Managing Heterogeneity in Future NoCs

Jose Duato
Parallel Architectures Group
Technical University of Valencia, Spain
email: jduato@disca.upv.es

***Abstract*—Although most research on NoCs has assumed the use of regular topologies like 2D meshes, some current trends in chip architecture, combined with expected technology limitations and usage models, will very likely oblige designers to consider less regular topologies to provide the best cost-performance trade-off. Moreover, the set of nodes interconnected by those NoCs will also be heterogeneous, including computational cores of different sizes and computing power, cache blocks and local stores, accelerators of different kinds, and memory controllers. The memory wall problem will likely be addressed by using 3D integration, which will increase heterogeneity significantly, due to the need for locating the hottest cores in the top layer.**

Therefore, in order to deliver the best cost-performance trade-off while minimizing resource and power consumption and providing the maximum flexibility, heterogeneity needs appropriate hardware support in the NoC. This talk motivates the need for efficiently supporting heterogeneity, and sketches some results along this direction, describing power-efficient routing algorithms that provide support for multiple heterogeneous, possibly overlapping regions (e.g. virtual machines, coherence domains) in the presence of faulty components.

I. Introduction

Most theoretical studies of interconnection networks assume homogeneous systems with regular topologies. Homogeneity simplifies the design of topologies, routing and load balancing strategies. It also allows the use of a single switch design as a building block for implementing larger switch fabrics. As the number of cores per chip increases, buses are becoming a bottleneck, and researchers started to consider switched networks with point-to-point links in order to make communication bandwidth more scalable. In doing so, the initial proposals for networks on chip (NoCs) inherited many properties of the interconnection networks that were proposed in the 80s. The reason for this is that they share a common goal: minimizing packet latency while devoting the minimal amount of resources to the network. In the case for the interconnection networks developed in the 80s, the goal was to implement single-chip routers. In the case for NoCs, a goal is to minimize silicon area requirements.

Therefore, it is not surprising that the initial proposals for NoCs are based on wormhole switching, two-dimensional meshes, and dimension-order routing (DOR). Wormhole switching delivers low latency with very small buffers, and hence, with small silicon area and power consumption. DOR is easy to implement with a finite-state machine, and leads to very compact and fast routers. Finally, 2D meshes not only were considered the optimal topology for wormhole networks in the 80s. They also match the 2D layout of current chips, enabling the lowest communication latency among neighbors and minimizing wiring complexity. Interestingly enough, some theoretical studies in the 80s concluded that the 2D mesh is the optimal topology under the assumption of constant bisection bandwidth but such a constraint never became true in practice. However, it may become true for NoCs, therefore emphasizing the use of 2D topologies.

II. Sources of Heterogeneity

A. Architectural Sources

Current chips have significantly more complex requirements than just minimizing latency and power consumption. First of all, they need to be connected to the external components (e.g. DRAM modules), and most current processors already integrate one or more on-chip memory controllers. Even if the number of memory controllers is increased as the number of cores increases, there will always be a smaller amount of memory controllers than processing cores or cache blocks. As an example, current graphics processing units (GPUs) integrate several hundred cores and less than ten memory controllers. This introduces heterogeneity in the topology of the interconnection network, which must properly interconnect those devices. Moreover, it also introduces asymmetry in the traffic patterns since memory controllers are usually located near the chip edges. Additionally, since memory bandwidth will become a scarce resource, traffic destined to memory controllers will very likely produce congestion within the NoC, no matter how overdimensioned it is.

Another potential source of heterogeneity for the NoC is the fact that the devices attached to it have different functionality. In addition to processing cores, there are also cache blocks, which are either shared by all the cores or by subsets of them. Those cache blocks will likely differ in size and shape from the cores, possibly introducing irregularities in the topology of the NoC. Even if caches and cores have a similar size, they will generate different traffic patterns that may recommend some asymmetry in the bandwidth of the different links, or even the use of separate networks for different purposes. For example, a design may implement a 2D mesh topology for the transmission of cache lines and a binary tree (or even a fat tree) for the ordered transmission of coherence commands. This kind of solutions have already been implemented in off-chip networks (e.g. Sun E10000), and therefore, are not unexpected for NoCs as well.

Although the number of cores per chip is increasing at a steady rate, many applications are still sequential. Therefore,

manufacturers are wondering whether increasing the number of cores per chip for the desktop, laptop and mobile markets is a good idea. In this situation, the best way of using the increasingly higher number of transistors per chip that will become available is by integrating more functionality into the processor chip. The next large component that will be integrated is the GPU, at least for application areas that do not require top graphics performance (e.g. all the desktop and laptop applications except for gaming and engineering design). Both Intel and AMD announced plans for this kind of integration. As a consequence, they need to figure out how to effectively use those GPUs in the server market so as to reuse CPU designs. The solution to this problem has already been provided by Nvidia, and consists of using the GPU as accelerator for numerical applications. High-performance computing (HPC) platforms and datacenters will make good use of those accelerators, not only to drastically increase computing power at a relatively low cost, but also to dramatically enhance the Flops/watt ratio. Again, the implementation of a GPU into the processor chip will introduce significant amount of heterogeneity, due to both the different size of this component and the very different traffic requirements with respect to general-purpose processing cores.

Another way of increasing the Flops/watt ratio is by replacing a small number of complex out-of-order cores with deep pipelines by a large number of simple in-order cores with shallow pipelines. However, manufacturers did not make that move because sequential applications would run much slower, and therefore, the number of sales for multicore chips would have dropped dramatically. A possible approach for increasing the Flops/watt ratio while being able to run sequential applications very fast consists of implementing a small number of complex cores (e.g. one or two) and a large number of simple cores, possibly with the same instruction set architecture (ISA). This is the approach followed by the Cell processor, although in this case the ISA is different for simple and complex cores, thus making it more difficult to generate code for this processor. Again, this strategy constitutes another source of heterogeneity, not only because of the different size of the different kinds of cores, which will force the use of irregular topologies, but also because of the different traffic requirements.

B. Technology Sources

In addition to the above mentioned architectural sources of heterogeneity, manufacturing processes will also force designers to adopt solutions that will end up introducing even more heterogeneity. One of the problems is that, as integration scale increases, the number of manufacturing defects is expected to increase. Therefore, yield will drop dramatically unless future designs incorporate support for fault tolerance. Fortunately, interconnection networks can implement relatively cheap solutions to increase fault tolerance by using the alternative paths provided by the network topology. Such a use, however, introduces asymmetries in the way links can be used, both to avoid deadlocks and also because fault-free regions will have more alternative paths, and therefore, will be less heavily loaded.

Another source of heterogeneity is the expected increase in manufacturing process variability. Up to now, chips are tested to determine the clock frequency at which they can safely run, and therefore, clock frequency is determined by the slowest devices in the chip. This is acceptable because variability is still relatively small. As process variability increases, the former approach becomes less interesting, and researchers are proposing different ways to allow different parts of the chip to run at different speeds. When those techniques are applied to a NoC, they result in links and/or switches that require a variable number of clock cycles to transmit information, depending on where they are located.

It is also predicted that VLSI technology will reach a point in which it will be feasible to integrate more transistors as long as they are not all active at the same time. Future processor chips will implement sophisticated temperature controllers able to dynamically adjust clock frequency for independent clock domains, thus introducing functional heterogeneity even in completely homogeneous systems. But this will affect performance guarantees for different virtual machines (see discussion at the next section). Moreover, pipelined switching techniques like wormhole and virtual cut-through perform quite poorly when some links and/or switches in the path are slower than others, because traffic accumulates at the buffers located before entering the slower regions and may even produce congestion. So, this problem needs to be addressed in order not to waste bandwidth.

The temperature problem will be aggravated with the introduction of 3D stacking technology. Effectively, 3D stacking is considered to be the most promising technology to alleviate the memory bandwidth problem of future multi-core chips. By stacking multiple DRAM chips together with a multi-core chip, it will be possible to drastically reduce the pressure on external memory bandwidth as well as memory access latency. However, those stacked chips will need to dissipate heat, and that heat must go through the already very hot, temperature-throttled multi-core chip. In addition to this, 3D stacking is an important source of heterogeneity. Not only chips in the stack will be different. They will also have very different communication requirements. Moreover, communication with each chip will be significantly faster than communication from chip to chip in the stack, due to the much larger number of wires in the metal layers with respect to the number of vias between chips.

C. Usage Model Sources

Finally, the usage model is also a source of heterogeneity. Current market trends, including outsourcing of IT services, have led to the massive adoption of virtualization as a solution to the problem of running applications from different customers in the same computer while guaranteeing security and resource availability. Moreover, in order to optimize utilization, resources are dynamically assigned to different virtual machines according to customer application require-

ments. As a beneficial side effect, virtualization splits a given computer into smaller chunks, thus reducing the severity of the problem of developing parallel code. Providing efficient virtualization support at the chip level is not trivial because splitting the set of cores into smaller disjoint subsets is not enough. It is also necessary to guarantee that each partition forms a region whose internal traffic does not interfere with traffic from other regions. This implies that regions should be formed by a contiguous set of nodes and that the routing algorithm is properly defined, which is not trivial because certain paths must be forbidden to avoid deadlocks. Moreover, each region should contain cores and caches in order to be as independent as possible from each other region, which again has some implications on the component layout and introduces heterogeneity at a finer granularity. Anyway, shared caches will introduce some interference among regions. Even if all the cache levels within the chip were private, memory controllers must be shared. This also implies that memory controllers must be an integral part of each and every region, thus introducing even more heterogeneity in the definition of regions.

Finally, there are application areas in which the application to run is fixed (e.g. some embedded devices). In those cases, it is very likely that the application generates non-uniform traffic, sometimes even using only a subset of the links in the NoC. In those cases, efficiently supporting heterogeneity can lead to significant savings in silicon area and power consumption. For instance, an application specific design may implement only the links and switches that are really needed. Also, different links may provide different bandwidth, according to the application requirements.

III. Some Proposed Solutions

Although there exists no generic solution addressing all of the problems mentioned in the previous section, several solutions have been proposed to address one or more of those problems. Some of those solutions are sketched below.

A. Efficient Unicast and Multicast Support for CMPs

bLBDR [2], an enhanced version of Logic-Based Distributed Routing (LBDR) [1], is a flexible and efficient unicast and multicast routing method that removes the need for using routing tables (both at end-nodes and switches), and provides support for disjoint and overlapped regions (or domains) in a NoC, thus enabling the concept of virtualization at the NoC level. bLBDR fulfills several of the requirements mentioned in the previous section, including support for virtualization, partitionability, fault tolerance, traffic isolation and broadcast across the entire network as well as constrained to coherency domains or regions. In particular, bLBDR allows the definition of deadlock-free unicast and multicast routing algorithms for irregular topologies and non-rectangular regions that deliver high performance, provide efficient support for cache coherence protocols by implementing collective communication operations, provide very efficient support for partitioning and virtualization by allowing the definition of multiple regions with traffic isolation, and provide support for reconfiguration and fault tolerance. It also provides support for accessing shared components like shared caches and memory controllers by supporting the definition of overlapped regions. All of this is achieved by a small and power efficient routing logic that delivers low latency as well as 4X area savings and 17X power reduction when compared to a routing table in an 8×8 mesh NoC. In fact, this design is almost as compact as the one for switches based on finite state machines (e.g. DOR), while providing almost the same degree of flexibility as the designs based on routing tables.

B. Application Specific Routing Algorithms for NoCs

A general-purpose multi-core processor is very flexible but may not be optimal for embedded systems that execute a single application. One way to specialize a general-purpose multi-core chip built using NoC principles is to provide a mechanism to configure an application-specific deadlock-free routing algorithm in the underlying NoC.

In [3], the authors present a methodology to specialize the routing algorithm in table-based NoC routers. It tries to maximize the communication performance while ensuring deadlock-free routing for an application. The paper demonstrates through analysis that routing algorithms generated by this methodology have higher adaptivity as compared to turn-model based deadlock-free routing algorithms for a mesh topology. Performance evaluation with traffic generated by real applications shows that the routing algorithms generated by the proposed methodology achieve an improvement in delay close to 50% and 30% over deterministic XY-routing algorithm and adaptive Odd-Even routing algorithm, respectively.

References

[1] J. Flich, S. Rodrigo, and J. Duato, “LBDR: Efficient Routing Implementation in NoCs”, in Workshop on Interconnection Network Architectures On-Chip, Multi-Chip, 2008.

[2] S. Rodrigo, J. Flich, J. Duato, and M. Hummel, “Efficient Unicast and Multicast Support for CMPs”, in 41st Annual IEEE/ACM International Symposium on Microarchitecture, 2008.

[3] M. Palesi, R. Holsmark, S. Kumar, V. Catania, “A methodology for design of application specific deadlock-free routing algorithms for NoC systems”, Proceedings of the 4th international Conference on Hardware/Software Codesign and System Synthesis (CODES+ISSS), pp. 142-147, 2006.