The PA 7100LC Microprocessor: A Case Study of IC Design Decisions in a Competitive Environment

Engineering design decisions made during the early stages of a product's development have a critical impact on the product's cost, time to market, reliability, performance, and success.

by Mick Bass, Patrick Knebel, David W. Quint, and William L. Walker

In today's competitive microprocessor market, successful design teams realize that flawless execution of product development and delivery is not enough to ensure that a product will succeed. They understand that defining the correct feature set for a product and creating design methodologies appropriate to implement and verify that feature set are just as important as meeting the product schedule.

The design decisions that engineers and managers make while defining a new product have a critical impact on the product's cost, time to market, reliability, performance, future market demand, and ultimate success or failure. Engineers and managers must make trade-offs based on these factors to decide which features they should implement in a new product and which they should not. Further, they must plan their product development effort so that the methodologies by which they develop their product are sufficient to ensure that they are able to implement the product definition within the required cost, schedule, and performance constraints.

Design choices arose frequently while we were defining and implementing the PA 7100LC microprocessor.[1] We were targeting the PA 7100LC to be the processing engine of a new line of low-cost, functionally rich workstation and server products. Our design goals for the CPU were to provide the system performance required for our target market at an aggressively low system cost and to deliver the CPU on a schedule that would not delay what was to become HP's steepest computer system production ramp to date. Fig. 1 shows a simplified block diagram of the PA 7100LC processor.

To meet these goals required that we sometimes had to shift our focus from the CPU to the impact of a particular feature upon performance and cost at the system level. Hewlett-Packard's position as a vendor of both microprocessors and computer systems allowed us to use this technique with much success.[2,3] Even with this focus, however, the correct

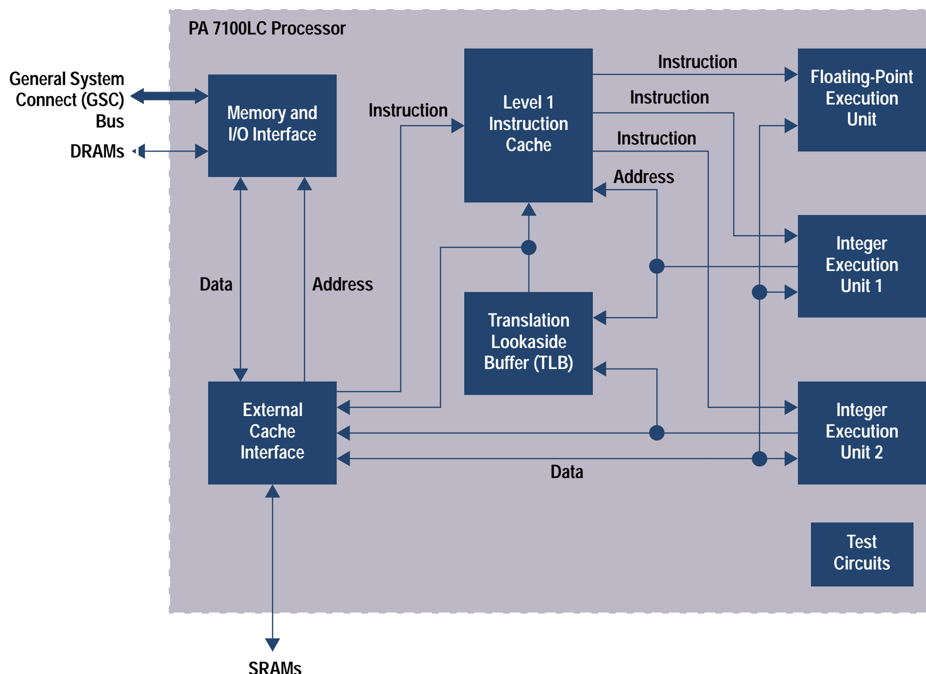


Fig. 1. A simplified block diagram of the PA 7100LC processor.

© Hewlett-Packard Company 1995

decision could be far from obvious. We often identified several alternative implementations of a particular feature, each with its own impact on cost, schedule, and performance. Trading these impacts against one another proved very challenging. Design decisions also impacted each other, with the outcome of one serving as a critical input to others. The effect of a decision, for this reason, was sometimes much larger than would have appeared at first glance. Sometimes decisions created additional requirements, either for new features or for new support methodologies. All of these factors played together to underscore the fact that it was critical to our product's success to have a decision process that worked well.

We knew that a good definition of the PA 7100LC would require that we make feature decisions in several areas, including:

- Cache organization
- Number of execution units and superscalar design
- Pipeline organization
- Floating-point functionality
- Package technology
- Degree of integration
- Multimedia enhancements.

We also knew that we needed to select development methodologies consistent with the feature decisions that we made. Product features and required design methodologies are often strongly connected. We couldn't consider the benefits of one without the costs of the other, and vice versa. Methodologies that were impacted by our feature-set decisions included:

- Synthesis
- Place and route
- Behavioral simulation
- Presilicon functional verification
- Postsilicon functional and electrical verification
- Production test.

These methodologies are discussed in the article on page 23.

The cumulative effects of our decisions led to the creation of a low-cost, single-chip processor core that includes a built-in memory controller, a combined, variable-size off-chip primary instruction and data cache, a 1K-byte on-chip instruction buffer, and a superscalar execution unit with two integer units and one floating-point unit. We reduced the size and performance of the floating-point unit, which we had leveraged from the PA 7100 processor.[4,5] We added I_{DDQ}, sample-on-the-fly, and debug modes to enhance testability, reduce test cost, and accelerate the postsilicon schedule. We tailored the methodologies by which we created the chip to match the features that we had decided upon.

This article provides examples of our decision-making process by exploring the decisions that we made for several of the features listed above. In each case, we present the alternatives that we considered, the costs and benefits of each, and the impact on other features and methodologies. We discuss our decision criteria. Since we strive to continually improve our ability to make good design decisions, we also present, wherever possible, a bit of hindsight about the process. In most cases, we still believe that we selected the correct alternative. However, if this is not the case, we discuss what we have learned and the modifications we made to our process to incorporate this new knowledge.

The Design Decision Process

Most design decisions ultimately come down to trade-offs between cost, schedule, and performance. Unfortunately, it is often difficult to determine the true cost, schedule, or performance for the wide variety of implementations that are possible. And since these three factors most often play against each other, it is necessary to make sacrifices in one or two of the areas to make gains in the others.

The cost of a processor core is determined by the cost of silicon die, packaging, wafer testing, and external SRAM and DRAM. Breaking this down, we find that cost of a die is determined by the initial wafer cost and the defect density of the IC process being used. Wafers are more expensive for more advanced processes because of higher equipment, development, and processing costs. The die packaging costs are determined primarily by package type and pin count. Large-pinout packages can be very expensive. An often ignored cost is the tester time required to determine that a manufactured part is functional. Reducing the time needed for wafer and package testing directly reduces costs. Finally, SRAM and DRAM costs are determined by the number, size, and speed of the parts needed to complete the design.

The schedule of a project is determined by the complexity of the design and the ability to leverage previous work. Each design feature requires certain time investments and has associated risks. Time is required for preliminary feasibility investigations, design of control algorithms, implementation of circuits, and presilicon and postsilicon verification. Schedule risks include underestimation of time requirements because of unexpected complexity and the extra chip turns required to fix postsilicon bugs associated with complex design features.

Performance is conceptually simple, but because of the intricacy of processor design it is often difficult to measure without actual prototypes. HP has invested heavily in performance simulation and analysis of its designs. Results from HP's system performance lab were invaluable in making many of the design decisions for the PA 7100LC. By supporting a detailed simulation model of each processor developed by HP, the system performance lab is able to provide quick feedback about proposed changes. HP also uses these models after silicon is received to help software developers (especially for compilers and operating systems) determine bottlenecks that limit their performance.

Engineers at the system performance lab design their processor simulators in an object-oriented language to allow easy leverage between implementations. All processor features that affect performance are modeled accurately by close teamwork between the performance modeling groups and the hardware design groups. As the hardware group considers a change to a design, the change is made in the simulator, and simulations are done to allow simple comparisons that differ by only a single factor. This is continued in an iterative fashion until all design decisions have been made,

© Hewlett-Packard Company 1995

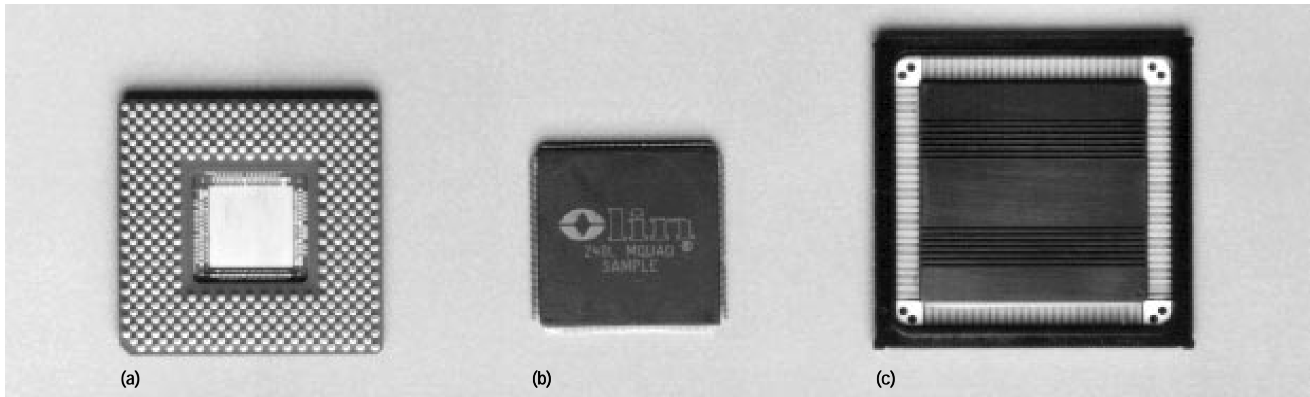


Fig. 2.* (a) 432-pin ceramic pin grid array (432-CPGA). (b) A 240-pin MQUAD and (c) a 304-pin MQUAD.

* The CPGA package is manufactured by Kyocera Inc. and the MQUAD packages are manufactured by Olin Interconnect Technologies.

after which we are left with a simulator that matches the hardware to be built.

Without performance simulations, it would be very difficult to estimate performance for a proposed implementation. Even something as simple as a change in operating frequency has effects that are difficult to estimate because of the interactions between fixed memory access times and processor features. As processor frequency increases, memory latencies increase, but this increased latency is sometimes (but not always) hidden by features such as stall-on-use. Stall-on-use allows the processor to continue execution in the presence of cache misses as long as the data is not needed for an operation. These interactions make accurate hand calculations impossible, creating a need to use simulations for comparing many different implementation options.

The performance simulations are based on SPEC and TPC benchmarks. While these benchmarks are useful for gathering performance numbers and making comparisons, they do not tell the whole story. Many applications are not represented by the benchmarks, including graphics, multimedia, critical hand-coded operating system routines, and so on. When evaluating features related to these applications, we work directly with people in those areas to analyze the impact of any decisions. Often this involves analyzing by hand critical sections of the code (e.g., tight loops) to evaluate the overall performance gain associated with a feature. For the PA 7100LC, this was especially true for the multimedia features.

The ability to quantify the impact of proposed features on cost, schedule, and performance was paramount to our ability to make sound design decisions.

Integration

The first design decisions that we made were related to the high-level question "How highly integrated should we make the chip?" This led to the questions: Should we include an on-chip cache or not? If so, how large should it be? If we have an off-chip cache, how should we structure it? How should the CPU connect to memory and I/O? Should the memory controller be integrated or not?

The primary question was whether the CPU, cache, and memory system should live on a single die in a single package, or whether we should partition this functionality onto two or more chips.

The trade-offs involved in this decision were numerous. Die cost would increase for a multichip solution. Package cost would vary with the partitioning that we chose, as would package type and maximum pin count. Required signal-to-ground ratios would vary with package type, which would either limit the signal count or require more pins (at a higher cost). Performance, design complexity, and schedule risk would be greatly impacted by the partitioning decision.

To sort out these trade-offs, we started with a packaging investigation that quantified cost, performance, and risk for different packaging alternatives. This investigation yielded a preferred package: a 432-pin ceramic pin grid array see (Fig. 2a). This package, with its large signal count, could accommodate the extra interfaces required to include a memory controller, an I/O controller, and an external cache controller.

The memory controller and cache investigations were tightly coupled. Performance simulations always included features from both subsystems because small changes in the behavior of one subsystem could drastically affect the performance of the other. In the end we realized that the performance gains brought by an integrated memory controller enabled smaller, cheaper caches without sacrificing overall performance. This realization drove the development of the cache subsystem.

Package Selection and CPU Partitioning. We targeted the IC package design with the objective of minimizing system cost with little compromise in performance. The customary package for CPU chips is either a quad flat pack (QFP) or a pin grid array. The QFP is a plastic, low-profile package with gull-wing connections on four sides. The QFP is inexpensive and easy to mount on a printed circuit board and has gained acceptance rapidly for surface mounting to printed circuit boards. It has the disadvantage that the number of pins is limited. Pin counts above 200 are fragile and difficult to keep

© Hewlett-Packard Company 1995

coplanar for surface mounting. The package also has very limited ability to dissipate heat because the chip is encased in plastic. A recent improvement to this package sandwiches the chip between two pieces of aluminum, which can dissipate up to four watts of heat (ten watts with a heat sink). It was this metal quad, or MQUAD, that became a candidate for a low-cost package for our high-performance CPU. HP's package of choice for previous CPUs has been the ceramic pin grid array, a complex brick of aluminum oxide and tungsten built in layers and fired at 2000°C. The PGA used for the PA 7100 processor (the basis for the PA 7100LC) was a 504-pin design that incorporated the following advanced features:

- A tungsten-copper heat-conducting slug for superior thermal conductivity to the heat sink
- Ceramic chip capacitors mounted on the package for power bypassing
- Thin dielectric layers that minimized power supply inductance
- Use of 0.004-in vias internally (most are 0.008-inch).

This package performed its thermal and electrical duties very well, but its cost had always been an issue.

Our strategy to develop a low-cost CPU coupled chip partitioning options with the packaging options of using either two low-cost MQUAD packages or placing a single large chip in a PGA. The two-chip CPU could be placed in one 240-pin and one 304-pin MQUAD (see Figs. 2b and 2c). The other alternative was to place a larger integrated chip in a single 432-pin PGA (see Fig. 2a). The first cost estimate assumed that the PGA would be priced similarly to the 504-pin package. The total cost of both MQUAD chips was initially thought to be about 75% less than the PGA estimate. This would seem to indicate that the MQUAD would be the definite candidate to meet our low-cost goals. However, that perception changed as our investigation continued.

We didn't expect the MQUAD's electrical performance to match that of the PGA because the MQUAD we were considering had only one layer of signals and no ground planes. Ground planes can be used to shield signal traces from each other and reduce inductances of signals and power supplies. The PGA could incorporate several ground planes if necessary. On the other hand, the MQUAD package can only approach the shielding effect of the ground planes by making every other lead a ground, which severely limits the number of usable signals. Gaining a lower package price by using the MQUAD would require redesigning the I/O drivers specifically to reduce rise times and thereby control crosstalk and power supply noise.

The PA 7100 PGA's electrical performance exceeded the needs of this chip, so the strategy shifted to trading away excess performance to gain lower cost. The number of power and ground planes was reduced to two. The design was also modified to optimize performance without using package-mounted bypasses or thin dielectric layers. The PGA design was reduced to four internal metal layers with no bypassing, no thin dielectric layers, and no 0.004-in vias, all of which reduced cost compared to the 504-pin PGA mentioned above.

The power dissipation of the chips would also have been an issue for the MQUADs. Heat sinking to further improve the thermal resistance of the packages might have been required. CPU designs are often upgraded to higher clock speeds after first release, so if package heat dissipation is marginal, upgrade capability is jeopardized. (Typically, power dissipation is proportional to operating frequency.) The 504-pin PGA had already been used to dissipate 25 watts, which left an opportunity for cost-saving modifications. With the thermal margin in mind, two design changes were investigated, one to use a lower-cost copper-Kovar-copper laminate heat spreader, and the other to eliminate the heat spreader entirely. The first option was dismissed because of failures found during a low-temperature storage test. (The laminate heat spreader detached from the ceramic body because of a disparity in thermal expansion rates.) The second option was also dismissed when the thermal resistance of the ceramic carrier was found to be too high.

The time schedule for the completion of reliability testing and manufacturing feasibility studies had to be considered when evaluating the two technologies. The PGA was a mature technology with considerable experience behind it, and the time schedule and results of the testing could be determined with some certainty. The MQUAD was a new technology by contrast. The design was solid, but had several new features that were untested in terms of long-term reliability. Despite the strong desire to exploit new technology, the schedule risk was a significant factor.

By the time the partitioning decision was to be made, the PGA cost had shrunk to almost half of its original cost, the 304-pin MQUAD was presenting schedule risks, and both MQUADs had marginal power dissipation. Possibly most important, the PGA provided a robust solution with thermal and electrical margins. The cost difference was still significant, but the PGA provided a flexibility to the chip designers that offset its disadvantages. Thus, the PGA package was chosen for the PA 7100LC.

Memory Controller Destiny. Whether or not to integrate the memory and I/O controllers onto the CPU die was one of the most direction-forming decisions that we made. To decide correctly, we had to consider the effects of integration on factors such as multiprocessor capability, system complexity, memory and I/O controller design complexity, die cost, memory system performance, and memory system flexibility.

Traditional multiprocessor systems have a single main memory controller and I/O controller (see Figs. 3a and 3b). These controllers maintain connections to the multiple processors. Systems organized in this way separate the memory and I/O controllers from the CPU. This organization allows users to upgrade entry-level systems to include multiple processors at the expense of reducing the memory and I/O performance of uniprocessor systems and adding significant complexity to both the memory and cache controllers.

Our design goals focused on maximizing uniprocessor performance. HP was already shipping desktop multiprocessor systems built around the PA 7100 microprocessor at the time we were making these decisions. The market segment that we were targeting for the PA 7100LC demanded peak uniprocessor performance at a low system cost. Since our target market didn't require multiprocessing as a system option, we directed our efforts toward the benefits that we

© Hewlett-Packard Company 1995

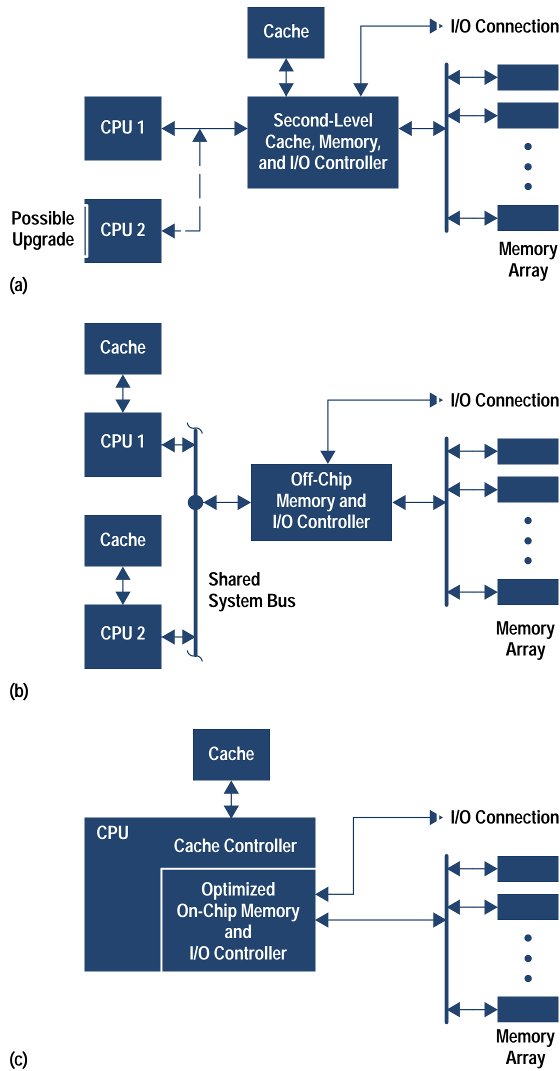


Fig. 3. (a),(b) Multiprocessor architectures in which the memory and I/O controller are separate from the CPUs. (c) A uniprocessor system in which the memory and I/O controller are integrated into the CPU chip.

could bring to a system through a focused uniprocessor design.

Integrating the memory and I/O controller with the CPU in a uniprocessor system (Fig. 3c) can have a dramatic effect on reducing cache miss penalties by decreasing the number of chip boundaries that the missing data must cross and by allowing the memory and I/O controller early access to important CPU internal signals. Miss processing on the memory interface can effectively begin in parallel with miss detection in the cache controller. An integrated memory controller can even use techniques such as speculative address issue to begin processing cache misses before the cache controller detects a cache miss.

The reductions in CPI (cycles per instruction) that we could achieve by integrating the memory controller allowed us the degrees of freedom that we needed to explore certain cache architectures in greater detail. Some of these architectures are described in the next section.

System complexity is reduced with an integrated memory and I/O controller. The 432-pin CPGA that we were considering for an integrated design had sufficient signal headroom to enable separate, dedicated memory and I/O connections. A two-chip approach, using the lower-cost MQUAD packages, would be forced to share pins between the memory and I/O connections to accommodate the low signal count of the MQUAD package, which would increase system complexity.

An integrated memory and I/O controller also simplifies the interface to the CPU. Since this interface connects two entities on the same die, signal count on the interface became much less important, which allowed us to simplify the interface design considerably.

On the down side, integrating a memory and I/O controller required enough flexibility in its design to satisfy the broad range of system customers that our chip would encounter. However, this requirement also exists for a nonintegrated solution. Historically, system partners have not redesigned memory controllers that the CPU team has provided as part of a CPU chipset. HP's advantage of providing both processors and systems has allowed us to work closely with system designers and enabled us to meet their needs in both integrated and nonintegrated chipsets.

In summary, integrating the memory and I/O controller onto the CPU core introduced a gain in performance, a reduction in complexity and schedule risk, and several possibilities for reduced cost in the cache subsystem. These were the compelling reasons to move the memory controller onto the CPU die and continue exploring cache alternatives and optimizing memory system performance.

Cache Organization. One of the distinguishing characteristics of HP PA-RISC designs over the past several implementations has been the absence of on-chip caches in favor of large, external caches. While competitors have dedicated large portions of their silicon die to on-chip RAMs, HP has continued to invest in aggressive circuit design techniques and higher pin count packages that allow their processors to use industry-standard SRAMs, while fetching instructions and data every cycle at processor frequencies of 100 MHz and above. This has allowed our system partners to take a single processor chip and design products meeting a wide range of price and performance points for markets ranging from the low-cost desktop machines to high-performance servers. For example, the PA 7100 chip has been used in systems with cache sizes ranging from 128K bytes to 2M bytes and processor frequencies ranging from 33 MHz to 100 MHz.

The main design goals for the PA 7100LC were low cost and high performance. Unfortunately, high-performance systems use large, fast, expensive caches. Obviously, trade-offs had to be made. As with previous implementations, the designers started with a clean slate and considered various cache options, including on-chip cache only, on-chip cache with an optional second-level cache, split instruction and data off-chip caches, and combined off-chip caches (see Fig. 4). Ultimately, the cache design was closely linked to the memory controller design because of the large effect of memory latency on cache miss penalties.

© Hewlett-Packard Company 1995

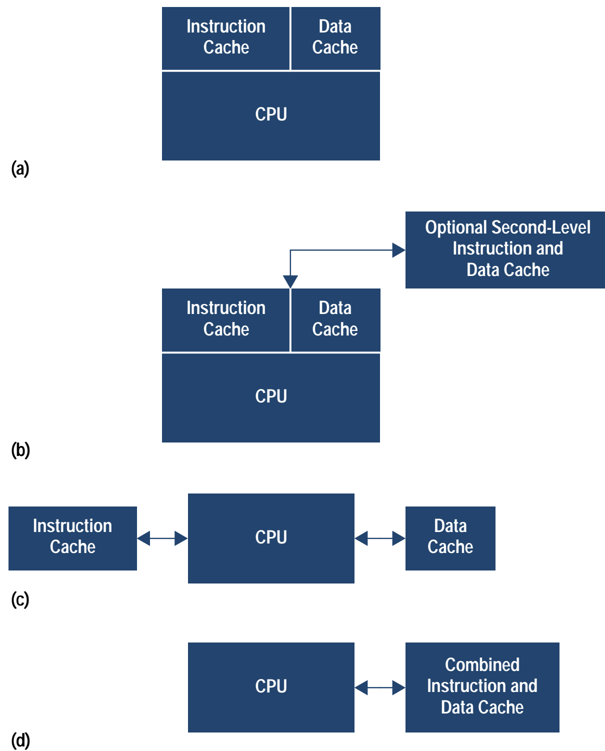


Fig. 4. Different cache organizations. (a) On-chip cache. (b) On-chip cache with an optional second-level cache. (c) Split instruction and data off-chip caches. (d) Combined off-chip caches.

On-chip caches have the obvious advantage that they can allow single-cycle loads and stores at higher chip frequencies than are possible with many off-chip cache designs. They also allow designers to build split and associative cache arrays which would be prohibitive for off-chip designs because of the large number of I/O pins required. Unfortunately, in current technologies on-chip caches tend to be fairly small (8K bytes to 32K bytes) and even with two-to-four-way associativity, they have higher miss rates than larger (64K bytes to 256K bytes) direct-mapped, off-chip caches. Also, on-chip caches require a substantial amount of chip area, which translates to higher costs, especially for chips using leading-edge technology with high defect densities. This extra chip area also represents lost opportunity cost for other features that could be included in that area. Examples include an on-chip memory and I/O controller, graphics controller, more integer execution units, multimedia special function units, higher-performance floating-point circuits, and so on.

Another drawback of on-chip caches is their lack of scalability; providing multiple cache sizes requires fabricating multiple parts. To overcome this limitation designers can allow for optional off-chip caches. The off-chip caches can range in size and speed and can provide flexibility for system designers looking to meet different price/performance choices. Low-end systems need not include the off-chip cache and can be built for a lower cost. High-end systems can get a performance boost by paying the extra cost to add a secondary off-chip cache. For most systems, the cost for this flexibility is added pin count to allow for communication with the off-chip caches. Other systems might be able to multiplex the cache lines onto some already existing buses such as the memory bus.

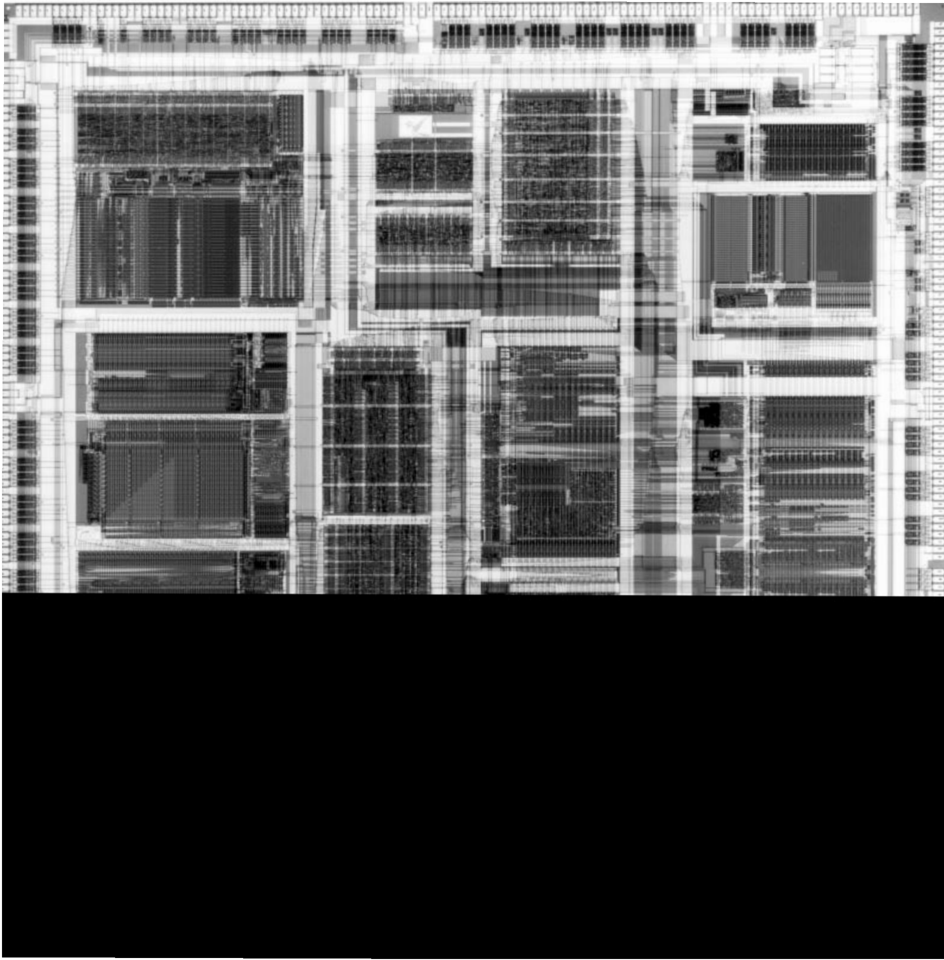
For the PA 7100LC, we determined that a primary on-chip cache would cost too much in terms of more expensive technologies, increased die size, and the lost opportunity of putting more functionality on the chip. Without a primary on-chip cache, we were able to design a processor with two integer units, a full floating-point unit including a divide and square root unit, and a memory and I/O controller. We achieved this functionality using only 905,000 FETs in 0.8 micrometer (CMOS26) technology on a die measuring 1.4 cm by 1.4 cm (see Fig. 5). CMOS26 is a mature HP process that has been used for several processor generations. As a result, it has a low defect density and thus, a low cost. A processor with an on-chip cache would have required a more advanced technology having higher wafer costs and defect densities. Of course, without an on-chip cache, we were challenged to design a low-cost off-chip cache that allowed accesses at the processor frequency.

HP's previous implementations of PA-RISC were built with independent instruction and data caches made up of industry-standard SRAMs (see Fig. 4c). It would have been easy to leverage the independent direct-mapped instruction and data cache design from the PA 7100, but we were determined to find a less expensive solution. Independent cache banks require a high pin count on the processor chip because each bank requires 64 data pins and about 24 pins for tag, flags, and parity. Thus, combining instructions and data into a single set of cache RAMs (Fig. 3d) saves about 88 pins on the processor chip. These extra pins directly affect packaging costs. Also, providing split caches requires using more SRAM parts in a given technology. Systems based on the PA 7100LC with a combined cache require only 12 SRAM parts using ×8* technology. By leveraging the aggressive I/O design from previous implementations, the PA 7100LC can access 12-ns SRAM parts every cycle when operating at frequencies up to 66 MHz. Since 8K × 8, 12-ns SRAMs are commodities in today's market, the cost of a 64K-byte cache subsystem for a 60-MHz PA 7100LC is comparable to the price we would have paid for a much smaller on-chip cache.

Combined instruction and data caches have one large drawback. Since the PA 7100LC processor can consume instructions as fast as the cache can deliver them, there is little or no cache bandwidth left to satisfy load and store operations. To solve this problem, we needed to implement some type of instruction buffer on the processor chip. A large instruction buffer would have all the drawbacks of the on-chip cache design discussed above, so we were determined to find a way to achieve the desired performance with a small buffer. We knew we would need a mechanism to prefetch instructions from the off-chip combined cache into the dedicated on-chip buffer during idle cache cycles. Thus, we started with a standard direct-mapped 2K-byte buffer and simulated various prefetch and miss algorithms. As expected, we found that performance was extremely sensitive to the buffer miss penalty, which ranges from zero to two states

* RAM sizes are quoted in depth by width (i.e., 64K ×8 is 65,536 deep by 8 bits wide).

© Hewlett-Packard Company 1995



 © Hewlett-Packard Company 1995