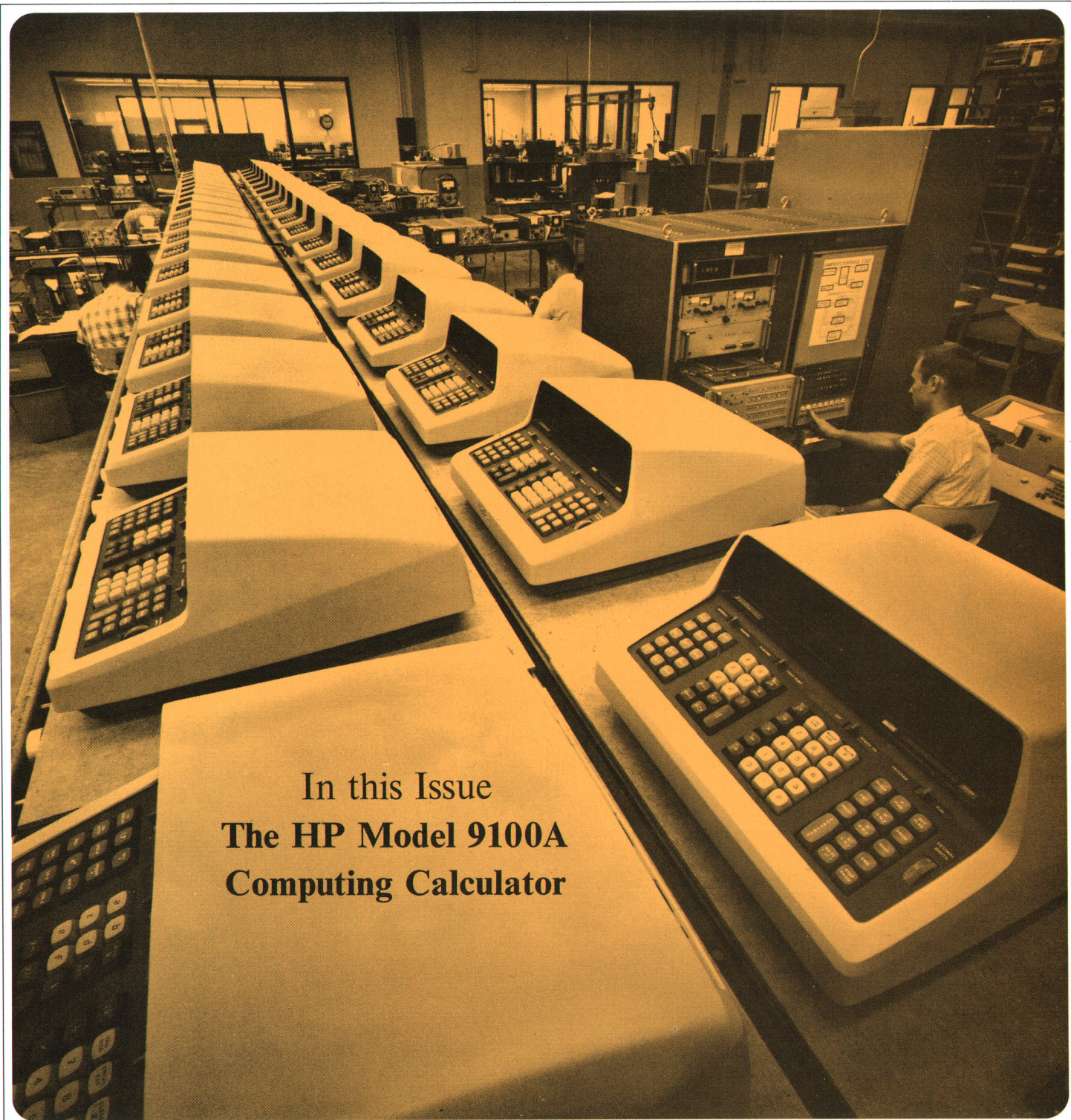


HEWLETT-PACKARD JOURNAL



In this Issue
The HP Model 9100A
Computing Calculator

SEPTEMBER 1968

A New Electronic Calculator with Computerlike Capabilities

By Richard E. Monnier

MANY OF THE DAY-TO-DAY COMPUTING PROBLEMS faced by scientists and engineers require complex calculations but involve only a moderate amount of data. Therefore, a machine that is more than a calculator in capability but less than a computer in cost has a great deal to offer. At the same time it must be easy to operate and program so that a minimum amount of effort is required in the solution of typical problems. Reasonable speed is necessary so that the response to individual operations seems nearly instantaneous.

The HP Model 9100A Calculator, Fig. 1, was developed to fill this gap between desk calculators and computers. Easy interaction between the machine and user was one of the most important design considerations during its development and was the prime guide in making many design decisions.

CRT Display

One of the first and most basic problems to be resolved concerned the type of output to be used. Most people want a printed record, but printers are generally slow and noisy. Whatever method is used, if only one register is displayed, it is difficult to follow what is happening during a sequence of calculations where numbers are moved from one register to another. It was therefore decided that a cathode-ray-tube displaying the contents

of three registers would provide the greatest flexibility and would allow the user to follow problem solutions easily. The ideal situation is to have both a CRT showing more than one register, and a printer which can be attached as an accessory.

Fig. 2 is a typical display showing three numbers. The X register displays numbers as they are entered from the keyboard one digit at a time and is called the keyboard register. The Y register is called the accumulator since the results of arithmetic operations on two numbers, one in X and one in Y, appear in the Y register. The Z register is a particularly convenient register to use for temporary storage.

Numbers

One of the most important features of the Model 9100A is the tremendous range of numbers it can handle without special attention by the operator. It is not necessary to worry about where to place the decimal point to obtain the desired accuracy or to avoid register overflow. This flexibility is obtained because all numbers are stored in 'floating point' and all operations performed using 'floating point arithmetic.' A floating point number is expressed with the decimal point following the first digit and an exponent representing the number of places the decimal point should be moved—to the right if the



Fig. 1. (left) This new HP Model 9100A Calculator is self-contained and is capable of performing functions previously possible only with larger computers. Fig. 2. (right) Display in fixed point with the decimal wheel set at 5. The Y register has reverted to floating point because the number is too large to be properly displayed unless the digits called for by the DECIMAL DIGITS setting is reduced.



TURN ON
SET: ☐ RUN ☐ ON
If flashing or no display after 30 sec.
PRESS: STOP CLEAR
Red light indicates illegal operation such as $\sin^{-1}2$, $\sqrt{-x}$, or division by 0.

DISPLAY
ENTER A NUMBER: 0 0 0 1 4 5 or 1 4 5 ENTER EXP CHG SIGN 3
Keyboard entries displayed in X. Any operation terminates entry. FLOAT FIXED .0015
145 -03
Up to ten significant digits. Power of ten multiplier.
Decimal wheel sets number of places to right of decimal (here 4). Rightmost digit rounded. Registers with left over-flow revert to floating display.

ENTRY KEYS
CLEAR X: Clears keyboard entry. Removes ARC and HYPER conditions. Not needed for new entry.
CLEAR: Clears display and accumulate registers (0 \rightarrow X, Y, Z, E, F). Removes ARC and HYPER conditions and clears the flag.
.: Enters decimal point. Not needed for entry of integers.
ENTER EXP: Causes next digits and change sign entries to affect the exponent of X. Overrides decimal point.
CHG SIGN: Changes sign of number in X. Changes sign of exponent if ENTER EXP was pressed.

CONTROL AND ARITHMETIC KEYS
x, y, and z refer to contents of X, Y, and Z.
ROLL \downarrow : $y \div x \rightarrow Y$
ROLL \uparrow : $y \times x \rightarrow Y$
X \rightarrow Y: $y - x \rightarrow Y$
Y \rightarrow X: $y + x \rightarrow Y$
entry to Y: $y \div x \rightarrow Y$
entry to X: $y + x \rightarrow Y$

FUNCTION KEYS
 \sqrt{x} : $\ln x$ ($\log_e x$) e^x
 $\sin x$ $\cos x$ $\tan x$ $\log x$ ($\log_{10} x$)
Replace x with the indicated function of x.
DEGREES/RADIANS: Sets entry and answer units for trig functions.
arc \vee : Prefix for inverse trig or hyper functions. Eliminates decimal part of x.
hyper \vee : Prefix for hyperbolic functions. Sets y positive.

STORAGE AND RECALL KEYS
STORAGE: Storage registers are 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, a, b, c, d, e and f.
X \rightarrow (): Transfer x or y to register indicated by next keystroke. x or y unchanged.
y \rightarrow (): Transfer y or x to register indicated by next keystroke. x or y unchanged.
RECALL: ALPHA REGISTERS ONLY: a, b, c, d, e, and f recall contents of alpha registers to X. Source register is unchanged.
ALPHA OR NUMERIC REGISTERS: Exchanges y with register indicated by next keystroke; only instruction for recalling contents of a numeric register.
y \leftrightarrow (): Exchanges y with register indicated by next keystroke; only instruction for recalling contents of a numeric register.

VECTOR KEYS
RECTANGULAR-POLAR COORDINATE CONVERSION
TO POLAR: with y in Y ($\theta = \tan^{-1}y/x \rightarrow Y$) and x in X ($R = \sqrt{x^2 + y^2} \rightarrow X$)
TO RECT: with θ in Y ($y = R \sin \theta \rightarrow Y$) and R in X ($x = R \cos \theta \rightarrow X$)
ACC \rightarrow : $E + y \rightarrow E$ ACCUMULATE + x and y unchanged.
ACC \leftarrow : $E - y \rightarrow E$ ACCUMULATE - x and y unchanged.
RCL \rightarrow : $E \rightarrow Y$ RECALL E and f unchanged.
RCL \leftarrow : $E \rightarrow X$

EXAMPLES
(8-3) \times 6 / 3 = 10 Answer \rightarrow Y
8 \downarrow 3 \downarrow - 6 \downarrow \times 3 \downarrow \div
(8) \div 2 = 4 Answer \rightarrow X
8 \downarrow \div 2 \downarrow \times 3 \downarrow \div \downarrow e^x
 $\tan^{-1}(-3) = -71.565^\circ$ Answer \rightarrow X
SET: DEGREES CHG SIGN 3 \downarrow $\tan x$
(3+4)(2+2) = -2+14 Imaginary \rightarrow Y Real \rightarrow X
CLEAR 4 \downarrow 3 \downarrow TO POLAR \downarrow IN X ACC \downarrow 2
TO POLAR IN X ACC \downarrow RCL \downarrow e^x TO RECT

PROGRAM OPERATION
TO ENTER A PROGRAM
SET: ☐ RUN PRESS: GO TO (11) Starting Address
From keyboard: SET: ☐ PROG depress keys in program sequence.
From magnetic card: Insert card, PRESS: ENTER
TO RECORD A PROGRAM
SET: ☐ RUN PRESS: GO TO (11) Starting Address
Insert magnetic card, PRESS: RECORD
TO RUN A PROGRAM
SET: ☐ RUN PRESS: GO TO (11) Starting Address
Enter data and press CONTINUE as required.
TO CHANGE A PROGRAM STEP
SET: ☐ RUN PRESS: GO TO (11) Step to be changed
SET: ☐ PROG PRESS: DESIRED key

PROGRAM CONTROL KEYS
GO TO (11): Causes unconditional branch to address given by next two program steps or keyboard entries.
IF $x < y$: CONDITION MET: Branches to address given by next two program steps if first step is alphabetic. (GO TO not necessary.) Otherwise, executes instructions in next two steps and continues with third step.
IF $x = y$: CONDITION NOT MET: Skips next two program steps. Continues with third program step. (IF FLAG clears the flag.)
IF $x > y$: SET FLAG: Sets a condition to be tested by the next IF FLAG. May be used manually or as a program step.
PAUSE: Forces a brief display during program execution. When held down, causes STOP at next prog PAUSE.
STOP: Stops program execution when used manually or as a program step.
END: Ends recording on magnetic card. Gives STOP and automatic GO TO (0) (0). Must be last program step.
CONTINUE: Starts program execution at present address. May be used as a "No Operation" step.
STEP PRGM: In ☐ PROG mode: Displays address and instruction code in X. Address \rightarrow 20-----43 \leftarrow Code In ☐ RUN mode: Executes one program step or all 3 steps of GO TO (). A met "IF" step branches to address in next two steps. If not an address, executes first step only. A not met "IF" step branches to and executes third step.

CODE	KEY	CODE	KEY	CODE	KEY	CODE	KEY	CODE	KEY	CODE	KEY	CODE	KEY	CODE	KEY	CODE	KEY
00	0	10	8	20	CLEAR	30	$x \leftrightarrow y$	40	$y \rightarrow (1)$	50	IF $x = y$	60	ACC \rightarrow	70	$\sin x$		
01	1	11	9	21	\cdot	31	ROLL \downarrow	41	STOP	51	IF $x < y$	61	RCL	71	$\tan x$		
02	2	12	π	22	ROLL \uparrow	32	CHG SIGN	42	FMT	52	IF $x > y$	62	TO POLAR	72	arc \vee		
03	3	13	π	23	$x \rightarrow (1)$	33	\div	43	IF FLAG	53	SET FLAG	63	ACC \leftarrow	73	$\cos x$		
04	4	14	b	24	$y \rightarrow (1)$	34	\div	44	GO TO (11)	54	\vee	64	int x	74	e^x		
05	5	15	f	25	\downarrow	35	\div	45	PRINT	55	\vee	65	ln x	75	$\log x$		
06	6	16	c	26	ENTER EXP	36	\div	46	END	56	\vee	66	TO RECT	76	\sqrt{x}		
07	7	17	d	27	\uparrow	37	CLEAR X	47	CONTINUE	57	PAUSE	67	hyper \vee				

Fig. 3. Pull-out instruction card is permanently attached to the calculator and contains key codes and operating instructions.

exponent is positive, or to the left if the exponent is negative.

$$4.398\ 364\ 291 \times 10^{-3} = .004\ 398\ 364\ 291.$$

The operator may choose to display numbers in FLOATING POINT or in FIXED POINT. The FLOATING POINT mode allows numbers, either positive or negative, from 1×10^{-99} to $9.999\ 999\ 999 \times 10^{99}$ to be displayed just as they are stored in the machine.

The FIXED POINT mode displays numbers in the way they are most commonly written. The DECIMAL DIGITS wheel allows setting the number of digits displayed to the right of the decimal point anywhere from 0 to 9. Fig. 2 shows a display of three numbers with the DECIMAL DIGITS wheel set at 5. The number in the Y register, $5.336\ 845\ 815 \times 10^5 = 533\ 684.5815$, is too big to be displayed in FIXED POINT without reducing

the DECIMAL DIGITS setting to 4 or less. If the number is too big for the DECIMAL DIGITS setting, the register involved reverts automatically to floating point to avoid an apparent overflow. In FIXED POINT display, the number displayed is rounded, but full significance is retained in storage for calculations.

To improve readability, 0's before the displayed number and un-entered 0's following the number are blanked. In FLOATING POINT, digits to the right of the decimal are grouped in threes.

Pull-Out Instruction Card

A pull-out instruction card, Fig. 3, is located at the front of the calculator under the keyboard. The operation of each key is briefly explained and key codes are listed. Some simple examples are provided to assist those using the machine for the first time or to refresh the memory of an infrequent user. Most questions regarding the operation of the

Model 9100A are answered on the card.

Data Entry

The calculator keyboard is shown in Fig. 4. Numbers can be entered into the X register using the digit keys, the π key or the ENTER EXP key. The ENTER EXP key allows powers of 10 to be entered directly which is useful for very large or very small numbers. 6.02×10^{23} is entered $\boxed{6} \boxed{0} \boxed{2} \boxed{\text{ENTER EXP}} \boxed{2} \boxed{3}$. If the ENTER EXP key is the first key of a number entry, a 1 is automatically entered into the mantissa. Thus only two keystrokes $\boxed{\text{ENTER EXP}} \boxed{6}$ suffice to enter 1,000,000. The CHG SIGN key changes the sign of either the mantissa or the exponent depending upon which one is presently being addressed. Numbers are entered in the same way, regardless of whether the machine is in FIXED POINT or

FLOATING POINT. Any key, other than a digit key, decimal point, CHG SIGN or ENTER EXP, terminates an entry; it is not necessary to clear before entering a new number. CLEAR X sets the X register to 0 and can be used when a mistake has been made in a number entry.

Control and Arithmetic Keys

ADD, SUBTRACT, MULTIPLY, DIVIDE involve two numbers, so the first number must be moved from X to Y before the second is entered into X. After the two numbers have been entered, the appropriate operation can be performed. In the case of a DIVIDE, the dividend is entered into Y and the divisor into X. Then the \div key is pressed causing the quotient to appear in Y, leaving the divisor in X.

One way to transfer a number from the X register to the Y register is to use the double sized key, \uparrow , at the left of the digit keys. This repeats the number in X into Y, leaving X unchanged; the number in Y goes to Z, and the number in Z is lost. Thus, when squaring or cubing a number, it is only necessary to follow \uparrow with \times or $\times \times$. The \downarrow key repeats a number in Z to Y leaving Z unchanged, the number in Y goes to X, and the number in X is lost. The $\uparrow \downarrow$ key rotates the number in the X and Y registers up and the number in Z down into X. $\downarrow \uparrow$ rotates the numbers in Z and Y down and the number in X up into Z. $x \leftrightarrow y$ interchanges the numbers in X and Y. Using the two ROLL keys and $x \leftrightarrow y$, numbers can be placed in any order in the three registers.

Functions Available from the Keyboard

The group of keys at the far left of the keyboard, Fig. 4, gives a good indication of the power of the Model

9100A. Most of the common mathematical functions are available directly from the keyboard. Except for $|y|$ the function keys operate on the number in X replacing it with the function of that argument. The numbers in Y and Z are left unchanged. \sqrt{x} is located with another group of keys for convenience but operates the same way.

The circular functions operate with angles expressed in RADIANS or DEGREES as set by the switch above the keyboard. The sine, cosine, or tangent of an angle is taken with a single keystroke. There are no restrictions on direction, quadrant or number of revolutions of the angle. The inverse functions are obtained by using the arc key as a prefix. For instance, two key depressions are necessary to obtain the arc sin x : arc $\sin x$. The angle obtained will be the standard principal value. In radians:

$$-\frac{\pi}{2} \leq \sin^{-1} x \leq \frac{\pi}{2}$$

$$0 \leq \cos^{-1} x \leq \pi$$

$$-\frac{\pi}{2} < \tan^{-1} x < \frac{\pi}{2}$$

The hyperbolic sine, cosine, or tangent is obtained using the hyper key as a prefix. The inverse hyperbolic functions are obtained with three key depressions. $\tanh^{-1} x$ is obtained by arc hyper $\tan x$. The arc and hyper keys prefix keys below them in their column.

Log x and $\ln x$ obtain the log to the base 10 and the log to the base e respectively. The inverse of the natural log is obtained with the e^x key. These keys are useful when raising numbers to odd powers as shown in one of the examples on the pull-out card, Fig. 3.

Two keys in this group are very useful in programs. $\text{int } x$ takes the integer part of the number in the X register



Fig. 4. Keys are in four groups on the keyboard, according to their function.

which deletes the part of the number to the right of the decimal point. For example $\text{int}(-3.1416) = -3$. **[Y]** forces the number in the Y register positive.

Storage Registers

Sixteen registers, in addition to X, Y, and Z, are available for storage. Fourteen of them, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, a, b, c, d, can be used to store either one constant or 14 program steps per register. The last registers, e and f, are normally used only for constant storage since the program counter will not cycle into them. Special keys located in a block to the left of the digit keys are used to identify the lettered registers.

To store a number from the X register the key **[X→i]** is used. The parenthesis indicates that another key depression, representing the storage register, is necessary to complete the transfer. For example, storing a number from the X register into register 8 requires two key depressions: **[X→i]** **[8]**. The X register remains unchanged. To store a number from Y register the key **[Y→i]** is used.

The contents of the alpha registers are recalled to X simply by pressing the keys a, b, c, d, e, and f. Recalling a number from a numbered register requires the use of the **[Y↔i]** key to distinguish the recall procedure from digit entry. This key interchanges the number in the Y register with the number in the register indicated by the following keystroke, alpha or numeric, and is also useful in programs since neither number involved in the transfer is lost.

The CLEAR key sets the X, Y, and Z display registers and the f and e registers to zero. The remaining registers are not affected. The f and e registers are set to zero to initialize them for use with the **[ACC+]** and **[ACC-]** keys as will be explained. In addition the CLEAR key clears the FLAG and the ARC and HYPER conditions, which often makes it a very useful first step in a program.

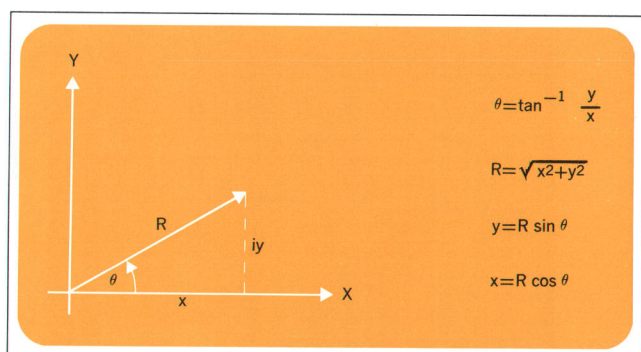


Fig. 5. Variables involved in conversions between rectangular and polar coordinates.

Coordinate Transformation and Complex Numbers

Vectors and complex numbers are easily handled using the keys in the column on the far left of the keyboard. Fig. 5 defines the variables involved. Angles can be either in degrees or radians. To convert from rectangular to polar coordinates, with y in Y and x in X, press **[TO POLAR]**. Then the display shows θ in Y and R in X. In converting from polar to rectangular coordinates, θ is placed in Y, and R in X, **[TO RECT]** is pressed and the display shows y in Y and x in X.

ACC+ and ACC- allow addition or subtraction of vector components in the f and e storage registers. ACC+ adds the contents of the X and Y register to the numbers already stored in f and e respectively; ACC- subtracts them. The RCL key recalls the numbers in the f and e registers to X and Y.

Illegal Operations

A light to the left of the CRT indicates that an illegal operation has been performed. This can happen either from the keyboard or when running a program. Pressing any key on the keyboard will reset the light. When running a program, execution will continue but the light will remain on as the program is completed. The illegal operations are:

- Division by zero.
- \sqrt{x} where $x < 0$
- $\ln x$ where $x \leq 0$; $\log x$ where $x \leq 0$
- $\sin^{-1} x$ where $|x| > 1$; $\cos^{-1} x$ where $|x| > 1$
- $\cosh^{-1} x$ where $x < 1$; $\tanh^{-1} x$ where $|x| > 1$

Accuracy

The Model 9100A does all calculations using floating point arithmetic with a twelve digit mantissa and a two digit exponent. The two least significant digits are not displayed and are called guard digits.

The algorithms used to perform the operations and generate the functions were chosen to minimize error and to provide an extended range of the argument. Usually any inaccuracy will be contained within the two guard digits. In certain cases some inaccuracy will appear in the displayed number. One example is where the functions change rapidly for small changes in the argument, as in $\tan x$ where x is near 90° . A glaring but insignificant inaccuracy occurs when an answer is known to be a whole number, but the least significant guard digit is one count low: $2.000\,000\,000 \simeq 1.999\,999\,999$.

Accuracy is discussed further in the 'Internal Programming' article in this issue. But a simple summary

is: the answer resulting from any operation or function will lie within the range of true values produced by a variation of ± 1 count in the tenth digit of the argument.

Programming

Problems that require many keyboard operations are more easily solved with a program. This is particularly true when the same operations must be performed repeatedly or an iterative technique must be used. A program library supplied with the Model 9100A provides a set of representative programs from many different fields. If a program cannot be found in the library to solve a particular problem, a new program can easily be written since no special experience or prior knowledge of a programming language is necessary.

Any key on the keyboard can be remembered by the calculator as a program step except STEP PRGM. This key is used to 'debug' a program rather than as an operation in a program. Many individual program steps, such as 'sin x' or 'to polar' are comparatively powerful, and avoid the need of sub-routines for these functions and the programming space such sub-routines require. Registers 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, a, b, c, d can store 14 program steps each. Steps within the registers are numbered 0 through d just as the registers themselves are numbered. Programs can start at any of the 196 possible addresses. However 0-0 is usually used for the first step. Address d-d is then the last available, after which the program counter cycles back to 0-0.

Registers f and e are normally used for storage of constants only, one constant in each register. As more constant storage is required, it is recommended that registers d, then c, then b, etc., are used starting from the bottom of the list. Lettered registers are used first, for the frequently recalled constants, because constants stored in them are more easily recalled. A register can be used to store one constant or 14 program steps, but not both.

Branching

The bank on the far right of the keyboard, Fig. 4, contains program oriented keys. **GO TO** is used to set the program counter. The two sets of parentheses indicate that this key should be followed by two more key depressions indicating the address of the program step desired. As a program step, 'GO TO' is an unconditional branch instruction, which causes the program to branch to the address given by the next two program steps. The 'IF' keys in this group are conditional branch instructions. With **IF** $x < y$, **IF** $x = y$, and **IF** $x > y$ the numbers contained in the X and Y registers are compared. The indicated condition is

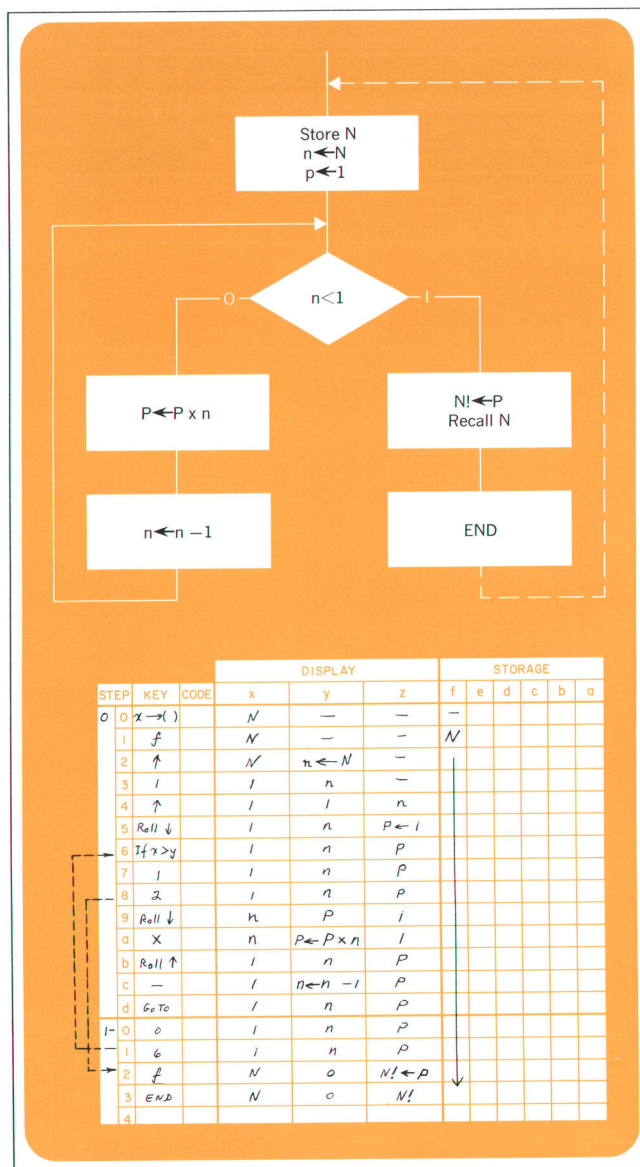


Fig. 6. Flow chart of a program to compute $N!$ (top). Each step is shown (bottom) and the display for each register. A new value for N can be entered at the end of the program, since END automatically sets the program counter back to 0-0.

tested and, if met, the next two program steps are executed. If the first is alphameric, the second must be also, and the two steps are interpreted as a branching address. When the condition is not met, the next two steps are skipped and the program continues. **IF FLAG** is also a very useful conditional branching instruction which tests a 'yes' or 'no' condition internally stored in the calculator. This condition is set to 'yes' with the SET FLAG from the keyboard when the calculator is in the display mode or from a program as a program step. The flag is set to a 'no' condition by either asking IF FLAG in a program

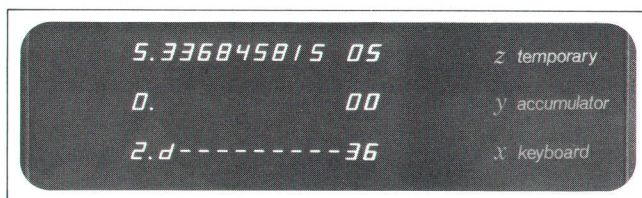


Fig. 7. Program step address and code are displayed in the X register as steps are entered. After a program has been entered, each step can be checked using the STEP PRGM key. In this display, step 2-d is 36, the code for multiply.

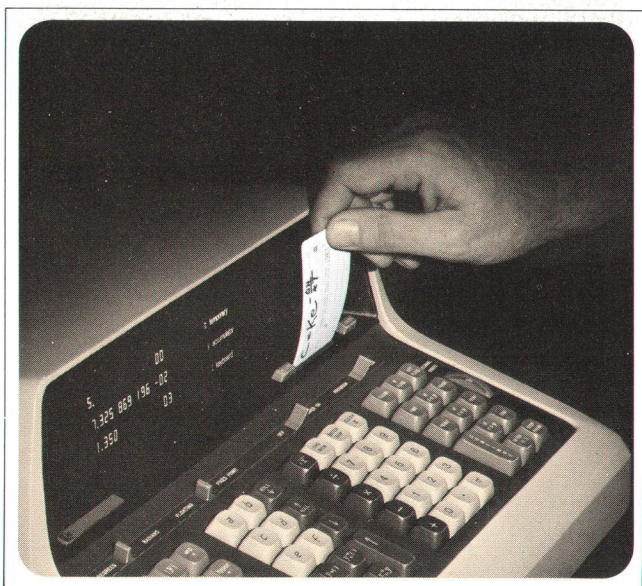


Fig. 8. Programs can be entered into the calculator by means of the magnetic program card. The card is inserted into the slot and the ENTER button pressed.

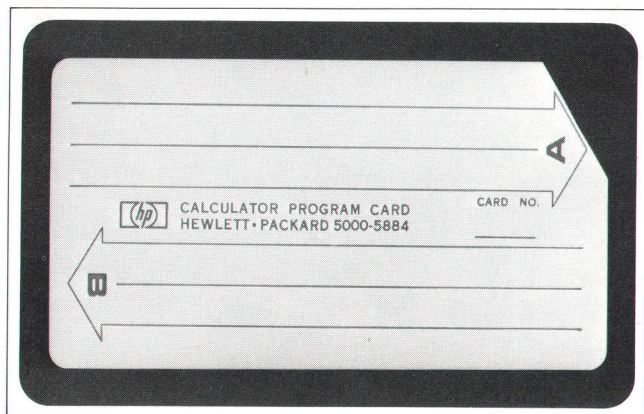


Fig. 9. Magnetic programming card can record two 196-step programs. To prevent accidental recording of a new program over one to be saved, the corner of the card is cut as shown.

or by a CLEAR instruction from the keyboard or from a program.

Data Input and Output

Data can be entered for use in a program when the machine is in the display mode. (The screen is blank while a program is running.) A program can be stopped in several ways. The **STOP** key will halt the machine at any time. The operation being performed will be completed before returning to the display mode. As a program step, STOP stops the program so that answers can be displayed or new data entered. END must be the last step in a program listing to signal the magnetic card reader; when encountered as a program step it stops the machine and also sets the program counter to 0-0.

As a program step, PAUSE causes a brief display during program execution. Nine cycles of the power line frequency are counted—the duration of the pause will be about 150 ms for a 60 Hz power line or 180 ms for a 50 Hz power line. More pauses can be used in sequence if a longer display is desired. While a program is running the PAUSE key can be held down to stop the machine when it comes to the next PAUSE in the program. PAUSE provides a particularly useful way for the user and the machine to interact. It might, for instance, be used in a program so that the convergence to a desired result can be observed.


Other means of input and output involve peripheral devices such as an X-Y Plotter or a Printer. The PRINT key activates the printer, causing it to print information from the display register. As a program step, PRINT will interrupt the program long enough for the data to be accepted by the printer and then the program will continue. If no printer is attached, PRINT as a program step will act as a STOP. The FMT key, followed by any other keystroke, provides up to 62 unique commands to peripheral equipment. This flexibility allows the Model 9100A to be used as a controller in small systems.



Sample Program — N!

A simple program to calculate N! demonstrates how the Model 9100A is programmed. Fig. 6 (top) shows a flow chart to compute N! and Fig. 6 (bottom) shows the program steps. With this program, 60! takes less than 1/2 second to compute.

Program Entry and Execution

After a program is written it can be entered into the Model 9100A from the keyboard. The program counter is set to the address of the first program step by using the

GO TO () () key. The RUN-PROGRAM switch is then switched from RUN to PROGRAM and the program steps entered in sequence by pushing the proper keys. As each step is entered the X register displays the address and key code, as shown in Fig. 7. The keys and their codes are listed at the bottom of the pull-out card, Fig. 3. Once a program has been entered, the steps can be checked using the STEP PRGM key in the PROGRAM mode as explained in Fig. 7. If an error is made in a step, it can be corrected by using the  key without having to re-enter the rest of the program.

To run a program, the program counter must be set to the address of the first step. If the program starts at 0-0 the keys  0 0 are depressed, or simply just  since this key automatically sets the program counter to 0-0. CONTINUE will start program execution.

Magnetic Card Reader-Recorder

One of the most convenient features of the Model 9100A is the magnetic card reader-recorder, Fig. 8. A program stored in the Model 9100A can be recorded on a magnetic card, Fig. 9, about the size of a credit card. Later when the program is needed again, it can be quickly re-entered using the previously recorded card. Cards are easily duplicated so that programs of common interest can be distributed.

As mentioned earlier, the END statement is a signal to the reader to stop reading recorded information from the card into the calculator. For this reason END should not be used in the middle of a program. Since most programs start at location 0-0 the reader automatically initializes the program counter to 0-0 after a card is read.



Richard E. Monnier

Dick Monnier has a BSEE from the University of California at Berkeley, 1958, and an MSEE from Stanford University, 1961.

Dick has been with HP since 1958 when he joined the group working on sampling oscilloscopes. He has been project leader on the Models 120B, 132A and 140A oscilloscopes and the Model 191A television waveform monitor prior to the transfer of the project to HP's Colorado Springs Division. Since early 1966 he has been Section Leader responsible for the 9100A Calculator project in Palo Alto.

Dick holds several patents and is a Member of IEEE.

The magnetic card reader makes it possible to handle most programs too long to be held in memory at one time. The first entry of steps can calculate intermediate results which are stored in preparation for the next part of the program. Since the reader stops reading at the END statement these stored intermediate results are not disturbed when the next set of program steps is entered. The stored results are then retrieved and the program continued. Linking of programs is made more convenient if each part can execute an END when it finishes to set the program counter to 0-0. It is then only necessary to press CONTINUE after each entry of program steps.

SPECIFICATIONS

HP Model 9100A

The HP Model 9100A is a programmable, electronic calculator which performs operations commonly encountered in scientific and engineering problems. Its log, trig and mathematical functions are each performed with a single key stroke, providing fast, convenient solutions to intricate equations. Computer-like memory enables the calculator to store instructions and constants for repetitive or iterative solutions. The easily-readable cathode ray tube instantly displays entries, answers and intermediate results.

OPERATIONS

DIRECT KEYBOARD OPERATIONS INCLUDE:

ARITHMETIC: addition, subtraction, multiplication, division and square-root.

LOGARITHMIC: $\log x$, $\ln x$ and e^x .

TRIGONOMETRIC: $\sin x$, $\cos x$, $\tan x$, $\sin^{-1}x$, $\cos^{-1}x$ and $\tan^{-1}x$ (x in degrees or radians).

HYPERBOLIC: $\sinh x$, $\cosh x$, $\tanh x$, $\sinh^{-1}x$, $\cosh^{-1}x$, and $\tanh^{-1}x$.

COORDINATE TRANSFORMATION: polar-to-rectangular, rectangular-to-polar, cumulative addition and subtraction of vectors.

MISCELLANEOUS: other single-key operations include—taking the absolute value of a number, extracting the

integer part of a number, and entering the value of π . Keys are also available for positioning and storage operations.

PROGRAMMING

The program mode allows entry of program instructions, via the keyboard, into program memory. Programming consists of pressing keys in the proper sequence, and any key on the keyboard is available as a program step. Program capacity is 196 steps. No language or code-conversions are required.

A self-contained magnetic card reader/recorder records programs from program memory onto wallet-size magnetic cards for storage. It also reads programs from cards into program memory for repetitive use. Two programs of 196 steps each may be recorded on each reusable card. Cards may be cascaded for longer programs.

SPEED

Average times for total performance of typical operations, including decimal-point placement:

add, subtract: 2 milliseconds
multiply: 12 milliseconds
divide: 18 milliseconds
square-root: 19 milliseconds
 \sin , \cos , \tan : 280 milliseconds
 $\ln x$: 50 milliseconds
 e^x : 110 milliseconds

These times include core access of 1.6 microseconds.

GENERAL

WEIGHT: Net 40 lbs. (18.1 kg.); shipping 65 lbs. (29.5 kg.).

POWER: 115 or 230 V $\pm 10\%$, 50 to 60 Hz, 400 Hz, 70 watts.

DIMENSIONS: 8 1/4" high, 16" wide, 19" deep.

ACCESSORIES FURNISHED AT NO CHARGE:

09100-90001 Operating and Programming manual, \$5.00.

09100-90002 Program library binder containing sample programs, \$30.00.

5060-5919 Box of 10 magnetic program cards, \$10.00.

09100-90003 Pad of 100 program sheets, \$2.50.

09100-90004 Magnetic card with pre-recorded diagnostic program, \$2.50.

9320-1157 Pull-out instruction card mounted in calculator, \$5.00.

4040-0350 Plastic dust cover, \$2.50.

ADDITIONAL ACCESSORIES AVAILABLE:

5000-5884 Single magnetic card, \$2.00

09100-90000 Box of 5 program pads, \$10.00.

PRICE: HP Model 9100A, \$4900.00.

PERIPHERALS: Printer, X-Y plotter and input/output interface will be available soon.

MANUFACTURING DIVISION: LOVELAND DIVISION

P.O. Box 301
815 Fourteenth Street N.W.
Loveland, Colorado 80537

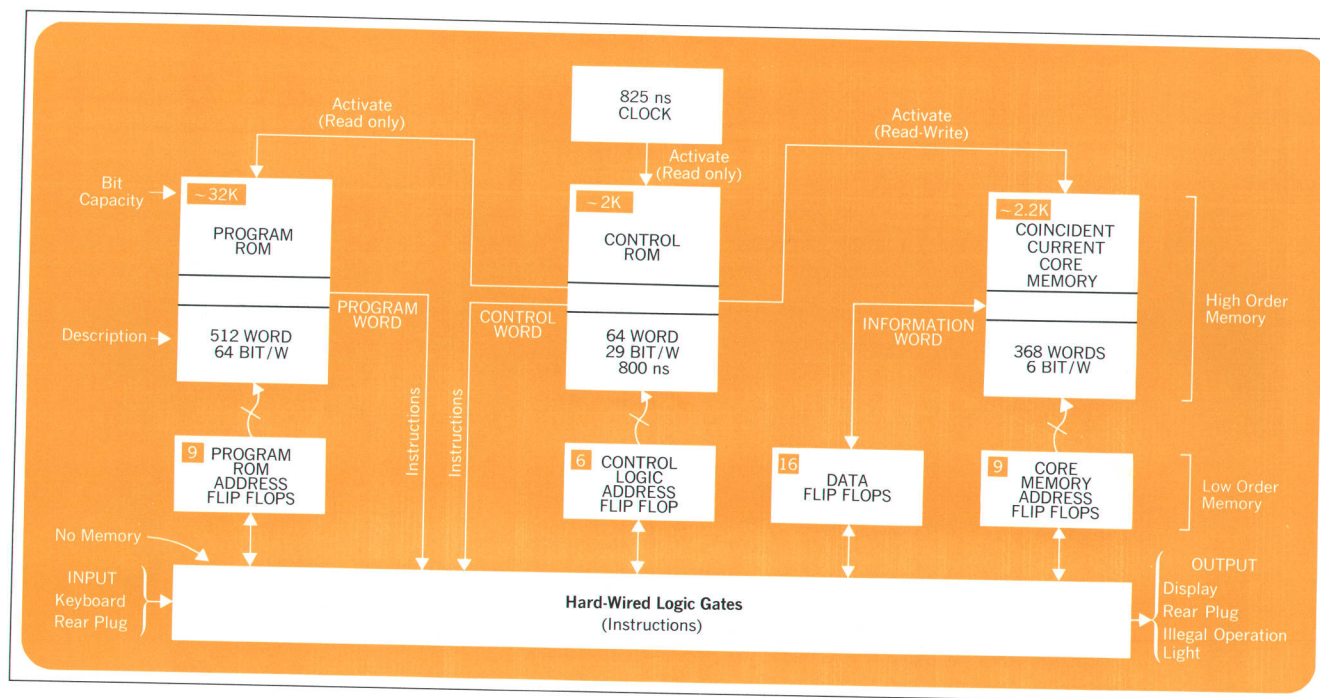


Fig. 1. Arithmetic processing unit block diagram. This system is a marriage of conventional, reliable diode-resistor logic to a 32,000-bit read-only memory and a coincident current core memory.

Hardware Design of the Model 9100A Calculator

By Thomas E. Osborne

ALL KEYBOARD FUNCTIONS IN THE MODEL 9100A are implemented by the arithmetic processing unit, Figs. 1 and 2. The arithmetic unit operates in discrete time periods called clock cycles. All operations are synchronized by the clock shown at the top center of Fig. 1.

The clock is connected to the control read only memory (ROM) which coordinates the operation of the program read only memory and the coincident current core read/write memory. The former contains information for implementing all of the keyboard operations while the latter stores user data and user programs.

All internal operations are performed in a digit by digit serial basis using binary coded decimal digits. An addition, for example, requires that the least significant digits of the addend and augend be extracted from core, then added and their sum replaced in core. This process is repeated one BCD digit at a time until the most significant digits have been processed. There is also a substantial amount of 'housekeeping' to be performed such

as aligning decimal points, assigning the proper algebraic sign, and floating point normalization. Although the implementation of a keyboard function may involve thousands of clock cycles, the total elapsed time is in the millisecond region because each clock cycle is only 825 ns long.

The program ROM contains 512 64-bit words. When the program ROM is activated, signals (micro-instructions) corresponding to the bit pattern in the word are sent to the hard wired logic gates shown at the bottom of Fig. 1. The logic gates define the changes to occur in the flip flops at the end of a clock cycle. Some of the micro-instructions act upon the data flip flops while others change the address registers associated with the program ROM, control ROM and coincident current core memory. During the next clock cycle the control ROM may ask for a new set of micro-instructions from the program ROM or ask to be read from or written into the coincident current core memory. The control ROM also has

the ability to modify its own address register and to issue micro-instructions to the hard wired logic gates. This flexibility allows the control logic ROM to execute special programs such as the subroutine for unpacking the stored constants required by the keyboard transcendental functions.

Control Logic

The control logic uses a wire braid toroidal core read only memory containing 64 29-bit words. Magnetic logic of this type is extremely reliable and pleasingly compact.

The crystal controlled clock source initiates a current pulse having a trapezoidal waveform which is directed through one of 64 word lines. Bit patterns are generated by passing or threading selected toroids with the word lines. Each toroid that is threaded acts as a transformer to turn on a transistor connected to the output winding of the toroid. The signals from these transistors operate the program ROM, coincident current core, and selected micro-instructions.

Coincident Current Core Read/Write Memory

The 2208 (6 x 16 x 23) bit coincident current memory uses wide temperature range lithium cores. In addition, the X, Y, and inhibit drivers have temperature compensated current drive sources to make the core memory insensitive to temperature and power supply variations.

The arithmetic processing unit includes special circuitry to guarantee that information is not lost from the core memory when power is turned off and on.

Power Supplies

The arithmetic processing unit operates from a single -15 volt supply. Even though the power supply is highly regulated, all circuits are designed to operate over a voltage range of -13.5 to -16.5 volts.

Display

The display is generated on an HP electrostatic cathode ray tube only 11 inches long. The flat rectangular face plate measures $3\frac{1}{4}$ " x $4\frac{13}{16}$ " inches. The tube was specifically designed to generate a bright image. High contrast is obtained by using



Fig. 3. Displayed characters are generated by modulating these figures. The digit 1 is shifted to the center of the pattern.

a low transmissivity filter in front of the CRT. Ambient light that usually tends to 'wash out' an image is attenuated twice by the filter, while the screen image is only attenuated once.

All the displayed characters are 'pieces of eight'. Sixteen different symbols are obtained by intensity modulating a figure 8 pattern as shown in Fig. 3. Floating point numbers are partitioned into groups of three digits and the numeral 1 is shifted to improve readability. Zeros to the left of the most significant digit and insignificant zeros to the right of the decimal point are blanked to avoid a confusing display. Fixed point numbers are automatically rounded up according to the decimal wheel setting. A fixed point display will automatically revert to floating point notation if the number is too large to be displayed on the CRT in fixed point.

Multilayer Instruction Logic Board

All of the hard wired logic gates are synthesized on

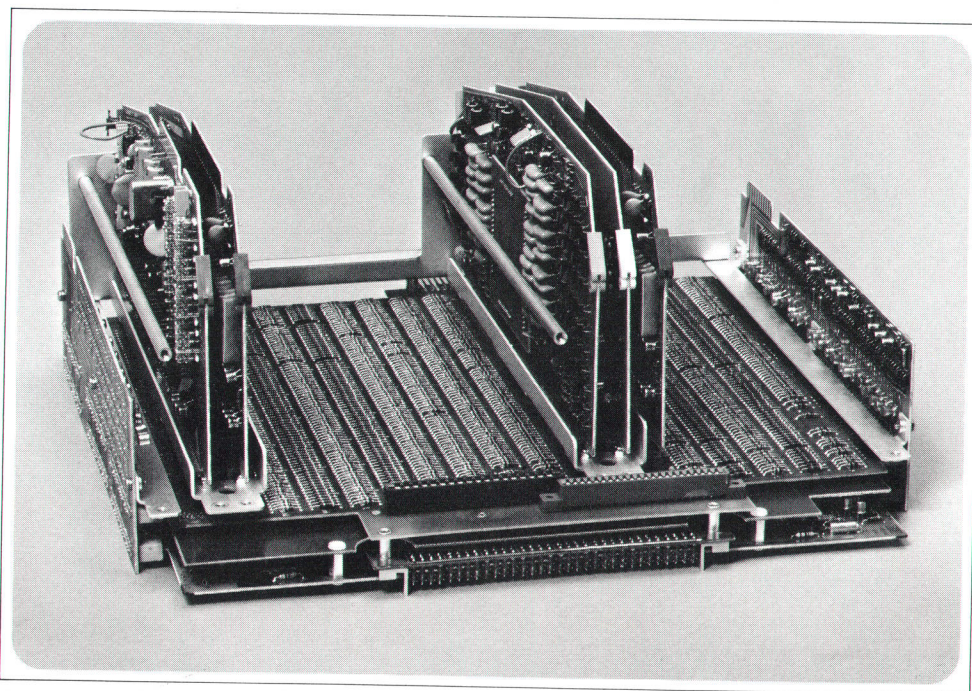


Fig. 2. Arithmetic unit assembly removed from the calculator.

the instruction logic board using time-proven diode-resistor logic. The diodes and resistors are located in separate rows, Fig. 4. All diodes are oriented in the same direction and all resistors are the same value. The maze of interconnections normally associated with the back plane wiring of a computer are located on the six internal layers of the multilayer instruction logic board. Solder bridges and accidental shorts caused by test probes leads beneath components are all but eliminated by not having interconnections on the two outside surfaces of this multilayer board. The instruction logic board also serves as a motherboard for the control logic board, the two coincident core boards and the two flip flop boards, the magnetic card reader, and the keyboard. It also contains a connector, available at the rear of the calculator, for connecting peripherals.

Flip Flops

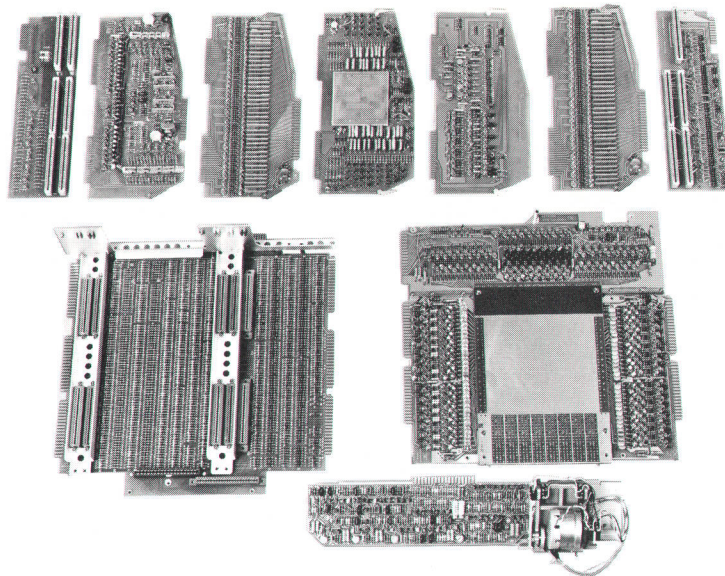
The Model 9100A contains 40 identical J-K flip flops, each having a threshold noise immunity of 2.5 volts. Worst case design techniques guarantee that the flip flops will operate at 3 MHz even though 1.2 MHz is the maximum operating rate.

Program Read Only Memory

The 32,768 bit read only program memory consists of 512 64-bit words. These words contain all of the operating subroutines, stored constants, character encoders, and CRT modulating patterns. The 512 words are contained in a 16 layer printed-circuit board having drive and sense lines orthogonally located. A drive line consists of a reference line and a data line. Drive pulses are inductively coupled from both the reference line and data line into the sense lines. Signals from the data line either aid or cancel signals from the reference line producing either a 1 or 0 on the output sense lines. The drive and sense lines are arranged to achieve a bit density in the ROM data board of 1000 bits per square inch.

The program ROM decoder/driver circuits are located directly above the ROM data board. Thirty-two com-

Fig. 4. Printed circuit boards which make up the arithmetic unit are, left to right at top, side board, control logic, flip-flop, core and drivers, core sense amplifiers and inhibit, flip-flop, and side board. Large board at the lower left is the multilayer instruction board and the program ROM is at the right. The magnetic card reader and its associated circuitry is at the bottom.



bination sense amplifier, gated-latch circuits are located on each side of the ROM data board. The outputs of these circuits control the hard wired logic gates on the instruction logic board.

Side Boards

The program ROM printed circuit board and the instruction logic board are interconnected by the side boards, where preliminary signal processing occurs.

The Keyboard

The keyboard contains 63 molded plastic keys. Their markings will not wear off because the lettering is imbedded into the key body using a double shot injection molding process. The key and switch assembly was specifically designed to obtain a pleasing feel and the proper amount of tactile and aural feedback. Each key operates a single switch having gold alloy contacts. A contact closure activates a matrix which encodes signals on six data lines and generates an initiating signal. This signal is delayed to avoid the effects of contact bounce. An electrical interlock prevents errors caused by pressing more than one key at a time.

Magnetic Card Reader

Two complete 196 step programs can be recorded on the credit card size magnetic program card. The recording process erases any previous information so that a card may be used over and over again. A program may be protected against accidental erasure by clipping off the corner of the card, Fig. 9, page 8. The missing cor-



Fig. 5. Internal adjustments of the calculator are easily accessible by removing a few screws and lifting the top.

ner deactivates the recording circuitry in the magnetic card reader. Program cards are compatible among machines.

Information is recorded in four tracks with a bit density of 200 bits per inch. Each six-bit program step is split into two time-multiplexed, three-bit codes and recorded on three of the four tracks. The fourth track provides the timing strobe.

Information is read from the card and recombined into six bit codes for entry into the core memory. The magnetic card reading circuitry recognizes the 'END' program code as a signal to end the reading process. This feature makes it possible to enter subroutines within the body of a main program or to enter numeric constants via the program card. The END code also sets the program counter to location 0-0, the most probable starting location. The latter feature makes the Model 9100A ideally suited to 'linking' programs that require more than 196 steps.


Packaging and Servicing

The packaging of the Model 9100A began by giving the HP industrial design group a volume estimate of the electronics package, the CRT display size and the number of keys on the keyboard. Several sketches were drawn and the best one was selected. The electronics sections were then specifically designed to fit in this case. Much time and effort were spent on the packaging of the arithmetic processing unit. The photographs, Figs. 2 and 5 attest to the fact that it was time well spent.

The case covers are die cast aluminum which offers durability, effective RFI shielding, excellent heat transfer characteristics, and convenient mechanical mounts. Removing four screws allows the case to be opened and locked into position, Fig. 5. This procedure exposes all important diagnostic test points and adjustments. The keyboard and arithmetic processing unit may be freed by removing four and seven screws respectively.

Any component failures can be isolated by using a diagnostic routine or a special tester. The faulty assembly is then replaced and is sent to a service center for computer assisted diagnosis and repair.

Reliability

Extensive precautions have been taken to insure maximum reliability. Initially, wide electrical operating margins were obtained by using 'worst case' design techniques. In production all transistors are aged at 80% of rated power for 96 hours and tested before being used in the Model 9100A. Subassemblies are computer tested and actual operating margins are monitored to detect trends that could lead to failures. These data are analyzed and corrective action is initiated to reverse the trend. In addition, each calculator is operated in an environmental chamber at 55°C for 5 days prior to shipment to the customer. Precautions such as these allow Hewlett-Packard to offer a one year warranty in a field where 90 days is an accepted standard. 



Thomas E. Osborne

Tom Osborne joined HP as a consultant in late 1965 with the responsibility for developing the architecture of the Model 9100A. Previous to joining HP, he had designed data processing equipment, then formed Logic Design Co., where he developed a floating point calculator upon which the Model 9100A is based.

Tom graduated from the University of Wyoming in 1957 with a BSEE, and was named 'Outstanding Electrical Engineer' of his class. He received his MSEE from the University of California at Berkeley.

Tom enjoys flying as a pastime, he is an ardent theater-goer and a connoisseur of fine wines. He is a member of Sigma Tau and Phi Kappa Phi honorary fraternities, and a member of IEEE.

Internal Programming of the 9100A Calculator

By David S. Cochran

EXTENSIVE INTERNAL PROGRAMMING has been designed into the HP Model 9100A Calculator to enable the operator to enter data and to perform most arithmetic operations necessary for engineering and scientific calculation with a single key stroke or single program step. Each of the following operations is a hardware subroutine called by a key press or program step:

Basic arithmetic operations

- Addition
- Subtraction
- Multiplication
- Division

Extended arithmetic operations

- Square root
- Exponential — e^x
- Logarithmic — $\ln x$, $\log x$
- Vector addition and subtraction

Trigonometric operations

- $\sin x$, $\cos x$, $\tan x$
- $\arcsin x$, $\arccos x$, $\arctan x$
- $\sinh x$, $\cosh x$, $\tanh x$
- $\operatorname{arsinh} x$, $\operatorname{arcosh} x$, $\operatorname{artanh} x$
- Polar to rectangular and rectangular to polar coordinate transformation

Miscellaneous

- Enter π
- Absolute value of y
- Integer value of x

In the evolution of internal programming of the Model 9100A Calculator, the first step was the development of flow charts of each function. Digit entry, Fig. 1, seemingly a trivial function, is as complex as most of the mathematical functions. From this functional description, a detailed program can be written which uses the microprograms and incremental instructions of the calculator. Also, each program must be married to all of the other programs which make up the hard-wired software of the Model 9100A. Mathematical functions are similarly pro-

grammed defining a step-by-step procedure or algorithm for solving the desired mathematical problem.

The calculator is designed so that lower-order subroutines may be nested to a level of five in higher-order functions. For instance, the 'Polar to Rectangular' function uses the \sin routine which uses multiply which uses add, etc.

Addition and Subtraction

The most elementary mathematical operation is algebraic addition. But even this is relatively complex — it requires comparing signs and complementing if signs are unlike. Because all numbers in the Model 9100A are processed as true floating point numbers, exponents must be subtracted to determine proper decimal alignment. If one of the numbers is zero, it is represented in the calculator by an all-zero mantissa with zero exponent. The difference between the two exponents determines the offset, and rather than shifting the smaller number to the right, a displaced digit-by-digit addition is performed. It must also be determined if the offset is greater than 12, which is the resolution limit.

Although the display shows 10 significant digits, all calculations are performed to 12 significant digits with the two last significant digits (guard digits) absorbing truncation and round-off errors. All registers are in core memory, eliminating the need for a large number of flip-flop registers. Even with the display in 'Fixed Point' mode, every computed result is in storage in 12 digits.

Multiplication

Multiplication is successive addition of the multiplicand as determined by each multiplier digit. Offset in the digit position flip-flops is increased by one after completion of the additions by each multiplier digit. Exponents are added after completion of the product. Then the product is normalized to justify a carry digit which might have occurred.

Division

Division involves repeated subtraction of the divisor

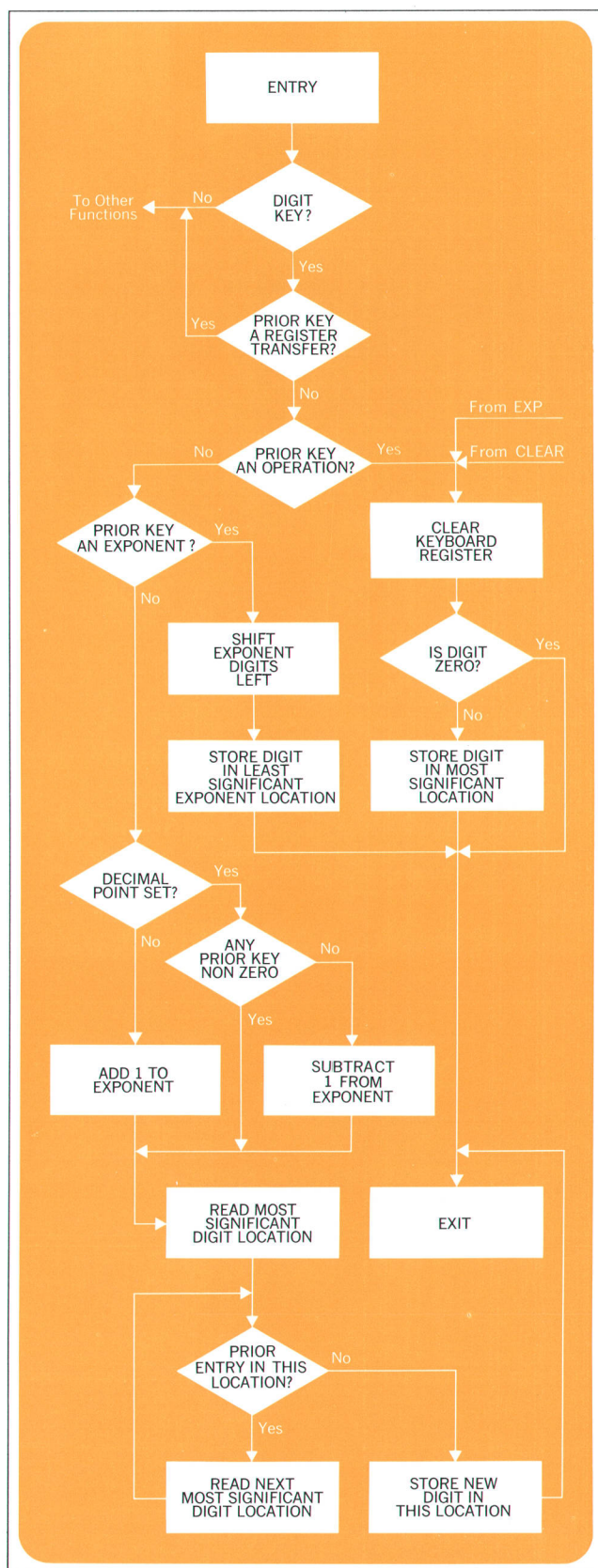


Fig. 1. Flow chart of a simple digit entry. Some of these flow paths are used by other calculator operations for greater hardware efficiency.

from the dividend until an overdraft occurs. At each subtraction without overdraft, the quotient digit is incremented by one at the digit position of iteration. When an overdraft occurs, the dividend is restored by adding the divisor. The division digit position is then incremented and the process continued. Exponents are subtracted after the quotient is formed, and the quotient normalized.

Square Root

Square root, in the Model 9100A, is considered a basic operation and is done by pseudo division. The method used is an extension of the integer relationship.

$$\sum_{i=1}^n 2i - 1 = n^2$$

In square root, the divisor digit is incremented at each iteration, and shifted when an overdraft and restore occurs. This is a very fast algorithm for square root and is equal in speed to division.

Circular Routines

The circular routines (sin, cos, tan), the inverse circular routines (arcsin, arccos, arctan) and the polar to rectangular and rectangular to polar conversions are all accomplished by iterating through a transformation which rotates the axes. Any angle may be represented as an angle between 0 and 1 radian plus additional information such as the number of times $\pi/2$ has been added or subtracted, and its sign. The basic algorithm for the forward circular function operates on an angle whose absolute value is less than 1 radian, but prescaling is necessary to indicate quadrant.

To obtain the scaling constants, the argument is divided by 2π , the integer part discarded and the remaining fraction of the circle multiplied by 2π . Then $\pi/2$ is subtracted from the absolute value until the angle is less than 1 radian. The number of times $\pi/2$ is subtracted, the original sign of the argument, and the sign upon completion of the last subtraction make up the scaling constants. To preserve the quadrant information the scaling constants are stored in the core memory.

The algorithm produces $\tan \theta$. Therefore, in the Model

9100A, $\cos \theta$ is generated as $\frac{1}{\sqrt{1 + \tan^2 \theta}}$ and

$\sin \theta$ as $\frac{\tan \theta}{\sqrt{1 + \tan^2 \theta}}$.

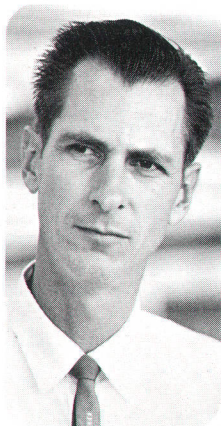
$\sin \theta$ could be obtained from the relationship $\sin \theta = \sqrt{1 - \cos^2 \theta}$, for example, but the use of the tangent relationship preserves the 12 digit accuracy for very small angles, even in the range of $\theta < 10^{-12}$. The proper signs of the functions are assigned from the scaling constants.

For the polar to rectangular functions, $\cos \theta$ and $\sin \theta$ are computed and multiplied by the radius vector to obtain the X and Y coordinates. In performing the rectangular to polar function, the signs of both the X and Y vectors are retained to place the resulting angle in the right quadrant.

Prescaling must also precede the inverse circular functions, since this routine operates on arguments less than or equal to 1. The inverse circular algorithm yields arc-tangent functions, making it necessary to use the trigonometric identity.

$$\sin^{-1}(x) = \tan^{-1} \frac{x}{\sqrt{1 - x^2}}.$$

If a $\cos^{-1}(x)$ is desired, the arcsin relationship is used and a scaling constant adds $\pi/2$ after completion of the function. For arguments greater than 1, the arccotangent of the negative reciprocal is found which yields the arc-tangent when $\pi/2$ is added.



David S. Cochran

Dave Cochran has been with HP since 1956. He received his BSEE from Stanford in 1958 and his MS from the same school in 1960. He has been responsible for the development of a broad range of instruments including the Model 3440A Digital Voltmeter and the 204B Oscillator. He has been working on development of electrostatic printing. Dave developed the internal programming for the Model 9100A Calculator.

He holds six patents in the instrumentation field and is the author of several published papers. Dave is a registered professional engineer, a member of IEEE and Tau Beta Pi.

Exponential and Logarithms

The exponential routine uses a compound iteration algorithm which has an argument range of 0 to the natural log of 10 ($\ln 10$). Therefore, to be able to handle any argument within the dynamic range of the calculator, it is necessary to prescale the absolute value of the argument by dividing it by $\ln 10$ and saving the integer part to be used as the exponent of the final answer. The fractional part is multiplied by $\ln 10$ and the exponential found. This number is the mantissa, and with the previously saved integer part as a power of 10 exponent, becomes the final answer.

The exponential answer is reciprocated in case the original argument was negative, and for use in the hyperbolic functions. For these hyperbolic functions, the following identities are used:

$$\sinh x = \frac{e^x - e^{-x}}{2}, \cosh x = \frac{e^x + e^{-x}}{2}, \tanh x = \frac{e^x - e^{-x}}{e^x + e^{-x}}.$$

Natural Logarithms

The exponential routine in reverse is used as the routine for natural logs, with only the mantissa operated upon. Then the exponent is multiplied by $\ln 10$ and added to the answer. This routine also yields these \log_{10} and arc hyperbolic functions:


$$\log_{10} x = \frac{\ln x}{\ln 10}; \sinh^{-1}(x) = \ln(x + \sqrt{x^2 + 1});$$

$$\cosh^{-1}(x) = \ln(x + \sqrt{x^2 - 1}); \tanh^{-1}(x) = \ln \sqrt{\frac{1+x}{1-x}}.$$

The $\sinh^{-1}(x)$ relationship above yields reduced accuracy for negative values of x . Therefore, in the Model 9100A, the absolute value of the argument is operated upon and the correct sign affixed after completion.

Accuracy

It can be seen from the discussion of the algorithms that extreme care has been taken to use routines that have accuracy commensurate with the dynamic range of the calculator. For example; the square root has a maximum possible relative error of 1 part in 10^{10} over the full range of the machine.

There are many algorithms for determining the sine of an angle; most of these have points of high error. The sine routine in the Model 9100A has consistent low error regardless of quadrant. Marrying a full floating decimal calculator with unique mathematical algorithms results in accuracy of better than 10 displayed digits. 

Computer-Testing the HP Model 9100A Calculator

By Charles W. Near

WITH SEVERAL THOUSAND DIODES AND TRANSISTORS plus three different memory arrays making up the 9100A Calculator, the probability is near zero that an untested calculator will operate perfectly at initial turn-on. Aging and pretesting of the transistors add greatly to reliability, but the total test time per calculator remains high unless rigorous testing is performed on the logic subassemblies. More than a million test measurements per calculator are required to assure acceptable performance of the 40 flip-flops, the braid memory control logic, the core memory and the main read only memory.

Use of a digital computer, Fig. 1, to control this testing offers a number of important advantages. Many tests may be performed in a short time and the test data stored in computer memory. The computer can analyze the test data and provide instantaneous print-out of fault diagnostics. By using programmable power supplies, the actual operating margins may be measured and printed, thereby simulating adverse environments or detecting processing changes which may influence the performance range of the assembly. Test automation also assures that the test is repeated identically on a day to day run or run to run basis for greatest accuracy.

Test System

An HP Model 2116B Computer with 8192 words of core memory has been coupled to two scanners, a digital voltmeter and three programmable power supplies for production testing of the calculator, Fig. 2. The scanners have 100 inputs each with reed relay contacts capable of 200 scan points per second. A Model ASR-35 teleprinter provides a printed output of test results and permits communication with the system during test.

A test interface unit has been built for each logic subassembly to be tested. Each interface unit contains circuits to produce dynamic stimuli compatible with calculator operation as well as to measure timing, voltage levels and logic states under computer control. A relay

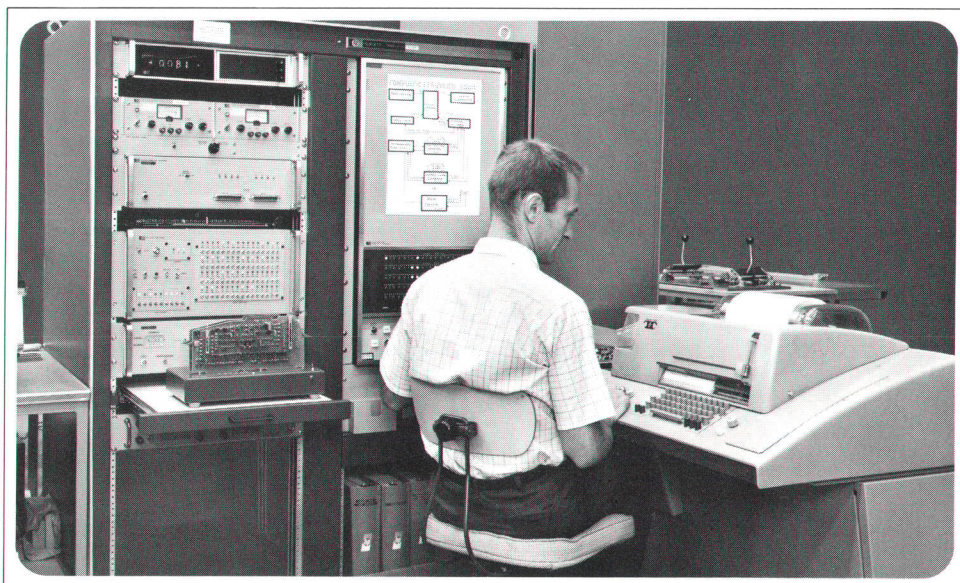


Fig. 1. Computer test station for the Model 9100A Calculator uses an HP Model 2116B computer, a teleprinter and test interface units for each logic assembly to be tested. Braid memory control logic is under test.

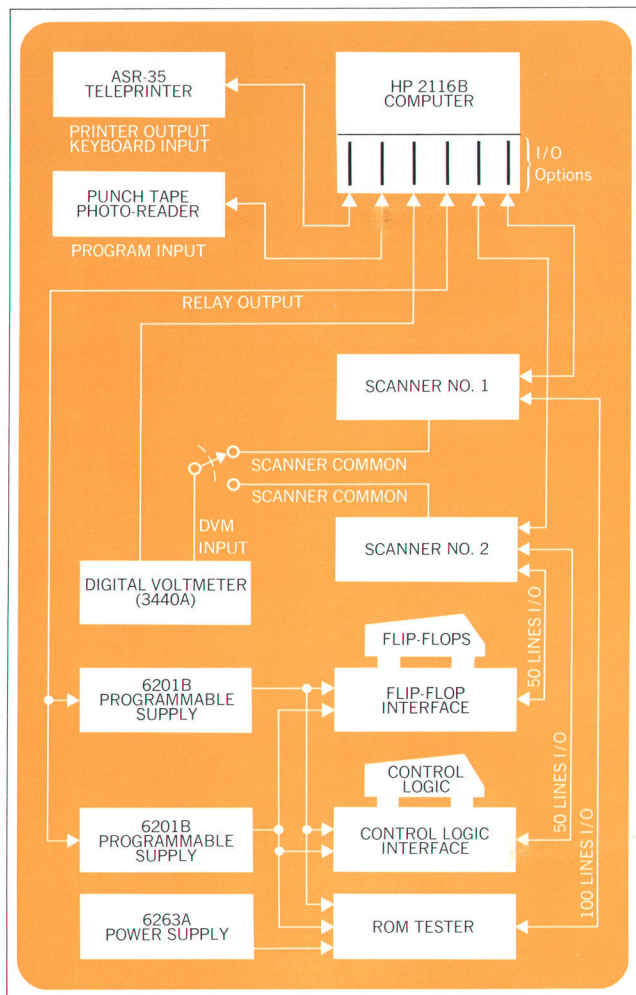


Fig. 2. System for pretesting logic subassemblies. Programmable power supplies check operating margins.

output register in the computer directly programs the power supplies, and the two general purpose registers provide input/output for the scanners and interface units.

Dynamic Testing

Response times of the calculator circuits are fully as important in testing as are the logic levels. For example, in testing the JK flip-flop a minimum width clock pulse is generated at a minimum time after shifting the input levels, thereby simulating a 'worst-case' operation. Similarly, a fixed duration forcing pulse applied to the output collector of each flip-flop detects any transistor that exceeds its stored base charge specification. The identifying numbers of the defective flip-flop and transistor are printed out together with the voltage and load at which failure occurred. If a collector voltage is outside of the programmed limits, the voltage will be printed out and again identified. The message from the teleprinter is placed with the tested board for repair or stockpiling and the next board is inserted. For the 2250 measurements made on a flip-flop board, test time is 30 to 40 seconds.

Woven Wire Memory

Testing of the Control Logic board requires verification of all bits stored in a 64 word by 29 bit toroid braid memory. Since the crystal-controlled clock is on this board, this clock is used in the tester interface, Fig. 2, to test the braid memory dynamically. As the word address is changed in a given clock time, the output to a test flip-flop is enabled on the next cycle, and the flip-flop state is read into computer memory. By switching the

<pre> CONTROL LOGIC TEST 31 MS TURN-ON SUPPLY VOLTAGES: -02.45 -16.48 62 ERRORS 00 SHUT-DOWN AT V +01.47 51 SHUT-DOWN AT V +00.81 RDM OUTPUT SHORTED 04 44 RCY 0 1 TMS 1 0 A 0 1 P 0 1 I 0 1 C60 1 0 C64 1 0 62 ERRORS </pre>	<pre> CONTROL LOGIC TEST 31 MS TURN-ON SUPPLY VOLTAGES: -02.45 -16.49 ** NO ERRORS ** -13.54 ** NO ERRORS ** 00 SHUT-DOWN AT V +01.47 51 SHUT-DOWN AT V +00.81 ** NO ERRORS ** </pre>
--	--

Fig. 3. Typical teleprinter output for a defective assembly (left), and printout for a perfect board (right).

scanner to each of the 29 output transistors and addressing the 64 states at each output, the 1856 bits of data are read into computer memory and compared against the correct data. If no errors are found at 10% high voltage, the sequence is repeated for 10% low voltage.

If errors are detected, the computer analyzes the error pattern to determine if an output transistor is shorted or open, if a word wire is open or exchanged, if the braid is incorrectly woven, or if various other errors exist. These diagnostics are printed as shown in the example, Fig. 3, which indicates two wires exchanged (address 04 and 44) and one output transistor shorted for a total of 62 bit errors. Also on this test the timing of the turn-on circuitry is measured and printed in milliseconds, as well as the voltage at which control is forced to the shut-down state. Typical time for the test and print-out is 50 to 60 seconds.

ROM Test by Comparison Method

The 32,768 bit main Read Only Memory is the most complex of the logic subassemblies and would require considerable input/output time to test by the method used for the braid memory. Thus a more sophisticated ROM tester interface, Fig. 4, has been built to compare the ROM assembly against a reference ROM at calculator speeds through 64 exclusive-OR gates. Each exclusive-OR output sets a corresponding buffer flip-flop, which in turn halts testing and flags the computer to input the error data. Thus the entire 32,768 bits are tested and the errors stored in milliseconds to seconds depending on the number of errors. If errors are detected at nominal operating conditions, diagnostics are performed much as with the braid memory test.

If no errors were found, the computer adjusts the sense amplifier bias in 0.25 volt increments until a specified number of failures occur or a $\pm 50\%$ to -50% bias margin has been tested. The computer then organizes this error data in table form showing failures as a function of bias voltage. Typically 600,000 to 1,200,000 bit comparisons are performed in a margin test in about two minutes. The margin tables aid in early detection of any process change which affects operating margins, and also serve to define an optimum bias value for each ROM assembly to assure greater reliability of operation.

A variation of the ROM tester is used to test the Read Only Memory data board before the drive circuits and sense amplifier circuits are soldered to it. This tester has drive and sense circuits that make pressure contact to the ROM data board and may be used with or without the aid of the computer.

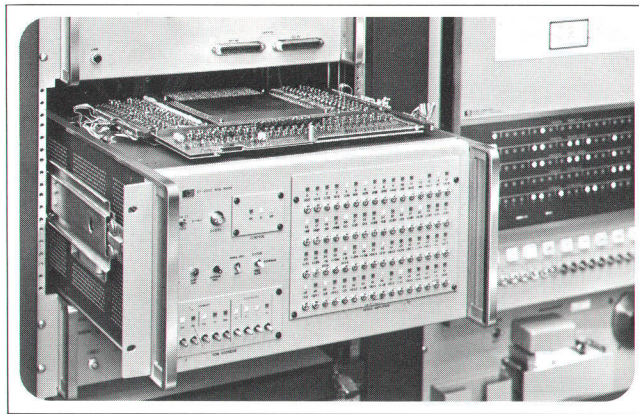


Fig. 4. Read-Only Memory tester interface compares the ROM assembly mounted on top with a reference unit.

The goal of minimizing total test time and increasing operating margins through use of an HP 2116B Computer test facility is fast becoming a reality. Based on data from early production runs, over 90% of fully assembled units operate perfectly at initial turn-on. By comparison, in subassembly testing about 50% of flip-flops, control logic and ROM subassemblies pass the computer test initially without the need for any further adjustment. For those problems that do arise in final test, trouble-shooting is still required, for the complexity at this level exceeds the diagnostic capabilities of the computer test facility.

Acknowledgments

I am pleased to acknowledge the contributions by Bob Watson who initially proposed use of the computer to test the 9100A Calculator, and by Jack Anderson who offered valuable suggestions for adaptation of the computer to production testing. Norm Carlson and John Scohy provided electronic tooling in construction of the test interface units; Rex James and Clair Nelson contributed to development of the read only memory tester.



Charles W. Near

Chuck Near received his B.S. degree in Electrical Engineering from Michigan State University in 1961. Upon joining Hewlett-Packard the same year, he helped design the HP Model 3440A Digital Voltmeter. In 1964 he earned his MSEE from Stanford University on the HP Honors Cooperative Program.

During development of the HP Model 9100A Calculator at HP Laboratories, Chuck had design

responsibility for the Calculator Read Only Memory, and has been issued a patent on this memory device. In July of 1967 Chuck was transferred from HP Labs to the Loveland Division where he designed the computer test facility for calculator production.

Chuck is a member of Tau Beta Pi, Eta Kappa Nu and IEEE.

How the Model 9100A Was Developed

By Bernard M. Oliver

Vice President for
Research & Development

Some lab projects are endothermic: the desired reaction proceeds only with the application of considerable heat and pressure and stops the moment these are relaxed. Others are exothermic: when the proper ingredients are brought together the reaction starts automatically and it is only necessary to harness and control the power that is generated. The 9100A project was one of the most exothermic I have known.

The ingredients started coming together in the late summer of 1965 when we were shown a prototype of a calculator invented by Malcolm McMillan that had one interesting feature: it could calculate all the common transcendental functions. The machine operated in fixed point and took a few seconds to calculate a function, but it did demonstrate the feasibility of providing these functions in a small calculator, and the power of the algorithm used to compute them.

The second ingredient was Tom Osborne who came to see us carrying a little green balsa wood calculator, which he had built on his own to demonstrate the virtue of some of his design concepts. What impressed us was its millisecond speed and its ten digit floating point operation and display.

A third ingredient was the imagination of Paul Stoft and other engineers in his group. It took no genius to see the appeal of a calculator that combined the speed and dynamic range of Tom's machine with the transcendental computing ability of the other machine. But to combine them into a small machine faster than either prototype, to adapt the transcendental algorithm to floating point, to add programmability and magnetic card storage and entry of programs, to provide the flexible display with automatic roundup, and to design the whole assembly for automated production required not only imagination but engineering skill of the highest order.

Another ingredient, the read only memory, which stores all the calculating and display routines, was already under development by Arndt Bergh and Chuck Near before we began the 9100A project. By carrying printed circuit techniques beyond the existing state of the art, Chuck was able to compress the required 32,000 bits into an amazingly small space. While the 9100A uses only discrete diodes and transistors, it is fair to describe the read only memory as one large integrated circuit with extremely long life expectancy.

As soon as the development began, everyone, it seemed, had ideas for new features. Hardly a day went by without someone proposing a new keyboard with a new key arrangement or new functions. It was Dick Monnier's responsibility as section leader to steer the project through these conflicting currents of ideas. Although we went down a couple of blind alleys, for the most part the course was held true. That we arrived at such an elegant solution in a short time is a tribute to Dick's navigation.

Tom Osborne joined the development team as a consultant on the general architecture of the machine. His contributions include the basic logical design, and the details of the control logic, flip-flops, mother board gates, and the memory drive and sense circuitry. He also contributed a large measure of sound judgment that resulted in an economy of design, low power consumption, high performance, and the ability of the 9100A to interface with peripherals and systems.

Tony Lukes developed the display routines, which include the features of automatic roundup in fixed point display, choice of decimal places, suppression of insignificant zeros, and the display of program step addresses and key codes, as well as numerical data. Tony also developed the program storage, editing, and execution routines which, together with the display, make the 9100A easy to use and program.

I should at this point emphasize that the 9100A, while small, is, in a sense, much more complicated than many general purpose computers. Most general purpose computers have relatively large memories but can execute directly only certain elementary machine instructions. To compute complicated functions and indeed, in some cases, even to perform simple arithmetic, the computer must be externally programmed.

By contrast the 9100A has a very sophisticated external instruction set: the entire keyboard. The 9100A is a small computer with a large amount of 'software' built in as hardware.

The task of compressing the floating point arithmetic operations and functional computations into the limited read only memory of the 9100A was accomplished by Dave Cochran. To make sure that all the calculations were accurate over the enormous range of arguments allowed by the floating point operation, to assure exact values at certain cardinal points, and above all to get so much in so little memory space was an enormous achievement.

I sometimes wonder if Dave realizes what a remarkable job he did. It took several passes. On the first pass it appeared hopeless to include all the functions. But by nesting routines and by inventing a number of space saving tricks he was able to save enough states to crowd them all in. Then various bugs were discovered and more states had to be freed to correct these. The 'battle of the states' continued for several months and the end result was one of the most efficient encoding jobs ever done.

The necessity for magnetic card program storage and entry became apparent as soon as we had an operating prototype. Don Miller, Dick Osgood and Bob Schweizer deserve praise for the speedy development of this unit, which adds so much to the convenience of the 9100A.

Clarence Studley supervised the overall mechanical design and assembly, reducing to manufacturing drawings and to the final metal the handsome cabinet styling of Roy Ozaki, Don Aupperle and others in the Industrial Design group, while Harold Rocklitz and Doug Wright handled much of the tooling.

Many other people contributed to the 9100A—too many to give proper credit to all; but I must mention the fine art work of Frank Lee on the read only memory and other printed circuit boards, Chung Tung's work on the core memory electronics and Bill Kruger's development of the short high brightness cathode ray tube. Chris Clare made many contributions to the project especially in the area of interfacing the calculator with printers and other peripherals. A special measure of recognition is due Ken Petersen whose expert technicianship and whose genius at trouble shooting saved us weeks of time and bailed us out of many tight spots. Ken also laid out the multilayer mother board with its thousand diode gates and interconnections.

The transfer of the 9100A from Hewlett-Packard Laboratories to the Loveland Division took place gradually rather than abruptly. As various portions reached the final prototype stages, responsibility for these was assumed by the Loveland group headed by Bob Watson in engineering and by Jack Anderson in production. Many visits both ways and some transfer of people to Loveland accomplished the transmission of much unwritten information. The Loveland team introduced several engineering improvements. Especially significant were the improved read only memory margins obtained by Rex James and the many contributions by Ed Olander, whose comprehensive understanding of the entire machine helped greatly. That we were able to go from an incomplete lab prototype stage in Palo Alto to a pilot run of final instruments in Loveland in only 10 months attests to the skill and dedication of the Loveland group and to the fine cooperation on both sides.

Finally, I must confess that very few projects receive as much direct attention from corporate management as this one did. Early in the spring of 1967 a skiing injury landed Bill Hewlett in the hospital. We learned about this right away when he called up to have some 9100A programming pads sent over. I found myself hypnotized by the project and unable to share my time equitably. Here was management in the unusual role of consumer, for if Bill and I did anything constructive it was mainly to assess and modify the developing product from the user's standpoint. I owe the 9100A group an apology for being constantly in their hair, and everyone else in HP Labs an apology for slighting their projects. Now that it's all over I find the 9100A as fascinating to use as it was to develop. *Caveat emptor!*

HP Archive

This vintage Hewlett-Packard document was
preserved and distributed by

www.hparchive.com

Please visit us on the web!

On-line curator: John Miles, KE5FX

jmiles@pop.net

