

Skeletal Programming for Reconfigurable Computing:

Using High Level Structured Parallel Programming to Design Image Applications on FPGAs at System Level

Carlos A. Acosta-León (Institute for Computing Systems Architecture)

Supervisors: Murray Cole & Aris Efthymiou

{c.a.acosta-leon@sms.ed.ac.uk, mic@inf.ed.ac.uk, aefthymi@inf.ed.ac.uk}

Abstract: We propose to use Cole's Algorithmic Skeletons to provide encapsulated parallel programming models as a high level sequential programming interface to reconfigurable computing devices. This method transparently exploits parallelism and reconfigurability on FPGA/CPU systems hiding low level details from end-users. We will apply our skeletons to a design space exploration of image processing applications.

Introduction

Field-Programmable Gate Arrays or FPGAs allow end-users to customize algorithms on hardware to realise specialised tasks with high performance. FPGAs are reusable VLSI templates, made up of a regular computation and interconnection network whose functionality is reprogrammable by software. Additionally, they support embedded complex components such as μP and μC cores, DSPs, high-speed I/O interfaces, RAM memory, etc.

This complexity, along with inherent parallelism, dynamically changing functionality, and coupling the work between FPGA and microprocessor makes a powerful Reconfigurable Computing System [1].

However, this brings a high programming complexity that must be managed by using suitable design techniques at the highest possible abstraction level.

Motivation

- Abstraction techniques which have been successful in parallel software development can be useful for programming FPGAs at high level.
- An approach is based on packaging complex operations as templates, patterns or *algorithmic skeletons* [2].
- Skeletons* encapsulate the necessary control and data flow,
- Software can be written in a way that is independent of particular architectures and hence portable.
- Skeletons* can be oriented to a generic or specific application area (for example, image processing).
- Skeletons-based* Parallel Image Processing Projects such as EASY-PIPE and SKIPPER, both are focused at the software level, with the Belfast Group project on Hardware skeletons, at a low level.

Aims

In contrast, we intend to use *Cole's Algorithmic Skeletons approach* [2] in the FPGA design process to provide:

- Parallel programming models encapsulated as *skeletons*;

- Coordination of concurrent Hw/Sw tasks between FPGA-microprocessor as appropriate;
- A high level API as a sequential interface for end-users;
- A tool for design space exploration on FPGA of image processing applications at system or algorithmic level.

Framework

We plan the design, implementation and evaluation of a small suite of *skeletons*. Figure 1 shows our proposed framework. It has the following characteristics:

- Programmed in SystemC or Handel-C, with an API which allows programmers to exploit the *skeletons* and customise each to the application;
- Internal implementations of *skeletons* which exploit parallelism, reconfiguration and separation of Hw/Sw tasks on CPU-FPGA transparently;
- Using image algorithms as a test application suite, an example is shown in Figure 3;
- Experiments comparing costs between *skeleton* based programs with programmed alternatives with no *skeletons*, measuring performance and resource utilisation.

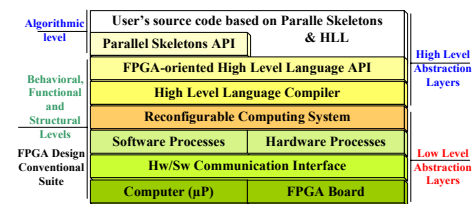


Figure 1. Architecture of the proposed Framework

In Figure 2 we illustrate our generic *skeletal programming methodology*. Here the programmer is responsible for using *skeletons* (1) and providing application specific customisations. The *skeleton library* (2) is the repository of *skeleton* implementations. We use a high level language for FPGA programming. Its compiler is standard, with no understanding of *skeletons*. The FPGA design suite will take

- (3) decisions to select between competing exploitations of parallelism/reconfiguration and about the mapping to FPGA on-chip core or traditional co-processor as generated from the library code.

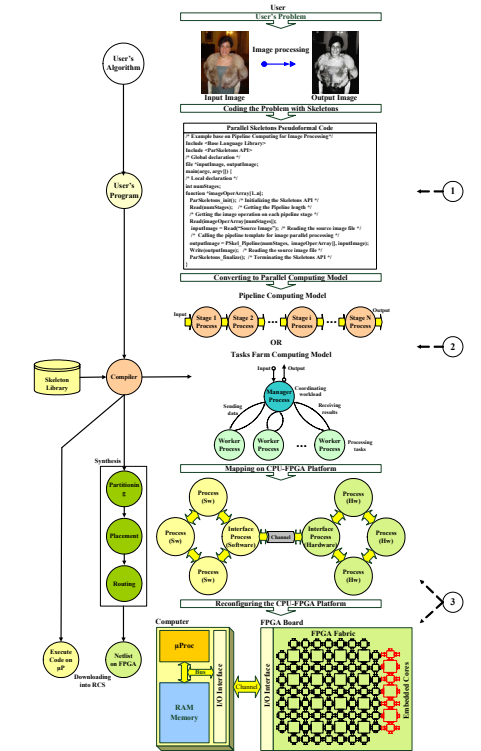


Figure 2. Generic Methodology for the Skeletons Framework

Application example

As our application area we have chosen image processing (double Thresholding edge detection algorithm, as shown in Figure 3) because this kind of application has the following characteristics:

- Compositional development and generalization for testing different image algorithms in each stage,
- Parallelism as Pipeline and Tasks Farm can be used, and
- Different levels of granularity and possibility of exploiting polymorphism.

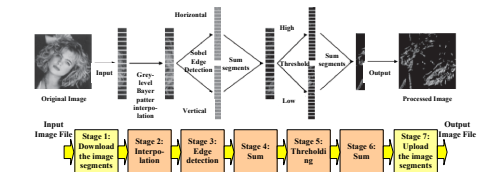


Figure 3. Pipeline Example for Image Processing (Double Thresholding edge detection).

Conclusions and further work

We will provide the FPGA programmer a library with:

- Support to take advantage of reconfiguration, parallelism,
- And support to make decisions about which parts of an overall system to implement in reconfigurable logic and which to implement in traditional software.

So far our current work has focused on a *skeleton* for pipelined processing, expressed in *SystemC RTL* for simulation on computer or synthesis on hardware. For future work we will use a real FPGA design suite, *Celoxica DK4/Handel-C* and *Xilinx ISE 7.1i* on a *Virtex-II Pro* or *RC10* FPGA board, and will integrate *SystemC* and *Handel-C* hardware/software processes.

References

- Maya B. Gokhale & Paul S. Graham, "Reconfigurable Computing: Accelerating Computation with FPGA". Springer 2005.
- Murray Cole, "Bringing Skeletons out of the Closet: A Pragmatic Manifesto for Skeletal Parallel Programming". Parallel Computing, Vol. 30, Issue 3, March 2004, Pages 389-406.

