OVP Design and Consulting Services

Virtual platforms, especially those using the Open Virtual Platforms (OVP), are valuable tools for hardware and software development. They provide an environment for running software earlier in the development cycle, or when using real hardware is not an option. They also provide more precise control and visibility into the target hardware, which allows for many other capabilities, including advanced debugging and automated testing and verification.

A variety of skills and knowledge are needed to take full advantage of these virtual platforms and implement cutting-edge solutions, including the following:

- Embedded software, including compilation tools, debuggers, and target operating systems

- Virtualized software execution and other simulated/emulated hardware techniques

- Modeling of hardware at various abstraction levels

- Electronic design automation, including languages such as C/C++/SystemC (including TLM2) along with verification methodologies

VIRTUALIZING HARDWARE

There are many benefits to using virtual instead of real hardware. One of the big advantages is that software testing and other tasks can be done earlier in the development cycle, before hardware is available or even while the design is still being finalized. Even if hardware is available, a development platform can be made available to more people, at a lower cost, by using virtualization. Implementing a virtual platform may require modeling of processors and peripherals, and combining these into a working platform.

BOOTING THE OPERATING SYSTEM

Once a hardware platform is available, it may be necessary to run an operating system or other core software on it. This initial booting up of the system can be a time consuming process, often requiring patching and recompiling of the kernel. Virtualization eases this process by providing more visibility into the running system, even when fatal errors occur. Several mechanisms are available in the OVP models, such as callbacks and intercepts, to assist with this. Booting an operating system requires execution of a significant amount of code. Having fast simulation models can save a significant amount of time during this process.

RUNNING THE TARGET APPLICATION

At some point in the development cycle, testing must be performed on the target application. This may be running individual unit tests on each module, or running the entire system-wide application. Having increased visibility into, and control of, the platform greatly assists with this. Assertions or other checks can be added to the platform to monitor for significant events without modifying the source code. If a problem is encountered, additional debugging capabilities are available that could not be done on real hardware, even with JTAG or other mechanisms.

Posedge Software, Inc.

Web: www.posedgesoft.com
Email: info@posedgesoft.com
Phone: (612) 548-4784

Posedge Software has expertise in the listed areas, and provides design and consulting services for the development of virtual platforms and embedded software using Open Virtual Platforms and the Imperas M* tools.

ANALYZING SOFTWARE EXECUTION

Tests are written to check that the software is giving the correct results. However, other metrics may be needed to verify the software is correct and sufficient testing has been done. Coverage, profiling, or other performance tests are often used. While it is usually difficult to gather this information without recompiling or making other changes to the software, using virtual platforms this can be done transparently, without making any modifications to the target software.

AUTOMATING SOFTWARE TESTING

In the hardware world, a number of tools and languages have been developed specifically for verification. They provide features such as automated running of tests, constrained-random generation, and coverage-driven verification. OVP can be used along with the Cadence Incisive platform, or specifically with the Virtual System Platform. This combination of virtualization technology and hardware verification methodology results in a robust, flexible verification environment needed to handle the complexity of current and future software projects.

CO-VERIFYING HARDWARE AND SOFTWARE

Besides testing hardware and software independently, or just in a system-wide context, hardware and software components can be tested together to check that they are interacting correctly. This co-verification is useful in testing corner cases, or other cases that would be difficult, if not impossible, to test otherwise. OVP models are well suited to this task as control and visibility into both the hardware and software is available.

VERIFYING THE SYSTEM

In addition to using hardware verification tools for software verification, they can also be used for system-level verification. While these tools have been used for system verification for some time, only a small amount of the system's software is executed. This is mostly due to slow processor models, often implemented in Verilog, which require enormous amounts of computing power, and are generally limited to running smaller portions of the software. By using OVP, the complete stack of software can be used without greatly increasing the amount of time needed to simulate them.