

High-Performance Processor Design Guided by System Costs

To minimize time to market and keep costs low, the PA 7300LC design was leveraged from a previous CPU, the chip area was reduced, cache RAM arrays with redundancy were added, and high-speed, high-coverage scan testing was added to reduce manufacturing costs.

by David C. Kubicek, Thomas J. Sullivan, Amitabh Mehra, and John G. McBride

While designing the PA 7300LC processor, the CPU team had to make design trade-offs between time to market, performance, and manufacturing costs. Occasionally these seemingly contradictory goals worked together to drive the team to a decision. More often, however, the team had to make hard decisions, weighing the benefits of each of the design goals.

This paper discusses the strategies used by the PA 7300LC physical design team to implement the design goals for the PA 7300LC.

Design Goals

One of the factors driving the design process was the desire to bring the product to market as fast as possible. To accomplish this goal, we employed three major strategies:

- Leverage as much as possible from the previous HP processors, including hardware, software, and methodologies for design and test
- Design quality into phase one, or the presilicon design stage, so that there would be fewer iterations of the design during phase two, after the first tape release
- Monitor project progress, avoiding any obstacles that might seriously impact or threaten our schedule.

Keeping the cost of the system as low as possible was another important goal of the project. Systems based on the PA 7300LC are meant to position HP in the low-to-midrange workstation market where prices are set by competition, not system cost. Therefore, savings in the system cost have a big impact on profit. To meet these aims, the team decided to:

- Integrate the first-level cache, a major system cost, into the processor, which had never been done before in an HP microprocessor
- Integrate the memory and I/O controller (MIOC), creating a system on a chip
- Reduce chip area to lower cost
- Add redundancy to the SRAM arrays on the chip, allowing some process defects to be repaired, thereby saving chips that would otherwise be thrown out
- Provide high-coverage, high-speed scan testing to lower the manufacturing cost of the processor.

Designs Leveraged to Minimize Time to Market

To reduce the time to market for the PA 7300LC, the CPU physical design team decided to leverage as many circuits as possible from the PA 7100LC. Except for the process shrink from CMOS26 to CMOS14, much of the superscalar integer data path on the PA 7300LC was leveraged from the PA 7100LC unchanged. Also, many of the cells used in the integer data path were used in other data path blocks on the chip. Although some of the circuits were reworked for speed improvements, the floating-point unit was also highly leveraged from the PA 7100LC. Furthermore, the floating-point unit was used in the geometry accelerator chip for the Visualize 48XL graphics product. This leverage strategy not only helped reduce time to market, but also split the design costs associated with the circuit between the ASIC and the CPU.

Control Blocks

While all of the control blocks leveraged from the PA 7100LC required some changes, much of the original control logic remained intact or was at least similar to the original code. This provided the opportunity to start the physical design early, providing the designers with the chance to work the bugs out of the tool flow, begin composition, and provide early feedback on difficult timing paths to the control designers.

For physical circuit layout, the control physical team initially used data scaled from the PA 7100LC in the CMOS26 process to the CMOS14 process. In several cases, the final artwork was almost entirely based upon the floor plan scaled from the

PA 7100LC. In other cases, the control equations were either vastly different (memory I/O control) or entirely new (the cache controllers), so we were unable to take advantage of earlier work.

In the case of the three main integer control blocks, the timing information and a significant portion of the control equations were usable. However, a study of interconnect between the three blocks indicated that they could be combined into a single block to simplify the design from a timing standpoint and to use global routing resources efficiently. By moving several hundred signals away from the center of the die into a more localized area near the integer data path, we also saved significant area.

Core Logic Library. While much of the logical design of the PA 7300LC was leveraged from the PA 7100LC, most of the standard cell libraries were borrowed from the PA 8000 project. The PA 8000 was fabricated using the same IC process technology as the PA 7300LC, but was farther along in the design cycle. The PA 7300LC team was able to use almost the entire PA 8000 core logic library unchanged. Unfortunately, a different clocking strategy meant that the driver library needed significant rework.

Standard Cell-Based Control Block Design. The use of a standard cell-based design for the control blocks, which was leveraged from the PA 7100LC, allowed great flexibility when fixing functional bugs, both in phase one (presilicon) and in phase two (postsilicon). During phase one, the standard cell approach permitted fairly quick turnarounds of a control block for rather complex changes. Often all that was required of the physical designer was to rerun the synthesis and routing tools, apply a few hand changes, and verify the design.

Use of Spare Gates. During the very late stages of phase one and all of phase two, the use of spare gates in the standard cell blocks allowed the physical designers to make logical changes by changing only the metal layers. One very late bug fix was made between the time the lower-level masks (e.g., diffusion, well, polysilicon) and the higher-level metal masks were released to the mask shop. Additionally, when phase two bugs were found, we were able to use the spare gates for metal-only changes. Because a number of wafers were held in the fabrication shop before M1 (the lowest level of metal) was placed, metal-only changes were run through the fabrication shop very quickly since the lower layers were already processed.

FIB Process. Another advantage of the metal-only changes was exploited during phase two. As control bugs were uncovered, we were able to rewire the logic using spare gates and the FIB (focused ion beam) process. The FIB process uses an ion beam to cut and expose various metal lines on a functional chip and to deposit platinum, reconnecting the gates into a new logic structure. A typical FIB repair is illustrated in Fig. 1. Use of the FIB process allowed the design team to verify bug fixes in a system that often ran at full operating speed. This resulted in a more complete functional verification, since tests run much faster in real silicon than in simulation.

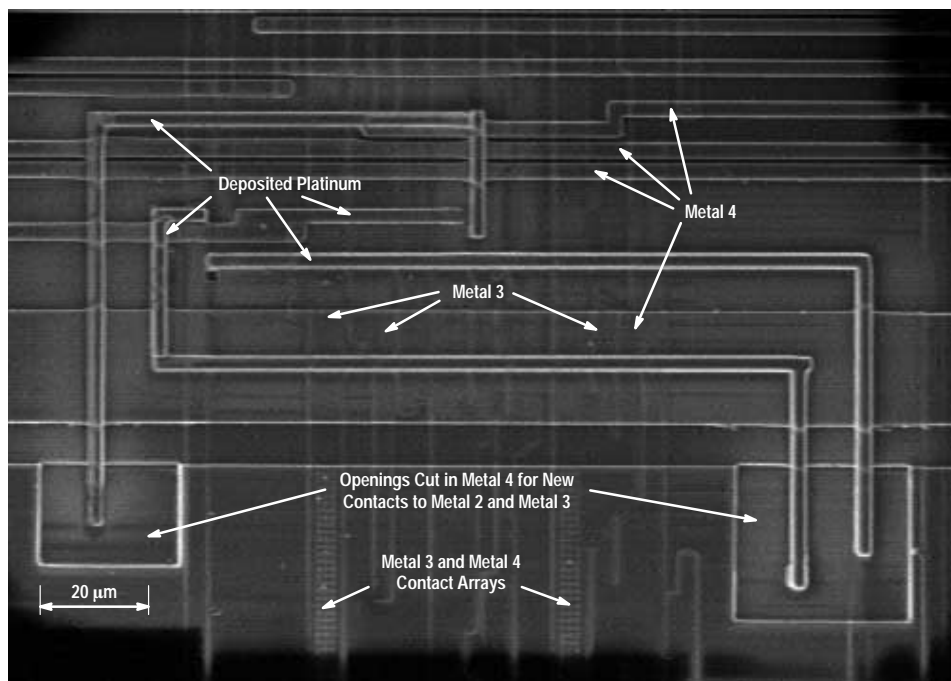


Fig. 1. The photomicrograph shows a typical FIB (focused ion beam) repair. For this FIB repair, portions of the circuit of interest were located under a metal four power bus. Therefore, openings had to be cut through the power bus to access the circuits below. Notice how the platinum deposited by the FIB runs over top of the metal four, separated by passivation.

New Tools. While the synthesis tool (Synopsys) and routing tool (Cell3 from Cadence Systems) were the same as on the PA 7100LC project, newer versions of these tools with additional features and problems were employed. The ability to work with the tools at an early stage allowed the physical control design team the chance to learn the strengths and weaknesses of the tools, so that they could be exploited or compensated for once full functionality was reached in the control equations. Even though new versions of these tools presented a few new problems, the basic method of operation was the same as for the PA 7100LC. Thus, use of these tools helped reduce our time to market by leveraging our previous experience with them.

Phase One Quality Equals Reduced Phase Two Debug Time

In addition to leveraging designs and methodologies, correct balance between the time and resources spent ensuring phase one quality and the time and resources spent finding functional problems in phase two can also reduce time to market. In this project, we gave great weight to ensuring phase one quality, since this would make debugging much easier in phase two. In return for our investment, the PA 7300LC had one of the shortest and smoothest phase two periods of all CPUs designed by HP.

Debugging Trade-offs

In phase one of the design cycle, simulation, emulation, and hand analysis are the key tools of the designer. With these tools, the designer can examine every detail of the design at any chip state and under any conditions.

In phase two, tests can be run much faster on a real chip than in simulation, accelerating bug detection. However, root-cause analysis of problems is a slow and difficult process because virtually all signals are hidden from the scrutiny of the designer. In addition, the signals that are available for the designer to examine are either chip I/Os, or are only indirectly available through scan paths. Hence, electrical phenomena such as glitches and power supply droops are not easily observed. Therefore, phase one debugging is a much simpler process because of the availability of detailed data about the internals of the chip.

Cross-Checking Designs to Improve Phase One Quality

To ensure phase one quality, each design on the PA 7300LC was subjected to a series of computerized automatic design checks, and then a set of manual checks was performed by designers who were not involved with the original design. These checks looked at:

- Circuit topologies
- FET size and connections
- Wire size
- Power routing
- Clock signal routing
- Signal coupling
- Signal types and timing.

Automated and Human Checks

For automated tests, the computer applied the same rules to each node on the chip quickly, without the bias that a human may have had. However, the computerized checks often generated spurious error messages and required significant human intervention to identify the real problems. Also, many rules were beyond the scope of computer algorithms and required human checks.

An example of a simple computerized check used on the PA 7300LC is a signal edge rate check. Every signal on the chip was checked against a set of criteria that depended on the signal context. The computer program blindly reported any signal that violated the specification set for that type of signal. It was the job of the designers to determine which errors flagged by the computerized check were real problems. The designers then fixed real errors and waived all others. Obviously, with this and all other automatic checks, a certain amount of skill and experience is needed to judge what constitutes a potential problem and what does not.

Because some quality checks do not lend themselves easily to computerized checking, each cell, subblock, and major block of the CPU had to be examined by an experienced engineer who was not the designer of the block. The cross-checking engineer had a list of guidelines to follow for checking each design, and any variance from these guidelines was discussed with the designer. This checklist was broken into categories so that the cross-checking engineer could focus on one particular area at a time, such as schematics, artwork, test, and so on.

An example of a check that is not automated is an artwork check, which ensures that all circuits have very solid power and ground networks. The subjective nature of this check makes it very difficult to implement with a computer check. Also, because of the subjective nature, the checking engineer must be very diligent about what constitutes a solid supply net and what does not.

Circuit Timing

When operating with a clock period of only a few nanoseconds, timing is of utmost importance as a phase one quality issue. Several different tools were used to this end, most notably Cadence's Veritime, EPIC's Pathmill, and HP SPICE (see Fig. 2).

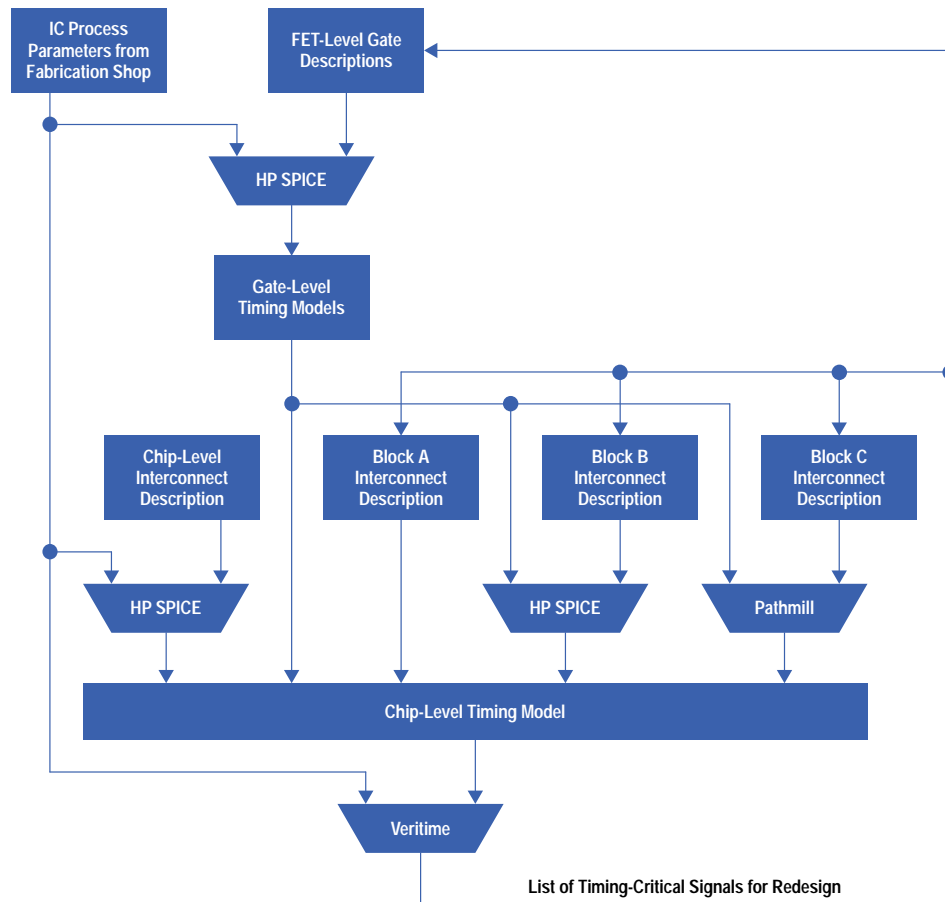


Fig. 2. The chip modeling tools used for the timing simulation.

Chip Model Tools. A Veritime model was generated and maintained for the top level of the chip. This model included either gate-level descriptions of blocks (generally for the standard cell blocks) or black box descriptions of blocks (for the custom data path blocks), as well as models for the delay due to the interconnects between blocks. On a regular basis, the timing team updated the model and performed timing analysis. The results were then given to the various block owners, who redesigned slow portions of critical timing paths.

HP SPICE and EPIC's Pathmill were used by a number of the custom data path designers to generate black box models of their blocks for the global Veritime model. Also, some designers analyzed larger standard cell blocks with Veritime. Additionally, a tool was developed that estimated the delays of all signal routes, which could then be hand-checked for anomalies.

Finally, HP SPICE was used extensively to simulate the timing of all major buses, many top-level routes, and other timing-critical paths. All elements of the standard cell libraries were also characterized with HP SPICE, using conservative parameters. While this approach caused a few more phase one headaches for the control designers, we uncovered no timing issues for standard cell blocks during phase two characterization.

Chip Composition Focused on Minimizing Cost

One of the ways that we were able to drive down the cost of systems that incorporate the PA 7300LC was to reduce the die size, thereby allowing more die per wafer in fabrication and improving yield. This was a key focus of the physical design group, and resources were dedicated to monitoring the impact of all changes on the manufacturing cost of the chip. We took several steps during phase one to ensure that the PA 7300LC was as small as we could reasonably expect.

Global Floor Planning. We started global floor planning and routing early in the design phase. Our initial floor plans, although they bear little resemblance to the final chip floor plan, provided the groundwork for early estimates on die size and feasibility. One of the early decisions was whether we would use three layers of metal, as on the PA 7100LC, or add a fourth metal layer. After extensive analysis, we concluded that, with only three metal layers, our stepper size would limit us to a

very small first-level cache, which would not meet our performance targets. So, we added the fourth metal layer. As it turned out, the fourth metal layer was essential to the success of the project for many other reasons, even though the decision was made over a year before tape release.

As the design matured, the floor plan and routes kept up with the changes and provided feedback on die size and potential timing problems. Fig. 3 shows the final floor plan. We made several major compositional changes early solely to remove congestion on the top metal layer and to compact the die area. These compositional changes included movement of a block in the data cache and changing the aspect ratio of our pad-ring bitslice.

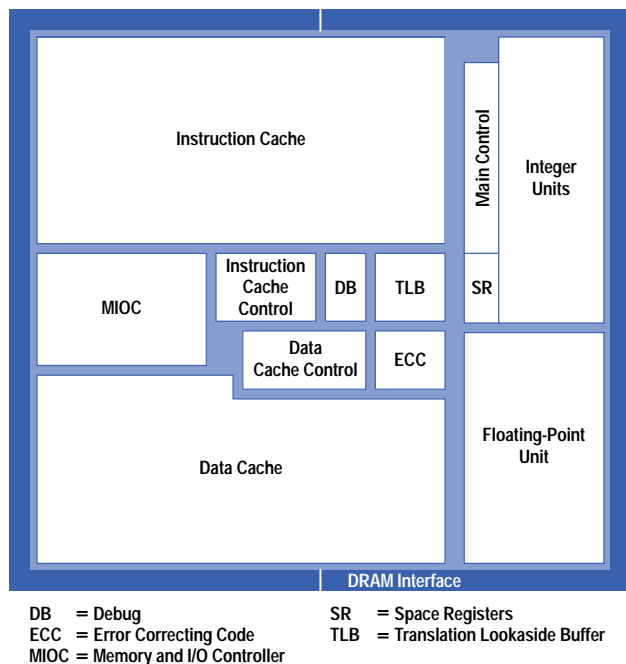


Fig. 3. The final PA 7300LC floor plan.

Using the “Dirty Trick.” One of the opportunities we saw was in the composition of the data cache. Originally the data cache was designed to be completely symmetrical, with a right side, a left side, and a data path in the middle to merge the data from the two sides. The design essentially had three blocks on each side stacked from bottom to top: the data RAM array, the tag RAM array, and the dirty block array.

As we started routing the chip in its early phases, we saw that we had much more routing congestion in the channels above the right side of the data cache than above the left side. The channels above the right side led to the integer and floating-point units, while those above the left ran towards the memory and I/O controller (MIOC) (see Fig. 4a). The congestion on the right side of the data cache would have increased the height, while leaving unused area above the left side.

To deal with this congestion problem, we employed what we called the “dirty trick.” The dirty bit block in the data cache is used to store one bit of information for each line in the cache. This bit tells the processor whether the information contained in that cache line has been modified by the CPU and is therefore dirty. In our original conception, each side of the cache had its own dirty block, which consisted of one bit of information per cache line and an address-to-cache-line decoder, the latter being ten times larger.

By putting both dirty data bits on the left side of the data cache and sharing one address decoder, the left side of the data cache grew by one bit of information, but the right side shrank by one bit of information and one address decoder (see Fig. 4b). This was a big win as it allowed the die size to be shrunk by the height of the dirty block. If we had not floor planned and routed early in the design phase, we would not have seen this opportunity in time to act on it and reduce the die size.

Outer Ring of Pads Limited. We were not in the clear yet, however. After all of this work on driving down the die size, we entered an interesting situation. Even though we were able to shrink the core dimensions, we were now limited by the outer ring of pads that connect the die to its package. This was not an issue earlier, since we had fewer pads and a larger core, but as the project progressed we added pads and shrank the core until we reached this predicament. However, the I/O ring team elongated our bit slice in the ring slightly and narrowed the width considerably, allowing us to reduce our die size until we were once again limited by the size of the core. Again, this trade-off on the bit slice dimensions was not readily apparent at the outset of the project, but was obviously a big win when we analyzed the situation.

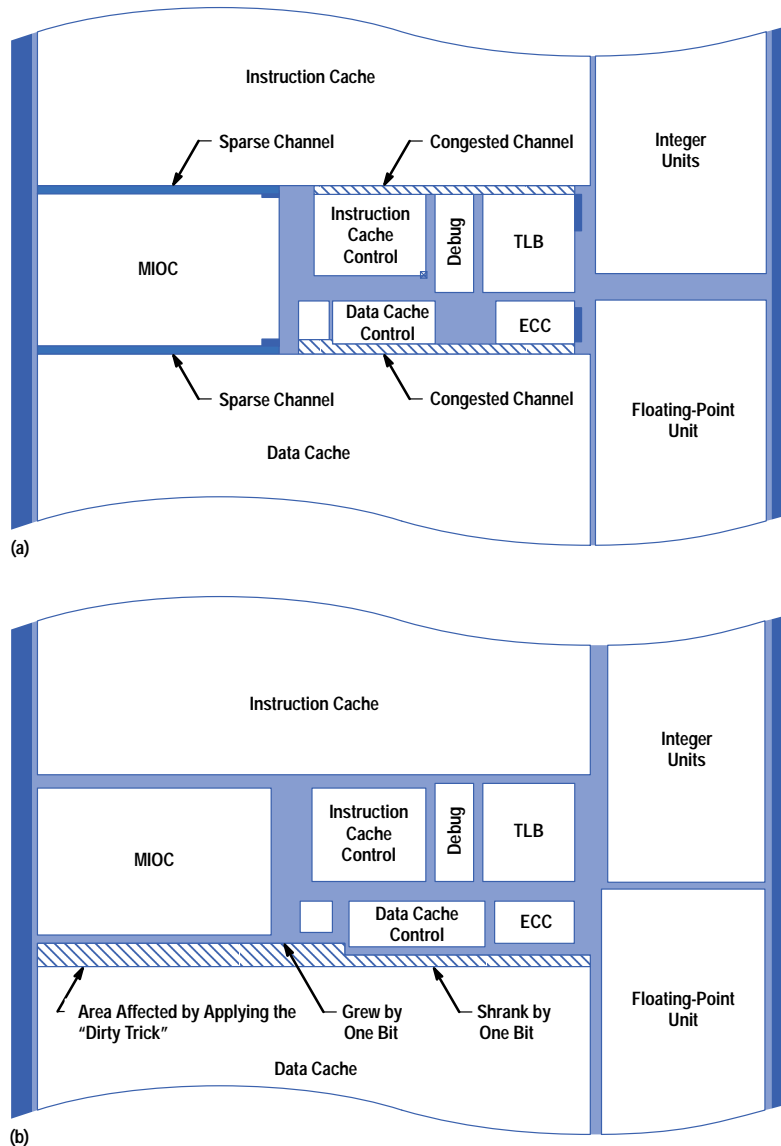


Fig. 4. (a) The floor plan before applying the dirty trick. (b) The floorplan after applying the dirty trick.

Metal-Four Routing. The last obstacle came after we finished automated signal routing. The router we used, HARP,* was designed for the three-metal-layer process used on the PA 7100LC and so it was unable to automate the fourth metal layer. It was a channel-based router, which allowed the block designers to use all three metal layers within their block boundaries, but required us to leave areas free between the top-level blocks for the interconnect. We used HARP to connect signals between top-level blocks, but we left major buses, power connections, clocks, and speed-critical signals for the hand-routed fourth metal layer. This meant, however, that any layer-four metal used within the blocks could interfere with the global metal four, which we planned to run over the blocks on the chip, not merely in the channels. Therefore, from the outset of the project, metal four was under the ownership and control of the composition team.

The cache array designers were given full control of layer-four metal in their areas, but all other block designers proceeded as if they only had three metal layers. As the global metal-four floor plan matured, metal four was released to the block owners to reduce area in places that did not conflict with the global route. In all other cases, the block owners were constrained to use the lower three metal layers instead of placing obstructions to the global metal-four route, even if it meant growing their blocks.

This stingy allocation of metal four became very important as new buses and timing-critical signals were promoted up to the metal-four “superhighway.” Near the end of the project, the connection of the metal-four power buses to the top-level blocks became more and more challenging, and would have been impossible if not for the freedom retained by keeping metal four clear of obstacles.

* HARP (Hewlett-Packard Automatic Routing Program) is an internal routing tool that was leveraged from the PA 7100LC toolset.

Leaving the flexibility to make last minute changes was critical to meeting our die size commitment. Since, at that point in the project, our packages had been ordered with a specified die cavity, changing would have had serious financial and schedule implications.

Practice Runs

The PA 7300LC team used several techniques to ensure that the project would proceed as smoothly as possible. These techniques included building an SRAM test chip and doing a mock tape release.

Using a Prototype Chip. Before the PA 7300LC, HP had never produced a CPU with any significant amount of on-chip memory. How could we ensure that the cache would work in first silicon? Without a working cache, running test code in an actual system would not be practical. To help ensure a working cache in first silicon, we designed and built an experimental memory chip, featuring various RAM cell topologies. This test chip provided a large amount of visibility into the workings of the RAM cells. It also proved to be an excellent tool for analyzing the workings of the on-chip cache. Because the RAM design was effectively “phase two verified” during phase one of the CPU design cycle, the PA 7300LC on-chip cache worked in first silicon, greatly easing the time and resources required for phase two debugging of the rest of the CPU.

Mock Tape Release. Tape release of a CPU is quite a complicated process, involving several steps of database copying, verification, translation, and data transmission. Also, it does not lend itself to leveraging. Any one of these steps could cause a delay of several days in fabricating the chip. Therefore, we performed a mock tape release in which each step was executed as if it were part of an actual tape release. The only exception was that the data used was not the final, fully designed CPU. When the time came to do the actual tape release, the process went very quickly and smoothly.

SRAM Redundancy Improves Yield

With about eight million of the nine million transistors on the PA 7300LC, the cache is the most likely block on the chip to fall victim to a fabrication defect. Therefore, we added redundant blocks of four columns each in the SRAM, so that a block that contains a fabrication defect can be replaced with a functional block via a set of multiplexers. Fig. 5 illustrates the shifted-block method used to replace the defective block with a redundant block.

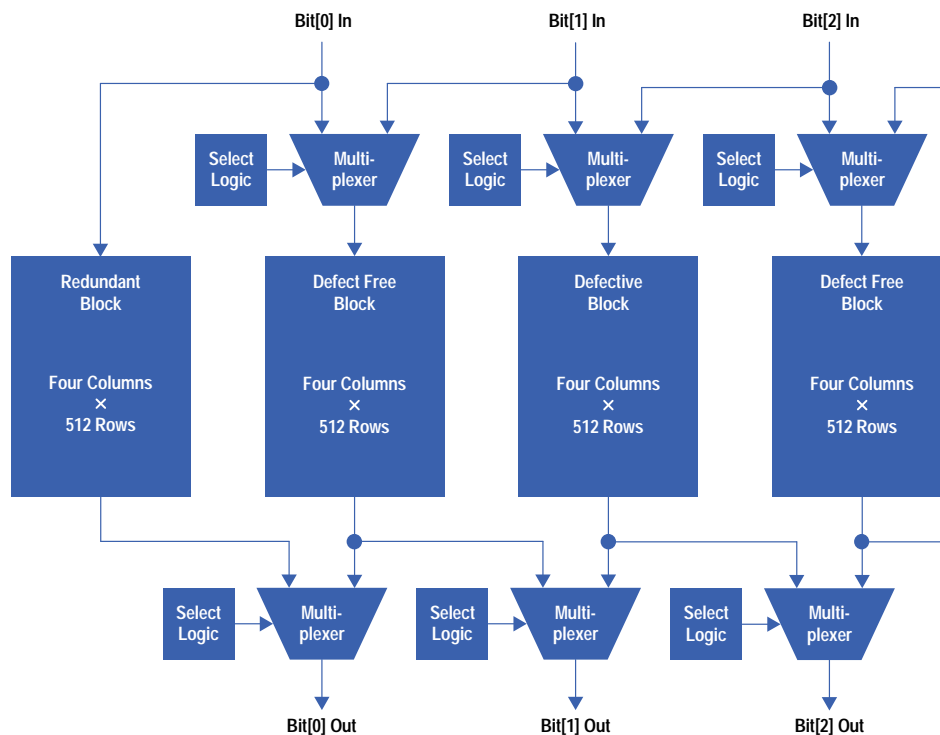


Fig. 5. The shifted-block column replacement method.

The select logic on the multiplexers shown in Fig. 5 is controlled by a fuse that can be blown with a laser to deselect the failing block of columns and select an adjacent block. The adjacent block’s multiplexer must also be programmed to select the next block. This ripple continues until one of the redundant blocks has been substituted into the RAM array. By substituting an adjacent block rather than immediately substituting the redundant block for the defective block, timing changes are minimized.

Adding a Serial Number

One of the new features incorporated on the PA 7300LC is a serial number individually programmed onto each die by the same laser that programs the redundancy selection multiplexers for the on-chip cache SRAM. As wafers are put on the laser for cache redundancy programming, we are able to blow the serial number fuses at the same time. The serial number feature was added to the production flow with very little overhead. Fig. 6 shows a set of serial number fuses. The serial number was added to the PA 7300LC because:

- It provides the ability to track any given die back to its original lot, wafer, or die designation, so we can analyze information gathered on the die at wafer test and at initial package test. On previous microprocessors, we were unable to track backwards in this fashion.
- It allows the design team to select specific dice off a wafer without having to remove the whole wafer from the production flow. This makes it much easier to grab interesting parts for further characterization.
- It provides a convenient way to refer to and classify production parts. The serial numbers became an invaluable part of the phase two debug effort, because we were able to know the history of the part we were debugging.

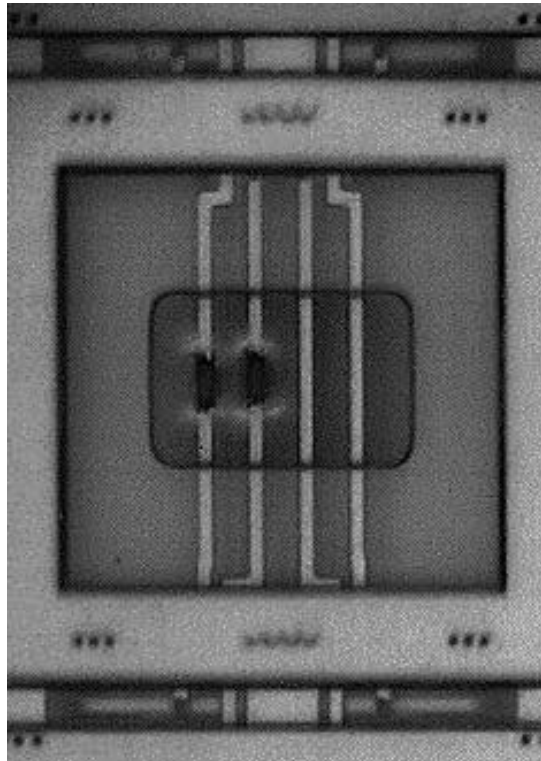


Fig. 6. Two of the serial number fuses in the photomicrograph above have been blown by the laser. The other two are left intact.

High-Speed, High-Coverage Testing Reduced Manufacturing Cost

Both broadside parallel and serial scan tests were used to test the PA 7100LC. Many of these tests were leveraged for use with the PA 7300LC. Some tests were simply copied from the PA 7100LC test suite and reformatted for use with the PA 7300LC. These tests included legal PA RISC assembly code for parallel vectors and serial scan tests of highly leveraged blocks, such as the integer data path.

Other tests required small changes. For instance, TLB tests on the PA 7100LC involved writing and then reading a variety of values for each TLB entry. Then the test simply looped through this process for each of the 64 entries in the TLB. Thus, to test the PA 7300LC's 96-entry TLB, we merely changed the loop value from 64 to 96 entries and reformatted the test.

Automated Test Generation. While many of the highly leveraged custom data path blocks could use scan tests leveraged from the PA 7100LC, this was not the case for the logic-synthesized standard cell blocks because any logic change rendered the old tests useless. Fortunately, the use of an automated test generation tool allowed the PA 7300LC team to have a significant portion of the serial tests written before we received first silicon. Shortly thereafter, we completed the rest of the serial tests, with high fault coverage. The control block test efforts were also helped by widespread use of state memory latches which were controllable and observable via serial scan testing.

Manual Test Generation. For custom data path blocks that were not leveraged, such as those in the MIOC and cache controller blocks, block designers wrote tests by hand, ensuring that each transistor in their design would be tested. Often, this daunting task was aided through the use of Perl* scripts to help generate the test vectors. Thus, many circuit designers found themselves becoming part-time software developers until their block tests were written.

Verifying Block Tests. As block designers began generating serial tests, the ability to verify these tests became an issue. Simulating a single block test on a model of the chip would take anywhere from a few minutes to several hours. However, a real chip could run even the largest test in just a few seconds. Therefore, a way to verify the block tests on an real chip could save a lot of simulation time without compromising test quality.

However, the testers used to test these chips in manufacturing were not readily available. Furthermore, they were too expensive to use for this purpose. Since we were running serial block tests, we only needed to control the chip's serial test pins. The other chip I/O pins could be tied to ground.

Fortunately, we had decided to make our serial test port comply with the JTAG (IEEE 1149.1) industry standard. This meant that a relatively inexpensive test port interface was readily available. We purchased a JTAG Industries PM 3720 and built what we called a "bench tester" around one of these interfaces. Fig. 7 shows a block diagram of the bench tester.

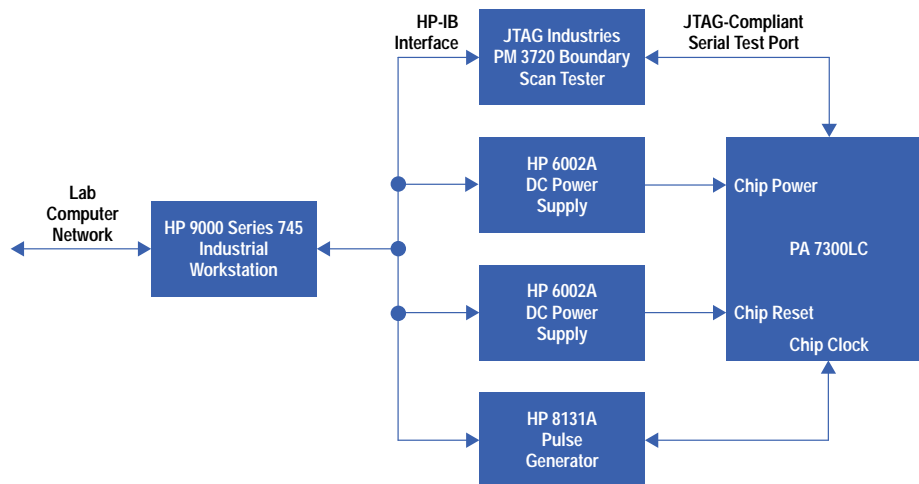


Fig. 7. The JTAG serial port bench tester.

We fed the chip power and controlled the reset pins with a couple of old HP 6002A dc power supplies. The system clock was provided by an HP 8131A pulse generator. Finally, all of these components were controlled via the HP-IB and connected to the lab's computer network through an HP 9000 Model 745 industrial workstation. The network connection allowed designers to run tests and monitor results from their desks, or even from home.

Many block designers pushed the bench testers well beyond their original intended use. By the end of the project, they could be used to create voltage-versus-frequency shmoo plots as the tests were executed over a range of power supply and clock frequency values. We even engineered a way to execute a loop of code in the instruction cache with no other system support logic, proving that the PA 7300LC is truly a system on a chip.

* Perl (Practical Extraction Report Language) is designed to handle a variety of UNIX® system administrative functions.

Summary

In conclusion, the PA 7300LC design team owes much of its success to previous project teams. Our aggressive time-to-market goals were met not only because of circuit leverage, but also because of methodologies from previous projects. Also, an early focus on quality prevented a lot of rework at the end of the project. Excellent performance from this highly integrated processor gives HP a competitive advantage in the cost-sensitive, performance-hungry market for which it was designed.

UNIX is a registered trademark in the United States and other countries, licensed exclusively through X/Open Company Limited.

X/Open is a registered trademark and the X device is a trademark of X/Open Company Limited in the UK and other countries.
