Open Kernel Labs™

Be open.
Be safe.

TECHNOLOGY WHITE PAPER

The Motorola Evoke QA4
A Case Study in Mobile Virtualization

Gernot Heiser, PhD
Chief Technology Officer
Open Kernel Labs, Inc.

July 22, 2009

Authors:

Gernot Heiser, PhD
Chief Technology Officer
Open Kernel Labs, Inc.

Contact Details:

Open Kernel Labs, Inc.
200 South Wacker Drive
Floor 15
Chicago, IL 60606
USA

email: **info@ok-labs.com**
web: **http://www.ok-labs.com/**

1 Introduction

Virtualization for mobile devices has been talked about for a number of years, and most handset manufacturers are pursuing evaluation or development projects using virtualization. However, at the time of writing of this document, only one such mobile phone has reached the pockets of consumers: the Motorola Evoke QA4, which has been on sale since 1 April 2009. It is implemented using the OKL4 Microvisor product from Open Kernel Labs (OK Labs).

The Motorola Evoke is the world's first mobile phone that uses virtualization. It is implemented using OKL4.

In this white paper we will have a closer look at the Evoke. We will see how it demonstrates some of the power of virtualization. Moreover, we will see how the Evoke benefits from the unique aspects of the technology provided by OK Labs, which enables designs not supported by plain hypervisors.

Figure 1.1. The Motorola Evoke QA4 mobile phone handset.

2 Device Requirements

The constraints imposed on the design can be summarised as:

- in order to meet the targeted price point, the phone had to be a single-core design using an ARM9 core;
- the phone had to run Linux as the operating system (OS) supporting the user interface (UI);
- for a number of reasons, the baseband stack had to run outside Linux;
- components from the BREW [Qua] UI framework, such as the media rendering engines, had to be reused, and porting them to Linux was out of the question.

The requirements for the phone could only be met by a design based on virtualization.

Clearly, these requirements taken together can only be met using virtualization: Linux had to run (de-privileged) in a virtual machine (VM), with the baseband stack and BREW running in a different VM. And all this needed to be achieved with excellent performance in order to provide a good user experience.

However, the virtualization approach once faced much scepticism. I remember many discussions with actual or potential customers, who would doubt that an ARM9 core could provide enough grunt for a Linux-based UI, even when running Linux on a core of its own, let alone virtualized. The main reason is that the ARM9 has virtually-addressed caches and lacks an address-space ID (ASID) tag in the translation lookaside buffer (TLB). As a consequence, Linux (like most other OSes) on ARM9 flushes TLBs and caches on each context switch. This leads to poor performance on context-switch intensive workloads, such as a touchscreen-based UI.

3 Architecture

The software architecture uses two virtual machines, one running Linux, the other BREW.

The software architecture that was chosen for the Evoke is schematically shown in Figure 3.1: Two virtual machines, running on top of the OKL4 microvisor, interact via OKL4 message-passing IPC as well as shared memory.

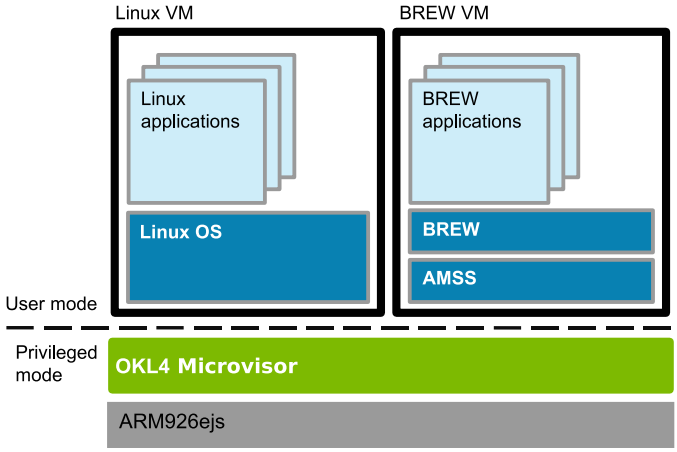

Figure 3.1. Software architecture of the Evoke.

The complete Linux system is de-privileged, executing in user mode. The complete AMSS/BREW baseband stack and OS also run in user mode. This is a no-compromises design using proper virtualization without shortcuts.

The virtualization overhead is kept to almost unnoticeable levels by the high-performance OKL4 Microvisor. In fact, in many respects it is better than what would be achievable with Linux running native.

The virtualization overhead is kept to almost unnoticeable levels by the high-performance OKL4 Microvisor.

This is made possible by the fast-context-switching technology [WTUH03, vSH07] that is unique to OK Labs products. Through smart management of the address space, and utilising every last feature of the ARM9's MMU, OKL4 avoids flushing caches and TLBs on context switches, *without sacrificing performance*. This results in the seemingly paradoxical situation that virtualized Linux can outperform native Linux.

Figure 3.2 demonstrates how context-switching latencies in OK:Linux are dramatically reduced compared to native Linux. The improvement is particularly large if the number of presently active processes is small—a typical situation for mobile phones. The boost in context-switching performance especially benefits the Evoke's user interface.

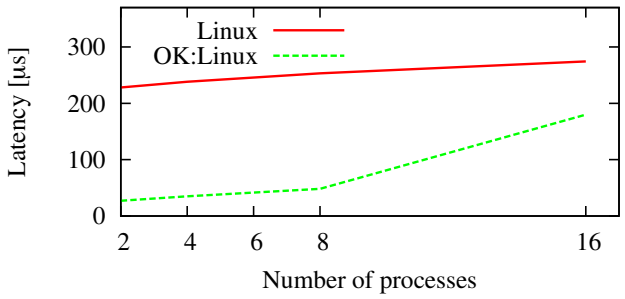

Figure 3.2. Context-switch latency as measured by LMbench for native and virtualized Linux.

4 User Interface Integration

The touch screen of the Evoke is normally owned by the Linux VM. BREW applications are represented by icons on the Linux desktop, indistinguishable (to the user) from Linux apps. Specifically, all the user-interface functionality is implemented by Linux apps, while video rendering uses a rendering engine running on BREW.

BREW and Linux applications co-exist on the same Linux desktop, integrated into the same user interface.

When the user requests invocation of a BREW app, Linux communicates with BREW in the other VM, which then starts up the app. In order to facilitate access to the screen, the BREW app is given access to the frame buffer by a shared-memory mapping.

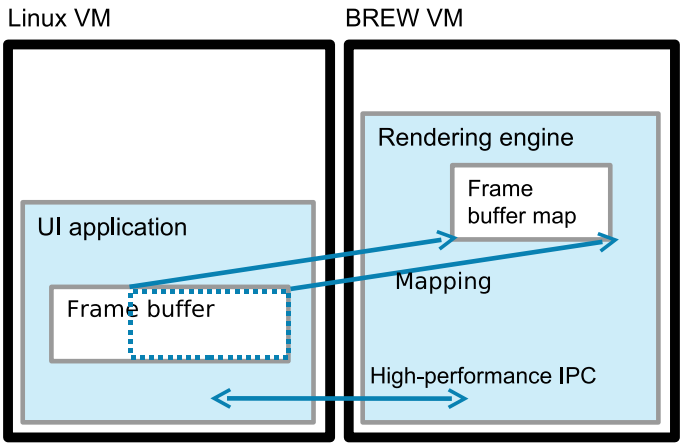

Figure 4.1. Memory sharing and IPC across virtual machines.

The seamless integration is enabled by OKL4's high-performance communication and its support for fast establishment and removal of shared memory mappings.

The seamless integration of the two subsystems is enabled by two critical features of OKL4, indicated in Figure 4.1: support for quickly establishing and removing shared-memory mappings between the virtual machines, and a message-passing communication (IPC) primitive with extremely low latency. This high-performance IPC has long been a hallmark of L4 microkernels, of which OKL4 is the commercial representative, and is one of the main features distinguishing OKL4 from competing products. It is hard to see how the Evoke design could have been realised with a virtualization offering other than OKL4.

5 Conclusions

The outcome is a device with an attractive, snappy user interface that has earned Motorola praise [Mie09]. In fact, the user interface is more responsive than some other (non-virtualized) phones, including some which are based on much more powerful ARM11 processors.

While BOM-cost savings are estimated to be US$10–15, even more important was low engineering costs made possible by maximising software reuse.

The bill-of-materials cost is estimated to be US$10–15 less than that of an equivalent dual-core design. But possibly even more important is that engineering costs were kept low by maximising software re-use, specifically the ability to invoke BREW applications from the Linux-based UI. This was enabled by the high-performance message-passing IPC operation provided by OKL4, which kept communication overheads between the virtual machines lower than would normally be achieved between OSes running native on separate cores.

The basic design with two highly-interacting VMs confirms our earlier claims about the necessity of supporting strong integration and interaction in embedded systems [Hei08]. It is at odds with claims made by competitors that communication efficiency between VMs is not relevant.

Bibliography

[Hei08] Gernot Heiser. The role of virtualization in embedded systems. In *1st Workshop on Isolation and Integration in Embedded Systems*, pages 11–16, Glasgow, UK, April 2008. ACM SIGOPS.

[Mie09] Ginny Mies. Hands on: Motorola Evoke QA4. http://www.pcworld.com/article/162459/hands_on_motorola_evoke_qa4.html, April 2009.

[Qua] Qualcomm BREW web site. http://brew.qualcomm.com.

[vSH07] Carl van Schaik and Gernot Heiser. High-performance microkernels and virtualisation on ARM and segmented architectures. In *Proceedings of the 1st International Workshop on Microkernels for Embedded Systems*, Sydney, Australia, January 2007. NICTA.

[WTUH03] Adam Wiggins, Harvey Tuch, Volkmar Uhlig, and Gernot Heiser. Implementation of fast address-space switching and TLB sharing on the StrongARM processor. In *Proceedings of the 8th Asia-Pacific Computer Systems Architecture Conference*, Aizu-Wakamatsu City, Japan, September 2003. Springer Verlag.

About the Author

Dr Gernot Heiser is co-founder and Chief Technology Officer of Open Kernel Labs (OK Labs). As CTO, his specific responsibility is to set the strategic direction of the company's research and development in order to maintain and further expand the technology leadership of OK Labs.

Prior to founding OK Labs, Dr. Heiser set up and lead the Embedded Operating Systems (ERTOS) research group at NICTA, Australia's Information and Communications Technology (ICT) Research Centre of Excellence, and has established ERTOS as a recognised world leader in embedded operating-systems research. Dr Heiser continues in this position on a part-time basis, which ensures the strategic alignment of OK Labs and ERTOS, and the smooth transfer of ERTOS research outcomes for commercialisation in OK Labs.

Prior to NICTA's creation in 2003, Dr Heiser was a faculty member at the University of New South Wales (UNSW), where he created a suite of world-class OS courses, lead the development of several research operating systems, and built the group that provided the foundation for ERTOS and later OK Labs. He still holds the *John Lions Chair of Operating Systems* at UNSW, and continues to teach advanced-level courses and supervise a large number of PhD students.

Gernot Heiser holds a PhD in Computer Science from ETH Zurich, Switzerland. He is a senior member of the IEEE, and a member of the ACM and of Usenix.

About Open Kernel Labs

Open Kernel Labs (OK Labs) is a leading provider of embedded systems software and virtualization technology. Spun out from NICTA, the Australian National Centre of Excellence for Information and Communications Technology, OK Labs is focussed on driving the state of the art in embedded operating systems. OK Labs provides products that improve the reliability, safety and security of embedded devices.

OK Labs believes that the best technology should have nothing to hide, and consequently distributes its code as open source. The company also believes that dramatic improvements in system reliability are possible in the near future, and to this end collaborates closely with NICTA and other research institutions on creating and commercialising the next generation of embedded operating-systems technology. For more information on OK Labs and its products visit `http://www.ok-labs.com`.