arm AI

Running Machine Learning on Arm's Ethos-U55 NPU

Arm

George Gekov

2nd November 2021

arm

Welcome!

Tweet us: @ArmSoftwareDev -> #AIVTT

Check out our Arm Software Developers YouTube channel

Signup now for our next AI Virtual Tech Talk: developer.arm.com/techtalks

Our upcoming Arm AI Tech Talks

Date	Title	Host
November 2nd	Getting started with running Machine Learning on Arm Ethos-U55	Arm
November 16th	Hands-on workshop with the Arm ML Embedded Evaluation kit for Ethos-U55	Arm
November 30th	Getting started with Arm NN on Android, in just 5 minutes	Arm
December 14th	Improve PyTorch App Performance with Android NNAPI Support	Arm

Visit: developer.arm.com/techtalks

Presenter

George Gekov

- Software engineer in Arm’s Machine Learning team
- Develop ML applications on Arm silicon
- Previously, part of Arm’s IoT team

Does anybody enjoy their ML software running slowly?

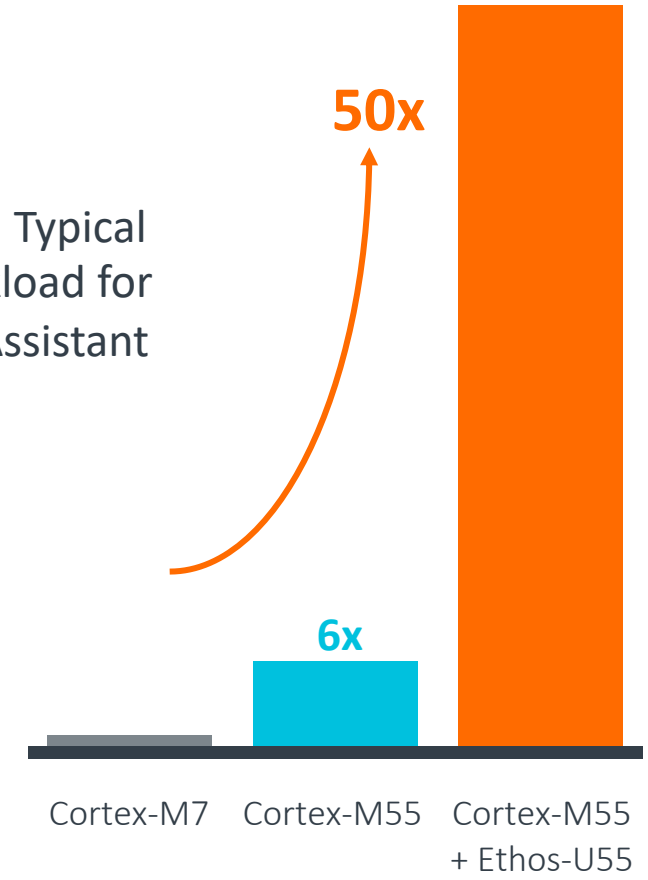
Agenda

- What is the Arm Ethos-U55 microNPU?
- What software stack to use on the Ethos-U55?
- How to optimise a neural network ?
- Demo!

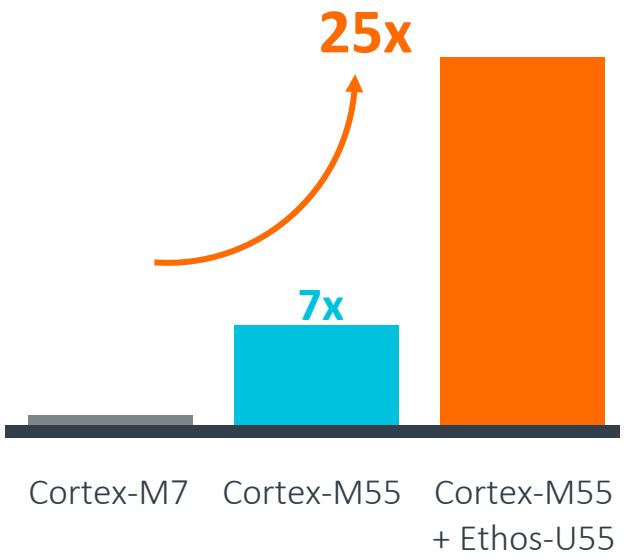
Ethos-U55: First microNPU for Cortex-M CPUs

Example: Typical ML Workload for a Voice Assistant

Speed to inference



Energy efficiency



- ✓ Faster responses
- ✓ Smaller form-factors
- ✓ Improved accuracy

Latency and energy spent for all tasks listed combined: voice activity detection, noise cancellation, two-mic beamforming, echo cancellation, equalizing, mixing, keyword spotting, OPUS decode, and automatic speech recognition.

arm

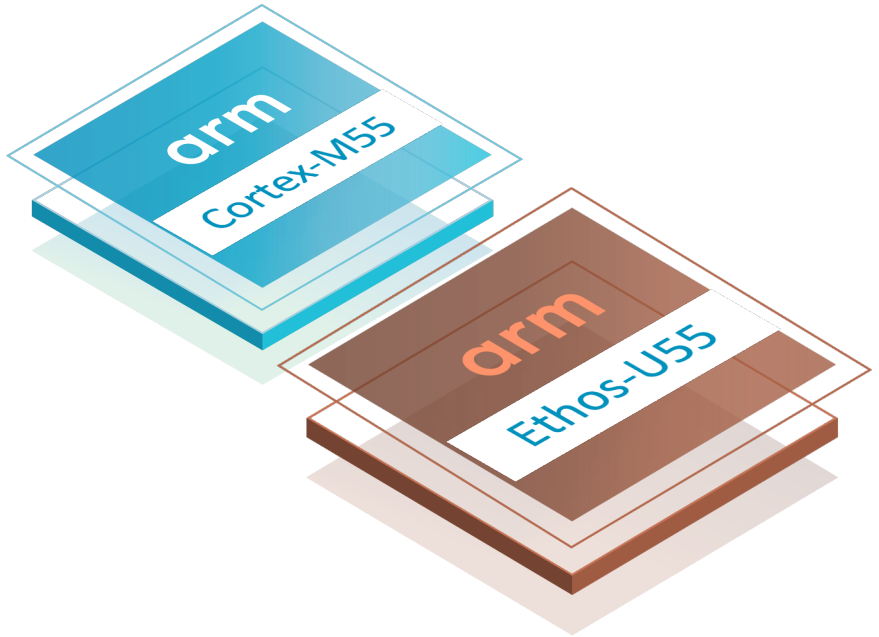
Develop for the Arm Ethos-U55 without a development board!

How to create software applications when NPU silicon is not commercially available yet?

Arm Virtual Hardware

- Fixed Virtual Platform(FVP) – digital twin of a development board with Ethos-U55 & Cortex-M55
- Corstone-300(sse-300), available as part of Arm Virtual Hardware
- MAC = Multiply Accumulate
  - Ethos-U55 supports 32,64,128,256 MACs

Arm Cortex-M55 and Arm Ethos-U55

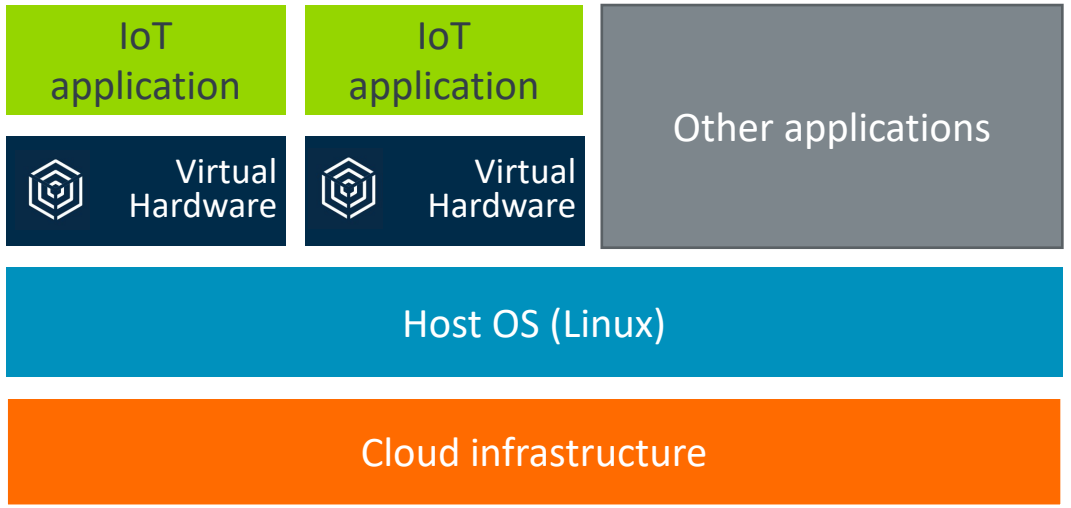


arm

What's Arm Virtual Hardware?

Virtual Hardware Targets are the IoT equivalent of Virtual Machines

- An Arm Virtual Hardware Target is a functionally accurate representation of a physical SoC, simulating its software-visible behavior
- Runs as a simple application in a Linux environment for easy scalability in the cloud
- Remove dependency from RTL or silicon availability
- Available as a public beta for multiple configurations of the Arm Corstone-300 subsystem, incorporating the Cortex-M55 CPU and Ethos-U55 uNPU.

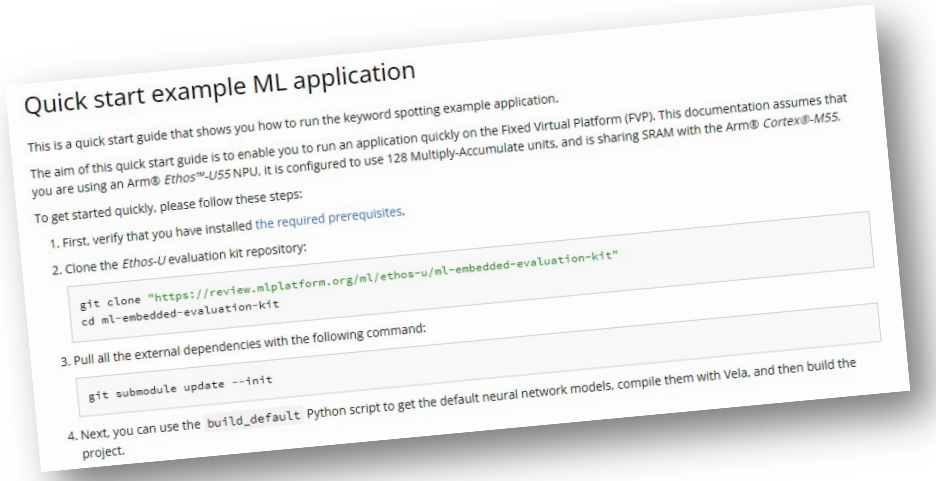


www.arm.com/virtual-hardware

ML embedded evaluation kit

Open-source, Apache 2.0

- https://review.mlplatform.org/plugins/gitiles/ml/ethos-u/ml-embedded-evaluation-kit
- Ready to use applications for Arm Ethos-U55



Quick start example ML application

This is a quick start guide that shows you how to run the keyword spotting example application.

The aim of this quick start guide is to enable you to run an application quickly on the Fixed Virtual Platform (FVP). This documentation assumes that you are using an Arm® *Ethos™-U55* NPU. It is configured to use 128 Multiply-Accumulate units, and is sharing SRAM with the Arm® *Cortex®-M55*.

To get started quickly, please follow these steps:

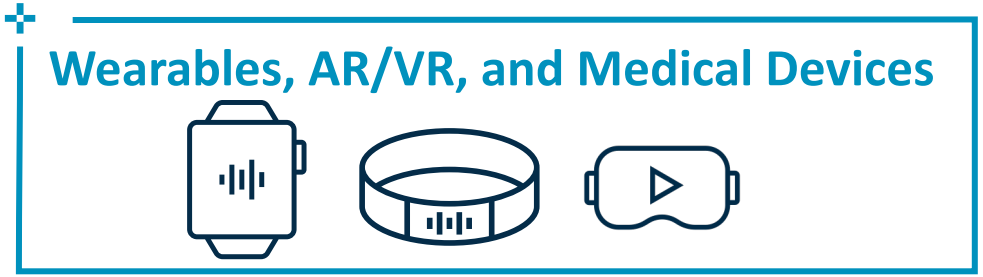
1. First, verify that you have installed the required prerequisites.
2. Clone the *Ethos-U* evaluation kit repository:

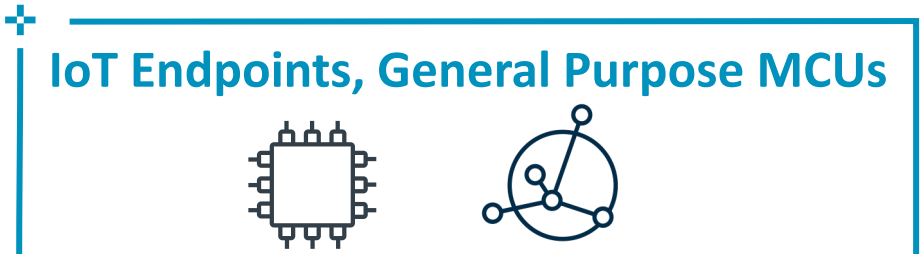
```
git clone "https://review.mlplatform.org/ml/ethos-u/ml-embedded-evaluation-kit"
cd ml-embedded-evaluation-kit
```

3. Pull all the external dependencies with the following command:

```
git submodule update --init
```

4. Next, you can use the `build_default` Python script to get the default neural network models, compile them with Vela, and then build the project.





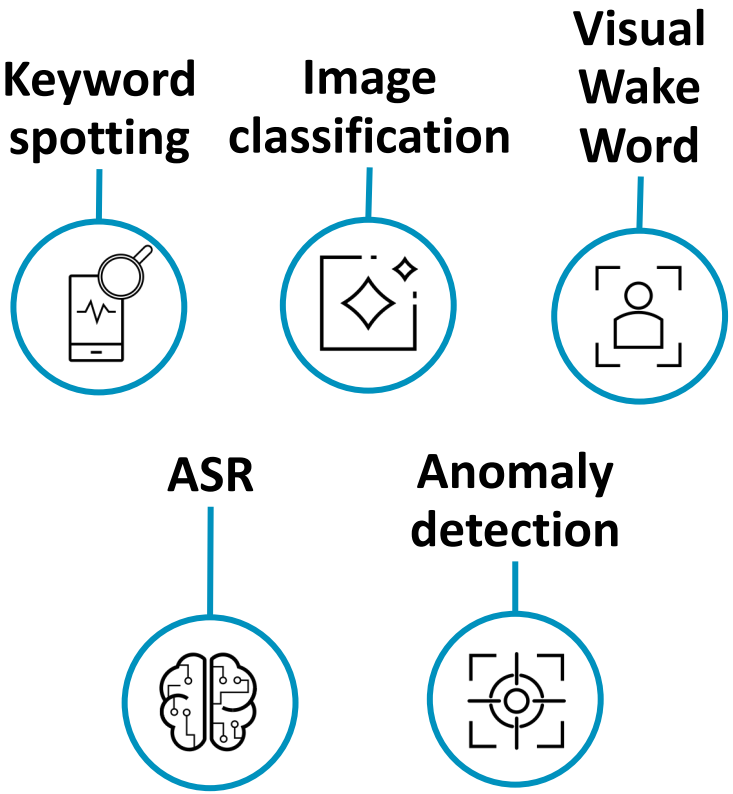
Why use the ML embedded evaluation kit?

Three main benefits

Performance evaluation

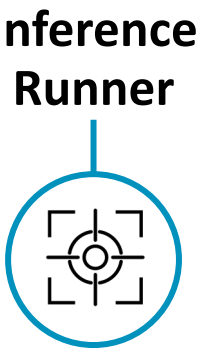
- Number of NPU cycles
- Amount of memory transactions

Software stack evaluation

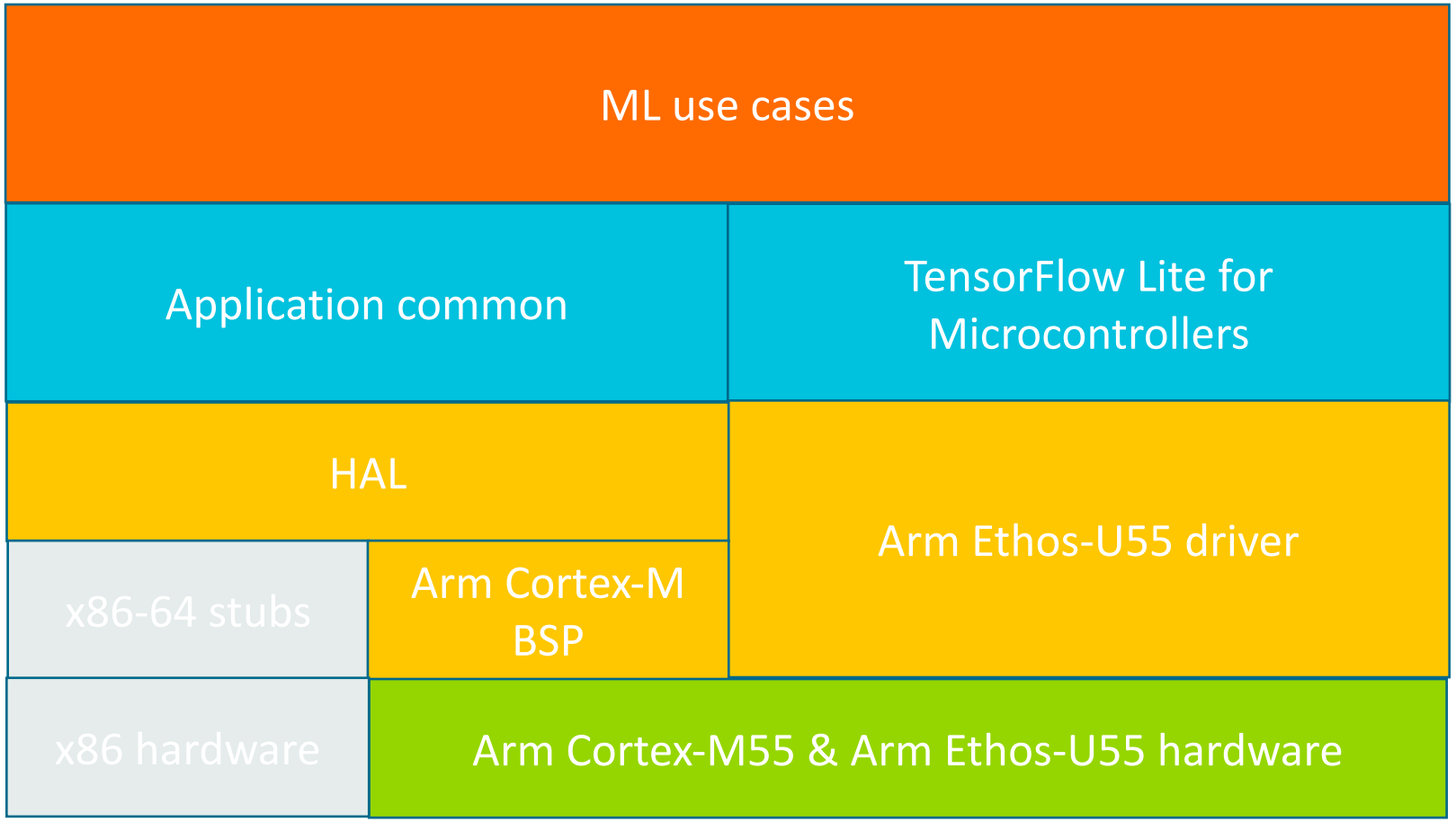


Custom workflow

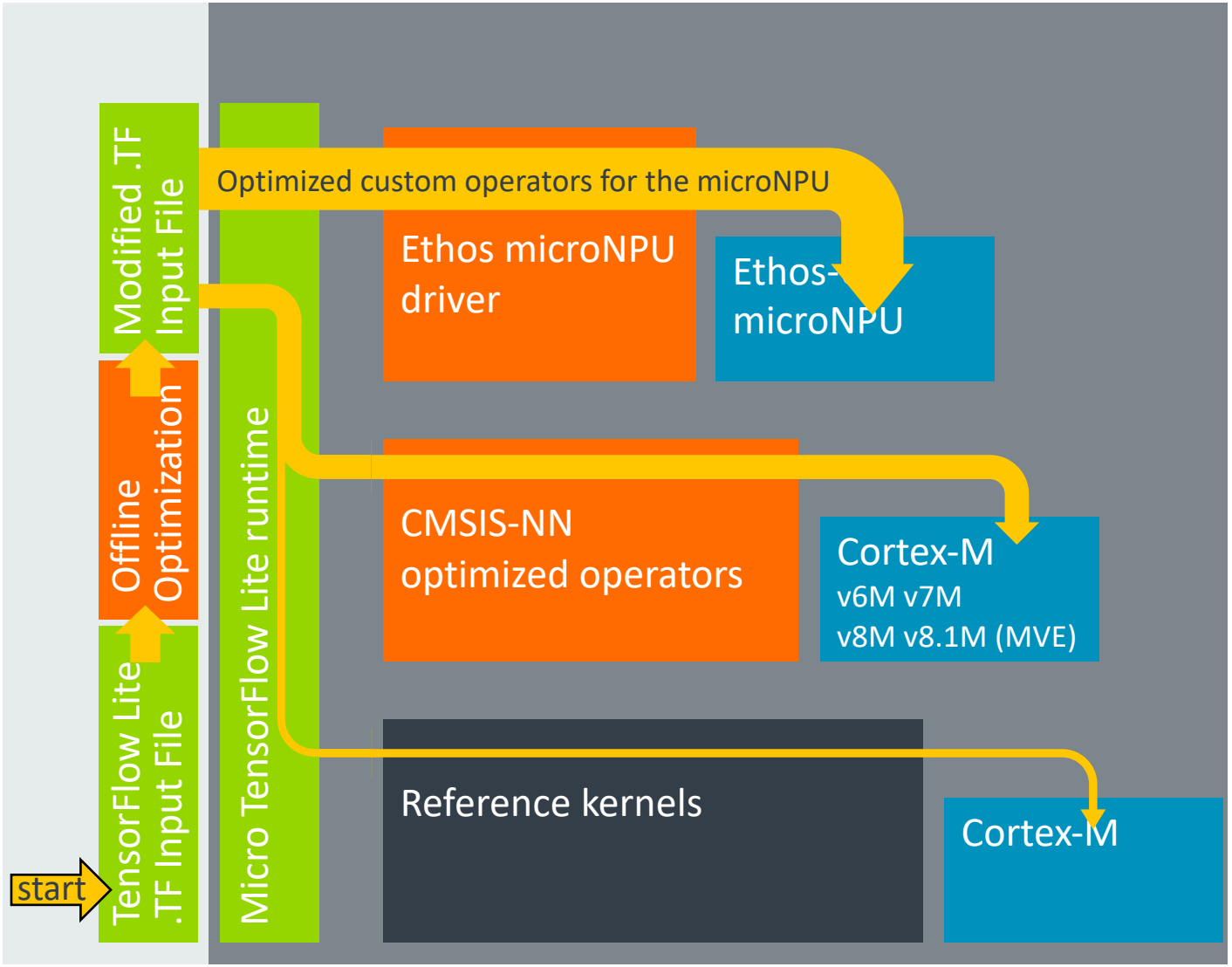
- Test custom NN performance on the Ethos-U55
- Framework to implement new ML use-cases



Software stack evaluation



TFLμ Operator Support – CMSIS-NN and Ethos-U NPU



Supported operators

Abs, Add, Average_Pool_2D, Concatenation, Conv_2D, Depthwise_Conv_2D, Fully_Connected, Leaky_ReLu, Logistic, Maximum, Max_Pool_2D, Minimum, Mul, Pack, Quantize, ReLu, ReLu6, ReLu_N1_to_1, Reshape, Resize_Bilinear, Slice, SoftMax, Split, Split_V, Squeeze, Strided_Slice, Sub, TanH, Transpose_Conv, Unpack and others. See SUPPORTED_OPS.md (generated from vela)

30+ operators

Optimized operators

The library has a roadmap of Quarterly releases to expand scope and improve performance

Fallback to reference kernels

80+ operators

arm

Vela compiler

- Open source python tool: https://review.mlplatform.org/admin/repos/ml/ethos-u/ethos-u-vela
- Pypi: https://pypi.org/project/ethos-u-vela/
  pip3 install ethos-u-vela
- Top level functionality:
  - **Parses a model**
  - **Optimises the graph**
  - **Tensor allocation**
  - **Command stream generation**
  - **Saves optimised model**
- Configurable behaviour: https://review.mlplatform.org/plugins/gitiles/ml/ethos-u/ethos-u-vela/+/refs/heads/master/OPTIONS.md
- Supported ops: https://review.mlplatform.org/plugins/gitiles/ml/ethos-u/ethos-u-vela/+/refs/heads/master/SUPPORTED_OPS.md

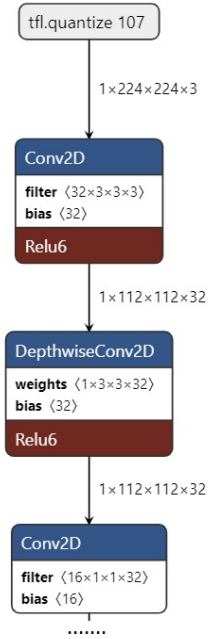
Vela workflow

Initial model

Vela configuration

Call Vela

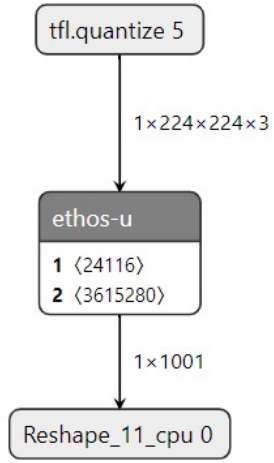
Optimised model



Initial model

$ vela mobilenet_v2_1.0_224_INT8.tflite --accelerator-config=ethos-u55-128 --optimise Performance --config vela.ini --memory-mode=Shared_Sram --system-config=Ethos_U55_High_End_Embedded

- Input: tflite file & vela configuration
- Output: tflite file
- Input model:
  - Can run on CPU (with CMSIS kernels if possible),
  - Cannot run on microNPU
- Output model:
  - "Ethos-u" op cannot run on CPU but can run on microNPU
  - All fallback ops run on CPU (with CMSIS kernels if possible)

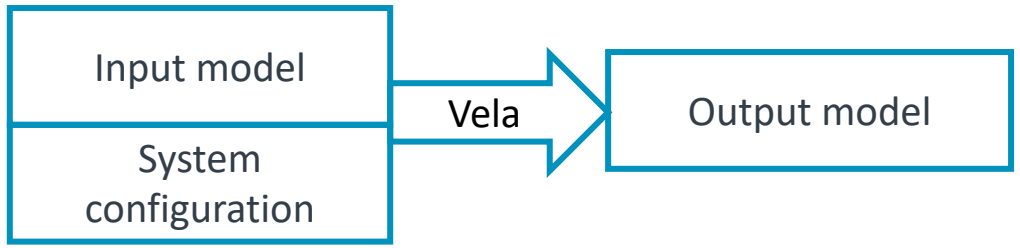


Vela model

Vela configuration

What needs to be configured ?

- Memory latencies and bandwidths (Deeply embedded, high-end,..)
- microNPU configuration(32,64,128,256 MACs)
- Memory mode
- Example configuration file:
  - https://review.mlplatform.org/plugins/gitiles/ml/ethos-u/ethos-u-vela/+/refs/heads/master/vela.ini



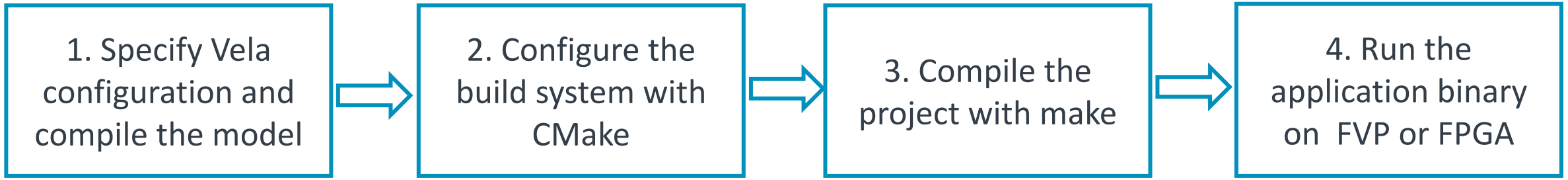

```
; System Configuration

; Ethos-U55 High-End Embedded: SRAM (4 GB/s) and Flash (0.5 GB/s)
[System_Config.Ethos_U55_High_End_Embedded]
core_clock=500e6
axi0_port=Sram
axi1_port=OffChipFlash
Sram_clock_scale=1.0
Sram_burst_length=32
Sram_read_latency=32
Sram_write_latency=32
OffChipFlash_clock_scale=0.125
OffChipFlash_burst_length=128
OffChipFlash_read_latency=64
OffChipFlash_write_latency=64
```

Run one of the available applications on the Ethos-U55 microNPU

Quick way to run an application & how to do a non-default build

- For a default build – use `build_default.py` script

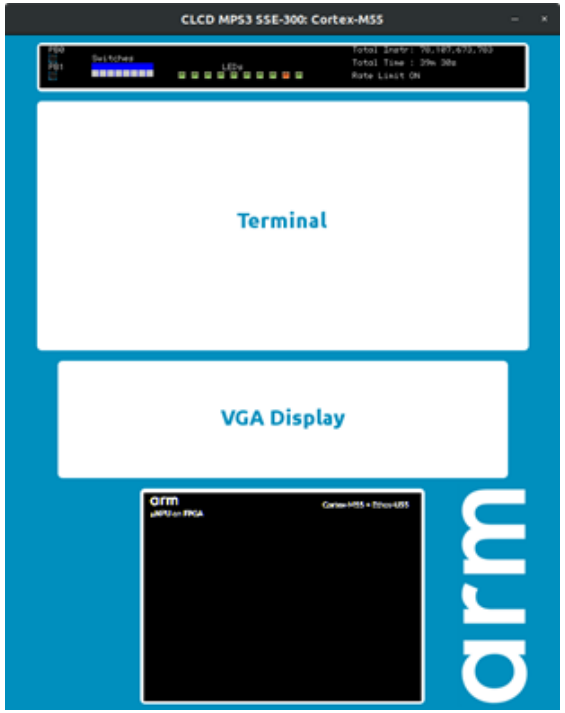
- For a non-default build



What is cycle accurate & what is not cycle accurate?

Fixed Virtual Platform(Arm Virtual Hardware)

- Arm Ethos-U55: cycle approximate
- Arm Cortex-M55: functionally accurate



MPS3 FPGA

- Arm Ethos-U55: cycle accurate
- Arm Cortex-M55: cycle accurate

arm

Demo time

Summary

- What is the Arm Ethos-U55 microNPU?
- What software stack to use?
- How can you optimise a neural network for the Arm Ethos-U55 microNPU?
- How can you run an application on the Arm Ethos-U55 microNPU?

Try it yourself!

- Download the source code
- Try running an application yourself
- If you have a custom neural network, try running it on the Ethos-U55 and tell us how you get on https://discuss.mlplatform.org/c/ml-embedded-evaluation-kit/
- Access Arm Virtual Hardware (AVH) on AWS marketplace as Amazon Machine Image – www.arm.com/virtual-hardware
  - Attend AI Tech Talk on Nov 16th for hands-on workshop with AVH
  - 100hrs of free AWS EC2 CPU credits for first 1,000 qualified users

arm

Q & A

arm

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