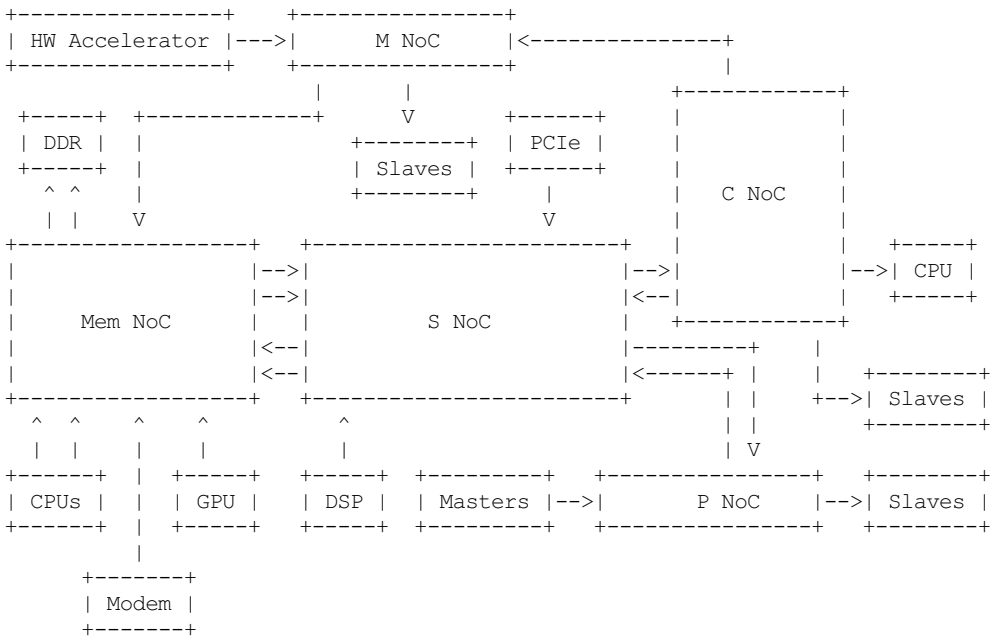
Generic System Interconnect Subsystem

Introduction

This framework is designed to provide a standard kernel interface to control the settings of the interconnects on an SoC. These settings can be throughput, latency and priority between multiple interconnected devices or functional blocks. This can be controlled dynamically in order to save power or provide maximum performance.

The interconnect bus is hardware with configurable parameters, which can be set on a data path according to the requests received from various drivers. An example of interconnect buses are the interconnects between various components or functional blocks in chipsets. There can be multiple interconnects on an SoC that can be multi-tiered.

Below is a simplified diagram of a real-world SoC interconnect bus topology.



Terminology

Interconnect provider is the software definition of the interconnect hardware. The interconnect providers on the above diagram are M NoC, S NoC, C NoC, P NoC and Mem NoC.

Interconnect node is the software definition of the interconnect hardware port. Each interconnect provider consists of multiple interconnect nodes, which are connected to other SoC components including other interconnect providers. The point on the diagram where the CPUs connect to the memory is called an interconnect node, which belongs to the Mem NoC interconnect provider.

Interconnect endpoints are the first or the last element of the path. Every endpoint is a node, but not every node is an endpoint.

Interconnect path is everything between two endpoints including all the nodes that have to be traversed to reach from a source to destination node. It may include multiple master-slave pairs across several interconnect providers.

Interconnect consumers are the entities which make use of the data paths exposed by the providers. The consumers send requests to providers requesting various throughput, latency and priority. Usually the consumers are device drivers, that send request based on their needs. An example for a consumer is a video decoder that supports various formats and image sizes.

Interconnect providers

Interconnect provider is an entity that implements methods to initialize and configure interconnect bus hardware. The interconnect provider drivers should be registered with the interconnect provider core.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\(linux-master)(Documentation)(driver-api)interconnect.rst, line 85)

Unknown directive type "kernel-doc".

```
.. kernel-doc:: include/linux/interconnect-provider.h
```

Interconnect consumers

Interconnect consumers are the clients which use the interconnect APIs to get paths between endpoints and set their bandwidth/latency/QoS requirements for these interconnect paths. These interfaces are not currently documented.

Interconnect debugfs interfaces

Like several other subsystems interconnect will create some files for debugging and introspection. Files in debugfs are not considered ABI so application software shouldn't rely on format details change between kernel versions.

`/sys/kernel/debug/interconnect/interconnect_summary`:

Show all interconnect nodes in the system with their aggregated bandwidth request. Indented under each node show bandwidth requests from each device.

`/sys/kernel/debug/interconnect/interconnect_graph`:

Show the interconnect graph in the graphviz dot format. It shows all interconnect nodes and links in the system and groups together nodes from the same provider as subgraphs. The format is human-readable and can also be piped through dot to generate diagrams in many graphical formats:

```
$ cat /sys/kernel/debug/interconnect/interconnect_graph | \
        dot -Tsvg > interconnect_graph.svg
```