

BY MICHAEL SANTARINI • SENIOR EDITOR

ASIC PROTOTYPING: MAKE

AS DIGITAL-IC DESIGNS HAVE BECOME MORE COMPLEX, SO TOO, HAS THE TASK OF VERIFYING THEIR FUNCTIONS. OVER THE YEARS, EDA VENDORS HAVE DONE LITTLE TO REDUCE THE GAP BETWEEN THE NUMBER OF GATES YOU CAN DESIGN AND THE NUMBER YOU CAN VERIFY IN A REASONABLE AMOUNT OF TIME.

C designers often spend as much as 60 to 80% of their time simply verifying designs, and that percentage is growing. To get a leg up on verification, many design groups have turned to hardware-assisted verification to prototype their designs. Breadboarding was the first form of hardware prototyping, and it has since grown in sophistication and popularity. Studies from Collett International (www.collett.com) and Deepchip.com (www.deepchip.com) suggest that 30 to 40% of all ASIC projects involve prototyping. You can now build your own multimillion-gate prototype using off-the-shelf FPGAs, but, for large or complex designs, you may want to buy predesigned prototyping systems or, if you can afford it, rent or buy a simulation accelerator or an in-circuit emulator. The decision to build, rent, or buy depends on several factors, including required clock speed, capacity, functions, cost, system-design skills, and the time you have to verify your design (**Reference 1**).

PROTOTYPING SYSTEMS

Prototyping ASICs and SOCs (systems on chips), say designers and vendors, is essentially a step backward—turning an SOC into what some refer to half-jokingly as an SOB (system on board). In building a prototyping system, some designers reconstruct the functions of their ASICs using a mixture of discrete components, legacy ASICs, and FPGAs providing new functions. Others construct or even buy a fast-prototyping system in which the design is programmed into a motherboard. The motherboard holds an array of FPGAs and daughterboards to connect to unique functions or a larger system.

Engineers can design or buy an ASIC-prototyping system that runs at speeds approaching 250 MHz—a performance level that in some cases approaches the speed at which the final chip will operate. Such a prototyping system is much faster than commercial emu-

VERSUS BUY



AT A GLANCE

- ▶ Acceleration is 10 to 50 times faster than RTL simulation.
- ▶ Emulation is 1000 to 5000 times faster than RTL simulation.
- ▶ FPGA prototypes can be 10,000 times faster than RTL simulation.
- ▶ EDA vendors now offer partitioning software that simplifies building prototypes.
- ▶ Multiple vendors offer FPGA-prototyping boards for less money than it would take you to build your own.

lators that top out at 2 MHz and 110 times faster than an RTL simulator. Designers can use fast-ASIC-prototyping systems to test the functions of the design in the system environment or to get a jump on embedded-software development.

The drawback of ASIC prototypes, however, is that they are harder to debug because it is difficult for designers to pinpoint the exact location of a bug on these systems. Design groups base their prototypes on extensive simulation-based verification. MIPS Technologies uses almost every form of hardware-based acceleration to verify new microprocessor-core designs and to help customers integrate cores (see sidebar “MIPS uses it all”).

ROLL YOUR OWN

Building a prototyping system from scratch is now in some ways easier and in other ways harder to do than it was in the past. The huge capacity and speed grades of today’s FPGAs allow users to prototype multimillion-gate ASIC designs. In recent years, EDA companies such as Synplicity and Synopsys have eased prototyping tasks by offering tools to help engineers partition ASIC designs and program their partition blocks into arrays of FPGAs. Today, commercial availability of ASIC-prototyping software has invigorated

the fast-prototyping business and made it a more formidable competitor to traditional emulation vendors, such as Cadence Design Systems and Mentor Graphics. Vendors, especially those offering off-the-shelf prototyping systems, say that the big question for those considering making or buying a prototype is: Do you have the time, the extra engineering staff with the pc-board- and system-design skills, and the budget to build a prototyping system on your own?

Mike Dini, president of The Dini Group ASIC-prototyping company, strongly suggests that it is cheaper to buy a fast prototype than to build one from scratch. Dini 10 years ago was an ASIC- and FPGA-design consultant, and, because he needed verification tools, he started building prototyping boards. He has now dropped the design-services gig for what has turned into a booming business in ASIC prototyping. Multiple vendors now offer such prototyping systems (Table 1). “Our competitors in emulation say we simply offer a bucket of FPGAs,”

TABLE 1 FAST-PROTOTYPING-SYSTEM, ACCELERATOR, AND EMULATOR VENDORS

Company	System	Type	FPGA or custom processor	No. of ASIC gates	Built-in software (partitioning, debugging)
Cadence Design Systems	Xtreme Series	Accelerator/emulator	FPGA with reconfigurable-computing coprocessors	4 million to 50 million	Partitioning and debugging software
	Palladium Series	Accelerator/emulator	Custom ASIC processor based	4 million to 256 million	Partitioning and debugging software
The Dini Group	DN8000k10	Prototyping system	Virtex-4 FPGA based	1.5 million to 24 million	None, must use Synplicity
EVE	ZeBu (Zero Bugs)	Emulator	Virtex-4 FPGA based	750,000 to 6 million for PCI-card-based system or 48 million for box	Automated partitioning and debugging
Flexody	FlexCube	Prototyping system	Virtex-4 FPGAs	1 million to 18 million	ARPCOM
Gidel	Processor board and tools	Emulator and prototyping systems	Stratix II FPGAs	60,000 to 20 million	Simulation acceleration, hardware/software integration, debugging
Hardi Electronics	HAPS (ASIC-prototyping system)	Prototyping system	Virtex-4 FPGAs	3 million to more than 30 million	Self-test, no configuration necessary
Mentor Graphics	VStationPro	Hardware emulator	FPGA	1.6 million to 60 million	Fully automated compilation, fully integrated debugging capability
ProDesign Electronics	Chipit (silver, gold, platinum)	Prototyping system	Virtex-4, -2, and -Pro FPGA based	30,000 to 20 million	Yes
Tharas Systems	Hammer S-Class	Accelerator/emulator	Multicore custom processor	8 million to 16 million	Compiler for Verilog and VHDL
	Hammer M-Class	Accelerator/emulator	Multicore custom processor	32 million to 256 million	Compiler for Verilog and VHDL

says Dini. "I don't take that as an insult. That's exactly what I do. I put a bucket of the most humongous FPGAs on a board, put it together for you, debug it, and sell it to you for cheaper than you can build it on your own." He says that prototyping can be invaluable, but putting together an application-specific board that you'll likely end up throwing away after the project is finished can be wasteful. He suggests that users create daughtercards of their specialized functions and purchase the FPGA part of the prototyping system from a fast-prototyping vendor (see sidebar "The case for buying" at the Web version of this article at www.edn.com/051121cs).

Dini and others say that, even with automated partition software from companies such as Synplicity and Synopsys, building a system on your own with more than three FPGAs can quickly become a nightmare, especially if you are unfamiliar with pc-board design, which, despite what EDA vendors claim, is still a tough task. "Prototyping is a make-versus-buy

decision," says Dini. "With bigger packages that have 700 pins, putting one Virtex FPGA on a board is OK, but putting two, three, or 16 becomes difficult to design, build, and test. For example, two 700-pin FPGAs are too much for pc-board autorouters, so a lot of manual work has to go into it. The pain threshold, at which homegrown FPGA-based ASIC prototyping becomes complex, is around three. Two is not bad, but when you get to implementing three FPGAs, you have to start looking at layer counts and how things connect in the pc board," he says. He points out that modern FPGAs are well-suited for automatic-partitioning software and that most fast-prototyping vendors tailor their systems to work with Certify from Synplicity, for example.

Dini says that the latest FPGAs tout advantages in I/O structure, rather than speed or logic. They integrate serializers/deserializers that connect with differential pairs, so if you connect one FPGA to another FPGA, you need a differential pair operating at approximately 350 MHz

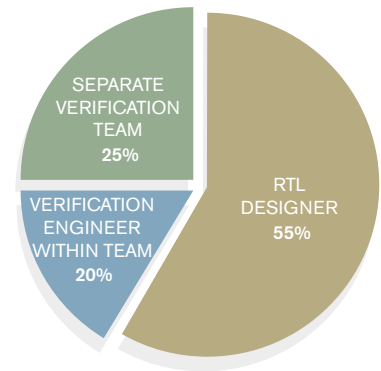


Figure 1 A verification team that is unfamiliar with the ASIC design typically does prototyping (courtesy Synplicity).

with 10-to-1 multiplexing. If your design operates at 35 MHz, you can get 10-to-1 multiplexing on 108 pairs with 1800 signals partitioned between two FPGAs. "That approach dramatically eases the partitioning task, which is a nightmare in homegrown systems," Dini says. One way (continued on pg 36)

Price per gate	Starting price	Price of fully loaded system	Rental available	Architecture
Starts at \$0.036	\$144,000, including maintenance	\$6 million for multiuser, 50 million-gate system; \$3 million for single-user, 50 million-gate system	Perpetual license or time-based license	Event-based
Starts at \$0.072	\$288,000, including maintenance	\$38 million for 256 million-gate Palladium II, \$15.4 million for 128 million-gate Palladium I	Perpetual license or time-based license	Cycle-based, scheduled
Less than \$0.01	\$11,000	\$135,000	No	Cycle-based
\$0.02	\$25,000	\$300,000	Yes	Both
\$0.03	\$120,000	\$440,000	Rent or buy	NA
Less than \$0.01	\$1345	\$140,000	Buy	Cycle-based to free clock
\$0.01	\$9000	\$69,900	No	Both
\$0.09	\$450,000	\$5.7 million	Yes	NA
\$0.01 to \$0.02	\$10,000	Less than \$500,000	No	Cycle-based
\$0.03	\$250,000	\$525,000 to \$925,000	Six-month minimum rental, perpetual license, or subscription	Event-based
\$0.06	\$1 million	\$1.275 million to \$1.875 million	Six-month minimum rental, perpetual license, or subscription	Event-based



MIPS USES IT ALL

Microprocessor-core vendor MIPS Technologies uses several forms of hardware-assisted verification to develop its cores. Donald Ramsey, director of CAD and information systems at MIPS, says that his group uses Mentor Graphics' VStation emulators and also builds its own prototyping boards for verification and for customers. Its processor cores have 500,000 to 1 million gates, so capacity is not an issue for running the core in either emulation or single-FPGA prototyping boards.

"We don't have a gate challenge, but we do have a complexity challenge in that the corner cases of general processor are deep, and the design requires an immense amount of directed testing, pseudorandom testing, and then prototyping and emulation as a verification vehicle, a debugging vehicle, and for software verification," says Ramsey. "When we create a core, we expect a number of software tools to work, as well."

Microprocessor-core design starts with RTL simulation on computer farms. The company then uses emulation upfront in the process. It has two VStation emulators and uses one for functional verification and the other, which connects to the system, for in-circuit emulation for debugging. For verification, the team uses emulation as a simulation accelerator for a fast, directed, random verification. "We get about a billion cycles a day out of our emulator," he says. "It is just like having a faster simulation engine."

MIPS uses another VStation for system-level simulation. "We can't run applications on our RTL simulator, partly because it is too slow and because, in a system environment, you have to abstract things because it is not just a core; it needs system controllers, memory, and all that," he says. "Then you run into a gate-count issue."

Rather than test the RTL version of the core for in-system testing, MIPS loads the design onto an emulator. Before using emulation, MIPS design groups used a previously developed prototyping board for

internal software developers to get a head start on embedded-software development. "It was basically a pc board with an AT form factor and PCI slots that allow you to connect disk drives, keyboards, and the like. Instead of a processor, the system had a daughtercard with two slots—one for an FPGA to program a system controller and another to place a MIPS processor," Ramsey says.

MIPS then created a new daughtercard that would allow the company to plug the emulator into the FPGA-prototyping system and perform in-depth system debugging. MIPS also directly programs the processor design into the FPGA on the daughtercard. FPGA prototyping runs faster than the in-circuit method and identifies bugs but lacks the in-

WHEN YOU STEP DOWN FROM 25 TO 1 MHz, YOU'VE STEPPED OUT OF THE BOUNDS OF WHAT A REAL SYSTEM LOOKS LIKE.

depth visibility into the code. Whereas the FPGA prototype runs at 25 to 40 MHz, the system with emulation runs at 1 to 3 MHz. This speed requires users to do an excessive amount of clock buffering to account for naturally occurring wait states in memory and hard-coded interval timers in an operating system, Ramsey says. An FPGA prototype is eight to 12 times slower than the actual processor and is better for debugging.

Many memory subsystems can tolerate that factor of eight- to 12-times speed decrease. "When you run the FPGA, it has all the characteristics of a normal system, but it just runs a bit slower," Ramsey says. "But when you step down from 25 to 1 MHz, you've stepped out of the bounds of what a real system looks like. Even if you have a slow memory on the

board, it looks like you return data on the next cycle. You have to be careful because you are really not modeling all the wait states in a real system because the memory and other peripherals are always available."

According to Ramsey, designers can also encounter issues with hard-coded interval timers in an OS. "You have to do some time-scaling, too," he says. Because of a shorter development cycle, MIPS used emulation and FPGA prototypes for hardware debugging on its last design. "We would run things on the emulator, quickly find a lot of issues, and then found that we needed a faster platform if we're going to meet schedule. The team then decided to use an FPGA prototype. It took a few minutes, rather than one to three hours, to find the OS bug."

Once the team found the bug, he says, it had difficulty locating its source. "In an emulator, it was 100% visibility, but, with this emulation system, you don't have full history, so you have to trigger close to when you think the event was to capture the event. In the emulation, you wind up doing multiple runs to find your trigger point, and, once you do, you have the smoking gun," he says. "In an FPGA run, it is the opposite. You know where the trigger point is, but you run it until you get the correct set of signals out, and then you go back and write a directive diagnostic and run it on the simulator to validate your theory."

In some cases, it proved impossible to write directive diagnostics; it became difficult to pinpoint problems because of the differences between time interrupts, memory speeds, and processor speeds in a complex processor with a lot of outstanding operations. One of the legacy problems with emulation systems is that users had to spend as much time debugging the emulator as they did debugging their designs.

EDA vendors have made strides in facilitating emulation systems, and Ramsey's group did extra work to remove the kinks from the emulator.

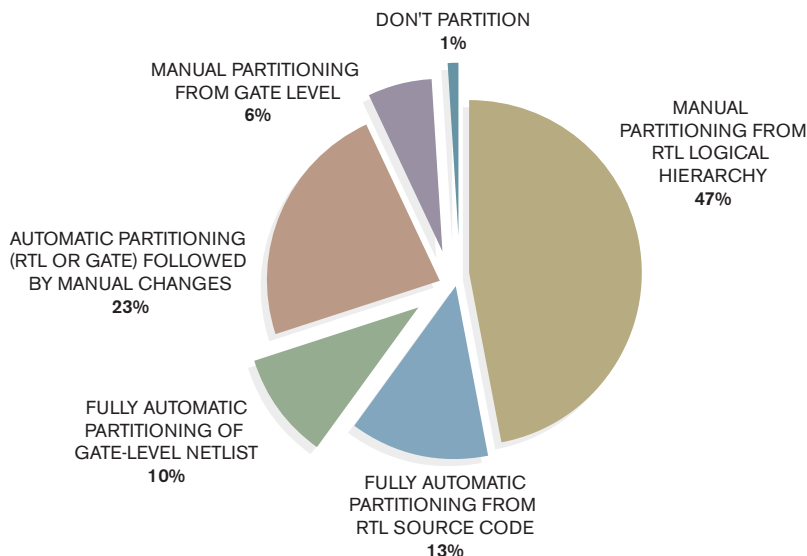


Figure 2 Even with automated partitioning software, programming an ASIC-prototyping system requires some manual partitioning (courtesy Synplicity).

to work ASIC prototyping into the design is to ensure before the design project that the ASIC developers design with prototyping in mind. Another approach is to arrange blocks in the ASIC to accommodate partitioning. Getting ASIC designers to buy into this idea is often difficult. “The ASIC designer is often the top of the food chain, however,” he says. “Verification and prototype are on the bottom of the food chain, even though they shouldn’t be.”

John Gallagher, director of ASIC-tool marketing at Synplicity, says that the partitioning task is generally easier the more familiar you are with the ASIC design. A Synplicity study with 450 respondents from the design community shows that 55% of RTL designers are responsible for partitioning the ASIC into a prototyping system, but 25% say that a separate verification team does the prototyping (figures 1 and 2).

ACCELERATION AND EMULATION

Design groups wanting to maintain the verification visibility of simulation while also speeding verification can do so by using simulation acceleration or emulation. With acceleration, part of the design compiles and runs on the hardware accelerator. With emulation, the entire design runs on hardware. Acceleration runs code hundreds or even thousands of times faster than pure simulation running on

workstations, but it is too slow to run a design in-circuit that connects to the rest of the system. Emulation, on the other hand, can be hundreds or even thousands of times faster than RTL simulation, and designers can streamline it to run at approximately 2 MHz, which allows its use in-circuit and as a way to speed functional verification. Designers can even tweak some systems, such as Cadence’s Palladium emulator, to run at tens of megahertz, the company claims.

Vendors build acceleration and emulation systems with big, commercial FPGAs, much like prototyping systems, or with custom processors. Those offering systems with custom processors claim that the devices are faster and more efficient simply because their architectures are designed for acceleration and emulation, unlike systems using off-the-shelf FPGAs. ATI Senior Verification Engineer Art Stamness buys into the use of acceleration but not emulation. His group develops the 3-D-graphics cores in the company’s graphics-processing units, and the group uses Tharas Systems’ accelerator to verify designs. “We map a portion of our simulation testbench onto a hardware accelerator that uses custom processors,” he says. “It gives us faster turnaround. Instead of running an FPGA netlist, we compile a binary, and that can run multiple magnitudes faster than doing the emulation.” It can take a day just to compile a netlist



with an emulator. “If you find a bug, you have to recompile, wait until the next day, and then, if you find another bug, you have to wait another day,” he says. Acceleration, on the other hand, can find bugs and have another netlist ready for more tests within an hour.

Acceleration is much closer to a simulation environment, and it can take a lot of code you wouldn’t otherwise feed into an emulator. And, acceleration provides a dramatic increase in performance over simulation but not the 5000-fold better performance that an emulator provides.

Stamness’ group works with a 150- to 250-PC server farm running Cadence NC-Sim with the Tharas accelerator in cosimulation mode. The Tharas compiler extracts the sizable code and creates a “stub” for the design. NC-Sim runs on the host machine that communicates with the hardware accelerator. Stamness suggests that using emulation involves not just start-up costs, but also support costs. “The amount of support infrastructure in

EDA VENDORS SAY THAT THE SOFTWARE THAT UNDERLIES EMULATORS IS NOW MORE SOPHISTICATED AND EASIER TO USE, AND MUCH OF THE COST OF THE SYSTEM COMES FROM ITS EASE OF USE.

hardware, personal, power, and lab space required is prohibitively expensive compared with an accelerator, which drops into a rack like any simulator box and gives faster performance,” he says. “Emulation still scares people and is reserved for the high end, where acceleration is in a good position to fill the space between the \$1 million box and slow simulation.”

Stamness’ views of emulation are typical of those who have invested in multimillion-dollar emulators only to find them gathering dust or obsolete some years later. Early emulators were difficult to program, and users often found them-

selves spending as much time debugging the emulator as the IC design. EDA vendors say that the software that underlies emulators is now more sophisticated and easier to use, and much of the cost of the system comes from its ease of use.

Tom Paulson, verification engineer at QLogic, is another emulation convert. His team uses Cadence’s Palladium PD II emulator for building fiber-channel-switch ASICs. The company has for more than three years been using the Palladium emulator to verify five ASICs. The QLogic team came up to speed on Quickturn emulation by using its QuickCycles



rent-to-own program and then eventually purchased a box. "It cost a lot, but we've kept it busy," he says. Paulson's division works with VHDL and uses Mentor's MTI ModelSim for initial verification. "We automated the process, as you would expect with companies that do something over and over again," says Paulson. "We build a set of ROMs, and each test calls out a different ROM, so the chip would operate slightly differently. We run a test or may run four or five tests against a ROM. We set up those configurations with TCL [tool-control-language] scripts. The end result of all that checking is simple: With a fiber-optic chip, you are checking that everything got to where it was supposed to go, and you account for all the fiber-channel frames—that is, that all the frames arrived correctly. We also do some internal checking and check status registers within the chip. It's a pass/fail when you get to that point," he says.

If the team finds a bug, it reruns the test

and then uploads the database to a workstation. The team then uses the debugging tool offline. "We have a certain amount of window we can view on a trace and a certain [number] of signals we can ask for in the beginning," Paulson says. The team is also starting to use Cadence's IXE 3.0 software, which reconstructs the design in software. "We are seeing remarkable increases in the amount of time it takes to get additional signals to look at," he says. "It has the Novas Debussy waveform, so it starts to look just like a simulator. From the time we find a failure, bring up the waveforms, and have the designer sitting next to us fixing the problem, resynthesizing the emulator, and rerunning it, takes an hour or two in some cases." **EDN**

REFERENCE

■ Moretti, Gabe, "Hardware tools aid engineers in design verification," *EDN*, Aug 30, 2001, pg 77, www.edn.com/article/CA152890.html.

FOR MORE INFORMATION

ATI
www.ati.com

Cadence Design Systems
www.cadence.com

The Dini Group
www.dinigroup.com

EVE
www.eve-team.com

Flexody
www.flexody.com

Gidel
www.gidel.com

Hardi Electronics
www.hardi.com

Mentor Graphics
www.mentor.com

MIPS Technologies
www.mips.com

Novas
www.novas.com

ProDesign
www.uchipit.com

Qlogic
www.qlogic.com

Synopsys
www.synopsys.com

Synplicity
www.synplicity.com

Tharas Systems
www.tharas.com

You can reach
Senior Editor
Michael Santarini
at 1-408-345-4424
and michael.santarini@reedbusiness.com.

