IEEE 754 SUPPORT FOR INTEL® ARCHITECTURE

Marius Cornea, Intel Corporation

ARITH 23, Santa Clara, CA

July 11, 2016

Intel® x87 and the IEEE Standard 754-1985

- The beginning: the Intel x87 floating-point unit (x87 FPU)
  - Intel worked with Prof. W. Kahan, U.C. Berkeley, on the Intel® 8087 Math Coprocessor; design started in 1976
- The IEEE Standard 754-1985 was in part a result of that collaboration (and then also IEEE 854-1987)
  - Basis for “C99 math” (C99 floating-point arithmetic ~= IEEE 754-1985 minus support for SNaN-s)
- IEEE 754-1985 conformance :
  - “All implementations conforming to this standard shall support the single format”
  - The standard defined two basic formats, and two extended formats – single and double
  - Four rounding modes, infinities, NaNs, signed zeros, and five exceptions
  - Operations: add, sub, mul, div, sqrt, rem, cmp, round to integer in floating-point (FP) format, conversions between FP formats, conversions between FP and integer formats, conversions between binary FP formats and decimal strings

Intel® x87 FPU Arithmetic

- IEEE 754-1985 conformance is ensured mostly in HW, as the x87 FPU implements almost everything from what is mandated by IEEE 754-1985
  - Single and double precision basic formats, the 80-bit double-extended format, and most operations
  - Separate FPU control and status words are used
  - Note: tininess (related to underflow) is detected 'after rounding'
- A few things were left to software, e.g.:
  - Some corner cases related to overflows/underflows, which could lead to different results compared to 'pure IEEE' computation; these could occur because the FPU stack computation is performed on 80-bit floating-point values, with 15-bit exponents (the significand can have 24, 53, or 64 bits).
    - Solution: always store results to memory (as 'true' single and double encodings exist only in memory); double rounding still possible on underflow
  - Some conversions, e.g. between floating-point formats and decimal strings

SIMD Floating-Point Arithmetic

- SIMD FP operations are now used for most floating-point computations in Intel® Architecture; a combined control and status word (MXCSR) is used; all operations follow the IEEE Standard 754 or its spirit
  - SSE (1999): 128-bit Single Precision floating-point (SP/binary32); MXCSR has flush-to-zero (FTZ) and denormals-are-zero (DAZ) flags
    ADD/SUB/MUL/DIV/SQRT/CMP/CVT/MIN/MAX/AND/OR/XOR/SHUF/MOV/…
  - SSE2 (2000): 128-bit Double Precision floating-point (DP/binary64)
    ADD/SUB/MUL/DIV/SQRT/CMP/CVT/MIN/MAX/AND/OR/XOR/SHUF/MOV/… [MIN/MAX different from IEEE 754-2008]
  - SSE3 (2004): 128-bit SP/DP ADDSUB, HADD, HSUB, MOV
  - SSE4.1 (2007): 128-bit SP/DP DP/ROUND/BLEND; SP INSERT/EXTRACT
  - AVX (2011): 256-bit extensions of SP/DP 128-bit instructions; new three-operand format; later added also support for IEEE binary16 (2012); CMP instructions were enhanced to support more ordered/unordered, signaling/non-signaling comparisons
  - AVX2 (2013): 256-bit SP/DP FMA, three-operand (conforming to IEEE 754-2008)
  - AVX-512 (2016): 512-bit extensions of the 256-bit FP instructions; AVX-512 Exponential and Reciprocal Instructions (ER); VRANGE improves IEEE 754 conformance in HW – computes min, max, minabs, or maxabs values according to the definitions from IEEE Standard 754-2008; static rounding modes and suppress-all-exceptions mode; conversion instructions to/from unsigned integers, and to 64-bit integer formats

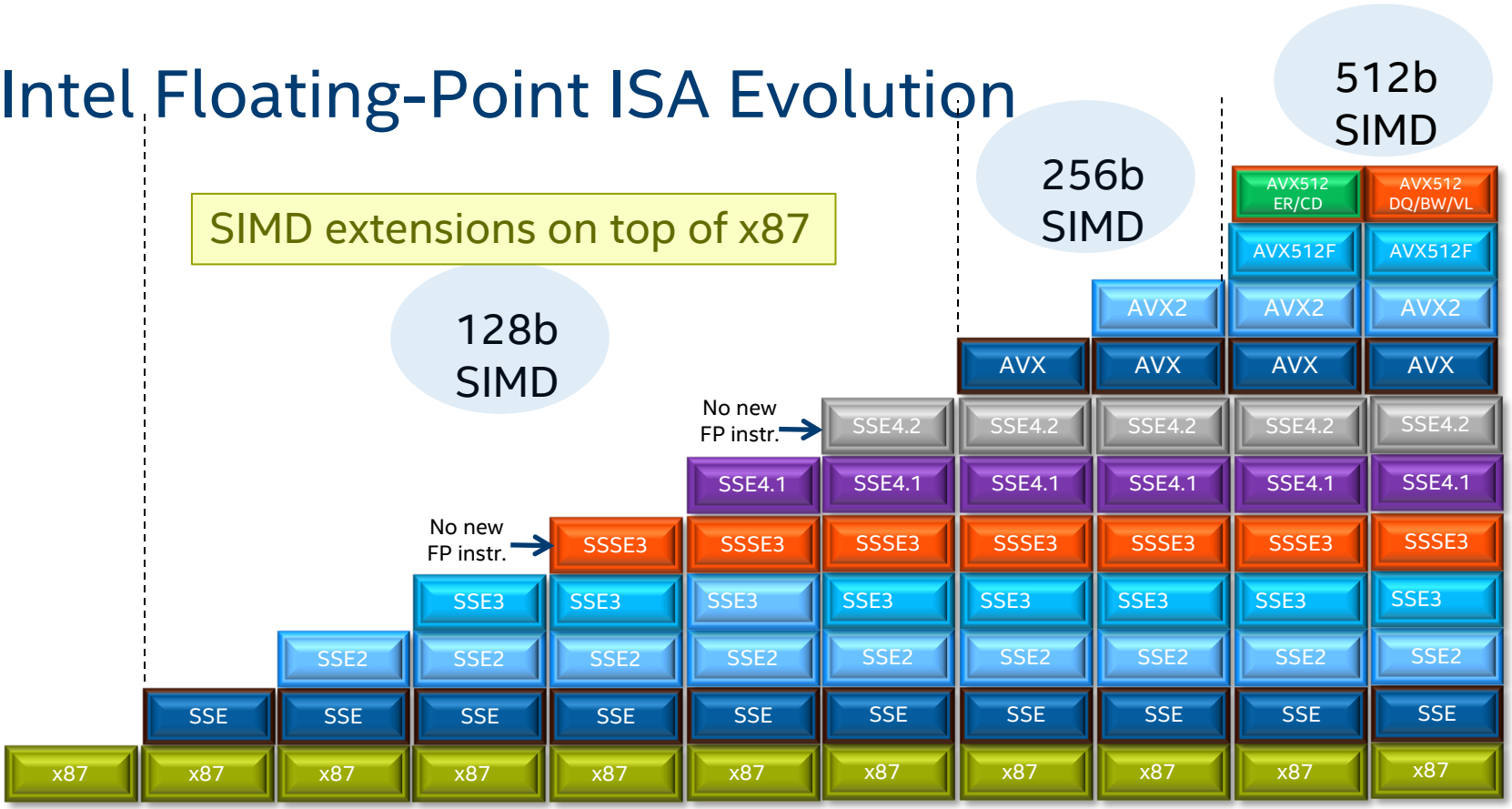
Intel Floating-Point ISA Evolution

SIMD extensions on top of x87

128b SIMD

256b SIMD

512b SIMD



From Intel 8087 to PII (8087, 1980 to Klamath, 1997)

PIII (Katmai, 1999)

P4 (Willamette, 2000)

P4 (Prescott, 2004)

Core (Merom, 2006)

Core (Penryn, 2007)

Core (Nehalem, 2008)

Core (Sandy Bridge, 2011)

Core (Haswell, 2013)

Xeon Phi™ (Knights Landing, 2016)

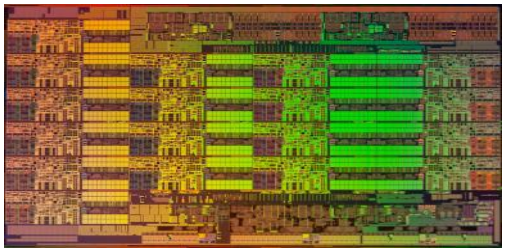
Core (Sky Lake)

Most Recent Intel® Xeon and Xeon Phi™ Platforms

Xeon*

Broadwell (14nm process)

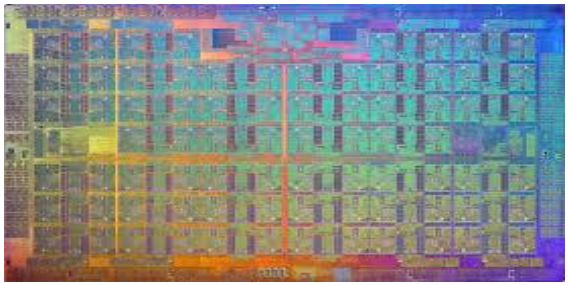
- Up to 22 cores; hyper-threading
- GPU DP/SP/FP16 operations also conform to IEEE 754-2008
- ~66 GB/s stream memory BW (4 ch. DDR4 2400)
- AVX2 – 256-bit (4 DP, 8 SP flops) → > 0.7 TFLOPS (SP)
- 20 PCIe lanes



Xeon Phi*

Knights Landing (14nm process),

- Optimized for highly parallelized compute-intensive workloads
- Common programming model & S/W tools w/ Xeon processors, enabling efficient application readiness and performance tuning
- Up to 72 cores, 490 GB/s stream BW, on-die 2D mesh
- AVX512– 512-bit (8 DP, 16 SP flops) → >3 TFLOPS (DP)
- 36 PCIe lanes
- Division and square root (and integer division) implemented as ucode sequences



*Intel Xeon and Intel Xeon Phi are trademarks of Intel Corporation in the USA and/or other countries.

IEEE 754-2008: What Needs to Be Supported?

- IEEE 754-2008: Conformance to this standard is a property of a specific implementation of a specific programming environment
- A programming environment conforms to this standard, in a particular radix, by **implement**ing **one or more of the basic formats** of that radix **as both a supported arithmetic format and a supported interchange format**
  - Must be able to initialize all supported formats, and convert between them
  - Must be able to read/write any supported interchange formats, using a specified encoding
  - Operations from Clause 5 must be available for supported arithmetic formats
- Very few programming environments today supporting more than one format conform to IEEE 754-2008
- The gap between the C Standard and the IEEE 754-2008 Standard is just being closed/narrowed now w/ ISO/IEC TS 18661, *Floating-point extensions for C*

What Has Changed in IEEE 754-2008

- Some requirements are a little harder; some new operations were added for binary; decimal arithmetic was added
- New: fused multiply-add (FMA)
- New storage formats: binary16, decimal32
- New heterogeneous operations:
  - Correct rounding even if operands' and result's precisions are different; only affects rounding-to-nearest
  - Example: single + double -> single with only one rounding
- New conversions to integer, independent of the global rounding mode
- New min/max operations, totalOrder predicate
- New signaling equal/not equal comparisons
- Scaled intermediate result delivered to trap handler are not required anymore (this was optional in IEEE 754-1985)
- Correctly rounded decimal character conversions
- Optional for binary floating-point: round-to-nearest-ties-to-away rounding mode

Intel® Architecture Support for IEEE 754-2008 Binary Floating-Point Arithmetic

- Most mandatory features are implemented in Intel® Architecture HW
- Not supported in HW: heterogeneous operations, some conversions (e.g. correctly rounded conversions from and to decimal), totalOrder predicate, some comparisons
- Must track two sets of control and status registers (x87 FPU and MXCSR)
- SW support added for binary32 and binary64 in the Intel® IEEE 754-2008 Binary Floating-Point Compatibility Library, libbfp754, provided with the Intel® C/C++ Compiler
  - 315 C functions, one for each mandated IEEE 754-2008 operation not supported in HW – complete list at https://software.intel.com/en-us/node/583391
  - Call-by-reference or call-by-value; supports rounding modes; flags reside in HW control status registers, but state can be a function argument
  - Example: icc -fp-model source -fp-model except my_application.c –lbfp754
- Can be used with Intel® Xeon, Intel® Xeon Phi™, Intel® Atom™, Intel® Quark™ processors

Intel® Architecture Support for IEEE 754-2008 Decimal Floating-Point

- Supported in SW, by using the *Intel® IEEE 754-2008 Decimal Floating-Point Math Library*
- Implements everything that is mandated in the standard for decimal floating-point
  - Open source version posted w/ BSD license at http://software.intel.com/en-us/articles/intel-decimal-floating-point-math-library/ and http://www.netlib.org/misc/intel
  - Makes decimal floating-point arithmetic available on all Intel platforms
  - Integrated in Intel Compiler 11.0 and newer (and other third party SW)
  - Conforms also to two C/C++ TRs: ISO/IEC TR 24732 - *Extension for the programming language C to support decimal floating-point arithmetic*; and ISO/IEC TR 24733:2011 - *Extensions for the programming language C++ to support decimal floating-point arithmetic*
  - Includes also transcendental math functions (not correctly rounded)
  - Note: tininess is detected 'before rounding' in the Intel Compiler (it is detected 'after rounding' for binary floating-point in Intel® Architecture); the open source library can detect tininess either way
- Can be used with Intel® Xeon, Intel® Xeon Phi™, Intel® Atom™, Intel® Quark™ processors

Reproducibility on Processors Supporting IEEE 754

- The Intel Compiler passes the IEEE 754-1985 test suite with /Qftz- /fp:source
- Why are there differences between results on different processors that all support IEEE 754?
  - Parallel computations, e.g. reordering operations at compile time
  - Use of FMA vs. MUL+ADD
  - Different implementations for some instructions, e.g. for reciprocal approximations or mathematical functions
- Intel Compiler options control the tradeoffs between accuracy, reproducibility and performance
- To improve consistency and reproducibility of FP results while limiting the impact on performance, use
  /fp:precise /fp:source (Windows*) or -fp-model precise -fp-model source (Linux* or OS X*)
- If reproducibility between different processor types of the same architecture is important when using the same executable, use also
  /Qimf-arch-consistency:true (Windows) or -fimf-arch-consistency=true (Linux or OS X)
- For best reproducibility between processors supporting FMA instructions and those that do not, use also
  /Qfma- (Windows) or -no-fma (Linux or OS X)
- For strict floating-point exception semantics
  except[-] (Windows*) or [no-]except (Linux* and OS X*)
- The following enables precise and except, disables contractions, and enables pragma stdc fenv_access
  /fp:strict (Windows*) or -fp-model strict (Linux* or OS X*)

References

- Intel® 64 and IA-32 Architectures Software Developer Manuals http://www.intel.com/products/processor/manuals/
- Intel® 64 and IA-32 Architectures Optimization Reference Manual http://www.intel.com/content/dam/www/public/us/en/documents/manuals/64-ia-32-architectures-optimization-manual.pdf
- Intel® Architecture Instruction Set Extensions Programming Reference https://software.intel.com/sites/default/files/managed/b4/3a/319433-024.pdf
- Intel® IEEE 754-2008 Binary Floating-Point Conformance Library https://software.intel.com/en-us/node/583388
- Intel® Decimal Floating-Point Math Library http://software.intel.com/en-us/articles/intel-decimal-floating-point-math-library/, http://www.netlib.org/misc/intel
- Intel® C++ Compiler 16.0 User and Reference Guide https://software.intel.com/en-us/intel-cplusplus-compiler-16.0-user-and-reference-guide
- Consistency of Floating-Point Results using the Intel® Compiler https://software.intel.com/en-us/articles/consistency-of-floating-point-results-using-the-intel-compiler/
- Differences in Floating-Point Arithmetic Between Intel® Xeon® Processors and the Intel® Xeon Phi™ Coprocessor x100 product family https://software.intel.com/en-us/articles/differences-in-floating-point-arithmetic-between-intel-xeon-processors-and-the-intel-xeon

Legal Disclaimer & Optimization Notice

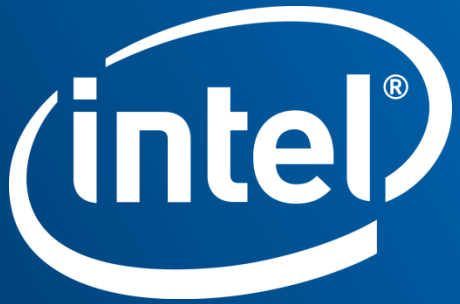
INFORMATION IN THIS DOCUMENT IS PROVIDED “AS IS”. NO LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE, TO ANY INTELLECTUAL PROPERTY RIGHTS IS GRANTED BY THIS DOCUMENT. INTEL ASSUMES NO LIABILITY WHATSOEVER AND INTEL DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY, RELATING TO THIS INFORMATION INCLUDING LIABILITY OR WARRANTIES RELATING TO FITNESS FOR A PARTICULAR PURPOSE, MERCHANTABILITY, OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT.

Software and workloads used in performance tests may have been optimized for performance only on Intel microprocessors. Performance tests, such as SYSmark and MobileMark, are measured using specific computer systems, components, software, operations and functions. Any change to any of those factors may cause the results to vary. You should consult other information and performance tests to assist you in fully evaluating your contemplated purchases, including the performance of that product when combined with other products.

Optimization Notice

Intel’s compilers may or may not optimize to the same degree for non-Intel microprocessors for optimizations that are not unique to Intel microprocessors. These optimizations include SSE2, SSE3, and SSSE3 instruction sets and other optimizations. Intel does not guarantee the availability, functionality, or effectiveness of any optimization on microprocessors not manufactured by Intel. Microprocessor-dependent optimizations in this product are intended for use with Intel microprocessors. Certain optimizations not specific to Intel microarchitecture are reserved for Intel microprocessors. Please refer to the applicable product User and Reference Guides for more information regarding the specific instruction sets covered by this notice.

Notice revision #20110804


intel®