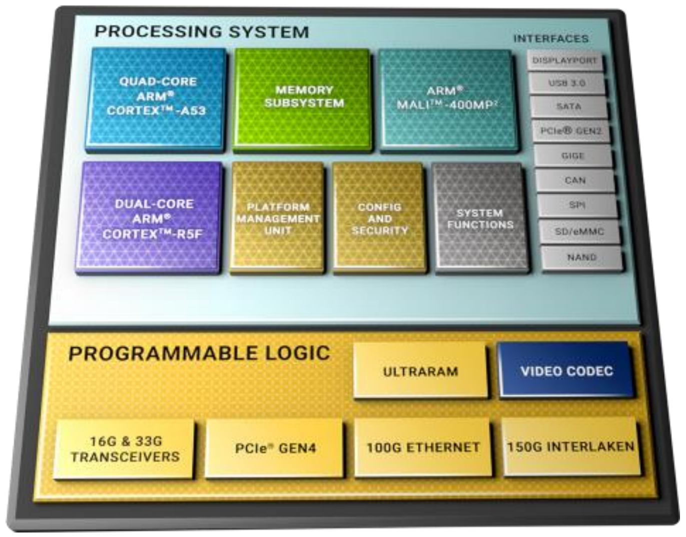
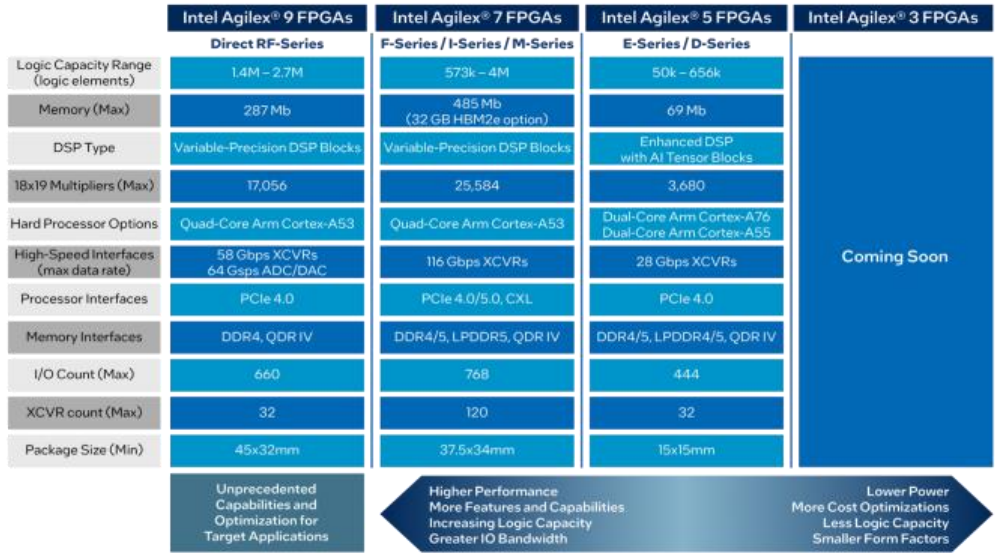
Programmable Logic for Future Frontends

FPGA goes System on Chip





	Intel Agilex® 9 FPGAs	Intel Agilex® 7 FPGAs	Intel Agilex® 5 FPGAs	Intel Agilex® 3 FPGAs
	Direct RF-Series	F-Series / I-Series / M-Series	E-Series / D-Series	
Logic Capacity Range (logic elements)	1.4M – 2.7M	573k – 4M	50k – 656k	Coming Soon
Memory (Max)	287 Mb	485 Mb (32 GB HBM2e option)	69 Mb	
DSP Type	Variable-Precision DSP Blocks	Variable-Precision DSP Blocks	Enhanced DSP with AI Tensor Blocks	
18x19 Multipliers (Max)	17,056	25,584	3,680	
Hard Processor Options	Quad-Core Arm Cortex-A53	Quad-Core Arm Cortex-A53	Dual-Core Arm Cortex-A76 Dual-Core Arm Cortex-A55	
High-Speed Interfaces (max data rate)	58 Gbps XCVRs 64 Gsps ADC/DAC	116 Gbps XCVRs	28 Gbps XCVRs	
Processor Interfaces	PCIe 4.0	PCIe 4.0/5.0, CXL	PCIe 4.0	
Memory Interfaces	DDR4, QDR IV	DDR4/5, LPDDR5, QDR IV	DDR4/5, LPDDR4/5, QDR IV	
I/O Count (Max)	660	768	444	
XCVR count (Max)	32	120	32	
Package Size (Min)	45x32mm	37.5x34mm	15x15mm	

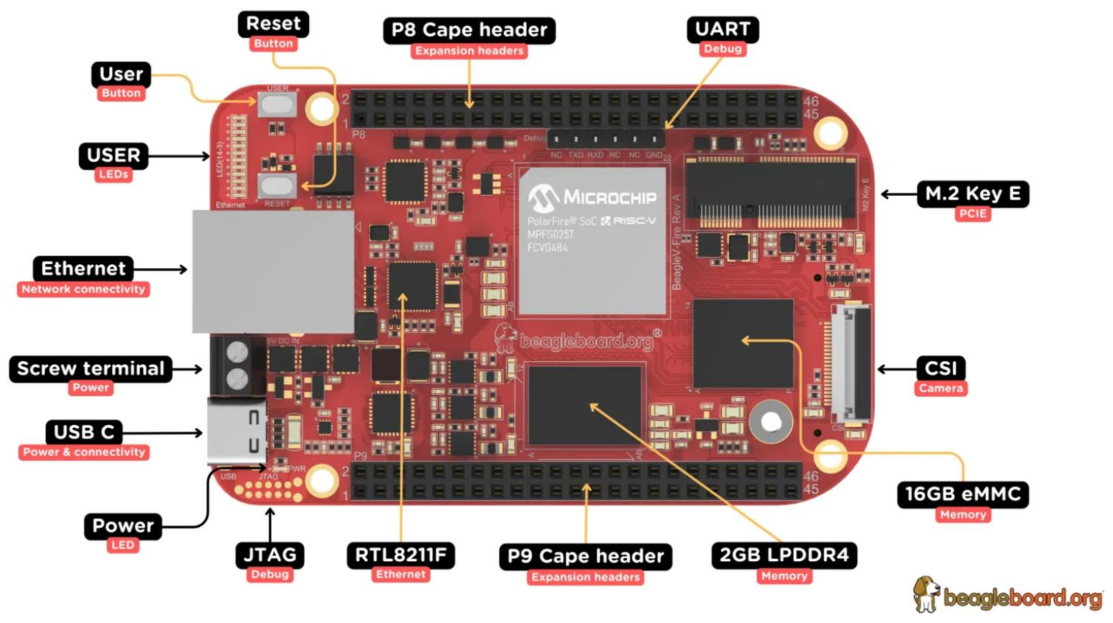
Unprecedented Capabilities and Optimization for Target Applications

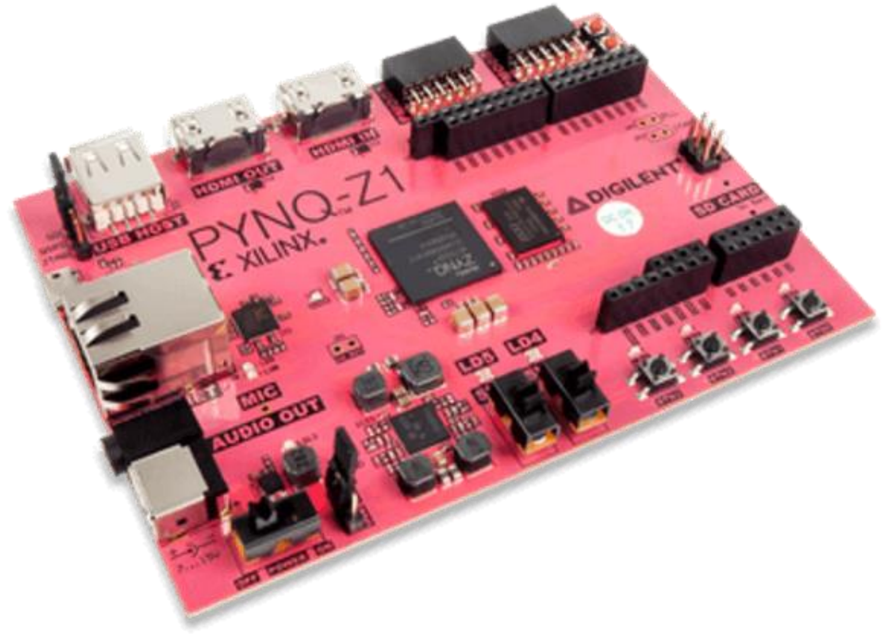
Higher Performance
More Features and Capabilities
Increasing Logic Capacity
Greater IO Bandwidth

Lower Power
More Cost Optimizations
Less Logic Capacity
Smaller Form Factors

- Integration of various ARM Cortex CPU cores as hard blocks alongside with programmable logic
- Initially the target was low cost application, with relatively small FPGA devices. Now top-of-the-line FPGAs implement CPU cores, network-on-chip, etc.
- The CPU cores may be operated as bear metal or running Linux/RTOS
- The programmable logic usually implements peripherals of CPUs or acceleration engine, but it may implement standalone functionality as well

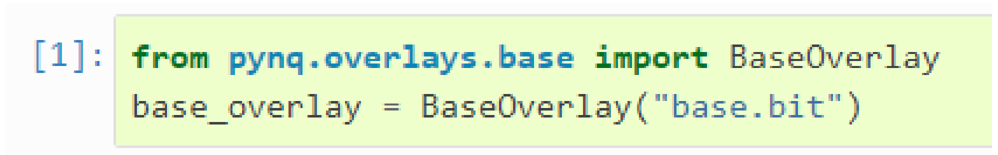
FPGA goes Arduino



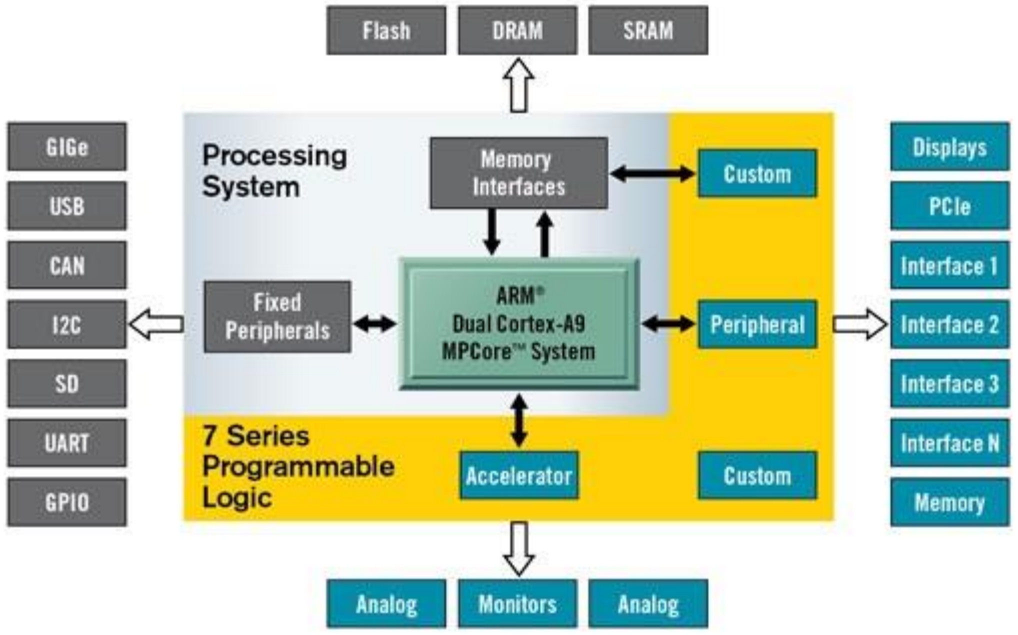


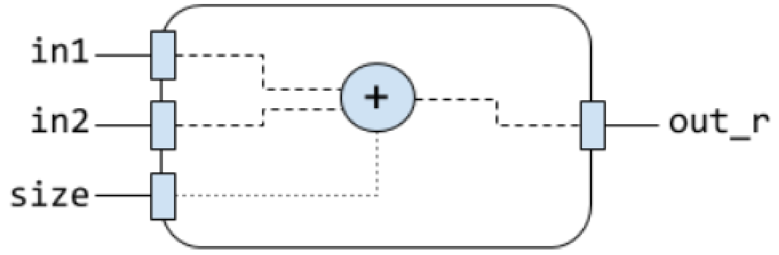
- There is a broad range of relatively low cost boards that mimic the established Arduino or RaspberryPi form factors
- Development may use less specific C or micropython programming framework; custom programmable logic cores still need HDL coding

FPGA goes Python

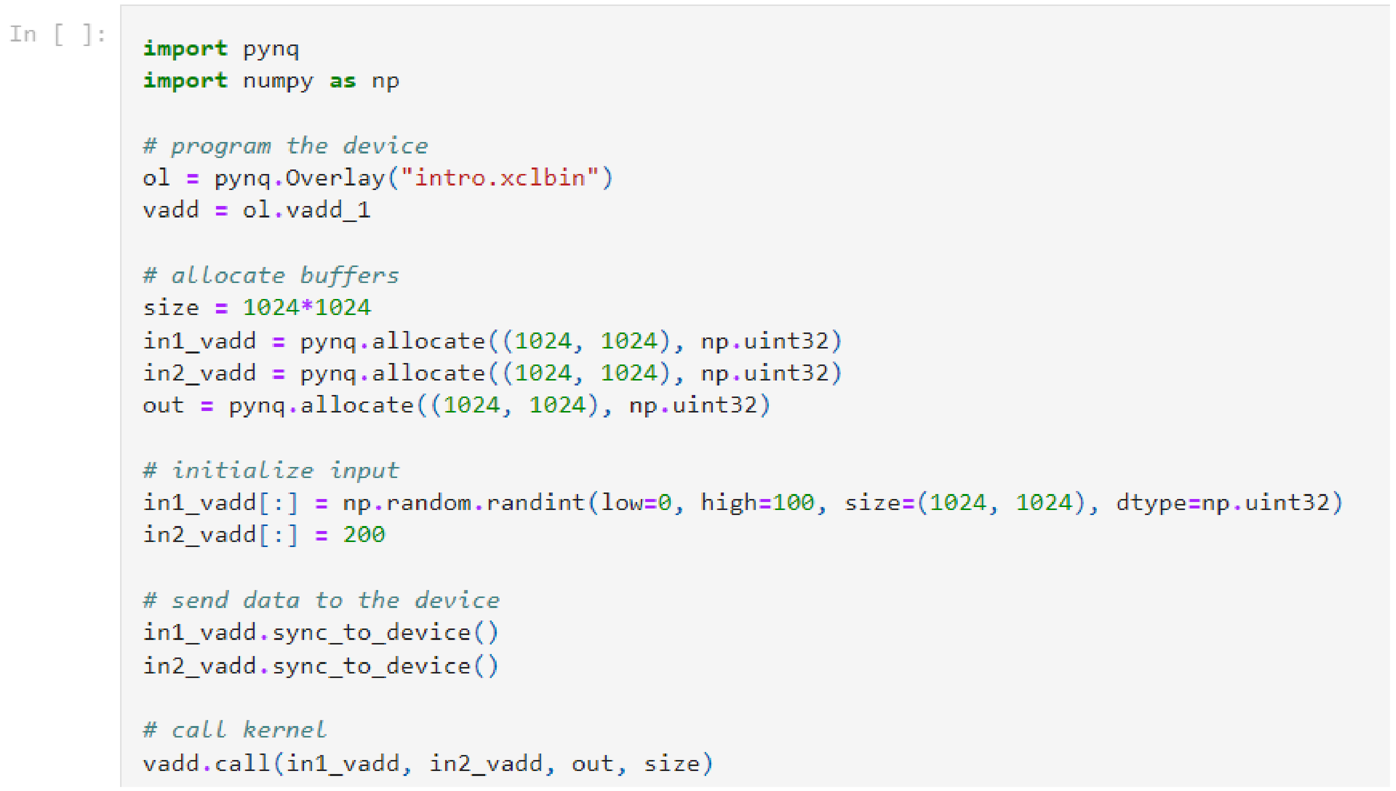


```
[1]: from pynq.overlays.base import BaseOverlay
     base_overlay = BaseOverlay("base.bit")
```





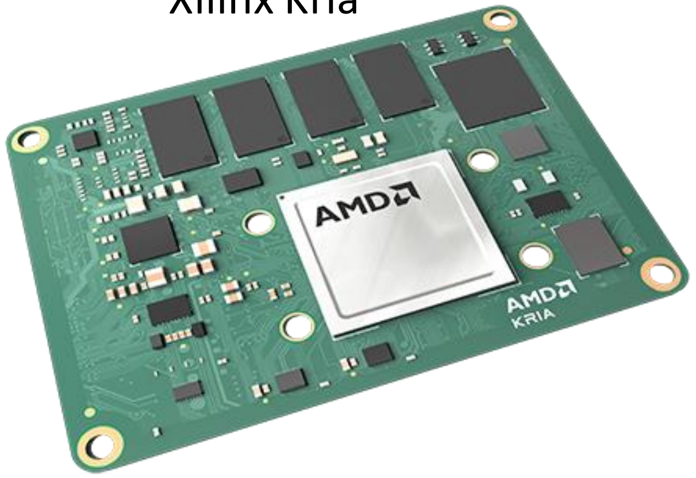
See below for a breakdown of the code.



```
In [ ]: import pynq
        import numpy as np

        # program the device
        ol = pynq.Overlay("intro.xclbin")
        vadd = ol.vadd_1

        # allocate buffers
        size = 1024*1024
        in1_vadd = pynq.allocate((1024, 1024), np.uint32)
        in2_vadd = pynq.allocate((1024, 1024), np.uint32)
        out = pynq.allocate((1024, 1024), np.uint32)

        # initialize input
        in1_vadd[:] = np.random.randint(low=0, high=100, size=(1024, 1024), dtype=np.uint32)
        in2_vadd[:] = 200

        # send data to the device
        in1_vadd.sync_to_device()
        in2_vadd.sync_to_device()

        # call kernel
        vadd.call(in1_vadd, in2_vadd, out, size)
```

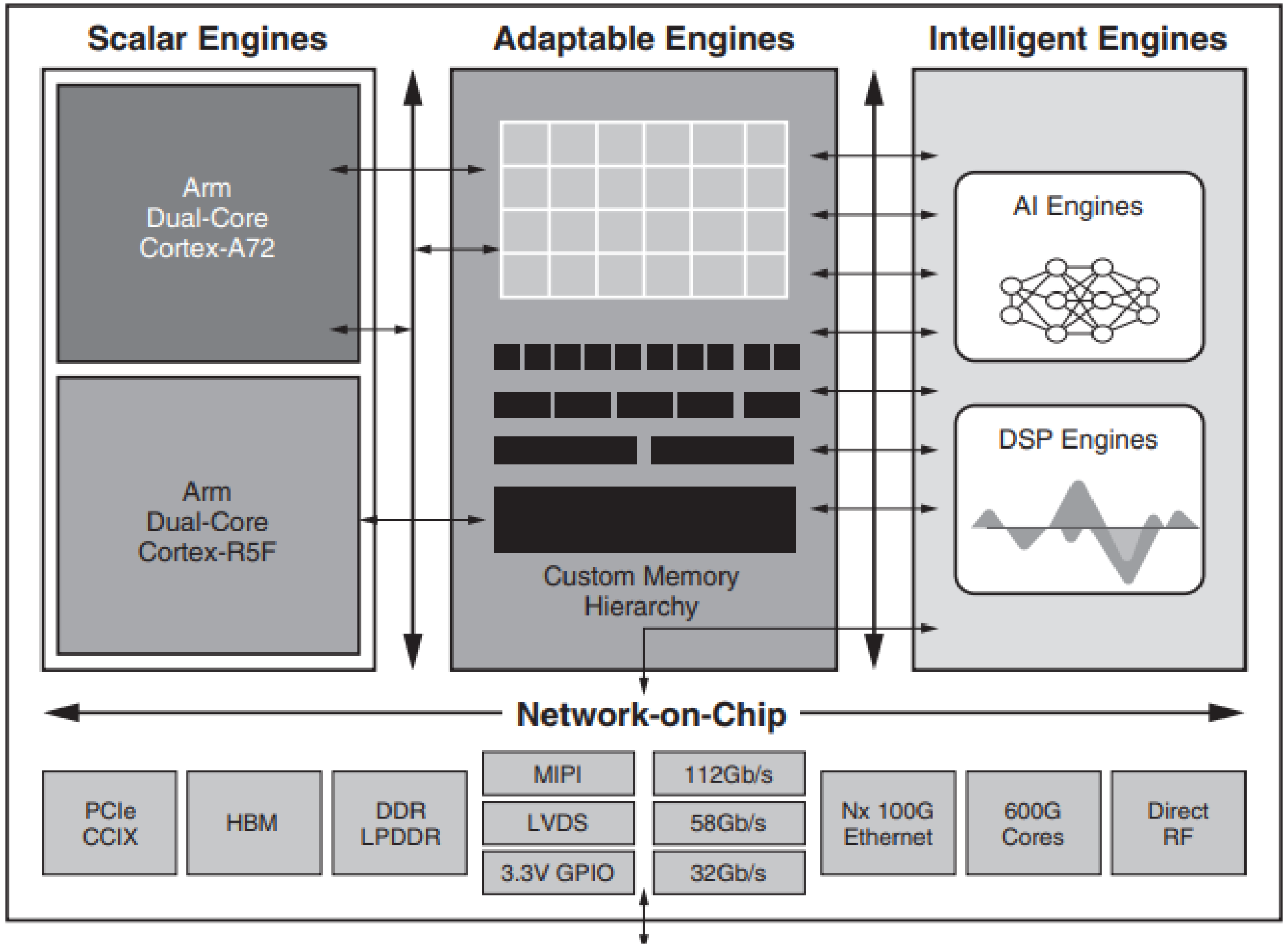
FPGA goes SOM

Xilinx Kria

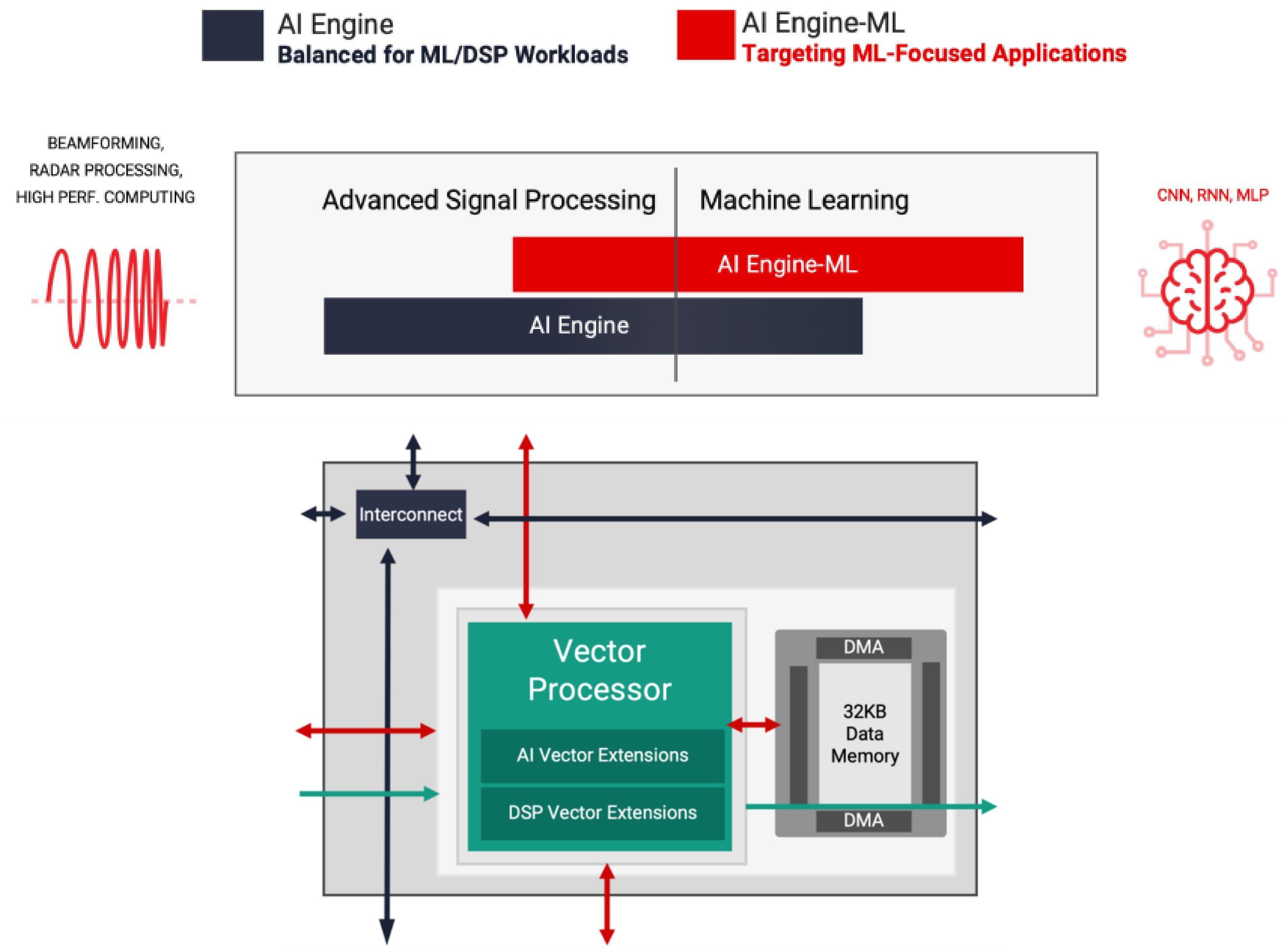


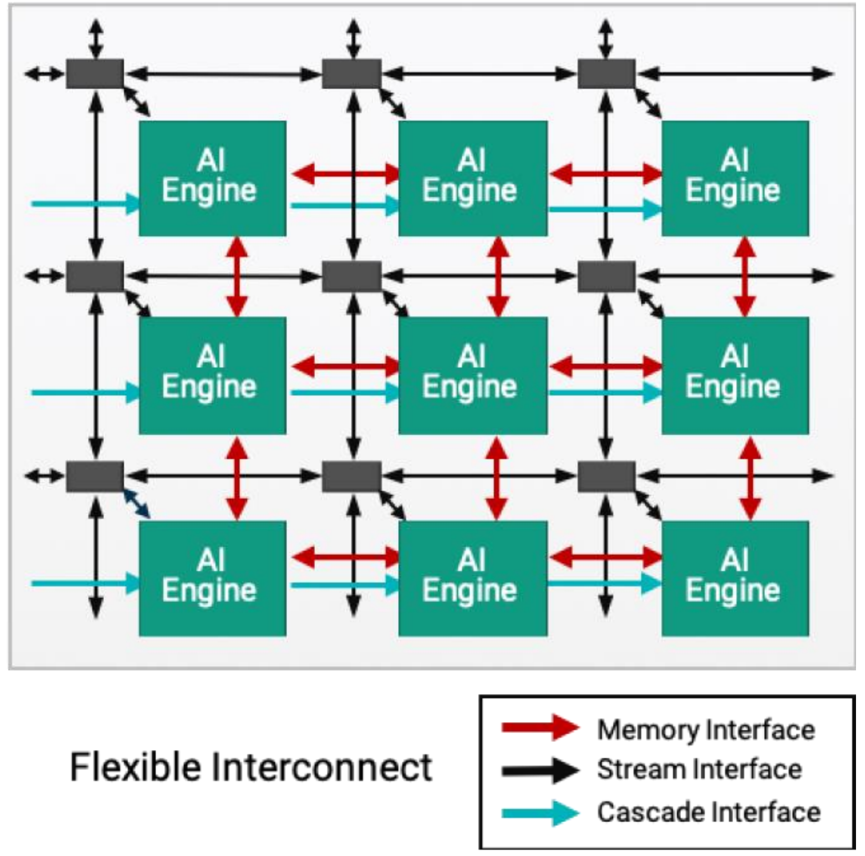
- Full FPGA System on Module, including RAM, PLLs, ETH Phy, power regulators, etc.
- Large choice of devices: SOCs or pure FPGAs, high or low cost,..
- There is no common form factor among vendors

FPGA goes heterogenous

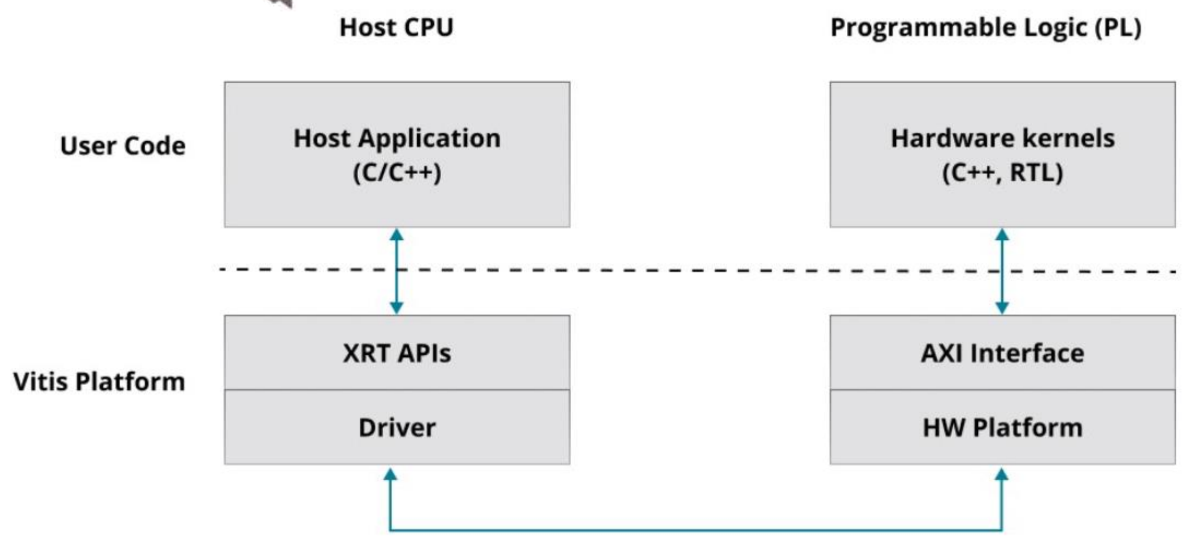


FPGA goes A.I./Vector Processor

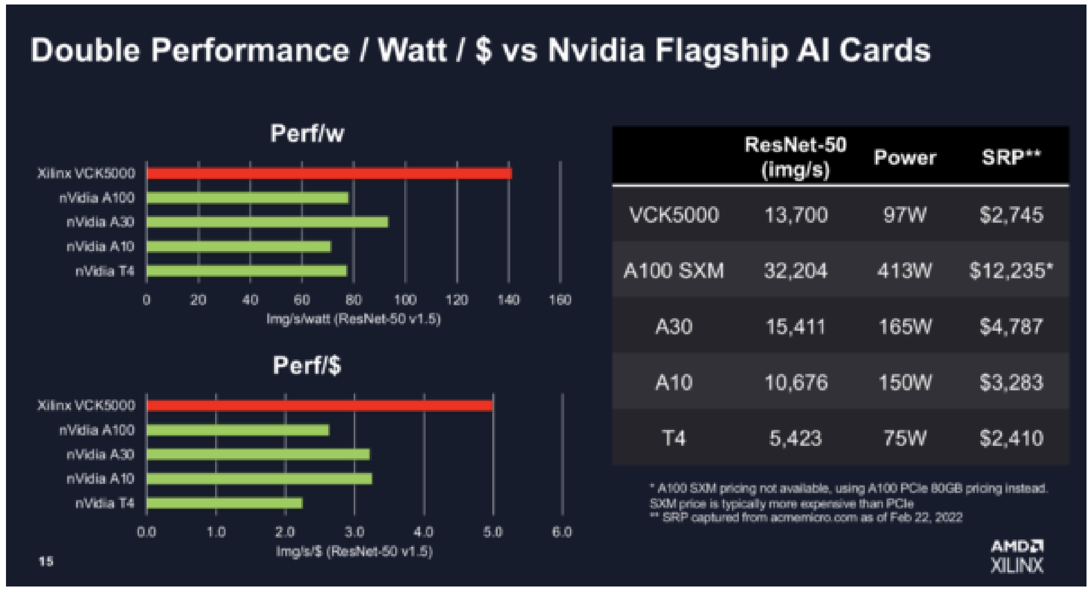




FPGA goes Hardware Accelerator



FPGA vs GPU



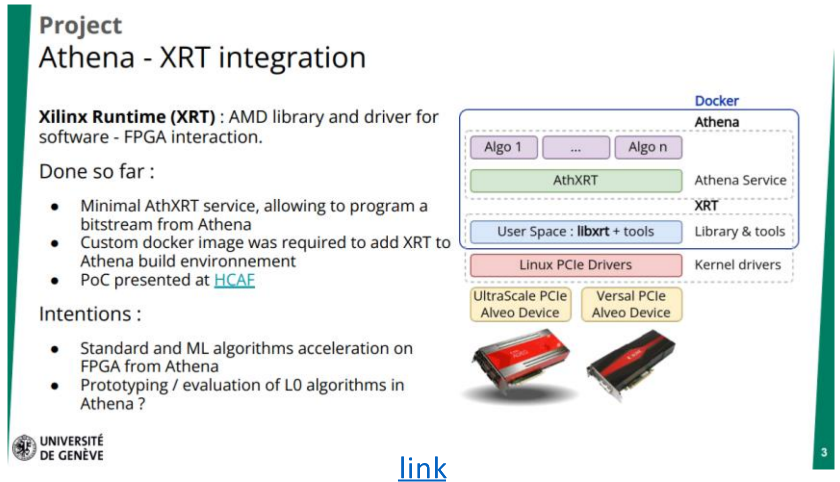
Double Performance / Watt / $ vs Nvidia Flagship AI Cards

	ResNet-50 (img/s)	Power	SRP**
VCK5000	13,700	97W	$2,745
A100 SXM	32,204	413W	$12,235*
A30	15,411	165W	$4,787
A10	10,676	150W	$3,283
T4	5,423	75W	$2,410

* A100 SXM pricing not available, using A100 PCIe 80GB pricing instead. SXM price is typically more expensive than PCIe
** SRP captured from acmemicro.com as of Feb 22, 2022

15

AMD XILINX



Project

Athena - XRT integration

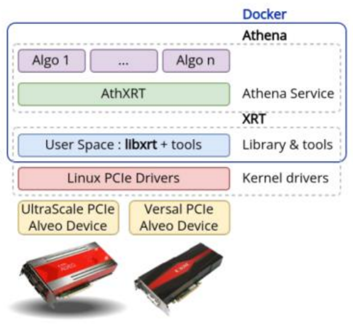
Xilinx Runtime (XRT) : AMD library and driver for software - FPGA interaction.

Done so far :

- Minimal AthXRT service, allowing to program a bitstream from Athena
- Custom docker image was required to add XRT to Athena build environnement
- PoC presented at HCAF

Intentions :

- Standard and ML algorithms acceleration on FPGA from Athena
- Prototyping / evaluation of L0 algorithms in Athena ?

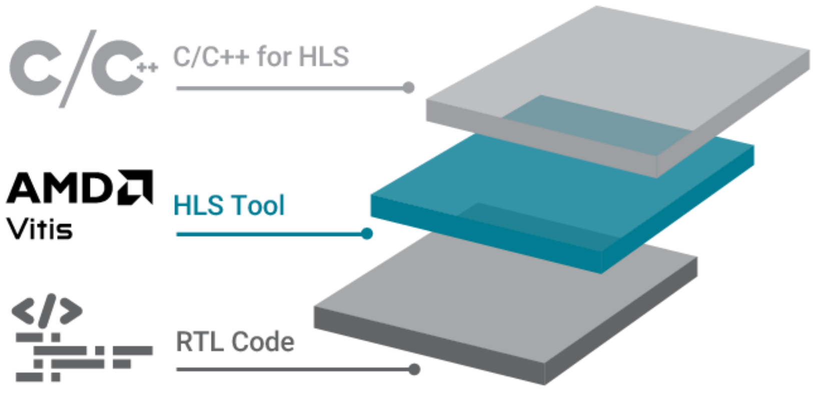


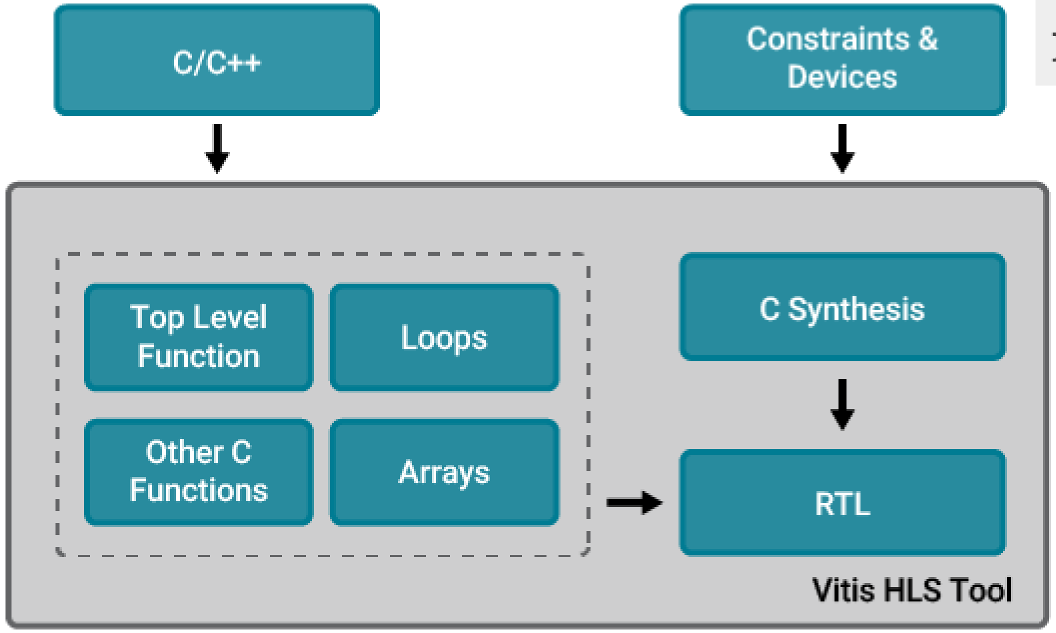
UNIVERSITÉ DE GENÈVE

3

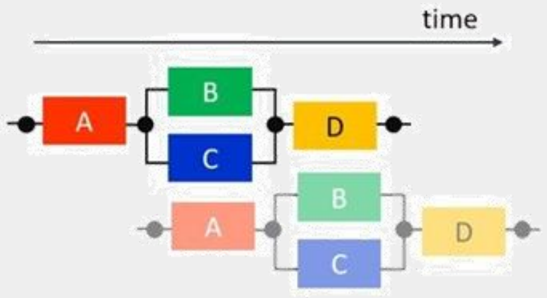
link

FPGA goes C++





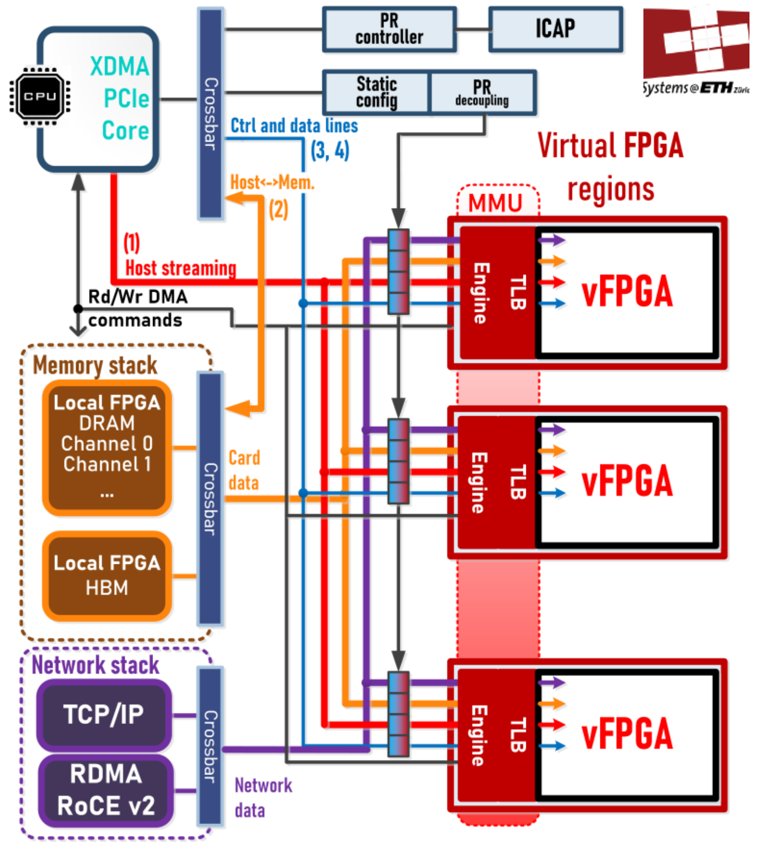
```
void foo(int data_in[N], int scale, int data_out1[N], int data_out2[N]) {
  int temp1[N];
  loop_1: for(int i = 0; i < N; i++) {
    #pragma HLS unroll region
    temp1[i] = data_in[i] * scale;
      loop_2: for(int j = 0; j < N; j++) {
        data_out1[j] = temp1[j] * 123;
      }
      loop_3: for(int k = 0; k < N; k++) {
        data_out2[k] = temp1[k] * 456;
      }
  }
}
```

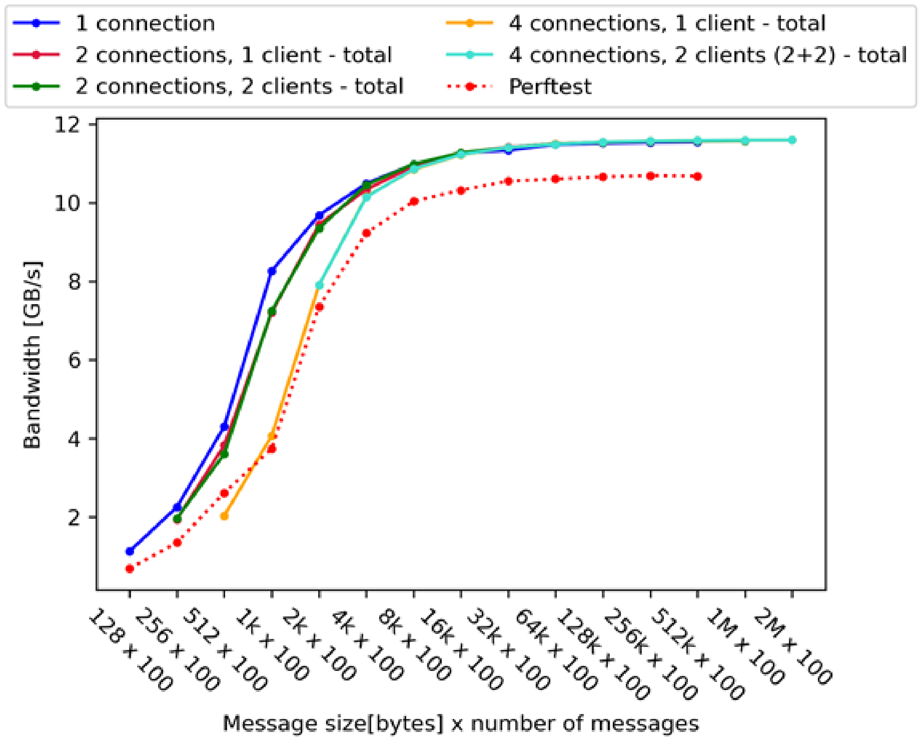


- Provides C++ to RTL code translation (Verilog/VHDL)
- Far more C++ developers out there than Verilog/VHDL
- Complex designs can be
- Most vendors (AMD, Intel,..) support HLS as development language

- Actual HLS implementation is not always consistent among vendors (sometimes also between tool versions also)
- Trully

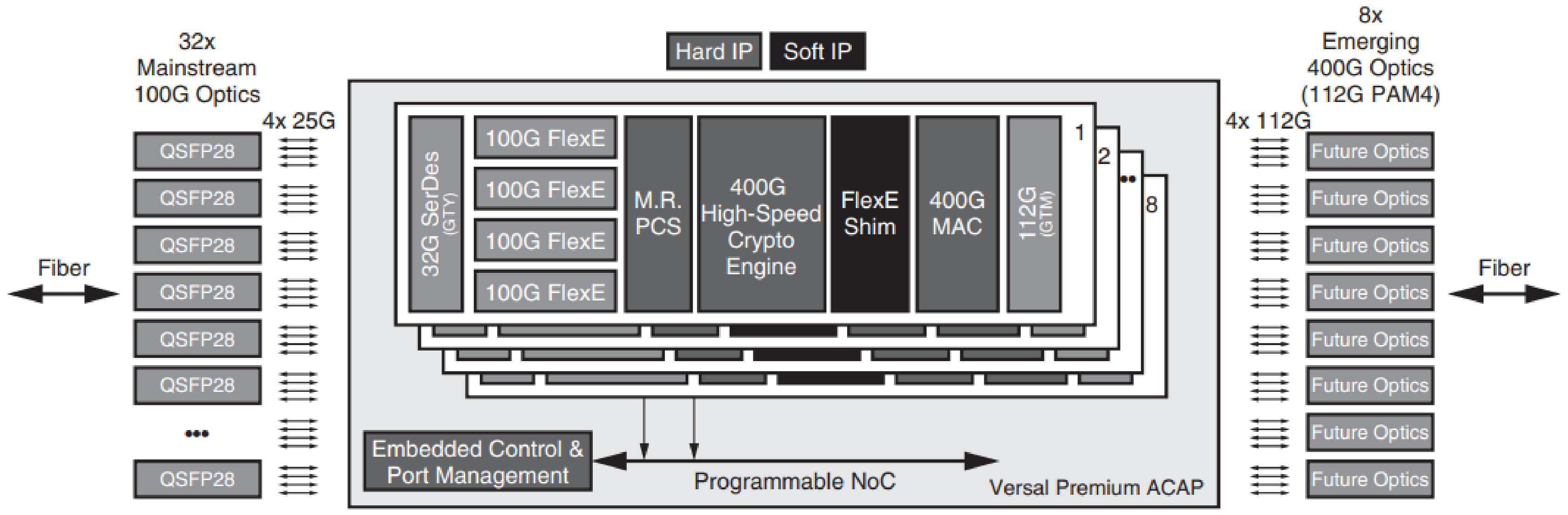
FPGA goes RDMA, TCP/IP





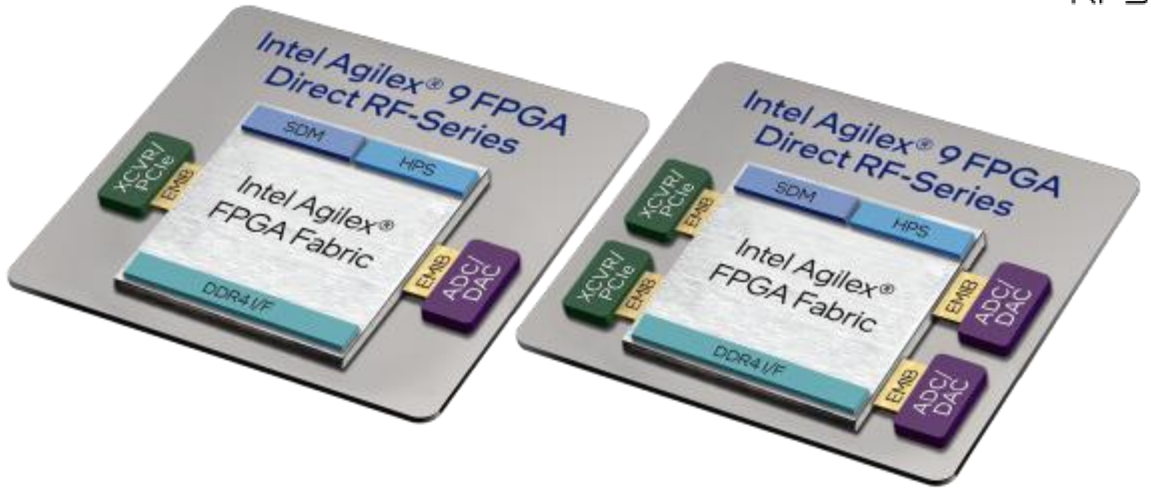
- Coyote - Framework providing operating system abstractions and a range of shared networking (*RDMA, TCP/IP*) - developed at ETH Zurich using *HLS* & *Verilog*
- DAQ-oriented FPGA-RDMA optimizations over 100Gb Ethernet Switched fabric @ IFIN-HH and Nickef
  - Data stream is sent directly to DAQ Computer RAM memory, without intervention of CPU.
  - Data can also be streamed directly to SSD via memory virtualization (further development needed)

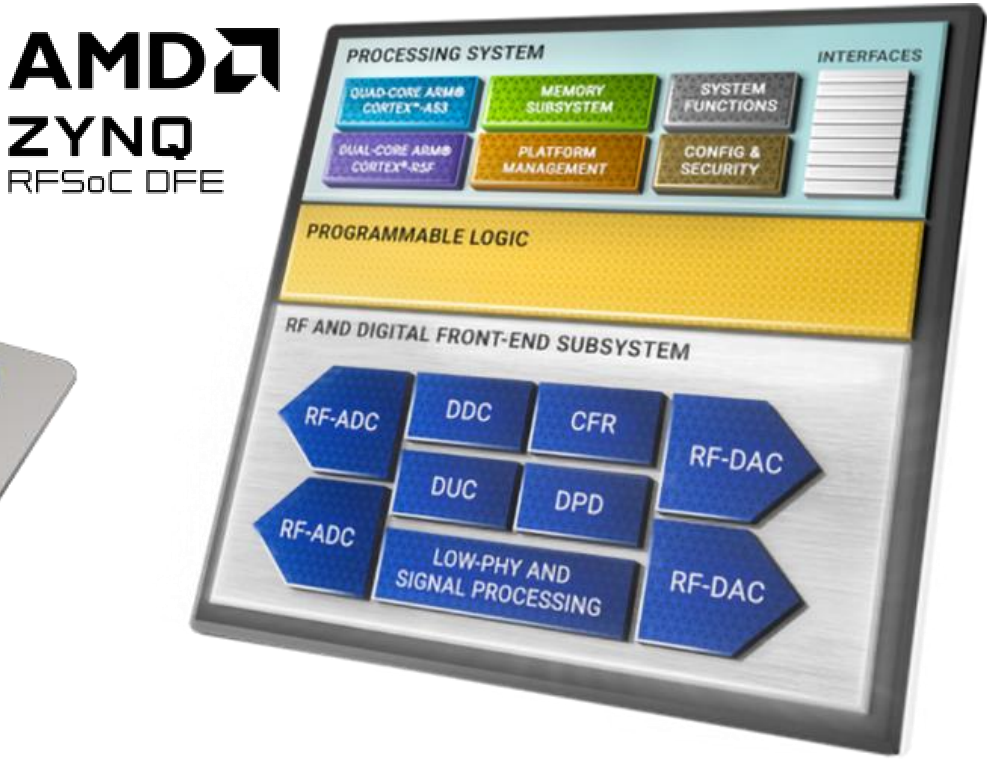
FPGA goes TBps



- High-end FPGA devices implement hard blocks for:
  - 100/400/600 GbEth
  - Gen4/Gen5 PCIe
  - Interlaken
  - etc.

FPGA goes Analog


Intel Agilex® 9 FPGA Direct RF-Series
SDM
HPS
XCVR/ PCIe
EMIB
Intel Agilex® FPGA Fabric
ADC/ DAC
DDR4 I/F
Intel Agilex® 9 FPGA Direct RF-Series
SDM
HPS
XCVR/ PCIe
EMIB
XCVR/ PCIe
EMIB
Intel Agilex® FPGA Fabric
EMIB
ADC/ DAC
EMIB
ADC/ DAC
DDR4 I/F



- Integrates ADCs and DACs up to 10 – 64 GSPS with 8 – 64 GHz bandwidth and 10-14 bit
- Primarily targeting RF applications

Thank You