Homare: Platform for Aggregating Embedded Systems on Multi-core Processor

Kanta Okugawa, Koichi Mouri
Ritsumeikan University
{kokugawa,mouri}@asl.cs.ritsumei.ac.jp

1 Introduction

Embedded systems are becoming increasingly sophisticated and multifunctional. In particular, advanced and complex in-vehicle systems now consist not only of bare-metal applications that perform single operations, but also a variety of software, including RTOSs, GPOSs, and their applications. As software in embedded systems increases, the number of hardware components also grows, including embedded devices, peripheral devices, and the wiring that connects them. This leads to problems such as increased weight, space constraints inside the product, and complexity of hardware design. On the other hand, multi-core processors are also becoming widely used in embedded devices. However, many existing embedded software are designed with single-core and cannot take full advantage of multi-core. Additionally, porting these software to use multi-core efficiently is difficult. We solve these problems by running multiple embedded software, which have been running separately, on a single hardware equipped with a multi-core processor.

Several problems exist when porting embedded software and running multiple systems on the same hardware. One is that embedded software programmed to depend on specific cores for operation on a single core does not anticipate multi-core, leading to increased porting complexity when aggregating them. In addition, although virtualization methods enable multiple OSs to run through hardware emulation, the hurdle to guaranteeing real-time performance is high for embedded systems. Therefore, we propose Homare, a platform that enables the aggregation of embedded software focusing on improving portability and ensuring real-time performance.

2 Approach

Virtualization technology is used to run multiple systems on a single hardware. For embedded systems, existing hypervisors like Bao[1] and SPUMONE[2] use lightweight virtualization layers and static resource partitioning to enable lightweight execution. Other systems such as LynxSecure, QNX Hypervisor, and eMCOS Hypervisor adopt similar methods to meet the requirements of embedded systems. However, existing virtualization is too complex to reduce code size and overhead in portability and real-time performance.

Homare combines LPAR (Logical Partition) with minimal paravirtualization to achieve the aggregation shown in Figure 1. Essentially, using the LPAR, Homare allocates resources and boots the guest OS, allowing direct control of exclusive resources to ensure real-time performance. Only certain core-dependent processes and memory virtualization, device sharing are managed by paravirtualization through API calls. The functions that enable these configurations include APIs and a library of stubs that call the APIs, boot emulation, and hardware resource allocation management.

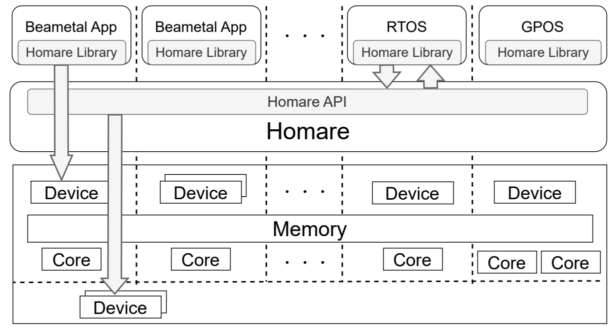

Figure 1: The overview of Homare.

3 Result

We implemented Homare on a Raspberry Pi 3 to aggregate bare-metal applications and four RTOSs: FreeRTOS, TOPPERS/ASP3, TOPPERS/FMP, and T-Kernel 2.0. Homare assigns and activates multiple cores for each RTOS, allowing guest operating systems to run independently on any core.

The evaluation focused on two essential requirements for aggregation: portability and real-time performance. The number of source code modifications required during the aggregation ranged from 4 to 19. These modifications can be extracted mechanically, so modification of the guest OS is easy enough to ensure portability of the aggregation. Measurements of API call and interrupt latency indicate a maximum overhead of approximately $6\mu s$ and $9\mu s$, respectively. Since the task cycle required in embedded systems is from 1 *ms* to 10 *ms*, an interrupt response time of a few μs is sufficient for real-time performance.

These results demonstrate Homare's effectiveness in aggregating multiple software while meeting the critical requirements of embedded systems.

References

[1] José Martins et al. 2020. Bao: A Lightweight Static Partitioning Hypervisor for Modern Multi-Core Embedded Systems. In *Workshop on Next Generation Real-Time Embedded Systems (NG-RES 2020) (OpenAccess Series in Informatics (OASIcs), Vol. 77)*. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 3:1–3:14.

[2] Wataru Kanda et al. 2008. SPUMONE: Lightweight CPU Virtualization Layer for Embedded Systems. In *2008 IEEE/IFIP International Conference on Embedded and Ubiquitous Computing*, Vol. 1. 144–151.

APSys '24, Sept 04–05, 2024, Kyoto Japan