Evolutionary Design of Approximate Circuits

Vimal Raj Krishna Raj
College of Engineering and Computer Science
University of Central Florida
Orlando, USA
Email:vimalraj@knights.ucf.edu

***Abstract*—This paper deals with the analysis of evolutionary design in different approximate circuit designs. The key character of these designs is to relax the functionality of the circuits to minimize energy consumption or area of the chip.**

Keywords—Approximate design, Cartesian genetic programming, gates scaling.

I. INTRODUCTION

Evolvable hardware (EHW) is usually defined as a non-traditional (unconventional) search-based method for hardware design and adaptation that uses various bio-inspired computational intelligence paradigms. The main reasons why EHW has mainly been studied and developed include its ability to (i) provide novel designs hardly reachable by means of conventional methods, (ii) deliver good solutions for problems whether the specification is inherently incomplete and any golden solution does not exist, and (iii) achieve adaptation/fault tolerance directly at the hardware level. [1] Approximate computing is a concept where the functionality of a circuit is compromised to optimize design constraints like chip area, energy consumption. It is found that few common lines can be drawn between approximate computing and evolutionary design concepts. In this paper we are going to see the different approaches carried out by various researchers in implementing approximate circuits using evolutionary design methodology to specific hardware. The main design objectives focused by these approaches are optimizing area, energy consumption, delay and Electrostatic Discharge (EDS) immunity. We will analyze how effectively these approaches have optimized these objectives and validate its worth to different types of circuit. This will draw a boundary between the scalability and acceptable functionality of the approximated circuit and hence motivate research to push the boundaries.

The flow of paper is depicted as follows. In the next section we will discuss some of the previous research done on lines of our goal. In Section III the design objective and methodology used by each approach is discussed. The experimental results obtained through these approaches are given in Section IV. Section V identifies the best approach that could be used for effective use of EHW in approximation techniques. Future work is given in Section VI and Section VII gives the conclusion to this paper.

II. RELATED WORK

In this section we will mention few related work done in approximate circuits as well as evolutionary design concepts. This will help us understand how both of these concepts are combined to give techniques which we will be discussing in the following sections .

Approximate circuits deal with the relaxation of functionality to improve computations. There are two types of techniques used in approximate circuits namely Over scaling and Functional approximation . The former involves techniques like voltage over scaling to reduce energy consumption. Also, by over-clocking the performance is seen to be increased significantly. However, they lead to many timing induced errors.

In the other type, functional approximation does not fully implement the logic that has been specified before design. This can be done by ignoring the least significant bits. An alternative method was proposed in [6] where the specifications were met but by implementing a logic synthesis technique the required amount of resources were reduced. By doing this they were able to achieve a power savings of 30%-50%. There was also another interesting approach for systematic synthesis called SALSA [9] which is fully automated technique to generate approximate circuits. The method used something called Q-function which evaluates output from the original and approximate circuits and sets a satisfaction value. While modifying the approximate circuit it is made sure that this value is not changed. These types of design methodologies can be widely used in image processing and media applications through approximating adders and multipliers. It has to be stressed that traditional design and verification techniques are not directly applicable for approximate circuits [1].

While focusing on purely evolutionary circuit design we can still see some relevance to approximate computing. The first and foremost research done on this front was by Thompson [11] where he said a genetic Algorithm could evolve a useful digital circuit without any knowledge of digital design techniques. He demonstrated intrinsic evolution of a tone discriminator on a Xilinx 6200-series FPGA. The circuit functionality started near zero and improved to near-perfect within a few thousand generations. This can be accounted as a type of approximate computing. In other research area, Miller [12] has introduced a technique called

Cartesian Genetic Programming (CGP) which used a searching strategy in the form of a grid containing nodes which represent a specific function. The search uses (1+**λ)** evolution strategy to select certain number of nodes and such a selected pattern of nodes form a functional circuit which is approximate in nature. For example Miller had used CGP based method for finite impulse response (FIR) filter design [13]. The target filters were composed of only elementary gates which hence ignored the conventional methods like multiply and accumulate structures. Apparently, this is done to minimize the area in the design by obtaining partial functionality which can be afforded to compromise.

Thus in EHW community these techniques based on inherent fault tolerance which means original or desired functionality is achieved in case of failures using evolutionary approaches, seems to be similar to the idea behind approximate computing. By this inference we can say that EHW principle could be used in designing approximate circuits provided the scalability problem can be eliminated for a particular application. To better understand the fusion between these two principles , a context diagram is given below Fig1.a where the smaller gears drive the bigger one and the only constraint that acts as friction to this process is the scalability factor.

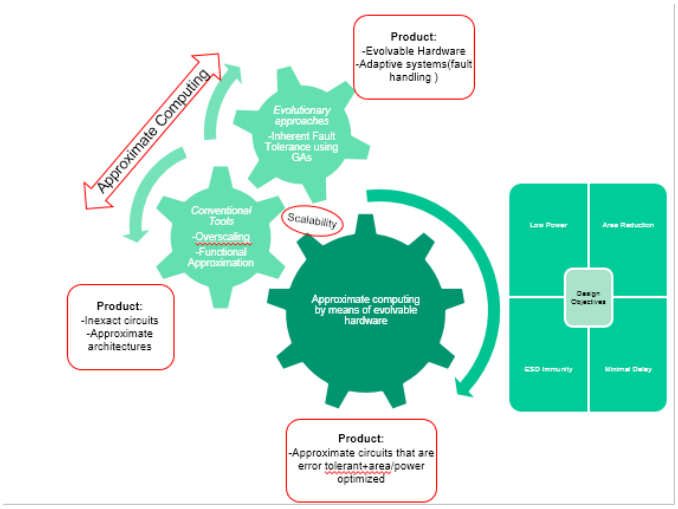


Fig1. Approximate Design Using Evolvable Hardware

III METHODOLOGY

The base technique used by all the approaches that we are going to discuss is CGP. According to [12] we will overview the concept behind CGP and move on to the specific strategies used by our approaches listed by the respective subsections.

A. *Cartesian Genetic Programming*

CGP is a representation of a list of integers (functions) that are mapped to directed graphs rather than trees. Graphs are used rather than trees because it is more general in form. It is basically a genotype-phenotype mapping where the former is of fixed length and the latter have variable length according to number of unexpressed genes. A Cartesian program (CP) denoted P is defined as a set {G, n_i, n_o,F, n_f, n_r, n_c, *l*} where G represents the genotype and is itself a set of integers representing the indexed n_i program inputs, the n_n node input connections and functions, and the no program output connections. The set F represents the n_f functions of the nodes. The number of nodes in a row and column are given by nr, nc respectively.[12]A general representation of CGP is given in Fig1.

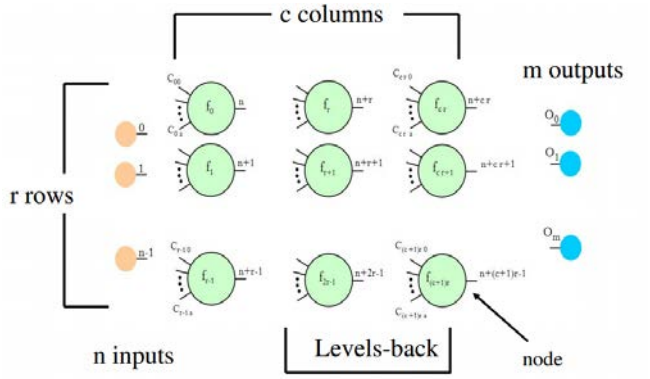


Fig2. General representation of CGP [17]

The thumb rule is that the nodes in the same column cannot be connected with each other. The following stages in circuit evolution describes how CGP is implemented,

1) Encoding: Utilizes a set of processing nodes arranged in c columns and r rows to denote a target circuit. Each node is assigned one function . A specific interconnection of the elements in the grid results in a functional circuit .

2) Search: Evolution strategy is used for searching in the CGP search space. Usually a (1+ λ) evolution strategy[12] is followed. At the beginning 1+λrandomly generated solutions are created to form the initial population.The quality of solutions is then evaluated using the fitness function. The best solution (the parent) is identified. New λ solutions are created using mutation operator from the parent. If the termination condition is not fulfilled, then step 2 is taken, otherwise, the fittest solution is the result of CGP.[3]

3)Fitness Function: For the evolutionary design of digital circuits the fitness value of a candidate circuit is usually defined as

$$fit\ 1 = b \qquad (1)$$

where b is the number of correct output bits. The goal is to minimize hamming distance between obtained truth table the the real truth table .Suppose this fitness function has to evaluate multiple output circuits then the goal would be to minimize the mean absolute error between a candidate circuit response *y* and target response *t*. The new fitness function defined would be :

$$fit2 = \sum_{j=1}^{k} |y(j) - t(j)| \quad (2)$$

k will be the number of fitness cases.

B. CGP based Approximation Design

This approach according to [1], constructs a set of circuits that target functionality using constrained resources and then tested for functionality violation. It mainly focuses on trading off accuracy with the chip area and thus energy consumption. On combinational single output circuits eqn1 was used and for combinational multiple output circuits eqn2 was used as the fitness function. The CGP based search method discussed in earlier sub-section is employed. The solution showing the best trade off between the number of gates and measured error is taken as the final solution. Suppose the number of gates represents area of the circuit, design methodology evolves a target circuit *C* to find out the minimum number of gates *n* required for its implementation. The error function is minimized by the CGP and the evolution stops when the condition is satisfied. Similarly this is repeated for more circuits and approximate circuits are designed with decrementing the number of gates supplied accordingly. Finally the circuit showing the best trade off between accuracy and number of gates is chosen by the user. One must note that here they have contrasted power consumption with the number of gates.

C. Multiobjective CGP based Approximation

In this paper [3], a rather different version of CGP technique is used to approximate Multiple Constant Multiplier(MCM) circuits. MCM is a digital circuit which multiplies its single input by N constants.It has no multipliers but adders , subtractors and shifters hence it's cheap and is used in low power FIR filters. The means of approximation of MCMs is done by multiplying the input by slightly different constants than the specification requires which leads to the delay and/or component reduction in comparison with the exact solution.

MCM design is multiobjective which means that we optimize a circuit with respect to more objectives.If the optimized objectives are conflicting, the goal of multiobjective optimization is find many trade-offs among these objectives.
A multiobjective CGP problem is defined as,

$$\begin{array}{rll} \text{minimize/maximize} & f_m(x), & m = 1, 2, ..., M \\ \text{subject to} & g_j(x) \geq 0 & j = 1, 2, ..., J \\ & h_k(x) = 0 & k = 1, 2, ..., K \end{array}$$

where fi are optimized functions. The inequity constraints are defined by functions g_j and equity constraints are defined by functions h_k.
The goal of multiobjective optimization is to find Pareto-Optimal solutions. The new version of fitness function reflects accuracy and is given by,

$$f_{err} = \sum_{i=1}^{N} (y_i - y_i')^2. \quad (3)$$

In order to optimize the number of components (f_1),adders(f_2) and delay (f_3), four fitness functions to optimize: f_1 , f_2 ,f_3 *and* f_{err} are obtained.

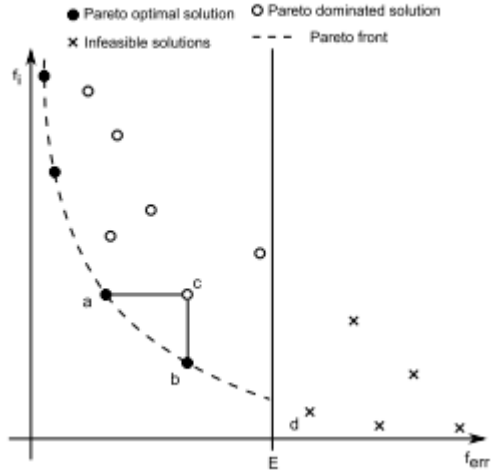


Fig2. Relation between fitness function and acceptable threshold.[3]
As shown in Fig2. a threshold value *E* has been set , in order to keep a check at the accuracy level not to go beyond an accepatable level determined by the user. It should be observed that the new fitness function in other words, can be said as the difference against desired outpit values.The vertical axis is f_i where *i* can be 1,2 or 3 which needs to me minimized. NSGA-II algorithm[18] has been used with controoled ellitism to achieve this goal.

D. Modified CGP based Approximation

Multiobjective optimization have been used on very small problem instances only. Scalability is an issue and hence smart algorithm must be developed to avoid unacceptable computational times. Hence CGP is employed where the fitness function is based on a formal equivalence checking algorithm rather than evaluating all possible input assignment because fitness evaluation is the most time consuming part. Also a modified selection strategy is used where the fitness value of a candidate digital circuit is defined as,

$$fit1 = \begin{cases} b & when \quad b < n_o 2^{n_i}, \\ b + (n_c n_r - z) & otherwise, \end{cases} \quad (4)$$

Here *b* is the number of correct output bits and z is the number of gates utilized in a particular candidate circuit and the product of *n*'s are the total available number of gates.

The key point here is that unlike usual fitness criteria , the best individual evolved and the parent individual need not necessarily be the same. So the new parent must be a fully functional circuit but number of gates is not important for its selection.
Another modification made in this CGP is the use of Fast-Fitness evaluation technique to reduce the computation time

taken for assessing fitness of the population. Conventionally the search algorithms uses a formal equivalence checking which starts from scratch. Fig3 explains the conventional equivalence approach of two combinational circuits.
FigF

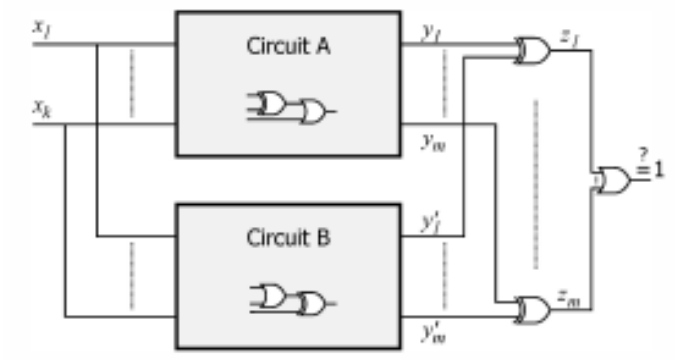


Fig3. Equivalence checking of two combinational circuits[4]

Now instead of applying all the assignments to the input , the new candidates are comapred with a reference candiadte and the functionally incorrect circuits are discarded. Otherwise, the number of gates in that functionally correct circuit is given by its fitness value.As the complexity of circuits (more number of gates) increase the comparison time till will be beyond affordable limits. This has motivated research in looking for a smarter technique which ended up in finding a Satisfiability (SAT) solver based equivalence checking. This can be explained by considering the same example in Fig3. Circuit A and Circuit B are compared using XOR gates and an OR gate which is also called a miter (a difference circuit). So this difference circuit along with the the two circuits A and B are converted to a boolean formula in the form of a conjunctive normal form(CNF). This works in a way that when if and only if A and B are functionally equivalent the condition is unsatisfiable. Further, this CNF is converted to gate by gate reperesentation using Tseitin's algorithm.[19]
Various methods are used for further reducing the decision time used by a SAT solver by reducing the number of cluases.

Hence an optimized approach which can be called *fast SAT solver* is proposed. This utilizes the knowledge of genes which have been modified by the mutation operator to calculate a difference between the parent individual and its offspring[4] In Fig4, this is explained by observing the total number of gates in Fig4.a and Fig4.b in addition to 2 XORs and an OR gate for the *All-output* approach where as the miter circuit in Fig4.c has optimized the number of gates to almost half the toal number used in previous approach.

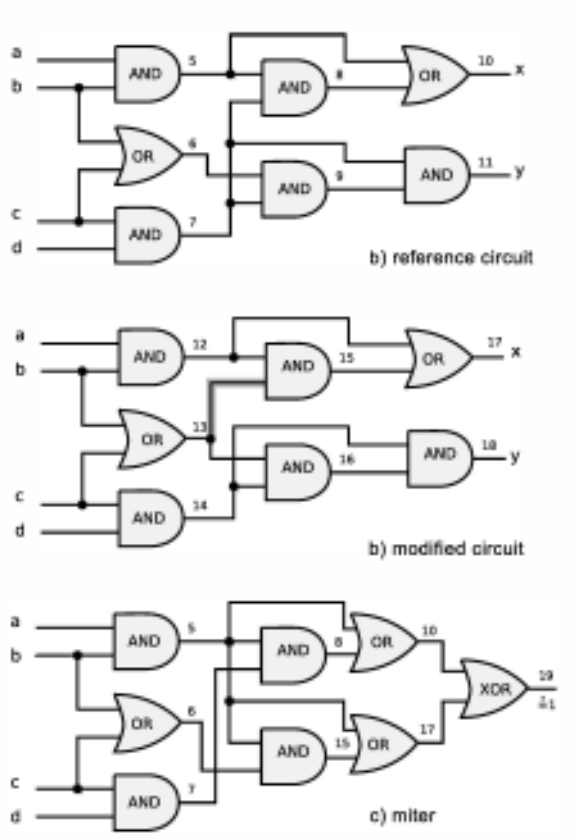


Fig4. Examples of different SAT solvers [4]

E. ESD Immune Circuit Design

ESD is nothing but an electric short, or dielectric breakdown in integrated circuits .It generates high voltage, current , transient field and can cause permanent damage in many scenarios. For example ESD current injected to I/O pins can destroy ICs and cause permanent damage. Conventional ESD protection like the one given below in Fig5 is no longer useful because of increasing parasitic effects causing degradation to internal circuit performance.

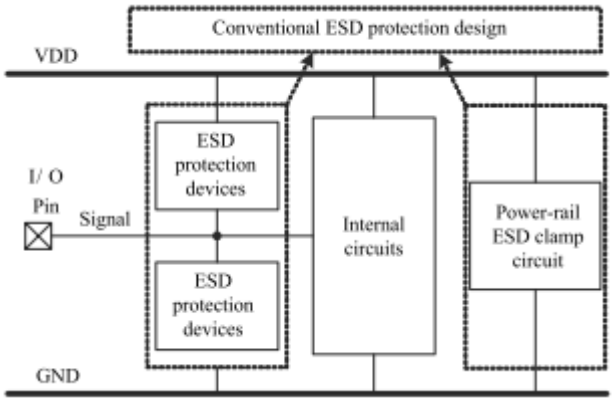


Fig5. Conventional ESD protection scheme [2]

Hence this paper tells us how an ESD immune circuit is designed using evolutionary design techniques and analyses whether degeneracy or redundancy in design is effective to a more robust circuit. The design involves two main components

1)Evolution Engine: Generates and downloads configuration bit streams to RDC and evaluates their logical functionality. It constitutes of an embedded processor and a genetic algorithm and it uses (1+λ) evolution strategy where λ is the population size.

2)Reconfigurable Digital Circuit(RDC): It is based on the traditional CGP model. The nodes placed in the grid are Multi-Function Logic Elements(MLEs) and each execute seven Boolean operations. The architecture of RDC is given below in Fig6. A group of Multi-function Logical Elements (MLEs) are placed in an array of n rows by m columns.Each MLE can execute one of seven Boolean operations. The cross-point switch arrays are used as the interconnection network to control the data flow among the MLEs. All of the MLE connections are feed-forward, and only neighboring columns can be connected. The circuit inputs can be connected to MLE inputs in the first column and the circuit outputs can be connected to MLE outputs in the last column [2].

ESD immunity (I) is the ability of a device, equipment or system to perform without degradation in the presence of an ESD disturbance. Under this definition, the ESD immunity of the evolved circuits with regard functionality against the ESD interferences can be mathematically described as

$$I = \sum_{d \in D} \psi(d) \cdot F(d),$$

D- Entire ESD interference space

Phi(d) – probability one interference 'd' to take place

F(d) is the functionality under interferecen d given by,

$$F = (1 - \frac{1}{2^n \cdot m} \cdot \sum_{j=1}^{2^n} \sum_{j=1}^{m} \left| d_{ij} - y_{ij} \right|) \cdot 100\%,$$

Where m and n- no of outputs and inputs of logic function

Yij is the digit in that respective row and column and Dij is the desired output at corresponding position .Hence it gives absolute difference and F is normalized . Redundancy and Degeneracy need to be measured in order to compare the immunity effect it has in design when tested on simple combinational circuits.

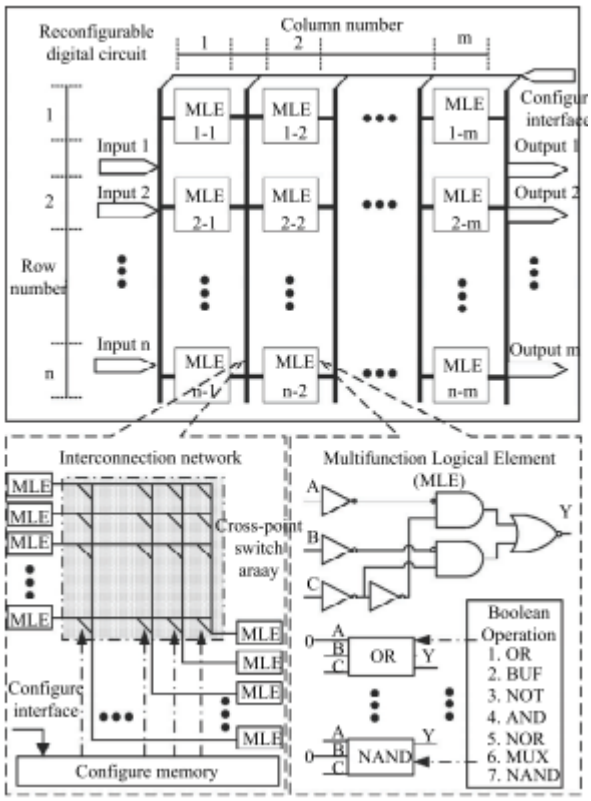


Fig6. Architecture of the Reconfigurable Digital Circuit(RDC)[2]

IV EXPERIMENTAL RESULTS

The approaches mentioned in previous section have experiments performed on various types of benchmarks. We will review the important results of each approach in its corresponding subsection.

A. CGP based Approximation Design

Experiments were done on two types of circuits,

1). Singe Output Circuits

Benchmark circuits like cm152,sym9,t481(LGSynth93 family) and 9-input majority circuit were used for the fitness runs. Table I summarizes the best evolved single output circuits.

TABLE I

BEST EVOLVED SINGLE OUTPUT CIRCUITS

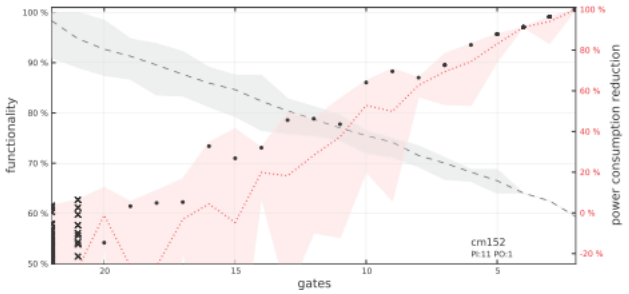
		power [uW]			area		
circ	n_g	best	worst	mean	best	worst	mean
maj9	30	170.2	259.8	216.0 ± 15.3	69.0	101.0	83.2 ± 5.0
cm152	21	143.4	460.1	235.5 ± 44.9	50.0	107.0	72.9 ± 9.0
sym9	23	432.1	1598.0	739.7 ± 179.1	115.0	165.0	138.4 ± 10.1
t481	21	197.1	871.4	303.2 ± 69.3	56.0	135.0	76.2 ± 10.0

We could see that the circuit *cm152* seems to be the best evolved approximate circuit when the mean of area is taken into concern hence the overall power reduction is the least.

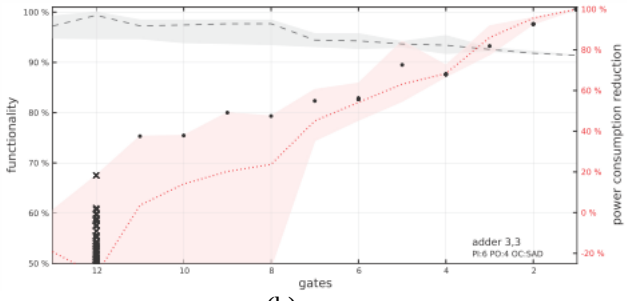
Fig4.a gives the plot of the Functionality with respect to power reduction. . The functionality tends to increase until 100% is reached with 1 gate which is the ideal case. The band in the graph is believed to be the overall trend , the line being the mean and the dots being the best.

2)Multiple output circuits

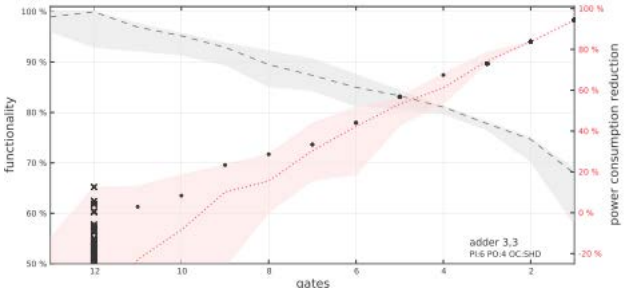
For multiple output circuits , the fitness function was to minimize the mean absolute error. Also two approaches in fitness evaluation was compared namely , SAD – Sum of absolute differences and SHD- Sum of Hamming distances in terms of computational costs and quality of reachable circuits for combinational adders (3, 3.5, 4 bits) . In Table II , the comparison between the two approaches are given. It seems that SAD turns out to better overall because SHD showed four times more mean error than SAD based fitness function. Fig7b gives SAD based plot and Fig7c SHD based plotMorover, adders evolved using SHD fitness function had errors in MSB which is not acceptable at any cost.



(a)



(b)



(C)

Fig7. Functionality with respect to power reduction [1]

TABLE II
BEST EVOLVED ADDERS UNDER SAD AND SHD - BASED FITNESS FUNCTIONS

		power [uW]			area		
circ	gates	best	worst	mean	best	worst	mean
adder 3,3 (SHD)	12	108.5	372.5	196.9 ± 35.8	41.0	73.0	55.1 ± 5.8
adder 3,4 (SHD)	14	143.5	443.7	241.6 ± 44.9	52.0	82.0	65.0 ± 5.9
adder 4,4 (SHD)	17	191.1	681.4	301.0 ± 55.3	62.0	102.0	77.6 ± 7.2
adder 3,3 (SAD)	12	118.4	339.4	193.3 ± 35.6	41.0	69.0	54.0 ± 5.8
adder 3,4 (SAD)	14	163.0	556.7	242.2 ± 50.4	54.0	81.0	64.9 ± 5.9
adder 4,4 (SAD)	17	201.5	570.0	305.8 ± 65.7	60.0	102.0	76.9 ± 7.1

B. Multiobjective CGP based Approximation

The objective is to minimise number of components, adders – subtractors and delay in the MCMs. The optimization is based on the pareto principe aka 80-20 rule where 80% of functionality is retained if 20% of the design objective is optimized. Comparison between three different 5constants MCMs Fig8. were done with different number of computational nodes. Delay is calculated as the the largest routing distance between the inputX and any of the output Yi. It is observed that the fitness value ferr=0 for the left most MCM and the right most's value is 85. But the middle one was the most optimum with 13. Also the delay was same as the left (delay=4) whereas the rightmost one had a delay of 5. Also it used a moderate number of computational node (10 nodes) where as the left most used 13 and the right most used 6 nodes only but had a greater delay .

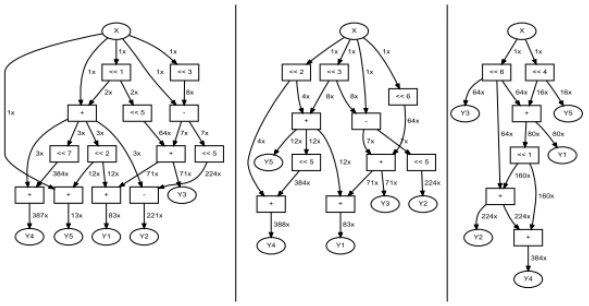
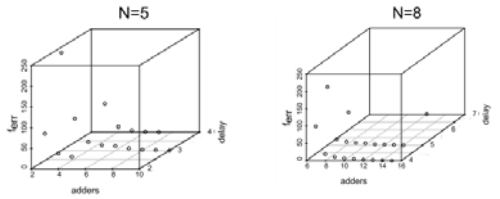
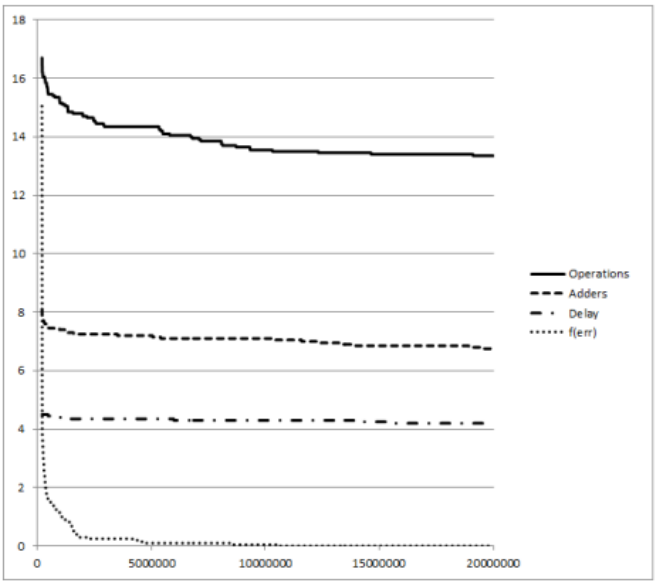


FIG8. COMPARISON OF THREE DIFFERENT MCMS

The best obtained trade-offs for MCMs Fig9a. with 5 and 8 constants have been shown and the progress of evolution for the MCM with 8 coefficients. Best results averaged from 20 independent runs are shown in Fig9b.
The independent runs are plotted against the evaluations and the evolution is observed for adders , fitness function and the delay.

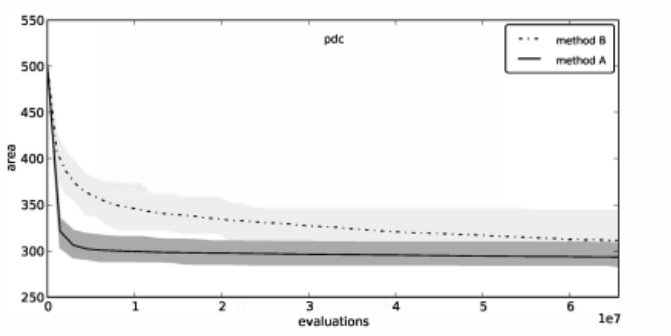


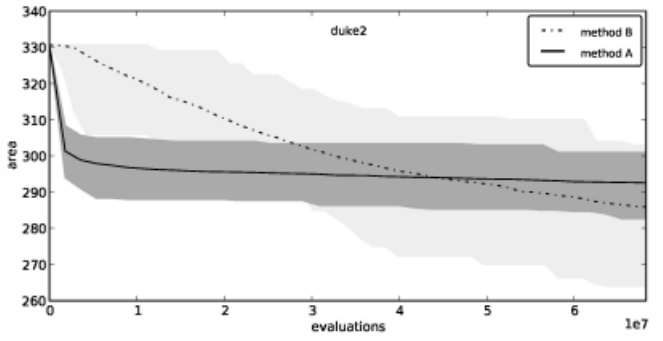
(a)

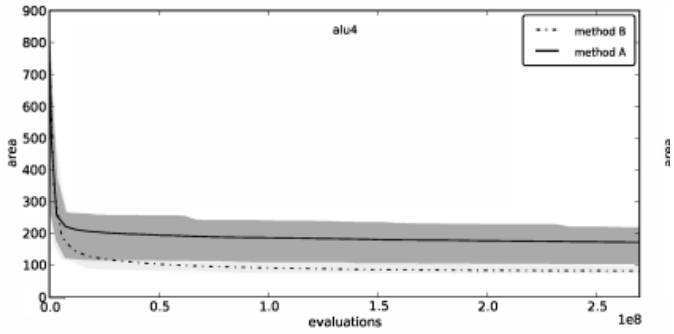


(b)

Fig9Trade off and .Progress of evolution for MCM with 8 coefficients.







C. Modified CGP based Approximation

Experiments were carried out on cluster of Intel Xenon X5670 2.4GHz processors using Sun Grid Engine that can run experiments in parallel.
Two methods have been employed for CGP optimization,
Method A- Standard CGP + Fast SAT based fitness function.
Method B-Modified CGP + Fast SAT based fitness function.

Many different complex benchmarks were experimented upon and extensice results were given based on method A and method B and also as well as using the conventional ABC tool. Fig 10 shows four selected (for diversity) convergence curves that are quite different from each other.

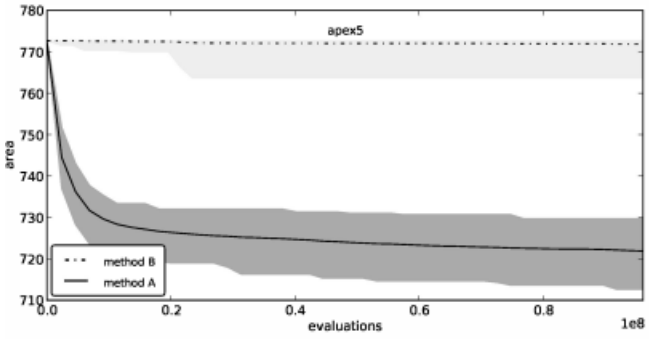


Fig10Convergence curves comparing method A and method B[4]

Table III shows overall comparison of the conventional ABC tool , method A and method B approximations over different benchmark circuits and the best and average area, delay.

TABLE III

circuit	N_i	N_o	Area	Delay
alu4	14	8	819.7	16
apex1	45	45	1761.3	14
apex2	39	3	206.3	13
apex3	54	50	1607.0	13
apex5	117	88	772.7	11
b12	15	9	61.0	6
cordic	23	2	55.0	8
cps	24	109	1021.7	12
duke2	22	29	330.7	11
e64	65	65	382.3	8
ex1010	10	10	2487.3	14
misex2	25	18	110.7	7
misex3	14	14	842.6	14
misex3c	14	14	499.3	13
pdc	16	40	503.0	13
sao2	10	4	132.3	9
spla	16	46	571.0	14
t481	16	1	25.0	5
table3	14	14	1382.3	15
table5	17	15	1068.0	14
vg2	25	8	92.3	9

circuit	BstF	BstF$_D$	BstD	BstD$_F$	AvgF	AvgDly	NDM(%)
alu4	106.7	17	13	113.7	173.1 ± 32.4	17.12 ± 1.76	68.22 ± 4.80
apex1	1410.7	19	18	1415.0	1441.5 ± 14.8	19.22 ± 0.64	9.89 ± 0.34
apex2	132.0	15	13	139.0	140.6 ± 4.3	14.30 ± 0.75	24.15 ± 1.43
apex3	1350.0	18	17	1356.3	1369.8 ± 11.6	18.40 ± 0.87	9.10 ± 0.28
apex5	712.7	13	12	715.0	721.9 ± 3.7	13.14 ± 0.77	6.78 ± 0.21
b12	51.7	8	6	53.7	53.7 ± 1.2	7.88 ± 1.19	10.58 ± 0.77
cordic	49.0	8	8	49.0	50.1 ± 0.4	8.38 ± 0.56	10.80 ± 0.53
cps	787.3	17	15	790.0	807.2 ± 9.3	16.60 ± 1.18	12.73 ± 0.38
duke2	282.7	13	12	287.0	292.5 ± 4.4	13.16 ± 0.70	10.46 ± 0.69
e64	262.7	14	11	274.3	278.4 ± 6.8	12.78 ± 1.10	18.75 ± 0.97
ex1010	2382.7	18	17	2390.7	2404.5 ± 7.7	17.64 ± 0.71	4.50 ± 0.09
misex2	85.3	10	8	92.7	95.1 ± 2.7	9.52 ± 0.88	10.68 ± 0.91
misex3	398.0	20	17	428.7	440.8 ± 16.5	19.14 ± 1.48	31.60 ± 1.44
misex3c	400.7	16	14	412.0	416.5 ± 7.0	16.46 ± 1.17	11.33 ± 0.65
pdc	282.7	14	13	295.3	293.4 ± 6.1	14.68 ± 0.84	29.82 ± 0.94
sao2	95.0	12	11	102.3	107.0 ± 3.9	12.72 ± 1.00	14.05 ± 1.46
spla	280.7	18	14	295.0	296.9 ± 7.1	16.12 ± 1.07	34.07 ± 1.03
t481	24.3	5	5	24.3	24.3 ± 0.0	5.00 ± 0.00	10.36 ± 0.00
table3	1055.3	20	18	1066.0	1081.0 ± 15.2	20.04 ± 1.04	11.76 ± 0.49
table5	756.3	18	17	799.0	789.6 ± 11.6	19.00 ± 1.02	15.17 ± 0.49
vg2	83.3	9	9	83.3	85.6 ± 1.0	10.18 ± 0.93	8.30 ± 0.37

circuit	BstF	BstF$_D$	BstD	BstD$_F$	AvgF	AvgDly	NDM(%)
alu4	78.0	14	12	83.0	83.0 ± 4.1	14.52 ± 1.32	83.41 ± 1.06
apex1	1657.0	21	19	1683.3	1693.7 ± 16.5	20.64 ± 0.95	9.18 ± 0.31
apex2	86.7	16	14	89.3	89.6 ± 1.5	16.40 ± 1.04	46.36 ± 0.61
apex3	1589.7	20	13	1603.3	1605.9 ± 3.0	13.88 ± 2.39	8.28 ± 0.22
apex5	763.7	14	11	772.7	771.8 ± 1.9	12.02 ± 1.79	9.00 ± 0.27
b12	49.0	7	6	51.3	51.4 ± 0.8	7.80 ± 1.08	13.54 ± 0.40
cordic	41.3	8	7	41.7	42.6 ± 0.6	8.72 ± 0.83	24.85 ± 0.51
cps	928.3	19	17	928.7	947.3 ± 11.2	19.78 ± 1.79	11.74 ± 0.36
duke2	264.0	18	15	292.3	285.8 ± 9.1	17.86 ± 1.70	16.72 ± 1.06
e64	237.3	41	26	240.7	245.6 ± 4.0	34.06 ± 4.05	27.82 ± 0.49
ex1010	2486.0	14	14	2486.0	2487.3 ± 0.2	14.00 ± 0.00	4.72 ± 0.08
misex2	78.3	9	8	80.7	81.5 ± 1.4	10.48 ± 1.42	21.80 ± 0.38
misex3	369.3	24	19	407.0	433.9 ± 32.6	22.58 ± 1.67	35.79 ± 2.55
misex3c	353.7	17	16	363.0	380.1 ± 11.7	19.68 ± 2.20	20.32 ± 1.40
pdc	292.3	22	17	309.0	311.5 ± 10.2	19.96 ± 1.87	32.52 ± 0.94
sao2	54.3	9	9	54.3	60.5 ± 3.4	11.60 ± 1.33	43.44 ± 2.81
spla	305.0	17	17	305.0	321.9 ± 7.3	20.62 ± 1.82	35.46 ± 0.83
t481	23.7	5	5	23.7	24.0 ± 0.2	5.00 ± 0.00	7.65 ± 0.08
table3	1290.0	21	20	1309.0	1331.3 ± 13.7	22.10 ± 1.17	10.77 ± 0.37
table5	900.3	21	19	917.0	937.2 ± 17.8	22.54 ± 1.42	14.57 ± 0.61
vg2	75.0	11	10	77.7	78.1 ± 1.7	12.52 ± 1.04	16.13 ± 1.36

It is observed that Method A and B are effective 50% of the time than the conventional method which is evident from the BstD – Best delay values. Also , the circuit alu4 is the most effective in using CGP based approximation where the delay had immensely dropped. More insights and interpretations will be derived in the coming section.

D. ESD Immune Circuit Design

Different sources were used for experiment such as 2-bit adder , 5-bit MAJORITY benchmark , C17 from North Carolina benchmark library. But no circuit has more than 20 gates because otherwise computation time goes unaffordable.

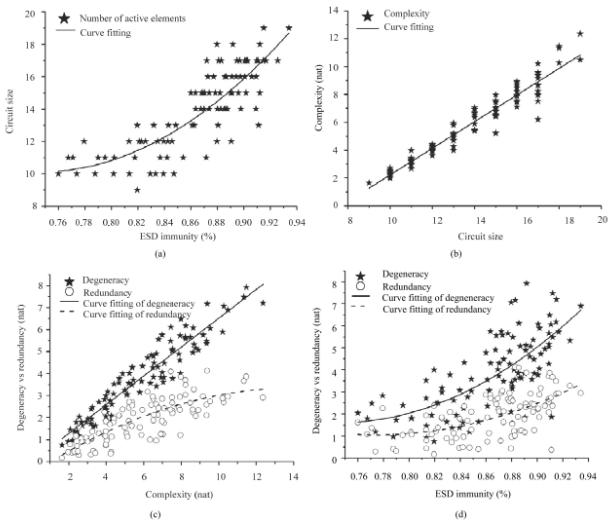


Fig11. The different topological properties of the evolved 2-bit ADDER circuits. (a) Correlations between ESD immunity and circuit size; (b) the relationship between circuit size and complexity; (c) the relationships of the redundancy and degeneracy to complexity; (d) the relationships of the redundancy and degeneracy to ESD immunity. [2]

Considering the Fig11.d we can see that degeneracy increases more rapidly which expected in evolved circuits with high immunity. Different circuits with similar redundancy have *varied levels* of immunity which is seen from Fig11.c. Also it could be noted that complexity vs degeneracy appears to linear which is a good reason to use degeneracy. Fig11a&b tells us that varying complexity in structures imply varying levels in immunity. Hence it is inferred that degeneracy is preferred mechanism in addition to redundancy elements to make the ESD immune circuit more robust and effective.

V. EFFECTIVE TECHNIQUE FOR APPROXIMATION USING EVOLVABLE HARDWARE

As discussed in the previous sections , we have seen different approaches to approximate circuits using evolutionary design techniques and interpreted the results from the experiments. I believe the modified CGP technique[4] proves to be the strongest approach since it has successfully achieved 24% of are reduction in comparison to conventional synthesis tool. Also the novelty in modification of the standard CGP approach by using a fast fitness evaluation technique where the computational time of fitness evaluation is greatly reduced with the use of *fast SAT solver* instead of the usual *all-output* approach for equivalence checking. More benefits could be interpreted by drawing a taxonomy of the approaches covered which is given in Fig12.

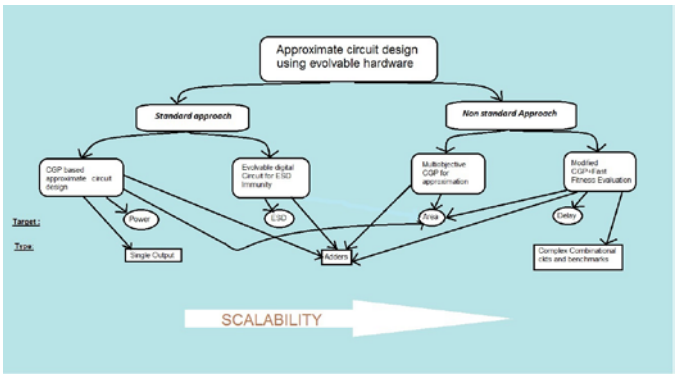


Fig12.Taxonomy of approaches covered.

Fig12 tells us that modified CGP is the only approach that accounts for delay in approximate design where others have played it safe with simple adder circuits and benchamarks. In [1] it has clearly been told that the technique will not hold good when scalability of evolvable hardware is taken into account. Similarly other approaches have declared scalablity of the circuit as their main drawback especially [2] which has declared their technique for evolution of ESD immune hardware cannot work on circuits having more than 20 gates.

However , in present scenario with the advent of digital media has tremendously increased , there is constant demand for energy conservation and chip area reduction which calls for approximation of circuits and when it is required to implemented through EHW , it is not possible to obtain good trade offs with the real time circuits in media as they are apparently complex except for few basic filters. Also the attractive option of customizing our approximation level in any circuit since we use GAs should be tapped the most out of from. Hence techniques that can approximate complex circuits are definitely most wanted. This is promised by the modified CGP though there are some pitfalls for it too. TableIV analyses the worthiness of Method A and Method B where we could see Method B has a few benefits and lot of potential as it is *explorative* in nature. Also Non destructive mutation rate NDM is a win-win for method B.

TABLE IV
ANALYSIS OF METHODS

Type	Compact circuits	Std deviation	Biggest success	Fitness landscape	exploration	NDM% A>B A<B
Method A	10/21	Low	Alu4	Smooth	less	Low (5/10)
Method B	11/21	Low	Alu4	Rugged	more	(10/11) (5/10)

The only disadvantage would be that as we could see from Table3 , only 50% of the circuits were able to be evolved effectively for with reduced delay. Otherwise the conventional method itself would suffice. Finally , Table V gives us the summary of the approaches covered so far where the promise for scalabilty is given by modified CGP approach.

TABLE V
SUMMARY

Approach	Technique	Area Savings	Energy Savings	Delay	Consistency	Scalability
[Sekanina 13]	CGP approx	Max 50% in (3,3)add	30-40% avg	Not minimized	>90% SAD and SHD	No
[Petrlik 13]	MultiObjective approx	Not mentioned	Not mentioned	Reduced	Most cases	Low
[Menghua 13]	Degeneracy in ESD ckt	No savings	Not mentioned	Avoids RC delay	>90%	No
[Vasicek12]	Modified CGP+Fast fitness	24% more	Not mentioned	Highly reduced	50%	High

VI. FUTURE WORK

The biggest issue faced in using evolutionary design for optimizing approximate circuits is the scalability of circuits . It is shown that smaller circuits are more effective in using Evolvable hardware but as larger circuits tend to increase the computation time to unaccepatable levels.

People interested in real time applications using approximate computing might want to replace their hardware with evolvable hardware to have an additional feature of fault tolerance. Also , there is no design methodology for

optimizing an approximate multiplier circuit by means of evolutionary design specifically .

High level logic synthesis of approximate circuits using evolutionary design approaches had high potential in Automated synthesis field. Brand new approach called ABACUS is discussed by Kumud Nepal[7] from Brown University. If we could use effective methods like modified CGP into the synthesis algorithm we might reap a lot of benefits.

VII. CONCLUSION

So we learnt that approximate circuits can be designed using search based (evolutionary) design approaches.The main objective of every approach is to find the best trade off between accuracy and chip area or power consumption or the delay. Cartesian Genetic Programming is a key technique that has facilitated the evolvable hardware community to explore in design of approximate circuits especially for area minimization. One point to note is that the authors have not optimized energy consumption much and have assumed that decrease in area contrasts with power conservation which might be not be true in few applications.

Optimization of complex circuits is a challenge for evolvable hardware community which demands an effective way to deal with controlling the input output combinations that needs to be considered for evolution , which exponentially increases when the size (number of gates) of the circuit is increased. Researchers can look into new fitness criteria in the GA of CGP or other kinds of objectives in multi objective(multiobjective GAs[15]) optimization for better trade offs between accuracy and area or power. These approaches in the end might turn out to be highly application specific but with further advancements and novelty we have the potential to expand out horizons.

REFERENCES

[1] Sekanina, L.; Vasicek, Z., "Approximate circuit design by means of evolvable hardware," Evolvable Systems (ICES), 2013 IEEE International Conference on , vol., no., pp.21,28, 16-19 April 2013, DOI= 10.1109/ICES.2013.6613278

[2] Menghua Mana, Shanghe Liua, Xiaolong Changa, Mai Lub, “The Biological Property of Synthetic Evolved Digital Circuits with ESD Immunity – Redundancy or Degeneracy?,” Journal of Bionic Engineering, pp. 396–403, 2013.

[3] Petrlik, J.; Sekanina, L., "Multiobjective evolution of approximate multiple constant multipliers," Design and Diagnostics of Electronic Circuits & Systems (DDECS), 2013 IEEE 16th International Symposium on , vol., no., pp.116,119, 8-10 April 2013. DOI= 10.1109/DDECS.2013.6549800

[4] Vasicek, Z.; Sekanina, L., "On area minimization of complex combinational circuits using cartesian genetic programming," Evolutionary Computation (CEC), 2012 IEEE Congress on , vol., no., pp.1,8, 10-15 June 2012. DOI= 10.1109/CEC.2012.6256649

[5] Ye, Rong; Wang, Ting; Yuan, Feng; Kumar, Rakesh; Xu, Qiang, "On reconfiguration-oriented approximate adder design and its application," Computer-Aided Design (ICCAD), 2013 IEEE/ACM International Conference on , vol., no., pp.48,54, 18-21 Nov. 2013, DOI= 10.1109/ICCAD.2013.6691096

[6] Trading Accuracy for Power with an Underdesigned Multiplier Architecture; Parag Kulkarni, Puneet Gupta, Milos Ercegovac , 2011 24th Annual Conference on VLSI Design, 1063-9667/11 $26.00 © 2011 IEEE DOI 10.1109/VLSID.2011.51

[7] Automated High-Level Synthesis of Low Power/Area Approximate ComputingCircuits , Kumud Nepal ,Yueting Li , R. Iris Bahar , Sherief Reda CONF ’13. In press. Full version to be expected in 2014 http://dx.doi.org/10.1145

[8] ACMA: Accuracy-Configurable Multiplier Architecture for Error-Resilient System-on-Chip, Kartikeya Bhardwaj, Pravin S. Mane, 978-1-4673-6180-4/13/$31.00 ©2013 IEEE

[9] Venkataramani, S.; Sabne, A.; Kozhikkottu, V.; Roy, K.; Raghunathan, A., "SALSA: Systematic logic synthesis of approximate circuits," Design Automation Conference (DAC), 2012 49th ACM/EDAC/IEEE , vol., no., pp.796,801, 3-7 June 2012

[10] Fault Tolerance Analysis and Self-Healing Strategy of Autonomous, Evolvable Hardware Systems, Ruben Salvador, Andres Otero, Javier Mora, Eduardo de la Torre, Lukáš Sekanina, Teresa Riesgo 2011 International Conference on Reconfigurable Computing and FPGAs,ISBN: 978-1-4577-1734-5.

[11] A. Thompson, “Silicon Evolution,” In Proceedings of the First Annual Conference on Genetic Programming (GECCO '96). . 1996. MIT Press, Cambridge, MA, USA, 444-452.

[12] J. F. Miller and P. Thomson, "Cartesian Genetic Programming," in Proc. of the 3rd European Conference on Genetic Programming EuroGP2000, ser. LNCS, vol. 1802. Springer, 2000, pp. 121-132.

[13] J. F. Miller, “On the filtering properties of evolved gate arrays,” in 1st NASA-DoD Workshop on Evolvable Hardware. IEEE Computer Society, 1999, pp. 2–11.

[14] R. Ashraf, F. Luna, D. Dechev, and R. F. DeMara, “Designing digital circuits for FPGAs using parallel genetic algorithms,” 2012 Spring Simulation Multi-conference (SpringSim 2012), Orlando, FL, USA, March25-28,2012.

[15] N. Imran, R. A. Ashraf, and R. F. DeMara, “Power and Quality-Aware Image Processing Soft-Resilience using Online Multi-Objective GAs,” accepted to International Journal of Computational Vision and Robotics onMarch6,2014,in-press.

[16] R. A. Ashraf and R. F. DeMara, “Scalable FPGA Refurbishment using Netlist-driven Evolutionary Algorithms,” IEEE Transactions on Computers, vol. 62, no. 8, pp. 1526 – 1541, August 2013.

[17] Miller J. F., Harding S. L. Tutorial on Cartesian Genetic Programming, Genetic and Evolutionary Computation Conference, Montreal, CA (2009)

[18] Deb, K. and Goel, T., Controlled Elitist Non-dominated Sorting GeneticAlgorithms for Better Convergence, In Proc. of the Evolutionary Multi-Criterion Optimization, LNCS 1993, Springer, 2001, pp. 67–81

[19] G. S. Tseitin, "On the complexity of derivation in propositional calculus," in Studies in Constructive Mathematics and Mathematical Logic,Part11,1968,pp.115-125.