APPLICATIONS OF THE INTERACTIVE COMPUTER GRAPHIC LANGUAGE, ICPL

H.G. Bown, W.A. Hartman, and R.E. Warburton
Communications Research Centre,
Department of Communications,
Ottawa, Ontario

Abstract

ICPL (Interactive Control Program Language) is a language for writing programs for an interactive computer driven display. This paper briefly describes ICPL by way of an illustrative example and presents a number of application packages that have been written for a PDP-9 based interactive computer graphics system at the Communications Research Centre. These application packages contain a number of programs referred to as "control programs" written in the ICPL language. These control programs generate light-button menus, interpret light-button strikes and control the execution of specialized functions such as data extraction, numerical computation and data display. Details such as initialization, declaration of global variables, communication with the monitor, etc., are all handled by the ICPL system. The scope of these applications demonstrates the suitability of the ICPL language for implementing the desired man-machine interface in an interactive graphical environment.

Abrégé

ICPL (Interactive Control Program Language) est un langage de programmation pour commander une unité d'affichage de façon interactive. Ce document définit brièvement ICPL à l'aide d'un exemple, et il décrit par la suite divers assortiments de programmes réalisés avec ICPL sur un système graphique (soutenu par un ordinateur PDP-9) du Centre de Recherches sur les Communications. Ces programmes servent à produire et à interpréter les menus de boutons lumineux et à commander l'exécution de fonctions spécialisées telles que l'extraction de données, le calcul numérique et l'affichage. Le système ICPL remplit d'autres fonctions: par exemple, l'initialisation du programme, la déclaration des variables, et la liaison avec le moniteur. L'ensemble de ces applications démontre comment le langage ICPL est approprié à la réalisation de l'interface homme-machine dans un système graphique interactif.

INTRODUCTION

ICPL[1] (Interactive Control Program Language) is a language for writing programs for the interactive computer driven display. It provides a simple tool to express the interaction between an application program and a human operator by means of a light-pen or other pointing device, and can be extended to other interaction devices such as function keys or an alphanumeric keyboard.

There is no intrinsic difficulty in programming a computer to display a series of messages and pictures on a CRT or in transferring control to specific routines when requested to do so by an operator interaction. However, a practical difficulty in programming an interactive display is that each hardware configuration tends to have its own idiosyncrasies and conventions, and a surprising amount of detailed coding can be required for even a simple task. This tends to discourage the use of interactive graphics and decreases the clarity and maintainability of the programs.

The development of ICPL was an attempt to identify the key concepts that are shared by the members of a broad class of interactive graphics programs and to embody these in a language and programming system that would be simple to use and would yield programs that are clear and easy to maintain. To a large extent the programs are hardware-independent so that the programmer need not become conversant with the fine details of the particular hardware employed.

Figure 1 is a block diagram of a typical interactive system. The data base contains the raw material of the interaction and may be edited and displayed in a variety of forms. The block identified as "Control Program" is concerned with the generation of light-button "menus", the identification of light-pen "strikes" and other means of interaction, and with the activation of program segments in accordance with a prearranged scheme. These program segments control the execution of specialized functions such as data extraction, numerical computation and data display. Most of the idiosyncrasies of the display and identification hardware are found in the control program. This program can be very tedious to write and to modify if the logic of the interaction is written in an improper programming language. ICPL has been developed especially for writing control programs with features that are tailored to this task. It is not intended to be a general-purpose language and the remainder of a complete interactive system will be written in other languages (e.g., Fortran and assembly language).

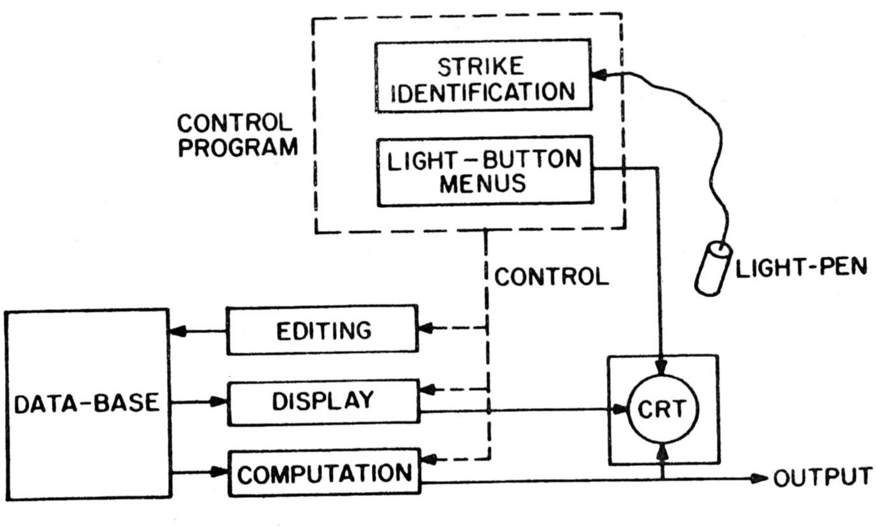

Figure 1: Structure of an Interactive System

An ICPL Program

Consider the following program written in ICPL:

```
* ANY STATEMENT WITH '*' IN COLUMN 1 IS A COMMENT.
CONTROL PROGRAM MOVE
* THE FOLLOWING THREE STATEMENTS ARE DECLARATIONS
INTEGER V1
SYMBOLS TEXT1
EXTERNAL PROGRAM PICT,MOVUP,MOVDN
* THIS BLOCK IS EXECUTED WHEN THE PROGRAM IS ENTERED
ENTRY
   LET V1 = 1
   SET 0A,12,S1
1  DISPLAY
   SEEK
* EACH 'OBJECT' STATEMENT HEADS A BLOCK OF CODE THAT GENERATES
* ONE OBJECT IN THE DISPLAY FILE.  THE FIRST OBJECT IS ONLY
* GENERATED FOR A PARTICULAR VALUE OF THE STATE VARIABLE V1,
* AND IS GENERATED BY CALLING THE EXTERNAL ROUTINE 'PICT'.
* NOTE THAT NOTHING HAPPENS WHEN THE LIGHT-PEN SEES IT.
OBJECT IF V1 EQ 2
   DO PICT
* THE NEXT OBJECT IS A LIGHT-BUTTON
OBJECT
   TEXT 1500,1500/'ON'
* THIS TIME WE GET SOME ACTION (BUT ONLY IF V1 = 1)
ACTION IF V1 EQ 1
   LET V1 = 2
   GOTO 1
OBJECT
   TEXT 1500,1400/'OFF'
ACTION IF V1 EQ 2
   LET V1 = 1
   GOTO 1
OBJECT
   TEXT 1500,1300/'UP'
ACTION IF V1 EQ 2
   DO MOVUP
   GOTO 1
* 'MOVUP' AND 'MOVDN' ARE EXTERNAL ROUTINES THAT
* PERFORM THE CHORE OF MOVING THE PICTURE
OBJECT
   TEXT 1500,1200/'DOWN'
ACTION IF V1 EQ 2
   DO MOVON
   GOTO 1
END
```

It can be seen that the language exemplifies the logical structure of the program model, as seen by the user, and provides him with a facility to express his particular requirements without unnecessary detail.

A picture of some sort is generated by the external routine PICT and can be switched on and off by pointing the light-pen at the light-buttons ON and OFF, and moved upwards or downwards on the screen by the light-buttons, UP and DOWN. This process could be represented by a finite-state machine having two states, V1 = 1 and V1 = 2, where V1 is an integer variable. When V1 is 1 the picture is switched off but when V1 is 2, it is displayed and the UP and DOWN light-buttons can be activated.

On entry to the program, the code in the ENTRY block is executed. This execution initializes the variable V1, sets three parameters of the display system (origin mode, beam intensity, symbol set), displays the light-buttons, and looks for a light-pen strike.

The five OBJECT statements, each with its associated block of code, define the contents of the display file each time a DISPLAY statement is executed. The first OBJECT statement is qualified by the condition that V1 should be 2 in order for the routine PICT to be executed. The other OBJECT statements are unqualified so that the four light-buttons are always displayed. The statements in OBJECT blocks, like those in the blocks headed by ACTION or ENTRY statements, are indented to column 4 and may be labelled.

An OBJECT statement may be followed by as many ACTION blocks as desired, each specifying a possible course of action when the display object associated with that OBJECT statement causes a light-pen strike. ACTION statements may be conditional or unconditional and are tested in the order in which they appear in the program. In the example program the actions consist of changing the state variable V1 and calling the external routines MOVUP and MOVDN to move the picture. The actions terminate with a branch to the statement labelled 1 in the ENTRY block which rewrites the display file. The following statement, SEEK, causes the system to await a new light-pen strike. If none of the action block conditions are satisfied when a light-pen strike occurs, an automatic SEEK is executed.

The macroprocessor, STAGE2[2,3] used to translate ICPL programs to assembly code necessitates a rigid format so that extra spaces should not be inserted in the statements. Statements that start with a '*' in column 1 are treated as comments and ignored during translation.

APPLICATIONS

Descriptions of a number of application packages that have been written for the PDP-9 based interactive computer graphics system at the Communications Research Centre are given below. These descriptions are not intended to give complete information about the detailed organization and operation of these packages but are rather an overview of their design and use.

CODER - Interactive Character Design

When the characters and/or symbols in an interactive graphics facility are generated by software rather than hardware techniques, there is usually a list (or "dictionary") of display commands which result in the generation of a particular symbol when processed by the display controller. If the display commands are of a certain format or if the list is of significant length, then the generation of the list can be a tedious job.

CODER is a program (written in ICPL) which is used to link together a number of FORTRAN and MACRO-9 (PDP-9 assembly language) programs so that a presently available interactive graphics facility (PDP-9 based) could be used to quickly and accurately generate these lists for a "destination" graphics system being developed. The system is designed in a modular fashion so that a change in the specification of the display commands would not necessitate a complete re-writing of the system to implement the change-over, thus giving the system portability. Approximately 25% of the 450 program statements are ICPL, 60% are FORTRAN and the remaining 15% MACRO-9.

The technique used for the specification of the desired character/ symbol shape is to use a light-pen to point to a sequence of cross-shaped marks arranged in a two-dimensional matrix of variable size on the display screen. When a mark is seen by the light-pen, it is joined to the previously seen mark with a vector. Light-buttons control the intensity of these vectors; low intensity representing a movement on the "destination" display with its beam off, and high intensity representing movement on the "destination" display with the beam on. Other light buttons control deletion and insertion capabilities for editing partially completed character shapes. The display commands necessary to generate the desired shape on the destination system screen are displayed in tabular form on the existing systems' CRT screen upon command.

The ICPL control program interprets the light pen strikes from light-buttons and the cross matrix and makes calls to an external FORTRAN subprogram to generate a tabular description of the operator's desired character/symbol shape in the form of matrix entity numbers and beam control commands. This table is processed by a FORTRAN subprogram to convert the entity numbers into direction-of-movement codes. This second table is scanned by another FORTRAN subprogram which generates the final display code by calling a MACRO-9 subprogram. This display code is then displayed on the existing CRT (in numerical fashion) by the ICPL program.

The vectors joining the crosses in the matrix give the operator a larger than full size preview of the final character/symbol shape as it is being designed, eliminating the possibility of coding errors. The time required to produce a "dictionary" has been reduced to approximately 10% (i.e., now about 2 hours are required to produce a 36 character upper case English alphanumeric set). A typical display during the interactive character design process is shown in Figure 3.

IDDS[4]: Interactive Data Display System

There are many graphical software packages available for pictorial representation of various kinds of data but most have been designed for a non-interactive environment. IDDS is a tool for interactively creating graphical presentations of data that are visually pleasing and/or facilitate further manipulation and extraction of useful information from the original data. The IDDS system is composed of a number of distinct program overlays each with a specific function and each utilizing interactive graphical techniques to a large degree. The user of the system requires little or no knowledge of the computer system itself or of programming languages.

IDDS is a system of overlays created by CHAIN consisting of a resident main program and a set of subroutines organized into four links. The resident program and the main entry program of each link are written in ICPL. Other programs are written in ICPL, FORTRAN and MACRO-9 assembly language. It is interesting to note that ICPL accounts for 50% of all code, excluding standard system routines, FORTRAN accounts for 45% and MACRO-9 the remaining 5%.

The resident main program acts as a dispatcher to all other IDDS control programs. The main program also includes storage for character-string and simple variables which are shared between ICPL programs. The file operations program allows the user to indicate on which I/O device the data is residing and provides facilities for selecting the name of the data file to be investigated. The format specification program provides the facilities needed to indicate the type of format for the graphical presentation (e.g., CARTESIAN, POLAR, PIECHART, etc.). The graph preparation program is used to prepare a graph for final output to an I/O device or to the x-y recorder. Figure 4 shows a typical graph created with the graph preparation program.

A short description of the man-machine interaction that takes place in the selection of a filename follows. It is assumed that a data file to be examined exists on one of the legal I/O devices. When the IDDS system is first started the resident main program, which acts as a dispatcher to all other IDDS control program overlays, is being executed. It displays on the CRT the names of the other program overlays to which control can be transferred. The user at this time, desiring to indicate the name and location of the data-file he wishes to examine, points the light-pen at the words "FILE OPERATIONS" on the CRT. The file operations program overlay is then loaded into core memory and program control is transferred to it. The file operations, program displays on the CRT the following light-buttons: RETURN, DECTAPE, DISC, PAPER TAPE and TELETYPE (Ref. Figure 5).

If the user points to one of the file-oriented I/O devices, dectape or disc, an alphanumeric keyboard is displayed on the CRT along with "DISPLAY DIRECTORY" and "FILENAME" light-buttons. The I/O device light-button picked is also marked by an asterisk to remind the user of the particular I/O device selected. The user can now, using the light-pen, pick characters from the keyboard and create a name on an area of the CRT just to the right of "FILENAME". By pointing to "FILENAME" the previously created filename will be deleted. Alternately, the user can point the light-pen at "DISPLAY DIRECTORY" and cause a directory listing of the selected I/O device to be produced on the CRT. Each filename displayed on the CRT, as part of this directory, is active and by pointing to it with the light-pen its name is entered just to the right of "FILENAME" as in the manual case. The user can now transfer control to the resident main program by pointing with the light-pen to "RETURN". Similar techniques are used in the other program overlays.

Many features remain to be added to the IDDS system. Interactive techniques can be provided for other facilities such as curve fitting and numerical integration. As a stand-alone system it can be used with experimental data or the output of analytical programs presented in the form of a magnetic tape or disc file. Alternatively, it can be the component of a larger package, such as a circuit analysis system.

HOUSE - Interactive Architectural Design

This program was especially devised (as a CRC open house demonstration) to show the capabilities of interactive graphics in a familiar application - the design of single storied buildings. It is representative of a whole class of computer-aided design applications in engineering and architecture.

Like the IDDS system, HOUSE is organized into several overlays according to the particular function to be carried out; there is a "building" overlay, a data file management overlay and a hard copy overlay. These overlays are composed of a mixture of FORTRAN, ICPL and MACRO-9 subprograms. The total system composition is about 65% FORTRAN (for functions such as "windowing" the display, etc.), 20% ICPL (for control of the main flow) and 15% MACRO-9 (for data base management and system dependent code).

The building overlay gives the user the necessary tools to create a two-dimensional layout (floor plan) of a house. By actuating the light-buttons presented, with a light-pen, the user has the capability of drawing and deleting horizontal or vertical walls of a standard thickness and measured length on a 6" grid. A scale graduated in 1, 5 and 10 foot increments is supplied when in "drawing" mode and on request for measuring operations. Standard size doors and windows can be inserted into or deleted from these walls. The display can be scaled up or down by factors of 2 (i.e., 2, 4, 8 or 1/2, 1/4, 1/8) and the inside or outside areas and perimeter of any enclosed space can be computed and displayed.

The process of creating a drawing results in the generation of a simple data-base consisting of the X and Y coordinates of the end points of each wall, door and window (doors and windows are treated as "special" types of walls). This data-base is then a complete description of the drawing and can be filed for future reference. This is accomplished by using the file management overlay. Here the user is given the capability to store a data-base on magnetic tape under a unique name for later use or to retrieve a data-base previously stored for reference or modifications.

The hard copy overlay allows the user to obtain an ink drawing (on 8-1/2" x 11" paper) of the data base being displayed. This is accomplished by means of an analog X-Y recorder and D/A converters.

Other useful layout problems that can be simulated using this system are; layout of furniture within a room, layout of houses in a subdivision or layout of subdivisions within a town. Due to the limited time spent on the program, many useful facilities have not been included in the present version but some logical extensions would be: variable wall, door and window size, bills of material automatically produced, cost to build estimates and diagonal line capability.

A typical screen display during "building" is seen in Figure 6.

PROFIL[5] - Interactive Design of Diffused Layers in Semiconductor Devices

In the design and manufacture of silicon devices with diffused impurity layers, it is necessary to be able to estimate, on the basis of the diffusion conditions, the impurity distributions that will result. Several diffusion and oxidation steps are required, with several impurities, and the distribution of each impurity deposited continues to be affected by every subsequent diffusion step. Because of the calculations involved in monitoring each redistribution, determining the final impurity profile and estimating the resulting device performance becomes complicated and time-consuming. It was felt that the design of such diffused layers could be simplified greatly by the application of interactive computer graphics, and so the ICPL program PROFIL was developed to aid the work of the Microelectronics project group at this establishment.

As with the other ICPL application programs, PROFIL links together a set of FORTRAN analysis programs; the method of analysis is based upon the standard predeposition-diffusion techniques of diffused layer production in silicon and follows from a consideration of basic solid-state theory as discussed by Grove[6]. These techniques consist of a predeposition of a shallow layer of a very high concentration of an impurity into the silicon substrate or epitaxial layer; this is followed by redistribution of the impurity by high-temperature drive-in treatments. Other impurities may be introduced in further predepositions, depending on the device structure desired, resulting in profiles such as that for a typical transistor in Figure 2. The PROFIL analysis routines convert the input data (time, temperature, etc.) for each step of the process, as well as the data describing the epitaxial layer (if present) into which the diffusions are made, into a set of number arrays, one for each of the impurities involved, containing the necessary information to calculate the profile for that impurity. This information is updated continually to take account of the further diffusion resulting as each processing step is encountered in turn. These individual profiles are finally combined, taking account of the conductivity type of each impurity, to provide a final net concentration distribution, which is then displayed on the screen. The analysis routines can also take account of such factors as impurity redistribution during thermal oxidation, concentration-dependent diffusion coefficients, and other anomalous effects for certain impurities. Experimental data (sheet resistivity and junction depth measured on test wafers) obtained from actual diffusion runs can be used as input to correct the profile for some of these effects.

The ICPL programs control the interactive data input as well as profile calculation and display; data input is by means of light-buttons on the CRT screen activated with a light-pen. The procedure for this can be described by discussing the actual screen displays in Figures 7 and 8. When a profile analysis is begun, display 1 (Figure 7) is seen, without the profile or axes present; the table at the bottom of the screen, showing the data for the displayed profile, is empty. Display 2 (Figure 8) is brought onto the screen by defining the number of a processing step and activating the light-button "PROCESS DESIGN". The desired values for the

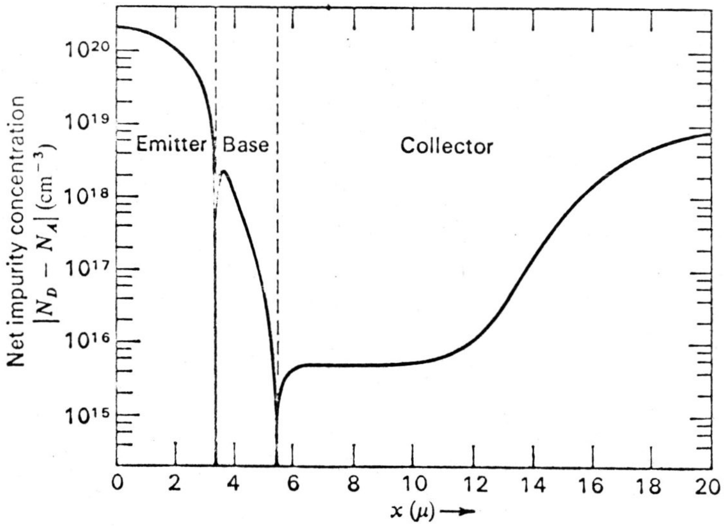

Figure 2: Impurity concentration profile for a pnp transistor (estimated from diffusion conditions).

process parameters are entered by selecting characters from the keyboard with the light-pen; when "RETURN" is activated, display 1 reappears with the data recorded for checking purposes in the correct column of the table. At any time, "PLOT" may be activated; the profile to that point in the process will then be calculated and displayed with correct axes. If information about the profile is desired, a third display can be brought onto the screen by activating "ANALYSIS". Using the tracking-cross features of this picture, the operator is able to select points on the profile and have information such as the concentration gradient and layer resistivity for a desired region calculated and displayed. A plot of the displayed profile can be obtained with an X-Y recorder, as well as a printout of the input data.

All of the features of the analysis routines could be used, in a very inefficient way, on a computer in a batch processing mode. But it is the interactive features supplied by the ICPL control program that make the PROFIL package a very useful design aid for semiconductor diffused-layer devices. There are obvious and significant advantages - borne out in practice - to using interactive graphics for such design work. The diffusion conditions can be specified and the finished profile obtained in a useful form very quickly and easily. This permits the operator to see rapidly the effects on a device of any changes in the diffusion conditions. Also, any information necessary for device performance evaluation can be obtained easily.

MAGNET - Interactive Design of Magnetic Deflection Electron Optical Systems

In the scanning electron microscope the electron optical system forms an image of the electron source on the image plane; a raster is then formed by scanning this image spot, using a magnetic deflection system, across the image plane in the x and y directions. The designer of such a deflection system must know how the size and shape of the spot will vary as the spot is scanned off the optical axis of the system and as the magnitudes of various system variables are changed. Theoretically, it should be possible to determine, from the physical properties of the deflection system, the magnetic field distribution; this distribution could then be related to the type of optical aberrations that would be produced by such a system. An aberration-free system, one for which the image spot size and shape does not vary as it is scanned, could then be designed. This is, however, not a practical method of design, because of the labor involved and the inaccuracies of the calculations. Nor is it efficient or practical to design by the "cut-and-try" method of making a large number of coils of different sizes, shapes, etc. until the deflection system is sufficiently aberration-free. Therefore, because there was a need in this establishment for a design aid for this type of work, the interactive graphics program package known as MAGNET was developed using ICPL.

The Fortran analysis routines for this application are based upon the fact that for a system with aberrations the deviation from the ideal aberration-free case can be expressed as a power series in certain system variables; the various terms and their combinations represent the standard aberrations, such as coma and astigmatism, that are present in optical systems. The coefficients of this series are known as the aberration coefficients. Using these routines, it is possible to determine the spot outline at any point in the image area to be scanned, for a chosen set of coefficients. ICPL programs control the entry of data - the magnitudes and phases of up to sixteen aberration coefficients, and the deflection coordinates of the point being considered - and display the spot outline for that point in the image plane. The operator can examine one point at a time (Figure 9), or the spot outline at as many as 25 points in the scan area can be shown simultaneously (Figure 10). As with the other application programs, plots of the displayed spot outlines can be made.

To use MAGNET in designing, first a deflection system is made, with a certain coil configuration. A photographic record is made of the actual spot size and shape at several different points in the image area; then as close a match as possible to this pattern of spots is made by using MAGNET and varying the aberration coefficients. When the patterns match, the coefficients and hence the types of aberration that are having the most effect are known. Using this knowledge, a different system configuration can be tried and the process repeated until most aberrations are eliminated.

This method is still somewhat "trial-and-error", but there are other factors that make it workable. First, for the type of electron optical systems being considered, there are relationships between the coefficients so that only about half of them need be varied independently of the others. Secondly, this interactive method is such that many combinations can be tried in a few minutes; in that time it can be seen which coefficients are having the most effect in working toward the experimental pattern. By concentrating on those coefficients, the match can then be made very quickly. Also, there are ways of determining at least some of the necessary changes in system configuration from a knowledge of the types of aberration present. Therefore, mostly because of the speed of the interactive graphics phase of the operation, this process has worked most satisfactorily.

CONCLUSION

The users of an interactive graphical system are usually in a position to criticize the designer of the system because in an interactive environment the organization of the system is clearly visible. Experience with the above application packages has shown that the designer of an interactive system is periodically under pressure to provide revisions and additions to accommodate user requirements. It is therefore very important that an interactive system employ a high level language, such as ICPL, for specifying the graphical interaction in order to simplify the writing of the initial system and also to facilitate future additions and revisions.

This paper has briefly discussed the ICPL language and has described a number of application packages utilizing it. These packages demonstrate the ability of ICPL for implementing the desired man-machine interface in an interactive graphics environment.

REFERENCES

1. Maclean, M.A., *Designing a Language for Interactive Control Programs*, 2nd Man-Computer Communications Seminar, 31 May - 1 June 1971, pp. 30-39.

2. Waite, W.M., *Building a Mobile Programming System*, Computer Journal 13 no. 1, February 1970, pp. 28-31.

3. Waite, W.M., *The Mobile Programming System: STAGE2*, Comm. ACM 13 no. 7, July 1970, pp. 415-421.

4. Bown, H.G., *Interactive Data Display System*, DECUS Spring Symposium, 10-13 May 1972, pp. 97-104.

5. Hartman, W.A., *Analysis of Impurity Diffusion Profiles in Silicon by Interactive Graphic Display*, International Electrical, Electronics Conference; 4-6 October 1971, Toronto (published in conference digest pp. 24-25).

6. Grove, A.S., *Physics and Technology of Semiconductor Devices*, New York: Wiley, 1967, Chap. 3.

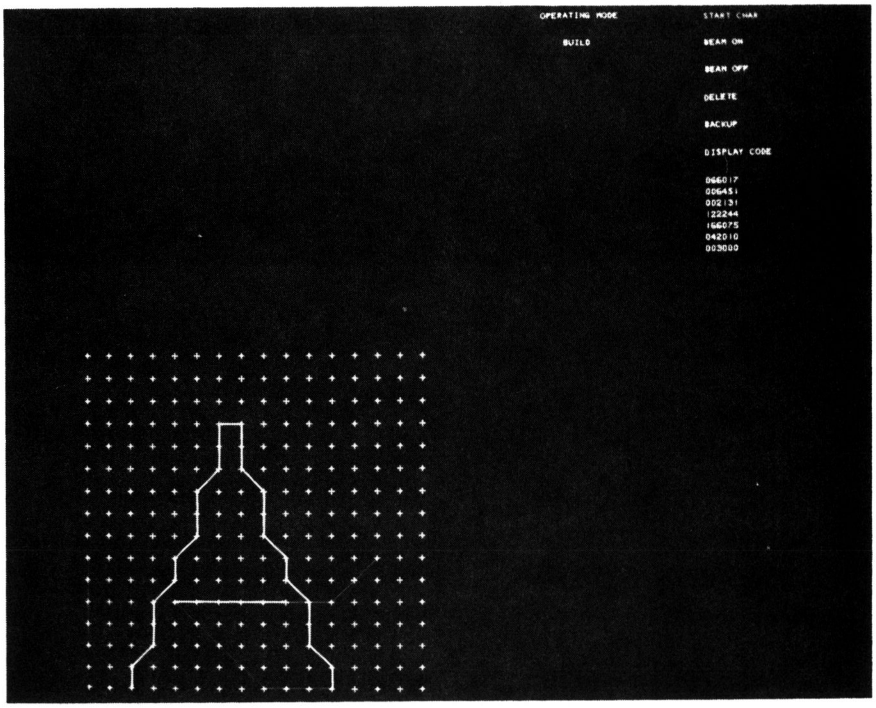

Figure 3. CODER: Typical screen display during the design of a character (i.e. letter "A").

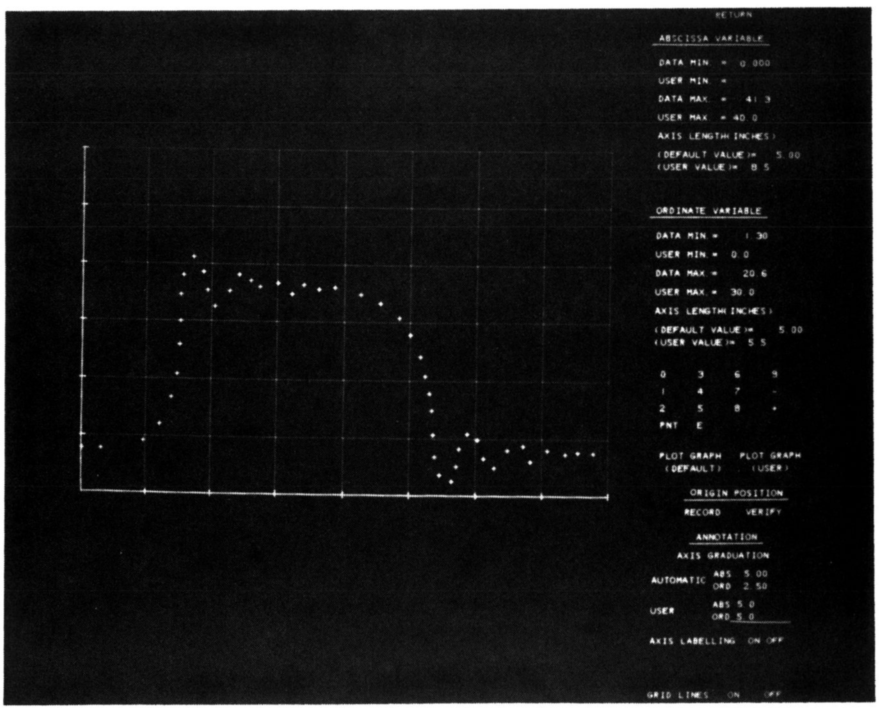

Figure 4. IDDS: Display showing typical graph as prepared with the graph preparation overlay.

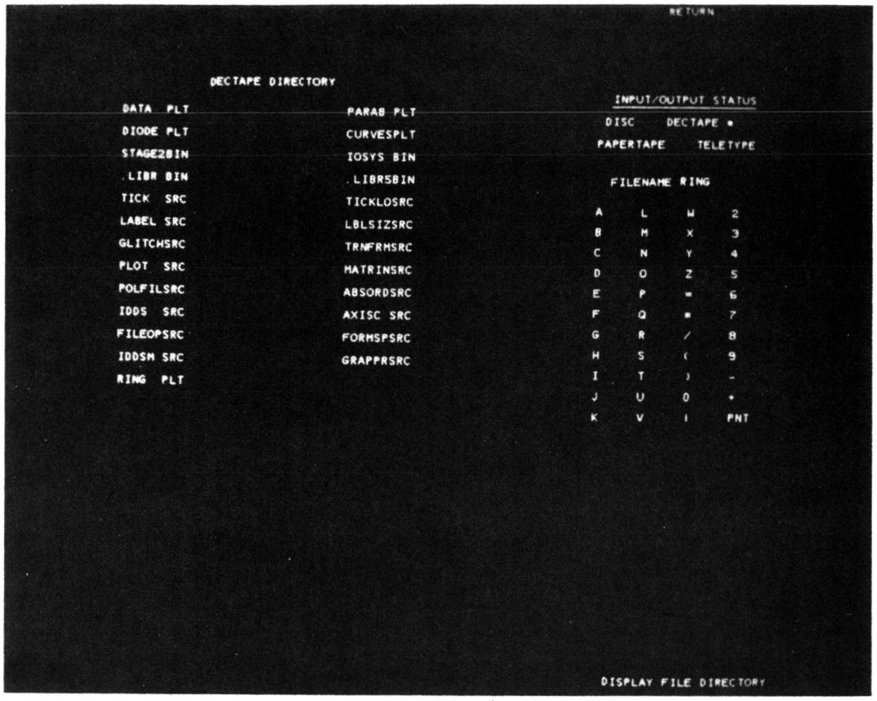

Figure 5. IDDS: Screen display during use of the file operations overlay.

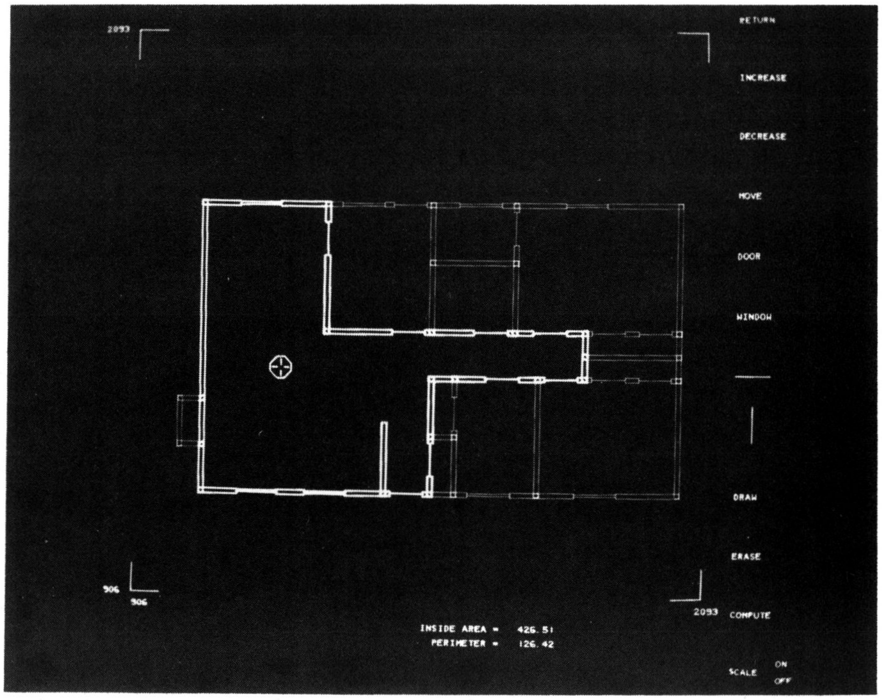

Figure 6. HOUSE: Typical screen display during the building process.

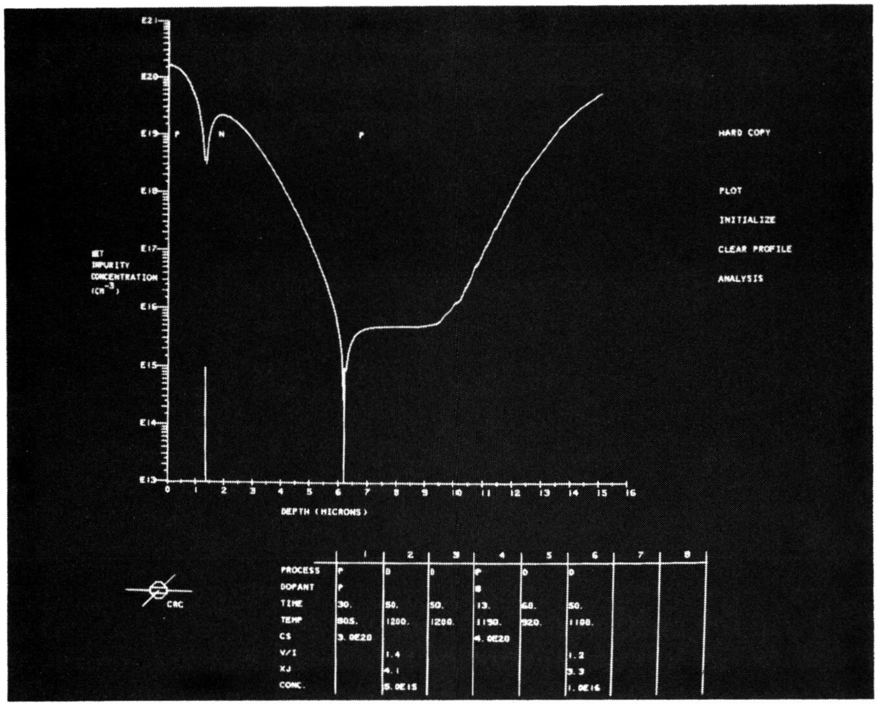

Figure 7. PROFIL: The control program light-buttons and computed impurity distribution for the diffusion conditions tabulated at the bottom of the screen.

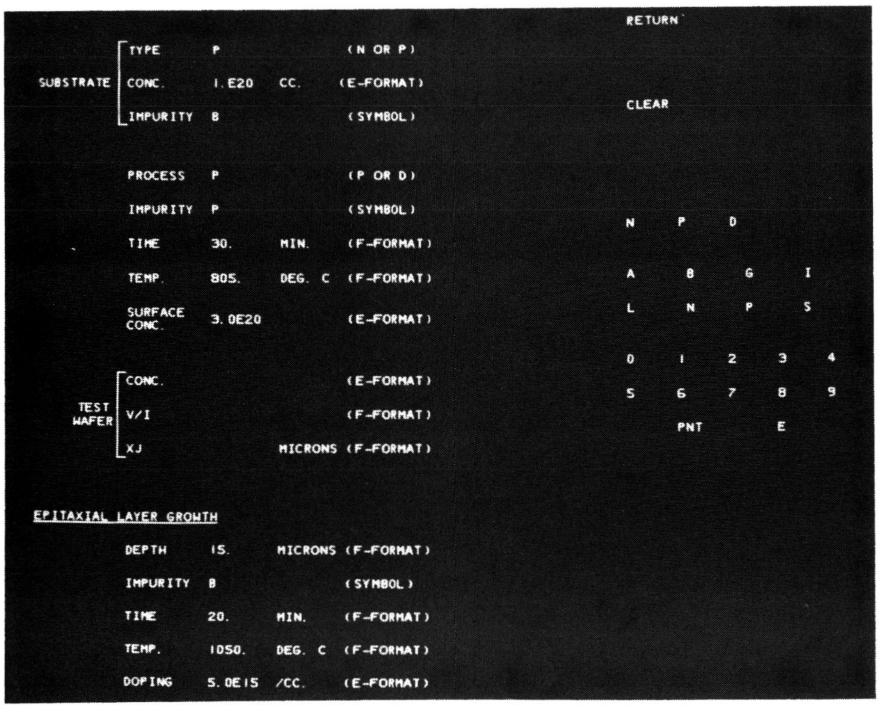

Figure 8. PROFIL: This display shows the entry of data for the first step (a predeposition of phosphorus) in the diffusion process.

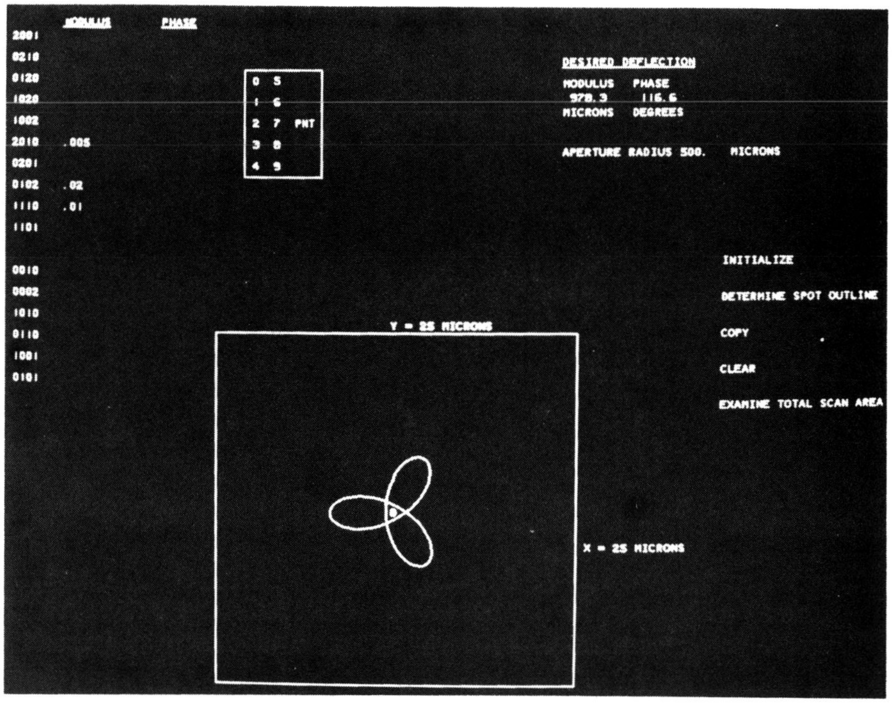

Figure 9. MAGNET: This is a display of the spot outline produced for the aberration coefficients, aperture radius and deflection co-ords selected.

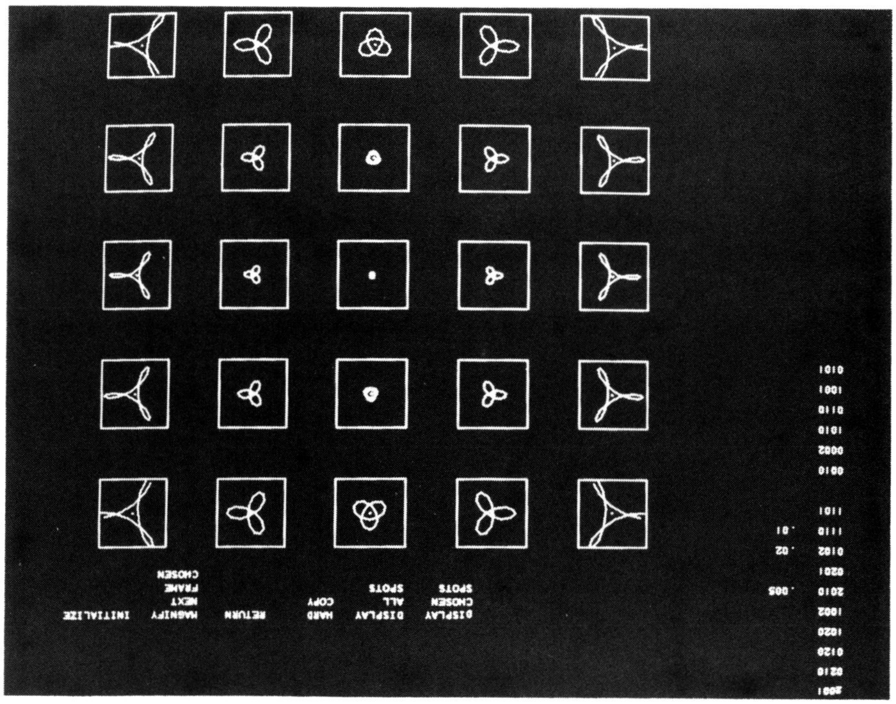

Figure 10. MAGNET: This display shows the spot outlines produced across the entire scan area (2x2mm.) under the same conditions as for fig.9.