Title

“AHIR: A competitive pathway from high-level programs to hardware”

Abstract

VLSI systems double in complexity every two years. As a result, it is possible to map increasingly complex applications to hardware. However, the design effort involved in mapping an algorithm to hardware is substantial.

We present a complete flow which starts with an algorithm specified in a high-level programming language (such as C) and ends with an RTL description of a circuit that implements the specified algorithm. We show that this flow is correct-by-construction, and can be applied to a very large (essentially unlimited) class of C programs.

The flow uses an intermediate representation (AHIR) which is an orthogonal factoring of the original specification into control, data and memory aspects. We show that AHIR can be statically analyzed in order to optimize the resulting hardware, and these optimizations are scalable to very large systems.

We demonstrate the flow using a variety of examples ranging from stream ciphers to database and linear algebra applications, and show that the resulting circuits are competitive with purelyprocessor based implementations of the same original specification.

Key Innovations

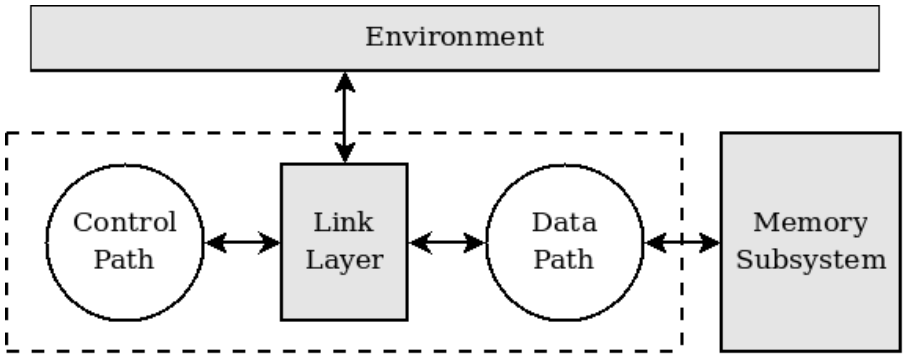
1. An orthogonal representation of the control, data and storage aspects of the input program, that can be routinely translated to a hardware implementation.
2. Support for static analysis and optimization procedures that can scale with the size of the program.
3. A verifiable compilation path from algorithms to digitial hardware, based on the intermediate representation.

Target Applications

1. Ditigal VLSI system design: fast mapping of complex algorithms to hardware.
2. High-performance computing: acceleration of algorithms by mapping them to programmable hardware.
3. Energy efficient embedded system implementations.

Technical Descriptions

AHIR is a compact graph-based hardware representation, that provides a delay-independent specification of the circuit. A specification in AHIR can be directly implemented by a straightforward translation to corresponding hardware components. But the implementor is free to produce other implementations as well, based on additional information available about the target platform. Features such as parallelism and resource sharing may be available in the input language, or introduced by the compiler. In either case, the intermediate representation is rich enough to describe them.



A module in AHIR consists of two closely interacting components: a control-path and a data-path. The data-path in a module is a pool of operators connected by wires. The control-path is a petri-net that specifies the ordering of events in the module. The control-path and data-path interact through a link layer by exchanging request-acknowledge handshakes that encapsulate delays occurring in the implementation. AHIR specifies a set of constraints on the delays involved — an implementation that satisfies these constraints is guaranteed to be correct.

An AHIR specification does not include any details about the memory architecture; only a simple linear address space is assumed. Each data-path can specify an arbitrary number of memory access ports. The only requirement from the memory subsystem is that a request made on an access port must be served eventually.

Publications

1. "AHIR: A Hardware Intermediate Representation for Hardware Generation from High-level Languages" by Sameer D. Sahasrabuddhe, Hakim Raja, Kavi Arya and Madhav P. Desai, presented at the *20th International Conference on VLSI Design, January 2007*.
2. PhD Thesis entitled "A competitive pathway from high-level programs to hardware specifications" submitted for review in October 2008.

Sector in focus

1. Embedded system designers.
2. Digital VLSI hardware designers.
3. High-performance computing.

Contact details

1. Sameer D. Sahasrabuddhe
   Department of Computer Science and Engineering
   sameerds@it.iitb.ac.in

2. Prof. Madhav P. Desai
   Department of Electrical Engineering
   madhav@ee.iitb.ac.in