Rethinking Instruciton Set Architecture and Compiler for Bit-serial SIMD Processing Using Memory in DRAM

Xiangjun Peng[†] Yaohua Wang[‡] Ming-Chang Yang[†]

Abstract

This paper summarizes and retrospects the issues of Bit-serial SIMD PUD architectures, and addresses them by rethinking the ISA and compiler design for them. ❶ This work first suggests the simplification of ISA for Bit-serial SIMD PUD architectures, by directly exposing in-DRAM analog computation code as the only instruction extension. In this way, the fine-grained in-DRAM analog computation codes can (1) benefit from memory-level parallelism more effectively; and (2) naturally form code obfuscation for both computation and control-flow handling. ❷ This work then delivers a new compilation abstraction called "Pseudo Control Paths", by exploiting a newly-identified property from in-DRAM analog computation (i.e. the polymorphic functionalities from a single instruction). In this way, the compilation abstractions can (1) benefit from the enlarged optimization space for code compaction (with concrete optimizations); and (2) enhances control-flow obfuscation for the generated binaries. An implementation is delivered to show that: the above ideas can be fully automated and integrated with the state-of-the-art proposals of Bit-serial SIMD PUD architectures, without affecting the programmablity. The evaluation suggests the potential of the above ideas, which can enhance the practicality of Bit-serial SIMD PUD architectures.

This work delivers a generic design to tighten the integration between the processor and memory (with the analog computation capability), which breeds a variety of implications on the microarchitecture, Instruction Set Architecture, and compiler designs. First, this work demonstrates the feasibility to make the ISA of the host processor RISCer - namely breaks assumption that the Turing Completeness is NOT necessarily realized at the ISA level only (incidentally, the impossibility of obfuscation). Second, this work delivers a compile-time abstraction for increasing the redundancy from the actual control dependency, and this translates control dependency into data dependency. Third, this work delivers a few optimizations to take advantage of the compile-time redundancy, to translate the compile-time redundancy into the runtime benefits. Such designs are demonstrated beneficial within our context, and it is expected to be beneficial to SIMD, MIMD and beyond.

[*]Note that [†] refers the affiliation to *The Chinese University of Hong Kong*, and [‡] refers the affiliation to *National University of Defense Technology.*