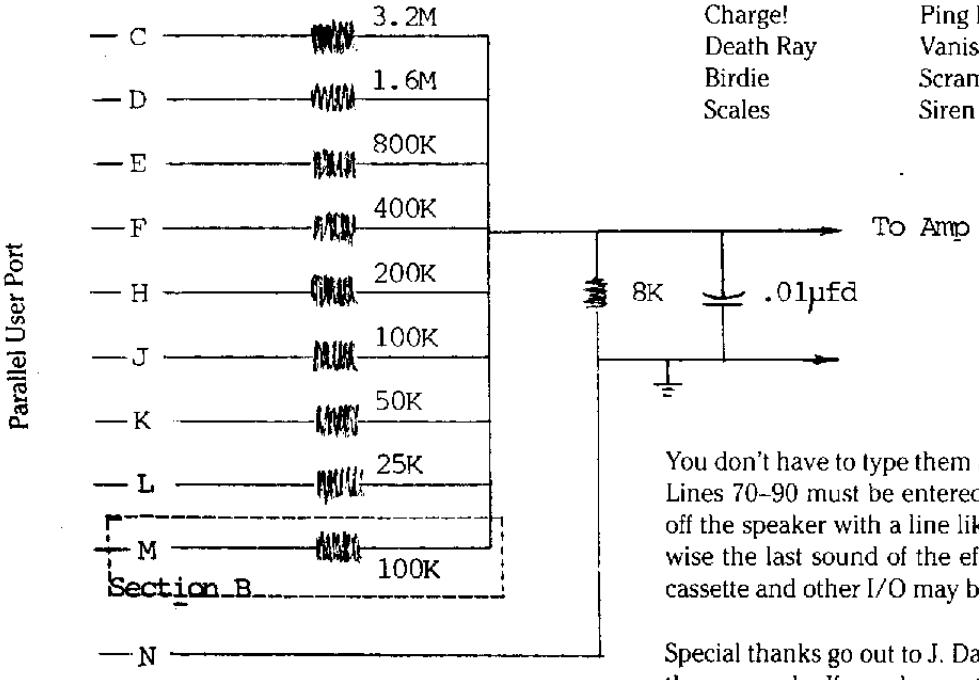
Sound Effects

Programming sound effects on any Commodore machine requires but two things: the knowledge of how it works and a vivid imagination. If you only one of the two, this next listing will give you the other.

The sound effects presented here were originally written for the PET/CBM. Late model PET/CBMs had a built-in beeper that you won't find in earlier models, the VIC 20 or the Commodore 64. But that's no reason why they won't work on these machines too. What you need is a "Digital to Analog Converter". Many firms have marketed D/As in the past, but they're ridiculously simple to make. Below is a circuit schematic of a passive D/A converter.

With the circuit in place, connect the output to the auxiliary input of any stereo system. It's probably best to use an RCA male phono jack.



The corresponding chip locations for the VIC and C64 are:

PET/CBM	VIC 20	Commodore 64
59464	37144	56582
59466	37146	56584
59467	37147	56585

If you're not inclined to build this item, it shouldn't take terribly long to convert the program to work with the internal sound already provided for in the VIC 20 or Commodore 64. For clues, take a look at VIC Sound by Dave Gzik in this issue, or Zoundz for the C64 by Howard Strasberg in the Bits and Pieces section.

The sound effects you will hear are:

Piano Gliss	Laser Fire	French Police
Bombs Away	Wolf Whistle	Dive! Dive!
Shave 'n' a Haircut	Dirge	R2D2
Charge!	Ping Pong	Raspberry
Death Ray	Vanishing Saucer	Eat My Dust
Birdie	Scramble	Bounces
Scales	Siren	Ticking Clock

You don't have to type them all in – just the ones you want. Lines 70-90 must be entered and it's good practice to turn off the speaker with a line like 160 after each effect. Otherwise the last sound of the effect will remain on, plus your cassette and other I/O may be adversely affected.

Special thanks go out to J. David of WHCI for work done on these sounds. If you have others, send them in and we'll publish another collection like this in a future issue.

To play music may require the use of data statements for the notes. The best way to set them up for ease is in this format:

DATA speaker1 note,speaker2 note,speaker3 note,duration

The corresponding read statement would then look like this:

READ S1, S2, S3, DUR

Now the basic program structure changes to this:

1 Set variables.
2 Set volume.
3 Read notes, duration from data.
4 If data is completed then exit program.
5 Turn on speaker.
6 Play it for a while.
7 Turn it off.
8 Go back to step 3
9 End reset all variables to zero
10 Data elements.

Below is a full fledged piece of music that puts the preceding logic to work. Study it carefully to see how it is structured.

Next issue I will go further into manipulating the previous ideas for special sound effects, etc.

I leave you with a section of BACH'S invention number 4. Have fun.

```
10 vol = 36878 : s1 = 36874 : s2 = 36875 : s3 = 36876
20 poke vol,5
30 read x, y, dur
40 if x = -1 then 80
50 poke s2,x : poke s3,y
60 for t = 1 to dur*18 : next
70 goto 30
80 poke vol,0 : poke s1,0 : poke s2,0 : poke s3,0
90 end
1000 data   0, 201,   8,   0, 207,   8,   0, 209
1010 data   8,   0, 215,   8,   0, 219,   8,   0
1020 data 221,   8,   0, 199,   8,   0, 221,   8
1030 data   0, 219,   8,   0, 215,   8,   0, 209
1040 data   8,   0, 207,   8, 201, 209,   8, 207
1050 data   0,   8, 209, 219,   8, 215,   0,   8
1060 data 219, 228,   8, 221,   0,   8, 199, 215
1070 data   8, 221,   0,   8, 219, 227,   8, 215
1080 data   0,   8, 209, 231,   8, 207,   0,   8
1090 data 209, 228,   8,   0, 231,   8, 219, 232
1100 data   8,   0, 235,   8, 228, 237,   8,   0
1110 data 238,   8, 207, 227,   8,   0, 238,   8
1120 data 215, 237,   8,   0, 235,   8, 227, 232
1130 data   8,   0, 231,   8, 201, 232,   8, 201
1140 data 228,   8, 228, 231,   8, 228, 232,   8
1150 data 209, 235,   8, 209, 237,   8, 215, 221
1160 data   8, 215, 237,   8, 219, 235,   8, 219
1170 data 232,   8, 221, 231,   8, 221, 228,   8
1180 data 195, 231,   8, 195, 225,   8, 225, 228
1190 data   8, 225, 231,   8, 207, 232,   8, 207
1200 data 235,   8, 209, 219,   8, 209, 235,   8
1210 data 215, 232,   8, 215, 231,   8, 219, 228
1220 data   8, 219, 225,   8, 201, 209,   8, 187
1230 data 215,   6, 187,   0,   4, 195, 215,   4
1240 data 195, 219,   4,   0, 215,   4, 135, 215
1250 data   8, 135, 209,   6, 135,   0,   8, 163
1260 data 209,   8, 175, 209,  32
5000 data  -1,  -1,  -1
```

VIC 20 Note Values

Where two values are shown, it is necessary to alternate between them to get the true note.
Voice frequency registers are 36874/5/6. Noise reg is 36877. Volume is Lo nybble of 36878. See Memory Map

	Octave 0		Octave 1		Octave 2		Octave 3	
Note	Value	Mod. Val.	Value	Mod. Val.	Value	Mod. Val.	Value	Mod. Val.
C	131		192	195	224		239	240
C#	140		197		226		240	241
D	145		200		227	228		
D#	151		203		229			
E	158		206	207	231			
F	161	162	208	209	232			
F#	166	167	211	212	233			
G	173	174	214		234	235		
G#	178		216		238	236		
A	181	182	218	219	237			
A#	185	186	220	221	237	238		
B	189	190	222	223	239			

```
70 tt=59464 : rem timer 2 low byte
80 sr=59466 : rem shift register
90 sc=59467 : rem shift register control
100 print" *** piano gliss *** "
110 poke sc,16 : poke tt,0 : poke sr,15
120 for k=0 to 100 step5
130 poke tt,k : x=tan(k) : next
140 for k=99 to 0 step-5
150 poke tt,k : x=tan(k) : next
160 poke sc,0 : poke sr,0 : poke tt,0
170 rem return
180 print" *** laser fire *** "
190 poke sc,16 : poke tt,0
200 poke sr,15 : for k=1 to 5
210 for r=0 to 100 step5
220 poke tt,r : for x=1 to 2 : next : next
230 poke tt,r : x=tan(x) : next
240 poke sc,0 : poke sr,0 : poke tt,0
250 rem return
260 print" *** french police *** "
270 poke sc,16 : poke tt,0 : poke sr,2
280 for k=1 to 4
290 for r=100 to 255 step40
300 poke tt,r : for x=1 to 500 : next
310 for r=255 to 100 step-40
320 poke tt,r : next : for x=1 to 500 : next
330 next
340 poke sc,0 : poke sr,0 : poke tt,0
350 rem return
360 print" *** bombs away *** "
370 poke sc,16 : poke tt,0 : poke sr,85
380 for r=50 to 150 : poke tt,r : for k=1 to 30 : next : next
390 poke sr,1 : poke tt,255 : for k=1 to 800 : next
400 poke sc,0 : poke sr,0 : poke tt,0
410 rem return
420 print" *** wolf whistle *** "
430 poke sc,16 : poke tt,0 : poke sr,13
440 for r=185 to 80 step-3 : poke tt,r : next
450 poke tt,0 : for k=1 to 200 : next
460 for r=205 to 105 step-3 : poke tt,r : next
470 for r=105 to 255 step3 : poke tt,r : next
480 poke sc,0 : poke sr,0 : poke tt,0
490 rem return
500 print" *** dive! dive! *** "
510 poke sc,16 : poke sr,9
520 for k=1 to 10
530 for r=250 to 180 step-1
540 poke tt,r : next : next
550 poke sc,0 : poke sr,0 : poke tt,0
560 rem return
570 print" *** shave 'n' a haircut, 2 bits *** "
580 poke sc,16 : poke tt,0 : poke sr,15 : t=3
590 poke tt,188 : for k=1 to 200 : next
600 poke tt,251 : for k=1 to 100 : next
610 poke tt,0 : for k=1 to t : next
620 poke tt,251 : for k=1 to 100 : next
630 poke tt,225 : for k=1 to 200 : next
640 poke tt,251 : for k=1 to 300 : next
650 poke tt,0 : for k=1 to 150 : next
660 poke tt,199 : for k=1 to 200 : next : t=50
670 poke tt,0 : for k=1 to t : next
680 poke tt,188 : for k=1 to 150 : next
690 poke sc,0 : poke sr,0 : poke tt,0
700 rem return
710 print" *** dirge *** "
720 poke sc,16 : poke tt,0 : gosub 750
730 poke sc,0 : poke sr,0 : poke tt,0
740 goto 890 : return
750 rem dirge
760 poke sr,15 : t=3
770 poke tt,237 : for k=1 to 300*t : next : gosub 880
780 poke tt,237 : for k=1 to 200*t : next : gosub 880
790 poke tt,237 : for k=1 to 100*t : next : gosub 880
800 poke tt,237 : for k=1 to 300*t : next : gosub 880
810 poke tt,199 : for k=1 to 300*t : next : gosub 880
820 poke tt,211 : for k=1 to 200*t : next : gosub 880
830 poke tt,237 : for k=1 to 100*t : next : gosub 880
840 poke tt,237 : for k=1 to 200*t : next : gosub 880
850 poke tt,251 : for k=1 to 100*t : next : gosub 880
860 poke tt,237 : for k=1 to 300*t : next
870 poke tt,0 : for k=1 to 5 : next : return
880 poke tt,0 : for k=1 to t : next : return
890 print" *** r2d2 *** "
900 poke sc,16 : poke tt,0 : poke sr,15
910 for k=1 to 30 : poke tt,10+100*rnd(1)
920 for i=1 to 6 : next : next
930 poke sc,0 : poke sr,0 : poke tt,0
940 rem return
950 print" *** charge! *** "
960 poke sc,16 : poke tt,0 : gosub 990
970 poke sc,0 : poke sr,0 : poke tt,0
980 goto 1080 : rem return
990 poke sr,15 : t=3
1000 poke tt,255 : for k=1 to 100 : next : gosub 1070
1010 poke tt,191 : for k=1 to 100 : next : gosub 1070
1020 poke tt,152 : for k=1 to 100 : next : gosub 1070
1030 poke tt,128 : for k=1 to 200 : next : gosub 1070
1040 poke tt,152 : for k=1 to 100 : next : t=0
     : gosub 1070
1050 poke tt,128 : for k=1 to 400 : next
1060 return
1070 poke tt,0 : for k=1 to t : next : return
```

```
1080 print " *** ping pong *** "
1090 poke sc,16 : poke tt,0 : poke sr,15
1100 for j = 1 to 5
1110 poke tt,255 : for k = 1 to 60 : next
1120 poke tt,0 : for k = 1 to 100*rnd(1)*20 : next
1130 poke tt,128 : for k = 1 to 60 : next
1140 poke tt,0 : for k = 1 to 100*rnd(1)*20 : next
1150 for x = 1 to 100 : next : next
1160 poke sr,63 : poke tt,255 : for k = 1 to 500 : next
1170 poke sc,0 : poke sr,0 : poke tt,0
1180 rem return
1190 print " *** raspberry *** "
1200 poke sc,16 : poke tt,0 : poke sr,9
1210 for k = 1 to 50 : poke tt,238 : poke tt,251 : next
1220 poke sc,0 : poke sr,0 : poke tt,0
1230 rem return
1240 print " *** death ray *** "
1250 poke sc,16 : poke tt,0 : poke sr,15
1260 for k = 1 to 200 : poke tt,150 : poke tt,200
     : poke tt,255 : next
1270 poke sc,0 : poke sr,0 : poke tt,0
1280 rem return
1290 print " *** vanishing saucer *** "
1300 poke sc,16 : poke tt,0 : poke sr,29
1310 for k = 160 to 0 step-.3 : poke tt,k : poke tt,k + 5
     : poke tt,k + 15 : next
1320 poke sc,0 : poke sr,0 : poke tt,0
1330 rem  return
1340 print " *** eat my dust *** "
1350 poke sc,16 : poke tt,0 : poke sr,3
1360 for k = 200 to 235 step.7 : poke tt,k : poke tt,k + 5
     : poke tt,k + 20
1370 for z = 1 to 20 : next : next
1380 for k = 1 to 300 : poke sc,0 : poke sr,16 : poke sc,16
     : poke sr,3 : next
1390 for k = 235 to 170 step-.6 : poke tt,k : poke tt,k + 5
     : poke tt,k + 15 : next
1400 for k = 170 to 220 step3 : poke tt,k : next
1410 for k = 220 to 129 step-.5 : poke tt,k : poke tt,k + 5
     : poke tt,k + 15 : next
1420 for k = 120 to 180 step3 : poke tt,k : next
1430 for k = 140 to 200 step2 : poke tt,k : poke tt,k + 5
     : poke tt,k + 15 : next
1440 k = 200 : poke sc,16 : poke sr,51
1450 for x = 1 to 50 : poke tt,k : poke tt,k + 7
     : poke tt,k + 14 : next
1460 poke sc,0 : poke sr,0 : poke tt,0
1470 rem return
1480 print " *** birdie *** "
1490 poke sc,16 : poke tt,0 : poke sr,85
1500 for k = 1 to 10
1510 for k1 = 152 to 56 step-8 : poke tt,k1 : next
1520 poke tt,0 : for k1 = 1 to int(rnd(1)*200) : next : next
1530 poke sc,0 : poke sr,0 : poke tt,0
1540 rem return
1550 print " *** scramble *** "
1560 poke sc,16 : poke tt,0 : poke sr,85
1570 for k = 1 to 20
1580 for k1 = 1 to 14 : poke tt,k1*16 : next : next
1590 poke sc,0 : poke sr,0 : poke tt,0
1600 rem return
1610 print " *** bounces *** "
1620 for n = 255 to 20 step-5 : poke sc,16 : poke sr,15
     : poke tt,n : poke sc,0
1630 for nn = 1 to 100 : next : next : poke sc,0
1640 rem return
1650 print " *** musical scales *** "
1660 gosub 1670 : goto 1850 : rem return
1670 gosub 1730
1680 poke tt,239 : poke sc,16 : poke sr,tc
1690 for n = 1 to 50
1700 read nn : poke tt,nn
1710 for z = 1 to 50 : next
1720 nextn : restore : poke sc,0 : return
1730 poke sc,0
1740 tc = 15 : gosub 1680 : gosub 1770
1750 tc = 51 : gosub 1680 : gosub 1770
1760 tc = 85 : gosub 1680 : gosub 1770 : return
1770 for n = 1 to 100 : next : return
1780 data 239, 225, 213, 201, 190, 179, 169
1790 data 159, 150, 142, 134, 127, 119, 113
1800 data 106, 100,  95,  89,  84,  80,  75
1810 data  71,  67,  63,  60,  60,  63,  67
1820 data  71,  75,  80,  84,  89,  95, 100
1830 data 106, 113, 119, 127, 134, 142, 150
1840 data 159, 169, 179, 190, 201, 213, 225, 239
1850 print " *** siren *** "
1860 poke sc,16 : poke sr,15
1870 for n = 1 to 4 : for nn = 250 to 80 step-2 : poke tt,nn
     : next
1880 for nn = 80 to 250 step 2 : poke tt,nn : next : next
1890 poke sc,0
1900 rem return
1910 print " *** ticking clock *** "
1920 m = 150 : mm = 50 : for n = 1 to 21
1930 poke tt,m + mm : poke sc,16 : poke sr,15
     : poke sc,0
1940 for nn = 1 to 250 : next : mm = -mm : next
1950 poke sc,0
1960 rem return
```