JANUARY 1981

HEWLETT-PACKARD JOURNAL

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

Contents:

JANUARY 1981 Volume 32 • Number 1

In this Issue:

Computer application programs tell a computer how to accomplish specific tasks. Typically, such tasks involve the processing of information that the computer finds stored in its memory or coming in from a terminal or other input device. The articles in this issue deal with ways of getting that information into the computer.

Featured on our cover is a new digital bar-code wand. You've seen bar codes on the labels of products at the supermarket. The bars and spaces in these codes represent numbers or letters, and the bar-code wand converts them into electrical signals a computer can read. It's a fast, reliable, relatively cheap and easy-to-use method of putting data or programs into a computer. The wand comes in two versions, Model HEDS-3000 (page 3) for general use, and Model 82153A (cover and page 11) for use with the HP-41C Calculator, featured in these pages in March 1980. HP-41C solutions books give you not only the program listings but also the equivalent bar code, so all you have to do to program the calculator is scan the pages of the book with the wand. Easy.

The article on page 15 describes another computer input device, Model 9111A Graphics Tablet. With this useful tool you can draw the computer a picture. For example, an engineer can draw circuit diagrams or mechanical structures and see them appear on a display or plotter as they are drawn. Compared to describing pictures to the computer in terms of coordinates, the tablet is a lot more convenient.

An article about a new data capture software package, DATACAP/1000, begins on page 25. DATACAP is a package of programs for HP 1000 Computers. Its purpose is to help manufacturing companies collect data from factories—information about inventories, work in process, time and attendance, distribution. Guided by DATACAP, factory personnel enter this data into HP data capture terminals using keyboards, cards, or bar-code wands. DATACAP receives this data, checks it, and stores it in a data base in the computer's memory, so that report-generating programs can process it and produce timely and accurate reports for management. DATACAP is designed to adapt easily to each individual factory's needs. It asks its owner questions to find out what those needs are and generates a tailor-made information-gathering system to meet them. This helps managers increase operating efficiency, minimize inventory investment, and improve customer service.

-R. P. Dolan

Handheld Scanner Makes Reading Bar Codes Easy and Inexpensive

This lightweight wand contains the light source, reflected-light sensor, and digital signal shaping circuitry needed for scanning bar-code patterns reliably.

by John J. Uebbing, Donald L. Lubin, and Edward G. Weaver, Jr.

BAR CODES ARE A RAPIDLY GROWING method of manual data entry that can be used as an effective alternative to keyboards. A bar code is a self-contained message with information encoded in the physical widths of bars and spaces in a printed pattern. Hewlett-Packard's new HEDS-3000 Digital Bar Code Wand (Fig. 1) is a reliable interface between these printed bar codes and a digital decoding system. When the handheld wand is used to scan the bar code, it converts the light reflected from the printed bars and spaces into TTL or CMOS-compatible logic levels. The resulting digital signal is available for input to a digital decoding system.

The wand contains a precision optical sensor, an analog amplifier, a digitizing circuit and an output driver. The integration of the emitter, detector, and optics of the optical sensor into a single package makes the scanner rugged and reliable. The output of the sensor is proportional to the reflectance of a 0.2-mm (0.008 in) diameter spot in front of the opening in the wand tip. The sensor signal is amplified and converted into a logic-level output by a circuit contained in the wand body. This output is a logic high (1) level when the sensor is looking at a black bar and a logic low (0) level when it is looking at a reflecting white space. The output of the wand is connected to the user's digital processor, which typically measures the time intervals corresponding to the widths of these bars and spaces as the wand is scanned over the bar code. The user's decoding algorithm can then decode these time intervals into binary, numeric, or alphanumeric information depending on the bar-code format. Parity and check-sum information can be used to verify that the read operation was error-free before the information is entered into the computer.

Fig. 1. *The HEDS-3000 Digital Bar Code Wand contains all of the components necessary to convert a printed bar-code pattern into a digital signal for use in data processing.*

The optical sensor has a spot-size resolution that allows bar-code widths as small as 0.3 mm (0.012 in) to be read reliably. This resolution is ideal for dot-matrix printed bar codes. In addition, the 700-nm wavelength of the sensor light source enables sensing of many colored bar codes, although the HEDS-3000 is primarily intended for black-and-white patterns.

The circuit in the HEDS-3000 bar-code scanner uses a push-to-read switch to save power in battery-operated systems. Another battery-oriented feature is the wide range of operating voltage. The wand circuit is designed to use a single power supply within the range of 3.6 to 5.75 volts. At maximum voltage the wand will draw less than 50 mA when the switch is depressed. The circuit's open-collector transistor output allows the wand to interface with either TTL or CMOS circuits.

The HEDS-3000 is packaged in a rugged ABS-plastic case. A strain-relieved one-metre cord on the wand is terminated in a nine-pin D-style subminiature connector with an integral squeeze-to-release retention mechanism. The low-friction tip unscrews for cleaning the sensor window or for replacement in the event of excessive wear.

A key specification of the HEDS-3000 is the accuracy with which the wand can measure the bar and space widths of bar-code patterns. This width-error specification is compatible with the specifications of bar-code printers to allow the system designer to evaluate the trade-offs in the design of a bar-code system. The wand can typically measure the width of the first bar in a code pattern within 0.1 mm (0.004 in) and the interior data bars and spaces with an accuracy of 0.05 mm (0.002 in). The wand is designed to read bar codes in all handheld orientations within a cone of

30° from the normal to the bar-code pattern. The wand will also operate over hand-scanned speeds ranging from 76 to 760 mm/s (3 to 30 in/s) and over an operating temperature range of 0 to 55°C (32 to 130°F).

Reflectance Sensor

The high-resolution reflectance sensor that is an integral part of the HEDS-3000 Bar Code Digital Wand is also available as a separate component (HEDS-1000) for use in other sensing applications. These include pattern recognition, object sizing, optical limit switching, tachometry, defect detection, dimensional monitoring, line location, paper-edge sensing and bar-code scanning.

To be useful in a low-cost, portable bar-code scanner the reflected-light sensor must be able to detect 0.25-mm (0.01-in) wide bars and spaces with high sensitivity, have low power consumption, exhibit good reliability, and have low manufacturing cost. The design of the HEDS-1000 meets all of these requirements.

Several optical configurations were considered for the reflectance sensor arrangement, including half-silvered mirrors, coaxial source and detector arrangements, and separate packages for the source and the detector. The bifurcated side-by-side approach (Fig. 2) was selected because it provides a compact structure that fits into the tip of a wand and allows both the source and detector to be mounted on the same substrate. One of the possible drawbacks of this configuration is that stray light reflected from the split lens system can generate a photocurrent when there is no object to be sensed. This is judged not significant because of the low level of the stray light relative to the signal and because the signal conditioning circuit can compensate for the presence of stray light.

To obtain high sensitivity, a large aperture is needed so that a substantial amount of light is focused on the bar-code pattern and a large portion of the reflected light can be collected and focused on the detector. Early experiments determined that spherical lenses with the necessary aperture exhibited spherical aberrations that were too great to allow 0.25-mm width resolution. Thus, to provide a numerical aperture of 0.4 and a sharp focus, an aspheric lens design was used. The two plastic lenses have hyperbolic surfaces and are precision molded together as a pair.

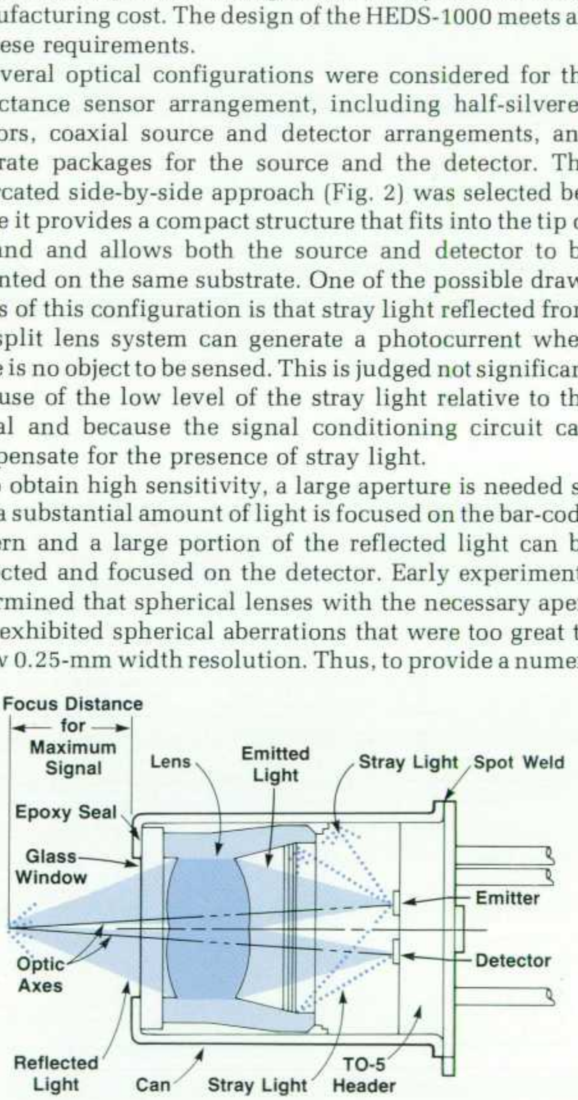

Fig. 2. *The HEDS-1000 High-Resolution Reflectance Sensor used in the HEDS-3000 is also available as a separate product. This sensor contains both an emitter and detector in a single package. A bifurcated lens design focuses the light from the sensor's emitter onto the area to be sensed and focuses the reflected light back onto the detector area in the sensor.*

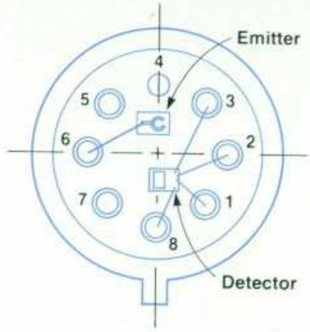

Fig. 3. *The emitter and detector chips in the HEDS-1000 are mounted on a standard TO-5 header as shown.*

The source is a small (0.18-mm diameter) light-emitting diode (LED) with a wavelength of 700 nm. The wavelength of the LED was chosen to provide the best light-generating efficiency combined with the ability to sense dye-based ink patterns with good contrast. The detector consists of a silicon photodiode and transistor integrated on a single chip. The reflectance signal can be obtained directly from the photodiode or through the integral transistor configured as a high-gain amplifier.

One half of the lens system focuses the light from the LED onto the paper. The other half focuses the reflected light back onto the integrated photodetector chip. The area of the photodiode is somewhat larger than the LED to compensate for assembly tolerance and to improve the depth range over which good sensing can occur. Letting the LED size determine the resolution in this way minimizes the LED power consumption for a given photocurrent level. This is important for battery-powered operation.

A standard TO-5 header used for the substrate provides a compact method for bringing out the leads (Fig. 3). Gluing the lens directly to the header was first evaluated, but the moisture-fogging and temperature-cycling test results were inferior with this arrangement. The configuration adopted uses a tall metal can with an adhesive-bonded glass window for good moisture resistance. The molded bifurcated plastic lens pair is bonded to the can with a soft silicone adhesive for good temperature cycling performance. Both the LED and the detector are die-attached with a eutectic alloy to the header using a precision collet tool that attaches the chips with placement accuracy better than 0.05 mm.

Each HEDS-1000 Reflectance Sensor is individually tested by an electrical test system controlled by an HP 9825A Computer/Controller.[1] The signal levels, transistor parameters and other data sheet values for the HEDS-1000 are guaranteed by this system.

Bar-Code Reader Circuit

The circuit in the HEDS-3000 is designed to convert the low-level analog signals from the photodiode in the sensor module to a compatible logic level that can be easily interfaced to digital systems.

If the optical sensor module can resolve widths much narrower than the minimum bar width and there is a great

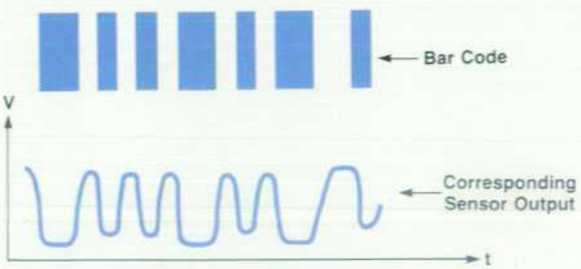

Fig. 4. *A typical bar-code pattern and corresponding waveform resulting from a scan of the pattern.*

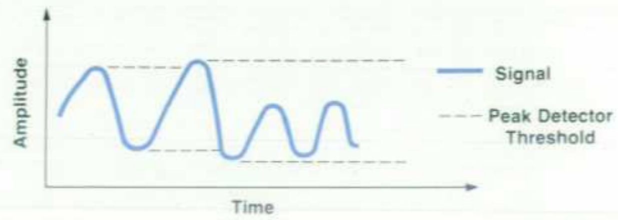

Fig. 5. *When the signal maxima and/or minima change as shown, the peak detectors must be reset frequently to correctly detect signal peaks.*

deal of contrast between bars and spaces, it would appear that only a comparator connected to the photodiode would be needed to perform the digitizing function. Unfortunately, this is not often the case and some type of circuit is required to determine where a bar ends and a space begins. A typical analog waveform from a bar-code scan is shown in Fig. 4. The reason why the signal changes gradually when going from a dark to a light region is that the finite spot size viewed by the sensor integrates the light and dark areas.

There are several techniques for determining precisely where a light-to-dark or dark-to-light transition occurs. One approach is to differentiate the signal, because the maximum rate of change occurs at the transition. The disadvantage of this technique is that it is very sensitive to both electrical and optical noise and is heavily dependent on scan rate. Some designs use ac coupled clipping circuits that amplify the signal to the point where it looks like a pulse train. This approach is very sensitive to changes in average signal value and does not digitize accurately enough for many requirements.

A potentially accurate technique is to detect the positive and negative signal peaks and set a threshold halfway between the two peak values. This point corresponds to the sensor viewing area being positioned half on the bar and half on the space. This works well if the signal level remains constant or the signal maxima increase or the signal minima decrease. In a typical scan, this is not the case because the height above the bar-code pattern (tag) and the wand angle can vary considerably during the scan, which changes the dc level as well as the modulation amplitude. If this technique is used, a means for resetting the peak detectors after each transition is needed to allow for the case of decreasing signal maxima or increasing minima (Fig. 5). Previous solutions of this problem have either compromised performance by using rapidly decaying peak detectors or have required the extensive use of digital circuitry to perform this function.

Another problem with peak detection, and some other schemes as well, is how to handle static conditions when there are no transitions and reliable references as to what is black and what is white. Arbitrary logic default states and fixed black-to-white thresholds could be used. The problem is that arbitrary states will not always be correct and fixed thresholds are subject to too many error sources, such as ambient light and supply voltage variations, sensor and amplifier circuit drift, and manufacturing tolerances.

The design of the bar-code reader circuit had to deal with these processing problems while complying with a number of other constraints. One major constraint is size, because the entire circuit has to fit in the wand body. To accomplish this with previous circuit approaches requires the design of a complex analog/digital integrated circuit. This was the approach initially taken before a simpler signal processing technique was found. Another constraint is the requirement for operation over a supply voltage range of 3.6 to 5.75 volts to be compatible with the HP-41C Calculator. Economic constraints dictate that the solution use low-cost components and assembly techniques. This eliminates approaches that require trimming for proper operation.

The circuit can be broken into three major blocks—amplifier, signal processor, and digital output. Fig. 6 shows the circuit schematic partitioned into these functions. The

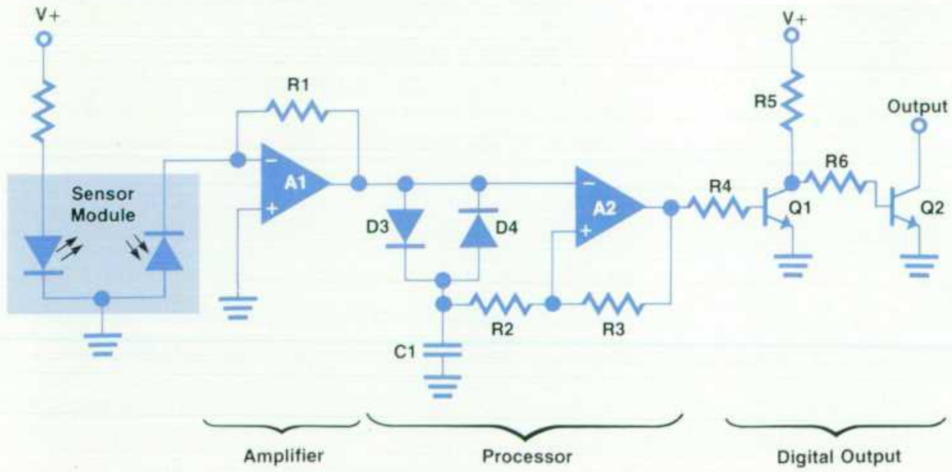

Fig. 6. *Circuit schematic for the HEDS-3000.*

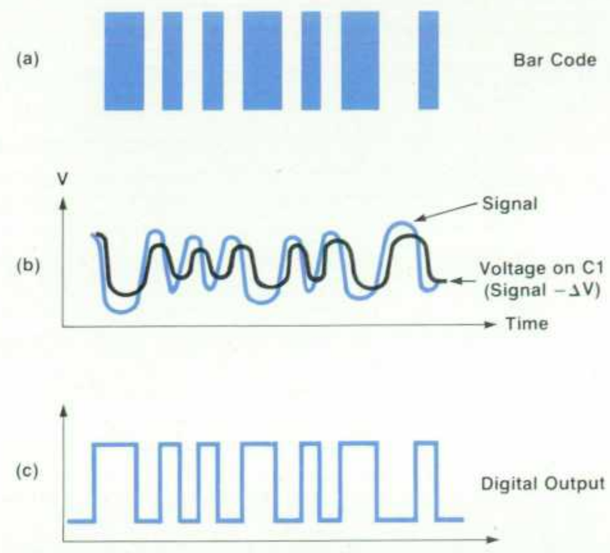

Fig. 7. *Operation of the processing circuit in the HEDS-3000. When the bar code in (a) is scanned, peak detection converts the resulting voltage across C1 (b) into a logic-compatible signal (c). Note: ΔV = diode forward voltage drop.*

amplifier increases the signal from the photodiode up to a processable level by converting a current on the order of 100 nanoamperes to a voltage of about 1 volt. An amplifier with a transimpedance of about 10 megohms is used to do this.

The processing circuit uses peak detection to decode bars and spaces. A simple positive and negative peak detector uses two diodes (D3 and D4) and a capacitor (C1). A positive voltage peak minus a diode drop is stored on C1. The comparator circuit (A2, R2, R3) compares this value with the output of amplifier A1. When the signal drops below the value stored on C1, the comparator changes state. As the signal goes through its negative excursion, D4 starts conducting and the negative peak voltage minus a diode drop is stored on C1. As the signal becomes more positive than the voltage on C1, the comparator changes state again. Diode D3 then conducts until the positive peak voltage minus a diode drop is stored on C1 again and the process repeats itself. Fig. 7 shows this operation. Notice that this circuit does not change state when the photodiode viewing area is located half on black and half on white. Instead it changes state before that point, introducing a leading phase shift which has little adverse effect. The advantage of this approach is that the peak detector always resets itself so that it can detect the peak of a decreasing signal maximum or an increasing signal minimum. It does this without requiring complex circuitry to track the signal correctly.

The comparator in the processing section provides two functions. First, it digitizes the signal by comparing the signal level to the voltage on C1. Resistors R2 and R3 provide hysteresis for this function to insure clean transitions in the presence of a noisy signal. The second, and much more subtle function of A2, R2, and R3 is to keep the output in the correct state under static conditions. As mentioned earlier, this is a problem with most decoding approaches. This scheme solves many of the problems associated with static conditions in a simple manner. It requires no absolute reference level or adjustments. The only constraint is to define what state the wand is in after power up. The circuit can be made to power up in either state depending on circuit details. Once the power-up condition is reached, the wand will correctly track all transitions dynamically and retain the correct state information statically.

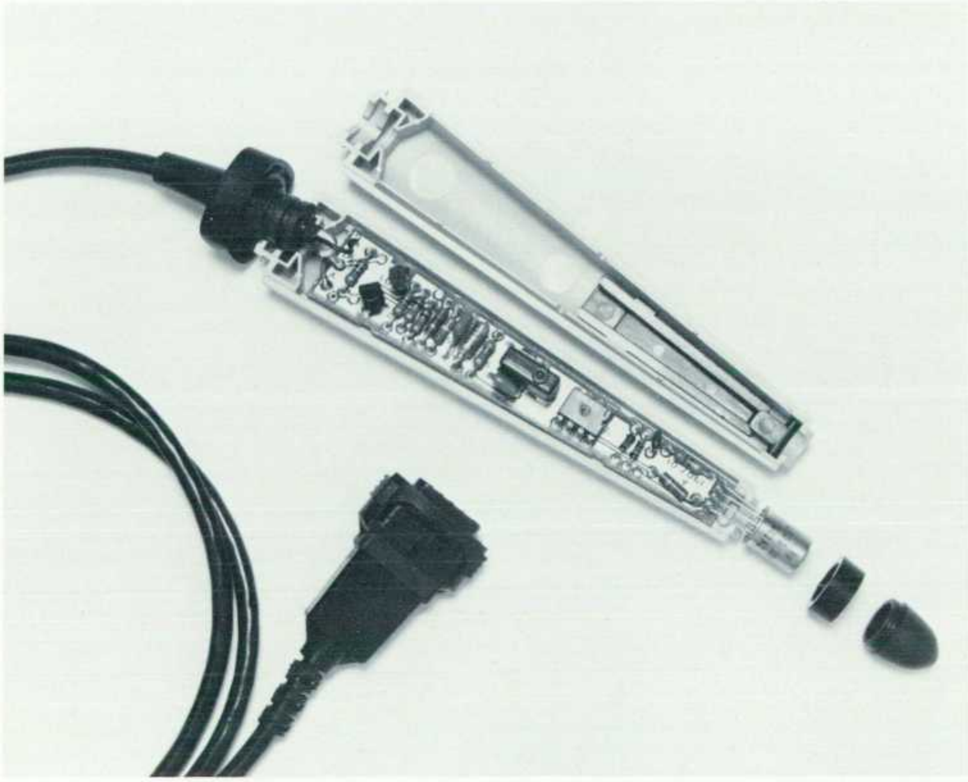

Fig. 8. *Wand case before assembly. The molded cylindrical body contains the sensor and the electronics and is held together by two rings.*

What Is a Bar Code?

Bar codes are messages with data encoded in the widths of printed bars and spaces on a piece of paper. If you consider that standard printed characters are two-dimensional, then bar codes are one-dimensional characters that have been stretched in the vertical dimension. From this perspective, bar-code scanning can be seen as the one-dimensional version of optical character recognition (OCR). The simplicity of scanning and decoding one-dimensional patterns is one main reason that bar-code systems are more prevalent than OCR systems and are expected to remain so for some time. An analogy for how bar-code systems work is to consider a printed bar code as a pulse-code-modulated (PCM) signal where linear distance on the paper is equivalent to time and the white and black reflectance levels of the bar-code pattern (tag) on the paper are equivalent to the high and low logic levels of the electrical signal.

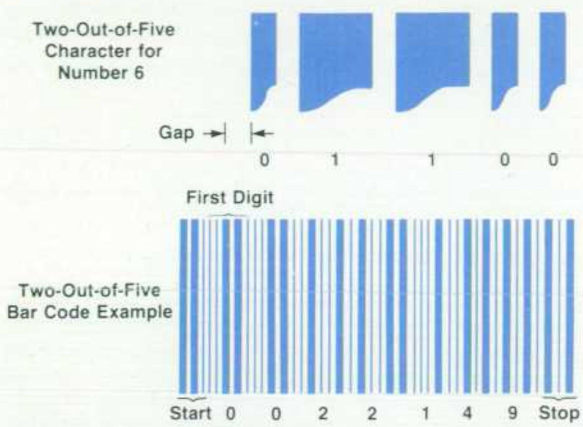

Fig. 1. *Illustration of two-out-of-five bar code. Each character is represented by a combination of five bars that are each one or three units wide (see Table I).*

The number of different possible coding schemes for bar codes is endless. However, the majority of schemes now in use can be classified as two-level codes. In a two-level code a wide bar or space represents a binary one and a narrow bar or space represents a binary zero (or vice versa). Usually the first two bars of a tag are used to define the initial value of a narrow width, then all bars and spaces read by the wand are compared to this standard value and assigned values of either one or zero depending on their respective widths.

Variations of the two-level code include versions where only bars carry information or where groups of bars and/or spaces (e.g., five bars and four spaces) represent single coded characters with internal parity checks. Bar codes can be read either bidirectionally or from one direction only. Almost all codes include a checksum digit encoded at the end of the bar code to provide security against improperly decoded characters. Code 39™ (developed by Interface Mechanisms) and two-out-of-five bar codes are examples of alphanumeric and numeric two-level character bar codes. Paperbyte™ and the HP-41C bar codes are two-level binary codes. Another common bar code is the four-level Universal Product Code (UPC) that is used for identifying grocery products. In this code a numeric character is defined by two bars and two spaces. Each bar and space is either one, two, three, or four modules wide and the total character width is constrained to be seven modules wide (a module is a unit of bar-code width). Security of the UPC code is further insured by internal parity checks and a checksum digit encoded at the end of the tag.

To get a better idea of exactly how a bar code works let's examine the two-out-of-five bar code more closely. Two-out-of-five code is a numeric code with ten digit-characters, a start symbol, and a stop symbol. The version described here is a

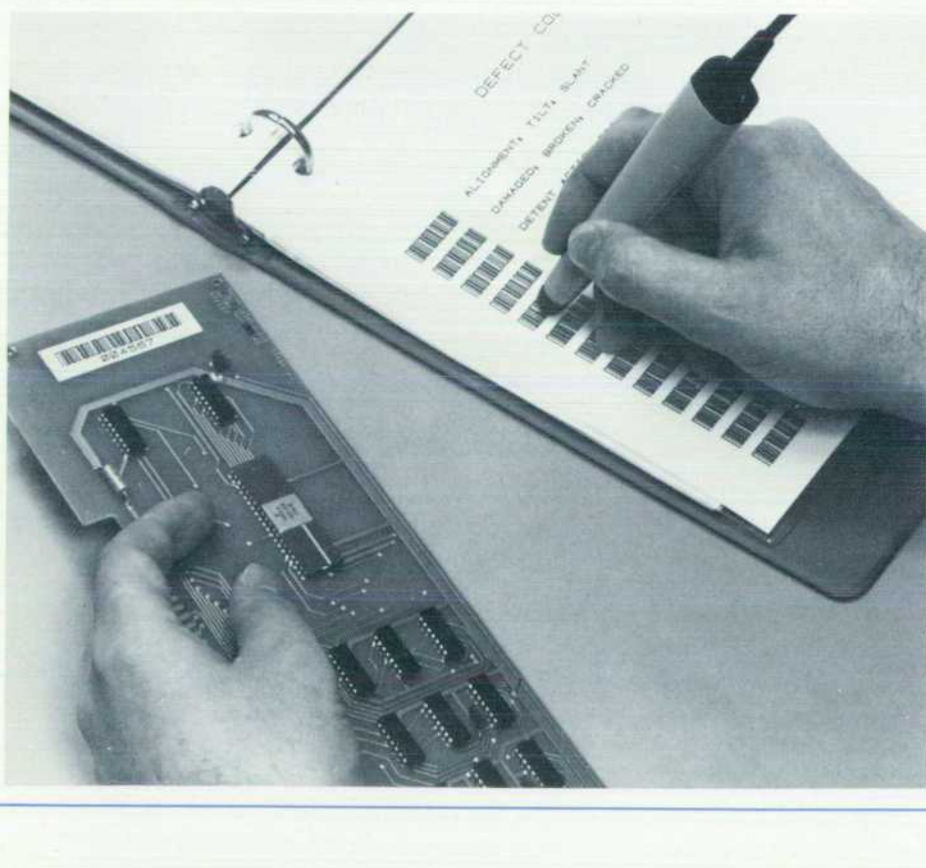

Fig. 2. *In this industrial application each assembly is identified by bar code and the QA defect code is read directly by a bar code wand from a code index, eliminating copying errors.*

Table I

Two-out-of-five bar-code character decoding. (0 = narrow bar, 1 = wide bar, M = margin, a very wide space at the beginning and end of a tag.)

Bar-Code Pattern	Character
00110	0
10001	1
01001	2
11000	3
00101	4
10100	5
01100	6
00011	7
10010	8
01010	9
M110	Start
010M	Stop

bar-only code with no information in the spaces. Narrow bars represent zero bits and wide bars (typically three times the width of the narrow bars) represent one bits. Each numeric character is a set of five consecutive bars of which two bars out of the five are wide. The character table and coding for this code is shown in Table I. A typical character and tag for two-out-of-five bar code is shown in Fig. 1. There are typically three levels of self checking that insure that the decoded two-out-of-five data is valid before it is entered into the computer. First, each character is checked to see that there are two and only two wide bars. Second, there must be an integer multiple of five bars between the start and stop symbols. Any extra or missing bars will cause the decoding system to void the read. Finally, the last character should be a software-determined checksum number to verify that the decoded data is the same as the encoded data. For example, if the last digit of the sum of the data values is encoded as your checksum digit, then it should also be the last digit of the sum of the decoded data values. If it is not, then the read is invalid.

The main applications for bar codes are as alternatives to keyboards. Bar codes allow fast and accurate hand entry of small amounts of data by minimally trained operators. The commercial market for bar codes is currently divided into three major types of users. The first and most widespread application of bar codes is in point-of-sale (POS) computer systems in grocery stores. Merchandise labeled with bar codes is scanned at the checkout counter to automatically enter the price and item on the customer's bill and update the store's inventory. This system allows faster checkouts, fewer errors, and more effective inventory management. The second major application of bar codes is in industrial data entry (Fig. 2). This category includes warehouse inventory control, identification of assemblies for process monitoring or work scheduling, and remote data collection. The purpose of bar codes in these industrial environments is to provide a simple error-proof means to hand-enter data into the central computer. The third and newest area of bar-code applications is low-cost data entry for microcomputers. Computer software in bar-code form can be mass produced inexpensively by the printing industry and distributed to a broad base of users. Bar-code-formatted software can be used for programming appliances, intelligent instruments, or personal computers. The bar-code option on the HP-41C described in the article on page 11 and the Paperbyte™ bar-code programs published by BYTE Magazine are excellent examples of the use of bar codes for low-cost consumer-oriented data entry.

If the voltage on C1 is greater than the output of A1, the output of A2 will switch to a high state. If the signal then remains constant, (static condition) and if R3 is not connected from the output to the noninverting input, C1 eventually charges via D3, D4 and the input bias current of A1 to a level comparable to the signal and the output state is indeterminate. With R3 connected, C1 charges through R2 and R3 toward the output level of A2, keeping the output of A2 in the high state. If the voltage on C1 becomes less than the signal, the output of A2 switches to a low state and under static conditions R2 and R3 discharge C1 toward that level, keeping the output in the low state. The circuit thus functions as a latch when static conditions are encountered.

The digital output section consists of transistors Q1 and Q2 and resistors R4, R5, and R6. The transistors serve to buffer the output of A2 and provide logic-compatible levels. The output stage shown in Fig. 6 provides a logic zero output for a dark-to-light transition. If the inverse is desired, the output can be taken from the collector of Q1, and Q2, R5, and R6 can be deleted.

Development of the Wand Package

After considering a number of alternatives such as a rectangular-box-type reader and a molded tube wand structure, the basic configuration shown in Fig. 1 and Fig. 8 was developed. The slim tip allows the operator to see the code while it is being read. The long, somewhat bulky body not only accommodates the printed circuit board, but feels comfortable in the operator's hand, somewhat like the handle of a paint brush. The switch actuation plate is long and comes up the side of the wand body so that the switch can be actuated by the operator's thumb or forefinger even though the wand is grasped in a number of different ways. The case itself is constructed of two separate halves, with the printed circuit board fitting in between. To provide aesthetic lines for the product, the case halves are held together with two molded rings of an ABS-and-polycarbonate-plastic alloy for maximum impact resistance and good temperature range. A vinyl strain relief is molded onto the end of the cord. The detailing at the end of the strain relief fits into grooves in the clamshell halves. The strain relief and cord system have successfully passed one thousand cycles of heavily loaded 180°-bend tests. Fig. 8 shows the wand case before assembly.

The tip of the wand has a slim profile and still accommodates the reflective sensor. The hole in the tip is large enough so that paper dust and other foreign material does not accumulate inside the tip, but falls out in normal use. The tip is molded of teflon-filled acetal since this material gives the smoothest ride over the paper and suffers the least wear. When the tip does wear out it can be replaced easily because it screws into the wand body.

Specification and Testing of the Wand

The HEDS-3000 uses a new approach to specifying bar-code wand performance. Historically, bar-code wands were analog devices and previous specifications dealt with such analog signal characteristics as amplitude modulation and noise. A bar-code wand with a digital output requires definition of a new set of performance measures. This digital performance specification properly describes the effects of mechanical, electrical, and optical parameters upon the wand's performance. This product specification then defines the conditions under which the wand will read bar codes successfully in a customer's system. However, different customers are interested in different specifications for different types of systems. Each system has requirements based on the specific bar code, bar-code printer, and decoding algorithm chosen by the system designer. The effect of the wand on the readability of the bar code in each of these systems is slightly different.

The desire to specify the performance of the wand in a manner that makes sense to a system designer led to the concept of system width error. The width error is a modification of the specifications used to describe bar-code printing tolerances. Bar-code printers generally specify edge resolution. The width of a printed bar or space can vary from the desired width by the printing uncertainty of the edges. Since these edge errors are independent of the width of the bar, printer performance for any bar is specified by a width error (e.g., narrow bars = 0.25±0.05 mm and wide bars = 0.50±0.05 mm). Like bar-code printing errors, wand reading errors are primarily edge errors. The performance of the wand is characterized by the accuracy with which it can measure bar widths. The wand bar-width error can then be summed with the width errors of the bar-code printer to determine the total width error at the start of the system decoding algorithm. A paper analysis of system performance accounting for printer errors, wand errors, and the specific software algorithm can then be done. Further, by specifying the bar-width errors and space-width errors separately, the designer can create software to compensate for offset errors characteristic of both wands and printers. Bars that consistently appear wider and spaces that appear narrower are examples of offset errors. When the specification is in terms of width error, the system designer can easily understand the trade-offs in bar-code system design.

The width error is the difference between the calculated bar or space width and the optically measured bar or space width. A two-level black-and-white code on photographic paper is used as the standard tag for characterization. The bars and spaces of these tags are optically measured with a toolmaker's microscope. When this standard tag is read by a wand under test at a constant scan velocity, the wand-measured bar width can be calculated from the duration of the bar output level from the wand. The width errors are separated into bar and space errors and into maximum and minimum errors. Because the magnitude of the width error is dependent on the width of the preceding space(bar) as well as the measured bar(space), the width errors are also sorted into bar/space and space/bar categories to give the system designer a more complete description of the wand performance. This information can be used to analyze the functionality of the designer's decoding software. For example, the system designer can see immediately that decoding schemes comparing bars with bars and spaces with spaces will cancel the systematic or offset errors that make bars appear wider and spaces appear narrower, while decoding software comparing bars with spaces will magnify these errors. To provide as complete a picture as possible of the HEDS-3000 performance in a variety of operating conditions, the wand's width errors are characterized as a function of wand height, angle and orientation, tempera-

Edward G. Weaver, Jr.

Eddie Weaver received the BSEE degree in 1975 and the MSEE degree in 1977, both from Rice University. He joined HP shortly after and has worked as a development engineer in the Optoelectronics Division since then. His work has included the testing and characterization of the HEDS-3000 Wand. Eddie is the co-author of three papers on the generation of continuous-wave ultraviolet radiation and optical effects in nonlinear crystals. He is a member of the IEEE and the Optical Society of America. Eddie is a native of Cortez, Colorado and now lives in Sunnyvale, California. He enjoys travel, hiking, backpacking, and gardening.

Donald L. Lubin

Don Lubin was born in Cleveland, Ohio and grew up in Yonkers, New York. He has been with HP since 1973 and has led projects dealing with optocouplers, fiber optics and the circuit design for the HEDS-3000 Wand. Don is currently the section manager for optocoupler, emitter, and detector products development. He was awarded the BSEE and MSEE degrees by Rensselaer Polytechnic Institute in 1972 and 1973, respectively. Don and his wife make their residence in Los Altos, California. Outside of work Don enjoys photography, running, sailing, and collecting clocks.

John J. Uebbing

John Uebbing is the author of a variety of papers on photocathodes, magnifiers, hybrid substrates and other optoelectronic topics. He came to HP in 1973 and has worked on alphanumeric and monolithic displays, was the manager of packaging and emitter detector groups, and is now involved with R&D work on advanced displays. John's previous experience included work on III-V semiconductor photocathodes and electron spectrometers. He has a BSEE degree awarded in 1960 by the University of Notre Dame, a MSEE degree awarded in 1962 by the Massachusetts Institute of Technology, and a PhD degree awarded by Stanford University in 1967. John is a member of the International Society for Hybrid Microelectronics and the Society for Information Display. He is a native of Chicago, Illinois and he and his wife and two children live in Palo Alto, California. John is interested in religious education and enjoys sailing, backpacking, and playing bridge.

ture, scan velocity, minimum bar-width size, and supply voltage.

The width errors of every wand are tested in production before packaging and shipment. The standard photographic test tag is attached to a wheel rotating at a constant speed. The wand is fixtured into the specified test position relative to the tag and the output of the wand is measured by a time-interval-measurement circuit and fed into a 9825A Computer/Controller. The 9825A compares the wand-output bar and space widths with the stored values of the optically measured bar and space widths. The 9825A then calculates the width errors for all the bars and spaces in the test tag, reduces this data, and stores the relevant information. Besides this final performance test, each printed circuit board is tested before final assembly to guarantee frequency response, output levels, and functionality. Each optical sensor is also pretested to guarantee its performance before final assembly. Finally, wands sampled from production lots are tested to ensure product operating life and humidity resistance. The design is characterized for mechanical integrity (shock, strain relief, etc.), operating reliability (temperature cycling, operating life, humidity, etc.), and performance quality (electromagnetic interference (EMI) and width error over diverse operating conditions).

Acknowledgments

The authors would like to thank Perry Jeung and John Lee for their project leadership, Ed Liljenwall, Fred Goodman, Matt Stein, Jim Casciani, and Bob Teichner for package development, Carl Trautman and Ray Wong for circuit work, Nate Walker and John Dunse for manufacturing engineering, Walt Heinzer for reliability testing and John Sien and Julian Elliott for marketing.

Reference

1. D.E. Morris, C.J. Christopher, G.W. Chance, and D.B. Barney, "Third Generation Programmable Calculator Has Computer-Like Capabilities," Hewlett-Packard Journal, June 1976.

SPECIFICATIONS

HEDS-3000 Digital Bar Code Wand

POWER SUPPLY:
V_S: 3.6-5.75 volts
I_S: 50 mA maximum.
DATA OUTPUT:
LOGIC LEVEL: TTL and CMOS compatible
WIDTH ERRORS:

First Bar	0.1 mm typical
Interior Bar	0.05 mm typical
Interior Space	0.05 mm typical

SCAN VELOCITY: 7.6-76 cm/s.
ILLUMINATION WAVELENGTH: 700 nm
TEMPERATURE RANGE
OPERATING: 0°C to +55°C.
STORAGE: −20°C to +55°C.
DIMENSIONS: 133 × 23 × 20 mm (5.2 × 0.9 × 0.8 in). Cable 1 m long.

HEDS-1000 High-Resolution Reflectance Sensor

POWER SUPPLY:
V_d, V_c, V_e: 20 volts maximum, 5 volts typical.
I_{LED}: 50 mA maximum average, 75 mA maximum peak.
POWER DISSIPATION: 120 mW maximum.
PHOTOCURRENTS:
PHOTOCURRENT (white surface): 120 nA, typical.
STRAY PHOTOCURRENT: 20 nA, typical.
FOCAL PROPERTIES:
IMAGE SIZE AT FOCUS (distance for 10-90% response over black-white transition): 0.17 mm.
DEPTH OF FOCUS (to 50% of maximum photocurrent): 1.2 mm.
MAXIMUM SIGNAL POINT: 4.3 mm from front of can.
SOURCE PEAK WAVELENGTH: 700 nm.
TEMPERATURE RANGE:
OPERATING: −20°C to +70°C.
STORAGE: −40°C to +75°C.
PACKAGE: 8-pin TO-5 style package, 12.9 mm long.
PRICES IN U.S.A.:
HEDS-3000 Digital Bar Wand, $99.50 each in small (1-99) quantities.
HEDS-1000 High Resolution Reflectance Sensor, $28.75 in small (1-9) quantities.
MANUFACTURING DIVISION: OPTOELECTRONICS DIVISION
640 Page Mill Road
Palo Alto, California 94304 U.S.A.

HP Model 82153A Wand

PHYSICAL SPECIFICATIONS: Same as for HEDS-3000.
ELECTRICAL SPECIFICATIONS: Supplied with interface plug-in for use with HP-41C Calculator.
PRICE IN U.S.A.
82153A Wand: $125.
MANUFACTURING DIVISION: CORVALLIS DIVISION
1000 N.E. Circle Boulevard
Corvallis, Oregon 97330 U.S.A.

Reading Bar Codes for the HP-41C Programmable Calculator

by David R. Conklin and Thomas L. Revere III

A SPECIAL VERSION of the HEDS-3000 Digital Bar Code Wand is supplied to Hewlett-Packard's Corvallis Division for use in the 82153A Wand (Fig. 1), an accessory to the HP-41C programmable calculator. Corvallis Division attaches an interface module containing two integrated circuits—a custom wand interface chip, and a 4096-word microcode ROM. In this article we describe the wand interface chip, the bar-code formats recognized by the HP-41C, and uses for the bar code and wand in the HP-41C calculator system.

The wand interface chip is a CMOS integrated circuit that converts the electrical signals from the wand into binary data, apportions the decoded data into eight-bit bytes, stores the byte(s) for retrieval by the HP-41C, and interfaces with the HP-41C bus lines to transfer the data to the calculator's CPU. The interface chip is located in the HP-41C-compatible plug which is attached to the 82153A Wand by a cable. To read a row of bar code, the wand is scanned across the bar code to generate a time-varying electrical signal that corresponds to the widths of the bars and spaces. The bar-code encoding scheme represents a logic zero by a bar with a relative width of one unit and a logic one by a bar with a relative width of two units. All spaces are one unit wide. A row of bar code may contain up to 16 bytes of data.

Fig. 1. *The 82153 Wand accessory to the HP-41C Calculator provides easy entry of data and programs printed in bar code.*

Decoding is done by counting the HP-41C system clock cycles (~360 kHz) between space-to-bar and bar-to-space transitions and comparing the result to a reference derived from the counts for the previous bar and space. Because inherent wand width bias, acceleration, rotation and other scanning irregularities introduce error into bar and space counts, the definitions of a one and a zero must include significant margins to reduce the possibility of an erroneous decode. A logic zero is defined as any bar less than 3/2 times the unit width established by the previous bar and space while a logic one bar must be greater than or equal to 3/2 times the unit width. The reference is created by first adding 1/2 the count for the previous space to 1/2 the count for the previous bar (1/4 the count for the bar, if the bar was decoded as a logical one or two-units-wide bar). The result is then added to 1/2 of itself to create a 3/2 unit-width reference.

The logic used to decode and store bar codes in one of two identical 16-byte buffers is controlled by a 64-state ROM machine which has 16 instructions and eight branch qualifiers. The decoding algorithm is illustrated by the flow chart in Fig. 2. The tests for bars and spaces include a test for maximum count ($\geq 2^{14}$) that branches back to the start state if the count is exceeded.

Maximum and minimum acceptable bar widths are determined by the scan speed and the interface chip. The maximum count for a bar is limited to less than 2^{14} HP-41C clock cycles. For a clock period of 2.63 μs, this is equal to a duration of about 43 ms. If the scan speed is 76.2 mm/s (3 in/s), the maximum bar width is about 3.2 mm (0.125 in). The minimum bar width is set by the time required to decode a bar, establish a new reference and store eight bits in a buffer. Seventy-six clock cycles are required, corresponding to a minimum bar or space width of about 0.17 mm (0.007 in) at a high scan speed of 762 mm/s (30 in/s).

In addition to the data bars, the interface chip requires that a row of bar code (Fig. 3) have four additional bars—one unit-width bar at each end of a row of bar code to be used in conjunction with the adjacent space to establish an initial reference and another bar next to each reference bar to determine scan direction. The direction bar encountered first in a right-to-left scan is two units wide; in a left-to-right scan it is one unit wide. The direction bars enable the chip to determine which direction the user is scanning and therefore in what order the decoded data should be sent back to the system CPU. The CPU always receives the least significant bit (LSB) of the leftmost byte first. The interface chip accomplishes this by conditionally storing the de-

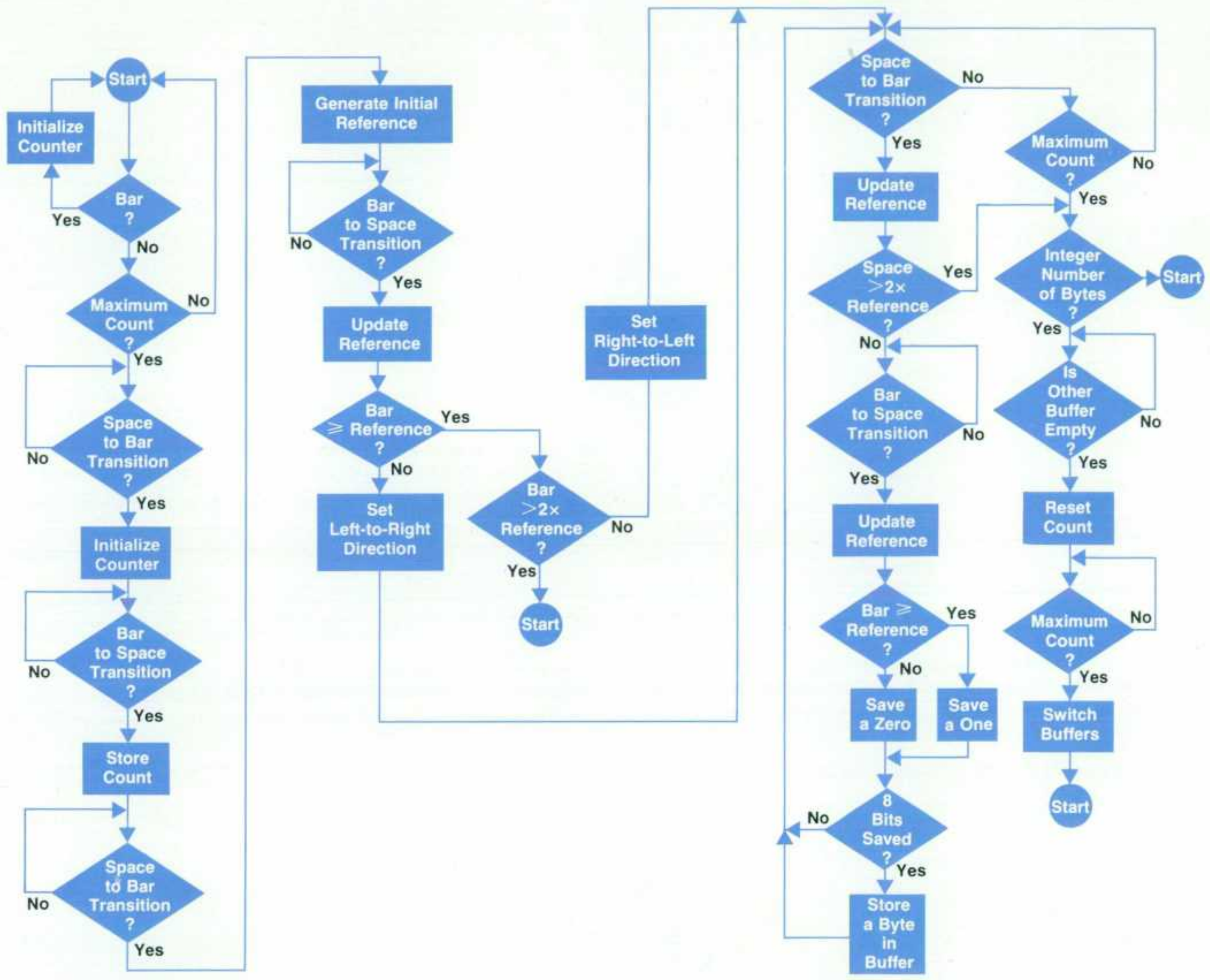

Fig. 2. *Flow chart of bar-code decoding algorithm used by the 82153A Wand interface chip.*

coded bytes in one of two sequences in a 16-byte buffer.

The location in which each byte is stored in a buffer is determined by an address pointer that is reset to the first location in the buffer at the beginning of each scan and is incremented if the scan is right-to-left and decremented if the scan is left-to-right. When a read operation is started by the CPU, and the scan is left-to-right, the pointer is reset to the first location in the buffer and is decremented for each subsequent byte. The pointer is not reset if the scan is right-to-left but is still decremented during readback.

During the time that decoded bar code is being stored in one of the buffers, the other buffer will send a byte from the location indicated by its pointer in response to an HP-41C CPU instruction requesting data from the interface chip. After a line of bar code has been successfully read and the other buffer has been emptied the newly filled buffer is allowed to communicate with the CPU. Two of the HP-41C's input flags are reserved for the wand. Because the wand chip uses the system clock for all internal functions, it prevents the calculator from returning to the clockless standby state by activating flag zero when the wand is turned on and pointed at a white surface and therefore presumably may be in the process of decoding bars. Flag two is used to signal the CPU that data is available and flag two will remain active until the buffer is emptied. The interface chip also wakes up the HP-41C from the off or standby states by pulling on the ISA bus line when the wand is turned on and pointed at a white surface.

Bar code generated for the HP-41C falls into one of four logical types—program bar code, data bar code, and bar code representing keystrokes (paper-keyboard bar code) or complete key phrases (direct-execution bar code). See Fig. 4 for a diagram of the bar code types. When the wand microcode sees a row of bar code only one or two bytes in length, the wand assumes that it is paper-keyboard bar code. One-byte rows have four bits for data and four bits for a checksum that is mirror symmetric to the data pattern. This convention makes one-byte rows immune to errors in decoding the direction bits. Two-byte rows have twelve bits of data and a four-bit checksum that is computed as a sum of the data in four-bit nibbles with end-around carry.

Rows of more than two bytes are assumed to be data,

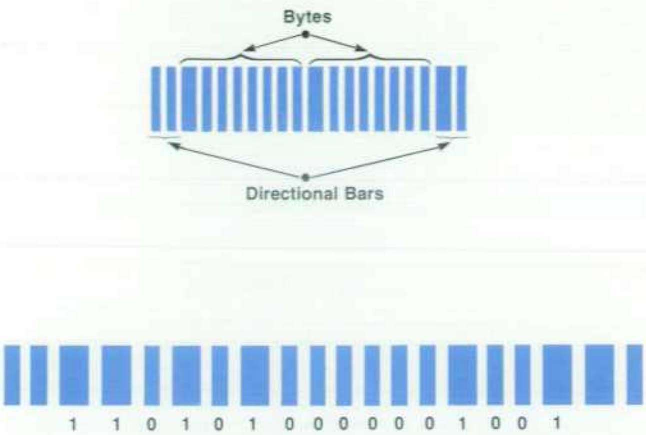

Fig. 3. *Basic format for bar codes read by the 82153A Wand in the HP-41C Calculator system.*

program, or direct-execution bar code. In each case, the leftmost byte is an eight-bit checksum and is followed by a four-bit type specifier. Program bar code includes a four-bit sequence number and two four-bit quantities that record the numbers of leading and trailing partial function code bytes (i.e., how much of the row is taken up with function codes that begin or end in other rows). The overhead—checksum, type, sequence number, and partial function code information—amounts to three bytes, which leaves up to thirteen bytes for the program itself. The sequence number, since it is four bits wide, only defines the sequence of the row within the surrounding sixteen rows; but this is adequate to warn the user when a row has been skipped or read twice. The checksum is a running eight-bit-wide sum with end-around carry of the current row and all preceding rows.

A data bar-code row may contain either a number or an alphanumeric string. The number may have up to ten digits of mantissa and two digits of exponent; the alphanumeric string may be as long as fourteen bytes.

Direct-execution bar code represents a complete key phrase (e.g., XEQ A or STO 12). After two bytes of overhead, the key phrase itself may be from one to nine bytes long.

Loading programs is the primary use envisioned for the 82153A Wand in the HP-41C Calculator system. The wand is less expensive than the card reader, although not as fast. It is much faster, more reliable, and less tiring than hand keying in a program from a program listing. To load a program with the wand, the user simply begins by scanning the program at the first bar-code row. It is not necessary to execute any function or do any other initialization beforehand. The medium of the printed page is widely available and inexpensive. A single standard sheet typically contains eighteen rows of bar code—the equivalent of both tracks of one magnetic card. This is comparable to the amount of space taken up by the printed listing of the program. It is our hope that in the future wherever calculator programs are printed in listing form, the bar code for the programs will also be printed (e.g., in textbooks, technical journal articles, newsletters, etc.).

Data bar codes make another important application possible. A number of large organizations—corporations and

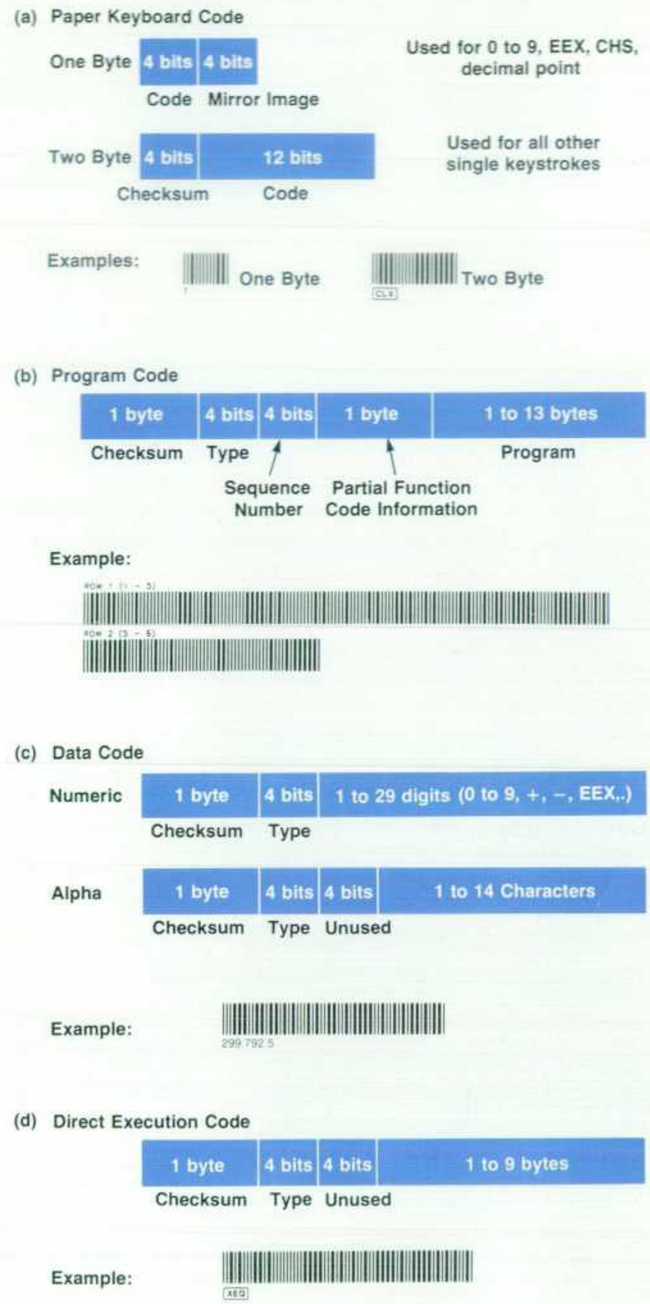

Fig. 4. *Four logical types of bar code are used by the HP-41C. (a) Paper keyboard code. (b) Program code. (c) Data code. (d) Direct execution code.*

government agencies—have standard sets of calculator programs developed for in-house use. Frequently these programs use data that is changed periodically. In such cases, the data can be printed and disseminated in bar code form.

The most novel applications for the 82153A Wand make use of the opportunity to mix machine-readable bar code with human-readable text in formats specifically adapted to the problem to be solved. The advantage in this sort of application is that the legends on the bar code can be written in the natural terms of the problem, and need not bear any resemblance to the technical meaning of the bar code to

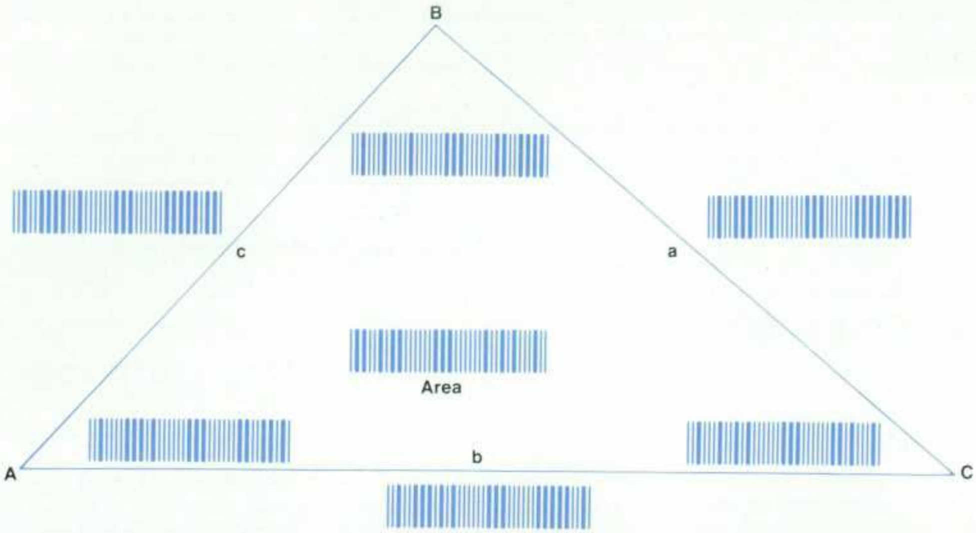

Fig. 5. *Problems can be easily solved by using bar codes to enter known values and to execute routines to determine the unknown value. An example for finding an unknown side or angle of a triangle is shown to the left.*

the calculator. In Fig. 5, this technique is used for an easy-to-use program for solving triangle problems. The user enters the known data about the triangle, indicating which part of the triangle it applies to by scanning the corresponding bar-code row. When all the known data is entered, the user queries the calculator for an unknown value simply by scanning the bar code for that part. The user never has to know that the bar-code rows on the diagram actually mean XEQ A, XEQ B, etc.

The paper-keyboard type of bar code derives its name from a paper keyboard provided in the box with the wand. The HP-41C calculator has available many more functions than keys, so a user occasionally must spell out the name of a function to invoke it from the keyboard. The paper keyboard, however, has bar-code rows for every function in the HP-41C catalog, and for the card reader, printer, and wand functions as well. A single sweep of the wand can replace a half dozen or more keystrokes when the user wants to invoke a function not currently assigned to a key.

During the project, it became clear to the design team that we would not be able to anticipate all the uses of the HP-41C/82153A Wand/bar code system. For this reason, a function (WNDSCN) was added that allows the HP-41C to read any bar code meeting the requirements of the interface chip. The data from the bar-code row is placed into registers 01 through n, one byte in each register. The WNDSCN function does not perform a checksum or any other consistency test. A submarine hunt game that makes use of this function is included in the owner's manual.

Acknowledgments

Bernie Musch conceived the product and provided inspiration and continuity for the project. Rich Whicker and Dave Lowe did the initial hardware design. Alan Peterson and Sheshadri Iyengar helped complete hardware development. Gaye Daniels, Steve Chou, and John Van Boxtel created the microcode. Jerry Hackett is the production engineer. Eric Henderson wrote the owner's manual. Tycho Howle was product manager. Other important members of the project team include: John Allen, Bob Dunlap, Earle Ellis, Alan Gill, Steve Gregg, Don Hale, and Judy Thompson. Special thanks goes to the PPC (a user's group) for help in testing the product, especially to Jake Schwartz for suggestions leading to the paper keyboard layout. The triangle program in Fig. 4 was adapted by Pam Raby from John Kennedy's program for the HP-67.

Thomas L. Revere III

Tom Revere joined HP in 1975 after working in the semiconductor industry for four years. He initially worked on IC design and is currently involved with continuing development of the HP-41C product system. Tom is a native of Mobile, Alabama and attended Brigham Young University, where he was awarded both the BESEE and MEEE degrees in 1971. He is a member of the IEEE and he, his wife, and five children live in Corvallis, Oregon. Tom's outside interests include home computing, camping, photography, and collecting science-fiction books.

David R. Conklin

Dave Conklin was born in Washington, D.C. and attended both the University of California at Berkeley, earning a BA degree in mathematics in 1967, and the University of Santa Clara, earning an MSCS degree in 1975. He worked for HP from 1973 to 1975 at the Santa Clara Division and joined the company's Corvallis Division in 1977 after working for a time elsewhere on process control programming. Dave has worked on the microprogramming for the HP-41C Calculator and is the project manager for the 82153A Wand. He is a member of the IEEE and the Association for Computing Machinery. Dave and his wife live in Monroe, Oregon.

A High-Quality, Low-Cost Graphics Tablet

It enables the user to interact easily with a computer graphics system to generate illustrations using predefined and user-defined shapes, point-to-point plotting, and continuous line drawing or tracing.

by Donald J. Stavely

THE HP 9111A GRAPHICS TABLET offers a new combination of features for the easy generation of computer graphics information. These features are contained in a low-cost human-engineered package (Fig. 1) and include high-level HP-IB* programming, comprehensive self-tests, and built-in softkey menu.

A graphics tablet is a peripheral device that provides a host computer with data corresponding to the position of a pen-like stylus relative to a surface, or platen. A pressure-activated switch in the stylus is the method used by the operator to inform the computer of a "picked" position.

Normally, the position data is used to manipulate a cursor on a graphics display device. This mode of operation, with the user's hand on the platen and eyes on the display, might sound awkward at first, but actually is quite natural. It often is the primary mode of graphical input in interactive graphics applications such as a printed-circuit, LSI, or mechanical design system.

A graphics tablet can be used in its own right for menuing applications, where the user picks items from a menu document placed on the platen. This menu can be thought of as a custom keyboard that can be easily understood by users with little or no technical training. The 9111A incorporates a built-in set of sixteen softkeys that are recognized by the tablet firmware. Program applications can use these softkeys for menuing or program control without X,Y coordinate analysis in the program. The ability to place a document on the platen surface also allows the tablet to act as a digitizer in applications where the high accuracy of the HP 9874A Digitizer[1] is not required.

*Hewlett-Packard's implementation of IEEE Standard 488 (1978).

Graphics Tablet versus Digitizer

The 9111A was designed using the basic technology developed for the 9874A Digitizer. In fact, the 9111A HP-GL command set is fully compatible with a large subset of the 9874A commands. While this product design used many of the technical aspects of the 9874A, new contributions in the areas of human engineering and low-cost design were required to satisfy the special requirements of an interactive graphics device.

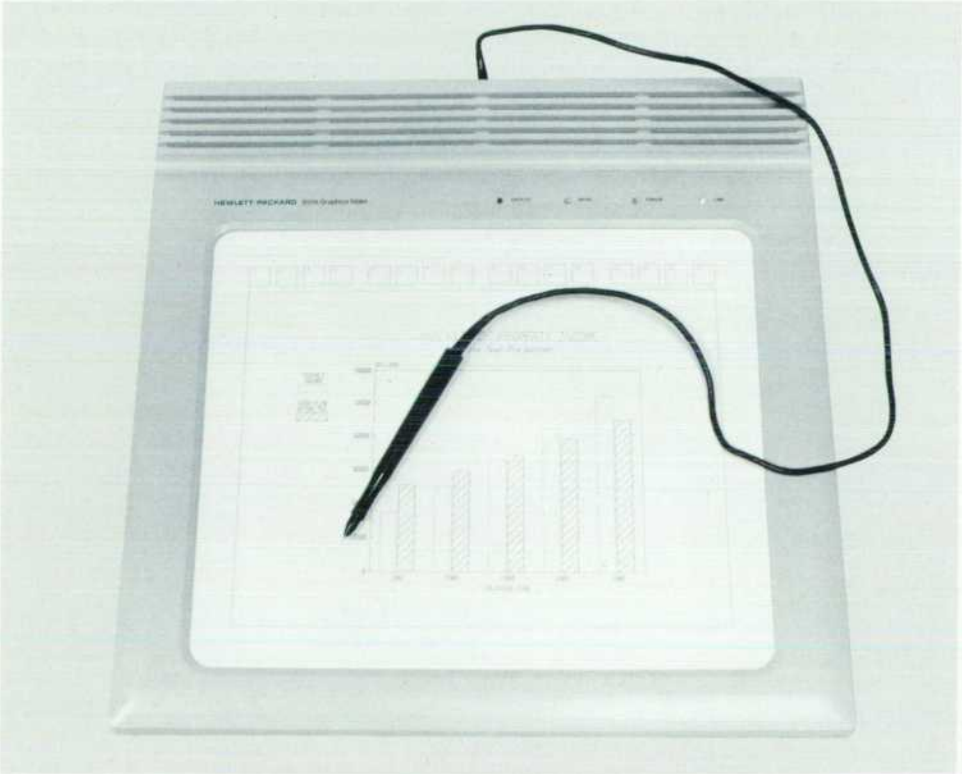

Fig. 1. *The 9111A Graphics Tablet provides easy user interaction with computer graphics systems. The stylus can also select shapes or functions from a menu overlay in addition to creating or modifying graphics images. Sixteen regions on the platen are designated for use as user-defined softkeys.*

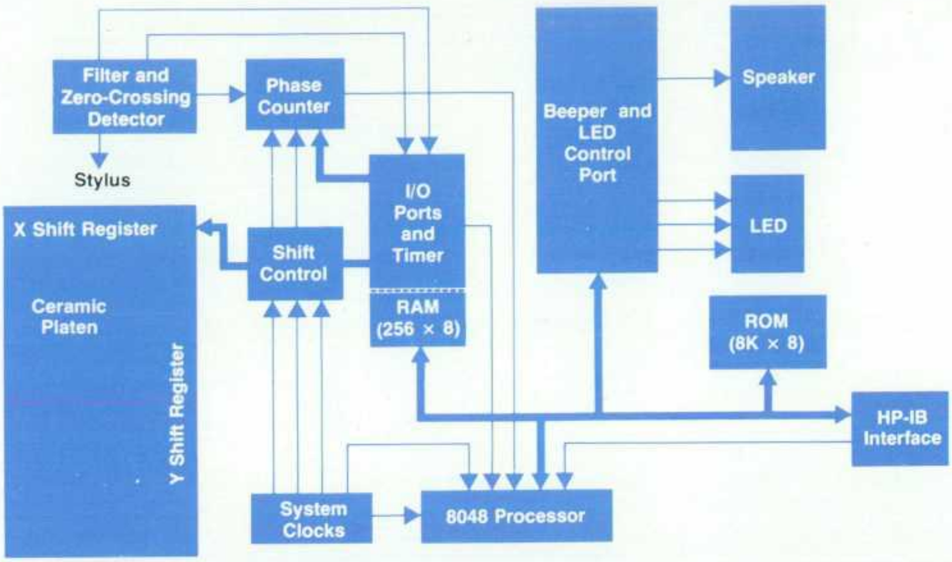

Fig. 2. *9111A system block diagram. The microprocessor in the system controls the interface functions, selects measurement modes, tests the system, and communicates with the host computer in addition to computing the position of the stylus.*

Somewhat different specifications are critical for the two operations. For digitizing, high accuracy with commensurate resolution is of paramount importance. Data rates do not have to be particularly high since the human can accurately trace figures only at limited speeds. Rates less than 20 points/second are normally sufficient. For interactive graphics, smoothness of cursor motion is a useful attribute. This means a high data rate is desirable. The 60 points/second maximum rate of the 9111A equals the refresh rate of most raster displays. Even if the logical position of the cursor is updated more rapidly, the user can only see changes at the display refresh rate.

An important characteristic is the stability of the position indicated by the tablet when the stylus is held motionless. If the X,Y coordinate has instability in the lowest-order digit, an annoying flicker of the cursor will result. The 9111A outputs data that typically doesn't vary when the stylus position is stable. The stability performance is achieved by data averaging techniques in the hardware and by data processing algorithms in the firmware.

Another parameter contributing to the perceived quality of cursor motion is synchronization between the refresh rate of the CRT and the data rate of the tablet. If the tablet is putting out 59 points/second, the CRT is refreshing at 60 sweeps/second, and the stylus is in constant motion, then the cursor on the CRT will move 59 times in a row. Every sixtieth sweep, the tablet will not have new data available —causing the cursor to freeze for one frame. The human eye is surprisingly sensitive to this interference effect between the devices. To overcome this problem, the tablet update rate can be accurately set to any integer value between 1 and 60, inclusive. Timing for this process is derived from the master crystal oscillator, with minor deviations introduced by firmware routines. The design goal was ±0.2 Hz from the programmed rate. Better than ±0.1 Hz is achieved at most frequencies. This means that a beat frequency will be present, but the period is so long that cursor jumping will be infrequent enough to be unobjectionable. This approach gives good performance with 60-Hz-refresh crystal-controlled CRTs as well as line-synchronized 50-Hz and 60-Hz displays.

Another advantage of data rate programmability in an interrupt-driven graphics system is that the computation load placed on the CPU can be accurately set by a single command to the tablet from a device driver or application program.

The time lag between the measurement of stylus position and the corresponding movement of the CRT cursor affects how fast a desired point can be picked by a human. A feedback loop is formed by the human to eliminate the error between the desired and actual location of the cursor. Delay in the feedback loop can cause overshoot and oscillation of the person's arm when rapidly moving to a point. The firmware in the 9111A is designed to minimize lag, regardless of the programmed data rate. Positional data is gathered from the hardware as late as possible during a measurement period. At the beginning of the next period the data is organized into the final format with 100-micrometre resolution. As soon as this is done, any pending commands can be executed. If the command is one that requests data, then the data delivered is as fresh as possible. Input of commands may occur virtually any time during the cycle. By limiting the time window in which commands are executed, the mainframe can synchronize easily with the data rate of the tablet. For example, the tablet has the computational power to execute approximately ninety OC (Output Cursor position) commands/second, but a fast controller will actually sync at the 60-points/second update rate of the tablet. The time window does not preclude more frequent execution of other more simple commands because up to 200 such commands can be executed per second.

Although optimized for interactive graphics, the tablet is also adequate for some digitizing applications. It is rated at ±600-micrometre accuracy (±0.024 in). This is suitable for the entering of small sketches and some strip chart applications.

How It's Done

The 9111A Graphics Tablet uses the capacitive-coupling, electrostatic-drive technique of the 9874A Digitizer to determine the X,Y coordinates of the stylus. A block diagram of the system is shown in Fig. 2. The platen is an epoxy-glass printed circuit board with a grid of metal traces spaced 6.4 mm apart. The traces run vertically on the circuit

Capacitive Stylus Design

by Susan M. Cardwell

The stylus for the 9111A satisfies a set of stringent design requirements. It is mechanically rugged. It can withstand being dropped on the floor or having its cable pinched in a drawer. Since the stylus will be held by the user for extended periods of time, it is slim, light, and provides clear tactile feedback of a "picked" position on the platen surface. These and other electrical performance and reliability criteria are met by an integrated system of transducers, body parts and cabling.

Signal Transducers

Two independent types of information are transmitted from the stylus to the tablet:

1. The signal that is interpreted as X-Y location, and
2. A switch signal used to identify a given point as chosen for purposes of digitizing, picking or program control.

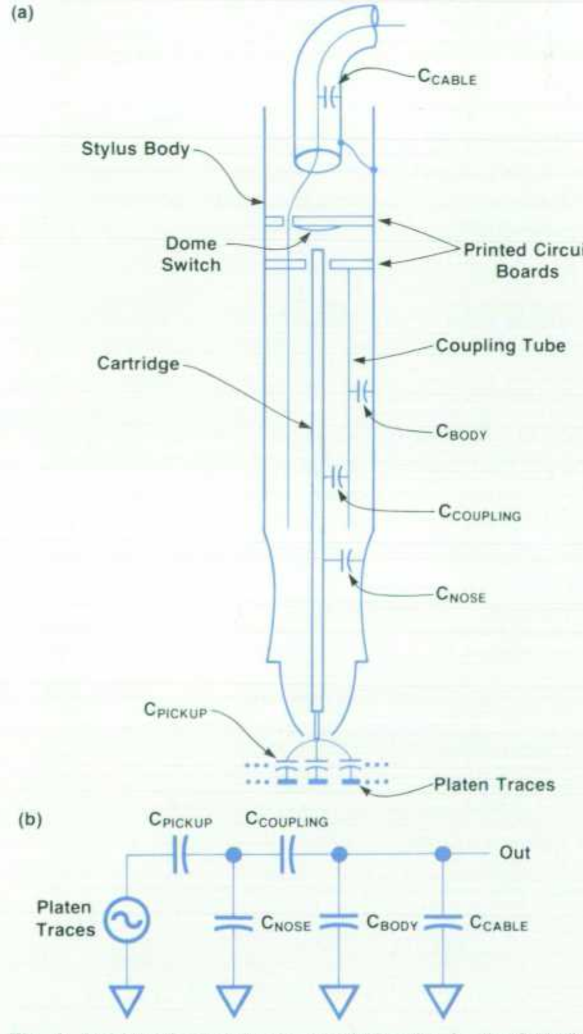

Fig. 1. *Design of capacitively coupled stylus for the 9111A. (a) Cross-sectional view showing the conductors and capacitive components. The stylus cartridge can be replaced by an ink-filled cartridge for users drawing images instead of tracing them. (b) Equivalent circuit for the stylus signal path from the platen traces to the instrument.*

The location of the stylus is determined by the capacitive coupling established between the platen grid and the tip of a metal pen cartridge. This cartridge is capacitively coupled to a coaxial tube within the stylus body (Fig. 1). To select a point, the pen tip is pressed, which activates a dome switch at the base of the cartridge. The snap action of the dome switch provides tactile feedback to the user.

The second stage of capacitive coupling is used for its advantage over direct or sliding contacts. Direct contact with the cartridge requires a permanently attached contact which precludes easily exchangeable cartridges (ink and inkless) and necessitates adding a flexing signal wire to accommodate switch operation. A sliding contact is likewise undesirable in that it impairs the distinct tactile feel of the dome switch and it creates electrical noise.

The equation for the series capacitance between the coupling tube and the cartridge is:

$$C = \frac{2\pi\epsilon_o\epsilon_r L}{\ln(b/a)}$$

where b is the larger diameter, a is the smaller diameter, L is the length of the tubes, ϵ_o is the dielectric constant for a vacuum (8.854×10^{-12} F/m) and ϵ_r is the relative dielectric constant for air ($\approx$1.0006). Within the constraints of mechanical tolerances, the coupling capacitance is maximized to a value of $C\approx11.5$ pF. This provides approximately two-thirds of the signal output of a direct contact scheme. Signal strength is maintained by slightly increasing pen tip exposure.

Body Parts

A combination of machined metal and molded plastic body parts supports and houses the transducer system. Both the dome switch and the capacitive coupling tube are mounted on two small, round, printed circuit boards at the back of the stylus. Shielding both the capacitive coupling tube and the signal-carrying metal pickup is essential to proper operation. Since the body parts must permit repeated exchange of cartridges and still insure a ground when connected, they are made of aluminum

Susan M. Cardwell

Susan Cardwell joined HP in early 1978 as a production engineer for the 9874A Digitizer. She worked on the development of the 9111A Tablet and is currently involved with LSI circuit design. Susan received the BS degree in engineering from Swarthmore College in 1977 and did graduate study at the Massachusetts Institute of Technology in the fall of 1977. She is a native of Detroit, Michigan and she and her husband make Loveland, Colorado their home. Susan is on the board of directors for the Canyon Concert Ballet Company and is an administrative member of the Loveland Bahá'i Spiritual Assembly. She enjoys the theatre, dance, and backpacking.

treated with a chromate conversion process to guarantee high conductivity.

Because the signal level is low, the system can be highly sensitive to triboelectric effects—friction forces at the interface between a conductor and an insulator that create charges and a resulting current. Molded polypropylene parts meet the multiple requirements of small geometries, dimensional stability, insulation and minimal tribolectric contribution at the cartridge bearing surfaces.

Cable

The cable carries two switch wires and a low-capacitance coaxial platen-signal wire from the stylus to the tablet. Because this component of the stylus experiences the greatest stresses, it greatly affects stylus reliability. Cable life is maximized by using multiple strands of a custom copper alloy wire with exceptional flex life. The switch wires are tightly coiled around the coax shield so that cable flexing places minimal tension and compression stresses on the conductors. Strain relief boots at both ends are carefully tapered to avoid sharp strain of the cable.

side and horizontally on the component side of the board. The signal picked up from each trace is a function of the distance between the tip of the stylus and the trace. The signal from any single trace is proportional to the drive voltage and coupling capacitance. Since the coupling capacitance is a function of distance, the stylus output voltage V_0 is the sum of the individual trace drive voltages reduced by the distance from each trace to the stylus (see Fig. 3).

In the 9111A the traces are driven by CMOS shift registers. The choice of 12-volt CMOS shift registers instead of the 5-volt registers used in the 9874A provides a proportionately greater signal to the stylus. Using the capacitive coupling technique, the stylus output voltage V_0 becomes cyclical at the same frequency as the 9.765-kHz signal driving the traces. This allows, after processing as described below, the 60-samples/second data rate desired for interactive graphics applications.

The stylus signal V_0 is buffered, filtered about 9.765 kHz and sent through a zero-crossing detector to recreate the original 9.765-kHz reference frequency. The recreated frequency is shifted in phase, depending on stylus position. To calculate the distance from the first trace to the stylus, we simply measure the phase shift of the stylus signal referred to the signal of the first trace.

The resolution of the 9111A is 0.1 mm. To achieve this, a series of phase measurements is made in calculating each X,Y coordinate pair. The sequence in which these measurements are made is:

X reference
X coarse
X fine
Y fine
Y coarse
Y reference

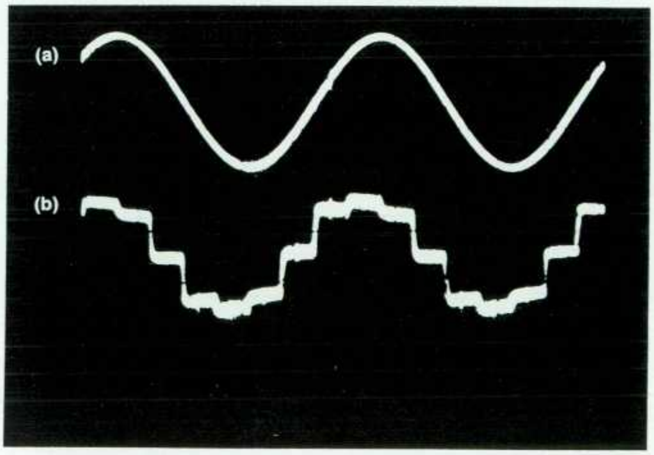

Fig. 3. *Stylus signals after (a) and before (b) bandpass filtering. The lower trace shows the stairstep waveform resulting from the summation of the signals from each of the platen traces.*

The reference measurements are used to compensate for the time delay added to the stylus signal by the filter electronics. This also eliminates error due to drift in component parameters because of aging, temperature and humidity. In reference mode, all the traces are simultaneously driven high and then all are driven simultaneously low at a 9.765-kHz rate. Since the signal on every trace is identical to the signal on the first trace, any phase shift in the stylus signal will be caused solely by the filter delay. The reference measurements are subtracted from the coarse and fine measurements.

In the measurement mode each trace is driven sequentially high and then each trace is driven sequentially low at a clock rate dependent on the resolution required. The coarse measurement rate addresses all of the traces during one cycle of the 9.765-kHz signal. The fine measurement clock rate is one-eighth that of the coarse rate so that eight cycles of the 9.765-kHz signal occur before all of the traces are addressed. The coarse mode creates a wavelength of 409.6 mm on the platen. Using the platen frequency, 9.765 kHz, we calculate the velocity of the signal and then find the wavelength.

$$\begin{aligned}\text{Velocity} &= (\text{trace spacing}) \times (\text{coarse mode clock rate}) \\ &= (6.4\ \text{mm}) \times (625{,}000\ \text{traces/second}) \\ &= 4000\ \text{m/s}\end{aligned}$$

$$\begin{aligned}\text{Wavelength} &= \text{velocity/frequency} \\ &= (4000\ \text{m/s}) / (9.765\ \text{kHz}) = 409.6\ \text{mm} \\ &= 16.12\ \text{in}\end{aligned}$$

This wavelength is longer than the platen's maximum dimension. The coarse measurement provides a rough but unambiguous guess of where the stylus is within this long wavelength.

The coarse position is refined by making another measurement with increased resolution. The wavelength of the fine mode (51.2 mm) is one-eighth the coarse wavelength. By keeping the same phase resolution, the physical resolution of the system is increased over the resolution of the coarse mode by a factor of 8. The 8:1 ratio of wavelengths is large enough to attain the desired accuracy and stability of the machine, yet small enough to eliminate any possibility of having the coarse measurement "guess the wrong fine wave" to be refined. An error of this kind would result in an annoying jump in the position data supplied to the host.

The firmware also reduces this possibility by continuously monitoring the accuracy of the coarse estimate. The sequence of measurements calls for adjacent X-fine and Y-fine measurements to minimize the effect of any linear motion of the stylus during the sequence.

When measuring the phase of the filtered signal it is necessary to avoid incorrect measurements when the signals are near the 0/360° wrap-around point. This is done by checking the phase relationship of the stylus and reference signals while the filter is settling. This process is called reference adjustment. If the stylus signal rising edge is less than 90° or greater than 270° from the rising edge of the reference signal, then 180° is added to the reference signal. At the completion of the reference adjustment (once the filter is settled), the phase difference between the stylus and reference signals is between 90° and 270°, which gives the system increased immunity to noise in the stylus signal. If added, this extra 180° phase shift is subtracted later.

The phase shift of the 9.765-kHz stylus signal in each of the six measurement modes is summed for sixteen periods to further reduce noise effects on the measurement. Firmware reduces the summed measurement into a number representing the phase shift of each mode. This is simplified by the 6.4-mm spacing of the traces on the platen. The microprocessor uses the count from each of the six modes to compute the X,Y position of the stylus. The quality of the actual position measurement compared to the conservative specification is illustrated in Fig. 4.

The microprocessor also performs all algorithms necessary to sequence the platen, control the light-emitting diodes (LEDs) and variable tone beeper, and communicate with the host computer. The microprocessor controls the shift registers by writing data to the shift control which in turn creates the proper signals to drive the shift registers in the various platen modes.

The microprocessor-based system (Fig. 2) has the capability to perform extensive verification of its proper operation. The electronics self-test is initiated by the processor every time the instrument is powered on. It is also entered upon receipt of the IN (Initialize) command from the host computer. This test flashes the LEDs (for operator verification of their functionality) and then performs a series of tests on internal hardware. First, the microprocessor's internal registers are tested. Memory tests are a read-only-memory (ROM) checksum and extensive bit pattern testing of the read/write memory (RAM). The functionality of the three I/O ports for the shift control and phase counter is checked. The phase counter is cleared and the normal operating sequence of bit patterns required to perform reference adjustment and measurement summation is sent to the phase counter.

Communication with the HP-IB interface chip is verified and, finally, the programmable countdown timer is tested. This timer interrupts the processor for the normal sequencing of the six platen modes and is used to pace the processor to create the proper frequencies for the variable tone beeper. If the electronics passes the tests, a three-tone pass beep is generated by the beeper. If any of the tests fails, a loud, warbling error tone is output and the **ERROR** LED is lit.

The proper operation of the shift control, shift registers, filter and stylus can be verified by performing the user interactive self-test. This test is normally performed immediately after the instrument is powered on. It is initiated by toggling the **SELF TEST** switch on the rear of the graphics tablet while holding the stylus away from the platen. It can also be initiated by a TD (Test Digitizer) command. The user interactive self-test first performs the electronics self-test. After the pass-beep tone from the electronics self-test, the

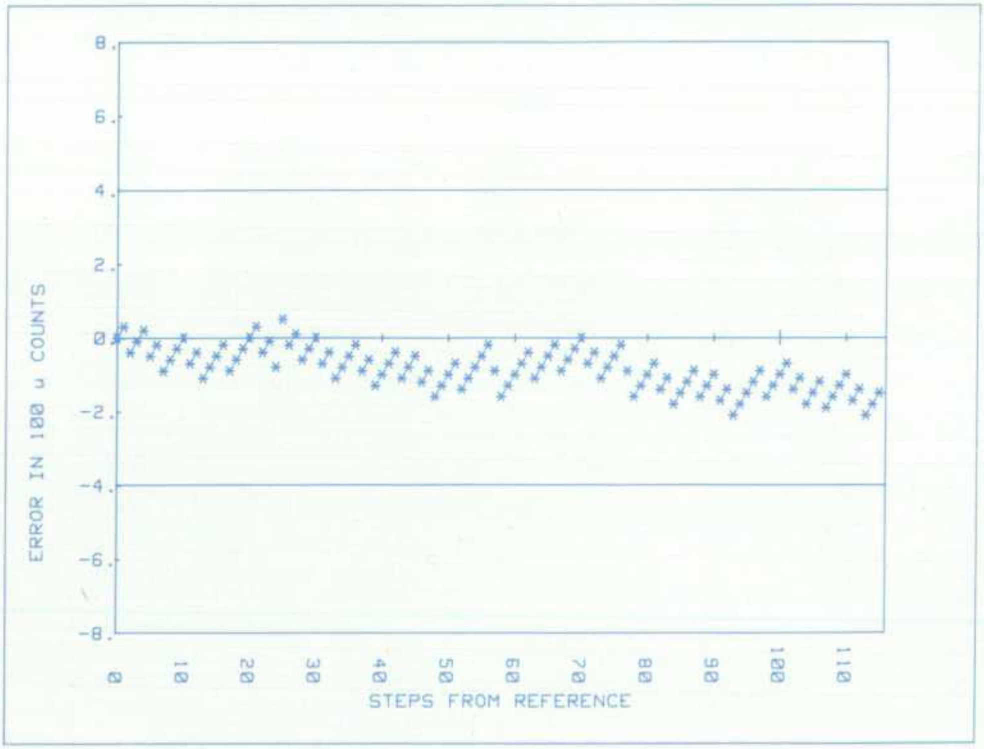

Fig. 4. *Typical position measurement accuracy plot. The system specification is ±600 micrometres with typical values within ±400 micrometres.*

Programming the Graphics Tablet

by Debra S. Bartlett

The firmware of the 9111A provides easy high-level programming. For example, software for the HP 9845B Computer has been written that allows the user to create drawings interactively using circles, rectangles, arcs, lines, and labels. The user can also pick, place, and transform elements, and set up and interpret a user-defined menu. This software takes advantage of the following features of the 9111A Graphics Tablet: interrupts, softkeys, single and continuous mode digitizing, information received from the cursor statement, scaling, and variable beeper. Software has also been written for the HP-85, 1350A, and System 45C, and is underway for the HP-1000.

The main driver for all the programs is set up on the basis of being able to interrupt the main program selectively. First, interrupts on the HP-IB SRQ line are enabled. Then by simply sending the IM (Input Mask) instruction to the 9111A Graphics Tablet, the programmer can specify the actions that will cause a service request. The programmer can set the input mask to interrupt on errors, a digitized softkey, a digitized point, or on proximity. The input mask can also be set to enable the recognized errors and to select status conditions that will cause a response from the 9111A to a parallel poll. The 9845B/9111A software is set up to generate a service request whenever the user digitizes on the active area of the tablet or digitizes one of the sixteen softkeys. This interrupt capability allows the program to track the cursor without having to check continually to see if the user digitized a point or selected a softkey. The code is simpler and the cursor moves faster.

Once the service request has been received, the program must determine what caused it. To do this, the program looks at the status word from the 9111A. This word is retrieved by simply sending an OS (Output Status) command to the tablet and then by reading it into the 9845B. Information contained in this word is pen press, new cursor position, proximity, digitized softkey, SRQ, error, and digitized point available.

If the seventh bit of the status word has been set, the program knows that the user has digitized one of the sixteen softkeys. Because of the firmware in the graphics tablet, the program can then execute a RS (Read Softkey) command to determine exactly which softkey (1-16) was selected. As a result, the program does not have to look at the exact X,Y coordinate digitized and see if it falls within certain boundaries to determine which softkey was digitized. This provides the programmer with an easily used predefined menu area. The 9845B/9111A software uses these softkeys to perform such operations as get a picture, save a picture, label, delete object, delete line, snap to a grid, plot, clear the CRT, and help.

If the second bit of the status word has been set, the program knows that the user has digitized a point on the active area of the platen. The drawing program and the editor program use this area for menuing, cursor moving and placing. Part of the platen is used for cursor movement and placement and the other part of the platen is used as additional menu space for such items as pen, line type, character, and element selection, or for rotating, scaling, and moving objects in the editor program. To determine which area has been selected, the program sends the tablet an OD (Output Digitized point) command and then reads in the digitized X,Y coordinate from registers in the graphics tablet. Because the 9111A has registers for storing the X and Y values of the digitized point, the user does not have to worry about moving the stylus too quickly and losing the X and Y values before the program can get to it. Thus, the program always knows the exact point that the user has digitized.

The firmware in the tablet allows for two modes of digitizing. The first is the single-point mode. In this mode, each time the stylus is pressed, a coordinate point is stored in the registers on the tablet. The software for the 9845B/9111A uses this mode for menu picking, placing elements, and drawing straight lines.

The second mode of digitizing is the continuous sample mode. This mode can be set to take points when the stylus has been pushed down and released, stop when the stylus is pressed down again (switch normal) or take points only when the stylus is pressed down (switch follow). The data rate for this continuous mode can be specified (one to sixty updates per second) by the programmer simply sending the graphics tablet a CR (Cursor Rate) command. The drawing program uses the continuous switch follow mode. This mode lets the user draw curved lines and trace pictures placed on the platen.

The graphics tablet's firmware is set up so that the program can tell what the user is doing without using interrupts. The CURSOR statement is one way to do this. The CURSOR statement returns the X,Y coordinate of the stylus location, the pen parameter, the number of the softkey if one has been selected, the status word and the error number. The 9845B/9111A software uses this method within the subprograms. For example, when in the subprogram for drawing single-point lines, the program uses this information to determine if the user has digitized a point on the active area of the platen. If the user has, the program checks to see if the digitized point lies within the placement area of the menu. If it does, the program reads in the point and draws to it. If it does not fall within the placement area, the program returns to the main program where the unread digitized point can cause an interrupt and branch to the subprogram that the user has selected. This versatility gives programmers a way to vary the method of retrieving information from the tablet.

The graphic tablet's implementation of HP-GL allows the programmer to use scaling commands as implemented in the 9845 graphics ROMs. The software takes advantage of this to do a one-to-one mapping of the placement area on the tablet to the CRT screen of the 9845B. First the program uses the LIMIT command to specify in metric units those areas of the CRT and the tablet that will be used for cursor tracking. Then the program executes a SHOW statement to scale the tablet and CRT to the same number of units in the X and Y direction. This makes the program simpler to code because it does not have to be concerned with transforming every point received from the graphics tablet.

The 9111A Graphics Tablet has a programmable beeper. The programmer can specify the frequency, duration, and amplitude of the beep. It has a range of four octaves, can last from one millisecond to 33 seconds, and can have six different degrees of loudness. The software for the 9111A/9845B takes advantage of this function to give feedback to the user by specifying five different sequences of beeps to indicate different conditions. There is a pick tone for indicating that a menu item was picked, an error tone indicating that the user has digitized an undefined area or didn't answer a question correctly, a data tone indicating that a point was placed, a question tone indicating that the user needs to answer a question, and a finished tone indicating that a particular operation has been completed.

The firmware has some other features that make higher-level programming easier. The first is the OI (Output Identity) command. This command is used to determine which peripheral device is on the bus. This lets a programmer use the same piece of software for the 9111A and the HP 9874A Digitizer. When the program starts executing, it checks to see which input device is connected and then the program can set the correct scaling parameters for that particular size platen. Another useful command is the OE (Output Error) command. The tablet is set up so that if a program statement sends the tablet an unrecognized command or illegal parameter, it will not cause the program to quit. Instead the **ERROR** LED on the tablet is lit to indicate an error and the error number is stored in a register in the tablet. After the tablet has received the error, the programmer can read in the error number using the OE command.

Debra S. Bartlett

Debbie Bartlett has a BS degree in mathematics awarded by Purdue University in 1977 and is currently studying for a master's degree in computer science at Colorado State University. She has been with HP since 1979 writing software for new products. Her previous work experience involved statistical analysis. Debbie is married (her husband works at HP's Loveland Instrument Division) and lives in Loveland, Colorado. She is a native Hoosier from Indianapolis, Indiana and likes water skiing and hiking.

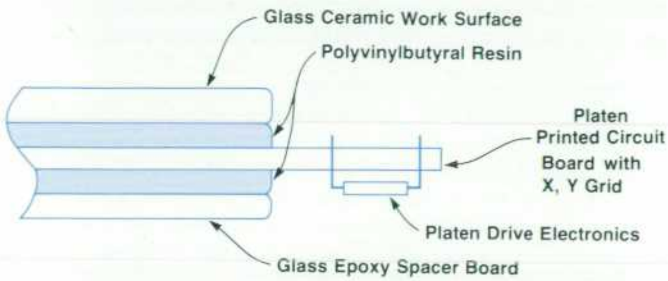

Fig. 5. *Cross-section of platen printed circuit board assembly. The ceramic work surface provides a durable layer that resists scratching and is dimensionally stable.*

user must digitize the dot at the lower right corner of the platen artwork. If the shift control, shift registers, stylus and filter circuitry are all operating properly, another pass-beep tone will be generated. Failure of the test is indicated by the error tone and the lit **ERROR** LED.

Any error from the electronics or user interactive self-tests, improper or unrecognized commands, or system failure can be analyzed by sending an OE (Output Error) command to the 9111A. A read or enter statement will retrieve the value representing the error.

The microprocessor-based system is designed to help a trained service technician troubleshoot the digital electronics using the 5004A Signature Analyzer. Signature analysis routines for the system clocks, microprocessor addressing, ROM and phase counter can be activated by proper placement of removable jumpers on the printed circuit board. Additionally, test points are available for signature analysis, microprocessor, clock frequencies, filter and phase counter signals, as well as for voltage and ground.

Mechanical Design

The 9111A platen assembly consists of an X-Y coordinate grid sandwiched between a work surface above and an insulating layer below (Fig. 5). These are laminated using a sheet form of polyvinylbutyral resin. The grid is built on a standard two-layer glass-epoxy printed circuit board with 56 X-traces on the top side of the board and 46 Y-traces on the bottom. By extending the edges of the printed circuit board, platen drive electronic components are loaded next to their respective traces, eliminating the need for extensive cabling.

Of the many materials considered for the work surface, the glass ceramic chosen stands out with its exceptional durability, high dielectric constant and homogeneity. The axis lines and sixteen menu boxes are permanently fired into the platen surface. Thermal stability over a wide temperature range enables straightforward lamination and wave soldering. Lamination ensures a homogeneous dielectric constant by excluding air pockets. Furthermore, it provides a safety-glass construction between the ceramic and the printed circuit board.

Simplicity of construction dominates the internal product design. A sheet-metal chassis holds the control printed circuit board and power supply. A second piece of sheet metal mounted on the chassis both supports the platen and shields its grid from the electronics below. The flex cable connecting the two printed circuit boards is positioned such that the instrument can be fully operational during service (Fig.6).

A survey conducted at the beginning of the design indicated that users prefer the comfort of a sloping surface to that of a flat pad. This simultaneously allows a low front edge and enough space to fit all the electronics in the rear. The resulting wedge shape is more convenient, comfortable

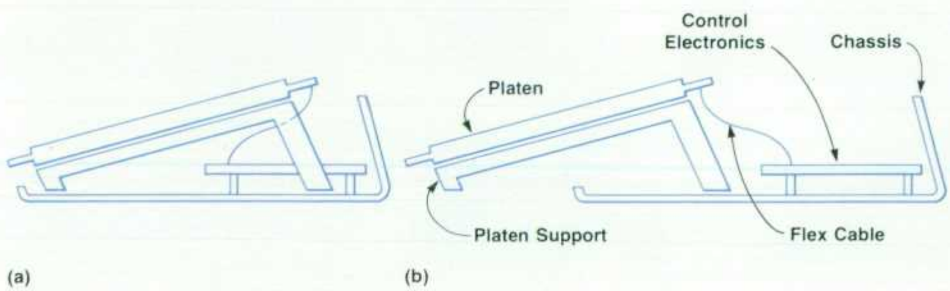

Fig. 6. *The 9111A hardware design allows service personnel to easily open the unit for servicing and still maintain normal operating functions. (a) Normal operating configuration. (b) Open for servicing.*

Tablet/Display Combination Supports Interactive Graphics

by David A. Kinsell

Interactive graphics systems have special I/O requirements that place a heavy burden on the computer. To move a cursor on the CRT requires a number of very short messages between the tablet and the display device. Even if the computer can devote full time to this cursor tracking, an unacceptably low update rate can result due to the system overhead required to set up each data transfer. To solve these problems, the 9111T was created.

The 9111T has all the capabilities of a standard 9111A, plus firmware enhancements that allow direct HP-IB communication to a 1350A Graphics Translator. The result is high-performance cursor movement and rubber-banding (stretching of lines) done by the peripherals that is totally independent of computer speed. In essence, the peripherals act in harmony as if they were a single graphics terminal. They also can be used independently with no physical reconfiguration of the bus.

This type of direct communication on the HP-IB, from one non-controller device to another, has always been provided for in the bus definition. It is rarely used because few combinations of peripherals have anything useful to say to each other. Data formats may differ and some form of mnemonic commands is usually required to control each device. For instance, let's look at the feasibility of having a digitizer device (9874A) plot digitized points directly to a plotter device (9872A).

After the user has digitized a point the digitizer must receive the OD (Output Digitized point) command to enable output of the digitized data. The command PA (Plot Absolute) must preface X,Y coordinates for the plotter to function properly. If a digitizer and plotter were hooked together on a bus and addressed as talker and listener, nothing would happen because the proper commands would not be received by the peripherals. Also, no scaling could be done to compensate for the different sizes of the devices. These details are normally taken care of by the computer, which is an acceptable approach for the low data rates of manual digitizing.

In contrast, the 9111T has the capability of issuing commands to the 1350S Display System in the language used by the 1350A Graphics Translator (Fig. 1). After being set up by the computer with a single command, the 9111T repeatedly streams out commands and data directly to the 1350A that perform cursor movement and rubber-banding functions. The scaling is changed to map most of the 9111T active area onto the display (Fig. 2). The stroke refresh used by the 1350A is ideal for rubber-banding and moving objects. Only a few vector locations need to be changed by the tablet, in contrast to the large number of pixels that must be altered in a raster refresh system.

The 9111T provides these additional functions, listed with the corresponding number of updates/second:

- Moving alpha or cursor symbols (60)
- Single-line rubber-banding — normal, forced horizontal, and forced vertical (60)
- Double-line rubber-banding (60)
- Rectangle rubber-banding (60)
- Variable-size cursor (single-dot to full-screen) (40)

The added capabilities are aimed primarily at IC and printed

Fig. 1. *The combination of a 9111T Graphics Tablet, a 1350S Display System and a controller such as the 9825 provides powerful interactive graphics in an inexpensive workstation.*

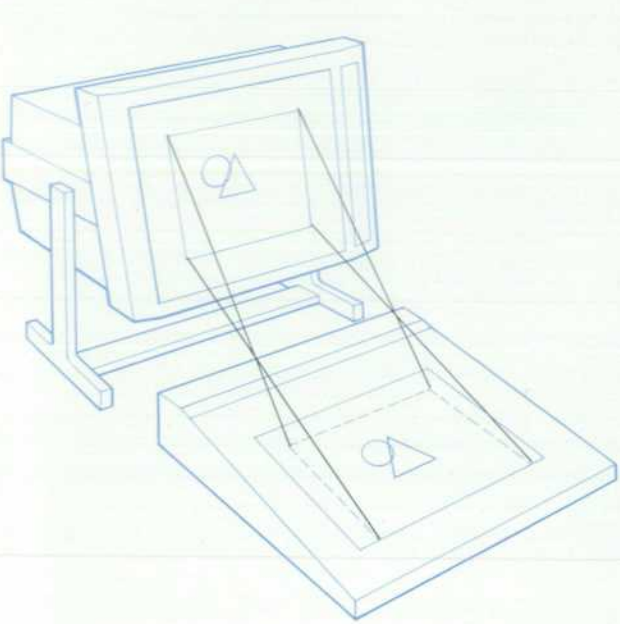

Fig. 2. *The platen area on the 9111T Graphics Tablet is directly mapped onto the CRT of the 1350S Display System as shown.*

circuit layout applications, but other applications involving menu picking, symbol picking, picture creation, or the use of valuators will also benefit.

To access one of these features, the following things need to be done:

1. The computer plots data in a few of the vector locations of the 1350A.
2. The computer sends the EE (Enable Echo) command with proper options and parameters to the 9111T to put it in the 1350A mode.
3. The computer configures the bus with the 9111T as talker, and the 1350A as listener.
4. The computer arms itself to respond to SRQ (the interrupt line on the bus).

After this setup is accomplished, the computer is uninvolved with the data transfer on the bus. It is free to handle other tasks (including servicing other 9111T/1350As on other bus systems).

The user moves the cursor or rubber-bands a line until it is in the proper position, then presses the pen to pick a particular point. When this happens, several things occur. The tablet stores the coordinates of that point, finishes any communication that might be in progress with the display, and then asserts the SRQ line on the HP-IB to request computer service.

The computer responds to the interrupt and interrogates the tablet for the digitized point. The value returned is scaled in terms of the display screen coordinates instead of the normal coordinate system of the 9111T. Thus, to convert from a user coordinate system to the peripheral coordinate system, a programmer need not worry about maintaining different scale factors for the tablet and the display. Typically, the user redraws a rubber-banded object in the 1350A's memory using the coordinates of the digitized point. Snapping points to a nearby grid intersection is easily done during this operation.

Instead of picking a point, the user could pick a menu box to control the program. The menu item selected is read by the computer in the normal manner.

The high resolution of the 1350A (1023×1023), along with the fast update rate and stable output from the tablet, combine to provide tracking with excellent aesthetics. The independence from computer I/O performance makes interactive graphics feasible for smaller computers (such as the HP-85) and opens the way to multi-workstation operation on large computer systems.

David A. Kinsell

Dave Kinsell started work at HP in 1975 after receiving the MSEE degree from Purdue University. He also earned the BSEE degree from Purdue in 1974. Dave has done NMOS IC R&D, graphics display studies, production support for the 9874A Digitizer, and, most recently, the implementation of firmware and system specification for the 9111A Tablet. Dave is a native of Remington, Indiana and now lives in Loveland, Colorado. He is a member of the IEEE, rides his bicycle to work, likes mountain hiking and skiing, and is a glider pilot.

and compact than a tablet that requires separate boxes for electronics and/or power supply.

Internal layout promoting natural convection paths, combined with low power dissipation, permits an instrument with no fan, hence a quiet machine. Air enters along the bottom of both sides and exits at slots in the rear.

While the tablet functions primarily to move a CRT-display cursor for some users, others have applications for which they need to place a menu directly on the work surface. To please both groups, it is necessary to provide an unobtrusive means to hold down paper. In the edge of the case where it meets the platen are four thin slots that accommodate tabs on a clear mylar overlay. With this design the user can protect a menu while holding it in place. When menus are not in use, the tablet is free from protruding or unsightly mechanisms.

Ease of assembly affects the people who build an instrument, those who service it, and the cost to the user who buys it. The case top was designed to be independent from instrument operation; no components are mounted in it. When assembled, two long conical bosses in the case top reach through holes in the platen assembly to align the platen in the case top window. The LEDs are mounted directly on the platen assembly as well. Therefore, both platen and LEDs are positioned with no hardware, no loose parts and no wiring.

Interface Language

For compatibility with other HP graphics devices, the tablet communicates primarily in Hewlett-Packard Graphics Language (HP-GL). This language was developed by several HP divisions for use with mechanical plotters and manual digitizers. It uses ASCII-encoded two-letter mnemonics and free-field integer representation of data. This is a very common type of data formatting used on the HP-IB. It can be handled by just about any general-purpose

I/O driver used with the bus. Its main disadvantage is the time required to do the numerical conversion from this format to the internal format of the mainframe. Conversions from the tablet's internal format to the integer representation are handled by the fast lookup algorithms in the tablet.

To help relieve the host computer of the conversion burden, binary formatted data of the X,Y position is also available from the tablet in a straightforward format supported by several HP computers. These include the 9835, the 9845, the HP-85, and the HP 1000. The I/O drivers can input two bytes of data and place them directly into a 16-bit integer variable. This format is also convenient to use when assembly language drivers are written to support the tablet.

The tablet can also be put into a Talk Only mode. The same binary formatted data is available in this case. Although commands cannot be sent to the tablet, it is easy to design a custom interface to read the data.

The HP-GL language gives coordinate values in an absolute peripheral coordinate system. To simplify programming, high-level support is available on a number of computers to provide scaling into user-defined units and to support the digitize function. The 9825 with a 9872A plotter ROM provides single-point digitizing support that can be used for continuous-mode digitizing with some additional HP-GL programming. The 9835 and 9845 computers have graphics ROMs that provide BASIC-language support of the tablet. The HP-85 printer/plotter ROM is essentially equivalent. A high-level graphics support package will also be available on the HP 1000.

The System 45C (Model 9845C) provides a powerful set of high-level programming features through its graphics ROM. The GRAPHICS INPUT IS ... statement allows a full complement of graphics input devices, including the 9111A Graphics Tablet, 9874A Digitizer, light pen, and arrow keys to be handled uniformly in an application program.

Once the tablet has been declared as the input device, the programmer can set up a software interrupt with a single command, ON GKEY .... With this command in effect, the program will branch to a desired location any time a point or softkey is digitized on the platen. Another command TRACK...IS ON allows automatic tracking of the cursor on the CRT. When this command is in effect, the cursor position is updated after every executed BASIC line. Finally, the CURSOR... command, while also implemented on the 9845A/B computers, returns additional tablet information concerning clipping and softkey values only on the 9845C. The additional capabilities of the 9845C graphics firmware together with the 9111A graphics tablet make applications programs shorter and easier to write and debug.

Acknowledgments

Several people played key roles in the success of the 9111A. Susan Cardwell was responsible for the total mechanical design of the instrument. Dave Kinsell wrote the 9111A firmware, and personally initiated and implemented the 9111T follow-on. Mark Gembarowski designed the digital hardware, and Tim Hitz designed the analog circuitry and power supply. Dave Jarrett implemented the 9111A stylus, and Debbie Bartlett implemented the application utility routines. Mo Khovaylo was responsible for the industrial design. Special thanks to Dave, Susan, and Mark for their contribution to this article. I am indebted to Larry Hall who, as initial project manager through breadboard, passed on an efficient, well-run project.

Reference

1. F.P. Carau, "Easy-to-Use, High-Resolution Digitizer Increases Operator Efficiency," Hewlett-Packard Journal, December 1978.

Donald J. Stavely

Don Stavely was born in Detroit, Michigan and attended the University of Michigan where he received the BS degree in electrical engineering in 1975 and the MS degree in computer engineering/computer science in 1976. He came to HP that same year and has worked on firmware for the HP 250 Computer and directed the completion of the 9111A Tablet development. Don is currently managing an LSI design project. His outside interests include canoeing, backpacking, and woodworking. Don and his wife live in Fort Collins, Colorado.

SPECIFICATIONS

HP Model 9111A Graphics Tablet

FEATURES:

MENU: Sixteen softkeys.

COMMAND SET: Twenty-five HPGL commands through HP-IB interface (Hewlett-Packard's implementation of IEEE Standard 488 (1978)).

BEEPER: Programmable in pitch, volume, and duration.

ACTIVE DIGITIZING AREA: 218.5 × 300.8 mm, not including menu area.

DATA RATE: Programmable from 1 to 60 coordinate pairs/second. Average rate ±0.2 Hz from nominal.

SELF-TEST CAPABILITY

PLATEN: Ceramic surface, artwork (origin, self-test dot, any border) accuracy measured versus documented is ±2.8 mm.

GRAPHICS DATA:

FORMAT: ASCII or binary X, Y coordinate data.

RESOLUTION: 0.100 mm.

ACCURACY: ±0.600 mm at 20°C, each measured point. Derate 0.004 mm/°C deviation from 20°C.

STYLUS MOTION RATE:

On paper: 500 mm/s.

On platen: 730 mm/s.

REPEATABILITY: ±one resolution unit from mode of data.

DOCUMENT MATERIAL: Single sheet, electrically nonconductive, homogeneous, less than 0.5 mm thick.

POWER REQUIREMENTS:

SOURCE (±10%): 100, 120, 220, or 240 Vac.

FREQUENCY: 48 to 66 Hz.

CONSUMPTION: 25 W, maximum.

SIZE/WEIGHT:

HWD: 85 × 440 × 440 mm.

WEIGHT: 5.8 kg, net; 10.8 kg, shipping.

OPERATING ENVIRONMENT:

TEMPERATURE: 0 to 55°C.

RELATIVE HUMIDITY: 5 to 90% at 40°C, noncondensing.

PRICE IN U.S.A.:

9111A: $1950.

88100A (Utility Software Package for 9845B Computer): $500.

9111T (Option for use with 1350S Display System): $2450.

MANUFACTURING DIVISION: GREELEY DIVISION
3404 E. Harmony Road
Fort Collins, Colorado 80525 U.S.A.

Programming for Productivity: Factory Data Collection Software

DATACAP/1000 is a software tool for designing and managing data collection networks. Running on an HP 1000 Computer System, it is flexible, easy to use, and compatible with user-written routines.

by Steven H. Richard

WHAT IS FACTORY DATA COLLECTION? For many people the whole concept of collecting or capturing data at its source is an unfamiliar one. Those who have had exposure to real-time factory automation are familiar with process or machine data acquisition and control loops (Fig. 1a). In factory data collection we are dealing with logistical control rather than machine control (Fig. 1b). The term factory data collection encompasses acquisition methods normally associated with manual input.

Examples of the kinds of data normally collected in the factory data collection environment include labor information (time and attendance), work-in-process tracking, inventory control, component/product testing, and so on. At the present time, most manufacturing companies keep track of this sort of information through paperwork systems or combinations of paperwork and batch computer systems.

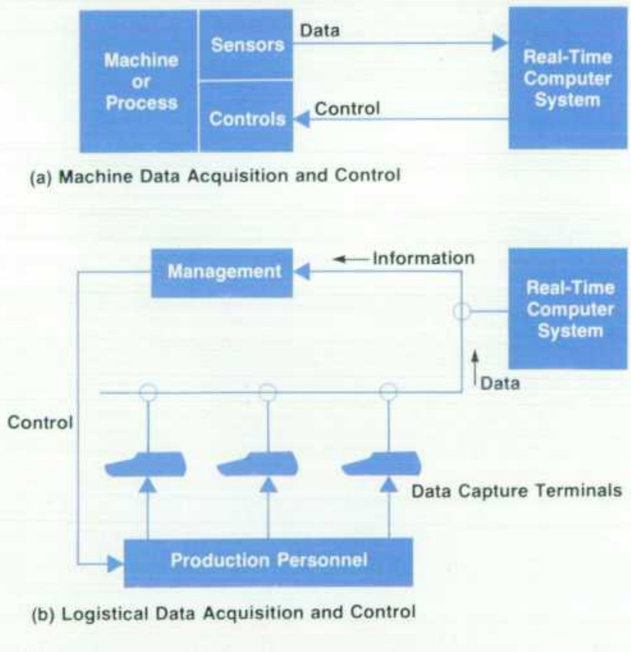

Fig. 1. *DATACAP/1000 is a software package for HP1000 Computer systems. It collects factory data such as time and attendance, work in process, inventory, and test statistics, and presents it to management. It is a system for logistical data acquisition and control (b) rather than the more familiar form of factory automation shown in (a).*

The result is that plant control information (inventory levels, order status, and so on) is at least several hours and often days old before management has the information in hand and can act on it. Without a comprehensive, real-time data capture system, the factory information system (management reports, graphs, bar charts) can be neither timely nor complete.

A Factory Data Collection System

A computer based system is particularly well suited to the relatively simple task of collecting data from the factory floor. The most powerful capability of the computer in helping to solve the data capture problem is that it can instantly catch errors made in submitting the data. There are frequently transposition, transcription, and other errors, which can be easily caught and corrected as they occur, using simple range checks, mask checks, or table lookups. This ability to validate data before it is accepted for further processing makes subsequent management reports more timely and accurate because the time-consuming and often impossible task of tracking down and correcting errors is substantially reduced.

Hewlett-Packard is one of several manufacturers of computer systems aimed at solving the factory data collection problem. The HP 1000 Computer System for factory data collection includes the following major components:

Hardware

HP 1000 E-Series Computer
HP 7906/20/25 Disc Drive
HP Factory Data Link Terminal Connection
HP Data Capture Terminals (Fig. 2) with optional
- -Displays (LED or CRT)
- -Keyboards (Numeric or Alphanumeric)
- -Card Readers (Type III, V, Magnetic)
- -Bar-Code Reader
- -Electrical I/O (RS-232C or IEEE-488)

Software

RTE-IVB Operating System
Standard Languages and Utilities
IMAGE/1000 Data Base Management System
DS/1000 Distributed Systems Software
DATACAP/1000 Factory Data Collection Software

Fig. 2. *Data collected by DATACAP/1000 is entered manually by factory personnel using a variety of HP data capture terminals like these.*

DATACAP/1000

The key software element in this application is DATACAP/1000, a new software application tool that is used to customize the general-purpose HP 1000 System to the factory data collection needs of a manufacturing facility. An important feature of DATACAP/1000 is that it is *not* an application program that makes the HP 1000 System a turnkey data collection system. Instead, it is a program that helps the user develop an application program suited to a particular factory's needs. In other words, it is a *program that generates a program*.

Basically, DATACAP/1000 generates a generalized program that operates under the control of a parameter table. This program, the transaction monitor program (TMP), uses parameters supplied by the table (the transaction specification) to control an easy-to-understand dialogue at the HP data capture terminal. Additional parameters are used to direct interaction with an IMAGE/1000 data base, data storage devices, various optional input and output devices available with HP data capture terminals, and any customized source-coded subroutines that the user might wish to use to extend the basic table-driven capabilities. A block diagram of the major components of DATACAP/1000 is shown in Fig. 3.

The major advantage of the table-driven approach is that the development of a particular application (transaction specification) involves only the construction of a new table, rather than writing, keying in, editing and compiling a source program and loading the resultant object code. The transaction generator program (TGP), a friendly interactive program, prompts the user through the process of table building. Included in TGP are sophisticated edits that prevent the user from building a table with any logical inconsistencies that might cause problems during the execution of the application. A typical screen from the transaction generator program is shown in Fig. 4.

Similarly, the transaction monitor program generator (TMPGN), is used to develop the tables needed for the TMP to manage the aspects of the application that are less dynamic, that is, the data capture terminals, user-provided extensions (subroutines), and IMAGE/1000 data bases. a typical screen from the TMPGN is shown in Fig. 5.

Finally, a very simple monitor program (DCMON), is used to provide access to the generation and operation of DATACAP/1000 through the programmable function keys of the HP 2645A System Console. This relieves the system designer and operator of ever having to remember program names or run string commands. Fig. 6 shows a typical DCMON screen.

The Transaction Generator Program (TGP)

TGP consists of one main program and 15 segments. The main program is very short and is used only to initialize some variables and to call the first segment. Thereafter, the segments perform all functions and control is never returned to the main program. All communication between segments is accomplished through variables declared in common.

A basic overview of the path of TGP operation can be seen in the flowchart of Fig. 7. In addition to the formal reference manual for DATACAP, a brief tutorial is built into the trans-

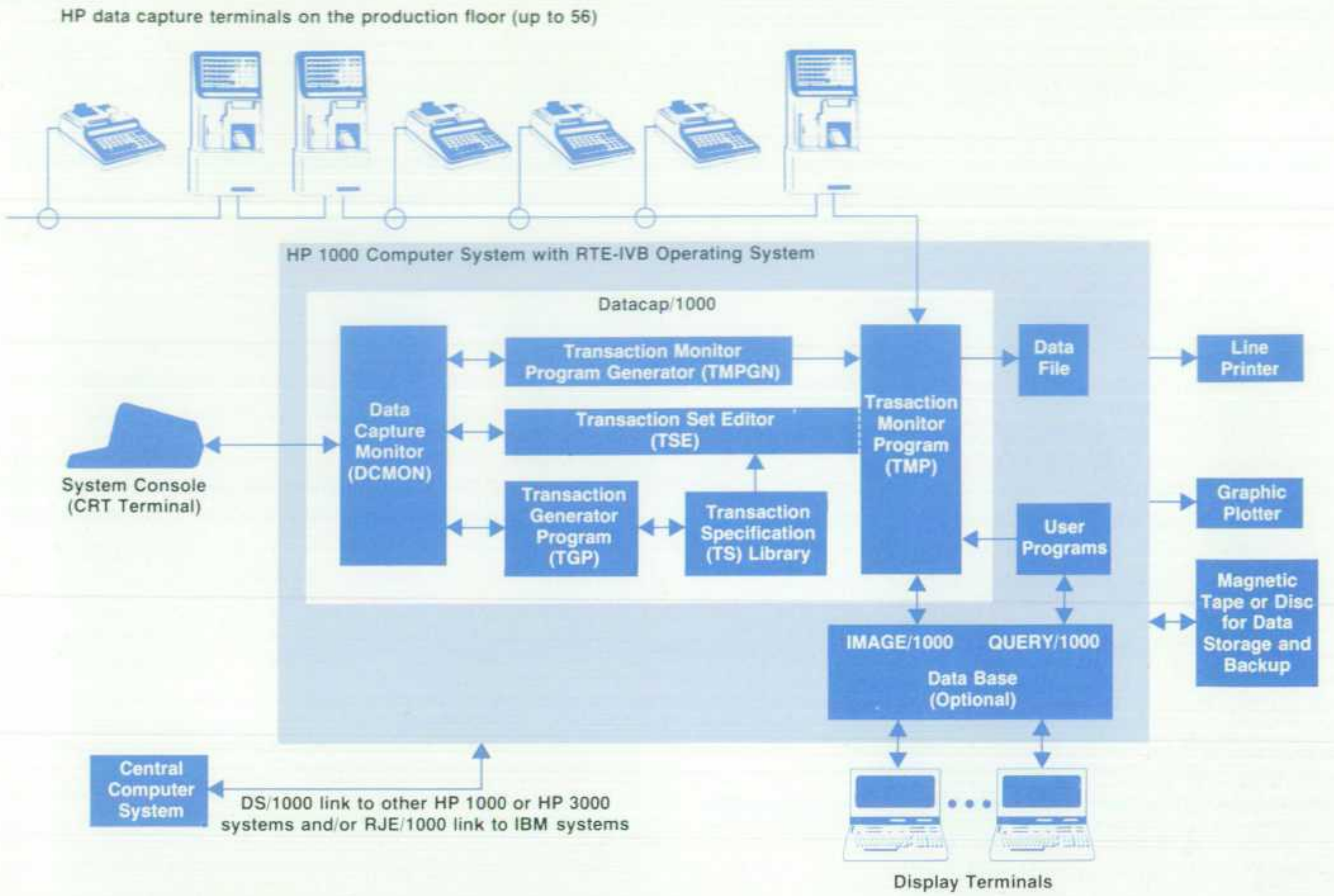

Fig. 3. *Major components of DATACAP/1000. The transaction generator program (TGP) and transaction monitor program generator (TMPGN) interactively lead the user through the process of building transaction specifications and the transaction monitor program (TMP). The TMP then manages the collection of factory data and creates data files or enters the data into the IMAGE/1000 data base, if present.*

action generator program (a sample screen in shown in Fig. 8), and each field of each screen is provided with further clarification if the HELP key is pressed.

As each screen is processed, the answers are reviewed for consistency with all previously provided answers to ensure that no illogical constructs are placed in the parameter table. Finally, after all questions have been answered, the answers are converted into the executable parameter table and written to a transaction library, a device file, or a disc file for later use by the transaction monitor program. The final operation performed by TGP is to provide the system designer with a listing of the transaction generated, including a model of the data capture terminal prompting-light label.

Fig. 9 shows the executable parameter table structure. Each executable transaction consists of a transaction header followed by up to twenty states consisting of three components each. Each component is a variable-length record with a fixed minimum length. The minimum-length component contains information about optional extra members of the component. The three components and their contents

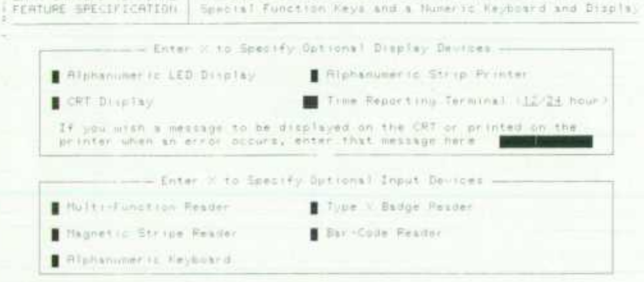

Fig. 4. *Typical transaction generator program (TGP) screen.*

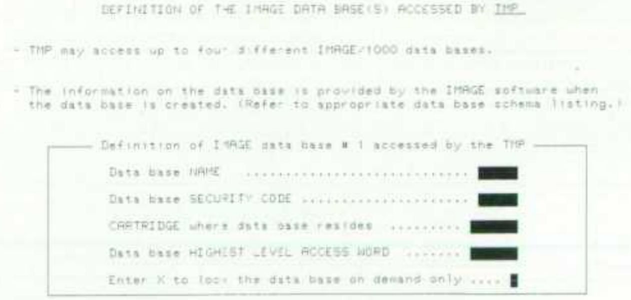

Fig. 5. *Typical transaction monitor program generator (TMPGN) screen.*

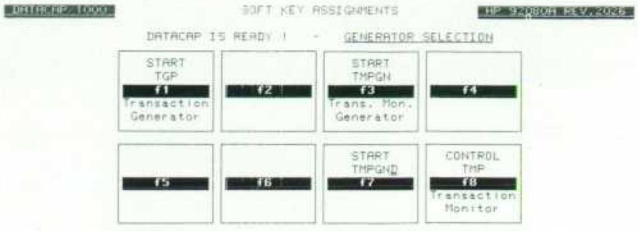

Fig. 6. *Typical data capture monitor (DCMON) screen.*

are:

State General Information
- Forward pointer to the next state
- Answer data type
- Data capture terminal configuration information
  - Prompting light
  - Alternate input device configuration
  - Display and printer use

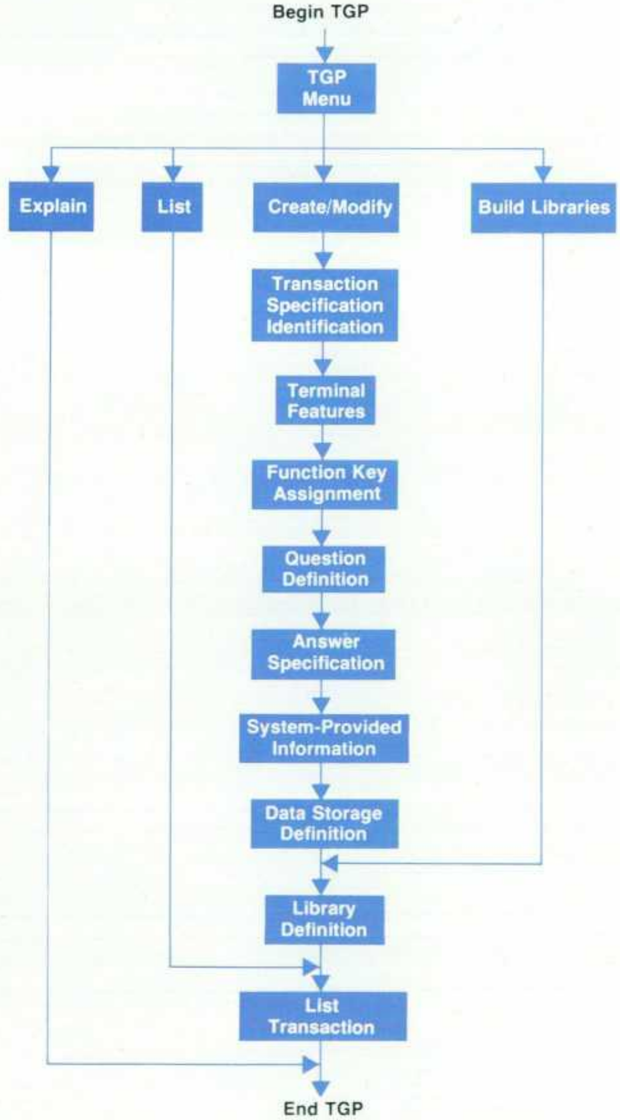

Fig. 7. *Basic transaction generator program flow.*

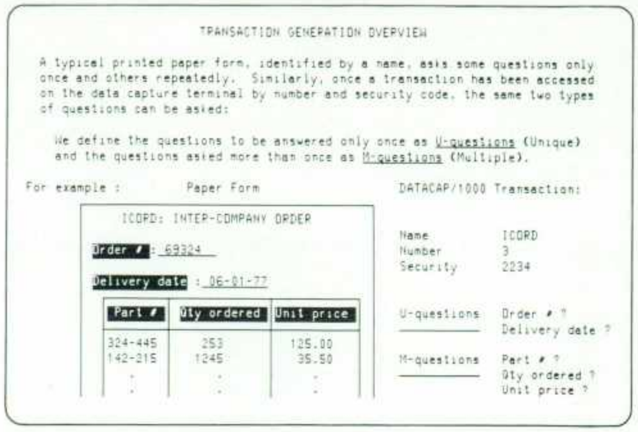

Fig. 8. *Tutorial screens like this one are built into the transaction generator program to provide the user with information about the system. A HELP key provides further clarification.*

Display Information
- User-written subroutine name, if any
- Display item and set name, if any
- Data capture terminal configuration information
  - Prompting light
  - Display and printer use

Edit Specification
- Range or mask depending on data type
- Default value
- User-written subroutine name, if any
- Automatically generated IMAGE edits, if any

In addition to the states consisting of these three components, there are three other states. The special function key state provides information about the data capture terminal special function keys. The storage state includes the disc file storage name, the device file storage name, the name of the user-written subroutine name, if any, and a list of IMAGE/1000 operations to be carried out, if any. The off-line printout state includes information that allows a summary of the transaction to be printed on the data capture terminal printer upon completion of the transaction.

Following this executable form of the transaction are all of the original answer buffers. This allows later modification of the transaction by TGP without uncompiling the executable form.

The Transaction Monitor Program Generator (TMPGN)

The TMPGN provides the definition of data capture terminal logical unit numbers, user provided subroutine relocatable names, and IMAGE/1000 data base names. Like TGP, TMPGN prompts the user through a set of interactive displays on the HP 2645A System Console. Again like TGP, the answers are maintained in a file for later modification of the TMP and a convenient listing is available to the system designer to review the application. Once all the answers have been provided, TMPGN generates the appropriate relocatable members of the DATACAP/1000 package and calls the RTE-IVB loader to relocate them into the structure shown in Fig. 10.

Taking advantage of the environment provided by a specially developed terminal management tool (see box,

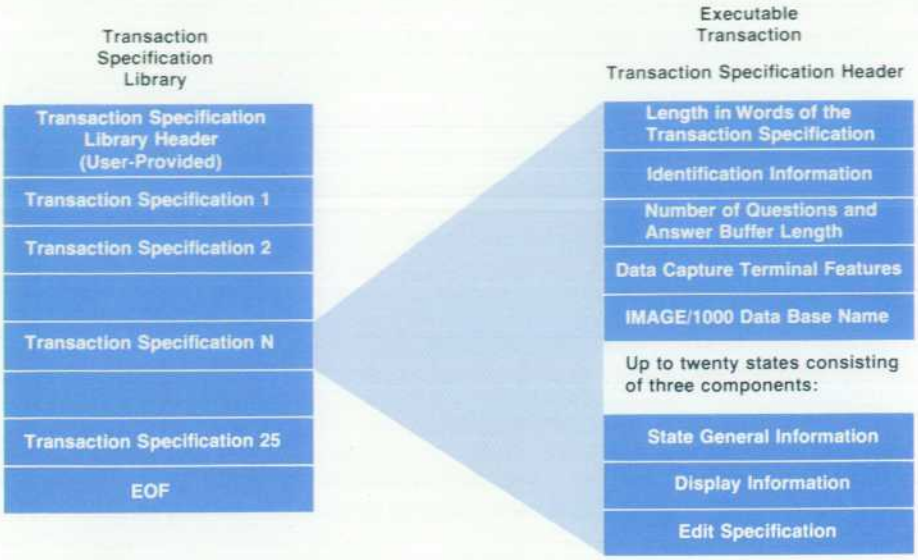

Fig. 9. *Each transaction specification in the TS library consists of a transaction header followed by up to 20 states consisting of three components each.*

page 30), each function of DATACAP/1000 can be handled by a separate subroutine:

- ZTMP This module contains the main logic of the TMP. When the operator of a data capture terminal requests a specific transaction, ZTMP calls the transaction specification manager (TSMG) subroutine to return the first state of the transaction specification. ZTMP then uses information from the first state to configure the special function keys, prompting lights, input devices, and so on. Once the appropriate input has been received from the operator (as determined from the edit component of the state), ZTMP calls TSMG for the next state, and so on, progressing to the storage state upon completion of the transaction by the operator.
- TSMG This module maintains the working set of transaction specifications while the TMP is running. TSMG uses its end-of-partition space as a buffer to hold several transactions, each of which can be called by any of the terminals on-line to the TMP (given that the operator knows the transaction identification number and security code). This buffer space is set at a minimum of 7500 words. Since the typical length of the executable form of the transaction is 250 words, about 30 transactions can be managed by TSMG simultaneously.
- TSE The transaction set editor is used by the system operator to load and unload transaction specifications from the working set. It also can be used to check on the status of the working set (that is, which transactions are active), the transaction library from which they were loaded, and so on.
- IOM7X The two modules IOM75 and IOM70 are functional drivers used by ZTMP to interface to the RTE system's terminal drivers DVR07 and DVA47, respectively. These functional drivers relieve ZTMP of the need to concern itself with the construction of escape sequences and the specifics of the two different drivers' calling sequences.
- STORX The two modules STORA and STORB are used to handle the two distinct types of data storage available to the DATACAP/1000 system designer. STORA handles the passing of data and offset information to the DATACAP/1000 data base hand-

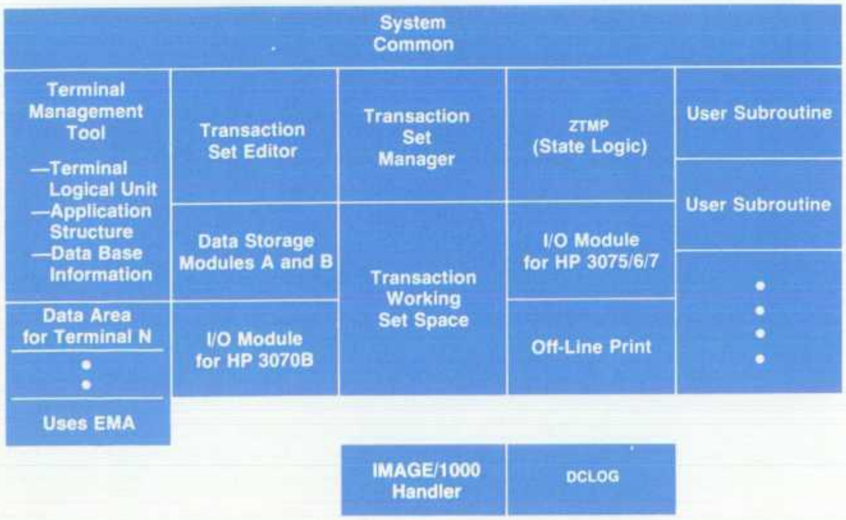

Fig. 10. *Once it has all of the answers it needs, the transaction monitor program generator generates the relocatable members of DATACAP/1000 and calls the RTE-IVB loader to relocate them into the memory structure shown here.*

A Terminal Management Tool

by Francois Gaullier

A user faced with the problem of running the same application concurrently on a number of terminals in the RTE-IVB environment may use the following techniques:

1. Write the program for one terminal, duplicate it for N terminals and run these programs concurrently. The amount of main memory available limits the number of terminals that can be supported with an acceptable response time, since swapping to the disc may be necessary.
2. Write the program for N terminals. In this case, the user must keep track of the program flow for each terminal, must access most variables through arrays, and cannot easily segment the program or spread processing among several programs. Enough memory must be provided for the maximum number of terminals even if only a few are actually being used. The development and maintenance of this type of program will be more complicated because of the bookkeeping required to keep track of each user's data.

The ideal solution to the problem of developing and maintaining large, multi-user applications is to write a single set of reentrant programs to handle all users. In a reentrant environment, applications program development and maintenance are reduced to the single-user case. Program swapping is minimized because only the users' data needs to be swapped. (Instruction spaces do not need to be swapped out, just overlaid.)

Since no facilities for large-scale reentrancy existed in RTE-IVB, a primary objective of the DATACAP/1000 project was to develop a terminal management tool to provide a reentrant environment for RTE. It was hoped that this tool would both ease the task of developing a multiterminal application like DATACAP and provide on-going benefits by reducing the amount of main memory required to support the application. DATACAP/1000 was developed using the features provided by this terminal management tool. The tool is also supplied as part of the DATACAP software, so that the final application running on a customer's system takes advantage of and depends upon the environment provided by the terminal management tool. Features of this environment are:

1. Code and data separation in FORTRAN with the capability of dynamically specifying which blocks of data are to be saved and restored when the program is reentered.
2. An easy means of spreading the processing among several program units in different partitions, that is, an easy means to do subroutine calls across partition boundaries. This eliminates the need for segmentation, and if sufficient main memory is available, lets the entire application reside in memory.
3. Recursive capabilities for specialized needs.

Programming Conventions

To use the features of the terminal management tool, the programmer must observe the following conventions:

1. I/O requests to the interactive terminals must be done through a special set of subroutines, TMLIB.
2. Code and data must be separated by declaring the latter to exist in the area of main memory designated as common.
3. I/O requests that wait for completion should be avoided whenever possible.
4. Calls to subroutines and processes that issue service requests must be done through TMLIB.
5. All programs in an application must be coded as subroutines.

The type of reentrancy provided by the tool can be termed "breakpoint reentrancy," because the programs that run in its environment can be reentered only when they issue a service request through TMLIB. Breakpoint reentrancy is quite suitable for large, multiterminal, interactive applications, because most of the execution time is spent issuing and waiting for I/O operations, which are performed by the tool's service requests.

Data Segments

The most important requirement for reentrant programming is the separation of code and data. The most convenient facility for accomplishing this in the RTE-IVB languages (FORTRAN, Assembly) is the common memory area. An added benefit of having data exist in common memory is that subroutines are provided with a very efficient means of communicating. Unlike accessing arrays, accessing variables in common is without any overhead. These two reasons are why common memory was chosen as the vehicle for separating code and data in the terminal management tool.

The primary restriction, then, for use of the terminal management tool is that a tool subroutine must assign all of its reentrant variables to common memory. Not all variables must be assigned to common memory, just those whose values are to remain intact after service requests, because it is only at these points that the subroutine can be reentered.

Since it is not necessary to preserve the integrity of all reentrant variables at all times (there is some overhead associated with this), the data segment is divided into six common blocks, which may be enabled or disabled according to the user's needs. When a common block is enabled, the data that it contains will be saved when a tool service request is issued and restored after the request completes. If a block is disabled, the integrity of its data is not preserved across the service request. By grouping the reentrant variables into common blocks and selectively enabling them, the user can reduce the overhead required by a program and the amount of main memory used at any given time. Only the data needed for the portion of code that is currently executing will be saved.

The library of subroutines, TMLIB, was designed to provide an easy interface between the user application program and the terminal management tool. TMLIB includes the following services:

- TMDFN To manage memory allocation
- TMCBL To manage memory allocation
- TMCBE To manage access of data blocks
- TMCBD To manage access of data blocks
- TMSUB To call a tool subroutine
- TMPRO To launch a son process from an executing one
- TMLUL To lock a logical unit
- TMIO To perform
- TMRD
- TMWR
- TMCTL I/O
- TMBWR
- TMBCT
- TMWRD operations
- TMPZ To suspend execution of a process for a given period of time
- TMSCH To schedule a non-tool program
- TMCST To place a process in critical/normal state
- TMSTP To stop the application in an orderly way

- TMSAB To abort the application

IMAGE/1000 Access

To access an IMAGE/1000 data base with the full capabilties (GET, ADD, DELETE, UPDATE) the programmer must open it in mode 3 (exclusively) or in mode 1 (shared read/write) and lock it, which prevents other programs from concurrently accessing (writing to) the data base. Under these conditions one way to have multiprogram access to a data base is to centralize all requests to IMAGE in a single program. IMAGE/1000 access through the tool uses this concept.

Given the above feature, the programmer is faced with the difficulties of managing all the different operations that may be performed simultaneously on the data base by all the programs (update of the same entry by more than one program at the same time, deletion of a chain entry while another program is going through the chain, etc.). One way to solve these problems is to implement a locking mechanism. The most sophisticated locking mechanism is to lock at the entry level in the IMAGE data base. The terminal management tool offers this capability to the user.

The tool also automatically performs the DBINF calls to save and restore the run table when necessary for each process, thereby saving the user the worry of keeping track of the run table when repeatedly accessing the data base (chain or serial read).

All calling sequences to the DATACAP/1000 IMAGE handler implemented using the terminal management tool are compatible with the IMAGE/1000 calling sequences with the addition of a single-word parameter to the calls analogous to DBFND and DBGET, which allows specification of the record locking action to be taken. These calls are accessible by the user through a library of routines, %XMLIM, whose members are the same as full IMAGE/1000 DBMS calls (DBYYY(parameters)) but whose entry points are XBYYY(parameters). To keep the demand of the terminal management tool IMAGE/1000 handler on system resources to a reasonable level, the following limitations are imposed on the user:

1. Calls for locking and unlocking the data base (DBLCK and DBUNL) are not provided.
2. Positions in a data set (current record, chain, etc.) are not remembered when a different data set is accessed. It is necessary for the user to save the run table for a data set before accessing a second data set, then restore the run table for the first data set to access it starting at its current position.
3. The maximum entry length allowed is 512 words instead of 2048 words.
4. The maximum number of data bases that can be opened simultaneously is eight (combined total in DATACAP/1000).
5. All data bases that are to be accessed must be defined in TMPGN.

ler, while STORB is responsible for generalized data storage to disc files, device files (magnetic tape), or user storage subroutines.

IMAGE/1000 Handler This module provides a front end to the standard IMAGE/1000 DBMS, specifically establishing a record locking mechanism and a link to the DATACAP/1000 transaction logging module.

DCLOG provides a means for writing standard, recoverable records and prevents the terminal user from proceeding with a second transaction until the first is safely logged to a disc or device file. DCLOG is complemented by the recovery utility DCRCV that can be used to recover an IMAGE/1000 data base that has been corrupted.

Acknowledgments

The DATACAP/1000 product was conceived at Hewlett-Packard's division in Grenoble, France in 1975 and was originally released for sale from Hewlett-Packard's Data Systems Division in 1978. It has undergone several revisions to incorporate new data capture terminals, revisions to the RTE operating system and IMAGE/1000 DBMS, and flexibility enhancements. The design team has included Marc Brun, Elizabeth Clark, Francois Gaullier, Scott Gulland, Ben Heilbronn, Tom Hirata, Jean-Charles Miard, Miles Nakamura, Daniel Pot, and Steve Witten. Production personnel have included Chengwen Chen, Ron Schloss, and Alice Woo. Product assurance was the responsibility of Clemen Jue, Julie Knox, Doug Larson, and Elizabeth Gates.

Francois Gaullier

Francois Gaullier received his diploma—equivalent to a master's degree—in 1971 from Ecole Speciale de Mecanique et d'Electricite in Paris. He joined HP that same year as a systems engineer. In 1975 he became a software design engineer, developing RTE drivers and modules of DATACAP, and he's now a lab project manager at HP France in Grenoble. Francois was born in Paris and now lives in St. Egreve near Grenoble. He's married and has two daughters. He enjoys hiking, skiing, and listening to classical and country music, and he recently designed and built a microprocessor-controlled model railroad.

Steven H. Richard

Steve Richard graduated from Stanford University in 1969 with BS and MS degrees in industrial engineering, and joined HP the same year. After four years of production engineering, followed by four more years of programming and EDP management, he became project manager for DATACAP/1000, a post he still holds. Steve is a member of AIIE and ACM and is working towards his MS degree in computer science at the University of Santa Clara. An Arizona native, he is married, has two sons, and lives in Los Altos, California. His interests include church activities, bicycling, downhill and cross-country skiing, sailing, and cooking breakfast.

SPECIFICATIONS

HP Model 92080A DATACAP/1000 Data Capture Software

MAXIMUM NUMBER OF TERMINALS: An HP 1000 Computer or System can support up to 56 HP data capture terminals. The number of terminals supportable by DATACAP/1000 depends upon the number of transactions per hour to be processed, i.e., response time required, on transaction complexity, such as IMAGE/1000 data base activity, user subroutines, etc., and on system main memory size.

COMMUNICATION WITH IMAGE/1000 DATA BASE: Transaction Monitor Program (TMP) can retrieve data from, and record data in up to four IMAGE/1000 data bases. No transaction specification can access more than one IMAGE/1000 data base. Any user program has read/write access to a data base that is under DATACAP control.

COMPATIBILITY WITH SESSION MONITOR: DATACAP/1000 does not use or require the Session Monitor in RTE-IVB and in certain respects may not be compatible with the Session Monitor. Where multi-user access to the RTE-IVB system is required concurrently with DATACAP/1000 operation, the Multi-Terminal Monitor is recommended instead of the Session Monitor. The DATACAP Configuration Guide (92080-90003) documents the particulars regarding DATACAP/1000 compatibility with Session Monitor.

DATA CAPTURE SYSTEM REQUIREMENTS: HP 92080A DATACAP/1000 is supported only on the 92068A RTE-IVB operating system, and therefore has the same minimum system requirements as the 92068A, plus additional requirements listed below.

DATA BASE SOFTWARE: If communication with an IMAGE/1000 data base is desired, 92069A IMAGE/1000 is required.

DATA CAPTURE TERMINALS: At least one (and up to 56) of any of the HP 3075A, 3076A, or 3077A terminals.

MAGNETIC TAPE: A magnetic tape unit is strongly recommended if IMAGE/1000 is used.

SYSTEM CONSOLE: HP 2645A, 2647A, 2648A, 2649B, 2649C, or 2649G system console on which DATACAP/1000 is readied for use must connect to the system via the 12966A+001 interface and must have the following options and accessories.

OPTION 007: Mini cartridge I/O (not required on 2647A/49G).

OPTION 032: Substitutes Extended Async Comm. Card for standard comm. card.

13231A Display enhancements with line drawing character set.

FAST FORTRAN PROCESSOR FIRMWARE: 12977B or 13306A, depending on computer model.

RTE-IVB SYSTEM MEMORY REQUIREMENTS: The RTE-IVB system configured to support DATACAP/1000 has the following memory requirements.

SYSTEM CODE AND TABLES: 56K bytes.

SYSTEM AVAILABLE MEMORY: 12K bytes.

SYSTEM COMMON FOR DATACAP USE: 4K bytes.

DATACAP/1000 MEMORY PARTITION REQUIREMENTS: The following partitions are required for DATACAP/1000:

DATACAP/1000 APPLICATION CODE: 122K bytes.

BUFFERING FOR DATA CAPTURE TERMINALS: 54K-40K bytes (see Table 1), using RTE-IVB mother partition.

TABLE 1. Memory buffering requirements by number of Data Capture Terminals

No. of Terminals	Buffer Size	No. of Terminals	Buffer Size	No. of Terminals	Buffer Size
1- 5	54Kb	21-25	194Kb	41-45	334Kb
6-10	89Kb	26-30	229Kb	46-50	369Kb
11-15	124Kb	31-35	264Kb	51-56	404Kb
16-20	159Kb	36-40	299Kb		

OPTIONAL MEMORY PARTITIONS: The following partitions are required for optional DATACAP/1000 capabilities.

EACH IMAGE/1000 DATA BASE ACCESSED: 56K bytes.

FOR INTERFACING WITH USER SUBROUTINES: 10K bytes, variable, for the user subroutines.

TRANSACTION GENERATOR PROGRAM: 40K bytes, to avoid swapping if creating transactions concurrently with real-time data capture operations.

DS/1000 NETWORK COMMUNICATIONS: 6-25K bytes, depending upon remote access capabilities supported.

PRICES IN U.S.A.:

92080A DATACAP/1000 Data Capture Software Package, $5000.

92080R Right to Copy DATACAP/1000 for use on an additional computer system, $2000.

MANUFACTURING DIVISION: DATA SYSTEMS DIVISION
11000 Wolfe Road
Cupertino, California 95014 U.S.A.

Hewlett-Packard Company, 1501 Page Mill Road, Palo Alto, California 94304

Bulk Rate
U.S. Postage
Paid
Hewlett-Packard
Company

HEWLETT-PACKARD JOURNAL

JANUARY 1981 Volume 32 • Number 1

Technical Information from the Laboratories of Hewlett-Packard Company

Hewlett-Packard Company, 1501 Page Mill Road
Palo Alto, California 94304 U.S.A.
Hewlett-Packard Central Mailing Department
Van Heuven Goedhartlaan 121
1180 AM Amstelveen The Netherlands
Yokogawa-Hewlett-Packard Ltd., Suginami-Ku
Tokyo 168 Japan

CHANGE OF ADDRESS: To change your address or delete your name from our mailing list please send us your old address label. Send changes to Hewlett-Packard Journal, 1501 Page Mill Road, Palo Alto, California 94304 U.S.A. Allow 60 days.