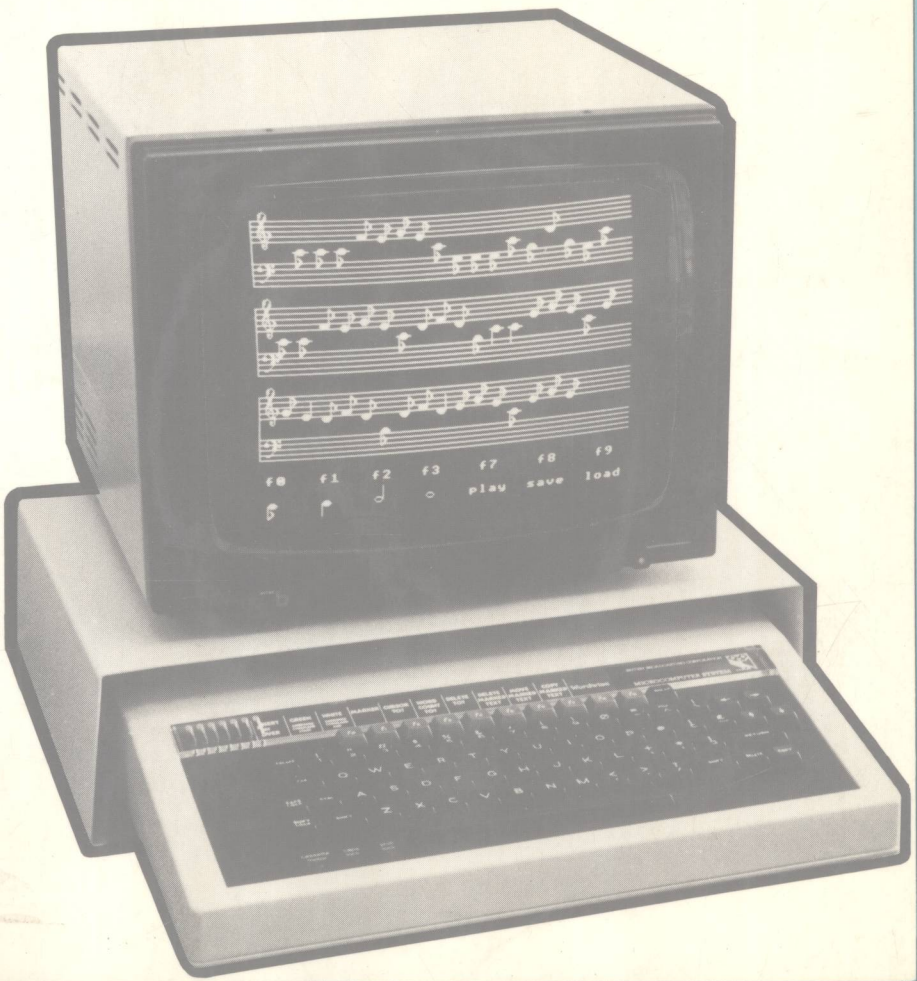
USING SOUND AND SPEECH on the BBC MICROCOMPUTER



Martin Phillips

Using Sound and Speech on the BBC Microcomputer

M. A. Phillips

MACMILLAN

Published 1984 by
Higher and Further Education Division
MACMILLAN PUBLISHERS LTD
Houndmills, Basingstoke, Hampshire RG21 2XS and London
Companies and representatives
throughout the world

Printed and bound in Great Britain at
The Camelot Press Ltd, Southampton

British Library Cataloguing in Publication Data

Phillips, Martin
Using sound and speech on the BBC microcomputer.——(Macmillan microcomputer books)
1. Computer sound processing 2. BBC Microcomputer——Programming
I. Title
001.64'43 TK7895.S6

ISBN 0-333-37155-0

Using Sound and Speech on the BBC Microcomputer

Macmillan Microcomputer Books
General Editor: Ian Birnbaum (Adviser for Microelectronics in Education, Humberside LEA)

Advanced Graphics with the Acorn Electron
Ian O. Angell and Brian J. Jones
Advanced Graphics with the BBC Microcomputer
Ian O. Angell and Brian J. Jones
Assembly Language Programming for the Acorn Electron
Ian Birnbaum
Assembly Language Programming for the BBC Microcomputer, second edition
Ian Birnbaum
Using Your Home Computer
Garth W. P. Davies
Beginning BASIC with the ZX Spectrum
Judith Miller
Using Sound and Speech on the BBC Microcomputer
M. A. Phillips

Also from Macmillan

Advanced Graphics with the Sinclair ZX Spectrum
Ian O. Angell and Brian J. Jones
Advanced Programming for the 16K ZX81 Mike Costello
Beginning BASIC Peter Gosling
Continuing BASIC Peter Gosling
Practical BASIC Programming Peter Gosling
Program Your Microcomputer in BASIC Peter Gosling
Codes for Computers and Microprocessors P. Gosling and Q. Laarhoven
Microprocessors and Microcomputers – their use and programming
Eric Huggins
The Sinclair ZX81 – Programming for Real Applications
Randle Hurley
More Real Applications for the ZX81 and ZX Spectrum Randle Hurley
Programming in Z80 Assembly Language Roger Hutty
Digital Techniques Noel Morris
Microprocessor and Microcomputer Technology Noel Morris
The Alien, Numbereater, and Other Programs for Personal Computers – with notes on how they were written John Race
Understanding Microprocessors B. S. Walker
Assembly Language Assembled – for the Sinclair ZX81 Anthony Woods

Preface

Most programs can be enhanced by the effective use of sound. If the programmer wishes to develop a variety of sounds then a range of skills is needed. First is a knowledge of the mechanism by which sounds are produced; second a knowledge of music; and third an ability to relate the computer sound and envelope parameters made by the computer. This book combines the basic elements of these three skills to give the reader sufficient background knowledge to make effective use of the BBC SOUND and ENVELOPE commands.

Throughout the book it is anticipated that readers will try out the various programs, experiment with them and program their own interesting sounds. It is not expected that the reader is an expert programmer and the programs have been written in such a way as to make typing them in as easy and error-free as possible.

I would like to thank Anne Kingston for her invaluable help with the chapter on music, Michael Free for the photography work and Acorn Computers for the permission to publish the circuit diagrams.

Martin Phillips

Contents

1. Introduction

Considering all the books that have been written about programming the BBC microcomputer, it might seem strange that the subject of this book is just two of the statements in BBC BASIC. However, these two statements are without doubt the most complex of any of the BBC BASIC statements. The SOUND statement needs to be followed by up to seven different numbers and the ENVELOPE statement needs to be followed by a staggering fourteen numbers. The sound and envelope commands are so complicated that many people do not know where or how to start, and as a result avoid using sound at all.

The aim of this book is to examine the features of the sound system, to show just what it is capable of, and of course, to show its limitations. It cannot be a definitive work on the subject and does not attempt to be, but aims to provide the reader with a basis on which to build his or her own ideas. It is hoped that readers will adapt and develop some of the programs given in the book to suit their individual requirements.

Finding out about sound is an interactive process. It is only possible fully to appreciate the sound system by a practical exploration using the computer, and the book has been written with the intention that the reader will try out the many examples. Several programs are specifically included to make this exploration easier.

By using a narrow field of study, the sound commands can also provide an exciting examination of the inner workings of the computer, and to this end an appendix on the workings of a sound buffer is included at the end of the book. Several other references to the way the computer works are made during the course of the book where appropriate.

To appreciate fully how the sound and envelope statements work, a knowledge of the physics of sound is required. Sound has always been the poor relation in physics. There are few books on the topic, most of them written some time ago. General books on physics usually spend only a short time on sound and do not provide enough detail to understand the topic clearly. This book looks first at the physics of sound as it is related to the BBC computer. To help with understanding and to try to make the subject a little less dry, the computer has been used as a learning aid, and two programs have been included to illustrate graphically points that are made in the text.

A chapter on music is also included. Again it is not intended to be a comprehensive summary of the subject; it is merely intended to give the non-musician a starting point from which to work and should enable a simple tune to be translated into the appropriate computer program.

The book does assume some knowledge of programming the BBC computer, but having said that, it does not assume that the reader is an expert programmer. Many of the longer programs have explanations at the end to show how they work, and it is hoped that the reader will benefit from these.

The programs in this book have been structured to some extent. Some programs do not lend themselves well to structuring, and indeed what is an obvious structure to one person may be unclear to another. Where possible programs have used procedures and the procedures have been given meaningful names. These names have been standardised where possible. For instance, PROCinitialise is the general 'housekeeping' procedure of several programs and is used to initialise any variables, dimension any arrays, define envelopes, etc. In the listings the procedures have been separated by blank lines; this makes for greater readability. While not essential, they do help to show the structure of the program, and the procedures can be found more easily when scanning through the program listing on the screen.

Although it is quite easy to give procedures meaningful names, it is harder in every case to do the same for variables. It is the writer's opinion that long variable names can sometimes make the program less intelligible, and so variables use appropriate names if they can be kept short and meaningful, but many general variables use the good old favourites such as N or T. This also makes for quicker typing in. For reasons that will be obvious to anyone who has entered listings before, the variables I, l and O have not been used.

All variable names that are longer than one letter are written in lower case lettering. This avoids the computer mistaking them for a BASIC keyword. For instance 'print' can be used as a BASIC variable whereas "PRINT" cannot. In this example the keyword is obvious, but the variable "TONE" would be mistaken as the keyword "TO" followed by the variable "NE"! If "tone" is written in lower case, no confusion can arise as the computer does not recognise keywords written in lower case.

All the programs in this book will work with level 1 BASIC and level 2 BASIC. The operating system is not so easy. Acorn modified this quite extensively from the 0.1 operating system (usually written as 0.1 OS) to the 1.2 OS. One of the less well publicised changes that the writer has discovered during the course of preparing the book is that the old OS made several mistakes in converting the pitch number included in a sound statement into information for the sound

synthesiser to change into a sound and as a result several anomalies could be found. These have been cured in the 1.2 version. Also the 1.2 version will support up to 16 envelope statements. As a result none of the sound programs can be expected to work quite as designed if using a 0.1 OS. The differences should be small but may be noticeable under certain circumstances. Many of the *FX calls mentioned in the book will not work with the 0.1 OS.

One very novel feature of the BBC microcomputer is the speech system. Talking computers have been the dream of many computer scientists, and now they are a reality. Speech systems have so far earned a poor reputation, chiefly because of the American accent. Here is one of the first systems to be tailored with a BBC English accent. As an add-on it is a very expensive feature. How versatile is it? Is it just a gimmick with no practical use? The chapter on the speech system, as well as showing how it is programmed, examines the basic speech system in detail, and because it is an extra, goes into details of its possible uses and limitations so that the reader will be better able to judge its value before purchase.

Hints on Entering Programs

A great deal can be learned about programming just by typing in listings. There are many short listings and several longer ones too, all of which will be found useful for better understanding of the sound system. There is nothing worse than typing a program into the computer only to find that it will not work, so at this stage some tips on typing in the programs are appropriate.

1. If a line number is typed in on its own with nothing to follow it, the computer will ignore and reject the line. To make it appear in listings it is necessary to follow the line number by a single space.
2. Some of the programs have an "ON ERROR GOTO..." line. It is better not to put this line in while typing in the program until the program works correctly. If a mistake in typing is made and this line is present, the program will not work correctly when run, but will usually go back to the start of the program again without listing the fault. It then often appears that the program has crashed completely.
3. Many of the DATA lines in programs have been split up into short sets of data. This has been done to avoid confusion when entering data.
4. Instead of having large gaps between quotes on print statements for spaces, the command SPC( ) has been used. In many of the programs screen layout is critical, and mistakes here cause endless problems.
5. It is useful to program the BREAK key when typing in listings so that the program cannot be destroyed

that way. The following line will program the BREAK key to return to mode 7 and list the program with page mode on.

```
*KEY10 OLD¦M ¦N LIST¦M
```

It can be added as line 1 of the program or typed in as a direct command.

6. Many of the programs mix upper and lower case lettering. To avoid taking the CAPS LOCK off each time, on machines fitted with 1.2 OS, if the CAPS LOCK is put on while the SHIFT key is pressed, then the action of the shift key is reversed, and lower case lettering can be obtained when SHIFT is pressed.
7. Do take care to insert all the punctuation marks correctly. Some very odd results can happen in the VDU statements if commas are typed instead of semi-colons for instance. Error messages are not generated here and the faults are obscure and hard to find. Forewarned is forearmed...!
8. Following the IF statements, the word THEN has been included for greater clarity. The use of the word THEN is optional on the BBC machine and it can be omitted provided that a space is inserted in its place.
9. Again for greater clarity, spaces have been inserted in some program lines. Unless you are confident about where spaces can be inserted and left out, it is advised that they are included or not as in the listings.
10. All the programs that start with a REM statement are available on a cassette which can be purchased separately.

The Sound System Demonstrated

The range of sounds that can be programmed into the computer is quite amazing. This book does not, and could not, attempt to be a comprehensive work in this respect, it attempts to outline the underlying theory and inspire the reader to try to create his or her own sounds.

Before the reader gets too engrossed in the theory of sound, two programs are presented here as practical demonstrations of the prowess of the BBC computer´s sound system.

The first program (Listing 1.1) uses several random terms to give an ever-changing sound effect. It is one that will easily be recognisable even if the reader does not appreciate the procedure names.

```
10 REM Listing 1.1 - Sound effect
20 PROCinitialise
30 REPEAT
40 PROCbreaker
50 PROCseagulls
```

```
 60 UNTILO
 70
 80 DEFPROCinitialise
 90 ENVELOPEl,5,0,0,0,5,2,3,10,-3,0,-1,126,70
100 vol=96
110 ENDPROC
120
130 DEFPROCbreakers
140 SOUND0,1,RND(2)+3,20
150 SOUND&1000,0,0,RND(20)+40
160 ENDPROC
170
180 DEFPROCseagulls
190 N=RND(3)
200 IF N=1 AND vol<118 THEN vol=vol+RND(8)
210 IF N=2 AND vol>80 THEN vol=vol-RND(8)
220 ENVELOPE2,3,0,-2,-1,2,2,30,50,-1,-12,-2,vol,vol
230 flock= RND(3)
240 FORgulls =1 TOflock
250 SOUNDl,2,220,20
260 IFflock>1 THEN SOUND2,2,200,25
270 IFflock>2 THEN SOUND3,2,240,10
280 NEXT gulls
290 ENDPROC
```

Description of program

20	PROCinitialise. This procedure sets up the envelope, and defines the variable vol. (Short for volume)
30-60	Set up an endless loop. This type of endless loop is used commonly throughout the book.
40	PROCbreaker. Procedure to produce one type of sound.
50	PROCseagulls. Procedure to produce another type of sound that will sound at the same time as the first.
90	The dreaded envelope statement! All is explained in chapter 4.
100	Set vol to 96 initially. It must be defined before the main program runs.
140-150	The sound commands which give the first effect.
190-210	N can take the value of 1 to 3. This random variation is used to increase, or decrease the value of vol.
220	This envelope statement has its volume changed every time the procedure is entered (by the variable vol) and therefore could not be defined in PROCinitialise.
230-280	Sound 1, 2 or 3 gulls depending on the value of flock.

The next program (Listing 1.2) turns the computer into a synthesiser. It is interesting to observe that most of the program is taken up with programming the screen display and with testing the keyboard for responses together with the subsequent decoding, and

that the actual parts of the program that make the sounds are quite short. Although complex to understand, once a knowledge of the sound and envelope statements has been acquired, the use of these two statements is quite elegant. The range of the synthesiser covers four octaves. These can be changed by using the function keys. As there are physically only enough keys on the computer keyboard for two octaves, the synthesiser is set to use the middle two octaves. By using the function keys, the octave range can be changed so that the keyboard plays the lower octaves or the higher octaves instead. There are four voices programmed into the synthesiser. They are similar to the musical instruments that bear their name; however the names chosen are rather subjective.

```
  10 REM Listing 1.2 - Organ
  20 MODE 1
  30 PROCinitialise
  40 PROCscreen
  50 PROCprint
  60 PROCrun
  70
  80 DEFPROCinitialise
  90 ENVELOPE1,2,0,0,0,0,0,0,127,-4,-4,-10,126,50
 100 ENVELOPE2,5,0,0,0,0,0,0,60,0,0,-10,126,126
 110 ENVELOPE3,3,1,-1,1,1,2,1,75,0,0,-75,126,126
 120 ENVELOPE4,1,0,0,0,0,0,0,127,-1,-1,-1,126,0
 130 VDU23;8202;0;0;0;
 140 DIM L%(32),M%(32),C%(32)
 150 octave=48
 160 env=2
 170 FOR key=0 TO 32
 180 READL%(key)
 190 M%(key)=0
 200 NEXT key
 210 DATA 2,17,66,34,82,51,52,68
 220 DATA 36,84,69,85,70,38,71,55
 230 DATA 87,88,72,73,57,89,41,74
 240 DATA 33,114,115,116,21,117,118,23,119
 250 ENDPROC
 260
 270 DEFPROCscreen
 280 VDU5
 290 VDU19,0,4,0,0,0
 300 VDU19,2,0,0,0,0
 310 VDU 24,0;256;1279;896;
 320 GCOL0,131:CLG
 330 FOR N=1 TO 5
 340 READstart
 350 VDU 24,start;512;start+64;896;
 360 GCOL0,130:CLG
 370 VDU 24,start+672;512;start+736;896;
 380 GCOL0,130:CLG
 390 NEXT N
```

```
400 DATA 32,128,320,416,512
410 VDU26
420 GCOL0,2
430 FOR N=0 TO 12
440 MOVE(64+N*96),256
450 DRAW(64+N*96),896
460 NEXT N
470 GCOL0,3
480 MOVE 48,936
490 PRINT"Q  W     R  T  Y     I  O     @  [  _"
500 MOVE 0,240
510 PRINT"C  A  S  D  F  G  H  J  K  L  ;  :  ]  R";
520 PRINT"T"SPC(38)"E";
530 PRINT"R"SPC(38)"T";
540 PRINT"L"SPC(38)"N";
550 COLOUR 2
560 MOVE 462,992
570 PRINT"SYNTHESISER"
580 VDU4
590 ENDPROC
600
610 DEFPROCrun
620 REPEAT
630 FOR key=0 TO 32
640 IFINKEY(-L%(key))=(M%(key)=0) THEN PROCsound
650 NEXT key
660 UNTILO
670
680 DEFPROCsound
690 IFkey>26 THEN env=key-26:PROCprint:ENDPROC
700 IFkey>23 THEN octave=48*(key-24):
    PROCprint:ENDPROC
710 IF M%(key)<>0 THEN J%=M%(key):M%(key)=0:
    C%(J%)=0:SOUND&1010+J%,0,0,0:ENDPROC
720 J%=0
730 REPEAT
740 J%=J%+1
750 UNTIL J%=4 OR C%(J%)=0
760 IFJ%>3 ENDPROC
770 C%(J%)=-1
780 M%(key)=J%
790 SOUNDJ%,env,octave+4*key,-1
800 ENDPROC
810
820 DEFPROCprint
830 RESTORE 930
840 FOR N=0 TO 6
850 READ key$,desc$,xpos,ypos
860 COLOUR1
870 PRINTTAB(xpos,ypos)key$;
880 COLOUR 2
890 IF N=env+2 THEN COLOUR 3
900 IF N<3 AND octave DIV48 =N THEN COLOUR 3
910 PRINTTAB(xpos+3,ypos)desc$;
920 NEXT N
930 DATA f0,Low octaves,4,28
940 DATA f1,Middle octaves,20,28
950 DATA f2,High octaves,4,29
```

```
 960 DATA f3,Piano,4,30
 970 DATA f4,Organ,20,30
 980 DATA f5,Vibraphone,4,31
 990 DATA f6,Chime bars,20,31
1000 ENDPROC
```

Description of program

30	PROCinitialise sets up all the envelopes and then initialises any data required by the program.
40	PROCscreen draws the screen keyboard layout.
50	PROCprint lists the function key functions and highlights the current keys active.
60	PROCrun is the main loop of the program.
90-120	Define the four envelopes.
130	Switch off the cursor.
140	Dimension arrays L%, M% and C%. L% stores the -INKEY values for the keys needed, M% is a temporary array to check whether a key has been held down, and C% is the sound channel number array.
150	octave is the variable that determines which octave the keyboard plays in.
160-240	Enter the - INKEY values into array L% and put each element of array M% to 0. See the User Guide page 275 for a list of the - INKEYs.
280	Join the text and graphics cursors.
290	Change the background colour to blue.
300	Change the yellow to black.
310-320	Fill a rectangle in white. Uses a graphics window and clears the screen to white.
330-400	Draw in all the black notes on the white background. Uses the same technique for colouring rectangles. A data statement is used to give the horizontal start point of each of the five black keys in an octave.
420-460	Draw black lines to divide the white keys.
470-540	Label each key. Note that the spaces in these lines are critical for correct layout, as are the semi-colons in lines 510-540.
570	Print title.
620-650	The main loop of the program. It scans through each key in turn and calls up PROCsound if a new keypress is detected.
690	Filter out the four function keys that change the envelope.
700	Filter out the three function keys that change the octave range.
710	This line stops the note playing if the key is no longer being pressed.
720-750	J% selects the appropriate sound channel.
790	Sound the note defined.
820-990	A routine to print on the screen the programmed function keys, colouring the key in red and the description of its function in black. The keys actually in use are

highlighted by having their function printed in white. This routine is accessed every time a function key is detected.

As can be heard when the program is run, the sound system is very versatile. There are however limitations in the system which one must consider and accept. The BBC machine is after all a general-purpose microcomputer and not a dedicated sound synthesiser. Once the envelope and sound commands have been understood, this program can easily be adapted and extended to have a wider range of voices. Indeed it is hoped all the way through this book that the reader will test and experiment with the various facets of the sound system.