}{16f_T}$$

$$b.\ CT_{10} + \frac{FT_{10}}{256} = \frac{TP_{10}}{256}$$

The following examples should help clarify the math involved:

Example 1:

f_T = 1 kHz

f_{CLOCK} = 2 MHz

$$TP_{10} = \frac{2\times10^6}{16(1\times10^3)} = 125$$

Substituting this result in equation (b):

$$CT_{10} + \frac{FT_{10}}{256} = \frac{125}{256}$$

$CT_{10} = 0 = 0000$ (B3—B0)

$FT_{10} = 125_{10} = 01111101$ (B7—B0)

Example 2:

f_T = 100 Hz

f_{CLOCK} = 2 MHz

$$TP_{10} = \frac{2\times10^6}{16(1\times10^2)} = 1250$$

Substituting this result in equation (b):

$$CT_{10} + \frac{FT_{10}}{256} = \frac{1250}{256} = 4 + \frac{226}{256}$$

$CT_{10} = 4_{10} = 0100$ (B3—B0)

$FT_{10} = 226_{10} = 11100010$ (B7—B0)

Register R6 controls the noise-generator frequency. The equation for determining the noise frequency is:

$$f_N = \frac{f_{CLOCK}}{16\ NP_{10}}$$

Where: f_N = desired noise frequency

f_{CLOCK} = input clock frequency

NP_{10} = decimal equivalent of the noise period register bits B4—B0.

Rearranging the equation:

$$NP_{10} = \frac{f_{CLOCK}}{16\ f_N}$$

Using that second equation, you can solve directly for the required value of NP for a specific noise frequency.

Register 7 controls the three noise/tone mixers and the two I/O ports (one I/O port in the 8912 IC). Bits B0-B2 control whether or not a tone is enabled and bits B3-B5 enable/disable the noise on each analog output channel. Two of the truth tables shown in Table 2 summarize the enable/disable conditions. Bits B6 and B7 control the status of the I/O ports as shown in the third truth table in Table 2.

Simply disabling the noise and tone generators does not turn off an analog output channel. Turning a channel off is done by writing all zeroes into the channel's amplitude-control register.

The outputs of the three mixers are sent to three D/A converters. Each D/A converter's output amplitude is either set to a user-specified level or modulated by the PSG's internal envelope gen-

TABLE 2

NOISE ENABLE TRUTH TABLE:

R7 Bits B5	B4	B3	Noise Enabled on Channel		
0	0	0	C	B	A
0	0	1	C	B	—
0	1	0	C	—	A
0	1	1	C	—	—
1	0	0	—	B	A
1	0	1	—	B	—
1	1	0	—	—	A
1	1	1	—	—	—

TONE ENABLE TRUTH TABLE:

R7 Bits B2	B1	B0	Tone Enabled on Channel		
0	0	0	C	B	A
0	0	1	C	B	—
0	1	0	C	—	A
0	1	1	C	—	—
1	0	0	—	B	A
1	0	1	—	B	—
1	1	0	—	—	A
1	1	1	—	—	—

I/O PORT TRUTH TABLE:

R7 Bits B7	B6	I/O Port Status IOB	IOA
0	0	Input	Input
0	1	Input	Output
1	0	Output	Input
1	1	Output	Output

erator. Registers 10, 11, and 12 control the amplitudes of channels A, B, and C, respectively. If the amplitude-mode bit, B4, is set to logic Ø, then the user can control the output amplitude by supplying a 4-bit code to BØ-B3. Figure 4 shows the output levels for all possible values of BØ-B3. As previously mentioned, writing all zeroes on bits BØ-B3 turns a channel off.

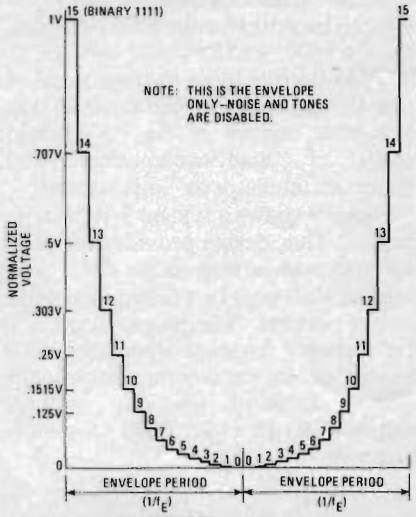

FIG. 4—OUTPUT LEVELS for all possible combinations of bits BØ-B3. Writing ØØØØ on bits BØ-B3 turns a channel off.

When the amplitude-mode bit of a channel's amplitude-control register is set high, then that channel's output amplitude is varied by the PSG's internal envelope generator. The envelope's frequency, shape, and cycle pattern can be varied, allowing quite complex envelope patterns.

Registers R13 and R14 control the period of one envelope cycle; R14 is the coarse-tune register while R13 is the fine-tune register. The envelope frequency equations are:

$$f_E = \frac{f_{CLOCK}}{256EP_{10}}$$

$$EP_{10} = 256CT_{10} + FT_{10}$$

Where: f_E = desired envelope frequency
f_{CLOCK} = input clock frequency
EP_{10} = decimal equivalent of the envelope-period bits
CT_{10} = decimal equivalent of the coarse-tune register bits B7—BØ
FT_{10} = decimal equivalent of the fine-tune register bits B7—BØ

Rearranging the equations:

a.) $EP_{10} = \frac{f_{CLOCK}}{256f_E}$

b.) $CT_{10} + \frac{FT_{10}}{256} = \frac{EP_{10}}{256}$

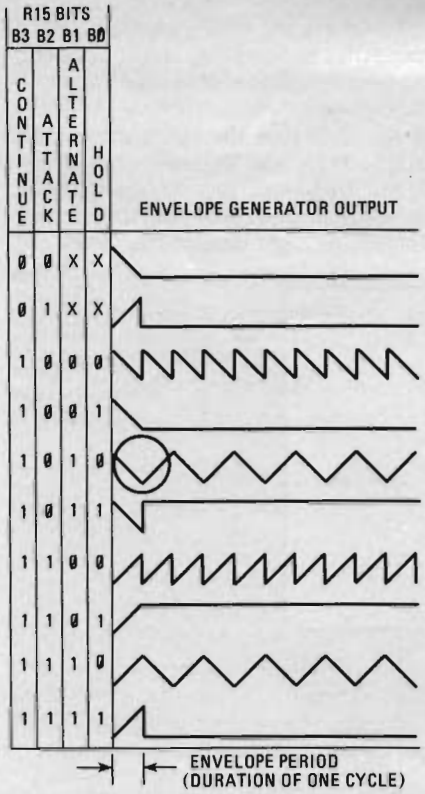

FIG. 5—THE ENVELOPE is determined by the R15 bits. The X's denote a don't-care condition.

An example, using the above equations:

$f_E = 0.5$ Hz
$f_{CLOCK} = 2$ MHz

$$EP_{10} = \frac{2\times10^6}{256(0.5)} = 15{,}625$$

Substituting this result in equation (b).

$$CT_{10} + \frac{FT_{10}}{256} = \frac{15{,}625}{256} = 61 + \frac{9}{256}$$

$CT_{10} = 61_{10}$ = ØØ111101 (B7—BØ)
$FT_{10} = 9_{10}$ = ØØØØ1ØØ1 (B7—BØ)

As shown in Fig. 3, register R15 provides control of the envelope shape and cycle pattern. Bit Ø is "hold", bit 1 is "alternate", bit 2 is "attack", and bit 3 is "continue". Figure 5 provides a graphical presentation of the patterns that can be generated, using the above functions (note: "X" in Fig. 5 indicates a don't-care condition—the bit can be either Ø or 1).

Interfacing

Figure 6 shows an economical circuit (using a color-burst crystal) that provides not only the clock for the PSG, but also the system clock for the microprocessor that controls the PSG.

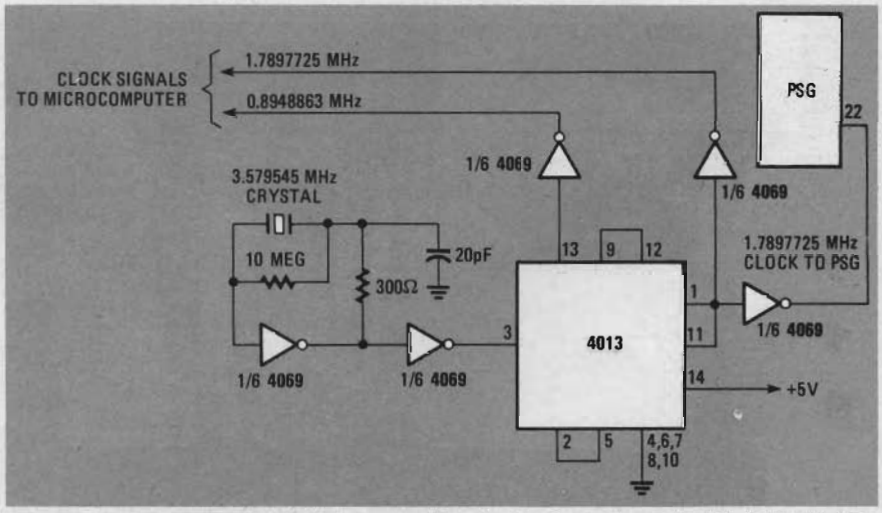

FIG. 6—A COLOR-BURST CRYSTAL is used in this circuit that provides a clock for the host microcomputer as well as for the PSG.

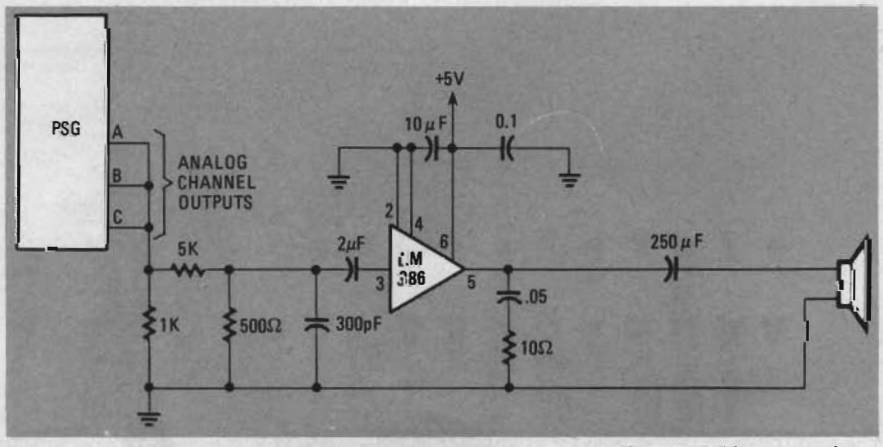

FIG. 7—THIS circit sums the PSG's three analog channels, amplifies them, and drives a speaker.

The circuit shown in Fig. 7 sums the three analog channel outputs and amplifies the sum with a standard low-voltage IC amplifier. The advantage of using that IC is obvious—you can tap the five-volt supply for the IC's biasing. Position all components as close to the IC as possible, especially the decoupling components.

To access the PSG's registers and to read-from and write-to those registers, the microprocessor must interface with the eight data/address lines (DA0-DA7) of the PSG plus the bus control lines, BDIR, BC2, and BC1. Since BC2 can be left high (see Table 1), the bus-control lines needed can be reduced to two. The PSG's eight data/address lines can be tied directly to the microprocessor's data lines if decoding is provided to place the PSG in its "inactive" mode (see Table 1) when the PSG is not being accessed.

The PSG's can be interfaced to the 6800 and 6502 microprocessors very easily, as shown in Fig. 8. A PIA (*P*eripheral *I*nterface *A*dapter) IC is used between the microprocessor and the PSG. (See **Radio-Electronics,** Nov. 1980 issue.) The setup for the 6800 is shown. For the 6502, a 6520 would be used as the PIA. Both of those microprocessors treat I/O and memory the same so that immediate control of the PSG would consist of "load accumulator" and "store accumulator at" instructions.

Figure 9 shows a typical S-100 bus interface. That design provides for reading-from and writing-to the PSG using only an 8080 IN or OUT instruction to the proper address. The proper address is the memory location where the PSG resides, set by the switches going to the 7485's. Table 3 provides software routines written in 8080 assembly language.

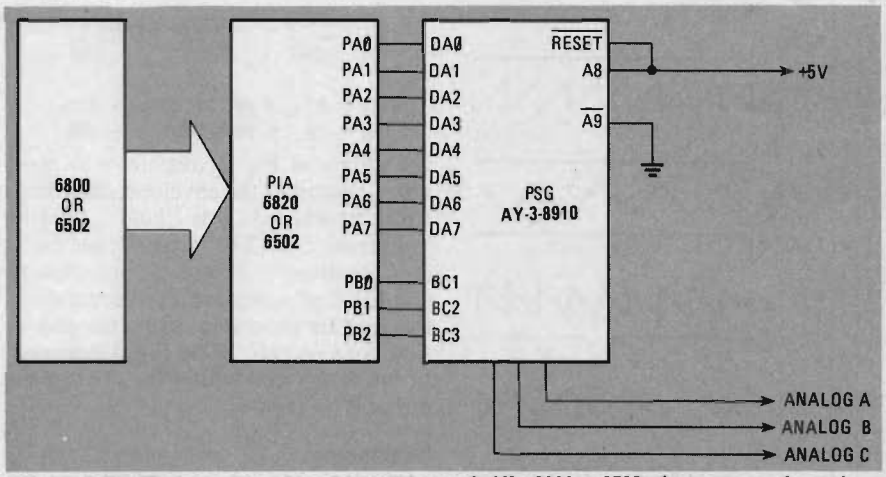

FIG. 8—A PIA (Peripheral *I*nterface *A*dapter) is needed if a 6800 or 6502 microprocessor is used.

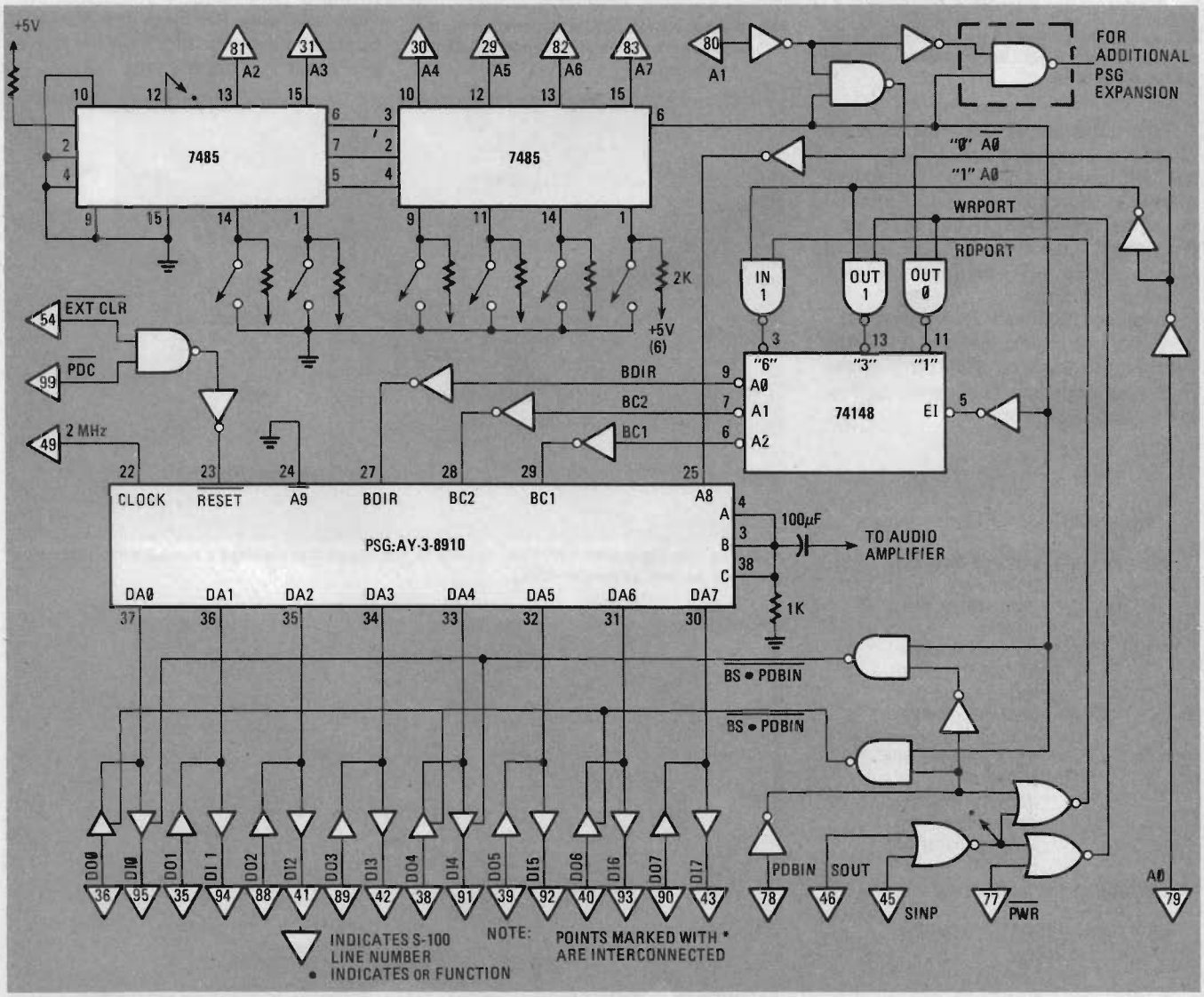

FIG. 9—TYPICAL INTERFACE for the S-100 bus. This design requires only an 8080 IN or OUT instruction to read-from or write-to the PSG.

TABLE 3

LATCH ADDRESS ROUTINE

```
PORTADDR EQU 80H ; ADDRESS TRANSFER PORT ADDRESS
PORTDATA EQU 81H ;DATA TRANSFER PORT ADDRESS
;
;THIS ROUTINE WILL TRANSFER THE CONTENTS OF
;8080 REGISTER C TO THE PSG ADDRESS REGISTER
PSGBAR   MOV   A,C ;GET C IN A FOR OUT
         OUT   PORTBAR ;SEND TO ADDRESS PORT
         RET
```

WRITE DATA ROUTINE

```
;
;
;ROUTINE TO WRITE THE CONTENTS OF 8080 REGISTER B
;TO THE PSG REGISTER SPECIFIED BY 8080 REGISTER C
;
PSGWRITE CALL  PSGBAR ;GET ADDRESS LATCHED
         MOV   A,B, ;GET VALUE IN A FOR TRANSFER
         OUT   PORTDATA ;PUT TO PSG REGISTER
         RET
```

READ DATA ROUTINE

```
;
;
;ROUTINE TO READ THE PSG REGISTER SPECIFIED
;BY THE 8080 REGISTER C AND RETURN THE DATA
;IN 8080 REGISTER B
;
PSGREAD  CALL  PSGBAR
         IN    PORTDATA ;GET REGISTER DATA
         MOV   B,A ;GET IN TRANSFER REGISTER
         RET
```

TABLE 4

GUNSHOT SOUND EFFECT

Register #	Octal Load Value	Explanation
Any not specified	000	—
R6	017	Set Noise period to mid-value.
R7	007	Enable Noise only on Channels A, B, C.
R10	020	Select full amplitude range under direct control of Envelope Generator.
R11	020	
R12	020	
R14	020	Set Envelope period to 0.586 seconds.
R15	000	Select Envelope "decay", one cycle only.

TABLE 5

EXPLOSION SOUND EFFECT

Register #	Octal Load Value	Explanation
Any not specified	000	—
R6	000	Set Noise period to max. value.
R7	007	Enable Noise only, on Channels A, B, C.
R10	020	Select full amplitude range under direct control of Envelope Generator.
R11	020	
R12	020	
R14	070	Set Envelope period to 2.05 seconds.
R15	000	Select Envelope "decay", one cycle only.

TABLE 6

LASER SOUND EFFECT

Register #	Octal Load Value	Explanation
Any not specified	000	—
R7	076	Enable Tone only on Channel A only.
R10	017	Select maximum amplitude on Channel A.
R0	060 (start)	Sweep effect for Channel A Tone period via a processor loop with approximately 3 mS wait time between each step from 060 to 160 (0.429 mS/2330Hz to 1.0ms/1000Hz).
R0	160 (end)	
R10	000	Turn off Channel A to end sound effect.

Application examples

Many computer programs, most obviously game programs, could be livened up by using appropriate sound effects. The PSG can generate a wide variety of sounds. The gunshot and explosion sound effects (Tables 4 and 5) consist of pure noise, modified by a decaying envelope. Table 6 shows a laser sound that uses a frequency-sweeping effect. Other sounds that can be created using that effect include a racing car sound, a wolf whistle, and a whistling bomb. The PSG can also produce simple tone sound effects.

Although not shown in Tables 4, 5, and 6, care must be taken to observe the proper sequence of bus-control signals to the PSG, as explained earlier in this article.

Sound effects do not have to be limited to game programs. They can signal you at the end of a long computation (if you have better things to do than watch a CRT screen for an answer); they can serve as a debugging tool, indicating audibly which direction a program is taking; and they can create a loud sound to wake you in the morning, just before the coffee pot is switched on.

There are eight full octaves of notes that the PSG can provide. Since the PSG has three channels, three note chords can also be played. Use of the various envelope patterns together with variable noise will allow the simulation of musical instruments to a certain degree. Because the tone frequency as well as output amplitude can be controlled by the microprocessor, effects such as vibrato and tremolo are possible.

Musical applications for the PSG seem endless. They will provide envelopes for an external musical instrument (such as a synthesizer). They can provide backup rhythm; with them you can easily transpose music to match your vocal range or "play" a composition you haven't yet mastered yourself. They will provide different instrumental sounds for your own compositions. And they will even give that automated coffee pot a musical tongue.

At first glance the PSG's may seem complicated to use and understand. That's because there is a wide variety of musical sounds and effects that can be generated by those devices and each of those sounds can be produced and controlled to a sometimes astonishing degree. So, although the PSG's may seem complex, that is only because of the versatility of those devices. For more information on the AY-3-8910/8912, consult General Instrument's excellent data manual. You can write to General Instrument Corporation, 600 West John St., Hicksville, NY 11802. **R-E**