

IMPROVEMENT OF UNUM COMPUTING SOFTWARE

STUDENT: Sidong Feng

SUPERVISOR: Josh Milthorpe



```
double a = 2.0

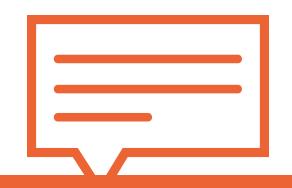
for i = 0 to 100

a = Math.sqrt(a)

end for

---> a = 1.0
```

IEEE STANDARD FOR FLOATING -POINT ARITHMETIC



Rounding Rule: numbers are rounded to the nearest floating-point number

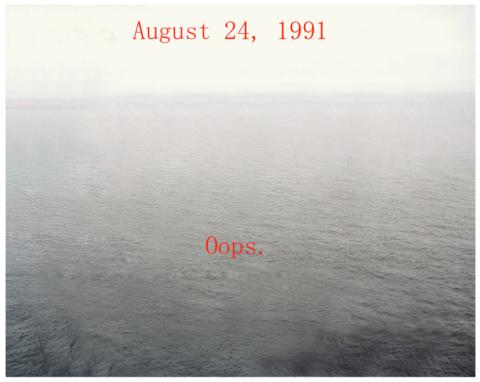
Example:

 $\sqrt{2} = 1.0$ if we round to 1 decimal

SLEIPNER OIL PLATFORM DISASTER

Collapsed to ocean floor; Float error in structural analysis





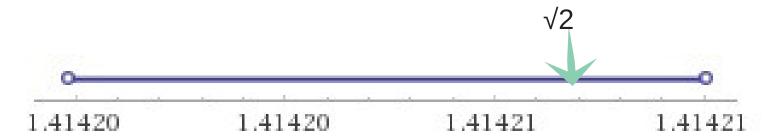


For example: Represent √2

Float: 1.4142135623730951

Unum: (1.4141998291015625,1.414215087890625)

Unum format: 0 01 011010100001001 1 001 1111



Unum Library by LLNL (Lawrence Livermore National Laboratory)

unum.h
unum
ubnd
unum_set()
unum_set_ui()
unum_set_si()
unum_set_d()
unum_set_str()
unum_get_si()

unum_get_si() unum_get_d() unum_get_str()

unum_nbits()

unum_cmp()

unum_add() unum_sub() unum_mul() unum_div()

unum_pow()

unum_sq() unum_sqrt() unum_neg() unum_abs() unum_guess()

Transcendental Fused ComplexFFT

ubnd *()

unum.c

unumxx.h

hlayer.h scan_*() print_*() uview_*() view_uenv()

hlaver.c

ulayer.h
unum_t
ubnd_t
unum_init()
ubnd_init()
unum_clear()
ubnd_clear()
ulayer.c

glayer.h
gnum_t
gbnd_t
gnum_init()
gbnd_init()
gnum_clear()
gbnd_clear()

uenv.h
MAX_*SIZE
variables
init_uenv()
set_uenv()

mpx.h MPX_VAR mpx_t mpx_*() gmp.h mpn_*() mpz_*() mpf_*() gmp.c

gmp_macro.h

gmp_aux.h mpn_*shift() mpn_*bit() mp*_import_b() mp*_export_b()

gmp aux.c

Unum Library Module Map, Future Work in Red

ubnd.h ltuQ() atuQ() neguQ() nneauQ() sameuQ() spanszerouQ() intersectuQ() plusu() minusu() timesu() divideu() powu() squareu() sartu() negateu() absu() expu() logu() cosu() sinu() tanu() cotu()

ubnd.c

conv.h *2q() - si,ui,d,f q2*() *2un() - si,ui,d un2*() *2ub() - si.ui.d ub2*() u2f() f2u() unum2q() ubnd2q() u2q() q2u() unify() smartunify() quessu()

conv.c

support.h
utag_t

utag()
signmask()
bigu()

scale()
ne()
inexQ()
infuQ()
nanuQ()

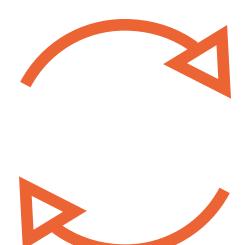
promotef()
promotee()
promote()
demotee()
demotee()

support.c

gbnd.h ItqQ() ataQ() neggQ() nnegaQ() samegQ() spanszerogQ() intersectgQ() plusq() minusq() timesa() dividea() powg() squareq() sqrtg() negateg() absq() expg() logg() cosq() sing() tang() cotq() gbnd.c

PROJECT GOAL

- Implement more arithmetic operations
- Improve performance



PLAN

- Understanding Unum
- Understanding Library
- Implementing Operations
- Debug and Analysis

- Implementing efficient
 operations and calculating
 - with no error !!!

BONUS

Unum format:



$$x = (-1)^{s} \times \begin{cases} 2^{2-2^{es-1}} \times \left(\frac{f}{2^{fs}}\right) & \text{if } e = \text{all 0 bits,} \\ \infty & \text{if } e, f, es, \text{ and } fs \text{ have all their bits set to 1,} \\ 2^{1+e-2^{es-1}} \times \left(1 + \frac{f}{2^{fs}}\right) & \text{otherwise.} \end{cases}$$

Unum format: 0 1 0 0 0 0 = 2

BONUS



UNUM vs FLOAT

- Easier to use
- Less demanding on memory and bandwidth

For example: Represent 2

Unum format: 0 1 0 0 0 0