Chapter 16
Choosing the Optimized OS for an MPSoC Embedded System

Abderrazak Jemai
National Institute of Applied Science and Technology, Tunisia

ABSTRACT
This chapter provides a comparative study between recent operating systems, designed for embedded systems. Our study focuses, in particular, on systems designed for Multiprocessors implementations called MPSoC. An OS can be seen as an abstract layer or an interface between the embedded application and the underlying hardware. In this chapter, we give a comparative study of main operating systems used in embedded systems. The originality of this chapter is that we specially focus on the OS ability to be optimized to support and manage a multiprocessor architecture. A multiprocessor system-on-chip is software driven and mastering the development complexity of the software part of MPSoC is the key to reduce developing time factor. This opportunity could be reached through the use of a document giving a detailed description and analysis for criteria related to MPSoC. The wide diversity of existing operating systems, the huge complexity to develop an application specific or a general purpose, and the aggressive evolution of embedded systems makes the development of such a system a so difficult task. These considerations lead to the realization that a work that provides guidance for the MPSoC designers will be very beneficial for these communities.

INTRODUCTION
This chapter presents the basic services that an embedded OS should provide and gives a comparative study between recent operating systems designed for embedded systems. Our study focuses, in particular, on those designed for Multiprocessors implementations called MPSoC. The originality of this work is that we specially focus on the OS ability to be optimized to support and manage a multiprocessor architecture.

A multiprocessor system-on-chip is software driven and mastering the development complexity of the software part of MPSoC is the key to reduce developing time factor. This opportunity could be reached through the use of a document giving a detailed description and analysis for criteria related to MPSoC. The wide diversity of existing operating systems, the huge complexity to develop an application specific or a general purpose, and the aggressive evolution of embedded systems makes the development of such a system a so difficult task. These considerations lead to the realization that a work that provides guidance for the MPSoC designers will be very beneficial for these communities.
Choosing the Optimized OS for an MPSoC Embedded System
giving a detailed description and analysis for criteria related to MPSoC.

Basically, embedded system developers for MPSoC, start by defining the specification of their application and then turn to OS available on the market. If one of them gives good response to their application requirements, the selected OS will be chosen. Otherwise, developers may opt to develop a specific one. The hard work, in the first case, is to find a detailed and objective comparative study between existing OS to decide which one suits their application needs. Many studies may be found giving comparison between existing OS, but, the comparison should be based on today available OS and take into account new application requirements (QoS, Performance, energy consumption, memory footprint, parallel programming, multiprocessor management, etc.).

If no existing OS suits the application requirements, designers may opt to develop their own OS. In this case, the developed OS will contain the only needed services. This chapter details basic services that a minimal OS should have.

This chapter is structured as follows: Section II provides a number of definitions and descriptions that will help the reader to better understand the rest of this chapter. Section III presents the OS structure, Section IV presents the OS functions and services and section V presents a survey of existing operating systems that target embedded systems.

**EMBEDDED SYSTEMS, RTOS AND SOC**

Through this chapter, some concepts such as embedded systems, real-time systems, SoC, recur very frequently. It is useful to provide a number of definitions and descriptions that will help the reader to better understand the rest of this chapter.

A System (Hardware and Software) is considered embedded if it is a sub-component of a larger system and is used to receive events from and monitor that system, using special hardware devices.

A system is said to be on Chip if an entire embedded system is integrated in the same chip. These on chip systems (SoC: System on a Chip) need a specific design flow taking into account the limited space constraints, and the high speed transmission context. With classic systems, the hardware is already designed when designing the software part. In contrast, in SoC, the two parts are often designed in parallel in order to choose the best hardware function fitting the software requirements.

A real-time system is a system whose performance depends not only on the values of its outputs, but also on the time at which these values are produced. A real-time system can be defined also as one in which the correctness of the computations depends not only on the logical correctness of the computation but also on the time at which the result is produced. If the timing constraints of the system are not met, system failure is said to have occurred. The most important feature for a real-time system is not speed, but how much the individual timing constraints of each task are satisfied. We can distinguish hard real-time and soft real-time applications. In hard real time, there is no value of computation if it is late. The activities of the system must be completed on time. A soft real-time system can tolerate some late answers. It is important to note that in classic applications, we don’t account for how late a computation is completed. A real-time operating system (RTOS) is an operating system able to provide a required service in a bounded response time.

**OS STRUCTURE**

An embedded OS is composed of five main layers: OS-API, Basic OS, Communication Layer, HALL-API and HALL.
Related Content

A Tool Support for Secure Software Integration
[www.igi-global.com/article/tool-support-secure-software-integration/43925?camid=4v1a](www.igi-global.com/article/tool-support-secure-software-integration/43925?camid=4v1a)

A Study on the Intention to Adopt Third Generation (3G) Wireless Service on a Small Community with Unique Culture: The Use of Hofstede Cultural Dimensions in Predicting the Interaction between Culture and the Technology Acceptance Model on Guam
[www.igi-global.com/article/a-study-on-the-intention-to-adopt-third-generation-3g-wireless-service-on-a-small-community-with-unique-culture/89388?camid=4v1a](www.igi-global.com/article/a-study-on-the-intention-to-adopt-third-generation-3g-wireless-service-on-a-small-community-with-unique-culture/89388?camid=4v1a)

An Adaptive System for Retrieval and Composition of Learning Objects
[www.igi-global.com/article/adaptive-system-retrieval-composition-learning/61315?camid=4v1a](www.igi-global.com/article/adaptive-system-retrieval-composition-learning/61315?camid=4v1a)

Emerging Real-Time Methodologies
[www.igi-global.com/chapter/emerging-real-time-methodologies/76955?camid=4v1a](www.igi-global.com/chapter/emerging-real-time-methodologies/76955?camid=4v1a)