Reverse Execution for Virtual Platforms using SystemC-TLM-2.0 and Qemu.

Mark Burton, Frédéric Konrad, Màrius Montón.

Presented by David Black.

Reverse Execution

GreenSocs®

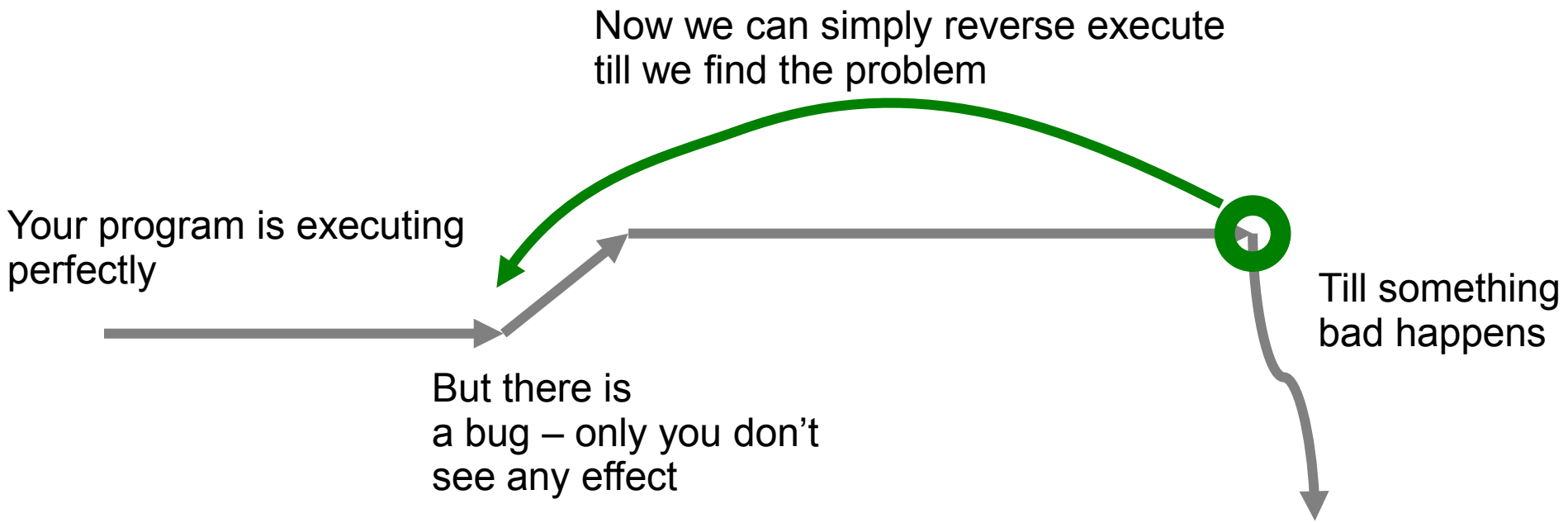
- Reverse execution
  - service provided by some debug environments
  - help catch those really hard bugs
- Simple
  - break on at place where the effect of bug is seen
  - single step backwards until problem found
- Reverse execution involves
  - dynamically restoring checkpoints
  - re-play forward to position just before you started
  - over and over
- To be useful, the mechanism need to be fast!

Reverse Execution with SystemC & QEMU

GreenSocs

- QEMU – Quick EMUlator
  - Performs processor virtualization
  - Dynamic binary translation – aka JIT (just in time)
  - Free and open-source software product
- SystemC – modeling technology for hardware
  - Interfaces to QEMU using TLM 2.0 – loosely timed
- GreenSoCs project Cexe
  - What else would you call Exec in reverse?
  - Checkpointing and reverse execution to both SystemC and Qemu

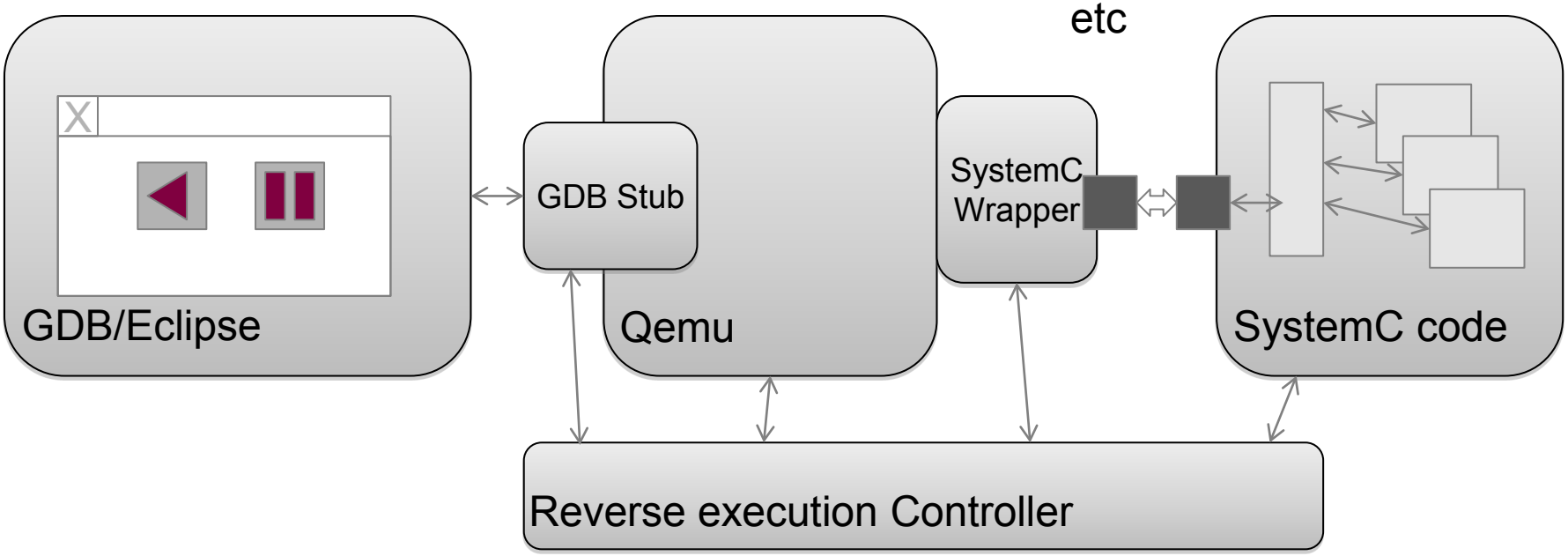
Debugging with reverse execution



Overview of Reverse Execution

The user will use Eclipse/GDB which already supports reverse execution to control the system.

TLM 2.0
Sockets
Supporting:
DMA (to/from Qemu memory),
DMI,
Quantum time keepers
IRQ's,
etc



Checkpointing for SystemC

- For SystemC there are two approaches that have been used:
  - Save and restore the entire 'machine' state.
    - This is very heavy and slow.
  - Save just the state variables from the SystemC model.
    - This is very fast and efficient!

Guess which one GreenSoCs chose!

Checkpointing for SystemC

GreenSocs

- User code needs to give its internal state
- Implicit
  - We support the use of CCI Parameters as state variables throughout the design.
  - We then collect checkpoint data directly through the CCI interface.
    - This can be seen now using the gs_param<> from GreenSocs
- Explicit
  - Designer use special callbacks to publish state variables
    - do_checkpoint()
    - do_restore()

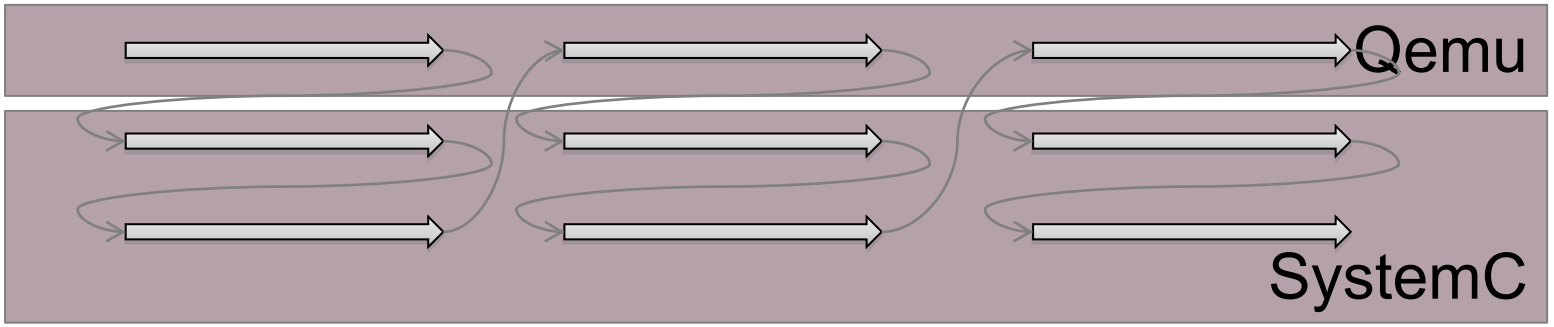
Bottom line – USE THE NEW CCI STANDARD!

Qemu SystemC integration

- For checkpointing and reverse execution to work, the integration between Qemu and SystemC has to be relatively tight – but for performance, it better not be too tight !

- We have ensured the wrapper works with the notion of Quantum time to ensure the best performance from models.

- Ideally, each 'master' in the system (Qemu or SystemC) executes for 1 quantum. Time moves to the next quantum only once all the masters have executed.



Quantum timing for attached devices

GreenSocs®

- When a SystemC model posts an event to a Qemu processor:
- Sometimes the event is destined for a different master:
  - The SystemC model should take care to post the event at the next quantum boundry.
- Sometimes the event is destined for This master:
  - The SystemC model may take care to post the event at the next quantum bountry.
- Events scheduled for a time within the next quantum (e.g. a fast interrupt) will be caught by the wrapper and scheduled correctly for Qemu.

Play that again Sam!

Simulated time seems to move backwards!

- Reverse execution works by re-running the same code again, and stopping at the right place.
- This means 2 things:
  - You must be able to re-run exactly the same way (Deterministic)
  - You must be able to "stop in the right place".
- For Qemu there are problems with both !

Play that again Sam!

Simulated time seems to move backwards!

- Fortunately…
- For determinism Qemu has the ability to record IO events
  - Put in place to assist in “live migration”
- We can re-use this mechanism to solve the problems of determinism.

Bleeding edge

- We have a working demo….but
- Instruction counting – we count instructions at the translation block boundary. That works… it's not trivial.
- Aligning timers – that's hard because Qemu internal timers are somewhat 'approximate' – working on this now…
- Other IO needs to be recorded and played back, we believe this might help with the the timers too.
- Active discussion on the Qemu email lists…

Result:

- Right now
  - Demonstrator capable of hitting a breakpoint
  - and taking a step backwards
- To do:
  - Not optimized for speed
  - Can not yet handle IO determinism
- Status:
  - This is work in progress
  - Have not solved all the problems yet!
- More information can be found on the project page at http://projects.greensocs.com/projects/cexe