Virtual-CPU Scheduling in the Quest Operating System

Matthew Danish, Ye Li and Richard West

Boston University

April 13, 2011

Focus of this Work

- Predictability and temporal isolation
  - Guaranteed resource allocation over specific windows of time
- Integrated management of tasks and I/O
  - Programs sleeping and waking
  - Periodic tasks
  - Interrupts from I/O devices
- Supporting temporal isolation using Virtual CPUs
  - Consolidate threads on Virtual CPUs
  - Divide up physical resources
  - Statically scheduled

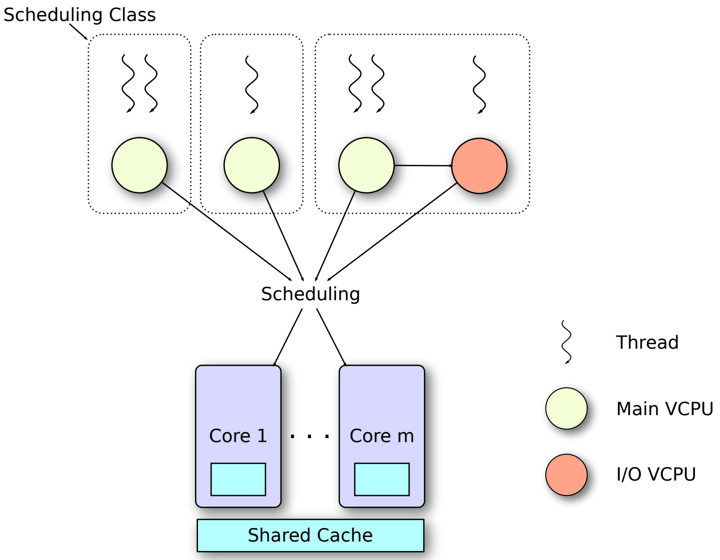
Background

- Building a new OS to focus on safety, predictability, and efficiency.
  - Memory safety, program correctness
  - Temporal isolation
  - Optimized use of hardware resources
- Why not Linux?
  - Start from a clean slate
  - Design freedom
  - Linux was never meant to be real-time
  - Linux is a constantly moving target

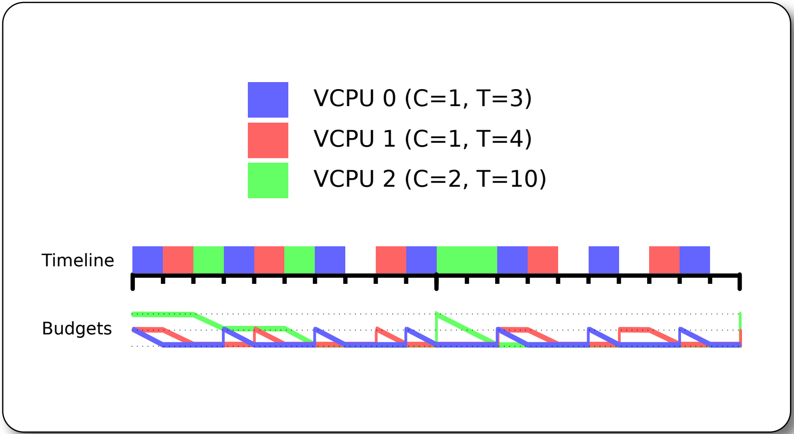
Quest

- Small x86 SMP research operating system
- Under 11,000 lines of core kernel code
- Support for ACPI, PCI bus, USB, ATA drives, TCP/IP (lwIP)

VCPU Scheduling Subsystem

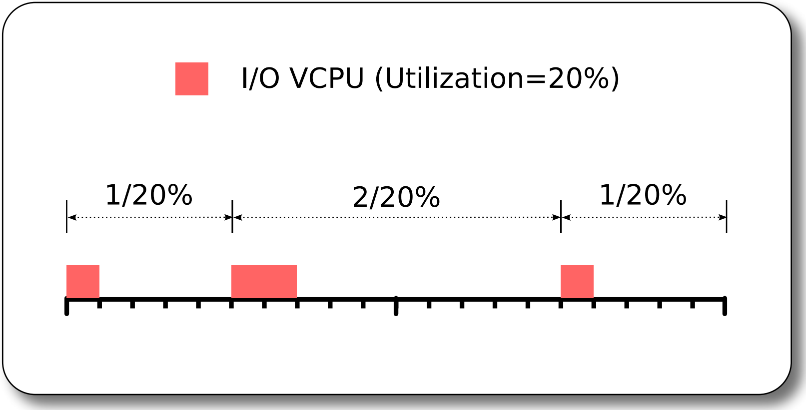


Main VCPUs



- Fixed Priority, Rate-Monotonic Sporadic Servers
- Budgets replenish on a timer
- Replenishments can be split, or merged

I/O VCPUs



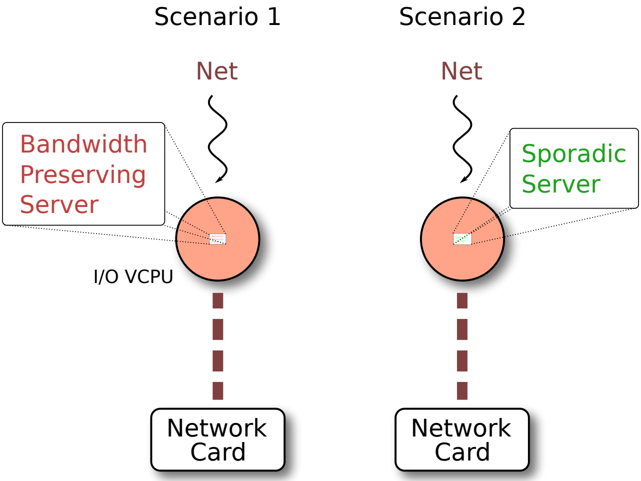
- For tasks that react to hardware
- Inherits priority from Main VCPU
- Bandwidth-preserving Server
- I/O VCPU arrangement is part of policy

Evaluation

- Using single core of an Intel Core2 Extreme QX6700 2.6GHz
- 4GB DDR SDRAM available
- Intel 8254x-series "e1000" network adapter
- UHCI-based USB connected to Mass Storage device
- Parallel ATA CD-ROM drive

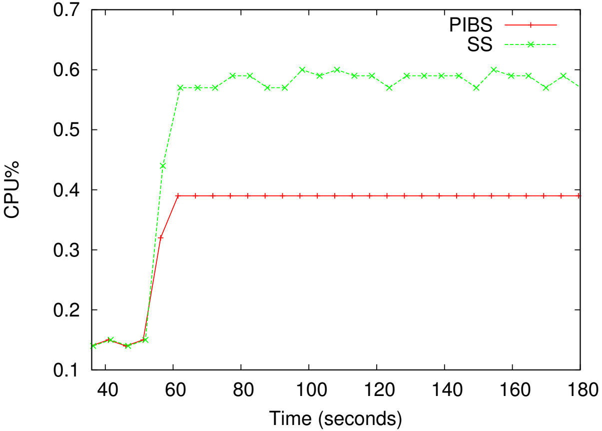
Sporadic vs Bandwidth-Preserving Servers

- Experiment: Packet handling scheduling overhead
- Ping-flood at $t = 50$



Sporadic vs Bandwidth-Preserving Servers

- Experiment: Packet handling scheduling overhead
- Ping-flood at $t = 50$
- Sporadic server overhead much higher



I/O VCPUs Inherit Priority

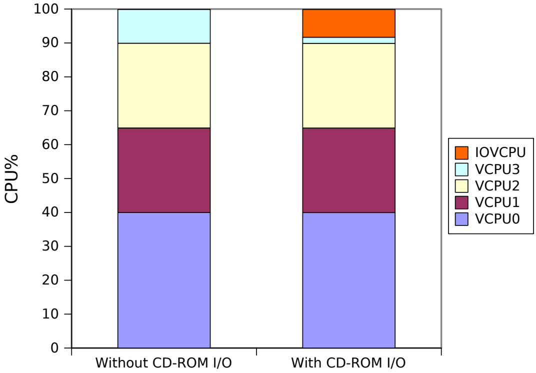
- Experiment: Reading from CD-ROM on behalf of VCPU 1

Name	Capacity	Period
VCPU2	1	4
VCPU0	2	5
VCPU1	2	8
VCPU3	1	10
IOVCPU	10%	

I/O VCPUs Inherit Priority

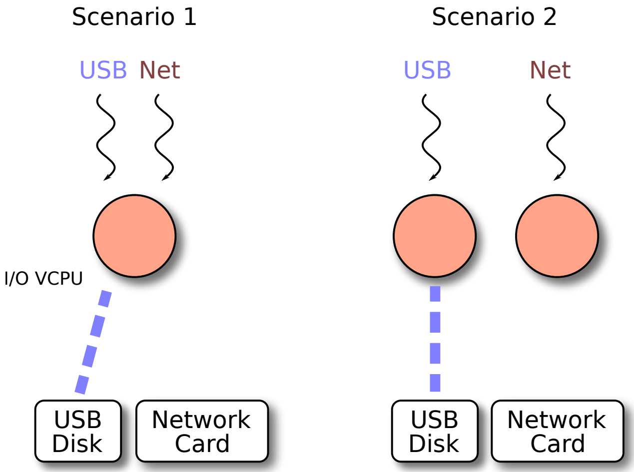
- Experiment: Reading from CD-ROM on behalf of VCPU 1
- Only lower-priority VCPU 3 loses CPU time

Name	Capacity	Period
VCPU2	1	4
VCPU0	2	5
VCPU1	2	8
VCPU3	1	10
IOVCPU	10%	



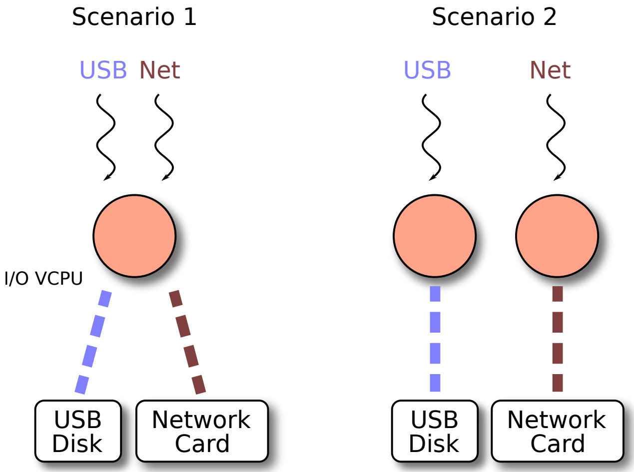
Shared vs Separate I/O VCPUs

- Two I/O tasks: USB storage reading, network packet handling
- Experiment: single shared I/O VCPU vs two separate I/O VCPUs



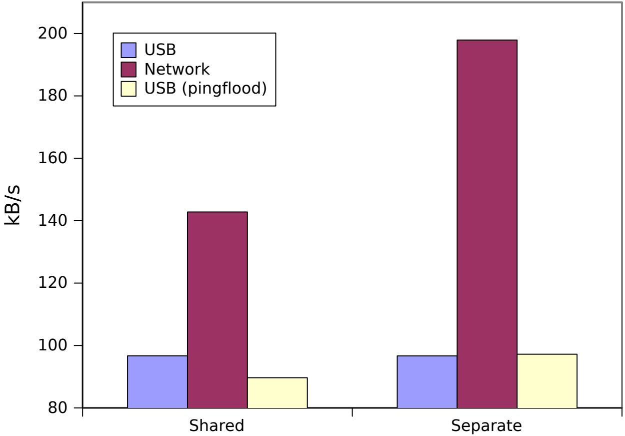
Shared vs Separate I/O VCPUs

- Two I/O tasks: USB storage reading, network packet handling
- Experiment: single shared I/O VCPU vs two separate I/O VCPUs
- Compare USB bandwidth during a network ping-flood



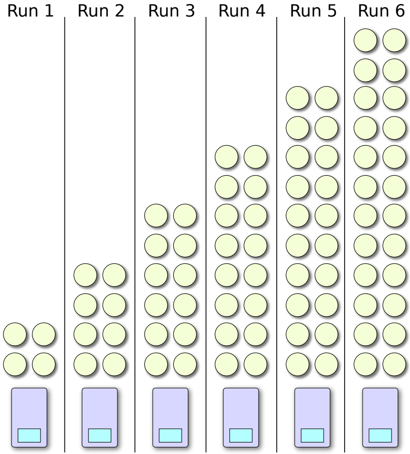
Shared vs Separate I/O VCPUs

- Two I/O tasks: USB storage reading, network packet handling
- Experiment: single shared I/O VCPU vs two separate I/O VCPUs
- Compare USB bandwidth during a network ping-flood



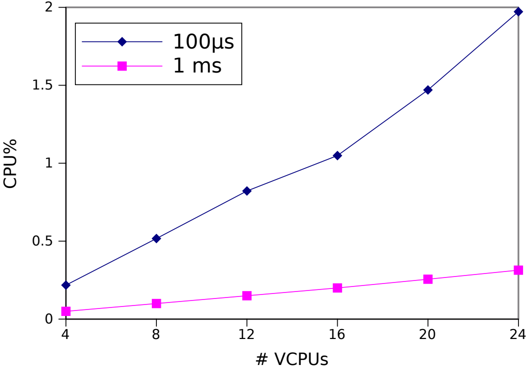
Scheduler Overhead

- Experiment: scaling as the number of Main VCPUs increases



Scheduler Overhead

- Experiment: scaling as the number of Main VCPUs increases
- Scheduler overhead increases linearly



Conclusions

- Virtual CPUs for predictability and temporal isolation
- Main VCPUs partition physical CPU resources
- I/O VCPUs regulate servicing of hardware events
- Two-tiered scheduling hierarchy simplifies design
- Basis for performance isolation
  - Real-time and embedded systems
  - Virtual machine scheduling
  - Partitioning of distributed cloud computing resources

Future Directions

- Real-time VCPU scheduling on multiple processors and cores
  - Minimize cache contention and communication costs
  - Maximize instructions per cycle
- Better safety with hardware sandboxing or software techniques
  - Virtualization
  - Programming language support
- Componentization with predictable communication
- Static verification of useful properties

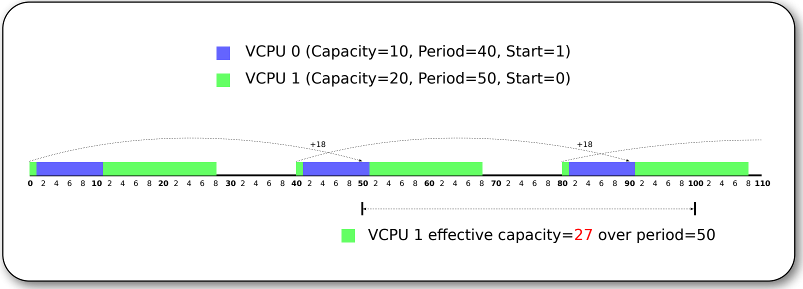
Development

- Acknowledgements
  - Richard West and Gary Wong (original developers)
  - Matthew Danish and Ye Li (SMP and VCPU implementation)
- Source: `http://QuestOS.github.com/`
- `http://www.cs.bu.edu/fac/richwest/quest.html`

Thank You

Extra Slides

Premature Replenishment



- Over-capacity exploit without proper splitting of replenishments
- *Defects of the POSIX Sporadic Server and How to Correct Them*

Fin