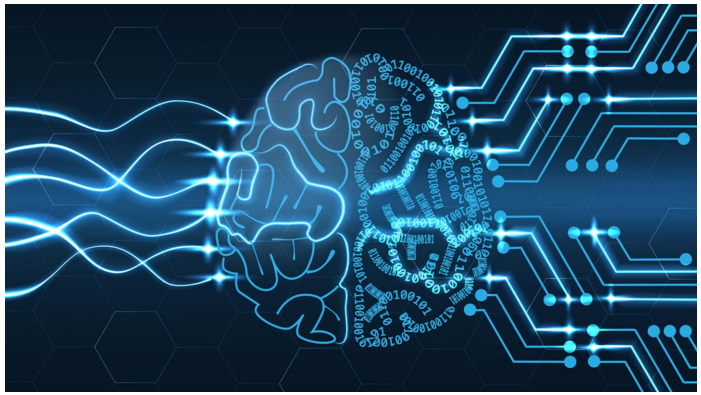
GreenSocs®

GreenSocs®

NVIDIA®

Open Source Virtual Platforms for SW Prototyping on FPGA

Mark Burton



- Nvidia has a Deep Learning Accelerator (called NVDLA)

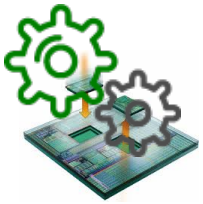
The NVIDIA Deep Learning Accelerator (NVDLA) is a free and open architecture that promotes a standard way to design deep learning inference accelerators. With its modular architecture, NVDLA is scalable, highly configurable, and designed to simplify integration and portability. The hardware supports a wide range of IoT devices. Delivered as an open source project under the NVIDIA Open NVDLA License, all of the software, hardware, and documentation will be available on GitHub. Contributions are welcome

- Nvidia also has a ‘c’ model of the DLA architecture (could be used as a systemc/tlm model)

GreenSocs

https://www.youtube.com/watch?v=3LVeEjsn8Ts

Goals



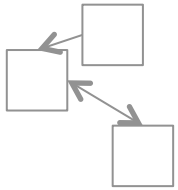
- Bring HW and SW together
- Minimize time to re-spin
  - (change in HW/change in SW)

- Enable simulation to be used by anybody
- Make it easy and quick to use

- Make the simulation FAST
- Enable S/W development

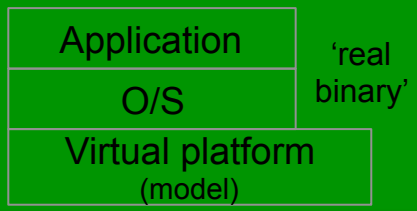
GreenSocs

(Para-)Virtualization



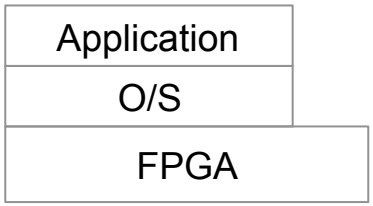
Algorithm execution
Or full system virtualization

Virtual Platform Virtualization



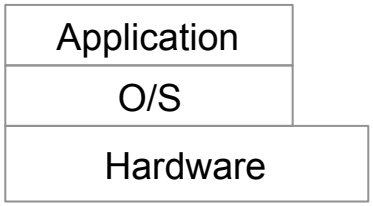
Full binary execution on virtual platform (model)

Emulation



Full binary execution on REAL platform (FPGA)

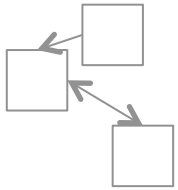
Hardware



Full binary execution on Final Hardware

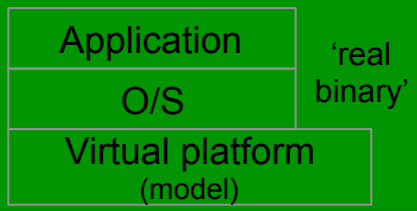
GreenSocs

(Para-)Virtualization



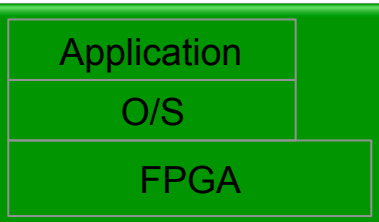
Algorithm execution
Or full system virtualization

Virtual Platform Virtualization



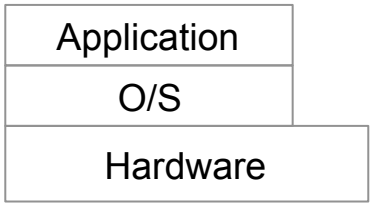
Full binary execution
on virtual
platform (model)

Emulation



Full binary execution
on REAL
platform (FPGA)

Hardware



Full binary execution
on
Final Hardware

Virtual Platform Standard is SystemC TLM-2.0 IEEE 1666

- Open Source Simulator available for download from **Accellera.org**

Corporate members 2016

- **GreenSocs** technology at the heart of TLM-2.0 standard.
- **All GreenSocs** interfaces use TLM-2.0
- **GreenSocs** helping Accellera forge a new Model to tool standard.
  - Preview available in GreenConfig.

- Our solutions are tool independent, and work with **all vendors**.

Qemu: Our Preferred source of CPU models

GreenSocs

- Qemu is the defacto standard Virtualizer.
- Free and Open Source.
- It is over 10 years old

18	1100	43000	1000	989,863
Architectures	CPU's	Commits	Contributors	Lines of code

- GreenSocs is a key contributor:
  Reverse execution and Multi-Core TCG Kernel.

- Regular committers from many organizations

Existing Model database overview:

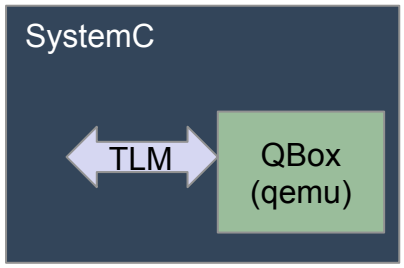
CPU Family coverage:

	X86	ARM	MIPS	Alpha	PowerPC	SPARC	Micro-blaze	Cold-fire	Cris	SH4	Xtensa
Fast SW dev model (LT)	✔	✔	✔		✔	✔	✔	✔	✔	✔	✔
Cycle Accurate HW dev model (AT)	✔	✔		✔		✔					

Full list (of several hundred) available on GreenSocs.com

QBox

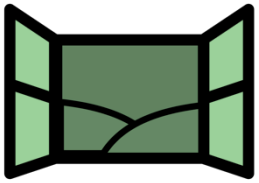
- Wraps up Qemu in a TLM2-0 API such that it can be used in standard SystemC



- QEMU is a generic and open source virtualizer – it covers almost all CPU architectures and achieves extremely high performance.

Qbox Syncronisation options

- Real Time
  - Each simulator runs as close to real time as possible.
  - Can be simple “run as fast as you can”, no sync.



- Windowed
  - Each simulator is allowed to run within a window, but if it reaches the end, it must stop and wait
  - The window will automatically extend as simulators run.
  - (Windowed ‘behind’ to keep SystemC behind and the tlm delta time positive)

- Deterministic/single threaded
  - Each simulator runs in turn.
  - Pseudo random ordering to ‘catch’ S/W bugs.
    - (The advantage of a model…)

GreenSocs is the XILINX ALL PROGRAMMABLE partner upstreaming their device models

Clock framework

- Enable the correct timing for events across the full Zynq device.

Large packet DMA framework

- Significantly increase the speed of DMA activity in the simulated device.

Fault Injection

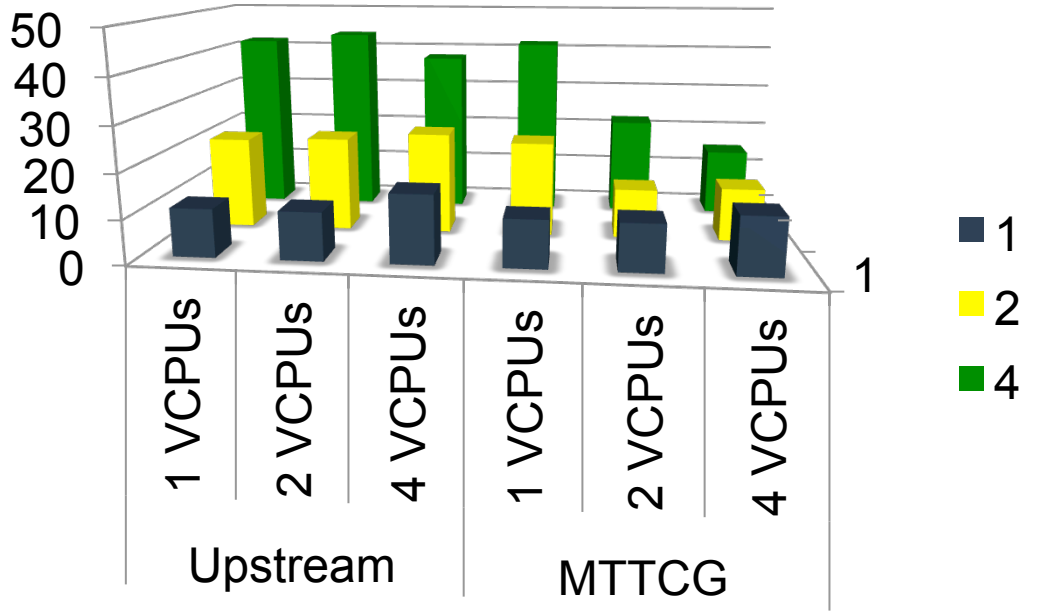
- Model fault injection in a convenient and scriptable way, to enable safety and test features to be validated.

Safety and Test Library extensions to devices

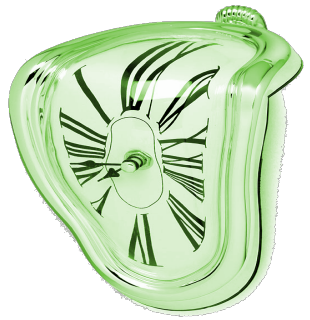
- Model the suite of devices in the Zynq that can be self tested.

MULTI Thread Qemu

- A massive speed improvement for Qemu to take advantage of multi-core hosts



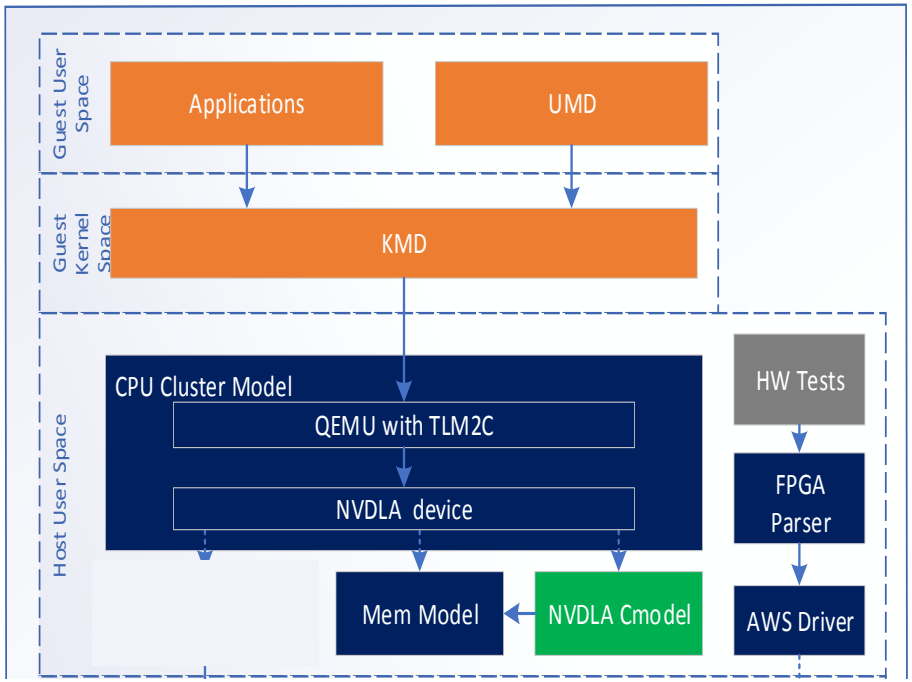
- **NON-Deterministic Reverse Execution**



  - Ability to debug from an error backwards, irrespective of input stimulus
  - Supporting LAUTERBACH DEVELOPMENT TOOLS
  - No H/W required, No 'JTAG collector' limit.

- **Cache modeling**
  - Cache Coherency performance estimation
  - Cache flushing S/W checking

- User Application and user level device code
- Kernel and kernel modules
- Virtual Platform model,
- Based on QEMU and SystemC
- 'C' model for NVDLA device itself

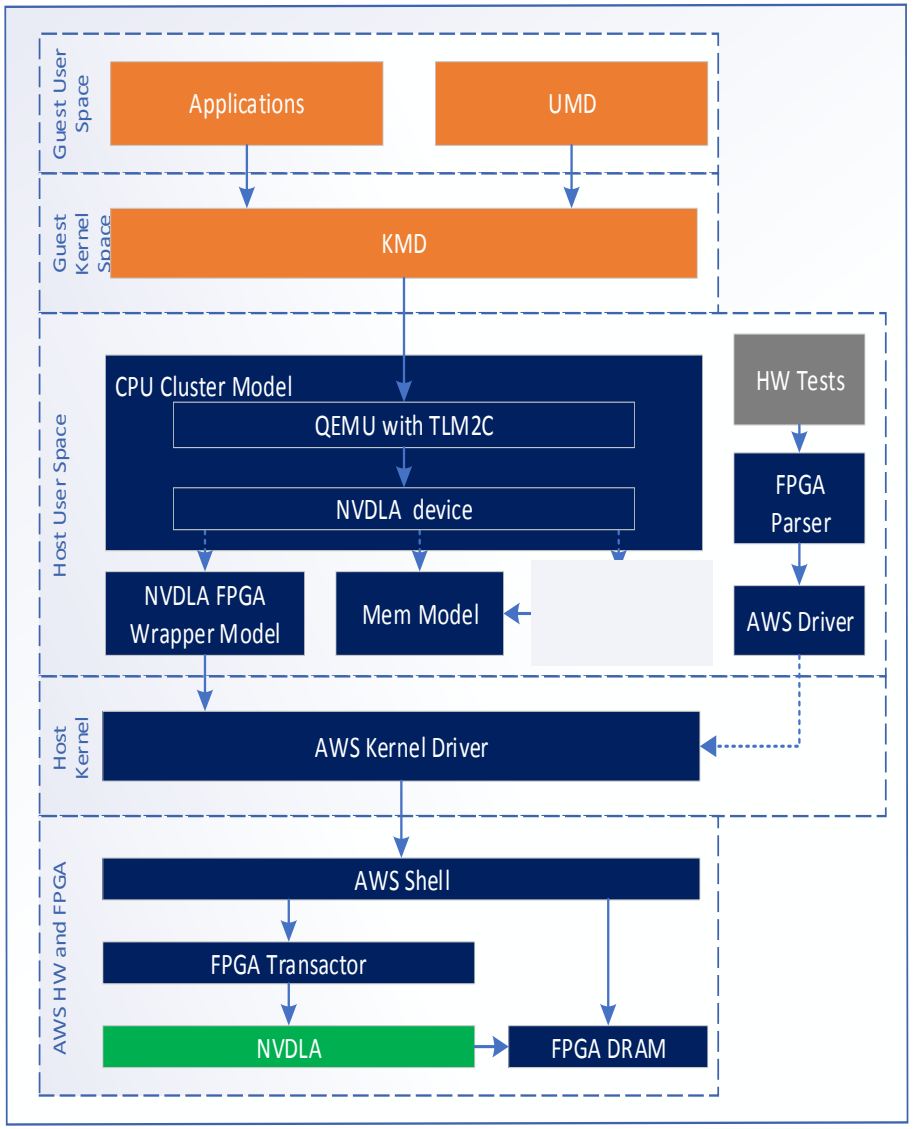


- Simulation speed··· the NVDLA – Accelerator – is modelled on the host, so it will not ‘accelerate’.

- Changes to the core NVDLA architecture require changes to the model.

Adding FPGA

- User Application and user level device code
- Kernel and kernel modules
- Virtual Platform model, with FPGA wrapper
- AWS framework
- NVDLA FPGA hardware module
- Runs at full speed!

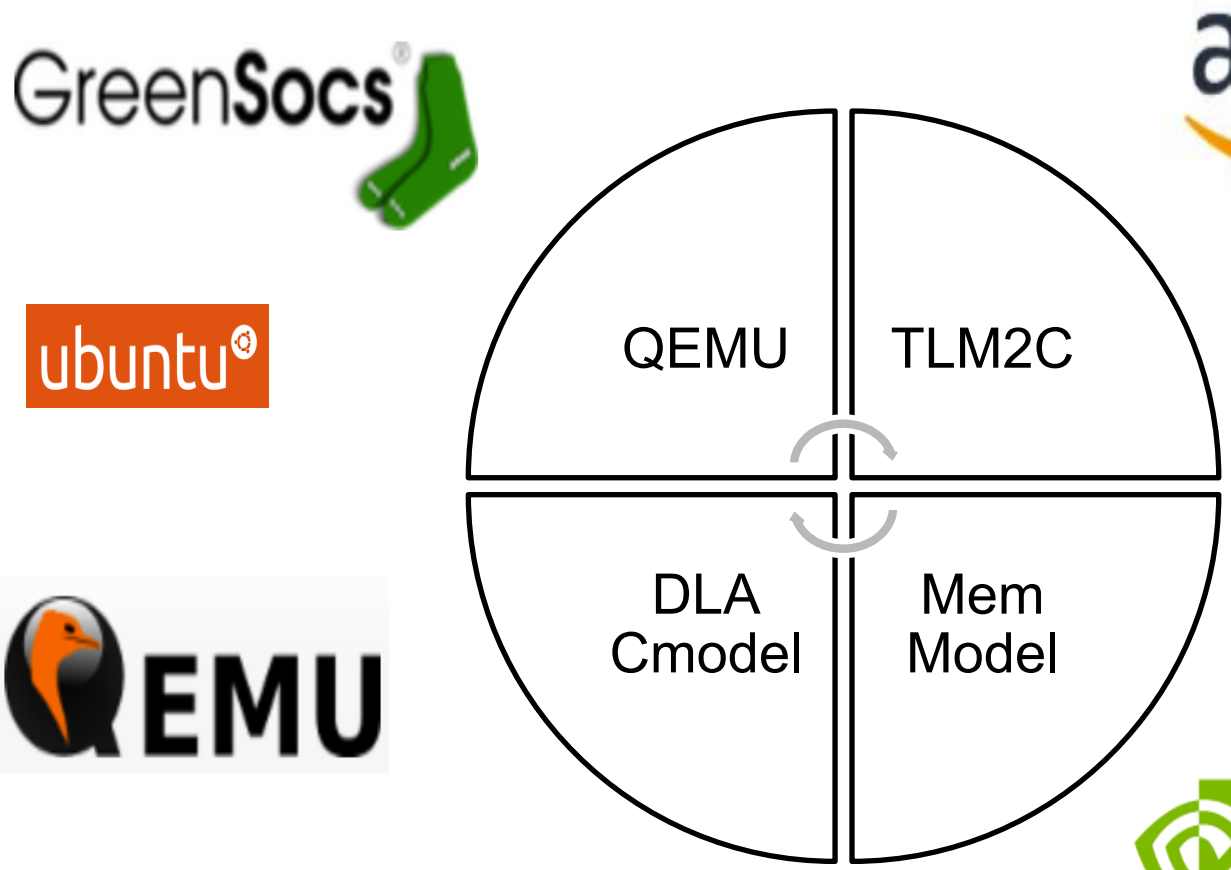


- SW on NVDLA C-Model
  - Anybody can download packaged Docker release
  - Configurable – build time ½ hour.
  - FAST TO SET UP.

- SW on FPGA with NVDLA RTL
  - Anybody can run AWS env with pre-packages AMI and AFI
  - With AWS setup, easy to alter both FPGA images and associated drivers. (e.g. less than a day).
  - FAST TO RUN.

Both available from nvdla.org

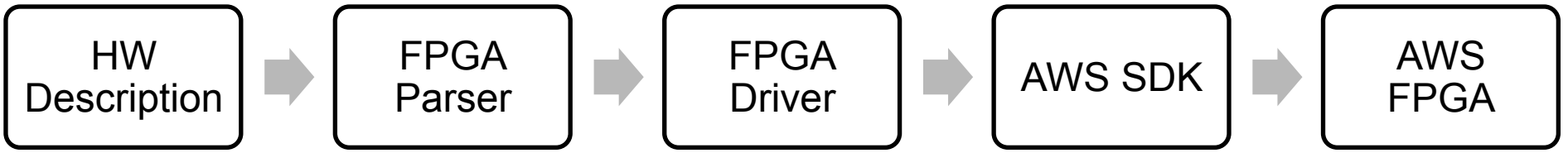
All the components…



HW Test on FPGA

GreenSocs

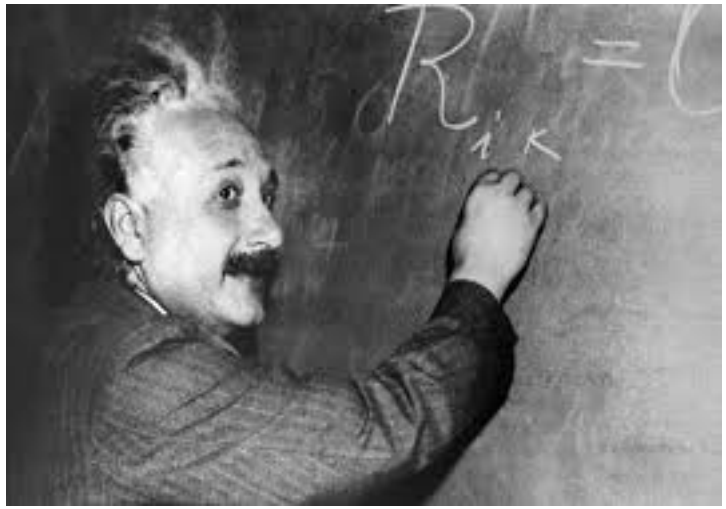
- Why we need HW tests on FPGA
  - To guarantee the quality of FPGA release
  - To identify corner case and issues in RTL



- Based on SW on Cmodel
- Replace all Cmodels (NVDLA, Mem model) with FPGA wrapper
- Full user code executable on combined QEMU + FPGA model

Generalisation

- Making this ‘generally’ applicable requires more work ☹



- Enable any architecture to be modeled in a ‘cloud’ (public/ private), off-loading onto FPGA when required/appropriate.
- Enable ‘Virtulization’ when host/ guest match.

Future Possibilities

- NVDLA Performance Model integration for Performance evaluation
- More AWS FPGA images release for different NVDLA configuration
- Enable RISCV in Virtual Platform
- ARM Project Trillium
- SiFive

GreenSocs®

NVDLA page http://nvdla.org/

OpenVP Doc http://nvdla.org/contents.html

OpenVP Github page
https://github.com/nvdla/vp
https://github.com/nvdla/vp_awsfpga

www.greensocs.com
mark@greensocs.com