

Hardware Virtualization for Pre-Silicon Software Development in Automotive Electronics

Frank Schirrmeister and Filip Thoen
Synopsys, Inc.

Copyright © 2009 SAE International

ABSTRACT

Recent surveys show that the number one issues in car electronics are cost reduction and better collaboration across the complex automotive design chain of IP providers, semiconductor providers, subsystem suppliers and system integrators (see [1]). With millions of lines of software code powering modern cars and the number of processors today already exceeding 80 (and further growing), software development becomes the critical component determining success of automotive development projects.

Virtualization of hardware to enable software development – a technique already quite adopted in other application domains like wireless communication - is entering mainstream in automotive electronics as well. However, different areas of the car electronics require different modeling accuracy. While driving and vehicle dynamic functions often require full timing accuracy, other areas like multimedia, telematics and some of the body functions, like air conditioning and theft control, can be modeled in a less accurate fashion.

This paper analyzes the modeling accuracy requirements for different areas of car electronics. It will introduce SystemC™ standards based modeling styles to meet these accuracy requirements with loosely timed (LT), approximately time (AT) and cycle accurate (CA) modeling styles. In addition this paper will discuss methodologies and tools to efficiently assemble, debug and verify virtual platforms to enable

pre-silicon software development. Based on case studies, this paper will illustrate how the use of virtualized hardware for software development and verification early in the design flow, not only improves software development productivity, but also increases software quality, serves as a front-end for hardware and software verification, leads to significant cost reduction and can become an essential mechanism to enable collaboration within the automotive design chain.

INTRODUCTION

Technically, the electronics industry is facing growing electronics content and complexity. Multi-node and networked engine control unit (ECU) architectures are commonplace and require complex intra-ECU behavior and inter-ECU interaction. Today already 40% of vehicle cost is attributed to electronics and software, 50-70% of ECU development cost is due to software. As a result of this dramatic increase in software content and complexity, meeting quality and reliability today requires analysis and diagnosis of complex hardware/software problems, and a change in development methodology.

In addition, the automotive industry is challenged by an increasing need to reduce design cycle times and cost. A recent Dupont study [1] from 2008 resulted in 32% of respondents answering the question what the number one challenge facing the automotive design and engineering community would be with "Cost Reduction". "Fuel Economy" with 28% and "Emissions/Clean Air Regulations" with 26% were the

number two and three most cited challenges. In contrast, “R&D and Technology Development” was only mentioned by 14% of respondents. When asked what improvement would be needed to strengthen the industry, overwhelming 53% of users responded with “Increase Collaboration across the Value / Design Chain”.

When combining these results with the increasing electronics content in vehicles and the dominance of processor-based electronics within that content, software becomes the central issue to which the challenges cost reduction and collaboration must relate.

THE AUTOMOTIVE DESIGN CHAIN

Figure 1 shows a simplified diagram of the automotive design chain. It is comprised of hardware intellectual property (IP) providers, semiconductor vendors, subsystem suppliers and system integrators. All of them interact with software suppliers of IP and tools.

All of the design chain partners have specific technical and business concerns when it comes to collaboration and interaction amongst them.

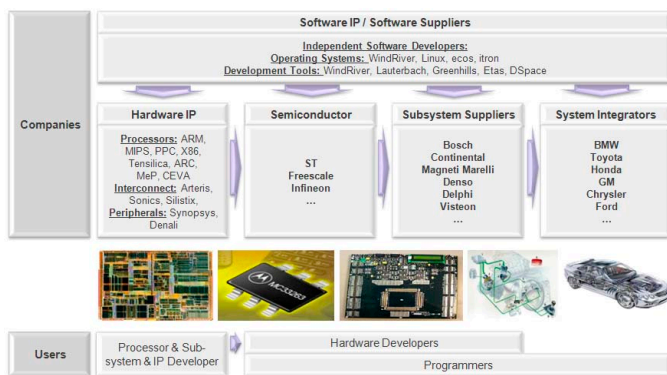


Figure 1: Automotive Design Chain

Hardware IP providers are focused on architecture analysis and feature definition of their components. The main objective is to get “designed in” by semiconductor houses. In order to achieve that, hardware IP providers seek early feedback from their customers and need to be able to perform early architecture analysis for hardware/software systems. While traditionally hardware IP providers have been focused on hardware only, today they need to ensure early that their components – like processors – interact appropriately with software executing on them, or even provide essential software support.

Semiconductor suppliers have similar concerns in terms of being able to do architecture analysis of their components and to improve the likeliness of being

“designed in” by their customers, the subsystem suppliers. Even more so than hardware IP providers they are today expected to take full responsibility for device drivers and operating system (OS) porting to their chips. As a result, they are increasingly concerned about delivering software as early as possible. As a result, they are seeking new ways to advance software development to earlier project stages and to increase software development productivity.

Sub-system suppliers are mostly concerned about quality, reliability and cost of their sub-systems. Given the large percentage software development contributes to ECU cost, early start of software development and its productivity are of special concern. Before committing to semiconductor providers, they would like to validate the architecture of ECUs and its semiconductor content. Overall they are concerned about reducing the product development cycle time, improving quality by testing as much as possible and using more systematic test methodologies, improving software development productivity and reducing cost by shortening the bench time and reducing the need for hardware prototypes.

Finally, system integrators are concerned about software development productivity, quality, reliability and cost. They need to do software integration often across multiple, communicating ECUs and require as early access as possible to software development environments representing what the supply chain will provide to them.

ACCURACY REQUIREMENTS FOR DIFFERENT TYPES OF AUTOMOTIVE ELECTRONICS

Virtual platforms have been used for quite a while to allow early development of software on a virtualized version of the target hardware.

A virtual platform is a fully functional software representation of a hardware design that encompasses a single- or multi-core SoC, peripheral devices, I/O and even the user interface. The virtual platform runs on a general-purpose PC or workstation and is detailed enough to execute unmodified production code, including drivers, the OS and applications, and this all at reasonable simulation speed. Users have articulated the need for virtual platforms to not be slower than 1 tenth of real time to be effective for embedded software development. The achievable simulation speed depends on the level of model abstraction, which also determines the platform’s accuracy.

Not all areas of automotive electronics require the same level of accuracy. Figure 2 (from [2]) outlines different types of automotive electronics, from information systems like navigation and internet access through body electronics like air conditioning, door and light modules requiring to be fail safe, to the essential driving and vehicle dynamic functions requiring to be fault functional.

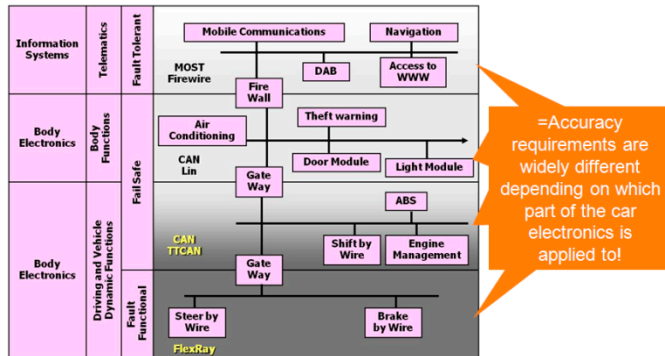


Figure 2: Different types of Automotive Electronics

When developing software on virtualized models of the target hardware, the different areas of automotive electronics require different levels of accuracy. For information systems typically pure functional models are sufficient. For more timing critical elements, the virtual models of the hardware need to reflect increased timing detail. For the essential functions like drive by wire etc. only part of the software functionality can be developed using not fully timed virtual models of the hardware. For these cases users require in most cases fully cycle accurate models of the hardware and ideally formal proof of correctness for the hardware/software functions.

With increasing processor and SoC complexity, both driven by increased silicon integration, it is becoming more and more difficult for cycle accurate models to be faster and easier to develop than the actual chip implementations in Register Transfer Level (RTL) language. Consequently, users more and more demand to use the actual physical prototype or to utilize models automatically derived from the RTL when using virtualization.

Traditionally virtual platforms – virtualized versions of the target hardware – have been available at different abstraction levels.

HISTORY OF VIRTUAL PLATFORMS

According to recent market research by IBS International, in 2011 (or effectively for chip designs starting in the next 12 to 18 months) more than 60 percent of the chip development cost will be

consumed by software. In addition, while today it is well known to chip developers that a delay in software development for their chip may delay volume production, 10 years ago that was not the case. Still, at that time, several companies formed to productize the virtualization of embedded hardware for pre-silicon software development.

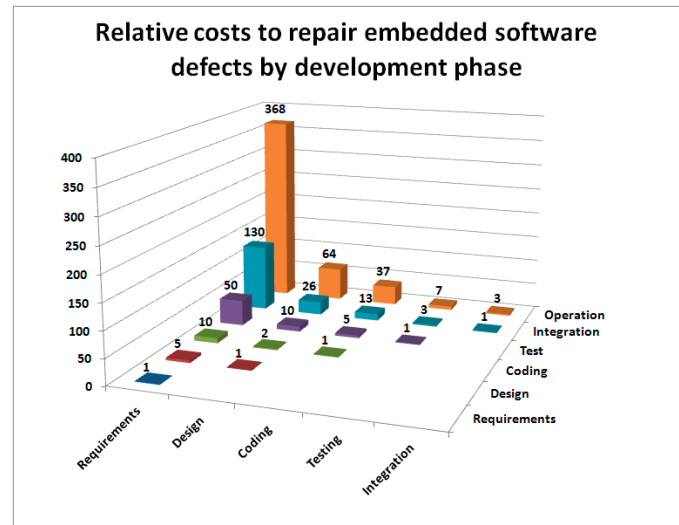


Figure 3: Relative costs to repair embedded software defects by development phase, from [3]

From the start, all of these virtual platform vendors addressed two key SoC trends – the escalating cost of software, and the fact that to avoid finding bugs late, the software development would have to start earlier. Figure 3 illustrates the relative costs to repair embedded software defects by development phase. The analysis shows that bugs that originate in the requirements phase are five times as expensive to repair during design, 10 times as expensive during coding and 368 times as expensive when a device is delivered and in operation. Pre-silicon embedded software development advances the integration phase, which was traditionally done after silicon availability, to an earlier project phase, thus allowing bugs to be found earlier, directly reducing the overall cost and project risk.

In the absence of a pre-existing modeling standard, all early providers of virtual platform technology developed proprietary modeling solutions. AXYS Design Automation was formed in 1997 as a spin-off of the Aachen University of Technology, and promoted the language for instruction set architecture (LISA) to describe processor architectures. Also in 1997, VaST Systems Technology Corporation was founded in Sydney, Australia and started to offer processor modeling based on binary code translation. A year later, Virtutech was founded as a spin-off from the Swedish Institute of Computer Science (SICS) in Stockholm, Sweden, followed by Virtio Corp. in 1999