

There's More to an RTOS than Threads

Jon Titus, Senior Technical Editor



When engineers think about using a real-time operating system they worry about task timing, interrupt latency, and other timing- and task-related concerns. But other aspects of adopting an RTOS deserve equal attention.

"Keep an eye on your project's future growth and expandability," said Andy Gryc, product marketing manager at QNX Software Systems. "As product requirements grow or change you might have to turn to a multicore processor, but many small RTOSs cannot adapt to a multicore environment. Do you lock into one approach or plan for a broader route?"

A multicore processor can help solve timing problems in a single-processor system. "Adding a multicore processor to a hardware design shouldn't require anything special when an RTOS includes symmetric multi-processing [SMP] capabilities," said Gryc. "Engineers might think they have to chop up their code and manually divide it between processors. Generally, though, most embedded software runs multiple independent tasks. And because those tasks already run in separate threads, they can run on a multicore system. The RTOS scheduler simply routes tasks onto whichever core becomes free. The code gets a 'boost' without you worrying about any complexities involved with using multiple cores."

"Many embedded-system engineers might still think of an RTOS as only a low-level kernel that handles threads, mutexes [mutual exclusions], and similar tasks," noted Bill Graham, product marketing manager for VxWorks at Wind River. "But today an RTOS must support a full networking stack, graphics with hardware acceleration, multicore processors, USB ports, CAN bus interfaces, wireless devices, and so on." It's unlikely you want to write the driver code or try to "attach" those services to a do-it-yourself or mini-RTOS.

Engineers should always think ahead about the connectivity options available for a given RTOS. "The industrial people use Modbus, Profibus, and EtherNet/IP, and wireless companies have ZigBee, RF4CE, and other protocols," explained Graham. "But we don't create drivers for all of them. Instead, we work with companies that do. Engineers should know this eco-system exists to help them. The integration of software is worth a lot when you want an off-the-shelf RTOS. Our pre-integrated tools and run-time components are part of the attraction of VxWorks."

There's More to an RTOS than Threads

Published on Electronic Component News (<http://www.ecnmag.com>)

"Moving away from a mini-kernel or a do-it-yourself RTOS frees you from having to support all the new technologies your software developers want in an operating system," stressed Gryc. "And a commercial RTOS vendor and its partners can better track industry standards and create drivers and services."

No matter what RTOS they choose, eventually all embedded-system engineers need to work with real hardware. "So it's important that an RTOS company have a board-support package, or BSP, for the processor family or architecture you plan to use," said Gryc. "We support a large number of BSPs, so you can develop code on an old X86 board and then move the code to the processor or board for your final design. There are always some hardware changes, but in general the drivers we create appear agnostic to the underlying hardware. So we can create one driver to support ARM, PowerPC, and other architectures. That's invisible to the engineer."

"Engineers who acquire an RTOS depend greatly on specific-board support," said Graham. "But when they come from the do-it-yourself world they might get a surprise: The processor or board they selected might not have full support by RTOS vendors. Even the largest vendors can't support every single platform out there. Engineers must think ahead and do some homework." RTOS vendors typically create a BSP for a hardware vendor's specific development board or reference design.

"There isn't always an exact fit between an RTOS and the engineers' final hardware," noted Graham. "So engineers and managers must understand they will need some commercial services from their RTOS vendor. When you research an RTOS, you should have on your list of questions, 'Does the vendor offer training, support, and services?' Manufacturers of consumer, avionics, and military equipment almost always need some customization."

When you get involved with aerospace or defense projects you must also think about software certifications," said Graham. "Does your RTOS vendor have certification evidence that shows conformance to DO-178B for avionics electronics or IEC-61508 for industrial equipment, for example? In those cases, an RTOS with the required certifications or certification evidence makes more sense than an open-source operating system."

"It's difficult for a small company that relies on Linux to get the certification needed for safety-integrity-level 3 or FDA approval," said Gryc. "Generally an engineer knows about those types of certifications for his or her industry, so they realize it's easier and safer to buy a commercial RTOS than to use DIY code or an open-source OS."

For further reading:

"System Architecture," QNX Software Systems.
www.qnx.com/developers/docs/6.3.0SP3/neutrino/sys_arch/ [1].

"Working with a BSP," QNX Software Systems.

There's More to an RTOS than Threads

Published on Electronic Component News (<http://www.ecnmag.com>)

www.qnx.com/developers/docs/6.4.1/neutrino/building/bsp.html [2].

Parkinson, Paul and Larry Kinnan, "Safety-Critical Software Development for Integrated Modular Avionics," Wind River. <http://www.windriver.com/whitepapers/> [3]. Registration required.

Wind River. <http://www.windriver.com/whitepapers/> [3]. Registration required.

Source URL (retrieved on 12/04/2013 - 11:49am):

<http://www.ecnmag.com/articles/2010/11/theres-more-rtos-threads>

Links:

[1] http://www.qnx.com/developers/docs/6.3.0SP3/neutrino/sys_arch/

[2] <http://www.qnx.com/developers/docs/6.4.1/neutrino/building/bsp.html>

[3] <http://www.windriver.com/whitepapers/>