Deploying FPGAs to Future-proof Genome-wide Analyses based on Linkage Disequilibrium

Dimitrios Bozikas[1], Nikolaos Alachiotis[1,2], Pavlos Pavlidis[2], Evripides Sotiriades[1], and Apostolos Dollas[1]

[1]School of Electrical and Computer Engineering, Technical University of Crete, Chania 731 00, Greece
[2]Institute of Computer Science, Foundation for Research and Technology - Hellas, Heraklion 700 13, Greece

***Abstract*—The ever-increasing genomic dataset sizes, fueled by continuous advances in DNA sequencing technologies, are expected to bring new scientific achievements in several fields of biology. The fact that the demand for higher sequencing throughput has long outpaced Moore's law, however, presents a challenge for the efficient analysis of future large-scale datasets, suggesting the urgent need for custom solutions to keep up with the current trend of increasing sample sizes. In this work, we focus on a widely employed, yet prohibitively compute- and memory-intensive, measure that is called linkage disequilibrium (LD), defined as the non-random association between alleles. Modern microprocessor architectures are not well equipped to deliver high performance for LD due to the lack of a vectorized population counter (counting set bits in registers). We present a modular and highly parallel reconfigurable architecture that, in combination with a generic memory layout transform, allows to rapidly conduct large-scale pairwise calculations on arbitrarily large one- and two-dimensional binary vectors, exhibiting increased bit-counting capacity. We map the proposed architecture to all four reconfigurable devices of a multi-FPGA platform, and deploy them synergistically for the evaluation of LD on genomic datasets with up to 1,000,000 sequences, achieving between 12.7X (4 FPGAs vs. 12 cores) and 134.9X (4 FPGAs vs. 1 core) faster execution than state-of-the-art reference software running on multi-core workstations. For real-world analyses that employ LD, such as scanning the 22nd human chromosome for traces of positive selection, the proposed system can lead to 6X faster processing, thus enabling more thorough genome-wide scans.**

I. INTRODUCTION

Recent years have witnessed significant breakthroughs in DNA sequencing technologies, reducing sequencing costs while improving accuracy and throughput. Consequently, vast amounts of genomic data are currently produced, contributing to dataset sizes that comprise thousands of whole genomes. While the–currently active–100,000 Genomes Project [1], for instance, is aiming at having sequenced 100,000 human genomes by the end of 2017, its predecessor, the 1,000 Genomes Project [2], was completed in 2015 having sequenced 2,504 human genomes. It is predicted that Genomics will be one of four dominant Big Data domains by 2025, along with Astronomy, Twitter, and YouTube, having sequenced between 100 million and 2 billion human genomes alone [3]. Currently, sequencing capacity doubles every seven months [3], leading to ever-increasing dataset sizes. This suggests the urgent need for custom solutions to ensure the efficient processing of future large-scale datasets without potential disruptions by computational insufficiencies.

In this work, we focus on a widely employed statistic in population genomics called linkage disequilibrium (LD) [4]. LD describes the non-random association between alleles[1] at different loci in a population, allowing to capture correlations between mutations that are inherited together. Computing LD has several practical applications, either when employed alone or as the basis for downstream analyses. Estimating the extent and shape of LD in humans [5], for instance, has implications in disease gene mapping, since it incorporates the effect of past evolutionary events. When the aim is to detect traces of positive selection, relying on LD achieves higher accuracy and sensitivity than alternative approaches [6], allowing to more accurately localize drug-resistant mutations in pathogens, thus paving the way for the design of more effective drugs [7]. In genome-wide association studies (GWAS) [8], where the focus is on identifying mutations associated with diseases, LD serves as a powerful optimization technique to identify highly correlated loci to prevent redundant computations.

LD calculations are conducted on so-called multiple sequence alignments (MSAs), i.e., two-dimensional matrices, $m \times l$, that comprise m DNA sequences of l nucleotides each. Alignment columns that contain two or more states, or in other words, at least one mutation has occurred at that location, are called single-nucleotide polymorphisms (SNPs). Only SNPs are informative for computing LD, with the remaining–non-polymorphic–sites discarded prior to an analysis. The time complexity of computing all pairwise LD scores for a number of SNPs n and a number of sequences m is $O(m \times n^2)$, with the number of SNPs increasing with an increasing sample size due to more genetic variants being discovered [8].

The current trend of increasing the sample size entails multiple challenges for LD calculations, both compute- and memory-related ones. On one hand, memory requirements for processing SNPs and storing LD scores increase quadratically with an increasing number of SNPs, an inevitable consequence of the sequencing of new genomes and the subsequent discovery of new genetic variants. On the other hand, LD requires the calculation of allele and haplotype frequencies, computations that heavily rely on the enumeration of set bits in registers (population count operation). As the sample size increases, population count operations are conducted on longer bit vectors, yielding bit enumeration the performance bottleneck for LD, even when computed for relatively small sample sizes, e.g., 1,000 sequences.

In addition to the computational ramifications that large sample sizes bring to LD, the fact that Moore's law is being continued by adding more cores in a processor and by widening the SIMD vector width brings little benefit to computing

[1]An allele is one of at least two alternative forms of a gene that are caused by one or more mutations.

LD, due to the lack of a vectorized population counter in current microprocessor architectures. This has been shown [9] for the simplest application of LD, which implements the so-called infinite sites model (ISM) [10]. Under the ISM assumption, no more than one mutation per genetic location is possible, which allows to encode each DNA state with a single bit. In this work, we focus on the more realistic and significantly more compute-intensive finite sites model (FSM) [11], motivated by the increasing interest that the particular evolutionary model receives in recent biological studies [12]. The FSM model allows for more than one mutation per genetic location, thus requiring more than one bit per DNA state to avoid loss of information. The contribution of this work is two-fold:

- We present a generic FPGA-based architecture to accelerate LD, with particular emphasis on supporting arbitrarily large numbers of whole genomes. With sample sizes of future datasets in mind, we analyze the design space of the proposed architecture and discuss performance-critical design decisions. The proposed accelerator architecture computes both the FSM and the ISM models, accounting for DNA ambiguous characters (to correct for DNA sequencing errors and nucleotide base mis-calls [13]), while considering missing data along the genomes.
- We develop a fully fledged, single-FPGA prototype system, as well as a performance system that exploits four reconfigurable devices synergistically, coupling the LD accelerators in both cases with an efficient parallel algorithm for computing LD at the whole-genome scale. This allows to deploy hardware acceleration efficiently for genome-wide analyses that comprise a large number of full genomes. To demonstrate this, and in addition to a series of performance comparisons based on simulated genomic data, we estimate possible performance gains from the potential deployment of our performance system in a real-world analysis for the detection of traces of positive selection in the 22nd chromosome of the human genome.

The remainder of this work is organized as follows. In Section II we describe the mathematical operations and concepts for computing LD under both the ISM and the FSM evolutionary assumptions. In Section III we review related work on software and hardware solutions for population genomics and linkage analyses. Section IV describes the proposed accelerator architecture, while Section V provides implementation details. Section VI presents performance comparisons with relevant software and hardware implementations, while Section VII concludes this work.

II. LINKAGE DISEQUILIBRIUM (LD) STATISTICS

Linkage disequilibrium (LD) quantifies the non-random association between alleles at different genomic locations. In the absence of evolutionary forces acting on the genomes during reproduction, such as recombination, mutation, genetic drift etc, all alleles that reside on the same chromosome are inherited to the offsprings. In reality, however, several evolutionary forces contribute to the genetic constitution of the offsprings' generation. LD is employed to identify which allelic combinations co-occur in a genome more frequently than expected by chance.

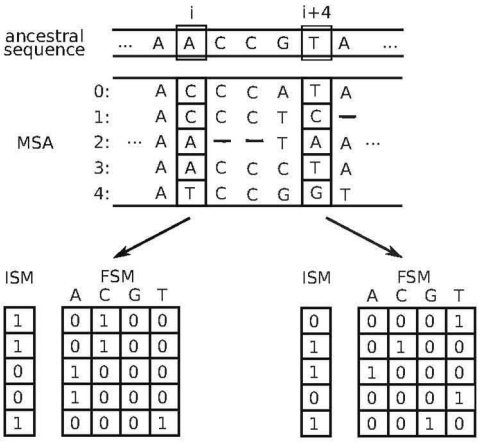

Fig. 1. Example of a MSA that consists of an arbitrary number of sites, including two SNPs at locations i and $i+4$, along with their respective representations under the ISM and the FSM evolutionary models.

A. Data preparation and SNP representation

LD computations are conducted on data structures that consist solely of SNPs, which we henceforth refer to as SNP maps. A SNP map is therefore a description of the existing mutations in a population along with their respective locations in the chromosome. The first steps for creating a SNP map entail the sequencing of the genetic material of the individuals under investigation, and the consequent mapping of the produced short DNA reads to a reference genome. While the former generates a DNA sequence per individual, the latter produces a multiple sequence alignment (MSA) for the population under investigation (see Fig. 1). The SNP map is generated by a so-called "SNP calling" procedure, which identifies the SNPs (note that invariable sites are not informative for LD).

Depending on the assumption that drives the occurrence of mutations, i.e., ISM vs. FSM, a SNP is either represented by a single binary vector (under ISM) or by a two-dimensional matrix (under FSM), as illustrated in Figure 1. To identify mutations under ISM, where a single binary vector per SNP is used and set bits indicate mutations, each sequence in the MSA is compared, on a per-allele basis, with an ancestral sequence S. Given an allele a_i at location i in the ancestral sequence, the binary vector V describing SNP i is constructed as follows, where [ ] is the Iverson bracket notation:

$$V_i^{a_i} = \{[V_{i,0} = a_i], [V_{i,1} = a_i], ..., [V_{i,N-1} = a_i]\}. \quad (1)$$

Based on Equation 1, a SNP is described under the FSM assumption as follows:

$$M_i = \{V_i^A V_i^C V_i^G V_i^T\}. \quad (2)$$

Note here that, a set bit does not indicate similarity with the corresponding location in the ancestral sequence, as was previously the case for a binary vector under ISM. In the above SNP description, each bit indicates one of the nucleotide bases, i.e., adenine (A), guanine (G), cytosine (C), and thymine (T).

The astute reader will have noticed by now that the above representation is seemingly not optimal since 2 bits would have been sufficient to accurately represent the 4 DNA bases. In reality, however, DNA sequencing errors and base mis-calls [13] do not allow to accurately determine which of the 4 nucleotides exists at each location in the genome.

Therefore, a total of 16 ambiguous DNA characters (see *http://www.bioinformatics.org/sms/iupac.html*) are employed in order to allow subsequent statistical analyses to account for such ambiguity.

B. Allele and haplotype frequencies

From a computational standpoint, a pairwise LD calculation between two SNPs consists of two distinct parts, the calculation of allele and haplotype[2] frequencies followed by the calculation of a score according to an LD measure. For the sake of clarity, we initially describe LD when the ISM assumption holds, and introduce thereafter the additional operations required by the FSM model.

For a SNP at location i, the allele frequency F under ISM is computed as follows:

$$F_i = \frac{\sum_{k=0}^{N-1}[V_{i,k} = 1]}{N}, \tag{3}$$

where N is the number of sequences in the dataset. This operation entails the enumeration of set bits at locus i, or in other words, the enumeration of mutated alleles. The haplotype frequency H, which is computed on a per-SNP-pair basis, is given for a pair of SNPs i and j by the following equation:

$$H_{ij} = \frac{\sum_{k=0}^{N-1}[V_{i,k} = 1][V_{j,k} = 1]}{N}. \tag{4}$$

Here, the numerator computes the total number of genomes that exhibit the mutated allele at both locations i and j.

C. Measures of Linkage Disequilibrium

LD relies on the probability of independent events. Once the allele and haplotype frequencies for SNPs i and j are known, we can then compute LD as follows:

$$D_{ij} = H_{ij} - F_i F_j. \tag{5}$$

This calculates the difference between the probability of observing two mutations at two chromosomal locations in the same genome and the product of the probabilities of observing the two mutations independently. When $D_{ij} = 0$, SNPs i and j are in linkage equilibrium, indicating that mutations at chromosomal locations i and j occur independently of each other. When $D_{ij} \neq 0$, SNPs i and j are in linkage disequilibrium, and the mutations at chromosomal locations i and j are not independent.

The above formulation of LD is rarely used due to the fact that the sign and range of values that D assumes vary with the frequency at which mutations occur, yielding the comparison of LD values highly problematic, even between genomic regions in the same genome. For this reason, several standardization approaches have been proposed, with a widely used one relying on the Pearson's correlation coefficient, as described below:

$$r_{ij}^2 = \frac{(H_{ij} - F_i F_j)^2}{F_i(1 - F_i)F_j(1 - F_j)}. \tag{6}$$

Employing r^2 as a measure of LD has multiple advantages since it assumes values between 0 and 1, which allows to trivially compare different genomic regions. Furthermore, interpreting the scores becomes rather intuitive since values closer to 1 indicate high LD, whereas values closer to 0 indicate reduced association.

The discussion so far has been focused on the ISM model, which represents an approximation of the FSM assumption regarding the occurrence of mutations. This approximation allows for a simpler SNP representation (binary vector vs. two-dimensional matrix), which, evidently, leads to lower computational requirements and shorter analysis times. To compute LD between two SNPs, i and j, under FSM, assume two sets of DNA characters S_i and S_j that contain the alleles found in SNPs i and j, respectively, along with their sizes V_i and V_j. For the two SNPs in Figure 1, for instance, we have $S_i = \{A, C, T\}$ and $S_j = \{A, C, G, T\}$, and set sizes $V_i = 3$ and $V_j = 4$, respectively, since no occurrence of G is found in SNP i. LD is then computed based on the previous mathematical description for ISM as follows:

$$LD_{ij} = N\frac{(V_i - 1)(V_j - 1)}{V_i V_j} \sum_{m \in S_i} \sum_{l \in S_j} r_{ml}^2, \tag{7}$$

where, now, Equation 6 is calculated separately for each pair of columns in the corresponding M_i and M_j matrices. When an allele is completely absent from a SNP, as is the case for guanine in SNP i, the corresponding matrix column is omitted in the calculations. Therefore, in comparison with ISM, computing LD under FSM can require up to 16 times more computations in the worst case, which occurs when all alleles are present in all SNPs.

III. RELATED WORK

Driven by the disproportion between the rate at which molecular data are currently being accumulated and Moore's law, novel software and hardware designs have been recently introduced, addressing both performance and scalability issues. PopGenome, an R package released by Pfeifer et al. [14], offers a variety of population genetics statistics (including LD). Although parallel programming is employed to reduce processing times for the analysis of whole-genome data, high-end processor features, such as vector intrinsics or advanced caching techniques, are not appropriately exploited.

A significant update to the widely used PLINK software [15] (over 11,500 citations according to Google Scholar) for GWAS was recently released by Chang et al. [16]. The second-generation PLINK release (version 1.9) achieves major performance improvements over the initial software by heavily employing multiple cores and bit-level parallelism. LD is computed using an efficient calculation of the squared Pearson's correlation coefficient that relies on SSE2 vector intrinsics.

OmegaPlus, released by Alachiotis et al. [17], represents a scalable software implementation that relies on LD to target traces of positive selection on whole genomes. Similarly to PLINK 1.9, the squared Pearson's correlation coefficient is employed as the preferred measure of LD, while a series of four parallelization alternatives are provided to address load imbalance issues that derive from the way LD calculations are conducted when positive selection is detected.

Over the past decade, Bioinformatics gradually became fertile ground for custom hardware accelerators ([18], [19]). However, to the best of the author's knowledge, the sole study that addresses the computational issues of LD at the hardware level is the recent exploratory work by Alachiotis

[2]A haplotype is a set of mutations that are inherited together.

and Weisz [20]. In this study, the authors map a simplified LD kernel for ISM to a Virtex 7 FPGA, adopting various assumptions that are broadly valid only when simulated data are analyzed, such as complete genomic data (no missing data), as well as absence of ambiguous characters and alignment gaps. In addition, the employed processing scheme relies on preloading sets of SNPs to on-chip memory blocks, imposing that way an upper bound on the number of sequences that can be processed correctly, while yielding performance efficiency highly dependent on the amount of available on-chip memory.

In the current study, we enhance the existing LD acceleration landscape by introducing a complementary architecture that provides all the functionality that can potentially be required by a real-world analysis. More importantly, and from a biological standpoint, we eliminate the upper bound on the number of genomes that can be processed correctly. This yields a future-proof solution since the number of whole genomes in future biological studies is only expected to increase. Furthermore, it paves the way to effectively analyze thousands to millions of individuals, thus increasing accuracy of population genomics analyses [21] and the chances for discovering rare human diseases [22]. From a computational standpoint, the current work facilitates the exploration and understanding of the involved trade-offs between binary vector sizes and acceleration performance attained by current FPGA technology when an increased amount of pairwise calculations between elements represented by binary vectors is required.

IV. SYSTEM DESIGN

Computing LD is intrinsically a memory-bound operation, and maximizing the amount of operations per memory access is of critical importance to the overall performance of the accelerator architecture. In the following subsections, we initially describe a series of memory layout transformations (Section IV-A) that allow to retrieve SNP data from memory efficiently, as well as to effectively utilize hardware resources by maximizing parallelism in performance-critical stages, e.g., pairwise combinations and bit counting. Thereafter, we present our parameterized architecture (Section IV-B) and explain the design space (Section IV-C).

A. Memory layout transformations

We initially consider the OmegaPlus memory layout, which stores SNPs contiguously in memory, as shown in Figure 2A. Given a sample size of N sequences and a word width of w bits, each SNP is represented by a total of five memory blocks of size B each:

$$B = \left\lceil \frac{N}{w} \right\rceil, \tag{8}$$

with zero padding. The first four memory blocks correspond to the four nucleotide bases[3], whereas the last one indicates validity of the corresponding bits, accounting this way for missing data along the genomes. This layout is highly inefficient for acceleration purposes due to the fact that operations cannot proceed until data for all nucleotide bases and the validation bits are retrieved. Loading SNP data by iteratively accessing

[3]More compact DNA encoding is not feasible due to the fact that 16 valid DNA states are considered in real-world data, i.e., A, C, G, T, and ambiguous characters. Therefore, 4 bits per state are required.

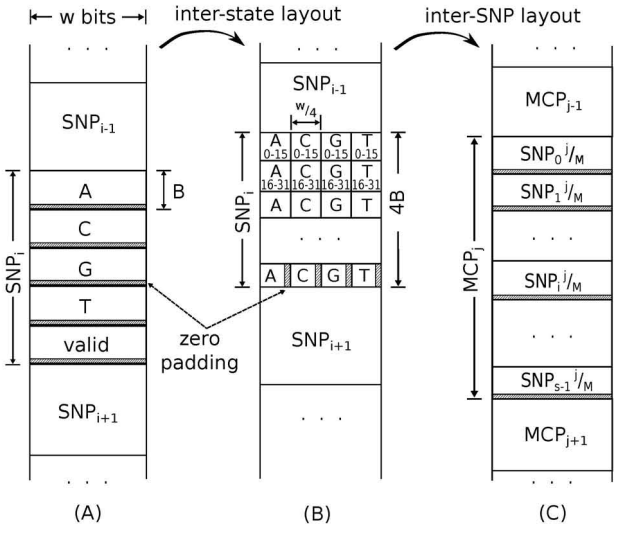

Fig. 2. The two-step transformation approach of the standard OmegaPlus memory layout (A) to an inter-state one (B), which facilitates processing of multiple states, and to an inter-SNP one (C), which allows shorter data retrieval times.

the five memory blocks requires longer overall retrieval times, as this essentially leads to accessing memory in strides that are far larger than the size of the memory pages (4KB).

The first memory layout transformation leads to the inter-state SNP representation illustrated in Figure 2B. Previously, all w bits in a word were indicating presence or absence of the same nucleotide base for w sequences. The transformed layout now comprises information for all DNA bases (A, C, G, T, and ambiguous characters) from $w/4$ sequences. Note the bit/sequence indices in Figure 2B, assuming a word width of 64 bits ($w = 64$). Additionally, note the absence of validation bits, which can be computed on the fly with a 4-bit logical *or* operation.

The second and final transformation, depicted in Figure 2C, interleaves data from all s SNPs within a genomic region of interest based on the number of input ports that the accelerator can deploy for SNP data retrieval. Remember that the accelerator architecture is designed to accommodate arbitrarily large sample sizes, where retrieving SNP data quickly becomes the performance bottleneck. Therefore, adopting a memory layout that allows to retrieve per-SNP data in parallel through multiple input ports is a prerequisite for the acceleration of large-scale LD calculations. In our design, SNP data are transferred from main memory to the accelerator via M pairs of memory controllers (MCPs), with each pair retrieving SNPs in pairs to facilitate the required pairwise calculations. Each MCP is being assigned a fraction of the total sample size, retrieving only the corresponding SNP part for all the SNPs in a region.

B. Accelerator architecture

The proposed accelerator architecture, illustrated in Figure 3, operates on a pairwise basis, calculating the LD score for a single pair of SNPs at each time. Processing starts with the counting of the mutated alleles (under ISM) or the DNA states (under FSM), achieved by the first two pipeline stages (see bottom of Fig. 3), denoted BC (Bit Count) Stages 1 and 2. Thereafter, the required allele and haplotype frequencies are

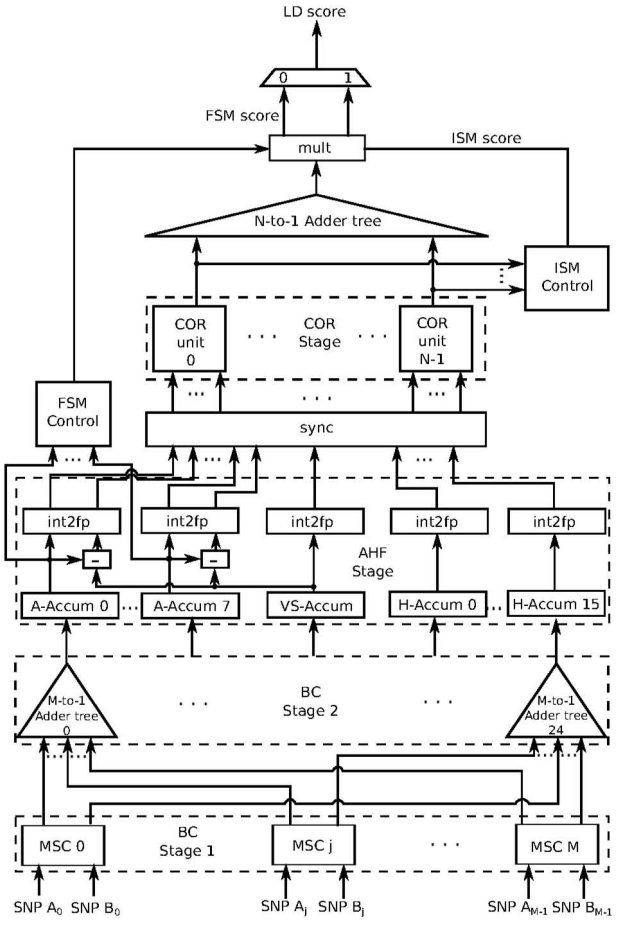

Fig. 3. Top-level design of the accelerator architecture. Processing proceeds from the bottom to the top, through four discrete pipeline stages, i.e., BC Stages 1 and 2 for the partial accumulation of population count results, AHF for a final accumulation and calculation of allele and haplotype frequencies, and a final COR stage for the floating-point correlation calculations.

computed in stage AHF (Allele/Haplotype Frequencies), and the required correlation values per pair of SNPs (under ISM) or per pair of DNA states of different SNPs (under FSM) are calculated in the final COR stage.

a) BC Stage 1: The first stage relies on multiple population counters operating in parallel to calculate the number of set bits in the incoming binary vectors/matrices. SNP data are provided via a number of $2M$ input ports, organized in M pairs to facilitate pairwise calculations. As already mentioned, each pair of ports (referred to as MCP in Section IV-A) retrieves a fraction of the entire SNP. Each MSC (Mutation/State Counter, Figure 4) initially calculates the validation bits that correspond to the pair of partially loaded SNPs and validates the SNP data. Thereafter, a total of 25 16-bit-wide population counters operate in parallel to count the set bits in each of the incoming SNP vectors and their combinations, i.e., 4 vectors for the 4 DNA states of SNP A, 4 vectors for the 4 DNA states of SNP B, one internally computed validation vector, and 16 vectors for all possible combinations of the per-state vectors for the pair of SNPs A and B. The number of valid sequences per SNP pair varies according to the amount of missing data along the genomes. Thus, counting the validation bits on a per-SNP-pair basis is a prerequisite for the correct calculation of LD scores along genomes with missing data. Each of the 16-bit

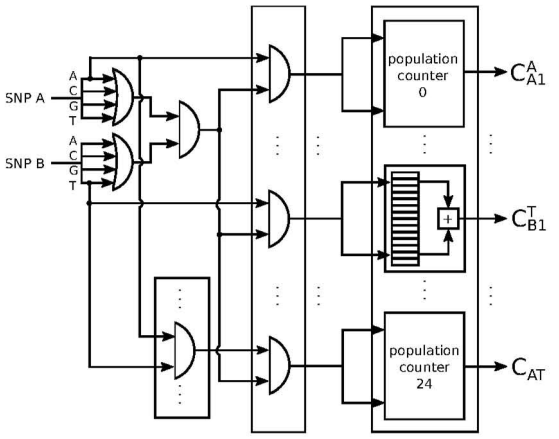

Fig. 4. Block diagram of the MSC unit. As the SNP length (number of genomes) increases, overall acceleration performance relies on the number of MSC units operating in parallel to boost population count capacity of the FPGA. Prior to the array of 16-bit population count blocks, dedicated logic calculates on the fly all the required pairwise combinations of the input SNPs.

population counters consists of a 4×256 dual-port ROM and a 5-bit adder, with each port providing the number of set bits for half of the input.

b) BC Stage 2: The previous stage computes 25 population count results per MCP, which are accumulated in the current stage with the use of 25 M-to-1 adder trees with increasing adder width per tree level. The 25 partial sums that are passed to the next pipeline stage reflect all mutations or states that have been previously counted by the M MSCs in BC Stage 1. This allows to proceed with the remaining computational steps in the following pipeline stages independently of the available number of input ports, thus facilitating a potential future migration to a different device/platform.

In addition to alleviating the migration overhead, organizing the enumeration step of alleles and haplotypes into a platform-specific (BC Stage 1) and a dataset-specific stage (BC Stage 2) allows to easily support other dataset types as well, such as RNA secondary structure (6, 7 states) or amino acids (20 states). This requires additional logic and population count instances in BC Stage 1, as well as additional M-to-1 adder trees in BC Stage 2, while the rest of the design remains largely intact.

c) AHF Stage: The partial per-state sums along with the validation bits are accumulated in the current stage with a total of 25 W-bit-wide accumulators. This additional accumulation stage allows to correctly calculate the number of mutations or DNA states in large-scale datasets that comprise thousands to millions of genomes, given that sufficiently wide accumulators are deployed. Once the final allele and haplotype counts are calculated by the A-Accum and H-Accum accumulators, having considered only the valid sequences for the given pair of SNPs, all accumulator outputs are converted to single-precision floating-point values, without loss of accuracy, to facilitate further processing. Note the subtractors after the A-Accum allele accumulators, which calculate the number of non-mutated alleles based on the number of mutated ones and the total number of valid sequences for the SNP pair. The number of valid pairs of bits in a SNP pair may vary, depending on the amount of missing data and alignment gaps in both SNPs. To yield comparable LD scores along different

genomic regions within the same chromosome, the valid SNP size per SNP pair is calculated on a pairwise basis as well, by the VS-Accum (Valid Size) accumulator.

d) COR Stage: The final stage of the accelerator pipeline computes a measure of LD, such as the squared Pearson's correlation coefficient. This is achieved via the instantiation of N floating-point pipelines (COR units) that operate in parallel, with each one of them implementing Equation 6 while calculating on a different pair of input data combinations, i.e., different DNA states under FSM or different SNPs under ISM. Note that, Equation 6 describes the Pearson's correlation coefficient as applied for the ISM model, while additional operations are required for FSM (N-to-1 adder tree, multiplier, see Equation 7). This allows to efficiently support both the ISM and the FSM models by facilitating the respective computational requirements using the same processing pipeline. If N is smaller than the required number of r^2 calculations, dedicated synchronization/scheduling logic ensures access to the available COR units in a round-robin fashion. The FSM Control calculates the numbers of valid states per SNP in the pair under examination, as required by Equation 7. Serializer logic is deployed prior to the output when ISM scores are calculated, due to the single output port that the accelerator architecture provides.

C. Design space analysis

LD is a memory-bound computation, relying on the rapid retrieval of data from memory and their efficient processing without delay. Given M pairs of input ports, an equal number of MSCs is required in BC Stage 1 in order to process data at the speed of arrival. As the number of sequences increases, allele/haplotype frequency computations become the limiting performance factor. Therefore, suboptimal processing/delays during the allele/state enumeration stages can have a profound effect on the success of the acceleration system.

In a similar manner, the number of pairs of parallel input ports dictates the size of the M-to-1 adder trees in BC Stage 2. The number of adder trees, however, as well as the number of accumulators in the AHF stage is driven by the data type. Assume, for instance, a data type of $S = 4$ states, as is the case for DNA. An equal number of S adder trees and, consequently, accumulators, are required for each of the two input SNPs, along with $S \times S$ pairs of adder trees and accumulators for all possible combinations of states for the two SNPs. Note that, the additional adder tree/accumulator pair, for the validation bits, is specific to the proposed architecture, trading hardware resources for a smaller memory footprint and shorter data retrieval times. The accumulator width needs to be sufficiently large to accommodate the largest possible number of mutated alleles/states for a sample size N, which can be as high as $N - 1$. Remember that non-polymorphic SNPs are discarded prior to an analysis. Therefore, alignment sites with 0 or N set bits do not exist in the data to be processed.

In the final COR stage, the number of COR units should not exceed $S \times S$, which is the number of pairwise combinations for the S states per SNP. Instantiating a smaller number does not affect performance due to the expectedly large sample size, which shifts the computational bottleneck to the enumeration of alleles and haplotypes, i.e., the first three stages of the accelerator pipeline. Our accelerator pipeline exhibits 4 correlation units. If the number of samples is small enough to permit the previous AHF stage to provide input to the COR stage periodically with a period of 4 cycles or less, then the COR stage becomes the bottleneck. This is the case if less than 448 sequences (binary vector length) are processed, which, however, is not the average use-case for which the current architecture is designed. Note that, the proposed architecture accommodates such sample sizes correctly, albeit suboptimally, by deploying zero padding to ensure that a minimum of 448 bits per binary vector are allocated and processed at all times.

TABLE I. OCCUPIED RESOURCES ON A VIRTEX 6 LX760 FPGA BY A SINGLE ACCELERATOR INSTANCE, AS WELL AS BY THE COMPLETE AND FULLY FUNCTIONAL SYSTEM THAT INCLUDES VENDOR-SPECIFIC I/O INFRASTRUCTURE, E.G., MEMORY CONTROLLERS.

	Accel. instance		Full system	
Resources	Amount	Occup.	Amount	Occup.
Occup. slices	11,778	9%	43,969	37%
Slice registers	42,453	4%	150,932	15%
Slice LUTs	36,717	7%	124,909	26%
BRAMs (18Kb)	175	12%	437	30%
DSPs	96	11%	96	11%

V. IMPLEMENTATION AND VERIFICATION

The proposed architecture is described using VHDL, while the floating-point operators are generated by the Xilinx Core Generator (*https://www.xilinx.com/products/design-tools/coregen.html*). In addition to exhaustive simulation tests, correctness was verified in hardware, initially employing one of the available Virtex 6 LX760 FPGAs on a Convey HC-2ex platform. Due to the memory-bound nature of LD computations, the specific platform was selected in order to exploit its high-end memory subsystem that is capable of providing up to 80GB/s of memory bandwidth when all four available FPGAs are deployed. For performance purposes, the accelerator architecture is mapped to two and all four FPGAs of the platform, with each FPGA assigned a different memory space of SNP data to process (see Section VI for more details). Correctness was additionally verified for the high-performance configurations of the platform, i.e., when two and four FPGA devices are deployed. Based on a total of 15 memory controllers that the accelerator architecture deploys for I/O purposes when mapped to the specific platform, the number M of MSC instances in the evaluated design point is set to 7. Table I provides post-place-and-route performance and resource utilization results on a single device.

As can be observed in the table, only a fraction of the available device resources are occupied on each device, which is mainly due to our design goal of supporting arbitrarily large sample sizes. This does not allow performance improvements to follow an increase of hardware resources, e.g., via the instantiation of additional population counters or accumulators operating in parallel, since the required time for the enumeration of alleles/haplotypes can not be further reduced before the memory bandwidth is improved. Setting a different design target, such as optimizing for a smaller sample size and a larger number of SNPs [20], allows to improve performance by occupying additional resources, since entire SNPs are loaded to on-chip memory prior to processing. Aiming, in this work, at supporting large sample sizes (arbitrarily long binary vectors), dictates that entire SNPs can not be temporarily stored on

TABLE II. PERFORMANCE COMPARISON FOR ANALYZING 10,000 SNPS AND SAMPLE SIZES OF 100,000 AND 1,000,000 SEQUENCES BASED ON FSM.

	OmegaPlus 3.0.0 (Reference)				OmegaPlus modif. (Optimized)				HW speedup vs. Reference				HW speedup vs. Optimized			
	Exec. time		kLD/sec		Exec. time		kLD/sec		1 FPGA		4 FPGAs		1 FPGA		4 FPGAs	
Thr.	10^5	10^6	10^5	10^6	10^5	10^6	10^5	10^6	10^5	10^6	10^5	10^6	10^5	10^6	10^5	10^6
1	11,307	113,647	4.42	0.43	3,960	39,818	12.62	1.25	37.8	38.3	101.3	134.9	13.3	13.4	35.5	47.3
2	5,824	56,736	8.58	0.88	2,048	19,771	24.40	2.52	19.5	19.1	52.2	67.4	6.9	6.7	18.4	23.5
4	2,921	29,176	17.11	1.71	1,080	9,991	46.28	4.99	9.8	9.8	26.2	34.6	3.6	3.4	9.7	11.9
8	1,528	14,883	32.71	3.37	707	5,357	70.65	9.34	5.1	5.0	13.7	17.6	2.4	1.8	6.3	6.4
12	1,689	11,920	29.59	4.19	594	4,303	84.10	11.61	5.7	4.0	15.1	14.2	2.0	1.5	5.3	5.1

on-chip memory prior to processing, since this would incur an upper bound on the number of sequences. Based on the amount of available on-chip memory on the employed Virtex 6 devices, we calculate that this upper bound would be as low as 128,256 sequences for just 32 SNPs, likely to be exceeded in real-world analyses in the very near future [3].

Support for whole genomes, i.e., typically thousands to millions of SNPs, is provided at the software level, relying on the so-called generic algorithm [20] for LD calculations. The underlying idea is to organize SNPs in fixed-size chunks, e.g., 32 or 64 SNPs, and combine such chunks in a pairwise manner (compute groups) guided by biological restrictions. Unlike a previous work that introduced and employed the generic algorithm to load several SNP chunks entirely to FPGA on-chip memory [20], we deploy the generic algorithm to direct LD acceleration to different genomic regions along a genome, operating each time on single genomic regions of significantly larger chunk size, e.g., 10,000 SNPs.

VI. PERFORMANCE EVALUATION

To evaluate performance of the proposed accelerator architecture when mapped to one, two, and four FPGAs, we deploy a workstation with two Intel Xeon E5-2620 6-core processors running at 2.00 GHz and 24 GB of main memory, as a test platform for the software benchmarks. We provide comparisons with two state-of-the-art software implementations, PLINK 1.9 [16] and OmegaPlus 3.0.0 [17], and report execution times in seconds and throughput performance in terms of number of scores per second (LDs/sec). PLINK 1.9 serves as the reference software for the ISM benchmarks since it outperforms the respective OmegaPlus implementation when more than four cores are employed [20]. Note however that, PLINK 1.9 computes LD based on genotypes and not alleles, thus exhibiting increased complexity in comparison with the allele-based OmegaPlus implementation for LD under FSM. For this reason, the OmegaPlus implementation is employed as reference for the FSM benchmarks. In the interest of a fair performance evaluation, we modified the source code to deploy the intrinsic population counter implemented in hardware, as previously suggested [20], achieving between 2.16 and 2.85 times faster execution (see Table II).

Table II provides evaluation results for the FSM model, based on simulated datasets that comprise 100,000 and 1,000,000 DNA sequences. Both the hardware accelerator and the performance-enhanced software implementation that deploys the $\_mm\_popcnt\_u64$ intrinsic instruction (denoted "Optimized" in the table) generate qualitatively identical results with the standard OmegaPlus 3.0.0 release. Table III provides ISM comparisons with PLINK 1.9, as well as with the FPGA-based LD accelerator described by Alachiotis and Weisz [20], for a sample size of 100,000 sequences. To

TABLE III. PERFORMANCE COMPARISON FOR ANALYZING 10,000 SNPS AND 100,000 SEQUENCES BASED ON ISM.

	PLINK 1.9		LD Accel. [20]	HW speedup	
Thr.	Exec. Time	kLD/sec	1 FPGA	1 FPGA	4 FPGAs
1	389.1	128	159.3	20.9	56.0
2	297.6	168	121.4	15.9	42.7
4	180.2	277	73.6	9.7	25.9
8	109.4	456	44.7	5.9	15.7
12	88.3	566	36.0	4.7	12.7

ensure a fair comparison with the ISM implementations, we include the evaluation results (execution times and throughput performance) reported by the authors (Table 3 in [20]). Note that, software execution times in Table III are measured on a Intel Xeon E5-2630 6-core processor running at 2.60 GHz, with 32 GB of main memory, while the LD accelerator [20] is mapped to a Virtex 7 VX980T FPGA. In addition to the fact that a larger FPGA device is employed by the authors, the evaluated design point is the outcome of extensive design space exploration with a distinctively different goal and assumptions, which is to maximize throughput performance for moderate sample sizes while assuming complete binary data.

In our work, the scope is to support arbitrarily large sample sizes under both the ISM and the FSM models while considering missing data and alignment gaps to allow potential deployment in future real-world large-scale analyses. To this end, we report performance gains from the deployment of the multi-FPGA accelerator system in a real-world analysis for the detection of positive selection on the 22nd human chromosome. The dataset, available by the 1,000 Genomes project [2], comprises 2,504 sequences and 1,055,736 SNPs. OmegaPlus required 36.9 minutes (20.3 minutes for data preparation, performed by a single CPU core, and 16.6 minutes for LD computations, using 12 CPU cores) to complete the analysis on the test platform, achieving a throughput performance of 1.78 mLD/sec for the processing stage (12 CPU cores). Excluding the preparatory sequential execution, during which data parsing, filtering, and encoding are performed (20.3 minutes, required by both the software and the hardware implementations), the multi-FPGA acceleration system achieves 6 times faster processing than 12 CPU cores, delivering a throughput performance of 10.64 mLD/sec with four FPGAs.

Figure 5 provides speedups obtained by 1, 2, and 4 FPGAs, for an increasing number of samples up to 1,000,000. Performance fluctuations are observed for small sample sizes, due to the fact that processing times are comparable to the setup/synchronization overhead. As the sample size increases, the computation-to-synchronization ratio improves, leading to steady performance gains over both the standard and the optimized (with $\_mm\_popcnt\_u64$) software implementations. The process of assigning compute chunks to multiple FPGAs relies on the computational load distribution mechanism of the generic algorithm [20], which is agnostic to the underlying

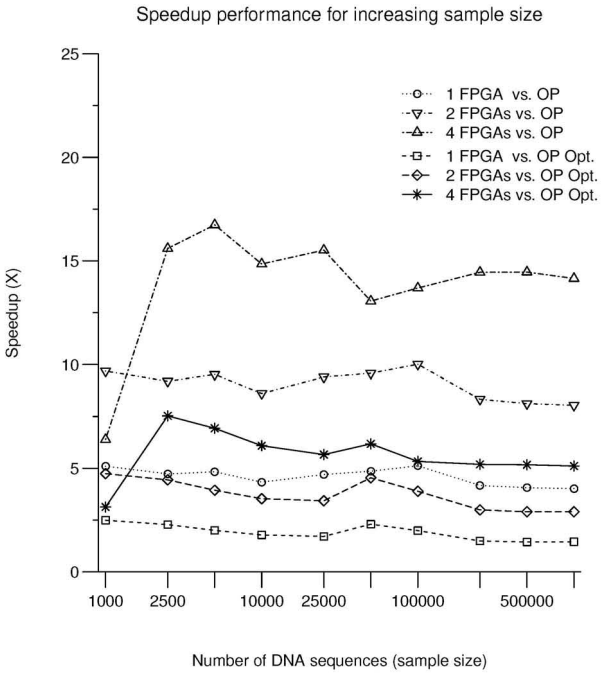

Fig. 5. Speedup performance for 1,000 to 1,000,000 sequences (OP: OmegaPlus 3.0.0, OP Opt: optimized with intrinsic population counter).

hardware, practically allowing to deploy an arbitrary number of devices by treating each device similarly to a processor core. This explains the poor speedup performance that is observed when 4 FPGAs are deployed for the analysis of small sample sizes. For 1,000 sequences, for instance, a second FPGA improves performance (4.7X with two FPGAs vs. 2.5X with one), while additional devices obliterate the attained speedup gain due to an increased synchronization overhead (3.1X with four FPGAs vs. 4.7X with two). When 1,000,000 sequences are analyzed, on the other hand, a favorable computation-to-synchronization ratio is maintained despite the additional devices used due to excessive compute requirements. Therefore, performance steadily improves with an increasing number of devices, allowing four FPGAs to deliver up to 3.5X faster processing than a single device.

VII. CONCLUSIONS

Motivated by the current trend of increasing sample sizes in genomic studies, we designed an accelerator architecture for LD to eliminate the upper limit of previous hardware solutions on the number of sequences. The accelerator architecture exhibits increased population count capacity while meeting real-world biological requirements, that, in combination with an appropriately modified parallel algorithm for deploying multiple FPGAs, yield the proposed solution highly suitable for large-scale analyses of real data.

Evaluations of computational methods that employ multiple statistics to analyze SNPs have revealed that dataset-specific features, such as the SNP location or the variance of SNP densities in subgenomic regions, can shift the performance bottleneck among the employed kernels/statistics. As future work, we therefore intend to devise a self-adaptable acceleration system that relies on partial reconfiguration at runtime in order to maximize performance on an as-needed basis during distinct computational stages in long-running biological analyses.

ACKNOWLEDGEMENT

This work was supported in part by EU H2020 ICT project dRedBox, contract #687632, and by EU H2020 FETHPC project EXTRA, contract #671653.

REFERENCES

[1] Genomics England, "The 100,000 genomes project," 2015. [Online]. Available: https://www.genomicsengland.co.uk/the-100000-genomes-project/

[2] P. H. Sudmant *et al.*, "An integrated map of structural variation in 2,504 human genomes," *Nature*, vol. 526, no. 7571, pp. 75–81, 2015.

[3] Z. D. Stephens *et al.*, "Big data: astronomical or genomical?" *PLoS Biol*, vol. 13, no. 7, p. e1002195, 2015.

[4] R. Lewontin and K. Kojima, "The evolutionary dynamics of complex polymorphisms," *Evolution*, pp. 458–472, 1960.

[5] D. E. Reich *et al.*, "Linkage disequilibrium in the human genome," *Nature*, vol. 411, no. 6834, pp. 199–204, 2001.

[6] J. L. Crisci *et al.*, "The impact of equilibrium assumptions on tests of selection," *Frontiers in genetics*, vol. 4, 2013.

[7] M. T. Alam *et al.*, "Selective sweeps and genetic lineages of Plasmodium falciparum drug -resistant alleles in Ghana." *The Journal of infectious diseases*, vol. 203, no. 2, pp. 220–7, Jan. 2011.

[8] P. M. Visscher *et al.*, "Five years of GWAS discovery," *The American Journal of Human Genetics*, vol. 90, no. 1, pp. 7–24, 2012.

[9] N. Alachiotis *et al.*, "Efficient computation of linkage disequilibria as dense linear algebra operations," *High Performance Computational Biology (HICOMB)*, 2016.

[10] M. Kimura, "The number of heterozygous nucleotide sites maintained in a finite population due to steady flux of mutations," *Genetics*, vol. 61, no. 4, p. 893, 1969.

[11] J. Felsenstein, "Evolutionary trees from dna sequences: a maximum likelihood approach," *Journal of molecular evolution*, vol. 17, no. 6, pp. 368–376, 1981.

[12] L. A. Mathew *et al.*, "Why to account for finite sites in population genetic studies and how to do this with jaatha 2.0," *Ecology and evolution*, vol. 3, no. 11, pp. 3647–3662, 2013.

[13] D. R. Kelley *et al.*, "Quake: quality-aware detection and correction of sequencing errors," *Genome biology*, vol. 11, no. 11, p. 1, 2010.

[14] B. Pfeifer *et al.*, "PopGenome: An Efficient Swiss Army Knife for Population Genomic Analyses in R." *Molecular biology and evolution*, vol. 31, no. 7, pp. 1929–36, Jul. 2014.

[15] S. Purcell *et al.*, "PLINK: a tool set for whole-genome association and population-based linkage analyses," *The American Journal of Human Genetics*, vol. 81, no. 3, pp. 559–575, 2007.

[16] C. C. Chang *et al.*, "Second-generation PLINK: rising to the challenge of larger and richer datasets," *Gigascience*, vol. 4, no. 1, p. 1, 2015.

[17] N. Alachiotis *et al.*, "OmegaPlus: a scalable tool for rapid detection of selective sweeps in whole-genome datasets," *Bioinformatics*, vol. 28, no. 17, pp. 2274–2275, 2012.

[18] I. T. Li *et al.*, "160-fold acceleration of the Smith-Waterman algorithm using a field programmable gate array (FPGA)," *BMC bioinformatics*, vol. 8, no. 1, p. 1, 2007.

[19] S. Zierke and J. D. Bakos, "FPGA acceleration of the phylogenetic likelihood function for Bayesian MCMC inference methods," *BMC bioinformatics*, vol. 11, no. 1, p. 184, 2010.

[20] N. Alachiotis and G. Weisz, "High Performance Linkage Disequilibrium: FPGAs Hold the Key," in *Proceedings of the 2016 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays*, ser. FPGA '16. ACM, 2016, pp. 118–127.

[21] P. Pavlidis *et al.*, "SweeD: likelihood-based detection of selective sweeps in thousands of genomes," *Molecular biology and evolution*, p. mst112, 2013.

[22] E. P. Hong and J. W. Park, "Sample size and statistical power calculation in genetic association studies," *Genomics & informatics*, vol. 10, no. 2, pp. 117–122, 2012.