

IT'S NOT WIDELY DISCUSSED, BUT ONE OF THE CRITICAL COMPONENTS IN A SUCCESSFUL SOC IS ON-CHIP PROVISION FOR BRINGING UP THE FIRST SILICON.

Design for debugging: the unspoken imperative in chip design

BY RON WILSON • EXECUTIVE EDITOR

Testing and debugging present different problems. In testing, the goal is to determine as quickly as possible whether the chip is working correctly, with high, but not absolute, certainty. Chip-design teams now universally recognize that doing so requires the addition of DFT (design-for-test) circuitry on the chip, and third-party-tool and IP (intellectual-property) companies can aid in this purpose.

Debugging is quite another story. The goal of debugging is not simply to determine that the chip is not working, but to find out why it is not working. This inquiry is not confined to a few seconds on a test floor, but may last weeks. It is not automatic, but requires

the participation of the chip-design team. And it occurs at discrete points in the design cycle: during first silicon bring-up, during reliability studies, and during field failure analysis.

Given this profile, you might think that a good DFT strategy would be sufficient to meet the needs of silicon debugging—and, in fact, it often was. But with the growing complexity of SOC (system-on-chip) designs, leading design teams report that they are dedicating more and more planning, implementation, and silicon area to circuitry that supports debugging rather than test.

“Ten years ago, when we were designing with three layers of metal, this was not a big issue,” observes Bay Microsystems’ Senior Vice President of Engineering Tony Chiang. “If there was a problem with the chip, you could probe the metal directly to watch the circuitry, and, with a focused-ion-beam system, you could even rewire it. Now, with nine metal layers and 0.2-micron metal pitch, that’s simply not possible. We have to make the circuitry observable and controllable from outside the chip, without exceeding our cost goals or overrunning our schedule.”

That situation, in a nutshell, describes the world of designing for debugging.

A PANORAMA OF TECHNIQUES

Debugging is not entirely divorced from DFT. Broadcom, for example, has a corporate-level team of about 70 engineers who work with all the chip-design teams in the company on both debugging and testing, according to that company’s senior director of test-development engineering, Kris Hublitz. And Hublitz repeatedly cites DFT vendor LogicVision as a key partner in Broadcom’s chip-debugging strategy.

Others agree. “Design for debug is not too distant from manufacturing test,” says David McCall, a vice president at CSR (Cambridge Silicon Radio). “Both start at about the same point.”

That point, numerous design managers emphasize, is the quest for controllability and observability. In debugging, as in manufacturing test, the fundamental problem is to place the circuitry into a known state, start it running, and observe its behavior. In the days of medium-scale integration, boundary-scan techniques could adequately accom-

plish this task. Because chips had little internal state, you could thoroughly test them by sending the inputs through a known series of states, clocking the circuit, and observing the outputs.

With the advent of the microprocessor, things became more complicated. Microprocessors have lots of internal state, so simply forcing the inputs to a known vector and watching the outputs is not particularly informative. Early on, the industry tried a variety of techniques to make microprocessors debuggable—from providing scan for each cloud of logic between registers to relying on the same sort of trace, breakpoint, and single-step functions that minicomputers use for software debugging. A combination of those two did the trick.

Designers use the same suite of tools on the digital portions of SOC today. A separate collection of techniques serves analog and mixed-signal circuits. But no one approach can encompass an entire complex SOC. So, the process of designing for debugging comprises partitioning the system into independently debuggable blocks, implementing a debugging strategy for each block, and integrating these strategies into a plan for the full chip that keeps the user interfaces for individual blocks similar and minimizes the silicon resources that the circuitry requires. As a final step, designers must double-check that, using these debugging resources, the fully integrated operation of the chip is both controllable and observable, because

AT A GLANCE

- ❑ For complex ICs, silicon debugging requires on-chip hardware.
- ❑ The need for on-chip hardware goes beyond the needs of design for testing.
- ❑ Digital and analog blocks demand different strategies.
- ❑ Teams should have a full plan for bringing up silicon during architectural design.

you cannot observe some bugs by looking at functional blocks in isolation.

DIGITAL SOCs

The most basic form of SOC is a CPU core surrounded by simple, often non-programmable, peripheral blocks and memories. In most cases, the CPU core is third-party IP, and it comes at least with the option of an internal debugging kernel, the inclusion of which the software-development team usually insists upon. This kernel combines with the normal DFT circuitry that the design team implements for the peripherals to give sufficient observability and controllability to isolate problem nets. You can use the debugging kernel in the CPU core to stimulate all but the asynchronous portions of the core and to capture results. The kernel can also, by making the CPU read and write peripheral registers, stimulate and observe the peripherals, usually allowing designers to pinpoint a malfunction down to the level at which scan chains can take over.

But such simple SOCs are not that common today (Figure 1). More often, the chip will have several CPUs, or a host CPU core and several other formidable processor cores of various kinds. Even some of the peripheral controllers may be sufficiently complex that simply stimulating them via the CPU and observing the results won't be sufficient to diagnose them. And there will be multiple clock domains, often not synchronized with each other. Such chips call for sterner measures.

In this case, several tactics are available. One, suggests Broadcom's Hublitz, is simply to make the inputs and outputs of all the major functional blocks accessible to pins on the chip. This approach can mean a great deal of multiplexing. In designs that incorporate a large number of I/O and memory interfaces, the die may be pin-limited before the inclusion of any additional access for debugging purposes, so designers find that they have to reuse pins for debugging access. And simply bringing out the inputs and outputs of very complex blocks may be no more useful than exercising them with the host CPU core; designers may need to bring out internal signals, as well.

All of this multiplexing and routing adds up and may simply be impractical. Further, the resulting additional interconnect may mean that even if all the blocks are physically accessible from the pins, they may not be accessible at speed. And this problem is a serious one. "We find that we must test circuits—and especially the interconnect between blocks—at-speed," Hublitz says. "This is particularly true at 65 nm. Otherwise, you'd be lucky to find half the faults in the chip."

Hublitz emphasizes that a good DFT strategy supported by ATE (automatic test equipment) can enormously help the debugging process. "We do our first debugging pass on the ATE systems," he says. "After we are sure the chip won't melt, we give it to the design guys on the bench, and, from there, it's a joint effort." Hublitz says that the chip may repeatedly go back to Broadcom's test floor so that the ATE sys-

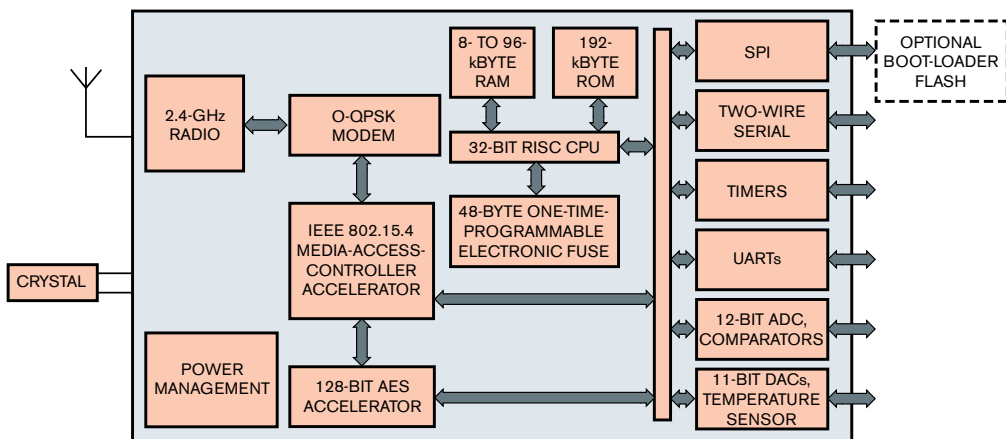


Figure 1 Single-chip network devices, such as this Jennic 513x device, present debugging engineers with a plethora of digital, analog, and RF challenges.

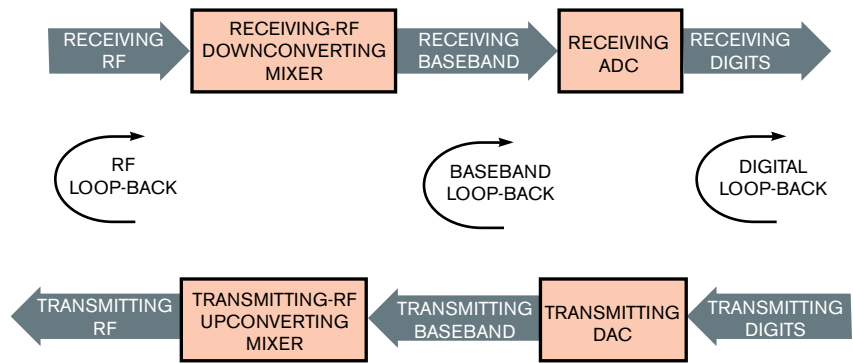


Figure 2 A series of loop-back connections renders this CSR transceiver more visible to the debugging engineer.

tems can collect large amounts of data or perform at-speed checks for the bring-up team. “It really helps to have your own ATE capability in-house,” he reports. “We have 28 systems, and we add a new one about every quarter. They are all primarily for debug, and new silicon has priority on the equipment.”

Even with access to ATE systems, however, some signals and states are inaccessible to a probe card. They require another strategy: internal stimulation and logic analysis. Sometimes, the only effective way to stimulate a block at speed and to capture its behavior is with circuitry built into the block itself. Bay, which organizes its network-processing chips as strings of independent processors, widely uses this technique, according to Chiang. Important blocks may have their own debugging kernels, including single-step and breakpoint capability and trace buffers to capture internal state in real time. This approach permits what Jun-wen Tsong, Bay’s director of logic design, describes as a multistage-verification process.

“First, we exercise the chip at the module level. In this mode, each module is isolated: We can inject enough state to start it running and then observe its behavior stand-alone.” These tests must be conducted at full clock speed for them to be accurate. In this way, the designers wring out each stage in the string of processors. At this time, designers also isolate the I/O ring from the internal blocks so that inputs go directly to the output FIFOs. Once Bay’s designers have independently verified the I/O ring and the internal blocks, they reunite the two and test the chip as a whole.

Obtaining meaningful data from full-chip, at-speed operation takes planning, however. The debugging kernels in the individual processors must be able to recognize not only local instructions and data words, but also big-picture data types important to the operation of the chip: packets and data grams. Additionally, a 36-bit bus runs through the entire chip, bringing out critical signals from any block to the package pins in real time, so that debugging engineers can watch the operation of blocks while the chip is processing packets at full speed. In addition, hardware monitors specific assertions, such as FIFO full/empty assertions, in real time. Broadcom has a similar approach. Hublitz says that the company’s wireless-LAN chips have enough internal debugging hardware that engineers can track vector magnitude all the way through the chip, from input to baseband to output.

Once they’ve isolated a problem to a function within a block, debugging engineers can turn to a lower level of diagnostic tools, based on familiar DFT strategies. “We have clock control for triggering and single-stepping within the blocks, and scan for what we consider to be the important signals,” explains Bay’s distinguished engineer and silicon architect, Barry Lee. “Ideally, we can see exactly how a particular pipeline is executing down to the pin and register level.”

THE ANALOG CHALLENGE

When analog circuits are involved, everything changes. “We partition the analog apart from the digital circuits for debug,” Lee explains. “The debug tech-

niques for the two are different. In the analog world, you want to peel open the loop-back paths. And you may have to bring everything out to the package pins." Because the primitives of activity in analog circuits are not synchronized to a clock, there is no way to capture them.

Analog, like digital, designers have seen their ability to probe and experiment with their designs vaporize with shrinking geometries, observes Analog Devices' fellow, Paul Ferguson. "We used to simply have a laser cutter on the probe station, and, if we wanted to modify a circuit, we did. Later, as geometries shrank, we moved to focused ion-beam systems. But they are only really useful at pitches of about 250 nm or greater. That means, practically speaking, that if you are working in a 65-nm process, you can make changes only to the top two metal layers."

This situation has led to an interesting change in analog-design style, according to Ferguson. "We were recently doing a PLL for a 90-nm design and found that we would have to complete the VCO [voltage-controlled oscillator] before we had really proper models. So, we brought some lines for adjusting the gain and some other parameters up to the top metal layers where we could get at them. It really helped in the debug process."

Matt Ball, mixed-signal project engineer at single-chip-radio vendor Jennic, also emphasizes the importance of bringing up critical analog signals where you can get to them. "We put in as much programmability and digital trimming as possible," he says. "Some things have to be metal-trimmed, though, and we bring all those locations up to a single mask level for accessibility."

Beyond bringing up live signals to the top metal layers or to the package pins, today's analog designers have other weapons at their disposal to set and observe the state of their circuits. The foremost is the reality that at fine geometries, there is intense cooperation between the analog circuits and the digital circuits that calibrate and monitor them.

CSR's McCall says that in its designs, ADCs monitor many points in the analog circuitry for digital supervisory circuits. These points naturally give debugging engineers access to the behavior of the analog section simply by bringing

MORE AT EDN.COM



 Go to www.edn.com/070621df and click on Feedback Loop to post a comment on this article.

the outputs of the converters to the outside of the package. "Often, important analog signals are going to get digitized at some point anyway," says Ball. "So, why not pick up the samples, filter them with your on-chip DSP, and output the result so we can see it?"

Designing a filter or an amplifier so that digital circuits can trim all its important electrical characteristics might seem like massive overkill. But it can make the difference between first-time-working silicon and silicon that has to have two new metal masks before debugging can even start on the digital portion of the design. And, given the increasing variability with which designers must contend at processes smaller than 90 nm, this much digital trimming may be necessary anyway, just to yield an adequate number of working chips.

But how do you see to trim? For signals of reasonable precision and frequency—the IF (intermediate-frequency) signals in a radio chip, for instance—you can simply use careful routing and analog multiplexers to bring the signals out of the package in test mode. "At IF, the buffers can be quite good," Ball says. "You can get signals from important nodes out to pins and see what you need to see." Analog Devices' Ferguson agrees. "For debug purposes, you often don't need more fidelity than an analog multiplexer can provide; you can see oscillations or a 20% gain error quite well."

If you can't bring the signals out of the package, you can sometimes route them to on-chip data converters. "We will typically have an auxiliary ADC on the chip to monitor die temperature, battery voltage, and the like," Ferguson explains. "We put a huge [multiplexer] on the front of it and use it to examine other nodes in the analog section during debug. But be wary: The extra measurement circuits you put in can corrupt something else. For instance, turning on the multiplexer to observe a node may add just enough capacitance to stabilize a circuit that was oscillating. And, if you inadvertently cross power domains with your debug signals, you can introduce

sneak current paths that you hadn't counted on."

Ball echoes the warning that you have to be selective with this approach. "The 10 or 20 fF you incur by buffering an analog signal can change the behavior of a node," he concurs. Jennic tends to build its debugging provisions around only those areas that have presented problems before, such as bandgap cells. "We tend to put in bypass circuits, just in case," Ball adds. This conservatism can minimize the chances of disrupting functional circuits.

With planning, good luck, and a bit of elegance, it is possible to reuse functional blocks for debugging purposes. Many analog signals terminate in a data converter and so are at least partially observable through it. Ferguson points out that you can easily switch sigma-delta converters to operate as filters, providing visibility into the incoming analog signal. Alternatively, you can carefully route their bit streams out to pins, giving observability of both sides of the converter. Once you digitize the data, you can use CPU or DSP blocks to condition and compress it or to test assertions against it.

It is also possible to build debug intelligence—the equivalent of a simple network analyzer, for example—into a block. A loop-back path can use a transmitter and a receiver to check each other (**Figure 2**), and a little more circuitry can extract the resulting analog waveforms. "On our Gigabit PHY [physical-layer] designs, we are capturing some analog data in the PHY block," Broadcom's Hublitz reports.

OUTLOOK FOR THE FUTURE

It is not hard to imagine a scenario in which, during early system design, each functional block receives enough self-test capability to diagnose itself during at-speed operation, down to the level at which the DFT-scan chains can take over. This approach would usually require an input buffer or a signal generator to stimulate the block, an output capture register or ADC to observe it, and enough internal breakpoint and trace capability to reveal the block's internal workings. Some SOC design teams now do this planning. Actual implementation then becomes a compromise between the level of debugging support the architects feel necessary and

FOR MORE INFORMATION

Analog Devices
www.analog.com

Bay Microsystems
www.baymicrosystems.com

Broadcom
www.broadcom.com

Cambridge Silicon Radio
www.csr.com

Jennic
www.jennic.com

LogicVision
www.logicvision.com

the amount of overhead the design can tolerate.

Taking the concept a step further, a designer of an elegant system can come up with ways to repurpose some functional blocks to serve as signal sources or capture devices for other blocks. The auxiliary ADC is an excellent example, but more such opportunities exist. The addition of a fast data converter, for instance, might turn a signal-processing block into the equivalent of a network analyzer or a digital oscilloscope. A few additions to the control logic might convert a buffer-SRAM array into a trace buffer.

In this manner of thinking, the functional blocks on the chip become a pool of debugging resources, available with only the resetting of a few multiplexers and mode switches. But this process requires forethought. Such an organization impacts floorplanning and global routing, so it must occur at the outset of the design rather than during late implementation.

And it is a process that could stand some tool support, as well, Ferguson argues. Elaborate tools exist to automatically install such structures as scan chains, scan controllers, and vector generators. And, DFT hardware can be indispensable in register-level diagnosis of problems. But no such tool support exists to create debugging structures. Ferguson, for one, would like at least to see an inspection tool that would rate a mixed-signal block for observability and controllability, as well as to scan it for simple mistakes. Ideally, a tool could traverse a design and propose a debugging architecture and process. But that matter is for the future. **EDN**

You can reach
Executive Editor
Ron Wilson at
1-408-345-4427,
ronald.wilson@reedbusiness.com.

