

## HW and SW in Embedded System Design: Loveboat, Shipwreck, or Ships Passing in the Night

**Moderator: Kurt Keutzer, University of California, Berkeley**

**Organizers: Raul Camposano, Synopsys, Inc., Kurt Keutzer, University of California, Berkeley**

### **Abstract**

The merging of hardware and software on a single integrated circuit is causing many to rethink their approach to embedded system design, and some are forecasting significant changes in the dynamics of the associated electronic-design automation (EDA) and embedded software industries as well.

For some, the ocean of silicon ahead is sure to host a *loveboat* for fully integrated hardware/software systems. In this scenario a unifying system-design-environment provides implementation-independent modeling that stands above the particulars of hardware and software implementation issues. At the implementation level, embedded software design tools and electronic-design automation tools work seamlessly together providing a variety of architectural targets for the functionality of higher-level descriptions.

Others see a *shipwreck* on the horizon, as hardheaded hardware designers and softheaded software developers clash. In this scenario the system-on-a-chip becomes a battleground in which the two respective communities attempt to dominate both the solution space, and the system-design budgets of their common customer.

There's an old adage that the most boring scenario is the most likely. Following this adage some predict that the future holds more "no show" than "showdown." The holders of this view note that as severe power constraints cause hardware-designers to eschew software solutions, and as the world of ubiquitous computing significantly broadens the market for embedded system software, then the hardware and software communities will be *simply ships passing in the night*.

### **Panelist Statements**

**Jerry Fiddler**

**Founder and Chairman, Wind River Systems**

The central problem being faced by builders of embedded systems is complexity. The hardware continues to become more powerful and easier to design (thanks in no small part to the EDA industry). Communications bandwidth becomes greater and more easily accessed. These trends

drive up customer expectations of what can, and therefore should be, accomplished in a product. Unfortunately, the problem of the resulting system complexity is square in the way. Suddenly, embedded system designers face problems of network management, changing and heterogeneous architectures, and complex user-interface design, in addition to the old issues of hardware and software design.

To date, the EDA and embedded software industries have happily coexisted (in seemingly orthogonal universes) because the problems have been big enough and diverse enough to go around. While it seemed at one point that these two universes might collide, the growth of complexity of embedded systems is forcing embedded systems software providers to look far beyond issues of silicon implementation, and onto new frontiers such as ubiquitous computing. Thus for the next few years it appears these universes will continue to evolve on their own, driven by the aforementioned issues of system complexity. While EDA and embedded software tool provider will be cooperating in areas such as system-on-a-chip, custom CPU design, co-design, etc., the silicon forces which are enabling system on a chip are also creating a number of other system complexity challenges that embedded system software providers must also answer.

**Raul Camposano**

**Senior Vice-President and CTO, Synopsys, Inc.**

The silicon (chips) content as a percentage of the total electronic systems revenue is growing. So is the embedded SW content. The EDA industry is growing at roughly 20%/year, with a slightly higher number reported for embedded SW tools (RTOS, IDE, DSP Tools). The EDA industry is 5 to 6 times larger than the embedded SW tools industry. It is widely believed that EDA and embedded SW design need to get much closer for HW/SW co-design. From a technical point of view, the EDA industry typically trades tool runtime for design quality of results (speed, size, power consumption) and extensive verification (simulation, static timing analysis, formal verification, etc.). SW tools put more emphasis on short tool runtimes. The technically hard challenges ahead in

HW/SW co-design are complete system verification (the bottleneck being HW simulation speed) and optimization of the complete system (quality of results of combined HW and SW). These have been core competencies of EDA. In conclusion, the EDA industry seems well positioned both technically and marketwise to move into embedded SW, while the embedded SW industry is likely to look for more aggressive growth closer to the applications.

**Alberto Sangiovanni-Vincentelli**

**Professor, University of California, Berkeley**

The important matter in embedded system design is the co-development of the functionality of the system (what the embedded system does) together with its architecture, *i.e.* the set of components that implements the functionality (how the system does what it is supposed to do). The result is influenced by cost, performance, power, and flexibility trade-offs. If some (or all) of the components are hardware components, the physical implementation of the hardware component reflects directly the part of the functionality that is mapped onto it. If some (or all) of the components are software programmable, the physical implementation of the block is the support for a customization carried out via the selection of the sequence of instructions to be executed on the processor. Each programmable component is characterized by a different "architecture" itself and this yields to different types of software implementations (the world of embedded system is heterogeneous both in the hardware and the software). Hence, software and hardware are two faces of the same coin: both originate

from a higher view of the design that is mapped onto heterogeneous components. Functionality and architecture are loving parents of many hardware and software siblings (I'll leave the attendance decide on the gender of each of the parents (functionality and architecture) and of the many hardware and software siblings.....)

**Jim Lansford**

**Wireless Systems Architect, Intel**

The partitioning of the architecture in future wireless systems is going to be dynamic; hardware implementations will be low power and low cost but inflexible. Certainly there are issues with multiple protocols as well, which would be best met with a software approach, especially when the protocol must be switched on the fly. These battles over optimal architectures rage every day within many companies, and the arguments five years from now will probably be significantly different from those today. My view is that there is no one clear winner; whether an implementation is software or hardware based depends on the application. In view of the panel theme, I agree that this is the "love boat"; a design manager should view HW and SW as complementary ways to solve a problem. A laptop computer might be able to absorb more cost to achieve interoperability across protocols using a SW based MAC, but a toy developer would give up flexibility to drive costs and power consumption down. So, the kind of ship you build should be based on you requirements...on the love boat, you choose!