Automated Error Prediction for Approximate Sequential Circuits

Amrut Kapare, Hari Cherupalli, John Sartori
University of Minnesota
{kapar001, cheru007, jsartori}@umn.edu

ABSTRACT

Synthesis tools for approximate sequential circuits require the ability to quickly, efficiently, and *automatically* characterize and bound the errors produced by the circuits. Previous approaches to characterize errors in approximate sequential circuits have been based on simulations spanning all cycles of a sequential computation. These approaches, however, are not scalable and only accommodate small circuit modules and short computation times. In this paper, we observe that the statistical properties of errors in many approximate sequential circuits follow patterns that can be easily learned. Therefore, statistical error characteristics of these approximate sequential circuits can be predicted with high accuracy using only a few cycles of characterization data. Based on this novel observation, we propose a methodology for predicting error statistics in approximate sequential circuits that is accurate, fully automated, and has significantly lower overhead than prior approaches. Our methodology is robust to changes in predicted error metrics, circuit input distributions, and types of approximate hardware modules used in approximate circuits. We demonstrate the accuracy and scalability of our approach over a range of sequential circuits. On average, prediction inaccuracy is less than 2% and error characterization time is reduced by 99% compared to a simulation-based approach.

I. INTRODUCTION

Approximate circuit design is being explored as a means of improving energy efficiency in computing domains where errors can be tolerated [1]. While earlier work in the area focused on point solutions, several recent proposals focus on automating the design of approximate circuits [2, 3, 4, 5, 6]. Even for computing domains where noisy computations can be tolerated, automation represents an important hurdle that must be overcome before evaluation and use of approximate circuits can be considered by the mainstream as a viable solution for improving energy efficiency.

One challenge in automating approximate circuit design is that approximate circuit design requires fast, automated evaluation of the approximation error characteristics of a circuit. CAD tools perform many design optimizations during circuit synthesis, and at each step, a synthesis or optimization tool must ensure that the optimized circuit meets the output quality constraints specified by the circuit designer. To date, Monte Carlo-style simulation has been the established means of checking the output quality of approximate circuits. While this approach works for small combinational circuits, the approach is not scalable for use in approximate sequential circuits, where guaranteeing that a circuit meets error constraints over the course of an entire computation requires simulating the circuit over multiple cycles.

For example, consider sequential circuits used in two common noise-tolerant computing domains – audio and image processing. An audio processing application might use an approximate digital filter or signal transform (several of our benchmark circuits would be applicable in such an application domain). Processing a two-minute song at CD quality (44.1 kHz) requires more than 5 million cycles of computation. Existing error characterization approaches would perform multiple 5M-cycle simulations to sample and characterize the errors produced in each cycle of the sequential computation. As another example, consider an image processing application (which might use approximate filtering, edge detection, signal transformation, motion estimation, etc.). Processing a 5 Megapixel image requires more than 5 million cycles of computation. Simulating an entire sequential computation over such durations of computation to evaluate error constraints for approximate sequential circuits incurs unreasonably large overhead. Performing such a simulation multiple times, e.g., after multiple iterations of optimization applied by a CAD tool, is simply infeasible. Clearly, automated design of approximate sequential circuits requires *scalable* error evaluation techniques.

Furthermore, checking error constraints in approximate *sequential* circuits is complicated by the fact that errors in one cycle can continue to affect circuit outputs in future cycles. This is because errors produced in one part of the circuit can take multiple cycles to propagate through the circuit, and errors at circuit outputs can even be fed back into the circuit through feedback paths. This leads to various scenarios where the output error in an approximate sequential circuit can grow, attenuate, remain constant, or follow other patterns over the course of a computation. For example, Figure 1 shows average approximation error for several approximate sequential circuits with respect to duration of computation (in cycles). The figure shows that different approximate circuits exhibit different patterns of approximation errors.

While the error characteristics of different approximate sequential circuits are different, we make a novel observation about the nature of approximation errors in approximate sequential circuits. Namely, the approximations applied in approximate circuits are deterministic [7, 8, 9, 10, 11]. We observe that because of the deterministic nature of the circuit approximations, the statistical properties of approximation errors produced by the circuits can be learned and predicted with high accuracy. As opposed to simulation, which is computationally expensive, error prediction is scalable for large designs and long computations. Also, because of the deterministic nature of approximation errors, error prediction is accurate for arbitrarily long durations of computation. Our paper makes the following contributions.

• We observe that the deterministic approximations applied to create approximate sequential circuits provide an opportunity to predict the approximation errors generated

ICCAD '16, November 07-10, 2016, Austin, TX, USA

DOI: http://dx.doi.org/10.1145/2966986.2967007

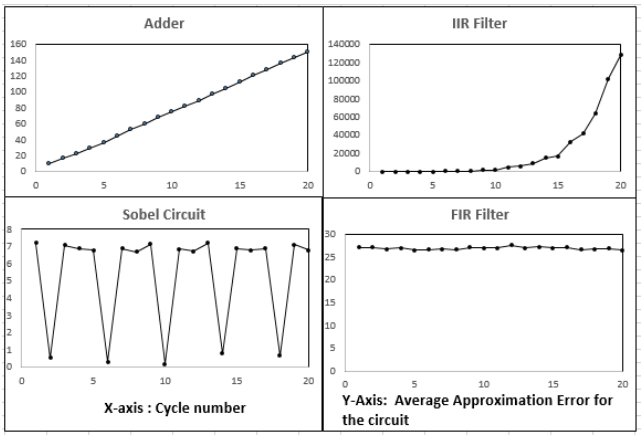

Figure 1: Different approximate circuit modules exhibit different patterns of approximation error with respect to cycle number. We will demonstrate that these patterns can be learned and predicted.

by the circuits with high accuracy and low overhead.

• We propose a methodology for estimating the statistical error characteristics of approximate sequential circuits that is both scalable and automated. Our *scalable automated* approach is designed to permit integration into the tight optimization / evaluation loops of modern CAD tools.

• We show that our methodology is robust to changes in predicted error metrics, circuit input distributions, and types of approximate hardware modules used in approximate circuits.

• We demonstrate that error prediction is both accurate and scalable. Over a collection of benchmark circuits, prediction inaccuracy is less than 2%, and error characterization time is reduced by 99% compared to a simulation-based approach. Reduction in characterization time increases with circuit complexity or duration of sequential computation.

II. RELATED WORK

Prior work on error evaluation for sequential circuits has been simulation-based, where simulations are performed to characterize the errors of a circuit in each cycle of a sequential computation. A recent work has proposed an automated approach for synthesis of sequential circuits [6], based on replacing accurate circuit modules with approximate circuit modules and evaluating approximation error constraints via circuit unrolling (time frame expansion) and simulation. Due to the large computational overhead of error evaluation based on simulation of unrolled sequential circuits, the approach is not scalable for use with large designs or long duration of computation. Even for small designs and short computations, the simulation-based error estimation approach has considerable overhead (see Section V). Our work uses error prediction to significantly reduce error evaluation time.

The work closest to ours [12] focuses on improving the efficiency of error evaluation in approximate *combinational* circuits using a regression-based approach. The work is based on characterizing the output error metric for an individual approximate arithmetic unit and propagating the error metrics from approximate arithmetic units connected together in a network to obtain the output error metric for the network. Since the error metric for an individual approximate arithmetic unit can depend on its input distribution and hardware characteristics, simulations are performed to characterize the error behavior of an approximate arithmetic unit for all possible combinations of input distribution and hardware configuration. Simulated error characterization data are stored in a set of lookup tables for reference during error estimation of an approximate combinational circuit consisting of multiple approximate modules.

The approach in [12] requires extensive simulations for pre-characterization of every error metric type, for each approximate arithmetic unit, under all possible combinations of input distributions and hardware configurations. The approach also requires multiple topological passes through a circuit to propagate error metrics between the nodes. Following these traversals, formulation and solving of a regression equation are required to determine the output error for a combinational circuit. In comparison, our approach works for sequential circuits and also has significantly lower characterization time, requiring only a single simulation for a limited number of cycles to characterize an entire circuit containing multiple approximate hardware modules. As an example of the difference in characterization time, our approach reduces error characterization time by a factor of 20,000× for an approximate MAC circuit containing 90 nodes, compared to the regression-based approach in [12].

In addition to its longer characterization time, the regression-based approach has a large memory overhead, since it must store two lookup tables for each possible combination of input distributions and hardware configurations, for every possible approximate arithmetic unit. The regression-based approach is also limited in its applicability to combinational logic circuits that only contain approximate adders. Our approach, based on prediction, has low memory and runtime overheads and can be used to evaluate error statistics for sequential circuits.

III. ERROR PREDICTION IN APPROXIMATE SEQUENTIAL CIRCUITS

We define the following terms to aid the discussion of our error prediction technique.

• *Approximate hardware module*: A hardware module that is functionally-incorrect by design, i.e., a circuit that only approximately implements the functionality of its accurate counterpart

• *Approximate circuit*: A circuit containing one or more approximate hardware modules

• *Accurate circuit*: A circuit containing only functionally-correct hardware modules

• *Approximation error (AE)*: A metric that quantifies the difference between the outputs of an accurate circuit and an approximate circuit with the same functionality

• *Predicted approximation error (PAE)*: A prediction of the approximation error for an approximate circuit

• *Prediction inaccuracy*: A metric that quantifies the difference between PAE and AE, relative to AE:

$$Prediction\ inaccuracy = \frac{PAE - AE}{AE}$$

Our automated methodology for evaluating error statistics of approximate sequential circuits is based on performing a simulation for a limited number of cycles to characterize the error behavior of a circuit and subsequently using the characterization data to generate an equation that is used to predict error statistics over an arbitrary number of cycles. The approach is based on our observation that the deterministic approximations applied in approximate hardware modules produce errors according to deterministic patterns. Consequently, the error statistics of approximate sequential circuits can be predicted accurately based on their initial behavior over a few cycles.[1]

Our methodology for predicting approximation errors works because the errors produced by approximate circuits are not random (in which case they would be unpredictable) but instead are determined by the nature of the approxima-

[1]To the best of our knowledge, all circuit approximation techniques applied in literature to date have been deterministic in nature and thus should be predictable.

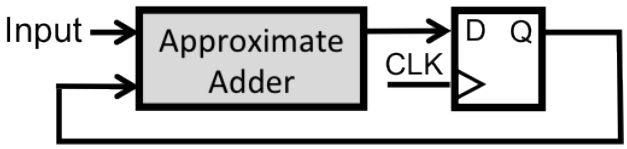

Figure 2: Approximate accumulator circuit.

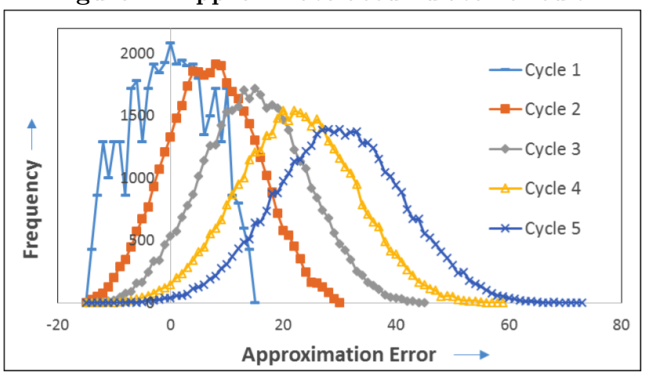

Figure 3: Histograms of sampled approximation error values for different cycle numbers in the approximate accumulator circuit of Figure 2.

tion(s) made in an approximate circuit's design. Approximate circuits are functionally incorrect by design and have deterministic, static approximations. Therefore, the statistical qualities of approximation errors are also deterministic and predictable.

We illustrate the predictable nature of approximation errors with examples. Consider the approximate N-bit accumulator circuit in Figure 2, where the least significant k bits of the adder are approximated by hardwiring them all to '1'. For a given value of $k < N$, approximation errors range from $-2^k - 1$ to $2^k - 1$. For a uniform input distribution, the average absolute error per cycle is $\frac{2^k-1}{2}$. Figure 3 shows the histograms of sampled approximation errors for the approximate accumulator over the first few cycles of operation. In the first cycle, the sampled errors are distributed over the range $[-2^k - 1, 2^k - 1]$. In each subsequent cycle, the average of the distribution increases by $\frac{2^k-1}{2}$, the average error per cycle. As explained above, deterministic approximations lead to predictable error statistics. Furthermore, sampling the errors produced over the first few cycles of the approximate accumulator's operation reveal a linearly increasing average error pattern with a slope of $\frac{2^k-1}{2}$. This deterministic pattern can be used to predict the average approximation error of the accumulator in any cycle during a computation.

As another example, consider an N-bit approximate multiplier circuit that does not perform the shift-and-add operation on the LSB of the multiplier. Thus, when the multiplier input is even (ending in '0'), the approximation error is zero. When the multiplier input is odd (ending in '1'), the approximation error equals the value of the multiplicand input. For a uniform input distribution, half of the samples will have approximation error equal to zero and half of the samples will have approximation error equal to the value of the multiplicand. For uniformly distributed multiplicands, the average value of the multiplicand is $\frac{2^N-1}{2}$, and the average approximation error is $\frac{2^N-1}{4}$ (since half of the results have errors and the other half have an error of zero).

Since predictability is based on the deterministic nature of the approximations applied in approximate hardware modules, errors in circuits containing multiple approximate hardware modules, each with their own deterministic approximations, are also predictable. Zooming out, an approximate circuit containing multiple approximate hardware modules could simply be viewed as a larger approximate hardware module with multiple deterministic approximations. For example, the errors in an approximate multiply-accumulator (MAC) circuit that approximately accumulates approximate multiplication results are also predictable. Since the input distribution of the approximate accumulator is influenced by the errors produced by the approximate multiplier, the average approximation error does not strictly increase as the sum of the average errors produced by the approximate adder and multiplier (i.e., $\frac{2^k-1}{2} + \frac{2^N-1}{4}$). However, it still follows a similar pattern that can be learned by sampling the approximation errors produced by the approximate MAC circuit and used to predict the error statistics for an arbitrary cycle. Given that input distributions of approximate hardware modules can be influenced by errors produced by other approximate modules, the exact behavior of approximation errors for an approximate sequential circuit can be learned by characterizing the circuit over a few cycles and using the characterization data to generate an equation that predicts errors for other cycles.

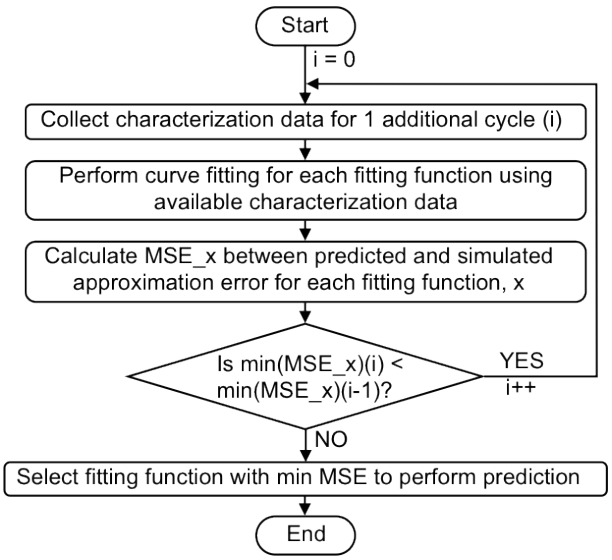

Figure 4: Our error prediction approach uses limited characterization to learn error behavior and predict error statistics over an arbitrary number of cycles.

Figure 4 summarizes our automated methodology for predicting error statistics in approximate sequential circuits. The automated methodology is described below.

Step 1: Gather characterization data: To characterize the approximation error behavior in an approximate sequential circuit, we simulate the accurate and approximate versions of the circuit to capture multiple samples of approximation error for a limited number of characterization cycles. An appropriate number of samples per cycle is predetermined empirically to minimize prediction inaccuracy and characterization time (see Figure 6 in Section V). We automatically determine the number of characterization cycles required to formulate an accurate prediction equation by incrementally increasing the number of cycles until the mean squared error (MSE) between predicted and simulated approximation error is minimized, as described in Step 2.

Step 2: Formulation of prediction equation: The characterization data gathered in Step 1 are used to determine the form and coefficients of the error prediction equation for a circuit. After gathering characterization data for one cycle, the data are used to perform curve fitting to determine coefficients for each of a collection of possible fitting functions that might describe approximation error vs. cycle

Table 1: Illustration for selecting a fitting function. Table values represent the MSE between the error predicted by a particular fitting function and the sampled error data, for a given number of characterization cycles. The linear fitting function minimizes MSE after using four cycles of characterization data to fit the error prediction equation.

Fitting Function	Cycle 1	Cycle 2	Cycle 3	Cycle 4	Cycle 5
Constant	30	38	49	56	65
Linear	35	25	11	2	2
Exponential	40	57	74	91	108
Square wave	30	42	59	76	94

number for a circuit. Constant, linear, periodic, and exponential fitting functions are evaluated.[2] Curve fitting is performed using numPy [13, 14]. The MSE between predicted and simulated approximation error values is calculated for each fitted prediction equation, and characterization data are collected for additional cycles as long as the minimum MSE among the candidate error prediction equations decreases with respect to the minimum MSE from the previous cycles. Since using more characterization data results in a better fit for an appropriate fitting function, MSE decreases monotonically for an appropriate fitting function but tends to increase for inappropriate fitting functions. When minimum MSE stops decreasing, the fitted prediction equation that minimizes MSE is selected to perform error prediction for the approximate circuit. Overflow in predicted error values is handled by ignoring (truncating) bits beyond the bit width of the approximate circuit (i.e., overflow bits), since these bits are likewise truncated by the hardware. Note that our high-level approach using simulation to perform characterization followed by inference is no less general than a purely simulation-based approach, so our approach can be used directly in a synthesis loop for an arbitrary approximate circuit.

Error prediction equation selection is illustrated with an example in Table 1. After characterization data are gathered for an additional cycle, the data are used to generate error prediction equations based on each possible fitting function. The values in Table 1 indicate the MSE between the error predictions of each of the prediction equations and the sampled error data. As additional cycles of characterization data are gathered and used to fit prediction equations, the MSE for constant, exponential, and square wave functions increases, indicating that these functions do not fit the sampled error data. On the other hand, the MSE for the linear prediction equation decreases as the set of characterization data grows, indicating that the error data fit a linear pattern. Since the MSE stops decreasing after four cycles, no additional characterization data are needed to fit the prediction equation. Furthermore, the linear equation determined by only four cycles of characterization data can be used to predict approximation error statistics in any arbitrary cycle of operation, even for a long computation over thousands or millions of cycles (see Section V).

IV. METHODOLOGY

We use the sequential circuits in Table 2 to evaluate our approach. The circuits are the same as those used in prior work on synthesis of approximate sequential circuits [6]. The circuits are also relevant in common noise-tolerant computing domains like audio and image processing. Note that exact selection of benchmark circuits is orthogonal to our work; our work is concerned with predicting error behavior in the circuits. Simulations to evaluate error statistics, inaccuracy, and characterization time are performed in Synopsys VCS. Collected data are processed in Python; curve fitting of characterization data is performed using the NumPy and SciPy libraries [13, 14] to derive analytical expressions that are used to predict approximation error statistics.

Table 2: Benchmark circuits

Benchmark	Description	Gate Count
Adder	16-bit Adder	112
Multiplier	8x8-bit Multiplier	378
MAC	16-bit Multiply Accumulator	1067
L1 Norm	Sum of Absolute Difference	639
L2 Norm	Euclidean Distance	1113
FIR Filter	4-tap FIR Filter	2775
IIR Filter	4-tap IIR Filter	1706
DCT	8-input Discrete Cosine Transform	5823
Butterfly	Butterfly operation (used in FFT)	474
Sobel	Sobel operation (used in edge detection)	708

Approximate sequential circuits are generated by replacing accurate hardware modules in the accurate versions of the circuits with approximate hardware modules (adders and multipliers).[3] Section III describes how approximations have been applied to adder and multiplier circuits. The approximate MAC circuit is created by connecting the output of an approximate multiplier to the input of an approximate accumulator. We use these three approximate hardware modules as basic building blocks in the benchmark circuits. For example, we replace the MAC unit that calculates $aX+b$ in the Butterfly circuit with an approximate MAC unit, where a and b are inputs and X is the twiddle factor. For the IIR filter, we replace the MACs farthest from output in both the feedforward and feedback paths with approximate MACs, since this keeps the approximation error within the output range of the filter. For the FIR filter, we replace all the MACs with approximate MACs. To create the approximate Sobel circuit, we replace all the multiplier and adder modules with approximate modules. To approximate L1 norm, we replace the adder that sums the absolute differences with an approximate adder. In the L2 norm circuit, we use approximate multipliers to calculate the square of the differences of the inputs, and we also use an approximate adder to sum the squared values. For DCT, we use approximate multipliers to calculate the product of the inputs from the multiplexers. We use an approximate adder to calculate the last 18 bits of the total input, in order to bound the approximation error within the output range of the circuit.

To measure prediction inaccuracy of derived equations for predicting approximation error statistics, we simulate approximate sequential circuits and the corresponding accurate circuits for 2000 cycles and compare simulated approximation error to predicted approximation error. The simulation length of 2000 cycles was chosen empirically, as we observed that prediction inaccuracy has settled and does not increase beyond 2000 cycles.

To provide a conservative bound on the approximation error reported for a given circuit, we inflate the predicted

[2]Experimentation with a large number of approximate circuit modules shows that the error behavior of all known approximate circuits in literature can be characterized by one of these fitting functions. Our methodology can easily be expanded to consider other fitting functions, if required.

[3]At this time, approximate hardware modules proposed in existing literature have largely been limited to arithmetic units – specifically, adders and multipliers.

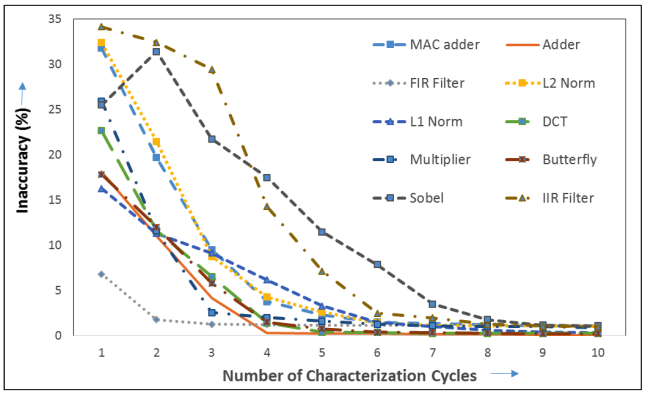

Figure 5: Prediction inaccuracy decreases and converges close to zero as the number of characterization cycles increases. For all circuits, around 10 cycles of characterization data are sufficient to minimize prediction inaccuracy.

approximation error by the maximum prediction inaccuracy. This approach guarantees that our tool never underestimates error statistics.

V. EXPERIMENTS AND RESULTS

This section presents the results of various experiments we performed to evaluate our automated error prediction methodology for a collection of approximate sequential circuits (Table 2). First, we demonstrate that only a small number of characterization cycles and samples are needed to accurately characterize the statistical error behavior of many common approximate sequential circuits. This is an important result, since characterization data are used to generate the prediction equation that is used to evaluate approximation error for all uncharacterized cycles. Collecting too few characterization cycles or samples might result in higher prediction inaccuracy. On the other hand, choosing too many characterization cycles or samples results in higher characterization time.

Figure 5 shows how prediction inaccuracy varies as the number of characterization cycles is increased. For these results, we used 100 samples per cycle (which is more than necessary for accurate characterization) to isolate the effect of varying the number of characterization cycles on the prediction inaccuracy. The next experiment (Figure 6) uses the minimum number of cycles determined in this experiment to determine the minimum number of samples per cycle needed. As mentioned in Section III, our automated methodology collects characterization data incrementally for additional cycles until MSE is minimized (which also corresponds to minimum prediction inaccuracy). The data show that the number of cycles required to minimize inaccuracy varies among the various circuits, but all the circuits can be characterized accurately with ten or fewer cycles of characterization data. The number of characterization cycles needed to accurately characterize approximation error varies because different circuits have different error behavior (see Figure 1). For example, characterizing an exponential function (e.g., IIR Filter) requires more data points than characterizing a linear function (e.g., Adder). Overall, the results demonstrate that only a small number of characterization cycles (around 10) are needed to accurately predict error behavior for these common approximate circuits.

Figure 6 shows how inaccuracy of prediction varies as the number of samples per characterization cycle increases. These results were collected using the minimum number of characterization cycles for each circuit that minimizes prediction inaccuracy, as determined by our automated methodology. The results show that collecting 32 samples of approximation error per characterization cycle is sufficient to minimize approximation inaccuracy for all the benchmark circuits. We use this empirically-determined number of samples in the rest of our experiments.

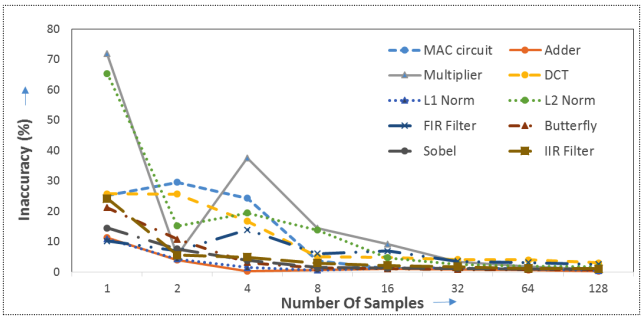

Figure 6: Prediction inaccuracy decreases and converges close to zero as the number of samples of approximation error collected for each characterization cycle increases. A small number of samples (around 32) is sufficient to accurately characterize error statistics for a given cycle.

Table 3: The table shows the prediction equations for average approximation error vs. cycle generated by our automated methodology for each of the benchmark circuits (k is the cycle number). Due to the deterministic nature of approximations, error statistics can often be described by simple functions, and a good fit can be achieved with only a few cycles of characterization data.

Circuit	Prediction Equation	Characterization Cycles
Adder	$7.52 \cdot k$	5
Multiplier	$63.46 \cdot k$	5
MAC	$69.71 \cdot k$	5
L1 norm	$7.41 \cdot k$	8
L2 norm	$68.50 \cdot k$	12
FIR Filter	26.94	5
IIR Filter	$85.09 \cdot e^{0.4192 \cdot k}$	16
DCT	$7.01 \cdot k$	8
Butterfly	253.19	6
Sobel	$\begin{cases} 7.20 : (k-2)(mod4) \neq 0 \\ 0.00 : (k-2)(mod4) = 0 \end{cases}$	16

We used our automated methodology to generate a prediction equation for each benchmark circuit that predicts the average approximation error in a given cycle. Table 3 shows the error prediction equations generated by our automated methodology. We have discussed the intuition behind the prediction equations for an approximate adder and multiplier in Section III. We find that the prediction equations for MAC, L1 norm, L2 norm, and DCT are also linear. This is not surprising, since these circuits all have an accumulator at the output stage, and the errors produced by approximate hardware modules in the circuits accumulate at the output. The equation for IIR filter, on the other hand, is exponential. This is due to the feedback paths in the circuit, which feed errors back into the circuit, causing error to grow exponentially. FIR filter has only feedforward paths, and therefore its average approximation error is constant and proportional to the filter coefficients. A similar scenario exists for the Butterfly and Sobel circuits. However, in the case of Sobel, the output flip-flops are reset periodically, causing the approximation error to behave like a square wave that oscillates between zero and a non-zero constant value.

Figure 7 shows prediction inaccuracy of the generated prediction equations as the number of computation cycles

increases, compared to simulation. Inaccuracy is slightly higher for initial cycles, as the circuit is initialized and settles into a stable state, but decreases rapidly and converges close to zero for all circuits, in most cases after only a few cycles. One nice characteristic of error prediction is that once error behavior is learned it can be applied to predict error statistics in any arbitrary cycle of a computation. Inaccuracy does not increase as the number of computation cycles increases; in fact, it decreases. This means that errors can be predicted accurately over computations lasting thousands or millions of cycles after performing only one low-overhead characterization of the circuit. Results are shown for up to 2000 cycles; however, we performed evaluations for a much larger number of cycles, confirming that inaccuracy does not increase for any circuit. On average, inaccuracy is less than 1.5% for all of the circuits. In many cases, inaccuracy is significantly lower, confirming that approximation error statistics are indeed predictable. As discussed in Section III, predictability of approximation errors is due to the nature of approximate circuits, which do not produce random errors but produce errors because of deterministic circuit approximations, which can be characterized.

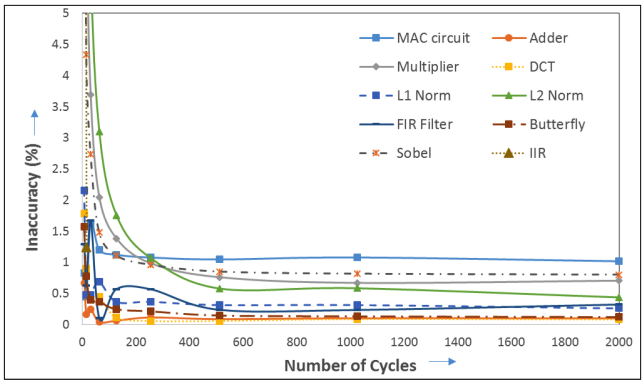

Figure 7: Prediction inaccuracy for average approximation error in a given cycle number.

Our error prediction methodology leverages the predictable nature of approximation errors to significantly reduce the time required to evaluate approximation error characteristics in an approximate sequential circuit. Figure 8 compares characterization time of a simulation-based approach for error evaluation against our prediction-based approach. All error characterization approaches used in prior work are simulation-based. The simulation-based approach used in our evaluations is based on the only prior work on synthesis of approximate sequential circuits [6]. Their simulation-based approach for evaluating error constraints characterizes a circuit for N cycles by unrolling the accurate and approximate versions of the circuit N times and adding a circuit to compare the outputs of the accurate and approximate circuits at the end of each unrolled cycle. Thus, characterization time of the simulation-based approach increases significantly as the number of cycles (N) or the circuit complexity increases. For our prediction-based error constraint evaluation, however, once characterization has been performed, the cost of predicting errors for any arbitrary number of cycles is small, since it only involves evaluating an equation. The figure also shows that since the number of characterization cycles needed is small, characterization time is small compared to that of a simulation-based approach. For example, for evaluating an error constraint over only 100 cycles, our methodology reduces characterization time by 88%, on average, compared to the simulation-based methodology used in [6]. Reduction in characterization time for our approach compared to simulation is even greater with increasing duration of computation or circuit complexity. For example, our approach reduces characterization time by 99%, on average, for a computation of only 1000 cycles. Note that, as discussed in Section I, many applications have significantly longer durations of computation, stressing the need for a scalable, automated methodology for evaluating error constraints.

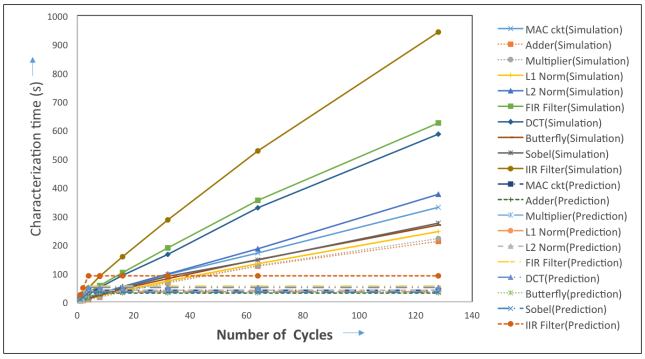

Figure 8: Characterization time for simulation-based error evaluation increases with circuit complexity and simulation duration. For error prediction, characterization time is small and bounded, since only a few characterization cycles are needed. Solid lines correspond to simulation and dashed lines correspond to prediction.

VI. GENERALITY

In this section, we discuss the generality of our prediction-based approach for error evaluation in sequential circuits. Specifically, we discuss the generality of our approach with respect to predicted error metrics, circuit input distributions, types of approximate hardware modules, and output error functions.

Error Metrics: The results in Section V have been presented for an average approximation error magnitude metric. However, our automated methodology can be used to predict other statistical properties of approximation errors as well, such as average relative error magnitude, mean squared error, error rate, maximum error, minimum error, etc. Our methodology to predict a different metric is unchanged, except that samples of the desired metric should be taken during characterization. In fact, since all these statistical error characteristics can be calculated using the same characterization data, we do not need to perform characterization again in order to predict other error metrics. The only scenario requiring additional characterization is when one metric requires more characterization cycles than another to minimize MSE, in which case our incremental automated methodology can be used to gather additional cycles of characterization data until MSE is minimized.

Figure 9 shows prediction inaccuracy as the number of computation cycles increases when using our automated methodology to predict maximum approximation error for the benchmark circuits. A designer or CAD tool might use maximum approximation error to bound the worst case errors produced by an approximate circuit. The results demonstrate that other types of approximation error metrics are also predictable for common approximate circuits, and our automated methodology can predict multiple types of error statistics with high accuracy.

Input Distribution: The exact prediction equation for an approximate circuit may depend on the circuit's input distribution. Our approach can easily handle a change in the input distribution by collecting a small number of error samples for the new input distribution and generating a new er-

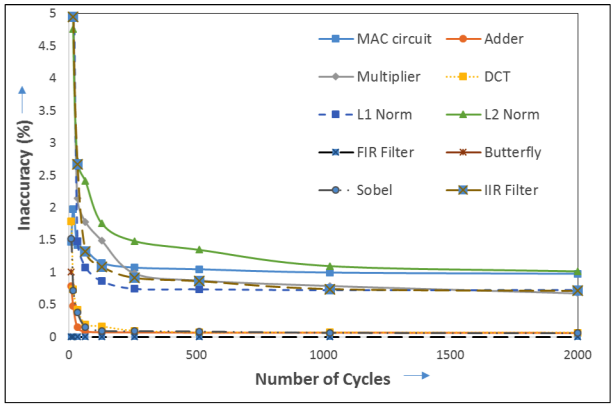

Figure 9: Prediction inaccuracy for maximum approximation error in a given cycle number.

Table 4: Characterizing prediction inaccuracy (%) across 100 different input distributions shows that our error prediction methodology is robust to changes in the input distribution.

Circuit	Average PI	Circuit	Average PI
Adder	0.33	Multiplier	1.21
MAC	1.26	L1 norm	0.35
L2 norm	1.48	FIR Filter	0.64
IIR Filter	1.24	DCT	0.43
Butterfly	1.67	Sobel	0.99

ror prediction equation based on the samples. Furthermore, since approximation errors are primarily determined by the type of deterministic approximation performed and are not necessarily sensitive to the input distribution, it may not even be necessary in most cases to perform a new characterization and generate a new prediction equation for a different input distribution. We studied sensitivity of prediction accuracy to changes in the input distribution by measuring prediction inaccuracy for all our circuits for 100 different input distributions – 50 uniform random distributions with randomly selected mean and range, and 50 normal distributions with randomly selected mean and standard deviation. Table 4 shows average prediction inaccuracy (PI) observed for the different input distributions. On average, the inaccuracy observed when using our prediction equations to predict errors across different input distributions was less than 1% over all the benchmark circuits. This shows that error prediction equations generated by our automated methodology are robust to changes in the input distribution.

Approximate Hardware Modules: The approximation errors produced by an approximate sequential circuit depend on the types of approximate hardware modules used in the circuit. To demonstrate that our error prediction methodology can predict errors accurately, independent of the types of approximate hardware modules used in an approximate circuit, we applied our methodology to our benchmark circuits implemented with many different types of approximate hardware modules, listed in Table 5. Table 5 represents all the relevant approximate hardware modules we could find in literature to replace the accurate hardware modules in our benchmark circuits.

Table 6 shows the error prediction equations generated by our automated methodology for each of the benchmark circuits implemented with the different types of approximate hardware modules listed in Table 5. The results show that even when different types of approximate hardware modules are used in the circuits, the errors produced by each benchmark circuit follow a characteristic pattern. Section V describes the reasons why errors produced by each sequential circuit follow a certain pattern. In general, the pattern of errors depends on the structure of the circuit, which determines how errors propagate to affect the circuit outputs. This result demonstrates that for a given sequential circuit, the type of fitting function that describes the output errors generally does not need to be relearned when optimizations are performed that substitute one type of approximate hardware module for another type. Such optimizations may be common for approximate synthesis and optimization tools [2, 3, 4, 5, 6] that attempt to find an implementation for an approximate circuit that maximizes energy savings while meeting specified output error constraints. As such, a CAD tool that uses our error prediction methodology may only need to learn the appropriate fitting function that characterizes errors once when it begins analyzing a circuit. Subsequent error characterizations can be faster, since they only need to determine new coefficients for a pre-learned fitting function.

Table 5: Approximate hardware modules found in literature to replace the accurate hardware modules in our benchmark circuits

Approximate Hardware Module	Reference
Accuracy Configurable Approximate Adder with 4-bit blocks (ACA-4B)	[7]
6-bit blocks (ACA-6B)	[7]
8-bit blocks (ACA-8B)	[7]
Almost Correct Adder (ACA-X)	[8]
Variable Latency Speculative Adder (VLSA)	[8]
Error Tolerant Adder Type I (ETAI)	[11]
Error Tolerant Adder Type II (ETAII)	[11]
Error Tolerant Adder Type II Modified (ETAIIM)	[9]
Lu's Adder (Lu)	[10]
Lu's Adder (Lu-8)	[10]

Table 7 shows prediction inaccuracy for our technique when different types of approximate hardware modules (from Table 5) are used in the benchmark circuits. The table characterizes the average prediction inaccuracy across all types of approximate module substitutions. The results show that approximation errors can be predicted accurately for approximate sequential circuits composed of many different types of approximate hardware modules.

Output Error Functions: Our prediction-based methodology for error evaluation determines an appropriate fitting function to describe the statistical error characteristics of an approximate sequential circuit in a given cycle. Constant, linear, periodic, and exponential fitting functions are able to characterize all of our benchmark circuits. A constant fitting function characterizes the errors in a circuit where erroneous outputs are not fed back into the circuit. A linear fitting function describes a scenario where errors accumulate at the output stage of a circuit. A periodic fitting function is possible when the output is cleared periodically by a reset signal, and an exponential fitting function describes a scenario where erroneous outputs are fed back into a circuit and amplified.

Our approach can be used to predict errors in circuits with deterministic approximations. Other types of sequential circuits with deterministic approximations may exhibit other characteristic error patterns, requiring additional fitting functions. Inadequacy of existing fitting functions would be indicated in our methodology if the MSE for all fitting functions continues to increase as additional cycles of characterization data are used to fit error prediction equations (see Section III). Our error prediction tool is parameterized to easily accommodate additional fitting functions.

VII. CONCLUSIONS

CAD tools that perform synthesis and optimization for approximate sequential circuits require the ability to quickly,

Table 6: The prediction equations for average approximation error vs. cycle (k is the cycle number) show that the pattern of errors produced by an approximate sequential circuit is a characteristic of the circuit. For a given circuit, our error prediction methodology determines different coefficients for the same fitting function when different approximate hardware modules are used in the circuit.

Approx. HW Module	Adder	MAC	L1 norm	L2 norm	FIR Filter	DCT	Butterfly	IIR	Sobel
ACA-4B	$127.42 \cdot k$	$69.91 \cdot k$	$86.58 \cdot k$	$11127.34 \cdot k$	980.51	$1192.04 \cdot k$	556.8	$263.13 \cdot e^{0.6225 \cdot k}$	$\begin{cases} 496.34 : ((k-2)(mod4) \neq 0) \\ 210.49 : ((k-2)(mod4) = 0) \end{cases}$
ACA-6B	$123.91 \cdot k$	$70.21 \cdot k$	$85.44 \cdot k$	$11104.63 \cdot k$	573.59	$1186.28 \cdot k$	484.15	$266.91 \cdot e^{0.6226 \cdot k}$	$\begin{cases} 495.06 : ((k-2)(mod4) \neq 0) \\ 235.39 : ((k-2)(mod4) = 0) \end{cases}$
ACA-8B	$128.49 \cdot k$	$70.08 \cdot k$	$86.19 \cdot k$	$11268.52 \cdot k$	817.23	$1167.23 \cdot k$	546.94	$264.89 \cdot e^{0.6219 \cdot k}$	$\begin{cases} 498.28 : ((k-2)(mod4) \neq 0) \\ 240.81 : ((k-2)(mod4) = 0) \end{cases}$
ACA-X	$106.82 \cdot k$	$70.24 \cdot k$	$86.48 \cdot k$	$11127.94 \cdot k$	570.46	$1178.91 \cdot k$	463.56	$262.1 \cdot e^{0.6234 \cdot k}$	$\begin{cases} 486.54 : ((k-2)(mod4) \neq 0) \\ 219.76 : ((k-2)(mod4) = 0) \end{cases}$
VLSA	$115.14 \cdot k$	$69.71 \cdot k$	$86.95 \cdot k$	$11253.71 \cdot k$	592.41	$1177.18 \cdot k$	461.33	$259.73 \cdot e^{0.6224 \cdot k}$	$\begin{cases} 503.25 : ((k-2)(mod4) \neq 0) \\ 241.74 : ((k-2)(mod4) = 0) \end{cases}$
Lu	$114.53 \cdot k$	$70.11 \cdot k$	$86.64 \cdot k$	$11004.86 \cdot k$	739.35	$1192.48 \cdot k$	480.73	$267.86 \cdot e^{0.6227 \cdot k}$	$\begin{cases} 482.43 : ((k-2)(mod4) \neq 0) \\ 211.86 : ((k-2)(mod4) = 0) \end{cases}$
Lu-8	$114.85 \cdot k$	$70.22 \cdot k$	$85.92 \cdot k$	$11229.1 \cdot k$	553.36	$1177.29 \cdot k$	456.39	$269.65 \cdot e^{0.6214 \cdot k}$	$\begin{cases} 504.85 : ((k-2)(mod4) \neq 0) \\ 242.96 : ((k-2)(mod4) = 0) \end{cases}$
ETAI	$123.58 \cdot k$	$70.19 \cdot k$	$86.23 \cdot k$	$11017.48 \cdot k$	997.42	$1182.93 \cdot k$	535.26	$260.49 \cdot e^{0.6231 \cdot k}$	$\begin{cases} 483.21 : ((k-2)(mod4) \neq 0) \\ 217.24 : ((k-2)(mod4) = 0) \end{cases}$
ETAII	$119.28 \cdot k$	$69.85 \cdot k$	$85.89 \cdot k$	$11449.67 \cdot k$	1009.95	$1179.19 \cdot k$	452.83	$256.77 \cdot e^{0.6227 \cdot k}$	$\begin{cases} 506.34 : ((k-2)(mod4) \neq 0) \\ 243.28 : ((k-2)(mod4) = 0) \end{cases}$
ETAIIM	$118.02 \cdot k$	$70.54 \cdot k$	$86.47 \cdot k$	$11052.29 \cdot k$	552.7	$1179.73 \cdot k$	455.47	$262.83 \cdot e^{0.6229 \cdot k}$	$\begin{cases} 484.89 : ((k-2)(mod4) \neq 0) \\ 212.90 : ((k-2)(mod4) = 0) \end{cases}$

Table 7: Characterizing prediction inaccuracy (%) across different types of approximate hardware modules shows that our error prediction methodology can accurately predict errors independent of the types of approximate hardware modules used in an approximate sequential circuit.

Circuit	Adder	MAC	L1 norm	L2 norm	FIR Filter	DCT	Butterfly	IIR Filter	Sobel	**Average**
Average PI	0.87	0.38	0.22	0.54	4.71	0.79	3.78	2.15	3.65	1.89

efficiently, and automatically characterize and bound the errors produced by the circuits. The previous approach [6] to characterize errors in approximate sequential circuits is based on simulations spanning all cycles of a sequential computation. This approach, however, is not scalable and only accommodates small circuit modules and short computation times. We observed that the statistical properties of errors in approximate sequential circuits follow patterns that can be learned. We leveraged this observation to demonstrate that the statistical error characteristics of many common approximate sequential circuits can be predicted with high accuracy using only a few cycles of characterization data. We proposed an automated methodology for predicting error statistics in approximate sequential circuits and demonstrated its accuracy and scalability over a range of circuits. Our methodology is robust to changes in predicted error metrics, circuit input distributions, and types of approximate hardware modules used in approximate circuits. On average, prediction inaccuracy is less than 2% and runtime is reduced by 99% compared to a simulation-based approach. Runtime reduction continues to increase as circuit complexity and duration of computation increase, emphasizing the scalability of our automated approach for error characterization in approximate sequential circuits.

Acknowledgments

The authors would like to acknowledge the contributions and invaluable feedback provided by Prof. Rakesh Kumar (UIUC), which significantly improved this work. Thanks are also due to Karthikeyan Thavasi, who performed experiments to characterize the impact of different input distributions on prediction accuracy.

VIII. REFERENCES

[1] Yen-Kuang Chen, J. Chhugani, P. Dubey, C.J. Hughes, Daehyun Kim, S. Kumar, V.W. Lee, A.D. Nguyen, and M. Smelyanskiy. Convergence of recognition, mining, and synthesis workloads and its implications. *Proceedings of the IEEE*, 96(5):790–807, May 2008.

[2] Swagath Venkataramani, Amit Sabne, Vivek Kozhikkottu, Kaushik Roy, and Anand Raghunathan. Salsa: Systematic logic synthesis of approximate circuits. In *Proceedings of the 49th Annual Design Automation Conference*, DAC '12, pages 796–801, New York, NY, USA, 2012. ACM.

[3] Jin Miao, Andreas Gerstlauer, and Michael Orshansky. Approximate logic synthesis under general error magnitude and frequency constraints. In *Proceedings of the International Conference on Computer-Aided Design*, ICCAD '13, pages 779–786, Piscataway, NJ, USA, 2013. IEEE Press.

[4] Swagath Venkataramani, Kaushik Roy, and Anand Raghunathan. Substitute-and-simplify: A unified design paradigm for approximate and quality configurable circuits. In *Design, Automation Test in Europe Conference Exhibition (DATE), 2013*, pages 1367–1372, 2013.

[5] K. Nepal, Yueting Li, R.I. Bahar, and S. Reda. Abacus: A technique for automated behavioral synthesis of approximate computing circuits. In *Design, Automation and Test in Europe Conference and Exhibition (DATE), 2014*, pages 1–6, 2014.

[6] A. Ranjan, A. Raha, S. Venkataramani, K. Roy, and A. Raghunathan. Aslan: Synthesis of approximate sequential circuits. In *Design, Automation and Test in Europe Conference and Exhibition (DATE)*, pages 1,6,24–28, 2014.

[7] Andrew B Kahng and Seokhyeong Kang. Accuracy-configurable adder for approximate arithmetic designs. In *Proceedings of the 49th Annual Design Automation Conference*, pages 820–825. ACM, 2012.

[8] Ajay K Verma, Philip Brisk, and Paolo Ienne. Variable latency speculative addition: A new paradigm for arithmetic circuit design. In *Proceedings of the conference on Design, automation and test in Europe*, pages 1250–1255. ACM, 2008.

[9] Ning Zhu, Wang Ling Goh, Weija Zhang, Kiat Seng Yeo, and Zhi Hui Kong. Design of low-power high-speed truncation-error-tolerant adder and its application in digital signal processing. *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on*, 18(8):1225–1229, 2010.

[10] Shih-Lien Lu. Speeding up processing with approximation circuits. *Computer*, 37(3):67–73, 2004.

[11] Ning Zhu, Wang Ling Goh, and Kiat Seng Yeo. An enhanced low-power high-speed adder for error-tolerant application. In *Integrated Circuits, ISIC'09. Proceedings of the 2009 12th International Symposium on*, pages 69–72. IEEE, 2009.

[12] W.-T.J. Chan, AB Kahng, S. Kang, R. Kumar, and J. Sartori. Statistical analysis and modeling for error composition in approximate computation circuits. In *Computer Design (ICCD), 2013 IEEE 31st International Conference*, pages 47,53,6–9, 2013.

[13] Eric Jones, Travis Oliphant, Pearu Peterson, et al. SciPy: Open source scientific tools for Python, 2001–.

[14] Stefan van der Walt, S. Chris Colbert, and Gael Varoquaux. The NumPy Array: A structure for efficient numerical computation,, 2011.