Reprinted from the MATHEMATICS MAGAZINE
Vol. 52, No. 4, September, 1979
p. 240

Negative Based Number Systems

WILLIAM J. GILBERT
University of Waterloo
Waterloo, Ontario, Canada N2L 3G1

R. JAMES GREEN, student
University of Toronto
Toronto, Ontario, Canada M5S 1A1

Many number systems, besides the decimal system, are used for arithmetical calculation. Computers, in their internal working, usually use the binary system or sometimes the base 16 hexadecimal system. The Babylonians used base 60 for their number notation and the remains of this system can still be seen in our division of the hour and minute into 60 parts. In fact, any integer larger than one can be used as a base for the positive numbers. However, what is not so well known is that all the numbers, both positive and negative, can be represented by means of a negative integral base [1], [3]. Besides its intrinsic interest, the study of such a system forces one to understand and reevaluate the properties of positive bases that one takes for granted.

The representation of numbers in a negative base is simplified because there is no need for a sign to be attached to each negative number; it is already built in. For example, $-326=(-10)^3+7(-10)^2+3(-10)+4$, so -326 is represented by 1734 in base -10. A computer using base -2 has even been built that exploits this fact [4]. As we shall show below, all integers may be uniquely represented in a negative base; in fact, this representation is "more unique" than with a positive base because, without signs, there is not the problem of $+0$ being equal to -0. We can add, subtract and multiply in a negative base, even though we may obtain an infinite series of carry digits. This problem, however, can be overcome in various ways. We shall also show how to divide by integers to obtain negative based expansions of fractions. As with decimals, certain fractions have two different periodic expansions.

There have been other systems proposed for representing both positive and negative numbers without using a sign as a prefix. These usually consist of allowing the digits used in the expansion to include negative numbers. Linderholm [2; p. 63] suggests that the symbols ᔭ,Ɛ,Ƨ,I 0,1,2,3,4,5 be used in decimal expansions, where ᔭ stands for -4, etc. For example, ƧƐ4 represents $-200-30+4=-226$. This is rather similar to the Yoruba system of numeration used in part of Nigeria [6]. In general, both positive and negative numbers can be represented in a positive base b if the digits allowed are any b consecutive integers that contain -1, 0 and 1. If b is odd, then the digits can be chosen symmetrically about zero and this simplifies many calculations [5].

To formalize negative bases, we say that the integer N is expressed in the base b if it is written in the form $N=\Sigma_{k=0}^{m} a_k b^k$, where $0 \leqslant a_k < |b|$. We denote this by $N=(a_m a_{m-1} \ldots a_1 a_0)_b$. If b is ten, we omit the parentheses and subscript b to obtain the usual decimal expansion. Whether the base is positive or negative, the digits a_k in the expansion can be calculated in the usual way by setting $q_0=N$ and then repeatedly using the division algorithm $q_k=q_{k+1}b+a_k$, where $0 \leqslant a_k < |b|$, until the quotient becomes zero. For example, let us convert 34 to negative decimal (base -10) and negative binary (base -2):

$$\begin{array}{rl} 34= & (-3)(-10)+4 \\ -3= & 1(-10)+7 \\ 1= & 0(-10)+1 \end{array} \qquad \begin{array}{rl} 34= & (-17(-2)+0 \\ -17= & 9(-2)+1 \\ 9= & (-4)(-2)+1 \\ -4= & 2(-2)+0 \\ 2= & (-1)(-2)+0 \\ -1= & 1(-2)+1 \\ 1= & 0(-2)+1 \end{array}$$

Hence, $34=(174)_{-10}$ and $34=(1100110)_{-2}$. We can check these calculations by expanding the numbers to obtain

$$(174)_{-10}=1(-10)^2+7(-10)+4=100-70+4=34$$

and

$$(1100110)_{-2}=(-2)^6+(-2)^5+(-2)^2+(-2)=64-32+4-2=34.$$

Some other examples of expansions in negative binary and negative decimal are given in TABLE 1.

Decimal	Negative Decimal	Negative Binary
-12	$(28)_{-10}$	$(110100)_{-2}$
-11	$(29)_{-10}$	$(110101)_{-2}$
-10	$(10)_{-10}$	$(1010)_{-2}$
-9	$(11)_{-10}$	$(1011)_{-2}$
-8	$(12)_{-10}$	$(1000)_{-2}$
-7	$(13)_{-10}$	$(1001)_{-2}$
-6	$(14)_{-10}$	$(1110)_{-2}$
-5	$(15)_{-10}$	$(1111)_{-2}$
-4	$(16)_{-10}$	$(1100)_{-2}$
-3	$(17)_{-10}$	$(1101)_{-2}$
-2	$(18)_{-10}$	$(10)_{-2}$
-1	$(19)_{-10}$	$(11)_{-2}$

Decimal	Negative Decimal	Negative Binary
0	$(0)_{-10}$	$(0)_{-2}$
1	$(1)_{-10}$	$(1)_{-2}$
2	$(2)_{-10}$	$(110)_{-2}$
3	$(3)_{-10}$	$(111)_{-2}$
4	$(4)_{-10}$	$(100)_{-2}$
5	$(5)_{-10}$	$(101)_{-2}$
6	$(6)_{-10}$	$(11010)_{-2}$
7	$(7)_{-10}$	$(11011)_{-2}$
8	$(8)_{-10}$	$(11000)_{-2}$
9	$(9)_{-10}$	$(11001)_{-2}$
10	$(190)_{-10}$	$(11110)_{-2}$
11	$(191)_{-10}$	$(11111)_{-2}$

Expansions in various bases.
TABLE 1

If the base is a negative integer less than minus one, say $b=-s$, then *every integer, positive or negative, can be expanded uniquely in base b*. To show the existence of the expansion, we have to prove that the algorithm for finding the digits a_k always terminates. The successive quotients in the algorithm are given by $q_{k+1}=(q_k-a_k)/(-s)$, where $0 \leqslant a_k < s$ and $a_k \equiv q_k (\text{mod } s)$. Now if $q_k>0$ then $|q_{k+1}| \leqslant |q_k/s| < |q_k|$. If $q_k<0$, then $|q_{k+1}|=|(-q_k+a_k)/s| < |q_k/s|+1 \leqslant |q_k|$, unless $q_k=-1$. In the case $q_k=-1$ we have $a_k=s-1$ and $q_{k+1}=1$; however $q_{k+2}=0$. Therefore, the sequence of the absolute values of the quotients, $|q_k|$, decreases until it eventually becomes zero; thus the algorithm always terminates. The uniqueness of the digits in the base b expansion of an integer can be proved as follows. If two expansions represent the same integer, by looking at congruences modulo $|b|$ it can be seen that their rightmost digits must be the same. Subtracting these digits from their representations, dividing by b, and then repeating the argument will show that in each position their digits are the same.

Since a number in a negative base has no sign prefix, how do we tell whether it is positive or negative? The answer is simple: it is positive if it contains an odd number of digits, not counting leading zeros, and negative otherwise. You can tell which of two numbers is larger by comparing the digits of the highest power of the base in which they differ. If they begin to differ in an even power of the base, then the larger is the one with the larger digit. However, if they begin to differ in an odd power of the base, the larger is the one with the smaller digit. For example, $(3326)_{-10}>(3354)_{-10}$ because they begin to differ in the first power of -10. Using this rule we see that $(3547)_{-10}>(3261)_{-10}$ and $(111)_{-2}>(1010)_{-2}$.

We can add, subtract and multiply numbers in negative bases in the usual way. However, the carry digits are more complicated. Since $s=(1\,\overline{s-1}\,0)_{-s}$, where the symbol $\overline{s-1}$ stands for the single digit with value $s-1$, instead of carrying 1 we have to carry 1 $\overline{s-1}$ and this affects the next two higher places. For example, $2=(110)_{-2}$, $10=(190)_{-10}$, $20=(180)_{-10}$, etc; hence, instead of carrying one in negative binary we carry 11, instead of carrying one or two in negative decimal we carry 19 or 18 respectively. The following sample calculations are all done in base -10. In subtraction, we can borrow 10 in one column by adding 1 to the next higher column. We have dropped the subscripts -10 for convenience, and displayed carry digits in smaller type.

$$\begin{array}{r} 204 \\ +107 \\ \hline 491 \\ \hline {\scriptstyle 19} \end{array} \qquad \begin{array}{r} 204 \\ \times\ 107 \\ \hline 19588 \\ 20400 \\ \hline 39988 \\ \hline \end{array} \qquad \begin{array}{r} {\scriptstyle 6\,1} \\ \not{5}7 \\ -48 \\ \hline 29 \\ \hline \end{array}$$

However, it often happens that the carry digits accumulate and we obtain an infinite series of carry digits. Compare the following two additions in base -10.

$$\begin{array}{r} 55 \\ +\ 27 \\ \hline \ldots 00062 \\ \hline {\scriptstyle 19} \\ {\scriptstyle 19\ } \\ {\scriptstyle 19\quad\ } \\ \cdot\quad\ \ \\ \cdot\quad\ \ \ \\ \cdot\quad\ \ \ \ \end{array} \qquad \begin{array}{r} 19 \\ +\ \ 1 \\ \hline \ldots 00000 \\ \hline {\scriptstyle 19} \\ {\scriptstyle 19\ } \\ {\scriptstyle 19\quad\ } \\ \cdot\quad\ \ \\ \cdot\quad\ \ \ \\ \cdot\quad\ \ \ \ \end{array}$$

It is clear from the second example above that this situation will always happen whenever we can represent both positive and negative numbers; since $-1=(1\,\overline{s-1})_{-s}$, adding 1 to this must give zero with an infinite series of carry digits. However, even though there is an infinite number of carry digits, the correct answer is obtained in a finite number of steps. It can be shown that if we add two numbers with r or fewer digits, then their sum contains $r+2$ or fewer digits. We could therefore program a computer to perform this arithmetic and not worry about the possibility of an infinite series of carry digits. In doing the arithmetic by hand, one soon recognizes those combinations of digits that sum to zero.

So far we have just considered integers in negative bases. However, it is also possible to represent any real number in a negative base by using an infinite expansion of the form $\Sigma_{k=-\infty}^{m} a_k b^k$ where $0 \leqslant a_k < |b|$. For example, $\frac{3}{8}=1-\frac{1}{2}-\frac{1}{8}=(1.101)_{-2}$. Any rational number p/q can be converted into base b by repeating the following division algorithm.

$$p=aq+r_0$$
$$br_0=a_{-1}q+r_{-1}$$
$$br_{-1}=a_{-2}q+r_{-2}$$
$$\vdots$$

We then obtain $p/q=(a.a_{-1}a_{-2}\ldots)_b$. When the base b and the numbers p and q are positive, the remainders are chosen so that $0 \leqslant r_{-k-1} < q$; this will automatically force the numbers a_{-k} to lie in the required range $0 \leqslant a_{-k} < b$. However this choice of the remainders r_{-k-1} does not work for negative bases. We have to adjust the remainder so that a_{-k} is an allowable digit in the range $0 \leqslant a_{-k} < |b|$ and so that the subsequent remainders stay bounded. There is sometimes a choice, as the following two ways of converting $1/3$ to negative binary shows. (All the numbers shown are in base 10.)

$$\begin{array}{rl} 1=0\cdot 3+1 & \\ \left.\begin{array}{r} 1(-2)=-2=0\cdot 3-2 \\ (-2)(-2)=4=1\cdot 3+1 \\ 1(-2)=-2=0\cdot 3-2 \end{array}\right\} & \text{algorithm repeats} \end{array} \qquad \begin{array}{rl} 1=1\cdot 3-2 & \\ \left.\begin{array}{r} (-2)(-2)=4=1\cdot 3+1 \\ 1(-2)=-2=0\cdot 3-2 \\ (-2)(-2)=4=1\cdot 3+1 \end{array}\right\} & \text{algorithm repeats} \end{array}$$

Hence $1/3=(.010101\ldots)_{-2}=(1.101010\ldots)_{-2}$. Both these repeating expansions can be checked by converting them back to fractional form in the usual way. This shows that the representation is not unique; this fact is well known in positive bases where, for example, $.5=.4999\ldots$ in the decimal system.

How do we decide which remainder to use at each stage of the algorithm? The remainder r_{-k} must of course be congruent modulo q to br_{-k-1}. The algorithm must be carried out as follows: In division by the positive integer q in the negative base $b=-s$, the digits a_{-k} are chosen in the range $0 \leqslant a_{-k} \leqslant |b|$ so that the remainders r_{-k} lie in the range $-[sq/(s+1)] \leqslant r_{-k} \leqslant [q/(s+1)]$, where [ ] denotes the greatest integer function. If q is not a multiple of $s+1$, then this range includes exactly one number from each congruence class modulo q and, as there is no choice for r_{-k}, there is only one representation. However, if q is a multiple of $s+1$, then the ends of the range are congruent modulo q and there is sometimes a choice for the digits. For example, when we divide by 3 in negative binary, the remainders must lie in the range $-2 \leqslant r_{-k} \leqslant 1$ and, as the second line in the previous expansions of $1/3$ show, we sometimes have a choice between -2 and 1 for the remainder. We leave it to the reader to show that the numbers with two different expansions in base $-s$ are those of the form $(-s)^k(a+(1/(s+1)))$, where a and k are integers.

The proof that the algorithm for division by q yields the correct expansion follows by showing that the remainders r_{-k} remain bounded if and only if they are chosen to lie in the stated range. This is done by induction. The basis for the induction is established by finding some integer a for which $p=aq+r_0$, where $-sq/(s+1) \leqslant r_0 \leqslant q/(s+1)$. This is always possible because the range of r_0 includes a complete congruence system modulo q. For the induction step, we split the range of the remainder r_{-k} into four cases and consider each separately.

Firstly, if $r_{-k} > q/(s+1)$, say $r_{-k} = (q/(s+1)) + a$, then $br_{-k} = (-s)r_{-k}$ is negative and a_{-k-1} is taken to be zero in order to keep the size of the remainder as small as possible. Then $br_{-k-1} = b^2 r_{-k} = (s^2 q/(s+1)) + s^2 a$ and, to reduce the size of the remainder as much as possible, we take $a_{-k-2} = s-1$ so that $r_{-k-2} = br_{-k-1} - (s-1)q = (q/(s+1)) + s^2 a$. By induction it follows that $r_{-k-2m} = (q/(s+1)) + s^{2m} a$ which shows that the expansion $(a.a_{-1}a_{-2}\ldots)_b$ will not converge to p/q. Secondly, if $r_{-k} < -sq/(s+1)$, say $r_{-k} = -sq/(s+1) - a$, it can be shown similarly that $r_{-k-2m+1} = (q/(s+1)) + s^{2m-1} a$ and the expansion still will not converge. Thirdly, if $0 \leqslant r_{-k} \leqslant q/(s+1)$ then br_{-k} is negative and we choose $a_{-k-1} = 0$. This means that r_{-k-1} lies in the allowable range $-sq/(s+1) \leqslant r_{-k-1} \leqslant 0$. Finally, if $-sq/(s+1) \leqslant r_{-k} < 0$, it follows that $0 \leqslant br_{-k} \leqslant s^2 q/(s+1) = (s-1)q + (q/(s+1))$. Then br_{-k} lies in the union of the closed intervals $[uq - (sq/(s+1)),\ uq + (sq/(s+1))]$ as u runs from 0 to $s-1$. Hence, there exists an integer a_{-k-1} with $0 \leqslant a_{-k-1} < s$ such that $br_{-k} = a_{-k-1}q + r_{-k-1}$, where $-sq/(s+1) \leqslant r_{-k-1} \leqslant q/(s+1)$. Since r_{-k-1} lies in the allowable range, this completes the induction step and shows that the algorithm for division works.

Because there can be at most $q+1$ choices for each remainder r_{-k}, it is clear that a rational number p/q will always yield a repeating (or terminating) expansion in any negative base. Conversely, as in positive bases, it can be shown that repeating expansions correspond to rational numbers.

The reader should try doing various arithmetical calculations in negative bases and should then check his answers. Addition, multiplication and subtraction can be checked by converting to decimals, while periodic expansions can be checked by finding the rational form in the usual way. Another exercise is to devise a scheme for converting numbers from base s to base $-s$ and then to find π in negative decimal to a certain number of decimal places. Since π is irrational, it will not have a periodic expansion in any base. It is also interesting to look at the standard algorithm for extracting the square root of a number. In negative bases there should be two answers if the number contains an odd number of digits and no answer otherwise.

References

[1] D. E. Knuth, The Art of Computer Programming, Vol. 2, Seminumerical Algorithms, Addison-Wesley, Reading, Mass., 1969.
[2] C. E. Linderholm, Mathematics Made Difficult, World Publishing, New York, 1972.
[3] A. H. Nelson, Investigation to discovery with a negative base, Math. Teacher, 60 (1967) 723–726.
[4] Z. Pawlak, An electronic digital computer based on the "−2" system, Bull. Acad. Polon. Sci., Sér. Sci. Tech., 7 (1959) 713–721.
[5] C. E. Shannon, Symmetrical notation for numbers, Amer. Math. Monthly, 57 (1950) 90–93.
[6] C. Zaslavsky, Africa Counts, Prindle, Weber and Schmidt, Boston, 1973.