WHITE PAPER

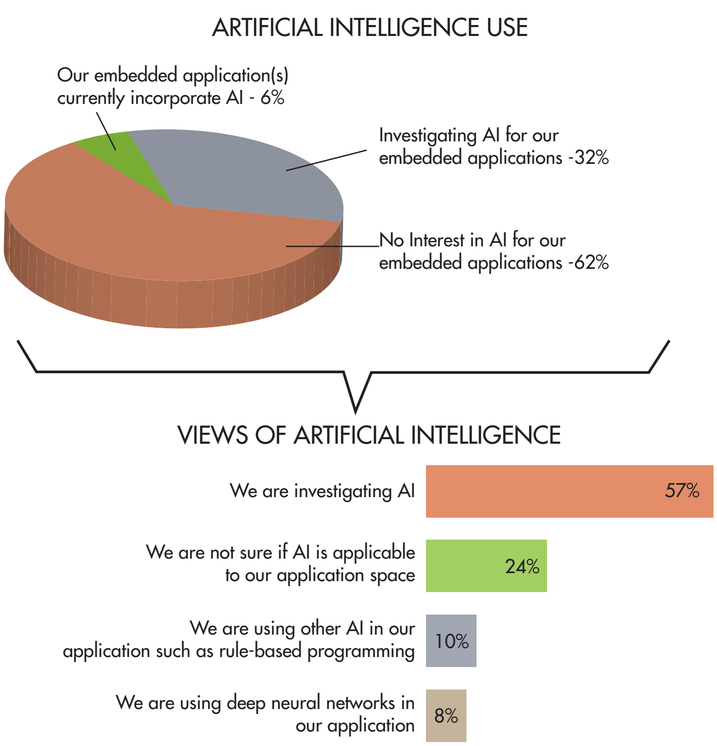
Developers are Chasing AI, Open-Source, and Security Technologies

Technology Editor **BILL WONG** *presents the results from* Electronic Design*'s Embedded Revolution survey.*

FINDING OUT WHAT TECHNOLOGIES DEVELOPERS ARE interested in learning more about is important to *Electronic Design* because we want to deliver interesting as well as useful technology articles to our readers. Our latest Embedded Revolution survey was designed to do this and now we can share the results with you.

CUTTING-EDGE TECHNOLOGIES

Artificial intelligence (AI) has been around since computers were in their infancy. AI covers a lot of ground, including machine learning. Machine learning incorporates a number of approaches such as behavior-based programming that was used on an 8-bit micro in the first iRobot Roomba.



1. More than half of the developers surveyed are looking into artificial intelligence, especially deep neural networks.

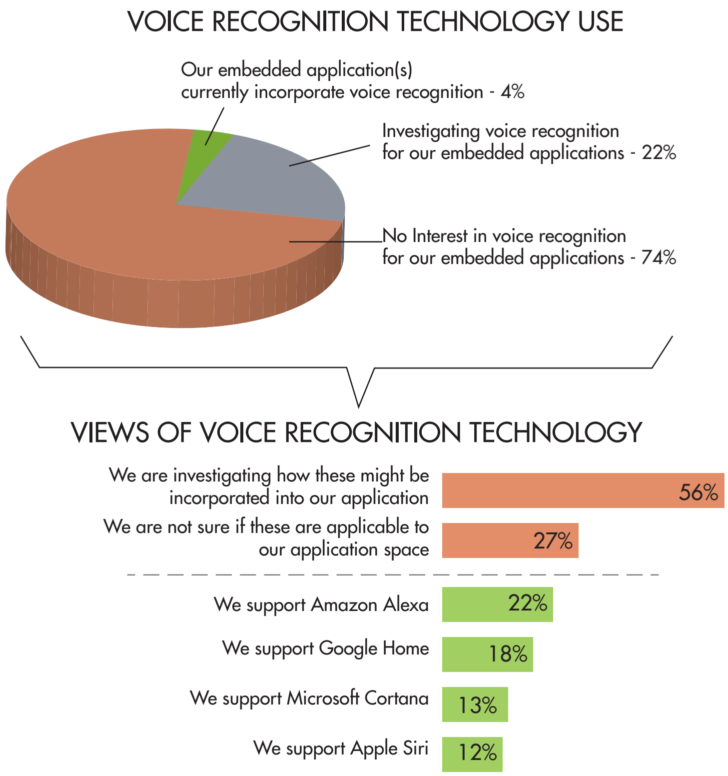
***CHRISTOPHER ROMMEL,* Executive Vice President, IoT & Embedded Technology: "For years, engineering organizations have pursued differentiation by leveraging the latest technology to build more intelligent systems. Now, the ability to harness machine learning and neural networks has extended beyond the data center and is reshaping the way OEMs think about product development. We expect this to drive adoption of new runtimes and analytics software as well as investment in more sophisticated algorithm engineering and simulation tools."**

These technologies are still useful and worth investigating but lately the buzz has been around deep neural networks (DNN), also known as deep learning *(see "Is It Time to Learn About Deep Learning?" on electronicdesign.com)*. DNN can be used for a wide range of machine learning applications from recognizing objects for self-driving cars to playing the game of Go. (Google DeepMind's AlphaGo program beat the South Korean Master, Lee Se-dol, this year.)

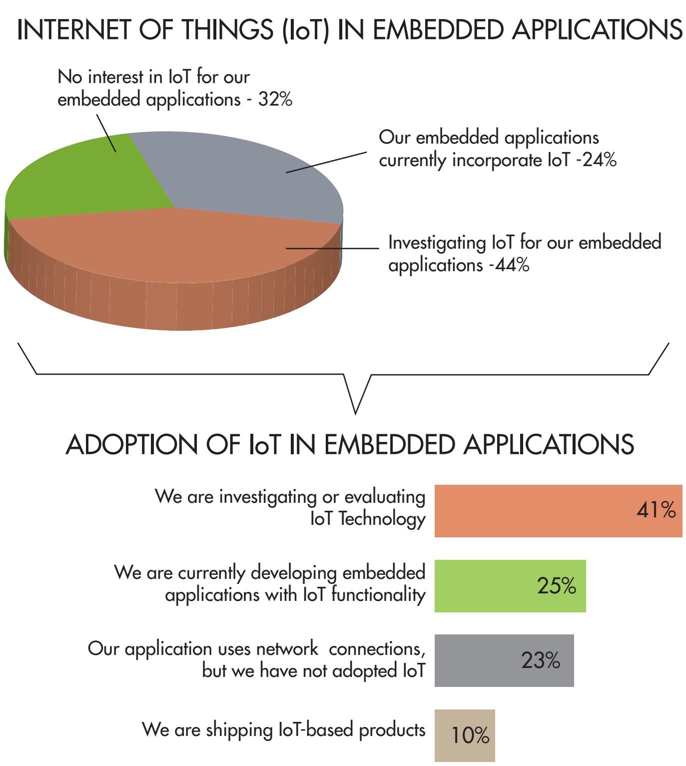
Over half of the developers surveyed are looking into AI *(Fig. 1)*. DNN is what most will be looking at. Only a small fraction of developers are actually using DNN at this point. We found that even more are using other AI technologies.

A number of things have changed the way people look at DNN. The cloud and high-performance embedded computing (HPEC), especially GPUs, have allowed DNN training to be practical. Training often means examining hundreds to millions of data entries such as pictures when a DNN is being used to identify objects in the photo. Training configures the weights used in a DNN.

The other change is high-performance platforms at the edge that include custom hardware or GPUs to utilize DNNs configured by HPEC platforms. It allows DNNs to be used in applications like advanced driver assistance systems (ADAS).



2. The majority of developers are taking a look at voice recognition systems like Amazon's Alexa and Apple's Siri.



3. About a third of the developers were not interested in IoT for their products. That leaves a lot where IoT may have an impact.

Machine learning is also playing a part in voice recognition systems like Amazon's Alexa, Google's Home, and Apple's Siri *(see "Speaking of Orders: Who's Winning?" on electronicdesign.com)*. Voice recognition systems are another technology that a majority of developers are interested in *(Fig. 2)*. These systems are part of the Internet of Things (IoT) because they typically employ devices that link to the cloud where much of the heavy lifting is done. It can also tie the user into cloud-based services like the ability to use Amazon Alexa to place orders for products from Amazon. Of course, it can also be used to get a joke or information, like the weight of Mars.

IoT and voice recognition systems are not applicable to all applications but our survey results show a surprising number of developers using one or more of the popular voice recognition systems. Younger developers are more apt to incorporate voice recognition technology into their products.

Of course, voice recognition is not the only way IoT is being incorporated into applications.

IoT CONTINUES THE WAVE

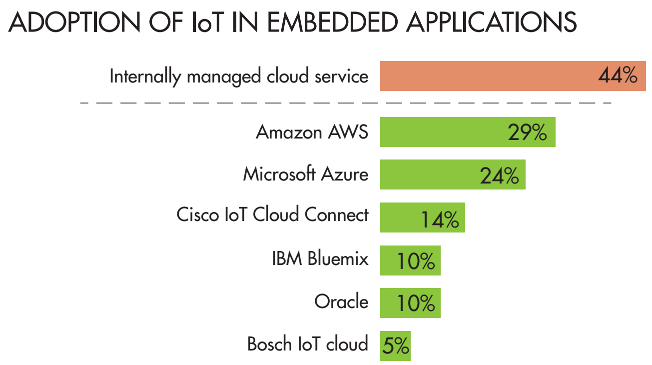
If it is wired or wireless, then developers are likely to see if the device can be connected to the internet, which leads to IoT. Wired connections usually mean Ethernet but wireless technologies are numerous and growing. It can be short-range Bluetooth or long-range LoRA or it might be cellular. 4G/LTE is dominant now but 5G is in the wings promising more bandwidth.

We found that a significant number of developers are delivering IoT solutions or examining how IoT might be incorporated into their projects *(Fig. 3)*. Of course, while many of these developers are investigating IoT, it is not a forgone conclusion that it will be applicable.

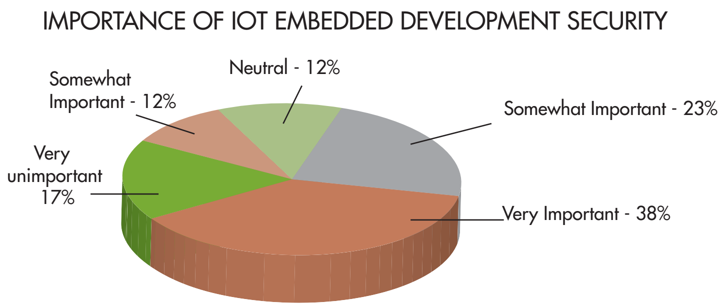
IoT technology is one area that *Electronic Design* is covering closely. The challenge with IoT is that there are many technologies, services, and techniques involved. It is also rare for a single firm to develop or deliver all aspects of an IoT solution and many implementations will bring a large number of vendors into the fold to provide a complete solution or product.

Cloud services are just one aspect of an IoT solution and there are a lot of major players in this space. We asked about a few of these vendors and found that many developers are using internally managed cloud solutions *(Fig. 4)*. Amazon AWS and Microsoft Azure are popular cloud services that are also being used as IoT cloud hosts.

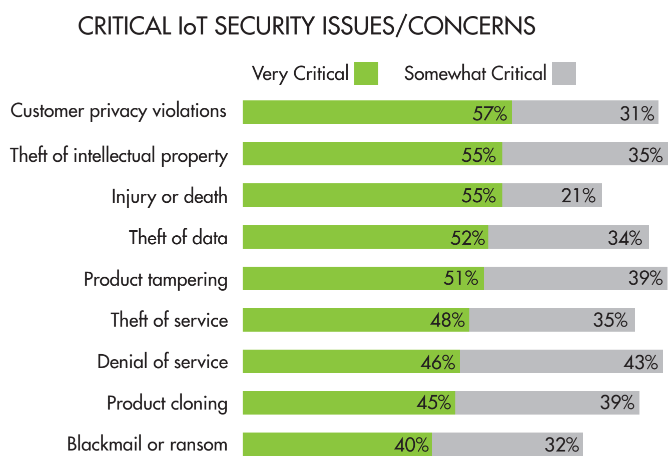
"The *Electronic Design* study correlates to our findings among our customers who are seeking embedded IoT development and cloud solutions," says *WARREN KURISU*, director of product management, Runtime Solutions, Mentor Graphics Embedded Systems Division. "To remain competitive, all of our customers are implementing an IoT strategy and are expressing concerns about development risk, time to market, and security. Mentor is responding by providing a commercial integration of



4. The public cloud is large and growing but a significant number of IoT application developers are using internally managed cloud services.



5. Security is important to the majority of IoT developers but this chart still shows a significant lack of concern for security.



6. Those that are concerned about security are placing safety and privacy concerns at the top of their list. Protecting intellectual property and the products themselves is key as well.

Microsoft Azure into our secure embedded platforms. This pre-integrated solution enables customers to quickly ramp to productivity and more easily address their competitive pressures."

SECURING EMBEDDED SYSTEMS

Security has become much more important to embedded developers as IoT has increased the need for connectivity. Many devices are connected directly to cloud-based services, greatly increasing the attack surface for the overall application environment. This also increases the number of points that need to be protected and that need to employ security-related code or configurations that were unnecessary for standalone embedded devices or even networked devices that would be on an isolated local area network (LAN). This also means that many experienced developers need to learn about new security risks and ways to address them as well as making sure new developers have these skills and understanding.

Gregory Rudy, Director of Business Development at INTEGRITY Security Services, a Green Hills Software company: "Cyber security is one of the most critical issues of our day and has become a question of not 'if' but 'when' an event will occur. Addressing these critical security issues becomes challenging when left unaddressed during product design. Certain issues relate to the target, while others pertain to the communication network and manufacturing supply chain. Retrofitting security after the fact may plug the leak, but usually just pushes the vulnerability to another point in the pipeline. Resolving these issues begins with road mapping cyber security throughout product lifecycle to protect both data and reliability from attack and defects. "

Our survey indicates that security is important to the majority of IoT developers, but this still shows a significant lack of concern for security from many respondents *(Fig 5)*. Keep in mind that these results are associated only with those dealing with IoT applications and not other embedded solutions where physical security is often the only requirement.

The problem is that any connected device is a possible target and there are many ways to attack and utilize a connected device for nefarious reasons from distributed denial of service attacks (DDOS) to ransomware *(see "A Malware Experiment Foreshadows Factories Held for Ransom" on electronicdesign.com)*. Vendors may not be concerned about data, cloning or other security related issues but having a device attacked or a compromised device being used in an attack should concern to everyone.

Those that are concerned about security are placing safety and privacy concerns at the top of their list *(Fig 6)*. Protecting intellectual property and the products themselves is key as well. It is reassuring that at least the importance of these areas is being taken into account.

How effective these efforts will be or how much effort is being used to address each of these areas remains to be seen.

The standard security mechanism are being marked as critical by embedded developers working on IoT applications *(Fig 7)*. Hopefully these results indicate that future IoT devices will be well protected. Of note is the importance of secure firmware updates. This has a twofold implication. First is that the updates are a critical component. Just because a device is connected does not mean that updates will be possible or forthcoming.

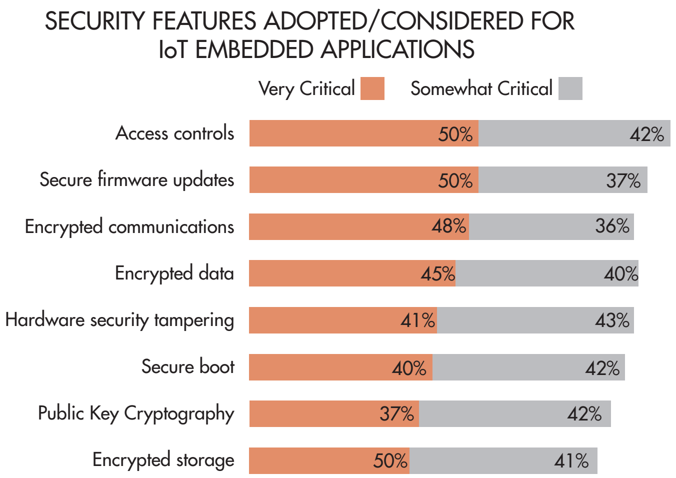


Fig 7: Hopefully these results indicate that future IoT devices will be well protected.

VARIETY HIGHLIGHTS CHIP ARCHITECTURE SELECTION

Chip selection is always a major step in creating product, and developers have never had more options to choose from. Off-the-shelf chips and systems make up the majority of platforms for developers, but a number are taking advantage of custom chip solutions *(Fig. 8)*. Not surprising is the number that are doing in-house system design but a surprising number are doing design in-house chip design. Chip design requires a whole other type of design expertise and the process can be significantly more costly than custom designs, although these costs and the difficulty of doing chip design has fallen. Of course, turning to a contractor is always an alternative.

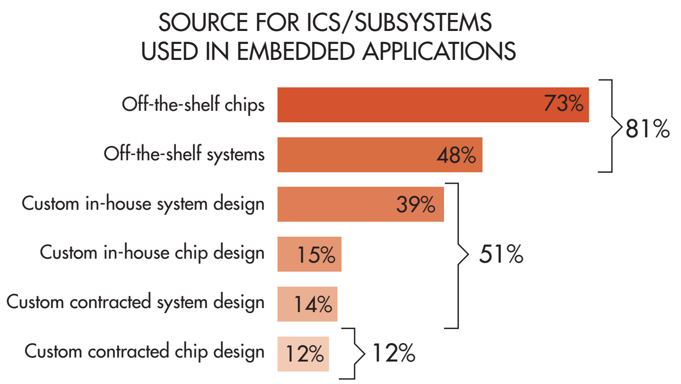


Fig 8: Off-the-shelf chips make up the majority of platforms for developers but a surprising number are taking advantage of custom chip solutions.

TOM STARNES, Chief Processor Analyst, Objective Analysis:
"It shouldn't be a surprise to anybody watching the industry for the past couple of decades that Microchip is doing a great job providing MCUs for embedded systems. The survey results highlight their broad use, especially when combining the PIC, AVR, and at least some (the only MCU) portion of the MIPS devices – even though the architectures are essentially proprietary and all are off-the-shelf. It also reinforces the wisdom of the recent Atmel (AVR) acquisition. On another hand, TI's MSP430 ranking confirms the value of low power consumption. All those ARM processors are spread across numerous vendors and a broader performance range."

A significant amount of designs have employed the 32- and 64-bit ARM architectures in its many forms, including Cortex-M, Cortex-A and Cortex-R but 8- and 16-bit designs are alive and well *(Fig. 9)*. Popular platforms like Microchip's PIC, Atmel's AVR, and Texas Instruments' MSP430 remain popular. Intel and AMD's 32- and 64-bit architectures are a significant presence in the embedded space and dominate in other areas like the cloud and desktop spaces. The number for RISC-V is one to keep an eye on.

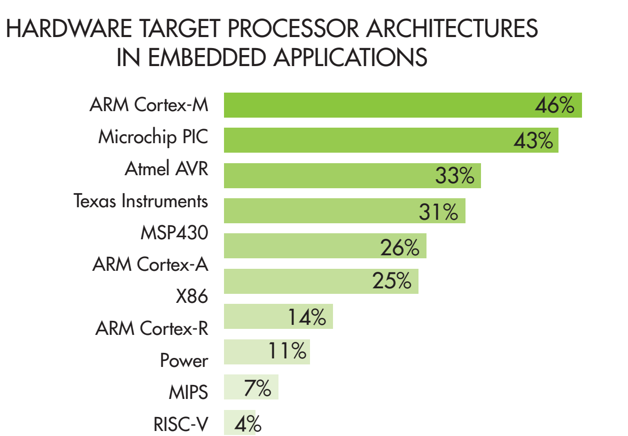
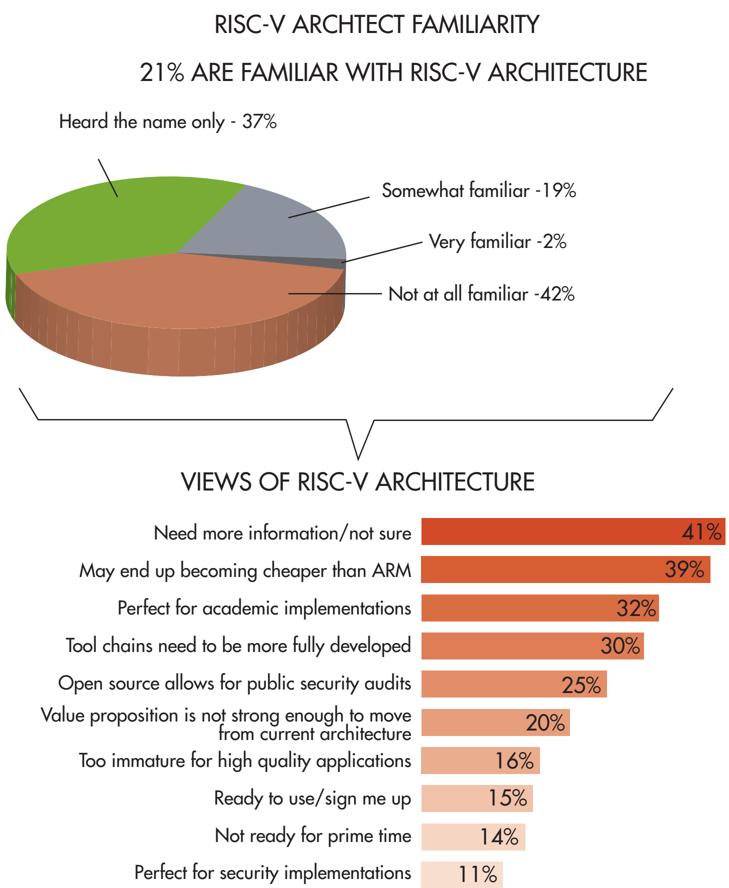
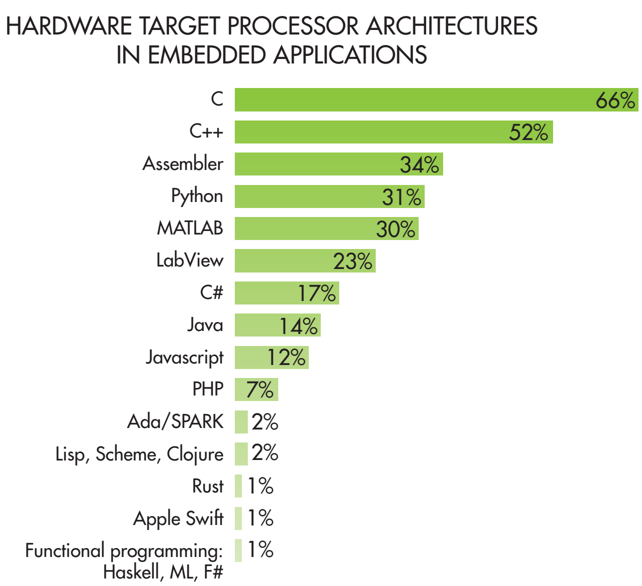


Fig 9: ARM has done a good job in getting their architecture into embedded designs but they have not displaced may of the popular platforms like those from Microchip, Atmel, Texas Instruments and Intel.

RISC-V is an open-source architecture *(see "RISC-V (Five) Is Alive!" on electronicdesign.com)* that is growing in popularity *(Fig. 10)*. Only a few are really familiar with it and that is not surprising given the only publicly available off-the-shelf incarnation has just been released *(see "First Open-Source RISC-V Chip Arrives" on electronicdesign.com)*. It is SiFive's Freedom E310 and its 32-bit microprocessor that is also found on the HiFive1 Arduino-compatible board. Prior to this, FPGAs or



10. RISC-V is an open source architecture that is growing in popularity.



11. C and C++ continue to dominate the embedded space, but the importance of assembler has not been diminished. Other languages are on the rise as IoT becomes more of an influence.

custom chip design was needed to take advantage of this architecture.

So what makes RISC-V interesting? First the instruction set and architecture is open-source but it does not specify an implementation. This can be proprietary, and SiFive is one vendor that will supply a custom design. The other is that the architecture scales from 16 to 32 to 64 to 128 bits. Likewise, features like floating point can be added in an incremental fashion and the tools, such as compilers and linkers, can take this into account.

The open-source Rocket chip generator can be used with the RISC-V chip specification to generate a custom core design. This can be used as is or further optimized. It needs to be combined with peripherals for a complete chip design but this significantly reduces the complexity of creating a custom chip.

EVERYONE NEEDS DEV TOOLS

C and C++ continue to dominate the embedded space but the importance of assembler has not been diminished *(Fig. 11)*. This is not to say the developers are churning out loads of assembler and an equal amount of C/C++, but rather getting down and dirty with the processor remains a requirement. This may be for developing boot code or device drivers where more intimate knowledge of the system is required to optimizing high value code that may require better performance or more power efficiency.

Other languages are on the rise as IoT becomes more of an influence too. Languages like Python, C#, Java, Javascript, and PHP provide a more insulated environment and can provide performance comparable to C or C++ depending upon whether the bulk of the execution is in optimized runtimes or in algorithms. C, C++, and even assembler tend to win out in performance when implementing code but an application that spends its time in a library routine like a matrix multiply or FFT will be more dependent on that implementation than the language used to call it.

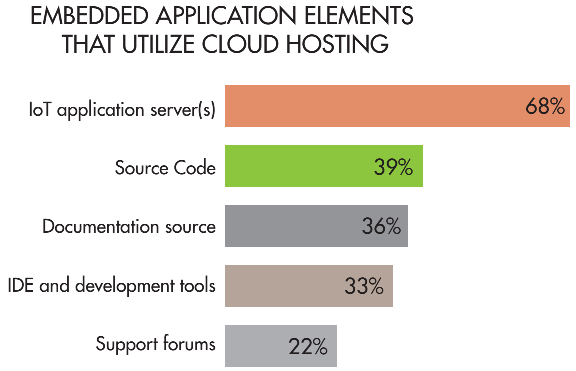
These other programming languages offer other features that can improve developer output and code quality including features like garbage collection. Using the same language throughout an IoT environment has benefits as well. Javascript and Python are interesting options for this.

The Mathworks MATLAB and National Instruments' LabView stand out as well. These are not only used for generating algorithms but they can also be used to create FPGA code and even be used for custom chip designs.

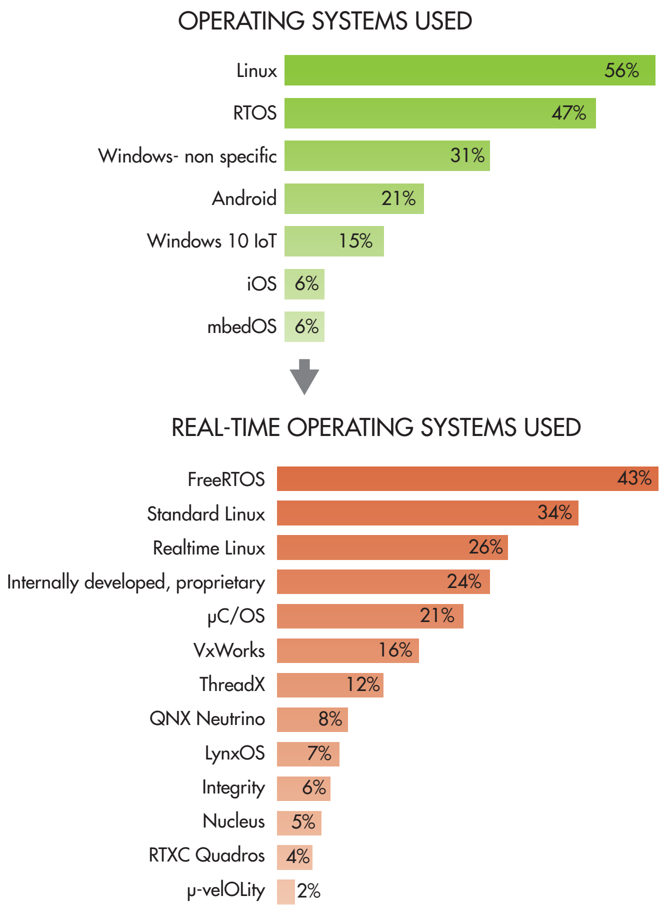
Where development works is occurring is changing as well *(Fig. 12)*. The cloud has come into play for everything from source code management to web-based IDEs. Hosting tools and development on the cloud is a small fraction at this point but growing.

Many chip vendors have started to provide development kits that work with web-based IDEs. This allows a quick, out-of-the-box start to development but it also provides a way for vendors to track usage as well as provide better support at a lower cost.

Still, the bulk of development is being done in-house on PC-based



12. Hosting tools and development on the cloud is a small fraction at this point, but growing.



13. Linux is not the only embedded OS being used, but it is clearly dominant where applicable. Proprietary RTOS implementations are significant but many are turning to third-party RTOS implementations.

platforms. This provides more control and sometimes faster response time. It also allows more custom integration with a developer's other tools and their company's infrastructure. Some also have security concerns that are not always addressed sufficiently with cloud-based solutions.

Operating systems remain a central component especially as development has moved to 32- and 64-bit platforms where Linux has become the de facto standard *(Fig. 13)*. Real-time and standard Linux implementations are being used where real-time operating systems (RTOS) are needed but they are not alone. FreeRTOS is being used in a significant number of projects. We did not break out whether these open-source solutions are being supported internally or through a third party. I suspect that many may start with the former and move quickly to the latter as the developers discover the amount of work required to support a platform.

"Linux continues to be the most popular embedded operating system for good reason," says Linux Foundation Executive Director *JIM ZEMLIN*. "Open source operating systems such as Linux and Linux distributions, including Android and Real-Time Linux, provide embedded developers with flexibility and adaptability to ensure the functionality, performance and security they need."

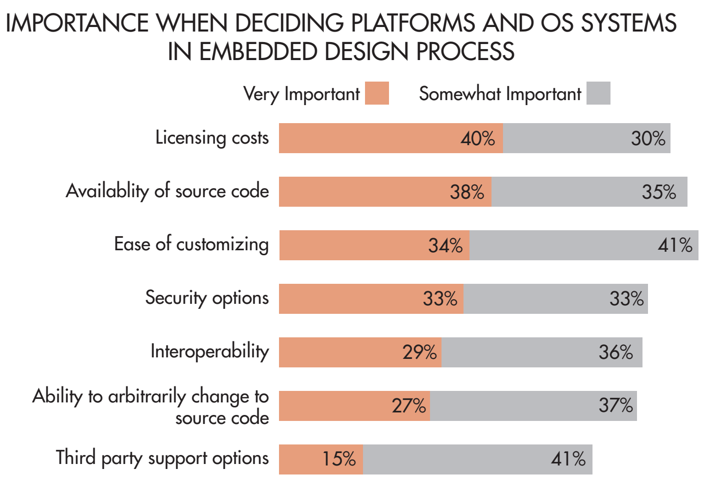
"The Yocto Project has become the de facto standard for those who need to build custom embedded Linux distributions for any architecture, supporting SOCs, development boards, and vertical solutions," says *JEFFREY OSIER-MIXON*, Yocto Project Community Manager.

Internally developed RTOS solutions sit near the top of the stack. We will need to keep an eye on this to see if there is a migration other alternatives. Internal support can be expensive and that is one thing a third-party solution offers. The third-party RTOS options are well represented in our survey. Most offer features or support that developers may need that are often not available on other platforms.

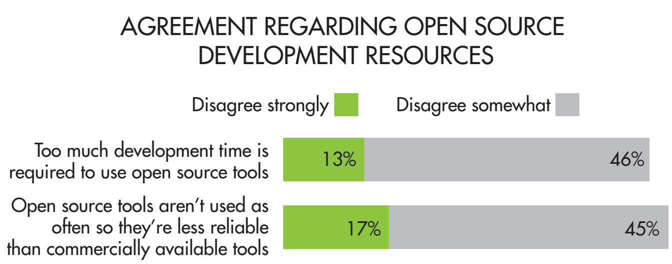
MORE SOFTWARE ISSUES

There are no unimportant attributes when it comes to software selection but availability of source code for customization and licensing costs rank high on the list *(Fig. 14)*. These all relate to the cost of creating, deploying and maintaining an application in the field. Licensing costs can have a significant impact on the profitability of a product and licensing has moved from simply that of a compiler and possibly an operating system to IoT middleware, cloud services, and much more. Per-device costs may be one time or a subscription model that requires additional costs to support. There is rarely a free lunch even with free software.

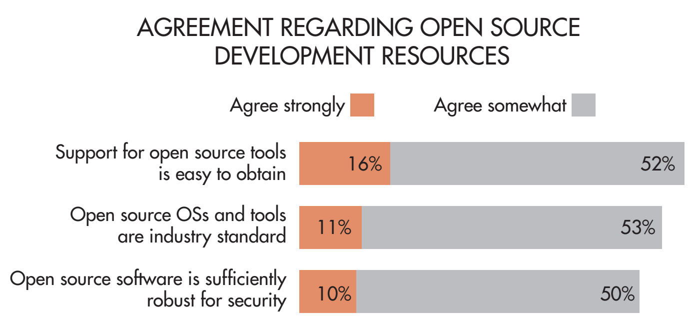
As noted earlier, security is now one of the top items on a developer's



14. There are no unimportant attributes when it comes to software selection, but availability of source code for customization and licensing costs rank high on the list.



15. Some say that open-source software and hardware is less reliable or takes too much time, but not our respondents.



16. Open-source tools appear to meet the needs for a significant number of embedded developers.

check list. Interoperability is also becoming more of a challenge as higher-level protocols and interfaces are used. Connecting to the network is one thing, the cloud another, and to an IoT service yet another. These issues were less common before the IoT wave, but are now ubiquitous for many product categories.

The use of open-source software and open-source hardware continues to increase. There are many that swear by it and many that swear at it. Some say that open-source software and hardware is less reliable or takes too much time, but not our respondents *(Fig. 15)*. Of course, your mileage may vary and every open-source project is different. Some, like Linux and Apache, have massive support and an equally massive community. Some, like MySQL have captive sponsors, in this case Oracle. Other projects languish. They may never die but unless you plan on bringing support in-house, they are usually not good candidates for embedded products.

Open-source tools appear to meet the needs for a significant number of embedded developers *(Fig. 16)*. The strength of the agreement would likely change if we specified particular projects versus open-source in general. The GNU compiler collection (GCC) and Low Level Virtual Machine (LLVM) compilers are examples of open-source solutions that are readily available and often bundled by chip and system vendors with their toolsets. These are often integrated with open source integrated development environments (IDE) like Eclipse.

GENERATING BETTER CODE

Buggy code is bad. Buggy code in a connected device can be catastrophic. Using coding best practices can make a difference in code quality. There are a number of standards available and it is apparent that developers are making use of them.

What coding standard are you using? Many incorporate more than one into their development process.

MICHAEL BARR, **CTO and co-founder, Barr Group: "When people debate about coding rules, they too often focus on the wrong things: like where the curly brackets go. When you are designing products that need to be reliable and secure, it's a better practice to adopt a coding standard of rules chosen because they can be shown to keep bugs out. That's what we did when creating Barr Group's Embedded C Coding Standard and that's the same approach everyone should follow."**

In the meantime, check out the *fiercely independent* coverage of the embedded and IoT space on Electronic Design where we continue to delve into the trends and product categories presented here
(see "These Trends Will Shape Embedded Technology in 2017" on electronicdesign.com).