An Analytical Approach for Dynamic Range Estimation[*]

Bin Wu Jianwen Zhu Farid N. Najm

Department of Electrical and Computer Engineering
University of Toronto, Toronto, Ontario, Canada
{bwu, jzhu, najm}@eecg.toronto.edu

ABSTRACT

It has been widely recognized that the dynamic range information of an application can be exploited to reduce the datapath bitwidth of either processors or ASICs, and therefore the overall circuit area, delay and power consumption. While recent proposals of analytical dynamic range estimation methods have shown significant advantages over the traditional profiling-based method in terms of runtime, we argue that the rather simplistic treatment of input correlation may lead to significant error. We instead introduce a new analytical method based on a mathematical tool called Karhunen-Loéve Expansion (KLE), which enables the orthogonal decomposition of random processes. We show that when applied to linear systems, this method can not only lead to much more accurate result than previously possible, thanks to its capability to capture and propagate both spatial and temporal correlation, but also richer information than the value bounds previously produced, which enables the exploration of interesting trade-off between circuit performance and signal-to-noise ratio.

Categories and Subject Descriptors

B.7 [**Integrated Circuits**]: Design Aids

General Terms

Algorithms, Design, Theory

Keywords

Dynamic range, bitwidth, Karhunen-Loéve Expansion (KLE), correlation

1. INTRODUCTION

Today's ASIC designers start with a design specification handed off by system designers. Often in the form of C code, the algorithm-level design specification needs to be converted into register transfer level (RTL) design, typically in the form of hardware description languages. A crucial decision to be made during this process is the datapath bitwidth, including the bitwidths of different registers and functional units. An aggressively designed datapath often replaces floating-point arithmetic operations contained in the design specification by their fixed-point counterparts. In addition, the redundant bits that do not contribute much to the accuracy of the application are often eliminated. Such datapath with minimal bitwidth always translates to superior *circuit performance* in terms of area, speed and power consumption. To make this possible, the dynamic range information of the application, and in the case of C code, the dynamic range of all declared variables and intermediate expressions (all referred to as variables in the following text), has to be obtained.

Unfortunately, the best practice today for dynamic range estimation is still profiling, which works by instrumenting the original application with code that can trace the value ranges at runtime. While this method can be made very accurate, the accuracy is achieved only by extremely long simulation, and worst of all, no confidence on the accuracy can be obtained. In contrast, analytical methods can avoid long simulation by analyzing the application at compile time. While many advances have been made on this front, the proposed methods have not been able to provide dynamic range information as accurate and as complete as profiling.

More specifically, most analytical methods provide only value bounds of the variables in terms of their minimum and maximum values. The lack of other information, such as variable statistics and probability, may limit our capability to accurately estimate circuit performance. For example, in order to estimate power consumption, the signal probability and correlation are required in the circuit power macro-models. These power macro-model parameters can be obtained from the variable statistics. As another example, from the joint-distribution of two variables used for comparison in a branch statement, we can distinguish a frequently taken branch from an infrequently taken one, and therefore more accurate estimation of power consumption. In addition, for different categories of circuits, designers need to make different tradeoffs between *algorithm reliability* in terms of signal-to-noise ratio (SNR) and circuit performance. If the distribution of variable values is available, every decision on datapath bitwidth reduction can be quantitatively associated with the reduced signal-to-noise ratio. Therefore, for designs with high reliability requirement, a wider bitwidth can be chosen, while for consumer electronics and cost-driven designs where certain level of overflow is tolerable, narrower bitwidth may be chosen to strike better balance between reliability, cost and power.

In this paper, we propose a new analytical framework based on Karhunen-Loéve Expansion (KLE) and demonstrate its effectiveness in linear systems, which characterize a large class of useful applications. KLE method allows us to decompose the system input, modeled as an arbitrary random process, into K number of deterministic signals, each multiplied by a random variable. Exploiting the linearity of the system, we can therefore obtain the system response of a random input by combining system responses of K deterministic inputs.

We demonstrate the following advantages over previous approaches: First, it is extremely *fast.* While profiling has to simulate the system for thousands or even millions of random input realizations, our method only needs to simulate K inputs, where K is typically a small number. Second, it offers *complete* infor-

[*] This research was supported by Micronet, with funding from ATI Technologies Inc. and from Altera Corp.

DAC 2004, June 7–11, 2004, San Diego, California, USA.

mation. In fact, the distribution of all variables can be obtained. As a result, important design tradeoffs discussed earlier can be explored. Third, it is highly *accurate*. Our method can capture the temporal correlation in the system input, which is not possible in the previous methods, but as we show later, contributes significantly to the accuracy of the result.

The remainder of this paper is organized as follows: Section 2 gives a brief account of the related literature. Section 3 gives the theoretical foundation of our method before presenting the CAD tool framework we developed. We give experimental result in Section 4 before drawing conclusion in Section 5.

2. RELATED WORK

Profiling or simulation-based approaches [1, 2] have been used extensively to estimate the dynamic range of variables in a program. As mentioned earlier, these method are computationally expensive, since huge amount of sample data need to be simulated.

The L_p norm method based on use of a transfer function is proposed in [3]. While theoretically well formulated, this method requires the explicit knowledge of the system transfer function, which may not be always available, and which may be difficult to extract from C code. In addition, it propagates only the maximum value of the data. Thus, the estimated dynamic range can be very conservative.

A moment-based method is presented in [4]. This method models the input as a random variable and propagates its moments (up to 7th order) through the system. The probability density function (pdf) of all variables can be constructed from the propagated moments. However, this method assumes that the input data is temporally uncorrelated, and that variables internal to the system (such as at the input to an arithmetic operator) are spatially uncorrelated. Both these assumptions are not true in practice, and can have significant impact on accuracy of the results, as we will demonstrate in our results section.

A bitwidth/interval propagation method is adopted by [5]. This method propagates the input bitwidth or interval through the whole system to obtain the dynamic range for intermediate variables. The estimated result from this method is too pessimistic, since it always considers the worst case. If the system is large, these estimation errors can accumulate, leading to significant errors in dynamic range.

An improvement on the interval propagation technique is given in [6], where they do take into account the spatial correlation between internal variables. However, this method does not consider the temporal correlation of the input signals. For many applications, spatial correlation between variables is a direct result of the temporal correlation of the input signals. Therefore, ignoring the temporal correlation of the input automatically causes loss of true correlation information internally. This can cause large errors, as we will show in Fig. 3.

Thus, all existing methods have certain shortcomings, especially in the way that correlation is dealt with. In our work, we take care of both temporal input correlation and spatial internal correlation and we provide, not only ranges of the data, but its statistical distribution as well.

3. PROPOSED METHOD

Consider an application that can be modeled as a linear system with a single input data channel. We will focus on this case in this paper. However, it is straightforward to extend the proposed method to linear systems with multiple input channels. In order to model the unknown input data, and since the size of the input space is typically huge, we will model the input data stream as a discrete-time random process, i.e., a sequence of random variables (RVs) that are presented at the system input at discrete time-steps. Correspondingly, the system internal state and output variables all become random processes. Given the distribution of the input process, the dynamic range estimation problem can be formulated as the determination of the statistics of the random processes corresponding to the system state and output variables. Typically, the variance may be sufficient to determine dynamic range. However, we will show that we are able to estimate the whole probability distribution function (pdf), if needed.

3.1 Capturing Temporal Relation of Random Process by Karhunen-Loéve Expansion

A random process $p(t)$ defined over $[0, t_0]$ with zero mean and autocorrelation function $R(t_1, t_2)$, can be expressed using the following Karhunen-Loéve (KL) expansion [7]:

$$p(t) = \sum_{i=0}^{\infty} \sqrt{\lambda_i} f_i(t) \mu_i \tag{1}$$

where $f_i(t)$ and λ_i are referred to as the eigenfunction and eigenvalues, respectively, of the autocorrelation function. The eigenfunctions are also known to be *orthonormal*, i.e.:

$$\int_0^{t_0} f_i(t) f_j(t) dt = \delta_{ij} \tag{2}$$

where δ_{ij} is Kronecker delta function:

$$\delta_{ij} = \begin{cases} 0, & \text{if } i \neq j \\ 1, & \text{if } i = j \end{cases} \tag{3}$$

and where the μ_i are a set of zero-mean *orthonormal* RVs, which means:

$$E[\mu_i \mu_j] = \delta_{ij} \tag{4}$$

where $E[\cdot]$ is the mean or expected value operator. When the random process $p(t)$ is Gaussian (i.e., it has a normal distribution), a model which is often useful in practice, it turns out that the μ_i are all independent standard normal RVs. They may be referred to as a Gaussian basis. In cases where $p(t)$ is near-normal, the Gaussian basis can still be a good way to decompose the process in practice. In cases of large deviation from the normal, the Gaussian basis is not appropriate and one can then determine the nature of the μ_i basis (their distribution type) using standard KL techniques. The details are unimportant because one often does not need to know the exact nature of the μ_i basis, but only their moments. For our work, actually the 2nd order moments would seem to be sufficient, but we will show how higher order moments can be generated as well. The literature on KL techniques and their practical implementations is quite extensive. Notice that, irrespective of the distribution of the μ_i, the orthonormality property (4) guarantees that:

$$E[\mu_i^2] = 1 \tag{5}$$

and this fact will be useful later on to compute the variance, without having to know exactly the distribution of the μ_i basis.

It can be shown that the KL expansion is *optimal*, in the sense that the mean square error resulting from replacing the infinite summation in (1) by a truncated finite summation is minimal. Finally, one can obtain $f_i(t)$ and λ_i by solving the following integral equation:

$$\int_0^{t_0} R(t_1, t_2) f_i(t_1) dt_1 = \lambda_i f_i(t_2) \tag{6}$$

The above results are applicable to a continuous-time process. If we consider the discrete-time random process $p[k]$ to be defined on the discrete time domain $[0, n]$, then, as was done in [8], a KL expansion can be expressed in discrete-time as:

$$p[k] = \sum_{i=0}^{n} \sqrt{\lambda_i} f_i[k] \mu_i \quad k = 0, 1, \cdots, n \tag{7}$$

where λ_i and $f_i[k]$ are the eigenvalues and eigenfunctions of the autocorrelation matrix of the discrete-time random process $p[k]$:

$$\begin{bmatrix} R(0,0) & R(0,1) & \cdots & R(0,n) \\ R(1,0) & R(1,1) & \cdots & R(1,n) \\ . & . & & . \\ . & . & \cdots & . \\ . & . & & . \\ R(n,0) & R(n,1) & \cdots & R(n,n) \end{bmatrix} \begin{bmatrix} f_i[0] \\ f_i[1] \\ . \\ . \\ . \\ f_i[n] \end{bmatrix} = \lambda_i \begin{bmatrix} f_i[0] \\ f_i[1] \\ . \\ . \\ . \\ f_i[n] \end{bmatrix} \tag{8}$$

where $R(k_1, k_2) = E\left[p[k_1]p[k_2]\right]$ is the autocorrelation function. Here too, λ_i and $f_i[k]$ are orthonormal, and the summation can be truncated, yielding a least-squares-optimal expansion:

$$p[k] \approx \sum_{i=0}^{m} \sqrt{\lambda_i} f_i[k]\mu_i \quad k = 0, 1, \cdots, n \tag{9}$$

In this case, it can be shown that the relative mean square error resulting from the truncation is given by:

$$e = 1 - \frac{\sum_{i=0}^{m} \lambda_i}{\sum_{i=0}^{n} \lambda_i} \tag{10}$$

where $\lambda_0, \ldots, \lambda_m$ are the eigenvalues that are *kept* in the truncated KL expansion. We will refer to this error term as the *truncation error* of the KL expansion.

3.2 Responses of Linear Systems

Let $u[k]$ be the random process at the system input and $x[k]$ be the random process at an arbitrary state variable or at a system output. Then, we can write:

$$x[k] = \mathcal{L}(u[k]) \tag{11}$$

where $\mathcal{L}(\cdot)$ denotes the linear system operator that transforms $u[k]$ to $x[k]$. Let $u[k]$ have the following KL expansion:

$$u[k] = \sum_{i=0}^{m} \sqrt{\lambda_i} f_i[k]\mu_i = \sum_{i=0}^{m} u_i[k]\mu_i \tag{12}$$

where $u_i[k] = \sqrt{\lambda_i} f_i[k]$. Combining (11) and (12), gives:

$$x[k] = \mathcal{L}(u[k]) = \mathcal{L}\left(\sum_{i=0}^{m} u_i[k]\mu_i\right) \tag{13}$$

By the superposition property of linear systems, it follows that:

$$x[k] = \mathcal{L}(u[k]) = \sum_{i=0}^{m} \mathcal{L}\left(u_i[k]\right)\mu_i = \sum_{i=0}^{m} x_i[k]\mu_i \tag{14}$$

where $x_i[k] = \mathcal{L}(u_i[k])$. In this way, we can obtain the KL expansion of the random process $x[k]$ in terms of the system responses to each of the *deterministic* (non-random) functions $u_i[k]$ applied as input. In specific cases where a system transfer function is available, one can solve for $x_i[k]$ directly. However, in the more general case, and this is the approach that we take, even when the system is specified with a high-level behavioral description such as a C program, $x_i[k]$ can be obtained by simply simulating the system (e.g., executing the C program) with $u_i[k]$ as input. The order m of the KL expansion is typically small, as we will show in the results section, so that the complete statistics of the system internals and outputs may be obtained by simulating the system m times. The initial state of the simulation and the length of the simulation period are parameters that can be set depending on the particular situation. For example, in order to study the dynamic range during a system transient, the simulations can be performed starting from any desired initial state, and the largest variance of the resulting responses may be monitored. If the steady-state dynamic range is of interest, then the simulation period must be set long enough for the statistics of the responses to reach steady state. This, to some extent, depends on the statistics of the input process and on the system dynamics. Thus, a KL expansion is not a magic remedy that eliminates the need for simulation. Instead, KL is a way to drastically reduce the number of required simulations, compared to profiling based methods, as we will demonstrate in the results section.

3.3 Obtaining Statistics of State Variables

Once the required m simulations are complete, we can assemble the complete KL expansion for any system variable or output response $x[k]$ as:

$$x[k] = \sum_{i=0}^{m} x_i[k]\mu_i \tag{15}$$

It is important to note that this is a *complete* statistical description of the process $x[k]$. One can use this expansion to compute any desired probability associated with $x[k]$, such as the probability that it would exceed a certain threshold value, or simply compute the overall distribution of $x[k]$ from the known distributions of the μ_i RVs. Often times, the moments of the distribution are very useful, and they are perhaps easiest to compute from this expansion. The first-order moment is the mean or expected value of the process, which is known to be zero because the input is zero-mean. The 2nd-order moment, which is required to compute the variance, is given by:

$$E[x[k]^2] = E\left[\left(\sum_{i=0}^{m} x_i[k]\mu_i\right)^2\right] = \sum_{j=0}^{n} x_i[t]^2 \tag{16}$$

which is true because the μ_i have zero mean and unity variance (due to (5)).

If the system input is Gaussian, then all the internal and output responses are also Gaussian, so that the mean and variance are sufficient to capture the complete distribution. In the general case, higher order moments may be required and they may be obtained by similar, although slightly more involved, expansions that turn out to require terms such as $E\left[\mu_{j_1}^{m_1}\mu_{j_2}^{m_2}\cdots\mu_{j_k}^{m_k}\right]$, where $m_1 + m_2 + \cdots + m_k = m$. These terms may be obtained by using the KL transformation (9) in the reverse direction to compute samples of the RVs μ_i from the input data samples, and using these samples to compute terms like $E\left[\mu_{j_1}^{m_1}\mu_{j_2}^{m_2}\cdots\mu_{j_k}^{m_k}\right]$ using simple (Monte Carlo) averaging. The process is fairly straightforward but is omitted for brevity. Once the moments of the response $x(t)$ are obtained, one can further estimate its pdf using pdf expansion techniques such as Gram-Charlier, Hermite, or Edgeworth expansion [9]. These pdf estimation methods have been extensively applied in many research areas.

3.4 Methodology

This approach has been implemented in a tool that embodies the flow-chart shown in Fig. 1. It takes as input the behavioral description of the application in the form of a C program. It also takes some information on the system input, such as either input statistics or input sample data consisting of a set of time-domain traces that are samples or realizations of the input random process. The tool can output the pdf and statistics of all state variables and outputs. Optionally, it can make decisions on the bitwidth of all state variables and outputs, given a certain desired signal-to-noise ratio (SNR).

The input statistics can be correlation functions, such as the $R(k_1, k_2)$ matrix in (8), and other moments of the input. If the input random process is Gaussian or nearly-Gaussian, its mean and correlation function are sufficient to compute the statistics of the state variables and outputs. In the general case, other moments may also be needed depending on the order of the required state variable statistics. If input statistics are *not* available and sample data traces are provided instead, as shown in Fig. 1, we first extract the mean and correlation matrix, from the sample data (other higher order moments are computed if needed). Extracting the mean is done by simple averaging. The correlation matrix is obtained by averaging the cross-terms, as follows:

$$\frac{1}{m}\begin{bmatrix} p_1(0) & p_2(0) & \cdots & p_m(0) \\ p_1(1) & p_2(1) & \cdots & p_m(1) \\ \cdot & \cdot & & \cdot \\ \cdot & \cdot & \cdots & \cdot \\ \cdot & \cdot & & \cdot \\ p_1(n) & p_2(n) & \cdots & p_m(n) \end{bmatrix} \begin{bmatrix} p_1(0) & p_1(1) & \cdots & p_1(n) \\ p_2(0) & p_2(1) & \cdots & p_2(n) \\ \cdot & \cdot & & \cdot \\ \cdot & \cdot & \cdots & \cdot \\ \cdot & \cdot & & \cdot \\ p_m(0) & p_m(1) & \cdots & p_m(n) \end{bmatrix} \tag{17}$$

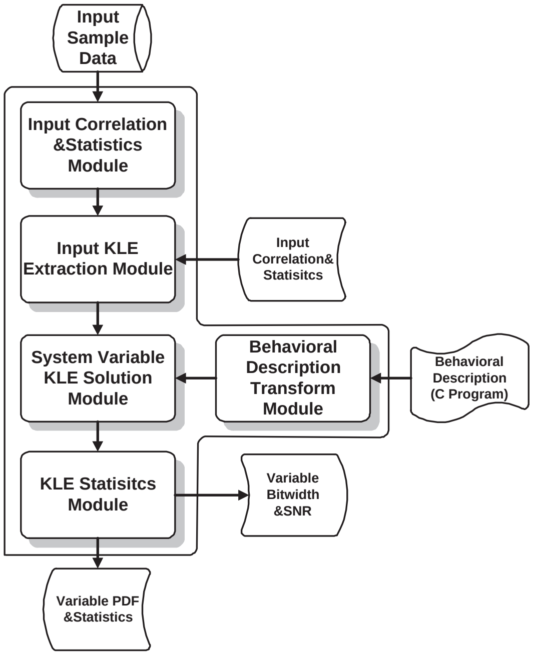
Input Sample Data
Input Correlation &Statistics Module
Input KLE Extraction Module
Input Correlation& Statisitcs
System Variable KLE Solution Module
Behavioral Description Transform Module
Behavioral Description (C Program)
KLE Statisitcs Module
Variable Bitwidth &SNR
Variable PDF &Statistics

Figure 1: Block diagram of KLE-based method.

If the mean is not zero, then the input process is decomposed into a deterministic term whose value is equal to the mean, and another zero-mean random process resulting from subtracting the mean from the original process. The corresponding mean of every system response can be computed by one execution of the system behavior using the input mean as excitation. Since the mean can be easily subtracted out up-front, and the response to the mean value easily added later on, it is sufficient to focus the discussion on the zero-mean case.

The KLE extraction module in the flow-chart accepts the correlation matrix as input and solves the eigensystem problem for this matrix. Solution techniques of such systems are standard. After the eigenvalues and eigenfunctions are found, the KL expansion of the input random process is available. It can easily be truncated according to the truncation error specified by the user using (10). Higher-order moments (higher than 2nd order) of the RVs μ_j are also computed by this module, from the moments of the input random process, if they are needed. In the system variable KLE solution module, the coefficient functions of the input KL expansion $\sqrt{\lambda_j}u_i[k]$ are utilized to compute the KL expansions for all variables in the C program (these represent all system state variables and system outputs), by simulating the system (executing the program) a total of m times.

The KLE statistics module uses the KLE model of all the variables to compute required statistics and probability distribution for them by the methods presented above. Users can specify a tolerable probability for the occurrences of overflow according to the noise and reliability requirement of the design. Then, dynamic ranges of all state variables associated with this probability can be determined based on their statistical information. According to the distribution of variables, this module can also determine the bitwidth of variables and compute the corresponding signal-to-noise ratio.

4. EXPERIMENTAL RESULTS

We construct a set of experiments to verify and demonstrate the effectiveness of the proposed methodology. All our experiments are conducted on a Sun workstation (Ultra 80, Model 4450). The set of benchmarks used includes FIR31, FIR63, CXFIR, IIR, IIR8, and FFT128. The first two, FIR31 and FIR63 are 31-order and 63-order FIR digital filters, respectively. CXFIR is a complex FIR filter, whose input and output are all complex numbers. IIR is a 2nd-order IIR digital filter. IIR8 is an 8th-order IIR digital filter. FFT128 performs a 128-point fast Fourier transform.

Table 1: KLE Extraction Result

Random Process	Truncation Error	Terms Kept	Correlation Func.(s)	KL Expan.(s)
rp1	2.91%	88	10.17	0.31
rp2	2.97%	82	10.19	0.28
rp3	2.99%	53	10.27	0.24
rp4	2.91%	19	10.45	0.13
rp5	2.92%	7	10.28	0.1

Sample sets of five input random processes are generated to conduct all experiments. These were applied to the benchmarks as input signals. These processes fit the Auto-Regression Moving Average (ARMA) model of time series [10], which is extensively used in engineering. Every sample set consists of 10,000 traces of 100-time-point each, for a total of 1 million data points in each sample set. The "randomness" of these sample sets were chosen to be different so as to provide extensive testing, as shown in Table 1. Thus, rp1 is the most noisy sample set, while rp5 shows the most correlated behavior. The "randomness" of data samples decreases from rp1 to rp5.

4.1 Results of KLE Extraction

We first demonstrate the accuracy and speed with which we extract a KL expansion and build a KLE model of random processes from the sample data set. Table 1 shows these results. It can be observed from columns 4 and 5 that all extractions can be accomplished within 10 seconds while retaining a truncation error of about 3%. Overall, the time complexity of our KLE extraction algorithm is linear in sample size. It takes about 1 second to extract a signal trace of 1,000 samples, and this scales linearly to about 10 seconds for a trace of 10,000 samples.

It is interesting to observe from column 3 that the more correlated the sample set is, the less terms are needed to capture its behavior for a specific truncation error. This is because the energy of more correlated sample sets are concentrated on only a few random terms in the KLE. On the other hand, for sample data sets which show more random behavior in the time domain, their energy is spread over many more terms. While the impact of the truncation on the estimation error is small and can be ignored, it can significantly reduce the computational work required.

One limitation of our approach is that, for the case when the input signal is white noise or nearly white noise, since every KLE term has the same significance, its KLE can not be truncated very effectively and almost all of the KLE terms are kept. However this case is rare, since one would expect that most practical signals include significant temporal correlation.

4.2 KLE versus Profiling

In this section, we demonstrate the accuracy and speed of KLE method by comparing against the traditional profiling-based method. The accuracy results are listed in Table 2. Note that these results are produced where all KL expansions are truncated to retain 97% of its energy (a truncation error setting of 3%). The second column and third column list the variances of the system outputs obtained from KLE-based method and profiling-based method respectively. The last column shows the difference in percentage between variances from KLE and profiling. The results from KLE is consistently close to the results from profiling. Even the largest difference is within 0.5%. It is important to note that this difference is only an artifact of truncation during input KL expansion. If all terms in KL expansion is kept, then the difference is essentially zero. KLE is accurate.

Fig. 2 further shows the histogram of an output variable in benchmark FIR31 when rp1 is applied as input. The bar graph and curve corresponds to the histograms from profiling and the estimated histograms from KLE-based method respectively. The

Table 2: Variance from profiling v.s. variance from KLE, with Sample data set rp1 as Input

Benchmark	KLE	Profiling	Difference
FIR31	0.050	0.050	-0.027%
FIR63	0.114	0.114	-0.009%
CXFIR	0.168	0.168	-0.009%
IIR	0.223	0.224	-0.460%
IIR8	57.997	57.998	-0.002%
FFT128	75.444	75.470	-0.038%

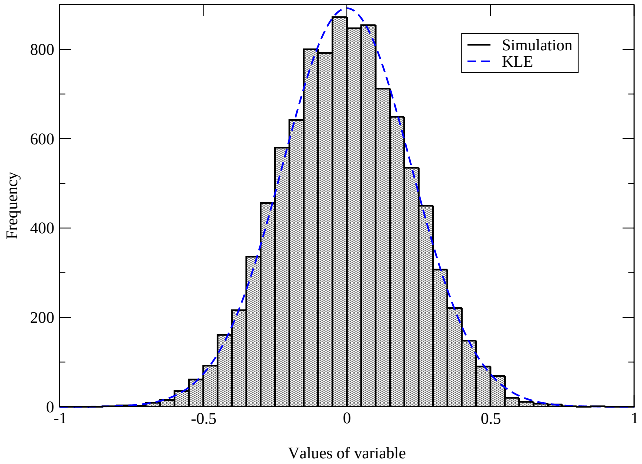

Figure 2: pdf and Histogram

total sample number for simulation is 10,000. It can be seen that these two histograms match very well. These results clearly verify that the KLE-based method is accurate and reliable. It is important to note that that while we use profiling result to verify the KLE method, it does not necessarily imply that profiling is more accurate than the KLE-based method. In fact, profiling can never reach the real theoretical values, because the simulated sample data can never be infinite. However, provided that the accurate statistics of input random processes are given, the exact statistics of state variables can always be obtained by KLE. For example, given the mean and correlation function of a Gaussian random process as input, then the exact models and statistics for state variables can be calculated.

Finally, Table 3 shows the computation time of our KLE-based method v.s. a profiling-based method. The KLE-based technique achieves about 100-time speedup. It is still several times faster than profiling even when the data processing time is included.

4.3 Bitwidth and SNR Tradeoff

As an application of our work, we will show how the distribution and statistics obtained by the KLE-based method can be used to make a tradeoff between bitwidth and the signal to noise ratio (SNR). Due to lack of space, and since this is only for demonstration purposes, this will be very brief, will focus on the overflow error only, and will be done only for the case of benchmark FFT128. This can be extended to cover round-off error as well, and can be applied to all the other benchmarks.

Table 3: Computation time, Simulation versus KLE

Benchmarks	KLE time(s)	Profiling time(s)	Speedup
FIR31	0.59	63.12	107
FIR63	0.62	69.48	112
CXFIR	0.53	58.12	110
IIR	0.52	57.48	111
IIR8	0.58	61.90	107
FFT128	0.33	27.28	83

Table 4: Bitwidth and Signal to Noise Ratio

Bitwidth	Dynamic Range	Probability	SNR(dB)
14	±8.191	65.44%	7.68
15	±16.383	94.07%	17.95
16	±32.767	99.98%	47.62
17	±65.535	≈ 1	148.34
18	±131.071	≈ 1	527.92

For a candidate bitwidth, one can easily compute the corresponding dynamic range, and the *range probability*, or the probability for the value to fall within the range, and the signal-to-noise ratio (SNR) using [11]:

$$SNR = 10\log_{10}\left(\frac{\sigma_s^2}{\sigma_n^2}\right) \tag{18}$$

where σ_s^2 and σ_n^2 are the variances of signal and noise respectively. Assuming the data is zero-mean Gaussian and is to be truncated to a dynamic range of $\pm z\sigma_s$, where $z > 0$, and ignoring discretization (round-off) error and focusing only on overflow error, it can be easily shown that the SNR is given by:

$$SNR = -10\log_{10}\left[2(1+z^2)\Phi(-z) - 2z\phi(z)\right] \tag{19}$$

where $\Phi(\cdot)$ and $\phi(\cdot)$ are the cumulative distribution function (cdf) and the probability distribution function (pdf) of the standard normal distribution. With this, one can generate results such as those in Table 4 that show the trade-off between bitwidth and SNR. Notice that, for a bitwidth b, and allowing for one sign bit, the dynamic range is $\pm(2^{b-1}\delta - 1)$, where δ is the discretization step size, i.e., the value of the least-significant bit. For the data in Table 4, $\delta = 0.001$. Based on the computed variances from KLE for any/all signals, one can generate and use such tables to manage the trade-off between bitwidth and SNR.

4.4 Impact of Temporal Correlation

In this section, we demonstrate that the temporal correlation in the input process can significantly impact the accuracy of the result. This experiment is important because all prior analytical methods did not consider temporal correlation. In fact, prior work either assumes that the input is a series of independent and identically distributed random variables (correlation coefficient $\rho = 0$), or a sequence of the same random variable ($\rho = 1$). This effectively leads to two extreme assumptions on temporal correlation, both unfounded: the former assumes that the input is completely uncorrelated, while the latter assumes that the input is completely correlated.

In order to show that these oversimplified assumptions may cause large estimation errors, we have repeated our experiments and artificially introduced a $\rho = 0$ (independence) assumption in one case and a $\rho = 1$ (correlated) assumption in another. The results are shown in Table 5. All of the three kinds of inputs in this table have exactly the same distribution at any specific time point; the only difference between them is the temporal correlation. In Table 5, the second column shows the variance from the KLE-based method. They are the same as those in Table 2 and verified by simulation. The variances listed in the third column are the results under the fully correlated assumption, while the variances in the fourth column are the results under independent assumption. We can see that the results from different temporal correlation are quite different. For FFT128, the variance estimated from the fully correlated assumption is about 30 times larger than the result from the KLE-based method, while the variance from independent assumption are about 4.4 times less than the KLE-based method.

Fig. 3 shows the pdf curves obtained by the KLE-based method and from the fully correlated case and the independent case, corresponding to the test case FIR31 in the second row of Table 5.

Table 5: Variances from KLE versus the cases where temporal correlation is simplified to be either 1 (full) or 0 (none), with sample data set rp1 as input.

Benchmarks	KLE	$\rho = 1$	$\rho = 0$
FIR31	0.050	0.224	0.012
FIR63	0.114	0.137	0.066
CXFIR	0.168	0.295	0.076
IIR	0.223	0.000	0.287
IIR8	57.997	173.124	14.784
FFT128	75.444	2218.93	17.335

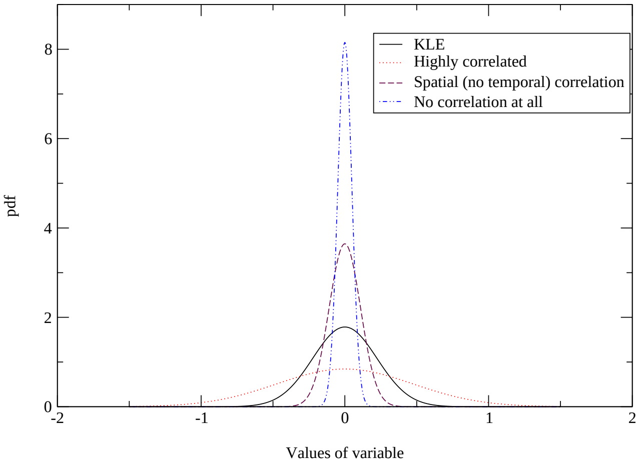

Figure 3: The pdf under various assumptions related to correlation, compared to the pdf from KLE.

The accuracy of pdf from KLE-based method has been verified by the histogram match in Fig. 2. The independence assumption gives a narrow distribution, while the extremely correlated assumption gives a flat distribution, both of which are quite wrong, as can be seen from the figure. The figure also shows what happens in the case when both temporal and spatial correlation is ignored, and the results are even worse in that case, as would be expected.

These results clearly demonstrate the advantages of the KLE approach and the need to maintain correlation information. Truncating one of the "wrong" pdfs in Fig. 3 would either give significant noise, or be very conservative. To illustrate this, we show the dynamic ranges under different range probabilities in Table 6. It can be observed that the extremely correlated assumption leads to dynamic range 5 times wider than necessary, a pessimistic result, while the independent assumption leads to dynamic range 2 times narrower than required, an overly optimistic result.

5. CONCLUSION

In this paper, a new method for dynamic range estimation is presented, based on the Karhunen-Loéve Expansion (KLE). It models the variables and intermediate results in the program by random processes. Starting from a description of the system behavior in the form of a C program, then by executing a transformed C program, the KL expansions for the random processes corresponding to all program variables can be generated. Based on this, full statistical information about the variables can be obtained. In this work, this technique has been applied to linear systems. We are currently extending this to the more general case.

Compared with currently available methods, this approach has the following advantages. Firstly, more detailed information about the dynamic range or values of the variables is obtained. It includes the probability distributions. Designers can associate every choice of bitwidth with a value of the signal-to-noise ratio and thus make judicious trade-offs between reliability and cost or power. Secondly, the proposed method fully considers both the spatial correlation and the temporal correlation. In contrast, previous methods that treat input as random variables cannot truly handle dynamic systems, especially those with feedback or operations on signal of multiple time points; and previous methods that treat input as white noise can lead to large estimation error. Thirdly, our method can construct random process models from real sample data or empirical statistics, instead of using oversimplified models or assumptions. Our method can therefore compete with profiling for accuracy and flexibility. Finally, our method is computationally efficient. If one only considers the propagation of random processes, it can be several orders of magnitude faster than simulation very easily. For designers who want to thoroughly explore the design space, they only need to extract the random process model once. When they modify their designs and redo the dynamic range estimation, their computation cost is only the propagation of the previously obtained models.

Table 6: Dynamic range from KLE versus from the cases where temporal correlation is simplified to be either 1 (full) or 0 (none).

Probability	KLE	$\rho = 1$	$\rho = 0$
99.98%	±32.14	±174.29	±15.41
99.02%	±22.41	±121.53	±10.74
95.00%	±17.02	±92.33	±8.16
90.10%	±14.33	±77.72	±6.87

6. REFERENCES

[1] Y. Cao and H. Yasuura. A system-level energy minimization approach using datapath width optimization. In *International Symposium on Low Power Electronics and Design*, pages 895–903, Huntington Beach, CA, August 2001.

[2] K. Kum and W. Sung. Combined word-length optimization and high-level synthesis of digital signal processing systems. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 20(8):921–930, August 2001.

[3] J. Carletta, R. Veillette, F. Krach, and Z. Fang. Determining appropriate precisions for signals in fixed-point IIR filters. In *Design Automation Conference*, pages 656–661, Anaheim, CA, June 2-6 2003.

[4] A. Garcia-Ortiz, L. Kabulepa, T. Murgan, and M. Glesner. Moment-based power estimation in very deep submicron technologies. In *International Conference on Computer Aided Design*, San Jose, CA, November 9-12 2003.

[5] S. Mahlke, R. Ravindran, M. Schlansker, R. Schreiber, and T. Sherwood. Bitwidth cognizant architecture synthesis of custom hardware accelerators. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 20(11):1355–1371, November 2001.

[6] C. F. Fang, , R. A. Rutenbar, M. Puschel, and T. Chen. Toward efficient static analysis of finite-precision effects in DSP applications via affine arithmetic modeling. In *Design Automation Conference*, pages 656–661, Anaheim, CA, June 2-6 2003.

[7] A. Papoulis. *Probability, Random Variables, and Stochastic Processes.* McGraw-Hill, New York, 1984.

[8] I. T. Jolliffe. *Principal Component Analysis.* Springer-Verlag, New York, 2002.

[9] J. E. Kolassa. *Series Approximation Methods in Statistics.* Springer-Verlag, New York, 1994.

[10] H. Shumway and D. S. Stoffer. *Time Series Analysis and Its Applications.* Springer-Verlag, New York, 2000.

[11] M. C. Jeruchin, P. Balaban, and K. S. Shanmugan. *Simulation of Communication Systems.* Kluwer Academic / Plenum Publishers, New York, 2000.