**Assignment**

In this assignment the students will work with the reconfigurable architecture developed at Eindhoven University of Technology. This architecture is a CGRA that allows you to construct a processor that matches the structure of the application that you want to run on it very closely. This leads to big performance improvements as well as improvements in energy efficiency.

You will get a reference architecture implementation (constructed with our CGRA) that is quite similar in resources to what you find in a typical RISC processor. You will also receive a naive implementation of the algorithm for this processor. Your job is to modify the architecture (which can be done by modifying a configuration file) and the application source code. By tuning one to match the other, you can tune the design to obtain a optimal balance between performance, power and area.

**Liquid Architectures Lab**

The purpose of this lab is to let students exercise the design-space exploration process of a Very Long Instruction Word (VLIW) processor given a selected set of applications/benchmarks. The student is expected to be familiar with general computer architecture concepts. Instead of following the "cookbook" lab paradigm where the steps of this process are outlined in detail, we decided to make available to the students a parameterized VLIW core and ask them to identify the best parameter values in order to optimize the application execution in terms of certain metrics, e.g., performance, resource utilization. Subsequently, the students are asked to substantiate their design decisions through measured results from simulation and describe their proposal and findings in a written report.

**Heterogeneous computing with GPUs: the lab**

Although heterogeneous platforms integrating GPUs and multi-core CPUs are becoming popular, most applications still use only one type of components in such platforms (the CPUs *or* the GPUs). Users often claim that the implementation effort required to exploit these heterogeneous resources is too large when compared with the potential benefits.

To make educated decisions about (1) the performance that can be achieved, and (2) the true implementation effort for deploying applications on heterogeneous platforms, we prepared hands-on exercises to empirically assess the usability of several programming models for heterogeneous GPU computing. We focus on representative tools from each class, covering static partitioning, dynamic partitioning, domain-specific heterogeneous computing (graph processing), and distributed heterogeneous computing. Our focus is on the usability and suitability of these tools for real applications, and the performance gain potential for each class.

The lab includes three main case-studies: matrix multiplication as a representative scientific kernel,

an image processing pipeline as a representative multi-kernel application, and graph processing as a domain for dedicated, high-level heterogeneous programming models. For each case-study, we provide code templates/skeletons to decrease the time-to-solution. Our aim is to allow the students to compare performance and ease-of-use of the selected models in a hands-on manner.

To complete the lab, the students should provide:

- running heterogeneous versions of all applications using the provided skeletons and programming models.

- a performance comparison between models (for each case-study)

- a reflection on the usability of each of the models.