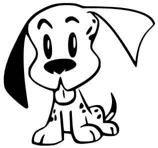
Nanvix

A Distributed Operating System for Lightweight Manycores

Pedro Henrique Penna
pedro.penna@sga.pucminas.br



Introduction

Lightweight (LW) Manycores: Performance Scalability and Energy Efficiency

- Hundreds of Low-Power Cores
  - MIMD workloads
  - Massive thread-level parallelism
  - Low-power consumption
- Distributed Memory Architecture
  - Grants performance scalability
  - Delivers predictability
- On-Chip Heterogeneity
  - Adaptability to computing demands
  - Enables high-energy efficiency
- Rich On-Chip Interconnects
  - Enable QoS
  - Expose asynchronous communications

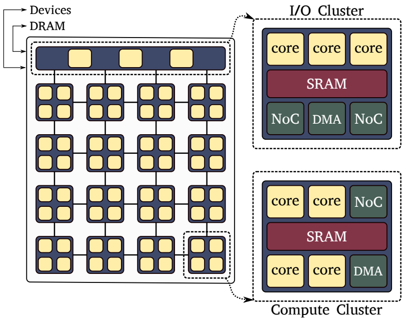


Figure: A 67-core lightweight manycore.

It Is A Distributed Architecture in a Single Chip!

Introduction

Lightweight (LW) Manycores: Programmability Challenges

- High Density Circuit Integration
  - Heat dissipation
  - Dark silicon
- Distributed Memory Architecture
  - Data tiling (small local memories)
  - Message passing communication
- On-Chip Heterogeneity
  - Thread scheduling
  - Data placement
- Rich On-Chip Interconnects
  - Network congestion
  - Security checking

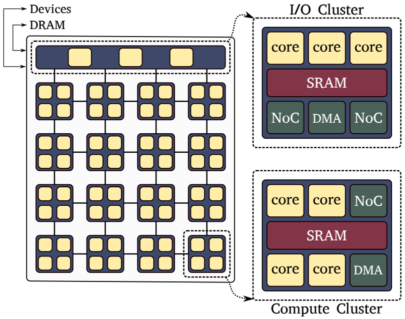


Figure: A 67-core lightweight manycore.

Performance vs Programmability vs Portability

Introduction

Centralized vs Distributed Operating Systems

- Centralized Operating Systems
  - One Full-weight OS is deployed in each cluster
  - User library exposes message-passing API
  - Distributed applications run on the OS

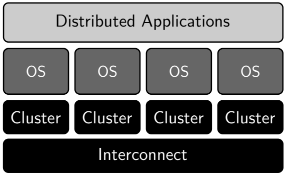


Figure: Centralized OS architecture.

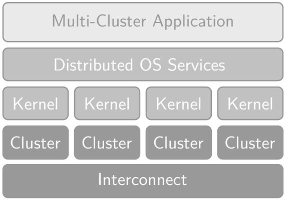


Figure: Distributed OS architecture.

Introduction

Centralized vs Distributed Operating Systems

- Centralized Operating Systems
  - One Full-weight OS is deployed in each cluster
  - User library exposes message-passing API
  - Distributed applications run on the OS
- Distributed Operating Systems
  - One kernel runs in each cluster
  - Subsystems are distributed across the kernels
  - Multi-cluster applications run on the OS

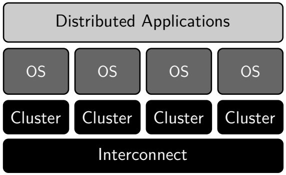


Figure: Centralized OS architecture.

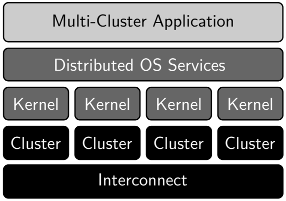


Figure: Distributed OS architecture.

Introduction
Centralized vs Distributed Operating Systems

- Centralized Operating Systems
  - One Full-weight OS is deployed in each cluster
  - User library exposes message-passing API
  - Distributed applications run on the OS
- Distributed Operating Systems
  - One kernel runs in each cluster
  - Subsystems are distributed across the kernels
  - Multi-cluster applications run on the OS
- What is the difference?
  - Scalability
  - Adaptability
  - Composability
  - Transparency
  - Fault tolerance

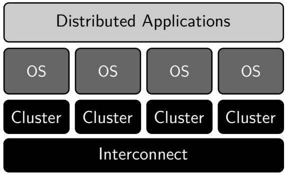


Figure: Centralized OS architecture.

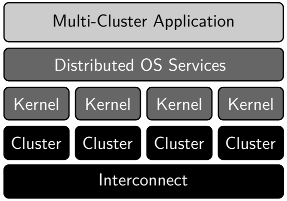


Figure: Distributed OS architecture.

Presentation Outline

1 Introduction
- LW Manycores
- Distributed OSes

2 Nanvix
- Project Overview
- System Architecture
- Multikernel
- Microkernel
- HAL

3 Perspectives
- Roadmap
- Research Ideas

Presentation Outline

1 Introduction
- LW Manycores
- Distributed OSes

2 Nanvix
- Project Overview
- System Architecture
- Multikernel
- Microkernel
- HAL

3 Perspectives
- Roadmap
- Research Ideas

The Nanvix Operating System
Project Overview

- Re-Engineered Version of Nanvix to LW Manycores
  - Home-grown research OS
  - 4 Professors (Brazil and France)
  - 1 PhD, 2 MSc and 4 BSc Students
  - +10 past contributors
  - UGA, PUC Minas, UFSC and Grenoble INP
- Project Guidelines
  - Be Open: invite others to collaborate
  - Be Permissive: enable free adaptability
- Design Principles
  - Be Portable: run on multiple architectures
  - Be Scalable: embrace distributed configuration
  - Be Flexible: expose multiple APIs

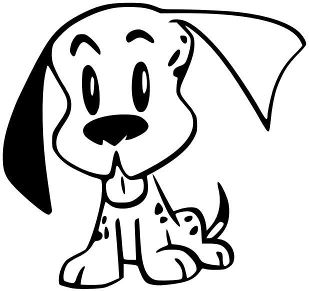


Figure: Bingo, our mascot.

https://github.com/nanvix

The Nanvix Operating System
Internals – System Architecture

- **Architectural Model**
  - Cores are grouped into clusters
  - Each cluster has its own physical address space
  - Shared memory is used for intra-cluster communication
  - Inter-cluster communication is achieved through the NoC
- **Multikernel with Three-Layers**
  - Kernels 🟦
    - Run (self-consciously) on each cluster
    - Provide minimum abstractions
    - Ensure policies and security
  - System Servers 🟥
    - Run on top of kernels at user-level
    - Provide traditional abstractions
    - Collaboratively implement subsystems
  - Runtime Libraries
    - Run alongside with user-applications
    - Interface with system servers
    - Expose standard APIs (i.e., POSIX)

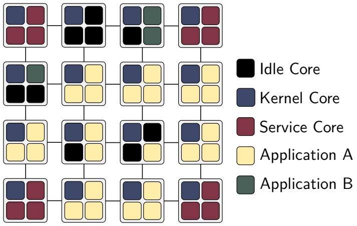


Figure: The multikernel OS structure.

The Nanvix Operating System

Internals – Multikernel

- Processor Management Fleet
  - Spawn Server
  - Name Server
- File System Management Fleet
  - VFS Server
  - Disk Driver
- Memory Management Fleet
  - Remote Memory (RMem) Server
  - Shared Memory (SHM) Server

https://github.com/nanvix/multikernel

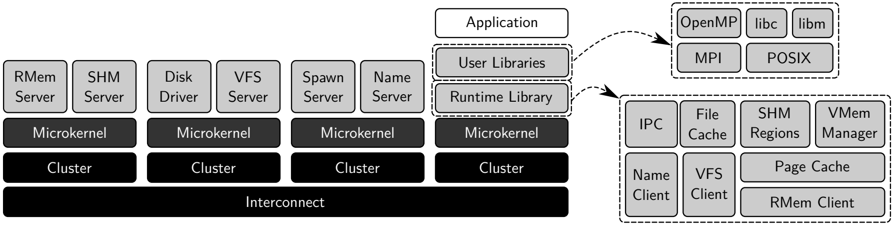


Figure: An overview of the Nanvix multikernel.

The Nanvix Operating System

Internals – Microkernel

- Microkernel Design
  - Asymmetric: runs on a dedicated core of a cluster
  - Small: provides only essential abstractions (about 5k LoC)
  - Portable: supports Bostan, RISC-V, OpenRISC and x86 based manycores
- IKC Facility
  - Inter-cluster synchronization
  - Inter-cluster communication
- Thread Management System
  - Non-interruptible kernel threads
  - Sleep/wakeup primitives
  - Exception handling forwarding
- Memory Management System
  - Single address space
  - Two-level paging scheme

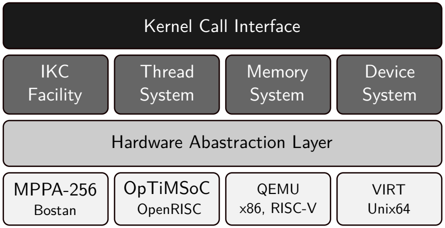


Figure: An overview of the Nanvix kernel.

https://github.com/nanvix/microkernel

The Nanvix Operating System
Internals – HAL

- Provide a standard interface across multiple platforms
- Enable portability of Nanvix
- Core Abstraction Layer
  - Core setup and shutdown
  - Interrupt and exception hooking
  - Trap forwarding
  - Memory cache manipulation
  - MMU and TLB management
- Cluster Abstraction Layer
  - Remote core start and stop
  - Events (inter-core interrupts)
  - Virtual memory
  - DMA driver
- Processor Abstraction Layer
  - Uniform cluster numbering
  - Inter-cluster communication

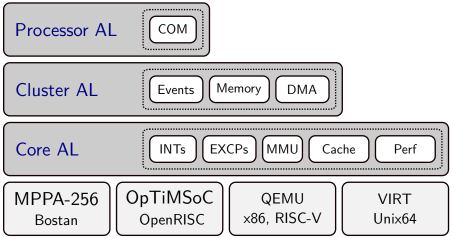


Figure: An overview of the Nanvix HAL.

https://github.com/nanvix/hal

Presentation Outline

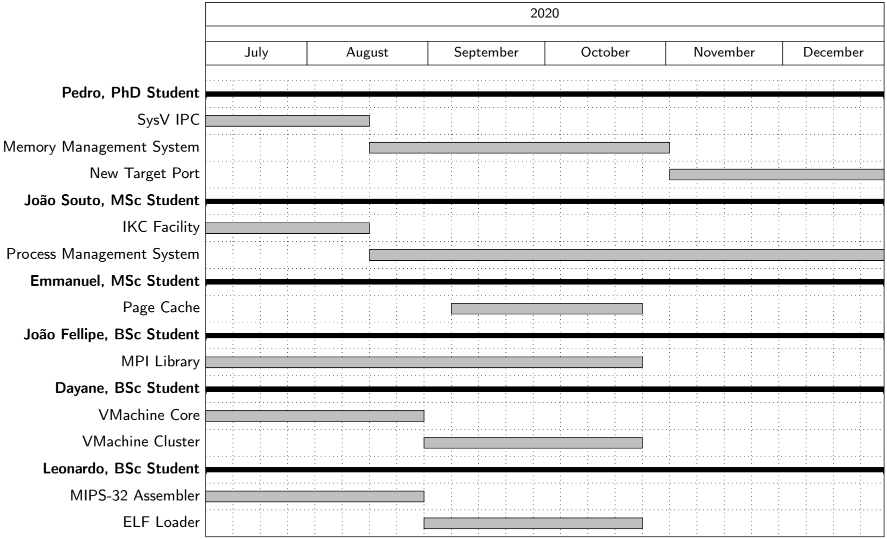
1 Introduction
- LW Manycores
- Distributed OSes

2 Nanvix
- Project Overview
- System Architecture
- Multikernel
- Microkernel
- HAL

3 Perspectives
- Roadmap
- Research Ideas

Perspectives

Roadmap



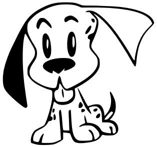
Perspectives
Research Ideas

- User-Level Thread Library
  - Enable support for core multiplexing
- Lightweight Containers
  - Enable dynamic process scheduling & migration
- x86 Multicore Support
  - Introduce multicore support to our vintage target
- Nanvix + Ariane
  - Deploy RISC-V Port of Nanvix on a baremetal platform
- Nanvix + Bluedragon
  - Deploy Nanvix in multi-cluster OpTiMSoC (FPGA) configuration

Thank You!

Nanvix
A Distributed Operating System for Lightweight Manycores

Pedro Henrique Penna
`pedro.penna@sga.pucminas.br`



References I

João Vicente Souto et al. "Mecanismos de Comunicação entre Clusters para Lightweight Manycores no Nanvix OS". In: Escola Regional de Alto Desempenho da Região Sul. ERAD/RS '20. Santa Maria, Brazil: SBC, 2020, pp. 1–4. URL: https://sol.sbc.org.br/index.php/eradrs/article/view/10741.

João Fellipe Uller et al. "Proposta de Suporte ao Padrão MPI sobre Infraestrutura de Comunicação de Baixo Nível no Nanvix". In: Escola Regional de Alto Desempenho da Região Sul. ERAD/RS '20. Santa Maria, Brazil: SBC, 2020, pp. 121–124. URL: https://ojs.sbc.org.br/index.php/eradrs/article/view/10771.

Pedro Henrique Penna, Davidson Francis, and João Souto. "The Hardware Abstraction Layer of Nanvix for the Kalray MPPA-256 Lightweight Manycore Processor". In: Conférence d'Informatique en Parallélisme, Architecture et Système. Anglet, France, 2019, pp. 1–11.

References II

Pedro Henrique Penna et al. "On the Performance and Isolation of Asymmetric Microkernel Design for Lightweight Manycores". In: Brazilian Symposium on Computing Systems Engineering (SBESC). SBESC '19. Natal, Brazil: IEEE, 2019, pp. 1–8. ISBN: 978-1-7281-6318-5. DOI: 10.1109/SBESC49506.2019.9046080. URL: https://ieeexplore.ieee.org/document/9046080.

Pedro Henrique Penna et al. "RMem: An OS Service for Transparent Remote Memory Access in Lightweight Manycores". In: International Workshop on Programmability and Architectures for Heterogeneous Mu
Valencia, Spain, 2019, pp. 1–16.

Davidson Francis Lima, Pedro Henrique Penna, and Henrique Freitas. "Uma Análise do Overhead Introduzido pelo Sistema Operacional Nanvix na Execução de Cargas de Trabalho". In: Workshop de Iniciação Científica do WSCAD. WSCAD-WIC '17. Campinas, Brazil: SBC, 2017, pp. 141–146.

Pedro H. Penna et al. "Assessing the Performance of the SRR Loop Scheduler". In: International Conference on Computational Science (ICCS). Zurich, Switzerland, 2017.

References III

- Pedro Henrique Penna et al. "Using the Nanvix Operating System in Undergraduate Operating System Courses". In: Brazilian Symposium on Computing Systems Engineering. SBESC '17. IEEE, 2017, pp. 193–198. ISBN: 978-1-5386-3590-2. DOI: 10.1109/SBESC.2017.33. URL: http://ieeexplore.ieee.org/document/8116579/.

- Matheus Souza et al. "CAP Bench: A Benchmark Suite for Performance and Energy Evaluation of Low-Power Many-Core Processors". In: Concurrency and Computation: Practice and Experience (CCPE) 29.4 (2017), pp. 1–18. ISSN: 1532-0626. DOI: 10.1002/cpe.3892.

- Matheus A. Souza et al. "CAP Bench: a benchmark suite for performance and energy evaluation of low-power many-core processors". In: Concurrency and Computation: Practice and Experience 29.4 (2017), e3892. ISSN: 15320626. DOI: 10.1002/cpe.3892.

- Pedro H. Penna and Márcio Castro. "Desenvolvimento de Aplicações Paralelas Eficientes com OpenMP". In: Tendências em Arquiteturas, Aplicações e Programação Paralela. 1st ed. Sergipe: Instituto Federal de Sergipe, 2016. Chap. 1, pp. 10–14.

References IV

Pedro H. Penna, Márcio Castro, and Henrique Freitas. "SRR: Um Escalonador de Laços Sensível a Carga de Trabalho". In: Escola Regional de Alto Desempenho: Fórum de Pós-Graduação (ERAD-RS). Ed. by SBC. São Leopoldo, Brazil, 2016.

Pedro H. Penna et al. "Design Methodology for Workload-Aware Loop Scheduling Strategies Based on Genetic Algorithm and Simulation". In: Concurrency and Computation: Practice and Experience (CCPE) (2016). ISSN: 1532-0626. DOI: 10.1002/cpe.3933.

Emmanuel Podestá Júnior et al. "PSkel-MPPA: Uma Adaptação do Framework PSkel para o Processador Manycore MPPA-256". In: Escola Regional de Alto Desempenho: Fórum de Iniciação Científica (ERAD-RS). São Leopoldo, Brazil: SBC, 2016.

João Saffran et al. "A Low-cost Energy-efficient Raspberry Pi Cluster for Data Mining Algorithms". In: International European Conference on Parallel and Distributed Computing Workhops
Grenoble, France, 2016.

References V

Emilio Francesquini et al. "On the Energy Efficiency and Performance of Irregular Application Executions on Multicore, NUMA and Manycore Platforms". In: Journal of Parallel and Distributed Computing. JPDC 76.C (2015), pp. 32–48. ISSN: 0743-7315. DOI: 10.1016/j.jpdc.2014.11.002. URL: https://doi.org/10.1016/j.jpdc.2014.11.002.

Pedro H. Penna et al. "Uma Metodologia Baseada em Simulação e Algoritmo Genético para Exploração de Estratégias de Escalonamento de Laços". In: Simpósio em Sistemas Computacionais de Alto Desempenho (WSCAD). Florianópolis, Brazil: SBC, 2015, pp. 156–167.

Pedro Henrique Penna et al. "Uma Metodologia Baseada em Simulação e Algoritmo Genético para Exploração de Estratégias de Escalonamento de Laços". In: Simpósio Brasileiro em Sistemas Computacionais de Alto Desempenho (WSCAD). Sociedade Brasileira de Computação, 2015, pp. 156–167.

References VI

Líntia Avelar, Pedro H. Penna, and Henrique Freitas. “Algoritmo K-means para Mapeamento Estático de Processos em Redes-em-Chip”. In: Simpósio em Sistemas Computacionais de Alto Desempenho (WSCAD). São José dos Campos, Brazil: SBC, 2014, pp. 204–215.

Pedro H. Penna and Henrique Freitas. “Investigando a Influência da Organização de Caches L2 no Desempenho de Processadores Multicore Superescalares”. In: Simpósio em Sistemas Computacionais de Alto Desempenho: Workshop de Iniciação São José dos Campos, Brazil: SBC, 2014, pp. 236–241.

Pedro H. Penna, Henrique Freitas, and Ricardo Ferreira. “Um Processo Automatizado para Modelagem e Prototipação de Redes Reguladoras em FPGA”. In: Simpósio em Sistemas Computacionais de Alto Desempenho: Workshop de Iniciação Vitória, Brazil: SBC, 2011, pp. 1–4.