Challenges of Virtualization in Many-Core Real-Time Systems

Matthias Becker, Mohammad Ashjaei, Moris Behnam, Thomas Nolte
MRTC / Mälardalen University, Västerås, Sweden
{matthias.becker, mohammad.ashjaei, moris.behnam, thomas.nolte}@mdh.se

ABSTRACT

Embedded real-time virtualization is used for single core and multicore platforms to consolidate multiple systems within a single chip. The number of cores on one processor is steadily increasing, making many-core processors with tens to hundreds of cores available in the near future. This gives rise to a number of new challenges for real-time virtualization on such systems. In this work we identify a number of those challenges. We also describe initial ideas on how to tackle them.

1. INTRODUCTION

Virtualization is a key component in todays industrial systems [4]. Many-core processors with tens to hundreds of cores, connected by a 2D-mesh based Network-on-Chip (NoC) bring new challenges. The large number of simple cores deliver enough resources to allocate whole cores to virtual partitions. Therefore, the new challenge will be how to divide the hardware topology in order to guarantee enough resources, computational and memory bandwidth, for all the guest systems. Hereby a guest system is abstracted by an interface, hence can be seen as a component. Such components describe newly developed parallel applications as well as legacy applications designed for single-core systems.

2. PROBLEM DESCRIPTION

An important challenge is the interface definition needed to abstract the resource requirement of guest systems. Given those interfaces, system integration needs to find a suitable composition of guest systems on the many-core. Runtime mechanisms are needed to police the resource access depending on the parameters defined during system integration.

2.1 Challenges for the guest interface

Each guest system has to operate under certain constraints. For many-cores, knowing the location of the cores relative to each other is important. This information is needed in order to perform schedulability analysis of the NoC communication [2]. On recent many-core processors (e.g. Tilera's Tile64 or Kalrays MPPA-256 processor [6, 3]) different NoCs are provided for core to core communication and memory access. Finding the right parameters and abstractions for the interface is an important challenge. Moreover, we may have dependencies among guests, hence guest communication abstraction is required. We propose to define those constraints by an interface: $G_n = \{\beta, N_x, N_y, \mathcal{L}\}$, where n is the index of the system and β is the required bandwidth to offchip memory. The parameters N_x and N_y specify the size and shape of the guest system (i.e. $N_x \times N_y$ yields the number of cores). The set $\mathcal{L}$ defines the individual communications to other guests. Evaluating the minimal values for the interface parameters is another important challenge.

2.2 Challenges for system integration

Since we consider real-time systems, we target static constellations. At design time, a set of guest systems, $\mathcal{G}$, needs to be mapped to the many-core hardware. Since this step is performed offline, sophisticated mapping techniques can be applied. Challenges for the mapping are multidimensional. A physical mapping to the cores needs to guarantee the required network bandwidth and timeliness of the guest communication. Communication latencies depend on the physical location of the cores.

2.3 Challenges for runtime mechanisms

At runtime, allocation of resources specified by a guest interface needs to be guaranteed. Computational resources are statically assigned, thus they do not need to be monitored. However access to the offchip memory is shared among guests. Therefore, distribution of the memory bandwidth during runtime is an important challenge. Lu et al. proposed the (σ, ρ)-based flow regulation for NoC [5] which itself is based on Network Calculus [1]. In order to regulate the outgoing traffic flows on a NoC link, a packet shaper is used to inject messages based on a given profile rather than as fast as possible. This mechanism is successfully implemented in the MPPA-256 processor [3]. We propose to include a software-based packet shaper in the hypervisor. Virtualization of the NoC for guest-to-guest communication may affect the communication inside a third guest as they use the same network. This raises another challenge which is to investigate a proper mechanism to handle communication on this network.

3. REFERENCES

[1] R. Cruz. A calculus for network delay. I. network elements in isolation. *IEEE Transactions on Information Theory*, 37(1):114–131, 1991.

[2] D. Dasari et. al. Noc contention analysis using a branch-and-prune algorithm. *ACM Trans. Embed. Comput. Syst.*, 13:113:1–113:26, 2014.

[3] B. D. de Dinechin et. al. Time-critical computing on a single-chip massively parallel processor. In *DATE '14*, pages 97:1–97:6, 2014.

[4] G. Heiser. The role of virtualization in embedded systems. In *IIES '08*, pages 11–16, 2008.

[5] Z. Lu et. al. Flow regulation for on-chip communication. In *DATE '09*, pages 578–581, 2009.

[6] Tilera. Tile64 processor. http://www.tilera.com/products/processors, Retrieved Oct. 30, 2014.