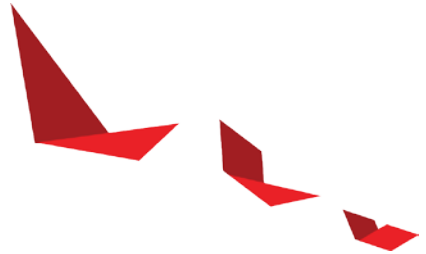
XILINX

Vitis tool flow

Introduction to Vitis

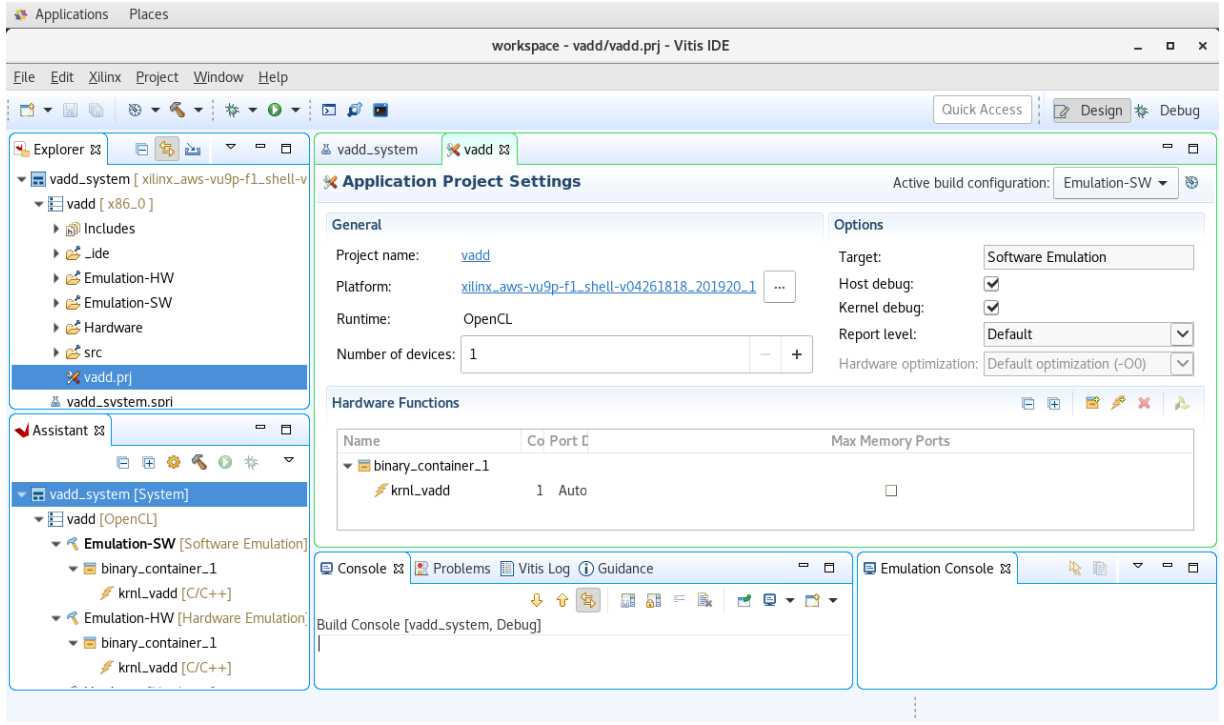
Goal



- Understand the Vitis tool flow
  - How a design is compiled to a software application and hardware kernel
  - How to boost application performance
  - How to debug and verify your design
  - What the Vitis Analyzer is for
  - What other help is available

Vitis Development Environment

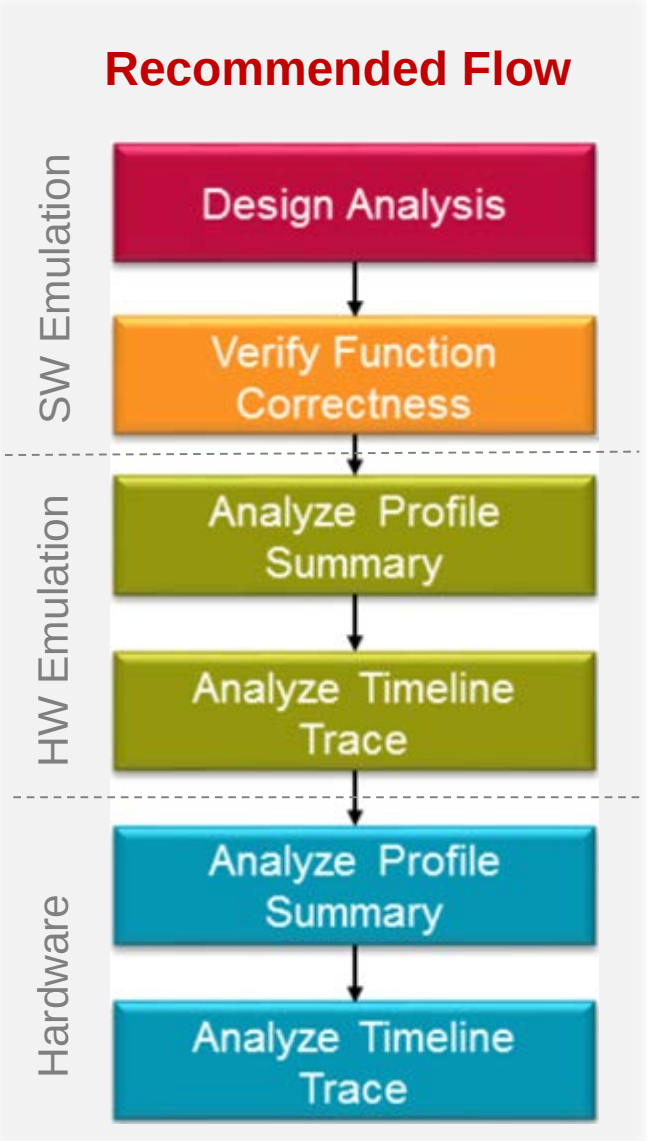
- Eclipse-based IDE
- Develop, profile and deploy Xilinx accelerated applications
- Concurrent programming of host application and hardware kernels
- Automatic build and execution flows
- Built-in debug, profiling and performance analysis tools



XILINX.

Vitis flow

- Three modes of operation
  - Software Emulation
  - Hardware Emulation
  - Run on Hardware
- Compilation time increases for each level
- Emulation is faster, but is less detailed
  - Good for verifying functional correctness
- Final performance evaluation needs to be done in real hardware

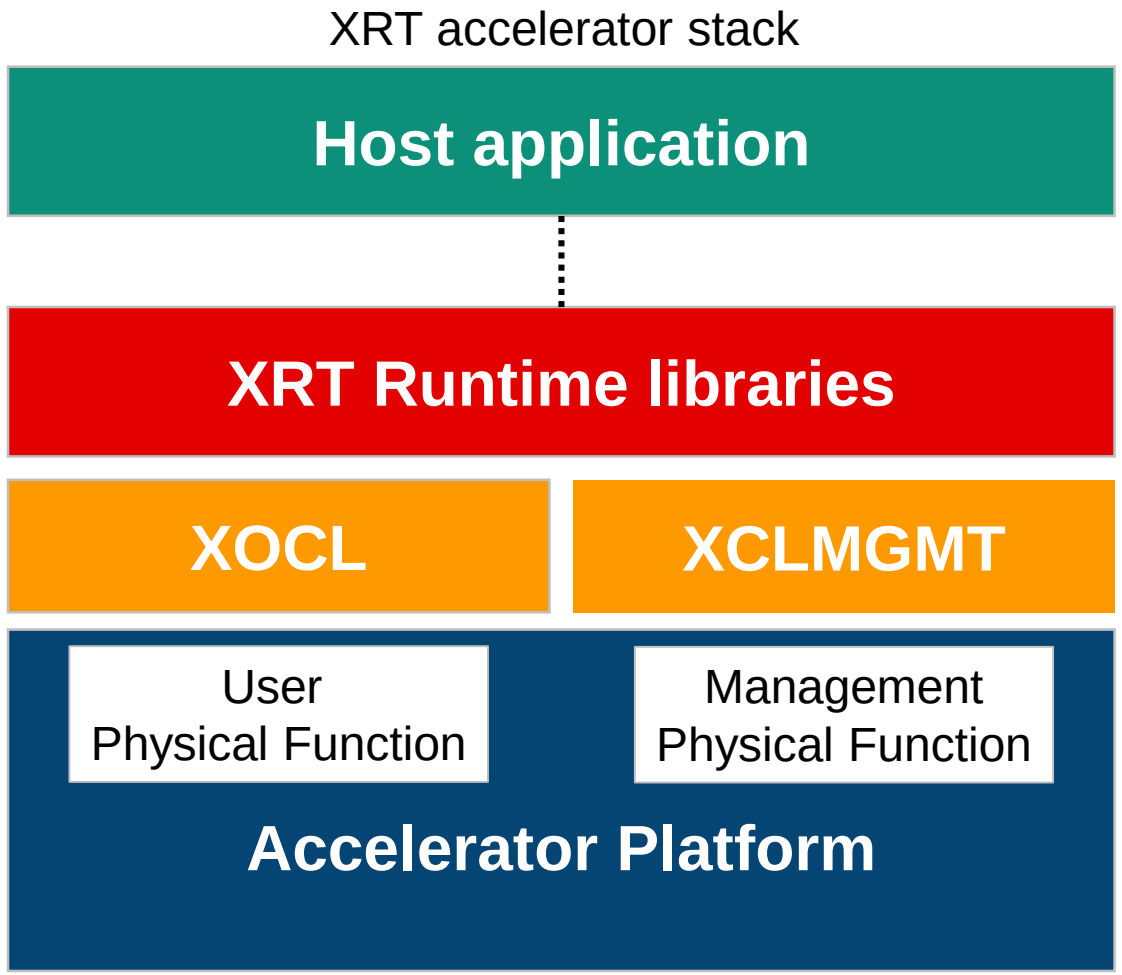


Testing and Execution Modes

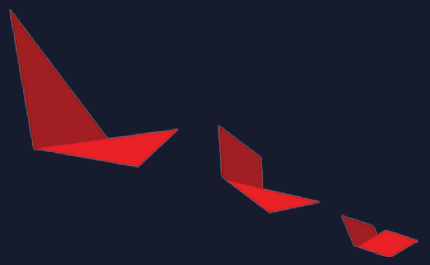
Software Emulation	Hardware Emulation	Run on Hardware
Host application runs with a C/C++ or OpenCL model of the hardware Kernels	Host application runs with a simulated RTL model of the hardware Kernels	Host application runs with actual FPGA implementation of the hardware Kernels
Confirm functional correctness of the system	Test the host / kernel integration, get performance estimates	Confirm system runs correctly and with desired performance
Fastest turnaround time	Best debug capabilities	Accurate performance results

Host Framework and Execution Stack

- XRT runtime manages communication with the hardware or emulator
- User Interface
  - OpenCL popular in parallel computing
  - C/C++ API, Python also supported
- OpenCL
  - Host application submits work to FPGA kernels using standard OpenCL API
  - OpenCL runtime to manage kernel scheduling during execution



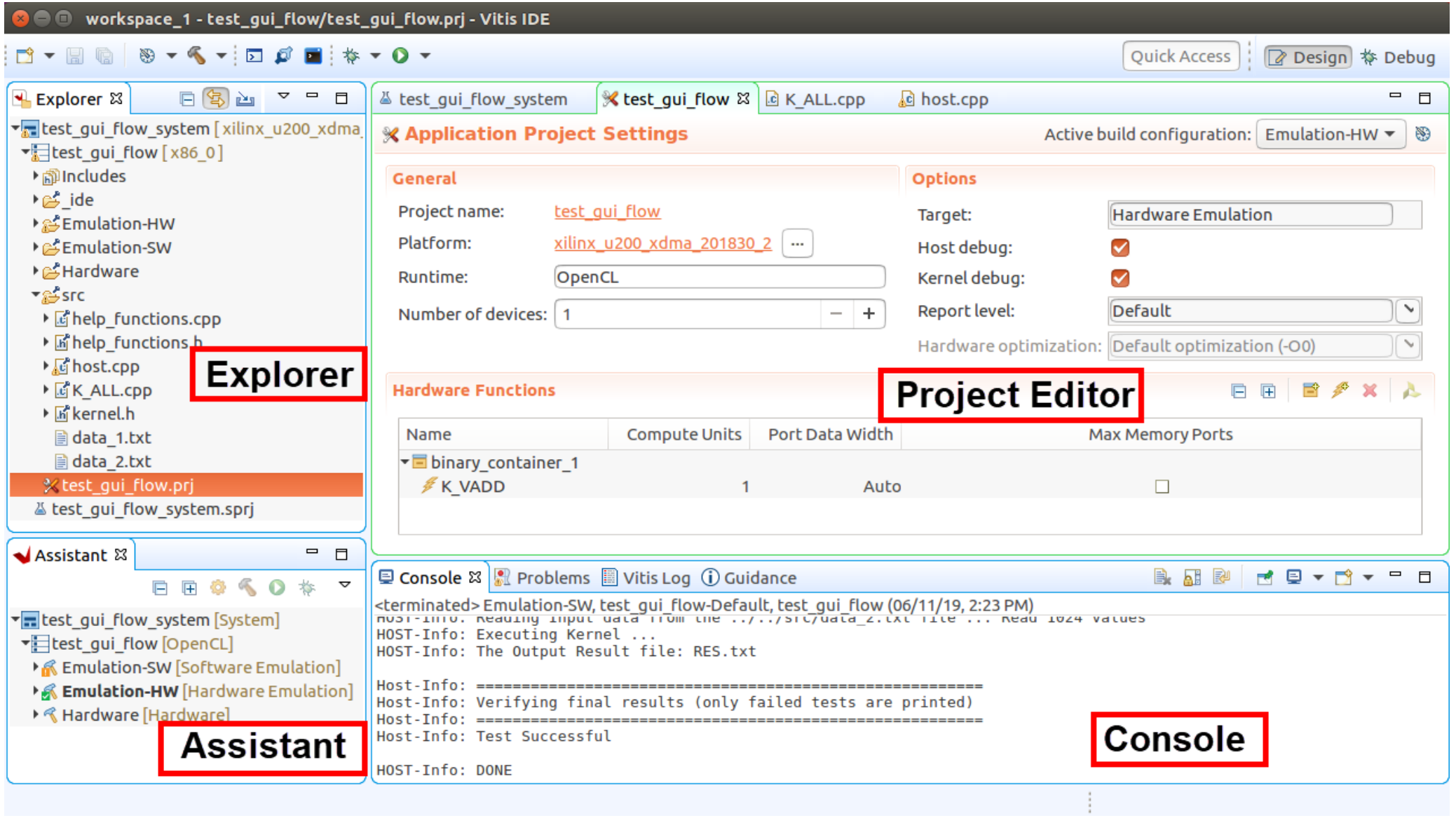
https://xilinx.github.io/XRT



Vitis environment

XILINX

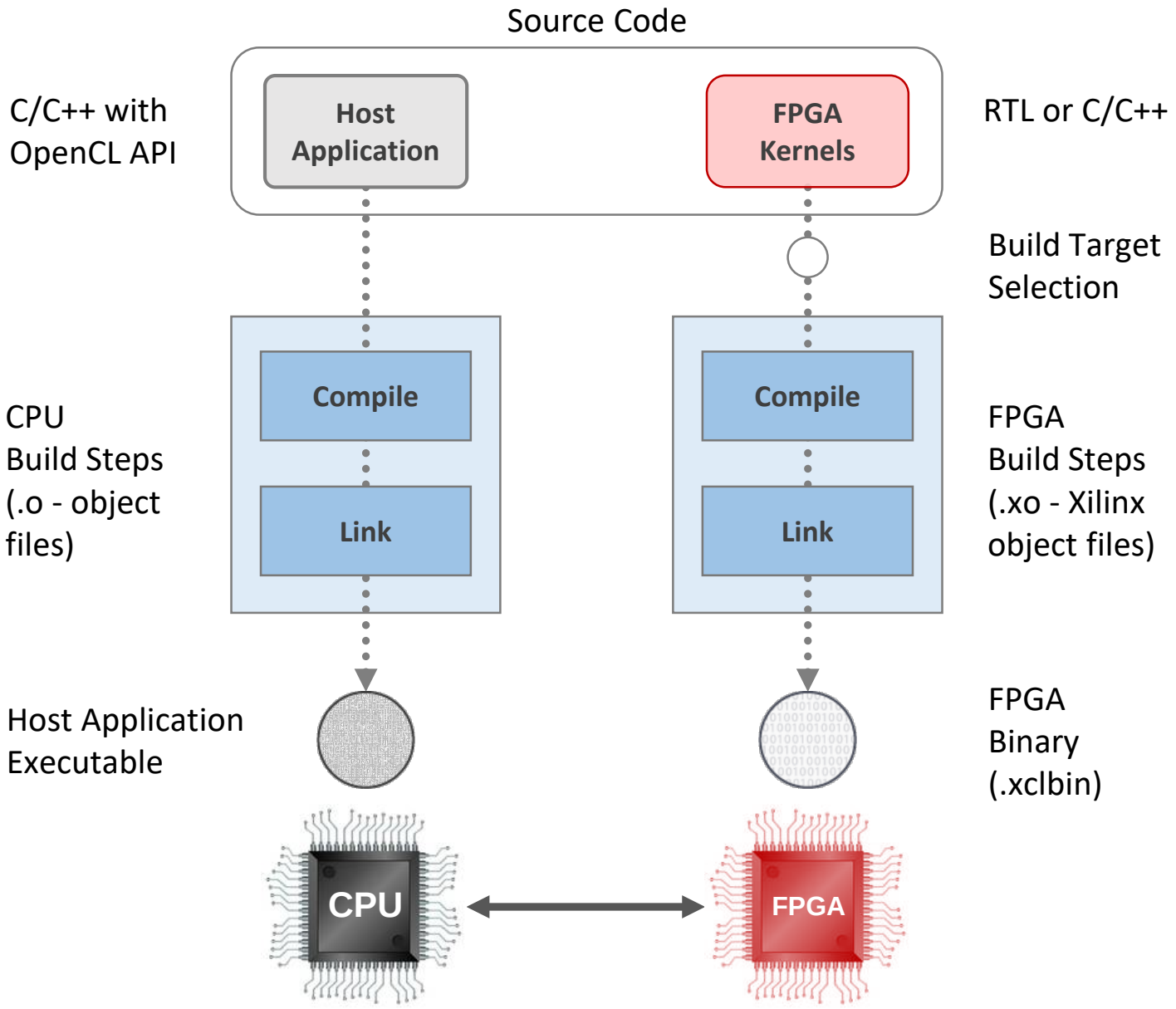
Vitis IDE



XILINX.

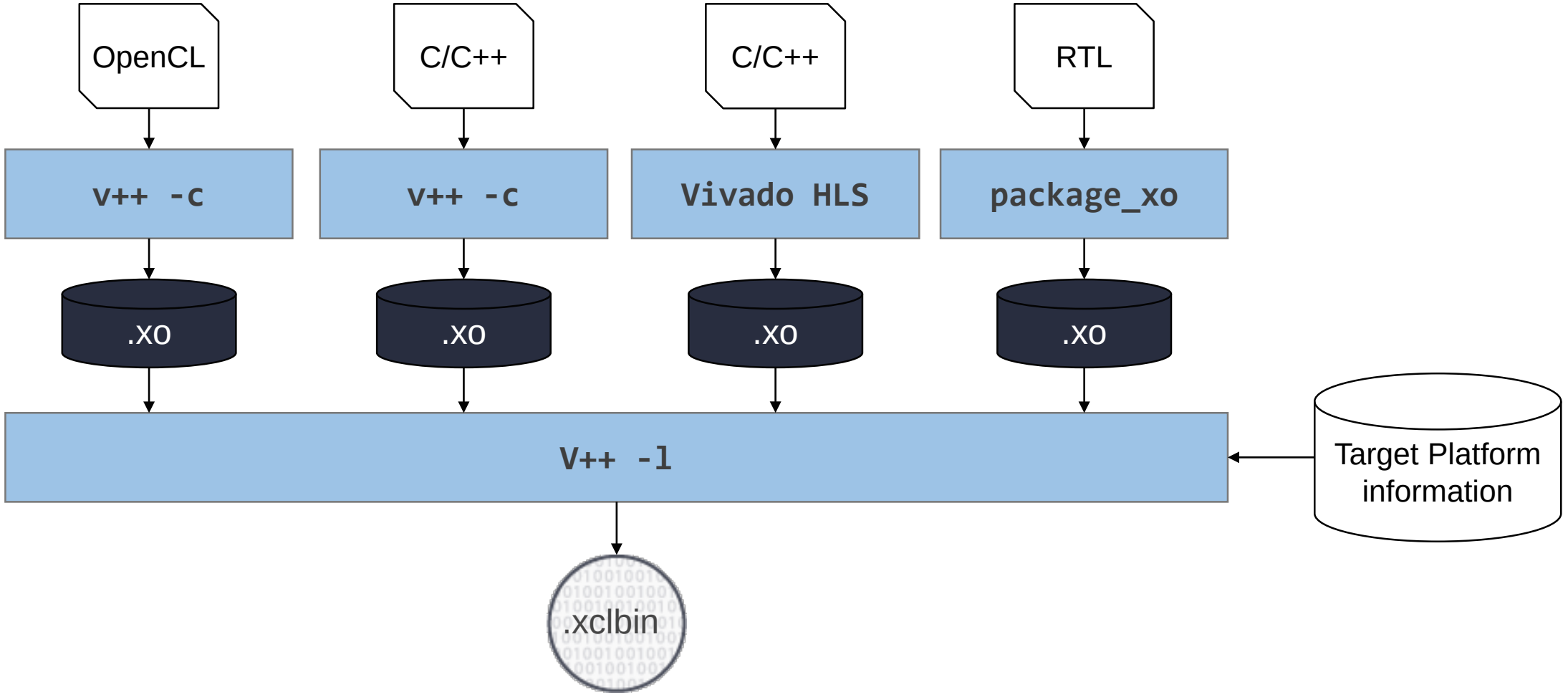
Building the Host Code

- Vitis builds host code and kernel code
- Host code
  - Written in C/C++, using OpenCL
  - Build using g++
  - Include XRT libraries
- Hardware kernel code
  - Written in C/C++/OpenCL or RTL
  - Built using the Vitis compiler v++



XILINX.

Building the hardware Kernel – FPGA Binary



XILINX.

Debugging

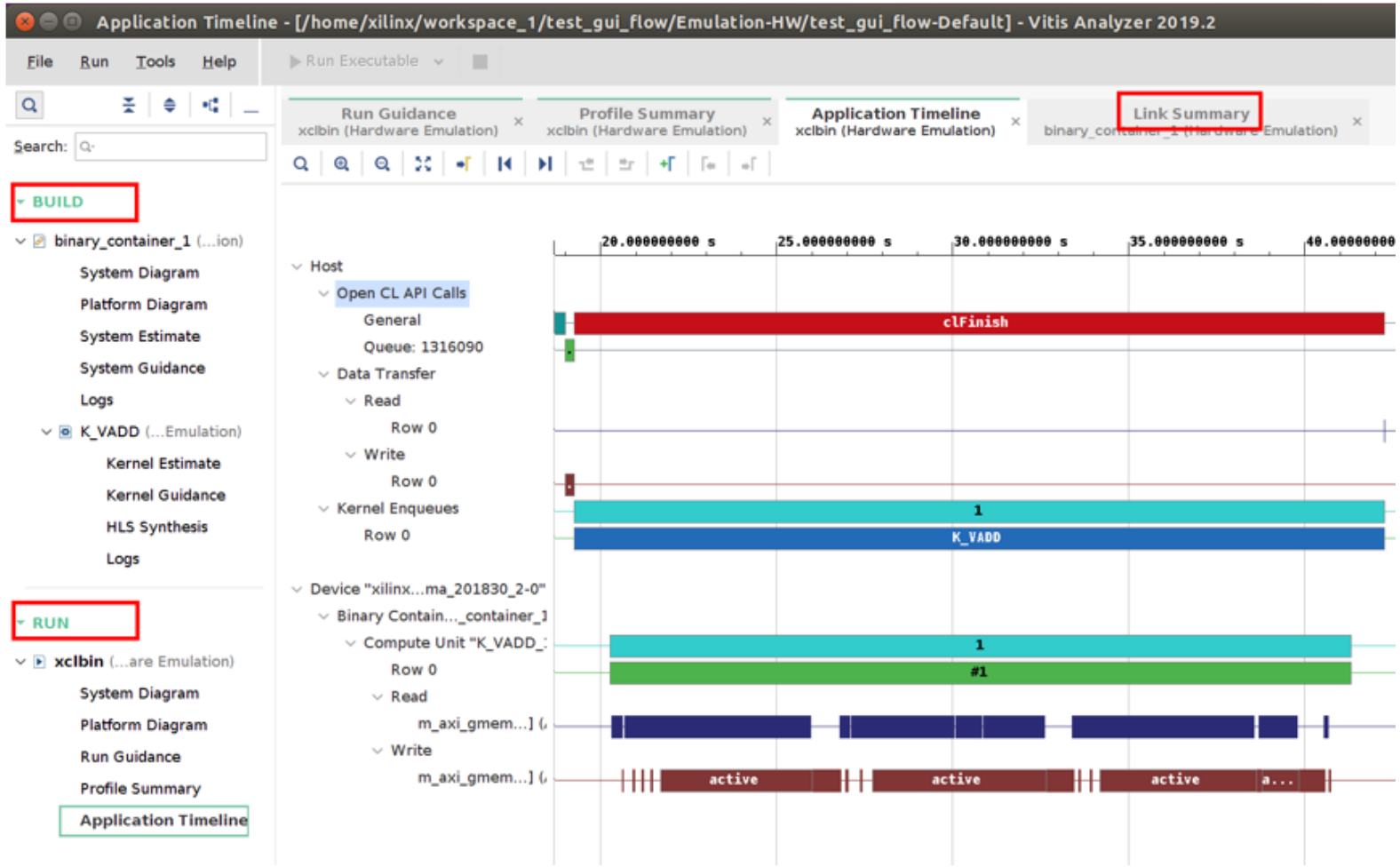
Debugger	Source	Software Emulation	Hardware Emulation	System Run
GNU debugging (GDB) tool	Host code	✓	✓	✓
	Kernel code	✓	Limited support	✗

- Waveform-based kernel debugging
  - Hardware designers use this approach
- `printf()` or `cout` can be used
  - C/C++-based kernels
    - Only supported during software emulation
    - Excluded from Vivado HLS synthesis
  - OpenCL kernels
    - Supports `printf()` for all build configurations: SW Emulation, HW Emulation, and system run

```
#ifndef __SYNTHESIS__
    printf("Checkpoint 1 reached");
#endif
```

Profiling: Vitis Analyzer

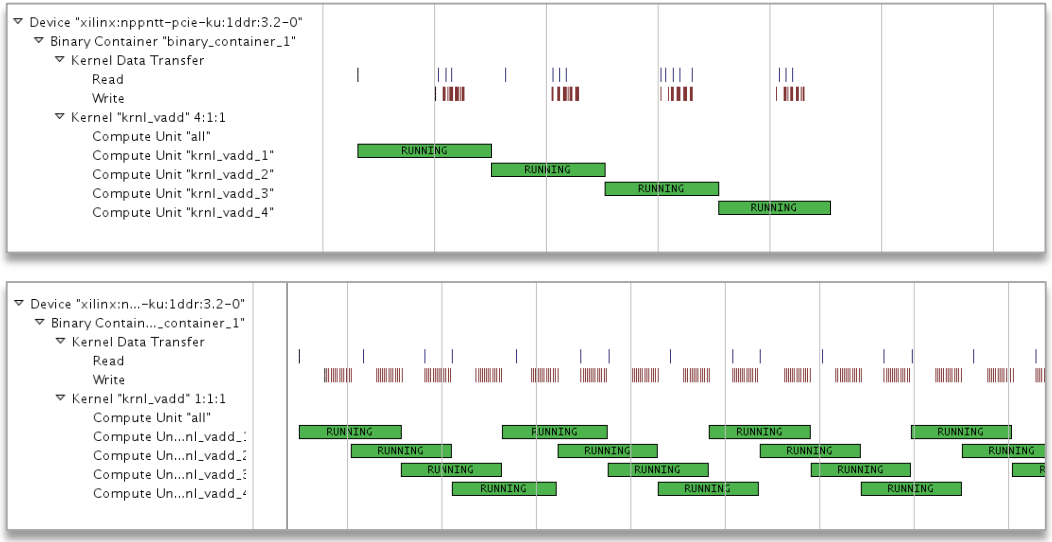
- Vitis analyzer
  - View and analyze the reports
- Useful reports
  - Compile Summary
  - Link Summary
  - Run Summary



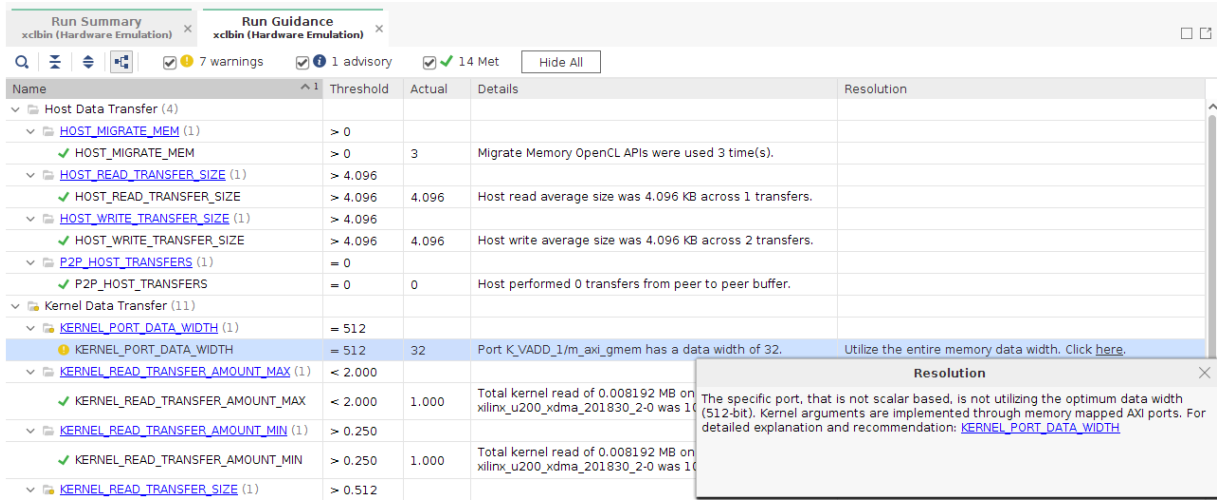
XILINX.

Vitis Platform Supports and Guides Optimization Process

Application Timeline View



Design Guidance



Thank You