

Published in issue of Chip Design Magazine

## What makes chips different?

### IBM, Samsung team up to differentiate chips with embedded software modules, but can it work this time?

By Ed Sperling, Contributing Editor

Re-using embedded software has been the subject of exuberant marketing for nearly two decades—and so far it has amounted to little more than that. But in recent months, quietly and far from the blaring hype, real-world testing is under way to use software as a cost-effective means to embedding functionality in a wide range of devices.

The experiment is a joint effort between Samsung and IBM's Haifa Research Lab in Israel. If all goes as planned, IBM plans to begin commercializing what is essentially a middleware framework that allows embedded software to plug in and differentiate semiconductors. That means in many cases it will no longer be necessary to build new chips for every new device.

The project is based on what IBM calls the Consumer Electronics Development Environment, known inside IBM as COMPETENCE. At the heart of this environment are architecture description languages, a class of descriptive languages that includes Darwin, C2 and Acme, to provide a high-level of abstraction for writing software. None actually exists for the consumer electronics world, a fact that IBM wants to change and to capitalize on.

Normally, embedded software is written for specially developed processors with very specific functionality. With COMPETENCE, a single chip design theoretically will be sufficient to replace many designs, because the embedded software modules will add different functionality.

The advantages are immediately obvious to anyone who has designed chips for such areas as cell phones or MP3 players. Each chip can cost many millions of dollars to develop, with those costs escalating at each new process node. At 65nm and 45nm, non-recurring engineering costs render it impossible to earn a profit without an enormous sales volume. Adding more functions only increases that cost, and it can slow the time it takes to bring products to market, in large part because of the difficulty of verifying the chip.

"The world of modeling is well known in hardware because it's so difficult to work with," said Alan Hartman, research scientist manager at IBM's Haifa Research Lab. "In software, this is new. But when you look at the cost of a luxury car, a large part of that cost is the software."

### History repeats...sort of

In the realm of software, component-based systems engineering is a relatively new field. Some of the groundbreaking research was performed by Philips Electronics a decade ago using various component models, and in the mid-1990s many software companies including IBM had plans for software objects—portions of applications rather than applications themselves—that could be bundled together like Legos and plugged into a framework.

The problem was that at the application level, these software objects never played together like a single-vendor plug-and-play system. IBM, however, was large enough and had enough of an investment in middleware that it could shift the development to where there was a strong business need. That need grew as the cost of developing ASICs and ASSPs escalated into the tens of millions of dollars.

"If you look at multifunction printers, some have faxes, scanners, and different speeds, and each is implemented by a separate component," said Hartman. "We can design a new printer with new components so you only have to write a very small amount of code."

The approach plays off of the holistic design that IBM has been pitching for the past several years. "What this allows you to do is manage a product line as a whole," said Julia Rubin, an IBM research scientist for industry solutions in the Haifa Lab. "You determine what features you want, and only the pieces you want go to market."

That works in principal, at least. But the real trick in this scenario is what is relegated to software. Tony Massimini, chief of technology at Semico Research, said certain functions are fine in software, particularly with a powerful PowerPC processor running those functions. But he said the model runs into trouble when it comes to video because the response time has to be so high that it needs to be

hard-wired into the processor.

"The PowerPC has a lot of performance, and if it's not a portable application you may get away with it in software," Massimini said. "If you look at a car, the response time versus streaming video is easy to accomplish in software. With a portable application, you can overload the processor."

### **What's in the package?**

What IBM and Samsung are developing goes beyond just writing embedded software, though. Hartman said the companies also are providing an editing environment, the ability to validate the software with rules and to configure a particular product.

"Then, at the touch of a wand, you get general building scripts," he said. "Companies in consumer electronics have already moved to a component-based architecture, so this is a natural fit. What we're also doing is taking legacy code and componentizing it."

When exactly that will become productized remains to be seen, however. At this point, the legacy code work is relegated to some future date. IBM will offer that type of work as a service through its consulting group. However, IBM researchers contend that fixing the bugs in software is a relatively simple and inexpensive process, versus trying to fix them in hardware. In part, that is because the programming languages for software are relatively flexible. C++, for example, is highly modular, compared with trying to scour through massive amounts of verification data to pinpoint a bug and then fix it in hardware.

### **Language issues**

One of the reasons that ADLs remain relatively unknown in design is that there is no single standard. As such, there is no agreement on what needs to be included in the overall description.

IBM is looking to establish such as standard for the consumer electronics industry, along with a modeling framework that third-party components can plug into to provide consistent behavior. IBM said it also plans to extend that framework to other industries in the future.

At least part of that will be based on a standard modeling framework, called the Unified Modeling Language. IBM's COMPETENCE relies on the Rational toolset (see diagram), which allows designers to develop components based on established models. But how quickly all of this technology comes to fruition, and just how widely it is deployed, remains to be seen. Building better software, particularly for the embedded world, is not a new problem.