



May 5, 2008

*The industry newsweekly for the creators of technology*

# Commentary: What's the deal with embedded Linux?

**By Kenton Williston**

Last week Green Hills Software wrote a scathing opinion piece on embedded Linux. Here's the opening:

*"Embedded Linux is the most hyped embedded operating system ever. It is promoted as inexpensive, high quality, high productivity, reliable, widely available, and well supported. It is none of these things..."*

Ouch! Green Hills goes on to congratulate Linux vendors for admitting that the OS is "CHAOS" (thanks, Wind River!) and "a money pit" (you too, MontaVista!). But the praise is short-lived: According to GHS, this is a cynical ploy to scare you away from do-it-yourself Linux and turn you into a Wind River or MontaVista licensee.

That's some tough talk, but Green Hills Software is hardly a neutral observer. The company sells its own operating systems, so it stands to benefit by sowing fear, uncertainty and doubt (FUD) about Linux. Indeed, every vendor I have spoken to has taken a distinctly self-interested position on Linux. Here's a sampling of the claims I've heard:

**MIPS:** Do-it-yourself Linux works great! Just go to LinuxMIPS and download! Our Linux support makes us a winner!

**Wind River, MontaVista:** Free Linux is a disaster. Buy "real" Linux from us or you will end up crying over your keyboard!

**QNX:** Linux is a disaster. But our Linux-like RTOS is great!

**Microsoft:** Geez, why are you even looking at Linux? Life is much easier with WinCE.

**Mentor Graphics:** Linux? WinCE? You have to be kidding. You can get everything you want from a light-weight RTOS like Nucleus.

One thing is for sure: There is no shortage of opinions on Linux. But which vendor is telling the truth? Or are they lying in hopes of building sales?

The reality is that each viewpoint contains elements of truth. Every project has unique requirements, so different projects call for different operating systems. Here is my (non-partisan) list of questions you should ask yourself when evaluating embedded Linux:

**Do you need a full-featured OS?** Embedded Linux is big; typical builds exceed 2 MB. Sure, you can shrink the OS by stripping out things like networking stacks and file systems, but these features are the main reason to use Linux. If you don't need these features, you are better off with a lightweight RTOS.

**Can you get an OS with application-specific features?** WinCE comes in numerous flavors, including versions designed specifically for automotive applications. (The same is true of QNX.) Nucleus has community special features for portable media players. And so on.

**What is the licensing model?** The Linux General Public License has its drawbacks. What if you want to modify the kernel, but don't want share your hacks with the rest of the world? What if unauthorized code sneaks into the kernel, and the owner decides to sue? Questions like these are mainly issues for products with long life spans, like cars and network infrastructure. The rest of us can often ignore the legal issues and simply update the kernel in the next product rev.

**Is Linux responsive and reliable enough?** I know what you're thinking: Isn't embedded Linux specifically designed to address these

points? Yes, but embedded Linux can't match OSs like INTEGRITY.

**Do you want to pay now or later?** Do-it-yourself Linux is royalty-free, but you have to make a major engineering investment to get it up and running. In contrast, a commercial Linux package (or a competing OS) can get you to market with minimal up-front cost.

**How many units will you ship?** If your volumes are low, it doesn't make sense to have your own OS team—so skip the do-it-yourself approach.

**What is your time to market?** The do-it-yourself approach doesn't make sense if you're in a hurry.

**Is there support for your specific processor, board, or reference design?** The OS with the best support will give you the lowest NRE and shortest time to market.

**Will you have to port your existing code?** Most projects are built on existing code. Waste too much time porting the code, and you lose the benefits of switching OSs. (You can sidestep this issue with virtualization, but that adds another layer of complexity to the system.)

**Does your tool chain support the OS?** If not, you will have to switch. This adds to the learning curve and makes the design team cranky. If you are willing to switch tool chains, look for OS-aware features like MIPS' Linux hot-spot analyzer.

Whatever you decide about embedded Linux, it's a good idea to take vendor claims with a grain of salt. For example, Green Hills talks tough about Linux, but the company's MUTLI IDE has supported embedded Linux since 2001. Ask yourself: if GHS *really* thinks embedded Linux is a disaster, why does the company support it?