Evariste II

Modular hardware system for fair TRNG benchmarking

Viktor Fischer
fischer@univ-st-etienne.fr
Laboratoire Hubert Curien
UMR 5516 CNRS Université Jean Monnet, Saint-Etienne, France

TRNGs and cryptography

- True random number generators (TRNGs)
  - Essential for security in cryptography
    - Generate confidential keys, nonces, random masks, padding values, …
  - Use physical phenomena to generate random numbers

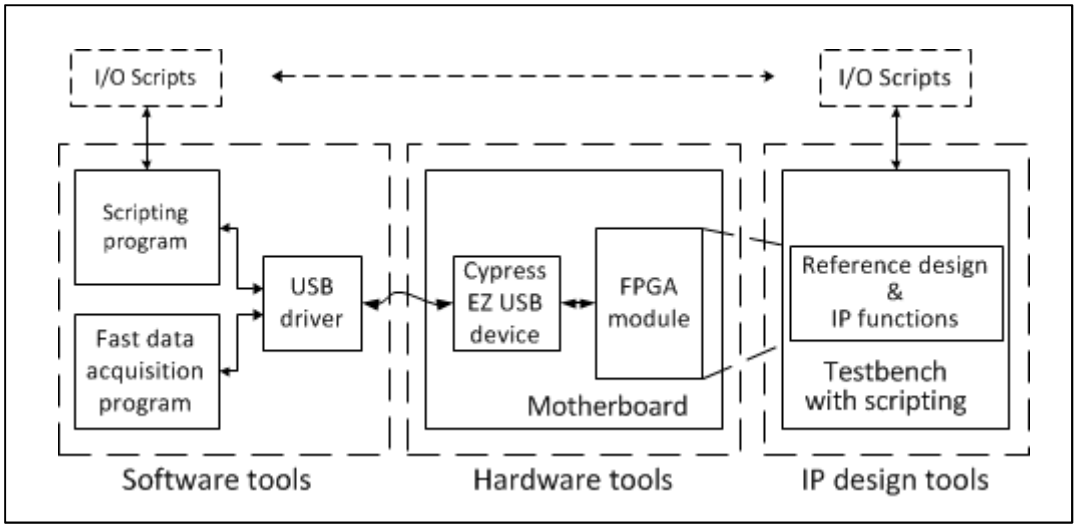
- Security
  - Good statistical characteristics – any weakness could be used for attacking cryptographic system
  - Unpredictability – preceding or new numbers (e.g. keys) cannot be guessed
  - Robustness – output cannot be manipulated

Motivation

- Behavior of the TRNG depends on the underlying hardware and operating environment
  - Device technology (inherent noise)
  - PCB (ground quality, low resistance, no loops)
  - Power supplies and filters (low noise)
  - Electromagnetic interferencies
- Objective: fair TRNG comparison and benchmarking
  - Compare some TRNG principle in various technologies
  - Compare various TRNG principles in the same conditions (device, PCB, power supplies, …)

Introduction

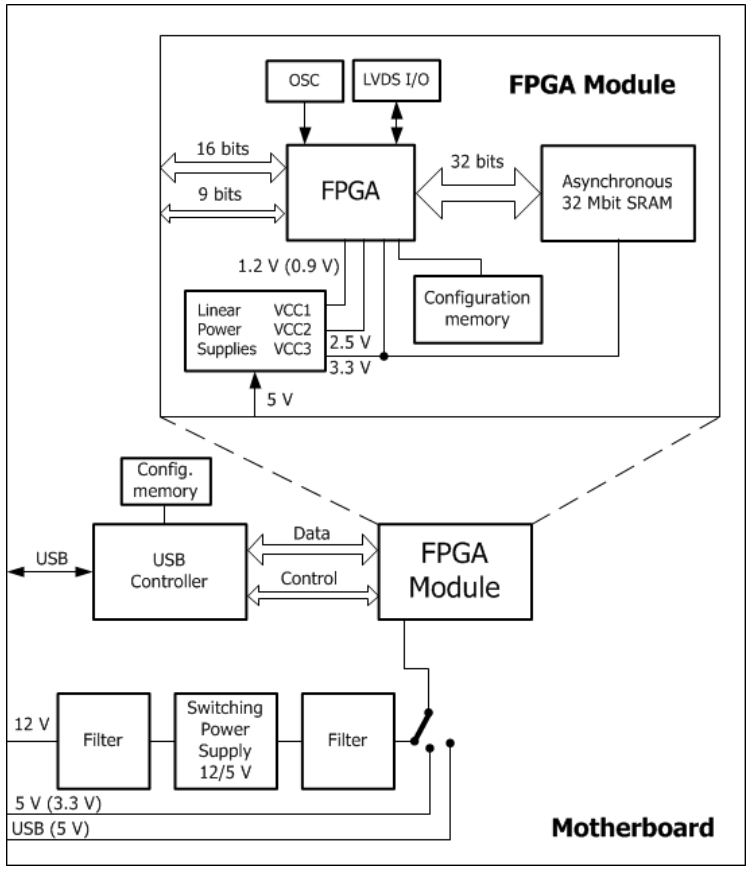
- The Evariste II toolkit – HW + two open source tools



  - The hardware/software design and verification is simplified by a scripting tool: the same scripts are used by the software interpreter and the VHDL testbench
  - Verified commands can be used in the final (fast) software version (the hardware remains the same)

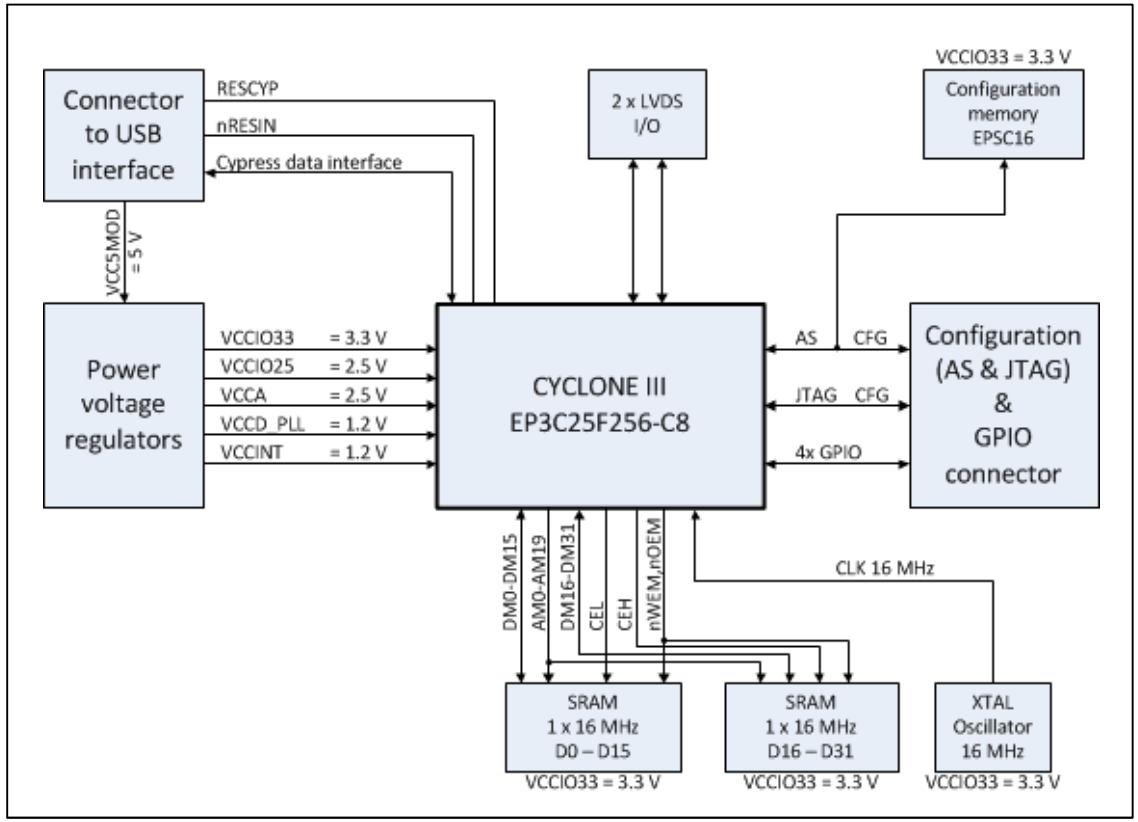
Hardware

- Motherboard and 5 FPGA modules
  - Altera Cyclone III
  - Altera Arria II
  - Xilinx Spartan 3
  - Xilinx Virtex 5
  - Microsemi Fusion



Hardware

- All FPGA modules have identical topology
  - Example: Altera Cyclone III

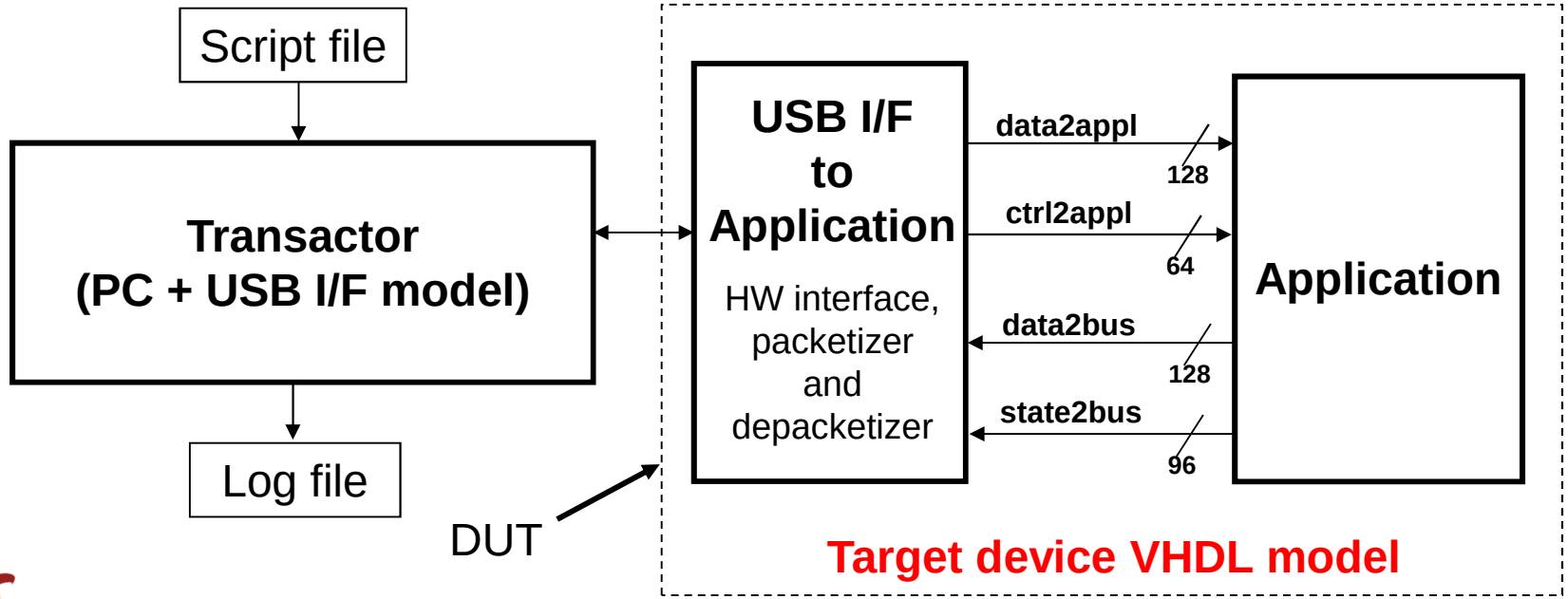


Scripting tools

- The hardware communicates with the PC via USB using packets
- Scripting tools enable efficient and flexible control of testbenches and hardware tests during the design
- Sending (in packets)
  - 64-bit control words
  - Up to 64k 128-bit data blocks
- Receiving
  - 96-bit state words
  - Up to 64k 128-bit data blocks

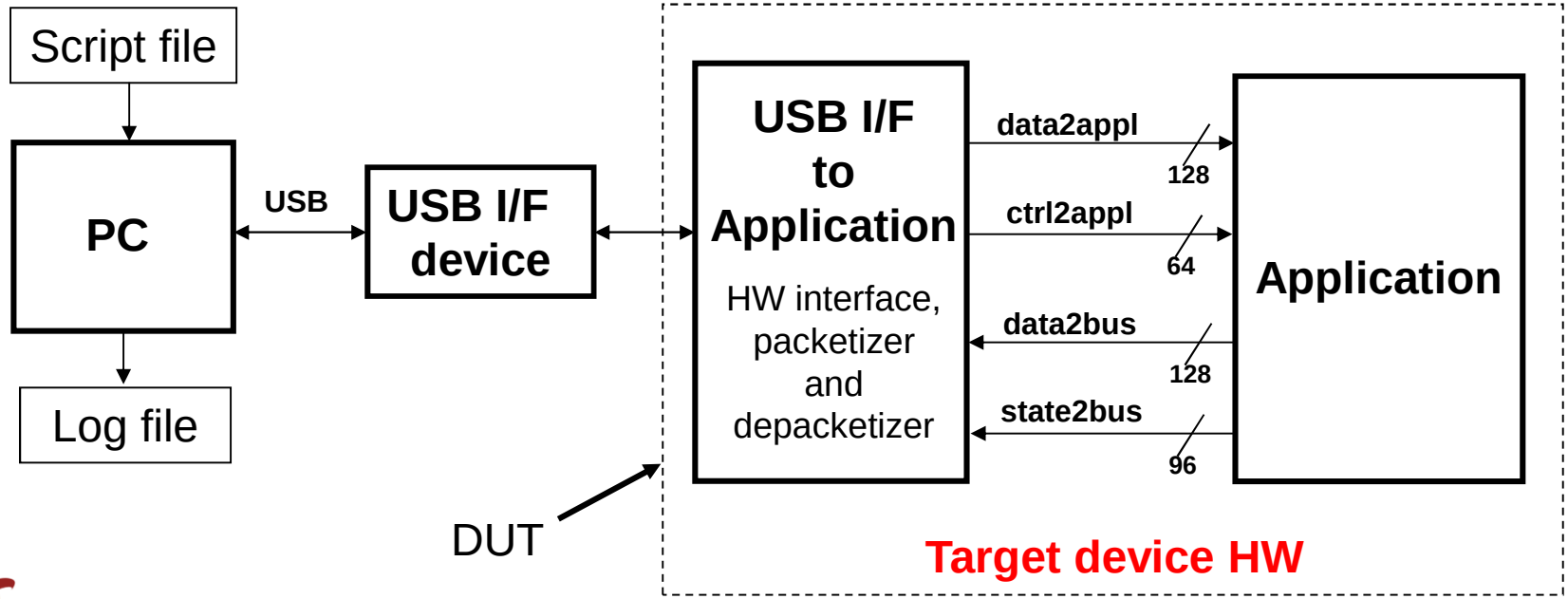
VHDL testbench transactor containing the script file interpreter

- The design under test (DUT) is accessed during the simulation using the VHDL transactor, which includes the script file interpreter



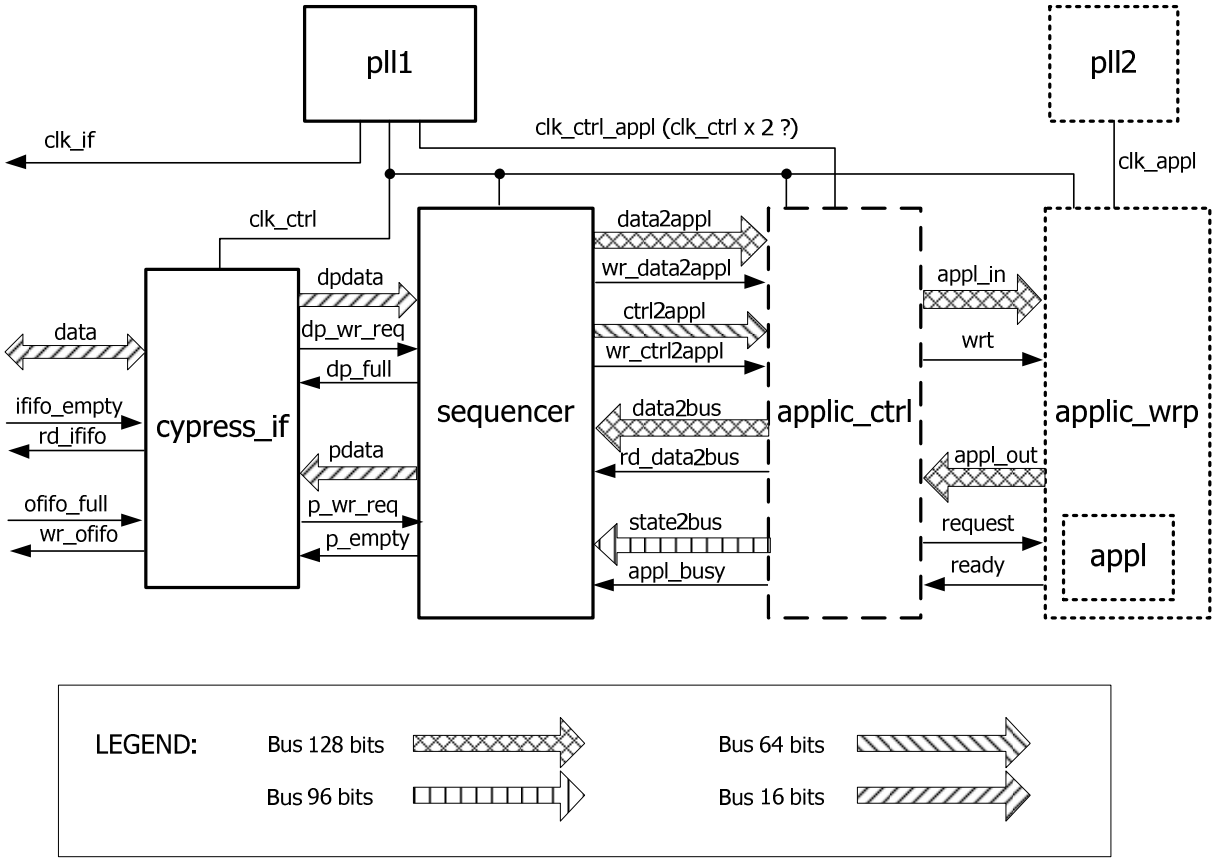
Script file interpreter software

- The hardware is accessed during the tests using the *script.exe* software interpreter
- The same scripts are used both in simulations and tests



Hardware and IP function design (1/2)

- Structure of a typical HW design



Hardware and IP function design (2/2)

- Separate projects for each FPGA family
- Design files organized in three folders inside the HW synthesis project folder
  - VHDL sources (synthesis and simulation)
  - VHDL simulation files (simulation only)
  - Automatic IP generator files (e.g. PLLs)

Sources	Quartus	ISE	Libero
Shared files	vhdl_src	vhdl_src	hdl
IP core files	ip_gen_src	ipcore_dir	smartgen
Simulation files	modelsim_prj	modelsim_prj	simulation

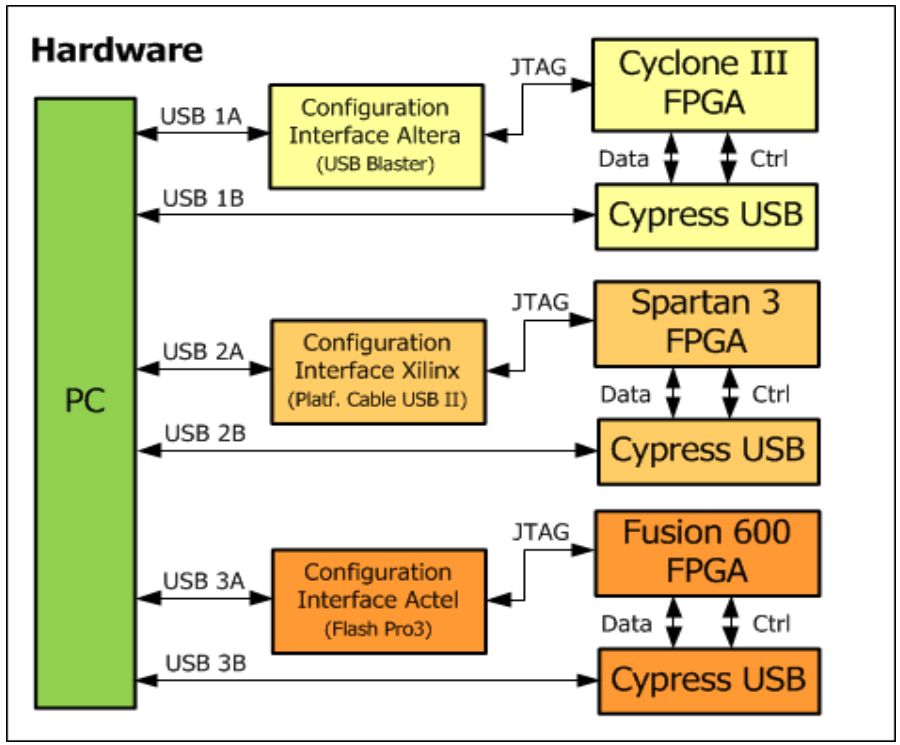
Reference designs

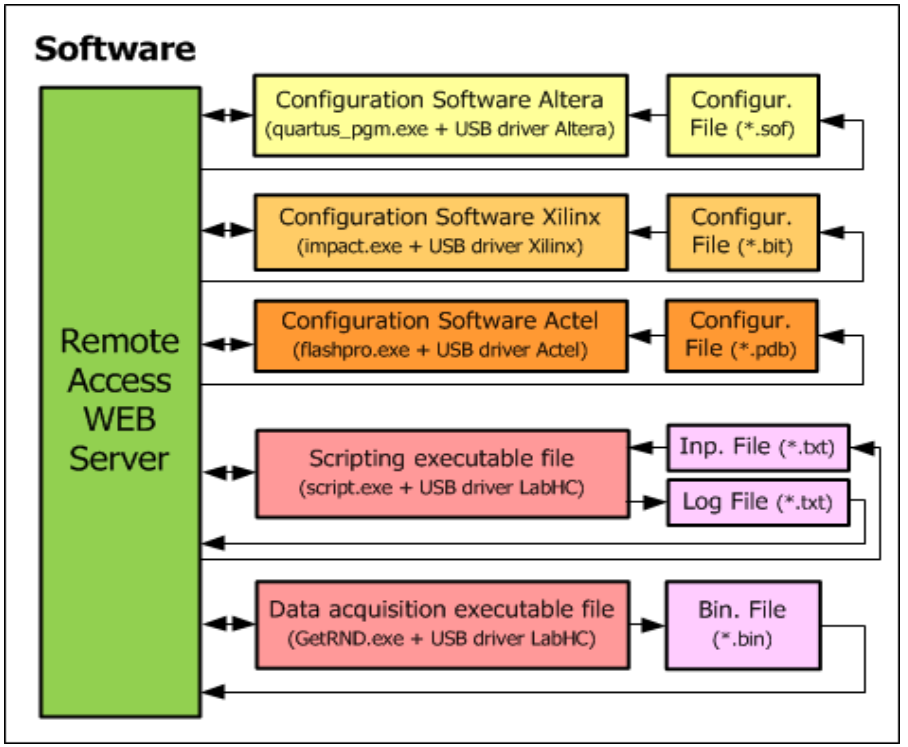
- Ref. design 1 – Data acquisition test
  - Contains bidirectional counter
  - Direction of counting changed using control word (bits(2..0))
  - Up to 48 MBits/s, unlimited size of the discontinuous data stream
- Ref. design 2 – Parameterized RO-TRNG
  - Contains two types of parameterized RO-TRNG (Sunar/Wold)
  - The generator type can be changed dynamically by ctrl_bit(2..0)
  - Up to 48 MBits/s, unlimited size of the discontinuous data stream
- Ref. design 3 – Very fast acquisition via 4 MB RAM
  - Can be used with any TRNG – one data and one clock input
  - Any input clock frequency up to 400 MHz
  - Up to 400 Mbits/s, up to 4 MB of the **continuous** data stream

Remote access to the Evariste II HW

- Remote server accessible from:

  http://labh-curien.univ-st-etienne.fr/wiki-evariste-ii





Expected workflow for remote access

- Download reference designs
- Modify a reference design and include you TRNG
- Simulate your design using modified testbench and scripts
- Upload your configuration bitstream and input script to the server
- Configure the device
- Launch script.exe on remote server and download generated log file, compare it with the simulation log file
- Launch GetRND.exe and generate random bitstreams
- Download generated bitstreams

Conclusion and future steps

- Many TRNGs (and other designs) tested, including an STR TRNG with 100Mbis/s bitrate and entropy assessment (CHES2013)
- The use of scripts make the system very flexible
- Open core – contributions welcome
- Next steps:
  - Data acquisition via optical fibers
  - Board placed in a Faraday cage
  - New FPGA families such as
    - Cyclone V
    - Spartan 6
    - SmartFusion 2