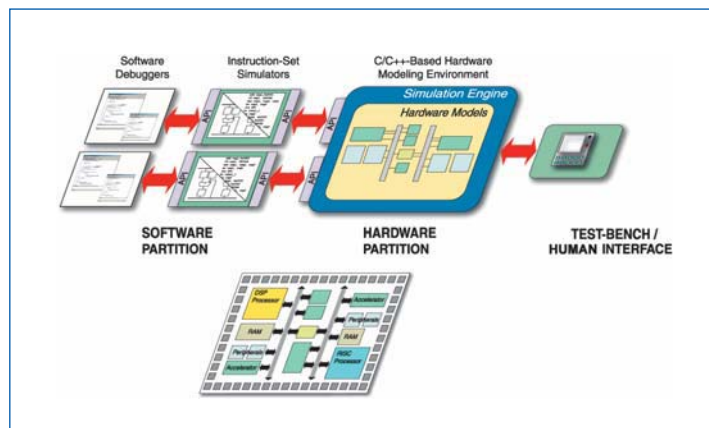


Improving software development productivity with virtual prototyping

by Filip Thoen, Virtio

VIRTUAL PROTOTYPING IS AN INNOVATIVE WAY OF DEVELOPING EMBEDDED SOFTWARE, COMPATIBLE WITH THE TOOLS A DEVELOPER IS FAMILIAR WITH, AND EFFECTIVE AS A TARGET IN AN ITERATIVE “EDIT-COMPILE-DEBUG” DEVELOPMENT CYCLE.



Components of virtual prototyping technology

Today's embedded systems need to verify that the combination of hardware and software matches the expected functionality and performance. Performing this task requires control and observation of the hardware, with awareness of the software execution. Another major impediment to starting software development and hardware/software integration is the late availability of the target hardware. Not being able to close on the complete system functionality until the end of the design cycle introduces even more development and product risks.

Today's high-speed deeply embedded processors have changed the landscape for embedded system tools. Several solutions are available featuring different trade-offs in execution speed, level of accuracy, invasiveness, flexibility and cost. Today's SoC technology has made the traditional ICE impractical, as system developers completely depend on processor designers and silicon vendors to provide internal access to processors. OS simulators do not provide sufficient detail and access to the target hardware as they simply emulate the API of the target OS, and no hardware detail at all. Depending on the host/target resemblance, developers often find themselves performing considerable effort in porting the application over to the target. Today's SoC hardware/software co-design and co-verification approaches are geared towards the silicon development. While valuable for

hardware development to evaluate small amounts of software interacting with the hardware, the detailed HDL models limit the performance, typically limited to ~50,000 instructions per second.

To solve the speed issue of co-verification, people have relied on FPGA prototyping systems, massive logic hardware emulation systems, early reference boards, and specifically developed board prototypes. The first two require considerable progress on the hardware development, a complex and elaborate mapping process and moreover, emulation systems are seldom justified because of their high cost. Early reference boards are typically too different, and as the prototyping boards, they can only be built when first silicon becomes available.

The software design process includes many stages, including specifications, coding and integration. But the techniques above usually leave the testing until the end of the process. Each stage introduces bugs. But if you do iterative design and were able to run tests via simulation and find your bugs you'd minimize this. Developers need to embrace newer, more-sophisticated software development tools, simulation modeling being one of them. With virtual prototyping, Virtio offers a technology which delivers a software-based approach for developing embedded software long before any hard-

ware is available, and thus increases hardware availability and accessibility. This technology enables the creation of a software-based embedded platform that can fully mirror the functionality of a target SoC or board. These platforms combine high-speed processor instruction-set simulators (ISSs) and high-level, fully functional, C/C++ models of the hardware blocks, to provide a high-level hardware simulation model.

Although not new, simulation adoption by software developers has been prevented in the past because of its low speed, mostly attributed to its accuracy level, the time to create the simulation model, the failure to integrate with existing development tools and the cost of the simulation tools.

Essential for software developers to adapt simulation technology is the abstraction level of the models employed. In our experience, transaction-level, register-accurate functional models meet the requirements for early software development and integration and essential to achieve the execution speeds required for these phases. Instruction-accurate processor models are combined with abstracted busses, where the bus behavior is modeled as abstract read or write memory transactions. Timing aspects which contribute to the correct system functioning are modeled, as for example real-time clock time

ticks delivered to the OS. With this level of detail, program and register binary compatibility is achieved.

Blamed as being insufficient, these models can provide detail profiling information at various levels, for example cache behavior, trapping and exception statistics and detailed instruction profiling, which can be extrapolated to first order clock cycle performance estimates. Inserting cycle- and/or clock-accurate information in these models renders them too slow for software development, plus the long model development times make them available too late and requires considerable effort.

A virtual prototype builds upon the following components, as depicted in the figure on page 40. Fast processor models or instruction-set simulators, connected to commercial software debuggers, enable the loading and execution of the real software on the virtual prototype. Standard processor peripherals, busses and logic accelerators can be captured as high-level C/C++ models, and compiled and executed on top of a system simulator to capture the hardware portion of a design. Test-bench/human-machine interface models mimic the real appearance of and interaction with the system being designed like the keyboard and LCD of a cellular phone or PDA.

Virtio's prototyping enables modeling ease and extendibility. Although virtual prototyping supports the re-use of hardware IP models, application-specific components are captured in matters of weeks rather than months. The choice of the appropriate abstraction level combined with optimized simulation technology renders models which feature approximately 15...20 MIPS (per GHz of the simulation host) performance. Commercial operating systems, including Windows CE and Linux, are booted in 10...15 seconds, making the virtual platform effective as a development target, in an iterative "edit-compile-debug" cycle. When employed with the right modeling accuracy, virtual prototyping supports the development of production-quality software, requiring no change when the physical system becomes available. Virtual platforms offer physical connections to the host computer's network, keyboard, mouse, USB, PCMCIA card and audio to provide developers access to all the functionality they need to write applications, middleware or device drivers. For example, remote software debugging tools can be connected to the platform over a virtual Ethernet connection.

Virtio's platforms model the touch screen, configuration switches, and support different product skins. This is required for application developers to render screen output on the actual screen form factor, and QA engineers testing

the end product with its actual user interface. Employing a graphical modeling language which extends the C programming language, complete hardware visibility and control is enabled, as supported by a Virtio hardware debugger enabling hardware tracing, single-stepping and break-pointing. Seamless integration with industrial software development environments (IDE) is essential not to disrupt developers' current workflow. Integration happens through standard debug interfaces for low-level control or through virtual physical connections for remote debugging solutions, employed typically for application development.

The integration between the embedded software IDEs and a virtual platform emulating the target hardware creates a powerful desktop development environment, boosting the developers' productivity. Virtual prototyping brings added value to multiple parties, in the first place to system software developers, either HAL, device driver or middleware developers and even application developers. It enables a true early start of software design and promotes the concurrent engineering practices for software and hardware development. Also, it replaces time-consuming hardware/software integration late in the development cycle with continuous integration throughout the whole software development, lowering development and product risk, and increasing development productivity. It has a number of advantages over its physical counterpart, making sure they still make sense even when the physical target hardware is available. It features unsurpassed flexibility and avoids board or silicon iterations when coping with change or design derivatives. A virtual prototype allows unlimited observability and controllability, not limited to the available pins or JTAG on the prototype. In addition, the virtual prototype can be halted and inspected as desired, and supports advanced synchronous multi-core debugging.

Virtio and TI created the VPOM-2420 virtual platform modeling the TI OMAP2420 software development platform, an advanced SoC solution for 3G wireless phones. Available 3 to 6 months ahead of silicon, it enables pre-silicon software development and integration. Seamlessly integrated with TI Code Composer Studio, Metrowerks CodeWarrior, ARM RealView and more, the virtual platform provides a complete software development environment. The platform was used effectively by TI to perform pre-silicon software development, including boot code, OS base porting, device driver, DSP RTOS porting, inter-processor communication and image and video accelerator software development, and this with a measured productivity increase by a factor 2 to 5. It is now available to mobile phone OEMs to accelerate pre-device software development. ■