Short guide to the HercuLeS web interface

Date:	2011-12-18
Author:	Nikolaos Kavvadias
Email:	nikolaos.kavvadias@gmail.com
Revision:	0.0.3 (2011-12-18), 0.0.2 (2011-11-04), 0.0.1 (2011-10-29)
Web site:	http://www.nkavvadias.com/hercules/
Copyright:	Nikolaos Kavvadias (C) 2011

Contents

1 Introduction

HercuLeS is a new high-level synthesis tool for targeting imperative languages to synthesizable RTL VHDL with the help of a typed assembly-like intermediate representation named NAC.

This short document stands for a direct how-to guide in order for early users to better utilize the HercuLeS web interface: http://www.nkavvadias.com/cgi-bin/herc.cgi

The main HercuLeS site (http://www.nkavvadias.com/hercules/) already contains a lot of information regarding its internals and typical usage. There is an EE Times article dealing with the same issues with a more user-oriented approach.

2 HercuLeS requirements

HercuLeS can be used by providing either ANSI C or NAC source files.

The basic requirements for HercuLeS is a VHDL simulator (GHDL and Modelsim are supported) and the bash shell. On Windows, the following are suggested:

- GHDL (http://ghdl.free.fr) or the commercial Modelsim simulator (http://www.mentor.com)

- A bash enviroment. For Windows, Cygwin provides a rich POSIX environment and includes bash (http://www.cygwin.com)

In order to run user simulations, it is adviced to obtain the following files:

1. This document, available as HTML (http://www.nkavvadias.com/hercules/hercules-web-guide.html) and PDF (http://www.nkavvadias.com/hercules/hercules-web-guide.pdf).
2. Ready-to-use examples of ANSI C and NAC programs from here: http://www.nkavvadias.com/hercules/small-examples.zip
3. A set of VHDL packages for running VHDL simulations: http://www.nkavvadias.com/hercules/hercules-contrib-vhdl.zip

The latter archive is mandatory for correctly running the simulations.

3 Quick start guide

The user should unzip the hercules-contrib-vhdl.zip archive so that the included VHDL packages reside in */yourpath/contrib/vhdl*.

The design files produced by HercuLeS should then reside in */yourpath/tests/design*.

Here, we assume that the user has accessed the HercuLeS web interface and synthesized a C source file named *design.c*. Then, an archive named design-userfiles.tar.gz containing all generated files, has been sent to the user's mailbox. This archive should be uncompressed to */yourpath/tests/design*.

To execute a VHDL simulation, the user should just type from within the *design* subdirectory:

```
./design.sh
```

That's all that is needed for a basic use of HercuLeS-generated files.

4 Manual changes to the user files

For use with the automatically generated self-checking testbench, a reference data file containing inputs and expected outputs (in hexadecimal notation) is used. This file is named `design_test_data.txt` so that it is visible by the automatically generated testbench (`design_tb.vhd`). The small-examples.zip package provides sample C code that can be compiled to produce the reference data files. If the user wants to adapt this file, then the corresponding data file has to be manually generated.

For this purpose, the user should compile the `design.c` source from the command prompt/terminal:

```
gcc -DTEST -Wall -O2 -o design design.c
```

Then, the `design` executable must be run to produce the test data:

```
./design > design_test_data.txt
```

To execute a VHDL simulation, the user should just type

```
./design.sh
```

This script invokes the simulation of a design with the top-level entity under the name `design` which is kept in file `design.vhd`. The actual makefile for running the simulation is `design.mk` which can be edited to suite the user's needs.

5 Rules and coding style for ANSI C code

HercuLeS expects that the user resorts to a synthesis-friendly coding style with the following basic rules:

- The `main()` function should not be included.
- Only single-dimensional fixed-size arrays are allowed.
- Output arguments of a function are declared as pointers.
- Non-root functions can have arrays as arguments but the root (top-level) procedure can't.
- Due to limitations with GIMPLE dumps, global arrays should be declared as static within the root procedure.
- Non-root procedures should not access global arrays.
- All functions return `void`.
- `goto` is not supported.
- Recursion is not supported.
- Structs, unions and all forms of compound data types (except single- dimensional arrays) are not supported.

6 Limitations of this version

This version comes with certain intentional limitations. Here is a quick list:

- The number of NAC code lines are limited to 25. This may not be easily visible when passing an ANSI C source file to HercuLeS. The examples pack (small-examples.zip) provides sample sources that respect this limitation.
- The ANSI C backend is not accessible.
- Fixed-point arithmetic is not accessible.
- Use of a very slow (combinational) divider.
- No third-party/user IP integration.
- Synthesis script is not generated.
- The RTL VHDL code is generated according to the `IEEE` standard packages. The `synopsys` de facto packages (`ieee.std_logic_arith` instead of `ieee.numeric_std`) are not used.
- No streaming outputs.
- No constant multiplication/division optimizations.
- Also, a lot of other optimizations are kept unused.

7 Known issues

- The current version of `IEEE` standard operators for `xdivmod`, `xdiv` and `xmod` is not synthesizable. This issue will be resolved in the near future.

8 Final notes

This document is a work-in-progress.