esi—Pointer to data held in the DS segment; source pointer for string operations

edi—Pointer to data held in the ES segment; destination pointer for string operations

esp—Stack pointer in the SS segment

ebp—Pointer to data held in the stack

The special-purpose registers consist of five control registers, CR0—4, that contain a number of status and enable bits. For example, when a page fault occurs, CR2 is specified to hold the address causing the fault, while CR3 points to the page table. There are three other special-purpose registers, the EFLAGS, EIP, and IDTR registers. The EFLAGS register contains a number of status flags, including condition codes used by conditional branch instructions. The EIP is the program counter (referred to as the *instruction pointer* in Intel terminology). The IDTR, or the interrupt descriptor table register, points to a table in memory that holds interrupt vectors.

Memory Architecture

The IA-32 memory architecture (Figure A.25) is based on segments, somewhat similar to the PowerPC, but it is often used in a special configuration that presents a flat, linear model to software. The IA-32 ISA supports up to 64K segments, each of which can be up to 4GB. The segments map into a single, contiguous 32-bit linear address space (4GB), though at any given time only six of the segments can be addressed via six segment registers.

Like the PowerPC, a logical load or store address specifies one of the segment registers and an offset of up to 32 bits. Some instruction encodings can only specify one of the first four segments. The logical address points to a location in the linear 4GB address space. When the offset value is computed, typically by adding the contents of a

general-purpose register to a displacement value, the segment register is not changed. This is in contrast to PowerPC, where the segment pointer is in the upper bits of the effective address and can be affected by address arithmetic.

When used in its most general form, a segmented memory model can be constructed as shown in Figure A.25a. However, by setting all the segment registers to point to the base of the 32-bit linear address space, a simpler linear memory model can be constructed, as in Figure A.25b. It is this model that is used by most UNIX systems.

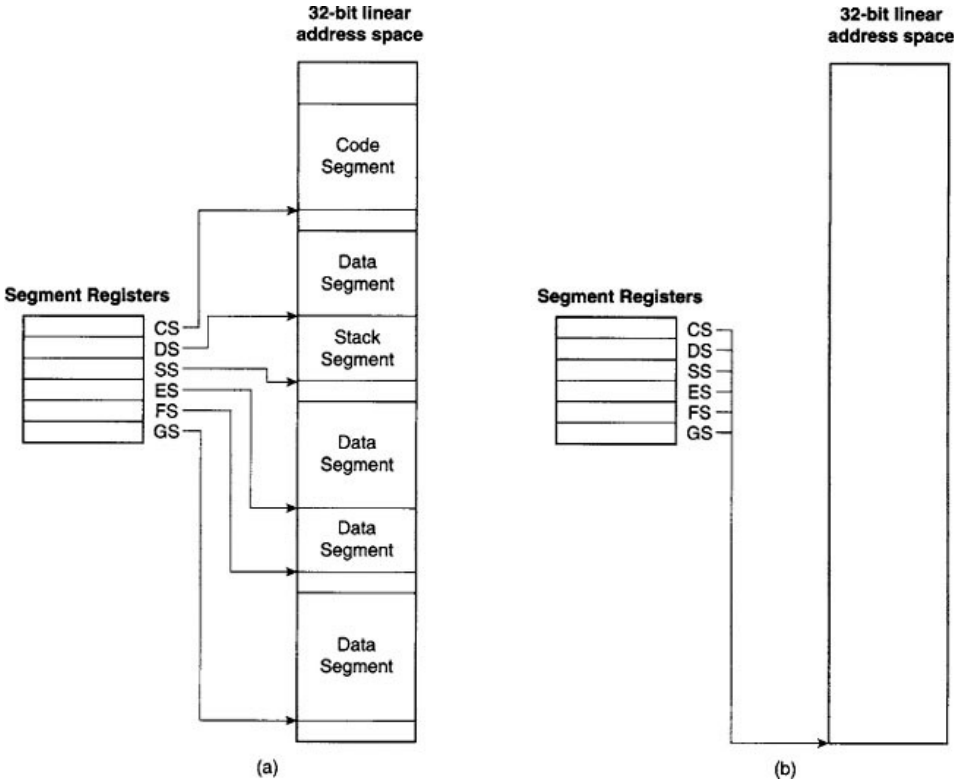


Figure A.25 Virtual memory Models Supported by IA-32. *(a) Segmented memory model; (b) flat memory model. The IA-32 also supports a real address model which is not shown.*

The linear address space is divided into 4KB pages, irrespective of the segments. The flat linear address space is then mapped onto a real memory that may have up to 4GB. As with the PowerPC, the Intel IA-32 has an architected page table, and hence the format of entries in the page table is defined in the ISA. The page table is similar to the one shown in Figure A.10, but it contains two levels rather

far to the one shown in Figure A.10, but it contains two levels rather than one. The 20-bit page number field of a virtual address is divided into an upper 10-bit directory pointer and a lower 10-bit page pointer. The directory pointer points to an entry in a 1K page directory. Each directory pointer points to a separate 1K entry page table, which is accessed using the lower 10-bit page pointer. The page directory is located at an address specified in CR3, which acts as the page directory base pointer.

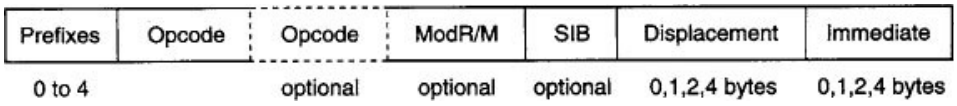


Figure A.26 General Format for IA-32 Instructions.

Instruction Set Summary

The IA-32 ISA has variable-length instructions, from one to fifteen bytes, although most of the commonly used instructions are relatively short. Instruction operands, including instructions that perform common arithmetic, logical, or shift operations, may come from either registers or memory. This is in contrast to RISC instruction sets, where the only instructions that take operands from memory are the load and store instructions.

Figure A.26 illustrates the general form of an IA-32 instruction. It consists of from zero to four prefix bytes. The prefix bytes indicate various special cases, for example, if there is repetition for string instructions and/or if there are overrides for addressing segments, address sizes, and operand sizes. Following the prefix byte(s) (if any) is an opcode byte, which may be followed by a second opcode byte, depending on the value of the first. Next comes an optional addressing-form specifier, ModR/M. It is present only for certain opcodes and generally indicates an addressing mode and register. The SIB byte is present only for certain ModR/M encodings and indicates a base register, an index register, and a scale factor for the indexing. The variable-length displacement field is present only when certain

addressing modes are specified. The last field is a variable-length immediate operand, present only if required by the opcode. When needed, a segment register can be specified with some of the opcode bits or by the ModR/M byte.

The IA-32 ISA divides the user instruction set into three major categories: integer instructions, floating-point instructions, and MMX (multimedia instructions). In this brief overview and in examples used in this book, we restrict ourselves primarily to the integer instructions. Within the category of integer instructions are jumps (both conditional and unconditional), load/store instructions, and a wide variety of ALU instructions. However, as we noted earlier, the ALU instructions can specify one or more memory operands, so, in effect, many of them also perform load/store operations.

Jump instructions include all control transfer instructions, regardless of whether they are conditional or unconditional. The conditional jumps test condition code bits that are held in the EFLAGS register. These bits are set as a side effect of many instructions.

jmp	reg	Jump indirect to the memory location held in reg.
jz	target	Jump conditionally to target if the zero-condition code bit in the EFLAGS register is set to 1.
jmp	target	Jump unconditionally to target.

Procedure calls and returns are performed using instructions that push and pop the return address to/from the stack, pointed to by esp.

call	target	Push the program counter of the next instruction onto the stack and jump to the target address.
ret		Pop the return address from the stack and jump.

Load and store operations, when not combined with arithmetic or logical operations, are performed by move instructions.

mov	reg1, disp(reg2)	Load register reg1 with the contents of memory addressed by (reg2) + disp. The disp field is treated as a signed quantity.
mov	disp(reg1), reg2	Store register reg2 into memory addressed by (reg1) + disp. The disp field is treated as a signed quantity.

Most of the ALU operations can take operands from a combination of registers and memory locations. Some representative examples follow.

addl	reg1, disp(reg2)	Add the contents of reg1 and the memory operand at address reg2 + disp. The result is placed in reg1. Condition codes in the EFLAGS register are set according to the result.
sub	reg1, immed	Subtract the contents of the immediate value immed from reg1 and place the result in reg1. Condition codes in the EFLAGS register are set according to the result.
xorl	reg1, reg2	Exclusive-OR the contents of reg1 and reg2 and place the result in reg1. Condition codes in the EFLAGS register are set according to the result.

1. Complex-instruction-set computer.

2. Reduced-instruction-set computer.

3. This distinction between branch and jump instructions is not universal. Some ISAs use only one of the two terms, and others interchange their meanings.

Previous chapter
< Chapter 10: Emerging Applications

Next chapter
References >

References

Adair, R., R. U. Bayles, L. W. Comeau, and R. J. Creasy. 1966. A Virtual Machine System for the 360/40, *Cambridge Scientific Center Report No. G320–2007* (May).

Adve, V., C. Lattner, M. Brukman, A. Shukla, and B. Gaeke. 2003. LLVA: A Low-level Virtual Instruction Set Architecture, *Proc. 36th Int. Symp. on Microarchitecture* (December).

Aho, A. V., R. Sethi, and J. D. Ullman. 1986. *Compilers: Principles, Techniques and Tools,* Addison-Wesley, Reading, MA.

Allcock, B., J. Bester, J. Breshanan, A. Chervenak, I. Foster, C. Kesselman, S. Meder, V. Nefedova, D. Quesnel, and S. Tuecke. 2001. Secure, Efficient Data Transport and Replica Management for High-Performance Data-Intensive Computing, *Proc. 18th Symposium on Mass Storage Systems and Technologies* (April).

Altman, E., K. Ebcioglu, M. Gschwind, and S. Sathaye. 2002. Methods and Apparatus for Reordering and Renaming Memory References in a Multiprocessor Computer System, US patent 6,349,361 (February).

Amdahl, G. M., G. A. Blaauw, and F. P. Brooks, Jr. 1964. Architecture of the IBM System 360, *IBM Journal of Research and Development* (April), pp. 87–101.

Anderson, D. P., J. Cobb, E. Korpella, M. Lebofsky, and D. Werthimer. 2002. SETI@home: An Experiment in Public-Resource Computing, *Communications of the ACM* (November), pp. 56–61.

Appel, A. W. 1991. Garbage Collection, *Topics in Advanced Language Implementations,* P. Lee (Ed.), MIT Press, Cambridge, MA, pp. 89–100.

ARM. 2002, ARM 9EJ-S Technical Reference Manual, Ref. DDI0222B.

Arnold, M. and B. G. Ryder. 2001. A Framework for Reducing the Cost of Instrumented Code, *Proc. Conf. ACM SIGPLAN '01 on Programming Language Design and Implementation* (May), pp. 168–179.

Arnold, M., M. Hind, and B. G. Ryder. 2002. Online Feedback-Directed Optimization of Java, *Proc. Conf. on Object Oriented Programming Systems, Languages, and Applications* (November), pp. 111–129.

Arnold, M., S. Fink, D. Grove, M. Hind, and P. F. Sweeney. 2000. Adaptive Optimization in the Jalapeno JVM, *Proc. Conf. on Object Oriented Programming Systems, Languages, and Applications* (October), pp. 47–65.

Arnold, M., S. Fink, D. Grove, M. Hind, and P. F. Sweeney. 2005. A Survey of Adaptive Optimization in Virtual Machines, *Proceedings of the IEEE* (February).

Artigas, P. V., M. Gupta, S. P. Midkiff, and J. E. Moreira. 2000. Automatic Loop Transformations and Parallelization for Java, *Proc. Int. Conf. on Supercomputing* (May), pp. 1–10.

Attanasio, C. R., D. F. Bacon, A. Cocchi, and S. Smith. 2001. A Comparative Evaluation of Parallel Garbage Collector Implementations, *Proc. 14th Workshop on Languages and Compilers for Parallel Computing* (August), pp. 177–192.

Aycock, J. 2003. A Brief History of Just-in-Time, *ACM Comp. Surveys* (June), pp. 97–113.

Ayers, A., R. Schooler, and R. Gottlieb. 1997. Aggressive Inlining, *Proc. ACM SIGPLAN Conf. on Programming Language Design and*

Proc. ACM SIGPLAN Conf. on Programming Language Design and Implementation (May), pp. 134–145.

Bala, V., E. Duesterwald, and S. Banerjia. 1999. Transparent Dynamic Optimization: The Design and Implementation of Dynamo, *Hewlett Packard Laboratories Technical Report HPL-1999-78* (June).

Bala, V., E. Duesterwald, and S. Banerjia. 2000. Dynamo: A Transparent Dynamic Optimization System, *Proc. ACM SIGPLAN Conf. Programming Language Design and Implementation* (June), pp. 1–12.

Ball, T. and J. R. Larus. 1994. Optimally Profiling and Tracing Programs, *ACM Transactions on Programming Languages and Systems* (July), pp. 1319–1360.

Ball, T. and J. R. Larus. 1996. Efficient Path Profiling, *Proc. 29th Int. Symp. on Microarchitecture* (November), pp. 46–57.

Ball, T., P. Mataga, and M. Sagiv. 1998. Edge Profiling Versus Path Profiling: The Showdown, *Proc. 25th ACM SIGPLAN-SIGACT Symp. on Principles of Programming Languages* (January), pp. 134–148.

Banerjia, S., W. A. Havanki, and T. M. Conte. 1997. Treegion Scheduling for Highly Parallel Processors, *European Conference on Parallel Processing* (August), pp. 1074–1078.

Banning, J., P. H. Anvin, B. Gribstad, D. Keppel, A. Klaiber, and P. Serris. 2002. Fine Grain Translation Discrimination, US Patent 6,363,336 (March).

Baraz, L, T. Devor, O. Etzion, S. Goldenberg, A. Skaletsky, Y. Wang, and Y. Zemach. 2003. IA-32 Execution Layer: A Two-phase Dynamic Translator Designed to Support IA-32 Applications on Itanium-based Systems, *Proc. 36th Annual IEEE/ACM Int. Symp. on Microarchitecture* (December), pp. 191–204.

Barham, P., B. Dragovic, K. Fraser, S. Hand, T. Harris, A. Ho, R. Neugebauery, I. Pratt, and A. Warfield. 2003. Xen and the Art of Virtualization, *Proc. 19th ACM Symp. on Operating System Principles* (October), pp. 164–177.

Bell, J. R. 1973. Threaded Code, *Communications of the ACM* (June), pp. 370–372.

Berndl, M. and L. Hendren. 2003. Dynamic Profiling and Trace Cache Generation, *Proc. Int. Symp. on Code Generation and Optimization* (March), pp. 276–285.

Bertsis, V. 1980. Security and Protection of Data in the IBM System/38, *Proc. 7th Int. Symp. on Computer Architecture* (June), pp. 245–252.

Boehm, H. and M. Weiser. 1988. Garbage Collection in an Uncooperative Environment, *Software—Practice and Experience,* pp. 807–820.

Borden, T. L., J. P. Hennessy, and J. W. Rymarczyk. 1989. Multiple Operating Systems on One Processor Complex, *IBM Systems Journal* (January), pp. 104–123

Boutcher, D. 2001. The Linux Kernel on iSeries, *Proc. 2001 Ottawa Linux Symposium* (July), **http://lwn.net/2001/features/OLS/pdf/pdf/iseries.pdf**.

Bovet, D. P. and M. Cesati. 2001. *Understanding the Linux Kernel, O'Reilly,* Sebastopol, CA.

Bowles, K. L. 1980. *Beginner's Guide for the UCSD Pascal System,* McGraw-Hill, New York.

Box, D. 2002. *Essential .NET, Volume 1: The Common Language*

Beck, B. 2002. *Essential JVM, Volume 1: The Solaris Language Runtime,* Addison-Wesley, Reading, MA.

Bressoud, T. C. and F. B. Schneider. 1996. Hypervisor-Based Fault Tolerance, *ACM Transactions on Computer Systems* (February), pp. 80–107.

Bruening, D. 2004. Efficient, Transparent, and Comprehensive Runtime Code Manipulation, Ph.D. dissertation (September), MIT, Cambridge, MA.

Bruening, D., E. Duesterwald, and S. Amarasinghe. 2001. Design and Implementation of a Dynamic Optimization Framework for Windows, *Proc. 4th Workshop on Feedback-Directed and Dynamic Optimization* (December).

Bruening, D., T. Garnett, and S. Amarasinghe. 2003. An Infrastructure for Adaptive Dynamic Optimization, *Proc. Int. Symp. on Code Generation and Optimization* (March), pp. 265–275.

Brunner, R. A. 1991. *VAX Architecture Reference Manual,* 2nd ed., Digital Press, Bedford, MA.

Bugnion, E., S. Devine, K. Govil, and M. Rosenblum. 1997. Disco: Running Commodity Operating Systems on Scalable Multiprocessors, *ACM Transactions on Computing Systems* (November), pp. 412–447.

Carr, S. and K. Kennedy. 1994. Scalar Replacement in the Presence of Conditional Control Flow, *Software—Practice and Experience* (January), pp. 51–77.

Chang, P. P., S. A. Mahlke, and W. W. Hwu. 1991. Using Profile Information to Assist Classic Code Optimizations, *Software—Practice & Experience* (December), table of contents, pp. 1301–1321.

Chang, P.P., S. A. Mahlke, W. Y. Chen, and W.-M. W. Hwu. 1992. Profile-Guided Automatic Inline Expansion for C Programs, *Software—Practice and Experience* (May), pp. 349–369.

Chambers, C. and D. Ungar. 1991. Making Pure Object-Oriented Languages Practical, *Proc. Conf. on Object-Oriented Programming Systems, Languages, and Applications* (October), pp. 1–15.

Charlu, D. 1999. HP Hyperplex Clustering Technology, *1st Int. Workshop on Cluster Computing* (December).

Chen, P. M. and B. D. Noble. 2001. When Virtual Is Better Than Real, *Proc. 8th IEEE Workshop on Hot Topics on Operating Systems* (May), pp. 133–138.

Chen, J. B. and B. D. D. Leupen. 1997. Improving Instruction Locality with Just-In-Time Code Layout, *Proc. the USENIX Windows NT Workshop* (August).

Chen, W.-K., S. Lerner, R. Chaiken, and D. M. Gillies. 2000. Mojo: A Dynamic Optimization System, *Proc. 3rd ACM Workshop on Feedback-Directed and Dynamic Optimization* (December).

Chernoff, A., M. Herdeg, R. Hookway, C. Reeve, N. Rubin, T. Tye, S. B. Yadavalli, and J. Yates. 1998. FX!32: A Profile-Directed Binary Translator, *IEEE Micro* (March), pp. 56–64.

Chien, A., B. Calder, S. Elbert, and K. Bhatia. 2003. Entropia: Architecture and Performance of an Enterprise Desktop Grid System, *Journal of Parallel Distributed Computing* (May), pp. 597–610.

Chilimbi, T. M., B. Davidson, and J. R. Larus. 1999. Cache-Conscious Structure Definition, *Proc. Conf. on Programming Language Design and Implementation* (May), pp. 13–24.

Choi, J.-D., M. Gupta, M. J. Serrano, V. C. Sreedhar, and S. P. Midkiff.

1999. Escape Analysis for Java, *Proc. Conf. on Object-Oriented Programming Systems, Languages, and Applications,* pp. 1–19.

Choi, Y., A. Knies, G. Vedaraman, J. Williamson, and I. Esmer. 2002. Design and Experience: Using the Intel Itanium-2 Processor Performance Monitoring Unit to Implement Feedback Optimizations, *2nd Workshop on Explicitly Parallel Instruction Computing Architecture and Compilers* (November).

Chou, Y. and J. Shen. 2000. Instruction Path Coprocessors, *Proc. 27th Int. Symp. on Computer Architecture* (June), pp. 270–281.

Cifuentes, C. and M. V. Emmerik. 2000. UQBT: Adaptable Binary Translation at Low Cost, *IEEE Computer* (March), pp. 60–66.

Cifuentes, C., B. Lewis, and D. Ung. 2002. Walkabout—A Retargetable Dynamic Binary Translation Framework, *Proc. Workshop on Binary Translation* (September).

Cmelik, B. and D. Keppel. 1994. Shade: A Fast Instruction-Set Simulator for Execution Profiling, *ACM Sigmetrics* (May), pp. 128–137.

Cmelik, R. F. and D. Keppel. 1996. Shade: A Fast Instruction-Set Simulator for Execution Profiling, *Technical Report UWCSE 93-06-06,* University of Washington, Seattle (June).

Cohen, D. 1981. On Holy Wars and a Plea for Peace, *IEEE Computer* (October), pp. 48–54.

Cohn, R. S., D. W. Goodwin, P. G. Lowney, and N. Rubin. 1997. Spike: An Optimizer for Alpha/NT Executables, *USENIX Windows NT Workshop* (August), pp. 17–23.

Collins, G. E. 1960. A Method for Overlapping and Erasure of Lists,

Communications of the ACM (December), pp. 655–657.

Comfort, W. T. 1964. Multiword List Items, *Communications of the ACM* (June), pp. 357–362.

Conte, T. M. and S. Sathaye. 1995. Dynamic Rescheduling: A Technique for Object Code Compatibility, *Proc. 28th Int. Symp. on Microarchitecture* (November), pp. 208–217.

Conte, T. M., K. N. Menezes, and M. A. Hirsch. 1996. Accurate and Practical Profile-Driven Compilation Using the Profile Buffer, *Proc. 29th Int. Symp. on Microarchitecture* (November), pp. 36–45.

Conte, T., S. Sathaye, and S. Banerjia. 1996. A Persistent Rescheduled-Page Cache for Low Overhead Object Code Compatibility in VLIW Architectures, *Proc. 29th Int. Symp. on Microarchitecture* (December), pp. 4–13.

Cooper, K. and L. Torczon. 2003. *Engineering a Compiler,* Morgan-Kaufmann, San Francisco.

Creasy, R. J. 1981. The Origin of the VM/370 Time-Sharing System, *IBM Journal of Research and Development* (September), pp. 483–490.

Culler, D. E. and J. P. Singh. 1999. *Parallel Computer Architecture: A Hardware/Software Approach,* Morgan-Kaufmann, San Francisco.

Cytron, R., J. Ferrante, B. K. Rosen, M. N. Wegman, and F. K. Zadeck. 1991. Efficiently Computing Static Single Assignment Form and the Control Dependence Graph, *ACM Transactions on Programming Languages and Systems* (October), pp. 451–490.

Czajkowski, K., I. Foster, N. Karonis, C. Kesselman, S. Martin, W. Smith, and S. Teucke. 1998. A Resource Management Architecture for Meta-computing Systems, *Proc. 4th Workshop on Job Scheduling Strategies for Parallel Processing* (March).

Dean, J., J. Hicks, C. Waldspurger, W. Weihl, and G. Chrysos. 1997. ProfileMe: Hardware Support for Instruction-Level Profiling on Out-of-Order Processors, *Proc. 30th Int. Symp. on Microarchitecture* (November), pp. 292–302.

Deaver, D., R. Gorton, and N. Rubin. 1999. Wiggins/Redstone: An Online Program Specializer, *Proc. 11th HotChips Symposium* (June).

Debaere, E. H. and J. M. Van Campenhout. 1990. *Interpretation and Instruction Path Coprocessing*, MIT Press, Cambridge, MA.

Dehnert, J. C., B. K. Grant, J. P. Banning, R. Johnson, T. Kistler, A. Klaiber, and J. Mattson. 2003. The Transmeta Code Morphing Software: Using Speculation, Recovery, and Adaptive Retranslation to Address Real-Life Challenges, *Proc. Int. Symp. on Code Generation and Optimization* (March), pp. 15–24.

Desoli, G., N. Mateev, E. Duesterwald, P. Faraboschi, and J. A. Fisher. 2002. DELI: A New Run-Time Control Point, *Proc. 35th Int. Symp. on Microarchitecture* (November), pp. 257–268.

Deutsch, P. and A. M. Schiffman. 1984. Efficient Implementation of the Smalltalk-80 System, *Proc. 11th ACM Symp. on Principles of Programming Languages* (January), pp. 297–302.

Dewar, R. B. K. 1975. Indirect Threaded Code, *Communications of the ACM* (June), pp. 330–331.

DHCP. Dynamic Host Configuration Protocol, **http://www.dhcp.org**.

Diffie, W. and M. E. Hellman. 1976. New Directions in Cryptography, *IEEE Trans. on Information Theory* (November), pp. 644–654.

Doran, R. W. 1988. Amdahl Multiple Domain Architecture, *IEEE Computer* (October), pp. 20–28.

Computer (October), pp. 20–28.

Dorward, D., R. Pike, D. L. Presotto, D. Ritchie, H. Trickey, and P. Winterbottom. 1997. Inferno, *Proc. IEEE Compcon '97.*

Drayton, P., B. Albahari, and T. Neward. 2002. *C# in a Nutshell,* 1st ed., O'Reilly, Sebastopol, CA.

Duesterwald, E. and V. Bala. 2000. Software Profiling for Hot Path Prediction: Less Is More, *Proc. 9th Int. Conf. on Architectural Support for Programming Languages and Operating Systems* (November), pp. 202–211.

Dunlap, G. W., S. T. King, S. Cinar, M. A. Basrai, and P. M. Chen. 2002. ReVirt: Enabling Intrusion Analysis Through Virtual-Machine Logging and Replay, *Proc. 5th Symp. on Operating Systems Design and Implementation* (December), pp. 211–224.

Ebcioglu, K., E. R. Altman, M. Gschwind, and S. Sathaye. 2001. Dynamic Binary Translation and Optimization, *IEEE Transactions on Computers* (June), pp. 529–548.

Engebretson, D., M. Corrigan, and P. Bergner. 2001. PowerPC 64-bit Kernel Internals, *Proc. 2001 Ottawa Linux Symposium* (July), **http://lwn.net/2001/features/OLS/pdf/pdf/ppc64.pdf**.

Ertl, M. A. and D. Gregg. 2001. The Behavior of Efficient Virtual Machine Interpreters on Modern Architectures, *Europar 2001,* pp. 403–412.

Ertl, M. A. and D. Gregg. 2003. Optimizing Indirect Branch Prediction Accuracy in Virtual Machine Interpreters, *Proc. Conf. on Programming Language Design and Implementation* (May) pp. 278–288.

Fahs, B., S. Bose, M. Crum, B. Slechta, F. Spadini, T. Tung, S. J. Patel,

and S. S. Lumetta. 2001. Performance Characterization of a Hardware Mechanism for Dynamic Optimization, *Proc. 34th Int. Symp. on Microarchitecture* (December), pp. 16–27.

Figueiredo, R., P. Dinda, and J. Fortes. 2003. A Case for Grid Computing on Virtual Machines, *Proc. 23rd Int. Conf. on Distributed Computing Systems* (May).

Fink, S. J. and F. Qian. 2003. Design, Implementation and Evaluation of Adaptive Recompilation with On-Stack Replacement, *Proc. Int. Symp. on Code Generation and Optimization* (March), pp. 241–252.

Fisher, J. A. 1981. Trace Scheduling: A Technique for Global Microcode Compaction, *IEEE Transactions on Computers* (July), pp. 478–490.

Flanagan, D. 1999. *Java in a Nutshell,* 3rd ed., O'Reilly, Sebastopol, CA.

FLEX. FLEX-ES: The New Mainframe, Fundamental Software Inc., **http://www.funsoft.com**.

Foster, I. and N. T. Karonis. 1998. A Grid-Enabled MPI: Message Passing in Heterogeneous Distributed Computing Systems, *Proc. Intl. Conf. ACM/IEEE on Supercomputing* (November).

Foster, I. and C. Kesselman (Eds.). 1998. *The Grid: Blueprint for a Future Computing Infrastructure,* Morgan Kaufmann, San Francisco.

Foster, I. and C. Kesselman (Eds.). 2004. *The Grid 2: Blueprint for a Future Computing Infrastructure,* Morgan Kaufmann, San Francisco.

Foster, I., C. Kesselman, and S. Tuecke. 2001. The Anatomy of a Grid: Enabling Virtual Scalable Organizations, *Int. Journal of High Performance Computing Applications,* pp. 200–222.

Fujitsu. 2003. The Next-Generation Engine for Consolidation, **http://www.computers.us.fujitsu.com/www/content/aboutus/analysts/data/** FujitsuGroup_ConsolidationEngine.pdf.

Gallagher, D., W. Y. Chen, S. A. Mahlke, J. C. Gyllenhaal, and W. W. Hwu. 1994. Dynamic Memory Disambiguation Using the Memory Conflict Buffer, *Proc. 6th Int. Conf. on Architectural Support for Programming Languages and Operating Systems* (October), pp. 183–193.

Garfinkel, T. and M. Rosenblum. 2003. A Virtual Machine Introspection Based Architecture for Intrusion Detection, *Proc. Symp. of the Internet Society's 2003 on Network and Distributed System Security* (February).

Garfinkel, T., B. Pfaff, J. Chow, M. Rosenblum, and D. Boneh. 2003. Terra: A Virtual Machine-Based Platform for Trusted Computing, *Proc. 19th Symp. on Operating System Principles* (October).

Gharachorloo, K., D. Lenoski, J. Laudon, P. Gibbons, A. Gupta, and J. Hennessy. 1990. Memory Consistency and Event Ordering in Scalable Shared-Memory Multiprocessors, *Proc. 17th Int. Symp. on Computer Architecture* (May), pp. 15–26.

Gill, S. 1951. The Diagnosis of Mistakes in Programmes on the *EDSAC, Proc. of the Royal Society Series A, Mathematical and Physical Sciences* (May), pp. 538–554.

Globus. The Globus Alliance, **http://www.globus.org**.

Goldberg, R. P. 1972. Architectural Principles for Virtual Computer Systems, Ph.D. dissertation, Harvard University, Cambridge, MA.

Goldberg, R. P. 1974. Survey of Virtual Machine Research, *IEEE Computer* (June), pp. 34–45.

Gong, L., G. Ellison, and M. Dageforde. 2003. *Inside Java 2 Platform Security: Architecture, API Design, and Implementation,* 2nd ed., Addison-Wesley, Reading, MA.

Gosling, J., B. Joy, and G. Steele. 1996. *The Java Language Specification,* Addison Wesley, Reading, MA.

Govil, K., D. Teodosiu, Y. Huang, and M. Rosenblum. 1999. Cellular Disco: Resource Management Using Virtual Clusters on Shared-Memory Multiprocessors, *Proc. 17th ACM Symp. on Operating System Principles* (December), NC, pp. 154–169.

Grant, B., M. Philipose, M. Mock, S. J. Eggers, and C. Chambers. 1999. An Evaluation of Run-Time Optimizations, *Proc. Conf. on Programming Language Design and Implementation* (May), pp. 293–304.

Gschwind, M. 1998a. Method and Apparatus for Determining Branch Addresses in Programs Generated by Binary Translation, *IBM Research Disclosures YOR819980334* (July).

Gschwind, M. 1998b. Method and Apparatus for Rapid Return Address Computation in Binary Translation, *IBM Research Disclosures YOR819980410* (September).

Gschwind, M. and E. R. Altaian. 2000. Optimization and Precise Exceptions in Dynamic Compilation, *Proc. Workshop on Binary Translation* (October).

Gschwind, M., K. Ebcioglu, E. R. Altman, and S. Sathaye. 2000. Binary Translation and Architecture Convergence Issues for IBM System/390, *Proc. Int. Conf. on Supercomputing* (May), pp. 336–347.

Gum, P. H. 1983. System/370 Extended Architecture: Facilities for Virtual Machines, *IBM Journal of Research and Development*

(November), pp. 530–544.

Gupta, M., J.-D. Choi, and M. Hind. 2000. Optimizing Java Programs in the Presence of Exceptions, *14th European Conference on Object-Oriented Programming* (June), pp. 422–446.

Halfhill, T. R. 2000. Transmeta Breaks x86 Low-Power Barrier, *Microprocessor Report* (February 14), pp. 9–18.

Hall, J. S. and P. T. Robinson. 1990. Virtualizing the VAX Architecture, *Proc. USENIX1990 Summer Conference* (June), pp. 380–389.

Hansen, G. J. 1974. Adaptive Systems for the Dynamic Run-Time Optimization of Programs, Ph.D. dissertation, Carnegie-Mellon University, Pittsburgh, PA.

Hazelwood, K. and J. E. Smith. 2004. Exploring Code Cache Eviction Granularities in Dynamic Optimization Systems, *Proc. Second Annual IEEE/ACM Int. Symp. on Code Generation and Optimization* (March), pp. 89–99.

Heil, T. and J. E. Smith. 2000. Relational Profiling: Enabling Thread-Level Parallelism in Virtual Machines, *Proc. 33rd Int. Symp. on Microarchitecture* (December), pp. 281–290.

Hennessy, J. and D. Patterson. 2002. *Computer Architecture: A Quantitative Approach,* 3rd ed., Morgan Kaufmann, San Francisco.

Hennessy, J., M. Heinrich, and A. Gupta. 1999. Cache-Coherent Distributed Shared Memory: Perspectives on its Development and Future Challenges, *Proceedings of the IEEE* (March), pp. 418–429.

Hewlett-Packard. 2000. HP Partitioning Continuum, *Technical Positioning White Paper* (June), **http://h30081.www3.hp.com/products/wlm/docs/HPPartitioningContinuum.pdf**.

Hinton, G., D. Sager, M. Upton, D. Boggs, D. Carmean, A. Kyker, and P. Roussel. 2001. The Microarchitecture of the Pentium 4 Processor, *Intel Technology Journal* (Ql).

Hohensee, P., M. Myszewski, and D. Reese. 1996. Wabi CPU Emulation, *Proc. 8th HotChips Symposium* (August), pp. 47–65.

Hölzle, U. and D. Ungar. 1996. Reconciling Responsiveness with Performance in Pure Object-Oriented Languages, *ACM Transactions on Programming Languages and Systems* (July), pp. 355–400.

Hölzle, U., C. Chambers, and D. Ungar. 1991. Optimizing Dynamically Typed Object-Oriented Languages with Polymorphic Inline Caches, *5th European Conference on Object-Oriented Programming* (July), pp. 21–38.

Hölzle, U., C. Chambers, andD. Ungar. 1992. Debugging Optimized Code with Dynamic Deoptimization, *Proc. Conf. on Programming Language Design and Implementation* (July), pp. 32–43.

Hookway, R. J. and M. A. Herdeg. 1997. Digital FX!32: Combining Emulation and Binary Translation, *Digital Technical Journal* (January), pp. 3–17.

Horspool, R. N. and N. Marovac. 1980. An Approach to the Problem of Detranslation of Computer Programs, *Computer Journal* (August), pp. 223–229.

Höxer, H.-J., K. Buchacker, and V. Sieh. 2002. UMLinux—A Tool for Testing a Linux System's Fault Tolerance, *Proc. LinuxTag* (June).

Hsu P.-T. and E. Davidson. 1986. Highly Concurrent Scalar Processing, *Proc. 13th Int. Symp. on Computer Architecture* (June), pp. 386–395.

Hunt, G., and D. Brubacher. 1999. Detours: Binary Interception of Win32 Functions, *Proc. 3rd USENIX Windows NT Symposium* (July), pp. 135–143.

Hwu W. W., S. A. Mahlke, W. Y. Chen, P. P. Chang, N. J. Warter, R. A. Bringmann, R. G. Ouellette, R. E. Hank, T. Kiyohara, G. E. Haab, J. G. Holm, and D. M. Lavery. 1993. The Superblock: An Effective Technique for VLIW and Superscalar Compilation, *Journal of Supercomputing* (May) pp. 229–248.

IBM. 1994. *The PowerPC Architecture,* Morgan Kaufmann, San Francisco.

IBM. 2001. *Autonomic Computing: IBM's Perspective on the State of Information Technology,* Armonk, NY (October).

Intel. 1999. *Intel(R) Architecture Software Developer's Manual, Volume 2: Instruction Set Reference Manual,* **http://www.intel.com/design/intarch/manuals/243191.htm**.

Intel. 2005. *Intel Vanderpool Technology for IA-32 Processors (VT-x) Preliminary Specification,* Order Number C97063-001, Santa Clara, CA (January).

Jacob, B. and T. Mudge. 1998. Virtual Memory: Issues of Implementation, *IEEE Computer* (June), pp. 33–43.

Jann, J., L. M. Browning, and R. S. Burugula. 2003. Dynamic Reconfiguration: Basic Building Blocks for Autonomic Computing on IBM pSeries Servers, *IBM Systems Journal* (January), pp. 29–37.

Jones, R. 1996. Garbage Collection: Algorithms for Automatic Dynamic Memory Management, Wiley, New York.

Kaeli, D. and P. G. Emma. 1991. Branch History Table Prediction of Moving Target Branches Due to Subroutine Returns, *Proc. 18th Int.*

Symp. on Computer Architecture (June), pp. 34–42.

Kane, G. 1996. *PA-RISC Architecture,* Prentice Hall, New York.

Keefe, D. D. 1968. Hierarchical Control Program for System Evaluation, *IBM Systems Journal,* pp. 123–133.

Kelly, E. J., R. F. Cmelik, and M. J. Wing. 1998. Memory Controller for a Microprocessor for Detecting a Failure of Speculation on the Physical Nature of a Component Being Addressed, US patent 5,832,205 (November).

Keltcher, C. N., K. J. McGrath, A. Ahmed, and P. Conway. 2003. The AMD Opteron Processor for Multiprocessor Servers, *IEEE Micro* (March/April), pp. 66–76.

Kim, H.-S. and J. E. Smith. 2003. Dynamic Binary Translation for Accumulator-Oriented Architectures, *Proc. Int. Symp. on Code Generation and Optimization* (March), pp. 25–35.

Kiriansky, V., D. Bruening, and S. Amarasinghe. 2002. Secure Execution via Program Shepherding, *Proc. 11th USENIX Security Symposium* (August).

Kistler, T. and M. Franz. 2000. Automated Data-Member Layout of Heap Objects to Improve Memory-Hierarchy Performance, *ACM Transactions on Programming Languages and Systems* (May), pp. 490–505.

Kistler, T. and M. Franz. 2001. Continuous Program Optimization: Design and Evaluation, *IEEE Transactions on Computers* (June), pp. 549–566.

Klaiber, A. 2000. The Technology Behind Crusoe Processors, *Transmeta Technical Brief.*

Klint, P. 1981. Interpretation Techniques, *Software Practice and Experience* (September), pp. 963–973.

Kogge, P. M. 1982. An Architectural Trail to Threaded-Code Systems, *IEEE Computer* (March), pp. 22–34.

Kozuch M. and M. Satyanarayanan. 2002. Internet Suspend/Resume, *Proc. 4th IEEE Workshop on Mobile Systems and Applications* (June).

Lam, M. 1988. Software Pipelining: An Effective Scheduling Technique for VLIW Machines, *Proc. ACM Conf. on Programming Languages Design and Implementation* (June), pp. 318–328.

Lamport, L. 1979. How to Make a Multiprocessor Computer That Correctly Executes Multiprocess Programs, *IEEE Transactions on Computers* (September), pp. 690–691.

Larus, J. 1991. *SPIM S20: A MIPS R2000 Simulator*, University of Wisconsin—Madison Technical Report, **http://www.cs.wisc.edu/~larus/spim.html**.

Larus, J. R. and E. Schnarr. 1995. EEL: Machine-Independent Executable Editing, *Proc. ACM SIGPLAN Conf. on Programming Language Design and Implementation* (June), pp. 291–300.

Lawton, K. The Bochs IA-32 Emulator Project, **http://bochs.sourceforge.net**.

Le, B. C. 1998. An Out-of-Order Execution Technique for Runtime Binary Translators, *Proc. 8th Int. Conf. on Architectural Support for Programming Languages and Operating Systems,* pp. 151–158.

Lidin, S. 2002. *Inside Microsoft .NET IL Assembler,* Microsoft Press, Redmond, WA.

Lindholm, T. and F. Yellin. 1999. *The Java Virtual Machine Specification,* 2nd ed., Addison-Wesley, Reading, MA.

Lowney, P. G., S. M. Freudenberger, T. J. Karzes, W. D. Lichtenstein, R. P. Nix, J. S. O'Donnell, and J. Ruttenberg. 1993. The Multiflow Trace Scheduling Compiler, *Journal of Supercomputing* (May), pp. 51–142.

Lu, J., H. Chen, R. Fu, W.-C. Hsu, B. Othmer, P.-C. Yew, and D.-Y. Chen. 2003. The Performance of Runtime Data Cache Prefetching in a Dynamic Optimization System, *Proc. 36th IEEE/ACM Int. Symp. on Microarchitecture* (December), pp. 180–190.

Lu, J., H. Chen, P.-C. Yew, and W.-C. Hsu. 2004. Design and Implementation of a Lightweight Dynamic Optimization System, *Journal of Instruction-Level Parallelism,* Vol. 6, pp. 1–24.

Lucco, S., O. Sharp, and R. Wahbe. 1995. Omniware: A Universal Substrate for Web Programming, *World Wide Web Journal* (winter), pp. 359–368.

MacKinnon, R. A. 1979. The Changing Virtual Machine Environment: Interfaces to Real Hardware, Virtual Hardware, and Other Virtual Machines, *IBM Systems Journal,* pp. 18–46.

Madnick, S. E. and J. J. Donovan. 1974. *Operating Systems,* McGraw-Hill, New York.

Magnusson, P. and D. Samuelsson. 1994. A Compact Intermediate Format for SIMICS, *Tech. Report R94:17,* Swedish Institute of Computer Science (September).

Mahlke, S. A., D. C. Lin, W. Y. Chen, R. E. Hank, and R. A. Bringmann. 1992. Effective Compiler Support for Predicated Execution Using the Hyperblock, *Proc. 25th Int. Symp. on Microarchitecture* (November), pp. 45–54.

May, C. 1987. Mimic: A Fast System/370 Simulator, *Conf. on Programming Language Design and Implementation,* archive, pp. 1–13.

McGhan, H. and M. O'Connor. 1998. Picojava: A Direct Execution Engine for Java Bytecode, *IEEE Computer* (October), pp. 22–30.

McGraw, B. and E. W. Felten, 1999. *Securing Java: Getting Down to Business with Mobile Code,* 2nd ed., Wiley, New York.

Meloan, S. 1999. The Java HotSpot Performance Engine: An In-Depth Look (June), **http://java.sun.com/developer/technicalArticles/Networking/HotSpot**.

Merten, M. C, A. R. Trick, E. M. Nystrom, R. D. Barnes, and W. W. Hwu. 2000. A Hardware Mechanism for Dynamic Extraction and Relayout of Program Hotspots, *Proc. 27th Int. Symp. on Computer Architecture* (June), pp. 59–70.

Merten, M. C, A. R. Trick, R. D. Barnes, E. M. Nystrom, C. N. George, J. C. Gyllenhaal, and W. W. Hwu. 2001. An Architectural Framework for Run-Time Optimization, *IEEE Transactions on Computers* (June), pp. 567–589.

Meyer, R. A. and L. H. Seawright. 1970. A Virtual Machine Time-Sharing System, *IBM Systems Journal,* pp. 199–218.

Myers, G. J. 1982. *Advances in Computer Architecture,* 2nd ed., Wiley, New York.

Nair, R. and M. E. Hopkins. 1997. Exploiting Instruction Level Parallelism in Processors by Caching Scheduled Groups, *Proc. 24th Int. Symp. on Computer Architecture* (June), pp. 13–25.

Nethercote, N., and J. Seward. 2003. Valgrind: A Program

Supervision Framework, *Proc. 3rd Workshop on Runtime Verification* (July).

Nick, J. M., B. B. Moore, J.-Y. Chung, and N. S. Bowen. 1997. S/390 Cluster Technology: Parallel Sysplex, *IBM Systems Journal,* pp. 172–201.

NIST. 2002. FIPS 180–2, Secure Hash Standard (SHS) (August).

Nori, K. V., U. Ammann, K. Jensen, H. H. Nageli, and C. Jacobi. 1975. *The Pascal P-Compiler: Implementation Notes* (rev. ed.), Institut fur Informatik ETH, Zurich, Tech. Rep. 10.

Nystrom, E., R. D. Barnes, M. C. Merten, and W. W. Hwu. 2001. Code Reordering and Speculation Support for Dynamic Optimization Systems, *Proc. Int. Conf. on Parallel Architectures and Compilation Techniques* (September), pp. 163–174.

OGSA. Open Grid Services Architecture Working Group, **https://forge.gridforum.org/projects/ogsa-wg**.

OpenSSH. OpenSSH, **http://www.openssh.com**.

Osisek, D. L., K. M. Jackson, and P. H. Gum. 1991. ESA/390 Interpretive Execution Architecture, Foundation for VM/ESA, *IBM Systems Journal,* pp. 34–51.

Pacheco, P. 1996. *Parallel Programming with MPI,* Morgan-Kaufmann, San Francisco.

Paleczny, M., C. Vick, and C. Click. 2001. The Java Hotspot Server Compiler, *USENIX Java Virtual Machine Research and Technology Symp.* (April), pp. 1–12.

Parmalee, R. P., T. I. Peterson, C. C. Tillman, and D. J. Hatfield. 1972. Virtual Storage and Virtual Machine Concepts, *IBM Systems Journal,*

pp. 99–219.

Patel, S. J., S. S. Lumetta. 2001. rePLay: A Hardware Framework for Dynamic Optimization, *IEEE Transactions on Computers* (June), pp. 590–608.

Perkins, C. E. 1998. Mobile Networking Through Mobile IP, *IEEE Internet Computing* (January), pp. 58–69.

Pettis, K. and R. C. Hansen. 1990. Profile Guided Code Positioning, *Proc. ACM SIGPLAN Conf. on Programming Language Design and Implementation* (June), pp. 16–27.

Popek, G. J. and R. P. Goldberg. 1974. Formal Requirements for Virtualizable Third-Generation Architectures, *Communications of the ACM* (July), pp. 412–421.

Richardson T., Q. Stafford-Fraser, K. R. Wood, and A. Hopper. 1998. Virtual Network Computing, *IEEE Internet Computing* (January), pp. 33–38.

Ridge, D., D. Becker, P. Merkey, and T. Sterling. 1997. Beowulf: Harnessing the Power of Parallelism in a Pile-of-PCs, *Proc. IEEE Aerospace Conference* (February), pp. 79–91.

Robin, J. S. and C. E. Irvine. 2000. Analysis of the Intel Pentium's Ability to Support a Secure Virtual Machine Monitor, *Proc. 9th USENIX Security Symposium,* Denver (August).

Romer, T. H., D. Lee, G. M. Voelker, A. Wolman, W. A. Wong, J.-L. Baer, B. N. Bershad, and H. M. Levy. 1996. The Structure and Performance of Interpreters, *Proceedings 7th Int. Conf. on Architectural Support for Programming Languages and Operating Systems* (September), pp. 150–159.

Ronsse, M. and K. De Bosschere. 2000. JiTI: A Robust Just-in-Time

Instrumentation Technique, *Proc. Workshop on Binary Translation* (October).

Rosenblum, M. 2000. VMware's Virtual Platform Technology, *Multi-University/Research Laboratory Seminar Series,* **http://murl.microsoft.com/LectureDetails.asp?530** (December).

Rotenberg, E., S. Bennett, and J. E. Smith. 1996. Trace Cache: A Low-Latency Approach to High-Bandwidth Instruction Fetching, *Proc. 29th Int. Symp. on Microarchitecture* (December), pp. 24–35.

Sapuntzakis, C. P., R. Chandra, B. Pfaff, J. Chow, M. S. Lam, and M. Rosenblum. 2002. Optimizing the Migration of Virtual Computers, *Proc. 5th Symp. on Operating Systems Design and Implementation* (December).

Sathaye, S., P. Ledak, J. LeBlanc, S. Kosonocky, M. Gschwind, J. Fritts, Z. Filan, A. Bright, D. Appenzeller, E. Altman, and C. Agricola. 1999. BOA: Targeting Multi-Gigahertz with Binary Translation, *Proc. Workshop on Binary Translation* (October).

Scheifler, R. W. 1977. An Analysis of Inline Substitution for a Structured Programming Language, *Communications of the ACM* (September), pp. 647–654.

Scott, K., N. Kumar, S. Velusamy, B. R. Childers, J. W. Davidson, and M. L. Sofia. 2003. Retargetable and Reconfigurable Software Dynamic Translation, *Proc. Int. Symp. on Code Generation and Optimization* (March), pp. 36–47.

Shannon, B., M. Hapner, V. Matena, E. Pelegri-Llopart, J. Davidson, and L. Cable. 2000. *Java 2 Platform, Enterprise Edition: Platform and Component Specifications,* Addison-Wesley, Reading, MA.

Sites, R. L., A. Chernoff, M. B. Kerk, M. P. Marks, and S. G. Robinson.

1993. Binary Translation, *Communications of the ACM* (February), pp. 69–81.

Smith, M. D., M. S. Lam, and M. A. Horowitz. 1990. Boosting Beyond Static Scheduling in a Superscalar Processor, *Proc. 17th Int. Symp. on Computer Architecture* (June), pp. 344–354.

Soltis, F. G. 1996. *Inside the AS/400,* Duke Press, Loveland, CO.

SPEC. Standard Performance Evaluation Corporation, **www.spec.org**.

Stichnoth, J., G.-Y. Lueh, and M. Cuerniak. 1999. Support for Garbage Collection at Every Instruction in a Java Compiler, *Proc. Conf. on Programming Language Design and Implementation* (May), pp. 118–127.

Suganuma, T., T. Yasue, and T. Nakatani. 2002. An Empirical Study of Method In-Lining for a Java Just-in-Time Compiler, *Usenix Java Virtual Machine Research and Technology Symp.* (August), pp. 91–104.

Suganuma, T., T. Ogasawara, M. Takeuchi, T. Yasue, M. Kawahito, K. Ishizaki, H. Komatsu, and T. Nakatani. 2000. Overview of the IBM Java Just-in-Time Compiler, *IBM Systems Journal* (February), pp. 175–193.

Suganuma, T., T. Yasue, M. Kawahito, H. Komatsu, and T. Nakatani. 2001. A Dynamic Optimization Framework for a Java Just-in-Time Compiler, *Proc. Conf. on Object Oriented Programming, Systems, Languages and Applications* (November), pp. 180–195.

Sugerman, J., G. Venkitachalam, and B.-H. Lim. 2001. Virtualizing I/O Devices on VMware Workstation's Hosted Virtual Machine Monitor, *Proc. USENIX Annual Technical Conference* (June), pp. 1–14.

Sun Microsystems. 1999. Sun Enterprise 10000 Server: Dynamic Systems Domains, *Technical White Paper* **http://www.sun.com/data-center/docs/domainswp.pdf**.

Sunderaraj, A. I. and P. A. Dinda. 2004. Towards Virtual Networks for Virtual Machine Grid Computing, *Proc. 3rd USENIX Virtual Machine Technology Symposium* (May).

Tabatabai, A., M. Cierniak, G.-Y. Lueh, V. M. Parikh, and J. M. Stichnoth. 1998. Fast Effective Code Generation in a Just-In-Time Java Compiler, *Proc. Conf. on Programming Language Design and Implementation* (June), pp. 280–290.

Tamches, A. and B. P. Miller. 1999. Fine-Grained Dynamic Instrumentation of Commodity Operating System Kernels, *Proc. 3rd Symp. on Operating Systems Design and Implementation* (February), pp. 117–130.

Traut, E. 1997. Building the Virtual PC, *Byte* (November), pp. 51–52.

Varian, M. 1997. VM and the VM Community: Past, Present, and Future, *SHARE 89* (August).

Venners, B. 1998. *Inside the Java Virtual Machine,* McGraw-Hill, New York.

VirtualCenter. *VMware VirtualCenter Users Manual, version 1.2,* VMware Inc., Palo Alto, CA, **http://www.vmware.com/pdf/VC_Users_Manual_ll.pdf**.

VMware. 2001. *VMware GSX Server User Manual, version 1.0,* VMware Inc., Palo Alto, CA.

VMware. 2000. VMware Virtual Platform Technical White Paper, **http://ftp.hit.edu.cn/linux/Datas/Docs/webmirror/**

www.vmware.com/virtualplatform.html.

Wahbe, R., S. Lucco, T. E. Anderson, and S. L. Graham. 1993. Efficient Software-Based Fault Isolation, *ACM SIGOPS Operating Systems Review* (December), pp. 203–216.

Wallach, D. and E. Felten. 1998. Understanding Java Stack Inspection, *Proc. IEEE Symp. on Security and Privacy* (May), pp. 52–63.

Whaley, J. 2001. Partial Method Compilation Using Dynamic Profile Information, *Proc. Conf. on Object-Oriented Programming Systems, Languages, and Applications* (October), pp. 166–179.

Whitaker, A., M. Shaw, and S. D. Gribble. 2002. Denali: Lightweight Virtual Machines for Distributed and Networked Applications, *Technical Report 02-02-01,* University of Washington, Seattle.

Wilson, P. R. 1992. Uniprocessor Garbage Collection Techniques, *Proc. Int. Workshop on Memory Management* (September), pp. 1–42.

Wilson, P. R., M. S. Johnstone, M. Neely, and D. Boles. 1995. Dynamic Storage Allocation: A Survey and Critical Review, *Proc. Int. Workshop on Memory Management.*

Zheng, C. and C. Thompson. 2000. PA-RISC to IA-64: Transparent Execution, No Recompilation, *IEEE Computer* (March), pp. 47–53.

Previous chapter
< Appendix A: Real Machines

Next chapter
Index >

INDEX

Abstraction, levels of, 1
Access flags, Java, 258
Addressing architecture, 77
ADORE system, 213
Amdahl systems, logical partitioning, 458, 465
Applets, 233
Application binary interface (ABI), 8, 83, 583–584
Application domains, 275
Application program interface (API), 8–9, 583
Java, 261–267
Application servers, 447
Architecture, 6–9
addressing, 77
register, 69–70
Aries system, 14, 118
Arithmetic, emulating, 74–75, 565
Array range checking, 318–320
Assists,
instruction emulation, 417–418
virtual machine monitor 418–419
Attributes
Java, 259
Microsoft common language infrastructure, 270–271
Backing store, 571
Backpointers, code cache, 135, 138
Basic block
dynamic versus static, 56–57
profile, 160
Big-endian byte order, 76
Binary classes, Java, 256–259
access flags, 258
attributes, 259
constant pool, 244–245, 258
field count and information, 259
interfaces, 258

methods count and information, 259
Super_Class, 258
This_Class, 258
Binary optimization, dynamic approach to, 180–181
backward and forward branches, 154
code optimizations, 201–208
code reordering, 186–201
common subexpression elimination, 147
compatibility, 181–184
compensation code, 150
control flow, 153–154, 159
data value predictability, 155
indirect jumps, 155
profiling, 148–150, 156–167
program behavior, 153–156
same-ISA, 15, 29, 208–218
staged emulation, 151–153
superblocks, 150, 173–178
techniques, 147–148
traces, 171–173
translation blocks, 167–180
tree groups, 178–180
Binary optimizers, 15
Binary rewriting, 515–519
Binary translation, 29, 49–52
incremental predecoding and, 55–62, 80, 82
locating program counter, 124–126
memory state, 127–128
precise guest state, determining, 124–128
register state, 126–127
Binary translators, dynamic, 13–15
Bootstrapping, 586–588
brk(), 129
Buffer maintenance, 48–49
Buffer overflow, 502–503
Bytecodes, 17
Java, 232, 246
MSIL, 271–273
Byte order, 76
Cache memory, 558–559

Callbacks, 130–132
Call graph, 309
Calling sequence, 309
Cellular Disco System, 475–476
fault containment, 480–481
memory borrowing, 482–483
memory mapping, 478–480
overview of, 477–478
recovery from faults, 483–485
Chaining, code cache, 64–66
Channels and channel command words, 414–415
Channel-to-channel adapter (CTCA), 421
Checkpoints, 194–195, 345–347
Child process, 584
CISC ISAs, 36, 38–49, 125, 329
Class loader subsystem, 284–286
clib, 9
Clustered computing, 590, 591–594
Code cache, 55, 85
chaining, 64–66
codesigned virtual machines and, 339–344
difference between conventional cache memory and, 134
FIFO, coarse-grained, 138–139
FIFO, fine-grained, 125, 138
flush when full, 136–137
implementations, 134
least recently used, 135–136
management, 63, 85, 133
performance, 139–140
preemptive flush, 137–138
replacement algorithms, 134–140
virtual machine monitor and, 393–395
Code discovery, 46, 52–54, 278
Code location problem, 54–55
Code optimizations
basic, 201–204
compatibility issues, 204–205
instruction-set-specific, 206–208
intersuperblock, 205–206
Code patching, 210, 211–213

Code reordering
branch instructions, 187, 188
condition code handling, 198–199
implementing a scheduling algorithm, 192–199
instructions, categories of, 187
join points, 187, 188, 190
memory update instructions, 187, 190
register update instructions, 187, 188–189
straight-line, 191–192
superblocks versus traces, 199–201
Codesigned virtual machines, 21–22
applications, 352–354
code caching, 339–344
early models, 330–331
IBM AS/400 system, 22, 330, 353, 357–367
IBM Daisy system, 334, 347, 353
input/output, 351–352
memory state mapping 333–337
precise traps, 344–351
role of, 332–333
self-referencing and -modifying code, 337–339
Transmeta Crusoe, 22, 24, 331, 338, 353, 354–357
Collective, Stanford, 526–532
Common language infrastructure (CLI), 17, 267–275
Common language runtime (CLR), 267
Common subexpression elimination, 147
Communication, inter-machine, 421–422
Compacting garbage collectors, 297–298
Compatibility
binary optimization and, 181–184
code optimization and, 204–205
extrinsic, 88, 184
intrinsic, 88, 184
process virtual machine, 87–95
state mapping and, 101–102
transparency, 88
Compensation code, 150, 196
Computation server, 549
Computer hardware, 554–561
Computing environments, migration of, 521–532

Concealed memory, 21–22, 334–337
Concurrent garbage collection, 300–302
Condition codes, 70–74, 198–199
Constant folding, 201
Constant pool, Java, 244–245, 258
Constant propagation, 201
Context block, 30
Continuation set, 175–176
Control flow, 153–154, 159
graph, 157, 160
Control Program (CP), 381–382, 395–396, 402, 414–415, 421
Control transfer
optimizations, 64–68
restricting, 516–517
Conversational monitor system (CMS), 370, 381
Copying garbage collectors, 299
Copy-on-write, 528–530
Copy propagation, 202–203
Counter decaying, 165
CPU timer, 384
CreateProcess, 141, 142
Critical instructions, 391, 392–393, 430–431
Daisy system, 334, 347, 353
Data formats, emulating, 74–75
Data server, 549
Data types, Java, 242–243
conversion, 249
Dead-assignment elimination, 202, 204
Decode-and-dispatch interpreter, 30, 80, 81
Device-driver interface, 408, 409–410
Devices, virtualizing
dedicated, 404–405
nonexistent physical devices, 407
partitioned, 405
shared, 405
spooled, 405–407
DIAGNOSE, 420
Digital Equipment Corp., FX!32
system, 14, 15, 46–49, 88, 118, 141, 142–145
Direct native execution, 382

Direct threaded interpretation, 37–38, 80, 81–82
Distributed shared memory (DSM), 590–591
Dual-address return address stack (DRAS), dual-address, 342–344
Dynamically linked library (DLL), 140, 142
Dynamic basic block, 56–57
Dynamic binary optimization. *See* Binary optimization, dynamic
Dynamic binary translators, 13–15
DynamoRIO system, 132, 139, 152, 209, 210, 214
See also RIO system
Dynamo system, 15, 112, 119, 129, 152, 209, 214–218
Edge profile, 161
Emulation, 12
basic, 305–306
binary translation and incremental predecoding, 55–62, 80, 82
decode-and-dispatch interpreter, 30, 80, 81
defined, 27
direct threaded interpretation, 37–38, 80, 81–82
engine, 85, 281, 283
exceptions, 86, 119–128
high-performance, 306–320
incremental predecoding and translation, 55–62
indirect threaded interpretation, 80, 81
manager (EM), 55, 58
memory architecture, 102–114
operating system, 128–133
same-ISA, 63–64
staged, 116–119, 151–153
Emulators, 13–15
Environments, migration of computing, 521–532
Epilog side table, 205–206
Escape analysis, 317
Event-driven modules, 511
Exception, use of term, 119
Exception emulation, 86
ABI invisible, 120
ABI visible, 119
exception detection, 120–121
interrupt handling, 121–122
precise guest state, determining, 122–128
Exceptions, precise, 277

Exceptions and errors, Java, 254–256
exec(), 141
Execution engine, 281, 283
Extrinsic compatibility, 88, 184
Fault(s)
containment, 480–481
recovery from, 483–485
Field count and information, Java, 259
FIFO, coarse-grained, 138–139
FIFO, fine-grained, 125, 138
File limits, enforcing, 133
Floating-point arithmetic, 74, 565
Flush-when-full, 136–137
Forth language, 29
Fujitsu, PrimePower systems, 456
Fully associative cache, 558–559
Fully qualified name, 284
Functional instructions, Java, 249–250
Fundamental Software Inc., FLEX-ES, 486
FX!32 system, 14, 15, 46–49, 88, 118, 141, 142–145
Garbage collection, 238–240, 283, 294–295
compacting, 297–298
copying, 299
generational, 300
incremental and concurrent, 300–302
mark-and-sweep, 296–297
optimizing, 320
root set, 302–303
summary of, 303
Generational garbage collectors, 300
getrlimit(), 133
Global memory, 243–244, 282
Globus project, 542–543
Grids
access issues, 540–541
characteristics of ideal, 539–542
compared with virtual machines, 543–546
computing model, evolution of, 537, 542–543
dependability, 539–540
evolution of, 535–538

implementing, on virtual machine systems, 546–551
resource sharing, 541–542, 544
use of, 546–551
Guard instructions, 311
Guests, 10, 28
improving performance, 419–422
integration of, into host system environment, 140–142
memory image, 85
precise guest state, determining, 122–128
state management, 375–377
virtualization with different host and, 485–496
Handle pool, 298
Handshaking, 420
Hardware, computer, 554–561
Hardware page table, 470
Hashed copy scheme, 531
Hazards, 597, 599
Heap-allocated objects, 316–318
Hewlett-Packard
-Compaq Superdome servers, 464, 465
Dynamo system, 15, 112, 119, 129, 152, 209, 214–218, 515
physical partitioning, 455–456
High-level language virtual machine (HLL VM), 15–17
High-level language virtual machine, architecture
common language interface, 267–275
ISA features, 275–279
Java, 241–261
object-oriented, 228–241
Pascal P-code, 225–228
platforms, 261–267
role of, 222–225
High-level language virtual machine, implementation
class loader, 284–286
components, 281–284
emulation, basic, 305–306
emulation, high-performance, 306–320
emulation/execution engine, 85, 281, 283
garbage collection, 238–240, 283, 294–303
memory, 281–282
native interface, 259–261, 283, 304–305

security, 286–294
Hitachi, logical partitioning, 458
Hoisting invariant checks, 319
Hoisting invariant expression out of a loop, 203–205
Host-based intrusion-detection systems (HIDS), 502–505
Hosted virtual machines, 379–381
input/output virtualization and, 412–414
VMware Virtual Platform, 426–435
Hosts, 10, 28
guest process integration into, 140–142
virtualization with different guest and, 485–496
Host-supported memory protection, 104–105
Hotspots, 159
Hybrid virtual machine system, 391
Hypervisors, 464, 474–475
services interface, 469–471
IBM
AS/400 system, 22, 330, 353, 357–367
Daisy system, 334, 347, 353
DK, 306
Interpretive Execution Facility (IEF), 424–426
iSeries servers, 464
Jikes Research Virtual Machine (RVM), 306, 320–327
PowerPC ISA, 35–36, 70, 74, 600–606
pSeries, 464, 600
System/38, 330, 357–358
System/360, 7–8, 381
System/370, 381–382, 395–396, 414–415, 427
System/390, logical partitioning, 458, 460–464, 473–474
zSeries, 465
IEEE floating point standard, 74
If conversion, 208
Image server, 549
Implementation
layers, 6–7
use of term, 6
Incremental garbage collection, 300–302
Incremental predecoding and translation, 55–62
Indirect jumps
binary optimization and, 155

map table lookups, 55–56
software prediction, 66–67, 340
Indirect threaded interpretation, 80, 81
Initialization
ISA system, 586–588
process virtual machine, 85
Inline caching, 66
Inlining
changing method calls, 310–312
guard instructions, 311
multiversioning and specialization, 312–313
partial procedure, 170
procedure, 169–170, 308–310
Input/output, codesigned virtual machines and, 351–352
IBM AS/400 system, 364–365
Input/output systems, 559–561
managing, 576–577, 582–583
system calls, 585
virtualization, 404–415, 431–435
Instruction checkpoints, 194–195
Instruction emulation, 114–119
assists, 417–418
Instruction set architecture (ISA), 7–8
examples, 600–611
resource management, 566–580
source, 13–14, 27–28
system, 8
target, 14, 27–28
user, 8, 561–566
virtual, 16–17
Instruction sets, 2
addressing architecture, 77
byte order, 76
condition codes, 70–74
conventional versus V-ISA, 278
data formats and arithmetic, 74–75, 565
IBM AS/400 system, 361–364
Java, 246–254
logically complete, 132
memory address resolution, 75

memory data alignment, 75–76
Pascal P-code, 227
register architectures, 69–70
Instrumentation-based profiling, 161–162
Intel IA-32, 427
arithmetic, 74
binary translation, 49–52
condition codes, 70, 71–74
description of, 606–611
EFLAGS, 71, 430, 431
high-performance interpretation, 46–49
interpretation of, 39–44
pop stack into flags register (POPF), 385–386, 391
threaded interpretation, 44–46
Intel IA-32 EL (execution layer), 14, 84, 119, 152
Intel IA-64 (IPF), 83, 186
Intel Itanium platforms, 83–84, 162
Intel VT-x (Vanderpool), 436–442
Interfaces
Java, 258
well-defined, 2–3
Internal Microprogrammed Interface (IMPI), 358
Internet Suspend/Resume (ISR), 524–526
Interpretation, 14, 29
basic, 29–32
of a complex instruction set, 38–49
decode-and-dispatch, 30, 80, 81
direct threaded, 37–38, 80, 81–82
indirect threaded, 80, 81
memory, 29–30
precise guest state, determining, 123
profiling during, 163–165
Interpretive Execution Facility (IEF), 424–426
Interpretive trap detection, 120
Interrupts
handling, 121–122, 577–580
hypervisor, 469
registers, 569
use of term, 119
Intersuperblock optimizations, 205–206

Intrinsic compatibility, 88, 184
Intrusion-detection systems (IDS), 502–505, 508–511
invoke instructions, 251
Itanium platforms, 83–84, 162
Java, 6
APIs, 262–263
bytecodes, 232, 246
constant pool, 244–245
garbage collection, 238–240, 283, 294–303
global memory, 244, 282
memory hierarchy, 245–246
objects, 243
platforms, 232, 261–262
reference type, 242
security, 286–294
stack, 243–244, 282
stack tracking, 251–254
threads, 265–267
java.awt, 263
JavaBeans, 265
java.io, 263
java.lang, 262
Java native interface (JNI), 259–261, 283, 259–261, 283, 304–305
java.net, 263
java.util, 262–263
Java virtual machine architecture, 17, 241–261
binary classes, 256–259
data storage, 243–246
data types, 242–243
exceptions and errors, 254–256
instruction set, 246–254
serializability and reflection, 263–265
Jikes Research Virtual Machine (RVM), 306, 320–327
Join points, 187, 188, 190, 200
Jumps
See also Indirect jumps
instructions, 566
profiling, 165
translation lookaside buffers (JTLB), 341–342
Just-in-time (JIT) compilation, 241, 305–306

Lazy evaluation, 71–73
Least recently used (LRU) algorithm, 135–136
Lie detection, 509–510
Linux, 104, 107, 113, 129, 130, 133, 141
Livewire system, 508–510
LISP, 29
Little-endian byte order, 76
Loader, 85
Load Processor Status Word (LPSW), 384, 417–418
Locality, improving memory, 167–171, 558
Load real address (LRA), 385
Local storage, 243
Logging, activity, 505–506, 511–515
Logical partitioning, 458–475
dynamic, 471–473
hardware support, 465–469
hypervisors, 464, 469–471, 474–475
IBM 390 logical partitioning (LPAR), 460–464, 473–474
major features, 458–460
system virtual machines compared to, 464–465
lookupswitch instruction, 250–251
Loop, hoisting invariant expression out of a, 203–205
Loop peeling, 319–320
Machine, use of term, 2, 9
Magic number, 257–258
Managed code, 230
Map table, 55–56
Mark-and-sweep garbage collectors, 296–297
Memory
address formation, 277
address resolution, 75
address space mapping, 96, 97–102
architecture, 563–564
borrowing, 482–483
cache, 558–559
Cellular Disco System, 478–480, 482–483
coherence, 488–490, 595–596
concealed, 21–22, 334–337
consistency models, 490–496, 596–600
conventional, 276–277

data alignment, 75–76
hypervisor, 466
IBM AS/400 system, 359–361
interpreter, 29–30
Java global, 244, 282
Java hierarchy, 245–246
leak, 240
load and store instructions, 565
locality, 167–171
main, 557–558
management, 581–582
mapping, 478–480
method, 257
Pascal P-code, 225–227
physical versus real, 397
system calls, 584–585
update instructions, 187, 190
virtual, 396–404, 435
volatile, 187–188
Memory architecture emulation access privilege types, 102
address space, 102
protection/allocation granularity, 102, 103–114
Memory protection, 103
fine-grain write-protection, 110–111
host-supported, 104–105
integrity checks, 511
page size issues, 105–106
protecting runtime, 112–114
self-referencing and -modifying code, 107–114, 337–339
Memory state
binary translation, 127–128
codesigned virtual machines and, 333–337
compatibility, 183, 184
Message Passing Interface (MPI), 449
Metadata, 275
Method inlining. *See* Procedure inlining
Method memory, 257
Methods count and information, Java, 259
Microsoft
See also Windows

common language infrastructure (CLI), 17, 232, 267–275
intermediate language (MSIL), 232, 271–274
shared-source CLR, 232
Middleware, 546
Migration of computing environments, 521–532
See also Mobility
MIPS ISA, 70
mmap(), 104–105
Mobility
Internet Suspend/Resume (ISR), 524–526
reducing memory before, 527–528
reducing start-up time, 530
reducing transmission time and bandwidth, 530–531
reducing transmitted packet size, 528–530
role of virtual machines, 520–535
Stanford Collective project, 526–532
VMotion, 532–535
Mojo same-ISA optimizer, 119, 138, 209
Most-frequently-used method, 175, 176, 177
Most-recently-used method, 175, 177–178
MOVE, 430
mprotect(), 104, 107, 113
Multiprocessor architecture, 588–600
Multiprocessor systems partitioning, 445
Cellular Disco System, 475–485
different host and guest, 485–496
logical, 458–475
mechanisms to support, 452
partitioning techniques, 453–454
physical, 455–457
reasons for, 446–452
Multiprogramming, 13, 588–600
Multiversioning, 312–313
Namespace, 285
Native interface, Java, 259–261, 283, 304–305
Native virtual machines, 11, 379–381
Network-based intrusion-detection systems (NIDS), 502–505
Networking, 231, 240–241
Network virtualization, 410–412
Node profile, 160

Null reference checking, 318–320
Object-oriented high-level language virtual machines, 228–231
networking, 231, 240–241
performance, 231, 241
robustness, 230–231, 237–240
security and protection, 230, 232–237
Objects, Java, 243
Omniware VM, 112
On-stack replacement (OSR), 313–316
Open Grid Services Architecture (OGSA), 540
Operand storage, 243
Operating system
call emulation, 86, 128–129
dependencies, 279
interface, 2, 13, 583–586
managed state mapping, 89
organization, 580–583
Operating system emulation different operating systems, 132–133
runtime-implemented, 129–130
same, 128–132
Operation-level interface, 408–409
Optimization, 12
See also Binary optimization, dynamic; Emulation
binary, 15
control transfer, 64–68
Organization-level virtual machines. *See* Grids
Overhead, profiling, 166–167
Package, 284
Page fault
active detection, 348–349
compatibility, 347–348
lazy detection, 349–350
nonpaged mode, 420
pseudo, 420–421
Page tables, 399–402, 572–573
architected, virtualization 399–402
translation lookaside buffer and, 575–576
Paravirtualization, 422
Partial procedure inlining, 170
Partitioning. *See* Multiprocessor systems

partitioning
Partition memory base register (PMBR), 466, 467
Partition memory limit register (PMLR), 466
Pascal, 16
Pascal P-code
instruction set, 227
mark pointer, 226
mark stack, 226
memory architecture, 225–227
summary, 227–228
Patching, 210, 211–213, 391, 392–393
Patch table, 211
Path profile, 161
Performance
code cache management, 139–140
high-level language virtual machines and, 306–320
instruction emulation and, 115
object-oriented high-level language virtual machines and, 231, 241
system virtual machines and, 415–426
Perl, 29
Physical partitioning, 455–457
Platform(s)
Java, 232, 261–262
use of term, 2
Policy engine, 509
Polling modules, 511
Polymorphic inline caching (PIC), 312
Pop stack into flags register (POPF), 385–386, 391, 430
Porting, 221–222
Power consumption. *See* Grids
PowerPC ISA, 35–36
arithmetic, 74
condition codes, 70
description of, 600–606
Precise exceptions, 277
Precise state reconstruction, 196–198
Precise traps, 62, 344–351
Predecoding threaded interpretation, 34–37
incremental, 55–62
Preemptive flush, 137–138

Preferred-machine assist, 423
Primordial class loader, 285
Privileged instructions, 384–385
decoding, 418
Privilege levels, 566–568
Probes, profiling, 157
Procedure inlining, 169–170, 308–310
Processors
management, 580–581
microarchitectures, 555–556
system calls, 584
virtualization, 382–396, 429–431
Process virtual machines, 10–11
code cache management, 133–140
compatibility, 87–95
components of, 84
emulators and dynamic binary translators, 13–15
exception emulation, 119–128
FX!32 example, 142–145
high-level language, 15–17
implementation, 85–87
implementation leaks, 93–95
instruction emulation, 114–119
memory architecture emulation, 102–114
multiprogramming, 13
operating system emulation, 128–133
operations and control transfers, 89–92
role of, 83
runtime software, 84
same-ISA binary optimizers, 15
state mapping, 89, 95–102
system environment, 140–142
Profile database, 85–86, 116
Profiling, 14–15, 67
basic block/node, 160
binary optimization and, 148–150, 156–167
collecting, 161–163
counter decaying, 165
edge, 161
interpretation and, 163–165

jump instructions, 165
logs, 157
overhead, 166–167
path, 161
role of, 157–159
translated code, 165–166
types of, 159–161
Program shepherding, 63, 515
Prolog side table, 206
Protection system references, 430
Pseudo page-fault handling, 420–421
Pseudo self-modifying code, 107–110
rdtsc() (read time-stamp counter), 440–442
Real address space, 557
Real map table, 397
Recursive virtualization, 390–391
Reference type, Java, 242
Reflection, 262, 263–265
Register(s)
architecture, 69–70, 562–563, 568–569
assign, 195–196
map (RMAP), 194
replicated, 465–466
update instructions, 187, 188–189
Register state
binary translation, 126–127
compatibility, 183–184
hypervisor, 465
Register state mapping, 50, 51–52, 69–70, 96
codesigned virtual machines and, 333–337
consistent, 127, 184–186
Remote Spooling Communications System (RSCS), 421
Repair code, 197
Replacement algorithm, 559
Resource control, system virtual machines and, 377–379, 387
Resource management, ISA, 566–580
Return address stack (RAS), dual-address, 342–344
Return instructions, 251
Reverse translation side table, 124–125, 138
ReVirt, 512–515

RIO system, 515–519
See also DynamoRIO system
RISC ISAs, 38, 125, 186, 206–207, 329
Robustness, 230–231, 237–240
Root set, garbage collection, 302–303
Runtime, 11
-implemented operating system functions, 129–130
memory, protecting, 112–114
in process virtual machines, 84
Runtime monitor, protecting, 519
Same-ISA dynamic binary optimization, 15, 29, 208–218
Same-ISA emulation, 63–64
Sampling-based profiling, 162–163
Sandbox. *See* Security and protection
Scalar replacement, 317
Scheduling algorithm, 192–199
Security, 501
binary rewriting, 515–519
intrusion-detection systems, 502–505, 508–511
logging, 505–506, 511–515
monitoring and recovering from attacks, 505–506
role of virtual machines, 506–520
Security and protection, 230, 232–237, 286–294
basic, 286–287
enforcement, 290–291
enhanced model, 291–294
intraprocess, 287–290
Segment sharing, 423
Self-modifying code, 62, 107–114, 278–279, 337–339
Self-referencing code, 62, 107–114, 278–279, 337–339
Sensitive instructions, 385–386, 430
Serializability and reflection, 263–265
SETI@home project, 545
setrlimit(), 133
Shade System simulation, 63, 77–79, 137, 153
Shadow page tables, 399
bypass assist, 423
Shadow stack, 68
Shared-memory processing (SMP), 446, 447, 590, 594–600
Side tables, 87

epilog, 205–206
prolog, 206
reverse translation, 124–125, 138
Signals, 585–586
Signature detection, 510–511
Signing, 292
SIGSEGV, 104, 105
Simulation, Shade System, 63, 77–79, 137, 153
Slots, Java stack, 243
Software indirect jump prediction, 66–67
Software pipelining, 47
Software translation tables, 97–100
Source program counter/code 37, 54, 56
locating, 124–126
tracking, 59–60
SPARC ISA, 70, 74
Spatial locality, 167
Specialization, 312–313
Spool files, 421
Stack(s)
architected, 314
implementation, 314
inspection, 293–294
replacement, 313–316
structure, Java, 243–244, 282
tracking, Java, 251–254
Staged emulation, 116–119, 151–153
Stanford Collective, 526–532
Start Interpretive Execution (SIE), 424–426
Start point, trace, 172
Start threshold value, 175
Start-up time, 115
State management, 375–377
State mapping
codesigned virtual machines and register, 333–337
compatibility issues, 101–102
direct, 200–101
memory address, 96, 97–102
process virtual machine, 89, 95–102
register, 50, 51–52, 69–70, 96, 333–337

Static basic block, 56–57
Static predecoding, 52–53
Static translation, 52–53
Steady state performance, 115
Stopping condition, superblock, 172
Storage, Java data, 243–246
Streams, MSIL, 271
Strength reduction, 201, 205
Sun Microsystems
HotSpot, 306
Java VM architecture, 17
physical partitioning, 455
Wabi system, 119, 133
Superblocks, 118, 150
continuation set, 175–176
dynamic formation, 173–178
inter-, optimizations, 205–206
most-frequently-used method, 175, 176, 177
most-recently-used method, 175, 177–178
starting points, 174–175
stopping points, 176–177
versus traces, 199–201
Super_Class, Java, 258
System call interface, 8, 408, 410
System clock register, 568–569
System virtual machines, 11–12, 17–22
applications, 370–373
codesigned, hardware optimization, 21–22
development of, 17–18, 369
IBM System/370, 381–382
implementation of, 19
input/output virtualization, 404–415, 431–435
Intel VT-x (Vanderpool), 436–442
logical partitioning and, 464–465
memory virtualization, 396–404, 435
native and hosted virtual machines, 379–381
performance enhancement, 415–426
processor virtualization, 382–396, 429–431
recursive virtualization, 390–391
resource control, 377–379

state management, 375–377
VMware Virtual Platform, 426–435
whole (emulation), 19–21
Tail duplication, 173
Target program counter 37, 54, 56
Technology-independent machine interface, 358
Temporal locality, 167
Terra system, 519
This_Class, Java, 258
Threaded interpretation, 29, 32–38 for CISC ISA, 44–46
direct, 37–38, 80, 81–82
indirect, 80, 81
predecoding, 34–37
Threads, Java, 265–267
Timer, virtual interval, 418–419
Time sharing, 369
Tokens, MSIL, 271–272
Traces, 171–173
superblocks versus, 199–201
Translated code, profiling, 165–166
Translation blocks, optimizing improving locality, 167–171
superblocks, 173–178
traces, 171–173
tree groups, 178–180
Translation lookaside buffer (TLB), 78–79, 338, 350–351, 399, 402–404
architected, virtualization 402–404
hit, 575–575
miss, 575
page tables and, 575–576
partitioning, 466–469
role of, 573–575
Translation tables, software, 97–100
direct, 100–101
pointers, 569
Transmeta Crusoe, 22, 24, 331, 338, 353, 354–357
Transparency, complete, 88
Traps
compatibility, 182–183
detecting, 120–121
handling, 577–580

precise, 62, 344–351
process virtual machine, 86
registers, 569
use of term, 119
Tree groups, 178–180
Trigger, 166
UNIX, 104–105 (*See also* Linux)
User ISA, 8, 561–566
User-managed state mapping, 89
VAX ISA, 39
Very long instruction word (VLIW) processors, 186, 331, 555
VirtualCenter, VMotion, 532–535
Virtual disks, 405
Virtual-equals-real virtual machine, 422–423
Virtual-ISA, 222
summary of features, 275–279
Virtualization, 3–4, 10
See also Multiprocessor systems partitioning
architected page table, 399–402
devices, 404–407
input/output, 404–415, 431–435
memory, 396–404, 435
network, 410–412
para-, 422
processor, 382–396, 429–431
recursive, 390–391
translation lookaside buffer (TLB), 402–404
with different guest and host, 485–496
Virtual Machine Communications Facility (VMCF), 421
Virtual machine monitor (VMM), 11, 331–332
See also System virtual machines
allocator, 386
assists, 418–419
components of, 386–387
dispatcher, 386
interpreter routines, 386–387
partitioning and, 452
properties of, 387
Virtual machines (VMs)
See also under type of

applications, 12–13
assists, 416–419
basics of, 9–13
process, 10–11, 13–17
role of, 4–6
system, 11–12, 17–22
taxonomy, 22–23
versatility of, 23–24
VMotion, 532–535
VMware Virtual Platform, 426–435
Volatile memory location, 187–188
VT-x (Vanderpool), 436–442
Wiggins/Redstone system, 209
Windows operating system, 130–132
Win32, 133, 141
World Wide Grid Forum, 540
Wrappers, 9
Write barriers, 302
Write-protect table, 338
Xen system, 422

Previous chapter
References
Next chapter
Instructions for online access

Instructions for online access

Thank you for your purchase. Please note that your purchase of this Elsevier eBook also includes access to an online version. Please **click here** (or go to **http://ebooks.elsevier.com**) to request an activation code and registration instructions in order to gain access to the web version.

APPENDIX SMITH02

```
leader  02246cam a22004094a 4500
001     9945403453406236
005     20200824152434.0
008     050509s2005    ne a     b    001 0 eng
010     ##$a  2005013376
015     ##$aGBA526375 $2bnb
016     7#$a013148297 $2Uk
020     ##$a1558609105
035     ##$a(OCoLC)ocm58830355
035     ##$a4540345
035     ##$a(ppiu)4540345-pittdb
035     ##$a(pittdb)4540345-pittdb
040     ##$aDLC $cDLC $dUKM $dC#P $dC@R $dOrLoB-B $dUtOrBLW
042     ##$apcc
049     ##$aPITT
050     00$aQA76.9.V5 $bS54 2005
082     00$a005.4/3 $222
100     1#$aSmith, James E. $q(James Edward), $d1950-
245     10$aVirtual machines : $bversatile platforms for systems and processes / $cJames E. Smith, Ravi Nair.
260     ##$aAmsterdam ; $aBoston : $bMorgan Kaufmann Publishers, $cc2005.
300     ##$axxii, 638 p. : $bill. ; $c25 cm.
504     ##$aIncludes bibliographical references (p. 613-628) and index.
505     00$gCh. 1. $tIntroduction to virtual machines --  $gCh. 2. $tEmulation : interpretation and binary translation --  $gCh. 3. $tProcess virtual machines -
520     1#$a"Virtual Machines carefully examines core topics of VM technology including instruction set emulation, dynamic program translation and optimization,
650     #0$aVirtual computer systems.
650     #0$aComputer architecture.
650     #0$aSystem design.
700     1#$aNair, Ravi.
906     ##$aULS
945     ##$aPromptCat $bApproval $cPITT
946     ##$ahill $bBookA $c31735051828923
947     ##$a54.56 $b10296137
994     ##$a92 $bPIT
```



BOOK

Virtual machines : versatile platforms for systems and processes

Smith, James E. (James Edward), 1950- / James E. Smith, Ravi Nair.
c2005
xxii, 638 p. : ill. ; 25 cm.

TOP
SEND TO
VIEW ONLINE
GET IT
DETAILS

Available at Science and Engineering Library (Benedum Hall) General Collection 2 (QA76.9.V5 S54 2005) >

Available Online >

Report a Problem

Chapters of this book (12) >

Send to

CITATION EMAIL ENDNOTE/ZOTERO PRINT PERMALINK

View Online

Full text availability

EBSCOhost Academic eBook Collection (North America)
Access Model: Unlimited Users

O'Reilly Online Learning: Academic/Public Library Edition
Access: Please note that the platform supports unlimited ac...

Get It

Please sign in to check if there are any request options. Sign in

BACK TO LOCATIONS

LOCATION ITEMS

Science and Engineering Library (Benedum Hall)
Available , General Collection 2 ; QA76.9.V5 S54 2005
(1 copy, 1 available, 0 requests)

Item in place

Details

Title	Virtual machines : versatile platforms for systems and processes
Creator	Smith, James E. (James Edward), 1950- >
Contributor	Nair, Ravi. >
Language	English

Related reading

Recommended items that are related to the record



Computer architecture : ...
Blaauw, Gerrit A.
ISBN 9780201105575
suggested by



Computation structures
Ward, Stephen A
ISBN 9780585393469
suggested by



Structure and interpretation of ...
Abelson, Harold.
ISBN 9780262011532
suggested by



Programming Perl
Christiansen, Tom.
ISBN 9781449321475
suggested by



Computer architecture : a ...
Hennessy, John L.
ISBN 9780128119051
suggested by

More

Publisher	Amsterdam ; Boston : Morgan Kaufmann Publishers
Date	c2005
Type	Book
Description	xxii, 638 p. : ill. ; 25 cm.
Notes	Includes bibliographical references (p. 613-628) and index.
Summary	"Virtual Machines carefully examines core topics of VM technology including instruction set emulation, dynamic program translation and optimization, high level language virtual machines and system virtual machines for both single-user systems and servers. It offers graduate students, researchers, and practicing professionals a unique resource that gathers and synthesizes the work from a number of fields into a single authoritative resource, defining a new and unified discipline."--BOOK JACKET.
Contents	Ch. 1. Introduction to virtual machines -- Ch. 2. Emulation : interpretation and binary translation -- Ch. 3. Process virtual machines -- Ch. 4. Dynamic binary optimization -- Ch. 5. High-level language virtual machine architecture -- Ch. 6. High-level language virtual machine implementation -- Ch. 7. Codesigned virtual machines -- Ch. 8. System virtual machines -- Ch. 9. Multiprocessor virtualization -- Ch. 10. Emerging applications -- App. A. Real machines.
Subject	Virtual computer systems › Computer architecture › System design ›
Related titles	Available in other form: Virtual machines : versatile platforms for systems and processes ›
Identifier	LC : 2005013376 ISBN : 1558609105 OCLC : (OCoLC)ocm58830355
Dedup Member Ids	9998552188206236
Source	Library Catalog

Virtual Browse



Mastering windows server hyper v ...
2017



Virtualization security : protecting virtualized environment ...
2013



Virtual machines : versatile platforms for systems and ...
c2005



Oracle VM Manager 2.1.2 : manage a flexible and ...
c2009

Links

Find this Item in Google Books ›
Display Source Record ›

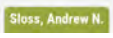
Syndetics Unbound

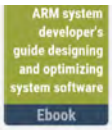
Summary
Also Available As
Browse Shelf
Series
Tags
Reader Reviews
Book Profile

SUMMARY

Virtual Machine technology applies the concept of virtualization to an entire machine, circumventing real machine compatibility constraints and hardware resource constraints to enable a higher degree of software portability and flexibility. Virtual machines are rapidly becoming an essential element in computer system design. They provide system security, flexibility, cross-platform compatibility, (read more)

SERIES



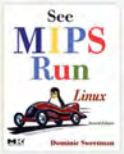












SEE FULL SERIES

READER REVIEWS

BE THE FIRST TO WRITE A REVIEW

ALSO AVAILABLE AS



Ebook

TAGS

technique | technology | electrical engineering | informatica | network | computers | computing | computer systems | **computer science** | **computer architecture** | programming

BROWSE TAGS

BOOK PROFILE

Person
Moreno, Jaime; Calder, Alexander (1898-1976)

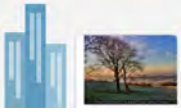
Topics
Operating systems; Software; Java; Product development

Location
New York; San Francisco California; United States--US; Florida; Massachusetts; Minnesota; Palo Alto California; Pittsburgh Pennsylvania; United Kingdom--UK

Organization
Microsoft Corp; IBM Research; Microsoft Research; University of Illinois; Harvard University; Philips Research; Princeton University; University of Wisconsin-Madison; Kronstadt; MIT Press

BROWSE PROFILE

BROWSE SHELF








QA76.9.V5 S53 2013 QA76.9.V5 S74 2015

SEE FULL SHELF

Catalog enrichment powered by Syndetics Unbound

APPENDIX SMITH03

```
leader  03687cam a22007334a 4500
001     991000139540704436
005     20240613123217.0
008     050509s2005    nw a     b    001 0 eng
010     ##$a  2005013376
015     ##$aGB45126275 $2bnb
016     7##$a011148297 $2Uk
019     ##$a71762682$
020     ##$a1558609105
020     ##$a9781558609105
024     ##$a9781558609105
035     ##$a(OCoLC)ocm58836055 $z(OCoLC)71762682$
040     ##$aDLC $beng $cDLC $dUKM $dCMP $dCQM $dBAKER $dIAA $dVRC $dBTCTA $dLVE $dYDXCP $dOKU $dIDB $dBERIS $dOCLCQ $dCHVHO $dILU $dBDX $dOCLCF $dOCLCO $dOCLCQ $dXRBSZ $dUKMGB $dILAOH $dOCLCO $dMMC $dOCLCO $dOCLCL $dOCLCQ
042     ##$apcc
050     ##$aQA76.9.V5 $b554 2005
072     #7$ass1$x $2bnco
082     ##$a005.4/3 $222
100     1#$aSmith, James E. $q(James Edward), $d1950- $1https://id.oclc.org/worldcat/entity/E39PB3KXxRRcBwPTtyC8YXGCP
245     10$aVirtual machines : $bversatile platforms for systems and processes / $cJames E. Smith, Ravi Nair.
260     ##$aAmsterdam ; $aBoston : $bMorgan Kaufmann Publishers, $c2005.
300     ##$axxii, 638 pages : $billustrations ; $c24 cm
336     ##$atext $btxt $2rdacontent
337     ##$aunmediated $bn $2rdamedia
338     ##$avolume $bnc $2rdacarrier
504     ##$aIncludes bibliographical references and index.
505     0#$aIntroduction to virtual machines -- Emulation : interpretation and binary translation -- Process virtual machines -- Dynamic binary optimization -- High-level language virtual machine architecture -- High-level language virtual machine implementation -- Codesigned virtual machines -- System virtual machines -- Multiprocessor virtualization -- Emerging applications.
520     1#$a"Virtual Machines carefully examines core topics of VM technology including instruction set emulation, dynamic program translation and optimization, high level language virtual machines and system virtual machines for both single-user systems and servers. It offers graduate students, researchers, and practicing professionals a unique resource that gathers and synthesizes
650     #0$aVirtual computer systems.
650     #0$aComputer architecture.
650     #0$aSystem design.
650     #6$aSystèmes virtuels [Informatique]
650     #6$aOrdinateurs $xArchitecture.
650     #6$aConception de systèmes.
650     #7$aComputer architecture $2fast
650     #7$aSystem design $2fast
650     #7$aVirtual computer systems $2fast
650     #7$aVirtuelle Maschine $2gnd
650     #7$aVirtual computer systems. $2nli
650     #7$aComputer architecture. $2nli
650     #7$aSystem design. $2nli
650     #7$aVirtuelle Maschine. $2swd
700     1#$aNair, Ravi.
776     08$iOnline version: $aSmith, James E. (James Edward), 1950- $tVirtual machines. $dAmsterdam ; Boston : Morgan Kaufmann Publishers, 2005 $w(OCoLC)607654898
856     41$3Table of contents $uhttp://catdir.loc.gov/catdir/toc/ecip0512/2005013576.html
856     42$3Publisher description $uhttp://catdir.loc.gov/catdir/enhancements/fy0622/2005013576-d.html
901     ##$aocm58836055
916     ##$a20060113
917     ##$a20060113
918     ##$aCATALOGER
919     ##$a203AM03
920     ##$aQATARSTAFF
994     ##$a92 $bPMC
999     ##$aQA76.9 .V5 S54 2005 $wLC $c1 $i13848201496137 7 $d12/6/2017 $e12/6/2017 $kCHECKEDOUT $lSTACKS $mEWGRASCT $n13 $p$51.95 $qI.$rM $sY $tBOOK $uAR/3/2005
999     ##$aQA76.9 .V5S54 2005 $wLC $c1 $i13848201425708 3 $d7/6/2011 $e7/6/2011 $lSTACKS $mQATAR $n1 $rY $sY $tBOOK $uI/13/2006
999     ##$aQA76.9 .V5S54 2005 $wLC $c2 $i13848201543636 9 $lSTACKS $mQATAR $rY $sY $tBOOK $u12/7/2011
999     ##$aQA76.9 .V5S54 2005 $wLC $c3 $i13848201695256 2 $d11/28/2012 $e11/28/2012 $kCHECKEDOUT $lSTACKS $mQATAR $n1 $rM $sY $tBOOK $u11/27/2012 FORM=MARC
```



BOOK

Virtual machines : versatile platforms for systems and processes

Smith, James E. (James Edward), 1950-; Nair, Ravi.

2005

Available at Engineering & Science Library STACKS (QA76.9 .V5 S54 2005) and other locations >

Available Online >

TOP
VIEW ONLINE
FIND IN LIBRARY
LINKS
DETAILS
SEND TO
VIRTUAL BROWSE

Find in Library

Please sign in to check if there are any request options. Sign in

LOCATIONS

Engineering & Science Library
Available , STACKS ; QA76.9 .V5 S54 2005

Qatar Library
Out of library , Q-STACKS (Qatar stacks area) ; QA76.9 .V5S54 2005

View Online

Full text availability

EBSCOhost Academic eBook Collection (North America)

O'Reilly Online Learning: Academic/Public Library Edition

Links

Table of contents >
Publisher description >
Report a Problem >
Signed in and still can't find what you're looking for? Let us know
Display source record >

Details

Title	Virtual machines : versatile platforms for systems and processes
Creator	Smith, James E. (James Edward), 1950- > Nair, Ravi. >
Subject	Computer architecture > Conception de systèmes > Ordinateurs -- Architecture > System design > Systèmes virtuels (Informatique) > Virtual computer systems > Virtuelle Maschine >
Description	"Virtual Machines carefully examines core topics of VM technology including instruction set emulation, dynamic program translation and optimization, high level language virtual machines and system virtual machines for both single-user systems and servers. It offers graduate students, researchers, and practicing professionals a unique resource that gathers and synthesizes the work from a number of fields into a single authoritative resource, defining a new and unified discipline."--Jacket.
Contents	Introduction to virtual machines -- Emulation : interpretation and binary translation -- Process virtual machines -- Dynamic binary optimization -- High-level language virtual machine architecture -- High-level language virtual machine implementation -- Codesigned virtual machines -- System virtual machines -- Multiprocessor virtualization -- Emerging applications.
Related titles	Available in other form: Online version: Smith, James E. (James Edward), 1950- Virtual machines. Amsterdam ; Boston : Morgan Kaufmann Publishers, 2005 > Available in other form: Virtual machines : versatile platforms for systems and processes >
Publisher	Amsterdam ; Boston : Morgan Kaufmann Publishers
Creation Date	2005
Format	xxii, 638 pages : illustrations ; 24 cm

Bibliography Note	Includes bibliographical references and index.
Source	Library Catalog

Send to

EXPORT TO EXCEL | PRINT | EMAIL | PERMALINK | QR | CITATION | EXPORT RIS | EXPORT BIBTEX | EASYBIB

Virtual Browse



are VI and iere SDK : aging the are structure ...

0



Virtualization : a manager's guide ...

c2011



Running Xen : a hands-on guide to the art of virtualization ...

2008



Virtual machines : versatile platforms for systems and ...

2005



Virtual machines : versatile platforms for systems and ...

2005



(In)visible : learning to act in the metaverse ...

2008

APPENDIX SMITH04

Planned maintenance: Realtime availability, I Want It, and My Account will be unavailable between 5:00 and 8:00 PM, Tuesday January 7, 2025.

PennState University Libraries | Catalog

Bookmarks 0 Course Reserves My Account

Keyword Search... Search Advanced search Start Over

Share Bookmark Report an Issue

MARC View

LEADER 06086cam a2200745 a 4500			
001 43324546			
003 SIRSI			
005 20240305141628.0			
006 m o d			
007 cr cn|||||||||			
008 070802s2005 ne a ob 001 0 eng d			
019			a| 233572835 a| 607654990 a| 648349469 a| 1035703609 a| 1103276534 a| 1153017613 a| 1159064878 a| 1160030149 a| 1160033124 a| 1162395413 a| 1179509662 a| 1181926131 a| 1189808409 a| 1192335357 a| 1240533920
019			a| MARS
020			a| 9781558609105
020			a| 1558609105
020			a| 0080525407 q| (electronic bk.)
020			a| 9780080525402 q| (electronic bk.)
020			a| 1281227714
020			a| 9781281227713
020			a| 9786611227715
020			a| 6611227717
035			a| (OCoLC)162129897
037			a| 97278:97278 b| Elsevier Science & Technology n| http://www.sciencedirect.com
040			a| OPELS b| eng e| pn c| OPELS d| OKU d| OCLCQ d| OSU d| YDXCP d| IDEBK d| NRU d| E7B d| N$T d| EBLCP d| OCLCQ d| OCLCF d| OCLCE d| OCLCO d| OCLCQ d| DEBSZ d| OCLCQ d| COO d| MERUC d| STF d| D6H d| VTS d| OCLCQ d| G3B d| VT2 d| K6U d| OCLCQ d| LDP d| AGLDB d| VLY d| BRF d| OCLCQ d| OCLCO d| AUW d| OCLCQ d| OCLCO d| OCLCL d| UtOrBLW
042			a| dlr
049			a| UPMM
050		4	a| QA76.9.V5 b| S54 2005eb
072		7	a| COM x| 046090 2| bisacsh
072		7	a| COM x| 088000 2| bisacsh
072		7	a| COM x| 056000 2| bisacsh
082	0	4	a| 005.4/3 2| 22
100	1		a| Smith, James E. q| (James Edward), d| 1950- 0| http://id.loc.gov/authorities/names/n94062575 1| https://id.oclc.org/worldcat/entity/E39PBJkXwXrBcBxPYtyC8YkCcP
245	1	0	a| Virtual machines : b| versatile platforms for systems and processes / c| James E. Smith, Ravi Nair.
264		1	a| Amsterdam ; a| Boston : b| Morgan Kaufmann Publishers, c| 2005.
300			a| 1 online resource (xxii, 638 pages) : b| illustrations.
336			a| text b| txt 2| rdacontent
337			a| computer b| c 2| rdamedia
338			a| online resource b| cr 2| rdacarrier
490	1		a| The Morgan Kaufmann Series in Computer Architecture and Design
504			a| Includes bibliographical references and index.
505	0		a| Introduction to virtual machines -- Emulation : interpretation and binary translation -- Process virtual machines -- Dynamic binary optimization -- High-level language virtual machine architecture -- High-level language virtual machine implementation -- Codesigned virtual machines -- System virtual machines -- Multiprocessor virtualization -- Emerging applications.
520			a| Virtual Machine technology applies the concept of virtualization to an entire machine, circumventing real machine compatibility constraints and hardware resource constraints to enable a higher degree of software portability and flexibility. Virtual machines are rapidly becoming an essential element in computer system design. They provide system security, flexibility, cross-platform compatibility, reliability, and resource efficiency. Designed to solve problems in combining and using major computer system components, virtual machine technologies play a key role in many disciplines, including operating systems, programming languages, and computer architecture. For example, at the process level, virtualizing technologies support dynamic program translation and platform-independent network computing. At the system level, they support multiple operating system environments on the same hardware platform and in servers. Historically, individual virtual machine techniques have been developed within the specific disciplines that employ them (in some cases they arent even referred to as virtual machines), making it difficult to see their common underlying relationships in a cohesive way. In this text, Smith and Nair take a new approach by examining virtual machines as a

ASK A LIBRARIAN

			unified discipline. Pulling together cross-cutting technologies allows virtual machine implementations to be studied and engineered in a well-structured manner. Topics include instruction set emulation, dynamic program translation and optimization, high level virtual machines (including Java and CLI), and system virtual machines for both single-user systems and servers. *Examines virtual machine technologies across the disciplines that use themoperating systems, programming languages and computer architecturedefining a new and unified discipline. *Reviewed by principle researchers at Microsoft, HP, and by other industry research groups. *Written by two authors who combine several decades of expertise in computer system research and development, both in academia and industry.
546			a| English.
588	0		a| Print version record.
650		0	a| Virtual computer systems. 0| http://id.loc.gov/authorities/subjects/sh85143804
650		0	a| Computer architecture. 0| http://id.loc.gov/authorities/subjects/sh85029479
650		0	a| System design. 0| http://id.loc.gov/authorities/subjects/sh85131736
650		6	a| Systèmes virtuels (Informatique)
650		6	a| Ordinateurs x| Architecture.
650		6	a| Conception de systèmes.
650		7	a| COMPUTERS x| Operating Systems x| Virtualization. 2| bisacsh
650		7	a| COMPUTERS x| System Administration x| General. 2| bisacsh
650		7	a| COMPUTERS x| Utilities. 2| bisacsh
650		7	a| Computer architecture. 2| fast
650		7	a| System design. 2| fast
650		7	a| Virtual computer systems. 2| fast
700	1		a| Nair, Ravi. 0| http://id.loc.gov/authorities/names/n2005034082
776	0	8	i| Print version: a| Smith, James E. (James Edward), 1950- t| Virtual machines. d| Amsterdam ; Boston : Morgan Kaufmann Publishers, 2005 z| 1558609105 z| 9781558609105 w| (DLC) 2005013376 w| (OCoLC)58830355
830		0	a| Morgan Kaufmann series in computer architecture and design.
856	4	0	u| https://ezaccess.libraries.psu.edu/login?url=https://search.ebscohost.com/login.aspx?direct=true&scope=site&db=nlebk&AN=130249
938			a| EBL - Ebook Library b| EBLB n| EBL334029
938			a| ebrary b| EBRY n| ebr10216756
938			a| EBSCOhost b| EBSC n| 130249
938			a| YBP Library Services b| YANK n| 2796340
949			a| Electronic resource w| ASIS c| 1 i| 43324546-1001 l| ONLINE m| ONLINE r| Y s| Y t| ONLINE u| 2/8/2024 o| .STAFF. PALCI/Ebsco ebooks - Academic collection 50000 records 20240208

PennState

Copyright @2019 The Pennsylvania State University. All rights reserved. Except where otherwise noted, this work is subject to a Creative Commons Attribution 4.0 license. Details and exceptions.

Legal Statements | PSU Hotlines

PENN STATE UNIVERSITY LIBRARIES

Libraries Home (814) 865-6368
Libraries Intranet (Staff Only)
Accessibility Help
Website Feedback
Policies and Guidelines
Staff Directory

CONNECT WITH PENN STATE UNIVERSITY LIBRARIES

Facebook
Twitter
Instagram

Planned maintenance: Realtime availability, I Want It, and My Account will be unavailable between 5:00 and 8:00 PM, Tuesday January 7, 2025.

PennState University Libraries | Catalog

Bookmarks 0 Course Reserves My Account

Keyword Search... Search Advanced search Start Over

Share Bookmark Report an Issue



Virtual machines : versatile platforms for systems and processes / James E. Smith, Ravi Nair

Author	Smith, James E. (James Edward), 1950-
Published	Amsterdam ; Boston : Morgan Kaufmann Publishers, 2005.
Physical Description	1 online resource (xxii, 638 pages) : illustrations
Additional Creators	Nair, Ravi

ASK A LIBRARIAN

Access Online

ezaccess.libraries.psu.edu

Search inside at Google Books

Series

Morgan Kaufmann series in computer architecture and design

Language Note

English.

Contents

Introduction to virtual machines -- Emulation : interpretation and binary translation -- Process virtual machines -- Dynamic binary optimization -- High-level language virtual machine architecture -- High-level language virtual machine implementation -- Codesigned virtual machines -- System virtual machines -- Multiprocessor virtualization -- Emerging applications.

Summary

Virtual Machine technology applies the concept of virtualization to an entire machine, circumventing real machine compatibility constraints and hardware resource constraints to enable a higher degree of software portability and flexibility. Virtual machines are rapidly becoming an essential element in computer system design. They provide system security, flexibility, cross-platform compatibility, reliability, and resource efficiency. Designed to solve problems in combining and using major computer system components, virtual machine technologies play a key role in many disciplines, including operating systems, programming languages, and computer architecture. For example, at the process level, virtualizing technologies support dynamic program translation and platform-independent network computing. At the system level, they support multiple operating system environments on the same hardware platform and in servers. Historically, individual virtual machine techniques have been developed within the specific disciplines that employ them (in some cases they arent even referred to as virtual machines), making it difficult to see their common underlying relationships in a cohesive way. In this text, Smith and Nair take a new approach by examining virtual machines as a unified discipline. Pulling together cross-cutting technologies allows virtual machine implementations to be studied and engineered in a well-structured manner. Topics include instruction set emulation, dynamic program translation and optimization, high level virtual machines (including Java and CLI), and system virtual machines for both single-user systems and servers. *Examines virtual machine technologies across the disciplines that use themoperating systems, programming languages and computer architecturedefining a new and unified discipline. *Reviewed by principle researchers at Microsoft, HP, and by other industry research groups. *Written by two authors who combine several decades of expertise in computer system research and development, both in academia and industry.

Subject(s)

Virtual computer systems
Computer architecture
System design
Systèmes virtuels (Informatique)
Ordinateurs—Architecture
Conception de systèmes
COMPUTERS—Operating Systems—Virtualization
COMPUTERS—System Administration—General
COMPUTERS—Utilities

ISBN

9781558609105
1558609105
0080525407 (electronic bk.)
9780080525402 (electronic bk.)
1281227714
9781281227713
9786611227715
6611227717

Bibliography Note

Includes bibliographical references and index.

View MARC record | catkey: 43324546

PennState

Copyright @2019 The Pennsylvania State University. All rights reserved. Except where otherwise noted, this work is subject to a Creative Commons Attribution 4.0 license. Details and exceptions.

Legal Statements | PSU Hotlines

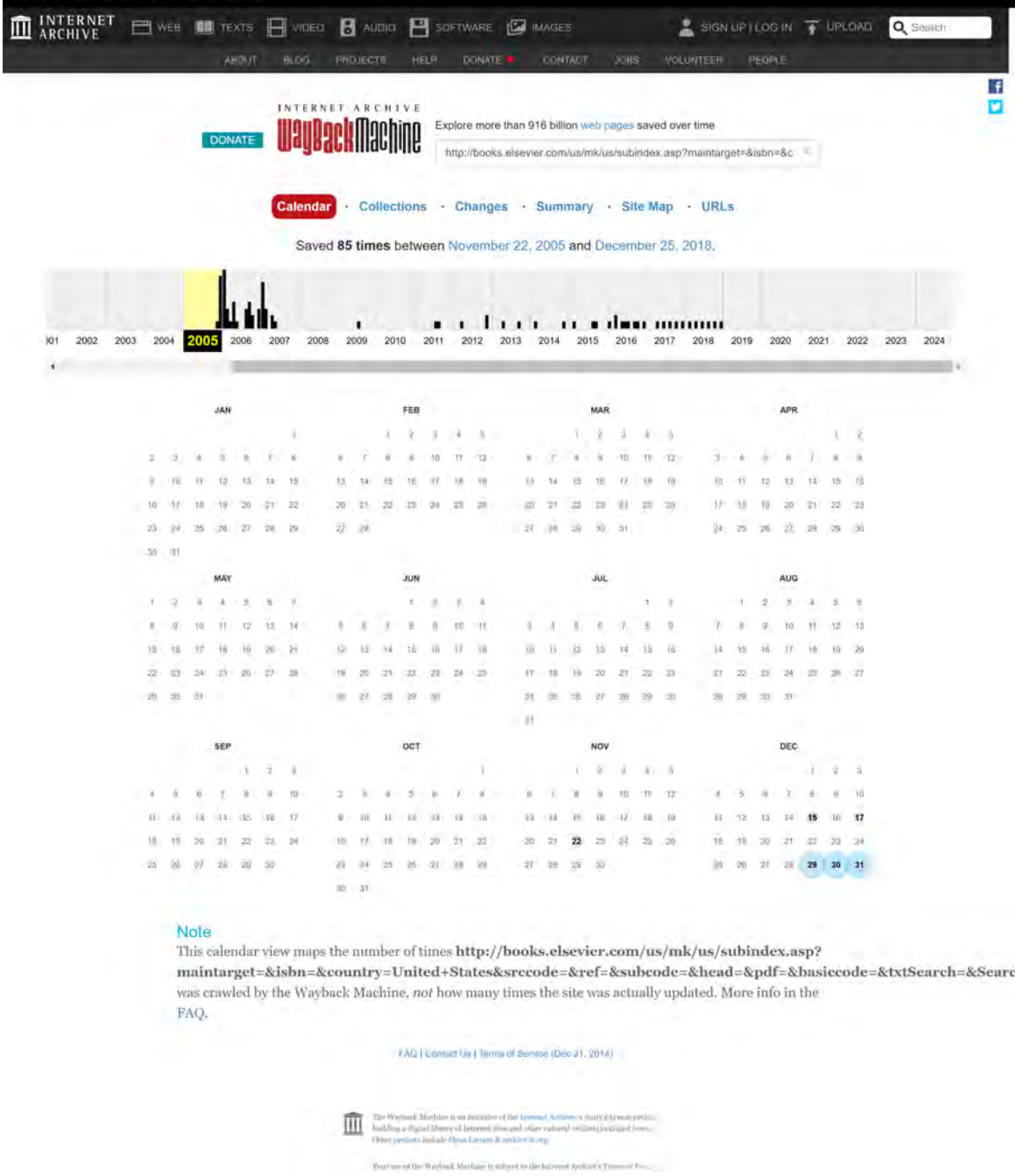
PENN STATE UNIVERSITY LIBRARIES

Libraries Home (814) 865-6368
Libraries Intranet (Staff Only)
Accessibility Help
Website Feedback
Policies and Guidelines
Staff Directory

CONNECT WITH PENN STATE UNIVERSITY LIBRARIES

Facebook
Twitter
Instagram

APPENDIX SMITH05


INTERNET ARCHIVE
WEB TEXTS VIDEO AUDIO SOFTWARE IMAGES
SIGN UP | LOG IN UPLOAD
Search
ABOUT BLOG PROJECTS HELP DONATE CONTACT JOBS VOLUNTEER PEOPLE
DONATE
INTERNET ARCHIVE
WayBackMachine
Explore more than 916 billion web pages saved over time
http://books.elsevier.com/us/mk/us/subindex.asp?maintarget=&isbn=&c
Calendar · Collections · Changes · Summary · Site Map · URLs
Saved 85 times between November 22, 2005 and December 25, 2018.
2002 2003 2004 2005 2006 2007 2008 2009 2010 2011 2012 2013 2014 2015 2016 2017 2018 2019 2020 2021 2022 2023 2024
JAN FEB MAR APR
MAY JUN JUL AUG
SEP OCT NOV DEC
Note
This calendar view maps the number of times http://books.elsevier.com/us/mk/us/subindex.asp?maintarget=&isbn=&country=United+States&srccode=&ref=&subcode=&head=&pdf=&basiccode=&txtSearch=&SearchField=&operator=&order=&community=mk was crawled by the Wayback Machine, not how many times the site was actually updated. More info in the FAQ.
FAQ | Contact Us | Terms of Service (Dec 31, 2014)

https://web.archive.org/web/20051122210123/http://books.elsevier.com/us/mk/us/subindex.asp?maintarget=&isbn=&country=United+States&srccode=&ref=&subcode=&head=&pdf=&basiccode=&txtSearch=&SearchField=&operator=&order=&community=mk December 23, 2024

INTERNET ARCHIVE Wayback Machine
http://books.elsevier.com/us/mk/us/subindex.asp?maintarget=&isbn=&country=United+States&srccode=&ref=&subcode=&head=&pdf=&basiccode=&txtSearch=&Sear Go
85 captures
22 Nov 2005 - 25 Dec 2018
OCT NOV DEC
22
2004 2005 2006
About this capture

Monthly Quick-Scan Email

How about a monthly heads-up on the best of brand-new books published by Morgan Kaufmann?

Quick Sign up

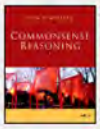
Customer Service

For general enquiries please email usbkinfo @elsevier.com

More about us

About us
Order a catalog
Reader feedback
Send us a book proposal

Just off the press at MK!!!



Commonsense Reasoning
Erik T. Mueller
Price: $49.95
ISBN: 0-12-369388-8
Published: January 19 2006
• Covers key areas of commonsense reasoning including action, change, defaults, space, and mental states • ... [more]



Data Mining, Second Edition
Concepts and Techniques
Jiawei Han
Price: $59.95
ISBN: 1-55860-901-6
Published: March 1 2006
This is the second edition of the premier professional reference on the subject of data mining, expanding ... [more]



The Persona Lifecycle
Keeping People in Mind Throughout Product Design
John Pruitt
Price: $59.95
ISBN: 0-12-566251-3
Published: April 28 2006

More NEW titles from MK!

Programming with Quartz
2D and PDF Graphics in Mac OS X
David Gelphman
Price: $69.95
ISBN: 0-12-369473-6
Published: 12/14/2005 available now!
* The first guide to the revolutionary new graphics system of Mac OS X Tiger that uses the Portable ... [more] / [related titles]

Introduction to Data Compression
Khalid Sayood
Price: $69.95
ISBN: 0-12-620862-X
Published: 12/1/2005 available now!
*35% NEW MATERIAL, plus each chapter is completely updated with new problems and solutions *ENTIRE ... [more] / [related titles]



High Dynamic Range Imaging
Acquisition, Display, and Image-Based Lighting
Erik Reinhard
Price: $69.95
ISBN: 0-12-585263-0
Published: 11/29/2005 available now!
* The first book on this important advance by the leading researchers * HDR will become the standard ... [more] / [related titles]

Browse our books

Artificial Intelligence
Business software / general
Computer Architecture & Design
Computer Graphics
Computer Networking
Data Management Systems
HCI/Interactive Technologies
MMIS
Operating systems
QuickTime
Security
Software engineering & programming
Storage & retrieval
The Practical Guides

https://web.archive.org/web/20051122210123/http://books.elsevier.com/us/mk/us/subindex.asp?maintarget=&isbn=&country=United+States&srccode=&ref=&subcode=&head=&pdf=&basiccode=&txtSearch=&SearchField=&operator=&order=&community=mk December 23, 2024

INTERNET ARCHIVE Wayback Machine http://books.elsevier.com/us/mk/us/subindex.asp?maintarget=&isbn=&country=United+States&srccode=&ref=&subcode=&head=&pdf=&basiccode=&txtSearch=&Sear Go OCT NOV DEC
85 captures 22 Nov 2005 - 25 Dec 2018 2004 22 2005 2006 About this capture

Published: April 28 2006
* Full color Presentation and discussion of the complete lifecycle of personas, to guide the designer ... [more]

Other recent releases



Joe Celko's SQL for Smarties
Advanced SQL Programming
Joe Celko
Price: $49.95
ISBN: 0-12-369379-9
Published: 7/29/2005 available now!
* Offers expert advice from a noted SQL authority and award-winning columnist--with an entertaining ... [more] / [related titles]



Foundations of Analog and Digital Electronic Circuits
Anant Agarwal
Price: $79.95
ISBN: 1-55860-735-8
Published: 7/18/2005 available now!
* Balances circuits theory with practical digital electronics applications * Illustrates concepts with ... [more] / [related titles]



Computers As Components
Principles of Embedded Computing System Design
Wayne Wolf
Price: $59.95
ISBN: 0-12-369459-0
Published: 7/15/2005 available now!



Distributed Systems Architecture
A Middleware Approach
Arno Puder
Price: $59.95
ISBN: 1-55860-648-3
Published: 11/16/2005 available now!
* Provides a comprehensive look at the architecture and design of middleware
* Includes a complete, ... [more] / [related titles]

Programming Language Pragmatics
Michael Scott
Price: $64.95
ISBN: 0-12-633951-1
Published: 11/7/2005 available now!
New Coverage + Addresses the most recent developments in programming language design - C99, C#, and ... [more] / [related titles]



Why Programs Fail
A Guide to Systematic Debugging
Andreas Zeller
Price: $54.95
ISBN: 1-55860-866-4
Published: 10/11/2005 available now!
* Winner of a 2006 Jolt Productivity Award for Technical Books
* Shows how to reproduce software ... [more] / [related titles]



Database Modeling and Design
Logical Design, Fourth Edition

https://web.archive.org/web/20051122210123/http://books.elsevier.com/us/mk/us/subindex.asp?maintarget=&isbn=&country=United+States&srccode=&ref=&subcode=&head=&pdf=&basiccode=&txtSearch=&SearchField=&operator=&order=&community=mk December 23, 2024

INTERNET ARCHIVE Wayback Machine
http://books.elsevier.com/us/mk/us/subindex.asp?maintarget=&isbn=&country=United+States&srccode=&ref=&subcode=&head=&pdf=&basiccode=&txtSearch=&Sear Go
85 captures
22 Nov 2005 - 25 Dec 2018
OCT NOV DEC
22
2004 2005 2006
About this capture

ISBN: 0-12-369459-0
Published: 7/15/2005 available now!
* Demonstrates concepts and techniques using two processors as case studies throughout the book: the ... [more] / [related titles]



Data Mining
Practical Machine Learning Tools and Techniques, Second Edition
Ian Witten
Price: $59.95
ISBN: 0-12-088407-0
Published: 6/8/2005 available now!
* Covers performance improvement techniques that work by transforming the input or output * Authors, ... [more] / [related titles]



Virtual Machines
Versatile Platforms for Systems and Processes
Jim Smith
Price: $64.95
ISBN: 1-55860-910-5
Published: 6/3/2005 available now!
* Examines virtual machine technologies across the disciplines that use them —operating systems, programming ... [more] / [related titles]



Comprehensive Functional Verification
The Complete Industry Cycle
Bruce Wile
Price: $59.95
ISBN: 0-12-751803-7
Published: 5/26/2005 available now!



Database Modeling and Design
Logical Design, Fourth Edition
Toby Teorey
Price: $49.95
ISBN: 0-12-685352-5
Published: 9/6/2005 available now!
* Covers the lifecycle of a database modeling and design project, from modeling language coverage, to ... [more] / [related titles]

Configuring Microsoft Outlook 2003
Sue Mosher
Price: $54.95
ISBN: 1-55558-326-1
Published: 8/26/2005 available now!
Readers will learn how to: * Configure the new Cached Exchange mode and RPC over HTTP connections in ... [more] / [related titles]

Complete Maya Programming Volume II
An In-depth Guide to 3D Fundamentals, Geometry, and Modeling
David Gould
Price: $59.95
ISBN: 0-12-088482-8
Published: 8/5/2005 available now!
* Offers authoritative guidance to artists illuminated by the author's unmatched professional experience * ... [more] / [related titles]



Comprehensive Functional Verification
The Complete Industry Cycle
Bruce Wile

https://web.archive.org/web/20051122210123/http://books.elsevier.com/us/mk/us/subindex.asp?maintarget=&isbn=&country=United+States&srccode=&ref=&subcode=&head=&pdf=&basiccode=&txtSearch=&SearchField=&operator=&order=&community=mk December 23, 2024

INTERNET ARCHIVE Wayback Machine http://books.elsevier.com/us/mk/us/subindex.asp?maintarget=&isbn=&country=United+States&srccode=&ref=&subcode=&head=&pdf=&basiccode=&txtSearch=&Sear Go OCT NOV DEC 22 2004 2005 2006 About this capture
85 captures
22 Nov 2005 - 25 Dec 2018

ISBN: 0-12-751803-7
Published: 5/26/2005 available now!
One of the biggest challenges in chip and system design is determining whether the hardware works correctly. ... [more] / [related titles]

Joe Celko's SQL Programming Style
Joe Celko
Price: $29.95
ISBN: 0-12-088797-5
Published: 4/17/2005 available now!
Are you an SQL programmer that, like many, came to SQL after learning and writing procedural or object-oriented ... [more] / [related titles]

Cost-Justifying Usability
An Update for the Internet Age, Second Edition
Randolph Bias
Price: $59.95
ISBN: 0-12-095811-2
Published: 4/4/2005 available now!
You just know that an improvement of the user interface will reap rewards, but how do you justify the ... [more] / [related titles]

New titles

A Practical Logic of Cognitive Systems
The Reach of Abduction: Insight and Trial
Dov M. Gabbay
Price: $160.00

The Complete Industry Cycle
Bruce Wile
Price: $59.95
ISBN: 0-12-751803-7
Published: 5/26/2005 available now!
One of the biggest challenges in chip and system design is determining whether the hardware works correctly. ... [more] / [related titles]

Joe Celko's SQL Programming Style
Joe Celko
Price: $29.95
ISBN: 0-12-088797-5
Published: 4/17/2005 available now!
Are you an SQL programmer that, like many, came to SQL after learning and writing procedural or object-oriented ... [more] / [related titles]

Network Security
A Practical Approach
Jan Harrington
Price: $44.95
ISBN: 0-12-311633-3
Published: 4/8/2005 available now!
Network Security is a comprehensive resource written for anyone who plans or implements network ... [more] / [related titles]

User Interface Design and Evaluation
Debbie Stone
Price: $59.95
ISBN: 0-12-088436-4

https://web.archive.org/web/20051122210123/http://books.elsevier.com/us/mk/us/subindex.asp?maintarget=&isbn=&country=United+States&srccode=&ref=&subcode=&head=&pdf=&basiccode=&txtSearch=&SearchField=&operator=&order=&community=mk December 23, 2024

INTERNET ARCHIVE Wayback Machine
http://books.elsevier.com/us/mk/us/subindex.asp?maintarget=&isbn=&country=United+States&srccode=&ref=&subcode=&head=&pdf=&basiccode=&txtSearch=&Sear Go
85 captures
22 Nov 2005 - 25 Dec 2018
OCT NOV 22 2005 DEC 2004 2006
About this capture

Dov M. Gabbay
Price: $160.00
ISBN: 0-444-51791-X
Published: 6/20/2005 available now!
The present work is a continuation of the authors' acclaimed multi-volume A Practical Logic of Cognitive ... [more] / [related titles]

Advances in Computers
Marvin Zelkowitz
Price: $145.00
ISBN: 0-12-012165-4
Published: 7/12/2005 available now!
This present volume describes some of the latest advances in the computer science field today. This ... [more] / [related titles]

Advances in Computers
New Programming Paradigms
Marvin Zelkowitz
Price: $140.00
ISBN: 0-12-012164-6
Published: 5/11/2005 available now!
The series covers new developments in computer technology. Most chapters present an overview of a current ... [more] / [related titles]

Advances in Computers
Parallel, Distributed, and Pervasive Computing
Marvin Zelkowitz
Price: $135.00
ISBN: 0-12-012163-8

Price: $59.95
ISBN: 0-12-088436-4
Published: 3/22/2005 available now!
Whether you are a professional new to the user-centered design field, or an experienced designer who ... [more] / [related titles]

Coming soon

Fuzzy Logic and the Semantic Web
Elie Sanchez
Price: $132.00
ISBN: 0-444-51948-3
Published: 4/3/2006 available here soon!
These are exciting times in the fields of Fuzzy Logic and the Semantic Web, and this book will add to ... [more] / [related titles]

Data Model Patterns
A Metadata Map
David Hay
Price: $59.95
ISBN: 0-12-088798-3
Published: 4/14/2006 available here soon!
This book shows the conceptual understanding plus the practical detail on how to model the business ... [more] / [related titles]

Expanding a Digital Content Management System
for the Growing Digital Media Enterprise
Magan Arthur
Price: $39.95

https://web.archive.org/web/20051122210123/http://books.elsevier.com/us/mk/us/subindex.asp?maintarget=&isbn=&country=United+States&srccode=&ref=&subcode=&head=&pdf=&basiccode=&txtSearch=&SearchField=&operator=&order=&community=mk December 23, 2024

INTERNET ARCHIVE WaybackMachine http://books.elsevier.com/us/mk/us/subindex.asp?maintarget=&isbn=&country=United+States&srccode=&ref=&subcode=&head=&pdf=&basiccode=&txtSearch=&Sear Go OCT NOV DEC 22 2004 2005 2006 About this capture
85 captures
22 Nov 2005 - 25 Dec 2018



Advances in Computers
Parallel, Distributed, and Pervasive Computing
Marvin Zelkowitz
Price: $135.00
ISBN: 0-12-012163-8
Published: 4/19/2005 available now!
The term computation gap has been defined as the difference between the computational power demanded ... [more] / [related titles]



An Ontological and Epistemological Perspective of Fuzzy Set Theory
I. Burhan Türksen
Price: $160.00
ISBN: 0-444-51891-6
Published: 12/1/2005 available now!
Fuzzy set and logic theory suggest that all natural language linguistic expressions are imprecise and ... [more] / [related titles]



BULLETPROOF WIRELESS SECURITY
GSM, UMTS, 802.11, and Ad Hoc Security
Praphul Chandra
Price: $49.99
ISBN: 0-7506-7746-5
Published: 6/14/2005 available now!
* Provides a solid foundation in RF/wireless engineering as well as software-based security methods * ... [more] / [related titles]



Expanding a Digital Content Management System
for the Growing Digital Media Enterprise
Magan Arthur
Price: $39.95
ISBN: 0-240-80794-4
Published: 4/14/2006 available here soon!
This is the first book that addresses the challenges of expaning an outgrown DCM system. It's a true ... [more] / [related titles]



Wireless Networks
From the Physical Layer to Communication, Computing, Sensing and Control
Giorgio Franceschetti
Price: $79.95
ISBN: 0-12-369426-4
Published: 4/21/2006 available here soon!
* Presents a uniquely panoramic view of wireless networks with viewpoints from engineering, computing, ... [more] / [related titles]



The Persona Lifecycle
Keeping People in Mind Throughout Product Design
John Pruitt
Price: $59.95
ISBN: 0-12-566251-3
Published: 4/28/2006 available here soon!
* Full color Presentation and discussion of the complete lifecycle of personas, to guide the designer ... [more] / [related titles]

https://web.archive.org/web/20051122210123/http://books.elsevier.com/us/mk/us/subindex.asp?maintarget=&isbn=&country=United+States&srccode=&ref=&subcode=&head=&pdf=&basiccode=&txtSearch=&SearchField=&operator=&order=&community=mk December 23, 2024

INTERNET ARCHIVE Wayback Machine http://books.elsevier.com/us/mk/us/subindex.asp?maintarget=&isbn=&country=United+States&srccode=&ref=&subcode=&head=&pdf=&basiccode=&txtSearch=&Sear Go OCT NOV DEC 22 2004 2005 2006 About this capture
85 captures 22 Nov 2005 - 25 Dec 2018

Price: $160.00
ISBN: 0-444-51891-6
Published: 12/1/2005 available now!
Fuzzy set and logic theory suggest that all natural language linguistic expressions are imprecise and ... [more] / [related titles]



BULLETPROOF WIRELESS SECURITY
GSM, UMTS, 802.11, and Ad Hoc Security
Praphul Chandra
Price: $49.99
ISBN: 0-7506-7746-5
Published: 6/14/2005 available now!
* Provides a solid foundation in RF/wireless engineering as well as software-based security methods * ... [more] / [related titles]



Control
Giorgio Franceschetti
Price: $79.95
ISBN: 0-12-369426-4
Published: 4/21/2006 available here soon!
* Presents a uniquely panoramic view of wireless networks with viewpoints from engineering, computing, ... [more] / [related titles]



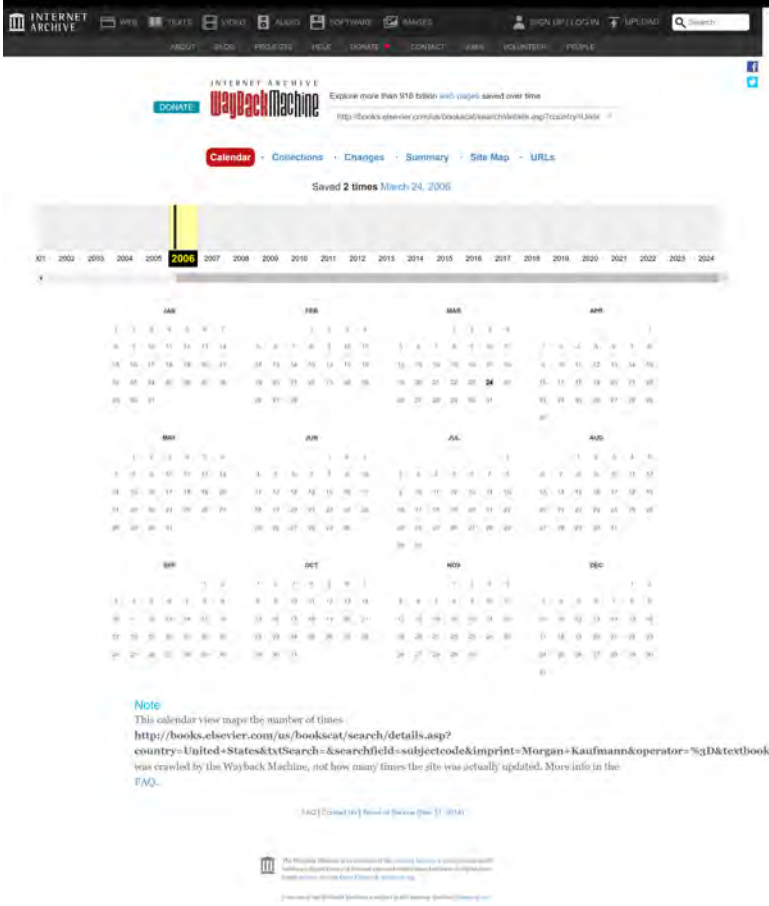
The Persona Lifecycle
Keeping People in Mind Throughout Product Design
John Pruitt
Price: $59.95
ISBN: 0-12-566251-3
Published: 4/28/2006 available here soon!
* Full color Presentation and discussion of the complete lifecycle of personas, to guide the designer ... [more] / [related titles]



Advances in Computers
Quality Software Development
Marvin Zelkowitz
Price: $140.00
ISBN: 0-12-012166-2
Published: 5/31/2006 available here soon!
This volume of Advances in Computers is number 66 in the series that began back in 1960. This series ... [more] / [related titles]

© 2006, Elsevier | www.elsevier.com

APPENDIX SMITH06



Note

This calendar view maps the number of times
http://books.elsevier.com/us/bookscat/search/details.asp?country=United+States&txtSearch=&searchfield=subjectcode&imprint=Morgan+Kaufmann&operator=%3D&textbook=&highlight=0&display=collapse&isbn=1558609105&srccode=&community=mk&ref=&mscssid=LJUXMV8TRRDW9L1T72DK85TF888R692F1
was crawled by the Wayback Machine, not how many times the site was actually updated. More info in the
FAQ.

https://web.archive.org/web/20060324094915/http://books.elsevier.com/us/bookscat/search/details.asp?country=United+States&txtSearch=&searchfield=subjectcode&imprint=Morgan+Kaufmann&operator=%3D&textbook=&highlight=0&display=collapse&isbn=1558609105&srccode=&community=mk&ref=&mscssid=LJUXMV8TRRDW9LH72DK85TF888R692F1 December 23, 2024

INTERNET ARCHIVE Wayback Machine http://books.elsevier.com/us/bookscat/search/details.asp?country=United+States&txtSearch=&searchfield=subjectcode&imprint=Morgan+Kaufmann&operator=%3D&t Go FEB MAR APR
2 captures 24 Mar 2006 2005 24 2006 2007 About this capture

This page is designed to work as part of a frameset and some functionality will not be available in your current view.

If you were already on our site and opened this page in a **new window**, please use the **original window** to continue browsing.

If you were directed here by **another site**, please visit our **main page** or select a subject area site, so that we can redirect you to this same page within that site.

Select a subject site

If you want, you can tell us the URL of the page that directed you here.

Send now

Price: $64.95
Quantity: 1
Offer code:
add to cart

Virtual Machines

Versatile Platforms for Systems and Processes

Features & Benefits | Description | Contents

Jim Smith
University of Wisconsin
Ravi Nair
IBM, Thomas J. Watson Research Center, New York

Click here for a larger image
send to a friend

Features & Benefits

- Examines virtual machine technologies across the disciplines that use them—operating systems, programming languages and computer architecture—defining a new and unified discipline.
- Reviewed by principle researchers at Microsoft, HP, and by other industry research groups.
- Written by two authors who combine several decades of expertise in computer system research and development, both in academia and industry.

Description

back to top

Virtual Machine technology applies the concept of virtualization to an entire machine, circumventing real machine compatibility constraints and hardware resource constraints to enable a higher degree of software portability and flexibility. Virtual machines are rapidly becoming an essential element in computer system design. They provide system security, flexibility, cross-platform compatibility, reliability, and resource efficiency. Designed to solve problems in combining and using major computer system components, virtual machine technologies play a key role in many disciplines, including operating systems, programming languages, and computer architecture. For example, at the process level, virtualizing technologies support dynamic program translation and platform-independent network computing. At the system level, they support multiple operating system environments on the same hardware platform and in servers.

Historically, individual virtual machine techniques have been developed within the specific disciplines that employ them (in some cases they aren't even referred to as "virtual machines"), making it difficult to see their common underlying relationships in a cohesive way. In this text, Smith and Nair take a new approach by examining virtual machines as a unified discipline. Pulling together cross-cutting technologies allows virtual machine implementations to be studied and engineered in a well-structured manner. Topics include instruction set emulation, dynamic program translation and optimization, high level virtual machines (including Java and CLI), and system virtual machines for both single-user systems and servers.

Contents

back to top

Foreword
Preface
Chapter 1: Introduction to Virtual Machines
Chapter 2: Emulation: Interpretation and Binary Translation
Chapter 3: Process Virtual Machines
Chapter 4: Dynamic Binary Optimization
Chapter 5: High-Level Language Virtual Machine Architecture
Chapter 6: High-Level virtual Machine Implementation
Chapter 7: Codesigned Virtual Machines
Chapter 8: System Virtual Machines
Chapter 9: Multiprocessor Virtualization
Chapter 10: Emerging Applications
Appendix A: Real Machines
References

Readership: Designers and developers in OS, Programming languages, and architecture; researchers in academia and industry; and graduate students in computer science and engineering

Series: The Morgan Kaufmann Series in Computer Architecture and Design

ISBN: 1-55860-910-5 Book/Hardback

Line Illustrations: 303
Measurements: 7 1/2 X 9 1/4 in
Pages: 656

Imprint: Morgan Kaufmann

Publication Date: 3 June 2005

Price: $64.95

© 2006, Elsevier | www.elsevier.com