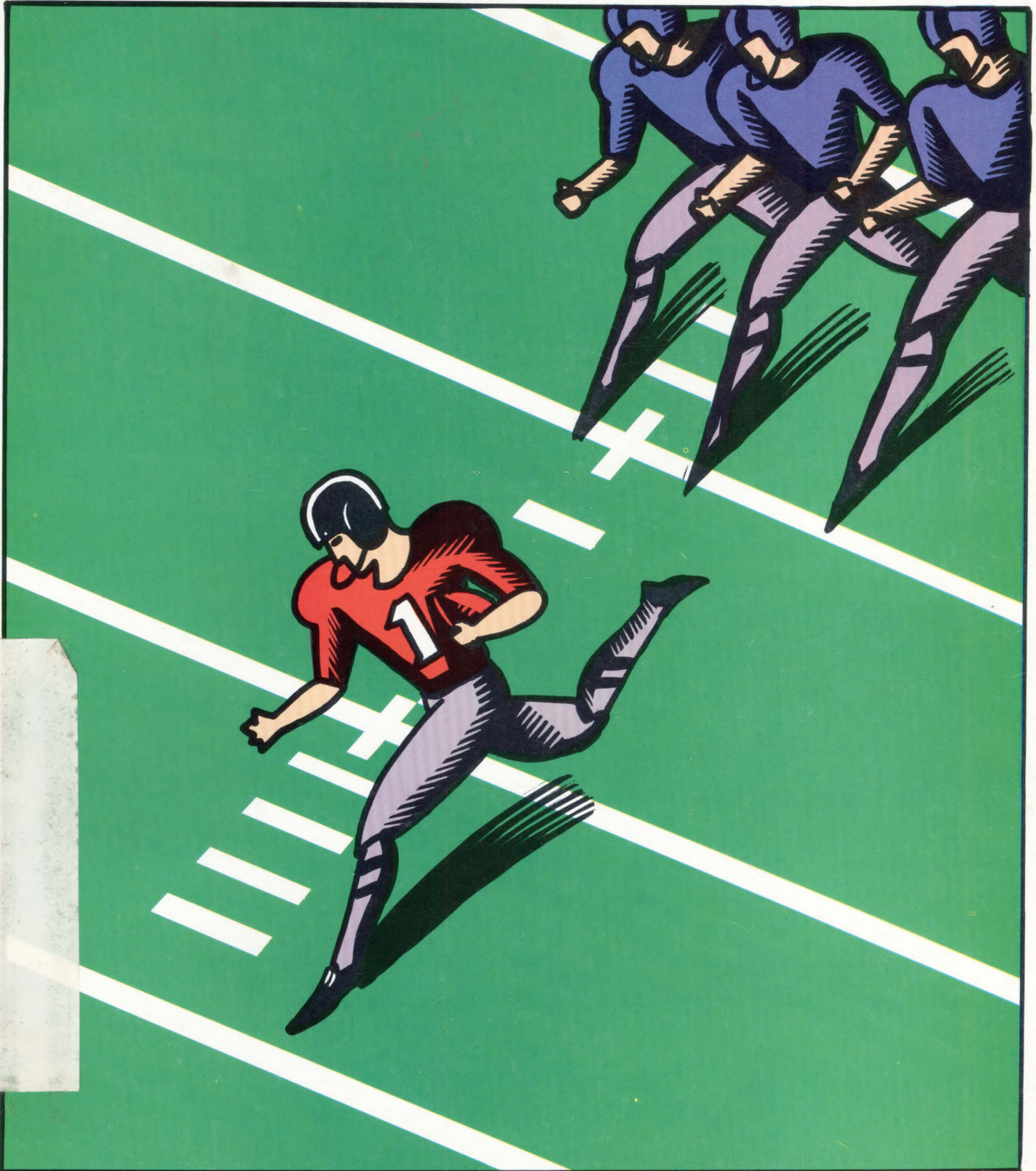# VLSI Systems Design

FOR DESIGNERS OF HIGH-PERFORMANCE SYSTEMS

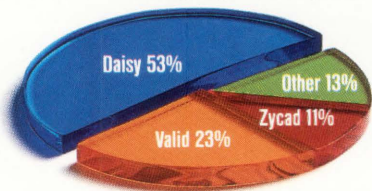ASIC FOUNDRIES: KEY PLAYERS IN CREATING YOUR SYSTEM

DO YOU HAVE WHAT IT TAKES TO MAKE IT BIG IN ASICS?

Not long ago, designing ASICs wasn't even part of your job. Now it's the part everyone's counting on.

That's where Daisy comes in.

Daisy CAE tools are used by more ASIC designers than any other CAE workstations.

Because from schematic creation through post-layout

Simulation accelerator market share. Source: Prime Data, 1985 and 1986 unit shipments.

simulation, Daisy has what it takes to keep ASICs on time and on budget.

For example, our MegaLOGICIAN™ simulation accelerator is seamlessly integrated with the schematic, so you can locate and correct design problems interactively. That means faster debugging and more time to improve the quality of your design. Plus no other accelerator is as well supported, with more than 170 design kits supplied by 70 different vendors. So you can build productivity instead of libraries.

Which may explain why more MegaLOGICIANs are in use today than all other accelerators combined.

Speaking of combining, you

Contents

The capabilities of the ASIC foundry are critical to the success of high-performance systems

64
Retargetable software tools come to the aid of microprogrammers

73

From PLDs to cell-based designs, semicustom ICs are handling more complex applications

10

New microprocessor development tools serve TI's DSP chips

COVER ILLUSTRATION BY ANTHONY RUSSO

20

An enhanced optimizer goes even further in reducing EPLD designs

---

Our ASIC line-up is only half the story.

When it comes to delivery of high-quality, reliable ASICs, S-MOS wrote the book.

We did it in collaboration with our manufacturing affiliate, Seiko Epson. With 18 years of CMOS experience, Seiko Epson is one of the world's most advanced CMOS IC manufacturers.

Through Seiko Epson's high-yield manufacturing technology, we ship millions of ASIC units a month, and with a reject rate of less than .0001%. That's our quality story.

Now we've added a new chapter on design. At our advanced R&D design facility, engineers from S-MOS and Seiko Epson are developing new software to simplify circuit design, simulation and the creation of new megacells for our extensive cell library.

Of course, you can still take advantage of our established design tools because S-MOS supports such workstations as Daisy, Mentor, Calma and PC-based systems using FutureNet, OrCAD and ViewLogic.

Our proprietary LADS simulator will speed up the design process. The S-MOS engineering team will support you from concept to production.

If you are looking for an ASIC program that can make your designs best sellers, call us. (408) 922-0200.

DAC: The Signs of Maturity?

The Design Automation Industry Is Beginning to Pay Attention to Important Details

By now the dust has settled on the 25th annual Design Automation Conference, held last month in Anaheim, Calif. It has become the premier event for the electronic design automation industry, and like the industry, it has changed its character over the two and a half decades that it's been around. Each succeeding conference seemed to be bigger and better than its predecessor. However, although this year's event ran smoother than ever, some of the excitement of past conferences seemed to be missing.

Logic synthesis and hardware description languages like VHDL were the subject of many papers, and related products were prominent on the exhibition floor. But most product unveilings were enhanced versions of existing tools. Joint agreements to port, link, or market were probably the most popular announcements. Perhaps it's a sign of maturity.

There was another sign of maturity in the industry: the increasing consolidation activity—in the form of mergers, acquisitions, or dropouts. Some of these consolidations were marriages holding great promise, but many were for the sake of survival. Some think these moves as a harbinger of bad times for the electronic design automation industry, but I don't. I see it as a growth plateau in which the industry regroups, reviews its strategy, and gets ready to move on to even greater heights.

Is this whistling in the dark? Not really; many of the mergers will benefit the system and chip designers. Vendors of heterogeneous tools that merge will be pressured by market forces to provide good working interfaces between these tools, rather than the token links or good intentions that have resulted from a number of the many highly touted "joint agreements."

In addition, those who closely follow the industry can testify that many vendors are now beginning to pay attention to the "not so little" things that are often neglected in a young and exploding market. These include such entities as databases, documentation, networking, testability, standards, and user-friendly universal interfaces. Attention to these "details" and the increasing system view taken by many of the players is good news, and it could provide the launching platform for reaching the next plateau in the design automation industry.

Roland Wittenberg

ROLAND WITTENBERG
EXECUTIVE EDITOR

Stop Wasting Power

Calendar

INTERNATIONAL WORKSHOP ON VLSI FOR ARTIFICIAL INTELLIGENCE

July 20–22
University of Oxford
Oxford, England

This workshop will provide a forum where AI experts and VLSI system designers can come to discuss trends in AI applications and their computational requirements, VLSI implementations, and computer architectures. Topics to be discussed will include alternative technologies, semantic and neural networks, functional-language architectures, knowledge-oriented machines, rule-based engines, Prolog and Lisp machines, and fifth-generation computers. Further information may be obtained by contacting Dr. Jose G. Delgado-Frias or Dr. Will R. Moore, Department of Engineering Science, University of Oxford, Parks Road, Oxford OX1 3PJ, England, U.K. Phone: (0865) 273188. ∎

SIGGRAPH '88

August 1–5
Atlanta, Ga.

The 15th annual conference on computer graphics and interactive techniques is sponsored by the Association for Computing Machinery's Special Interest Group on Computer Graphics in cooperation with the IEEE Technical Committee on Computer Graphics. It will feature panel sessions, courses, and exhibitions, as well as a film and video show and an art show. Technical presentation topics will include fast polygon algorithms, applications of computer graphics, volume rendering, lighting models, user interfaces, physically based modeling, curves and surfaces, filtering and texturing, hardware systems, and animation. Courses on user interface management systems, computer graphics in science, and solids modeling will be featured. For further information about the conference, contact SIGGRAPH '88 Conference Management, Smith, Bucklin, and Associates Inc., 111 E. Wacker Drive, Suite 600, Chicago, Ill. 60601. (312) 644-6610. ∎

OIS '88

September 7–9
Washington Sheraton
Washington, D.C.

Sponsored by Meckler Corp., the eighth annual Optical Information Systems Conference and Exhibition will focus on write-once and erasable optical storage systems and digital document-image automation. Sessions are planned in such areas as electronic image and document storage systems, erasable optical disk media developments, erasable optical disk drives and systems, evaluating and selecting a WORM subsystem, write-once and erasable optical media manufacturing approaches, operating-system software for write-once optical disk, future trends and new developments, converging optical information technologies, and integrated system development. Additional information may be obtained by contacting Marilyn Reed, OIS '88 Conference Manager, Meckler Corp., 11 Ferry Lane West, Westport, Conn. 06880. (800) 635-5537. ∎

7TH VLSI AND GaAs PACKAGING WORKSHOP

September 12–14
San Jose, Calif.

This workshop is being sponsored by the IEEE's Components, Hybrids, and Manufacturing Technology Society and the National Bureau of Standards. Topics that will be addressed include package thermal design and interconnection options, GaAs IC packaging, die attachment solutions for large chips, VLSI and wafer-scale package design, new failure mechanisms in VLSI packaging, and VLSI package materials advances. Additional information may be obtained by contacting Paul Wesling, IEEE Council Office, 701 Welch Road, Suite 2205, Palo Alto, Calif. 94304. (415) 327-6622. ∎

INTERNATIONAL TEST CONFERENCE 1988

September 12–14
Sheraton Washington Hotel
Washington, D.C.

The International Test Conference provides a major forum for an exchange of information about the testing of electronic devices, assemblies, and systems. This year the conference focuses on the test techniques and equipment needed to meet the challenges of new technologies. Technical papers will be presented on such topics as analog devices, yield modeling and process diagnosis, testability analysis, education and training, application-specific devices, microcontrollers and microprocessors, printed circuit boards, wafer-scale assemblies, hardware and software, process and test data management, surface-mount assemblies, computer-aided engineering, quality and reliability, fault modeling and simulation, design for testability, contactless probing, fixturing, and computer-aided test generation. For additional information, contact Doris Thomas, Executive Secretary, International Test Conference, Millbrook Plaza, Suite 104D, P.O. Box 264, Mount Freedom, N.J. 07970. (201) 895-5260. ∎

BIT Stream

Sony Launches Double-Barreled Threat . . .

In a joint announcement with Motorola Inc., Sony Microsystems (Palo Alto, Calif.) unveiled a new workstation based on two 25-MHz 68030s and a 25-MHz 68882 floating-point coprocessor. Expected to be priced between $35,000 and $45,000, the NEWS 1800 series of platforms will run Sony's version of Unix 4.3 BSD and the X-11 Window System. Sony rates the platform at 5.3 Dhrystone MIPS (normalized to a VAX-11/780). With one 68030 dedicated to I/O processing, the machine is touted at excelling in applications with extensive I/O, networking, and graphics requirements. ■

. . . But Apollo Goes It 'One' Better

Apollo Computer Inc. (Chelmsford, Mass.)—like Sony, Hewlett-Packard, and Masscomp—is eager to cash in on the wealth of applications running on 680x0-based machines. The company's answer is two price/performance-targeted systems, the Series 3500 and 4500, that are binary-compatible with the 3000 and 4000 series. Based on a single 68030, the 3500 uses the 25-MHz version and the 4500 a 33-MHz device.

The 3500 series, which claims a performance of 4 MIPS, starts at a low $7,990, and the 4500 series, which promises 7 MIPS, sports price tags as low as $18,990. The 3500 will be available this month; the 4500 is scheduled for the end of the year. ■
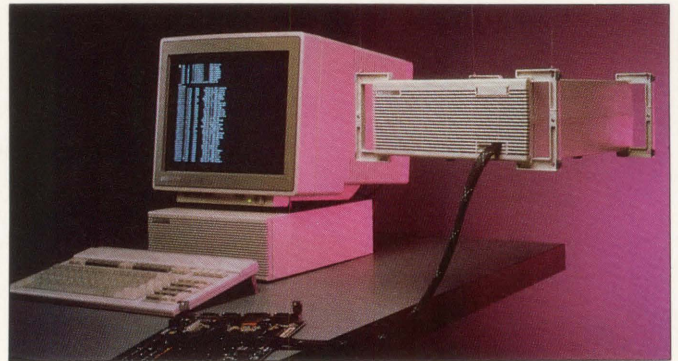
Hewlett-Packard Emulates TI DSPs

The HP 64700 series of emulator/analyzers from Hewlett-Packard Co. combines the company's "logic analyzer on a chip" with its new emulation technology to provide a set of high-performance microprocessor development tools for Texas Instruments' TMS-32020 and TMS320C25 digital signal processors.

The emulators deliver real-time, no-wait-state execution up to 20 MHz for the 32020 and up to 32 MHz for the 320C25. They also support 64K words of dual-ported emulation memory, eight-level sequencing, and code coverage analysis. The optional integrated 16-channel logic analyzer functions as a 100-MHz state analyzer and 25-MHz timing analyzer. ■

Allegro Is Accelerated

A new release of the Allegro PC board design system from Valid Logic Systems Inc. (San Jose, Calif.) provides more than 30 new features that increase the support for high-speed design and analysis; 2D drafting; and manufacturing, testing, and other third-party interface applications.

Release 2.0's new and enhanced interfaces to mechanical design and analysis packages, such those offered by Structural Dynamics Research Corp., permit the designer to check PCB placement for mechanical interference and also to perform stress, vibration, and shock analysis. ■

Photo Courtesy: Structural Dynamics Research Corp.

Zeiss Laser Focuses on Quality Control

Carl Zeiss Inc. (Thornwood, N.Y.) has built a new laser autofocus system into its Axiotron AF and Axiomos wafer inspection microscopes to allow real-time automatic focusing for precision inspection of semiconductors. The system also provides for remote operation with a TV camera, as well as proportional manual speed control. The dual infrared pulsed-laser system can be operated at all magnifications without the necessity of realignment when changing objective lenses.
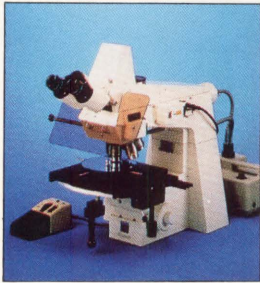
Continuous spot focusing is provided by the system as it tracks the surface of the specimen undergoing inspection. This feature provides a speed advantage over more conventional TV-based systems, in which the microscope's objective lens must arrive at the desired coordinates before the system attempts to focus.

The Axiotron microscope is designed for in-process inspection, quality control, and failure analysis on wafers and masks. The Axiomos system includes a software package for controlling illumination, magnification, and stage position. Its motorized stage can run at speeds of up to 100 mm/s with 0.1-μm reproducibility.

TTL-Compatible GaAs 22V10 Beats 10 ns

Gazelle Microcircuits Inc. (Santa Clara, Calif.) promised to put LSI-level functionality on GaAs standard products. The first product off the drawing board is a GaAs version of the 22V10 PLD with a 10-ns propagation delay (input to output), a 3.5-ns setup time, and a clock-to-output time of 7.5 ns.

Not only is this GaAs part fast, its characteristics are roughly flat over the military temperature range. The figure shows a comparison of combinational input-to-output-delay time vs. temperature for bipolar, CMOS, and the new GaAs device. The new part not only exhibits a temperature stable delay characteristic, but also provides less than half the delay of the other implementation technologies.

By providing TTL-compati-
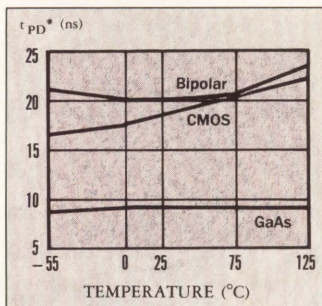
Source: Gazelle Microcircuits Inc.

ble input and output circuits, the part can be easily dropped into any standard 22V10 circuit. The catch? For the near future, only Gazelle can program the $55 parts, which can have a turnaround time as long as one week.
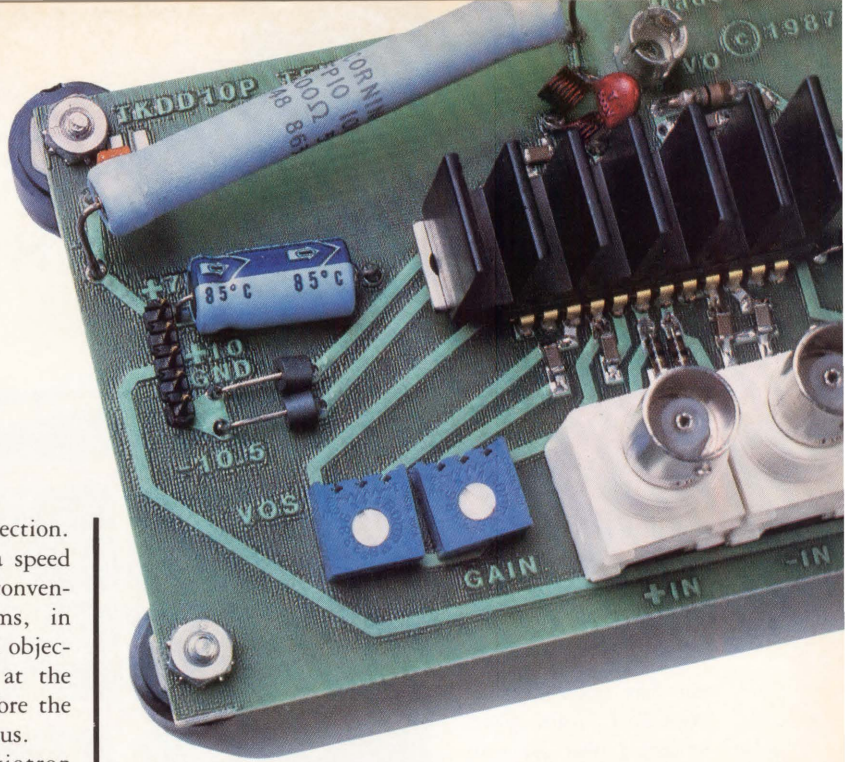
High-Voltage IC Drives Video Displays

Tektronix Inc. (Beaverton, Ore.) has rolled out a monolithic IC, the TKDD10P, that can drive a CRT with 50 V peak to peak at 175 MHz with up to a 6-pF load. Its gain can be varied linearly from 0 to 80 with an external potentiometer.

The display driver chip is based on Tektronix's proprietary Quickchip ASIC technology. Available in a 24-pin power tab package, it runs $35 in 1,000-unit quantities. The chip is also available on an evaluation board, the TKDD10EB. ■

Cadnetix, HHB Systems: A $77 Million Merger

Bruce M. Holland, president and CEO of Cadnetix Corp. (Boulder, Colo.), and Lutz P. Henckels, president of HHB Systems (Mahwah, N.J.), shook hands on an agreement to merge the two design automation companies in a stock deal valued at $77 million. Cadnetix is widely known for its strength in printed circuit board CAD tools, and HHB is strong in logic and fault simulation and automatic test generation. Under terms of the pact, Cadnetix will issue 1.4 shares of new common stock for each share of HHB, which will become a wholly owned subsidiary. Holland will retain his positions at Cadnetix and Henckels will be president of the new HHB subsidiary. ■

HOLLAND

HENCKELS

News Analysis

PLD Breakthroughs Threaten Standard Logic

What has been keeping standard logic parts alive? Invented over 20 years ago, they established one of the most important standards in the industry. But their demise has been contemplated continuously for the past 10 years. However, the 7400 functions, in their many implementations, from LS to AS, FAST, FACT, and ACT, hang in there in the face of competition from ASICs, both LSI and VLSI. Annual worldwide sales for standard logic parts is now about $3.5 billion.

But advances in the speed of PALs that went to market for the first time last month may finally break the 7400 hold. Till now, PLDs could shrink the number of 7400 parts and bring cost savings at chip and board levels, but only at the cost of lower performance.

The result has been that the CMOS segment of the standard logic market is growing very slowly, and lower-speed bipolar versions—L and LS—are flat or shrinking. Only the faster, newer families, like FAST and AS are growing at reasonable rates. These growth rates indicate that the users of standard logic are going to the devices mainly for speed.

Indeed, speed has kept standard logic alive. That's why CMOS has grown slowly, even while LSI and VLSI counterparts in the technology are surging in sales.

Surely, though, there are other attractions for the continued use of 7400 parts besides speed. For one, everyone is familiar with them, they're cheap, they have multiple sources, and they are commodities. (However, the speedy versions are not commodities.)
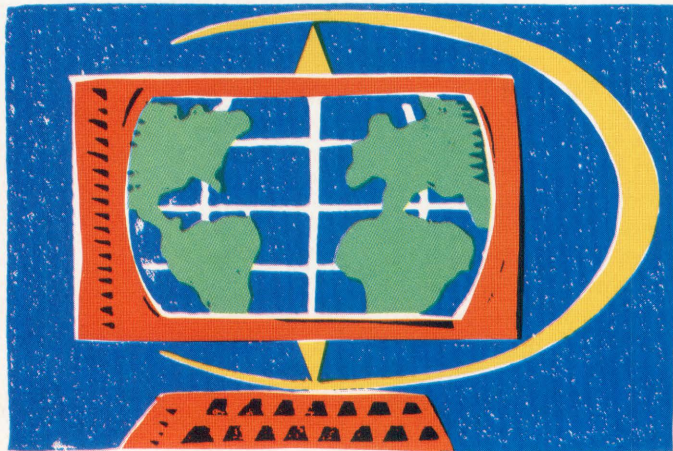
Now that breakthroughs in bipolar PALs are offering devices that are faster than the 7400 parts they can replace, the impact on standard logic will grow rapidly. That's because PALs or other PLDs, being field-programmable, easily replace scores of 7400 functions. If the standard-logic makers wanted to offer a new and complete line of standard logic with better speed, it would take many months and huge investments.

All the room for maneuvering is now held by PLDs. So far we have seen the fastest PLDs at 15 ns, 10 ns, and now Advanced Micro Devices Inc.'s new 7.5-ns series, offered in standard architectures. But soon, and probably first from AMD (Sunnyvale, Calif.), we will see the fastest-propagation-delay processes used to make more application-specific parts that can even compete with the higher-level LSI logic parts. This month, AMD will announce a new PAL with a 10-ns propagation delay that will offer solutions for "common system timing problems." Filling out the line with an array of application-specific models appears to be an added direction for AMD.

Other companies, using the PAL structures, are already charging down the application-specific route, most notably Altera and Intel.

Another reason for PALs to become more important is that the inventiveness and aggressiveness that was exhibited by MMI for the last 10 years is now backed by the considerably larger manufacturing capability of AMD. MMI was always slow in ramping its latest PAL technology and was often production-limited. Staring at several hundred millions of dollars in hard investments to cure that problem is one big reason why MMI chose to merge with AMD.

Why Not Gate Arrays?

But what about gate arrays? Aren't they replacing those 7400 functions? Gate arrays have been around a long time, and standard-logic sales have still continued to climb. Andy Robin, marketing manager for PLDs at AMD, believes FAST and AS are still growing rapidly because gate arrays are having difficulty replacing them, typically becoming economically wise only gate ranges of 10,000 to 20,000 gates and higher, and that's a large commitment for most logic designers. Normally, they have been keeping tight control of their designs, particularly those sections that contain the most critical speed paths for a subsystem, using FAST or AS parts. Also, such designs are often not solidified until late in the system development cycle.

But while gate arrays have been having some trouble penetrating these critical applications, PALs have become popular choices because they can be programmed by the designer, often late in the design cycle, without the turnaround time and NRE costs of gate arrays. In addition, approvals to make this type of design change can usually be avoided. An engineer doesn't have to announce that he had problems with his design; the old PAL simply goes in the garbage can.

A Bounty of Speedy PLDs

Over the last few weeks there has been a bounty of new, fast PLDs offered to the world, from AMD, Lattice Semiconductor, Gazelle Microcircuits, and Signetics.

The new AMD 7.5-ns PAL series consists of four standard 20-pin devices: the PAL16R8, PAL16R6, PAL16L8, and PAL16R4. They are rated at a maximum propagation delay of 7.5 ns from 0° to 75°C and an operating frequency of up to

NEWS ANALYSIS

74 MHz. The frequency rating is important. It should be approximately double that of the latest microprocessors. This allows the devices to interface directly with the microprocessors without absorbing too much of their cycle time.

AMD has also upgraded the speed of the 22V10, the most popular of all PAL devices. The propagation delay has been reduced from 25 to 15 ns. Also, the company is now shipping 10-ns 20-pin PALs in volume.

Many users of the 22V10 have felt the pinch of speed, especially with microprocessors rapidly upgrading their performances. Moving from 25 ns to 15 ns is a

It is a direct drop-in for the bipolar 22V10. But AMD's announcement of its 15-ns version took some of the spring out of Gazelle's leap into the market. One other drawback of the Gazelle chip is that it must be programmed at the factory using laser programming.

Gazelle cannot showcase its potential capabilities very well with the 22V10 equivalent because it is not dense enough. GaAs gives very fast gate transitions internally, but that can be lost with the I/O circuits, which have been purposely slowed down to emulate standard TTL I/Os. In denser circuits, it would take about 50 gates in series to increase the overall delay time appreciably, since the GaAs gates have transition times hovering around 100

PAL16R8-7 FAMILY VS. FAST AND AS

	PARAMETER	FAST	AS	PAL16R8-7
COMBINATORIAL FUNCTIONS				
■ 74138 DECODER	t_{PD}	8.0	10.0	7.5
■ 74151 MULTIPLEXER	t_{PD}	11.0	15.0	7.5
REGISTER, LATCH				
■ 74374 OCTAL REGISTER	t_{CO}	10.0	9.0	6.5
■ 74373 OCTAL LATCH	t_{PD}	8.0	6.0	7.5
	t_{LEO}	13.0	11.5	7.5
COUNTERS				
■ 74161 4-BIT COUNTER	t_S	5.5	8.0	7.0
	t_{CO}	11.0	13.5	6.5
■ 74269/869 8-BIT COUNTER	t_S	2.5	5.0	7.0
	t_{CO}	10.0	11.0	6.5

NOTE: All times are in nanoseconds.   *SOURCE: Advanced Micro Devices Inc.*

major step, making the 22V10 more of a candidate to replace groups of SSI and MSI logic parts. AMD expects soon to boost the operating frequency from 28.5 to 50 MHz. That puts the rate at twice that of the new 20- and 25-MHz microprocessors, making the device very attractive for interfacing at these higher processor speeds.

A week after AMD made its 7.5-ns announcements in June, Signetics Corp. (Sunnyvale, Calif.) announced a line of 20-pin and 24-pin PALs with 10-ns propagation delays. It also unveiled two of its own PLAs, devices that have much more logical flexibility but a 12-ns delay.

Lattice Semiconductor Corp. (Hillsboro, Ore.) also pushed the speed in its CMOS EEPLD end of the market. It now has GAL devices with 12-ns propagation delays that replace common 20- and 24-pin PAL devices. The new parts, the GAL16V8A-12 and GAL20B8A-12, both have a maximum power drain of 115 mA.

In addition, Gazelle Microcircuits Inc. (Santa Clara, Calif.) made its introduction of the 22V10 in gallium arsenide. The device has a typical 6-ns delay (10-ns maximum) but sports TTL I/O parameters.

ps. However, Gazelle can use this part to demonstrate that GaAs is an good alternative for silicon logic at the MMI and LSI levels. That needs to be proven for GaAs to take off as a logic alternative mixed into the critical timing circuits of systems based on silicon.

Gazelle must also demonstrate its claim that GaAs will become as economical as silicon because it will use the same manufacturing equipment and the same design tools. Currently it delivers twice the speed of standard logic but for approximately twice the price. Thus Gazelle has a lot to prove. It will take a few years to convert the skeptical, of whom there are many. Along the way, watch for Gazelle to bring out a string of programmable parts of much higher density, in addition to moving toward application-specific solutions.

All in all, this has been a banner month for PLDs and especially PALs. When future dismantlers of today's systems examine them with an archaeologist's eye, they may also reckon 1988 as the time when the 7400 families first started their relentless downturn.   ■

—Stan Baker

John Birkner: Born to Innovate

The Driving Force

Behind the PAL

Moves On

JOHN Birkner has a fascination for electronics that could inspire zealots and fanatics. Continually drawn to innovations in digital computers, he became the driving force in the invention of programmmable array logic (PAL) devices at Monolithic Memories Inc. in the mid-1970s, and PALs propelled MMI to the front ranks of semiconductor makers.

Before MMI, Birkner was at Computer Automation Inc., where he designed an entire 16-bit processor on a half-card–sized board with only 80 ICs. Though the board was hailed as a great achievement at the time, Birkner knew there had to be a better way. "TTL was a helter-skelter collection of devices from different companies. PLAs were in fat 28-pin DIPs, cost $20, had 60-ns delay times, needed $10,000 programmers, and lacked design software—and there was a big education problem," he recalls. However, the sudden appearance of microprocessors convinced him to move from Newport Beach, Calif., in 1975 to where systems were being put onto silicon.

Birkner's contribution was that he brought a special vision to MMI in connecting applications to IC design. At MMI, he says, "it was like standing on a fence and seeing both sides." IC companies misunderstood the shortcomings of PLAs, but he saw an opportunity. Engineers at computer companies had told him, "If you give us PLAs in 20-pin packages with speeds close to that of TTL, we'll never use TTL again."
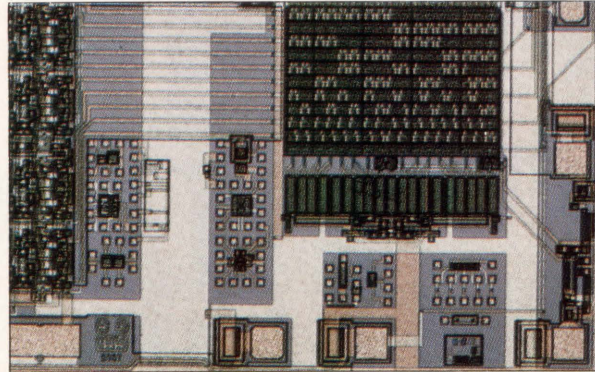
Birkner had his cause, and MMI had the wherewithal in PROM technology; a silicon design genius named H.T. Chua; and an applications manager, Clive Ghest, who supported Birkner. Out of this chemistry was born the PAL business.

The first priority was to make PLAs that were as fast as standard TTL logic. The PLA architecture was simplified to use only programmable OR gates, reducing the chip size and delay and making the new PAL parts easier to use. Then Chua innovated simpler, smaller device structures that enabled PALs to cover the functions of 8-bit octal devices—a key to success. The delay dropped quickly to 35 ns, putting the new PLDs in the TTL league.

To many, Birkner was a blond, bushy-haired hippie from Southern California. His first proposal for development of the PAL line got a ho-hum reception from MMI's management. He gives Ghest, originally from Britain, credit for backing his ideas. "The English have a way of putting up with eccentrics," he says.

His hair has come under somewhat more control, but not his thirst for innovation. He ran his own company, Structured Design

'**P**EOPLE WILL GROW, BUT AS ORGANIZATIONS

GROW THEY WANT TO CONTROL THE ZEALOTS'

Inc., part-time since 1979, while employed at MMI. It made design tools for PLDs but was eventually scaled back to serve only a few customers. It's now run by Birkner's wife, Noel Hendricks.

As a 12-year-old, Birkner was inspired by the *Boy's Book of Electronics*. He first used a digital computer at the University of California at Berkeley in 1964. In two weeks, he used all of the class's computer time, he relates with a tinge of embarrassment but no regret.

Birkner's leisure time is saved for his only child, 30-month-old Samantha. They hike mountains and fly kites on windy hills near the Pacific Ocean. At 45, he has geared down from the long-distance runs of a few years ago.

He is, however, gearing up more innovations. He left MMI in 1986 and recently formed Peer Research Inc. with Chua and some of the old PAL development team. "It will provide the spreadsheet, the Lotus 1-2-3, of ASICs and bring silicon compilation to desktop computers," he says. (He often refers to PALASM, the programming language developed at MMI for PAL design, as "the first silicon compiler.")

Why did Birkner leave MMI, a company that rewarded him and Chua with Porches and Mercedes Benzes? "PALs were doing OK. I had completed a cycle. People will grow, but as organizations grow they want to control the zealots. You can't outgrow the perceptions others have about you. I'm an innovator, not an organization person; I have to have the hours alone to think. To many, I was always the bushy-haired hippie." ■

—STAN BAKER

Floorplanning

IN YOUR April issue, David Hightower, in his Industry Insights "Floorplanning for the Future" (p. 18), states that a "sophisticated, highly interactive floorplanner is required. Even the 'automatic' placement tool in the floorplanner should be interactive; that is, it should have a 'human subroutine' through which it calls the operator from time to time to solve some local problem and then continues searching for the global solution."

Such a floorplanning tool was developed at IBM in 1984. It is described in the paper "CRAFT: A Customizable Refinable Algorithmic Floorplanning Tool" presented at the 1986 ICCD and written by Peter Van Dyke and me. The paper is primarily devoted to the algorithms, but the graphics are briefly described. We also discussed the ability to invoke the algorithms through the graphics interface and switch between interactive and automatic procedures.

The graphics tool displays the placement, histograms of wire congestion in the x and y directions, the maximum congestion in x and y, the total wire length in the x and y directions, and the x and y dimensions of the floorplan. Options are listed on the screen next to the placement.

The designer can invoke the automatic placement algorithm (for a user-specified number of moves), exchange objects at any level of the hierarchy with any other object at that level (including exchanges with an empty space in the hierarchy), rotate a macro (the lowest-level object in the hierarchy), and invoke a global channel program to space the macros apart for wiring.

The user-driven moves are made in the context of the hierarchical structure on which the algorithm operates. As a result, the entire placement is modified in response to efficiently remove empty space. After any action the resulting placement and its corresponding statistics are displayed.

There is also a capability to change the scoring parameters driving the automatic placement. Consequently, the designer can tune the algorithm as he proceeds.

This tool was originally developed to observe the type of manual moves that would improve floorplans in order to create a function that would mimic these types of moves. Thus, in a sense, it is also an expert system.

The tool for the future I would like to see is one that would store information about the types of moves a designer makes and automatically modify the algorithm to produce these moves.

YEHUDAH RELIS
IBM Corp.
Poughkeepsie, N.Y.

Missing Cells

ALTHOUGH I applaud the idea of your recent cell synthesis "rally" ("Cell Synthesis in Action," May, p. 52), I feel there were several problems in the way that it was conducted and the way that you represented our entry.

- When we received the benchmark schematic, we entered it into our tools using the SLICC language, and two bugs were detected immediately. We contacted your editor about the bugs and agreed on a revised circuit, which is the one you published.
- You did not publish our layout. Instead, you published Caeco's layout twice.

- You mentioned that we submitted three layouts but did not explain why. There were three factors left open in the rally. First, since your benchmark did not specify transistor sizes, we ran ours once for minimum size, which gave a 28-ns cycle time; once with sizes optimized for speed by our TILOS sizer, which gave a 5-ns cycle time.

The second factor was the use of second-level metal. Although your rules said that second-level metal was available, several of the published layouts did not use it. Second-level metal reduces resistance and delay but may increase size a little bit in small layouts, since it has larger design rules. In big designs, though, it saves area and improves performance.

The third factor was the question of multiple strips of logic. Again, small layouts tend to be a little more compact in a single row, whereas bigger designs require multiple rows.

- You state that the SCS tool is the only system capable of producing multiple strips. Not only can SC2D do this, but several of the layouts published in your article demonstrate this capability.
- You identified our entry only as "the AT&T system." In fact, it is called SC2D. The distinction is important, since SC2D is one of several research prototype cell layout tools being developed at AT&T for chip designs. These tools are compatible but distinct; they produce different styles of layout (for example, cell/strip-based, single/multiple devices per polysilicon column, abutted rows). SC2D is part of a much larger CAD system, called IDA, that includes editors, compacters, simulators, logic synthesis tools, and spe-

cial-purpose generators.

- Finally, you failed to mention what is perhaps SC2D's biggest advantage: it regularly handles tens of thousands of individually sized transistors at one shot, synthesizing dozens of strips.

Thank you for allowing us to set the record straight on these issues.

DWIGHT D. HILL, DON SHUGARD, and JACK FISHBURN
AT&T Bell Laboratories
Murray Hill, N.J.

EDITOR'S REPLY: The original idea of the article was to give our readers information about some of the different automatic cell layout systems that exist today. It was an informal project and there was no intention of running a "race" between the participating vendors. In the article, we discussed some of the features and results but deliberately avoided picking "winners" and "losers," since the ground rules were not defined well enough to provide a level playing field for all the players.

We included AT&T's tools, even though they are proprietary, since they offered another viewpoint on cell synthesis tools and included some advanced features that would be of interest to our readers.

However, we must apologize for the inadvertent error in mixing up AT&T's layout with that of another vendor.

We also agree that AT&T was the first to discover the bugs in the original circuit. As soon as the bugs were discovered, a corrected schematic was sent to all participants.

The original article was written by Ernest Meyer. Unfortunately, although Ernest received credit on the table of contents, his byline was omitted from the article itself.

Or...

The choice is really quite simple. Dozens of different PLDs expensively stocked to meet every need. Or the generic GAL® family of programmable high-performance logic devices.

Reprogrammable CMOS.

Behind this family is our Generic Array Logic (GAL) architecture. First developed and produced by Lattice, this architecture lets one GAL device replace all common 20 or 24 pin PAL® devices.

Fabricated with Lattice's proprietary E²CMOS™ technology, GAL devices give you bipolar speed and lower power. Half-power parts are available with maximum gate delays of 15ns. Quarter-power delays are 25ns. Both reduce heat without loss of performance.

Gone are discarded fuse-link parts caused by misprogramming. Gone is the need for expensive window packaging. Gone are long UVerase cycles. Instead, you simply plug your GAL device back into the programmer and erase and repro-

gram in less than a second. Design changes are simple and easy, too.

Inventory reductions.

From a manufacturing and management perspective, the GAL family offers even more. Now PAL and other PLD inventories can be dramatically reduced. At the same time, design alternatives multiply.

Because GAL devices are reprogrammable, they are also reuseable. And last-minute design changes mean simple reprogramming rather than costly replacement.

100% tested.

Unlike other PLDs, GAL devices are 100% tested for optimum system quality. There is simply no need to overstock in anticipation of a high failure rate.

So forget about that pile of "Eithers." And learn more about the Lattice "Or." To find out how GAL devices fit all of your PLD needs, ask for a free copy of the Lattice GAL Data Book today.

E2CMOS is a trademark of Lattice Semiconductor.
GAL is a registered trademark of Lattice Semiconductor.
PAL is a registered trademark of Monolithic Memories, Inc.

L Lattice
Semiconductor
Corporation ™

5555 N.E. Moore Ct. • Hillsboro, OR. 97124

*The GAL Revolution Starts Here*SM

CIRCLE NUMBER 9

The Tuning of a Logic Minimizer

MICHAEL C. GALVIANO, ALTERA CORP., SANTA CLARA, CALIF.

Upgraded tool reduces designs for EPLDs further and faster

Minimization of logic expressions is critical for the efficient use of PLD resources. If the needs of a designer's project exceed the fixed resources of a programmable logic device, the part becomes useless to the designer. Because designers' needs are growing, minimization programs must be upgraded to produce efficient designs in the short amount of time characteristic of PLDs.

From the beginning of the development of the A + Plus software package, for use with the company's erasable PLDs, Altera has put significant effort into the functional module assigned the task of product-term minimization. This minimization module, called the Standard A + Plus Logic Simplification Algorithm (Salsa), is also found in Altera's Max + Plus design system (Figure 1).

We have reevaluated Salsa to judge the effectiveness of its results against a widely used and understood minimization program, Espresso. In addition, we have upgraded the algorithms within Salsa to increase its throughput. The enhanced Salsa now resides in Altera's design tools.

Salsa has gone through several incarnations since its first release in 1984. Each release was

17 MIPS.
210K transistors.
And one chance
to get it right.

To build the world's highest performance RISC microprocessor, AMD turned to Mentor Graphics IC layout tools.

AMD's new RISC-based, 32-bit Am 29000 microprocessor operates at a 25 MHz clock rate with a 40 ns instruction cycle time. It can hit a peak execution rate of 25 MIPS, with a sustained performance of 17 MIPS. Because of its exceptional speed and price-performance ratio, the Am 29000 can be used in a wide range of applications—from embedded controller designs to engineering workstations.

It was a bold, ambitious project: build the fastest 32-bit μP in existence. One delivering a 5X to 7X performance improvement over existing 32-bit processors.

So Advanced Micro Devices turned to Mentor Graphics' Chip Station® to get the Am 29000 into silicon in a single iteration. Why? Because Chip Station provided the most advanced capabilities available, yet could also emulate AMD's existing tools.

Sharpen your competitive edge.

Look inside Chip Station and you'll find features like advanced traversal capabilities that make moves through the hierarchy as simple as point and click. And polygon-based editing that prevents problems like self-intersecting data. Also, programmable stroke recognition, which immediately converts cursor movements into commands. And now, the industry's fastest VLSI color plotting solution.

Complete compatibility with your existing environment.

Chip Station's programmable user interface shortens your learning curve by emulating the commands of your current system. At the same time, Mentor Graphics helps you adapt to the rich feature set of Chip Station and graduate to a truly advanced IC layout methodology.

Chip Station also brings you both the capacity and performance to deal with the coming generation of ULSI designs. Structured

Chip Design lets you represent cells symbolically so the organization and management of large designs is greatly simplified. And REMEDI™ expands DRACULA II™ LVS checks to include graphic correlation of errors in both schematics and layouts.

The emerging standard.

Mentor Graphics' IC design and layout tools have already earned a solid reputation for productivity on large, complex VLSI projects. As a result, we're the world's fastest growing supplier of IC layout systems.

It's all part of a vision unique to Mentor Graphics, the leader in electronic design automation. Let us show you where this vision can take you.

Call us toll-free for an overview brochure and the number of your nearest sales office.
1-800-547-7390 (in Oregon call 284-7357).

DRACULA II is a trademark of Cadence.

Mentor Graphics®

Your ideas. Our experience.
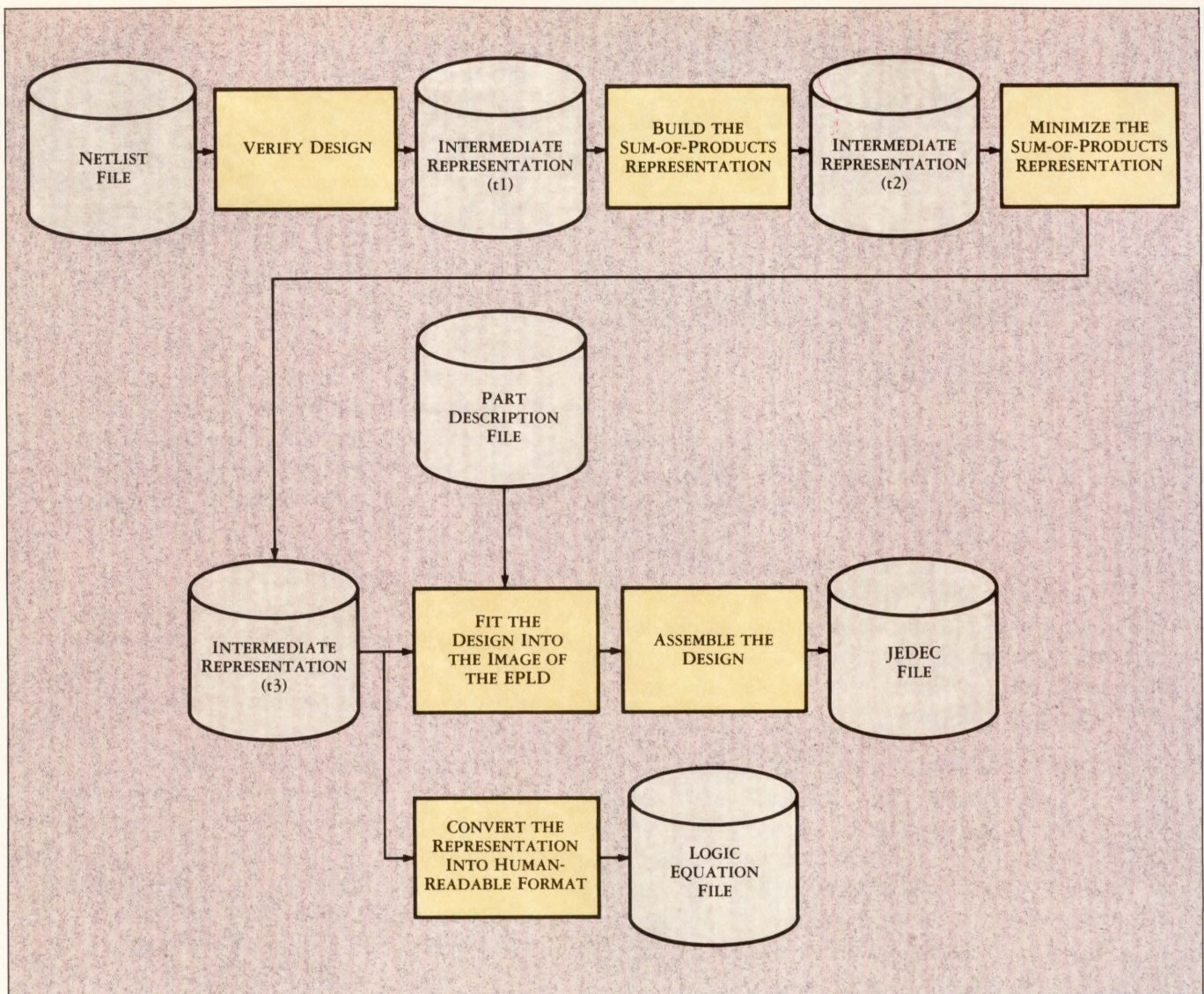
Figure 1. The A + Plus and Max + Plus design systems use the Salsa program to minimize the sum-of-products representations of designs that are targeted for Altera's EPLDs.

intended to improve the capabilities or speed of the program. When we became aware of the software offerings made available by the University of California at Berkeley, we had to decide whether to adopt its PLA minimization tool, Espresso, or to continue to refine Salsa. Espresso is a fine program that enjoys firm theoretical foundation. It has a reputation for being efficient and for producing results that were at least close to a given problem's optimum solution. The question of interest to us was whether it would improve our software's performance when used by our customers to create EPLD designs.

Because designers run the A + Plus and Max + Plus design software on personal computers, we needed to compare the code size and memory requirements of Espresso and Salsa. PC-DOS limits total code and data memory usage to 640 kilobytes. We found that although Espresso could be ported to run on DOS machines, its code was somewhat larger than Salsa's. We also know very precisely how much

data space Salsa requires because it has been tailored to work with the rest of Altera's design tool suite. The extent of Espresso's memory utilization depends on the particular problem being solved. It may use significantly more memory while processing complex expressions than it does for more simple ones.

We also needed to evaluate failure modes that resulted from expression complexity. Salsa always produces a logically correct expression—even if the expression is too complex to minimize sufficiently to fit within one of our EPLDs. If Espresso encounters an expression with too many prime product terms, or implicants, in either its complemented or uncomplemented form, the program aborts.

As mentioned above, Salsa was designed to work with the rest of Altera's tools. The I/O routines and file formats it uses fit comfortably into the tools' overall framework. We had some concern that processing speed would suffer and code size increase if we had to build special I/O

routines for Espresso. We were confident, though, that this potential problem could be overcome, should Espresso prove the best choice based on other criteria.

Another area of concern was what we call "non-Boolean minimization." Our EPLDs feature fully programmable I/O architectures. In many cases, the A + Plus and Max + Plus software must determine which of the possible flip-flop configurations available on a particular macrocell result in the best utilization of an EPLD. Altera has added algorithms to Salsa that select the best I/O configuration for customers' designs. Obviously, Espresso was not designed to include this function. We could pull the non-Boolean minimization functions out of Salsa and add it as a separate routine to Espresso. Such an approach, however, could reduce processing speed and increase code size.

Finally, we needed to evaluate maintenance and reliability issues. In the event that a software defect is found, we must be in a position to provide a timely response
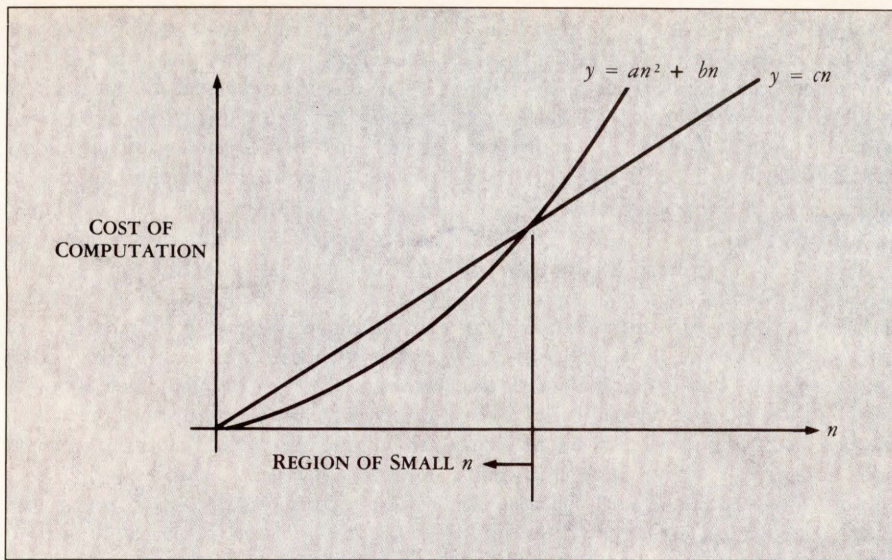
Figure 2. If the number (*n*) of implicants is kept small, *n*-squared algorithms (represented by $y = an^2 + bn$) can execute more quickly than linear algorithms ($y = cn$). "Pruning," or reducing, *n* is the primary method used to improve the performance of Salsa.

The graph labels: $y = an^2 + bn$, $y = cn$, COST OF COMPUTATION, REGION OF SMALL *n*, *n*

to the problem. Salsa is a mature, stable product with a well-understood, simple design. At each stage of its development, Salsa's progress was tracked through predefined milestones, with design issues receiving careful review by members of Altera's software development team. Also, throughout its history, Salsa has been subjected to Altera's large and growing regression-test database.

Salsa follows a classical model for the reduction of Boolean functions. First it generates all prime implicants in the solution; then it tries to throw out the most obviously redundant ones (Dietmeyer, 1978).

In contrast, Espresso is a more complex program written and supported by graduate students at Berkeley. Of some concern is the disclaimer that accompanies the Berkeley CAD-related software distribution (of which Espresso is a part). This copy of the disclaimer is dated March 1986:

"Please note that the Industrial Support Office/Industrial Liaison Program is not a technical services office. The Department and the ERL do not have the resources to offer technical support, although graduate student authors do appreciate being notified of bugs. If you have a technical question or a bug to report, you may send a brief description of the problem in writing to the Industrial Support Office and we will forward the letter to the appropriate professor or graduate student. However, we cannot promise any response, nor even an acknowledgement of receipt of your letter."

If we were to offer Espresso as a part of the Altera's design tools, it would obviously be our responsibility to maintain the

software. If we were to encounter a problem, however, we would clearly prefer help from the original source of the software. We feel that the users of our EPLDs would benefit indirectly from such help as well, and its absence adds a modicum of vulnerability that we would like to avoid.

■ MINIMIZE REAL DESIGNS

We decided that these potential areas of concern could be evaluated quantitatively. We would simply compare the programs in the problem domain of interest. We also know that the kinds of designs that will be successfully implemented in an Altera EPLD will be reduced to 10 or fewer product terms. In this domain, Espresso and Salsa would be compared according to two criteria: their relative speed and their relative effectiveness. By relative speed, I refer to the "wall clock" time required for each program to process the same data. By relative effectiveness, I refer to the number of product terms remaining in the expressions after minimization.

If we found Espresso to be significantly faster or significantly better at product term reduction than Salsa, we would assess the software development costs associated with achieving the best possible overall performance of Boolean and non-Boolean minimization. We would decide whether it would cost less to improve Salsa's performance relative to Espresso's, or whether it would cost less to provide the requisite interfaces to allow Espresso to work within Altera's tool suites.

We compared a stand-alone version of Salsa, which was functionally equivalent to the version released as a part of A + Plus and Max + Plus, with the version of Espresso that we ordered from Berkeley in

May of 1986. Espresso was run with its "single output" and "Boolean equation format" options set. This configuration results in an output suitable for comparison with Salsa's output. The only difference between the tested version of Salsa and the production version of Salsa was that the test version was initiated by a small driver program instead of from the A + Plus or Max + Plus environments. The actual program code was unchanged.

The tests were run under Berkeley Unix 4.2 on a Pyramid 90x superminicomputer. We used the Unix environment because it contains convenient profiling tools and because a PC-based version of Espresso was unavailable. Salsa was already in a form that allowed us to compile its code for execution under either environment. Unix's timer utility was used to guarantee that the test results were unaffected by other users on the system. On the Pyramid, the Espresso executable code is about 35 KB more than that of Salsa.

Obviously, we expect that the raw numbers would be different if the tests were run on a PC, but there is nothing in Salsa's code which takes advantage of the Unix environment or the 32-bit word size of the supermini. For the class of problems that Espresso and Salsa solve, the efficiency of the algorithms is many times more significant than the efficiency of the underlying platform. Thus we expect results in the PC-based product that are proportional to the supermini results, and experience has borne this expectation out.

The programs were run on a set of designs chosen from Altera's regression test database. These designs fall roughly into three categories:

1. Designs created by programmers to test some aspect of the software
2. Designs created by Altera's applications engineers to demonstrate specific features of Altera's parts
3. "Problem" designs submitted by customers

It is important to note that designs falling into any of these categories are likely to be harder to process than the "average" design. Therefore our test database provides data of more significance than a random sample of test designs.

As was expected, there was no significant difference in the product term counts produced by Espresso and Salsa. These results concur with a similar comparison of algorithms on p. 158 of Brayton's book (1984).

The results of the timing measurements, however, were surprising (see the table). For the 24 designs used in the test, Espresso has an average elapsed time of 81.6 seconds.

COMPARISON OF TIMING MEASUREMENTS FOR ESPRESSO, NONOPTIMIZED SALSA, AND OPTIMIZED SALSA

DESIGN	ESPRESSO	SALSA (NON-OPTIMIZED)	SALSA (OPTI-MIZED)
16bitpii	151.4	13.4	2.3
68kmemap	5.2	11.1	0.1
900pat4	243.2	6.2	4.9
bicrevd	4.7	50.7	0.2
bus682a	26.0	20.0	0.7
ep1800a	571.6	3.7	1.3
ex3	2.4	48.2	0.3
ex7	100.6	52.3	1.9
maxio18	41.4	3.7	0.2
mxglfb18	39.0	0.3	0.2
mxsig18	676.7	16.9	2.3
schweb12	14.4	29.1	0.2

NOTE: All times are given in CPU seconds.

Salsa's average elapsed time was 12.6 seconds. The difference in median times was not as great, but it was still significant: Espresso's median elapsed time was 12.2 seconds; Salsa's, 6.8 seconds. Espresso's best time was about 0.1 second; its worst was 571.6 seconds. Salsa's best time was less than 0.05 second and its worst was 52.3 seconds. Salsa was faster than Espresso on 16 of the 24 designs.

We concluded from these results that for the types of problems of interest to us, Espresso does not generally perform better than Salsa. Thus it seemed prudent to do what we could to enhance Salsa rather than replace it with another module.

To improve the performance of Salsa further, we used the Unix profiling tool to identify areas in the program where excessive amounts of time were spent. As a result, the optimization efforts focused on three areas: the pruning of so-called "n-squared" algorithms, the improvement of the De Morgan's inversion algorithm, and the improvement of the "end game" (where we try to throw out as many redundant prime implicants as we can).

N-SQUARED ALGORITHMS

Many of the methods used to expand the input Boolean function to its "prime implicants"—its fullest expression in terms of the inputs—are what we term "n-squared" algorithms. Such algorithms arise under many circumstances; for example, when each element of a list that contains n elements must be compared with all the rest, there will be $n \times (n-1)$ comparisons. Because the leading term is n^2, we say the comparison involves an n-squared algorithm.

The number of prime implicants of a Boolean expression using x variables grows as $3^x/x$. In this case, n-squared algorithms can quickly become unwieldy with increasing x. But that is not the end of the story. In the example given above of the $n \times (n-1)$ comparisons, both the n^2 term and the n term have the same coefficient (unity). That is not always the case in more complicated algorithms. Consider two algorithms (Figure 2): one that has n-squared behavior but with a small coefficient (a) on the n^2 term and one with linear behavior (the best-case behavior for such algorithms) but with a relatively large coefficient (c) on the linear term. In such a case, there will be a region of small n in which the n-squared algorithm will be less complex than the linear algorithm. Eventually, of course, the n^2 term will dominate as n grows large.

For EPLD logic minimization, which uses n-squared algorithms, we should try to "prune" n—that is, keep it as small as possible. When we optimized Salsa, we used the profiler to identify potential n-squared tasks and tried to modify the program to keep n from growing quickly. For example, when reading an expression into the program, we can apply the absorption theorem of Boolean algebra (A + A*B = A) on on each product term as it appears. Although using the theorem introduces a new processing step, this technique is much faster in most cases than one big absorption of the entire expression after it is input to the program. Because the expression is subsequently expanded to prime implicants, this initial pruning has no effect on the eventual output of the program.

ORDER OF EXPANSION

Another opportunity for speed improvement through pruning comes from the order in which product terms are expanded into prime implicants. We found that certain prime implicants had a good chance of absorbing other implicants and thereby reducing the complexity of the expression; others had a lower probability. Furthermore, some implicants were more likely to generate many prime implicants than were others. Because each new prime implicant must be checked for absorption with the partially generated set, it is obviously to our advantage to expand the ex-pressions in the order that causes our prime implicant set to grow most slowly.

The criterion we chose for our order of expansion was the degree of "unateness" (Brayton, 1984). Unateness relates to how often a given literal (variable) appears in both its complemented and uncomplemented forms in a particular expression. Such occurrence of a literal is directly related to the likelihood of eliminating redundant implicants. If a literal is "very unate" it appears mostly either in its complemented or uncomplemented form. A literal that appears about equally in both its complemented and uncomplemented forms is "not very unate."

The actual expansion algorithm we chose produced few prime implicants in the early stages of its expansion if it expanded terms containing the most unate variables first. Thus we have another way of keeping n small—at least during the initial phases of the expansion.

DE MORGAN'S INVERSION

Next, we optimized execution of De Morgan's inversion algorithm. Salsa's basic minimization algorithms do not always use logical inversion. Sometimes, however, the minimized, complemented form of an equation has fewer product terms than the uncomplemented form. Salsa first minimizes the expression passed to it and then determines if the minimization of the expression is likely to benefit from inversion. If so, the expression is inverted and minimized and the results of the two minimizations (inverted and noninverted) are compared. Unless the user requests otherwise, the form with the fewer product terms is retained.

Logical inversion may be very expensive (for example, see the discussion of Espresso's "Achilles' heel function" on pp. 130 and 173 of Brayton, 1984). We want to avoid doing inversion whenever we think it will not improve the results. When we do invert an equation, we want it to complete quickly or give up when it seems that the expression is starting to get very large. To do so, we developed two proprietary heuristics that allow Salsa to decide when to avoid and when to abandon an inversion attempt. We also found pruning useful in keeping the inverted form of the expression as small as possible during the inversion process.

The heuristics were verified by a detailed analysis of our regression test database. It is, however, theoretically possible for the inversion process to contain a steep "hump," or local maximum, in the product term count. In such cases, the product term count would grow for a time during inversion but would eventually shrink to

```
A' * B * C + A' * B * D + A' * C * D'

[ A ]


A' * C * D' * ( A * B * C + A' * B * D )'
  = A' * C * D' * ( A + B' + C' ) * ( A + B' + D' )
  = A' * B' * C * D' * ( A + B' + D' )
  = A' * B' * C * D'

[ B ]


A' * B * D * ( A' * B * C + A' * C * D' )'
  = A' * B * D * ( A + B' + C' ) * ( A + C' + D )
  = A' * B * C' * D * ( A + C' + D )
  = A' * B * C' * D

[ C ]


A' * B * C * ( A' * B * D + A' * C * D' )'
  = A' * B * C * ( A + B' + D' ) * ( A + C' + D )
  = A' * B * C * D' * ( A + C' + D )

[ D ]
```

Figure 3. For the first expression (A), the AND-with-complement algorithm produces new implicants for the third implicant (B) and the second implicant (C). For the first implicant, however, the result is null, so that implicant is redundant and therefore unnecessary (D).

contain fewer product terms than in the original expression. For such cases, we allow the user to circumvent the heuristics by requesting control of logical inversion on an equation-by-equation basis. In addition, we carry on with the inversion process in cases where the noninverted form of the expression is too large for implementation in his EPLD. In these cases, the customer has little to lose: either the inverted expression is worth waiting for or else the design doesn't fit in the target EPLD.

■ THE END GAME

Finally, we replaced our extremely simple end-game routine with a straightforward routine that successively performs a Boolean AND of the complement of each prime implicant with the rest of the expression—the AND-with-complement (AWC) algorithm. If this algorithm produces a split (that is, generates a new implicant), the prime implicant that produced the split is retained. On the other hand, if an implicant and the complement of the rest of the expression produce a null result, then whenever the implicant is true, the rest of the expression is true and the implicant is redundant.

To demonstrate, Figure 3A shows an expression prepared for the AWC algorithm. Performing a logical AND of the last implicant with the rest of the expression (Figure 3B) results in an expression that may be true (a new implicant). The AWC produces a similar result for the second implicant (Figure 3C). For the first implicant, however, the result is always

null because the terms within the parentheses will be ANDed with their complements. The first implicant is therefore redundant and can be thrown out.

Our end-game algorithm is easily extensible to allow some splitting should we encounter many expressions that are close to, but not quite yet, fitting in an EPLD.

■ CONCLUSIONS

The table shows the results of the same set of designs run under the same set of conditions using the optimized version of Salsa. In all cases, Salsa performs substantially faster than Espresso. Even in the case in which Espresso performed at its best, Salsa was approximately eight times faster. Salsa's average elapsed time was less than 0.8 second per design.

Lest these results be misinterpreted, I would like to restate that I feel that Espresso is a very fine program. It is optimized for large systems of Boolean equations encountered in VLSI designs rather than for the somewhat more modest needs of PLDs. It seems clear, however, that for the types of designs we are likely to encounter with our tools, our decision to retain and optimize Salsa is will address our customers' expectations in the area of logic reduction. ■

REFERENCES

BRAYTON, R., ET AL. 1984. *Logic Minimization Algorithms for VLSI Synthesis*, Kluwer Academic Publishers, Dordrecht, the Netherlands; Hingham, Mass.; Lancaster, U.K.

DEITMEYER, D. 1978. *Logic Design of Digital Systems*, 2nd ed., Allyn and Bacon, Newton, Mass.

ABOUT THE AUTHOR

MICHAEL GALVIANO *is currently a researcher at Stanford University. When he wrote this article, he was a software engineer at Altera, where he designed, developed, and supported CAD tools. Before that, he spent four years at NCA, a CAD company. He holds a BS in physics from California Polytechnic State University in San Luis Obispo and a PhD in mathematical physics for the New Mexico Institute of Mining and Technology.*

HARDWARE

M·O·D·E·L·I·N·G

L. CURTIS WIDDOES JR. AND HOLLY STUMP,
LOGIC MODELING SYSTEMS INC., SAN JOSE, CALIF.

A clear consensus is developing among the engineers who are designing state-of-the-art digital electronic systems: whatever doesn't get simulated usually doesn't work. This rule has been known to designers of complex ICs for over a decade and is now being painfully learned by system designers as their printed circuit boards become as complex, unobservable, and immutable as the ICs they contain. Fortunately, the industry has finally developed simulation environments that meet the system designers' needs. These include major improvements in workstation performance, simulation algorithms, integration of tools, human interfaces, special-purpose hardware, and simulation model libraries.

HARDWARE MODELERS MAY BE THE IN-CIRCUIT EMULATORS OF THE 1990s

One of the key advances in system simulation was the introduction of hardware modeling in 1984. Before hardware modeling, accurate, full-function simulation models weren't available for most complex devices, which made it impossible to fully simulate systems containing these devices. However, access to today's hardware modelers can guarantee an engineer accurate simulation models of complex devices on demand—as long as the devices are available. To model a new device the engineer simply plugs a physical "reference element" into the modeler.

Figure 1. In addition to modeling standard devices and existing ASICs, first-generation hardware modelers allow the design engineer to run real diagnostic programs under simulation, verifying the operation of the entire system working together.
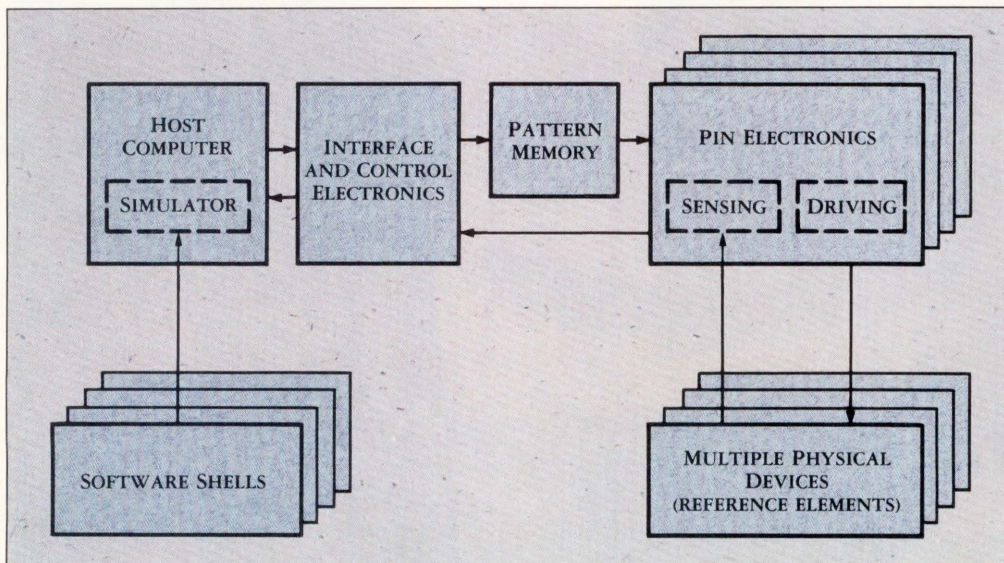
Figure 2. A composite hardware modeler would allow a large number of reference elements to be used simultaneously. When an event occurs on a modeled device, the simulator would call the modeler to perform an evaluation one model at a time.

The number of complex devices that are available to the system designer is increasing exponentially as new tools bring IC design into the reach of more engineers. Ten years ago there were only a few standard devices for which hardware modeling was appropriate; today there are hundreds. When ASICs are included, the system designer is buried in devices for which no other modeling method is appropriate.

However, hardware modeling is a relatively new technology that is not widely understood. In particular, there has been no comprehensive exposi-

tion of the key hardware modeling attributes that differentiate current products. Although U.S. patents 4,590,-581 and 4,635,218 describe the fundamentals of early hardware modeling technology, there have been few papers published on the topic. (The papers that have been published are listed in the bibliography at the end of this article.)

This article describes current hardware modeling technology and how it relates to other modeling techniques, testers, and in-circuit emulators. It describes their use and lists key modeler attributes for

evaluating current products.

■ TYPES OF SIMULATION MODELS

Simulation models can be divided into three major classes: behavioral, structural, and hardware models. Behavioral models represent device behavior using code written in an algorithmic language such as C. They are compiled and linked with the simulator and run on a general-purpose computer. Structural models come in many varieties, including functional, gate-level, and switch-level. Basically, a structural model mimics the actual internal structure of a

device and is usually a by-product of the design. Hardware models couple an actual known-good physical device (called a "reference element") with the simulator. Events on the device's input pins drive the reference element, and the functional response of this element is sampled and returned to the simulator, with output timing inferred from a table in a simple text file called a "software shell." The shell also specifies pin names, pin types, output delays, and other information normally found on the device data sheet.

Any of these model types can be either full-function or reduced-function. Full-function models correctly mimic the full logical behavior of a real device for any legally timed input stimuli. Reduced-function models mimic only parts of the logical behavior range of a device, producing unknown or incorrect results in response to certain legally-timed input stimuli. Behavioral models are usually reduced-function except for simple devices; for more complex devices, the amount of the reduction in function is variable and not necessarily specified. Structural models are usually full-function. Theoretically, hardware models are full-function by definition, although some current products fall short of this ideal (for instance, some modelers are unable to measure the high-impedance state of three-state output pins).

Behavioral, structural, and hardware models are complementary; they excel in different applications, and they can generally be used together in a single simulation. Behavioral models are most appropriate for low- and medium-complexity devices and for reduced-function models used for coarse design verification early in the design cycle. Structural models are most appropriate for very simple devices or for complex devices before first silicon. Hardware models are ideal for full-function models of medium- and

WITH THE WRONG ASIC DESIGN TOOLS, YOU COULD BLOW THE WHOLE PROJECT.

WITH VLSI'S ASIC TOOLS, YOU CAN

VLSI Technology's software is the fastest, surest way to create successful ASIC chips.

Our tools accelerate each and every step of the design process.

And our years of experience insure the success of your finished ASIC chip.

EVERYTHING'S FASTER WHEN YOU HAVE AN ASSISTANT.

Our Design Assistant™ tool partitions your chip and estimates chip size, power consump-
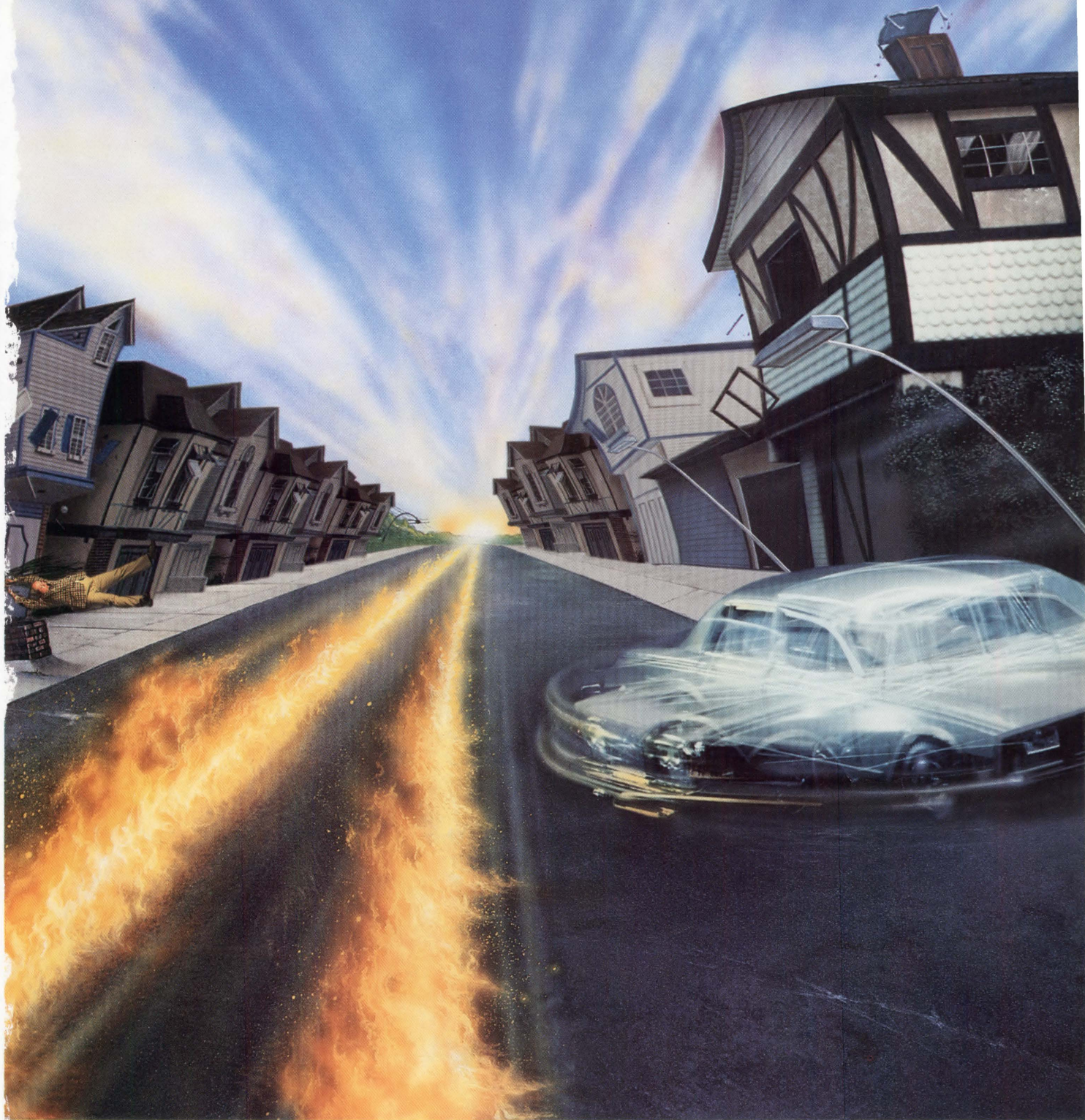
BLOW RIGHT BY THE COMPETITION.

tion, and packaging possibilities. In short, it tells you the best silicon solution to your problem.

Simply enter your design in block diagrams and global interconnect forms. Design Assistant shows you ways to implement it.

Gate array, standard cell or cell-based.

You can run dozens of "what if" configurations in a few hours. In no time, you'll have the crucial information you used to wait forever for vendors to supply.

STREAMLINED DESIGN, COMPLIMENTS OF OUR COMPILERS.

Our logic and memory compilers create multipliers, PLAs, or super fast SRAM memory blocks as fast as 8ns.

And we've got a couple of "expert" compilers for cell-based or gate array designs that think exactly the way you do.

When you design a datapath, you think of it as a linear schematic, right?

Well, our Datapath Compiler just happens to use schematics as input. Complex multi-bit datapaths practically pop right out of it.

When you design state machine and other logic blocks, you think in terms of equations. Our State Machine Compiler does, too.

And it even optimizes your equations.

GET IT TOGETHER. FAST.

Ready to place and route your cell-based design? Just call up our Chip Compiler.

It provides floor-planning and auto-routing. It works with our compilers and standard cell libraries to produce an optimized layout.

It even pours standard cells into the gaps between blocks to make sure you use the minimum chip area required.

How fast does it work? Glad you asked.

A company came to us with a layout that took them three months. We did it in two days.

TO BE FAST, YOU HAVE TO BE FLEXIBLE.

Only our Portable Library lets you choose gate array or cell-based, 1.5μ or 2μ, *after* you've completed your logic design.

It can allow you to do that because your library always remains stable. No matter what process you use.

Process obsolescence is now obsolete.

IF YOU NEED TO GET THERE FAST, TAKE THE EXPRESS.

You can buy VLSI's tools in six configurations. From our quick, easy-to-use Logic Express™ to the powerful Design Express.™

If you'd like to find out how quickly you can design successful ASIC chips, give us a call at (800) 872-6753.

Because when you're driving to market this fast, it's good to have insurance.

CONCEPT EXPRESS™:
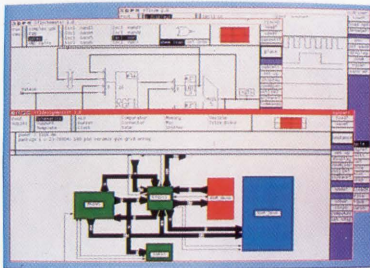
The Concept Express Design System's highly productive logic tools and silicon compilers were used to develop this very-large-scale ASIC. It incorporates a 2901 datapath, RAM, ROM, and over 3,400 gates of random logic.
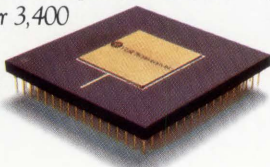
DESIGN EXPRESS™:
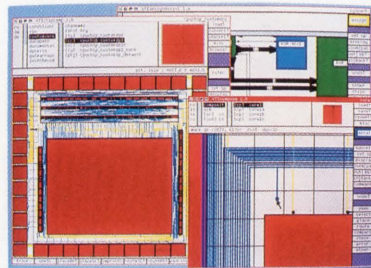
This highly-integrated design combines control logic, a register file, a refresh counter, and five peripheral chips onto a die size of 275x315 mils. The logic design, layout, and verification were completed in only 12 weeks.

SILICON EXPRESS™:
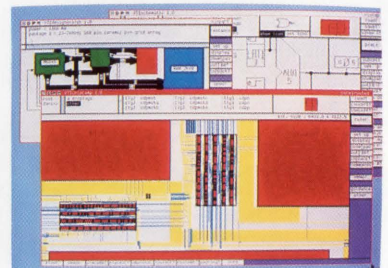
This design integrates all the peripheral chips for an AT computer with six megacells and control logic. Using the Silicon Express Design System, logic and physical designs like these can be implemented in under two man months.

VLSI TECHNOLOGY, INC.

high-complexity devices after first silicon, particularly ASICs.

■ HARDWARE MODELS VS. BEHAVIORAL MODELS

The primary strength of hardware models is that they are readily available. Typical vendor libraries contain hundreds of full-function hardware models of complex standard devices. In addition, the user can create a new hardware model in a few days. In contrast, very few full-function behavioral models are actually available for complex devices. Ten years ago a full-function behavioral model of the most complex microprocessor could be developed in less than a man-year; today, a full-function behavioral model for a modern microprocessor requires 10 to 20 man-years and amounts to between 100,000 and 200,000 lines of code. Clearly, tools for conceiving and implementing advanced devices have far outpaced tools for formally describing their behavior.

Full-function hardware models can be available as soon as silicon operates. Since hardware modelers use the reference element only for their logical functions (timing is derived from a table), the models can even use preproduction silicon. The early silicon does not need to run at full speed, because the hardware modeler will return correct results as long as the device functions logically.

Complex devices by definition have a large number of behaviors, and they often have undocumented or poorly documented behaviors that must be deliberately coded to create a behavioral model. In particular, full-function models for use in fault simulation must produce correct responses even for "nonsense" stimuli that result from faults elsewhere in the system. Coding a behavioral model to correctly respond to all nonsense stimuli in fault simulation is so difficult that behavioral models are usually called "full-function"

even when they don't have that capability. The problem is compounded by the need for behavioral models to work with multiple platforms and simulators.

Another advantage of hardware models is that they always produce correct results, given that the hardware modeler functions properly and the reference element is good. Indeed, a hardware model is inherently more accurate than the device manuals and data sheets and can be used to cross-check them.

In contrast, it's difficult to verify the accuracy of behavioral models. A full-function behavioral model of a complex device is comparable in size to a small operating system. To find the bugs in such a program, test vectors must be developed that stress all the device's behaviors—manufacturing test vectors are not sufficient. In addition, some device manufacturers do not make manufacturing test vectors available, since they can reveal proprietary technology.

■ HARDWARE MODELING MODES

First-generation hardware modelers are currently being used in several distinct modes (Figure 1). Their primary mode of use is modeling standard devices and existing ASICs during system simulation. Since hardware models are normally full-function, the design engineer using them can run real diagnostic programs under simulation, verifying the operation of the entire system. The simulator can do better verification than a prototype, because the simulation is easier to stimulate and observe, and it allows analysis of worst-case timing.

Entire existing subsystems such as PC boards can be used as reference elements in a hardware modeler. For example, a design engineer could simulate the design of a new personal computer CPU board that operates with an existing graphics controller board without needing schematics and detailed

specifications of the controller.

As the performance of workstations increases, system simulation using hardware modeling is becoming the preferred tool for debugging embedded software and microcode. Before hardware modeling made full-function models of complex devices widely available, engineers counted on using the prototype for software debugging. But embedded software is often inaccessible, and changing it can affect the hardware design. Today, a high-performance simulator and a hardware modeler can be used together to verify thousands of executed instructions before the first system is fabricated.

Hardware modelers also are used to examine the behavior of existing standard devices and ASICs for which specifications are inadequate or even inaccurate. In this mode, the design engineer uses the device in question as a reference element, stimulating and observing it with the simulator's normal user interface.

For the ASIC designer, hardware modelers can fill a critical need. An important step in the design of a complex new ASIC is verification of its design in the context of its embedding system. According to both ASIC vendors and customers, approximately 50% of the ASICs that are designed and verified in isolation do not operate properly within the system. The problem is that an ASIC generally has complex interactions with its environment, some of which fail to be considered during the design process. However, hardware modelers give the design engineer access to full-function models of all the complex devices in the embedding system and enable simulation of the entire system before ASIC fabrication.

The ASIC design engineer is also interested in the functional verification of prototype ASICs. In this mode, the engineer simply uses the prototype ASIC as the reference element in a hardware modeler in place of the earlier gate-level model

and runs the same simulation test cases that were run during the design phase. Normally, the system simulation runs much faster when a hardware model replaces a gate-level model. Tests can be altered or added interactively through the simulator's normal user interface, and test failures can be diagnosed like design problems. The engineer doesn't need to create or diagnose test vectors.

■ HARDWARE MODELING THEORY

Let's now consider hardware modeling theory in the context of a hypothetical system combining various features of currently available hardware modeling products. *No existing product incorporates all the technology discussed here.*

Our composite hardware modeler allows many reference elements to be used simultaneously, each mounted on a simple passive adapter board and connected to a set of active pin electronics (Figure 2). If the system being designed contains several different types of complex devices, reference elements for all types must be mounted in the hardware modeler before simulation begins. When an event occurs on an input of a hardware-modeled device instance in the simulated design, the simulator calls the hardware modeler to perform an evaluation, which occurs one model at a time in response to simulator requests.

To a first approximation, evaluation of a hardware model in response to an event on an input proceeds as follows:

The modeler forces the reference element to the internal state that the device instance had prior to the current event, presents the event on the appropriate pin, samples all outputs of the reference element after the device has settled, and finally returns any changes to the simulator with scheduling delays attached.

The scheduling delay for each output event is inferred from a look-up table in the software shell.
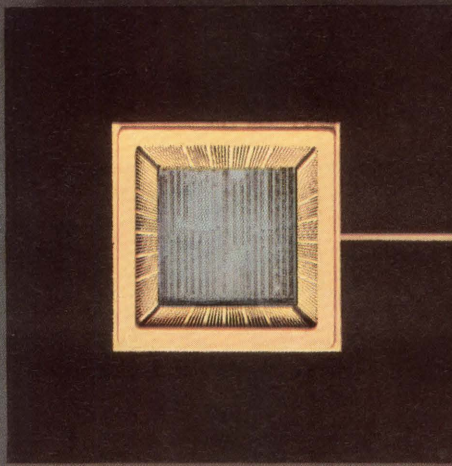
50,000 Gates At 0.7 Nanoseconds.

Toshiba's process technology has made us the world's leading supplier of CMOS devices. From this experience comes the capability to build a 50,000 gate ASIC.

And from Toshiba's manufacturing capabilities comes the volume production to meet your smallest or greatest needs.

These new Toshiba "Sea of Gates" Compacted Arrays™ provide high packing density for more effective silicon utilization and ultra-high speed. And they are available in 5 base arrays with usable gates from 15,000 to 50,000.

This new family is compatible with Toshiba's existing channelled array families, using compatible macrocell libraries and CAD software.

If you are working on advanced products – or working to advance existing products, talk to Toshiba at one of our conveniently located Design Centers. We're leading the way. Toshiba. The power in gate arrays.

TOSHIBA. THE POWER IN GATE ARRAYS.				
	3 MICRON CHANNELLED ARRAY	2 MICRON CHANNELLED ARRAY	1.5 MICRON CHANNELLED ARRAY	SEA OF GATES COMPACTED ARRAY™
SERIES	TC15G	TC17G	TC19G	TC110G
GATES	880 to 6,000	540 to 10,000	3,200 to 10,000	2,100 to 50,000
DESIGN RULE	3µm	2µm	1.5µm	1.5µm
GATE SPEED	2.5ns	1.5ns	1.0ns	0.7ns
PART NO's	6	9	5	5
AVAILABILITY	NOW	NOW	NOW	NOW

All Si-Gate CMOS double layer metal. Compacted Array is a trademark of LSI Logic Corporation.

RAM & ROM Memory In CMOS ASICs.

Now add custom-built memory to your standard cell designs. Add ROM in various word and bit combinations up to 16 kbits. Add RAM in word and bit configurations up to 4 kbits.

Based on Toshiba's proprietary 2-micron, double-metal CMOS process, this new series of devices is compatible with existing and future Toshiba standard cell and gate array families.

These advanced devices integrate up to 10,000 gate equivalents with typical speeds of 1.5 nanoseconds for a loaded 2-input/2-output NAND gate.

Look to Toshiba for the newest advances in custom LSIs and for the manufacturing strength that means unmatched production capability. For complete details, call your Regional Sales Manager at one of Toshiba's Regional Sales Offices:

NORTHWESTERN: Sunnyvale, CA (408) 737-9844.
SOUTHWESTERN: Tustin, CA (714) 259-0368.
NORTH CENTRAL: Chicago, IL (312) 945-1500.
SOUTH CENTRAL: Dallas, TX (214) 480-0470.
NORTHEASTERN: Burlington, MA (617) 272-4352.
SOUTHEASTERN: Atlanta, GA (404) 368-0203.

TOSHIBA. THE POWER IN STANDARD CELLS.		
SERIES	TC2ISC	TC22SC
GATES	MAX. 10K	MAX. 10K
CAD	VLSI CAD-I	VLSI CAD-IA
DESIGN RULE	2.0 μm	2.0 μm
SPEED	1.5 ns	1.5 ns
MACRO FUNCTIONS	TC17G Gate Array MACRO Cell	TC21SC MACROs, plus: RAM 4K ROM 16K Functional MACROs 74 Series
AVAILABILITY	NOW	NOW
All Si-Gate CMOS double layer metal.		

TOSHIBA AMERICA, INC.

Figure 3. Although testers have an advantage in measuring dc parametrics and high-precision timing characterization (A), hardware modelers permit the design engineer to easily alter the circuitry or embedded software at the simulation stage (B).

For static devices, which can retain their internal state indefinitely without a clock, evaluation can be simple. Assuming that the hardware modeler contains one reference element for each device instance in the design, the current internal state of each device instance is stored in its corresponding reference element. In this case, the hardware modeler simply skips the first step of the evaluation procedure (forcing the state).

However, dynamic devices usually cannot retain their internal state between evaluations, because the simulator response time is too great. For such devices, the state-forcing step is not trivial: To a first approximation, the hardware modeler must first force the device to a *fixed* internal state (for example, by resetting it), then present the entire sequence of events produced by the simulator from the time simulation began until just prior to the current event (we call this the "history sequence"). As long as the device behaves deterministically, the history sequence will restore the device's internal state exactly. The hardware modeler then proceeds with the evaluation of the dynamic device in the normal way, by presenting one additional event and sampling the result. After evaluation, the internal state of the device is allowed to deteriorate or to be overwritten.

We call the first mode of operation "private," since the reference elements used to store the device's internal states cannot be shared, even between multiple device instances within a single design. We call the second mode of operation "public," since a reference element used in that way can be shared between multiple instances and multiple users.

An advantage of the public mode (see the table) is its ability to support fault simulation by allowing concurrent evaluation of multiple faulty device instances. Whereas static devices can be operated in either the private or the public mode, dynamic devices can be operated only in the public mode. However, the public mode imposes limits on simulation length and evaluation speed, since the history sequence must be stored in a memory of finite size and presented completely for each new evaluation. The private mode imposes no such limits.

■ HISTORY SEQUENCE COMPRESSION

The history sequence of public mode operation is stored as a sequence of patterns in fast memory, each pattern consisting of 2 bits for each reference element input pin. The 2 bits represent the state of the device pin's "external net," that is, the net connected to the device pin in the simulated circuit. States of the external net can be hard 0, hard 1, high impedance, or soft 1. To present a history sequence, the hardware modeler presents successive patterns of the sequence at successive times to each input pin of the reference element, thus imitating the action of the entire simulated external circuit on the simulated device. It presents these patterns synchronously with a variable-frequency "pattern clock."

Since the pattern memory can't be arbitrarily fast and large, the hardware modeler uses several methods of pattern sequence compression to dramatically reduce history sequence lengths without altering the logical effect of the history sequence on the reference element.

The first compression method is based on distinguishing between two fundamental device input pin types. Device input pins are separated into pins that *cause* internal device state to be stored (called "store pins") and pins that can only affect the *value* of the internal state stored (called "data" pins). Formally, a data pin is an input pin that, when transitioned at any frequency any number of times and then returned to its initial state, has no effect on the internal device state regardless of the states of any other inputs, assuming all resulting pin states and timing are allowed within the device specifications.

From this definition, it is

clear that the order of data-pin transitions relative to each other cannot affect the device's internal state. Therefore, in compressing the history sequence, the hardware modeler combines all data-pin transitions occurring between any two store-pin transitions into a single pattern. This compression still preserves the order of store-pin transitions relative to data- and other store-pin transitions. Since most input pins tend to be data pins, a very large compression factor is achieved.

The type of each pin must be declared in the software shell. If there is doubt about a particular pin, it must be declared as a store pin. The penalty of misdeclaring data pins as store pins is a lower compression factor, but the model will still deliver accurate results.

The next compression is achieved by placing both store-pin and data-pin transitions in the same pattern. For some modelers, placement of store-pin transitions is programmable, as in testers. Each pin electronics channel is identically programmable, so that any pin can be programmed to be a store pin. Store-pin transitions are appropriately delayed relative to data to satisfy the reference element's worst-case setup time and hold time requirements. For hardware modelers without this feature, separate patterns must be used for store-pin and data-pin transitions, reducing the modeler's effective pattern capacity and maximum device-clock frequency—each by a factor of two.

Further compression is achieved by allowing programmable return-to-one and return-to-zero formats for store pins. For devices that sample data on only one edge type, such as the 80186 microprocessor, both edges of a single store-pin pulse can therefore be represented in a single pattern. For hardware modelers without this feature, the capacity and frequency are *again* reduced by factors of two.

The number of patterns that can be stored for presentation to reference elements is not relevant for the private mode, but for the public mode it limits the total duration of the simulation (in simulated time) that can be supported by the hardware modeler.

■ PATTERN CAPACITY

History sequences will grow as the simulated time advances. However, with the use of the compression techniques discussed above, growth is limited to a single pattern for each store-pin transition. For example, a history sequence for the 68020 microprocessor would grow by one pattern for each simulated clock pin transition, or about 10 patterns per simulated microprocessor instruction. Simulation of 10,000 microprocessor instructions would require about 100,000 patterns, each containing 32 bytes, for a total of 3.2M bytes.

Each instance of a public device in each simulated design requires its own history sequence. Consequently, if two users were simultaneously sharing a hardware model of the 68020, 6.4M bytes would be required to store both history sequences.

The growth rates for history sequences depend on the number of store pins that are transitioning and the frequency of the transitions in simulated time. Therefore, even for a single device instance in a single design, the history sequence may grow at different rates from simulation to simulation. In fact, history sequence growth rates vary greatly from device type to device type and from design to design.

Note that our composite modeler allows a large number of reference elements to be mounted simultaneously, but they are not used simultaneously. Therefore, the composite modeler contains a single large pattern memory that can be used to contain history sequences for any of the reference elements. One long simulation can use the entire mem-

ory, or multiple device instances or multiple users can share the memory. The pattern memory is dynamically allocated in response to requests from all the simulators.

■ HARDWARE MODEL EVALUATION TIMES

Private-mode evaluation of a hardware model consists of presenting a single pattern containing the new input event, sampling the outputs, and returning the changes together with scheduling delays. Because only a single pattern is presented for each evaluation, the evaluation time is extremely short and does not increase as the simulation progresses. In fact, the overhead of communicating with the simulator is always much larger than the actual evaluation time for a private device.

The evaluation time for a public device is proportional to the length of the history sequence and is greater than that for a private device by the amount of time required to present the sequence.

For the 68020 example, after simulating 10,000 instructions, the history sequence would be about 100,000 patterns. If we assume a pattern frequency of 16 MHz, the next evaluation would require 6.25 ms, plus the normal overhead of a private-device evaluation and communication with the simulator. However, a single evaluation can create multiple output events, and the total evaluation time is small compared with the time required by most software-based simulators to simulate the circuitry surrounding the hardware model in preparing the next event for presentation to the hardware model.

The use of archive memory for storing history sequences can have a much more dramatic effect on evaluation time. All hardware modelers store the current history sequence in a high-bandwidth memory so that pattern rates keep the dynamic devices refreshed. Because fast pattern memory is

expensive, some hardware modelers use either disk or slow RAM as an archive for all but the current history sequence. Unfortunately, before each evaluation the current history sequence must be transferred from archive memory to fast memory, increasing the evaluation time.

Again consider the 68020 example using a hardware modeler with an archive pattern memory having a transfer rate of 2 MB/s. After simulating 10,000 instructions, the history sequence consists of 3.2 MB, and the transfer time is 1.6 seconds—or 256 times longer than a modeler that doesn't use archive memory.

Finally, in computing the evaluation time for a hardware model of a public device, it is necessary to use the *effective* pattern rate. Modern complex devices require very wide patterns. For example, the 68020 requires patterns that are 128 pins wide. Most hardware modelers "fold" wide patterns to fit them in a narrow pattern presentation memory, reducing the effective pattern presentation rate for wide devices. A hardware modeler having a 16M-pattern/s memory 64 pins (128 bits) wide would have an effective pattern presentation rate for the 68000 of 16M patterns/s, but the effective pattern presentation rate for the 68020 would be halved.

This reduction in pattern rate due to folding wide patterns is tied to a reduction of the maximum attainable device clock frequency, which in the case above would be 8 MHz for the 68000 and 4 MHz for the 68020.

■ SAMPLING THE OUTPUTS

After forcing the reference element to the desired internal device state and presenting a stimulus event, the hardware modeler waits for the reference element to settle and samples all outputs simultaneously. It returns any observed transitions to the simulator, with scheduling delays attached.

Each input event causes a
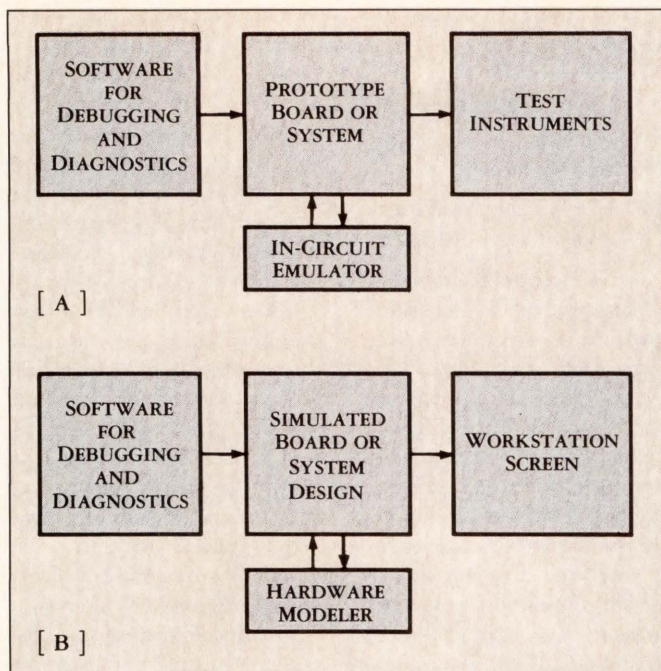
Figure 4. In-circuit emulators are used to debug system hardware and software after fabrication of a prototype system (A), but hardware modelers can be used for debugging in the simulation stage, before construction of a system prototype (B).

separate evaluation to occur, and after each evaluation the hardware modeler returns at most one output transition per output pin. The transition returned, if any, is the transition that exists after the device settles; pulses and glitches are lost. To be suitable for hardware modeling, then, a device must produce at most one *significant* transition on each output in response to a single input transition. This restriction is normally not a problem, unless the device has an internal oscillator or pulse generator.

Much of the utility of a TTL system simulation is lost if the high-impedance (Z) state is not simulated correctly. Therefore most hardware modelers directly measure three states of each output pin (0, 1, and Z) and return the measured result to the simulator. Hardware modelers that cannot directly measure the Z state of output pins depend on the hardware model's associated software shell to predict whether each pin is a Z and to overlay that information on the measurement taken by the hardware modeler. In order to do this prediction, much of the internal device state needs to be known to the software shell.

Obviously, coding such a shell is exceptionally difficult and is generally not practical for the user creating his own model.

■ DEVICE INITIALIZATION

Before a history sequence is presented to a public device, the device must be initialized to a known internal state. If some unknown internal state remains when the history sequence starts, it could cause unpredictable errors. In fact, different evaluations at different times within a single simulation run will be potentially inconsistent.

Many devices can be forced to a known internal state by activating a reset pin or presenting some other simple sequence of patterns to the device. However, some devices have internal states that cannot be initialized in this way. Usually, such states are part of free-running counters. For example, the counter which produces the E signal of the MC68010 cannot be directly reset.

An initialization technique used by testers is applicable and used by several hardware modelers: the hardware modeler simply loops, presenting a

specified sequence of patterns until it senses a specified feedback condition from the reference element. For the 68010, such a loop would toggle the clock pin repeatedly until the E signal transitioned to a 1. At that point the state of the E counter would be known.

Less sophisticated hardware modelers require the user to design custom logic and to install it on the device adapter with the reference element. Obviously, this initialization method can be a problem for the user creating a hardware model.

Some dynamic devices contain a substrate bias generator that requires a clock in order to function properly. Initialization times for such bias generators are very long. The composite hardware modeler includes a "keep-alive clock" to keep the substrate bias generator running even when patterns are not running, so that the device will operate as soon as the next evaluation begins.

Hardware modelers fundamentally depend on complete device initialization, in the sense defined above, and on determinate logical behavior for all legally timed input stimuli. Fortunately, testers depend on the same attributes, so that any device that can be tested can also be modeled in a hardware modeler.

■ COMPARISON WITH TESTERS AND IN-CIRCUIT EMULATORS

Figure 3 illustrates the key difference between hardware modelers and testers: the modeler is coupled with the simulator so that the hardware models appear to be embedded in the simulated design. Indeed, the device outputs sampled by the hardware modeler are fed back into the simulated design and are used by the simulator to compute future device inputs. The design engineer can change this feedback loop by altering the circuitry or embedded software in the simulated design. Some simulators allow this alteration to be done interactively. The

design engineer can use all of the simulator user interface for stimulating and observing the hardware model and diagnosing problems he finds in its interaction with the rest of the system. He never needs to see test vectors.

On the other hand, testers are designed to handle device characterization and device screening based on precise timing measurement. Although hardware modelers can be used for basic functional testing, they do not support dc parametric measurements or high-precision timing characterization.

Figure 4 compares hardware modelers with in-circuit emulators. In-circuit emulators are used to debug system hardware and software after fabrication of a prototype system. Emulators are available only for microprocessors. For modern microprocessors, they are expensive and time-consuming to develop ($2 million to $3 million, and one to two years). Because they monitor and interpret microprocessor inputs and outputs, designing them requires detailed knowledge of the microprocessor's structure and operation. In-circuit emulators are confused by internal pipelining. They may not run at maximum device speeds. (An in-circuit emulator must run at the clock speed of the target system, whereas the clock speed of a reference element in a hardware modeler is independent of the system clock speed.) Because of these problems, the design engineer may find that the appropriate in-circuit emulator is not available or not reliable when the prototype system is being debugged. These problems are getting worse with time, owing to the increasing complexity and performance of microprocessors.

On the other hand, hardware modelers are generic. A user himself can prepare a hardware model for a new device cheaply and quickly without detailed knowledge of the device's operation (although he *Continued on page 98*)

What's waiting for you at the end of the rainbow?

CĀDENCE. For a full spectr

ECAD and SDA have joined forces to offer you an integrated family of IC design solutions from a single source – CADENCE.

Out of the merger of ECAD and SDA, a clear leader has emerged from the storm of IC CAD vendors. CADENCE. A company with the products, support and R&D resources to provide the integrated, high-performance design solutions you need today.

What you can expect from CADENCE.

...A merging of the best IC design technologies—SDA's Design Framework™ combined with high-performance tools, like ECAD's Dracula™ and Symbad.™ The combination increases your productivity and gets your designs to market faster than ever before.

...A uniting of two world-class software development groups to
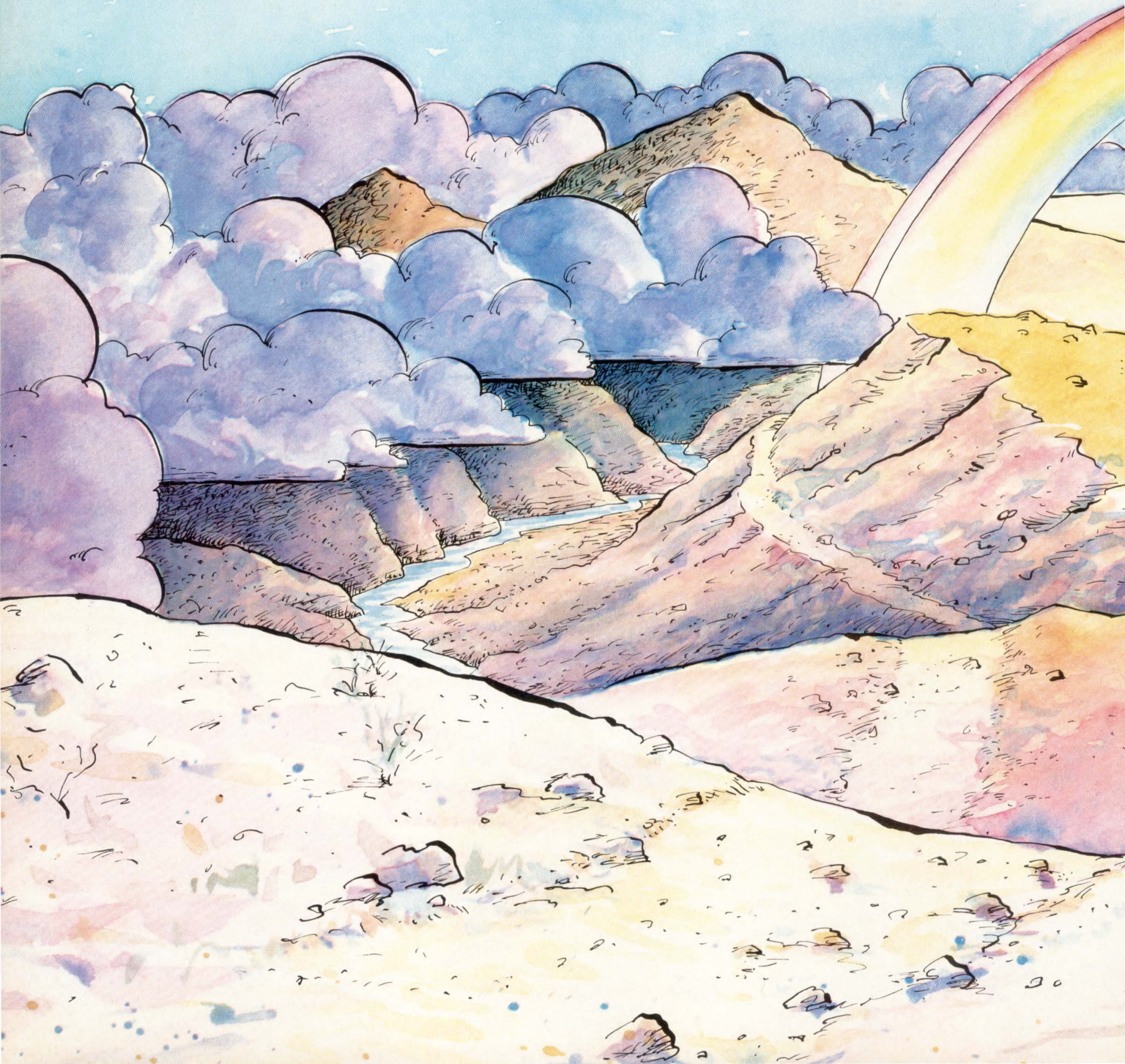
Survey of Semiconductor

FOUNDRIES

VLSI SYSTEMS DESIGN STAFF

*I*n conducting this year's survey of semiconductor foundries, in addition to our usual questions we asked the companies involved about the types of military qualification they were prepared to do. We thought the question pertinent because military programs are one of the prime users of foundries, particularly for application-specific ICs, and military contractors, with their specialized circuits and small production runs, depend on qualified foundries.

■ MILITARY SEEKS GENERIC QUALIFICATION OF FOUNDRIES ■

The answers indicate that although the majority of foundries do 883-C and 38510 testing, only three—General Electric, Marconi, Sprague, and Supertex (see the column "Military Certification and Qualification Services")—mention the new Defense Electronic Supply Center (DESC; Dayton, Ohio) certification, which covers the generic qualification of ASIC lines (Mattson, Howland, and Buchanan, 1987). In checking, however, we found that a growing number of ASIC-oriented foundries that do military circuits were aware of the generic approach to ASIC qualification and many are in the process of obtaining such certification.

Generic qualification is being promoted as a better way to qualify ASICs, like gate arrays, for the

military and perhaps the only way to provide qualification for next-generation VHSIC devices. As explained by Russ Pate, a staff engineer at Harris Semiconductor, one of the companies undergoing certification, the goal is to eliminate the waste in time and money of getting each and every ASIC variation on the military's QPL (Qualified Parts List).

Instead of relying on testing after the part is made, the design and manufacturing process is qualified beforehand on a generic basis, so that any part that comes from the ASIC line can be assumed to be qualified. According to Pate, the parts of an ASIC line that might be generically qualified would be:

■ The design system
■ Cell libraries
■ Wafer fabrication

■ Process flow control
■ Packaging

Pate pointed out that the /600, /605 and /606 "slash" sheets of Mil-Std-38510 have made a start toward generic qualification of gate array ASICs by treating gate arrays as standard families. This view makes the qualification process independent of the individual design personalizations. Instead, the qualification is based on standard evaluation circuits and other "observables" by which baselines can be established for the basic gate array.

The generic qualification approach will extend beyond gate arrays, Pate explains. Method 5010 of Mil-Std-883-C, which covers test procedures for complex monolithic integrated circuits addresses custom and cell-based designs.

Charles Messenger, generic qualification program manager at the U.S. Air Force's Rome Air Development Center (RADC; Rome, N.Y.), is guiding the generic qualification effort. One of his goals is to progressively broaden the scope of generic qualification so that it will be in place in time for VHSIC devices. The relationship between RADC and DESC is that RADC prepares the specification and DESC enforces it.
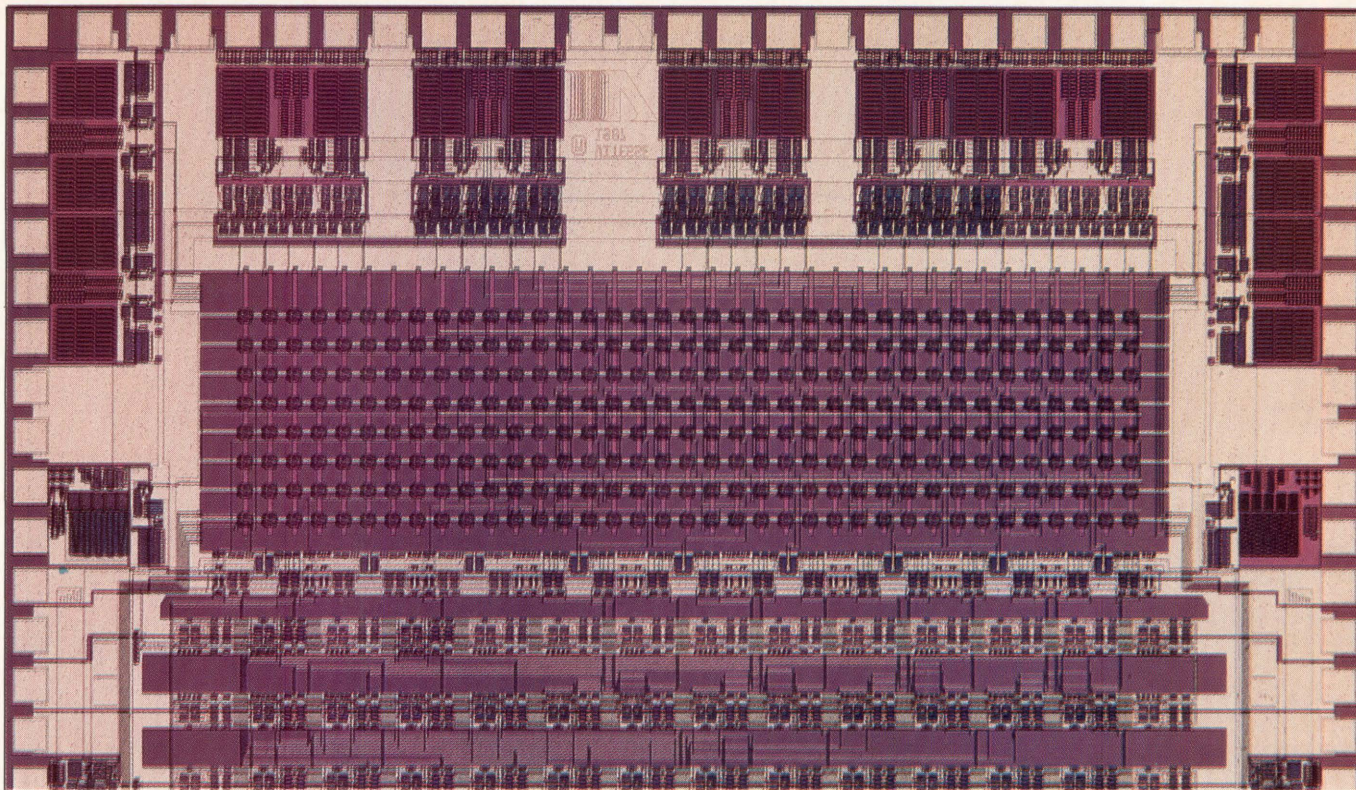
Daniel Fayette, an electrical engineer on the RADC generic qualification team, elaborates further on the team's goals: "We'd like to sell the semiconductor industry on generic qualification so that we'd be able to use the same lines that are mass-producing commercial circuits."

Fayette says that the generic qualification was patterned after a similar effort initiated with Mil-Std-1772 for hybrid circuits in 1984. But the semiconductor qualification is unique in that it particularly includes the CAD system that prepares the circuits for the foundry.

The basis for developing and maintaining quality will be an error-correcting feedback loop that goes through the CAD tools, the models in the CAD library, the actual foundry processing, and the tests, Fayette says. The standard evaluation circuit (SEC) will be periodically sent through the loop, and any differences between what the CAD tools and library models were supposed to produce and what the process did produce would be used for fine tuning, to bring the CAD steps and the process into agreement.

Just what the SEC vehicle should be is still the subject of much debate, Fayette says. It is one of the items being worked out with the help of the semiconductor companies serving as beta sites for the generic qualification. About the only consensus so far is that the SEC should be fairly complex and should include all the types of logic and memory circuits that could be found in end products produced by the line. For example, it ought to include sequential circuits that would provide baseline control of the propagation delays produced by the CAD and foundry combination.

Three companies—General Electric, AT&T, and Honeywell—are helping with the effort to establish generic qualification for VHSIC devices. They are part of the Industry Coordinated Working Group (ICWG), in which government and industry representatives are meeting to develop generic qualification criteria.

One of the first foundries to receive generic qualification of an ASIC line is United Technologies Microelectronics Center (which, like several companies, elected not be included in this year's directory). Ronald Hehr, manager of semicustom products at UTMC (Colorado Springs, Colo.), says that the generic qualification was for UTMC's 3.0-μm gate arrays and that similar generic qualification for its 1.5-μm gate array family is under way.

For the present, he says, the main benefit of qualification is the gain in credibility, but such qualification is worth the considerable expense, as it might be a necessary condition for serving military contractors in the future. However, judging from UTMC's reason for not wanting to be included in the directory this year—it said it didn't want to encourage inquiries that it might not be able to service—being generically qualified has not hurt the foundry.

As for gallium arsenide, full military qualification is being held off, according to Sven A. Roosild, assistant director of the electronic sciences division of the Defense Advanced Research Projects Agency (DARPA; Arlington, Va.), because GaAs fabrication has not yet settled down to standard processes. Nevertheless, note the number of GaAs foundries listed in the directory, which starts on the next page.

REFERENCE

MATTSON, S.; D. HOWLAND; AND W. BUCHANAN. DECEMBER 1987. "Qualifying the Military ASIC Vendor," *VLSI Systems Design*.

FOUNDRIES' CAD LIBRARIES ARE TO BE QUALIFIED, TOO

VENDOR AND CONTACT	TECH-NOLOGIES, FEATURE SIZE (METAL PITCH) (μm)	WAFER	REQUIRED PRO-DUCTION COMMIT-MENT	COSTS FOR FABRIC-ATION AND SERVICES	ACCEPT-ABLE FORMATS FOR DESIGN SUB-MISSION	MILITARY CERTIFI-CATION AND QUALIFI-CATION SERVICES	TURN-AROUND TIMES	SECOND SOURCES
ABB HAFO *11501 Rancho Bernardo Rd., San Diego, Calif. 92128* Kay Baird, Marketing Administrator, (619) 485-8200	Metal-gate CMOS, 5 (10) Si-gate CMOS, 3 (6) Si-gate CMOS/SOS, 3 (8), 2 (5) Si-gate CMOS, DLM, 2 (poly, 4.5; metal 1, 6; metal 2, 7)	4″	1 lot of 20 wafers mini-mum; produc-tion volume negotiable	$20k–$30k, including masks, processing, and prototype packaging	PG tape (Mann 3000 pref.), CIF, Calma	883 screening, 38510	PG tape to tested wafers: 6–8 weeks (add 2 weeks for packaged prototypes)	RCA
ADAMS RUSSELL ELECTRONICS CO. INC. *80 Cambridge St., Burlington, Mass. 01803* A.D. Barlas, Director of Operations, (617) 273-3333	GaAs, 0.5, 1 (4 min.)	3″	Not specified	$35k–$60k, including tooling	Calma tape, CIF	883 screening, 38510	10–12 weeks	None
ANADIGICS INC. *35 Technology Dr., Box 4915, Warren, N.J. 07060* Michael P. Gagnon, Director, Sales and Marketing, (201) 668-5000	GaAs, 0.5	3″	Not specified	$9.5k–$50k	PG, Calma tape; masks; CIF	Assembly testing, including 883 screening	3 weeks for mask set; 8 weeks for wafer fab	None
CALIFORNIA DEVICES INC. *500 Central Ave., P.O. Box 280, Louisville, Colo. 80027* Richard M. Morley, Manager of Foundry Business Unit, (303) 665-8111	HCMOS, 2, 3 (metal 1: 6 pref., 5.4 min.; metal 2: 7 pref., 6 min.) Planarization; nMOS, pMOS, custom, and unit step processing available	4″	No specific production requirement; minimum lot size, 25 wafers	$19–$28 per mask level per wafer. Prototype or design-check lots with quick turnaround, 30% premium	Calma stream at 1,600 bpi or GDS/Mann PG format at 800 bpi (pref.)	Paragraph 1.2.1 of Mil-Std-883-C (provisions for use of Mil-Std-883 in conjunction with compliant non-JAN devices). Level B and S screening available in wafer form and packaging.	SLM: 6 weeks std., 4.5 weeks prototype. DLM: 7 weeks std., 5 weeks prototype. Cycle time starts 2 days after receipt of PG tape	None (process com-patible with NCR, VLSI Tech-nology, Seiko-Epson)
CALIFORNIA MICRO DEVICES *2000 W. 14th St., Tempe, Ariz. 85281* Steve McGrady, Marketing Foundry Engineer, (602) 921-6527	CMOS, nMOS, bipolar, biMOS; 2–5	4″	Not specified	Not specified	PG, Calma tape; masks; CIF	883 screening	2–4 weeks upon receipt of masks and/or design submission	Mitel (CMOS)
CALOGIC CORP. *237 Whitney Pl., Fremont, Calif. 94539* Manny Del Arroz, President, (415) 656-2900	Metal-gate CMOS, linear bipolar, DMOS; 4 (9)	4″	200	CMOS: $150–$200. Bipolar: $180–$275	PG, Calma tape or masks	883 screening	6–8 weeks on produc-tion start; 6–12 weeks on qualifi-cation	None

VENDOR AND CONTACT	TECH-NOLOGIES, FEATURE SIZE (METAL PITCH) (μm)	WAFER	REQUIRED PRO-DUCTION COMMIT-MENT	COSTS FOR FABRIC-ATION AND SERVICES	ACCEPT-ABLE FORMATS FOR DESIGN SUB-MISSION	MILITARY CERTIFI-CATION AND QUALIFI-CATION SERVICES	TURN-AROUND TIMES	SECOND SOURCES
COMMODORE SEMI-CONDUCTOR GROUP *950 Rittenhouse Rd., Norristown, Pa. 19446* Ben Rappaport, Director of New Product Technology, (215) 666-2585	Si-gate nMOS, 5 (11), 3 (9), 2.5 (7.7) Si-gate CMOS, 3 (9), 2 (4.5) Si-gate CMOS, DLM, 2 (4.5, 5.5)	4", 5"	25 wafers minimum prototype run	Masks: $800–$1600 per level. Prototype: $190–$400 per wafer. Production: $165–$350 per wafer.	Masks, PG tape, Calma GDS II (pref.)	None	PG tape to masks: 1 week. Turnaround time from masks—CMOS: 4 weeks for prototypes, 6 weeks for production. DLM CMOS: 5 weeks for prototypes, 7 weeks for production	None
ECI SEMI-CONDUCTOR *975 Comstock St., Santa Clara, Calif. 95054* Georgene Bennett, Customer Service Manager, (408) 727-6562	Metal- and Si-gate CMOS, pMOS, nMOS; 3 (6) Bipolar (IC and discrete) JFET discrete High-voltage CMOS or DMOS ICs on DI wafers	4", 3"	No minimum annual purchase requirement; normal start increments of 24 wafer lots; special lot size and splits available	$3k–$20k per lot	Schematic diagram, Calma tape, PG tape, master plates, working plates	883-C–com-pliant pro-cessing	Typically 4 to 6 weeks; special fast-turnaround service available for engineering prototypes	None
EXAR INTEGRATED SYSTEMS INC. *2222 Qume Dr., San Jose, Calif. 95131* Surjit Nijjer, Marketing Manager, (408) 434-6400	Metal-gate CMOS, 4 Si-gate CMOS, 2 Bipolar (linear, I²L)	3", 4", 5"	Negotiable	Standard process: $4.2k for 10 wafers	Mask (pref.); database tape, PG tape, reticles	883-C–quali-fied	Masks to PCM-tested wafers: 6 weeks	Rohm
FORD MICRO-ELECTRONICS INC. *10340 State Highway 83, Colorado Springs, Colo. 80918* Charlotte Diener, Marketing Manager, (719) 528-7600 (800) 777-FORD	GaAs (MMIC and digital), 1.25 (6)	3"	5 wafers	$75k	Calma tape	None	13 weeks	Vitesse (E/D MESFET digital process)
GENERAL ELECTRIC MICROELECTRONICS CENTER *3026 Cornwallis Rd., P.O. Box 13049, Research Triangle Park, N.C. 27709* Donald B. Dickerman Jr., Manager, Marketing and Program Management, (919) 549-3100	Bulk CMOS, DLM, 1.25; bulk, base-line, and Megarad CMOS/SOS, DLM, 1.25; baseline and rad-hard	4"	12 wafers per run minimum	Depends on tests and packaging required	Schematics, netlist, PG tape, Calma Stream file, GDS II	DESC-certified to Mil-M-38510 for 1.25- and 2-μm bulk CMOS, Classes B and S screening	5 weeks for prototype from PG tape; 9 weeks for Class B production; 18 weeks for Class S production	Other GE locations

Nobody makes it denser
45,000 gates in a CMOS channel-type gate
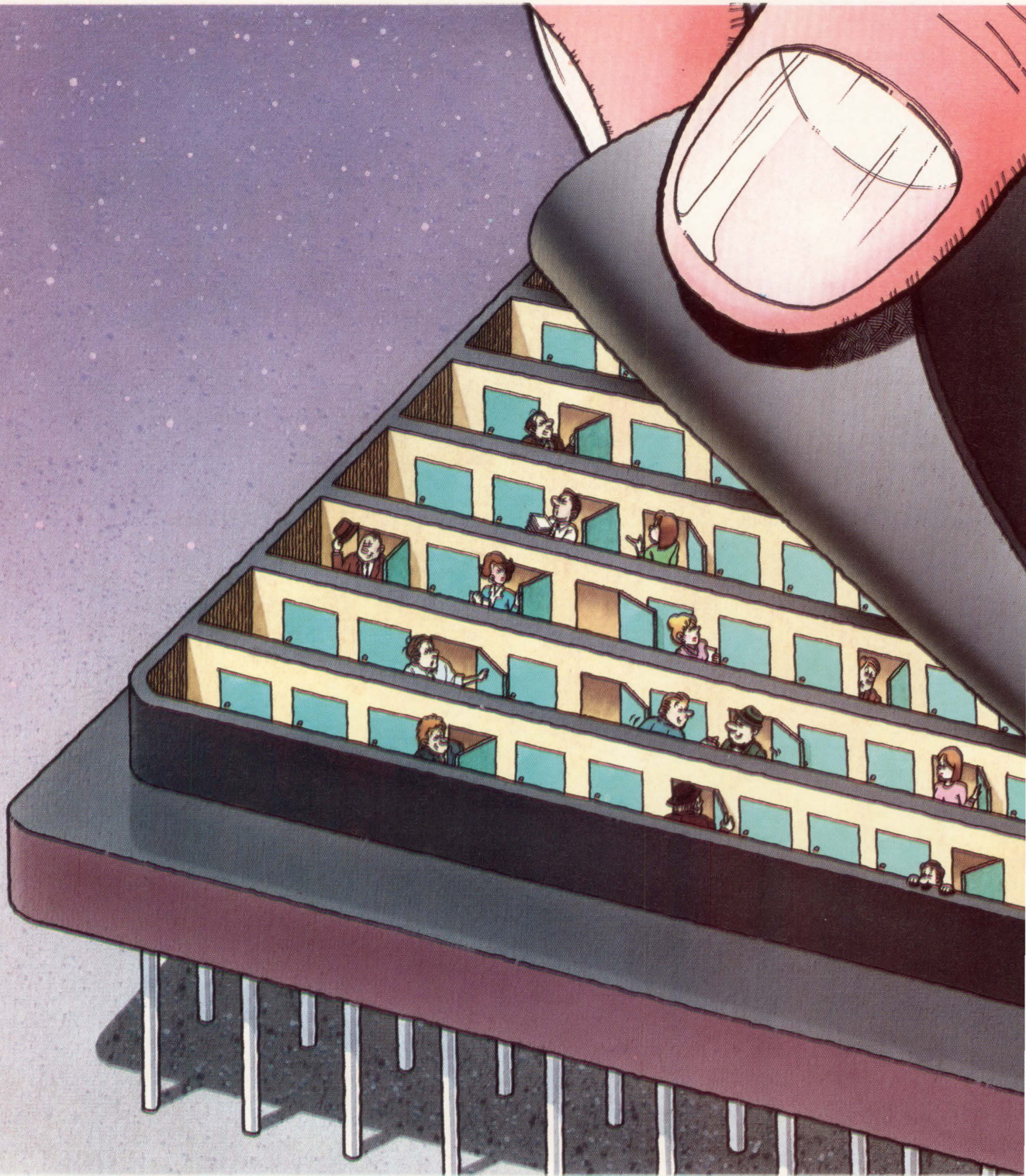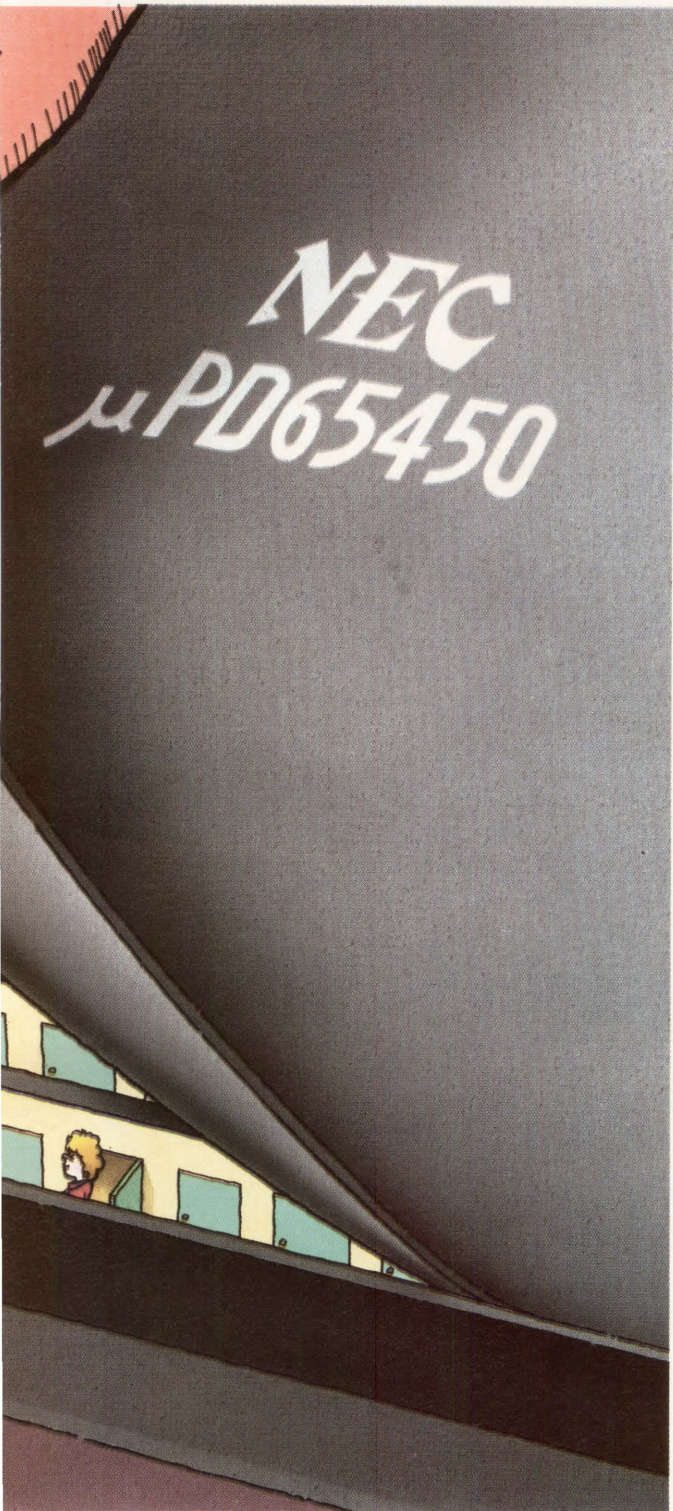
array

High integration and high speed—you get it all from NEC's new CMOS-5 and -5A channel-type gate array families. Gate counts start at 2000 and range all the way up to 45000, the world record for density in a channel-type CMOS device. Gate utilization is over 90% (μPD65450).

Speed rivals ECL performance. Internal gate delays are 1.0ns for a standard gate; 0.65ns for a power gate (F/O = 3, L = 3mm).

CMOS-5 and -5A families protect your system with high resistance to latchup—over 1A (typ). That's more than enough for security in most applications. Other features include:

☐ High drive capability: I_{OL} = 3 — 18mA.
☐ 5K-ohm pull-up resistor-incorporated buffers; ready for TTL bus lines.

NEC now offers you the broadest CMOS gate array lineup—six families, with 37 masters, including the world's densest channel-type gate array. For more information, call NEC today.

Family	Master	Gate Count	I/O Pads
CMOS-5	μPD65025	2016	71
	032	3366	81
	044	4440	112
	051	5292	120
	061	6348	132
	071	7500	144
	082	8748	152
	103	10800	164
	140	14256	188
	180	18144	224
	240	24000	256
CMOS-5A	μPD65300	30600	256
	450	45012	256

NEC

VENDOR AND CONTACT	TECH-NOLOGIES, FEATURE SIZE (METAL PITCH) (μm)	WAFER	REQUIRED PRO-DUCTION COMMIT-MENT	COSTS FOR FABRIC-ATION AND SERVICES	ACCEPT-ABLE FORMATS FOR DESIGN SUB-MISSION	MILITARY CERTIFI-CATION AND QUALIFI-CATION SERVICES	TURN-AROUND TIMES	SECOND SOURCES
GE SOLID STATE *724 Route 202, P.O. Box 591, Somerville, N.J. 08876* Bill Allen Manager, Military and Aerospace ASIC Marketing; Paul Sferrazza, Manager, Commercial ASIC Marketing; (201) 685-7460	Si-gate CMOS, 3 (9) High-voltage Si-G CMOS, 3 (9) Si-gate CMOS, DLM, 2 (6, 8), 3 (9, 11), 1.5 (3.5) Rad-hard Si-G CMOS/SOS, 3 (10), 2 (10), 1.5 (6, 8)	4", 5"	Negotiable	Depends on the application	Calma tape, PG tape, netlist, schematic	M38510 facilities, classified facilities, Mil-Std-883-C screening	Calma tape to design verification units: 6–8 weeks	LSI Logic (gate arrays), VLSI Technology (gate arrays, standard cells), Siemens and Toshiba (standard cells)
GIGABIT LOGIC INC., *1908 Oak Terrace Lane, Newbury Park, Calif. 91320* Tony Conoscenti, Product Marketing, (805) 499-0610	GaAs, 1 (4)	3"	4 wafers minimum	$5k for depletion-mode process; $7k for enhancement/depletion-mode process; both for 4-wafer quantity	Masks, Calma GDS II, CIF format tape	MIL-STD-883 screening	4 to 8 weeks for wafers from masks; 2 weeks for masks	Tachnonics (SC5000 cell library)
GOULD INC. *Semiconductor Division, 2300 Buckskin Rd., Pocatello, Idaho 83201* Mary Ann Gentry, Marketing Manager, (208) 233-4690	CMOS, 1.25	5"	Not specified	Not specified	Most formats are acceptable	883-C–compliant	Not specified	Not specified
HARRIS CORP. *Semiconductor Sector, P.O. Box 883, Melbourne, Fla. 32901* Russ M. Pate Staff Engineer, Semicustom Applications, (305) 729-5727	Metal- and Si-gate pMOS, 5 (10) Metal-gate CMOS, 5 (10) Si-gate CMOS, rad-hard, 1.2 (3.4), 2.5 (8), 2 (8), 3 (10), 1.5 (8) Bipolar (various analog-digital processes)	4", 5"	20 wafers minimum lot	$300–$1k per wafer	PG tape, Calma tape, masks, cell library netlist	38510/883 screening available; Classes B and S; JAN-qualified facility; secret clearence	Mask to prototypes: 5 weeks	Yes (not specified)
HARRIS MICROWAVE SEMICONDUCTOR *1530 McCarthy Blvd., Milpitas, Calif. 95035* James Klug, Director, Business and Program Development, (408) 433-2222	Depletion-mode MESFET, 1.0–0.5	2", 3"	Small-volume prototype runs accepted; 2 wafers minimum	$8k–$18k per wafer	Calma tape, Applicon, CIF	Military screening through Class S equivalent	Two wafers with via holes, scribing, separation, Visula in-spection: approx. 14 weeks	None

| VENDOR AND CONTACT	TECH-NOLOGIES, FEATURE SIZE (METAL PITCH) (μm)	WAFER	REQUIRED PRODUCTION COMMITMENT	COSTS FOR FABRICATION AND SERVICES	ACCEPTABLE FORMATS FOR DESIGN SUBMISSION	MILITARY CERTIFICATION AND QUALIFICATION SERVICES	TURN-AROUND TIMES	SECOND SOURCES
HOLT INTEGRATED CIRCUITS INC. *9351 Jeronimo Rd., Irvine, Calif. 92718* Roger Smith, Manager, Sales, (714) 859-8800	Metal-gate nMOS, pMOS; 5 (8) Si-gate nMOS, CMOS, DLP; 3 (8) Si-gate pMOS, 3 (8) Metal-gate CMOS, 5 (8)	4"	10 wafers minimum engineering run	Production: $115–$400 per wafer	Masks (pref.), PG tape	Mil-Std-883-C	Masks to PCM-tested wafers: 2–4 weeks, prototype; 4–6 weeks, production	None
HUGHES AIRCRAFT CO. *Semiconductor Products, 500 Superior Ave., Newport Beach, Calif. 92658* Mike Friedman, Applications Manager, (714) 759-2727	Metal-gate CMOS, 6.8 (12) Si-gate CMOS, 3 (6) Si-gate CMOS, DLM, 2 (6), 7	4"	Typically 1k–10k wafers annually	Mask charge: $1.8k–$2.8k per layer. Prototype run: $9k–$15k	GDS II tape	Full 883 and 38510	From Calma tape to prototypes: 8–12 weeks	None
INTERNATIONAL MICROELECTRONIC PRODUCTS INC. *2730 N. First St., San Jose, Calif. 95134* Tom Flageollet, Marketing Manager, (408) 434-1362	CMOS, 5 (10); 3 (8); 2 (6); 1.2 (4) nMOS, 5 (10); 3 (8)	4"	Consult company	Prototyping: $20k–$40k. Wafers: $300–$1k	Applicon, Calma tape, masks, CIF, PG tape	None	Database tape to prototypes: 8–10 weeks	National Semiconductor, VLSI Technology
ITT SEMI-CONDUCTORS *15 Progress Dr., Shelton, Conn. 06484* Joseph J. Fabula, Manager, Technology, (203) 929-9790	Si-gate nMOS, 5 Si-gate HMOS, 2.4, 2 P-well Si-gate CMOS, 5 P-well Si-gate CMOS, DLP, 4 P-well Si-gate CMOS, DLM, 3 N-well Si-gate CMOS, DLM, 2	4"	10 wafers minimum lot size; an on-going business relationship is preferred	$200–$1000 per wafer; $35k typical prototype run, including up to 10 mask levels	ETEC drive tape biased to ITTSC requirements (pref.)	None	Prototype wafers: 10 weeks after receipt of order and mask tapes; faster turnaround available	None
LANSDALE SEMI-CONDUCTOR INC. *3600 W. Osborn, Phoenix, Ariz. 85019* Terry Green, Advertising/ Marketing, (602) 269-6262	Bipolar— analog, TTL, DTL, RTL, ECL (DI, gold doping)	3"	20 wafers minimum	Evaluation lot $10k; typical (20 wafer); production negotiable	Mask (pref.), PG, Calma	883-C	From PG tape to package: 12 weeks; from mask: 4–6 weeks	None
LSI LOGIC CORP. *1551 McCarthy Blvd., Milpitas, Calif. 95035* Bill Wirth, Director of Components Technology Marketing, (408) 433-8000	CMOS, 1.5, 2	5", 6"	25 wafers (negotiable)	Product- and process-dependent	PG, Calma tape, masks	Full military capability up to level S	Prototypes: 4 weeks from masks; production: 12 weeks	Process-dependent

VENDOR AND CONTACT	TECH-NOLOGIES: FEATURE SIZE (METAL PITCH) (μm)	WAFER	REQUIRED PRO-DUCTION COMMIT-MENT	COSTS FOR FABRIC-ATION AND SERVICES	ACCEPT-ABLE FORMATS FOR DESIGN SUB-MISSION	MILITARY CERTIFI-CATION AND QUALIFI-CATION SERVICES	TURN-AROUND TIMES	SECOND SOURCES
MARCONI ELECTRONIC DEVICES INC. *45 Davids Dr., Hauppauge, N.Y. 11788* Dale Wilson, Vice President, Sales and Marketing, (516) 231-7710	Si-gate CMOS, 6 (12), 5 (10), 3 (6) Si-gate CMOS, DLM, DLP— analog, 3 (6) Si-gate CMOS, DLM, 3 (8), 2 (6) Si-gate CMOS/SOS, SLM, SLP, 5 (10), 2.5 (6) Si-gate CMOS/SOS, DLM, 1.5 (5, 6); 2.5 (6)	4″	1 lot of 25 wafers minimum	From $275 per wafer; additional charges for mask manufacture, testing, etc.	All forms acceptable	Mil-Std-883-C; DESC certification for Mil-M-38510 QPL being sought; M.O.D./ NATO– AQAP1	Masks to tested wafers: 6–10 weeks. From PG tape: 8–13 weeks plus 2–4 weeks for packaged, tested devices	None
MCE SEMI-CONDUCTOR INC. *1111 Fairfield Dr., West Palm Beach, Fla. 33407* James P. Skoutas, (407) 845-2837	Metal-gate CMOS, 5 Bipolar, analog (20, 40, 80 V), Smart power (20, 40 V)	4″	$30k preferred but will negotiate	$9k–$150k	GDS II; PG tape (Mann 3000)	None	PG tape to packaged prototypes: custom— 10–30 weeks; semi-custom— 4–10 weeks	Yes (not specified)
MICREL INC. *560 Oakmead Pkwy., Sunnyvale, Calif. 94086* James Crow, Silicon Foundry Sales Manager, (408) 245-2500	Si-gate nMOS, pMOS, CMOS (5–30 V), 3–12 (7) Metal-gate nMOS, pMOS, CMOS (5–35 V); bipolar 10–40 V); 5–12	4″	No minimum annual volume required; 25 wafers minimum	Contact company	All database, PG tape formats; P and E projection and Cannon proximity mask	883 screening, 38510	Database to masks: 1–2 weeks. Mask to wafers out of fab: 1–4 weeks. Wafer sort to final package: 1–6 weeks	None
MICROCHIP TECHNOLOGY INC. *(formerly General Instrument Microelectronics)* *2355 W. Chandler Blvd., Chandler, Ariz. 85224* R. Malboeuf, Product Marketing Manager, (602) 963-7373	Si-gate CMOS, 1.5, 2.5 Si-gate nMOS, 2.5	4″, 5″	Annual dollar volume: $500k	$40k–$75k (including tooling)	GDS II (pref.), PG tape	None	8–14 weeks	None
MICRO-REL *2343 W. 10th Place, Tempe, Ariz. 85281* Dave Rigg, Marketing Manager, (602) 921-6411	CMOS, 3 (7), 5 (10) Bipolar (20, 40, 100 V), 8 (16)	4″	50 production wafers per month minimum	$7k, engineering 1st lot. Bipolar: $200–$300 per wafer; CMOS: $250–$400 per wafer	PG, Calma tape, masks (conditional)	1772 certifica-tion, 883-C, 38510	From masks to wafers (1 lot): CMOS—6 weeks; bipolar—8 weeks	None

Most simulation systems tell you as much about your ASIC design as this tells you about the Statue of Liberty.
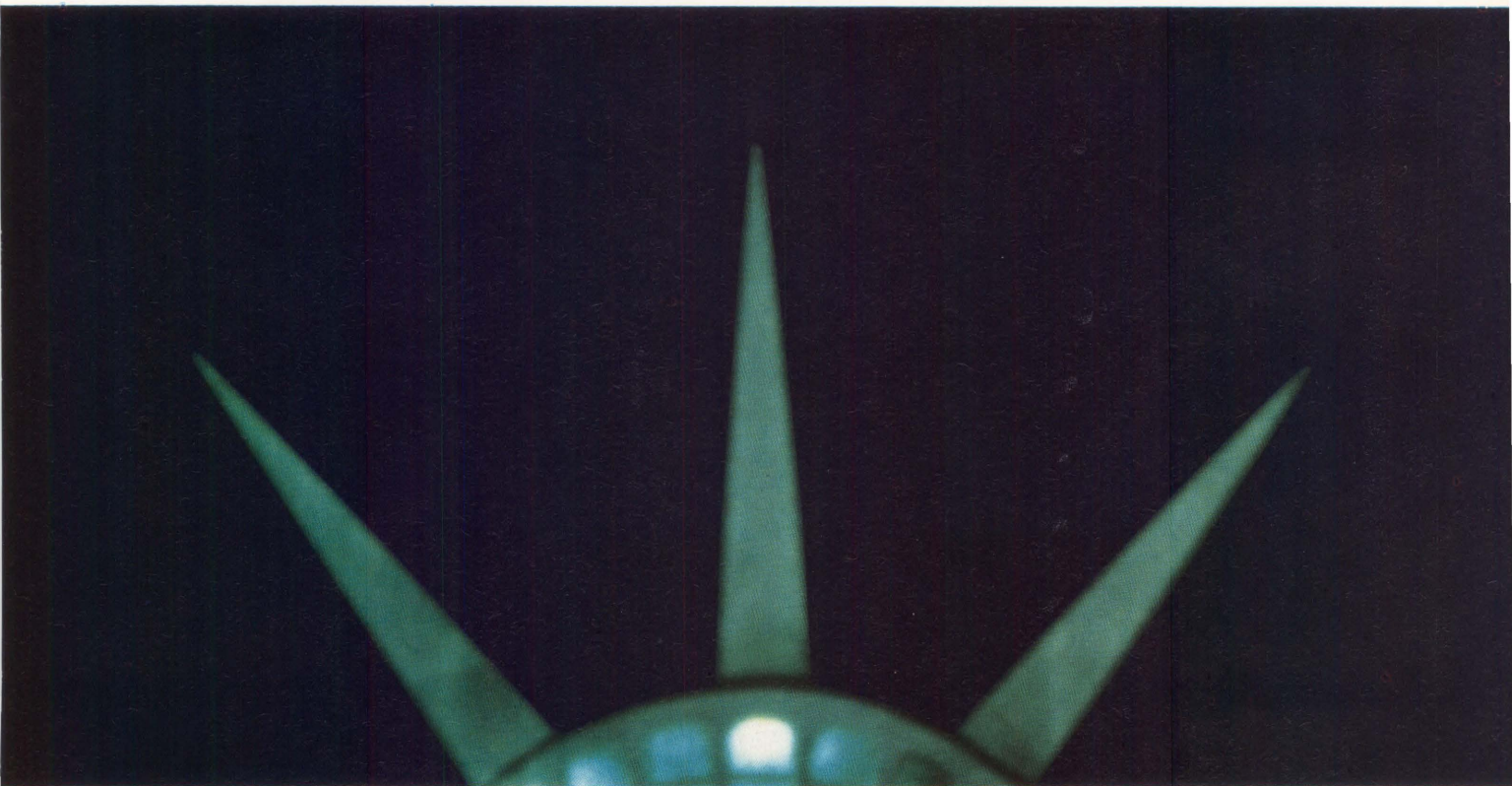
Get the full picture of your ASIC design's performance. Get the IKOS Simulation System. IKOS gives you more simulation information in less time than any other system. Simulate 15,000 gates with 34,000 vectors in 34 seconds. Or 8,500 gates with 5,100 vectors in 4 seconds. Complete with detailed timing information!

IKOS is a comprehensive ASIC simulation system incorporating a combination of sophisticated software and hardware components that run on the Sun, Apollo, or IBM PC/AT platforms. It provides you with a streamlined, high-speed approach to stimulus entry, logic and fault simulation, analysis, and ASIC library support. And, it integrates easily into your existing design environment.

IKOS gives you a tremendous performance advantage.

You enter stimulus the way you visualize it—by drawing waveforms. Powerful signal modeling features allow you to easily and realistically simulate asynchronous signals and interfaces.

With IKOS' real time logic analyzer you can run simulations interactively, pinpointing signals and conditions of interest. Gone are the tedious hours spent pouring over long output listings.

There's no limit to the number of test cases you can generate and analyze with IKOS. You'll be able to simulate seconds, *even minutes* of real time system operation. And, gain the confidence that your ASIC design works as you intended it to.
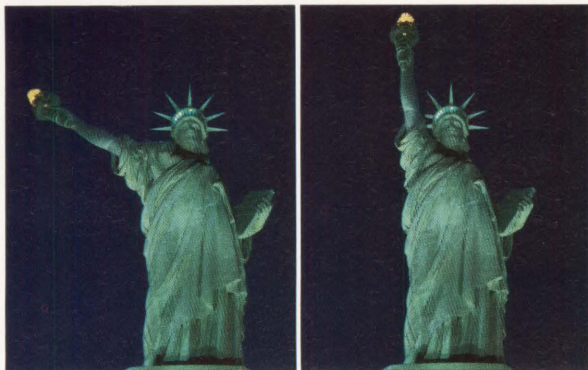
Without the IKOS Simulation System, most ASIC design mistakes show up after production.

IKOS comes complete with all the simulation tools you need—at a low price. Use IKOS to get your ASIC design right *before* committing to manufacturing. It's almost as easy as snapping a picture.

Give us a call.

IKOS

Now available on Sun!
408 • 245 • 1900
145 N. Wolfe Road · Sunnyvale, CA 94086

Sun is a trademark of Sun Microsystems. Apollo is a trademark of Apollo Computer, Inc. IBM PC/AT is a trademark of International Business Machines Corporation.

CIRCLE NUMBER 13

VENDOR AND CONTACT	TECHNOLOGIES, FEATURE SIZE (METAL PITCH) (μm)	WAFER	REQUIRED PRODUCTION COMMITMENT	COSTS FOR FABRICATION AND SERVICES	ACCEPTABLE FORMATS FOR DESIGN SUBMISSION	MILITARY CERTIFICATION AND QUALIFICATION SERVICES	TURNAROUND TIMES	SECOND SOURCES
MICROWAVE SEMICONDUCTOR CORP. *100 School House Rd., Somerset, N.J. 08873* Curtis S. Kraut, Product Marketing, Bruce C. Hoffamn, Product Marketing Manager, (201) 563-6300	GaAs D-MESFET, (1A/1D), 1	3″	2 wafers minimum	$42k: 1A process with thin-film resistors; $36k: 1D process, includes masks and 2 wafers	GDS II tape (pref.), masks, schematics	Full complement of rf and dc packaging and testing options	13 weeks from Calma tape to PCM-qualified wafers	TriQuint Semiconductor
	GaAs MESFET, epitaxy (MMICs), 0.5, 1	3″	2 wafers minimum	$42k: 0.5-μm gates; $35k: 1-μm gates; prices include 2 wafers, masks, E-beam—written gates and via hole etching and plating	GDS II (pref.), masks, schematics	Full complement of assembly and screening options	13 weeks from submitted GDS II tape, includes masks and PCM-qualified diced wafers	None
MIETEC N.V. *Westerring 15, 9700 Oudenaarde, Belgium* Eric Schutz, Business Development Director, (011) 32-55-33-22-11	CMOS, 1.5 (5), 2.4 (5.6), 3 (7) BiMOS, 8 (16) SbiMOS, 3 (8) nMOS, 3 (7)	4″	To be negotiated	Contact company	PG tape, GDS II netlist	883-C screening, ISO 9000, CECC 90000, NFC 96883	Quotation requests welcomed	IST, Sprague
MITEL CORP. *360 Legget Dr., P.O. BOX 13320, Kanata, Ontario Canada K2K 1X5* Luc Gagnon, Product Manager Marketing, Custom Product, (613) 592-5630	Si-gate CMOS analog, 5 (10), 4 (8), 3 (7), 2 (4.5)	4″	Minimum batch size is 24 wafers; no production commitment required	$250–$600 per wafer	PG tape: format Mann 3600, Mann 3000, E Mask 2000. Calma tape: GDS II	883-C Class B, method 5004, screening; method 5005 qualification and quality conformance	7 weeks	California Micro Devices, Gould
NATIONAL SEMICONDUCTOR CORP. *2900 Semiconductor Dr., Santa Clara, Calif. 95051* Chuck Botchek, Marketing Manager, (408) 721-5000	Si-gate CMOS, 2 (4.75, 6.25)	6″	$250k per year and $750k for the life of the product	$20k–$35k	Calma database tape, GDS II format	883 probing, packaging, packaged-part testing, burn-in	Database tape to packaged prototype: 10–14 weeks	International Microelectronic Products
NCR MICROELECTRONICS *2001 Danfield Ct., Fort Collins, Colo. 80525* Al Brown, Director, COT/Foundry Marketing, (303) 223-5100	CMOS, DLM (analog option), 1.5 (4.2, 4.7) CMOS, DLM (analog and SLM option), 2 (4.95, 5.5) CMOS, SLM (analog and DLM option), 3 (7.5, 10) nMOS, 3, 4	4″	25-wafer lot minimum	NRE pricing: $30k, includes photomasks. Prototype costs: $8k–$10k (15 wafers or 20 finished units)	GDS II, CIF, MEBES	DESC drawings, 883-C screening	Mask-profiled wafers: 4–5 weeks; mask to probed wafer or dice: 4–6 weeks; mask to tested units: 6–9 weeks; PG tape to photomasks: 1 week	None

VENDOR AND CONTACT	TECH-NOLOGIES, FEATURE SIZE (METAL PITCH) (μm)	WAFER	REQUIRED PRODUCTION COMMITMENT	COSTS FOR FABRICATION AND SERVICES	ACCEPTABLE FORMATS FOR DESIGN SUBMISSION	MILITARY CERTIFICATION AND QUALIFICATION SERVICES	TURN-AROUND TIMES	SECOND SOURCES
OKI SEMI-CONDUCTOR *650 North Mary Ave., Sunnyvale, Calif. 94086* Clifford Vaughan, Marketing Manager, (408) 720-1900	Al-gate and Si-gate CMOS, 1.2–7 (4.5)	4", 5", 6"	Over 1k wafers per year or over 250k assemblies per year	$25k plus mask tooling for first qualification lot of 1.5-μm wafers; projection masks at $3k each; stepper masks at $6k each	PG tape to logic diagram	None	10 weeks from film sign-off	None
ORBIT SEMI-CONDUCTOR INC. *1230 Bordeaux Dr., Sunnyvale, Calif. 94089* Gary P. Kennedy, President, (408) 744-1800	Si-G CMOS— SLP/SLM, SLP/DLM, DLP/SLM, DLP/DLM; 1.5 (4, 4) CCD Older technologies, (e.g., nMOS)	4"	None	Prototypes: $12k to $19.5k	GDS II, MEBES, CIF	Appendix A, Mil-Std-38510	Tape to chips: (SLP, SLM—4 weeks guaranteed; DLP, DLM—5 weeks guaranteed	None
PLESSEY SEMI-CONDUCTORS *1500 Green Hills Rd., Scotts Valley, Calif. 95066* John Kitzrow, U.S National Sales Manager, (408) 438-2900	Bipolar, 3, 1.5 CMOS, 4, 2, 1.5 nMOS (2 Si), 6	4", 6"	Minimum $100k p.a.	Subject to negotiation	GDS II Calma or PG	DESC drawings, 883 screening, 38510	Typically 6 weeks from masks	Internal
POLYCORE ELECTRONICS INC. *1107 Tourmaline Dr., Newbury Park, Calif. 91320* S.K. Leong, Vice President, (805) 499-6777	Bipolar (40, 12 V), DLM, 5	3", 4"	One lot minimum, 24 wafers	About $5k per engineering run; about $200 per wafer in production	PG tape, GDS II	883 and 38510	From masks: 4–6 weeks	None
RAYTHEON CO. *350 Ellis St., Mountain View, Calif. 94043* Peter Nghiem, (for ECL products), Senior Product Marketing Engineer, Pete Goshgarian, Marketing Manager, CMOS Products, (415) 966-7628	ECL, 1.4, (metal 1 and 2: 4)	4"	Negotiable	About $55k, including masks making and wafer production	GDS II (pref.), PG, masks, CIF	Mil-Std-883, Revision B, method 5004, Class B screening	8 weeks—2 weeks for mask making, 4 weeks for wafer production, 2 weeks for assembly and tests	Bipolar Integrated Technology (Bit1)
	CMOS, 1.25 (metal 1: 4; metal 2: 6)	5"	Negotiable; 20 wafers minimum	Masks: $2k. Processing: $45k	PG/Calma	883 Revision C	8–12 weeks	None

9 REASONS TO CHOOSE INTEL FOR YOUR EPLD DESIGN.

INTEL 5AC312 ARCHITECTURE

PROGRAMMABLE INPUTS (8)   LOGIC ARRAY   I/O MACROCELLS (12)

⑨ "SUM OF PRODUCTS" LOGIC ON ALL REGISTER AND OUTPUT CONTROLS.

⑧ ALLOCATION OF LOGIC ALLOWS USE OF 0-16 P-TERMS PER MACROCELL.

⑦ PROGRAMMABLE "D" OR "T" FLIP-FLOP.

⑥ MORE I/O PER DEVICE.

LATCH/REGISTER

PUT

CLK

MACROCELL REGISTER

I/O

① PROGRAMMABLE LATCH OR REGISTER INPUT OPTION.

② INPUT LATCH/REGISTER DRIVEN FROM PROGRAMMABLE SYNCHRONOUS OR ASYNCHRONOUS CLOCK.

③ PROGRAMMABLE ASYNCHRONOUS PRESET OPTION

④ PROGRAMMABLE SYNCHRONOUS OR ASYNCHRONOUS CLOCK ON OUTPUT REGISTERS.

⑤ DUAL MACROCELL FEEDBACK SO THAT REGISTER CAN BE BURIED WHILE PIN IS UTILIZED.

CLK

If you're looking for architecturally advanced EPLDs, look no further than Intel's innovative product family.

Start with our 5AC312. Based on Intel's 1.0-micron CHMOS* EPROM technology, this high-performance EPLD clips along at 50MHz with a very low power consumption of only 50mA, 150μA while idling in low power mode. And its enhanced programmable inputs can be configured as latches, allowing the 5AC312 to latch and hold data—a must for complex microprocessor-

AND 12 MORE.

and microcontroller-based systems. Plus you'll save valuable board real estate and improve overall system performance.

In addition, the 5AC312 has an exclusive programmable product term allocation scheme. It improves device utilization up to 33% and boosts performance by allowing implementation of applications requiring complex logic functions.

But we didn't stop there. These 5AC312 features are being incorporated into future EPLDs with even higher integration. So your upgrade path is assured.

But, architectural innovation is not the only reason to design with Intel's highly advanced EPLDs.

Even though these EPLDs offer more than six times the logic density per chip than other PLDs, we've dramatically simplified your design process. With our inexpensive and powerful PC-based development tool, iPLDS II, you can use your PC/XT,* PC/AT,* PS/2* or fully compatible system to turn EPLD design concepts into working silicon quickly, automatically and efficiently.

Of course, we provide extensive technical documentation to help you in system design. And, to make it even easier, we've taken service one step further. To ensure that the design process is as smooth as possible, users have access to Intel's toll-free EPLD "Logical Hotline" and a 24-hour electronic bulletin board for any technical assistance they might need. It saves hours of valuable time.

Best of all, the entire Intel EPLD family and all the development support you need are available now.

So call (800) 548-4725 and ask for Literature Dept. #W457. We'll tell you everything you need to know about the most technologically innovative EPLDs around.

Intel's 5AC312. And family.

intel®

VENDOR AND CONTACT	TECHNOLOGIES, FEATURE SIZE (METAL PITCH) (μm)	WAFER	REQUIRED PRODUCTION COMMITMENT	COSTS FOR FABRICATION AND SERVICES	ACCEPTABLE FORMATS FOR DESIGN SUBMISSION	MILITARY CERTIFICATION AND QUALIFICATION SERVICES	TURNAROUND TIMES	SECOND SOURCES
RICOH CORP. *2071 Concourse Dr., San Jose, Calif. 95131* Alan Sue, Manager, Key Account, (408) 434-6700	CMOS, 2, 1.5 nMOS, 2	4", 6"	50 wafers per month	Contact company	Calma, masks (pref.), CIF	None	6–8 weeks after receipt of masks	None
SIERRA SEMICONDUCTOR *2075 N. Capitol Ave., San Jose, Calif. 95132* Don MacLennan, Director, Custom Product Marketing, (408) 263-9300	CMOS, 2 (5, 7); 1.5 (4, 5.5)	5", 6"	None	Function of volume	GDS II, CIF	None	9 weeks, tape to parts	VLSI Technology
SIGNETICS CORP. *811 E. Arques Ave., P.O. Box 3409, MS 41, Sunnyvale, Calif. 94088* Don Schare, Marketing Manager, (408) 991-5436	CMOS, SLM, 3 CMOS, DLM, 2	4"	Volume production	NRE and production pricing quoted after review of circuit	GDS II	Full service through final test	6 to 14 weeks from availability of tooling	Texas Instruments (2- and 3-μm SystemCell designs); European fab available through Philips
SILICON SYSTEMS INC. *14351 Myford Rd., Tustin, Calif. 92680* Peter Putnam, Foundry Section Manager, (714) 731-7110	CMOS—DLP, SLM, 12-V analog/digital: 3.6 (6.4) Bipolar, DLM, 2.5 × 4 μm poly emitter, (metal 1: 9; metal 2: 14 (nitride capacitor, poly resistor)	4"	Contact company	Contact company	Calma, CIF, PG	Contact company	Contact company	Contact company
S-MOS SYSTEMS INC. (SEIKO-EPSON) *2460 N 1st St., San Jose, Calif. 95131* Coleen Muhal, Senior Product Marketing Engineer, Foundry Services, (408) 922-0200	Si-gate CMOS, 1.2, 1.5, 1.8, 2.5, 3	4", 5", 6"	100 wafers per month minimum	Prototype (25 wafers min. per lot): $750 (1.5, 1.8 μm, 5"), $2k (1.2 μm, 6"); Production (50 wafers min. per lot): $500–$600 (1.5, 1.8 μm, 5"), $1.4k–$1.8k (1.2 μm, 6")	PG, Calma, MEBES, masks	883 negotiable	MEBES/PG tape to wafers: 6–7 weeks. Calma to wafers: 9–10 weeks. Masks to wafers: 5–6 weeks	None
SPRAGUE SEMICONDUCTOR GROUP *115 N.E. Cutoff, Worcester, Mass. 01613* Allan Ledoux, Product Marketing Manager, (617) 853-5000	Metal-gate CMOS, metal-gate nMOS, pMOS; 3 (6, 2) Si-gate CMOS, Si-gate nMOS, pMOS, bipolar, biMOS; 6 (6, 8)	4"	$250k per year	$200–$1k per wafer	Calma or Applicon data tape, PG Mann 3000 or 3600, masks	DESC certification, DESC drawings, 883 screening, 38510	Mask making: 2 weeks. Wafer fab: 6–8 weeks	Yes (contact marketing)

VENDOR AND CONTACT	TECH-NOLOGIES: FEATURE SIZE (METAL PITCH) (μm)	WAFER	REQUIRED PRO-DUCTION COMMIT-MENT	COSTS FOR FABRIC-ATION AND SERVICES	ACCEPT-ABLE FORMATS FOR DESIGN SUB-MISSION	MILITARY CERTIFI-CATION AND QUALIFI-CATION SERVICES	TURN-AROUND TIMES	SECOND SOURCES
STANDARD MICROSYSTEMS CORP. *35 Marcus Blvd., Hauppauge, N.Y. 11787* Linda Abbattista Smith, Administration Manager, Custom Products Marketing, (516) 273-3100	Si-gate nMOS, 3 (6), 4 (8), 5 (10) Si-gate CMOS, 3 (6)	4″	3 lots minimum (20 wafers per lot)	$5k for a full lot; $10k for projection alignment quality masks (from Calma database)	Calma tape (pref.), masks	None	Calma to PG tape: 1 week; to masks, add 2–4 weeks; to probed wafers, add 2–4 weeks; to packaged devices, add 1–5 weeks	NCR
SUPERTEX INC. *1225 Bordeaux Dr., P.O. Box 3607, Sunnyvale, Calif. 94088* James Beaton, Product Marketing Engineer, (408) 744-0100	Metal-gate pMOS, 5 (10) Metal-gate CMOS, 5 (10) CMOS/DMOS (high-voltage)	4″	Engineering run (24 wafers)	CMOS and pMOS: $150–$360 per wafer. High-voltage CMOS: $200–$450 per wafer	Masks for pMOS and CMOS, GDS II for high-voltage CMOS	DESC qualified by 9/30/88	Masks to wafers: 4–10 weeks for pMOS and CMOS, 6–12 weeks for high-voltage CMOS	None
TACHONICS CORP. *107 Morgan Lane, Plainsboro, N.J. 08536* Michael Zyla, Sales Manager, (609) 275-2504	GaAS (MMIC, analog, digital), 1, 0.5 (metal pitch varies with feature size)	3″	Wafer lot size: 5, 10, 25 (normal). Annual volume: over 100k devices; can mix devices on wafer	$4.2k– $11.5k	Digital: Mentor and VLSI Technology CAD. Analog: Calma GDS II. Mixed analog/digital: consult company	Parts tested to 883-C Level B	SC5000 ASICs: 11 to 18 weeks	GigaBit Logic (SC5000 cell library)
TAIWAN SEMI-CONDUCTOR MANU-FACTURING CO. *3150 Almaden Expwy., Ste. 111, San Jose, Calif. 95118* Steve Pletcher, Vice President of Marketing, (408) 978-1322	CMOS—SLM and DLM, n-well, twin-tub: 2, 1.5, 1.2	6″	As negotiated	As negotiated	Calma, MEBES, CIF tape format	883, 38510 quality and reliability compliance	3 days per mask layer from complete mask set	None
TEKTRONIX INTEGRATED CIRCUITS OPERATION *P.O. Box 14928, Portland, Ore. 97214* Customer Inquiries, (800) 835-9433 ext. 100	Bipolar, 1.25 (4)	4″	Not specified	$30k–$60k	Calma tape and Quickic tape	None	6 weeks	None

VENDOR AND CONTACT	TECHNOLOGIES, FEATURE SIZE (METAL PITCH) (μm)	WAFER	REQUIRED PRODUCTION COMMITMENT	COSTS FOR FABRICATION AND SERVICES	ACCEPTABLE FORMATS FOR DESIGN SUBMISSION	MILITARY CERTIFICATION AND QUALIFICATION SERVICES	TURNAROUND TIMES	SECOND SOURCES
TOSHIBA AMERICA INC. *2692 Dow Ave., Tustin, Calif. 92680* Jerry Goetsch, Marketing Manager, (408) 733-3223	Si-gate nMOS, 1.5 (3.5, 5) Si-gate CMOS, 1.5 (3.5, 5)	5″	5 engineering wafers minimum	$5k–$10k engineering run	Calma tape, masks (pref.)	None	Mask to PCM-verified wafer: 6 weeks	None
TRIQUINT SEMICONDUCTOR INC. *Group 700, P.O. BOX 4935, Beaverton, Ore. 97076* Louis Pengue, Product Marketing Manager, (503) 644-3535	GaAs depletion-mode MESFET: digital, 1 (7); analog, 0.5, 1 (7) GaAs E/D MESFET, digital and analog, 1 (5)	4″, 3″	2 wafers minimum	$34k–$56k	GDS II, MEBES, masks	Limited 883 screening	7–12 weeks	Microwave Semiconductor Corp. (mask-compatible depletion-mode MESFET process)
UNITED MICROELECTRONICS CORP. *3350 Scott Blvd., Santa Clara, Calif. 95054* Richard McMillan, Director of Foundry Marketing and Sales, (408) 727-9239	Si-gate CMOS, 1.5 (4), 2 (5), 3 (6), 5 (10) Si-gate nMOS, 3 (7), 5 (10) Metal-gate CMOS, 6 (12) —All processes: DLM, DLP, poly capacitor	4″	24 wafers minimum*	Prototype run: $3k–$8k (no NRE); per-wafer charges only	Masks (pref.), DB tape, PG tape	Dynamic burn-in, method 2010 visual (std.)	Masks to finished wafers: 5–7 weeks. Quick turnaround: 3–4 weeks	None
UNIVERSAL SEMICONDUCTOR INC. *1925 Zanker Rd., San Jose, Calif. 95112* Mike Wilson, Director Marketing/Sales, (408) 436-1906	CMOS (5–15 V), nMOS; 2, 3, 4, 5 (10)	4″	25 wafers minimum	$185–$400	PG, Calma, Applicon, CIF	883 screening	4 weeks for electrically tested and good wafers	Siliconix, Silicon Systems
VITESSE SEMICONDUCTOR CORP. *741 Calle Plano, Camarillo, Calif. 93010* Ray Milano, Foundry Engineering Manager, (805) 388-3700	GaAs E/D MESFET, 1 min. (4)	3″	5 wafers minimum	$33k per mask set; $6k per wafer minimum order	GDS II, CIF	883/38510	4 weeks for processed wafers; 7 weeks, including package and tests	Ford Microelectronics
VLSI TECHNOLOGY INC. *1109 McKay Dr., San Jose, Calif. 95131* Deborah Harvey, Marketing Manager, Foundry Products, (408)434-3000	CMOS, 3 (7, 10), 2 (5, 7.5), 1.5 (4, 5.6), 1 (3.9, 3.9) HMOS, 4.5 (10); 3 (7); 2 (5, 4)	5″, 6″	Negotiable	Prototypes: $30k–$80k	CIF (pref.), GDS II, masks	Source control drawing for CMOS 3- and 2-μm and HMOS 3-μm	Prototype wafers or untested units: 4 weeks. Fully tested units: 5 weeks	GE Solid State, Rockwell

The state-of-the-art in gate array design is a Silvar-Lisco product. Again!

Introducing AVANT GARDS.™ For VLSI sea-of-gates arrays. With 150,000 gates on a chip. And more to come.

Until now, the state-of-the-art in gate array design systems was Silvar-Lisco's GARDS.™ But not anymore.

AVANT GARDS,™ Advanced Gate Array Design System can place and route over 150,000 gates on a single chip using sea-of-gates and three-layer designs. With gate utilization as high as 90%. Adhering to your timing specifications. In less than a week. And supported by the most popular foundries.

That's the *new* state-of-the-art in gate array design. And it's also a Silvar-Lisco product.

But AVANT GARDS™ isn't alone. Silvar-Lisco's cell-based and full custom chip layout software are state-of-the-art as well. And they get better all the time because we improve them all the time. All of them fully integrated with Silvar-Lisco's full line of CAE software that takes you from design capture through mask data preparation. All production proven in thousands of designs.
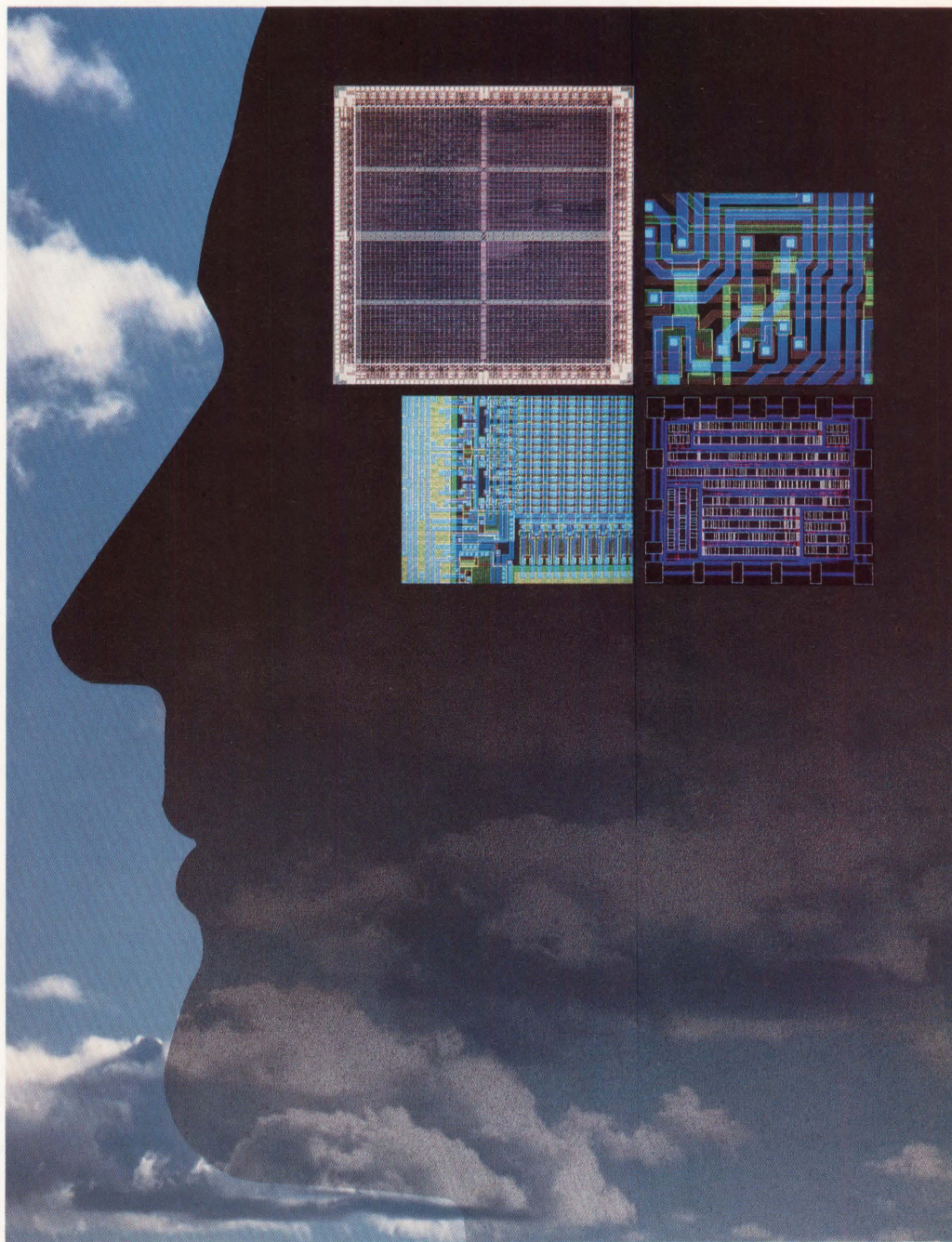
Silar-Lisco Offices

Corporate Headquarters
1080 Marsh Road
Menlo Park, CA 94025
Telephone: (415) 324-0700

European Headquarters
Brussels, Belgium
Telephone: 32-16-200016

Far East Headquarters
Nihon Silvar-Lisco
Tokyo, Japan
Telephone: 81-3-449-5831

SILVAR-LISCO

CAE Software products for electronic design

Call or write to get detailed literature about the IC layout products that have made Silvar-Lisco the world's leading CAE software company.

CIRCLE NUMBER 15

SURVEYING the RISC REALM

A quick rundown of eight current processors

BOB CUSHMAN, SENIOR EDITOR

*I*n this second part of our series on RISC microprocessors, we take a high-level look at eight RISC chips in roughly ascending order of their architectural complexity and system-level completeness. The first three—VLSI Technology's ARM, Intel's 80960, and Advanced Micro Devices' 29000—are aimed at embedded applications. It is not so much that they couldn't be used effectively for user-reprogrammable computers (such as workstations), but that their suppliers see more immediate and higher-volume potential in the large embedded controller market.

The next three—Sun Microsystems' SPARC, Intergraph's Clipper, and Motorola's 88000—are being aimed at the "Unix box" user-reprogrammable market, specifically for workstations. Some of these are already in well-known workstations, such as the Sun-4 and the Interpro-32C.

The last RISC chip—designed by Texas Instruments in conjunction with Control Data—is an R&D prototype. It is significant because it demonstrates that a 32-bit RISC processor can be put into gallium arsenide and reach well beyond the 10- to 30-MIPS performance of present silicon-fabricated RISC devices, perhaps up to the 200 MIPS projected for this R&D effort.

■ THE 86C0X0 ARM: THE LOWEST COST

VLSI Technology Inc. (Tempe, Ariz.) acquired the ARM microprocessor and architecture from the British personal computer maker Acorn, for whom it served as the foundry. Created for the Acorn Archimedes personal computer, ARM (Acorn

RISC Machine) has true RISC simplicity—just an elementary three-stage pipeline—and a modest 27-register data file, which accounts for its small chip size and economy. (Its complexity—or simplicity—is similar to that of TI's GaAs CPU.)

In its present 2.0-μm geometry, the device, designated the 86010, has a die size of 230×230 mil^2. With the pending shrink to 1.5-μm geometry, that will go down to 180×180 mil^2. A future shrink to 1.0-μm geometry will cut the size to about 150×150 mil^2. Accordingly, the price should drop from its present $50 level toward $15. Meanwhile its speed should increase from the present 12-MHz clock rate to 20 MHz and then perhaps to 40 MHz. These values contrast sharply with the 400×400-mil^2 die sizes and $1,000 price tags of high-end RISC chips.

Ron Cates, ARM product manager at VLSI Technology, says his company expects to ship 90,000 to 100,000 units this year. Users are said to like the ARM's low cost even compared with that of CISC 32-bitters like the 68020. It is finding use as a flexible software-programmable substitute for DMA controllers, as the 32-bit word size speeds the handling of data streams.

So far ARM hasn't had much software support, but VLSI Technology, seeing interest building up, is working on more support chips and software tools such as compilers. Additional software support in the form of a real-time operating system may be forthcoming from a third party, and Acorn is planning on a Unix operating system for the processor.

■ THE 80960: EXTENDING THE INTEL CONTROLLER DYNASTY

Intel Corp.'s announced motivation for the 80960 was to extend its 8048/8051/8096 spectrum of embedded controllers

designers to use, even at the assembly-language level. In particular, they say they have made the RISC pipeline transparent so that assembly-language programmers will not have to worry about timing mixups.

Fabricated with 1.5-μm geometry, the 80960 comes in at 390×390 mil^2. Though this area is much larger than the 86C0X0 ARM die, it is still smaller than the 29000's 418×418 mil^2 or the 88000's 430×430 mil^2.

Like everybody else in the RISC chip business, Intel is in the process of converting to a finer geometry (1.0 μm), with the expectation that the 80960 die will shrink to 295×295 mil^2.

■ THE 29000: ELEVATING THE INNER MICROMACHINE TO THE MICROPROCESSOR LEVEL

The usual way of describing how RISC processors evolve from CISC machines is to say that the inner microcoding of a CISC is removed. But computer architect Dan O'Dowd, who was responsible for National Semiconductor's 32000 CISC family (and went on to found Green Hills Software), points out that you could just as well form a RISC machine by removing the outer, "macrocomputer" level from a CISC, leaving the inner microcoding "computer within a computer."

The fine-grained microcoding would now be exposed (and be in RAM rather than ROM) so that the compiler would be able to optimize the instruction stream. That would satisfy the prime RISC goal of letting nothing stand between the compiler and the hardware.

This "keep the microcode" view suits AMD's thrust with the 29000, for the company has long been involved in supplying 2900-family bit-slice components to designers doing microcoded architectures. It rationalizes the 29000 as an evolutionary step in the higher integration of the 2900 bit-slice approach. It makes sense for AMD (Sunnyvale, Calif.) to provide a RISC chip for its 2900 bit-slice market since the advent of high-performance 32-bit RISC processors makes many bit-slice approaches obsolete; they are now too bulky and expensive.

Architecturally, the 29000 bears little resemblance to the 2900 bit-slice family, since it follows the University of California at Berkeley school of RISC, as can be seen by the large number of on-chip data registers—192 (Figure 2). So many registers would probably be prohibitively expensive in a bit-slice design.

That AMD is indeed finding that the 29000 serves its controller-oriented 2900 bit-slice markets is attested to by the

up to 32 bits (see also Product Showcase, p. 96). But industry observers agree that another motivation of the Santa Clara, Calif.–based company was to have a timely presence in the RISC arena without damaging the Intel 80386 microprocessor's very promising future (the odds are that the CISC-architecture 80386 will continue to be the most successful 32-bit microprocessor for at least the next three years).

Perhaps because of these split motives, the 80960 stands out architecturally as very different from previous more or less "mainstream" RISC chips. For one thing, its designers have freely used microcode in a number of places on the chip, with the

main, $3K \times 42$-bit microcode ROM feeding a wide, 42-bit chip control bus (Figure 1). For another, they have thrown in a full IEEE-754 floating-point unit complete with its own 80-bit registers. For yet another, they have included a large number of specialized controller-type instructions, like Boolean logic operations and bit manipulation.

This curious mix of RISC, CISC, and one-chip microcontroller architectures makes the 80960 seem a real challenge to understand as you read the manual. However, the designers insist that the 80960 is just the opposite. They say they have taken pains to make it easy for control system
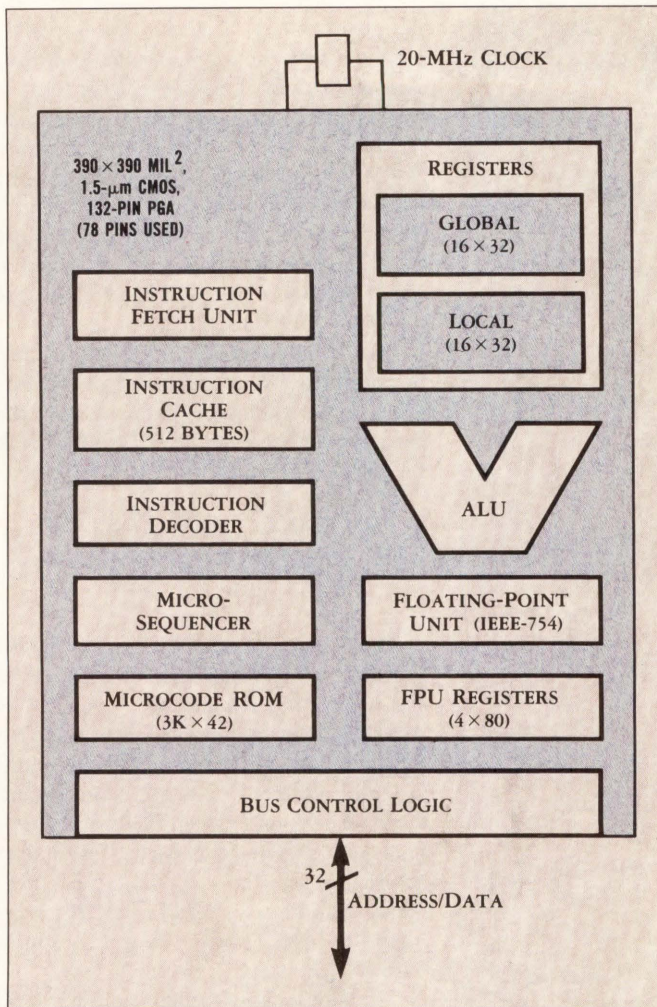
Figure 1. Intel's 80960 RISC chip seems to have the most original architecture. Unlike most other RISC processors, the 80960 makes use of microcode ROM. The single multiplexed external bus indicates that the designers were more interested in low cost than in maximum performance.

Figure 1 labels: 20-MHz CLOCK; 390 × 390 MIL^2, 1.5-μm CMOS, 132-PIN PGA (78 PINS USED); INSTRUCTION FETCH UNIT; INSTRUCTION CACHE (512 BYTES); INSTRUCTION DECODER; MICRO-SEQUENCER; MICROCODE ROM (3K × 42); REGISTERS; GLOBAL (16 × 32); LOCAL (16 × 32); ALU; FLOATING-POINT UNIT (IEEE-754); FPU REGISTERS (4 × 80); BUS CONTROL LOGIC; 32 ADDRESS/DATA

Figure 2. AMD's 29000 has the most data registers—192—of any RISC chip, making an extreme example of the UC Berkeley RISC philosophy. One reason for that was to have high performance for embedded applications.

Figure 2 labels: 25-MHz CLOCK; 418 × 418 MIL^2, 1.2-μm CMOS, 169-PIN PGA; BRANCH TARGET CACHE (2 × 64 × 32); REGISTER FILE (3 PORTS, 192 × 32); 4-STAGE PIPELINE ■ FETCH ■ DECODE ■ EXECUTE ■ WRITE BACK; TRANSLATION LOOK-ASIDE BUFFER (2 × 32 × 64); ALU (32); SHIFTER; 32 INSTRUCTION; 32 ADDRESS; 32 DATA

number of controller design wins AMD has announced for the chip. A good example is a military communications controller that the customer, Magnavox Government and Industrial Electronics Co. (Fort Wayne, Ind.), says would otherwise have required a great deal of dedicated special-purpose hardware. Magnavox says that the 29000 had enough speed to do pseudoreal-time DSP filtering of pulses and then perform signature recognition and decoding.

■ SPARC: THE MOST ITERATIONS

The goal of Sun Microsystems Inc. (Mountain View, Calif.) in creating the SPARC (*Scalable Processor Architecture*) architecture was to kick off a de facto standard that might interest a large number

of semiconductor foundries. Such a development would ensure Sun wide supply of processors at competitive prices and with continually upgraded performance that it could use for its workstations.

The CPU was purposely kept somewhat bare-bones—it has no cache or floating-point unit (Figure 3)—so that it could be one of the first RISC devices implemented in ECL and GaAs, according to Bill Joy, Sun's vice president of research and development. The CPU's simplicity allowed the first SPARC implementation to be realized in a 20,000-gate, 1.5-μm gate array from Fujitsu Microelectronics Inc. (San Jose, Calif.), which is the chip currently used in the Sun-4 workstations.

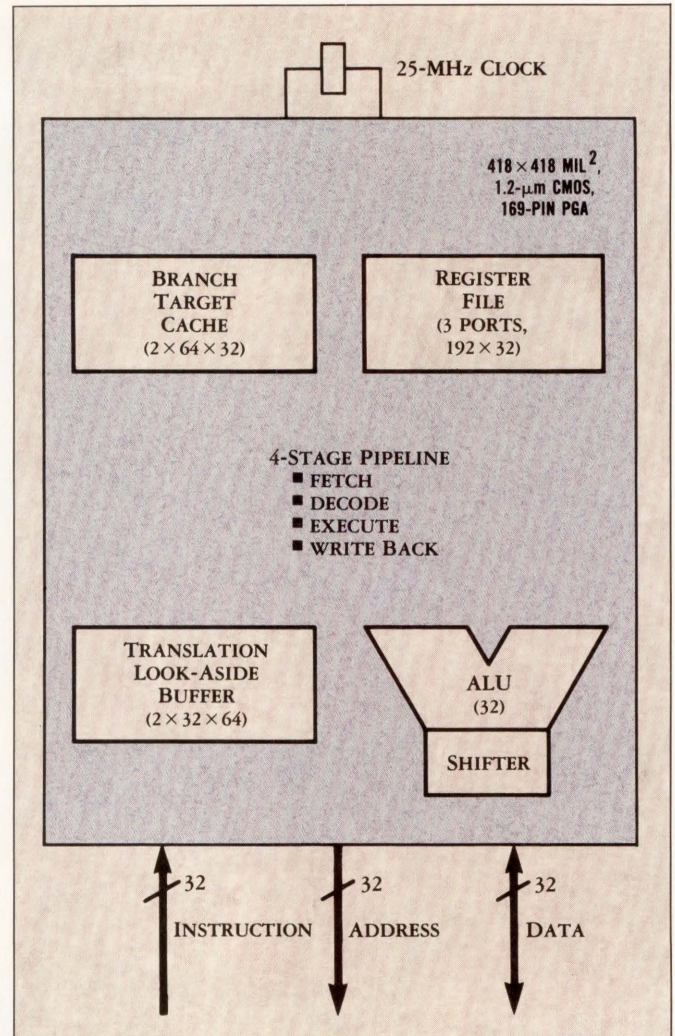But though the SPARC CPU is simple, it still has the large

number of data registers typical of Berkeley-style RISC designs. It is these efficiently addressed CPU registers, as discussed in the previous article, that allow a RISC processor to process data rapidly without having to constantly load fromand store in external memory, like CISCs. It is possible for the software to use these on-chip data registers in different ways. Usually they are assigned to as many groups as there are software tasks. Further, these groups are overlapped when it is necessary to pass parameters in context switching.

In line with Sun's continual encouragement of ever-higher-performance implementations, Cypress Semiconductor Corp. (San Jose), one of the growing family of SPARC suppliers, is bringing out a slightly en-

hanced version in full-custom 0.8-μm CMOS. The 310 × 310-mil^2 chip is designed to run at a peak 33 MIPS and produce a sustained 20 MIPS. Cypress is also working on companion chips to flesh out SPARC systems. It is developing a cache controller that will mate with some special latched versions of its 15- to 20-ns SRAMs. It is also creating an interface to Texas Instruments' new 74ACT8847 floating-point unit, which performs a single-cycle multiplication in 40 ns.

Future plans at Cypress call for a memory management chip designed to the newly defined Sun MMU specification. Future plans at Sun are looking aggressively toward 500-MHz clocks, 50-ps gates, and 4-million-transistor CPUs, according to Dave Ditzel, man-
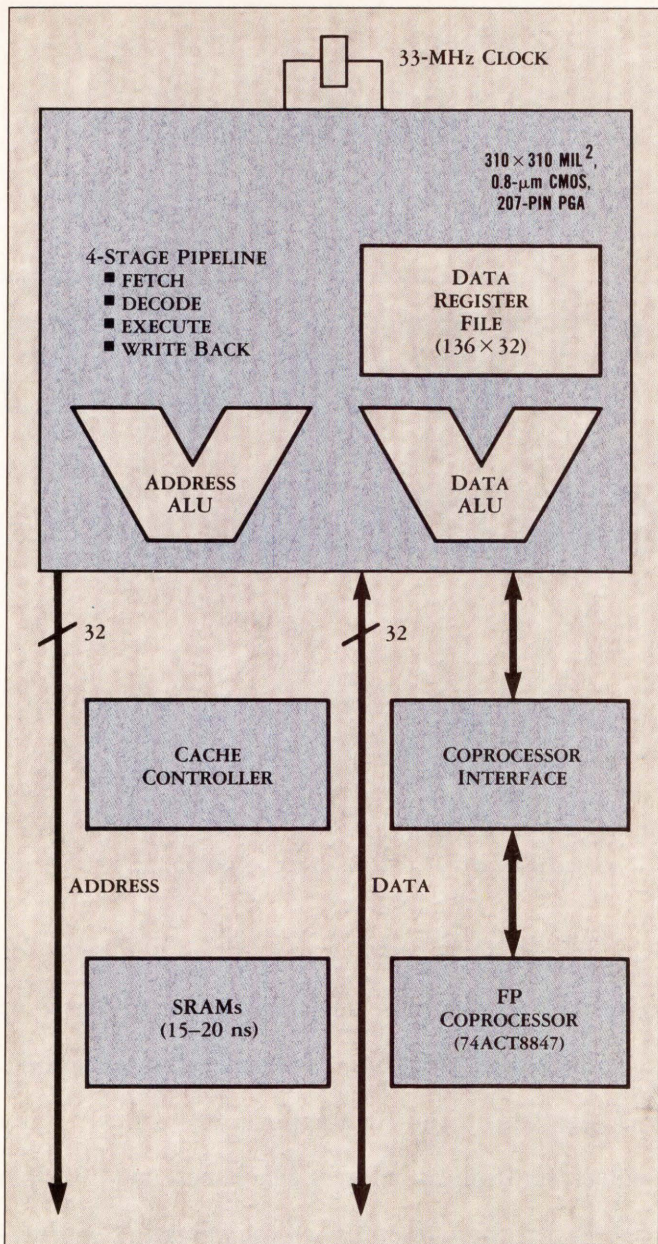
310×310 MIL^2,
0.8-μm CMOS,
207-PIN PGA

4-STAGE PIPELINE
- **FETCH**
- **DECODE**
- **EXECUTE**
- **WRITE BACK**

DATA REGISTER FILE
(136×32)

ADDRESS ALU

DATA ALU

32

32

CACHE CONTROLLER

COPROCESSOR INTERFACE

ADDRESS

DATA

SRAMs
(15–20 ns)

FP COPROCESSOR
(74ACT8847)

Figure 3. Designed for performance growth through progressive conversion to new semiconductor technologies, Sun Microsystems' SPARC follows the UC Berkeley philosophy of lots of on-chip data registers with overlapped windows for fast context switching.

25-MHz CLOCK (PLL)

430×430 MIL^2,
1.5-μm CMOS,
180-PIN PGA

REGISTER FILE
(32×32)

SCOREBOARD

ARCHITECTURE PROVIDES HOOKS FOR ADDING MORE FUNCTION UNITS

INTEGER AND BIT FIELD UNIT

FPU

INSTRUCTION UNIT

DATA UNIT

32 **ADDRESS**

32 **DATA**

32 **ADDRESS**

32 **DATA**

500×500 MIL^2,
1.5-μm CMOS,
180-PIN PGA

16-KB INSTRUCTION CACHE AND MMU

500×500 MIL^2,
1.5-μm CMOS,
180-PIN PGA

16-KB DATA CACHE AND MMU

MAIN MEMORY

Figure 4. Motorola's 88000 RISC architecture stresses performance growth through extendable parallelism. At present there are five "function units" on the CPU chip that can be operated in parallel, and the architecture can extend that to 11 paralleled function units.

ager of advanced architecture at Sun.

■ MIPS: THE MOST OPENLY ADMIRED RISC CHIP

When we asked RISC designers which RISC microprocessor besides their own they respect the most architecturally, many named the device from MIPS Computer Systems Inc. (Sunnyvale), because it is the closest to a "classic" RISC design. This design came out of Stanford University, and in contrast to the two "Berkeley" RISC processors just discussed,

it has fewer general registers—only 32.

The processor has been upgraded to the R3000 version, which runs at 25 MHz. In addition, a 40-ns-cycle floating-point coprocessor, the R3010, is in silicon and said to be fully functional. The latter can perform single-precision multiplications in four cycles.

Perhaps most important for the competitive standing of MIPS's processor among other RISC devices is the support it has finally built up among three foundries: LSI Logic Corp. (Milpitas, Calif.), Per-
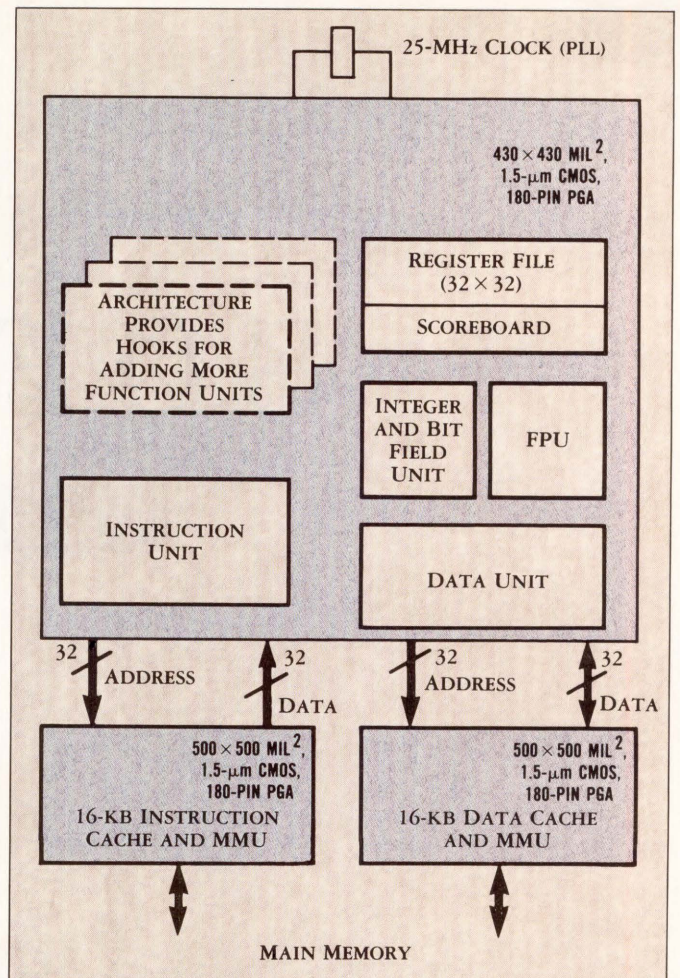
formance Semiconductor Corp. (Sunnyvale), and Integrated Device Technology Inc. (Santa Clara, Calif.). LSI Logic, also a source for the SPARC chip, has been pushing the MIPS processor as an ASIC core, saying that the small size of the basic CPU leaves room for other add-on functions.

Like the other RISC suppliers, MIPS is openly talking about its schedule for progressively stepping up to more advanced semiconductor processing to decrease the die size and increase the speed. It says it will be at 0.8- to 0.5-μm CMOS by 1990 and delivering nearly 80 MIPS and 11 MFLOPS.

■ THE CLIPPER: IMPLEMENTING CRAY'S CONCEPTS

The Clipper chip set embodies several of the architectural concepts advocated by

Seymour Cray when he was at Control Data, before the RISC buzz word was coined. Foremost of these is that the CPU should have co-equal integer and floating-point units. This is hardly a RISC concept—most of the original RISC designs had only minimal integer ALUs—but it has turned out to be a most desirable feature for 32-bit microprocessors. In fact, it is so desirable that all the RISC chips that don't have floating-point are being mated with external floating-point units, just like the 32-bit CISC microprocessors.

The Clipper was introduced in 1985 (by what was then Fairchild Semiconductor Corp.) as a 33-MHz "RISC-like" processor. Its implementation of Cray's concepts in a CMOS chip set apparently served as a guide for Motorola's 88000

Retargetable

SOFTWARE

DEVELOPMENT TOOLS

Retargetable tools easily handle most custom architectures, using high-level languages

BOB NORIN AND KEVIN NOLAN, QUANTITATIVE
TECHNOLOGY CORP., BEAVERTON, ORE.

Execution speed is the most important requirement for many system applications, including signal processing, image processing, graphics, and telecommunications. The architectures of computers designed to run such applications contain many functional elements, such as adders, multipliers, address generators, and multiple memory and data bus resources. When these resources all are operating simultaneously on many streams of data, the resulting machine has a very long instruction word (VLIW) architecture, which is considered a subset of a single-instruction, multiple-data (SIMD) architecture.

VLIW processors present serious difficulties for the microcode application programmer. Very long instruction words, which can contain as many as 1,000 bits, are composed of numerous instruction fields. During each machine cycle, at the stage when a conventional processor would fetch an instruction that controls a single execution unit, VLIW machines fetch instructions that control multiple execution units simultaneously. Each field must be programmed for each instruction clock cycle. Moreover, because any of the instruction units may be pipelined, complex operations such as floating-point addition can take several instruction cycles to actually complete.

PHASE LOCK LOOP.
ANALOG/DIGITAL ASIC.
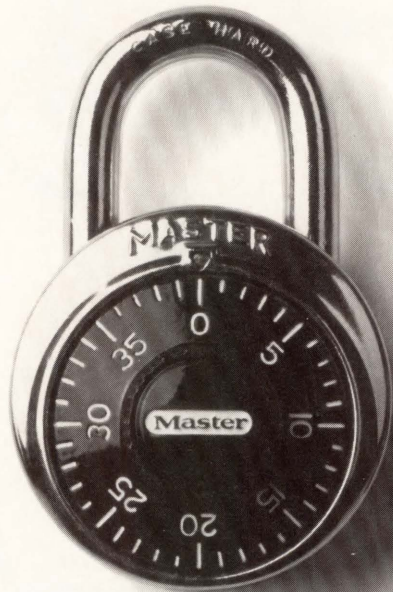
NO ONE ELSE HAS
THE COMBINATION.

System level designers can access new levels of integration designing products with a combination of advanced analog/digital circuitry on a single ASIC. Obtaining the smallest form factor, lowest power, and highest performance in data capture applications is now possible.

Consider our 34MHz Phase lock loop subsystem – the PLL34M – an ideal solution for data separators.

The PLL34M is an open loop phase lock subsystem containing a voltage controlled oscillator, digitally controlled charge pump, trimmable reference voltage, and high speed driver I/O circuitry.

Combined with our other analog and digital standard cells, the PLL34M is ideal for increasing system integration of disk drive systems. (See application diagram.)

For designers seeking to incorporate analog processing functions, such as amplification, filtering, and data conversion, Sierra offers a wide range of solutions for applications in disk and tape drive systems, local area networks, and servo controllers.

SIERRA integrates PLL onto a single data separator ASIC.

DATA IN

PHASE DETECTOR

PLL 34M

VCO OUTPUT

USER LOGIC

SIERRA DATA SEPARATOR ASIC

Lock into our CMOS standard cell library which includes over 250 digital standard cells, 50 analog standard cells, and 20 EEPROM standard cells. Our Triple Technology™ gives you the flexibility to combine analog, digital, and E² on the same chip.

Sierra reduces packaging size and cost by offering low profile, surface mount packages including SOIC, PLCC, and PQFP Gull wing packages.

CMOS analog standard cells are available now in the same design environment as digital. Importantly, on-site mixed analog/digital simulation is easily achieved through access to our MIXsim™ software tools, which verify design functionality and assure first time silicon success.

Access the unbeatable combination of analog/digital ASIC from Sierra. Lock into the latest data on our CMOS standard cell library and provide your customers with the most advanced products. Just write or call today for our complete library card.

Sierra Semiconductor
Triple Technology™ In CMOS.

2075 North Capitol Avenue, San Jose, California 95132. Telephone (408) 263-9300

CIRCLE NUMBER 16   Triple Technology and MIXsim are trademarks of Sierra Semiconductor. Master is a trademark of Master Lock Co.
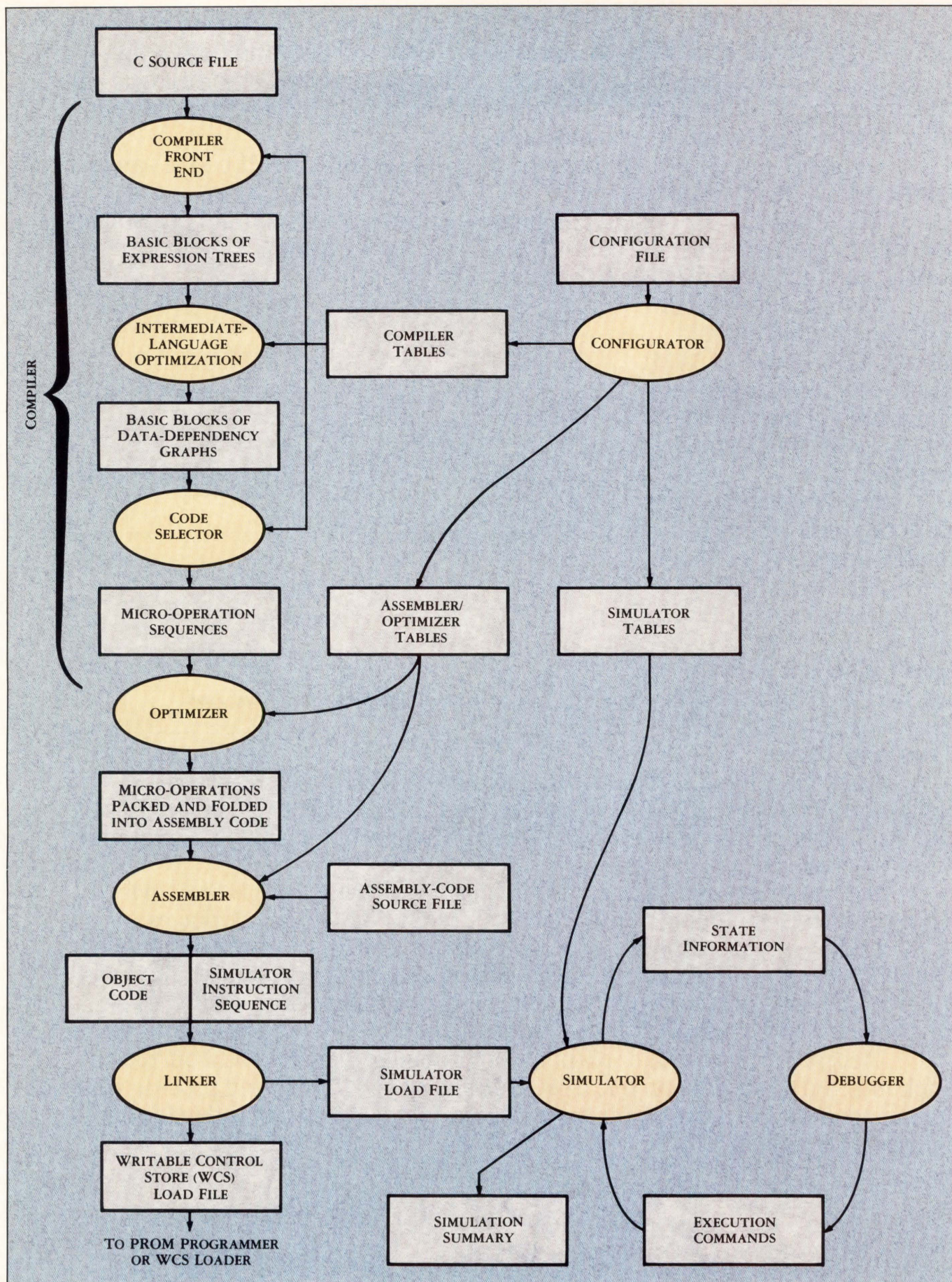
Figure 1. The SF/Configurator maps a target architecture, specified in a configuration file, into tables used by the rest of the Software Foundry. Source-code files expressed in C can then be compiled, optimized, and assembled into microcode for that architecture. Simulation tables from the configurator enable the user to simulate the execution of the microcode under the control of an interactive debugger.

Processor complexity isn't the only difficulty with VLIW architectures. Developments in the design of hardware have outpaced software development for the last several years. For example, numerous chips have become available for digital signal-processing systems, but few advances in software design have appeared to ease the programming of those systems.

The performance of VLIW processors, however, is intimately dependent on the efficiency of their microcode—the programming language that directly controls the setting of microinstruction bits on each instruction cycle. Current design practice creates major components of the application code directly in microcode to improve the code's execution rate. This practice results in an exponential growth in the complexity of developing reliable and efficient programs. Also, the lifetime of microprogrammed machines is decreasing, meaning that new microcode is needed more frequently. This situation renders hand coding and hand optimizing economically and technically obsolete.

The alternative is automation of the microcode development process. There are two key issues in this automation: optimization and retargetability. Optimization increases throughput and reduces memory requirements. Microcode written for VLIW architectures must be optimized to exploit the processor's resources most efficiently. Efficient optimization requires intimate knowledge of the target architecture, making optimization a formidable challenge in a retargetable (machine-independent) system.

Retargetability is the ability of a software system to adapt code for a wide variety of target architectures. Retargetable software development tools free the programmer from much of the complex labor of writing each bit of microcode, decreasing development time and cost and increasing reliability. As software development routinely consumes the majority of a project budget, retargetable tools can significantly reduce development schedules through increased software engineering productivity.

■ THE USE OF RETARGETABLE TOOLS

Microcode development is a growing challenge. Today's designers employ specialized hardware components that make software development at the microcode level a slow, exacting effort, and the volume of application software migrating to these high-performance processors is expected to increase significantly over the next decade. In addition, good microcode programmers are relatively scarce. Soft-

ware-design tools can play a major role in meeting this challenge.

For any development effort, the most convenient programming tools are compilers that allow the programmer to write in a high-level language, such as Fortran, C, or Ada. These tools simplify programming and improve reliability and maintainability. Unfortunately, code produced by a compiler is typically 3 to 10 times slower than code that is crafted using low-level assembly or microcode languages, according to our experience. This performance degradation underscores the need for tools that automatically optimize microcode.

Microcode optimizers, though, are highly machine-dependent. They require an intimate knowledge of processor resources, timing, and instruction word encoding. Each new machine requires a new optimizer. In addition, simulators and symbolic debuggers also improve microprogrammers' productivity.

However, the cost of developing a complete tool set is extremely high. For an average project, the development effort can easily exceed 12 man-years at a cost of over $1 million. One-of-a-kind, throwaway tools, therefore, are almost always impractical. One way to amortize the development costs is to create a retargetable tool set.

■ ALL IN THE FAMILY

Along these lines, we have developed the Software Foundry family of software development tools, which builders of custom high-speed processors can use to create microcode for their architectures. The tools are particularly well-suited to VLIW processors built from VLSI components like bit-slice processors, digital signal processors, floating-point arithmetic chips, sequencers, address generators, and application-specific ICs.

The Software Foundry product family comprises the SF/C Compiler, SF/Optimizer, SF/Assembler, SF/Linker, SF/Librarian, SF/Simulator, SF/Debugger, and the SF/Configurator. Through a user-specified "configuration file," the tools adapt to new and highly complex architectures, including those with high degrees of parallelism and multiple-level pipelines. Written in a flexible Lisp-like language, the file describes the resources, timing, and interconnection of the target architecture. Once the file is constructed, it is compiled to create the configuration database that drives the other components of the Software Foundry.

Although there are other retargetable tools (such as meta-assemblers), the Software Foundry is the first to offer a fully

retargetable microcode optimizer. Ordinarily, optimization is tied to a specific architecture, but we have found a way to design machine-independent optimization algorithms that can extract machine-specific details from the configuration database.

The Software Foundry C Compiler translates the widely used C programming language directly into assembly code (Figure 1, top). In doing so, it performs an initial optimization based on a machine-independent analysis of an intermediate representation of the program. For example, it moves statements out of loops that do not affect the results of the loop, thereby decreasing the execution time of the loop.

The SF/C Compiler produces directives within the code so that the code can be further optimized by the Software Foundry Optimizer. The input to the SF/Optimizer is syntactically correct, "straight line" assembly code (in which only one operation is performed in each instruction). Code efficiency improvements during this phase include code compaction, loop folding, and trace scheduling procedures that relate specifically to the target architecture. Another optimization at this point involves assigning frequently used variables to registers. The output from the SF/Optimizer is assembly code in which many operations are performed within each instruction, to the extent permitted by the architecture. The code output from the SF/Optimizer typically executes many times faster than the input code.

In addition to this automatic optimization, the SF/Optimizer also has an interactive mode within which the designer can create microcode manually. In the interactive mode, the optimizer uses an editor/assembler user interface to assist the programmer in manipulating micro-operations and tracking resource usage, data dependency information, and other information crucial to achieving an optimally written microcode program.

The SF/Optimizer operates in conjunction with the other tools in the Software Foundry. It shares the same configuration database and makes use of common assembly functions with the SF/Assembler, and the fully automatic feature allows the SF/Optimizer to serve as the back end for the SF/C Compiler.

The SF/Assembler permits the developer to customize a microcode programming environment for a specific target machine. It supports features that enable programmers to microcode VLIW pipelined machines efficiently. It has a syntax that lets developers define constructs that are normally found in high-level languages.

```
(machine 'super_cpu (
  (memory 'Data_RAM (
    (width 32)
    (upper_address_limit 32767)
  ))
  (memory 'PC_reg (
    (width 32)
    (upper_address_limit 0)
    (type program_counter
  ))
  (memory 'Reg_file (
    (width 32)
    (upper_address_limit 32)
  ))
  (memory 'Data_bus (
    (width 32)
    (lifetime 0)
  ))
  (memory 'Instruction_reg (
    (width 57)
    (upper_address_limit 0)
  ))
  (memory 'Control_Store (
    /* Pipeline mode is on, Don't-care bits are 0 */
    (width 57)
    (upper_address_limit 4095)
    (type control_store)
    /* Layout of instruction word */
    (define_field 'immediate (range 0 31))
    (define_field 'Alusel (bit 32))
    (define_field 'AluFunct (range 33 35))
    (define_field 'ReadSel (bit 36))
    (define_field 'WriteSel (bit 37))
    (define_field 'EnableRW (bit 38))
    (define_field 'SroSel (range 39 42))
    (define_field 'DestSel (range 43 45))

    (mo "<br_op>" ()) /* Branch instruction */
    (non_terminal 'br_op () (
      (can_be "BR <dest_label>" (
        (set_bits (field 'AluFunct) (direct 1))
        (set_conflict_flag 'dbus (at_cycle 1))
        (set_bits (field 'immediate) (relative 'dest_label))
    ))))
    (non_terminal 'dest_label () (
      (can_be (generic (label 'Control_store)) ()
    )))
    ⋮
    (control_node 'goto '(BR dest_label)
      (where (is_dest 'dest_label)))
  ))
))
```

Figure 2. The designer creates a configuration file that details machine resources and how to use them. For example, memory and register resources are called out with "(memory . . .)" statements, and bit-field settings are described in "(define_field . . .)" statements.

The SF/Simulator allows a programmer to simulate the execution of an assembly-language or microcode program using a hardware description of the target machine based on the configuration database. It thus provides an environment in which code can be executed and debugged without having working hardware. This feature permits hardware and software development to proceed independently. The simulator works in concert with the symbolic debugger for debugging at the source-code level.

The Software Foundry is adapted to an architecture through the construction of a hardware configuration database. This database, generated from the user's configuration file, contains information about the target machine of relevance to the tools. This information includes the syntax and semantics of microcode operations, machine resources, instruction bit settings, and features that relate C constructs to corresponding assembly language statements and hardware resources (see the table).

■ THE CONFIGURATION PROCESS

The configuration file is written in a unified configuration language that is reminiscent of Lisp (Figure 2). Once the file is constructed, the SF/Configurator produces a configuration database that is accessed by the tools. The configuration process occurs only once per architecture, although, when the hardware is modified, the user must modify the configuration file, rebuild the database, and recompile the application.

The process of configuring the tools is the most crucial aspect of the Software Foundry development system. The tools are retargetable because the machine-specific details of processes that are normally hand-coded into assemblers, simulators, and compilers are now expressed in the form of a database. The user retargets the tools by describing the target architecture to the SF/Configurator, and the configurator generates the database. This database instructs the tools how to perform the various transformations necessary to compile, assemble, and simulate a program.

Within the configuration language, elements of a list are separated by a space. The position of elements within the list is important to the processing of the configuration file; in other words, they are *position-dependent*. The meaning of a particular list element is determined by the context in which it was specified. The context of a list is determined by the aggregate context of all the parents of the list. For example, the memory directive— "(memory . . .)"—appears in several different contexts in the configuration file. If it appears under the machine directive— "(machine . . .)"—it means "define the following memory." If it appears under the default storage class directive— "(default_storage_class . . .)"—it means "bind the specific storage classes to the specified memory."

Given the configuration database, the compiler translates ("maps") the C language into assembly language, which in turn is assembled into the microcode of the target machine. If the semantics of the original C program are preserved during this process, the resultant microcode is said to be correct. In other words, when

MAPPING OF CONFIGURATION DATABASE

C LANGUAGE COMPONENT	TARGET ARCHITECTURE
C operations (+ , – , *, /, ...)	Hardware functions and the associated control pin settings (of the device performing the function) corresponding to a particular microcode field set to a specific value
C storage classes (register, auto, ...)	Machine storage resources (e.g., data memory, floating-point registers, etc.)
C control flow (if-then-else, for, ...)	Sequencer operations and associated microcode field and value settings to perform the specified operation
C function invocation (a = foo(c,d), ...)	Specific sequence of micro-operations for activation, record creation, argument passing, and return value retrival

executed on the target machine, the program will execute exactly as it was programmed. This translation process can be broken down into four distinct mapping functions, as presented in the table.

Of course, the efficiency of the mapping depends on how completely the user specifies his architecture. The compiler itself is not intrinsically intelligent. Rather, it provides the means to make intelligent decisions that effect code efficiency.

In a way, configuring the compiler is much like building an expert system; the knowledge has to be put into the system to allow algorithms to make decisions. Likewise, the compiler employs intelligent algorithms to perform code selection and optimization. However, the user must describe the knowledge with which the algorithms operate in the configuration file.

The configurator goes through a three-stage process to configure the tool set. First, the SF/C compiler must be told (through the compiler tables) how to convert from C syntactic structures into assembly code. Second, the SF/Assembler must be told how to translate from assembly language into microcode. Finally, the SF/Optimizer must be told what operations can occur in parallel.

To understand how the configured tools work, consider that the SF/C Compiler encounters a branch instruction in the C source code. Given the configuration file excerpted in Figure 2, the compiler finds the machine operation *goto* in the configuration database which was defined in the statement "(control_node . . .)." The compiler inserts this operation, along with the information that indicates where to find the branch address, into the sequence of micro-operations.

Similarly, if the compiler encounters a multiplication/accumulation function (a multiply followed by an add) in the C source, it looks into the configuration

database to find the pattern that identifies a multiplication and addition hardware function. If it doesn't see that pattern, it breaks the operation down into simpler operations and tries to match those patterns in the database.

Configuring the compiler is a detailed process. For example, the C plus operator, " + ," can be used to add characters, short and long integers, and both single- and double-precision floating-point numbers. If the machine supports addition of each of these types, the compiler must be told how to add two operands of each type. Another mapping must then be defined for the subtraction operator ("–"), the multiplication operator ("*"), the division operator ("/"), and so on. This process continues until mappings for all the operators and types to be supported are described.

Mapping the functions configures all the tools, not just the compiler. For example, in Figure 2 the first five "(memory . . .)" statements provide the data path width, program addresses, and other information about memory and register resources in the architecture. This information is used by all tools during compilation, optimization, and assembly steps. The "(define_field . . .)" statements define the microinstruction bits that the assembler will set during assembly. The "(set_conflict . . .)" statement is used by the optimizer to recognize, in this case, what resources are captured by the branch instruction *br_op*. Captured resources cannot contribute to parallel operation that the optimizer tries to implement.

■ ALTERNATIVES

Retargetable tools can be easily configured into versions tailored to specific processors, then reconfigured over again for any number of completely different architectures. Retargetable tools make it easy

to build the software development programs required by the programmers who are developing the actual application software. Such tools also enable the user to create new architectures and reuse existing application software. The reusability of familiar tools on multiple projects means shorter learning curves and decreased maintenance. These capabilities increase the productivity of the programmers who must write highly efficient code for time-critical applications.

The question naturally arises whether retargetable compilers are as good as those directly targeted at a specific architecture. The major issues here are access to the talent to develop the tools, timeliness, improving programmer productivity to reduce the cost per bit, and optimization for greater code efficiency. If you have the time, expertise, and a very liberal budget, you can build high-level–language development tools that produce more efficient code more quickly than retargetable tools. The cost of doing so, however, must be compared with the attributes of retargetable tools as listed above.

The other choice, developing applications at the microcode level, is even less desirable. The cost of writing and debugging code on VLIW architectures is between $1 and $2 per bit. An application requiring 10,000 microinstructions, with each microinstruction 100 bits wide, could cost between $1 million and $2 million to develop by writing microcode. If one considers this cost in addition to the other major issues mentioned above, retargetable compilers would seem more effective economically for all but the highest-performance applications, where the execution of critical operations must be optimized by hand, with the aid of microcode CASE (computer-aided software engineering) tools like the SF/Optimizer. ■

ABOUT THE AUTHORS

BOB NORIN *is the vice president of engineering at QTC. Previously, he managed applications software engineering at Floating Point Systems and at Genrad. He has also held a senior software engineering position at the Hossdorf Laboratory for experimental mechanics in Basel, Switzerland. He holds a PhD in electrical engineering from Cornell University.*
KEVIN NOLAN *is a senior software engineer at QTC and the project leader for the SF/C Compiler and the SF/Configurator. He was the designer of the configuration file and database of the Software Foundry. Before being employee number 1 at QTC, he worked at Modular On-line Systems. Previous projects include the radar simulator for the B-1 bomber and a retargetable compiler for a QTC client.*

SEMICUSTOM

Applications Report

ILLUSTRATION BY CHERI DORR

ONE CHIP Implements a Medical Measurement and Control System

TERENCE E. CRIMMINS, MIT DEVELOPMENT CORP., NORWALK, CONN.,
AND ZAHEER HASSAN, SIERRA SEMICONDUCTOR CORP., SAN JOSE, CALIF.

Sometimes the only way to sufficiently reduce the size of electronic equipment is to integrate a wide variety of functions on one chip. If any one of the necessary functions cannot be put on the chip, the value of integration is reduced considerably.

This was the situation faced by MIT Development Corp. (MDC) in decreasing the size of a handheld medical instrument. The unit had already been made very small using an efficient layout of discrete parts. It comprised approximately 70 components and was about 20 square inches in area, roughly the size of a paperback book. However, MDC wanted to reduce the size even further—to approximately 6 square inches (the size of a matchbox), with a third of the components—so that the instrument could be carried easily in a user's pocket. The instrument sells for about $100, and the sales volume was expected to be about 100,000 units per year.

The instrument includes a sensor that reads the characteristics of an inserted sample, computes an analysis of the sample, and displays the results on an LCD. Everything except the sensor and the display had to fit into one chip. The circuitry in the original instrument made it clear what the new chip had to incorporate:

- An 8-bit microcontroller (the original instrument contained a 4-bit microcontroller; MDC wanted to upgrade to an 8-bit device to employ a more accurate analysis algorithm)
- 4K bytes of ROM
- 64 bytes of RAM
- 256 bits of EEPROM with on-chip high-voltage generator (V_{PP}) for self-contained programming
- A 12-bit A/D converter
- A 36-segment LCD driver with temperature compensation
- Low-noise op amps and comparators
- Digital I/O circuitry
- Power-on/reset circuits
- Low-voltage signal detection circuitry
- Keyboard interface logic

The core microcontroller acts as the primary computing element. The EEPROM stores calibration values and the results of the user's tests. For versatility, the LCD driver had to be compatible with several types of liquid-crystal displays. In fact, the entire chip had to be designed with the goal of applying it in a number of different instruments.

■ A SINGLE-CHIP SYSTEM

MDC had considered partitioning the analog and digital circuitry into separate chips for design simplicity but decided that the space-saving goal prohibited a two-chip approach. Furthermore, a single chip would cost less, be more reliable, lower inventory requirements, require dealing with fewer vendors, and make it more difficult for competitors to copy the design.

Until very recently, the level of system integration attempted here would not have been possible because of some barriers. For example, although an abundance of suppliers did exist that could offer highly efficient CMOS digital solutions, there were very few that
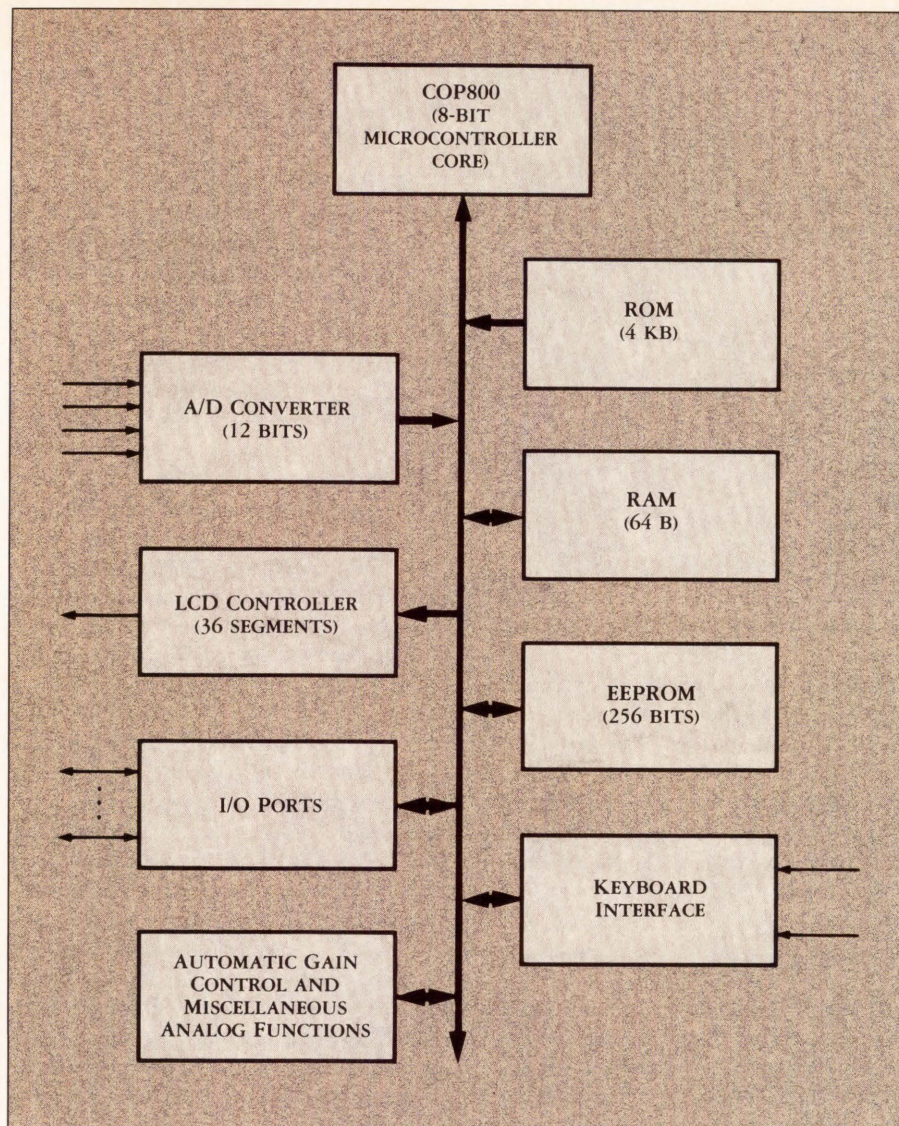
Figure 1. The single-chip measurement and control system integrates a variety of digital and analog functions, including an 8-bit core microcontroller core, EEPROM, analog-to-digital converter, and LCD driver. The RAM and ROM are actually added to the core cell.

could also integrate sophisticated analog circuits along with the digital on the same chip, and of those that did offer both digital and analog functions on the same chip, only a handful offered any appreciable selection of precharacterized ASIC cells to choose from. Thus, in the past, a custom development effort for the analog portion of the chip would have been prohibitive from a time and a cost standpoint.

Additional barriers appeared if the custom design required EEPROM and a core microcontroller, as does the application described here. Suppliers of EEPROMs have never been in big supply, and there are even fewer manufacturers that offer them as ASIC cells—especially as part of a library where analog and digital cell functions can be combined. What's more, to implement efficiently any design that involved a core microcontroller was an equally difficult task. Here, the major barrier was the physical size of the available microcon-

trollers, many of which were designed several years ago using the design rules and technology of that time. These microcontrollers were converted into core cells without having the benefits of reduced design rules and the latest process technology. For example, it is not uncommon to find a core cell in a 2-μm double-metal standard-cell library drawn with single metal and shrunk to fit the technology.

After looking closely at several semicustom IC suppliers, MDC chose Sierra Semiconductor as the design and production house. Because of the chip's functions and complexity and because the MDC engineers had never designed a custom chip before, Sierra was selected for its design expertise, as well as its ASIC production capabilities. More specifically, MDC picked Sierra because it could integrate all the necessary functions cost-effectively.

With the design teams from MDC and Sierra working together, it was decided

that a single 2-μm CMOS chip could incorporate all the necessary circuitry (Fig. 1).

■ CONVERTING DISCRETE CIRCUITS INTO CELL FUNCTIONS

The first step in the design process was to map the circuitry in the original instrument onto the functions in Sierra's cell library. A critical consideration in doing the mapping was to find out whether any cells would have to be altered to perform the functions required by MDC. Although digital cells tend to be fairly standard across discrete and semicustom devices, analog designs often need specific characteristics. It was important at the beginning of the design process to discover cells that would require customization, because Sierra could then modify those cells while the rest of the design effort went forward.

Sierra's cell library includes over 200 digital cells, 50 analog cells, more than 20 EEPROM cells, and a peripheral library designed to interface with the core microcontroller cell. The library covered every function on the chip, so that no new cells were required. Some modifications, however, were necessary to tailor the op amps and comparators to MDC's needs. Because Sierra's cell library includes 10 different op amps and 5 comparators, it was possible to choose cells that came very close to MDC's specifications and make only minor modifications.

Nearly all of the chip's functions were handled in large cells from Sierra's library. MDC designed only about 200 gates' worth of digital logic using Sierra's standard-cell library for keyboard interface and specialized I/O purposes.

On the digital side of the chip, the 8-bit microcontroller served as the foundation for system-level integration. With the microcontroller at the heart of the design, the rest of the chip's components were arranged as peripherals of the microcontroller. In fact, all of the chip's other components are memory-mapped into the microcontroller's address space so that it can easily communicate with and control everything from the A/D circuitry to the EEPROM and LCD driver.

As mentioned earlier, the microcontroller's physical size played an important role in determining the design's cost-effectiveness. In most cases, it could very well be the single most area-consuming function on the chip. Sierra's core microcontroller cell, designed using 2-μm double-metal CMOS technology, is the smallest microcontroller available in the industry. The cell measures only 66×66 mils, less than 20% of the area of the final chip. (Provisions to shrink the cell to a 1.5-μm process will mean an even smaller micro-
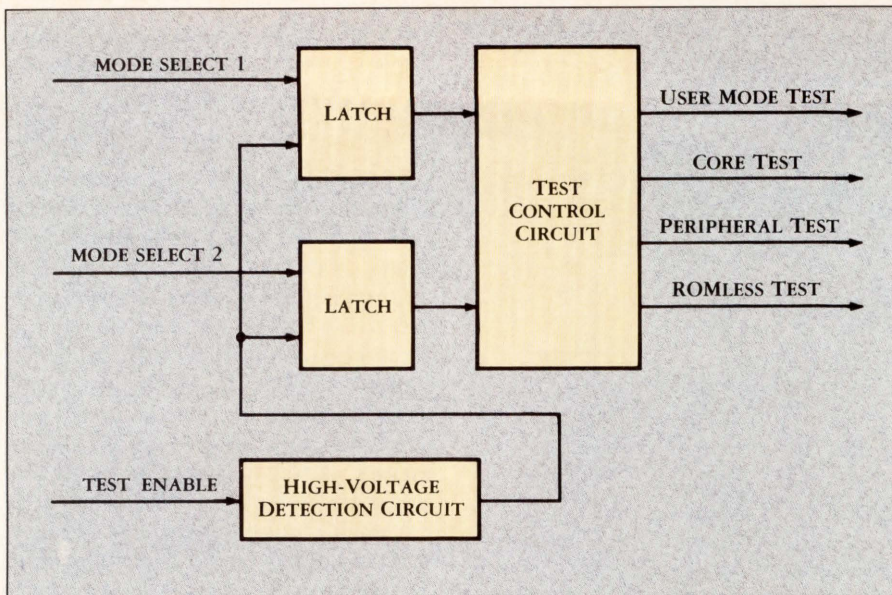
Figure 2. Sierra's standard test interface enables the chip to be tested in both the test and external ROM modes.

controller for future designs.) The small core size enables designers to put enough custom logic around the microcontroller to get a cost-effective one-chip system.

The core cell includes an ALU, the control and clock logic, a 16-bit timer/counter, interrupt logic, a serial interface, registers, core test logic, an 8-bit I/O port, and a slave-mode interface for ganging several core microcontrollers together on a single chip. The core cell runs at an internal speed of 10 MHz when operating from a 5-V power supply.

■ TAKING ADVANTAGE OF MICROCONTROLLER OPTIONS

The core microcontroller is National Semiconductor's COP800, with modifications to the pin-out to make the cell more flexible in a standard-cell design. For example, designers can create as many as 160 8-bit I/O ports and add up to 32K bytes of ROM and 192 bytes of RAM to the core cell. Additional RAM can be added alongside, with the core cell using memory bank switching schemes, but the internal approach allows for a smaller chip by reducing interconnections.

When partitioning the MDC design for the chip, the requirements for RAM and ROM mapped easily into the core microcontroller's capabilities. Thus the only memory external to the microcontroller was the EEPROM, which was available as separate cells.

At this stage of the design, Sierra recommended to MDC that the chip take advantage of a special feature of the core microcontroller that allows operation with an external ROM. In this mode, the microcontroller bypasses the internal ROM. This feature has two main benefits. First, it makes the chip easier to test; the micro-

controller can be checked out thoroughly, independent of the other internal components. Second, in case of an error in the ROM code or a future need to change the code, MDC would have a fall-back position of using the controller chip with an external ROM. External ROM also makes the chip more easily adaptable to other applications.

■ CONFIGURING THE SYSTEM

Once the microcontroller was configured and the other components arranged as peripherals, the memory-map addresses were assigned. Critical signal paths were identified, and circuit specifications such as ac timing and current consumption were determined.

Then pin-outs were defined. This step is important because of Sierra's practice of multiplexing test pins with the customer's functional pins. The test pins access the chip's internal functions only when the chip is in the test mode; otherwise these pins function as normal I/O pins. This approach avoids any need to add extra pins just for test purposes, which would increase the cost to the customer.

To ensure that the test mode would not interfere with the chip's operation, Sierra and MDC agreed on the pins to be multiplexed. Sierra's standard interface was used to enable the chip to be operated in the test mode and in the external ROM mode (Figure 2). The mode-changing algorithm included raising one multiplexed pin to 12 V, and the external ROM bus was also multiplexed with other functional pins.

At this point in the design process, Sierra provided a list of standard-cell equivalent functions so that MDC could begin breadboarding the chip. In parallel,

Sierra performed schematic capture and modified some of the analog cells.

■ CHECKING OUT THE HARDWARE

Sierra supplied MDC with a packaged version of the microcontroller for use on the breadboard. Sierra also supplied a personality board and interface board that permitted a development system to emulate the core microcontroller and peripherals. The evaluation board was necessary to adapt the core microcontroller's 68 pins to the 24 expected by the development system while still giving external access to the extra pins. The development system is a self-contained computer with its own firmware, which provides for all system operation, emulation control, communication, PROM programming, and diagnostic operations.

MDC used an IBM PC to write assembly code for the microcontroller. The assembly code was then downloaded to the development system. Breadboarding enabled MDC to evaluate the functionality and performance of the system before committing the design to silicon. In addition, it allowed MDC to run the application code and make trade-offs between implementing functions in hardware or software. If, for example, MDC had found that the analysis algorithm ran too slowly in software, a hardware multiplier could have been added to the chip. Adding this circuit would also free the microcontroller to perform other tasks, thus speeding up the entire device.

■ SIMULATION VERIFIES BOTH ANALOG AND DIGITAL FUNCTIONS

Upon verification of the code, Sierra performs system-level simulations over voltage, temperature, and process ranges using Sierra's MIXsim mixed-mode simulator. This proprietary simulator checks out both the analog and digital portions of the design thoroughly, but it does so without resorting to the overly detailed treatment that a simulator like SPICE would provide at the expense of days or weeks of computer time.

Unlike SPICE's network-equation approach, Sierra's MIXsim simulator employs behavioral analog models that describe an analog function in terms of what it does, which in turn is based on the relationship between signals and their histories. MIXsim eliminates unnecessary computation by using an event-driven approach, as well as variable voltage and time-step resolution. Consequently, the entire simulation phase for the MDC design at Sierra took less than a week. In addition to running faster,

OP AMPS AND COMPARATORS

12-BIT A/D CONVERTER

PORT LOGIC

RAM (64 B)

ROM (4 KB)

8-BIT CORE MICROCONTROLLER

AGC CIRCUIT

KEYBOARD INTERFACE LOGIC

POWER ON/RESET AND LOW-VOLTAGE DETECTION CIRCUIT

HIGH-VOLTAGE GENERATOR

EEPROM (256 BITS)

LCD CONTROLLER (36 SEGMENTS)
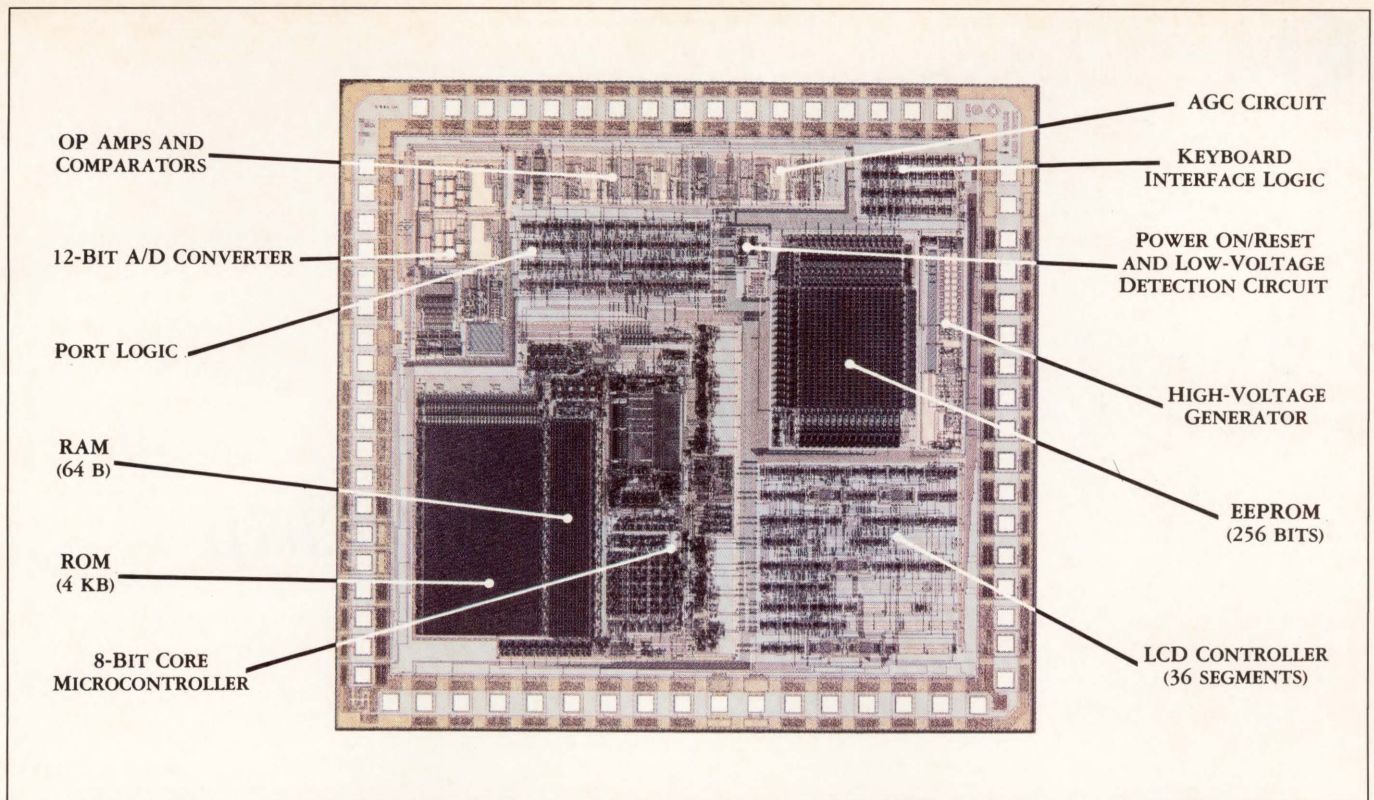
Figure 3. The chip measures some 215×215 mil^2. The core microcontroller occupies only about 20% of the area without RAM and ROM and about 30% with the memories.

MIXsim's behavioral models allow it to detect and flag faulty cell usage and specification violations.

Using fast behavioral models makes possible full-chip simulations rather than having to perform separate analog and digital simulations. The full-chip method eliminates timing and logic errors at the analog-digital interface and lets users make performance and accuracy trade-offs. This method also verifies that the chip will be testable.

Sierra has a simulation model of the core microcontroller into which the application code can be loaded and run. To reduce the simulation time, Sierra simplified the code so that some of the functions, such as the A/D section and the EEPROM, are simulated only once. Although MDC had already tested the overall design in the instrument's original version, simulation was necessary to ensure that this implementation met the required timing and other specifications.

After MDC and Sierra were satisfied with the simulation results, Sierra performed automatic placement and routing of the chip. The core microcontroller, including RAM and ROM, occupies approximately 30% of the chip area, which came in at approximately 215×215 mil^2 (Figure 3). Other functions were fit in around the microcontroller, with the LCD driver and the EEPROM (with its attendant V$_{PP}$ generator) fitting snugly to the microcontroller's right. This was a logical location

for these cells because the microcontroller's I/O connections come off to its right.

■ SIMULATION LEADS TO TESTING

When the layout was complete, the entire chip was resimulated using capacitances extracted from the layout, and the results were compared with those of the prelayout simulation. With MDC's approval, the chip was released for mask making and then prototype fabrication. The prototypes were mounted in ceramic packages for fast assembly turnaround. Sierra fully tested the prototypes on a Sentry Series 80 tester, which is capable of testing mixed analog-digital circuits.

To create the test program, vectors from the MIXsim simulator were used extensively. Parametric tests were generated both automatically and manually. To save overall time, the test programs were prepared during the chip's final development phase. Because Sierra provides test vectors for the core microcontroller and peripherals, MDC only had to write test vectors for the approximately 200 gates of random logic. For each set of the standard test vectors, pin-out designations were modified to match the actual circuit pin-outs.

The test programs were organized in four phases. In the first two phases, the core microcontroller was tested with and without its internal ROM. The microcontroller was then used to test the rest of the

chip's circuitry in the next two phases, which covered the peripheral functions and the customized I/O logic. Sierra concatenated the programs for all the phases into a comprehensive test program.

With the exception of an error in the ROM code, the first prototypes of the chip worked perfectly and the design has gone into successful production. Incorporating external ROM interface logic paid off, and the chip's functionality was verified using external ROM. The ROM code error was fixed with one mask change. ■

ABOUT THE AUTHORS

TERENCE E. CRIMMINS *is an electronic development engineer at MIT Development Corp., a leading designer and producer of medical diagnostic and monitoring systems. With MDC for the last two years, he is the project leader for an upcoming line of medical diagnostic devices. He holds his BSEE from Cornell University and is actively pursuing a master's degree in physics from the University of Connecticut.*

ZAHEER HASSAN *is the marketing manager for custom products at Sierra Semiconductor. He has nine years' experience in semiconductor design and applications and in ASIC marketing and sales. He joined Sierra two years ago from Zymos Corp., where he likewise was marketing manager for custom products. Hassan holds a BS in electronics engineering from the University of Karachi, Pakistan, and an MS in electrical and computer engineering from Oregon State University.*

CREATING

a Custom HDLC Chip

When designing a chip to meet a system requirement, designers face a mixed bag of often contradictory cost, density, performance, and time-to-market requirements. To satisfy them, designers must choose among a wide range of design approaches and methodologies.

AT&T was faced with these choices in its design of an HDLC chip for its 5ESS switch, a telephone line switching network that routes telephone calls in a central office environment. The application for the chip was in a packet-

switching unit (PSU) that was being designed to adapt the 5ESS switch for ISDN switching.

The 5ESS switch receives multiple ISDN channels by way of any of the standard interfaces of the Integrated Services Digital Network (ISDN), such as the four-wire T and two-wire U interfaces. ISDN channel information (consisting of standard channels such as the 16-kb/s D, 64-kb/s B, and 384-kb/s HO channels) is passed along to the integrated services line unit (ISLU), which separates voice data from D and B channel data packets. Voice data is routed to the standard 5ESS switch's circuit structure, which handles it just as it would any other voice channel.

D channel and B channel data

packets are routed to the packet-switching unit. Within the PSU, each ISDN channel (less the voice information) is routed to a network of protocol handlers, which implement OSI levels 2 and 3 for such standards as X.25 and X.75. Dynamic load balancing is employed so that the protocol processing load is distributed evenly among the protocol handlers.

The protocol handler terminates the protocol where appropriate and routes the packet to the proper destination. The outputs of the various protocol handlers feed a common packet bus, which serves as an embedded local-area network (LAN). This LAN provides a backbone for the distribution of packets among the protocol han-

dlers, some of which provide access to other parts of the 5ESS switch.

■ **CHIP REQUIREMENTS**

The chip that AT&T needed for the protocol handler had to meet several requirements. First, it had to perform HDLC conversion for incoming bit streams of various widths. This entailed handling part of the level 2 protocol, as well as handling frame buffering and error checking.

Second, it had to support an asynchronous interface with the Motorola 68020 and 68030 microprocessors. Third, it had to possess a large FIFO for frame buffering, necessary for enabling the system to efficiently handle a packet bus traffic phenomenon known as burstiness.

Burstiness is a situation of uneven traffic on the packet bus. Statistically, the number of packets going to and coming from each protocol handler is even. However, under some circumstances (for example, at a certain time of the day, multiple remote terminals may be sending data to the same mainframe host), the number of packets destined for a particular protocol handler may be higher than usual. In this situation, the protocol handler must be able to
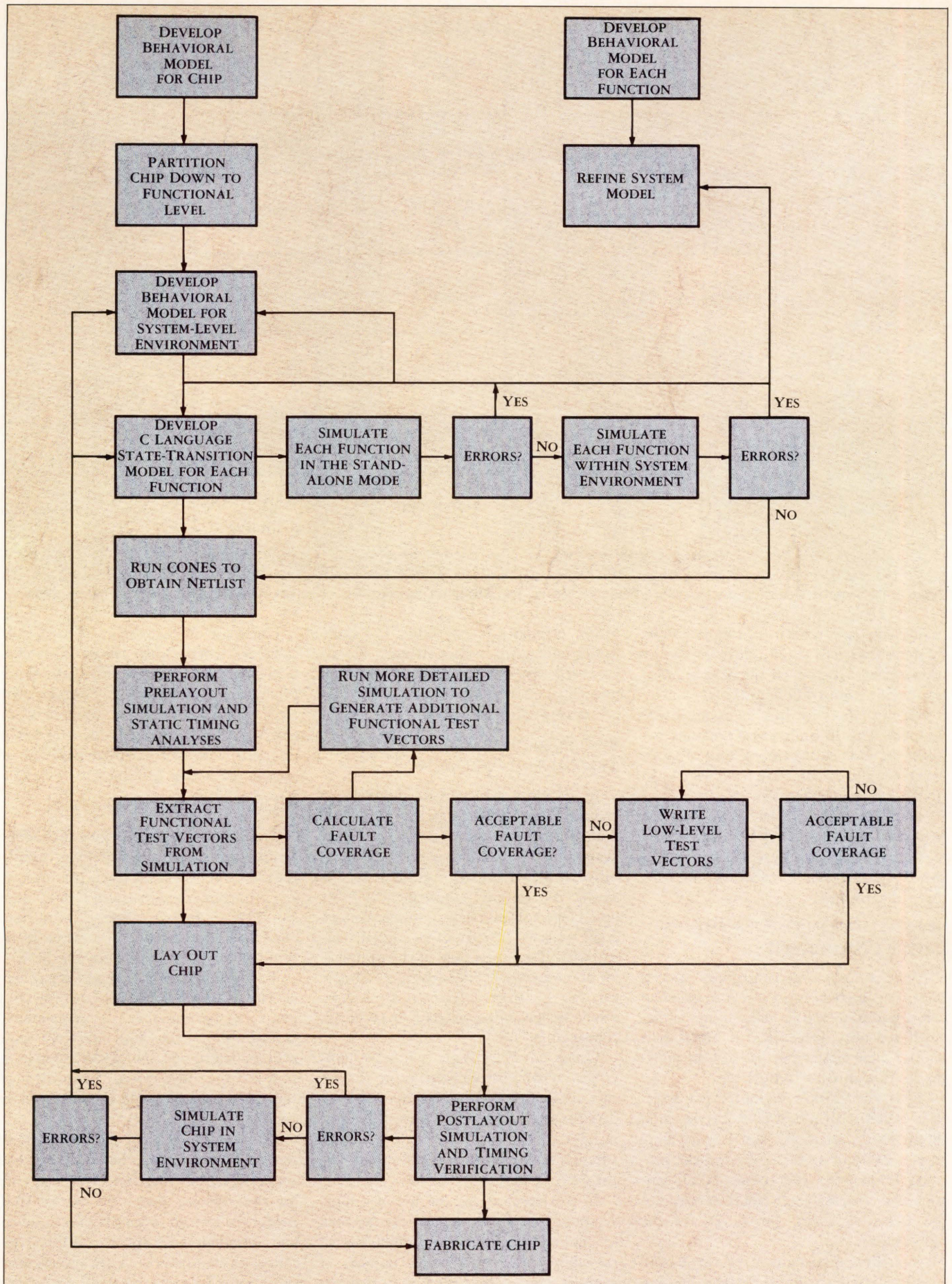
Figure 1. In the top-down design methodology chosen, the HDLC chip was first specified behaviorally and then partitioned at the functional level. A C description of each function (state transition model) was written to enable CONES to synthesize the logic using a standard-cell library.
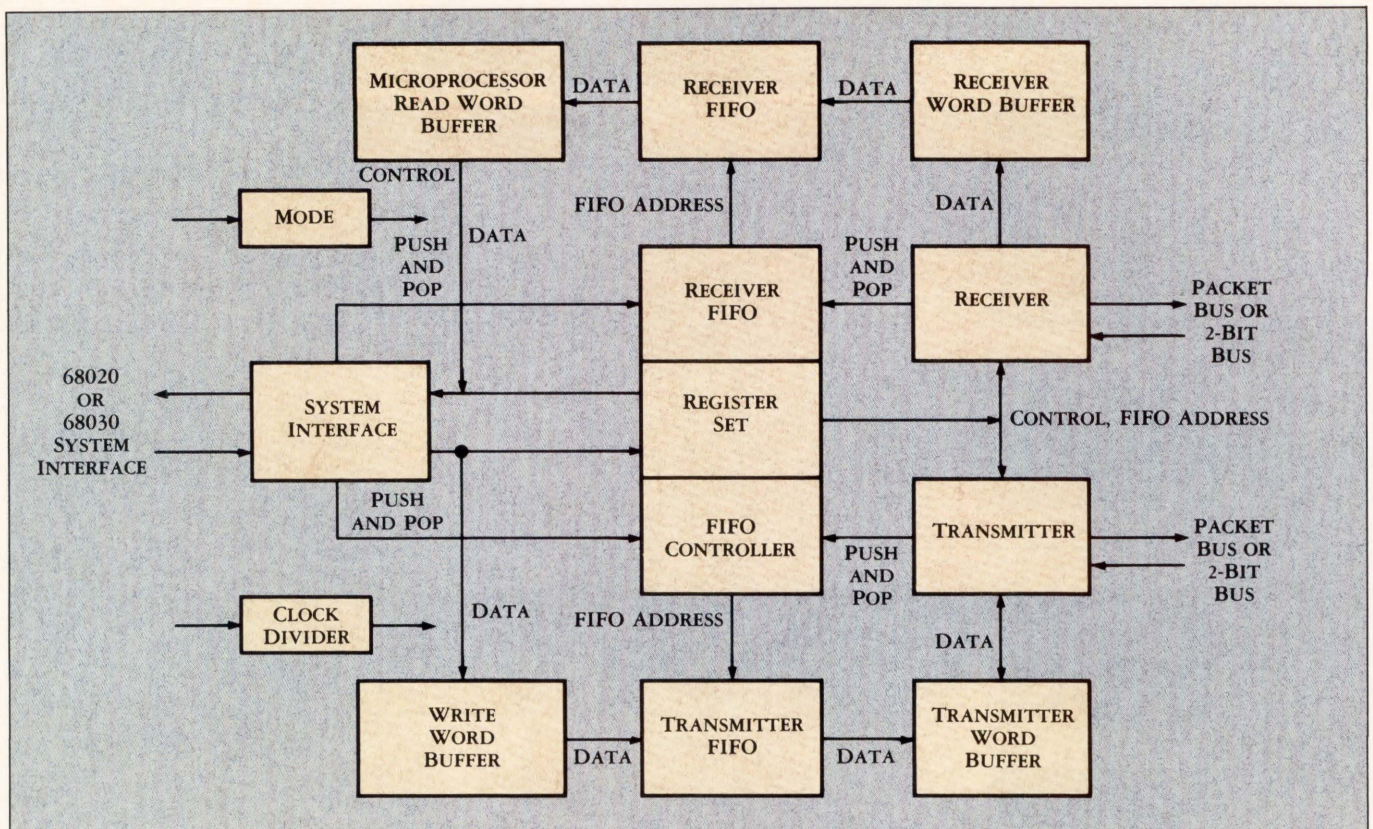
Figure 2. The packet bus media access controller (PBMAC), designed for the 5ESS switch, performs HDLC conversion for incoming bit streams of various widths.

buffer the data; otherwise, the data must be retransmitted, significantly degrading performance. Based on queuing studies, we decided on 4-kilobyte FIFOs.

Finally, the chip had to be unusually flexible. This flexibility was necessary because in addition to taking care of HDLC conversion in the protocol handlers, it had to be adaptable for other HDLC applications in other parts of the 5ESS switch. In addition, it had to be general-purpose enough to handle future HDLC applications whose specific requirements had not even been determined.

■ EVALUATION OF OFF-THE-SHELF COMPONENTS

Given the tight time constraints, AT&T's first step in satisfying the requirements of the application was to look for an off-the-shelf solution. Intel's 82586 HDLC controller came closest but had several drawbacks that made it unsuitable.

First, the 82586's microprocessor interface is designed specifically for the Intel 8086/80286 microprocessor family. Since the controlling processor used in the protocol handler was a Motorola 68020, considerable interface logic would have had to be added to adapt the 82586 for the application.

A second disadvantage of the 82586 was the lack of a sufficiently large on-chip FIFO, needed to effectively handle burstiness. Finally, the Intel 82586 lacked flexi-

bility, primarily in its ability to handle multiple data input data streams. For all three reasons, AT&T decided to design its own chip, known as the packet bus media access controller (PBMAC).

■ STANDARD–CELL AND LOGIC SYNTHESIS APPROACH

In deciding which ASIC approach to choose, AT&T had to weigh the conflicting requirements of fast time to market against the need for a large FIFO and flexibility, which required high density. To satisfy the density requirements, AT&T opted for a standard-cell approach, using a fairly typical 1.25-μm CMOS digital standard-cell library. Because of the complexity of the state-machine design, AT&T opted for a logic synthesis design methodology, often referred to as silicon compilation.

Using an AT&T logic synthesis tool, CONES, rather than implementing the design at the gate level, we were able to describe the chip at a functional level using C. From a C description, CONES automatically generates an optimized netlist (see "CONES: An Overview," p. 82).

In this application, the use of a logic synthesis tool did more than cut the time to market. Because of the complexity of the chip and the enormous number of states required to give the chip its flexibility, we believe the design would have been impractical without the use of such a tool.

One trade-off that designers have typically had to make when using tools such as CONES has been design efficiency versus productivity. With CONES, however, that trade-off isn't necessary. In most cases, in fact, the density and performance of a chip designed with CONES either matches or exceeds what can be achieved by hand.

■ DESIGN IMPLEMENTATION

The overall design strategy was to adopt a top-down design methodology, specify the chip at a behavioral level, partition it, and refine it at the functional level (Figure 1). The C functional/state description, once specified at the bit/register level, could be given to CONES for synthesis.

To implement this methodology, models had to be developed for both the chip and the system-level environment, through which the chip interacted with the remainder of the packet-switching unit and the 5ESS switch. Refinement of these two models would then proceed in parallel.

Initially, the bulk of the design effort was spent developing the model for the system-level environment. This was true for two reasons. First, the system-level model was needed to supply the stimulus- and response-checking capabilities required to simulate the chip's system-level operation. Second, much of the chip's functionality was initially incorporated into the system-level environment so that

CONES: An Overview

CONES is a menu-driven logic synthesis tool that AT&T uses internally to design many of its ASICs. Unlike many commercially available tools, which can synthesize logic only from a sum-of-products description, CONES can synthesize optimized logic using a C language functional description that specifies the circuit's state transitions at the bit/register level.

CONES uses a three-step process to generate a netlist. In the first step, it takes a description written in C and generates a Boolean equation (sum of products) for each bit in the circuit (output or flip-flop) description. Next, it generates a netlist. If the target design is to be implemented with standard cells, CONES develops the netlist using an optimal combination of the logic elements contained in the standard-cell library. If the target is one or more PLDs, CONES partitions the design and generates a netlist using an optimal combination of PLDs (Munoz and Stroud, 1987).

In the final stage, whether the target device is a standard-cell chip or a PLD, CONES looks for common circuitry (for example, each bit in a register may have decoding and chip-selection circuitry in common), forms a single occurrence of this circuitry, and updates the netlist.

Though CONES uses a generous portion of the C language, designers don't have access to the entire language. As a result, CONES uses audits that check for illegal usage. These audits also flag syntax errors and common C programming errors, like the use of double equals signs and the specification of out-of-range array elements.

CONES also performs some preliminary timing analysis, estimating critical-path performance and identifying obvious race conditions. It can also automatically generate built-in self-test circuitry, after which, it updates the netlist.

Once the layout process is completed, CONES can extract a final netlist based on the final layout information. It can even generate a schematic. Typically, however, a C language specification of the circuit and the netlist is deemed adequate documentation for most designs. ∎

REFERENCE

MUNOZ, R.R., AND C.E. STROUD. OCTOBER 1987. "Automatic Partitioning of Programmable Logic Devices," *VLSI Systems Design*.

---

portions of the chip could be simulated without having access to completed models for the remainder of the chip.

For example, in order to test the receiver's operation, a portion of the microprocessor control circuitry was needed. Rather than build a complete model for this control circuitry, a more abstract model was built into the system environment that included only those functions that were needed to test the receiver.

Later in the design cycle, as the control circuitry was defined, those portions of the system environment that modeled the control circuitry could be eliminated. Generally speaking, as the design cycle progressed, functionality was transferred from the system environment to the chip, thereby increasing the complexity of the latter.

■ REDUCED DESIGN CYCLE AND SIMPLIFIED DEBUGGING

Incorporating models for portions of the chip within the system environment early in the design cycle provided several advantages. First, since the model used in the system environment was less complex (because it was abstract), it could be built more quickly, thereby enabling designers to simulate and debug the chip design more quickly.

Second, incorporating models for portions of the chip within the system environment made the design easier to control and observe, because circuitry that's not necessary to the portion of the circuit being simulated need not be included in the model, enabling the circuit to be be driven and monitored more directly.

Third, it simplified debugging. Because the interface circuitry can be modeled abstractly, the chance of bugs occurring in that circuitry is reduced. Therefore designers can concentrate on debugging the major circuit blocks, rather than the interface circuitry.

■ MENU-DRIVEN SYSTEM INTERFACE

To make it easier to stimulate and observe the design, a menu-driven interface was built into the drivers and monitors. Through this interface, designers could specify the stimulus to the circuit and its expected response using high-level commands. For example, to get the 68020/68030 to write to one of the PBMAC's control registers, the designer simply specifies the write operation and the destination address. The driver handles all of the operation actually needed to effect the transfer.

At a higher level, designers can specify the transmission and reception of packets. They can even write C application programs that are comparable to the programs that software developers might write for the 5ESS switch in its target environment. The 68020/68030 emulator uses function calls to translate these programs into the read and write operations needed to communicate with the PBMAC.

■ PARTITIONING THE DESIGN

Spending more time up front developing the system environment greatly simplified the top-down design of the PBMAC. In refining the PBMAC design, the first step was to partition the behavioral model into functional blocks. The key functional units are the transmitter, receiver, transmitter and receiver FIFOs, and system interface (Figure 2). The transmitter and receiver each contain the HDLC conversion and error-checking circuits.

The on-chip FIFO provides intermediate storage for frames between system RAM and and the packet bus or the 2-bit ("bibble") bus. Both the transmitter and receiver FIFOs are implemented with $1K \times 32$-bit static RAMs operating at a maximum frequency of 9 MHz. Pipeline registers are used in the data path in both directions to minimize PBMAC access times.

While receiving a frame, the receiver pushes the data and frame information into the receiver FIFO. After the frame has been received, the system processor pops them out. In the transmission direction, the system processor pushes the frame information word and data into the transmitter FIFO. The transmitter then pops them out and sends them.

The access contention for each FIFO is resolved by an on-chip arbiter, which gives the receiver and transmitter priority access to the FIFOs.

```
                    /* select proper outs based on mode */
                    if (in->MODE0 == PACKET)
                    (
                        if (in->FTTM == 0)
                        (
                            if (in->FTS == 0)
                            (
                                XFER (outs, 0, 17, in->I__ON0, 0)
                            )
                        )
                    )

                    out->WAIT = outs[17];
                    /* need to reverse order */
                    out->TAN[0] = outs[13];
                    out->TAN[1] = outs[12];
                    out->TAN[2] = outs[11];
                    out->TAN[3] = outs[10];

                    /* store ones, save, and shift */
                    XFER (state->XONES, 0, 2, outs, 7)
                    XFER (state->XSAVE, 0, 3, outs, 3)
                    XFER (state->XSHIFT, 0, 2, outs, 14)

FD1S3AX: TAN0, (TAN0__P,BCLK,BCLK),TAN0,TAN0__N);
AOI332: TAN1__P, (FTS__N,XSMEN__N,C2G2,FTS,XSMEN__N,C2G3,TAN1__N,
        XSMEN), TAN1__P;
AOI332; C2G3, (MODE0__N,FTTM__N,ONG12,MODE0,FTTM__N, ON212,FTTM,
        ON312), C2G3;
AOI3333: C2G2, (MODE0__N,FTTM,ON412,MODE0__N, FTTM__N, ON412,MODE0,
        FTTM__N,ON012,MODE0,FTTM,ON112), C2G2;
FD1S3AX: TAN1, (TAN1__P,BCLK,BCLK), (TAN1,TAN__N);
AOI332: TAN2__P, (FTS__N,XSMEN__N,C3G2,FTS,XSMEN__N,C3G3,TAN2__N,
        XSMEN), TAN2__P;
AOI332: C3G3, (MODE0__N,FTTM__N,ON611,MODE0,FTTM__N,ON211,FTTM,
        ON311), C3G3;
AOI3333: C3G2, (MODE0__N,FTTM__N,ON411,MODE0__N, FTTM__N,ON411,MODE0,
        FTTM__N, ON011,MODE0,FTTM,ON111), C3G2;
FD1S3AX: TAN2, (TAN__P,BCLK,BCLK), (TAN2,TAN2__N);
AOI332: TAN3__P, (FTS__N,XSMEN__N,C4G2,FTS,XSMEN__N,C4G3,TAN3__N,
        XSMEN), TAN3__P;
AOI332: C4G3, (MODE0__N,FTTM__N,ON6 1,MODE0, FTTM__N,ON210,FTTM,
        ON310), C4G3;
AOI3333: C4G2, (MODE0__N,FTTM,ON410,MODE0__N, FTTM__N,ON410,MODE0,

        FTTM__N,ON010,MODE0,FTTM,ON110), C4G2;
FD1S3AX: TAN3, (TAN3__P,BCLK,BCLK), (TAN3, TAN3__N);
AOI332: WAIT__P, (FTS__N,XSMEN__N,C5G2,FTS,XSMEN__N,C5G3,WAIT__N,
        XSMEN), WAIT P;
AOI332: C5G3, (MODE0__N,FTTM__N,ON617,MODE0,FTTM__N,ON217,FTTM,
        ON317), C5G3;
AOI3333: C5G2, (MODE0__N,FTTM,ON517,MODE0__N, FTTM__N,ON417,MODE0,
        FTTM__N,ON017,MODE0,FTTM,ON117), C5G2;

TAN0 = !XSMEN * !MODE0 * FTS * !FTTM * ON613\
     + !XSMEN * !MODE0 * !FTS * FTTM * ON513\
     + !XSMEN * MODE0 * FTS * !FTTM * ON213\
     + !XSMEN * MODE0 * !FTS * FTTM * ON113\
     + !XSMEN * !MODE0 * !FTS * !FTTM * ON413\
     + !XSMEN * MODE0 * !FTS * !FTTM * ON013\
     + !XSMEN * FTS * FTTM * ON313\
     + XSMEN * TAN0;
TAN1 = !XSMEN * !MODE0 * FTS * !FTTM * ON612\
     + !XSMEN * !MODE0 * !IFTS * FTTM * ON512\
     + !XSMEN * MODE0 * FTS * !FTTM * ON212\
     + !XSMEN * MODE0 * !FTS * FTTM * ON112\
     + !XSMEN * !MODE0 * !FTS * !FTTM * ON412\
     + !XSMEN * MODE0 * !FTS * !FTTM * ON012\
     + !XSMEN * FTS * FTTM * ON312\
     + XSMEN * TAN1;
TAN2 = !XSMEN * !MODE0 * FTS * !FTTM * ON611\
     + !XSMEN * !MODE0 * !FTS * FTTM * ON511\
     + !XSMEN * MODE0 * FTS * !FTTM * ON211\
     + !XSMEN * MODE0 * !FTS * FTTM * ON111\
     + !XSMEN * !MODE0 * !FTS * !FTTM * ON411\
     + !XSMEN * MODE0 * !FTS * !FTTM * ON011\
     + !XSMEN * FTS * FTTM * ON311\
     + XSMEN * TAN2;
TAN3 = !XSMEN * !MODE0 * FTS * !FTTM * ON610\
     + !XSMEN * !MODE0 * !FTS * FTTM * ON510\
     + !XSMEN * MODE0 * FTS * !FTTM = ON210\
     + !XSMEN * MODE0 * !FTS * FTTM * ON110\
     + !XSMEN * !MODE0 * !FTS * !FTTM * ON410\
     + !XSMEN * MODE0 * !FTS * !FTTM * ON010\
     + !XSMEN * FTS * FTTM * ON310\
     + XSMEN * TAN3;
```

[A]
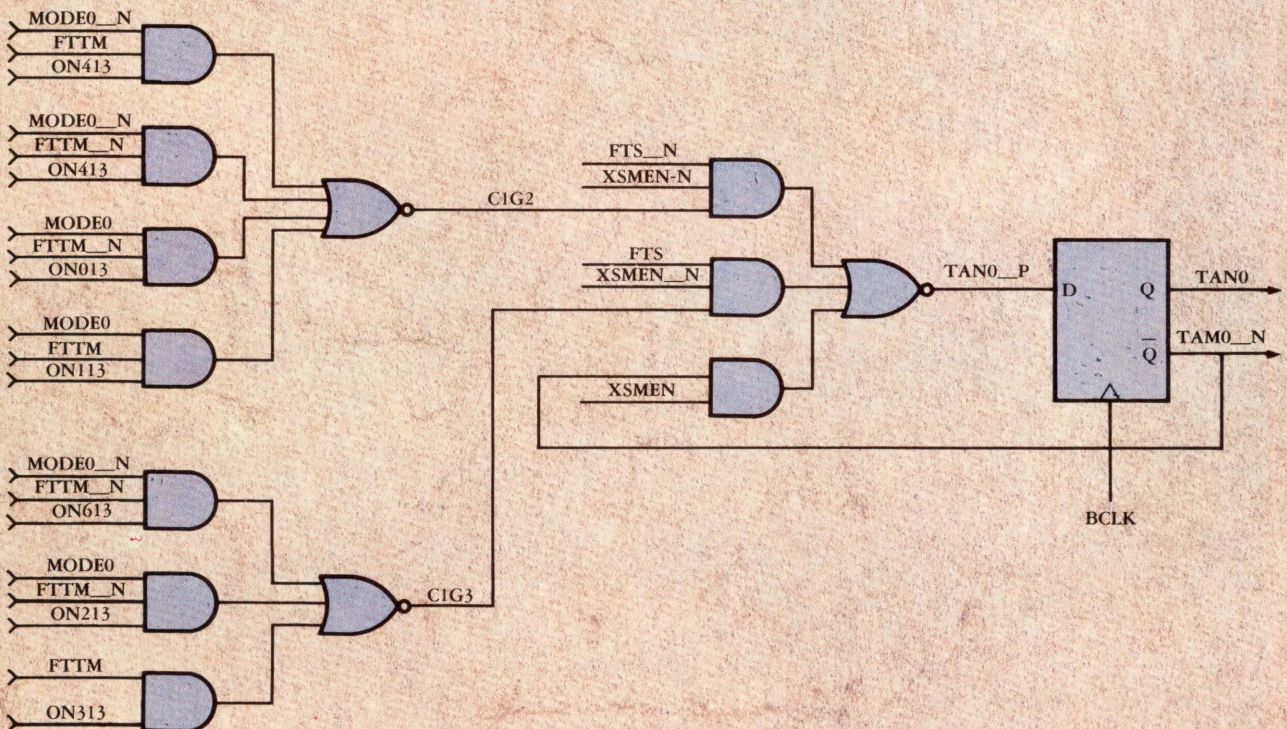
[B]

Figure 3. This sample specification describes functional description (A) specifies the conditions, when the PBMAC is in the packet mode, under which an 18-bit word will be transferred to one of four temporary arrays (TAN0–TAN3). In synthesizing the logic, CONES saw that ON513 was redundant and eliminated it (B).
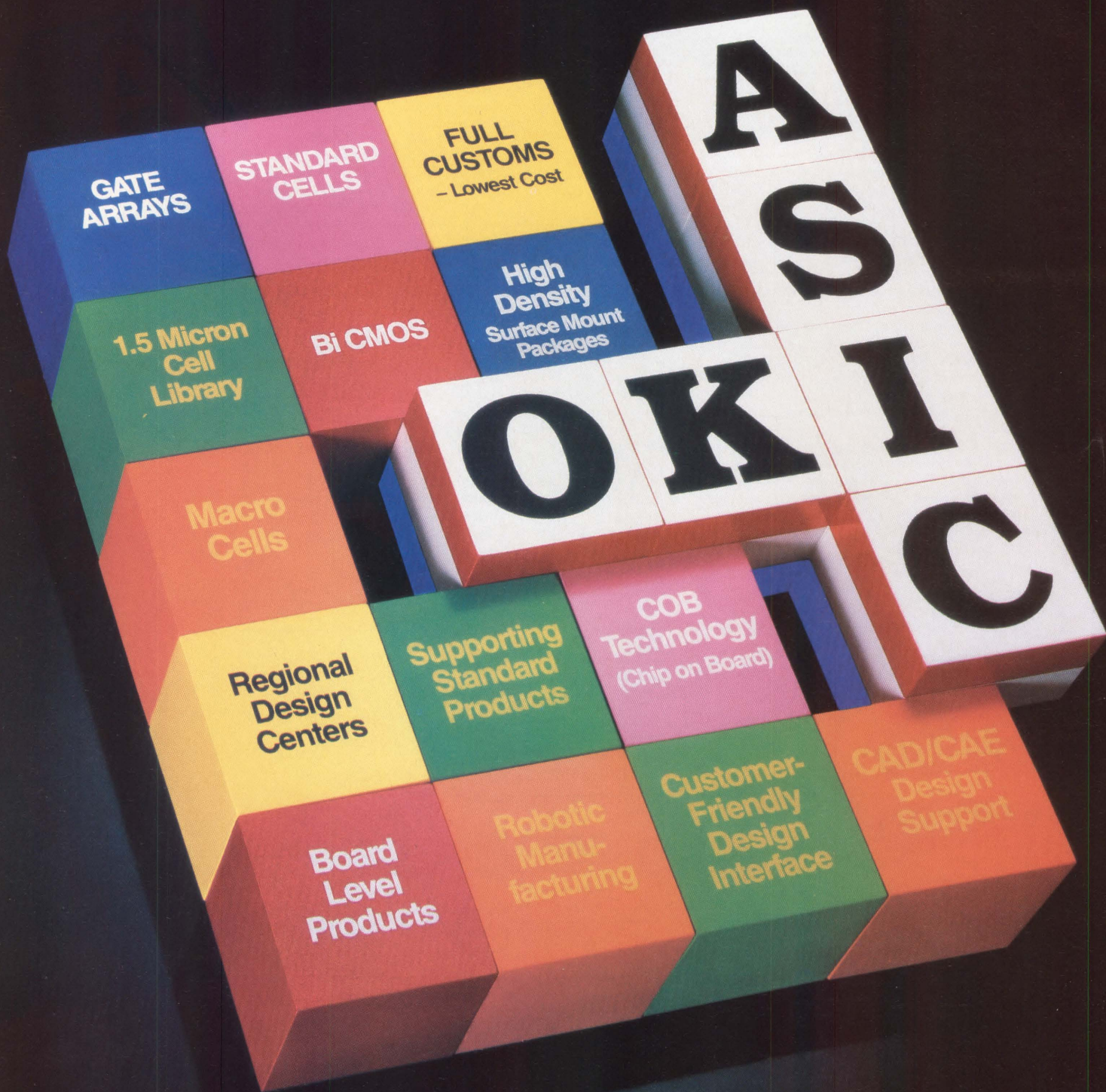
Only from OKI:

- ## Most **complete** ASIC building blocks.
- ## Most **versatile** design/package options.
- ## Most **experienced** ASIC technology.

Nobody but nobody puts ASIC technology together like OKI Semiconductor can.

Ease into ASIC with OKI as your close working partner—and you instantly support your VLSI application with the most comprehensive ASIC capabilities on the world market today. Bar none.

From gate array, standard cell and full custom chips to standard components to integration to advanced board level products, OKI alone puts you on the leading edge of ASIC technology and its complete implementation.

OKI: the totally logical choice.

Opt for OKI ASIC, and you open up your options across the board. Only OKI now offers the system designer the unique security and entry ease that only a proven track record in CMOS ASIC problem-solving can provide. This history of performance built up since 1977 has produced the widest range of solid building blocks yet: advanced ASIC products and packaging including surface-mount, backed up with the most flexible cell libraries, CAD/CAE design tools and development aids.

As your working partner, OKI ASIC expertise is available to you at any stage of the development process. We'll help you define system requirements, determine the most cost-effective product solutions and supply complete design software—accessible at your own workstation or through our regional design centers. And then we take it from there: with high volume fabrication, assembly and testing completed in one of the world's most highly robotized manufacturing facilities.

Compare
Total ASIC Capabilities

	OKI ASIC	Source "B"	Source "C"
Gate Arrays to 10K Gates	●	●	
Standard Cells to 30K Gates	●	●	
Full Customs – Lowest Cost	●		
1.5 Micron Cell Library	●	●	●
Macro Cells	●	●	
Bi CMOS	●		
High Density Surface Mount Packages	●		
Board Level Products	●		
Supporting Standard Products	●		
COB Technology (Chip on Board)	●		
CAD/CAE Design Support	●	●	●
Customer-Friendly Design Interface	●	●	●
Regional Design Centers	●	●	●
Robotic Manufacturing	●		

ASIC Solutions from **OKI:**
You can't beat the logic!

Check out OKI ASIC data: VLSI 1788

Please rush complete technical data/specs on OKI capabilities in:

( ) Gate Arrays
( ) Standard Cells
( ) Full Customs

( ) Please call: we have immediate requirements:

Name/Title _____

Company _____

Attach coupon to business card or letterhead and return to: ASIC Customer Service, **OKI Semiconductor,** 650 North Mary Avenue, Sunnyvale, CA 94086. Tel: (408) 720-1900.

CIRCLE NUMBER 18

The chip's system interface consists of a 68020/68030-compatible 32-bit interface and a packet bus interface (a function of the transmitter and receiver). The PBMAC interfaces with the system bus through a 32-bit data bus and a set of internal registers. It acts as a slave device, relying on the 68020/68030 to supply the driving signals. The 68020/68030 initializes the PBMAC through 32-bit internal registers.

The PBMAC accepts the traditional system bus interface signals. These are Chip Select ($\overline{CS}$), Read/Write ($\overline{RW}$), Address Strobe ($\overline{AS}$), Data Strobe ($\overline{DS}$), and Data Transfer Acknowledge (DTACK).

For the interface between the PBMAC and the packet bus, handshaking signals (Request to Send and Clear to Send) interface with the bus arbitration circuit and control the flow of data out onto the packet bus. At any given time, only one protocol handler will be transmitting onto the packet bus, with all other protocol handlers listening. Each protocol handler watches for an address match before accepting a packet.

After partitioning the design at the functional block level, the next step was to write a behavioral-level description for each block. The behavioral model was then refined to the functional level. Written in C at the bit/register level, the functional model provides a state transition description of the function. Essentially a C function call, it accepts as input a specification of the present state and current inputs and describes the resulting outputs and the next state.

Figure 3 shows a sample specification and a schematic drawing of the resulting logic. The functional description describes the conditions, when the PBMAC is in the packet mode, under which an 18-bit word will be transferred to one of four temporary arrays (TAN0–TAN3).

Once a functional description was completed for the PBMAC, the model for each function was simulated in a stand-alone mode to verify its operation. Then, to verify its system-level operation, it was substituted back into the system model.

■ LOGIC SYNTHESIS TOOL HANDLES LOGIC DESIGN

At the point at which conventional top-down design would have progressed to the component and gate level, once a functional model was completed for each section of the chip, CONES was invoked to complete the logic design. Its use not only sped logic design, but also avoided design errors, making it possible to obtain functionally correct logic on the first pass.

Meanwhile the two FIFOs were implemented with a macrocell. The macrocell

was created using a RAM generator that works in conjunction with the standard-cell library.

After running CONES and obtaining a netlist, regression testing was used to verify the gate-level operation of the circuit. Since the boundary between the chip and the system environment was implemented as a detailed wire list, the gate-level model for the chip could be plugged directly into the system-level model.

To reduce simulation time, the bulk of the regression testing was performed for only one portion of the PBMAC at a time. Most of the chip was simulated at the functional level. Once all the individual functions were verified at the gate level, regression testing was performed for the entire chip.

■ DEVELOP TEST VECTORS

Once regression testing was completed and the logical operation of the chip was verified, the next step was to generate test vectors. Here, considerable time was saved by generating vectors at the functional level. Rather than writing test vectors from scratch, the strategy was to stimulate the chip using high-level commands such as Read/Write a Packet and Read/Write a Control Register. Since the circuit was known to be logically correct, the circuit's response (outputs) to this high-level stimulus could be captured as test vectors.

Once we had derived a valid set of functional test vectors, the next step was to compile the netlist and vectors and send them to the Zycad fault simulation engine for fault coverage calculation. Although functional test vectors verify a chip's functional integrity, fault coverage must be performed to determine how effective these vectors are in exercising all of the chip's nodes. For those portions of the chip that aren't being exercised, designers must either derive additional functional vectors or write low-level test vectors from scratch.

One technique that proved useful in exercising those portions of the circuitry that weren't being exercised with the functional vectors was taking advantage of the chip's built-in self-test circuitry (BIST). BIST verifies chip functionality by using a technique known as signature compression and analysis.

The basic strategy is to generate a pseudorandom test pattern using a register near the chip's input. The pattern that's generated in this register, known as a linear feedback shift register, is used to stimulate the remainder of the chip for a certain period of time. The results of this stimulus are collected and compressed in another linear feedback shift register near

the chip's output. The resulting "signature" is compared with a known good signature to verify the chip's operation.

■ STATIC TIMING ANALYSIS

Once we obtained acceptable fault coverage, the next step was to perform timing verification. Static timing analysis, also known as path delay analysis, was performed first. This analysis, which is done without the use of test vectors, allows designers to locate all circuit paths in a synchronous circuit that function too slowly to meet the required operating frequency. It is done by counting the number of gate delays in each path and factoring in loading due to fan-in, fan-out, and estimated interconnection capacitance. In this phase, CONES automatically beefs up critical paths where necessary (adds more powerful drivers, reduces load capacitance, and so on).

Once layout was completed, final timing verification was performed using actual timing values that were extracted from the layout data. After verifying timing, the completed model was substituted back into the system environment to perform final regression testing.

The goal of any CAD methodology is to achieve success on first silicon as quickly as possible. First-pass success was possible for the PBMAC because of the solid design practices that CONES follows in generating logic. The key is that CONES performs only synchronous, single-clock design.

Although functionally correct, asynchronous circuits that use multiple clocks, often don't behave as intended when fabricated over the full range of process conditions and operated over the full rated range of temperature and power supply voltages. Once fabricated, such circuits may exhibit a variety of race conditions, glitches, and functional errors. Testability may also be impaired.

By using single-clock, synchronous design, CONES is able to design logic that's generally free of race and timing problems and easier to write test vectors for. Even for a chip as complex as the PBMAC, the use of CONES, in conjunction with a top-down methodology, made first-pass success a reality. ■

ABOUT THE AUTHOR

RALPH WILSON *is supervisor of the 5ESS switch packet processing design group at AT&T. He has worked at AT&T for 10 years, where he has been involved in a wide variety of circuit design projects. He holds a BSEE from the University of Montana and an MSEE from Stanford University. His hobbies include hunting, fishing, and volleyball.*

CIRCLE NUMBER 20

USER-*Programmable Chips*
Take On a Broader Range
of Applications

DAVID SMITH, WESTERN REGIONAL EDITOR

Programmable chips, comprising PLDs and programmable gate arrays, can implement the logic needed by a designer when he can't find an off-the-shelf solution. Perhaps because of its humble beginnings in the 20-pin PAL and the FPLA, PLDs have a reputation as collectors of spare logic gates and as small-scale custom sequencers. But the increasing complexity and varying architecture of the devices make them suitable for much more complex, diverse and (sometimes) downright weird applications.

■ **INCREASING CAPACITY AND NEW ARCHITECTURE FIT THEM FOR MORE COMPLEX USES** ■

The logic arrays found in most PLDs can implement many if not most common logic functions. In fact, National Semiconductor Corp. (Santa Clara, Calif.) offers a line of standard parts, called the National Masked Logic (NML) series, that are metal-mask–programmed versions of PLD dice. The series includes multiplexers, counters, and shift registers.

As discussed in the article "Programmable Logic Overview" in last October's *VLSI Systems Design* (Meyer, 1987), three trends are prevalent within the architecture of PLDs: improvement on standard architectures, hybridization of PLDs and standard parts, and the abandonment of PLD architectures to one more closely resembling that of gate arrays.

Within each of these architectural trends, the addition of flexible register macrocells, higher numbers of pins, and greater circuit density, results in PLDs that can implement much more complex circuits than can the NML series. Discussions with PLD manufacturers and users uncovers applications as complex as Micro Channel interface controllers and serial communications subsystems.

■ SAME LOGIC CORE, MORE INPUTS

The basis of most PLDs, the AND-OR logic plane, can potentially implement any Boolean expression of its inputs. Because newer PLDs can replace older PLDs, without redesign, to decrease power consumption or increase performance, immediate improvements in system performance are possible without exploring all the capabilities of PLDs.

The possibilities for PLD logic have hardly been exhausted, however. In a somewhat fanciful example, Intel Corp. (Folsom, Calif.) used one of its EPLDs, the 5C060, to implement a rudimentary A/D converter (Figure 1). The eight data outputs, D_0–D_7, are connected through a resistor network to provide feedback to a voltage comparator on the input. The EPLD's logic plane implements a successive-approximation algorithm, setting each bit from most significant to least significant.

After all the outputs are cleared on power-up, an on-chip state machine controls the approximation as follows: In the first state, the most significant bit is set high, which, through the resistor network, raises the reference voltage to about half the supply voltage. In the next state, if the output of the comparator is low, the reference voltage is higher than the input, so that D_7 is reset; if the output of the comparator is high, D_7 is left high. In either case, the next state sets D_6 high and repeats
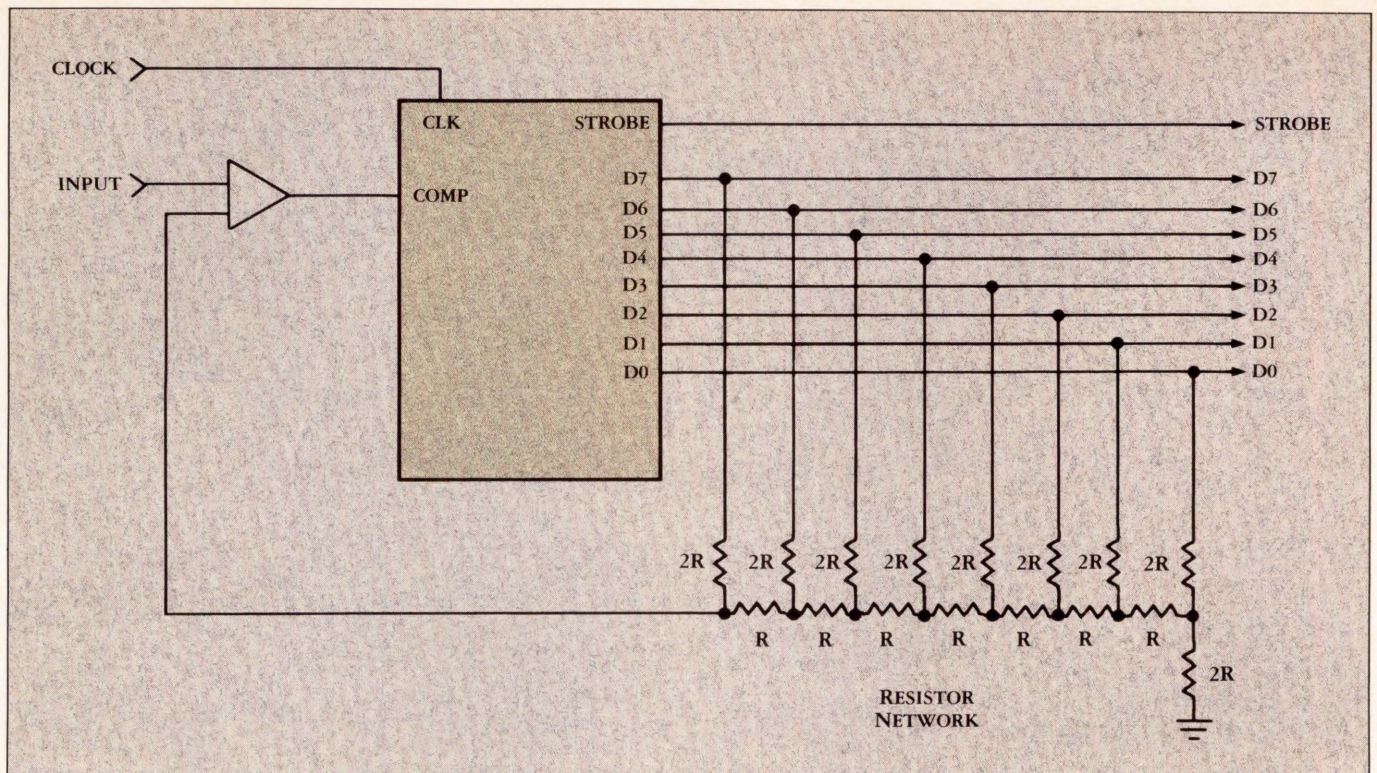
Figure 1. A PLD-based analog-to-digital converter demonstrates how Boolean-based PLDs can implement nondigital functions.

the comparison process, and so on.

The complete approximation takes eight cycles; a ninth cycle is included for latching the results and a tenth cycle resets all the registers. This circuit—used in an Intel demonstration application—is probably not as economical as commercial A/D converters. It does, however, show how a PLD's Boolean logic can be used to implement nondigital circuitry.

In another example, a simple PLD from Signetics was used by William Freeman of Raynet Corp. (Menlo Park, Calif.) to build a Walsh function generator. A Walsh function is a discrete Fourier transform that is used in the design of optical communications links. Combining the Walsh function generator and a correlator circuit enables the link's detection circuitry to pick one conversation out from a multichannel optical transmission medium. Freeman designed the generator by describing the Walsh functions, based on a power of two, in Boolean logic. Such applications are not typically associated with PLD AND-OR planes.

Around the logic plane, PLDs are gaining more I/O pins, allowing them to implement functions with large input requirements. In particular, because of its programmable I/O cells, a 24-pin GAL from Lattice Semiconductor Corp. (Beaverton, Ore.) can implement a 16-line priority encoder, a useful circuit in any system with, for example, a variety of sources of interrupts. The encoder provides 16 inputs, 4 output pins with the hexadecimal

representation of which input is strobed, and an output that indicates if any of the input lines have gone high.

Large numbers of inputs enable some newer PLDs to work in a 32-bit environment. For example, the PLHS501 Programmable Macro Logic chip from Signetics Corp. (Sunnyvale, Calif.) can implement any gating function of 31 inputs. In a 32-bit system, for example, it can be used to identify addresses to the resolution of any bit pair. Such capability is useful for communications in systems with multiple processors.

To illustrate this application, Topologic Inc. (Denver, Colo.) builds a parallel-processing board that plugs into the VMEbus in Sun Microsystems workstations. The company used PLHS501s to implement control logic in the interface between the buses. Basically, the company used it as a decoder with some internal RS latches, to implement address decoding, interrupt-level comparison, and interrupt service cycle circuitry. Five PLDs are used on board: one in conjunction with a Signetics 68172 VMEbus controller and one next to each of the four processors on the board to implement communications among the processors.

■ FLEXIBLE REGISTERS WITH LOGIC CONTROL

Building more flexible registers into PLDs and allowing them to be decoupled from outputs—"buried"—increases the applicability of PLDs as well. For example,

a 20-pin GAL16V8 can implement a 6-bit shift register. The design is cascadable, can shift left or right, and has parallel load and store. The logic plane of the PLD implements control logic and the multiplexing to implement the shift function; the output cells implement the register with D-type flip-flops. Such capability can also be used to implement a registered multiplexer; a dual 7:1 multiplexer, for example, fits in a 24-pin GAL20V8.

An 8-bit barrel shifter can let any system perform a variety of bit manipulations and arithmetic functions. Such a shifter can be implemented in a PLHS501 or a 24-pin GAL20V8. Either implementation demonstrates a brute-force implementation of the shifter. Each of the inputs passes through the equivalent of an eight-input multiplexer to each of the outputs. The degree of shift is determined by a 3-bit shift input. The circuit uses all 72 internal fold-back NAND gates in the PLHS501. Variations on this circuit could be mirror imaging (bit reversal) and byte swapping. In addition, the shift register can be designed with or without registered outputs, thereby fitting more seamlessly into an existing data path.

The SAM programmable microsequencer from Altera Corp. (Santa Clara, Calif.) has a bank of registers that can be used under control of a microprogram in the part's EPROM memory. One company (which asked to remain anonymous) used the registers and an on-chip counter in a satellite application—a deep-space probe for
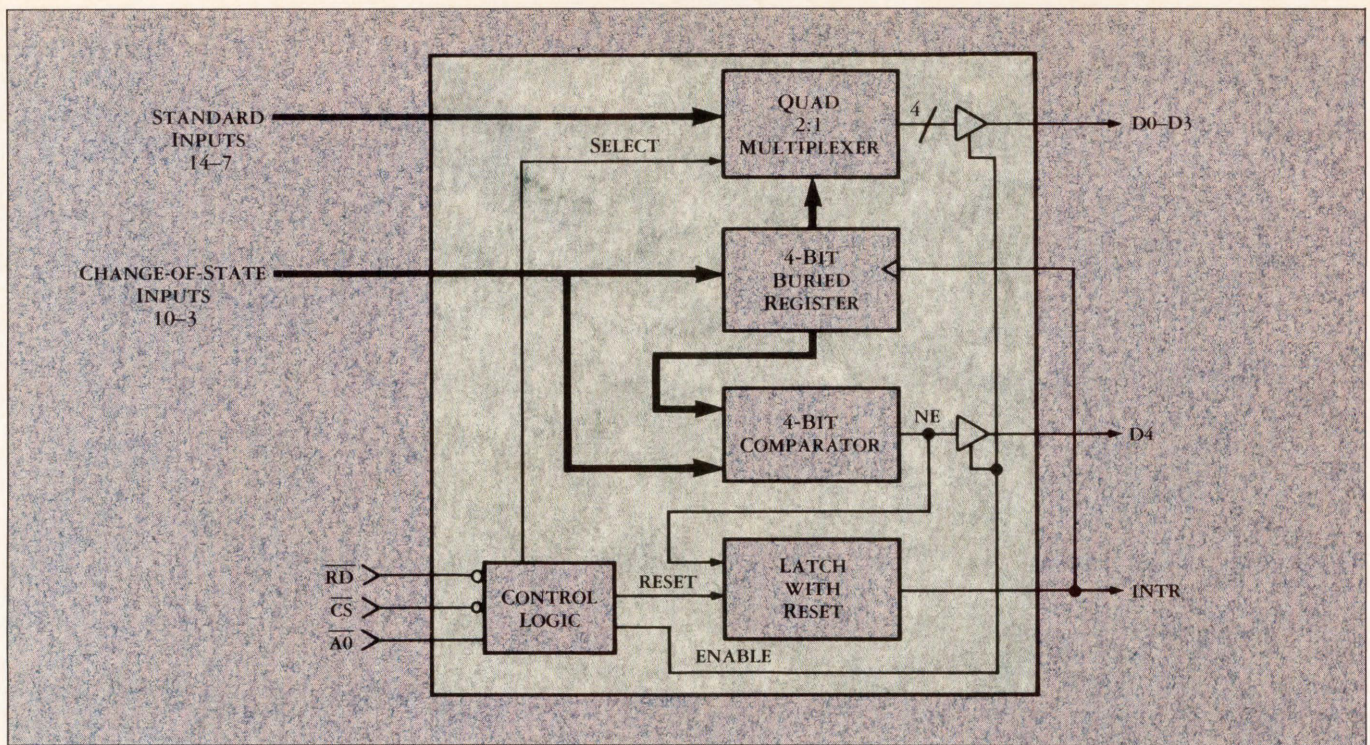
Figure 2. A PLD (here, a PEEL device) can implement an input port that is sensitive to changes in state, polling a microprocessor with the INTR signal when inputs I_0–I_3 change.

NASA—as a programmable-wait timer. The application requires timing with millisecond resolution after long elapsed periods of time. For example, a Voyager fly-by of Jupiter would need split-second timing, after a year of deep-space travel, to take photographs of the planet. Along with programmable branch logic and microcode blocks, the SAM contains a 15-level stack of 8-bit registers and an 8-bit loop counter. Using the registers and counter, routines within the microcode block could implement nested timing loops. Intervals as long as $2^{8 \times 15}$ power can be generated by the timing loops. Resolution of this timing circuit is as accurate as the device's clock period, which can be as short as 40 ns (with the maximum 25-MHz clock frequency).

A more down-to-earth application is the ubiquitous microprocessor bus. The PEEL device from International CMOS Technology Inc. (San Jose, Calif.) can form a generic input port, with four of its eight inputs sensitive to changes in state (Figure 2). On a change of state, the part can interrupt a local processor using the INTR output. The processor can then read the new state on the D_0–D_3 outputs. The D_4 output is used to poll the status of any state changes that occurred after the one that caused the interrupt.

Programmable parts are also available to implement interfaces for specific buses, most significantly the Micro Channel and the VMEbus. Such parts must have enough logic to implement the control sequences as well as enough output drive for the bus

signals. According to PLD manufacturers, bus interface circuitry can occupy as much as a third of a plug-in board.

The PLX 488 from PLX Technology Inc. (Sunnyvale, Calif.) was designed to consolidate this lightly integrated portion of plug-in boards. To implement bus interface logic and drivers, PLX Technology combined a programmable logic array with the buffers required to drive bus signals. The part has 10 inputs, 8 bidirectional I/O pins (each with its own output-enable and feedback terms) and between 8 and 14 product terms for each output macrocell. In addition, two on-chip clock signals help the designer to synchronize signals from the bus to the on-board clock. Four 24-mA drivers, four 48-mA drivers, metastable-hardened registers, and inputs with hysteresis meet bus specifications for the VMEbus, the VME subsystem bus (VSB), the Micro Channel, the NuBus, and other buses.

PLX Technology has, in fact, designed "preprogrammed" versions of the PLDs for the first three buses. The VME 1200 and VME 2000 serve to connect a master or a slave processor, respectively, to the VMEbus. Similarly, the VSB 1200 and VSB 2000 connect master and slave processors to the VME subsystem bus. Finally, the MCA 1200 serves as a bus master, bus requester, and local arbiter for the Micro Channel, allowing either single-cycle or burst-data transfer requests. All devices use the programmable logic plane of the PLX 448 to implement the control logic and the device's 24- and 48-mA drivers to

drive directly the buses' control signals. Customers can work with the company to customize these designs for the peculiarities of their own systems; customized designs can then be quickly implemented in the programmable PLX 488. Force Computers Inc. (Los Gatos, Calif.) has already designed the parts into some of their VMEbus-based products.

Altera offers a Micro Channel interface part (Figure 3) that implements Micro Channel control circuitry and other features, particularly the Programmable Option Select (POS) registers used in most Micro Channel designs. Altera separated the DMA unit (the EPB2002) from the rest of the controller (the EPB2001) because, according to its experience, less than 50% of the Micro Channel interfaces need DMA capability. Altera claims that the part replaces at least 14 TTL and PLD circuits, saving as much as 25% of the Micro Channel board area. By not having the data path run through the chips, moreover, the parts can control the access of 8-, 16-, or 32-bit data exchanges. The PLD's 24-mA drivers ensure compatibility with the bus specifications.

In the EPB2001, all general-purpose interface functions, such as the Programmable Option Select registers, are included. The chip allows access to POS register contents through 16 adapter-accessible I/O lines that replace jumper wires and DIP switches. Programmable chip selection logic permits adapter address remapping. The chip's EPROM provides for storage of the board ID, chip select ranges, and POS
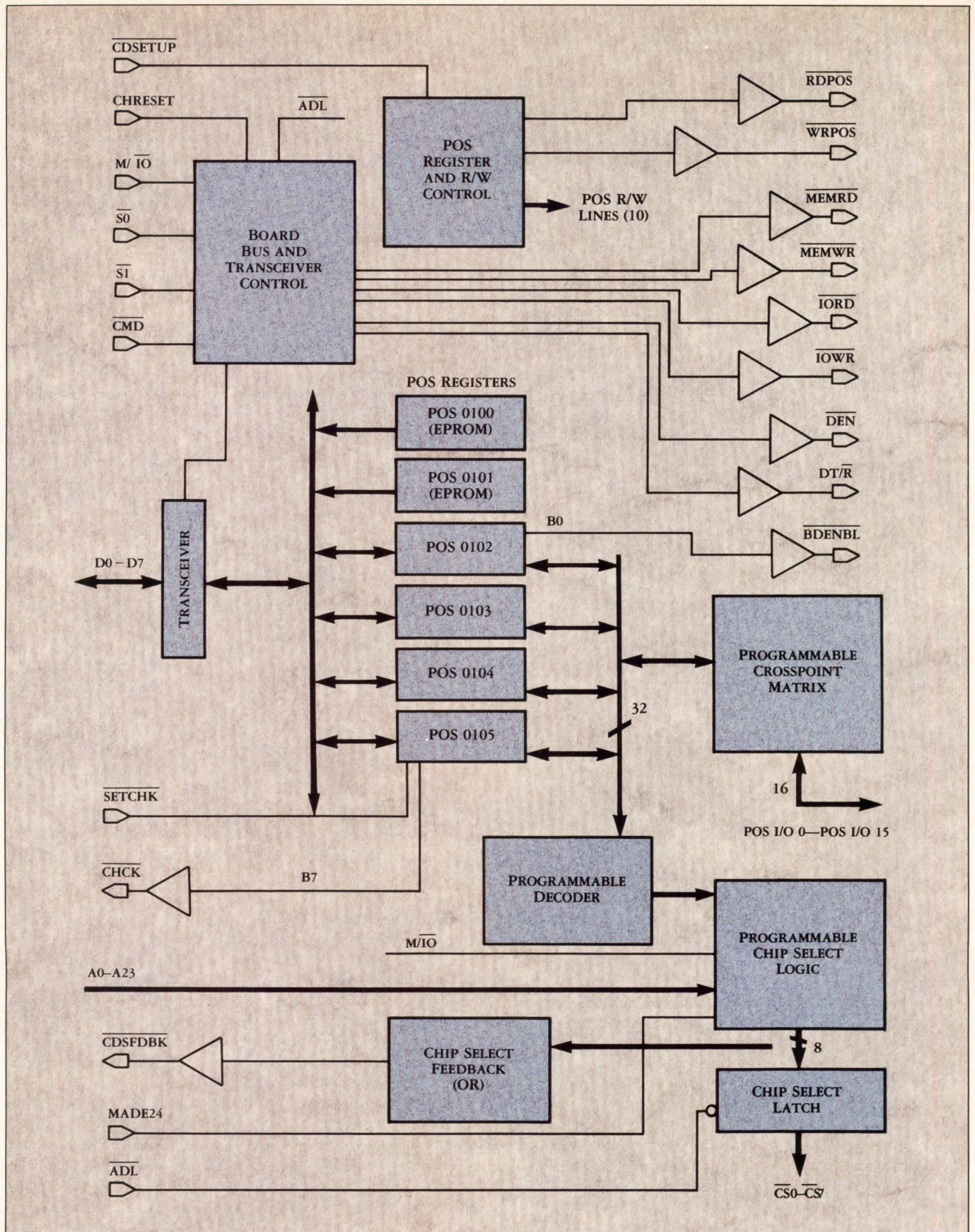
Figure 3. The EPB2001 provides the control circuitry for a Micro Channel interface integrated with programmable decoding, chip selection, and access to registers.

I/O selection for reduced component count and added design security.

As the amount of logic, I/O signals, and registers increases, PLDs can implement entire subsystems. For example, one bar-code reader circuit, which synchronizes itself with the rate of incoming data, uses just one Altera EP1800. Sync circuitry adjusts the device to the speed of the wand, so that the device can read the pulses according to how fast the user is sweeping the wand over the bar code. This
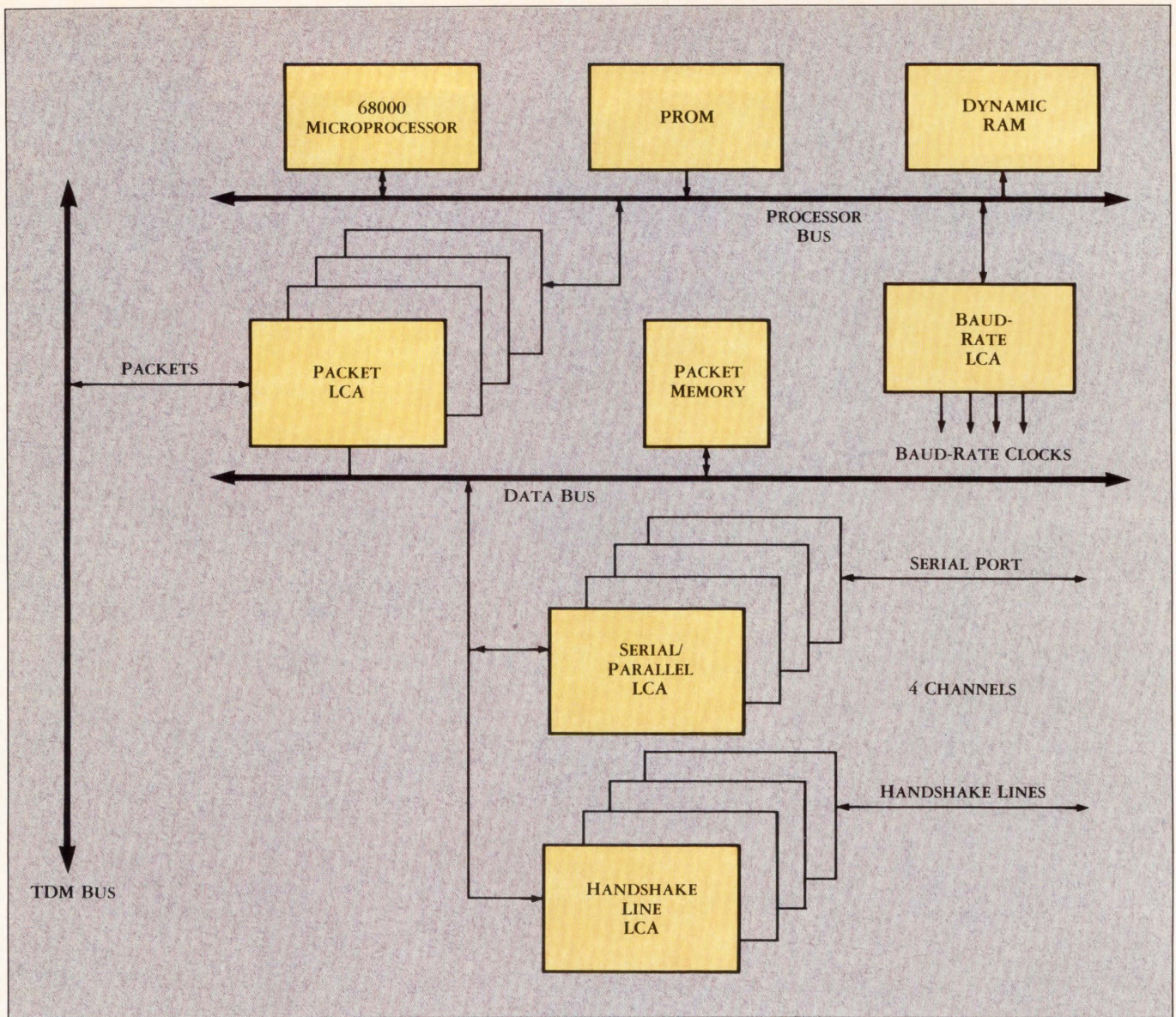
Figure 4. For each channel on this serial communication system, two programmable gate arrays (LCA devices) replaced a 68000 microprocessor, an 8530 serial communications controller, and various support logic.

particular circuit contains five modules: a controller state machine, a sync counter, a byte counter, a shift register, and a microprocessor interface.

The sync counter synchronizes the reading of data by timing the width of the barcode bar. It increments a counter to measure how fast the wand crosses the first "start" bar. This value is used to determine the middle of each subsequent data pulse.

The byte counter implements a Gray code counter. A Gray code counter increments in a manner whereby only one bit of the counter changes at a time, so that the decoded count outputs will experience no glitches.

The shift register stores the synchronized data stream. The microprocessor interface contains open-collector output signals that indicate byte detection and status bits for active read, direction of the wand,

and header validity. Finally, the state machine controls the operation of the other four blocks.

Under control of the state machine, the circuit idles until the wand passes over a dark region, indicating the start of start bars. The sync counter determines the width of the start bar, which determines the width of incoming data pulses. As the machine reads in the data bits, it checks the header for bit sequences that determine a forward read (00) or a reverse read (01), corresponding to the direction of the wand over the code. The part then reads in the 4 bytes of data and check-sum bytes into the shift register.

Another PLD subsystem illustrates the flexibility of PLDs with architectures similar to those of gate arrays. When Stratocom Inc. (Campbell, Calif.) needed to upgrade the serial transmission in its Fastpacket system, it found that it would need

to add hardware to its present design. Instead of adding ICs and making its boards more complex, it designed its own serial interface controller using the LCA2018 programmable gate array from Xilinx Inc. (San Jose, Calif.). The company eventually replaced other discrete parts with LCAs, as shown in the system diagram in Figure 4.

Originally, each data channel in the Stratocom systems used a 68000 to control a 8530 serial communications controller. Each data channel provides physical interfaces for four RS-232-C, V.35 or RS-422/449 channels. Two Xilinx parts replace the microprocessor, its code, and the controller, in effect implementing all of the hardware functions and software algorithms. Not only do the parts add features without changing board size, the reprogrammability allowed the company to exercise several design iterations.

New Directions for PLDs?

At this year's International Solid-State Circuits Conference, two new technologies appeared that may influence the future of PLDs. First, Actel Corp. (Sunnyvale, Calif.) revealed its programmable gate array, which combines the architecture of a gate array with user programmability. Like a gate array, the die contains rows of logic "modules" separated by wiring tracks with segmented horizontal and vertical wire segments. Tiny "antifuses," which become conductive when subjected to programming voltages, are used to program both the function of the logic cells and the interconnections. Unfortunately, the first Actel device is small—295 logic modules and 55 I/O signals; on the other hand, the company says it has a way to test all modules and wiring tracks before programming.

Ramtron Corp. (Colorado Springs, Colo.) introduced a shadow RAM device that uses a ferroelectric film to permanently store a digital state (Baker, 1988). Tiny ferroelectric lead zirconate titanate (PZT) elements have two stable states of orientation, which are set by the state of connected memory cells. When power is removed from the device, the polarization remains; when power is restored, the polarization of the PZT element induces currents that restore the state of the memory cell.

This technique could be applied to RAM-based PLDs like those from Xilinx. Although the manufacturing of such devices is a major obstacle to widespread use, they have at least one distinct advantage over EPROM and EEPROM technology: the charge density of the PZTs is 10 times greater, so denser memory storage is possible. ∎

REFERENCE

BAKER, S. MAY 1988. "Ferroelectric Chips," *VLSI Systems Design.*

For each channel, one of the two Xilinx XC2018s provides the serial transmitter and receiver logic that converts the parallel words of data on the data bus into the required serial data for the serial buses. The other part implements all of the handshake logic for controlling the flow of data between the Fastpacket channel and the external peripheral.

By reprogramming the Xilinx parts in situ, the design can perform enhanced bit-error rate testing. In this mode, they introduce distortion into the transmitted signal so that bit-error rates can be monitored under controlled conditions. This function is normally implemented with external test equipment. ∎

REFERENCES

GOETTING, E., M. HAKANSSON, AND S. PATTERSON. APRIL 1988. "Multilevel EEPLD Implements Biotelemetry System," *VLSI Systems Design.*

MEYER, E. OCTOBER 1987. "Programmable Logic Overview," *VLSI Systems Design.*

Uses span

a spectrum of

design services

and products

VHDL: Documentation or Development?

At LAST month's Design Automation Conference in Anaheim, Calif., numerous companies hawked strategies and products that support the VHSIC Hardware Description Language, VHDL. This language is gathering momentum and seems destined to have a significant impact outside the military electronics sector that spawned it.

VHDL arose from the need for a standard documentation language for electronic systems created for military projects. Says Ken Bakalar at CAD Language Systems Inc., "It soon became clear that the language should also have the ability to be processed by computers and simulated."

The number of goals that VHDL embodies results in a spectrum of design services and products from companies trying to capitalize on the language standard. For documentation and transmission purposes, translators like the Helix-to-VHDL translator from Silvar-Lisco and the ISP'-to-VHDL translator from Zycad's Endot division allow VHDL designs to travel from one design environment to another. On the other hand, the use of VHDL as a development language, as typified in technology at Intermetrics, represents the full flower of VHDL.

The VHDL Tool Integration Platform (VTIP) from CAD Language Systems Inc. (CLSI) expands the use of VHDL as a documentation medium by adding design database and management facilities and allowing developers to create an interface to simulators. VTIP is built around a Design Library System that holds design data in CLSI's intermediate format. A Software Procedural Interface (SPI) provides access to the data for entry, retrieval, or manipulation of the design by other tools. Three CLSI tools support the transmission and creation of designs expressed in VHDL: the VHDL Analyzer accepts VHDL designs and
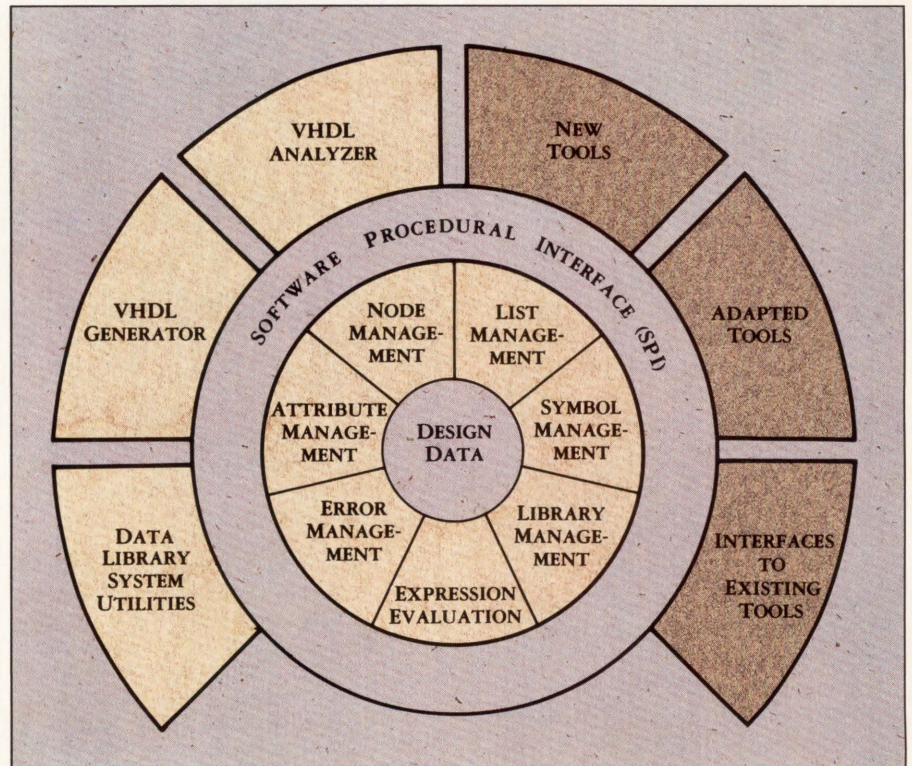
Around data in the Design Library System (DLS), CLSI adds its Software Procedural Interface to control access and manipulation of data. Plugging into the resulting DBMS are tools to read (the VHDL Analyzer), write (the VHDL Generator), and manipulate (DLS utilities) designs expressed in VHDL.

enters them into the DLS; the VHDL Generator builds a VHDL description of a component from the DLS database; and the VHDL Editor guides a designer during the construction of a VHDL model.

The VHDL Toolset from Intermetrics is similar to VTIP (primarily because it adheres to the same standard). Intermetrics' Design Library Manager is somewhat like DLS, and its Analyzer and Reverse Analyzer perform the same functions as the VHDL Analyzer and VHDL Generator.

A simulator, however, distinguishes the Intermetrics tool suite, filling it out into a hardware-development environment. It supports all levels of modeling that are supported by the language, permitting simulation at various levels of abstraction. In addition, a Simplifier program reorganizes the hierarchy of design descriptions, collecting lower-level elements into higher-order descriptions to simplify modeling and analysis. A user uses a "conceptual design bench" to create stimuli and gather information about the behavior of the VHDL circuit.

Although this environment has more analysis capability than CLSI's tool suite, it lacks some of the capabilities of CLSI's SPI that simplify the development of interfaces with other simulation environments.

The Intermetrics simulator joins the VantageSpreadsheet VHDL simulator, which was discussed in last month's issue of *VLSI Systems Design* (p. 104), and the

HOW WE CREATE AN UNEQUALLED OPPORTUNITY.

CMP is one of the fastest growing publishers of business newspapers and magazines, with advertising sales up better than 20% a year for each of our fifteen years. But growth isn't an end. Rather, it's our means to unequalled opportunity.

By growing, we create new opportunities. Opportunities for good reporters to advance to responsible editors. Opportunities for staff to become supervisors. Opportunities for sales people to become sales managers. Opportunities for stars to go anywhere their talent takes them. The fact is that you can't hold outstanding people unless you offer them a better opportunity than they can find elsewhere. And you can't offer an unequalled opportunity without consistent growth.

So we plan our growth. We plan to expand the scope and coverage of each of our publications. And we plan to add new publications in a variety of emerging industries. It's not just that we find bigger numbers exciting. What's exciting is seeing good people respond to bigger opportunities. And good people make CMP publications usually the best read in their industries.
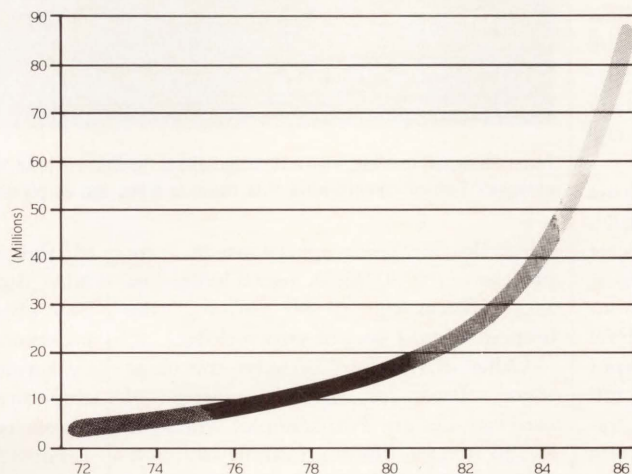
CMP Publications, Inc.
600 Community Drive
Manhasset, NY 11030
(516) 562-5000

Electronic Buyers' News
Electronic Engineering Times
VLSI Systems Design
Computer Systems News
Computer Reseller News

InformationWEEK
CommunicationsWeek
Business Travel News
Tour & Travel News

High-performance

software can

run on low-cost

hardware

80386 Derivatives Aim at Low-Cost PCs and Embedded Systems

ATTEMPTING TO leverage the ubiquity of 80386-based personal computers, which could serve as development systems for other 80386-compatible systems, Intel has tailored new versions of the 80386 in terms of price and functionality, as well as the surrounding support circuitry, to the needs of low-cost personal computers and embedded systems.

The first type of application—general-purpose platforms with lower prices and less performance than 80386-based machines—can be implemented with the 80386SX. For this chip, Intel replaced the 80386's 32-bit interface with a 16-bit data bus and a 24-bit address bus and reduced the clock frequency to 16 MHz (see the table). Narrower buses and longer clock periods simplify system design and lower cost.

The chip's 32-bit internal architecture with memory management, meanwhile, can execute the same complex software (including the Unix operating system) as the 80386, as well as software written for the other 8086-family processors. Intel rates the processor's sustained throughput at 2.5 to 3 MIPS. In comparison, the full 32-bit interface and 25-MHz clock frequency of the 80386 increases system throughput (of machines like Sun Microsystems' 386i) to at least 5 MIPS.

There are a few functional differences between the 80386SX and the 80386 that result from their different bus sizes. Some of these differences influence the design of support hardware for the processors. For example, the 80386's 32-bit data bus requires four byte selection signals (BE0#–BE3#) to distinguish between bytes of data on the bus. The 80386SX, on the other hand, requires only two byte selection signals (BHE# and BLE#) to select bytes on its 16-bit data bus. Similarly, to
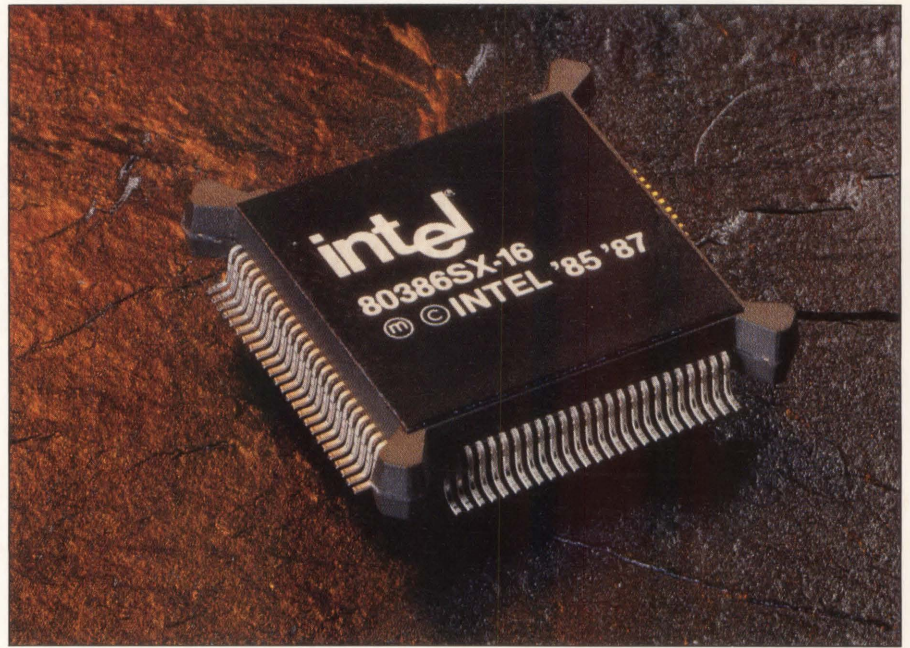
Fewer I/O signals resulting from a 16-bit bus allows the 80386SX to be placed in this surface-mount package. The package's bumper corners allow it to travel in tubes and be placed by automatic assembly equipment.

select floating-point operations on a coprocessor, the 80386SX uses address line A_{23}, whereas the 80386 uses A_{31} (the highest address bits of either chip).

Other distinctions between the chips affect software or microcode instead of hardware design. For example, when the 80386 is reset, the high nibble of its DX register (which holds the component and revision identifiers) is set to "3"; for the 80386SX, it is set to "23H." The operating system running on the 80386SX must be aware of its 16M-byte physical address space; applications that exceed this limit must make use of the 80386SX's page mode. Software written for these processors could use the identifier to determine which component it is running on.

Reducing bus widths also eliminates I/O signals, resulting in fewer package pins for the device. As a result, the 80386SX fits into a 100-lead plastic quad flat pack (see

the figure), instead of the 80386's 132-pin PGA. Bumpers on the corners of the flat pack protect the leads, so the chip can be packaged in tubes and mounted on boards by automatic handlers. In this way, the chip can simplify assembly and push down system cost.

Furthermore, the lower clock frequency and plastic flat pack lower the price of the chip itself; initial 80386SXs sell for $219—73% of the price of an 80386. Intel intends to drive the price of the 80386SX down, making it as inexpensive as an 80286 by 1990.

For embedded systems, Intel removes from the 80386SX two features that are unnecessary (and unwanted): memory management and the virtual-8086 mode. The resulting 80376, introduced in April, provides the same system economies as the 80386SX for embedded systems that do not run general-purpose software.

How the 80386SX and 80376 Stack Up

	80286	80386SX	80386	80376
INTERNAL ARCHITECTURE (BITS)	16	32	32	32
DATA BUS SIZE (BITS)	16	16	32	16
ADDRESS BUS SIZE (BITS)	24	24	32	24
PHYSICAL ADDRESS SPACE	16 MB	16 MB	4 GB	16 MB
VIRTUAL ADDRESS SPACE	1 GB	64 TB	64 TB	64 TB
PAGING	NO	YES	YES	NO
PROTECTION	YES	YES	YES	YES
VIRTUAL 86 MODE	NO	YES	YES	NO
OPERATING FREQUENCIES (MHz)	8, 10, 12.5	16	16, 20, 25	16
ARITHMETIC COPROCESSOR	80287	80387SX	80387	80387SX

Both the 80386SX and 80376 use a new floating-point coprocessor, the 80387SX, which is compatible at the object-code level with the 80387. It contains the same 16/24-bit bus structure as the processors. The chip, comes in a 68-lead PLCC, implements the IEEE-745 floating-point standard and delivers three to five times the performance of the 80287. It executes the new floating-point software instructions of the 80386, such as FSIN, FCOS, and FSINCOS.

For general-purpose computing, therefore, Intel offers the 80286 for the lowest-cost systems and the 80386SX and 80386 for ascending levels of performance and price. Designers of embedded systems who want the performance of a 32-bit architecture now have a reasonably priced alternative, the 80376, to the numerous RISC-based embedded processors (including Intel's recently announced 80960). ■

Intel Corp.
Santa Clara, Calif.
(408) 765-8080

Endot_VHDL simulator, discussed below. Vantage's simulator not only accepts designs in the VHDL format, it also uses a proprietary database that supports concurrent execution of simulation, schematic capture, and waveform-viewing programs. These features reduce and, in some cases, eliminate compilation between design steps. In addition, Vantage plans to develop an interface with the Silcsyn logic synthesis tools from Silc Technologies Inc. (Burlington, Mass.; see the June issue, p. 12), providing the first direct link between VHDL and the fledgling logic synthesis technology.

Silvar-Lisco's translator creates VHDL models (at an architectural level) from models in HHDL, the language for the company's Helix behavioral simulator. Herman Beke, vice president of marketing, says that "HHDL is the hardware modeling language closest to VHDL," so that users of Helix won't sacrifice any Helix capabilities to maintain compatibility with the VHDL specifications.

Silvar-Lisco is integrating Helix into a unified simulation environment—Unisim—that will also include the Cadat simulator from Cadnetix's recently acquired HHB Systems Division. Unisim, which should be available by the end of September, will accept designs expressed in both Helix and Cadat models. Mixed-level simulations will use both simulators running concurrently.

Future enhancements to Unisim include an analog simulator, logic synthesis, and Cadat-like models in Helix that will allow it to interact with Cadat's hardware modeling system. The environment will also get a VHDL-to-Helix translator that, through logic synthesis, will be able to translate VHDL designs into Cadat models.

Zycad demonstrated a VHDL design environment that used the VHDL Developer design entry tool from Vista Technologies. In a manner similar to the VHDL Editor, the VHDL Developer simplifies the creation of VHDL models by providing context-sensitive menus with automatic formatting, searching, and substitution. It also contains a librarian program that stores and organizes the VHDL-defined database. Design files expressed in EDIF 2 0 0 can also be translated into VHDL files.

Zycad announced that the VHDL simulator from its recently acquired Endot division should be released this year. Zycad plans to integrate the Endot_VHDL simulator more tightly with its simulation accelerators. This integration provides an environment for mixed-level simulation with VHDL models and a hardware accelerator for gate- and switch-level models. At present, the Endot tools are linked through stimulus translation and comparison tools that simplify the correlation of high- and lower-level modeling and simulation.

The Endot_VHDL design environment features a superset analyzer (compiler) that is switchable to standard VHDL, an inter-active simulator with expression evaluation, and software development support tools for hardware/software verification and integration. It also includes graphical and statistical analysis tools, an easy-to-learn graphical user interface with a programmable command language for customizing analyses, and an ISP'-to-VHDL translator. For users of the Endot_ISP' simulator (formerly called N.2), the ISP'-to-VHDL translator is also available now, separately. ■

CAD Language Systems Inc.
Rockville, Md.
(301) 424-9445

Intermetrix Inc.
Cambridge, Mass.
(617) 661-0072

Silvar-Lisco
Menlo Park, Calif.
(415) 324-0700

Vantage Analysis Systems Inc.
Fremont, Calif.
(415) 659-0901

Vista Technologies Inc.
Rolling Meadows, Ill.
(312) 640-4415

Zycad Corp.
Cleveland, Ohio
(800) 545-8765

must know its pin-out). Hardware models are appropriate for any complex device, not just microprocessors. Hardware models don't have to run at real clock speeds, so they can incorporate prototype silicon, and they can be used in the simulation stage, before construction of a system prototype.

On the other hand, in-circuit emulation cannot be used until a prototype *system* is constructed, and bugs in the prototype hardware may be difficult to observe. The prototype will operate with typical timing, making worst-case analysis impossible.

One major drawback to system simulation today is that it is much slower than prototype operation and is not a practical tool for verification of large, complex programs. It is practical for verification of small kernels of embedded code (for instance, 10,000 executed instructions). However, thorough system simulation can guarantee that the prototype hardware is perfect, greatly facilitating software debugging.

■ KEY ATTRIBUTES OF HARDWARE MODELERS

To help the potential hardware modeling user understand the differences between existing hardware modelers, this section catalogs the key attributes of current-generation products.

■ *Round trip evaluation time*, or the time required for one evaluation of a hardware model, from the initiation of the evaluation request by the simulator until the simulator receives the result. In general, the simulator simply waits during the evaluation, so this time will affect overall simulator performance.

■ *Maximum effective device-clock frequency*, or the maximum frequency at which the hardware modeler can clock a dynamic reference element. The hardware modeler should be able to clock the reference element at the device's minimum specified operating frequency.

■ *Maximum simulated cycles for public devices*, or the maximum number of cycles that a public device can be simulated. It is limited by the depth of fast pattern memory, the pattern compression, and folding.

■ *Width of pattern memory (in pins)*, or the number of device pins that can be represented by a single pattern in fast pattern memory. If it is too small, pattern folding may be needed.

■ *Support for private-mode models*. The length of simulation is not constrained by the size of fast pattern memory, and the evaluation time does not increase with the simulated time.

■ *Hardware three-state sampling*. Preferred pin electronics distinguish between three states of output pins (0, 1, and Z), not just 0 and 1.

■ *Maximum device capacity*, or the number of reference elements that can be used simultaneously.

■ *Maximum supported device pin count*. Fast pattern memory width, pin electronics width, fixturing hardware, and completeness of the control software all are factors in determining the maximum supported device pin count. The apparent maximum may not be what is actually supported.

■ *Custom wiring of adapters*. Some modelers require extensive custom wiring to connect a reference element; others perform this operation in software.

■ *Shared use*. Some modelers can only support a single user.

■ *Networking*. Can the modeler be fully utilized over a network? If so, what additional hardware is required?

■ *Live insertion*. Can a reference element be changed without powering down the central unit for a shared modeler?

■ *Simplicity of software shells*. A modeler that requires complex software shells can be difficult to use and may contain inaccuracies.

■ *Device initialization*. The method of device initialization affects the difficulty of creating models.

■ *Model library*. The size and accuracy of a vendor-supplied library of hardware models are of concern, especially when large, complex software shells are required.

■ *Simulator support*. Does the modeler support your simulator? None of the other attributes matter if the hardware modeler doesn't.

■ CONCLUSION

In the 1970s, examining a prototype was the only practical method for debugging a complex electronic system. In those years, an engineer designed his system on paper and debugged it with an oscilloscope, a logic analyzer, and an in-circuit emulator.

In the 1980s, electronic systems became much more complex, more susceptible to human design errors, and more difficult to observe and change after construction. Product life cycles decreased. These changes were the driving force behind the dramatic progress we have seen in CAE tools. By the end of the decade, over 100,000 digital logic simulators will be in use.

In the 1990s, it will be just as unthinkable to skip full simulation of an electronic system design as it would have been to skip debugging the prototype 10 years ago.

Hardware modelers will be the in-circuit emulators of the 1990s. They enable the modern system engineer to debug his design with a simulator just as the in-circuit emulator enabled his predecessor to debug his design with a prototype. ■

REFERENCES

GILES, D., ET AL. 1985. "Maintaining Simulation Accuracy through Physical Device Models," *International Test Conference 1985 Proceedings*.

JOHNSON, P. SEPTEMBER 1986. "Considerations in Selecting a Physical Modeling System," *ADEE Technical Session Proceedings*.

JOHNSON, P. 1988. "Software vs. Hardware Models for System Simulation," *VLSI Systems Design's 1988 User's Guide to Design Automation*.

MANAGEMENT ROUNDTABLE INC. JULY 1987. "Are Hardware Modelers a Real Alternative for Simulation?" *CAE Electronics Alert*.

STOLL, P. OCTOBER 1985. "PMX: A Hardware Solution to the VLSI Model Availability Problem," *Proceedings of the International Conference on Computer Design*.

WIDDOES, L.C., JR. AND B. HARDING. MARCH 22, 1984. "CAE Station Uses Real Chips to Simulate VLSI-Based Systems," *Electronic Design*.

WIDDOES, L.C., JR. MAY 20, 1986. "Method and Apparatus for Modeling Systems of Complex Circuits," U.S. Patent Number 4,590,-581.

WIDDOES, L.C., JR. JANUARY 6, 1987. "Method for Simulating System Operation of Static and Dynamic Circuit Devices," U.S. Patent Number 4,635,218.

ABOUT THE AUTHORS

L. CURTIS WIDDOES, *president and founder of Logic Modeling Systems, was previously vice president and cofounder of Valid Logic Systems, where he developed the industry's first hardware modeler. Before that he was technical director of the S-1 project at Lawrence Livermore National Laboratory. He holds a BS in engineering from the California Institue of Technology and a PhD in computer science from Stanford University.*

HOLLY STUMP, *director of marketing at Logic Modeling Systems, was previously product marketing manager for application-specific hardware at Valid Logic Systems. She holds a BSEE from the Illinois Institute of Technology and an MS in engineering management from Stanford University.*

```
                    68 → 200-MHz CLOCK

                         440 × 415 MIL²,
                         2.0-μm GaAs,
                         256-PIN SPECIAL PACKAGE
                         (71 SIGNAL PINS)

 6-STAGE PIPELINE
 ■ FETCH:                REGISTER
    ADDRESS OUT           FILE
    INSTRUCTION IN       (16 × 32)
 ■ DECODE AND EXECUTE
 ■ MEMORY:
    ADDRESS OUT
    DATA MOVED
 ■ WRITE BACK                 ALU
                             (32)

   24      32    26      32      32
 ADDRESS  DATA  ADDRESS DATA    DATA

   GaAs         GaAs          GaAs
   CACHE        CACHE         FPU
  (2.5 ns)     (2.5 ns)
```

Figure 5. Texas Instruments' GaAs RISC for DARPA is significant in that it shows that its possible to put a 32-bit CPU in gallium arsenide. Developed jointly with Control Data, it contains 12,895 gates, which TI says may make it the largest functional logic device ever built in GaAs.

Continued from page 63

RISC architecture (described below and shown in Figure 4). Be aware, however, that the new C300 50-MHz version of the Clipper has a very large CPU chip, which is said to be a result of enhancements of the floating-point unit. Even in 1.5-μm geometry, the C300 CPU is 537×561 mil².

To make room for the FPU, the Clipper and the 88000 have a reduced number of CPU data registers, but they say they make up for this by a "scoreboarding" mechanism, which sets bits to keep track of which registers need to be kept free to receive data.

The Clipper's most visible use has been in workstations built by Intergraph Corp. (Huntsville, Ala.), which bought the Clipper operation (the Advanced Processor Division, Palo Alto, Calif.) after Fairchild Semiconductor was sold to National Semiconduc-

tor Corp. (Santa Clara).

To some degree, the Clipper's architect, Howard Sachs, discounts some of his competitors' better performance on standard benchmarks, saying that the frequently used benchmarks are too small to represent actual performance in full-blown Unix applications. In those larger real-world applications, Sachs believes, having caches as part of the chip set will put the Clipper ahead, especially when system cost and board space are considered.

■ THE 88000:
IMPRESSIVE BUT
EXPENSIVE
CONCURRENCY

In designing the M88000, Motorola Inc. (Austin, Texas) has taken Cray's concepts (which project technical leader Roger Ross freely admits served as his inspiration) and followed the lead of the Clip-

per RISC in reducing those concepts to silicon (which Motorola does not admit). But at the same time, because it has come out after so many other RISC chips, Motorola has pushed into higher levels of parallelism to give the 88000 a performance edge.

The company says its architecture can support 11 "function units'" on the CPU chip operating in parallel. At present, the MC88100 CPU (Figure 4) has five units: the instruction (fetch) unit; the integer and bit field unit (ALU); the data memory (access) unit; and the floating-point multiplier and the floating-point adder, which are grouped together as a single "special function unit." Future CPUs could incorporate six additional units, which might, according to Motorola, be such functions as a graphics floating-point unit, an artificial-intelligence "garbage collection" unit, or an MS-DOS emulation unit.

In addition, the 88000 chip set includes sophisticated four-way set-associative caches, integrated with MMUs, that can be used on both sides of their Harvard external busing. Not only are they large 16K-byte caches, but the 88000 architecture permits them to be used four deep on both sides. This parallelism makes the 88000 potentially very powerful—but it has its price. The CPU is 430×430 mil² and the caches are nearly 500×500 mil². Most of Motorola's competitors are dourly predicting that it will be some time before the 88000 chip set drops below the several-thousand-dollar level and ramps up to production volumes. Meanwhile, though, none of them are taking the 88000 lightly.

■ GAAS R&D: A
GLIMPSE OF THE
FUTURE

Current RISC chips are but prologues to the next iterations that are currently being worked on by most suppliers. To stay competitive, they are pushing past the present per-

formance level of 10 to 30 MIPS to 50 to 100 MIPS and beyond. CMOS will get them to 50 MHz, but to go further they must move to ECL, and they may have to switch from silicon to GaAs to move past 100 MHz.

A GaAs RISC processor jointly developed, for the Pentagon's Defense Advanced Research Projects Agency (DARPA), by Texas Instruments Inc. (Dallas) and Control Data Corp. (Minneapolis) provides a tangible glimpse of what future 200-MIPS GaAs RISC chips might be like. The chip has a reasonable die size of 440×415 mil² and has executed instructions at 68 MHz. Its 12,895 gates (using bipolar HI²L, heterojunction integrated injection logic) are sufficient to give the chip good RISC capability (despite its frugal appearance). The data register file contains just sixteen 32-bit registers and the ALU is a full 32 bits (Figure 5). The CPU has a deep, six-stage pipeline, with the extra stages allowing data to be accessed while it is still in the pipeline—thus permitting decreased memory access time. The planned external GaAs caches—note the separate "Harvard" access to instructions and data—will be essential for bridging the speed gap between the CPU and external memory.

This first version of the chip is only expected to get up to 100 MHz, or 100 MIPS maximum. But future versions, which will have 1.5-μm rather than the present 2.0-μm device geometries, are expected to reach the goal of 200 MHz (as well as shrink the chip size to 300×280 mil²).

TI says that to reach 200 MHz it will have to achieve gate delays of 160 ps so that the signals can traverse the 30 gates of the CPU critical path within the 5-ns clock period.

The next article in the series will examine how these various RISC processors tackled some of the major problems in squeezing performance out of a RISC architecture without too much of a system cost penalty. ■

This index is provided as an additional service. The publisher does not assume any liability for errors or omissions.
This index is provided as an additional service. The publisher does not assume any liability for errors or omissions.

A close look at digital testers will give you a new Outlook for design verification.

The DAS 9200 from Tektronix is a high performance digital test system. But for even higher performance, take a close look at the T-100 from Outlook Technology.

Both products have a maximum recording clock of 2 GHz. But only the T-100 uses intelligent sampling for precise data recording with 100 ps resolution. That's up to five times the resolution of the 9200...the difference between just seeing what happened and finding out why.

Both instruments find timing problems, but only the T-100 can trigger on and track down setup and hold time violations to save countless hours of searching for logic problems.

Both products can be used in automated set-ups to test boards and chips at high speed and high resolution. But the T-100 can perform up to ten times more tests per hour. And it can act as a 250 MHz pattern generator (stimulus), 250 MHz logic recorder (response), or both.

The T-100 also comes with a friendly human interface, including LogicProbe, a new utility program that makes setup and use faster and easier than ever before.

For a new outlook on digital testing, look into the T-100 family, with prices starting at just $15,000. Contact Outlook Technology, Inc., 200 E. Hacienda Ave., Campbell, CA 95008 (408) 374-2990.

Rented exclusively by United States Instrumental Rentals, Inc. 1-800-824-2873.
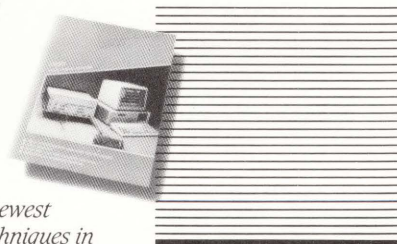
Discover the newest measurement techniques in our 12-page brochure.

OUTLOOK

OTI-46B

CIRCLE NUMBER 21