A Massively Parallel Architecture for Bioinformatics

Gerd Pfeiffer¹, Stefan Baumgart¹, Jan Schröder², and Manfred Schimmmler²

¹ SciEngines GmbH, 24118 Kiel, Germany
<http://www.sciengines.com>

² Christian-Albrechts Universität, Department of Computer Science,
Hermann-Rodewald-Str. 3, 24118 Kiel, Germany
<http://www.informatik.uni-kiel.de>

Abstract. Today's general purpose computers lack in meeting the requirements on computing performance for standard applications in bioinformatics like DNA sequence alignment, error correction for assembly, or TFBS finding. The size of DNA sequence databases doubles twice a year. On the other hand the advance in computing performance per unit cost only doubles every 2 years. Hence, ingenious approaches have been developed for putting this discrepancy in perspective by use of special purpose computing architectures like ASICs, GPUs, multicore CPUs or CPU Clusters. These approaches suffer either from being too application specific (ASIC and GPU) or too general (CPU-Cluster and multicore CPUs). An alternative is the FPGA, which outperforms the solutions mentioned above in case of bioinformatic applications with respect to cost and power efficiency, flexibility and communication bandwidths. For making maximal use of the advantages, a new massively parallel architecture consisting of low-cost FPGAs is presented.

1 Introduction

Bioinformatics algorithms are most demanding in scientific computing. Most of the times they are not NP-hard or even of high asymptotic complexity but the sheer masses of input data make their computation laborious. Furthermore, the life science thus bioinformatics research field is growing fast and so is the input data that wait to be processed. Besides the massive amount of data to be processed, the field of bioinformatics algorithms have another characteristic in common: simplicity. Algorithms for the big areas of bioinformatics like sequence alignment, motif finding, and genome assembly that are used all around the world are impressively simple in their computation. We refer to Smith, Waterman [9], Needleman, Wunsch [10], or BLAST [11] as examples for simple yet effective and popular sequence alignment algorithms; expectation maximization (EM/MEME) [12], projection algorithm [13] for motif finding; EulerSR [8] and Edena [7] for genome assembly. All these programs use only very simple but repetitive operations on the input data: very simple arithmetic operations, string matching and comparison.

Standard CPUs are designed for providing a good instruction mix for almost all commonly used algorithms [2]. Therefore, for a class of target algorithms they can not be as effective as possible in terms of the design freedom. The same with the described purpose: here the arithmetic requirement is too simple to demand a full CPU and the string matching does not fit well at all to the hardware structure of a CPU. Hence many clock cycles will be consumed even for simple comparisons of nucleotides. Also, most of the algorithms above have a trivial approach to parallelization inherent another characteristic that does not fit too well with standard CPUs. The result of this are high runtimes or necessity for PC clusters, massive consumption of computer memory, and not to forget a vast amount of energy used - thus bare money to be spent.

This motivates the use of special purpose hardware. Several approaches exist to introduce special hardware to bioinformatics: Special ASIC [16], GPUs [17] and FPGAs [19] [20] [21] for example. FPGAs are to be named in particular because they can be adjusted exactly to the demands of an application like an ASIC and yet being able to readjust if the application changes slightly or even completely which is a powerful ability in a fast changing field like bioinformatics with a lot of different applications as mentioned above [1]. In this article we present a massively parallel special purpose hardware based on low cost FPGAs called COPACOBANA 5000. Its architecture and name is based on a similar machine presented earlier. It is the Cost-Optimized PARallel COde Braker ANd Analyzer (COPACOBANA) 1000 [4] The original purpose of this machine was cryptanalysis: fast code breaking of the DES standard and some further attacks [3]. It also proved to be applicable for bioinformatics though, allowing speedups of a factor 10,000 at a fraction of the energy costs in motif finding application compared to a single PC [14] [15].

The new architecture presented here makes use of the high applicability of FPGA chips in bioinformatics. The design goal is gaining a maximal speedup by maximizing the number of chips working together at low hardware costs and low energy consumption. The connection of the chips is realized by a sophisticated bus system allowing as much throughput as possible in order to be able to deal with the huge amounts of input data.

This paper is organized as follows: In Section 2 a first approach to massive parallelization with FPGAs is shown. Section 3 describes the new architecture allowing higher data throughput and bigger FPGA chips to challenge the big problems of bioinformatics in short time. In Section 4 the applicability and the estimated speedups gained and energy saved by this architecture are shown. Section 5 concludes the paper.

2 Copacobana 1000

COPACOBANA 1000 (Fig. 1) is a massively parallel reconfigurable architecture consisting of 120 low-cost FPGAs. Its hardware architecture has been developed according to the following design criteria: First, it was assumed that computationally costly operations are trivial parallelizable in terms of interprocess

Fig. 1. The Copacobana 1000 machine

Fig. 2. The Copacobana 1000 architecture

communication requirements. Second, the demand for data transfer between host and nodes is low due to the fact that computations heavily dominate communication requirements. A communication of low bandwidth between the parallel machine and a host computer is transferring instructions and results. Hence, a single conventional computer as front end is sufficient to transfer the required data packets to and from the acceleration hardware. The COPACOBANA is connected by a local area network (LAN) interface. Third, all target algorithms and their corresponding hardware nodes demand for very little local memory, which can be provided by the on-chip RAM modules of an FPGA.

Since the cryptanalytical applications demand for plenty of computing power, a total of 120 FPGA devices on the COPACOBANA cluster have been installed. By stripping down the hardware functionality of COPACOBANA to the bare minimum, an optimal cost-performance ratio for code breaking has been achieved. For the optimal cost-performance ratio, the system was designed

on basis of small FPGA chips which come with a high ratio of logic resources per cost. Further optimization on system level lead to a partitioning into 20 subunits that can be dynamically plugged into a backplane. Fig. 2 illustrates this architecture. Each of these modules in DIMM form factor hosts six low-cost Xilinx Spartan-3 XC3S1000 FPGAs that are directly connected to a common 64-bit data bus on-board. The data bus of the module is interfaced to the global data bus on a backplane. While disconnected from the global bus, the FPGAs on the same module can communicate via the local 64-bit data bus. Additionally, control signals are run over a separate 16-bit address bus. For simplicity, a single master bus was selected to avoid interrupt handling. Hence, if the communication scheduling of an application is unknown in advance, the bus master will need to poll the FPGAs. The front end of COPACOBANA is a host PC that is used to initialize and control the FPGAs, as well as for accumulation of results. Programming can be done in different levels for all or one specific subset of FPGAs. Data transfer between FPGAs and a host PC is accomplished by a dedicated control interface. This controller has also been designed as a slot-in module so that COPACOBANA can be connected to a computer either via a USB or Ethernet controller card. A software library on the host PC provides low-level functions that allow for device programming, addressing individual FPGAs, and storing and reading FPGA-specific application data. With this approach, more than one COPACOBANA device can easily be attached to a single host PC.

3 Copacobana 5000

The new architecture COPACOBANA 5000 consists of an 18 slot backplane equipped with 16 FPGA-cards and 2 controller cards. The latter connect the massively parallel FPGA-computer to an in-system off-the-shelf PC. Each of the FPGA-cards carry 8 high performance FPGAs interconnected in a one dimensional array. Additional units are supporting the mentioned functional units as for example a $1.5kW$ ¹ main power supply unit, 6 high-performance fans and a 19-inch rack of 3 high units (3HE) for the housing.

3.1 Bus Concept

The interconnection between the individual FPGA-cards and between the FPGA-cards and the controller is organized as a systolic chain:

There are fast point-to-point connections between every two neighbors in this chain. The first controller communicates with the first FPGA on the first card and the last FPGA on the last card is connected to the second controller. To speed up global broadcasts there are possible shortcuts in the chain between adjacent FPGA-cards. The point-to-point interconnections consist of 8 pairs of wires in each direction. Each pair is driven by low voltage differential signalling

¹ On output site: 125A + 12V.

Fig. 3. The Copacobana 5000 System

(LVDS) with a speed of 250MHz, thus achieving a data-rate of 2Gbit/s. Figure 3 shows the overall architecture of COPACOBANA 5000.

3.2 Controller

The root entity of control is running on a remote host computer. This machine is integrated into a local area network (LAN) which allows to connect to the COPACOBANA 5000. Two scenarios are considered here. The first is that the database is physically located inside the COPACOBANA machine e.g. on an SATA hard drive attached to the embedded PC. In this case the remote computer functions as user terminal only. The second scenario is that the embedded PC is used for an easy access to standard interfaces only. Here 2 Gigabit Ethernet LAN ports which is standard for recent main boards are of interest for the second scenario. The database has to be transferred over the LAN. In any case the remote host computer is initiator of every activity. The next instance is the embedded PC. Here the control information is translated for COPACOBANA and dropped into the system. In most cases it is not required for the higher level control instance to check if a command is executed successfully or not in the lower level of control hierarchy. However, it is possible if demanded by the implemented algorithm for example.

One board is plugged into a PCIe interface connector of the embedded PC. On this card an FPGA transceives the data and control by a proprietary wire

Fig. 4. Data Path of the Copacobana 5000 FPGA-Card

connection to the second half of the controller. Here another FPGA decodes the incoming data and instructions and drops them into the bus system of COPACOBANA 5000.

3.3 FPGA-Card

The FPGA-card is depicted in Fig. 4. It consists of 8 FPGAs of the type Xilinx Spartan-3 5000 for running the application and an additional FPGA which comes with a fixed configuration. The latter is routing the systolic datastreams. This simple protocol provides address information in header fields of the data stream. All FPGAs are globally clocked synchronously. Each clock cycle the data are transferred from one FPGA to the adjacent one building a huge communication pipeline. Inside the system this systolic data flow allows a throughput of 2Mbit/s for each direction. The protocol acquires an overhead of 20%. Between the controller-cards and the embedded PC the maximum data-rate is limited to 250MByte/s due to the PCIe connection. The disadvantage of the high throughput is a considerable amount of latency depending on the path the data are travelling.

3.4 Backplane

The backplane is holding the plugged cards mechanically, generating and distributing the clock signals, distributing the power, and finally connecting the cards for communication. The FPGA-card can transfer the incoming data to the next slot or it can take the data out of the stream. In this case an empty data field will go systolically through the bus pipeline from slot to slot. Another card can insert new data into this empty slot. The two counterrotating systolic dataflows allow to minimize the worst case latency to half of the total number of slot times the clock cycle time. But instead of a single master shared bus, the new machine connects one ascending slot and one descending slot to each single card. This leads to a ring of point to point connections. As the system is globally clocked the incoming data are registered. As result the bus system is working similar to a parallel shift register.

3.5 Memory

The big problem of using FPGAs in bioinformatics is the memory consumption. As mentioned in the introduction, the volume of input data is huge. FPGAs on the contrary are rather small in memory capacity. Our approach to solve this disparity is to use the connecting bus as a kind of memory replacement for the input data. Most of the mentioned algorithms have in common that they use only a small portion of memory (kilobytes) to store intermediate and end results. On the other hand they have to access huge amounts of data (gigabytes) to generate these results. The latter data needed for the application will be provided on the bus so every chip has access to the input data without storing it on chip site. However, some algorithms demand quite a lot of memory for the intermediate results as well some sequence alignment or assembly algorithms for example. To be able to handle those algorithms too (for a decent set of input parameters) we provide external local memory located next to each FPGA chip to guarantee applicability for most purposes.

In order to avoid a bottleneck, the raw input data have to pass the system with high throughput. Fortunately the target algorithms are latency insensitive due to the high locality of the parallelized bioinformatics algorithms. For ensuring a very high throughput in the design of the bus system, the I/O capabilities of the FPGA has to be analyzed carefully. The highest throughput can be achieved by connecting communicating chips by point-to-point lines of a short length [5] [6].

In addition, each FPGA is equipped with some external local memory. There is a 32MByte chip connected to each individual Spartan-3 5000 chip.

3.6 Software

Communication with COPACOBANA 5000 follows the principle of Memory-mapped-I/O (MMIO). By this a host software writes and reads data to and from addressed FPGAs, and inside to addressed IO-registers. The latter are application specifically used for data or control words. The control words are commonly incooperated into finite state machines. A communication framework builds a bridge between the two user programmed entities, the host software and the user FPGA core. Both are connected by a framework interface. At host site an Application Programming Interface (API) provides easy access to the machine by a set of communication library calls for the MMIO commands. The API supports Java and C++. On the other side of the framework interface an Relationally Placed Macro (RPM) has to be embedded into the user FPGA configuration. One side of this macro connects to the physical bus system via IO-blocks and inside the FPGA the macro connects to the user implementation as API for FPGA design, for example in VHDL. The common communication principle of Memory-mapped-I/O is easy to understand by the user. Together with the given communication framework a user does not need to know the COPACOBANA 5000 in detail. However, the knowledge about the architecture is helpful for implementing parallel algorithms efficiently.

Table 1. COPACOBANA Performance in Exhaustive Key Search on DES

	Intel Pentium-4 3.0GHz	Copacobana 1000	Copacobana 5000 (est.)
encryption speed	$2 \cdot 10^6 \frac{1}{s}$	$6.5 \cdot 10^{10} \frac{1}{s}$	$3.0 \cdot 10^{11} \frac{1}{s}$
average search time	571years	6d10h	1d10h
energy consumption	750MWh	93kWh	62kWh

Table 2. COPACOBANA Performance in Motif-Finding

	Intel Xeon 5150 2.6GHz dual core	Copacobana 1000	Copacobana 5000 (est.)
Cowpox	144h	1h40m	21m
Virus (230kbp)	21.6kWh	1.0kWh	0.63kWh
Rickettsia	2,575h	4h10m	52m
canadensis (1.2Mbp)	386.2kWh	2.5kWh	1.6kWh
Bacillus	11,236h	16h45m	3h30m
subtilis (5.1Mbp)	1685.4kWh	10.1kWh	6.3kWh

kbp : kilo base pairs, Mbp : Mega base pairs.

4 Performance Estimation

In this section the performance is compared between a PC, a COPACOBANA 1000 and the estimated performance of the COPACOBANA 5000.

The COPACOBANA 1000 originally was intended for running cryptanalytical applications. In Table 1 the performance of an exhaustive key search on DES is compared between an Intel Pentium-4 3.0GHz and the implementation on the COPACOBANA 1000. The estimated values for the COPACOBANA 5000 are simply based on the number of FPGAs and the size of the chip. Each Spartan-3 5000 comes with 4.5 times of the logic resources compared to the Spartan-3 1000. Furthermore the new machine hosts 128 user FPGAs and the old one 120. Hence, the new machine has approximately 4.8 times more computing performance. This assumption is legitimately due to fact that both chips are based on the same technology and are different in the size only.

Despite that fact, a proof of concept for the applicability in bioinformatics has been developed. In Table 2 the results of this research are shown. The target application is motif finding on DNA sequences. The implementation on the COPACOBANA 1000 has been tested and the performance compared to a standard computer has been measured. The estimation of the performance of the COPACOBANA 5000 is based on the number and size of FPGAs as explained above. Probably the performance will exceed these values due to the optimized bus.

One of the most interesting aspects is the one of power consumption. COPACOBANA 1000 and COPACOBANA 5000 are extremely power efficient for the considered applications. Observe that already a single run of one exhaustive key search on the Data Encryption Standard (DES) by use of PCs ² saves costs of power consumption in the scale of the purchase costs of a COPACOBANA 5000.

5 Conclusion

Bioinformatics applications are computationally extremely demanding. It is fair to believe that analyzing biological research data as for example DNA sequence data with conventional PCs and super computers is far too expensive. The only promising way to tackle existing computing machines is to build special purpose hardware, dedicated solely to suitable algorithms such as those presented in this paper.

Conventional parallel architectures turn out to be far too complex and, thus, are not cost efficient in solving bioinformatics problems. Most of these problems can be parallelized easily and we show the architecture of the recent design the COPACOBANA 5000 which results from the algorithmic requirements of the targeted cryptanalytic problems.

Recapitulating, the COPACOBANA machines are the first and currently the only available cost efficient massively parallel FPGA-computers. The COPACOBANA 1000 was intended to, but is not necessarily restricted to solving problems related to cryptanalysis. A proof of concept confirmed the applicability for high-performance bioinformatics computing.

The work at hand presents the design and architecture of a cost efficient advancement of the old design fulfilling the request for bioinformatics computing. The COPACOBANA 5000 will host 128 low-cost FPGAs. We showed, by extrapolating a successfully implemented proof of concept on the COPACOBANA 1000 that the new hardware architecture will reach increased performance by a factor of five compared to the old machine and better than a standard computer in orders of magnitude.

Future work includes completion and optimization of all performance relevant design issues. The implementations have to be optimized to guarantee best possible throughput. The first prototype of COPACOBANA 5000 is to be presented in May 2009.

References

1. DeHon, A.: The Density Advantage of Configurable Computing. *IEEE Computer Magazine* 33(4), 41–49 (2000)
2. Hennessy, J.L., Patterson, D.: *Computer Architecture: A Quantitative Approach* (1995)
3. Güneysu, T., Kasper, T., Novotny, M., Paar, C., Rupp, A.: Cryptanalysis with COPACOBANA. *Transactions on Computers* 57, 1498–1513 (2008)

² Cluster of Intel Pentium-4 3.0GHz.

4. Kumar, S., Paar, C., Pelzl, J., Pfeiffer, G., Schimpler, M.: COPACOBANA - A Cost-Optimized Special-Purpose Hardware for Code-Breaking. In: IEEE Symposium on Field-Programmable Custom Computing Machines - FCCM 2006, Napa, California, April 24-26 (2006)
5. Montrose, M.I.: EMC and the Printed Circuit Board. IEEE Press, New York (1999)
6. Montrose, M.I.: Printed Circuit Board Design Techniques for EMC Compliance, 2nd edn. IEEE Press, New York (2000)
7. Hernandez, et al.: De novo bacterial genome sequencing: Millions of very short reads assembled on a desktop computer. *Genome Res.* 18(5), 802–809 (2008)
8. Chaisson, P.: Short read fragment assembly of bacterial genomes. *Genome Research* (December 2007)
9. Smith, T.F., Waterman, M.S.: Identification of common molecular subsequences. *Journal of Molecular Biology* 147, 195–197 (1981)
10. Needleman, S.B., Wunsch, C.D.: A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of Molecular Biology* 48(3), 443–453 (1970)
11. Altschul, S.F., Gish, W., Miller, W., Myers, E.W., Lipman, D.J.: Basic Local Alignment Search Tool. *Journal of Molecular Biology* 215(3), 403–410 (1990)
12. Bailey, T.L., Elkan, C.: Unsupervised Learning of Multiple Motifs in Biopolymers Using Expectation Maximization. Techn. Report, Department of Computer Science and Engineering, University of California, San Diego, La Jolla, California, USA (1993)
13. Buhler, J., Tompa, M.: Finding Motifs Using Random Projections. *Journal of Computational Biology* 9(2), 225–242 (2002)
14. Schroeder, J., Schimpler, M., Tischler, K., Schroeder, H.: BMA Boolean Matrices as Model for Motif Kernels. In: 2008 International Conference on Bioinformatics, Computational Biology, Genomics and Chemoinformatics (2008)
15. Schroeder, J., Wienbrandt, L., Pfeiffer, G., Schimpler, M.: Massively Parallelized DNA Motif Search on the Reconfigurable Hardware Platform COPACOBANA. In: Chetty, M., Ngom, A., Ahmad, S. (eds.) *PRIB 2008. LNCS (LNBI)*, vol. 5265, pp. 436–447. Springer, Heidelberg (2008)
16. Blas et al.: The UCSC Kestrel Parallel Processor. *IEEE Transactions on Parallel and Distributed Systems* 16(1) (January 2005)
17. Manavski, S.A., Valle, G.: CUDA compatible GPU cards as efficient hardware accelerators for Smith-Waterman sequence alignment. In: *BMC Bioinformatics 2008*, vol. 9(suppl. 2), p. 10, March 26 (2008)
18. Oliver, T.F., Schmidt, B., et al.: Multiple Sequence Alignment on an FPGA. *ICPADS* (2), 326–330 (2005)
19. Oliver, T., Schmidt, B., Maskell, D.: Reconfigurable Architectures for Bio-sequence Database Scanning on FPGAs. *IEEE Transactions on Circuits and Systems II* 52(12), 851–855 (2005)
20. Oliver, T., Schmidt, B., Nathan, D., Clemens, R., Maskell, D.: Using reconfigurable hardware to accelerate multiple sequence alignment with ClustalW. *Bioinformatics* 21(16), 3431–3432 (2005)
21. Oliver, T., Leow, Y.Y., Schmidt, B.: Integrating FPGA Acceleration into HMMer. *Parallel Computing* 34(11), 681–691 (2008)