Abstract

Advances in computer hardware often have little impact until they become accessible to programmers using high-level languages. For example, the IEEE floating-point arithmetic standard provides various rounding modes and exceptions, but it is difficult or impossible to take advantage of these from most high-level languages, so the full capabilities of IEEE-compatible hardware are seldom used. When they are used by writing in machine or assembly language, there is a high cost in program development and testing time, lack of portability, and difficulty of software maintenance.

In this talk we discuss several areas in which computer hardware, especially arithmetic hardware, can or should significantly influence programming language design. These include: vector units, floating-point exception handling, floating-point rounding modes, high/extended precision registers/arithmetic, and use of unusual number systems. Relevant application areas include interval arithmetic, high-precision integer arithmetic for computer algebra and cryptography, and testing of hardware by comparison with software simulations.