FPGA Implementation of a Prototype Hierarchical Control Network for Large-Scale Signal Processing Applications

Jérôme Lemaitre$^1$ and Ed Deprettere$^2$

$^1$ ASTRON, Oude Hoogeveensedijk 4, 7991 PD Dwingeloo, The Netherlands  
$^2$ LIACS, Leiden university, NielsBohrweg 1, 2333 CA Leiden, The Netherlands

Abstract. The performance of a high throughput and large-scale signal processing system must not be compromised by the control and monitoring flow that is inherently part of the system. In particular, the interfacing of data flow and control flow components should be such that control does not obstruct the signal flow that is of higher priority. We assume that the signal processing is modeled as a distributed hierarchy of data flow networks, and that the control and monitoring is modeled as a distributed hierarchy of communicating Finite State Machines. The interfaces between leaf-nodes of the control and monitoring network, and the signal processing nodes in the dataflow networks are specified in such a way that the semantics of both network types are preserved. In this paper, we present the prototyping of a control network and its interfacing with a data flow network in a FPGA-based platform, and we analyze the performance of the interfacing in a case study. The HDL code that is involved in the interfaces is generated in a semi-automated way.

1 Introduction

Large-scale signal processing systems such as phased array radio telescopes \cite{15} typically comprise of a hierarchically distributed data flow network (DFN), a hierarchically distributed control network (CN), and an interfacing between these two networks. Depending on the nature of the astronomical source that is observed, the system must be able to operate in modes that range from spectroscopy, pulsar observation or searches for transients. Moreover, disturbances in the high throughput streaming data paths, which are mainly due to radio frequency interferences and changes in the ionosphere, must be monitored and mitigated \cite{14} by re-configuring the dataflow processing at run-time. Thus, to each operational mode corresponds a different set of high-level dataflow processing parameters (e.g. frequency resolution and integration time) and control parameters (e.g schedule to update the number of beams and blanked channels).

We assume that signal processing tasks such as filtering, FFT, beamforming and correlation in the DFN are modeled as nodes of Kahn Process Networks (KPN \cite{12}). The control data for re-configuring and monitoring the processes in the KPNs and/or the components onto which they are mapped is sent over the
CN that has a tree or lattice-like structure, until they reach CN leaf-nodes which interact with KPN nodes through specific interfaces. As shown schematically in Figure 1, the interaction between these nodes is synchronized by means of periodic pulse trains that are distributed to control nodes in a synchronization network. The periods of the pulse trains are so chosen that a command that is sent over the CN to a dataflow process reaches this process during the period that precedes its execution. The process will then execute this command concurrently and complementarily to the DFN data processing.

This paper focuses on the way the CN is modeled and reports on a prototype FPGA implementation. In [7], the interfaces between CN leaf-nodes and KPN processing nodes have been so modeled that the two networks that are designed separately can work together without compromising their individual semantics. This paper also demonstrates that these interfaces can be implemented in a semi-automated way, taking IP re-use and scalability constraints into account, and avoiding error prone and time consuming handcrafting of FPGA implementations [10].

![Fig. 1. Interface between a control network (CN) and a data flow network (DFN). The root, nodes and leaf-nodes receive periodic pulse trains in a synchronization network.](image)

In the remaining of this section we give our problem statement, solution approach and related work. The rest of the paper is organized as follows. In Section 2 we explain how to map nodes of the CN onto soft-cores and how to interface CN leaf-nodes with KPN processing nodes that wrap hardware IPs. In Section 3 we present a case study for the control of processes in a KPN in the DFN from leaf-nodes in the CN, and discuss the results concerning the Hardware Description Language (HDL) semi-automation, the IP re-use, and the scaling in the design. Finally, conclusions and future work are drawn in Section 4.