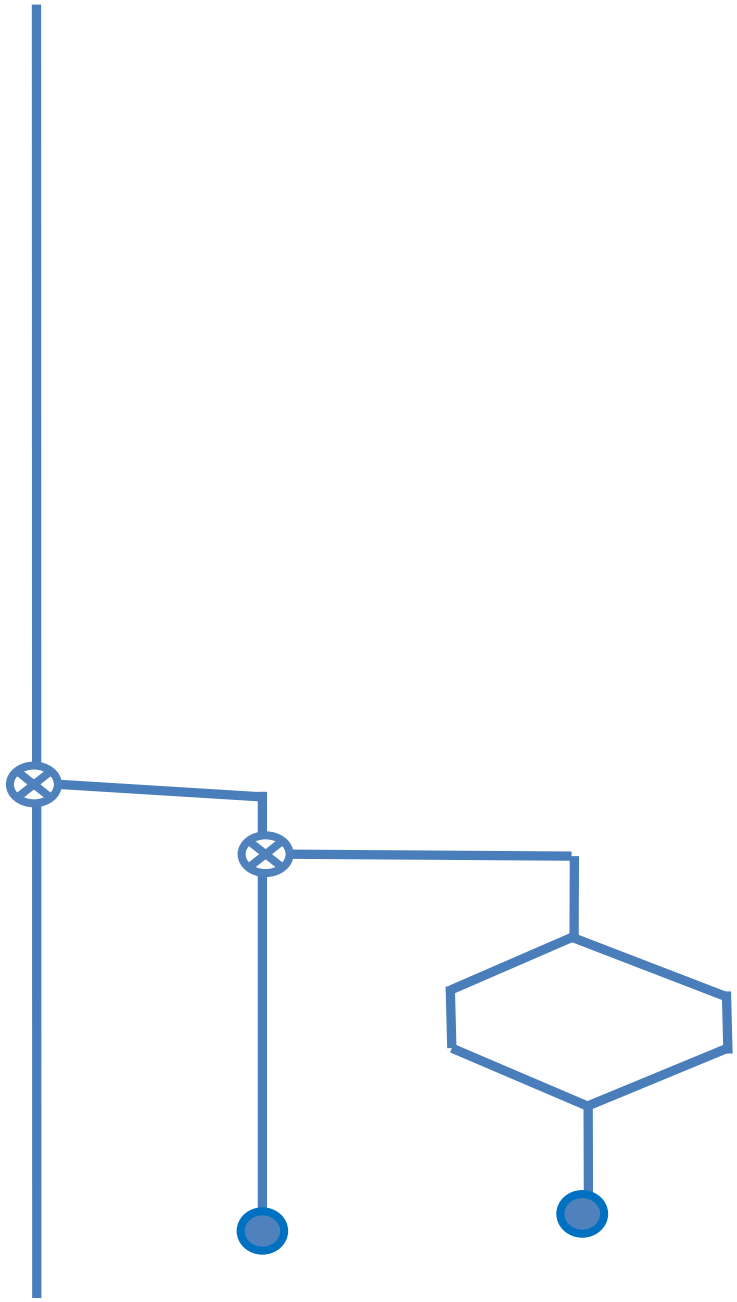


Examples Of Floating Point Conversions

Forth2020

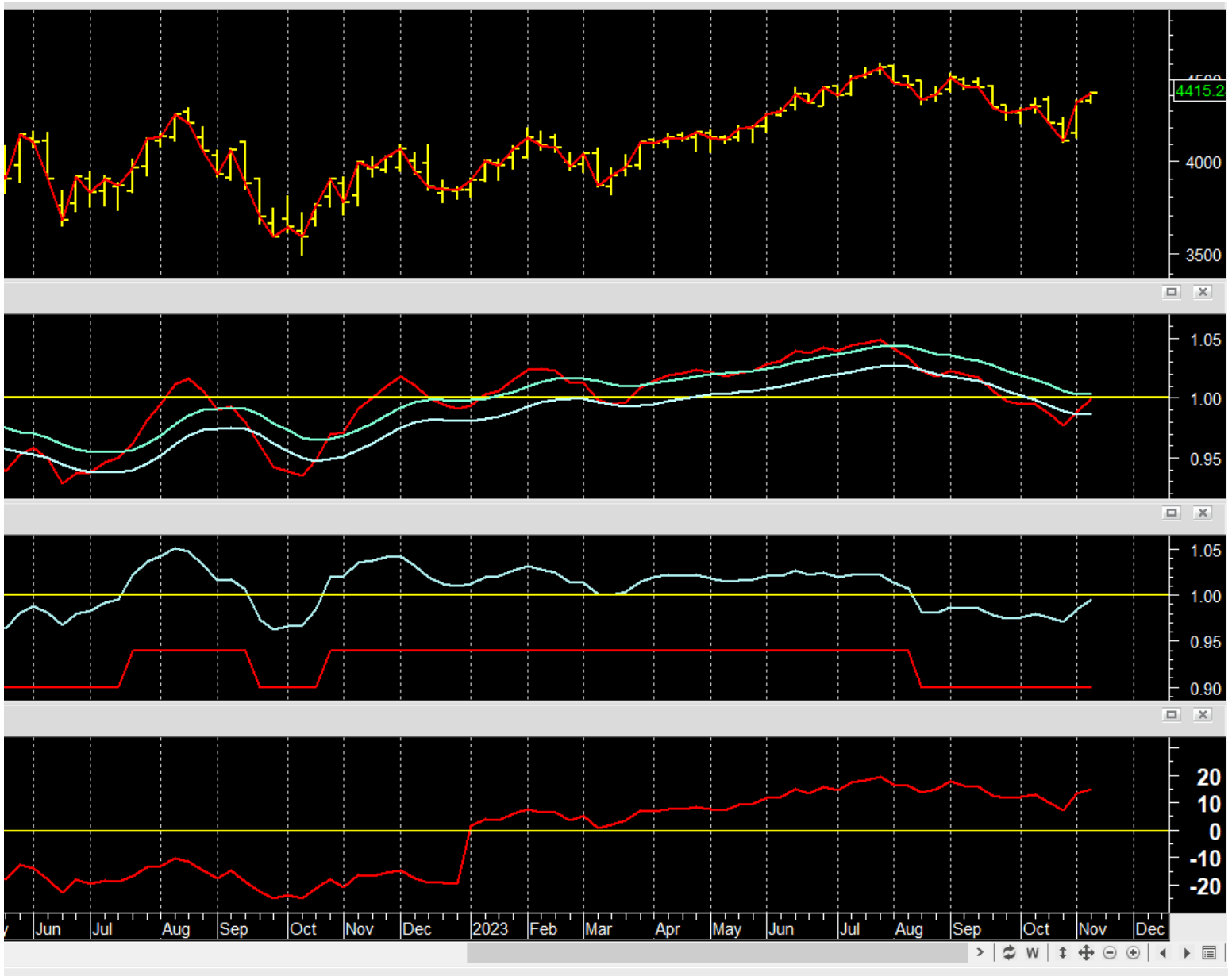
Dec. 9, 2023

Bill Ragsdale

The Background

- In 1984 ComputTrac introduced a stock market analysis system.
- It sold for $2,000 or about $8,000 today.
- Over the years, its data format was adopted by most other consumer level stock market programs.
- It used a MicroSoft 32 bit floating-point format.

Example


4415.2
4000
3500
1.05
1.00
0.95
1.05
1.00
0.95
0.90
20
10
0
-10
-20
Jun
Jul
Aug
Sep
Oct
Nov
Dec
2023
Feb
Mar
Apr
May
Jun
Jul
Aug
Sep
Oct
Nov
Dec

More

The most popular system to day is MetaStock, still with the 1984 format, for data compatibility.

I've attempted to use the files with Systat, MathLab and R.

Never got too far but have written several floating point conversions.

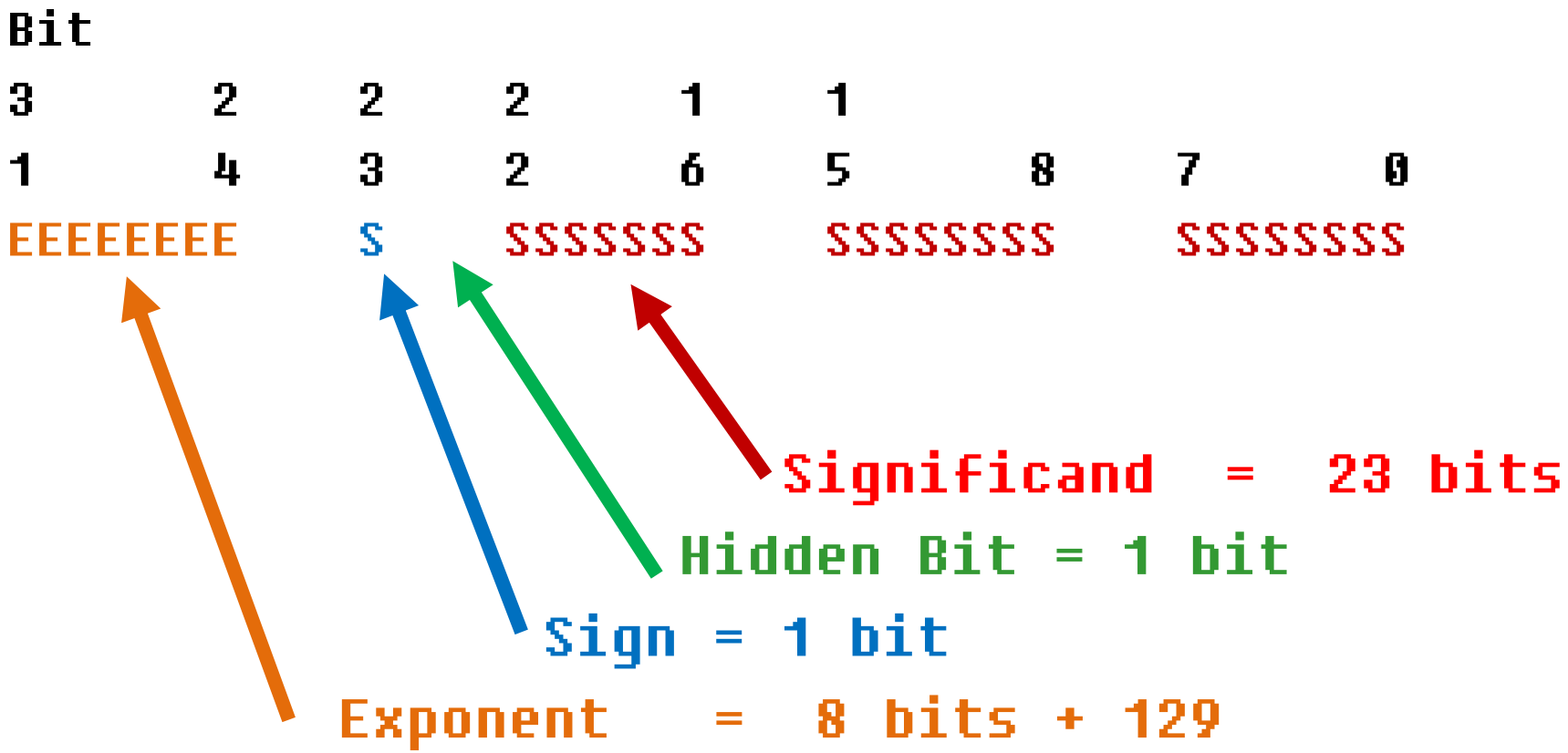
Now it is Win32Forth's turn.

Binary Display

```
: B. ( n -- ) \ A diagnostic binary print.
 cr ."     3  2    2    2    1    1    0    0    0 "
 cr ."     1  8    4    0    6    2    8    4    0 "
 base @ binary    cr 6 spaces
 swap s>d  <#  8 0 do  # # # # bl hold loop #> type
 base !  ;
```

```
   3  2    2    2    1    1    0    0    0
   1  8    4    0    6    2    8    4    0
   0000 0000 0000 0000 0000 0000 0000 0000 ok
```

Q-Basic MS Float Format



MicroSoft to IEEE

1. Extract low 23 bits over 0..22.
2. OR in a 1 as the 24th bit at 23.
3. Pass to floating point stack.
4. Shift decimal point 23 bits to the left.
5. Extract exponent, 0xFF000000 AND.
6. Right shift by 24 bits.
7. Subtract bias of 129.
8. Raise to the power of the exponent by f**.
9. Extract the sign and apply to the final float.

MicroSoft to IEEE.

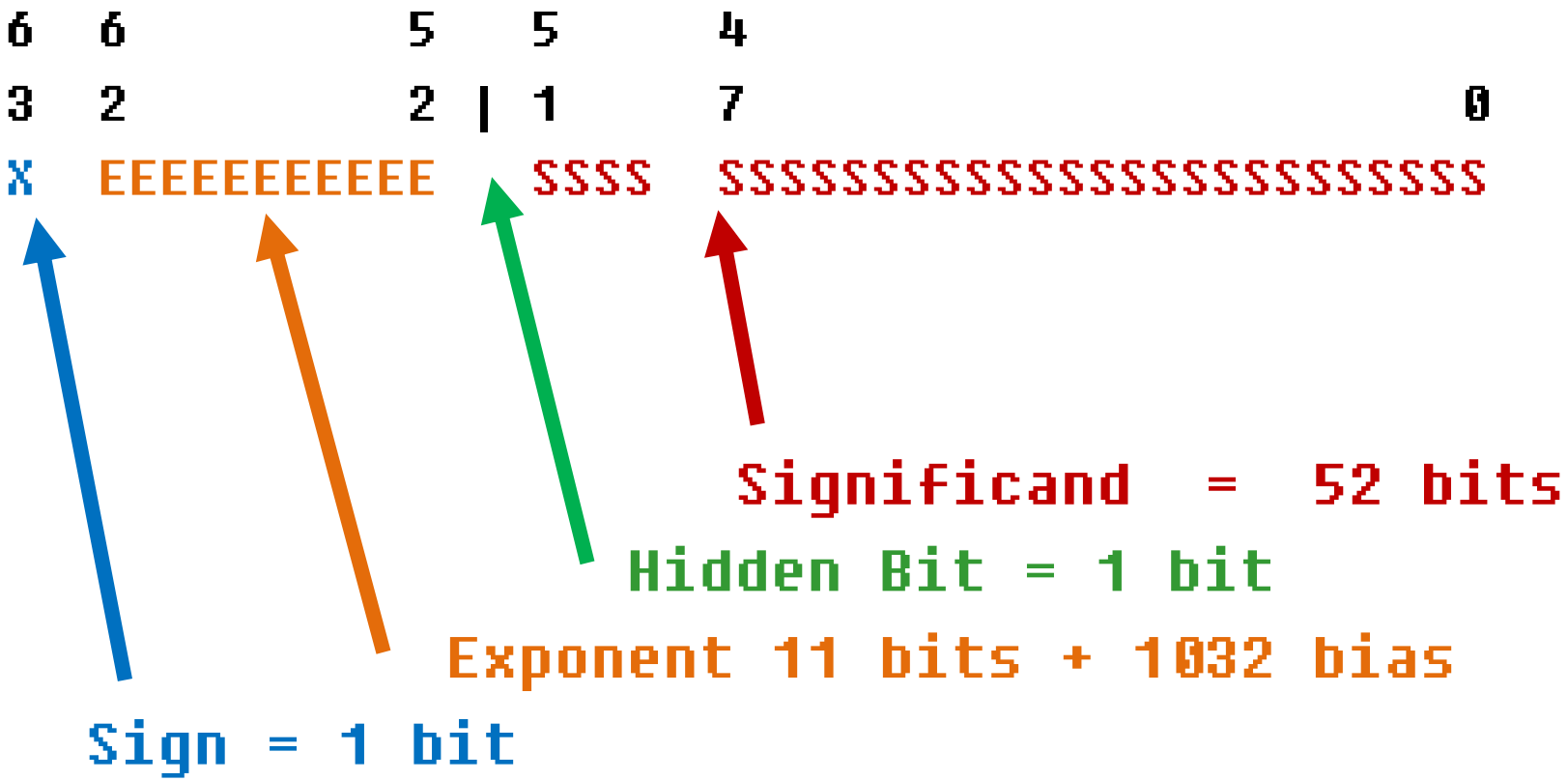
```
: MS>IEEE  ( n -- ) ( fs: -- n ) \ MicroSoft to IEEE
 dup
 0x007FFFFF and 0x00800000 or s>f     \ significand
 2e 23e f** f/                    \ 23 bit right shift
 dup 0xFF000000 and 24 rshift 129 -      \ exponent
 2e s>f f**  f*          \ scale by the float exponent
 0x00800000 and          \ test the sign
   if fnegate then  ; \ adjust sign?
```

Examples MS>IEEE

```
0x85780000 MS>IEEE cr f. .( 31.00   )
    and see:  31.00000 31.00 ok

0x951628A8 MS>IEEE cr f. .( 1230101 )
   and see:  1230101.  1230101  ok  ( a date)
```

IEEE Format


6 6 5 5 4
3 2 2 | 1 7 0
X EEEEEEEEEEE SSSS SSSSSSSSSSSSSSSSSSSSSSSS
Significand = 52 bits
Hidden Bit = 1 bit
Exponent 11 bits + 1032 bias
Sign = 1 bit

IEEE Pseudocode

1. If zero, force zero as final float and exit.
2. Copy float image as two 32 bit values to data stack.
3. Shift the low 32 bits, 29 bits to the right filling with zeros.
4. Swap to access high 32 bits.
5. 0x7FF00000 AND to extract exponent.
6. 20 bit right shift to bring exponent for calculation.
7. Subtract 1032 bias add 129 bias.
8. Shift this exponent 24 bits left.
9. Extract sign and shift 8 bits to the right.
10. Extract low 20 bits.
11. Shift 3 bits left to make room for the low 3 bits.
12. OR significand and OR the sign to finish.

IEEE to MicroSoft

```
: IEEE>MS ( f: f -- ) ( -- ms_float ) \ IEEE to 32 bit MS
   fdup f0= if fdrop 0 exit then      \ quick exit on zero
   FS>DS                \ float to two 32 bit stack values
   29 rshift            \ low 3 bits to left, filling with zero
   swap dup 0x7FF00000 and   \ extract exponent
   20 rshift 1023 - 129 +    \ translate the exponent
   24 lshift                 \ move exponent high
   over 0x80000000 and  8 rshift or  \ sign bit copied
   swap 0x000FFFFF and 3 lshif  t    \ significand positioned
   or  or    ;          \ combine exponent, sign & significand
```

Examples IEEE>MS>IEEE

```
.000001e  fdup IEEE>MS MS>IEEE cr f. f.
   and see:  .0000010 .0000010

-0.12345e fdup IEEE>MS MS>IEEE cr f. f.
   and see: -.1234500 -.1234500
```

Data Example

	Date	High	Low	Close	Volume
1800	6/9/2023	149.720	147.370	149.460	0
1801	6/16/2023	155.470	151.840	154.440	0
1802	6/23/2023	154.440	152.710	152.960	0
1803	6/30/2023	156.420	151.320	156.420	0
1804	7/7/2023	156.860	155.300	155.460	0

```
   Date    High    Low     Close   Vol
23 0609 149.72 147.36 149.46    0
23 0616 155.47 151.83 154.44    0
23 0623 154.44 152.71 152.96    0
23 0630 156.41 151.32 156.41    0
23 0707 156.86 155.30 155.46    0   ok
```

Float To Scaled 32-Bit Integer

```
: IEEE>Scaled   ( fs: f -- ) ( n1 -- n2  )
   \ 64 bit IEEE to a scaled integer
   \  n1 sets the decade scaling.
   10e s>f  f** f*  f>s ;


1234.567e 2 ieee>scaled .( Scaled see 123456 ) .
   and see:  123456  ok   \ scaled by 100.
```

This might be useful for analysis using Forth's integer support.

Observations

As for most problems, the solutions looked simpler after they were done.

For future stock market analysis, it appears I'll need the floating point for exponentials and integer for bit logic, MOD and scaling /MOD.

Next time, I'll address file access and reporting.

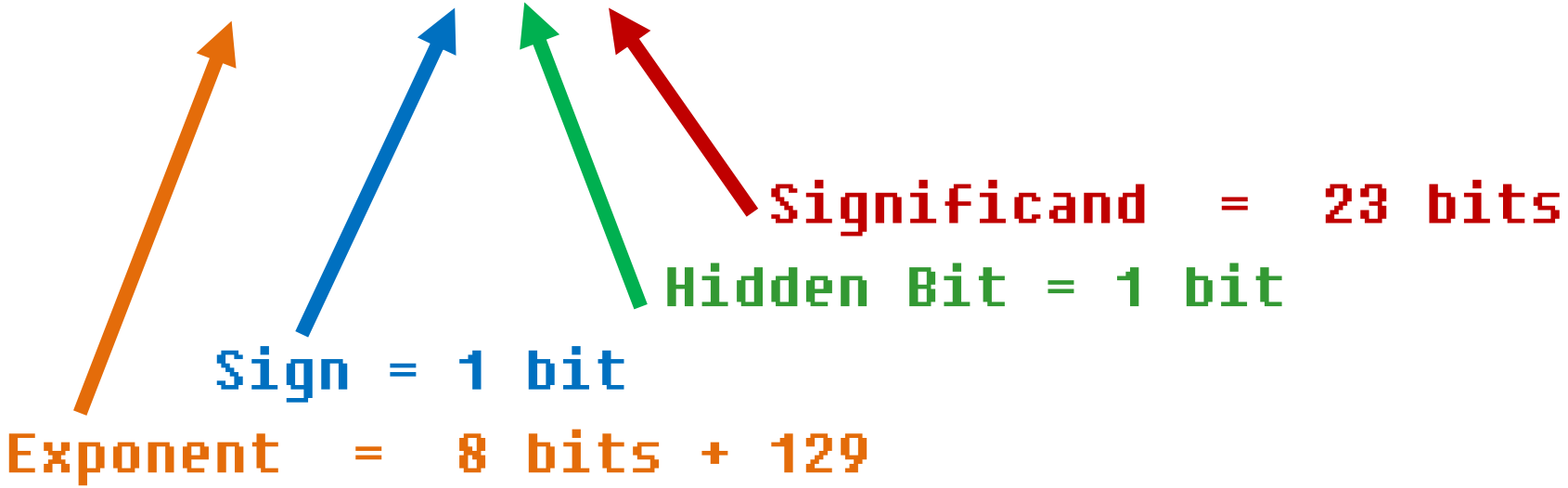
References

- https://github/BillRagsdale/
- https://www.pcjs.org/documents/books/mspl13/basic/qblang/ Section 2.2.1.2 Floats formats.

Binary Of MS Float

```
15e  ieee>ms  b.

    3  2     2  2     2     1     1     0     0     0
    1  8     4  3     0     6     2     8     4     0
    1000 0100  0  111 0000 0000 0000 0000 0000 0000 ok
```



Win32Forth Cosmology

The Complete Forth Textbook