CPU development - Looking for HW and SW engineers

(CPU design, verification, modelling, implementation, development flows)

[Ref 2017-01] – Scalable Vector Extension technology investigations
[Ref 2017-02] – Machine learning for testbench stress optimization
[Ref 2017-03] – Level 1 Cache memory encryption
[Ref 2017-04] – Real-time dynamic power analyser
[Ref 2017-05] – Model driven property generation
[Ref 2017-06] – Innovative verification methodology based on execution path exploration
[Ref 2017-07] – Advanced, high-accuracy branch predictor
[Ref 2017-08] – New benchmarking strategy for internal production flow
[Ref 2017-09] – Formal equivalence checking with high-level models
[Ref 2017-10] – Scalable register bank design
[Ref 2017-11] – Clock tree methodology for high performance processors
[Ref 2017-12] – Impact of transactional memory support on CPU micro-architecture
[Ref 2017-13] – Multi-threading micro-architecture investigations

Typical internship progress:
- Introduction on the existing topic – microarchitecture, existing infrastructure,…
- Bibliographic study (if applicable)
- Propose solutions and select the optimal one (with the mentor’s help)
- Implement the chosen solution
- Verify the functional correctness, and evaluate the costs and benefits of the implemented solution on a real case scenario

Essential skills:
Different set of skills and knowledge are required depending on the selected internship topic. These should desirably include:
- Microprocessor, ASIC systems
- HDL and Synthesis design knowledge
- Programming language (C/C++, Python, …)
- Use of UNIX and shell programming
- Strong analytic skills
- Ability to express ideas and communicate effectively
- Ability to work in French and English environments
**Keywords:** ARM, processor, CPU model, SW development (C language), vectorization, parallel computing

**References:**
- [Ref 2017-01] – Scalable Vector Extension technology investigations
  ARM recently announced the SVE (Scalable Vector Extension) technology, its latest update to the ARMv8-A architecture. The goal of this study is to implement support for the SVE technology in a fast abstracted CPU model. In a final step, the student will evaluate and characterize the possible performance improvements brought by this new technology on a range of various benchmarks.

**Keywords:** ARM, processor, CPU model, SW development (C language), vectorization, parallel computing

**References:**
- [Ref 2017-02] – Machine learning for testbench stress optimization
  The goal of this internship is to implement machine learning capabilities to an existing CPU verification framework. The end goal is to set up a self-improving system aiming at optimizing the generation of complex verification tests to increase their stress level.

**Keywords:** ARM, processor, machine learning, functional verification, Python, shell programming

**References:**
- [Ref 2017-03] – Level 1 Cache memory encryption
  The goal of this internship is to research and benchmark various algorithms that could be used to encrypt the transactions between a processor and its memory. The student will implement the optimal solution(s) in a real CPU design.

**Keywords:** ARM, processor, microcontroller, L1 cache, RTL design, Verilog

**References:**
- [Ref 2017-04] – Real-time dynamic power analyser
  Along with pure performance, power efficiency is a key differentiator of modern CPU. The goal of this internship is to study and develop a real-time power analysis environment that will be used to investigate and optimize the power consumption of state-of-the-art ARM CPUs during their development phase.

**Keywords:** ARM, processor, power analysis, data mining, automation

**References:**
- [Ref 2017-05] – Model driven property generation
  Generating formal properties to test complex RTL modules represents the Holy Grail of the verification activity. The goal of this internship is to develop a state of the art automatic property generator, and develop a methodology around this tool to generate properties that’d help verify advanced ARM CPUs.

**Keywords:** ARM architecture, verification, model, property

**References:**
- [Ref 2017-06] – Innovative verification methodology based on execution path exploration
  The goal of this internship is to investigate and implement innovative verification methods based on execution path exploration, developing a prototype that would check whether the CPU execution of a given program is compliant with the ARM architecture.

**Keywords:** ARM architecture, algorithm, modelling, SW development, C++

**References:**
- [Ref 2017-07] – Advanced, high-accuracy branch predictor
  Branch prediction is a key element in modern processors performance. The goal of this internship is to study different high-accuracy branch prediction algorithms, implement some in a CPU model and provide a precise comparison of their characteristics (performance, implementation cost...).

**Keywords:** ARM, processor, CPU model, micro-architecture, branch prediction, performance analysis

**References:**
- [Ref 2017-08] – New benchmarking strategy for internal production flow
  Project success heavily relies on the quality of the development flows, which are in constant evolution. The goal of this internship is to investigate and develop a new strategy to benchmark these engineering production flows that would allow monitoring the quality and performance of recently developed features.

**Keywords:** SW development, Database, Performance analysis

**References:**
- [Ref 2017-09] – Formal equivalence checking with high-level models
  During a processor design, high-level abstracted models of many processors parts are often developed early in the project. The goal of this internship is to develop an innovative verification methodology, using those models as a reference for formal comparison with the RTL design. The internship will allow the student to study and develop this technique on a practical example.

**Keywords:** processor verification, sequential equivalence checking, high-level model, formal methods

**References:**
- [Ref 2017-10] – Scalable register bank design
  Accessing the register bank has become one of the hotspots in modern, out-of-order RISC CPU design, with a significant impact on the overall performance and power consumption of the processor. The goal of this internship is to investigate alternative ways to design and implement the registers banks that would scale and match the increase required by modern CPUs.

**Keywords:** ARM, processor, micro-architecture, RTL design, Verilog

**References:**
- [Ref 2017-11] – Clock tree methodology for high performance processors
  Clock tree implementation may be a key differentiating factor for both higher performance and lower power consumption in modern CPU designs. The goal of this internship is to investigate various clock-tree implementations and choose the optimal solution(s) that matches the requirement of a high-performance processor.

**Keywords:** ARM, processor, implementation, performance analysis, clock tree, synthesis

**References:**
- [Ref 2017-12] – Impact of transactional memory support on CPU micro-architecture
  The goal of this internship is to study transactional memory, and its impact on the micro-architecture of a high-performance out-of-order CPU. After an initial research on transactional memory concepts, the student will implement transactional memory support in the existing RTL design of a modern ARM application processor to assess its impact on the core pipeline.

**Keywords:** ARM, processor, transactional memory, speculation, RTL design, verification

**References:**
- [Ref 2017-13] – Multi-threading micro-architecture investigations
  The goal of this internship is to implement and analyse the benefits of adding multi-threading capabilities in advanced application processors. After an initial study of multi-threading technologies, the student will implement the most promising solution(s) in a micro-architectural CPU model written in C, and analyse the performance improvements on a given set of multi-threaded benchmarks.

**Keywords:** ARM, processor, CPU model, SW development (C language), multi-threading, performance analysis

**References:**