

Four-Way Superscalar PA-RISC Processors

The HP PA 8000 and PA 8200 PA-RISC CPUs feature an aggressive four-way superscalar implementation, speculative execution, and on-the-fly instruction reordering.

by Anne P. Scott, Kevin P. Burkhart, Ashok Kumar, Richard M. Blumberg, and Gregory L. Ranson

The HP PA 8000 and PA 8200 PA-RISC CPUs are the first implementations of a new generation of microprocessors from Hewlett-Packard. The PA 8000¹⁻³ is among the world's most powerful and advanced microprocessors, and at the time of introduction in January 1996, the undisputed performance leader. The PA 8200,⁴ introduced in June 1997, continues this performance leadership with higher frequency, larger caches, and several other enhancements. Both processors feature an aggressive four-way superscalar implementation, combining speculative execution with on-the-fly instruction reordering. This paper discusses the objectives for the design of these processors, some of the key architectural features, implementation details, and system performance. The operation of the *instruction reorder buffer* (IRB),⁵ which provides out-of-order execution capability, will also be described.

PA 8000 Design Objectives

The primary design objective for the PA 8000 was to obtain industry-leading performance on a broad range of real-world applications. To sustain high performance on large applications, not just on benchmarks, we designed large, external primary caches with the ability to hide memory latency in hardware. We also chose to implement dynamic instruction reordering in hardware to maximize the instruction-level parallelism available to the execution units. Another goal was to provide full support for 64-bit applications. The processor implements the new PA-RISC 2.0 architecture, which is a binary compatible extension of the previous PA-RISC architecture. All previous code will execute without recompilation or translation. The processor also provides glueless support for up to four-way multiprocessing via a high-bandwidth Runway system bus.⁶ The Runway bus is a 768-Mbyte/s split-transaction bus that allows each processor to have several outstanding memory requests.

PA-RISC 2.0 Enhancements

The new PA-RISC 2.0 architecture incorporates a number of advanced microarchitectural enhancements. Most of the extensions involve support for 64-bit computing. Integer registers and functional units, including the shift/merge units, have been widened to 64 bits. Flat virtual addressing up to 64 bits is supported, as are physical addresses greater than 32 bits (40 bits were implemented on the PA 8000). A new mode bit has been implemented that governs address formation, creating increased flexibility. In 32-bit addressing mode, it is still possible to take advantage of 64-bit compute instructions for faster throughput. In 64-bit addressing mode, 32-bit instructions and conditions are still available for backwards compatibility.

Other extensions help optimize performance in the areas of virtual memory and cache management, branching, and floating-point operations. These include fast TLB (translation lookaside buffer) insert instructions, load and store instructions with 16-bit displacement, memory prefetch instructions, support for variable-sized pages, half-word instructions for multimedia support, branches with 22-bit displacements and short pointers, branch prediction hinting, floating-point multiply-accumulate instructions, floating-point multiple compare result bits, and other carefully selected features.

Hardware Design

The PA 8000 features a completely redesigned core that does not leverage any circuitry from previous-generation HP processors. This break from previous CPUs allowed us to include new microarchitectural features we deemed necessary for higher performance. Fig. 1 is a functional block diagram of the processor showing the basic control and data paths.

The most notable feature of the chip, illustrated in the center of the diagram, is the industry's largest instruction reorder buffer of 56 entries, which serves as the central control unit. This block supports full register renaming for all instructions in the buffer, and tracks interdependencies between instructions to allow data flow execution through the entire window.

The PA 8000 features a peak execution rate of four instructions per cycle, made possible by a large complement of computational units, located on the left side of the diagram. For integer operation, two 64-bit integer ALUs and two 64-bit shift/merge units are included. All integer functional units have a single-cycle latency. For floating-point applications, dual floating-point multiply and accumulate (FMAC) units and dual divide/square root units are included. The FMAC units are optimized for performing the very common operation $A \times B + C$. By fusing an add to a multiply, each FMAC can execute two floating-point operations in just three cycles. In addition to providing low latency for floating-point operations,

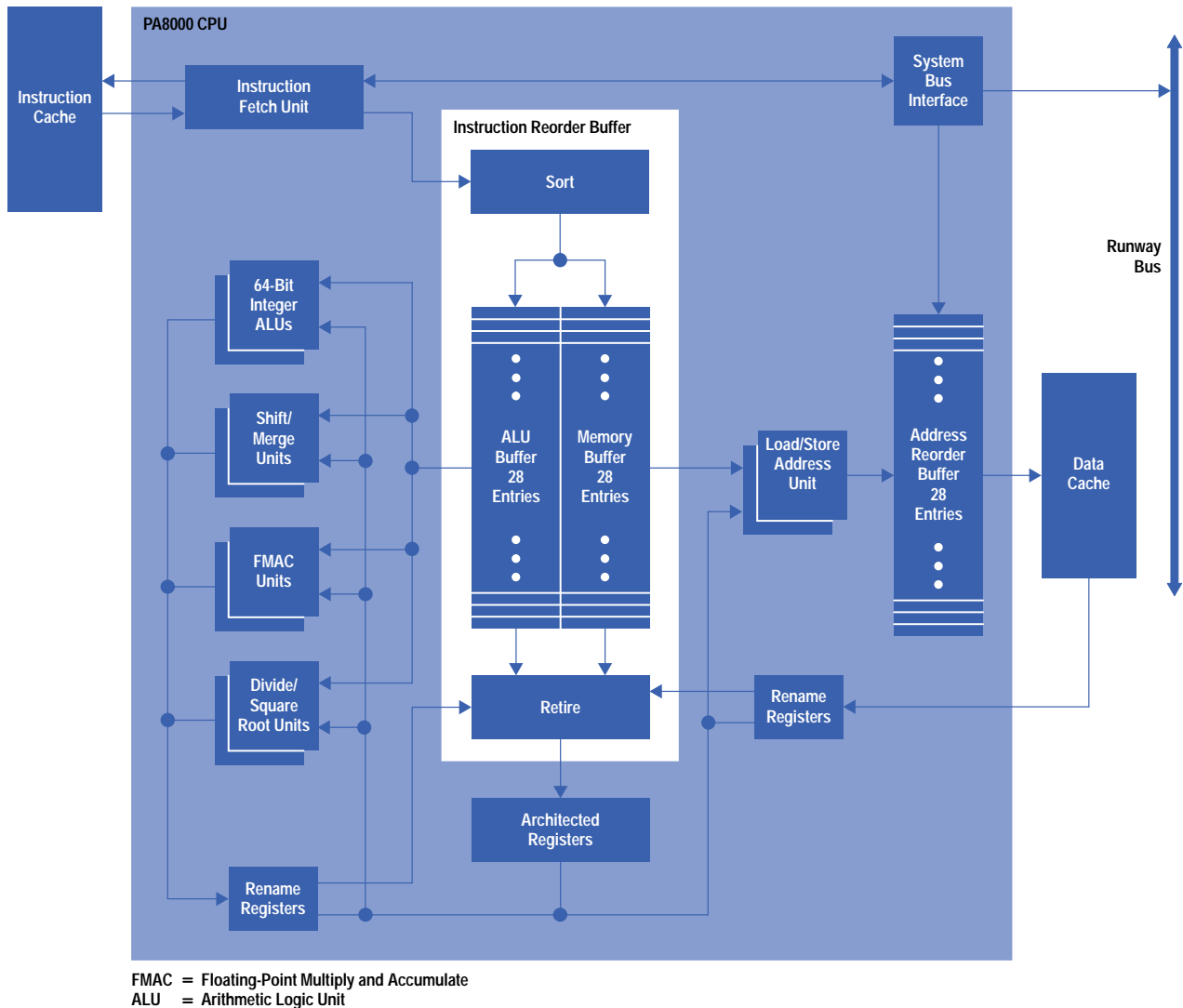


Fig. 1. Functional block diagram of the HP PA 8000 processor.

the FMAC units are fully pipelined so that the peak floating-point throughput of the PA 8000 is four floating-point operations per cycle. The two divide/square root units are not pipelined, but other floating-point operations can be executed on the FMAC units while the divide/square root units are busy. A single-precision divide or square root operation requires 17 cycles, while double precision requires 31 cycles.

Having such a large array of computation units would be pointless if those units could not be supplied with enough data upon which to operate. To this end, the PA 8000 incorporates two complete load/store pipes, including two address adders, a 96-entry dual-ported TLB, and a dual-ported cache. The right side of Fig. 1 shows the dual load/store units and the memory system interface. The symmetry of dual functional units throughout the processor allows a number of simplifications in the data paths, the control logic, and signal routing. In effect, this duality provides for separate *even* and *odd* machines.

As pipelines get deeper and the parallelism of a processor increases, instruction fetch bandwidth and branch prediction become increasingly important. To increase fetch bandwidth and mitigate the effect of pipeline stalls for branches predicted to be taken, the PA 8000 incorporates a 32-entry *branch target address cache*, or BTAC. This unit is a fully associative structure that associates the address of a branch instruction with the address of its target. Whenever a branch predicted to be taken is encountered in the instruction stream, an entry is created in the BTAC for that branch. The next time the fetch unit fetches from the address of the branch, the BTAC signals a hit and supplies the address of the branch target. The fetch unit can then immediately fetch the target of the branch without incurring any penalty, resulting in a zero-state taken branch penalty for branches that hit in the BTAC. In an effort to improve the hit rate, only branches predicted to be taken are kept in the BTAC. If a branch hits in the BTAC but is predicted not to be taken, the entry is deleted.

To reduce the number of mispredicted branches, the PA 8000 implements two modes of branch prediction: *dynamic mode* and *static mode*. Each TLB entry has a bit to indicate which prediction mode to use. Thus, the mode is selectable on a page-by-page basis. In dynamic prediction mode, a 256-entry *branch history table*, or BHT, is consulted. The BHT stores the results of the last three iterations of each branch (either taken or not taken), and the instruction fetch unit predicts that the

outcome of a given branch will be the same as the majority of the last three outcomes. In static prediction mode, the PA 8000 predicts most conditional forward branches to be untaken, and most conditional backward branches to be taken. For the common compare-and-branch instruction, the PA-RISC 2.0 architecture defines a branch prediction bit that indicates whether this normal prediction convention should be followed or whether the opposite convention should be used. Compilers using either heuristic methods or profile-based optimization can use static prediction mode to communicate branch probabilities effectively to the hardware.

Cache Design

The PA 8000 features large, single-level, off-chip, direct-mapped instruction and data caches. Both caches support configurations of up to four megabytes using industry-standard synchronous SRAMs. Two complete copies of the data cache tags are provided so that two independent accesses can be accommodated and need not be to the same cache line.

Why did we design the processor without on-chip caches? The main reason is performance. Competing designs incorporate small on-chip caches to enable higher clock frequencies. Small on-chip caches support benchmark performance but fade on large applications, so we felt we could make better use of the die area. The sophisticated IRB allows us to hide the effects of a pipelined two-state cache latency. In fact, our simulations demonstrated only a 5% performance improvement if the cache were on-chip and had a single-cycle latency. The flat cache hierarchy also eliminates the design complexity associated with a two-level cache design.

Chip Statistics

The PA 8000 is fabricated in HP's 0.5-micrometer, 3.3-volt CMOS process. Although the drawn geometries are not very aggressive, we still obtain a respectable 0.28- μm effective channel length (L_{eff}). In addition, extensive investment was made in the design process to ensure that both layout and circuits would scale easily into more advanced technologies with smaller geometries. There are five metal layers: two for tight pitch routing and local interconnect, two for low-RC global routing, and a final layer for clock and power supply routing.

The processor is designed with a three-level clock network, organized as a modified H-tree (see [Article 2](#)). The clock sync signals serve as primary inputs. They are received by a central buffer and driven to twelve secondary clock buffers located in strategic spots around the chip. These buffers then drive the clock to the major circuit areas, where it is received by *clock gates* featuring high gain and a very short input-to-output delay. There are approximately 7,000 of these gates, which have the ability to generate many flavors of the clock: two-phase overlapping or nonoverlapping, inverting or noninverting, qualified or nonqualified. The qualification of clocks is useful for synchronous register sets and dumps, as well as for powering down sections of logic when not in use. Extensive simulation and tuning of the clock network were done to minimize clock skew and improve edge rates. The final clock skew for this design was simulated to be no greater than 170 ps between any two points on the die.

Under nominal operating conditions of room temperature and 3.3-volt power supplies, the chip is capable of running at frequencies up to 250 MHz. Although we cannot guarantee processor performance based on results obtained under ideal conditions, there appears to be an opportunity for greater frequency enhancement. The die measures 17.68 mm by 19.1 mm and contains 3.8 million transistors. Approximately 75% of the chip is either full-custom or semicustom. A photograph of the die with all major areas labeled is shown in Fig. 2. Again, the IRB is in the center of the chip, providing convenient access to all the functional units. The integer data path is on the left side of the chip, while the right side contains the floating-point data path.

By using flip-chip packaging technology, we were able to support a very large number of I/O signals—704 in all. In addition to the I/O signals, 1,200 power and ground solder bumps are connected to the 1,085-pin package via a land grid array. There are fewer pins than the total of the I/Os and bumps because each power and ground pin can be connected to multiple bumps. A picture of the packaged part is shown in Fig. 3. The chip is flipped onto the ceramic carrier using solder bump interconnect, and the carrier is mounted on a conventional printed circuit board. This packaging has several advantages. The wide off-chip caches are made possible by the high-pin-count capability. The ability to place I/O signals anywhere on the die improves area utilization and reduces on-chip RC delays. Finally, the low inductance of the signal and power supply paths reduces noise and propagation delays.

Performance

At 180 MHz with one megabyte of instruction cache and one megabyte of data cache, the HP PA 8000 delivers over 11.8 SpecInt95 and greater than 20.2 SpecFP95, making it the world's fastest processor at the time of introduction. A four-way multiprocessor system has also produced 14,739.03 TpmC (\$132.25/TpmC), where TpmC is an industry-standard benchmark for online transaction processing. That system configuration was made available in June 1996.

Enabling the PA 8000 to achieve this level of performance are several distinguishing features. First, there are a large number of functional units—ten, as described previously. However, multiple units alone are not enough. To sustain superscalar operation beyond two-way demands advanced instruction scheduling methods to supply a steady stream of independent tasks to the functional units. To achieve this goal, an aggressive out-of-order execution capability was incorporated. The instruction reorder buffer provides a large window of available instructions combined with a robust dependency tracking system.

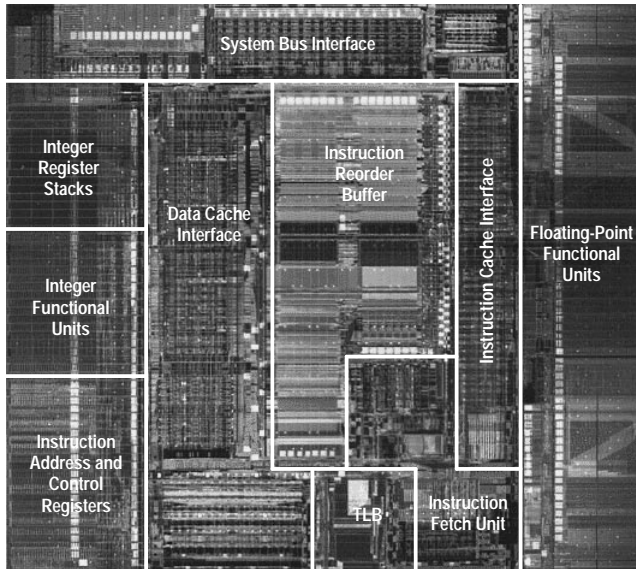


Fig. 2. PA 8000 CPU with major areas labeled.

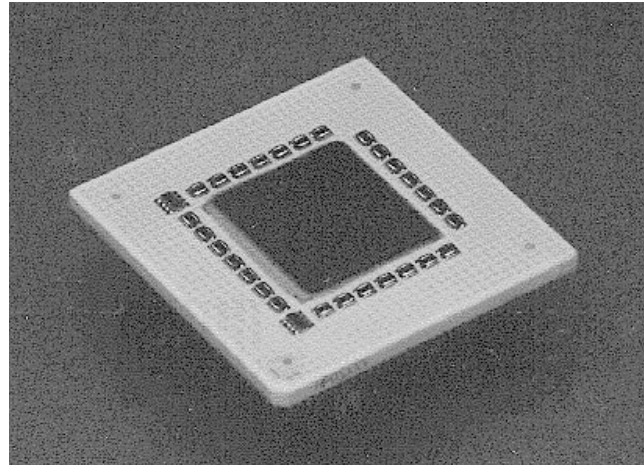


Fig. 3. Packaged PA 8000 CPU.

Second, having explicit compiler options to generate hints to the processor helps a great deal. These special instructions can be used to prefetch data and to communicate statically predicted branch behavior to the branch history table, as described previously.

Finally, the system bus interface is capable of tracking up to ten pending data cache misses, an instruction cache miss, and an instruction cache prefetch. Since multiple misses can be serviced in parallel, the average performance penalty caused by each is reduced.

Instruction Reorder Buffer

Because of restrictions on compiler scheduling, a key decision was made to have the PA 8000 perform its own instruction scheduling. To accomplish this task, the PA 8000 is equipped with an instruction reorder buffer, or IRB, which can hold up to 56 instructions. This buffer is composed of two pieces: the ALU buffer, which can store up to 28 computation instructions, and the MEM (memory) buffer, which can hold up to 28 load and store instructions. These buffers track over a dozen different types of interdependencies between the instructions they contain, and allow instructions anywhere in the window to execute as soon as they are ready.

As a special feature, the IRB tracks branch prediction outcomes, and when a misprediction is identified, all instructions that were incorrectly fetched are flash-invalidated. Fetching then resumes down the correct path without any further wasted cycles.

The IRB serves as the central control logic for the entire chip, yet consists of only 850,000 transistors and consumes less than 20% of the die area. A high-performance IRB is of paramount importance, since today's compilers simply lack run-time information, which is useful for optimal scheduling. The reorder buffer on the PA 8000 is 40% larger than that of the nearest competitor.

Instruction reordering also leads to the solution for another bottleneck: memory latency. Although the dual load/store pipes keep the computation units busy as long as the data is cache-resident, a data cache miss can still cause a disruption. Execution can continue for many cycles on instructions that do not depend on the data cache miss. The PA 8000 can execute instructions well past the load or store that was missed, since the IRB can hold so many instructions. When useful work can be accomplished during a data cache miss latency, the net impact on performance is significantly reduced.

The large window of available instructions also allows overlap of multiple data cache misses. If a second data cache miss is detected while an earlier miss is still being serviced by main memory, the second miss will be issued to the system bus as well.

Life of an Instruction

A block diagram of the PA 8000's instruction reorder buffer is shown in Fig. 4. Instructions enter through the sort block and are routed to the appropriate portion of the IRB based on instruction type, where they are held until they retire. The functional units are connected to the appropriate section of the IRB based on what types of instructions they execute. After execution, instructions are removed from the system through the retire block.

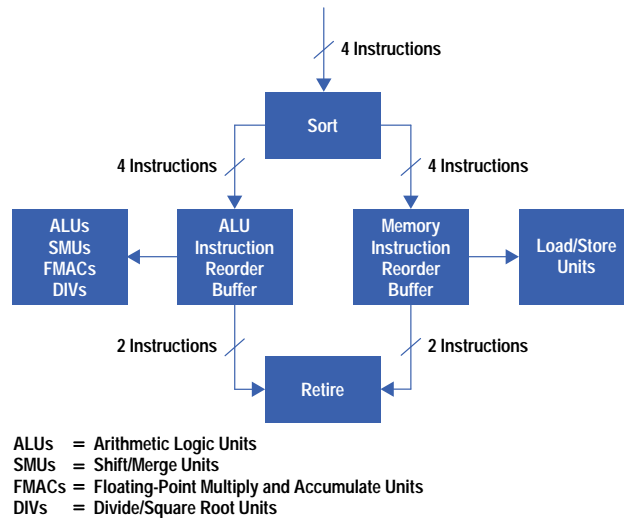


Fig. 4. PA 8000 instruction reorder buffer block diagram.

Instruction Insertion. The IRB must be kept as full as possible to maximize the chances that four instructions are ready to execute on a given cycle. A high-performance fetch unit was designed to maximize IRB occupancy. This unit fetches, in program order, up to four instructions per cycle from the single-level off-chip instruction cache.

Limited predecode is then performed, and the instructions are inserted in a round-robin fashion into the appropriate IRB. Each IRB segment must be able to handle four incoming instructions per cycle, since there are no restrictions on the mix of instructions being inserted.

There are several special cases. Branches, although executed from the ALU IRB, are also stored in the MEM IRB as a placeholder to indicate which entries to invalidate after a mispredicted branch. Instructions that have both a computation and a memory component and two targets, such as the load word and modify (LDWM) instruction, are split into two pieces and occupy an entry in both portions of the IRB.

Instruction Launch. Instructions are allowed to execute out of order. During every cycle, both segments of the IRB allow the oldest even and the oldest odd instruction for which all operands are available to execute on the functional units. Thus, up to four instructions can be executed at once: two computation instructions and two memory reference instructions. Once an instruction has been executed, its result is held in a temporary rename register and made available for use by subsequent instructions.

Instruction Retire. Instructions are removed or retired from the IRB in program order once they have executed and any exceptions have been detected. Enforcing strict retirement order provides software with a precise exception model. As instructions are retired, the contents of the rename registers are transferred to the general registers, stores are placed in a queue to be written to cache, and instruction results are committed to the architected state. The retire unit can handle up to two ALU or floating-point instructions and up to two memory instructions each cycle.

The HP PA 8200 Processor

After the successful introduction of PA 8000 processor-based products, the PA 8000 design team initiated a follow-up program. Performance analysis on key applications identified several opportunities for future products. The PA 8200 CPU team formulated a plan for improvement based on the following goals set by HP customers and management:

- Improved performance
- Compatibility with existing applications
- Leverage of the PA 8000 design foundation
- Rapid time to market.

Improved Performance. Application trace studies identified branch prediction, TLB miss rates, and increased cache sizes as significant opportunities. The availability of next-generation 4M-bit SRAMs with improved access times allowed the design team to increase processor clock speed and double cache size to 2M bytes for both the instruction cache and the data cache. The faster access time of 4M-bit SRAMs allowed higher processor clock rates without changes to the cache access protocol. The combination of increased clock frequency, larger caches, improvement of branch prediction accuracy, and reduction of TLB miss rates enables performance improvements of 15% to 30% on key applications.

Compatibility with Existing Applications. Follow-on products using the PA 8200 had to preserve our customers' investment in PA 7200-based and PA 8000-based software and hardware. It was considered essential to maintain binary compatibility with existing PA-RISC applications and provide an upgrade path for improved performance.

Leverage of PA 8000. The PA 8200 design team leveraged the extensive functional and electrical verification results accumulated during the prototyping phase of the PA 8000 development. A wealth of design data is collected in the process of turning a design into a product. This information identified the paths limiting CPU operating speed and the performance limiters in the branch and TLB units. Characterization of the PA 8000 cache design provided the basis for a new design using high-speed 4M-bit SRAMs.

Rapid Time to Market. The competitive situation dictated that speed upgrades to the PA 8000 were needed to maintain HP's performance leadership in the high-performance workstation and midrange server markets. Therefore, design changes and characterization of the expanded cache subsystem had to be completed within a very aggressive schedule.

In the following sections, PA 8200 design changes to the PA 8000 processor will be detailed.

PA 8000 Performance Analysis

Given the goals of increased performance with low risk and a short time to market, it was necessary to understand fully where the PA 8000 excelled and where significant improvements could be made. Key customer applications were examined to determine how real-world code streams were being executed on the PA 8000.

For the PA 8000, the expectation was set that no code recompilation would be necessary to see a $2 \times$ speedup over the PA 7200. We did not want to change this expectation for the PA 8200, so all code experiments were performed using nonrecompiled, nontuned code. It was shown that the PA 8200's performance could be enhanced significantly over that of the PA 8000 by reducing the amount of time the PA 8200 spent waiting for instructions or data. The branch history table (BHT) and translation lookaside buffer (TLB) are architectural features that are intended to reduce wasted cycles resulting from penalties, particularly in pipelined machines. For mispredicted branches, TLB misses, and cache misses, the number of penalty cycles increased from the PA 7200 to the PA 8000. It was expected that a corresponding reduction in mispredictions and misses and the ability to hide penalty cycles using out-of-order execution would result in an overall decrease of wasted cycles. The analysis of the application suite showed otherwise, as the number of wasted cycles increased from the PA 7200 to the PA 8000, accounting for 36 to 86 percent of the total number of cycles spent on each instruction (CPI). If the number of mispredictions and misses could be decreased, a significant performance boost would be realized. As a result, increases in the size of the BHT, TLB, and caches were examined as potential high-benefit, low-risk improvements to the PA 8000.

BHT Improvement

The biggest performance weakness observed was the mispredicted branch penalty. By its nature, out-of-order execution increases the average penalty for mispredicted branches. Therefore, significant design resources were allocated for the PA 8000's branch prediction scheme to lower the misprediction rate, thereby offsetting the higher penalty. The results of the performance analysis revealed that cycles wasted because of branch penalties were still significantly impacting performance. Relative to the PA 7200, the misprediction rate is generally about 50% lower across the sample workload of technical applications. However, the cycle penalty for a mispredicted branch rose by 200%, more than offsetting the reduction in miss rate. There are clearly two possible solutions: decreasing the miss rate or decreasing the miss penalty. Because of the short time schedule of the program, redefining how mispredicted branches are handled to reduce the penalty was not a viable alternative. The more practical solution was to improve branch prediction accuracy.

Improvements to the BHT focused on two areas. The first was the table size and the second was the branch prediction algorithm. The PA 8000 uses a three-bit majority vote algorithm and a 256-entry BHT. Since the PA 8000 also allows up to two branches to retire simultaneously, the table ideally would be able to update two entries per cycle. Parallel BHT update was not implemented on the PA 8000, resulting in the outcome of one of the branches not having its information entered into the BHT. Analysis of this limitation revealed a minor penalty that could easily be eliminated in the PA 8200.

Initial investigation for BHT improvements focused on the size of the table since it is easier to increase the size of an existing structure than to start from scratch and redefine the algorithm. To have the minimum impact on control logic, it was desirable to increase the table size by a multiple of two. Visual inspection of the area around the BHT revealed that the number of entries could be increased to 512 with little impact.

Next, possible changes in the prediction algorithm were explored. Using a more common algorithm became the key to allowing the BHT to grow to 1024 entries. The new algorithm requires only two bits of data compared to the three-bit algorithm implemented on the PA 8000. Analysis of the two algorithms showed that they result in almost the same predictions with only a few exceptions. The reduction in the number of bits per entry from three to two allowed the BHT to grow from 512 to 1024 entries. The increase from the algorithm change was shown through simulation to provide more of an incremental improvement than was lost by the switch to the two-bit algorithm.

One additional improvement was made to the BHT concerning the handling of multiple branches retiring at the same time. Allowing two entries to be updated simultaneously required the data entries to have two write ports. This functionality was not included in the PA 8000, so implementing a two-port solution on the PA 8200 would be very expensive in die area.

Therefore, a control-based solution was devised. When two branches retire on the same cycle, the information necessary to update the cache for one of the branches is held in a one-entry queue. On the next cycle, the data in the queue is used to update the table. If another branch also retires on the next cycle, the queue data is written into the BHT and the newly retiring branch's data is stored in the queue. Only if two branches retire while the queue contains data is the data for one branch lost. This condition is considered to be quite rare, since it requires that multiple pairs of branches retire consecutively. The rarity of this situation makes the performance impact of losing the fourth consecutive branch's data negligible.

The risk involved with making the described changes to the BHT was relatively low. The data storage elements are well-understood structures and could be expanded with little risk. The control for the new BHT could mostly be leveraged from the PA 8000 implementation with the exception of the new branch store queue. Significant functional verification was done to ensure correctness of the new BHT. Since control and data paths remained almost the same as the old BHT, there was high confidence that the changes would not introduce new frequency limiters.

TLB Improvement

The second major area of improvement involved the TLB. Relative to the PA 7200, the PA 8000 uses significantly more cycles handling TLB misses on most of the applications used to analyze performance. The reason for this increase is twofold. First, the penalty for a TLB miss increased from 26 cycles on the PA 7200 to 67 cycles on the PA 8000. The increase in TLB miss penalty was mainly caused by an increase in control complexity resulting from the out-of-order capability of the PA 8000. Second, the TLB miss rate for most of the applications examined also increased. The total number of entries decreased by 20% from 120 to 96 between the PA 7200 and the PA 8000. However, the PA 8000 has a combined instruction and data TLB while the PA 7200 has separate instruction and data TLBs. At the time, a decrease in size seemed an acceptable trade-off since instruction and data TLB entries could now use the entire TLB.

Since the penalty for a TLB miss could not be reduced without significant redefinition of how a TLB miss is handled, the number of entries was the area of focus. Simulation revealed that increasing the number of entries provided a nearly linear improvement in the TLB miss rate, leveling off at about 128 entries. In looking at the area the TLB occupied and the surrounding routing channels, it became clear that 128 entries would involve an unacceptable design risk. Since the implementation is most efficient in multiples of 8, we next examined 120 entries. Initial examination of the artwork showed that this target would be aggressive, yet reasonable. Simulations were done assuming 128 entries to provide some additional timing margin and to allow for increasing to 128 entries if it became possible. Most of the circuit timing paths were found to have nearly the same performance with 120 entries as 96 entries since the critical variable for timing is generally the width of an entry and not the number of entries. Some minor changes to transistor sizing provided the additional margin necessary on critical paths that traversed the TLB array. The goal of these changes was to increase the number of TLB entries over the PA 8000 without impacting speed.

The biggest risk that the TLB changes posed was to the project schedule. The area affected by the changes was much larger than that of any other change, and there were hard boundaries to other functional units that constrained design area. To increase the size of the TLB, two complex signal channels were rerouted. Although necessary to provide the additional room, the changes were time-consuming and presented significant schedule risk. Routing changes also increased the chance of a change in the electrical performance of the affected signals. To minimize this risk, a tool was written to verify that signal integrity was not compromised. Overall, the rerouting of the channels was the critical path to tape release and also the highest risk.

Frequency Improvement

In addition to improving the BHT and TLB performance, the target frequency for the PA 8200 was increased over that of the PA 8000. We took a two-pronged approach to timing analysis. The first approach consisted of analyzing a software model of PA 8000 timing and the second approach consisted of examining data from prototype systems in which we increased the frequency to the failing point.

The PA 8000 timing was modeled using Epic's Pathmill and Timemill suite and Verilog's Veritime. These tools provided an ordered set of paths ranked according to predicted operation frequency. We grouped the data into paths that were internal to the chip (*core paths*) and paths that received or drove information to the cache pins (*cache paths*). It became readily apparent that there was a very small set of core paths and a much larger set of cache paths that could potentially limit chip frequency. The core paths tended to be independent of all other core paths and could be improved on an individual basis within the CPU. The cache path limiters tended to funnel into a couple of key juncture points and could be globally improved by addressing those points. As an additional degree of freedom, cache paths could be addressed through a combination of CPU, board, and cache SRAM improvements.

Once it was determined which core paths might limit chip frequency, we had to devise a method to correlate the simulated frequency with actual chip performance. Targeted tests were written to exercise potential core limiters. Paths were chosen based on their independence from known limiters and for their ability to be completely controlled by the test. The targeted tests ran consistently faster on silicon than the model predicted, giving us confidence that core paths would not be frequency limiters.

We then looked at correlating cache paths between the model and the system. Cache paths tend to be multistate paths dependent on the timing of the cache SRAMs. Because of these attributes, it was not feasible to craft a chip-level test to exercise specific cache paths. Therefore, we decided to rely upon system data for determining worst-case cache paths and then use the model data to show the frequency of other cache paths relative to the worst case. System work revealed two cache path frequency limiters. Both paths were predicted by and correlated with the timing model.

Based on the cache paths exposed through system work, an additional timing investigation was launched. Both paths funnelled into a similar set of circuits to send addresses to the SRAMs. All other inputs into those circuits were examined and individually simulated using SPICE to determine if they had the potential to become frequency limiters. From this effort, one additional set of inputs was identified as having a high risk of becoming a frequency limiter once the known limiters were improved. The proposed improvements to the known limiters improved the newly identified path as well, keeping it from becoming a critical path.

The final step taken to understand the frequency limitations of the PA 8000 was to devise a way to look beyond the known limiting paths in a system. The lowest frequency speed limiter was a cache path related to an architectural feature to improve performance. On the PA 8000, this feature can be disabled. However, the second speed limiter was not programmable and was therefore capable of masking other paths. We turned to focused ion beam (FIB) technology to help us solve this problem.

The second speed limiter was a single-phase path that started with the rising edge of a clock and ended with the falling edge of a derived clock. By delaying the falling edge of the derived clock, we could increase the frequency at which the path could run, creating a new region in which we could search for failing paths. We used the FIB to cut away and rebuild the circuitry for the derived clock. In the process of stripping away the metal on the chip and then redepositing it to rebuild the circuit, resistance is added, slowing down the circuit. We were able to add 220 ps to the path, increasing the failing frequency for this limiter by approximately 22 MHz. The FIB-modified chip was placed in a system for extensive testing. No additional failing paths were found in the newly opened frequency region.

In improving the critical paths for the PA 8200, a conservative design approach was adopted. Most of the improvements involved moving clock edges, allowing latches to update earlier than before. Such changes can expose races or setup violations. The paths were carefully simulated to eliminate the risk of introducing a race. In cases where it was difficult to precisely determine the setup time needed for a signal, conservative changes were made.

Cache Improvement

Yet another area for improvement on the PA 8200 was the cache subsystem. The cache size plays an integral role in determining how well the system performs on both applications and benchmarks. In addition, the off-chip cache access path can limit the operating frequency of the system because of the tight coupling between the CPU and the SRAMs.

The PA 8000 offered a maximum cache size of 1M bytes for both the instruction and data caches. A total of 20 1M-bit industry-standard late-write synchronous SRAMs were employed for this configuration. The printed circuit board design was cumbersome because of the large number of SRAM sites. The design resulted in relatively long round-trip delays. As the PA 8200 was being defined, the next generation of SRAMs became available. These 4M-bit parts were fully backwards compatible with those used with the PA 8000. The emergence of these higher-density components made possible a 2M-byte instruction cache and a 2M-byte data cache while reducing the number of SRAMs to 12. The resulting board layout was more optimal, contributing to shorter routes and better signal integrity.

In addition to cache size, the frequency limitation of the off-chip cache was carefully addressed. For much of the post-silicon verification of the PA 8000, the two-state cache access presented a frequency barrier that limited the amount of investigation beyond 180 MHz. Two main contributors allowed the frequency of the PA 8200 to be increased well beyond 200 MHz. The first was the new SRAM placement and routing for the cache subsystem. The 12-SRAM configuration yielded a new worst-case round-trip delay that was 500 ps shorter than the 20-SRAM configuration previously used. The second enabler was linked to the next-generation SRAMs. Not only did these parts provide four times the density, they also reduced their access times from 6.7 ns to 5.0 ns. The combined benefit of these two enablers resulted in raising the maximum cache-limited frequency from 180 MHz to 230 MHz. The value of this improvement was really twofold. First, it enabled system-level electrical characterization and CPU core speed path identification in a space previously unexplored. Second, it resulted in a manufacturable product that could meet the performance needs of our workstations.

PA 8200 Performance

Under nominal operating conditions of room temperature and 3.3-volt power supplies, the PA 8200 is capable of running up to 300 MHz, 70 MHz faster than its predecessor. Table I summarizes its performance.

Benchmark	Estimated Performance	Frequency
SPECint95	16.1	230 MHz
SPECfp95	25.5	230 MHz

Conclusion

The HP PA 8000 RISC CPU achieved industry-leading performance across a wide variety of applications by using an aggressive out-of-order design and carefully balancing hardware utilization throughout the system. The PA 8200 leverages that design, improving key areas identified by customer needs and applications. The number of TLB and BHT entries was increased, chip operating frequency was increased, and the cache configuration was updated to include the latest available SRAM technology. Together these changes improved system performance across customer applications up to 23%, once again delivering industry-leading performance.

Acknowledgments

The authors would like to acknowledge the contributions of the technical and management staffs at HP's Engineering Systems, Enterprise Systems, Workstation Systems, Operating Systems, Compiler, and System Performance Laboratories located in Fort Collins, Colorado, Roseville, California, and Cupertino, California. Without the tremendous effort put forth by so many individuals on these projects, the PA 8000 and PA 8200 would not have been possible.

References

1. D. Hunt, "Advanced Performance Features of the 64-bit PA 8000," *Comcon Digest of Papers*, March 1995.
 2. A. Kumar, "The Hewlett-Packard PA 8000 RISC CPU: A High Performance Out-of-Order Processor," *IEEE Micro*, April 1997.
 3. J. Lotz, G. Lesartre, S. Naffziger, and D. Kipp, "A Quad-Issue Out-of-Order RISC CPU," *ISSCC Digest of Technical Papers*, February 1996.
 4. P. Perez, "The PA 8200: A High-Performance Follow-On to the PA 8000," *Microprocessor Forum*, October 1996.
 5. N. Gaddis, J. Butler, A. Kumar, and W. Queen, "A 56-Entry Instruction Reorder Buffer," *ISSCC Digest of Technical Papers*, February 1996.
 6. W.R. Bryg, K.K. Chan, and N.S. Fidducia, "A High-Performance, Low-Cost Microprocessor Bus for Workstations and Midrange Servers," *Hewlett-Packard Journal*, Vol. 47, no. 1, February 1996, pp. 18-24.
-
-