libhdlfltp – FPGA coprocessor math library

2008

Authored by: Alan Coppola

libhdlfltp – FPGA coprocessor math library

Open Source License: GPLv3, LGPLv3

Overview

The libhdlfltp library is a coherent collection of floating point VHDL operators, constructed for synthesis to FPGA devices, as part of a coprocessor acceleration system, or as part of an embedded acceleration system. The main characteristics of the library are that it is fully pipelined and parameterized.

The history of this library is that it is a sanctioned fork of the FPLibrary project, located at

http://www.ens-lyon.fr/LIP/Arenaire/Ware/FPLibrary/

To quote the FPLibrary website:

> ***A VHDL Library of Parametrisable Floating-Point and LNS Operators for FPGAs***
>
> ***This page presents FPLibrary, a VHDL library of hardware operators for the floating-point (FP) and the logarithm (LNS) number systems, developed in the Arénaire project, at ENS Lyon.***
>
> ***By Jérémie Detrey and Florent de Dinechin***

Most of the documentation and information contained in the many good papers written about FPLibrary operators is still applicable to libhdlfltp. Please read them when in doubt about the behavior of the components.

Purpose

The libhdlfltp library was created to start supporting math.h like functionality for FPGA coprocessor and embedded systems design, in a vendor independent, open source, and easy to use way. With more usage, different Platform Support Packages(PSP) will be supported.

License

This package is licensed under the GPLv3 license, for any code derived from the FPLibrary package,as per the licensing restrictions on FPLibrary. Any code in this package, not derived from the FPLibrary package, is licensed under the LGPL, v3 license.

Differences

The differences between FPLibrary and libhdlfltp are the following:

1. **Simplification:** All references to Logarithmic Normal System(LNS) components were removed.
2. **Composition:** All operators were equipped with a Slack Size(SlkSz) parameter, which increased the latency of the operator by SlkSz number of clock cycles. This simply adds FIFO buffers to the output ports of the operators. This enables composition of operators to be described easily, and allows latency coherency to be expressed simply.
3. **Refactoring:** All of the operators, including exp, log and trig operators were given coherently described wrappers and were packaged in library directories and files, with a simple naming scheme.
4. **Validation and Verification and Synthesis:** Scripts, samples, and test benches are supplied for the simulation and synthesis of the components. A VHDL testbench is supplied which reads in streams of real number inputs and expected outputs, in standard decimal form, from a file, converts the numbers to binary form, feeds the Device Under Test, and validates the library computations, adjusting for the component latency.

Quick Start

The purpose of this section is to get a design using libhdlfltp, both simulating and synthesizing with a minimum of effort. The sample application is, Norm.vhd, which solves the following problem:

Given an input stream of vectors of pairs of real numbers, compute a stream of real numbers, one for each input pair, which is the distance from the origin. The function computed is (Norm(a,b)= Sqrt(a^2 + b^2))of that stream, with one computation per clock cycle.

Install

1. Download the latest version of libhdlfltp.zip from http://sourceforge.net/projects/libhdlfltp

2. Unzip libhdlfltp.zip or libhdlfltp.tar.gz in a directory of your choice.(See Directory Structure section below for the structure of what is unpacked.) Call this directory $HDLRT.
3. Pick the Platform Support Package (PSP) for the implementation desired. [NOTE: For the current release, there is only one main PSP, and that is:
    a. Operating System and Environment: Windows, with MSYS and Mingw installed.
    b. Synthesis: Xilinx Webpack 9.2i.
    c. Simulation
        i. ModelSim XE Starter Edition, coupled with Webpack 9.2i release.
        ii. [Preparation for Linux ghdl simulation, using auto-generated Makefile.oper and build script build_fplib.sh. (Thanks to Phil Tomson)]
    d. Board Validation: None, but work proceeds on an Opal Kelly board.

Simulation

1. cd $HDLRT
2. cd tb/hdlmdl
3. ./run_verif.sh libhdlfltp
4. Look at Input File ../data/ut_fpnorm_001_in_strm.txt and
5. Look at Output File ../data/ut_fpnorm_001_out_strm.txt. The Err1 column in the output file should have all errors ~<10^{-5} after the latency has passed. Also look at libhdlfltp_simpart.log file for the Modelsim log file of the run.
6. Another way to run this example is to startup the Modelsim GUI, and in the cmdline window of the GUI issue the commands:
    a. cd $HDLRT/tb/hdlmdl
    b. do {make_sim_libhdlfltp_modelsim.do}

Synthesis

1. cd $HDLRT
2. cd tb/implmdl
3. ./run_synth.sh norm

4. Check the answer files in the directory norm_syn(norm_xst.log), and make sure the file norm.bit was created, as that is the bitstream file to be loaded into the device.

Definitions

1. **Real Number Format**

   R = $(-1)^{sgn}$ *(Exp)*(1.Frac) = floating point number representation
   Sgn = sign of R,
   Exp= $(2)^{TrueExp}$ where TrueExp is an integer.
   Frac = number in [0,1)
   Note that R is normalized, in that the fractional part is in the interval [1,2).

2. **Libhdlfltp Number Representation Format**

   Size = We + Wf + 3 bits

   ValExp = TrueExp-Bias hss wE bits allocated.

   Bias = $2^{(We-1)}$ -1, allows for negative powers of the base,
   represented for [0, …, Bias-1],
   and positive powers of the base represented for [Bias,…, 2^{wE}]

Exn(2-bits)	Sign(1-bit)	Exponent(wE-bits)	Fraction (wF-bits)

3. **Representation Exceptions(Exn)**: IEEE754 –like, except for no denormalized numbers.

Exn	Meaning
00	R=$(-1)^{sgn}$ *0 = ieee754 Signed Zero
01	R=normalized number (Normal case)

10	R= =$(-1)^{sgn}$ * infinity
11	R=Not a Number (NaN), as per ieee754

Directory Structure

```
libhdlfltp
+---doc
|       libhdlfltp.docx
|       libhdlfltp.pdf
|
+---tb
|   +---data
|   |       bsmcmm__mctrls100matm_ostrm.txt
|   |       bs_instrm.txt
|   |       test_bs_002_in_strm.txt
|   |       test_bs_002_out_strm.txt
|   |       test_bs_003_in_strm.txt
|   |       test_bs_003_out_strm.txt
|   |       test_bs_in_strm.txt
|   |       test_norm_in_strm.txt
|   |       test_norm_real_in_strm.txt
|   |       ut_fpmult_000_in_strm.txt
|   |       ut_fpmult_000_out_strm.txt
|   |       ut_fpmult_001_in_strm.txt
|   |       ut_fpmult_001_out_strm.txt
|   |       ut_fpnorm_000_in_strm.txt
|   |       ut_fpnorm_000_out_strm.txt
|   |       ut_fpnorm_001_in_strm.txt
|   |       ut_fpnorm_001_in_strm.txt~
|   |       ut_fpnorm_001_out_strm.txt
|   |       ut_fptrig_001_in_strm.txt
|   |
|   +---hdlmdl
|   |   |   build_fplib.sh
|   |   |   Makefile.oper
|   |   |   bc_comp_order.prj
|   |   |   fp_strm_norm_avg_tb.vhd
|   |   |   lhdl_oper_utb.vhd
|   |   |   libhdlfltp_runsim.bat
|   |   |   libhdlfltp_simpart.log
|   |   |   libhdlfltp_vltb.log
|   |   |   make_sim_libhdlfltp_modelsim.do
|   |   |   modelsim.ini
|   |   |   ongn_fpaccum.vhd
|   |   |   ongn_ut_fptrig_tb.vhd
|   |   |   run_verif.sh
|   |   |
|   |   +---pkg_ut
|   |   |       norm.vhd
|   |   |       pkg_ut.vhd
|   |   |
|   |   \---tbutils
|   |           pkg_tbutils.vhd
|   |
|   +---implmdl
|   |       bc_comp_order.prj
|   |       norm_xst.scr
|   |       run_synth.sh
|   |
|   \---metamdl
\---vhdl
    +---pkg_add
    |       fpadd_lzc.vhd
    |       fpadd_normround.vhd
    |       fpadd_shift.vhd
    |       fpadd_shift_clk.vhd
    |       fpadd_swap.vhd
    |       pkg_fpadd.vhd
    |
    +---pkg_div
    |       fpdiv_division.vhd
    |       fpdiv_division_clk.vhd
    |       fpdiv_srt4_step.vhd
    |       pkg_fpdiv.vhd
    |
    +---pkg_exp
    |       fp_exp_exp_y1.vhd
    |       fp_exp_exp_y2.vhd
    |       pkg_exp.vhd
    |       pkg_fp_exp.vhd
    |
    +---pkg_fplib_std
    |       fplib_std.vhd
    |       pkg_fplib_std.vhd
    |
    +---pkg_fp_misc
    |       fp_format.vhd
    |       fp_round.vhd
    |       pkg_fp_misc.vhd
    |
    +---pkg_libhdlm
    |       fpadd.vhd
    |       fpadd_clk.vhd
    |       fpdiv.vhd
    |       fpdiv_clk.vhd
    |       fpmul.vhd
    |       fpmul_clk.vhd
    |       fpsqrt.vhd
    |       fpsqrt_clk.vhd
    |       pkg_fplib_fp.vhd
    |
    +---pkg_log
    |       pkg_fp_log.vhd
    |       pkg_fp_log_log.vhd
    |       pkg_log.vhd
    |
    +---pkg_misc
    |       delay.vhd
    |       mux_n_to_1.vhd
    |       pkg_misc.vhd
    |
+---pkg_mul
|       fpmul_addtree_clk.vhd
|       fpmul_product.vhd
|       fpmul_product_clk.vhd
|       pkg_fpmul.vhd
|
+---pkg_sqrt
|       fpsqrt_sqrt.vhd
|       fpsqrt_sqrt_clk.vhd
|       fpsqrt_srt2_step.vhd
|       pkg_fpsqrt.vhd
|
\---pkg_trig
    +---common
    |       left_shift.vhd
    |       lzc_norm.vhd
    |       mux.vhd
    |       pkg_common.vhd
    |       right_shift.vhd
    |
    +---fp_common
    |       format.vhd
    |       pkg_fp_common.vhd
    |       round.vhd
    |
    \---fp_sin_cos
        |   pkg_fp_sin_cos.vhd
        |   pkg_trig.vhd
        |   sin.vhd
        |   sin_cos.vhd
        |   sin_cos_2pi.vhd
        |   sin_cos_fxp.vhd
        |
        +---cos
        |       cos.vhd
        |       cos_10.vhd
        |       cos_13.vhd
        |       cos_16.vhd
        |       cos_20.vhd
        |       cos_23.vhd
        |       pkg_fp_sin_cos_cos.vhd
        |
        +---fcos
        |       fcos.vhd
        |       fcos_10.vhd
        |       fcos_12.vhd
        |       fcos_13.vhd
        |       fcos_15.vhd
        |       fcos_16.vhd
        |       fcos_19.vhd
        |       fcos_20.vhd
        |       fcos_22.vhd
        |       fcos_23.vhd
        |       fcos_9.vhd
        |       pkg_fp_sin_cos_fcos.vhd
        |
        +---fsin
        |       fsin.vhd
        |       fsin_10.vhd
        |       fsin_12.vhd
        |       fsin_13.vhd
        |       fsin_15.vhd
|       fsin_16.vhd
|       fsin_19.vhd
|       fsin_20.vhd
|       fsin_22.vhd
|       fsin_23.vhd
|       fsin_9.vhd
|       pkg_fp_sin_cos_fsin.vhd
|
+---fsin2
|       fsin2.vhd
|       fsin2_10.vhd
|       fsin2_13.vhd
|       fsin2_16.vhd
|       fsin2_20.vhd
|       fsin2_23.vhd
|       pkg_fp_sin_cos_fsin2.vhd
|
\---sin
        pkg_fp_sin_cos_sin.vhd
        sin.vhd
        sin_10.vhd
        sin_13.vhd
        sin_16.vhd
        sin_20.vhd
         sin_23.vhd
```

Components

The following operators are supported:

Arithmetic	Add, Mul, Div, Sqrt	Pipelined, Comb	
Transcendental	Exp, Log	Pipelined, Comb	
Trigonometric	Sin,Cos	Comb	