Interrupt-driven

Many popular video games nowadays have music ringing in the background during play. The Amstrad CPC-464 computer has some excellent facilities for sound generation but attempting to add background music to a Basic program using the Every instruction generally gives poor results with the game being slowed down considerably.

The assembly language program presented here allows interrupt driven background music to be added to Basic programs through the use of some simple RSX-Resident System extension-commands. It also allows the us of one, two or all three sound channels.

The following commands are provided. Note that each one must be preceded by a vertical bar (Shift '@') symbol. A parameter enclosed in square brackets is optional.

(1). MUSICON ,t, a1 (,a2) (,a3) — turn on music. where t is the tempo (the length in fiftieths of a second, of the shortest available note) and a1,a2 and a3 are two less than the addresses of the data for channels 1,2 and 3 respectively.

(2). MUSICOFF — turn off music.

(3). VOLUME v,1 (,v2) (,v3) — set for channel (s), where v1, v2 and v3 are the respective amplitudes of channels 1,2 and 3.

(4). PAUSE — stop playing music.

(5). CONTINUE — continue playing music after pausing.

To enter the program, you can either enter and assemble the source code — listing 1 — using an assembler. I have used Devpac but any good one should do). Or if you do not have an assembler I have provided a Basic loader program — listing 2 — which should be typed in and Run to store the code in memory.

Once the code is in memory, it can be saved on tape (or disc) using the command

SAVE "MUSIC",B,&A000,&2A0 (enter)

To make the computer recognise the new commands the instruction CALL &A1C5 should be given. At this point the program can be tested by entering and Running listing 3 which tells the computer to repeatedly play all the notes available until a Musicoff command is given.

Any program using the extension commands should include the following line to load the code and log-on the commands MEMORY &9FFF: LOAD "!MUSIC":CALL&A1C5

The data for the music to be played through a channel is arranged as follows:-

byte 1 — duration of note 1 (1-127), byte 2 (upper nybble) — note 1 octave (0-5), byte 2 (lower nybble) — note 1 number (0-11), byte 3 — duration of note 2, byte 4 (upper nybble) — note 2 octave, byte 4 (lower nybble) — note 2 number....., 0

Once the music playing routine discovers the 0 in place of a duration number it then loops back to the beginning again. If bit 7 of the duration number is

Listing 1.

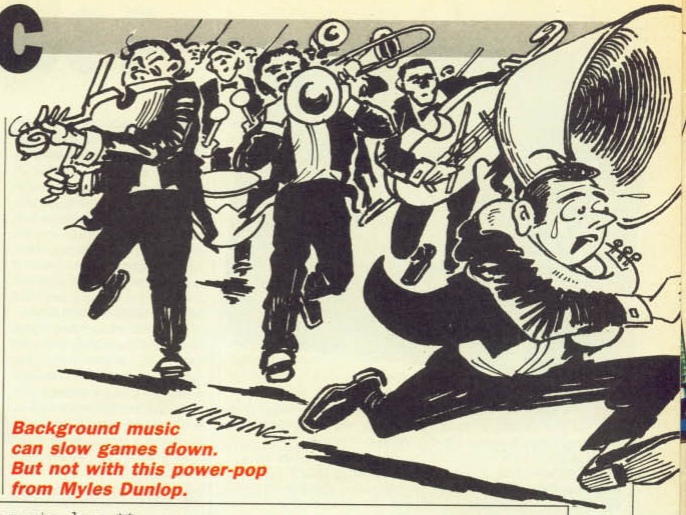
```
10 ; Amstrad Background Music
20 ; (c) 1985 Times Journal
30
40          ORG &A000
50 TICKER:  DEFS 2
60 EBLOCK:  DEFS 7
70 POINTR:  DEFS 2
80 DELAY:   DEFS 1
90 DATA:    DEFS 6
100 NCHAN:  DEFS 1
110 TEMPO:  DEFS 1
120 VOLUME: DEFS 6
130 DELT:   DEFW 0
140 MUSON:  CALL RESMP
150         CALL RESSND
160         LD  C,0
170         LD  A,7
180         CALL &BD34
190         LD  HL,EBLOCK
200         LD  B,129
210         LD  DE,MUSIC
220         CALL &BCE
230         LD  HL,TICKER
240         LD  DE,D0
250         LD  BC,(TEMPO)
260         CALL &BCE7
270         RET
280 MUSOFF: LD  HL,TICKER
290         CALL &BD0A
300         CALL &BCEC
310 RESSND: LD  D,9
320 RESSND: LD  A,7
330         ADD A,D
340         LD  C,0
350         CALL &BD34
360         DEC D
370         JR  NZ,RESSND
380         RET
390 RESMP:  LD  HL,(DATA)
400         LD  (POINTR),HL
410         LD  HL,(DATA+2)
420         LD  (POINTR+2),HL
430         LD  HL,(DATA+4)
440         LD  (POINTR+4),HL
450         LD  A,1
460         LD  (DELAY),A
470         LD  (DELAY+1),A
480         LD  (DELAY+2),A
490         RET
500 MUSIC:  LD  HL,DELAY
510         LD  BC,DATA
520         LD  DE,POINTR
530         XOR A
540         CALL MUSIC2
550         LD  A,(NCHAN)
560         CP  1
570         RET Z
580         LD  A,1
590         INC HL
600         INC DE
610         INC BC
620         INC DE
630         INC DE
640         CALL MUSIC2
650         LD  A,(NCHAN)
660         CP  2
670         RET Z
680         LD  A,2
690         INC HL
700         INC BC
710         INC BC
720         INC DE
730         INC DE
740         CALL MUSIC2
750         RET
760 MUSIC2: DEC (HL)
770         RET NZ
780         PUSH HL
790         PUSH BC
800         PUSH DE
810         PUSH AF
820         LD  (DEL),HL
830         LD  A,(DE)
840         LD  L,A
850         INC DE
860         LD  A,(DE)
870         LD  H,A
880         INC HL
890         INC HL
900         LD  A,H
910         LD  (DE),A
920         DEC DE
930         LD  A,L
940         LD  (DE),A
950         LD  A,(HL)
960         CP  0
970         JR  NZ,MUSIC3
980         LD  A,(BC)
990         LD  (DE),A
1000        INC DE
1010        INC BC
1020        LD  A,(BC)
1030        LD  (DE),A
1040        LD  A,1
1050        LD  HL,(DEL)
1060        LD  (HL),A
1070 MUSIC6: POP AF
1080        PUSH AF
1090        ADD A,8
1100        LD  C,0
1110        CALL &BD34
1120 MUSIC4: POP AF
1130        POP DE
1140        POP BC
1150        POP HL
1160        RET
1170 MUSIC3: BIT 7,A
1180        JR  Z,MUSIC5
1190        RES 7,A
1200        LD  BC,(DEL)
1210        LD  (BC),A
1220        JR  MUSIC4
1230 MUSIC5: LD BC,(DEL)
1240        LD  (BC),A
1250        INC HL
1260        PUSH HL
1270        LD  A,(HL)
1280        AND 240
1290        SRL A
1300        LD  L,A
1310        SLA A
1320        ADD A,E
1330        LD  E,A
1340        LD  D,0
1350        LD  HL,NOTDAT
1360        ADD HL,DE
1370        LD  B,H
1380        LD  C,L
1390        POP HL
1400        LD  A,(HL)
1410        AND 15
1420        LD  L,A
1430        SLA E
1440        LD  H,B
1450        LD  L,C
1460        ADD HL,DE
1470        POP AF
1480        PUSH AF
1490        SLA A
1500        PUSH AF
1510        LD  C,(HL)
1520        CALL &BD34
1530        POP AF
1540        INC A
1550        INC HL
1560        LD  C,(HL)
1570        CALL &BD34
1580        POP AF
1590        PUSH AF
1600        LD  HL,VOLUME
1610        LD  E,A
1620        LD  D,0
1630        ADD HL,DE
1640        ADD A,8
1650        LD  C,(HL)
1660        CALL &BD34
1670        JR  MUSIC4
1680 NOTDAT: DEFW &077,&070C
1690        DEFW &06A7,&0647
1700        DEFW &05ED,&0598
1710        DEFW &0547,&04FC
1720        DEFW &04D4,&0470
1730        DEFW &0431,&03F4
1740 ;
1750        DEFW &03DC,&0386
1760        DEFW &0353,&0324
1770        DEFW &02F6,&02CC
1780        DEFW &02A4,&027E
1790        DEFW &025A,&0238
1800        DEFW &0218,&01FA
1810 ;
1820        DEFW &1DE,&1C3
1830        DEFW &1AA,&192
1840        DEFW &17B,&166
1850        DEFW &152,&13F
1860        DEFW &12D,&11C
1870        DEFW &10C,&FD
1880 ;
1890        DEFW &EF,&E1
1900        DEFW &D5,&C9
1910        DEFW &BE,&B3
1920        DEFW &A9,&9F
1930        DEFW &96,&8E
1940        DEFW &86,&7F
1950 ;
1960        DEFW &77,&71
1970        DEFW &6A,&64
1980        DEFW &5F,&59
1990        DEFW &54,&50
2000        DEFW &4B,&47
2010        DEFW &43,&3F
2020 ;
2030        DEFW &3C,&38
2040        DEFW &35,&32
2050        DEFW &2F,&2D
2060        DEFW &2A,&28
2070        DEFW &26,&24
2080        DEFW &22,&20
2090 ;
2100 ; add extension commands
2110 ;
2120 LOCRSX: LD  BC,COMTAB
2130        LD  HL,SPACE
2140        CALL &BCD1
2150        RET
2160 SPACE: DEFS 4
2170 COMTAB: DEFW NTABLE
2180        JP  MONRSX
2190        JP  MUSOFF
2200        JP  PAUSE
2210        JP  CONT
2220        JP  VOLRSX
2230 NTABLE: DEFM "MUSI"
2240        DEFB "C","O","N"+128
2250        DEFM "MUSI"
2260        DEFB "C","O","F","F"+128
2270        DEFM "PAUS"
2280        DEFB "E"+128
2290        DEFM "CONT"
2300        DEFB "I","N","U","E"+128
2310        DEFM "VOLU"
2320        DEFB "M","E"+128
2330        DEFB 0
2340 ;
2350 ; music-on
2360 ;
2370 MONRSX: PUSH AF
2380        PUSH IX
2390        CALL MUSOFF
2400        POP IX
2410        POP AF
2420        CP  2
2430        RET C
2440        JP  Z
2450        RET NC
2460        JP  Z
2470        JR  Z,MON5
2480        CP  3
2490        JR  Z,MON4
2500        LD  A,3
2510        LD  (NCHAN),A
2520        LD  L,(IX)
2530        LD  H,(IX+1)
2540        LD  (DATA+4),HL
2550        INC IX
2560        INC IX
2570 MON3:  LD  A,(IX)
2580        LD  H,(IX+1)
2590        LD  (DATA+2),HL
2600        INC IX
2610        INC IX
2620 MON2:  LD  L,(IX)
2630        LD  H,(IX+1)
2640        LD  (DATA),HL
2650        INC IX
2660        INC IX
2670        LD  A,(IX)
2680        LD  (TEMPO),A
2690        CALL MUSON
2700        RET
2710 MON4:  LD  A,2
2720        LD  (NCHAN),A
2730        JR  MON3
2740 MON5:  LD  A,1
2750        LD  (NCHAN),A
2760        JR  MON2
2770 ;
2780 ; PAUSE
2790 ;
2800 PAUSE: CALL RESSND
2810        LD  HL,EBLOCK
2820        CALL &BD0A
2830        RET
2840 ;
2850 ; CONTINUE
2860 ;
2870 CONT:  XOR A
2880        LD  (EBLOCK+2),A
2890        RET
2900 ;
2910 ; VOLUME
2920 ;
2930 VOLRSX: CP  0
2940        RET Z
2950        CP  4
2960        RET NC
2970        CP  1
2980        JR  Z,VOL2
2990        CP  2
3000        JR  Z,VOL3
3010        LD  A,(IX)
3020        LD  (VOLUME+2),A
3030        INC IX
3040        INC IX
3050 VOL3:  LD  A,(IX)
3060        LD  (VOLUME+1),A
3070        INC IX
3080        INC IX
3090 VOL2:  LD  A,(IX)
3100        LD  (VOLUME),A
3110        RET
```

music

set (just add 128), then the note is a rest otherwise 1 is the shortest note available, 2 is twic as long, 3 is three times as long etc. The octaves available correspond to octaves −2 to +3 shown in the CPC-464 manual appendix VII and in the firmware manual appendix VIII. e.g. bytes 2,52 define a note of duration 2, octave 3, note 4; 129, 0 define a rest of duration 1. A simple formula for working ot the second byte of a note is therefore:

byte stored = 16*octave + note

If you do not wish to type in the program I can supply copies of it on cassette for £4 each. Please include a 24 pence SAE. Write to: M.B.L. Dunlop, 19 Drordige, Dartington, Totnes, Devon, TQ9 6JG.



Background music can slow games down. But not with this power-pop from Myles Dunlop.

```
10 REM ** background music demo **
20 REM ** By Myles Dunlop       **
30 REM
35 CLS:PRINT"Please wait..."
40 MEMORY &9FFF:LOAD "!",&A000:CALL &A1C5
50 a=&A300:GOSUB 1000
60 a=&A380:GOSUB 1000
70 |VOLUME,15,15
80 |MUSICON,10,&A300-2,&A380-2
90 MODE 2:LIST
1000 READ b:IF b=-1 THEN RETURN
1010 POKE a,b:a=a+1:GOTO 1000
1020 REM
1030 REM channel 1 data
1040 DATA 152,0,3,&37,131,0,134,0,3,&37,136,0,1,&37,3,&40,1,&3b,1,&39,1,&3b
1050 DATA 2,&40,4,&37,3,&40,1,&3b,1,&39,1,&3b,2,&40,4,&37,3,&40,1,&3b,1,&39
1060 DATA 1,&3b,2,&40,3,&37,141,0,0,-1
1070 REM
1080 REM channel 2 data
1090 DATA 3,&34,3,&32,6,&30,3,&34,3,&32,6,&30,131,0,3,&35,6,&34,131,0,3,&35
1100 DATA 5,&34,129,0,3,&34,3,&32,6,&30,3,&34,3,&32,6,&30,3,&34,3,&32,3,&30
1110 DATA 130,0,1,&35,3,&34,3,&32,6,&30,0,-1
```

Listing 2.

```
10 REM machine code loader
20 check=0
30 FOR I=&A000 to &A2A0
40 READ J:check=check+J
50 NEXT
1000 DATA 0,0,0,0,0,0,0,0
1010 DATA 0,0,0,0,0,0,0,0
1020 DATA 0,0,0,0,0,0,0,0
1030 DATA 0,0,0,0,0,0,0,0
1040 DATA 0,0,0,0,0,0,0,205
1050 DATA 100,160,205,86,160,14,60,62
1060 DATA 7,205,52,189,33,6,160,6
1070 DATA 129,17,130,160,205,239,188,33
1080 DATA 0,160,17,30,0,237,75,29
1090 DATA 160,205,233,188,201,33,0,160
1100 DATA 205,10,189,205,236,188,22,3
1110 DATA 62,7,130,14,0,205,52,189
1120 DATA 21,32,245,201,42,22,160,34
1130 DATA 13,160,42,24,160,34,15,160
1140 DATA 42,26,160,34,17,160,62,1
1150 DATA 50,19,160,50,20,160,50,21
1160 DATA 160,201,33,19,160,1,22,160
1170 DATA 17,13,160,175,205,176,160,58
1180 DATA 28,160,254,1,200,62,1,35
1190 DATA 3,3,19,19,205,176,160,58
1200 DATA 28,160,254,2,200,62,2,35
1210 DATA 3,3,19,19,205,176,160,201
1220 DATA 33,192,229,197,213,245,34,37
1230 DATA 160,26,111,19,26,103,35,35
1240 DATA 124,18,27,125,18,126,254,0
1250 DATA 32,26,10,18,19,3,10,18
1260 DATA 62,1,42,37,160,119,241,245
1270 DATA 198,8,14,0,205,52,189,241
1280 DATA 209,193,225,201,203,127,40,9
1290 DATA 203,191,237,75,37,160,2,24
1300 DATA 229,237,75,37,160,2,35,229
1310 DATA 126,230,240,203,63,95,203,39
1320 DATA 131,95,22,0,33,53,161,25
1330 DATA 68,77,225,126,230,15,95,203
1340 DATA 35,96,105,25,241,245,203,39
1350 DATA 245,78,205,52,189,241,60,35
1360 DATA 78,205,52,189,241,245,33,31
1370 DATA 160,95,22,0,25,198,8,78
1380 DATA 205,52,189,24,170,119,7,12
1390 DATA 7,167,6,71,6,237,5,152
1400 DATA 5,71,5,252,4,212,4,112
1410 DATA 4,49,4,244,3,220,3,134
1420 DATA 3,89,3,36,3,246,2,204
1430 DATA 2,164,2,126,2,90,2,56
1440 DATA 2,24,2,250,1,222,1,195
1450 DATA 1,170,1,146,1,123,1,102
1460 DATA 1,82,1,63,1,45,1,28
1470 DATA 1,12,1,253,0,239,0,225
1480 DATA 0,213,0,201,0,190,0,179
1490 DATA 0,169,0,159,0,150,0,142
1500 DATA 0,134,0,127,0,119,0,113
1510 DATA 0,106,0,100,0,95,0,89
1520 DATA 0,84,0,80,0,75,0,71
1530 DATA 0,67,0,63,0,60,0,56
1540 DATA 0,53,0,50,0,47,0,45
1550 DATA 0,42,0,40,0,38,0,36
1560 DATA 0,34,0,32,0,1,211,161
1570 DATA 33,207,161,205,209,188,201,0
1580 DATA 0,0,0,228,161,195,7,162
1590 DATA 195,77,160,195,98,162,195,108
1600 DATA 162,195,113,162,77,85,83,73
1610 DATA 67,79,206,77,85,83,73,67
1620 DATA 79,70,198,80,65,85,83,197
1630 DATA 67,79,78,84,73,78,85,197
1640 DATA 86,79,76,85,77,197,0,245
1650 DATA 221,229,205,77,160,221,225,241
1660 DATA 254,2,216,254,5,208,254,2
1670 DATA 40,65,254,3,40,54,62,3
1680 DATA 50,28,160,221,110,0,221,102
1690 DATA 1,34,26,160,221,35,221,35
1700 DATA 221,110,0,221,102,1,34,24
1710 DATA 160,221,35,221,35,221,110,0
1720 DATA 221,102,1,34,22,160,221,35
1730 DATA 221,35,221,126,0,50,29,160
1740 DATA 205,39,160,201,62,2,50,28
1750 DATA 160,24,213,62,1,50,28,160
1760 DATA 24,219,205,86,160,33,6,160
1770 DATA 205,10,189,201,175,50,8,160
1780 DATA 201,254,0,200,254,4,208,254
1790 DATA 1,40,24,254,2,40,10,221
1800 DATA 126,0,50,33,160,221,35,221
1810 DATA 35,221,126,0,50,32,160,221
1820 DATA 35,221,35,221,126,0,50,31
1830 DATA 160,201,0,0,0,0,0,0
5000 IF check() 60526  THEN PRINT"
error in data"
```

Listing 3.

```
10 REM ** music test prog **
20 MEMORY &BFFF
30 a=&9000                          ' start of data
40 FOR octave=0 TO 5
50 FOR note=0 TO 11
60 POKE a,1                         ' set duration
70 POKE a+1,16*octave+note          ' set octave & note number
80 a=a+2:NEXT note,octave
90 POKE a,0                         ' set end of data marker
100 |VOLUME,15
110 |MUSICON,15,&9000-2
```

Demo program.