Why Use a Commercial RTOS?

Reasons to NOT Develop An In-House RTOS

by
Ralph Moore
***smx* Architect**

The Objections

"Our application is too simple for an RTOS"

I hear this statement at least twice a week. It tells me very little about the project, but it does tell me that the speaker has no vision of using a commercial RTOS in his project. This statement is comparable to "I don't need a power saw because I only have to cut a few boards." Clearly, that speaker is aware only of the speed advantage of a power saw. He is not aware that the power saw makes much more precise cuts and thus his boards will fit together better and his project will come out better.

A commercial RTOS is similar to a power saw. Many people think multitasking is the only benefit of a commercial RTOS. They have not taken the effort to look deeply into the features it offers and they have not made an effort to envision how these features would benefit their project.

"It will only take 1-2 weeks."

People who think they can invent things quickly often know little about what they plan to invent. It is common sense that if the average RTOS costs over $10,000 and the average programmer costs $7500 per month, then an acceptable RTOS cannot be designed, coded, and debugged in 1-2 weeks.

"The learning curve is too long"

Other than for Linux or Windows, I am surprised that anyone has the chutzpah to use this argument against buying a commercial RTOS. It is obviously much harder to invent than to learn. Most RTOSs are designed to be easy to learn and use (maybe because those of us in the RTOS business run into this objection so often!)

Reality Check

Don’t pay expert wages for apprentice work

Would you pay a race car mechanic’s wages to change tires? Of course not. Yet, your staff is undoubtedly expert in the requirements of your industry and you are paying them accordingly. So, does it make sense to assign them to do something they know little or nothing about? Absolutely not! Such work should be subcontracted out – and buying a commercial RTOS is the best way to do so. (In fact, you should be looking at the cost of an RTOS in exactly the same way as you look at the cost of an outside consultant, because you are buying outside expertise in both cases. Embedded products are not sold in sufficient volumes to achieve the mass market prices of WindowsXP or Word.)

What’s not in the RTOS ends up in the application

Lack of vision is the cause of this. Instead of making maximum use of proven commercial code, your crew is reinventing the wheel with fresh, buggy code. Is this the best use of their time?

An in-house RTOS is never done

As your crew learns what they didn’t know about RTOSs, the one to two week estimate lengthens, thus robbing time from development of your product. This is the downside of not hiring outside experts in a craft that is not central to your business.

There is never time to document an in-house RTOS

Naturally! You want your crew doing more important things – like finishing your project. Hence, comments in the RTOS code are out of date or non-existent and there is no manual. Nor has the RTOS been kept up to date with advancements in tools and processors. (Nor is there much hope of it being compatible with any other software package that you might want to add to your product.)

The inevitable crew change

A couple of years down the road you are left with a quirky, undocumented RTOS, which is the heart of your product. The crew who designed it has moved on to greener pastures. (Too bad about yours, but at least they will never make that mistake again!) The replacement crew is faced with a really steep learning curve (and cursing the creators of this monster). Thank God for component obsolescence! It justifies chucking the whole design and starting over with a commercial RTOS. (You have learned something too!)

Commercial RTOS Benefits

Now let's talk about doing it right. What does a commercial RTOS offer, that makes business sense?

No wasted time deciding what to do, doing it, etc.

The RTOS is already done. Just learn it and get on with the project!

Thorough documentation

Most commercial RTOSs come with a User's Guide, a Reference Manual, and well-documented code. Not only that, these are accurate and up to date. Most RTOS vendors also offer training.

Proven code

A commercial RTOS is not only debugged, it is *proven*. It has been used by hundreds of users in a multiplicity of ways. Moreover, it continues to be used and tested and latent bugs continue to be found and fixed. A tough RTOS bug can take weeks to find and fix.

Board support and tool integration

A good RTOS comes with a BSP and integration with a good tool suite. These, alone, are often worth the cost, in programming time, of the RTOS. (BSP means "Board Support Package". It includes startup code and device drivers for most on-chip peripherals as well as for a specific development board.)

Well-structured result

The most important benefit of using a commercial RTOS is the resulting well-structured application code. A commercial RTOS requires a programmer to implement his application in the form of pre-defined objects such as tasks, messages, semaphores, etc. There are firm rules for creating and using these objects. A good RTOS detects misuse and informs the programmer immediately, thus saving days of debug time. Equally important, the code ends up with a structure that makes it easier to understand and change in the future – even by a different programmer. The lack of this capability by the typical non-RTOS based spagetti code is a major problem faced by many companies, and it limits their ability to compete successfully.

Universality

Most commercial RTOSs do standard things in standard ways. The result is greater compatibility with other embedded software products. Good commercial RTOSs offer a wide selection of

already integrated products such as file managers, networking stacks, graphical user interfaces, etc. You may need one of these in the future. The inability to easily plug one in hurts competitiveness.

Technical Support

A few good words can save hours and days of frustrating, wasted effort. Where do you get support for an in-house RTOS?

Conclusion

The idea to develop an in-house RTOS may be defensible technically (in a few cases), but it seldom is a good business decision.