연구산출물 블라인드 처리 가이드라인

한국조선해양기자재연구원 공개채용은 「**평등한 기회, 공정한 과정을 위한 공공기관 블라인드 채용**」을 따르고 있습니다. 이에 지원서 작성 시 첨부하는 연구산출물 증빙자료 블라인드 처리방법에 대해 다음과 같이 안내드리며 반드시 유의사항을 숙지하시어 전형과정에서 불이익을 받지 않도록 유의 바랍니다.

1. 논문(학위논문 초록 포함) 실적의 블라인드 처리 가이드

가. 블라인드 처리 사항

- 저자 인적사항 : 지원자 본인을 포함한 모든 저자의 성명, 소속, e-mail(출신학교 노출 가능) 등
- 사사문구(Acknowledgement)
- 학위논문(초록 포함) 내 학교 워터마크(watermark)
- 첨부파일명은 논문1, 논문2와 같이 변경

나. 블라인드 미처리 사항

- 게제 학술지(저널)명, 논문명, 논문 정보(게재권호, ISSN 등)

2. 특허 실적의 블라인드 처리 가이드

가. 블라인드 처리 사항

- 특허권자 인적사항 : 지원자 본인을 포함한 모든 공동발명자 성명, 주소, 소속(출신학교 노출 가능) 등
- 사사문구(Acknowledgement)
- 첨부파일명은 특허1, 특허2와 같이 변경

나. 블라인드 미처리 사항

- 특허번호, 등록일자 및 발명의 명칭 등 기본정보

3. 기타 사항

가. 학술대회 발표 논문, 연구보고서, 소프트웨어(프로그램) 등 연구산출물 및 자격증 : 논문, 특허에 관한 블라인드 처리 가이드를 동일하게 적용

나. 블라인드 처리하여 제출하는 연구산출물에 대해 2페이지 이내의 요약서 함께 제출

한국조선해양기자재연구원
Korea Marine Equipment Research Institute

블라인드 처리 샘플(논문)

Future Generation Computer Systems 70 (2017) 26–41

Contents lists available at ScienceDirect

Future Generation Computer Systems

journal homepage: www.elsevier.com/locate/fgcs

Effects of dynamic isolation for full virtualized RTOS and GPOS guests

[redacted] 저자 이름: 블라인드처리

[redacted]

저자 소속: 블라인드처리

HIGHLIGHTS

- We examine and analyze how a RTOS VM and a GPOS VM interact and influence each other.
- We analyze the explicit and implicit effects of dynamic isolation for vCPUs.
- The dynamic isolation shows low scheduling delay of RTOS and high throughput of GPOS.
- All of the proposed concepts are implemented on a full-fledged hypervisor.

ARTICLE INFO

Article history:
Received 14 May 2016
Received in revised form
25 October 2016
Accepted 17 December 2016
Available online 23 December 2016

Keywords:
Embedded virtualization
Dynamic isolation
Mixed criticality system
vCPU scheduling
Full virtualization

ABSTRACT

Industrial systems currently include not only control processing with real-time operating system (RTOS) but also information processing with general-purpose operating system (GPOS). Multicore-based virtualization is an attractive option to provide consolidated environment when GPOS and RTOS are put in service on a single hardware platform. Researches on this technology have predominantly focused on the schedulability of RTOS virtual machines (VMs) by completely dedicated physical-CPUs (pCPUs) but have rarely considered parallelism or the throughput of the GPOS. However, it is also important that the multicore-based hypervisor adaptively selects pCPU assignment policy to efficiently manage resources in modern industrial systems. In this paper, we report our study on the effects of dynamic isolation when two mixed criticality systems are working on one platform. Based on our investigation of mutual interferences between RTOS VMs and GPOS VMs, we found explicit effects of dynamic isolation by special events. While maintaining low RTOS VMs scheduling latency, a hypervisor should manage pCPUs assignment by event-driven and threshold-based strategies to improve the throughput of GPOS VMs. Furthermore, we deal with implicit negative effects of dynamic isolation caused by the synchronization inside a GPOS VM, then propose a process of urgent boosting with dynamic isolation. All our methods are implemented in a real hypervisor, KVM. In experimental evaluation with benchmarks and an automotive digital cluster application, we analyzed that proposed dynamic isolation guarantees soft real-time operations for RTOS tasks while improving the throughput of GPOS tasks on a virtualized multicore system.

1. Introduction

Traditional industrial computers consist of control processing software for simple missions. In recent years, industrial systems (e.g., consumer electronics, automobile, aeronautic sectors, smart phone, factory automation and grid computing) have been launched with more powerful devices, to interface to more networks and sensing devices. Moreover, a larger variety of application software is now required, with different levels of quality and service. This trend has also increased the number and volume of electronic units, as well as their power requirements. Thus, the job of software now includes not only hardware control but also information processing for sensing devices. In general, real-time operating systems (RTOS) are used as control processing software, since its tasks are mostly time-critical, whereas information processing software can be written on top of general purpose operating systems (GPOS) to maximize throughput. Those systems can be consolidated into a single system by multicore hardware and virtualization techniques. Some research can be found in industrial domains that require RTOS and GPOS applications to be simultaneously executed on a single multicore-based virtualized platform [1,2]. As illustrated in Fig. 1,

[redacted]

http://dx.doi.org/10.1016/j.future.2016.12.020

저자 이름, 저자 관련 정보(E-mail, 연락처): 블라인드처리

저자 이름: 블라인드처리

simultaneously, performance of the mixed system significantly improved. All works were implemented in a well-known open source hypervisor, KVM.

In addition, the discussion section treats limitations of this research and areas of application for the future works. Nevertheless, our research introduced novel work to explore and achieved improvements in the operation of VMs in a multicore-based virtualization system with a RTOS and a GPOS running simultaneously.

Acknowledgments

사사문구: 블라인드처리

[redacted]

References

[1] W. River, Applying multi-core and virtualization to industrial and safety-related application. http://leadwise.mediadroit.com/files/6535WP_Multicore_for_Industrial_and_Safety_Feb2009.pdf (accessed on 24.10.16).
[2] G. Heiser, Virtualizing embedded systems: why bother? in: Proceedings of the 48th Design Automation Conference, 2011, pp. 901–905.
[3] Frank Rowand, Using and Understanding the Real-Time Cyclictest Benchmark, Embedded linux conference 2013. http://elinux.org/images/0/01/Elc2013_rowand.pdf (accessed on 24.10.16).
[4] Felipe Cerqueira, Bjorn Brandenburg, A Comparison of Scheduling Latency in Linux, PREEMPT RT, and LITMUSRT, in: 9th annual workshop on Operating Systems Platforms for Em-bedded Real-Time applications July 9, 2013. Paris, France.
[5] Ruhui Ma, Fanfu Zhou, Erzhou Zhu, Haibing Guan, Performance tuning towards a KVM-based embedded real-time virtualization system, J. Inf. Sci. Eng. 29 (2013) 1021–1035.
[6] J. Kiszka, Towards Linux as a real-time hypervisor, in: Proceedings of the 11th Real-Time Linux Workshop, 2009, pp. 205–215.
[7] Tomoki Sekiyama, Improvement of Real-time Performance of KVM CloudOpen 2012. http://events.linuxfoundation.org/images/stories/pdf/lcna_co2012_sekiyama.pdf (accessed on 24.10.16).
[8] Seehwan Yoo, et al., Mobivmm: a virtual machine monitor for mobile phones, in: Proceedings of the First Workshop on Virtualization in Mobile Computing, ACM, 2008.
[9] L. Cereia, M. Bertolotti, Asymmetric virtualisation for real-time systems, in: IEEE International Symposium on Industrial Electronics, 2008, ISIE 2008, IEEE, Cambridge, 2008, pp. 1680–1685.
[10] H. Takada, S. Iiyama, T. Kindaichi, S. Hachiya, Linux on ITRON: A hybrid operating system architecture for embedded systems, in: 2002 Symposium on Applications and the Internet (SAINT) Workshops, 2002. Proceedings, IEEE, IEEE Computer Society, Washington, DC, USA, 2002, pp. 4–7.
[11] D. Sangorrin, S. Honda, H. Takada, Dual operating system architecture for real-time embedded systems, in: Proceedings of the 6th International Workshop on Operating Systems Platforms for Embedded Real-Time Applications, OSPERT, Brussels, Belgium, 2010.
[12] Kevin Scharpf, Dave Bryan, Mike Anderson, The Use of Carrier Grade Linux in Space (2007), in: Proceedings of the AI-AA/USU Conference on Small Satellites, Technical Session XII: Software, SSC07-XII-9.
[13] Christopher Huffine, Linux on a small satellite, Linux J. 2005 (132) (2005) 9.
[14] Kara Nance, Brian Hay, Matt Bishop, Virtual machine introspection, IEEE Comput. Soc. 6 (05) (2008) 32–37.
[15] T. Friebel, S. Biemueller, How to deal with lock holder preemption Presentation at Xen Summit North America, 2008.
[16] Xiaoning Ding, Phillip B. Gibbons, Michael A. Kozuch, A Hidden Cost of Virtualization when Scaling Multicore Applica-tions, in: 5th USENIX Workshop on hot topics in clound compu-ting(HotCloud'13), Jun 2013, San Jose, CA.
[17] A. Kivity, U. Lublin, A. Liguori, KVM: the Linux virtual machine monitor, in: Proceedings of the Linux Symposium, Vol. 1. 2007.
[18] Hyunwoo Joe, et al., Full virtualizing micro hypervisor for spacecraft flight computer, in: 2012 IEEE/AIAA 31st Digital Avionics Systems Conference, (DASC), IEEE, 2012, pp. 6C5-1–6C5-9.
[19] J. Calandrino, H. Leontyev, A. Block, U. Devi, J. Anderson, LITMUSRT: A testbed for empirically comparing real-time multiprocessor schedulers, in: Real-Time Systems Symposium, 2006. (RTSS'06). 27th IEEE International, IEEE, 2006, pp. 111–123.
[20] Real-time linux cyclictest https://rt.wiki.kernel.org/index.php/Cyclictest (accessed on 24.10.16).
[21] Christian Bienia, et al., The PARSEC benchmark suite: Characterization and architectural implications, in: Proceedings of the 17th International Conference on Parallel Architectures and Compilation Techniques, ACM, 2008.
[22] Yoav Etsion, Dan Tsafrir, Dror G. Feitelson, Effects of clock resolution on the scheduling of interactive and soft real-time processes, in: ACM Sigmetrics Performance Evaluation Review, Vol. 31, ACM, 2003, pp. 172–183. no. 1.
[23] Mathieu Desnoyers, Michel Dagenais, LTTng: Tracing across execution layers, from the hypervisor to user-space. in: Linux Symposium, Vol. 101, 2008.
[24] Perf Tool. https://perf.wiki.kernel.org/ (accessed on 24.10.16).
[25] Intel. Intel 64 and ia-32 architectures software developer's manual, volume 3b: System programming guide, part 2, 2010.
[26] Orathai Sukwong, Hyong S. Kim, Is co-scheduling too expensive for SMP VMs? in: Proceedings of the sixth conference on Computer systems, (Eurosys), ACM, 2011.
[27] Wei Jiang, et al., CPU optimizations to kvm threads on multi-core environments, in: 2009 15th International Conference on Parallel and Distributed Systems, (ICPADS), IEEE, 2009.
[28] Hwanju Kim, et al., Demand-based coordinated scheduling for SMP VMs, in: Proceedings of the Eighteenth International Conference on Architectural Support for Programming Languages and Operating Systems, ACM, 2013.
[29] Hiroshi Mitake, et al., Using virtual CPU migration to solve the lock holder preemption problem in a multicore processor-based virtualization layer for embedded systems, in: 2012 IEEE 18th International Conference on Embedded and Real-Time Computing Systems and Applications, (RTCSA), IEEE, 2012.
[30] V. Milanovic, A. Kasturi, V. Hachtel, High brightness mems mirror based head-up display (hud) modules with wireless data streaming capability, in: SPIE OPTO, International Society for Optics and Photonics, 2015, pp. 93 750A–93 750A.
[31] Chiyoung Lee, Se-Won Kim, Chuck Yoo, VADI: GPU virtualization for an automotive platform, IEEE Trans. Ind. Inf. 12 (1) (2016) 277–290.
[32] Lui Sha, et al., Single Core Equivalent Virtual Machines for Hard Real–Time Computing on Multicore Processors, 2014, https://www.ideals.illinois.edu/handle/2142/55672 (accessed on 24.10.16).
[33] U.M. Tai-Won, et al., Dynamic resource allocation and scheduling for cloud-based virtual content delivery networks, ETRI J. 36 (2) (2014) 197–205.
[34] Attila Kertesz, Gabor Kecskemeti, Ivona Brandic, Auto-nomic sla-aware service virtualization for distributed systems, in: 2011 19th Euromicro International Conference on Parallel, Distributed and Network-Based Processing, (PDP), IEEE, 2011.
[35] Liang Zhao, Sherif Sakr, Anna Liu, Consumer-centric SLA manager for cloud-hosted databases, in: Proceedings of the 22nd ACM International Conference on Conference on Information & Knowledge Management, ACM, 2013.
[36] Tatsuo Nakajima, Yuki Kinebuchi, Hiromasa Shimada, Alexandre Courbot, Tsung-Han Lin, Temporal and spatial isolation in a virtualization layer for multi-core processor based information appliances, in: Proceedings of the 16th Asia and South Pacific Design Automation Conference, 2011.
[37] Gilles Kathama, et al., Proteus hypervisor: Full virtualization and paravirtualization for multi-core embedded systems, in: Embedded Systems: Design, Analysis and Verification, Springer, Berlin, Heidelberg, 2013, pp. 293–305.
[38] P. Garcia, et al. Towards hardware embedded virtualization technology: architectural enhancements to an ARM SoC, in: VtRES 2013 Workshop on Virtualization for Real-Time Embedded Systems August 21st, 2013, Taipei, Taiwan.

[redacted]

저자 이름, 저자 관련 정보(E-mail, 연락처): 블라인드처리

블라인드 처리 샘플(특허)

(19) 대한민국특허청(KR)
(12) 등록특허공보(B1)

(45) 공고일자 2013년12월05일
(11) 등록번호 10-1337444
(24) 등록일자 2013년11월29일

(51) 국제특허분류(Int. Cl.)
H04L 12/02 (2006.01)
(21) 출원번호 10-2010-0113959
(22) 출원일자 2010년11월16일
심사청구일자 2012년04월12일
(65) 공개번호 10-2012-0052686
(43) 공개일자 2012년05월24일
(56) 선행기술조사문헌
KR1020010073555 A
KR1020090065878 A
KR1020090021695 A

(73) 특허권자
[redacted]
(72) 발명자
[redacted]
(뒷면에 계속)
(74) 대리인
특허법인이상

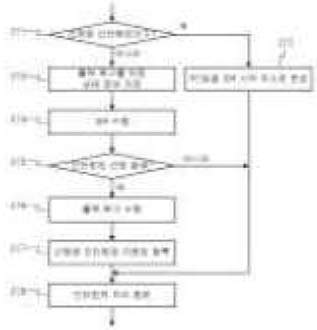
전체 청구항 수 : 총 10 항 심사관 : 이동하

(54) 발명의 명칭 **센서 네트워크 에뮬레이션 장치 및 그 동작 방법**

(57) 요 약

센서 네트워크의 에뮬레이션 성능을 향상시킬 수 있는 센서 네트워크 에뮬레이션 장치 및 그 동작 방법이 개시된다. 먼저, 발생된 인터럽트가 선정된 인터럽트 이벤트인가를 판단하고, 발생된 인터럽트가 선정된 인터럽트가 아닌 경우 완벽 복구를 위한 상태 정보를 저장하고, ISR(Interrupt Service Routine)을 비동기적으로 수행한다. 따라서, 높은 정확성을 유지하면서 에뮬레이션의 수행 시간을 향상시킬 수 있고, 이를 통해 센서 네트워크 응용 프로그램의 개발 시간을 단축시킬 수 있다.

대 표 도 - 도4



(72) 발명자
[redacted]

이 발명을 지원한 국가연구개발사업
과제고유번호
부처명
연구사업명
연구과제명
기 여 율
주관기관
연구기간
[redacted]