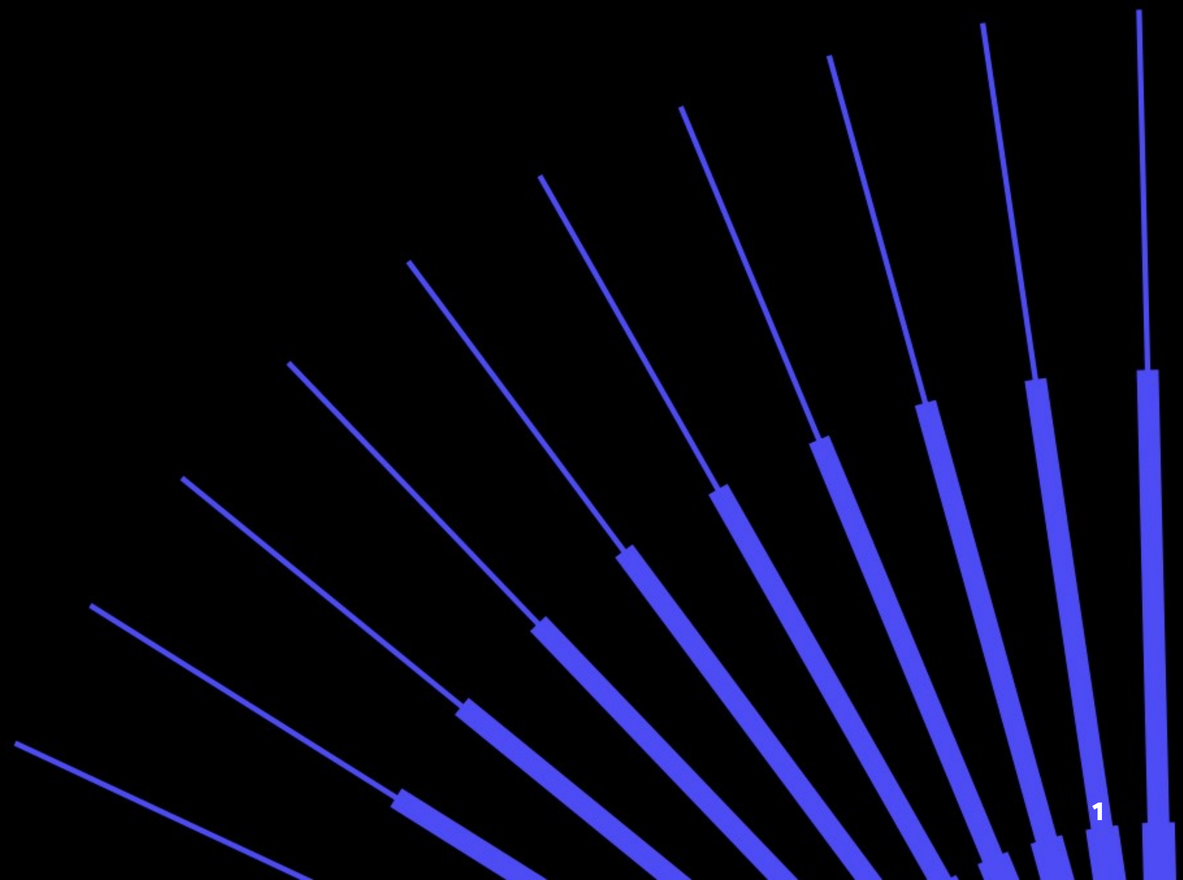
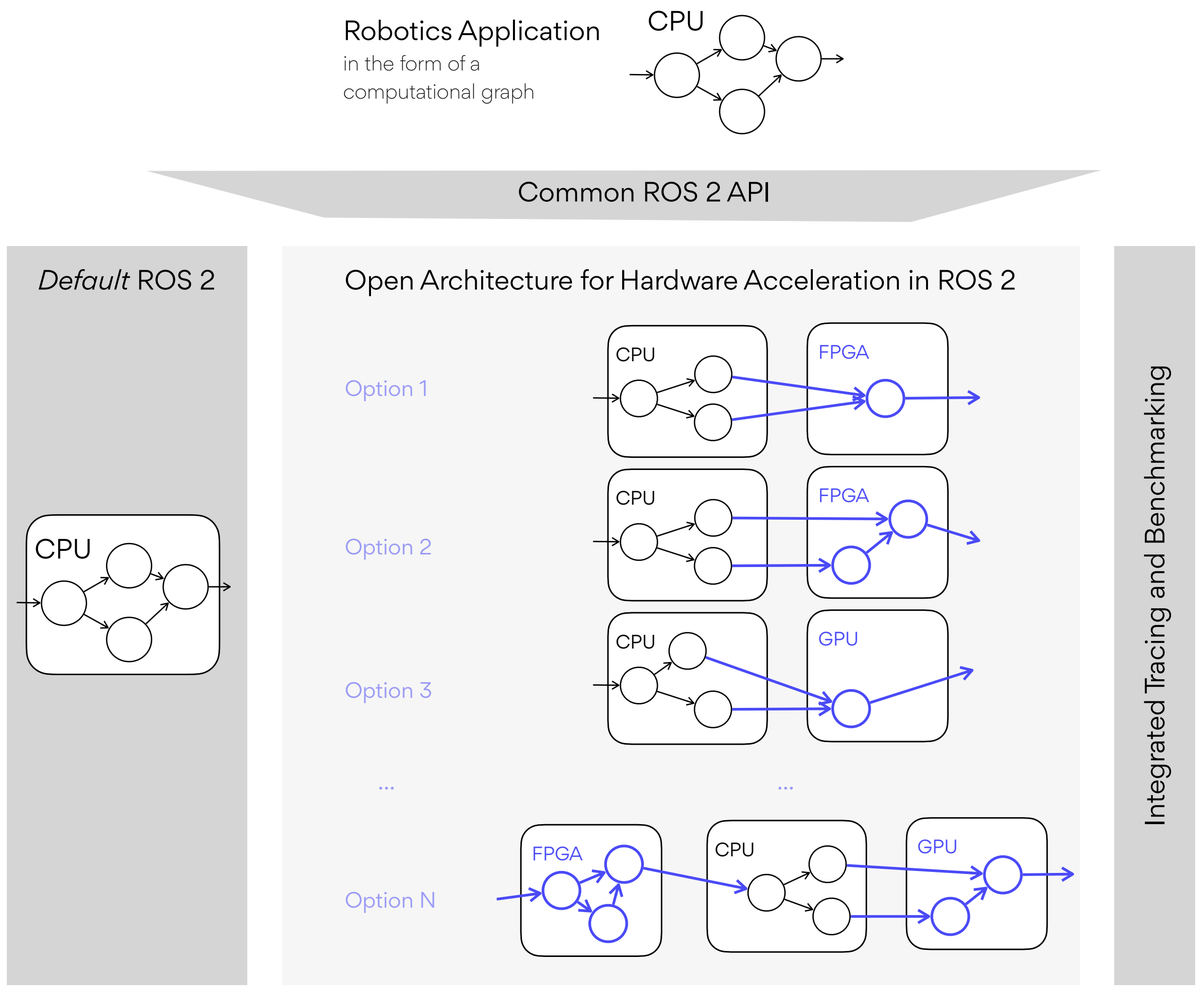
ROBOTCORE

Hardware Acceleration framework for ROS

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

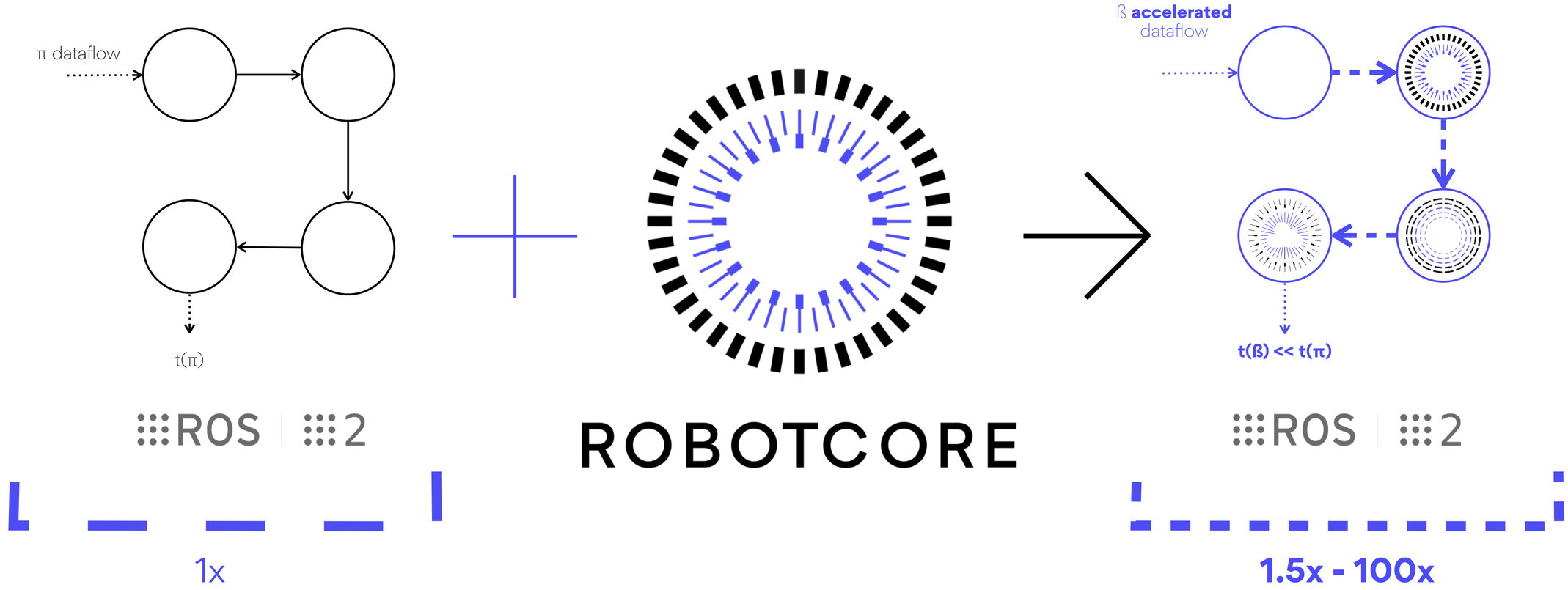


ROBOTCORE


Robotics Application
in the form of a
computational graph
CPU
Common ROS 2 API
Default ROS 2
CPU
Open Architecture for Hardware Acceleration in ROS 2
Option 1
CPU
FPGA
Option 2
CPU
FPGA
Option 3
CPU
GPU
...
...
Option N
FPGA
CPU
GPU
Integrated Tracing and Benchmarking

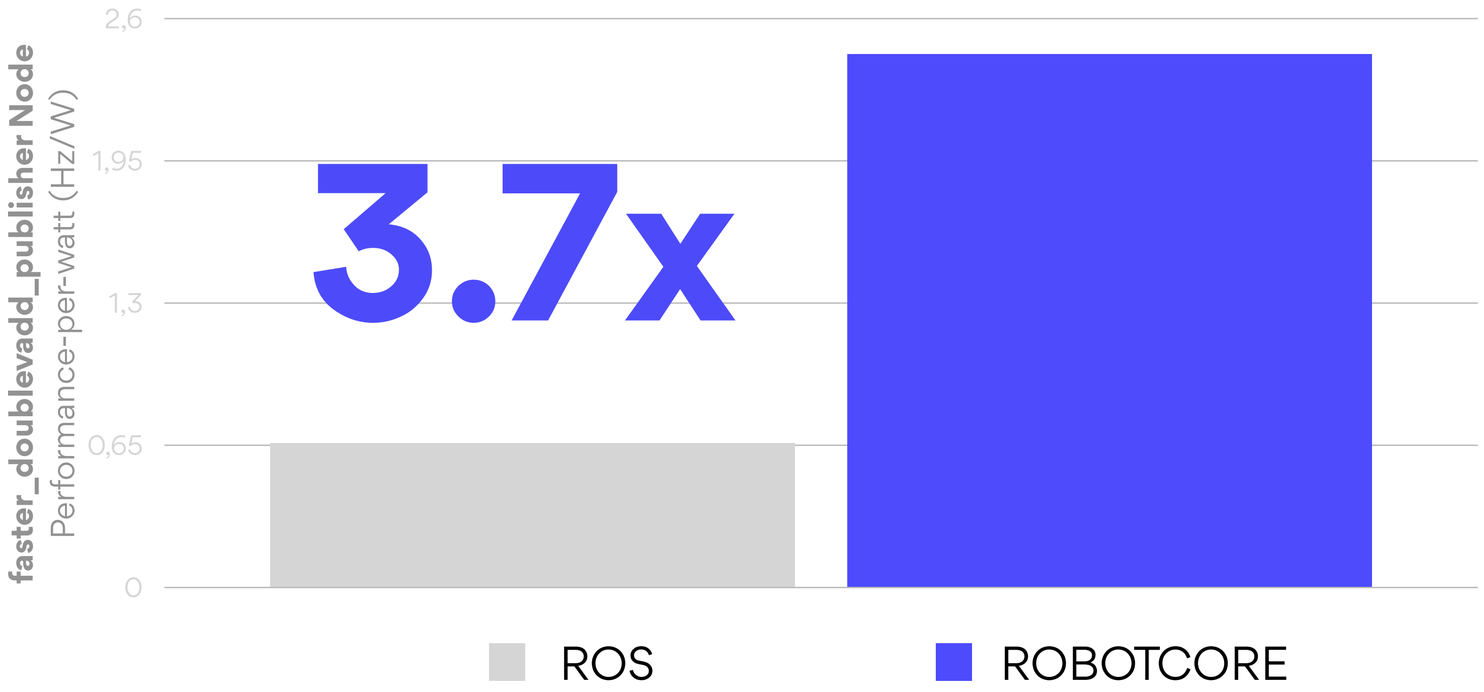
ROBOTCORE

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.

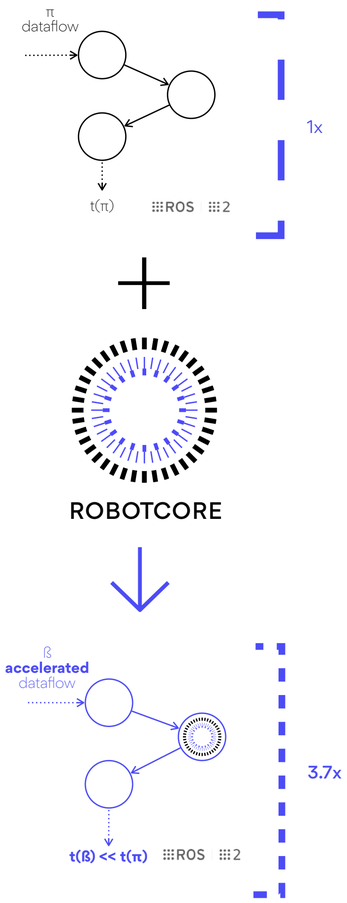


ROBOTCORE

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.

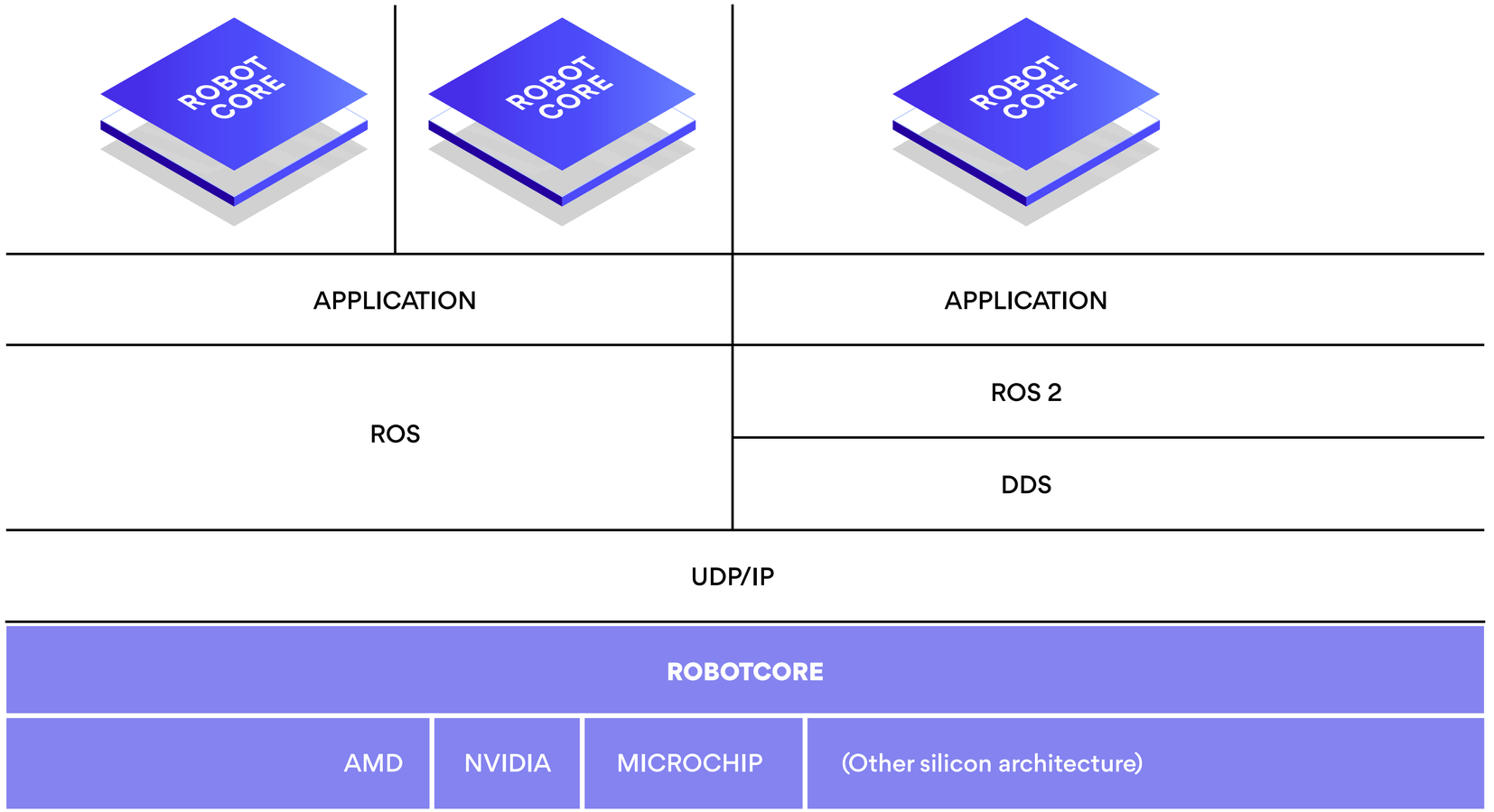


Performance-per-watt (Hz/W) measured during iterations 10-100 using faster_doublevadd_publisher. ROS baseline produced using an AMD KV260 with a Quad-core arm Cortex-A53. ROBOTCORE numbers with an AMD KV260 leveraging both CPU and FPGA.



ROBOTCORE:
ROS and ROS 2 support

Extends the ROS and ROS 2 build systems to allows roboticists to generate acceleration kernels in the same way they generate CPU libraries. Support for legacy ROS systems, and extensions to other middlewares is also possible.



ROBOTCORE: Production-grade multi-platform ROS support with **Yocto** (REP 2000 ➦)

Instead of relying on common development-oriented Linux distros (such as Ubuntu), our contributions to Yocto allow to build a customized Linux system for your use case with ROS, providing *unmatched granularity, performance and security*.

XILINX | AMD

NVIDIA | MICROCHIP

KR260

KV260

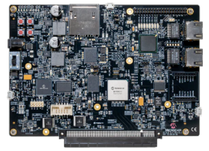
ZCU102

ZCU104

Jetson Nano

Jetson Xavier NX

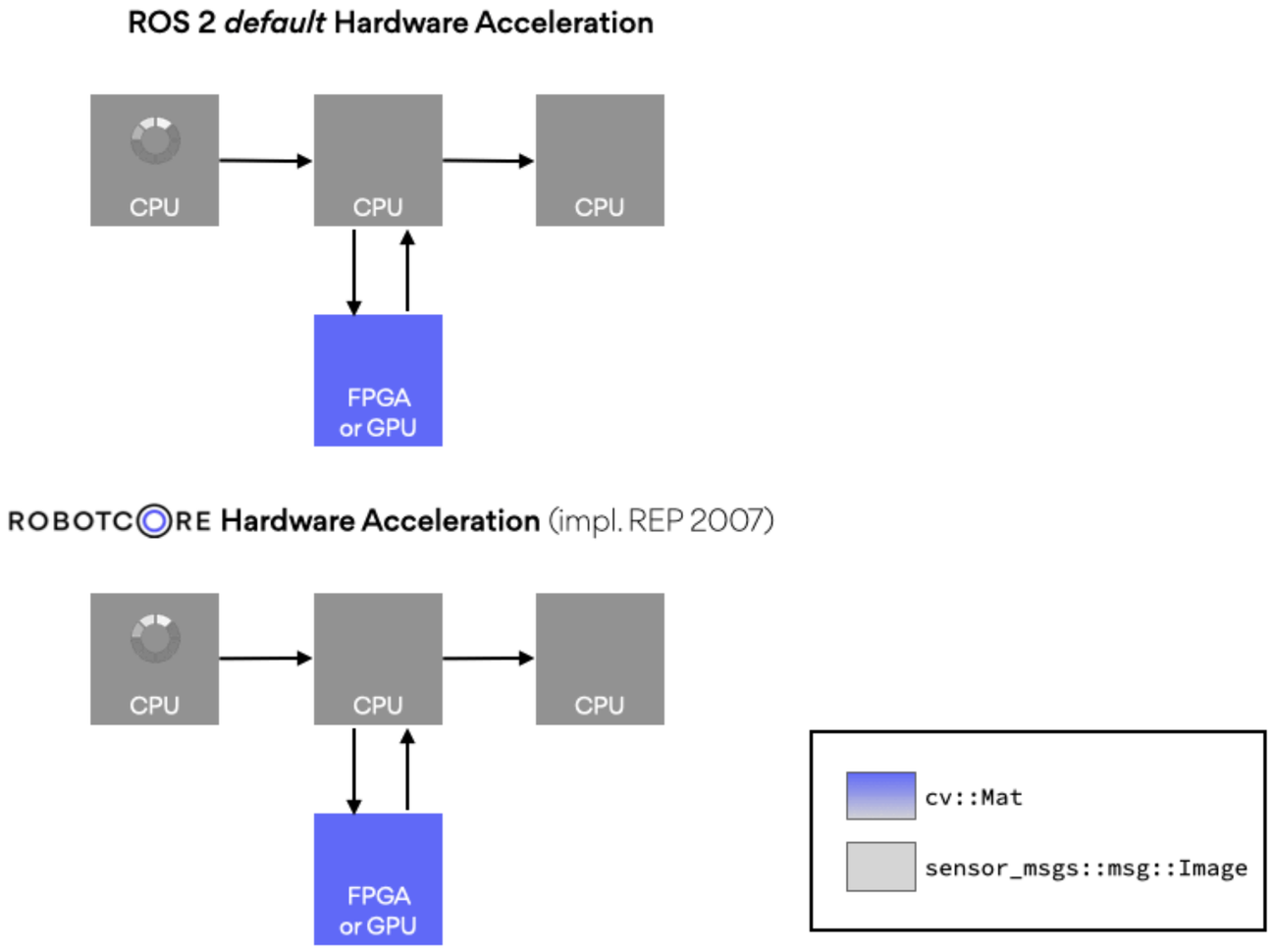
Jetson AGX Xavier



PolarFire Icicle

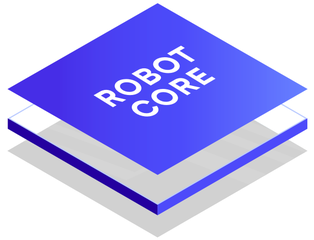
ROBOTCORE:
Type adaptation (REP 2007 ➦)

An extension to `rclcpp` that will make it easier to convert between ROS types and custom, user-defined types for Topics, Services, and Actions.



ROBOTCORE:
ROS 2 Hardware Acceleration Architecture and Conventions (REP 2008 ➦)

Implementing the open architecture for hardware acceleration of REP 2008, ROBOTCORE™ deals with vendor proprietary libraries for hardware acceleration in robotics. It helps accelerate computations, increase performance and abstract away the complexity of bringing ROS computational graphs to any of the supported silicon architectures. All while delivering the common ROS development flow.



APPLICATION

ROS 2

DDS

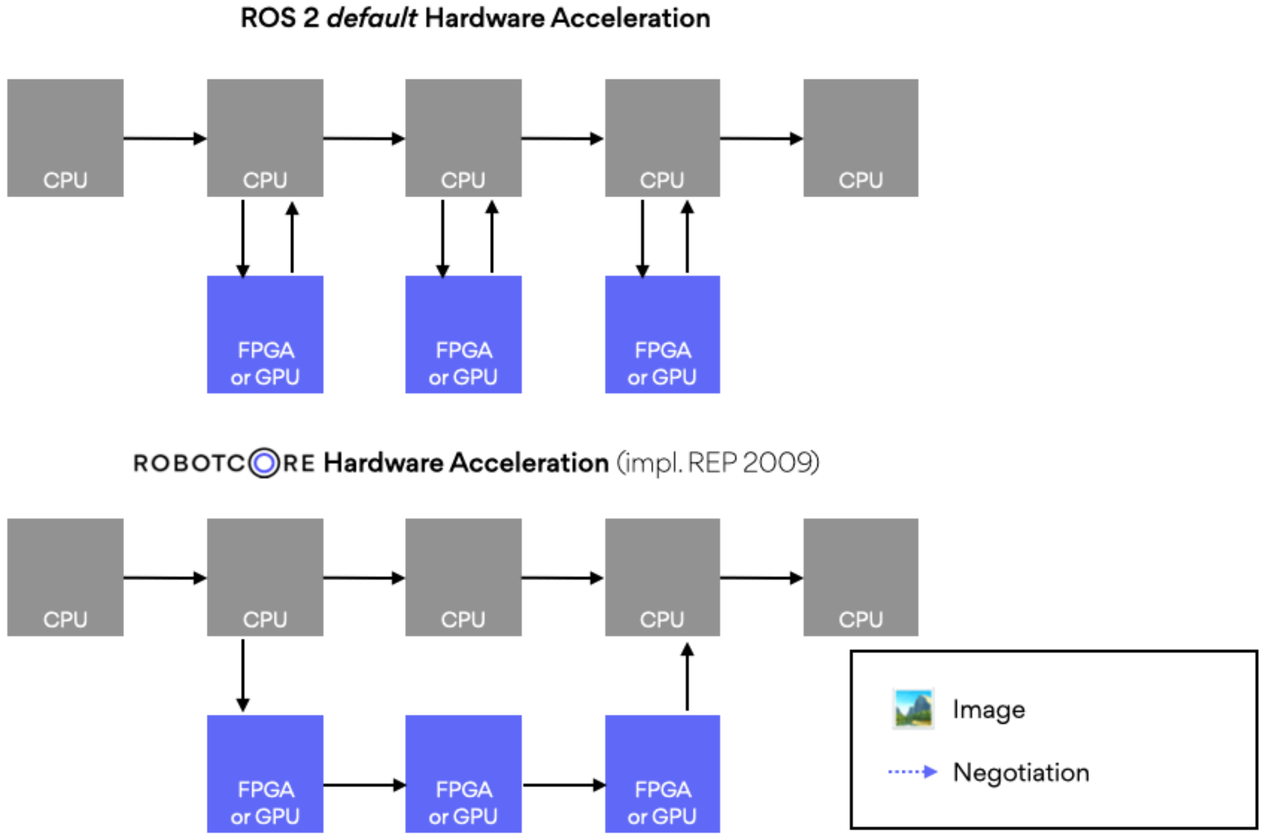
UDP/IP

ROBOTCORE

AMD | NVIDIA | MICROCHIP | (Other silicon architecture)

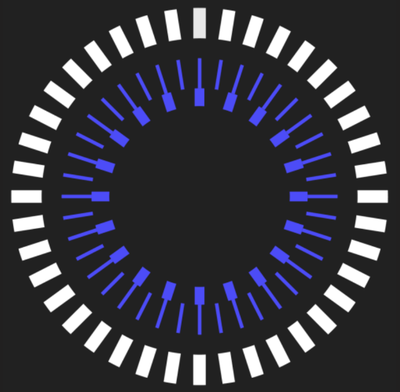
ROBOTCORE:
Type negotiation (REP 2009 ➦)

Type negotiation feature allow ROS 2 `Nodes` to dynamically negotiate the message types used by publishers and subscriptions, as well adaptively modifying the behavior of publisher and subscriptions. With type negotiation `Nodes` can a) publish different types of messages depending on the graph, b) publish multiple formats at the same time, c) enable only necessary publishers and subscriptions, d) delay publisher and subscription preferences while waiting for additional information from the graph.



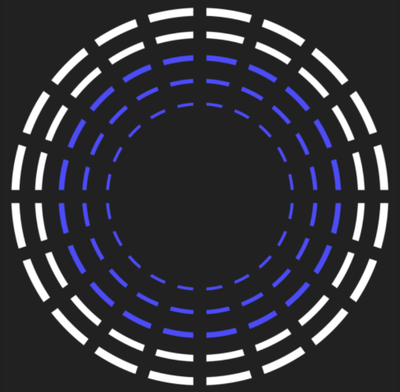
ROBOTCORE and robot cores

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™

Hardware acceleration framework for ROS and ROS 2.



ROBOTCORE™ Perception

Accelerated ROS 2 perception stack.

ROBOTCORE™ Transform

Accelerated ROS 2 coordinate transformations (`tf`).

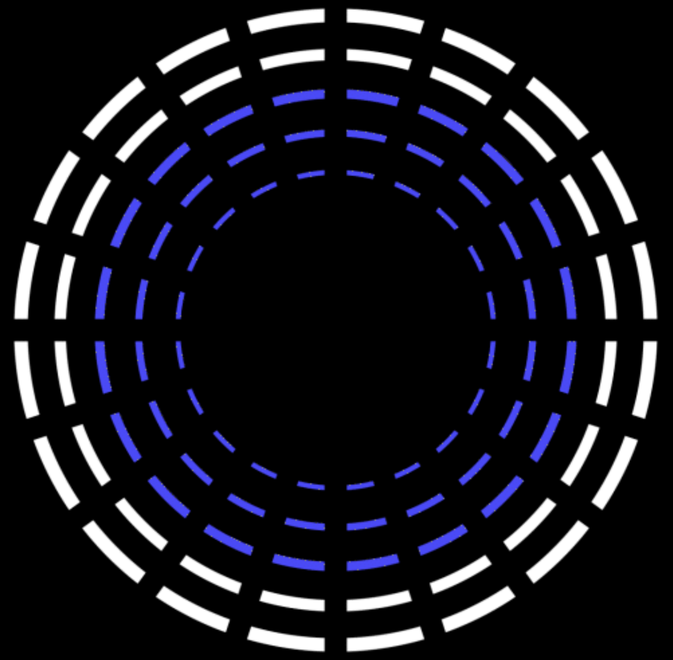
ROBOTCORE™ Cloud

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

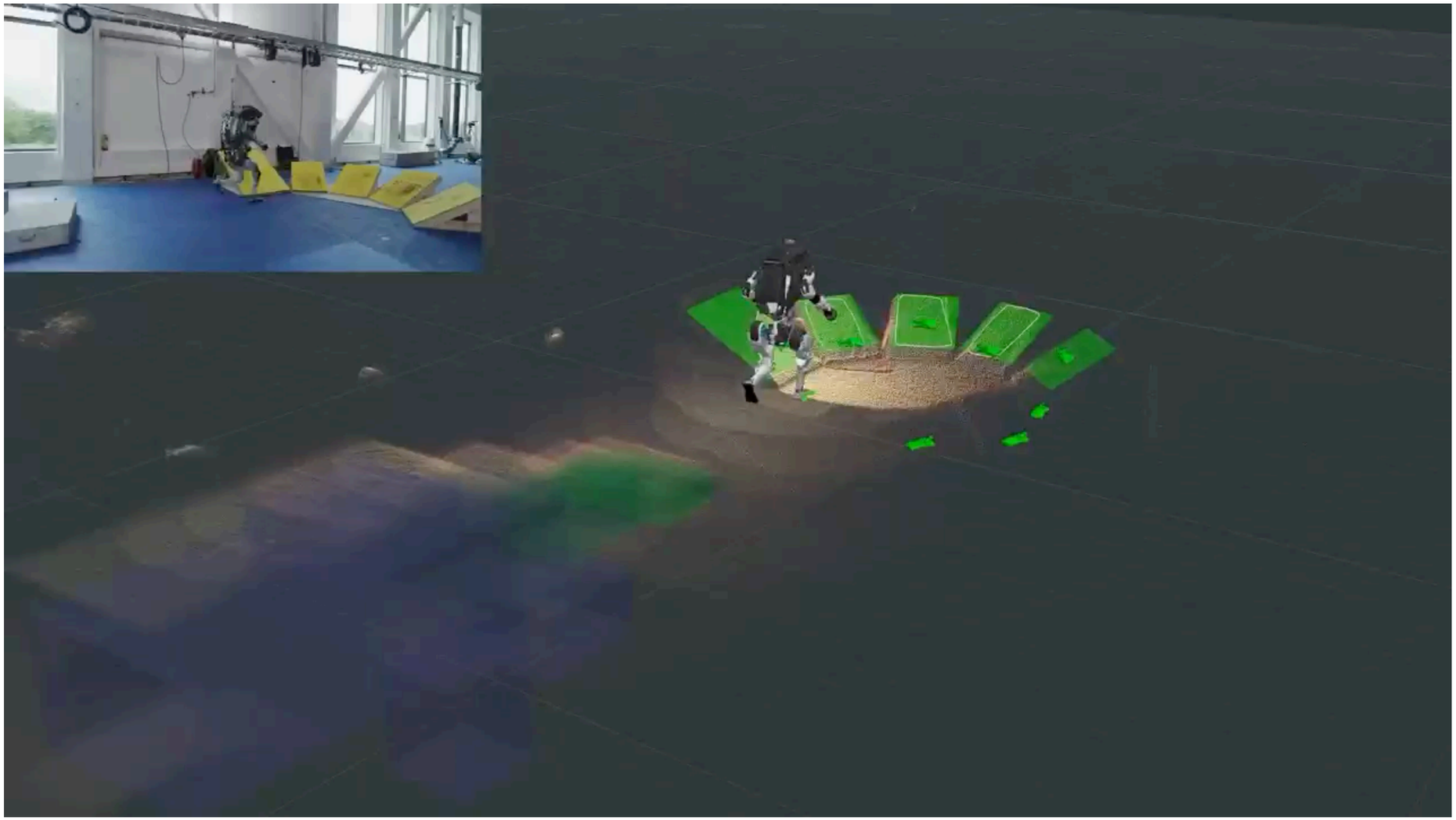
ROBOTCORE Perception

Speed up your ROS perception pipelines

An optimized hardware accelerate robotic perception stack to reduce runtime latency and increase determinism and throughput. API-compatible with the ROS 2 perception stack.

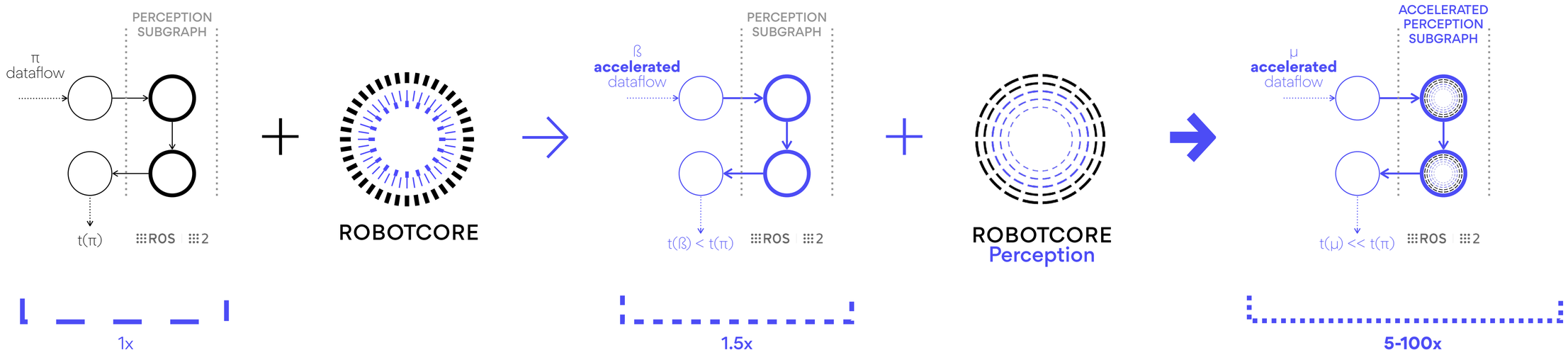


ROBOTCORE Perception



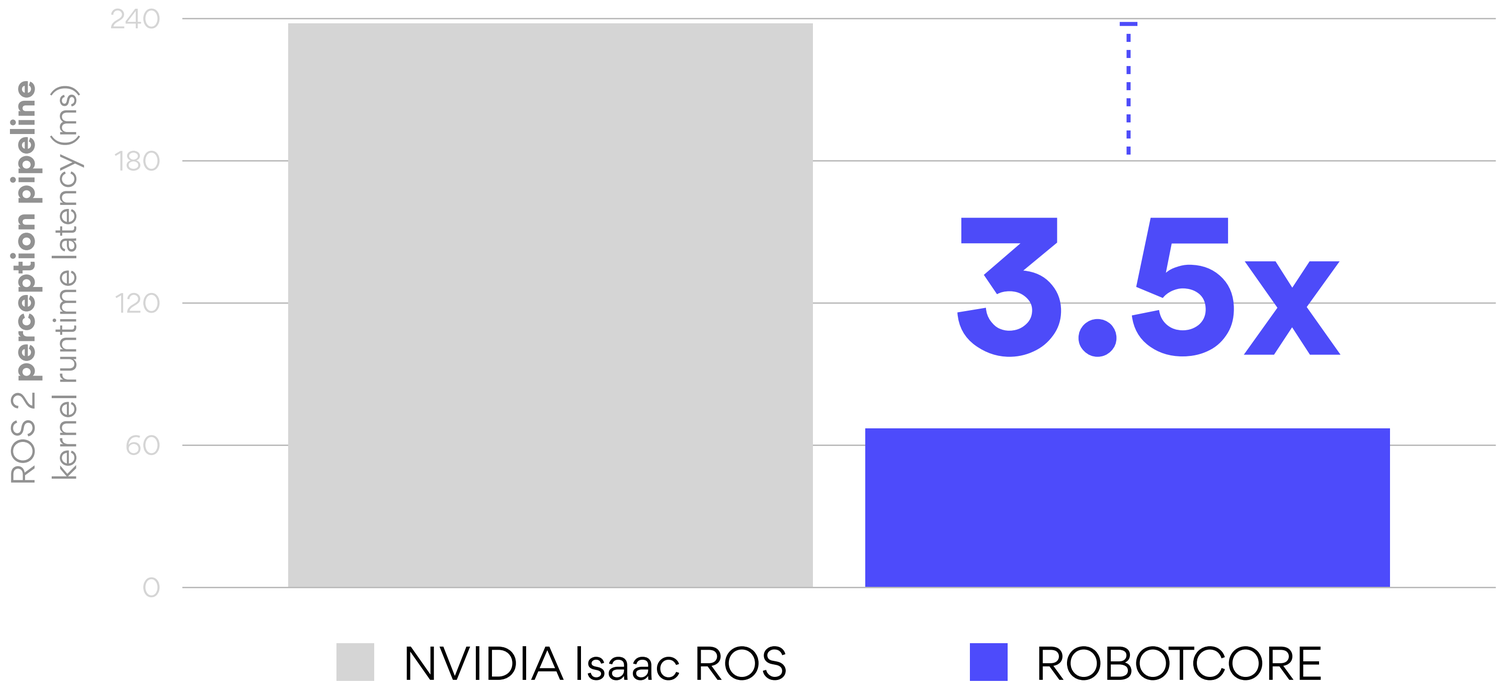
ROBOTCORE Perception

ROBOTCORE Perception is an **optimized robotic perception stack 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.

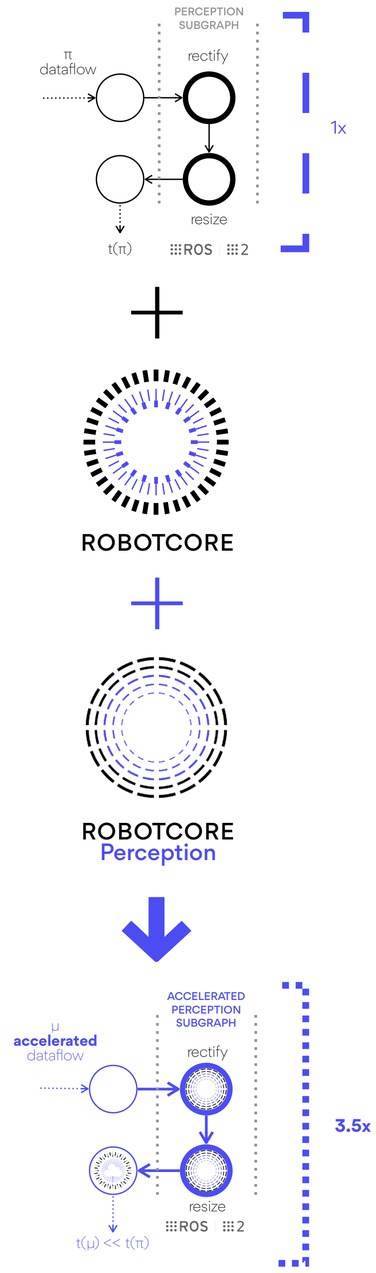


ROBOTCORE Perception: 2 nodes perception pipeline

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.

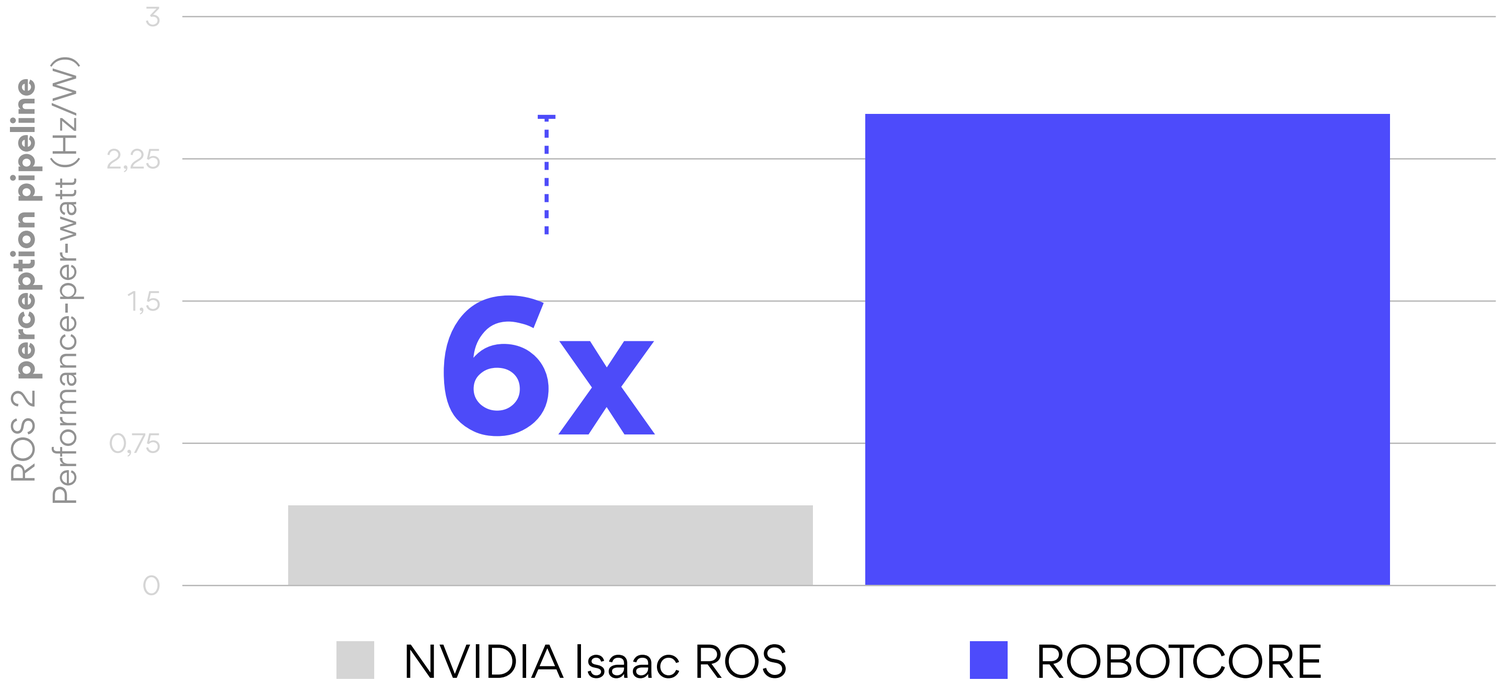


ROS 2 perception pipeline (2 nodes) - kernel runtime latency (ms). Measured ROBOTCORE Perception pipeline runtime latency running on an AMD KV260 versus NVIDIA Isaac ROS running in a Jetson Nano 2GB. Measurements present the kernel runtime in milliseconds (ms). Source code available at perception_2nodes.

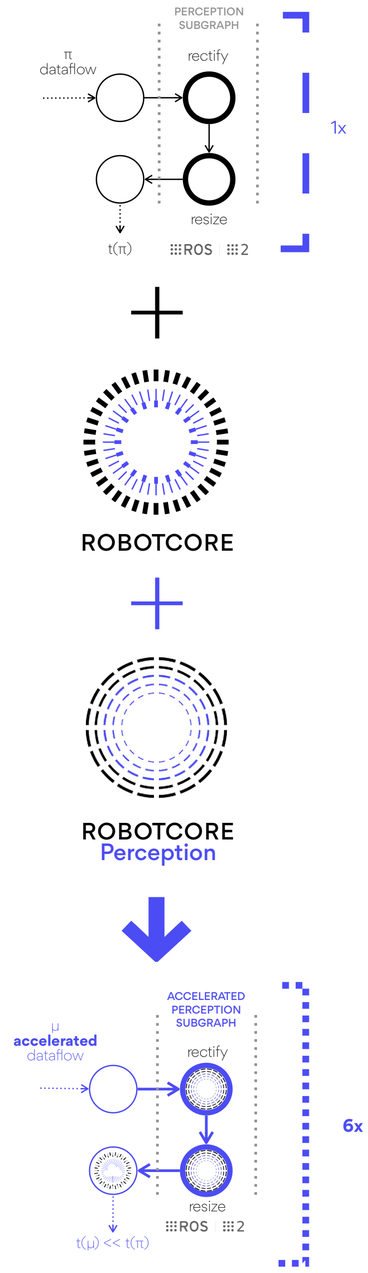


ROBOTCORE Perception: 2 nodes perception pipeline

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.

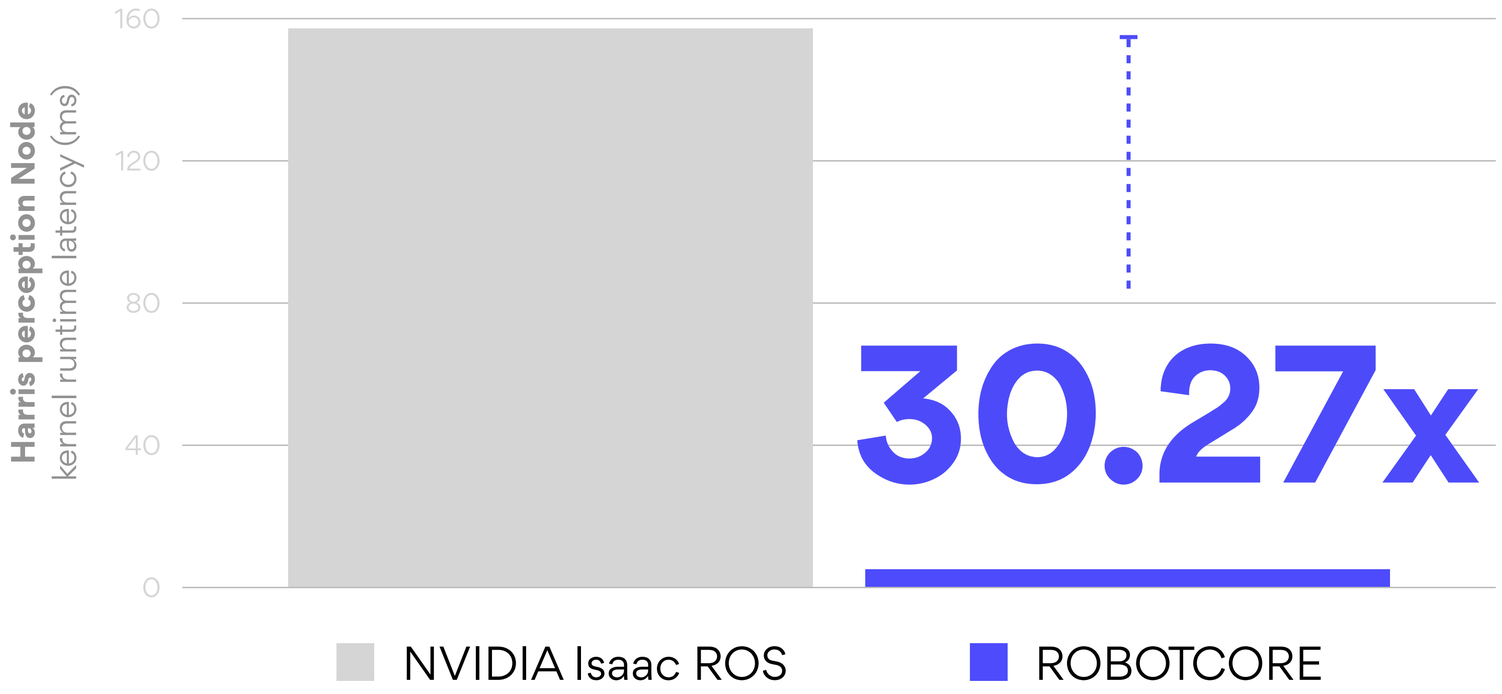


ROS 2 perception pipeline (2 nodes) - Performance-per-watt (Hz/W). Measured ROBOTCORE running on an AMD KV260 versus NVIDIA Isaac ROS running in a Jetson Nano 2GB. Measurements present a 2-Node (`rectify` and `resize` operations) pre-processing perception graph performance-per-watt (Hz/W). Source code available at perception_2nodes.

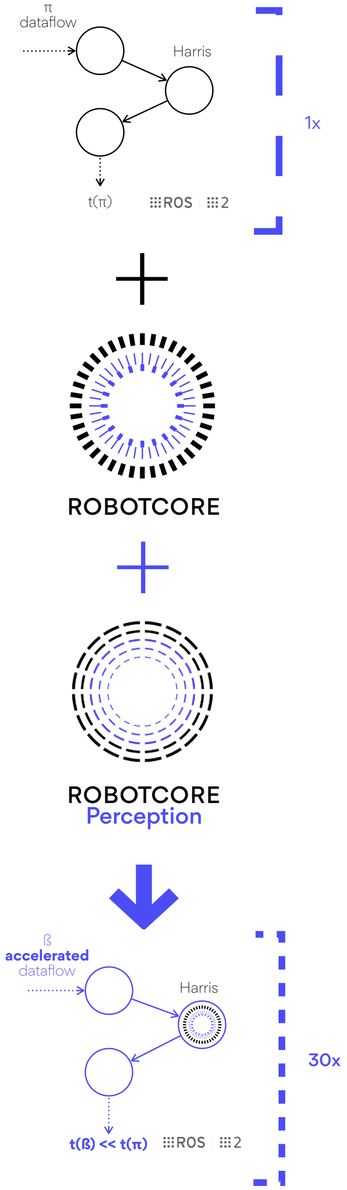


ROBOTCORE Perception: Harris perception Node

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.

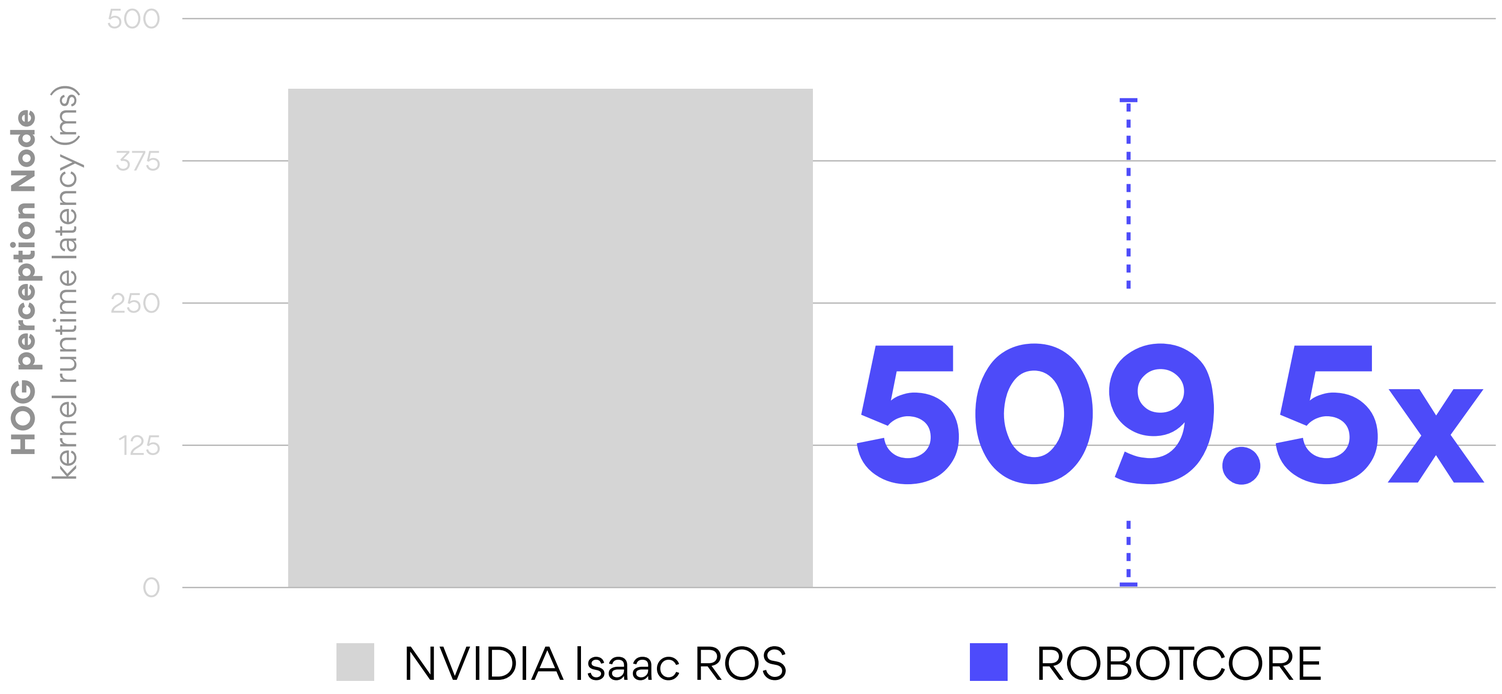


Harris - kernel runtime latency (ms). Measured ROBOTCORE Perception running on an AMD KV260, NVIDIA Isaac ROS running in a Jetson Nano 2GB. Measurements present the kernel runtime in milliseconds (ms) and discard ROS 2 message-passing infrastructure overhead and host-device (GPU or FPGA) data transfer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

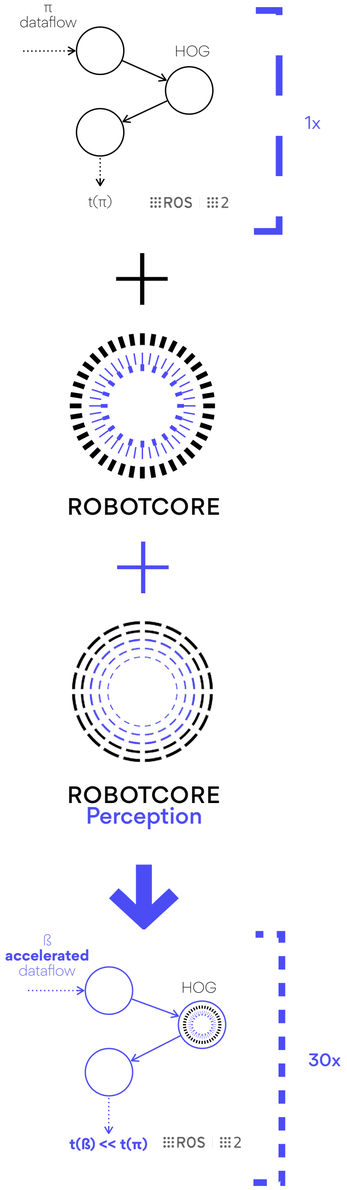


ROBOTCORE Perception: HOG perception Node

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.



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



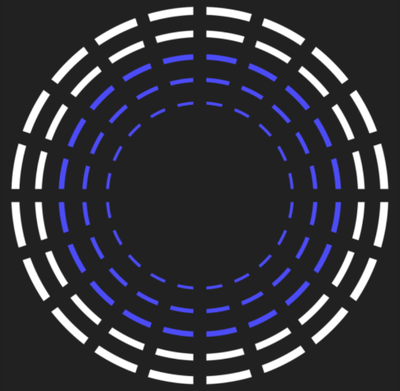
ROBOTCORE and robot cores

ROS 2 API-compatible **pre-built Intellectual Property (IP) cores**, so that you don't spend time reinventing the wheel and re-developing what already works.

ROBOTCORE™

Hardware acceleration framework for ROS and ROS 2.

robot cores



ROBOTCORE™
Perception

Accelerated ROS 2 perception stack.

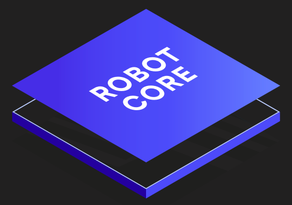
ROBOTCORE™
Transform

Accelerated ROS 2 coordinate transformations (`tf`).

ROBOTCORE™
Cloud

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

Build your own **robot cores**



APPLICATION

ROS 2

DDS

UDP/IP

ROBOTCORE			
AMD	NVIDIA	MICROCHIP	(Other silicon architecture)