MORGAN & CLAYPOOL PUBLISHERS

Hardware and Software Support for Virtualization

Edouard Bugnion
Jason Nieh
Dan Tsafrir

SYNTHESIS LECTURES ON COMPUTER ARCHITECTURE

Margaret Martonosi, *Series Editor*

Hardware and Software Support for Virtualization

Hardware and Software Support for Virtualization

Edouard Bugnion, Jason Nieh, and Dan Tsafrir

www.morganclaypool.com

ISBN: 9781627056939 paperback
ISBN: 9781627056885 ebook

DOI 10.2200/S00754ED1V01Y201701CAC038

A Publication in the Morgan & Claypool Publishers series
SYNTHESIS LECTURES ON COMPUTER ARCHITECTURE

Lecture #38
Series Editor: Margaret Martonosi, *Princeton University*
Series ISSN
Print 1935-3235 Electronic 1935-3243

Synthesis Lectures on Computer Architecture

Editor
Margaret Martonosi, *Princeton University*

Synthesis Lectures on Computer Architecture publishes 50- to 100-page publications on topics pertaining to the science and art of designing, analyzing, selecting and interconnecting hardware components to create computers that meet functional, performance and cost goals. The scope will largely follow the purview of premier computer architecture conferences, such as ISCA, HPCA, MICRO, and ASPLOS.

Hardware and Software Support for Virtualization
Edouard Bugnion, Jason Nieh, and Dan Tsafrir
2017

Datacenter Design and Management: A Computer Architect's Perspective
Benjamin C. Lee
2016

A Primer on Compression in the Memory Hierarchy
Somayeh Sardashti, Angelos Arelakis, Per Stenström, and David A. Wood
2015

Research Infrastructures for Hardware Accelerators
Yakun Sophia Shao and David Brooks
2015

Analyzing Analytics
Rajesh Bordawekar, Bob Blainey, and Ruchir Puri
2015

Customizable Computing
Yu-Ting Chen, Jason Cong, Michael Gill, Glenn Reinman, and Bingjun Xiao
2015

Die-stacking Architecture
Yuan Xie and Jishen Zhao
2015

Single-Instruction Multiple-Data Execution
Christopher J. Hughes
2015

Power-Efficient Computer Architectures: Recent Advances
Magnus Själander, Margaret Martonosi, and Stefanos Kaxiras
2014

FPGA-Accelerated Simulation of Computer Systems
Hari Angepat, Derek Chiou, Eric S. Chung, and James C. Hoe
2014

A Primer on Hardware Prefetching
Babak Falsafi and Thomas F. Wenisch
2014

On-Chip Photonic Interconnects: A Computer Architect's Perspective
Christopher J. Nitta, Matthew K. Farrens, and Venkatesh Akella
2013

Optimization and Mathematical Modeling in Computer Architecture
Tony Nowatzki, Michael Ferris, Karthikeyan Sankaralingam, Cristian Estan, Nilay Vaish, and David Wood
2013

Security Basics for Computer Architects
Ruby B. Lee
2013

The Datacenter as a Computer: An Introduction to the Design of Warehouse-Scale Machines, Second edition
Luiz André Barroso, Jimmy Clidaras, and Urs Hölzle
2013

Shared-Memory Synchronization
Michael L. Scott
2013

Resilient Architecture Design for Voltage Variation
Vijay Janapa Reddi and Meeta Sharma Gupta
2013

Multithreading Architecture
Mario Nemirovsky and Dean M. Tullsen
2013

Performance Analysis and Tuning for General Purpose Graphics Processing Units (GPGPU)
Hyesoon Kim, Richard Vuduc, Sara Baghsorkhi, Jee Choi, and Wen-mei Hwu
2012

Automatic Parallelization: An Overview of Fundamental Compiler Techniques
Samuel P. Midkiff
2012

Phase Change Memory: From Devices to Systems
Moinuddin K. Qureshi, Sudhanva Gurumurthi, and Bipin Rajendran
2011

Multi-Core Cache Hierarchies
Rajeev Balasubramonian, Norman P. Jouppi, and Naveen Muralimanohar
2011

A Primer on Memory Consistency and Cache Coherence
Daniel J. Sorin, Mark D. Hill, and David A. Wood
2011

Dynamic Binary Modification: Tools, Techniques, and Applications
Kim Hazelwood
2011

Quantum Computing for Computer Architects, Second Edition
Tzvetan S. Metodi, Arvin I. Faruque, and Frederic T. Chong
2011

High Performance Datacenter Networks: Architectures, Algorithms, and Opportunities
Dennis Abts and John Kim
2011

Processor Microarchitecture: An Implementation Perspective
Antonio González, Fernando Latorre, and Grigorios Magklis
2010

Transactional Memory, 2nd edition
Tim Harris, James Larus, and Ravi Rajwar
2010

Computer Architecture Performance Evaluation Methods
Lieven Eeckhout
2010

Introduction to Reconfigurable Supercomputing
Marco Lanzagorta, Stephen Bique, and Robert Rosenberg
2009

On-Chip Networks
Natalie Enright Jerger and Li-Shiuan Peh
2009

The Memory System: You Can't Avoid It, You Can't Ignore It, You Can't Fake It
Bruce Jacob
2009

Fault Tolerant Computer Architecture
Daniel J. Sorin
2009

The Datacenter as a Computer: An Introduction to the Design of Warehouse-Scale Machines
Luiz André Barroso and Urs Hölzle
2009

Computer Architecture Techniques for Power-Efficiency
Stefanos Kaxiras and Margaret Martonosi
2008

Chip Multiprocessor Architecture: Techniques to Improve Throughput and Latency
Kunle Olukotun, Lance Hammond, and James Laudon
2007

Transactional Memory
James R. Larus and Ravi Rajwar
2006

Quantum Computing for Computer Architects
Tzvetan S. Metodi and Frederic T. Chong
2006

Hardware and Software Support for Virtualization

Edouard Bugnion
École Polytechnique Fédérale de Lausanne (EPFL), Switzerland

Jason Nieh
Columbia University

Dan Tsafrir
Technion – Israel Institute of Technology

SYNTHESIS LECTURES ON COMPUTER ARCHITECTURE #38

MORGAN & CLAYPOOL PUBLISHERS

ABSTRACT

This book focuses on the core question of the necessary *architectural support provided by hardware* to efficiently run virtual machines, and of the corresponding design of the *hypervisors* that run them. Virtualization is still possible when the instruction set architecture lacks such support, but the hypervisor remains more complex and must rely on additional techniques.

Despite the focus on architectural support in current architectures, some historical perspective is necessary to appropriately frame the problem. The first half of the book provides the historical perspective of the theoretical framework developed four decades ago by Popek and Goldberg. It also describes earlier systems that enabled virtualization despite the lack of architectural support in hardware.

As is often the case, theory defines a necessary—but not sufficient—set of features, and modern architectures are the result of the combination of the theoretical framework with insights derived from practical systems. The second half of the book describes state-of-the-art support for virtualization in both x86-64 and ARM processors. This book includes an in-depth description of the CPU, memory, and I/O virtualization of these two processor architectures, as well as case studies on the Linux/KVM, VMware, and Xen hypervisors. It concludes with a performance comparison of virtualization on current-generation x86- and ARM-based systems across multiple hypervisors.

KEYWORDS

computer architecture, virtualization, virtual machine, hypervisor, dynamic binary translation

Preface

> "Virtual machines have finally arrived. Dismissed for a number of years as merely academic curiosities, they are now seen as cost-effective techniques for organizing computer systems resources to provide extraordinary system flexibility and support for certain unique applications".
>
> Robert. P. Goldberg, *IEEE Computer*, 1974 [78]

The academic discipline of computer systems research, including computer architecture, is in many aspects more tidal than linear: specific ingrained, well-understood techniques lose their relevance as tradeoffs evolve. Hence, the understanding of these techniques then ebbs from the collective knowledge of the community. Should the architectural tide later flow in the reverse direction, we have the opportunity to reinvent—or at least appreciate once more—old concepts all over again.

The history of virtualization is an excellent example of this cycle of innovation. The approach was popular in the early era of computing, as demonstrated from the opening quote. At high tide in the 1970s, hundreds of papers were written on virtualization with conferences and workshops dedicated to the topic. The era established the basic principles of virtualization and entire compute stacks—hardware, virtual machine monitors, and operating systems—were designed to efficiently support virtual machines. However, the tide receded quickly in the early 1980s as operating systems matured; virtual machines were soon strategically discarded in favor of a more operating system-centric approach to building systems.

Throughout the 1980s and 1990s, with the appearance of the personal computer and client/server era, virtual machines were largely relegated to a mainframe-specific curiosity. For example, the processors developed in that era (MIPS, Sparc, x86), were not explicitly designed to provide architectural support for virtualization, since there was no obvious business requirement to maintain support for virtual machines. In addition, and in good part because of the ebb of knowledge of the formal requirements for virtualization, many of these architectures made arbitrary design decisions that violated the basic principles established a decade earlier.

For most computer systems researchers of the open systems era, raised on UNIX, RISC, and x86, virtual machines were perceived to be just another bad idea from the 1970s. In 1997, the Disco [44] paper revisited virtual machines with a fresh outlook, specifically as the founda-

tion to run commodity operating systems on scalable multiprocessors. In 1999, VMware released VMware Workstation 1.0 [45], the first commercial virtualization solution for x86 processors.

At the time, researchers and commercial entities started building virtual machines solutions for desktops and servers. A few years later, the approach was introduced to mobile platforms. Disco, VMware Workstation, VMware ESX Server [177], VirtualPC [130], Xen [27], Denali [182], and Cells [16], were all originally designed for architectures that did *not* provide support for virtualization. These different software systems each took a different approach to work around the limitations of the hardware of the time. Although processor architectures have evolved to provide hardware support for virtualization, many of the key innovations of that era such as hosted architectures [162], paravirtualization [27, 182], live migration [51, 135], and memory ballooning [177], remain relevant today, and have a profound impact on computer architecture trends.

Clearly, the virtualization tide has turned, to the point that it is once more a central driver of innovation throughout the industry, including system software, systems management, processor design, and I/O architectures. As a matter of fact, the exact quote from Goldberg's 1974 paper would have been equally timely 30 years later: Intel introduced its first-generation hardware support for virtual machines in 2004. Every maintained virtualization solution, including VMware Workstation, ESX Server, and Xen, quickly evolved to leverage the benefits of hardware support for virtualization. New systems were introduced that assumed the existence of such hardware support as a core design principle, notably KVM [113]. With the combined innovation in hardware and software and the full support of the entire industry, virtual machines quickly became central to IT organizations, where they were used among other things to improve IT efficiency, simplify provisioning, and increase availability of applications. Virtual machines were also proposed to uniquely solve hard open research questions, in domains such as live migration [51, 135] and security [73]. Within a few years, they would play a central role in enterprise datacenters. For example, according to the market research firm IDC, since 2009 there are more virtual machines deployed than physical hosts [95].

Today, virtual machines are ubiquitous in enterprise environments, where they are used to virtualize servers as well as desktops. They form the foundation of all Infrastructure-as-a-Service (IAAS) clouds, including Amazon EC2, Google CGE, Microsoft Azure, and OpenStack. Once again, the academic community dedicates conference tracks, sessions, and workshops to the topic (e.g., the annual conference on Virtual Execution Environments (VEE)).

ORGANIZATION OF THIS BOOK

This book focuses on the core question of the necessary *architectural support provided by hardware* to efficiently run virtual machines. Despite the focus on architectural support in current architectures, some historical perspective is necessary to appropriately frame the problem. Specifically, this includes both a theoretical framework, and a description of the systems enabling virtualization despite the lack of architectural support in hardware. As is often the case, theory defines

a necessary—but not sufficient—set of features, and modern architectures are the result of the combination of the theoretical framework with insights derived from practical systems.

The book is organized as follows.

- Chapter 1 introduces the fundamental definitions of the abstraction ("virtual machines"), the run-time ("virtual machine monitors"), and the principles used to implement them.
- Chapter 2 provides the necessary theoretical framework that defines whether an instruction set architecture (ISA) is virtualizable or not, as formalized by Popek and Goldberg [143].
- Chapter 3 then describes the first set of systems designed for platforms that failed the Popek/Goldberg test. These systems each use a particular combination of workarounds to run virtual machines on platforms not designed for them. Although a historical curiosity by now, some of the techniques developed during that era remain relevant today.
- Chapter 4 focuses on the architectural support for virtualization of modern x86-64 processors, and in particular Intel's VT-x extensions. It uses KVM as a detailed case study of a hypervisor specifically designed to assume the presence of virtualization features in processors.
- Chapter 5 continues the description of x86-64 on the related question of the architectural support for MMU virtualization provided by extended page tables (also known as nested page tables).
- Chapter 6 closes the description of x86-64 virtualization with the various forms of I/O virtualization available. The chapter covers key concepts such as I/O emulation provided by hypervisors, paravirtual I/O devices, pass-through I/O with SR-IOV, IOMMUs, and the support for interrupt virtualization.
- Chapter 7 describes the architectural support for virtualization of the ARM processor family, and covers the CPU, MMU, and I/O considerations. The chapter emphasizes some of the key differences in design decisions between x86 and ARM.
- Chapter 8 compares the performance and overheads of virtualization extensions on x86 and on ARM.

In preparing this book, the authors made some deliberate decisions. First, for brevity, we focused on the examples of architectural support for virtualization, primarily around two architectures: x86-64 and ARM. Interested readers are hereby encouraged to study additional instruction set architectures. Among them, IBM POWER architecture, with its support for both hypervisor-based virtualization and logical partitioning (LPAR), is an obvious choice [76]. The SPARC architecture also provides built-in support for logical partitioning, called logical domains [163]. We also omit any detailed technical description of mainframe and mainframe-era architectures. Readers

interested in that topic should start with Goldberg's survey paper [78] and Creasy's overview of the IBM VM/370 system [54].

Second, we focused on mainstream (i.e., traditional) forms of virtual machines and the construction of hypervisors in both the presence or the absence of architectural support for virtualization in hardware. This focus is done at the expense of a description of some more advanced research concepts. For example, the text does not discuss recursive virtual machines [33, 158], the use of virtualization hardware for purposes other than running traditional virtual machines [24, 29, 31, 43, 88], or the emerging question of architectural support for containers such as Docker [129].

AUTHORS' PERSPECTIVES

This book does not attempt to cover all aspects of virtualization. Rather, it mostly focuses on the key question of the interaction between the underlying computer architecture and the systems software built on top of it. It also comes with a point of view, based on the authors' direct experiences and perspectives on the topic.

Edouard Bugnion was fortunate to be part of the Disco team as a graduate student. Because of the stigma associated with virtual machines of an earlier generation, we named our prototype in reference to the questionable musical contribution of that same decade [55], which was then coincidentally making a temporary comeback. Edouard later co-founded VMware, where he was one of the main architects and implementers of VMware Workstation, and then served as its Chief Technology Officer. In 2005, he co-founded Nuova Systems, a hardware company premised on providing architectural support for virtualization in the network and the I/O subsystem, which became the core of Cisco's Data Center strategy. More recently, having returned to academia as a professor at École polytechnique fédérale de Lausanne (EPFL), Edouard is now involved in the IX project [30, 31, 147] which leverages virtualization hardware and the Dune framework [29] to build specialized operating systems.

Jason Nieh is a Professor of Computer Science at Columbia University, where he has led a wide range of virtualization research projects that have helped shape commercial and educational practice. Zap [138], an early lightweight virtual machine architecture that supported migration, led to the development of Linux namespaces and Linux containers, as well as his later work on Cells [16, 56], one of the first mobile virtualization solutions. Virtual Layered File Systems [144, 145] introduced the core ideas of layers and repositories behind Docker and CoreOS. KVM/ARM [60] is widely deployed and used as the mainline Linux ARM hypervisor, and has led to improvements in ARM architectural support for virtualization [58]. MobiDesk [26], THINC [25], and other detailed measurement studies helped make the case for virtual desktop infrastructure, which has become widely used in industry. A dedicated teacher, Jason was the first to introduce virtualization as a pedagogical tool for teaching hands-on computer science courses, such as operating systems [136, 137], which has become common practice in universities around the world.

Dan Tsafrir is an Associate Professor at the Technion—Israel Institute of Technology, where he regularly appreciates how fortunate he is to be working with brilliant students on cool projects for a living. Some of these projects drive state-of-the-art virtualization forward. For example, vIOMMU showed for the first time how to fully virtualize I/O devices on separate (side)cores without the knowledge or involvement of virtual machines, thus eliminating seemingly inherent trap-and-emulate virtualization overheads [12]. vRIO showed that such sidecores can in fact be consolidated on separate remote servers, enabling a new kind of datacenter-scale I/O virtualization model that is cheaper and more performant than existing alternatives [116]. ELI introduced software-based exitless interrupts—a concept recently adopted by hardware—which, after years of efforts, finally provided bare-metal performance for high-throughput virtualization workloads [13, 80]. VSwapper showed that uncooperative swapping of memory of virtual machines can be made efficient, despite the common belief that this is impossible [14]. Virtual CPU validation showed how to uncover a massive amount of (confirmed and now fixed) hypervisor bugs by applying Intel's physical CPU testing infrastructure to the KVM hypervisor [15]. EIOVAR and its successor projects allowed for substantially faster and safer IOMMU protection and found their way into the Linux kernel [126, 127, 142]. NPFs provide page-fault support for network controllers and are now implemented in production Mellanox NICs [120].

TARGET AUDIENCE

This book is written for researchers and graduate students who have already taken a basic course in both computer architecture and operating systems, and who are interested in becoming fluent with virtualization concepts. Given the recurrence of virtualization in the literature, it should be particularly useful to new graduate students before they start reading the many papers treating a particular sub-aspect of virtualization. We include numerous references of widely read papers on the topic, together with a high-level, modern commentary on their impact and relevance today.

Edouard Bugnion, Jason Nieh, and Dan Tsafrir
January 2017

Acknowledgments

This book would not have happened without the support of many colleagues. The process would have not even started without the original suggestion from Rich Uhlig to Margaret Martonosi, the series editor. The process, in all likelihood, would have never ended without the constant, gentle probing of Mike Morgan; we thank him for his patience. Ole Agesen, Christoffer Dall, Arthur Kiyanovski, Shih-Wei Li, Jintack Lim, George Prekas, Jeff Sheldon, and Igor Smolyar provided performance figures specifically for this book; students will find the additional quantitative data enlightening. Margaret Church made multiple copy-editing passes to the manuscript; we thank her for the diligent and detailed feedback at each round. Nadav Amit, Christoffer Dall, Nathan Dauthenhahn, Canturk Isci, Arthur Kiyanovski, Christos Kozyrakis, Igor Smolyar, Ravi Soundararajan, Michael Swift, and Idan Yaniv all provided great technical feedback on the manuscript.

The authors would like to thank EPFL, Columbia University, and the Technion—Israel Institute of Technology, for their institutional support. Bugnion's research group is supported in part by grants from Nano-Tera, the Microsoft EPFL Joint Research Center, a Google Graduate Research Fellowship and a VMware research grant. Nieh's research group is supported in part by ARM Ltd., Huawei Technologies, a Google Research Award, and NSF grants CNS-1162447, CNS-1422909, and CCF-1162021. Tsafrir's research group is supported in part by research awards from Google Inc., Intel Corporation, Mellanox Technologies, and VMware Inc., as well as by funding from the Israel Science Foundation (ISF) grant No. 605/12, the Israeli Ministry of Economics via the HIPER consortium, the joint BSF-NSF United States-Israel Binational Science Foundation and National Science Foundation grant No. 2014621, and the European Union's Horizon 2020 research and innovation programme grant agreement No. 688386 (OPERA).

Edouard Bugnion, Jason Nieh, and Dan Tsafrir
Lausanne, New York, and Haifa
January 2017

Contents