AI/ML with Arm HPC

Centre for Development of Advanced Computing (C-DAC) / National Supercomputing Mission (NSM)

Arm in HPC Course

Phil Ridley

phil.ridley@arm.com

3rd March 2021

Agenda

- Containers
- ML and AI
  - Processor developments
  - Community support
  - Libraries and Applications
- ISA Developments

Containers

Arm enables containerization through standardization

Ensuring standard interfaces work on Arm enables multiple technologies

Approach

Standards bodies define the interfaces between layers

Example: Open Containers Initiative (OCI)

Arm works with the standards bodies to see that Arm is well supported

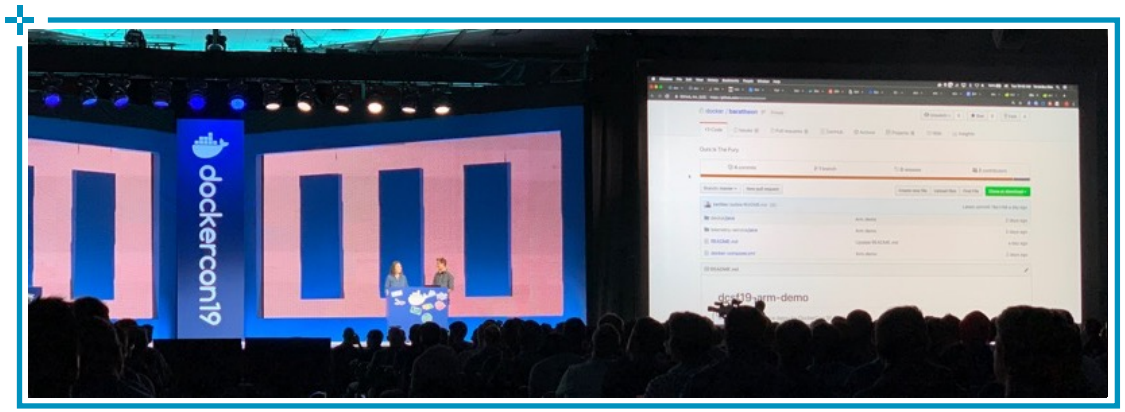
Example: Arm supporting containers drives Aarch64 support in Docker

All the "level-set" work is done.

Arm is now focusing on enabling leading technologies

arm

Arm & Docker Partner to Deliver Frictionless Cloud-Native Software Development



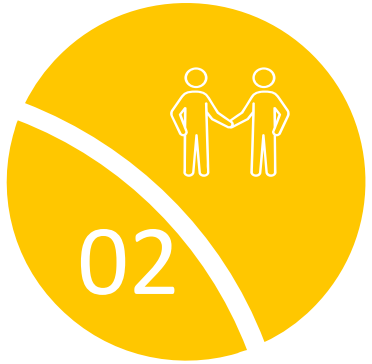
01 Initial phase is focused on Integration of Arm capabilities into Docker Desktop Community to enable a seamless developer environment

02 Docker Enterprise Engine for Amazon EC2 A1 instances

03 Additional work will address end-to-end management of full product life cycle; unified development environments for heterogeneous compute and scaling cloud-native benefits to consolidate edge workloads

Docker on Arm

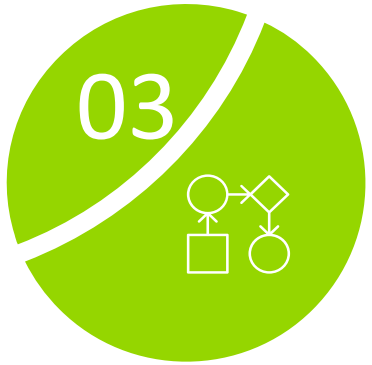
Docker Desktop is the de facto standard Cloud Native development platform for containerized applications



This partnership makes it easier for millions of developers already using Docker to develop containers on Arm

arm docker

5,386,145 base images on Docker Hub
51,460 Arm images
46,167 Arm64 images
Official **166** Docker images
Arm support **118** out of **166** official images



No changes needed to Docker tooling & processes in order to start building for Arm

docker × arm

Machine Learning

Machine Learning

TensorFlow

- Arm actively involved

Deepbench

- Arm actively involved

Torch

- Community maintained

Mahout

- Available via Apache Bigtop

Weka

- Community maintained

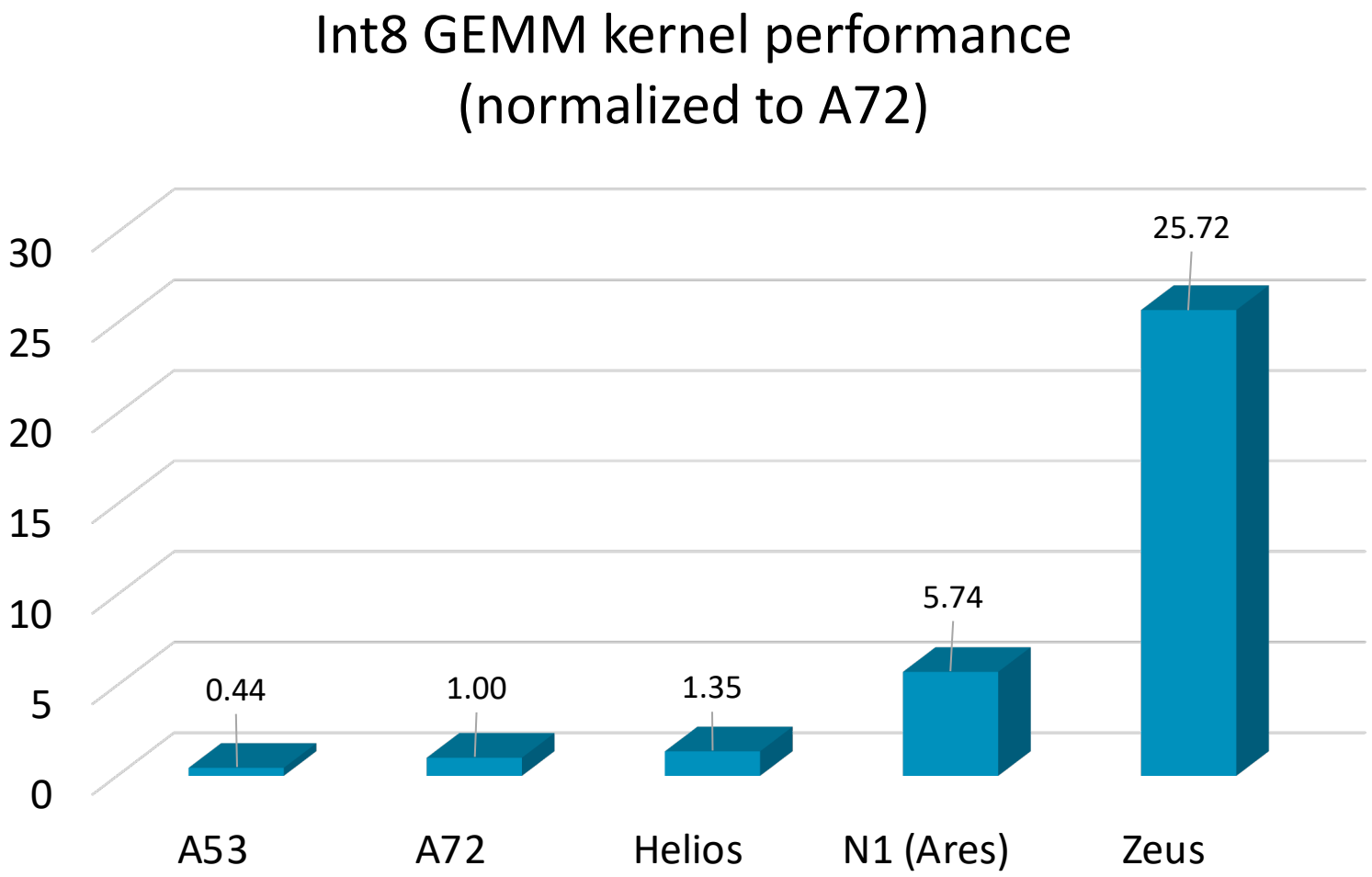
Caffe

- Community maintained

Theano (EOL)

- Community maintained

Increasing ML performance over CPU generations



A72
2x ML performance improvement over Cortex-A53

Helios
>3x ML performance improvement over Cortex-A53 (First Multi-threaded CPU)

N1
>5x ML performance improvement over Cortex-A72 (PPA leadership & ML enhancements)

Zeus
>25x ML performance improvement over Cortext-A72 (Breakthrough ML performance)

arm

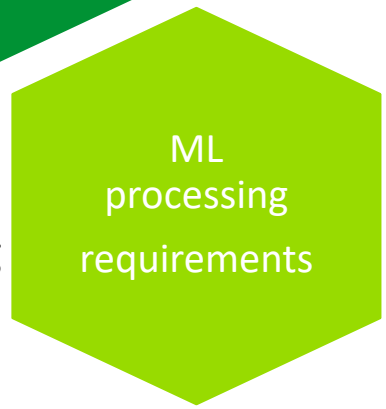
On-CPU ML processing

Primary drivers for on-CPU ML

ML is evolving – design optimization space for accelerators not sufficiently converged

ML programming looks like CPU programming as much as possible

Sub-optimal to integrate specialized accelerators for intermittent ML processing



Features enhancing ML performance on Arm CPUs

- Arch
  - Dot product instructions (v8.0 – v8.4)
- Arch
  - Matrix-multiply-and-accumulate instructions (*MMLA*) (v8.6)
- Micro Arch
  - SVE vector length
- Arch
  - Bfloat16 support (v8.6)

arm

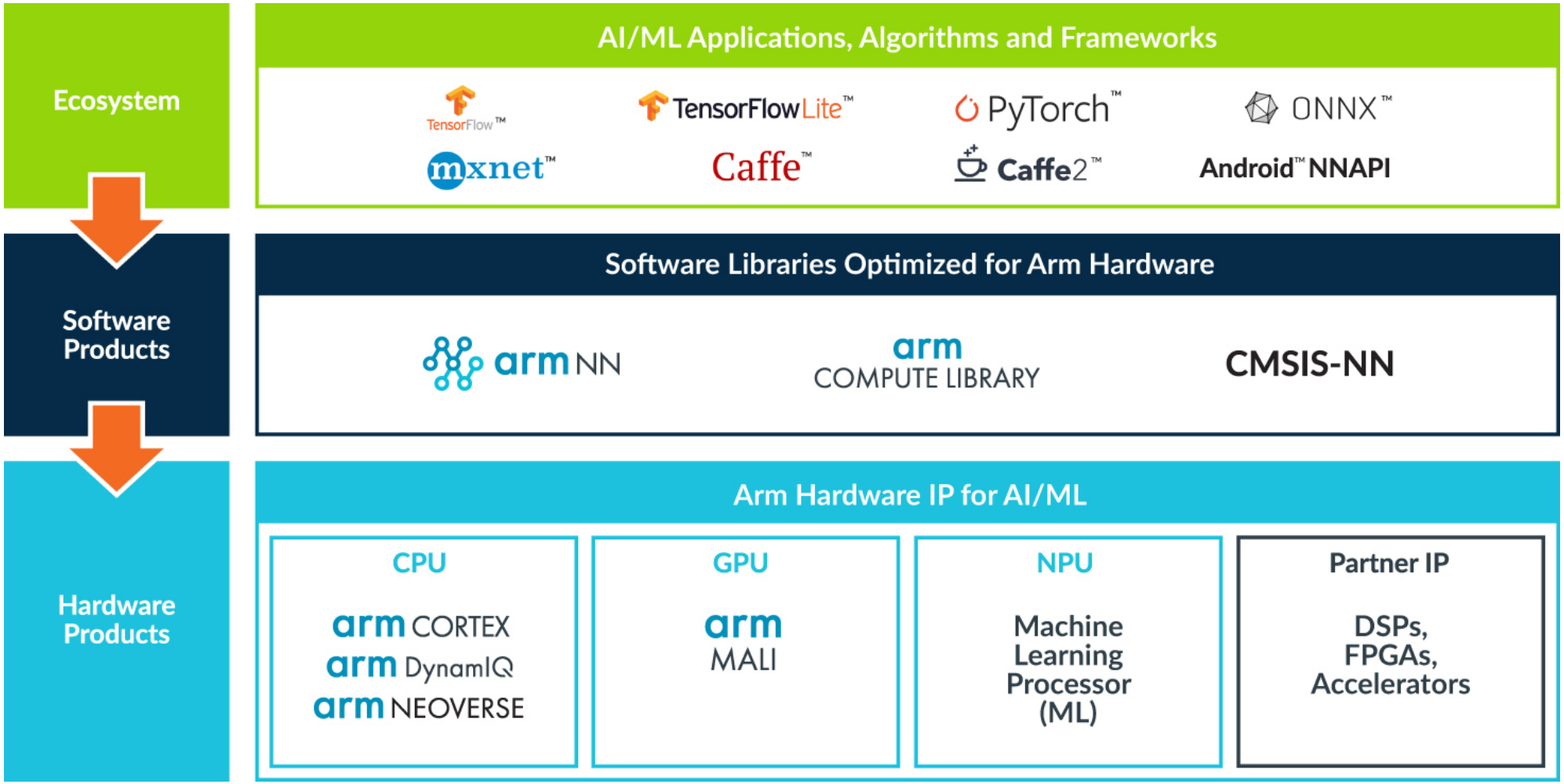
On-CPU Machine Learning

Easy to use, high performing ML software stack on Aarch64 using ML-specific CPU features

Easy to use	Wide variety of inference and training workloads	Using Arm architecture features	On the latest Aarch64 hardware
Container images and Python Packages	Image classification	Large core count	Arm Neoverse N1, Zeus, Poseidon
Popular ML frameworks support Arm as a first-class citizen	Object detection	INT8, Bfloat16, FP16, FP32	Marvell ThunderX2
		SVE / SVE2	Fujitsu A64FX
		Matrix Multiplier Extension	

Machine Learning and Artificial Intelligence

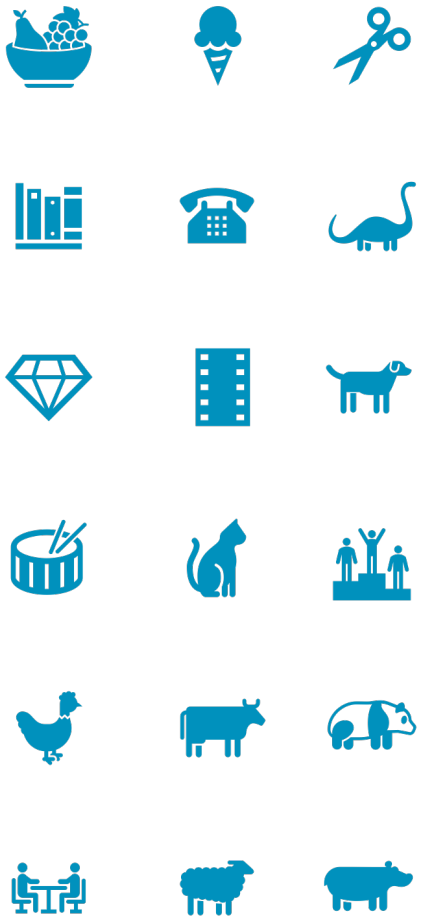
Machine Learning and Artificial Intelligence



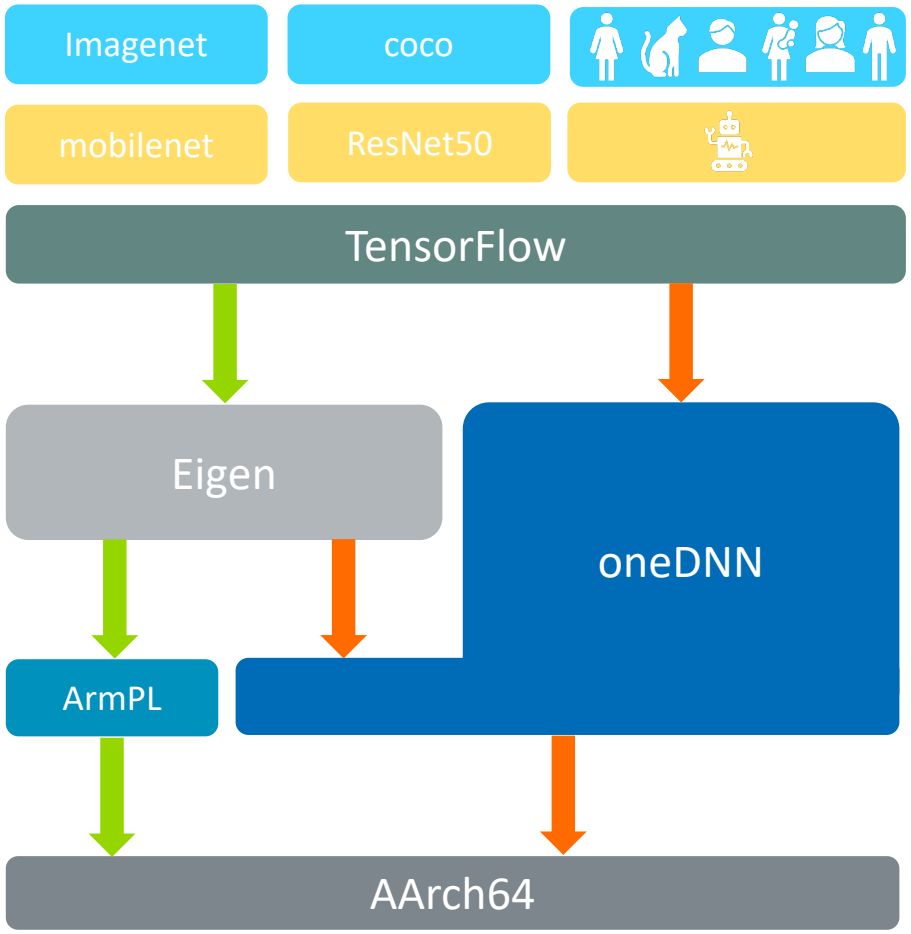
arm

ML Frameworks on server-class Aarch64 platforms

- Recent effort to enable server-scale on-CPU ML workloads on AArch64
- Build guides for key frameworks available:
  - Tensorflow - https://gitlab.com/arm-hpc/packages/wikis/packages/tensorflow
  - PyTorch - https://gitlab.com/arm-hpc/packages/wikis/packages/pytorch
  - MXNET - https://gitlab.com/arm-hpc/packages/wikis/packages/mxnet
  - And guides for key dependencies: CPython; NumPy etc.
- Currently focusing on inference problems
- ML Perf (https://mlperf.org) for realistic workloads.



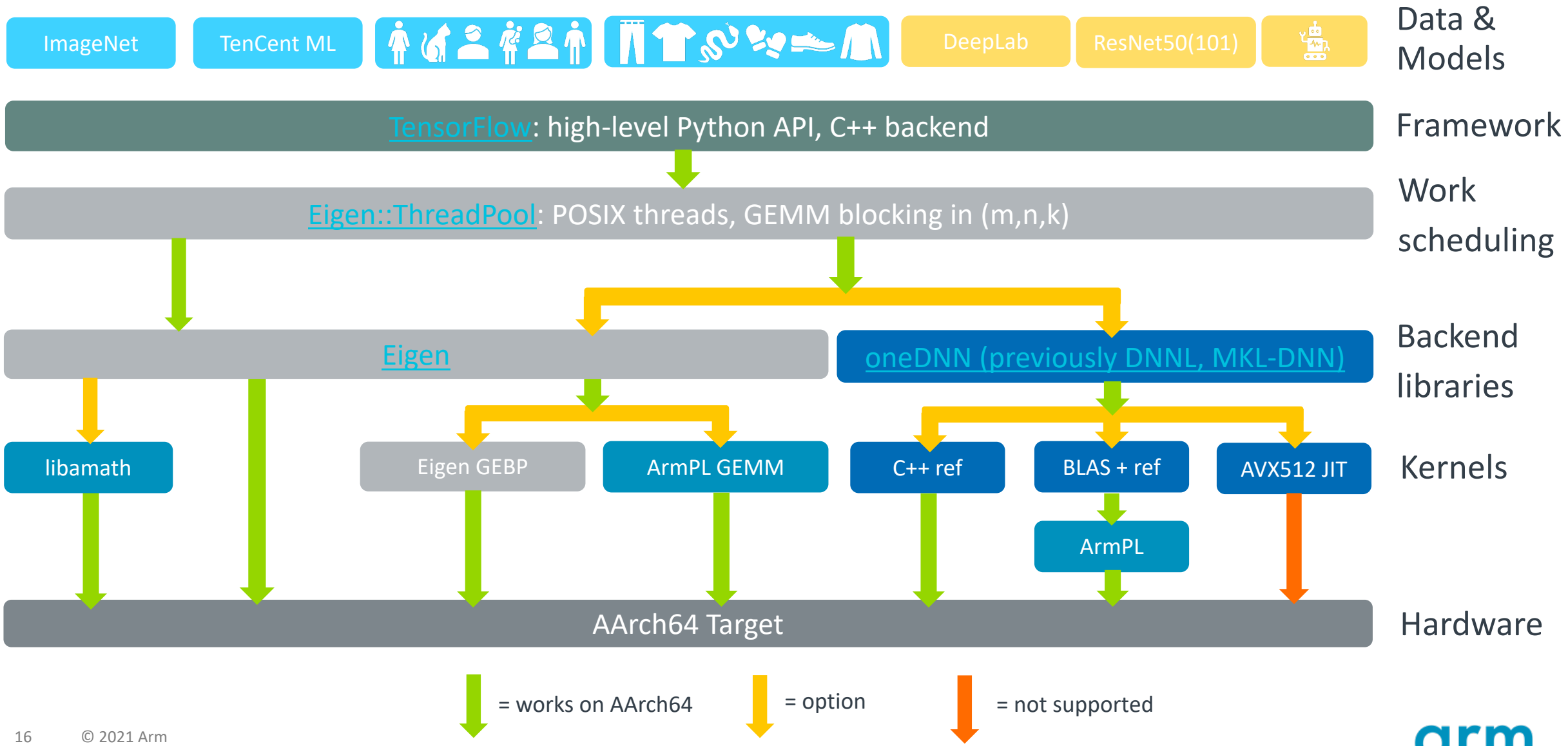
TensorFlow and maths libraries: on AArch64



- Arm Performance Libraries
  - Micro- architecture optimized
  - Targeting server class cores
  - High release cadence
- GEMMs at the core of matmul and convolutions
- Leveraging ArmPL has potential to deliver optimal performance in key kernels for on-CPU, server scale ML workload.

arm

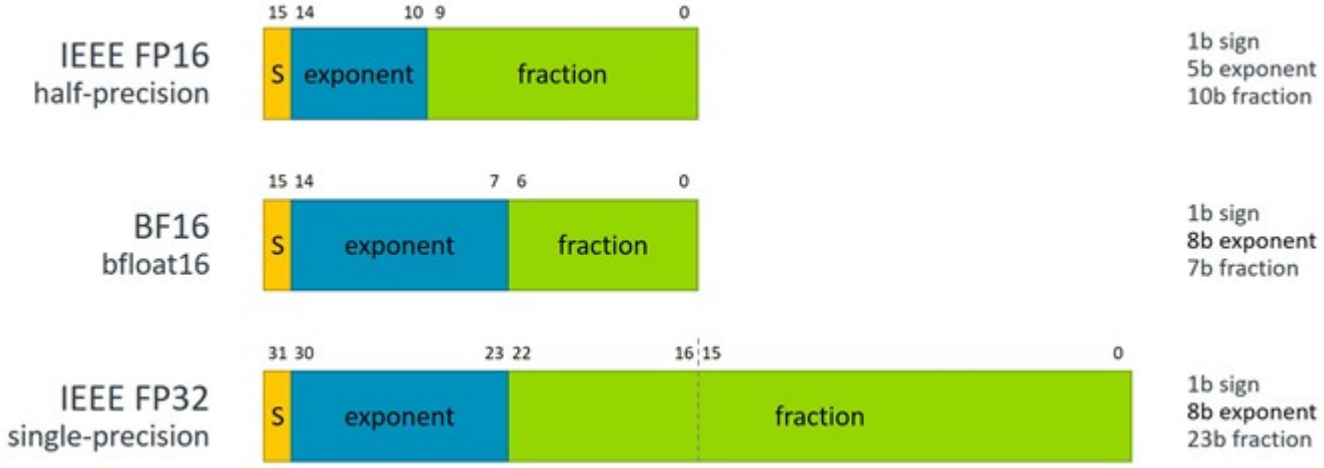
Where to optimise: TensorFlow and its backend



ISA Developments

New Data Type Support: BFloat16

- New addition to Armv8.6-A
  - Adds support for BF16
- Instructions for NEON and SVE
  - Including:
    - **BFDOT:** Dot Product (1x2)x(2x1)
    - **BFMMLA:** Mat Multiply (2x4)x(4x2)
- Significant performance gains
  - ML training and inference workloads
- Supported in Arm libraries
  - Arm NN and Arm Compute Libraries



arm

FMMLA: High Performance Matrix Multiplication

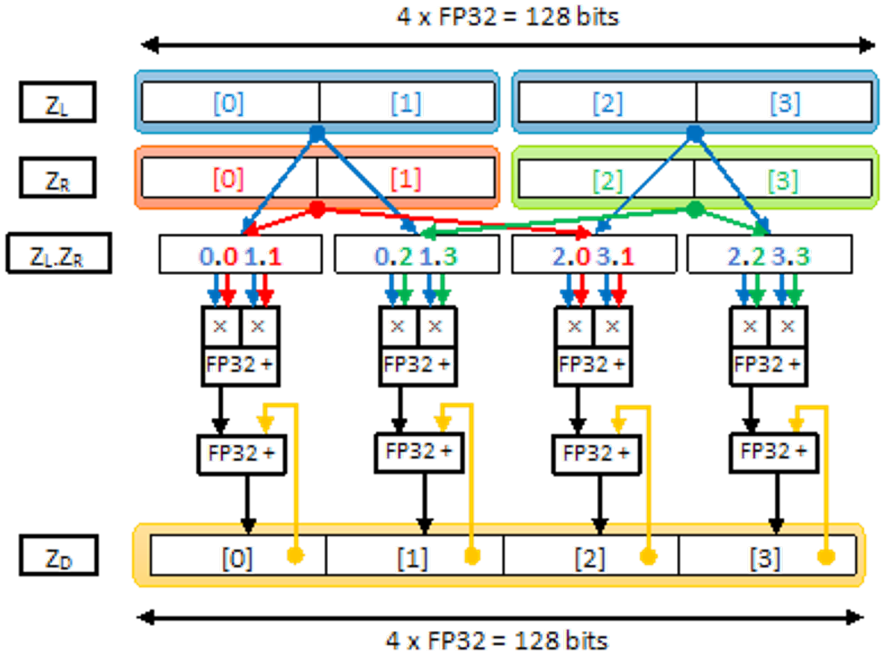
- Added to Armv8.6
  - NEON support for INT and BF16
  - FMMLA instructions for FP (SVE)

FMMLA <Zda>.S, <Zn>.S, <Zm>.S

FMMLA <Zda>.D, <Zn>.D, <Zm>.D

- 2x2 matrix multiplication
  - Works on multiple of vector granules
  - 2x2xFP32 = 128-bit granules
  - Assumes vector length is multiple
- May require layout transformations
  - Outer loop to avoid cost
- Will accelerate maths libraries

D[0] += (L[0] * R[0]) + (L[1] * R[1])
D[1] += (L[0] * R[2]) + (L[1] * R[3])
D[2] += (L[2] * R[0]) + (L[3] * R[1])
D[3] += (L[2] * R[2]) + (L[3] * R[3])



Thank You
Danke
Gracias
谢谢
ありがとう
Asante
Merci
감사합니다
धन्यवाद
Kiitos
شكرًا
ধন্যবাদ
תודה