Mapping Multiple Multivariate Gaussian Random Number Generators on an FPGA

Chalermpol Saiprasert, Christos-Savvas Bouganis and George A. Constantinides

Department of Electrical and Electronic Engineering
Imperial College London
Exhibition Road, South Kensington, London, SW7 2AZ, UK
Email: cs405@imperial.ac.uk, ccb98@imperial.ac.uk, gac1@imperial.ac.uk

***Abstract*—A Multivariate Gaussian random number generator (MVGRNG) is an essential block for many hardware designs, including Monte Carlo simulations. These simulations are usually used in applications such as statistical physics and financial mathematics. Field Programmable Gate Arrays (FPGAs) are often used to implement these generators as the design can be effectively optimized. Many applications require random samples from a number of multivariate Gaussian distributions leading to a problem of efficiently mapping of the required MVGRNG on an FPGA. The proposed approach presented in this paper exploits any redundancy that exists between different distributions under consideration leading to designs with improved resource usage. Experimental results demonstrate that the proposed approach outperforms the existing approaches by producing MVGRNG designs that utilize less hardware resources in comparison to existing approaches achieving up to 50% reduction of hardware resource utilization.**

I. Introduction

The behaviour of many stochastic processes can be simulated using Monte Carlo methods in which the element of randomness originates from a set of probability distributions. In many applications such as statistical physics and financial mathematics, such stochastic processes are dependent on many correlated factors which are often captured by the multivariate Gaussian distribution [1] [2]. In order to estimate a function of the stochastic process with a certain level of confidence, a large number of simulations is often required making the entire process very time consuming. Field Programmable Gate Arrays (FPGA) are an ideal candidate to accelerate the multivariate Gaussian random number generator block (MVGRNG) as they offer fine grain parallelism and customizable computational path. Examples of existing works in the acceleration of financial applications and random number generators on FPGAs can be found in [3], [4], [5] and [6].

Existing approaches for the implementation of the FPGA-based MVGRNGs have only focused on the design and optimization of a single multivariate Gaussian random distribution [5],[6],[7]. However, since the MVGRNG block is usually a part of a larger application which is mapped on an FPGA, its area requirement can be crucial as more of the hardware resource on the FPGA can be allocated for the main application. The proposed approach adds an extra dimension to the above optimization problem by considering the mapping of multiple multivariate Gaussian distributions. Many applications which are driven by the MVGRNG do not require very high throughput of random numbers and so the throughput of the design can be traded off for an improved hardware resource usage. Hence, random numbers from many multivariate Gaussian distributions can be sampled from a single hardware block which is optimized for all distributions of interest. In more details, the approach presented in this paper exploits similarities in the precision requirement that exist between different distributions under consideration by exploring the similarities in the correlation matrices which characterize the target distributions. This subsequently creates an opportunity for an effective resource sharing by trading off the throughput of the random number generator for an improved resource usage.

This paper is organized as follows. The background theory is explained in Section II, while a description of current related work is given in Section III. Section IV focuses on the detailed description of the hardware architecture and the proposed algorithm is discussed in Section V. Experimental results are presented in Sections VI and VII. Finally, Section VIII concludes the paper.

II. Background: Generating Multivariate Gaussian Random Numbers

A multivariate Gaussian distribution is characterized by its mean vector $\mathbf{m}$ and correlation matrix $\mathbf{\Sigma}$. Several methods to sample from multivariate Gaussian distribution exist in the literature. As far as the hardware implementation of the MVGRNG on an FPGA is concerned, some of the most widely used techniques are Cholesky factorization and Eigenvalue decomposition. Using Cholesky factorization method, $\mathbf{\Sigma}$ is decomposed using Cholesky decomposition into a product of a lower triangular matrix and its transpose [8]. The required samples are generated through a linear combination of univariate Gaussian samples that follow a standard Gaussian distribution $N(0,1)$. This method reduces the number of computations by half due to the lower triangular property of the matrix in comparison to full matrix-vector multiplication.

An alternative method is to decompose $\mathbf{\Sigma}$ by Eigenvalue decomposition [9]. As a result of the decomposition, $\mathbf{\Sigma}$ can be expressed as a linear combination of three separable matrices $\mathbf{U\Lambda U}^T$ where $\mathbf{U}$ is an orthogonal matrix ($\mathbf{UU}^T = \mathbf{I}$) containing eigenvectors $\mathbf{u}_1, ..., \mathbf{u}_N$ while $\mathbf{\Lambda}$ is a diagonal matrix with diagonal elements being the eigenvalues $\lambda_1, ..., \lambda_N$, where N denotes the rank of $\mathbf{\Sigma}$. If we let $\mathbf{A} = \mathbf{U}\mathbf{\Lambda}^{1/2}$, multivariate Gaussian random samples that follow $N(\mathbf{m}, \mathbf{\Sigma})$ can be generated as in (1), where $\mathbf{z} \sim N(\mathbf{0}, \mathbf{I})$.

$$\begin{aligned} \mathbf{x} &= \mathbf{A}\mathbf{z} + \mathbf{m} = \mathbf{U}\mathbf{\Lambda}^{1/2}\mathbf{z} + \mathbf{m} \\ &= (\sqrt{\lambda_1}\mathbf{u}_1 z_1 + \sqrt{\lambda_2}\mathbf{u}_2 z_2 + ... + \sqrt{\lambda_K}\mathbf{u}_K z_K) + \mathbf{m} \\ &= \sum_{i=1}^{K}(\sqrt{\lambda_i}\mathbf{u}_i z_i) + \mathbf{m}, \qquad (1) \end{aligned}$$

where $\mathbf{x}$ is a vector of multivariate Gaussian random samples. The original correlation matrix $\mathbf{\Sigma}$ can be fully recovered when the number of decomposition levels K is equal to the rank of the correlation matrix. Therefore, the correlation matrix of the original distribution can be approximated by taking into account K levels of decomposition, where $K \leq rank(\mathbf{\Sigma})$.

It is important to note that Cholesky decomposition method can only be applied to positive semi-definite matrices while the Eigenvalue decomposition is able to handle any given correlation matrix. In addition, the Eigenvalue decomposition allows the approximation of correlation matrix by using a certain number of decomposition levels. This translates to many possible designs for a target correlation matrix with different hardware resource requirement and approximation errors. Hence, this will be the method used in this work for the generation of multivariate Gaussian samples.

III. Related Work

In the literature, the earliest approach to implement a multivariate Gaussian random number generator on an FPGA was presented in [5] where the Cholesky decomposition technique was applied to the input correlation matrix in order to take advantage of the lower triangular property of the resulting matrix. If N denotes the dimensionality of the distribution, the authors use a set of N DSP48 blocks to implement the multiply-add operation producing a vector of independent multivariate Gaussian random samples over N clock cycles. A drawback from their approach is that the number of required DSP blocks is dictated by the dimension N of the distribution to be sampled from.

The resource restriction in [5] has been addressed by an approach in [6] where an alternative method based on the use of Singular Value Decomposition was introduced. In [6], the correlation matrix is approximated by trading off the error in the approximation of the input correlation matrix for an improved hardware resource usage. Similar to [5], the architecture is implemented using DSP48 blocks. However, $2K$ DSP48 blocks are required to generate a vector of size N, where K denotes the number of decomposition levels required to approximate the lower triangular matrix while maintaining the same throughput as in [5]. The approach in [6] provides the flexibility to accommodate any given resource constraints as well as optimizing for hardware resource usage. However, the design methodology in [6] is not fully optimized as it has been demonstrated that allocating a fixed precision to all of the computation paths of the architecture does not lead to the optimum resource utilization.

A mixed precision approach based on word-length optimization techniques has been proposed in [7] to handle the issues encountered in [6]. It has been demonstrated in [7] that the hardware resource utilization can be further reduced by exploring different precision requirements in the datapaths of the architecture in comparison to other approaches. Moreover, the impact of the correlation errors due to truncation operations has been investigated and accounted for in an optimized framework, leading to designs with optimum hardware resource usage.

To the best of the authors' knowledge, none of the existing approaches in the literature has discussed the optimization of hardware architecture when multiple multivariate Gaussian distributions are targeted. The approach presented in this paper provides a methodology for the efficient mapping of the multiple distributions MVGRNG block in an FPGA by exploiting any redundancy in the different input correlation matrices leading to a single generator which is optimized across all distributions of interest targeting an improved hardware resource usage.

IV. Proposed Algorithm

The proposed algorithm is based on the MVGRNG architecture proposed in [7], where word-length optimization techniques to produce generators with mixed precisions in the architecture are utilized. The hardware architecture in [7] is based on the Eigenvalue decomposition algorithm, which decomposes an input correlation matrix into a set of vectors. Then, an optimization process that selects the appropriate precision for each decomposition level is performed in order to achieve a good approximation to the input correlation matrix minimizing at the same time the resource utilization.

The main idea behind this work is to reuse each decomposition block calculation, which is the most resource demanding part of the design, across many MVGRNGs. In order for this to be achievable, an optimum selection of the precision to be used in the blocks should be performed such that the chosen precision should minimize the approximation error across all correlation matrices under consideration leading to optimum designs for all distributions of interest.

Fig.1 gives an overview of the proposed algorithm. The main input to the proposed algorithm are the correlation matrices which characterize the target distributions, hardware blocks specification, choice of objective function and the termination condition. Firstly, all M correlation matrices are individually decomposed using the eigenvalue decomposition algorithm resulting in vectors of coefficients $\mathbf{c}$. These coefficients are quantized into fixed point representation using the user-specified word-lengths from a hardware library. As a result, many possible candidate designs are generated for each input correlation matrix exploring the approximation error-hardware resource usage design space. In order to assess how good a design is, an error between the target correlation matrix and the approximated matrix obtained after the decomposition and quantization step is calculated. Note that the flexibility of the proposed algorithm allows for any error metric to be used as an objective function for the optimization. In this work, the mean square error is used as the objective function (2).

$$MSE_K^i = \frac{1}{N_i^2} \left\| \mathbf{R}_{K-1}^i - (\mathbf{C}_K^i + \mathbf{W}^i) \right\|_F^2, \qquad (2)$$

where K denotes the decomposition level and i denotes the i^{th} distribution of interest, $i \in [1, ..., M]$. $\mathbf{R}_{K-1}^i$ denotes the remaining of the original correlation matrix after $K-1$ levels of decomposition with $\mathbf{R}_0^i$ defined as the original correlation matrix at the start of the algorithm. The $\|.\|_F$ operator denotes the Frobenius norm while N is the order of the matrix. The approximated correlation matrix for each candidate design $\overline{\mathbf{\Sigma}}^i$ can be expressed as $\overline{\mathbf{\Sigma}}^i = \mathbf{C}_K^i + \mathbf{W}^i$, where $\mathbf{C}_K^i$, approximates the input correlation using K levels of decomposition taking into consideration the quantization effects and $\mathbf{W}^i$ is an expected truncation error matrix, which is a function of the correlation of the introduced errors due to truncation operations [7].

After obtaining individual approximation errors, the algorithm estimates the overall approximation error of all M input matrices by adding all individual errors together. In the next stage of the algorithm, inferior designs are discarded. An inferior design is defined as one that uses more hardware resources but produces worse approximation error in comparison to another design. At this point, the algorithm only stores the designs which are optimum for all of the input distributions by exploiting any redundancy that exists between the M correlation matrices. It is important to note that all input matrices share the same precision requirements for the datapath of the computational block at each level of decomposition while having distinct coefficients corresponding to their own distribution. The last step of this stage calculates the remainder of the original matrices which will be used as the starting point for the next iteration.

The algorithm repeats until the termination condition is met. Examples of the termination condition are a specified approximation error, a given resource constraint or the number of decomposition levels to be used. The output of the proposed algorithm is a set of vectors of coefficients which best approximate the input M correlation matrices under a given objective function. Therefore, the proposed algorithm constructs a custom MVGRNG from computational blocks with different precision requirements targeting the minimization of approximation error and resource usage of each of the input correlation matrices, resulting in optimized designs for all target distributions. The expression in (3) illustrates how a vector of three multivariate Gaussian samples is calculated for each correlation matrix.

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} c_1^1 \\ c_2^1 \\ c_3^1 \end{bmatrix} z_1 + \begin{bmatrix} c_1^2 \\ c_2^2 \\ c_3^2 \end{bmatrix} z_2 + ... + \begin{bmatrix} c_1^K \\ c_2^K \\ c_3^K \end{bmatrix} z_K, \quad (3)$$

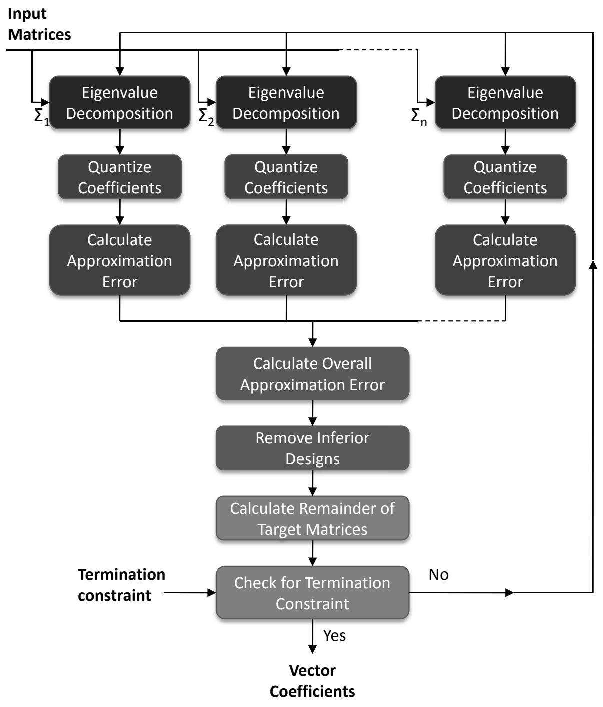
Input Matrices
Σ1
Σ2
Σn
Eigenvalue Decomposition
Quantize Coefficients
Calculate Approximation Error
Calculate Overall Approximation Error
Remove Inferior Designs
Calculate Remainder of Target Matrices
Termination constraint
Check for Termination Constraint
No
Yes
Vector Coefficients

Fig. 1. Proposed Algorithm.

where c_1^1 refers to the first coefficient in vector $\mathbf{c}^i$ in the first decomposition level. Note that different set of univariate Gaussian random samples are used to drive each generator, in order to avoid introducing any unwanted correlation between the distributions of interest.

In summary, a methodology to construct a hardware architecture for multiple multivariate Gaussian distributions under a shared generator has been described. The proposed algorithm targets any redundancy between the input distributions in order to select the most effective coefficients and the corresponding precisions for each computational block under a given objective function to minimize the approximation error across all target distributions. It is important to note that no correlation between the input correlation matrices is introduced.

V. FPGA Hardware Architecture

In this section, the general hardware architecture for a multiple multivariate Gaussian random number generator is described. The proposed hardware architecture is shown in Fig.2. The generator consists of K building blocks called computational block (CB) where K denotes the number of decomposition levels used in the approximation of the correlation matrices under consideration. Each block contains an adder and a multiplier with its own precision requirements for its datapath. The architecture is mapped to logic elements only, so that the precision of the datapath can be finely tuned. The architecture can be constructed using any number of CBs accommodating any given resource constraints. The precision in the datapath of each CB is optimized for all vectors of coefficients $\mathbf{c}_j^i$, where j denotes the decomposition level and

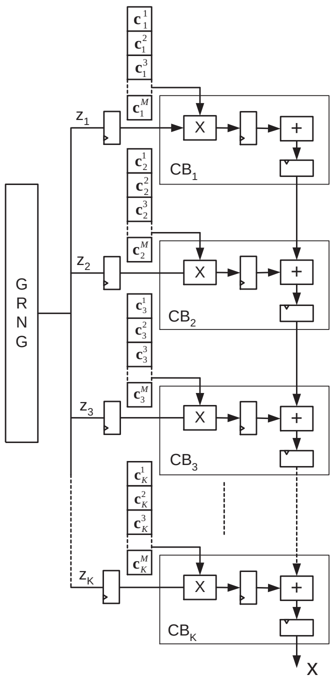

Fig. 2. Hardware Architecture.

i denotes the number of correlation matrix. The precision in the adder path is fixed to the maximum of the precisions of all CBs. Independent univariate Gaussian samples z are produced from the GRNG block using different seeds in order to avoid unwanted correlation between the resulting vectors.

In order for a high throughput to be achieved, the multiply-add operation is pipelined and all the computational blocks operate in parallel. Moreover, fixed point number representation is deployed throughout the entire design as it produces designs which consume fewer resources and operate at a higher frequency in comparison to designs that use floating point arithmetic. The elements of each multivariate Gaussian random vector are serially generated so that one complete vector is produced per $\sum_{i=1}^{M} N_i$ clock cycles, where N_i denotes the dimensionality of the i^{th} multivariate Gaussian distributions and M is the number of distributions of interest.

VI. Experimental Setup

In order for a fair comparison to be achieved between the proposed approach and the existing approaches, the throughput of all the generators under consideration must be adjusted to be at the same level. In this work the throughput of all generators under investigation should match the value of the proposed approach which is one complete vector per $\sum_{i=1}^{M} N_i$ clock cycles.

A. Extension of the Existing Approaches

The architecture of the approach in [5] does not require modification to match the desired throughput as no optimization across all input correlation matrices can be performed due to the fixed precision in the datapath of the DSP blocks. Thus, a single generator consisting of N DSP blocks will be used for the generation of samples from M multivariate Gaussian distributions.

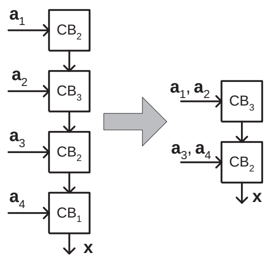

Fig. 3. Extension of Previous Approach [7].

Since the algorithm in our previous work [7] only optimizes for one correlation matrix at a time, M generators are required to sample from M distributions. However, all generators constructed using [7] will have a throughput of $\frac{1}{N}$. Hence, a modification to [7] is carried out in order to match the desired throughput by reducing the hardware resources. This is reflected in the selection of the computational blocks where the vector of coefficients of M consecutive decomposition levels are forced to share the same computational block. However, the optimization of each CB takes into account M consecutive decomposition levels. Fig. 3 illustrates an example of such modification where a 2x2 matrix $\mathbf{A}$ is considered. The first two consecutive decomposition levels in the approximation of $\mathbf{A}$ are shared by using one CB, in which the optimization is performed within the matrix between the first two levels. The same procedure applies to the next two consecutive decomposition levels. In summary, all of the generators under consideration have been modified in order to have the same throughput.

VII. Performance Evaluation

In this section, the performance of the proposed approach is assessed in comparison to the existing approaches. The designs generated from the proposed methodology, as well as the extension of the approach in [7], have been implemented on a Stratix III EP3SE50F484C2 FPGA from Altera and Quartus II was utilized as a hardware synthesis tool. The word-length of the univariate Gaussian samples $\mathbf{z}$ is fixed to 18 bits precision throughout the entire design, where 3 bits are allocated for the integer part and 14 bits for the fractional part.

In this experiment, four sets of input correlation matrices are considered. Each of the first three sets contains four matrices with identical matrix order which are 2x2, 4x4 and 6x6 matrices respectively. The last set of input correlation matrices is a mixture of matrices with different orders, namely two 2x2 and two 4x4 matrices. The empirical approximation error in the sample correlation matrix using 100,000 vectors of multivariate Gaussian random samples is obtained for each of the four sets.

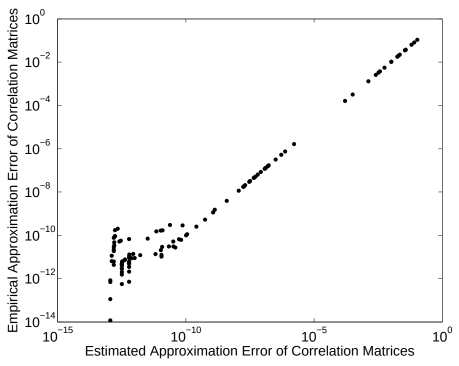

Fig. 4. Estimate vs Empirical Approximation Error.

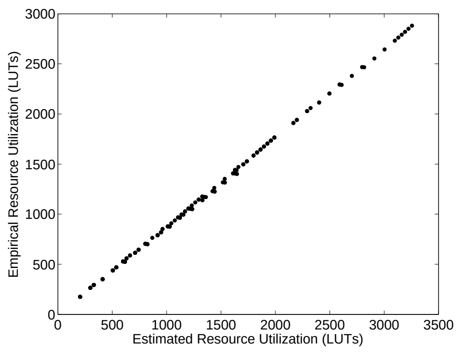

Fig. 5. Estimate vs Empirical Resource Utilization.

A. Analysis of Approximation Error and Resource Estimation Models

This section focuses on the comparison between the estimated approximation error of each of the generated MVGRNG from the proposed approach with the empirical approximation error from the sample correlation matrix. This will enable us to assess how well the error estimation model in the proposed algorithm performs. Fig. 4 shows a plot of estimate versus empirical approximation error of the MVGRNGs designed using the proposed approach. It can be clearly seen that the plot is a straight line with a gradient of 1. The more scattered points in the lower left hand corner are considered insignificant as these errors are in the range of 10^{-10} to 10^{-15}. Thus, the error calculation in the proposed approach is accurately modeled as it provides almost a perfect match between estimate and empirical approximation errors. This signifies that the error model used to drive the optimization in the proposed approach is valid.

Since the MVGRNGs are constructed from a combination of CBs, the proposed approach estimates the total resource usage of the design based on the pre-calculated usage of each CB. The plot in Fig. 5 shows that the estimate resource usage increases linearly with the empirical resource usage. The plot is fitted with a straight line with equation $y = 1.13x + 16$, where x and y denote the empirical and estimate resource usage respectively. Hence, a more accurate estimation of resource usage is fitted back to the proposed algorithm. The reason for this mismatch is due to the extra optimization steps performed during the technology mapping stage which are not modeled in the proposed approach. However, a method to provide feedback to the proposed algorithm on these optimization is provided.

In terms of the throughput of the MVGRNGs produced by the proposed approach, the maximum operating frequencies range from 395 MHz to 512 MHz with a mean of 457 MHz, achieving similar throughput to [5] and [7].

B. Analysis of Empirical Data

In this section, the performance of the MVGRNGs designed from the three approaches under investigation, which are the proposed approach, the extension of [7] and the approach in [5], are examined. The graphs in Fig. 6, 7 and 8 illustrate the empirical approximation error of the target correlation matrices using the first three sets of input correlation matrices. These are matrices with orders of 2, 4 and 6 respectively. Each point in the plots represents a design produced from a corresponding approach using a certain number of LUTs. It is important to note that the proposed approach and the approach in [7] have the capability to produce designs across a range of resource constraints for a given set of correlation matrices while only one fixed design is possible using [5]. The largest reduction in resource usage is observed in Fig. 8 where for the same approximation error of approximately 1.2×10^{-7} the proposed approach provides a 50% improvement on resource usage over the extension of [7]. The same comparison is made between the proposed approach and [5] where it was found that the proposed approach offers up to 38% improvement on resource usage.

The largest reduction in correlation matrix approximation error is observed in Fig. 7. For similar resource utilization of approximately 1250 LUTs, the proposed approach offers 7 orders of magnitude of improvement on the approximation error over the extension of [7] while 5 orders of magnitude of improvement is achieved over [5]. It can be concluded from the plots that for all matrix orders, the proposed approach provides the best MVGRNGs in comparison to [5] and the extension of [7] as these generators achieve lower approximation error while utilizing the same hardware resources. This was achieved by taking into account precision similarities that exist between the input correlation matrices in a given set and effectively map them in the proposed algorithm. Thus, the overall performance of the MVGRNG is further improved in comparison to existing approaches. A similar trend is observed for a set of input correlation matrices with different matrix orders. The results of this test are depicted in Fig. 9. It can be seen that the proposed approach produces designs that achieve better approximation error for the same resource usage in comparison to existing approaches.

In order to assess the impact of the finite number of univariate Gaussian samples used to drive the described generators, to their performance, two "virtual" designs are considered. These designs are described as "virtual", as only their em-

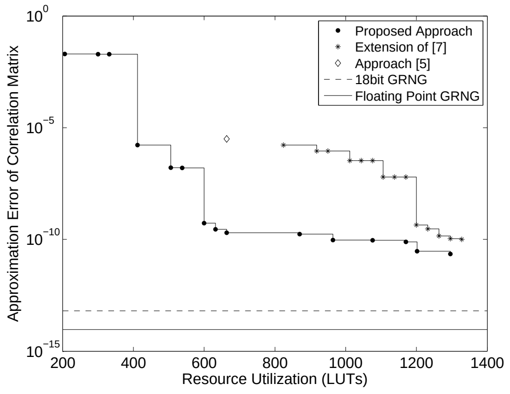

Fig. 6. Comparison of Four 2x2 Correlation Matrices.

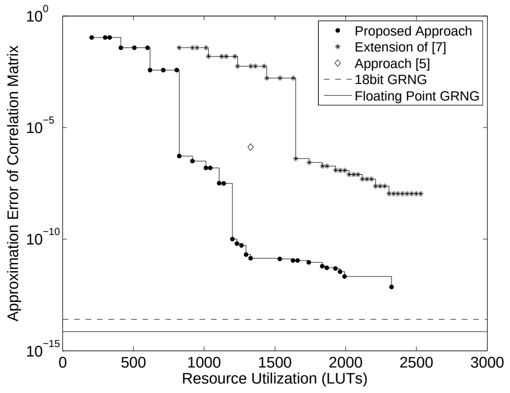

Fig. 7. Comparison of Four 4x4 Correlation Matrices.

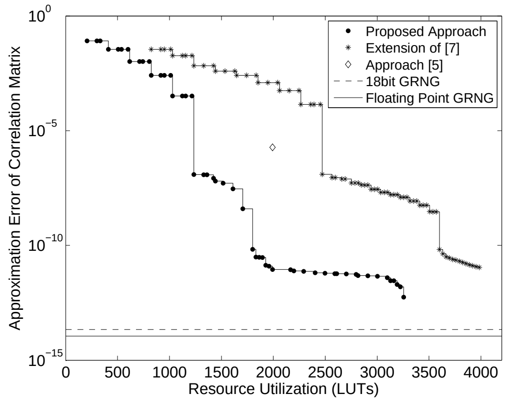

Fig. 8. Comparison of Four 6x6 Correlation Matrices.

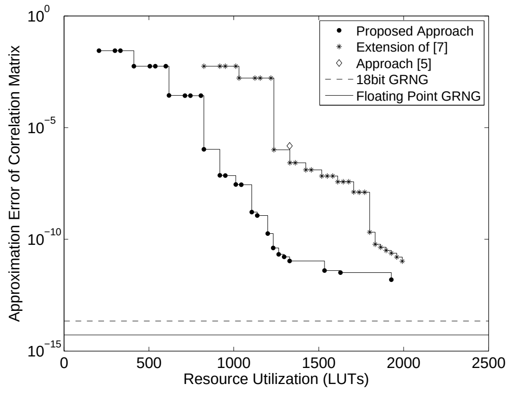

Fig. 9. Comparison of Correlation Matrices with orders of 2 and 4.

pirical approximation error is considered. The first "virtual" design uses the same quantized univariate Gaussian samples as the mentioned designs, but their mapping to sample from multivariate Gaussian distributions is performed by employing the lower triangular matrix using Cholesky decomposition and double precision floating point arithmetic (MATLAB). The achieved approximation error is denoted by the dotted lines on the plots in Fig. 6 ,7, 8 and 9. The second "virtual" design, is similar to the first one, but no quantization is performed to the set of univariate Gaussian samples. This case models the "best" possible performance using the specific set of univariate samples, and it is denoted by the solid lines. The results demonstrate that the approximation error achieved by the generators from the proposed approach is very close to the virtual designs.

VIII. Conclusion

In this paper, a novel approach is presented to design and implement a hardware architecture that effectively maps multiple multivariate Gaussian random number generators on an FPGA. The proposed approach exploits redundancy that exists between the multivariate Gaussian distributions leading to designs which are optimized for all target distributions while minimizing the hardware resoure utilization. Experimental results have demonstrated that the proposed approach outperforms the existing approaches achieving up to 50% reduction in hardware resource utilization without any compromise in the approximation error of the target correlation matrices.

References

[1] K. Binder and D. Heermann, *Monte Carlo Simulation in Statistical Physics. An Introduction (4th edition)*. Springer, 2002.

[2] P. Glasserman, P. Heidelberger, and P. Shahabuddin, "Variance reduction techniques for value-at-risk with heavy-tailed risk factors," in *Proceedings of the 32nd conference on Winter simulation*, 2000, pp. 604–609.

[3] G. Zhang, P. H. Leong, D.-U. Lee, J. D. Villasenor, R. C. Cheung, and W. Luk, "Ziggurat-based hardware gaussian random number generator," in *Proceedings IEEE International Symposium on Field Programmable Logic and Applications*, 2005, pp. 275–280.

[4] N. A. Woods and T. VanCourt, "FPGA acceleration of quasi-monte carlo in finance." in *Proceedings IEEE International Conference on Field Programmable Logic and Applications*, 2008, pp. 335–340.

[5] D. B. Thomas and W. Luk, "Multivariate gaussian random number generation targeting reconfigurable hardware," *ACM Transactions on Reconfigurable Technology and Systems*, vol. 1, no. 2, pp. 1–29, 2008.

[6] C. Saiprasert, C.-S. Bouganis, and G. A. Constantinides, "Multivariate gaussian random number generator targeting specific resource utilization in an FPGA," in *ARC '08: Proceedings of the 4th international workshop on Reconfigurable Computing*. Berlin, Heidelberg: Springer-Verlag, 2008, pp. 233–244.

[7] ——, "An optimized hardware architecture of a multivariate gaussian random number generator," *ACM Transactions on Reconfigurable Technology and Systems*, 2009 (to appear).

[8] P. Glasserman, *Monte Carlo Methods in Financial Engineering*. New York, USA: Springer, 2004.

[9] N. H. Chan and H. Y. Wong, *Simulation Techniques in Financial Risk Management*. New Jersey, USA: Wiley, 2006.