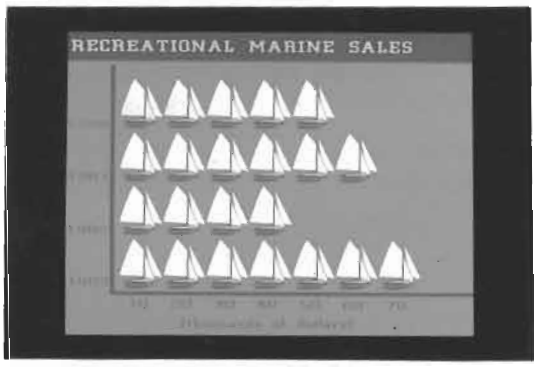
SEPTEMBER 1983

HEWLETT-PACKARD JOURNAL

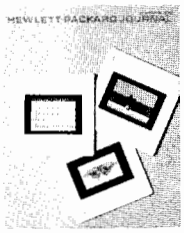


HEWLETT-PACKARD JOURNAL
Technical Information from the Laboratories of Hewlett-Packard Company

SEPTEMBER 1983 Volume 34 • Number 9

Contents:

In this Issue:



Computer art isn't so new anymore. Examples of it appear frequently in magazines—photographs of television screens with pretty pictures on them. You can usually see right away that these pictures are made up of dots, each a different color. The dots are called pixels. If you zoom in on a portion of one of these pictures—magnify it so it fills the screen—the dots in most cases just get larger. The number of pixels the screen can display limits the amount of detail the artist can put into the picture.

Now come with me to a demonstration of the HP 2700 Color Graphics Workstation, the subject of this issue. A bright picture of the United States of America fills the screen—so far a conventional-looking television-like display. As we spin one of the thumbwheels on the keyboard, the display zooms in first on the west coast, then on a small area near San Francisco. Details appear that couldn't be seen in the original picture. The dots don't get larger, at least up to a point. Where did that extra detail come from? It's stored in the HP 2700's memory, which can hold information on 5000 times the number of pixels that can be displayed on the screen. On command, the workstation can send this information to a copy camera to be put on film with all of the fine detail preserved.

Now watch as Scott Connor, an HP graphic artist, sits down to create a picture on the HP 2700. He's never seen the machine before. One and one-half hours later he's produced the brightly colored space-fantasy slide shown on the cover. He's been able to draw objects freehand, move these objects, change their size and orientation, select a palette of 16 colors to work with out of 4096 available colors, and fill in areas with colors automatically. In about the same amount of time, Journal Associate Editor Ken Shaw produced the sailboat bar chart on the cover. The third cover slide, the butterfly, shows what an accomplished artist can do.

The HP 2700 isn't a computer. It's more like a computer terminal, but it doesn't need a host computer to create graphics. Most of the capabilities graphic artists need are built into it. However, with a host computer, the range of possibilities expands. Get the full story from the HP 2700's designers in the next 38 pages.

-R.P. Dolan

A Color Presentation Graphics Workstation

Here's a remarkable new workstation family for presentation graphics design, decision support graphics, and graphic art. It features powerful, easy-to-use application software and full block-mode terminal capabilities.

by Sharon O. Mead, William R. Taylor, Kenneth A. Mintz, and Catherine M. Potter

THE HP 2700 (Fig. 1) is a family of high-performance color graphics workstations offering local graphics design and output features that enable users to create professional presentation graphics and graphic art at a fraction of the cost of manual methods of design and preparation. The workstation's powerful features are accessible by a host computer so that it is possible to implement demanding graphics applications that do not require the computer to perform complex transformation calculations or transmit copious amounts of data to the workstation. The HP 2700 can also function as a block-mode computer terminal, making it a suitable display station for many general graphics applications. It has a high-quality color display, a keyboard with many specialized functions, a graphics input device consisting of a pair of thumbwheels and a button, and an optional graphics tablet.

Local Applications

Local applications software combined with high-resolution 35-mm slide output let even a first-time user produce presentation graphics quickly and easily.

PAINTBRUSH/2700 combines sophisticated picture creation capabilities with the ability to manipulate and edit pictures created locally or by a host computer. Pictures can be created freehand using a graphics tablet or with a variety of drawing aids such as pen tips, arcs, curves, and defined shapes.

AUTOPLOT/2700 is a charting package that can accept input from a flexible disc, the keyboard, or a computer. The only information that must be supplied is the data to be graphed. However, most attributes of the graph can be changed interactively, including the size and location of axes, scaling, and the colors and patterns of lines, bars, and pie segments.

PRESENTATION/2700 is a utility software package that outputs pictures to a high-resolution film recorder for making high-quality 35-mm slides or other film copies. It also has the ability to upload pictures to the HP 3000 Computer in the same figure file format used by the DSG/3000, HPDRAW, and TDP/3000 software subsystems. These figures can then be transmitted to any device supported by the HP 3000 graphics capabilities, including the HP 2680A Laser Printer. Pictures can also be output to local plotters, to black and white printers, or through a video interface to cameras and monitors.

Graphics Features

The HP 2700 combines the benefits of vector graphics with raster graphics. The display of an HP 2700 is refreshed from a raster memory that has four memory bits for each screen color dot, or pixel. By contrast, a vector graphics display is refreshed from a display list of commands that direct an electron beam to move and draw lines on the display. Vector technology produces clean lines, but is usually monochromatic and limited in the number of lines that can be drawn before picture quality deteriorates because of flicker. However, the image on the screen can be changed quickly, simply by changing the display list.

The advantages of raster-scan technology are that color is readily available, the image can be arbitrarily complex, and solid areas can be filled quickly and without flicker. However, since lines are made up of dots, stairstepping can occur in low-angle lines. Also, changing the image can be difficult without resending the entire picture.

A major contribution of the HP 2700 is that, while it is a raster display, commands sent to the terminal to create graphic images are also retained in memory in a vector list. Just as with the vector display, these stored commands can then be manipulated by further commands to change the displayed image without retransmitting all of the original commands needed to create the image. Parts of the picture can be identified as objects which can be moved, scaled, or rotated independently of the rest of the display. This local storage of graphics commands allows true zoom and pan, and actually produces higher resolution on the screen instead of just making each dot bigger as you zoom in. While the screen is a rectangular array of 512 by 390 pixels, the user can address 32767 by 32767 points. A small area on the screen can be expanded by a single command to reveal a much higher level of detail than was initially visible. In fact, the user can view as much or as little of the address space as desired, and can look at up to 256 noncontiguous or possibly overlapping parts of the space at one time.

A palette of 16 colors out of a choice of 4096 is displayed at any one time. Up to 256 palettes can be stored in the workstation. Interesting animation effects can be created by rapidly rotating the active palette among several stored in the terminal. Colors can easily be selected for a palette by use of a slide-bar menu that allows interactive manipulation of the colors. The results can be seen in the picture on the screen.

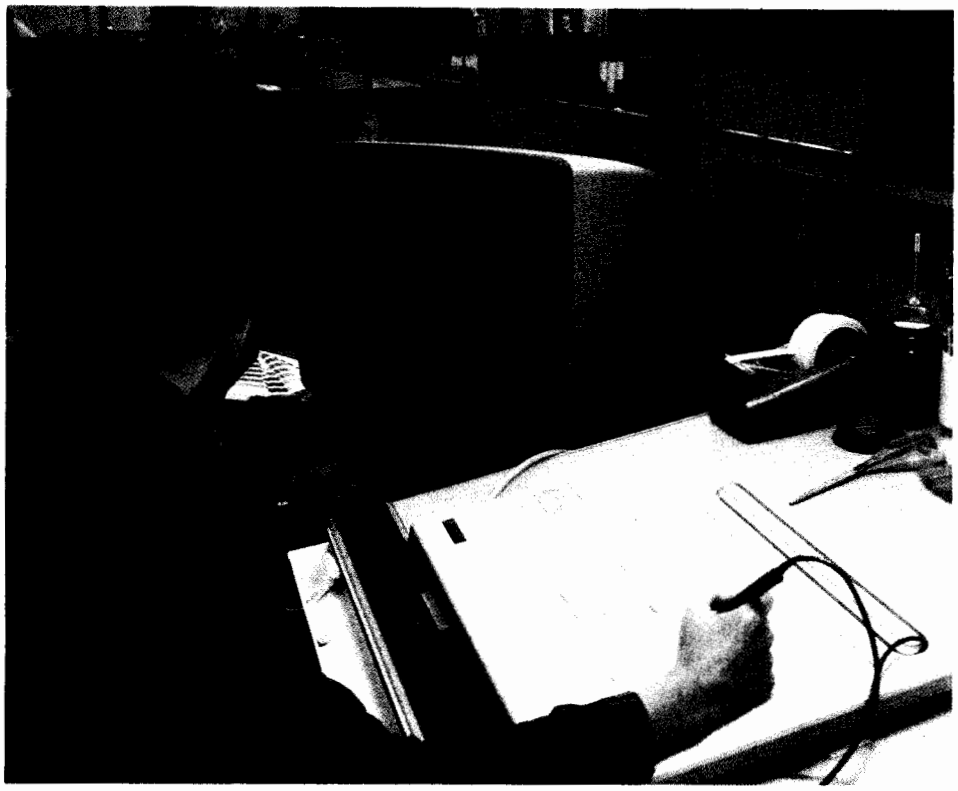


Fig. 1. *The HP 2700 Color Graphics Workstation speeds the preparation of high-quality presentation graphics. The designer works with a keyboard, thumbwheels and a button, or an optional graphics tablet.*

The HP 2700 supports a full polygonal area fill capability. This feature allows application programs to specify the vertices of any arbitrary polygon including concave, convex, and doughnut shapes and have the area filled in with either a solid color or a pattern.

Color mapping, full polygonal area fill, multiple views and windows, object manipulations, and other advanced graphics features of the HP 2700 are explained in detail in other articles in this issue.

Alphanumeric Features

In addition to all of its graphics features, the HP 2700 is a complete alphanumeric terminal compatible with the HP 2622A Data Entry Terminal. Applications that depend on block mode and format mode, such as VPLUS/3000, will run unchanged on the HP 2700. There are new display enhancements that allow applications to make use of color in alphanumeric applications as well. The use of color on a menu or form allows the reader to discern much more information than is possible with a monochromatic display.

A local mass storage option allows pictures and alphanumeric information to be stored for later viewing or retransmission to the computer. There is a friendly command interface for manipulating files and hard-copy devices. English-like commands and a system of prompting the user through softkey labels that change as a command is typed in make the I/O devices very easy to use.

Hardware Architecture

To provide the performance necessary to implement the advanced feature set of the HP 2700, a multiprocessor architecture was chosen. A multiprocessor architecture allows the device-dependent details of various I/O and display functions to be offloaded from the main terminal processor. This frees the main processor to deal with other computations, thereby increasing total system performance. The multiprocessor architecture also increases the modularity and expandability of the terminal by placing the device-dependent functions into intelligent I/O and display controllers. Thus, individual modules can be enhanced or replaced with minimal impact on the main terminal firmware.

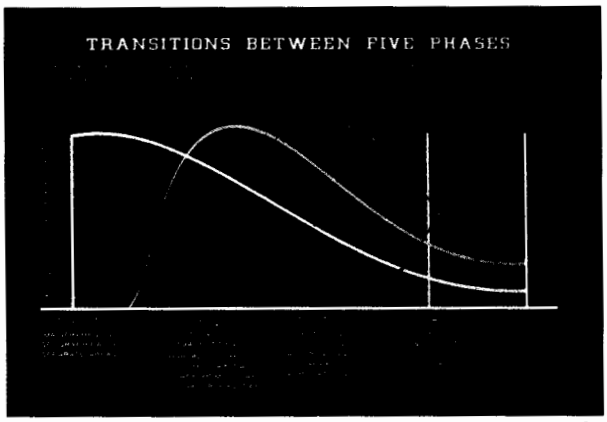
Processor-Independent Bus

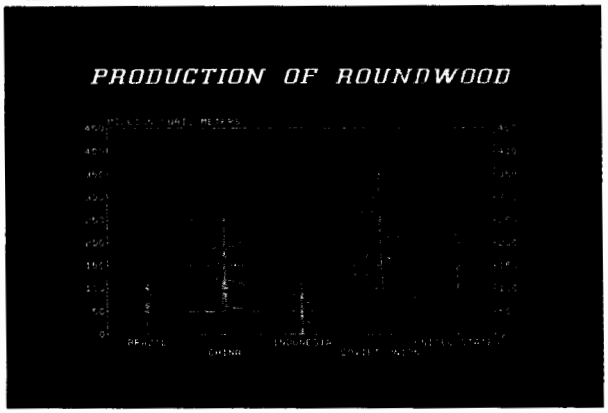
To implement the multiprocessor architecture, a new bus named the processor-independent bus (PIB) was defined. The processor-independent architecture of the PIB allows different modules to use different processors and lets the product grow with technology. As new, more-powerful processors and device controller chips become available, new boards can be designed to enhance the capabilities of the product with minimal impact on the rest of the system.

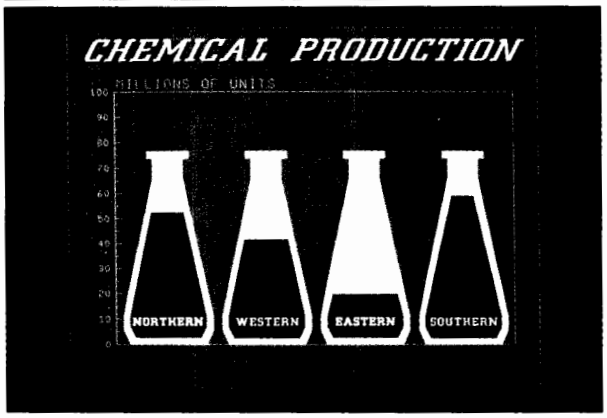
The HP 2700 modules are designed to minimize the need for jumpers for configuration. For example, the starting address for the memory board is configured by firmware instead of by hardwired jumpers; this simplifies assembly and reduces configuration errors. The modules are self-identifying, so that, with few exceptions, they can be installed in any PIB slot.

PIB modules can be categorized into four major types: channels, memory, master controllers, and dumb slaves. A channel is a microprocessor-based controller that supports communication with other modules through system (shared) memory. By using system memory as a communication path for control information and a buffer for the transfer of data, the channel initiating a complex data trans-

Examples of HP 2700 Graphic Art







fer operation is relieved of the burden of managing the details of the transfer. The channel that initiates these operations is called the master controller. An alternative implementation for a controller is the dumb slave, which supports only direct communication with the master controller. This implementation is used when the complexity of the interaction with the master controller is low, obviating the need to pass large amounts of data or control information to perform a single operation. The advantage of the dumb slave implementation is that it requires less hardware than a channel because the bus arbitration logic is not required.

The PIB has a 16-bit data bus and a 24-bit address bus capable of addressing 16 megabytes of memory. Memory can be accessed on both word and byte bases. Eight interrupt lines are provided along with power, clocks, and control signals for a total of 100 lines. Bus arbitration for each channel is controlled by the bus controller chip (BCC), a proprietary HP IC. By integrating the control functions in this 48-pin IC, a complex bus interface can be incorporated into each channel using a minimum of board space.

Three Module Groups

The HP 2700 can be divided functionally into three main groups of modules: the system group, the I/O group, and the display group (see Fig. 2). The system group provides the overall control of the terminal functions, including the execution of application programs. The core of the group is the MC68000-based processor board and associated ROM and RAM. The system group also includes the shared (backplane) RAM, which is used for interprocessor communication and for storage of vector lists. The modules in the I/O group provide the interface between the terminal and the wide range of devices supported by the HP 2700 including graphics tablets, plotters, printers, and discs. The digital modules in the display group translate vector endpoints and ASCII data into the individual picture elements (pixels) and manage the refreshing of the display. The sweep and monitor system converts the digital signals into visual information on the color CRT.

Software Architecture

Reflecting the modularity of the hardware, the HP 2700 functions are distributed among the maincode firmware and the firmware for the channels. The maincode firmware resides in topplane ROM and in local ROM on the main processor board. It interprets keyboard, datacom, and tablet input and controls the transfer of information between devices. Channel firmware resides in local ROM on each channel module. Each channel also has its own local RAM for internal data structures. Channels generally respond to requests specified by the maincode in shared memory data structures called channel programs. The channel is responsible for translating the high-level request into the sequence of low-level operations necessary to control the device. This high-level interface facilitates a high degree of parallelism which contributes to the performance of the HP 2700.

Maincode Organization

The maincode firmware is organized into four major modules (see Fig. 3): the alpha subsystem, the file subsystem, the I/O subsystem, and the graphics subsystem. The alpha subsystem controls the alphanumeric personality of the terminal. It consists of the main loop, the escape sequence interpreter (ESI), the alpha personality module, and low-level device routines. The main loop polls the terminal input devices (e.g., keyboard, datacom, etc.) and dispatches the data to the ESI, the file subsystem, the graphics subsystem, or a datacom output port. The ESI collects characters within an escape sequence and dispatches the escape sequence to the file subsystem, graphics subsystem, or alpha personality module for execution. The ESI also dispatches characters that are not part of an escape sequence to the

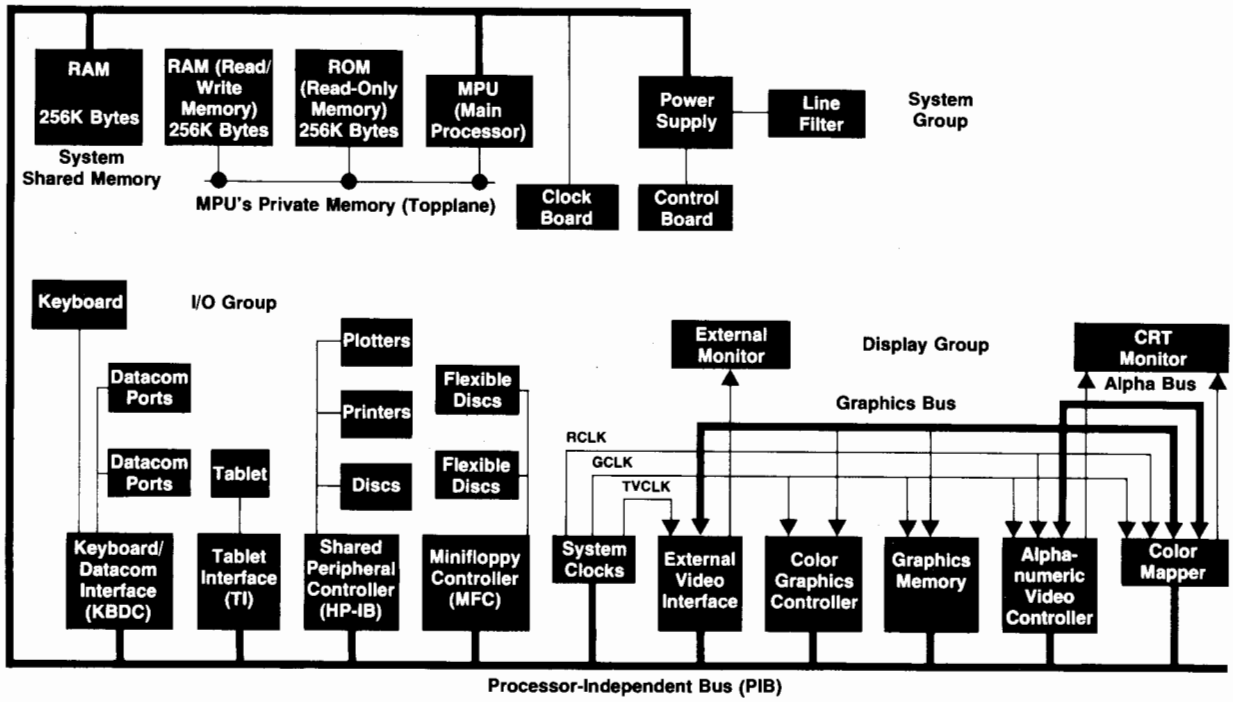


Fig. 2. *The HP 2700 has a multiprocessor architecture based on a processor-independent bus. Hardware modules can be divided into three groups.*

alpha personality module. The alpha personality module provides routines to control alpha windows and softkeys (screen-labeled function keys) and to provide for configuration options.

The file subsystem provides user-friendly access to HP 2700 peripherals by interpreting commands that are specified in the command window and in device-control escape sequences. It also manages the loading of RAM-based applications and intrinsics. The I/O subsystem dispatches the channel programs, which are used to communicate with the channels. It allocates control and data blocks in shared memory, sends the channel program to the channel, and interprets the completion interrupt from the channel. The I/O subsystem supports parallel I/O operations, permitting concurrent access to several channels as well as multibuffered transfers between the maincode and a channel.

The graphics subsystem performs all of the graphics functions of the HP 2700. It manages the graphics viewports and performs the necessary object transformations invoked through the graphics keys, escape sequences, and tablet input. The graphics subsystem maintains object and vector

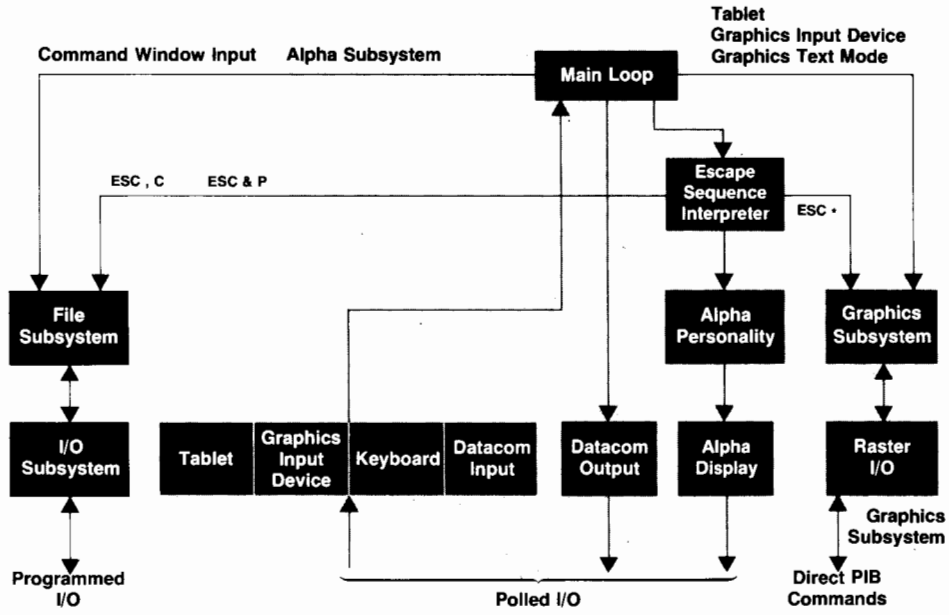


Fig. 3. *Maincode firmware in the HP 2700 is organized into four major modules, i.e., the alpha, file, I/O, and graphics subsystems.*

A System for Creating Graphics Presentations

The HP 2700 Model 65 Presentation Graphics Workstation provides the user with an HP 2700, 256K bytes of program memory, 224K bytes of vector memory, double-buffered graphics memory, dual flexible disc drives, a graphics tablet, the two applications software packages AUTOPLOT/2700 and PAINTBRUSH/2700, and the PRESENTATION/2700 utility package.

Graphics Presentation Software

Since a graphics presentation system is used by nonprogrammers, it must be both friendly and interactive. In PAINTBRUSH and AUTOPLOT, users invoke operations by selecting icon-like objects in a menu, instead of typing commands at the keyboard. This allows users to rely on their recognition memory; they are not required to memorize command names. To increase user confidence, instructional messages, quick system response, and information data in the menu give the user immediate, reassuring feedback, and defaults are provided wherever possible. In addition, AUTOPLOT provides help messages for explanations of all functions.

AUTOPLOT provides a user with the ability to transform data into pictures. This application provides pie, bar, linear, log, and scattergram charts, and means for combining several charts together in a single picture.

Generic charts can be customized by changing colors, placement, area patterns, and axes of different portions of the charts. Annotation can also be added to charts. For monthly reports, where the data changes but the format of the charts remains the same, only a data file needs to be changed for automatic generation of the new report.

PAINTBRUSH provides a user with picture and object library creation, picture manipulation and merging, and complete annotation ability. Using a graphics tablet, a user can draw pictures freehand or use drawing aids such as circles, arcs, rectangles, curves, and grids.

Once a picture is created, objects or groups of objects can be scaled, rotated, moved, copied, or deleted. Annotation can be added to both new and existing pictures, and other pictures can be merged with an existing picture.

Software Integration

The real power of a presentation graphics system is in the integration of the components of the system. Both AUTOPLOT and PAINTBRUSH are integrated with the outside world and with each other. The output of one can become the input of the other and vice versa.

PAINTBRUSH can take single and multiple charts as input. Each portion of a chart can become a separate object which can be scaled, rotated, moved, and copied singly or as a group. The chart frame, background, axes, tics, labels, titles, annotations, pie sections, bars, and scattergram markers can all become separate objects that can be manipulated. Text can be added or modified. A chart can be customized by replacing portions of the chart with a picture, and a single chart can be manipulated as a group.

AUTOPLOT can also accept PAINTBRUSH pictures as input and add charts into a picture. The PAINTBRUSH picture cannot be modified in AUTOPLOT, but it can be added to. For example, a background picture can be created in PAINTBRUSH and then charts can be added to it in AUTOPLOT.

AUTOPLOT works well with host applications. Data generated by a host program can be used as input data for AUTOPLOT. AUTOPLOT can work in remote mode and accept data from a host. This allows AUTOPLOT to be a local postprocessor to host data. For example, a chart can be easily produced from current data by sending a report from INFORM/3000 directly to AUTOPLOT/3000 instead of the terminal screen.

PAINTBRUSH can be integrated with other applications in several ways. Pictures created by PAINTBRUSH can be used in both local and remote applications. In a process control environment, the process control pictures (i.e., tanks, valves, fluid levels) can be created using PAINTBRUSH and then manipulated by a host program that controls the process.

PAINTBRUSH can also act as a postprocessor for other applications. Pictures made by other applications can be customized in the same manner that PAINTBRUSH can customize AUTOPLOT charts. For example, DSG/3000 charts can be modified, combined, or annotated in PAINTBRUSH.

-John Alburger
-Diane Rodriguez

lists in shared memory and, through the raster I/O module, reads the raster memory when a raster dump is performed through the file subsystem.

Interprocessor Communication

Three methods of communication are used to exchange information between the maincode and controllers: direct PIB commands, programmed I/O, and polled I/O. Direct PIB commands are I/O-mapped instructions which can be used to transmit up to 16 bits of data to a controller. This method of communication requires a minimum of overhead and is used when little or no exchange of information is needed to perform the operation. Although this is a relatively low-level form of communication, it can permit two processor-intensive operations to be performed concurrently (e.g., vector drawing by the graphics controller and object transformation by the graphics subsystem).

Programmed I/O communication uses a channel program which is initiated by a PIB command (start program). A channel program consists of a channel control block for transmitting I/O control and status information, a sequence of channel instructions, and one or more data buffers used in the interchange. Each channel instruction specifies a very high-level operation (e.g., read N sectors starting at address S). The channel reads each channel instruction and translates the operation into a sequence of device-dependent commands (e.g., seek track, transfer sector S, transfer sector S+1, seek next track, etc.). Multibuffering techniques are used to overlap channel I/O with the processing of the data by the maincode.

ROM/RAM Intrinsics Strategy

The HP 2700 terminal maincode firmware occupies 38 8K-byte ROMs for a total of 304K bytes. To protect this substantial investment, a strategy of dividing the maincode into intrinsic functions is used to achieve firmware flexibility and extensibility in the following ways. First, it should be possible to make slight changes in one ROM without having to replace any other ROMs. Second, appropriate mechanisms should be in place so that additions and changes can be implemented by adding ROMs with new code. Lastly, it should be possible to load system code from disc into topplane RAM, thereby providing a means to alter the terminal's capabilities on a temporary basis.

To achieve these aims, the HP 2700 ROMs are modular and relocatable. Each ROM contains only complete, relocatable assembly (or compilation) units, combined into one virtual assembly unit. Furthermore, each ROM contains the necessary information to link its code to the rest of the maincode, once the terminal is powered up and the locations of the ROMs are determined. Linking is done by the system initialization routine in the MPU board's system startup ROM.

The linkage information of each ROM (see Fig. 1) consists of:

- An inventory of its global procedures or intrinsics, the procedures contained in the ROM that are callable from the outside

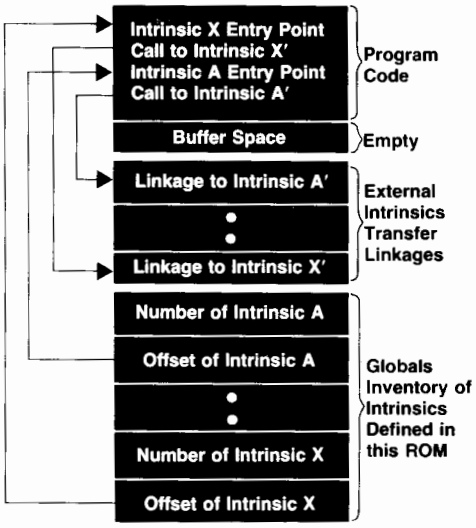


Fig. 1. *HP 2700 ROMs are modular and relocatable. Each ROM contains information to link its code to the rest of the maincode.*

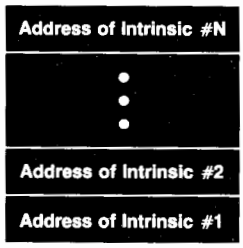


Fig. 2. *Each intrinsic is identified by an integer. The system intrinsics table contains the addresses of the intrinsics.*

- A set of transfer linkages for the external calls within the ROM to intrinsic procedures outside the ROM.

Each intrinsic in the system is identified by a unique integer, which indexes to an entry in the system intrinsics table containing the intrinsic's absolute address (see Fig. 2). This table resides in a reserved area of the local MPU-board RAM. To link the maincode ROMs, the system initialization routine fills in the system intrinsics table, calculating the entries by traversing the globals inventory list in each ROM while taking into account the ROM's starting address. This action binds all the ROMs' external transfer linkages, which are merely indirect jumps through the appropriate entries in the system intrinsics table.

ROMs are linked in order of ascending address. Therefore, ROMs at higher addresses may redefine intrinsics previously defined by lower-addressed ROMs. This mechanism allows new ROMs to replace intrinsic procedures in already existing ROMs, so code can be modified or added. Each intrinsic provides a potential access point into the maincode for making changes or upgrades. Therefore, in addition to the procedures needed for linking ROMs, the system intrinsic procedures include many strategically significant procedures such as device drivers, plus some stub procedures called from key points in the code.

ROM images can also be loaded from disc into topplane RAM, and then linked into the system following the physical ROM. In this way, special capabilities or personalities can be activated on a temporary basis (see "Disc Utility," page 23). RAM-based system code is also very useful for development purposes. For instance, the international language capabilities of the HP 2700 were developed in RAM after the rest of the maincode was already committed to ROM.

Polled I/O communication uses a shared memory data structure to transfer data and control information between the maincode and a channel. The maincode uses a channel program to transmit the location of the poll buffer to a channel and to initiate the polling mechanism. Poll buffers are unidirectional; that is, one processor fills the buffer, and the other processor removes the data. As with programmed I/O, double buffering facilitates the overlapping of input/output and data processing. This type of communication is used for relatively slow, asynchronous events such as tablet input and datacom input and output.

Extensibility

The HP 2700 software is designed to facilitate future enhancements of the workstation's feature set. All ROMs containing the maincode firmware are linked through a RAM-based vector table, which is constructed dynamically after power is turned on (see "ROM/RAM Instrinsics Strategy," above). Thus, an intrinsic function can be replaced by providing a new entry point with the same name in a different ROM. Similarly, RAM-based intrinsics (routines) can be loaded into topplane RAM and linked into the vector table. This provides a convenient method for

altering the terminal's personality and for adding restricted capabilities (see "Command Window," page 23).

The HP 2700 also supports RAM-based programs, which can be loaded into topplane RAM from a disc file. These programs provide local application capability for the HP 2700. The intrinsic organization of the maincode firmware permits a program to emulate normal terminal functions while providing specialized application support.

Because of the functional modularity of the firmware, new devices can be supported by replacing the ROM on individual channels with no or minimal changes to the maincode firmware. Additionally, each channel contains a limited amount of local RAM into which firmware extensions can be downloaded. This feature is used in the test strategy for the HP 2700. Downloaded software can provide lower-level access to the channel features. Thus, the full capabilities of the complex channels can be tested, and carefully constructed diagnostics can pinpoint problems during production as well as in the field.

Acknowledgments

Many thanks to all of the people of Data Terminals Division who helped make the HP 2700 a reality. There were many people from all parts of the division, especially product marketing and production engineering, who made important contributions and without whom it could not have been done.

For the top-level design of the PIB we would like to recognize Jon Gorman. For his guidance and encouragement through all phases of the software development, we would like to thank Doug Miller.

Designing Software for High-Performance Graphics

by Robert R. Burns and Dale A. Luck

A MAJOR CHALLENGE in the development of the HP 2700 Color Graphics Workstation was to provide the new and enhanced graphics features that earn the HP 2700 its appellation "high-performance" while maintaining compatibility with other HP graphics terminals. All HP terminals, including the HP 2700, receive graphics commands from a host computer via escape sequences. These commands and the HP 2700's default conditions are designed to allow a host application written for other members of HP's graphics terminal family to run on the HP 2700 with little or no modification.

Like all of HP's graphics terminals, the HP 2700 uses raster technology. The display is composed of 199,680 picture elements, called pixels, arranged in a rectangular matrix 512 pixels wide and 390 pixels high. The color memory associated with each pixel is four bits deep, and thus contains one of 16 values. On HP's black and white terminals, pixels are only one bit deep, and represent either black or white, depending on whether the pixel value is 0 or 1, respectively. On the HP 2700, one of 16 colors is displayed for each pixel value. These colors are determined by using a color lookup table, or color map, with 16 entries. For example, a pixel with the value 5 displays the color associated with entry 5 in the color map. The color map value for each entry is a 12-bit number, so there are 4096 possible color choices. Each 12-bit entry is composed of four bits each of the red, green, and blue components of the color. These four-bit component values are converted to one of 16 levels for the red, green, and blue CRT electron guns, as described in the article on page 18.

The host and user interfaces to the color map have been designed so that the user doesn't have to know these hardware details. Multiple color palettes, up to 256, can be defined to allow one-step changes of the entire palette of 16 colors in the color map. Color is specified using one of two color models: red, green, and blue (RGB), or hue, saturation, and lightness (HSL). These two color models are standard

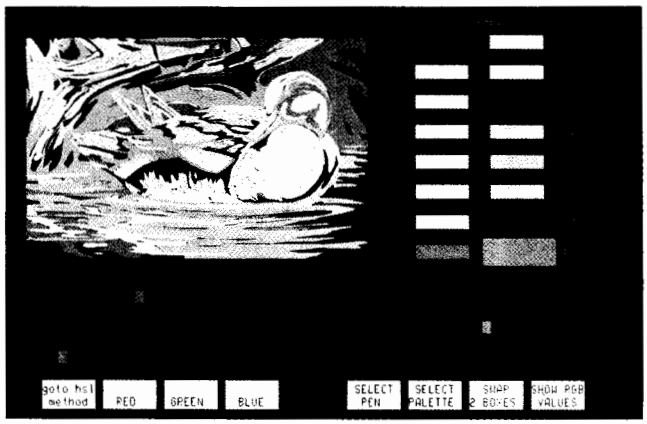


Fig. 1. *The* **COLOR** *key brings up this softkey and menu-driven environment and displays a graphic representation of three slide bars that can be used to select and alter colors. The slide bars control the hue, saturation, and lightness of the color or the red, green, and blue content of the color.*

for HP color graphics displays.[1] The RGB color model most closely describes the hardware implementation. The HSL color model simplifies color specification by making it more intuitive—hue is the color, saturation is how drab or colorful it is, and lightness is how non-black the color is.

A user can interactively change the color map, and thus the colors in the picture, with the **COLOR** key. This key brings up the screen-labeled function keys and menu shown in Fig. 1. The picture is scaled and displayed in the upper left corner, the current 16 colors are displayed as boxes in the right margin, and the current value of the selected color (the color highlighted with the enlarged box) is graphically represented in the slide bars along the bottom. The selected color is changed using the thumbwheels to highlight successive boxes in the color menu. Different components of the selected color are modified using the thumbwheels to manipulate the slide bars, which represent components in either the RGB or HSL color model. The function keys are used to specify what color component is to be changed.

Local Vector List

The vector list is the significant feature that separates the HP 2700 from all other HP graphics terminals. The HP 2647/48 and the HP 2623/27 Terminals store the graphics picture only in the raster memory. The HP 2700 stores the picture not only in its raster memory, but also in a separate list of graphics commands. When the user tells the HP 2700 to draw a vector, it draws the vector into raster memory and stores a copy of the command to draw the vector in its vector list memory. The screen is capable of displaying vectors that have (x,y) coordinates in the range of (0,0) to (511,389), but the vector list can store vectors in the range of (−16383,−16383) to (+16383,+16383). Along with the vectors, the vector list contains other primitives that choose pens, drawing modes, patterns, text, and area fill specifications. Vectors drawn while interpreting escape sequences proceed at a maximum rate of about 150 vectors/second, whereas vectors drawn from the vector list are drawn at up to 5100 vectors/second. This factor-of-30 improvement and the new high-level graphics features make highly interactive and user-friendly graphics application programs easy to write.

The HP 2700 can maintain more than one vector list. Each vector list is given a unique ASCII name.

The power of local transformations is a very useful feature. The HP 2700 provides this with a second level of picture structure, called objects, which are also identified with ASCII names. Objects make it possible to perform rotation, translation, and scaling on the vector list. The object is a separate entity and may reference any one of the vector lists in the terminal. A vector list, HOUSE, that describes a house may be used by objects HOUSE1 and HOUSE2 to place two different-sized houses at different parts of the picture. The order in which these objects are redrawn may also be specified so that it is possible to place an object on top of another and make objects invisible.

When drawing pictures the HP 2700 plots all the vector lists on an internal virtual space before putting the vectors on the screen. The virtual space is an imaginary square extending from (−16383, −16383) to (+16383, +16383). Vectors are defined in virtual space, and then moved to display space. This process is called viewing. Viewing requires two pieces of information: what part of the virtual space to get the vectors from (the window), and what part of the screen to place the vectors in (the viewport). The HP 2700 can maintain more than one of these window/viewport relations. Each is known as a view. One can specify the background color of each view as well as the color of a border that outlines the viewport. At any time there is only one active view. It is only through this view that vectors can be redrawn or changed.

An interactive user interface for the view features is provided by the graphics keypad. The **FULL VIEW** key sets the active window to ±16K, the entire virtual space. Hitting **FULL VIEW** again returns the window to its original setting. The **WINDOW** key puts up a box cursor. By moving the box and varying the size of the box the user describes a new window. The **ZOOM** key also uses the thumbwheels to vary the mapping of the window to the viewport by changing the dimensions of the window equally on both sides so that the window grows or shrinks around the center of the window and the viewport. The **PAN** key moves the window around while keeping it the same size. The **G AIDS** (graphics aids) key sets up some function keys that can be used to get information about the current view and change its values by means of the keyboard.

Polygon Area Fill

Local polygonal area fill is one of the HP 2700's major additions to the graphics feature set of HP terminals. Polygonal area fill is the shading in of a polygonal area on the screen with a color or pattern. In the HP 2700, a boundary color can be specified and turned on or off for any edge of the polygon. Up to two other colors can be specified for interior fill. All the HP 2700 drawing modes are supported. Predefined area fill patterns are provided that match fill patterns of the DSG/3000 software package.

As an example of the savings achieved by using local area fill, consider a 60-sided circle of radius 100 with the center at (100,100). The total number of horizontal vectors needed to fill this area is 200 (the diameter). Since these are all contiguous vectors, each vector requires a move and a draw and takes about 12 characters to specify, for about 2400 total characters. Using polygonal area fill, only the 60 exterior vectors are required to describe the circle. Since these are all contiguous vectors, the number of points is also 60, so only six characters are required per vector. The total number of characters sent from the host computer is 360. This represents an 85% reduction in the number of characters transmitted and a 500% improvement in the time it takes to draw the circle.

Double-Buffered Graphics Memory

A minimally configured HP 2700 has one set of four raster planes. This is where the pixels are stored to create the 16-color display. When a second set of raster memories is installed in the terminal, the user may use double-buffered redraws. The HP 2700 will coordinate redraws of the changing picture with alternating graphics raster memories to achieve hidden refreshes. This gives the user a less distracting update of the picture to work with.

Pick

Local storage of the vector lists provides performance benefits not only in display operations, but in user/host interaction as well. The keyboard's integral thumbwheels or the optional data tablet can be used to position the graphics cursor over any part of the display. The host computer can inquire what structure in the picture is being pointed to and the HP 2700 can respond. First, the user selects which of the graphics cursor styles is to be used by pressing **SHIFT GRAPHICS CURSOR**, which cycles through the three different graphics cursor styles: short crosshair, long crosshair, or box cursor. Next, the user positions the graphics cursor over the item on the display to be picked. The host can then inquire, perhaps after waiting for a keystroke to proceed, what object lies near the cursor. Nearness is specified by the box cursor's size, or a like-sized rectangle centered about a crosshair cursor. The host can distinguish selections not only by object and vector list names, but also by numeric tags, called pick IDs, applied by the host to primitives in the picture. If the HP 2700's memory contains an object near the cursor location, it can tell the host what object name, vector list name, and pick ID are associated with the item. Selection ambiguity between two or more items within the pick aperture is resolved in favor of the object that has been given the highest detectability (an object attribute) by the host. If ambiguity still exists, the most visible item (i.e., the last drawn) is reported.

This local pick capability can be exploited by a host in an interactive environment by building menus with unique pick IDs for each selection, creating an object with high detectability out of them, and making visible each menu as it becomes active (invisible objects are not pickable). The host then prompts the user to position the cursor over the desired menu selection, performs a pick, and identifies the user's selection based on the pick ID. This process allows the host application to remain independent of the actual menu layout, making customized user menus feasible.

Fonts and Labels

Another new graphics feature of the HP 2700 is enhanced flexibility for graphics text. This is achieved through the use of vector character descriptions and multiple character fonts. The vector character descriptions can include area fill specifications, allowing the font designer to control the apparent thickness of the characters. The character fonts are organized to allow full support of international character fonts. A stick font paired with its international language extension characters is supplied in ROM. The remaining seven character font pairs (or 14 fonts if international language extension characters are not required) are user-definable. The AUTOPLOT/2700 software (see page 31) provides two additional fonts: Roman and area filled bold. The PAINTBRUSH/2700 software (see page 34) provides a third additional smooth stick font.

The HP 2700 uses vector graphics character descriptions to overcome the limitations associated with raster characters. Raster characters can only be scaled by integer amounts and rotated by 90-degree increments, whereas vector characters can be scaled, rotated, and italicized with arbitrary coordinate transformations. This flexibility, and a new text enhancement allowing control over intercharacter spacing, allows exact placement of graphics text. The AUTOPLOT/2700 and PAINTBRUSH/2700 applications take advantage of these enhanced graphics text capabilities, producing good screen previews and excellent hard-copy presentation material.

Hard Copy and Other Output

The HP 2700 has a variety of drivers for local hard copy. Supported hard-copy devices come in two basic types, vector and raster, and interface to the HP 2700 through one of three ports: RS-232-C, HP-IB, and RGB video.

Raster hard copy of the screen can be sent to HP printers or non-HP color printers. Menu options allow flexible control over picture size, orientation, and content. The printer driver can automatically generate eight-color, dithered 125-color, halftoned, or black-and-white pictures and output them to either RS-232-C or HP-IB devices. Users can also connect commercially available color cameras to the optional external video interface for raster image hard copy.

Vector hard copy is generated from the local vector lists and can be sent to an HP plotter over either RS-232-C or the HP-IB. The HP-GL commands generated drive a variety of HP plotters, from the A-size two-pen 7470A to the E-size eight-pen 7585A. The plotter driver takes advantage of the HP 2700's 32K×32K virtual space, and objects described in this large virtual space are plotted at high resolution, avoiding roundoff problems associated with the 512×390-pixel raster display. The AUTOPLOT/2700 application package can take advantage of the chart advance plot option of the HP 9872T Plotter to produce unattended multiple plots. This same capability is available to host applications, as are all RS-232-C and HP-IB printer and plotter hard-copy options.

Not only can a host describe a picture to the HP 2700, but also the HP 2700 can describe its current picture file, in escape sequence form, to a host. Thus, for the first time in an HP graphics terminal, locally generated graphics, perhaps as the result of a PAINTBRUSH/2700 or AUTOPLOT/2700 session, can be described to another device in vector form. This capability opens up new opportunities for local-versus-host partitions of graphics work.

Vector List Implementation

Each vector list consists of three parts. The first is the vector list header. It contains all of the primitive values that were set at the time the first primitive was stored. This includes items such as initial pen position, pen color, drawing modes, text slant, etc. Thus vector lists do not change values depending upon their order of drawing; the header block sets up an initial environment for a vector list draw that is identical to the environment in which the list was initially created. The second part of the vector list is the list itself, a serial stream of primitive graphics instructions. It is broken up into blocks that are easily managed, about 2K bytes per block. There are primitives that tell the vector list parser to jump to the next primitive block or to stop parsing. The third part of the vector list consists of text blocks. These are separated from the vector list to make text editing easier and independent of the other primitive attributes.

The first byte of each primitive in the vector list is an index number that is used to look up the corresponding

low-level handler to process this primitive. The second byte is the primary argument and is passed to the handler in a data register. Any other arguments in the list for this primitive are taken from the list by the handler. After the primitive has been executed, control returns back to the vector list parser, which then calls the next primitive handler.

The table that contains the address of the primitive handlers has room for new primitives to be added as needed. Also, the table is soft-coded so that a completely new set of primitive handling routines can be exchanged for the default set. This capability is used by the plotter driver.

Polygon Fill Implementation

The arbitrary polygonal (includes islands) area fill algorithm is conceptually simple:

- Break the polygon into edges
- Sort the edges, top to bottom, by topmost vertex
- Starting at the topmost edge, step down scan lines, generating all edges incrementally as they intersect the current scan line
- For each scan line, sort all edges in left-to-right order, then fill the scan segment for each odd-even pair starting at the head of the list
- Stop when the bottom-most vertex is reached.

The local transformations done by the HP 2700 (see below) transform the specified vector list area fill into a new polygon. To do this efficiently, the fill is done in multiple steps.

Step 1: Convert the standard polygon into a list of edges for the untransformed polygon. While doing this, accumulate the logical AND of the clip-code* for all vertices.

Step 2: If this is a nonrotated transformation, check the accumulated clip-code. If it is nonzero then no portion of the polygon will appear in the window so this polygon can be thrown away.

Step 3: Traverse the edge list transforming all vertices to display coordinates. Calculate the slope of the edges for use in determining scan line intersection during the actual fill (see Fig. 2a). The edge list must be preprocessed for the fill algorithm to work correctly. This is because the fill algorithm relies on the theory that any straight line going through an arbitrary polygon will intersect the polygon at an even number of points. Basically, what goes in must come back out. This theory has problems with edges that have one or both of their vertices on the line passing through the polygon. These are called special cases.

- Special case number 1: local maxima or minima (Fig. 2b). A fill line that starts on the outside of a polygon and encounters two lines that meet at a local maximum or minimum remains on the outside of the polygon and the fill generates a vector one pixel long for this intersection. A fill line that begins on the inside before encountering the maximum remains on the inside as it leaves that point. Fill lines can then be generated on either side of the point. The algorithm doesn't need any special work here.
- Special case number 2: knees (Fig. 2c). A horizontal line passing through the point of intersection of a knee passes

*Clip-code is a value returned from a routine that compares a point to the clip matrix. This routine is an implementation of the 2D clipping algorithm described in reference 2.

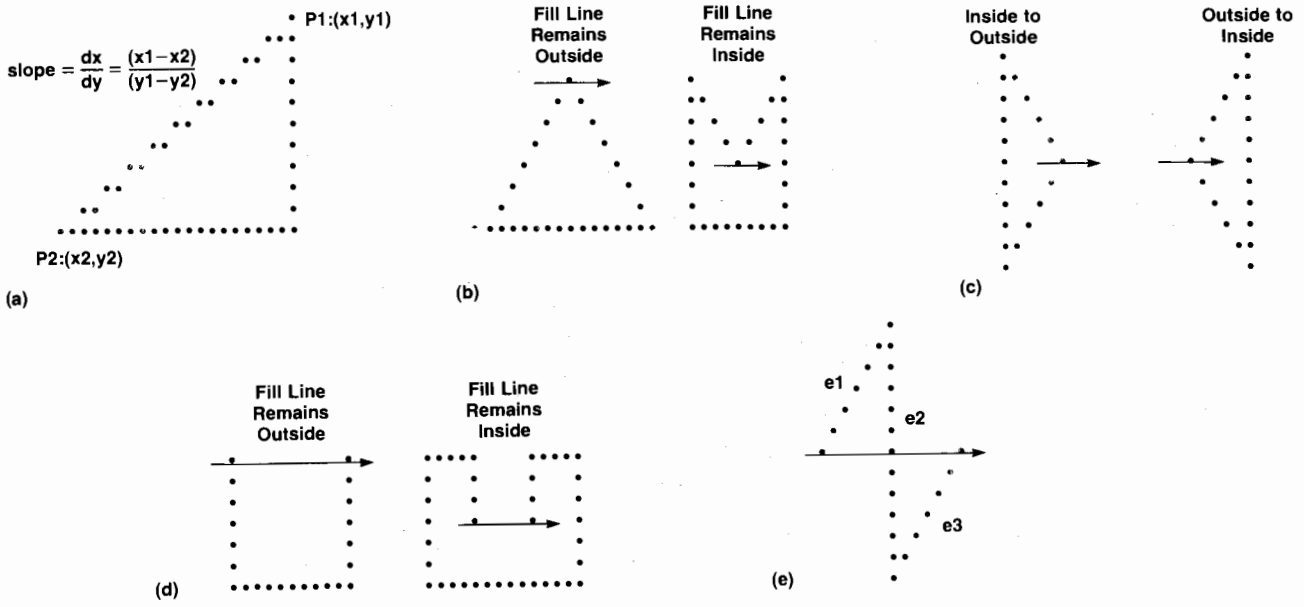


Fig. 2. *(a) Basic polygon area fill algorithm. To get from P1 to P2 in (y1−y2) downward steps requires (x1−x2)/(y1−y2) steps left per downward step. If P1 is (20,10) and P2 is (0,0) then two steps left must be taken for each step down. Special cases: (b) local maxima or minima, (c) knees, (d) horizontal edges. (e) If a horizontal edge is ignored, incomplete area fill may result. In this example, either e1 or e3 will have been shortened, depending on the order of the vertices. If this were not done, there would be three intersections on the scan line, which violates the primary assumption. Assume that e1 was shortened and does not intersect the horizontal line. If the horizontal line is ignored, the fill will start at the e2 intersection and miss the left part of that line.*

through two edges, but whether it is inside or outside of the polygon changes. During the preprocess pass through the edge list one of the edges is truncated so that only one edge appears at the intersection point.

- Special case number 3: horizontal edges (Fig. 2d). Horizontal edges require special care during the fill process and don't need to be worked on during this pass. However, they cannot be ignored since this may result in incomplete polygon fills (Fig. 2e). If the horizontal line in Fig. 2e is ignored, then half of the horizontal line will not be filled in. The half not filled in depends on the way the polygon was specified.

Two different algorithms are used for the top-to-bottom sort of the edge list. If fewer than eight edges exist in the polygon, a simple insertion sort is done. If more than eight edges exist and enough memory exists for a temporary key-pointer table, then a quick sort is used. The quick sort improves the speed of the fill substantially for large polygons.

A simple bubble sort is used to sort the edges from left to right on the same scan line. This algorithm turns out to be the most efficient for this application. In most cases there is no change in the order of edges from one scan line to the next. Bubble sort is fast at sorting a sorted list. If there are changes then they are either additions at the end of the list or they are crisscrosses of two adjacent edges. Bubble sort is fast in these cases, too. Horizontal lines that appear in the edge list are arranged in front of nonhorizontal lines if their first x values are the same.

Intersections of scan lines are calculated by incrementing the last intersection value by the slope stored in the edge information. To avoid accumulated roundoff the slope is calculated to a precision of 16 fractional bits. Because important tests are done at the ends of edges, the last intersection point uses the real value that was originally specified.

An edge list with nonhorizontal vectors is traversed in the standard way: fill the line between successive pairs of edges. Horizontal lines require special treatment. Horizontal lines that lie on or adjacent to each other are made into single horizontal lines. All nonhorizontal lines that intersect in the range of the horizontal line affect the parity of the first nonhorizontal line that appears after the horizontal lines are drawn.

Because of the need to support all of the drawing modes during area fill, this algorithm was designed to fill all the dots in the area fill only once. The boundary capability is provided by calculating the required dots on either side of each fill line. This results in some speed degradation when filling boundaried polygons. The ability to turn boundaries on and off is useful for filling 2D polygons generated from a 3D data base.

Cross-hatched polygonal fills of PAINTBRUSH and AUTOPLOT applications use the same area fill firmware in the terminal but have a special interface to prerotate the polygon before filling and then postrotating the resulting fill vectors.

Font and Label Implementation

A graphics label token in the vector list consists of the token code, the number of characters in the label, and a pointer to a block containing the information needed to display the label. This block consists of a header describing the label attributes and transformation, followed by the ASCII characters in the label. Fonts are described independently of the labels that reference them. There is a system table containing the addresses of each of the 16 character fonts. Also included in this table are flags indicating whether the font is a ROM-based system font or a user-defined font, and if the font has been redefined since it was last used. The first flag protects system fonts from redefinition and deletion and the second is used during label output, which is described below. The system table for a font is nil (zero address) if the font is undefined, and points to a 95-entry table of 32-bit character-definition offsets if the font is defined. (There are 95 graphic characters.) These character definition offsets are nil (zero) if there is no corresponding character definition, otherwise they are address offsets to be added to the address of the 95-entry table to point to the character definition. The first entry in this table is associated with the first printable ASCII character, a blank, and the last entry is associated with the tilde symbol. Each character definition consists of a four-byte header defining the lower left and upper right corners, followed by a vector description of the character in consecutive, one-byte, x and y components. All numbers are stored in excess 64 notation, with the most significant bit of the x byte reserved to indicate that the coordinate is a move instead of a draw. A zero x byte, whose value translates to −64 in excess 64 notation and is therefore never used as a coordinate value, indicates that the y byte is to be interpreted as a

Object Transformations Matrix:

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -Cx & -Cy & 1 \end{bmatrix} \times \begin{bmatrix} Sxcos & Sxsin & 0 \\ -Sysin & Sycos & 0 \\ Tx+Cx & Ty+Cy & 1 \end{bmatrix}$$

Viewing Matrix:

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -Wxmin & -Wymin & 1 \end{bmatrix} \times \begin{bmatrix} Zfx & 0 & 0 \\ 0 & Zfy & 0 \\ Vxmin & Vymin & 1 \end{bmatrix}$$

(Final) Combined Matrix:

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -Cx & -Cy & 1 \end{bmatrix} \times \begin{bmatrix} ZfxSxcos & ZfySxsin & 0 \\ -ZfxSysin & ZfySycos & 0 \\ Zfx(Tx+Cx-Wymin)+Vxmin & Zfy(Ty+Cy-Wymin)+Vymin & 1 \end{bmatrix}$$

Fig. 3. *Matrix arithmetic is used to calculate HP 2700 viewing transformations. The final transformation matrix results from combining the object transformations with the viewing matrix.*

command. Valid commands are "end character definition," "start area fill," and "end area fill."

When the label token is processed during a redraw operation, the length of the label is saved and the label block is accessed. In the label block is the font number associated with the label, and this is used to get an address for the primary and extension fonts. The extension font has a font number that is the higher of the paired fonts 1-2, 3-4, 5-6,... For example, font 9's extension font is 10, and font 10's extension font is 10. These fonts may have nil addresses, in which case default fonts are selected. Other setup operations before the output of the label include setting the line style and area fill pattern to solid, ensuring that the correct color is used, and locating the transformation matrix in the label block header. If the font flag shows that the font has changed since the transformation and placement of the label were saved in the label block header, the transformation matrix is recomputed, since a new justification factor or scale might be needed as a result of the newly defined characters.

Once setup has been done, all that remains is to step through the ASCII characters in the label block, index into the font or extension font to get the vector definition, apply the label transformation to the definition, and process the resulting coordinate as any other move or draw.

Transformations

The HP 2700's local object and viewing transformations offload many calculations from the host onto the terminal. The host not only has to do less arithmetic, but also doesn't have to do as much I/O to send the transformed vectors from the host to the terminal. The local transformations include rotate, translate, scale, and transformation center. This transformation center capability makes the specification of how the object is to be placed, stretched, or spun more intuitive. The viewing transformations are applied along with the object transformations to provide a complete set of attributes for viewing any portion of the picture in a convenient viewing area of the screen.

Matrix arithmetic is used to calculate these transformations. The result of the object transformations combined with the viewing matrix is shown in Fig. 3.

The first subtraction of the transformation center is not included in the total matrix. Its inclusion would require a larger dynamic range of the individual multipliers of the new matrix than could be provided. The final work done on each coordinate pair consists of the following two computations:

$$\begin{aligned} xnew = {} & (vectorx - centerx) \times m11 \\ & + (vectory - centery) \times m12 + m13 \end{aligned}$$

$$\begin{aligned} ynew = {} & (vectorx - centerx) \times m21 \\ & + (vectory - centery) \times m22 + m23 \end{aligned}$$

- vectorx/y is a 15-bit value.
- centerx/y is a 15-bit value.
- The subtraction of these two quantities yields a 16-bit value.
- m11, m12, m21, and m22 are 32-bit floating-point values.
- m13 and m23 are 48-bit fixed-point values, 32 bits of integer and 16 bits of fraction.
- The 16×32 bit multiplication yields a 48-bit product. All partial sums are maintained to 48 bits of precision.

The above equations require eight 16-bit multiplies, four 32-bit adds, four 16-bit adds, and eight shifts of 32-bit data. A 68000 microprocessor may be powerful, but it isn't powerful enough to grind through these vectors at a very high rate. Using average instruction timing on the 8-MHz 68000 results in 100 microseconds for a basic transformation. The need to check for overflow when transforming unclipped vectors adds in an extra 15 microseconds. Some other overhead, such as bit shifting, moving data to temporary registers, etc., brings the total time for a complete tranformation to about 150 microseconds.

The main loop that calls the transformation routine is another 150 μs. It takes care of all of the drawing modes, line types, and overflow conditions. The total time is about 300 μs or a little more than 3000 vectors per second.

Since many vectors do not require all the sophisticated calculations that are required for these transformations, the process is optimized to do only what is required. The full transformation process is a considerable amount of code that can be broken down into some modular pieces: a piece that subtracts the transformation center from the endpoints, a piece that multiplies the endpoint by the upper value of the multiplier, one that takes care of the lower 16 bits of the multiplier, a piece to multiply by 1 or by 0, and pieces to shift the results the correct amount of times. In the simplest cases, most of these pieces are not required. If the transformation center is 0 why subtract it from the endpoint? If the multiplier is 1 why do the multiply? With these in mind, the optimization is done by dynamically examining the final transformation matrix, selecting the appropriate code from a table of routines, and placing that code in high-speed RAM. This results in the building of a routine that is highly optimized for the particular transformation the vectors will be going through.

A particular transformation is constant for the processing of an entire object and vector list. For each object, the HP 2700 computes a new transformation matrix, which includes the object as well as the view transformations. Then it builds the required transformation routine. This technique nearly doubles the speed for simple transformation matrices. There is also some improvement in speed for vectors going through complex transformations because the code can use immediate data instead of variables that require extra memory accesses to retrieve.

References

1. R.A. Jewett and R.W. Fredrickson, "The System 45C User's Firmware Interface," Hewlett-Packard Journal, Vol. 31, no. 12, December 1980.

2. W.M. Neumann and R.F. Sproull, "Principles of Interactive Computer Graphics," 2nd Edition, McGraw-Hill, 1979.

Logic Design for a Graphics Subsystem

by Craig W. Diserens, Curtis L. Dowdy, and William R. Taylor

A MAJOR CONTRIBUTION of the HP 2700 to graphics users is its response time to draw and redraw complex pictures on the CRT. Under best-case conditions, redraws from the local vector list can approach a rate of 40,000 vectors per second. Fast response time is achieved by distributing raster graphics drawing tasks to hardware optimized for this purpose.

The hardware pipeline that presents graphics information to the HP 2700 color monitor is shown in Fig. 1. This pipeline consists of three plug-in hardware modules known as the graphics controller, the graphics image memory, and the color mapper. An additional path from the graphics image memory to an external raster device such as a hard-copy camera or another display monitor is provided by a fourth optional plug-in module called the external video interface.

Graphics Controller

The graphics controller offloads vector drawing tasks from the system MPU. The MPU passes the (x,y) coordinates for the vector endpoints to the graphics controller, which then calculates which dots on the display to use to achieve the best approximation of a straight line. This allows the MPU to spend more of its time doing higher-level graphics functions such as scaling, rotating, and translating. The graphics controller also controls the graphics image memory and performs the function of scanning the image memory to output pixel data to the color mapper.

A block diagram of the graphics controller is included in Fig. 2. The major functional blocks are the processor-independent bus (PIB) interface, the vector processor, the raster-scan logic, and memory control.

The vector processor is the highest-level functional block within the graphics controller. It is a dedicated processor designed specifically for raster graphics operations. Its instruction set offers the ability to draw patterned and nonpatterned color vectors, store and recall vector parameters, fill blocks of memory with a color, write and read individual pixels, write and read segments of raster planes, and test processor functionality.

As can be seen in its block diagram, Fig. 3, the vector processor is implemented with hardware from the 2901 bit-slice family. The processor architecture is defined by the microcoded control program found in the microcontrol store. The control program consists of 512 32-bit control words, grouped into microroutines corresponding to the graphics controller command set. Each individual bit of a control word is linked directly to a controlling function on one of the major logic devices within the processor section.

A microroutine is invoked by writing an address to the command register. This register is connected directly to five PIB address lines and appears to the MPU as a set of 32 locations within the PIB address space. The address written to the command register is then passed by the 2911A Sequencer upon completion of the current microroutine to the microcontrol store, where a jump table maps the command address to a particular microroutine.

The control and status pipeline registers shown in Fig. 3 allow controlling operations and processing operations to occur simultaneously, so that a control instruction may be processed while the next instruction is being fetched. This allows control instructions to be executed in continuous 8.8-MHz cycles.

In the processor section, the 12-bit 2901A Arithmetic and Logic Unit is assisted in vector drawing functions by a set of external registers and counters. Instead of placing the data most used when drawing vectors in the registers internal to the 2901A, where multiple cycles would have to be executed to make this data available, vector data is placed in external devices. This speeds vector drawing so that in the best case, a pixel can be drawn with each image memory cycle (2.2 MHz).

The raster-scan address logic, shown in Fig. 2, is responsible for keeping track of the current position of the video signal as it sweeps across the CRT. It then addresses the image memory once every 16 pixels of video display time

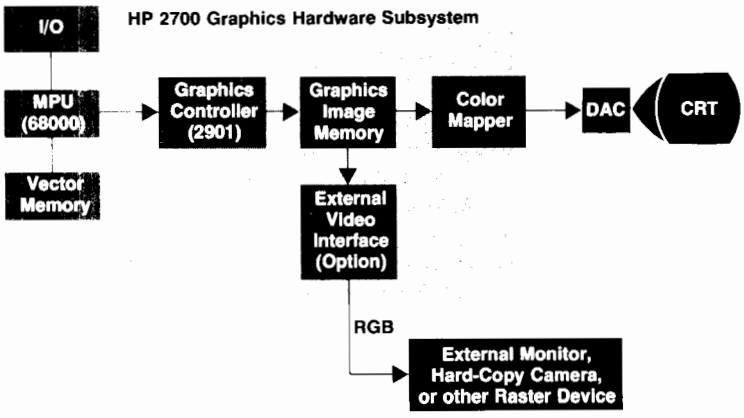


Fig. 1. *Graphics hardware architecture.*

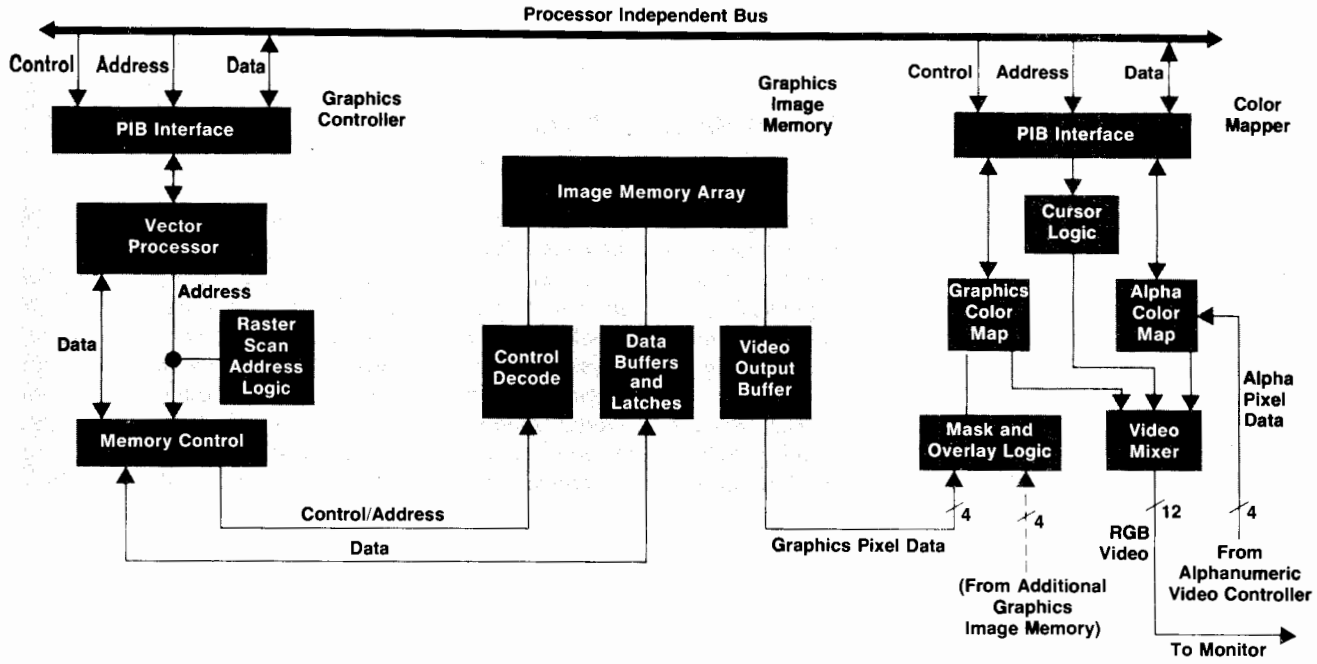


Fig. 2. *Graphics hardware modules and functional blocks.*

and instructs the image memory to load the next 16 pixels to be displayed into the video output buffers of the graphics memory.

Both the vector processor and the raster-scan address logic generate image memory addresses in logical form (i.e., in logical two-dimensional space with (x,y) = (0,0) located at the lower left-hand corner of the display). It is the job of the memory control functional block to map logical image memory addresses into physical memory addresses and generate the proper timing and control signals for the graphics image memory.

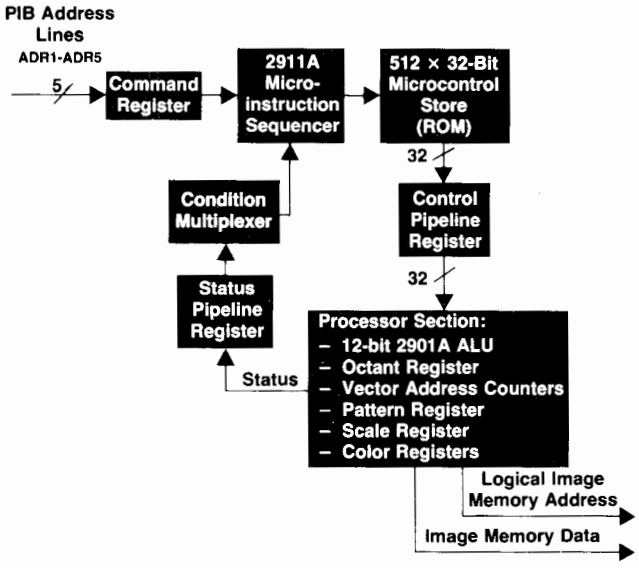


Fig. 3. *Vector processor block diagram. Bit-slice hardware from the 2901 family is used.*

Graphics Image Memory

The graphics image memory, whose functional block diagram also appears in Fig. 2, serves as the display buffer for the graphics image. Address locations are provided for a 512×512-pixel display space with four bits per pixel. (However, only 512 by 390 pixels are displayed on the HP 2700 CRT.) As an option, a second graphics image memory plug-in module may be added to the system to provide double-buffering capabilities and overlays. The organization of this hardware module provides flexibility in reading and writing the raster image in the form of multiple addressing modes. The sequencing of row and column addresses is optimized to allow raster scanning to refresh the dynamic memory, thus allowing more bandwidth for vector plotting while the display is being refreshed.

The image memory array is composed of sixty-four 16K×1 dynamic RAMs, which are organized so that the array can be addressed in one of four ways. The various addressing modes can best be understood by first looking at the logical organization of the array, shown in Fig. 4. The cross section at the top of the array represents a 64-bit block, one bit from each RAM, for any of the 16K addresses. Four boundaries define the four types of addressing modes. A pixel is four bits deep into the array, and defines the color of a dot on the display. A raster plane is a vertical slice composed of one bit from each pixel in the array, and a plane segment is a four-bit nibble from one memory plane. A word is 16 bits making up four adjacent pixels. A video word is a 64-bit block composed of 16 adjacent pixels. The array can be accessed by individual pixel, by plane segment, by word, or by video word.

Pixel access is used by the vector processor when drawing vectors or when in pixel read or write mode. Plane segment access is used for raster dumping between the image memory and mass storage and for supporting

monochrome hard-copy output. The external video interface uses word access to read four pixels at a time for external raster output. Output to the color mapper is done by video word so that the memory bandwidth is not consumed by the action of raster scanning.

Color Mapper

The color mapper board provides graphics color mapping and image manipulation, alphanumeric color mapping, and a hardware graphics cursor for the HP 2700. Color mapping greatly expands the user's choice of colors without increasing the size and cost of the graphics image memory. In a color-mapped graphics system, the pixel data in the image memory does not define a color, but rather a code used to select a color from a table.

As an example of color mapping, consider the following table function:

Pixel Value	Primary Color			Corresponding Color
	Red	Green	Blue	
1	3	3	3	gray
2	4	4	0	yellow
3	0	0	4	blue

Given a pixel value of 1 to 3, the result of the table function is a color triple ⟨Red, Green, Blue⟩ determined by the pixel value. A pixel value of 3 selects a color triple corresponding to a full bright blue. If each primary color (red, green, blue) can have five intensity levels, there are $5\times5\times5=125$ different combinations or colors. However, only three values have to be stored in the image memory.

If the map table is changed, then the color assignments and the corresponding visual image are modified without changing the image memory. A second map,

Pixel Value	Primary Color			Corresponding Color
	Red	Green	Blue	
1	3	3	3	gray
2	4	4	0	yellow
3	4	0	0	red

changes everything that was blue with the first map into red. Thus, in addition to presenting the user with a large spectrum of possible colors, color mapping lends itself to flexible and interactive changes in color.

The HP 2700 implementation has 16 values per graphics pixel (4 bits), and 16 intensity levels for each primary color. Consequently, the user has $16\times16\times16=4096$ color choices, with 16 colors available at a time. A high-speed 16-word-by-12-bit memory operating at the graphics dot rate is used as the lookup table. Changes of the map memory are synchronized with display blanking (off) intervals to avoid visual flashes.

Alphanumeric color mapping follows the same basic notions, but is implemented slightly differently. Each character is assigned a color code whose value is used to select the color definition for that character. The character matrix dots that form a character, the foreground, may be assigned a color using a ⟨red, green, blue⟩ representation. Similarly, the background dots in a character cell are independently assigned a color. There are 16 available foreground/background pairs, with 64 color choices.

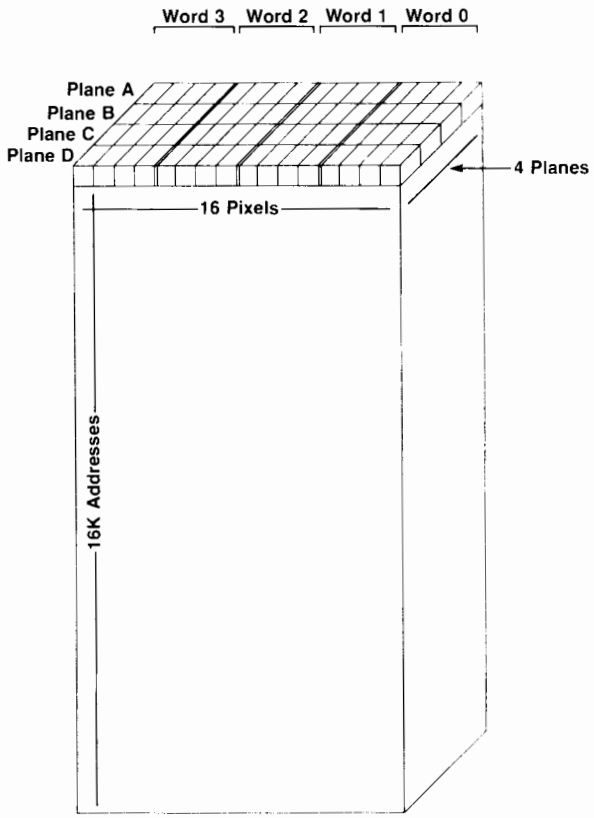


Fig. 4. *Logical organization of the image memory array.*

The HP 2700 can manipulate its two graphics images in several ways. One way is double buffering, described in the preceding article. This is effected using the color mapper mask/overlay logic.

In overlay mode, one image can be superimposed on the other and both images effectively displayed simultaneously. This allows dynamic objects in one memory to be altered without redrawing the entire image. A potential application might place a fixed pattern such as a menu or map into one memory and move objects in the other. Overlay is implemented by displaying the background image whenever the foreground pixels have a zero value.

The high speed of most of the color mapper posed some interesting testing challenges. Test pattern registers are placed at the video and control inputs, and the PIB interface allows processors to read the final video outputs to the CRT amplifiers. For functional testing, the maps are loaded with predefined values, the board inputs varied, and the final video output compared with the expected result. This test method has proved very effective and economical in both software and hardware (five additional ICs out of 81 total).

External Video Interface

The external video interface (EVI) permits video devices such as color display monitors and color graphics cameras to be connected to the HP 2700 Terminal. The EVI is a graphics-only interface; alphanumeric characters are not displayed. In contrast with the noninterlaced internal dis-

play of the HP 2700, the EVI generates an interlaced output suitable for lower-bandwidth devices. Because of its interlaced display, the EVI cannot simply tap the data stream going to the internal display and convert it to appropriate analog levels. It is an integral part of the graphics subsystem, and pixel data is accessed directly from graphics memory, contending with the graphics controller for memory cycles. A state machine arbitrates the competition for memory cycles, initiating an access to graphics memory when the EVI needs data for its display and the memory cycle is not being used by the graphics controller for internal display refresh. Pixel data is buffered on the EVI in two FIFO (first in, first out) memory arrays which are accessed asynchronously as needed by the EVI display refresh logic. Data is passed to the color mapper on the EVI, which duplicates the functions of the main graphics color mapper. Digital signals are then combined with video timing signals and converted to an RGB video output conforming to a subset of the Electronics Industry Association (EIA) RS-170 standard.

A High-Resolution Color Monitor

by Mark Hanlon, Geoffrey G. Moyer, and Paul G. Winninghoff

TO GIVE THE USER a wide choice of colors for the 16 pens available in the HP 2700, the color monitor is designed to produce 4096 pure colors. A delta-gun CRT is used to display vivid, well converged colors with sharp focus, very high contrast, and almost no reflections. A high-speed digital-to-analog converter (DAC) converts the digital information from the color mapper to the fast-rise-time, large-voltage-swing signals needed to drive the cathodes of the CRT. Raster-scan deflection is used. Electronic signals correct the nonlinearities in both horizontal and vertical directions, so that circles look like circles and are the same size when they are moved from one portion of the display to another. A new approach to convergence circuitry eliminates the headaches of conventional television-type monitor alignment. A convenient, simple-to-use control panel aids the user in the quick convergence procedure.

To put 4096 different colors on the display, very sharp focus and precise convergence are a must. These constraints can only be met with the use of a delta-gun CRT, so called because of the placement of an electron gun on each vertex of a triangle. Each of the three guns is aimed at a particular color phosphor dot on the screen, either red, green, or blue. The dots are in triangular groups of three, called triads, that are spaced less than a third of a millimeter apart. These dots are so small that at a normal viewing

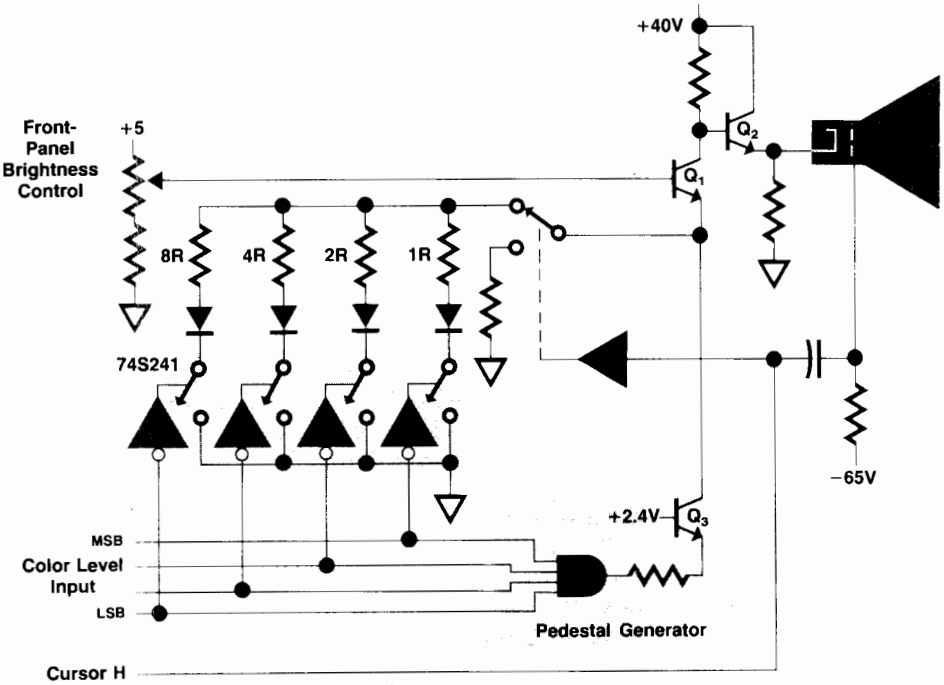


Fig. 1. *The video drive circuitry provides 16 color levels to each gun of the CRT and requires four bits of information from the color mapper. This diagram shows the video driver for one color.*

distance the eye averages their effect and sees a pure color.

The video drive circuitry provides 16 color levels to each gun of the CRT, so four bits of information per color are sent from the color mapper (see Fig. 1). The cursor is extra-bright white and information for it is sent on a separate line. The DACs provide a set of voltages to each cathode to produce colors that appear to be in uniform brightness steps. Spot shape is determined in part by the rise and fall times of the cathodes. For a crisp display, these times must be kept to a small percentage of the dot time.

A TTL line driver provides a simple, economical solution to implement three high-speed, well matched DACs. Incoming data is coded negative true, so for the brightest level, all four emitter resistors are switched to ground, each through a single section of an octal driver. The user has access to a front-panel control that varies the bias on Q1, thus varying the gain of the DACs. This changes the amplitude of each of the steps in brightness level and is thus a contrast control. Separate internal adjustments set the cutoff levels of the three guns, so the user can reduce the brightness level of full-bright colors without making the darker colors too dim to see. Each of the DACs generates a large step in voltage (a pedestal) so that the black level (all inputs high) is extra-black, making only a single internal adjustment necessary to control both brightness and spot cutoff.

The video driver is in a rather precarious position in the monitor system. It sits between extensive digital hardware, most of which runs at TTL levels, and the CRT, in which most of the potentials are measured in kilovolts. It is likely that the anode voltage (+20 kV) will arc over to the lower-potential cathodes sooner or later in the life of any CRT. Applying +20 kV to the output of a TTL gate, even for a few microseconds, can severely limit the useful life of the gate!

During a tube arc-over, the anode capacitor formed by the inside anode coating on the CRT and the conductive (aquadag) ground on the outside of the CRT tries to discharge through the low-voltage grids at the base of the neck. The rise time of the arc is on the order of a few tenths of a microsecond, while the peak voltage can be the full 20 kV. A very low-impedence, separate ground path is provided to the aquadag. To keep the return path of the arc-over current as short as possible, spark gaps to the anode return ground are provided in the CRT socket. Still, more protection is needed, since gaps can only reduce the voltage to a few kilovolts. Each of the grids and cathodes has its own protection scheme because each operates at a different voltage and frequency. Careful attention is paid to the placement of the arc-over devices relative to the CRT pin and the anode return ground. Current-limiting resistors are also used to limit the peak energy any circuit will be asked to absorb in an arc-over.

Easy Convergence

For a properly designed, uniform deflection field, the convergence error of a CRT is a geometry error caused by the difference between the radius of curvature of the CRT faceplate and the distance from the deflection plane to the

EMI Entanglements

Computers and their peripheral devices, like many common household appliances, emit electromagnetic interference (EMI). Ordinarily these emissions pass around us, unnoticed until the television or radio show we are listening to is interrupted by noisy reception. Current government standards aimed at limiting EMI, as well as a desire to reduce unwanted interactions between computers, have prompted the following design goal for the HP 2700: EMI suppression commensurate with the many requirements of a powerful desktop peripheral.

EMI is generated in computer products by the action of digital gates. When a gate switches from a logical high level to a low level repeatedly, the possibility of EMI is high. Fourier analysis of logic signals shows that emission is possible at all frequency multiples of the pulse repetition rate. The level of disturbance diminishes at higher frequencies, as determined by the repetition rate and rise time of the logic signal. The duty cycle of the signal modifies this diminishing trend with a sinusoidal function. Fast gates typically use appreciable currents to switch quickly. These rapidly changing currents must have a low-impedance return path nearby or EMI will occur. It is interesting to note that, given a choice of paths to follow, return currents will choose the one that produces the smallest emission. Even so, it is very difficult for a design to provide "good" paths for every one of the multitude of traces, so some EMI occurs.

A thin shield, such as conductive paint, is an attractive method of EMI suppression, but is only effective in stopping EMI from high-impedance (low-current, high-voltage) sources. Thick (metal) shielding is effective against low-frequency emissions from low-impedance circuits but is an unattractive solution for a desktop unit. The HP 2700 takes a third approach to EMI suppression, incorporating ground-plane printed circuit boards to form a return path network that prevents EMI. A ground-plane board is produced by laminating a copper sheet between the ordinary layers of circuit traces. The early commitment to this mode of manufacture alleviated the need for a heavy metal enclosure.

To test HP 2700 compliance with emission limits in all of the many system configurations, different combinations of devices were placed at one end of a 50-yard metal surface and allowed to perform worst-case communications. The measuring receiver was at the other end of the surface. At each frequency, the disturbance level was measured, transcribed, and examined to ascertain possible culprits. An appropriate noise reduction technique was then applied.[1] The HP 2700 uses grounding, isolation, and shielding techniques to solve radiated EMI problems. Filtering and source suppression methods are used to reduce conducted EMI on the power cord.

The HP 2700 EMI solution permits many peripherals to be attached simultaneously without producing significant EMI while allowing a plastic package to surround a generous amount of processing power.

-Geoff Moyer

Reference

1. H. Ott, *Noise Reduction Techniques in Electronic Systems*, Wiley, 1976.

center of the faceplate. This difference can be expressed as a power series that contains only even-order terms, the high-order terms becoming very small very quickly. The first two terms of this error expressed as a function of the distance from the center of the faceplate are:

$$E = k_1 + k_2D^2 \quad (1)$$

where k_1 and k_2 are constants, D is the distance from the center of the faceplate, and E is the error term in one direction.

The faceplate is two-dimensional, so the total error is a product of the error in the X direction and the error in the Y direction:

$$E_c = k_a + k_by^2 + k_cx^2 + k_dx^2y^2 \quad (2)$$

where: k_a = magnitude of the error of the entire screen
k_b = magnitude of the error on the top and bottom
k_c = magnitude of the error on the left and right

HP 2700 Power Supply

The major design challenge of the HP 2700 power supply was packaging a large amount of power into a comparatively small space. The 1W/in³ power density of the 450-watt output supply is over twice that of previous terminal designs, yet does not require high levels of forced air cooling. Effective system protection and diagnostic capabilities are additional features of the supply.

Because of limited space and cooling, power conversion effeciency (power output ÷ power input) is extremely important in this design. The overall supply architecture, shown in Fig. 1, allows most of the power output to be converted from ac to dc in only one active conversion step. High-frequency, high-voltage switching circuits are used to convert the incoming ac line voltage into dc output voltages. For +5V and +44V, the conversion process is

110-220V 50/60 Hz ac → ±150V dc → ±150V 25 kHz square wave ac → dc out

The remaining output voltages are generated with secondary switching and linear regulators. Using this supply organization, the overall efficiency is a high 78%. When compared to a typical 70% efficient supply, the HP 2700 supply eliminates 65 watts of heat dissipation.

Protection and Diagnostic Capability

Extensive voltage and current protection circuits are included because of the high power levels sustained by the supply. The +5V, 40A output is capable of welding metal to the outputs. Current limiting on all outputs and the high-voltage primary effectively protects the supply from load (system) component failures and short circuits. Protection of the system is accomplished by overvoltage and undervoltage monitoring of all outputs. If any voltage exceeds the nominal voltage +10%, or is less than the nominal voltage −15%, the supply shuts down all outputs and remains off until the ac power is turned off and on again. The overvoltage shutdown delay is 100 μs, while the undervoltage delay is 0.7 second.

As a direct outgrowth of the voltage protection circuit implementation, diagnostic indicators and special test points are available for servicing and repair. Two status lights, a green Up and red Down, indicate that all voltages are either in tolerance or have caused a shutdown. Consequently, it is not necessary to measure output voltages to know the supply is working correctly. The cause of a shutdown can be deduced from the lights during servicing.

The supply is partitioned onto two printed circuit boards. The control board contains only low-level circuitry and is well suited to automated manufacturing techniques, while the main board uses numerous large components and includes ac line voltages. The +5V, 40A currents are run through copper bus bars and through four printed circuit board layers.

-Craig Diserens

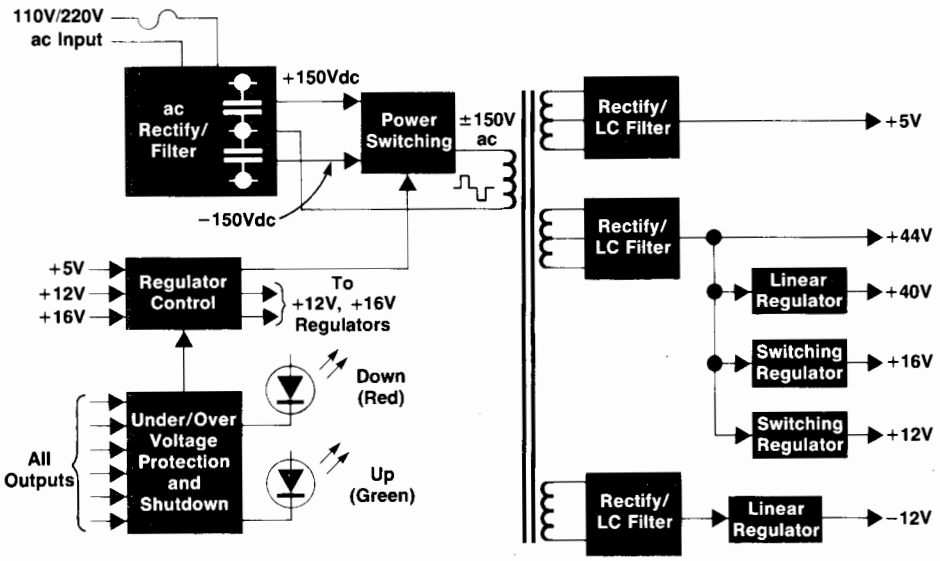


Fig. 1. *The HP 2700 power supply converts most of the output power from ac to dc in one switching step. Efficiency is 78%.*

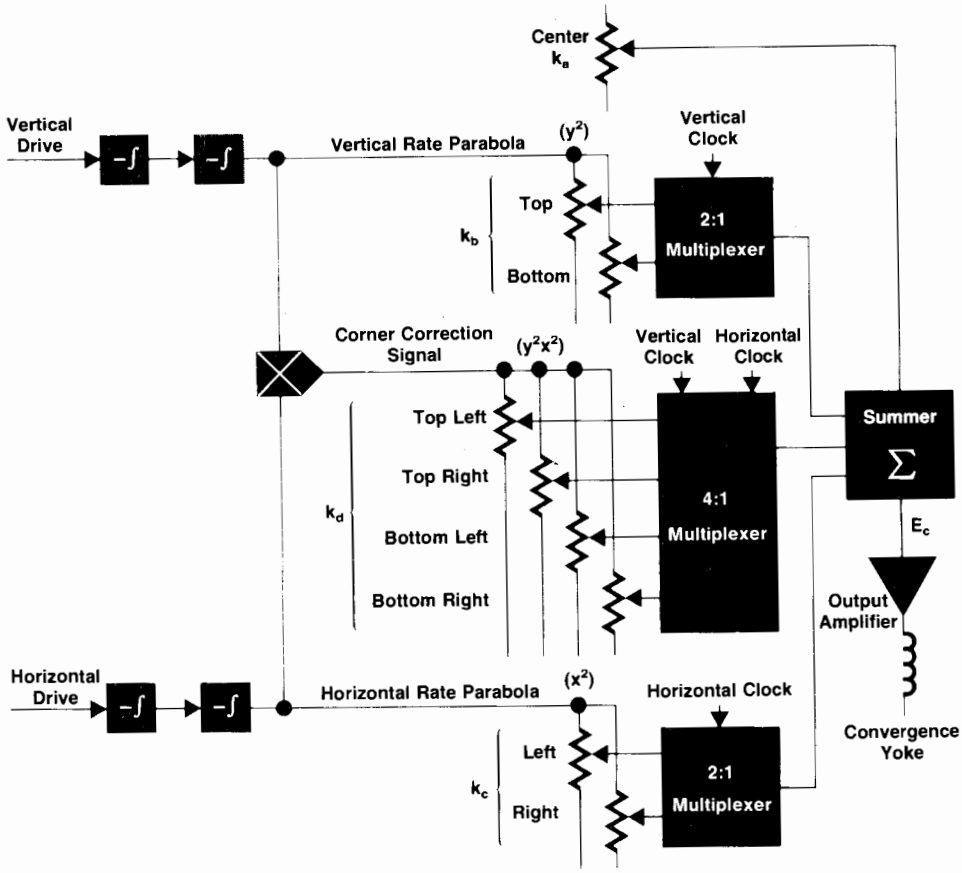


Fig. 2. *Convergence circuit allows the user to adjust each coefficient of equation 3 (see text).*

k_d = magnitude of the error in the corners
x = horizontal distance from the center of the faceplate
y = vertical distance from the center of the faceplate
E_c = total convergence error.

In the ideal case, where the guns, deflection yoke, faceplate, and video information are all perfectly aligned, the k_a (dc) term vanishes. In a real system, the dc term is present, and indeed, the error on the left is not necessarily the same as that on the right, the top and bottom errors are not the same, and the corners each have their own errors. So, the total error function becomes:

$$E_c = k_{dc} + (k_t + k_b)y^2 + (k_l + k_r)x^2 + (k_{tl} + k_{bl} + k_{tr} + k_{br})y^2x^2 \quad (3)$$

where t=top, b=bottom, l=left, and r=right.

The y^2 term is a vertical rate parabola, the x^2 term is a horizontal rate parabola, and the y^2x^2 term is a product of the two parabolas. In the HP 2700, convergence correction is implemented by allowing the user to determine the values of each of the k terms by adjusting potentiometers that are connected to the appropiate correction waveforms (Fig. 2). The k_{dc} term is a dc voltage level and corrects errors at screen center. Each of the other correction signals is set to zero at screen center so that only the k_{dc} term affects the screen center. The horizontal rate parabola is already zero at the vertical centerline. The same holds for the vertical parabola along the horizontal centerline and the cross product signal along both centerlines. The voltages of all of the pots for one color are multiplexed at the horizontal and vertical rates and then summed. There is a pot for each of the nine constants in equation 3 for each of the three colors, red, green, and blue, for a total of twenty-seven pots. The pots are arranged in groups of three, one for each color, and in a 3×3 grid that corresponds with the position of the screen that each pot will adjust. When adjusted in order, the controls are not dependent on each other because the signals are zero on the centerlines. Therefore, obtaining optimal convergence is easy.

Acknowledgments

Special thanks to Phil Garcia for his engineering to improve the reliability of the monitor system. The authors would also like to thank Art Garcia, Tim Hastings, Steve Hopper and Mike Maloney for their constant help maintaining the development systems.

The Graphics Workstation as an Extensible Computer Terminal

by Edward Tang, Otakar Blazek, Thomas K. Landgraf, Paula H. Ng, and Stephen P. Pacheco

IN ITS ROLE AS A COMPUTER TERMINAL, one of the key design objectives for the HP 2700 Color Graphics Workstation was extensibility. This means it should be easy to incorporate new technologies as they become available or to coincide with the changing needs of users.

In fulfilling the role of a terminal, the HP 2700 must provide certain capabilities: an alphanumeric display, keyboard input, data communications, and local device control. These functions are provided by the intelligent I/O controllers. The actions of the I/O controllers are coordinated by the main processor. The architecture, interface, and software of the I/O controllers and main processor are designed not only for extensibility, but also for compatibility with existing HP terminals.

Main Processor

The main processor consists of the main processing unit (MPU) and the universal ROM and universal RAM boards. The MPU is a single-board computer featuring an MC68000 microprocessor, 32K bytes of RAM, and 96K bytes of ROM. With the universal RAM and ROM boards, memory can be expanded to 8M bytes. Also included on the MPU board is a serial datacom port used for error diagnosis and in the computer-aided test and tracking system used in HP 2700 production (see article, page 25). The MPU controls all of the I/O controllers, interprets all alphanumeric and graphics commands, and coordinates the processing required to perform each command.

The universal RAM board can be configured for 64K, 128K, or 256K bytes of additional RAM, and the ROM board can provide an additional 256K, 512K, or 1M bytes of ROM. The amount of additional memory is determined by the number and type of memory chips installed on the boards. This flexibility improves manufacturing productivity. Since only two types of boards have to be manufactured, only two sets of manufacturing and test tooling are required to yield six different board assemblies.

The memory boards can be used in two ways by the MPU: as local memory accessible only by the MPU, or as shared memory accessible by any intelligent board. The type of connection to the MPU determines the scope of the memory board. Memory boards connected to the MPU via the topplane bus are local memories, while those connected via the backplane only are shared memories. Topplane memory is used where high-speed memory access is required. The terminal firmware and application RAM are accessed via the topplane. Backplane memory is used to communicate with the I/O controllers and for storing noncritical data.

MPU Software

The main processor software coordinates all the activities of the HP 2700. It provides all of the advanced graphics and alphanumeric features in a manner compatible with existing HP terminals. This allows existing HP software (e.g., VPLUS/3000 and DSG/3000) to operate on the HP 2700 with no changes. Thus, a user can make immediate use of the HP 2700 with existing applications. Gradually, modifications may be made to take advantage of the HP 2700's features. For example, color enhancements could be added to a form to make it easier to distinguish the various fields.

As in existing HP terminals, configuration is performed via menus. However, to accommodate the modularity and expandibility of the HP 2700, the underlying structure is different.

In previous HP terminals, the number and types of configurable devices are fixed. In the HP 2700, the number and

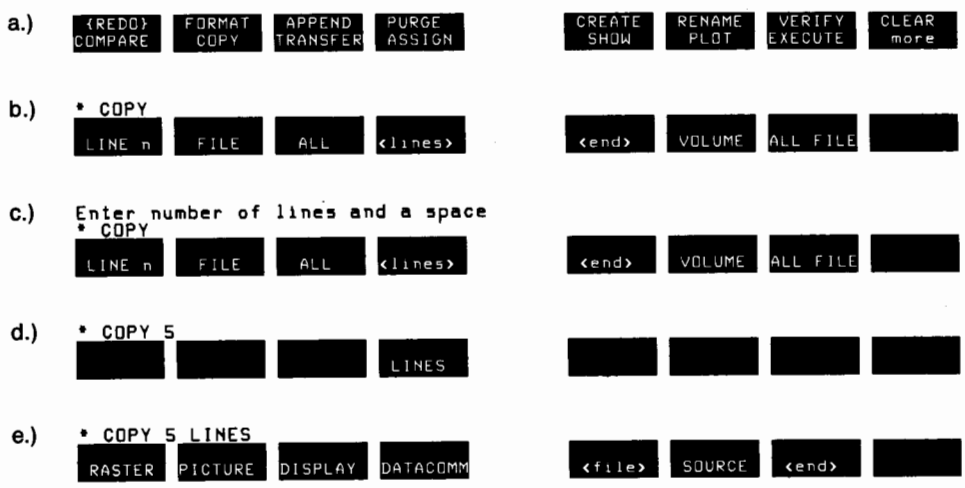


Fig. 1. *An example showing the operation of the command window. (a) The top level of softkeys. (b) The user has pressed COPY to enter the keyword into the command window and display the next level of softkeys. (c) The user now presses ⟨lines⟩ and a message is displayed to tell the user what is expected next. (d) After entering the numeric parameter, LINES is the only keyword allowed. (e) The user can continue adding to the command by specifying source and destination files or end the command by pressing* **RETURN**, *in which case the default source file is copied to the default destination file.*

types of devices are not limited. Also, the configuration needs of future devices are not predictable. With this in mind, a general-purpose, extensible configuration subsystem was designed. The configuration data consists of a menu descriptor and a menu data block. The menu descriptor defines the screen layout, the type of each option field, and the allowable values for each field. The menu data block contains the current configuration values. The configuration data is stored in the I/O controllers, so when controllers change, their configuration menus can change accordingly.

When the user elects to configure a device, the configuration data is retrieved from the I/O controller that controls the device. The configuration subsystem displays the menu as specified with the current values from the menu data block. User input is accepted and edited according to the menu descriptor. When the configuration is activated by pressing either the TEMP SAVE or SAVE CONFIG softkeys, the updated menu data is sent to the I/O controller. If SAVE CONFIG is pressed, then the values are stored in nonvolatile RAM to be saved when the HP 2700 is turned off. This RAM is dynamically allocated and can grow and shrink as devices are added and deleted. Nonvolatile RAM is organized by PIB board identification number so that a board's configuration is maintained regardless of which PIB slot it is in.

Command Window

A user can access devices by entering commands into the HP 2700's DEVICE CONTROL command window as on HP 2647 terminals. Users can transfer data to and from internal devices, such as the flexible discs, alphanumeric display, graphics raster display, or graphics picture file with a single COPY command. Hard-copy output can be obtained on printers, plotters, or other peripherals connected to the terminal via the RS-232-C/V.24 or HP-IB (IEEE 488) interfaces.

The command window is also designed to minimize the number of keystrokes required to enter a command and to assist the user in composing syntactically correct commands without needing to refer to a manual. The resulting user interface is similar to that of the HP 64000 Logic Development System.[1,2] Softkey labels assist the user in composing the commands by displaying the next allowable keywords or parameters for the command line. The softkey labels are updated on the screen whenever a new word is entered into the command line. A message window is displayed above the command window whenever a help or error message needs to be displayed for the user (see Fig. 1).

The command parser has been made extensible by taking advantage of the ROM routine replacement capability described in the box on page 8. A dummy routine is called whenever a syntax error is encountered in parsing a command. The dummy routine simply returns the syntax error status, which is reported to the user. This dummy routine can be replaced with a routine that calls the command parser again with the same command, but with a different syntax table so that new commands are accepted by the parser. The status returned by this new routine would be the status of the command just performed, or a syntax error status if the command is still incorrect.

Disc Utility

The ability to expand the command set is used in implementing a disc patcher utility, which is included as part of the terminal's service kit for service engineers. This utility is designed to patch discs that may have lost data through long use or exposure. When the utility is loaded, the terminal recognizes new commands that allow the user to access individual disc sectors to patch directories and bad disc files.

The terminal's base command set allows users to access only the records of the disc files, not any of the directory or file information on the disc. Included in the disc utility command set is the MODIFY command, which allows a user to use the terminal's screen editing capability to alter one sector of data. Also included are more general user commands, including an extended COMPARE command which reports all of the miscompares between two files, instead of stopping at the first miscompare (which is done in the COMPARE command in the base command set). Being able to include this utility outside of the ROM code uses less ROM space and limits access to potentially harmful commands for users unfamilar with the disc format.

Intelligent Subsystems

Although the MPU dictates the functions to be performed, the intelligent I/O controllers determine how these functions act on the devices they control. The controllers also provide the input from the user or host to direct the MPU's operation. Besides offloading the device-dependent details of various I/O functions, the intelligent subsystems enhance the modularity and extensibility of the HP 2700. Since the MPU does not have to deal with the device-dependent functions, I/O controllers can be enhanced or replaced with minimal impact on the main terminal code. There are currently five intelligent I/O controllers:

- Alphanumeric Video Controller (AVC)
- Keyboard/Datacom Controller (KBDC)
- Tablet Interface
- Shared Peripheral Controller (HP-IB)
- Minifloppy Controller (MFC).

All of these controllers have a similar hardware kernel, consisting of a Z80A microprocessor, on-board RAM, on-

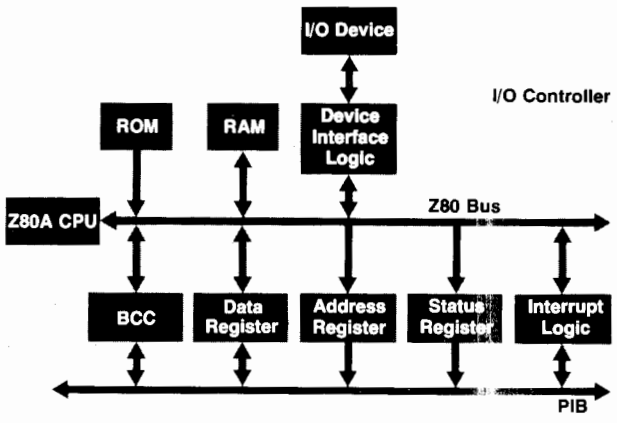


Fig. 2. *An I/O channel on the processor-independent bus (PIB).*

board ROM, and a PIB interface. The amounts of RAM and ROM are different on each controller. Each controller contains special hardware to interface with the device it controls (see Fig. 2).

All the controllers also have a similar software kernel to process PIB commands and channel programs. For example, the same channel program is used to download code into each controller for diagnostic purposes. The download feature is used during board burn-in in manufacturing. Specialized software and commands are used to access the controller device.

Alphanumeric Video Controller

The AVC acts as the interface between the MPU and the alphanumeric display subsystem. The AVC uses a proprietary video controller chip (VCC) to produce a serial video stream and raster timing signals.[3] The serial video stream along with four character color bits and a character clock are sent to the color mapper for color processing (see article, page 15). The raster timing signals are sent to the display monitor.

The AVC software translates the logical display structure used by the MPU from shared memory into a structure compatible with the alphanumeric video hardware. The translated structure is stored in a local 32K-byte display buffer.

In doing the translation, the AVC emulates certain functions indicated by the MPU display structure. For example, the VCC does not have half-bright enhancement. However, the display feature set of the MPU includes the half-bright enhancement. The AVC emulates half-bright by translating the enhancement to a selection of the yellow alphanumeric color. Thus, all half-bright fields are displayed in yellow.

Another transformation is performed with the native language characters. The character ROM on the AVC does not contain a separate native language character set. The native language characters are stored in the control code space of three alternate character sets. The AVC maps the native language character codes used by the MPU into the appropriate character set and control code selection to display the proper character.

Both of these transformations are transparent to the MPU. In the future, one could redesign the alphanumeric display subsystem with half-bright and true native language character sets. As long as the new subsystem conforms to the current MPU-AVC interface, it could be substituted with no change required in the MPU firmware.

Keyboard/Datacom Controller

The KBDC interfaces the MPU with two datacom ports and the keyboard. The KBDC also contains the nonvolatile RAM used to store terminal configuration data when the terminal is turned off.

The KBDC supports RS-232-C, RS-449, and RS-422 connections on its datacom ports. The KBDC handles all datacom protocol operations (i.e., handshakes, baud rate selection, parity checking, etc.). The MPU does not have to be concerned with any of the details of host data communications. To implement a different datacom protocol, only the KBDC needs to be altered.

The KBDC and the external keyboard each use a Z8 microcomputer to communicate with each other. The keyboard's Z8 microcomputer scans the keyboard. The KBDC firmware performs all the mapping from keyboard key translation to MPU key codes. The KBDC processes the conversion of keycodes according to the keyboard qualifiers (SHIFT, CAPS LOCK, and CONTROL keys) and maps the keys to conform to the configured native language. The conversion and mapping are dictated by transform maps downloaded into the KBDC from the MPU.

The keyboard also contains a graphics input device, which consists of two spinning thumbwheels for X-Y coordinates and a pushbutton switch. The KBDC code converts the thumbwheel motion into X-Y translation data for the MPU. The sensitivity of the thumbwheels is a configuration option of the KBDC.

Tablet Interface

The graphics tablet provides an example of the extensibility designed into the HP 2700. The inclusion of the tablet came toward the end of the project as an alternative graphics input device to the thumbwheels on the keyboard. However, the modular and device-independent nature of the PIB made the addition of the tablet straightforward.

The tablet interface connects the MPU and the graphics tablet assembly. The tablet assembly consists of a platen and a stylus. The stylus has a switch at the end that emulates the keyboard's graphics input device pushbutton. The tablet has three pushbutton switches used to select relative or absolute mode, cursor centering, and left or right-handed mode. Pen position accuracy is 0.5 mm.

To the MPU, the tablet interface is transparent, that is, there is no direct control of the tablet interface by the MPU during normal terminal processing. The effects of the relative/absolute and left/right-handed selection switches are local to the tablet interface. Pressing these buttons affects only the manner in which the tablet interface interprets input from the tablet. Pressing the center cursor button causes an absolute positioning command to the MPU to center the cursor on the display. Stylus movement is translated by the tablet interface to thumbwheel input from the keyboard and stylus switch input is translated to a keyboard graphics input device switch input. The input from the tablet interface is placed into the same buffers as the keyboard input, so the MPU cannot differentiate between tablet and keyboard data.

Shared Peripheral Controller

The shared peripheral controller provides the interface between the MPU and HP-IB devices. It allows HP-IB printers, plotters, tablets, disc drives, and other HP-IB devices to be connected to the HP 2700. The HP-IB board provides a high-level, device-independent command access to HP-IB devices.

At power-on, the HP-IB board identifies all HP-IB devices connected to the HP 2700. This information is given to the MPU. Each device is identified as a member of the printer, plotter, tablet, or disc class of devices. The class designation determines how each device may be accessed. All devices of a class are handled identically by the MPU.

Minifloppy Controller

The minifloppy controller interfaces the MPU to two 5.25-inch flexible disc drives integral to the HP 2700. The discs are double-density and double-sided, and have a capacity of 264K bytes per formatted disc. The MPU accesses the discs just as it does other discs, and the same PIB commands and channel programs are used.

Like the HP-IB board, the MFC handles all of the lower-level commands required to control the flexible disc drives. The MFC translates the MPU commands to move the read/write head to the appropriate sector and track, executes the requested I/O operation, and performs any error recovery, if required.

Acknowledgments

The success of the terminal subsystem was the result of the hard work of many people. John Harwell was involved in the early development of the MPU firmware. He and Ria Labato implemented the alphanumeric display subsystem software. Dan Akerhielm did the initial design of the memory boards and KBDC. Grant Head and Dave Breuer developed the MPU firmware, Greg Woods the hardware and software of the HP-IB controller, Cory Chan the universal RAM board, Steve DeSimon the KBDC and universal ROM board, and Ken Waln the tablet interface. Jon Krakower developed many of the automated tests for the controllers.

References

1. Hewlett-Packard Journal, Vol. 31, no. 10, October 1980.
2. Hewlett-Packard Journal, Vol. 34, no. 3, March 1983.
3. J.C. Roy, "A Silicon-on-Sapphire Integrated Video Controller," Hewlett-Packard Journal, March 1981.

A Computer-Aided Test and Tracking System

by Michael R. Perkins, Susan Snitzer, and Charles W. Andrews

TO HELP ENSURE that the HP 2700 Color Graphics Workstation would meet HP's high quality and reliability standards, special emphasis was placed on its manufacture. During early development of the HP 2700, a laboratory team was assembled to develop a testing system that would enhance the manufacturability and reliability of the HP 2700 series of products. This represented a significant departure from standard product development philosophy both in timing and in the quantity of resources invested. Typically, this type of effort would have begun much later in the development cycle and would have taken place in the manufacturing environment. By beginning the design and development earlier in the laboratory, the test system team was able to influence the structure of the product toward testability.

The test system tests HP 2700 printed circuit assemblies (PCAs). It provides quick sorting of functional and nonfunctional PCAs, PCA burn-in, terminal test, and a station for technicians to reproduce failures that occur elsewhere in the system. The system also collects data and checks for trends. This information can be fed back to the appropriate group to help prevent recurrence of the failure in future assemblies.

The test system is primarily a burn-in system, which provides the opportunity to eliminate the product's weaknesses in the factory so they do not lead to failures after delivery to customers. Burn-in is the process of applying temperature stress to an assembly to cause weaknesses in the assembly to become failures.

Before this system was developed, burn-in was accomplished at the assembled terminal level. After top-level assembly, terminals were placed in a temperature chamber where they were left unattended while performing their internal self-tests. When any board in the terminal failed and could not be repaired, a new board was substituted. At this point, restarting the burn-in cycle for the complete terminal might delay shipment, while continuing the old cycle would log insufficient burn-in time on the newly introduced assemblies. By doing board-level burn-in, the new system eliminates the need to burn in entire terminals, which requires large temperature chambers and a large inventory tied up on the manufacturing floor.

The HP 2700's architecture lends itself to solving most of the unit-level burn-in problems. HP 2700 PCAs are tested using only the HP 2700's main processing unit (MPU) to run tests, memory for program storage, and a special electronic tool (ET) PCA which communicates with the host computer and provides LEDs to display information for the operator. This combination, connected through a backplane bus, forms the basis of three of the test system's four test stations. Each test station is connected to the host computer, an HP 3000, whose primary task is data collection and distribution.

To store and retrieve information for each PCA, a unique identification number is required. Quick and accurate identification of each assembly is facilitated by bar-code labels mounted on the PCAs. The information on each label is displayed in both bar-coded and alphanumeric form. Bar-

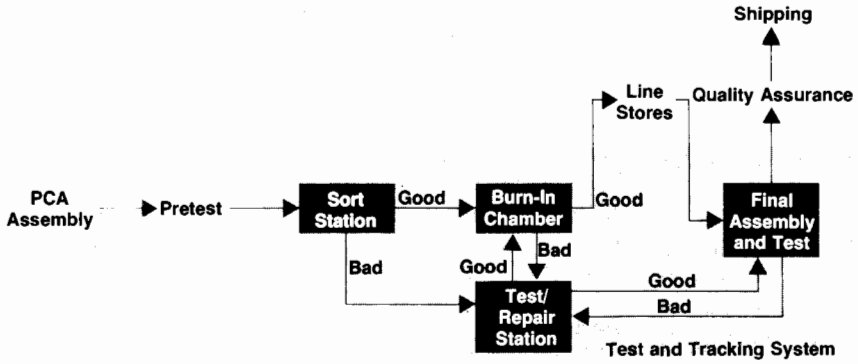


Fig. 1. *HP 2700 manufacturing process flow. After a printed circuit assembly (PCA) is assembled, it enters pretest, where power is first applied and rudimentary tests are performed. The PCAs enter the test and tracking system at the sort station. Burned-in PCAs are kept in line stores awaiting final assembly and test.*

code readers at each station are used to identify a PCA so that data for that assembly can be accessed from the system data base and the assembly can be tracked through the manufacturing process.

Fig. 1 outlines the manufacturing process flow. From pretest, PCAs enter the test system at the sort station where functional assemblies are separated from nonfunctional assemblies. Good PCAs proceed to the burn-in chamber where they undergo 48 hours of continuous testing and temperature cycling. After burn-in they are sent to line stores, from which they are pulled to build terminals at the final assembly and test station. Failing PCAs from any station go to the test/repair station where the operator can interactively execute all test programs in the test system, request looping on individual tests, and probe the PCA while the tests are being performed. From this station the operator also has access to the PCA's history stored in the data base, including what test failed and at what temperature. PCAs that fail after terminal assembly are replaced by equivalent ones from line stores. Thus shipments are not affected by having to restart burn-in.

Software Implementation

The test system software consists of many processes that need access to PCA and status information. All PCA data is stored in a data base, while information such as test unit status and chamber status is distributed throughout the various programs. Therefore, central to the software structure is the concept of shared data and interprocess communication. The process interconnect diagram, Fig. 2, depicts the software structure.

So that processes can communicate, each process has a command file associated with it. This sequential file is organized as two circular queues, a normal queue and an immediate queue. A file header is used to maintain the pointers to the queues within a file (Fig. 3). A program receives commands and data from other programs through its own command file. To send a message to another program, the source program simply places a message in the destination's command file and updates the pointers within that command file.

All data base accesses in the system are performed using command files through a program called the data base man-

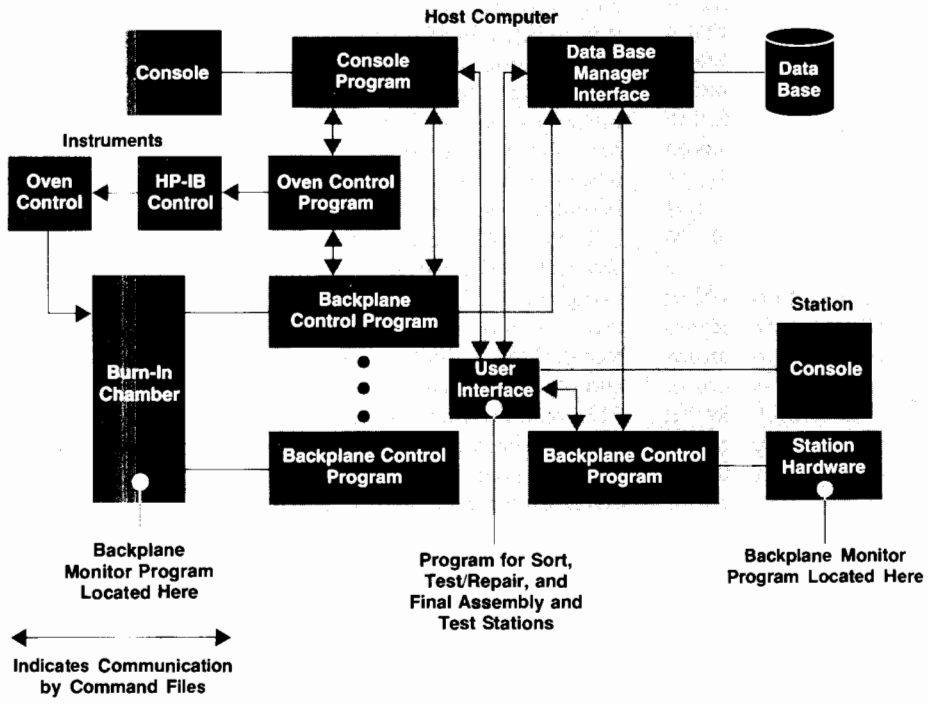


Fig. 2. *Processes in the test and tracking system's software structure communicate by means of command files.*

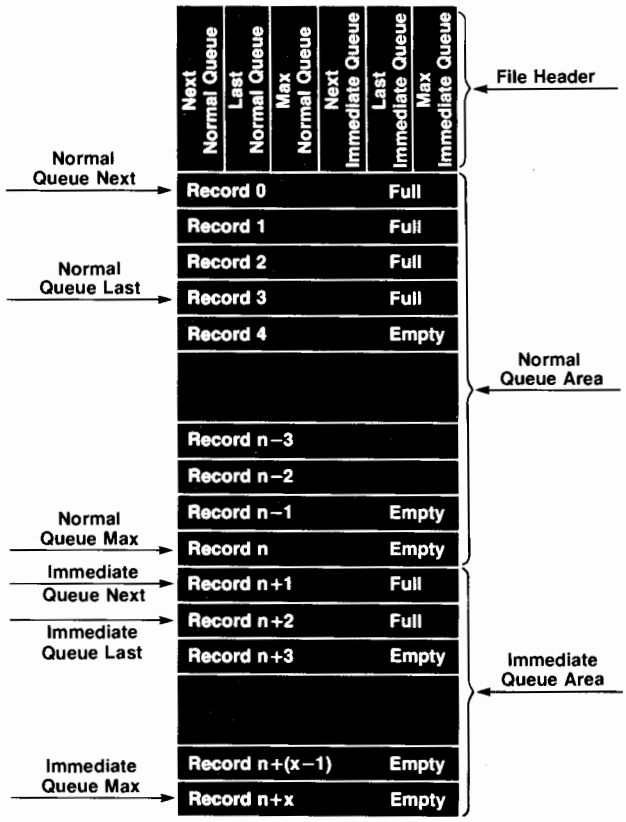


Fig. 3. *Internal structure of a command file.*

ager, which uses the IMAGE/3000 software subsystem. The primary function of the data base manager is to read and write information in the data base for the other programs in the system. Multiple copies of this program can run simultaneously and access the data base concurrently. All test system programs simply send the predefined commands (Fig. 4) to the manager and poll their command file for a response.

To administer the data base manager programs, there is a program called the data base manager initiator. The initiator activates multiple copies of the manager program and assigns each to a test program. Like all other programs, the initiator has a command file associated with it. To access the data base, all programs must request use of a manager from the initiator. The initiator then assigns a manager to that program based on current system load.

How Failures Are Detected

Data in the test system is collected at four locations: the sort, chamber, final assembly and test, and test/repair stations. A test unit monitor program running at each station exercises all boards and reports failure results to the data base. First, the monitor program requests all boards in the test station to identify themselves. If the identification does not correspond with the one indicated in the data base, an error message is displayed on the ET LEDs. Second, a downloading of all exhaustive test programs associated with the current test station configuration is requested. As the programs are downloaded, they are placed in the test station's memory.

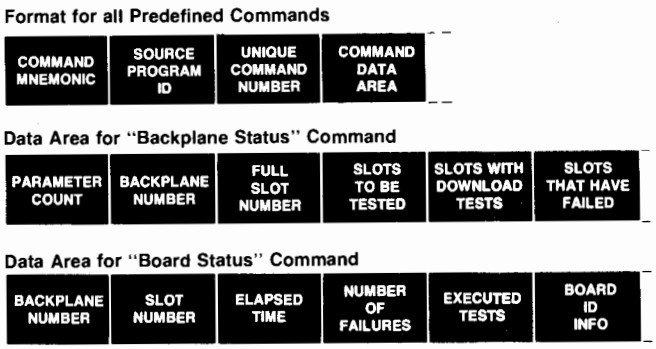


Fig. 4. *Structure of the board status and backplane status commands.*

Boards in the test station are exercised in sequential order. Intelligent PCAs perform four predefined self-tests plus the downloaded exhaustive tests. If the PCA is not intelligent, only the downloaded tests are performed. At the end of each test, the monitor program checks the results. If there was a failure, an error message is formatted, sent to the data base, and displayed on the LEDs.

The console program monitors all processes. This program acts as the central reporting mechanism for all test system software components. It runs on a printing terminal on the manufacturing floor and serves as the primary input/output mechanism for the manufacturing line supervisor.

Hardware Structure

The test system hardware was designed with two objectives: first, that all stations be similar, and second, that they use standard HP equipment or HP 2700 assemblies.

Standard HP 2700 cards—backplane, MPU, and memory—form the basis for the sort, test/repair and chamber stations. The only special-purpose electronic hardware is the ET PCA, which provides the data communication link between the HP 3000 and the MPU board, drives the LED displays on an attached display expander board, and provides the clock signals.

The sort and test/repair stations are identical and consist of the basic test hardware mounted on a metal box that also contains a power supply and cooling fans. The card cage is an HP 2700 part that has been cut away to permit access to the test points. The addition of a bar code reader and a terminal makes the stations complete.

The chamber station uses the basic test hardware mounted in specially constructed cabinets. Partitions allow the MPU, memory, and ET boards to remain at room temperature while up to 13 PCAs under test are subjected to temperature cycling. Power is supplied through bus bars to all the PCAs in the chamber from three HP power supplies.

Control of the chamber and its power supplies is achieved through a system of HP-IB (IEEE 488) instruments and a chamber control program running on the HP 3000. The combination of an HP 5328A Scanner and an HP 3495A Counter/Digital Voltmeter permits the chamber control program to control the temperature and the power supplies for the chamber. User control of these functions can also be exercised from the system console.

Conclusion

The test and tracking system provides a means not only to test and burn in HP 2700 PCAs, but also to provide data for quality assurance. All data base information can be accessed by means of a utility program according to board type, date, or identification number. In this way, faulty PCAs do not reach the customer and the failure data can be used to improve the quality of the product.

Acknowledgments

Special acknowledgments go to Jack Frost, John Pattinson, Don Bennett, Ngo Viet, Anne Park, Tony Williams, Peggy Labell, and Roland Hwang for their contributions to the project.

Product Design of a Friendly Color Graphics Workstation

by Dennis C. Thompson, Kenneth D. Boetzer, Mark A. Della Bona, and Badir M. Mousa

THE HP 2700 IS A POWERFUL YET FRIENDLY series of high-performance color graphics workstations. Its advanced technical capabilities for generating and manipulating images require not only a significant amount of local intelligence and memory, but also an equally capable user interface. The HP 2700 product design challenge was the integration of these features in an ergonomic desktop package that does not intimidate the user because of size, noise level, or apparent complexity.

Cooling System Design

The HP 2700 is a complex assembly dissipating nearly 500 watts of power. Desktop packaging requires a relatively small volume, placing heavy demands on the cooling system. Reliability objectives seek to minimize the internal temperature rise. Unfortunately, merely installing larger fans for extra cooling capacity produces higher noise levels than are acceptable in an office environment, and fans themselves are sources of reliability problems.

Numerous layouts and models were created to examine the placement of component assemblies and their effect on cooling strategies and product ergonomics. The final configuration (see Fig. 1) provides adequate cooling with the use of only two fans. Noise levels are maintained well below an acceptable 55 dBA.

The cooling system is designed to take full advantage of the various natural phenomena involved in thermal systems. The logic cards are oriented vertically to let free convective flow help the fans. Sensitive logic components are placed near the openings for cool incoming air. Less sensitive high-power components are mounted farther along the airflow path to make best use of the system's available cooling capacity.

Two fans exhaust the heated air, creating a negative pressure inside the package. A negative pressure system directs cooling airflow with appropriate placement of inlets and baffles. The ability to direct this cooling air and tailor the flow more than compensates for the decrease in fan efficiency caused by moving heated, less dense air.

External Design

Design models and experiments provided insight regarding display height and angle, location of local mass memory storage, controls placement, keyboard layout, and graphic input device preference. The HP 2700 has the power switch and brightness controls at the front of the machine where the user has easy access to them. The detachable keyboard plugs into the front of the terminal, minimizing the length of the coiled-cord cable so that clutter on the operator's work surface is kept to a minimum and the keyboard can be comfortably positioned.

Controls for adjustment of the display's convergence are easily accessible at the front of the terminal while viewing

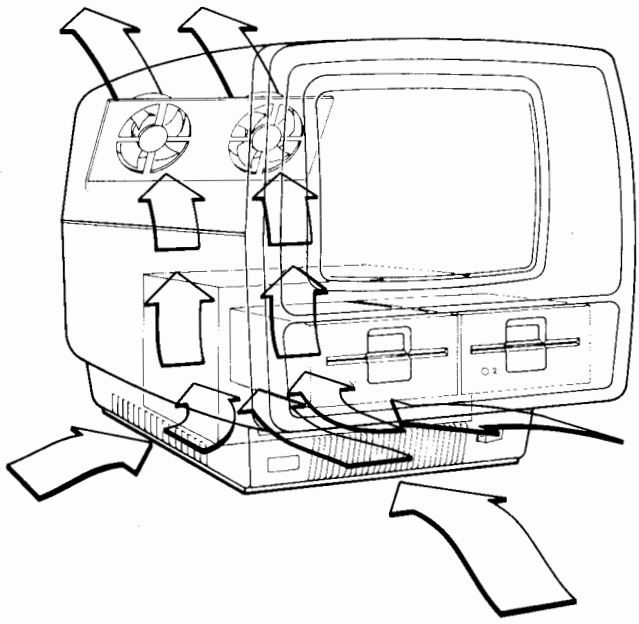


Fig. 2. *For convergence adjustment, color-coded potentiometers behind a swing-down panel are adjusted while watching a target pattern on the screen.*

the display (Fig. 2). A swing-down panel reveals color-coded potentiometers that directly correlate with the software-generated target pattern on the screen. Simplified graphics images on the panel communicate the convergence procedure so that no written instructions or manual are needed to converge the display accurately.

Color Selection

Product and industrial designers cooperated closely with software and hardware engineers in successfully developing an effective and easy-to-use color selection system for the HP 2700. This system is described on page 10 of this issue.

Serviceability

The designer's influence is also felt by service personnel. Quick and convenient access to parts and assemblies can dramatically reduce service costs. The logic cards of this machine are user-accessible by design, allowing easy configuration. Door lock details provide the customer the option of locking the door with a novel type of key (Fig. 3). Should disassembly be required, interlocking details on the plastic housing skins enable nearly complete access with minimum removal of hardware. The top cover is removed by undoing two quarter-turn fasteners, providing access to all power supply and display adjustments and test points. Further disassembly to a fully opened chassis requires the removal of only two additional screws.

Display Enhancement

The eye is a delicate, although highly adaptable optic system that can be easily strained to the point of fatigue. The problem becomes more severe with age because the lens of the eye loses its elasticity, thus limiting the range of focus. With this in mind the prime concern was to create a display with exceptional clarity, legibility, focus, resolution, and contrast and a minimum of surface reflections.

Typically in the use of a CRT display, stray light from windows, overhead lighting, and bright surfaces bounces off the screen's face and causes undesirable reflections. This can reduce the contrast between the characters and background, thus negatively affecting the operator's visual acuity.

There are a number of ways in which the designer can provide antireflection and contrast enhancement. The most effective solution for dramatically reducing reflections is achieved by bonding a thin-film optically coated panel onto the CRT face. HEA® coating uses the physics of light reflection on glass and allows more light to be transmitted instead of reflected. Reflections are reduced to a minimum and there is no compromise in character focus or brightness. What does affect brightness is the lower transmission of the dark glass panel. However, this panel increases the contrast between the characters and background significantly. The relatively small loss in brightness is far outweighed by the benefit gained in increased contrast and richer colors.

The HEA coating solves the glare problem and improves the contrast of the display. It maximizes readability by maintaining sharp, crisp images with no loss in resolution. The coating is unaffected by temperature or humidity and is virtually abrasion-proof in normal use. It also enhances the color display providing richer, deeper colors.

-Bud Mousa

Acknowledgments

This product represents the efforts of many people. Dave Lima, Richard Bergquam, Michael Perkins, and Zong Luo were responsible in part for the product design through manufacturing release. Donn Gooch was involved from an industrial design perspective for some of the software. Baltazar Franco produced hundreds of square feet of meticu-

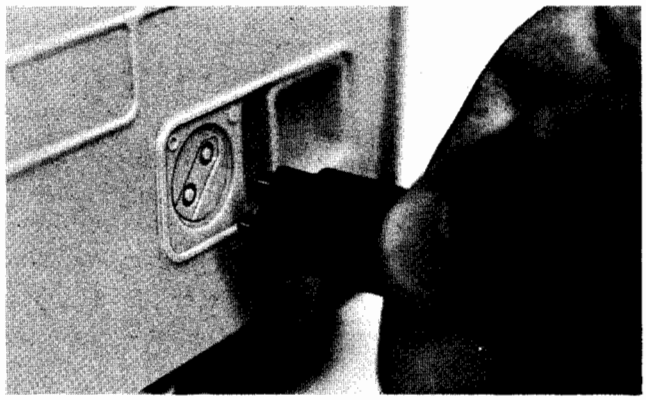


Fig. 3. *A novel key can be used to lock the door to the card cage.*

lous mechanical drawings including exploded isometric views for assembly aids. Roger Slocum and his team contributed with their model-making expertise throughout the project.

HP 2700 Graphics Input Devices

The two principal graphics input devices for the HP 2700 are the keyboard graphics input device, which consists of two thumbwheels and a button, and the optional HP 13273T Data Tablet.

The HP 13273T Data Tablet (Fig. 1) makes software applications such as PAINTBRUSH/2700 possible. It allows the user to do freehand sketching and easily enter data into the workstation without the keyboard. Operation of the data tablet is similar to that of the HP 9111A Graphics Tablet.[1] However, the 13273T does not have all of its electronics within its case. Much of the tablet's interfacing electronics resides on a printed circuit board inside the HP 2700 card cage, and the tablet derives its power from the HP 2700 power supply.

The design goals for the data tablet included minimum package size and weight, ease of use, and low cost. Package size, form, and weight are critical because the tablet may be held in the user's lap. For this reason the package is very slim, about the thickness of a writing pad, and weighs only six pounds. For use on a work surface, a flip-up support holds the tablet at an angle typical of a drafting surface.

There are three function keys on the tablet. One of these keys lets the user set up the tablet for left-handed or right-handed use.

Stylus cable management is an area of contribution. In the past, tablet products were designed without much concern for where the stylus cable might be stored, with the result that it can dangle off the work surface onto the floor, often incurring damage. In the 13273T design, a groove is provided for storing the stylus cable, thus providing a solution to this problem without incurring any additional cost (Fig. 1).

The platen surface is designed for low cost and improved human factors. This surface is a combination of a Lexan™ overlay bonded with a pressure-sensitive adhesive to a fiberglass substrate, which is then mounted into the case with the platen printed circuit board. This assembly makes improvements in two areas. First, the Lexan overlay bonded to the fiberglass is a more friendly interface. The working surface of the tablet is warmer to the touch than other tablets, which use a cold glass surface. The Lexan is much less costly than glass. In addition, bonding glass to a printed circuit board is difficult and costly, because the epoxy lamination requires heat and high pressure. Extensive testing was done to ensure that the Lexan overlay and the fiberglass substrate do not delaminate.

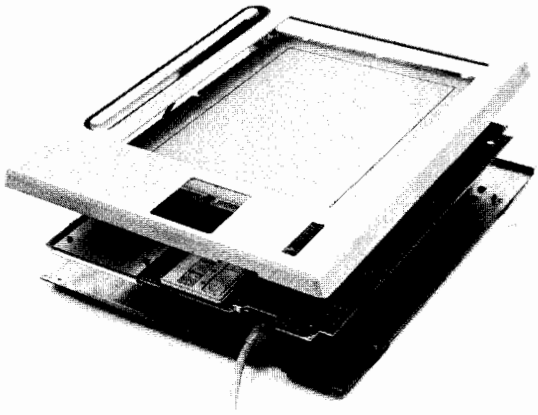


Fig. 1. *The HP 13273T Data Tablet is simple in design and has few parts. A groove in the top cover is provided for storing the stylus and cable when not in use.*

Assembly of the tablet is simple and the number of parts is kept to a minimum. This helps keep the manufacturing cost low.

Keyboard Graphics Input Device

Many keyboard graphics input devices were surveyed in designing the HP 2700 keyboard. Some OEM products were purchased for evaluation and some new ideas were designed and prototyped.

After extensive investigation, the concept that was decided upon uses two digital thumbwheels (see Fig. 2). The wheels are

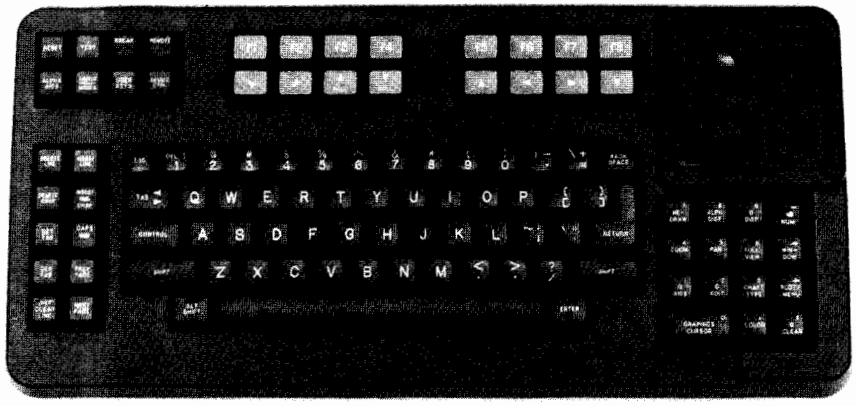


Fig. 2. *The HP 2700's keyboard graphics input device consists of two thumbwheels and a button. The thumbwheels are connected to digital shaft encoders.*

placed in orthogonal positions with vertical and horizontal relationships to the screen, and allow the user to move the cursor vertically and horizontally. They are designed to spin freely to produce a desirable cursor coasting effect. HP optical shaft encoders are used. They are compact enough to meet the space requirements and they are perfectly compatible with HP 2700 logic levels. No other electrical hardware is necessary to support the encoders. The shaft bearing requires a special blend of polycarbonate and Teflon™ and the shaft is a hardenable grade of steel. The bearing and shaft material are carefully matched to achieve long life.

The encoders and shaft supports are attached to a mounting plate, which is mounted into the keyboard case. A momentary switch and button are added to the plate to digitize the cursor position.

-Bud Mousa
-Dennis Thompson

Reference

1. D.J. Stavely, "A High-Quality, Low-Cost Graphics Tablet," Hewlett-Packard Journal, Vol. 32, no. 1, January 1981.

AUTOPLOT/2700: A Simple Approach to Custom Chart Generation

by Stanley A. Balazer and John M. Perry

AUTOPLOT/2700 is a powerful applications software package for the HP 2700 Color Graphics Workstation family. It provides an extremely friendly menu-driven interface for high-quality decision support graphics such as pie charts, bar charts, line charts, log charts, and scattergrams. In addition, text slides can be designed in a totally interactive environment.

AUTOPLOT/2700 uses the extensive features of the HP 2700. In addition to the menu-driven interface, there is an interactive interface which allows the user to point to a portion of a chart with the graphics cursor and then change its color, size, shading, font, or other attribute immediately. Whole charts can be moved, scaled, and combined with other charts and text to form a complete graph. Since all the chart types use the same data menu, the user can choose between pie, bar, and linear charts and see the results in a matter of seconds.

Once a set of charts is ready for hard copy, AUTOPLOT/2700's spooling capability allows unattended plotting of multiple copies of up to 17 different charts.

Using AUTOPLOT

Placing the AUTOPLOT disc in drive #1 and pressing the **CHART TYPE** key starts up the application and brings up a chart type display (Fig. 1). This serves as a home base from which all the major parts of the system can be accessed. The parts are data entry, chart format specification, text slide and labels, saving and recalling charts, combining charts, hard copy, and multiple hard copy. Pictures of the major chart types are used for selection of the type of chart to be created. The keyboard's thumbwheels are used to place the cursor on the picture of the chart type desired. Pressing the thumbwheel button brings up the interface required to construct that type of chart. Softkeys on this display give access to the hard-copy, chart saving and recalling, and visual chart formatting features. At any point in the chart design process, the **PLOT/MENU** key on the keyboard presents the current chart on the display. Pressing it again returns the chart design display. The **CHART TYPE** key on the keyboard brings up the chart type display. The **G AIDS** key on the keyboard displays help text, which explains the basics of the current level of the application. Pressing any softkey brings up details explaining the key. Pressing **G AIDS** again returns the previous display.

Simple Charts and Transparencies

AUTOPLOT can create a chart with only minimal data from the user. The software makes appearance decisions that are consistent and produce good-looking results. This reduces graph production time and gives a quick preview, providing an opportunity to make changes without laborious input. One set of data can be applied to any available chart type. Numbers can be input without being tied to any particular graphic format. These features mean that a user can enter data, evaluate it in any graphic format, and output a finished chart to hard copy very quickly.

When a chart type is chosen from the pictures on the main display (Fig. 1), a basic data menu is presented (Fig. 2). Choosing a bar chart and entering the data shown in Fig. 2 is all that is necessary to create the chart shown in Fig. 3. AUTOPLOT chooses colors and positions of chart elements, axis scaling, and text layout. Returning to the chart type display allows a different chart type to be chosen and plotted with no further input required.

The data menu is the same for all charts except pie charts. Data columns are labeled according to the chart type. The

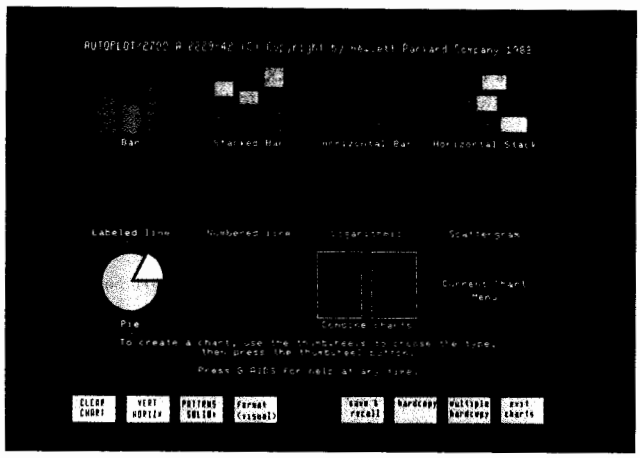


Fig. 1. *AUTOPLOT/2700 main chart type display provides access to all major parts of this applications software package.*

menu allows lines of data to be rolled, paged, deleted, and inserted. Columns can be paged (20 columns are available) and cleared.

To use data from a host computer, the save & recall softkey is chosen at the chart type level. The user can then go into remote mode and run a host application. AUTOPLOT can capture data from any host application that outputs a table of data to the display. To capture the data, the RECALL DATA softkey is pressed before instructing the host to send the data. The data is stored in the local AUTOPLOT data memory rather than being displayed on the screen. After the data is collected, all the local features of AUTOPLOT can be used to display the data without the need to retransmit the data.

The text slide system is optimized for making simple text transparencies easily. Therefore, the flexibility of the text options is limited. The high-speed graphics of the HP 2700 are used to allow dragging text and pointing at a text feature display to modify text instantly. The text chart is displayed at all times just as it will appear on hard copy.

When the text slide and labels picture is chosen on the chart type display, the display shown in Fig. 4 appears. It includes a workspace for the slide, text feature choices, and softkeys. To create text, the thumbwheels are used to place the graphics cursor where the text is to be positioned, and the text is entered through the keyboard. As it is typed, it appears on the display. The text can be dragged and features chosen by pointing at the display on the right side of the screen with the cursor. Lines can be deleted and inserted. Groups of text can be moved and text features modified by using the GROUP MODIFY softkey. A box is placed over the text to be included in the group, and then the box is moved to move the text, or features from the text feature display are chosen to change the appearance of the text.

Once a chart has been created, the save & recall softkey at the chart type level offers the opportunity to save the chart on a flexible disc. At the same level, charts can be recalled, listed, and purged, and the hardcopy softkey can be selected for output of the chart to a hard-copy device.

Custom Charts

Whether a chart is for the board room or a weekly report, customization is the key to more effective presentation. With AUTOPLOT/2700 the user can create a custom chart format. While the casual user need not go beyond the simple chart stage, it is easy to change a few colors or placements and have a custom chart.

Pressing the format (visual) softkey brings the user and a newly formed chart to a new program level (see Fig. 5). On the right-hand side of the screen a menu with colors and patterns is drawn. To the left of this is a workspace displaying the current chart. Format changes are specified using the thumbwheels and softkeys. A basic set of formatting operations can be done at this level. Keeping the set small enhances the level's ease of use.

In chart makers that rely solely on alpha menus for format control, the menus are complicated for the casual user, so complicated that many potential users avoid the program. With the HP 2700's visual formatting the user sees the changes happen as they are selected.

To locate the region to change, the user first presses the LOCATE softkey. The graphics cursor then points to the current region and the name of the region appears in the message window directly above the softkeys.

For example, to change the CABLES bar color in Fig. 5 from red to green, the user presses the NEXT REGION softkey until the graphics cursor is over the CABLES bar. The words BAR 3 (CABLES) will appear in the message window. To select a new bar color the user now presses the SELECT softkey. The graphics cursor points to the bar's current color in the upper right-hand menu. A new pen is selected by placing the graphics cursor over the desired color using the thumbwheels and then pressing the thumbwheel key. At this time the bar chart in the work area is updated with the new green bars. Area fill and line patterns are selected in a similar manner using the lower right-hand menu.

The LOCATE AXIS softkey gives the user powerful screen management ability. With it a user can position a graph anywhere in the visual format workspace. At the same time the graph can be rescaled in both height and width. Axis labels keep their same screen-based size while the graph is scaled down. The axis tic and label intervals also vary with the graph's screen size.

Additional text can be added to any area of any graph with the previously described TEXT SLIDE level of AUTOPLOT. The full feature set of the TEXT SLIDE level is available to add text with various fonts, sizes, and colors. The ability to reenter TEXT SLIDE to add more text or edit the text

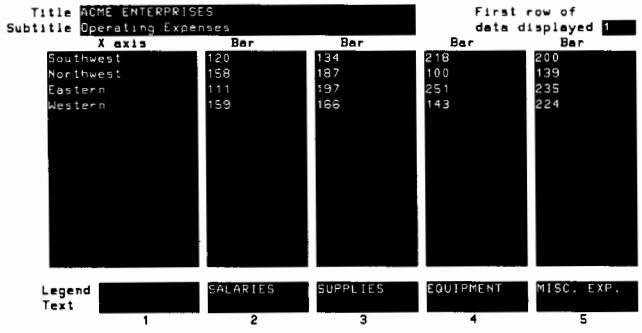


Fig. 2. *After a chart type is chosen, a basic data menu is presented for the user to fill in.*

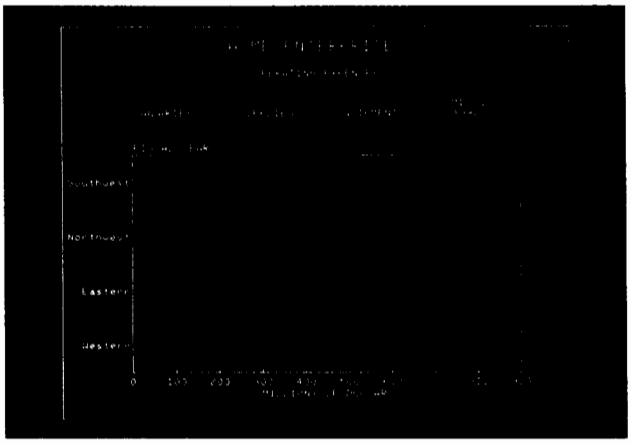


Fig. 3. *This is the result of choosing a bar chart and entering the data shown in Fig. 2. AUTOPLOT chooses colors and positions of chart elements, axis scaling, and text layout.*

previously added gives AUTOPLOT integrated capabilities not found in most chart making applications.

Visual formatting is not the only way to change chart format. There is also a format menu (Fig. 6). A user reaches this alpha-based format menu from the data menu described earlier. Everything that can be done in visual format can also be done via this menu, and any changes done in visual format are reflected in this menu.

Another alpha menu, the axis menu, is reached from either the data or format menus. Its purpose includes axis label control, grid/tic placement, and axis scaling.

Saving Chart Formats

The user can save any chart on a flexible disc via the SAVE & RECALL section of AUTOPLOT. When a chart is saved, two separate files are created, a data file and a format file.

The concept of the format file is important. The format file contains all of the custom chart details: axis placement, colors, data ranges, etc. Data can be saved and recalled separately. This allows many sets of data to be graphed using one custom chart format. This feature is especially useful in periodic reporting.

Multichart Pages

One of the main goals of a business graphics presentation system is the production of graphs in their final presentation form. Previous systems have not carried this goal to multichart pages, that is, two or more graphs on a page.

The COMBINE CHARTS level provides an integrated solution to this problem. Presentation-ready charts as shown in Fig. 4 can be made quickly. This demonstration chart was made from two previously saved charts. The text was added to the combined chart with the TEXT SLIDE level.

Charts can be placed anywhere on the workspace at any scale. The placement is much like the LOCATE AXES procedure in visual formatting. When the user presses ADD CHART, the terminal asks for the previously saved chart file name. The graphics box cursor appears on the screen and the user positions and scales it via the thumbwheels. Pressing the return key causes the chart to be drawn in the selected area.

For standard two-and four-chart-per-page layouts, NEXT HALF and NEXT QUARTER softkeys are provided. They step the box cursor through the hemispheres or quadrants of the chart for exact chart placement.

Periodic Reporting

Periodic reporting is the use of a series of charts to convey information about a subject, usually updated and distributed in weekly or monthly intervals. The charts themselves usually have the same format each period; only the data changes.

AUTOPLOT is designed to make both the initial creation of the report and its periodic updating simple and efficient. The AUTOPLOT file organization is designed so that a chart format can serve as a template for other charts differing only in data.

Once a report has been created, up to seventeen pages can be output to hard copy automatically using the multiple hardcopy softkey on the CHART TYPE level. The names of the pages are entered onto a menu, along with specification of

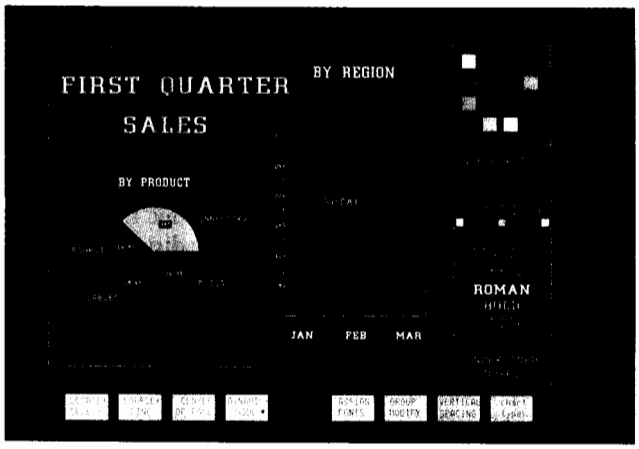


Fig. 4. *When the text slide and labels picture is chosen on the chart type display, a display like this appears. It includes a workspace for the slide, text feature choices, and softkeys.*

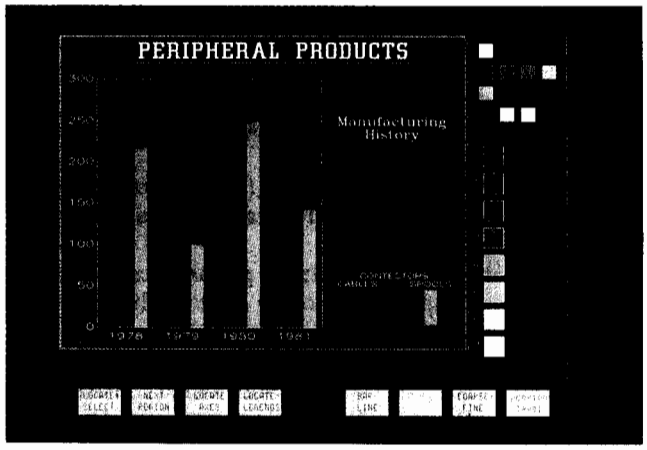


Fig. 5. *Charts can be left as drawn by AUTOPLOT or customized. The visual formatting program level shows the current chart and a menu with colors and patterns. Format changes are specified using the thumbwheels and softkeys.*

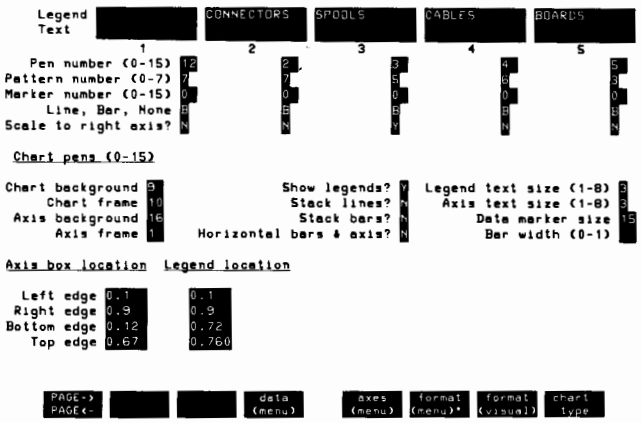


Fig. 6. *This alpha-based format menu is an alternative to visual formatting for customizing charts.*

the number of copies of each page to be output. The entries on this menu can be saved on a flexible disc so the report can be called up and output to hard copy with very few keystrokes.

The report update procedure is supported by a data modification procedure that deals only with data. Data can be added, deleted, or modified without having to deal with the chart format. This allows for fast update, whether the data is edited from the keyboard or is gathered from host computer. In addition, many charts can be linked to a single data file, so only one file needs to be recalled, modified and saved to update the report. This results in significant time savings, and also allows data to be shared by many charts without appearing many times in different data files.

PAINTBRUSH/2700: A General-Purpose Picture Creator

by John R. Alburger, Jim L. Davis, Diane A. Rodriguez, and Barbara A. Stanley

FOR APPLICATION PROGRAMMERS, the HP 2700 Color Graphics Workstation provides an extremely powerful graphics feature set. However, novice and casual users demand easy-to-use graphics systems. Designed to meet this challenge, PAINTBRUSH/2700 is a general-purpose graphics application program that allows users to create and manipulate pictures easily, naturally, and interactively.

PAINTBRUSH/2700 is designed to allow nonprogrammers to:

- Create and edit pictures and text
- Combine independently created pictures
- Interface easily with other HP 2700 graphics application programs, for example to:
  - Create graphics object libraries (i.e., collections of previously drawn objects) for users and applications programs
  - Edit or manipulate arbitrary pictures created with other local applications or by host application programs.

PAINTBRUSH/2700 is available with the HP 2700 Model 65 Presentation Graphics Workstation, which consists of an HP 2700 Color Graphics Terminal, a graphics tablet with stylus, one-half megabyte of memory, two flexible disc drives, and a flexible disc containing PAINTBRUSH. This application runs locally on the HP 2700 Terminal; no host computer is required.

PAINTBRUSH consists of a set of hierarchically organized levels. As shown in Fig. 1, each level displays a maximum of sixteen softkey labels across the bottom of the screen, a menu along the right-hand side of the screen, and a graphics workspace area. The user selects a level or function by using the stylus to position the cursor over the softkey label and then picking (i.e., pressing the stylus tip). Items such as pen color, line type, or area fill type can be picked from the menu.

The user interface of PAINTBRUSH is versatile. Users can perform operations using either the tablet or the keyboard function keys and the terminal's graphics input device. Thus, users are not forced to bounce between the keyboard and the tablet.

Functions of PAINTBRUSH

PAINTBRUSH provides a complete set of graphics functions. These graphics functions can be divided into the four categories of picture creation, picture editing, terminal graphics functions, and input/output.

Picture creation is made simple by using the two drawing levels of PAINTBRUSH. The user can choose aided drawing for doing flowcharts and other drawings that might make use of fixed shapes (rectangles, circles, circular arcs) or curved and angular lines. If the picture requires a special touch, the free drawing level gives the user the option of selecting from a menu of pen tips, including tips for thick lines, calligraphy, and airbrushing.

Area fills can be done with previously defined fixed

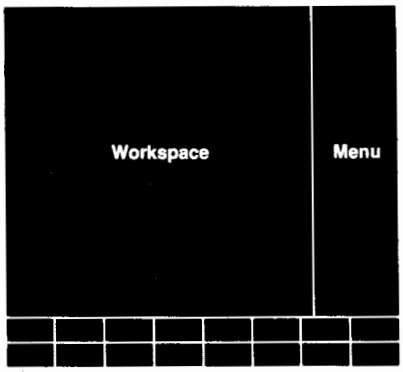


Fig. 1. *Each of PAINTBRUSH/2700's levels displays up to sixteen softkey labels, a menu, and a graphics workspace area. Two drawing levels support aided or free drawing.*

shapes or by freehand drawing. Complex area fills (i.e., areas within areas) are also allowed.

The vector is the smallest unit of the picture file. A collection of vectors defines an object. Objects are created by a single keystroke and contain all the vectors that were drawn since the last object was created. Objects are the smallest building blocks in picture creation. Objects can be moved, scaled, rotated, copied, or deleted. For example, a forest can be drawn by creating a single tree as an object and then replicating that object to make several trees (see Fig. 2).

The user may wish to add some text to the picture. The labels level of function keys allows for text addition, manipulation, and deletion. Text is entered by placing the cursor at the desired start point and typing the label. A highlighted box shows the placement and size of the new label. The text size, color, angle, italics, spacing, justification, and font can be selected from either the softkeys or the menu. Fig. 3 shows examples of the text capabilities. The label is drawn and the cursor moves to the next line when the **RETURN** key is pressed.

The HP 2700 supports user-defined fonts. This means that customized text styles can enhance labels. PAINTBRUSH can store as many as seven fonts at one time and comes with Roman, bold, and smooth fonts in addition to the standard stick font. The standard fonts as well as any user-defined fonts are displayed on the font menu for selection.

PAINTBRUSH has a configuration section to aid in the picture creation process. The configuration level allows specification of alignment and increments of a grid. When active, the grid forces the cursor to align to a grid point. This grid can help in the placement of shapes or text. Multicolumn text alignment is easy with grids. A grid can be lines or dots, and can overlay or underlie the picture.

Picture Editing

Once a picture is created, either by a host application, by another local application, or by PAINTBRUSH, PAINTBRUSH allows the user to edit or manipulate the picture. Picture editing functions can be categorized into three areas: text editing, object manipulation, and picture merging.

Existing text can be moved, scaled, rotated, copied, or deleted. The change is displayed immediately to provide a high degree of interactiveness. Any text attribute may be applied to a single label or to a group of labels. The label group can be selected either by picking the labels one at a time or by positioning a box so that it encloses the desired group. Label groups allow for manipulation of multiple rows of text.

The labels level offers two modes of text scaling. Sizing mode alters the size of the text. Spacing mode can be used to compress or expand the distance between text characters. This can be helpful when squeezing a label into a tight spot. The group modification function along with the attribute modification functions provide a powerful set of text layout tools.

PAINTBRUSH allows the user to manipulate objects (which may consist of drawn figures and/or text) or groups of objects. The user can delete, move, scale, rotate, or copy objects, or place objects on top of objects. The user always has the ability to abort or undo any operation. PAINTBRUSH maintains a history list of operations so that the user may undo an operation many times. While performing object transformations, the user may choose to activate and/or show grids. This level provides a choice of sixteen active colors (selectable from 4096 colors) for the picture background.

Objects can be individually selected (picked) by moving the stylus until the cursor is positioned over a line, letter, or area fill of the object, and then pressing down on the tip of the stylus. Groups of objects can be selected by individually picking the objects, by specifying them by their object names, or by positioning a box so that it encloses the desired group. Objects or groups are deselected when another object or group is selected.

Fig. 4b was created by performing picture editing operations (i.e., copy, move, scale) on Fig. 4a.

PAINTBRUSH also allows pictures to be combined easily and naturally. To merge pictures, the user positions a box where desired and scales the framed area to the size desired. With the aspect ratio preserved, the picture is copied from the disc and displayed on the screen within the bounds of the box, becoming a part of the currently displayed picture. Fig. 5 shows the results of combining four different pictures.

Terminal Graphics Functions

PAINTBRUSH allows the user to do some terminal functions while still in the application subsystem. This saves the user many keystrokes and avoids extra picture redraws that would be done on exiting and entering the application. Most of the graphics keypad fuctions are available. An added feature of the **FULL VIEW** key, **SHIFT FULL VIEW**, restores the virtual window to what it was when PAINTBRUSH was entered. This provides a quick return to the original picture after using the **ZOOM**, **PAN**, **FULL VIEW**, or **WINDOW** keys.

Input/Output

The input/output level of PAINTBRUSH allows pictures to be copied from flexible disc, saved on disc, and plotted. When pictures are merged, aspect ratio is preserved, object

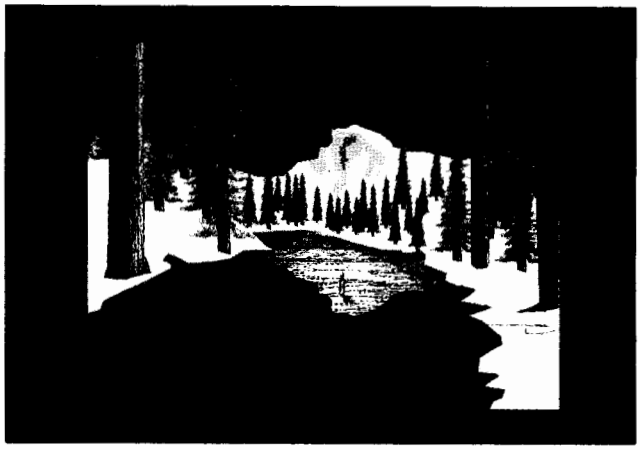


Fig. 2. *A forest can be drawn by creating a single tree as an object and then replicating the object.*

Fig. 3. *Text capabilities of PAINTBRUSH. Seven fonts can be stored at one time and user-defined fonts are supported.*

Implementing HP 2700 Applications Software

Applications on the HP 2700 Color Graphics Workstation run on the same processor as the terminal main code. Therefore, the application exists in a tightly integrated environment.

Applications are developed on the HP 3000 Computer System and downloaded to the workstation. AUTOPLOT/2700 and PAINTBRUSH/2700 are mainly written in Pascal, although there are a few routines written in Motorola 68000 assembly language. The tools required for their development were a Pascal compiler, a 68000 assembler, an applications linker, and a downloader from the HP 3000 host computer.

Program Environment

The application has a set of standard entry points. These entry points make up the first four procedures in the first segment. The first entry point is called when the program is entered. The second entry point is called on exit from the application. The third handles application soft reset. The fourth routine is called just before the application is deleted from memory.

The application environment has three different models. The first model involves a single call to the application. The application then runs to completion and gives control back to the workstation. In the second model the workstation, acting as the main program, calls application subroutines when receiving user input. AUTOPLOT/2700 uses this application environment. The third model is structured so that the application polls the workstation for input. The workstation calls subroutines in the application for the specified input. For example, input could come from the thumbwheels, the graphics keypad, or the softkeys. This third model is the application environment used by PAINTBRUSH/2700.

Once the application is loaded, it is resident in memory until a hard reset occurs or another application is loaded. On soft reset the workstation gains control and calls the application reset handler. The handler can return the application to a known state. Soft reset can be controlled by the application by bracketing code sections as critical or noncritical.

There are five different Pascal procedure calls to the graphics intrinsics. They allow for integer, real, or string (long and bounded) parameter passing. These intrinsics call the graphics escape sequence action routines.

Terminal functions are accessed by calling routines in an intrinsics library. These routines are external calls in the source code and are mapped to the correct intrinsic by an intrinsic number at link time. Some examples of intrinsics commonly used in applications are routines to display the softkey labels, display a message on the screen, draw a vector, and ring the bell.

Segmentation and Swapping

An application may be divided into as many as 32 segments. Each segment may contain multiple object files. At link time, the application linker resolves external calls to routines within the same segment as well as in other segments.

Applications can determine the amount of program memory required at link time or at run time. Depending on the terminal configuration at run time, an intrinsic enables the application to ask for different amounts of memory for swapping or nonswapping. Both PAINTBRUSH and AUTOPLOT must swap segments with the 128K program memory size. The remaining memory not allocated to the application is used for application data and may be allocated and freed dynamically.

The application must load and free its own segments. Segments are freed in a stack-like manner. During execution, the application swaps segments into the allocated application memory. Segments may need to be freed to make room for new segments. The 256K program memory configuration avoids the need for swapping segments, removing the delay of reading from the flexible disc.

-Jim Davis
-Diane Rodriguez

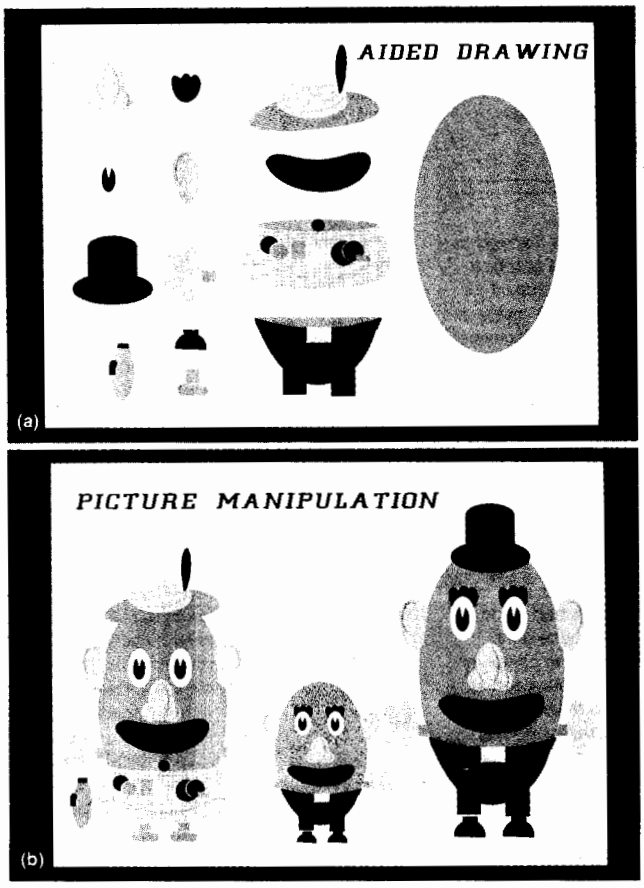


Fig. 4. *Picture editing (copy, move, scale, etc.) was done on (a) to arrive at (b).*

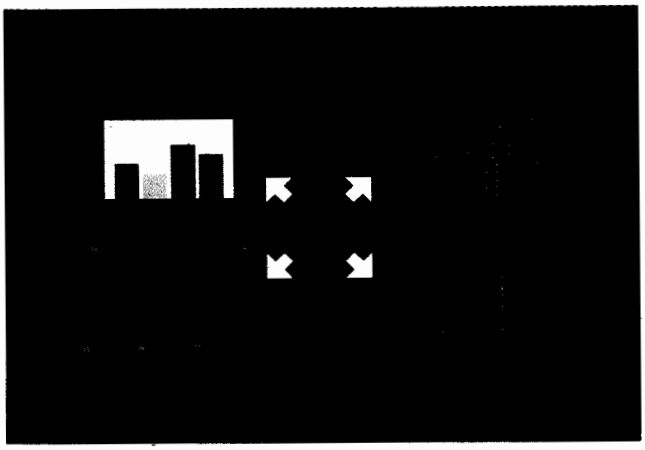


Fig. 5. *Four pictures were combined to form this one.*

names and vector list names that conflict with names in the existing picture are renamed, and the user can select whether or not to restore the color palettes that were used to create the picture.

PAINTBRUSH also allows users to plot to HP-IB and RS-232-C/V.24 plotters. Before plotting, the user can set terminal configuration parameters for plotting on 8½×11-inch paper or transparencies by selecting one softkey. Alternatively, the user can set each terminal configuration parameter as desired.

In addition to this input/output level, PAINTBRUSH users have access to the **DEVICE CONTROL** key. By pressing this key, the user can interface to any terminal-supported device (e.g., printers, cameras, discs, etc.).

Authors

September 1983

3

Sharon O. Mead

Born in Philadelphia, Pennsylvania, Sharon Mead earned a BA degree in mathematics from the University of Texas in 1974 and an MS in computer science from Purdue University in 1976. She came to HP in 1978, worked on the file system for the HP 2700, and then became project manager. Sharon is a member of the Association for Computing Machinery (ACM) and SIGGRAPH. She is married, lives in Saratoga, California, and enjoys horseback riding and vegetable gardening.

Catherine M. Potter

Cathy Potter received her BSEE in 1972 and her MSEE: Computer Engineering in 1977, both from Stanford University. She joined HP in 1972, where she initially worked on CAD tools, and then participated in the software development for the 8450A Spectrophotometer. As a 2700 team member, she has worked on the operating system, math library, and firmware development tools. Cathy lives in Los Altos, California and in her spare time she reads, jogs, and dabbles in crafts.

William R. Taylor

Bill Taylor attended the University of California at Berkeley, where he consecutively received BSEE and MBA degrees in 1974 and 1976. He was involved in the development of the HP 2700 minifloppy controller and external video interface. Bill, his wife, and their eleven-month-old daughter live in San Jose, California. He owns and plays a four-manual Wurlitzer theatre pipe organ and is a member of the American Theatre Organ Society.

Kenneth A. Mintz

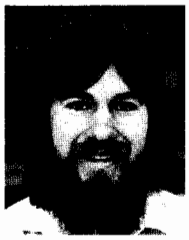
Since joining HP in 1971, Ken Mintz has worked on the operating system for the HP 3000 Computer, the alpha kernel system for the HP 2700, and software development for personal computers. He is a member of ACM and holds a BA degree in computer science awarded by the University of California in 1971 and the MSEE degree (1979) from Stanford. Born in Madison, Wisconsin, Ken is married, has two children, and lives in Cupertino, California. He is interested in photography and software methodologies.

9

Dale A. Luck

Dale Luck was involved in picture file and graphics firmware development for the HP 2700. He received a BS in computer science from Michigan Technological University in 1979 and is a member of the IEEE, the ACM, and SIGGRAPH. Dale is interested in artificial intelligence and the future. His hobbies are shooting pool, volleyball, stamp collecting, and fast motorcycles, airplanes, and cars.

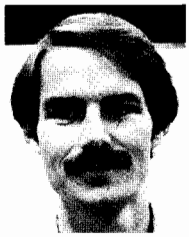
Robert R. Burns



Bob Burns joined HP in 1979 after receiving a BSEE degree in computer science and engineering from Rice University in Houston, Texas. He has been a member of the design team for graphics firmware development. Bob and his wife live in Santa Clara, where he is the secretary of the local Baha'i community and teaches Baha'i children's classes. He enjoys volleyball, motorcycling, and photography.

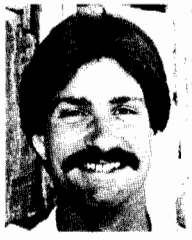
15

Craig W. Diserens



Craig Diserens grew up in Texas and graduated from Rice University in 1978 with a double major in electrical engineering and mathematical sciences. He received an MSEE degree from Stanford in 1982. He joined HP in 1978 and became involved in the development of the HP 2700 Terminal, designing the color mapper, power supply, and video amplifiers. He's now an R&D project manager for personal computer products. Craig plays trumpet for the Stanford Savoyards, who perform Gilbert and Sullivan operas. Living in San Jose, he likes skiing and bicycling.

Curtis L. Dowdy



Curt Dowdy joined HP in 1981 after earning a BS degree in electrical engineering from the University of Missouri at Rolla. He was responsible for completing the development of the HP 2700 graphics controller and graphics image memory. Curt is a member of the IEEE and lives in Sunnyvale, California. He enjoys snow skiing, backpacking, listening to jazz, and driving and cranking a wrench on his Jensen-Healey sports car.

18

Geoffrey G. Moyer

Geoff Moyer received a BS degree in electrical engineering from Cornell University in 1980. After graduating he came to HP and became involved in the HP 2700 monitor and EMI development. Originally from Haverford, Pennsylvania, Geoff is married and lives in Palo Alto, California. He spends his free time roller skating, going to the theatre, working on his house, and making leaded glass windows.

Paul G. Winninghoff

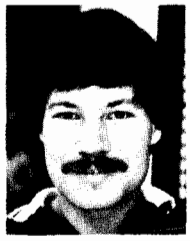
Paul Winninghoff has been with HP since 1964. His work in digitally controlled transmission-impairment measuring resulted in a patent, and he has been on the design team of several HP products including the 8552A Spectrum Analyzer, the 8442A Tracking Generator/Counter, the 4900 Series of TIMS, and the HP 2700. Paul has coauthored three previous HP Journal articles and teaches engineering technology classes at a local college. Educated at Montana State University (BSEE degree in 1962 and MSEE in 1963), he is married, has two daughters, lives in Sunnyvale, California, and is building a home computer from scrap parts.

Mark Hanlon

Born and raised in New Shrewsbury, New Jersey, Mark Hanlon attended the University of Arizona, earning a BSEE degree in 1977 and the MS in 1979. After coming to HP in 1980, he developed the monitor system for the HP 2700 and is currently working on displays for personal computers. Mark lives in Sunnyvale, California. His leisure activities include boardsailing, camping, cross-country skiing, woodworking, and sailing.

22

Stephen P. Pacheco



California native Steve Pacheco received a BA degree in computer science from the University of California at Berkeley in 1975. With HP since 1978, he contributed to the I/O system firmware for the HP 2700. Steve is single and enjoys silk screen printing, painting (his favorite color is pink), bicycle riding, and collecting unusual toys.

Edward Tang

Ed Tang, software project manager on the HP 2700, attended the University of California at Berkeley and received the BSEE/CS degree in 1970 and the MSEE/CS in 1973. A member of IEEE and ACM, he has taught computer science classes at San Jose State University. Ed is married and is a board member of the West Valley Chinese Language School. He enjoys woodworking, photography, racquetball, and being with his family.

Paula H. Ng

A native of San Francisco, Paula Ng first came to HP in 1978. After graduating from the University of California at Berkeley in 1979 with a double major in computer science and mathematics for teachers, she began working on the file system and device control of the HP 2700. She and her husband live in San Jose, California, where she tutors high school students and is a member of the Chinese American Women's Club. In her free time Paula enjoys tennis, calligraphy, softball, sewing, and UC-Berkeley football games.

Otakar Blazek

A native of Pilsen, Czechoslovakia, Oty Blazek has been with HP since 1972. He worked on an I/O card for the HP 3000 Computer and designed the 2640A Terminal's keyboard and the graphics controller for the 2648A Terminal. On the HP 2700 project, he designed the alphanumeric video controller and modified the color mapper. Oty earned MS degrees from the Technical University of Pilsen in 1963 and the University of California at Berkeley in 1971. Married and living in Sunnyvale, California, his leisure activities are tennis, diving, and reading novels.

Thomas K. Landgraf

Tom Landgraf was born in Los Angeles and attended the University of California at Davis, receiving a BS degree in electrical engineering in 1976. In 1981 he earned the MS degree in computer science at the University of Santa Clara, where he is currently working toward an MBA. With HP since 1979, Tom was involved in the design of the main processor for the HP 2700. He is married, has an 8-month-old daughter, coaches Little League baseball, and is interested in skiing, photography, cooking, gardening, and model railroads.

25

Michael R. Perkins

Mike Perkins received a BS degree in applied mechanics and bioengineering from the University of California at San Diego in 1981. His background was in physical oceanography when he joined the HP 2700 design team as a development engineer. Mike is single, lives in Santa Clara, California, and enjoys scuba diving, volleyball, woodworking, and backpacking.

Charles W. Andrews

Charles Andrews joined the Andover division of HP in 1972 after earning his BS degree in electrical engineering from Massachusetts Institute of Technology. He was the project manager for the HP 2700 logic and test systems. Originally from Atlanta, Georgia, Charles is married and now lives in Sunnyvale, California. His hobbies are photography and motorcycle riding.

Susan Snitzer

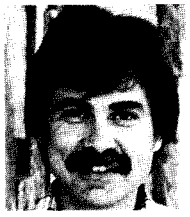
As a development engineer at HP, Susie Snitzer has been involved with interface software design. She is a graduate of the State University of New York at Stony Brook (1981), where she earned a BS degree in computer science while supporting herself as a waitress. Susie lives in San Francisco. Her leisure activities are backpacking, volleyball, skiing, and travel. She also likes to cook.

28

Mark A. Della Bona

Mark Della Bona received a BSE degree in 1976 and an MS in product design in 1977 from Stanford University, where he now teaches a class in mechanical engineering. He has worked on several fiber-optic datacom products and is currently a project manager. Before joining HP he was involved in the design of the space shuttle galley and zero-G toilet. The results of his work appeared in a published paper. Mark is married, holds a private pilot's license, and enjoys backpacking and cross-country flying.

Dennis C. Thompson



Born in Quincy, Illinois, Denny Thompson studied mechanical engineering at the University of Utah, receiving a BS degree in 1971 and an MS in 1972. He is a member of the American Society of Mechanical Engineers (ASME) and has worked on calculator, plotter, and terminal projects at HP. His work has resulted in patents for plotter design and a sealable vial. Active in his church in Campbell, California, Denny is married, has two children, and enjoys woodworking, singing in choral groups, skiing, backpacking, and running.

Kenneth D. Boetzer

Before joining HP as a development engineer in 1980, Ken Boetzer worked as a counselor for high school dropouts and as a race car designer. He has BS degrees in biology from Arizona State University (1970) and in mechanical engineering from San Jose State University (1979). Ken teaches a 4H auto mechanics class in Scotts Valley, California, where he lives with his wife, who is a physician, their son, and a menagerie of dogs, geese, and chickens.

Badir M. Mousa

Born in Ramallah, Palestine, Bud Mousa attended San Jose State University and received a BS degree in industrial design in 1975. He worked as a design consultant for consumer, medical, and electronic products before coming to HP, where he has contributed to the design of terminal products, modems, graphics tablets, and monitors. Bud has taught an industrial design class at San Jose State, is married, has one daughter, and likes backpacking, camping, canoeing, photography, and woodworking.

31

John M. Perry

Born in Detroit, Michigan, John Perry received a BSE degree in 1977 and an MS in 1980 from the University of Michigan. He worked as an analyst specializing in graphics for a consulting engineering firm and then joined HP as a development engineer on the HP 2700 project. Living in San Jose, California, he is married, has one daughter, and enjoys photography and sailing on San Francisco Bay.

Stanley A. Balazer

Stan Balazer started with HP's San Diego Division in 1977 and came to the San Francisco Bay Area in 1979 after graduating from the University of California at San Diego with a BS degree in computer science. Stan has contributed to the development of several graphics software projects, including the applications software for the HP 2700. He lives in Santa Clara, California and enjoys computer animation, tennis, volleyball, and frisbee.

Jim L. Davis

Jim Davis attended California State Polytechnic University at San Luis Obispo and received a BS degree in computer science in 1981. He joined HP after graduating and began working as a development engineer on HP 2700 graphics applications. A native of Ventura, California, he uses his free time to pursue interests in karate, animation, woodworking, winemaking, bicycling, and tennis.

John R. Alburger

John Alburger is a project manager responsible for HP 2700 development tools and applications. He earned a BS degree in computer science and electrical engineering from the University of California at Irvine in 1977 and an MS in electrical engineering from Stanford University in 1980. Originally from La Canada, California, he now lives in Cupertino with his wife and daughter. His outside interests include scuba, swimming, and home remodeling.

Barbara A. Stanley

Barbara Stanley was involved in systems programming before coming to HP, where she has designed and developed graphics applications and user interfaces. She earned a BA degree in music from Brigham Young University in 1975 and an MS in computer information science from San Jose State University in 1981. A member of ACM, she performs with the HP choir and is continuing her education at Stanford University. Her hobbies are piano and racquetball.

Diane A. Rodriguez

Diane Rodriguez received a BS degree in computer science from Michigan Technological University in 1980 and is currently studying at Stanford University. She worked on graphics applications for the HP 2700. Married and living in San Jose, California, she is a member of ACM, active in her church, and enjoys outdoor sports, sewing, and board games.

HEWLETT-PACKARD JOURNAL

HEWLETT PACKARD

5953-8515