Let's MicroROS: Learning the Basics of MicroROS

What is Microros ?

- Microros is a stack which integrates microcontrollers seamlessly with standard ROS 2 and brings all major ROS concepts such as nodes,publisher, subscriptions, parameters, and lifecycle onto deeply embedded systems.
- This enables accessing all software using the same ROS tools and APIs, regardless of the underlying computing hardware and operating system.
- The micro-ROS stack can be used with various open-source RTOS, including FreeRTOS, Zephyr, and NuttX,but also comes with support for bare-metal use-cases.

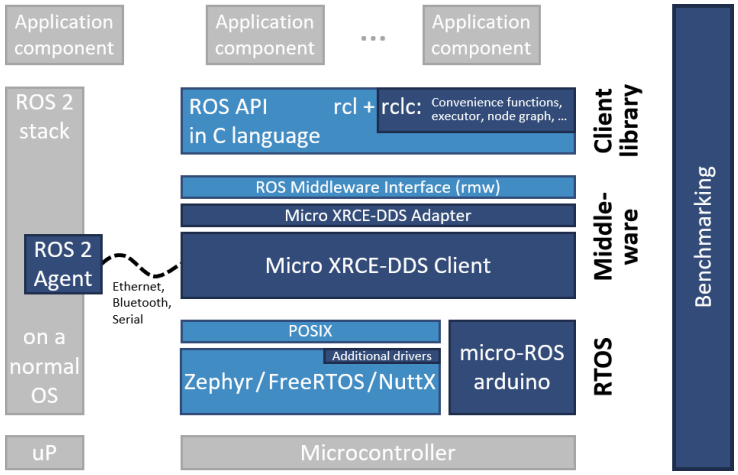
History of Microros..

The RIOT-ROS2 project and the preparation of a new standard DDS For Extremely Resource Constrained Environments (DDS-XRCE) led to the foundation of the micro-ROS project, backed by the EU-funded research and innovation action OFERA.The micro-ROS project was launched in 2018 with the following three goals :

1. Seamless integration of microcontrollers with ROS 2.
2. Ease portability of ROS 2 code to microcontrollers.
3. Ensure long term maintenance of micro-ROS stack.

To achieve those goals, the founding partners of micro-ROS have designed a software stack that utilizes the layered architecture of the standard ROS 2 stack and integrates seamlessly with DDS. The micro-ROS stack reuses as many packages as possible of the standard ROS 2 stack.

Architecture



1. Client Library :

- The rcl implements the most fundamental ROS concepts like node, publisher,subscription, client, service, and clock in the C programming language so that the high-level language client libraries can rely on the same implementation by binding rcl via native or foreign-function interfaces.
- In addition, micro-ROS provides the rclc packages,which extend rcl to make the combination rcl+rclc a feature-complete client library for the C programming language.Also, few features from rcl that are not available on a microcontroller are disabled.

2. Middleware layers :

- The middleware layer consists of :-

1. **ROS middleware interface (rmv) :** The rmw defines a generic and slim interface to abstract from the underlying middleware being used for communication. It assumes three common concepts only: (1) naming of endpoints, (2) publish-subscribe communication, and (3) client-service (request-response) communication. These concepts are provided not only by implementations of the DDS standard but also by many other middlewares.
2. **Adapter :** The adapter acts as a bridge between micro-ROS and the selected middleware. It translates the communication protocols and features of micro-ROS into the language understood by the chosen middleware. This adapter needs to conform to the specifications set by the rmw interface.
3. **Micro XRCE-DDS :** The micro-ROS stack uses an implementation of the new DDS-XRCE standard provided by eProsima as open-source software under the name Micro XRCE-DDS . It comes with an agent called microros agent,that bridges between standard DDS and DDS-XRCE. Micro XRCE-DDS supports several transports for the connection between the microcontroller and the main microprocessor running the Agent. These transports include UDPv4, UDPv6, TCPv4, TCPv6, and serial

3. RTOS :

Micro-ROS adopts the assumption of POSIX but makes smaller changes to rcutils, using the same changeset mechanism as explained above for rcl: filesystem operations are disabled, dynamic memory allocations (in the context of error reporting) are prevented, and a user-level implementation of 64-bit atomic operations is provided.

Now, let get hands on microros through examples

References :

- Microros offical documentation
- ROS Book Series