ET4 171 Processor Design Project

I. Assignment

Improve the LEON3 core performance by focusing on computer architecture and computer arithmetic relevant aspects. To evaluate your new core you are given a set of benchmarks from the Dhrystone, Stanford, Whetstone, GMPbench, and MiBench suites, and a VIRTEX4 ML410 FPGA board. As a start point for this project you have to use the grlib-gpl-1.1.0-b4104.zip archive available on the ET4 171 Processor Design Project Blackboard page in the Course Documents folder.

II. Schedule

The project kickoff meeting will take place on April 24th from 8:45 to 10:30 in EWI-Lecture hall @ (D). At the meeting the project teams will be formed and I'll present the project assignment in more details. Given that the project has to be completed in the 4th quarter of the academic year you have to keep the pace with the following schedule:

April 24th	Kickoff Meeting
May 14th	Milestone Group Meetings (tentative)
June 10th	Report Submission
June 12th	Project Presentation

The milestone meetings are per group and they are meant to be a midterm check. **They are NOT optional!** For the milestone you have to present a short progress report that describes your approach and the project status (up to date achievements). I'll make myself available the entire day for the meetings.

Apart of the meetings you can always email your questions to Mihai Lefter or myself. You can also make use of our "open door" policy and/or make an appointment.

III. (Recommended) Approach

Before you start the core redesign you should exercise the LEON3 core first in order to get familiar with the toolset and with the VHDL core description as well as to evaluate the performance of the base line design. To assist you in that the leon3-xilinx-virtex4-ml410.rar archive includes:

1. A prototype design of LEON3 processor specified for Virtex4 ML410 FPGA board.
2. A document that assists you in the compilation and synthesis of the processor core and on the way this can be programmed and exercised on the VIRTEX4 ML410 FPGA board.

After you are sure that you master those aspects (at least the first point is mandatory) you may proceed with the core optimization. Generally speaking there are several avenues you may follow in order to improve the performance of the LEON3 Core. In this project however you should try to focus on computer arithmetic relevant issues as well as on (micro) architectural aspects. In view of that you can concentrate on the following aspects:

1. **Improve the arithmetic parts of ALU.** In the version of LEON3 given to you, there is a 32bit multiplier (*$GRLIB\lib\gaisler\arith\mul32.vhd*) and a radix-2 non-restoring iterative divider (*$GRLIB\lib\gaisler\arith\div32.vhd*). For a multiplication, it takes 5 cycles (with multiple configuration, but configured at 5-cycle in prototype design) to get the result, and for a division, it needs 36 cycles. You can certainly improve the performance of both of them significantly. You're highly recommended to read the IP manual of these two cores (grip.pdf) before you start your design.

2. **Decrease the time delay of critical path.** You can find the timing information of LEON3 core from the "Synthesis Report" in ISE. Maybe you can improve the performance in the critical path, which will increase the frequency significantly. Bear in mind that between the benchmark score and the maximum clock frequency of LEON3 core there is a linear dependence. Thus if an optimization can make the frequency to be improved 1.2 times, a 1.2 times benchmark score can be achieved.

3. **Architectural improvements.** You may also operate changes on the LEON3 core architectural features that may improve the performance but do not require compiler changes.

In general, you may follow any other path as long as it results in performance improvement, you do computer architecture & computer arithmetic relevant work, and **do not change the instruction set architecture**.

When you make any change into the LEON3 core, a simulation is required before you synthesize this project to make sure your design runs correctly. Follow the instructions in the project manual to do the simulations. To save simulation time, in this simulation project, the Dhrystone runs for one loop, instead of 0x00100000 loops as it is the case when running in the FPGA.

If your design is bug free you may follow the implementation procedure described in the project document to evaluate your design on FPGA. To be more effective you may skip the final step in early stages of the project when the ISE synthesis report provides you enough information to further change your design. However, when you are converging to the final solution you need to upload the design in the FPGA in order to get performance evaluation scores.

IV. Project Submission

In your final submission you should include a short report as well as all the source ISE project files and your modified source code of multiplier and divider, in the way you downloaded them from Blackboard. **DO NOT** just compress the entire project directory, because the file size will be too big!

The report should include the following:

1. The general idea behind you overall optimization proposal. Related to the parts that you decided to change or to append you have to answer (at least) these questions: What, how, and why?

2. The design of the improved and/or appended part(s). There is no need to go at the gate level! The basic algorithm you decided to implement and the RTL designs are in general enough. In case some parts are relevant you may go to full adder/gate level if this is essential for your proposal.

3. Performance results for the baseline design (the original LEON3 core) and your improved core. Those should include the following: Detailed synthesis results including timing information, critical path information, and resource utilization information from synthesis report of ISE. Power consumption figures. Screenshot of the GRMonRCP terminal, which shows the result of Dhrystone benchmark and Stanford benchmark.

4. A comparison between your design and the baseline design which includes the basic metrics, i.e., area (A), critical path delay (deduced from clock frequency) (D), benchmarks scores (BSs), power consumption (P), as well as compound metrics, i.e., A*D, A*BS, P*D, and P*BS products, etc. Comment on the obtained results and try to identify which improvement is contributing to which figure of merit.

5. Conclusions.

The report may also include, but it is not mandatory, suggestion for this project! Your feedback is very much appreciated.

V. Final Presentation

Each group has to give a 20 minutes presentation of their project in the context of a symposium to be held on June 12th 2013. Details on the scheduling will be available in due time. In the presentation you have to highlight the main ideas of your proposal and present the results. After the presentation 5 minutes are reserved for questions from the auditorium.

VI. Evaluation Procedure

The projects functionality will be verified and checked for (between groups) plagiarism.

If the project is not functional you do not pass the project. Plagiarism can also make you fail.

Your final score for the project is determined based on the following criteria:

- The performance of your design (DP). The benchmark scores you report are important. The higher the benchmark score you get the better but this is not the only relevant aspect. In the performance evaluation we also take into consideration the other metrics with more emphasis on the compound ones.
- The technical merit of your approach (TM). Aspects as innovation level and implementation quality are considered.
- The report (R). Report organization, content, and language are important aspects at this point.
- The presentation (P). Here we also look at the capability to ask questions and to answer questions from the auditorium.

The ET4 171 final grade is computed as:

$$Grade = 0.35 * DP + 0.35 * TM + 0.20 * R + 0.10 * P - C.$$

C can assume values between 0 and 1.5 and reflects:

- The lack of collaboration in the group;
- The amount of consultancy you asked me for during the project completion.

If the group is functioning like a team and you do not ask too much help C is 0, otherwise it can get values up to 1.5.