BIG SYSTEM - INTERACTIVE GRAPHICS FOR THE ENGINEER

Charles E. Quenneville
Boeing Computer Services, Inc.

SUMMARY

The BCS Interactive Graphics System (BIG System) approach to graphics is presented, along with several significant engineering applications. The BIG System precompiler, the graphics support library, and the function requirements of graphics applications are discussed. The paper concludes that graphics standardization and device independent code can be developed to assure maximum graphic terminal transferability.

INTRODUCTION

A great many users of interactive graphics have common graphic display requirements. Some of these requirements are the abilities to display and modify input and computed program variables, to interactively control the logical program execution, to display two- and three-dimensional data in a variety of graphical forms, and to obtain offline hardcopy output. There are available in the commercial world a wide variety of graphic display terminals, each with its own unique software to support utilization of its unique terminal features. This software, though becoming increasingly sophisticated, generally provides only the basic capabilities for displaying random characters and vectors. This requires that the application programmer become familiar with the terminal software in order to develop software on a more convenient, though more costly, level. Further, once the application has been developed, it becomes "locked" into a particular terminal and its portability to other graphic terminals is seriously limited. It is becoming increasingly important that we identify functional requirements between graphic programs and graphic terminals, and provide standard graphics software which can automatically perform these functions.

The BIG (BCS Interactive Graphics) System (ref. 1) is such an approach to standardization in the field of interactive graphics software. BIGS at Boeing dates back to 1967 when graphics first became available to the engineering community as an application tool. After a short period of application development, it was readily apparent that the development of graphics within an application was by no means a trivial task. We identified four major functional requirements that were common to most of our graphic users: (1) logical control over program execution; (2) high level graphic display

capabilities; (3) ability to display and modify the value of program variables and arrays; and (4) offline hardcopy output. The ability to draw arbitrary lines and characters did not meet all the requirements of our graphics applications. In the fall of 1967 Boeing began the development of a standardized graphics system to meet these functional requirements through a project known as CADLIB (Computer Aid to Design Library). A precompiler was also introduced to help eliminate the tedious task of generating the data statements required when creating displays with textual information. Over the years the name CADLIB was changed to BIG System and the system was improved and expanded until it is today accepted and actively used by Boeing, BCS and commercial clients.

The BIG System is presently operational on the IBM 360/370 with the IBM 2250 graphic display terminal, the CDC 6600 with the CDC 241/243 graphics terminal, the Digital Equipment Corp. PDP 11/45 with either the Vector General 3-D hardware four color system or the Tektronix 4010 series display terminals, and the Prime 300 with Tektronix display terminals. The BIG System is also fully operational on BCS's CDC and IBM timesharing networks (MAINSTREAM-EKS and MAINSTREAM-CTS) using Tektronix 4010 series terminals.

THE BIG SYSTEM PRECOMPILER

The BIG System precompiler translates macro-level, graphics oriented statements into FORTRAN code which can then be processed by the FORTRAN compiler and linked to the graphics library to form executable code (see Figure 1). The precompiler is operational on both the IBM 360/370 and the CDC 6600 series computers, and it provides complete transferability of the graphics portion of programs written on those machines. The precompiler provides the facility by which an unsophisticated computer user can easily add graphics to his programs. It is not necessary that he know all the details of the graphics system, simply that a given macro-statement will produce a given type of display. It is an important distinction that a system implementor is more interested in <u>how</u> something is done, while the user is interested in <u>what</u> it does.

The precompiler language consists of macro-level display statements which provide the capability for controlling the logical execution of the program and displaying and/or modifying program variables.

One such powerful macro-level display statement for controlling the execution flow of the program is the "DISPLAY MENU" statement. A "menu" can be generated which consists of a number of lines of text, each one of which represents an option in the program. By selecting a line in the menu, control is transferred to a corresponding point in the program. Other devices for directing program control will be discussed later in this paper.

The use of the precompiler for display and program control are best illustrated with an example:

The equation for a projectile fired from a gun at an initial angle of inclination (A) with a fixed muzzle velocity (V) over an interval (T) under a constant acceleration (g) is given in the following equation:

$$X = VT \cdot \cos(A)$$

$$Y = VT \cdot \sin(A) - .5gT^2$$

The program listed below allows for the display and modification of the values of V, A, and plotting step size; the plotting of the values of X vs Y for a range of T's; the display of the X and Y arrays so they may be modified; and the program control over data display and program execution.

```
      DIMENSION   X(50), Y(50)
      DATA  NPTS/50/
      DATA    V,A,TINC,G/600.,30.,.3,32.2/
      START  DISPLAY
1     DISPLAY MENU $PROJECTILE$
     1   :DISPLAY CONTROL PARAMETERS:  (10):SOLVE:(20)
     2   :DISPLAY TABLE:  (30)
      GO TO 1
10    DISPLAY DATA *VELOCITY*V,*ANGLE IN DEGREES*A,TINC
      GO TO 1
20    T = 0
      DO 21 I=1,NPTS
      X(I)=V*T*COS(A/57.3)
      Y(I)=V*T*SIN(A/57.3)-.5*G*T**2
21    T = T + TINC
      DISPLAY PLOT LINXLINY $PROJECTILE PLOT$ X,Y,NPTS
      GO TO 1
30    DISPLAY TABLE $PROJECTILE TABLE$ X,Y,NPTS
      GO TO 1
      END
```

The resulting displays are shown in Figures 2 through 5. Interactive commands are automatically provided for control, variable update and offline plotting. The "START DISPLAY" command performs all the initialization required by the graphics terminal. Statement Number 1 and the corresponding Figure 2 illustrate the "DISPLAY MENU" statement. The simple format provides for titles enclosed in dollar signs ($), and lines of text enclosed in colons (:) and statement numbers enclosed in parenthesis [()]. When a select is made on a particular line, program control is automatically transferred to the associated statement number.

Statement Number 10 in the example and the corresponding Figure 3 illustrate the simplicity with which program variables can be displayed. The program variable names are listed separated by commas (,) following DISPLAY DATA. The display is automatically generated with the variable name followed by its value. The variables can be modified at the terminal. When the user exits from the display the variables are modified within the program. Variable names may be replaced by up to twenty characters of text enclosed in asterisks (*) preceeding the variable name.

The "DISPLAY PLOT" statement automatically scales the data and generates the plot shown in Figure 4. The grid is drawn and "good" grid values are automatically generated for maximum resolution. This reduces the time to program the operation and reduces the probability of errors.

The DISPLAY TABLE statement (see Figure 5) illustrates another type of data display statement. A table of X and Y values is displayed; although only the first twenty points are displayed, the user may reposition this twenty point window to view any of the remaining points. He may also plot and/or modify the data points. This macro-statement in effect invokes a mini-editor that allows the user to manipulate and replot the data. The data is modified only in core. It is the user's responsibility to update any disk or multiple files.

THE GRAPHICS DISPLAY LIBRARY

Each of the hardware, software or user directed operations are implemented by one or more calls to library subprograms. These subprograms provide capabilities for accurate, fast and easy generation of the display portion of the application program. In order to provide for the device independence described above, the system consists of two types of subprograms: those that communicate with the terminal (called the low-level routines) and those that manipulate the user's data (called high-level routines). A simple, flexible interface between the two levels allows the implementor to exchange low-level routines when he changes terminal systems. The low-level routines are provided by the terminal vender (ref. 2) and simply draw lines, erase lines, and change the terminal state.

The higher level routines are the routines called by either the user or the precompiler in order to express the sequence of operations required to generate an image display or a user interaction. These allow the user to select and define defaults for a graphics terminal (initialization), and to define the logical program control and the display format. The program control can be provided through menus, function keys, input options and other input devices such as keyboards, light pens, data tablets and crosshairs. On terminals not equipped with certain devices, the control functions can be simulated. For example, a function key interrupt on key number 13 can be simulated by inputting "KEY, 13" on the keyboard of a terminal not equipped with a function keyboard.

Graphic displays are generated by defining a coordinate system (rectangular or polar), scales (linear, logarithmic, etc.), labeling and text generation. The scales are determined either automatically in order to fit all the data or a section of the data (blowup), or explicitly by the user. Where data falls outside the screen scale, a scissoring routine is used to exclude all elements outside the region. Where the data are three-dimensional, routines are available to rotate the objects. If the terminal is equipped with 3-D hardware, the display is generated directly with rotation provided automatically through dials, a joy stick or other input devices. On terminals not equipped with three-dimensional hardware, an orthographic projection is implicitly performed to project the data into a two-dimensional coordinate system. Textual data display and editing routines are also provided for displaying and modifying program variables, data arrays, tables and matrices.

Since all terminals do not have a device to generate a permanent copy of the displays generated, a technique to preserve a representation of the object on the screen, for later hardcopy plotting, has been developed. Interactive plot commands are available to the user to selectively create plots on the Stromberg-Carlson (SC4020), the CalComp, Gerber, or other remote plot devices, by means of a post-interactive processor.

BIGS USERS

Within the Boeing Company the BIG System has been used in many application areas such as: Finite Element Structural Analysis, Antenna and Radar Design, Control Systems, Weapon Systems, Flight Simulation, Rapid Transit, Project Management, Master Dimensions, and Data Analysis and Reduction. Two of these significant applications are the STRIP and GGP programs.

STRIP

STRIP (Structural Interactive Plotters) is a graphically oriented interactive preprocessor for NASTRAN (NASA Structural Analysis) used for verifying the basic geometry of a NASTRAN defined structural model (grid locations and element connections). A similar postprocessor for NASTRAN has also been developed for displaying the NASTRAN output (refs. 3 and 4).

The data used to define a NASTRAN model and the results generated by a NASTRAN analysis are inherently graphical. This is one of the reasons the plotting capability provided within NASTRAN has been so widely used. The major problem with that plotting package is one of computer operations. The time required to get a NASTRAN job scheduled, run on the computer and the plots returned is often several days. This is too long a time for an analyst to effectively check his input. As a result many expensive NASTRAN runs are made that solve the wrong model.

STRIP was developed using the BIG System and operates on BCS's CDC and IBM based timesharing network using comparatively low cost Tektronix terminals (see Figure 6). The program was written using both the precompiler and the BIG System library directly. The graphic system provides for all terminal initializations, graphical displays, program control and offline hardcopy output. Program control is provided through menus, simulated functions keys and crosshairs. Offline plotting (Gerber, SC4020 and/or CalComp) is provided as an automatic option.

The program logic for the NASTRAN preprocessor operates in the following way. The program reads the NASTRAN deck. The grid cards are converted to the base XYZ system and stored in data arrays. The element connectivity cards are read and an element connectivity table is built. At the time the connectivity cards are read a check is made to see that each grid reference on the connectivity card has been defined by a grid card. An error message is printed out that identifies any missing grid points. After the NASTRAN data has been read the remainder of the program is executed interactively from the terminal. This provides the user with the opportunity to select those operations that contribute most to the data verification process. Since many types of errors might occur and only the user can detect most of them this type of operation is both cost-effective (fewer spoiled runs) and productive (the user can choose to observe only the useful data displays). The user then chooses the next step from the following menu:

DEFINE DISPLAY PARAMETERS

DEFINE DISPLAY SET

DISPLAY STRUCTURE

DEFINE DISPLAY PARAMETER allows the user to choose the view angle. By defining different view angles the structure may be rotated in any direction. Detail which is hidden because of a given orientation can be interactively reoriented to show clearly the structural idealization. Often several viewing angles are necessary in order to examine all of the geometry and connectivity of the model.

DEFINE DISPLAY SET allows the user to select any part of the model to be displayed. The linear elements, triangular elements and/or quadilateral elements within one or more ranges of element ID's may be interactively selected as the display set. The linear elements, triangular elements and/or quadilateral elements may be selected to be displayed individually or in combination. In addition specific element ID's or a range of ID's may be selected. The set selection option allows the model to be displayed section by section. Duplicate lines can be displayed in separate views.

DISPLAY STRUCTURE causes the program to create the previously defined display with user chosen viewing angles. The user has the option to display gridpoint ID's and/or element ID's. Another important capability of this option is the BLOWUP feature. The diagonal corners of a new display window are defined as two points on the display. The window defined by these points are the new scale limits. The structure is displayed so that the window is expanded to the boundaries of the display screen. This is an important capability in viewing complex structural details.

The preprocessor described above has been extensively used. Figure 7 shows the NASTRAN demonstration problem for a 3-dimensional Delta Wing. The model includes several linear, triangular and quadilateral elements. Figure 8 shows a NASTRAN spherical cap problem. The spherical cap model has been rotated counterclockwise so that the gridpoint ID's are more visible. Figure 9 shows a blown up portion of the same spherical cap. Notice the lines on the right and bottom are clipped at the display boundary. Planned extensions of the system include the display of single point constraints, multipoint constraints, forces and concentrated masses.

GGP

GGP is a general purpose data visualization program used to view, manipulate and analyze vast amounts of data. Versions of this program operate on the CDC 6600 using a 243 display terminal, a PDP 11/50 with a Vector General 3-D four color display terminal, and Tektronix terminals, and on BCS' CDC timesharing system with Tektronix terminals. A variety of application programs have been developed on the PDP 11/50; however this discussion concerns itself only with the GGP applications.

The PDP 11/50 is an independent minicomputer dedicated to the Boeing Commercial Aircraft Company's Propulsion Computational Technology Group. Its primary intent is to provide, through graphics and structured random access files, tools and techniques for engine risk assessment, fluid and structural analysis, and development of computer design systems for engine related hardware components.

The PDP 11/50 graphic display terminals consist of a Vector General (VG) display system, a Tektronix 4014 and a Tektronix 4010. The Vector General is a three-dimensional graphics display terminal with a four color monitor, keyboard, dials, joy stick and a data tablet. The installation allows the engineer to directly interact with either his data, data at the test facilities, or data at the central large-scale computer, the CDC 6600.

The full PDP 11/50 system functional configuration is given in Figure 10. The operating system software for the PDP is RSX 11D Version 4.

The BIG System provides all the graphic interfaces between the user programs, the vendor supplied software, the graphic terminals and graphic input devices. The precompiler described above is as yet not available on this installation. A user task written using BIGS may operate on any of the three graphic terminals without the necessity of rebuilding the program. The method of accessing the terminals through the BIG System is shown in Figure 11. When the user program begins execution the user has the option of selecting which graphic terminal is to be used. Once a terminal has been selected, a BIGS user task is automatically initiated to service the terminal and the user task. All graphics initialization, program control, graphic display, updating and hardcopy output are provided automatically. When three-dimensional data are generated on the VG, rotation and translations are automatically provided through dial inputs. When three-dimensional data are generated for the Tektronix, an orthographic projection into a two-dimensional display is provided automatically.

The GGP System is a system of intercommunication modules designed to permit rapid access to and analysis of vast data. The modules included within the GGP System and their functions are shown in Figure 12.

The data to be analyzed by GGP may be input in a variety of forms: cards, magnetic tape or from a RJE link to our CDC 6600 facility. The data is then sorted and rewritten into random access files. The GGP System can simultaneously display information from three similar files with each file consisting of up to 360 data sets, each containing up to 3,500 points per data set. The maximum data access capacity at any one time is 3.78 million data points.

The GGP System allows the engineer the following direct and immediate interaction with his data:

1. Immediate visualization of data in almost any desired form (i.e., linear, logarithmic or polar X-Y plots, either two- or three-dimensional).

2. Easy data editing and manipulation.

3. Easy comparison of any portion of one data set to other similar data.

4. Several curve fitting and pairing techniques.

5. Cross plotting and averaging techniques.

6. Hardcopy output.

Examples of these types of graphical output are shown in Figures 13 and 14.

Program control for GGP is provided primarily through the use of menus. The options available to the user at a particular point in the program are displayed in a menu. Figure 15 is an example of one of the many menus available to the user. Selections on a menu may cause either other menus or other plots to be displayed, allowing the user to control the types of plots to be displayed and the method of analysis to be performed on the data. Other menus are available that allow the user to select information for display or modification.

In addition to menus, program control is also provided through the input devices: keyboard, dials, joy stick, and data tablet. Keys are provided on the keyboards which when depressed will automatically generate a hardcopy plot.

CONCLUDING REMARKS

Graphic display systems should have the ability to logically control the program execution, generate high level graphic displays, provide the capability to display and modify the values of program variables and arrays, and automatically create hardcopy output. Graphics standardization and device independent code should also be developed that assure machine independence and terminal transferability. Two application programs were discussed, using different host computers, operating systems, communicator networks and graphic display terminals. Though vastly different in scope and operation, the application program approaches to graphics through BIGS and the graphic routines called were, in general, identical.

REFERENCES

1. Quenneville, C. E., BCS Interactive Graphic System, Boeing Computer Services, Inc., 10201-044, June 1975.
2. Information Display Products: Tektronix Plot-10 Terminal Control System. Document No. 062-1474-00, Beaverton, Oregon (1972).
3. Herness, E. D. and Kriloff, H. Z., NASTRAN Pre- and Postprocessors Using Low-Cost Interactive Graphics. Proceedings of the 4th NASTRANS User's Colloquium, Langley Research Centers, August 1975 (NASA TM X-3278).
4. Herness, E. D. and Tocher, J.L., Design of Pre- and Postprocessors. Proceedings of the International Symposium on Structural Mechanics Software, University of Maryland, June 1974.

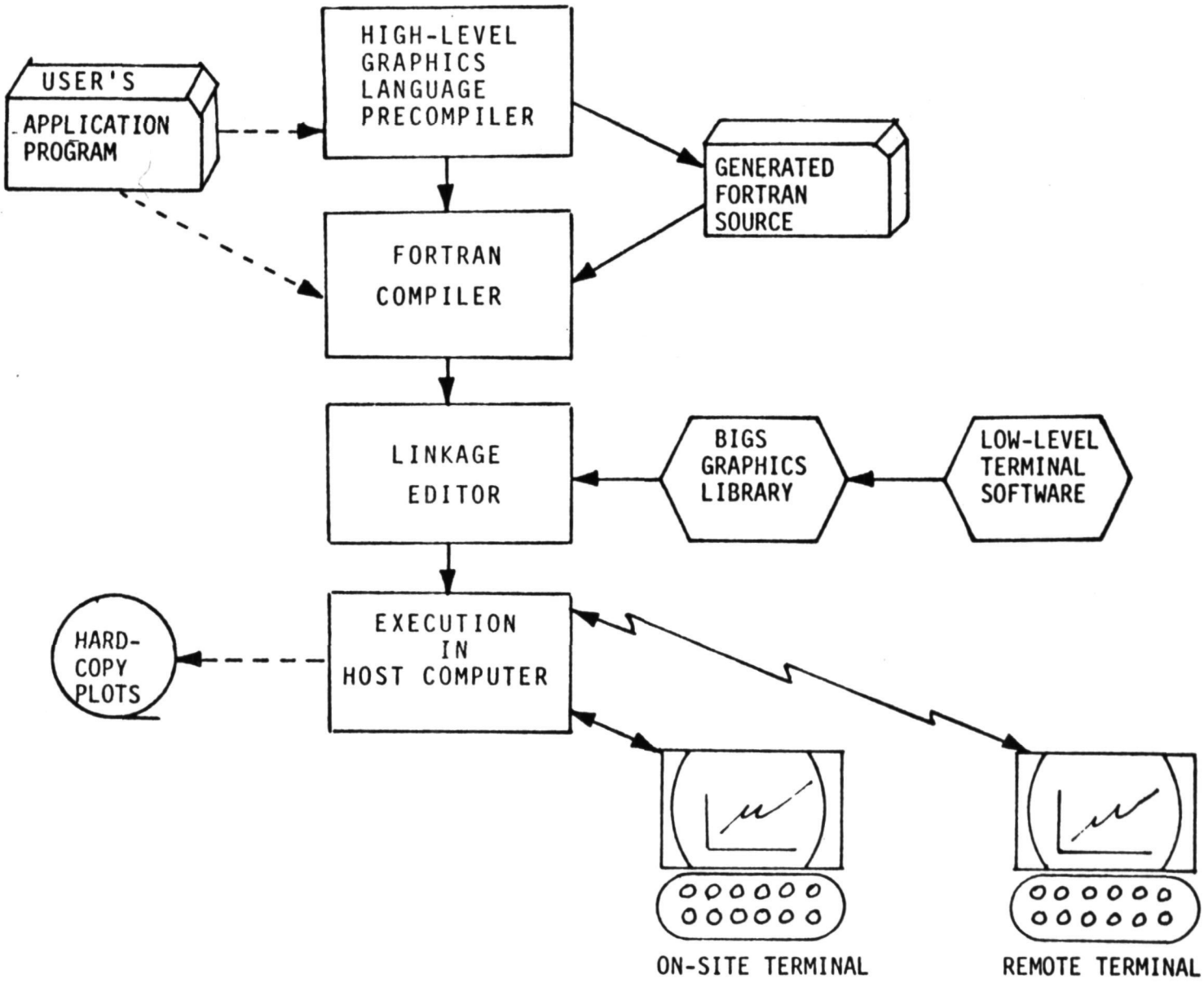

Figure 1.- BIG System Configuration - Large Scale Computers.

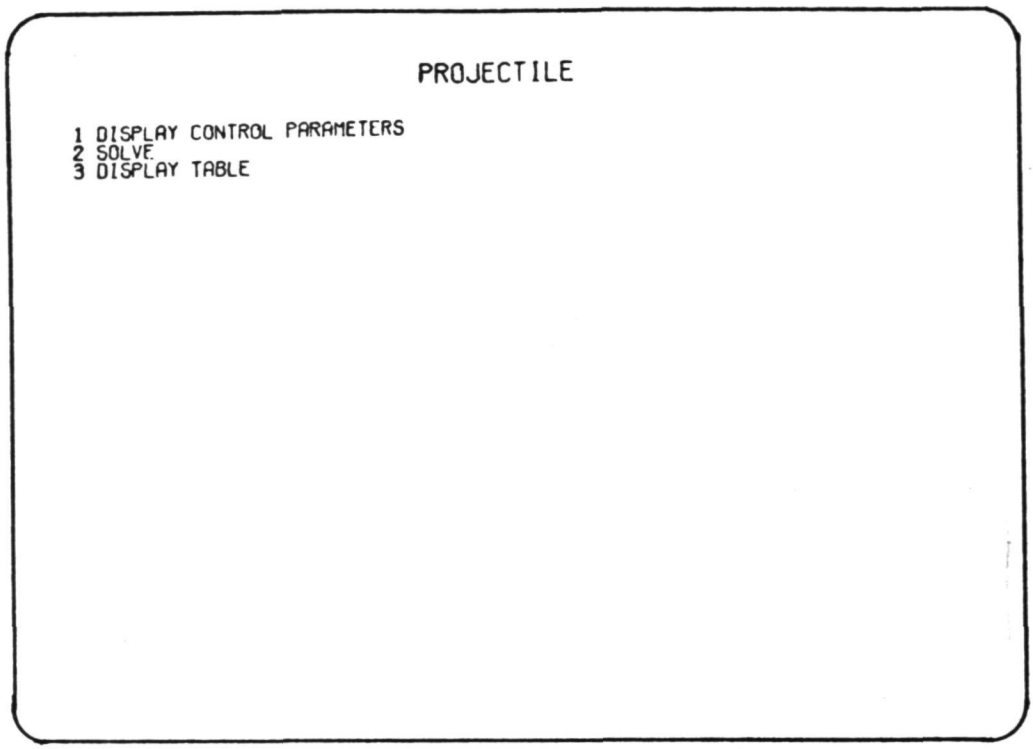

Figure 2.- Example of Display Menu.

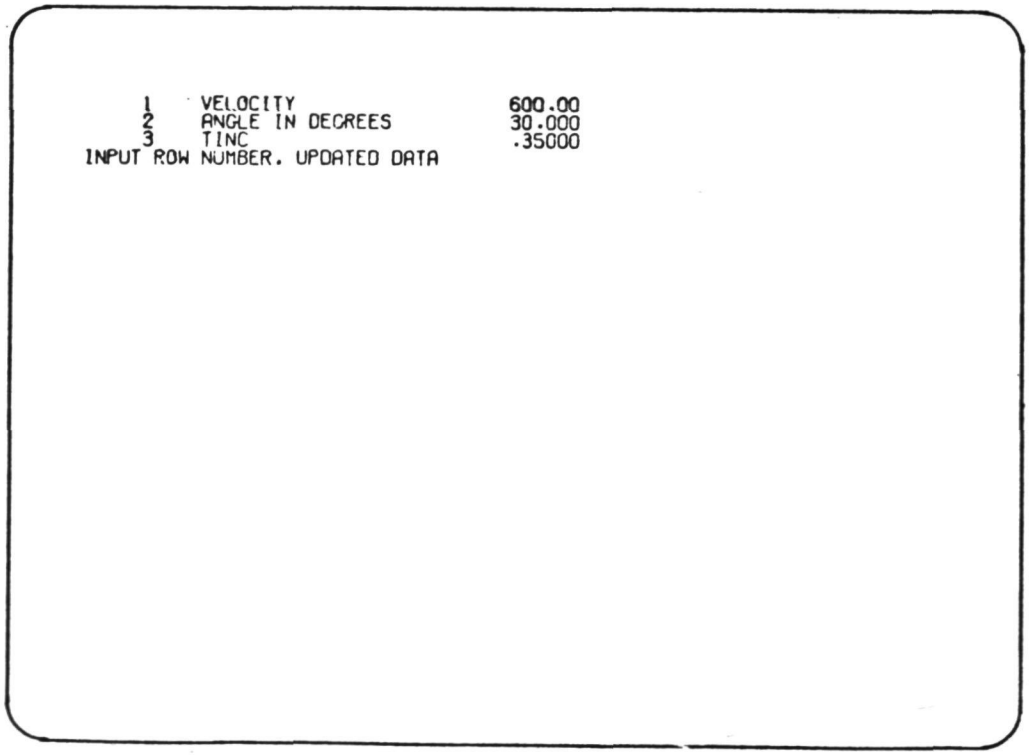

Figure 3.- Example of Display Data Statement.

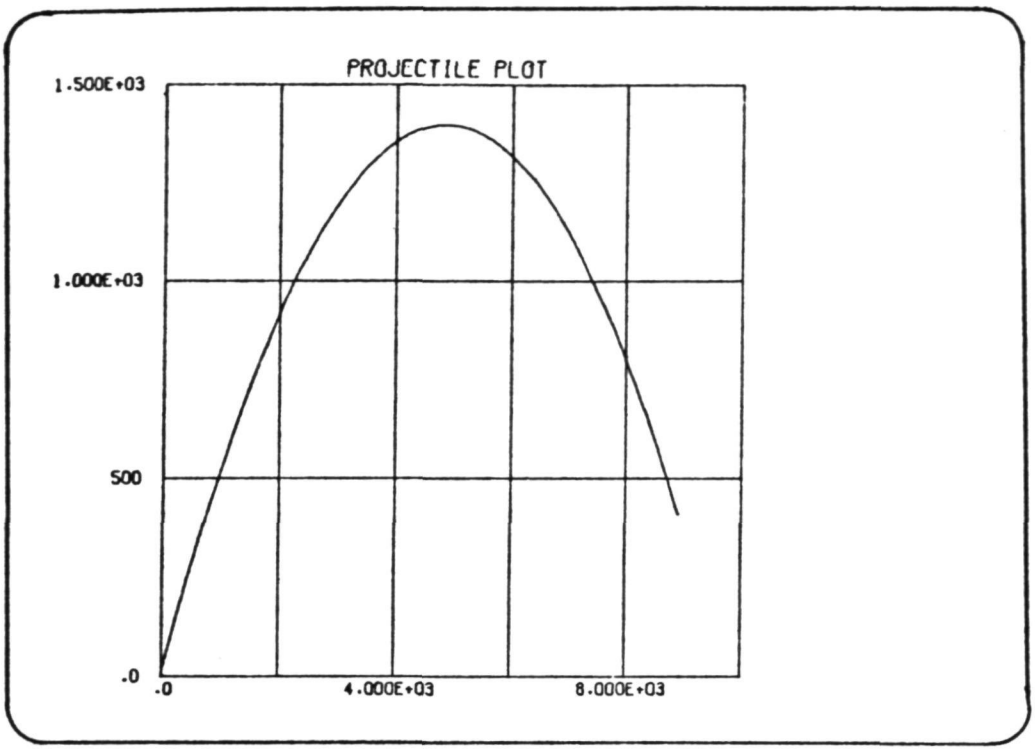

Figure 4.- Example of Data Plot.

PROJECTILE TABLE

NUMBER OF POINTS	INDEPENDENT	SQ DEPENDENT
1	.0	.0
2	181.87	103.02
3	353.74	202.10
4	545.61	297.23
5	727.48	388.42
6	909.35	475.66
7	1091.2	558.96
8	1273.1	638.31
9	1455.0	713.72
10	1636.8	785.18
11	1818.7	852.70
12	2000.6	916.28
13	2182.4	975.91
14	2364.3	1031.6
15	2546.2	1083.3
16	2729.0	1131.1
17	2909.9	1175.0
18	3091.8	1214.9
19	3273.6	1250.9
20	3455.5	1282.9

OPTIONS: REPLACE.I.X.Y DELETE.I INSERT.I.X.Y UP.N DOWN.N UPDATE
LNLN LGLN LNLG LGLG PL PLG NPTS.N

Figure 5.- Example of Data Table Display.

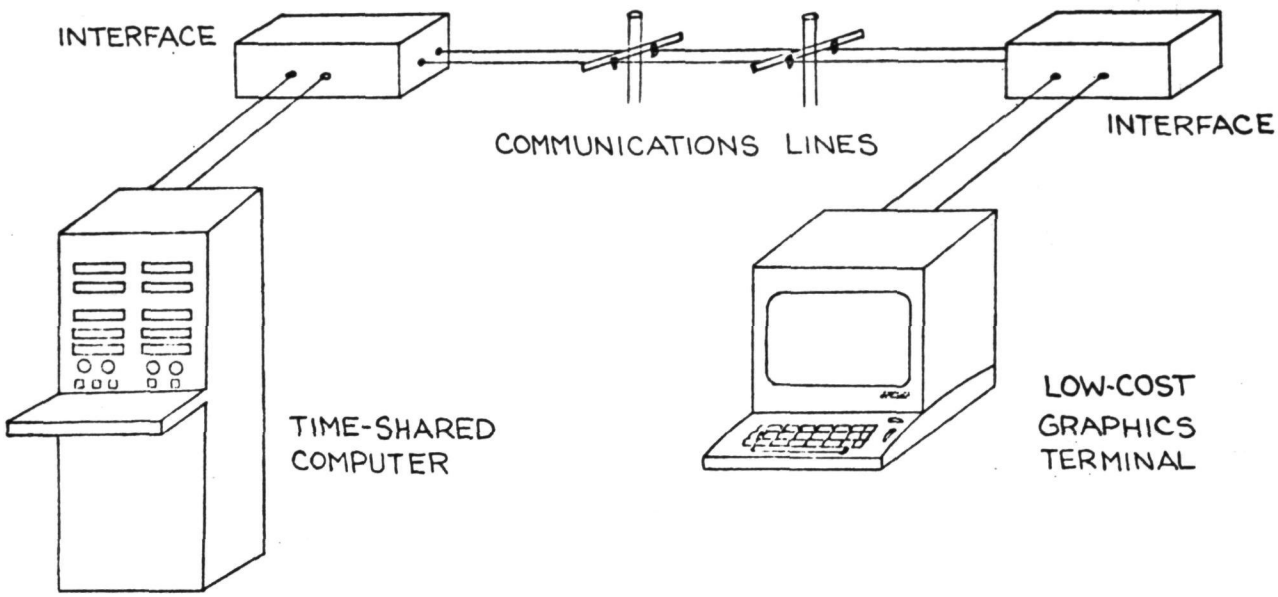

Figure 6.- Hardware Elements of a Low-Cost Graphics System.

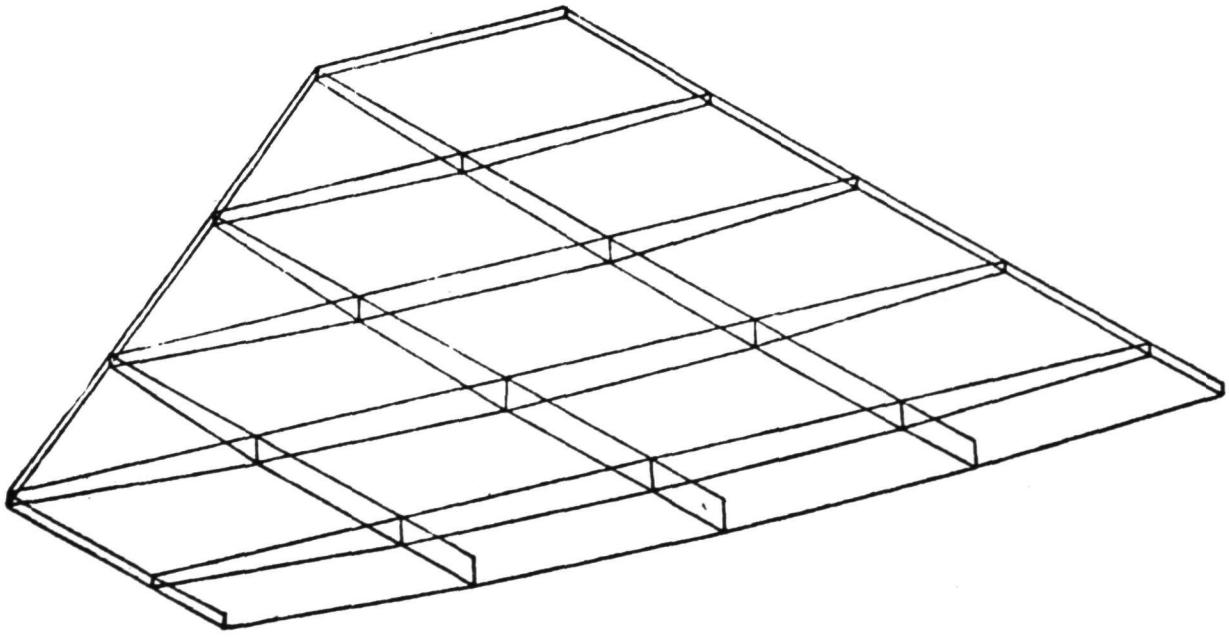

Figure 7.- Demo 1-1 Delta Wing.

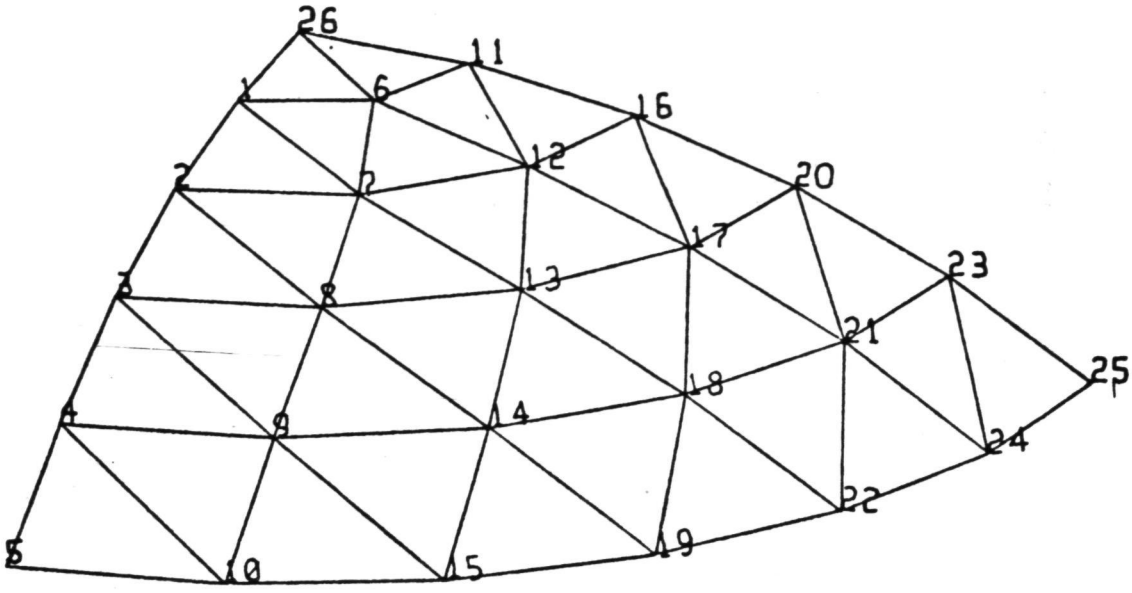

Figure 8.- Demo 1-2 with Grid ID's.

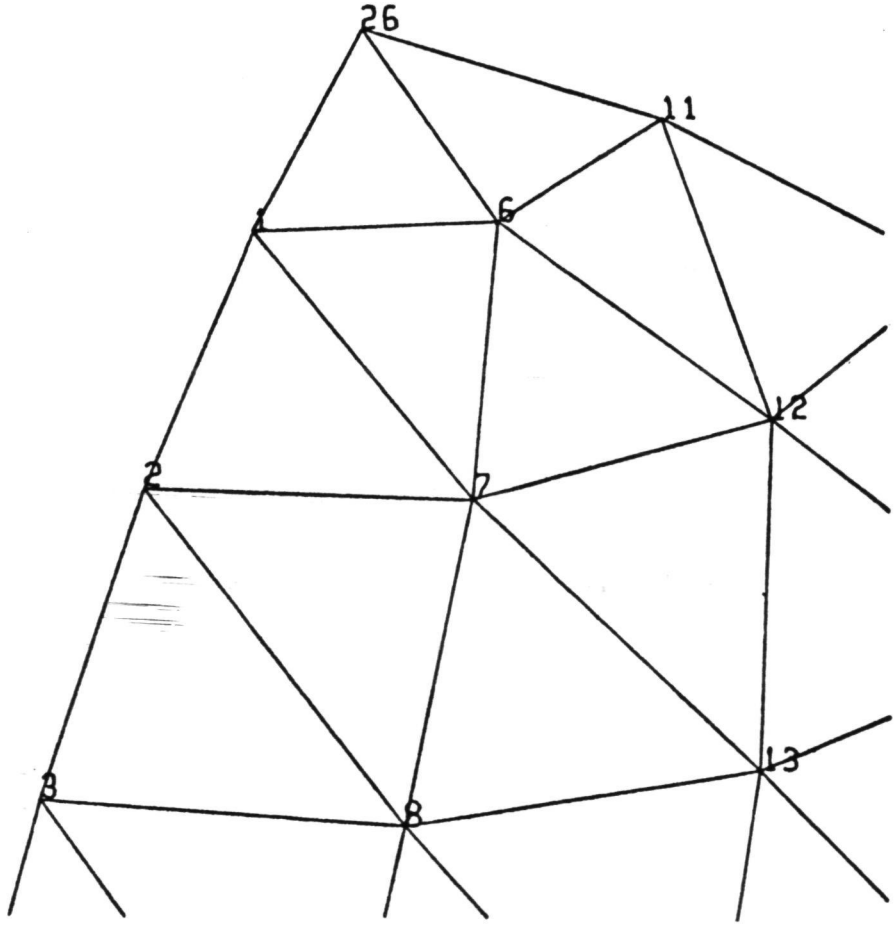

Figure 9.- Demo 1-2 Blowup.

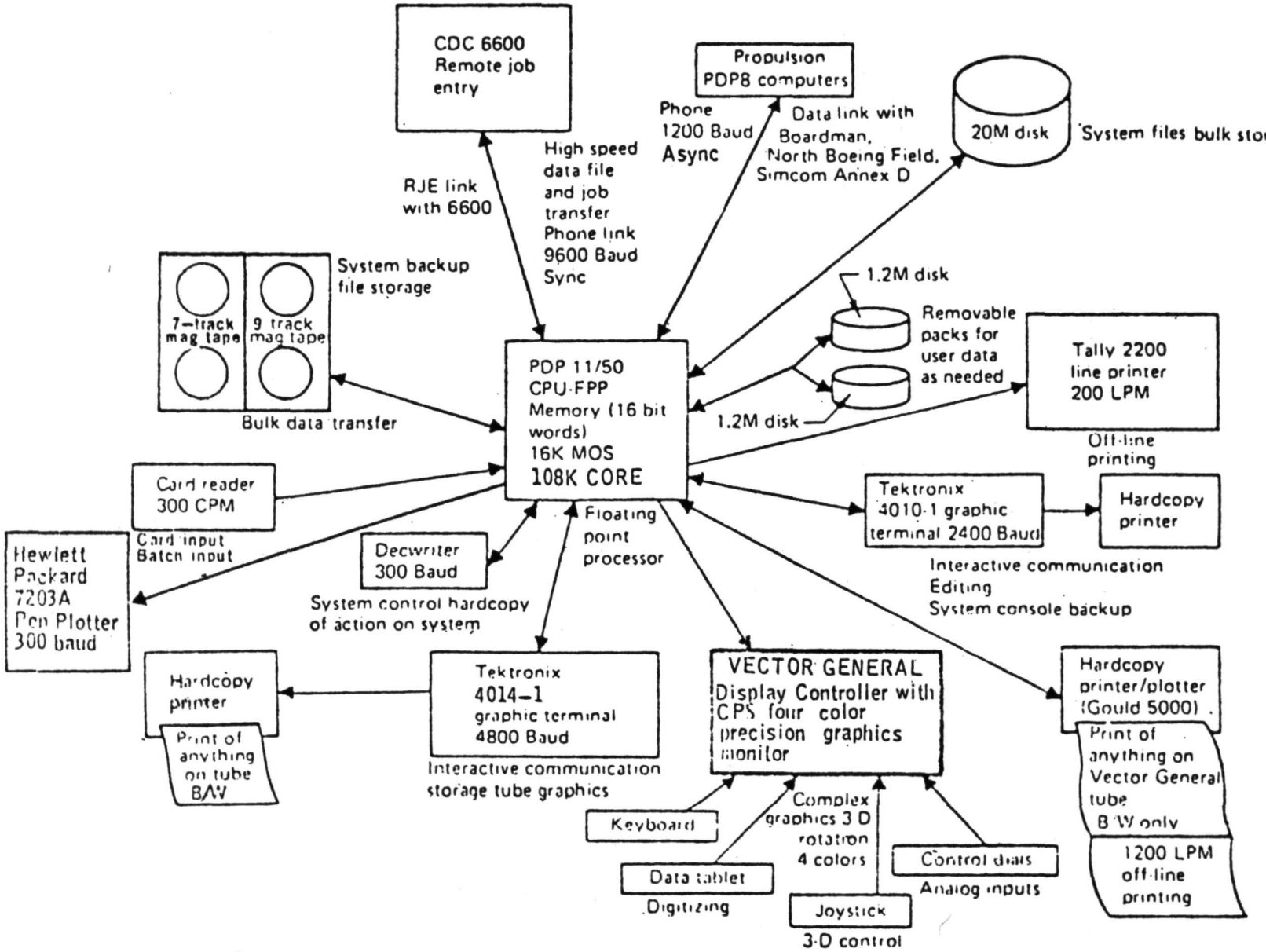

Figure 10.- Propulsion Computer System PDP 11/50.

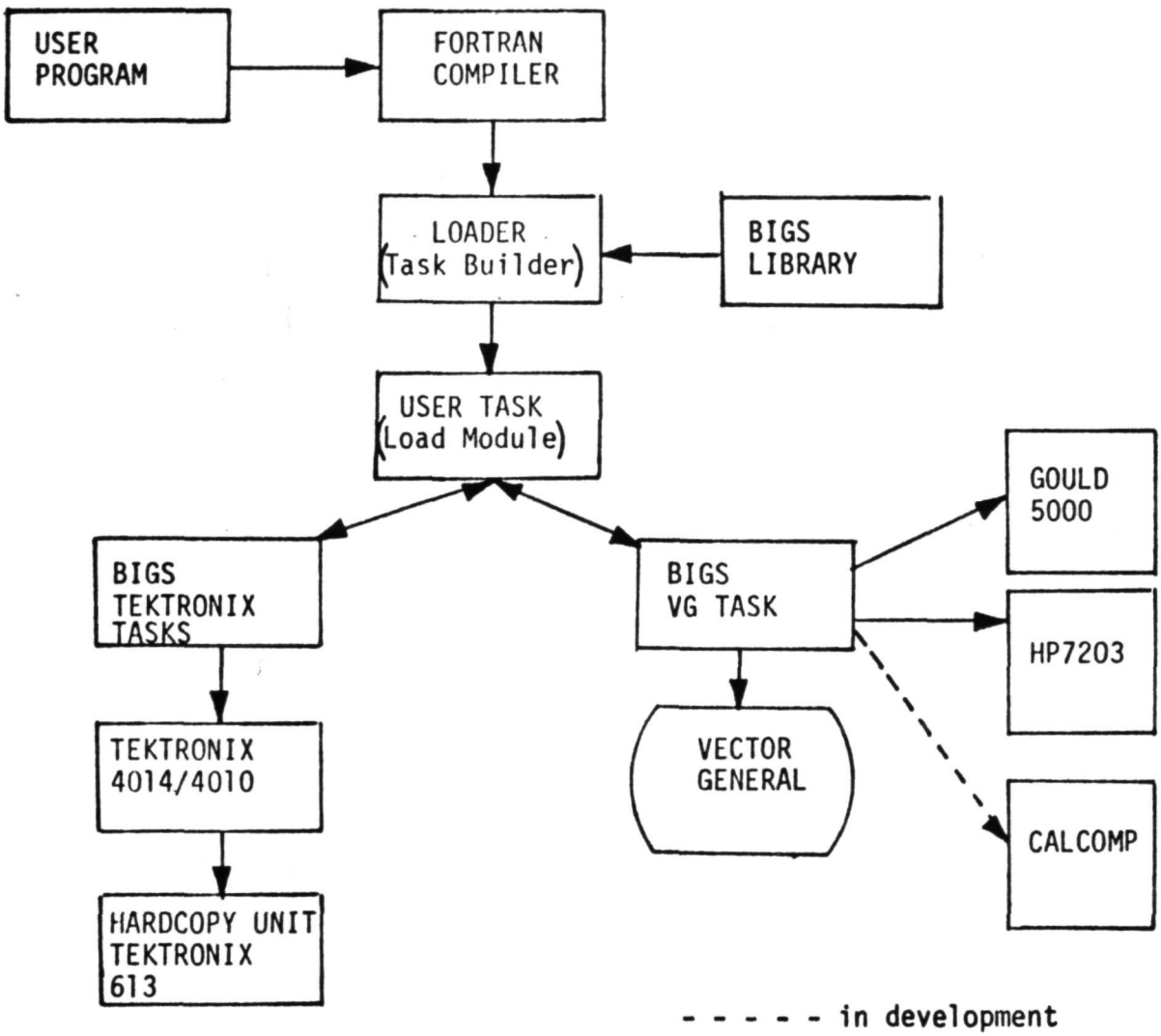

Figure 11.- BIGS Mini Configuration - PDP 11/50.

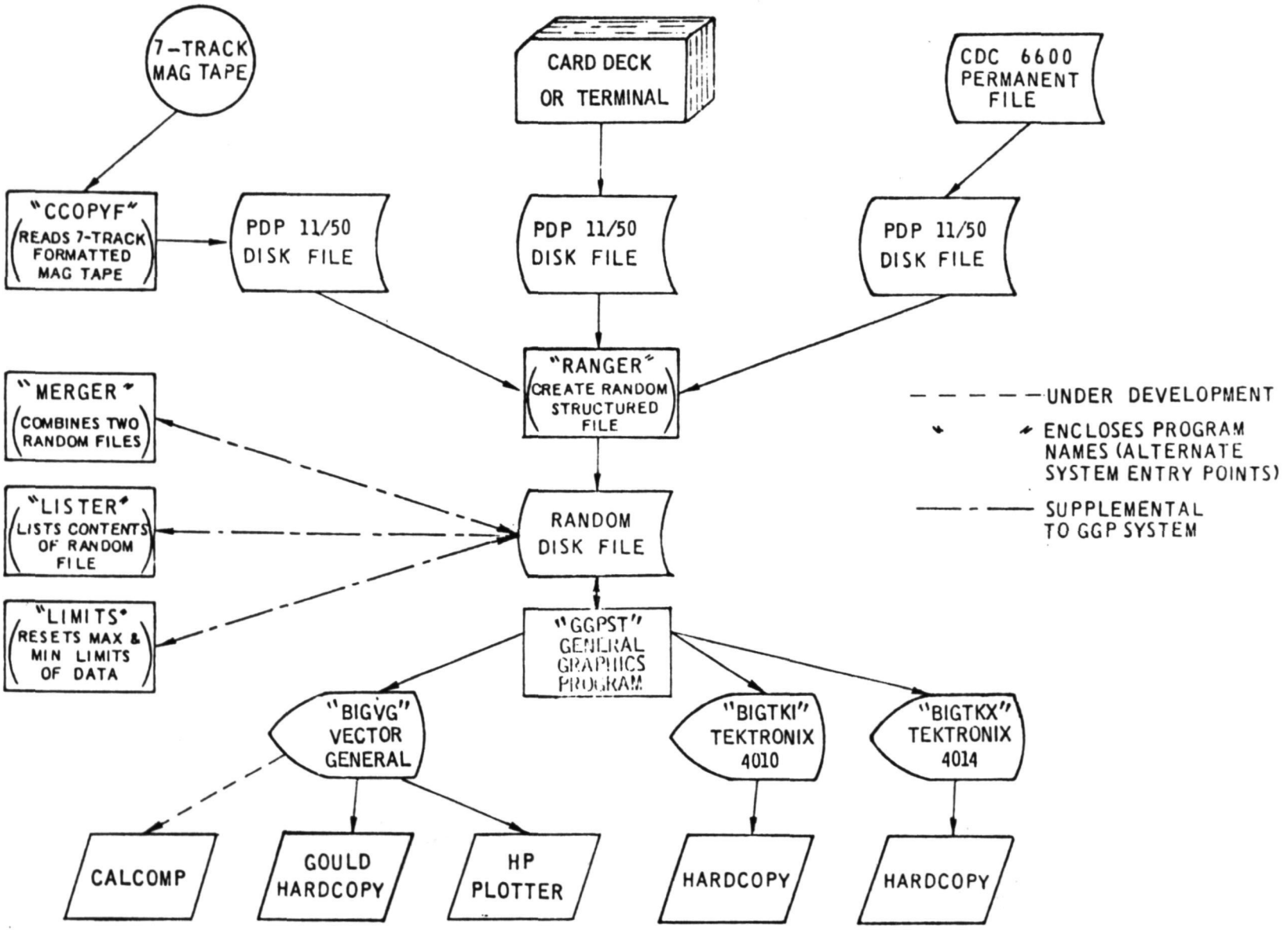

Figure 12.- GGP System.

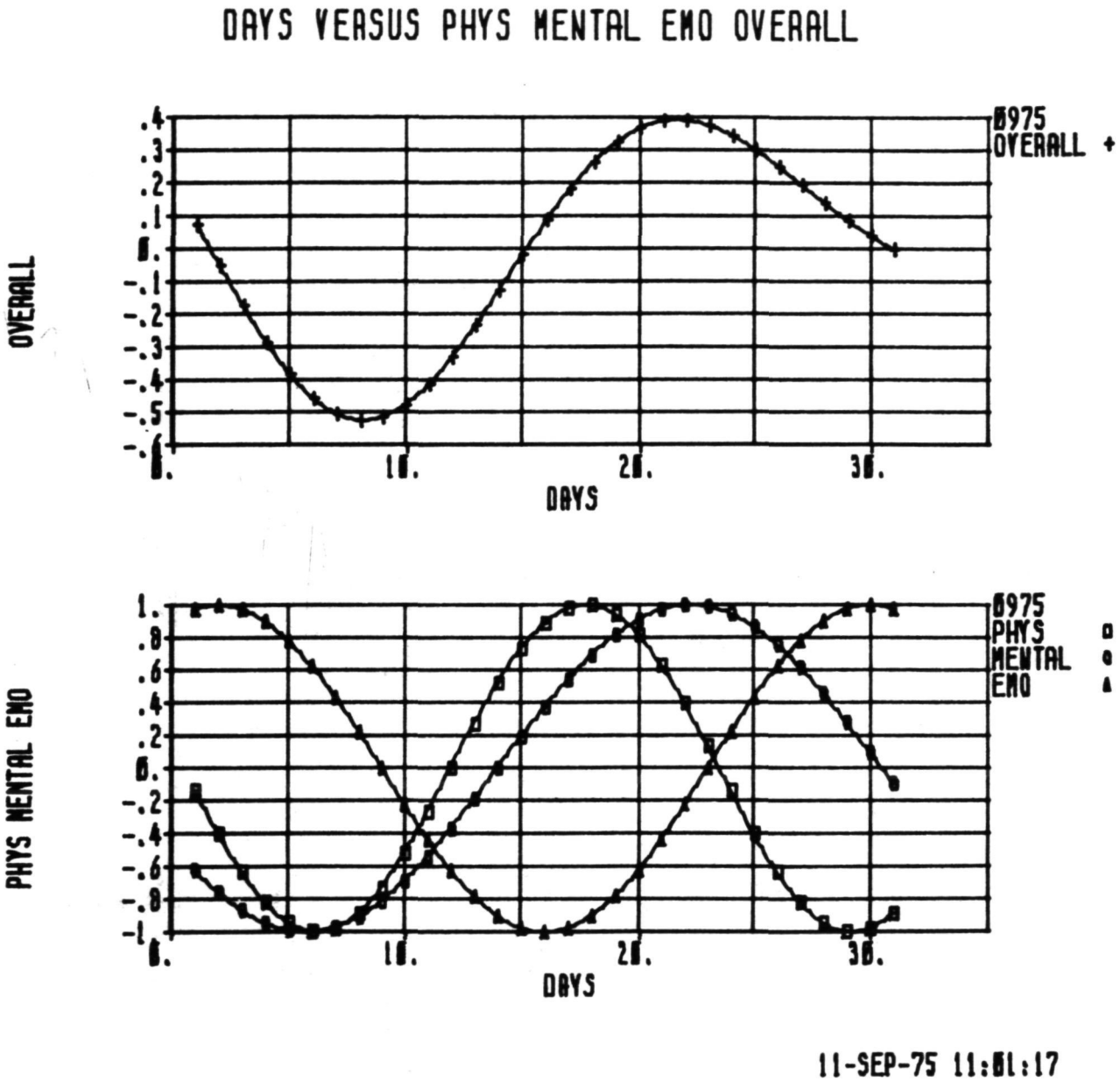

Figure 13.- Multiple Plot.

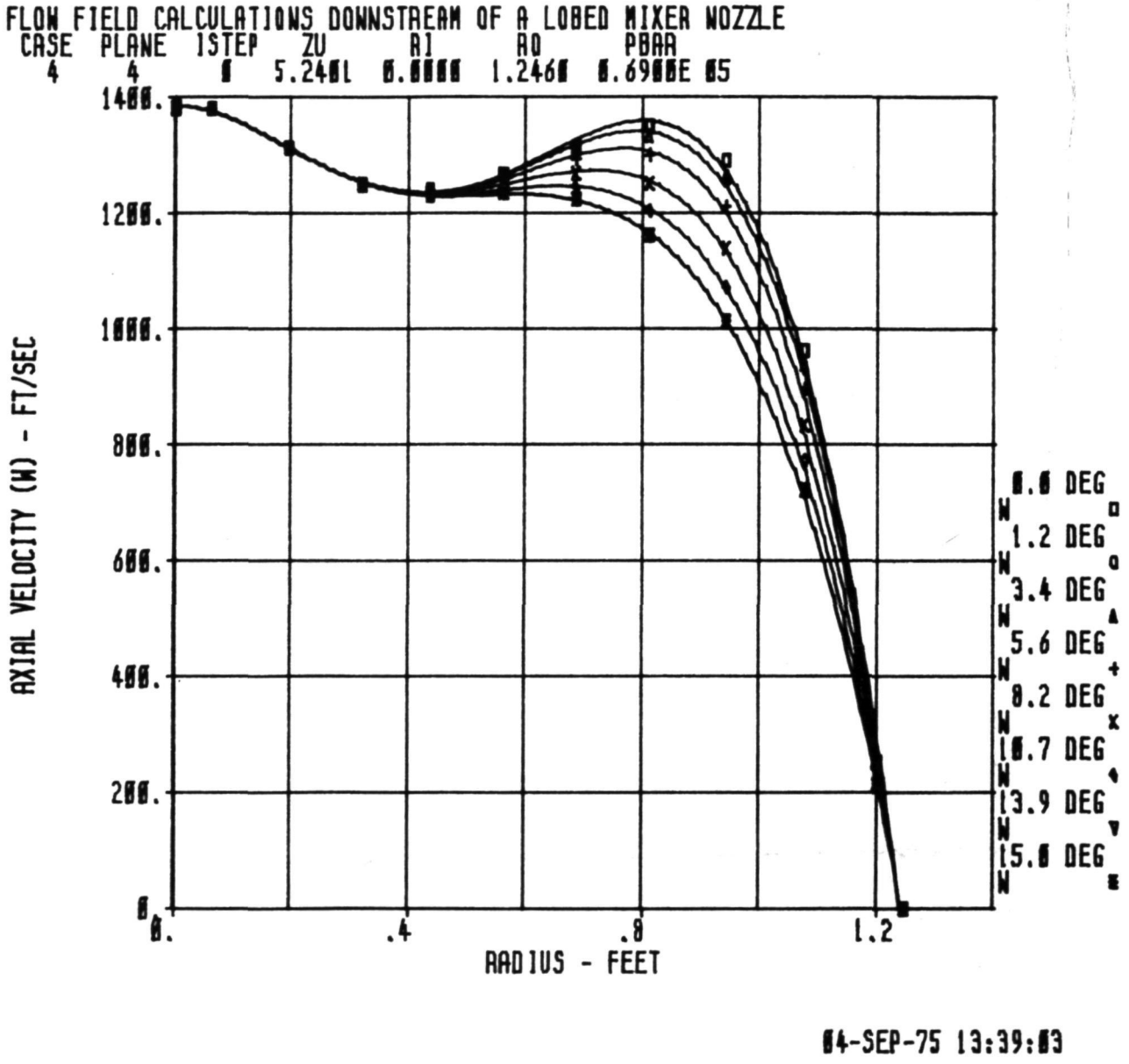

Figure 14.- Typical Display Plot.

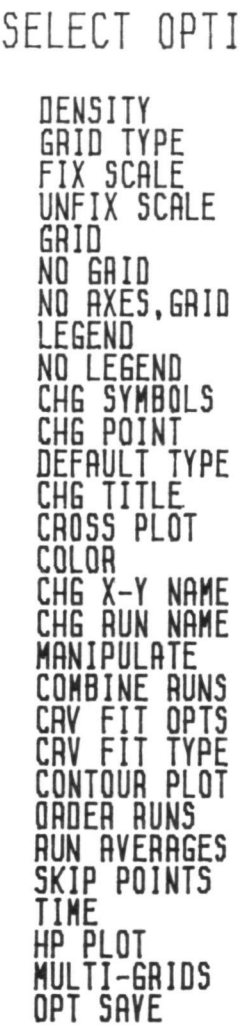

Figure 15.- Example of Menus Available with GGP.