WHITE PAPER

Mobile Handset Teardown:

Designing and Deploying with Mobile Virtualization

The Path to Building Cheaper Smartphones and Smarter Featurephones

October 21, 2009

Prepared by

Bill Weinberg, Linux Pundit

TABLE OF CONTENTS

Abstract

In the last decade, virtualization has evolved from esoteric mainframe technology to mainstream enabling software for enterprise servers and desktop computers. With the rise of ubiquitous mobile computing, on smartphones and netbook computers, virtualization is coming into play to enable an increasingly itinerant workforce and personal computing users on-the-go.

This white paper examines the economic impact of virtualization as a core component of mobile handset design. In particular, it offers readers a “teardown” of key device components (hardware and software) and analyzes how mobile virtualization reduces costs on the Bill of Materials. It also identifies incremental benefits from embedding virtualization in mobile devices, including performance and power management, as paths to both cost reduction and differentiation in a dynamic marketplace.

Mobile Handset Teardown: Designing and Deploying with Mobile Virtualization

Introduction

In the last decade, virtualization has evolved from esoteric mainframe technology to mainstream enabling software for enterprise servers and desktop computers. With the rise of ubiquitous mobile computing, on smartphones and netbook computers, virtualization is coming into play to enable an increasingly itinerant workforce and personal computing users on-the-go.

This white paper examines the economic impact of virtualization as a core component of mobile handset design. In particular, it offers readers a "tear-down" of key device components (hardware and software) and analyzes how mobile virtualization reduces and re-allocates costs on the Bill of Materials (BOM). It also identifies incremental benefits from embedding virtualization in mobile devices, including performance and power management, as paths to both cost reduction and differentiation in a dynamic marketplace.

In illustrating the advantages of mobile virtualization, this white paper makes frequent reference to actual shipping handset designs, and to the Motorola QA4 Messaging Phone in particular. This document, however, is not strictly speaking a literal teardown of that device, the details of which are not fully disclosable.

Figure 1. – Motorola Evoke QA4

Who Should Read This White Paper?

This document is intended to help technical decision makers assess the promise and the reality of mobile virtualization. In particular, this white paper strives to assist:

- Wireless device OEMs and mobile operators
- Developers and Development Team Managers
- CTOs, VPs, and Directors of Engineering
- Product Management Teams

Mobile Phone Design Influences

The arguments presented in this white paper are best appreciated in context. That is, to understand *why* mobile virtualization favorably impacts the cost and functionality of wireless handsets, it is important to understand *how* those devices are developed and deployed.

Mobile market analysts, ecosystem participants, and increasingly sophisticated end-users segment the mobile handset market into three tiers: smartphones, featurephones, and Entry-Level Handsets. To this three-way paradigm let us add the emerging categories of MIDs (Mobile Internet Devices) and Netbooks.

The emphasis of this white paper is using virtualization to build smartphones at featurephone price points, and conversely building smarter featurephones.

Ecosystem Coupling

Traditional product definition and design cycles for mobile handsets entail a close coupling among mobile/wireless ecosystem participants:

- **Mobile Network Operators (MNOs)** – specify wireless network interface (baseband) and myriad other requirements and design details to handset OEMs
- **Original Equipment Manufacturers (OEMs)** – integrate operator specifications, market requirements, chipset information, third-party software, etc. to design and manufacture actual handsets
- **Semiconductor Suppliers** – provide CPU, displays, and other strategic components to OEMs, as well as design assistance and technology roadmap information
- **OS Vendors** (OSVs) – provide platform, SDK, and features to OEMs (or are internal to a OEM, as with Apple, Nokia, and Palm). Increasing OSVs also support and nourish application developer communities

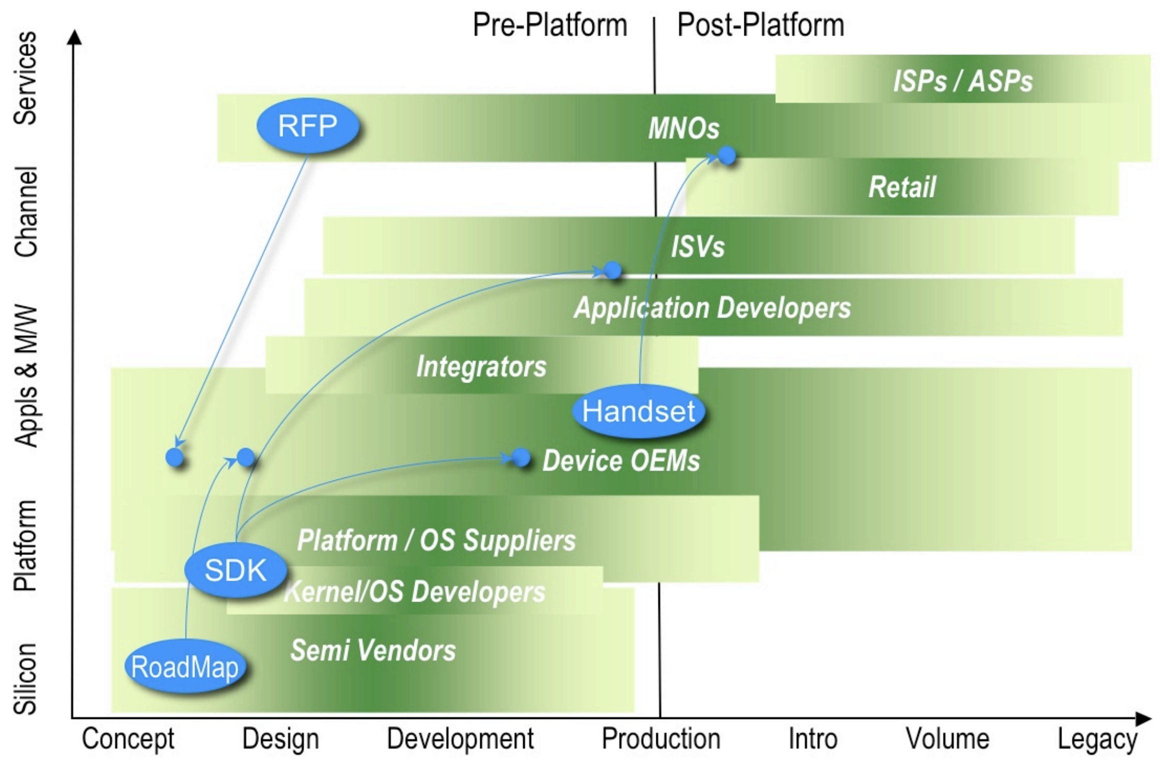

Figure 2. – Participants and Transactions in the Mobile/Wireless Ecosystem

Why Mobile Virtualization?

Until very recently, putting together the terms *mobile* and *virtualization* elicited looks of puzzlement and even dismay. Virtual machine hosting of entire operating systems (OSes) and software stacks at first blush seems out of line with lean-and-mean provisioning of mobile devices. However, with the convergence of mobile device and desktop computing capabilities (if not form factors and markets), virtualization is poised to become a mainstream mobile technology.

Many of the benefits conferred by mobile virtualization, then, do not arise in a vacuum. Rather, they are the result of the interconnectedness of the mobile/wireless ecosystem and its shared requirements. In that vein, mobile virtualization provides device OEMs and their partners a tool to facilitate and support

- Hardware consolidation
- Legacy migration
- Applications and services security
- Mobile-to-Enterprise (M2E) connectivity

To spark and grip the imagination of the mobile/wireless ecosystem, then, mobile virtualization must ultimately improve device margins and enable ARPU enhancement.

The Impact of Virtualization on Mobile Handset Design

Even in today's software-centric mobile device economy, it's the hardware Bill of Materials (BOM) that sets the price of handsets and their go-to-market position. The "smart" in smartphones was traditionally predicated upon BOM budget for high performance applications processors (APU), dedicated multimedia and baseband CPUs, ample DRAM and flash, and the budget to include a battery capable of powering these mobile behemoths.

Unfortunately, returns on design and deployment investments in smartphones have not always met market expectations. Deficits have included:

- Poor battery life in real-world usage
- Lackluster performance from aggressive power management or under-spec'ing of APUs
- Skimping on DRAM and Flash to save power and BOM costs
- Inter-chip connectivity bottlenecks (serial buses, etc.) and resulting performance deficits in multimedia, networking, and network access

Hardware Consolidation

While most attempts at remedying these issues focus on optimizing components in a multi-chipset legacy design, mobile virtualization re-invents the smartphone paradigm by:

- Consolidating multiple chipsets into a unified CPU, along with dedicated DRAM, glue logic, etc.
- Migrating applications, baseband, multimedia, and other ancillary processing from dedicated devices onto a single virtualized CPU
- Exchanging dedicated interconnects (serial buses, etc.) for shared memory interfaces
- Introducing mobile virtualization software (the hypervisor)

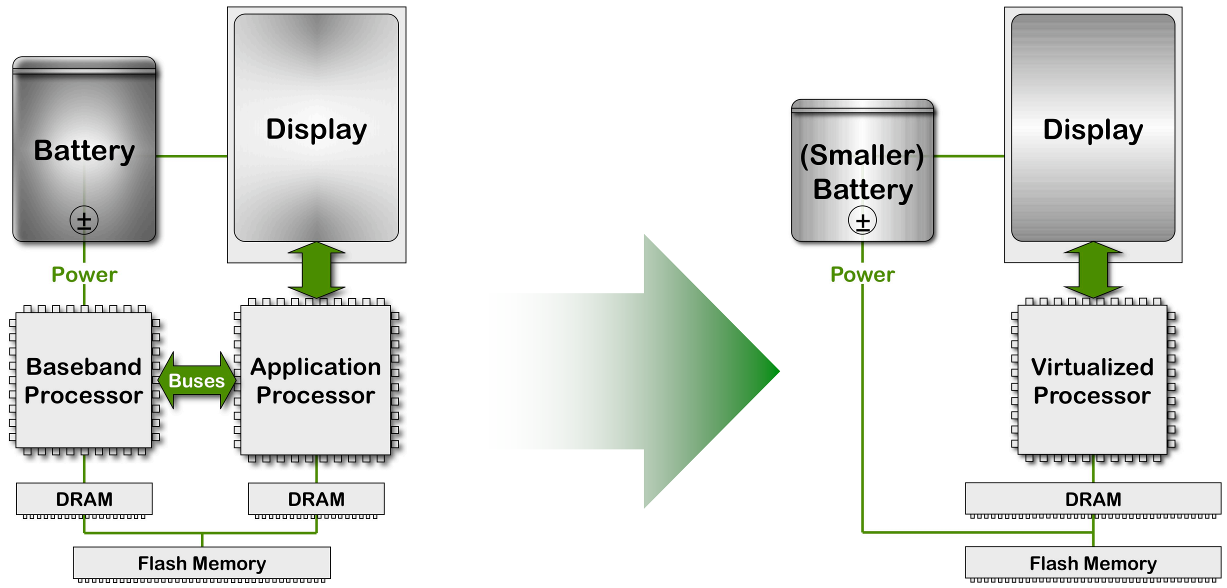

Figure 3. – Architecture Optimization with Mobile Virtualization

The introduction of virtualization into handset design most obviously institutes a revised architecture (one more reminiscent of featurephones), and has implications for the BOM and for many aspects of handset performance.

Virtualization and Handset Bill of Materials

Figure 3. highlights top-level architectural changes that impact handset BOM. Beyond a simplification of hardware blocks, it also implies how mobile virtualization can optimize costs by cutting chip count and consolidating resources:

- Single, consolidated CPU for application, baseband, multimedia, and other processing
- Consolidation of separate SDRAM dedicated to multiple CPUs
- Smaller battery to support fewer individual CPUs, fewer support components
- Preservation of existing preload software – application and baseband OSes, middleware, etc. – migrated from multiple physically isolated CPUs onto virtual processors on a single CPU
- Introduction of a Type I "bare metal" hypervisor to host legacy and new software on that CPU

These changes translate into actual line-item savings in the BOM:

	Cost Range	Legacy	Virtualized		Savings		Notes
			Nominal	Optimal	$$	%	
Total BoM		**$165.50**	**$149.50**	**$119.50**	**($46.00)**	**28%**	**Optimal Case**
Hardware							
Chipsets							
APU	$25-$35	$31.00	$31.00		**($31.00)**	**100%**	Migrate to APU
BPU	$10-$15	$13.00		**$13.00**			Migrate to BPU or equivalent
Storage							
SDRAM	$10-$15	$14.00	$14.00	**$10.00**	**($4.00)**	**29%**	Unify RAM from CPUs
Flash	$10-$20	$12.00	$12.00	**$12.00**			Flash shared among CPUs
Display/Touch screen	$35-$48	$45.00	$45.00	**$37.00**	**($8.00)**	**18%**	Display/OS trade-off
Camera	$10-$14	$11.00	$11.00	**$11.00**			
Accelerometer	$1-$1.30	$1.00	$1.00	**$1.00**			
GPS	$2-$3	$2.50	$2.50	**$2.50**			
Battery	$10-20	$17.00	$12.50	**$12.50**	**($4.50)**	**26%**	1100MAh → 850MAh
Other	$12-$15	$13.00	$11.00	**$11.00**	**($2.00)**	**15%**	Bluetooth, WiFi, USB, audio, logic & analog
Software							
Application OS	$0-$8						Linux, Android, etc.
Baseband OS	$0-$1.5						Nucleus, BREW, OSE
Other S/W	$5-$10	$6.00	$6.00	**$6.00**			M/W, CODECs, etc.
Hypervisor			$3.50	**$3.50**	**$3.50**	**-100%**	OKL4, etc.

Figure 4. – Strawman Bill of Materials for Legacy and Virtualized Smartphones

BOM Analysis

The BOM costs in Figure 4. represent a point-in-time market view. Costs are derived from multiple sources, including BOM data from actual devices and industry analyses [*iSuppli*, *inCode* et al.], for 2H2009. Figure 4. makes every effort to capture negotiated and volume pricing concessions and other factors, but may not reflect the precise experience of readers of this white paper. Rather, the above BOM represents a *strawman*, the purpose of which is to illustrate trends. Furthermore, Figure 4. begs explanation of its basis, methods, and conclusions:

Nominal and Optimal Approaches

The most obvious *nominal* path from a legacy multi-chip design to a virtualized phone would involve consolidation of application processing, baseband processing, and multimedia onto what is essentially the legacy Application Processor (APU). With the legacy APU as the locus of migration, savings would accrue primarily from storage consolidation, a smaller battery, and elimination of "glue" connecting the legacy APU and Baseband Processor (BPU). In our strawman BOM, these savings would amount to a modest 12 percent hardware cost savings.

A less obvious approach is to migrate to a virtualized equivalent of the BPU. This path is exactly the one taken by the Evoke QA4 team at Motorola – the Evoke is essentially a legacy BREW-based featurephone design that is able to run a "smart" Linux-based application OS thanks to mobile virtualization.

In such an *optimal* approach, OEMs can realize significant savings by cutting out the more expensive APU, and consolidating applications and baseband processing on a single cheaper ARM9 chipset. This approach enjoys the same savings in SDRAM, battery, and "glue" as the nominal approach. Moreover, real-world conversations with OEMs highlight the opportunity to reduce the cost of the display as well, moving from dedicated hardware UI support to software-based graphics processing (analogous to baseband migration).

On phones like the Evoke, mobile virtualization supports rich functionality in a modest BOM. For performance-intensive application OSes like Android, more costly CPU provisioning may be required.

Storage

The virtualized phone architecture confers no particular advantage to flash storage, given that multi-chip legacy designs often share a single flash device. In legacy designs with dedicated flash memory for APU, BPU, and other processors, migration to virtualization would certainly offer opportunity for further cost optimization.

Mobile virtualization most definitely supports cost reduction through consolidation of SDRAM from separate memories dedicated to APU, BPU, etc. into a single shared, larger but ultimately cheaper device – one larger SDRAM is usually cheaper than multiple smaller ones.

Battery

Elimination of physical components – processors, buffers, serial interfaces, etc. – can markedly reduce the energy profile of a virtualized handset, giving handset OEMs the choice of cutting BOM costs with less expensive lower-rated batteries or of giving the virtualized handset longer battery life (see also *Power Management* below).

Other Component Costs

Today's smartphones sport a desktop-like array of peripherals including interfaces for USB, WiFi, Bluetooth, and more phone-specific components like microphone and speaker, switches, and jacks. Mobile virtualization has little impact on these non-trivial BOM items, but would eliminate the abovementioned "glue" components associated with and connecting multiple legacy CPUs – serial buses, buffers, power management chips, and a range of discrete components[1].

Baseband and Application OSes and Stacks

Migration to mobile virtualization as described in this white paper assumes a mostly one-to-one relationship among legacy software components running separate CPUs and virtualized guest equivalents executing in virtual machines on a single CPU. For COTS application OSes like Linux, SymbianOS, Android, etc., virtualization suppliers like Open Kernel Labs supply pre-built paravirtualized versions of those OSes ready for integration on virtualized handsets[2].

This white paper also assumes parity of licensing costs: if a legacy application OS carried a run-time license in the legacy design, it would still do so in the context of mobile virtualization. Off-the-shelf open source Linux and Android are royalty-free, but derived OSes and mobile application stacks (e.g., from members of LiMo and of OHA) can carry project fees and per-unit charges that do impact the software BOM.

The situation for baseband OSes is rather more complex.

First, baseband OSes and stacks are not usually as open and free as today's open source application OSes. Examples include proprietary Nucleus OS from Mentor Graphics and OSE from Enea. These legacy Real-time OSes (RTOSes) and the baseband stacks they host can migrate to virtual machine hosting with little or no modification on a virtualized phone, but may require new software licenses, project fees, and/or new licensing terms in new virtualized designs.

Second, baseband OSes and stacks are usually closely tied to the BPUs they support. BREW[3], for example, is tightly coupled to Qualcomm wireless chipsets and is not usually licensed separately from those devices. Negotiating rights to re-host this type of baseband OS and stack can be more challenging, since suppliers usually regard baseband code as supporting intellectual property (IP) for cost-bearing silicon.

Third, not all BPUs share the ARM architecture with the APU. Some are based on Digital Signal Processors (DSPs) with unique architectures and instruction sets. DSP-based baseband code can certainly be retargeted for ARM execution – Motorola performed such a migration for their DSP Star OS and baseband stack in Linux-based ROKR and RAZR handsets.

The Embedded Hypervisor

The hypervisor at the core of mobile virtualization is a Type I "bare metal" virtual machine manager[4]. The same technology has been used for generations to support mainframe virtualization and is today mainstream in enterprise data centers. Type I hypervisors boot as primary system software, loading and hosting baseband and applications OSes and stacks as "guest" software.

---

[1] For chipsets that already physically consolidate APU and BPU in a single package, e.g., some Qualcomm devices, the savings of "glue" could be smaller.

[2] OK:Linux, OK:Android, OK:Symbian and others. Contact OK Labs for additional information.

[3] Originally with REX and today with OKL4 at its core

[4] Type II hypervisors run as applications over an OS, prevalent in desktop applications. Examples include VMware Workstation, VMware Fusion, Parallels, Sun Virtual Box and Microsoft VDI.

Device OEMs and also mobile OSVs can pursue several paths to acquiring a mobile virtualization platform. There are several notable examples[5] of OEMs and OSVs creating ad hoc partitioning to support application and baseband processing on a single CPU. These efforts historically proved to be resource-intensive, costly to maintain, with limited re-use. Another possible path lies in open source virtual machine technology like Xen and Linux KVM. While both these projects enjoy wide deployment and enthusiastic developer communities, neither exists in mature, commercial quality ARM-based[6] versions suitable for mobile deployment.

The shortest path to implementation of mobile virtualization lies with ISVs with mature COTS offerings. Open Kernel Labs (OK Labs) leads this segment with deployments in over 300 million handsets worldwide[7].

The Motorola Evoke QA4 employs the OKL4 Microvisor (embedded hypervisor) from OK Labs[8]. OKL4 is a commercial open-source offering and thereby can incur costs (and confer benefits) at development and deployment. The software BOM line item for OKL4 is admittedly a "swag" (based on informal conversations with OK Labs) and is necessarily not representative of negotiated deployment pricing. It is included to remind readers that *There Ain't No Such Thing as a Free Lunch*. Virtualization, like other BOM line items, carries some finite cost. Contact OK Labs for actual pricing and availability.

Technical Benefits of Mobile Virtualization

Readers of this white paper with a focus on the *meat* of mobile device economics will probably stop reading at this point – the rest is *gravy*. However, the technical impact of mobile virtualization has implications on a par with the financial ones documented in the previous sections, delivering advantages in development costs, time-to-market, security, device functionality, and performance.

Development Costs and Time-to-Market

Developing and deploying mobile handsets with virtualization represents a new path to deployment. While the mobile/wireless ecosystem is famously cautious and conservative, mobile virtualization offers OEMs and other mobile ecosystem members some very compelling advantages:

Streamlined Interconnects – Legacy multi-chip handset designs require dedicated interconnects (serial, USB, etc.) between the APU, BPU, and other components. These interconnects require interface-specific buffers/drivers and accompany device driver software, increasing design complexity, BOM costs, S/W and H/W testing requirements, source code management burden and limiting throughput. Mobile virtualization substitutes shared memory interfaces and secure IPCs, with enhanced performance and decreased development and test expenditures.

Software Integration – Legacy designs entail multiple physically separate and incompatible software platforms (APU, BPU, stacks, etc.). Changes and additions to each of these software environments are expensive, entail risk, and can invalidate hard-won certification and homologation efforts. Virtualization, by contrast, provides a built-in integration and software change mitigation platform. Legacy software can be enhanced and new software added to guest OSes, without perturbing existing tested software stacks, at the OEM factory or in-channel by integrators, MNOs, and other third parties.

[5] E.g., the Motorola RAZR and ROKR and stacks from Innopath.

[6] Current FOSS hypervisors for x86 architectures may prove interesting for Intel Atom-based mobile devices.

[7] MobileVision 2008.

[8] The Motorola Evoke QA4 actually employs the OKL4 Microvisor in two roles: as a mobile virtualization platform and as the kernel and partitioning manager underlying the BREW OS.

Smaller Device Footprint – Cutting handset chip count not only reduces BOM costs, it results in simpler, more reliable printed circuit boards, involves less physical material with lower environmental impact, and offers OEMs more packaging options.

Hardware Independence – Mobile virtualization facilitates new project starts and support code reuse with a small portable code base, more easily retargeted than entire guest OS kernels.

Higher Manufacturing Throughput – With fewer physical parts, virtualized handsets move through the manufacturing and provisioning process faster, accelerating time-to-volume

Testing and QA – Mobile virtualization, by reducing physical chip count and system complexity, helps limit testing time and costs by simplifying test jigs and moving testing activities to the software domain, facilitating in-channel testing as well.

Security, Device Functionality, and Performance

Mobile virtualization also enables and enhances a range of device capabilities:

Higher Reliability and Security – A virtualized mobile handset offers a more secure applications and services platform to MNOs, ISVs, and end users by having fewer items in the BOM (fewer points of failure) and by enabling removal of complex OS kernels from the trusted computing base (TCB) leaving only trusted code in the mobile hypervisor.

Multicore Support – While the strawman BOM in this white paper focuses on cost-down CPUs, virtualization also provides management for resource allocation and task dispatch on 2X multicore mobile processors and beyond. Better utilization of multiple parallel cores can yield superior voice and multimedia performance, especially in the absence of dedicated media and graphics co-processors.

Power Management – As our strawman BOM illustrates, mobile virtualization can enhance battery life simply by involving fewer physical energy-consuming components in a design. But mobile virtualization also provides new opportunities for unified power management of guest OSes and applications previously dispersed among multiple isolated CPUs. In its role of virtual machine manager, the mobile hypervisor is in an ideal position to monitor and manage activity levels of its guest environments, and also to manage usage and energy profiles of multi-core CPUs.

Telephony Performance – In legacy multi-chip designs, Telephony API (TAPI) commands voice and data flow over serial interconnects between the BPU and APU. Migration to virtualization-based shared memory interfaces can greatly improve throughput of telephony operations, despite losing silicon dedicated to processing voice and data[9].

Networking Capabilities – By unifying baseband and applications processing on a single CPU with a single (shared) TCP/IP stack, mobile networking can realize improved security (cleaner unified Network Address Translation routing and firewall), and better performance through multiple TCP/IP sessions. These capabilities can also bring tethering more transparently and easily to lower-cost devices.

Mix and Match Applications – the Motorola Evoke QA4 demonstrates how mobile virtualization supports seamless integration of legacy and next-generation applications. The Linux-based OS runs Linux applications but also can load and invoke legacy BREW applications in a separate virtual machine. Those BREW applications in turn utilize the Evoke's Linux-based GUI (X11/GTK) for display services while running in their otherwise "headless" BREW virtual machine.

[9] One OEM reports up to 10X improvement in telephony throughput.

Conclusion

Virtualization is already a mainstream business-critical and mission-central technology in the data center and for Cloud Computing. Its value is today extending across the desktop and making its way into emerging mobile applications as well. To date, arguments for adopting and deploying virtualization in mobile/wireless computing have focused on technical benefits, limiting the audience and the total impact of this exciting technology.

The goal of this white paper has been to bolster technical rationale for mobile virtualization with equally or perhaps more interesting financial benefits. The cost data and BOM impact presented here derive from real-world deployment in handsets that include the Motorola Evoke QA4, and have relevance to the global handset market at large. Together, these financial and technical arguments provide a compelling case for OEMs and other mobile/wireless ecosystem participants to evaluate the mobile virtualization technology for near-term development and deployment.

References

Heiser, Gernot [2009]. *The Motorola Evoke QA4 - A Case Study in Mobile Virtualization*. http://www.ok-labs.com/_assets/image_library/evoke.pdf

Klein, Gerwin; Elphinstone, Kevin; Heiser, Gernot et al. [2009]. *seL4: Formal Verification of an OS Kernel.* http://ertos.nicta.com.au/publications/papers/Klein_EHACDEEKNSTW_09.pdf

inCode [2009]. *Mobile Device Product Intelligence*. “2009 Device Price Breakdown”.

iSuppli [2009]. *iPhone Teardown*. http://www.isuppli.com/News/Pages/iPhone-3G-S-Carries-178-96-BOM-and-Manufacturing-Cost-iSuppli-Teardown-Reveals.aspx

VisionMobile [2008]. *The 100 million club: some surprising facts about mobile software.* http://www.100millionclub.com/

Weinberg, William [2008]. *Uniting Mobile Linux Application Platforms*.   A white paper for Purple Labs. http://www.linuxpundit.com/cv/docs/Platforms_WP_LP.pdf

Weinberg, William [2008]. *Moving Legacy Applications to Linux: RTOS Migration Revisited.* http://www.linuxpundit.com/cv/docs/RTOS_transition.pdf