Volume 7
Number 2

35¢

September, 1956

THE ORIGINAL SCIENCE FICTION STORIES

NOVELETS

SHORT STORIES

FEATURES

READERS' DEPARTMENTS

Editor: ROBERT W. LOWNDES MARIE A. PARK, Asso. Ed.
Art Director: WILL LUTON DOROTHY B. SEADOR, Asso. Ed.
COVER BY EMSH Illustrations by Emsh, Freas, Garrett and Orban

SCIENCE FICTION STORIES, September, 1956, published bi-monthly by COLUMBIA PUBLICATIONS, INC., 1 Appleton Street, Holyoke, Mass. Editorial and executive offices at 241 Church Street, New York 13, New York. Entered as second class matter at the Post Office at Holyoke, Mass., under the act of March 3, 1879. 35¢ per copy; yearly subscriptions $2.10. Printed in the U. S. A.

HOW TO COUNT ON YOUR FINGERS

ARTICLE

by the author
of "Wapshot's Demon"

FREDERICK PHOL

EVERYONE knows that the decimal system of counting, which is based on the ten digits 0 through 9, has driven out all other systems and has become universal, by virtue of being simplest and best. Like a good many things that "everyone knows", there is one thing wrong with that statement. It isn't so.

True, it is not that any of the predecessors of the decimal system is likely to make a comeback. There is, for instance, vanishingly little chance that we will return to the Babylonian sexagesimal (to the base 60) system—though that is a tough old bird, and will not be finally dead as long as we count sixty minutes to the hour, and 360 degrees to the circle. There are traces of systems to other bases surviving in such terms as "score" and the French word for eighty, *"quatre-vingt"*, suggesting an extinct system to the base twenty; and in terms like "dozen", "gross", and so on, which appear to derive from a system to the base twelve.

In science fiction, most of

the speculation on numbering-systems of the future has dwelt on this base twelve "duodecimal") system; but it is hard to understand why. It is argued that a twelve-digit system simplifies writing "decimal" equivalents of such fractions as 1/3 and 1/6; but that seems a small reward for the enormous task of conversion. Setting aside the merits or demerits of the duodecimal system itself, think of the cost of such a change. For a starter, our decimal system of coinage either goes down the drain, to be replaced by a new one, or lingers on as a clumsy anachronism like the British L/s/d. And that cost is only the bare beginning. Science is measurement and interpretation; without measurement, interpretation is foggy soul-searching; and measurement is number. Change our system of writing numbers, and you must translate nearly the entire recorded body of human knowledge—lab reports and tax returns, cost estimates and time studies, knowledge about the behavior of *mu* mesons, and knowledge about transactions on the New York Stock Exchange.

The project of converting the world's essential records from one system of numbering to another staggers the mind. Its cost is measureable not merely in millions of dollars, but in perhaps millions of man-years.

That being so, why is this enormous project now in process?

THE ANSWER is, simply, that machines aren't any smarter than Russian peasants.

This is not meant to run down the Russians, but only to observe that UNIVAC and Ivan have a lot of things in common—and one of these things is a lack of skill in performing decimal multiplication and division.

Let's take a simple sum—say, 87 x 93—and see how it would be done by us, by Ivan and by UNIVAC. You and I, having completed at least a couple of years of grade school, write down a compact little operation like this:

$$\begin{array}{r} 87 \\ \times 93 \\ \hline 261 \\ \hline 783 \\ \hline 8091 \end{array}$$

That wasn't hard to do. If we had to, we probably could have done it in our heads.

However, Ivan would find that pretty hard, because he didn't happen to go to grade school. (And neither did

UNIVAC.) What Ivan would do in a similiar case is a process called "Russian multiplication"—or, sometimes, "mediation and duplication." (Which is to say, "halving and doubling".) It consists merely of writing down two columns of figures, side by side. The first column starts with one of your original figures, which is successively halved until there is nothing left to halve. Ivan didn't understand fractions very well, so he simply threw them away—he would write half of 25, for instance, as 12.

The second column starts with the other number, which is successively doubled as many times as the first number was halved. As follows:

87	93
43	186
21	372
10	744
5	1488
2	2976
1	5952

Having got this far, Ivan examines the left-hand, or halved, column for even numbers. He finds two of them—the fourth number, 10, and the sixth, 2. He strikes out the numbers next to them in the right-hand (or doubled) column—that is, 744 and 2976. He then adds up the remaining numbers in the right-hand column:

93
186
372
1488
5952
8091

Having gone all around Robin Hood's barn to do it, as it appears, he has wound up with the same answer we got.

That may not seem like much of an accomplishment, at first glimpse, until you stop to think of Ivan's innocence of the multiplication table; and then it becomes pretty ingenious indeed. Ivan turns out to be a clever fellow.

Yet he was not so clever, all the same, but what he would have laughed in your face if you had accused him of seeking help from the binary system of numbering.

BUT THAT is what he did; and that, of course, is what UNIVAC and its electronic brothers do today.

To see *how* UNIVAC does this, let's take some numbers apart and see what is inside them.

Our own decimal numbers—87, for example—are simply a shorthand, "positional" way of saying (in this case) 8×10^1

plus 7 x 10^0. The larger the number the shorter the shorthand becomes. 1956, for instance, is shorthand for one-times-ten-cubed, plus nine-times-ten-squared, plus five-times-ten, plus six-times-one. Or:

$$\begin{aligned} 1 \times 10^3 &= 1{,}000 \\ 9 \times 10^2 &= 900 \\ 5 \times 10^1 &= 50 \\ 6 \times 10^0 &= 6 \\ &\hline \\ &1{,}956 \end{aligned}$$

(In case it has been a long time since you went to high-school, 10^1 just means 10; 10^0 means ten divided by ten, or 1. No matter how long it has been since you went to high school, you ought to remember that 10^2 means ten times ten, or a hundred, and so on.)

It has been said in many science-fiction stories (and not very often anywhere else) that this is homo sapiens, "natural" system of counting, because, look, don't we have ten fingers on our hands? As a theory, let's not worry ourselves about this too much; if true, it will have plenty of chance to prove itself when our exploring rockets turn up some twelve-digited and duodecimal extraterrestrials. (Or alternatively, when our archaeologists discover that the Babylonians had six times as many fingers as the rest of us.) Still, if we assume the fable is true we can conveniently "explain" UNIVAC by saying that the computer, not having ten fingers to count on, has to use a simpler system. The name of this simpler system is the "binary" or "dyadic" system; and it is this system that most of the world's numbers are being translated into now, in order to be taped and fed into computers.

The binary system obeys all the laws of the decimal. It is positional; it can represent any finite number; it can be used for addition, subtraction, multiplication, division, exponential functions and any other arithmetical process known to man or to UNIVAC. The only difference is that it is to the base 2 instead of the base 10. It lops off eight of the ten basic decimal digits—0, 1, 2, 3, 4, 5, 6, 7, 8 and 9—retaining only 0 and 1.

You can count with it, of course. 1 is one. 10 is two. 11 is three. 100 is four. 101 is five, 110 is six; 111 is seven; 1000 is eight; 1001 is nine; 1011 is ten, and so on. You can subtract or add with it:

Four	100
Plus three	11
Is seven	111

You can multiply or divide with it:

Six	110
Divided by three	11
Is two	10

And you can do all of these things rather simply, without the necessity of memorizing multiplication tables, thus freeing your pre-adolescent evenings for baseball and doorbell-ringing.

LOOK BACK at Ivan's system of Russian multiplication; let us do it over again in a slightly different way. Let's halve both columns, the right as well as the left. And instead of striking out any numbers, let us write a "1" next to the odd numbers and a "0" next to the even ones. As follows:

87	1	93	1
43	1	46	0
21	1	23	1
10	0	11	1
5	1	5	1
2	0	2	0
1	1	1	1

Now, you might not know what you have just accomplished—and Ivan certainly wouldn't—but you have translated two decimal numbers into their binary equivalents. Reading from bottom to top, 1010111 is binary for 87; 1011101 is binary for 93.

To see what these mean, remember how we dissected a decimal number. A binary number comes apart in the same sort of pieces; the only difference is that the pieces are multiples of powers of 2, not of powers of 10. 1010111, then, is a shorthand way of saying:

$$
\begin{aligned}
1 \times 2^6 &= 64 \\
0 \times 2^5 &= 0 \\
1 \times 2^4 &= 16 \\
0 \times 2^3 &= 0 \\
1 \times 2^2 &= 4 \\
1 \times 2^1 &= 2 \\
1 \times 2^0 &= 1 \\
\hline
&87
\end{aligned}
$$

which is what we said it was in the first place.

When you feed numbers like 87 and 93 into UNIVAC, its digestion gets upset—in fact, it won't accept them until they are predigested. So you must first convert them into binary digits ("bidgets" or "bits"), just as we did above. Such binary numbers as 1010111 and 1011101 UNIVAC handles very well indeed. Multiply them? No trouble at all. UNIVAC, in its electronic way, does something like this:

```
    1010111
   x1011101
  ---------
    1010111
          0
   1010111
  1010111
 1010111
          0
1010111
-------------
1111110011011
```

That may look frightening, because it is unfamiliar; but it is still the same old product of 87 x 93; it is shorthand for:

1×2^{12}	4096
1×2^{11}	2048
1×2^{10}	1024
1×2^{9}	512
1×2^{8}	256
1×2^{7}	128
0×2^{6}	0
0×2^{5}	0
1×2^{4}	16
1×2^{3}	8
0×2^{2}	0
1×2^{1}	2
1×2^{0}	1
	8091

Observe the simplicity! True, the number is long; but see how simple manipulating it becomes. Addition, for instance, is reduced to simple counting. (Binary counting, of course—1, 10, 11, 100 and so on. You can call it "one", "ten", "eleven" and "one hundred" and so on, if you like, with no great harm.) To add a column of figures, like

```
  101
  100
  110
  111
-------
10110
```

you simply count the ones in the right-hand column (1, 10; write down 0 and 1 to carry); then count the ones in the middle column, starting of course with the one you carried (1, 10, 11; write down 1 and 1 to carry); then count the ones in the left-hand column, again remembering the one you carried (1, 10, 11, 100, 101; write down 1 and 10 to carry; write down the 10.)

That is, I submit, about as simple as an arithmetical operation can get; and multiplication is nearly as much so. Multiplication becomes merely a matter of writing down the number, moved an appropriate number of places to the left, or not writing down the number at all (depending on whether the digit you are multiplying by is "1" or "0"). Thereafter it is addition; and addition, as we have seen, is merely counting. No multiplication tables! No tedious memorizing! No wonder UNIVAC and Ivan like it!

If binary arithmatic has a

fault, it is that it is so excessively easy. It becomes boring.

But the world's work is full of boring operations that get done, anyhow. We have found two good ways to handle them —either to turn them over to machines (like UNIVAC), which do not have the capacity for boredom; or to learn to do them as a matter of mechanical routine.

MY WIFE observes (as most wives sometimes observe) that it doesn't much matter what sort of change she suggests, I can usually find a dozen splendid reasons for keeping things just as they are. Since the human animal is conservative, most of us can find objections to any sort of change. ("Better the devil you know.") Since the human animal is also educable, we often, however, overcome our objections when the change promises rewards.

Let us see what the drawbacks and rewards of change-over to binary notation may be. Not that the case is really arguable, since the silent vote of the computers constitutes a carrying majority over our human veto; but let us see if there are any advantages for us borable, error-prone humans.

The drawbacks stand out immediately, starting with the sheer physical size of a binary number as contrasted with its decimal equivalent. Still, a binary number isn't so *very* much longer than a decimal (about three times) as to be *ipso facto* out of the question. As a matter of fact, really large numbers are hopelessly unwieldy in any notation at all. In the prevailing decimal system, scientific people express large numbers either as approximations (3×10^{47}, for instance) or in terms of their prime factors and exponents ($19^3 \times 641^5 \times 1861$), or in other factored or shorthand ways. Even the headlines in our daily papers are more likely to read $6.5 BILLION than $6.500,000,000.

For "household-sized" numbers—oh, up to a million, let's say—it doesn't seem as though the mere matter of length ought to be a prevailing count against binary notation. You might use twenty binary digits to write a number that big (as against seven in decimal notation); and a number such as—to take one at random—10100111100101100010 is pretty hideous. But is 1372866, its decimal equivalent, utterly lovely?

Perhaps the number itself isn't so bad; perhaps the way we are reading it could stand some improvement. Look at

the number 1111110011011, for instance. You just came across it a couple of pages ago (our old friend, the product of 87 and 93), and yet you almost certainly failed to recognize it. Is it because its recognition value is intrinsically low? Or because we lack practice in reading (and in establishing conventions of writing) that sort of number?

In decimal notation, remember, we simplify the reading of such large numbers by setting off groups of three. 5000000000 is pretty hard to read by itself, though 5,000,000,000 reveals itself to be five trillion rather conveniently. Why should we not adopt a similar convention for binary numbers? There is no reason to stick to groups of three; let's make it groups of five, and thus write the expression for the product of 87 x 93—that is, 8091—as follows: 111,11100,11011.

Well, that's a help; but as is often the case, a little progress in one direction merely brightens the light on a related problem still unsolved. The related problem here is the problem of subvocalization. All of us are lipreaders; even if the motion of the lip muscles is so thoroughly suppressed as to be invisible to the naked eye, the larynx is still forming the sounds of everything we read—or think, for that matter. And groups like oneoneone comma oneoneoneohoh comma oneoneohoneone simply do not pronounce well.

But being able to state a problem is progressing far toward solving it. It is apparent that there is no difficulty involved in assigning more pronounceable phonetic values to the parts of binary notation.

ONE SUCH system is, in fact, already widely in use. If you walk into the *Bank of Ireland* bar in Chelsea on a noisy night, you may come across a couple of Merchant Marine officers having a relaxed and private conversation which is not subject to either eavesdropping or interference, regardless of the surrounding noise. If you do, they are probably radiomen; and they are talking to each other in code. For the dots and dashes of Morse there is a well established pronouncing convention; "dit" is a dot, "dah" is a dash. If we merely appropriate this convention for our binary numbers, we may sacrifice some efficiency —no doubt an even more compact and clear system could be worked out from basic phonetic principles. But it offers a very special advantage: it

works. We don't have to test it or doubt it; we know it works; it has worked all over the world for countless radio operators for a period of decades.

Let us then pronounce "1" as "dit" and "0" as "dah". 111,-11100,11011 then becomes dididit dididididahdah dididahdidit—

And we notice something odd. We already conceded that the binary system had an intrinsic drawback, in that its terms were by definition *always* less compact than the decimal.

Yet if we wish to transmit the decimal number 8091 in Morse code, it must be expressed like this: dahdahdahdidit dahdahdahdahdah dahdahdahdahdit didahdahdahdah. That is, four groups, each comprising five "bits", or twenty "bits" in all.

But its binary equivalent needs only three groups totalling thirteen "bits", as we have just seen.

Our concession was evidently premature. In this one special case, at least—and it is far from an unimportant one—the binary system can be made *more* compact than the decimal.

Having found one such case, let us be encouraged enough to look for more.

WHEN I was around ten, we kids used to kill time on long auto rides by playing a game of counting. We would pick a common phenomenon—cows, or Fords, or "For Sale" signs on farms—and see who, in a given period, had spotted the most. It almost always kept us quiet and out of the driver's hair for the first mile or two—and almost never beyond that.

The trouble was that we counted on our fingers. That worked beautifully for numbers up to ten, of course. It worked passably for numbers up to twenty, or even to thirty—it wasn't much of a trick to remember that we were on the second, or third, go-round of finger-counting. But when we got to numbers much above that we began to rely pretty heavily on our individually differing memories of just how many times we had counted up to ten; and that's when the fights would start.

Naturally, we were counting by the decimal system.

Could we have done better with the binary?

Spread out the ten fingers on your two hands before you (don't let's get into semantic arguments about whether a "thumb" is a "finger"—you know what I mean), and let's see what can be done with them.

We start by establishing a convention. An extended finger is a "1". A retracted finger is a "0".

Clench your fists, and begin to count:

Extend the right little finger. That is 1—in both binary and decimal notation.

Retract the little finger and extend the right ring finger. Read it as 10 (or, in decimal notation, two).

Keep the ring finger out and extend the little finger beside it. Read: 11. (Decimal notation, three.)

Retract both those fingers, and extend the middle finger of the right hand. Read 100 (binary) or four (decimal).

And so on. You may find that waggling your fingers like this requires practice or natural flexibility—unless, of course, you make it easy on yourself by resting the fingers against the edge of a table.

Your fingers are now indeed "digits", and you are using them in positional notation. Observe that you can represent any number from 00000,00000 (both hands clenched) to 11111,11111 (both hands extended). Next time you want to count a reasonably large number—say, the number of cars ahead of you in a tunnel jam, or the number of hits against a Giant pitcher—you might try this system. It's good anywhere from 0 to 1023. Indeed, by a few obvious extensions—for instance, by adding on successively extended or retracted positions of the wrist, elbows and so on—you can soon reach numbers beyond which you are never likely to count.

Moreover, your running total is available to you at any time (as it is not in the decimal system of finger counting, for instance, where you must count the fingers themselves to get such a total); you merely read it off. Suppose, for instance, you are out hiking with a companion (having lost your pedometer, let us say), and your friend wants to know how many paces you can go in a given period of time. You keep count on your fingers; and at the end of the time you see you have the little finger, index finger and thumb of the left hand, and the thumb and ring finger of the right hand, extended. Reading off your hands according to our established convention, you find you have come 10011,10010 paces, and you pass on the information to him according to the pronouncing convention: "didahdahdidit didahdahdidah."

Of course, your friend may

be a square who still uses the old stick-in-the-mud decimal system, so you may want to translate for him. That's easy enough, if you remember the decimal equivalents of each of the fingers:

Left Hand

Little finger:	$2^9 = 512$
Ring finger:	$2^8 = 256$
Middle finger:	$2^7 = 128$
Index finger:	$2^6 = 64$
Thumb:	$2^5 = 32$

Right Hand

Thumb:	$2^4 = 16$
Index finger:	$2^3 = 8$
Middle finger:	$2^2 = 4$
Ring finger:	$2^1 = 2$
Little finger:	$2^0 = 1$

Accordingly, to convert your finger-count into decimal figures, just add up the finger equivalents given above; for the aforementioned 10011,10010, read:

Left little finger:	512
Left index finger:	64
Left thumb:	32
Right thumb:	16
Right ring finger:	2
	626

And inform your friend you have come 626 paces.

AS PROMISED, we have found another case where an ingenious use of binary notation is actually more compact than decimal—*by a factor of 100*, as it turns out. Let us turn, then, away from the demolished "drawbacks" of binary notation in order to take a quick look at some of its more attractive features.

We recall that it has already been demonstrated that binary arithmetic is about as simple as arithmetic can get. That is what makes it so uniquely right for UNIVAC; but even on a less complicated level of computer design, it presents lovely aspects. Think, for example, of the beautifully compact desk adding machine that might be designed for binary numbers. No wheels and gear trains—therefore, at least for normal-sized calculations, no necessity for a power source to drive them. To handle addition or subtraction of, say, ten-place numbers (and multiplication and division are only slightly more demanding) you need only a row of ten levers with an up ("1") and down ("0") position. Of course, you wouldn't have to spend much money on a calculator as simple as that. You could build it yourself. Or alternatively, you could merely use the built-in ten-place binary computer we've just been talking about, the one that grows out of the ends of your arms.

For instance: You're remodeling your house; you have 13 4x8 panels of sheet rock on hand, and you discover that you have 650 square feet of wall to cover. Question: How many additional panels of sheet rock will you have to go out and buy?

That's not the most difficult problem in the world, true; but let's run through it once in binary arithmetic, using our fingers as computers. First we need to convert to binary numbers—*but only because we chose to start out with decimal ones;* it isn't fair to include conversion time as part of the time required for solving the problem.

In binary numbers, you have 1101 100x1000 panels on hand, and 10100,01010 square feet of wall to cover.

1101x100x1000, obviously, is merely a matter of pointing off places; you represent 01101 on your left hand, and 00000 on your right hand; that's how many square feet of sheet rock you have—uh, on hand, so to speak. Then the subtraction* is merely a matter of considering the successive digits, reading from the right, subtracting the digit shown on your finger from the corresponding digit in the written number you are subtracting from, *and carrying "borrowed" numbers.* (Are you able to remember how much trouble you had with "carrying" when you first learned the principles of decimal subtraction? Then don't give up on binary subtraction, if it takes you a few minutes to get the hang of "carrying" here.)

The result you "write", one digit at a time, on your fingers. That is, by the time you are subtracting your right-thumb digit from the written figure, the remaining fingers on your right hand are already indicating the last four digits of the answer. When you're done, you read the answer off.

AS ALREADY shown, the number of square feet of sheet rock you need to buy is 111,01010 (we padded the left-hand group out with zeroes to indicate all five finger positions in writing the subtraction). There are 1,00000 square feet in a panel; 111,01010 di-

* 10100,01010 sq/ft wall to cover
-01101,00000 sq/ft sheet rock on hand

00111,01010 sq/ft needed

vided by 1,00000 is obviously 111 and a fraction. But you can't buy part of a panel so you add 1 to the 111 and get 1000. Answer: You need to buy 1000 panels. (Or, in decimal numbers, 8.)

Look hard? Once again, consider it from the perspective of *relative* difficulty. After all, this is probably your first binary problem. Make up a few more; by the time you've done six, it won't be hard at all; by the time you've done a hundred, it will be semi-automatic; by the time you've done a thousand—

Well, hold on for a moment before you do your thousand; perhaps it will cheer you up to know that there are some special cases of binary arithmetic which aren't *ever* hard, not even the first time.

For example: Multiplication (or division) by powers of 2 is an obvious case; you simply point off and add zeroes. True, the decimal system has a similar situation in regard to powers of 10. But you still have to give the verdict to binary on this point, simply because in any finite series there are more powers of 2 than powers of 10.

But if you want to see something really *easy*, consider the strange case of the problem 1023-*n*.

Let's arbitrarily take *n* as 626 (because we happen to have a binary equivalent conveniently to hand—any other number less than 1023 would do as well, of course). Do this one on your fingers. First show yourself the binary representation of 1023:

11111,11111

Then cancel that, and represent on your fingers the binary equivalent of 626:

10011,10010

Don't bother about subtracting; you've already done it! Just reverse your convention for reading finger representations; read an extended finger as "0", a retracted finger as "1", and you get:

```
  11111,11111
- 10011,10010
-------------
  01100,01101
```

In other words, any number *n* in binary notation is always the "reverse" of the number 1023-*n*. Not only that, but the same sort of rule can be made for the cases 511-*n*, 255-*n*, 127-*n*, etc.—for any number whose binary representation is "all ones", as you may already have realized. Try it and see.

It may be objected that such

special cases are comparatively rare. This is true enough; but in the decimal system, they are not only rare, they do not exist at all. And we have not, by any means, exhausted binary's bag of tricks. It is, in fact, hardly possible that any reader of this publication can spend as much as a single evening trying out experiments in binary arithmetic without discovering additional shortcuts to this one.

Decimal system?

That clumsy, s p r a w l y, quaint old thing!

———— ★ ————