A Simulation Environment for Design Space Exploration for Asymmetric 3D-Network-on-Chip

Jan Moritz Joseph*, Sven Wrieden†, Christopher Blochwitz†, Alberto García-Oritz‡, Thilo Pionteck*

*Otto-von-Guericke-Universität Magdeburg
Institut für Informations- und Kommunikationstechnik, 39106 Magdeburg, Germany
Email: {jan.joseph, thilo.pionteck}@ovgu.de

†Universität zu Lübeck
Institut für Technische Informatik, 23562 Lübeck, Germany
Email: {wrieden, blochwitz}@iti.uni-luebeck.de

‡University of Bremen
Institute of Electrodynamics and Microelectronics, 28359 Bremen, Germany
Email: agarcia@item.uni-bremen.de

***Abstract*—We present a comprehensive simulation environment for design space exploration in Asymmetric 3D-Networks-on-chip (A-3D-NoCs) covering the heterogeneity in 3D-System-on-chips (3D-SoCs). A challenging aspect of A-3D-NoC design is the consideration of interwoven parameters of the communication infrastructure and characteristics of the manufacturing technologies. Thus, simultaneous evaluation of multiple design metrics is mandatory. Our simulation environment consists of three parts. First, it comprises a NoC simulator that supports a multitude of different manufacturing technologies, router architectures, and network topologies within a single design. As a key feature, the NoC and technologies parameters per chip layer are fully configurable during simulation runtime permitting flexible and fast evaluation. Second, a central reporting tool facilitates system analysis on different abstraction levels. Third, the evolution tool provides various synthetic and real-world based benchmarks. Thus, our tool allows for an incremental approach to systematically explore the A-3D-NoC's design space.**

***Keywords*—Asymmetric 3D-Network-on-Chip, Simulation Environment, SystemC, Benchmarking, Design Space Exploration**

I. Introduction

Asymmetric 3D Networks-on-chip (A-3D-NoCs) are *the* on-chip communication method targeting today's heterogeneous 3D-System-on-Chips (3D-SoCs) [1]. The design space exploration of A-3D-NoCs is more demanding than traditional (2D- and 3D-) NoC design due to the manifold challenges: First, the design space itself is larger in comparison to homogeneous SoCs since not only the parameters of the network but also characteristic of the manufacturing technologies of each individual chip layer are to be considered. Second, the traditional design metrics, area, power, and performance, are interwoven in A-3D-NoCs. As an example, it was shown in [1] that the overall area footprint of the NoC's memory and power consumption of the routers can be reduced via buffer reorganization among dies. This reduction is tightly coupled to a decrease of performance. However, the buffer reorganization effects the structural metrics more than the performance and, thus, this technique offers an acceptable compromise from a system perspective. Third, and in consequence, a simultaneous evaluation of multiple design metrics is mandatory for A-3D-NoC design. Neither practical approaches to the systematical design space exploration nor tools for simulative evaluation and verification exist for design guidance in case of A-3D-NoCs. For instance, NoC simulators such as Noxim [2] and Booksim [3] offer diverse NoC features and evaluation methods. Extending these to a third dimensions is, in general, straight forward. Still, these tools do not support heterogeneous technologies and, thus, asymmetric NoC designs.

Our A-3D-NoC simulation environment is capable of simulative evolution and verification of communication infrastructures for heterogeneous 3D-SoCs. It consists of three parts to tackle the aforementioned challenges in design space exploration. First, it comprises an A-3D-NoC simulator in SystemC. It supports a multitude of different manufacturing technologies, router architectures, and network topologies. Within a single design the network's and technology's parameters are configurable prior to simulation for flexible and fast evaluation. Second, we provide a central reporting tool. This facilitates system analysis on different abstraction levels and is highly relevant to evaluate interwoven design metrics. Third, the evolution tool provides various synthetic and real-world based benchmarks. Thereby our environment covers many use cases and provides comparable performance results. Summing, our tool allows for an incremental approach to systematically explore the A-3D-NoC's design space.

This paper is structured as follows: In Sec. II, an overview is given about heterogeneous 3D-SoCs and A-3D-NoCs, on systematical design space exploration with tool guidance and on popular NoC simulators. Then, in Sec. III, we explain, why incremental approaches are relevant for design space exploration in A-3D-NoCs and which requirements yield for simulation environments. Afterwards, in Sec. IV, our simulation environment and its tool flow are introduced. Before concluding we analyze the simulation environment's capabilities with focus on performance and design time in Sec. V.

II. Related work

Through new production methods the fabrication of heterogeneous 3D-System-on-chips (SoCs) is possible. They are a promising approach to reduce the challenges of today's chip design with 2D-architectures: In general, 3D-chips offer reduced costs and power consumption, and the performance is increased [4]. In addition, 3D-manufacturing facilitates heterogeneous iterated chips, which consist of stacked silicon dies manufactured in different technology nodes interconnected by Through-Silicon Vias (TSVs). Thus, die layers can be optimized for the electrical and functional requirements for its components: In [5] a heterogeneous 3D-vision chip is proposed, in which a conservative mixed-signal technology is connected to a high-speed digital technology. Moreover, in [6], layers dedicated for memory are connected to layers, which are optimized for processing units, resulting in improved energy efficiency. In addition, a wide variety of designs has been proposed. Prominent examples are 3D-FPGAs [7] and 3D-DRAM subsystems [8].

Since 3D-SoCs comprise an even larger number of components than 2D-designs, a scalable communication infrastructure is required. 3D-NoCs meet this requirement and provide additional advantages such as high throughput and flexibility [9]. NoCs allow for straight-forward integration of routers running at different clock speeds. This requirement is critical for heterogeneous 3D-chips since die layers in different technology nodes are part of varying clock regions. An individual technology-aware design of NoC routers per die offers large optimization potential. These on-chip networks are called **Asymmetric 3D-NoCs (A-3D-NoCs)** [1]. In this work, optimizations for A-3D-NoC router architectures are proposed which cover buffer reorganization among dies with focus on power and area savings. In comparison to conventional, symmetric 3D-NoC routers, area savings of 8.3% and power savings of 5.4% for link buffers are achieved along with a small average system performance loss of 2.1%.

Systematical design space exploration in NoCs is a highly relevant topic since the size of the design space is large. Multiple linked design metrics increase the complexity. In consequence, most methods for design space exploration focus on a subtopic such as application specific designs or limit their approach to a subset of design metrics. For instance, one method is the mathematical analysis of the optimization problem. In [10], principal component analysis is used to find appropriate parameter values for the components. In another technique, the design space is reduced in size via early-stage exploration using a power and area model with high accuracy [11]. Thus, large parts of the design space can be disregarded at an early on in NoC design.

Since network performance is a main design metric, numerous NoC simulators have been proposed. The capabilities of common simulators such as Booksim [3] and Noxim [2] are mature and the tools offer various features. Most often, they are implemented in C++ and SystemC. Among their features are customizations of network parameters, e.g. sizes of packets and router buffers depths, 2D-topologies, exchangeable and parametrized routing algorithms, and synthetic traffic patterns for benchmarks. They allow for the evaluation of NoC design metrics. The performance, i.e. throughput or latency, and energy consumption are estimated. Routers in both simulators are modeled at a low level of abstraction. For instance, traffic is generated at the network interfaces of processing elements (PEs). Other simulators have been proposed covering a wider range of abstraction levels [12], [13] for a comprehensive design space exploration. These interconnect cycle-accurate router models with application or system simulation via transaction-level modeling (TLM).

All of the previous listed simulators focus on 2D-NoCs. In general, they allow for extension to 3D-SoCs. There are also simulators specifically for 3D-NoCs designs, e.g. [14]. However, these simulators do not target technology-specific features of dies. To the best of our knowledge, the only work including this aspect is [15], in which a standard router is divided and implemented on multiple die layers. This work also comprises a simulator, which is used for simulative evaluation with both synthetic and application specific benchmarks also considering dynamic shutdown of components for power savings. This simulator is specifically implemented for this router design and lacks of generality. Though, it is crucial for the evaluation of A-3D-NoCs to have a general NoC simulator including manufacturing technology specific features of dies.

Summing up, our simulation environment's scope is orthogonal to all existing software tools and is the first simulation environment for systematical evaluation of A-3D-NoCs: It provides dynamically configurable asymmetric 3D-NoC simulation comprising technology-specific parameters. In addition, the simulator covers multiple layers of abstraction, since cycle-accurate routers are implemented beside transaction-level-modeled applications. This allows for manifold benchmarks and system configurations. Moreover, a reporting tool for detailed evaluation is provided.

III. Design Space Exploration and Simulation Environment's Requirements

The design space exploration in A-3D-NoCs is especially challenging. This is the result of two main characteristics. Already during early stages of the NoC design, *technology parameters must be considered*. These parameters comprise, for instance, different area footprints, power consumption, and thermal emissions of structurally identical layouts which are implemented in different technology nodes. Furthermore, depending on the technological parameters, optimizations during floorplanning can be more or less efficient. As an example, in [1] was demonstrated that area and power are more impacted than the NoC's performance for asymmetric buffer distributions in heterogeneous 3D-SoCs. Thus, without considering the manufacturing technological's characteristics, designs with negative performance influence would have been discarded at an early stage, although they offer a good compromise when the three metrics are considered simultaneously. This introduces the second hurdle which appears during the

evaluation phase: *Interwoven metrics that must be considered simultaneously*. As another example, this effect appears for asymmetric buffer depths, as well. In general, reduced buffer depths increase the network latency especially for high loads. However, in case the memory area is very expensive for certain chip layers, reduced buffer depths nonetheless are promising to reduce the chip's costs. To successfully evaluate this, a third challenge must be tackled: *The demand for highly flexible evaluation platforms*, since multiple manufacturing technological, network methodical, and different topological design approaches must be concurrently considered.

These characteristics of the design space in A-3D-NoCs leads to the following requirements for a simulation environment:

1) Technological parameters must be included in the simulation.
2) Methods must be provided to retrospectively add initially unknown parameters to comprise the effects of interwoven metrics.
3) The simulation tool must provide a parametrized and configurable simulator, whose parameters are set during initialization to reduce unnecessary compile times and meta programming. Thus, the requirement for flexible evaluation platforms due to large architectural varieties is satisfied.
4) The tool must cover multiple levels of abstraction to cover both effect of transitions on the system level but also consequences of multiple, cycle-accurate router models. On an even lower level it must also incorporate the possibility to model electrical characteristics of TSVs.
5) Simultaneous evaluation of linked design metrics is essential during the evaluation. Therefore, interconnected design tools and an integrated tool flow are required.

A simulation environment that incorporates these features enables an *incremental approach* during design space exploration. In this method, the level of detail is gradually raised to tackle the difficulties in A-3D-NoC design targeting heterogeneous 3D-SoCs. The design space is iteratively explored with different sets of fixed and variable parameters, in which the fixed parameters define the position in the current design space (i.e. the level of detail), and the variable parameters set its size and allow for its exploration. After the metrics are calculated via simulation and synthesis for each set of parameters, the results are evaluated and the parameters can be adopted for the next iteration. In best case, the results indicate, how to increase the level of detail. Otherwise, the level of detail is fixed and different variable parameters are evaluated. The method already proved itself useful in [1].

IV. Simulation Environment

A. Tool Flow

The tool flow of our simulation environment is shown in Fig. 1. It also comprises the three central parts of the environment, the *NoC simulator*, the *benchmarking tool*, and the *reporting tool*.

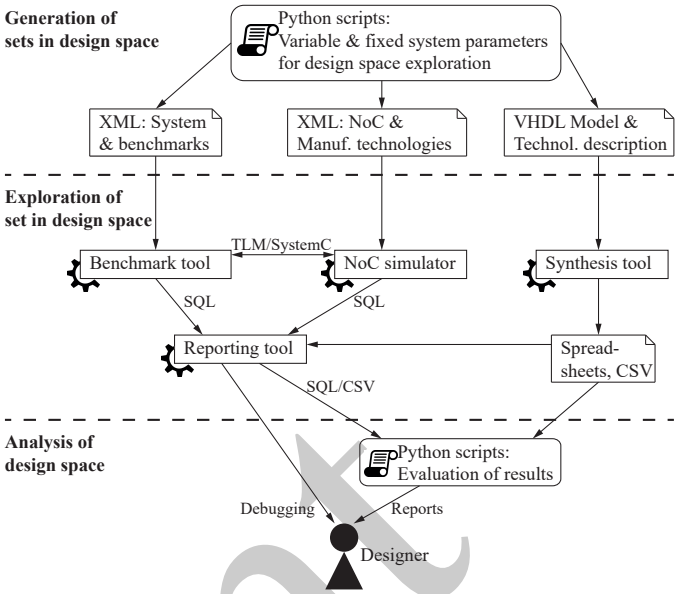

Fig. 1: Tool flow of simulation environment

The A-3D-NoC design space exploration starts at a given level of detail, which is chosen by the user according to an incremental approach. The parameters of this level are passed to initialization scripts written in Python as shown at the top of Fig. 1. It consists of the fixed system parameters, which define the level of detail, and the variable parameters, which define the size of the design space. The scripts generate three sets of input data for the simulation environment as shown in first section of the Fig. 1. First, depicted on the left-hand side, there are the XML description files for the *benchmarking tool*. These files contain information about the number and type of accessible processing elements, the type of traffic pattern, application, or software and, if necessary, the task or function mapping. Second, in the middle part of Fig. 1, the XML description for the *NoC simulator* is shown. It contains information about the parameters of the manufacturing technologies per layer. Furthermore, these files comprise the NoC's size, layout, topology, router model, router parameters, etc. of each individual network layer. Third, as shown on the right-hand side of Fig. 1, there are script files that select the correct VHDL models of the routers and assign the parameters of the manufacturing technology.

After this step is completed, the simulation and syntheses are started to explore the design space. Both are shown in the second section of Fig. 1. The *NoC simulator* is launched for every set of parameters as defined in the design space exploration using the selected benchmarks. The simulator and the *benchmark tool* communicate on Transaction-modeled level (TLM) in SystemC. Furthermore, the *reporting tool* with its SQL database is connected to the benchmark tools and the NoC simulator. The SQL interface is encapsulated in C++ and collects data about any type of event from the NoC simulation and benchmark generation depending on the level of reporting abstraction, which was set by the user.

Furthermore, the script files invoke a standard cell synthesis. The output is written into spreadsheets and CSV files. As a special feature of the reporting tool, parts of the data in the database can retrospectively be modified to incorporate results from the synthesis.

Finally, as shown in the lowermost section in Fig. 1, after the software executed the design space exploration for each set of input parameters, reports are generated by evaluation scripts for the designer. If required, debugging information can be received from the reporting tool or traditional `vcf`-files. For the subsequent iteration, the designer can analyze the results and adopt the input parameters for the next step following the particular incremental approach.

B. Abstraction Levels and Communication

The simulation environment covers multiple levels of abstraction to evaluate performance and energy consumption of A-3D-NoC models. An overview about the components of the environment and their abstraction levels is given in Fig. 2. On the bottom of the Fig. 2 the NoC simulator and its components are shown. Routers communicate via flits. These are generated in the network interface of processing elements from network packets. All these components and their transmitted messages are modeled on cycle-accurate abstraction, which is the lowest abstraction level here. The processing elements connect via a technology-aware TLM interface to the benchmark tool. It encapsulates the NoC's properties and, thus, closes the gap between system simulation and NoC router evaluation. This component is critical since it abstracts the technological heterogeneity of the 3D-SoC for each features, which is relevant form a system point of view. The SoC's heterogeneity in terms of router features is already included in the cycle-accurate router models and, thus, not relevant in this scope. In this component, the level of abstraction is raised up to TLM in the benchmark tool. This ensures a high performance of the simulator while generating significant results.

C. NoC Simulator

The simulator for A-3D-NoCs in our simulation environment comprises router models, the network interfaces of the processing elements, and a basic PE model. The components are shown at the bottom of Fig. 2 and are modeled on cycle-accurate abstraction. The PE model consists of the interface for the technology-aware TLM interface, and a queue for outgoing packages. The queue is processed in the given clock speed for PEs. The network interface translates network packages into flits and vice versa and, thus, manages the states of the outgoing queues of the PEs. It allows for clock asymmetry. The network interfaces and routers are using the PE address space.

The simulator is configurable during initialization and supports a variable number of layers, of which the structure is parsed from XML files during the instantiation of the simulator. The structure of the XML file is given in Fig. 3. In general, the routers can be connected following variable topologies. The simulator supports standard topologies for

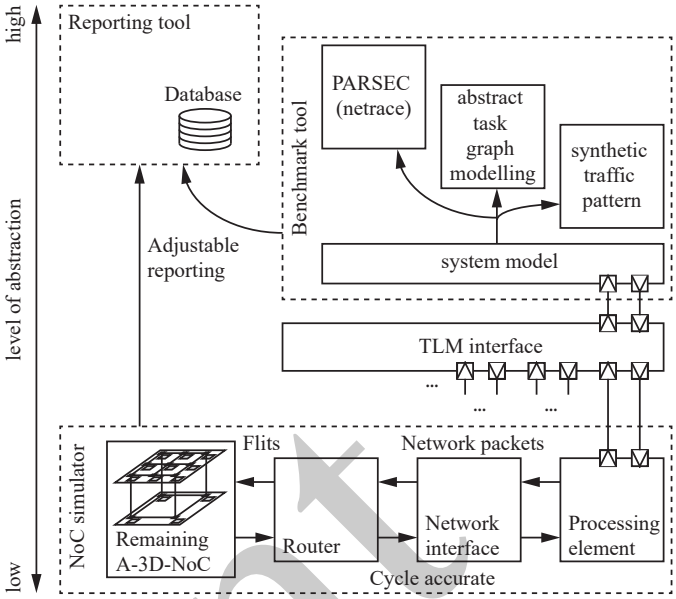

Fig. 2: Abstraction levels and communication

intra-layer connections such as 2D-mesh, which are configured using the `topology` field. Arbitrary layouts are omitted since these are not practical due to complications in design and routing algorithms. Inter-layer connections, which are implemented using TSV models, can be freely configured using the `layerInterconnect` fields in the configuration file. Here, any upper and lower node can be connected using their coordinates with a x-, y-, and z- component. These free interconnection schemes are crucial since a common practice in 3D-NoC design is the reduction of the number of TSVs due to their individual high costs. Thus, our framework also covers network optimizations via TSV serialization and partial connections.

Routers are connected among each other and to PEs via asynchronous Fifo models. This is required, since heterogeneous layers may have independent clocks. Thus, different clock regions within a layer are also supported. In addition, the processing elements themselves are in a different clock domain than the routers, as well. Within the description file, the clock frequencies are set via the `clock speed` values in the router and processing element field. The buffer depth of the routers can be configures at runtime using the `bufferDepth` field in the router configuration.

The router architecture can be modeled based on an existing router class, which comprises the router Fifos, a receive routine, and basic routing algorithms. The routing algorithm can be exchanged. In general, it is possible to implement different routing algorithms in each layer, which is a promising feature of A-3D-NoCs. For instance, layers with lower costs (i.e. in a more advanced technological node) are more prone to errors and thus may implement error correction methods in their routing algorithms. To implement different types of router models, the send routine can be modified. For instance, we implemented a standard three and four stage router model [16],

```
<network-on-chip>
    <layer>
        <name value="high speed layer"/>
        <technology value="130"/>
        <topology value="mesh"/>

        <router>
            <bufferDepth value="7"/>
            <clockSpeed value="8"/>
            <routerModel value="normal"/>
        </router>

        <processingElements>
            <clockSpeed value="8"/>
        </processingElements>

        <x_nodes value="4"/>
        <y_nodes value="4"/>
    </layer>

    <layer> ... </layer>
    ...

    <layerInterconnects>
        <con lowerNode="0 0 0"
             upperNode="0 0 1"/>
        <con lowerNode="0 1 0"
             upperNode="0 1 1"/>
        ...
    </layerInterconnects>
</network-on-chip>
```

Fig. 3: Exemplary layer description file

and an architectural model of a router, which accelerates semi-static data streams via pre-allocation of prioritized paths. Moreover, interfaces are provided to connect technological TSV-models. Using this features, the simulator comprises realistic TVS models which consider the TSV sizes, their geometry, electrical properties, and delays. In a concurrent work, these TSV models are prototyped using MATLAB. As future work, we will implement them in SystemC as well to include them into the NoC simulator. Since SystemC even allows for VHDL co-simulation, a very detailed design space exploration of TSV characteristics is enabled by our simulation framework. Furthermore, we will also model temperature properties as future work as it is an important issue of 3D-ICs.

D. Technology-aware TLM interface

The NoC simulator and the benchmarks tools operate on different levels of abstraction. An intermediate TLM interface encapsulates this abstraction level transition as shown in the middle of Fig. 2. This is done by translation of packages between tasks or units into network packages, which operate on cycle-accurate abstraction level. This functionality is already described in [13]. In addition, the TLM interface's functionality is extended to support technology parameters. Thus, it also encapsulates system characteristics of the network: The technology-aware TLM interface comprises structural parameters such as the PE model. It also contains link widths, number of flits per packet and converts the address spaces from system level to network level.

E. Benchmarking tool

The benchmarking tool offers three benchmark options as shown in Fig. 2: PARSEC benchmarks, abstract task graph models, and synthetic traffic pattern. These are ordered by declining level of abstraction. The task or application address space is used here. The PARSEC benchmarks [17] are currently realized using the netrace libray [18]. Since these trace files are tailored for 2D-NoCs, their validity for A-3D-NoCs is restricted. Thus, as future work, Gem5 co-simulation with heterogeneous 3D-SoC models is planned. The tasks in the abstract task graph model use TLM-interfaces for communication. Our task model was described in detail in [13]. It supports different types of applications such as multimedia codecs, the multi-window application, and the picture-in-picture application. Finally, the synthetic traffic pattern are generated on a cycle-wise scheme to avoid errors from traffic bursts. The benchmark tool offers uniform random, complement, and transpose traffic pattern. These pattern are adopted to A-3D-NoCs to cover irregular topologies (i.e. a different number of nodes per layer).

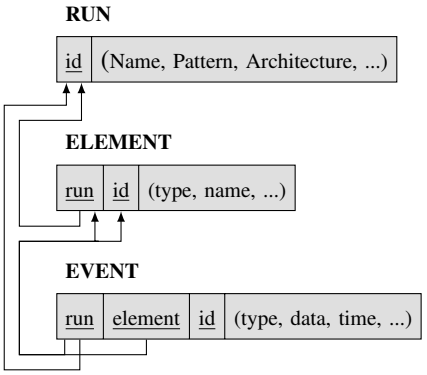

Fig. 4: Database structure in the reporting tool

F. Reporting Tool

The reporting tool consists of a MySQL database and C++ classes, which encapsulate the SQL commands. The database is connected via network (Ethernet) and can be executed on an external server for reduced performance impact in the simulations. The database uses a generic approach. This allows for rapid development of the simulator without compatibility issues. In addition, the reporting tool can be connected to an FPGA-based prototype of the NoC using a PCIe connection between host workstation and the extension card without any modifications in the reporting tool.

1) Database Structure: The SQL database holds three tables named *Run*, *Element*, and *Event*. The structure is also shown in Fig. 4. In the *Run*-Table an ID and various attributes such as the benchmark, its parameters, information about the router architectures, and general settings of the simulation are stored.

In each run of the simulation, multiple elements are created. In the *Element*-table individual routers, processing elements, but also structural components, like a traffic generators, are stored. Each element has a unique identifier based on a associated ID and the run's ID. In addition, elements also have

an assigned type and name field for easy filtering of specific elements.

The *Event*-table contains the actual simulation data. A write access is triggered by an element in the simulation. Events are stored with an associated time stamp, type of event, and a data field, which can range from a simple debug text to references of other elements. Thus, a very flexible reporting is possible. For instance, the trace of a single packet can be generated. Every router and PE, which transmits packet, stores an event for receiving and sending to the *Event*-table.

2) Reports: The simulation environment generates reports for the designer. In general, there are three types of data access. First, the evaluation scripts generate spreadsheets about the design space exploration including the most relevant parameters of the simulation. This process is partly automated: The designer can choose, which metrics are relevant, and the scripts automatically include those into the result files. Second, a flexible graphical user interface is provided. This dashboard is focused on graphical representation of relevant data from the reporting tool and is implemented in HTML5. Thus, it can be accessed via any standard web browser. As an example, a screen shot of the dashboard is shown in Fig. 5. Here, the average flit latency of a $2 \times 8 \times 8$-NoC with uniform traffic pattern is presented. It converges to the final value already after 1,200 simulated clock cycles. The user interface also prints the final results for average and maximum latency in clock cycles. The dashboard is customizable: The frames can be moved and adjusted in size. In addition, the user can select the scope and type of data, which are shown. Thus, the designer can easily access any relevant metric from the simulation. Third, the data, which are stored in the database can be directly accessed using SQL. This is useful in case the level of detail provided by the other two options is not high enough. This method can be used for debugging as well since it allows to trace moving objects in the network associated with advanced conditions to find error states. This extends the functionality of traditional `vcf`-files because these are constricted to functional units such as routers or PEs. Our design offers many advantages since it allows for report generation with great flexibility and retrospective data manipulation. However, even a short simulation generates gigabytes of data, if for instance each hop of every flit is registered and indexed. Since this function is not required for design space exploration but only for debugging, reporting of events can be exuded by simple flags in the XML configuration files. This features significantly increases the solution's performance.

V. Results

A. Performance of the reporting tool

In order to demonstrate the performance of the database, we consider the same simulation of a $2 \times 8 \times 8$-NoC with uniform random traffic, which was also used to generate Fig. 5. The database holds approx. 3 million events, of which 500,000 are send and receive events of flits at network interfaces (*SEND_FLIT, RECV_FLIT*). To compute the average time that each flit travels, the query listed in Fig. 6 was used.

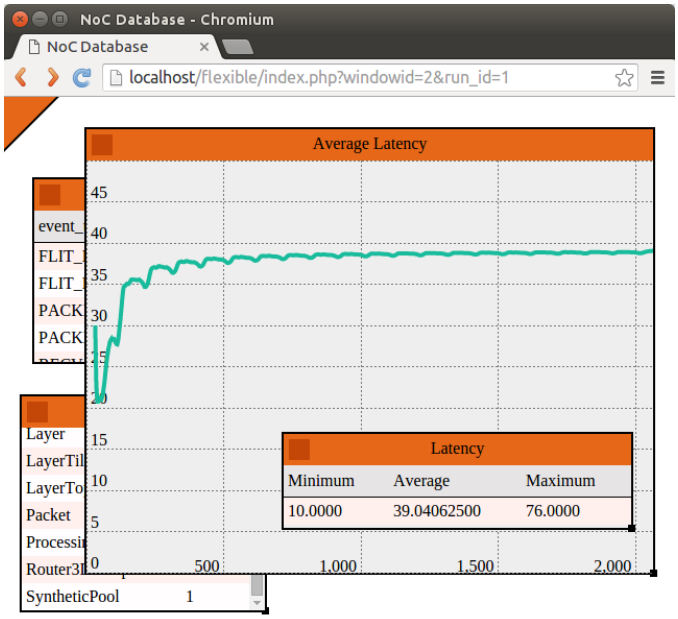

Fig. 5: Exemplary GUI

Since this is a common metric, it is a standard part of the reporting tool. Using this, the performance of the database query can be estimated. The execution time is about ten seconds on a server with an Intel Xeon E3-1230, running Ubunutu Server 14.04.4 LTS and MySQL Server 5.6, and grows linear with the count of the two relevant events. On the upside, the reporting tool is flexible. This is crucial for an incremental approach with interwoven design metrics. However, on the downside, this approach is considerably slower in comparison to an implementation which simply adds up the delays during simulation runtime. We also benchmarked our approach against traditional `vcf`-files, in which every router generated a single file with all relevant model parameters. The performance impact of writing the `vcf`-files to the hard disk only became relevant for very large NoCs. The reporting tool circumvents this effect since the database can be stored on an external server. Summing up, it is advantageous to reduce the amount of events, which are stored in the database to provide the advantages of our solution and maintain a satisfying performance.

B. Design Space Exploration Time

This simulation environment was used to assist during an incremental design space exploration as part of our work on area and power savings in A-3D-NoCs [1]. We systematically explored different buffer depths and buffer reorganization among dies with two different commercial technologies. The simulators allows to gradually increase the level of detail. For a fast initial evaluation, we benchmarked data streams generated with abstract task graph models such as video object plane decoder applications and MPEG-4, mp3, and h.256 converters from [19]. The main advantage of abstract task graphs is the manageable location of data streams. Moreover,

```
SELECT avg(diff)
FROM (
SELECT
    sum(CASE WHEN
        event_type='SEND_FLIT'
        THEN
        -1 * event_time
        ELSE
        event_time END)
    AS diff
FROM event
WHERE run_id = 1
    AND event_type
    IN ('SEND_FLIT','RECV_FLIT')
GROUP BY element_id
) table
```

Fig. 6: Example query to compute average flit delay

heterogeneous 3D-SoC models can be evaluated as well using standard task mapping methods. On the downside, these models lack of generality since they are application specific. These simulations run between 20 seconds and 1 minute on an workstation with an Intel Core i7-4770 CPU and 16 GB RAM using CentOS 7.1. Thus, the designer gets a report in less than five minutes and can modify the system parameters if necessary. Applying this method, unnecessarily large buffer sizes could be excluded from our further evaluation in the aforementioned work [1]. After finding parameters that are candidates for a more detailed evaluation, it is conducted using synthetic traffic patterns. For instance, the simulation of uniform random traffic pattern with a NoC consisting of 32 routers and PEs and injection rates between 20% and 95% with 32 restarts per injection rate for sufficient statistical stability takes between 1.5 and 2 hours per parameter set. With these data the designer can find further appropriate candidates of parameters for a final evaluation using real-world based benchmarks such as PARSEC benchmarks using either Gem5 co-simulation of trace files for instance from the Netrace library. For a single set of parameters, the simulation of 25 ms of CPU time in the PARSEC region of interest (ROI) from trace files consumes approx. 41 hours on our workstation. Since Netrace comprises eight benchmarks, this evaluation method is very time consuming and should only be conducted at the end of the incremental design space exploration with a very limited set of parameters. In general, a further speedup can be achieved using multiple CPU cores simultaneously. Due to restrictions of SystemC, the simulator is only threaded on a single core. However, parallelization of simulation runs is possible. The simulation time did not scale linearly, which is an effect of Intel's "turbo" modes and cache misses.

This exemplary case study demonstrates another advantage of our environment as well, since we are able to analyze interwoven design metrics. As explained, the performance of the NoC was evaluated using our NoC simulator. In addition, we estimated the area costs and power consumption of our design. This was done using VHDL models of the router and synthesis with Synopsys. Using Python scripts, the characteristics of the manufacturing technologies could be passed between the simulator and the synthesis. Thus, the designer receives an comprehensive overview about the systems figures during every iteration of the design space exploration.

VI. Conclusion

A comprehensive A-3D-NoC simulation environment is introduced for design space exploration targeting communication infrastructures in heterogeneous 3D-SoCs. The tool flow enables the consideration of interwoven parameters of the communication infrastructure and characteristics of the manufacturing technologies. Simultaneous evaluation and retrospective manipulation of design metrics is possible. The simulation environment consists of three parts: An A-3D-NoC simulator, which allows for flexible NoC configuration, a benchmark tool with a wide variety of evaluation methods, and a reporting tool for flexible measurement. Our simulation environment's scope is orthogonal to all existing software tools and is the first simulation environment for systematical evaluation of A-3D-NoCs. It enables incremental approaches for design space exploration.

Acknowledgements

This work is funded by the German Research Foundation (DFG) project PI 447/5-1.

References

[1] J. Joseph, C. Blochwitz, A. Garcia-Ortiz, and T. Pionteck, "Area and power savings via buffer reorganization in asymmetric 3d-nocs for heterogeneous 3d-socs," in *IEEE Nordic Circuits and Systems Conference (NORCAS)*, 2015.

[2] F. Fazzino, M. Palesi, and D. Patti, "Noxim: Network-on-chip simulator," *URL: http://sourceforge. net/projects/noxim*, 2008.

[3] N. Jiang, G. Michelogiannakis, D. Becker, B. Towles, and W. Dally, "Booksim interconnection network simulator," *Online, https://nocs. stanford. edu/cgibin/trac. cgi/wiki/Resources/BookSim*.

[4] X. Dong and Y. Xie, "System-level cost analysis and design exploration for three-dimensional integrated circuits (3d ics)," 2009.

[5] Á. Zarándy, *Focal-plane sensor-processor chips*. Springer, 2011.

[6] X. Yu, L. Li, Y. Zhang, H. Pan, and S. He, "Performance and power consumption analysis of memory efficient 3d network-on-chip architecture," *ICCA*, 2013.

[7] V. F. Pavlidis and E. G. Friedman, *Three-dimensional Integrated Circuit Design*. Elsevier Science, 2010.

[8] C. Weis, I. Loi, L. Benini, and N. Wehn, "Exploration and optimization of 3-d integrated dram subsystems," *TCAD*, vol. 32, no. 4, pp. 597–610, 2013.

[9] B. Feero and P. Pande, "Networks-on-chip in a three-dimensional environment: A performance evaluation," *Computers, IEEE Transactions on*, vol. 58, no. 1, pp. 32–45, Jan 2009.

[10] X. Li and O. Hammami, "Nocdex: Network on chip design space exploration through direct execution and options selection through principal component analysis," in *Industrial Embedded Systems, 2006. IES '06. International Symposium on*, Oct 2006, pp. 1–4.

[11] A. B. Kahng, B. Li, L. S. Peh, and K. Samadi, "Orion 2.0: A fast and accurate noc power and area model for early-stage design space exploration," in *Design, Automation Test in Europe Conference Exhibition, 2009. DATE '09.*, April 2009, pp. 423–428.

[12] M. Lis, K. S. Shim, M. H. Cho, P. Ren, O. Khan, S. Devadas *et al.*, "Darsim: a parallel cycle-level noc simulator," in *MoBS 2010-Sixth Annual Workshop on Modeling, Benchmarking and Simulation*, 2010.

[13] J. Joseph and T. Pionteck, "A cycle-accurate network-on-chip simulator with support for abstract task graph modeling," in *System-on-Chip (SoC), 2014 International Symposium on*, Oct 2014, pp. 1–6.

[14] C. H. Chao, K. Y. Jheng, H. Y. Wang, J. C. Wu, and A. Y. Wu, "Traffic- and thermal-aware run-time thermal management scheme for 3d noc systems," in *Networks-on-Chip (NOCS), 2010 Fourth ACM/IEEE International Symposium on*, May 2010, pp. 223–230.

[15] D. Park, S. Eachempati, R. Das, A. Mishra, Y. Xie, N. Vijaykrishnan, and C. Das, "Mira: A multi-layered on-chip interconnect router architecture," in *Computer Architecture, 2008. ISCA '08. 35th Int. Symp. on*, June 2008, pp. 251–261.

[16] W. J. Dally and B. Towles, *Principles and Practices of Interconnection Networks*. Elsevier, Inc., 2004.

[17] C. Bienia, "Benchmarking modern multiprocessors," Ph.D. dissertation, Princeton University, January 2011.

[18] J. Hestness and S. W. Keckler, "Netrace: Dependency-tracking traces for efficient network-on-chip experimentation," The University of Texas at Austin, Department of Computer Science, Tech. Rep. Technical Report TR-10-11, May 2011.

[19] P. K. Sahu and S. Chattopadhyay, "A survey on application mapping strategies for network-on-chip design," *Jour. of Sys. Arc.*, 2013.