Computer Vision & Machine Learning with OpenVX™

Kiriti Nagesh Gowda
OpenVX Chair | The Khronos Group
Staff Engineer | AMD

AMD

Agenda

- OpenVX Overview
- OpenVX 1.3 - Highlights & New Features
- Common Questions About OpenVX
- Conformant OpenVX Implementations
- OpenVX Sample Implementation
- OpenVX for Raspberry Pi
- An OpenVX Cross-Platform Application: Case Study
- Conclusion

KHRONOS GROUP

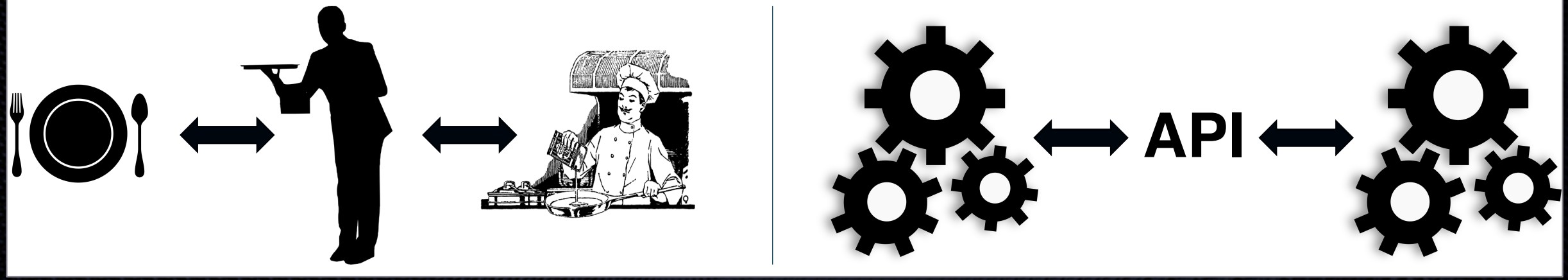
OpenVX

OpenVX Overview

OpenVX™ is an **open**, **royalty-free** API standard for a **cross-platform** acceleration of computer vision applications

What is an API?

Application Programming Interface



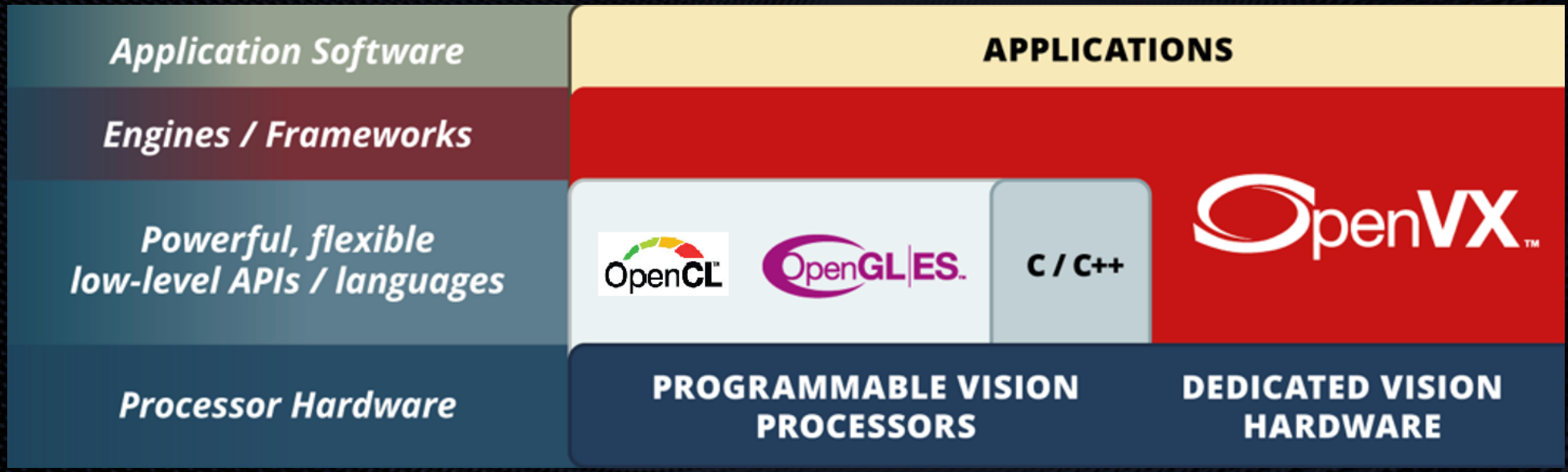
OpenVX Overview

Why APIs are important

- **Building Blocks**
- **Speeds up development**
- **Portability**
- **Innovation**

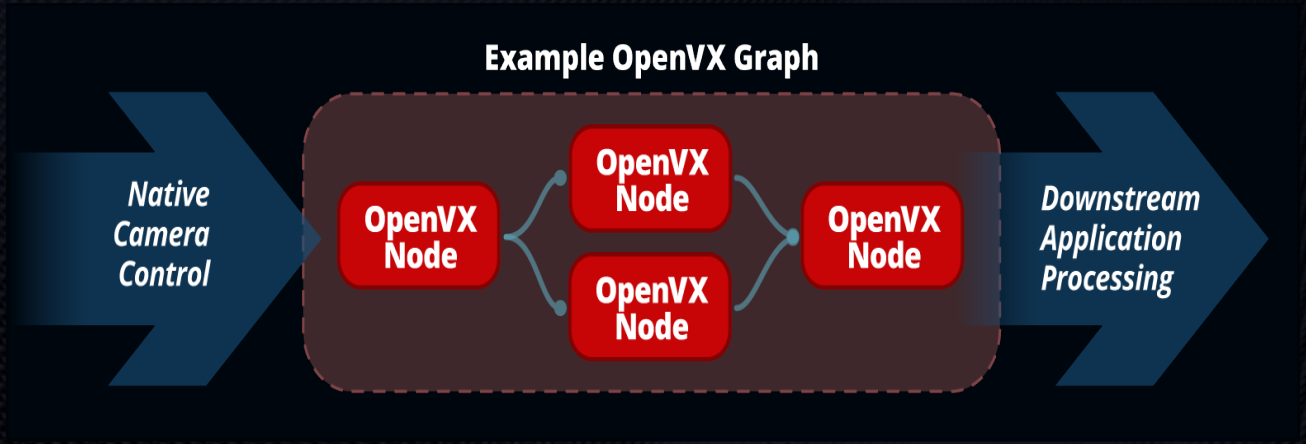
OpenVX™ enables portable, performance, and power-optimized computer vision processing, especially important in embedded and real-time use cases

OpenVX™ is unique in being the only vision API shipped as an optimized driver



OpenVX Overview

OpenVX™ Graph Framework



- OpenVX allows graph-level processing **optimizations**, which lets implementations to **fuse nodes** when possible to achieve better overall **performance**
- The graph also allows for auto graph-level **memory optimizations** to achieve a **low memory footprint**
- OpenVX graph-optimized workloads can be **deployed** on a **wide range of computer hardware**, including small embedded CPUs, ASICs, APUs, discrete GPUs, and heterogeneous servers

AMD

OpenVX 1.3 - Highlights & New Features

OpenVX™ 1.3 – Released on October 22nd, 2019

- Enable **deployment flexibility** while **avoiding fragmentation**, OpenVX 1.3 defines several **feature sets** that are targeted at common embedded use cases
- Hardware vendors can include **one or more** complete feature sets in their implementations to meet the needs of their customers and be fully conformant
- The flexibility of OpenVX enables deployment on a **diverse range** of accelerator architectures, and feature sets are expected to dramatically increase the breadth and diversity of available OpenVX implementations

OpenVX 1.3 - Highlights & New Features

The defined OpenVX 1.3 feature sets include:

- **Graph Infrastructure** - baseline for other Feature Sets
- **Vision** - core vision functionality
- **Enhanced Vision** - functions introduced in OpenVX 1.2
- **Neural Network Inferencing** - including tensor objects
- **NNEF Kernel import** - including tensor objects
- **Binary Images** – one bit images
- **Safety Critical** - reduced features to enable easier safety certification

Common Questions About OpenVX

Question: Is OpenVX an Open-Sourced Library?

- Callable API implemented, optimized OpenVX drivers are created, optimized, and shipped by processor vendors

Question: Must I pay royalties and licensing fee to use OpenVX?

- Protected under Khronos IP Framework - Khronos members agree not to assert patents against API when used in Conformant implementations

Question: Must I be a Khronos member to use OpenVX?

- Khronos members and non-members develop conformant implementations to be used by all

Common Questions About OpenVX

Question: Is OpenVX Functions limiting?

- Tight focus on dozens of core hardware accelerated functions plus extensions and accelerated custom nodes
- Users can create custom nodes and vendors can create custom extensions, with some cost in terms of portability

Question: Are Implementations different in functionality?

- Core API provides identical functionality across platforms due to strict conformance testing
- Implementations must pass Khronos Conformance Test Suite to use trademark
- Conformance does not extend to vendor extensions and custom nodes created by users
- Popular vendor extensions adopted as Khronos extensions with conformance testing

Conformant OpenVX Implementations

- Conformant Implementations **must pass** exhaustive conformance test suite
- Hardware vendors provide optimized OpenVX drivers, architected to get the best performance from their silicon architecture and ready for developers to use

Conformant Implementations of OpenVX from the following vendors:

OpenVX Sample Implementation

Open-Sourced OpenVX Sample Implementation available on **GitHub** - https://github.com/KhronosGroup/OpenVX-sample-impl

The purpose of this software package is to provide a sample implementation of the OpenVX 1.3 Specification that passes the conformance test. It is NOT intended to be a reference implementation

Optimized OpenVX libraries available from **vendor implementations**

IS:

- **passing OpenVX 1.3 conformance tests**

IS NOT:

- a reference implementation
- optimized
- production ready

OpenVX - NNEF Kernel Import

Provide a **minimum** set of functions to **import** and **execute neural networks** described in the **NNEF standard format**.

Applications using this feature set will use the **vxImportKernelFromURL** function to import an **NNEF file** at the location of the **URL** to create an OpenVX kernel representing the neural network.

This kernel can subsequently be used to create a **node in an OpenVX graph**, which can be executed using the normal OpenVX functions from the Base Feature Set. The **inputs** and **outputs** of the neural network node will be vx_tensor objects.

This feature set is dependent on the **Base feature set** and the **tensor data object**, which must also be supported in order to support this feature set.

The name of this feature set is **vx_khr_nnef_import**

AMD

OpenVX for Raspberry Pi

July 2020

- The **Khronos Group** and the **Raspberry Pi Foundation** have worked together to implement an **open-source implementation of OpenVX™ 1.3**, which passes the conformance on Raspberry Pi
- The open-source implementation passes the **Vision**, **Enhanced Vision**, **Neural Net**, & **NNEF Kernel Import** Conformance Profiles specified in OpenVX 1.3 on Raspberry Pi
- The Implementation is **NEON optimized**

Conformant hardware

- Raspberry Pi 3 Model B Rev 1.2
- Raspberry Pi 4 Model B Rev 1.2

KHRONOS GROUP

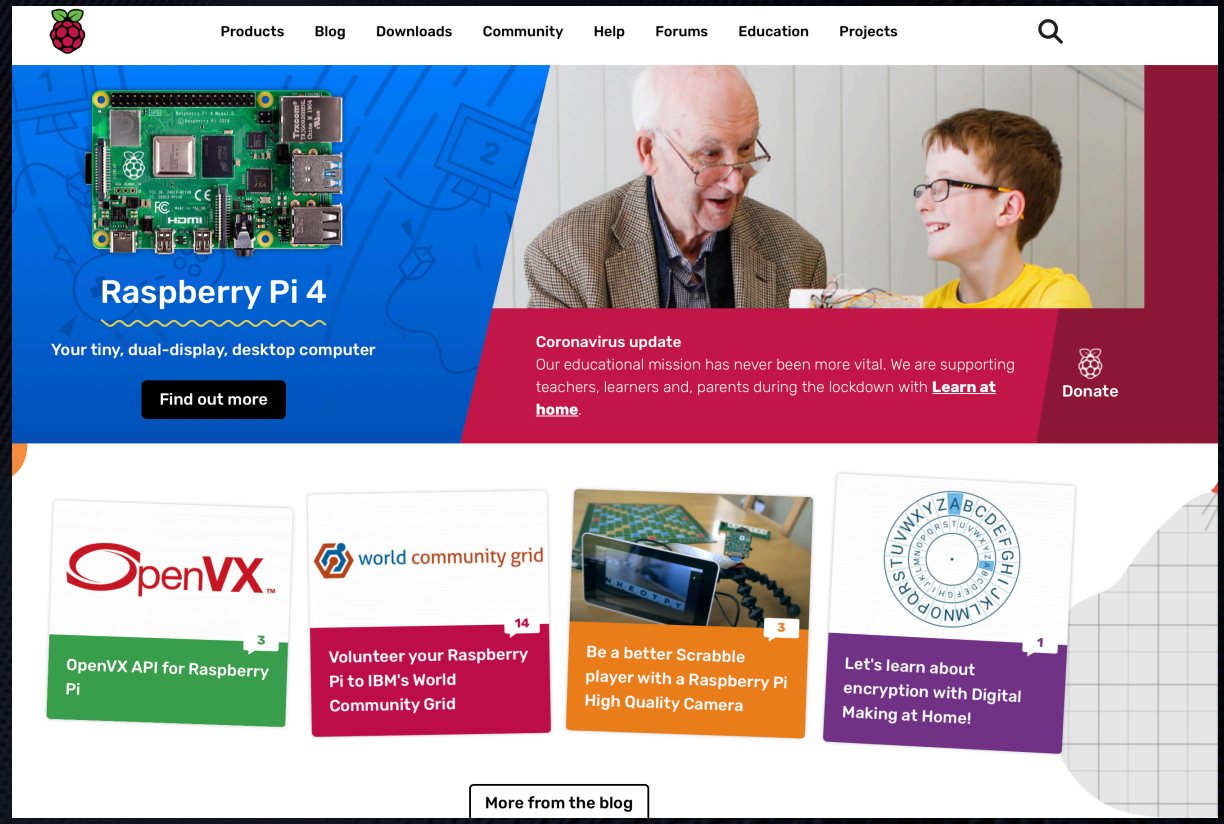
OpenVX™

OpenVX for Raspberry Pi

"Raspberry Pi is excited to bring the Khronos OpenVX 1.3 API to our line of single-board computers. Many of the most exciting commercial and hobbyist applications of our products involve computer vision, and we hope that the availability of OpenVX will help lower barriers to entry for newcomers to the field."

Eben Upton

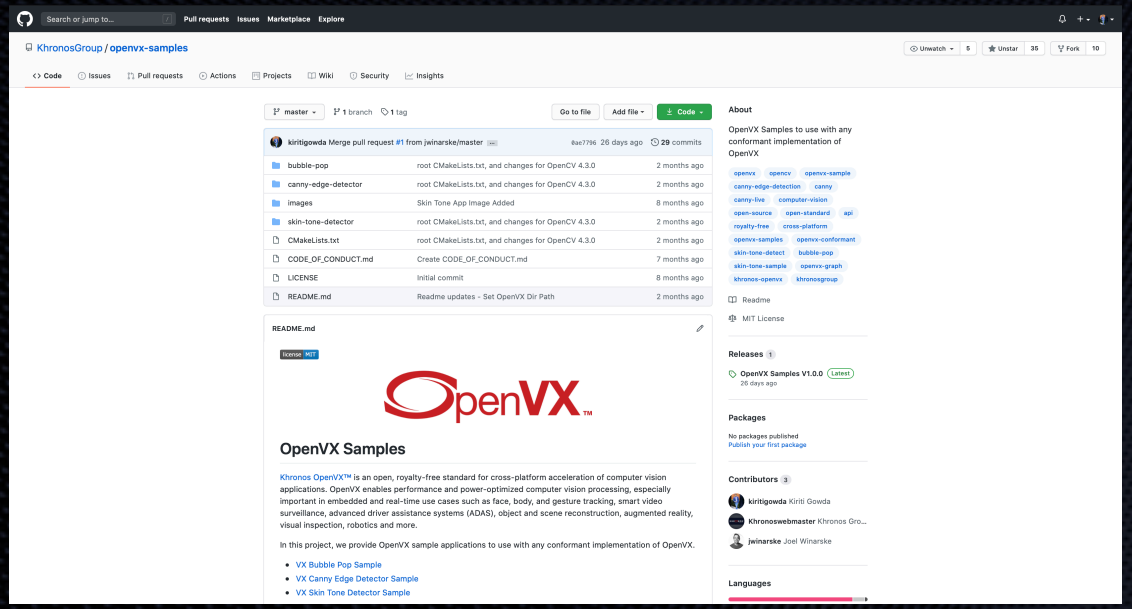
Chief Executive Raspberry Pi Trading

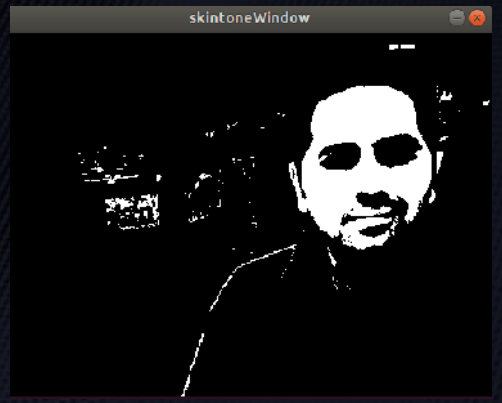


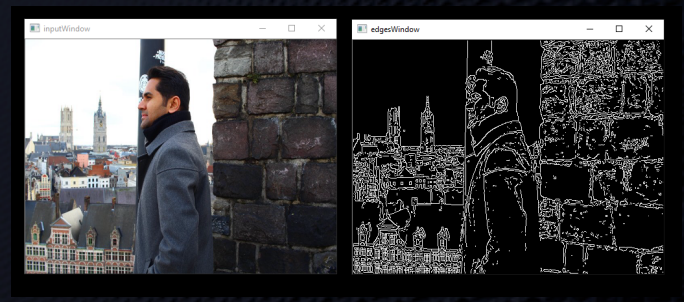
An OpenVX Cross-Platform Application: A Case Study

Open-Source OpenVX Samples

Open-source OpenVX sample applications, to use with **any conformant implementation** of OpenVX available on **GitHub**







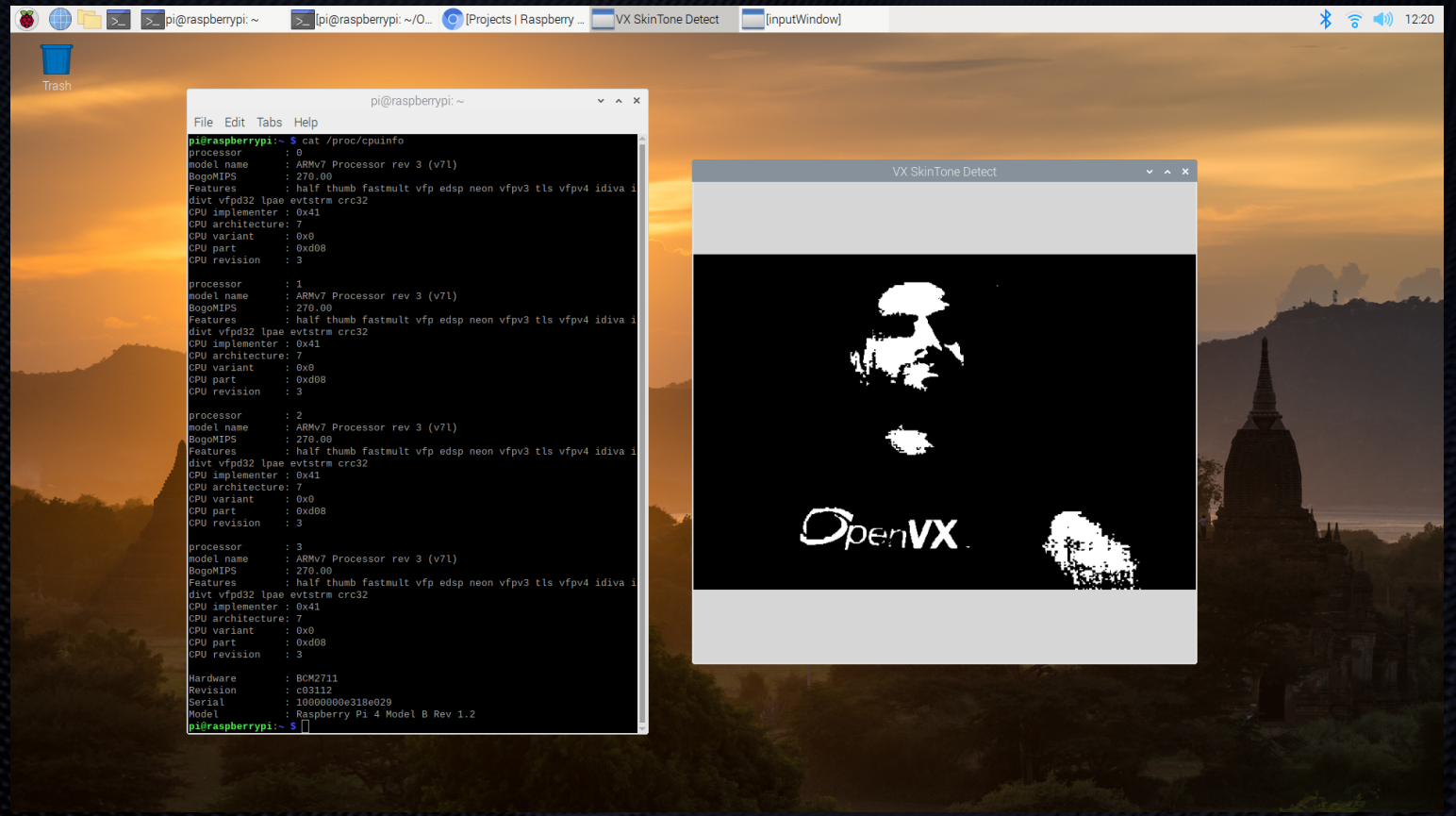
An OpenVX Cross-Platform Application: A Case Study

SkinTone Detector Sample

```
# extract R,G,B channels and compute R-G and R-B
node org.khronos.openvx.channel_extract input !CHANNEL_R R          # extract R channel
node org.khronos.openvx.channel_extract input !CHANNEL_G G          # extract G channel
node org.khronos.openvx.channel_extract input !CHANNEL_B B          # extract B channel
node org.khronos.openvx.subtract R   G   !SATURATE RmG              # compute R-G
node org.khronos.openvx.subtract R   B   !SATURATE RmB              # compute R-B
# compute threshold
node org.khronos.openvx.threshold R   thr95 R95                     # compute R > 95
node org.khronos.openvx.threshold G   thr40 G40                     # compute G > 40
node org.khronos.openvx.threshold B   thr20 B20                     # compute B > 20
node org.khronos.openvx.threshold RmG thr15 RmG15                   # compute RmG > 15
node org.khronos.openvx.threshold RmB thr0  RmB0                    # compute RmB > 0
# aggregate all thresholded values to produce SKIN pixels
node org.khronos.openvx.and R95   G40   and1                        # compute R95 & G40
node org.khronos.openvx.and and1  B20   and2                        # compute B20 & and1
node org.khronos.openvx.and RmG15 RmB0  and3                        # compute RmG15 & RmB0
node org.khronos.openvx.and and2 and3 output                        # compute and2 & and3 as output
```

An OpenVX Cross-Platform Application: A Case Study

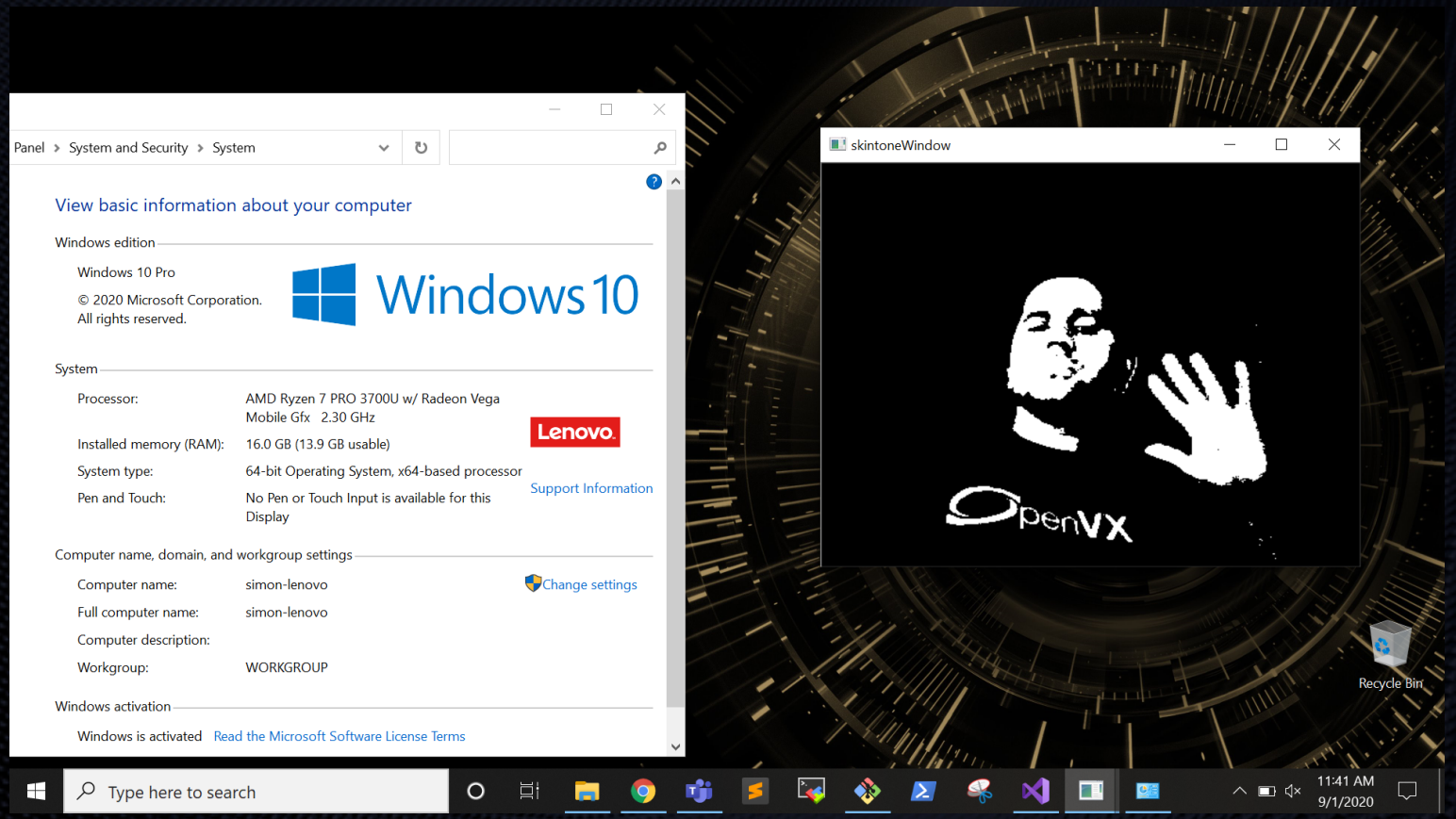
SkinTone Detector Sample - On Raspberry Pi 4 Model B Rev 1.2



* using open-source OpenVX Raspberry Pi Implementation for OpenVX Libraries

An OpenVX Cross-Platform Application: A Case Study

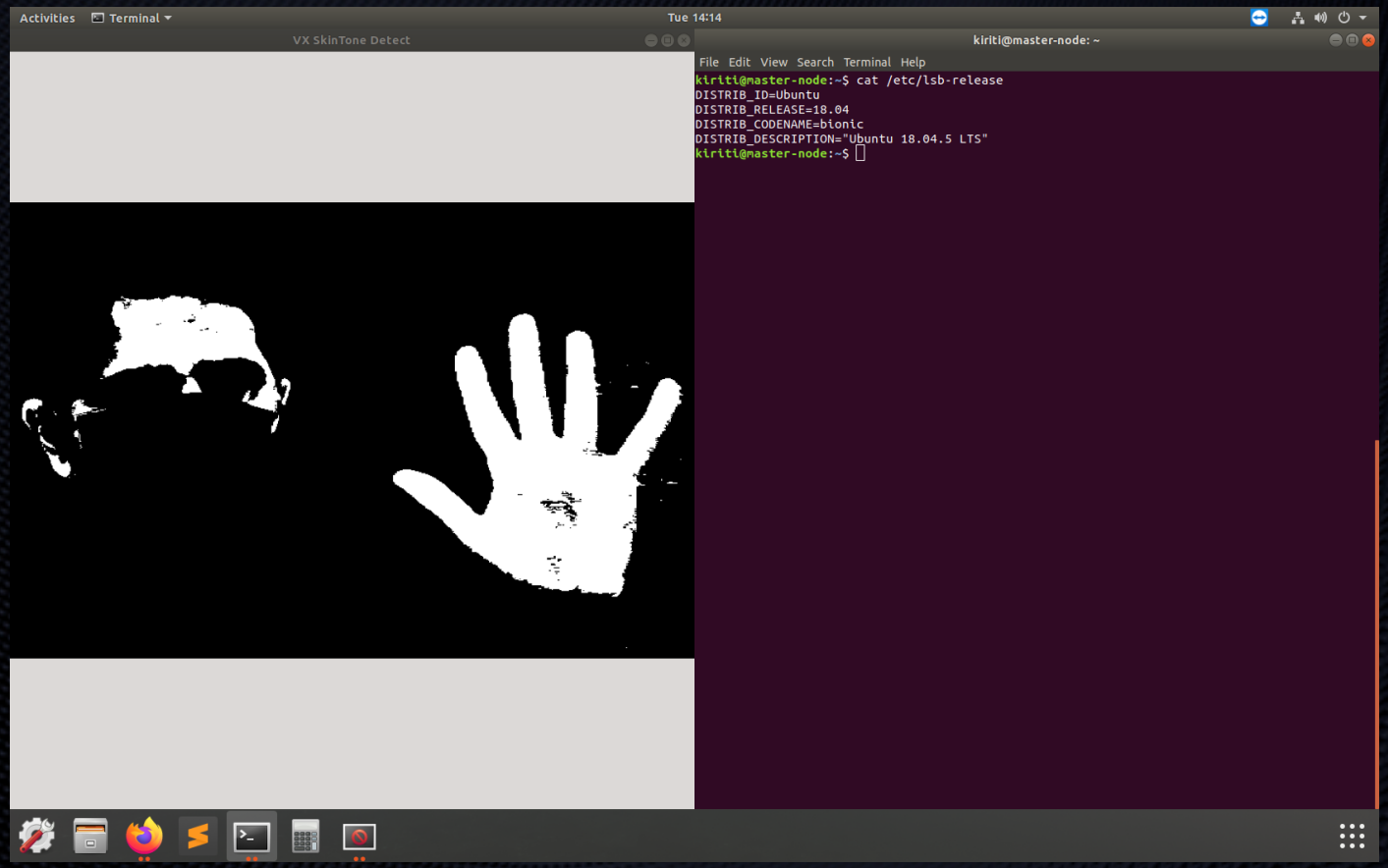
SkinTone Detector Sample – On X86 Processor Windows



* using AMDs open-sourced MIVisionX for OpenVX Libraries

An OpenVX Cross-Platform Application: A Case Study

SkinTone Detector Sample – On X86 Processor Linux



* using Khronos OpenVX open-sourced Sample Implementation for OpenVX Libraries

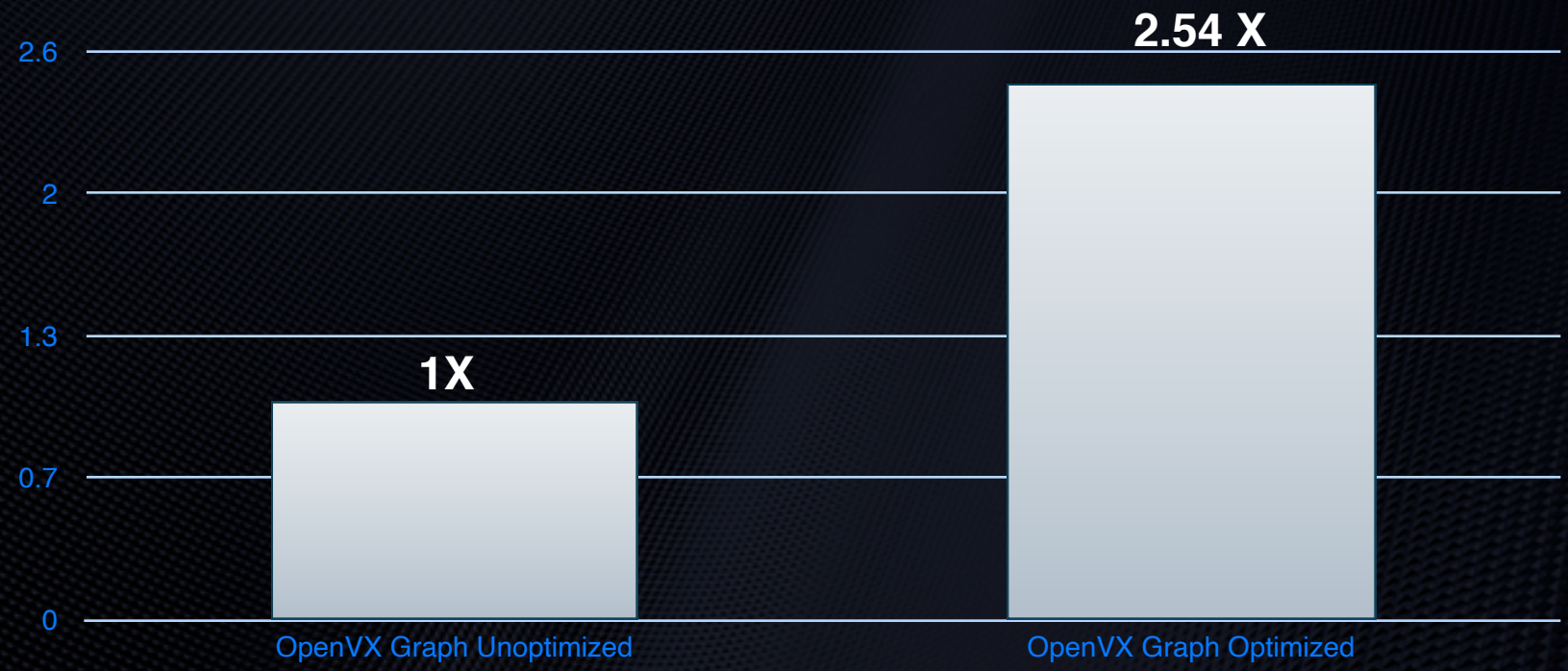
An OpenVX Cross-Platform Application: A Case Study

SkinTone Detector Sample – On MacOS

* using AMDs open-sourced MIVisionX for OpenVX Libraries

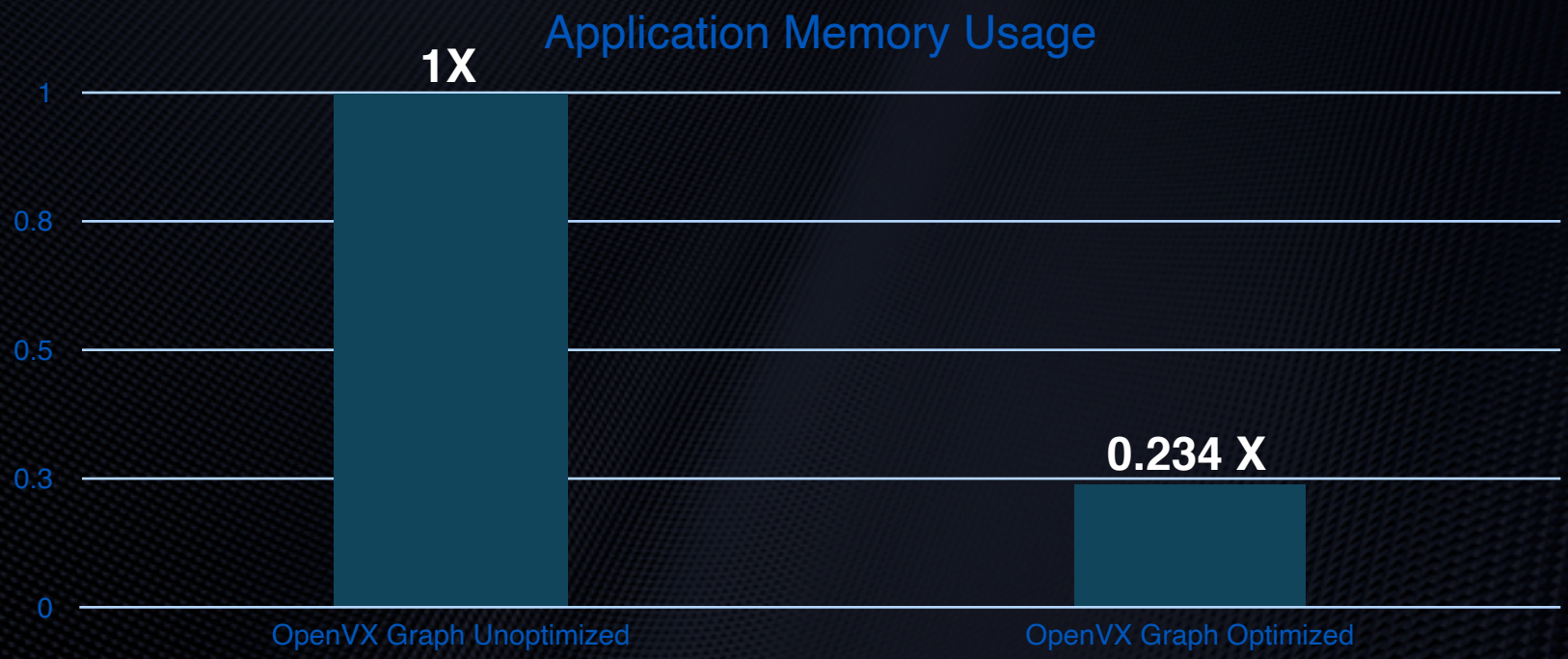
An OpenVX Cross-Platform Application: A Case Study

SkinTone Detector Sample – Performance



An OpenVX Cross-Platform Application: A Case Study

SkinTone Detector Sample – Memory Footprint



Conclusion

- OpenVX is unique in being the only vision API shipped as an **optimized driver**
- OpenVX delivers performance comparable to **hand-optimized, non-portable code**
- **Acceleration** on a **wide range** of **vision hardware architectures**
- OpenVX provides a high-level **Graph-based abstraction**
  - Enables Graph-level optimizations
  - Can be implemented on almost any hardware or processor
- **Portable, Efficient Vision Processing!**

Acknowledgement

Thanks To

- Mike Schmit - Director of Software Engineering, AMD
- AMDs MIVisionX Team
- OpenVX Working Group
- Neil Trevett – President, The Khronos Group
- Khronos Team

Resource Slide

Sample Implementation:

https://github.com/KhronosGroup/OpenVX-sample-impl

Sample Applications:

https://github.com/KhronosGroup/openvx-samples

Tutorial Material:

https://github.com/rgiduthuri/openvx_tutorial

OpenVX Programming Guide:

- https://www.elsevier.com/books/openvx-programming-guide/brill/978-0-12-816425-9

Conformant Implementations

https://www.khronos.org/conformance/adopters/conformant-products/openvx

Khronos OpenVX API Registry

https://www.khronos.org/registry/OpenVX/

OpenVX for Raspberry Pi

https://www.raspberrypi.org/blog/openvx-api-for-raspberry-pi/

AMD ROCm MIVisionX - OpenVX

https://gpuopen-professionalcompute-libraries.github.io/MIVisionX/

Disclaimers and attributions

The information presented in this document is for informational purposes only and may contain technical inaccuracies, omissions, and typographical errors. The information contained herein is subject to change and may be rendered inaccurate for many reasons, including but not limited to product and roadmap changes, component and motherboard version changes, new model and/or product releases, product differences between differing manufacturers, software changes, BIOS flashes, firmware upgrades, or the like. Any computer system has risks of security vulnerabilities that cannot be completely prevented or mitigated. AMD assumes no obligation to update or otherwise correct or revise this information. However, AMD reserves the right to revise this information and to make changes from time to time to the content hereof without obligation of AMD to notify any person of such revisions or changes.

THIS INFORMATION IS PROVIDED 'AS IS." AMD MAKES NO REPRESENTATIONS OR WARRANTIES WITH RESPECT TO THE CONTENTS HEREOF AND ASSUMES NO RESPONSIBILITY FOR ANY INACCURACIES, ERRORS, OR OMISSIONS THAT MAY APPEAR IN THIS INFORMATION. AMD SPECIFICALLY DISCLAIMS ANY IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY, OR FITNESS FOR ANY PARTICULAR PURPOSE. IN NO EVENT WILL AMD BE LIABLE TO ANY PERSON FOR ANY RELIANCE, DIRECT, INDIRECT, SPECIAL, OR OTHER CONSEQUENTIAL DAMAGES ARISING FROM THE USE OF ANY INFORMATION CONTAINED HEREIN, EVEN IF AMD IS EXPRESSLY ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

AMD, the AMD Arrow logo, EPYC, Radeon, ROCm and combinations thereof are trademarks of Advanced Micro Devices, Inc. Other product names used in this publication are for identification purposes only and may be trademarks of their respective companies.


AMD