

As mcus become more capable and multicore devices appear, less functionality needs to be implemented in discrete hardware. But more capable hardware often brings an increase in software complexity.

Software development has long been regarded as time consuming, so with time to market an important factor, what is the best starting point for embedded software development: build, buy or open source?

Our experts were asked what proportion of the product development process is now software? John Carbone, vp of marketing for Express Logic, said:



**Carbone: "Follow best practices related to the selection of the tools and solutions."**

"Our customers tell us that up to 75% of a product's development is now software. This represents a significant shift from just a few years ago." David Kleidermacher, Green Hill Software's cto, said software can represent up to 90% of some projects. Meanwhile, Yi Zheng, product manager with QNX Software Systems, pointed out: "A large percentage of our customers do not develop, or no longer develop, hardware. Typically, they choose their hardware at the beginning of a project and focus mainly on software. This is a cost effective approach, supported by a highly consolidated and fiercely competitive hardware market."

How has the embedded software development task changed? Carbone

# Build, buy or open source?

As software takes the lion's share of project time, which option should developers take? By **Graham Pitcher.**

said embedded software developers today use more highly integrated hardware, rather than trying to implement all of a product's functions themselves. "Overall, new software development tools enable increased productivity," he said, "and commercial rtos and middleware offerings, coupled with highly integrated mcus, save software developers huge amounts of time in getting new products to market."

Kleidermacher noted that greater software content/complexity has forced embedded designers to focus on how to optimise software development. "How to minimise software bugs early in the process, increase software reliability through process controls and minimise software integration time."

Robert Day, LinuxWorks' vp of marketing, added: "Embedded developers have to deal with the growing availability of multicore processors, which is changing the software development process to be more system orientated in its approach."

What has brought about this change? Zheng said several things came to mind. "First, rapid advances in hardware make it possible to support a sophisticated software system. Second, the pressure to lower the Bill of Materials through hardware consolidation is forcing developers to fit more and more functionality onto a single board or processor. And let's not forget customer expectations: customers invariably want more, not fewer, features."

Jim Douglas, senior vp of marketing with Wind River, noted: "Open source has changed the software development process. But it's not trivial to take a bunch of components and integrate them; you can't get, for example, open source carrier grade Linux."

Day added: "A major trend is the expectation that embedded systems of all kinds will be 'connected' – which adds extra complexity to the system, and increases the protection mechanisms required to ensure that devices remain secure from external tampering."

So what challenges does increased complexity mean for developers and for suppliers to the market alike?

"For developers," said Carbone, "the challenge is to select the best technology

**Kleidermacher: "Bring in the experts; it's really as simple as that."**





**Zheng: “Focus on good software design practice.”**

for the job. It is easy for them to be misled into using a product whose features or speed might not be relevant to the success of the product they are developing. In a sense, ‘less’ is often ‘more’ and developers should be careful not to get caught up in ‘feature envy’. For suppliers, the challenge is providing developers with tools that actually help them reduce time to market, rather than simply making something faster or more functionally robust.”

Douglas contended: “While multicore processors are being adopted, software isn’t being written to take advantage of them. The big challenge for the industry is to help people to implement processes, educate them and get them to the point where they’re productive. And tool providers should also be aggregators, building vertical solutions.”

Kleidermacher: “Developers must embrace modern methods for managing software complexity, including new debugging techniques and new run time environments, in order to keep pace and prevent the software schedule from getting out of hand.”

Zheng said software suppliers must decide which hardware to support, and to what extent. “At the same time, they must decide how long to keep supporting older hardware. Striking the right balance is a mixture of art, science, and experience:

suppliers must understand what customers need today and anticipate what they will need in the future.”

What are the relative benefits of building software, buying it in and accessing open source software?

Douglas: “Embedded mobile needs an OS, middleware and something for the application space. It’s not all available from open source and therefore it will require integration. While we won’t go back to the days of pure vertical integration, pieces of core technology are needed to differentiate products.”

Carbone believes building with open source software can make software development more costly, error prone and slower. “Buying software offers faster time to market, portability to other hardware platforms, integrated components and the availability of support from the supplier. A proven solution with a strong track record of adoption and successful use can reduce risk of failure, just as using modern mcus saves time and reduces hardware cost.”

Kleidermacher said it depends on many factors. “A small amount of open source code can be managed by a small set of developers. A large amount of open source code must be managed by an exponentially larger team.

“Also, there are varying levels of



**Douglas: “Have a scalable system that meets customer expectations.”**



**Day: “Don’t reinvent the wheel.”**

software quality, licensing and IP concerns and varying levels of technical support. In short, the decision to incorporate open source software into an end product must not be taken lightly. That being said, open source has the potential to bring dramatic gains in functionality relative to cost and time to market, if incorporated properly.”

Day claimed building software will always have a place: “Without it, there is no differentiation. Using open source is often a cost driven decision; you don’t have to pay for the software, but you will have to pay to support and modify it. Whether the economics are attractive really does vary on a case by case basis.”

Final thoughts from the experts? Carbone: “Follow best practices related to the selection of the tools and solutions. By avoiding the ‘reinvent the wheel’ process, developers can open up more time for the addition of differentiating features.”

Kleidermacher: “Bring in the experts; it’s really as simple as that.”

Day: “I’d like to inscribe the words ‘don’t lock yourself in’ on every embedded developer’s screen saver.”

Zheng: “Choose your hardware prudently; ‘bleeding edge’ may not be the best option. And focus on good software design practice.”

Douglas: “Set up your process so you have a scalable system that meets customer expectations.”