Approximation through Logic Isolation for the Design of Quality Configurable Circuits

Shubham Jain, Swagath Venkataramani, Anand Raghunathan
School of Electrical and Computer Engineering, Purdue University
{jain130,venkata0,raghunathan}@purdue.edu

***Abstract*—Intrinsic application resilience, a property exhibited by many emerging application domains, allows designers to optimize computing platforms by approximating selected computations within an application without any perceivable loss in its output quality. At the circuit level, this is often achieved by designing circuits that are more efficient but realize slightly modified functionality. Most prior efforts on approximate circuit design hardwire the degree of approximation into the implementation. This severely limits their applicability, as intrinsic resilience significantly varies both across and within applications, and often the same computation needs to be executed at different levels of accuracy when the application processes a different input or used in a different context.**

To address this limitation, in this work, we propose a new approach to design *quality configurable circuits* that are equipped to modulate their output accuracy and energy at runtime. Our approach, *approximation through logic isolation*, identifies portions of logic in the circuit that consume significant power, but contribute only minimally to output accuracy. One or more approximate modes of circuit operation are then enabled by isolating the identified logic (using muxes, latches or power gating cells) to benefit power while satisfying the desired output accuracy. We propose a systematic methodology to transform a given circuit into a quality-configurable circuit by applying the proposed technique. Our methodology generates a favorable energy-quality trade-off by deliberately creating opportunities for error compensation between multiple logic islands that are simultaneously isolated. This enables more aggressive approximation for a given output quality, leading to a superior power benefits. We evaluate the proposed methodology using a wide range of arithmetic circuits, complex modules and datapaths. The synthesized quality configurable circuits support 3 quality modes *viz.* accurate, <0.1% average error, and <0.2% average error. Power improvements achieved in the approximate modes are 8.4%-34.5% and 17%-51.5%, respectively.

***Index Terms*—Approximate Computing, Approximate Circuits, Logic Synthesis, Quality Configurable Logic Isolation**

I. Introduction

In recent years, new application domains such as recognition, data mining, search, and analytics have gained prominence, and are expected to shape the demands placed on future computing platforms [1]. Applications from these domains exhibit significant *intrinsic application resilience*, or the ability to produce outputs that are acceptable even if some of their computations are carried out in an imprecise or approximate manner [2], [3].

Realizing the forgiving nature of applications, several research efforts spanning across the entire computing stack, from software to architecture and circuits [3]–[8], have proposed design techniques that achieve improvements in efficiency (performance or energy) at acceptable output quality. At the circuit level, which is focus of this work, many manual approximate design techniques [5], [6], [9], [10] that are applicable to specific circuits such as adders and multipliers have been proposed, followed by automatic design methodologies [7], [8], [11]–[13] that can be applied to arbitrary circuits.

Quality-configurable circuits. A key limitation of almost all the above efforts is that they hardwire the degree of approximation into the circuit implementation, *i.e.*, the implementation is specialized to meet a single quality constraint, precluding the possibility of achieving accurate outputs or outputs of other quality levels. This severely limits their applicability for the following reasons: (i) It is common for multiple computations within an application to be mapped to the same hardware (resource sharing). If these computations are error-resilient to different degrees, then the underlying hardware needs to support multiple quality levels; (ii) The resilience of even a single application (and its computations) varies significantly based on the input being processed, and the context in which its output is used [3], [14]. Consequently, the degree to which each computation is approximated needs to be modulated; (iii) Finally, approximate circuits may be used to design a programmable platform that executes a range of applications with varying resilience. As a result, the degree of approximation needs to be varied based on the application under execution.

All the above scenarios underscore the need for *quality configurable circuits*, or circuits whose accuracy can be reconfigured at runtime. In this work, our objective is to develop a systematic methodology for the design of such circuits.

Despite their widespread applicability, very few efforts have previously explored the design of quality configurable circuits. An adder whose accuracy can be modulated by selectively accumulating partial summations was proposed in [15]. This technique is specific to adders and cannot be generalized to arbitrary circuits. SASIMI [13] proposed an approach in which inputs to selected nodes in the circuit are substituted by signals with slightly different functionality. A significant limitation of this approach is that it impacts the interface timing behavior of the circuit, *i.e.*, the resultant quality configurable circuit takes a variable number of execution cycles across inputs, even for a single quality level. As a result, it is difficult to compose such circuits into larger systems.

Approximation through logic isolation. To address the above limitations, we propose a new approach to design quality configurable circuits, called *approximation through logic isolation*. The key idea behind our approach is to identify portions of logic in the circuit that contribute minimally to output accuracy, and isolate them (using operand isolation or power gating) from the circuit, effectively setting their outputs to fixed values. This leads to improvements in power both within the logic that is isolated as well as in its downstream. On the other hand, the errors introduced at the outputs of the isolated logic may propagate all the way to the circuit outputs, thereby impacting output quality. Thus, logic isolation yields a trade-off between the circuit's energy and quality. Quality configurable execution can be achieved by identifying the isolation regions in a quality-aware manner at design time, and at run time selecting which regions should be isolated based on the desired output quality.

We develop a systematic methodology to construct quality

978-3-9815370-6-2/DATE16/

configurable versions of any given circuit using our approach. The methodology iteratively identifies parts of the circuit to isolate that result in maximal energy savings while introducing the least amount of error. An interesting phenomenon inherent to logic isolation based approximation, which is leveraged by our methodology, is that two or more logic islands may mutually compensate their errors when isolated simultaneously, *i.e.*, the total error at the circuit output decreases as a result of isolating both logic portions compared to isolating any one of them. This provides further room to approximate the circuit, leading to superior energy benefits for a given quality. When selecting candidates for isolation, our methodology actively searches for compensation opportunities to maximize the energy benefits.

The key contributions of this work are summarized as follows:

- We propose *approximation through logic isolation*, a new approach to design quality configurable circuits whose accuracy and energy can be modulated at runtime.
- We develop a systematic methodology to realize our approach. Our methodology iteratively identifies logic portions to isolate, and leverages error compensation to achieve a superior energy *vs.* quality trade-off.
- We evaluate our approach on a wide range of arithmetic circuits, complex modules and complete datapaths, and demonstrate 8.4%-35.5% (17%-51.5%) improvement in energy for $<$0.1% ($<$ 0.2%) loss in output quality.

The rest of the paper is organized as follows. Section II presents an overview of prior research efforts related to this work. Section III and Section IV describe the proposed approach and design methodology, respectively. Section V explains the experimental setup used in our evaluation. The results are subsequently presented in Section VI. Finally, Section VII concludes the paper.

II. Related Work

Approximate computing techniques have been proposed across all layers of design abstraction. In this section, we limit our discussion to the design of approximate circuits, which is the focus of this work.

Approximations can be introduced in circuits in two different ways, *viz.* overscaling based approximation and functional approximation. In overscaling based approximation, the voltage supply to the circuit is lowered to a point where critical paths in the circuit fail to complete, resulting in timing errors. To mitigate catastrophic errors due to the large number of near-critical paths, methodologies based on cell sizing [16], logic restructuring [17], and retiming [18] have been proposed to shape the path delay distribution of overscaled circuits.

Functional approximation techniques modify the Boolean functionality of the circuit to achieve power/performance benefits. Research in this direction began with manual designs of specific arithmetic components such as multipliers [6] and adders [5], [9], [10], [15]. Later, to make these approaches scalable and applicable to arbitrary circuits, a range of design methods were proposed that approximate circuits by complementing selected minterms [12], introducing stuck-at-faults and propagating the redundancies [8], pruning paths based on activation probability [7], and identifying don't cares arising from relaxed quality specifications [11]. Approximate circuits generated by the above approaches can operate only at a fixed quality level. In this work, we focus on the design of quality configurable circuits to address the need for quality reconfigurability at runtime.

In the context of quality configurable circuits, [15] proposed an adder design in which the input operands are split into multiple segments, and each segment is independently summed up. The partial results from the segments are conditionally composed together based on the desired output accuracy. A programmable truncated multiplier based on power-gating was proposed [19], in which the output precision is traded off to obtain power savings. SASIMI [13] proposed the first quality configurable synthesis methodology that is applicable to arbitrary circuits. For a given node in the circuit, SASIMI identifies another node that assumes the same value with high probability and substitutes one for another, thereby approximating the circuit. When accurate operation is desired, the original node is retained and the circuit is re-evaluated using an additional cycle.

A key shortcoming of SASIMI is that it inherently turns a combinational circuit into a variable latency circuit, *i.e.*, even for a given quality level, two inputs to the circuit can take different numbers of cycles to complete. Therefore, integrating SASIMI circuits into larger systems is quite challenging, especially in the case when multiple such circuits operate in synchrony. For example, consider the common scenario of an accelerator that contains an array of compute blocks operated in a SIMD fashion [4]. If all the compute blocks were replaced with SASIMI circuits and operated in parallel, the slowest block would limit the overall progress of the array, diminishing the benefits from approximate computing in the other blocks, while also adding overheads in terms of control logic for synchronization *etc.* In contrast, our proposal, approximation through logic isolation, *does not alter the timing behavior of the circuit*, and therefore the resulting circuits can easily be plugged into larger designs.

III. Design Approach

Quality Configurable Circuits (QC-Ckts) are circuits that can dynamically modulate the degree to which they approximate their functionality based on the desired output accuracy. Figure 1 shows a conceptual block diagram of a quality configurable circuit. As seen in Figure 1, in addition to the circuit's inputs and outputs, QC-Ckts take in quality control inputs ($Q - CTRL$) that can be used to select a quality level. QC-Ckts typically contain a quality control unit that interprets the quality control inputs, and regulates the extent to which different parts of the circuit needs to be approximated, so that the overall output quality is maintained. We propose a new design approach and a systematic methodology to design QC-Ckts. Given an input circuit and a list of quality levels that needs to be supported, our methodology generates a quality configurable version that can seamlessly transition across quality levels at runtime, while maximizing efficiency at each quality level. In this section, we describe our design approach and the various trade-offs involved in the design of QC-Ckts.

A. Approximation through logic isolation: Concept

This section explains the key concepts behind our design approach. The idea is to identify portions of logic in the circuit and isolate them in a quality-aware manner, *i.e.*, suppress the selected logic portions from being evaluated provided the output satisfies the desired quality requirement. Figure 1 explains the concept in further detail. Some portions of logic in the circuit (shaded ellipses) are identified as candidates

for isolation, and the circuit is embedded with logic that suppresses them from being evaluated while forcing their outputs to predefined values. During circuit operation, the quality control unit interprets the input quality specifications, and activates a subset of these logic islands, while isolating the rest. When fully accurate outputs are desired, all the logic islands are activated. More and more islands are isolated from the circuit as the quality constraints are progressively relaxed. Thus, quality configurable execution is achieved at runtime.

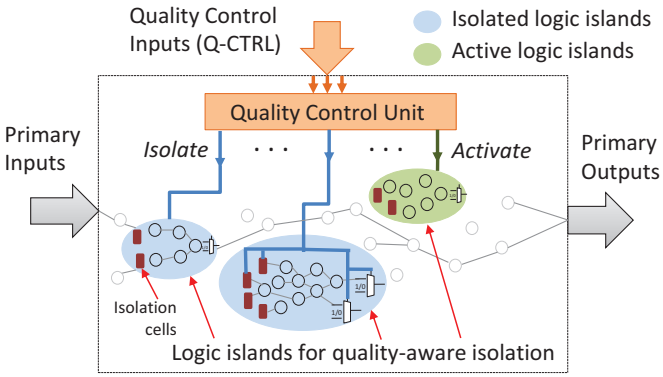

Fig. 1. Approximation through logic isolation: Concept

There are various ways in which a portion of logic can be isolated. The simplest approach would be to use latches or AND/OR gates at the inputs of the island to prevent switching activity within it. Similarly, muxes can be inserted at the island outputs to force them to fixed logic values. An alternative approach would be to power-gate the island by inserting isolation cells at the island outputs, and employing power gating transistors to cut off both dynamic power and leakage. The benefits and overheads associated with these approaches in the context of logic isolation are discussed in Section III-B.

The key question that arises in the proposed approach is how the isolated logic islands are identified in the circuit. Clearly, considering all possible subsets of gates in the circuit leads to a large design space. To explore this search space in a structured manner, we divide the circuit into fan-out free cones, which are considered as candidates for isolation. We propose heuristics to decide which of these candidates should be isolated for each given quality level.

B. Efficiency Gains and Overheads

We now discuss the improvements in efficiency enabled by logic isolation and the overheads involved in its implementation. When a portion of logic is isolated, it results in dynamic power benefits within the island as all its constituent gates do not switch. Also, since the island outputs are held to a constant value, dynamic power is additionally saved in its downstream logic. Further, if power-gating is employed to isolate the island, then leakage power consumed by the island logic is also virtually eliminated. The power overheads primarily stem from the additional logic that needs to be embedded in the circuit to dynamically activate or isolate the island. This can be easily quantified based on the type of the cells used for isolation (*e.g.* latches, AND/OR gates *etc.*) and the number of such cells used. The number of isolation cells required to isolate an island equals the number of its inputs and outputs.

The overheads involved in isolating an island can be controlled, as shown in Figure 2. Consider an internal signal in the circuit and the fanout free cone of logic that drives it. Typically, as the depth of the fanout free cone increases, the number of inputs to the cone, and consequently the number of cells required to isolate the cone also increases. However, a key observation is that we are not constrained to isolate the entire fanout free cone; a subset of logic within the cone can also be considered an island. In the most trivial case, the island will just be a single wire (in this case, power savings would only be in the downstream logic). As shown in Figure 2, as more levels of logic are added, the size of the island progressively grows, and the power benefits and overheads correspondingly vary. The proposed methodology analyzes this trade-off for each internal signal in the circuit, and identifies the most power-efficient island to isolate. It is worth noting that choosing a subset of the fanout free cone does not impact quality, as the island output is still set to same predefined value.

In the case of power gating, each island requires only a single isolation cell at its output. However, each island additionally requires a power gating transistor to dynamically turn on/off its power supply. It is worth noting that the switching power dissipated in the power gating transistor is amortized over a large number of execution cycles, as we expect the quality control inputs to be modulated in a coarse-grained fashion in practical designs. In addition to the overheads mentioned above, the quality control unit that selects the islands that should be active for a given quality level also consumes additional power depending on the number of quality levels supported and the number of logic island used. Our design methodology considers all the aforementioned sources of power improvements and penalties, and maximizes the power benefits for each quality level.

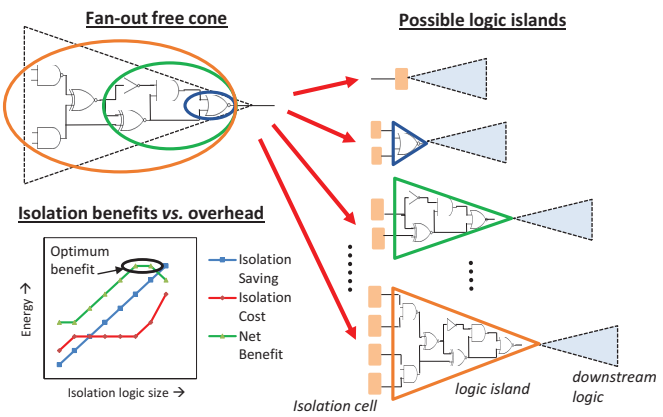

Fig. 2. Power gains and overheads associated with logic isolation

C. Error Compensation

A key aspect of approximation through logic isolation is that in some scenarios when multiple logic islands are isolated simultaneously, they can potentially cancel each others' errors, leading to an improvement in the overall circuit quality. On the other hand, the power benefits derived from simultaneously isolating multiple islands are typically additive. Therefore, in effect, error compensation leads to improved power vs. quality tradeoffs. We illustrate error compensation in Figure 3 using a cone of logic extracted from an 8-bit array multiplier circuit. After isolating signal $n38$ in the circuit, the error probability (EP) at the output bit MUL[3] is 0.406. However, if signal $n35$ is also simultaneously isolated, the error probability drops to 0.125. We can also consider the numerical significance of the shown output bit to compute the average error for the

array multiplier. The average error reduces from 0.0049% to 0.0015% when $n38$ alone is isolated and both $n38$ and $n35$ are isolated, respectively. The reason behind such compensations is that, a new isolation, apart from creating errors for new input vectors, can also undo errors for some earlier erroneous input vectors.

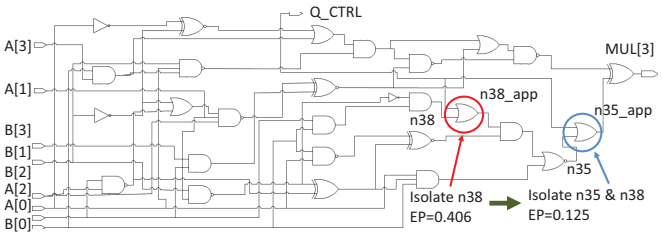

Fig. 3. Error compensation: Illustration

In our methodology, we explicitly prioritize error compensation, *i.e.*, after deciding to isolate an island, we actively identify other islands that lead to error compensation. This allows us to achieve a superior energy *vs.* quality tradeoff.

IV. Design Methodology

In this section, we describe the systematic methodology employed to design quality configurable circuits using logic isolation. Algorithm 1 describes the pseudo code of the overall procedure. The algorithm takes an original circuit (Ckt_{orig}) and a sorted list of quality modes (Q_{list}) as inputs and generates a quality configurable version (Ckt_{qc}) that can operate at all quality levels listed in (Q_{list}). First, a list of signals (Sig_{list}) in Ckt_{orig} is obtained (line 2). Then, for each signal, the fanout free cone rooted at the signal that yields the best improvement in energy when isolated is found (line 3). The best fanout free cone is derived by evaluating the power benefits from isolation as well as the isolation overheads. The list of candidates for logic isolation LI_{list} is thus populated. Next, the logic needed to interpret the quality inputs and generate the necessary control signals for isolation is added to Ckt_{orig} (line 4).

Algorithm 1 QC-Ckts design using logic isolation

Input: Ckt_{orig}: Original circuit, Q_{list}: Sorted quality list
Output: Ckt_{qc}: Quality configurable circuit

1: Begin
2: Sig_{list} = get_all_signals (Ckt_{orig})
3: LI_{list} = {get_best_fanout_free_cone (S) $\forall$ $S \in Sig_{list}$}
4: Ckt_{new} = add_quality_control_unit (Ckt_{orig})
5: **for** each Q in Q_{list} **do**
6: **do**
7: Ckt_{qc} = Ckt_{new}
8: LI_{top} = get_best_candidate (LI_{list},Ckt_{qc})
9: Ckt_{new}= isolate_logic(LI_{top},Q,Ckt_{qc})
10: LI_{list} = remove LI_{top} and overlapping candidates
11: **while** (quality(Ckt_{new}) $> Q$ && $|LI_{list}| > 0$)
12: **end for**
13: return Ckt_{qc}
14: End

Now the task boils down to identifying which candidates in LI_{list} should be isolated for each quality level. To this end, we adopt an iterative process (lines 5-12), where in each iteration we commit one logic island for isolation. For a given quality level, we identify the best candidate (LI_{top}) in LI_{list}, *i.e.*, the candidate that yields the best benefits while introducing the least error (line 8). The exact procedure employed to identify the isolation candidate is described in Algorithm 2. Then, the logic required to dynamically isolate LI_{top} is added to the circuit (line 9). Subsequently, LI_{top} and other isolation candidates that overlap with LI_{top} are removed from LI_{list} (line 10). The isolation candidate LI_{top} is committed if it meets the desired quality level. If not, the process (lines 6-11) is repeated for the next quality level, until Q_{list} is exhausted. Finally, the quality configurable circuit (Ckt_{qc}) is returned at the output.

We now present the procedure used to identify the best candidate for isolation in Algorithm 2. The algorithm takes a circuit (which may contain previously committed logic islands) and a list of possible isolation candidates (LI_{list}) and returns the best candidate in the list (LI_{top}). The best candidate can either be a compensation candidate, *i.e.*, one that decreases the overall error in the circuit, or an approximation candidate, which improves power while introducing a small error. For each candidate (LI) in LI_{list}, we embed logic necessary for isolation (line 4) and compute the net power savings (ΔP) and quality difference (ΔQ). We also compute the net efficiency gain (NEG) for the candidate as the ratio of ΔP to ΔQ. Now if the quality difference is positive, then LI is a compensation candidate, else it is an approximation candidate. Correspondingly, the best compensation (CLI_{top}) and approximation candidates (ALI_{top}) are identified and recorded (lines 8-13). The algorithm prioritizes compensation candidates over approximation candidates. Accordingly, if compensation candidates exist, then CLI_{top} is returned at the output. If not, the best approximation candidate is returned (line 14).

Algorithm 2 Select isolation candidate

Input: Ckt: Circuit, LI_{list}: List of isolation candidates
Output: LI_{top}: Best isolation candidate

1: Begin
2: $QCOMP_{top}$ = 0; NEG_{top} = 0
3: **for** each LI in LI_{list} **do**
4: Ckt_{new} = insert_isolation_logic (Ckt, LI)
5: ΔP = est_power (Ckt_{new}) - est_power (Ckt)
6: ΔQ = est_quality (Ckt_{new}) - est_quality (Ckt)
7: NEG = $|\Delta P|$ / $|\Delta Q|$
8: **if** ($\Delta Q > QCOMP_{top}$ && $\Delta P > \delta$) **then**
9: $QCOMP_{top}$ = ΔQ; CLI_{top} = LI
10: **end if**
11: **if** ($\Delta Q < 0$ && $NEG > NEG_{top}$ && $\Delta P > \delta$) **then**
12: NEG_{top} = NEG; ALI_{top} = LI
13: **end if**
14: **end for**
15: LI_{top} = ($QCOMP_{top} \neq 0$) ? CLI_{top} : ALI_{top}
16: return LI_{top}
17: End

In summary, the algorithms described in this section systematically generate quality configurable circuits that support multiple quality levels using approximation through logic isolation.

V. Experimental Methodology

Benchmark circuits. We evaluated the proposed design approach on a range of circuits: (i) arithmetic components, *viz.* array Multiplier (MUL), Carry lookahead adder (CLA), and Kogge stone adder (KSA), (ii) complex modules, *viz.* Multiple and Accumulate (MAC), Sum of Absolute difference (SAD) and Euclidean distance (EU_DIST), and (iii) complete datapaths, *viz.* Discrete cosine Transform (DCT), Fast Fourier Transform (FFT) and Finite Impulse Response (FIR) filter. The

key characteristics of these circuits are summarized in Table I. The benchmarks were synthesized and mapped to IBM's 45nm technology library using Synopsys Design compiler. The technology library was equipped with the cells required for logic isolation.

TABLE I
BENCHMARK CIRCUITS USED IN OUR EXPERIMENTS

Circuit	CLA	DCT	EU DIST	FFT	FIR	KSA	MAC	MUL	SAD
Gate Count	227	592	651	614	1310	277	836	770	800
Bit Width	32	8	8	8	8	32	8	12	24
I/O	64/33	32/36	16/24	32/32	64/20	64/33	16/24	24/24	48/24

Power and quality evaluation. Synopsys Power compiler was used to estimate the circuit's power consumption. Both the original circuit and its quality configurable version were synthesized using the same delay constraints. For 6 of the 9 circuits, there was no impact on the lowest clock period at which the circuit could be synthesized, indicating that the critical paths were not affected. For the remaining 3 circuits (MUL, EU_DIST, FFT) the lowest clock period increased marginally. However, Design Compiler's synthesis optimizations (up-sized or lower V_T cells) recover the impact on timing at the cost of area/power. These overheads are reflected in the results that we report. While we did not explicitly avoid critical paths when choosing isolation regions, it is straightforward to add this constraint to our methodology and thereby ensure no impact on timing. We utilized average error, *i.e.*, the average difference in magnitude between the outputs produced by the original and quality configurable circuits, as our quality metric. As shown in Equation 1, the average error is computed using the probabilities of the original circuit outputs (O_{Orig}) differing from the quality configurable circuit outputs ($O_{QC-Ckts}$). A weighted summation of these probabilities based on their numerical significance gives the average error.

$$\text{Average Error} = \frac{\sum_{k=0}^{n-1} P(O_{Orig_k} \text{XOR } O_{QC-Ckts_k}) * 2^k}{2^n} \quad (1)$$

VI. RESULTS

In this section, we present results that quantify the benefits of quality configurable circuits designed using our approach.

A. Power Benefits

First, we demonstrate the power benefits of approximation through logic isolation by synthesizing quality configurable versions of all benchmark circuits. The synthesized circuits support 3 quality levels *viz.* accurate, <0.1% average error, and <0.2% average error. We utilize muxes and other two input gates to isolate logic in the approximate quality modes. Figure 4 shows the power consumed by the benchmarks in each quality mode, normalized to the original circuits' power. We observe power benefits between 7.5%-27% (average of 18%), and 15%-47% (average of 30%) for quality modes of <0.1% and <0.2% average error, respectively. Quality configurable circuits consume higher power compared to the original circuit in the accurate mode of operation, as none of the logic is isolated, while isolation cells and the logic required for quality regulation continue to add overheads. These overheads amount to ∼6% of the overall power. The area overheads incurred in the quality configurable circuits over the baselines are also shown in Figure 4(below the horizontal axis labels). On average, the area overhead amounts to 6.1%. In summary, quality configurable circuits yield substantial power benefits even for very tight quality constraints, but incur a small area overhead and a small power overhead in the exact mode of operation.

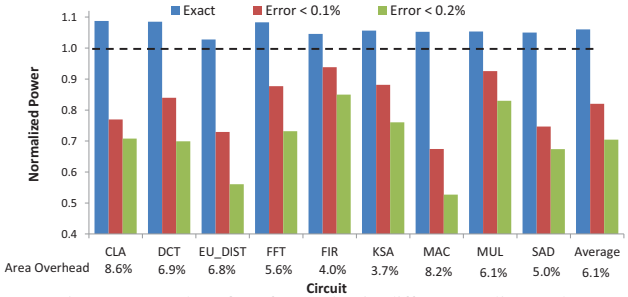

Fig. 4. Power benefits of QC-Ckts in different quality modes

We now present the power benefits when power gating is employed to isolate logic. In this case, we expect the benefits to improve as leakage power in the isolated logic is saved, but at the cost of additional power switches. In accordance, as shown in Figure 5, we achieve benefits of 8.4%-34.5% (average 23%) and 17%-51.5% (average 35%) for average errors of <0.1% and <0.2% respectively. However, due to the additional power switches required to realize power gating, the overheads in power and area in the accurate mode is 7.6% and 8.7%, respectively.

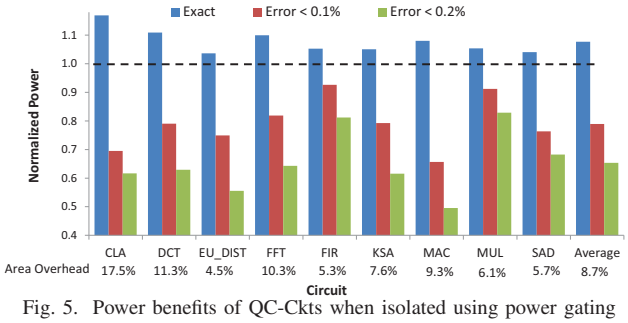

Fig. 5. Power benefits of QC-Ckts when isolated using power gating

B. Quality-Power trade-off curve

To demonstrate the ability of our approach to support various quality constraints, we build quality configurable circuits with 2 quality modes—an accurate mode and an approximate mode. We vary the quality constraint for the approximate mode, and present the resulting power *vs.* quality tradeoff in Figure 6. We observe that as the quality constraints are relaxed, power consumed in the approximate mode decreases monotonically, underscoring the ability of our approach to provide a fine-grained quality tradeoff.

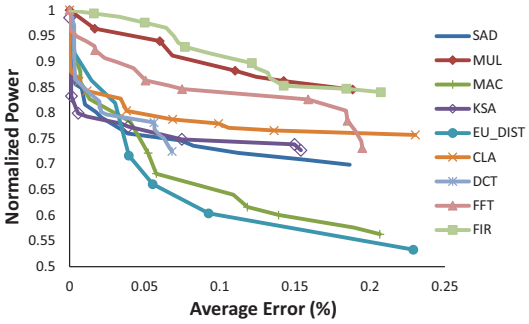

Fig. 6. Quality-Power trade-off curves for benchmark circuits

C. Comparison with dynamic precision scaling

Next, we compare quality configurable circuits generated using logic isolation with dynamic precision scaling, in which the precisions of individual operands are varied to achieve different output accuracies. The power *vs.* quality trade-offs

for two different benchmarks are presented in Figure 7. We find that logic isolation based approximation yields a strictly superior trade-off at all quality levels in both circuits. This is not surprising, as precision scaling could be viewed as a special case of logic isolation, when the isolation candidates (points where switching activity is suppressed) are restricted to the primary inputs of the circuit.

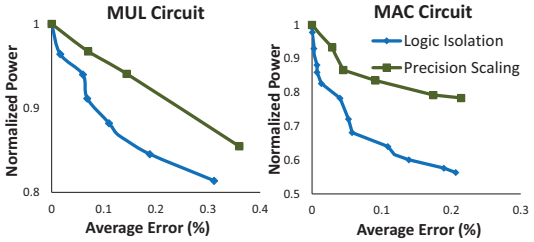

Fig. 7. Comparison with dynamic precision scaling

D. Error Compensation

We demonstrate the impact of error compensation on the efficiency of the quality configurable circuits synthesized using our methodology. To this end, we design quality configurable circuits using 2 different strategies, one which is oblivious to error compensation, and the other in which error compensation is explicitly prioritized. Figure 8 (graph on the left) shows the power *vs.* quality trade-off achieved in both the cases for a quality-configurable circuit with 2 quality modes. We find that prioritizing error compensation yields superior benefits at all quality levels. To provide more insights, Figure 8 (graph on the right) shows the power reduction and error introduced at the end of each iteration of the methodology when error compensation is prioritized. We find that in several iterations (circled) the error decreases when additional islands are isolated, due to error compensation. However, power still decreases monotonically. This is a favourable trade-off, as it provides additional opportunities for approximation for a given quality constraint.

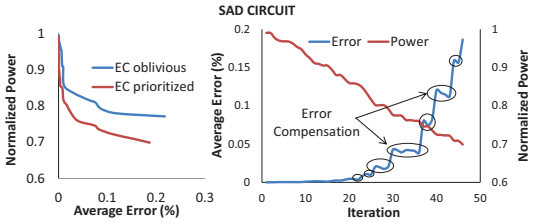

Fig. 8. Benefits of prioritizing error compensation

E. Application Level Energy Benefits

Finally, we demonstrate the benefits of quality configurable circuits at the application level, by replacing the compute units of the approximate vector accelerator presented in [4]. We utilize a deep neural network with 50000 neurons [20], proposed for handwriting recognition, as our application benchmark. In the baseline implementation, multiplication operations account for 72% of the overall application energy. Therefore, we synthesize a quality configurable multiplier with 4 quality modes *viz.* accurate, <1%, <6%, <25%. The quality modes were determined by analyzing the resilience of the multiplication operations in the application. The energy benefits obtained at the application level are shown in Figure 9. We find that quality configurable circuits yield ∼25% improvement in application energy for negligible loss (<0.1%) in classification accuracy. The benefits increase to 29.5% and 34.5% when <1% and <2% classification accuracy loss can be tolerated, respectively. This exercise suggests that quality configurable circuits generated using our approach can be easily integrated into larger designs, and yield significant benefits at the application level.

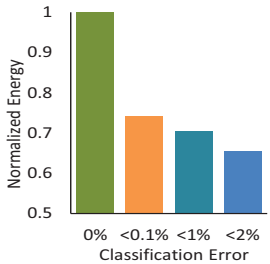

Fig. 9. Application-level energy savings using QC-Ckts

VII. Conclusion

Approximate computing leverages the ability of many applications to tolerate errors in selected computations to improve the performance/energy of computing platforms. A major challenge in approximate computing is that the degree to which computations need to approximated varies at runtime due to several factors. To address this challenge, we propose a systematic approach based on approximation through logic isolation, to design quality configurable circuits, whose output accuracy and energy can be reconfigured at runtime. The key idea is to identify islands of logic in a quality-aware manner, and dynamically isolate a subset of them during circuit operation based on the output quality desired. A key feature of our approach is error compensation, in which errors originating from multiple isolated logic portions mutually cancel each other, leading to better energy savings for a given quality. We evaluate our approach on a range of benchmarks and demonstrate the benefits of quality configurable circuits.

Acknowlegdement: This work was supported by the National Science Foundation under grants 1018621 and 1423290.

References

[1] P. Dubey. Recognition, mining and synthesis moves computers to the era of tera. *Intel Tech. Magazine*, 9(2):1–10, Feb. 2005.
[2] M. A. Breuer. Multi-media applications and imprecise computation. In *Proc. Euromicro Conf. on Digital System Design*, pages 2–7, 2005.
[3] S. Chakradhar et. al. Best-effort computing: Re-thinking parallel software and hardware. In *Proc. DAC*, June 2010.
[4] S. Venkataramani et. al. Quality programmable vector processors for approximate computing. In *Proc. MICRO*, Dec. 2013.
[5] V. Gupta et. al. IMPACT: Imprecise adders for low-power approximate computing. In *Proc. ISLPED*, pages 409–414, Aug 2011.
[6] P. Kulkarni et. al. Trading accuracy for power with an underdesigned multiplier architecture. In *VLSI Design*, pages 346–351, Jan. 2011.
[7] A. Lingamneni et. al. Energy parsimonious circuit design through probabilistic pruning. In *Proc. DATE*, pages 1–6, 2011.
[8] D. Shin and S. K. Gupta. A new circuit simplification method for error tolerant applications. In *Proc. DATE*, Mar. 2011.
[9] D. Shin and S. K. Gupta. A re-design technique for datapath modules in error tolerant applications. In *Proc. ATS*, pages 431 –437, Nov. 2008.
[10] N. Olivieri et. al. Analysis and implementation of a novel leading zero anticipation algorithm for floating-point arithmetic units. *Circuits and Systems II: Express Briefs, IEEE Trans. on*, 54(8):685–689, 2007.
[11] S. Venkataramani et. al. SALSA: Systematic logic synthesis of approximate circuits. In *Proc. DAC*, 2012.
[12] D. Shin and S. K. Gupta. Approximate logic synthesis for error tolerant applications. In *Proc. DATE*, pages 957–960, Mar. 2010.
[13] S. Venkataramani et. al. Substitute-and-simplify: A unified design paradigm for approximate and quality configurable circuits. In *Proc. DATE*, pages 1367–1372, 2013.
[14] V. K. Chippa et. al. Analysis and characterization of inherent application resilience for approximate computing. In *Proc. DAC*, 2013.
[15] A.B. Kahng and S. Kang. Accuracy-configurable adder for approximate arithmetic designs. In *Proc. DAC*, pages 820–825, 2012.
[16] A. B. Kahng et. al. Slack redistribution for graceful degradation under voltage overscaling. In *Proc. ASP-DAC*, pages 825–831, Jan. 2010.
[17] L. Wan et. al. CCP: Common case promotion for improved timing error resilience with energy efficiency. In *Proc. ISLPED*, 2012.
[18] S. Ramasubramanian et. al. Relax-and-retime: A methodology for energy-efficient recovery based design. In *Proc. DAC*, 2013.
[19] Albicocco et al. Truncated multipliers through power-gating for degrading precision arithmetic. In *Proc. Asilomar Conf. on Signals, Systems and Computers*, pages 2172–2176, Nov 2013.
[20] Y. Lecun et. al. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11), Nov 1998.