



Virtual Platforms for Software Development

Adapting to the changing face of software development

November 2005

Virtual Platforms for Software Development

Adapting to the changing face of software development

November 2005

Executive Overview

The explosion of software content used by mobile device manufacturers and silicon vendors to differentiate their products is driving these companies to re-engineer their software development methodologies. Today's mobile devices have become full-fledged clients that contain a rich array of applications and their supporting infrastructure software including operating systems, security, management, messaging, and browsers. Support for multimedia content and software in excess of several millions of lines of code is commonplace. According to Venture Development Corporation (VDC), mobile development projects are scheduled to take an average of 14 weeks and, in reality, end up taking 20% longer than that. These time overrides and time-to-market delays have significant cost and profitability implications. New development processes and tools must be deployed to address software development.

Virtual Platforms for software development is an emerging market that serves to accelerate the development of software in embedded and mobile devices and rapidly replace previous tools such as prototype boards. Virtual platforms are fast and scalable models of the system hardware, including the device hardware and the environment it evolves in. Virtual platforms provide the combined execution speed, early availability, controllability, observability and determinism, with pre- and post-silicon usability, that none of the traditional software development methods provide.

Virtual platforms have a direct impact on software and device development by enabling early software development and validation of the entire device software stack, increasing developer and development team productivity and lowering capital expenses. The result of such an approach is additional time and budget is available and can be dedicated to new market opportunities, increased differentiation or improved quality.

This paper describes the benefits software developers should expect from virtual platforms as well as considerations that companies interested in this approach should take into account when creating and/or outsourcing the creation of a virtual platform.

According to a survey of embedded system designers by Embedded Market Forecasters, 65% of respondents cited "limited visibility into the complete system" as the primary cause of failure to meet specification or schedule requirements, followed by limited ability to trace (54%), and limited ability to control execution (42%).

What are Virtual Platforms for Software Development?

Software represents a significant avenue for device manufacturers to create high-value, differentiated features in their mobile device. A mobile device software stack includes software such as the operating systems (hardware abstraction layers, networking protocols, OS services, graphical and text utilities), application frameworks (Browser, Java Virtual Machine, Messaging), middleware utilities (security, management, provisioning) and applications. Software content in mobile devices is increasing at a fast pace and a full-fledged mobile device can now reach several millions of lines of code.

Although software content is increasing, the approaches for software development have not evolved at the same pace. Traditional software development approaches, such as native host development and hardware based-development, have not provided the required solution—one being non-representative of the hardware being developed to support the software and the other being available too late in the design cycle. As a result, software developers have turned towards simulation-based approaches such as Instruction Set Simulator or RTL-based simulation. These approaches, although initially valuable, have not been able to scale with the software explosion nor with the requirements of

Virtual platforms for software development are fast and scalable models of the system hardware including the device hardware and the environment it evolves in. They provide the combined execution speed, early availability, controllability, observability and determinism, with pre- and post-silicon usability, that none of the traditional software development methods provide.

Virtual platforms are not a new concept for hardware design and low-level hardware-dependent software. However, recent technology advances are demonstrating that a complete systems' functional simulation can be created with virtual platforms. Software development organizations that deploy virtual platforms into their development process can achieve significant reductions in cycle-time.

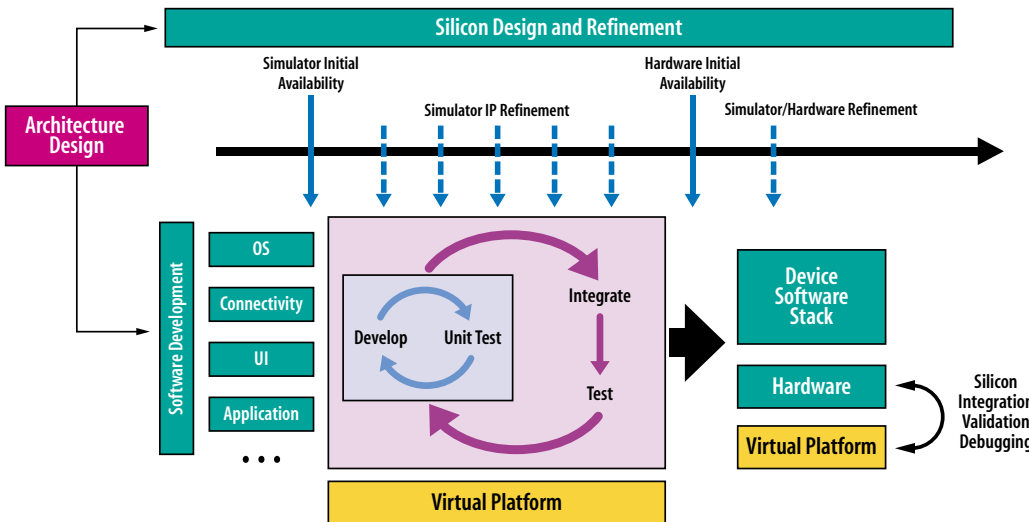
Benefits of Virtual Platforms for Software Development

Virtual platforms for software development provide four major benefits

A virtual platform enables earlier software development

In order to be tested, system software requires the hardware prototypes to be available. Bringing up untested software is a time-consuming process. In addition, targeted software testing can only begin after bring up. Virtual platforms enable software development to start in advance of silicon, RTL simulation or FPGA prototype availability,

Virtual Platforms are scalable to the software roadmap. Initial software development may start on a simple Instruction set simulator and some peripherals, enabling the boot of an operating system. As software development milestones are progressing, so does the content of the virtual platform. Ultimately, a virtual platform can provide a complete virtual test bench environment including a virtual mobile device, a virtual base station and software-based development tools providing the functionality of logic analyzers, debuggers, etc.



large, distributed software development teams. They have been either incomplete from a hardware perspective (ISS) or too slow (RTL based simulation). A new approach is needed and is emerging in the form of virtual platforms for software development.

A virtual platform increases a developer's productivity

According to a survey of embedded system designers by Embedded Market Forecasters, 65% of respondents cited “limited visibility into the complete system” as the primary cause of failure to meet specification or schedule requirements, followed by limited ability to trace (54%), and limited ability to control execution (42%). Virtual platforms can significantly improve the productivity of a software developer by providing more controllability, observability and determinism.

- A virtual platform provides more controllability: Physical hardware cannot be stopped at once. When a break point is set, services will continue executing. With a virtual platform, the entire system can be stopped at once. Virtual platforms provide controllability of their initial state whereas physical hardware needs to be rebooted.
- A virtual platform provides more observability: At any time, a user can get information regarding any part of the system (core, buses, peripherals or environment models). The information can be analyzed and acted upon in any way by the end user. For example, debugging a software DMA problem is difficult with hardware while a virtual platform can provide full visibility.
- A virtual platform provides determinism: Being able to reproduce intermittent problems or complex timing-related problems on physical hardware is difficult. A virtual platform for software development provides determinism which is the repeatability of a test case where the user will get the exact same behavior over and over again.
- A virtual platform for software development is fast: virtual platform technologies enable an operating system such as Linux to boot in a matter of seconds rather than minutes, thus enabling the software developer to test and access functionality of the device software stack quickly.

A virtual platform:

- Enables earlier software development
- Increases a developer's productivity
- Increases development teams productivity
- Lowers capital expenses

A virtual platform increases development teams productivity

In an analysis of software development challenges for mobile software stack, VDC stated “one of the most significant factors impacting mobile development complexity and schedule performance is that solutions are frequently developed with incomplete specifications. In addition, specifications frequently get changed during the development cycles”. Virtual platforms provide an efficient communication vehicle between development teams.

- Virtual platforms optimize hardware/software group interaction. Virtual platforms can be considered within the context of software development only. A software-only solution provides cycle time improvement for software development but does not optimize it. Virtual platforms do not take into consideration the system (hardware and software) development cycle-time. Models developed within the context of an Electronic System Level (ESL) methodology are scalable for multiple development activities—including architecture design, hardware implementation and software implementation—resulting in improved efficiency across the organization. This attribute also allows hardware/software co-optimization and co-verification at a juncture in the design cycle where hardware modification is easiest.
- A virtual platform is more accessible than hardware. It can be easily shared across multiple remote sites.
- A virtual platform's determinism, controllability and visibility make team communication easier. Software developers can reproduce problems in a similar manner even if they are located in a different site. They have access to the same user interface and tools for controllability and visibility.

A virtual platform lowers capital expenses

Multiple and complex hardware equipment is used in the development of devices such as mobile phones. They include hardware prototype boards and connectivity, logic analyzer and environment test set up. Capital costs can be controlled by reducing the dependency on hardware prototype boards for software development or providing visibility enabling easier understanding of problems previously analyzed with a logic analyzer.

A virtual platform simplifies the development environment. It brings onto the developers' desktop an entire physical hardware set up which includes a virtual mobile device, a virtual base station, virtual logic analyzers and software development tools.

Example of use and early adopter results

Mobile software development can take multiple forms depending on the type of software being developed. Examples of virtual platform usages include

- Operating system development: Changing the operating systems port due to processor, MMU or cache changes or debugging new device drivers

- **User interface development:** bringing up a browser platform, validating that a phone can get into service (interacting with a base station model)
- **Application development:** validating applications where a web session can be set up
- **Automated testing:** if enabled, a virtual platform can be integrated within an automated test suite more easily running regression tests in a batch mode.
- **System test:** validating the behavior of the device within its environment.

Early adopters of virtual platforms are experiencing significant gains in their development cycles. One such user indicated speed improvement that resulted in simulation time of minutes rather than several hours while keeping the accuracy of the results acceptable over cycle-accurate models. Savings are also measurable in terms of dollars spent on hardware prototypes, tools and accessories as well as engineering staff effort. For a 3 to 4 month development of a handset savings could average several hundreds of thousand of dollars and more than 1500 staff days.

Deploying Virtual Platforms

Virtual platforms can provide significant value for software and device development. However, as a software developer for a given device, it is important to carefully consider how a virtual platform is created (either internally or by a third party company). The creation of a virtual platform has a direct impact on the capability and flexibility that will be provided to its users. Thus, directly affecting when the developer can access the virtual platform, what productivity enhancement he or his team will be able to achieve and, ultimately, what time and budget savings his company will be able to gain.

The following section provides the virtual platform end user and the virtual platform creator an overview of considerations that companies interested in this approach should know when creating and/or outsourcing the creation of a virtual platform.

Things to Consider When Creating Virtual Platforms

Device software developers can significantly benefit from using virtual platforms for software development; however they need to consider the creation of the virtual platform itself. Who should build it? What technology/methodologies should be considered in the creation in order to ensure they will reach the promised benefits?

Software development teams do not necessarily need to know the details of the virtual platform's construction, but it must be aware of some of the factors that contribute to a model's success. These factors have a direct link with the benefits that will be derived from using the virtual platform.

Integration with Electronic System Level (ESL) design methodologies

ESL design involves the tools and methods for capturing the initial product specification. It provides the modeling environment to capture, visualize and utilize the product within the environment it is intended to operate. Within ESL, platform architects and hardware verification teams are creating models. These models should be developed in a way that is reusable for the purpose of creating a virtual platform for software development. A single model supporting multiple uses should be a company objective.

The derived benefits for a virtual platform user are that the virtual platform in use matches the hardware under development and that the development of a virtual platform can be optimized, thus accelerating its availability.

Modeling languages and standards

While the created virtual platform represents a company's intellectual property, the infrastructure used to model should be as open as possible. Standard model formats and languages should be used to ensure compatibility and interoperability with third-party intellectual property and commercial design tools that are widely used—and re-used—in system design. The use of a proprietary format and language can result in significant re-modeling and tool interface costs, as well as incurring the maintenance costs usually associated with proprietary tools and intellectual property. It can also render impotent an otherwise highly-productive design re-use. The industry language of choice for system-level modeling is SystemC. SystemC provides all modeling constructs for very abstract functional modeling down to modeling necessary details at the RT-Level. A supplier's support for SystemC should also ensure that the languages support exposing the relevant model information (for example, registers).

Given today's large number of SystemC users, using the standard SystemC modeling language will ensure that the virtual platform can be easily updated. It could even be considered, given that SystemC is based on C++—a well known language in the software community—that the software developers could make some modifications to the virtual platform or more easily help with potential implementation errors in the virtual platform.

Simulation technologies

The simulation technology providing the execution infrastructure of a virtual platform must be capable of operating at high speeds while achieving the required accuracy. Moreover, its performance should be evaluated at multiple levels including the component level (ISS) and on a system-wide basis. Using the same system simulator as the one used by the hardware design team significantly eases comparability of results. The ability to simulate at different levels of accuracy is an important consideration. There are two critical components of simulation technology required to deliver a useful virtual platform. One is the availability of ultra-fast, 100+ Mega Instructions Per Second (MIPS) processor models, as the application code will run for most of the time entirely on the processor. Second, the simulation of the bus, memory and peripheral interaction with the processor needs to utilize optimization techniques, which are enabled by the modeling methodology, to minimize the number of time events being simulated.

Simulation technologies have a direct impact on the productivity of the software developer and the development team.

Packaging and distribution infrastructure

The environment used to develop a virtual platform contains capabilities that are not relevant to its end-user, such as a graphical specification environment or the SystemC development tools. As a result, the creator of a virtual platform must have the ability to package the capabilities provided to the software developer in an easy-to-use environment. The creator also needs the ability to iterate and release updates quickly.

Automatically packaging and managing a pool of virtual platforms is important to the productivity of distributed software development teams. It is also important when development involves third-party companies to protect a company’s intellectual property and provide a common development environment.

Enabling controllability and observability

Virtual platforms developed with the right modeling and simulation technologies will provide more observability and controllability for the end user. The additional features provided to the user are not enabled through traditional software development tools. As a result, the creation and packaging of a virtual platform needs to include a set of tools complementing, and not replacing, existing software development tools. These tools need to provide an easy-

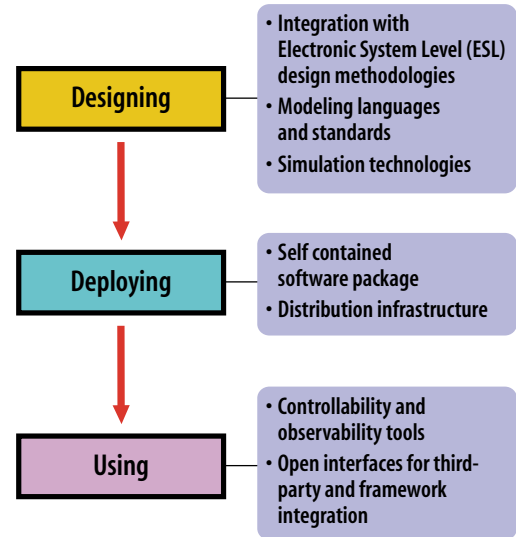
to-use and intuitive interface for the software developer to control and observe the behavior of the virtual platform and its associated software stack. Examples of such capabilities include setting breakpoints and watch points platform wide, force register value, and customized views.

Controllability and observability tools enable higher productivity for virtual platform users while lowering capital costs by reducing the dependency of tools required once the physical hardware is available.

Open interfaces

Additional differentiated tools enabling controllability and observability need to be available. These tools should be complementary to existing software development tools which means that APIs must be available in order to integrate the virtual platform within the software development environment. Examples of software development tools to be integrated include integrated development environments, debuggers, automated test tools, human machine interface tools, and other simulators.

The adoption and deployment of virtual platforms is facilitated with open interfaces enabling the integration



From Design to Use: Virtual Platform Technologies

of commonly-used software development tools.

Who will create a virtual platform?

There is no single answer to this question as it depends on the company focus and structure. However, there are some starting points to consider when evaluating the deployment of a virtual platform for software development:

Within a semiconductor company, platform/system architects and hardware developers are modeling chipsets using ESL methodologies; these teams could be generating the virtual platforms as well as generating virtual platform components to be used by mobile phone manufacturers.

Within a mobile device manufacturer, a modeling or hardware prototype development team integrating models provided by semiconductor and IP companies could be who generates the virtual platform

Companies with modeling and ESL design methodology expertise can also be contracted to provide virtual platforms based on a specified design. Such companies will often provide value around the virtual platform infrastructure technology (modeling, simulation, ESL expertise, tools) which is not the core expertise of a mobile device manufacturer or a semiconductor company.

Modeling teams creating the virtual platform need close coordination with the software development team to ensure the alignment of software development milestones with the virtual platform availability as well as the hardware development teams to gain insight into hardware functionality.

When considering deploying virtual platforms, simply envisioning the use of the virtual platform will provide benefits but it will not optimize the return on investment companies can obtain in using such an approach. Integration with design methodologies, modeling and simulation technologies, packaging and distribution as well as the openness of the virtual platform must also be considered.

Imagine! ... It's possible.

Virtual platforms for software development are a novel approach that address the challenges faced by mobile device software developers currently using traditional software development approaches. Virtual platforms enable earlier software development, increased productivity and lower capital costs. As a result, companies have additional time and budget that can be allocated to earlier product availability, higher quality, additional differentiated functionality or new opportunities. Early adopters of virtual platforms for software development are experiencing these benefits—now.

When creating virtual platforms for software development, considering a software-only point solution is not sufficient. Consideration must be given to the creation of the virtual platforms in terms of methodology, technology and packaging/distribution framework provided in order to optimize the return on investment.

CoWare is the leader in ESL design. Today, CoWare technology and services are used by the top 20 semiconductor manufacturers and by 14 of the top 20 electronics companies. CoWare expertise covers ESL methodologies, modeling and simulation technologies, and software development tools.

Contact CoWare today if you would like to explore the benefits of creating and/or deploying virtual platforms for your next mobile device software development project.

Author: Marc Serughetti

Marc Serughetti drives the ESL software solution strategy for CoWare, Inc. For more than 10 years he has established solution strategies for device software companies such as Wind River Systems, Inc. and Integrated Systems, Inc. He can be reached at marc@coware.com.



CoWare, Inc.
Corporate Headquarters
1732 N. First Street
San Jose, CA 95112

Main: 408-436-4720
Fax: 408-436-4740
www.CoWare.com