A Requirements-Driven Reconfigurable SoC Communication Infrastructure Design Flow

Alessandro Meroni, Vincenzo Rana, Marco Santambrogio and Donatella Sciuto
Politecnico di Milano - Dipartimento di Elettronica e Informazione
Via Ponzio 34/5 - 20133 Milano, Italy
alessandro.meroni@dresd.org
{rana, santambr, sciuto}@elet.polimi.it

Abstract

Classical approaches to dynamic reconfiguration address computational cores reconfiguration in order to configure the device with the best set of computational elements that are able to accomplish the desired task, thus focusing on a task-driven reconfiguration. This paper introduces a novel approach that exploits performance-driven reconfigurations, in order to adapt the current system configuration (in particular its communication infrastructure) to the required performances, in addition to the required functionalities. This is possible due to the Reconfiguration Oriented Metrics (ROMe) design flow, that leads to realize reconfigurable systems including a reconfiguration controller to manage the reconfiguration itself. This controller has to be able to make coherent decisions (with respect to a well defined set of reconfiguration oriented metrics) about communication infrastructure reconfiguration processes, in order to make it possible to achieve the desired performance (e.g. latency, throughput or power consumption).

1 Introduction

Nowadays, one of the most important aspect of FPGAs is their ability to be reconfigured at run-time [1]; this makes it possible both to dynamically change the communication infrastructure of a system and to deploy on the reconfigurable device the set of cores (implemented at design time in order to achieve a desired functionality) that are the most suitable for the current status of the device. These features increases both the flexibility of the developed system and number of different scenarios in which it can be employed.

The approach proposed in this paper proposes a complete methodology that supports the designer in the development of both the best communication infrastructure [2] for a given system and the reconfiguration controller that is able to dynamically change the communication infrastructure itself. This can be achieved by analyzing, with several different thresholds, a set of reconfiguration oriented metrics (e.g. area usage or power consumption) and by providing the reconfiguration controller with the results of this analysis, in order to perform the reconfiguration process that will lead to obtain the most promising communication infrastructure for the current status of the reconfigurable system.

In particular, the contributions presented in this paper are:

• the definition of the ROMe design flow, that drives the designer to the implementation of a flexible communication infrastructure that represents the best choice for the given system;

• the definition of a methodology that completely decouples the computational cores of a system from the underlying communication infrastructure, allowing to adapt the communication infrastructure to the cores communication and not vice versa.

This paper is organized as follows. Section 2 presents related works, while Section 3 introduces the rationale of this work. Section 4 describes the details of the proposed approach, starting from the scenarios definition, simulation runs and rules identification, and ending with the reconfiguration controller characterization, SystemC [13] simulations and reconfiguration controller implementation. Finally, Section 5 presents the experimental results related to the proposed approach, while Section 6 draws the concluding remarks.

2 Related Works

Recent approaches to the design of dynamically reconfigurable systems are mainly focused on the development of static communication infrastructures, that cannot be deeply
modified at run-time to improve the overall performance of the system. Examples can be found in works such as [3], [4], [5] and [6], that are all based on bus-based communication infrastructures.

In [3], for instance, the authors modeled the communication behavior of each system module as a set of Communication LifeTime Intervals (CLTIs), trying to optimize the overlaps among them, the bus width and the number of bus used, exploiting ILP (Integer Linear Programming). They obtained good results with respect to the communication topology in complex systems, but the whole work has to be done at design time, thus not allowing to completely exploit run-time reconfigurations capabilities. In addition to this, they considered just a multi-bus based communication infrastructure, without taking into account other possibilities.

Other approaches, such as the one proposed in [7], describe the possibility to modify at run-time the routing protocol, for instance on a mesh based network-on-chip. These approaches are mainly focused on the dynamic modification of a small set of parameters of the current communication infrastructure, while they do not consider the possibility to completely change it, for instance by substituting a network-on-chip with a bus-based communication infrastructure when needed.

3 Rationale

The proposed approach arises from the need to satisfy performance requirements in a system characterized by a large number of cores that can be dynamically reconfigured. Since different sets of cores could need different communication infrastructure topologies, the system has to adapt its initial underlying communication infrastructure in order to avoid a continuous performance degradation.

In order to cope with this issue, it is possible to define a set of rules (coming from structural simulations performed on a set of reconfiguration oriented metrics) that will drive the system to the choice of the particular reconfiguration process that has to be performed to adapt the communication infrastructure. These rules are based on fuzzy logic, and involve the definition of different thresholds through which it is possible to find the right communication infrastructure.

Another relevant aspect of the proposed approach consists of both the definition and the realization of a design framework for the automatic generation of flexible and modular communication infrastructures that can be dynamically changed at run-time. This dynamic adaptation of the underlying communication infrastructure can be very useful to solve problems that arise at run-time and that are difficult to identify at compile time, such as a bottle neck due to a heavy communication among two or more cores that have been dynamically placed in distant parts of the reconfigurable device.

4 Reconfiguration Oriented Metrics

This section deals with the description of the ROMe design framework, that consists of three phases, as shown in Figure 1.
collection. The next phase is presented in Section 4.2 and deals with the characterization of the reconfiguration oriented metrics, with the definition of the reconfiguration manager rules and with SystemC simulations. Finally, the third phase is described in Section 4.3 and concerns the reconfiguration controller implementation and deployment of the whole reconfigurable system on the reconfiguration device.

4.1 Simulations and metrics

As previously described, the first phase of the ROMe design flow concerns the definition of a set of different scenarios, simulation runs and data acquisition. Aim of this phase is the definition and the analysis of a set of metrics that will be useful during the reconfiguration controller characterization phase. These metrics have been evaluated by using three different ways: simulations, theoretical formulae and model estimation. With respect to simulations, OMNeT++ [8], a well known network simulator, has been used to generate a set of communication infrastructure based on the concepts of nodes and routers. Nodes have been partitioned in master and slave nodes. A master node has been defined as an IP-Core that can both generate and receive packets, representing a single communication process. On the other hand, a slave node can just receive these packets, sending an ack-packet to the sender to inform it about the correctness of the communication process. Each network has been characterized with coherent parameters and values (e.g. timing, delay, etc.). This leads to obtain a huge number of simulation runs, each one performed on different communication infrastructures, such as network-on-chips [9] (i.e. mesh topology and star topology), buses [10] [11] and point-to-point [12]. These simulation runs have been used to collect data about delivery rate, loss rate, throughput and latency, that will be detailed in the following, of each simulation scenario and each communication infrastructure.

- **Delivery rate** has been defined as the number of completed packets -those arrived at their actual destination- over the processed ones -those that are still circulating in the network- by each ip-core (excepts the router nodes).

- **Loss rate** represents the ratio between the dropped packets and the processed ones, by each router node or bus or central node (for the star topology).

- **Throughput** has been defined as packets per seconds. In particular, a conservative formula has been used: total number of completely routed packets (of each ip-core) over the total simulation time (50000sec per simulation performed in about 2 actual minutes).

- **Latency** has been evaluated using two different formulae, due to the presence or not of fifos in the system: the point-to-point and the bus infrastructures have a constant latency, while the star and the mesh one have been evaluated subtracting start time at which a packet has been sent from the time at which it has been received (for each completed packet).

Finally, theoretical formulae [14] [15] were used to obtain data with respect to the area usage, considering it as the links complexity for each communication infrastructure, while a model estimation has been used for the power consumption estimation [12].

4.2 Rules characterization

The second phase of the ROMe design flow deals with the characterization of the information generated by the previous one. All the obtained values have been stored on a database and have been processed through a C++ application in order both to identify common patterns and to define rules that will be used to implement the reconfiguration controller.

Due to the high flexibility of SystemC [13], this environment has been used to make structural simulations of real communication infrastructures. Once a communication infrastructure has been implemented in SystemC, it is possible to implement algorithms that emulate the rules previously defined, in order validate both their correctness and their goodness. During these simulations, the reconfiguration controller is responsible to detect if the current communication infrastructure has to be modified, adapting it to meet the current status of the system.

Finally, it has to be noticed that reconfiguration oriented rules can be based on particular requirements, that can be defined by the designer; for instance, if the main constraint is the area usage and the throughput, rules will try to modify the communication infrastructure in order both to minimize the area usage and to maximize the throughput, while other parameters are taken into account with a very small weight.

4.3 Reconfiguration controller implementation

The last phase of the ROMe design flow concerns the realization of the reconfiguration controller, implementing all the rules defined and validated during the previous phase. By following this rules, in fact, it is possible to make decision about reconfiguration processes that involve modification of the communication infrastructure. The reconfiguration controller can be used to realize a reconfigurable system that is able to detect and to recognize the current
status of the communication, performing an internal partial reconfiguration each time it is convenient to modify the communication infrastructure in order to improve one of the performance parameters.

5 Experimental Results

Results presented in this section are obtained by applying the ROMe design flow to the development of a complete self-reconfigurable system that is able to dynamically and automatically modify its communication infrastructure, when it is necessary to satisfy the required performance. In particular, we intend to focus on results related to the simulation phase, to better explain how it is possible to use them in order to perform the rules definition phase.

On one hand, as stated in Section 4, considering the area usage as the link complexity (in other words as the number of links required by a particular communication infrastructure topology), the approximated formulae that indicate area usage can be found in Table 1. The point-to-point communication infrastructure has been defined as a fully connected one. For what concerns the link complexity of the bus approach, the terms $k$ represents the complexity of the arbiter, while the term $l$ of the star based network-on-chip represents the complexity of the central node of the star.

### Table 1. Theoretical formulae used

<table>
<thead>
<tr>
<th>CI</th>
<th>Number of elements</th>
<th>Link complexity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Point to point</td>
<td>$n$</td>
<td>$n(n - 1)/2$</td>
</tr>
<tr>
<td>Bus</td>
<td>$n$</td>
<td>$n + k$</td>
</tr>
<tr>
<td>Square mesh</td>
<td>$n^2$</td>
<td>$2(n^2 - n)$</td>
</tr>
<tr>
<td>Star</td>
<td>$n + 1$</td>
<td>$n + l$</td>
</tr>
</tbody>
</table>

On the other hand, the database has been populated with several values coming from the simulation of each kind of communication infrastructure combined with different levels of communication traffic intensity. In order to perform simulations with different levels of packet generation intensity, while keeping acceptable the simulation time (about 400-800 minutes for each simulation), three different levels of traffic (obtained by using a continuous distribution function with a different parameter) have been considered: high, medium and low. An example of considerations that can be retrieved with these data are presented within the two graphs in Figure 2.

As shown in Figure 2, each graph presents, on the X-axis, the total number of the elements of the system, while on the Y-axis it can be found the Delivery Rate (a non-dimensional value due to the ratio between the completed packets and the processed ones). Each line of the graph is related to a different number of slave (and then also of master) elements of the system.

These two graphs represent the delivery rate trend of the bus-based approach and of the square mesh network-on-chip. The first graph of Figure 2 shows that the system behavior is basically independent with respect to the number of slave elements, since, as a matter of facts, all the waveforms follow the same trend. It is also important to notice that with the increasing of the number of elements the delivery rate goes quite rapidly to zero.

On the other hand, a square mesh based system behaves in a quite different way, due to the presence of router nodes that increase with the number of elements of the system, and so the slaves as well; also this approach is asymptotic to zero, even if in a slightly different way.

Information provided by the previously presented simulations can thus be used to generate rules. For instance, it is possible to set a Deliver Rate threshold in order to avoid a degradation of the timing performance over that value. When the threshold is reached, it is possible to modify the communication infrastructure as suggested by the data collected during the rules characterization phase in order to carry back the Delivery Rate under the required threshold. This modification of the communication infrastructure can be related either to a simple changing on the communication infrastructure topology or to a changing of the communication infrastructure kind, if the current communication infrastructure cannot allow to satisfy the required constraints.

6 Concluding remarks

The ROMe design flow defines a complete methodology that is able to drive the designer during the definition of flexible and adaptable communication infrastructures. In addition to this, the ROMe design flow allows to define both a set of reconfiguration oriented metrics and a set of reconfiguration rules that will be used to design the reconfiguration controller. This controller will be able both to recognize a performance degradation and to perform a set of reconfigurations of the communication infrastructure in order to cope with it.

This allows to realize a system in which the reconfigurations flow is driven at run-time by the internal status of the communication infrastructure, and not only by the application flow defined at design time. The proposed approach, in conclusion, makes it possible to design a system in which the underlying communication infrastructure can be dynamically changed or adapted (e.g., modifying the topology of a network-on-chip) at run-time, allowing to obtain a requirements and performances-driven approach to system reconfigurations.
Figure 2. An example of data acquired through the OMNeT++ simulator with a medium traffic level. (A) Delivery rate trend on a bus-based system. (B) Delivery rate trend with a square mesh.

References


