QEMU and SystemC

March 2011

QUF'11
Grenoble

Màrius Montón

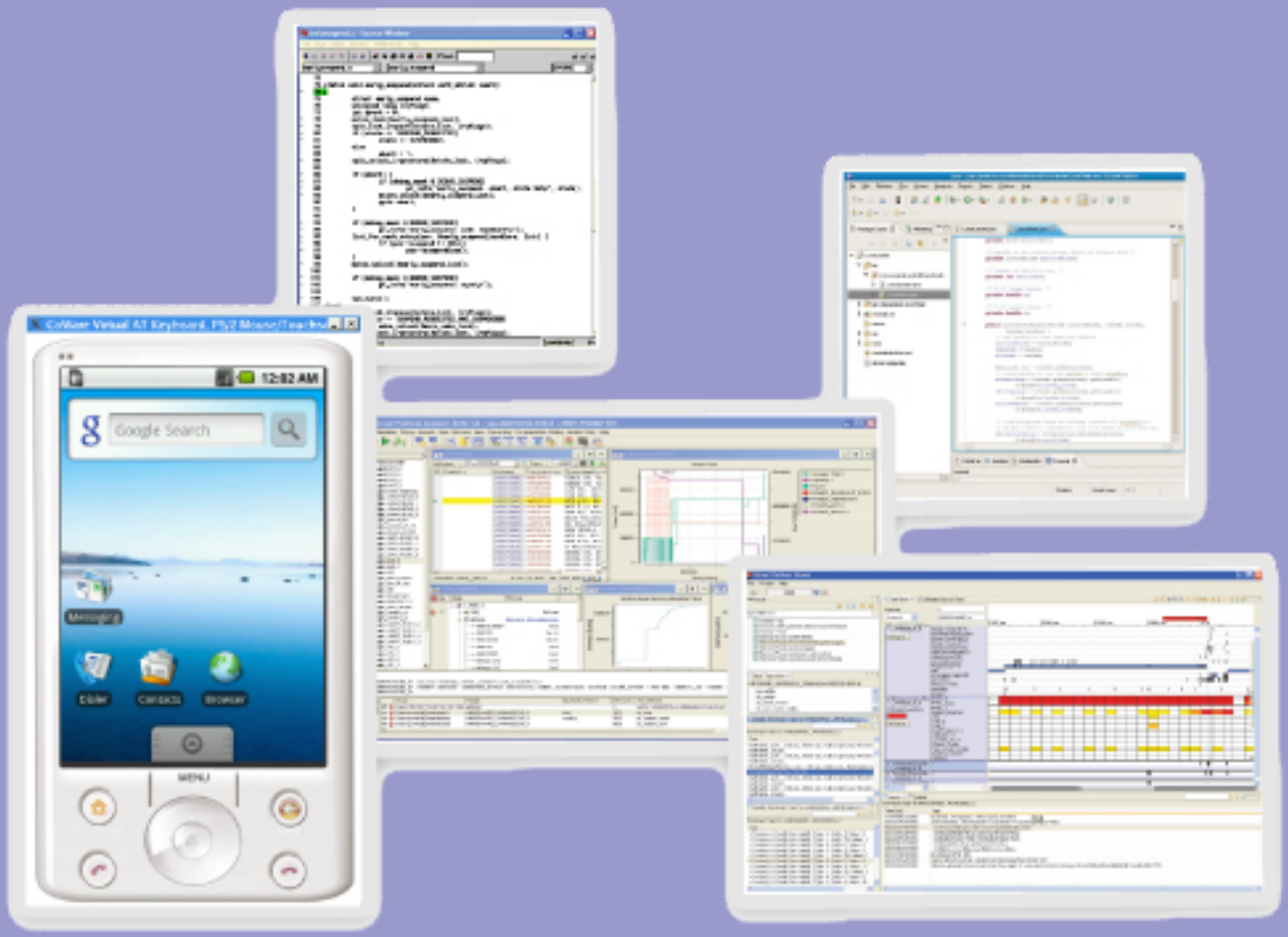
Outline

- Introduction
- Objectives
- Virtual Platforms and SystemC
- Checkpointing for SystemC
- Conclusions

Introduction – Virtual Platforms

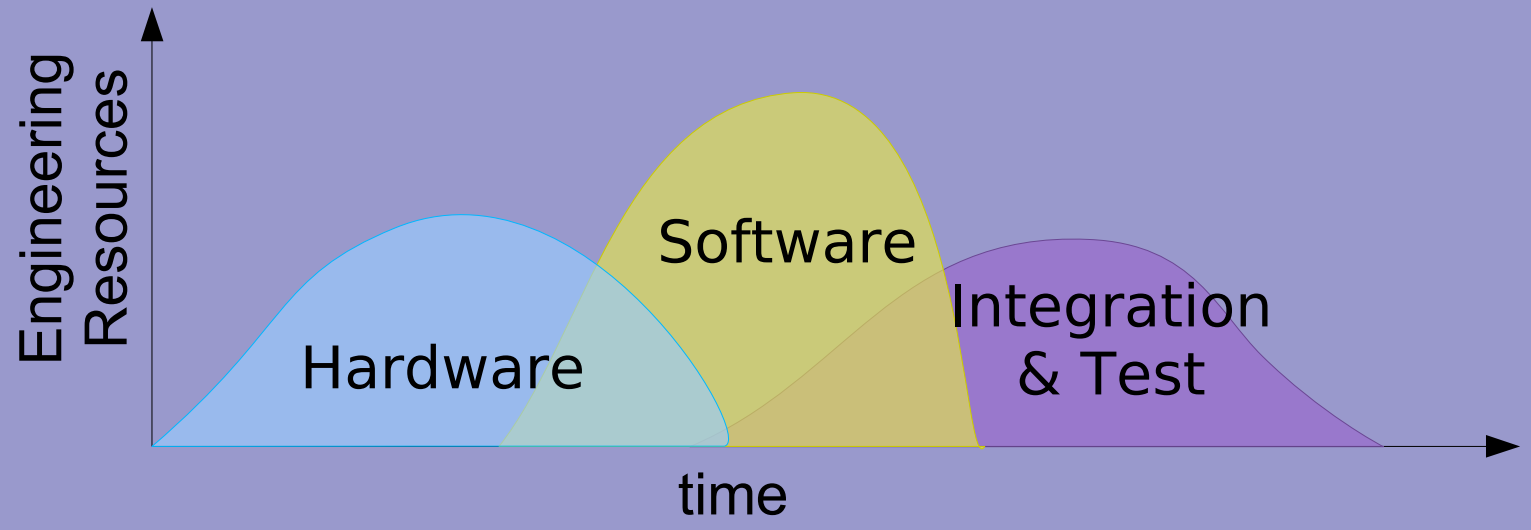
- Functional models of physical platforms
- Target SW unable to distinguish virtual platform from real HW
- Run SW or OS on Virtual HW
- Develop SW for non-existing HW
- Simulate complex system interconnectivity

Introduction – Virtual Platforms

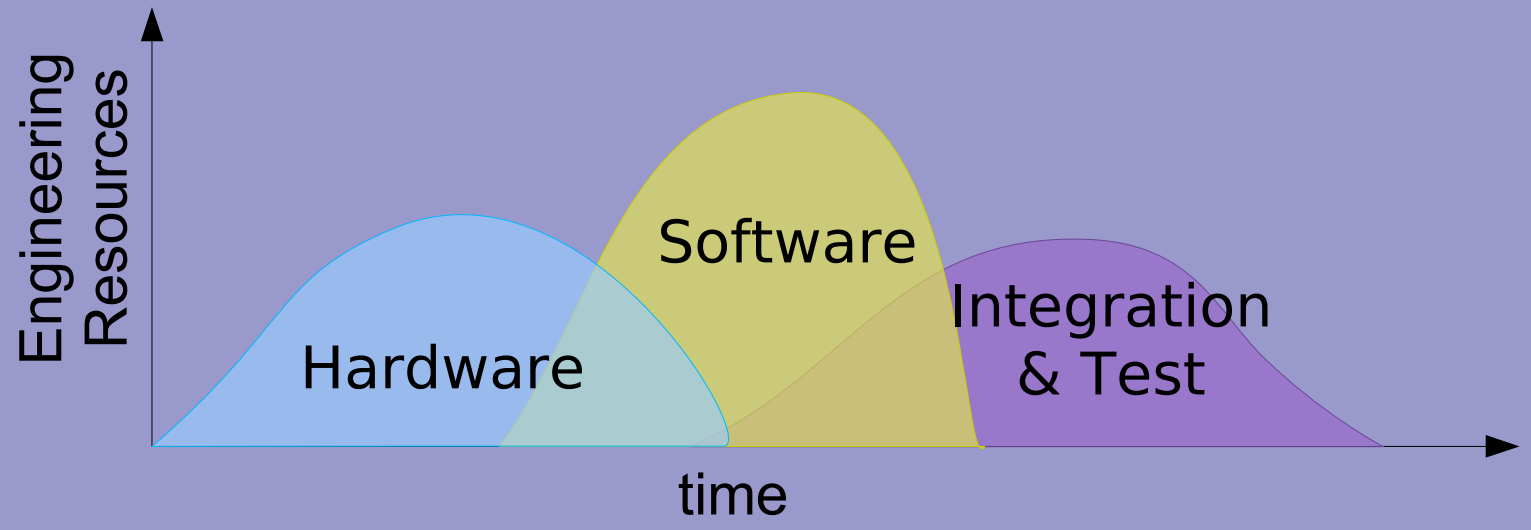


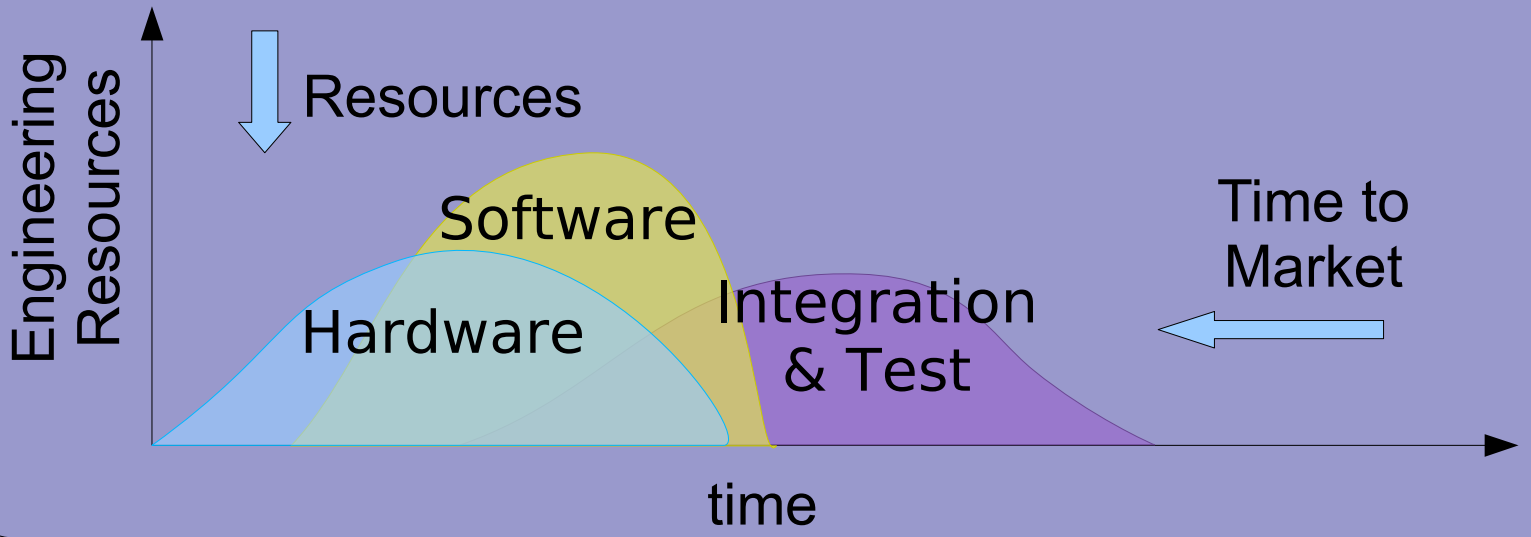
Introduction – Virtual Platforms

Virtual Platform Design

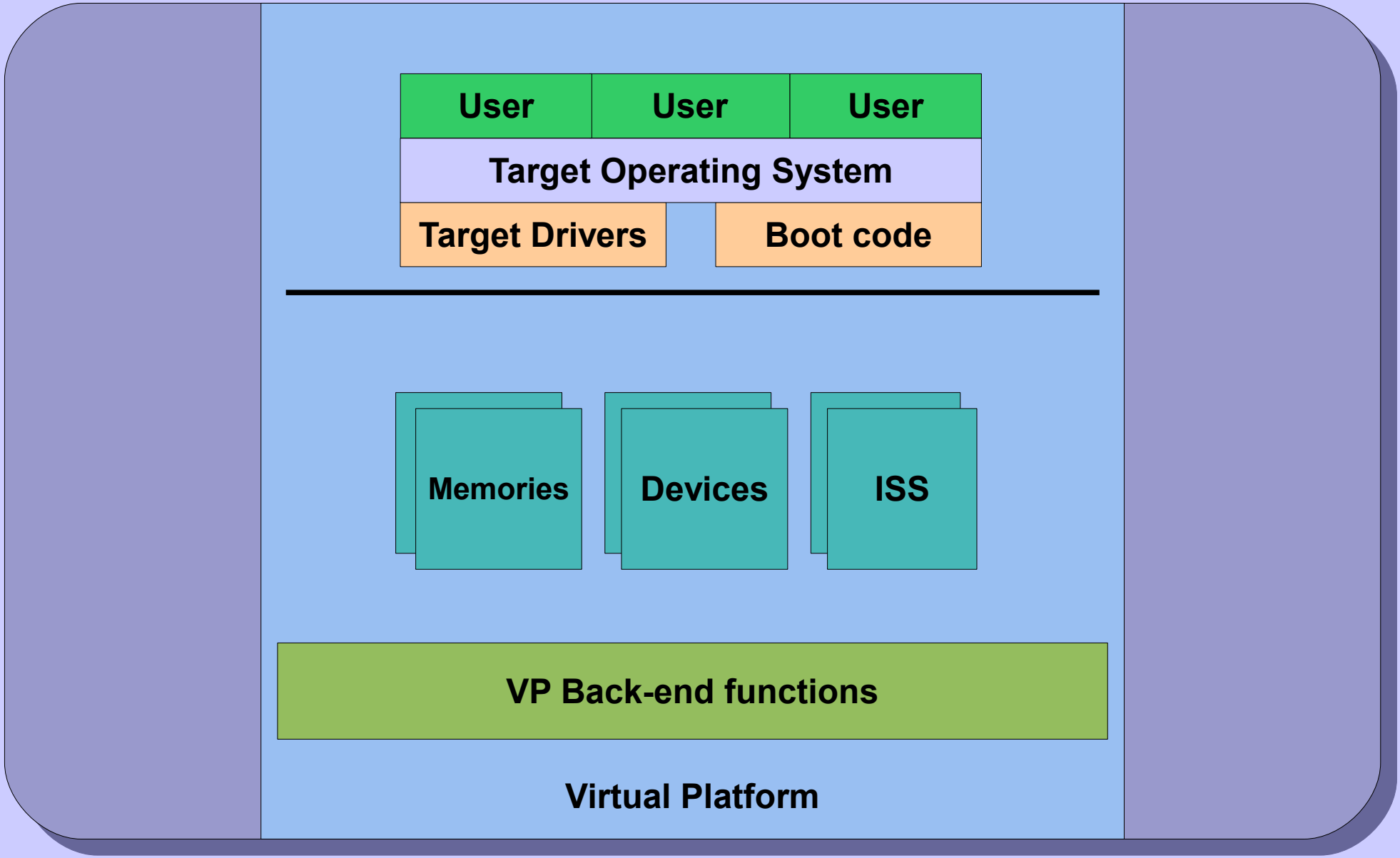


Virtual Platform Design





Generic Virtual Platform Diagram



Introduction – SystemC & TLM-2

- SystemC language for systems description (HW mainly)
- OSCI simulator
- TLM-2 standarizes communication model
  - Sockets to emulate any memory-mapped bus
- De facto standard for system modeling

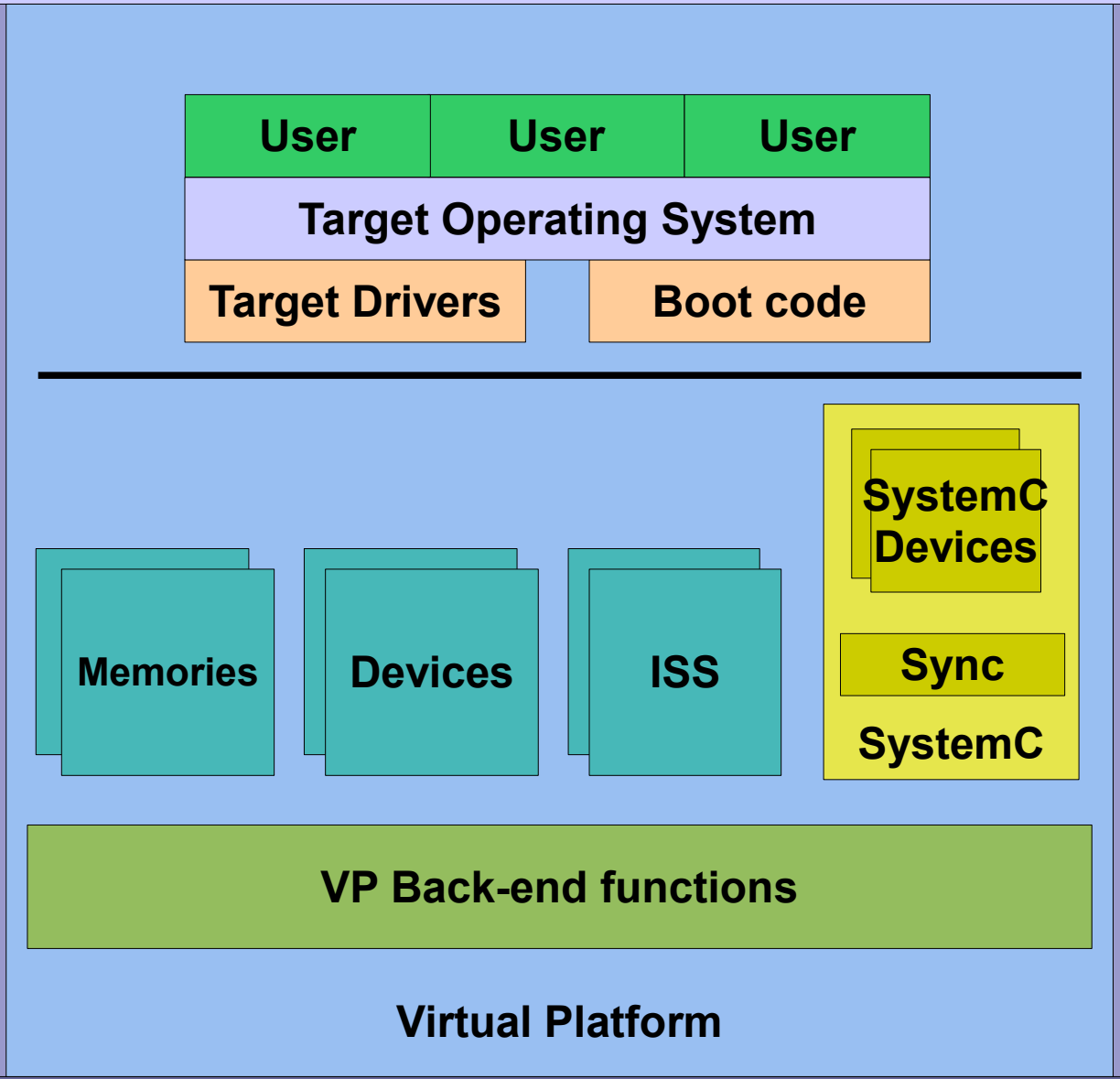
Introduction – VP Languages

- Different Virtual Platforms uses different languages (C, C++, DLM, ...) and APIs
- HW engineers know (or should) SystemC, not other languages
- Different VP → Rewrite own models
- No interoperability

Objectives

- Add SystemC-TLM to any Virtual Platform
  - Different strategies to add two simulators
  - Add SystemC-TLM support to an open-sourced VP
- Add checkpointing support to SystemC
  - C++ not checkpointable
  - Overcome limitations

Generic Virtual Platform Diagram

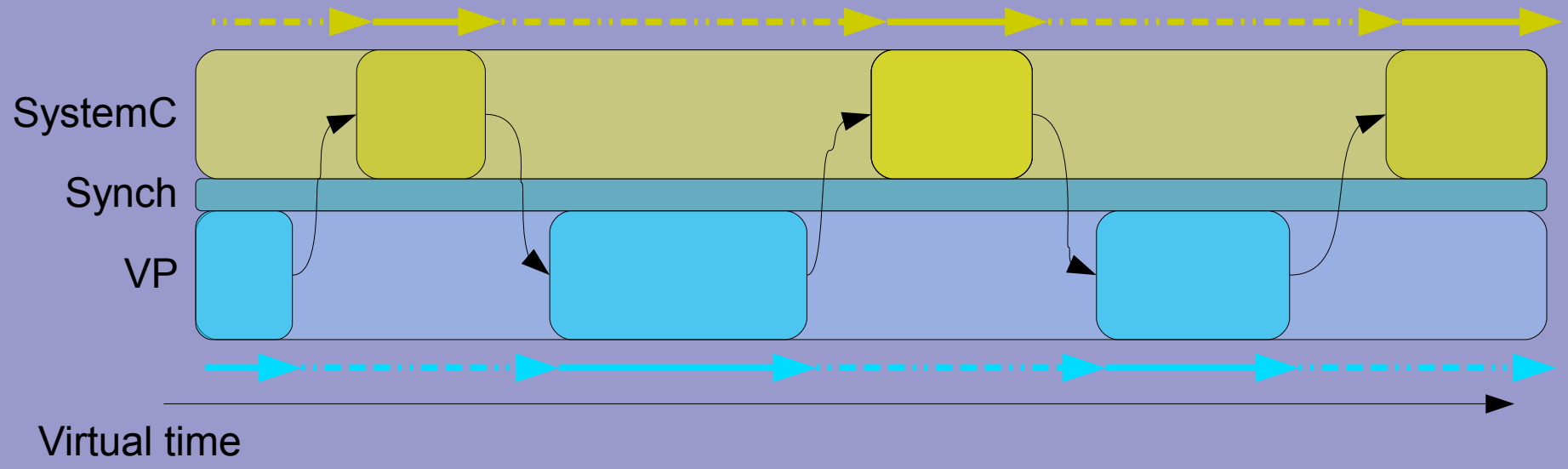


Virtual Platforms and SystemC

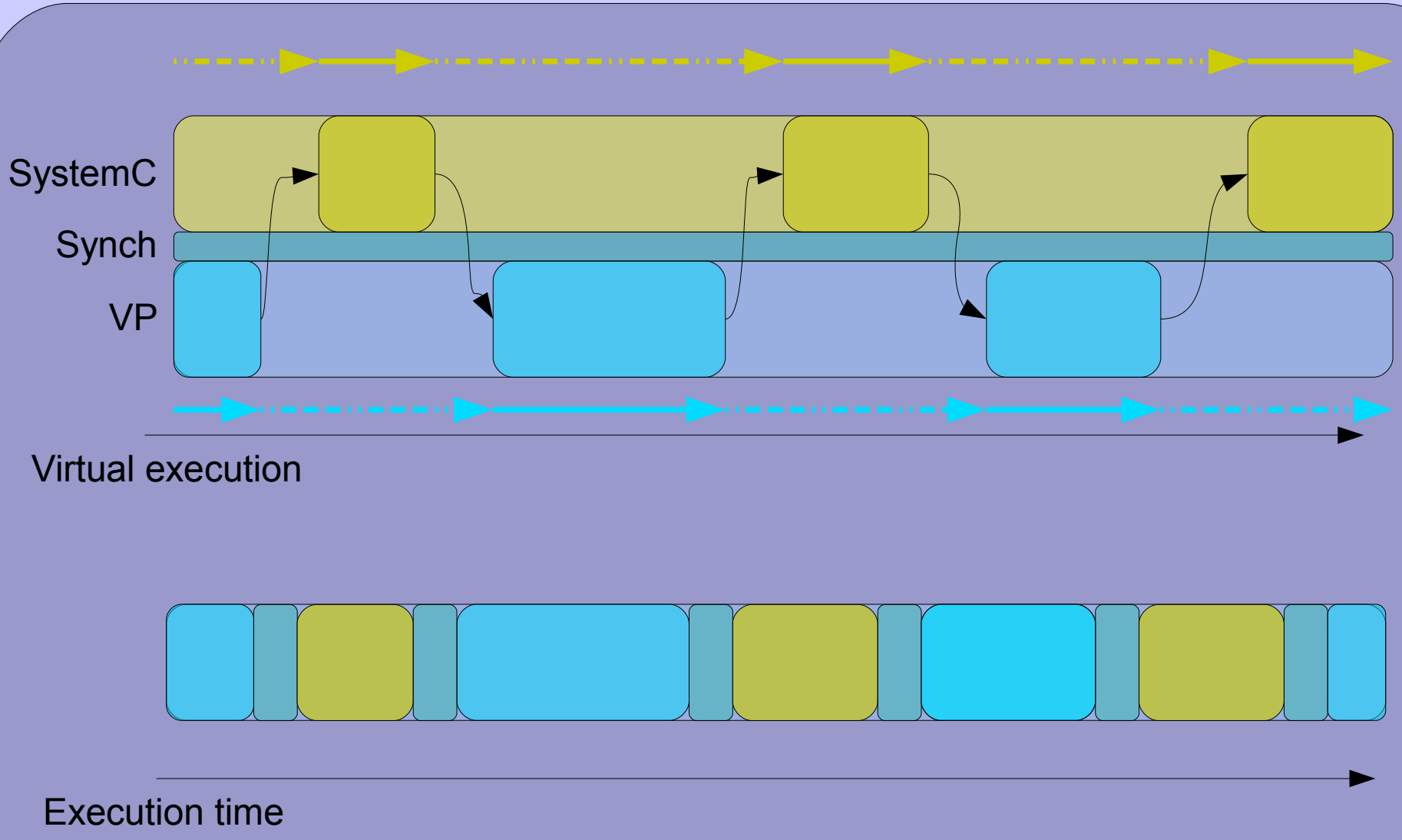
- Link together two simulators
- Synchronization strategy
- Generic bridge
- Support for generic TLM-2 devices
  - LT, AT, DMI…

Virtual Platforms and SystemC

- Link together two simulators

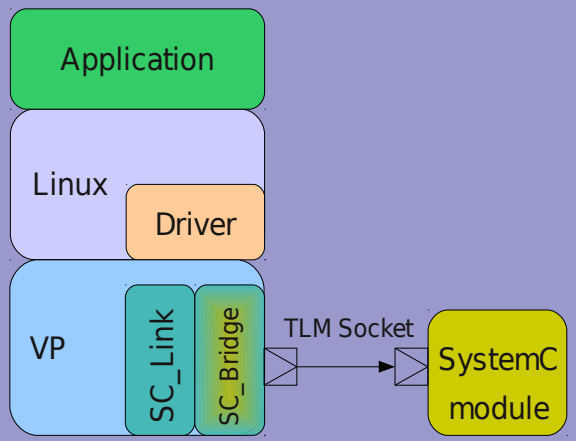


Virtual Platforms and SystemC

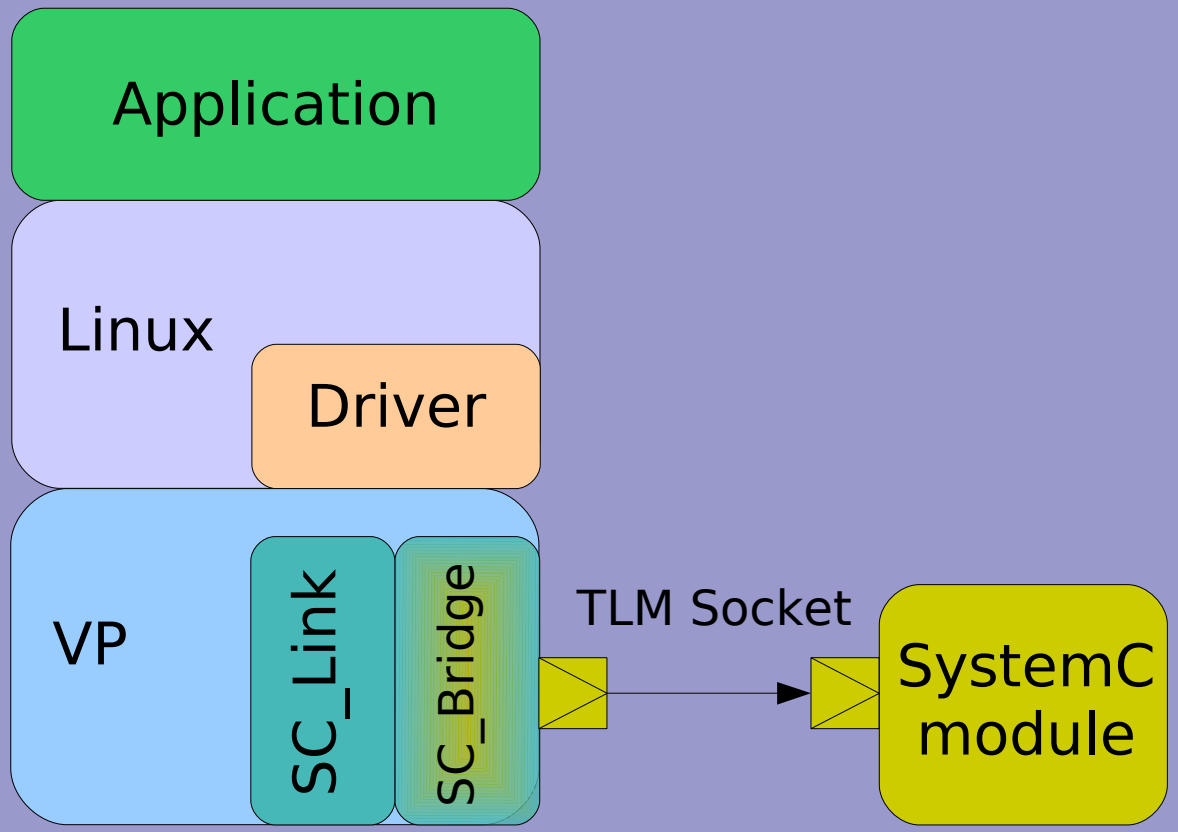


QEMU-SC

- Transition from RTL to TLM
- Generic fabric bus (TLM) → any architecture
- QEMU master and SystemC slave (simulation)
  - QEMU manages simulation
- Focus on few SystemC devices



QEMU-SC


Application
Linux
Driver
VP
SC_Link
SC_Bridge
TLM Socket
SystemC module

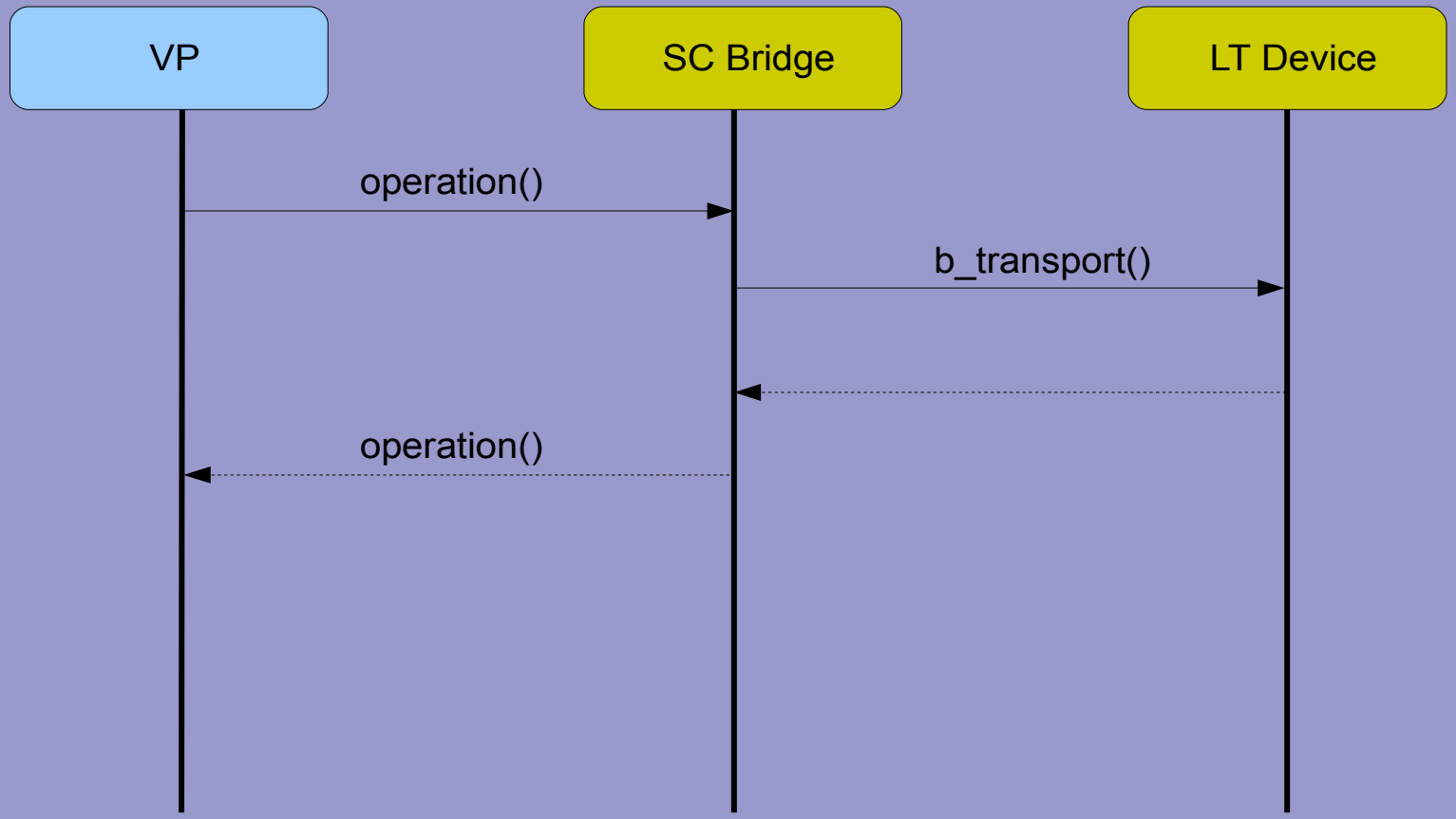
Synchronization

- Only synchronize when needed (*sc_start()*)
  - When SystemC devices are accessed
  - When pending event in current simulation time
    - SystemC events list
  - When I/O to/from SystemC device
    - Capture or notify all I/O in SystemC device
  - QuantumKeeper asks to
    - To adhere to standard

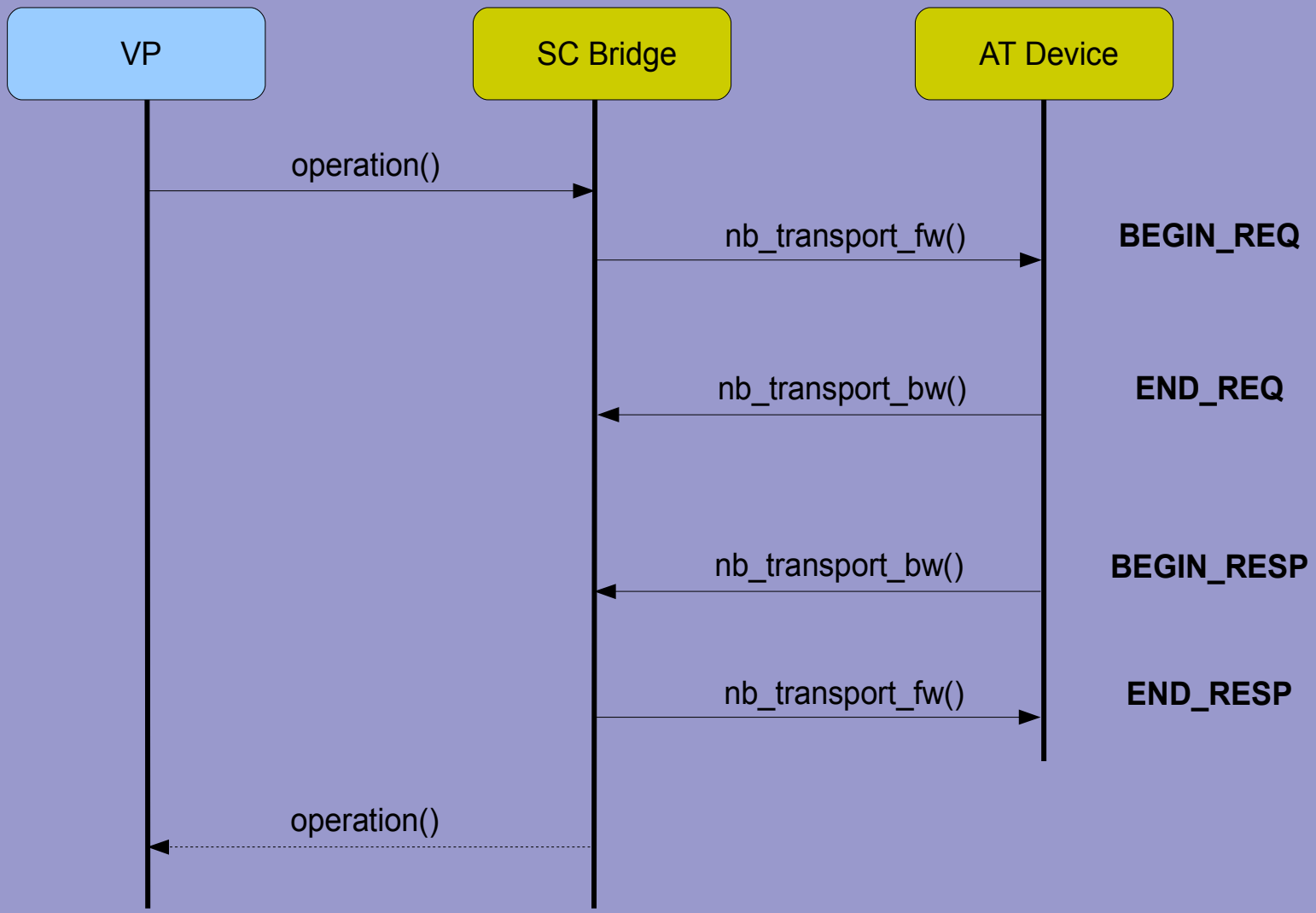
Communication

- QEMU works with zero-delay communication
  - CPU accesses one device and the device responds immediately
- Fit to LT devices
- Need to manage AT devices
  - Special synchronization
  - Finish all protocol phases before return to VP

Communication - LT



Communication - AT

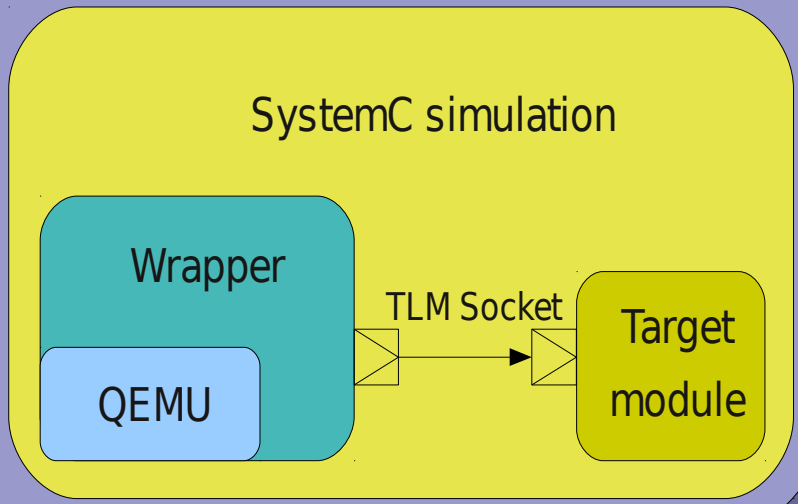


Synchronization

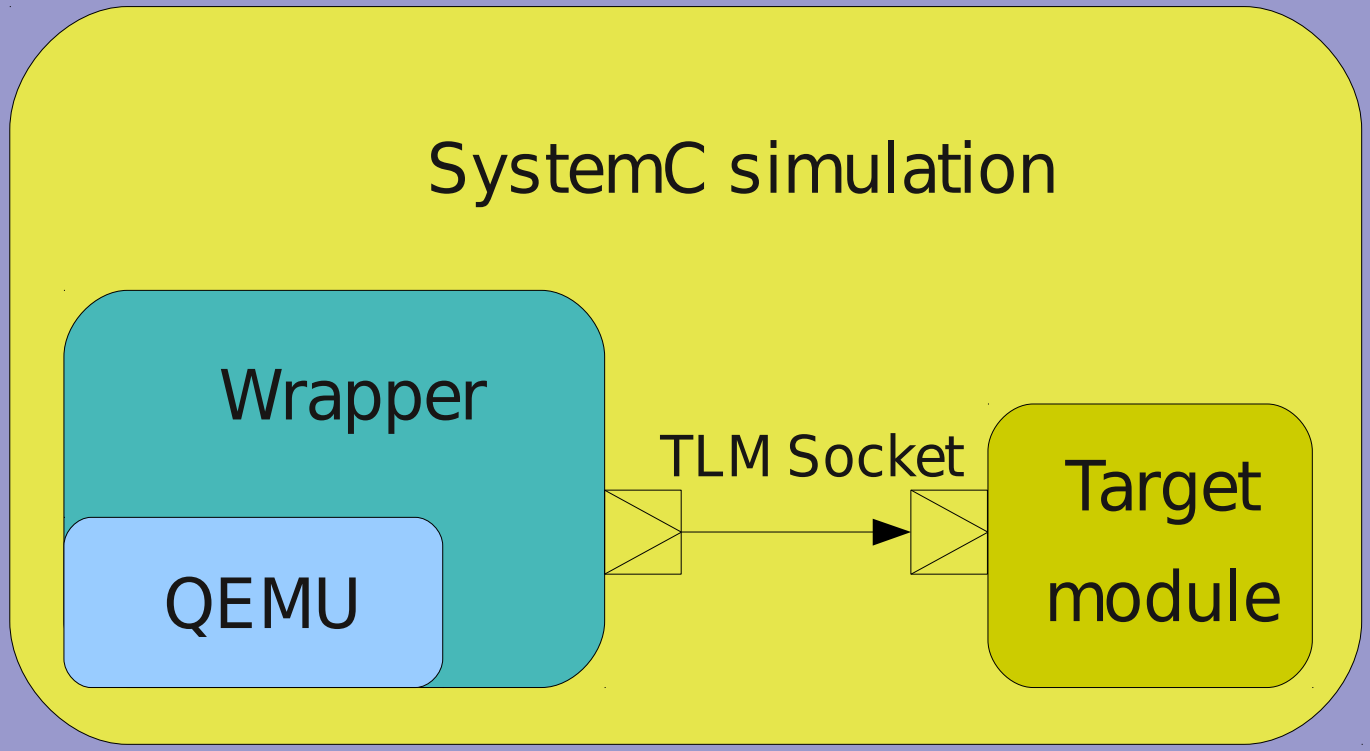
- Manage external I/O from/to SC device
  - Devices use QEMU callbacks for I/O
  - Bridge knows when a I/O is performed
    - Synchronize simulators
- Continue SystemC simulation when
  - VP time arrives to first SC event
  - Every quantum time (TLM-2)
- Advance SystemC time until transaction ends

QBox

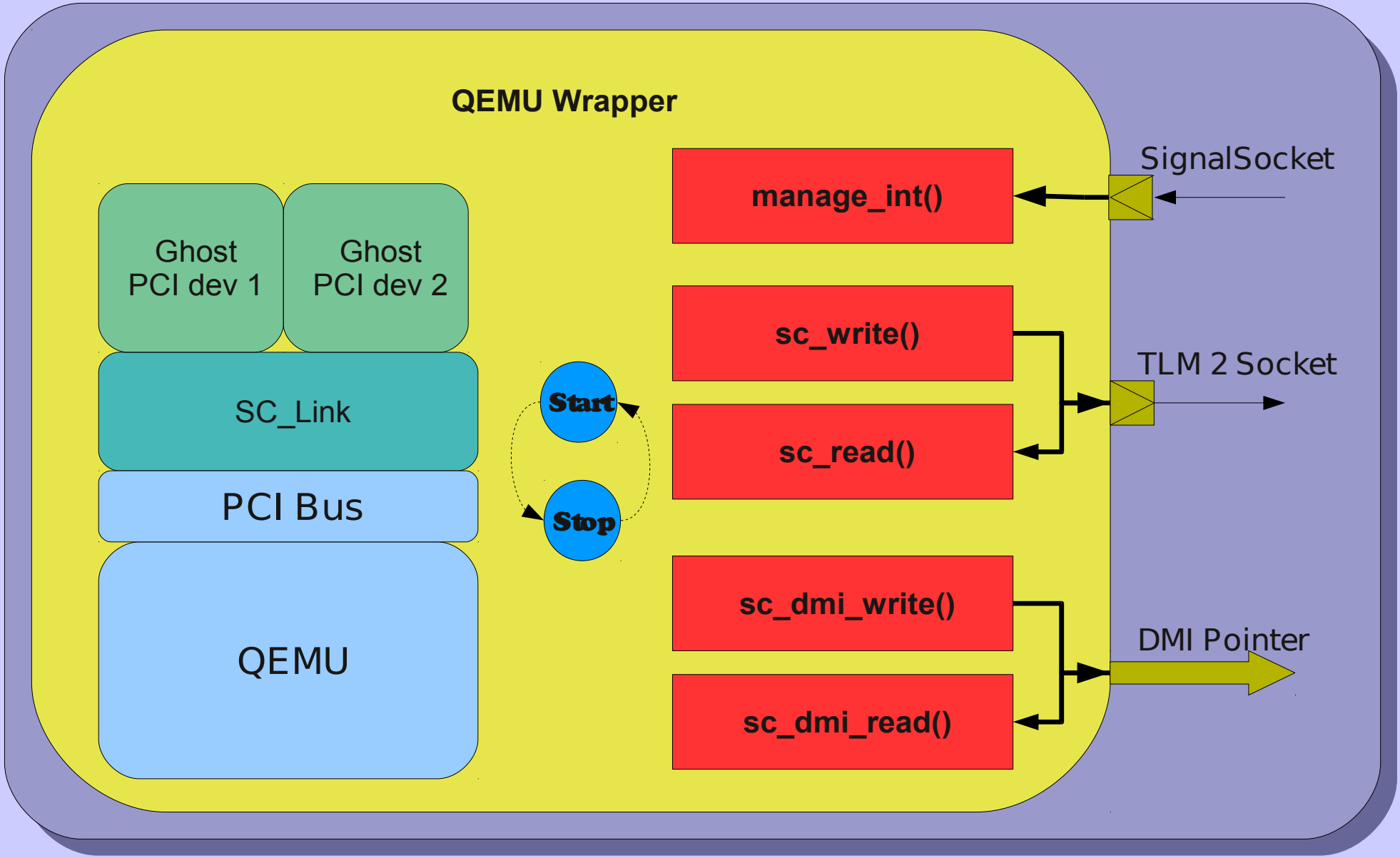
- Change simulation manager
  - QEMU becomes simulator slave
- SystemC manages simulation
- QEMU is a TLM-2 Initiator module
- Easy integration
- Focus on many SystemC models



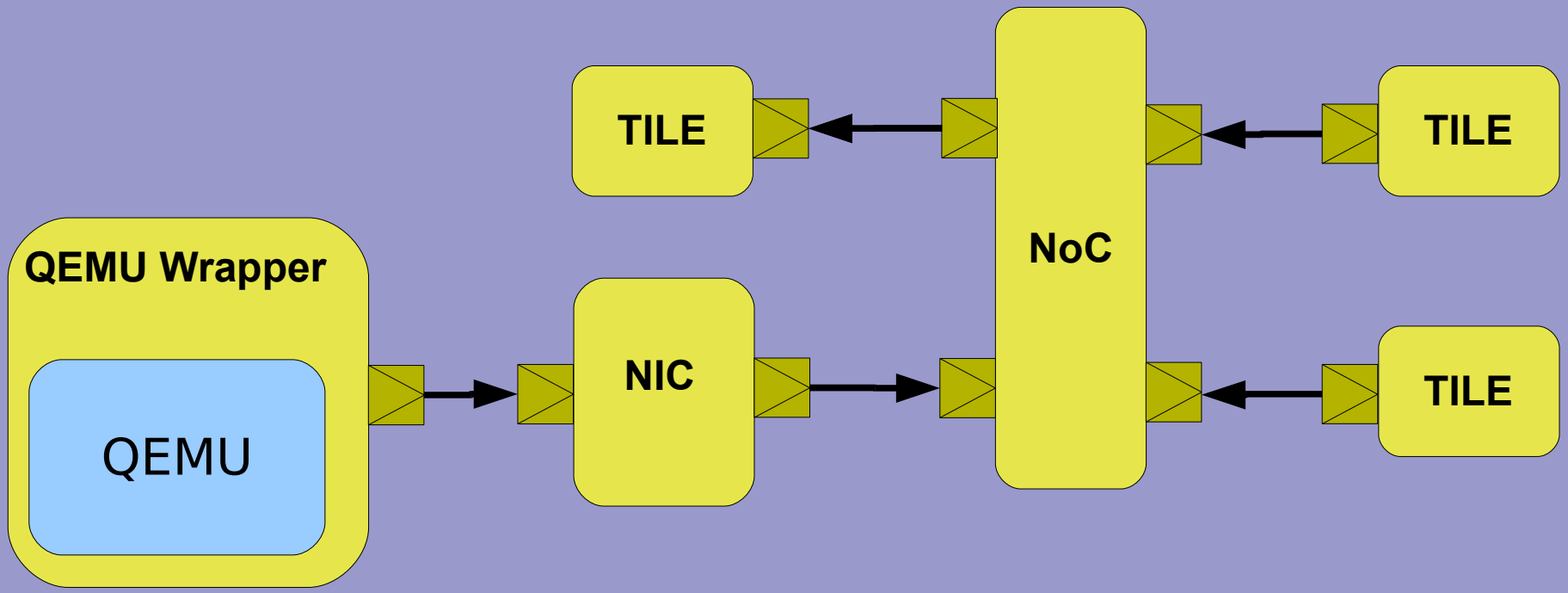
QBox



QBox internal architecture



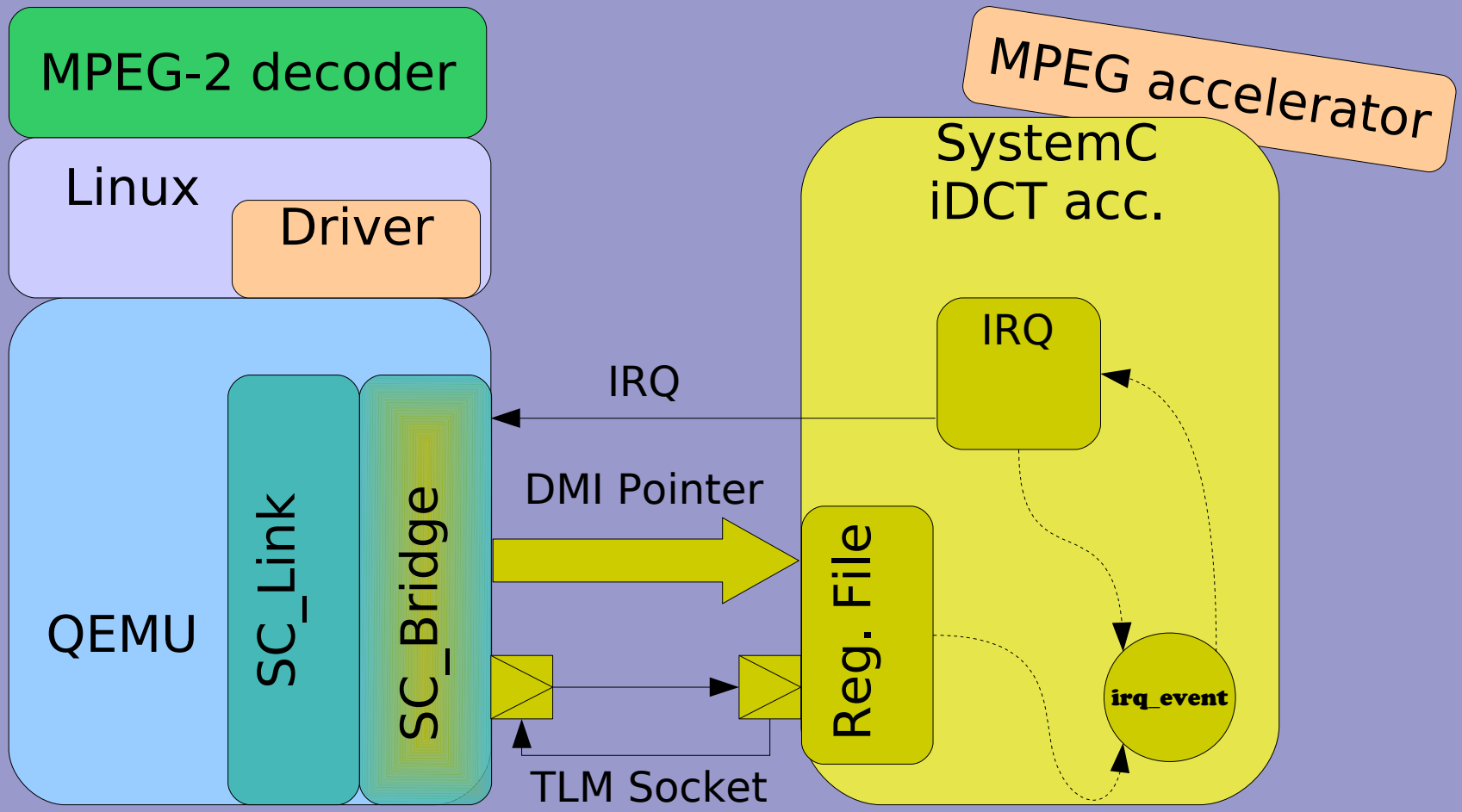
QBox complex example



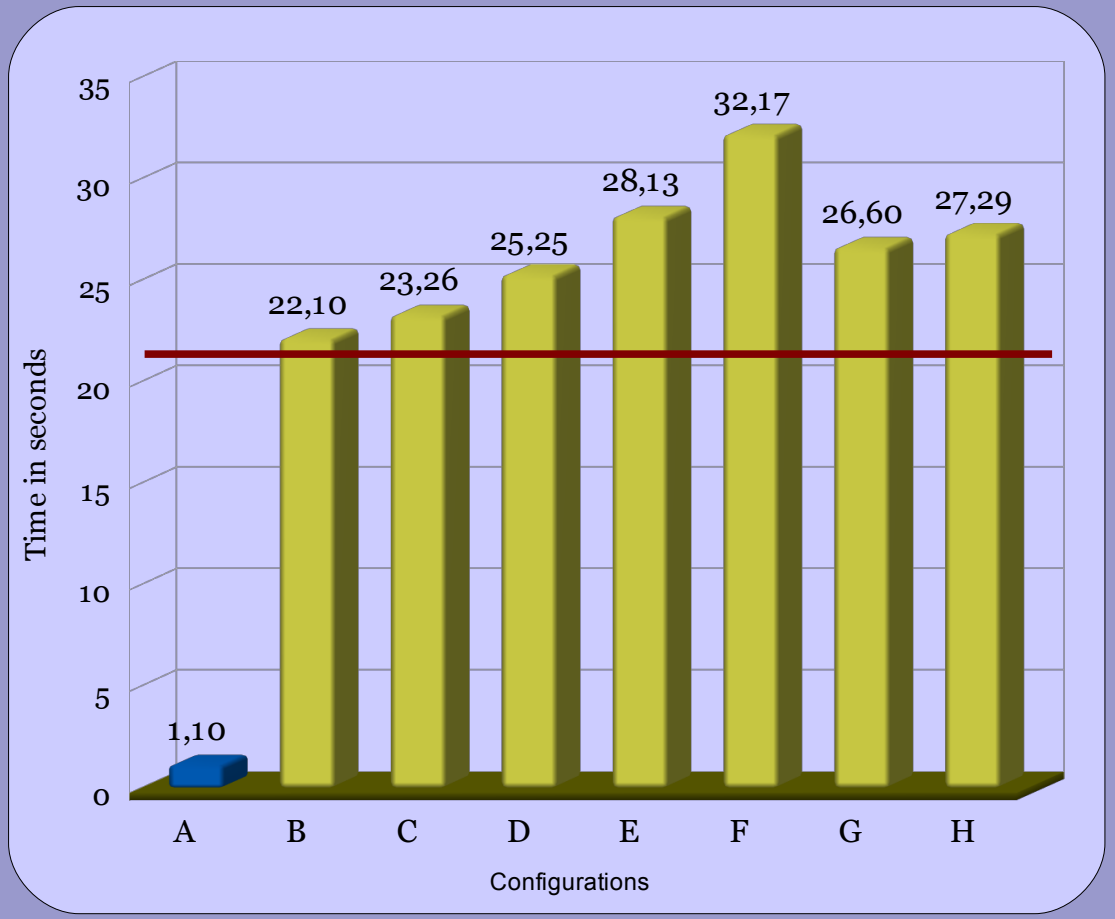
Test & results

- Different tests and examples
  - Validate implementation
  - Extract performance metrics
- Results for performance
  - relatives to same system in native language
    - C in QEMU & Qbox

QEMU-SC Test System

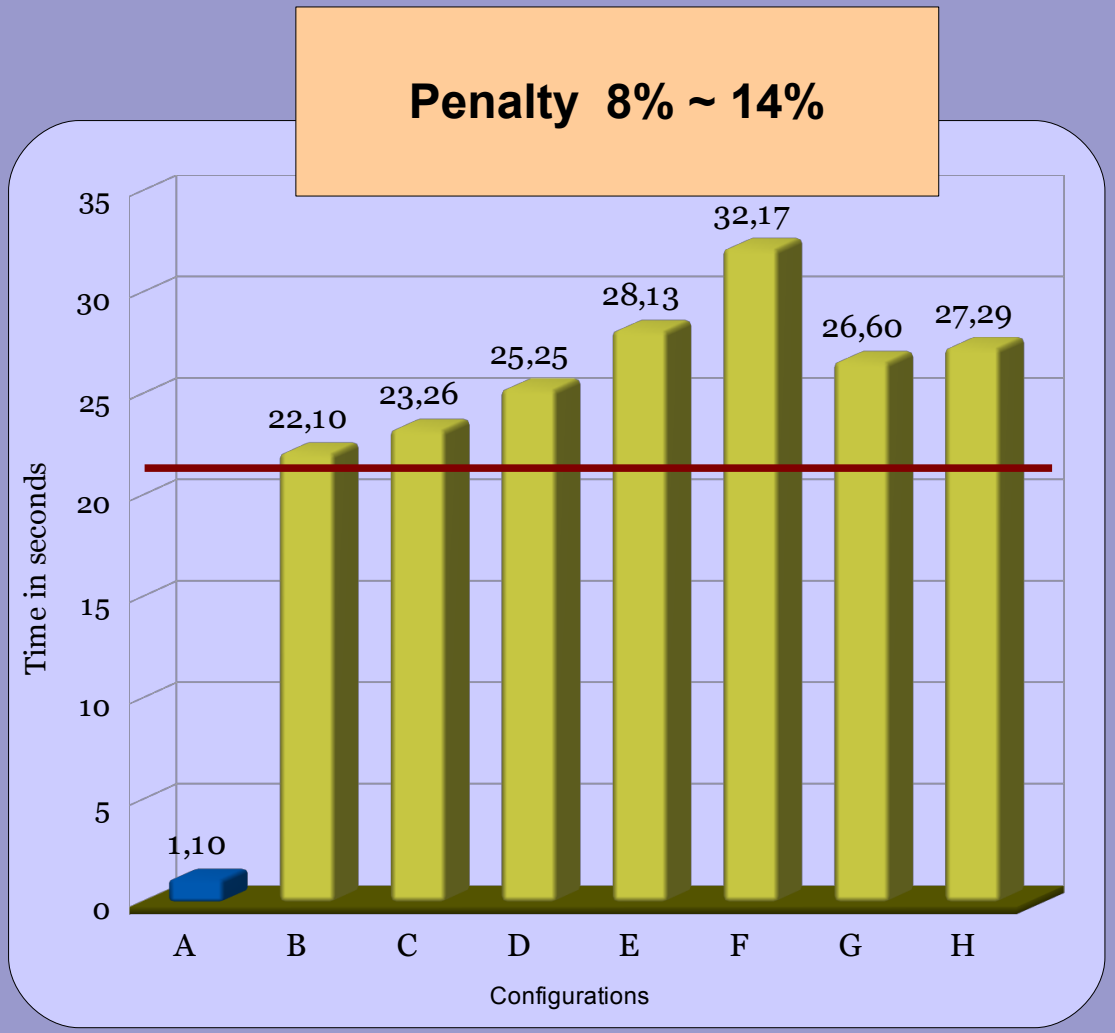


QEMU-SC Results



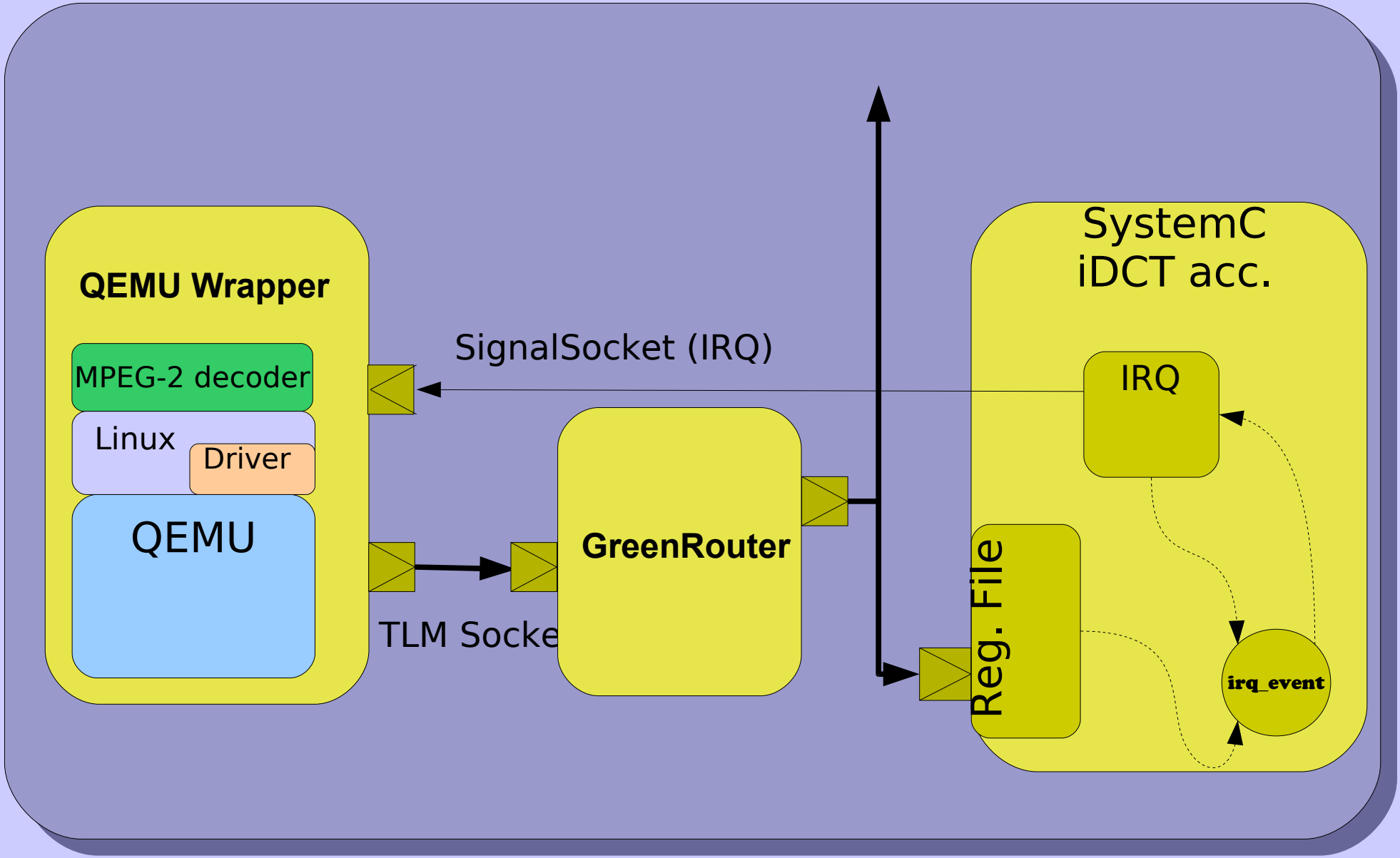
- No acc.
- No acc. w/ drivers
- QEMU style 1 acc.
- SystemC 1 acc.
- QEMU style 2 acc.
- SystemC 2 acc.
- SystemC skel. (1 acc.)
- SystemC skel. (2 acc.)

QEMU-SC Results

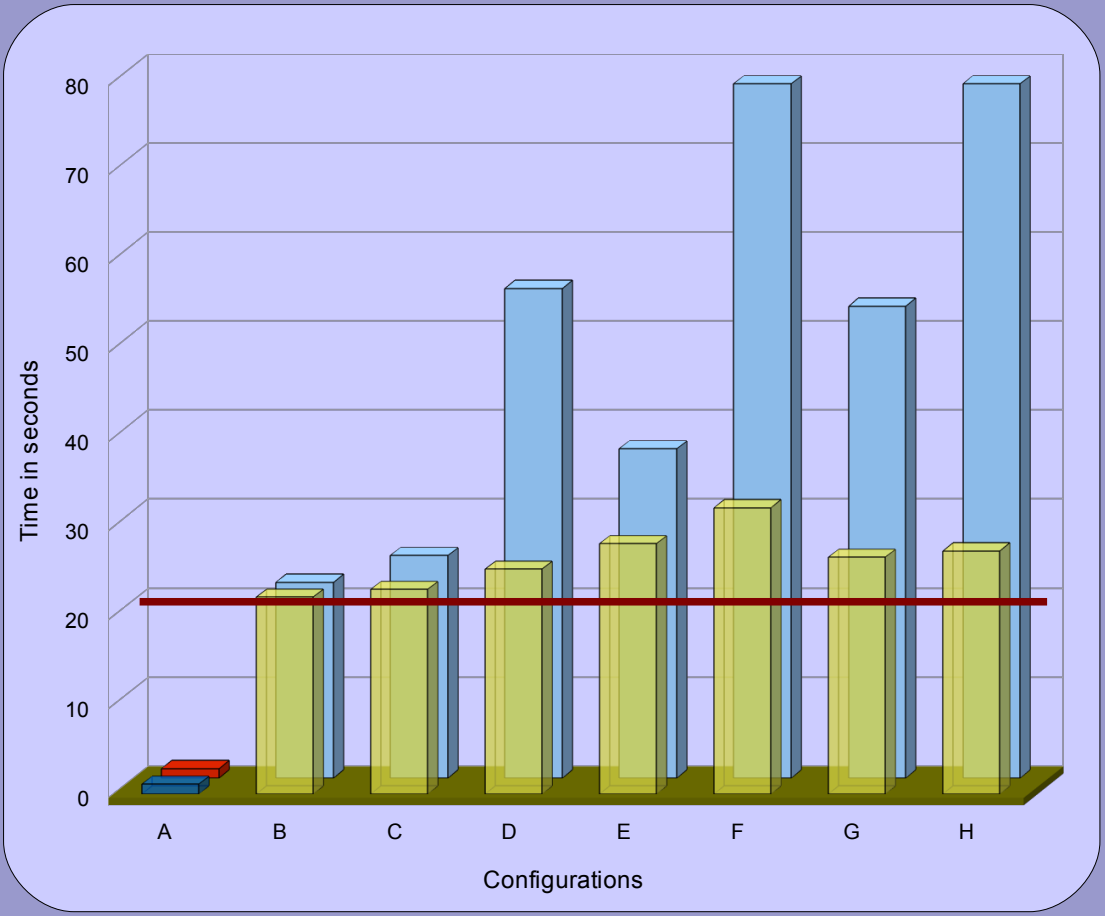


- No acc.
- No acc. w/ drivers
- QEMU style 1 acc.
- SystemC 1 acc.
- QEMU style 2 acc.
- SystemC 2 acc.
- SystemC skel. (1 acc.)
- SystemC skel. (2 acc.)

QBox Test System

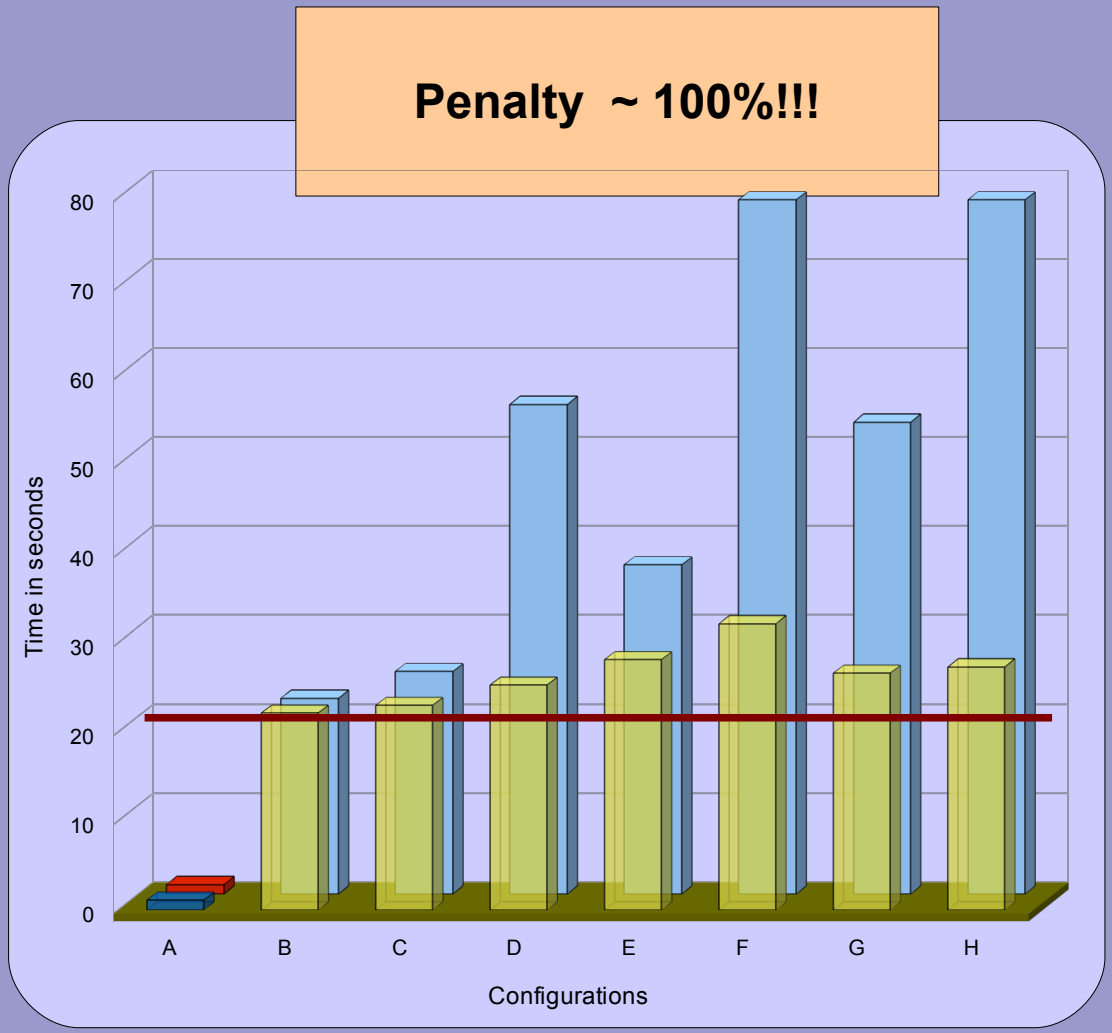


QBox Results



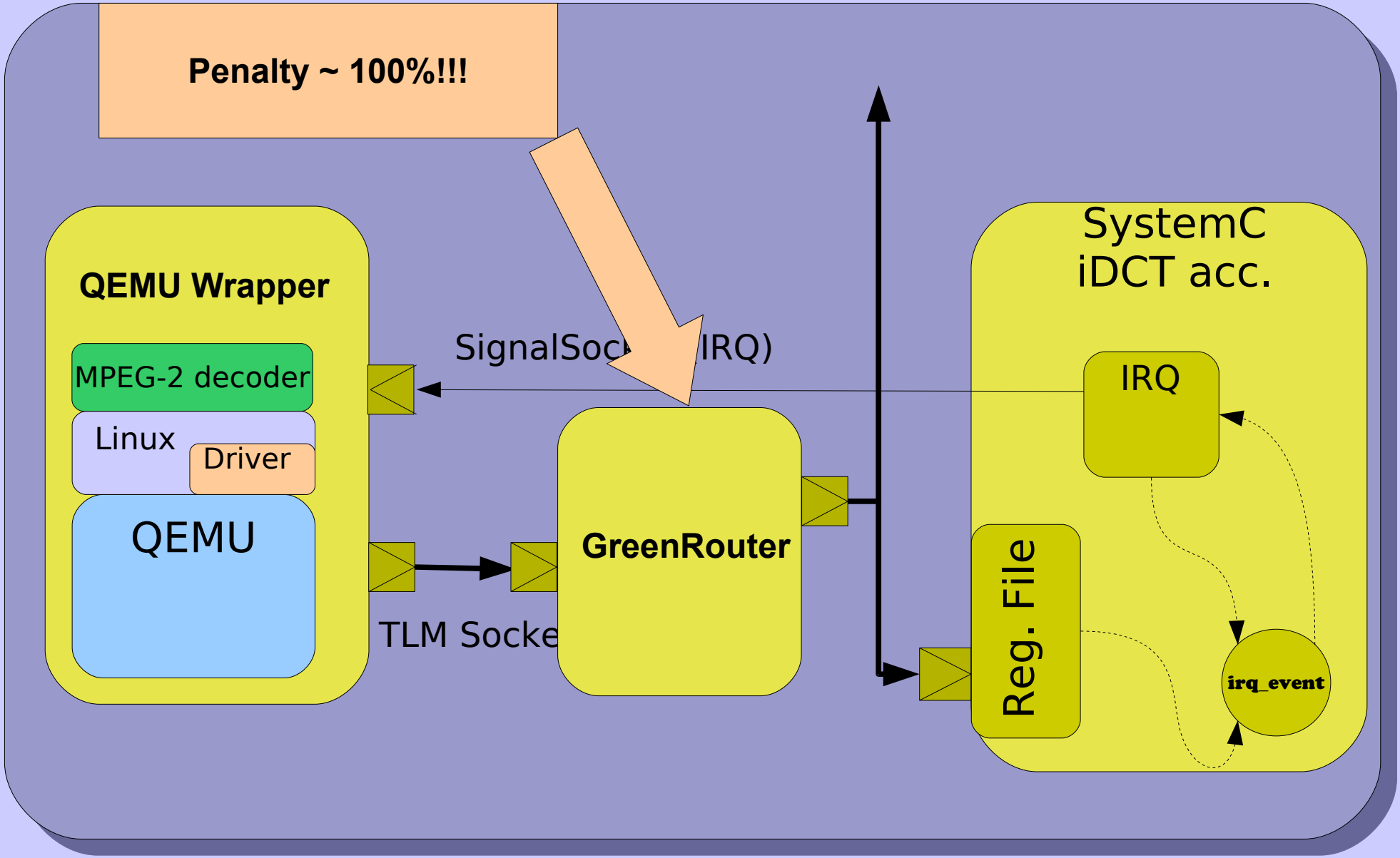
- No acc.
- No acc. w/ drivers
- QEMU style 1 acc.
- SystemC 1 acc.
- QEMU style 2 acc.
- SystemC 2 acc.
- SystemC skel. (1 acc.)
- SystemC skel. (2 acc.)

QBox Results

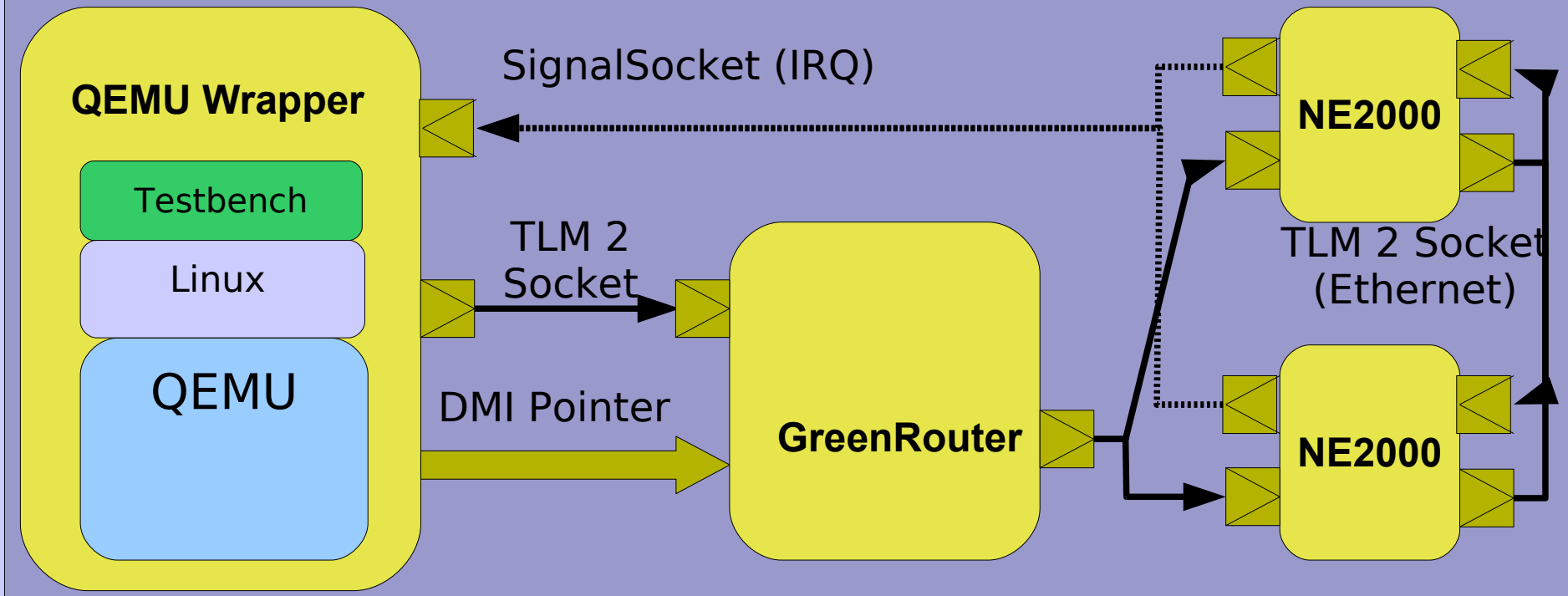


- No acc.
- No acc. w/ drivers
- QEMU style 1 acc.
- SystemC 1 acc.
- QEMU style 2 acc.
- SystemC 2 acc.
- SystemC skel. (1 acc.)
- SystemC skel. (2 acc.)

QBox Test System



Complex QBox system



Conclusions – SystemC

- Joined SystemC to two Virtual Platforms
- Tested two different strategies for joining two simulators
  - QEMU-SC
  - QBox
- Minor performance impact SystemC bridge
- Published as open-sourced projects www.greensocs.com

Conclusions – SystemC

	Simulation Manager	Penalty	System for
QEMU-SC	Yes	10%	SW
QBox	No	25~30%	HW

Future Work

- Automagic configuration of QBox systems
  - Manage map-address in QEMU, Router, BIOS, etc.
- Enhance QEMU time management
  - Hard to measure virtual time in QEMU
- Add multiple instances from QEMU
  - Current QBox library allows one
- Explore QEMU user mode
  - Simplified version, only ISS, run applications

Future Work

- Merge SystemC methods into QEMU kernel
  - Remove OSCI simulator and write an API SystemC <-> QEMU
- Merge QEMU functionality into OSCI kernel
  - Make QEMU a truly SystemC ISS
- Both merges increase simulation speed due to removed synchronization

Proposal

- Join efforts and teams to develop
  - just one QEMU & SystemC virtual platform

Thank you!

Questions?