ABACUS FOR THE BINARY NUMBER SYSTEM

The abacus is a primitive digital calculating device used mainly for addition and subtraction. The Chinese have used it for well over 2000 years and only over the last fifty years or so have switched over to the use of electronic handheld calculators. In some Chinese restaurants one still sees use of the abacus for adding up dinner bills. We have already discussed the operation of the standard abacus some eight years ago in one of our MATHFUNC web page articles. In it we showed how an abacus(which I built in my woodshop) can be used to add, subtract, multiply, and divide numbers when expressed in decimal form. In re-examining this article recently it became clear to me that there is no need the restrict such devices to a base ten (decimal) system. A binary abacus using the base two should work just as well requiring only a memorized knowledge of two taken to positive and negative integer powers. It is our purpose here to show how such a binary abacus works. It also will serve the purpose of allowing one to become proficient in binary manipulations.

Let us begin with a table giving decimal and binary forms of the first twelve integers. Here they are-

decimal	1	2	3	4	5	6	7	8	9	10	11	12
binary	1	10	11	100	101	110	111	1000	1001	1010	1011	1100

The construction of these binary forms follows by noting any number N may be broken up into sums of 2^n=1,2,4,8,16,32,64, 128,256, etc. So the decimal number 27=16+8+2+1 in binary reads 11011. The 1 refers to an existing power of two and 0 refers to its non-existence. The exponents in the sum are written left to right from the largest to the smallest. Consider the decimal number 341=256+64+16+4+1. In binary it becomes 101010101. Although binary will have more characters per number than the decimal form, its big advantage is that it involves only an on (1) or an off (0) which is easy to simulate with electronic or quantum gates. Doubling a number in binary is accomplished by just adding a zero to the right of its sequence. Thus 4 reads 100 and 8 reads 1000 .

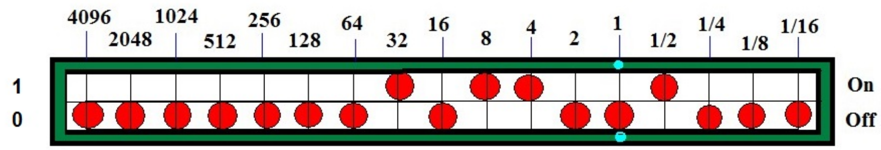
In order to work a binary abacus it is helpful if the operator is familiar with the powers of two for both positive and negative powers. Here is a brief table whose values should be stored in one's memory-

n	-2	-1	0	1	2	3	4	5	6	7	8	9	10
2^n	1/4	1/2	1	2	4	8	16	32	64	128	256	512	1024

Note the negative powers yield values of 2^{-n} which are the reciprocals of the positive powers. That is $2^{-n}\cdot 2^n=1$.

With the above information one can now construct a binary abacus. Here is a schematic of such a device -

BINARY ABACUS SHOWING THE NUMBER 44 1/2=101100.1



Doubling a number just adds a 0 to the right of its binary sequence. Thus

23=10111 with 46=101110 and 94=1011100

This particular abacus consists of 17 equally spaced parallel mounted dowels each of which has a single bead capable of sliding up or down along it. An up postion means ‘on’ and corresponds to 1 and a down position means ‘off’ and is designated by 0. In the schematic we indicate four ‘on’ beads which represent the number N=44.5=32+8+4+1/2. Its binary form reads 101100.1.

Let us demonstrate how one goes about adding two numbers. Consider the numbers-

36=32+4=100100 and 57=32+16+8+1=111001

The addition procedure looks like this-

```
   100100
 + 111001
  -------------
   1011101 =64+16+8+4+1=93
```

On the abacus one first sets the binary number 100100 and then adds 111001 term by term using the recipe that a column containing two zeros stays zero and a column with a zero and a one remains 1. When a column has two ones one gets a zero plus generates a one for the next higher column. We further demonstrate things with the following addition

```
   1101011011=859
 +1011100101=741
   ---------------------
 11001000000 =1600
```

The Mersenne Numbers are given by $M(p)=2^p-1$, with p being a prime. So in binary M(3)=111 and M(5)=11111. So the binary form of any Mersenne Number is just p ones. To handle the addition of several numbers we work things out on the abacus as follows-

1010+1100+1110=10110 +1110= 100100

To subtract two numbers on the binary abacus we handle things as follows-

```
 1011 =11
- 111  =7
 ----------
  100  =4
```

So this is identical to adding 111 to 100 to yield 1011. Electronic computers can handle such manipulations at breakneck speeds in a manner similar to what one does with a binary abacus.

Multiplication and division of numbers in binary can become a bit more complicated. A relatively simple calculation using the binary abacus occurs when any number in binary form is multiplied by the binary equivalent of 2^p. There one needs to only add p zeros to the right of the binary form of the first number. Consider multiplying 131 by 32. In binary this reads-

10000011 x 100000=1000001100000

This last binary product equals 4192 in decimal. For the product of two numbers were neither equals a single power of two, one uses the following partition approach-

11001 x 1001=11001x(1000+1)=11001000+11001=11100001

In decimal this result is-

25 x 9=25(8+1)=225

The square of the number N=17 is set up on the binary abacus as follows-

(10001)x(10000+1)=100010000+10001=100100001

In decimal form we have the equivalent-

17 x(16+1)=289

When it comes to division with abacus we work things as follows-

(1111)/(101)=11

That is, 15/5=3. One notes that 101 x11=1111 Note that when a binary number lies one unit below 2^p, the binary form is always a sequence of p ones. So 15=16-1 reads 1111 in binary.   If it lies one unit above 2^p then the binary form starts with one followed by p-1 zeros and ending in one. That is, 33=32+1 reads 100001 in binary.

The first nine primes 2,3,5,7,11,13,17,19,23 written in binary read-

{10,11,101,111,1011, 1101, 10001, 10011,10111}

The fact that they all end in 1 (with the exception of 2) stems from the fact that primes above two are odd numbers. Otherwise there is no obvious connection between the primes written in binary. The semi-prime 77=7 x 11 is written in binary as-

1001101=111 x 1011

Although we have concentrated in this article on the heretofore unrecognized binary abacus, there is no reason that an abacus cannot be designed for other bases besides 2 and 10. A base 16 (hexadecimal) comes to mind. There the sixteen symbols used to express a number would be a sequence of symbols taken from 0 through 9 plus the letters A,B,C,D,E,F.

U.H.Kurzweg
Feb. 20, 2019
Gainesville, Florida