16 Projects

I Use your BBC micro to draw a digital clock. Produce the special large characters for the digits and use a colon to separate them. Your clock can be made to keep correct time by using the internal clock of the Model B (see the user guide).

II Make a program that tests the Morse Code proficiency of the user. The data for the program will be a paragraph of text, which the Model B should translate into Morse and then print out by means of dots and dashes using the teletext block graphic characters. It can also use SOUND to simulate the sound of Morse. Your program should have a variable rate of production of the code so that the speed of the test increases as the user gets more proficient.

III Draw a set of International Road Signs. Your program should draw the background of the figures, such as red triangles, and then use your own special routines or the programs of chapters 5 and 6 to finish off the foreground.

IV Construct crossword puzzles on your television set. Each square of the puzzle should be 2 by 2 character blocks. The four blocks will either be black (in which case nothing goes in the square), or white with the bottom left-hand comer holding the letter of a solution, and either the top two characters holding the clue numbers (if any) or the top left character holding two ‘thin' digits. This allows space for a 16 by 16 puzzle.

You also have to place the clues on the screen, and these can be printed on request on the remaining area of the screen. Solutions to the puzzle can be added by a ‘cursor’ method or by having a special input code, such as letter 'A’ (across) or ‘D’ (down) followed by the number of the clue, followed by your solution.

V Write programs or use the character generator and the diagram routines (chapters 5 and 6) to produce the flags of all nations. You can draw company logos, or even design new ones. Use the techniques in chapter 14 for accessing screen memory locations to add an extra option to the

diagram routines. lItis option should allow you to make a copy of a set of character blocks (already on the screen and specified by ‘cursor’) to another set of blocks of the same size elsewhere on the screen (also specified by ‘cursor'). You could even rotate or reflect them!

VI The user manual shows you how to use SOUND to create music(?). While SOUND is making the noises you can draw the musical notation on the screen. You can construct the staves and then use special characters to place quavers, minims etc. on the screen. The old Music Hall method ofthe ‘bouncing ball’ could be used to beat the time of the tune.

VII Use the character block method to draw mazes. Naturally your program must generate mazes with real solutions. You can give yourself time limits for getting through the maze. You can make them dynamic by changing them as the game progresses. Add extra problems: man-eating monsters that roam the maze; holes that suddenly appear and can swallow you up; ‘space warps’ that can transfer you anywhere in the maze if you do not move fast enough.

VIII Extend the ideas of chapter 6. Draw your own special histograms, pie-charts and graphs. Make them dynamic (either by the ‘movie’ method of chapter 13, or by drawing a new graph on the old axes). Generate whole sets of special characters. Draw solid data graphs by joining each point on the graph to the *x*-axis. Extend the mathematical surface program (listing 10.6) to draw three-dimensional histograms. For every rectangle on the *x*/*z* grid there must now be one *y*-value. So for each such grid rectangle, starting at the back and working forward, you simply need to draw the rectangular block from the grid base up to this *y*-value.

IX Create patterns. Use EOR with large numbers of random lines or triangles about the screen. Or draw lines in a dense but regular way to get Moire patterns. For example, draw lines that join the points (0, 1) to (1279, 1023 – I) for $0 \le I \le 1023$, and points (1, 0) to (1279 – I, 1023) for $0 \le I \le 1279$ in varying step sizes. Extend the ideas of the pattern program of chapter 5 to produce complex symmetrical patterns – any introductory book on crystallography (such as Phillips, 1956) will give you lots of ideas. Draw two spirals in mode 0, one slightly off centre, and see what patterns emerge.

X Crystallography books (such as Phillips, 1956) will give you many ideas for three-dimensional objects. Extend into four dimensions – each vertex is simply a vector of four numbers and so requires 5×5 matrices for transformations. The simplest orthographic projection of a four-

dimensional point is where we ignore two of the coordinates (as opposed to one, *z*, in three dimensions). There are many more complex projections. What are translation, scale and rotation in four dimensions?

XI We have already presented one board game – Chess. There are many more possibilities: draughts (or checkers), Scrabble, Hangman, ludo, Master Mind. You can create a compendium of games. The BBC micro can simply act as the board, or it can also act as a referee. If you feel really adventurous it can even act as an opponent.

XII Use special characters to construct a deck of playing cards. These can be incorporated into a program to play blackjack (or pontoon) with the Model B acting as the bank and opponent.

XIII You can draw certain types of brain-teasers on your television screen. For example, suppose you have nine squares and each is divided into quarters down the diagonals . Each quarter has a colour (red '1', green '2', yellow '3' and blue '4') and a sign (+ or –). Denote each square as a sequence of four numbers that represent the areas taken clockwise around the centre. For example we could have (–1, –2, 1, 4), (–1, 3, 4, –2), (1, –4, –2, 3), (1, 2, –3, –4), (1, 3, –2, –4), (1, 4, –3, –2), (2, –3, –4, 3) and two occurrences of (–1, –4, 3, 2). The problem is to place the nine squares in a four by four arrangement, so that if two quarters on neighbouring squares touch, then they must be of the same colour but of opposite sign. You can use the BBC microcomputer to draw the squares initially on the left side of the screen and a three by three grid of the same sizes on the right. Then you take squares from the left and place them in the grid, or replace them back to the left. Write a program to find the two independent solutions of the above problem.

XIV Use the low-resolution graphics package of chapter 13 to manipulate the teletext screen so that it produces an approximation to a photograph. Take any photograph of yourself and superimpose a grid of 80 by 75 on it that corresponds to the graphic blocks of 2 by 3 small squares that will be placed on the screen in a matrix of 40 by 25. For each small square, decide whether it is mainly light or dark, and colour the corresponding 1/6th block accordingly. This seems like a lot of work. But note that most of the picture will be a light background, so if we use white background and blue foreground then most of the squares need not be considered. You could draw two heads side by side on the screen.

XV Write a Pac-Man type ofvideo game in BASIC. This involves drawing five moving objects on the screen at a time. In order to make the game move faster allow only two of the ghosts to move with each move of

when they are in hunting mode, and the best escape route in running mode. Because of the complex layout of the screen you will have to con promise. Simply move towards (or away from) the player if there is no wall in the way. Find a quick way of coding their movements as this will be the most time-consuming parts of the game. Speed is the essence of a good video game. Perhaps a simple machine-code routine could be used to print all five figures on the screen after altering their PRINT TAB positions.

XVI Write a program that first prints a graphics menu of special symbols on some part of the screen in mode 4. For example, use the stylised components for electronic circuits (resistors, capacitors etc.). These symbols should consist of combinations of lines. Keep a copy of the area under the menu in the same way that the whole screen is buffered in the teletext editor (listing 13.6). Use a cursor to point at any menu-symbol and then (using EOR (GCOL 3, COL)) drag a copy of it to a required position on the screen. Also add a facility for drawing connecting lines and labelling them with 'thin' numeric and special characters (such as W for ohms). You should also allow the deletion of symbols that are inadvertently placed in the wrong position. Extra options could include saving and loading.

XVII In all our perspective diagrams it is assumed that the objects lie totally in front of the eye. Change our programs so that they deal with the general case where vertices may lie anywhere in space, including behind the eye. See Newman and Sproull (1979) and Foley and van Dam (1982) for details about this concept of three-dimensional clipping.

Change the three-dimensional procedures so that they form an interactive program. Set up a complex scene (such as a group of houses) and use the menu technique to specify and change the observation point. You can change the observation point or a point on the straight-ahead ray by moving it a specified distance in the *x*-direction, *y*-direction or *z*-direction – see listing 9.12.

XVIII Produce stereoscopic hidden line views of three-dimensional objects. Now the facets must be coloured in background white and the edges in red or cyan. Remember that when you are drawing the facets for the second eye that they must not obliterate the lines drawn for the first eye.

XIX Draw a Rubik's cube or Rubik's Revenge. Enter the rotation details from the keyboard and redraw the cube in each new position.

References and Further Reading

References

Birnbaum, I. (1982). *Assembly Language Programming for the BBC Microcomputer*. Macmillan, London

Cohn, P. M. (1961). *Solid Geometry*. Routledge and Kegan Paul, London

Coxeter, H. S. M. (1974). *Regular Polytopes*. Dover Publications, New York

Davenport, H. (1952). *The Higher Arithmetic*. Hutchinson, London

Finkbeiner, D. T. (1978). *Introduction to Matrices and Linear Transformations*, 3rd edition. W. H. Freeman, San Francisco

Foley, J. D. and van Dam, A. (1982). *Fundamentals of Interactive Computer Graphics*. Addison-Wesley, Reading, Massachusetts

Horowitz, E. and Sahni, S. (1976). *Fundamentals of Data Structures*. Pitman, London

Knuth, D. (1973). *The Art of Computer Programming*. *Volume 1: Fundamental Algorithms*, 2nd edition, 1973. *Volume 2: Semi-numerical Algorithms*, 2nd edition, 1981. *Volume 3: Sorting and Searching*, 1972. Addison-Wesley, London

Liffick, B. W. (1979). *The BYTE book of Pascal*. Byte Publications, New Hampshire

Mandelbrot, B. B. (1977). *Fractals*. Freeman, San Francisco

McCrae, W. H. (1953). *Analytical Geometry of Three Dimensions*. Oliver and Boyd, London

Newman, W. M. and Sproull, R. F. (1979). *Principles of Interactive Computer Graphics*. McGraw-Hill, London

Phillips, F. C. (1956). *An Introduction to Crystallography*, 2nd edition. Longrnans, London

Stroud, K. A. (1982). *Engineering Mathematics*, 2nd edition. Macmillan, London

Tolansky, S. (1964). *Optical Illusions*. Pergamon, New York

Zaks, R. (1978). *Programming the 6502*. Sybex, Berkeley, California

Further Reading

Read any periodical, magazine orjournal that is relevant to computer graphics such as *SIGGRAPH*, *CADCAM*, *CAD journal* (and there are many, many more), and the more advanced graphics textbooks (such as Newman and Sproull, 1979: Foley and van Dam, 1982), as well as the general computer newspapers and

monthly magazines (such as *Byte*, *Personal Computer World*, *Practical Computing*, *Interface*, *BEEBUG* etc.). It does not matter if you do not immediately understand the more advanced articles: it is important to appreciate the flavour, excitement and achievement ofthe subject. Obtain the promotional and advertising literature of the major graphics companies (Tektronix, Imlac, A.E.D., Sigma, Hewlett-Packard, D.E.C. etc.), and get as much information as possible about graphics software packages. Keep an eye on the television usage ofcomputer graphics, whether it be in science programs, science-fiction melodramas or advertisements. Study video games and try to understand from the display how the characters are drawn and manipulated.

Appendix: Contents and Availability of the Two Related Software Cassettes

There are two companion audio-cassette tapes for this book. TAPE 1 contains complete listings of the larger programs relating to high-resolution graphics (that is, two-dimensional and three-dimensional graphics), while the listings on TAPE 2 relate to low-resolution graphics (that is, characters, data diagrams, video games, the disassembler etc.). Each tape contains two CATALOGUE programs (one at the start of each side of the tape) that give concise details of each listing on the tape and explains the PAGE setting for LOADing, MODE of execution etc. A brief description of the two tapes is given below.

TAPE 1

File Name	*Contents*
PTOPT	Program to join points on a regular polygon (listing 2.8 etc.)
ROSE	Program to draw rose pattern (listing 2.9 etc.)
ILLUSION	Draws rotating spirals by varying logical/actual colours and the illusion of the expanding square (listing 2.1 1 etc.)
ENVELOPE	Draws a simple envelope (listing 2.12 etc.)
SPIROGRAPH	Emulation of Spirograph (listing 2.13 etc.)
SQINSQS	Draws 21 squares, one inside another (listing 3.1 etc.)
1NTER2L2D	Intersection of two lines in two-dimensional space (listing 3.2)
FLAGS2D	Draws four flags in two-dimensional space (Dstings 4.11, 4.12 etc.)
INTERLP3D	Intersection of a line and a plane in three-dimensional space (listing 7.1)
INTER2L3D	Intersection of two lines in three-dimensional space (listing 7.2)
DOTVEC	Example program of dot and vector product (listing 7.3)
INTER3P3D	Intersection of three planes in three-dimensional space (listings 7.4, 7.5)
INTER2P3D	Intersection of two planes in three-dimensional space (listings 7.4, 7.6)
ORIENT2	Grientation of a triangle in two-dimensional space (listing 7.7)
GENROT	Rotation of point about a general axis in three-dimensional space (listing 8.5 etc.)

ORTHOCUBES	Orthographic projection of two 'wire' cubes (listings 9.4, 9.5 etc.)
MOVIE	Movie program of a rotating 'wire' cube (listing 9.12 etc.)
ORIENT3	Orientation of a triangle in three-dimensional space (listing 10.1 etc.)
JET	Orthographic view of a 'wire' jet (listing 9.9 etc.)
REVBOD	Orthographic view of a 'wire' body of revolution (a sphere) (listing 9.10 etc.)
SPHERE	Orthographic hidden surface view of a sphere (listings 9.10, 10.5 etc.)
FLAGCUBE	Orthographic hidden surface drawing of a cube with flags on faces (listing 10.3 etc.)
MATHSURF	Orthographic hidden surface view of a mathematical surfice (listing 10.6 etc.)
PERSPCUBES	Perspective view of two 'wire' cubes (listings 9.6, 11.1 etc.)
CUBESTACK	Perspective hidden surface view of a stack of 27 cubes (listing 11.2 etc.)
STEREO	Stereoscopic view of two 'wire' cubes (listing 11.3 etc.)
HIDSURFACE	Hidden surfitce view of two cubes on a coloured background (listings 12.1, 12.2, 12.3 etc.)
EXSTARS	EXEC file to overwrite SETUP procedure in previous program. It draws two stars on the same background (listing 12.4)
EXBOX	EXEC file to overwrite SETUP procedure in program HIDSURFACE. It draws a hollow cube on the same background (listing 12.5)
lib1	EXEC library file of real-to-pixel procedures
lib2	EXEC library file of two-dimensional procedures
lib3	EXEC library file of three-dimensional procedures

TAPE 2

LOGACT	Program to show relationship between logical and actual colours (listing 1 .9)
TRIANGPAT	Spiral pattern formed from triangles. Example of animation: rotation caused by redefinition of logical colours (listing 1 .12)
WORMGAME	The worm game (listing 1.13)
PIC1.16	Listing 1.16
FONT	Character font calculations
COLFONT	Coloured-character fbnt calculations (listing 5.2)
CHARG0	Simple two-colour character generator (listing 5.3)
THIN	Construction of 'thin' characters (listing 5.4)
CHARG1	CHARACTER GENERATOR 1(listing 5.6)
CHESS	Chess board program (listing 5.7)

CHESSP1	characters for CHESS program
CHESSP1	More characters for CHESS program
PREPRYNT	Assembly code construction of the PRYNT routine (listing 5.8)
CHARG2	CHARACTER GENERATOR 2 (listing 5.9)
CHARG3	Program that enables keyboard to print out multi-coloured characters (listin, 5.10)
COLCHAR	Sample character set for use with CHARG3
HISTO1	First histogram (listings 6.1, 6.2)
HISTO2	Second histogram (listings 6.1, 6.3)
PIE	Pie-chart (listings 6.1, 6.3)
GRAPHS	Scientific graphs gistings 6.1 , 6.4)
LABELS	Labelling program (listings 6.1 , 6.5)
TRIG7	Low-resolution trigonometric curves with TELETEXT (listing 13.3)
BRICKOUT	TELETEXT game 'Brickout' (listing 13.5)
TELETEXTED	TELETEXT editor (listing 13.6)
DCODE	Assembly code for generation of D1SPLAY (listing 13.7)
CLOVER	Animation of a 'clover leaf' using, rapid transfer of screen data (listing 13.8)
DISASSEM	The disassembler (listing 14.1)
SANDR	Search and replace program listing 14.2)
BASCROLL	Screen scrolling using BASIC (listing 14.3)
MCSCROLL	Screen scrolling using machine code (listing 14.4)
APPLES	Apples and bricks game (listing 14.5)
SPLAT	Simple cascade program (listing 15.3)

The following files are needed for the the-scale video gaine RON:

XYDATAP	should be RUN to produce a file like XYDATA below
START	Sets up soUND envelopes, resets PAGE boundaries and CHAINs in BJPRSNT
BJPRSNT	Logo at the beginning of the game
XYDATA	Tables of random data needed for gaine and produced by XYDATAP above
GAME	Machine-code routine to play game produced by ACGAME below
RONCHAR	Character set for game
ACGAME	Assembly code for game

Availability

These cassettes are available through all major bookshops, but in case of difficulty order direct from

Globe Book Services
Canada Road
Byfleet
Surrey KT14 7JL

Cassette 1	**ISBN 0 333 35053 7**	£9.00 (including VAT)
Cassette 2	**ISBN 0 333 36141 5**	£9.00 (including VAT)

If bought together as a set, the cost is £16.00 (including VAT) for the two cassettes. The prices quoted apply to the United Kingdom.