Microcomputing in materials teaching and research

*by T W Clyne and P J Goodhew**

SYNOPSIS

The versatility, power and availability of microcomputers has grown to the extent where their systematic use in tertiary education is now very attractive. In the field of materials, there is scope for useful exploitation of micros in a variety of ways. Some ideas are presented here about how the integration of microcomputing into teaching and research activities can be approached. Reprints are available from the authors, who will also be prepared to discuss details concerning the programmes described and other educational software they have developed.

Introduction

In the materials field, as in many others, there is considerable current interest in the use of microcomputing as a teaching tool. The increased availability of relatively powerful microcomputers has in recent years opened up a number of possibilities. Indeed, the important advantages (when compared with traditional mainframe machines) of machine dedication and real time continuous display have actually resulted in some research computation being switched to micros. An important feature of current developments is that the distinction between research and teaching usage of microcomputers is not a sharp one and the resultant cross-fertilization is potentially very valuable.

Of course, activity in this field dates back some time, and previous articles[1, 2] have covered certain features. However, it has only recently become possible to contemplate the possibility of a wide spectrum of software being integrated into materials teaching on a nationwide or even worldwide basis. In this article, an attempt is made to summarise salient features of the current position. The main types of program are categorized and illustrated by reference to a number of specific examples. These have been drawn from current teaching/research activities at Surrey, and they all run on an Acorn BBC B machine. However, the discussions presented are in general not specific to any particular type of machine. Indeed, much of the text may be considered applicable to the use of micros in any scientific field of higher education.

No attempt is made to describe individual programming points in detail, nor to consider the merits of specific commercial hardware. Such information is, of course, abundant in current computing literature. Equally, a basic background of terminology and concept is assumed. The handbook of Varey and Graham[3] is a good example of the many texts available to provide such a grounding. A prime aim of the present article is to focus attention on the most fruitful course to be followed by those concerned with materials education in the widest sense.

Objectives

Programs designed for teaching with a microcomputer may serve a wide range of useful purposes. Although some potential benefits (such as stimulation of poorly-motivated students) are side-effects rather than objectives, it is possible to classify such software according to whether the prime aims fall into one or more of the following categories:—

(1) to give an insight into the nature of a physical phenomenon by means of a direct pictorial representation, which may be on one of a number of possible levels of scale
(2) to enable the student to explore the characteristics of a process or effect by manipulation of the conditions or parameters
(3) to facilitate visualisation of features in a spatially complex system, for example by presentation of suitable sections
(4) to encourage and demonstrate the analysis and quantification of the controlling factors in a range of situations
(5) to introduce the student to practical usage of (novel) mathematical techniques
(6) to demonstrate features of program organisation and structure.

Of course, although this list gives an indication of possible goals, a number of factors will determine the likely degree of success. It is, for example, important that the student has some measure of involvement and control. This is useful, not only because it effectively allows more information to be

*T W Clyne BA, MA (Cantab), PhD, CEng, MIM is a lecturer and P J Goodhew DSc, PhD, CEng, FIM, MInstP is a reader in the Department of Materials Science and Engineering, University of Surrey.

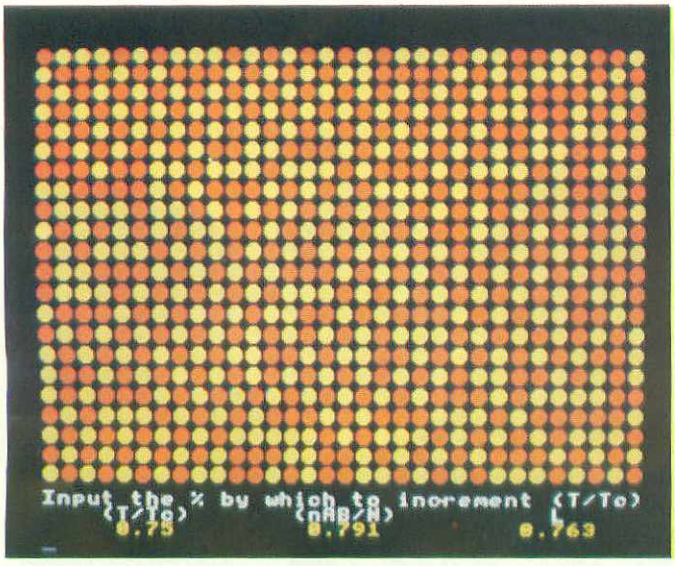

Fig 1. Representation of atomic structure in an A-50 at%B alloy, for varying values of the Bragg–Williams long range order parameter L. *Also shown is the ratio of the number of A-B bonds to the total number of bonds.*

Fig 2. Map of the contours of maximum shear stress as a result of the elastic strain field around a single edge dislocation for a selected set of variables.

available to the student, but also in view of the strong correlation between extent of interaction and level of interest. Furthermore, a program allowing student involvement leads to possibilities for setting problems, assignments and practicals which are centred around the software.

Other factors to be considered include several dependent on good programming style. This should encompass effective error trapping, for example, so that keyboard mistakes are not fatal to the program. While it is probably not realistic to expect the highest professional programming standard in specialist (tertiary) educational software, the value of such programs is undoubtedly optimized if they are well written. Not only does this eliminate frustration during use, but it increases the potential benefits available on inspection of the program listing. This latter point highlights an important facet of this type of teaching: the value is considerably enhanced if programs are easily legible. A final point about effective programming concerns optimal use of graphics and, in some cases, sound (which can, for example, be useful for event flagging). This is to a certain extent determined by features built into the micro concerned, but full exploitation of the capabilities available can markedly improve the visual impact and power of a program.

TYPES OF PROGRAM

In addition to the broad subdivisions given above, programs may be classified according to the logic of the mathematics involved. It is in some ways more appropriate to consider examples of specific programs under these headings, although it does group together some programs that appear quite diverse. The classification presented does not encompass question and answer type programs applied to specific situations. In the case of most amateur attempts, these are too clumsy and limited to merit serious attention; professional approaches to the setting up of a meaningful dialogue, on the other hand, involves entering the realm of expert systems, which can introduce a high degree of programming complexity and may require considerable memory storage.

Exploration of analytical equations

This probably consititutes the most common and obvious type of program. The significance of many such equations can, of course, be explored by a student with a hand calculator. However, the main aim of a microcomputer lies in allowing complex sets of equations to be rapidly manipulated and the physical significance of the data to be immediately presented, either in a conventional graph or in a pictorial representation or map. An example of the latter is shown in Fig 1, which is taken from a program depicting the degeneration of an ordered array of two types of atom. This software is based on the Bragg–Williams equation, which relates the degree of long-range order L to the temperature. Disordering is simulated by the exchange of suitable atom pairs selected on a random basis, in conformity with the nearest neighbour requirements. The equation concerned is a simple one, and the main value of this program lies in allowing immediate correlation with an impression of the structure.

A rather different type of presentation is involved with the display shown in Fig 2, which was generated from equations describing the shear stresses around a single edge dislocation. In this case the value of a selected parameter is being superimposed on a spatial map, allowing rapid illustration of the probable nature of strain field interactions at different points. It would also be possible to display this information in the form of atomic position maps.

An alternative type of mapping, largely pioneered by Ashby[4], involves representation of domains in which different mechanisms are operative for diverse combinations of conditions. Figure 3 shows an example taken from a program describing mechanisms of bubble growth (in irradiated material). The section shows which mechanisms are predicted to predominate in different regimes of temperature and bubble size (those appearing in this case being Ostwald ripening, vacancy collection, surface diffusion and, for the red area, a regime in which shrinkage is expected). Such maps are produced by repeatedly evaluating the reaction rates for the competing mechanisms as the two parameters, temperature and bubble size, are varied.

As a final example of a program in this category, Fig 4 shows the penetration volume and x-ray excitation volume for an electron beam entering a bulk specimen, together with schematic illustrations of escaping electrons and x-rays. As with most programs of this type, the mathematics involved is simple and the software easily written. Nevertheless, the visual impact and ease of interpretation are of some value. There are also many cases in which repeated application of analytical equations will reveal information not available from cursory calculations. For example, molecular dynamics representations, based simply on Newton's laws of motion with a suitable intermolecular attraction/repulsion expression, can reveal interesting information on vibrational characteristics. Although many such programs have previously been employed primarily in a research context, the software involved is such that simple versions might usefully be employed at a teaching level. In general, most manipulations of analytical equations present few problems of computing time or data storage when carried out on small micros.

Vector and matrix algebra

An important branch of materials analysis is concerned with the investigation of spatial relationships. An obvious application of micros lies in presentation of crystallographic relationships in the form of, for example, atomic positions in selected planes, or to illustrate stereographic projections; indeed, many programs have already been written with these objectives in mind. In mathematical terms, this type of information

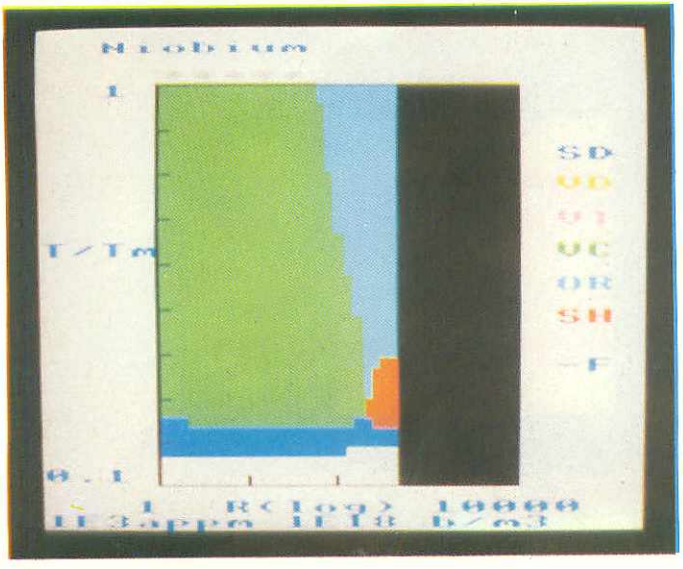

Fig 3. Mechanism map indicating the dominant growth mode for gas-filled bubbles in niobium over a range of temperature and bubble size for selected gas atom concentrations and bubble densities.

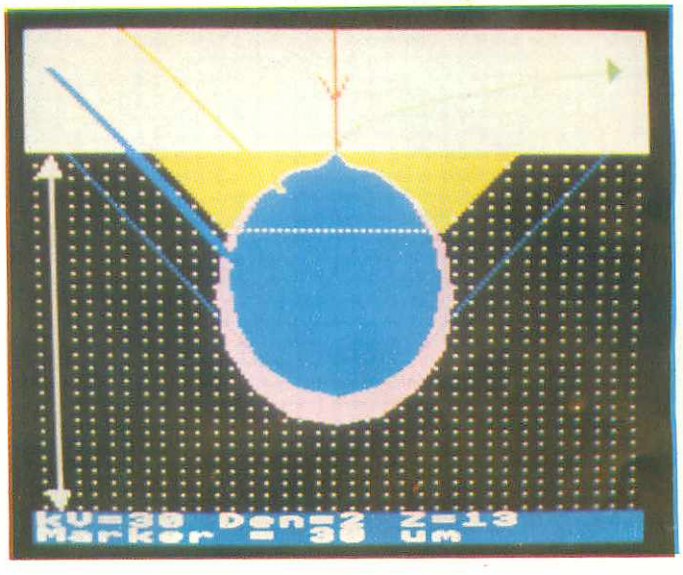

Fig 4. Sectional representation of the shape and size of the volumes with which electron penetration and from which escape of x-rays and secondary electrons will take place in a bulk specimen for selected values of accelerating voltage and specimen properties.

is frequently handled in vector form and the manipulations may involve matrix algebra. The use of a micro in such cases increases enormously the scope of what is in practice achievable; this is partly because the manipulations involved may be time consuming if carried out manually, but another significant point concerns the versatility of immediate graphical presentation. For example, Fig 5(a) shows the display during use of a program describing the Ewald sphere construction. The upper portion of this is a section through reciprocal space (for an orthorhombic structure), showing the intersection of the Ewald sphere with certain reciprocal lattice points. The lower left area then gives a schematic illustration of the expected appearance of the electron diffraction pattern, showing the system axis and undiffracted beam. This type of device has the advantage of allowing immediate correlation between theory and practical observation. Figure 5(b) shows a display from the same program after introducing a reduction in specimen thickness: the resultant increase in streaking of the reciprocal lattice points gives a greater number of intersections and this is manifested as a change in the appearance of the diffraction pattern.

Of course, usage of vectors and matrices covers a vast range of computer applications in materials-related fields. These encompass many problems in classical physics, such as solution of eigenvalues for resonant modes of vibrating systems, as well as a vast range of applications across the whole spectrum of engineering, particularly in stress analysis. The software involved is often fairly simple and computing time requirements may be quite modest; the manipulation of large matrices can introduce time and memory difficulties with a micro, but this is rarely necessary at a teaching level.

Monte Carlo methods

A powerful computing tool for simulation of physical phenomena is based around the generation of random numbers to determine event paths subject to stochastic control. Representations based on this principle are categorised as Monte Carlo techniques and their implementation is frequently dependent on estimates of probabilities for various events. This introduces an interesting element into usage of the program, as its exact behaviour will never be entirely predictable, although the emerging trends should always be reliable.

However, the grouping may actually encompass a wide range of approaches. For example, in Fig 6 an initially disordered atomic array of two types of atom has been seeded by the random selection of four centres (white dots) for subsequent growth of ordered domains (black dots). In this model the domains are made to grow by simple colour switching of appropriate atoms; by the stage shown, domain impingement has led to creation of anti-phase domain boundaries, which are readily identifiable.

In fact, classification of the above example as a Monte Carlo method is rather marginal, as the stochastic element is limited to the initial location (and phase) of the domains. A more classical illustration is provided by the program depicted in Fig 7. This display represents an electron beam incident on a thin foil. Random number generation is used to determine whether successive electrons are: (1) transmitted without interaction or experience, (2) elastic or (3) inelastic scattering, or (4), for high-angle nuclear collisions, backscattered. The numbers of each interaction event are monitored, together with the current total T and, for electrons experiencing an interaction, a further random number generation is employed to determine the angle through which they are deflected. Such events are then simulated visually at the time they occur (by red traces for elastic events and blue for inelastic ones). A typical feature evident from this type of simulation is the predominance of small angular deviations of the scattered electrons, for most conditions of practical interest.

Many Monte Carlo methods are not excessively expensive in computing time, although memory limitations may be encountered with micros if large arrays are required in order to retain, for example, a series of atomic positions. However, some problems of execution rate may arise in programs describing effects such as diffusion on an atomic scale, in terms of jump probabilities and vacancy distributions. A typical display during running of a program of this type is shown in Fig 8. This depicts a 10-deep atomic array near a diffusion couple boundary, with both vacancies and solute atoms visible through the 10-layer stack. (Individual jumping events of different types, controlled by migrational enthalpy terms, are flagged with characteristic sounds). This representation is potentially useful for illustrating net effects such as relative interpenetration rates and Matano interface movement. However, a common problem in generating detectable net effects from atomic scale phenomena lies in the very large number of events and/or atoms that must be simulated before reliable trends emerge.

A program of the above type takes some time to reveal reliable information and should really be written in a compiled language such as FORTRAN. Unfortunately, the majority of micros do not at present have facilities for running compiled languages as part of the standard equipment, although the routine incorporation of, for example, a BASIC compiler would be a useful development, which may take place in the near future. (A potential solution in the meantime is to write the critical portion of the program directly in an assembly language[5], but legibility and scope for modification by the student is then lost and software development becomes very time consuming.)

Numerical techniques

There is, of course, a field of intense research activity concerned with simulation of physical phenomena, centred around the concept of numerical solutions. Many complex pro-

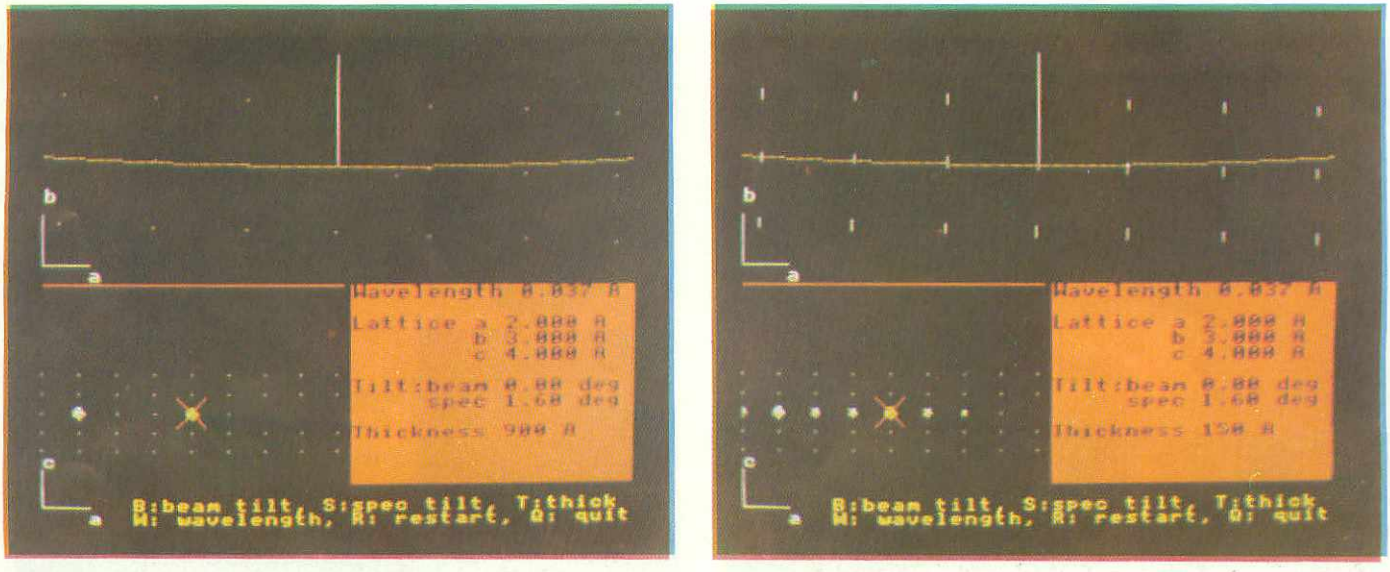

Fig 5. (a) Illustration of a section through reciprocal space, showing interception of the Ewald sphere with some reciprocal lattice points, and the corresponding appearance of the electron diffraction pattern, (b) as for (a), demonstrating the effect of a decrease in specimen thickness.

cesses and effects are described by relatively simple differential equations; for example, diffusional and convective transport phenomena, the quantum mechanical wave function, electric field distributions and many other important cases obey standard second order linear partial differential equations, under appropriate boundary conditions. Unfortunately, solutions in the form of analytical expressions do not exist for most cases of practical interest and recourse to numerical methods is often necessary. These techniques consititute prime examples of the power and versatility of computing methods, but they have not traditionally been taught in any depth at undergraduate level in standard science courses. (This situation may change, as the use of computing in general, and numerical modelling in particular, continues to expand in industrial and scientific circles.)

A very simple example of an iterative method of solution is provided by the program depicted in Fig 9. This illustrates the growth of a crystal from a nucleation point (and at a specified initial undercooling) on the surface of a fine droplet under Newtonian heat flow conditions. Advance of the growth front releases latent heat over a changing interfacial area, tending to raise the temperature of the system, reduce the interfacial undercooling (ie lead to recalescence) and thus slow down the growth rate. It is necessary to select a suitable time increment in order to consider the competing effects of surface heat extraction and interfacial heat release. This is the only discretisation involved in the program, which may thus be followed with virtually no background understanding of numerical methods.

However, many applications demand spatial discretisation, of the type involved in finite difference and finite element methods. These methods frequently invoke substantial computing time requirements, and have thus been considered unsuitable for micros. However, this situation is changing as hardware development continues and programs which are, at the least, usefully illustrative can now be run successfully on many micros. An example is provided by the program which produced the display shown in Fig 10(a). This is based on a finite difference version of the heat conduction equation, applied to predict the effect of a moving heat source on the temperature distribution in a thin sheet. This type of program has value both in terms of a practical introduction to numerical techniques and as a means of exploring features of the phenomenon and its analysis. The material and dimensions of the sheet (and of the superimposed grid) may be selected beforehand, while the source velocity and intensity can be changed during the run. For example, a second set of conditions led to the display shown in Fig 10(b). There is clearly scope for the linking of this type of program to simple experimental investigations.

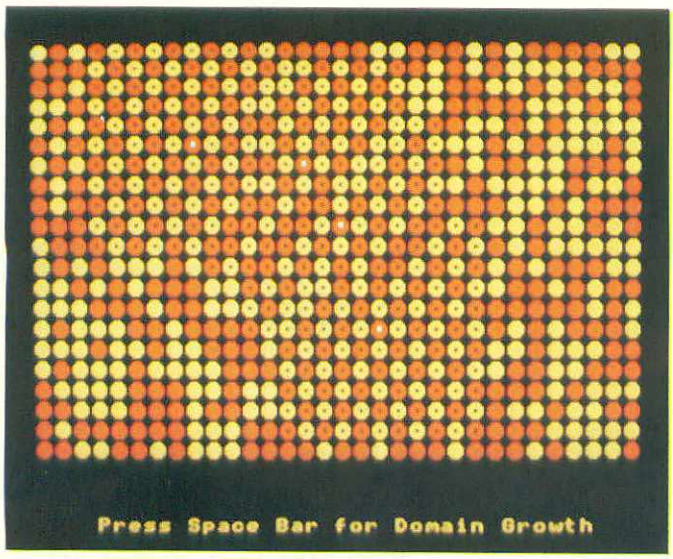

Fig 6. Schematic illustration of the growth of ordered domains in an A-50at%B alloy, from an initial state of total disorder. The atoms marked with white dots are centres for domain growth and several anti-phase domain boundaries are forming between them.

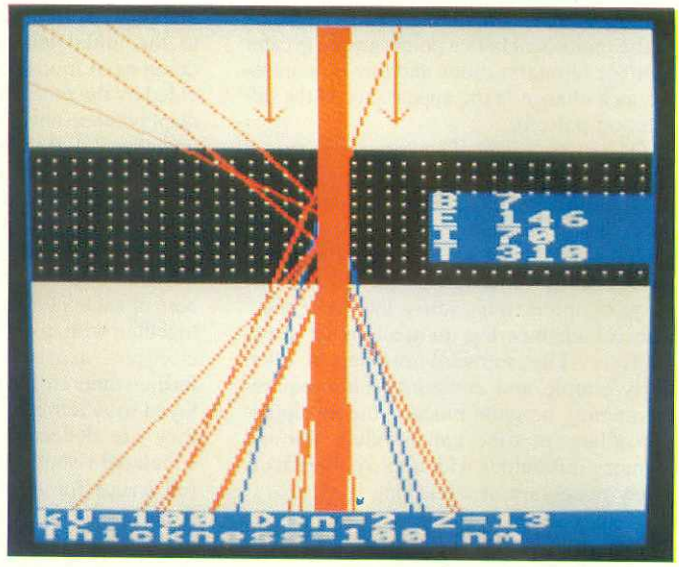

Fig 7. Schematic representation of an electron beam passing through a thin foil. The tracks of electrons experiencing scattering events, shown in red (elastic) and blue (inelastic), are controlled by random number generation.

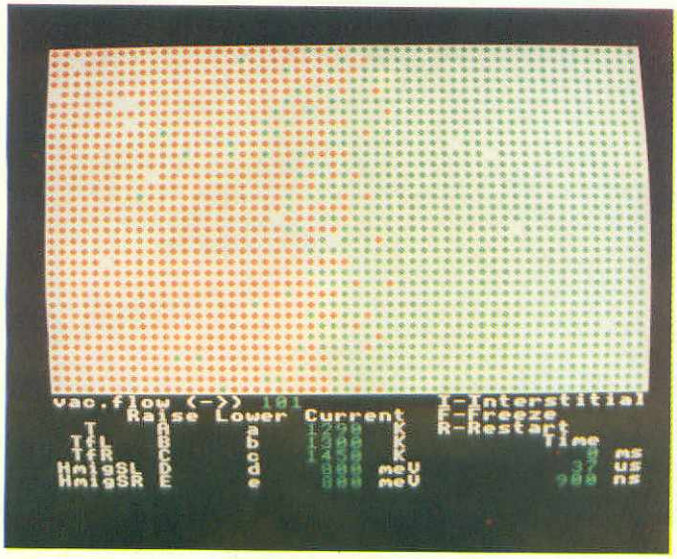

Fig 8. Monte Carlo simulation of atomic movements in a diffusion couple.

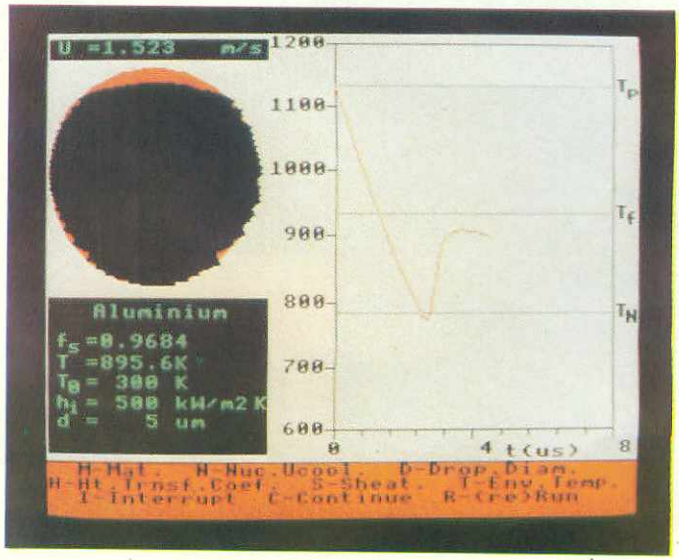

Fig 9. Prediction of the thermal history development in a fine droplet during crystal growth from a single nucleus.

Software development

Ideally, educational software should be both efficient (in terms of information manipulation and display) and simple (to facilitate in-house development and encourage student involvement with the programming side). Compatibility between these requirements is greatly assisted by good program structuring. The programs illustrated here were written in BBC BASIC, which has a number of features designed to encourage structured programming. The colour graphics and sound facilities are also good, particularly when used on a model B. However, the mapping required for graphics can be expensive in memory; for example, although 32K is normally available on a BBC B, use of high resolution graphics, together with a disk filing system (and allowing for areas reserved for the operating system), means that there is actually less than 6K available to the user for the program, variable storage and computation. This is in some ways productive as it encourages efficient programming and use of several instructive devices. For example, programs can be 'chained' so that a first program might set up a graphical display and then run a sister program which actually carries out the calculations. Similarly, if a large number of variable values are being handled, space may be saved by holding them on a disk in the form of data files, to be read as necessary.

Software which facilitates student involvement is clearly to be encouraged. For example, menu-driven programs are simple to operate and amenable to easy error-trapping. Similarly, if a series of variables is to be selected, it is often preferable to display a default set which can be modified (if only because this discourages input of unrealistic values). It is also a good idea to allow program interruption by offering a 'freeze' facility, which is achievable with many micros via a facility for repeated scanning of the keyboard buffer. This enables the student to select points at which to change variables, note information or simply to think about the implications of the display. It is also a simple matter to incorporate into the program a facility for saving to disk a copy of the screen display at any time. This facilitates both student study and demonstrations by the teacher.

System development

We are generally concerned with micros falling into what might be termed the 'intermediate' category between simple machines for home use (currently costing £200 or less) and business machines for offices costing several thousands of pounds for a complete system. Including a disk drive, which is considered almost essential, the hardware involved here might retail at between £500 and £1500. Part of the attraction of these systems in a teaching/research environment lies in the potential for expansion and overlap with other activities. In addition to the obvious scope for use in exercises, lecture demonstration and practicals, the free availability of micros facilitates access to word processing, printing, data logging, image analysis, equipment control, or on-line terminal usage (to a host mainframe computer or, via a customized networking system, to other micros), to name but a few of the spin-off possibilities. In order to optimize the versatility in these respects, it is useful to aim for equipment standardisation, preferably with a system for which hardware and firmware is easily expanded and interfaced. The Acorn and Apple systems certainly constitute examples where this can be done, although there are viable alternatives on the market.

Another important aspect relating to choice and development of the system concerns the availability of commercial software and firmware. The distribution of educational software among teaching establishments may occur on a scale that was never possible in purely research activities. There is, as yet, little suitable software freely available in the area of tertiary (materials) education, but this situation is starting to change. As translation between languages (and even between different dialects of BASIC) can be rather troublesome, it is arguably advisable that an attempt be made on a national level to establish language guidelines. Although specialist educational software is still in short supply, a wide range of more general facilities are freely available as firmware (eg ROM chips), and many of these assist in improving the scope of educational programs; for example, chips are available to increase memory and/or cpu power, to give improved graphics capabilities, to facilitate production of hard copy screen images, to synthesise speech and to widen the range of languages that can be used. Of course, a problem with software written to take advantage of such facilities is that portability is thereby reduced.

Summary

There is no doubt that microcomputing usage will continue to expand, and scientific tertiary education will obviously be in the forefront of this development. The familiarity which a typical student intake has in this field is now undergoing a dramatic upswing and this both facilitates and demands greater exploitation of the potential in this area. The examples given here illustrate some of the directions that may usefully be followed in the field of materials, although the range of topics covered is far from exhaustive. The primary need now is probably for a national consensus on how developments should be rationalised and co-ordinated.

References

1. D Phelan, A computer game for teaching alloy properties, The Metallurgist and Materials Technologist, Jan 1983, 39–40.
2. J E Glen, Computerized Learning, The Metallurgist and Materials Technologist, July 1983, 349–350.
3. H Varley and I Graham, The personal computer handbook, London, 1983, Pan.
4. H J Frost and M F Ashby, Deformation mechanism maps, London, 1982, Pergamon.
5. I Birnbaum, Assembly language programming on a BBC microcomputer, London, 1982, MacMillan.

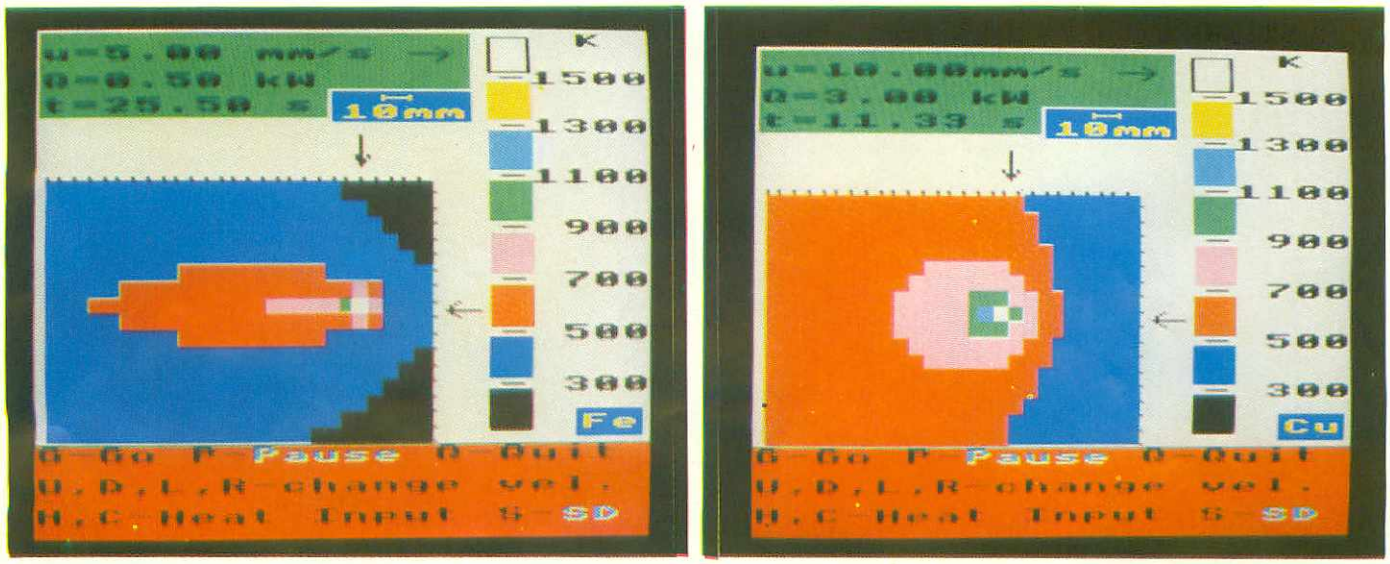

Fig 10. (a) Illustration of the changing temperature distribution in a thin sheet heated by a moving source, the tick marks indicating the finite difference mesh employed, (b) thermal field with a different material and conditions.