

# Efficient 16-bit floating point interval processor for embedded applications

Michel Kieffer

Telecom ParisTech

on leave from

Laboratoire des Signaux et Systèmes

CNRS – Supélec – Université Paris-Sud, 11

Plateau de Moulon; 91192 Gif-sur-Yvette, France

kieffer@lss.supelec.fr

In the last ten years, interval techniques [1, 2] have allowed original solutions for many problems in engineering to be proposed, see, *e.g.*, [3]. One of the main features of interval techniques is their ability to provide *guaranteed* results, *i.e.*, with a verified accuracy or which are numerically *proved*. Consider for example, a bounded-error parameter estimation problem: the value of some parameter vector has to be estimated from measured data using a given model structure and bounded measurement errors. In such a context, one may obtain a set which can be proved to contain all values of the parameter vector that are consistent with the model structure, the measured data, and the hypotheses on the noise. Nevertheless, the application of interval techniques in embedded real-time applications is far less developed. The lack of efficient interval hardware support may be a reason for this slower development.

Hardware implementations of interval arithmetic have been mentioned twenty years ago in [4]. Extension of existing hardware platforms have been proposed, *e.g.*, in [5] and [6]. Nevertheless, chip builders were not yet convinced of the usefulness of performing specific adaptation of chips to implement interval analysis. This is why interval analysis is mainly performed by software implementations on general-purpose processors. Interval computations are however quite inefficiently performed on such processors, since the recurrent rounding mode switchings required by interval computations results in recurrent flushes of the processor pipeline [7]. This specific problem led people to study and design dedicated floating-point units (FPU) well suited to double rounding modes (towards  $-\infty$  and towards  $+\infty$ ) [6]. Moreover, in many applications, 32-bit FPU are oversized. Measurements, corrupted by errors, do not require to be processed with such an accuracy and in many cases, smaller FPU with reduced precision may fit the application constraints and provide a satisfying accuracy. Thus, for example, 16-bit floating-point computations is an efficient way to tackle both accuracy and dynamic problems encountered in signal and image processing [8], for filtering and convolution-based algorithms.

This paper introduces 16-bit floating-point arithmetic adapted to interval computations. The main idea is inspired by [6], which proposed to implement two 32-bit FPU on the 64-bit FPU of a general-purpose processor. Here, similarly, noticing that a 16-bit FPU is smaller than a 32-bit FPU, two 16-bit FPU (managing the two rounding modes required for interval computations) are shown not being much bigger than a single 32-bit FPU. The main advantage is that no rounding mode switching is required, preventing them from flushing the processor pipeline. The implementation of such a 16-bit FPU is performed on the FPGA based NIOS-II soft processor [9, 10], which allows instructions to be added to its instruction set. Customizable processors represent an opportunity to propose efficient and low-cost on-chip interval applications which may be used in embedded applications.

To compare the performance of 16 and 32-bit FPUs, an example of source localization using a network of acoustic or electromagnetic sensors is considered. In such network of sensors, power consumption and computational complexity are strong constraints when one is concerned with the increase of operability and autonomy [11]. Distributed interval constraint propagation [12] has been proposed as an efficient and low-complexity solution for source localization using a network of wireless sensors.

This talk first presents centralized and distributed source localisation problems and describes solutions based on interval analysis. The architecture of the 16-bit FPU is then presented. Attention is paid to accuracy and dynamic range. Results provided by a 32-bit FPU are compared to those obtained with two 16-bit FPU on realistic simulated data. Then, the hardware implementation on the three targeted architectures (Pentium4, Pentium 4-M, and NIOS-II) is presented and benchmarks for execution time and energy consumption are provided.

## References

- [1] R. E. Moore. *Interval Analysis*. Prentice-Hall, Englewood Cliffs, NJ, 1966.
- [2] A. Neumaier. *Interval Methods for Systems of Equations*. Cambridge University Press, Cambridge, UK, 1990.
- [3] L. Jaulin, M. Kieffer, O. Didrit, and E. Walter. *Applied Interval Analysis*. Springer-Verlag, London, 2001.
- [4] U. Kulisch and W. L. Miranker. The arithmetic of digital computer: A new approach. *Siam Review*, 28(1), 1986.
- [5] Jürgen Wolff von Gudenberg. Hardware support for interval computation. In G. Alefeld, A. Frommer, and B. Lang, editors, *Scientific Computing and Validated Numerics*, pages 32–37. Akademie-Verlag, Berlin, Germany, 1996.
- [6] R. Kolla, A. Vodopivec, and J. Wolff Von Gudenberg. The IAX architecture: Interval arithmetic extension. Technical report, 1999. <http://wwwi2.informatik.uni-wuerzburg.de/mitarbeiter/wvg/Public/iax.ps.gz>.
- [7] Intel. Desktop performance and optimization for intel pentium 4 processor. Technical Report 249438-01, 2001.
- [8] L. Lacassagne, D. Etiemble, and S.A. Ould Kablia. 16-bit floating point instructions for embedded multimedia applications. In *Proc. IEEE Computer Architecture and Machine Perception*, Palermo, 2005.
- [9] Altera. *NIOS Custom Instructions, Tutorial*, 2002. [http://www.altera.com/literature/tt/tt\\_nios.ci.pdf](http://www.altera.com/literature/tt/tt_nios.ci.pdf).
- [10] D. Etiemble, S. Bouaziz, and L. Lacassagne. Customizing 16-bit floating point instructions on a NIOS II processor for FPGA image and media processing. In *Proc. Estimedia*, New York, 2005.
- [11] Location is everything: Positioning in wireless networks (a special issue). In A. Dogandzic, J. Riba, G. Seco, and A. Lee Swindlehurst, editors, *IEEE Signal Processing Magazine*, volume 22, 2005.
- [12] M. Kieffer and E. Walter. Centralized and distributed source localization by a network of sensors using guaranteed set estimation. In *Proceedings of ICASSP*, volume 4, pages 977–980, 2006.