3 MCAPI Overview

The major MCAPI concepts are covered in the following sections.

Editors note: Request that technical writer includes MCAPI header file information such as attributes and other function relevant information directly in the function sections.

Also section references have to be validated]

3.1 MCAPI Domains

An MCAPI domain is comprised of one or more MCAPI nodes in a multicore topology and used for routing purposes. The scope of a domain is implementation defined and its scope could for example be a single chip with multiple cores or multiple processor chips on a board. The domain id is specified once at node initialization. A domain can contain multiple nodes. Some example potential uses for domains are topologies that may change dynamically, include non-MCAPI sub-topologies, require separation between different transports or have open and secure areas.

3.2 MCAPI Nodes

An MCAPI node is a logical abstraction that can be mapped to many entities, including but not limited to: a process, a thread, an instance of an operating system, a hardware accelerator, or a processor core. The node id is specified at the call to `mcapi_initialize()`.A given MCAPI 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 MCAPI standard aims to be implementable on both process-oriented and thread-oriented systems. MCAPI defines a transport layer between 'nodes', which could be processes, threads, or some other thread-of-control abstraction. The standard explicitly avoids having any dependence on a particular shared memory model or process protection model. The communication operations defined by MCAPI should be implementable with identical semantics on either thread- or process- based systems. Thus, programs that use only the MCAPI APIs should be portable between MCAPI implementations. On the other hand, programs that make use of APIs outside of MCAPI, for example pthreads, will only be portable between systems that include those extra APIs.

THINGS TO REMEMBER:

- The MCAPI domain and node numbering plan is established when the MCAPI communication topology is configured at design-time, so programmers don't have to worry about this at run-time.
- It is up to the MCAPI implementation to configure communication topology interfaces to enable communication with other nodes in the communication topology.

3.3 MCAPI Endpoints

MCAPI endpoints are socket-like communication termination points. MCAPI endpoints are created with the mcapi_endpoint_create() function. An MCAPI node can have multiple endpoints. Thus, an MCAPI endpoint is a topology-global unique identifier. Endpoints are identified by a <`domain_id, node_id, port_id`> tuple. An endpoint is created by implicitly referring to the node_id of the node on which the endpoint is being created, and explicitly specifying a port_id. Alternatively, MCAPI allows the creation of an endpoint, by requesting the next available endpoint. . Static names allow the programmer to

Comment [ML1]: Do you want to include a Definitions section similar to what is found in MRAPI Overview section 3.1?

Comment [S2]: yes, makes sense.

Comment [S3]: Some of the functions have attributes that are listed in the header files. They should also be included in the function description, similar to what we have in the MRAPI spec.

Comment [S4]: For tech editor.