Supporting TOPPERS/ASP3 Kernel to mROS to Improve its Capability

Hiroi Imanishi[1,a)] Hideki Takase[1,2] Kazuyoshi Takagi[1] Naofumi Takagi[1]

Abstract: We are developing and researching mROS, that is a lightweight runtime environment. It enables ROS to run on mid-range embedded devices which Linux cannot operate on. In this research, we address to make a real-time kernel used in mROS to adopt TOPPERS/ASP3 kernel. We expect to control ROS nodes with high precision while reducing power consumption and improve mROS versatility. In this paper, we report current status of development and contribution of our work.

Keywords: ROS, Embedded devices, Real-time kernel

1. Introduction

These days, the demand for mobile robots for supporting social life is increasing. These robots often run with energy of the internal power supply. In addition, the services requested by users often require high accuracy. So, it is essential to improve accuracy and energy consumption of robots in order to enhance quality of services of them.

Many kinds of software frameworks to support robot systems development are attracting attention. Among them, ROS (Robot Operating System) [1][*1] is the most popular framework among them. ROS expresses a program to have functions as one unit called "node" and realizes a robot system by combining nodes according to purposes. Nodes are registered with ROS master, which plays the role of a name server to manage topic names and node names. Communication between nodes is performed based on subscription/publication method that identifies the type of data by topic name and subscribes messages from publisher inquired from ROS master. ROS provides the communication layer between nodes as a middleware while exploiting the process management system and file system provided by Linux kernel.

ROS provides the implementation that is supposed to run on Linux/Ubuntu. Therefore, ROS requires to adopt high performance and power hunger devices which Linux can be operated on. To solve this issue, we are researching a lightweight runtime environment to operate ROS nodes on embedded devices. mROS [2] provides the communication library to enable ROS nodes to run on embedded devices.

In this research, we try to support TOPPERS/ASP3 kernel to mROS. ASP3 kernel is the next generation version of ASP kernel which is currently adopted in mROS. By this research, it is expected to improve the capability of mROS including versatility, accuracy and power consumption.

[1] Kyoto University,Yoshida-Honmachi, Kyoto 606-8501, Japan
[2] JST PRESTO, Kawaguchi-shi, Saitama 332-0012, Japan
[a)] emb@lab3.kuis.kyoto-u.ac.jp
[*1] In this paper, we only discuss ROS 1.

2. mROS

We have developed a lightweight runtime environment for ROS nodes, called mROS, which is designed to be operated on embedded devices with a mid-range microprocessor. We assume a distributed robot system that consists of the host device, on which native ROS is operated in Linux, and edge devices, on which Linux cannot run. We aim to execute ROS nodes as tasks on real-time kernel to make it easy to guarantee real-time performance of the ROS nodes.

The microcomputer used in mROS must adopt TCP/IP protocol stack for embedded devices to communicate with the communication layer of ROS. In addition, it is necessary to manage the communication process as well as the program resources on the embedded device. We satisfy the former requirements by adopting lwIP, the latter requirement by employing TOPPERS/ASP kernel, a real-time kernel. Also the task structure of mROS consists of four system tasks, which provide communication functions of the system. They are as follows, as shown in Fig. 1.

XML_MAS TASK
It communicates with ROS master and other nodes in XML-RPC protocol. In particular, publisher nodes and subscriber nodes are registered with ROS master via this task.

XML_SLV TASK
It communicates with ROS master and other nodes in XML-RPC protocol. In particular, it accepts responses from ROS master and topic requests from subscriber nodes.

SUB TASK
It registers subscriber to ROS master via XML_MAS TASK, and if its topic request is accepted and TCPROS connection is established, subscibes the data periodically.

PUB TASK
It registers publisher to ROS master via XML_MAS TASK. Also, if XML_SLV TASK accepts topic requests from subscriber nodes, this task sends TCPROS connection header to establish the TCPROS connection with them and then publishes data.

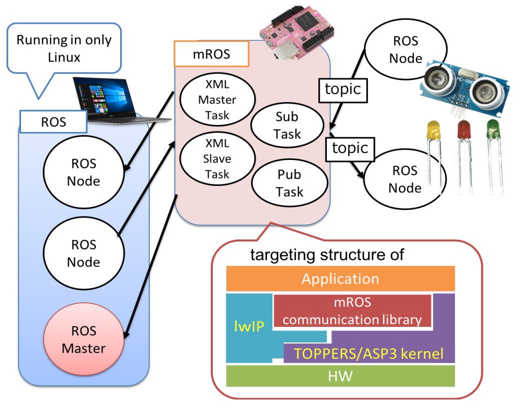

Fig. 1 The overview of mROS

3. Supporting ASP3 Kernel to mROS

3.1 Merits of ASP3 Kernel

TOPPERS/ASP kernel [3] currently used in mROS is a real-time kernel that has been expanded and improved based on the standard profile of the μITRON 4.0 specification. TOPPERS/ASP kernel is suitable for embedded systems because several resources such as tasks, semaphores and data queue are statically created. ASP3 kernel is implemented by adding various functions to ASP kernel. The features of ASP3 kernel are 1.being tickless and 2.high-resolution time management function in microseconds.

Regarding 1., ASP kernel updates the system time by increasing ticks by periodic interrupts. However, in this case, even if the CPU is idle with no execution process, timer interrupt processing is executed, so a certain amount of power is consumed. By making the real-time kernel tickless, the timer interrupt becomes on-demand type and interrupts only the required timed event, so it can be expected that more power consumption is reduced while CPU is idle. Then, regarding 2., the unit for the system time of ASP kernel is milliseconds, on the other hand, one of ASP3 kernel is microseconds. The latter kernel enables users to control with higher accuracy. We think that supporting ASP3 kernel to mROS and acquiring these benefits will result in power savings and improved accuracy of a distributed robot system including embedded microcomputers.

3.2 Enhancing Versatility of mROS

Currently, mROS employs GR-PEACH as board, TOPPERS/ASP kernel as a real-time kernel and mbed library including lwIP as TCP/IP protocol stack. It is necessary to clarify dependencies and interfaces with other layers to support various boards, real-time kernels and TCP/IP protocol stacks to mROS in the future. In particular, we try to clarify ideas and related APIs needed in mROS, for example, tasks, semaphores, data queues of various real-time kernels, and aim for a configuration that allows mROS users to select a real-time kernel with simple settings. In this research, we expect that supporting ASP3 kernel to mROS becomes a foothold to improve the versatility of it.

3.3 Current Status

We are developing mROS employing ASP3 kernel using SOLID [4] developed by Kyoto Microcomputer Co.,Ltd. SOLID is a development platform which a real-time kernel and a Clang compiler are integrated. The board for evaluation is equipped with RZ/A1H microcomputer like GR-PEACH, on the other hand, the size of Flash ROM is 4MByte as half as GR-PEACH. This platform employs ASP3 kernel as a real-time kernel and lwIP as TCP/IP protocol stack, so it is suitable for this research. Currently, this development is in progress in minimally configured ROS system that publishes messages from edge devices and subscribes them on the host PC. So far, we finished the following implementations.

- Registeration PUB TASK on an edge device with ROS master as a publish node.
- Acception the response from ROS master to XML_SLV TASK.

Figure 2 shows the task structure of mROS and current status of development.

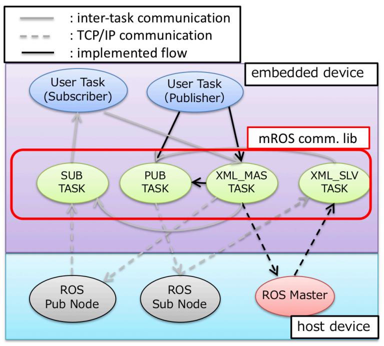

Fig. 2 The task structure of mROS and development status

4. Conclusion

In this research, we worked on supporting TOPPERS/ASP3 kernel to mROS, which is a lightweight runtime environment for ROS nodes. By achieving this challenge, it can be expected that we control ROS nodes running on embedded devices with less power consumption and more accurately. In addition, it is thought that the versatility of mROS can be improved by clarifying interfaces between mROS and other layers.

As application examples of mROS employing ASP3 kernel, in terms of power savings, IoT devices and mobile robots running by an internal battery are mentioned, and in terms of high accuracy, acquisition of high speed FA equipment location information is mentioned.

Acknowledgments The part of this work was supported by JST PRESTO Grant Number JPMJPR18M8 and JSPS KAKENHI Grant Numbers JP18K18024.

References

[1] Quigley, Morgan, et al. “ROS: an open-source Robot Operating System.” ICRA workshop on open source software. Vol. 3. No. 3.2. 2009.
[2] Takase, Hideki, et al. “mROS: A Lightweight Runtime Environment for Robot Software Components onto Embedded Devices.” Proceedings of the 10th International Symposium on Highly-Efficient Accelerators and Reconfigurable Technologies. ACM, 2019.
[3] TOPPERSproject:TOPPERS/ASP kernel, available from ⟨https://www.toppers.jp/asp-kernel.html⟩
[4] Kyoto Microcomputer Co.,LTD. SOLID available from ⟨https://solid.kmckk.com/SOLID/⟩