|  |
| --- |
| **Introduction** |

The HW/SW Co-Design Lab is intended to give a first practical access to this topic and experiencing the potential of the conjoint design of hardware and software. For this purpose, the reconfigurable processor flow from the company **Tensilica**is used (shown right). The tool suite provides convenient means for analyzing and optimizing the performance of the software. Especially the possibility to easily extend the processor’s hardware architecture by customer specific instructions using the **Tensilica Instruction Extension (TIE)**language is well suited to demonstrate the interaction between hardware and software design.

Within the scope of this lab the following 2 exercises should be done:

1. Implementation of a FIR filter design including performance analysis; HW/SW optimization: e.g. by Fusion, SIMD extension, etc.; also consider different implementation alternatives for the FIR
2. Improve the performance of a given FFT/IFFT algorithm by using basic instruction extension concepts (Fusion/SIMD/FLIX/etc.). Also consider different implementations such as the Decimation in Time (DIT) and Decimation in Frequency (DIF) algorithm.

*Tensilica®, Instruction Extension (TIE) Language, User’s Guide, 02/2014, p.4*

**The written report should only include the results of the second task and need not exceed 3 pages (source code excerpts not included)!**

The detailed project description can be found at the lab webpage:

<https://bildungsportal.sachsen.de/opal/auth/RepositoryEntry/21521498113>

|  |
| --- |
| **Author(s)** |

|  |  |
| --- | --- |
| Name | E-Mail |
|  |  |
|  |  |

|  |
| --- |
| **FFT/IFFT algorithm acceleration** |

**Source code C files (only excerpts from the parts that have been changed; please add line numbers to the code and highlight important parts):**

|  |
| --- |
|  |

**Source code TIE files (please add line numbers to the code):**

|  |
| --- |
|  |

**Questions:**

Which acceleration technique(s) has/have been used and why?

|  |
| --- |
|  |

Which part of the FFT algorithm did you accelerate and why?

|  |
| --- |
|  |

What is the impact on execution time (speedup)?

|  |
| --- |
|  |

What is the impact on the Hardware (Area)?

|  |
| --- |
|  |

What is the impact on the software (new instructions, modified source code, compiler intrinsics)?

|  |
| --- |
|  |

**Detailed Report (min. 1 page, max. 3 pages):***Step-by-step description of each optimization step that has been tried (even if no performance increase could be achieved):*

* *What software optimization or hardware acceleration strategies you tried?*
* *Refer your explanations to the C-/TIE-code making use of the lines numbers*
* *How did the performance increase (or decrease)? If you obtain a performance decrease, explain why and due to which acceleration method.*
* *What are the costs for the performance increase?*
* *Make a trade off performance vs. increased costs.*

*Conclude with an overall summary: Which combination of optimization and hardware acceleration strategies do you suggest, i.e., yields the best performance results or speed/area tradeoff? Summarize the experiences you gained with this task.*

|  |
| --- |
|  |