ISQED
Wednesday March 4, 2015
Session 5B

A hypervisor approach with real-time support to the MIPS M5150 processor

Authors: Samir Zampiva (samir.zampiva@acad.pucrs.br)
Carlos Moratelli (carlos.moratelli@pucrs.br)
Fabiano Hessel (fabiano.hessel@pucrs.br) (**speaker**)

Schedule

- Introduction
- Virtualization Model
- Model Implementation
- Results and Discussion
- Conclusion and Future Work

Introduction

- **Widespread for enterprise and server applications, virtualization has been applied for ES in the recent years.**

- **The common motivation for Embedded Virtualization and General Purpose Virtualization usage include:**
  - Decrease costs**;**
  - Improve the security;
  - Stronger spatial isolation.

- **Embedded Virtualization brings new challenges:**
  - Real-time constraints (stronger temporal isolation);
  - Smaller memory footprint.

- **Embedded architectures like MIPS and ARM purposed hardware virtualization extensions.**
  - Hardware-assisted virtualization can improve performance and footprint of hypervisor implementations.

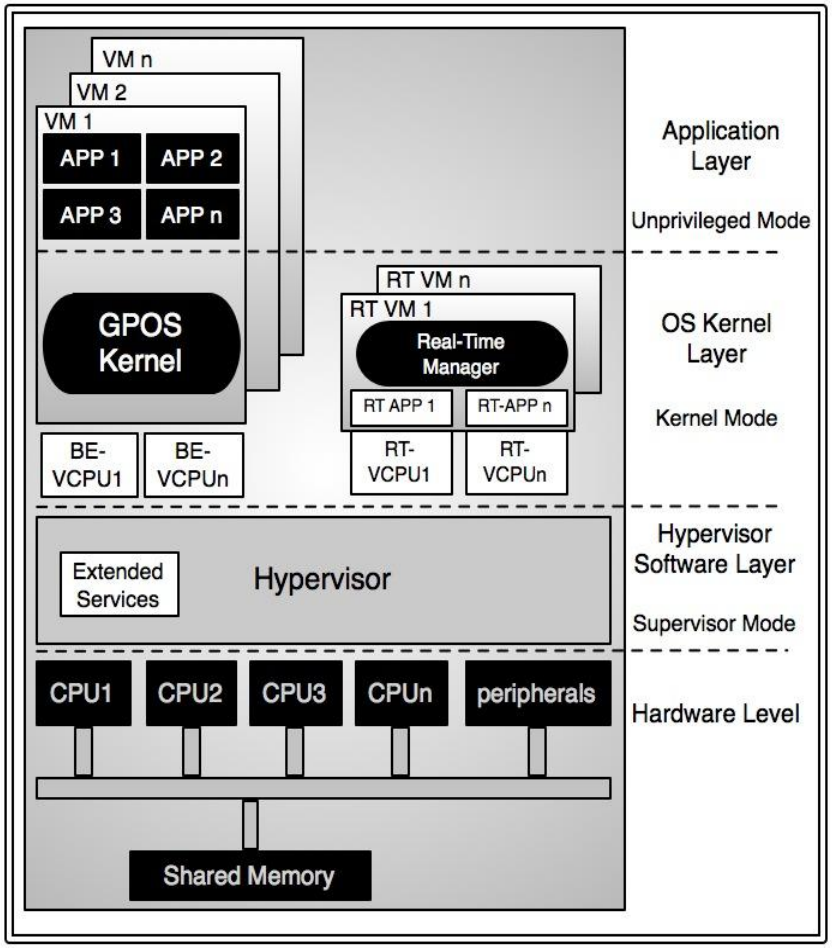
Introduction

- **This work presents a hypervisor for embedded systems.**
  - Hellfire Hypervisor.
- **Designed to the MIPS M5150 embedded processor.**
  - Hardware-assisted virtualization (MIPS VZ Module).
- **Lightweight virtualization layer.**
- **Real-time support.**
- **Extended services.**
  - System services;
  - Real-time management;
  - Inter-VM communication.
- **Supporting HellFire OS (RTOS).**
  - Linux support is an ongoing work.

Schedule

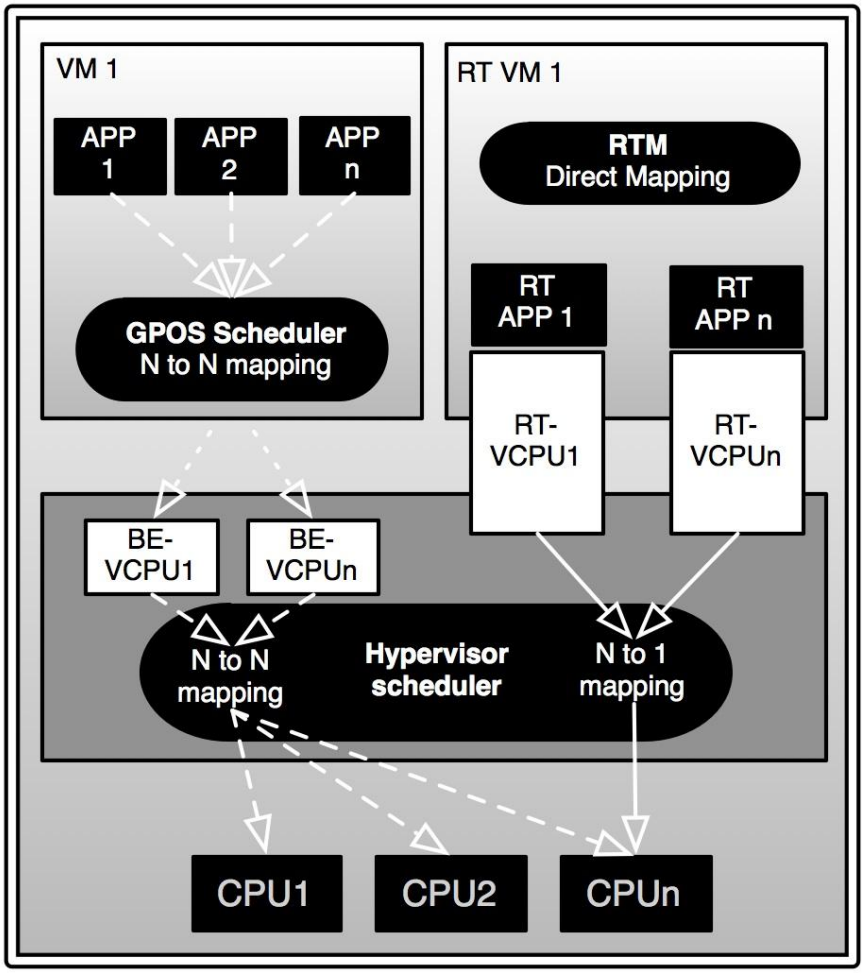
Virtualization Model

Virtualization Model



Virtualization model and Application Domain Unit for multiprocessor embedded systems.

Virtualization Model



Flexible Mapping model for multiprocessor embedded systems with real-time support.

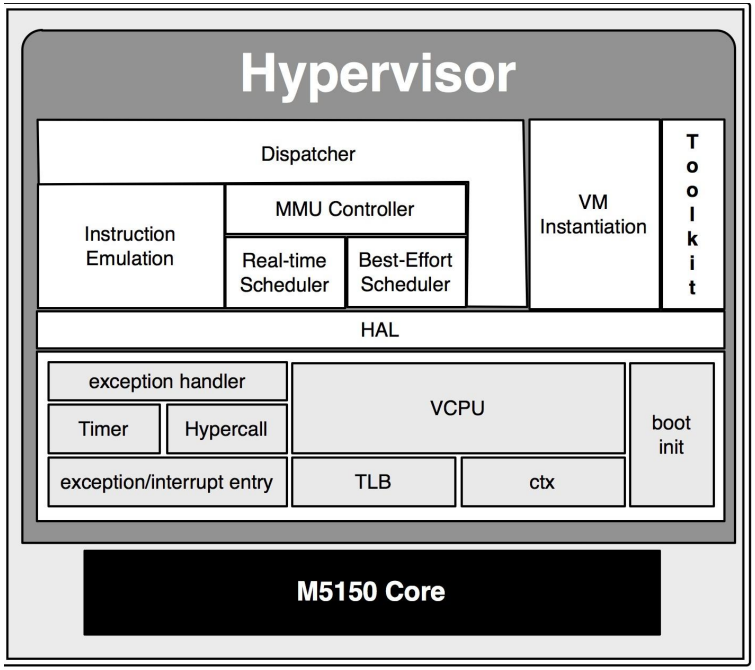
Schedule

Model Implementation

Model Implementation

- **Hardware Aspects.**
  - MIPS M5150 (Released by the end of 2013).
  - MIPS Virtualization Module Support (VZ);

- **Software aspects.**
  - Small footprint.
    - 6072 lines of code in C language and Assembly;
    - Supporting Hellfire OS (RTOS);
    - Support for Linux is an ongoing work.
  - Support up to 8 guests instances.
  - Full-virtualization of the processor;
    - Para-virtualization for extended services;
  - Strong temporal isolation.
    - EDF algorithm for real-time VCPUS.



Block diagram of the hypervisor.

Model Implementation

- **Extended Services**
  - **Extended services are hypervisor calls that provides services to the guest OSs.**
    - Implemented using hypercall concept.
  - **The extended services are divided into three groups:**
    - VM identification;
    - RT-VCPU management, and;
    - Communication services.
  - **Guest OSs that not desire to use the extended services is not required to implement the hypercalls.**
    - Our hypervisor implements full-virtualization of the processor.

Model Implementation

Hypercalls as hypervisor's extended services.

Extended Service	Hypercall	Description
VM identification	HCALL_INFO_GET_ID	Return the VM ID.
RT-VCPU Manag.	HCALL_RT_CREATE_APP	RT app. Instantiation.
	HCALL_RT_LAUNCH_APP	RT app. launch.
	HCALL_RT_DELETE_APP	RT app. delete.
Commun. Services	HCALL_IPC_SEND_MSG	Send messages.
	HCALL_IPC_RECV_MSG	Receive messages.

Results and Discussion

Results and Discussion

- **We used the Instruction Accurate Simulator (IASim) to evaluate our implementation.**

- **We conducted experiments to determine the context switching overhead among VMs and the overhead penalty for communication between VMs.**

- **Context switching overhead:**
  - **Experiment 1**: Aim to determine the guest OS overhead in number of instructions when comparing native versus virtualized execution.
  - **Experiment 2**: Overhead for an increasing number of guest OSs being virtualized.

- **Communication Overhead:**
  - We measured the number of instructions needed to deliver a message from a VM to another.

Context Switching Overhead - Experiment 1

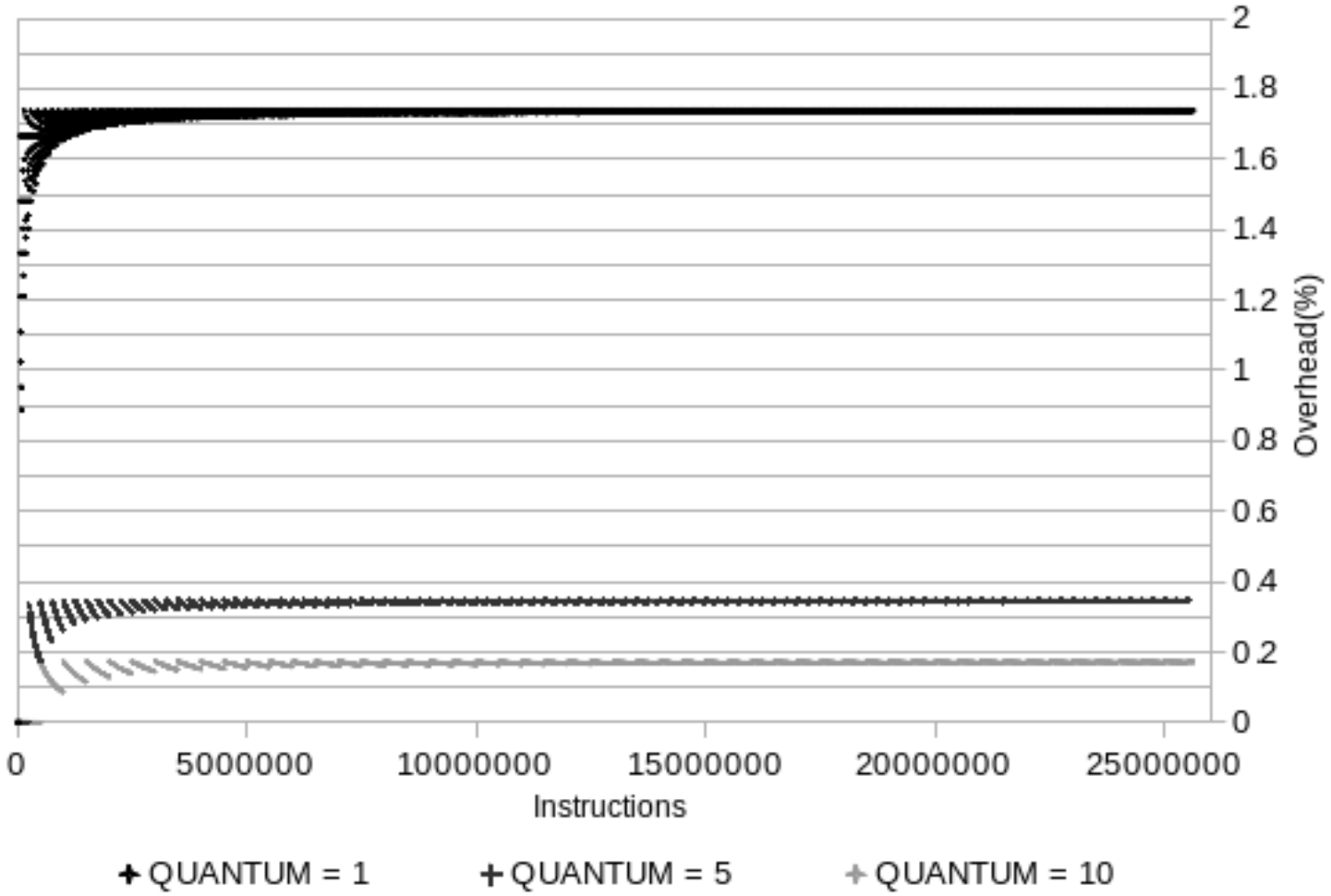
- **This experiment determines the overhead in number of instructions imposed by the hypervisor during context switching between VMs.**

- **The experiment consists of a Guest OS (Hellfire OS) running a port of ADPCM (Adaptive Differential Pulse Code Modulation) algorithm from WCET benchmark.**
  - The benchmark does not use any extended services.
    - Focus on the context switching overhead.

- **The hypervisor quantum scheduler was configured to 1, 5 and 10ms.**

- **The guest OS quantum was kept in 10ms.**

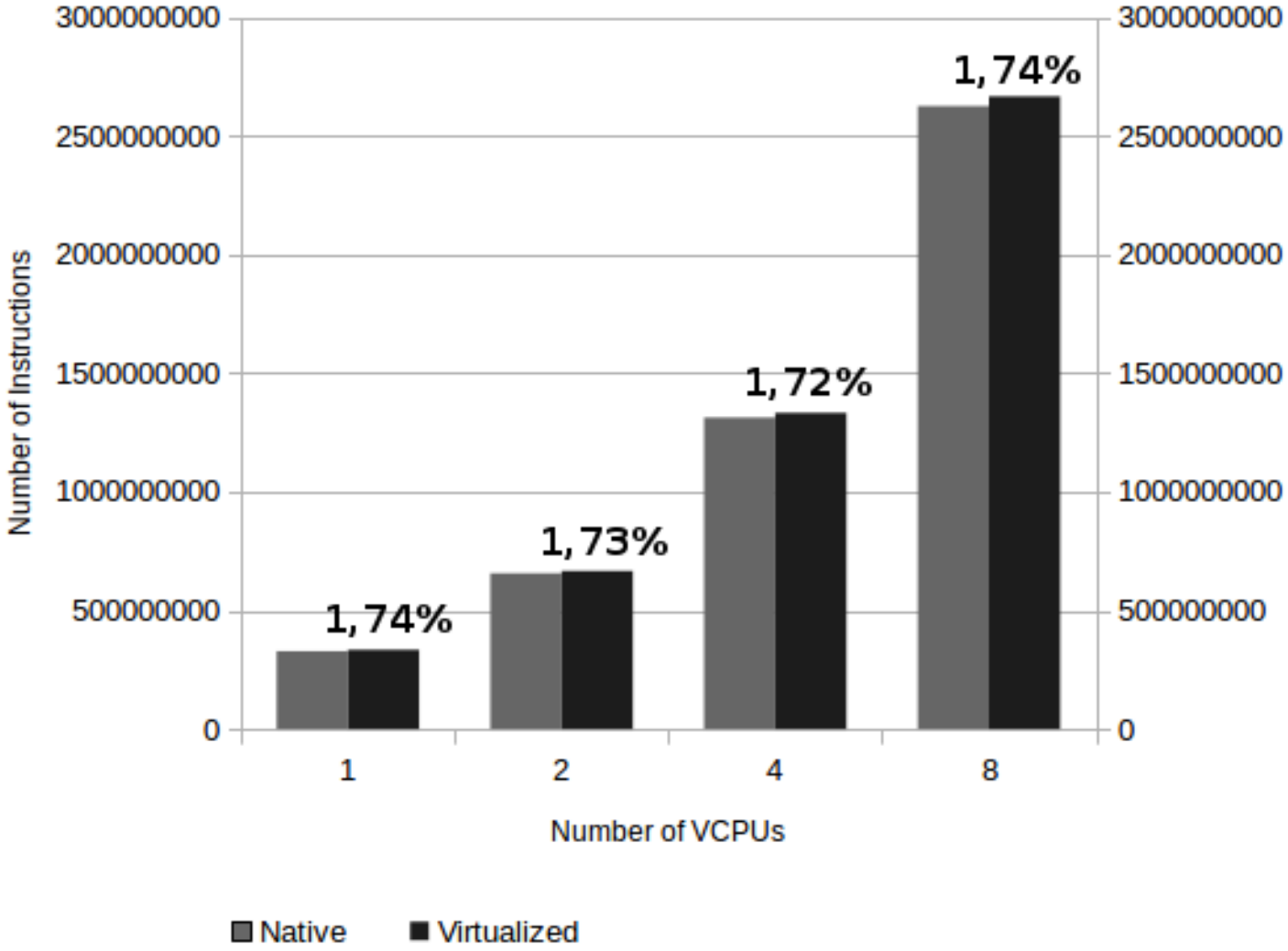
Context Switching Overhead - Experiment 1



Context Switching Overhead - Experiment 2

- This experiment was conducted to determine the imposed overhead for an increasing number of guest OSs.
- Each guest OS executes the same ADPCM application.
- We executed up to 8 guest OS concurrently.
- Hypervisor scheduler quantum was kept in 1 ms.
- The result shows the total number of instructions performed during the tests.

Context Switching Overhead - Experiment 2



Extended Services Overhead - Communication

- **This experiment measures the number of instructions needed to deliver a message from a VM to another.**

- **The scenery setup consists of two communicant instances messages of different sizes: 20, 40, 60, 80 and 100 bytes long.**

- **It was measured the overhead of the hypercalls:**
  - Message send (HCALL_IPC_SEND_MSG);
  - Message receive (HCALL_IPC_RECV_MSG);
  - Message delivery consisting in both calls HCALL_IPC_SEND_MSG and HCALL_IPC_RECV_MSG.

Context Switching Overhead - Experiment 2

Overhead of extended services for communication.

Experiment	Message Size (#bytes) 20	40	60	80	100
HCALL_IPC_SEND_MSG	1420	1485	1550	1615	1680
HCALL_IPC_RECV_MSG	1042	1107	1172	1237	1302
Message delivery	2594	2724	2854	2984	3114

Conclusion and Future Work

Conclusion and Future Work

- **We presented a full-virtualized hypervisor to the MIPS M5150 embedded processor.**
  - The hypervisor implements extended services using the hypercall technique.
    - Resulting in a mixed virtualization system.

- **The simulation results show optimistic performance results.**
  - Low overhead for context switching and communication.

- **At this moment, the hypervisor is already being executed by the SEAD-3 development board.**
  - Performance tests are being conducted in a real platform.

- **Support for Linux is an ongoing work and should finished by the second semester.**

Questions ?