FPGA acceleration of DNA sequence pattern matching: 4th year Informatics Honours project plan

Vytautas Mizgiris, s1451552

October 6, 2017

1 Introduction

For my Honours Project, I will investigate ways of accelerating **DNA sequence comparison and matching**, a method commonly used in bioinformatics, on a dedicated FPGA, implement an approach proposed in recent studies and test and evaluate the speedup gained over a software-based solution. I will base my project on existing research in the field of pattern matching, FPGA logic design and specifically DNA matching using an FPGA processor as an accelerator.

2 Research

Papers whose studies on pattern matching engine design for an FPGA are candidates for the implementation:

- DNA Sequence Matching System Based on Hardware Accelerators Utilized Efficiently in a Multithreaded Environment (Khan, Aurangzeb, Khan, 2009). Algorithm: a variation of dynamic programming (DP).
- A Scalable Parallel Reconfigurable Hardware Architecture for DNA Matching (Segundo, Nedjah, Mourelle, 2013). Algorithm: heuristic, BLAST (Basic Local Alignment Search Tool). Includes algorithm pseudo-code, proposed micro-architecture and performance test results
- A study of the partitioned dynamic programming algorithm for genome comparison in FPGA (Hu, Georgiou, 2013). Explores a potential for using an improved partitioned DP algorithm for pattern matching in an FPGA. Material for possible future elaboration on the technique / extension of project, if chosen to work on implementing the dynamic programming approach.

Other sources, related to benchmarking the performance of a desktop CPU for the same / similar pattern matching technique:

- Benchmarking tools for the alignment of functional noncoding DNA (Pollard, Bergman, Stoye, Celniker, Eisen, 2004). Studies tools: Avid, BlastZ, Chaos, ClustalW, DiAlign, Lagan, Needle, and WABA
- NSimScan: DNA Comparison Tool with Increased Speed, Sensitivity and Accuracy. Cited from the introductory article: 'NSimScan outperforms industry standard tools in combined sensitivity, accuracy and speed, operating at sensitivity similar to BLAST, accuracy of ssearch and speed of MegaBLAST.' Source: `https://github.com/abadona/qsimscan`

3 Plan

The plan along with draft notes on the project, that will later be a part of the final write-up, are going to be changed and revised over time.

- **Done:** Setup the Digilent™ Zynq™-7010 board to run embedded Linux, connect to the network using an ethernet cable and SSH into the machine. Setup a `webmin` interface for easier access to the system.
- **Up next:** Find an FPGA interface driver and set it up to access FPGA fabric
- **In progress:** Evaluate existing research in pattern matching algorithms (comparison in performance and memory consumption) and implementing a pattern matching algorithm on the FPGA. Investigate algorithms based on dynamic programming and heuristics (see Research).
- **In progress:** Write a brief introduction (draft for an abstract later on) for the project, i.e. write up the context behind the choice in pattern matching algorithm and the main goals of the project
- **In 2-4 weeks time:** Source the logic diagram for a possible hardware implementation of the selected algorithm. Take note of required FPGA resources for the implementation
- **In 3-4 weeks time:** Star implementing the described logic.
- **In 3-8 weeks time:** Optional: modify the FPGA driver code to fit the specifics of the pattern matching logic that will need to be implemented

4 During the research and development

Continue by writing down the implementation and progress details, during and after the implementation process. Note any major or minor accomplishments in the way, as well as problems that arise, or findings that do not directly relate to the final goal but are worth mentioning. Test the final project code with relevant examples and evaluate the performance by comparing it to a DICE machine running a software logic emulation. Optionally compare the results to any research that was done previously. Make deductions. Write the conclusion.