Microkernel virtualization under one roof
- dare the impossible -

Alexander Böttcher
<alexander.boettcher@genode-labs.com>

Outline

1. Introduction
2. Kernel interfaces
3. VM interface harmonization
4. VMMs harmonized
5. Conclusion

Outline

1. Introduction
2. Kernel interfaces
3. VM interface harmonization
4. VMMs harmonized
5. Conclusion

Motivation

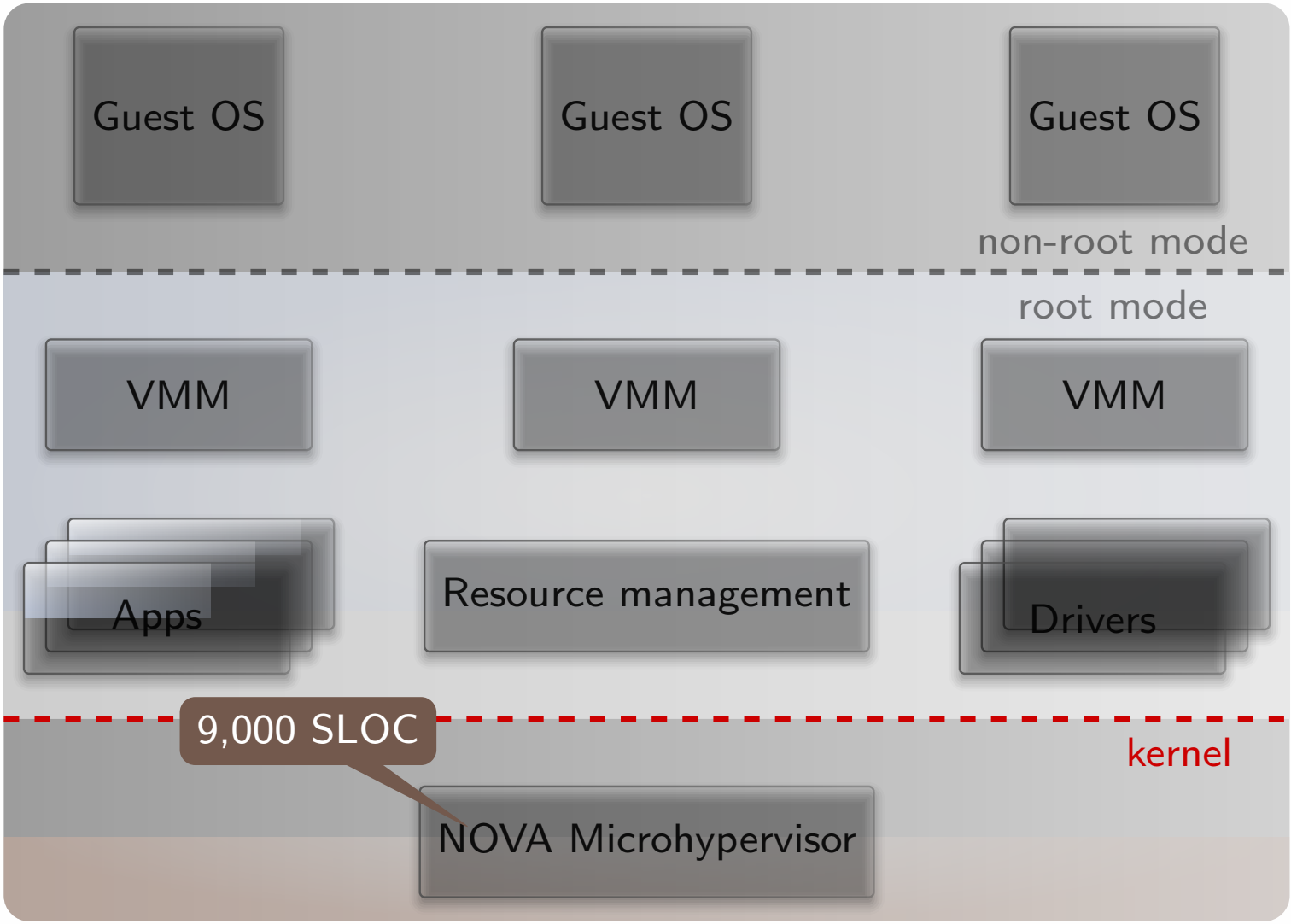
Off-the-shell virtualization solution ridden with complexity.

Application of virtualization call for trustworthy solutions.

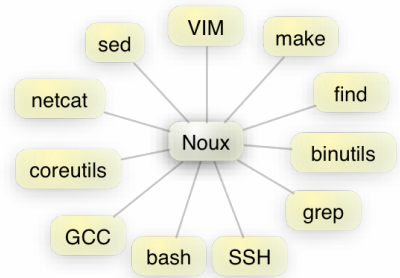
Complexity defeats trust.

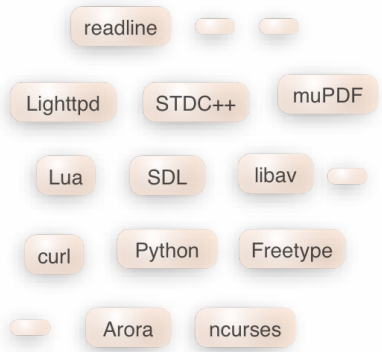
Alternative approach
$\rightarrow$ Microkernels with hardware assisted virtualization extensions

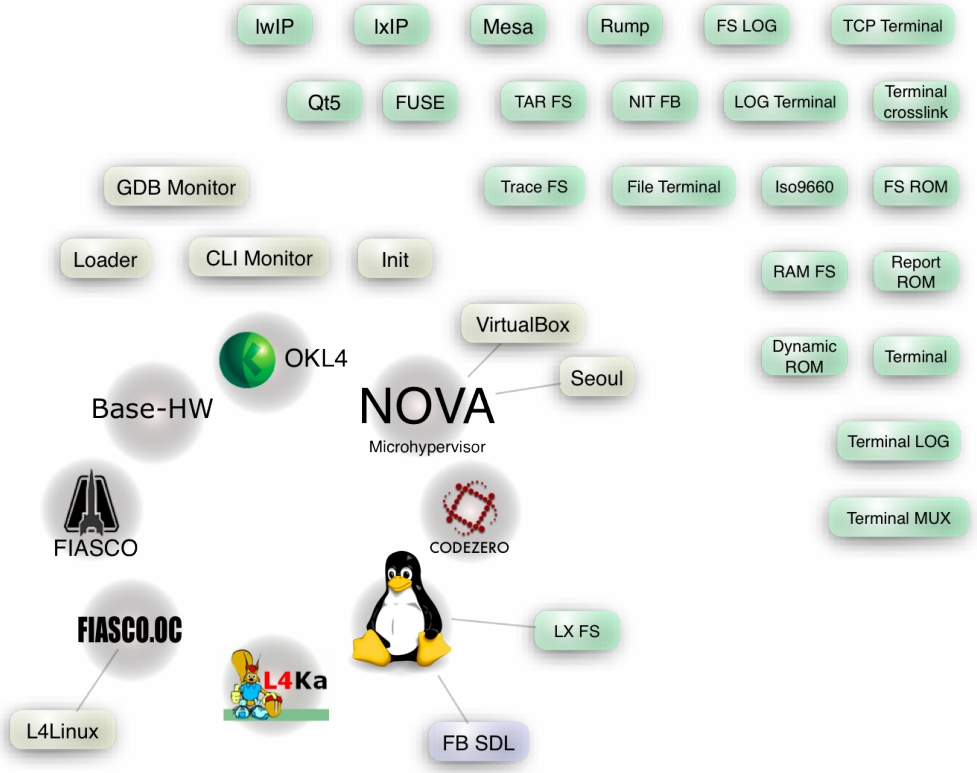
Component based virtualization architecture



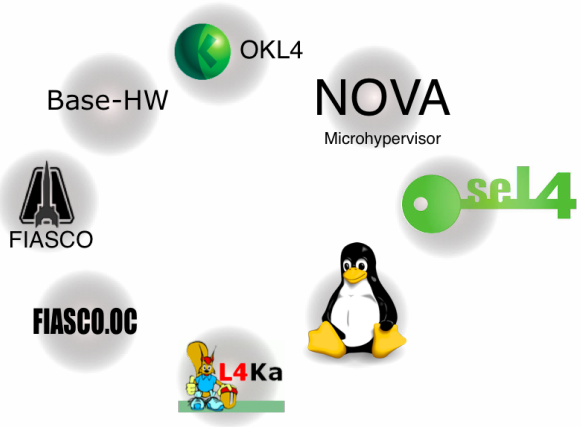
Genode OS framework



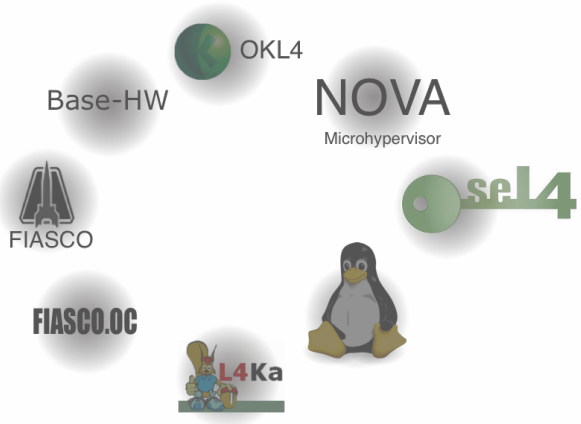




General supported kernels on Genode



Kernels with hardware assisted virtualization



VMM inventory of Genode

Hardware assisted virtualization/separation support

Microkernel	Host	VMM	Guest vCPU
hw	ARM, 32bit	custom	1, 32bit
hw/trustzone	ARM, 32bit	custom	1, 32bit
hw with Muen	Intel, 64bit	VBox 4	1, 32bit
NOVA	Intel & AMD 32bit, 64bit	Seoul	**N**, 32bit
		VBox 4	**N**, 32bit, **64** bit
		VBox 5	**N**, 32bit, **64** bit

Research challenge

Vision: VMMs runnable on all kernels w/o re-compilation

Research challenge

Vision: VMMs runnable on all kernels w/o re-compilation

- Focus on x86 microkernels for now
- $\rightarrow$ NOVA, seL4, Fiasco.OC, and -hw-

Research challenge

Vision: VMMs runnable on all kernels w/o re-compilation

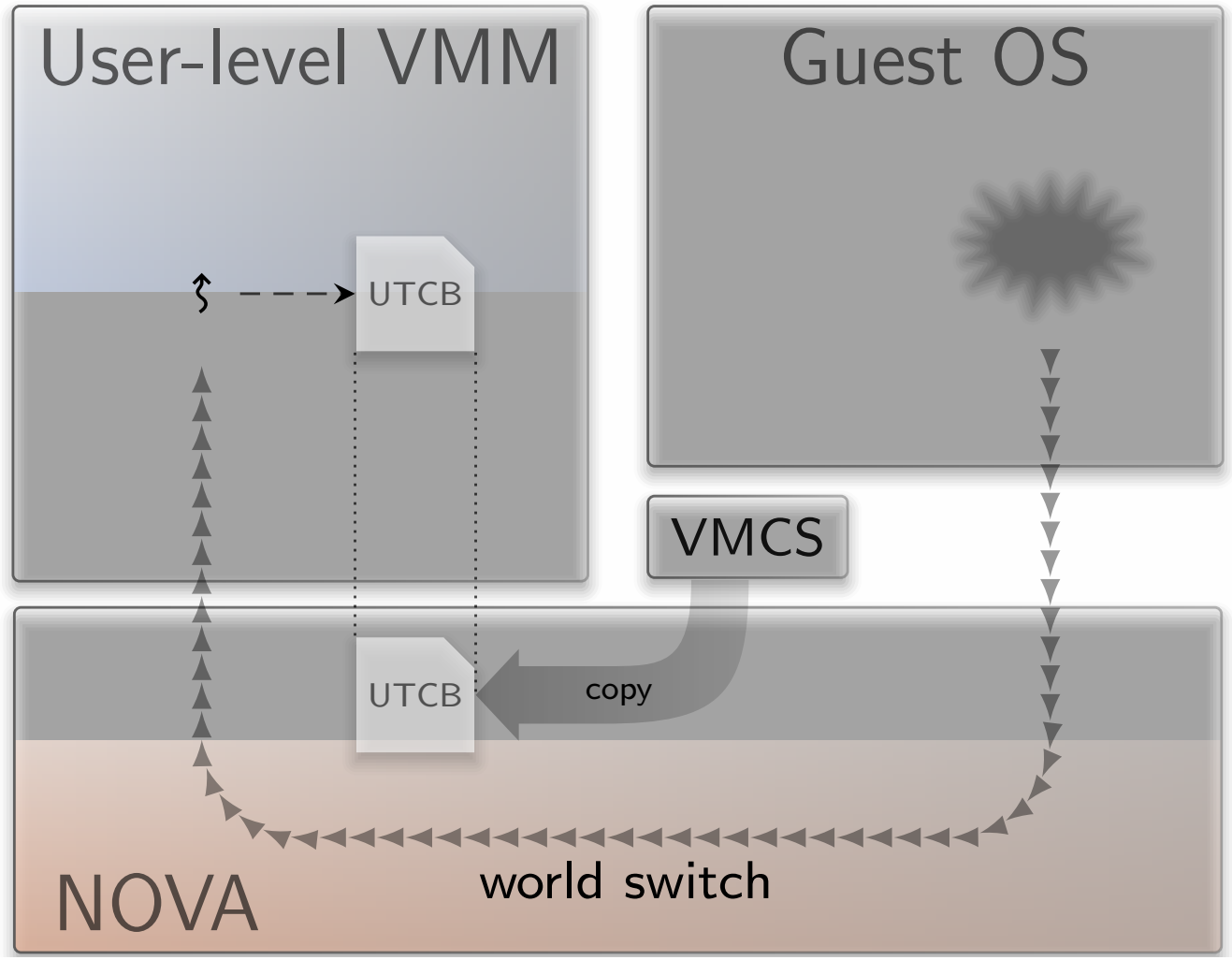
- Focus on x86 microkernels for now
- $\rightarrow$ NOVA, seL4, Fiasco.OC, and -hw-

Approach: Generalize VM interface as used by -hw-

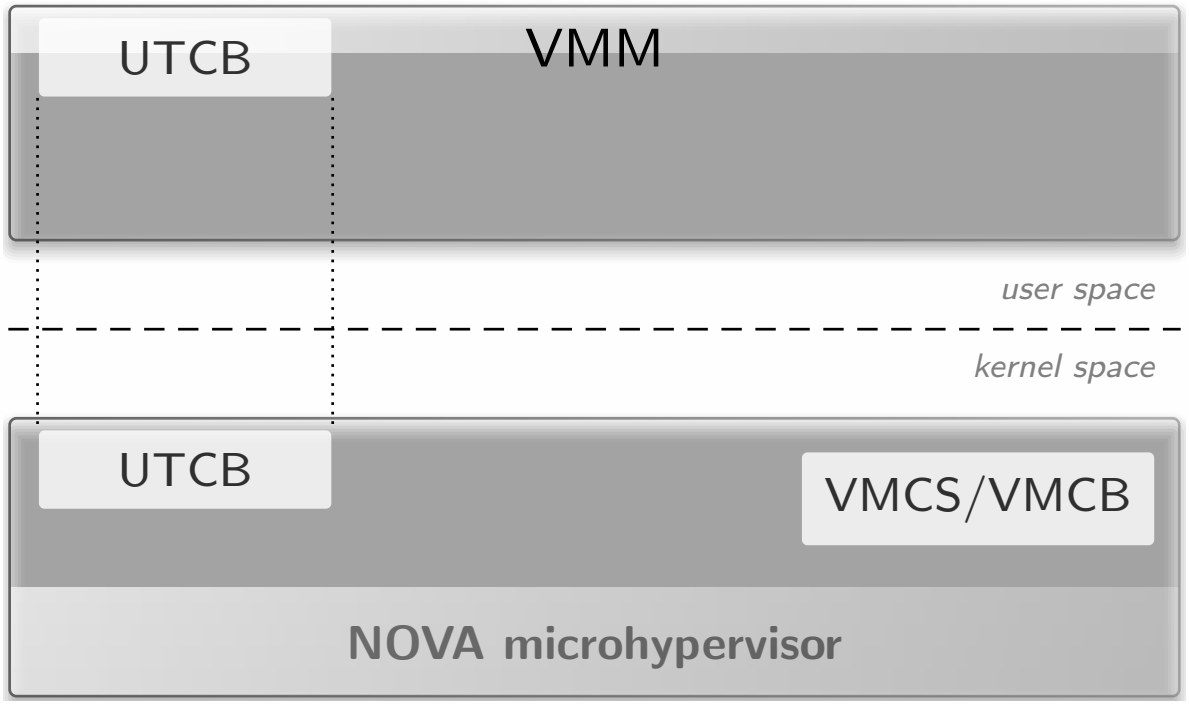
Outline

1. Introduction
2. Kernel interfaces
3. VM interface harmonization
4. VMMs harmonized
5. Conclusion

Flow of a virtualization event

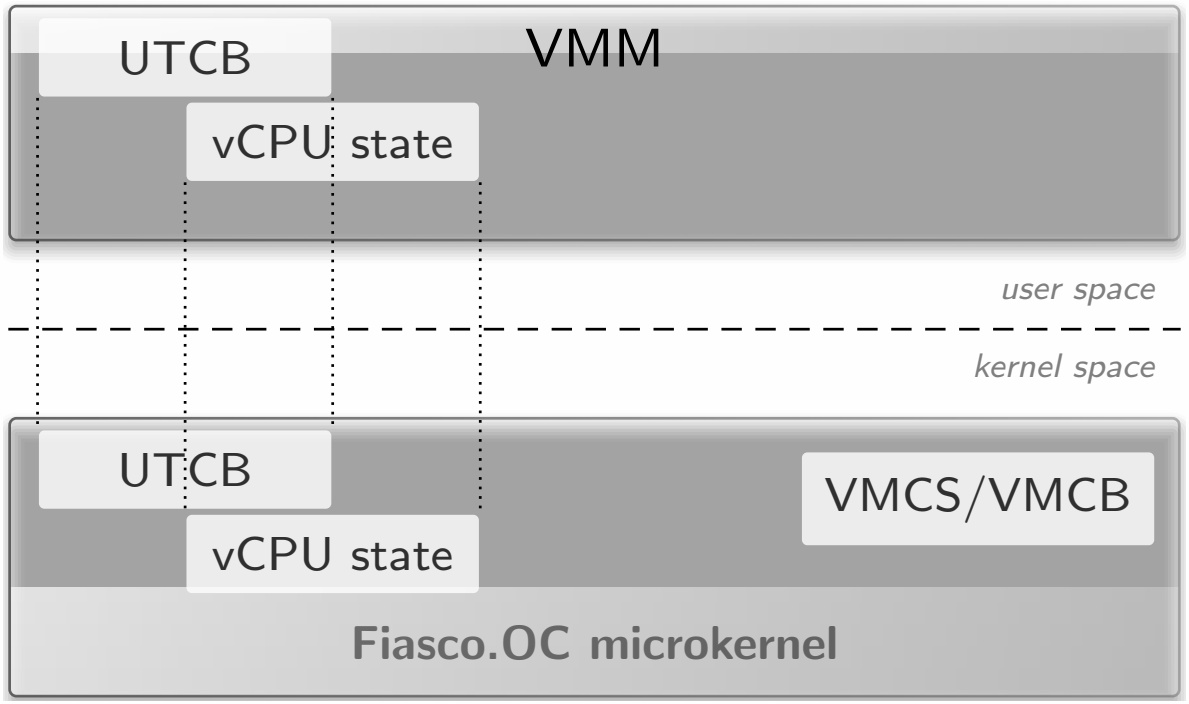


vCPU state on NOVA



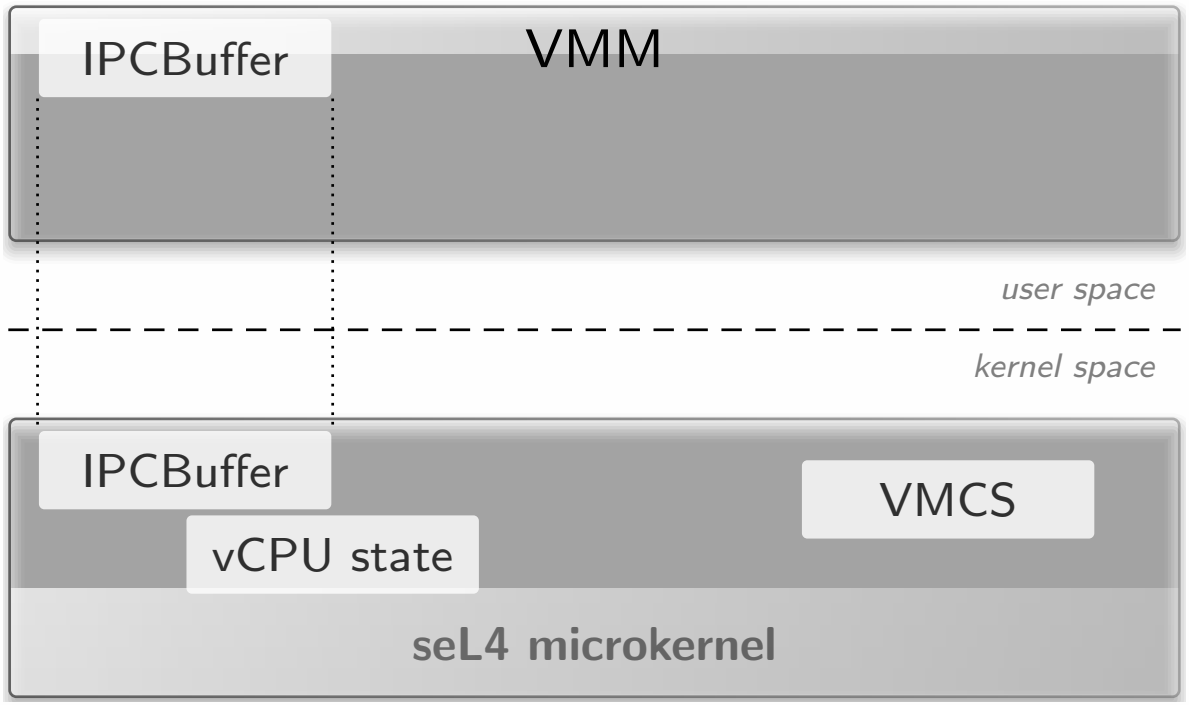
Transfer: UTCB, VMCS/VMCB **agnostic**, **partial** state support

vCPU state on Fiasco.OC



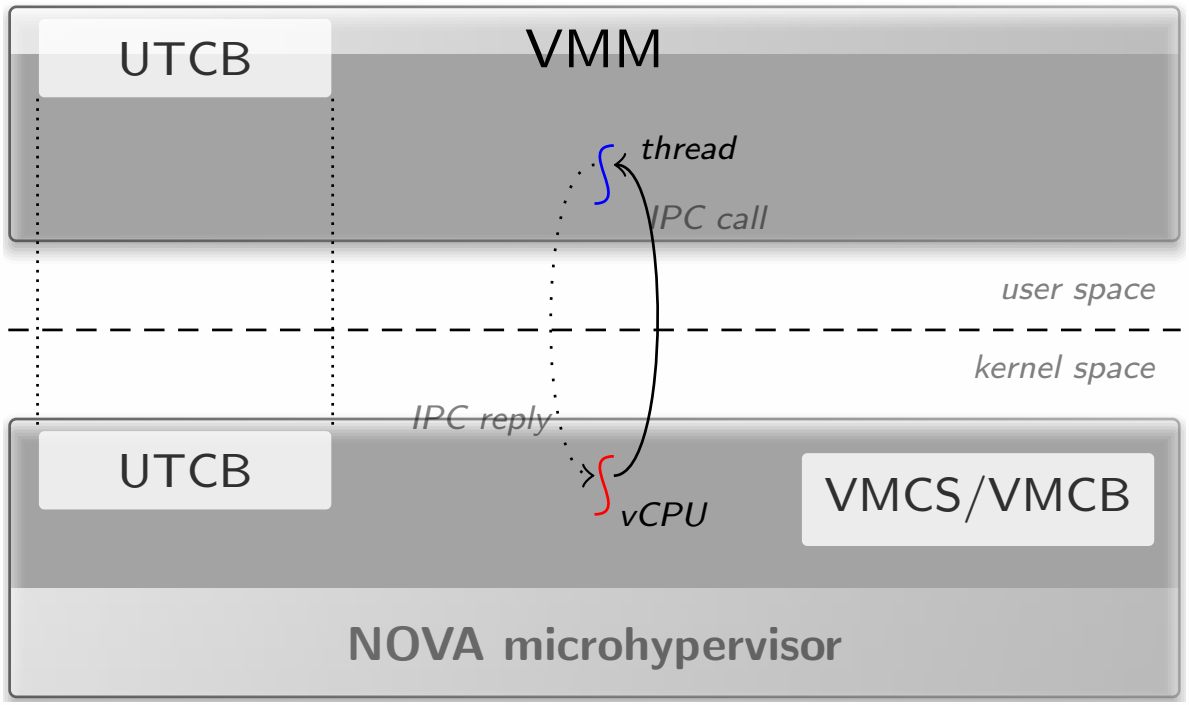
Transfer: vCPU state, **not** VMCS/VMCB **agnostic**, **full** state

vCPU state on seL4

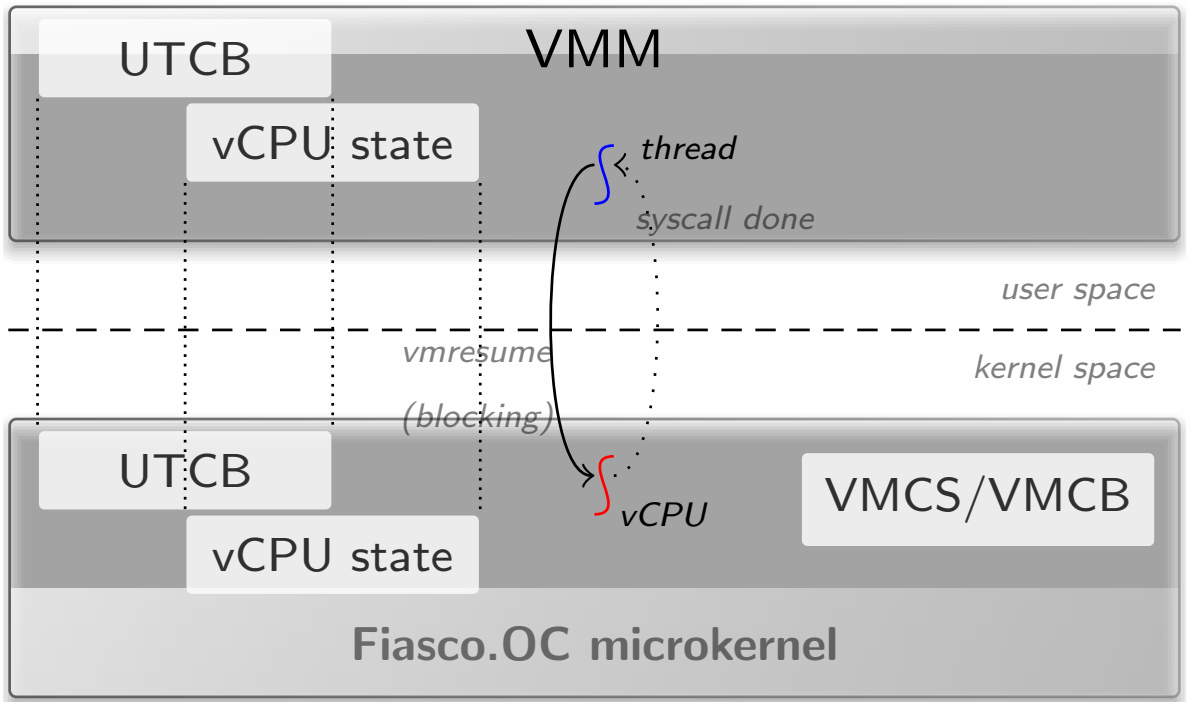


Transfer: **hybrid** - IPCBuffer & **syscall per** VMCS **register**
IPCBuffer: VM exit - 17 registers, VM enter - 3 registers

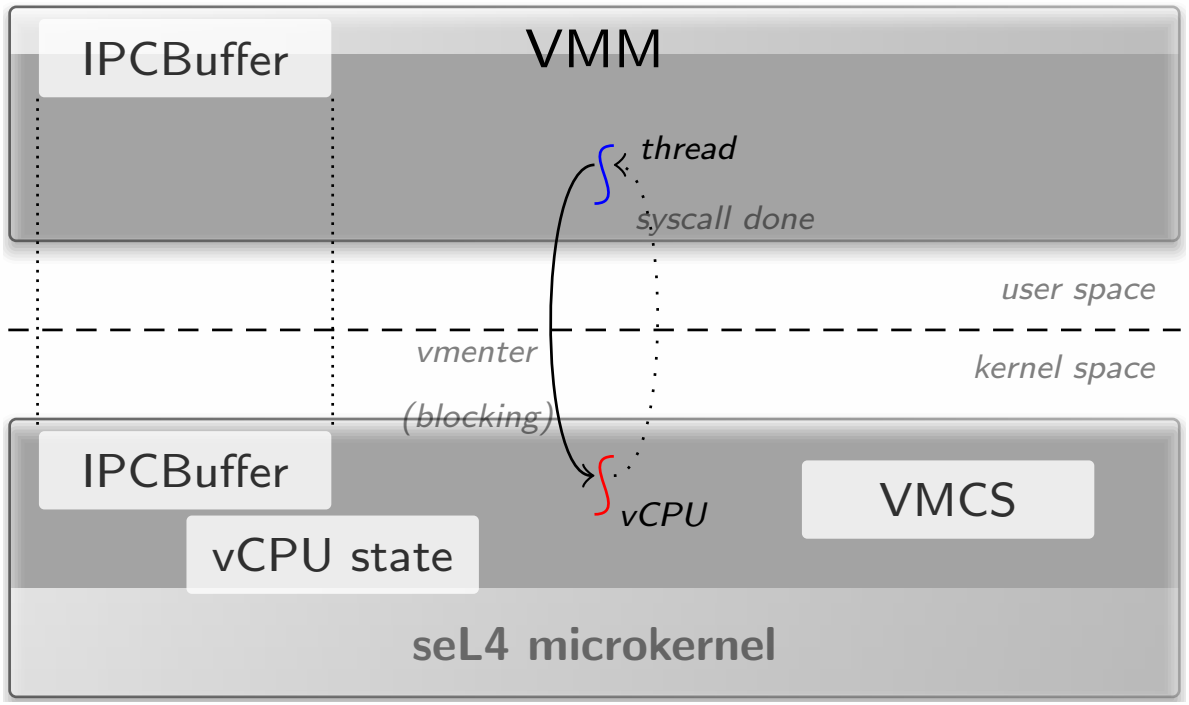
Control flow on NOVA



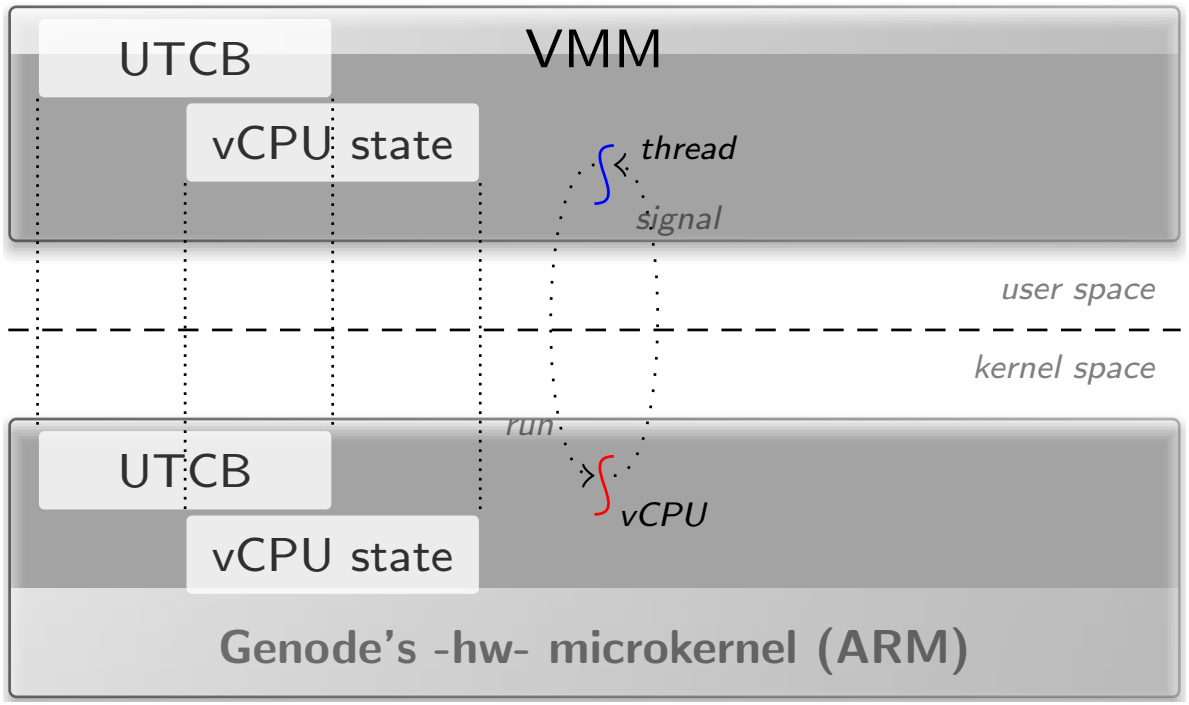
Control flow on Fiasco.OC



Control flow on seL4



Control flow on Genode's -hw-



Outline

1. Introduction
2. Kernel interfaces
3. VM interface harmonization
4. VMMs harmonized
5. Conclusion

Design goals

VMM $\rightarrow$ just **a component**

Genode **components** designed event driven

- Non-blocking thread (**entrypoint**) register for event sources
- Events cause transition in state machine
- State transition by Genode signal or RPC

Design goals

VMM $\rightarrow$ just **a component**

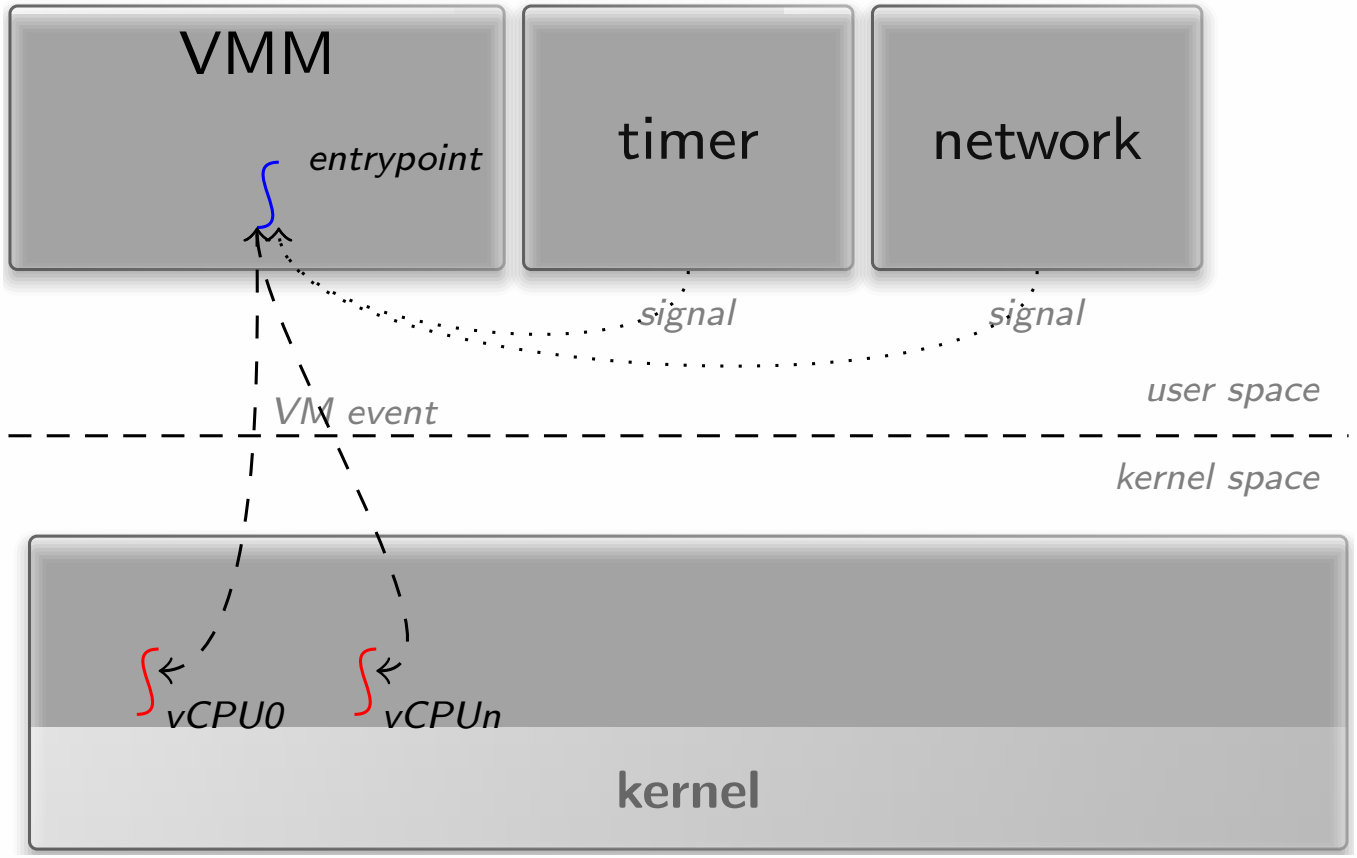
Genode **components** designed event driven

- Non-blocking thread (**entrypoint**) register for event sources
- Events cause transition in state machine
- State transition by Genode signal or RPC

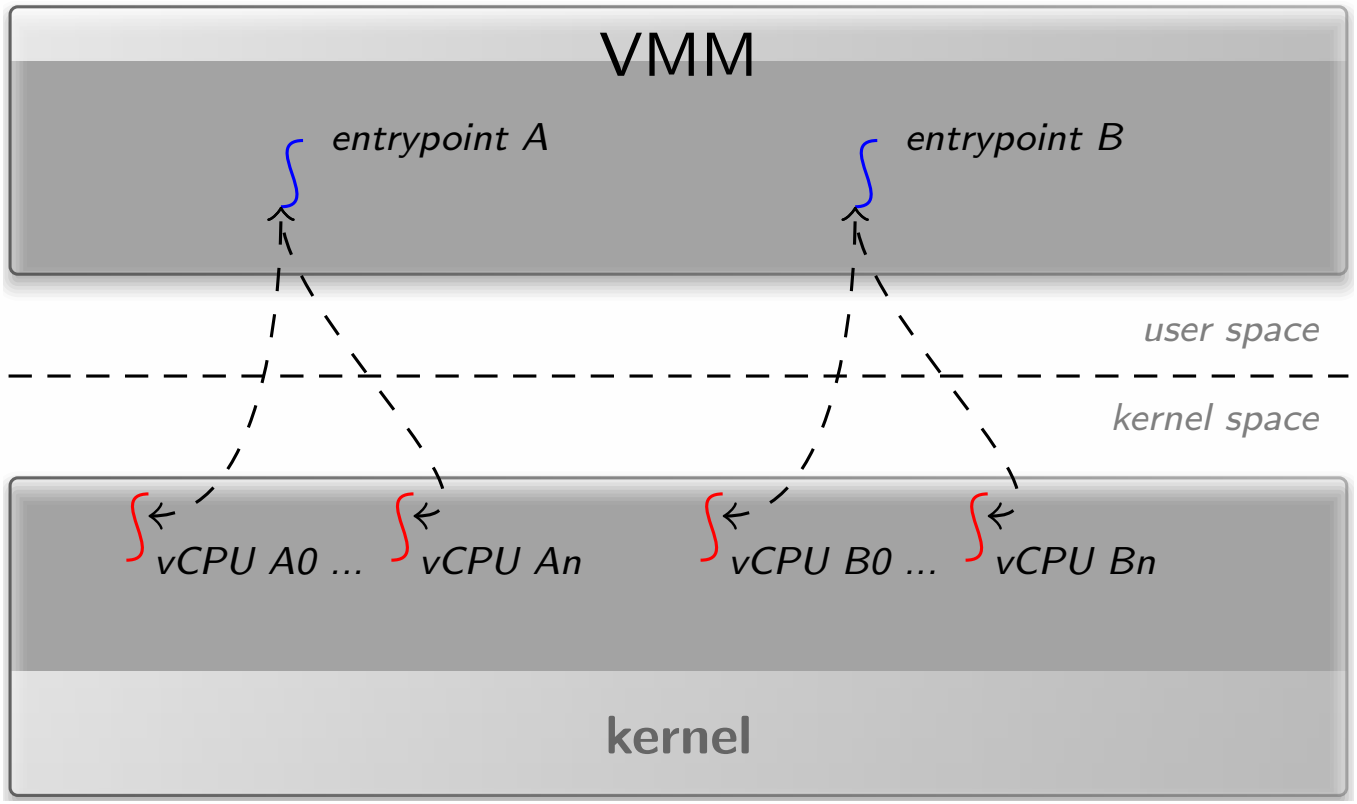
VM event $\rightarrow$ just another event source
I/O event $\rightarrow$ just another event source

Kernel agnostic ABI
Unified vCPU state per platform

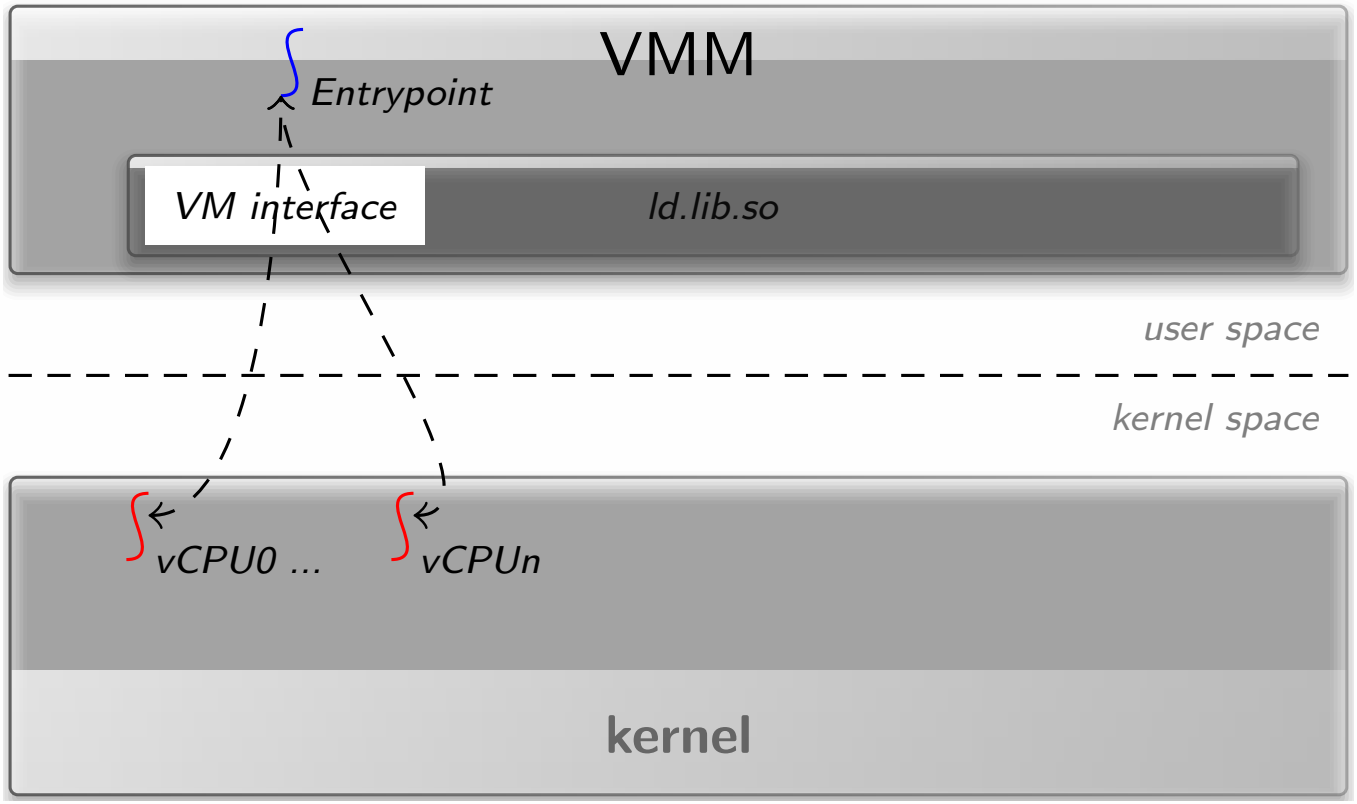
Envisioned vCPU handling



Envisioned vCPU handling - multi core



VM interface - kernel agnostic



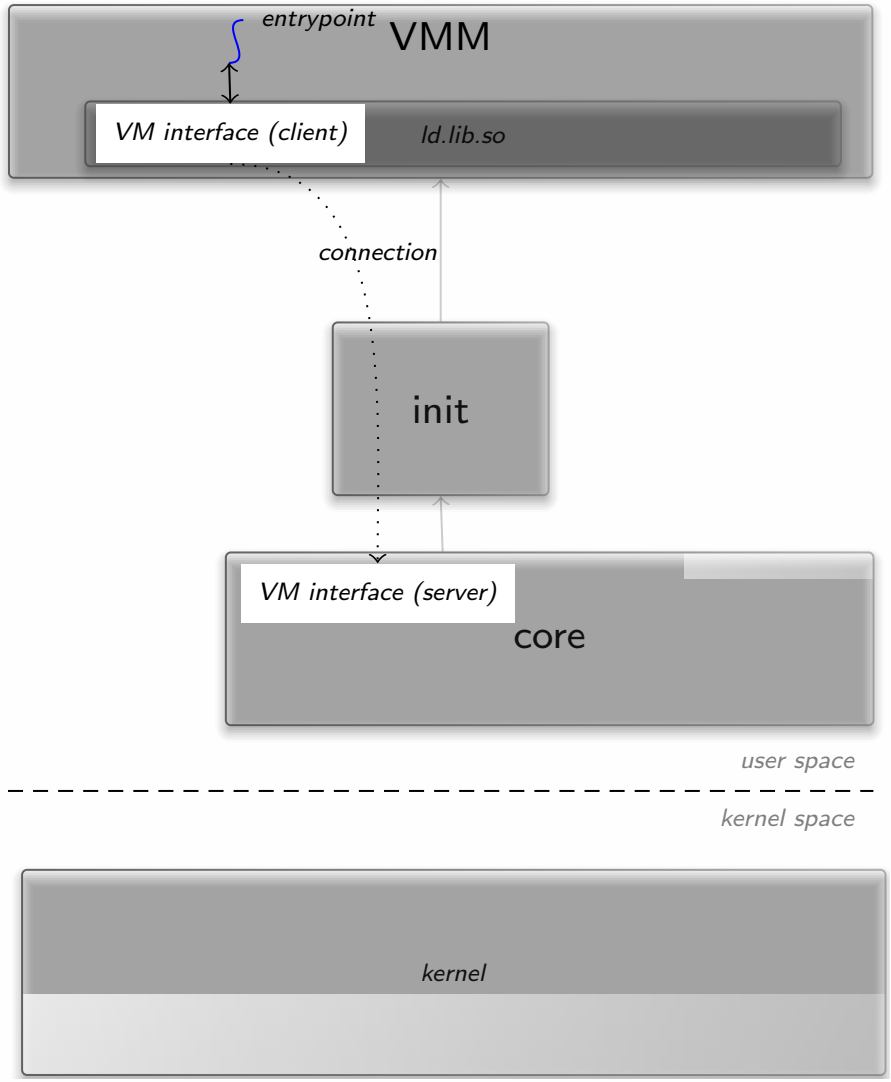
Genode -base- library with unified ABI in ld.lib.so

VM interface - kernel agnostic

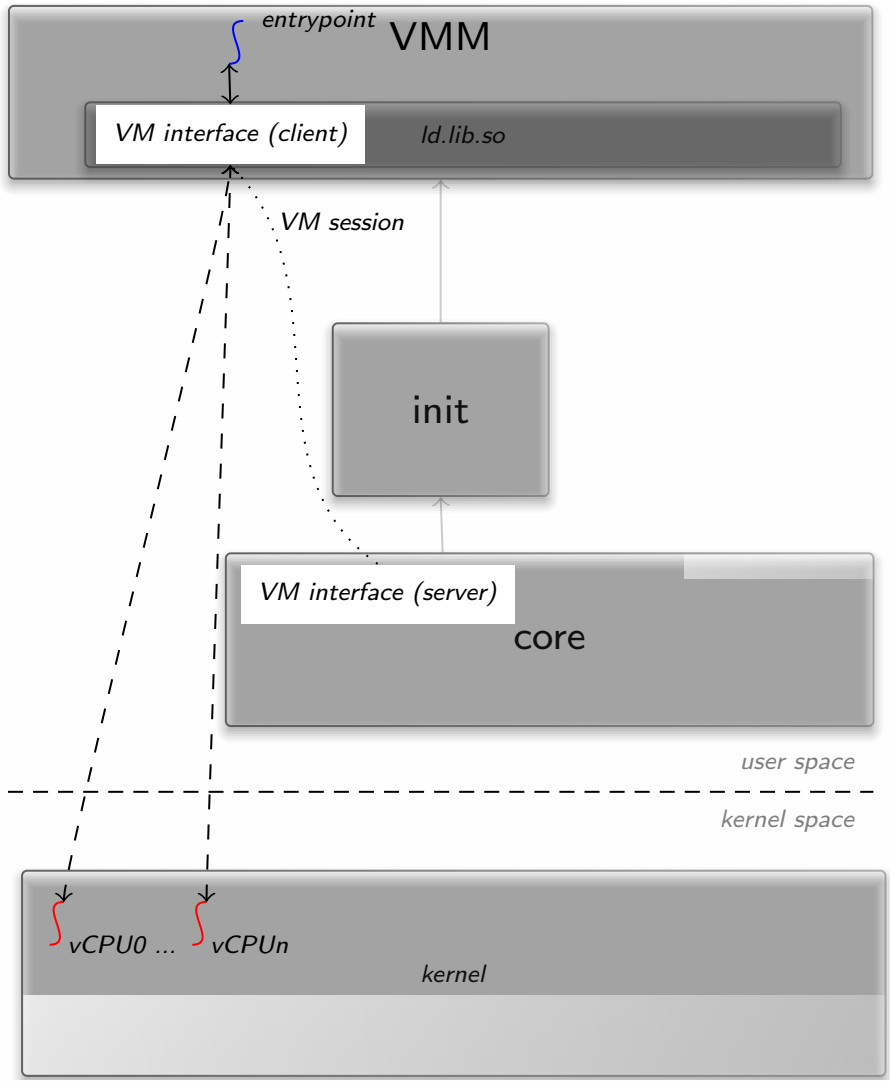
VM connection/session $\rightarrow$ VM address space established

- create_vcpu() - setup new vCPUs
- cpu_state() - access to guest state
- attach/detach() - memory management of VM
- VM_handler class - registration for VM event handling
- run/pause() - control execution of vCPUs - non-blocking

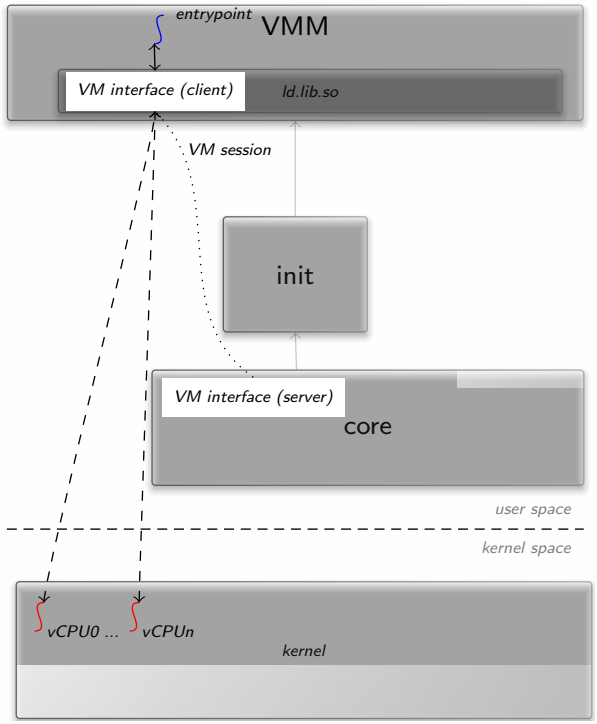
VM interface - kernel agnostic



VM interface - kernel agnostic



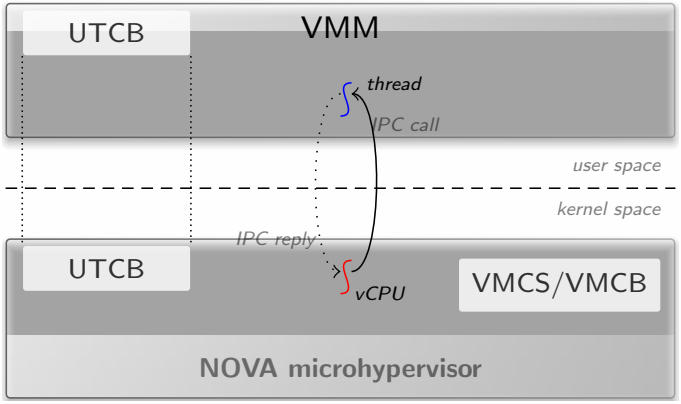
VM interface - kernel agnostic

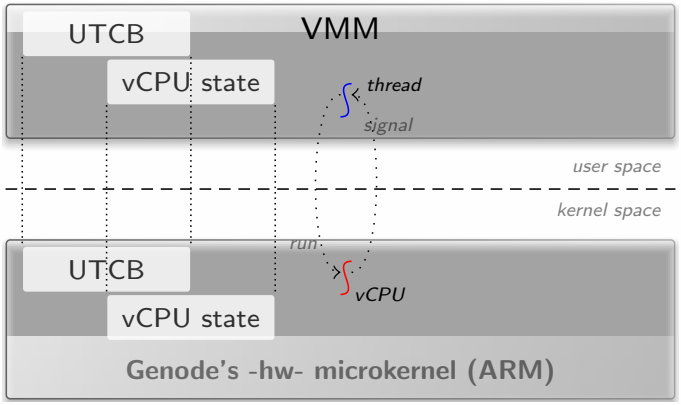


Server: 200-400 LOC

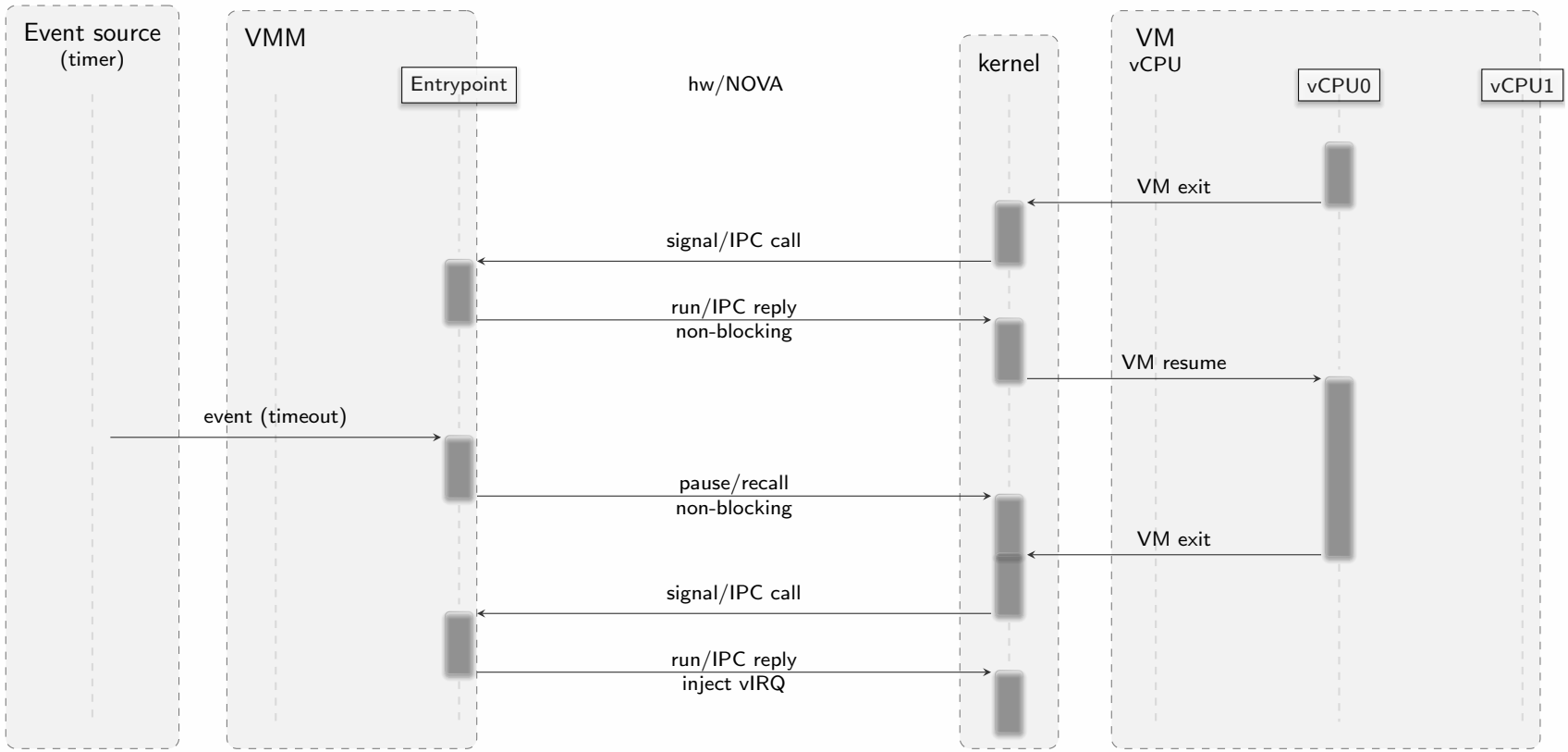
Client: NOVA, seL4: ~500 - Fiasco.OC: ~1000 - hw: ~30 LOC

Control flow on Genode's -hw- and NOVA

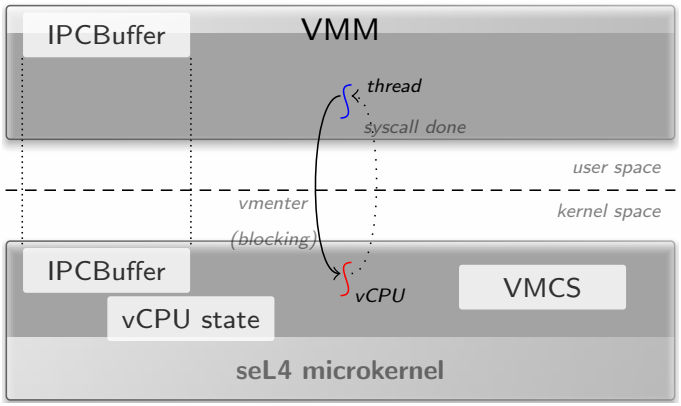


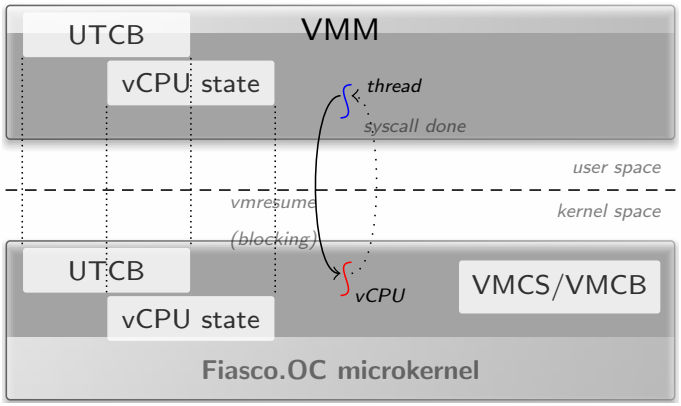


Control flow on Genode's -hw- and NOVA

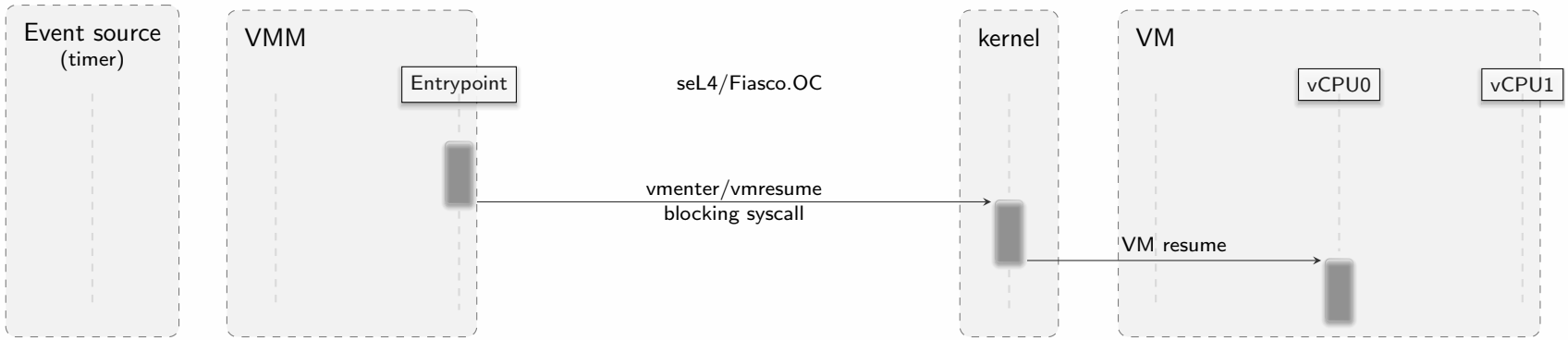


Control flow on seL4 and Fiasco.OC





Control flow on seL4 and Fiasco.OC

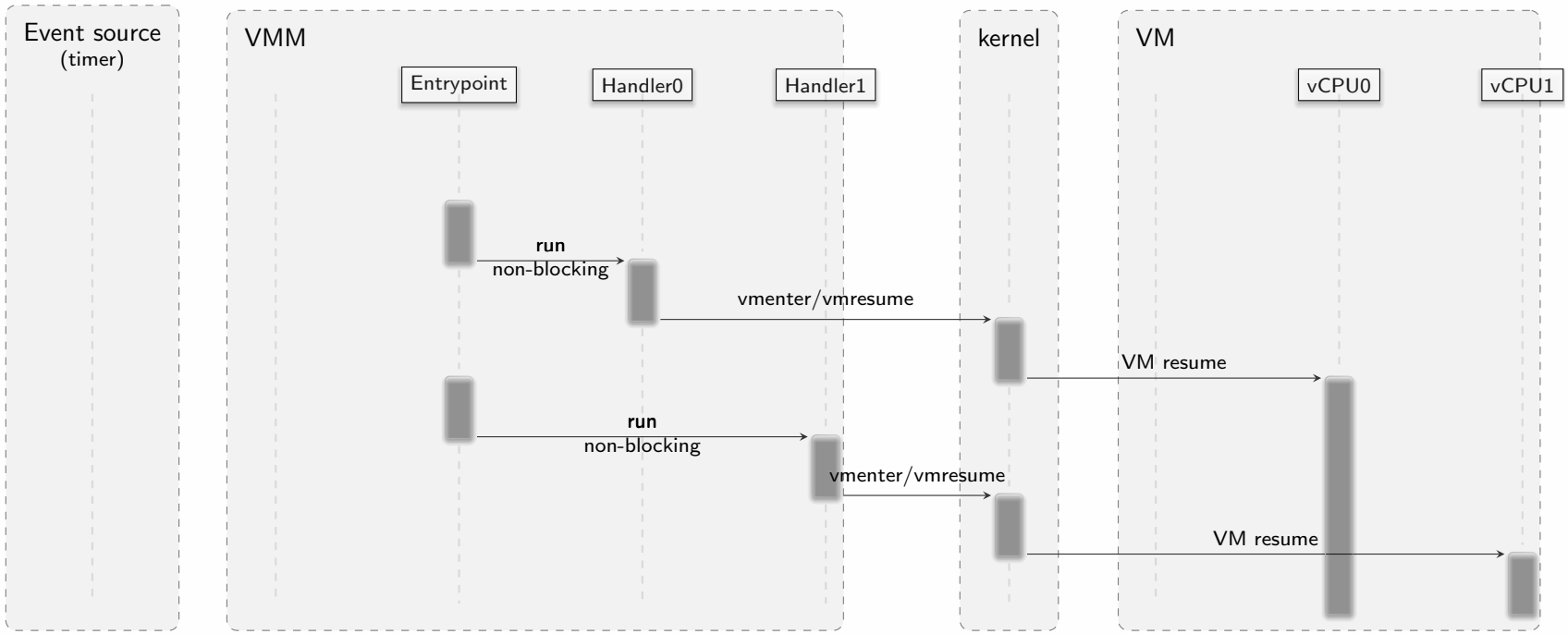


Blocking syscall unfortunate → complicates life

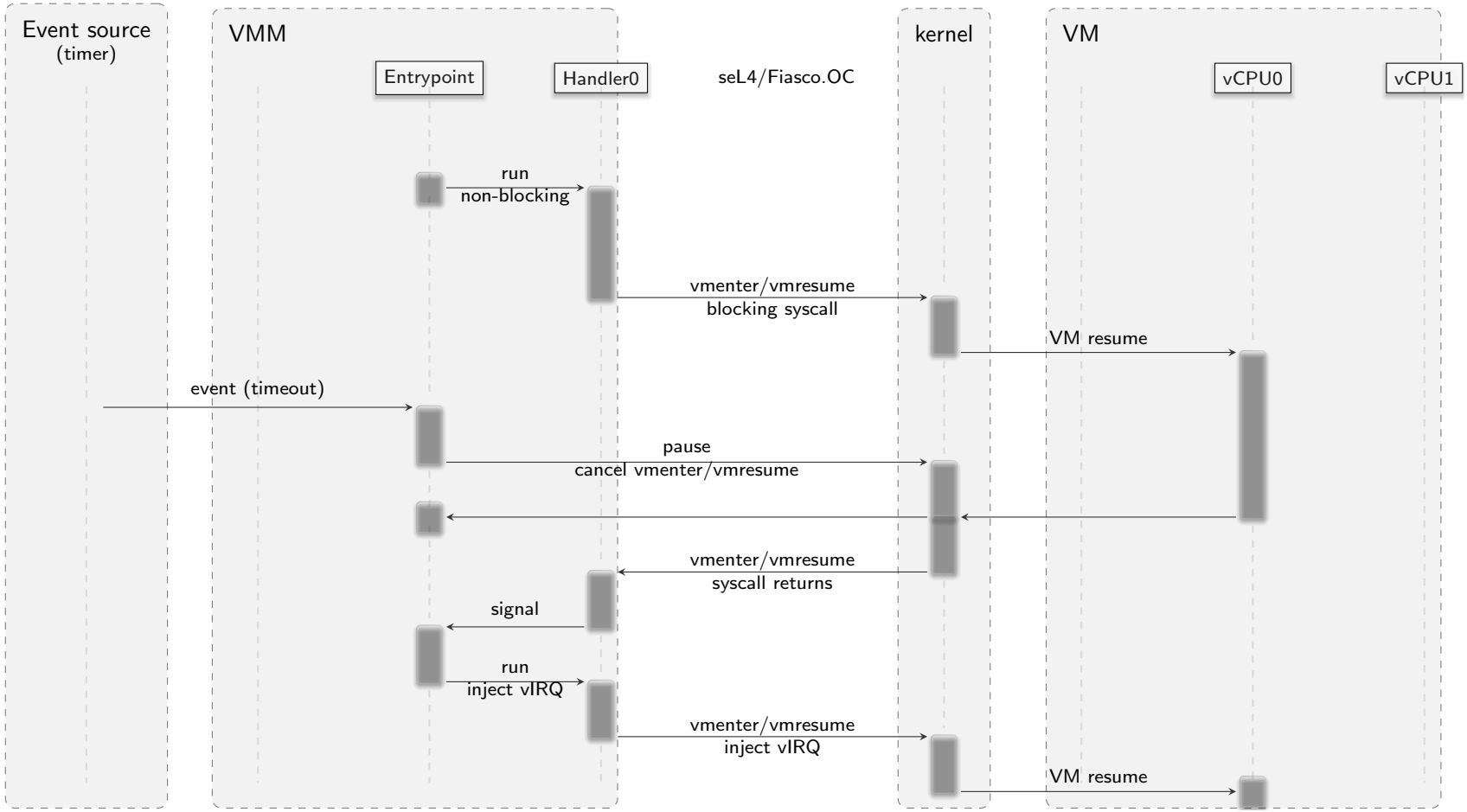
- Kernels provide mechanism to cancel

Avoid special case handling in Genode for first take
→ Workaround: spawn per vCPU extra thread

Control flow on seL4 and Fiasco.OC



Control flow on seL4 and Fiasco.OC



Outline

1. Introduction
2. Kernel interfaces
3. VM interface harmonization
4. VMMs harmonized
5. Conclusion

VMM unit test

Control flow and exit handling on few instructions

- Multiple vCPUs, multiple EPs, multiple physical CPUs

VMM unit test

Control flow and exit handling on few instructions

- Multiple vCPUs, multiple EPs, multiple physical CPUs

sel4 v9.0:

- Kernel fault on VMEnter by non vCPU thread $\rightarrow$ patch

VMM unit test

Control flow and exit handling on few instructions

- Multiple vCPUs, multiple EPs, multiple physical CPUs

sel4 v9.0:

- Kernel fault on VMEnter by non vCPU thread $\rightarrow$ patch
- No unrestricted guest support $\rightarrow$ patch

VMM unit test

Control flow and exit handling on few instructions

- Multiple vCPUs, multiple EPs, multiple physical CPUs

sel4 v9.0:

- Kernel fault on VMEnter by non vCPU thread $\rightarrow$ patch
- No unrestricted guest support $\rightarrow$ patch
- Scheduling bug if vCPU spins $\rightarrow$ starvation $\rightarrow$ patch

VMM unit test

Control flow and exit handling on few instructions

- Multiple vCPUs, multiple EPs, multiple physical CPUs

sel4 v9.0:

- Kernel fault on VMEnter by non vCPU thread $\rightarrow$ patch
- No unrestricted guest support $\rightarrow$ patch
- Scheduling bug if vCPU spins $\rightarrow$ starvation $\rightarrow$ patch
- Kernel denies to boot on non VT-x platforms $\rightarrow$ patch

VMM unit test

Control flow and exit handling on few instructions

- Multiple vCPUs, multiple EPs, multiple physical CPUs

sel4 v9.0:

- Kernel fault on VMEnter by non vCPU thread → patch
- No unrestricted guest support → patch
- Scheduling bug if vCPU spins → starvation → patch
- Kernel denies to boot on non VT-x platforms → patch

→ Working toy VMM on all 3 kernels
→ no AMD support by seL4

Seoul VMM

Replaced all NOVA specific parts

- Simple Genode based guests for testing

Running again after few days on Genode/NOVA

Seoul VMM

Replaced all NOVA specific parts

- Simple Genode based guests for testing

Running again after few days on Genode/NOVA

Various debugging sessions on Fiasco.OC and seL4
→ war stories (backup slides)
→ 1 kernel patch for seL4 and 1 for Fiasco.OC

Seoul VMM

Replaced all NOVA specific parts

- Simple Genode based guests for testing

Running again after few days on Genode/NOVA

Various debugging sessions on Fiasco.OC and seL4
→ war stories (backup slides)
→ 1 kernel patch for seL4 and 1 for Fiasco.OC

State: kernel agnostic Seoul VMM on all 3 kernels

- Guests: Genode VMs, Linux VM+network+SMP
- seL4: kernel fault on Linux SMP VM → not investigated

VBox 5 VMM - current state

Work in progress - current state:

- Kernel agnostic VBox5 binary ready and runnable
- NOVA: simple Genode VMs running again
- seL4/Fiasco.OC: VM gets up, fails/hangs early

Known remaining challenges:

- Guest FPU state access required
  - Missing in VM interface
  - Support by seL4 and Fiasco.OC unclear
- seL4: no support for 64bit guests

Outline

1. Introduction
2. Kernel interfaces
3. VM interface harmonization
4. VMMs harmonized
5. Conclusion

Conclusion

Dare the impossible $\rightarrow$ possible*

- Restrictions depending on the kernel

Roadmap:

- Finish VBox5 adaptation
- Extend -hw- kernel with VT-x extensions
- Optional: support other platforms, e. g. ARM

Benefits:

- Portable VMMs across kernels
- Genode users have the ultimate kernel choice

Thank you

Genode OS Framework

https://genode.org

Source code at GitHub

https://github.com/genodelabs/genode

Stories around Genode

https://www.genodians.org

Genode Labs GmbH

https://www.genode-labs.com

Backup

Seoul VMM - war stories I

Fiasco.OC:

- In-guest faults during protected→page mode transition
- reason: EFER status of host taken instead of guest
- Fiasco.OC: can be runtime configured → good

seL4:

- seL4: EFER register not saved on VMexit → kernel patch

Seoul VMM - war stories II

- CR* shadow/mask handling required on seL4 & Fiasco.OC
- Took some time, caused friction
- Open issue:
  - Kernels overwrites some bits in CR* to adhere to hardware requirements
  - Overriden bits not known/announced to VMM
  - Read back CR* modifications contains changes of hypervisor and VM mixed
    - Leads to various invalid guest states
  - Heuristics required - unexpected but manageable:
    - Job of Fiasco.OC/seL4 vs VMM ?

Seoul VMM - war stories III

Another test VM:

- seL4 and NOVA: worked fine
- Fiasco.OC: invalid guest state

Long long sessions of VM state diffs between kernels

- Happens on switch from protected $\rightarrow$ real mode

Source reason:

- vIRQ injection can not be reset by VMM on Fiasco.OC
- Patching kernel helps