Accuracy and Physical Characterization of Approximate Arithmetic Circuits

Daniela Catelan, Ricardo Santos, Liana Duenha

[1]College of Computing (FACOM)
Federal University of Mato Grosso do Sul - UFMS

{daniela, ricardo, lianaduenha}@facom.ufms.br

Abstract. *With the end of Dennard's scale, designers have been looking for new alternatives and approximate computing (AC) has managed to attract the attention of researchers, by offering techniques ranging from the application level to the circuit level. When applying approximate circuit techniques in hardware design, the program user may speed up the application while a designer may save area and power dissipation at the cost of less accuracy on the operations results. This paper discusses the compromise between accuracy versus physical efficiency by presenting a set of experiments and results of tailor-made approximate arithmetic circuits on Field-Programmable Gate Array (FPGA) platforms. Our results reveal that an approximate circuit with accuracy control could not be useful if the goal is to save circuit area or even power dissipation. Even for circuits that seem to have power efficiency, we should care about the size and prototyping platform where the hardware will be used.*

1. Introduction

The inability to fit more cores within constrained power budgets has been projected to threaten even the multicore scaling paradigm. As a result, designers are being pushed to seek new sources of computing. The workloads that drive computing demand has also changed fundamentally across the computing spectrum, from mobile devices to the cloud [Venkataramani et al. 2015]. Such workloads exhibit intrinsic application resilience to approximations, an ability to produce acceptable outputs even when some of their computations are not accurate.

Approximate computing (AC) has gained the interest of researchers offering techniques ranging from the application level to the circuit level and as dedicated hardware modules. Approximate computing is based on the intuitive observation that while performing an exact computation or maintaining peak-level service demand requires a high amount of resources, allowing selective approximation or occasional violation of the specification can provide gains in efficiency [Mittal 2016].

There are several proposals for arithmetic circuits, mainly for approximate additions, subtractors, and multipliers [Jiang et al. 2015, Gorantla and Deepa 2019, Kulkarni et al. 2011]. Most of such studies present circuits with only one bit, showing details such as error positioning, area use, power and delay, and results estimates based on an application, usually an image, where an approximate calculation is most used. By evaluating small circuits, those work lack the opportunity to analyse how the circuit accuracy and physical behavior is in the presence of long inputs and outputs and even in a real-world design platform.

This paper presents a detailed study on approximated arithmetic circuits designed on a field-programmable gate array (FPGA) platform. Our goal was to explore how approximated circuits could be fitted to largely-used and flexible prototyping platforms such as FPGAs but with less flexibility on power and area management. Our results have shown that some circuits are well adapted to FPGA platforms and can take advantage of its organization better than others. By evaluating those circuits considering accuracy, area, and power dissipation, this work aims to characterize approximate arithmetic circuits showing that, on one hand, there is a tradeoff between accuracy and physical parameters and, on the other hand, approximate circuits are dependent on hardware design to provide better area and power savings.

2. Approximate Circuits

Approximate computing techniques range from software techniques to select code regions able to explore approximate computations to hardware designs providing approximate circuits where any application can use it. Looking at approximate circuit techniques, the research field has many and different proposals ([Gupta et al. 2013], [Yang et al. 2013], [Almurib et al. 2016], [Jiang et al. 2015], [Muthulakshmi et al. 2018], [Gorantla and Deepa 2019]), mostly focusing on approximate arithmetic such as adders and subtractors. Some studies present circuits with just one bit, showing their details such as error positioning, area usage, power, and delay.

Most researches present only a single circuit where the goal is to control the accuracy. As an example, [Muthulakshmi et al. 2018] present four types of subtractors (APSC4-APSC7), where the difference between them is only the position where the error was inserted in the truth table.

The approximate adder designs of AMA type (*Approximate Mirror Adder* - AMA) [Gupta et al. 2013] are based on the Mirror Adder. In an AMA design, the transistors of the conventional circuit were removed one by one, always ensuring that for any combination of the inputs A, B and Cin would not result in a short circuit or open circuit. A few errors in the truth table were inserted and a set of approximate adder circuits were derived (AMA1, AMA2, AMA3, and AMA4). AMA1, AMA3, and AMA4 were evaluated in this work.

The AXA adder (*Approximate XOR/XNOR - based Adder*) [Yang et al. 2013] was designed based on the transistor removal technique. The complete AXA1 adder is an exact adder with XOR gates and its implementation has used 8 transistors. The AXA2 adder is based on XNOR ports requiring 6 transistors. The AXA3 adder was based on the AXA2 adder with two more transistors for better accuracy. The three models were evaluated in this work.

The approximate InXA1, InXA2, and INXA3 adder models [Almurib et al. 2016] were developed with inexact adder cells with less circuit complexity compared to other approximate circuits found in the area (AMA, AXA, LOA, ESA). The InXA1 adder has an approximate design on the $Cout$ output. InXA2 and InXA3 has approximate designs on the S output. InXA1 and InXA3 were evaluated in this work.

Table 1 presents the logical expressions of the approximate adder circuits. One can observe that the AXA1 adder has a complex logic in its $Cout$ output, AMA1 and

AMA3 adders have the same expression in the $Cout$ output. AMA4 and InXA1 adders have the simplest expressions.

Table 1. Logical expressions for approximate adders

Circuits	Output	Logical Expression
AMA1	S	$\overline{A}\overline{B}Cin + AB\overline{Cin}$
	Cout	$\overline{ACin + B}$
AMA3	S	$\overline{(ACin + B)}$
	Cout	$\overline{ACin + B}$
AMA4	S	$\overline{A}Cin + B\overline{Cin}$
	Cout	A
AXA1	S	$\overline{Cin}$
	Cout	$\overline{((A \oplus B)Cin + \overline{AB})}$
AXA2	S	$\overline{(A \oplus B)}$
	Cout	$(A \oplus B)Cin + AB$
AXA3	S	$(A \oplus B)'Cin$
	Cout	$(A \oplus B)Cin + AB$
InXA1	S	$A \oplus B \oplus Cin$
	Cout	$\overline{Cin}$
InXA3	S	$\overline{Cout}$
	Cout	$(A \oplus B)Cin + AB$

Approximate subtractors were designed in [Gorantla and Deepa 2019], using the circuit minimization technique (Karnaugh Maps) to reduce the number of logical components the circuit. Two approximate subtractor circuits were designed where the first one performs approximation only at the S output. The second circuit performs the approximation at the $Cout$ output. Table 2 shows the logical expressions of the four approximate subtractors. They are called APSC4, APSC5, APSC6, and APSC7. In all subtractors, a single error was inserted in the S output.

Table 2. Logical expressions for the approximate subtractors

Circuits	Output	Logical expressions
APSC4	S	$\overline{A}B + B\overline{Cin} + \overline{A}\overline{Cin} + A\overline{B}Cin$
APSC5	S	$A\overline{B} + \overline{B}\overline{Cin} + \overline{A}B\overline{Cin} + ACin$
APSC6	S	$AB\overline{Cin} + \overline{A}\overline{B}\overline{Cin} + \overline{A}\overline{B}Cin$
APSC7	S	$A\overline{B}\overline{Cin} + \overline{A}\overline{B}Cin + AB\overline{Cin}$

All the subtractors have the same logical expression for the $Cout$ output:

$$Cout = \overline{(A \oplus B)}Cin + \overline{A}B$$

Kulkarni et al [Kulkarni et al. 2011] propose an approximate multiplier circuit by modifying the logic function of its building block (2-bits multiplier). The experiments showed an average error ranging from 1.39% to 3.35% and the energy savings were from 30% to 50% using a 4-bit approximate multiplier.

The circuit of an approximate divider using subtractor circuits of the APSC type is presented in [Gorantla and Deepa 2019]. A 4-bit divider uses 16 subtractors, divided into 4 rows with 4 subtractors in each. In the proposed divider circuit, i approximate subtractors are inserted in row i, $1 \leq i \leq 4$. The complete divisor circuit has 6 exact subtractors and 10 approximate subtractors. The approximate subtractors are positioned at the least significant bits of each row.

3. Experimental Setup

This section presents the setup of experiments on approximate arithmetic circuits built on the top of a reconfigurable FPGA platform. The experiments have been divided into 3 types of circuits: adders, subtractors, and multipliers with different sizes (8-, 16-, 32-, and 64-bit). Approximate and accurate (exact) circuits are compared from accuracy, area occupation, and power dissipation.

The experiments have used 8 approximate adder circuits: AMA1, AMA3, AMA4, AXA1, AXA2, AXA3, InXA1, and InXA3, and 4 types of approximate subtractor circuits, APSC4, APSC5, APSC6, and APSC7. The approximate multiplier circuits design were built based on the approximate adder as building-blocks.

All circuits were designed in VHDL, synthesized on the Cyclone IV GX FPGA using the ALTERA Quartus II Web Edition 13.1 IDE tool. Simulation scripts and testbenches were simulated on the Altera-ModelSim 10 software. The area usage results come from the number of logic elements used by the circuit design. The total power dissipated was obtained by the PowerPlay Power Analyzer tool. PowerPlay estimates the design power dissipation (PD) by the sum of static power (SP), dynamic power (DP), and I/O power (IOP).

Google Colab was used to generate 2^{20} random inputs to evaluate the 16-, 32- and 64-bit circuits. We have applied statistical tools to validate if the random 64-bit inputs are representative of the complete truth table. We assume a standard normal distribution for the samples and build a confidence interval for the population mean and the test statistic. The 64-bit inputs has a population mean $\mu = 3.402824e + 38$. The randomly inputs for the 64-bit circuits had a sample mean $\overline{x} = 3.402707e + 38$. The calculated confidence interval was $[3.398946e + 38; 3.406468e + 38]$ for a confidence level of 95% meaning that the samples are representative for the mean population. We also calculated the test statistic $t = -0.06096544$, also indicating that the population mean is in the confidence interval of the randomly generated inputs.

The experiments were performed considering accurate circuits, approximate circuits, and mixed (accurate and approximate) circuits. The mixed circuits were designed so that the first half (least significant bits) was comprised of accurate circuits and the second half of approximate circuits. Mixed circuit results will be represented with an "M" as the first letter (M_AMA1).

The accuracy metric is based on the relative error of the approximate circuits over the accurate circuits. To calculate the relative error we developed a testbench performing a line-by-line comparison between the results of the exact circuit and the approximate circuit, making it possible to know which output of the approximate circuit had an error or hit. Equation 1 outlines the relative error (RE) by dividing the Approximate Circuit Correct Outputs (AO) to the Accurate Circuit Outputs (EO). The adder and subtractor circuits had the relative error calculated from its outputs $Cout$ and S.

$$RE = 1 - \frac{AO}{EO} \tag{1}$$

The area usage results were based on the relative area (RA) metric calculated from the number of total logic elements used by the approximate circuit design (ALE) and the accurate circuit (ELE) as presented in equation 2.

$$RA = \frac{ALE}{ELE} \tag{2}$$

Considering the power dissipation experiment, a simulation script was developed to generated the signals transitions into a .VCD file so that PowerPlay could estimate the total power dissipation (TP) from the average dynamic power (DP), the average static

power (EP), and the average I/O power (IOP) according to equation 3. The total power was the baseline metric to calculate the relative power (RP) of the circuits, considering Power Approximate Circuits (AP) and Power Exact Circuits (EP) in equation 4.

$$TP = DP + EP + IOP \tag{3}$$

$$RP = \frac{AP}{EP} \tag{4}$$

4. Results and Discussion

The results presented in this section are divided into accuracy experiments, area usage, power dissipation in approximate circuits and the impact of approximate arithmetic circuits on programs.

4.1. Accuracy Results

The relative error (Equation 1) for the approximate adders from $8-$ to 64-bit are shown in Table 3. The results on circuits 8-bit present high accuracy (some circuits even present small relative errors of 21.9%). The 8-bit circuits present a lesser relative error when compared to the 64-bit circuits. The ripple carry effect of an error that will be propagated to the most significant bits is the reason behind the best performance of small bit circuits.

Table 3. Accuracy result (relative error) for adder circuits

Circuits	8 Bits	16 Bits	32 Bits	64 Bits
AMA1	0.760	0.932	0.995	1.000
M_AMA1	0.547	0.759	0.933	0.995
AMA3	0.968	0.999	1.000	1.000
M_AMA3	0.826	0.968	0.999	1.000
AMA4	0.996	1.000	1.000	1.000
M_AMA4	0.938	0.996	1.000	1.000
AXA1	0.995	1.000	1.000	1.000
M_AXA1	0.918	0.995	1.000	1.000
AXA2	0.933	0.993	1.000	1.000
M_AXA2	0.789	0.933	0.994	1.000
AXA3	0.760	0.932	0.995	1.000
M_AXA3	0.547	0.759	0.933	0.995
InXA1	0.867	0.987	1.000	1.000
M_InXA1	0.578	0.866	0.987	1.000
InXA3	0.900	0.990	1.000	1.000
M_InXA3	0.684	0.900	0.990	1.000

Mixed circuits have a smaller number of errors (lesser relative error) compared to the approximate circuits of the same type. For example, the 8-bit AMA1 approximate adder circuit has 76% of relative error while its mixed type (M_AMA1) has 54.7%. Mixed circuits show better results also for 16-, 32-, and 64-bit circuits. This behavior was expected since that accurate circuits would decrease the relative error in an add operation. Another benefit from mixed approximate circuits is that both accurate and approximate designs into one larger adder could provide an option for using the same design to get accurate and approximate results.

To evaluate the best configuration for mixed circuits, the relative error was measured using several configurations: fully approximate circuits (TA) meaning that all building blocks are approximate adder circuits; mixed circuits in which the least significant building blocks are exact and the most significant blocks are approximate (M_E/A);

and mixed circuits in which the least significant bits come from approximate and the most significant ones come from exact circuits (M_A/E).

Most mixed adder circuits showed better results with the 4-bit M_E/A configuration. Only AMA4, AXA1 and InXA1 circuits showed better results in the M_A/E configuration. For 4-bit AMA3, InXA1, and InXA3 showed better results for the fully approximate type. For 8-bit circuits, the AXA1 and InXA1 adders show better results in the M_E/A configuration and the others show the same results in both configurations. The 16-bit mixed circuits showed the same results for both configurations so that for mixed large circuits the positioning of approximate and exact building blocks did not improve the accuracy of the whole circuit.

Table 4 presents the relative error for the subtractor circuits.One can notice that approximate subtractor circuits show better results than approximate adder circuits. Mixed circuits also show better results than fully approximated circuits.

Table 4. Accuracy result (relative error) for subtractor circuits

Circuits	8 Bits	16 Bits	32 Bits	64 Bits
APSC4	0.576	0.810	0.962	0.998
M_APSC4	0.365	0.575	0.809	0.962
APSC5	0.766	0.963	0.999	1.000
M_APSC5	0.473	0.766	0.963	0.999
APSC6	0.576	0.810	0.962	0.998
M_APSC6	0.365	0.576	0.810	0.962
APSC7	0.766	0.963	0.999	1.000
M_APSC7	0.473	0.766	0.963	0.999

The 8-bit fully approximated APSC4 subtractor has a relative error of 0.576 (57.6%) while the mixed subtractor has 0.365 (36.5%). The same behavior can be observed for all the subtractor circuits. Even so, as the circuits get larger the impact of mixed circuits on the accuracy is minimized, even still better than the fully approximate circuits, by the effect of carrying out the results.

Table 5 presents the multiplier relative error. The InXA1 based multiplier circuit presents the best accuracy results for all tests, while the M_InXA1 mixed circuit had the best accuracy result for the 8- and 16-bit circuits. The 64-bit multiplier circuits have 4032 adders, so the error propagation negatively impacts on the multiplier accuracy.

Table 5. Accuracy result (relative error) for multiplier circuits

Circuits	8 Bits	16 Bits	32 Bits
AMA3	1.000	1.000	1.000
M_AMA3	1.000	1.000	1.000
AMA4	0.984	1.000	1.000
M_AMA4	0.983	1.000	1.000
AXA1	0.974	1.000	1.000
M_AXA1	0.962	1.000	1.000
InXA1	0.880	0.999	1.000
M_InXA1	0.827	0.992	1.000
InXA3	1.000	1.000	1.000
M_InXA3	0.999	1.000	1.000

In order to analyze the best configuration for the mixed multiplier circuits, we have evaluate the mixed circuits with the least significant adders comprised of exact circuits (M_E/A) and mixed circuits with the least significant adders of the approximate

type (M_A/E). Most circuits show better results with the M_E/A 4-bit configuration; only AMA4 and InXA1 circuits show better results in the M_A/E configuration. For 16-bit, the AMA3 and InXA3 circuits have the same result regardless of the order of the approximate additions. For all the evaluated circuits, the best relative error results were found by using mixed circuits.

4.2. Area Usage

Table 6 shows the relative area of the approximate adder circuits. All the AMA1 approximate adders presented an area larger than the reference (accurate) adder circuit. The 16-bit AMA1 is up to 18.8% larger than the accurate adder circuit. The AMA4 circuit presented the lesser area usage, for all size configurations. The 64-bit is just 29.1% of the reference adder area, a decrease of 71% on the area usage.

Table 6. Relative area of approximate adder circuits

Circuits	8 Bits	16 Bits	32 Bits	64 Bits
AMA1	1.125	1.188	1.054	1.068
M_AMA1	1.063	1.094	0.959	0.853
AMA3	0.500	0.500	0.514	0.527
M_AMA3	0.750	0.813	0.716	0.784
AMA4	0.375	0.344	0.297	0.291
M_AMA4	0.688	0.656	0.581	0.642
AXA1	0.500	0.500	0.514	0.622
M_AXA1	0.750	0.813	0.784	0.851
AXA2	1.000	1.000	0.946	0959
M_AXA2	1.000	1.000	0.946	1.007
AXA3	1.000	1.000	1.000	1.000
M_AXA3	1.000	1.000	1.000	1.000
InXA1	0.500	0.500	0.432	0.432
M_InXA1	0.750	0.750	0.649	0.716
InXA3	0.500	0.500	0.432	0.432
M_InXA3	0.750	0.750	0.730	0.696

As expected, the mixed circuits do not show better results than the fully approximated circuits. Among all the adder circuits, the AMA4 approximate adder circuit shows the better area usage results once its output S is formed by just 3 logic gates ($S = ACin + BCin$) and its $Cout$ output is formed only by the A input of the circuit.

Table 7 presents the relative area usage of the subtractor circuits. It can be noted that most of the approximated circuits have not improved the area usage when compared to the accurate circuits. The approximated subtractor circuit has an area usage larger than the accurate circuit since the design of such circuits were focused on accuracy improvement. The 32-bit mixed subtractor circuit M_APSC5 has an area 8.1% smaller than the accurate subtractor circuit, being the circuit that presents the best result among all the circuits.

Table 8 shows the relative area usage for the approximate multiplier circuits. The 32-bit multiplier circuit has 992 adder circuits, so that the multiplier size (area usage) is dependent on the adder circuit area. One may observe that the AMA4 circuit, both the fully approximate and the mixed one, have the best results in area, followed by the fully approximated InXA1 circuit.

4.3. Power Dissipation

Regarding the total power dissipation, the AMA4 and M_AMA4 approximate adder circuits present the best results as presented in Table 9. The results also corroborate to the

Table 7. Relative area of subtractor circuits

Circuits	8 Bits	16 Bits	32 Bits	64 Bits
APSC4	1.000	1.125	1.081	1.068
M_APSC4	1.000	1.000	1.081	1.041
APSC5	1.000	1.000	1.081	1.068
M_APSC5	1.000	1.000	0.919	1.041
APSC6	1.188	1.094	1.095	1.074
M_APSC6	1.188	1.000	1.081	1.041
APSC7	1.000	1.125	1.081	1.068
M_APSC7	1.000	1.000	1.081	1.041

Table 8. Relative area of multiplier circuits

Circuits	8 Bits	16 Bits	32 Bits	64 Bits
AMA3	0.483	0.461	0.478	0.487
M_AMA3	0.700	0.743	0.746	0.747
AMA4	0.125	0.063	0.031	0.016
M_AMA4	0.183	0.188	0.157	0.140
AXA1	0.542	0.517	0.508	0.577
M_AXA1	0.733	0.760	0.754	0.759
InXA1	0.358	0.345	0.339	0.334
M_InXA1	0.658	0.693	0.676	0.668
InXA3	0.583	0.543	0.523	0.559
M_InXA3	0.767	0.774	0.761	0.759

area usage results once that small area circuits will have lesser power static power thus impacting on the final circuit total power dissipation.

Table 9. Relative power of adder circuits

Circuits	8 Bits	16 Bits	32 Bits	64 Bits
AMA1	0.699	1.005	0.997	0.999
M_AMA1	0.864	0.978	0.980	0.999
AMA3	0.524	1.000	1.003	0.999
M_AMA3	0.081	0.994	1.006	0.999
AMA4	0.137	0.958	0.976	0.986
M_AMA4	0.781	0.982	0.977	0.984
AXA1	0.759	0.961	1.007	1.000
M_AXA1	0.852	0.978	1.006	1.000
AXA2	0.502	1.000	0.995	1.001
M_AXA2	0.787	0.967	0.997	1.000
AXA3	0.833	1.000	1.000	0.999
M_AXA3	0.902	1.002	1.000	1.000
InXA1	1.622	1.001	0.991	1.000
M_InXA1	1.278	0.967	0.992	1.001
InXA3	0.851	1.000	0.991	0.999
M_InXA3	0.922	0.978	0.997	0.999

As expected, the subtractor circuits did not show a lesser relative power when compared to the reference circuits (Table 10). The approximate subtractor circuits had a larger relative area thus increasing the static power dissipation of the circuit.

The 32-bit subtractor circuit has the total average power, dynamic, static, and I/O quite akin to the fully approximate and mixed circuits. Only the average dynamic power has a small difference between the fully approximate circuit (8.10mW) to the mixed circuit (8.89mW).

Table 11 shows the relative power results of the approximate multiplier circuits. The results on the multiplier relative power are in the range from 8-bit to 32-bit since the Altera-ModelSim version 10 was not able to simulate multiplier circuits designs on 64-bit.

Table 10. Relative power of subtractor circuits

Circuits	8 Bits	16 Bits	32 Bits	64 Bits
APSC4	0.880	0.983	0.995	1.000
M_APSC4	0.940	1.000	1.004	1.001
APSC5	0.880	0.967	0.995	1.000
M_APSC5	0.940	1.000	0.992	1.001
APSC6	0.905	0.994	1.003	1.000
M_APSC6	0.930	1.000	0.988	0.999
APSC7	0.916	0.983	0.995	1.000
M_APSC7	0.957	1.000	1.003	1.000

Even being built on the top of approximate adder building blocks, most of the approximate multiplier circuits had better relative power compared to the accurate multiplier circuit.

Table 11. Relative power of multiplier circuits

Circuits	8 Bits	16 Bits	32 Bits
AMA3	1.024	0.967	0.998
M_AMA3	1.022	0.988	0.996
AMA4	0.196	0.968	0.982
M_AMA4	0.288	0.938	0.981
AXA1	1.034	1.006	0.998
M_AXA1	1.027	0.985	0.997
InXA1	0.676	0.960	0.982
M_InXA1	0.774	0.947	0.983
InXA3	0.298	0.969	0.988
M_InXA3	0.576	0.994	0.999

One may note that the relative power for the AMA4 8-bit is 0.196 while the 32-bit configuration is 0.982. The $5\times$ relative power increase is mostly due to the increase in I/O power dissipation. Large circuits designed on a FPGA platform will require the usage of more logical blocks (logical elements) and the routing among them is determinant to the increase of I/O power and its impact on the circuit total power. As for the 32-bit multipliers, fully approximated circuits show better results, unlike adder and subtractor circuits, the average power is equal to 18311.23mW for fully approximated and 18342.87mW for mixed circuits.

Figure 1 shows the relationship between the accuracy (relative error) and power dissipation (relative power) for the approximate adder, subtract, and multiplier circuits prototyped on the FPGA platform.

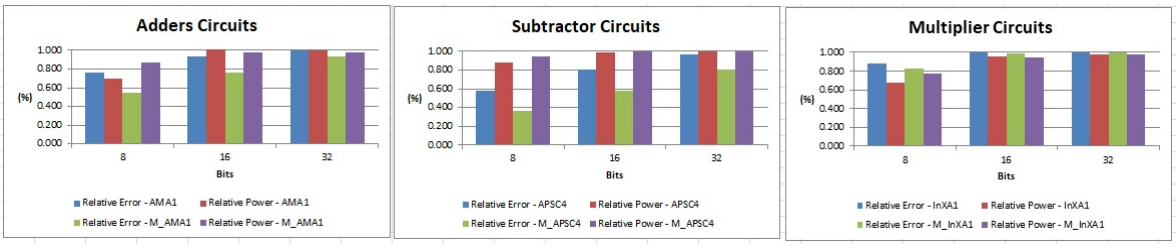

Figure 1. Relative Error X Relative Power

The accuracy is linearly increased according to the circuit size. The relative power, however, did not present power dissipation decrease from the approximate circuits even in 16-bit circuits. There are many proposals presenting circuits and results on power and

area benefits greater than 50%. A general-purpose, rapid-prototyping digital hardware platform such as FPGAs, where the designer does not have the flexibility to control or regulate the input voltage, the area usage, and other hardware elements, may not bring such benefits by synthesizing approximate circuits.

4.4. Impact of Approximate Arithmetic Circuits on Programs

Tables 12 and 13 present the number and percentage of instructions groups of 7 programs of the ParMiBench [Iqbal et al. 2010] and 4 programs of the SPLASH-2 [Woo et al. 1995] benchmark suites, obtained from the ArchC MIPS simulator [Azevedo et al. 2005]. The goal is to estimate potential benefits from approximate circuits on the add, subtraction, and multiplication instructions so that the total power dissipation by the programs could be improved compared to accurate circuits. We calculate the relative power for each program considering the usage of approximate adder AMA4, subtractor APSC4, and multiplier AMA4 circuits on add, subtraction, and multiplication instructions.

To calculate the relative power per program (RP), we compute the relative power of each 32-bit approximate circuit $(RP_{circuit})$ by the percentage of each instruction group (P_{instr}), according to equation 5.

$$RP = RP_{AMA4} \times P_{sum} + RP_{APSC4} \times P_{subtraction} + RP_{AMA4multiplier} \times P_{multiplication} + P_{others} \quad (5)$$

Table 12. ParMiBench benchmark relative power

	Sum	Subtraction	Multiplication	Others	Relative Power
Basicmath	821970 (15%)	88000 (2%)	39612 (1%)	4360685 (82%)	0.996
Dijkstra	936498 (20%)	288218 (6%)	3834 (0%)	3344451 (73%)	0.985
Susancorners	16030167 (19%)	4101981 (5%)	458096 (1%)	65500219 (76%)	1.005
susanedges	36558718 (15%)	3240678 (1%)	2739311 (1%)	198543246 (82%)	0.986
SHA	138802 (15%)	403 (0%)	2 (0%)	781216 (85%)	0.996
Stringsearch	20995070 (22%)	141705 (0%)	1045 (0%)	74211889 (78%)	0.995
susansmoothing	12903133 (14%)	1898885 (2%)	3299095 (4%)	71522713 (80%)	0.996

Table 13. SPLASH2 benchmark relative power

	Sum	Subtraction	Multiplication	Others	Relative Power
FFT	12035926 (20%)	753080 (1%)	342893 (1%)	46485596 (78%)	0.995
LU	11671541 (19%)	957264 (2%)	440495 (1%)	47309344 (78%)	0.995
water	31682029 (18%)	3147636 (2%)	663635 (0%)	140190263 (80%)	0.996
waterspatial	207016168 (18%)	19768670 (2%)	4457714 (0%)	913849588 (80%)	0.996

The relative power results per program show small benefits from using approximate circuits on a small number of instructions and operations. The reason for such lesser relative power is that the three groups of instructions represent less than a quarter (25%) of the instructions in most of the programs. A more realistic and optimistic program analysis should consider that may there are instructions of 8- and 16-bit that could take advantage of better relative power of 8- and 16-bit circuits mostly on adder and multiplier circuits. Additionally, even different instructions groups could use the approximate arithmetic circuits thus increasing the number of program instructions that could use the approximate circuits.

5. Conclusions

This work presented a comprehensive set of experiments encompassing from accuracy to physical characterization of arithmetic approximation circuits on an FPGA platform. The characterization was built based on accuracy, area, and power dissipation metrics comparing accurate (exact) and approximation adder, subtractor, and multiplier circuits in sizes ranging from 8- to 64-bit.

Even being tailor-made approximation circuits design where the accuracy is one of the main goals, we have observed that most of the evaluated circuits do not maintain its accuracy levels as the circuit increases. As an example, the mixed AMA1 design has a relative error of 54.7% at 8-bit and a relative error of 99.5%at 64-bit. The same behaviour was observed in all other evaluated circuits. Considering the area usage, the AMA4 adder has an usage of only 29.1%. The subtractor circuits had not any benefits from the area usage. The multiplier circuits were based on the adder approximation circuits so that the multiplier based on AMA4 had an area usage of only 1.6%.

Our results show that tailor made approximation circuits bring useful accuracy levels, area usage and power dissipation at small circuits but the benefits are reduced for large input circuits. The physical characterization results show that the best results on area usage and power dissipation are dependent on the hardware platform. A rapid prototyping platform such as FPGA may not bring the area and power efficiency that would be required to get the benefits of a large approximate circuit.

On the other hand, rapid prototyping platforms are able to make the design of mixed (accurate and approximate) circuits scalable and simple to use, so that a digital hardware design may take advantage of the accurate and approximate circuits according to the program features.

This work can be extended by new experiments on tailor-made multiplier, divider, and floating-point approximate circuits under the same metrics and hardware platforms here presented. Another future work will be on using approximate circuits as building blocks of a design space exploration system.

References

Almurib, H. A. F., Kumar, T. N., and Lombardi, F. (2016). Inexact designs for approximate low power addition by cell replacement. In *Proceedings of the 2016 Conference on Design, Automation & Test in Europe*, DATE '16, page 660–665, San Jose, CA, USA. EDA Consortium.

Azevedo, R., Rigo, S., Bartholomeu, M., Araujo, G., Araujo, C., and Barros, E. (2005). The ArchC Architecture Description Language and Tools. In *International Journal of Parallel Programming. Vol. 33, No. 5*, pages 453–484.

Gorantla, A. and Deepa, P. (2019). Design of approximate subtractors and dividers for error tolerant image processing applications. *Journal of Electronic Testing*, pages 1–7.

Gupta, V., Mohapatra, D., Raghunathan, A., and Roy, K. (2013). Low-power digital signal processing using approximate adders. *IEEE Trans. on CAD of Integrated Circuits and Systems*, 32(1):124–137.

Iqbal, S., Liang, Y., and Grahn, H. (2010). Parmibench - an open-source benchmark for embedded multiprocessor systems. *Computer Architecture Letters*, 9(2):45–48.

Jiang, H., Han, J., and Lombardi, F. (2015). A comparative review and evaluation of approximate adders. In *Proceedings of the 25th Edition on Great Lakes Symposium on VLSI*, GLSVLSI '15, page 343–348, New York, NY, USA. Association for Computing Machinery.

Kanani, A., Mehta, J., and Goel, N. (2020). Aca-csu: A carry selection based accuracy configurable approximate adder design. In *2020 IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*, pages 434–439.

Kulkarni, P., Gupta, P., and Ercegovac, M. D. (2011). Trading accuracy for power in a multiplier architecture. *Journal of Low Power Electronics*, 7(4):490–501.

Lahari, P. L., Bharathi, M., and Jyothi Shirur, Y. (2020). An efficient truncated mac using approximate adders for image and video processing applications. In *2020 4th International Conference on Trends in Electronics and Informatics (ICOEI)(48184)*, pages 1039–1043.

Lee, J., Seo, H., Kim, Y., and Kim, Y. (2020). Approximate adder design with simplified lower-part approximation. *IEICE Electronics Express*.

Mittal, S. (2016). A survey of techniques for approximate computing. *ACM Computing Surveys (CSUR)*, 48(4):1–33.

Muthulakshmi, S., Dash, C., and Prabaharan, S. (2018). Memristor augmented approximate adders and subtractors for image processing applications: An approach. *AEU - International Journal of Electronics and Communications*, 91.

Nayar, R., Balasubramanian, P., and Maskell, D. L. (2020). Hardware optimized approximate adder with normal error distribution. In *2020 IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*, pages 84–89.

Venkataramani, S., Chakradhar, S. T., Roy, K., and Raghunathan, A. (2015). Approximate computing and the quest for computing efficiency. In *2015 52nd ACM/EDAC/IEEE Design Automation Conference (DAC)*, pages 1–6. IEEE.

Woo, S. C., Ohara, M., Torrie, E., Singh, J. P., and Gupta, A. (1995). The Splash-2 Programs: Characterization and Methodological Considerations. In *Proceedings of the 22nd International Symposium on Computer Architecture - ISCA'95*, pages 24–36. ACM.

Yang, Z., Jain, A., Liang, J., Han, J., and Lombardi, F. (2013). Approximate xor/xnor-based adders for inexact computing. *Proceedings of the IEEE Conference on Nanotechnology*, pages 690–693.