3 MRAPI Overview

The major MRAPI concepts are covered in the following sections. The concepts and supporting datatypes are defined to meet the goals stated in Section 2.1, including source code portability.

3.1 Definitions

timely - operation will return without having to block on any IPC to any remote nodes.
IPC – Inter-processor communication.
MCAPI – Multicore Communications API: communication standard defined by Multicore Association.
MRAPI – Multicore Resource API: resource management standard defined by Multicore Association in this document.
MTAPI – Multicore Task API: task management standard to be defined by Multicore Association.

3.2 MRAPI Domain

An MRAPI system is composed of one or more MRAPI *domains*. An MRAPI domain is a unique system global entity. Each MRAPI domain comprises a set of MRAPI *nodes* (Section 3.3). An MRAPI node may only belong to one MRAPI domain, while an MRAPI domain may contain one or more MRAPI nodes. The concept of a domain is shared amongst Multicore Association APIs and must be consistent (i) within any implementation that supports multiple APIs, and (ii) across implementations that require interoperability.

3.3 MRAPI Nodes

An MRAPI node is a process, a thread, or a processor. In other words, an MRAPI node is an independent thread of control. The working group explicitly chose to leave the definition of a node flexible in this regard because it allows applications to be written in the most portable fashion that can be allowed by underlying embedded SoC multicore hardware, while at the same time supporting more general-purpose multicore and manycore devices. Therefore, it is the decision of the working group that overly-specifying the definition of a node would unnecessarily restrict the degrees of freedom that are allowed for mapping functionality to the resources on an embedded multicore SoC. Such systems may include processor cores, hardware accelerators, and various virtualization technologies. The MRAPI working group believes that this definition of a node allows portability of software to be maintained at the interface level (e.g., the functional interface between nodes). However, the software implementation of a particular node cannot (and often should not) necessarily be preserved across a multicore SoC product line (or across product lines from different silicon providers) because a given node's functionality may be provided in different ways -- depending on the chosen multicore SoC.

The `mrapi_initialize()` call takes node number and domain number arguments, and an MRAPI application may only call `mrapi_initialize()` once per node. It is an error to call `mrapi_initialize()` multiple times from a given thread of control unless `mrapi_finalize()` is called between such calls. A given MRAPI implementation will specify what is a node (i.e., what thread of control - process, thread, or other -- is a node) for that implementation. A thread and process are just two examples of threads of control, and there could be others.

The concept of nodes in MRAPI is shared with other Multicore Association (MCA) API specifications. Therefore, implementations that support multiple MCA APIs must define a node in exactly the same way, and initialization of nodes across these APIs must be consistent. In the future the Multicore