FlashTVM

Optimizing Deep Learning Computation on OpenCL-compatible Hardware Accelerators

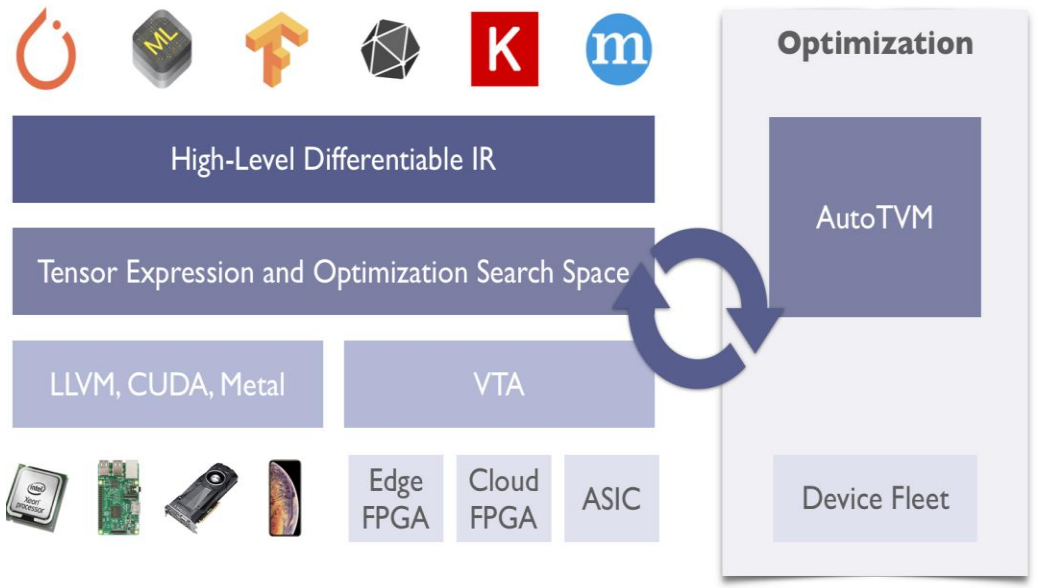
第四范式 Fourth Paradigm Southeast Asia PTE LTD

4Paradigm 第四范式

Background

TVM is an open deep learning compiler stack for CPUs, GPUs, and specialized accelerators.

- Compilation of deep learning models in Tensorflow, Keras, PyTorch, etc, into minimum deployable modules.
- Infrastructure to automatically generate and optimize tensor operators on diverse backends with optimized performance.
- VTA: Versatile Tensor Accelerator

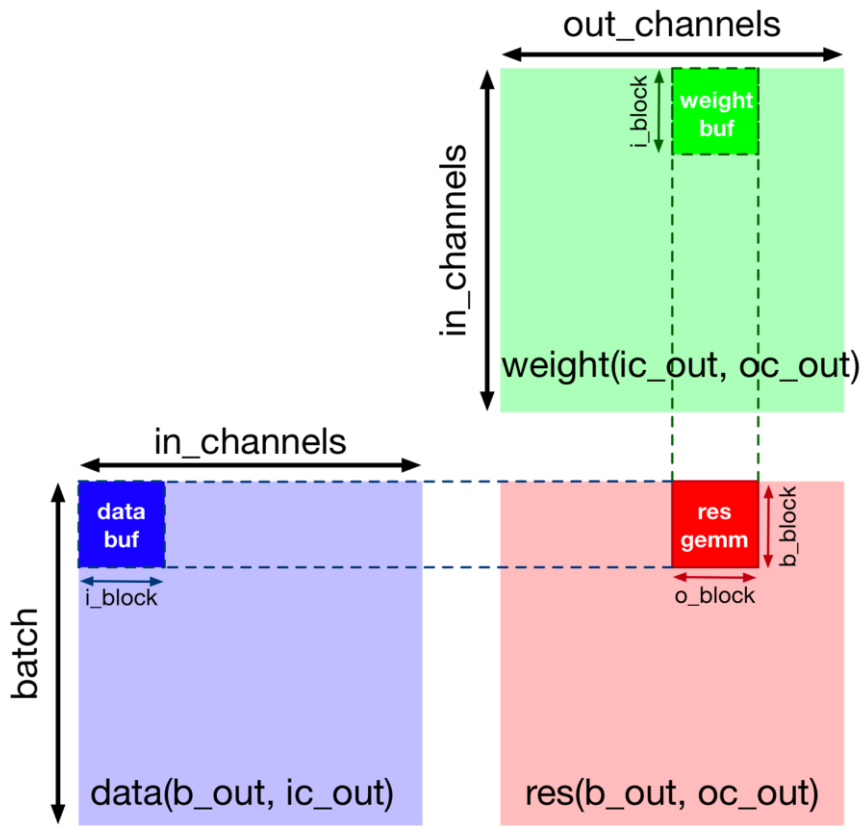


Source: https://tvm.apache.org/

Background

- Auto-tune
  - Determine schedules: Split, tile, fuse, re-order, etc.
  - Search the best schedules available for different OPs.
  - Search Algorithms: Random, grid search, XGBTuner.

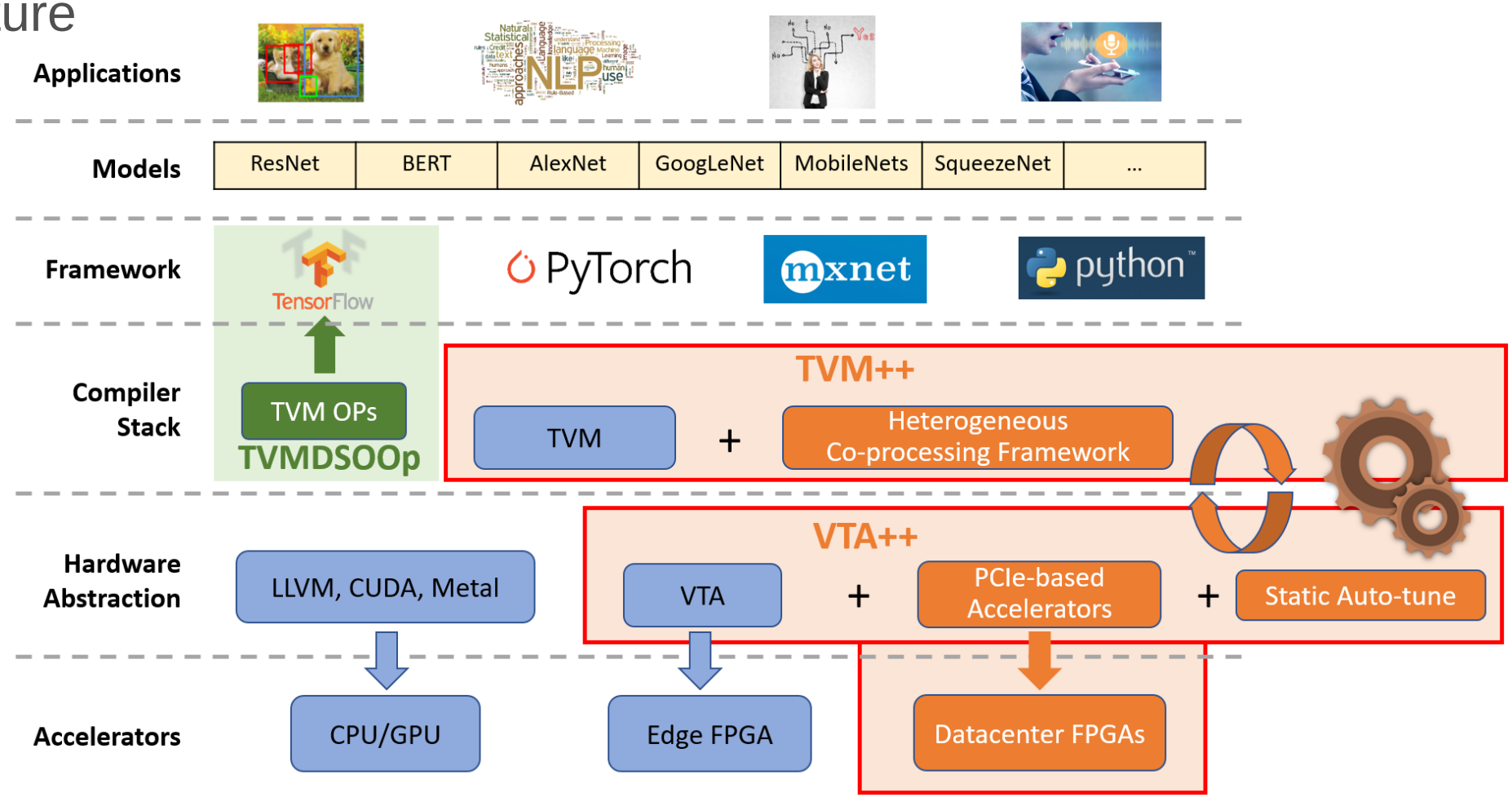
- Tuning Process
  - Retrieve specific workload according to network structures. (e.g. conv2d(14, 14, 32, 32))
  - Profiling trail-runs on actual hardware, determining the best schedule available



Source: https://tvm.apache.org/

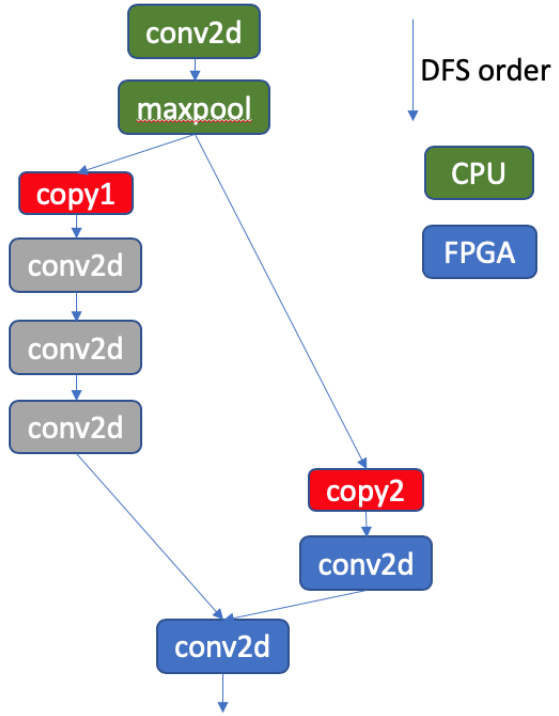
FlashTVM

- Architecture



Framework Enhancements

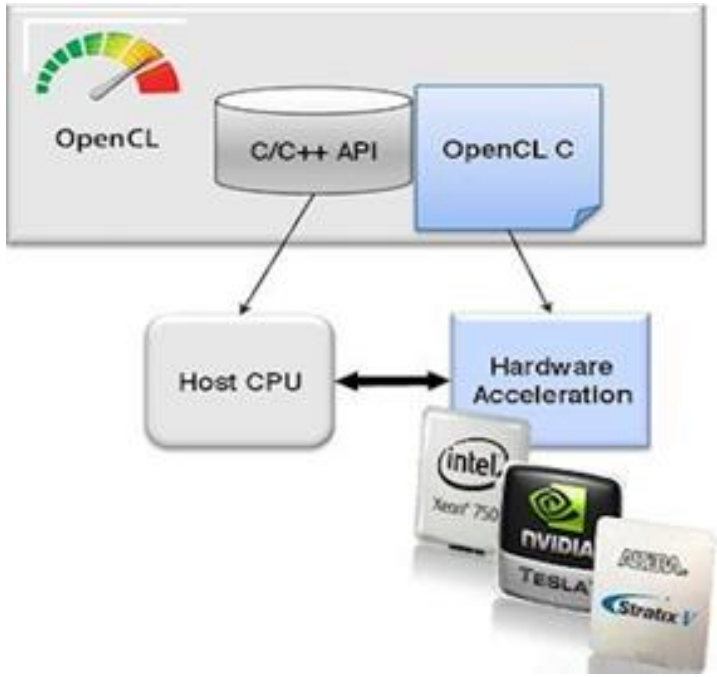
- The Current VTA only works on edge SoC FPGA Devices
  - Based on Shared-Memory Model
    - Shared DDR memory via internal bus
  - Most datacenter FPGA accelerations cards are PCIe cards
    - Memory access via PCIe Link
    - High communication cost
    - Need to minimize data exchange



- Heterogeneous Co-processing Framework
  - IR pass: annotate device types (*on_device*)
  - Enhanced IR pass: Tag and propagate device types
  - IR pass: Perform *device_copy* automatically
  - Pointer type support

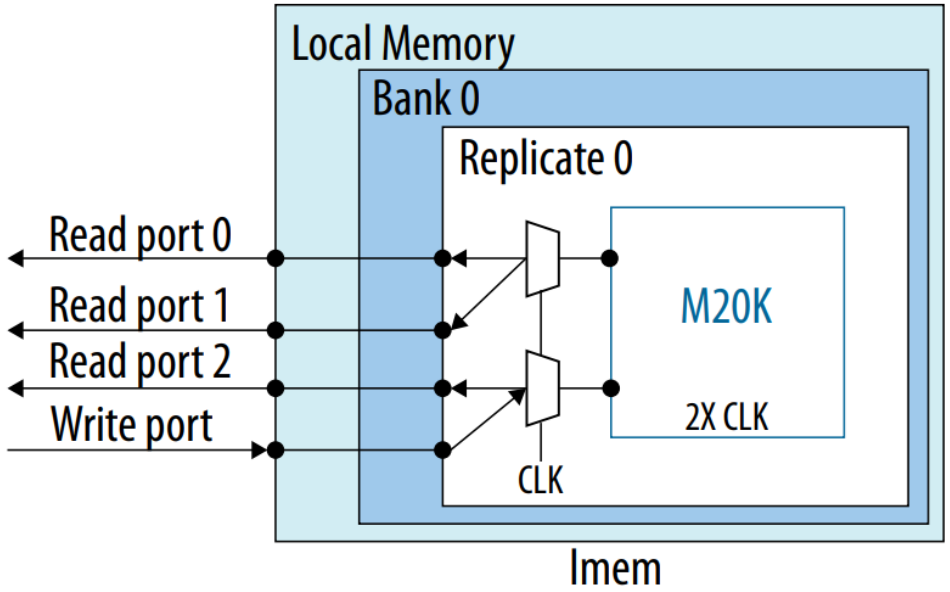
Versatile Tensor Accelerator

- We refactored the codes to conform to OpenCL standard
  - The original Xilinx HLS VTA core only works on Xilinx Edge FPGA devices
  - Why OpenCL?
    - Support from both Xilinx and Intel
    - Proven portability and scalability
      - For both official and custom boards
    - Vendor-specific optimizations are built-in within their respective official SDKs



HDL or HLS?

- Emphasize on official toolchain support?
  - FPGA Devices are not blank canvases
    - Pre-defined logic cells, memories, DSP slices, etc.
    - Different from vendor to vendor, from generation to generation, and even from product family to family.
  - To build FPGA applications of high performance
    - Design our circuit based on underlying elements
  - “Double Pumping” of BlockRAM
    - Number of available physical ports
    - Available on Arria 10 but not Stratix 10



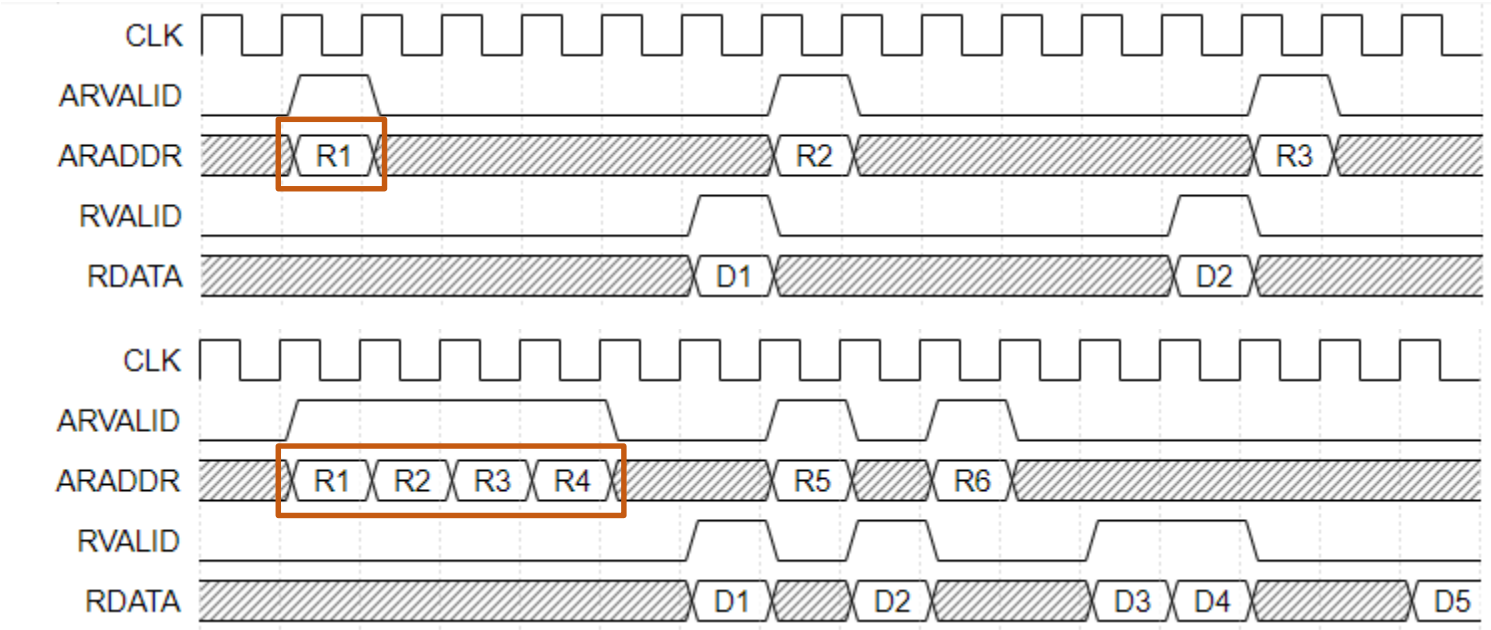
FlashTVM

- Some operations are not supported on existing VTA core
  - Get rid of CPU-assisted calculations
  - Additional instructions for continuous calculations on FPGA

- As we have more resources available on datacenter FPGAs
  - The original HLS VTA coding can scale up
  - Performance degradations observed!
  - Little improvement over the original design

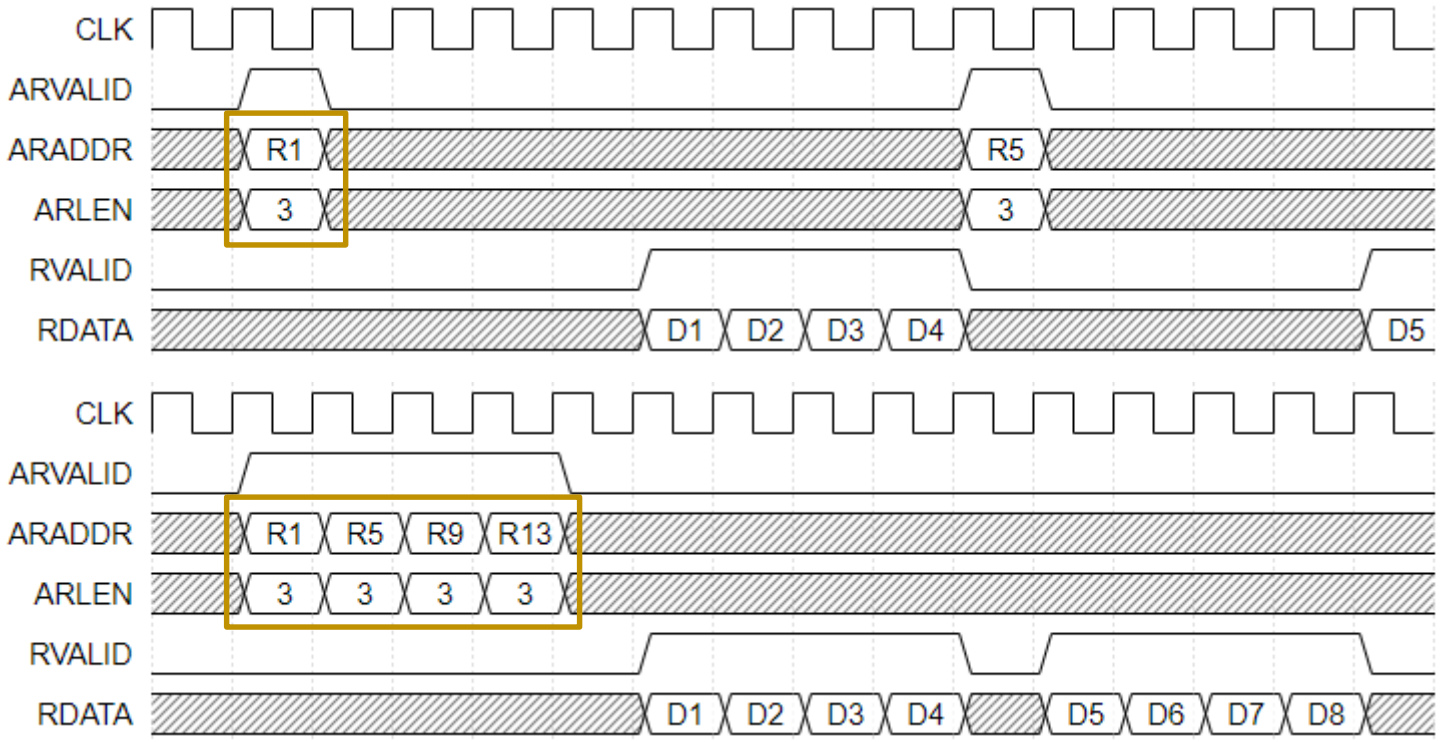
Performance Considerations

- External memory bandwidth
  - Hardware Capability (number of DDR channels, HBM, etc.)
  - Memory Access Efficiency



Performance Considerations

- Burst mode for sequential memory access

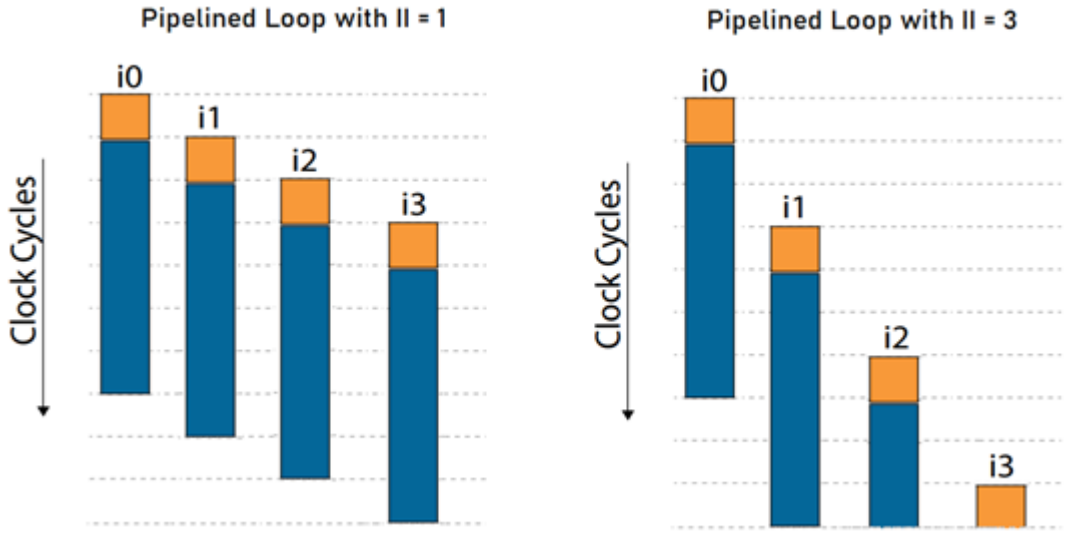


- Burst-coalesced LSUs

Performance Considerations

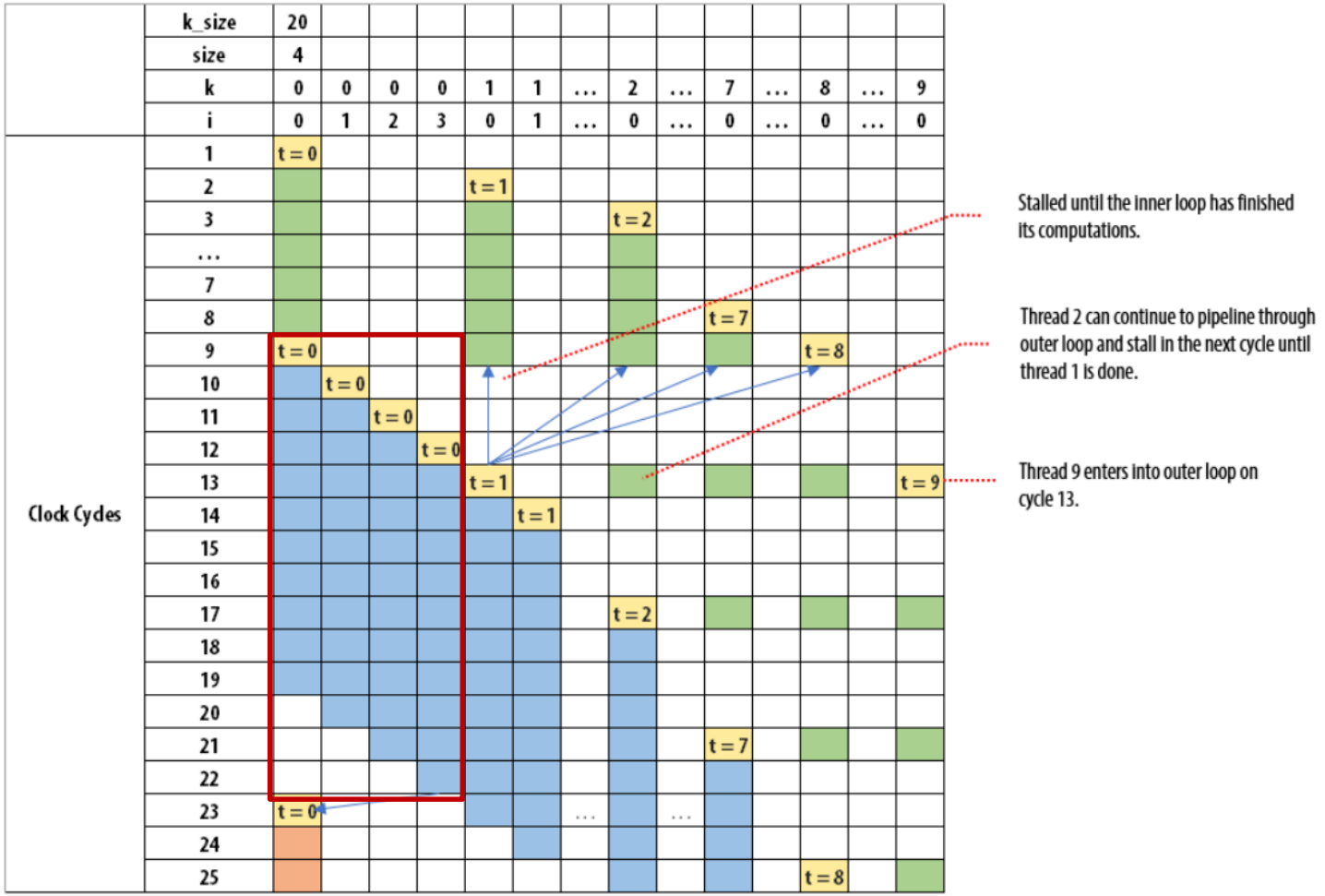
- Initiation Interval (II)
  - Number of clock cycles between the start times of consecutive loop iterations
  - Crucial for loop pipelining in terms of performance
  - II = 1 for optimal loop pipelining

- Undesired loop performance
  - Loop-carried Dependencies
  - Conflicted Resource Access
  - Pipeline Stall



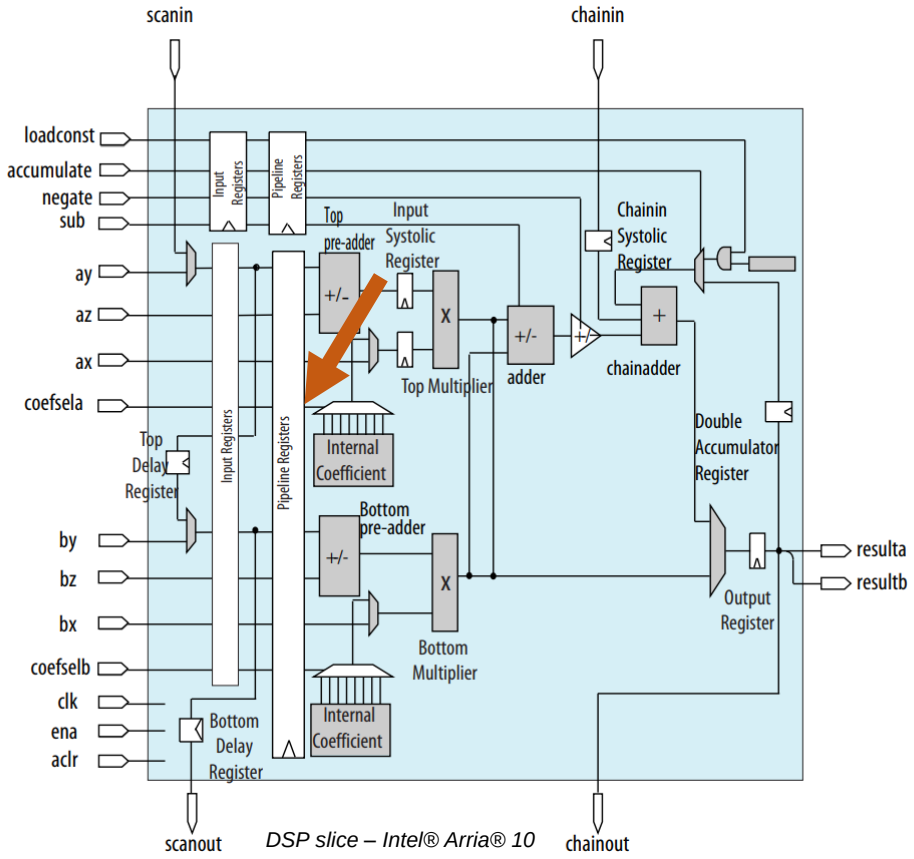
Performance Considerations

- Nested Loops
  - Unroll
  - Flatten
  - Loop Coalescing
  - Nested Pipelining
- Trade-off between resource usage and performance

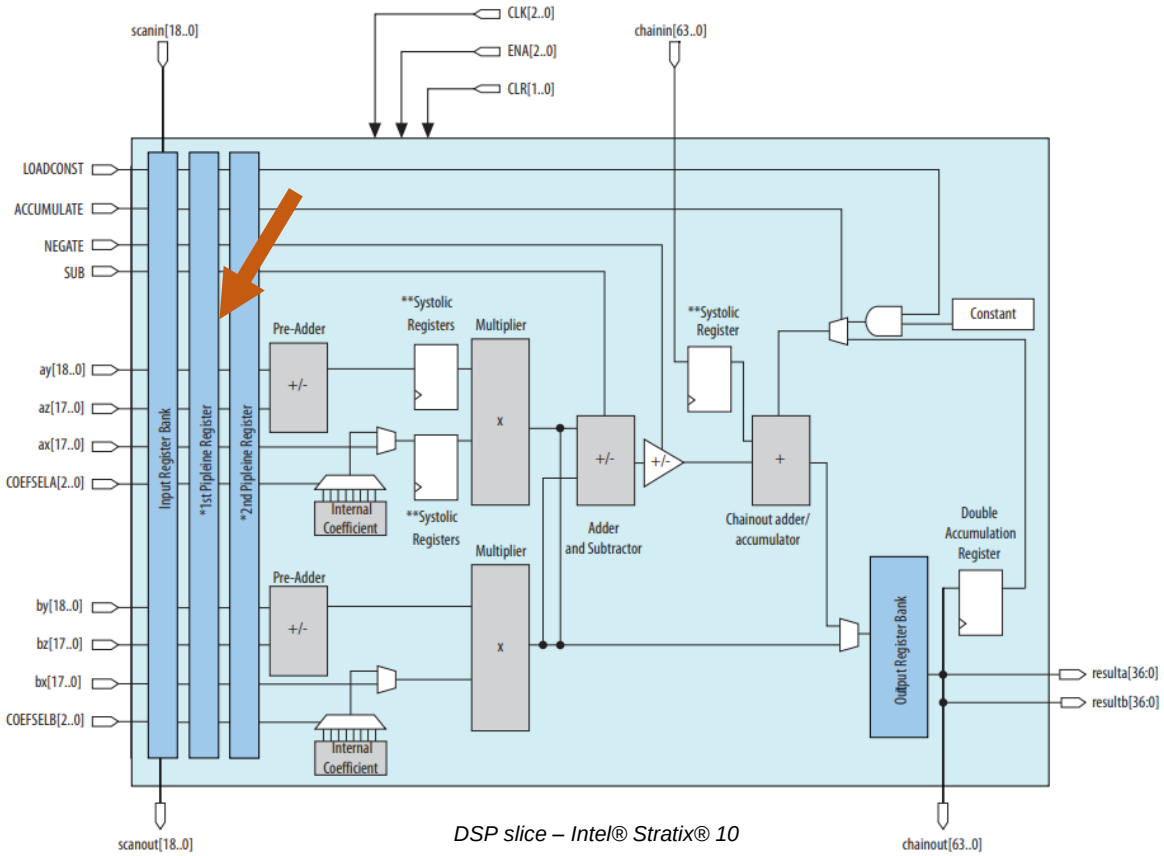


Performance Considerations

- Even more precise controls achievable by embedding RTL modules
  - Keep in mind of underlying structure when designing circuits



DSP slice – Intel® Arria® 10

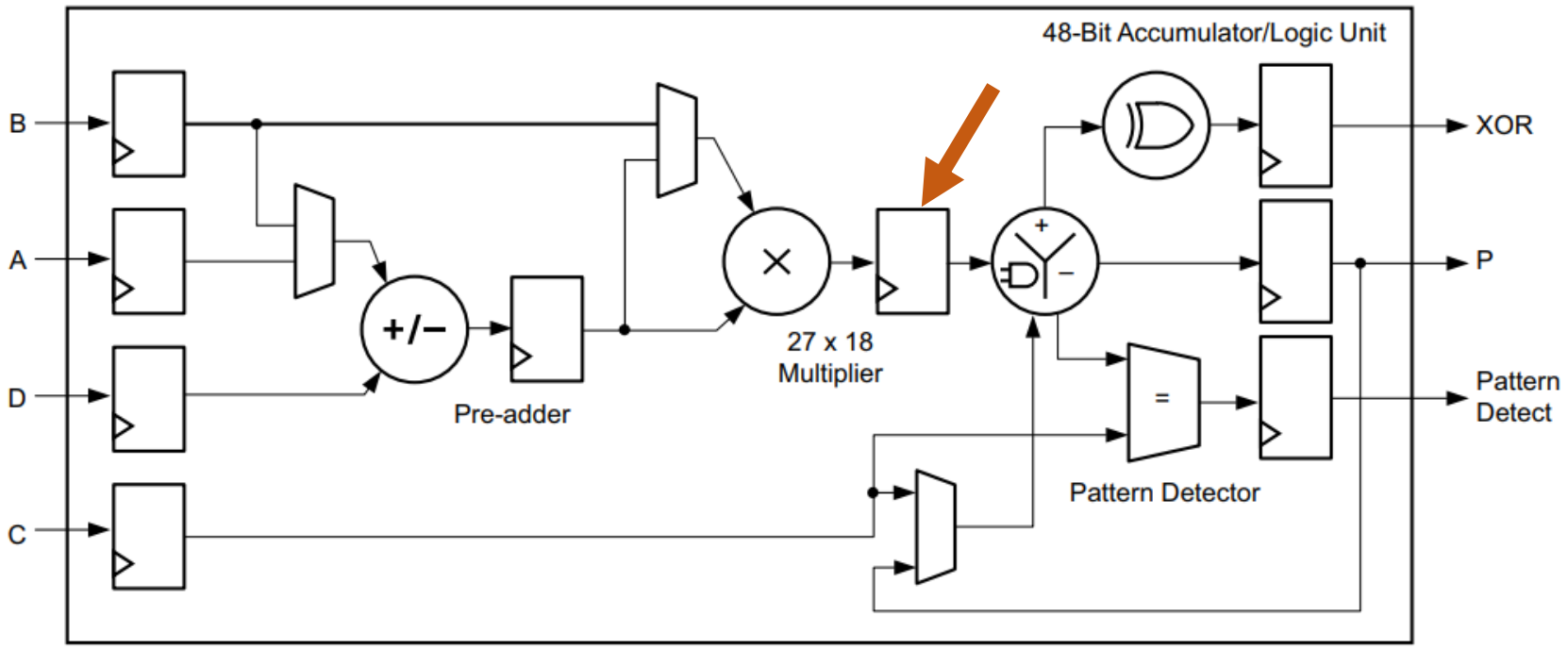


DSP slice – Intel® Stratix® 10

Performance Considerations

- Where to insert registers?

DSP slice – Xilinx® UltraScale™



- How to fully utilize the 27x18 multipliers available?

Future Work

- Enabling FlashTVM for more Deep Learning Networks
  - Instruction Set Extensions to support more ops in VTA
  - New framework features required by more complicated models (e.g. Dynamic Shape)

- Further Performance Enhancement
  - Schedule enhancement for graph computation
  - Multi-core parallelism for better resource utilization
  - Explore new hardware features available (e.g. HBM2, UPI, DCPMM, etc.)

References

- Apache (incubating) TVM. https://tvm.apache.org
- Intel FPGA SDK for OpenCL Pro Edition: Programming Guide
- Vivado Design Suite User Guide: High-Level Synthesis
- Xilinx SDAccel Programmers Guide

Q & A

- Any Questions?

Thank You!

AI for everyone.

For business enquiries
contact@4paradigm.com

TEL
010-8278-0800

For media enquiries
pr@4paradigm.com