LEVERAGING OPENAMP IN EMBEDDED MIXED-SAFETY CRITICAL SYSTEMS

Mridula Prakash

L&T Technology Services, CTO Office, Mysore, India

ABSTRACT

This aim of this paper is to understand OpenAMP framework and provide details on how to use OpenAMP for designing mixed-safety critical systems. OpenAMP is an open-source software framework that provides software components for working with Asymmetric multiprocessing (AMP) systems. The paper explains the software components of OpenAMP and provides details to use OpenAMP for embedded system designs.

KEYWORDS

OpenAMP, Multicore, Mixed Critical, & Embedded Systems.

1. INTRODUCTION

The electronics industry has seen a dramatic rise in the use of multicore processing in recent years. Multicore designs in embedded systems are now becoming mainstream due to market demand for increased performance, lower power consumption, and optimized costs. CPUs with diverse capabilities are therefore being clustered together to optimally handle different tasks in a single System on Chip (SOC). Multicore systems will enable data and task parallelism, with each core working independently.

1.1. Multicore Systems

In homogenous computing, all core in a multicore CPU are identical and execute the same instruction set. To overcome limitation of homogenous computing it gave rise to heterogenous computing, where the cores are not identical and implement different instruction. Essentially, there are two multicore system architectures – Asymmetric Multicore Processing (AMP) and Symmetric Multicore Processing (SMP). SMP provides an approach to multicore design in which all cores share the same memory, operating systems, and other resources. AMP, on the other hand, is an approach to multicore design in which cores operate independently and perform dedicated tasks.

An AMP system may be constructed from any combination of core architectures; all the cores may be identical or there may be a rich mixture of core types that includes conventional processing units as well as specialized cores, like digital signal processors (DSPs) for instance. Each core executes independently in an AMP architecture, with or without an operating system, and their operating system may be selected based on the required functionality.

Combinations of SMP and AMP yield good results in scenarios in which the main system runs on a few cores that use SMP and are helped by cores running AMP modes as software accelerators. It should be noted here that multicore applications can be implemented using a SMP-enabled

David C. Wyld et al. (Eds): EMSA, SEA, AIFU, NLCAI, BDML, BIoT, NCOM, CLOUD, CCSEA, SIPRO - 2022
pp. 41-46, 2022. CS & IT - CSCP 2022 DOI: 10.5121/csit.2022.120604

operating system, but that approach does not allow for independent workloads to be executed on different cores and does not support utilization of heterogeneous cores.

1.2. Challenges with an AMP System

In an AMP system, there is no unified operating system or scheduler managing all resources in the system.

1. An inter-core communication facility issues that require the cores to be protected from one another.
2. Boot order – the sequence in which the software on each core starts – may be important to avoid synchronization and security issues.
3. Debugging the disparate workloads running on the potentially heterogeneous cores can be quite challenging.

Hence, there is a need for a software framework which helps manage boot order, data exchange, and power consumption.

2. OPENAMP

Open AMP came into existence to manage boot order and communication between the cores that are running independently.

OpenAMP stands for Open Asymmetric Multi-Processing, an open-source software framework that provides software components for working with Asymmetric multiprocessing (AMP) systems. The framework is maintained by the OpenAMP project, which comprises of member companies, including, Xilinx, ARM, STMicroelectronics, Linaro, Texas Instruments, Wind River and Nordic Semiconductor.

The key components and capabilities provided by the OpenAMP Framework include:

1. remoteproc – This component allows for the Life Cycle Management (LCM) of remote processors from software running on a master processor. The remoteproc API provided by the OpenAMP Framework complies with the remoteproc infrastructure that is present in upstream Linux 3.4.x kernel onward. It has the ability to load firmware and to start and stop remote processors.
2. RPMsg – The RPMsg API facilitates Inter Processor Communications (IPC) between the independent software contexts running on heterogenous or homogeneous cores present in an AMP system. This API is compliant with the RPMsg bus infrastructure present in upstream Linux 3.4.x kernel onward.

The API enables applications to send and receive variable length binary message data, with the message format being defined by the application. It is well suited for exchanging asynchronous and event-based messages with remote processors.

2.1. OpenAMP Bootup

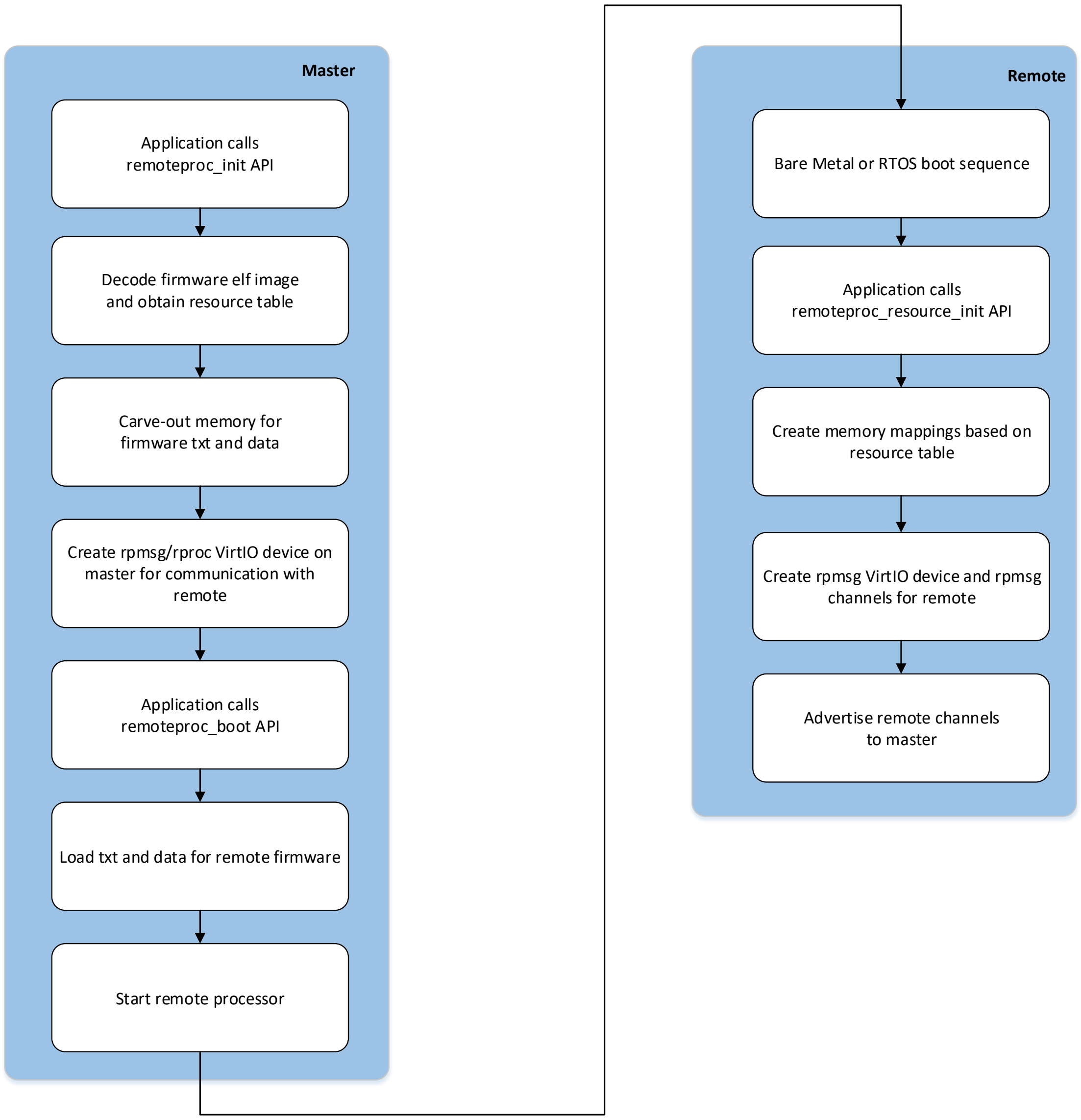
Master
Application calls remoteproc_init API
Decode firmware elf image and obtain resource table
Carve-out memory for firmware txt and data
Create rpmsg/rproc VirtIO device on master for communication with remote
Application calls remoteproc_boot API
Load txt and data for remote firmware
Start remote processor
Remote
Bare Metal or RTOS boot sequence
Application calls remoteproc_resource_init API
Create memory mappings based on resource table
Create rpmsg VirtIO device and rpmsg channels for remote
Advertise remote channels to master

Figure 1. Bootup with remoteproc: Conceptual Diagram

2.2. Software Environment or Configurations For OPENAMP

The OpenAMP Framework can be used with RTOS or bare metal contexts on a remote processor to communicate with Linux applications (in kernel space or user space) or other RTOS/bare metal-based applications running on the master processor through the remoteproc and RPMsg components. The architecture can be used in Data Intensive Application where more focus is on display interfaces in non-critical subsystems. The architecture can be tweaked to work in safety-critical systems. The OpenAMP Framework also serves as a stand-alone library that enables RTOS and bare metal applications on a master processor to manage the life cycle of remote processor/firmware and communicate with them using RPMs

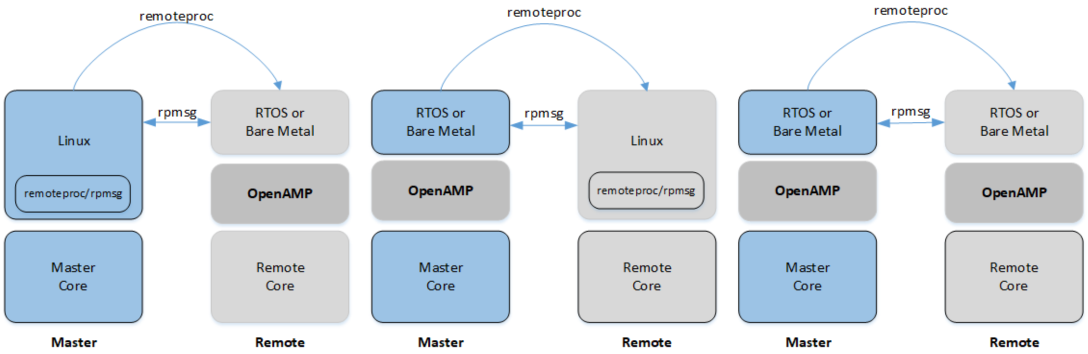

Figure 2: Software Environments/Configuration for OpenAMP

3. Design for Mixed Safety Critical Systems

The main requirement for a safety-critical system is the isolation to separate different software components from each other. Today, several hardware-assisted separation capabilities are provided by various MPSOC manufacturers to isolate safe and non-safe domain. The separation is required for the processing blocks, memory blocks, peripherals, and system functions. The OpenAMP framework helps facilitate the implementation of fault-tolerant systems. The OpenAMP framework can enable an RTOS on the safety critical processor, which is the system master and manage the critical system operations, to control the lifecycle on the application processor. If there is a failure, the RTOS can simply reboot the system without impacting the operation of the rest of the system.

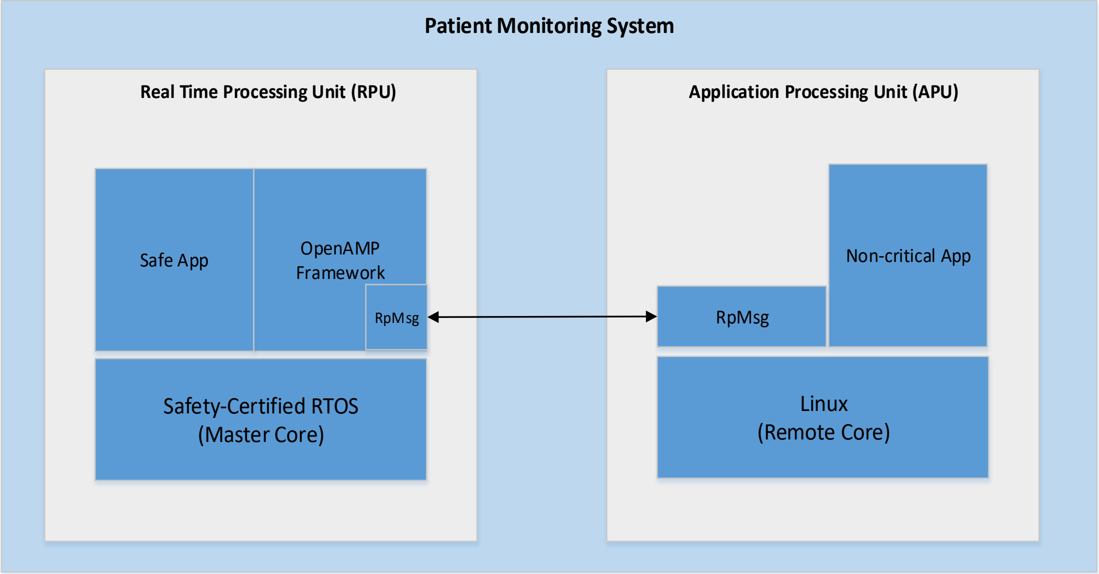
Patient Monitoring System
Real Time Processing Unit (RPU)
Safe App
OpenAMP Framework
RpMsg
Safety-Certified RTOS (Master Core)
Application Processing Unit (APU)
Non-critical App
RpMsg
Linux (Remote Core)

Figure 3. Example architecture diagram illustrating a mix-critical safety system in Patient Monitoring System

Let us take the example of a patient monitoring system (Figure 2), with a platform that compromises a cluster of ARM Cortex A5 as a Real-time Processing Unit (RPU) and ARM Cortex-A53 core as an Application Processing Unit (APU). The functional safety-certified software context RPU obtains the sensor data. The sensor data contains the patient vitals. The RPU main purpose is to monitor the patient vitals. The non-critical subsystem in the APU consists of the high-level operating system, which displays the data on an LCD and provides internet connectivity. The OpenAMP helps to isolate the critical from non-critical subsystems and communicate person vitals from RPU to APU using OpenAMP RPMsgs.

3.1. Hybrid Design Approach

The current trend is to merge both hypervisor and OpenAMP framework to form a hybrid design. This helps solve problems related to mixed critical systems in various applications areas across automotive, medical, banking, and many others.

In modern automotive applications like Digital cockpit, there is a need to separate safety critical from the non-critical components. To address such a situation, a hybrid design (Figure 3) can be used where both the advantages of hypervisor and OpenAMP framework can be combined into the architecture to create a zone of trust for the communication where data exchange can happen safely between remote and master.

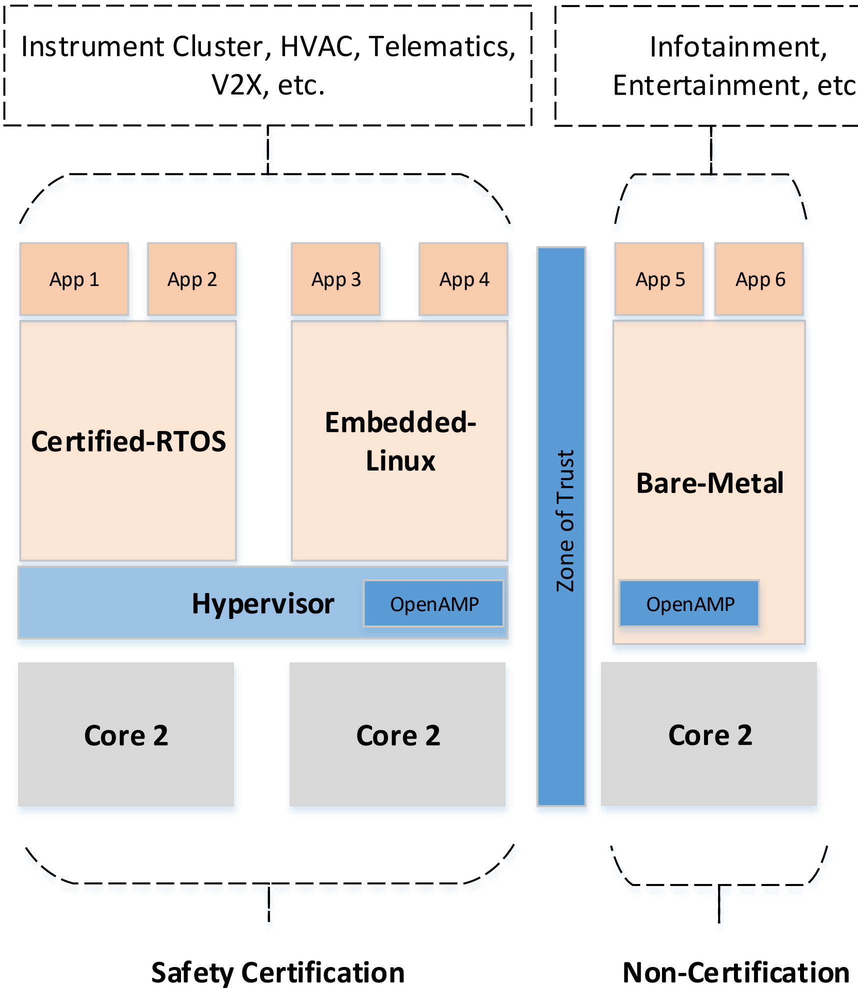

Figure 4. Hybrid design approach in a Digital Cockpit

4. CONCLUSION

The multi-core processor SoCs can deliver an enhanced level of unified peripheral and CPU clusters which enable engineers to lower BOM costs, reduce power consumption and accelerate design implementation. Multicore designs are becoming increasingly common and years to come will be mainstreamed. With the increase of multicore design, we will see rise in usage of OpenAMP. It is worth mentioning that OpenAMP has indeed opened up many new doors to relook at the mixed-safety critical applications. OpenAMP is a platform to stay and provide a solution to implement mixed safety-critical solutions in a robust and cost-effective manner.

ACKNOWLEDGEMENTS

The author would like to thank my organization L&T Technology Services for giving me opportunity to work on OpenAMP. Also like to thank the OpenAMP open source community who is trying to help and contribute to this innovative software framework. OpenAMP as an emerging API standard managed under the umbrella of Multicore Association. This project is jointly maintained by Mentor Graphics, Xilinx and other software and hardware vendors. A current reference implementation of the proposed OpenAMP standard is available at: https://github.com/OpenAMP/open-amp. Mentor Embedded Multicore Framework (MEMF) is a proprietary implementation of the OpenAMP standard.

REFERENCES

[1] OpenAMP GitHub: https://github.com/OpenAMP/open-amp
[2] OpenAMP Project: https://www.openampproject.org/
[3] Mentor Graphics, OpenAMP Framework User Reference.pdf
[4] Xilinx, Libmetal and OpenAMP User Guide
[5] OpenAMP Wiki: http://www.wiki.xilinx.com/OpenAMP
[6] Xilinx OpenAMP source code: https://github.com/Xilinx/open-amp

[7] Embedded Computing Design: Multicore and OpenAMP: https://www.embeddedcomputing.com/technology/processing/compute-modules/multicore-and-openamp

AUTHORS

Mridula Prakash has over 13 years' experience in the electronics industry, largely dedicated to embedded software. She is a senior executive with extensive experience in architecting and building embedded products in the Industrial Products domain. She is an active member in embedded system community and has been working on various technologies of microprocessors and microcontrollers, including x86, PIC, AVR, MIPS, PowerPC and ARM, developing firmware and low-level software in C/C++ on Linux, Android, FreeRTOS and many other kernel and operating systems.

In her current role as Specialist – Embedded Architect at L&T Technology Services (LTTS), she is responsible to understand major trends in the embedded sector and help in the implementation of embedded design software and modernization of legacy systems.