

JUNE 1976

HEWLETT-PACKARD JOURNAL



Third Generation Programmable Calculator Has Computer-Like Capabilities

A new programming language, HPL, helps the user apply the many technological advances in this personal computing machine to a wide variety of computation and control problems.

by Donald E. Morris, Chris J. Christopher, Geoffrey W. Chance, and Dick B. Barney

A POWERFUL MEDIUM-PRICED desk-top programmable calculator with many features previously found only on minicomputers, the new 26-pound HP 9825A Calculator (Fig. 1), is designed primarily for use in the fields of engineering, research and statistics. The new calculator's speed, interfacing abilities and computer-like features make it particularly well suited for use as the controller of an instrument system, for pilot process control applications, for remote data collection, and for production control. It can also be used as a powerful stand-alone computing tool.

Significant contributions that provide major user benefits include two-level priority interrupt, live keyboard, direct memory access with input speeds up to 400,000 16-bit words per second, high-performance bidirectional tape drive, multidimensional arrays, automatic memory record and load, and extended internal calculation range ($\pm 10^{511}$ to $\pm 10^{-511}$). Some of these are standard features and others are available in optional plug-in read-only memories (ROMs).

The 9825A uses a high-level programming language called HPL. This formula-oriented language is easy to learn and is designed for controller applications as well as for data processing. HPL provides for subroutine nesting and flags, and allows 26 simple variables and 26 multidimensional array variables, limited only by the size of the calculator memory. Editing of lines and characters is simple, and error locations are identified by a flashing cursor in the display. Fixed and floating point formats can be set by the user from the typewriter-like keyboard.

The keyboard has twelve special function keys that, combined with the shift key, can handle 24 different operations. These keys help in program writing and in peripheral and instrument control. They can serve as immediate-execute keys, as call keys for subroutines, and as typing aids.

With the live keyboard, never before found on a desktop calculator, the user can examine and change program variables, perform complex calculations, call subroutines, and record and list programs while



Cover: Two important aspects of the 9825A Calculator story are its high-performance NMOS LSI processor, pictured at the top of the cover photo and described in the article on page 15, and the 9825A's powerful system-controller capabilities, sym-

bolized by the HP-IB-compatible instruments in the lower background and described in the article on page 2.

In this Issue:

- Third-Generation Programmable Calculator Has Computer-Like Capabilities*, by Donald E. Morris, Chris J. Christopher, Geoffrey W. Chance, and Dick B. Barney **page 2**
- High-Performance NMOS LSI Processor*, by William D. Eads and David S. Maitland **page 15**
- Character Impact Printer Offers Maximum Printing Flexibility*, by Robert B. Bump and Gary R. Paulson **page 19**
- Mid-Range Calculator Delivers More Power at Lower Cost*, by Douglas M. Clifford, F. Timothy Hickenlooper, and A. Craig Mortensen **page 24**

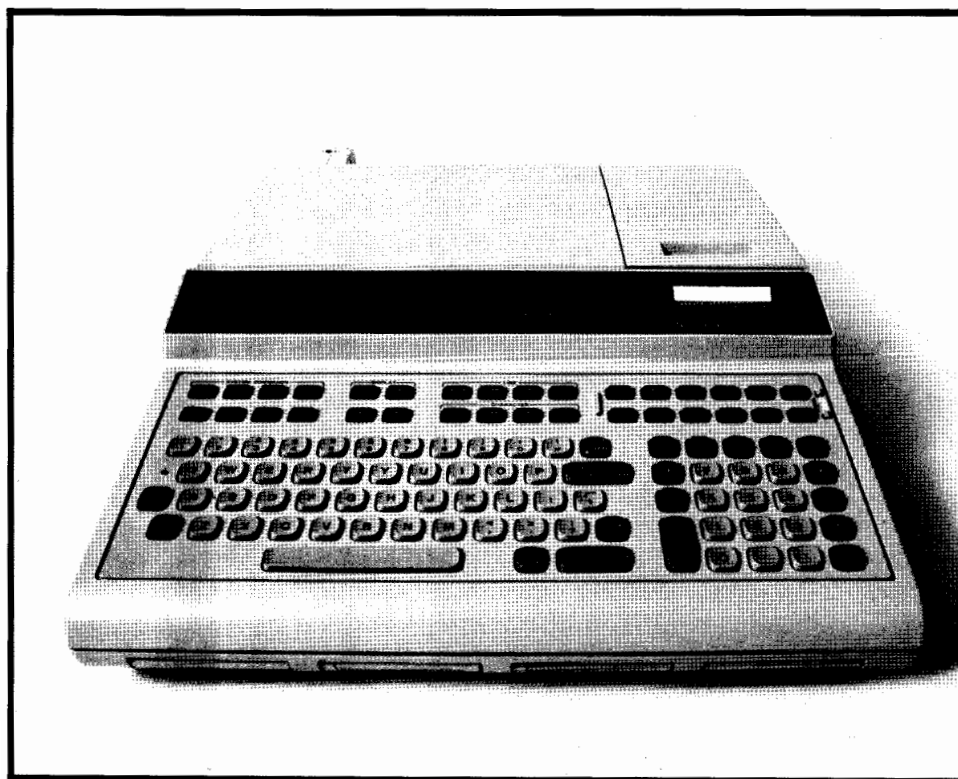


Fig. 1. Model 9825A Calculator can be used as a stand-alone programmable calculator or as a system controller. Major new features include two-level priority interrupt, direct memory access, live keyboard, multidimensional arrays, high-performance tape drive with automatic memory record and load capability, and extended internal calculation range. The keyboard layout is based on that of a standard typewriter with the addition of a number pad, system command and editing keys, and twelve user-definable keys. The programming language is called HPL.

the calculator is performing other operations.

Interrupt capability, available in the Extended I/O ROM, permits the calculator to act as a controller for several instruments or peripherals that require attention at unpredictable rates or times.

A 32-character LED display and a built-in 16-character thermal printer provide alphanumeric readout including both capital and lower-case letters. Some European and Greek characters are also available in an optional ROM.

The display was specifically designed for the 9825A Calculator. Each character has a maximum height of 0.37 cm and is formed on a 5-by-7 dot matrix. Four characters are packaged together in a dual in-line package. The dots are red light-emitting diodes fabricated on a monolithic substrate. Also on this substrate are two bipolar ICs, each of which contains two seven-bit shift registers for holding the column information and decoding and enabling the appropriate column. The display is treated as a peripheral device by the processor. Display refresh and flashing of the insert and replace cursors are all handled by hardware. Therefore, no loss of processing speed is incurred by having the display active during program execution.

The printer is the same as the 9815A Calculator's built-in printer (see article, page 24).

The built-in tape cartridge holds 250,000 bytes and has a 2,750-byte-per-second data transfer rate. Using the cartridge, the operator can perform rapid

memory load and record operations. In the event of a power failure, this feature, coupled with the autostart feature, can reload the calculator's memory with whatever had been recorded on the tape and continue from that point. Or, while a long program is running, an operator can stop it, record the entire memory onto the cartridge, run another program and then reload and continue the first program from the point of interruption. A 90-ips search and rewind speed and a 22-ips read/write speed give an average access time of six seconds. The tape drive provides automatic verification during recording, and the calculator always knows the position of the tape, so it saves time by being able to search at high speed in either direction for the next file.

The Language: HPL

The 9825A's language, called HPL, is a powerful formula-oriented language that provides features of BASIC, FORTRAN, ALGOL and PL/I. HPL's features minimize the translation process that a user has to go through to program an algebraically formulated problem.

HPL's basic building block is the statement. The language allows several statements per line; this reduces memory requirements and increases execution speed. In HPL, key words are written in lower-case characters and variables in upper-case or capital letters. Some of the language's features are demonstrated in the following program, which solves for the

of $(-B \pm \sqrt{(BB-4AC)})/2A$.

```
0: enp "value of A", A, "value of B", B, "value of C", C
1: if 4AC>BB;gto "imag"
2: prt "real roots"; spc 1
3: prt  $(-B + \sqrt{(BB-4AC)})/2A$ ; spc 1
4: prt  $(-B - \sqrt{(BB-4AC)})/2A$ ; spc 4; jmp -4
5: "imag":
6: prt "complex roots"; spc 1
7: prt "real", "imaginary"; spc 2
8: prt  $-B/2A, \sqrt{(4AC-BB)}/2A$ ; spc 1
9: prt  $-B/2A, -\sqrt{(4AC-BB)}/2A$ ; spc 4; gto 0
10: end
```

Line 0 allows the user to enter and print (enp) the values of the variables A, B, and C. When the RUN key is pressed, the first message, "value of A", will appear on the display and be printed on the printer. After typing the answer, the user presses the CONTINUE key to resume program execution. In lines 3 and 4, the solution of the equation is programmed in a very simple manner similar to the pencil-and-paper solution. The implied multiply feature reduces the number of parentheses and "*" multiplication signs that would otherwise be required. Lines 1, 4, and 9 contain branching statements with label, expression, and line number arguments.

Arrays

When dealing with large arrays of numbers, it is normally necessary to reserve a memory space large enough to solve the problem. HPL provides dynamically allocated multi-dimensional arrays with selectable lower bounds. Thus, the user can easily specify his data memory requirements. The dimension statement `dim A[0:N]` reserves memory space for $N+1$ numeric elements. The first element of the array is `A[0]` and the last is `A[N]`.

Multiple arrays can be declared in the same "dim" statement. For example:

```
dim A[0:N],B[3,3],K[3+L:2X,Y,Z].
```

In this case, the system will reserve memory space for arrays A, B, and K. Array B is a two-dimensional (3×3) array. Array K is a three-dimensional array. The size of the first dimension is $2X - (3+L) + 1$. Y and Z specify the size of the second and third dimensions of the array K respectively. The first element of the array is `K[3+L,1,1]` and the last is `K[2X,Y,Z]`.

Strings

Many problems are numeric and can be solved using traditional mathematical functions and numeric arrays. Others are presented as character "strings" and can best be handled by HPL's string functions and string arrays. Character strings are assigned to string variables, which are denoted by a capital letter followed by a "\$" sign. The following example demonstrates string arrays and some of the string

functions.

```
0: dim A$[80]
1: "BOB " & "JONES" → A$
2: A$ & " 2100 McKinley Ave." → A$
3: dsp A$; wait 500
4: dsp cap(A$)
5: end
```

In line 0, the "dim" statement is used to declare the string array A\$, which is 80 characters long. String arrays can be of one or two dimensions. The lower bound is one and the upper bound is given by the dimension size specification. In line 1, the character strings "BOB " and "JONES" have been connected with the concatenation operator "&" to form a new string "BOB JONES" which is assigned to the string array A\$. In line 2, the present contents of A\$ are appended with the string " 2100 McKinley Ave." and the result is reassigned to the same string array. In line 3, the system will display the contents of the string array as: BOB JONES 2100 McKinley Ave. Then, after waiting for one-half second, it will display the same information with upper-case characters throughout. The "cap" function transforms all lower-case alpha characters to upper-case characters without modifying the original string stored in A\$. This function simplifies string comparisons and string sorting.

Additional string functions calculate the length of a string, the position of a substring within a string, and the value of a substring of numeric characters.

The relational operators, =, <, and > may be used in string comparisons, as the following example demonstrates.

```
0: dim A$[10]
1: "AB" → A$
2: if A$ < "ABC"; prt "AB<ABC"
3: if "AC" > A$; prt "AC>AB"
4: end
```

Both relational operations are true, so lines 2 and 3 will be executed and the printer will print:

```
AB<ABC
AC>AB
```

Looping and Branching

As seen in the quadratic solution program above, the absolute and relative "go to" and "jump" statements provide simple and computed unconditional transfers of program control. These, in conjunction

with the relational operators, can be effective in developing controlled loops that allow repetitive execution of program segments. In many cases the looping task can be simplified by the use of HPL's "for-next" statements, similar to those in the BASIC language. For example:

```
0: for I=0 to 90 by 10
1: dsp I,sin(I); wait 1000
2: next I; end
```

The "for" statement initiates the loop and specifies the control variable's initial value, 0, and final value,

9825A Product Design

The product design of the 9825A Calculator was based on the belief that a product, like an organism, interacts with its environment, imposes demands, and creates relationships between itself and all who come in contact with it during its odyssey of development, production, testing, shipment, and ultimate maturity in the field. Requirements that consistently appear are small size, light weight, and ruggedness. It is also desirable that the internal parts be sensibly arranged and that subassemblies be easily accessible for assembly and service.

These needs led to the first major concept in the product design, that the mechanical elements must provide a maximum of functions with minimum structure and weight. Traditional notions of sheet metal card cages mounted in an external shell were rejected because of tolerance, size, and weight problems. After considering many manufacturing methods, structural foam molding appeared to best solve the problems of providing a combination of external shell with detailed internal structure for mounting and locating the various subassemblies.

The structural foam case, however, did not provide RF interference shielding that previous metal cases had provided and was determined to be necessary. To meet the RFI requirements, a conductive coating is applied to the inside of the case and connected to calculator ground. Two techniques are used, flame spraying aluminum and painting with a silver colloid conductive paint, both of which have proved to be satisfactory.

Four case parts are foam molded; two of them, the base and the keyboard housing, are the two major subassemblies. The printed circuit boards are stacked horizontally in the base with a new HP-designed printed circuit board stacking hinge and are interconnected with flat cable along the hinged edge, eliminating the need for a mother board (see Fig. 1). The flat cable also allows physical separation of the two major subassemblies while leaving them connected electrically.

The printed circuit boards in the stack have two of the new hinges attached to one side. These provide a rigid mounting when closed, but also allow random access to any of the boards by hinging them up individually or together (Fig. 2). Access to both sides of a board or to many boards at once is possible. The hinges slide apart for easy board exchange without tools. Thus the calculator can be completely opened and continue running.

The entire package is designed to be assembled and tested



Fig. 1. 9825A Calculator is fully operational when opened for servicing. Printed circuit boards are hinged to provide access to both sides. Plug-in ROMs at front and side were pulled halfway out for this photo.

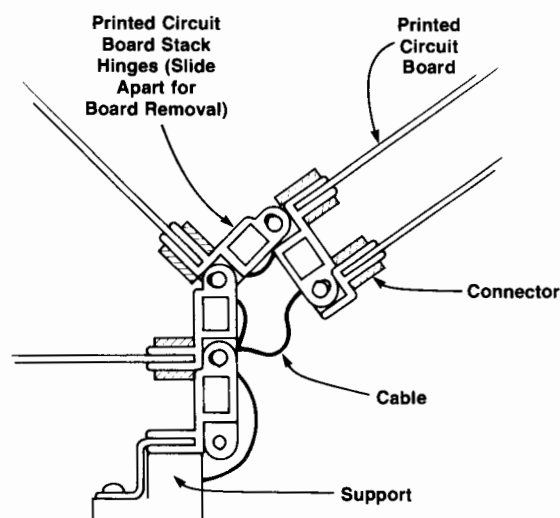


Fig. 2. Printed circuit board stacking hinge detail.

right side up. Hence there are no more knee and back exercises required to remove or install access screws. Two screws removed from the rear free the top shell, which is interlocked to cause power disconnect in case of inadvertent opening. Loosening (but not removal) of two screws in the base allows the keyboard housing to slide forward and be set aside. At that point, any of the secondary subassemblies (printed circuit boards, printer, keyboard, and so on) are accessible immediately and independently.

One of the subassemblies in the keyboard housing is the keyboard, which was developed specifically for the 9825A and 9815A Calculators. The goals were to provide a positive key action, to keep the front profile of the machine low, and to lower the cost over previously used switches. The "buckling strip" technology originally used in HP hand-held calculators was modified to suit a larger keyboard, primarily by increasing key spacing and key cap travel. This new keyboard is only 0.79 cm thick, saving about 1600 cm³ and about half the cost per switch of previous 9800-Series keyboards.

The first step of the keyboard manufacturing process is the preparation of the printed circuit board that carries the key-switch scan matrix and the pads for the individual switches. The surface of the board is plated with bright alloy that gives corrosion proofing and the weldable surface needed for the next step. Then a beryllium-copper strip plated with bright alloy is cut into 0.4×1.65-cm pieces, which are electrically welded to the printed circuit board in such a way as to give an arch 0.06 cm high. The buckling of this arch gives tactile feedback to the user. This keyswitch board is then securely staked to a plastic insert that contains the keycaps. Each keycap is hinged and has an actuator that depresses the arch beneath it.

The operating system ROM is housed in a separate package that is accessible from the outside. This allows testing independently of the mainframe final assembly and very easy field replacement. Critical internal components are placed directly in front of the fan to achieve more than adequate cooling in spite of a high total heat dissipation inside the package. Internal mounting bosses, reference surfaces, and guiding grooves are molded into the structural parts to provide accurate location of the subassemblies with a minimum of hardware and labor. As a result, such elements as the display, tape transport, and printer are automatically located once they are installed.

90, and the step size, 10. All program statements enclosed between the "for" and "next" statements will be executed ten times before the program terminates with the "end" statement.

Interactiveness: Live Keyboard

The HPL command set by itself does not make the calculator usable. The human-machine interface is a very important aspect of calculator design. The 9825A provides a number of interactive features that allow the user to receive information and control the calculator system in a very flexible manner.

Normally, when a calculator is executing a program, the keyboard is disabled. In the 9825A, the keyboard is active and the user is allowed to perform certain tasks without stopping the job currently in progress. This "live keyboard" feature allows the user to evaluate algebraic expressions, examine the status of the running program, control the program flow, and execute specific subprograms without aborting the main task.

The following example contains two subprograms. The first, lines 0 to 5, approximates the value of π according to the series, $1 + 1/3^2 + 1/5^2 + 1/7^2 + \dots = \pi^2/8$. The second, lines 6 to 9, evaluates the factorial of an integer.

```

0: "π APPROXIMATION":
1: 1→A→B;0→C→N
2: "L":B+2→B
3: 1/BB+A→A;√(8A)→P
4: C+1→C;if N#0;gsb "FACT"
5: gto "L"
6:
7: "FACTORIAL EVALUATION":
8: "FACT":1→X;for I=1 to N
9: XI→X;next I;dsp X;0→N
10: ret

```

While the calculator is busy executing lines 2 through 5, the user can sample the value of the counter C by simply entering C EXECUTE. The value of C will appear on the display, indicating the number of terms accumulated thus far. All mathematic functions and most of the HPL commands can be executed under live keyboard. These include variable assignments and execution of subroutines designed for a specific job. While the calculator is continuing the approximation of π , the user can enter $n \rightarrow N$ EXECUTE, where n is the integer whose factorial is desired. Upon execution of line 4, the calculator will perform a subroutine jump to "FACT", evaluate the factorial of n , display the result and return to the approximation program, which was temporarily suspended.

The keyboard is serviced as an interrupting device while the program is running. To initiate live keyboard execution, an EXECUTE command must be given. The command will be logged and interrogated when the end of the current program line is reached. If no errors are encountered, the live keyboard operation

will be completed and the system will return to the main task to begin execution of the next program line. If an error has been detected during the live keyboard operation, it will be displayed and the calculator will abort the live keyboard job and resume the main task.

Data Entry

In the quadratic solution program, the "enp" (enter and print) statement allowed the user to enter a value for the specified variable and print the request and the answer. If printing is not wanted, the "ent" (enter) statement can be used instead. Entering data, numerics, or character strings is a very important function of a computing system, so HPL makes it as interactive as possible. As shown in the quadratic example, the enter statement causes the calculator to prompt the user and accept data. In the numeric case, the data can be given as a number or as an expression, which will be evaluated and the result supplied as an answer to the enter statement. If an error is detected during the syntax analysis or evaluation of the expression, the error will be displayed and the calculator will remain in enter mode. While the calculator is in the enter mode, the user can execute expressions and most of the HPL commands without affecting the pending enter statement.

Often the user will enter array data element by element. In such cases, it is important that the enter prompts provide relevant information. The 9825A accomplishes this very simply as shown by the following program.

```

0: dim A[10]
1: for I=1 to 10;ent A[I];next I
2: end

```

The first time the "ent" statement is executed, the display will show the prompt "A[1]?". The following prompts will be "A[2]?", ..., "A[10]?". Thus the array element I or an equivalent expression is evaluated and the numeric result is presented to the user along with the array name.

At times, the entered expressions or the program lines will be syntactically incorrect. When such an error is discovered the calculator will take no further action other than to display an error message and give an audible signal. Additional error information can be obtained from the error booklet or by simply recalling the line. If a syntactical error is discovered, the line will be displayed with a flashing rectangular cursor. The cursor identifies the problem area so the error can be corrected quickly.

Editing and Debugging

In the process of composing a program and entering it into the calculator, logical and typographical

errors will inevitably occur. The 9825A offers improved editing capabilities for modifying existing information without retyping complete lines. Also available are many debugging aids, tools the programmer can use to locate and identify programming errors.

The editing features include both character and line editing. The design objective was to minimize the time and number of keystrokes needed to edit the program.

Even though the programmer can minimize the debugging effort by correct problem analysis and good programming techniques, some troubleshooting will usually be necessary. The 9825A facilitates such activity by means of the following commands: step, trace, normal, stop, and reset. The programmer can step through sections of a program line by line. Stopping at certain lines and tracing of lines can be requested before program execution begins or from the live keyboard while the program is running. When tracing is specified, the calculator will print the number of the line that is currently being executed, followed by any variable assignments and flag assignments. In the case of array assignments, the array name and the specific numeric subscripts will be given followed by the value assigned to the array element.

Tape Cartridge Commands

The 9825A has a versatile set of cartridge commands for recording and loading programs, data, special key definitions, and assembly language programs.

When the calculator is turned on, the verify-after-write feature is enabled. Following every record command, the calculator will back up, read the recorded information, and compare it byte-by-byte to the corresponding information in the calculator read/write memory. If an error is detected the calculator will issue a terminal error signal and abort the operation. Two commands, auto-verify enable and disable, allow the user to enable or disable this feature under program control. Another verify command verifies the recorded file and returns the status of the operation. When a bad file is encountered, the information can be recorded on another file and the task continued.

During load operations, files are read up to four times if an error is detected in the check word. Thus a transient error, such as a loose dirt particle, is not visible to the user.

Use of the verify features and proper cartridge handling will assure very infrequent loss of recorded information. To minimize loss of programs and data when failures do occur, every file is divided into partitions 256 bytes long. When a program file partition is lost, the corresponding lines will be substituted by

a line of asterisks, *. When the program is listed, the user can immediately spot the location of the missing lines. Otherwise, the partitions are managed by the system and are completely transparent.

Often the programmer will have to interrupt the debugging process or swap complete jobs. Depending on program complexity and system requirements, these may not be trivial tasks. The 9825A includes a "record memory" command which simplifies these tasks. The command records all the data and programs and internal pointers, so when the file is loaded back into the calculator the system's status is the same as before the record operation. This allows continuation of the debugging process or complete redefinition of the system with a single command.

Optional Language Features

The Advanced Programming ROM includes a cross-reference command and subroutine and function calls. The cross-reference command scans the entire user program in the calculator memory, locates all the simple and array variables used in the program, and prints them in alphabetical order. Every variable is followed by a list of the line numbers in which that variable occurs. This cross-reference listing can be very helpful in editing and appending large programs.

The "go sub" statement provides basic subroutine capability in HPL. The subroutine "call" statement implemented in the Advanced Programming ROM provides additional capability by allowing parameter passing and local variables. Subroutines can return one or more parameters. Functions allow parameter passing and local variables, but return only one parameter. The following example demonstrates the use of subroutines and functions.

```
0: ent A,B
1: c11 'sum,diff'(A,B,C,D)
2: prt C,D
3: prt 3+'sum'(A,B);stp
4: "sum,diff":p1+p2→p3;p1-p2→p4
5: ret
6: "sum":AA→p3;BB→p4
7: ret√(p3+p4)
```

The values of the variables A and B are passed to the subroutine 'sum,diff' and are assigned to the local variables p1 and p2. In turn, the results, p3 and p4, are returned to the variables C and D. In line 3, the function 'sum' returns a numeric value and the result of the evaluated expression is printed. The p-variables are local to the particular subroutines and functions. When these routines are exited, the p-variables are deallocated, thus returning to the program the temporarily used read/write memory. The simple and array variables are global variables. They can be used in subroutines and functions as well as in the main program.

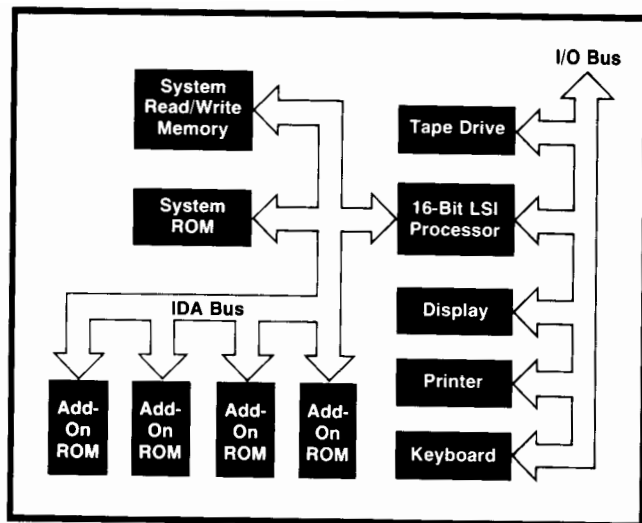


Fig. 2. 9825A hardware organization. The processor and ROMs are HP-developed NMOS LSI circuits.

With the Matrix and Array optional ROM, a 40×40 matrix can be inverted in 80 seconds. Matrix transposition and multiplication are among the matrix operations. Array operations (+, -, *, ÷) are performed on corresponding elements of identical-size arrays.

Hardware Organization and Memory Structure

Fig. 2 is a block diagram of the 9825A hardware. The key elements of each block were developed especially for this machine, except for the 4K RAMs in the read/write memory block.

The processor is described in the article beginning on page 15. Communication between the processor and memory is over the 16-line, bidirectional IDA bus. The ROMs are 16,384-bit HP-developed N-channel MOS devices. To save power and improve reliability, these ROMs operate in a low-power standby mode until addressed.

The total memory address space for read/write and read-only memory is 65,536 bytes. Memory cycle time is 880 nanoseconds. The standard 9825A includes 8K bytes of read/write memory and 24K bytes of operating system in ROM. The user can expand both types of memory if needed.

Input/Output Structure

The internal input/output bus of the 9825A Calculator is the same I/O bus that goes to the three I/O slots. Therefore, the printer, display, keyboard, beeper, and tape transport are all treated as peripherals.

Communication between the processor and the peripherals is done asynchronously with respect to the system clocks. Signals on the data and control lines are valid only when the I/O strobe (IOSB) is low.

The selection of peripherals for data transfer is

done via four signal lines. These lines are defined by loading memory address 9₁₀, which is also known as register R9. Each peripheral must know its own address and decode that address during an I/O strobe.

Registers (memory addresses) R4 through R7 are phantom registers to the processor; they exist only in the electronics associated with a peripheral. To transfer data between the processor and a peripheral device, the peripheral address is written into R9, and then either a "Load RN" or "Store RN" command is executed, where N is 4, 5, 6, or 7. The register numbers are encoded on two lines.

If the command is a store command, then the signal DOUT goes low, denoting a transfer to the peripheral register selected. If the command is a load command, the transfer will be from the peripheral into the processor's A register or R0.

I/O Interrupt Structure

The standard I/O system can operate without the use of interrupt, but the interrupt capability is defined only in conjunction with the standard I/O protocol. The interrupt capability can be turned on or off by software command.

There are two levels of interrupt. On each level there are eight ranked priorities. The interrupt priority of a device is the same as its peripheral address or select codes. Low-level interrupt devices have select codes 0 to 7; select code 0 has the lowest priority. High-level interrupt select codes are 8 to 15; 15 has the highest priority.

If interrupt is enabled, then any interrupt can normally be serviced. If more than one peripheral on a given level wants service, the device with the highest priority is serviced first. Once an interrupt service request has been initiated for a given level, another request from the same level will be ignored until the first request is completed. A high-level interrupt will interrupt a low-level service routine. The termination of an interrupt service request is similar in function to the RETURN at the end of a subroutine.

To minimize interrupt service times, the device polling and software branching are handled entirely by the interrupt hardware. Actual service response times are software-dependent.

There are two interrupt request lines that are pulled low by peripheral devices to indicate that they need service. One is the low-level request line and the other is the high-level request line. As soon as either (or both) of these lines is low and interrupt has been enabled, the I/O chip initiates the polling routine. Any operation on the I/O or IDA buses is allowed to complete.

When an interrupt request is initiated by a peripheral device, it holds the appropriate line low

until it is actually granted service. The fact that an I/O poll occurs does not necessarily mean that interrupt was granted, because more than one device might have been requesting service. If a request occurs on both levels, then the high level will be the first polled and serviced. The processor automatically selects the highest-priority request on each level.

Before the poll (usually during the turn-on routine) an interrupt vector is stored in register R10. This interrupt vector is twelve bits long; concatenated with the select code of the interrupting peripheral it forms the address of the appropriate entry in the interrupt table. The interrupt table contains sixteen entries; these are the starting addresses of the peripheral service routines. The I/O chip automatically branches through the interrupt table to the selected service routine.

The selected peripheral doesn't receive any indication that it has been selected until the service routine resets the interrupt request. This delayed acknowledgment permits interrupting a low-level request by a high-level request without regard to when the interrupt occurs.

The 9825A has a single channel of DMA (direct memory access). One of the high-level select codes is assigned to DMA. The I/O chip handles the protocol for the memory bus cycles and signals the processor after the necessary number of transfers has been completed.

Interfacing External Peripherals

The 9825A Calculator has enhanced interfacing capabilities. It has fast, versatile interface hardware, high-level interface firmware, and extensive facilities for control of the HP Interface Bus (Hewlett-Packard's implementation of IEEE Standard 488-1975). These capabilities help make the new calculator a powerful controller for external peripherals, automatic stimulus-response test systems, and real-time process control systems.

Interfacing capability was a major consideration in the development of the new calculator. Speed, in particular, was important. The 9825A mainframe is roughly eight times faster than its 9821A predecessor. The interface hardware can transfer data at speeds up to 400,000 words per second, and the HP-IB interface performs at up to 45 kilobytes per second. Interrupt and direct memory access (DMA) capabilities have been added.

To interface the 9825A to a peripheral, a hardware interface is plugged into one of the three interface slots on the back of the 9825A, and a ROM containing interface instructions is plugged into one of the I/O slots on the front of the calculator.

Flexible interface hardware includes a 16-bit duplex interface card with interrupt and DMA capabilities, a binary-coded-decimal (BCD) interface

card with interrupt capabilities, an HP-IB interface card with extensive interrupt capabilities that implements most of the interface functions defined in IEEE Standard 488-1975, and powerful HPL interface instructions that may be added by plug-in read-only memories (ROMs). Features of the interface language include instructions for control of the HP-IB, interrupt operations, and input and output data buffers operating in interrupt, DMA, or burst transfer modes. The main consideration in the design of the HPL interface firmware was to make the task of programming external devices as easy as possible. Three I/O ROMs are available. They are the General I/O ROM, the 9862A Plotter ROM, and the Extended I/O ROM.

16-Bit Duplex Interface

The 98032A 16-Bit Duplex Interface is a general-purpose interface that enables the HP 9825A Calculator to exchange input and output information with a wide variety of devices. This interface features 16-bit input and 16-bit output data registers that provide temporary storage during data transfer, plus control, interrupt, and direct memory access (DMA) logic. Also provided is a peripheral control line (take action) to the peripheral device and a peripheral flag line (action complete) from the peripheral device. In addition to these basic control lines, the interface also features two extended control lines for multiplexing applications and three device status lines.

The calculator initiates an input operation by requesting data from the peripheral device via the peripheral control line. The peripheral device indicates the data is ready via the peripheral flag line. This signal enters the data into the input data register and indicates the completion of the I/O operation by setting the calculator flag line and setting one of the interrupt request lines if interrupt has been enabled. The calculator responds with an input instruction that enters the input data into one of the calculator's accumulators.

An output operation is initiated when the calculator transfers the contents of one of its accumulators to the output data register. The calculator also issues a "take action" command via the peripheral control line, indicating that the data is ready to be acted upon. The peripheral device indicates acceptance of the data via the peripheral flag line. This signal also marks the completion of the I/O operation by setting the calculator flag line and setting one of the interrupt request lines if interrupt has been enabled.

The DMA logic on this interface permits the 9825A to exchange input and output information with a peripheral device at high data rates. This logic requests the calculator to perform a DMA transfer whenever the interface has been enabled for DMA and the inter-

face is ready for an input/output operation. Data transfer rates up to 400,000 16-bit words per second can be accommodated with this mode of operation.

BCD Input Interface

The 98033A BCD Input Interface provides the interfacing hardware necessary to interface the 9825A to a variety of instruments that provide binary-coded-decimal (BCD) output data in a bit-parallel, digit-parallel form. This interface permits the input of up to 10 BCD digits with overload and sign information. The interface can also be easily programmed to input BCD digits from two instruments.

This interface contains a multiplexer and code converter that converts the bit-parallel, digit-parallel BCD input data into a string of 16 ASCII characters. It also contains control and interrupt logic.

The calculator initiates a reading from the interfaced device by issuing an output instruction that causes the interface to set the device control lines true. Initiation of a reading causes the interface to indicate "busy" to the calculator. The interface will remain in this mode until a device flag is received indicating that a valid reading is available on the input lines. If interrupt has been enabled, one of the interrupt lines will be set true when a valid reading is available. The calculator responds with input operations until the ASCII character that represents the end of the reading is detected. Unlike most other interfaces, storage is not provided on this interface. Data must be retained on the input lines until the calculator has

input the information.

The input string of 16 ASCII characters has two formats. If only one device is connected to the interface, the format is an eight-digit signed mantissa with a one-digit signed exponent, a one-digit function code, and an overload indication. If two devices are connected to the interface, the format is a four-digit signed mantissa and a five-digit signed mantissa with a positive exponent.

HP-IB Interface

The combination of the 98034A HP-IB Interface and the Extended I/O ROM makes the 9825A Calculator a powerful and versatile controller for the HP Interface Bus. An extensive message oriented instruction set eliminates the need for the user to have a detailed knowledge of the structure of the HP Interface Bus. The table below shows the HP-IB functions implemented in this interface.

IEEE 488-1975 Interface Function	98034A Capabilities
Source Handshake (SH1)	Complete Capability
Acceptor Handshake (AH1)	Complete Capability
Talker (T5)	Basic Talker Serial Poll Talk Only Mode Unaddress if my listen address (MLA)
Listener (L3)	Basic Listener Listen Only Mode

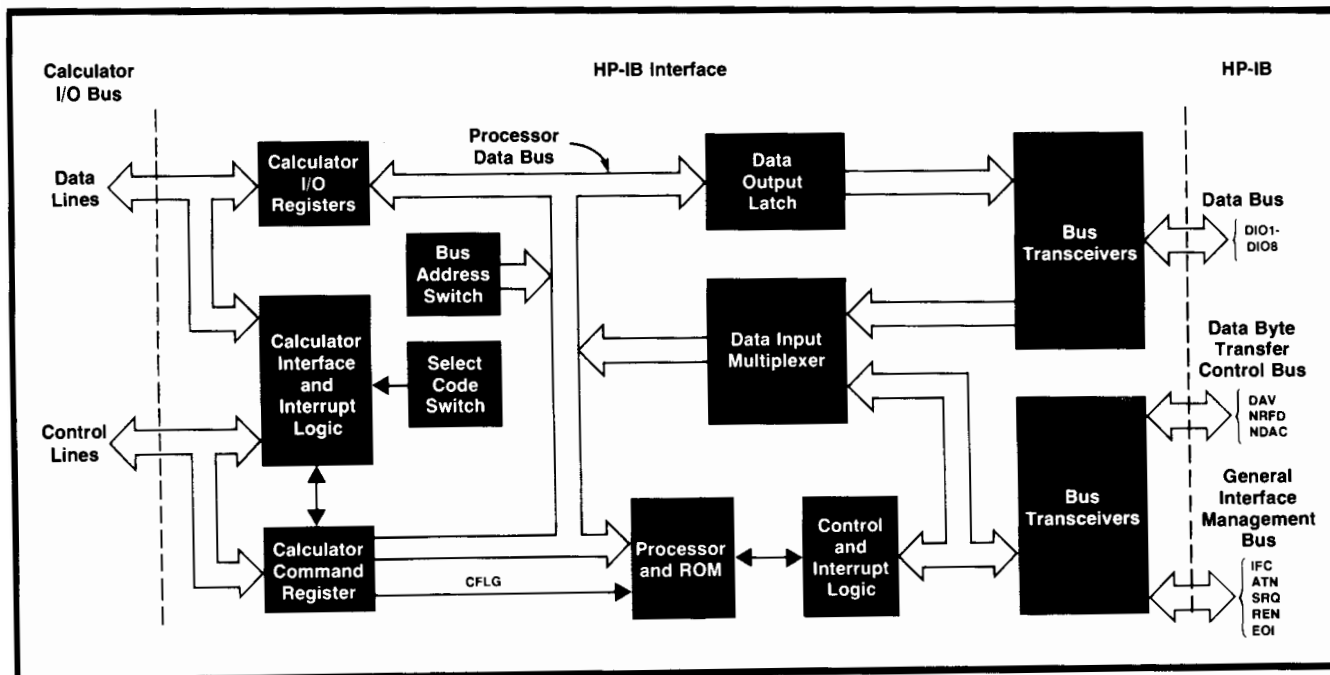


Fig. 3. HP-IB Interface, used with the Extended I/O ROM, implements most of the functions defined in IEEE standard 488-1975. Some HP-IB functions are also provided in the General I/O ROM. Binary and BCD interfaces are also available.

Service Request (SR1)
Parallel Poll (PP2)

Device Clear (DC1)
Controller (C1,2,3,4,5)

Unaddress if my talk
address (MTA)
Complete Capability
Omits capability of
being configured by
controller
Complete Capability
System Controller
Send Interface Clear (IFC)
Send Remote Enable
(REN)
Respond to Service
Request (SRQ)
Send Interface Message
Receive Control
Pass Control
Parallel Poll
Take Control
Synchronously

The HP Interface Bus can also be controlled by the 9825A when it is equipped with the 98034A HP-IB Interface and the General I/O ROM. This combination is sufficient when extensive control of the bus is not required and the primary job to be performed is the transfer of data between the 9825A and peripheral devices on the bus.

The design of the HP-IB interface is based on an eight-bit processor developed by HP. Fig. 3 is a simplified block diagram of the interface, showing the interconnections between the processor, ROM, the I/O registers associated with the 9825A I/O bus and the HP Interface Bus, and the other electronic modules required to support the processor and the 9825A I/O system.

The firmware algorithms contained in the ROM implement the state diagrams for the interface functions and control the flow of messages between the calculator and the HP-IB. Both the calculator I/O bus and the HP-IB are monitored for conditions that require the interface to take action. The calculator can initiate action by addressing the interface and requesting an I/O operation via the command register. When a calculator request for an I/O operation is detected, the processor issues the appropriate instruction to the other modules to decode and execute the requested I/O operation. The status of the HP-IB control bus is periodically sampled via the HP-IB input multiplexer. If a condition is detected that requires action, for example, the service request line (SRQ) is true, the processor issues the appropriate instructions to complete the operation requested.

Interface Firmware: General I/O ROM

The General I/O ROM provides the basic firmware necessary to communicate with external peripherals. Instructions such as formatted read/write and byte or

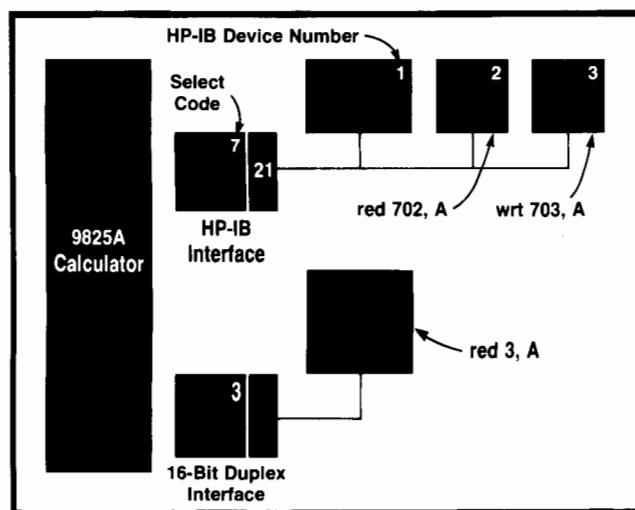


Fig. 4. HP-IB instruction set eliminates the need for the user to have detailed knowledge of the structure of the HP interface bus. Instructions for reading from and writing to devices on the HP-IB are similar in form to other read/write instructions. Bus protocol is handled automatically and is transparent to the user.

word read/write are included in this ROM. A feature of this ROM is the ability to exchange data with the HP-IB interface or any other interface with the same read or write statement.

Fig. 4 shows several devices connected to the 9825A calculator. Three are connected to the HP-IB interface and one is connected to the 16-bit duplex interface. Each of the interfaces has its own select code. The interfaces do not have to be plugged into any particular slot in the rear of the calculator. Also, each device on the HP-IB has its own address.

To read data from the device on the interface with select code 3 and replace the value of the variable A with this data, the program statement is "red 3,A". Similarly, to send variable A to the device with select code 3, the statement is "wrt 3,A".

If the data is to be sent to a device on the HP-IB, then a larger address designates the interface select code and the device number. Reading data from HP-IB device 3 to variable A is accomplished by the statement "red 703,A". The HP-IB device number is the decimal equivalent of the least significant 5 bits of the HP-IB talk/listen address. This device address is appended to the HP-IB interface select code, 7. The general I/O ROM determines that the device is on the HP-IB by the larger select code. It then goes into an HP-IB firmware routine and issues instructions to the HP-IB interface to unlisten all devices, address device 3 as a talker, and address the HP-IB interface as a listener. Data is then transferred from device 3 to the calculator. Thus HP-IB protocol is transparent to the user.

In these examples, it was assumed that the peripherals all had similar data formats. This is not always the case, so a format statement may have to be

used in conjunction with a read or write statement.

Plotter ROM

The 9862A Plotter ROM supplies high-level instructions to the 9825A for controlling the HP 9862A Plotter. Included are instructions for scaling, generating axes, and labeling.

Extended I/O ROM

The Extended I/O ROM is an extension of the interface instructions in the General I/O ROM. These add to the basic interfacing instructions the following capabilities: extensive control of the HP-IB, buffered I/O, interrupt operation, code conversion, and bit manipulations.

When devices are interfaced to the HP-IB, many bus operations are sent between devices. The power of the extended I/O instructions can be illustrated by comparing what HP-IB operations must occur to get a status byte from a bus device, and how this same message may be sent by an extended I/O instruction. Status bytes are received when performing a serial poll on the HP-IB.

Referring to Fig. 4, assume that the 9825A is to read into variable A the status byte of device 3. The bus operations that must occur are:

Unlisten

Calculator Listen Address (Device Number 21)

Serial Poll Enable

Device Talk Address (Device Number 3).

The 9825A can now read the status byte on the HP-IB, store the result in variable A, and then send the command: Serial Poll Disable.

Most HP-IB controllers require that the user initiate each of these bus operations to obtain the status byte. However, with the 9825A and the Extended I/O ROM the user need only execute the statement "rds (703)→A". Thus with one instruction the user can perform this very basic operation of receiving a status byte.

The device statement of the Extended I/O ROM enables one to assign mnemonics to select codes. The

device statement is usually executed at the beginning of a program and equates device mnemonics to select codes. At a later point in the program, the device may be referred to by name, rather than by select code. In the following program, "DVM" is substituted for select code 702, and "Printer" is substituted for select code 703:

```
5: dev "DVM", 702, "Printer", 703
```

```
20: red "DVM", A
```

```
20: wrt "printer", A
```

Interrupt

The Extended I/O ROM adds the necessary instructions for control of the two levels of hardware interrupt available in the 9825A. Fig. 5 shows how interrupt subroutines are implemented. The "On-Interrupt" statement assigns the address the program will jump to when a given peripheral interrupts. The "Enable Interrupt" statement enables a particular interface to interrupt. Each interrupt subroutine must be terminated by the "Interrupt Return" statement. In Fig. 5, lines 5 and 6 assign the location of interrupt subroutines for peripheral devices 2 and 8. Line 7 turns on interrupt for those two devices. Then, during the execution of line 25 in the main program, device 2 interrupts. After line 25 is executed, the program jumps to the interrupt subroutine starting at label A. While line 507 of this subroutine is being executed, higher-level device 8 interrupts, and after line 507 is executed, the program jumps to the interrupt routine beginning at label B. At the completion of this interrupt program (line 612), the program returns to line 508. After completion of this low-level interrupt routine (line 510), the program returns to line 26 of the main program and continues.

The "On-Interrupt" statement can be programmed to cause an immediate jump to a service routine on interrupt, before the current line is completed. However, the information processed in that line will

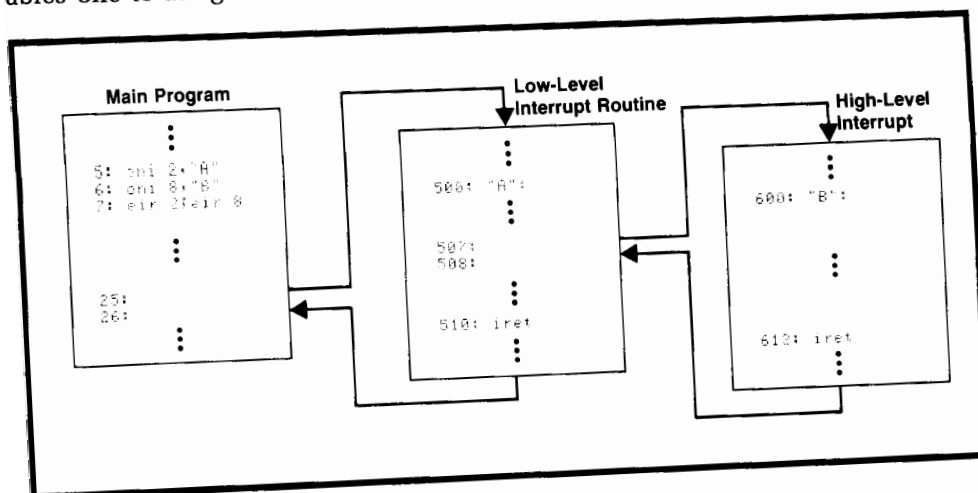


Fig. 5. The extended I/O ROM adds high-level instructions for control of the 9825A's two levels of interrupt. This example shows how control is transferred during the servicing of low-level and high-level interrupts.

be lost and the interrupt program cannot return to the line in which the interrupt occurred. This mode of operation could be used, for example, if an emergency demanding immediate attention should occur.

Buffered I/O

For most I/O transfers, the read and write instructions in the General I/O ROM are sufficient. However, in some instances the calculator may need to communicate with a device that requires data at a higher rate. Another problem situation occurs when the calculator is executing a program and needs to communicate with a very slow device. If the write or read statement were used here, the calculator would spend most of its time waiting for the slow device. Assuming that there were other tasks which the calculator

had to perform during this wait period, it would be advantageous if data from the calculator could be placed in a buffer to be sent to the peripheral whenever it interrupted to indicate that it was ready for the next word or byte of data.

The Extended I/O ROM has buffered input and output capability that helps solve these problems. Shown in Fig. 6 is an output buffer in the 9825A. This is a software buffer located in the calculator memory. The buffer can be filled by write statements that transfer data from data registers to the data buffer. The data is then transferred from the data buffer to the peripheral using the transfer statement in one of three modes: burst, direct memory access (DMA), or interrupt.

In the burst mode the program stops while data is

9825A Cartridge Tape Unit

The cartridge tape unit in the 9825A Calculator has a careful mixture of mechanical, electronic, and software features that combine to give the user high performance and at the same time improve reliability and ease of use. Its operating characteristics are:

Capacity: 250,000 bytes
Average Access Time: 6 seconds
Read/Write Speed: 559 mm/s (22 in/s)
Search Speed (bidirectional): 2286 mm/s (90 in/s)
Typical Transfer Rate: 2750 bytes/s
Typical Access Rate: 14300 bytes/s
Rewind Time: 19 s (one entire track)
Verification: Automatic on recording

The tape cartridge is the same mini data cartridge that is used in the 2644A Terminal¹ and the 9815A Calculator. It measures only 63.5 mm × 82.5 mm × 12.7 mm (2.5 in × 3.25 in × 0.5 in).

The tape is encoded with a delta distance technique. A short distance (time) between magnetic flux transitions on the tape represents a data zero, and a long distance represents a one. During each interval between transitions a capacitor is charged by a constant current, so at the end of the interval the capacitor voltage is proportional to the time between transitions. This voltage is compared with a threshold voltage to determine whether the data bit is a zero or a one. The threshold voltage is also stored on a capacitor and is updated slightly after any transition interval that is shorter or longer than expected. Thus the threshold tracks low-frequency variations in the transition intervals. Tracking makes it possible for the ratio of a one interval to a zero interval to be smaller than it would have to be otherwise, thereby increasing cartridge capacity.

At nominal read/write tape speeds the following transition intervals are typical.

0's	= 28.4 μ s
1's	= 49.7 μ s
threshold	= 38.3 μ s
partition gaps	= 550 μ s
file gaps	= 45 ms
end of valid data gap	= 300 ms

The partition gap is used within files so that if an error occurs, only a small portion of data will be lost instead of the entire file. Partitions are otherwise invisible to the calculator user.

Tape System Design

Tape speed in the 9825A tape transport is sensed by a 1000-line optical tachometer. This high-resolution tachometer makes possible precise control of acceleration and deceleration, helping to minimize file gap length and maximize capacity.

Pulses from the tachometer trigger a monostable multivibrator (one-shot). Low-pass filtering the one-shot output produces a voltage proportional to speed. This signal is compared with a forcing function that is proportional to the desired speed, and the difference signal drives the motor by way of an amplifier that compensates for the mechanical pole of the motor. The overall servo system bandwidth is 200 Hz.

Speed changes occur at a constant acceleration or deceleration produced by changing the forcing function linearly with time.

The output of the magnetic head is proportional to the time derivative of magnetic flux, so the flux reversals on the tape are coincident with peaks in the head output waveform. These peaks are detected by differentiating the signal and sensing zero crossings. The head signal is also level detected, and only peaks that exceed a threshold are passed on. This eliminates problems with noise during gap regions on the tape.

The threshold circuit performs an important function during automatic verification after recording. During normal read operations the threshold is set at 10% of the nominal output pulse amplitude. After a record operation the new information is read and compared with memory with the threshold set at 45% of nominal amplitude. This assures not only data validity, but also that the recording is not marginal in amplitude or pulse quality.

The output of the read electronics is fed to the decoder, which reconstructs binary data from the times between flux transitions, as described above.

The write system of the 9825A Calculator is the same as that of the 9815A (see article, page 24).

Reference

1. R.G. Nordman, R.L. Smith, and L.A. Witkin, "New CRT Terminal Has Magnetic Tape Storage for Expanded Capability," Hewlett-Packard Journal, May 1976.

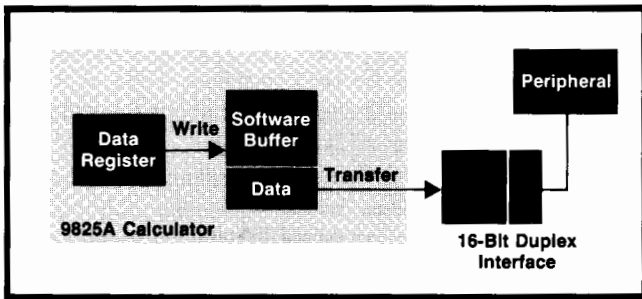



Fig. 6. Buffered I/O provides software buffers for input and output data transfer in DMA, interrupt, or burst modes.

transferred to the peripheral in the fastest possible software loop. Data rates of up to 45 kilobytes per second may be achieved on the HP-IB.

In the DMA mode, upon execution of the transfer statement, the program continues while data is transferred from the data buffer to the peripheral under DMA. Data transfer rates of up to 400,000 words per second may be achieved. DMA operation is available only with the 16-bit duplex interface.

In the interrupt mode, upon execution of the transfer statement, the program continues while data is transferred from the data buffer to the peripheral under interrupt operation. The calculator fills the data buffer and instructs it to begin the transfer to the slow device. The data transfer between buffer and peripheral is then transparent to the main program. 

Donald E. Morris



Serving as overall project manager, Don Morris launched the 9825A project, then continued in that capacity for the hardware design when the software became a separate project. With HP since 1970, he had previously worked on the 9805A and 46A Calculators. Born in Dallas, Texas, Don attended the University of New Mexico, graduating in 1967 with a BSEE degree. From Oklahoma State University, he received MSEE and PhD degrees in 1968 and 1970. He's married, has two

sons, and lives in Loveland, Colorado. Don owns and flies his own airplane and devotes a significant portion of his free time to teaching mathematics and programming at a local vocational education center.

Dick B. Barney



With HP since 1969, Dick Barney has served as development engineer for 9800-Series Calculators and the 9866A Printer, as product manager for the 9800 Series interface cards, and as project leader for the 9825A's interface cards. Born in Helena, Montana, he served in the U.S. Navy for three years, then enrolled at Montana State University and received his BSEE and MSEE degrees in 1965 and 1969. He's a member of IEEE. Now living in Loveland, Colorado, Dick is married,

has three children, and is an active member of Toastmasters International. He also enjoys reading, fishing and backpacking in the mountains of Colorado and Montana.

Geoffrey W. Chance



Geoff Chance was project manager for the 9825A I/O System. Geoff joined HP in 1965. He was the project manager for the memory and power supply of the 9810A Calculator, and served as production engineering manager after the 9810A was transferred to production. He holds BSEE and MSEE degrees from Montana State University, received in 1963 and 1965, respectively. Geoff is a native of Montana. He's married, the father of two small girls, and an active Toastmaster. He enjoys backpacking, woodcarving, and handball.

Chris J. Christopher



Chris Christopher was project manager for the 9825A's language, HPL. A native of Greece, he came to the U.S.A. in 1961 and received his BSEE and MSEE degrees from Colorado State University in 1968 and 1974. Joining HP in 1968, he worked on computerized testing and production of 9100 Series Calculators and peripherals, and developed the 9830A tape cassette software and the software for the 9880A/B Mass Memory. He's a member of IEEE. Now living in Loveland, Colorado,

Chris is married and has one child—a daughter. He enjoys skiing, and invests much of his free time in his garden and his bee colonies.

High-Performance NMOS LSI Processor

William D. Eads and David S. Maitland



THE HEART OF THE 9825A Calculator, and the basis of its performance, is an HP-developed 16-bit processor.

The processor consists of three N-channel metal-oxide-semiconductor (NMOS) large-scale integrated (LSI) circuits. The usual computer instruction set is provided by the BPC chip, the IOC chip provides input/output functions, and the EMC chip provides a set of binary-coded decimal arithmetic macroinstructions. The three NMOS LSI chips are packaged in a hybrid circuit with four bipolar bidirectional buffer chips. Fig. 1 is a photograph of the hybrid circuit.

A block diagram of the processor is shown in Fig. 2. The IDA bus is 16 bits wide and bidirectional. It time multiplexes instructions, data, and addresses. The IDA bus signals go through the bidirectional buffers to either the input/output (I/O) port or the memory port. The control bus contains all the control signals for memory, I/O, and communication between the processor chips.

Communication between the processor chips is the same as interacting with memory. Thirty-two addresses in memory are assigned to the various registers on the processor chips. Whenever a memory cycle is executed that specifies one of these 32 addresses, the chip registers are accessed instead of memory.

Each of the three NMOS chips has its own internal controller, which operates on 16 bits of parallel data or 15 bits of address. Each chip also has an instruction register and a decoder for recognizing instructions.

Table I shows typical processor instruction execution and I/O times.

Binary Processor Chip

The primary chip of the system is the binary processor chip (BPC). It executes 59 instructions and can function independently of the other chips.

The BPC is built around two general-purpose 16-bit accumulator registers, A and B, as well as other registers. The memory address space of the processor is 32,768 words, which requires a 15-bit address field. The 16th bit of an address word is used to define indirect addressing. If the most significant bit of an ad-

dress is clear, the word at that address is data; if the most significant bit of the address is set, the word at that address is treated as another address.

The memory reference group of instructions forms a memory address by adding the least significant 10 bits of the instruction to the current value of the 15-bit program counter register (P) giving an address relative to P. Or, depending upon a bit in the memory reference instruction, this bit can be used to address a fixed 1024-word block of memory, called base page. Thus any memory reference instruction can directly

Table I

Typical Instruction Execution Times

Instruction	Description	Time (μ s)
LDA m	Load accumulator A from address m	1.83
CPB m	Compare contents of B with contents of m; skip next instruction if unequal.	2.33
JMP m	Unconditional branch to address m	1.17
JSM m	Jump to subroutine at m. The contents of return stack register R are incremented, and P is stored at the address contained in R.	2.5
RET n	Return from subroutine. The contents of the address in R are placed in P and added to n ($-32 \leq n < 32$). The processor jumps to this address and R is decremented by 1.	2.33
ABR n	Arithmetic right shift of B n places, with sign bit filling all vacated positions.	2.67 ($n=8$)
PBC m	Place byte content of m into stack addressed by C. Increment C.	3.5
FXA	12-digit decimal add	4.67
FMP	12-digit-by-1-digit decimal multiply giving 13 digit result.	17.3

Typical I/O Times

LI	Lockout time (maximum) of interrupt acknowledgment	Instruction time + .33 μ s
LD	Lockout time (maximum) of DMA	1.67 μ s
	Maximum transfer rate in DMA	667,000 16-bit words/s

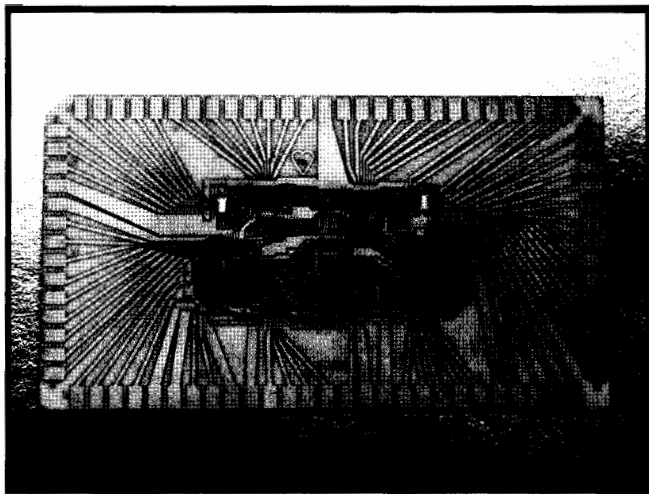


Fig. 1. An HP-developed 16-bit processor is the heart of the 9825A Calculator. It consists of three NMOS LSI chips and four bipolar chips in a hybrid circuit on a 7.6×3.8-cm ceramic substrate.

access the 1,024 words of base page memory and the words that lie within ± 512 words of itself. If the instruction specifies an indirect address, any word in memory can be accessed via the address word. The memory reference instructions include loading, storing, adding, comparing A or B with memory, jump, jump subroutine, increment or decre-

ment memory and skip on zero, and logical AND/OR operations with A.

The shift-rotate instructions contain a four-bit field that allows 1 to 16 bits of shift or rotation on the A or B register. Left and right shifts are allowed. The alter-skip instructions contain a six-bit field to allow skipping up to 32 instructions forward or backward. These instructions may be used to test for various conditions of the BPC registers.

To minimize the time it takes to form the address for the next instruction when a skip is executed, a dedicated 15-bit full adder is associated with the program counter. This adder also decreases instruction time by forming the next address while executing the current instruction.

A stack pointer for subroutine returns is also on the BPC. Two other single-bit registers that can be set, cleared, and interrogated act as software flags. These flags are also set if an arithmetic overflow occurs. There are also four external flag inputs that can be monitored with software instructions. BPC functional units are shown in Fig. 3.

Input/Output Chip

The input/output chip (IOC) acts as the controller for all communication via the I/O port and executes 12 instructions. One of the functions it provides is con-

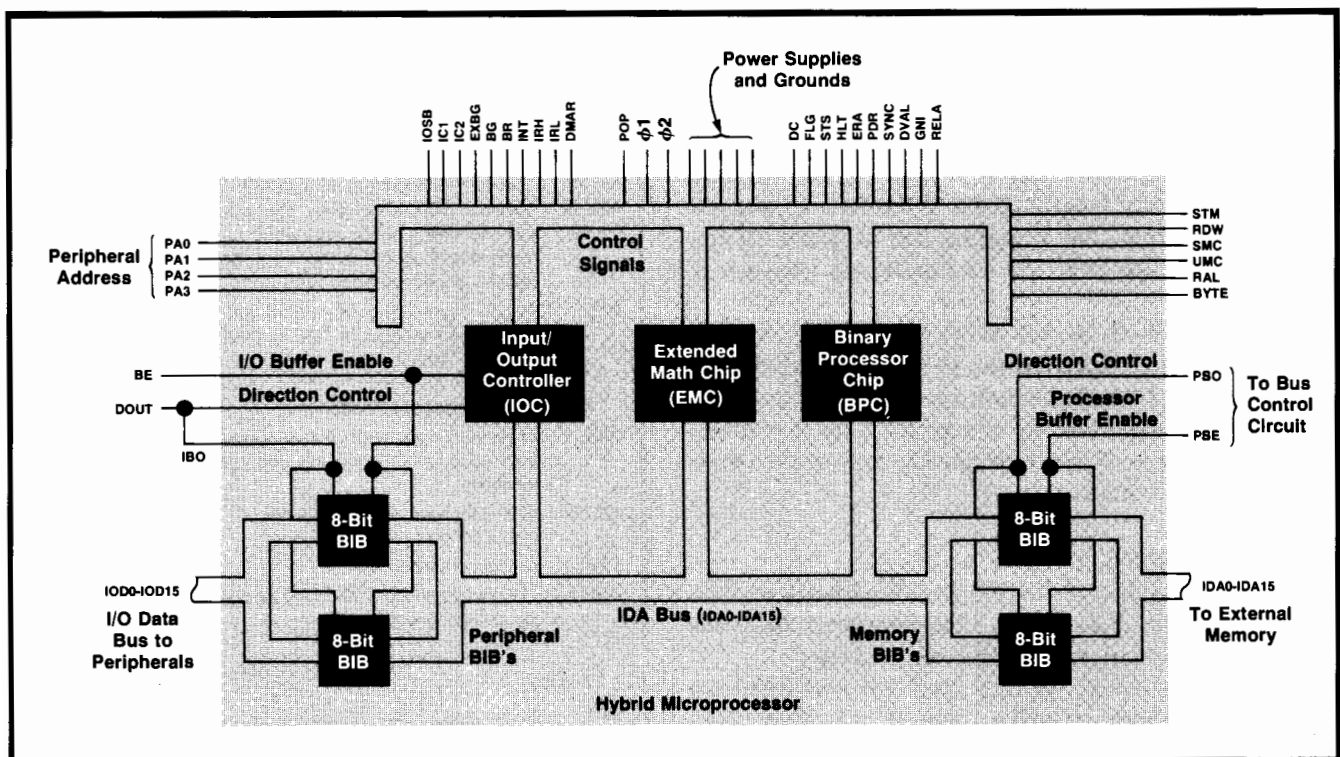


Fig. 2. Processor block diagram. The primary chip is the BPC, which executes 59 instructions. The 12-instruction IOC controls input and output functions. The EMC executes 15 instructions, mostly for mathematical routines.

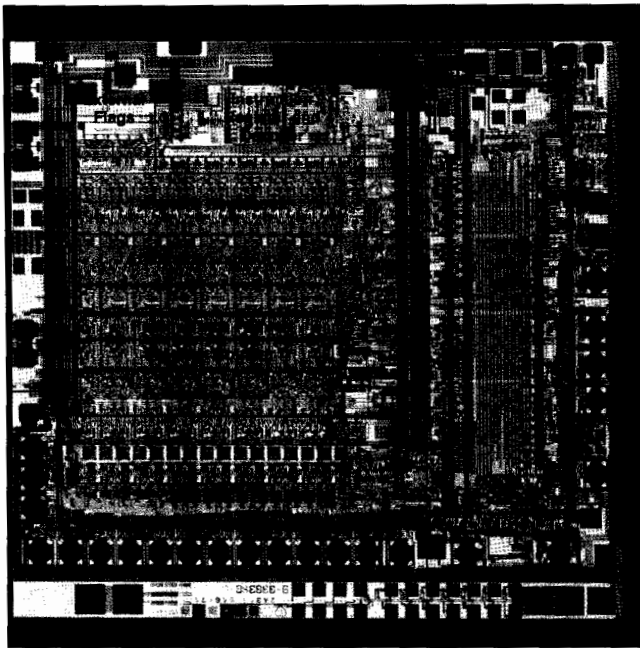


Fig. 3. BPC chip layout.

trol of two signal lines on the control bus by means of software instructions. If any of four particular register addresses is specified in a memory reference, these two lines go to a particular coded binary state depending upon which register was addressed. The two signals remain in the same state until the next instruction. Another I/O control function centers around a four-bit peripheral address register that can be accessed with memory references. All four bits are output continuously as select codes on the control bus. This saves external latching and decoding circuitry in the peripheral interfaces. The select code remains in the peripheral address register until it is altered by a memory write cycle to the peripheral address register or by an interrupt.

All the hardware for two levels of vectored interrupt is on the IOC. It determines if an interrupt is valid, and does a parallel poll to determine which peripheral has requested the interrupt. It concatenates the peripheral address with a 12-bit interrupt vector register to form the indirect address of the routine. The select code of the interrupting peripheral is put into the peripheral address register. Because it is usually necessary to save the select code that was in the peripheral address register before interrupt occurred, the contents of this register are placed on a stack whenever interrupt is granted. The original select code is returned to the peripheral address register at the end of the interrupt service routine.

Another I/O function is direct memory access (DMA). If DMA is requested, the IOC inserts memory cycles within the current instruction. Two 16-bit counter registers keep track of the number of DMA

transfers and the sequencing of the memory address. A four-bit DMA peripheral address register holds the select code of the DMA operation.

In addition to interrupt and DMA operations, the IOC can place or withdraw words or bytes sequentially into or out of two areas of memory. The stack group of instructions provides for addressing into two data stacks using two hardware registers, C and D, as pointers. The data stacks may be accessed either by word or by byte, depending upon the instruction, and may grow as memory addresses increase or decrease, again depending upon the instruction.

Extended Math Chip

The extended math chip (EMC) executes 15 instructions, mostly for implementing math routines. Each of these instructions actually causes several machine level instructions to be executed. Having these special macroinstructions in hardware makes the math routines over ten times faster than they would be in software.

There are two types of macroinstructions, differing in the numerical format on which they operate. The first type is designed to be used with any type of binary data. Two macroinstructions in this group clear or transfer one to sixteen contiguous words of memory. A third microinstruction can multiply two 16-bit two's complement numbers and obtain a 32-bit product.

The second type of macroinstruction is designed to be used with a three-word (48-bit) binary-coded-decimal (BCD) 12-digit mantissa. The EMC has a four-digit arithmetic unit capable of performing decimal addition and optionally generating the 9's complements necessary for subtraction operations. The controller in the EMC interacts heavily with memory, but is also important for on-chip, multi-step microinstructions. In processing floating point, 12-decimal-digit mantissas, the controller directs the flow of data from a decimal accumulator register and memory to the arithmetic unit. When adding four-digit words, the generated carry goes to the decimal carry register where it is added to the next most significant digit during the subsequent add. The sum is placed in the 12-decimal-digit accumulator register.

Bidirectional Interface Buffers

The four other integrated circuit chips shown in the processor block diagram (Fig. 2) are bidirectional interface buffers (BIBs). These 8-bit bipolar buffers perform two main functions. On the MOS side, they provide a four-volt logic high level and a high-impedance load. On the other side they have the low impedance necessary to drive the capacitance of the instrument's internal bus. The BIBs have two control inputs. One determines the direction of transmission, and the other disables all transmission.


Processor Details

The seven chips shown in Fig. 1 are on a ceramic substrate 7.6 cm long and 3.8 cm wide. Thin-film single-layer gold interconnect traces are used. The chip signals are directed to these gold traces through 234 aluminum wires only 38.1 μm in diameter. The traces come off the hybrid via 82 legless electrical connections. The aluminum die cast heat sink keeps the surface of the LSI chips at less than 70°C in a 55°C ambient while dissipating six watts in the processor.

Dimensions of the LSI chips average approximately 4.7 mm per side. The output drivers on the chips are connected to the +5V supply to keep the output level clamped below 5 volts. The output circuits on the processor chips are capable of switching a 50 pF load between 0.5V and 4.5V levels in less than 10 ns. Typical gate delays within the chips vary from 2 to 20 ns, depending upon speed requirements.

Each processor chip has an internal control array approximately equivalent to an 8000-bit ROM. Also, on the average, about 6000 MOS transistors, or 2000 equivalent gates, are contained on each LSI chip.

The processor operates with square wave, two-phase non-overlapping clocks at speeds from 1 kHz to greater than 10 MHz. These clocks have a minimum high level of 6.5 volts and a maximum low level of 2 volts. Signal inputs have high levels above 3 volts and low levels below 2.2 volts. Input pads are protected from static discharges of greater than 3 kilovolts.

The processor uses four power supplies: +12, +7, +5, and -5 volts. Many types of circuits are used in the processor chips, such as ratio and ratioless logic elements, dynamic storage devices, bootstrap circuits, and static and dynamic array structures. 



David S. Maitland

Dave Maitland has been designing calculator circuits for HP since 1967, starting with the 9100A, HP's first calculator. He designed the 4K ROM for the 9800 Series, worked on the 9805A, and was project manager for the 9825A processor. He's authored half a dozen patent applications. An Illinois product born in Belvedere, Dave received a B.A. degree in liberal arts from Rockford College (in Rockford) in 1963, and a B.S. degree in electrical engineering from the University of

Illinois (in Champaign) in 1964. Married six years, he and his wife have a son, 2, and live in Loveland, Colorado. Dave's interests are many and varied; they include fishing, hiking, his farm and motorcycle, skiing, travel, music and books, photography, and the Denver Broncos football team.

Processor Tester

An Application of the 9825A Calculator

To test the NMOS LSI processor hybrid, a 9825A Calculator is used as an environment for the processor under test and a second 9825A is used as an analytical controller. The purpose of the test is to determine if the device is functional, and if not, which chip has failed.

Executing go/no-go software and taking dc voltage readings are the two parts of the test. The go/no-go software exercises 99% of the transistors and is executed separately for each LSI chip. If and only if any of the three LSI chips fails, then the dc test is executed. This dc test determines if the bidirectional interface buffers are at fault. These two tests require approximately two minutes, 30 seconds of which is the go/no-go software test.

By tripling the load capacitance and increasing the clock rate by 15%, a worst-case test is performed. A much wider range of supply voltages is tested than is required by the 9825A design. Using 35 combinations of voltages and checking each point twice can mean that the go/no-go software is executed 170 times for each of the LSI chips. The output from the tester for a good device is a voltage plot of the operating range for each LSI chip. All output information is printed on the 9825A's built-in printer.

Other test equipment used in these tests includes two scanners (3495A), two relay actuators (59306A), a multimeter (3490A), a counter (5300 System), an ASCII digital clock (59309A), and five resistance programmable power supplies controlled by a multiprogrammer (6940B). All this equipment is on the HP Interface Bus. The scanners are used to multiplex either the multimeter or a power supply to any one of 82 pins of the processor for the dc test. The other power supplies become the four processor supplies. Two of these are switched separately through a relay actuator to each LSI chip. This allows the outputs of any LSI chip to be put in a high-impedance state, thus eliminating the possibility of one LSI chip influencing the other's test.

For thoroughness, each device is tested a minimum of three times during assembly. The first test is done after bonding the processor and indicates if repair is necessary. If the device is good, it is encapsulated and tested again. The last test is done after high-temperature reverse-bias burn-in, and failures are indicated by a change in the operating range. Storing the data from the previous test on the internal cassette allows the analytical controller to look for a change. This data also helps determine what chip or bond has failed.



William D. Eads

Texan Bill Eads was born in Dumas and attended Rice University in Houston, graduating in 1966 with a B.A. degree in electrical engineering. He continued at Rice for his PhD degree, received it in 1970, and joined HP that year. Bill has been concerned with computer-aided artwork and computer-aided design techniques, and was project manager for two of the NMOS chips in the 9825A processor. He likes to relax by hiking, camping, biking, or working in his garden. He's

married, has two children, and lives in Loveland, Colorado.