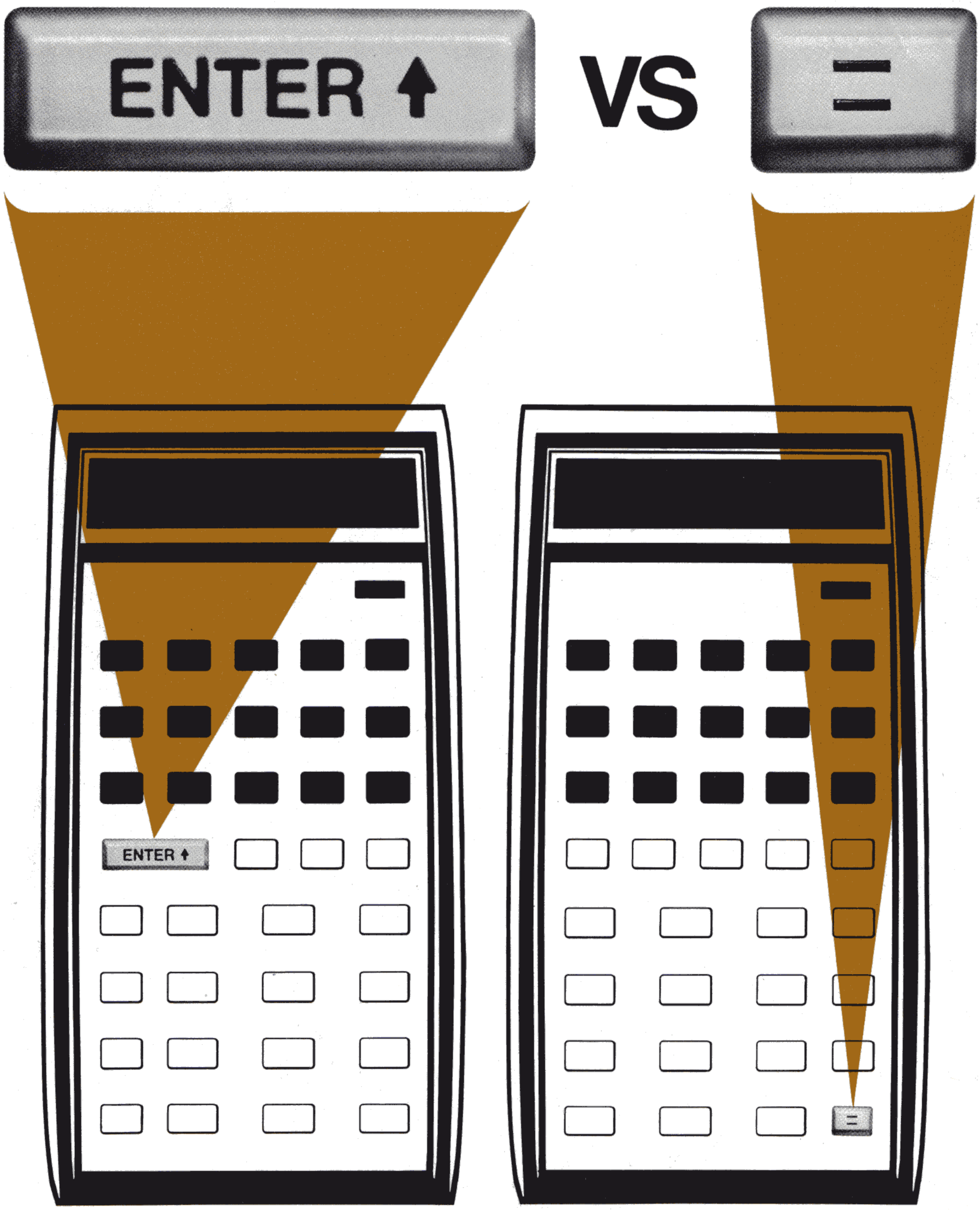


A practical, problem-by-problem discussion of the logic systems used in today's pocket-sized scientific calculators.

Scientific pocket calculators-the inside story.

Even at today's lower prices, a scientific pocket calculator can be a significant investment. So before you buy, you should consider the facts presented in this booklet.

If you're about to spend more than a hundred dollars for a pocket-sized scientific calculator, you owe it to yourself to choose carefully. And, you should know that a calculator's external appearance often reveals little about its true calculating power.

To help you solve complex — "real-world" — scientific problems, Hewlett-Packard believes a calculator should give you more than the log, trig and exponential functions apparent from the keyboard. It should give you storage registers to hold constants and intermediate answers. It should give you a logic system that's simple, consistent and practical to use.

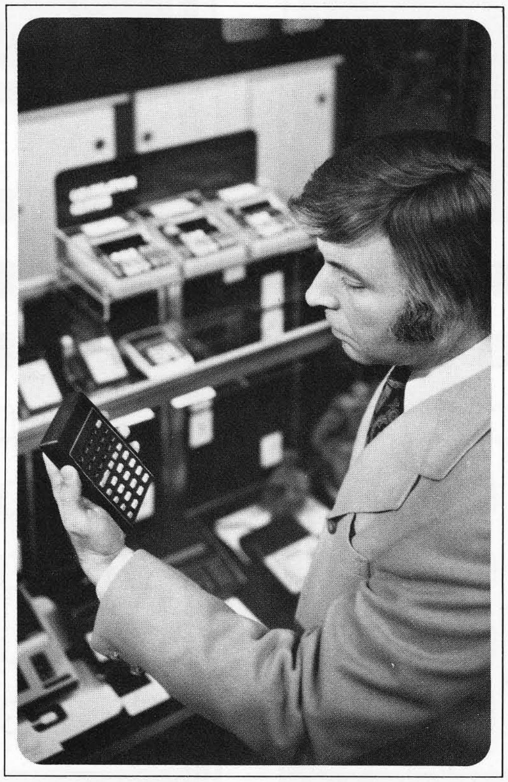
Most important, it should give you confidence. The confidence that comes from knowing that every problem can be approached in the same, consistent way; that you can trust every answer you get.

Three calculator logic systems: a comparison

On the following pages we'll demonstrate that calculators which look alike often do not work alike, using five sample problems. Why? So that you can compare the simplicity, power and convenience of the three different logic systems used in today's scientific pocket calculators:

1. Reverse Polish Notation (*RPN*) with 4-register operational memory stack and a minimum of one addressable storage register. This is the system used in all Hewlett-Packard pocket-sized calculators.
2. Modified algebraic notation without parenthesis key but with operational hierarchy, three internal working registers and one addressable storage register. We'll call this "System A."
3. Modified algebraic notation without parenthesis key or operational hierarchy but with two internal working registers and one addressable storage register. We'll call this "System B."

We think you'll want to understand the differences between these systems before deciding which scientific pocket calculator meets your needs. And, we hope that you'll agree with us that RPN offers the most efficient, most consistent way to evaluate complex problems on a pocket-sized calculating device.



Today's scientific pocket calculators use one of three different logic systems: Reverse Polish Notation, modified algebraic notation with operational hierarchy, and modified algebraic notation without hierarchy. Before buying any scientific pocket calculator, you should know the differences between these systems and determine the best for your calculating needs.

RPN-the only language that lets you "speak" with confidence and consistency to a pocket-sized computer calculator.

In 1967, Hewlett-Packard embarked on a major new development effort: to design a family of advanced computer calculators powerful enough to solve complex engineering/scientific problems yet simple enough to be used by anyone who works with numbers.

As part of this effort, HP carefully evaluated the strengths and weaknesses of the various languages which an operator might use to communicate with an electronic calculating device. Among those studied were:

- computer languages such as BASIC and FORTRAN,
- various forms of algebraic notation, and
- RPN (*Reverse Polish Notation*), a parenthesis-free but unambiguous language derived from that developed by the Polish mathematician, Jan Lukasiewicz.

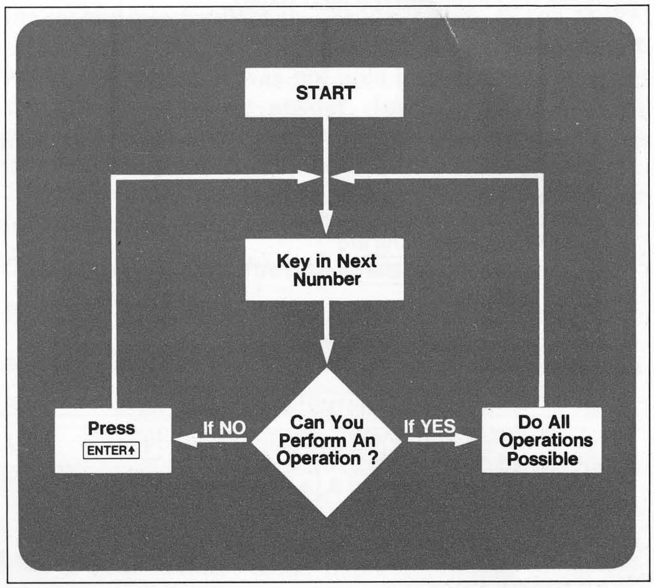
As might be expected, each of these languages was found to excel in a particular application. For its biggest programmable desktop calculators, HP selected BASIC. For its other powerful desktop calculators, with less extensive storage capacity, HP chose algebraic notation.

But, given the design constraints of a pocket-sized scientific computer calculator, RPN proved the simplest, most efficient, most consistent way to solve complex mathematical problems.

Only RPN offers these powerful advantages

Compared to alternative logic systems, Hewlett-Packard believes that only RPN — in combination with a 4-register operational memory stack — gives you these powerful advantages.

1. You can always enter your data the same way, i.e., from left to right—the same way you read an equation. Yet, there is no need for a parenthesis key; nor for a complicated "operational hierarchy."
2. You can always proceed through your problem the same way. Once you've entered a number, you ask: "Can I perform an operation?" If yes, you do it. If no, you press ENTER↑ and key in the next number.
3. You always see all intermediate answers — as they are calculated — so that you can check the progress of your calculation as you go. As important, you can review all numbers stored in the calculator at any time by pressing a few keys. There is no "hidden" data.
4. You don't have to think your problem all the way through beforehand unless the problem is so complex that it may require simultaneous storage of three or more intermediate answers.
5. You can easily recover from errors since all operations are performed sequentially, immediately after pressing the appropriate key.



The RPN method consists of four, easy-to-remember steps. Once learned, it can be applied to almost any mathematical expression.

6. You don't have to write down and re-enter intermediate answers, a real time-saver when working with numbers of eight or nine digits each.
7. You can communicate with your calculator confidently, consistently because you can always proceed the same way.

If all this sounds too good to be true, bear with us — you'll soon get the chance to see for yourself. But first, we need to describe how RPN and the 4-register operational stack operate.

The RPN method — it takes a few minutes to learn but can save years of frustration.

Yes, the RPN method does take some getting used to. But, once you've learned it, you can use the RPN method to solve almost any mathematical expression — confidently, consistently.

There are only four easy-to-follow steps:

1. Starting at the left side of the problem, key in the first or next number.
2. Determine if any operations can be performed. If so, do all operations possible.
3. If not, press ENTER↑ to save the number for future use.
4. Repeat steps 1 through 3 until your calculation is completed.

A diagram of the RPN method is shown above.

Simple arithmetic, the RPN way.

Just to show how it works, let's try the RPN method on two simple problems (we'll use them again in the comparisons that begin on the next page).

Problem: $3 \times 4 = 12$

RPN solution:

Step	Press	See Displayed
1. Key in first number.	[3]	3
2. Since only one number has been keyed in, no operations are possible. Press [ENTER↑].	[ENTER↑]	3
3. Key in next number.	[4]	4
4. Since both numbers are now in calculator, multiplication can be performed.	[×]	12

Problem: $(3 \times 4) + (5 \times 6) = 42$

RPN solution:

Step	Press	See Displayed
1. Key in first number.	[3]	3
2. No operations possible. Press [ENTER↑].	[ENTER↑]	3
3. Key in second number.	[4]	4
4. Since both numbers are in calculator, first multiplication is possible.	[×]	12
5. Key in next number. (*First intermediate answer will be automatically stored for future use.*)	[5]	5
6. No operations possible. Press [ENTER↑].	[ENTER↑]	5
7. Key in next number.	[6]	6
8. Second multiplication is possible since both numbers are in calculator.	[×]	30
9. Addition is possible since both intermediate answers have been calculated and are stored in 4-register operational stack.	[+]	42

If you've followed us this far, you've noticed two important facts:

1. Both of these problems were solved in the same, consistent manner, using the same simple set of rules.
2. All intermediate answers were displayed as they were calculated, and stored and retrieved as needed to complete the calculation. With RPN and a 4-register operational memory stack, there is almost never a need to write down intermediate answers.

How the operational stack works.

The four registers of HP's exclusive operational stack can be represented by the following diagram.

T		**Top**
Z		
Y		
X		**Display**

When a number is keyed in, it goes into the X register for display. Pressing the [ENTER↑] key duplicates the contents of the X register into the Y register and moves all other numbers in the stack up one position.

When an operation key ([+], [−], [×], [÷], [x^y]) is pressed the operation is performed on the numbers in the X and Y registers, and the answer appears in the X register for display. Numbers in the other registers automatically drop one position.

To demonstrate these points, we'll show what happens to the stack as we solve the problem: $(3 \times 4) + (5 \times 6) = 42$.

T									
Z						12	12		
Y		3	3		12	5	5	12	
X	3	3	4	12	5	5	6	30	42
	[3]	[ENTER↑]	[4]	[×]	[5]	[ENTER↑]	[6]	[×]	[+]

As you can see, all numbers are automatically positioned in the stack on a last-in-first-out basis, in the proper order for subsequent use.

Now that we've described how RPN logic operates, we can proceed with our problem-by-problem comparison of this system versus two others used in today's scientific pocket calculators.

We think you will find it interesting.

RPN vs. algebraic – practical comparison using three representative calculators and "real-world" scientific problems.

As we pointed out earlier in this booklet, calculators that look alike don't necessarily work alike.

How, then, do you decide which scientific pocket calculator is best suited for your needs? We propose these criteria and suggest that you keep them in mind as you read through the following pages:

Criteria for evaluating a scientific pocket calculator.

1. You are buying a scientific pocket calculator because you work with complex scientific problems. Therefore, the calculator itself shouldn't add complexity.
2. You should select the calculator that gives you the most confidence—confidence that you can trust the answers you get.
3. You should select a calculator that can solve complex equations according to a consistent, easily remembered set of rules.
4. And, you should select a calculator that provides the features and functions you *really* need to solve *your* kinds of problems.

Comparison 1: simple arithmetic.

Let's start our comparisons with the same simple arithmetic problem we used before.

Problem: $3 \times 4 = 12$

System	Solution	No. of Keystrokes
RPN:	3 [ENTER↑] 4 [×]	4
A:	3 [×] 4 [=]	4
B:	3 [×] 4 [=]	4

As we have already seen, the RPN solution is based on a few simple rules. But, the fact that you can solve this problem on an algebraic machine just the way it's written offers a distinct advantage—one that explains why most pocket calculators use algebraic logic.

Chances are, though, that you wouldn't be looking at a *scientific* pocket calculator if you only needed to solve simple arithmetic problems. So, let's look at something a bit more complicated.

Comparison 2: sum-of-products.

This second problem is also one of the ones we used to explain the RPN method. But, let's see how calculators using the other two logic systems would solve it.

Problem: $(3 \times 4) + (5 \times 6) = 42$

System	Solution	No. of Keystrokes
RPN:	3 [ENTER↑] 4 [×] 5 [ENTER↑] 6 [×] [+]	9
A:	3 [×] 4 [+] 5 [×] 6 [=]	8
B:	3 [×] 4 [=] [STO] 5 [×] 6 [+] [RCL] [=]	11

The System A keystroke sequence is easy to remember since, as in the first problem, it follows the algebraic equation. Parentheses are not necessary because the operational hierarchy performs the second multiplication before the addition. It is worth noting, however, that the second intermediate answer is not displayed.

Because the System B calculator has only two internal working registers, it was not able to automatically store the first intermediate answer. Instead, the user must manually store and recall it (at the appropriate times) from the calculator's only addressable memory register. This, of course, means that you cannot use the register for storage of constants or other data.

Comparison 3: product-of-sums.

Problem: $(3 + 4) \times (5 + 6) = 77$

System	Solution	No. of Keystrokes
RPN:	3 [ENTER↑] 4 [+] 5 [ENTER↑] 6 [+] [×]	9
A:	3 [+] 4 [=] [STO] 5 [+] 6 [=] [×] [RCL] [=]	12
B:	3 [+] 4 [=] [STO] 5 [+] 6 [×] [RCL] [=]	11

Although this problem is very similar to the previous one, it demonstrates some important points about the advantages of RPN over modified algebraic systems.

1. With RPN, both problems are solved in the same consistent way. The only difference is that the [+] and [×] operations are reversed—just as in algebraic equations.
   In both problems, the 4-register operational memory stack automatically saves and retrieves the intermediate answers. And, both intermediate answers are displayed as calculated so that you can check the progress of your calculation as you go.
2. With System A, there are significant differences between the solutions of these two very similar problems. This is because the effects of the operational hierarchy must be carefully considered before you key in your problem.
   If you forget that multiplication is performed before addition, you might key in the product-of-sums problem just as written: 3 [+] 4 [×] 5 [+] 6 [=]. And, you would get an incorrect answer because the operational hierarchy would interpret the problem as: $3 + (4 \times 5) + 6 = 29$.
   Another approach to this problem might be: 3 [+] 4 [STO] 5 [+] 6 [×] [RCL] [=]. This would also give you a wrong answer because the operational hierarchy would now interpret the problem as: $5 + [6 \times (3 + 4)] = 47$.
3. With System B, both problems are at least approached in the same, consistent manner. But, as we will see in the following comparisons, this calculator's limited storage capacity severely restricts its ability to solve complex problems unless the user writes down and re-enters intermediate answers.

Comparison 4: mixed calculation.

Problem: $\text{LOG}\,[(4 \times 5) + (29 \div 3)] \times \{[19 \div (2 + 4)] + [(13 + \pi) \div 4]\} = 10.60337500$

System	Solution	No. of Keystrokes*
RPN:	4 [ENTER↑] 5 [×] 29 [ENTER↑] 3 [÷] [+] [log] 19 [ENTER↑] 2 [ENTER↑] 4 [+] [÷] 13 [ENTER↑] [π] [+] 4 [÷] [+] [×]	28
A:	2 [+] 4 [=] [1/x] [×] 19 [=] [STO] 13 [+] [π] [=] [÷] 4 [+] [RCL] [=] [STO] 4 [×] 5 [+] 29 [÷] 3 [=] [log] [×] [RCL] [=]	34
B:	4 [×] 5 [=] [STO] 29 [÷] 3 [+] [RCL] [=] [log] Write down intermediate answer 2 [+] 4 [=] [1/x] [×] 19 [STO] 13 [+] [π] [÷] 4 [+] [RCL] [×] re-enter intermediate answer [=]	32 †

*On some calculators, including the HP-45 and HP-65 models, a prefix key is required to activate some transcendental functions. Such prefix keystrokes have not been shown in this booklet.
†Does not include keystrokes required for re-entry of intermediate answer.

Although this mixed-calculation problem is more complex than our previous examples, it is also more typical of the kinds of problems you are apt to encounter in "real-world" situations.

With RPN, you can approach this problem exactly the same way we approached the previous problems:

1. Starting at the left side of the equation, key in the first or next number.
2. Determine if any operations can be performed. If so, do all operations possible.
3. If not, press [ENTER↑] to save the number for future use.
4. Repeat steps 1 through 3 until your calculation is complete.

Thus, on a calculator with RPN logic you key in all data in the same order as it appears in the problem to be solved. And, as before, the operational memory stack automatically keeps track of the intermediate answers. However, in this problem, all four registers are required.

With System A, we had to proceed as follows:

1. Keeping the operational hierarchy rules in mind (as well as the calculator's storage capacity) think through the entire problem to determine the best method of approach.
2. Mentally restructure the problem so that it conforms to machine logic.
3. Decide which intermediate answers must be manually stored in and recalled from the addressable memory register.
4. Attempt to solve the problem.

Unless you do steps 1 through 3 carefully, there's a good chance that you will run into a "dead-end" part way through the problem.

To solve this equation, for example, we had to start at the right side of the equation to avoid running out of storage registers. Had we started at the left side, the problem couldn't be solved without writing down an intermediate answer.

The System B solution continues to be relatively straight-forward. But, since our mixed-calculation problem involves several intermediate answers, it exceeds the storage capacity of calculators using this type of logic. Thus, the user must stop midway through the problem to write down a partial answer. And, he must key it in again later.

This may seem like a small inconvenience unless it happens several times during a lengthy calculation involving numbers of eight or nine digits each. In such cases, manual re-entry of data can significantly increase the chance for error.

Comparison 5: another mixed calculation.

To close this three-calculator comparison, we'd like to invite you, the reader, to figure out how each type of calculator would solve the following problem. Although you'll find the answers at the bottom of this page, cover them up for a moment and see if you can work out the correct keystroke sequences.

Problem: $\left[(2+3)\times(7-4)\right] + \left[\text{LOG}\sqrt{(5+8)\times(9-2)}\right] = 15.97952070$

Hint: all three calculators have [√x] and [log] keys. On all machines you press them *after* the number whose function you want to take appears in the display; i.e., to take the square root of 9, you'd press 9 [√x] on all three machines.

Your RPN solution: ______________________

____

Your System A solution: ______________________

____

Your System B solution: ______________________

____

Our solutions:

1. On the RPN calculator, this problem can be solved the same way as all the others we have looked at in this booklet. The correct keystroke sequence is: 2 [ENTER↑] 3 [+] 7 [ENTER↑] 4 [−] [×] 5 [ENTER↑] 8 [+] 9 [ENTER↑] 2 [−] [×] [√x] [log] [+]
2. On the System A and System B calculators, you must write down an intermediate answer. The keystroke sequence for both machines would be:
   2 [+] 3 [=] [STO] 7 [−] 4 [=]* [×] [RCL] [=]
   Write down intermediate answer
   5 [+] 8 [=] [STO] 9 [−] 2 [=]* [×] [RCL] [=] [√x] [log] [+]
   re-enter intermediate answer
   [=]

*These two [=] keystrokes are not necessary on the System B calculator.

Hewlett-Packard confidently invites you to compare the problem-solving power of its pocket-sized computer calculators against any other brand.

But when you do, be sure to consider the inside story.

We sincerely hope that the preceding pages have convinced you that there's more to choosing a scientific pocket calculator than size, price or the number of features and functions apparent from the keyboard.

And because only Hewlett-Packard pocket-sized computer calculators offer RPN and the powerful 4-register operational memory stack, we believe that only they give you:

- a simple, efficient and consistent way to solve complex, "real-world" scientific problems, and
- the confidence that comes from knowing that every problem can be approached the same way; that you can trust the answers you get.

That's why we invite you to compare our pocket-sized scientific computer calculators against any other brand. We feel sure that once you've considered the inside story, you'll select Hewlett-Packard.

All these Hewlett-Packard pocket-sized and desktop computer calculators give you the advantages of RPN and a 4-register operational memory stack. There's one for almost every calculating need.

A word about HP.

The calculators described in this booklet are only an example of Hewlett-Packard's capabilities in precision electronic measurement, analysis and computation. Other HP products range from the atomic clocks used by most observatories throughout the world through programmable desktop calculators and minicomputers to full system installations.

HEWLETT hp PACKARD

Sales and service from 172 offices in 65 countries.
19310 Pruneridge Avenue, Cupertino, CA 95014

Pub: 5952-6035 4/74 Printed in U.S.A.