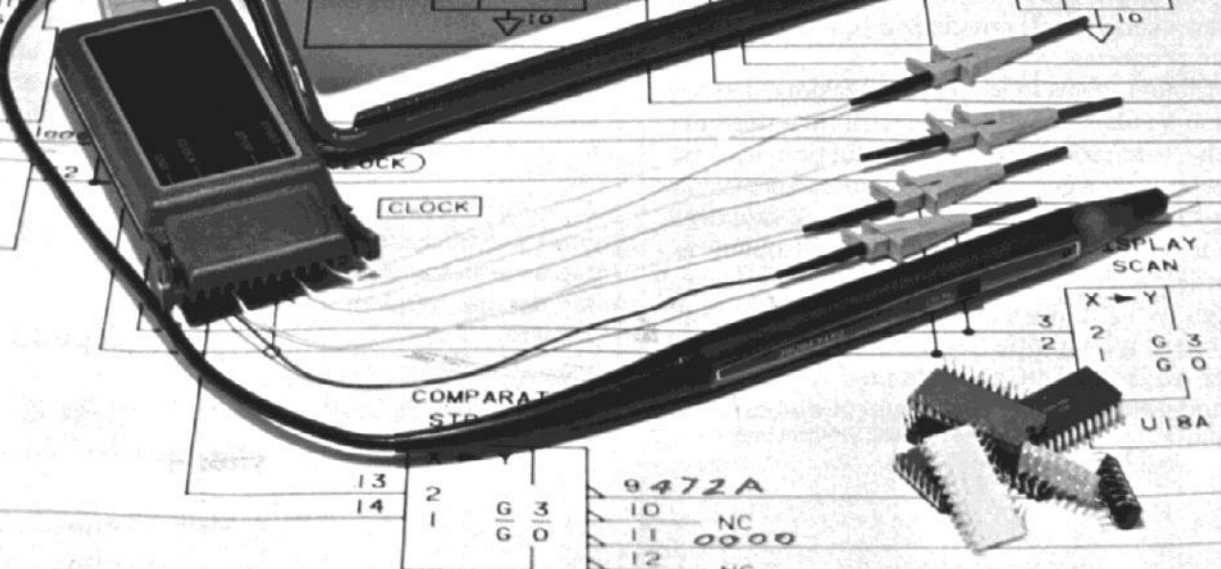
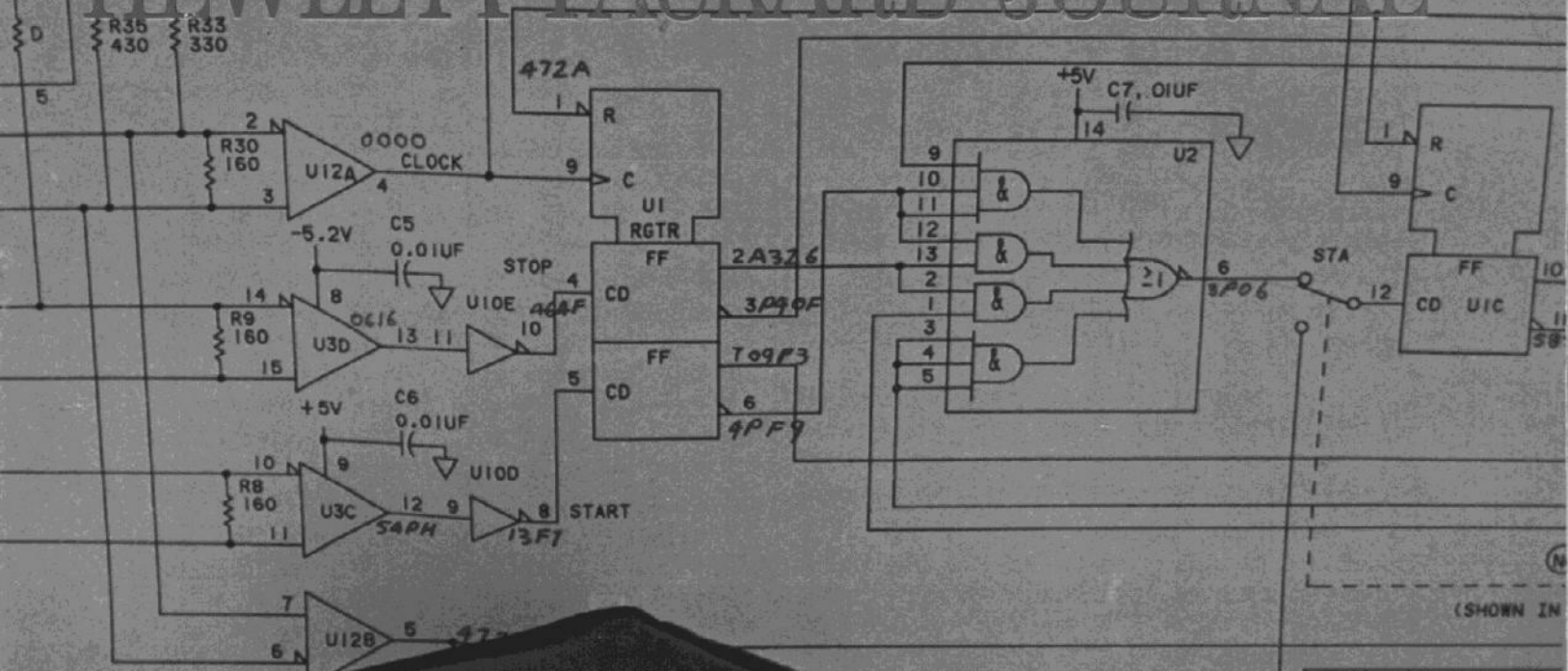


HEWLETT-PACKARD JOURNAL



X	Y
4	4C78
5	525CF
6	67661
7	7508U

Signature Analysis: A New Digital Field Service Method

In a digital instrument designed for troubleshooting by signature analysis, this method can find the components responsible for well over 99% of all failures, even intermittent ones, without removing circuit boards from the instrument.

by Robert A. Frohwerk

WITH THE ADVENT OF MICROPROCESSORS and highly complex LSI (large-scale integrated) circuits, the engineer troubleshooting digital systems finds himself dealing more with long digital data patterns than with waveforms. As packaging density increases and the use of more LSI circuits leaves fewer test points available, the data streams at the available test points can become very complex. The problem is how to apply some suitable stimulus to the circuit and analyze the resulting data patterns to locate the faulty component so that it can be replaced and the circuit board returned to service.

The search for an optimal troubleshooting algorithm to find failing components on digital circuit boards has taken many directions, but all of the approaches tried have had at least one shortcoming. Some simply do not test a realistic set of input conditions, while others perform well at detecting logical errors and stuck nodes but fail to detect timing-related problems. Test systems capable of detecting one-half to two-thirds of all possible errors occurring in a circuit have been considered quite good. These systems tend to be large, for factory-based use only, and computer-driven, requiring program support and software packets and hardware interfaces for each type of board to be tested. Field troubleshooting, beyond the logic-probe capability to detect stuck nodes, has been virtually neglected in favor of board exchange programs.

The problem seems to be that test systems have too often been an afterthought. The instrument designer leaves the test procedure to a production test engineer, who seeks a general-purpose solution because he lacks the time to handle each case individually.

Obviously it would be better if the instrument designer provided for field troubleshooting in his original design. Who knows a circuit better than its original designer? Who has the greatest insight as to how to test it? And what better time to modify a circuit to accommodate easy testing than before the circuit is in production?

New Tools Needed

But here another problem arises: what do we offer the circuit designer for tools? A truly portable test instrument, since field troubleshooting is our goal, would be a passive device that merely looked at a circuit and told us why it was failing. The tool would provide no stimulus, require little software support, and have accuracy at least as great as that of computer-driven factory-based test systems.



Cover: Those strange-looking strings of four alphanumeric characters on the instrument's display and the schematic diagram are signatures, and the instrument is the 5004A Signature Analyzer, a troubleshooting tool for field repair of digital systems.

With a failing system operating in a self-stimulating test mode, the service person probes various test points, looking for incorrect signature displays that can point to faulty components.

In this Issue:

- Signature Analysis: A New Digital Field Service Method*, by Robert A. Frohwerk **page 2**
- Easy-to-Use Signature Analyzer Accurately Troubleshoots Complex Logic Circuits*, by Anthony Y. Chan .. **page 9**
- Signature Analysis—Concepts, Examples, and Guidelines*, by Hans J. Nadig **page 15**
- Personal Calculator Algorithms I: Square Roots*, by William E. Egbert .. **page 22**

Personal Calculator Algorithms I: Square Roots

A detailed description of the algorithm used in Hewlett-Packard hand-held calculators to compute square roots.

by William E. Egbert

BEGINNING WITH THE HP-35,^{1,2} all HP personal calculators have used essentially the same algorithms for computing complex mathematical functions in their BCD (binary-coded decimal) microprocessors. While improvements have been made in newer calculators,³ the changes have affected primarily special cases and not the fundamental algorithms.

This article is the first of a series that examines these algorithms and their implementation. Each article will present in detail the methods used to implement a common mathematical function. For simplicity, rigorous proofs will not be given, and special cases other than those of particular interest will be omitted.

Although tailored for efficiency within the environment of a special-purpose BCD microprocessor, the basic mathematical equations and the techniques used to transform and implement them are applicable to a wide range of computing problems and devices.

The Square Root Algorithm

This article will discuss the algorithm and methods used to implement the square root function.

The core of the square root algorithm is a simple approximation technique tailored to be efficient using the instruction set of a BCD processor. The technique is as follows:

\sqrt{x} is desired

1. Guess an answer a
2. Generate a^2
3. Find $R = x - a^2$
4. If the magnitude of R is sufficiently small, $a = \sqrt{x}$.
5. If R is a positive number, a is too small.
If R is a negative number, a is too big.
6. Depending on the result of step 5, modify a and return to step 2.

The magnitude of R will progressively decrease until the desired accuracy is reached.

This procedure is only a rough outline of the actual square root routine used. The first refinement is to avoid having to find a^2 and $x - a^2$ each time a is changed. This is done by finding a one decade at a time. In other words, find the hundreds digit of a , then the tens digit, the units digit, and so on. Once

the hundreds digit is found, it is squared and subtracted from x , and the tens digit is found. This process, however, is not exactly straightforward, so some algebra is in order.

The following definitions will be used:

x = the number whose square root is desired

a = most significant digit(s) of \sqrt{x} previously computed

b = the next digit of \sqrt{x} to be found

j = the power of 10 associated with b

$R_a = x - a^2$, the current remainder

a_j = the new a when digit b is added in its proper place. $a_j = a + (b \times 10^j)$ (1)

R_b = the portion of remainder R_a that would be removed by adding b to a . $R_b = a_j^2 - a^2$ (2)

For example, let $x = 54756$. Then $\sqrt{x} = 234$.

Let $a = 200$.

$b =$ the digit we are seeking (3, in this case)

$j = 1$ (the 10's digit is being computed)

$R_a = 54756 - (200)^2 = 14756$.

Note that a_j and R_b will vary with the choice of b .

The process of finding \sqrt{x} one decade at a time approaches the value of \sqrt{x} from below. That is, at any point in the computation, $a \leq \sqrt{x}$. Consequently, $R_a \geq 0$.

With this in mind it is easy to see that for any decade j , the value of b is the largest possible digit so that

$$R_a - R_b \geq 0$$

or

$$R_b \leq R_a. \quad (3)$$

Using equations 1 and 2 we have

$$R_b = [a + (b \times 10^j)]^2 - a^2.$$

Expanding and simplifying,

$$R_b = 2ab \times 10^j + (b \times 10^j)^2. \quad (4)$$

Inserting (4) into (3) yields the following rule for finding digit b .

Digit b is the largest possible digit so that

$$2ab \times 10^j + (b \times 10^j)^2 \leq R_a \quad (5)$$

When the digit that satisfies equation 5 is found, a new a is formed by adding $b \times 10^j$ to the old a , the decade counter (j) is decremented by 1, and a new R_a is created; the new R_a is the old R_a minus R_b .

Continuing the previous example,

$$\begin{aligned} x &= 54756 \\ j &= 1 \\ a &= 200 \\ x - a^2 &= R_a = 14756 \end{aligned}$$

Applying equation 5 to find b:

b	$R_b = 2ab \times 10^j + (b \times 10^j)^2$	$R_a - R_b$
0	0	14756
1	4100	10656
2	8400	6356
3	12900	1856
4	17600	-2844

Thus $b=3$, since $b=4$ causes overdraft, i.e., $R_a - R_b < 0$. The new $a = 200 + 3 \times 10^1 = 230$. The new $R_a = 1856$, the new $j=0$. With these new parameters, the units digit can be found.

This process may seem vaguely familiar, which is not surprising since upon close inspection it turns out to be the (usually forgotten) scheme taught in grade school to find square roots longhand. Of course, trailing zeros and digits are not written in the long-hand scheme.

To make this process efficient for a calculator, still another refinement is needed.

$(b \times 10^j)^2$ can be expressed as a series, using the fact that the square of an integer b is equal to the sum of the first b odd integers. Thus,

$$\begin{aligned} (b \times 10^j)^2 &= b^2 \times 10^{2j} \\ &= \sum_{i=1}^b (2i-1) \times 10^{2j} \end{aligned}$$

For example,

$$\begin{aligned} (3 \times 10^1)^2 &= 1 \times 10^{2j} + 3 \times 10^{2j} + 5 \times 10^{2j} \\ &= 9 \times 10^{2j} \end{aligned}$$

Thus $2ab \times 10^j + (b \times 10^j)^2$ can be expressed as:

$$2ab \times 10^j + (b \times 10^j)^2 = \sum_{i=1}^b 2a \times 10^j + (2i-1) \times 10^{2j}$$

or

$$R_b = \sum_{i=1}^b 2a \times 10^j + (2i-1) \times 10^{2j} \quad (6)$$

Now comes a key transformation in the square root routine. It was shown earlier how inequality 3 will

give the value b for the next digit of a . Since multiplying both sides of an inequality by a positive constant does not change the inequality, equations 3 and 6 can be multiplied by the number 5.

$$5R_b \leq 5R_a$$

$$5R_b = \sum_{i=1}^b 10a \times 10^j + (10i-5) \times 10^{2j} \quad (7)$$

b becomes the largest digit so that $5R_b \leq 5R_a$. The new $5R_a$ is equal to the old $5R_a$ minus $5R_b$.

These transformations may seem useless until we examine a few examples of the last term of the right side of (7) for various values of b .

$$\begin{aligned} 10a \times 10^j + 05 \times 10^{2j}, b=1 \\ 10a \times 10^j + 15 \times 10^{2j}, b=2 \\ 10a \times 10^j + 25 \times 10^{2j}, b=3 \end{aligned}$$

Notice that the two-digit coefficient of 10^{2j} consists of $(b-1)$ and a 5. These two digits will be expressed as $(b-1) \mid 5$ in succeeding equations. $10a$ is formed by a simple right shift and does not change between terms. If the sum defined in equation 7, as b is incremented by 1, is subtracted from $5R_a$ until overdraft occurs, the digit in the next-to-last digit position is b . Best of all, it is in the exact position to form the next digit of a without further manipulation. Redoing the previous example may help clarify matters.

$$\begin{aligned} R_a &= 14756 \\ j &= 1 \\ a &= 200 \\ 5R_a &= 73780 \end{aligned}$$

b	$10a \times 10^j + (b-1) \mid 5 \times 10^{2j}$	$5R_a - 5R_b$
1	20500	53280
2	21500	31780
3	22500	9280 new $5R_a$
4	23500	-14220 overdraft

new value of a
digits

Notice that when overdraft occurs the new value of a is already created and the new value of $5R_a$ can be found by restoring the previous remainder.

Decrementing the value of j would cause, in effect, $(10a \times 10^j)$ to shift right one place, and $(b-1) \mid 5 \times 10^{2j}$ to shift right two places. The result is that the final 5 shifts one place to the right to make room for a new digit. Continuing with the same example,

$$\begin{aligned} 5R_a &= 9280 \\ a &= 230 \\ j &= 0 \end{aligned}$$

b	$10a \times 10^{1+(b-1)} 5 \times 10^{2^b}$	$5R_a - 5R_b$
1	2305	6975
2	2315	4660
3	2325	2335
4	2335	0 remainder
5	2345	-2345 overdraft

final $a = \sqrt{x}$


For ease of understanding, the preceding example treated a large positive number. A number in the calculator actually consists of a mantissa between 1 and 10 and an exponent. The problem is to find the square root of both parts of this argument. Happily, if the input exponent is an even number, the portion of the answer resulting from it turns out to be the exponent of the final answer and is simply the input exponent divided by 2. Thus to find \sqrt{x} , the exponent of x is first made even and the mantissa shifted to keep the number the same. The exponent of \sqrt{x} is found by dividing the corrected input exponent by 2. The method described above is then used to find the square root of the shifted input mantissa, which (after possibly being shifted) can be between 1 and 100. The result will then be between 1 and 10, which is the range required for the mantissa of \sqrt{x} .

During the process of finding \sqrt{x} the remainder R_a progressively decreases. To avoid losing accuracy, this remainder is multiplied by 10^1 after finding each new digit b . This avoids shifting a at all, once the square root extraction process begins. A 12-digit mantissa is generated, which insures accuracy to ± 1 in the tenth digit of the mantissa of \sqrt{x} .

In summary, the computation of \sqrt{x} proceeds as follows:

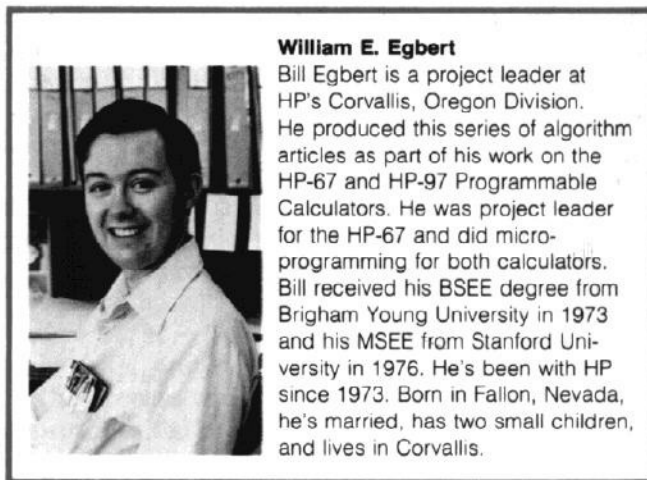
1. Generate exponent of answer.
2. Multiply mantissa by 5 to create original $5R_a$

3. With an original a of 0, use the method described above to find 12 b digits to form the mantissa of the answer.
4. Round the mantissa and attach the exponent found previously.
5. Display the answer.

The calculator is now ready for another operation. 

References

1. T.M. Whitney, F. Rodé, and C.C. Tung, "The 'Powerful Pocketful': An Electronic Calculator Challenges the Slide Rule," Hewlett-Packard Journal, June 1972.
2. D.S. Cochran, "Algorithms and Accuracy in the HP-35," Hewlett-Packard Journal, June 1972.
3. D.W. Harms, "The New Accuracy: Making $2^3=8$," Hewlett-Packard Journal, November 1976.



William E. Egbert

Bill Egbert is a project leader at HP's Corvallis, Oregon Division. He produced this series of algorithm articles as part of his work on the HP-67 and HP-97 Programmable Calculators. He was project leader for the HP-67 and did micro-programming for both calculators. Bill received his BSEE degree from Brigham Young University in 1973 and his MSEE from Stanford University in 1976. He's been with HP since 1973. Born in Fallon, Nevada, he's married, has two small children, and lives in Corvallis.

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

HEWLETT-PACKARD JOURNAL

MAY 1977 Volume 28 • Number 9

Technical information from the Laboratories of
Hewlett-Packard Company

Hewlett-Packard Central Mailing Department
Van Heuven Goedhartaan 121
Amstelveen-1134 The Netherlands
Yokogawa Hewlett-Packard Ltd., Shibuya-Ku
Tokyo 151 Japan

Editorial Director • Howard L. Roberts
Managing Editor • Richard P. Dolan
Art Director, Photographer • Arvid A. Danielson
Illustrator • Susan E. Wright

Administrative Services, Typography • Anne S. LoPresti
European Production Manager • Dick Leekma

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

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