MITSUBISHI ELECTRIC
Changes for the Better

MITSUBISHI ELECTRIC R&D CENTRE EUROPE

1, ALLEE DE BEAULIEU, CS 10806
35708 RENNES CEDEX 7, FRANCE
TEL. (+33) 02.23.45.58.58 – FAX (+33) 02.23.45.58.59

for a greener tomorrow eco Changes

Fixed term contract research position proposal F/M (12 months)

Machine-checked verification of domain-specific Floating-Point Unit and its interface with software

Location:	*Rennes (35), France*
Web site:	*http://www.fr.mitsubishielectric-rce.eu/*
Job reference:	*INS_CDD_112020*
Contract:	*12 months contract, from Dec 2020*

Supervisor

Mitsubishi Electric R&D Centre Europe: Florian Faissole, Researcher

Overall context

With over 90 years of experience in providing reliable, high-quality products, MITSUBISHI ELECTRIC CORPORATION is a recognized world leader in the manufacture, marketing and sales of electrical and electronic equipment used in information processing and communications, space development and satellite communications, consumer electronics, industrial technology, energy, transportation and building equipment. The company is notably involved in safety critical applications like space, transportation, electrical power systems or automotive. Technologies used in these applications are particularly subject to high safety requirements, that could be achieved through the use of formal methods. This fixed term contract research position will take place within Information and Network Systems team of MITSUBISHI ELECTRIC R&D CENTRE EUROPE, either on site or fully remote, depending on the pandemic situation and national COVID protocol.

Fixed term contract research position subject

As most current microprocessors incorporate floating-point computing units (FPUs), the use of floating-point (FP) numbers on an industrial scale has become widespread, including for the most critical applications. Compared with fixed-point numbers, floating-point numbers give the developer a sense of simplicity, with many and somewhat dangerous misconceptions. There is indeed a belief which consists in thinking that floating-point numbers behave like real mathematical numbers, possibly with a negligible – and deterministically behaved – error. However, history has shown that 1) the miscalculations inherent in the use of finite precision data structures could lead to disasters, *e.g.* the Patriot anti-missile system's failure in 1994 [1], 2) that the clever manipulation of floating-point numbers would require a very high level of expertise and 3) that testing software based on floating-point arithmetic is not sufficient to avoid these disasters [2]. Since 1985, floating-point numbers are governed by the IEEE-754 international standard [3], specifying several floating-point formats and the elementary operations and rounding modes associated to these formats. Recent compilers and architectures generally declare themselves compatible with the IEEE-754 standard. Nonetheless, it has been noticed that most of them may lead to different results for identical floating-point computations

[2]. Formal development efforts have been made to increase confidence in floating-point computations. As an example, the Flocq library [4] for the Coq proof assistant [5] provides a formalization of generic floating-point formats with an important number of associated theorems, *e.g.* about floating-point operations, rounding modes or rounding error analysis. The main target of this library is software verification, for which a mathematical abstraction of floating-point numbers is generally sufficient. Nonetheless, Flocq includes a lower-level description of IEEE-754 binary FP numbers. Other formal developments focus on hardware-level floating-point arithmetic, *e.g.* the FPU's verification frameworks proposed by Russinoff in ACL2 [6] and Jacobi in PVS [7]. Nonetheless, these hardware-oriented frameworks are designed for the verification of state-of-the-art commercial or academic processors so that it might be difficult to directly apply their approach to other architectures, especially problem-specific architectures used for critical and highly-constrained industrial applications.

With a goal of technological transfer between fundamental research and industrial applications, the formal methods activity at Mitsubishi Electric favours the verification of whole chains of tools. In the context of floating-point arithmetic, it highlights the interest for bridging the gap between proofs of low-level properties of hardware and mathematical properties of software executed on it. For the sake of consistency, we postulate that the toolchain's verification should be performed in a unique framework, without losing formal guarantees at the interface between the components, from the mathematical model of floating-point numbers to the RTL and netlist hardware designs. The goal of this fixed term contract research position is to initiate the Coq development of a whole correct-by-construction domain-specific FPU. This includes the design of the hardware instructions set, its formal implementation and the verification of its compliance with IEEE-754 high-level properties as formalized in the Flocq library. In order to complete the formalization effort, it would be interesting to investigate the interaction of these hardware instructions with formally-verified compilers. The project scope also includes the verification of lower-level properties of the floating-point-units, *e.g.* the compliance of the circuits with the floating-point instructions they are intended to execute, taking the inherent parallelism of hardware (pipelining, instruction scheduling…) into account. An approach would be to compare the formalized instructions with RTL/Netlist implementations. Finally, we envision the formal integration of the FPU into a global processor and its verification.

Detailed objectives (and different steps)

- Understanding and documenting the precise structure of problem-specific hardware floating-point units and their interaction with the software
- Formalizing the instructions of a domain-specific floating-point unit in Coq and proving their properties (especially their compliance with the IEEE-754 standard)
- Proving the equivalence between the hardware implementation of these instructions and their Coq model definitions, either by certified equivalence checking or hardware synthesis
- Investigate the interface between formally-verified compilers and the verified FPU
- Implementing an external tool to automatically generate relevant documentation

Prerequisites

- Holding a PhD in computer science is mandatory
- An experience with proof assistants (e.g. Coq, Isabelle, HOL-Light, PVS, ACL2, Lean…) is mandatory, systems formalization would be a plus
- Interest in formal methods, more particularly for safety-critical software or hardware design
- Strong programming skills and good understanding of computer architecture
- Strong mathematical knowledge, at least in real analysis and logic
- Knowledge or experience in at least one of the following domains would be a plus:
    - Good knowledge of computer arithmetic and IEEE-754 standard for FP arithmetic
    - Experience with Coq and OCaml

    - Hardware design, circuits synthesis, VHDL/Verilog, FPU design
- Autonomy, ability to propose original solutions to solve research problems
- English: read and written

Duration: 12 months

Period: from December 2020

Contact for application: Magali BRANCHEREAU (jobs@fr.merce.mee.com)

Contact for further technical enquiry: Florian FAISSOLE (f.faissole@fr.merce.mee.com)

Thank you to provide us an application letter and your CV mentioning the reference of this fixed term contract research position (both in PDF versions).

References

[1] M. Blair, S. Obenski et P. Bridickas, «Patriot missile defense: Software problem led to system failure at Dhahran.,» Report GAO/IMTEC, 1992.

[2] D. Monniaux, «The pitfalls of verifying floating-point computations,» *ACM Transactions on Programming Languages and Systems (TOPLAS),* vol. 30, n° 13, pp. 1-41, 2008.

[3] W. Kahan, IEEE standard 754 for binary floating-point arithmetic, 1985.

[4] S. Boldo et G. Melquiond, Computer Arithmetic and Formal Proofs: Verifying Floating-point Algorithms with the Coq System, Elsevier, 2017.

[5] Y. Bertot et P. Castéran, Interactive theorem proving and program development: Coq'Art: the calculus of inductive constructions, Springer Science & Business Media, 2013.

[6] D. M. Russinoff, Formal Verification of floating-point hardware design: a mathematical approach, Springer, 2018.

[7] C. Berg et C. Jacobi, «Formal verification of the VAMP floating point unit,» in *Advanced Research Working Conference on Correct Hardware Design and Verification Methods*, 2001.