Engineers’ Guide to Embedded Linux & Android

How the Yocto Project Improves Embedded Development

Sweet New Android Versions Still Leave Room for Embedded Linux
Extending the Scope of Android Platforms via USB Connectivity
COMs Bring the Power of Embedded Linux in Small Package

www.eecatalog.com/embeddedlinux
Your reliable hardware platform supplier

9” TFT LCD with resistive touchscreen
Wide screen 1024x600 dpi
Fanless and low power consumption
Panel mounting IP65 rating
WiFi support (Optional)

PC/104 form factor single board computer
DM&P processor SX / DX / DX2
Fanless and low power consumption
RS-232 / 422 / 485 support
Customized BIOS support
5+ years long life cycle support
Legacy standard support

Reliable  Green  Compact  Fanless
We Make What You Need to Create What You Want

Agile, yet Stable.
Flexible, yet Solid.
Ubiquitous, yet Unique.
Open, yet Secure.

Maximum from Minimum.

To most, these are contradictions.
To you, these are requirements.
To us, these are a mission... to ensure you get the most out of every project.

With Over 50 million devices across all industries, all ways lead to MontaVista.

CG Linux
Cloud, SDN, NFV Solutions
Professional Services
Welcome to the Engineers’ Guide to Embedded Linux & Android 2014

Android and embedded Linux continue to find their way into new markets, moving from traditional smartphone applications to medical, automotive, mil/aero and M2M of every flavor. In our roundtable discussion, “Sweet New Android Versions Still Leave Room for Embedded Linux,” our experts address these new market opportunities as well as challenges still to be overcome. One of those challenges is certainly the complexity of developing embedded hardware on Linux or Android platforms, and the Yocto Project is one solution. Mentor Embedded’s John Cherry says, “The goal behind the Yocto Project is to provide open source software and high-quality tools to help embedded developers make their own custom Linux-based system—regardless of the underlying hardware architecture.” You can get the full scoop in his article, “How the Yocto Project Improves Embedded Development.”

As we move intrepidly into the era of everything-connected computing, interfaces become another challenge for embedded developers. Gordon Lunn of FTDI Chip explains how the Android Open Accessory Initiative supports USB—the interface that’s taking over the world—to control external hardware in Android platforms in “Extending the Scope of Android Platforms via USB Connectivity.”

Of course, with everything connected, security is a perennial hot topic, and we provide a couple of different views. Metaforic’s Andrew McLennan explains the stakes: a PC can expect no more than 40 to 200 minutes of freedom before an automated probe reaches it to determine whether it can be penetrated. According to McLennan in “Needed: Self-Protecting, Security-Aware Mobile Applications with Anti-Tamper Technology,” application providers need to step up and begin building in sufficient security for mobile devices. And SafeNet’s Michelle Nerlinger looks at another security angle, describing how stolen code can end up in the hands of competitors or be used to reproduce knockoff versions of a similar product in “Smart Software Monetization for Smart Devices.”

In other articles, Wind River’s Jens Wiegand explains “Smarter Ways to Embrace the Internet of Things,” calling data “the new currency of business.” Alexandru Voica and Saraj Mudigonda of Imagination Technologies Group describe the requirements for “Video and Voice Applications for Tomorrow’s Mobile World.” And our own Editor-in-Chief Chris Ciufio digs into one of his personal hot buttons: in-vehicle systems in “HTML5 Is What’s Needed To Rapidly Develop IVI Automotive Apps.” And as always, there a lot more online at www.eecatalog.com/embeddedlinux/. Enjoy!

Cheryl Berglund Coupé
Managing Editor, EECatalog.com

www.eecatalog.com/subscribe
FINALLY...

Easy Android™ Data Acquisition

Acquire analog and digital data with your Android device

New BTH-1208LS

- Bluetooth and USB connectivity
- Full Android™ driver support with examples
- 8 SE/4 DIFF analog inputs
- 1 kS/s continuous sampling, 50 kS/s burst I/O mode
- 2 analog outputs, 8 digital I/O, one 32-bit counter
- Rechargeable batteries, over 8 hours of continuous use
- OEM version available for easy integration

Collect data wirelessly with the BTH-1208LS DAQ device

Risk Free 30-Day OEM Evaluation Kit Available. Call for Details.

mccdaq.com/BTH1208

Contact us
1.800.234.4232
## CONTENTS

**Sweet New Android Versions Still Leave Room for Embedded Linux**  
*By Cheryl Coupe, Managing Editor..........................................................................................................................................6*

**How the Yocto Project Improves Embedded Development**  
*By John Cherry, Mentor Graphics Corporation....................................................................................................................... 11*

**Extending the Scope of Android Platforms via USB Connectivity**  
*By Gordon Lunn, FTDI Chip..................................................................................................................................................... 13*

**Needed: Self-Protecting, Security-Aware Mobile Applications with Anti-Tamper Technology**  
*By Andrew McLennan, Metaforic ......................................................................................................................................... 17*

**Smart Software Monetization for Smart Devices**  
*By Michelle Nerlinger, SafeNet ............................................................................................................................................. 19*

**Smarter Ways to Embrace the Internet of Things**  
*By Jens Wiegand, Wind River...............................................................................................................................................23*

**Video and Voice Applications for Tomorrow’s Mobile World**  
*By Alexandru Voica and Saraj Mudigonda, Imagination Technologies Group............................................................... 26*

**HTML5 Is What’s Needed To Rapidly Develop IVI Automotive Apps**  
*By Chris A. Ciufo, Senior Editor..............................................................................................................................................28*

**Computers-on-Module Bring the Power of Embedded Linux in an Incredibly Small Package**  
*By Dr. W. Gordon Kruberg and Andrew Simpson, Gumstix, Inc. ..................................................................................................................32*

## DATA ACQUISITION

**Data Acquisition**

**EMAC, Inc.**  
ARM System on Module .............................................................31
Attend the Largest Dedicated Android Conference in the Universe!

AnDevCon
SAN FRANCISCO
November 12-15, 2013

"AnDevCon is a great opportunity to take your Android skills to the next level, get exposed to technologies you haven’t touched yet, and to network with some of the best Android developers in the world."
—Joe Mitchell, Software Engineer, Quicken Loans

"AnDevCon has very good information and networking opportunities. It is pretty much the only choice to get info on Android without a Google spin, which leads to developers driving the technology and not what Google wants."
—Cory Bair, Mobile Software Engineer, j2 Global

Get the best real-world Android developer training anywhere!
• Choose from more than 75 classes and tutorials
• Network with speakers and other Android developers
• Check out more than 40 exhibiting companies

Register Early and Save at www.AnDevCon.com

AnDevCon™ is a trademark of BZ Media LLC. Android™ is a trademark of Google Inc. Google’s Android Robot is used under terms of the Creative Commons 3.0 Attribution License.

A BZ Media Event  Follow us: twitter.com/AnDevCon
Sweet New Android Versions Still Leave Room for Embedded Linux

As challenges such as security and multicore processing are addressed, Android and Embedded Linux find their way into new markets, moving from traditional smartphone applications to medical, automotive, mil/aero and M2M of every flavor.

By Cheryl Coupe, Managing Editor

EECatalog: Android’s biggest appeal in embedded is its combination of Linux underpinnings, huge commercial momentum in smartphones, and off-the-shelf GUI support. What new embedded markets is Android finding its way into?

Karthik Ranjan, ARM: Traditional embedded markets have historically been a mix of custom Linux solutions combined with Windows Embedded Compact (CE) operating systems. Android is finding its way into a range of both consumer and enterprise devices not only because of the aforementioned value and momentum but also because of the appealing royalty-free business model. This has enabled it to penetrate new markets such as set-top boxes and digital televisions such as those found in China—for example Geniatech and TCL. Android is also finding its way into the new emerging smartwatch category such as those from Italian-designed I’m Watch. However Android is not just limited to consumer electronics; it’s also finding its way into traditional enterprise strongholds such as the lucrative enterprise handheld terminal market. Once a market dominated by Windows Embedded, traditional vendors such as Honeywell and Motorola Solutions have all adopted Android-based solutions.

Willard Tu, ARM: As mentioned in the question, Android is getting more adoption in areas that leverage its off-the-shelf GUI support. Lots of embedded devices are considering or adding a graphical display. I can think of medical instrumentation, toys, scanning devices and digital signage—many of these are not high-volume apps that can warrant creation of a custom Linux solution, hence they try to pursue something that already exists such as Android, which in many ways fits the bill and is why it is getting a lot of consideration or adoption.

Chris Buerger, Wind River: Android continues to provide a flexible option across numerous embedded opportunities. Some of the most active areas continue to be aerospace and defense, automotive and medical device applications.

Android is steadily gaining traction in areas that require rich graphics and/or increasingly need multiple OSes on one device... imagine a device that must include an RTOS but also Android for graphics-heavy UI. For example, we’re also seeing the community that previously used Windows Embedded now looking for a change and investigating Android.

Keep in mind that open source is already well-adopted in the mission-critical and carrier-grade arenas of networking such as core and edge devices. Also, it is growing significantly in strict security-related applications and it’s evident by the significant adoption of Linux in military, aerospace and government.

There are mixed industry views of using open source in safety- or mission-critical applications. While Linux and Android are not designed for applications requiring higher levels of safety and are not formally certified, there is an ongoing effort to grow acceptance of open source in safety-related systems which could provide a sound basis for building safety systems in the future.

Warren Kurisu, Mentor Embedded Software Division: We are seeing a fast-growing demand for Android apps in the car as one example. Consumers want to repeat their smartphone experience when driving and this is causing an evolution of new embedded software architectures for in-vehicle use. One of the concerns car makers have is security, and the potential risk caused by introducing “untrusted” software apps into the vehicle. Mentor Graphics has embedded solutions that allow Android operating systems to run separately from core vehicle functions based on Linux or any other operating system.
Incidentally, some car makers have opted for “native” Android infotainment solutions, making use of the full Android software stack. Implementers will normally customize the UI layer, and restrict app support to trusted and approved in-vehicle applications. Android out-of-the-box is not well-suited for this type of in-vehicle infotainment (IVI) use.

**EECatalog**: On the flip side, Linux advantages include its performance, its community development, its massive amount of available code packages, the GPL and its ability to give vendors control over their destiny. For the embedded market, how do you compare Linux to Android, which some argue is too tightly controlled by Google?

**Ranjan, ARM**: For consumer electronics, proximity to Google is a clear concern and has prevented the growth of Android into CE devices in U.S. markets for devices like set-top boxes for example, where operators prefer embedded Linux over Android. This primarily resulted in the concerns about a potential conflict of interest with Google over products like Google TV in the most profitable U.S. pay-TV market. However, across the pond in Europe where pay-TV is somewhat less profitable, there have been fewer concerns about using Android for set-top boxes from operators like Swisscom, for example. Another major example of this is the adoption of Android by Amazon for its Kindle Fire product line, where Amazon has been more than happy to replace the default Google Play Store with Amazon’s own App Store.

**Tu, ARM**: As I mentioned earlier, some embedded devices just do not have the scalability to involve investment in creating a custom Linux distribution. These markets want to leverage what is existing and not re-invent the wheel. Off-the-shelf IP is highly desirable. The only real issue is not control but fit. If Android can meet the application needs, then it makes sense; otherwise, the developer is likely to look to open source to find something they can modify to meet their needs or commercially to acquire.

**Buerger, Wind River**: The use case and requirements for a project will determine when Linux or Android may be the most appropriate option. For example, Wind River Linux is a very well-established offering and has all the proven proof points and requirements for a commercial Linux distribution. Keep in mind that Android can be thought of as a type of Linux. One reason customers are taking a strong look at Android is that it is a solution stack with a tremendously rich ecosystem of applications that you don’t get with other Linux distributions worldwide.

As a commercialization partner, we don’t treat Linux or Android differently—we’re mainly interested in helping customers find a solution that best fits their needs. Wind River has deep expertise in open source, regardless of whether it’s embedded Linux, Android or others.

**Kurisu, Mentor**: Linux has over 100 variants, and is truly an open source solution with strong community support. A specific open source community project for vehicle infotainment called the GENIVI Alliance was established in 2009 to allow designers to leverage Linux core capabilities for in-vehicle infotainment (IVI) systems, and the organization now has around 170 members. The first GENIVI Linux infotainment platform goes into production later this year. A true open source platform such as Linux has lower risk for a car maker as compared to one which is managed by a commercial organization (for example Android and Google, Inc). No one is saying that Google will change the licensing model of Android, but the frequency of releases (Key Lime Pie is due next year) is entirely under their control, and car makers will be obliged to absorb any changes in license terms or technical content if they want to keep their designs in step with the Android releases.

**EECatalog**: From security issues to data-exchange protocols; low-power requirements and new types of CPUs/MCUs—what affect is the rise of M2M computing having on Linux?

**Tu, ARM**: M2M computing has many new requirements. Security is certainly one of them. How do you secure data in the mesh network, or from the mesh network to the gateway to the cloud? Data is becoming much more valuable and in the past, M2M solutions were based on proprietary networks which were closed loops. Now M2M wants to leverage the Internet. There will need to be development of M2M standards that will help provide security. Currently, I think businesses are still trying to figure out how to effectively create IoT and M2M business models such as software as a service. Security will come to the forefront, once people figure out how to make money.

**Kurisu, Mentor**: Linux is continuing to establish itself in new-found roles in leading-edge solutions, as implementers
look to upgrade from the traditionally more predictable RTOS-based solutions. As the scope of M2M grows, such as vehicle-to-vehicle automated communication, the demands on the computing platform also increase. Linux is a truly scalable, multi-tasking operating system that can take advantage of multicore high-performance SoC platforms and support an extensive array of peripheral devices that exist in a multi-machine environment. Implementers are being forced to consider security and safety issues more carefully; traditionally a potential weak spot in Linux systems.

EECatalog: Samsung just announced that some versions of Galaxy devices—likely tablets at first—can be equipped with Green Hills’ INTEGRITY Multivisor for Trusted Mobile Devices. In effect, this is an NSA-quality partitioned operating environment now baked into a commercial off-the-shelf device. Do you see this kind of architecture as becoming important on Android- or Linux-based embedded platforms?

Ranjan, ARM: ARM is very pleased that Samsung has chosen to adopt the use of ARM TrustZone to be part of the Samsung Knox solution. Combined with other security capabilities such as Hypervisor, TrustZone provides a strong secure execution environment for securing a variety of services on a mobile device including DRM, BYOD and payment solutions. Trustzone can equally be leveraged in the same way to provide solutions for DRM in STBs, secure payment for mobile point of sale (PoS) solutions, as well as enhancing security in traditional embedded enterprise applications such as handheld terminals, industrial automation, etc.

Buerger, Wind River: The concept of embedded virtualization and partitioning is gaining interest. Especially with the rising adoption of multicore chipsets, the use of Linux and other operating systems in a dual-architecture are ripe. In this scenario, it is important to match the characteristics of the OS to the right task. Furthermore, it also allows the reuse of legacy code while using newer operating systems for new development.

As open source continues to grow, especially in areas historically reluctant to use it (such as A&D, auto, industrial and other areas sensitive to safety-critical requirements), we’ll continue to see further experimentation of multiple OSes and partitioning in order to innovate faster while staying on budget and schedule. With multicore and hypervisor technologies, Linux can be used alongside an RTOS in an overall mission-critical/safety-related system. You can “contain” Linux in a non-safe partition and use an RTOS for safety applications, making the overall system safe.

Kurisu, Mentor: Software developers are looking for novel architectural solutions that allow trusted and untrusted domains to co-exist. Separation can be managed at a hardware level, or further up the stack using core separation, firewalls, Linux containers and hypervisors. Mentor Graphics believes it is important to have a range of options available, allowing the implementer to select the right option for each particular job. By consolidating different operating systems onto a single multicore SoC, manufacturing and total system component costs can be reduced.

EECatalog: What affect does multicore processing—including heterogeneous multiple cores—have on embedded Linux development?

Tu, ARM: Multicore processing brings a level of complexity to embedded design. The majority of embedded designs are MCU oriented, and the majority are probably single-core designs. There are multicore devices in MCU where you see a combination of Cortex CPUs. For example Texas Instruments has a family of TMS570 devices that might have dual Cortex-R4 working lock-step for safety applications. Additionally the TMS570 family can offer Cortex-R4 and a Cortex-M3 for other designs that want to leverage the Cortex-R4 for its computational power for algorithm processing, and the Cortex-M3 for its command and control for more interrupt handling. Similarly, NXP LPC4000 devices which pair a Cortex-M4 and a Cortex-M0, where the Cortex-M4 is better suited for some digital signal processing for audio or motor control applications, and having the Cortex-M0 offload the Cortex-M4 and handle other I/O-intensive operations. There are many more multicore processing designs that may use big LITTLE implementation. In these more application-centric markets, Linux is certainly a favorite. ARM big LITTLE, is about energy saving. It combines a high-performance CPU with a more energy-efficient CPU so that the application can decide seamlessly which processor to access to optimize user performance.

Buerger, Wind River: The trend and promise of multicore is finally catching real momentum and is a key consideration in all types of projects across the embedded industry. With the power and cost savings that come with consolidation, multicore technology provides companies with capabilities to meet their continued need for faster time-to-market and reduced material/development costs. Multicore will continue to be a factor in most new projects moving forward.
To recap, open source continues to gain traction in areas with safety- and mission-critical applications, such as in A&D, automotive, industrial and medical markets, especially as multicore technologies are maximally leveraged. With an increased adoption of multicore and hypervisor technologies for multiple OS scenarios, Linux can be used alongside an RTOS in mission-critical and safety-related systems through the use of partitioning technologies. Additionally, as the use of multicore and embedded virtualization grows, there is an increasing need for powerful multicore-conscious tools.

Finally, given the added complexity that comes with multicore (multiple OSes, connectivity, richer applications, graphics, etc.), companies will continue to require assistance from expert commercial vendors, such as Wind River. Not only can companies turn to Wind River for its unique blend of deep embedded, open source and vertical-industry expertise, they can rely on our global support and long-term maintenance. While multicore presents increased software complexity, resulting in an entirely new set of challenges, the potential that it offers is undeniable.

Kurisu, Mentor: We are seeing multicore processing in which Linux is one component of a heterogeneous multicore architecture. An example of this is Freescale’s Vybrid Controller Solutions.

In this scenario, a robust OS like Linux runs on the more powerful ARM core and the smaller footprint of an RTOS, such as Nucleus, runs on a core where memory, power and overall resources are more constrained. Obviously, having a common communication infrastructure that can be leveraged between OSes is critical. Mentor Graphics has specifically invested in Sourcery CodeBench and Sourcery Analyzer to help implementers take full advantage of multicore processing. Mentor offers a compatible toolchain and debug environment necessary to optimize the distribution of load between different cores, and also debug the performance of a multitasking system. Staying within a single development environment our debug and analysis tools help to maximize developer productivity and reduce cost.

Security will come to the forefront, once people figure out how to make money.

Cheryl Berglund Coupé is managing editor of EE-Catalog.com. Her articles have appeared in EE Times, Electronic Business, Microsoft Embedded Review and Windows Developer’s Journal and she has developed presentations for the Embedded Systems Conference and ICSPAT. She has held a variety of production, technical marketing and writing positions within technology companies and agencies in the Northwest.
You’ve achieved organizational mobility beyond your wildest dreams. Now, get the answers you need to own your mobile space at MobileCON 2013. Whether your focus is security, apps, privacy, M2M, MDM, networks, mobile platforms or devices, MobileCON delivers answers to your most serious mobile IT challenges. Plus, you’ll experience first-hand the latest innovations and opportunities in mobile.

Register today at MobileCON2013.com
How the Yocto Project Improves Embedded Development

By John Cherry, Mentor Graphics Corporation

While open source software enjoys a rather loyal following in the embedded community today, the complexity of developing embedded hardware platforms with Linux or Android is often underestimated. Embedded software developers have an array of open source features and functionality options from which to choose, along with the availability of a broad range of SoC processor architectures which have also increased in complexity. And then there’s integrating the software from a variety of disparate sources. It can be said that Android offers developers more functionality quickly, and within a consistent software development environment. But in many cases, Android is not always the best choice since it was intended for one specific application—mobile phones.

So if not Android, then what? Linux is the obvious choice, but there are so many flavors and distributions of Linux, how does a developer even begin to know which Linux to use?

The Yocto Project: A Multi-Vendor Collaboration

For products targeting other markets, a customizable, nimble and efficient operating system like embedded Linux is a good choice. The Yocto Project is a multi-vendor initiative that was started in the fall of 2010. The goal behind the Yocto Project is to provide open source software and high-quality tools to help embedded developers make their own custom Linux-based system—regardless of the underlying hardware architecture. With membership comprising software companies, semiconductor manufacturers, device vendors and embedded services providers, the Yocto Project has proven its value for custom embedded Linux development and is recognized as the most powerful Linux build system framework today. Essentially, the Yocto Project frameworks is designed to facilitate integration of disparate software elements into a coherent set of platform images capable of booting a specific board. This framework also facilitates integration of kernel patches and configuration changes. Working within the Yocto Project framework, it is not particularly difficult for a developer with good Unix/Linux skills to make the changes necessary to add a kernel patch or modify a kernel configuration.

Differentiated Product Development

Product manufacturers strive to produce differentiated products with compelling features for their customers. Product differentiation usually involves some element of custom hardware as well as the application software that provides the user experience. The challenge for development managers is to limit the cost of development for the non-differentiating elements (the Linux OS and its supporting software) while maintaining maximum investment in the core competency of the company’s products.
Unfortunately, two factors conspire to complicate this challenge:

- Building the customized Linux operating system image for a unique hardware design can be technically challenging and resource intensive.
- The company’s development staff may not have the depth of Linux and hardware expertise to cost effectively develop the Linux operating system and required supporting software.

Custom hardware usually requires specialized support from the operating system: either device drivers or custom application software, or more commonly, elements of both. Writing device drivers in any operating system is never trivial. It requires technical competency in the OS itself. Technology from the Yocto Project provides a framework for building custom embedded Linux systems, but it will not teach a developer how to write a Linux device driver. It is for this reason many developers prefer to work with a software vendor who has been declared Yocto Project Compatible.

To achieve Yocto Project Compatible status, the product must be registered and declared to meet the requirements of the Yocto Project Compatibility program, and approved by the Yocto Project Advisory Board members. This allows for an easy transition from a free and unsupported Yocto Project-based board support package (BSP) to a commercially supported BSP and Linux distribution. A developer could develop a BSP, package recipes or define a target image for the Yocto Project, and it would translate directly into any commercial distribution based on the Yocto Project.

**Comprehensive Solution for Embedded Development Success**

The Yocto Project consists of over 100 repositories representing a wide range of technologies, but only a small subset is packaged and released every six months—which undergoes some testing and has a reasonable chance of working in a limited environment on which it was targeted. There are many advantages to using a commercially supported project such as Mentor’s Embedded Linux, but the single largest advantage is that it packages and integrates the Yocto Project technology around a complete development IDE and a broad product family which, when used together, allows software developers the ability to work more rapidly, with less risk and associated costs.

Further, it integrates the runtime components necessary to perform rapid profiling and analysis of complex multicore embedded systems. Building multicore systems using either SMP or AMP architectures can be accomplished with tested and proven components integrated and supported from a single vendor on the customer’s choice of platform.

Success in today’s marketplace requires not only delivering a quality product with differentiated features, but delivering it quickly to capitalize on untapped opportunities that may have a limited time frame. Linux and open source software certainly have the potential to reduce time to market and associated development costs—and provide developers with a unique product. But the complexity of developing embedded hardware platforms can seem daunting at times.

The Yocto Project helps software developers build and get to market sooner with a customized Linux-based system. The Yocto Project serves as a collaboration of like-minded developers who just want to make Linux easier to use and as a repository of metadata for Linux development. It’s important to make the distinction that the Yocto Project is not a Linux distribution, but rather it provides the tools, means, and methodologies for developers to create their own custom-made Linux distribution.

**Figure 1: The Yocto Project is both a collaboration and repository to facilitate the development of custom built Linux-based systems. Mentor’s Embedded Linux is a Yocto Project Compatible Linux distribution along with its associated build system that plays a key role within the Yocto Project development ecosystem.**

---

**John Cherry** is senior engineering manager at Mentor Graphics and runs the Linux Runtime Services organization. John’s organization provides runtime services for a variety of both embedded Linux projects and Android projects for companies around the globe. As an advocate for open source software and the communities surrounding them, John is on the Advisory Board for the Yocto Project. In the past, John has also chaired a number of the OSDL and Linux Foundation initiatives, including the Mobile Linux Initiative, the Carrier Grade Linux Initiative, and the Data Center Initiative.
Extending the Scope of Android Platforms via USB Connectivity

Through the Android Open Accessory Initiative, USB interconnection can support the control of external hardware via Android platforms.

By Gordon Lunn, FTDI Chip

Connectivity is now a vital part of the way we lead our lives. The computing devices that we use in a multitude of daily tasks need to be able to interface with all manner of different peripheral items: keyboards, mice, storage drives, printers, cameras, etc. In the early days of personal computing, each peripheral had its own specific connection scheme—a DIN plug for the keyboard, a serial port for the mouse, a SCSI for drives and so on. Then with the emergence of USB, there was a standard connection scheme that could effectively serve all of them.

The USB specification defines a strict host-peripheral arrangement. It also allows multiple peripherals to connect to a given USB host through the use of one or more hubs. In all cases, though, the host PC ultimately controls each of the peripherals, as well as supplying them with the power to operate while they are connected. More recently, however, peripherals have been getting considerably smarter and the lines that demarcate them from their host are becoming blurred.

The underlying operating system utilised by such peripherals could be an “embedded” version of what is found on its desktop/laptop counterparts. As we will see, such modern embedded operating systems have the capability to provide a USB host port. In some cases, the USB host may simply be stripped down—referred to as an Embedded Host. These Embedded Hosts allow connection of a limited number of peripherals (for example a thumb drive for data logging, but not a keyboard or sound device). This peripheral list enables a common hardware connection to be used for specific, pre-defined applications.

In addition to the Embedded Host, one more type of USB host is present on some peripheral hardware. It’s a USB port that can flip between host or peripheral status on-the-fly—what the USB-Implementers Forum refers to as On-The-Go (OTG) functionality. While the full specification allows for a port to be a USB host or USB peripheral through this capability, in practice OTG ports actually lock down the functionality to being a host or client, based upon the execution of protocols, for chips that are designed for OTG support (note: OTG support is not mandatory). The advent of the now highly popular Android operating system has major implications for how engineers look at the whole host/peripheral arrangement.

Android: A Brief History

In 2007, Google first announced the introduction of the Android operating system. About a year later the earliest examples of Android handsets (with ARM processors running Version 2 of the operating system) started to enter the market. As the Android kernel has been based on Linux, it can offer support for both USB host and peripheral functions. The user interface element of the kernel dispensed with the traditional desktop interface mechanisms and utilised one that is inspired by the Java programming language. The flexibility of using Java and Java applications had many benefits, including the ability for easy updates and the addition of software capabilities, both of which have had a large and beneficial impact on computing in the mobile arena.

In the years that followed, the prevalence of Android has evolved considerably. It now has dominance in both the smartphone business (recently published figures from consumer analysts Kantar show that, in the first quarter of this year, Android accounted for 64.2% of all smartphone handset sales on average across key Western markets) and the tablet computer business (with IDC estimating that Android-based models racked up shipments of 27.8 million units in comparison to 19.5 million for iPad products over the course of the first quarter of this year).
Delivering vital information on hardware, software, tools, services and solutions

www.eecatalog.com

A network dedicated to the needs of engineers, developers, designers and engineering managers
For Advertising Opportunities in Print, Online, Email Newsletters, Inserts & Reprints, contact:

Clair Bright
+1 415-255-0390, ext 15
cbright@extensionmedia.com

A network dedicated to the needs of engineers, developers, designers and engineering managers

To apply for a FREE subscription visit:

www.eecatalog.com/subscribe
Connection of Peripherals to Android Platforms

Although the underlying Linux kernel on Android provides USB host support, it is important to note that most Android platforms are portable/mobile and therefore require battery power to operate. It is this that sets them apart from the conventional host/peripheral set up. In such cases, it is clearly advantageous to bypass the host and only include the peripheral function, as by doing so, the requirement for the Android platform’s USB port to provide power to the connected peripheral is eliminated. With this in mind, the Android Open Accessory Initiative was established by Google in May 2011.

In the medical sphere, it will enable data acquired from patients on their current physical condition (heart rate, blood glucose level, body temperature, blood pressure, etc.) to be transferred to a tablet PC for subsequent analysis.

Likewise within a sporting environment it will be possible to download workout data (calories burnt, distance run/ cycled/rowed, heart rate, etc.) from gym equipment to a smartphone. The acquired data could then be compared with data from earlier workout sessions.

For the household appliance market it could have notable benefits, as through it engineers could upload software upgrades to white goods, or conversely download diagnostic information during maintenance checks.

In home/building automation systems, it adds a new dimension to how interfacing with thermostats, sprinkler systems, security alarms and audio-visual systems is conducted so that more intuitive smartphone- or tablet-based applications with vibrant screens and touch capabilities can be used to configure the equipment rather than the clunky user interfaces currently found on these items.

In response to the Android Open Accessory Initiative, the semiconductor industry needs to be able to offer engineers a new breed of USB host ICs. These devices should fully support the initiative and facilitate the integration of USB links into various peripherals. They should make the integration of hardware and software for USB host and Android support easy, with minimal design effort, while providing fast data rates, low power consumption and compliant connectivity to standards. In addition, the ICs being offered should have the capacity to bridge USB ports to a wide variety of different interface types (such as GPIO, UART, PWM, I2C, SPI, etc.), so that designers have the flexibility to connect to the available IO in their systems. Finally, charging is a major benefit for USB connectivity, but with Android Open Accessory the charging responsibility is shifting to the peripheral/non-Android platform. This shift mandates that the host ICs are able to source the required power in the USB standard and that products are architected correctly to be an efficient power source.

In conclusion, through the Android Open Accessory Initiative, USB connectivity is maintained but with a shift of responsibilities that must be comprehended by the system designer. Moving forward USB will clearly continue to have a very important role to play in both the data transfer to/from the latest portable products and in providing a conduit for efficient battery charging. It shows that the “universal” connectivity of this long standing interface standard remains valid—well into its third decade.

Having spent a total of 8 years with FTDI, Gordon Lunn currently holds the position of global customer engineer support manager within the company. He graduated from Heriot Watt University with a BEng (Hons) in electrical & electronic engineering and has built up 15 years’ worth experience in the industry. Gordon used to work for Racal Defence and Indigo Vision in the digital design area.
Needed: Self-Protecting, Security-Aware Mobile Applications with Anti-Tamper Technology

Application providers need to step up and begin building in sufficient security for mobile devices, including vulnerability mitigation, re-evaluation of trust and incorporation of secure authentication channels.

By Andrew McLennan, Metaforic

During the last 20 years, malware has evolved from occasional "exploits" to a global multimillion-dollar criminal industry. We hear about viruses such as Flame and Stuxnet, which can infect whole country infrastructures with relative ease. It seems to be getting simpler for hackers and malware to breach private companies and government agencies alike. For example, for at least two years, Flame has been copying documents and recording audio, keystrokes, network traffic and taking screenshots from infected computers. And passing all the information to servers operated by its creators. If it's that easy to attack governments and infrastructures, how difficult do you think it is to hack a smartphone?

In network security, perimeter-based and scanning techniques are penetrated and circumvented with alarming regularity. This has resulted in the more widespread use of application layer security technologies, which are now considered to be a critical component for security engineers who have come to realize how important in-depth defense techniques are in the current threat landscape.

A PC currently can expect between 40 and 200 minutes of freedom before an automated probe reaches it to determine whether it can be penetrated. This just shows how little time one needs to be connected to the Internet—wireless or not—before it’s touched and potentially hacked. If you think that PCs aren’t very secure, the smartphone (with little to no security in the apps or on the phone itself) is even less so.

And, of course, the latest trend is custom malware for attacking smartphones.

Custom Malware Designed for Smartphones
Application providers need to step up and begin building in sufficient security for mobile devices, including vulnerability mitigation, re-evaluation of trust and incorporation of secure authentication channels.

The need for these techniques is magnified on mobile platforms and perhaps none more so than on Android. A recent study by AV-TEST showed that more than 75 percent of anti-malware solutions ignored at least one in every 10 of the main families of malware in the wild. Add to this that Android malware is increasing dramatically, quadrupling between 2011 and 2012, and it seems that failing to protect mobile applications in general, and Android applications in particular, might be inviting a disaster.

The open source nature of the Android platform means that there are a plethora of free, widely available and powerful tools. While these have legitimate uses, they also make it simple to reverse-engineer unprotected applications or even elements of the OS itself, in order to assess vulnerabilities and create attacks. Add to this the fact that there are a wide range of largely unpoliced Android marketplaces where practically any application can be uploaded, making it unsurprising that the security situation has been likened to the Wild West. Even Google’s own marketplace and its use of its 'Bouncer' malware detection system is far from infallible, as researchers recently showed.

Mobile Security Critical for Businesses
With the huge growth of smartphones and the applications that run on them, mobile security is becoming a critical
area for all businesses. The sheer volume of commercially sensitive, personal employee and other key data both stored in and transmitted via these devices, makes them an attractive target for hackers. They also are an obvious route for threats that seek to penetrate the back office to corrupt data, capture it, or maliciously alter software through mobile application attacks.

Unfortunately, to date, security in Android has been ineffective. Custom malware attacks on Android applications are increasing exponentially and theft of software, data and content is rising to match. Hackers create and input malware that can change the behavior of applications, substitute account numbers, modify amounts, initiate egregious transactions, capture PINs, passcodes and more. Applications running on remote devices, with unknown configurations, need to be able to defend themselves, their communication, and to clearly signal if they have been compromised.

A recent study by AV-TEST showed that more than 75 percent of anti-malware solutions ignored at least one in every 10 of the main families of malware in the wild.

Apple’s iOS is not an impervious walled garden that many would have you believe either. A number of malicious applications have been removed from the App Store and Russian malware was recently pulled after managing to pass through Apple’s normal protections around their market.

Approaches to Secure Mobile Devices

There are various means to secure mobile device transactions. Strong security for mobile devices offers a comprehensive portfolio of embedded security solutions; the most obvious being anti-tamper technology, to prevent code and data changes. Anti-tamper is the most significant development in information security since the advent of the firewall and is perhaps the most advanced item in the security professional’s toolkit. The principle behind anti-tamper is quite simple: rather than relying on the security of the environment (by making the assumption that firewalls and virus checkers are installed, correctly configured and updated) anti-tamper ensures that the application can defend itself and its own data.

Clearly this approach will become the standard method for securing applications in the next few years as it is obvious that traditional approaches to security are now insufficient. ‘Defense-in-depth’ is now required for any applications that need to ensure the integrity of their operation.

There are numerous ways anti-tamper technology can help secure smartphone apps for financial transactions:

1. Protect the application itself against subversion. If it is possible to alter the application’s operation, any security methods inherent in it are open to trivial attack; data validation can be avoided, transactions can be altered or rerouted, data can be captured, and routines can be called at will to have previously unintended consequences.

2. Protect application data. In the same way as application code can be prevented from alteration, its data can be protected.

3. Protect data and keys within the application from capture or extraction by using cryptographic primitives, which prevent malware from being able to access the values of keys and other sensitive information by not holding them ‘in the clear’ in memory but instead by holding their values programmatically/algorithmically (e.g., to ensure bank account details are not captured and stolen).

4. Prevent ‘code lifting’ to extract individual functionalities (e.g., hackers might wish to use a code fragment that signs data with a key to sign some of their own data for a Man-In-The-Middle attack to reroute a payment transaction to a bogus account).

5. Trigger a response. Once an application is protected against subversion, any detection of an application level attack can trigger a response. While that may typically be as simple as alerting the user to a problem and exiting the application, anti-tamper technology typically allows custom responses; e.g.; sending a message to a server, perhaps to blacklist a device on which a compromise attempt has been made at the server-side.

6. Repair attacked applications or data. Should even one bit of an application or its data be altered and this be detected, the technology is available to repair the damage in order that the application may still be used.

As malware continues to attack smartphones, financial institutions must strive to provide the needed security to their applications. Malware won’t go away and companies need to be more proactive in securing apps from the inside out using anti-tamper technologies to produce that added level of security. We all know firewalls alone aren’t enough.

Andrew McLennan is an experienced entrepreneur who has founded five start-up companies since 1993, including Metaforic. Andrew has held all the key management roles in startups including CEO, CMO, CCO and COO. Andrew has an honors degree from Strathclyde University in mechanical engineering with aerodynamics.
Embedded software may come in a different package than traditional applications hosted on servers and installed on PCs, but that doesn't mean it isn't susceptible to many of the same challenges. The software that powers intelligent hardware found everywhere—even if it’s not as visible—still constitutes valuable intellectual property (IP). That means embedded developers shouldn’t shy away from deploying the same software protection and monetization techniques used for more traditional software. The goals are the same: preventing IP theft, and using advanced licensing systems to better monetize the hard work of the developer.

Intelligent device manufacturers are witnessing a change in where the value of their IP truly lives. The hardware they make is valuable increasingly because of the software embedded within it. So as hardware vendors become de-facto software vendors, cutting costs and maximizing profitability will involve some changes to the way they operate.

The four aspects of an effective software monetization strategy are: packaging, control, tracking, and ongoing management. Each aspect directly affects profitability by either helping reduce costs or increase revenue.

Protecting embedded software against product tampering and reverse engineering is just the beginning. Software monetization strategies for intelligent device manufacturers must take into account how these core elements are connected. Considerations such as profitability, user experience and usage control directly impact each other and should be approached comprehensively. When software monetization strategies are implemented successfully, the intelligent device manufacturer is able to offer both a more efficient experience for the user and a more profitable solution. It all begins with controlling IP.

Taking Control of IP Access and Usage
Controlling IP is the foundation of software monetization. Intelligent device manufacturers face problems with deliberate and unintentional misuse of their software, as well as product and feature overuse, competitive IP theft, product reverse engineering and code tampering—all problems that have plagued traditional software organizations for years. The key to controlling access and use of a device involves controlling who is granted access to the software running the device, when they’re granted access and to what extent.

Stolen code can end up in the hands of competitors or be used to reproduce knockoff versions of a similar product. That’s a big part of the reason why embedded software is a vendor’s most valuable asset. It not only holds all the development secrets hackers or competitors would love to
gain access to, it also determines how the product functions as well. A good use case illustrating the importance of IP protection is a software publisher who provides a compression algorithm that is nearly lossless. It is imperative that the algorithm not be deciphered because it is critical IP and unique in this company’s industry. The software publisher can leverage code-wrapping functionality of a software licensing solution to protect against reverse engineering and therefore protect their competitive IP from getting into the hands of pirates or the competition.

Aside from the possible theft of trade secrets, another major threat facing intelligent device manufacturers is tampering—manipulating the software embedded within a device to change how the device functions. This can provide users with access to features they have not paid for, or even worse, result in regulatory compliance problems. Without proper protection, intelligent device vendors are unknowingly leaving their code vulnerable to this risk. An example here is a company in the manufacturing industry that develops machines that create end-to-end packaging of consumer food products such as milk and orange juice. The software that runs these machines is programmed to comply with the dozens of public health and safety regulations. The company’s IP protection concerns center around controlling access to the software running the machines and the ability to tamper with key parameters that control processes such as pasteurization. This company used a software monetization solution to protect the software from being accessed and control who can change the parameters that control the machines.

Both stolen code and tampering have great potential to damage overall market share and therefore decrease revenue potential. By effectively controlling access to software source code, intelligent device vendors can protect revenue and safeguard the integrity of their brands and their products by preventing product tampering, reverse engineering and IP theft.

Usage control is the next piece of the monetization puzzle. Vendors must be able to control the use of their software at the product and feature level to prevent overuse of their offerings—deliberate or unintentional—and ensure that they are being fairly compensated. As the intelligent device market continues to mature, it will be critical for vendors to minimize manufacturing costs while achieving greater flexibility in their product packaging. This is accomplished through feature-based licensing. By providing customers the flexibility to license software features of intelligent devices already on premise, and by controlling access to that software, vendors can create new revenue opportunities.

Improving the Customer Experience

Vendors need more than just the means to control how their software is accessed and by whom. An effective monetization strategy will also provide them with a tool to help develop sophisticated packaging and pricing models. Along with preventing unauthorized access, these tools can also lay the groundwork to change how the intelligent device industry does business. Control over software at the feature level enables vendors to consolidate hardware stock-keeping units and provide remote upgrade and support services, in addition to opening the door to a whole new world of marketing and sales tools.

Historically, if a software vendor wanted to offer a premium and a standard version of a piece of equipment, they would build two applications for installation on two different hardware platforms. If a standard customer wanted to upgrade to a premium
They say the future doesn’t fit in a box.

Good thing Las Vegas is far from square.

Over four days, those who shape the future gather in a city built on reinvention. Here, brands, markets and economies converge in what’s far more than a tradeshow. And in 2014, there’s more opportunity than ever to connect with those who matter. The only question now, why wouldn’t you fit it in your future?

Register today at CESweb.org.

Tuesday, January 7 through Friday, January 10, 2014
Las Vegas, Nevada • CESweb.org • #CES2014
device, they would have to return their old device and wait for the vendor to ship them a new one. That is no longer the case.

By taking a feature-based approach to licensing and entitlement management, device manufacturers can develop and maintain a single, feature-rich application installed on a single device. The functionality of the device is then controlled through licensing. This enables software vendors to ship the same product with different functionality to different customers at varying price points and upgrade products remotely with lower support and fulfillment costs, thus delivering a better customer experience. A good example of this approach is a large networking company that used a sophisticated software licensing and entitlement management solution to protect its code and create smart, feature-based licensing packages for their enterprise customers that they could activate once physical equipment was already with the customer, cutting costs and enabling them to maximize the revenue generated by their IP.

Usage Tracking
The next critical element of software licensing is usage tracking. A sophisticated licensing and entitlement management system provides a means to start tracking product activation and usage right down to the feature level. Intelligent device vendors can use this information to drive decision-making around product packaging, roadmap investment, sales and marketing strategies. Product management and engineering teams can discontinue feature combinations that are unpopular and create software packages containing the most valued features that customers and prospects want.

Marketing and sales teams can utilize customized reports to better determine what, when and how products are being used and leverage this invaluable data to plan, launch and execute more effective sales and marketing activities. End-user registration data can also help vendors who sell via multiple channels to identify and gain direct access to every individual who uses one of their products.

Get Smart with a Software Monetization Strategy
Equipment manufacturers are evolving into software companies. Intelligent device manufacturers who embrace this transition and employ the right software monetization tactics will be poised to seize greater market share while reducing manufacturing and inventory costs. They will also be able to expand their product lines and bring innovative devices to market, all with the confidence that they are protected from any threats to their IP. The ongoing proliferation of intelligent devices means there are new embedded software development opportunities everywhere. For savvy hardware vendors, it’s a new frontier for them to conquer.

Michelle Nerlinger is vice president of marketing for SafeNet’s software monetization solutions.

Browse the EECatalog White Paper Library

MontaVista Linux Carrier Grade Edition

Real-Time Technology for Embedded Linux: The MontaVista Advantage

By William Weinberg

ABSTRACT
Communications networks are very different from other kinds of computing applications. Networks require both very high reliability and very high performance. Not only must they process large volumes of data at high speed, but they must do so with zero downtime for minutes per year. Network failures can have huge economic consequences and can even result in the loss of life. Additionally, in the post-9/11 world, failure also has a national security dimension. On top of all this, providers are under pressure to lower costs and become more efficient in their operations.

This technical whitepaper introduces MontaVista Linux Carrier Edition—a feature-rich, low-cost solution that provides a foundation for developing and deploying carrier-grade applications like those described above.
Smarter Ways to Embrace the Internet of Things

Business benefits have been constrained by the complexity of producing real-world applications, but this will change once operators and device manufacturers are freed to focus on their value-add.

By Jens Wiegand, Wind River

The Internet of Things (IoT) is transforming not only businesses, but also our lives. The ability of intelligent devices to perceive and respond to the environment around them makes them incredibly valuable for complex decision-making in a broad range of industries. The growth potential is explosive: billions of units are generating more than $1 trillion in revenue today, and according to market analyst IDC, the market for intelligent systems will reach nearly four billion units by 2015, representing more than $2 trillion in revenue. And many experts predict that there will be anywhere from 20 to 50 billion connected devices by 2020. In addition, the evolution of machine-to-machine (M2M) concepts into IoT concepts is greatly increasing and growing the market opportunity into billions of connected devices at work in a myriad of applications.

It is important to define some terms. M2M is a key technology for intelligent distributed systems using network resources to communicate with remote application infrastructure for the purposes of monitoring and control, either of the "machine" itself, or the surrounding environment.

IoT is where the physical world merges with the digital world and enables the new experience of interacting with this environment. IoT could be considered a more horizontal and meaningful approach where some vertical domains such as cars, smartphones, traffic control systems, as well as payment systems are pulled together to address larger business to business (B2B) needs as well as business to consumer (B2C) needs.

IoT concepts and architectures are driving significant innovations in network connectivity, mobile and wireless technologies, multicore processing, M2M communication, sensor technologies, cloud computing and data analytics. This has resulted in a convergence of a new form of intelligence with astonishing new capabilities to optimize the productivity of processes and efficiency of decision-making.

For example, smart-metering hubs can automatically report on usage via networks, saving the time and money to check meters manually and allowing companies to optimize consumption in response to supply conditions. Intelligent devices can provide heartbeat monitoring that gives doctors the data they need to determine diagnosis and treatment. Or they can send real-time traffic data to navigation equipment, helping to optimize traffic flow and reduce consumption and emissions.

Driving Factors
The momentum behind IoT architectures derives from macro-economic trends and other developments that impact specific industries or groups of adopters. These driving factors include high labor costs, as it typically costs at least three times as much for a human to perform a task—such as utility meter reading or smart building monitoring—as it does for a machine to do it.

Another is the real-time demand for “Big Data.” As data becomes the new currency of business, IoT architectures can supply both the raw material and sophisticated real-time analytics that shape and guide more intelligent business decisions. IoT architectures can also be both a ramp to the cloud and a means of exploiting the cloud’s potential, enabling businesses to develop new B2B and B2C services that create new efficiencies and economies.

A further factor is the ecological perspective: machines can perform power-management tasks with finer precision and faster response times than manual human-dependent systems, thereby saving energy, prioritizing usage and setting policies for response to outages, for example.

Challenges
IoT architectures can enable and accelerate many new service opportunities and also accelerate revenue generation, but...
there are significant challenges that impede scalability across vertical markets, including differing requirements of those involved in the industry.

The companies that are building the market for IoT plays have to address a series of questions. What is the best way to allow the wealth of new applications, systems and devices to connect to complex and often fragile networks? How can Big Data inform and guide the design of systems and devices for a better connectivity experience? How to deal with the data exchange between still stovepiped vertical markets, systems and applications? How can the operational efficiencies of IoT-enabled systems be scaled and create higher profit potential? And how can successes and lessons learned be leveraged more broadly across multiple vertical markets to compound the benefits?

Two of the key groups of solution providers for IoT concepts today are operators and device or system manufacturers. They have very different perspectives on the opportunities, but all of them are looking to develop solutions that will scale efficiently, increase average revenue per device and create competitive differentiation, while responding to the needs of specific vertical industries.

A major challenge is market fragmentation. The market is composed of many different vertical industries and their applications tend to have little overlap, making it difficult to scale solutions. There are also complexity and customization requirements, as the technologies involved in creating intelligent systems are extremely broad and complex, and most solutions do not provide a seamless end-to-end experience between the business backbone and the system or device domain, and thus must be customized to some degree. There is also a lack of specialized skills and expertise, as the skills required to build intelligent devices—in addition to the requisite market strategies—typically reside outside the core competency of operators and device manufacturers.

Slowly evolving standards in technology or application deployment is another challenge as the core components of IoT architectures have often been implemented in an ad-hoc fashion, using multiple competing standards in development and deployment. And finally, few operators or device manufacturers can create IoT-based solutions without significant assistance from partners; and typically these partners are not part of their current ecosystem.

**Ecosystem**

IoT will change well established ecosystems. Although it is still yet to be seen how the new ecosystem will build up over the next few years, we do know that IoT enables a wealth of new applications or services, i.e., Security-aaS, Platform-aaS, Infrastructure-as-S, Tools-aaS, etc. This will shift former device-centric monetization towards service- or software-centric monetization with significant new business opportunities. Former OEMs and even distributors will need to rethink their business strategies (see Figure 1).

The power of IoT comes though with the shift from former well-established but isolated, vertical ecosystems into a seamless horizontal approach, enabling seamless data exchange. Independent software vendors (ISVs) will need to become horizontal cross-domain innovators and enable IoT service owners with highly reliable services enabling IoT architectures, allowing for service-level agreements by spanning or aggregating multiple verticals. Seamless end-to-end

As data becomes the new currency of business, IoT architectures can supply both the raw material and sophisticated real-time analytics that shape and guide more intelligent business decisions.
are looking to serve the market as service providers, with bundled offerings for B2B and B2C customers, along with IT services and service management offerings, in order to establish new service-centric revenue streams.

A Smarter Approach to the Internet of Things Development

Technology providers like Wind River are facilitating these efforts by reducing complexity, aggregating supply chains through higher integrated software solutions and enabling rapid innovation and time-to-market for IoT-based solutions at lowered cost. Solutions like Wind River Intelligent Device Platform (see Figure 3) simplify development, integration and deployment of IoT gateways or M2M end devices, with a focus on delivering capabilities in four core categories:

1. **Connectivity**: Simplifying device connectivity for wireless and wired networks, speeding time-to-market and reducing expense for device manufacturers

2. **Manageability**: Delivering pre-integrated and supported management software—and collaborating with best-in-class hardware and software and system integration partners—making it much easier to manage remote connected devices and reduce total cost of ownership

3. **Security**: Providing tightly integrated, state-of-the-art security capabilities for protecting devices and their data, while at the same time allowing for an end-to-end protection strategy in close cooperation with open standard partners and Intel family members such as McAfee.

4. **Intelligence**: Enabling a seamless concept for data acquisition, aggregation and normalization of data allows for innovation on IoT architectures and enables IoT service owners to offer key differentiation in terms of new services and applications.

**Conclusion**

The market potential enabled through the Internet of Things is huge, but actual benefits achieved by businesses have been constrained by the complexity of producing real-world applications. This will change—rapidly—once operators and device manufacturers are freed to focus on their true value add: innovative new services and applications.

Jens Wiegand is vice president and general manager of strategic marketing at Wind River. A veteran in the industry, he brings over two decades of high-tech industry expertise in defense, automation and embedded computing sectors.
Video and Voice Applications for Tomorrow’s Mobile World

New standards provide a low-latency dedicated pipe for real-time voice and video applications that may lead to the development of a V.VoIP super-app.

By Alexandru Voica and Saraj Mudigonda, Imagination Technologies Group.

Today we know them as a feature phones but ten years ago a vast majority of mobile phones were used just for voice calling and messaging. The gradual roll-out of Internet services, the development of more powerful hardware and the evolution of software platforms coupled with increased network coverage and improved communication standards started a new mobile revolution. This has now grown to include content sharing, social networking or video calling—concepts that were not thought possible for handheld devices a decade ago.

The Evolution of Mobile Computing Platforms

Smartphones and tablets now have the hardware resources and required specifications (scalable, fast processors, video encoder/decoders and advanced camera sensors, large screens, flexible APIs) to make them suitable for video and voice over IP (V.VoIP) applications. But are these video and voice clients just like any other application that you find in most mobile stores? To address this question, we need to understand the initial purpose of mobile phones: enabling real-time communication for people on the go which relies on both the device itself and the network operator to maintain the call link.

Before iOS and Android became leading mobile operating systems, developers had few resources and tools for designing compelling applications. Most VVoIP features were embedded into the proprietary firmware with third-party software relying on Java ME or BREW. Because the hardware system was designed to deliver a limited set of functionality, any optimization was done at the target platform level and the number of devices that supported voice and video calling over the Internet Protocol was very limited.

Soon enough, smartphones became computing powerhouses with multicore processors and extra RAM and operating systems were able to run multiple applications at the same time including real-time software for VVoIP.

Development Options for Mobile Apps

When looking at mobile stores across various platforms, applications can be split into two major families: native and Web-based. There are a number of advantages of relying on native apps, as more and more companies realize that HTML5 may not be suitable to their needs (see http://techcrunch.com/2012/09/11/mark-zuckerberg-our-biggest-mistake-with-mobile-was-betting-too-much-on-html5/).

Embedded applications can be more deeply integrated in the overall experience, which provides users with a familiar set of characteristics (for example a unified phone dialer for voice and video). Pre-loaded applications are bundled software packages used by most manufacturers as a way to differentiate and get consumers a quick head start into the whole OS experience when they turn on their device for the first time. Downloadable apps offer a much wider choice as price points, popularity and user feedback determine different options and features.

Real-time V.VoIP applications have specific requirements such as low audio and video latency and a guaranteed quality of service (QoS) that set them apart from the rest of the crop. Network delays and packet losses were a common thing in the wireless environment but with HelloSoft’s smart concealment algorithm (see http://www.imgtec.com/hellosoft/hellosoft_ims_stack.asp), these issues can now be successfully mitigated.

Applications designed for V.VoIP solutions can be optimized for specific platforms and benefit from the various processing resources available which will save precious battery life. The rapid development of APIs and operating systems enables
these solutions to run across multiple platforms and devices such as smartphones, tablets, and ultrabooks while offering the same consistent experience across all of them.

The Main Requirements for V.VoIPApps
An integrated native application should not drain battery quickly and provide low latency HD voice and video experience even in a lossy wireless environment. This can only be achieved by tightly integrating with the handset platform and operators network, as is the case with embedded applications.

The developer community has now started to work more closely with operators and handset manufacturers to deliver performance-optimized apps while operators have begun deploying GSMA standards like 4G LTE (see http://www.gsmarena.com/ee_uk_lte_network_launchin_on_oct_30th_pricing_revealed-news-4986.php) which enable voice/video over LTE and rich communication services, including social presence, group chat, messaging, video/image and file sharing.

These standards provide a low-latency dedicated pipe for real-time voice and video applications to meet the QoS requirements. This means the handset manufacturers are opening up platform APIs for tighter integration, enabling a single integrated experience which would hopefully lead to the development of a VVoIP super-app.

Alexandru Voica is technical marketing executive and Saraj Mudigonda is business development manager for Imagination Technologies Group plc (LSE:IMG; www.imgtec.com). Contact Mr. Voica at Alexandru.Voica@imgtec.com and Mr. Mudigonda at Saraj.Mudigonda@imgtec.com
HTML5 Is What’s Needed To Rapidly Develop IVI Automotive Apps

Is HTML5 the right answer for the rabid consumer’s taste for car tech, while still giving the auto manufacturer the safety and security they’re required to offer by law?

By Chris A. Ciufo, Editor-in-Chief

Car manufacturers know that in-car technology like navigation systems sells cars. The pace of the smartphone movement is impacting the painfully slow speed with which automotive manufacturers develop new cars and tech features. Consumers trade out their phones every 2 years, but a two-year-old car is still considered nearly “new” by Kelly Blue Book. So how can the auto OEMs satisfy consumers’ tastes for updated, red-hot in-vehicle infotainment (IVI) systems and add-on apps?

Automotive software supplier Elektrobit thinks HTML5 is the answer. Coincidentally, so does RIM’s QNX division, along with Intel. QNX supplies “CAR 2” software to several auto OEMs, and Intel is behind Tizen, an HTML5-based competitor to Android. While Samsung has endorsed Tizen for a handful of smartphones, Intel has publicly stated that Tizen is also targeting automotive IVI systems.

At a webinar hosted by Automotive World magazine, Elektrobit’s VP of Automotive Rainer Holve, argued that HTML5 is the perfect language in which to develop and deploy the fast-changing IVI HMI software. Most importantly, the car’s core “native” IVI functions should stay separate and subject to safety-critical coding practices.

By partitioning the IVI software in this manner, the two ecosystems are decoupled and can run on their own market- and OEM-driven schedules. This means that native IVI—like GPS navigation, audio, HVAC or OBDII diagnostic information like fuel consumption—can be developed slowly and methodically on the typical 2-5+ year automobile OEM cycle.

But the faster-moving, consumer smartphone-inspired IVI portion, and its fast-moving add-on apps ecosystem, can move very, very quickly. This allows consumers to refresh not only the apps, but allows the OEMs to upgrade the entire HMI experience every few years without having to replace the whole car.

While the OEMs would love for an HMI refresh to force the consumer to replace the car every two years, it’s not going to happen. HTML5 is a reasonable alternative and they know it. According to Elektrobit, Chrysler, GM and Jaguar/Land Rover (JLR) have already started projects with HTML5.
HTML5 is an “evolution and cleanup of previous HTML standards,” said Elektrobit’s Holve, and is composed of HTML+CSS+JavaScript, along with new features for A/V, 2D graphics canvas, a 3D API, support for hardware acceleration, and much more. HTML5 is based upon open standards and is supported by Web Hypertext Application Technology Working Group (WHATWG) and the World Wide Web Consortium (W3C). Independently, W3C is working on a standardized API for JavaScript, which makes the HTML5 value proposition even sweeter.

Besides decoupling the HMI software from the “core” HMI functions, HTML5 would allow third-party apps developers to swiftly write and deploy applications for IVI systems. Besides Internet connectivity itself, this is the one IVI feature that consumers demand: a choice of what apps to add whenever they so choose. And since every automobile OEM will have to certify an app for safe in-vehicle use with their particular system, HTML5 allows app developers to create one core app that can be easily modified for multiple manufacturers and their myriad (and differentiated) vehicle models. In short: HTML5 makes things easier for everyone, yet still allows a robust third-party market to flourish.

It’s important to note how this is both similar to, and differs from, the current IVI strategy of many OEMs that rely solely on the smartphone for Apps. Chevrolet, Peugeot, Renault, Toyota and others tether the smartphone to the IVI system and “mirror” the phone’s apps on the screen (see my blog on Mirroring at http://eecatalog.com/caciufu/2013/01/). This allows the wildly robust iOS and Android app ecosystems into the car (and soon RIM/Blackberry and Windows 8 Phone), but it comes at a price.

In this scenario, the auto OEM must certify every app individually for use in their vehicle to assure safety or that critical car systems can’t be hacked or compromised. Or, the OEM can allow all apps to run and hope for the best. One hopes a rogue app doesn’t access the CAN bus and apply the ABS or electric steering.

HTML5, on the other hand, gently forces developers to create apps destined for IVI systems, but adds only a slight burden on them to make minor changes for each manufacturer’s certification. In this way they’re not barred from the car indiscriminately, but can develop a business of IVI apps separate from their smartphone iOS, Android and other apps.

Will HTML5 be successful? Is it the right answer for the rabid consumer’s taste for car tech, while still giving the auto manufacturer the safety and security they’re required to offer by law? I was skeptical about Tizen until Samsung’s announcements at Mobile World Congress 2013. With Tizen pushing HTML5 for “openness,” it may just gain traction in automotive, too.

Watch this space. We’ll keep you updated.

Chris A. Ciufo is editor-in-chief for embedded content at Extension Media, which includes the EE-Catalog print and digital publications and website, Embedded Intel® Solutions, and other related blogs and embedded channels. He has 29 years of embedded technology experience, and has degrees in electrical engineering, and in materials science, emphasizing solid state physics. He can be reached at cciufo@extensionmedia.com.
COME TO RTECC
REGISTER
IT’S COMPLIMENTARY!
AND MORE AWESOME THAN WORK!
RTECC.COM

TAKE A DAY TO LEARN ABOUT THE NEWEST IDEAS IN THE EMBEDDED INDUSTRY.
CHECK OUT THE LATEST DEMOS.
LISTEN TO TALKS FROM THE EXPERTS.
GET OUT OF YOUR OFFICE.
RETURN WITH INSIGHT ABOUT THE FUTURE OF THE INDUSTRY.

RTECC
REAL-TIME AND EMBEDDED COMPUTING CONFERENCE
WWW.RTECC.COM
ARM System on Module

Compatible Operating Systems: Linux
Supported Architectures: ARM

Made in the USA; the SoM-9X25 is based on the Atmel AT91SAM9X25 processor. This ARM9 core processor provides Dual Ethernet, 6 Serial ports and Auto RS-485 capability. The SoM-9X25 utilizes up to 4GB eMMC Flash, up to 16MB of serial data flash, up to 128MB of DDR2 RAM, and additional Flash provision is provided by a SD/MMC Flash Card Interface. All of the ARM processor core is included on this tiny module including: flash, memory, serial ports, Ethernet, SPI, I2C, I2S audio, CAN 2.0B, SDIO, PWMs, timer/counters, A/D, digital I/O lines, clock/calendar, and more.

The SoM-9X25 is designed to plug into a carrier board that contains the connectors and any custom I/O required for the application. This approach allows the customer to EMAC to design a custom carrier board that meets the customer’s I/O, dimensional, and connector requirements without having to worry about the processor, memory, and standard I/O functionality. With the SoM approach, a semi-custom hardware platform can be developed in as little as a month.

Customers have the option of developing a custom carrier board or one can be purchased off-the-shelf from EMAC. EMAC provides off-the-shelf Carrier boards that feature A/D, D/A, MMC/SD card, keypad, LCD, audio, and modem interfaces. The recommended off-the-shelf carrier board for the SoM-9X25 is the SoM-150ES which allows the user to immediately start coding their application using the Linux operating system and tools. Qty 1 price starts at $180.

For more information on the SoM-9X25 go to our website:
http://www.emacinc.com/som/som9x25.htm

FEATURES & BENEFITS

◆ Small, 144 pin SODIMM form factor (2.66” x 1.5”)
◆ Typical power requirement less than 1 Watt
◆ Battery backed Real Time Clock
◆ Additional Flash Provision Provided by SD/MMC Flash Card Interface
◆ Industrial Temperature Range (-40 to 85C)

TECHNICAL SPECS

◆ Atmel AT91SAM9X25 400MHz Processor
◆ 6 Serial ports, 3 with handshake, 1 High Speed USB 2.0 Host Port, 1 Full Speed USB 2.0 Host Port, 1 High Speed USB 2.0 Device Port
◆ 2 SPI & 2 I2C ports, CAN 2.0B Controller, 1 I2S Audio port. Timer/Counters & PWM Ports
◆ 10/100BaseT Ethernet with on-board PHY (2nd Ethernet Optional)
◆ Up to 128 MB of DDR2 RAM, Up to 4 GB of eMMc Flash, Up to 16 MB of Serial Data Flash
◆ 4 Channel, 10-bit Analog-to-Digital converter

AVAILABILITY

Now

NOW APPLICATION AREAS

Web/Network, Data Acquisition and Control.
Computers-on-Module Bring the Power of Embedded Linux in an Incredibly Small Package

With the flexibility to be morphed into virtually any application, computers-on-module are a technology that every developer who works with embedded Linux should seriously consider.

By Dr. W. Gordon Kruberg and Andrew Simpson, Gumstix, Inc.

Linux has, without a doubt, taken the world of embedded systems by storm. From satellites to smartphones, Linux can and has powered at least one of each, thanks to its flexibility, stability and performance. Even with its flexibility, however, the ability to run Linux depends on hardware requirements that can substantially increase the complexity of an embedded design (e.g., an ARM processor, which is naturally more complicated than a microcontroller). This is almost always a favorable trade-off as embedded solutions begin to rely increasingly on more sophisticated software combined with Internet connectivity.

Computers-on-module (COMs) solve the problem of keeping embedded designs small, flexible and relatively inexpensive, while allowing for a full implementation of Linux. COMs essentialize the most complicated features of electronic design—the processor, RAM, flash memory and wireless networking—onto a single circuit board with standardized connectors. These can then be used with expansion boards to break out the functionality of the computer into whatever a design requires, such as Ethernet, DVI output, USB connectivity and even GPIO pins. As long as the connector remains the same, COMs and expansion boards can be swapped with relative ease. This allows users to do things like use the same processor (on the COM) with a new revision of an expansion board, perhaps with an added feature; or conversely, to upgrade the processor (by using a different COM) that powers a highly specialized expansion board. This has the added advantage of allowing embedded Linux, deployed on the COM, to reuse the same software solutions with minimal code updates.

Finally, COMs have the advantage of being reliable, professional hardware designed for industrial and serious hobbyist applications. Just as the Linux kernel began as Linus Torvalds’ personal project in 1991 and grew to have the backing of industry giants like IBM, HP and Intel, COMs have grown from a hobbyist’s dream into a hardware platform used by organizations like the US Army (http://tinyurl.com/k5hczww), the International Space Station (http://tinyurl.com/n82x8la) and research institutions around the globe. COMs are particularly well suited to robotics applications, where, for quite some time, Intel x86-based netbooks were the standard platform for hobbyists and researchers. COMs are not only smaller than netbooks, their power requirements are so low that they can use the same power source as the robot itself, and with Linux-based robotics solutions like ROS, COMs have a lot to offer roboticists.

There are many small form-factor computers that can run Linux, (http://tinyurl.com/mmavno4) but none that come close to offering the performance, expandability and professional value delivered by the computer-on-module with expansion board model. COMs make things easier for hardware developers by reducing the investments in time and funding needed for an electronic design, as well as for software developers by providing access to a standard development environment and well-supported, rock-solid operating system in Linux. With the flexibility to be morphed into virtually any application (as evidenced by many hobbyist and commercial users creating so many successful and diverse projects), computers-on-module are a technology that every developer who works with embedded Linux should seriously consider.

Dr. W. Gordon Kruberg, president and CEO of Gumstix, founded the company in 2003. Prior to founding Gumstix, he was CEO of Deersoft, acquired by Network Associates in 2002. He holds an AB degree in human biology, an MS degree in industrial engineering from Stanford University, and an MD degree from Northwestern University.

Andrew Simpson is a content developer and writer at Gumstix. He holds a bachelor’s degree in English from the University of British Columbia.
WE DESIGNED THIS WEBSITE FOR YOU...

Complete Coverage of 30+ Key Embedded Technologies

Valuable Company and Product Information

Opinions and Insight

www.eecatalog.com
Leading Embedded Development Tools

ARM® DS-5™ Toolchain

A full featured development solution for all ARM Powered® platforms

1-800-348-8051
www.arm.com/ds5