Advanced Monte Carlo Simulations On FPGA

Financial Derivatives Pricing Applications

Mikhail Tushentsov
VP Engineering, SciComp Inc.

XILINX SciComp

- Getting started with Xilinx FPGA
- Monte Carlo simulations
- Equity Linked Note case study

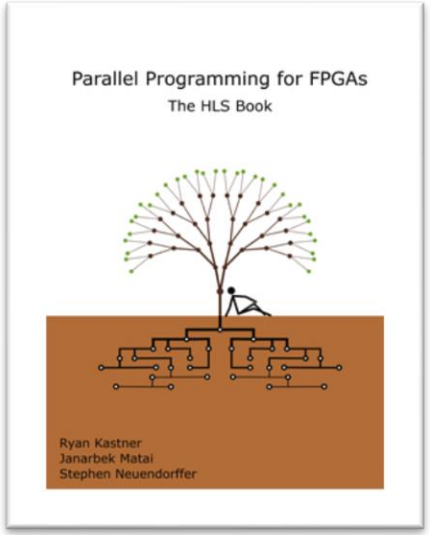
Getting started with Xilinx FPGA

How to get started developing for Xilinx FPGA?
No prior FPGA experience?

[Accelerating C, C++, OpenCL, and RTL Applications with the SDAccel Environment]

coursera

[Developing FPGA-accelerated cloud applications with SDAccel]



[Parallel Programming for FPGAs The HLS Book]

Development Tools: IDEs

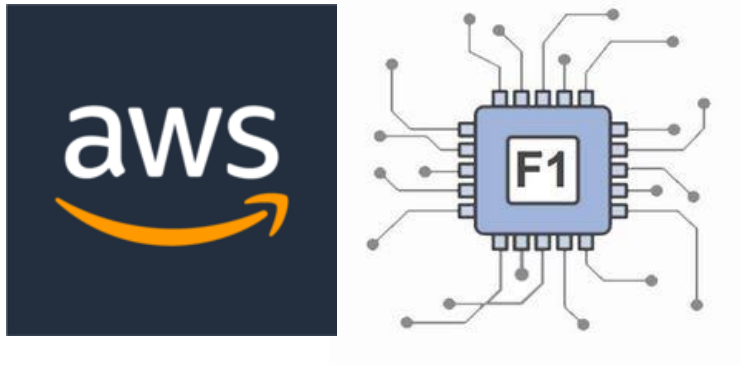
SDAccel™ Environment

- Eclipse-based IDE
- Enables a CPU/GPU-like development experience for FPGA
- Manages both host code (software) and kernel code (hardware) builds
- Debugging, profiling and code analysis tools
- SDAccel Getting started (GitHub)

- High Level Synthesis (HLS) environment for C/C++ kernel development
- C Simulation
- Synthesis to Hardware Description Languages (Verilog, VHDL)
- The HLS Book

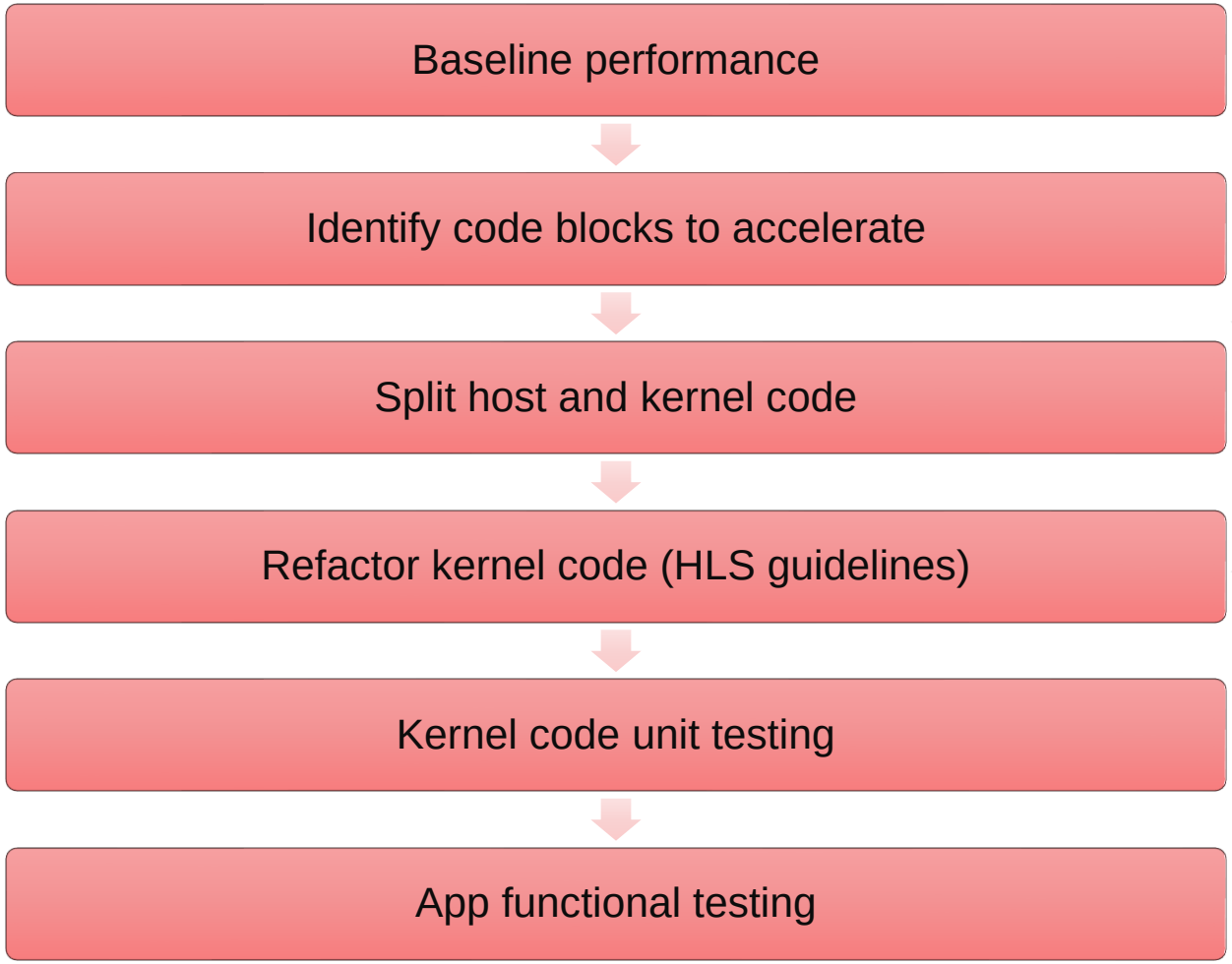
Getting started with real hardware

- **Amazon AWS**
  - F1 instances with UltraScale+™ VU9P FPGA
  - FPGA developer AMI
  - Secure deployment as Amazon FPGA Images
  - AWS Marketplace publishing

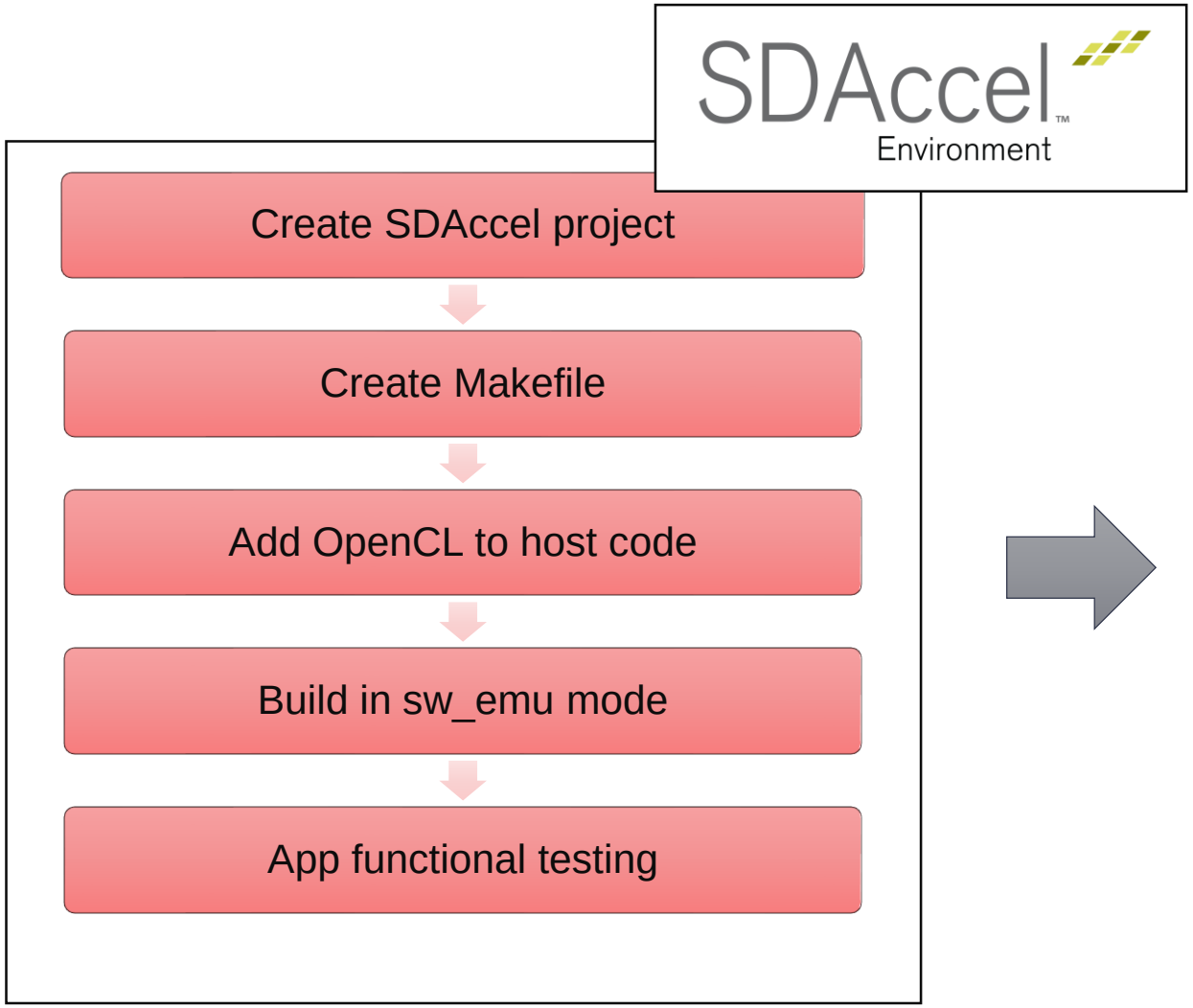


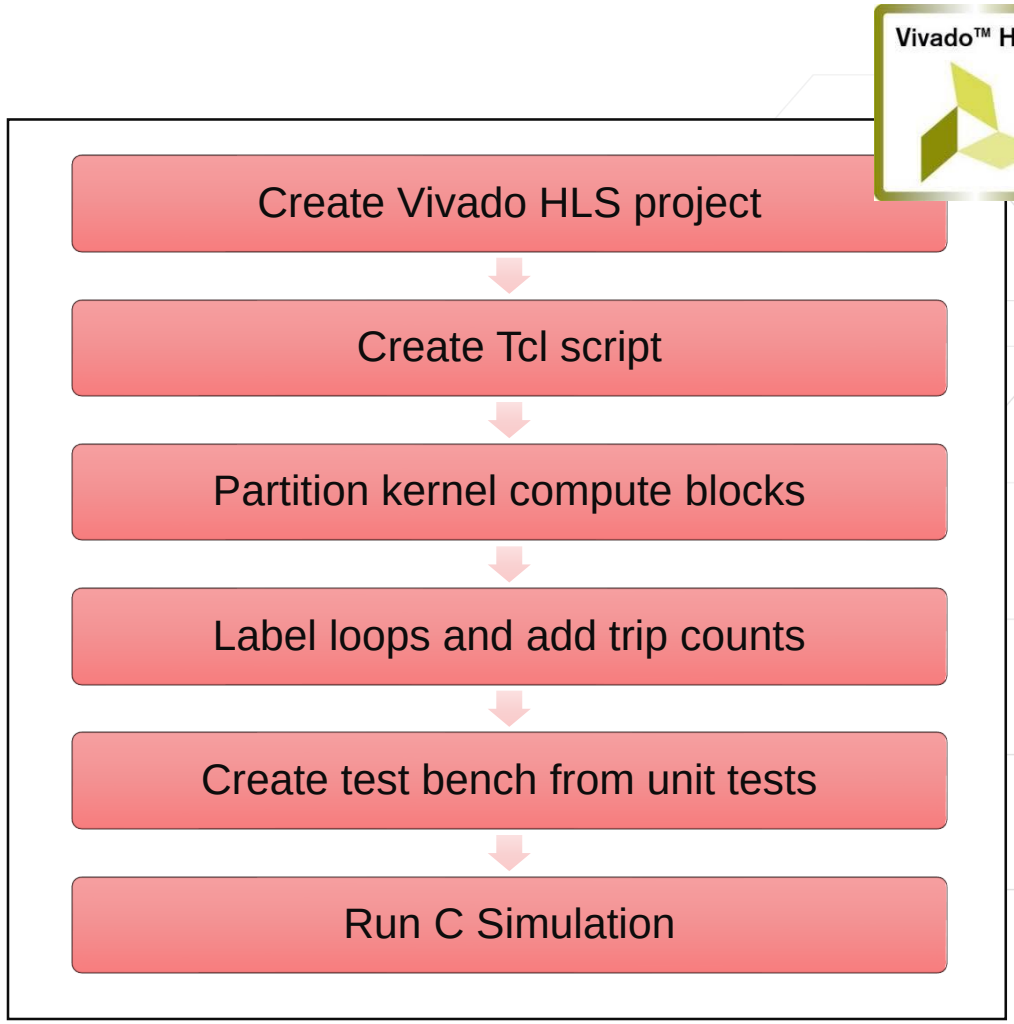
- **Nimbix Cloud (JARVICE™ platform)**
  - Alveo U200 and U250 boards
  - Quick getting started with FPGA dev
  - Simplified deployment model

Development Methodology: Original code

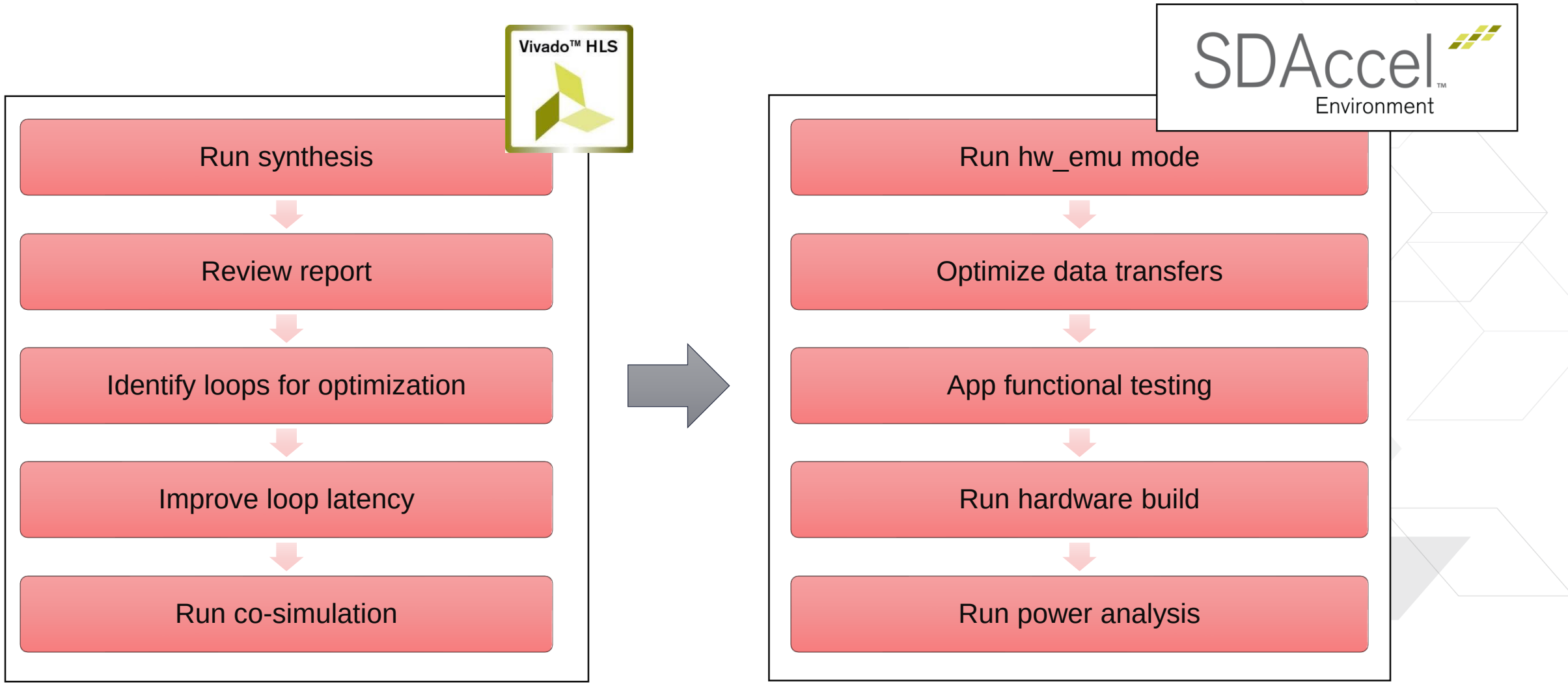


Development Methodology: Accelerated code Baseline version



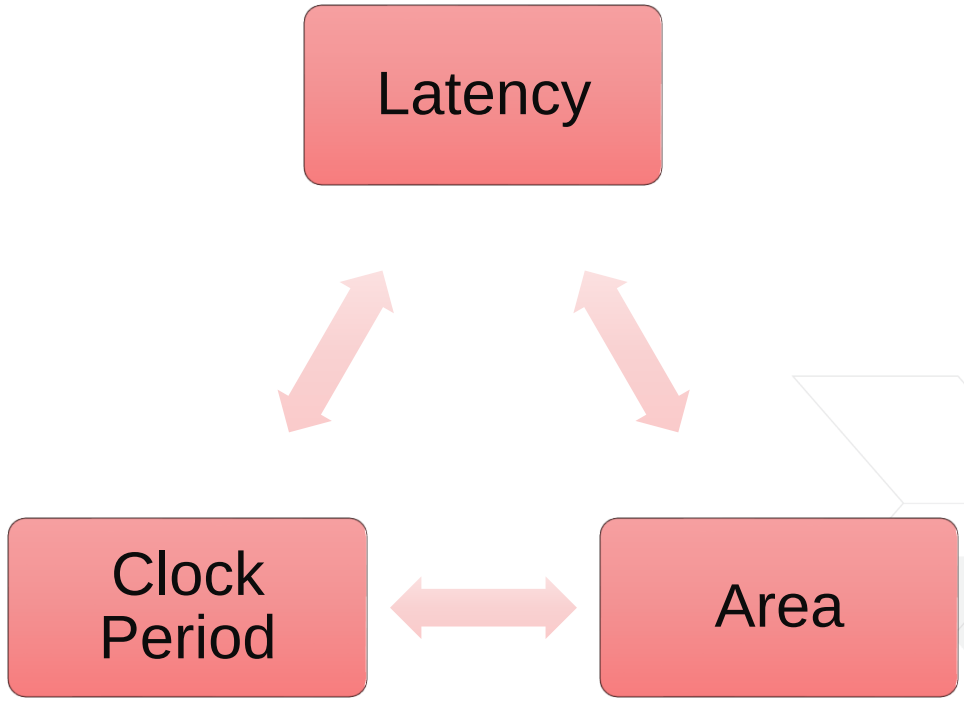


Development Methodology: Accelerate code Optimized version



The three axes of optimization

- **Design goals:**
  - Latency: number of clock cycles
  - Clock period / frequency
  - Area utilization
- **Area constraint:**
  - BRAM_18K, URAM
  - LUT, FF, DSP48
- **Performance = f(clock, latency)**
- **Power consumption = f(clock, area)**



Additional learning resources

- **Developer tools sites**
  - SDAccel: Enabling Hardware-Accelerated Software
  - Vivado High-Level Synthesis

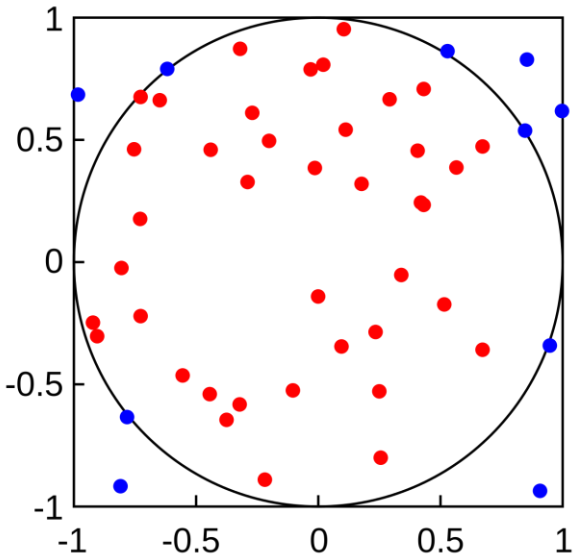
- **Xilinx Design Hubs**
  - Embedded Design Tools & IP: SDAccel Development Environment
  - Vivado Design Suite: High Level Synthesis
  - Vivado Design Suite: Power Estimation and Optimization

Monte Carlo Simulations

Monte Carlo simulations

Monte Carlo simulation example:

Finding the value of π (shooting darts)



13

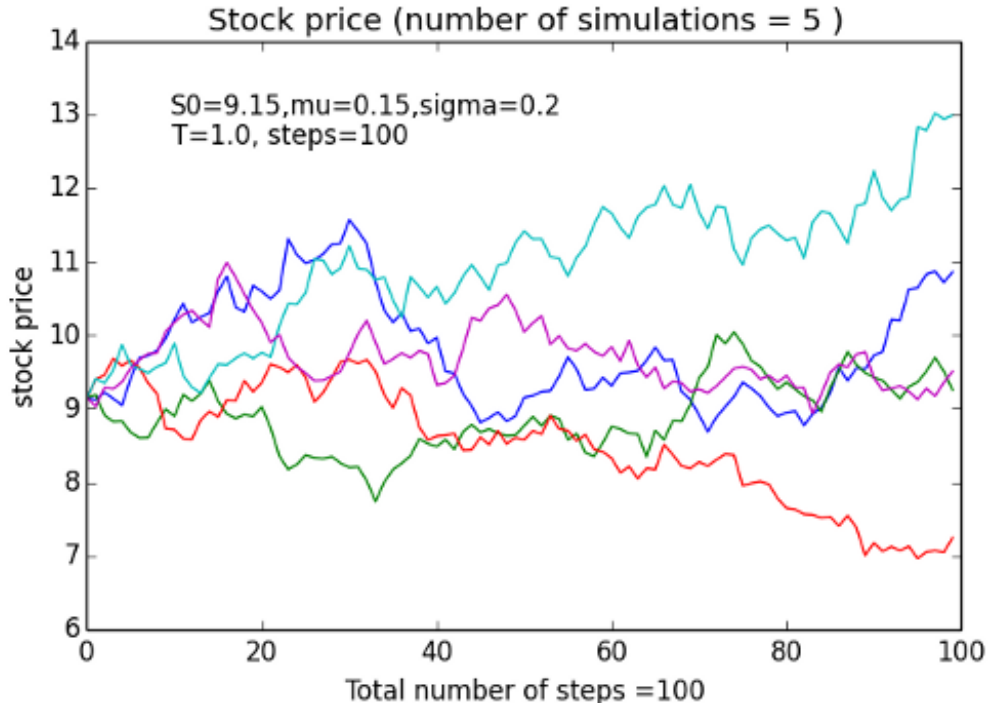
$$\frac{\text{Area of a circle}}{\text{Area of a square}} = \frac{\pi}{4}$$

$$\iint_{x^2+y^2<1} \mathrm{d}x\,\mathrm{d}y = \pi$$

$$\pi = 4\,\frac{\text{inside a circle}}{\text{total}}$$

Financial Monte Carlo Simulations – Option Pricing Arithmetic Average Asian Call Option

- Generate random sequences
- Transform to normal distribution
- Generate paths (GBM model)
- Calculate average price along the path
- Payoff = max(Average – Strike, 0)
- Average payoffs
- Present value (discount to present)



Monte Carlo components for FPGA

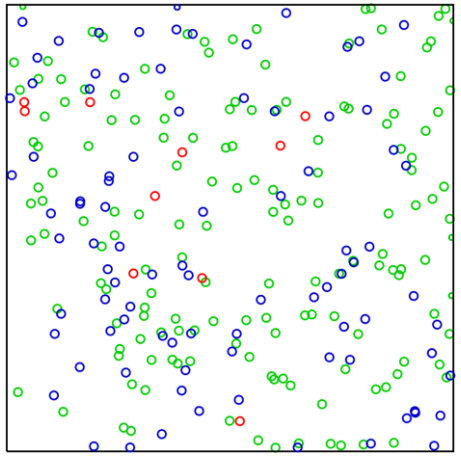
- **Pseudo-random numbers generator**
  - Small state (area efficient)
  - Simple operations for the sequence advancement
  - Jump-ahead (statistically independent sub-sequences)
- **Distribution generators**
  - Single and double precision support
  - Area efficient

Xoshiro128** pseudo RNG

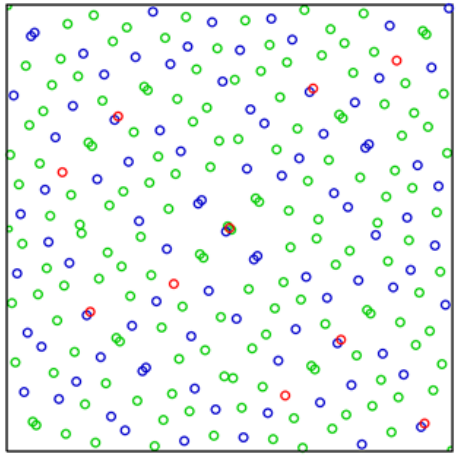
- XOR/shift/rotate
- Small state (128bit)
- Well-tested (passes TestU01 and PracRand test suites)
- Jump-ahead (statistically independent sequences, parallel generation)
- FPGA and GPU friendly
- http://xoshiro.di.unimi.it

Quasi Monte Carlo simulations

- **Computational efficiency (faster convergence, less simulations required)**
- **Low-discrepancy numbers instead of pseudo-random**
- **Brownian Bridge variance reduction**



Pseudo-random



Low-discrepancy

Sobol QRNG

- **Supports up to dimension 1023 (direction numbers available up to 21201)**
  Example: (6 index vars + 6 vols) x 48 time steps = 576 dimensions

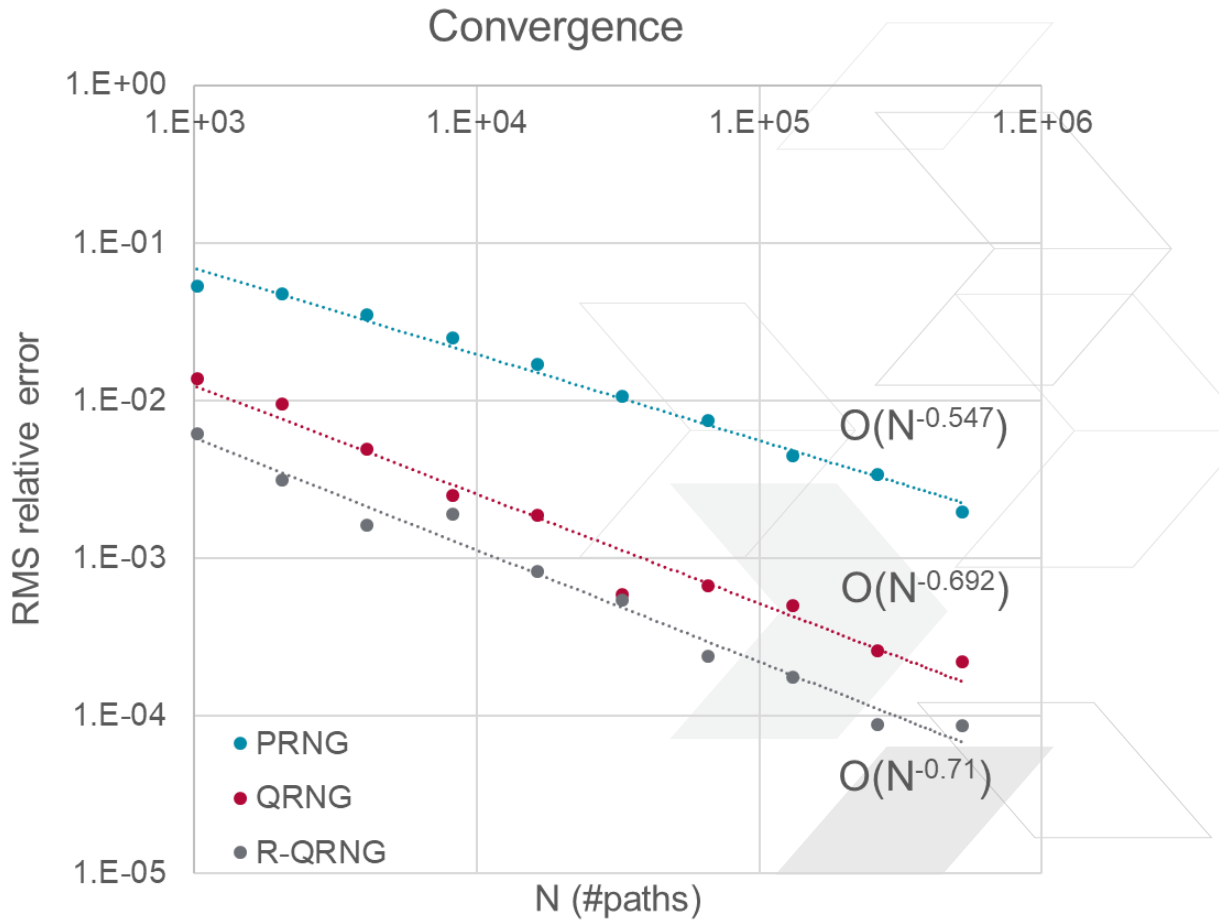
- **Owen's random scrambling technique for Sobol sequence implemented**
  Benefits: bias elimination and tighter confidence intervals

- **Sobol fast-forward (skip-ahead) for independent parallel sequences**
  Why: Different seeding and/or initial state scrambling does not guaranty the sequence independence

Case Study: Advantages of QRNG
Root mean square (RMS) relative error. (vs analytics)

- **Portfolio of Asian Geometric Average Price options**
  45 – 120 fixings

- **PRNG: xoshiro128****
- **QRNG: Sobol (JoeKuo6.21201)**
- **R-QRNG: Scrambled QRNG**

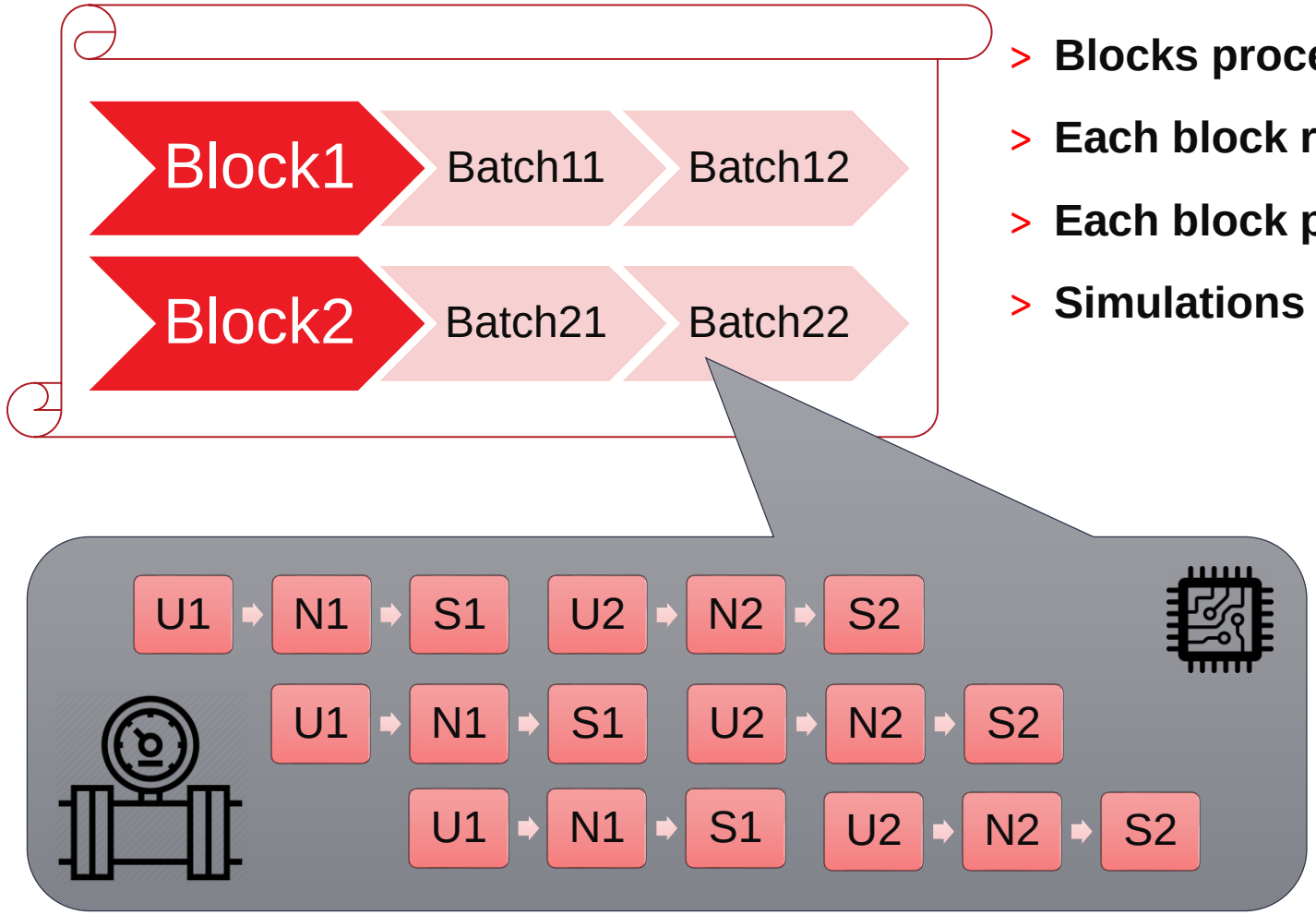


SciComp Monte Carlo HLS components

- **Has two parts: Host and Kernel HLS**
- **Host module handles the initialization (Sobol direction numbers, etc.)**
- **Kernel module performs generation and transforms using pre-calculated data**

- **Library components:**
  - (Q)RNGs
  - Distributions
  - Brownian Bridge transform

Monte Carlo FPGA implementation



- **Blocks processed in parallel (full unroll)**
- **Each block runs an independent RNG**
- **Each block processed in batches**
- **Simulations are pipelined within a batch**

U = uniform random
N = normal random
S = stock price at a time step

Equity Linked Note Case Study

XILINX

Equity Linked Note (ELN)

- **Allow clients to customize return to suit their investment needs**
- **Enhanced return if the underlying equity asset rises**
- **Varying levels of protection if the market falls***
- **Can be linked to a variety of underlying assets:**
  - Indices and single stocks
  - Commodities and currencies

*** Traditional equity investments provide full exposure to the market, whether the performance is positive or negative.**

SVEqLinkStructNote case study

- **Features:**
  - Basket of indices (6 assets)
  - Maturity coupon, bonus coupon
  - Continuously monitored knockout barrier
  - Semi-annual coupons, 3 years maturity, not cancellable
- **Model:**
  - Heston stochastic volatility model for each index
  - Cross-correlation of index levels and volatility of variance

Quasi Monte Carlo code complexity

- **24 time steps, 1M (2^20) paths, 6 indices, 3 factors (index, vol, monitoring):**
  - 3x24x6 = 432 dimensional Sobol sequence
  - 432M Sobol quasi-random numbers
  - 288M inverse cumulative normal transforms + 144M uniforms
  - 12M Brownian Bridge transforms (24 steps each)
  - 24M Correlated normal transforms (12x12 lower tridiagonal matrix)
- **Simulation**
  - Heston process with full-truncation Euler time stepping
  - Continuous monitoring of the barrier
  - 24 times steps for 6 indices, index level and volatility simulation

FPGA code metrics

FPGA timing: 250MHz design

FPGA area utilization (pseudo-random version)

Name	BRAM_18K	DSP48E	FF	LUTS
Utilization (%)	19	75	35	57

FPGA area utilization (quasi-random version)

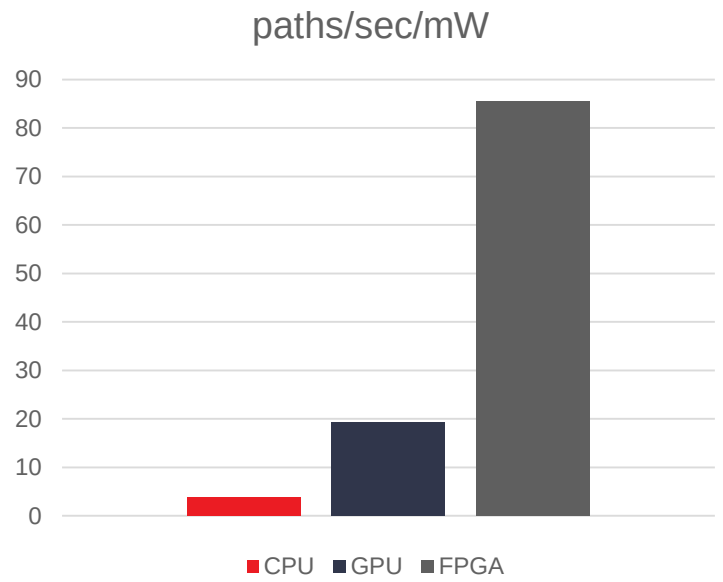
Name	BRAM_18K	DSP48E	FF	LUTS
Utilization (%)	58	77	41	70

CPU vs GPU vs FPGA benchmarking

- **Hardware setup:**
  - CPU: Intel Xeon E5-2686v4 (8 core)
  - GPU: Nvidia V100 16GB SXM2
  - FPGA: Xilinx Alveo U200

- **Software setup:**
  - OS: Ubuntu 16.04LTS
  - CPU dev: Intel Parallel Studio XE 2019
  - GPU dev: CUDA 10.1, GCC 5.4.0
  - FPGA dev: SDAccel 2018.3

HW	Model	MMpath/sec	W	paths/sec/mW
CPU	Intel Xeon E5-2686v4	0.427	110	4
GPU	Nvidia V100 16GB	3.922	203	20
FPGA	Xilinx Alveo U200	2.222	26	85



- **Power estimation :**
  - CPU: Intel SoC Watch
  - GPU: NVIDIA SMI
  - FPGA: Vivado Power Estimation

Conclusions

FPGA Benefits

- FPGA = Financial Programming Greatly Accelerated*
- Deterministic latency (no jitter) provides consistent performance
- Significant reduction in operational costs relative to both CPUs and GPUs through lower energy usage
- Small form factor FPGA data center cards are easy to deploy into existing commercial servers.

* Levyx Inc. and Intel Corp.

Conclusions

- Complex financial Monte Carlo codes are feasible for FPGA porting*
- FPGA codes demonstrate a good performance using significantly less power
- Cloud providers make FPGA development accessible with a minimal initial investment
- Xilinx provides a great set of FPGA development tools enabling FPGAs for software developers without a hardware development experience

*The case study, SciComp Monte Carlo HLS components and how-to guide will be available at GitHub https://github.com/SciCompInc/hls_montecarlo

Adaptable.
Intelligent.

XILINX