参考文献

1. Hennessy J L, Patterson D A. A new golden age for computer architecture[J]. Communications of the ACM, 2019, 62(2): 48-60.
2. Celio C, Chiu PF, Asanović K, et al. Broom: an open-source out-of-order processor with resilient low-voltage operation in 28-nm CMOS [J]. IEEE Micro, 2019, 39(2):52-60.
3. XiangShan-doc. UCAS & ICT, PCL. 2021. <https://github.com/OpenXiangShan/XiangShan-doc>
4. LeBlanc T J, Mellor-Crummey J M. Debugging parallel programs with instant replay[J]. IEEE Transactions on Computers, 1987, 36(4): 471-482.
5. Di Tucci L, Baghdadi R, Amarasinghe S, et al. SALSA: A domain specific architecture for sequence alignment[C]//2020 IEEE International Parallel and distributed processing symposium workshops (IPDPSW). IEEE, 2020: 147-150.
6. Navarro Leija O S, Shiptoski K, Scott R G, et al. Reproducible containers[C]. Proceedings of the Twenty-Fifth International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS). 2020: 167-182.
7. O'Callahan R, Jones C, Froyd N, et al. Engineering record and replay for deployability[C]. 2017 USENIX Annual Technical Conference (USENIX ATC 17), 2017: 377-389.
8. Narayanasamy S, Pokam G, Calder B. Bugnet: Continuously recording program execution for deterministic replay debugging[C]. 32nd International Symposium on Computer Architecture (ISCA'05), IEEE, 2005: 284-295.
9. Srinivasan S M, Kandula S, Andrews C R, et al. Flashback: A lightweight extension for rollback and deterministic replay for software debugging[C]. USENIX Annual Technical Conference, General Track, 2004: 29-44.
10. Saito Y. Jockey: a user-space library for record-replay debugging[C]. Proceedings of the sixth international symposium on Automated analysis-driven debugging, 2005, 69-76.
11. Dunlap G W, King S T, Cinar S, et al. ReVirt: Enabling Intrusion Analysis Through Virtual-Machine Logging and Replay[C]. 5th Symposium on Operating Systems Design and Implementation (OSDI 02). 2002.
12. Sheldon M, Weissman G V B. Retrace: Collecting execution trace with virtual machine deterministic replay[C]. Proceedings of the Third Annual Workshop on Modeling, Benchmarking and Simulation (MoBS 2007). 2007.
13. Asanović K, Patterson D A. Instruction sets should be free: The case for risc-v[J]. EECS Department, University of California, Berkeley, Tech. Rep. UCB/EECS-2014-146, 2014.
14. Waterman A S. Design of the RISC-V instruction set architecture[M]. University of California, Berkeley, 2016.
15. Asanovic K, Avizienis R, Bachrach J, et al. The rocket chip generator[J]. EECS Department, University of California, Berkeley, Tech. Rep. UCB/EECS-2016-17, 2016, 4.
16. Herdt V, Große D, Pieper P, et al. RISC-V based virtual prototype: An extensible and configurable platform for the system-level[J]. Journal of Systems Architecture, 2020, 109: 101756.
17. Farshchi F, Huang Q, Yun H. Integrating NVIDIA deep learning accelerator (NVDLA) with RISC-V SoC on FireSim[C]//2019 2nd Workshop on Energy Efficient Machine Learning and Cognitive Computing for Embedded Applications (EMC2). IEEE, 2019: 21-25.
18. Karandikar S, Biancolin D, Amid A, et al. Using FireSim to Enable Agile End-to-End RISC-V Computer Architecture Research[J]. 2019.
19. Sá B, Martins J, Pinto S E S. A First Look at RISC-V Virtualization from an Embedded Systems Perspective[J]. IEEE Transactions on Computers, 2021.
20. Asanovic K, Avizienis R, Bachrach J, et al. The rocket chip generator[J]. EECS Department, University of California, Berkeley, Tech. Rep. UCB/EECS-2016-17, 2016, 4.
21. Asanovic K, Patterson D A, Celio C. The berkeley out-of-order machine (boom): An industry-competitive, synthesizable, parameterized risc-v processor[R]. University of California at Berkeley Berkeley United States, 2015.
22. Chen C, Xiang X, Liu C, et al. Xuantie-910: A commercial multi-core 12-stage pipeline out-of-order 64-bit high performance RISC-V processor with vector extension: Industrial product[C]//2020 ACM/IEEE 47th Annual International Symposium on Computer Architecture (ISCA). IEEE, 2020: 52-64.
23. XiangShan-doc. UCAS & ICT, PCL. 2021. https://github.com/OpenXiangShan/XiangShan-doc
24. Feng E, Lu X, Du D, et al. Scalable Memory Protection in the PENGLAI Enclave[C]//15th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 21). 2021: 275-294.
25. Bergan T, Hunt N, Ceze L, et al. Deterministic Process Groups in dOS[C]. 9th USENIX Symposium on Operating Systems Design and Implementation (OSDI 10), 2010.
26. Devecsery D, Chow M, Dou X, et al. Eidetic systems[C]. 11th USENIX Symposium on Operating Systems Design and Implementation (OSDI 14). 2014: 525-540.
27. Pan D Z, Linton M A. Supporting reverse execution for parallel programs[C]//Proceedings of the 1988 ACM SIGPLAN and SIGOPS Workshop on Parallel and Distributed Debugging. 1988: 124-129.
28. Bacon, D.F., Goldstein, S.C.: Hardware-assisted Replay of Multiprocessor Programs. In: Proceedings of the 1991 ACM/ONR Workshop on Parallel and Distributed Debugging. pp. 194–206. PADD ’91, ACM, New York, NY, USA (1991), DOI: 10.1145/122759.122777
29. Patil H, Pereira C, Stallcup M, et al. Pinplay: a framework for deterministic replay and reproducible analysis of parallel programs[C]//Proceedings of the 8th annual IEEE/ACM international symposium on Code generation and optimization. 2010: 2-11.
30. Bhansali S, Chen W K, De Jong S, et al. Framework for instruction-level tracing and analysis of program executions[C]//Proceedings of the 2nd international conference on Virtual execution environments. 2006: 154-163.
31. Xu M, Bodik R, Hill M D. A" flight data recorder" for enabling full-system multiprocessor deterministic replay[C]//Proceedings of the 30th annual international symposium on Computer architecture. 2003: 122-135.
32. Hower D R, Hill M D. Rerun: Exploiting episodes for lightweight memory race recording[J]. ACM SIGARCH computer architecture news, 2008, 36(3): 265-276.
33. Montesinos P, Ceze L, Torrellas J. Delorean: Recording and deterministically replaying shared-memory multiprocessor execution ef? ciently[J]. ACM SIGARCH Computer Architecture News, 2008, 36(3): 289-300.
34. Pokam G, Danne K, Pereira C, et al. QuickRec: Prototyping an Intel architecture extension for record and replay of multithreaded programs[C]//Proceedings of the 40th annual international symposium on computer architecture. 2013: 643-654.
35. Devietti J, Lucia B, Ceze L, et al. DMP: Deterministic shared memory multiprocessing[C]//Proceedings of the 14th international conference on Architectural support for programming languages and operating systems. 2009: 85-96.
36. Bergan T, Anderson O, Devietti J, et al. CoreDet: A compiler and runtime system for deterministic multithreaded execution[C]//Proceedings of the fifteenth International Conference on Architectural support for programming languages and operating systems. 2010: 53-64.
37. Hower D R, Dudnik P, Hill M D, et al. Calvin: Deterministic or not? free will to choose[C]//2011 IEEE 17th International Symposium on High Performance Computer Architecture. IEEE, 2011: 333-334.
38. Devietti J, Nelson J, Bergan T, et al. RCDC: a relaxed consistency deterministic computer[J]. ACM SIGARCH Computer Architecture News, 2011, 39(1): 67-78.
39. Liu T, Curtsinger C, Berger E D. Dthreads: efficient deterministic multithreading[C]//Proceedings of the Twenty-Third ACM Symposium on Operating Systems Principles. 2011: 327-336.
40. Merrifield T, Eriksson J. Conversion: Multi-version concurrency control for main memory segments[C]//Proceedings of the 8th ACM European Conference on Computer Systems. 2013: 127-139.
41. Lu K, Zhou X, Bergan T, et al. Efficient deterministic multithreading without global barriers[J]. ACM SIGPLAN Notices, 2014, 49(8): 287-300.
42. Merrifield T, Devietti J, Eriksson J. High-performance determinism with total store order consistency[C]//Proceedings of the Tenth European Conference on Computer Systems. 2015: 1-13.
43. Merrifield T, Roghanchi S, Devietti J, et al. Lazy determinism for faster deterministic multithreading[C]//Proceedings of the Twenty-Fourth International Conference on Architectural Support for Programming Languages and Operating Systems. 2019: 879-891.
44. Ronsse M, De Bosschere K. RecPlay: A fully integrated practical record/replay system[J]. ACM Transactions on Computer Systems (TOCS), 1999, 17(2): 133-152.
45. Devecsery D, Chow M, Dou X, et al. Eidetic systems[C]//11th USENIX Symposium on Operating Systems Design and Implementation (OSDI 14). 2014: 525-540.
46. Lee D, Wester B, Veeraraghavan K, et al. Respec: efficient online multiprocessor replay via speculation and external determinism[J]. ACM Sigplan Notices, 2010, 45(3): 77-90.
47. Mashtizadeh A J, Garfinkel T, Terei D, et al. Towards practical default-on multi-core record/replay[J]. ACM SIGPLAN Notices, 2017, 52(4): 693-708.
48. Aviram A, Hu S, Ford B, et al. Determinating timing channels in compute clouds[C]//Proceedings of the 2010 ACM workshop on Cloud computing security workshop. 2010: 103-108.
49. Hunt N, Bergan T, Ceze L, et al. DDOS: taming nondeterminism in distributed systems[J]. ACM SIGPLAN Notices, 2013, 48(4): 499-508.
50. Bergan T, Hunt N, Ceze L, et al. Deterministic Process Groups in dOS[C]. 9th USENIX Symposium on Operating Systems Design and Implementation (OSDI 10), 2010.
51. 徐子晨, 崔傲, 王玉皞, 等. 基于 RISC-V 架构的强化学习容器化方法研究[J]. 计算机工程与科学, 2021.
52. Popek G J, Goldberg R P. Formal requirements for virtualizable third generation architectures[J]. Communications of the ACM, 1974, 17(7): 412-421.
53. Bellard F. QEMU, a fast and portable dynamic translator[C]//USENIX annual technical conference, FREENIX Track. 2005, 41(46): 10.5555.
54. Bernstein D. Containers and Cloud: From LXC to docker to Kubernetes[J]. IEEE Cloud Computing, 2014, 1(3): 81-84.
55. Dunlap G W, Lucchetti D G, Fetterman M A, et al. Execution replay of multiprocessor virtual machines[C]//Proceedings of the fourth ACM SIGPLAN/SIGOPS international conference on Virtual execution environments. 2008: 121-130.
56. Sheldon M, Weissman G V B. Retrace: Collecting execution trace with virtual machine deterministic replay[C]//Proceedings of the Third Annual Workshop on Modeling, Benchmarking and Simulation (MoBS 2007). 2007.
57. Dolan-Gavitt B, Hodosh J, Hulin P, et al. Repeatable reverse engineering with PANDA[C]//Proceedings of the 5th Program Protection and Reverse Engineering Workshop. 2015: 1-11.
58. Dovgalyuk P. Deterministic Replay of System's Execution with Multi-target QEMU Simulator for Dynamic Analysis and Reverse Debugging[C]//CSMR. 2012: 553-556.
59. Srinivasan D, Jiang X. Time-traveling forensic analysis of vm-based high-interaction honeypots[C]//International Conference on Security and Privacy in Communication Systems. Springer, Berlin, Heidelberg, 2011: 209-226.
60. Burtsev A, Johnson D, Hibler M, et al. Abstractions for practical virtual machine replay[J]. ACM SIGPLAN Notices, 2016, 51(7): 93-106.
61. Bhansali S, Chen W K, De Jong S, et al. Framework for instruction-level tracing and analysis of program executions[C]//Proceedings of the 2nd international conference on Virtual execution environments. 2006: 154-163.
62. Olszewski M, Ansel J, Amarasinghe S. Kendo: efficient deterministic multithreading in software[C]//Proceedings of the 14th international conference on Architectural support for programming languages and operating systems. 2009: 97-108.
63. Roelke A, Stan M R. Risc5: Implementing the RISC-V ISA in gem5[C]//First Workshop on Computer Architecture Research with RISC-V (CARRV). 2017, 7(17).
64. Ta T, Cheng L, Batten C. Simulating multi-core RISC-V systems in gem5[C]//Workshop on Computer Architecture Research with RISC-V. 2018.
65. Herdt V, Große D, Pieper P, et al. RISC-V based virtual prototype: An extensible and configurable platform for the system-level[J]. Journal of Systems Architecture, 2020, 109: 101756.
66. Hin P Y H, Liao X, Cui J, et al. Supporting RISC-V Full System Simulation in gem5[J]. 2021.