● Xilinx and Intel FPGA software portfolio overview

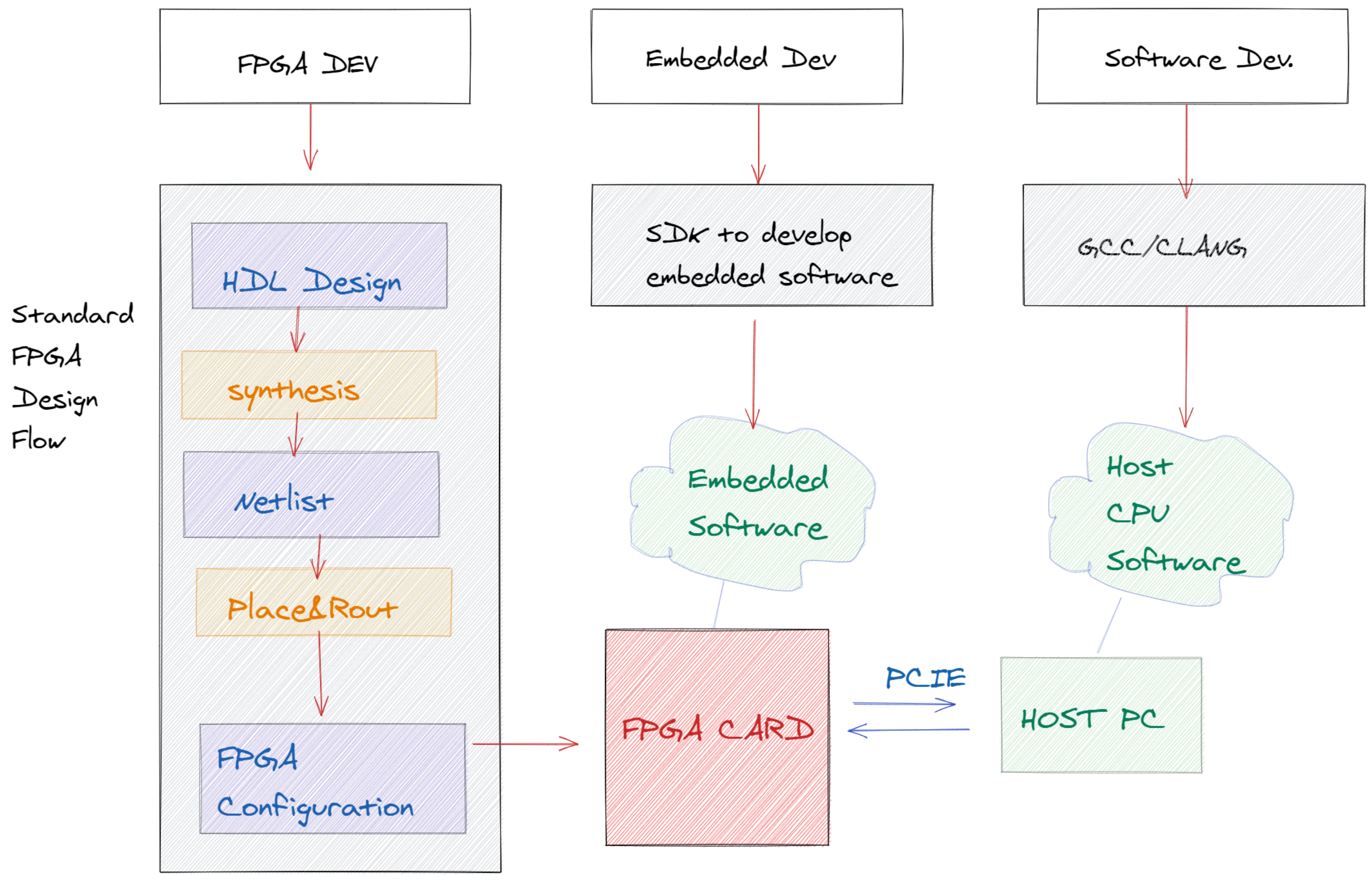
By Konstantin V.
Join us t.me/pdp11ml

Two main FPGA vendors, Xilinx and Intel (ex. Altera) offer a rich and quite wide portfolio of different tools with very similar names and description. This work covers 15 of them, including obsolete ones, but still referenced in publications. This paper is software developer oriented, still some basic knowledge of what is an FPGA is required.

Industry standard FPGA design flow consists of the following stage

- HDL code development (Verilog, SystemVerilog, VHDL)
- HDL simulation - time simulation, produce waveforms and logs. (Not shown in the diagram)
- Synthesis - transform HDL code into the netlist, or schematic.
- Place & Route - map the netlist into available FPGA blocks and route connectivity between block
- Configure FPGA, use on-chip debug tools to verify it

FPGAs may have CPU and applications executed on this processor are called embedded software. FPGA board may be connected to computer via PCIE interface. Applications executed on this computer are called a host CPU software.



Embedded software and FPGA developers roles can be unified in the single person or divided among large groups for each zone of responsibilities.

Xilinx performs FPGA design flow in Vivado HLx IDE and Altera provides Quartus.

> Note : Previously Xilinx used ISE, but Since 2012, Xilinx ISE has been discontinued in favor of Vivado Design Suite, that serves the same roles as ISE with additional features for system on a chip development.

As mentioned, FPGA may have

- Hard processor subsystem - silicon implemented, in addition to FPGA resources.
- Soft processor, implemented with FPGA resources.

FPGA with processors are often referred as System-on-Chip(SoC). For the hard processors subsystem both Altera and Xilinx uses A9 ARM core in the previous generation of FPGA SoC and A53 in the latest SoC families. Xilinx also

ARM Cortex CPU	Xilinx	Intel
Dual Core A9	a Zynq-7000 SoC	Arria 10, Arria V SoC Cyclone V SoC
Quad Core A53	Zynq UltraScale+ MPSoC	Agilex 10 SoC. Stratix 10 SoC
Dual Core A72	Versal ACAP (Adaptive Compute Acceleration Platform)	----

For soft CPU Xilinx uses Microblaze and Altera NiosII. Soft CPUs can be implemented in any FPGAs, but this CPUs are very slow (~100MHz) and cannot be used for intensive computations.

To develop embedded software under SoC CPUs Xilinx earlier offered XSDK and Intel offers Intel SoC FPGA Embedded Development Suite. Note, that both on A9 and A53 one may run not only bare metal applications, but also Linux OS and real-time OS.

Years ago Xilinx used Platform Studio and the Embedded Development Kit (EDK) but both were replaced with XSDK.

It was approximately 10 years ago, when FPGA vendors encountered with the same problem. FPGA grew slowly in comparison with the software development area.

> *How can we expand FPGA business and make FPGA platform more attractive for software developers?*

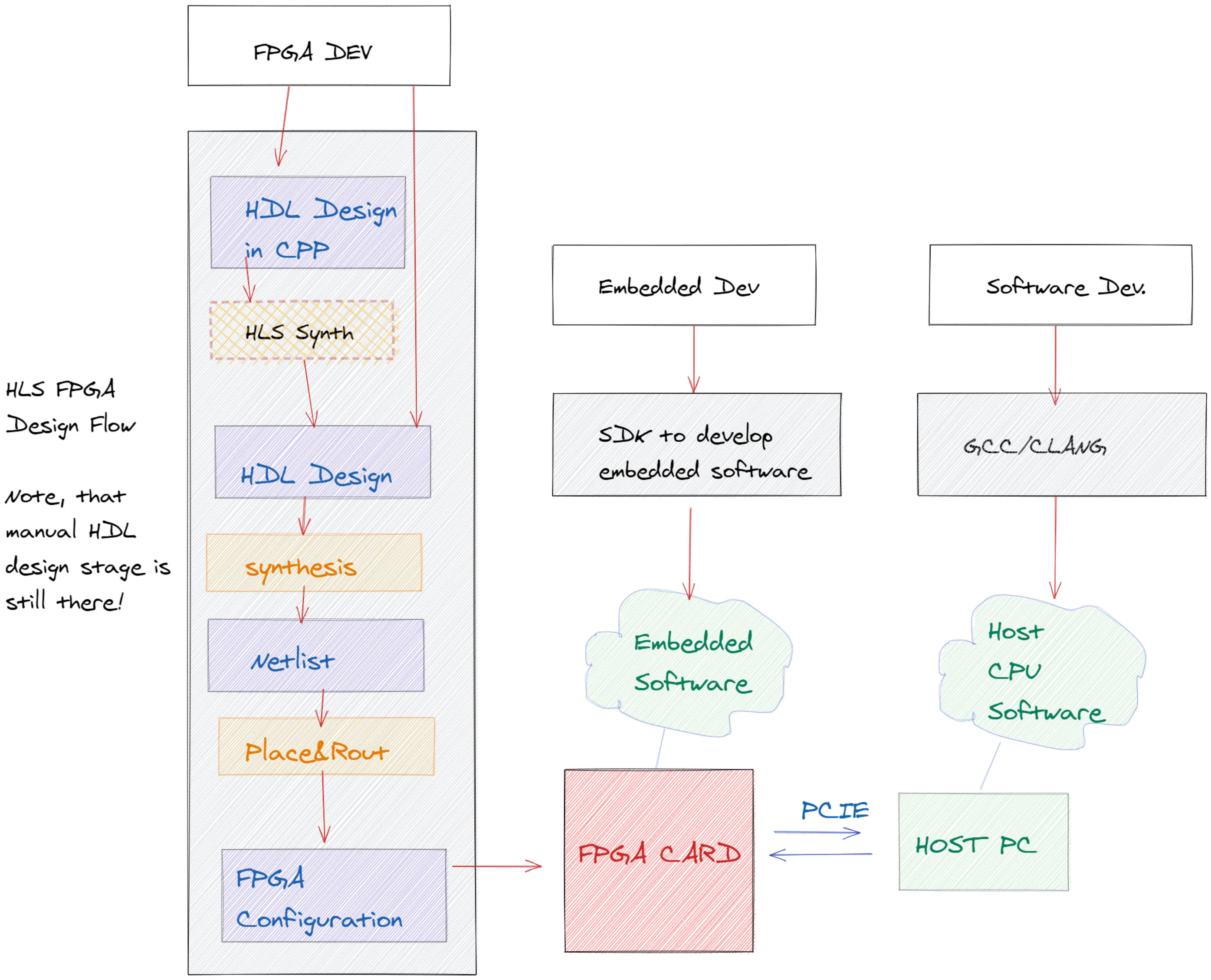
Classical FPGA design flow has several problems

- HDL languages are too weak, has very low level of abstractions, which reduces code quality and code reusability. Let's say you will be forced to develop neural networks in Pascal.
- HDL development process is very time-consuming. Proper design simulations takes hours, synthesis and place and route stages may take ~6 hours. So the loop of "change-feedback" can be counted in days.
- There aren't many FPGA developers on the market, it's hard to grow and hire more people.

There are two ways of how can you make FPGA design process faster and easier:

- HLS Approach. Use High Level programming language (Python, Scala, Haskell, CPP) to generate HDL. This a tool for FPGA developers mostly and partly for software devs.

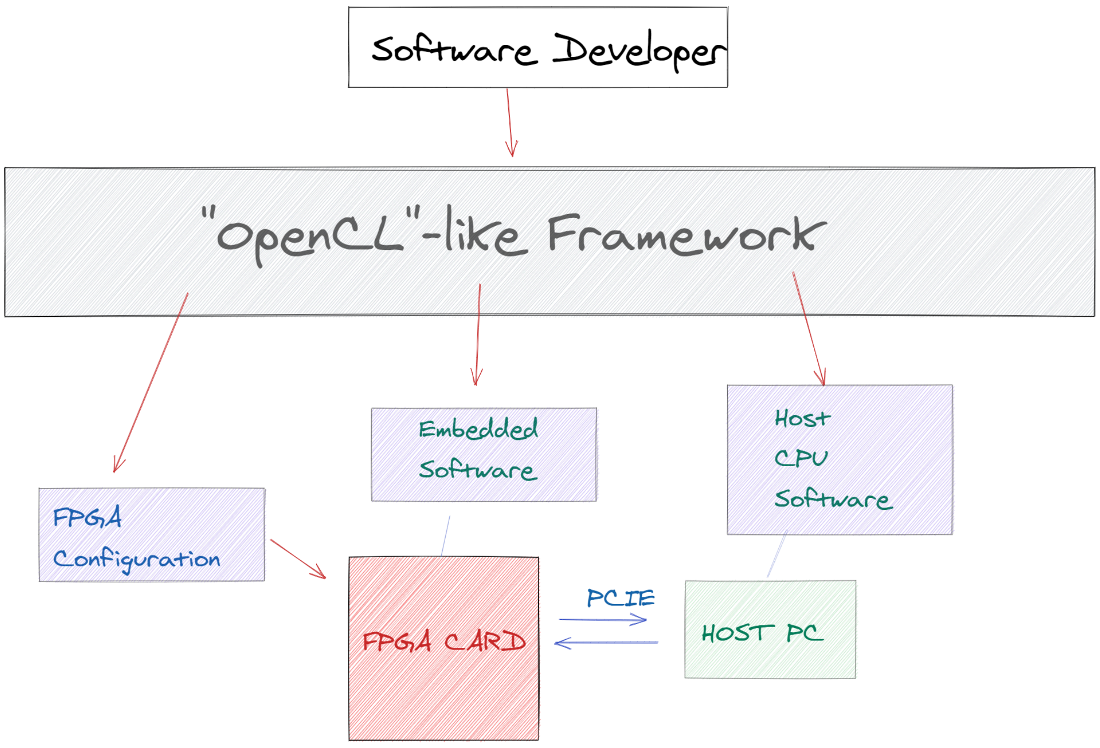
- OpenCL-like framework. Use FPGA directly from software development tools as an accelerator card. Both host program and FPGA configuration will be developed in the same flow, so this a strong software developer oriented way.Note that it's not necessary should be Open



Intel heavily Invested into OpenCL heterogeneous programming framework, as they have very wide hardware portfolio - CPUs(x86), GPUs, NPUs, FPGAs and SoC-FPGAs. So Intel crucially needed single tool to rule them all.

Xilinx is still pure FPGA/FPGA SoC oriented company and focused on HLS, tool for FPGA developer. Xilinx HLS works better than Intel. But Xilinx also started SDAccel tool - OpenCL approach for Xilinx FPGA cards.

Starting 2019.2, Xilinx SDK, SDSoC™ and SDAccel™ development environments are unified into an all-in-one Vitis™ unified software platform for application acceleration and embedded software development. There will be no 2019.2 or future releases of Xilinx SDx tools.



There are 2 issues with OpenCL framework itself

- OpenCL concept is still comparable rare. Community and support is still dramatically slower than proprietary NVidia CUDA framework.
- OpenCL still too level for software developer, requires from them to go deeper into FPGA design issue to perform optimizations. But on the other hand RTL developers consider OpenCL as too high, it's hard to use classes to define triggers and RAM blocks.

Can we do something better here? Intel said "Yes, we can"

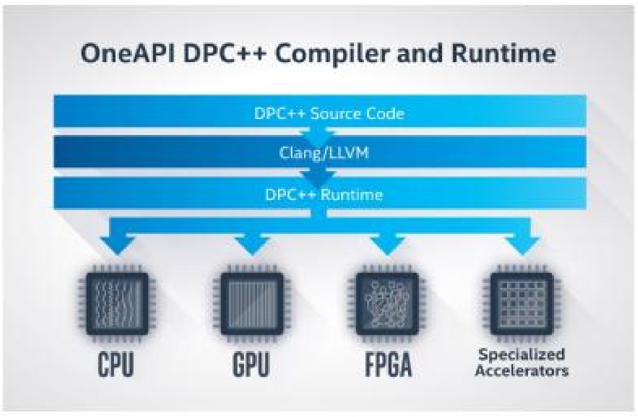
At the end November 2019 Intel initially released OneAPI specification

> *oneAPI is a cross-industry, open, standards-based unified programming model that delivers a common developer experience across accelerator architectures—for faster application performance, more productivity, and greater innovation.*

OneAPI uses Data Parallel C++ (DPC++) and SYCL

> *SYCL (pronounced sickle) is a royalty-free, cross-platform abstraction layer that builds on the underlying concepts, portability and efficiency of OpenCL that enables code for heterogeneous processors to be written in a single-source style using completely standard C++. SYCL single-source programming enables the host and kernel code for an application to be contained in the same source file, in a type-safe way and with the simplicity of a cross-platform asynchronous task graph.*

> *Data Parallel C++ (DPC++) is a high-level language designed for data parallel programming productivity. It is based on C++ for broad compatibility and uses common, familiar C and C++ constructs. The language seeks to deliver performance on par with other compiled languages, such as standard C++ compiled code, and uses C++ class libraries to allow the compiler to interpret the code and run on various supported architectures.*
> *DPC++ is based on SYCL* from the Khronos* Group to support data parallelism and*



heterogeneous programming. In addition, Intel is pursuing extensions to SYCL with the aim of providing value to customer code and working with the standards organization for adoption. For instance, the DPC++ language includes an implementation of unified shared memory to ease memory usage between the host and the accelerators. These features are being driven into a future version of the SYCL language.

So roughly speaking, DPC++ language includes

- ISO C++ 11
- Khronos SYCL
- Community extensions, i.e. Intel® oneAPI DPC++ Library

See OpenAPI Overview video

Intel OneAPI Toolkit, includes but not limited to

- Intel® oneAPI DPC++ Compiler.
- Migration tool from CUDA to OneAPI with DPC compatibility tool and from OpenCL to DPC++ .
- VTune Profiler
- Intel Adviser - code analyzer, find things good for accelerator

Xilinx Vitis Tool on the very first look may be considered as OneAPI equivalent, but it's not. Before 2019Q2 Xilinx supported wide range of own software oriented tools, but all of them were unified in Vitis. Vitis tool also included Vitis HLS. Difference from Vivado HLS is that Vivado HLS produces only RTL code for you, whenever Vitis HLS

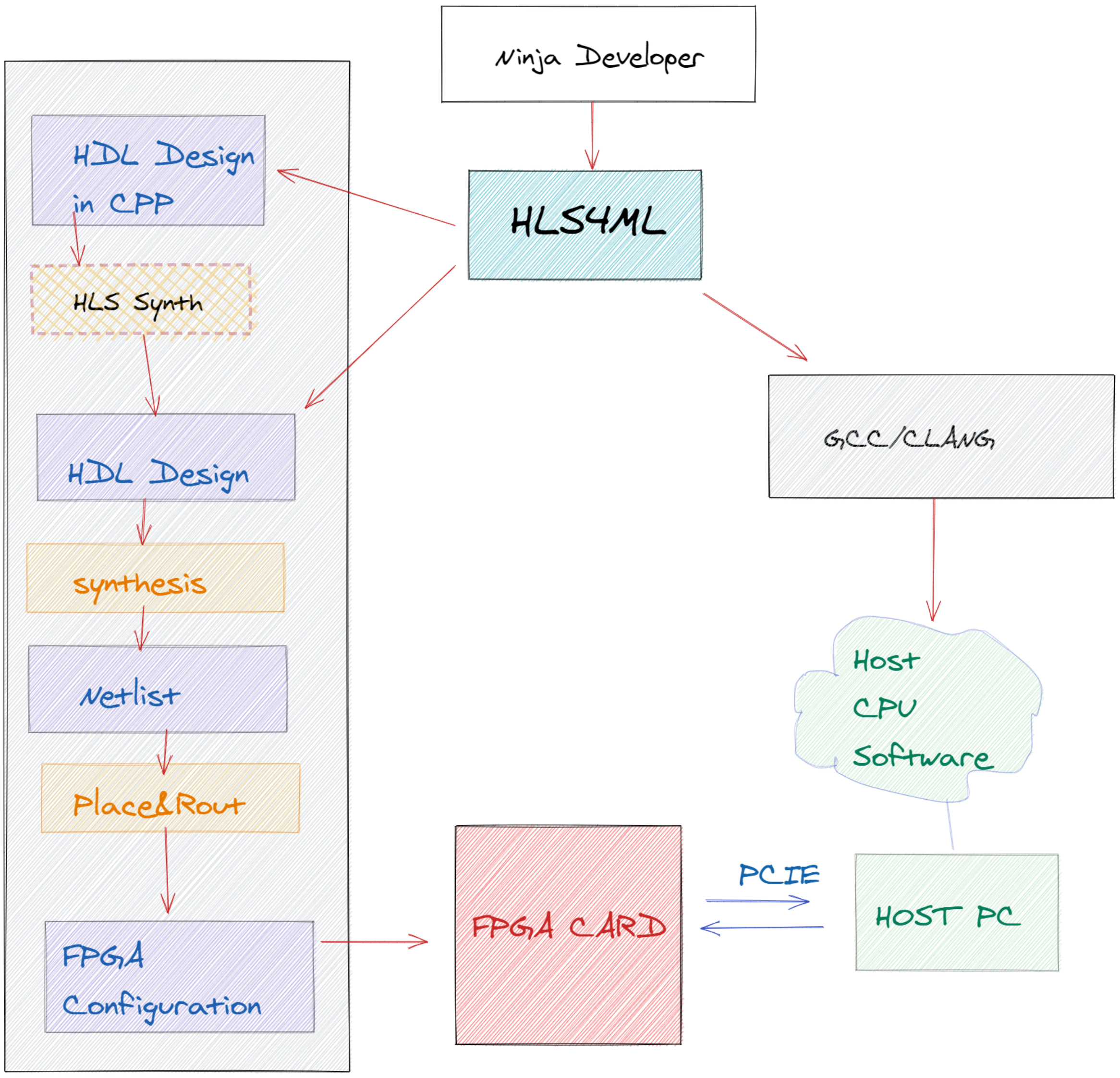
Prime user	For what?	Intel	Xilinx
FPGA developer	Design FPGA from scratch to end	Quartus	Vivado
Software developer	Design single IP Core for FPGA	Quartus HLS	Vivado HLS
Software developer	Use FPGA as an accelerator for my software	Intel OpenCL	SDAccel*
Embedded developer	Use FPGA SoC to develop code for hard of soft processor system	Intel FPGA SDK	Xilinx SDK*
Software developer	Use primarily Intel hardware platforms (CPU, GPU, NPU, FPGA) in a common environment for software acceleration	Intel OneAPI	---
Software developer	Use Xilinx hardware platforms (Alveo cards) for software acceleration	--	Xilinx Vitis

* - deprecated tools

The place of HLS4ML

HLS FPGA Design Flow

Note, that manual HDL design stage is still there!



Ninja Developer
HLS4ML
HDL Design in CPP
HLS Synth
HDL Design
synthesis
Netlist
Place&Rout
FPGA Configuration
GCC/CLANG
Host CPU Software
FPGA CARD
PCIE
HOST PC