Decimal/Hexadecimal Conversion

For these first two methods, you only need to know the multiples of fifteen from one through nine:

1 - 15
2 - 30
3 - 45
4 - 60
5 - 75
6 - 90
7 - 105
8 - 120
9 - 135

Conversion of Hexadecimal Whole Numbers to Decimal

The method is to add a hexadecimal digit, starting with the most significant, to an accumulated sum, then multiply the sum by 16, using decimal arithmetic, then repeat the process until all the hexadecimal digits have been added into the sum. The sum is not multiplied by sixteen after the least significant hexadecimal digit is added. By using “multifactorial multiplication” we can multiply by fifteen rather than sixteen, making the process easier. To multiply a number by N with multifactorial multiplication you simply multiply it by N-1 and add the product to the original number

Example: 7D3B → 32059

```
  00000  clear some rods to hold the accumlated decimal number
+     7  add the first hexadecimal digit
--------
      7
+   105  multiply the digit by 15 and add it to the sum
--------
    112
+    13  add the second hexadecimal digit (hex D = decimal 13)
--------
    125
+  15    multiply 1 by 15 and add
+   30   multiply 2 by 15 and add
+    75  multiply 5 by 15 and add
--------
   2000
+     3  add the third hexadecimal digit
--------
   2003
+ 30     multiply 2 by 15 and add
+    45  multiply 3 by 15 and add
--------
  32048
+    11  add the fourth hexadecimal digit (hex B = decimal 11)
--------
  32059  done (no multiply after adding the least significant digit)
```

Conversion of Decimal Fractions to Hexadecimal Fractions

This method requires successively multiplying the decimal fraction by sixteen (or by fifteen using multifactorial multiplication) and after each multiplication, taking the “spillover” to the left of the decimal point as the next hexadecimal digit of the hexadecimal fraction. Of course, spillover numbers larger than nine must be converted to hexadecimal:

10 → A, 11 → B, 12 → C, 13 → D, 14 → E, 15 → F

Example: 0.6397 → 0.A3C36....

```
 00.63970   enter the decimal fraction
+ 9.0       multiply 6 by 15 and add
+   45      multiply 3 by 15 and add
+   135     multiply 9 by 15 and add
+    105    multiply 7 by 15 and add
----------
 10.23520   spillover is 10, so first hexadecimal digit is A; so far, 0.A

 00.23520   clear spillover
+ 3.0       multiply 2 by 15 and add
+   45      multiply 3 by 15 and add
+    75     multiply 5 by 15 and add
+     30    multiply 2 by 15 and add
----------
 03.76320   spillover is 3, so next hexadecimal digit is 3; so far, 0.A3

 00.76320   clear spillover
+10.5       multiply 7 by 15 and add
+   90      multiply 6 by 15 and add
+    45     multiply 3 by 15 and add
+     30    multiply 2 by 15 and add
----------
 12.21120   spillover is 12, so next hexadecimal digit is C; so far, 0.A3C

 00.21120   clear spillover
+ 3.0       multiply 2 by 15 and add
+   15      multiply 1 by 15 and add
+    15     multiply 1 by 15 and add
+     30    multiply 2 by 15 and add
----------
 03.37920   spillover is 3, next hexadecimal digit is 3; so far, 0.A3C3

 00.37920   clear spillover
+ 4.5       multiply 3 by 15 and add
+ 1.05      multiply 7 by 15 and add
+   135     multiply 9 by 15 and add
+     30    multiply 2 by 15 and add
----------
 06.06720   spillover is 6, next hexadecimal digit is 6; so far, 0.A3C36
```

and this can be continued as far as you like.

Conversion of Decimal Whole Numbers to Hexadecimal

To use a method similar to the above, you would need to use a suan pan and perform the computations in hexadecimal. There is a simpler method which can be used on the soroban as well as suan pan, in which the calculations are done in binary, then converted to hexadecimal (http://webhome.idirect.com/~totton/soroban/Marcos/Decimal%20to%20Binary.htm). To use this method, you only need to know the binary codes for the digits '0' through 'F':

0 = 0000
1 = 0001
2 = 0010
3 = 0011
4 = 0100
5 = 0101
6 = 0110
7 = 0111
8 = 1000
9 = 1001
A = 1010
B = 1011
C = 1100
D = 1101
E = 1110
F = 1111

Clear the abacus and choose a units rod near the right end. Enter the binary code for the most significant decimal digit, then:

a) multiply the binary number by 'A' as follows: working from left to right, for each binary bit which is a '1', set a '1' at the left neighbor rod and another '1' two rods left of that, then clear the original bit.

b) add the binary code for the next most significant decimal digit

c) working from right to left, normalize the binary number by clearing each pair of bits on a rod and setting a bit on its left neighbor rod. For example, if three rods have 002, they will change to 010; 003 will change to 011; 012 will change to 020, then to 100. The object is to end up with only '0' or '1' on every rod.

d) If the last digit added was the least significant digit, you are done, else go to a)

Example: 32059 → 7D3B

0000 0000 0000 0000 clear rods (units rod on right)

0000 0000 0000 0011 enter binary code for 3

0000 0000 0001 0101 multiply leftmost 1 by A

0000 0000 0001 1110 multiply rightmost 1 by A

0000 0000 0001 1120 add binary code for 2 (0010)

0000 0000 0010 0000 normalize

0000 0001 0100 0000 multiply by A

```
0000 0001 0100 0000     add code for 0 (normalization not needed afterward)

0000 1020 1000 0000     multiply by A

0000 1020 1000 0101     add binary for 5 (0101)

0000 1100 1000 0101     normalize

0111 1101 0010 2010     multiply by A

0111 1101 0010 3011     add binary for 9 (1001)

0111 1101 0011 1011     normalize

   7    D    3    B     convert to hexadecimal
```

Here's a variation of the same method which uses a binary/octal code to represent each hexadecimal digit, and like the previous method, works on a soroban as well as suan pan. In this method, each hexadecimal digit occupies three rods: the leftmost is always zero and just serves as a spacer between digits, the center one is a binary rod with a zero if the hexadecimal digit is less than eight or one if it is equal to or greater than eight, and the rightmost rod contains an octal code for the hexadecimal digit's excess over zero or eight. On a soroban, it is convenient to always place the rightmost (octal) rod of each digit on a marked units rod.

Hexadecimal digit	binary rod	octal rod
0	0	0
1	0	1
2	0	2
3	0	3
4	0	4
5	0	5
6	0	6
7	0	7
8	1	0
9	1	1
A	1	2
B	1	3
C	1	4
D	1	5
E	1	6
F	1	7

The octal rod should never contain a number larger than 7 (I prefer to count the heaven bead as four rather than five and to only use three earth beads, to simplify the addition). The binary rod will sometimes contain a number greater than one temporarily, but will always be normalized back to either zero or one by clearing a pair of beads on the binary rods and incrementing the octal rod of the left neighbor hexadecimal digit.

This method requires the use of a table of hexadecimal multiples of nine (we will be using multifactorial multiplication to effectively multiply by A by adding nine times a digit to the digit).

Hexadecimal digit	Binary/octal code	digit x 9	Binary/octal code for x9
1	001	09	000 011
2	002	12	001 002
3	003	1B	001 013
4	004	24	002 004
5	005	2D	002 015
6	006	36	003 006
7	007	3F	003 017
8	010	48	004 010
9	011	51	005 001
A	012	5A	005 012
B	013	63	006 003
C	014	6C	006 014
D	015	75	007 005
E	016	7E	007 016
F	017	87	010 007

To rework the previous example by this method: 32059 → 7D3B

```
  000 000 000 000       clear some rods for the hexadecimal digits
+             003       add binary/octal code for 3
  ---------------
  000 000 000 003
+         001 013       multiply 3 by 9 using table and add
  ---------------
  000 000 001 016
+             002       add binary/octal code for 2
  ---------------
  000 000 001 020       octal addition for the octal rod - carry to the binary rod

  000 000 002 000       normalize: two on the binary rod = 1 on the left neighbor
+     001 002 000       multiply 2 by 9 using table and add
  ---------------
  000 001 004 000
+             000       add zero
  ---------------
  000 001 004 000       normalization not required
+ 000 011               multiply 1 by 9 using table and add
+     002 004           multiply 4 by 9 using table and add
  ---------------
  000 014 010 000
+             005       add binary/octal code for 5
  ---------------
  000 014 010 005       normalization not required
+ 006 014               multiply C by 9 using table and add
+     004 010           multiply 8 by 9 using table and add
+         002 015       multiply 5 by 9 using table and add
  ---------------
  006 034 022 022
+             011       add binary/octal code for 9
  ---------------
  006 034 022 033

  007 015 003 013       normalize
    7   D   3   B       the hexadecimal equivalent
```

Conversion of Hexadecimal Fractions to Decimal Fractions

To avoid performing computations in hexadecimal, this method begins with the hexadecimal fraction converted to binary, then involves successive multiplications by 'A' (1010 in binary) as in a previously shown method, and after each multiplication, taking the binary spillover left of the decimal point (converted to decimal) as the next decimal digit of the decimal fraction. The spillover is then cleared before the next multiplication.

Example: 0.A3C36 → 0.6397 approximately

0000.1010 0011 1100 0011 0110	convert the hexadecimal to binary
0102.0101 1221 1001 1121 1100	multiply by A
0110.0110 0101 1010 0001 1100	normalize - the first decimal digit is 6
0000.0110 0101 1010 0001 1100	clear the spillover
0011.1110 2112 0100 1121 1000	multiply by A
0011.1111 1000 0101 0001 1000	normalize - the next decimal digit is 3
0000.1111 1000 0101 0001 1000	clear the spillover
0112.2211 0010 2010 1111 0000	multiply by A
1001.1011 0011 0010 1111 0000	normalize - the next decimal digit is 9
0000.1011 0011 0010 1111 0000	clear the spillover
0102.1111 1111 0212 2110 0000	multiply by A
0110.1111 1111 1101 0110 0000	normalize - the next decimal digit is 6
0000.1111 1111 1101 0110 0000	clear the spillover
0112.2222 2221 2021 1100 0000	multiply by A
1001.1111 1110 0101 1100 0000	normalize - the next decimal digit is 9

so far, we have 0.63969 or approximately 0.6397

The previous use of a binary/octal code to represent the hexadecimal digits can also be used to convert a hexadecimal fraction to a decimal fraction. This method also works on the soroban and does not require hexadecimal addition.

To rework the previous example by this method: 0.A3C36 → 0.6397 approximately

```
  000.012 003 014 003 006          convert the hexadecimal to binary/octal code
+ 005 012                          multiply A by 9 using table and add
+     001 013                         "     3 "     "     "     "
+         006 014                     "     C  "     "     "     "
+             001 013                 "     3  "     "     "     "
+                 003 006             "     6  "     "     "     "
  ------------------------
  005.025 024 031 021 014

  006.006 005 012 001 014          normalize - the first decimal digit is 6

  000.006 005 012 001 014          clear the spillover
+ 003 006                          multiply 6 by 9 using table and add
+     002 015                         "     5  "     "     "     "
+         005 012                     "     A  "     "     "     "
+             000 011                 "     1  "     "     "     "
+                 006 014             "     C  "     "     "     "
  ------------------------
  003.016 027 024 020 030

  003.017 010 005 001 010          normalize - the next decimal digit is 3

  000.017 010 005 001 010          clear the spillover
+ 010 007                          multiply F by 9 using table and add
+     004 010                         "     8  "     "     "     "
+         002 015                     "     5  "     "     "     "
+             000 011                 "     1  "     "     "     "
+                 004 010             "     8  "     "     "     "
  ------------------------
  010.032 022 022 016 020

  011.013 003 002 017 000          normalize - the next decimal digit is 9

  000.013 003 002 017 000          clear the spillover
+ 006 003                          multiply B by 9 using table and add
+     001 013                         "     3  "     "     "     "
+         001 002                     "     2  "     "     "     "
+             010 007                 "     F  "     "     "     "
  ------------------------
  006.017 017 014 026 000

  006.017 017 015 006 000          normalize - the next decimal digit is 6

  000.017 017 015 006 000          clear the spillover
+ 010 007                          multiply F by 9 using table and add
+     010 007                         "     F  "     "     "     "
+         007 005                     "     D  "     "     "     "
+             003 006                 "     6  "     "     "     "
--------------------------
```

```
  010.036 035 025 014 000

  011.017 016 005 014 000          normalize, the next decimal digit is 9
```

So far, we have 0.63969 or approximately 0.6397 just as before.

Steve Treadwell
December, 2014