Running RISCs

John K. Foderaro, Korbin S. Van Dyke, and David A. Patterson
Computer Science Division, University of California, Berkeley

Last year, we described the design of RISC I (Fitzpatrick et al. *1981), a 44,500-transistor 32-bit microprocessor. At that time, we had not yet received our first chips. Now we can tell a complete story, including a moral and a happy ending, in which promptness is punished, Murphy's Law is proved, and perseverance and patience are rewarded.*

RISC stands for *Reduced Instruction Set Computer*, a new class of simpler computers promising higher performance using simpler hardware. Examples of RISCs are the IBM 801 (Radin 1982), the Berkeley RISC I (Patterson and Séquin 1981) and the Stanford MIPS (Hennessy *et al.* 1982). RISC I was the first chip built as part of a new graduate curriculum of the Electrical Engineering and Computer Sciences Department at the University of California, in which students propose and evaluate architectural concepts, learn Mead-Conway design methods, form teams to build the system, and then test their design.

RISC CAD

In our environment, design tools dictate design style. Because we had only five students, and had to complete a 32-bit computer in 23 weeks, we had to rely on programs to increase productivity. We used existing programs whenever possible, building our own only when other solutions were not available. We designed RISC I at two levels: a low-level mask description and a high-level functional description. The masks were entered via *Caesar*, a color graphics layout editor developed by John Ousterhout (Ousterhout 1981). We used Clark Baker's *DRC* program to check for layout errors. We counted on visual inspection to discover the few layout errors that *DRC* overlooked (implant-to-gate spacing, gate overhang, and implant overlay around gates).

The functional description was written in *Slang*, a LISP-based simulation language created by John Foderaro during the development of RISC I (Van Dyke 1982). The most difficult parts of our logical design were the timing and the miscellaneous gates to drive the control lines. Because we expected to have more errors in this area than in the data path, a program that would simulate a description of the control circuitry to discover timing and control errors was more important. The RISC I *Slang* description, including all control lines, the PLA's, and miscellaneous control logic, explicitly corresponds to about 2000 transistors in RISC I; *Slang* simulates the rest of the chip—42,500 transistors corresponding to the registers, ALU, and shifter—at a high level. We debugged the description by running about a dozen small RISC I programs, called diagnostics, on the *Slang* description of RISC I. Limited time kept us from using formal fault-coverage models to decide whether these diagnostics adequately exercised the chip.

Two more programs linked these two descriptions. *Mextra*, a circuit extraction program created by Dan Fitzpatrick, takes mask descriptions and derives a transistor-level description. *Esim*, a switch-level simulator created by Chris Terman, simulates the derived description. *Slang* "swallowed" *Esim* to monitor the values of several dozen interesting nodes in both the functional- and switch-level simulations. This multi-level simulation found dozens of disagreements between the two levels of simulation—errors that would have kept RISC I from working.

RISC Fabrication

On June 22, 1981, the RISC I design, using single poly and single metal layers with no buried contacts, was complete: it passed all software checks. RISC I then became part of the experiment to see whether commercial silicon foundries would provide fabrication for small-volume custom designs.

When inexperience is combined with ambition, you get a very large chip. The standard fabrication services were giving fast turnaround to small chips using 5-micron minimum features. Even using 4-micron features, RISC I measures 10.3 x 7.75 mm (406 x 305 mil). Fortunately, the MOSIS Implementation Service at USC/ISI agreed to use RISC I to explore the problems of fabricating large chips at 4 microns. Alan Bell and Lynn Conway of Xerox PARC also offered to fabricate our design, and we replied with a copy of the CIF file on July 17, 1981.

Although Xerox and MOSIS selected different maskmakers, both selected the same vendor for fabrication. The Xerox masks arrived first, but promptness was not rewarded. A new vendor employee ran the wafers through the line using the wrong process. The MOSIS masks arrived later and were sent through the right process steps, but the poly lines were too wide because of problems with one of the polysilicon processing steps. As predicted by Murphy's Law, two more events at the vendor's site kept us from receiving the chips until November: a management reorganization and a fire in the ventilation system.

In November, Michael Arnold finished *Lyra*, a new layout-rule checker (Arnold and Ousterhout 1982). *Lyra* discovered four layout errors—gate overhang and implant-to-gate spacing—overlooked by *DRC*. These probably would not have kept the chip from working, but they might have reduced the yield. The folly of visual inspection was illustrated while verifying the errors. This early version of *Lyra* gave the location and layer of the error; nevertheless, three people, using *Caesar* to explore the design, needed more than two hours to find the errors. The

current version of *Lyra* pinpoints the error and gives complete error messages.

MOSIS offered a new run in December 1981; therefore, we submitted the corrected CIF to be fabricated by a second vendor. This was an experimental run on a research line to try 4-micron Mead-Conway designs. Thus, the need for a second run in April 1982 did not surprise us. Wafers with good processing arrived at Berkeley in May 1982. MOSIS started another run with the original vendor in late January, and we also received their wafers in May.

In the meantime, because of the processing problems, the original vendor refabricated chips from the original masks (M17A) for Xerox. We received chips from this run in January. Table 1 shows the chronology of RISC I fabrication.

Date	Event	Name
April 1980	Experimental Architecture Class	CS292R
September 1980	Mead/Conway Class	CS248
January 1981	VLSI Systems Class	CS292X
April 1981	VLSI 'Testing' Class	CS292Y
June 22, 1981	CIF sent to MOSIS	M17M
July 17, 1981	CIF sent to Xerox PARC	M17A
October 22, 1981	Chips from Xerox PARC	M17A
October 25, 1981	Chips from MOSIS	M17M
November 1981	*Lyra* finds layout errors	
December 4, 1981	Submit new CIF to MOSIS	M1DT
January 7, 1982	Chips from Xerox PARC via refab	M17A
January 29, 1982	MOSIS sends new CIF	M21Z
February 9, 1982	MOSIS M1DT failed fab	M1DT
April 1982	MOSIS refab	M1DT
May 1982	Wafers from MOSIS	M21Z,M1DT
June 11, 1982	RISC I runs first program on board	M17A/CS251
June 15, 1982	Design error discovered	M1DT
July 2, 1982	Find 4 RISC I die without yield errors	M21Z

TABLE 1. History of RISC I.

RISC Testing

Although the fabrication delays were longer than we had expected, building the hardware and software to test the chips kept us busy. The tester receives bit patterns for a test from a host computer, applies these bit patterns to the inputs of the chip under test, and sends the resulting output bit patterns back to the host computer for analysis.

The tester was simply a buffer that could drive and record any pattern of 64 bits every 250 ns. Figure 1 shows a photograph of the tester hardware. This tester provided variable-speed clocks and power supplies, and could repeat a test indefinitely (although it could record only the last 1024 entries). We needed programs on the VAX to prepare the test patterns and massage the test results.

Once again *Slang* came to the rescue, as shown symbolically in Figure 2. *Slang* uses the test programs and computes the correct patterns for the tester and the correct results for those patterns. *Slang* then checks the results from the real system.

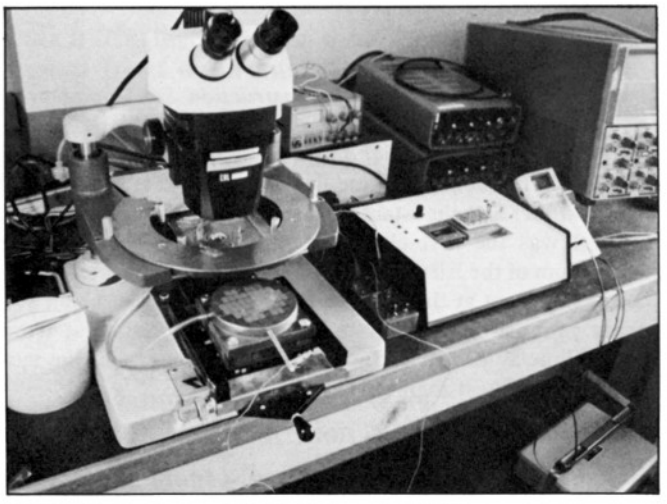

FIGURE 1. Rucker and Kolls 250 probe station with tester (white box on the right).

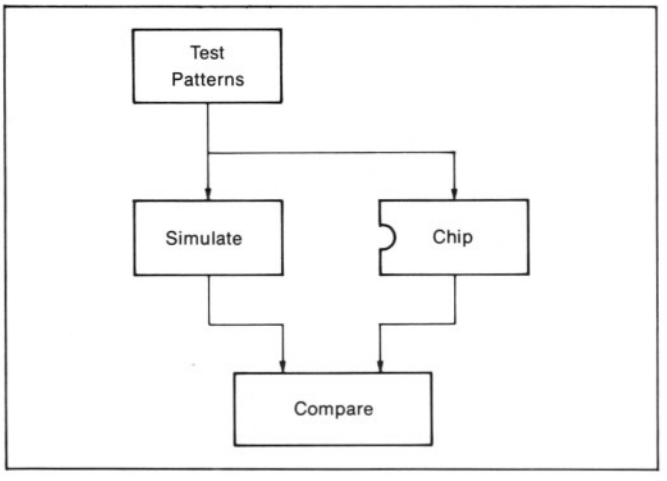

FIGURE 2. Use of Slang in testing.

RISC Results

Out test plan was founded on pessimism. We planned to drive Scan-In-Scan-Out (SISO) hardware to test each block. There are 5 SISO loops in RISC I, one each for the shifter, ALU input, ALU output, the program counters, and control. We tested the first chips from MOSIS (M17M), but no chip emerged with all SISO loops working. We depended on a functioning control loop to test some other modules in the chip, but this loop rarely functioned. The SISO loops were routed after the main area was laid out, resulting in very long poly lines which were potentially more susceptible to yield errors.

Following the same plan, we tested the second batch from Xerox during the winter. Testing proceeded slowly, owing to both the debugging of the RISC I testing software and hardware, and to the other educational requirements of Foderaro and Van Dyke (the only RISC I designers still at Berkeley). We never found working SISO loops on this batch of chips, but two chips displayed "signs of life" when programs were fed to them. Further testing showed that in spite of yield flaws or design errors, these chips performed most of the intended functions. A few bits of the data path were stuck at 0 or 1; nevertheless these chips could still execute some instructions.

Van Dyke designed and built a board for RISC I including the miscellaneous "glue" around the CPU, the memory, the I/O, and the memory management elements. A Xerox refab chip, even with some of the upper bits stuck high, successfully ran the first RISC I program on June 11, 1982. Figure 3 shows a photograph of that board. The first RISC I program read characters from the terminal, changed the text, and wrote the characters back out.

Because we considered the logical design to be largely correct, we ignored the SISO loops and began running diagnostic programs on the chips. Before this, we were testing diced and

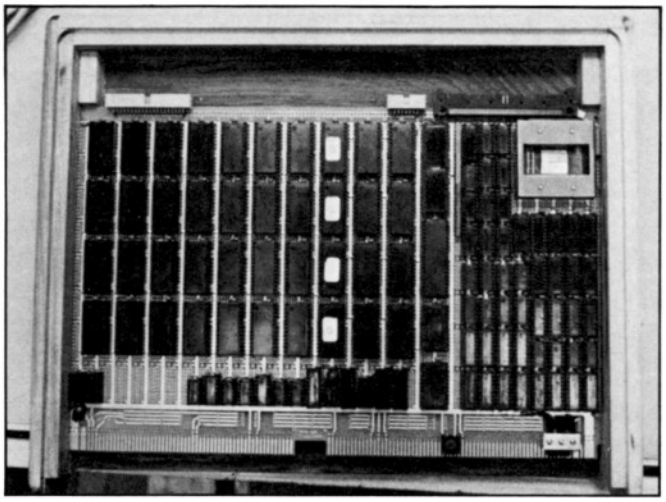

FIGURE 3. Photo of RISC I board.

Machine	Speed MHz	wait states	Language	Time (milliseconds) search	sieve	puzzle	acker
8086	5	0	Pascal	7.3	764	44000	11100
iAPX-432	8	4	Ada	4.4	978	45700	47800
MC68000	8	2	C	4.7	740	37100	7800
Average				5.5	827	42300	22200
RISC I	1.5	0	C	2.5	698	23500	16000

TABLE 2. Execution time of four microprocessors on four programs.

bonded chips. As the students' "lifetimes" were running short, we skipped the dicing and bonding stages and tested whole wafers from the second round of MOSIS fabrication. Figure 1 also shows the probe station we connected to our VLSI tester. These new chips ran the diagnostic programs used to verify our original design. We (foolishly) created new diagnostics and uncovered a design error associated with the optional setting of condition codes on the load and shift instructions. In defiance of historical precedent for solving a design problem by turning it around and calling the problem a new architectural "feature," we decided to fix this error by modifying the RISC I assembler. (This was possible because ALU operations properly set all condition codes, whereas load and shift instructions do not set the negative condition bit. The patch consists of inserting an arithmetic test instruction when a conditional jump needs the N condition from a load or shift operation.) At this writing, we have tested about 40 chips from the second MOSIS wafers. We have found four fully functional chips, (although the SISO circuitry has not been verified yet), giving a 10% yield. This is a better yield than we had expected from such a large chip.

The fastest of these chips runs all diagnostics at 1.5 MHz at room temperature, using the probe card with a floating substrate bias, or 2 μsec per RISC I instruction. This rate was calculated by running the tester at 4-μsec per instruction, with the gap between the three non-overlapped clock phases being as large as the clock. We had based our original performance projections for RISC I on a .4-μsec register-register instruction, derived from the .4-μsec register-register operation of the 10-MHz Motorola MC68000 and the .3-μsec register-register operation of the 10-MHz National NS16032.

Several factors caused this difference. To understand the speed of this fabrication process, we measured the speed of the ring oscillator on the test strip (which is routinely inserted by MOSIS). It ran at 11 MHz. Previous chips processed with the same design-rule features have run the same oscillator at 20 MHz. The later stages of design involved connecting cells, and we concentrated on logical correctness rather than circuit speed. Although we followed the rule of avoiding long unbuffered lines, we had no tool to check for such mistakes. We recently re-examined the design, and found four long clocked control lines that *SPICE* predicts will limit the maximum clock speed to 4 MHz. Furthermore, many of our diagnostics can be run with a 3-MHz clock, suggesting that only a few RISC I instructions (CALL and LOAD) are limiting the performance. Finally, because we still have 200 more chips to test, we may well find faster RISC I's.

Table 2 compares commercial microprocessors to a 1.5-MHz RISC I (including the assembler changes to fix the error). Hansen *et al.* (1982) ran programs on a 5-MHz 8086 with no wait states on an Intellec MDS III development system, an 8-MHz MC68000 with two wait states on a Dual Systems Corporation Dual 8312, and a simulator of the newest version (release 3) of the Intel 432/800, (an 8-MHz, 4-wait state system). As the table shows, a 1.5-MHz RISC I runs these programs a bit faster than do current commercial microprocessors.

A Retrospective Look

Hindsight lets us see our mistakes and offer warnings for future designers. First, don't rely on visual inspection to catch any layout errors. (A second layout-rule checker would have found our mistakes in the first masks.) Our second mistake was incomplete diagnostics. A few more diagnostics would likely have found our only design error. SISO proved difficult to use; if we had written SISO diagnostics, we would have noticed the difficulty and changed the chip.

Although we were concerned with performance, the lack of a simulator between analog-level *SPICE* and switch-level *Esim* precluded performance-tuning of the complete design. Existing higher-level timing analyzers, such as *MOTIS-C* and *LOGIS*, were not integrated into the UNIX environment; perhaps more importantly, they required a new description of the design which could not easily be extracted from the other descriptions. Manufacturers apparently have budgets for hours of *SPICE* runs on CRAY-1's—a luxury not likely to be found soon in academe. We believe higher-level timing analyzers have a promising future.

We hope this article has made it clear that the work required to build hardware and software to test a chip of this size approached the amount of effort required to design it. If we had started over, we would have used more resources on this tedious but important chore. Many people were working on methods to improve chip design, but very few are working on testing. This research area is ripe for new ideas.

The short student "lifetime" requires fast-turnaround silicon. If we have chips with good processing within 3 months after design, we can rely on students to test and perhaps even to improve the chips. If it takes a year, students can rarely enjoy that important advantage. Therefore, one must balance ambi-

tion with die size. MOSIS can provide a 6-week turnaround for standard-size chips that use the standard process (Cohen and Tyree 1982). Furthermore, even special runs have problems with designs larger than 8 mm on a side, because the makers of vendor software (to compensate the masks for the process) never dreamed of a chip that big.

Fortunately, hindsight has also shown successes. For first silicon, the chip had acceptable yield. (One terrible possibility was that the chip might be too large ever to result in a working die.) The working chips testify to the quality of the design tools. (For more information about these tools, see *University Scene* in this issue.) Perhaps the most unusual aspect of this approach to design was that we kept all knowledge of the chip on-line in "program-understandable" form. Furthermore, the on-line description superseded any written documentation. Because the description was finished before the layout was done, we could write a few programs to ask *Slang* what was connected to a given line, or what nodes should be connected. Designers usually keep a set of official logic diagrams for the chip in a project notebook, and derive all other descriptions of the chip from those logic diagrams. We believe that if we had done this, the chip would have taken longer to build and contained more errors, because you can't compare a sheet of paper with an extracted circuit using a program. Unlike the procedure used by most systems, control circuitry was drawn "on-line" using specifications from our *Slang* simulator description.

Perhaps the most unusual aspect of this approach to design was that we kept all knowledge of the chip on-line in "program-understandable" form. Furthermore, the on-line description superseded any written documentation.

The bottom line of the RISC I effort is that as part of the graduate curriculum, students designed and evaluated an architecture, learned VLSI design methods, built new CAD tools, and tested their design. The end product, a 44,000-transistor integrated circuit, had one minor design error, worked on the first good silicon, and ran diagnostic programs faster than commercial microprocessors.

Acknowledgements

Carlo Séquin supervised the master's-degree projects by Wing-Cho Feng and Bob Cmelik, who built the initial VLSI tester hardware and software. The tester built by Jim Beck, which we used to test RISC I, is a revision of this design. John Foderaro and Korbin Van Dyke spent several months testing RISC I, and John wrote many programs to automate testing. Jim Peek did the recent *SPICE* studies of RISC I performance.

We thank Danny Cohen, Lee Richardson, Vance Tyree, and the rest of the MOSIS crew at USC/ISI; and Alan Bell, Alan Paeth, Gaetano Borriello, and Lynn Conway of Xerox PARC for their cooperation and hard work exploring this new field. Their commitment made RISC I possible.

We also thank Alan Bell, Barbara Borske, Danny Cohen, Lynn Conway, Dan Fitzpatrick, Paul Losleben, John Ousterhout, Jim Peek, Lee Richardson, Carlo Séquin, and Jerry Werner for their suggestions on this article.

This research was sponsored in part by Defense Advance Research Projects Agency (DoD), ARPA Order No. 3803, and monitored by the Naval Electronic System Command under Contract No. N00039-81-K-0251. Our thanks also go to Duane Adams, Paul Losleben, Robert Kahn, and DARPA for their foresight in providing resources that let universities attempt projects involving high risk.

References

Arnold, M., and J. Ousterhout. 1982. "Lyra: A New Approach to Geometric Layout Rule Checking," *Proceedings of the 19th Design Automation Conference*, Las Vegas, NV.

Cohen, D. and V. Tyree. July/August 1982. "Quality Control from the Silicon Brokers Perspective," *VLSI DESIGN*.

Fitzpatrick, D.T., J.K. Foderaro, M.G.H. Katevenis, H.A. Landman, D.A. Patterson, J.B. Peek, Z. Peshkess, C.H. Séquin, R.W. Sherburne, and K.S. Van Dyke. Fourth Quarter 1981. "A RISCy Approach to VLSI," *VLSI DESIGN*.

Hansen, P.M., M.A. Linton, R.N. Mayo, M. Murphy, and D.A. Patterson. June 1982. "A Performance Evaluation of the Intel iAPX 432," *Computer Architecture News*.

Hennessy, J., N. Jouppi, F. Baskett, A. Strong, T. Gross, C. Rowen, and J. Gill. February 1982. "The MIPS Machine," *Proceedings of COMPCON Spring*, San Francisco, CA.

Ousterhout, J. Fourth Quarter 1981. "Caesar: An Interactive Editor for VLSI Circuits," *VLSI DESIGN*.

Patterson, D.A., and C.H. Séquin. May 1981. "RISC I: A Reduced Instruction Set VLSI Computer," *Proceedings of the Eighth International Symposium on Computer Architecture*.

Radin, G. March 1982. "The 801 Minicomputer," *Proceedings of the Symposium on Architectural Support for Programming Languages and Operating Systems*.

Van Dyke, K.S. June 1982. *SLANG: A Logic Simulation Language*, M.S. Report, U.C. Berkeley, Berkeley, CA.

Afterword

We thought the readers of *VLSI DESIGN* might like to know what happened to the Berkeley authors who appeared in the Fourth Quarter 1981 issue. Most graduates joined small start-up companies and, in one way or another, are capitalizing on their IC design experience. First, the students:

Dan Fitzpatrick is finishing his Ph.D. thesis, and working on CAD tools at CADLINC in Palo Alto.

John Foderaro, the only RISC I designer still at Berkeley, plans to finish his Ph.D. in symbolic computation next year. He won the 1982 Dimitri Angelakos award as the person who gave the most help to his fellow students.

Manolis Katevenis is working on RISC II, and doing his Ph.D. dissertation on VLSI computer architecture.

Howard Landman finished his M.S. thesis, and is now building CAD tools at Metheus in Portland, Oregon.

Jim Peek is finishing his M.S. thesis, and is now building a VLSI graphics chip at CADLINC.

Zvi Peshkess finished his M.S. thesis, and is designing analog chips for a communications system at Silicon Systems in Tustin, California.

Bob Sherburne is doing his dissertation on VLSI computer constructs, and also working with Katevenis on RISC II.

Korbin Van Dyke finished his M.S. thesis and is now a VLSI systems engineer at VLSI Technology in San Jose, California. Korbin is probably one of the few people in the world who has investigated architecture, designed a microprocessor, tested the chip, built memory and I/O boards for the chip, written programs for the chip, and seen it all work.

Now the faculty:

Carlo Séquin, chairman of the Computer Science Division, has been teaching seminars on VLSI design across the U.S. He was named a Fellow of the IEEE this year.

John Ousterhout is teaching the VLSI layout class at Berkeley. In addition to supervising new tools such as *Lyra*, he is beginning to work on his next CAD system. His first system, *Caesar*, has been distributed to 70 universities and companies.

Dave Patterson is teaching the Experimental Architecture and VLSI Systems classes that produced RISC I. He is leading the design of an instruction cache for RISC II, and is looking for a new vehicle to investigate cost-effective, VLSI-based software systems. He was awarded the 1982 Distinguished Teaching Award by the Academic Senate of the University of California.