Mapping Irregular MPSoC Topologies onto 2D-meshes

José Cano*,[1], José Flich*,[1],
José Duato*,[1], Marcello Coppola†,[2],
Riccardo Locatelli†,[2]

* *Department of Computer Engineering (DISCA), Universitat Politècnica de València, Camí de Vera s/n, 46022, Valencia, Spain*
† *STMicroelectronics, 12 rue Jules Horowith, 38019 Grenoble Cedex, France*

ABSTRACT

In application-specific SoCs the topology is usually irregular. In this paper we present a mapping tool able to map the initial irregular topology of a SoC system into a logical regular structure (2D-mesh) where a Logic-Based Distributed Routing mechanism (LBDRx) can be applied. The goal is to implement the routing paths without using memory resources. The original topology is not changed. Evaluation results show the effectiveness of the mapping tool.

KEYWORDS: Systems-on-Chip; Networks-on-Chip; Mapping

1 Introduction

As technology advances, systems-on-chip (SoC) designs become more complex with the inclusion of many IP components. Usually, tens of elements need to be connected within the same chip, thus requiring an efficient on-chip interconnect [BDM06, FBe10] with no regular shape and with varying switch complexities and link bandwidths. Figure 1 (a) shows an example where IP blocks are connected by using an on-chip network with 28 switches. As can be observed, the network topology is totally irregular and heterogeneous.

Two key pillars of an interconnect are the topology and the routing algorithm. The topology sets the physical connection pattern between end nodes, and as indicated previously, in application-specific SoC systems is usually irregular. The routing algorithm, on the other hand, sets the paths that messages need to take within the network.

In this paper we provide a mapping tool for application-specific SoC systems where the topology is set by the application, thus being totally irregular. The tool is able to map the initial irregular topology into a logical regular structure where the LBDRx approach can be used. LBDRx enables the use of table-less distributed routing on every switch with a constant and reduced logic cost, regardless of system size.

[1] E-mail: jocare@gap.upv.es, {jflich, jduato}@disca.upv.es
[2] E-mail: {marcello.coppola, riccardo.locatelli}@st.com

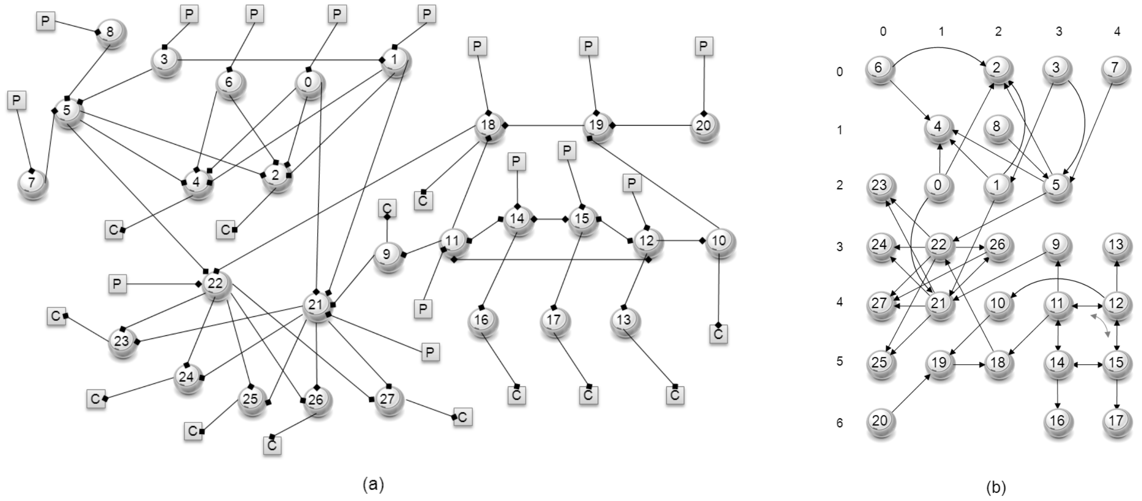

Figure 1: (a) Example of a complex irregular topology for an application-specific SoC system. P means producers and C means consumers. (b) Mapping example for the complex topology.

2 Mapping Tool Description

The mapping tool can be adapted to different versions of LBDRx (for details of LBDRx refer to [CFD+11]), e.g. with 3-hop links, 2-hop links, and with/without non-minimal path support. The tool takes as an input the topology and the type of LBDRx support and outputs several possible solutions, each of them able to be used with the target LBDRx version. Indeed, the mapping tool provides for any possible solution the set of configuration bits together with the mapping coordinates of every switch into a 2D grid. It is worth highlighting that the mapping tool does not physically change the topology, indeed it only logically maps the topology onto a 2D grid. Figure 2 shows the different stages of the mapping tool. In the next subsections we describe the details of each stage.

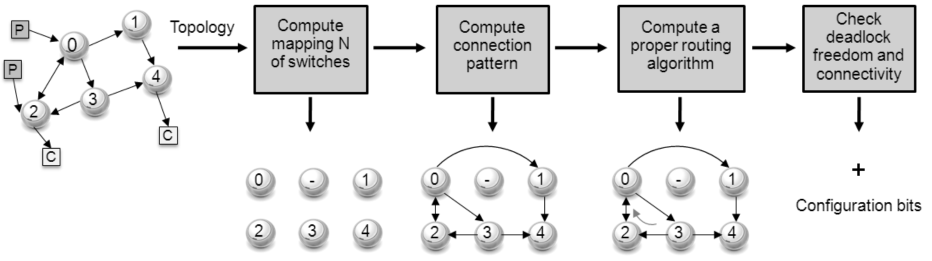
Topology
Compute mapping N of switches
Compute connection pattern
Compute a proper routing algorithm
Check deadlock freedom and connectivity
+
Configuration bits
P
C

Figure 2: Stages in the mapping tool.

2.1 Compute Mapping of Switches

The first stage provides an initial mapping of the switches into a 2D grid. Some basic assumptions considered are: (i) Only switches are considered for the mapping, thus not considering end nodes, (ii) The 2D-mesh diameter will be minimized and made as square as possible and (iii) Every possible mapping of switches onto the 2D mesh is explored and further analyzed in the following stages (most of them will result in mappings not supported by LBDRx). After this stage we cannot deduce which mappings can be supported. For this, we need to compute the connection pattern.

2.2 Compute the Connection Pattern

The connection pattern considers only links connecting switches (switch-to-switch links) and the direction of each link (unidirectional links are considered). Several restrictions in this step are: (i) Any switch has at maximum 20 outgoing ports and 20 incoming ports, possibly having less number of ports, and not necessarily the same number of input and output ports, and (ii) In every switch one possible direction out of 20 can be taken through a single output port. The supported directions are the ones supported by LBDRx (N, S, E, W, NN, SS, EE, WW, NE, NW, SE, SW, NNE, NNW, SSE, SSW, EEN, EES, WWN, WWS). Taking into account the previous restrictions, mappings with link lengths longer than the targeted LBDRx version will become not valid.

2.3 Compute a Proper Routing Algorithm

Once we have obtained a correct mapping we need to check whether the mapped topology contains cycles. In order to avoid them, it will be necessary to apply a routing algorithm. In our case, the routing algorithm used is the Segment-Based Routing (SR) [FMLD07], a technique that divides the network into segments and puts a routing restriction in each segment. A routing restriction is placed between two consecutive links and prevents any message from using both links sequentially. In order to compute the routing restrictions, only 1-hop links in the mapped topology are assumed. This assumption simplifies the LBDRx logic and still allows to reach our objective of obtaining correct mappings. Notice that LBDRx computes the routing bits from the routing restrictions defined by the routing algorithm.

2.4 Check Deadlock-Freedom and Connectivity

The last step of the mapping tool is to verify the mapping is deadlock-free and guarantees the connectivity of the initial topology. The routing algorithm applied in the previous step ensured deadlock-freedom between 1-hop links, thus, now it must be checked along 2-hop and 3-hop links. On the other hand, when applying the routing algorithm, some pair of end nodes may be unconnected. The reason is because LBDRx relies exclusively on minimal paths, those always getting closer to its destination. Therefore, a routing restriction may lead to a path being routed non-minimally. In this situation a deroute is placed (more details in [CFD$^+$11]). At this stage, the tool iterates on all the communicating flows of the application (a flow is defined as the path from a producer to a consumer). For each flow, the tool searches a valid LBDRx path using the connectivity and routing bits set by the mapped topology. If for a flow, there is no connectivity, then the mapping is not valid and we will need to search a new mapping.

On success of a mapping topology, the final output is the mapping of each switch into the 2D grid and the configuration (connectivity and routing) bits. Notice that the mapping tool succeeds if at least one mapping solution is obtained. Also, if no mapping solution exists for a grid size, the mapping tool extends the grid by one row and/or column thus having much larger flexibility. Figure 1 (b) shows a successful deadlock-free mapping for the initial topology depicted in Figure 1 (a), where connectivity between switches is assured.

3 Evaluation

Table 1 shows the results of applying the mapping tool to different sets of topologies with increasing complexity. The tool was run on AMD Opteron (2,8 Ghz dual core, 8Gb RAM) computers. The main purpose was to compute the number of correct mappings generated for every analyzed topology and the time required to complete the procedure. In each case, the table shows the minimum grid size needed to map the topology (4x4, 5x5, ..., 5x7), the number of correct mappings obtained, and the average number of deroutes used (per mapping), if necessary. Note that correct mappings will be those which met the restrictions imposed by the LBDRx version applied in each case. In the last column the computation time to complete the entire process is shown. This time depends mainly on the complexity of each topology, and for these examples ranges from some minutes to several hours.

	Grid size	Correct	Deroutes	Time
Type 1	4x4	>10.000	not needed	3-5 min
Type 2	5X5	>100.000	not needed	10-15 min
Type 3	5X6	>100.000	not needed	30-45 min
Type 4	5X7	>100.000	10-15	1-2 hours
Figure 1 (a)	5X7	578.952	13-15	2 hours

Table 1: Topology mappings

4 Conclusions

We have presented a mapping tool able to obtain different mappings of the same irregular topology onto a 2D grid. The tool is key for the application of the LBDRx mechanism. The provided results demonstrate the applicability of the mapping tool onto a wide set of topologies. In all cases, a valid mapping was achieved. As future work we plan to focus on performance issues. Different mappings will end up in different performance numbers.

Acknowledgment

This work was supported by the Spanish MEC and MICINN, as well as European Comission FEDER funds, under Grant CSD2006-00046. It was also partly supported by the COMCAS project under Grant CA501.

References

[BDM06] Luca Benini and Giovanni De Micheli. *Networks on Chips: Technology and Tools*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2006.

[CFD$^+$11] J. Cano, J. Flich, J. Duato, M. Coppola, and R. Locatelli. Efficient routing implementation in complex systems-on-chip designs. In *Fifth ACM/IEEE International Symposium on Networks-on-Chip (NOCS)*, 2011.

[FBe10] Jose Flich and Davide Bertozzi (editors). *Designing Network On-Chip Architectures in the Nanoscale Era*. CRC Press, Inc., Boca Raton, FL, USA, 2010.

[FMLD07] J. Flich, A. Mejia, P. López, and J. Duato. Region-based routing: an efficient routing mechanism to tackle unreliable hardware in networks on chip. In *First ACM/IEEE International Symposium on Networks-on-Chip (NOCS)*, 2007.