biquad_filter

A biquad filter design made in VHDL and synthetized using Synopsis

Note: This is not the report. For the location of the report, please see the file structure section below.

Authors

- Frédéric St-Pierre
- Vahid Khojasteh Lazarjan

Dependencies

- Xilinx ISE
- Jupyter Notebook (optional, to open the jupyter notebook file for presentation)
- Python (3.6 or above)

File Structure

Some files are omitted since they are only kept for informational purposes and are not used in the project anymore.

- final.pdf : A report for the project
- biquad_filter_clk.ctstsh : The clock specification parameters for Encounter
- biquad_filter.conf : The Encounter project configuration file
- biquad_filter_gate.v : The generated gate-level logic description from Design Vision, Verilog version
- biquad_filter_gate.vhd : The generated gate-level logic description from Design Vision, VHDL version
- biquadfilter.gds : The generated layout file from Encounter
- biquad_filter_placement.io : The pin placement description for Encounter
- biquad_filtr.script : The script used to convert the initial vhdl representation of the project into a gate-level representation with Design Vision
- figures.pdf : A collection of figures used for the report
- others :
  - wallace_tree_generator.py : The Python code used to generate the wallace tree vhdl code
  - wallace_tree_generator.ipynb : The Python code used to generate the wallace tree vhdl code, version using jupyter notebooks for demonstration

- vhdl_project :
  - biquad_filter.xise : The project file for Xilinx ISE
  - biquad_filter.vhd : The top-level vhdl file for the biquad filter project
  - clock_divider : The clock division module folder
  - FA : A Full-Adder module for the Wallace Tree multiplier
  - HA : A Half-Adder module for the Wallace Tree Multiplier
  - lt_comparator : A less-than signed number comparator, used by the unsigned divider module
  - nbitregister : A general-purpose n-bits register
  - shift_register : A shift register with parallel load function used by the unsigned divider module
  - signed_adder : A combinational signed adder, implementing two different architectures
  - signed_contracter : A signed number contraction module with overflowing representation detection
  - signed_divider : A wrapper for the pipelined unsigned divider
  - signed_expander : A signed number width expansion module
  - signed_inverter : A signed number inversion module
  - signed_multiplier : A wrapper for the combinational unsigned multiplier
  - unsigned_divider : A sequential divider, using shift registers to implement classic shift-and-substract division
  - unsigned_multiplier : A Wallace Tree implementation of an unsigned multiplier