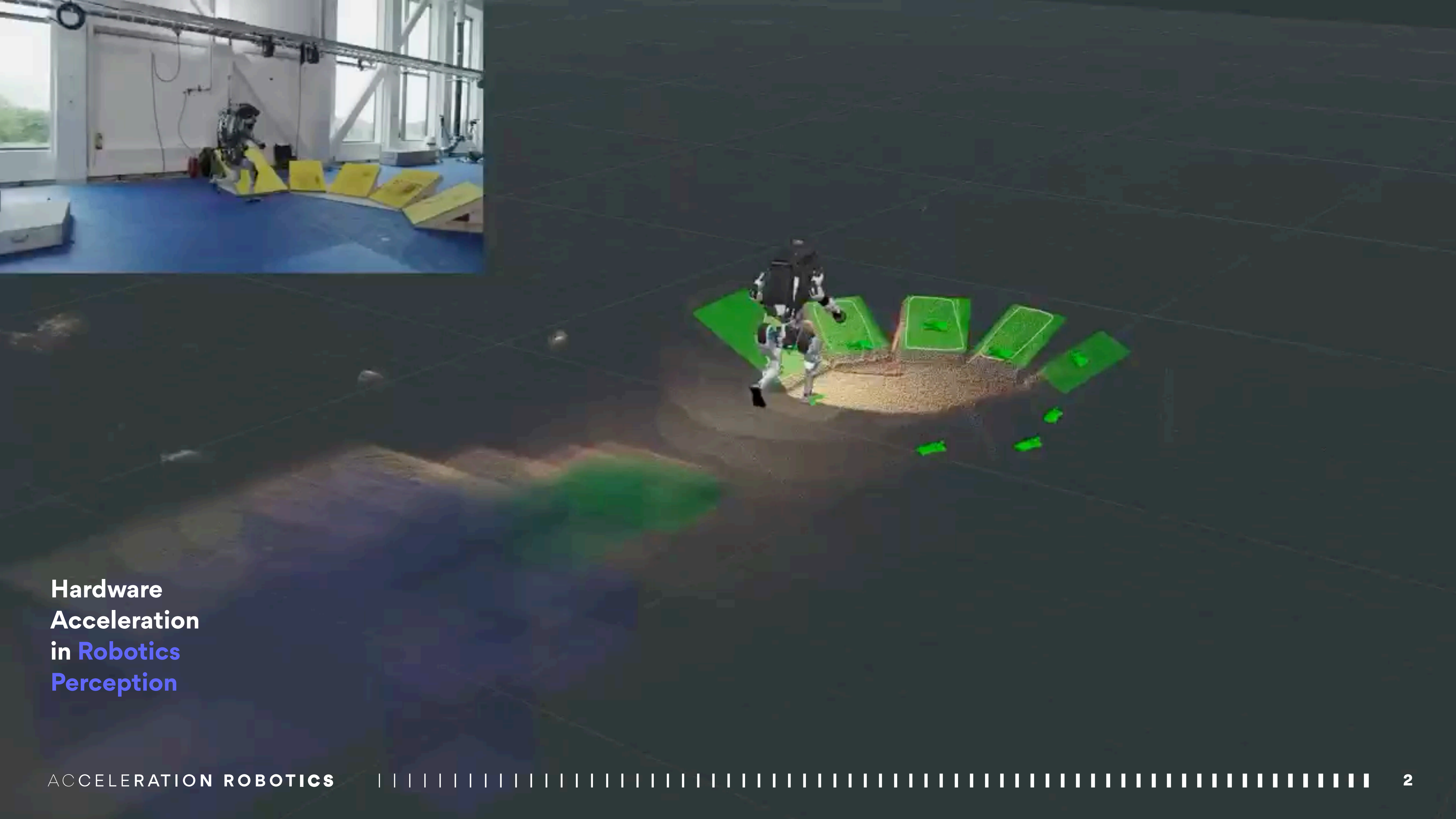


An open architecture for **Hardware Acceleration** in ROS 2

# ROBOTC ROBOTC ORE Framework

Easily leverage hardware acceleration in a ROS-centric manner and build custom compute architectures for robots, or robot cores.



Hardware  
Acceleration  
in **Robotics**  
**Perception**

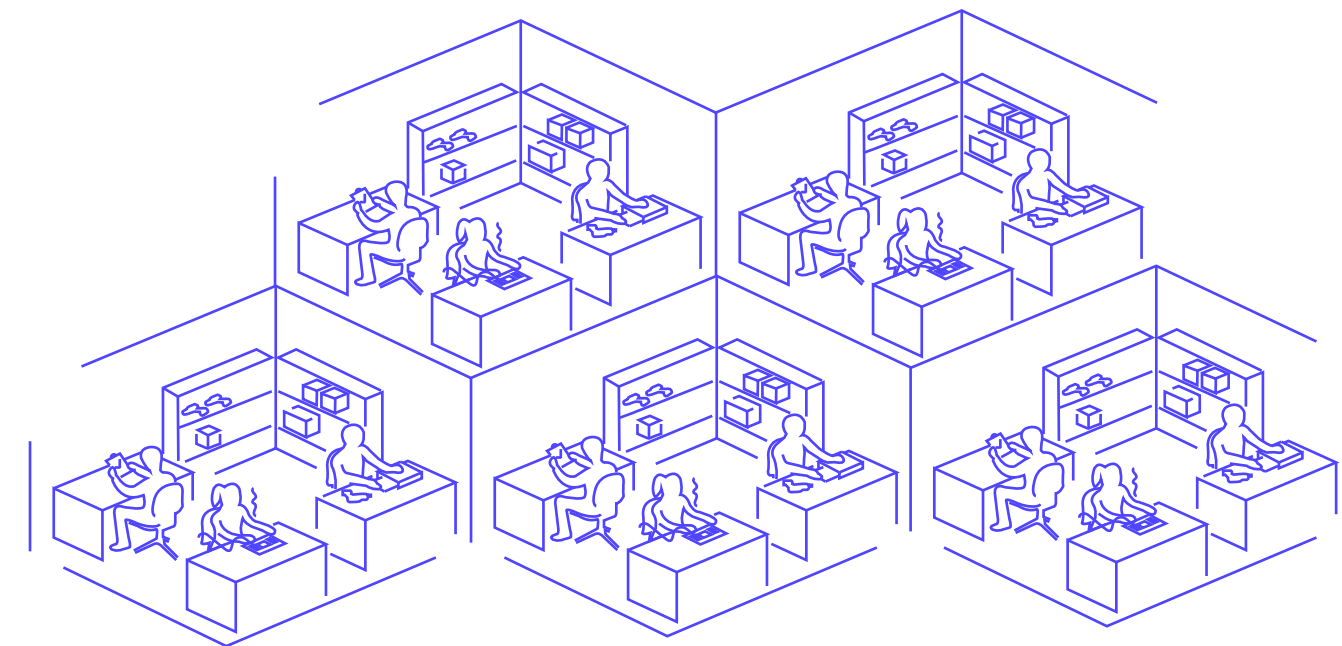
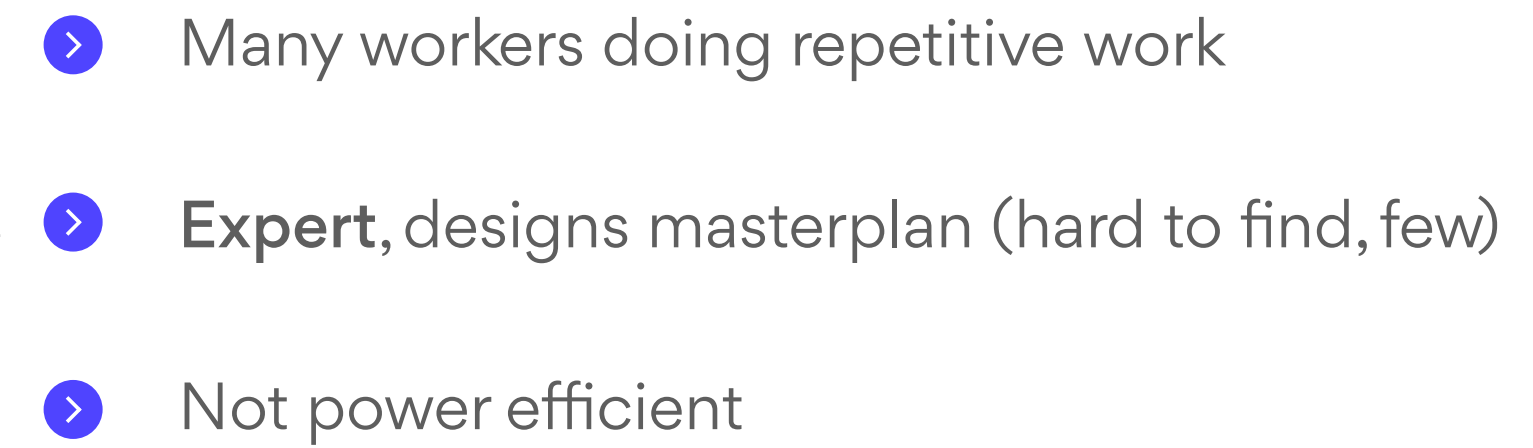
# CPU



- 

- ACCELERATION ROBOTICS

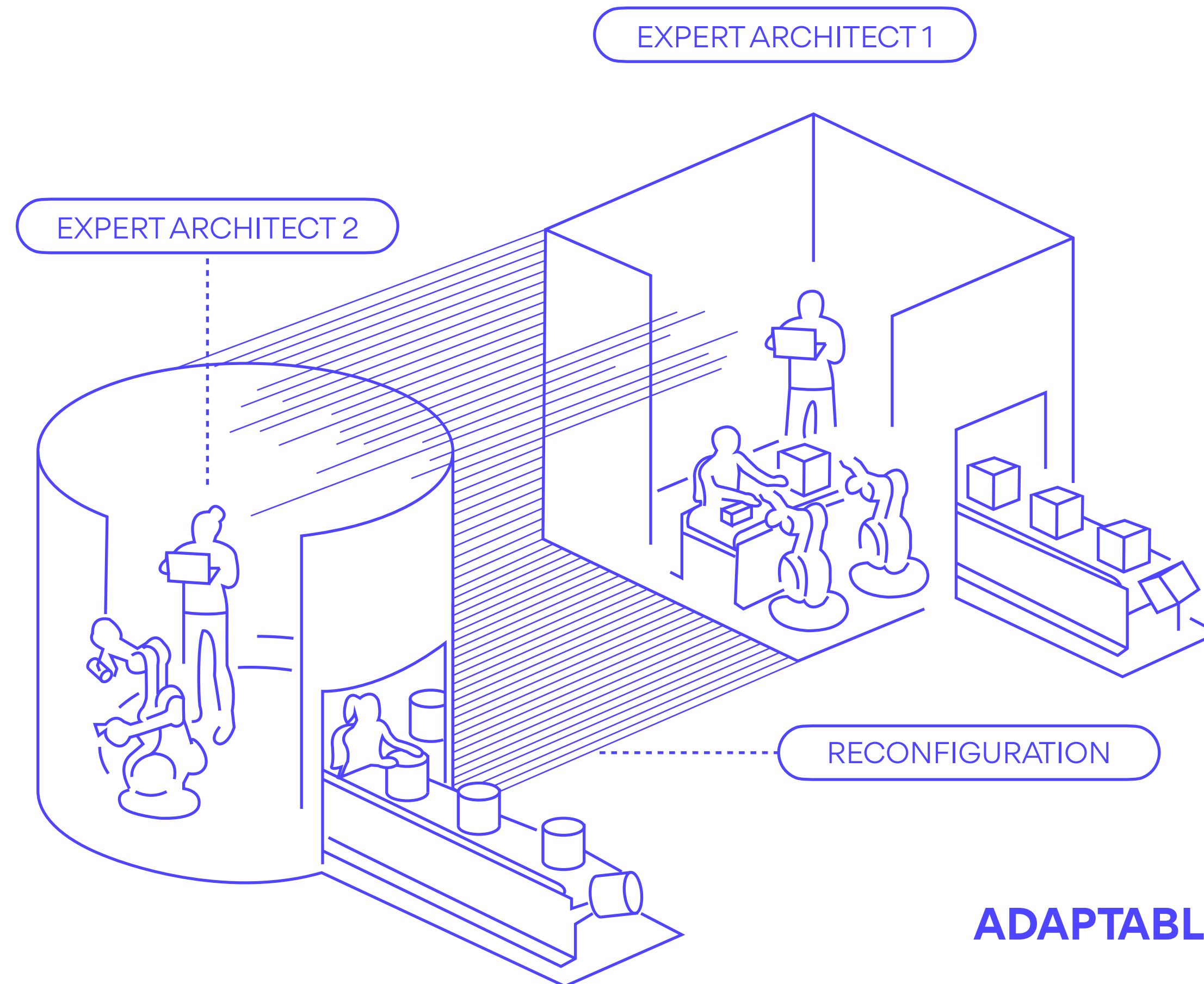
# GPU



- GPUs have many industrial workshops

# Acceleration Robotics: Understanding robotics computations in FPGAs

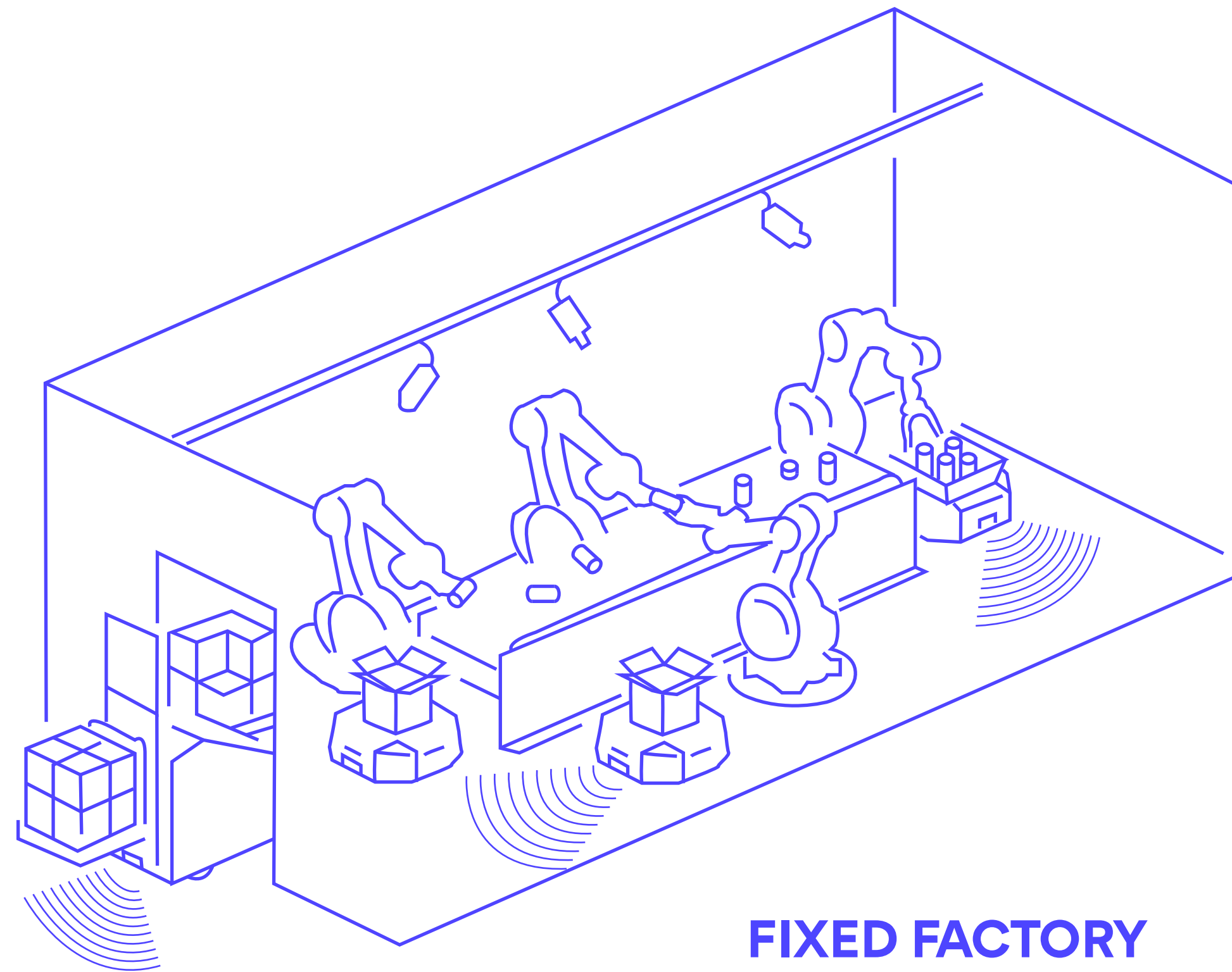
# FPGA



- Flexible and adaptable factory (software defined)
- Expert-driven architectures
- Low power, high performance

# Acceleration Robotics: Understanding robotics computations in ASICs

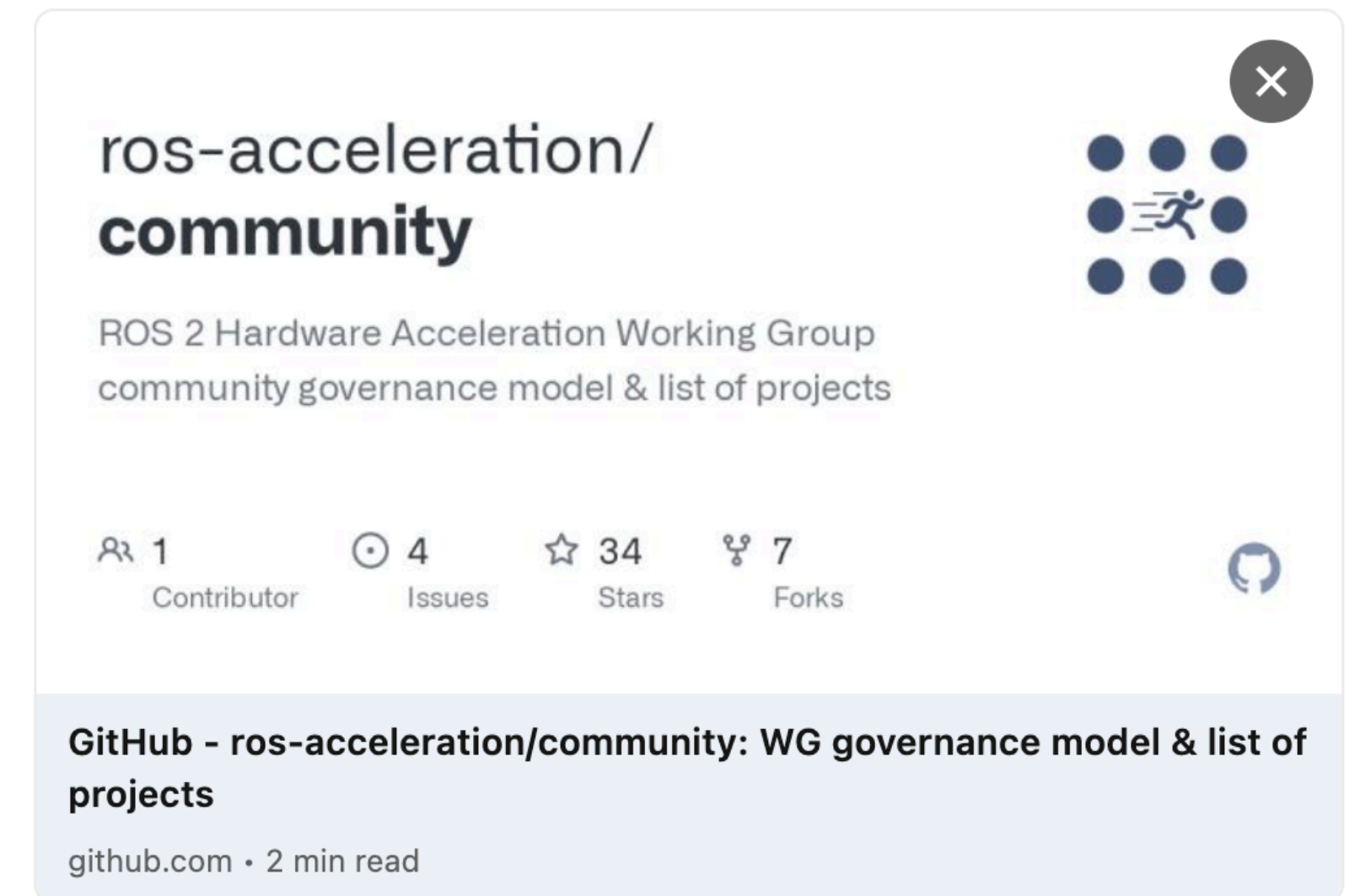
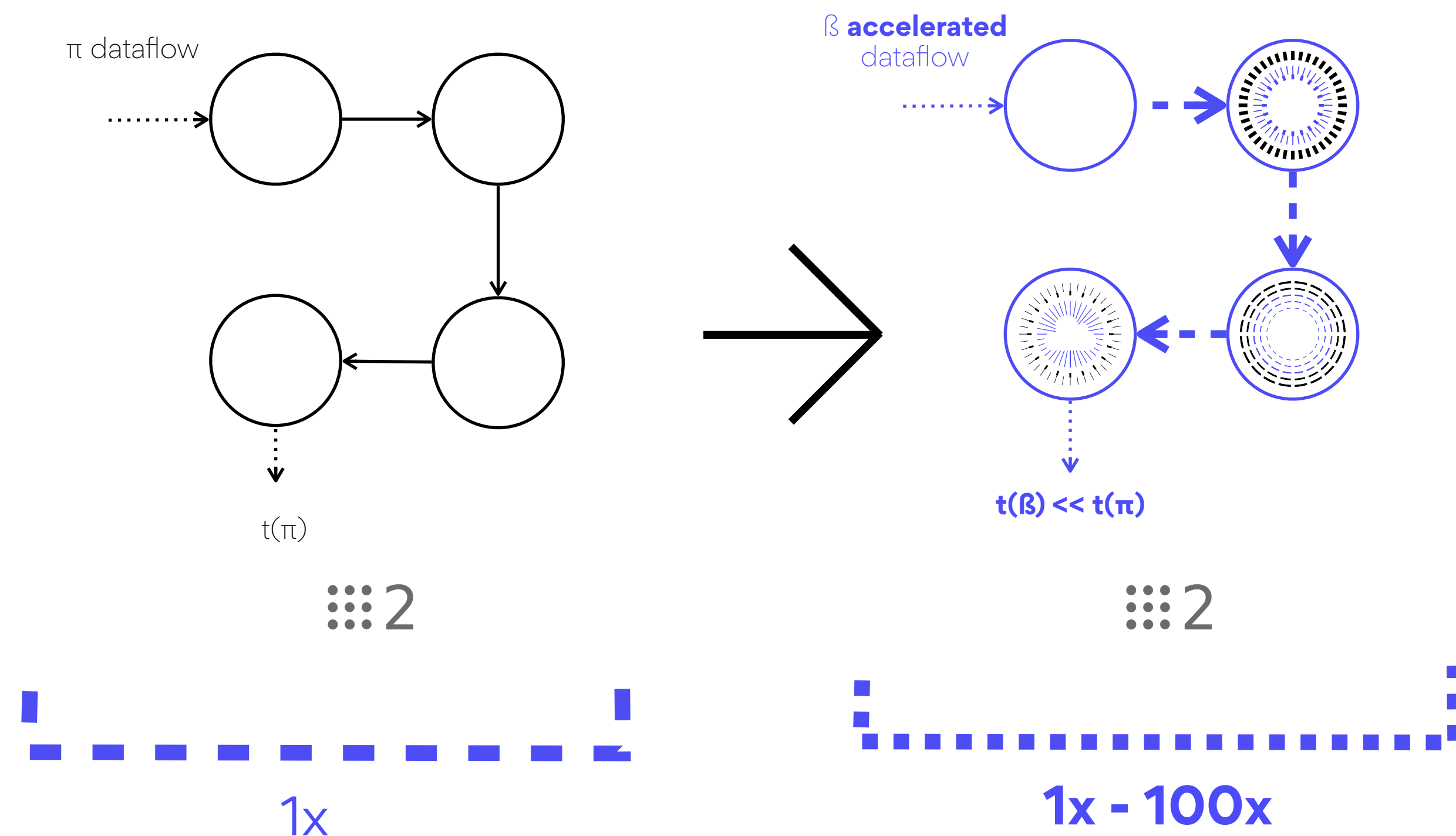
# ASIC



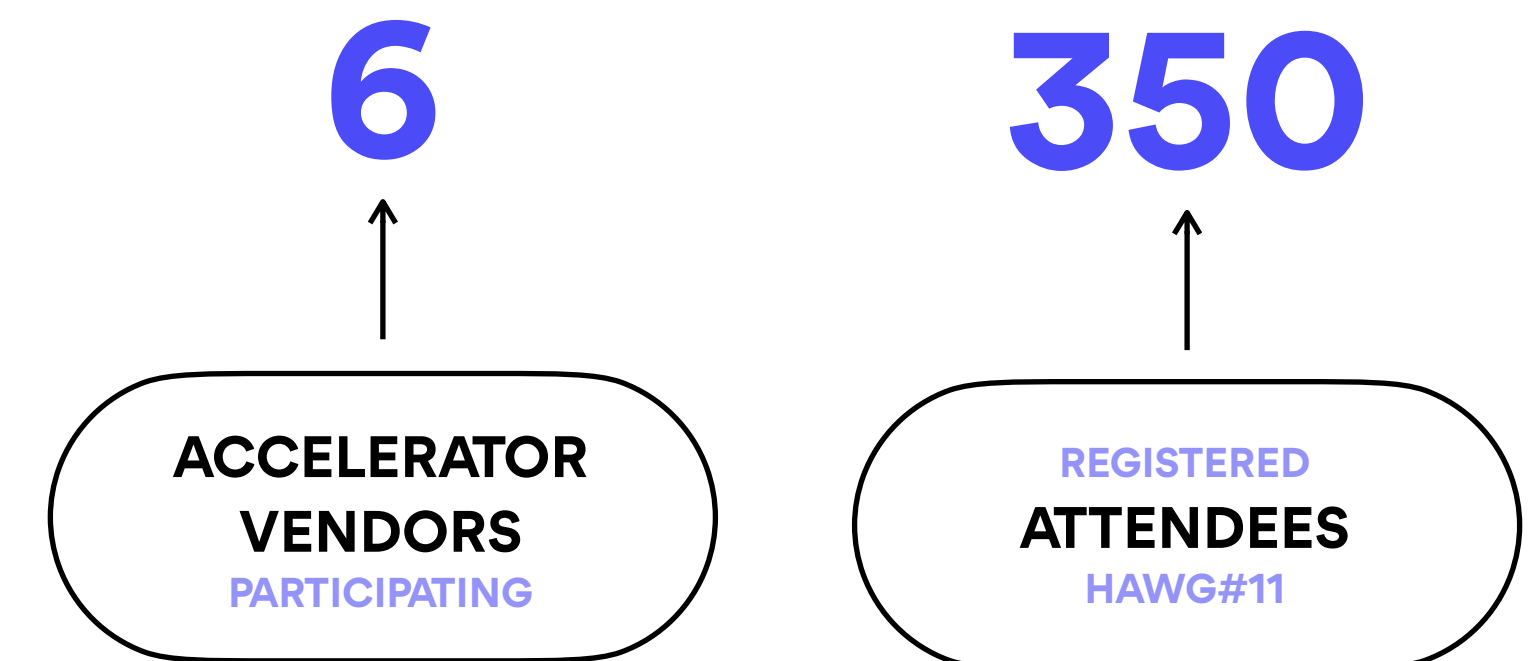
- > Fixed, fully automated factory
- > Specialized in one process
- > Very expensive to design and build
- > Best throughput at lowest power consumption

# ROS 2 Hardware Acceleration Working Group (HAWG)

The **ROS 2 Hardware Acceleration Working Group** drives the creation, maintenance and testing of acceleration kernels on top of open standards for optimized ROS 2 and Gazebo interactions over different compute substrates (including FPGAs, GPUs and other accelerators).



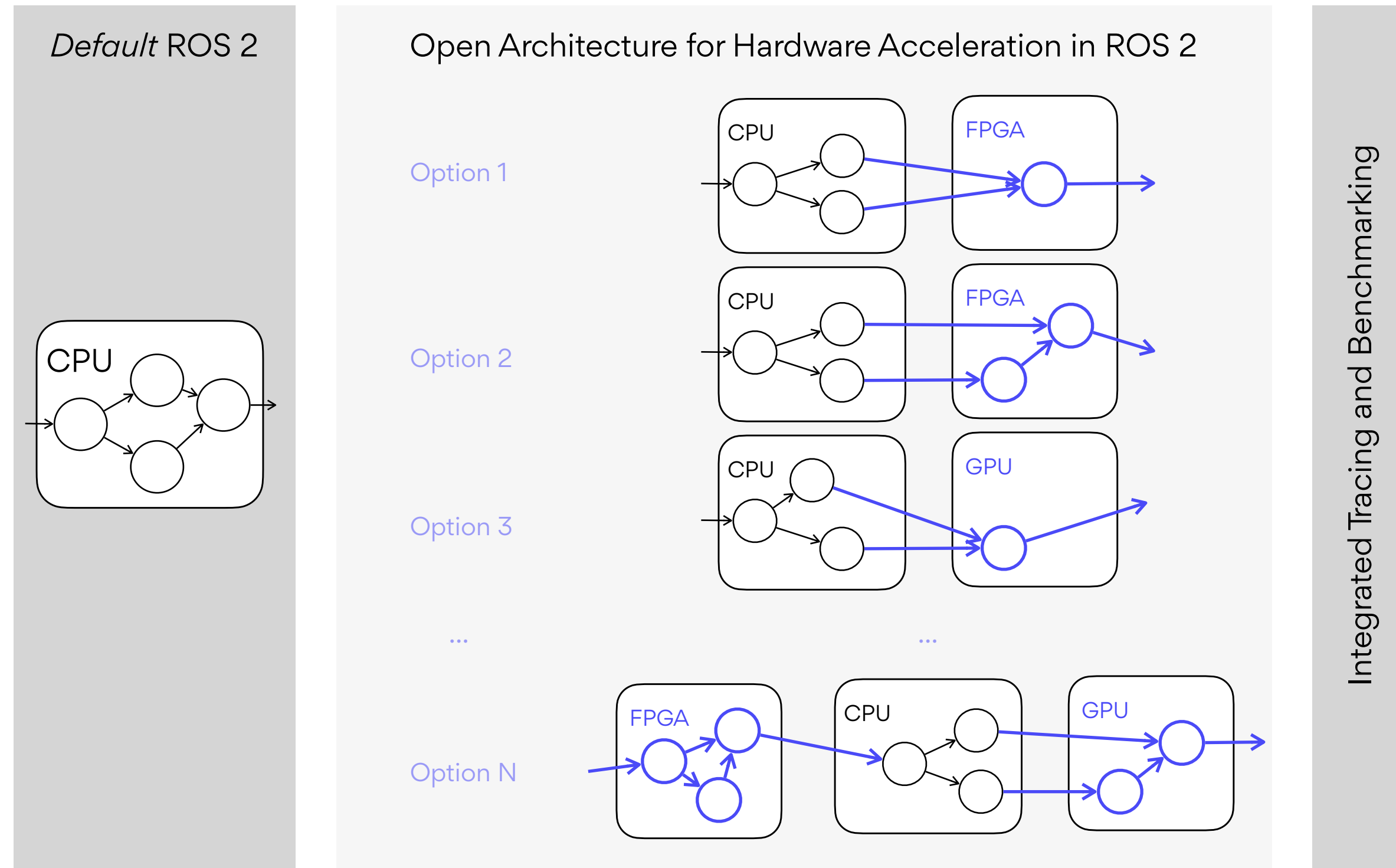
<https://github.com/ros-acceleration/community>



# ROBOTCORE Framework: An open architecture for Hardware Acceleration in ROS 2

## Hardware Acceleration Framework for ROS

It helps build custom compute architectures for robots through acceleration kernels, or **robot cores**, that make robots faster, more deterministic and power-efficient. *Simply put, it provides a development, build and deployment experience for creating robot hardware and hardware acceleration kernels similar to the standard, non-accelerated ROS development flow.*

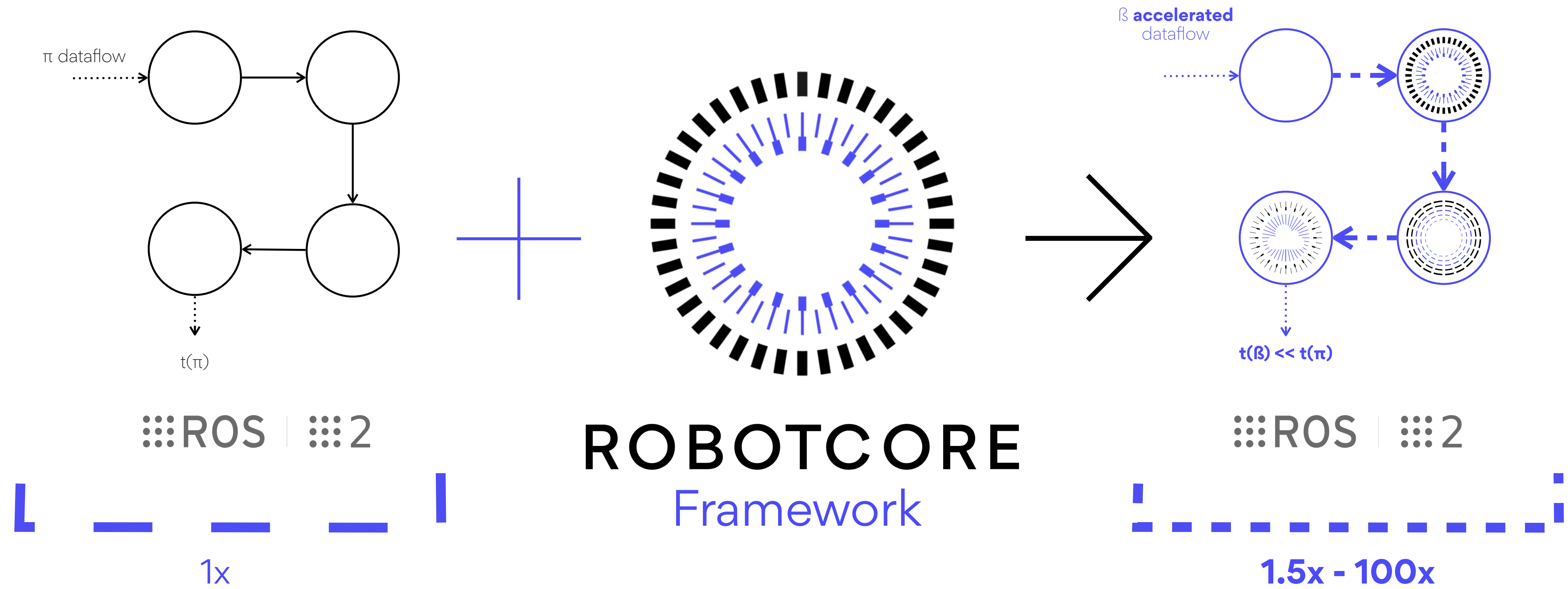


<https://arxiv.org/pdf/2205.03929.pdf>

# ROBOTCORE Framework

**Hardware Acceleration Framework for ROS.** It helps build custom compute architectures for robots, or **robot cores**, that make robots faster, more deterministic and power-efficient. Simply put, it provides a development, build and deployment experience for creating robot hardware and hardware accelerators similar to the standard, non-accelerated ROS development flow.

**Public** **community-driven and open source**  
**implementation** available at <https://github.com/ros-acceleration>

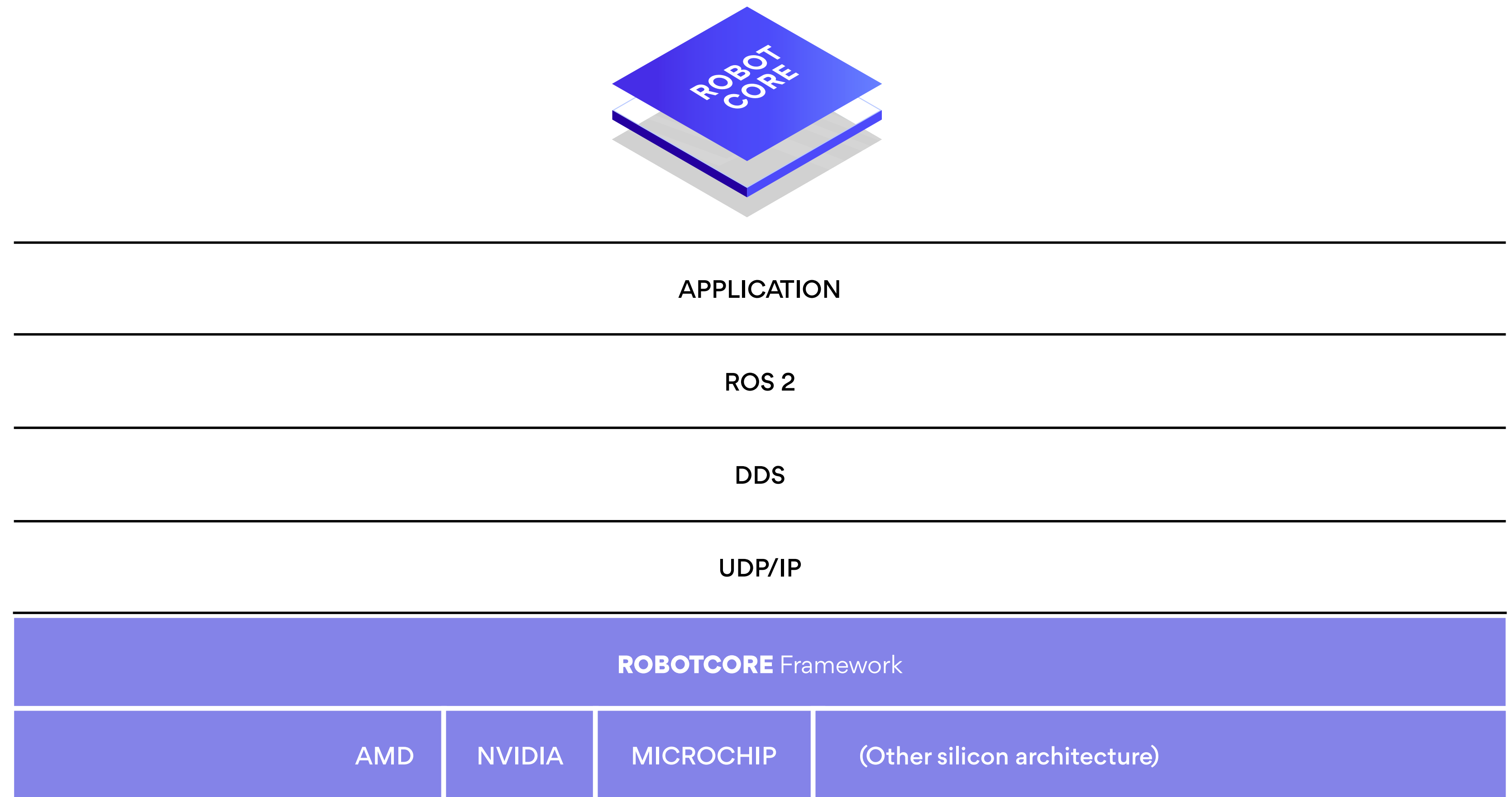




# From ROBOTCORE Framework to an open standard ROS 2 Hardware Acceleration Architecture and Conventions ([REP 2008](#) ➡)

We are bringing the lessons learned while developing ROBOTCORE Framework into a REP that describes the architectural pillars and conventions required to introduce hardware acceleration in ROS 2 in a vendor-neutral, scalable and technology-agnostic manner.

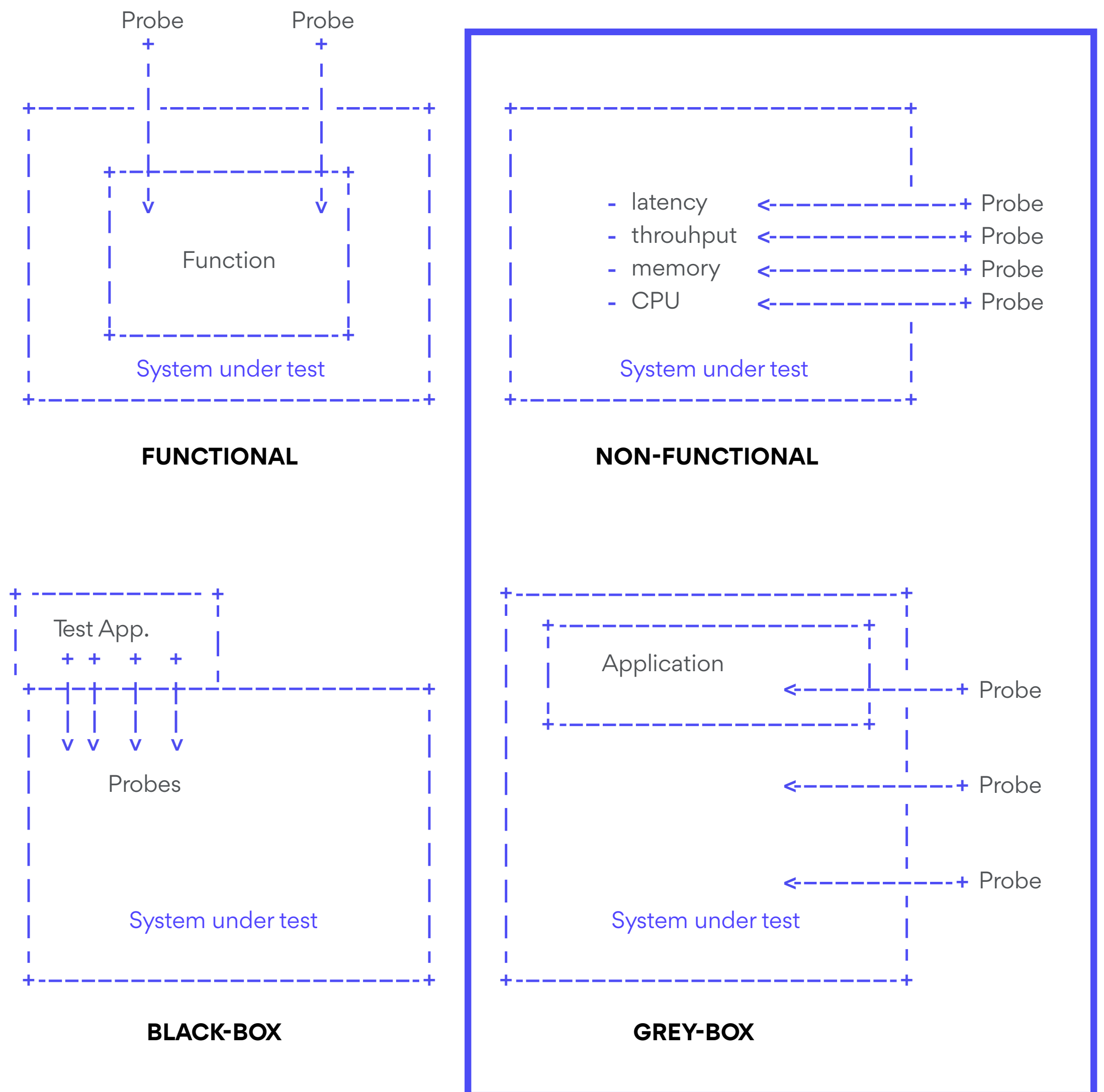
All while maintaining the common ROS development flow.



## From ROBOTCORE Framework to an open standard Benchmarking performance in ROS 2 ([REP 2014](#) ➡)

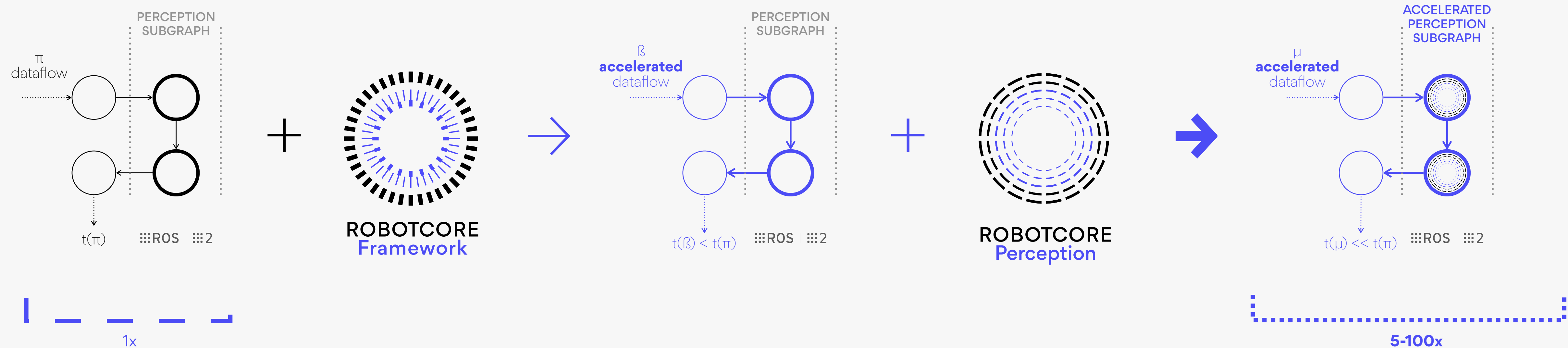
Benchmarking is the act of running a computer program with a known workload to assess the program's relative performance.

We adopt a grey-box and non-functional benchmarking approach for hardware acceleration with a **low-overhead tracing and benchmarking framework**, and select the Linux Tracing Toolkit next generation (LTTng) to implement it.

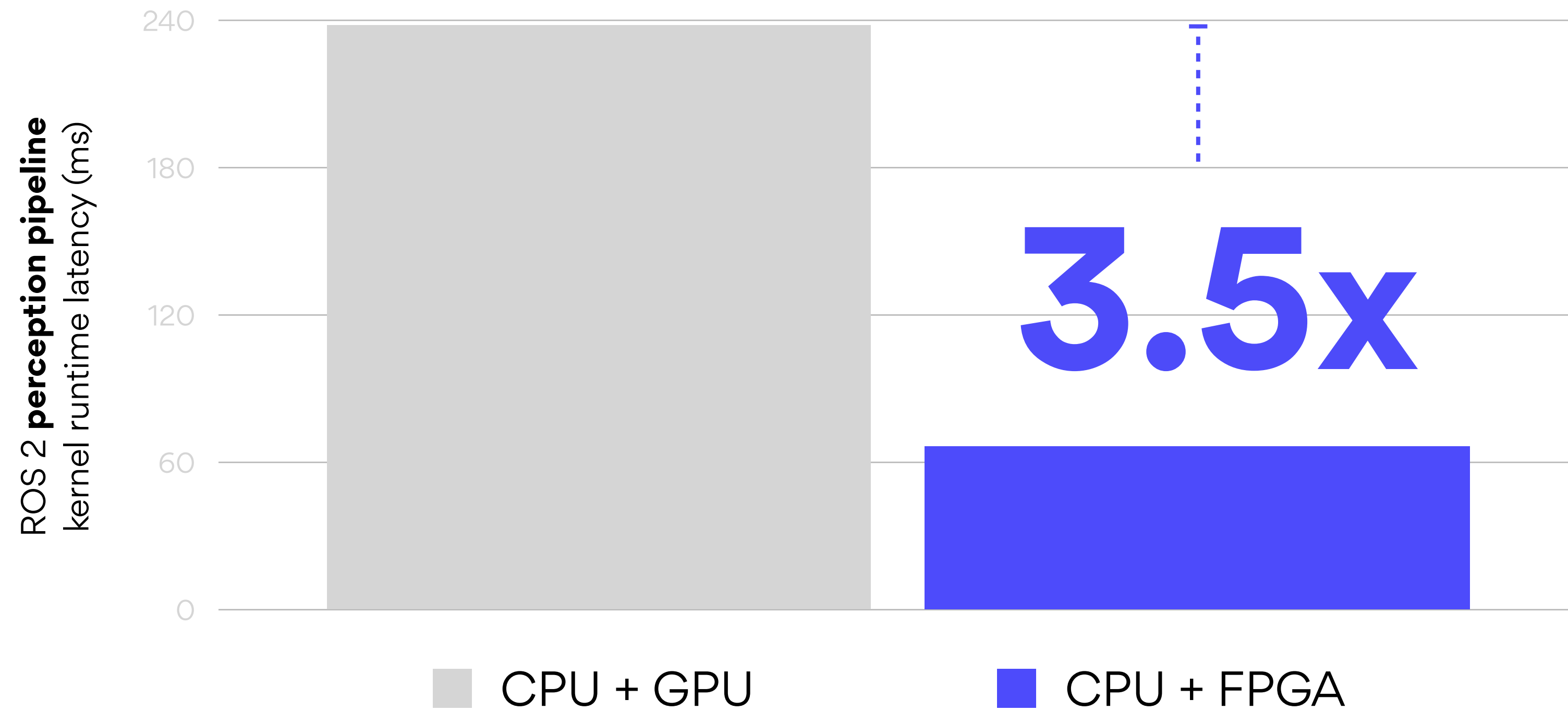


## Case Study: ROS 2 Perception graph

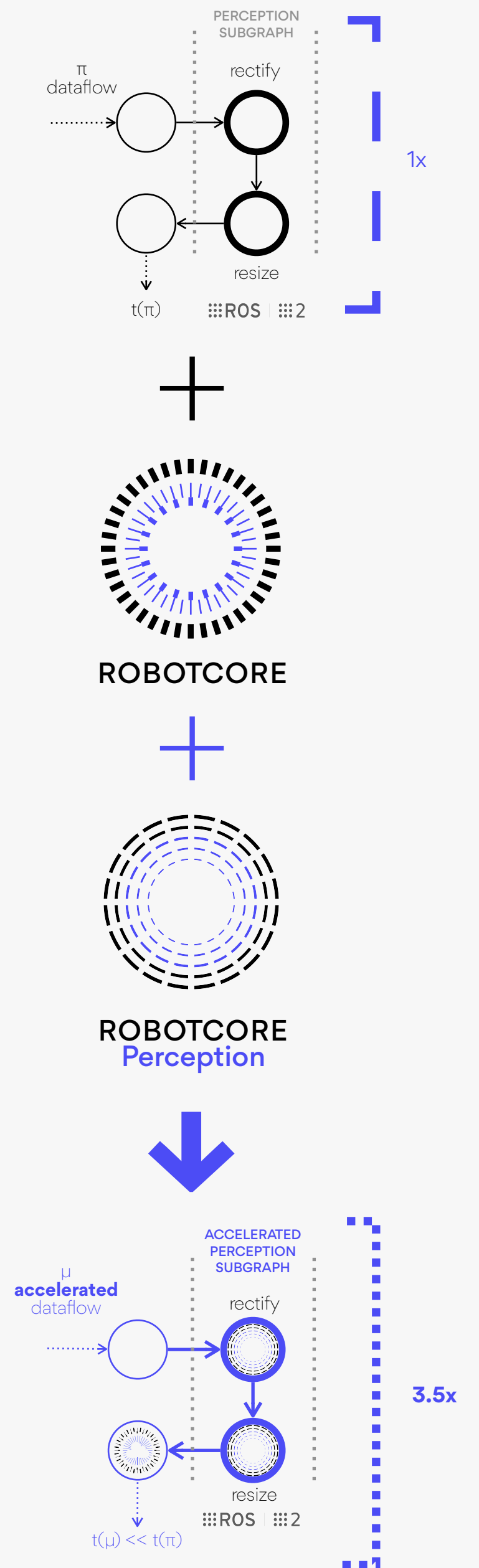
ROBOTCORE Perception is an **optimized robotic perception stack built with ROBOTCORE Framework that leverages hardware acceleration** to provide a speedup in your perception computations. API-compatible with the ROS 2 perception stack, ROBOTCORE Perception delivers high performance, real-time and reliability to your robots' perception.



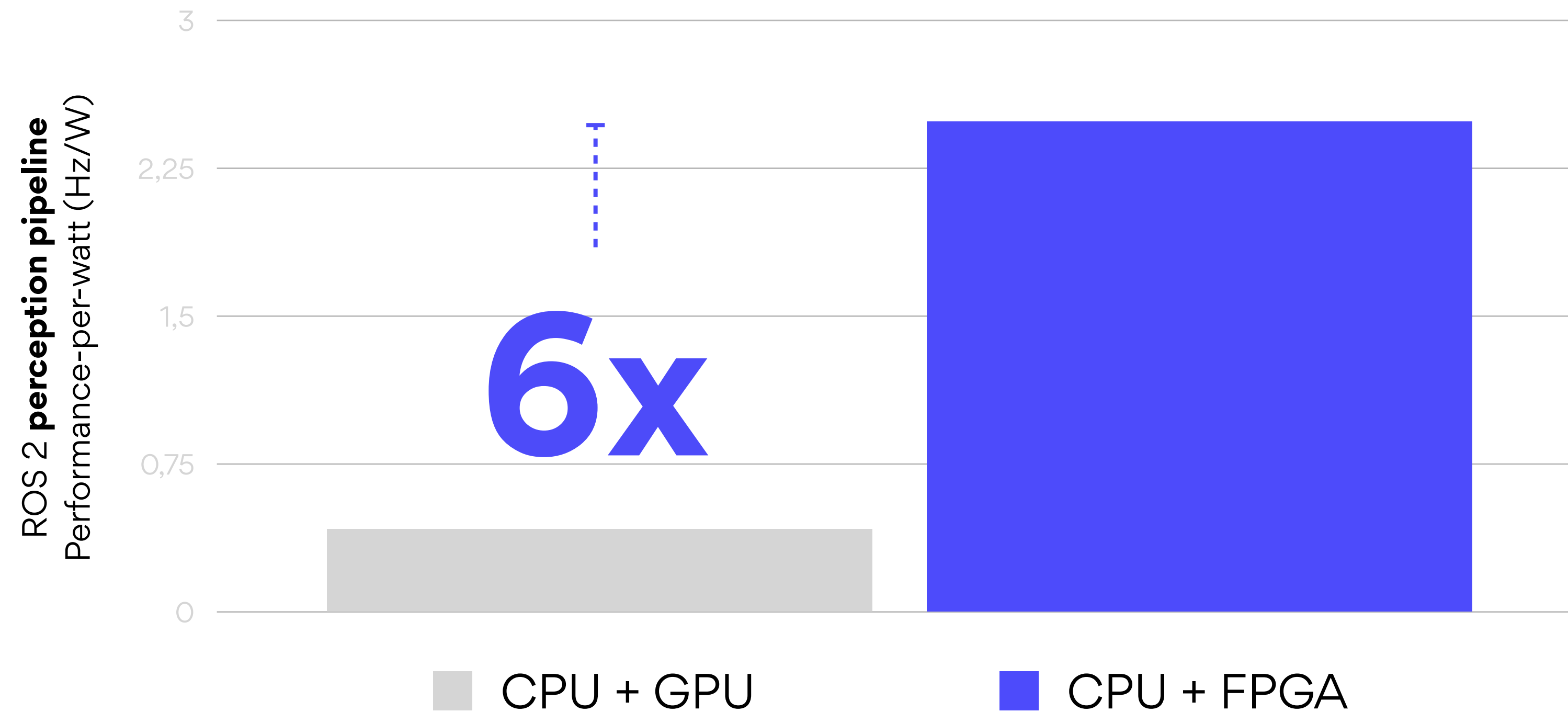
## ROS 2 Perception graph: 2 nodes perception pipeline



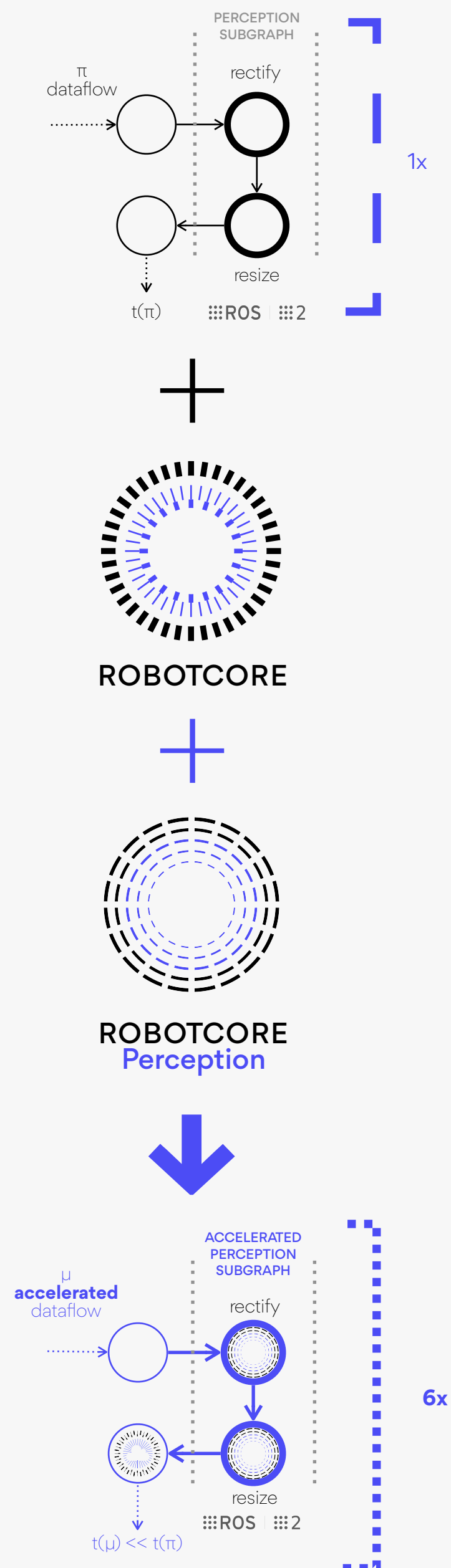
**ROS 2 perception pipeline (2 nodes) - kernel runtime latency (ms).** Measured ROBOTCORE Perception pipeline runtime latency running on an AMD KV260 versus NVIDIA Isaac ROS running in a Jetson Nano 2GB. Measurements present the kernel runtime in milliseconds (ms). Source code available at [perception\\_2nodes](#).



## ROS 2 Perception graph: 2 nodes perception pipeline

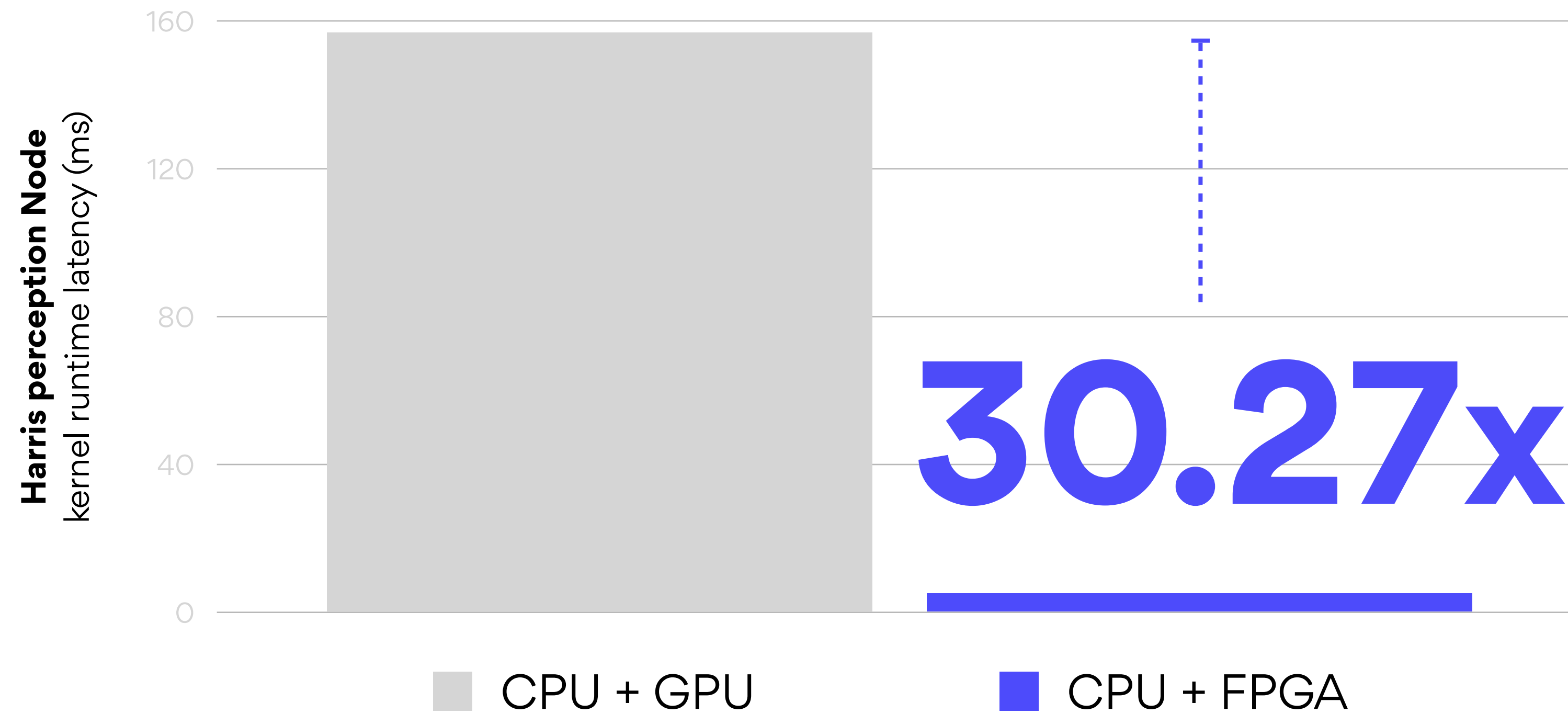


**ROS 2 perception pipeline (2 nodes)** - Performance-per-watt (Hz/W). Measured ROBOTCORE running on an AMD KV260 versus NVIDIA Isaac ROS running in a Jetson Nano 2GB. Measurements present a 2-Node (rectify and resize operations) pre-processing perception graph performance-per-watt (Hz/W). Source code available at [perception\\_2nodes](#).

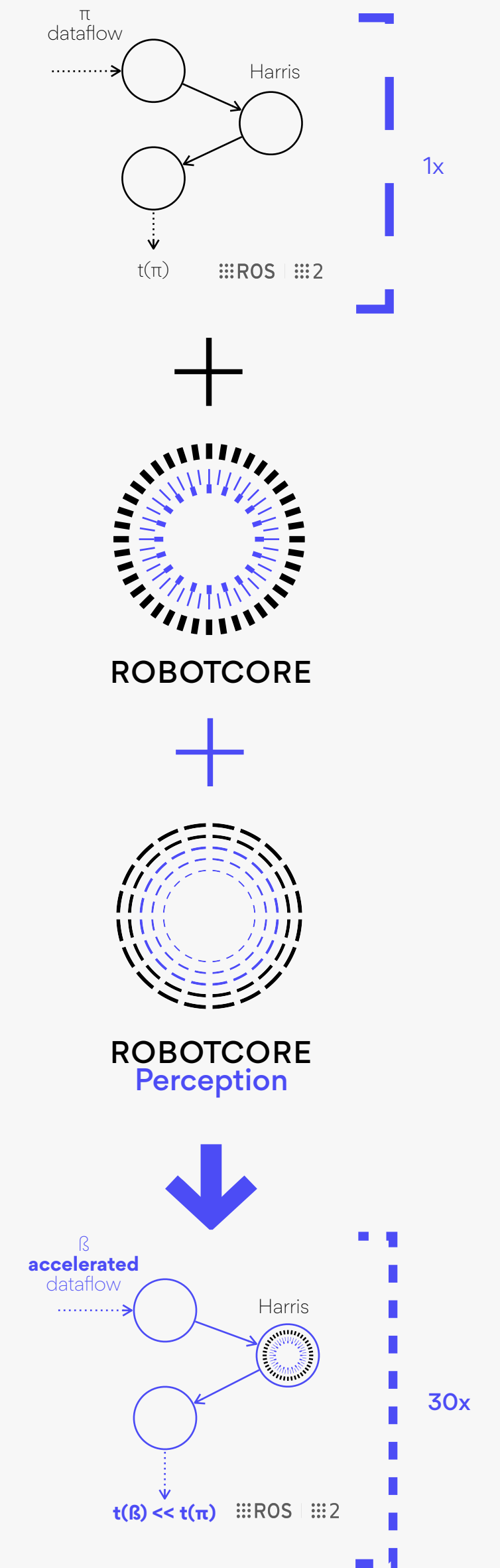




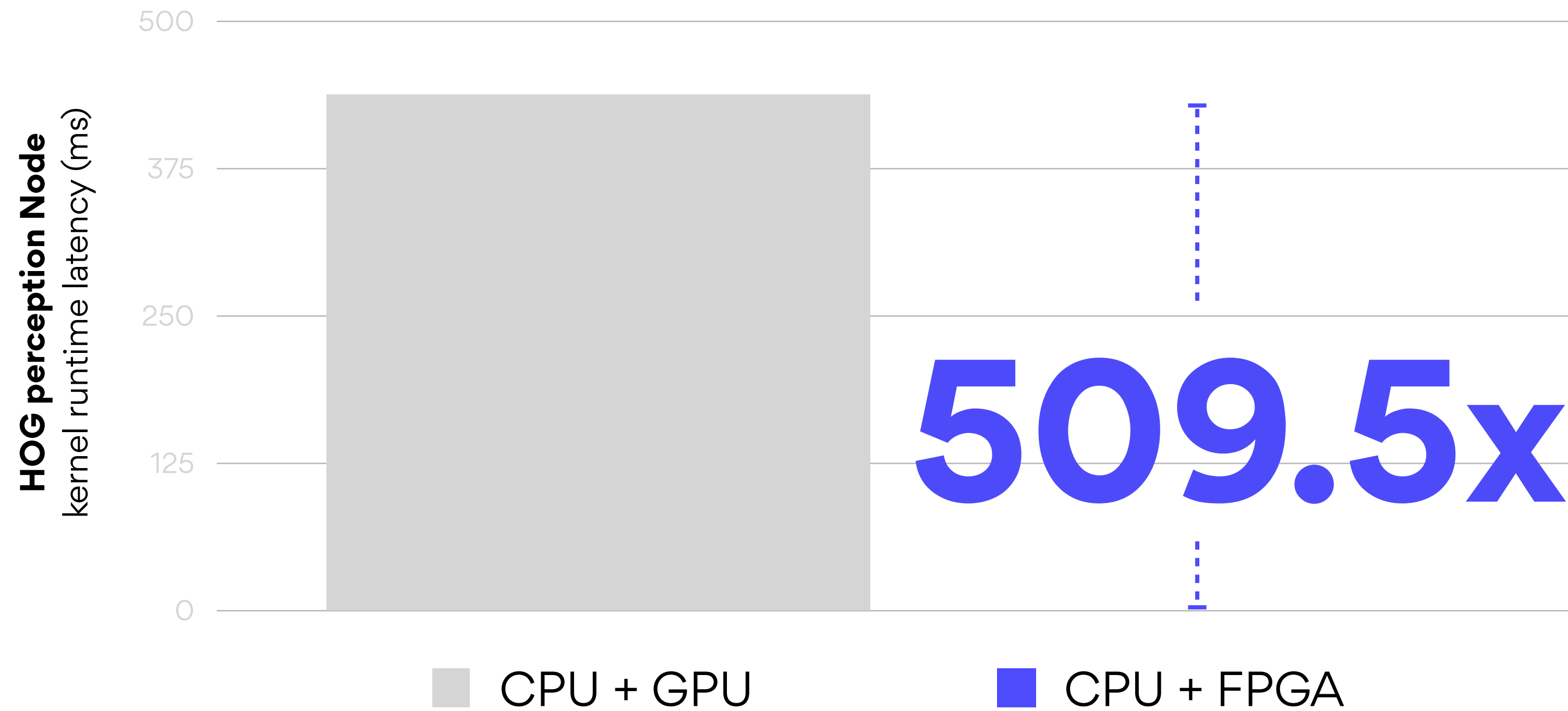
## ROS 2 Perception Nodes: Harris perception Node



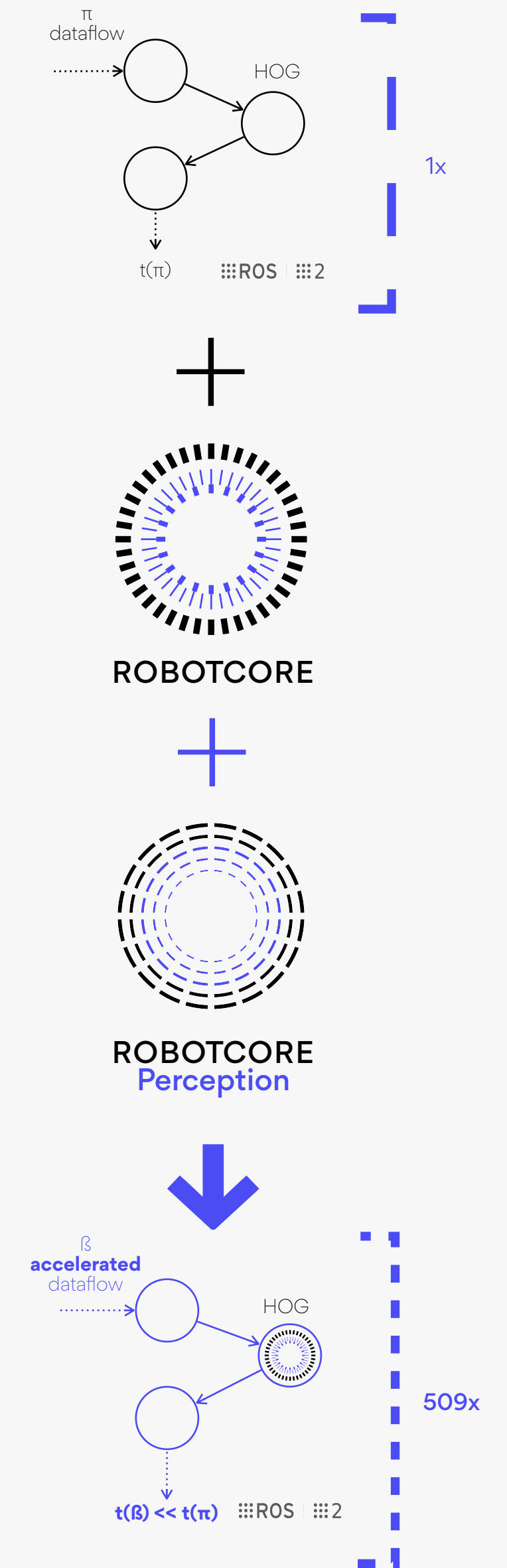
**Harris** - kernel runtime latency (ms). Measured ROBOTCORE Perception running on an AMD KV260, NVIDIA Isaac ROS running in a Jetson Nano 2GB. Measurements present the kernel runtime in milliseconds (ms) and discard ROS 2 message-passing infrastructure overhead and host-device (GPU or FPGA) data transfer overhead



## ROS 2 Perception Nodes: HOG perception Node



**Histogram of Oriented Gradients (HOG) - kernel runtime latency (ms).** Measured ROBOTCORE Perception running on an AMD KV260, NVIDIA Isaac ROS running in a Jetson Nano 2GB. Measurements present the kernel runtime in milliseconds (ms) and discard ROS 2 message-passing infrastructure overhead and host-device (GPU or FPGA) data transfer overhead





# 2022

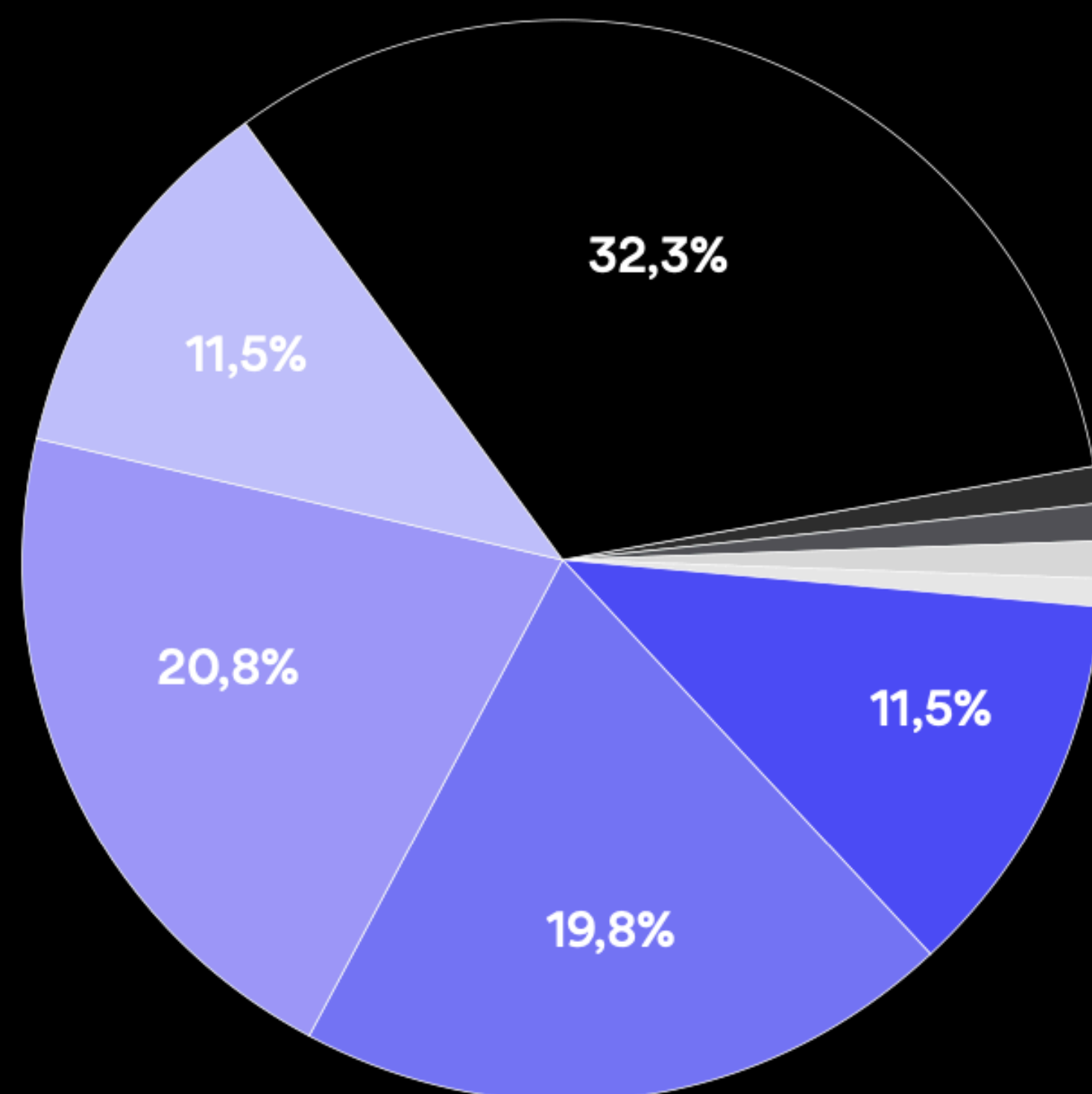
## Hardware Acceleration Report in Robotics

Captures the state-of-the art of hardware acceleration in robotics by following a quantitative approach and presents robotic architects with a resource to consider while designing their robot computational architectures.

- > First phase, a **community survey** conducted in both the ROS and the overall robotics communities helped grasp the interest behind the use of hardware acceleration in robotics. Input from this community survey was then used to drive the
- > Second phase, a **hardware acceleration benchmarking effort**

# 2022 Hardware Acceleration Report in Robotics

First phase: **We're pushing forward REP-2008 initiative to better integrate hardware acceleration with ROS and Gazebo, what's most important for you?** ([report ↗](#))



96 answers

- |   |  |  |
|---|--|--|
| <input checked="" type="radio"/> Integration with ROS 2 build system (ament)  | <input checked="" type="radio"/> Benchmarking capabilities for hardware acceleration                   | <input type="radio"/> None of the above  |
| <input checked="" type="radio"/> Integration with ROS 2 build tools (colcon)  | <input type="radio"/> Capabilities to easily switch between hardware accelerated and CPU-centric Nodes | <input checked="" type="radio"/> Adding acceleration to important packages that just work without thinking about it  |
| <input checked="" type="radio"/> Acceleration firmware integrated into ROS 2 workspaces (cross-compilers, hypervisors, etc.)? | <input type="radio"/> They are all equally important to make offloading transparent for the user       | <input checked="" type="radio"/> Complete and accurate documentation is the priority to me. Likely, large part of the community is unfamiliar with hardware acceleration |

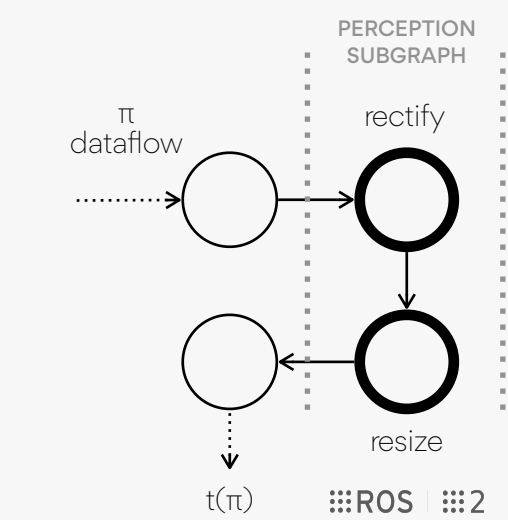
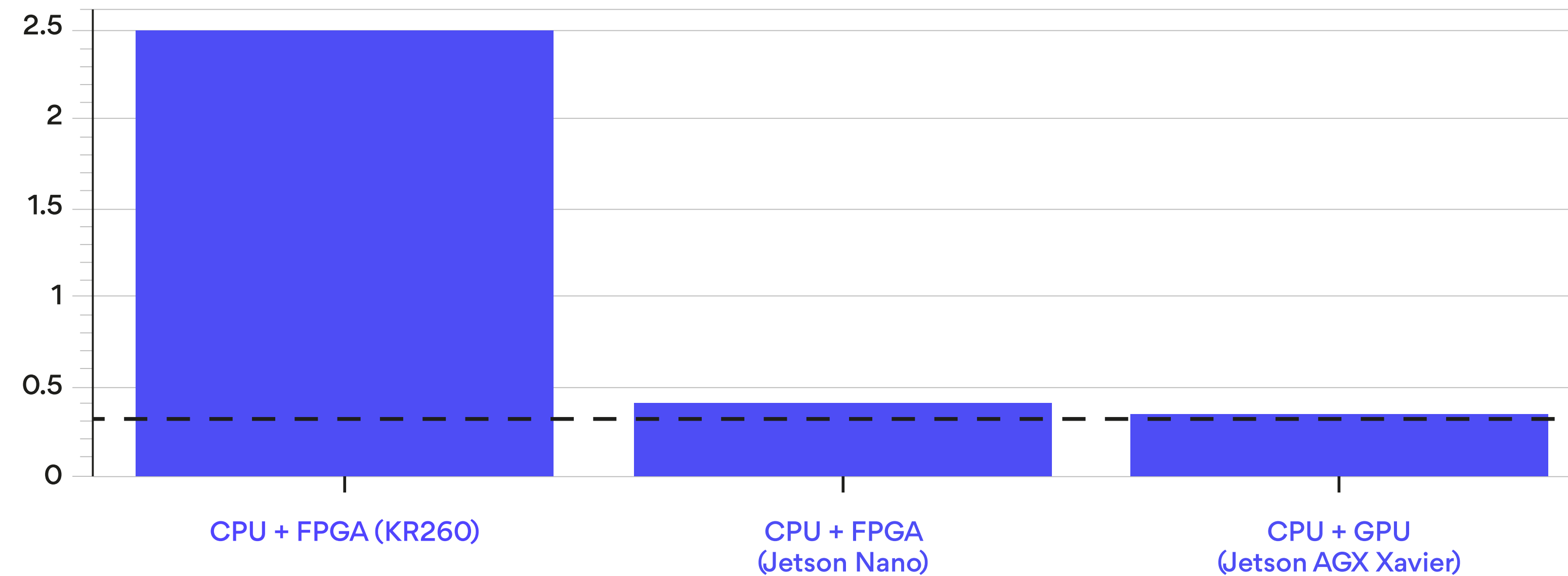




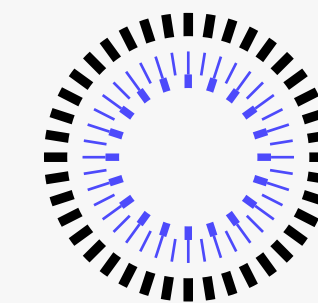
# 2022 Hardware Acceleration Report in Robotics

Second phase: **Benchmarking hardware acceleration** ([report](#)→)

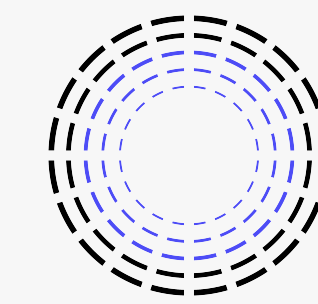
## ROS 2 perception graph performance-per-watt with hardware acceleration (Hz/W)



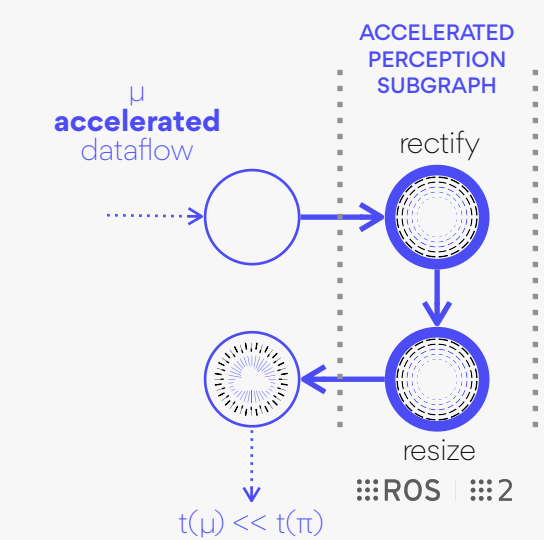
+



# ROBOTCORE

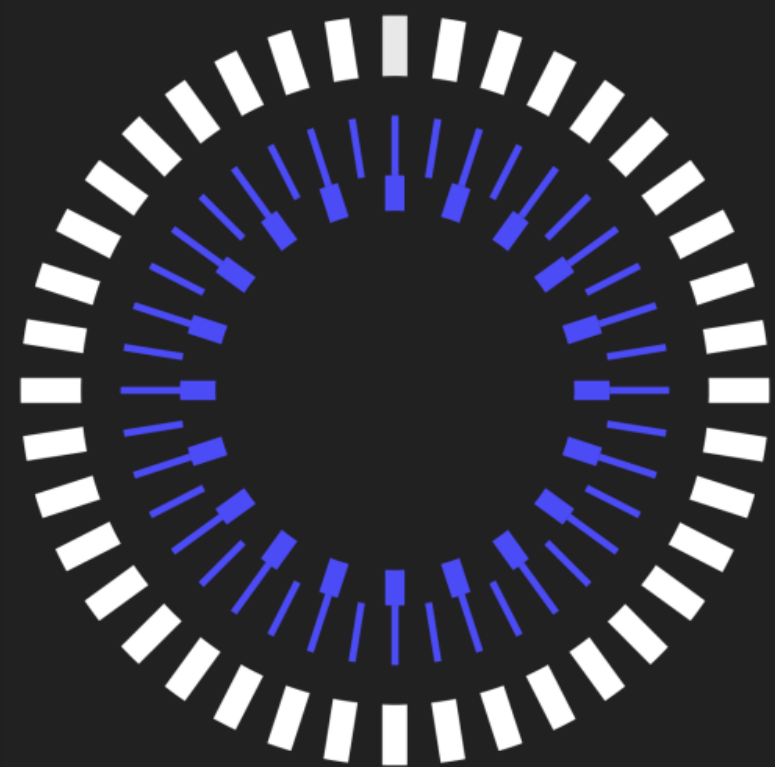


## ROBOTCORE Perception



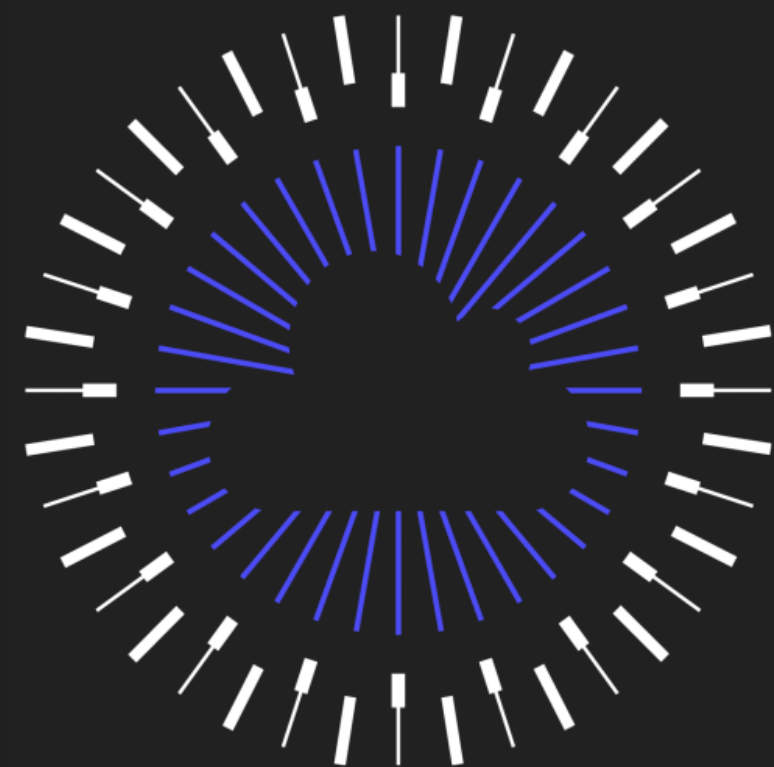
# ROBOTCORE tools and robot cores

ROS 2 API compatible hardware acceleration tools and robot Intellectual Property (IP) cores. Increasing your robot's performance, including latency, power efficiency and platform scalability.



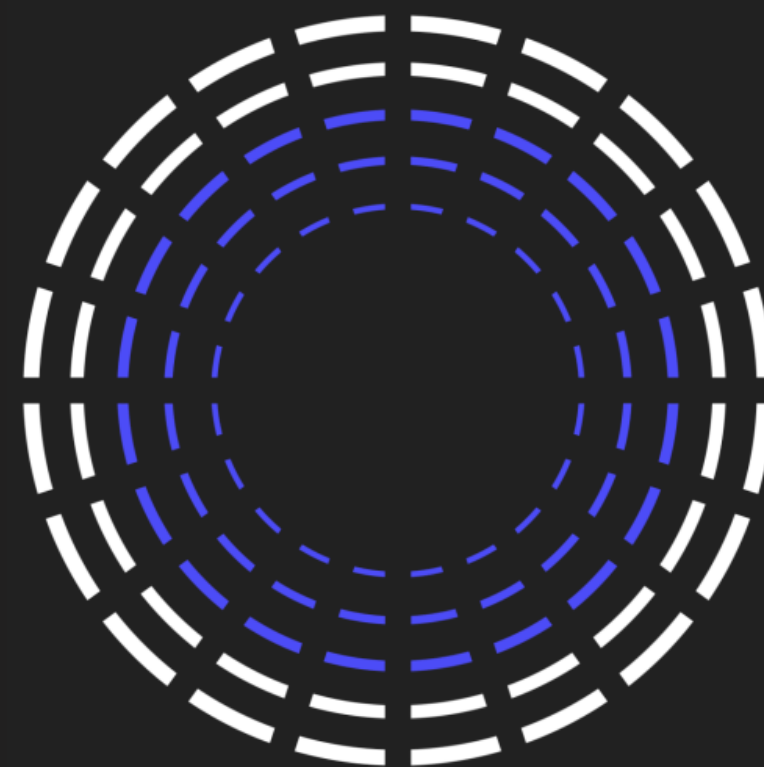
**ROBOTCORE®**  
**Framework**

# Hardware acceleration framework for ROS and ROS 2.



**ROBOTCORE®**  
**Cloud**

Speed-up ROS 2  
graphs with/in  
the cloud.



**ROBOTCORE®**  
Perception

# Accelerated ROS 2 perception stack.

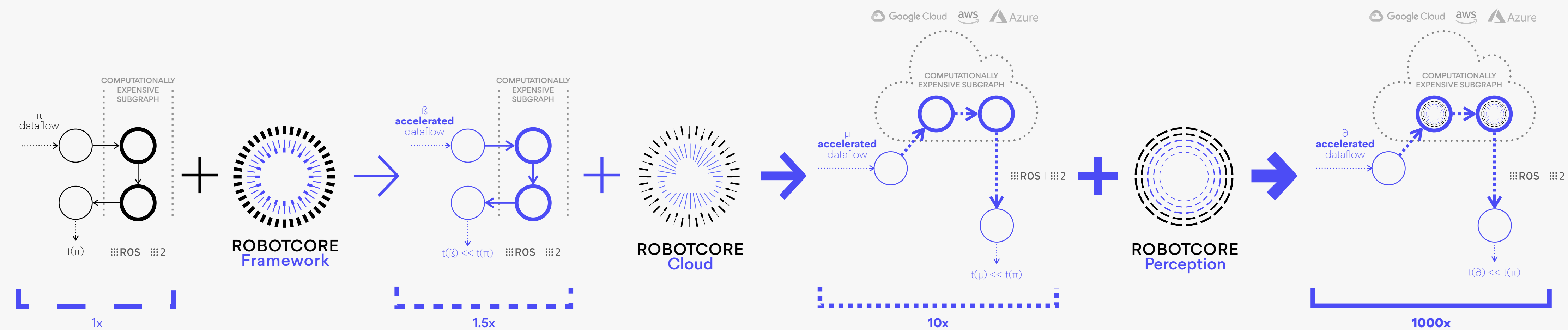


**ROBOTCORE®**  
**Transform**

# Accelerated ROS 2 coordinate transformations (**tf2**).

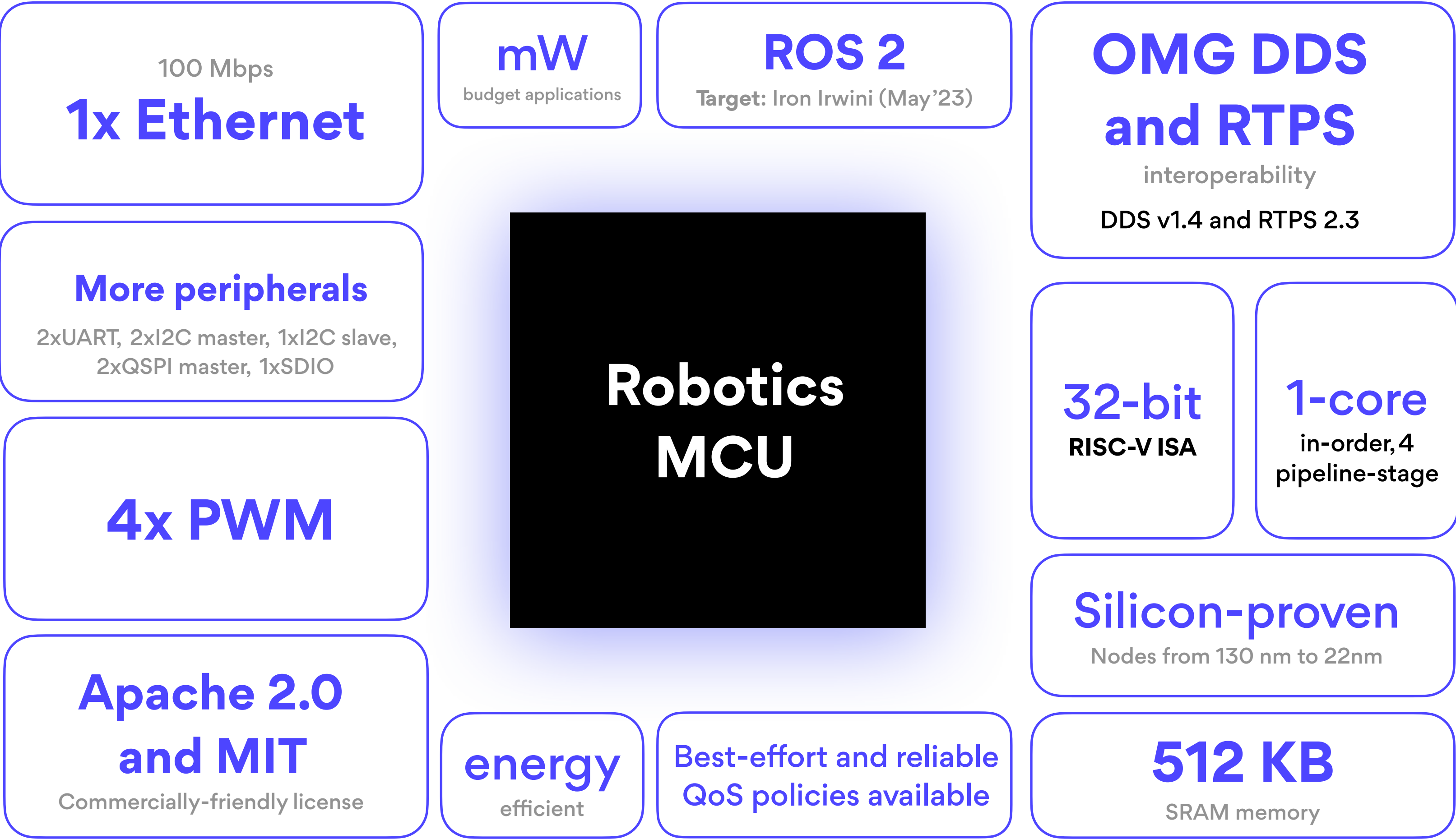


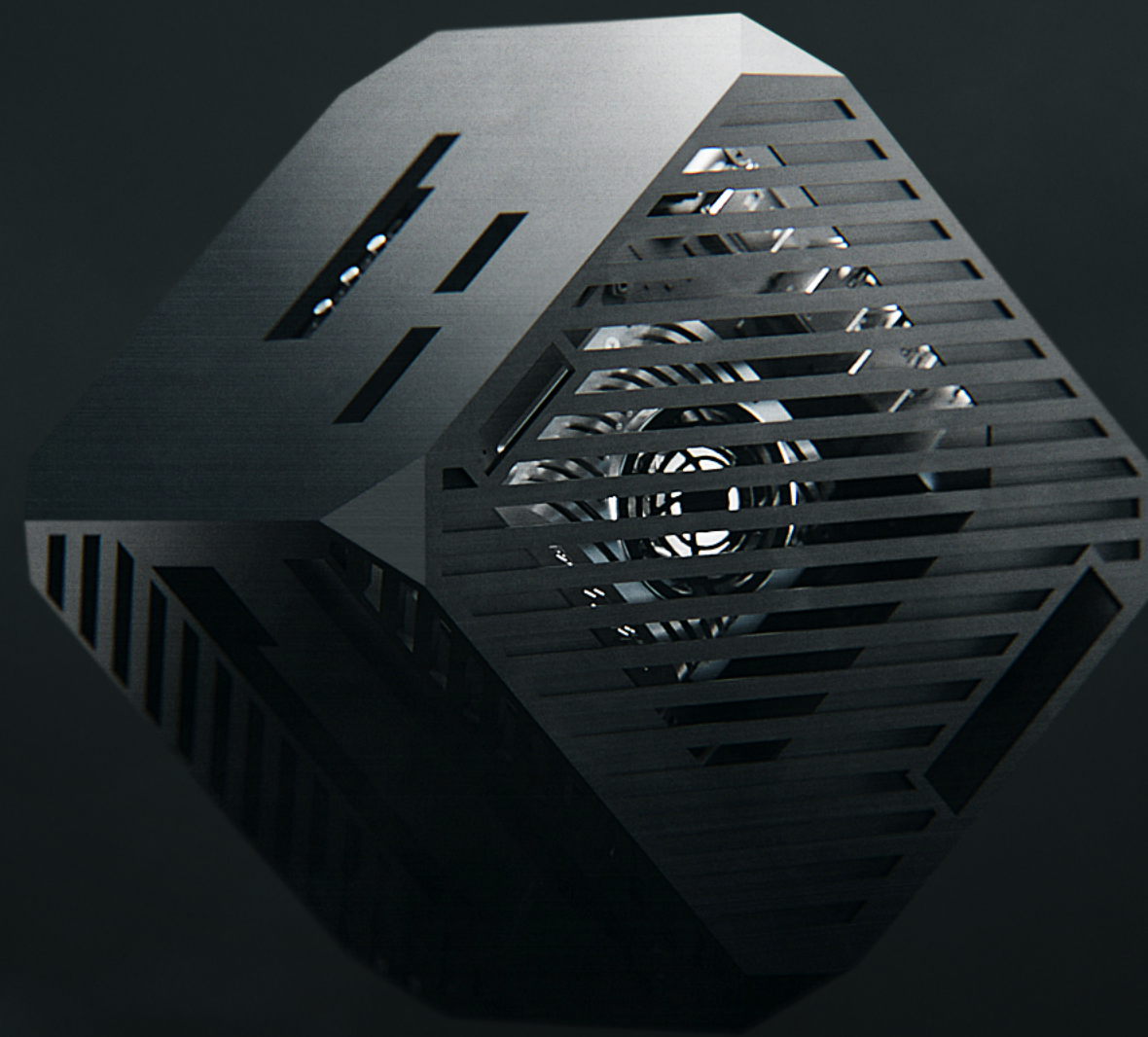
**FogROS2 and ROBOTCORE Cloud:** Tools to speed-up ROS 2 graphs with the cloud, and in the cloud.



<https://github.com/BerkeleyAutomation/FogROS2>

**Robotics MCU:** A **robotics microcontroller** unit (MCU) powered by RISC-V and ROS 2

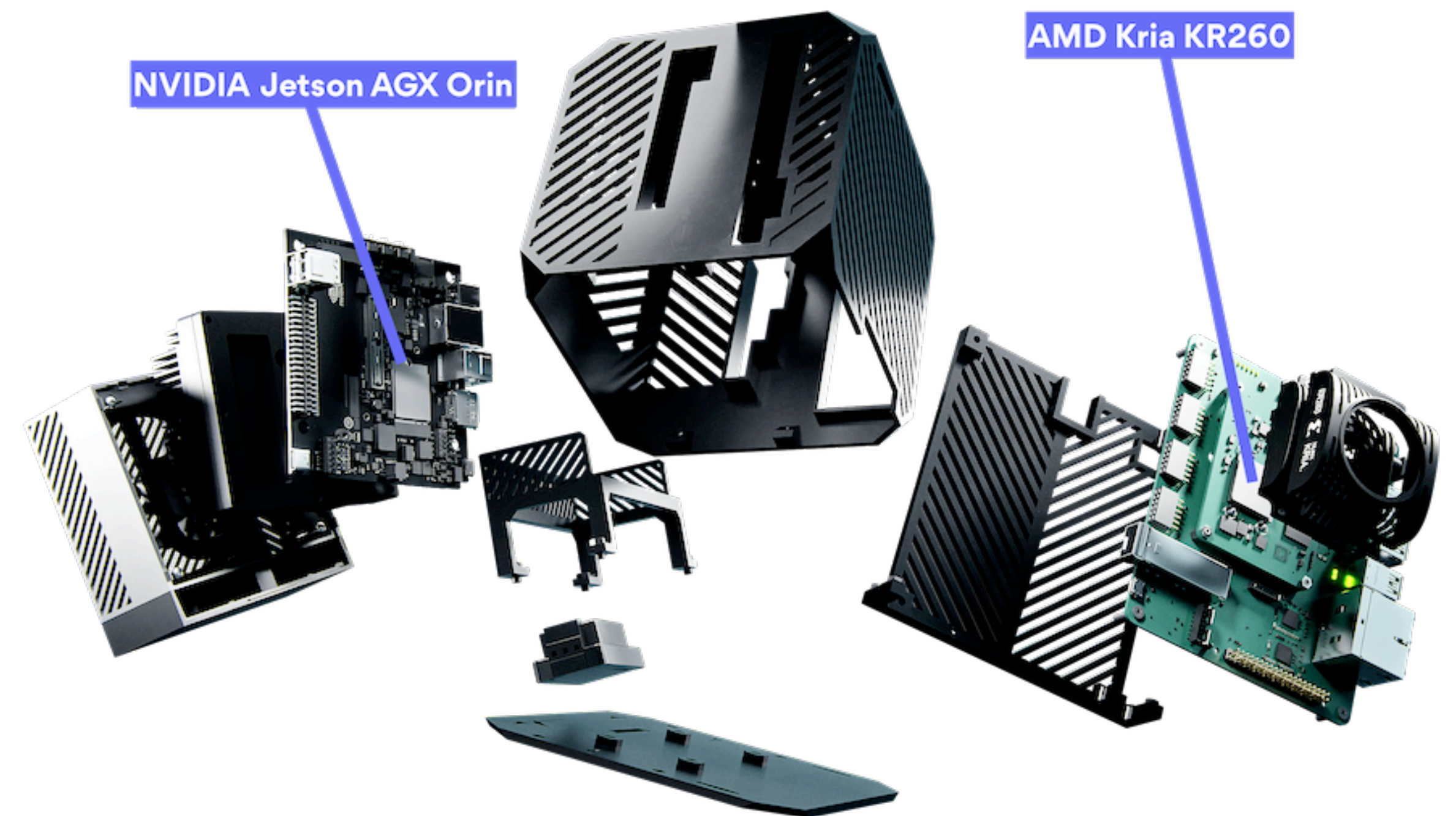




ROBOTC●RE

## ROBOTCORE: The **Robotic Processing Unit** specialized in ROS computations ➡

ROBOTCORE® is a robot-specific processing unit that helps map Robot Operating System (ROS) computational graphs to its CPUs, GPU and FPGA efficiently to obtain best performance. It empowers robots with the ability to **react faster**, consume **less power**, and deliver **additional real-time** capabilities.



[https://github.com/ros-acceleration/robotic\\_processing\\_unit](https://github.com/ros-acceleration/robotic_processing_unit)

# Thanks

Q&A