AMD Tech Day

AI#1 - AMD Vitis AI: Its Capabilities

AVNET SILICA

Dimitrios Kolosov, Senior Field Application Engineer - FPGA Specialist

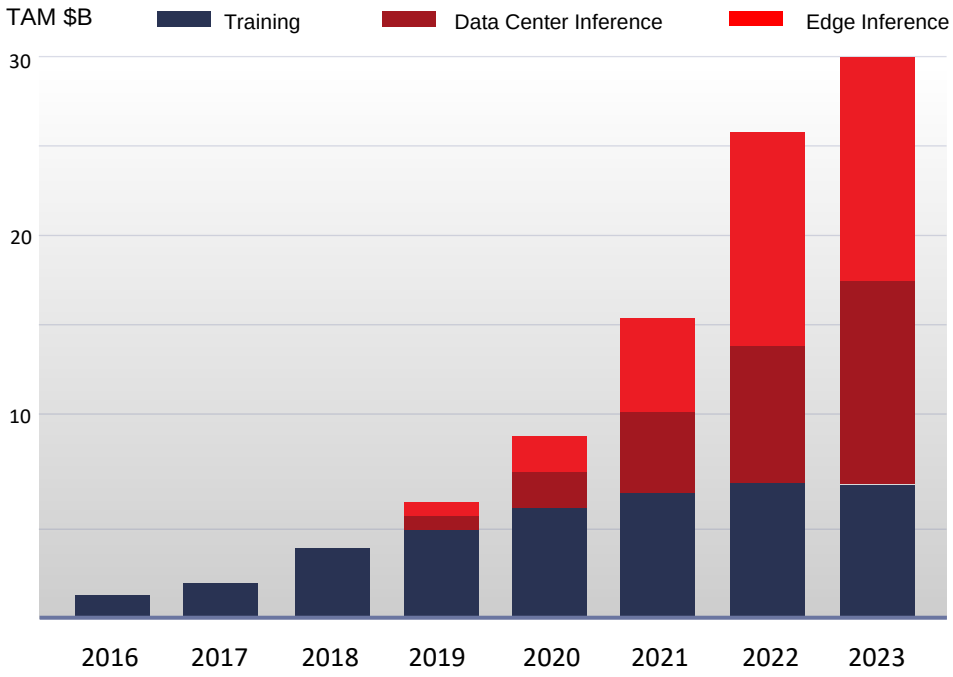
8th November, 2023

Agenda

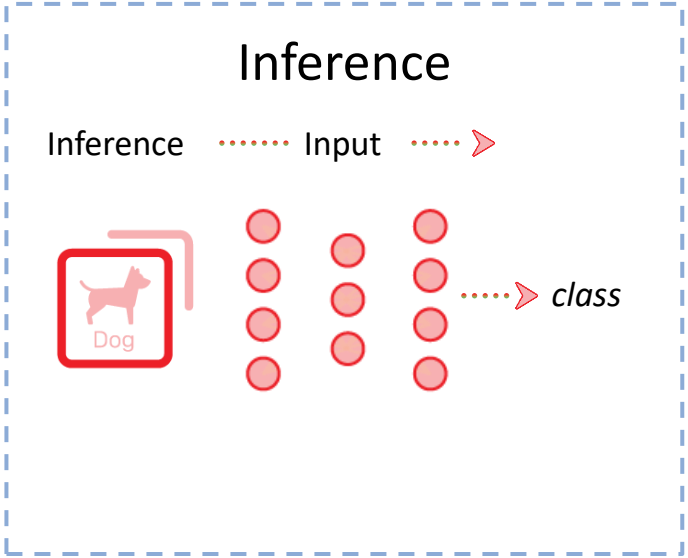
- Introducing AMD Vitis-AI
- Deeper Dive into AMD Vitis-AI
- Q&A

Introducing AMD Vitis-AI

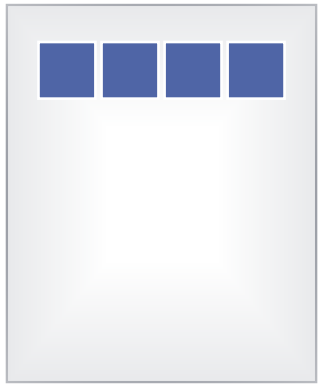
Inference Projected Growth



Barclays Research, Company Reports May 2018

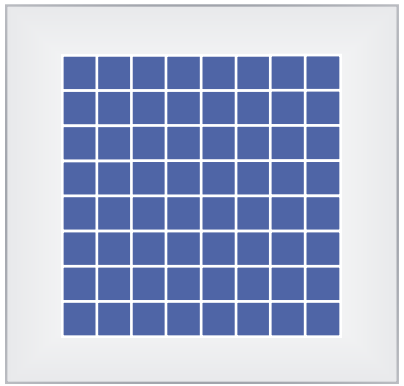


Era of Domain-Specific Architectures (DSAs)



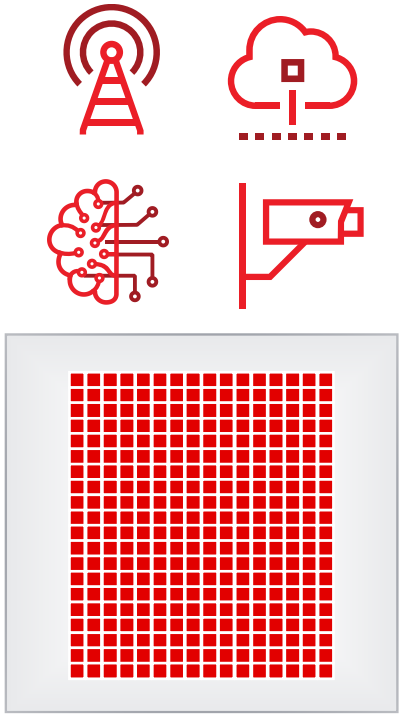
CPU

CISC → RISC → Multi-Core



Fixed HW Accelerators

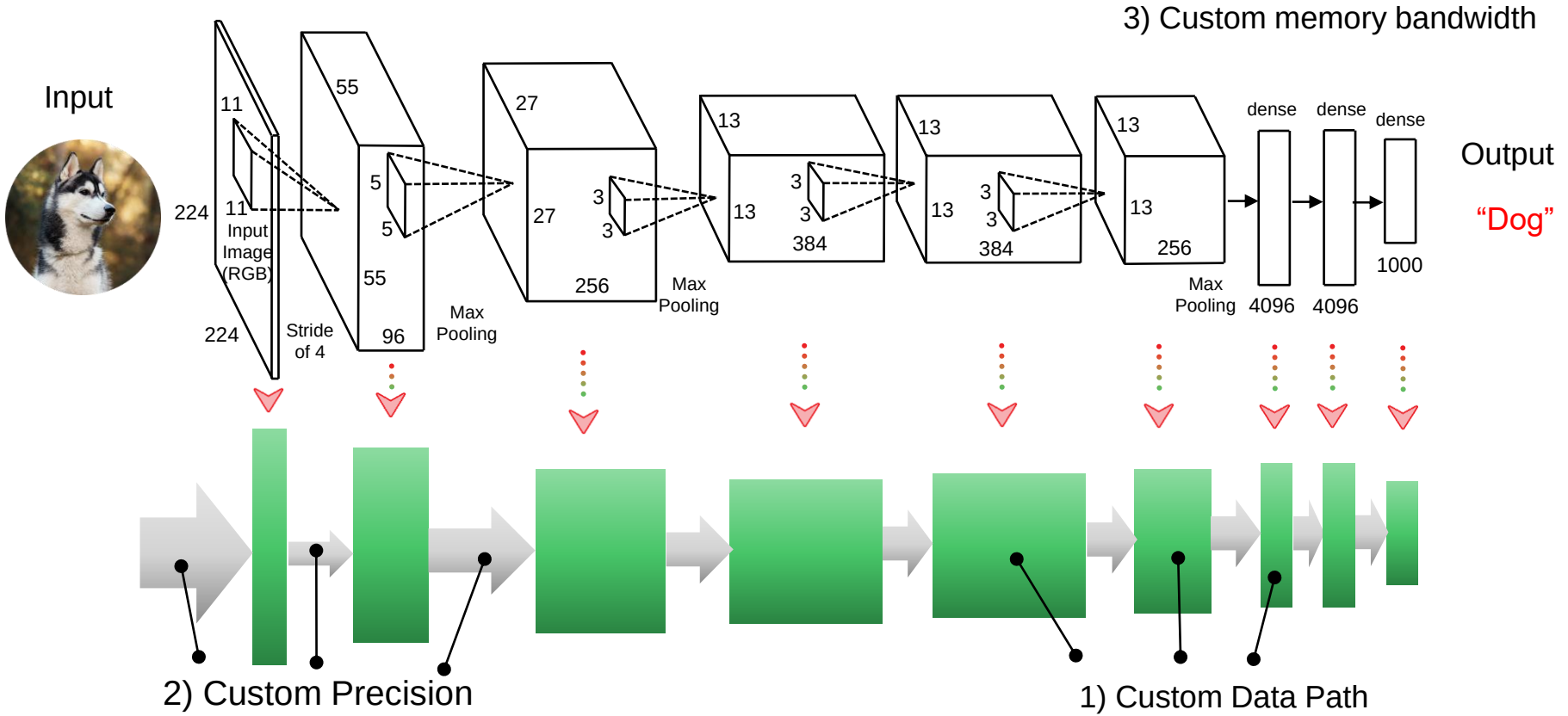
GPU, ASSP, ASIC



DSA

TPU, DLA
FPGA, Zynq, ACAP

What is a Domain-Specific Architecture?



AI Applications are Everywhere

APPLICATIONS

Classification

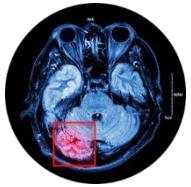
Object Detection

Segmentation

Speech Recognition

Recommendation Engine

Anomaly Detection



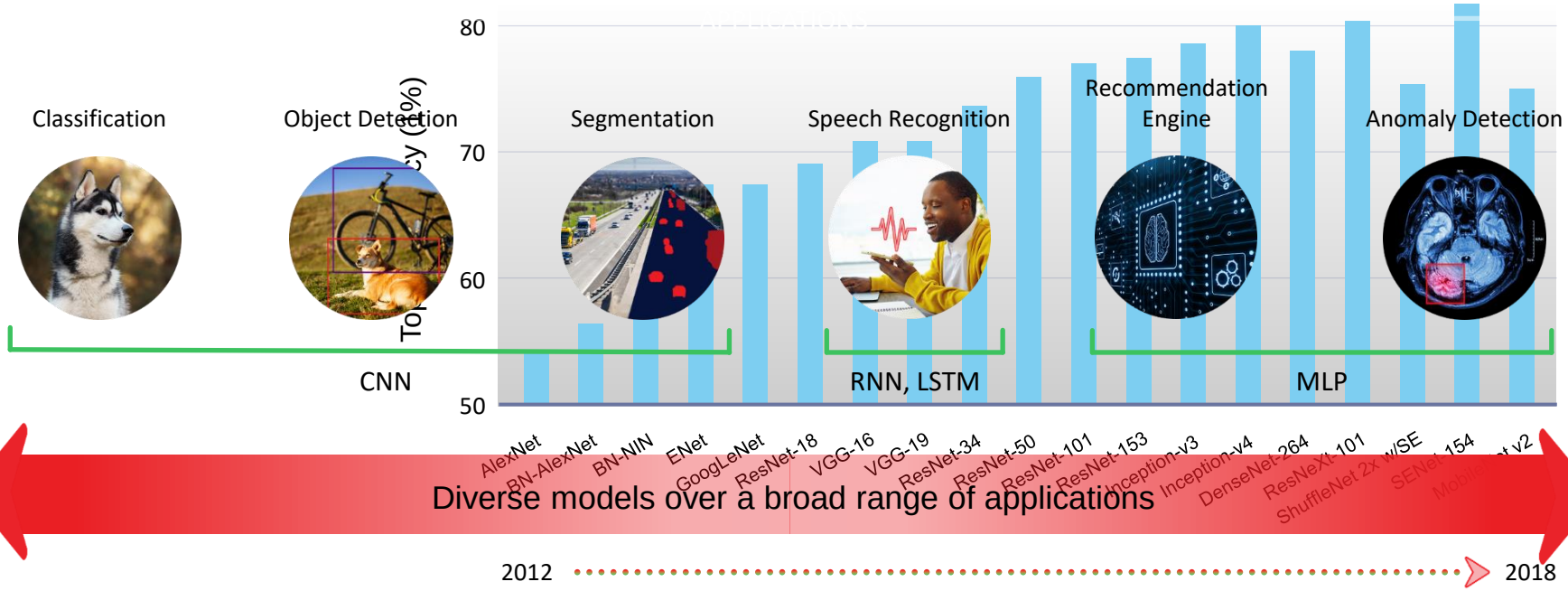
CNN

RNN, LSTM

MLP

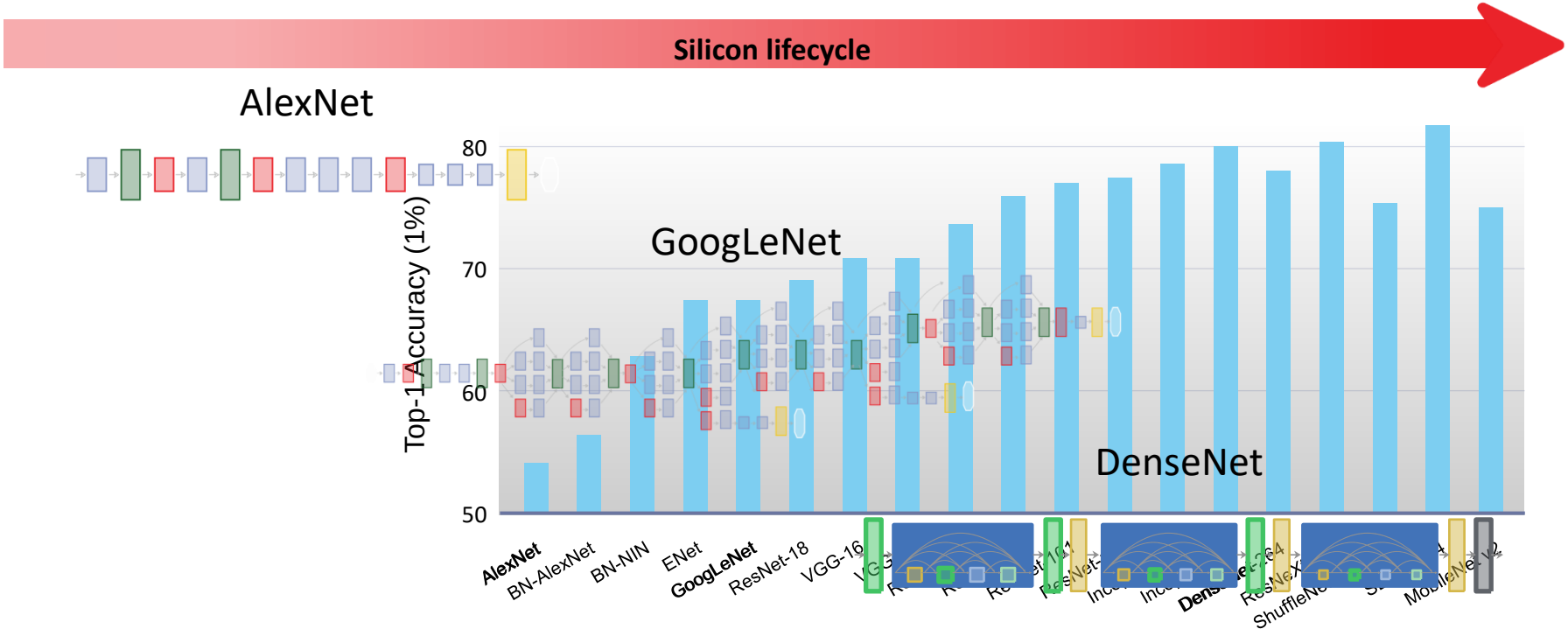
Diverse models over a broad range of applications

AI is Evolving Rapidly



Source:
https://arxiv.org/pdf/1605.07678.pdf https://arxiv.org/pdf/1608.06993.pdf
https://arxiv.org/pdf/1709.01507.pdf https://arxiv.org/pdf/1611.05431.pdf

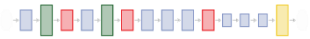
AI is Evolving Rapidly



Silicon Hardware Design Cycle Can't Keep Up with the Rate of AI Innovation

AMD Adaptive Hardware Enables *Dynamic* DSAs

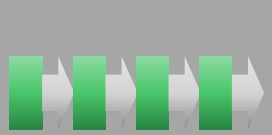
AlexNet

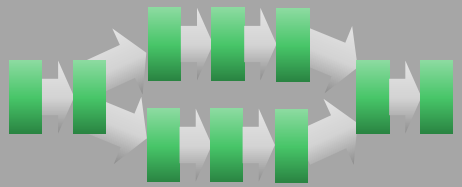


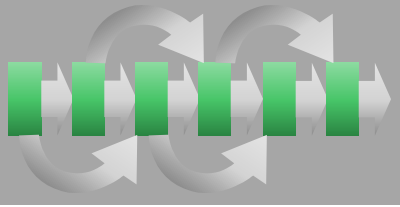
GoogLeNet

DenseNet

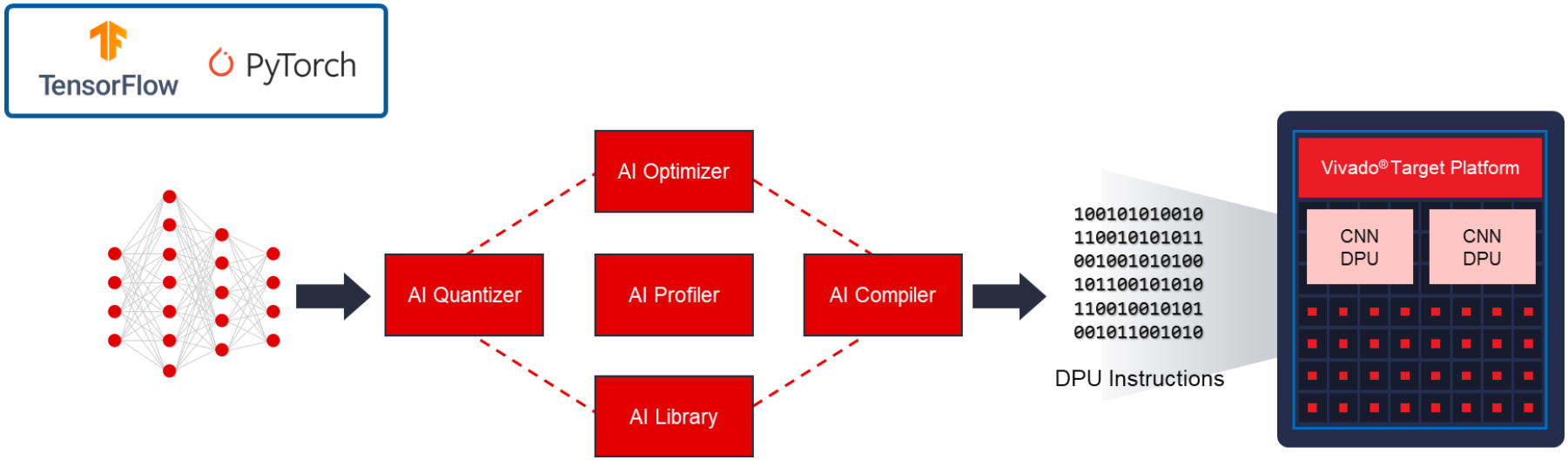
Adaptable devices allow DSA's to be updated without new Silicon Hardware







AMD Vitis AI - 1000-foot view



Enables All Developers to Build and Deploy AI to All Platforms

Embedded Developers

Enterprise Application Developers

Enterprise Infrastructure Developers

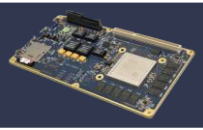
Data & AI Scientists

Build

XILINX VITIS AI

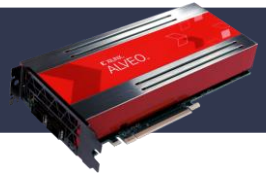
XILINX VITIS

Deploy



Zynq

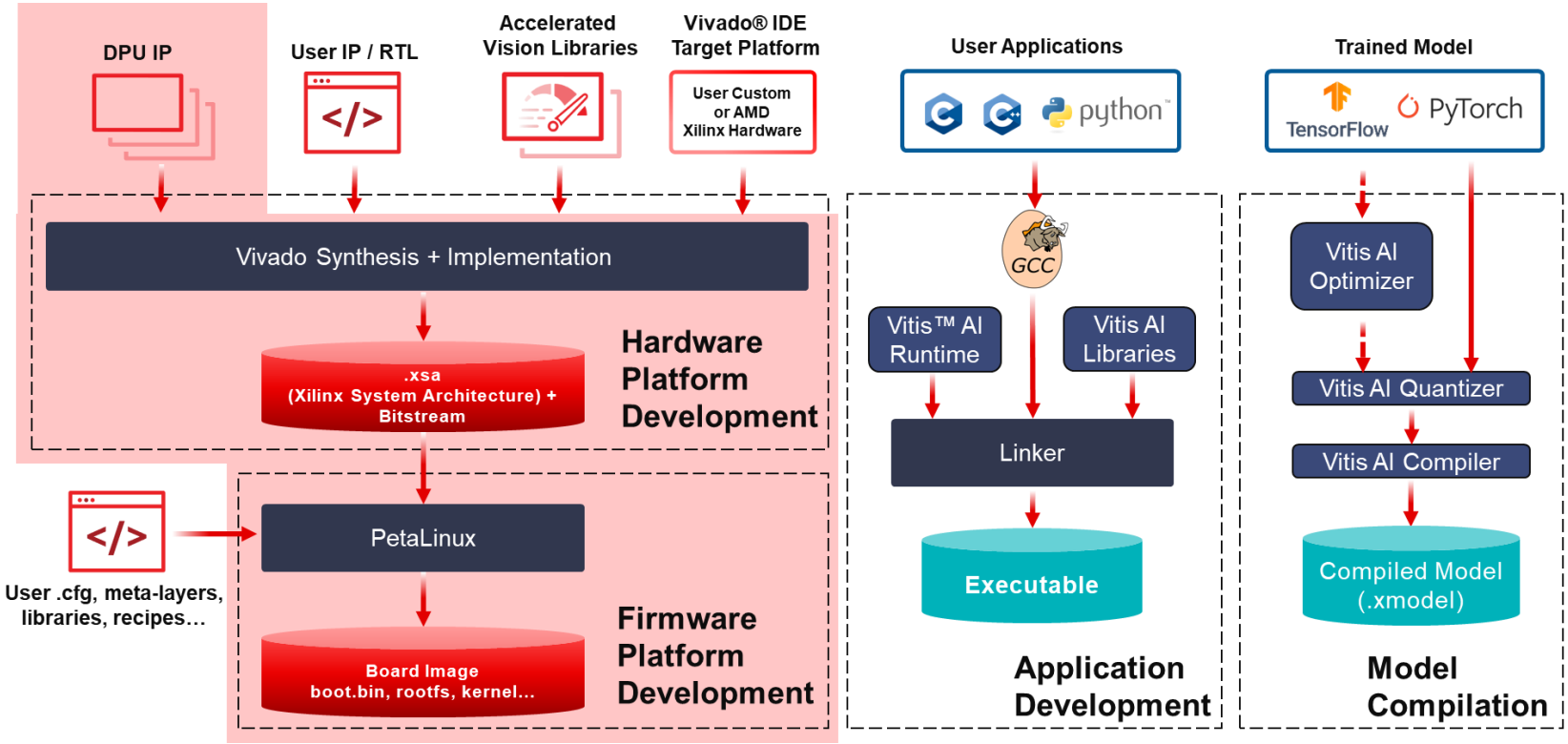
Ultrascale



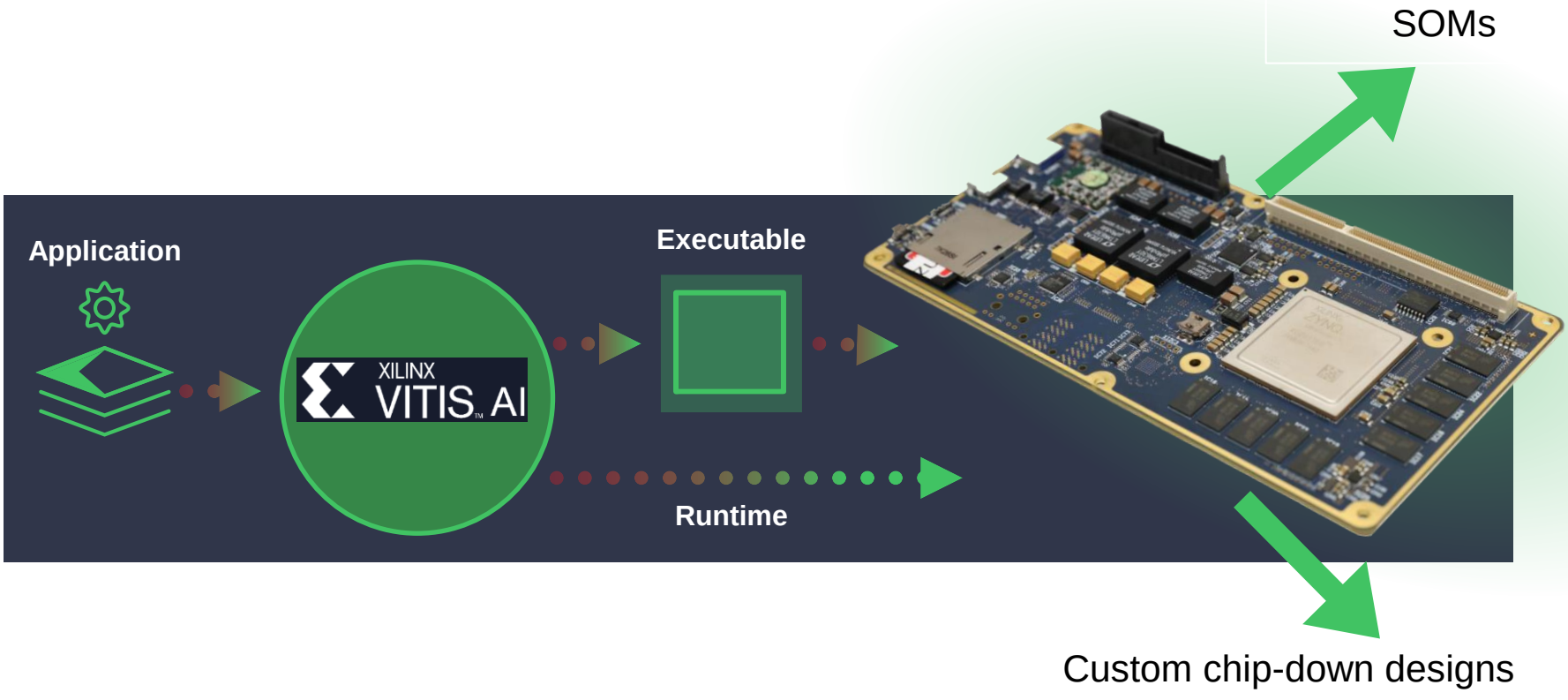
Alveo

Data Center Rack

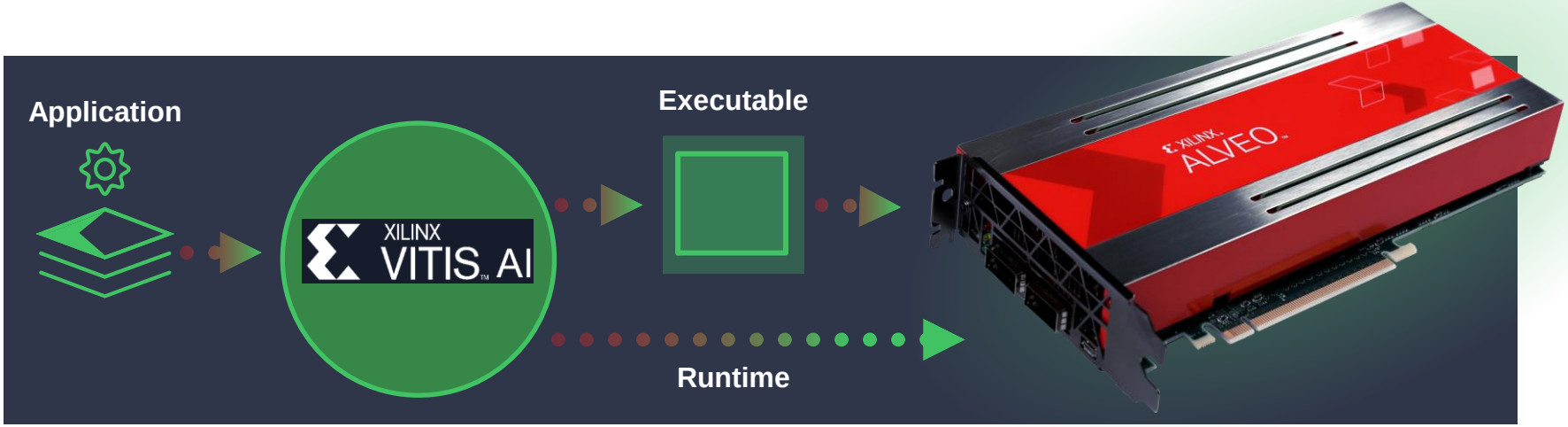
Development Flow for AI/ML



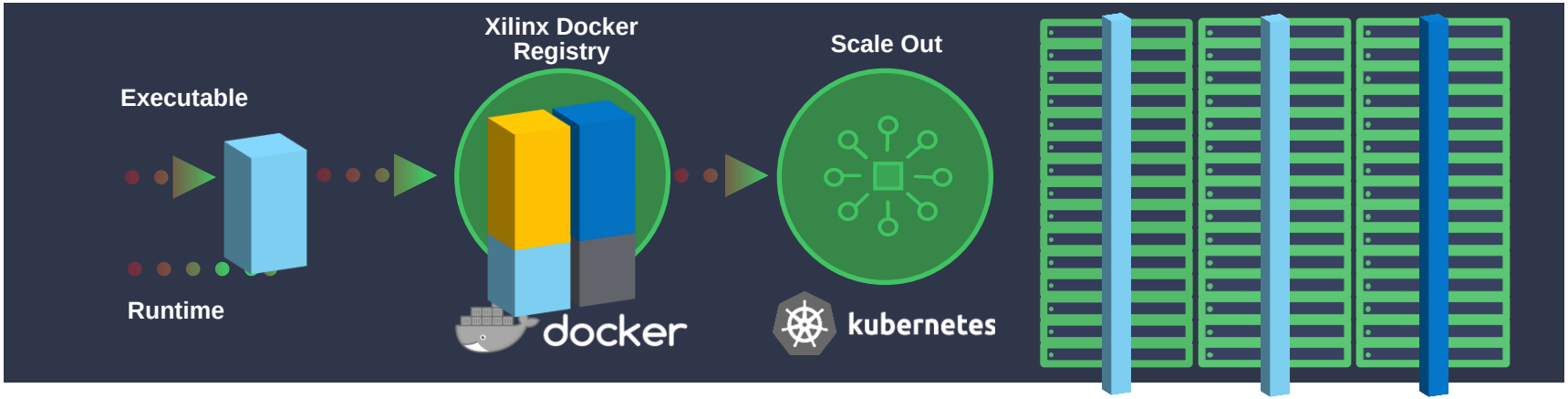
Deploy: Embedded Deployment



Deploy: Single Server Deployment



Deploy: Scale Out Deployment



From Model to Implementation in Minutes

DNN Processing Unit (DPU)

Direct Model Compilation

Minutes of Compile Times

MIN HRS

AI Model

PyTorch

TensorFlow

Vitis AI

Adaptive
Vitis Platform

Adaptable Platforms to Your AI Models

High

Throughput

DPUCVDX8H, Throughput optimized
- VCK5000, 384 AIE cores, max 8PE
- Popular CNN and NLP models
- Higher throughput with low latency to compete against GPUs

DPUCADF8H, Throughput optimized
- Alveo™ U200 / U250
- Optimized for 100% INT8 models and ResNet/Inception/MobileNet Series
- High Throughput at medium latency

DPUCVDX8G, Latency optimized
- VCK190, 96 AIE cores, C32/C64 batch, 1-5 batch
- Popular CNN models
- Scalable with power and cost

DPUCADX8H, Throughput optimized (HBM)
- Alveo U280 / U50 / U50LV / U55C
- Optimized for CNN models and HBM devices
- Low-cost and high-power efficiency solution

DPUCZDX8G, Scalability optimized
- Zynq®-7000, Zynq UltraScale+™ MPSoC, Kria™ SoM
- Popular CNN models
- Scalable with power and cost

Latency

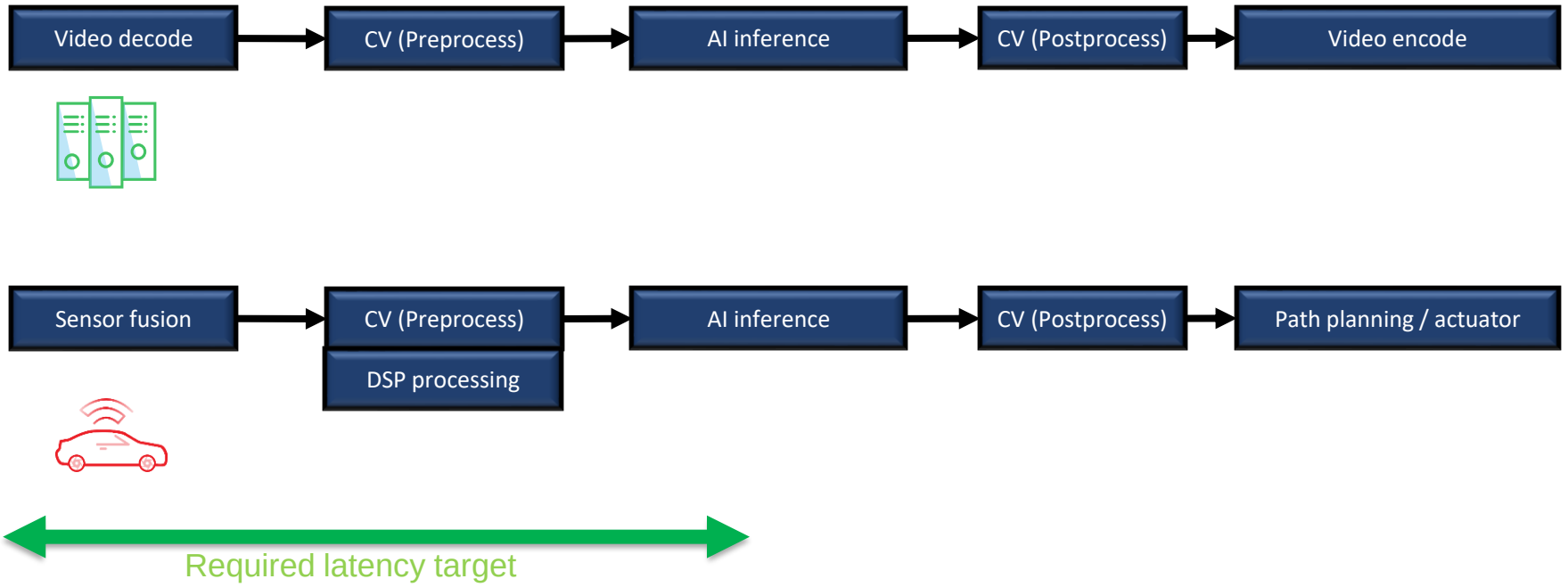
Low

High

7nm

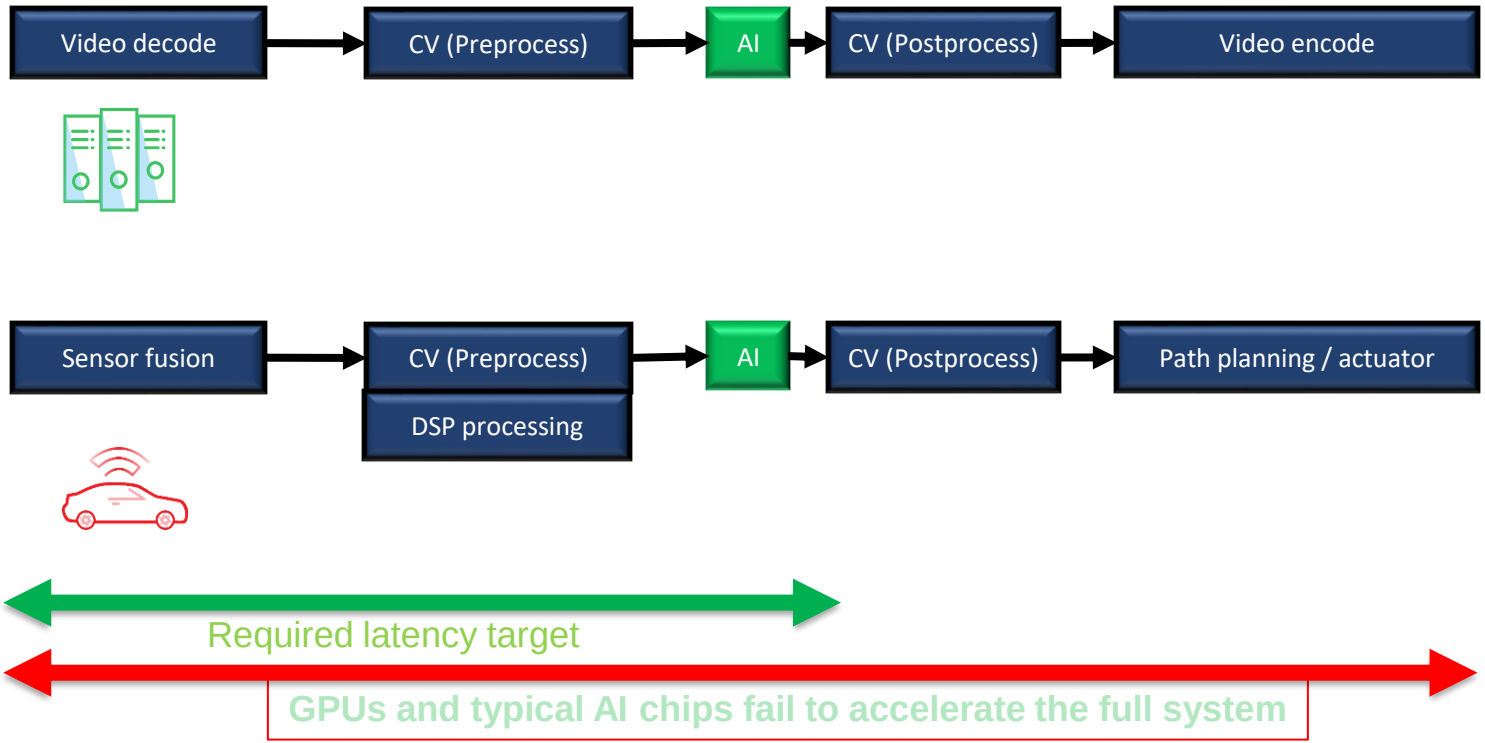
16/28nm

AI Application is Much More than AI

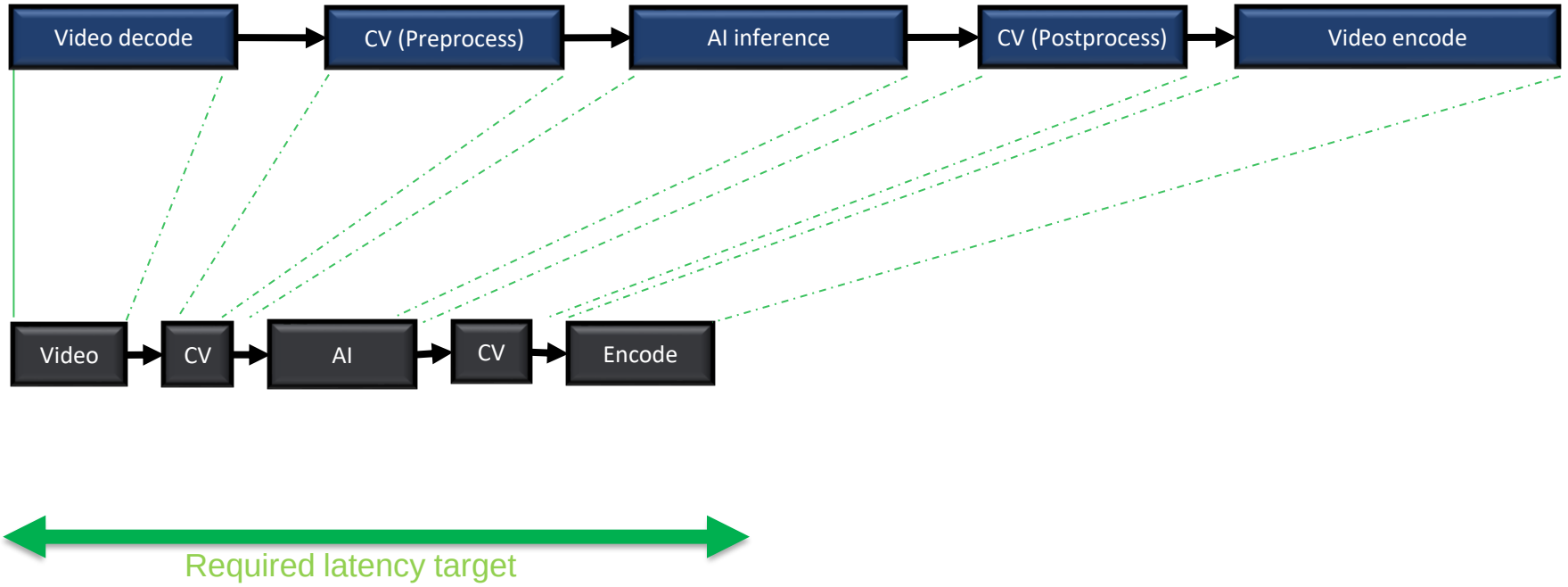


AVNET SILICA

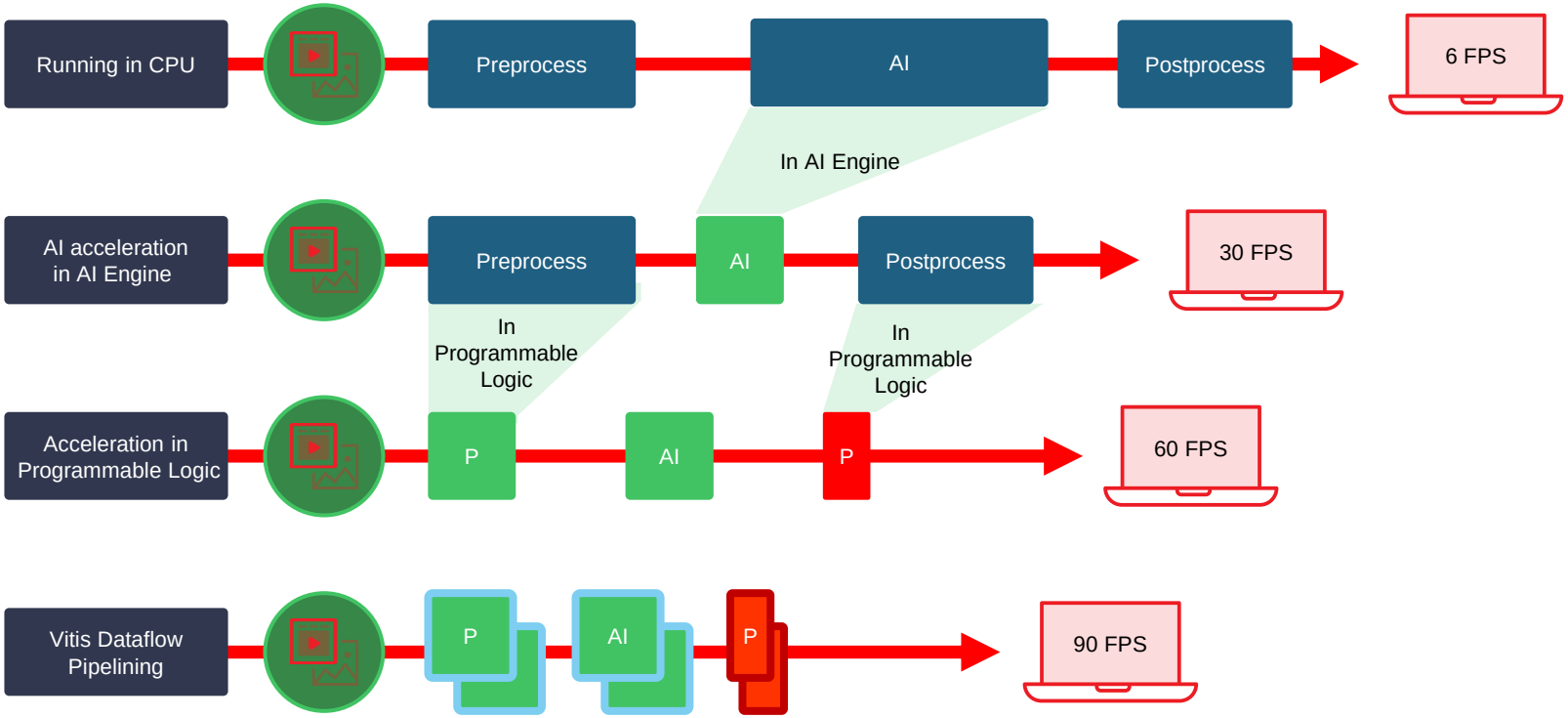
Accelerating AI only is NOT enough



DSA Needed to Accelerate AI and Non-AI

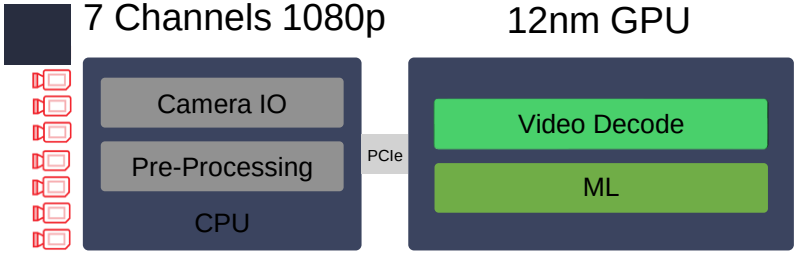


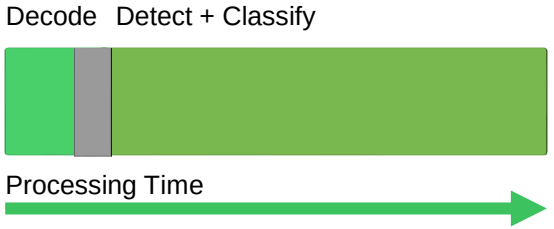
Adaptive Architecture for Smart City Application



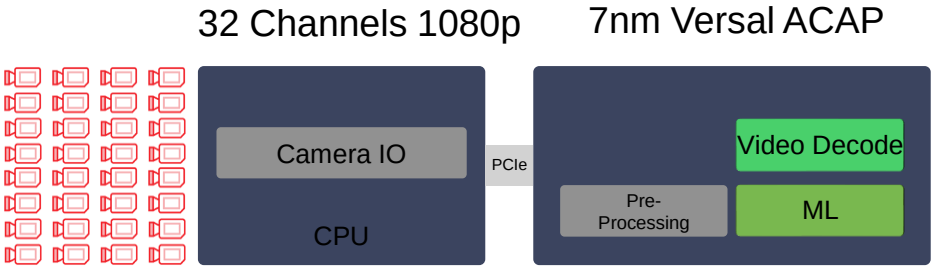
Impact of Whole Application Acceleration

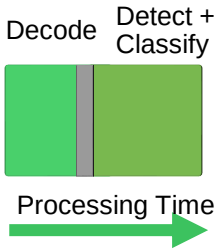
GPU





ACAP

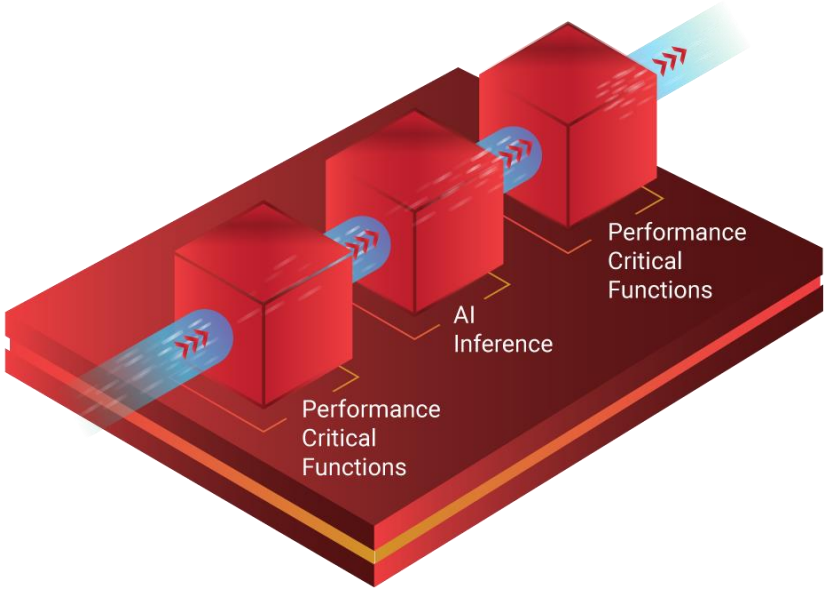




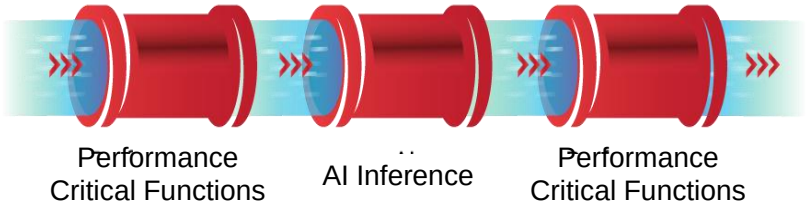
4.5X throughput

1/6 latency

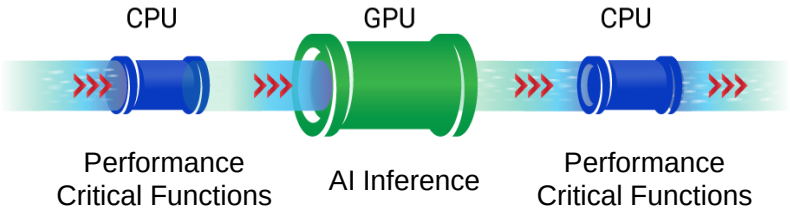
Throughput



AMD FPGAs – Matched Throughput

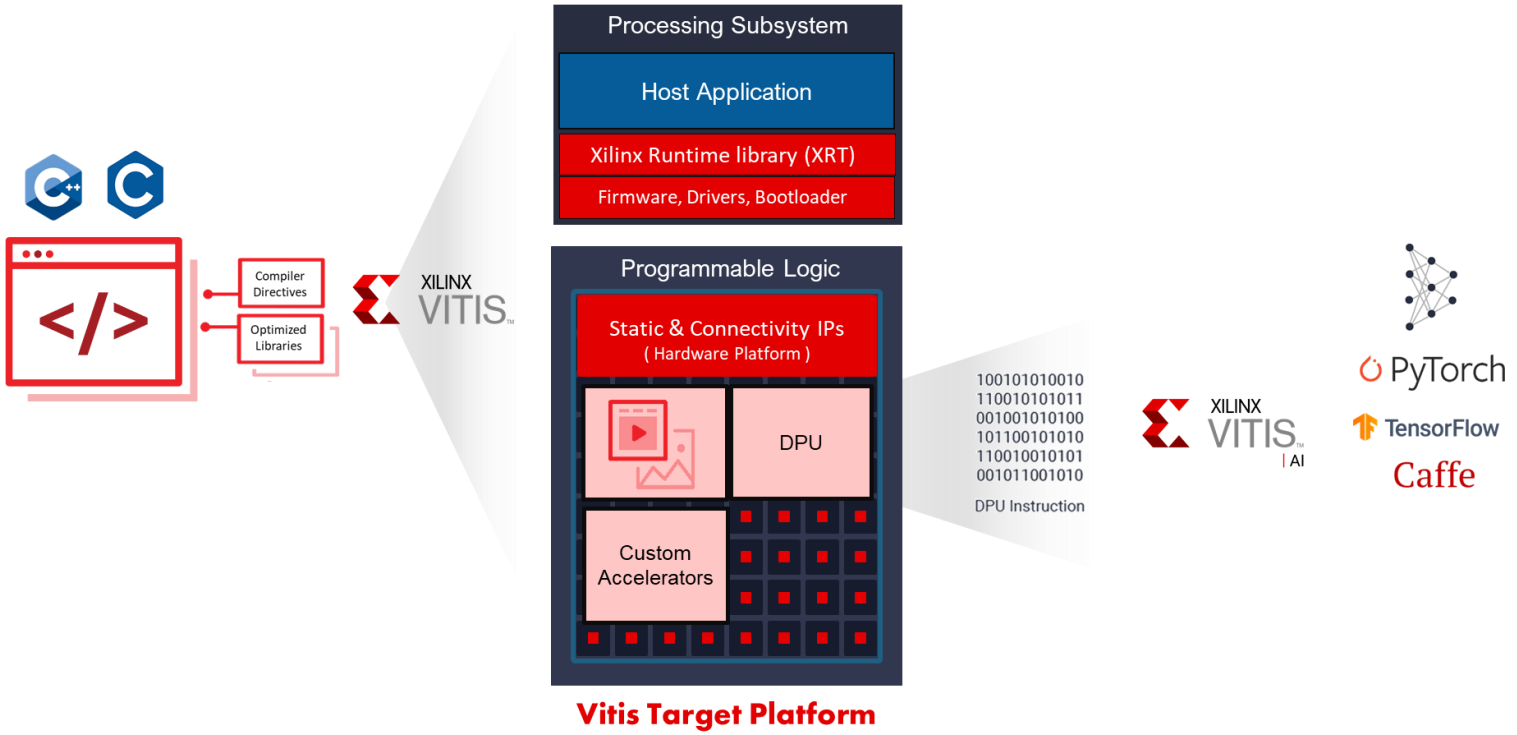


GPU & CPU – Mismatched Throughput

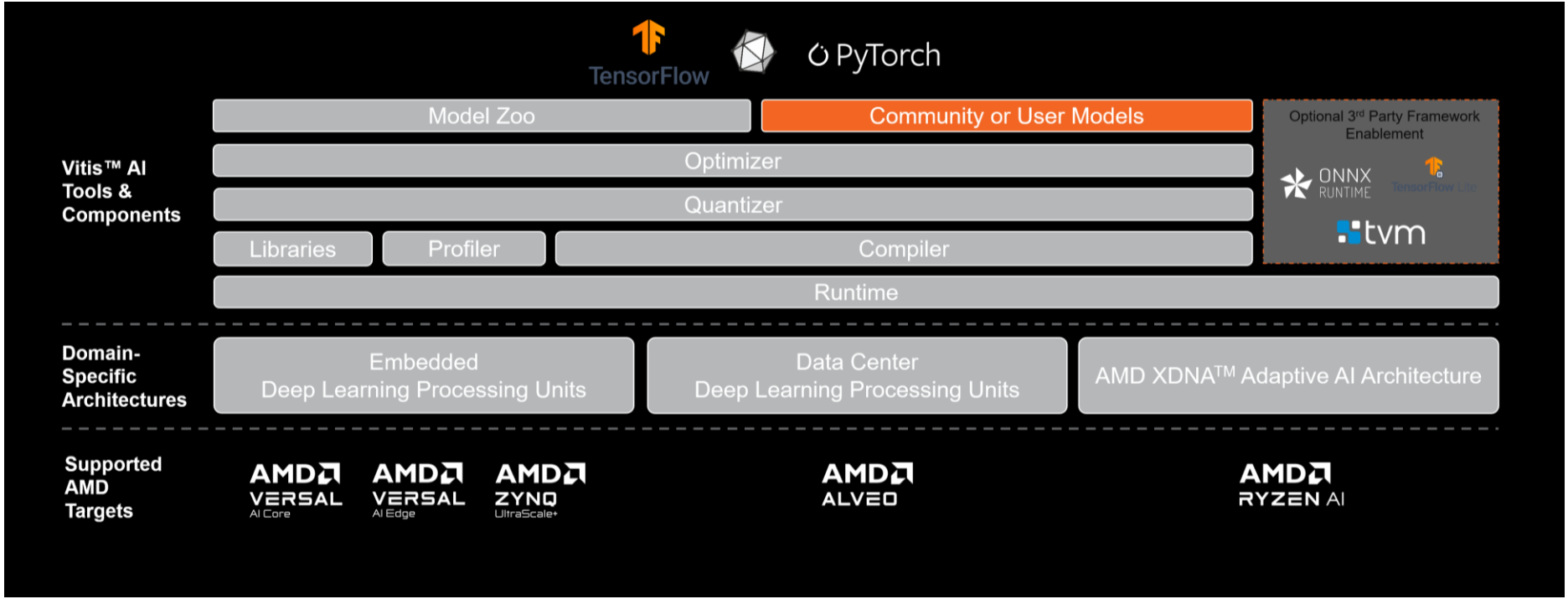


Deeper Dive into Vitis-AI

AMD Vitis AI Or Vitis… *What's the Relation ?*



AMD Vitis™ AI Integrated Development Environment



Who is it for?

XILINX VITIS AI

ML Engineers,
Data Scientists

PyTorch Caffe
TensorFlow
python

XILINX VITIS

Application &
Algorithm
Engineers

C++ C python
MATLAB

OPTIONAL

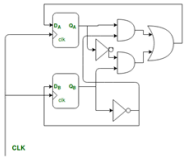
XILINX VITIS
GCC open source

Embedded
Software
Developers

C++ C
yocto PROJECT
FreeRTOS
Green Hills SOFTWARE INC.

VIVADO

Hardware/FPGA
Developer



VHDL Verilog
TCL

Vitis AI Key Components

DPUs
Configurable computation engines optimized for convolution neural networks. Efficient and scalable IP cores that can be customized to meet the needs of many different applications and devices.

Model Zoo
A comprehensive set of pre-trained and pre-optimized models that are ready to deploy on Xilinx devices.

Model Inspector
A tool and methodology through which developers can verify model architecture support.

Optimizer
An optional, commercially licensed tool that enables users to prune a model by up to 90%.

Quantizer
A powerful quantizer that supports model quantization, calibration, and fine tuning.

Compiler
Compiles the quantized model for execution on the target DPU accelerator.

Runtime (VART)
An inference runtime for Embedded applications.

Profiler
Performs an in-depth analysis of the efficiency and utilization of AI inference implementations on the DPU.

Library
Offers high-level C++ APIs for AI applications for embedded and data center use-cases

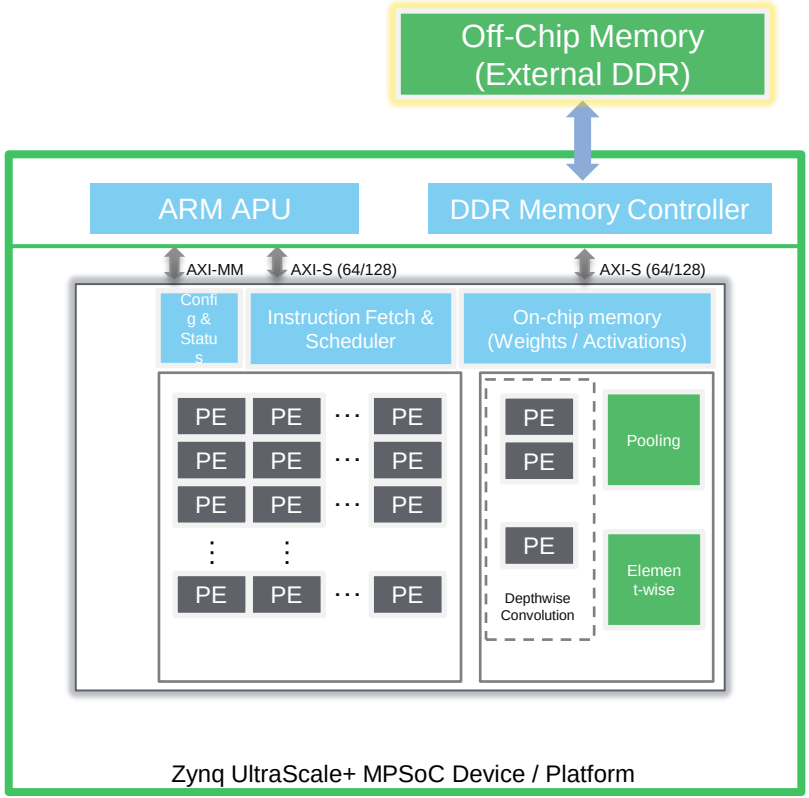
DPU Variants

DPUCZDX8G	• *Zynq-7000 SoC & Zynq US+ MPSOC (PG338)*
DPUCAHX8H	• Alveo U280/U55C and U50/U50LV (PG367)
DPUCVDX8G	• Versal AI Core Series (PG389)
DPUCVDX8H	• Alveo VCK5000 (PG403)
DPUCV2DX8G	• Versal AI Core, Versal AI Edge Series and Alveo V70 (PG425)

DPUCZDX8G for the Zynq™ UltraScale+™ MPSoC

- **Configurable DPU for Zynq SoC/ZU+MPSoC**
  - Supports multiple CUs in Vitis flow & 4 separate DPU cores in Vivado flow
  - Available DPU cores: B512,B800,B1152,B1600,B2048,B3136,B4096
  - B128/B144/B256 configurations supported upon request
- **Key operators supported:**
  - ALU engine
    - Support large-kernel-size MaxPool, AveragePool, rectangle-kernel-size AveragePool, 16bit const weights
    - support HardSigmoid and HardSwish
    - support DepthwiseConv + LeakyReLU
    - support the parallelism configuration (1~PP; B4096: 1/2/4/8)
- **New features available in AMD Vitis™ AI 3.0**
  - 1D and 2D Correlation, Cost-Volume, Argmax and Max

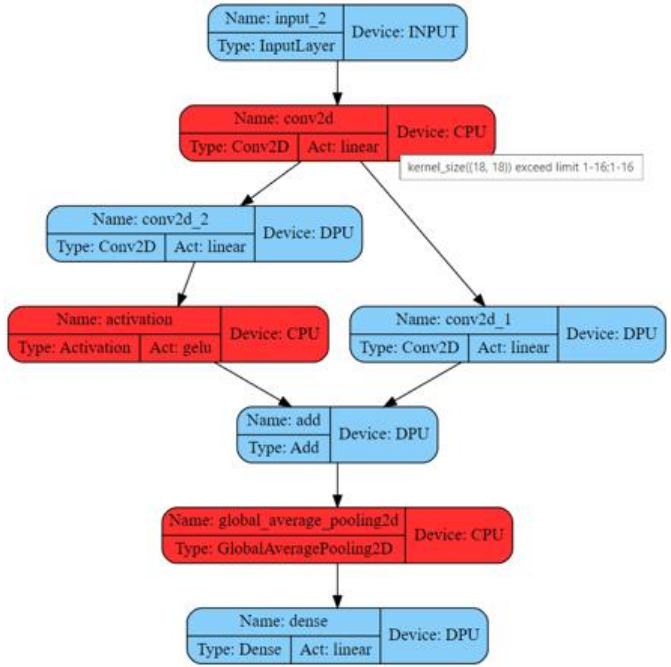
See UG1414 for DPU supported operators and limitations
See PG338 for feature configurations and integration



Vitis AI Model Inspector

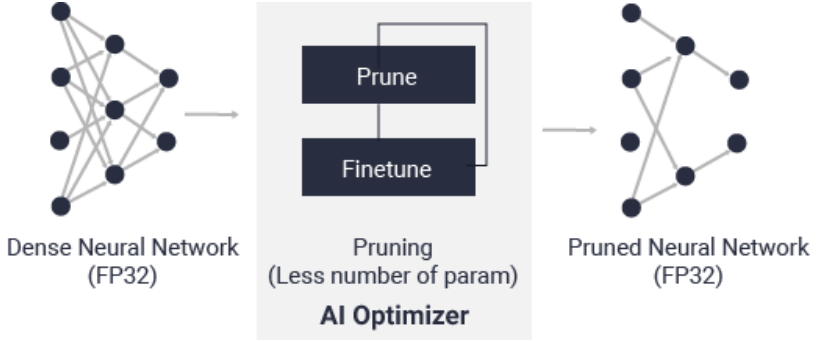
Output Example in TensorFlow 2

```
[VAI INFO] Inspect Results:
[MODEL INFO]:

Model Name: mnist_model

ID          Name                        Type                        Device      Notes
==========  ==========================  ==========================  ==========  ========================================================================
0/7         input_2                     InputLayer                  INPUT
----------  --------------------------  --------------------------  ----------  ------------------------------------------------------------------------
1/7         conv2d                      Conv2D<linear>              CPU         kernel_size((18, 18)) exceed limit 1-16;1-16
----------  --------------------------  --------------------------  ----------  ------------------------------------------------------------------------
2/7         conv2d_2                    Conv2D<linear>              DPU
----------  --------------------------  --------------------------  ----------  ------------------------------------------------------------------------
3/7         conv2d_1                    Conv2D<linear>              DPU
----------  --------------------------  --------------------------  ----------  ------------------------------------------------------------------------
4/7         activation                  Activation<gelu>            CPU         Conv2D<gelu> not supported, supported act types
                                                                                are ['relu', 'relu_six']; Standalone activation
                                                                                `gelu` is not supported
----------  --------------------------  --------------------------  ----------  ------------------------------------------------------------------------
5/7         add                         Add                         DPU
----------  --------------------------  --------------------------  ----------  ------------------------------------------------------------------------
6/7         global_average_pooling2dGlobalAveragePooling2D          CPU         Converted to VitisGlobalAveragePooling2D;
                                                                                kernel_size (28, 28) exceed limit [2, 3, 5, 7,
                                                                                8];[2, 3, 5, 7, 8]
----------  --------------------------  --------------------------  ----------  ------------------------------------------------------------------------
7/7         dense                       Dense<linear>               DPU
----------  --------------------------  --------------------------  ----------  ------------------------------------------------------------------------
================================================================================================================================================
[SUMMARY INFO]:
- [Target Name]: DPUCAHX8L_ISA0
- [Target Type]: DPUCAHX8L
- [Total Layers]: 8
- [Layer Types]: InputLayer(1) Conv2D<linear>(3) Activation<gelu>(1) Add(1) GlobalAveragePooling2D(1) Dense<linear>(1)
- [Partition Results]: INPUT(1) CPU(3) DPU(4)
================================================================================================================================================
[NOTES INFO]:
- [1/7] Layer conv2d (Type:Conv2D<linear>, Device:CPU):
    * kernel_size((18, 18)) exceed limit 1-16;1-16
- [4/7] Layer activation (Type:Activation<gelu>, Device:CPU):
    * Conv2D<gelu> not supported, supported act types are ['relu', 'relu_six']
    * Standalone activation `gelu` is not supported
- [6/7] Layer global_average_pooling2d (Type:GlobalAveragePooling2D, Device:CPU):
    * Converted to VitisGlobalAveragePooling2D
    * kernel_size (28, 28) exceed limit [2, 3, 5, 7, 8];[2, 3, 5, 7, 8]
================================================================================================================================================
[VAI INFO] Start plotting model to model2.svg
[VAI INFO] Inspected model has been plotted in: model2.svg.
```



Vitis AI Optimizer

- **World's leading model compression technology**
  - Iterative, coarse-grained pruning
  - Reduce model size 5 – 30x
  - Increase performance 2 – 10x
  - Minimal accuracy loss, <1%
- **Support mainstream frameworks**
  - TensorFlow 1.15
  - TensorFlow 2.4 - 2.12
  - PyTorch 1.4 - 1.13, 2.0
- **Support mainstream frameworks**
  - As of the Vitis AI 3.5 release, the Vitis AI Optimizer is now free-of-charge and is provided in the release repository.



Check more optimized models on Github AI Model Zoo

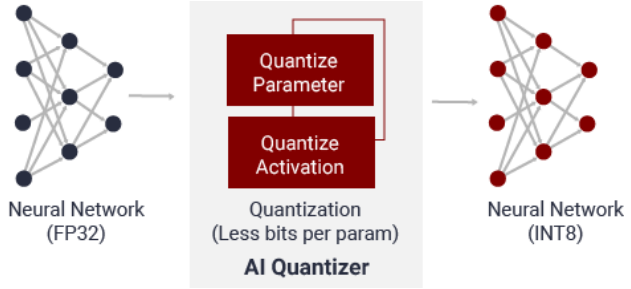
Model	Pruning Ratio	Operation (GOP)	Latency (ms)*	Throughput (FPS)**
RefineDet	-	123	115	18
	80%	25	31	76
	92%	10	16	154
	96%	5	12	228

* Latency is measured with a single thread
** Throughput is measured on ZCU104, with dual B4096 cores integrated

Vitis AI Quantizer

Powerful quantization tool

- 32-bit FP weights & activations → fixed INT8
- Reduce computing complexity without losing prediction accuracy
- Less memory bandwidth, faster speed and higher power efficiency
- PTQ and QAT flows supported



Support mainstream frameworks

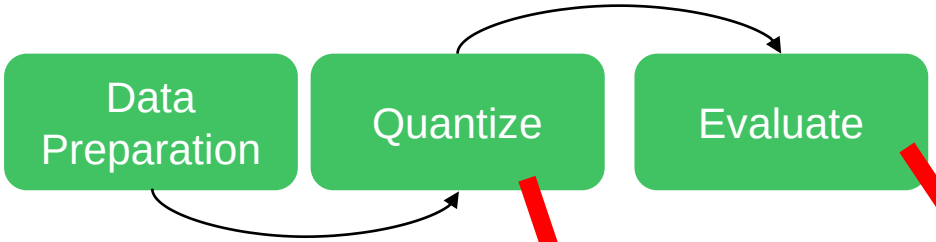
- TensorFlow 1.15
- TensorFlow 2.4 - 2.12
- PyTorch 1.4 - 1.13, 2.0

Vitis AI Quantizer - PTQ

Post-training quantization (PTQ) of the float model, includes model optimization, weights quantization and activation quantize calibration.

Requirements:

- (FP32) Trained Model
- Calibration Dataset (100-1000 samples sufficient)




```
# run quantization
quantizer = vitis_quantize.VitisQuantizer(orig_model)
ptq_model = quantizer.quantize_model(calib_dataset=x_train[0:CALIB_SIZE])
```

```
[VAI INFO] Update custom_layer_type: []
[VAI INFO] Start CrossLayerEqualization...
10/10 [==============================] - 0s 40ms/step
[VAI INFO] CrossLayerEqualization Done.
[VAI INFO] Start Quantize Calibration...
32/32 [==============================] - 1s 10ms/step
[VAI INFO] Quantize Calibration Done.
[VAI INFO] Start Post-Quantize Adjustment...
[VAI INFO] Post-Quantize Adjustment Done.
[VAI INFO] Quantization Finished.
```

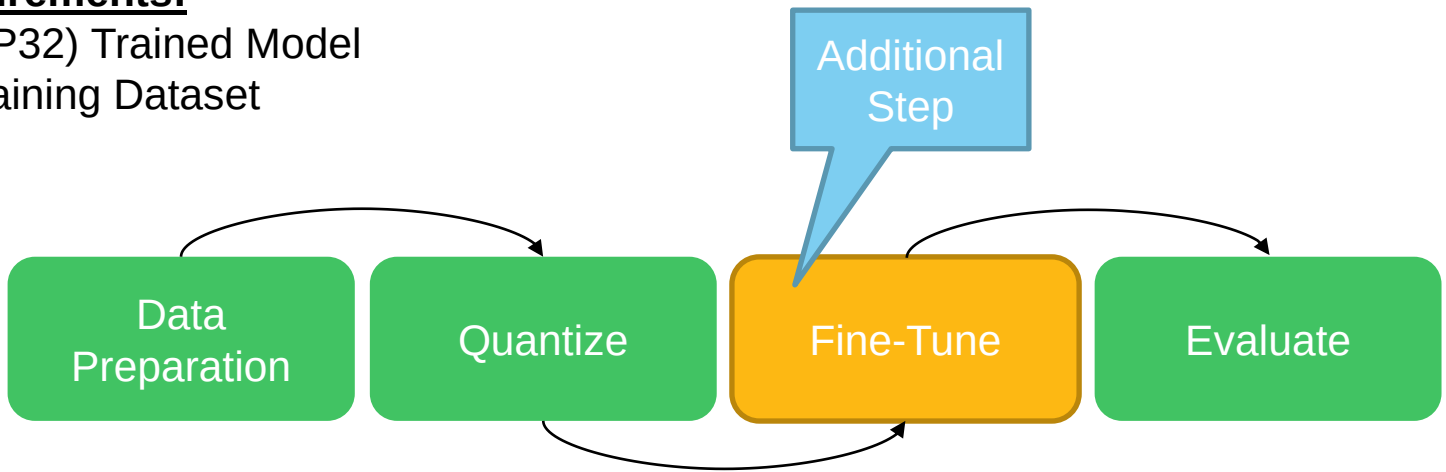
```
Original Accuracy: 98.99%
PQT Accuracy:      99.00% - Difference: -0.01
```

Vitis AI Quantizer - QAT

Quantization aware training (QAT) is similar to float model training/finetuning, but in QAT, the vai_q_tensorflow APIs are used to rewrite the float graph to convert it to a quantized graph before the training starts. Useful in cases where PTQ results in noticeable accuracy drop.

Requirements:

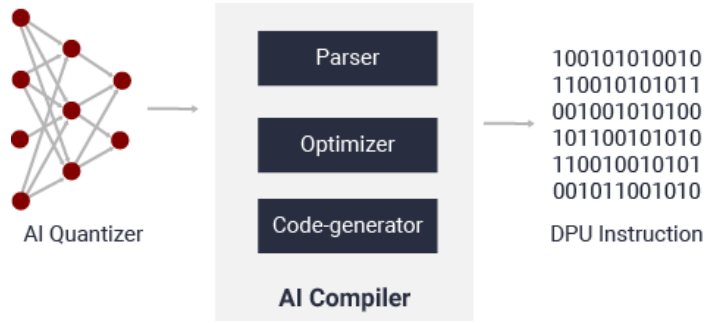
- (FP32) Trained Model
- Training Dataset



Vitis AI Compiler

A Unified compilation flow

- Maps AI model to highly-optimized DPU instructions
- IR-based for multiple DPUs & frameworks
- Auto-partitioning for custom layer plug-and-play




ZU+ MPSoC


Alveo


Versal ACAPs

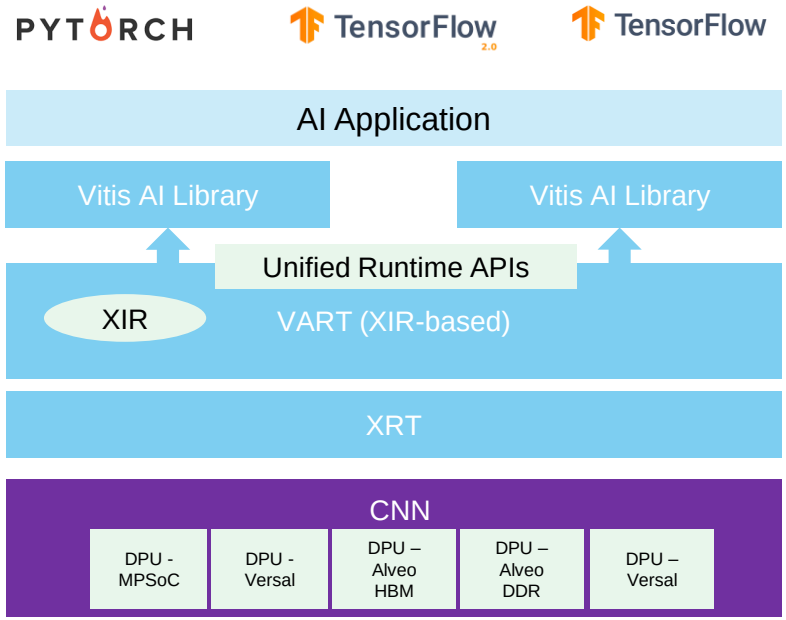

VCK5000

DPU Name	Hardware platform
DPUCZDX8G	Zynq® UltraScale+™ MPSoC
DPUCAHX8H	Alveo™ U50LV, U55C Data Center accelerator cards
DPUCADF8H	Alveo U200, U250 Data Center accelerator cards
DPUCVDX8G	Versal® ACAP VCK190 evaluation board, Versal AI Core Series
DPUCVDX8H	Versal ACAP VCK5000 evaluation kit

Vitis AI Compiler generates the compiled model based on the DPU microarchitecture.

Vitis AI RunTime (VART)

- **High-level API-based libraries and runtime for edge and cloud**
  - Asynchronous submission of jobs to the accelerator
  - Asynchronous collection of jobs from the accelerator
  - C++ and Python implementations
  - Support for multi-threading and multi-process execution
- **Programming with VART (UG1414)**
  - C++ examples
  - Python examples



Vitis AI Profiler

Vitis AI Profiler – Application-level Tool

Optimize the Whole Application

Some parts of an AI application run on the DPU and some on the CPU

Vitis AI profiler can help put the running status of all parts together

Profiles

- Pre-processing
- Post-processing
- DPU kernels

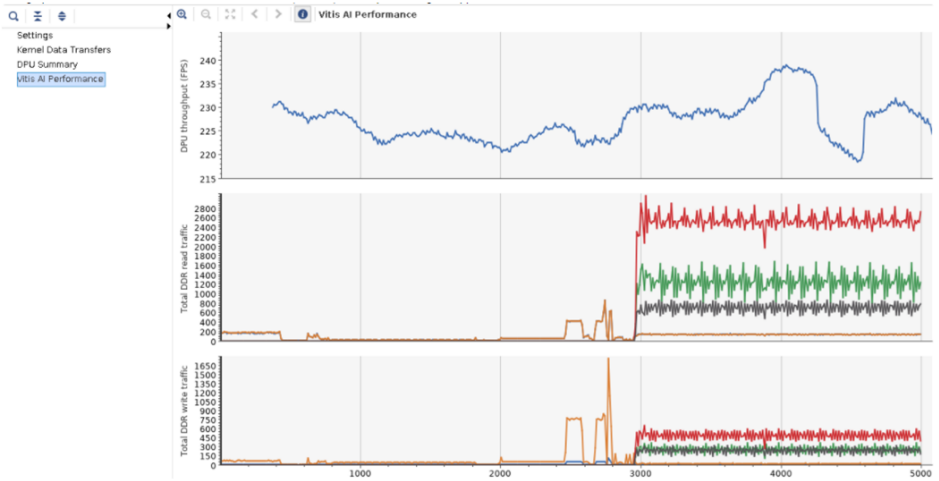
Pre-processing function takes long time

- High CPU utilization
- DPU cores wait

Bottlenecks: Pre-processing and CPU

Improve performance: Rewrite the preprocessing function(s) in HLS

Detect Bottlenecks



Q&A

Typical Questions

Are all the components of Vitis AI free?

Yes!

As of the 3.5 release all components are free!

For releases <3.5, the Vitis AI Optimizer does require a separate license which can be obtained free-of-charge upon request.

Is Vitis AI a separate download?

Yes! Users can get started by cloning the Vitis AI GitHub repository:

GitHub - Xilinx/Vitis-AI: Vitis AI is Xilinx's development stack for AI inference on Xilinx hardware platforms, including both edge devices and Alveo cards.

What Xilinx Target Device Families and Platforms does Vitis AI Support?

Vitis AI DPUs are available for:

- Zynq 7000
- Zynq Ultrascale+ MPSoC
- Versal AI Edge
- Versal AI Core

How does FPGA compare to CPU and GPU acceleration?

FPGA accelerated networks can run up to 90x faster as compared to CPU. FPGA accelerated networks are on par with GPU accelerated networks for throughput critical applications yet provide support for more custom applications.

FPGA accelerated networks are far superior to GPU accelerated networks for latency critical applications such as autonomous driving.

Lot of scope in reducing on a system level power / cost and creating scalable designs.

Is it possible to deploy the DPUCZ using Yocto flows, or even Ubuntu, rather than PetaLinux?

Yes!

What is important to consider is that each release of the Vitis AI tool and the DPUCZ IP is provided with drivers and a runtime that targets a specific Linux kernel release. Misalignment between the target kernel version can pose challenges and may require extensive code changes.

My DPU implementation does not meet my latency/throughput targets. Is there anything else I can do?

Yes!

Besides modifying architecture and/or taking advantage of pruning within Vitis AI, you can also explore FINN.

FINN implements each layer of a neural network separately and creates like this a custom very low latency and high throughput design in the PL.

Thank You