Signal Processing: *Image Communication* 28 (2013) 1278–1294

Contents lists available at ScienceDirect

Signal Processing: *Image Communication*

journal homepage: www.elsevier.com/locate/image

Methods to explore design space for MPEG RMC codec specifications

Simone Casale-Brunet [a,*,1], Abdallah Elguindy [a], Endri Bezati [a,1], Richard Thavot [a], Ghislain Roquier [a], Marco Mattavelli [a], Jorn W. Janneck [b,1]

[a] *École Polytechnique Fédérale de Lausanne (EPFL), Switzerland*
[b] *Department of Computer Science, Lund University, Sweden*

ARTICLE INFO

Available online 30 August 2013

Keywords:
Reconfigurable Media Coding
Design space exploration
Parallel computing
Dataflow programming
Parallel Analysis and Optimization

ABSTRACT

The recent MPEG Reconfigurable Media Coding (RMC) standard aims at defining media processing specifications (e.g. video codecs) in a form that abstracts from the implementation platform, but at the same time is an appropriate starting point for implementation on specific targets. To this end, the RMC framework has standardized both an asynchronous dataflow model of computation and an associated specification language. Either are providing the formalism and the theoretical foundation for multimedia specifications. Even though these specifications are abstract and platform-independent the new approach of developing implementations from such initial specifications presents obvious advantages over the approaches based on classical sequential specifications. The advantages appear particularly appealing when targeting the current and emerging homogeneous and heterogeneous manycore or multicore processing platforms. These highly parallel computing machines are gradually replacing single-core processors, particularly when the system design aims at reducing power dissipation or at increasing throughput. However, a straightforward mapping of an abstract dataflow specification onto a concurrent and heterogeneous platform does often not produce an efficient result. Before an abstract specification can be translated into an efficient implementation in software and hardware, the dataflow networks need to be partitioned and then mapped to individual processing elements. Moreover, system performance requirements need to be accounted for in the design optimization process. This paper discusses the state of the art of the combinatorial problems that need to be faced at this design space exploration step. Some recent developments and experimental results for image and video coding applications are illustrated. Both well-known and novel heuristics for problems such as mapping, scheduling and buffer minimization are investigated in the specific context of exploring the design space of dataflow program implementations.

1. Introduction

The idea at the core of MPEG RMC (ISO/IEC 23001-4 and 23002-4) is to standardize video or 3DG algorithms with the minimum level of constraints required to achieve systems interoperability [1]; thereby leaving all other degrees of freedom open to improvements and competitive developments. Such an objective is similar to the one pursued by other MPEG standards for which only the decoding process is normative, whereas the encoding process is not standardized. Fig. 1 depicts both the normative and non-normative components of the RMC framework. In the upper part the libraries and the languages used to specify a standard abstract decoder model are represented. Likewise, in the bottom part the non-normative transformation of the standard specification into specific

* Corresponding author.
E-mail address: simone.casalebrunet@epfl.ch (S. Casale-Brunet).
[1] Part of this work has been supported by the Fonds National Suisse pour la Recherche Scientifique (grant number 200021.129960 and 200021.138214) and by the strategic research area ELLIIT.

0923-5965/$ - see front matter © 2013 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.image.2013.08.012

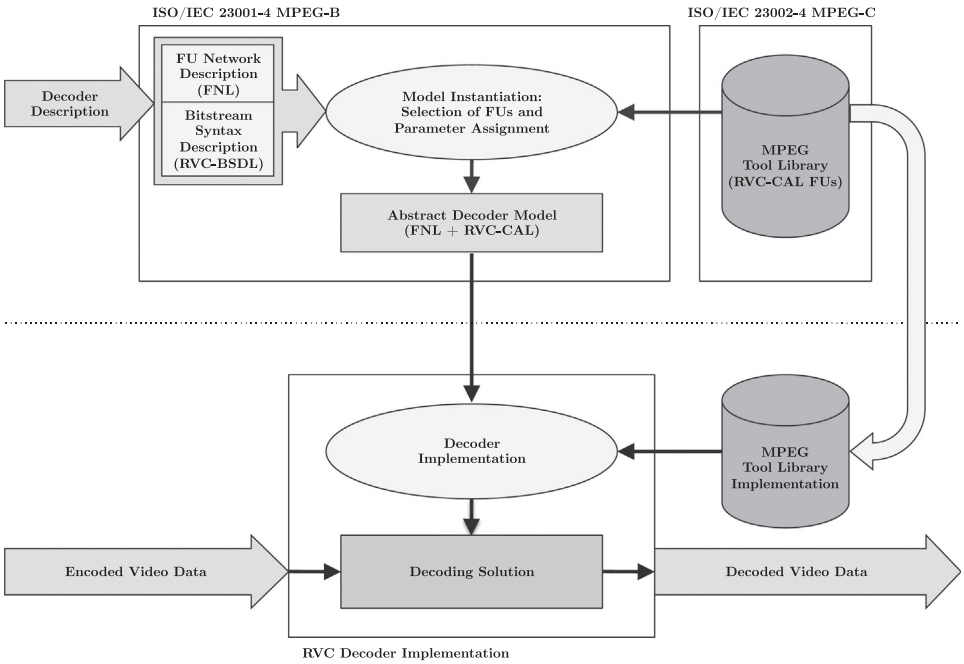

Fig. 1. The normative and non-normative components of the RMC framework.

implementations is represented. In other words, the bottom part just represents any design flow employed to obtain an implementation starting from a RMC specification.

In this perspective, the important element that differentiates RMC from the previous MPEG reference specifications is its underlying computational model. In fact, besides previous reference code where sequential languages such as C and C++ were used, RMC defines its reference code specification as a dataflow program. This description abstracts from the timing characteristics of the algorithm and exposes the data dependencies and parallelism. This approach eliminates much of the incidental *sequentialization* from the abstract specification, thus it opens a vast design space for implementing a RMC specification on a wide range of computational fabrics.

Such an implementation includes: (a) the algorithm partitioning and the mapping of its resulting partitions onto processing elements, (b) the scheduling of the sub-computations that occur on a single processing element, (c) the sizing of communication buffers between parts of the application (i.e. *Functional Units* in MPEG language or *Actors* in the dataflow literature). This design space represents a great amount of flexibility in implementing a coding algorithm, but also it presents a considerable challenge in the form of a very large combinatorial problem that needs to be studied and solved.

With an abstract MPEG RMC as input model, the main steps of the design flow are the following:

- A design exploration stage, which requires the following:
  1. the definition of the partitioning of the model components (i.e. the actors of the dataflow network) onto processing elements;
  2. the scheduling of the operations on each element;
  3. the dimensioning of the buffers between the components.
- A refactoring stage, in which the standard abstract model is transformed with the objective of increasing and/or decreasing the level of explicit parallelism. In other words, changing the structure of the dataflow model of computation according to specific objectives outlined by the exploration stage.
- A synthesis stage, in which the abstract model associated to the parameters defined in the exploration stage is transformed into the executable code for each SW or HW processing element.

Obviously the refactoring and synthesis stages are as well fundamental components of the design flow. Particularly, the synthesis is in practice necessary to provide information to the design space search algorithms and ultimately accurate evaluations of the outlined “optimal” design points. In most of the cases, manual (re-)writing executable code is not only error prone, but also results practically infeasible when complex design needs to be evaluated for several candidate design points. However, refactoring and synthesis are not the focus of this paper and more details can be found in [1–4] and relative references.

This paper provides an overview of the design space exploration in MPEG RMC designs. Both the state of the art and some new contributions on the more used design space exploration techniques are illustrated. Firstly, an overview on the dataflow programming, and more precisely on the RVC-CAL programming language, is discussed in Section 2. Afterwards, the profiling of a dynamic dataflow program is illustrated in Section 3, where the main properties of a dataflow program causation trace are outlined. Then, based on the causation trace, the design space critical path exploration is defined and outlined in Section