

MIPS to the core: Q&A with Noam Shendar, director of Strategic Marketing, MIPS Technologies, part five

Chris Hall, DigiTimes.com, Taipei [Wednesday 18 January 2006]

Embedded processors are now central to the converged environment that many in the IT and electronics industries are calling the digital home. Noam Shendar, director of Strategic Marketing for MIPS Technologies, argues that MIPS offers superior performance in the embedded processor space, and for this reason has secured a dominant position in the set-top box market, for instance, a dominance the company is determined to extend to IPTV and the IP set-top box. Now capable of clock speeds of up to 400MHz in 130-nanometer geometry, MIPS processors also offer stability, a proven architecture and software base, a secure "trusted environment" for digital rights management, and dynamic compilation for faster execution of Java.

DigiTimes.com spoke to Noam Shendar in Taipei, covering a range of market issues as well as the technologies that are now available and enabled with MIPS cores.

This is Part V of a six-part interview. Part VI will follow on 19 January. Part I appeared on 12 January, Part II on 13 January, Part III on 16 January, and Part IV on 17 January.

Q: I understand that the MIPS32 4KSd secure core can also use software from Trango Systems, for enhanced security. How does that work?

A: There are two models for the use of a core like the 4KSd. First, it can be a host core. In other words, it could be implemented as a single core running both secure and insecure software. In this case, the secure memory management unit on the 4KSd is used to enforce isolation of the secure software functions from the insecure software. That can be done, for example, by running the secure software at a higher privilege level than the insecure software.

The second model consists of using two cores. Several of our customers are doing that. For a host core, they typically take a larger core, such as our 24K, which offers the performance needed for running those rich user interfaces that I mentioned, and then next to it they place the 4KSd core, which they then use to run a secure box – that is, a trusted environment – for DRM. The end result, as you can imagine, is definitely isolation since you have two cores, and what's running on the host core cannot get to the other core. So if, say, a hacker is trying to access the host core, they are really talking to the wrong part of the device. The sensitive information is being run on a separate core, one that is not accessible to the user.

Before we get to Trango's software, let's take a look at the relative advantages and disadvantages of the two approaches. Using the 4KSd alone occupies less space, but it requires very careful architectural design to ensure that there is a true separation between the secure and insecure software. In contrast, with the two-core solution, the security is trivial. You run what is not secure on the host core, and what is secure on the second core, the secure 4KSd. The drawback to using two cores is size since you now have two cores instead of one.

What Trango does is give you the best of both worlds. It takes a single MIPS core and splits it into two or more virtual cores, by software means. And this can be done on any of MIPS' cores. Using the 4KSd would be preferable from the point of view of security, but actually any MIPS core will run the Trango software. Through a very thin layer, the Trango software virtualizes the MIPS core into two or more – three, four or more – virtual CPUs, and these are isolated by means of the Trango software, which is known as a hypervisor. The hypervisor makes sure that one virtual CPU cannot talk to another virtual CPU, and in this way it provides the same isolation that two cores would provide.

The thinness is important for two reasons. One is performance. A thick layer would create inefficiencies and add a lot of latency to the system. A thin layer does not. The other reason is that a thin layer can be verified to be secure. In other words, there can be one-hundred-percent monitoring of the different states of that software, and it can be proven mathematically that the layer is secure, that there is no flaw or back door in the layer, which could open the door to a breach of system security. What's very exciting about this is that process and data isolation can be achieved on any MIPS core, past present or future, using Trango's software.

Q: How would you characterize the relationship between MIPS and Trango?

A: It is a collaborative relationship. Trango is a third-party software provider. If any of our customers are interested in licensing software direct from Trango, we are happy to arrange an introduction. And there is also collaboration between the two companies at the technical level, where the technical discussion between the two companies serves to improve the performance and security of the Trango software as it runs on the MIPS architecture. In other words, we help them make the most of our architecture.

Q: Those discussions can be between MIPS and Trango and also between MIPS' customers and Trango?

A: Correct.

To sum up, thanks to Trango, customers and designers now have three choices: a single hardware core, two hardware cores, or a single hardware core that is hypervisor enabled for dual virtual cores. Customers are embarking on all three architectures, depending on their requirements.

Q: Let's switch the software focus to Java. I understand that MIPS is now convinced that dynamic compilation under Java is superior to hardware acceleration. Can you explain how MIPS achieves dynamic compilation and why it is superior?

A: The execution of Java introduces a lot of inefficiencies to the system. When people say that Java requires a lot of performance, they are right. What they mean by that is that running Java can be an inefficient process, and what would otherwise be a simple program when written in C, when written in Java requires performance equivalent to a much more complicated C program.

There are many reasons for that. One reason is that Java bytecode, as defined by Sun, is unlike any machine language, and it cannot be translated efficiently into machine language. Also, it does not allow pointers, so that when data needs to be moved from one function to another, or from one procedure to another, it needs to be copied, and this introduces a lot of overhead. Pointers are much more efficient.

As well, Java does not support the use of registers. It uses a stack instead. As any programmer or CPU architect will tell you, especially in the RISC world, register operations are far more efficient than data operations where you have to push data onto the stack and then pop it from the stack when you use it. All of that adds up to a lot of processing for a relatively simple program.

Some have tried to improve the performance or alleviate the problem in hardware by adding extra gates and creating a pipeline to run Java more efficiently. These attempts to run Java directly in hardware are, in effect, attempts to create Java processors. Sun famously tried that. Other companies have also created Java chips, but none of them has had much success, for one reason – hardware execution of Java is either fast or small, but it can't be both.

Efficient execution of Java directly in hardware requires a lot of hardware resources, and any time a shortcut is attempted there are costs in terms of performance. What MIPS and its software partners realized – especially in partnership with a Swiss company by the name of Esmertec – is that software approaches can be much more successful than hardware-based solutions. The reason is that you can take Java and compile it to machine language and thereby by-pass all of the limitations that I mentioned.

If you take Java bytecode and compile it to machine language – in this case, MIPS machine language – the machine language is practically the same machine language you would get if you were to write the software in C in the first place and then compile it. So the overall performance is native performance, unbound by the limitations of Java. The very impressive numbers that come out of that show that dynamically compiled Java running on MIPS runs 50% faster than hardware-accelerated Java running on some other architectures. That's very significant. It validates the assertion that compilation is a more efficient technique than hardware acceleration of Java execution.

This is Part V of a six-part interview. Part VI will follow on 19 January. Part I appeared on 12 January, Part II on 13 January, Part III on 16 January, and Part IV on 17 January.



*Noam Shendar, director of Strategic Marketing, MIPS Technologies
Photo: Company*

Related stories:

Despite aggressive moves from AMD and VIA, Intel still dominates embedded x86 market (Jan 18)

MIPS to the core: Q&A with Noam Shendar, director of Strategic Marketing, MIPS Technologies, part four (Jan 17)

MIPS to the core: Q&A with Noam Shendar, director of Strategic Marketing, MIPS Technologies, part three (Jan 16)

MIPS to the core: Q&A with Noam Shendar, director of Strategic Marketing, MIPS Technologies, part two (Jan 13)

MIPS to the core: Q&A with Noam Shendar, director of Strategic Marketing, MIPS Technologies (Jan 12)

SMIC to produce configurable processor for ARC (Jan 9)

The core of the matter: Talking with Tensilica (Sep 7)

ARM advances: Q&A with ARM VP John Cornish, part two (Aug 15)

A briefing on the Blackfin: Q&A with Analog Devices, part two (Jul 13)

Designing for the digital home: Q&A with Kun-shan Lin, VP, Texas Instruments, part two (Jun 17)

NEC Electronics and ARM announce partnership on next-generation multi-processor-based CPU cores (Jan 12)

© DigiTimes Publication. All rights reserved.
Please do not republish, publicly broadcast or publicly transmit content from this website without written permission from DigiTimes Publication. Please contact us if you have any questions.