HercuLeS: Overview and features

HercuLeS (http://www.nkavvadias.com/hercules/) is an extensible high-level synthesis environment for automatically mapping algorithms to hardware. Essentially, HercuLeS translates programs in a typed-assembly language named N-Address Code (NAC) to a collection of Graphviz CDFGs (Control-Data Flow Graphs) which are then synthesized to vendor-independent self-contained RTL VHDL. HercuLeS is also used for push-button synthesis of ANSI C code to VHDL.

Overview

The basic steps in the HercuLeS flow are shown in the figure below. C code is passed to GCC for GIMPLE dump generation, optionally following an external source-level optimizer. Textual GIMPLE is then processed by **gimple2nac**; alternatively the user should directly supply a NAC translation unit or use an owned frontend. Alternative frontends which are currently WIP are being developed for LLVM (C, C++, Objective-C) and domain-specific languages (DSLs).

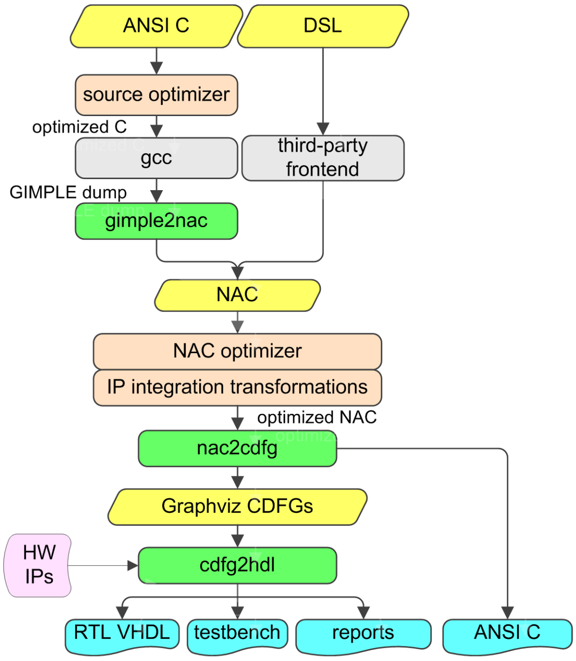

Figure 1: HercuLeS overview.

Various optimizations can be applied at the NAC level; peephole transformations, if-conversion, and function call insertion to enable IP integration. Heuristic basic block partitioning avoids the introduction of excessive critical paths due to operation chaining. The core of HercuLeS comprises of a frontend (**nac2cdfg**) and a graph-based backend (**cdfg2hdl**). **nac2cdfg** is a translator from NAC to flat CDFGs represented in Graphviz. **cdfg2hdl** is the actual synthesis kernel for automatic FSMD hardware from Graphviz CDFGs to VHDL and self-checking testbench generation.

nac2cdfg is used for parsing, analysis and CDFG extraction from NAC programs. **cdfg2hdl** maps CDFGs to extended FSMDs (Finite-State Machines with Datapath). An ANSI C backend allows for rapid algorithm prototyping and NAC verification. VHDL code can be simulated with GHDL and Modelsim and synthesized in Xilinx XST and Vivado Design Suite using automatically generated scripts.

Features

HercuLeS supports a wealth of features. The following graphic provides a visualization of the most important existing or planned features.

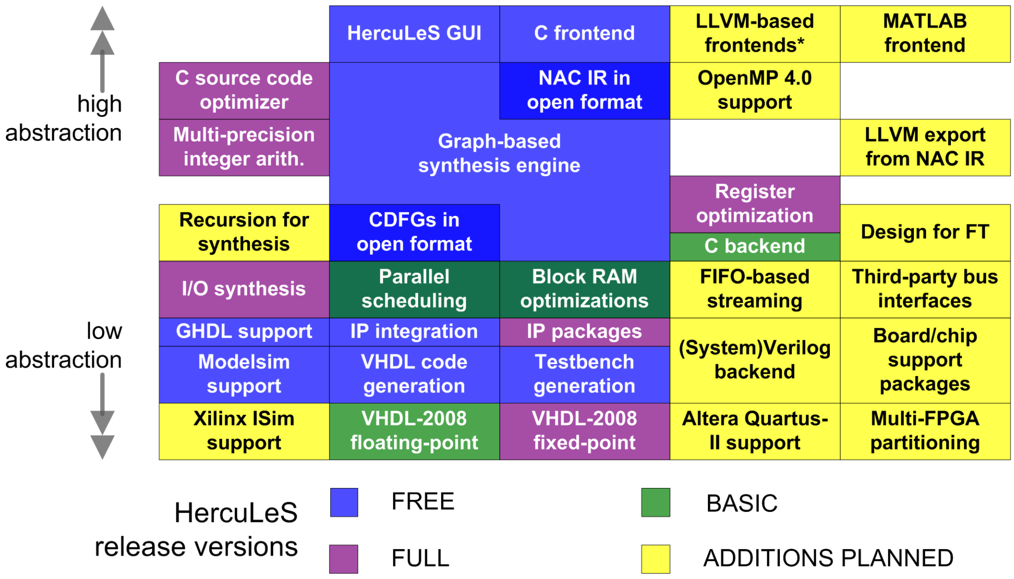

Figure 2: HercuLeS features.

Third-party IP integration

Third-party hardware IP blocks can be seamlessly intergrated to HercuLeS. HercuLeS allows for automatic IP integration given that the user supplies builtin functionalities. The HercuLeS flow user is able to import and use an owned IP by the following process:

1. Implement IP with expected interface and place in proper subdirectory.

2. Add corresponding entry in a textual database.

3. Use TXL transformations for replacing an operator use by a black-box function call via a script.
4. A list of black box functions is generated.
5. HercuLeS automatically creates a hierarchical FSMD with the requested callee(s).

The following graphic illustrates the combined TXL/C approach. The first two steps apply preprocessing for splitting local variable declarations and removing those that are redundant or unused. Then, they are localized and subsequently procedure calls to black-box functions are introduced. These routines are the actual builtin functions. If the corresponding builtins are listed in the IP database, an interface-compatible VHDL implementation to HercuLeS caller FSMDs is assumed. Then, **cdfg2hdl** automatically handles interface generation and component instantiation in the HDL description for the caller FSMD description. In addition, simulation and synthesis scripts already account for the IP HDL files.

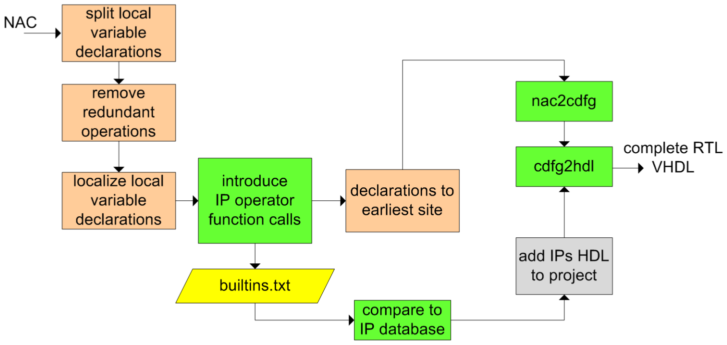
NAC
split local variable declarations
remove redundant operations
localize local variable declarations
introduce IP operator function calls
declarations to earliest site
builtins.txt
compare to IP database
nac2cdfg
cdfg2hdl
complete RTL VHDL
add IPs HDL to project

Figure 3: Automatic IP integration in HercuLeS.

This approach is also valid for floating-point computation, while both pipelined and multi-cycle third-party components are supported.

Summary

HercuLeS delivers a contemporary HLS environment that can be comfortably used for algorithm acceleration by predominantly software-oriented engineers. In this article, we only covered a high-level overview of how HercuLeS work, walked through existing and planned features and briefly discussed automatic hardware IP integration.

Useful literature and links

- Commercial webpage: http://www.ajaxcompilers.com/technology/hercules-high-level-synthesis
- Technical webpage: http://www.nkavvadias.com/hercules/index.html
- Technical presentation: http://www.nkavvadias.com/hercules/hlstool_pres.pdf

- Reference manual: http://www.ajaxcompilers.com/publications/user-manuals/hercules-refman.pdf
- ASAP 2012 paper: http://www.ajaxcompilers.com/publications/kavvadias_asap2012.pdf
- FPL 2013 paper: http://www.nkavvadias.com/publications/hercules-fpl13-demo.pdf
- PCI 2013 paper: http://www.nkavvadias.com/publications/hercules-pci13.pdf