ARGO

WCET-Aware Parallelization of Model-Based Applications
for Heterogeneous Parallel Systems

H2020-ICT-2015

Project Number: 688131

Deliverable D4.1

D4.1 Specification of a WCET-aware abstract architecture description

Editors:	Simon Reder
Authors:	Simon Reder, Reinhold Heckmann, Steven Derrien, Timon ter Braak, Leonard Masing, Georgios Keramidas
Version:	1.10
Status:	FINAL
Dissemination level:	Public (PU)
Filename:	D4.1_ARGO_ADL_1v10.docx

ARGO Consortium

Karlsruhe Institute of Technology	DE
Scilab Enterprises	FR
Recore Systems B.V.	NL
Université de Rennes I	FR
Technological Educational Institute of Western Greece	GR
AbsInt Angewandte Informatik GmbH	DE
Deutsches Zentrum für Luft- und Raumfahrt	DE
Fraunhofer IIS	DE

Document Revision History

Version	Based on	Date	Author	Comments / Changes
v1.00		30.06.2016	Simon Reder	Final Version
v1.10	v1.00	06.12.2016	Simon Reder	Revised version addressing the review comments

About this Document

This document specifies the first version of the ARGO ADL which is used to provide detailed hardware information to the ARGO tool-chain. The specification is based on the SHIM 1.0 ADL [1] which is extended to meet the ARGO specific requirements. The ADL specification is complemented by guidelines defining the level of detail and description styles that are required to be compatible with the ARGO tools. Examples of using the revised ADL for the two ARGO platforms (i.e., FlexaWare and InvaIC) are provided.

Table of Contents

1. Introduction

In order to generate highly efficient program code especially for multi- and many-core architectures, the presence of sufficiently detailed hardware information is inevitable. This is particularly true, when real-time constraints with the need for Worst Case Execution Time (WCET) analyzability come into play. To statically estimate the WCET for a given program, detailed hardware timing information must be available. In case of multi-core processors this includes details on the communication busses and/or networks connecting the processor cores, memories and peripherals.

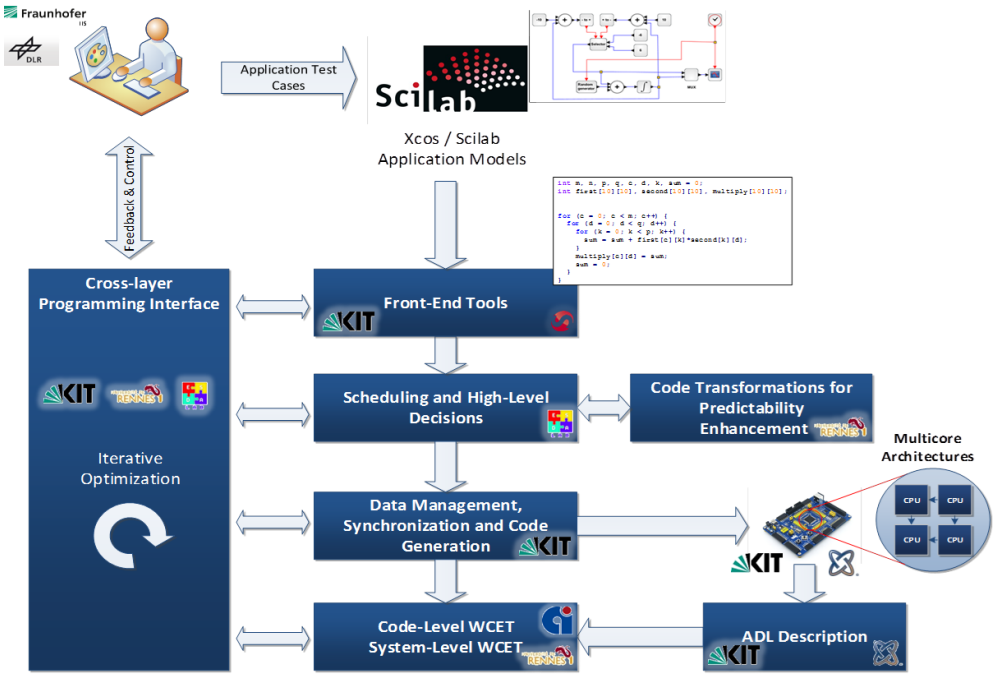

Figure 1: The ARGO Tool-Flow

In ARGO an Architecture Description Language (ADL) will be used to provide this hardware information to the toolchain (see also Figure 1). Within the toolchain, mainly the WCET analysis tools rely on the ADL description. Additionally, the scheduling and code-generation tools use the hardware information from the ADL description in order to generate more efficient parallel code for the target platform.

This document specifies the ADL which will be used in ARGO. The specification is based on an analysis of the hardware information required by the different tools. The requirements then are matched with a state-of-the-art ADL in order to identify and define potential extensions that are necessary for the ARGO approach.

The remainder of this document is structured as follows: Section 2 describes the requirements for the ADL with respect to the different toolchain components that require information from the hardware layer. Section 3 specifies which characteristics of the two ARGO target platforms are relevant for ADL descriptions. Section 4 defines which state-of-the-art ADL will be used in ARGO and specifies the ADL extensions which will add all necessary ARGO specific information. Section 5 shows how the extended ADL is used to

describe the ARGO target platforms. In Section 6, the integration of the ADL into the ARGO toolchain is described. Finally, Section 7 concludes the document.

2. Requirements for the Architecture Description Language

This section elaborates the requirements for the ARGO ADL with respect to the different toolchain components that take the ADL as input.

2.1 Required Hardware-Information for Code-Level WCET analysis

This section describes which hardware-specific information should be provided by an ADL description for performing code-level WCET analysis with AbsInt's aiT tool.

The aiT tool can be used to automatically compute tight upper bounds for the worst-case execution times of tasks in real-time systems. The tool performs static program analysis based on abstract interpretation. The analysis results hold for all inputs and execution scenarios.

aiT analyzes binary executables for specific processor architectures and takes the timing behaviour of the hardware into account, in particular the cache and pipeline behaviour and the memory latency.

There are aiT versions for various major processor architectures, including MicroBlaze and Leon3. An aiT for Xentium is under construction. Part of the hardware configuration is built into these specialized aiT versions in a fixed way, while other parts have to be given as parameters to the tool, in a mixture of GUI options and textual descriptions in an annotation language. The exact distribution between these possibilities depends on the target architecture. We try to abstract from these differences and to describe the required information in a high-level way.

2.1.1 Caches

Caches are used to improve the access times of fast microprocessors to relatively slow main memories. The number of caches included in the memory hierarchy as well their configurations and topology varies. There may be a hierarchy of different cache levels L1, L2, ... For L1 caches, common cases are

- There is no cache at all;
- There is a single cache, which is used only for instructions;
- There is a single "unified" cache for data and instructions;
- There are two different caches, one for instructions and one for data.

Sometimes, there are additional small cache-like buffers for code and/or data accesses.

What is needed in the hardware description is a list of all available caches that are actually in use as caches, i.e. not disabled and not completely locked. For each cache, its type (instruction, data, or unified) and its internal organization have to be specified.

There are three commonly used cache architectures: direct-mapped caches, fully associative caches, and *A*-way set-associative caches (where *A* is a natural number).

An *A*-way set-associative cache consists of *S* cache sets. Each cache set consists of *A* ways or lines, where the number *A* denotes the associativity of the cache. Each cache line can hold the copy of a memory block consisting of *B* consecutive bytes. Hence the total capacity

of the cache is $S \times A$ memory blocks, or $S \times A \times B$ bytes. Usually, the numbers S, A, and B are powers of 2.

The other two cache architectures can be considered as degenerate special cases of A-way set-associativity: direct-mapped caches correspond to the case $A = 1$ (each set has only one line), and fully associative caches to the case $S = 1$ (there is only one cache set).

Often, it is possible to lock some or all ways of a cache, which means that these ways correspond to fixed pieces of memory; effectively, the locked part of a cache behaves like a scratch pad. Therefore, only the unlocked part of the cache is considered as cache memory for the purposes of WCET analysis.

Thus, the cache architecture can be described by the following parameters:

- The size of the unlocked part of the cache in bytes;
- The line size in bytes;
- The number of unlocked ways in the cache;
- The replacement strategy.

Cache analysis has been implemented for the following three replacement strategies:

- FIFO first-in first-out cache replacement strategy,
- LRU least-recently used strategy,
- PLRU a special replacement strategy called Pseudo-LRU.

2.1.2 Memory Map

Timing analysis needs to know about the existing memory types and their properties, in particular the access times. Therefore there should be a list of memory areas, and for each memory area the following information:

- The size of the area, specified by both start address and end address, or by start address and width of the area, i.e. number of memory cells.
- The fact that the area contains only code or only data.
- The access properties stating whether the area is readable and whether it is writable. An area may be non-writable because it resides in a physical ROM area, or because it contains code and static data that is never modified.
- The property whether the area is volatile. Values stored in volatile memory areas may change at any time without being written by the analyzed program. Typical examples are memory areas to which I/O devices are mapped, and areas holding variables used for synchronizing different threads. Such areas have the property that values written there by the analyzed program may be overwritten by some external process. Therefore, the analyzer should not assume anything about the result of reading from a volatile area even if it knows what has been written into the area before.
- The "guarded" property means that the area does not admit speculative loads.
- The property whether the area is cached and/or locked. Locked means that the area has been preloaded into the cache and locked there.
- The memory type, which is something like SRAM, SDRAM, PROM, PCI. Each processor architecture and hence each aiT version comes with different supported memory types.
- For a few architectures, it is important to define the number of the chip select signal to which the area is mapped.
- The port width of the memory, e.g. 16 bit or 32 bit.

- The information whether accesses larger than the port width are allowed. Such accesses are split into several smaller accesses by the memory controller.
- The information whether misaligned accesses are allowed. If allowed, misaligned accesses often take longer than aligned accesses.
- The information whether the area supports burst accesses. Burst accesses consist of a sequence of so-called beats that access consecutive memory cells.
- The burst length, i.e. the maximum number of words that may comprise a burst access for this memory region.
- The access time(s) of the memory area. There may be different times for code accesses, data reads, and data writes. The access time for a single-beat access is a single number, whereas the access time for a burst access is a sequence of numbers, one for every beat of the burst.
- The clock ratio between the bus clock and the CPU clock, which gives the factor by which the afore-mentioned access times have to be multiplied to obtain the number of CPU cycles needed for the access.

For flash memory, more information may be needed, e.g. whether the flash supports prefetching or pipelining.

2.1.3 Pipeline

The properties of the pipeline are usually built-in into an aiT version for a special target architecture. Yet sometimes, a few parameters can be set. In case of the Leon3 architecture for instance, it is possible to specify in the GUI which multiplier and which floating-point units are used, and whether multiply-accumulate instructions are enabled.

On the other hand, when a new aiT version is developed, the characteristics of the pipeline have to be known: the number and kind of the functional units, the data flow and the characteristics of buffers and queues between them, the handling of branches w.r.t. prefetching and speculation, the existence and kind of branch target buffers, etc.

2.2 Required Hardware-Information for System-Level WCET analysis

This section describes which hardware specific information shall be provided by an ADL description in order to allow a proper system-level WCET analysis. System-level WCET analyzers operate by gathering and combining the results of code-level WCET estimation in a *safe* and as *tight* as possible manner. Code-level WCET estimation produces the WCET of a sequential code snippet, running on a single core, ignoring the indirect interferences from code running on the other cores.

The difficulty of system-level WCET estimation lays in the presence of shared hardware resources between cores, which introduce a dependency between otherwise independent code snippets running on different cores. Shared hardware resources can include caches, on-chip memory scratchpad, off-chip memory, and bus and/or Network-on-a chip interconnect.

The platform description must therefore indicate for each storage/interconnect component whether it can be shared or not. Similarly, the description should also specify which processing/communication component has access to that shared resource. In the following we will refer to this information as the *structural description* of shared resources.

This structural data is however not enough for obtaining a system level WCET estimate, as it does not provide any timing information that could be used to derive the WCET. The description must therefore be enriched with a description of the shared resource temporal behavior. This temporal behavior model must provide enough information for deriving a safe and tight system level WCET.

Each shared resource is then characterized by a list of triplets, where the first element of the triplet specifies the access type (**read**, **write**), the second element specifies the ***access context*** (exclusive or shared), and the third one provides the information needed to derive the resource worst case access time. The latter information may be expressed differently depending on the access context and on the modeled resource, as shown in the Table below.

Access context	Worst case access time (in cycles)
Exclusive: during a given execution stage S_j, the resource R_i is accessed by at most one component C_k.	Constant value (expressed as a number of clock cycles)
n-Shared : during a given execution stage S_j, the resource R_i is accessed by at most *n* components (that we write as the set $C_{i,j}$).	Constant value or closed form formula taking the value of *n* as argument.

2.3 Required Hardware-Information for Data Management and Code Generation

The Data Management and Code Generation step of the ARGO toolchain generates code that is optimized for the target hardware/software platform. In doing so, the toolchain step relies on the ADL description of the platform to determine the platform API, elaborate data partitioning and make fine-grained parallelization decisions. The resulting requirements for the ADL can be divided into 3 different areas:

- Information to determine the general platform type and the available API(s)
- Parameters describing the platform topology including relevant memory segments with the associated address spaces
- Information on the instruction set including relevant ISA extensions (e.g. SIMD instructions)

2.3.1 Platform Type and API

In order to generate correct code, information on the general platform type, the platform compiler toolsets as well as the platform API must be available. The code generation tool will support generating C code for a set of platform types which particularly includes the two platforms FlexaWare and InvasIC that are used to evaluate the ARGO approach (see also Section 3). Furthermore a set of different platform API levels might be supported for the platform types in order to enable different ways of inter-processor communication. Depending on the available hardware resources, the platform API might e.g. provide message passing and/or shared-memory management support.

In order to derive this information from an ADL description, the ARGO ADL should include the following attributes:

- The general platform type (e.g. the FlexaWare or InvasIC platform)
- The types of the processing resources to be used. This is necessary to determine target compilers and the relevant processor features. Since ARGO also aims for heterogeneous platforms, the individual processing resources may have different types.
- Description of inter-processor communication features supported by the hardware (e.g. FIFO queues, shared-memory or interrupt-based communication). These hardware features are required to determine the set of communication primitives supported by the platform API.

2.3.2 Description of the Platform Topology

For efficient data management, the code generation tool requires basic information on the platform topology. This includes information on the available peripherals, interconnections and the memory hierarchy. To access the memory segments in the generated code, the address spaces with respect to the accessing processing resources are of particular interest.

In detail, the following information should be provided by the ADL description:

- A list of available memory segments (e.g. off-chip DRAM, on-chip scratchpad memories, …) and peripherals
- The address spaces that are used to access the memories/peripherals from a specific processing resource
- Parameters stating if and with which mode (read, write or both) a processing resource can access a memory segment

2.3.3 ISA Description

During the code generation step, special features and extensions of the instruction set architecture (ISA) can be used for processor specific optimization and fine-grained parallelization (e.g. using SIMD instructions). To support this, the ADL should contain the following details:

- Instruction set family (e.g. ARM, MicroBlaze, etc.) of each processing resource
- Information which subset/variant of the ISA is implemented
- Supported ISA extensions for each processor-core. SIMD extensions are of particular interest for the code generation.

3. ARGO Target Platform Characteristics Description

3.1 Recore FlexaWare Characteristics

The FlexaWare architecture is a non-uniform memory access (NUMA) system, composed of multiple subsystems (clusters) spread over multiple chips, hosted on an aggregation of boards. These boards, chips and subsystems are connected with various technologies and speeds, resulting in a heterogeneous interconnect. The interconnect has a memory mapped interface that is mostly transparently in use. This aspect of the architecture not only results in non-uniform latency of (memory) requests, but also in non-uniform accessibility of locations in the platform. Resources provided by the platform are considered to be accessible at the level of the core, the subsystem, the board, and globally.

Figure 2 shows an overview of the FW400 system, which is a specific instantiation of the FlexaWare architecture. Each subsystem consists of some common infrastructure components that are used to manage the platform, to provide a user interface and to control the processing elements (orange), which make up the remainder of the subsystem. The amount and type of processing elements is configurable within some restrictions set by the specific FlexaWare product. In the default configuration, the processing elements are made up of reduced instruction set cores (RISC) with limited instruction cache and no data cache. The subsystems provide a shared scratchpad memory, and larger memories (DDR) can be accessed at the board-level (memory is shown in pink color).

The main user interface to the processing fabric is made up of a dual ARMv7 core subsystem (left board in Figure 2) running an off-the-shelf Linux environment. This subsystem can be (partly) used for processing, managing I/O streams, and for controlling the active set of applications. In Figure 2, the rightmost board hosts the I/O peripherals that should be used to stream input data into the platform.

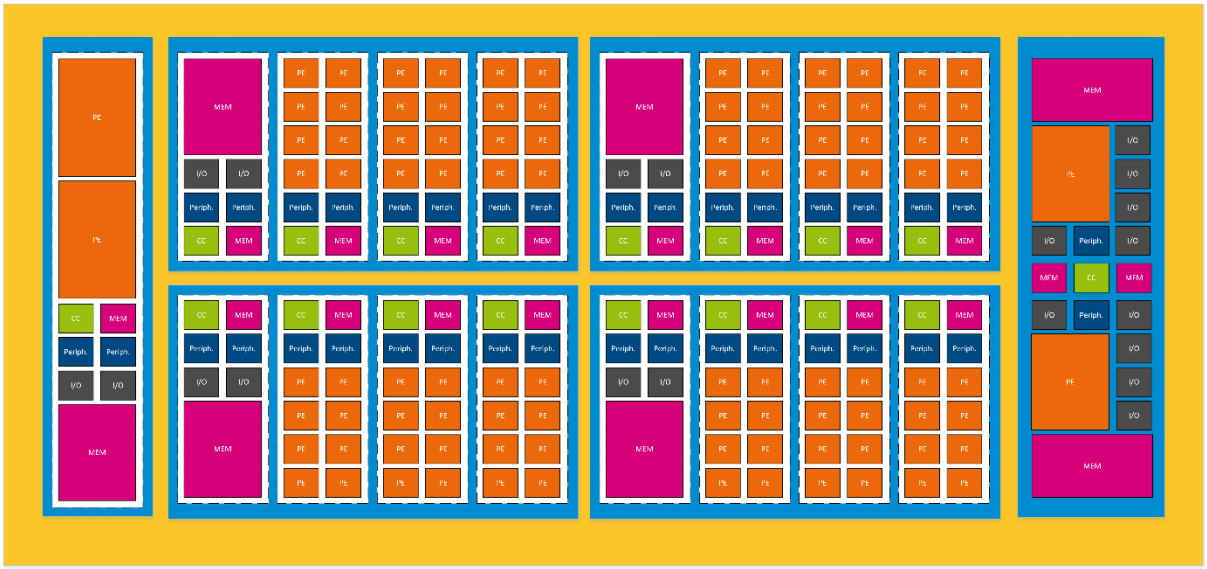

Figure 2: Architecture overview of the FlexaWare FW400 system.

The most interesting characteristic of the FlexaWare architecture is the presence of a middleware layer. The layer takes the responsibility of managing the resources and abstracting from the hardware in order to adhere to the configuration restrictions of the hardware and to facilitate a more convenient interface to the processing platform. One of the

challenges is to find in what degree this layer should be leveraged and to what extent the ARGO-generated configuration should contain the details of the required hardware configuration. The ADL should then either describe the resources available on a bare-metal system, or should describe the resource availability left for ARGO, once the middleware is up-and-running.

3.2 InvasIC Architecture Characteristics

The InvasIC architecture is a tiled architecture that consists of two major parts, namely the tiles and the global interconnect infrastructure.

Each tile is typically a multicore system in itself that is connected via a shared bus. The tiles are based on the Gaisler IP library and feature a number of LEON3 cores as well as several peripherals available from the GRLIB.

The characteristics of the architecture that are available for an ADL are based on the configuration options available in the GRLIB. Specifically, the LEON cores and the AMBA bus can be configured in various ways according to the data sheets, including for example the associativity of the L1 cache. Besides configuring each LEON core itself, the number of cores in each tile can be varied up to 5 cores.

A number of peripherals are connected to the local AMBA bus. These may be scrapped or added as needed but since they are typically only slaves on the bus they do not impact the arbitration.

The global interconnect is represented by a scalable network-on-chip (NoC) featuring best-effort traffic and guaranteed service connections. The NoC consists of the network adapter (NA) that acts as the interface to the tiles and the routers, which are connected to form a meshed network. Remote data is cached by a level 2 cache, which can be configured to cache only certain parts of the memory.

The NoC routers are configurable in several aspects, e.g. data width, amount of virtual channels, buffer depth and amount of guaranteed service connections. The NoC traffic is deterministic yet typically the best-effort traffic is very difficult to predict as soon as contention comes into play. The guaranteed service on the other hand is trivially predictable as long as a connection can be established. Furthermore, a service level can be assigned to each guaranteed service connection. The arbitration of the link is split into time slots that are assigned to a virtual channel and thus, to a connection. This allows assignment of a higher fraction of the bandwidth than usual to a guaranteed service connection.

The setup of such a guaranteed service connection requires a free virtual channel in each router, otherwise it fails. If the communication is known in advance, a model for establishing the guaranteed service connections may be developed.

4. ADL Specification

In this section, the ADL for ARGO is specified based on a state-of-the-art ADL which will be extended to fit the needs of the ARGO tools. First, the selection of SHIM as basis for the ARGO ADL is motivated. Then, the specification of the ARGO extensions is described followed by guidelines for ARGO-compatible ADL descriptions.

4.1 Selection of the extendable State-of-the-art ADL

Besides the ability to provide the information elaborated in the requirements described in Section 2, the state-of-the-art ADL building the basis for the ARGO ADL should meet further criteria:

- The ADL should be well-suited for describing heterogeneous multi- and many core processor systems
- The ADL should be easy to extend
- Focus on the hardware resources and interconnects (the description of software modules or of peripherals like sensors/actors is not necessary)
- Easy to use (easy implementation of a parser and an in-memory representation of the hardware information)

There is a wide variety of ADLs with different use cases and complexities. The Architecture Analysis & Design Language (AADL) [2] for example is a powerful ADL capable of describing application software, processor platforms and whole embedded systems including distributed bus systems and peripherals like sensors and actors. For ARGO however the description of software and distributed systems is not necessary and the high complexity of the AADL could prevent an easy integration into the toolchain.

One candidate that better matches the criteria mentioned above is the “Software-Hardware Interface for Multi-Many-Core” (SHIM) [1] ADL which has been developed by the Multicore Association [3]. SHIM is explicitly designed to describe multi-core processor systems and does not bother with the description of software components. It is based on the widely used “Extensible Markup Language” (XML) [4] specification which makes it easy to extend. XML documents are defined by a so called XML schema defining the allowed structure of the XML hierarchy as well as possible attributes. XML schemas are typically described by XSD (XML Schema Definition) files which are XML documents themselves. An extension of SHIM can simply be done by extending the SHIM XSD with custom attributes, elements and types. The XML basis also allows the use of standard XML parsers together with the XSD schema to parse SHIM ADL descriptions. An in-memory IR can be automatically generated using code generators which generate object-oriented class structures based on XSD schemas. Such tools are e.g. available for the Java [5] and C++ programming languages. Most of the information demanded in Section 2 can already be described using SHIM which on the same time keeps the required ARGO specific extensions minimal.

These advantages make SHIM a very well-suited candidate to build the ARGO ADL on top of it. The SHIM 1.0 specification therefore will be used as basis for the ARGO extensions described in the following section.

4.2 Specification of the ARGO Extensions to SHIM 1.0

This section specifies the ARGO extensions for the SHIM ADL. The SHIM 1.0 specification is only extended in areas, where the available attributes are not sufficient to meet the

requirements described in Section 2. The SHIM extensions being made are enumerated and shortly motivated in the following subsections. Guidelines on how a whole platform should be described to provide all necessary information for the ARGO toolchain can be found in Section 4.3.

4.2.1 New Attribute for the Cache Replacement Strategy

In order to take caches into account during the code-level WCET analysis, it is necessary to know the cache replacement strategy (see Section 2.1.1). Since SHIM does not include this information, a new attribute named "*replacementStrategy*" is introduced for the "*Cache*" element. The attribute is of type "*CacheReplacementStrat*" which is a string type constrained to the values *"FIFO"*, *"LRU"* and *"PLRU"*. In the XML schema, this change appears as follows:

```
<xs:complexType name="Cache">
  <xs:sequence>
    <xs:element name="cacheRef" type="xs:IDREF"
                minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:attribute name="name" use="required" type="xs:string"/>
  <xs:attribute name="id" use="required" type="xs:ID"/>
  <xs:attribute name="cacheType" use="required" type="CacheType">
    <xs:annotation>
      <xs:documentation>soft / hard</xs:documentation>
    </xs:annotation>
  </xs:attribute>
  <xs:attribute name="cacheCoherency" use="required" type="CacheCoherencyType"/>
  <xs:attribute name="size" use="required" type="xs:int"/>
  <xs:attribute name="sizeUnit" use="required" type="SizeUnitType"/>
  <xs:attribute name="nWay" use="optional" type="xs:int"/>
  <xs:attribute name="lineSize" use="optional" type="xs:int"/>
  <xs:attribute name="lockDownType" use="optional" type="LockDownType"/>
  <xs:attribute name="replacementStrategy" use="optional"
                type="CacheReplacementStrat"/>
</xs:complexType>

<!-- ... -->

<xs:simpleType name="CacheReplacementStrat">
  <xs:restriction base="xs:string">
    <xs:enumeration value="FIFO"/>
    <xs:enumeration value="LRU"/>
    <xs:enumeration value="PLRU"/>
  </xs:restriction>
</xs:simpleType>

<!-- ... -->
```

4.2.2 Average Case Performance Attributes are optional

SHIM includes the *"Performance"* element which is capable of describing the best case, average case and worst case performance for latencies as well as pitches. Since ARGO is dealing with real-time systems, mainly the worst case performance is of particular interest. SHIM however defines the average case performance attribute as required while the worst case performance is optional. In ARGO this is changed in the way, that the average case timing attribute now is optional as well, which makes it possible to only specify a worst case.

The worst case attribute however remains optional in order to maintain compatibility to plain SHIM.

The XSD schema is changed as follows:

```
<xs:complexType name="AbstractPerformance" abstract="true">
  <xs:sequence/>
  <xs:attribute name="best" use="optional" type="xs:float"/>
  <xs:attribute name="typical" use="optional" type="xs:float"/>
  <xs:attribute name="worst" use="optional" type="xs:float"/>

</xs:complexType>
```

4.2.3 Instruction Only and Data Only Memory Segments

In SHIM, caches have an attribute (*cacheType*) stating if the cache can hold only data, only instructions or both content types. For memory address spaces however no differentiation of the content type can be expressed in SHIM. In ARGO this information is required for code-level WCET (see Section 2.1.2), so the *"SubSpace"* type in SHIM is extended with the optional attribute *"contentType"* of the new type *"AddrSpaceContentType"*. The type has the possible values "DATA", "INSTRUCTION", "UNIFIED" and the attribute defaults to "UNIFIED". The SHIM schema is extended as follows:

```
<xs:complexType name="SubSpace">
  <xs:sequence>
    <xs:element name="MemoryConsistencyModel" type="MemoryConsistencyModel"
                minOccurs="0" maxOccurs="unbounded"/>
    <xs:element name="MasterSlaveBindingSet" type="MasterSlaveBindingSet"
                minOccurs="0" maxOccurs="1"/>
  </xs:sequence>
  <xs:attribute name="name" use="required" type="xs:string"/>
  <xs:attribute name="id" use="required" type="xs:ID"/>
  <xs:attribute name="start" use="required" type="xs:long"/>
  <xs:attribute name="end" use="required" type="xs:long"/>
  <xs:attribute name="endian" use="optional" type="EndianType"/>
  <xs:attribute name="contentType" use="optional"
                type="AddrSpaceContentType" default="UNIFIED" />
</xs:complexType>

<!-- ... -->

<xs:simpleType name="AddrSpaceContentType">
  <xs:restriction base="xs:string">
    <xs:enumeration value="DATA"/>
    <xs:enumeration value="INSTRUCTION"/>
    <xs:enumeration value="UNIFIED"/>
  </xs:restriction>
</xs:simpleType>
```

4.2.4 Memory Access Behavior

The code-level WCET analysis in ARGO supports memory areas with different caching behaviors. A memory area can either be cached, uncached or locked in the cache (see Section 2.1.2). Furthermore a *"guarded"* (see Section 2.1.2) property stating that speculative loads are forbidden is supported by the WCET analysis. For both properties, SHIM does not include adequate attributes, so ARGO adds the attributes *"guarded"* and *"cacheBehaviour"* to the *"Accessor"* type. The *"guarded"* attribute is of boolean type and defaults to *"false"* while *"cacheBehaviour"* is of type *"CacheLockType"* which is a string constrained to the values *"UNCACHED"*, *"CACHED"* and *"LOCKED"* and defaults to *"CACHED"*.

The *PerformanceSet* element in *Accessor* can be used to describe fixed memory access times. If the memory is e.g. accessed through a network on chip (NoC), the memory accesses however are indirect. The timing then depends on the route taken through the NoC. Since routing is typically determined at runtime or on program initialization, this generally makes the access times software dependent. For the worst case timing it would be possible to assume the worst possible routing path in order to get a single predetermined value for the access times. This however is a very conservative assumption which would lead to significantly overestimated worst case timings. In most cases, it would rather be reasonable to calculate the access times based on timing attributes of the NoC combined with the software-defined routing properties.

For that reason, the ARGO ADL adds the boolean attribute *routedAccess* to the *Accessor* element which indicates that memory accesses are indirect and need to be routed through a NoC. If *routedAccess* is set to true, the performance set in *Accessor* is meant to be interpreted as the access timing without routing delays. The actual access time then is the delay along the routing path plus the access times in the *Accessor* element. If the *routedAccess* attribute is omitted, the default value is false.

In the XSD, the described changes appear as follows:

```
<xs:complexType name="Accessor">
  <xs:sequence>
    <xs:element name="PerformanceSet" type="PerformanceSet"
                minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:attribute name="masterComponentRef" use="required" type="xs:IDREF"/>
  <xs:attribute name="guarded" use="optional" type="xs:boolean" default="false" />
  <xs:attribute name="cacheBehaviour" use="optional"
                type="CacheLockType" default="CACHED" />
  <xs:attribute name="routedAccess" use="optional"
                type="xs:boolean" default="false" />
</xs:complexType>
<!-- ... -->

<xs:simpleType name="CacheLockType" >
  <xs:restriction base="xs:string">
    <xs:enumeration value="UNCACHED"/>
    <xs:enumeration value="CACHED"/>
    <xs:enumeration value="LOCKED"/>
  </xs:restriction>
</xs:simpleType>
```

4.2.5 Performance of Shared Resources

To get a tight system-level WCET estimation, it is reasonable to denote the worst case performance with respect to the maximum possible number of concurrent accesses on a shared resource (see Section 2.2). SHIM allows to specify different performance values using a *PerformanceSet.* However the selection of an actual performance value can only be made based on the access type (the *accessTypeRef* attribute of the *Performance* element). In order to differentiate performance values by the number of concurrent accesses, in ARGO the new attribute *interferenceCount* is introduced for SHIMs *Performance* element. The optional attribute holds an integer value which denotes for how many interfering accesses the given performance data is valid. In the XSD schema, the extensions appear as follows:

```
<xs:complexType name="Performance">
  <xs:sequence>
    <xs:element name="Pitch" type="Pitch" minOccurs="1" maxOccurs="1"/>
    <xs:element name="Latency" type="Latency" minOccurs="1" maxOccurs="1"/>
```

```
  </xs:sequence>
  <xs:attribute name="accessTypeRef" use="optional" type="xs:IDREF"/>
  <xs:attribute name="interferenceCount" use="optional" type="xs:int"/>
</xs:complexType>
```

4.3 ARGO ADL Description Guidelines

SHIM offers some degrees of freedom for the level of detail, the designer chooses to add to an architecture description. In ARGO however there is a minimum amount of information, the tools require to be contained in an ADL description. Given this, further constraints have to be added for ADL descriptions in order to be usable with the ARGO toolchain. For example, for WCET analysis in ARGO, sufficient timing information has to be annotated even though the SHIM specification itself does not force *"Performance"* elements to be present. To ensure compatibility with the ARGO requirements, some essential guidelines for writing ARGO ADL descriptions are presented in this section.

4.3.1 Top Level Elements

On the top-level of the XML document, exactly one *SystemConfiguration* element and no other elements should occur. The clock frequency in *SystemConfiguration* should be interpreted as the worst case (i.e. slowest possible) basic clock frequency of the overall system (In compliance to the SHIM specification, each *MasterComponent* can optionally define a deviating worst case frequency). The *name* attribute of the *SystemConfiguration* will be used by the ARGO tools to identify the basic platform type which determines the toolset and API being used for code generation (see Section 2.3.1). Accordingly, the name has to match one of the supported platforms of the employed ARGO toolchain version.

4.3.2 Processing Resources and Instruction Sets

Processor cores are described using *MasterComponent* and setting the *masterType* attribute to processing unit (PU). The basic processor architecture must be specified using the *arch* attribute and additional information such as available instruction set extensions or the supported ISA subset (necessary for code generation; see Section 2.3.3) can be added using the optional *archOption* attribute. Again, the clock frequency in the component should be set to the worst case frequency. The *CommonInstructionSet* from the SHIM specification can be used to add special instructions and/or instruction performance information.

4.3.3 Caches

For caches, detailed attributes like the line size, the number of ways and the replacement strategy are defined as optional. These details however are required for the code-level WCET analysis. If they are not specified in the ADL description, the impact of the cache might be ignored by the analysis tools.

4.3.4 Usage of the SHIM Performance element

The *Performance* and *PerformanceSet* elements of SHIM are used to denote timing information in an ADL description. For the ARGO ADL, the *worst* attribute is required to be a safe upper bound. If the attributes *accessTypeRef* and *interferenceCount* are used, then the timing information corresponds to the respective access type when no more than the defined

number of interferences (concurrent accesses) can occur. If none of the attributes is set, the timing information is considered as default/fallback case that comes into consideration when no other *Performance* element matches. If only performance elements without *interferenceCount* attribute are used, the times for *N* interfering accesses are assumed to be *N* times the default values.

4.3.5 Memory Map

The description of the memory map required by the code-level WCET analysis (see Section 2.1.2) is done using the *AddressSpace* elements of SHIM. To denote the performance and cache behavior for the address spaces, it is necessary to specify master-slave bindings with all relevant accessors. The (worst case) performance of accesses can be specified with respect to the access type and the number of interferences using the *PerformanceSet* element of *Accessor*. The timing for concurrent accesses on a memory segment can be described using the new attribute *interferenceCount* of the *Performance* type (see 4.2.5).

In some cases such as the network on chip (NoC) used in the InvasIC platform (see Section 3.2), the timings for accesses to remote memories cannot be described statically without knowing how the data is routed over the NoC (e.g. in the InvasIC case, how guaranteed service channels are configured). In such cases the newly added *routedAccess* attribute (see Section 4.2.4) needs to be *true* in order to indicate, that the access to the address space is indirect. The performance given in the *Accessor* then defines only the fixed latency/pitch independent from the routing path. The overall access times can then be determined by additionally adding the delay of the routing path. The timings of the routing paths can be calculated using the structural architecture description depicted in the following section.

4.3.6 Structural Description

The structural description of the platform is required for the system-level WCET estimation (see Section 2.2). For simple platforms the master-slave bindings defined for the address spaces can be sufficient to describe all necessary connections and timings. The master-slave bindings e.g. allow to determine which memories are shared among processor cores and therefore are prone to interference. However if routing via networks on chip comes into play, a more detailed description of the network and bus topology is required. In this case the ADL description should define a *CommunicationSet* with more details on the interconnection topology between the master components. Within the communication set, different types of communications (FIFO, Shared Memory etc.) can be defined and it is also possible to annotate performance/timing information using the *PerformanceSet* within *Connection* elements. The structural description in the *CommunicationSet* will also be used to determine the inter-processor communication topology for code generation and scheduling (see Sections 2.3.1 and 2.3.2).

Routers in a NoC can be described using *MasterComponent,* setting the *masterType* attribute to transfer unit (TU) and the *arch* field to “Router”. The router type and routing scheme can be denoted using the *archOption* attribute of *MasterComponent.* In order to describe the coordinates of a router in a two dimensional NoC, the *pid* property of the *MasterComponent* should contain the X and Y coordinate of the router separated by a comma sign (x,y). An internal routing delay time can be specified by adding an *Instruction* with the name *“routeFlit”* to the instruction set (*CommonInstructionSet*) of the router component and defining the timing values in the instruction’s *PerformanceSet* element. Network interfaces in a NoC architecture can be described similar to routers using a *MasterComponent* with type “TU” and the architecture name “NI”.

Connections in the *CommunicationSet* can only be added between master components such as processing resources, network interfaces and routers. To calculate the timing of routed memory accesses it is necessary to know possible routing paths from a processor (master component) to a memory which is described as slave component. With the connections being only defined between master components, it is additionally required to know which master component has direct access to a memory. Since a *MasterSlaveBinding* can also occur when accesses are routed via different master components, they are not sufficient to derive this information. In ARGO it is therefore assumed, that all memories being in the same *ComponentSet* as a certain master component can be directly accessed by this master component. Furthermore this assumption holds if the master component is in one of the subsets on the slave's *ComponentSet.*

5. ADL description of the ARGO target platforms

The ARGO tool flow supports two different platform concepts. The FlexaWare platform on the one hand features an extensive runtime system and API with support for runtime scheduling and hardware abstraction. In contrast, InvasIC offers highly configurable hardware which is programmed using a very thin run time software layer which does not hide the hardware details. The ARGO ADL is capable of describing both platforms with a sufficient level of detail in order to enable code generation and system-level WCET estimation within the ARGO toolchain.

The following subsections provide more details on how the proposed ADL can be used to describe the two platforms. The FlexaWare section focuses more on the relation of the abstract software entities provided by the runtime system to the hardware components described by the ADL. The subsequent section about the InvasIC platform shows how hardware structure and timing information can be represented by means of an example ADL description.

5.1 FlexaWare

The FlexaWare platform features a runtime system which is capable of hiding the hardware details from the programmer. In order to relate the software components on top of this abstraction layer to the underlying hardware resources described in the ADL, it is necessary to specify additional mapping information.

At design-time, constraints can be specified on the mapping of tasks [1] to processing elements [2]. The mapping may be specified with the aid of the FlexaWare's software development environment (SDE) or can be specified using a FlexaWare specific syntax in a text file. The latter may be more convenient when the mapping is generated by tooling. The syntax of the mapping constraints requires both the identifiers of the cores present in the system, as well as identification keys of the tasks subject to the mapping constraints.

[1] The term task in this context refers to the terminology of the FlexaWare SDE which supports runtime scheduling of the referred FlexaWare tasks (runtime scheduling is not planned to be used in ARGO). It should not be mixed up with the concept of compile time tasks that are used for the static scheduling and optimization in the ARGO toolchain.

[2] A static-order schedule is not supported explicitly by the FlexaWare runtime system, but may be implicitly defined due to the synchronization semantics (blocking behaviour) on the communication channels. In addition, the ARGO toolchain supports static scheduling at compile time in order to generate one FlexaWare task per processor core. The runtime scheduler is not used in this case.

5.1.1 Resource description

A resource is described by an *amount* of a specific resource *type* and a *location*. The ARGO ADL is not explicit on the location of a resource in the system. Therefore, each component in the system is referred to with a unique *id.* Using an identification convention, the location of a resource can be derived from its *id*. In FlexaWare, a hierarchical identification scheme is adopted that considers the module (board), subsystem and core level. The ARGO ADL has two major types of components; *SlaveComponent* and *MasterComponent*. Again, to properly model a heterogeneous system, a convention is used to encode the type of hardware component in the *name* attribute of the ARGO ADL components. The amount of resources provided by each component is represented by the *size* attribute of each ARGO ADL component.

A full ARGO ADL specification of the FlexaWare configuration thus yields the identification, capacity and resource types found in the system. The connections between the individual components and their performance are modelled in the sections describing the address space, communication facilities and master-slave bindings.

5.1.2 Mapping Constraints

In order to execute the application correctly, some metadata is required about the type, amount and location of the required resources. This metadata first needs to be specified, after which it will be integrated into the application binary. Two lookup tables that are part of the metadata are relevant to the user. The first lookup table lists the activities included in the application binary. A task can only be created if the specified activity is present in the activity table. For a given activity, it specifies per architecture supported by that activity the required stack size, the required dynamic allocatable (pool) memory, the size of the structure containing the parameters and the required scheduling budget.

Next to the activity table, a lookup table is provided to contain any resource allocation constraints. These constraints allow a user to restrict the assignment of a task to a specific architecture, to a specific board, subsystem or core, or to a location relative to its parent task (that is the one creating the task). In the context of the ARGO project, each task is assigned at design-time to a specific processing element. The resource constraint table should then list for each task in the application a single entry that specifies the identifier of the core to which the task has been assigned. The ADL specifies which resources are available in the system to choose from.

The table exemplifies the information that needs to be partly provided by the user.

(Task) Identifier	(Activity / Function) Name	Resources Required		
		Type	**Location**	**Amount**
0	example	Processor	Subsystem 1, Core 2	1
		Memory	Subsystem 1, SPM 0	16KB

This completes the cycle in the workflow, where the FlexaWare platform configuration is defined in the ARGO ADL, which in turn is interpreted by the ARGO toolchain that generates code together with mapping constraints that define the specific resources to be used in the application. This is summarized by Figure 3.

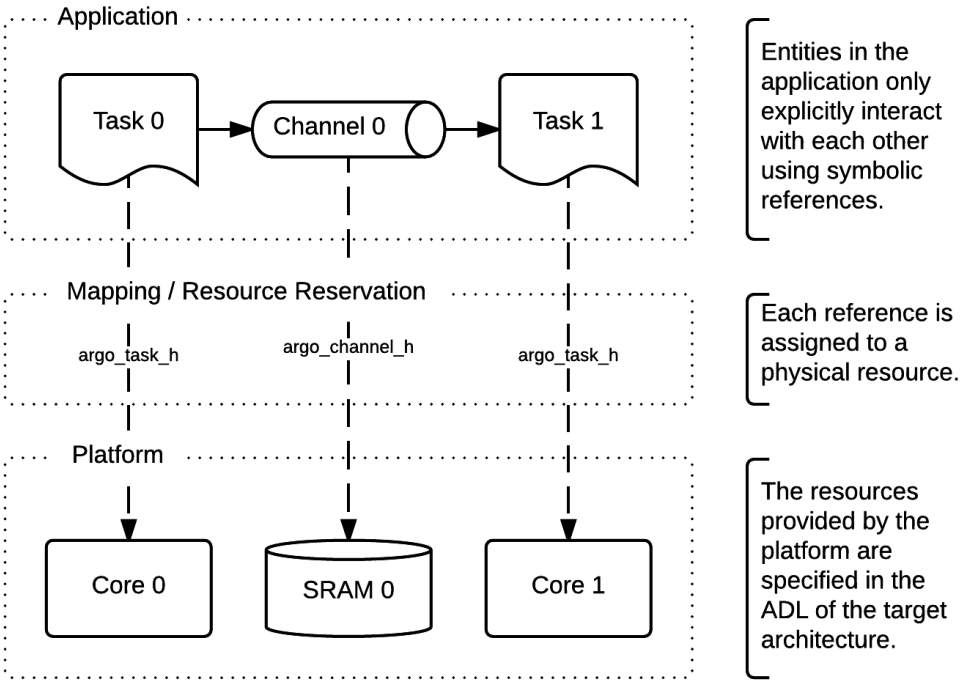

Figure 3: Relation between application structures and hardware components.

5.2 InvasIC descriptions

InvasIC is a highly configurable research platform realized as FPGA prototype. In the ARGO context, this provides a lot of flexibility to change the platform configuration and to evaluate how the different parameters affect the WCET performance in conjunction with the ARGO toolchain. The ARGO toolchain will support a range of InvasIC configurations which can formally be described using the ARGO ADL. The description guidelines in Section 4.3 thereby ensure compatibility with the ARGO tools.

Appendix B contains an example ADL description for a possible configuration of the InvasIC platform. It illustrates how the components are described for this platform. Note, that this is an example configuration which is not meant to fully reflect the future platform prototype implementation. The InvasIC platform is still under development and the given parameters may change significantly in the future. Especially the worst case performance values given in the example description are only rough estimates.

The InvasIC description defines a top-level component set which holds the entire platform. This set is subdivided into one component set for each of the tiles as well as a component set containing the routers of the NoC. The component descriptions comply with the guidelines from Section 4.3. Generally, it should be noted that a detailed behavioural description of the components is out of the scope of the ARGO ADL. Such detailed information is assumed to be known by the toolchain based on the architecture fields defined in the *MasterComponent* elements. This especially includes the details about the Leon3 processor cores, the ISA and the routing scheme used by the iNoC routers.

The connections between processor cores, network interfaces and routers are modelled using a communication set with FIFO and shared register communications. The latter are

used to model communication between a processor core and the corresponding network interface based on a shared register set. Each register represents one of the possible end points of a guaranteed service channel in the NoC (see also ARGO Deliverable 2.1). Write accesses by a processor core to one of the registers will cause the data item to be transmitted on the NoC, while read accesses return the data at the front of the incoming FIFO channel and remove the item from the FIFO. The FIFO communication elements in the ADL model the communication lines and internal data buffers of the NoC components (routers & network interfaces). Each of these interconnects can be shared between multiple guaranteed service channels in a predictable way due to the round robin arbitration at flit level in the NoC. The maximum amount of channels that can share one specific interconnect is limited by the hardware resources that are available in the routers (buffer space, etc.). The amount of NoC resources in general depends on the configuration of the hardware design. For the future, it is planned to specify such parameters in the ADL using the *archOption* field of the router components. The timing information for all communication connections is given as latency and pitch values according to the definition in the SHIM specification while sticking to the conventions described in Section 4.3.4.

In order to provide information about the physical address spaces for each processor, an address space set is defined in the description. It contains one individual address space per processor core and several subspaces reflecting the memories (scratchpad, tile local memory, DDR background memory, etc.) that can be accessed by the respective core.

Together with the description conventions and guidelines in Section 4.3, this platform description allows the ARGO toolchain to determine the involved hardware components for communication and memory accesses. The performance details in the description can be used to derive worst-case memory access and communication times.

6. ADL Toolchain Integration

As described in Section 4.1, the XML basis of SHIM and the ARGO extensions allows easy generation of a parser as well as a corresponding in-memory intermediate representation e.g. for Java and C++. Such an in-memory representation of the XML tree will constitute the main interface to make ADL data available to the ARGO toolchain. The ADL-IR will be complemented by validation routines which can be used to check if the ADL description matches all requirements (Section 2) and guidelines (Section 4.3).

Beyond the plain data model, the ADL interface will be enriched with several utility functions that will e.g. provide frequently used information (such as the number of processor cores in the system) on a more abstract level. Altogether the IR, the validation routines and the utility functions will be bundled to a library with a well-defined API for hardware information access and management.

7. Concluding remarks

This document specifies the first version of the ARGO specific extensions to the SHIM ADL as well as ADL description guidelines which ensure compatibility of an ADL description with the ARGO tools. The specification takes the requirements from different components of the ARGO toolchain into account. The most part of hardware information is required to generate a tight WCET estimation, but also parallelization and code generation tools rely on ADL data. The ADL is complemented by a library for parsing, accessing and managing ADL data within different ARGO tools.

The ADL description guidelines, the parser and data management library and if necessary the ADL specification itself will be continuously improved during the project. This facilitates to react to new requirements which may arise e.g. from improved WCET analysis techniques or new versions of the hardware platforms.

References

[1] "SOFTWARE-HARDWARE INTERFACE FOR MULTI-MANY-CORE (SHIM)," [Online]. Available: http://www.multicore-association.org/workgroup/shim.php. [Accessed 17 05 2016].

[2] P. H. Feiler, D. P. Gluch and J. J. Hudak, "The architecture analysis & design language (AADL): An introduction," 2006.

[3] "The Multicore Association," [Online]. Available: https://www.multicore-association.org/. [Accessed 17 05 2016].

[4] World Wide Web Consortium , "Extensible Markup Language (XML) 1.1 (Second Edition) - W3C Recommendation 16 August 2006," [Online]. Available: https://www.w3.org/TR/2006/REC-xml11-20060816/. [Accessed 17 05 2016].

[5] The Eclipse Foundation, "Eclipse Modeling Project," [Online]. Available: https://eclipse.org/modeling/. [Accessed 17 05 2016].

Appendix A: Full XML schema of the ARGO ADL based on SHIM 1.0

```
<?xml version="1.0"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="ComponentSet" type="ComponentSet"/>
  <xs:complexType name="ComponentSet">
    <xs:sequence>
      <xs:element name="ComponentSet" type="ComponentSet"
                  minOccurs="0" maxOccurs="unbounded"/>
      <xs:element name="SlaveComponent" type="SlaveComponent"
                  minOccurs="0" maxOccurs="unbounded"/>
      <xs:element name="MasterComponent" type="MasterComponent"
                  minOccurs="0" maxOccurs="unbounded"/>
      <xs:element name="Cache" type="Cache" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attribute name="name" use="required" type="xs:string"/>
  </xs:complexType>
  <xs:element name="SlaveComponent" type="SlaveComponent"/>
  <xs:complexType name="SlaveComponent">
    <xs:annotation>
      <xs:documentation>Memory</xs:documentation>
    </xs:annotation>
    <xs:sequence/>
    <xs:attribute name="name" use="required" type="xs:string"/>
    <xs:attribute name="id" use="required" type="xs:ID"/>
    <xs:attribute name="size" use="required" type="xs:int"/>
    <xs:attribute name="sizeUnit" use="required" type="SizeUnitType"/>
    <xs:attribute name="rwType" use="required" type="RWType"/>
  </xs:complexType>
  <xs:element name="MasterComponent" type="MasterComponent"/>
  <xs:complexType name="MasterComponent">
    <xs:sequence>
      <xs:element name="CommonInstructionSet" type="CommonInstructionSet"
                  minOccurs="0" maxOccurs="1"/>
      <xs:element name="Cache" type="Cache" minOccurs="0" maxOccurs="unbounded"/>
      <xs:element name="ClockFrequency" type="ClockFrequency" minOccurs="0" maxOccurs="1"/>
      <xs:element name="AccessTypeSet" type="AccessTypeSet" minOccurs="1" maxOccurs="1"/>
    </xs:sequence>
    <xs:attribute name="name" use="required" type="xs:string"/>
    <xs:attribute name="id" use="required" type="xs:ID"/>
    <xs:attribute name="masterType" use="required" type="MasterType"/>
    <xs:attribute name="arch" use="required" type="xs:string"/>
    <xs:attribute name="archOption" use="optional" type="xs:string"/>
    <xs:attribute name="pid" use="optional" type="xs:string"/>
    <xs:attribute name="nThread" use="optional" type="xs:int"/>
    <xs:attribute name="endian" use="optional" type="EndianType"/>
  </xs:complexType>
  <xs:simpleType name="RWType">
    <xs:restriction base="xs:string">
      <xs:enumeration value="RW"/>
      <xs:enumeration value="WX"/>
      <xs:enumeration value="RX"/>
      <xs:enumeration value="R"/>
      <xs:enumeration value="W"/>
      <xs:enumeration value="X"/>
      <xs:enumeration value="RWX"/>
    </xs:restriction>
  </xs:simpleType>
  <xs:element name="AddressSpaceSet" type="AddressSpaceSet"/>
  <xs:complexType name="AddressSpaceSet">
    <xs:sequence>
      <xs:element name="AddressSpace" type="AddressSpace"
                  minOccurs="1" maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
  <xs:element name="AddressSpace" type="AddressSpace"/>
  <xs:complexType name="AddressSpace">
    <xs:sequence>
      <xs:element name="SubSpace" type="SubSpace" minOccurs="0" maxOccurs="unbounded"/>
```

```
    </xs:sequence>
    <xs:attribute name="name" use="required" type="xs:string"/>
    <xs:attribute name="id" use="required" type="xs:ID"/>
  </xs:complexType>
  <xs:element name="SubSpace" type="SubSpace"/>
  <xs:complexType name="SubSpace">
    <xs:sequence>
      <xs:element name="MemoryConsistencyModel" type="MemoryConsistencyModel"
                  minOccurs="0" maxOccurs="unbounded"/>
      <xs:element name="MasterSlaveBindingSet" type="MasterSlaveBindingSet"
                  minOccurs="0" maxOccurs="1"/>
    </xs:sequence>
    <xs:attribute name="name" use="required" type="xs:string"/>
    <xs:attribute name="id" use="required" type="xs:ID"/>
    <xs:attribute name="start" use="required" type="xs:long"/>
    <xs:attribute name="end" use="required" type="xs:long"/>
    <xs:attribute name="endian" use="optional" type="EndianType"/>

    <xs:attribute name="contentType" use="optional"
                  type="AddrSpaceContentType" default="UNIFIED" />
  </xs:complexType>

  <xs:simpleType name="MasterType">
    <xs:restriction base="xs:string">
      <xs:enumeration value="PU">
        <xs:annotation>
          <xs:documentation>Processing Unit</xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="TU">
        <xs:annotation>
          <xs:documentation>Transffer Unit</xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="OTHER"/>
    </xs:restriction>
  </xs:simpleType>

  <xs:element name="Instruction" type="Instruction"/>
  <xs:complexType name="Instruction">
    <xs:sequence>
      <xs:element name="Performance" type="Performance" minOccurs="1" maxOccurs="1"/>
    </xs:sequence>
    <xs:attribute name="name" use="required" type="xs:string"/>
  </xs:complexType>
  <xs:element name="InterruptCommunication" type="InterruptCommunication"/>
  <xs:complexType name="InterruptCommunication">
    <xs:complexContent>
      <xs:extension base="AbstractCommunication">
        <xs:sequence/>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
  <xs:element name="Latency" type="Latency"/>
  <xs:complexType name="Latency">
    <xs:complexContent>
      <xs:extension base="AbstractPerformance">
        <xs:sequence/>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
  <xs:element name="AbstractPerformance" type="AbstractPerformance"/>
  <xs:complexType name="AbstractPerformance" abstract="true">
    <xs:sequence/>
    <xs:attribute name="best" use="optional" type="xs:float"/>
    <xs:attribute name="typical" use="optional" type="xs:float"/>
    <xs:attribute name="worst" use="optional" type="xs:float"/>
  </xs:complexType>
  <xs:element name="Pitch" type="Pitch"/>
  <xs:complexType name="Pitch">
    <xs:complexContent>
      <xs:extension base="AbstractPerformance">
        <xs:sequence/>
```

```
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
<xs:element name="MasterSlaveBinding" type="MasterSlaveBinding"/>
<xs:complexType name="MasterSlaveBinding">
  <xs:sequence>
    <xs:element name="Accessor" type="Accessor" minOccurs="1" maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:attribute name="slaveComponentRef" use="required" type="xs:IDREF"/>
</xs:complexType>
<xs:element name="CommunicationSet" type="CommunicationSet"/>
<xs:complexType name="CommunicationSet">
  <xs:sequence>
    <xs:element name="SharedRegisterCommunication" type="SharedRegisterCommunication"
                minOccurs="0" maxOccurs="unbounded"/>
    <xs:element name="SharedMemoryCommunication" type="SharedMemoryCommunication"
                minOccurs="0" maxOccurs="unbounded"/>
    <xs:element name="EventCommunication" type="EventCommunication"
                minOccurs="0" maxOccurs="unbounded"/>
    <xs:element name="FIFOCommunication" type="FIFOCommunication"
                minOccurs="0" maxOccurs="unbounded"/>
    <xs:element name="InterruptCommunication" type="InterruptCommunication"
                minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>
<xs:element name="AbstractCommunication" type="AbstractCommunication"/>
<xs:complexType name="AbstractCommunication" abstract="true">
  <xs:sequence>
    <xs:element name="ConnectionSet" type="ConnectionSet" minOccurs="0" maxOccurs="1"/>
  </xs:sequence>
  <xs:attribute name="name" use="required" type="xs:string"/>
</xs:complexType>

<xs:element name="Connection" type="Connection"/>
<xs:complexType name="Connection">
  <xs:sequence>
    <xs:element name="Performance" type="Performance" minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:attribute name="from" use="required" type="xs:IDREF">
    <xs:annotation>
      <xs:documentation>Reference to the instance of MasterComponent</xs:documentation>
    </xs:annotation>
  </xs:attribute>
  <xs:attribute name="to" use="required" type="xs:IDREF">
    <xs:annotation>
      <xs:documentation>Reference to the instance of MasterComponent</xs:documentation>
    </xs:annotation>
  </xs:attribute>
</xs:complexType>
<xs:element name="PerformanceSet" type="PerformanceSet"/>
<xs:complexType name="PerformanceSet">
  <xs:sequence>
    <xs:element name="Performance" type="Performance" minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>

<xs:element name="FIFOCommunication" type="FIFOCommunication"/>
<xs:complexType name="FIFOCommunication">
  <xs:complexContent>
    <xs:extension base="AbstractCommunication">
      <xs:sequence/>
      <xs:attribute name="dataSize" use="required" type="xs:int"/>
      <xs:attribute name="dataSizeUnit" use="optional" type="SizeUnitType"/>
      <xs:attribute name="queueSize" use="required" type="xs:int"/>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
<xs:element name="CommonInstructionSet" type="CommonInstructionSet"/>
<xs:complexType name="CommonInstructionSet">
  <xs:sequence>
    <xs:element name="Instruction" type="Instruction" minOccurs="1" maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:attribute name="name" use="required" type="xs:string"/>
```

```
  </xs:complexType>
  <xs:element name="Cache" type="Cache"/>
  <xs:complexType name="Cache">
    <xs:sequence>
      <xs:element name="cacheRef" type="xs:IDREF" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attribute name="name" use="required" type="xs:string"/>
    <xs:attribute name="id" use="required" type="xs:ID"/>
    <xs:attribute name="cacheType" use="required" type="CacheType">
      <xs:annotation>
        <xs:documentation>soft / hard</xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="cacheCoherency" use="required" type="CacheCoherencyType"/>
    <xs:attribute name="size" use="required" type="xs:int"/>
    <xs:attribute name="sizeUnit" use="required" type="SizeUnitType"/>
    <xs:attribute name="nWay" use="optional" type="xs:int"/>
    <xs:attribute name="lineSize" use="optional" type="xs:int"/>
    <xs:attribute name="lockDownType" use="optional" type="LockDownType"/>
    <xs:attribute name="replacementStrategy" use="optional" type="CacheReplacementStrat"/>
  </xs:complexType>
  <xs:element name="SystemConfiguration" type="SystemConfiguration"/>
  <xs:complexType name="SystemConfiguration">
    <xs:sequence>
      <xs:element name="ComponentSet" type="ComponentSet" minOccurs="1" maxOccurs="1"/>
      <xs:element name="CommunicationSet" type="CommunicationSet"
                  minOccurs="0" maxOccurs="1"/>
      <xs:element name="AddressSpaceSet" type="AddressSpaceSet" minOccurs="0" maxOccurs="1"/>
      <xs:element name="ClockFrequency" type="ClockFrequency" minOccurs="1" maxOccurs="1"/>
    </xs:sequence>
    <xs:attribute name="name" use="required" type="xs:string"/>
    <xs:attribute name="shimVersion" use="required" type="xs:string"/>
  </xs:complexType>
  <xs:element name="ConnectionSet" type="ConnectionSet"/>
  <xs:complexType name="ConnectionSet">
    <xs:sequence>
      <xs:element name="Connection" type="Connection" minOccurs="1" maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
  <xs:simpleType name="CacheCoherencyType">
    <xs:restriction base="xs:string">
      <xs:enumeration value="SOFT"/>
      <xs:enumeration value="HARD"/>
    </xs:restriction>
  </xs:simpleType>
  <xs:element name="MemoryConsistencyModel" type="MemoryConsistencyModel"/>
  <xs:complexType name="MemoryConsistencyModel">
    <xs:sequence/>
    <xs:attribute name="rawOrdering" use="optional" type="OrderingType">
      <xs:annotation>
        <xs:documentation>Read After Write</xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="warOrdering" use="optional" type="OrderingType">
      <xs:annotation>
        <xs:documentation>Write After Read</xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="wawOrdering" use="optional" type="OrderingType">
      <xs:annotation>
        <xs:documentation>Write After Write</xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="rarOrdering" use="optional" type="OrderingType"/>
  </xs:complexType>
  <xs:simpleType name="OrderingType">
    <xs:restriction base="xs:string">
      <xs:enumeration value="ORDERD"/>
      <xs:enumeration value="UNORDERD"/>
    </xs:restriction>
  </xs:simpleType>
  <xs:simpleType name="EndianType">
    <xs:restriction base="xs:string">
```

```
      <xs:enumeration value="LITTLE"/>
      <xs:enumeration value="BIG"/>
    </xs:restriction>
  </xs:simpleType>
  <xs:element name="SharedRegisterCommunication" type="SharedRegisterCommunication"/>
  <xs:complexType name="SharedRegisterCommunication">
    <xs:complexContent>
      <xs:extension base="AbstractCommunication">
        <xs:sequence/>
        <xs:attribute name="dataSize" use="required" type="xs:int"/>
        <xs:attribute name="dataSizeUnit" use="required" type="SizeUnitType"/>
        <xs:attribute name="nRegister" use="required" type="xs:int"/>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
  <xs:element name="SharedMemoryCommunication" type="SharedMemoryCommunication"/>
  <xs:complexType name="SharedMemoryCommunication">
    <xs:complexContent>
      <xs:extension base="AbstractCommunication">
        <xs:sequence/>
        <xs:attribute name="operationType" use="optional" type="OperationType"/>
        <xs:attribute name="dataSize" use="optional" type="xs:int"/>
        <xs:attribute name="dataSizeUnit" use="optional" type="SizeUnitType"/>
        <xs:attribute name="addressSpaceRef" use="optional" type="xs:IDREF"/>
        <xs:attribute name="subSpaceRef" use="optional" type="xs:IDREF"/>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
  <xs:element name="EventCommunication" type="EventCommunication"/>
  <xs:complexType name="EventCommunication">
    <xs:complexContent>
      <xs:extension base="AbstractCommunication">
        <xs:sequence/>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>

  <xs:element name="ClockFrequency" type="ClockFrequency"/>
  <xs:complexType name="ClockFrequency">
    <xs:sequence minOccurs="1"/>
    <xs:attribute name="clockValue" use="required" type="xs:float"/>
  </xs:complexType>
  <xs:element name="Accessor" type="Accessor"/>
  <xs:complexType name="Accessor">
    <xs:sequence>
      <xs:element name="PerformanceSet" type="PerformanceSet"
                  minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attribute name="masterComponentRef" use="required" type="xs:IDREF"/>
    <xs:attribute name="guarded" use="optional" type="xs:boolean" default="false"/>
    <xs:attribute name="cacheBehaviour" use="optional" type="CacheLockType" default="CACHED"/>
    <xs:attribute name="routedAccess" use="optional" type="xs:boolean" default="false"/>
  </xs:complexType>
  <xs:element name="AccessType" type="AccessType"/>
  <xs:complexType name="AccessType">
    <xs:sequence minOccurs="1"/>
    <xs:attribute name="name" use="required" type="xs:string"/>
    <xs:attribute name="id" use="required" type="xs:ID"/>
    <xs:attribute name="rwType" use="optional" type="RWType"/>
    <xs:attribute name="accessByteSize" use="optional" type="xs:int"/>
    <xs:attribute name="alignmentByteSize" use="optional" type="xs:int"/>
    <xs:attribute name="nBurst" use="optional" type="xs:int"/>
  </xs:complexType>
  <xs:element name="MasterSlaveBindingSet" type="MasterSlaveBindingSet"/>
  <xs:complexType name="MasterSlaveBindingSet">
    <xs:sequence>
      <xs:element name="MasterSlaveBinding" type="MasterSlaveBinding"
                  minOccurs="1" maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
  <xs:simpleType name="CacheType">
    <xs:restriction base="xs:string">
      <xs:enumeration value="DATA"/>
```

```
      <xs:enumeration value="INSTRUCTION"/>
      <xs:enumeration value="UNIFIED"/>
    </xs:restriction>
  </xs:simpleType>
  <xs:simpleType name="CacheReplacementStrat">
    <xs:restriction base="xs:string">
      <xs:enumeration value="FIFO"/>
      <xs:enumeration value="LRU"/>
      <xs:enumeration value="PLRU"/>
    </xs:restriction>
  </xs:simpleType>
  <xs:element name="Performance" type="Performance"/>
  <xs:complexType name="Performance">
    <xs:sequence>
      <xs:element name="Pitch" type="Pitch" minOccurs="1" maxOccurs="1"/>
      <xs:element name="Latency" type="Latency" minOccurs="1" maxOccurs="1"/>
    </xs:sequence>
    <xs:attribute name="accessTypeRef" use="optional" type="xs:IDREF"/>
    <xs:attribute name="interferenceCount" use="optional" type="xs:int"/>
  </xs:complexType>
  <xs:element name="AccessTypeSet" type="AccessTypeSet"/>
  <xs:complexType name="AccessTypeSet">
    <xs:sequence minOccurs="1" maxOccurs="1">
      <xs:element name="AccessType" type="AccessType" minOccurs="1" maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
  <xs:simpleType name="SizeUnitType">
    <xs:restriction base="xs:string">
      <xs:enumeration value="KiB"/>
      <xs:enumeration value="B"/>
      <xs:enumeration value="GiB"/>
      <xs:enumeration value="MiB"/>
      <xs:enumeration value="TiB"/>
    </xs:restriction>
  </xs:simpleType>
  <xs:simpleType name="LockDownType">
    <xs:restriction base="xs:string">
      <xs:enumeration value="LINE"/>
      <xs:enumeration value="NONE"/>
      <xs:enumeration value="WAY"/>
    </xs:restriction>
  </xs:simpleType>
  <xs:simpleType name="OperationType">
    <xs:restriction base="xs:string">
      <xs:enumeration value="TAS">
        <xs:annotation>
          <xs:documentation>Test and Set</xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="LLSC">
        <xs:annotation>
          <xs:documentation>Load Link/Store Conditional</xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="CAX">
        <xs:annotation>
          <xs:documentation>Compare and Exchange</xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="OTHER"/>
    </xs:restriction>
  </xs:simpleType>
  <xs:simpleType name="AddrSpaceContentType">
    <xs:restriction base="xs:string">
      <xs:enumeration value="DATA"/>
      <xs:enumeration value="INSTRUCTION"/>
      <xs:enumeration value="UNIFIED"/>
    </xs:restriction>
  </xs:simpleType>
  <xs:simpleType name="CacheLockType" >
    <xs:restriction base="xs:string">
      <xs:enumeration value="UNCACHED"/>
      <xs:enumeration value="CACHED"/>
```

```
      <xs:enumeration value="LOCKED"/>
    </xs:restriction>
  </xs:simpleType>
</xs:schema>
```

Appendix B: InvasIC platform example description

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<SystemConfiguration name="InvasIC" shimVersion="1.0">
    <ComponentSet name="InvasIC System">
        <ComponentSet name="Tile_0_0">
            <SlaveComponent name="TileLocalMemory_0_0" id="SC_858952163142620161104105347204"
                  size="8" sizeUnit="MiB" rwType="RW"/>
            <SlaveComponent name="ScratchpadMemory_0_0" id="SC_230643635746020161104105347208"
                  size="128" sizeUnit="KiB" rwType="RW"/>
            <SlaveComponent name="NIRegisterSet_0_0" id="SC_153994703719052016112117202625 4"
                  size="255" sizeUnit="B" rwType="RW"/>
            <MasterComponent name="Core_0_0" id="MC_611572016123020161104150944371"
                  masterType="PU" arch="Leon3" pid="64" nThread="1" endian="BIG">
                <Cache name="Cache_0_0_0" id="C_155912251366632016110416025736 7"
                  cacheType="INSTRUCTION" cacheCoherency="SOFT" size="64" sizeUnit="KiB"
                  nWay="4" lineSize="16" lockDownType="LINE" replacementStrategy="LRU" />
                <ClockFrequency clockValue="2.5E7"/>
                <AccessTypeSet>
                    <AccessType name="AT_0_0_0" id="AT_772698783293201611041509443 73"
                      rwType="RWX" accessByteSize="4" alignmentByteSize="4" nBurst="4"/>
                    <AccessType name="AT_0_0_1" id="AT_936292831997720161110162723057"
                      rwType="RWX" accessByteSize="1" alignmentByteSize="1" nBurst="1"/>
                </AccessTypeSet>
            </MasterComponent>
            <MasterComponent name="NI_0_0" id="MC_13606572236621201611041053471 96"
                  masterType="TU" arch="NI" pid="0" nThread="1" endian="BIG">
                <AccessTypeSet>
                    <AccessType name="AT_0_0_0_0" id="AT_1922464006683920161104105440857"
                      rwType="RWX" accessByteSize="4" alignmentByteSize="4" nBurst="1"/>
                </AccessTypeSet>
            </MasterComponent>
        </ComponentSet>
        <ComponentSet name="Tile_1_0">
            <SlaveComponent name="TileLocalMemory_1_0" id="SC_108941827220092016110410534720 5"
                  size="8" sizeUnit="MiB" rwType="RW"/>
            <SlaveComponent name="ScratchpadMemory_1_0"
                  id="SC_16361826552841201611041053472 08" size="128" sizeUnit="KiB"
                  rwType="RW"/>
            <SlaveComponent name="NIRegisterSet_1_0" id="SC_161413356304062016112117210188 6"
                  size="255" sizeUnit="B" rwType="RW"/>
            <MasterComponent name="Core_1_0" id="MC_589404864217201611041053471 98"
                  masterType="PU" arch="Leon3" pid="1" nThread="1" endian="BIG">
                <Cache name="Cache_0_0_1" id="C_161755016033092016110410544085 7"
                      cacheType="INSTRUCTION" cacheCoherency="SOFT" size="64" sizeUnit="KiB"
                      nWay="4" lineSize="16" lockDownType="LINE" replacementStrategy="LRU" />
                <ClockFrequency clockValue="2.5E7"/>
                <AccessTypeSet>
                    <AccessType name="AT_1_0_0" id="AT_1437941060035720161110162835736"
                      rwType="RWX" accessByteSize="4" alignmentByteSize="4" nBurst="4"/>
                    <AccessType name="AT_1_0_1" id="AT_1210830415374520161110162835736"
                      rwType="RWX" accessByteSize="1" alignmentByteSize="1" nBurst="1"/>
                </AccessTypeSet>
            </MasterComponent>
            <MasterComponent name="NI_1_0" id="1336996537832020161104151112625"
                  masterType="TU" arch="NI" pid="0" nThread="1" endian="BIG">
                <ClockFrequency clockValue="2.5E7"/>
                <AccessTypeSet>
                    <AccessType name="AT_0_0_0_0" id="AT_7967307695420161104151112625"
                      rwType="R" accessByteSize="4" alignmentByteSize="4" nBurst="1"/>
                </AccessTypeSet>
            </MasterComponent>
        </ComponentSet>
        <ComponentSet name="Tile_1_1">
            <SlaveComponent name="TileLocalMemory_1_1" id="SC_184700847166732016110410534720 6"
                  size="8" sizeUnit="MiB" rwType="RW"/>
            <SlaveComponent name="ScratchpadMemory_1_1"
                  id="SC_193283145014912016110410534720 9" size="128" sizeUnit="KiB"
                  rwType="RW"/>
            <SlaveComponent name="NIRegisterSet_1_1" id="SC_178604517211320161121172120341"
                  size="255" sizeUnit="B" rwType="RW"/>
```

```
            <MasterComponent name="Core_1_1" id="MC_199785917102862016110410534719 9"
                  masterType="PU" arch="Leon3" pid="2" nThread="1" endian="BIG">
                <Cache name="Cache_0_0_2" id="C_132512418686262016110410544085 7"
                      cacheType="INSTRUCTION" cacheCoherency="SOFT" size="64" sizeUnit="KiB"
                      nWay="4" lineSize="16" lockDownType="LINE" replacementStrategy="LRU" />
                <ClockFrequency clockValue="2.5E7"/>
                <AccessTypeSet>
                    <AccessType name="AT_1_1_0" id="AT_84351272620952016111016290859 6"
                      rwType="RWX" accessByteSize="4" alignmentByteSize="4" nBurst="4"/>
                    <AccessType name="AT_1_1_1" id="AT_77398990672652016111016290859 6"
                      rwType="RWX" accessByteSize="1" alignmentByteSize="1" nBurst="1"/>
                </AccessTypeSet>
            </MasterComponent>
            <MasterComponent name="NI_1_1" id="10724106415300201611041511241 64"
                  masterType="TU" arch="NI" pid="0" nThread="1" endian="BIG">
                <ClockFrequency clockValue="2.5E7"/>
                <AccessTypeSet>
                    <AccessType name="AT_0_0_0_0" id="AT_28331893885372016110415112416 5"
                  rwType="R" accessByteSize="4" alignmentByteSize="4" nBurst="1"/>
                </AccessTypeSet>
            </MasterComponent>
        </ComponentSet>
        <ComponentSet name="Tile_0_1(MemoryTile)">
            <SlaveComponent name="DDR_Memory" id="SC_203612783804292016110410534720 6" size="2"
                  sizeUnit="GiB" rwType="RW"/>
            <SlaveComponent name="TileLocalMemory_0_1" id="SC_257608164903520161104105347207"
                  size="8" sizeUnit="MiB" rwType="RW"/>
            <SlaveComponent name="NIRegisterSet_0_1" id="SC_48230769855062016112117213202 3"
                  size="255" sizeUnit="B" rwType="RW"/>
            <MasterComponent name="MemoryControlCore_0_1"
                  id="MC_95986940787092016110410534720 0" masterType="PU" arch="Leon3" pid="3"
                  nThread="1" endian="BIG">
                <Cache name="Cache_0_0_3" id="C_46116082883852016110410544085 7"
                  cacheType="UNIFIED" cacheCoherency="SOFT" size="64" sizeUnit="KiB" nWay="4"
                  lineSize="16" lockDownType="LINE"  replacementStrategy="LRU" />
                <ClockFrequency clockValue="2.5E7"/>
                <AccessTypeSet>
                    <AccessType name="AT_0_1_0" id="AT_75200156715722016111016293538 6"
                      rwType="RWX" accessByteSize="4" alignmentByteSize="4" nBurst="4"/>
                    <AccessType name="AT_0_1_1" id="AT_77737908462702016111016293538 6"
                      rwType="RWX" accessByteSize="1" alignmentByteSize="1" nBurst="1"/>
                </AccessTypeSet>
            </MasterComponent>
            <MasterComponent name="NI_0_1" id="11180785045492201611041511400 02"
                  masterType="TU" arch="NI" pid="0" nThread="1" endian="BIG">
                <ClockFrequency clockValue="2.5E7"/>
                <AccessTypeSet>
                    <AccessType name="AT_0_0_0_0" id="AT_69169048657422016110415114000 3"
                      rwType="R" accessByteSize="4" alignmentByteSize="4" nBurst="1"/>
                </AccessTypeSet>
            </MasterComponent>
        </ComponentSet>
        <ComponentSet name="NoC">
            <MasterComponent name="Router_0_0" id="MC_116237769821020161104105347200"
                  masterType="TU" arch="Router" archOption="iNoC" pid="0,0" nThread="1"
                  endian="BIG">
                <ClockFrequency clockValue="2.5E7"/>
                <AccessTypeSet>
                    <AccessType name="AT_0_0_4_0" id="AT_10083150450889201611041054408 57"
                      rwType="R" accessByteSize="4" alignmentByteSize="4" nBurst="8"/>
                </AccessTypeSet>
            </MasterComponent>
            <MasterComponent name="Router_0_1" id="MC_159419980810872016110410534720 1"
                  masterType="TU" arch="Router" archOption="iNoC" pid="0,1" nThread="1"
                  endian="BIG">

                <ClockFrequency clockValue="2.5E7"/>
                <AccessTypeSet>
                    <AccessType name="AT_0_0_5_0" id="AT_12397599900414201611041054408 57"
                      rwType="R" accessByteSize="4" alignmentByteSize="4" nBurst="8"/>
                </AccessTypeSet>
            </MasterComponent>
            <MasterComponent name="Router_1_0" id="MC_191296276703542016110410534720 2"
```

```
                masterType="TU" arch="Router" archOption="iNoC" pid="1,0" nThread="1"
                endian="BIG">
            <ClockFrequency clockValue="2.5E7"/>
            <AccessTypeSet>
                <AccessType name="AT_0_0_6_0" id="AT_902830499521720161104105440857"
                  rwType="R" accessByteSize="4" alignmentByteSize="4" nBurst="8"/>
            </AccessTypeSet>
        </MasterComponent>
        <MasterComponent name="Router_1_1" id="MC_176919068309342016110410534720 3"
              masterType="TU" arch="Router" archOption="iNoC" pid="1,1" nThread="1"
              endian="BIG">
            <ClockFrequency clockValue="2.5E7"/>
            <AccessTypeSet>
                <AccessType name="AT_0_0_7_0" id="AT_12786778726239201611041054408 57"
                  rwType="R" accessByteSize="4" alignmentByteSize="4" nBurst="8"/>
            </AccessTypeSet>
        </MasterComponent>
    </ComponentSet>
</ComponentSet>
<CommunicationSet>
    <SharedRegisterCommunication dataSize="4" dataSizeUnit="B" nRegister="16"
              name="CoreNI_AHB">
        <ConnectionSet>
            <Connection from="MC_61157201612302016110415094437 1"
                  to="MC_13606572236621201611041053471 96">
                <Performance accessTypeRef="AT_192246400668392016110410544085 7">
                    <Pitch typical="1.0" worst="2.0"/>
                    <Latency typical="3.0" worst="3.0"/>
                </Performance>
            </Connection>
            <Connection from="MC_13606572236621201611041053471 96"
                  to="MC_61157201612302016110415094437 1">
                <Performance accessTypeRef="AT_192246400668392016110410544085 7">
                    <Pitch typical="1.0" worst="2.0"/>
                    <Latency typical="3.0" worst="3.0"/>
                </Performance>
            </Connection>
            <Connection from="MC_58940486421720161104105347198"
                  to="13369965378320201611041511126 25">
                <Performance accessTypeRef="AT_192246400668392016110410544085 7">
                    <Pitch typical="1.0" worst="2.0"/>
                    <Latency typical="3.0" worst="3.0"/>
                </Performance>
            </Connection>
            <Connection from="13369965378320201611041511126 25"
                  to="MC_58940486421720161104105347198">
                <Performance accessTypeRef="AT_192246400668392016110410544085 7">
                    <Pitch typical="1.0" worst="2.0"/>
                    <Latency typical="3.0" worst="3.0"/>
                </Performance>
            </Connection>
            <Connection from="MC_95986940787092016110410534720 0"
                  to="11180785045492201611041511400 02">
                <Performance accessTypeRef="AT_192246400668392016110410544085 7">
                    <Pitch typical="1.0" worst="2.0"/>
                    <Latency typical="3.0" worst="3.0"/>
                </Performance>
            </Connection>
            <Connection from="11180785045492201611041511400 02"
                  to="MC_95986940787092016110410534720 0">
                <Performance accessTypeRef="AT_192246400668392016110410544085 7">
                    <Pitch typical="1.0" worst="2.0"/>
                    <Latency typical="3.0" worst="3.0"/>
                </Performance>
            </Connection>
            <Connection from="10724106415300201611041511241 64"
                  to="MC_19978591710286201611041053471 99">
                <Performance accessTypeRef="AT_192246400668392016110410544085 7">
                    <Pitch typical="1.0" worst="2.0"/>
                    <Latency typical="3.0" worst="3.0"/>
                </Performance>
            </Connection>
            <Connection from="MC_19978591710286201611041053471 99"
```

```
                    to="10724106415300201611041511241 64">
                <Performance accessTypeRef="AT_1922464006683920161104105440857">
                    <Pitch typical="1.0" worst="2.0"/>
                    <Latency typical="3.0" worst="3.0"/>
                </Performance>
            </Connection>
        </ConnectionSet>
    </SharedRegisterCommunication>
    <FIFOCommunication dataSize="4" dataSizeUnit="B" queueSize="4"
             name="iNoC_Interconnects">
        <ConnectionSet>
            <Connection from="MC_116237769821020161104105347200"
                  to="MC_159419980810872016110410534720 1">
                <Performance>
                    <Pitch typical="3.0" worst="5.0"/>
                    <Latency typical="3.0" worst="5.0"/>
                </Performance>
            </Connection>
            <Connection from="MC_116237769821020161104105347200"
                  to="MC_191296276703542016110410534720 2">
                <Performance>
                    <Pitch typical="3.0" worst="5.0"/>
                    <Latency typical="3.0" worst="5.0"/>
                </Performance>
            </Connection>
            <Connection from="MC_176919068309342016110410534720 3"
                  to="MC_191296276703542016110410534720 2">
                <Performance>
                    <Pitch typical="3.0" worst="5.0"/>
                    <Latency typical="3.0" worst="5.0"/>
                </Performance>
            </Connection>
            <Connection from="MC_176919068309342016110410534720 3"
                  to="MC_159419980810872016110410534720 1">
                <Performance>
                    <Pitch typical="3.0" worst="5.0"/>
                    <Latency typical="3.0" worst="5.0"/>
                </Performance>
            </Connection>
            <Connection from="MC_159419980810872016110410534720 1"
                  to="MC_116237769821020161104105347200">
                <Performance>
                    <Pitch typical="3.0" worst="5.0"/>
                    <Latency typical="3.0" worst="5.0"/>
                </Performance>
            </Connection>
            <Connection from="MC_191296276703542016110410534720 2"
                  to="MC_116237769821020161104105347200">
                <Performance>
                    <Pitch typical="3.0" worst="5.0"/>
                    <Latency typical="3.0" worst="5.0"/>
                </Performance>
            </Connection>
            <Connection from="MC_191296276703542016110410534720 2"
                  to="MC_176919068309342016110410534720 3">
                <Performance>
                    <Pitch typical="3.0" worst="5.0"/>
                    <Latency typical="3.0" worst="5.0"/>
                </Performance>
            </Connection>
            <Connection from="MC_159419980810872016110410534720 1"
                  to="MC_176919068309342016110410534720 3">
                <Performance>
                    <Pitch typical="3.0" worst="5.0"/>
                    <Latency typical="3.0" worst="5.0"/>
                </Performance>
            </Connection>
        </ConnectionSet>
    </FIFOCommunication>
    <FIFOCommunication dataSize="4" dataSizeUnit="B" queueSize="4" name="NIRouter">
        <ConnectionSet>
            <Connection from="MC_136065722366212016110410534719 6"
                  to="MC_116237769821020161104105347200">
```

```
                    <Performance>
                        <Pitch typical="3.0" worst="5.0"/>
                        <Latency typical="3.0" worst="5.0"/>
                    </Performance>
                </Connection>
                <Connection from="133699653783202016110415112625"
                      to="MC_191296276703542016110410534720 2">
                    <Performance>
                        <Pitch typical="3.0" worst="5.0"/>
                        <Latency typical="3.0" worst="5.0"/>
                    </Performance>
                </Connection>
                <Connection from="111807850454922016110415114 0002"
                      to="MC_159419980810872016110410534720 1">
                    <Performance>
                        <Pitch typical="3.0" worst="5.0"/>
                        <Latency typical="3.0" worst="5.0"/>
                    </Performance>
                </Connection>
                <Connection from="107241064153002016110415112 4164"
                      to="MC_176919068309342016110410534720 3">
                    <Performance>
                        <Pitch typical="3.0" worst="5.0"/>
                        <Latency typical="3.0" worst="5.0"/>
                    </Performance>
                </Connection>
            </ConnectionSet>
        </FIFOCommunication>
        <FIFOCommunication dataSize="4" dataSizeUnit="B" queueSize="1" name="RouterNI">
            <ConnectionSet>
                <Connection from="MC_116237769821020161104105347200"
                      to="MC_136065722366212016110410534719 6">
                    <Performance>
                        <Pitch typical="3.0" worst="5.0"/>
                        <Latency typical="3.0" worst="5.0"/>
                    </Performance>
                </Connection>
                <Connection from="MC_191296276703542016110410534720 2"
                      to="133699653783202016110415112625">
                    <Performance>
                        <Pitch typical="3.0" worst="5.0"/>
                        <Latency typical="3.0" worst="5.0"/>
                    </Performance>
                </Connection>
                <Connection from="MC_159419980810872016110410534720 1"
                      to="111807850454922016110415114 0002">
                    <Performance>
                        <Pitch typical="3.0" worst="5.0"/>
                        <Latency typical="3.0" worst="5.0"/>
                    </Performance>
                </Connection>
                <Connection from="MC_176919068309342016110410534720 3"
                      to="107241064153002016110415112 4164">
                    <Performance>
                        <Pitch typical="3.0" worst="5.0"/>
                        <Latency typical="3.0" worst="5.0"/>
                    </Performance>
                </Connection>
            </ConnectionSet>
        </FIFOCommunication>
    </CommunicationSet>
    <AddressSpaceSet>
        <AddressSpace name="AS_Core_0_0" id="AS_176650543626672016110410543847 7">
            <SubSpace name="TileLocalSubspace" id="SS_116444041347272016110410543847 7"
                  start="0" end="8388608" endian="BIG">
                <MemoryConsistencyModel rawOrdering="ORDERD" warOrdering="ORDERD"
                  wawOrdering="ORDERD" rarOrdering="ORDERD"/>
                <MasterSlaveBindingSet>
                    <MasterSlaveBinding slaveComponentRef="SC_858952163142620161104105347204">
                        <Accessor masterComponentRef="MC_611572016123020161104150944371">
                            <PerformanceSet>
                              <Performance accessTypeRef="AT_772698783293201611041509443 73">
                                  <Pitch typical="24.0" worst="26.0"/>
```

```
                            <Latency typical="24.0" worst="26.0"/>
                        </Performance>
                        <Performance accessTypeRef="AT_93629283199772016111016272305 7">
                            <Pitch typical="6.0" worst="8.0"/>
                            <Latency typical="6.0" worst="8.0"/>
                        </Performance>
                    </PerformanceSet>
                </Accessor>
            </MasterSlaveBinding>
        </MasterSlaveBindingSet>
    </SubSpace>
    <SubSpace name="ScratchpadSubspace" id="SS_1414744767681520161104142924026"
      start="8388608" end="8519680" endian="BIG">
        <MemoryConsistencyModel rawOrdering="ORDERD" warOrdering="ORDERD"
      wawOrdering="ORDERD" rarOrdering="ORDERD"/>
        <MasterSlaveBindingSet>
            <MasterSlaveBinding slaveComponentRef="SC_230643635746020161104105347208">
                <Accessor masterComponentRef="MC_611572016123020161104150944371">
                    <PerformanceSet>
                        <Performance accessTypeRef="AT_772698783293201611041509443 73">
                            <Pitch typical="8.0" worst="8.0"/>
                            <Latency typical="8.0" worst="8.0"/>
                        </Performance>
                        <Performance accessTypeRef="AT_93629283199772016111016272305 7">
                            <Pitch typical="2.0" worst="2.0"/>
                            <Latency typical="2.0" worst="2.0"/>
                        </Performance>
                    </PerformanceSet>
                </Accessor>
            </MasterSlaveBinding>
        </MasterSlaveBindingSet>
    </SubSpace>
    <SubSpace name="DDRSubspace" id="SS_1098568947728220161104142939959"
      start="8585216" end="4294967295" endian="BIG" contentType="DATA" >
        <MemoryConsistencyModel rawOrdering="ORDERD" warOrdering="ORDERD"
      wawOrdering="ORDERD" rarOrdering="ORDERD"/>
        <MasterSlaveBindingSet>
            <MasterSlaveBinding
                    slaveComponentRef="SC_203612783804292016110410534720 6">
                <Accessor masterComponentRef="MC_611572016123020161104150944371"
                    routedAccess="true" cacheBehaviour="UNCACHED" guarded="true" />
            </MasterSlaveBinding>
        </MasterSlaveBindingSet>
    </SubSpace>
    <SubSpace name="NIRegisters" id="SS_1719072416924920161121171615377"
          start="8519680" end="8519936" endian="BIG" contentType="DATA" >
        <MemoryConsistencyModel rawOrdering="ORDERD" warOrdering="ORDERD"
          wawOrdering="ORDERD" rarOrdering="ORDERD"/>
        <MasterSlaveBindingSet>
            <MasterSlaveBinding
                    slaveComponentRef="SC_153994703719052016112117202625 4">
                <Accessor masterComponentRef="MC_611572016123020161104150944371"/>
                <Accessor masterComponentRef="MC_136065722366212016110410534719 6"/>
            </MasterSlaveBinding>
        </MasterSlaveBindingSet>
    </SubSpace>
</AddressSpace>
<AddressSpace name="AS_Core_1_0" id="AS_771775563145920161104105438477">
    <SubSpace name="TileLocalSubspace" id="SS_161052599138942016110410543847 7"
      start="0" end="8388608" endian="BIG">
        <MemoryConsistencyModel rawOrdering="ORDERD" warOrdering="ORDERD"
          wawOrdering="ORDERD" rarOrdering="ORDERD"/>
        <MasterSlaveBindingSet>
            <MasterSlaveBinding
                    slaveComponentRef="SC_108941827220092016110410534720 5">
                <Accessor masterComponentRef="MC_589404864217201611041053471 98">
                    <PerformanceSet>
                        <Performance accessTypeRef="AT_143794106003572016111016283573 6">
                            <Pitch typical="24.0" worst="26.0"/>
                            <Latency typical="24.0" worst="26.0"/>
                        </Performance>
                        <Performance accessTypeRef="AT_121083041537452016111016283573 6">
                            <Pitch typical="6.0" worst="8.0"/>
```

```
                            <Latency typical="6.0" worst="8.0"/>
                        </Performance>
                    </PerformanceSet>
                </Accessor>
            </MasterSlaveBinding>
        </MasterSlaveBindingSet>
    </SubSpace>
    <SubSpace name="ScratchpadSubspace" id="SS_194481521813722016110414295655O"
              start="8388608" end="8519680" endian="BIG">
        <MemoryConsistencyModel rawOrdering="ORDERD" warOrdering="ORDERD"
              wawOrdering="ORDERD" rarOrdering="ORDERD"/>
        <MasterSlaveBindingSet>
            <MasterSlaveBinding
                      slaveComponentRef="SC_16361826552841201611041053472O8">
                <Accessor masterComponentRef="MC_589404864217201611041053471 98">
                    <PerformanceSet>
                        <Performance accessTypeRef="AT_1437941060035720161110162835736">
                            <Pitch typical="8.0" worst="8.0"/>
                            <Latency typical="8.0" worst="8.0"/>
                        </Performance>
                        <Performance accessTypeRef="AT_12108304153745201611101628357 36">
                            <Pitch typical="2.0" worst="2.0"/>
                            <Latency typical="2.0" worst="2.0"/>
                        </Performance>
                    </PerformanceSet>
                </Accessor>
            </MasterSlaveBinding>
        </MasterSlaveBindingSet>
    </SubSpace>
    <SubSpace name="DDRSubspace" id="SS_14975585323689201611041429565 0"
              start="8585216" end="4294967295" endian="BIG" contentType="DATA" >
        <MemoryConsistencyModel rawOrdering="ORDERD" warOrdering="ORDERD"
              wawOrdering="ORDERD" rarOrdering="ORDERD"/>
        <MasterSlaveBindingSet>
            <MasterSlaveBinding
              slaveComponentRef="SC_203612783804292016110410534720 6">
                <Accessor masterComponentRef="MC_589404864217201611041053471 98"
                      routedAccess="true" cacheBehaviour="UNCACHED" guarded="true" />
            </MasterSlaveBinding>
        </MasterSlaveBindingSet>
    </SubSpace>
    <SubSpace name="NIRegisters" id="SS_3323331218832016112117162836 2" start="8519680"
          end="8519936" endian="BIG" contentType="DATA" >
        <MemoryConsistencyModel rawOrdering="ORDERD" warOrdering="ORDERD"
              wawOrdering="ORDERD" rarOrdering="ORDERD"/>
        <MasterSlaveBindingSet>
            <MasterSlaveBinding
                      slaveComponentRef="SC_161413356304062016112117210188 6">
                <Accessor masterComponentRef="MC_589404864217201611041053471 98"/>
                <Accessor masterComponentRef="133699653783202016110415111262 5"/>
            </MasterSlaveBinding>
        </MasterSlaveBindingSet>
    </SubSpace>
</AddressSpace>
<AddressSpace name="AS_Core_1_1" id="AS_1596009860945420161104142719609">
    <SubSpace name="TileLocalSubspace" id="SS_1243495105730820161104142841080"
              start="0" end="8388608" endian="BIG">
        <MemoryConsistencyModel rawOrdering="ORDERD" warOrdering="ORDERD"
              wawOrdering="ORDERD" rarOrdering="ORDERD"/>
        <MasterSlaveBindingSet>
            <MasterSlaveBinding
                      slaveComponentRef="SC_184700847166732016110410534720 6">
                <Accessor masterComponentRef="MC_199785917102862016110410534719 9">
                    <PerformanceSet>
                        <Performance accessTypeRef="AT_8435127262095201611101629085 96">
                            <Pitch typical="24.0" worst="26.0"/>
                            <Latency typical="24.0" worst="26.0"/>
                        </Performance>
                        <Performance accessTypeRef="AT_7739899067265201611101629085 96">
                            <Pitch typical="6.0" worst="8.0"/>
                            <Latency typical="6.0" worst="8.0"/>
                        </Performance>
                    </PerformanceSet>
```

```
                    </Accessor>
                </MasterSlaveBinding>
            </MasterSlaveBindingSet>
        </SubSpace>
        <SubSpace name="ScratchpadSubspace" id="SS_594916129972020161104142959182"
              start="8388608" end="8519680" endian="BIG">
            <MemoryConsistencyModel rawOrdering="ORDERD" warOrdering="ORDERD"
                  wawOrdering="ORDERD" rarOrdering="ORDERD"/>
            <MasterSlaveBindingSet>
                <MasterSlaveBinding
                        slaveComponentRef="SC_193283145014912016110410534720 9">
                    <Accessor masterComponentRef="MC_199785917102862016110410534719 9">
                        <PerformanceSet>
                          <Performance accessTypeRef="AT_84351272620952016111016290859 6">
                              <Pitch typical="8.0" worst="8.0"/>
                              <Latency typical="8.0" worst="8.0"/>
                          </Performance>
                          <Performance accessTypeRef="AT_77398990672652016111016290859 6">
                              <Pitch typical="2.0" worst="2.0"/>
                              <Latency typical="2.0" worst="2.0"/>
                          </Performance>
                        </PerformanceSet>
                    </Accessor>
                </MasterSlaveBinding>
            </MasterSlaveBindingSet>
        </SubSpace>
        <SubSpace name="DDRSubspace" id="SS_153791239658562016110414295918 2"
                  start="8585216" end="4294967295" endian="BIG" contentType="DATA" >
            <MemoryConsistencyModel rawOrdering="ORDERD" warOrdering="ORDERD"
                  wawOrdering="ORDERD" rarOrdering="ORDERD"/>
            <MasterSlaveBindingSet>
                <MasterSlaveBinding
                        slaveComponentRef="SC_203612783804292016110410534720 6">
                    <Accessor masterComponentRef="MC_199785917102862016110410534719 9"
                          routedAccess="true" cacheBehaviour="UNCACHED" guarded="true" />
                </MasterSlaveBinding>
            </MasterSlaveBindingSet>
        </SubSpace>
        <SubSpace name="NIRegisters" id="SS_33269994999732016112117163024 8"
              start="8519680" end="8519936" endian="BIG" contentType="DATA" >
            <MemoryConsistencyModel rawOrdering="ORDERD" warOrdering="ORDERD"
                  wawOrdering="ORDERD" rarOrdering="ORDERD"/>
            <MasterSlaveBindingSet>
                <MasterSlaveBinding slaveComponentRef="SC_178604517211320161121172120341">
                    <Accessor masterComponentRef="MC_199785917102862016110410534719 9"/>
                    <Accessor masterComponentRef="107241064153002016110415112416 4"/>
                </MasterSlaveBinding>
            </MasterSlaveBindingSet>
        </SubSpace>
    </AddressSpace>
    <AddressSpace name="AS_MemoryControlCore_0_1" id="AS_81749065336342016110414272140 8">
        <SubSpace name="TileLocalSubspace" id="SS_20087466773524201611041428575 1"
                  start="0" end="8388608" endian="BIG">
            <MemoryConsistencyModel rawOrdering="ORDERD" warOrdering="ORDERD"
                  wawOrdering="ORDERD" rarOrdering="ORDERD"/>
            <MasterSlaveBindingSet>
                <MasterSlaveBinding slaveComponentRef="SC_257608164903520161104105347207">
                    <Accessor masterComponentRef="MC_959869407870920161104105347200">
                        <PerformanceSet>
                          <Performance accessTypeRef="AT_75200156715722016111016293538 6">
                              <Pitch typical="24.0" worst="26.0"/>
                              <Latency typical="24.0" worst="26.0"/>
                          </Performance>
                          <Performance accessTypeRef="AT_77737908462702016111016293538 6">
                              <Pitch typical="6.0" worst="8.0"/>
                              <Latency typical="6.0" worst="8.0"/>
                          </Performance>
                        </PerformanceSet>
                    </Accessor>
                </MasterSlaveBinding>
            </MasterSlaveBindingSet>
        </SubSpace>
        <SubSpace name="DDRSubspace" id="SS_55770592264092016110414301929 6"
```

```
                    start="8585216" end="4294967295" endian="BIG" contentType="DATA" >
                <MemoryConsistencyModel rawOrdering="ORDERD" warOrdering="ORDERD"
                    wawOrdering="ORDERD" rarOrdering="ORDERD"/>
                <MasterSlaveBindingSet>
                    <MasterSlaveBinding
                            slaveComponentRef="SC_203612783804292016110410534720 6">
                        <Accessor masterComponentRef="MC_95986940787092016110410534720 0"
                            cacheBehaviour="CACHED" >
                          <PerformanceSet>
                            <Performance accessTypeRef="AT_75200156715722016111016293538 6">
                                <Pitch typical="32.0" worst="42.0"/>
                                <Latency typical="32.0" worst="42.0"/>
                            </Performance>
                            <Performance accessTypeRef="AT_77737908462702016111016293538 6">
                                <Pitch typical="12.0" worst="25.0"/>
                                <Latency typical="12.0" worst="25.0"/>
                            </Performance>
                          </PerformanceSet>
                        </Accessor>
                    </MasterSlaveBinding>
                </MasterSlaveBindingSet>
            </SubSpace>
            <SubSpace name="NIRegisters" id="SS_53940034768320161121171632152" start="8519680"
                    end="8519936" endian="BIG" contentType="DATA" >
                <MemoryConsistencyModel rawOrdering="ORDERD" warOrdering="ORDERD"
                    wawOrdering="ORDERD" rarOrdering="ORDERD"/>
                <MasterSlaveBindingSet>
                    <MasterSlaveBinding slaveComponentRef="SC_48230769855062016112117213202 3">
                        <Accessor masterComponentRef="MC_19978591710286201611041053471 99"/>
                        <Accessor masterComponentRef="10724106415300201611041511241 64"/>
                    </MasterSlaveBinding>
                </MasterSlaveBindingSet>
            </SubSpace>
        </AddressSpace>
    </AddressSpaceSet>
    <ClockFrequency clockValue="2.5E7"/>
</SystemConfiguration>
```

List of Figures

Glossary of Terms

ADL	Architecture Description Language
AMBA	Advanced Microcontroller Bus Architecture
API	Application Programming Interface
CPU	Central Processing Unit
FIFO	First-In First-Out
IR	Intermediate Representation
ISA	Instruction set architecture
NoC	Network on Chip
NUMA	Non-Uniform Memory Access
SDE	FlexaWare software development environment
SIMD	Single Instruction Multiple Data
TU	Transfer Unit
WCET	Worst-Case Execution Time