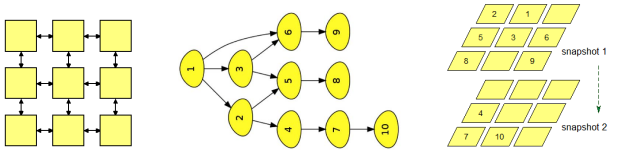
Mapping Tasks to a Dynamically Reconfigurable Coarse-Grained Array

Mansureh S. Moghaddam, Kolin Paul and M. Balakrishnan
Department of Computer Science and Engineering
IIT Delhi, New Delhi, India
{mansureh,kolin,mbala}@cse.iitd.ac.in

Coarse-Grained Reconfigurable Architectures (CGRAs) have become popular in recent times as the increased transistor densities have enabled greater integration of increasingly complex "compute cores". These devices pack massive compute power and can be effectively used to build efficient solutions for applications which have a significant degree of parallelism. In many cases, these CGRAs are also partially reconfigurable. Clearly to make effective use of these highly "parallel compute platforms", a good mapping flow is required to map the parallelism that is present in a target application.

We propose an **optimal** mapping flow which also exploits the partial reconfiguration property of modern CGRAs. Hence this algorithm is not only suitable for applications which can be accommodated in the available silicon but also for larger applications (or a set of applications) which need more area than that provided by the CGRA platform.



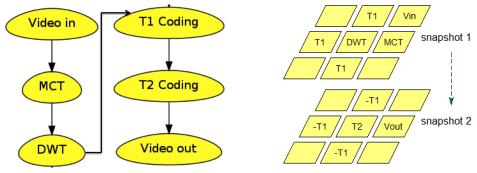
(a) grid of size 3×3 (b) a sample task graph (c) Mapped onto a 3×3 grid

Fig. 1. A sample grid and task graph

In this work partitions of the application are mapped partially and dynamically (during the life time of the application) onto the platform. While in most mapping flows, *scheduling*, *binding* and *place* & *route* steps are carried out in sequence, our proposed mapping flow integrates all three steps in one *mapping* step using an Integer Linear Problem formulation. The objective is to either minimize the total reconfiguration time or minimize the total execution time.

An effective strategy for mapping this parallelism to the many compute elements available in the CGRA fabric is paramount for extracting the required performance from the chosen platform. There are two main input elements in each mapping flow (a) the application and (b) the CGRA platform. In this work we choose a CGRA which is a 2D array of processing elements (PEs) connected with a mesh-like interconnect network; each PE is connected to its four nearest neighbors (4NNs) i.e. top, left, buttom and right. In most CGRAs, the PEs are optimized processors working with a high clock frequency, providing high performance, while consuming very low power. In some CGRAs, local data and instruction memories are embedded along with the PE, e.g. tiles in reMorph [1] is an example in this category. Many CGRAs allow a MIMD style of functioning and hence can exploit task level parallelism in applications.

Embedded and multimedia applications can be described as task graphs where the tasks are statically defined with respect to code size and runtime. The mapping of such tasks to a 2D mesh of dynamically reconfigurable tiles requires "partitioning" of the application task graph in cases where the number of tiles is less than the number of tasks in the graph. This is shown in Figure 1. The algorithm is implemented using TOMLAB/Cplex toolbox and we assess its efficacy on a set of 40 synthetic task graphs. The target applications are the ones for which there is an outer loop of size I, each iteration of this outer loop will go to process one block/frame/chunk of data. There are two possibilities to go through these I iterations; one is going through all the snaphots, and then go through the same process for next frame. The other possibility is to have a loop of size J for each task in each snapshot; each task will process J frames instead of one at each time, and then forward the frame to next task in order. From the experimental results we can conclude that if the boundary tiles in each snapshot are large enough, breaking the outer loop bound to a smaller value and making an inner loop to each tile for each snapshot, the total execution time of the application decreases. For real applications, the algorithm was able to produce a mapping of JPEG2000 application as shown in Figure 2.



(a) Task Graph (b) mapping

Fig. 2. JPEG2000 Application and Mapping to CGRA

References

[1] Kolin Paul, Chinmaya Dash and Mansureh S. Moghaddam, *reMORPH – A Runtime Reconfigurable Architecture*, 15th Euromicro Conference on Digital System Design (DSD2012), September 5-8, 2012, Cesme, Izmir, Turkey, 2012.

978-1-4799-5111-6/14 $31.00 © 2014 IEEE
DOI 10.1109/FCCM.2014.20