The libieee754 compliance library for the IEEE 754-2008 standard

Olga Kupriianova – Christoph Lauter

Équipe PEQUAN, LIP6, UPMC Paris 6
okupri@gmail.com – christoph.lauter@lip6.fr

SCAN 2012, Novosibirsk, Sept. 24th, 2012
Outline of the talk

New Features

Algorithmical details

Future work

Conclusion
Floating point arithmetics was standardized by IEEE 754
Standardized Interval Arithmetic can easily be based on fully compliant IEEE 754
New standard in 2008
No full support by C99, GNU/Linux, compilers
Floating point arithmetics was standardized by IEEE 754
Standardized Interval Arithmetic can easily be based on fully compliant IEEE 754
New standard in 2008
No full support by C99, GNU/Linux, compilers
Only a closed-source library provided by Intel
Floating point arithmetics was standardized by IEEE 754

Standardized Interval Arithmetic can easily be based on fully compliant IEEE 754

New standard in 2008

No full support by C99, GNU/Linux, compilers

Only a closed-source library provided by Intel

Our library is an open-source library and brings the full support for IEEE 754-2008
IEEE 754-1985 in the base + some new features

- FMA: multiplication + addition with only one rounding
- Heterogeneous operations
- Correctly rounded conversion from binary to decimal string and vice versa with support for all rounding modes
- Decimal FP arithmetic
- Recommended part: correctly rounded elementary functions
e.g. $c = \circ_k(a + b)$, where $\circ_k(x)$ is round to nearest

- Computing yields $c$ with only one rounding for $a$, $b$, $c$ in different formats
  - e.g. $a$ in binary32,
  - $b$ in binary64,
  - $c$ in binary32
- IEEE 754-1985 forbids support for these operations
- IEEE 754-2008 requires them
Decimal string to binary conversion

- char* → binary64
- binary64 → char*

scanf/printf can do it
Decimal string to binary conversion

- char* → binary64
- binary64 → char*

`scanf/printf` can do it but in GNU libc

- it does only *round-to-nearest*
- the result is NOT always correctly rounded
- it *allocates* lots of memory
- it does not get the flags right
- it has problems with corner cases *e.g.* the least subnormal, the largest normal
Our library supports

- Correct rounding for any input length string
- All rounding modes
- All flags are set correctly
- No `malloc` usage $\Rightarrow$ memory consumption is known beforehand for arbitrary length strings
Decimal string to binary conversion in libieee754

User input $x$

$x = 0.12345678912345678912$

Read $r$ decimal digits to $10^E m$, $r$ is pretty large.
Set a flag if $10^E m$ is inexact

$m = 123456789123456789120...0$

Read $\bar{r}$ decimal digits to $10^E \bar{m}$.
$\bar{r}$ is small.

$\bar{m} = 12345678912345678912$

Produce the binary FP number $2^F n$
with $n$ on 64 bits and $2^F \bar{n}$

$n = 455475158624672704$
$\bar{n} = 17791998383768253$

Check if the rounding is easy:
$|n - 2^8 \bar{n}| > 2^{62} \cdot 2^{-56.75}$

Rounding is hard

Transform $2^E \bar{m}$ to $10^{E_1} m_1$
without any rounding

$m_1 = 12345678912345679073547799475818469981558229064941406250...0$

Compare $10^{E_1} m_1$ with $10^E m$
and produce indicator $\delta$

Rounding is easy

Perform final rounding according to $x_{53}$

Overflow and underflow

Subnormal rounding

Normal rounding

O. Kupriianova – Ch. Lauter

The libieee754 compliance library for the IEEE 754-2008 standard – SCAN 2012, Novosibirsk
libieee754 is currently based in a IEEE 754-1985-compliant underlying system:

- The IEEE 754-1985 FP operations are supported to be finally done in hardware
- Wrappers encapsulate operations where the hardware/system is not 100% compliant
- Almost no decimal hardware is available
- Support of the decimal IEEE 754-2008 FP arithmetic is more and more asked by users
The Work to Do

- Add possibility to compile libieee754 for systems that don’t have IEEE 754-1985 compliant hardware
- Add decimal arithmetics
- Emulate everything with integer operations
Future Work
The Recommended Part of the Standard

IEEE 754-2008 recommends (but does not mandate) support for

- alternative exception handling
- correctly rounded elementary functions
  - hard to achieve because of the so-called Table Maker’s Dilemma
  - very expensive precomputation of so-called worst-cases required
  - formal proofs and code generation required

libieee754 long-term goal
Library is reentrant

The main target: 100% correctness and completeness

Speed is reasonable but not fully optimized

Algorithms are fully proven on paper

All the 354 operations mandated by the standard for binary32, binary64:

- easy wrappers to map the operations directly to hardware
- libieee754 functions that call and use other libieee754 functions
- specialized algorithms that have been designed, proven and implemented with care
Conclusions

- `libieee754` supports all 354 operations required for both binary32 and binary64
- It is an open source library
- Novel algorithm for decimal string → binary conversion is provided
- 100% IEEE 754-2008 compliance with an easy-to-use interface
- Reasonably fast and getting better
- Fully proven, proofs are available on demand
Thank you for your attention!

Questions?