

# Accelerating Applications with the Xilinx Quantitative Finance Library

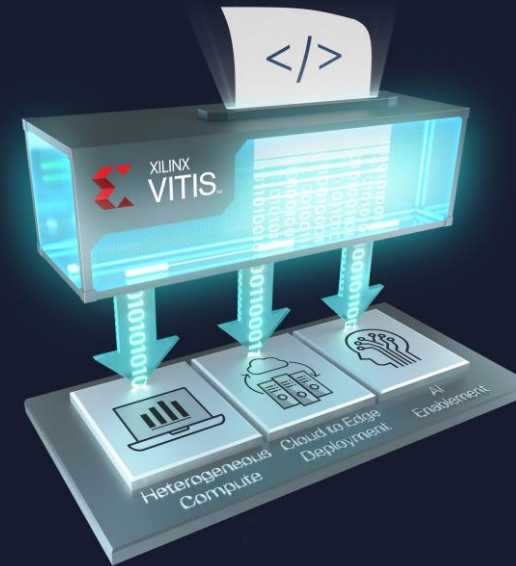
John Courtney  
Nov 2019



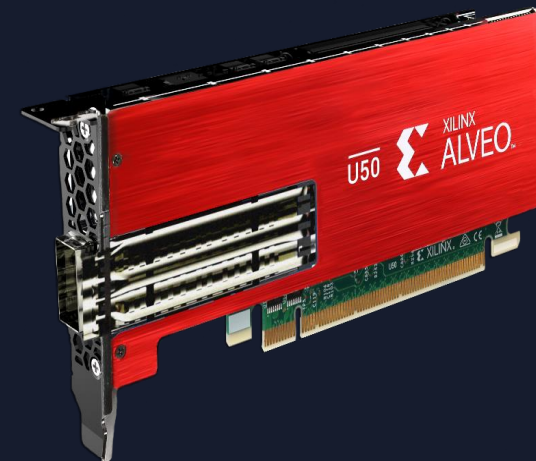
# Vitis Software Platform and Alveo Cards.



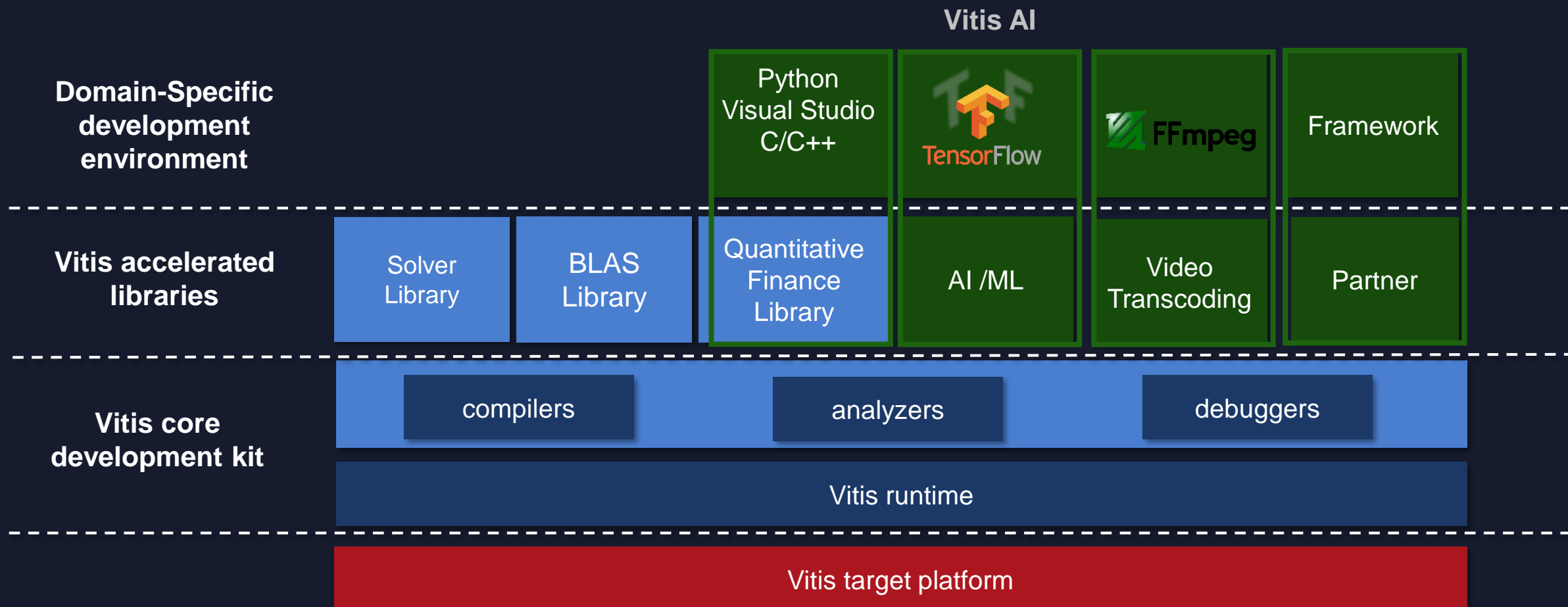
- Standards, Open



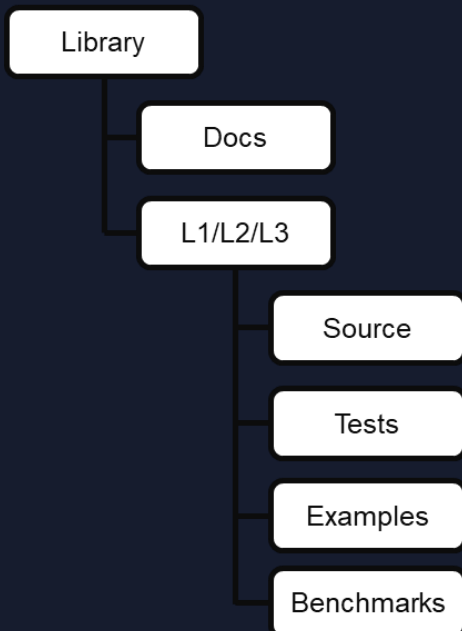
- Acceleration Cards



# Vitis: Unified Software Platform



# Vitis Accelerated Libraries



**XILINX** [Developers](#) [Support](#)

🏠 **Vitis Quantitative Finance Library**  
2019.2

Search docs

**LIBRARY OVERVIEW**

- Requirements
- License
- Trademark Notice
- Release Note

**USER GUIDE**

- Pricing Models and Numerical Methods
- L1 Module User Guide
- L2 Kernel User Guide**
  - Pricing Engine Overview
  - Pricing Engine Kernel Design
  - Pricing Engine Kernel APIs
  - Pricing Engines Demo
  - Other Engine Kernel Design
- L3 Overlay User Guide

🏠 » L2 Kernel User Guide

## L2 Kernel User Guide

- Pricing Engine Overview
- Pricing Engine Kernel Design
  - Internal Design of MCEuropeanEngine
  - Internal Design of MCEuropeanHestonEngine
  - Internal Design of Asian Option Pricing Engine
  - Internal Design of Digital Option Pricing Engines
  - Internal Design of Barrier Option Pricing Engine
  - Internal Design of Cliquet Option Pricing Engine
  - Internal Design of American Option Pricing Engine
  - Internal Design of MCMultiAssetEuropeanHestonEngine
  - Internal Design of MCHullWhiteCapFloorEngine
  - Internal Design of MCEuropeanHestonGreeksEngine
  - Internal Design of Closed Form Black-Scholes-Merton
  - Internal Design of Closed Form Heston
  - Internal Design of Closed Form Merton 76
  - Internal Design of Garman Kohlhagen
  - Internal Design of Quanto
  - [Internal Design of Cox-Ross-Rubinstein Binomial Tree](#)
  - Internal Design of Finite-Difference Hull-White Bermudan Swaption Pricing Engine

# Xilinx VITIS Quantitative Finance Library

Equity Product

Credit Product

Interest Rate Product

Commodity Product

FX Product

Black-Scholes  
Heston

European  
American

Asian  
Barrier

Digital  
Cliquet

**Linear Algebra**  
Cholesky Decomp  
LU/SVD/QR Decomposition  
Dense Matrix Multiply  
Sparse Matrix Multiply  
...

**Statistics**  
Random Number Generator  
Box-Muller Transform  
Distributions  
...

**Financial**  
Finite Difference  
Monte-Carlo Methods  
Brownian Bridge  
Closed Form Solutions  
Greeks/Sensitivities  
...

**Solver**  
Tridiagonal Solver  
Pentadiagonal Solver

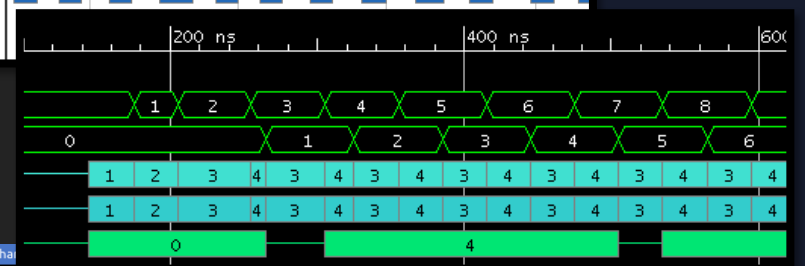
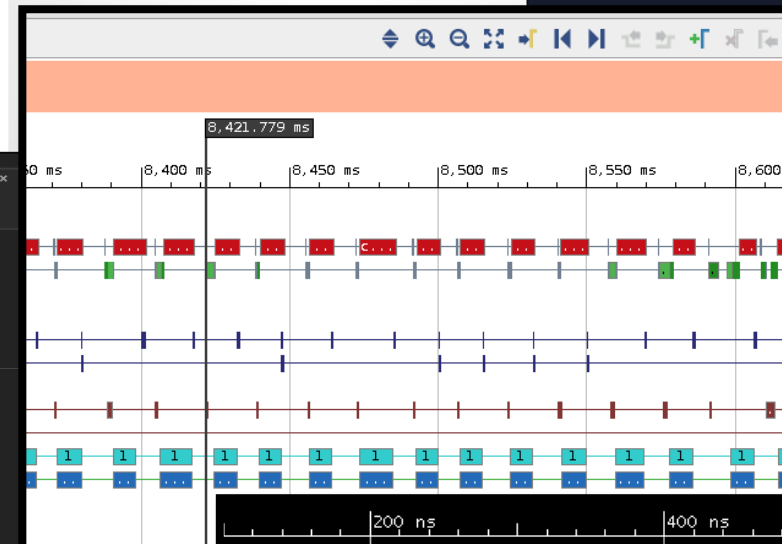
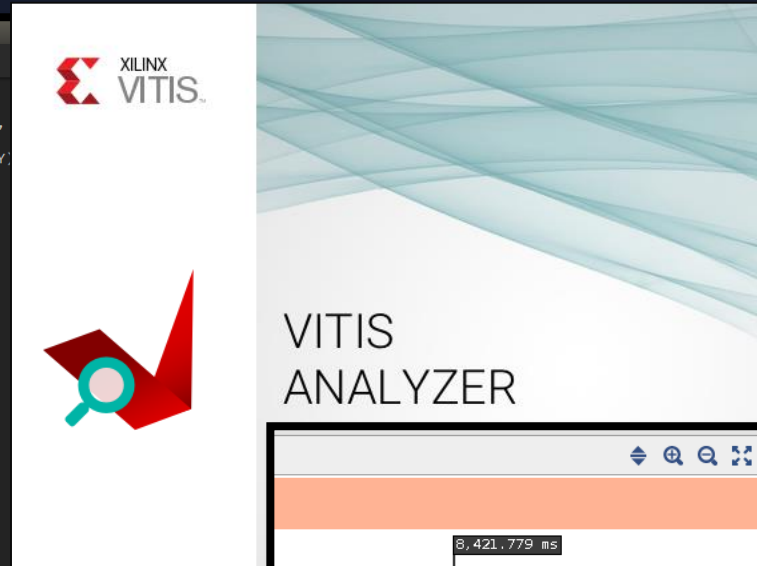
**Basic Math Function**

sqrt, abs, fabs, exp, log, pow, sin, cos, asin, acos, sinh, cosh, floor, fmod, modf, etc.

# Vitis -- Familiar SW Environment

The screenshot shows the Visual Studio Code editor with the following components:

- EXPLORER:** Lists project files including `webcam_densebox.cpp`, `zcu102_dpu_test`, `drm_test.cpp`, `zynq_drm.cpp`, `zynq_drm.hpp`, `resources`, `util`, `event_timer.cpp`, `event_timer.hpp`, `line_exception.hpp`, `threaded_queue.hpp`, `v4l2`, `xv4l2src_webcam.cpp`, `xv4l2src_webcam.hpp`, `webcam_densebox`, `densebox.cpp`, `densebox.hpp`, `qos.cpp`, `qos.hpp`, `resizer.cpp`, `resizer.hpp`, `webcam_densebox_common.hpp`, `xilinx_ocl`, `line_exception.hpp`, `xilinx_ocl.cpp`, `xilinx_ocl.hpp`, `dpu_test.cpp`, and `modetest.c`.
- OUTLINE:** Shows a tree view of the code structure with symbols like `num_frames`, `used_frames`, `apm_counter`, `display_splash`, `is_key_pressed`, `logger`, `main`, `pageflip_callback`, `restore_termios_at_exit`, `initial_term_settings`, and `drm_pflip_data`.
- EDITOR:** Displays C++ code for `webcam_densebox.cpp`, including headers, `main` function, and a loop that processes video frames using `cv::Mat` and `cv::cvtColor`.
- DEBUG CONSOLE:** Shows the output of the program, including the message `Resizer joined` and the number of faces detected (`num_faces`).
- DEBUG WINDOW:** Shows the state of a `Buffer` object, including `count`, `data`, `fd`, `id`, `initialized`, `length`, `metadata`, `num_faces`, `ocl_initialized`, `vbuf`, and `first`.



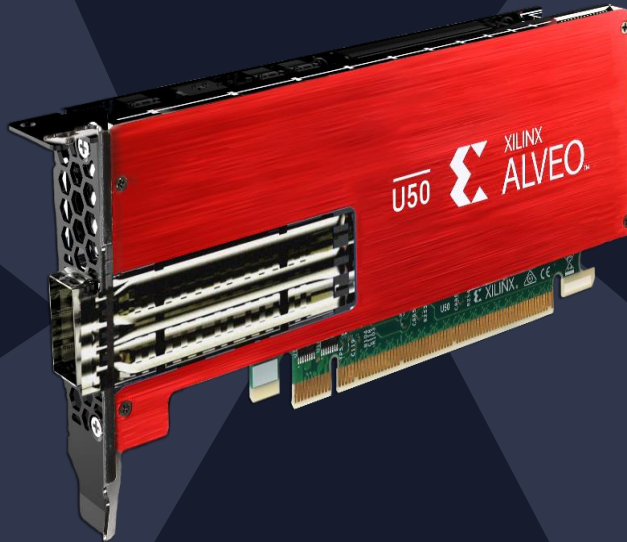
# Alveo U50 – Low Profile Acceleration Card



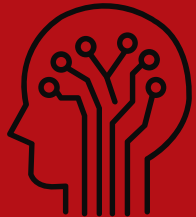
**Fintech**



**Storage**



**Machine Learning**



**Database**

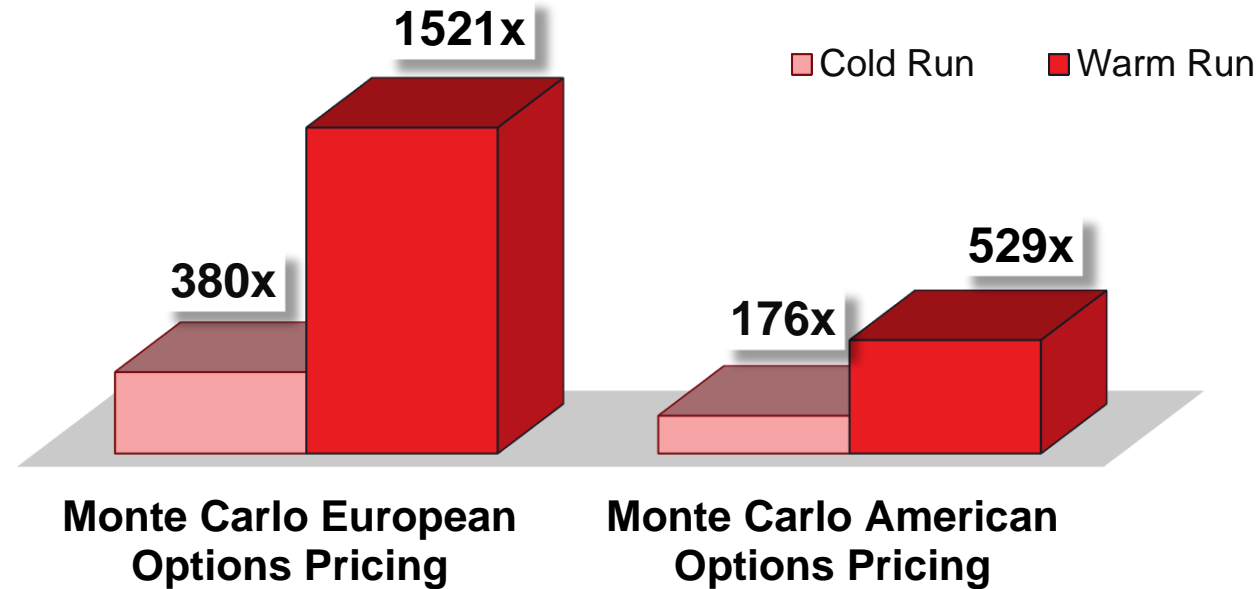


Card	Alveo U50
Primary Application	Fintech + Storage + Database + ML
FPGA Design	XCU50
CCIX	Yes
Device VCCINT	0.85V
Width	Single slot
Form Factor (Passive)	HHHL
Memory Target	8 GB HBM
Memory Config	Dual Stack, 32 pseudo-ports
PCIe	2x Gen4x8, 1x Gen4x8, Gen3x16, CCIX
Network I/F	2x SFP- DD* or 1X QSFP28
Thermal	Passive
Power (Max TDP)	75W
KLuts	872K

\* During ES, U50 card will have 2 SFP-DD ports

# Quantitative Finance Library (NON-STAC Benchmark)

## Speed-Up Vs. QuantLib on CPU



Monte Carlo European Options Pricing		
	Cold Run	Warm Run
QuantLib	20.155 ms	20.155 ms
Vitis Quantitative Finance Library	0.053 ms	0.01325 ms
Speed-Up	380X	1521X

Monte Carlo American Options Pricing		
	Cold Run	Warm Run
QuantLib	1038.105 ms	1038.105 ms
Vitis Quantitative Finance Library	5.87 ms	1.96 ms
Speed-Up	176X	529X

**CPU** : 2 Intel(R) Xeon(R) CPU E5-2690 v4 @3.20GHz, 8 cores per processor and 2 threads per core.

**Xilinx** : Vitis Quantitative Finance Library v1.0 running on 1 Alveo U250

**Cold Run**: Pricing Engine starts up in response to a request.

**Warm Run**: Pricing Engine is already running, with sufficient memory allocated to handle the request



**Adaptable.**  
**Intelligent.**

