Design for Manufacturability

Design for manufacturability (DFM) has always been important but is vital as we move to 90-nanometer technologies, and it requires the efforts of design, test, and yield engineering. Historically, test engineering has been concerned with identifying defects before shipment, while yield engineering has been concerned with preventing defects from happening in the first place. With 130 nm, the line between defects and process variation has blurred to the point that in some cases it is no longer meaningful. This means that yield and quality level are becoming more design dependent than ever, and that test effectiveness cannot be guaranteed without strict DFM guidelines. Without DFM, no company will be able to successfully make the jump to 90-nm technology. The panelists explore DFM from a variety of perspectives.

IEEE Design & Test thanks the roundtable participants: moderator Rob Aitken (Artisan Components), Stefan Eichenberger (Philips Semiconductors), Sandip Kundu (Intel), Gary Maier (IBM), and Hank Walker (Texas A&M University). *D&T* gratefully acknowledges the help of Kaushik Roy (Purdue University), our Roundtables Editor, who organized the event. Special thanks go to the IEEE Computer Society for sponsoring the roundtable.

Aitken: There are many definitions of design for manufacturability. Is it a set of design rules or recommendations? Does it come from a foundry, an IP provider, an EDA vendor, or someone else? Can it even work without a vertically integrated operation?

Eichenberger: What is DFM and what is a design rule? If the chip doesn't work at all—has zero yield at the target specification—we would call the culprit a design rule violation. If we don't see any manifestation of an issue at all, we're fine, and then there is this little area in between where the chip kind of works, but not really, and then we call it DFM.

Aitken: A design rule checker (DRC) has historically always been a binary system: You either do it this way or you don't; you have this metal spacing or you don't. But I think with decreasing feature sizes, it's obvious that wires don't magically work perfectly when they're 0.12 microns apart and cease to work altogether when they're 0.11 microns apart. There's a transition and a slope, and DFM lets you take advantage of that and to think of a DRC not so much as binary—yes or no—but to think about the tradeoffs that allow something to be more manufacturable than it might be otherwise.

Kundu: I don't think that design for manufacturability necessarily or exclusively belongs to the design engineers. The individuals who make the masks, for example, know a lot about what happens when a design enters the manufacturing process, and they do mask compensation. For example, after the designers do the layout and hand it over to the mask makers, they say, "Okay, I find that your metal track utilization is only 20%, and my metal track utilization needs to be at least 50%." So the mask house inserts these metal wires, which really don't connect to anything but are simply there so that when you do the CMP [chemical-mechanical planarization] process there's no dishing. As far as DFM goes—I don't know at that point if it makes sense for design engineers to be worrying about that. For example, if I'm an ASIC designer, the heuristics that a certain fab uses may be totally different from what another fab uses, so should I worry about how these things will be more manufacturable downstream? Or do I just leave the design at some point, hand it over, and let somebody else worry about it?

Walker: To me, the big difference is that in the past you could hide things in the mask processing, but now there's too much variability that can't be hidden anymore, or you lose too much of your capability in the process if you hide it from the designer. So the challenge from the design-for-manufacturing point of view is that those variabilities now have to show up in the design process in terms of the tools, like the physical design and static timing analysis, for example. Also, it's a challenge to try to encapsulate those variabilities into the design tools so that the designer can make sure that they've handled them explicitly, because they can't just be dealt with on a downstream basis only. In the microprocessor world, handling variability explicitly is not new, but in the ASIC world, it has now become brutally clear in the current technology node that variability can no longer be hidden.

Walker:
In the ASIC world, it has now become brutally clear in the current technology node that variability can no longer be hidden.

Maier: I work in yield management at IBM's new 300mm fab on 0.13- and 0.09-micron technology, and I agree with Hank [Walker]. We need to emphasize that variations that occur during one process step can propagate to a problem in another step of the process. If the first variation didn't occur, the second event likely wouldn't happen. So these variations interact with one another more often today. This occurs because of the minimum space and design features we're working with today. Very small variations that used to just affect AC performance are starting to affect overall functional yield. It's become much harder to guard-band the process window. Some of the ownership of this problem lies with quality design. We need to take some of these process variables that occur in the line and feed them back into our design teams, then build them into our design tools, and try to bring new features into the next generation of design tools to have DFM built in.

Some examples of DFM items that should be discussed would eliminate mask houses' having to perform data postprocessing on incoming designs. Certainly things like an intelligent DFM router, with a wire spreader built in, obviously make wider spaces between long lengths of wires and could minimize the effect on certain types of defects in this critical area. Everyone wins: The cost of postprocessing data in the mask house is eliminated, cost of goods sold is reduced, and yield and time to market are improved. I don't think we'll ever get design tools intelligent enough to understand these new technology multivariations without process and design tool integration, so these things will be designed in. This integration provides the channels to quickly feed back all the variables we're up against for better quality designs the first time out. More focus will also be required on things like static timing and how it's affected by multiple physical and parasitic attributes of the manufacturing integration process.

Kundu: If we look at today's chips, we have multiple sets of design rules. For example, for dense arrays, we have one set of design rules; when we have an analog design block, we have a separate set of design rules. On top of that, there may be things directly imported, where the fab house—instead of publishing design rules—will provide hard, design macros. For example, many of the foundries today will offer us I/O cells, so we don't need to worry about I/O design. In some cases, there are memory compilers for designing specific instances of memory arrays, and we don't need to get into the nitty-gritty design details. What these encapsulations—or multiple sets of design rules—do is that they transfer the empirical knowledge from the fab to the design house. A designer doesn't need to know everything, because if there are some critical blocks—like the really dense arrays—they could be co-owned, and critical knowledge could be supplied by the fab to a memory compiler. I don't think it is incumbent upon a design house to know everything about a manufacturing process.

Aitken: We need to distinguish between different classes of designers. So when we're talking about arrays or I/Os or standard cells, the people designing those things have to know the DFM rules; they have to understand the process. And it's necessary for the people who are designing the overall chip to be convinced that these various components are meeting the DFM requirements. Their interaction with DFM is much more likely to be at the routing level. In that case, if the routing tools are DFM-aware and know to spread wires and know what the rules are—know how to double up vias and that kind of thing—then that's the sort of DFM that they ought to be interacting with. But I agree that those designers shouldn't care what the poly geometry rules are. They should just be confident that somebody else has addressed them earlier. But that's not necessarily the foundry; it could also be an IP provider.

Eichenberger: Which raises the question, How do we verify these rules? You [Rob Aitken] said something interesting, that DFM is running across generations. What you're really saying is that the memory designer needs to know these rules up front, and that's hard, because he is in the very early stage of the process life cycle when a lot of things are still bubbling around. And once the memory designer hands something off, the system designer wants to be assured that he can use that memory exactly as is. So how do we go about handling the characterization of these things?

Maier: I see several problems where our process window, guard bands, and controls worked well 10 years ago but which have drastically changed within the past three or four years. Let's discuss DFM in standard cells versus custom: Standard library cells certainly have a lot of upfront information. Library designers understand how to lay these out using provided technology-design ground rules. There are also dependencies on whether the cells are analog or digital. It's an integration problem. It's more than a back-end or front-end issue—we're wrestling with routing-tool weaknesses in the integration, for example.

A second point is that we're qualifying new technologies into manufacturing much faster today. New product qualifications and manufacturing ramps now take about six months to a year instead of the year and a half or two years we used to have. We need to be able to quickly feed back process ground rule changes or build flexibility into design tools and the design methodologies. This may provide a more accurate, dynamically changing design rule manual during this short cycle.

Some of the things we're doing at IBM include speeding up the process for a better feedback mechanism—but more importantly to try to get a better qualification vehicle—to help set the process up. The industry is seeing a larger distribution of the type of product coming in from outside. These new SoCs are no longer ASICs. They're custom designs; they're like microprocessors. Some of these new foundry products coming in—and the new SoCs—are very complex designs; I mean complex in the way of performance, power, number, and complexity of circuits—mixed signal and so on. Integration is very complex from a manufacturing (physical and parametric) standpoint. All the issues we've been wrestling with for high-end custom designs over the years are now coming into the foundry. We have to discover ways that we can do this economically in a foundry environment.

Kundu: Essentially, we are converging on a position that because the chip design happens at different stages, phases, and partitions, it means different things to different people to be doing design for manufacturability. For example, if you are doing chip integration: To you, the DRC rules probably represent DFM. You probably don't want to know more levels of detail than that. On the other hand, if you're going to be taking huge risks, you probably need more collateral in place. An example would be matching the sense transistors of a sense amplifier. Clearly, many designers don't spend time thinking about what engineers in the fab are doing when they're running process qualifications and publishing design rules such as "Sense amplifier sense voltage should be no less than 80 millivolts." They are applying their generational learning experience with the manufacturing process and with real-life designs.

Clearly, if you're going to be taking risks with design, you need to be in that process, involved with doing that piece of the work. By and large, it doesn't make sense to educate every designer about the manufacturing process beyond what the standard DRC rule provides, because we have set up these rules to dissociate the designer from having to know all the details in the first place.

Eichenberger: Some things you have to put in the tools, not in the rule book. Specifically, I mean something like wire spreading: If you finished calling your router, you don't want to have to go back to check that optimal spreading has been done. You want to be sure the tools took care of that.

Walker: We want the tools for things like the interconnect, but we also want to have the IP suppliers solving this problem as well. If we're in a fabless environment, the IP suppliers have to have enough leverage with the foundry to be able to get enough of the inner working data of the fabs to make sure their IP is manufacturable, as opposed to an IDM [integrated device manufacturer], where in the same company design and manufacturing engineers can talk directly about manufacturability. Otherwise, if we're having a custom chip manufactured in a foundry, we need to have enough volume in that foundry to get access to the process data.

Aitken:
As an IP supplier, we have to look across different foundries, and ask, "What are the commonalities here?"

Eichenberger: Even for an IP supplier, if it were memory, then I absolutely agree with what you're saying. But if you're doing an ARM or a DSP core, then you need to have the tools to implement those rules for you. Otherwise, the rule book can't be applied to actual practice.

Maier: Therein lies the problem. We have very sensitive IPs with our processes. We must be able to provide the kind of data and the kind of problems that occur in the fab to an outside tool source or IP vendor so that they could develop IP for evaluating designs to deliver higher quality. But it's a catch-22 situation. There are certain things I'd like to share with the outside world and tool suppliers to improve these design tools and deliver better quality designs the first time out—the goal is to drive higher quality designs from this process data. So we've got to find a way to share new information in these fabs.

Many multivariate things can occur in a fab that cause real yield loss, and they don't occur unless they all happen at the same time. And without the design tools having information—such as line resistance, via resistance, line width variation across the chip and across the wafer, V_T, leakage, and so on—it may take combinations of three or four of these things that cause real yield loss. I'm talking functional yield loss, not just performance and reliability failures.

But we can't expect IP designers to have all this in their heads. This is not just from a physical perspective but also from a circuit perspective. What should we do to check the design to be sure it's going to work in 0.09 microns and beyond? I think the only way that's done is to build some intelligence into the tool, then to use more of these process variables in design, process, and performance simulations and in DRC checking.

Aitken: What we're dancing around here is the idea that there's a strict algorithmic representation of these rules. And I think up to 0.18 microns that was true, that we could have a DRC and it would effectively tell us whether this was going to yield or it wasn't. But when we're in 90 nanometers, or even 130, it's not always clear. Some things are easily expressed with DRC rules; other things are less well expressed. It's a case of "this is good; this is better." And we can't necessarily say, "Here's a strict rule for it." A lot of these things currently get expressed as DFM recommendations, such as "Try to avoid having vertices." And some of the recommendations can't even be expressed in a way that's checkable by DRC; they're just sort of guidelines.

What I think we'll see going forward is that as an IP supplier, we have to look across different foundries, and ask, "What are the commonalities here? What are the sorts of things that everybody is running into?" And then we get to the point where we ask the vendors who build fab equipment, "What sorts of things are problematic for your fab equipment?" That lets us look at what the commonalities will be and what we can do, given that the recommendations we see at 90 nanometers are almost certainly going to be design rules at 65 nanometers. How can we make sure that the design is correct going forward?

Kundu: The main point is that encapsulation is still the key. We don't really want to educate designers about how to improve yield on every process-related issue. For example, some of the large design projects I've been involved with have had a large lead time. I've found that, aside from the yield issues, even the files that I worked with early in the design process are completely different from what the process looked like when we entered manufacturing. Because when we start designs three or four years ahead of the manufacturing process, what we're really working with is a good-faith estimate of what the process technology will ultimately look like and what the various parameters are. Often, the problems that are predicted four years ahead of time receive the most attention, yet they can become the least of the problems in the manufacturing process. So it's probably not a good idea to overwhelm the designers with requirements up-front in the design when we don't know what in the process is going to have the greatest impact on yield a few years down the line.

Walker: We're talking about inexact manufacturing, about design for yield maximization or cost minimization, and that's inherently a statistical design problem. What that means is that the tools would have to have statistics, and often those statistics are not available because they're not collected in the fab because they're not needed for fab control; they're only really needed by the design tool. A good example is having a handle on intra-die process variation and intra-die line width variation. And so we often end up with a chicken-and-egg problem where there are no tools to consume statistical data because it's not collected in the fab, but the fab isn't going to spend the money to collect the data because there's no one to consume the data.

I've faced this problem before—where we asked questions of the fab and we heard "Well, I don't know," and so we could build a tool but couldn't do much with it because we couldn't get any good data to feed into the tool. At an IDM that's a more solvable problem because it—the design and fab—is ultimately one cost center. But in a fabless environment, it's not clear; one side is eating the cost and the other side is getting the benefit in some sense. I don't know how that's solved in the long run.

Maier: That's a constant battle in most fabs: never getting enough data to do the statistical analysis they need to do. If the fab does collect it, the data is likely not sampled from the right places, as it's sampled across the wafer. What's required is more frequent sampling, better resolution with sampling, spread across multiple lots, wafers, and dies. We also need to explore new test tools and methods like product intra-die, contactless, electrical data sampling. Realistically, it's time to decide what it takes to make these designs profitable, not just manufacturable.

I can tell you the fabs are not going to take it on the chin any longer. If we don't get better-quality designs, they won't get manufactured because no one will be able to afford the losses resulting from preventable design issues. Fabs can likely get the statistical data needed to make better quality designs. Instead of trying to make measurements and statistics specific to every variable, why not measure a cumulative set of parameters with a single measurement? We always seem to limit ourselves by the old way of doing things. My goal is to provide more economical, higher-quality measurements, and reduced data volume through new fab test tools and methods. Semiconductor process tools have made breakthroughs but test and design tools and methods have lagged.

We've got to start taking these measurements on the product, not just on test structures, which means we've got to think about new technology for contactless test and probing. For instance, there have been papers at this conference on photon emission, lasers, and other techniques for contactless power, signal injection, and measurement. With the much lower power of devices today, this technology is much closer to reality. This kind of statistical data gathering changes the data we collect for the future, and I think it's more meaningful. Basically, we can start to extrapolate what multiple variables are affecting the design, through a single measurement.

Eichenberger: I think it comes back again to learning over multiple generations. We're always too slow to bring what we didn't anticipate beforehand into the current node. But we need at least to be able to collect this data, so that the next time around we can use it in the next node.

Maier: Correct; data gathering needs to be fast. Volume, yes—there needs to be a lot of process volume, and the data collected needs to be quickly and efficiently fed back to design and testing so they can react to it for the next design. The point's been made that the design teams can't be expected to control these things that happen in the fab. We're not asking for design to control the fab—that's our job. What we're asking for is help. I see a lot of careless design today. Perhaps it's innocent ignorance or the result of auto place-and-route design methodology. Either way, we need to work together toward a new generation of design and test tools.

I would like to see, as part of the fab's characterization team's responsibility, the critiquing of design quality before committing yields, through a DFM, DFT, and diagnostics analysis. Our analysis is going to be centered on how we price parts coming into the fab, so some of the yield burden ultimately falls on the design team. The yield we'll get from the fab in the future will be based on the quality of our design. The yield will be based on not only DFM and meeting the design rule manual requirements but also on things in a kind of gray area—what we call design methodology. I can see a business model where the customer is rewarded with a higher yield parameter file and lower prices if they bring in a quality design integrating corrections on fab advisement. On the other hand, they could be penalized or rejected for poor quality. The decision to accept or reject designs will inevitably hinge on the cost savings to the fab and the customer. This new analysis will be part of the business sizing.

Aitken: I agree. The cost eventually lies on the customer as well. Often the design team is the closest in the supply chain to the customer; the design companies are the ones who are going to have to sell this product to somebody, and so they're going to have to make sure they get enough yield to cover the cost. The design time issue is really interesting in that respect. If we're building a product that has a life span of a year and we've got three years to design it, then we can make all sorts of design tradeoffs that are not possible if our product is only going to last six months.

Similarly, we have a different tradeoff if we're going to make a million of these products a week versus a hundred every couple of months. And that allows us to distinguish between DFM and design for yield. I like to think of design for manufacturability as what we do to make sure we get nonzero yield. And design for yield is what we do to optimize that yield once we've got it.

If you're a low-volume person, you really don't care about optimizing yield very much as long as you get to a certain minimal level. But if you're in a highly competitive, high-volume area with low margins, then you should care about optimizing a couple of points of yield; it's worth making those decisions as part of design.

Kundu: There are two key points that I just heard. One is the statistical design process, and the other is the cost that goes into the design. We've been accounting for the statistical process, to some degree, in the design for years, just by looking at the min and the max delays. In some ways, when we have statistical variations, we're capturing plus-or-minus two-sigma points, or plus-or-minus one-and-a-half-sigma points, or plus-or-minus one-sigma points.

By choosing those two ranges of numbers, we basically are focusing on design for manufacturability. The downside is that if we really want to go overboard and choose plus-or-minus four-sigma points, for example, we have to make sure that there are no min delay problems within that range. We will end up doing too many buffer insertions and paying in terms of either oversizing the devices, not meeting our cycle time goal, or taking too long through the design process to really converge on a design.

Having an aggressive DFM goal up front for timing convergence will affect the number of iterations we need for design convergence. It's going to affect design quality in terms of power-performance tradeoffs. Essentially, we cannot just focus on one thing to the detriment of the others—we have to take a holistic picture. Unfortunately, I haven't seen a very scientific or systematic study of that economic model yet. Not to say it doesn't exist, but until now it pretty much exists in the heads of those who are making these choices early in the design process, which are based much more on intuition and experience than on scientific data generated through some process.

Walker: To follow up on that point and on a comment Gary [Maier] made earlier about using the product as the process monitor—it's the phrase I use for that—process monitoring touches on the issue of using testability features to aid in the manufacturing process. If we can collect that much more data from the product, then we can feed that back into the line to improve the yield. But that can cost us die area, and so on the one hand we have a larger die, which is normally bad considered from a yield point of view. But if it gains us enough data in the test side, then we can use that data to improve yield to overcome that larger die. I don't yet see a good way in this disaggregated business model to make those tradeoffs, although in IDMs it's a little bit easier to look at them.

Maier: I think it's important—and this again falls under fab and the characterization teams too—to understand that we need statistically accurate models based on the end circuits we have, whether it's bit circuitry, process-monitoring vehicles, or whatever. I'm advocating a new approach for utilizing structures that are already built into the product; they're not additional test structures. For instance, some of these SoCs have long parallel buses in many applications, and they're integrated at different levels from the back end.

There are built-in process and defect monitors: We can certainly use those things and ask our test vendors—our ATPG or pattern generation vendors—to target certain patterns, specifically to target either a bridge fault or an open, for instance. Using a simple test methodology change can reduce time and cost by reducing time to detect the root cause. Perhaps some DFT can also be used to enable this technique.

Now, about the design for manufacturing versus the design for process issue: I focus my efforts on design for process. So once we've got the baseline yield learning up, it's a difference between whether we're profitable or not, or getting serviceable yields or not.

Aitken: One thing that's interesting in testability is the distinction between logic and memory. Once we get the baseline yields up in a memory, we can use redundancy and repair to cover some of the yield issues that might emerge, the minor problems where a bit goes bad here or a column goes bad there. We can use redundancy and repair to fix those problems. But there isn't a similar solution in logic. Logic either works or it doesn't, which leads to the need for bridging fault patterns, or for multiple detection ("n detect") and the like. It's interesting to speculate on whether in the future some kind of redundancy and repairability will be required in the logic itself.

Walker: In the past I worked on wafer-scale integration. And we always focused on the redundancy in logic and reconfiguration. That's come back again as a popular idea associated with carbon nano tubes and nanotechnology. In the wafer-scale field, it turned out to be more economical to simply reduce defect levels to the point that we could yield at high transistor densities, so as a packaging technology, wafer-scale integration was never competitive; it was just a brute-force solution. It's not clear if the brute-force solution—just get rid of all of the atoms you don't want—will work in the future, or if we'll have to have redundancy. We're just beginning to look at that.

Kundu: To address Rob's [Aitken's] point: In memories today, we know we can do something to bring the yield up by redundancy techniques of adding spares and so forth. On the logic side, to some degree we're addressing that issue by having a different set of design rules. For example, the minimum transistor width you can have in the logic is significantly wider than the minimum width you can have on the memory side. Similarly for line spacing: We have different rules on the logic side than on the memory side. By being conservative, what we're trying to do on the logic side is make sure that the yield there is comparable to what we have on the memory side. And that's basically where most designers stop setting the limit—on the conservatism of the logic side.

Eichenberger: Based on that, beyond a certain point it wouldn't make sense; is that correct? Scaling would make sense for getting higher and higher density out of your arrays. That will continue to make sense.

Aitken: It may make sense, I think, at an architectural level. So I don't think that it makes sense to say, "Well here's my AND gate; I'm going to have redundant inputs on my AND gate and some kind of error correction on it, and produce a new 72-transistor AND gate." But on the other hand, if you have a design that, say, has 50 or 60 embedded microcontrollers of some kind or another, it might make sense to have more, or more spares, and then just have a muxing capability to allow repair. I've heard of people contemplating that sort of thing. I'm not sure that anyone's doing it yet, but they're certainly thinking about it.

Kundu: By looking at what's happening with some of the server chip designs, I see a clear trend. The number of transistors we're adding on the array side is significantly higher than the number of transistors we're adding on the logic side. So the benefit gained from the density of arrays is significant.

Eichenberger: But even on the memory side, there's a distinction between large arrays where you can really benefit from sound validation, and many, many small arrays where you're looking at irregular topologies, similarly as in logic. You have all these column row decoders, sense amps, and so on, which make things look almost as complex as logic because the periphery is dominating your chip area.

Kundu:
I see a clear trend. The number of transistors we're adding on the array side is significantly higher than the number of transistors we're adding on the logic side.

Maier: The critical area is not only the density of the RAM but also the peripheral circuitry that dominates smaller RAMs. The critical area yield problem we associated with large RAMs before is now distributed across the entire chip in much smaller RAMs. We can't possibly have redundancy for all these smaller RAMs, so it's almost like the logic problem, where we can't recover from those yield losses. It's the same critical areas but distributed across the chip.

The concentration of critical areas is key, but the density of this distribution is getting higher, so it's interesting how we're wrestling with the ability to repair problems that we can't recover from in fab versus trying to be proactive by fixing, up front, the quality of design. And I would certainly advocate redundancy, to get by in the short term, but we've got to get away from fixing a problem after the fact; to focus more on being proactive by providing better quality designs the first time. This is a responsibility owned by the fab, IP, and tool vendors, not just the design team. We need to work closely together to understand the issues to develop better design and test tools. This will ensure that we develop a profitable design versus a manufacturable design.

Eichenberger: I think that points back to the question, how do you evaluate your statistical data? How do you feed back to your IP supplier and your memory supplier what you actually see in production? Often, designers of new chips who go into 90 nanometers—and soon go into 65 nanometers—say, "Why do you bother me with this 120-nanometer or 180-nanometer design?" But that's actually where they get the statistics and where they could do the learning.

Maier: There needs to be a continuous flow; it can no longer lag by six months like we used to do in previous technologies. New ground rules are being set. We need continuous real-time feedback to the design and test teams. If it's going to be a part that runs only six months, it may make sense to rip a new mask at a single metal level.

It's not always a solution in real time. We need the ability to process data and feed it back in a connecting set of tools and databases so we can correlate the data more efficiently to improve our designs and manufacturability.

Eichenberger: I think the data mining is essential. You mentioned the bus structures that you have: If you have one design produced 100 million times, you understand it, you know it, you find your structures. If, on the other hand, you have 100 designs, each produced 1 million times, then you better make sure that you have a good database so that you can learn across these 100 different designs, because they're all seeing the same physical defect mechanisms. But you need to understand where they occur across those 100 layouts and filter out the similarities.

Maier: Good point; you would then leverage your volume designs for each technology and for all products and applications.

Aitken: Yeah, and this is where I think this disaggregated model actually works well, because the foundries and the IP suppliers and so on have the experience with the 100 different designs and can try to make that information available to the person who's just doing one design.

Walker: Does this apply to soft IP or hard IP? Or will it apply to more hard IP in the future, because you get it right and then it's always right?

Eichenberger: That's basically the question: How good is your database? If you know only that you have a certain core in there, then you need to rely on hard IP. But if you know you have certain critical areas there, because you have the ability to extract that from a design database, then you can also apply it to soft IP.

Walker: But that's only true if you understand all the different interactions for the soft IP, right? Versus the hard IP, which is basically not changing from design to design. So you don't have all the different environments that it's going in affecting it very much, assuming that it's a big enough chunk of IP.

Eichenberger: It's two sides of the same coin. On one hand, you have the advantage that then you compare the hard IP configuration in different top-level designs. On the other hand, you have the disadvantage that if there's a more subtle problem in that hard IP, you'll never figure it out, because you have the consistency on all designs.

Kundu: I disagree a little on the point that if we have a very high volume part, we can afford to be lax about evolving empirical design knowledge because we can learn everything from the fab side. So no matter whether we have many parts, selling in the thousands, or a few parts, selling in the millions, the problem essentially is evolving a full set of empirical knowledge that lets us maximize yield. Even if we have very high volume, we can't afford to be blind to this fact and simply do the learning purely on the fab side.

Eichenberger: I didn't say to relax; I said we have an easier task of understanding what we observe on the fab side. We have an easier learning cycle with one big team understanding our design very well than with many small teams understanding a little bit of each and every one of their designs.

Kundu: While we're addressing this memory issue: We cannot underemphasize the importance of evolving a good set of empirical rules. For example, if density is key, as it might be in some of the very large arrays, we clearly need to worry about the soft-error problem associated with ultrasmall array cell sizes, and we need to have appropriate error correction to deal with soft errors.

On the other hand, within the same design, if we're going to have small arrays where we can't afford that redundancy, encoding, error correction, and so on, we've got to proportionately scale the design with wider transistors and a different set of design rules. So everything has to fit together.

Aitken: I agree that everything has to fit together, but I think that along with the issue of small versus large, we also have to bear in mind the sheer volume. In some applications we have hundreds of small memories, and individually we can say soft errors are not a problem for this memory; repairability is not a problem for this memory—but when overall it adds up to millions of bits, then suddenly both of those things are important. So that means that architecturally we may find that it's more advantageous to add repairability or error correction than it is to relax the design rules and make all 200 of those instances bigger than they used to be.

There's also an interesting point on the hard versus soft IP. In designs involving analog and mixed signal, for instance, it's clear that it has to be hard IP. And then in digital soft IP, some of the soft IP—given the nature of today's design tools and design flows—winds up being "semisoft" IP in that everybody is going to run the same physical tools on it; they're going to get similar placement and similar floorplans. So even though it's ostensibly soft IP, it really has more in common with a memory generator, where we're producing a very similar structure, just with slightly different parameters such as a different cache size.

Eichenberger: Well, the difference of course is that you're mapping to different libraries. And what Sandip [Kundu] mentioned earlier is how do you specify your min and max? And different libraries may have taken different margins there.

Aitken: True, but if the libraries all come from a similar source for a similar process, they ought to be designed to operate under the same conditions and with the same margins, whatever number of process variants you're talking about. But you're right, if you're mapping them across different process generations.

Eichenberger: Let me try to shift the subject of this discussion a bit. Here we are at the test conference, which is essentially about outgoing quality less than yield. (We all like yields much more because there's more money there.) What DFM, or defensive design as we might call it, is doing to us in terms of getting the last few percent of yields is one question—also in getting better quality parts out the door. For one thing, we have via doubling, which we do for yield, because if one via is bad, we still have the other one, but it also makes our delay test a lot more relaxed. It's easier to get good timing there, because we always have redundancy. So is DFM purely a yield issue? Or is it also a quality issue?

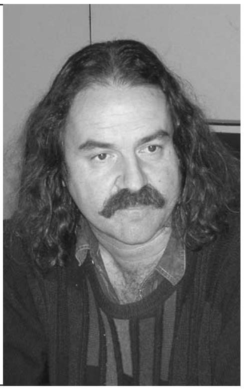
Kundu: Well, clearly, DFM is not just about yield. To me, as the ITC keynote speaker pointed out, yield is the quality at time *t* equal to zero. But we also need to worry about quality at times *t* greater than zero. And we clearly need to have a set of good design rules or design checks in process that make sure we don't suffer from electromigration or hot electron effects, and that we have a good and robust design for the duration of the product life. To me, that's all part of design for manufacturability. Because, ultimately, the product has to work.

Maier: I'd like to comment on yield versus factors such as performance. It's fair for a design team, or a customer, to come to a fab and expect that if the fab published a design rule manual, and the customer or design team has designed in keeping with that manual, they should expect to get process yield to design yield. But this must be a shared responsibility, because it's becoming more difficult to achieve not only yield but high-performance design in any kind of reasonable yield. This is particularly true in pervasive applications where we have the process pushed to the edge to get longer product life cycles. Then we can afford lower yield on higher performance and larger designs—in the high-end server market, for instance. These high-performance designs are falling into the foundry applications with much lower profit margins. There are certain process controls we can put in place, but the reality is that we're moving so fast into the next-generation technologies that the design teams and the customers have to share with us the problem of controlling the edge in order to get their product to market.

Performance is crucial, and we need to get all these variables from the fab into design modeling. I don't know what design teams use in their simulations today for that two-sigma process variation. Are they strictly using things like V_T and temperature? Are the design teams throwing in things like via and line resistance, and capacitance, for example? These parameters can now not only affect performance, but also functional yield. This certainly can determine whether you get 2 gigahertz or 2.2 gigahertz, for example. If you don't reach the performance target, you lose the business, so it might as well have been a supply problem due to functional yield loss.

Aitken: Those are excellent points. When you claim you have two-sigma margins, what does each sigma represent? How do the variations interact statistically? What process models are you basing them on and what is the understanding of what your design is doing with the variation? And you're right, we have to consider what's happening to the transistors and so on, related to where they're going in the layout, what is the resistance, where your extracted capacitors are, is the extraction being done correctly, and so on.

Eichenberger:
So is design for manufacture purely a yield issue? Or is it also a quality issue?



A related issue that we haven't touched on is that, in order for a designer to use those modeling details, they have to go into the EDA tool libraries themselves. Which means that the EDA tools must support the kinds of delays and effects that we're talking about. Tool vendors are starting to recognize that, and we're starting to see a lot more complex modeling that they're willing to support—they're willing to support delays that aren't just lookups but are based on polynomials and so on. To really describe what's going on in the silicon, we have to make sure that the models available to the EDA tools can reflect what's going on in the manufacturing process as well, in order to grant you any number of sigmas.

Maier: Getting back to the need for the database, I think we need comprehensive, empirical knowledge to develop those models for the future. It's not going to happen fast enough. I'm hoping that we can get established with this kind of database, and if nothing else, maybe even do some improved statistical yield and performance modeling in fab. If the fab can't share certain pieces of that data, perhaps we can provide the integrated models ourselves, and then hand those models over to the full fabricators and tell them to include some of this information. I'm not sure how it's going to all roll out yet, but that's the direction we need to head.

Kundu: We can develop a model that explains some of these behaviors, or at least we should be able to explain the physics of it. But as we know, our product design process is based on static timing analysis, which already incorporates a lot of pessimism into the process. Often we're already assured by the pessimism that goes into the design process that it will work. If we try to increase that pessimism by talking about various other physical phenomena, we get push back from the designers because they don't trust these models. Experience shows that chips often run faster than they were designed to do. At that point, the designers aren't willing to add greater degrees of pessimism into the design process, based on their empirical experience. On both the design side and the design tool side, we need to work hard to make sure we don't make the process overly pessimistic at different points.

Aitken: We have to distinguish between pessimism and accuracy. One of the main reasons that we have pessimism in the tools or in any models is to say, "Well we don't understand everything that can go on here, so we'll put in a little bit of margin to account for it." The more we understand what's going on, the more we can reduce some of those margins and allow people to design such that they're working with a more accurate representation of the process. Because, naturally, if designers repeatedly discover that their products run 25% faster than the tool said it would, we know that at some point they'll say, "Okay, we only have to meet 80% of our speed target and this thing will work."

Maier: As an example of what might happen if the models are more accurate (and we've demonstrated it on several occasions in our fab), consider this: If you could, for instance, control the tails of a normal Gaussian distribution of performance on a product, you can squeeze that distribution to be tighter at a higher peak and remove the tail. You can then design it to operate within a certain set of parameters. You don't have time to spin a new design for every technology and you're fighting to beat time to market, you need to leverage your process. This is where the design team can go to the fab and ask, "What can you do in your process to get higher performance and/or yield with my old design." Tightening that distribution with a tighter, more accurate set of models will let you push this large, profitable distribution closer to the edge.

Kundu: Those are good goals, but in practice, it's a moving target. That fact, plus the fact that typically we want to shrink the design, at least optically, a few times means that we have to incorporate certain pessimisms to guard-band the design. Once we've designed it, even if we had a very accurate model for that process, we know it's going to be optically shrunk, and we don't want to invest additional effort into design at the shrink point because we want to get that design for free. So building in pessimism is already inherent in our design process, and we can't do away with guard-banding for the foreseeable future, because it's tied to how the economics of design work.

Maier: These design and manufacturing philosophies must change. To repeat: Our 18-month and 2-year technology and product qualification life cycle has shrunk to 6 months to a year. For the design team to go into the fab with their old design and these new margins, expecting to be able to shrink this two or three times for free is no longer possible. If we expect to be first to market with the fastest part and the highest yield possible, we need more accurate models of higher distribution. There's just no getting away from it.

Kundu: When I'm talking about shrinking, I'm not talking about shrinking to technology generations. I'm talking about using the same technology generation to take that product and shrink it 5% or 10%.

Aitken: That's primarily a custom design versus a commercial, off-the-shelf design issue. I think it's possible in an Intel-type business and much less possible in the foundry and ASIC world.

Walker: For most ASICs, which are single speed bin chips, the real problem in some sense is that design has been done by worst-case analysis. That's why design engineers complain about excessive pessimism in the tools. Well, by definition the worst case hopefully doesn't happen very often. I've known designers who've said, "I'm going to design for the average case, because my experience tells me that the design will achieve an adequate speed bin distribution. Alternatively, if I have statistical data that indicates my probability of being over my spec is a certain amount, I can directly feed that into my cost model in figuring out what the product cost will be. I can ignore that tail of the distribution if it's small enough; I can rely on the test people to throw those chips away."

Aitken: The danger of doing that, of course, is if you design only for the typical case and not for the slow, the fab might just wander into the slow case and stay there for a month, and then you get no yield at all over that month, which could be bad for your product chip.

Walker: That could be bad for our foundry, too. [laughter]

Maier: This is a shared responsibility between design and fab, and you're right, if we push things to the edge with tighter distribution, better and more accurate models, then the fab at that point is expected to control that position of that distribution, and not allow it to shift over the line in terms of serviceability. If we're committed to serviceability, we all then have to make some additional commitment. We're all taking some risks getting close to the edge, but that's where we operate today in technology, speed, performance, size, yield—everything.

Eichenberger: This is nothing new. If you design for portable devices and you have leakage currents varying by a factor of 10^3 to 10^4 over your process window, you have a difference of stand-by time you cannot accommodate: If your phone operates somewhere between, say, an hour and a week, that's no good! [laughter]

Aitken: It boils down to the difference between a spec part of the product—in which case in a phone, for example, that battery should last for more than a second's worth of leakage current—and something that's part of the design flow, part of the manufacturing process that isn't related to the specs. The drift of some parameters is something we must accommodate in design, and the drift of other parameters is something that puts the product right out of spec—we have to reject those parts in test. Which parameters those are depends a lot on what our goal is.

Eichenberger: I think, we talked a lot about transistor parameters: V_T and L_{eff}, speed and leakage. Now that we are more and more dominated by the back end, I think we yet have to learn what the key parameters are that vary in the fab. If we say everything can vary—that's probably a little over the top. Over the years, we will maybe come back with two or three parameters to characterize our metal.

Aitken: Do you have any idea what those might be at this point? Anything in particular?

Maier:
We need a better understanding, from a fab perspective, of the new design issues and how they're affecting the foundry and the ASIC providers.

Eichenberger: Not completely yet.

Maier: This is again a shared responsibility in this complex industry that we're in. We need a better understanding, from a fab perspective, of the new design issues and how they're affecting the foundry and the ASIC providers. It's a shared responsibility to understand those design issues to make sure that we build our new business models appropriately. If these issues are not proactively addressed, it drives more resources to correct and control, and therefore, higher costs and lower profit margins.

We've got to get out of this thought that the high-end foundry processors and SoCs are going to be positioned in a business model as economically affordable as they used to be. ASICs and foundries once had a very low cost. Now these new designs are something in between, somewhere between a cell phone chip and an advanced microprocessor. Our foundry business community needs to reexamine the fab's costs to build these designs with the performance and yield expected by their customers. Pushing higher-quality designs using DFM-aware design tools, integration of process, test, and design tools and databases, and new test and measurement tools and methods—that's my approach to how we get there. Perhaps only then can we begin to provide more complex ICs for the foundry and pervasive market at old foundry prices.

Walker: If we do what we'd like to do rather than just meet these design rules, we'll get this additional information that gets encapsulated into design tools. Ideally, they're synthesis tools so the designer doesn't have to worry about process variability so much. That basically pushes some process knowledge into the design world, rather than having an artificial legal boundary (design rules), if you will, between the design and the fab. That's more like an IDM where we can have some wiggle room on that boundary.

Kundu: The design rules, when they were originally set up, were really rules for design for manufacturability. The reason we're having this discussion today is because we have other variations, notably the transistor channel length variation, transistor threshold voltage variation, and interconnect and dielectric thickness variations that have a much greater impact on design performance today than previously. So the question is, Do we need to think differently or can we continue to work with a set of encapsulations so that only a few designers need to worry about these things, but by and large the design integration and the design can be done with a set of rules?

My position is that we will be able to encapsulate these things. Perhaps we'll need to propagate some of that knowledge to the library cell designers, and to the circuit designers. They'll need to run a few tests with the fab to get the learning in place, but overall we'll be able to live with design encapsulation without having to educate all designers about basic principles and how to improve yield and manufacturability.

Eichenberger: I agree with Sandip [Kundu]—that is, we should live with the encapsulation, but not just because a design rule says there should be 200 nanometers of space between metals, so we should always space the metal at 200 nanometers. It's about defensive design. If it becomes encapsulated in the tools in a decent manner, then we can achieve that.

And the second part is enabling rapid yield learning—what we called the database, or data mining, before—so we can quickly identify critical items. Is it wire spreading? Is it density that counts most, or is it via doubling? Or is it something completely different? The learning cycle is something we need to think about.

Aitken: Given that the complexity and the cost of building a chip at 90 nanometers make it a $10-million-plus project, the economic incentive is there for all pieces of the puzzle to work together—for the foundries, the IP vendors, and the designers to all work together. And the disaggregation of the industry allows these different specialists to collect enough data and have enough knowledge from working with enough different people that they can provide some of the encapsulation that once was done strictly in design rules. And they can now do it in slightly different ways but accomplish the same end.

Thank you, everyone.

About the participants

Rob Aitken, our moderator, is senior architect of product technology at Artisan Components in Sunnyvale, California.

Stefan Eichenberger is responsible for process-related test improvement at Philips Semiconductors BV in Nijmegen, the Netherlands.

Gary Maier is the 300 mm accelerated-yield-learning project manager at IBM Microelectronics in East Fishkill, New York.

Sandip Kundu is a principal engineer in the Design Technology Group at Intel in Austin, Texas; he looks after circuit and design marginality issues.

Hank Walker is an associate professor of computer science at Texas A&M University in College Station.