Preface

The *IBM Journal of Research and Development* has committed this issue to a description of the architecture and machine design of the IBM RISC System/6000* processor, along with some of the innovations in the implementation of the AIX* operating system, Version 3, that is used with that processor. Written by the system designers, the articles presented here explain in detail many of the advances in architecture, machine organization, hardware design, and the management of instructions, programs, and storage facilities that characterize this new family of superscalar RISC workstations and servers.

IBM RISC technology originated in 1974, and evolved continuously through a number of laboratory architectures and experimental machines. That experience, traced in the article by Cocke and Markstein, provided a number of technology transfers to the IBM product line—the I/O processor in the IBM 3090, the microprocessor in the IBM 9370, and the RISC platform of the IBM RT System. Most notably, however, it led to an understanding of design optimization, based only partially on traditional RISC principles, that is directly reflected in the IBM POWER (Performance Optimization With Enhanced RISC) architecture and its implementation in the RISC System/6000 (RS/6000) processor:

First, it was recognized that gains in pure cycle time afforded by a reduced instruction set are often offset by the hardware complexity arising from parallel structures and pipelining in modern RISC designs. Cycle time is thus affected more by basic device and circuit technology than by RISC implementation as such. Further, since execution time depends not only on cycle time but also on instruction path length and the average number of cycles per instruction required for execution, it was recognized that the main advantage of RISC lay in optimizing the trade-off between instruction complexity and cycles per instruction. Finally, it was recognized that the penalties of that trade-off, regardless of which way it is resolved, had become much smaller than they were when RISC was first conceived. Optimizing compilers had become available that could be used by highly parallel architectures to maximize parallel instruction execution, and advances in VLSI and CMOS technology had made possible hardware implementations of such architectures that would minimize physical loading and delay.

The design goal, then, was to *minimize the average number of cycles required by the instruction set* by minimizing the net *product* of instruction complexity (path length) and cycles per instruction. This approach led directly to the attributes of POWER architecture and the RS/6000 processor:

- A highly parallel *superscalar* architecture, employing separate but integrated functional units and caches and wide data buses, which permits the dispatch and execution of multiple instructions per cycle.
- An optimized hardware implementation that achieves both a low cycle time and a small cycles-per-instruction ratio. It is hard-wired rather than microcoded, and pipelined. Pipeline delay due to branching is minimized by a large number of instruction buffers and a wide-bandwidth interface between the instruction cache and the branch unit, which permits sufficient branch look-ahead that branches are effectively eliminated from the instruction stream. Since the arithmetic units never see branches directly and the instruction stream is rarely interrupted by branches, branching in most cases occurs in, effectively, zero cycles. In addition, an innovative floating-point execution unit, which is not a coprocessor but is tightly coupled to the rest of the CPU, embodies an accumulate function that combines a floating-point multiply with a floating-point add (or subtract) in a single operation that executes with no more delay than either function alone. This combination of parallelism and functional optimization permits an execution rate as high as five instructions per cycle—a branch, a condition-register operation, a fixed-point operation, and a floating-point operation (which, if it happens to be the accumulate instruction, has the effect of two operations).
- A register-oriented instruction set, which incorporates string operations and floating-point operations as well as primitives and is fully exposed to the compiler. Complex instructions are (in general) included *when the equivalent power and function cannot be achieved as quickly by sequences of simple instructions.*
- An optimizing compiler that employs optimized scheduling algorithms to take full advantage of the parallel hardware architecture.
- An operating system implementation that is also optimized for the parallel architecture, meeting special requirements for the management of programs, program libraries, and storage facilities.

The papers which follow describe this design in detail.

Bakoglu, Grohoski, and Montoye provide an overview of the hardware as a complete system. They describe the system configurations currently available and describe their performance and functional capabilities.

Oehler and Groves describe the evolution of the superscalar POWER architecture and the design decisions that led to the present implementation. Grohoski describes the actual implementation in the machine organization of the RS/6000 processor, with

*RISC System/6000 and AIX are trademarks of International Business Machines Corporation.

special emphasis on the synchronization of register operations and instruction-stream processing.

Montoye, Hokenek, and Runyon describe the floating-point execution unit, with emphasis on the motivation and design of the "multiply-add-fused" (MAF) unit which performs the special accumulate operation. They also provide a summary of floating-point operations as affected by that function, discuss the two-stage pipeline that was designed to be consistent with the over-all architecture, and review the integration of logical and physical design required for VLSI implementation. In a second paper on the FPU, Hokenek and Montoye describe a novel technique called leading-zero anticipation (LZA), which is used in the MAF to normalize floating-point results and allows normalization and addition to take place in a single cycle.

The RISC System/6000 processor incorporates a unified hardware self-testing procedure that can be run during system bring-up. The scheme employs embedded on-chip test processors, an on-card control sequencer, and an external processor used during bring-up to set breakpoints and display and modify the machine state. The method is especially suited to the testing of logic chips containing embedded RAM arrays (which describes most of the chips in the RS/6000 processor), and is described in the paper by Ratiu and Bakoglu.

Well-designed instruction scheduling algorithms are essential to the performance of the optimizing compiler; in his paper, Warren describes the algorithm that was developed to meet the scheduling requirements of the RS/6000 instruction set and architecture. As Warren notes, however, there is potential for even greater scheduling sophistication in future systems; Golumbic and Rainish explore one such possibility in their paper on scheduling beyond basic blocks.

The AIX operating system, Version 3, as implemented for the POWER architecture, incorporates a number of new developments in program and program library management and in the organization and management of storage facilities. These are discussed, respectively, in papers by Auslander and by Chang, Mergen, et al.

The RISC System/6000 processor was designed to comply with the IEEE standard for binary floating-point arithmetic. This requirement, and the speed and precision available in the RS/6000 FPU, motivated a reexamination of the algorithms used to perform division, square root, and the elementary functions such as *sin* and *exp*. In his paper on computation, Markstein describes new results which ensure correct last-bit rounding for these functions without special testing.

It is not possible to capture in a dozen papers the complete inventory of research, design, and development achievements that are reflected in the RISC System/6000 product family. We believe the most important ones are here, however, and we appreciate very much the cooperation of the authors, who spent time they could scarcely afford to help the *Journal* present them accurately.

N. M. Donofrio, President
IBM Advanced Workstations Division