Implementation of Dynamically reconfigurable Arithmetic unit for Video Encoding

[1] Mahesh Kumar N [2] Vijay Kumar S Patil, [3] Sadyojatha K M
[1] M.Tech Student, [2] Assistant Professor , [3] Professor, Dept. Of ECE
BITM-Ballari-583104

Abstract: The field of approximate computing has received significant attention from the research community in the past few years, especially in the context of various signal processing applications. Image and video compression algorithms, such as JPEG, MPEG, and so on, are particularly attractive candidates for approximate computing, since they are tolerant of computing imprecision due to human imperceptibility, which can be exploited to realize highly power-efficient implementations of these algorithms. However, existing approximate architectures typically fix the level of hardware approximation statically and are not adaptive to input data. For example, if a fixed approximate hardware configuration is used for an MPEG encoder (i.e., a fixed level of approximation), the output quality varies greatly for different input videos. This paper addresses this issue by proposing a reconfigurable approximate architecture for MPEG encoders that optimizes power consumption with the goal of maintaining a particular Peak Signal-to-Noise Ratio (PSNR) threshold for any video. Toward this end, we design reconfigurable adder/subtractor blocks (RABs), which have the ability to modulate their degree of approximation, and subsequently integrate these blocks in the motion estimation and discrete cosine transform modules of the MPEG encoder. We propose two heuristics for automatically tuning the approximation degree of the RABs in these two modules during runtime based on the characteristics of each individual video. Experimental results show that our approach of dynamically adjusting the degree of hardware approximation based on the input video respects the given quality bound (PSNR degradation of 1%–10%) across different videos while achieving a power saving up to 38% over a conventional non approximated MPEG encoder architecture. Note that although the proposed reconfigurable approximate architecture is presented for the specific case of an MPEG encoder, it can be easily extended to other DSP applications. Index Terms—Approximate circuits, approximate computing, low power design, quality configurable.

Keywords— MPEG, Carry look ahead (CLA) adder, Pixel, RGB component, Video compression.

I. INTRODUCTION

Introducing a limited amount of computing imprecision in image and video processing algorithms often results in a negligible amount of perceptible visual change in the output, which makes these algorithms as ideal candidates for the use of approximate computing architectures.

Approximate computing architectures exploit the fact that a small relaxation in output correctness can result in significantly simpler and lower power implementations. However, most approximate hardware architectures proposed so far suffer from the limitation that, for widely varying input parameters, it becomes very hard to provide a quality bound on the output, and in some cases, the output quality may be severely degraded. The main reason for this output quality fluctuation is that the degree of approximation (DA) in the hardware architecture is fixed statically and cannot be customized for different inputs. One possible remedy is to adopt a conservative approach and use a very low DA in the hardware so that the output accuracy is not drastically affected. However, such a conservative approach will, as expected, drastically impact the power savings as well. This paper adopts a different approach to addressing this problem by dynamically reconfiguring the approximate hardware architecture depending on the inputs. Specifically, this paper makes the following contributions.

1) We demonstrate that, for a fixed level of hardware approximation in an MPEG encoder, the output quality varies widely across different videos, often going below acceptable limits. This shows that setting the level of hardware approximation statically is insufficient.

2) We investigate, for the first time, the use of dynamically reconfigurable approximate hardware architectures that vary the DA during run-time across multiple computational cycles, depending on the inputs. Toward this end, we propose the design of reconfigurable adder/subtractor blocks (RABs) for four commonly used adder architectures, viz., ripple carry adder (RCA), carry look ahead adder (CLA), carry bypass adder (CBA), and carry select adder (CSA), and subsequently integrate them into the MPEG encoder to enable quality configurable execution.

3) We propose a design methodology to adapt the DA dynamically based on the video characteristics with the goal of ensuring that output quality is within a specified bound.

4) We have implemented the proposed architecture for an MPEG encoder on an Altera DE2 field-programmable gate array (FPGA) board and evaluated it using eight benchmark videos. Our experimental results show that the proposed architecture results in power savings equivalent to a baseline approach that uses fixed approximate hardware while

respecting quality constraints across different videos. The remainder of this paper is organized as follows. Section II gives an account of related work in the domain of approximate computing. Section III provides a concise summary of the MPEG compression standard as well as a brief description of the metrics used for video quality evaluation. A case study that serves as the motivation for our work and the proposed reconfigurable approximate architecture for MPEG encoding are described in Sections IV and V, respectively. Section VI reports the results obtained through hardware implementation for our design on an FPGA, and Section VII concludes this paper.

II. METHODOLOGY

There has been a lot of effort in constructing energy-efficient video compression schemes. Many of them are related to the specific case of an MPEG encoder. Different methods of power-reduction include algorithmic modifications [1], [2], voltage over-scaling [3], and imprecise computation of metrics [4]. The introduction of approximate computing techniques has opened up entirely new opportunities in building low-power video compression architectures. Approximate computing methods achieve a large amount of power savings by introducing a small amount of error or inaccuracy into the logic block.

Different approaches for approximation include error introduction through voltage overscaling [5], [6], intelligent logic manipulation [7], and circuit simplification using don't care-based optimization techniques [8]. The methods in [9] and [10] introduce imprecision by replacing adders with their approximate counterparts. The approximate adders are obtained by intelligently deleting some of the transistors in a mirror adder. An important point to note is that these approximate circuits are hardwired and cannot be modified without resynthesizing the entire circuit. There also exist instances of approximations introduced in an MPEG encoder [5], [11]–[13]. Most of them exploit the inherent error resilience of the motion estimation (ME) algorithm, which results in minor quality degradation. For example, Moshnyaga *et al.* [11] use a bitwidth compression technique to reduce power consumption of video frame memory. He and Liou [12] and He *et al.* [13] use bit truncation to introduce approximations in the ME block of an MPEG encoder. An adaptive bit masking method is proposed in [13], where the authors propose to truncate the pixels of the current and previous frames required for ME depending upon the quantization step.

However, such a coarse-grained input truncation is applicable only to the specific case of ME and gives unsatisfactory results for other blocks, such as discrete cosine transform (DCT), which requires a finer regulation over error. As in [9] and [10], this paper also aims in approximating the adders of the ME and DCT blocks of an MPEG encoder. However, this paper introduces the concept of dynamically reconfigurable approximation, which, as we will show, helps in maintaining better control over application-level quality metrics while simultaneously reaping the power consumption benefits of hardware approximation. Our proposed technique can automatically adjust the extent of hardware approximation dynamically based on the video characteristics. In addition, such dynamic reconfiguration also provides users with a control knob for varying the output quality of the videos and the power consumption for the battery-powered multimedia devices. Note that a preliminary version of this paper appeared in [14]. Compared with that work, this paper includes a number of additional features as described here. We augment the heuristics for modulating the DA of the reconfigurable hardware blocks by adding the feature of most significant bit (MSB) truncation, which improves the energy-quality tradeoff during the video encoding process.

We also extend the RAB to include three additional adder architectures, viz., CLA, CBA, and CSA. In addition, for the carry look ahead based RAB, we propose dual-mode carry lookahead and propagate-generate blocks as its constituent basic building blocks. Finally, we provide a comparative study of the power consumption of the different RABs and also demonstrate how the DA is automatically regulated across different frames during run-time.

## III. TRADITIONAL METHOD.

A. MPEG Compression Scheme

MPEG has for long been the most preferred video compression scheme in modern video applications and devices. Using the MPEG-2/MPEG-4 standards, videos can be squeezed to very small sizes. MPEG uses both interframe and intraframe encoding for video compression. Intraframe encoding involves encoding the entire frame of data, while interframe encoding utilizes predictive and interpolative coding techniques as means of achieving compression. The interframe version exploits the high temporal redundancy between adjacent frames and only encodes the differences in information between the frames, thus resulting in greater compression ratios. In addition, motion

compensated interpolative coding scales down the data further through the use of bidirectional prediction. In this case, the encoding takes place based upon the differences between the current frame and the previous and next frames in the video sequence. MPEG encoding involves three kinds of frames: 1) I-frames (intraframe encoded); 2) P-frames (predictive encoded); and 3) B-frames (bidirectional encoded). As evident from their names, an I-frame is encoded completely as it is without any data loss.

An I-frame usually precedes each MPEG data stream. P-frames are constructed using the differences between the current frame and the immediately preceding I or P frame. B-frames are produced relative to the closest two I/P frames on either side of the current frame. The I, P, and B frames are further compressed when subjected to DCT, which helps to eliminate the existing interframe spatial redundancy as much as possible.

A significant portion of the interframe encoding is spent in calculating motion vectors (MVs) from the computed differences. Each nonencoded frame is divided into smaller macro blocks (MBs), typically 16×16 pixels.
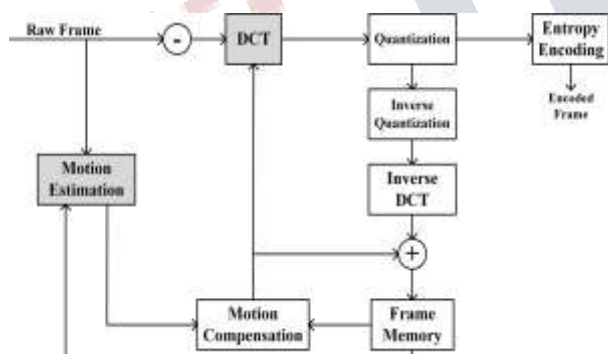
Fig1. MPEG encoder block diagram.

Each MV has an associated MB. The MVs actually contain information regarding the relative displacements of the MBs in the present frame in comparison with the reference. These are calculated by extracting the minimum value of sum of absolute differences (SADs) of an MB with respect to all the MBs of the reference frame. The resultant vectors are also encoded along with the frames. However, this is not sufficient to provide an accurate description of the actual frame. Hence, in addition to the MVs, a residual error is computed, which is then compressed using DCT. It has been proven that the ME and DCT blocks are the most computationally expensive

components of an MPEG encoder [10], [15]. The different steps involved in performing MPEG compression are shown in Fig. 1.

B. Quality of a Video
The merit of the encoding operation can be determined from the output quality of the decoded video. Objective metrics, such as Peak Signal-to-Noise Ratio (PSNR), SAD, and so on, have a very good correlation with the subjective procedures of measuring the quality of the videos [16], [17]. Hence, we have utilized the popular and simple PSNR metric as a means of video quality estimation. PSNR is a full-reference video quality assessment technique, which utilizes a pixel-to-pixel difference with respect to the original video. In this paper, PSNR of a video is defined as the average PSNR over a constant number of frames (50) of the video.

Therefore, the extra ECW N/4 is removed by the transformation of 4 partial product variables and one partial product row is saved in RB multipliers with any power-of-two word-length. In the second stage, a 4-stage RBA summing tree is used to sum 16 RB partial products. Each RBA block contains 64 RB full adder (RBFA) cells and a varying number of RB half adder (RBHA) cells depending on where it is located. The proposed RBMPPG-2 can be applied to any bit RB multipliers with a reduction of a RBPP accumulation stage compared with conventional designs. Although the delay of RMPPG-2 increases by 1-stage of TG delay, the delay of one RBPP accumulation stage is significantly larger than a 1-stage TG delay. Therefore, the delay of the entire multiplier is reduced. The improved complexity, delay and power consumption are very attractive for the proposed design. The multiplier consists of the proposed RBMPPG-2, three RBPP accumulation stages, and one RB-NB converter. Eight RBBE-2 blocks generate the RBPP they are summed up by the RBPP reduction tree that has three RBPP accumulation stages. Each RBPP accumulation block contains RB full adders (RBFAs) and half adders (RBHAs).

IV. MOTIVATION: EFFECT OF HARDWARE APPROXIMATION ON VIDEO QUALITY

Images and videos differ in a variety of properties, such as color, resolution, brightness, contrast, saturation, blur, format, and so on. Thus, a naive static approximation technique, which provides satisfactory viewing quality for some specific types of
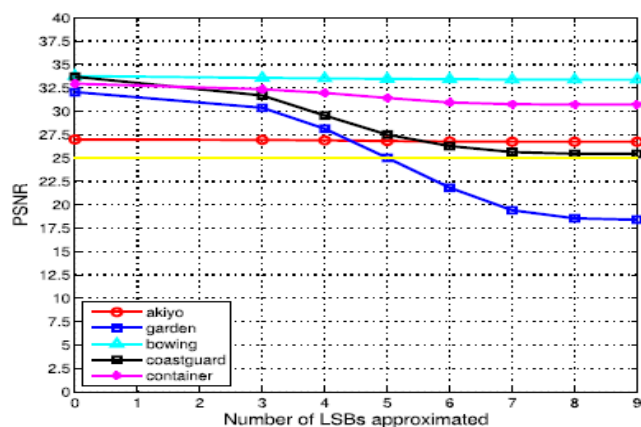
videos, will fail to give adequate quality for some others. In that case, the viewing experience is significantly worsened if the approximate mode is not customized for the present type of video being watched.
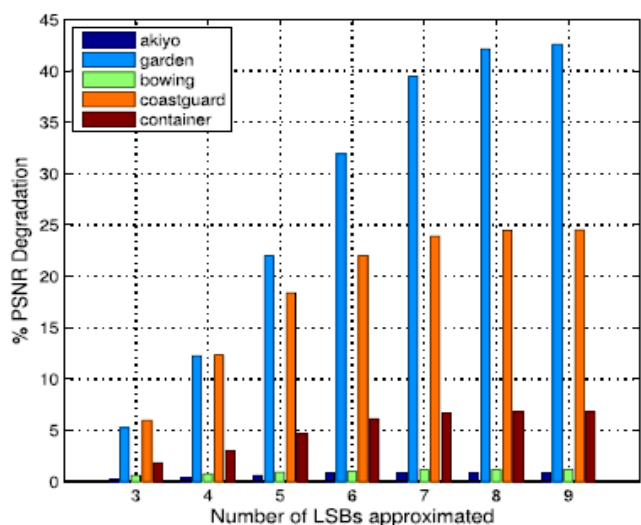
This is not possible for fixed hardware, and therefore a need arises for reconfiguring the architecture based on the characteristics of the video being viewed. To support this claim, Figs. 2 and 3 present the PSNR variation of different videos when encoded using an

Fig. 2. Variation of absolute PSNR with number of LSBs approximated.

Fig. 3. %PSNR degradation with number of LSBs approximated.

MPEG encoder that used a fixed approximation technique. As an example of an approximation mode, we have chosen approximation mode 5 from [10] for implementing the fixed approximation hardware. We replaced all the adders/ subtractors in the ME and the DCT blocks with approximate versions.
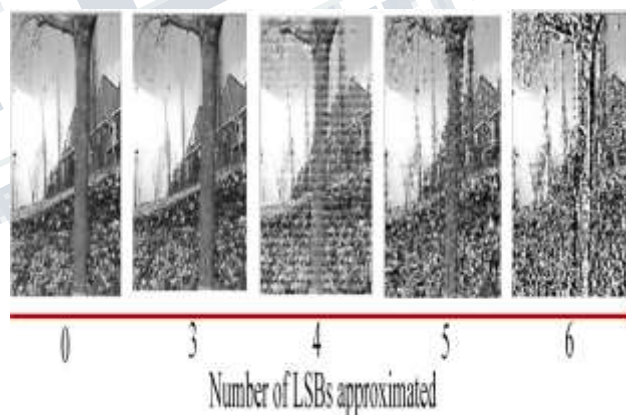
Fig. 2 gives the absolute PSNR and Fig. 3 gives the percentage degradation in PSNR (compared with an accurate version of the MPEG encoder) for the five randomly chosen video benchmarks (*Akiyo, Garden, Bowing, Coastguard,* and *Container*) [18] when the number of bits to be approximated (also termed the DA) is varied. In this case, we have approximated the least significant bits (LSBs) of the adders. There are multiple ways of setting the hard threshold for the output PSNR, which determines whether the quality of a video is acceptable or not. For the sake of simplicity, it is assumed that either the absolute PSNR or the percentage change in PSNR serves as a faithful yardstick for evaluating the quality of videos outputted by the approximated MPEG encoder. In this regard, we define two metrics: 1) absolute error threshold (AET) and 2) relative error margin (REM) to demarcate between the acceptable and unacceptable videos.

Fig. 4. Output quality of benchmark Garden for different number of LSBs approximated.

AET is defined as a fixed absolute PSNR value below which the video is termed to be unacceptable. REM is expressed as a certain percentage of the base PSNR value, which gives the maximum permissible degradation in output PSNR. Either of them can be utilized for judging the merit of a video. In the case AET is fixed at 25 (evaluated by a subjective assessment of the video qualities), Fig. 2 shows that for DAs of 6 or more,

Garden violates the quality bound while others give satisfactory results. The results are worse when REM is selected as the metric for measurement. For REM of 5%, both *Garden* and *Coastguard* fail for all DAs ranging from 3 to 9, while *Container* fails for degrees of 6 or more. In case REM is further relaxed to 10%, quality of *Container* remains within the bound; however, *Garden* and *Coastguard* still fail for DAs of 4 or more.

Thus, these two graphs clearly establish the fact that an MPEG encoder with fixed approximation architectures is unable to guarantee amenable quality of video across different inputs. Fig. 4 shows snapshots of the output quality of the video *Garden* when subjected to different DAs. It clearly demonstrates the gradual degradation in output video quality with increasing DA as also verified by the PSNR values in Figs. 2 and 3.

It is obvious that lowering the AET and increasing the REM provide more opportunity for larger power savings, since they allow for an increased level of hardware approximation. In order to get the maximum power savings, it is required that DA be closest to the maximum allowable for the given AET or REM. For example, when REM is 10%, the degree should be 3 for both *Garden* and *Container*; however, it can extend up to 9 or more for the other videos.

A straightforward way to ensure this is to dynamically vary the DA of the adders and subtractors across computational cycles depending upon the characteristics of the input video. Our work automates this process with the help of the heuristics described in Section V-C. As a result, designers only need to specify the required quality bound (e.g., in %PSNR degradation) instead of fixing the DA statically. Our approach computes the best DA dynamically such that the quality is within the acceptable limits and the power savings are maximized.

V. PROPOSED ARCHITECTURE

This section describes the different steps ensued in constructing our proposed reconfigurable architecture and how it was embedded within the MPEG encoder. A. Reconfigurable Adder/Subtractor Blocks.

Dynamic variation of the DA can be done when each of the adder/subtractor blocks is equipped with one or more of its approximate copies and it is able to switch between them as per

requirement. This reconfigurable architecture can include any approximate version of the adders/subtractors. As a reference, Gupta et al. [9], [10] proposed six different kinds of approximate circuits for adders.
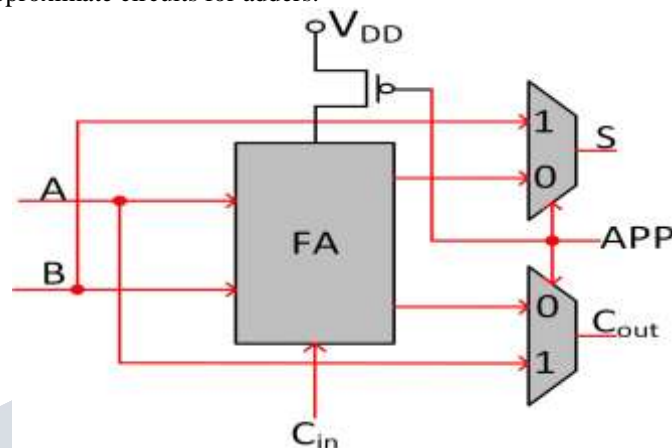

Fig. 5. 1-bit DMFA.

TABLE IPOWER CONSUMPTION OF DIFFERENT DMFA MODES

Original FA (μW)	DMFA Accurate Mode (μW)	DMFA Approximate Mode (μW)
1.53	1.74	0.01

However, it also needs to be ensured that the additional area overheads required for constructing the reconfigurable approximate circuits are minimal with sufficiently large power savings. As examples, we have chosen the two most naive methods presented in [2], namely, truncation and approximation 5, for approximating the adder/subtractor blocks. The latter one can also be conceptualized as an enhanced version of truncation as it just relays the two 1-bit inputs, one as Sum and the other as Carry Out (Choice 2). In case A, B, and Cin are the 1-bit inputs to the full adder (FA), then the outputs are Sum = B and Cout = A. The resultant truth-table [10] shows that the outputs are correct for more than half of all input combinations, thus proving to be a better approximation mode than truncation. The proposed scheme replaces each FA cell of the adders/subtractors with a dual-mode FA (DMFA) cell (Fig. 5) in which each FA cell can operate either in fully accurate or in some approximation mode depending on the state of the control signal APP. A logic high value of the APP signal denotes that the DMFA is operating in the approximate mode. We term these adders/subtractors as

RABs. It is important to note that the FA cell is power-gated when operating in the approximate mode. Synthesis and evaluation of power consumption of a 16-bit RCA were performed in Synopsys Design and Power Compiler and the corresponding results are described in Table I. Our experiments have shown a negligible difference in the power consumption of DMFA when operated in either of the two approximation modes. Hence, without any loss of generality, approximation 5 was chosen for its higher probability of giving the correct output result than truncation, which invariably outputs 0 irrespective of the input.
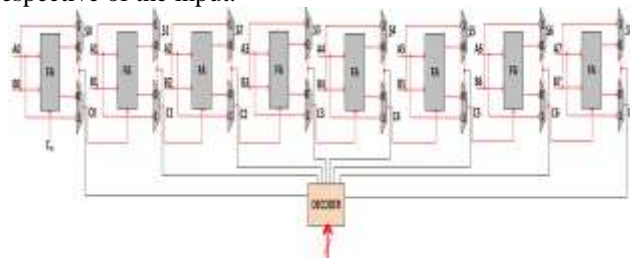
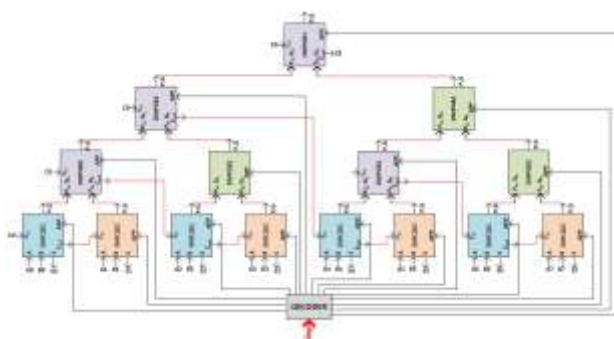*Fig. 6. 8-bit reconfigurable RCA block.*

Fig. 5 shows the logic block diagram of the DMFA cell, which replaces the constituent FA cells of an 8-bit RCA, as shown in Fig. 6. In addition, it also consists of the approximation controller for generating the appropriate select signals for the multiplexers. A multimode FA cell would provide even a better alternative to the DMFA from the point of controlling the approximation magnitude. However, it also increases the complexity of the decoder block used for asserting the right select signals to the multiplexers as well as the logic overhead for the multiplexers themselves. This undermines the primary objective as most of the power savings that we get from approximating the bits are lost. Instead, the two-mode decoder and the 2:1 multiplexers have negligible overhead and also provide sufficient command over the approximation degree.

1) DMFA Overhead: The power gating transistor and the multiplexers of the DMFA are designed to incur the least possible overhead. Our experiments show that switching power of the CMOS transistors contributes toward most of the total power consumption of the FA and DMFA blocks. Table I presents the power consumption of FA and DMFA for different modes obtained by exhaustive simulation in Synopsys NanoSim. It shows that the power increases by 0.21 µW when we operate DMFA in accurate mode as compared with the original FA block.

This difference in power can be attributed mainly to the increase in load capacitance of the FA block due to the addition of the input capacitance of the interfaced multiplexers. A small portion of the total power is contributed by the additional switching of the multiplexers. Table I also shows that the power consumed during DMFA approximate mode is almost negligible when compared with the accurate mode, which is due to the power gating of the FA block by the pMOS transistor, as shown in Fig. 5. Reduction in the input switching activity of the multiplexers is also a secondary cause for this small amount of power. The additional overhead due to switching of the power gating transistor can be neglected, since its switching activity is very small due to the nature of our switching algorithms. This is mainly due to the spatial and temporal locality of the pixel values across consecutive frames. The concept of RAB can also be extended to other adder architectures as well. Adder architectures, such as CBA and CSA, which also contain FA as the fundamental building block, can be made accuracy configurable by direct substitution of the FAs with DMFAs. Other varieties, like CLA and tree adders, use different types of carry propagate and generate blocks as their basic building units, and hence require some additional modifications to function as RABs. As an example, we implemented a 16-bit CLA consisting of four different types of basic blocks (Fig. 8) depending upon the presence of sum (S), Cout, carry propagation (P), and carry generation (G) at different levels. We address the basic blocks present at the first (or lowermost) level of a CLA, which have inputs coming in directly, as carry lookahead blocks, CLB1 and CLB2. The difference among them being that CLB1 produces an additional Cout signal compared with CLB2. Their corresponding dual-mode versions, DMCLB1 and DMCLB2, have both S and P approximated by input operand B and both Cout and G approximated by input operand A, as shown in Fig. 7. The basic blocks present at the higher levels of CLA hierarchy are denoted as propagate and generate blocks, PGB1 and PGB2. In this case, PGB1 produces an extra Cout output as compared with PGB2. As shown in Fig. 7, the configurable dual-mode versions, DMPGB1 and DMPGB2, use inputs PA and GB as approximations for outputs P and G, respectively, when operating in the approximate mode. These approximations were selected empirically ensuring that the ratio of the probability of correct output to the additional circuit overhead for each of the blocks is large.

Fig. 7. 8-bit reconfigurable CLA block.

TABLE II
DUAL-MODE BLOCK OUTPUTS FOR ACCURATE AND APPROXIMATE MODES

Basic Block (*adder type*)	Outputs for APP = 0 (accurate mode)	Outputs for APP = 1 (approximate mode)
DMFA (*RCA, CBA, CSA*)	$S = A \oplus B \oplus C_{in}$ $C_{out} = AB + BC_{in} + AC_{in}$	$S = B$ $C_{out} = A$
DMCLB1 (*CLA*)	$P = A \oplus B$ $G = AB$ $S = P \oplus C_{in}$ $C_{out} = G + PC_{in}$	$P = B$ $G = A$ $S = B$ $C_{out} = A$
DMCLB2 (*CLA*)	$P = A \oplus B$ $G = AB$ $S = P \oplus C_{in}$	$P = B$ $G = A$ $S = B$
DMPGB1 (*CLA*)	$P = P_A P_B$ $G = G_B + G_A P_B$ $C_{out} = G + PC_{in}$	$P = P_A$ $G = G_B$ $C_{out} = G + PC_{in}$
DMPGB2 (*CLA*)	$P = P_A P_B$ $G = G_B + G_A P_B$	$P = P_A$ $G = G_B$

Table II summarizes the outputs of each of the dual-mode blocks when operating in either accurate or approximate mode. For a reconfigurable CLA, DMCLB1 and DMCLB2 blocks are approximated in accordance with the DA. However, the DMPGB1 and DMPGB2 blocks are approximated only when each and every DMCLB1, DMCLB2, DMPGB1, and DMPGB2 block, which belongs to the transitive fan-in cones of the concerned block, is approximated.

Otherwise, the block is operated in the accurate mode. For example, any DMPGB block at the second level of CLA can be made to operate in approximate mode, if and only if, both of its constituent DMCLB1 and DMCLB2 blocks are operating in the approximate mode. Similar protocol is ensued for the blocks residing at higher levels of the tree, where each

DMPGB block can be approximated only when both of its constituent DMPGB1 and DMPGB2 blocks are approximated. This architecture can be easily extrapolated to other similar type CLAs, such as Kogge–Stone, Brent–Kung, Manchester-carry chain, and so on. Figs. 9–12 show a comparative study of the power consumption of the different types of adders when the DA is varied. In particular, the figures denote the normalized power consumption of the different types of RABs when the number of bits approximated is varied. An interesting observation for CSA is that approximating its MSBs gives greater power savings than LSB approximation per bit. This can be attributed to the architecture [19] of the carry save adders, where approximating each bit in the MSB results in power gating of two FAs compared with one FA when the LSBs are approximated. The charts also depict that actual power savings are initiated when the DA is equal to or above 5. This is the point where the savings due to approximation surpasses the overhead incurred due to the additional multiplexers, power gating transistors, and controller. The inherent error resilience shown by the ME and the small inputs to the DCT block provide sufficient opportunities for achieving a high DA (much greater than 5) and thereby high power savings.

REFERENCES

[1] M. Elgamel, A. M. Shams, and M. A. Bayoumi, "A comparative analysis for low power motion estimation VLSI architectures," in Proc. IEEE Workshop Signal Process. Syst. (SiPS), Oct. 2000, pp. 149–158.

[2] F. Dufaux and F. Moscheni, "Motion estimation techniques for digital TV: A review and a new contribution," Proc. IEEE, vol. 83, no. 6, pp. 858–876, Jun. 1995.

[3] I. S. Chong and A. Ortega, "Dynamic voltage scaling algorithms for power constrained motion estimation," in Proc. IEEE Int. Conf. Acoust., Speech, Signal Process. (ICASSP), vol. 2. Apr. 2007, pp. II-101–II-104.

[4] I. S. Chong and A. Ortega, "Power efficient motion estimation using multiple imprecise metric computations," in Proc. IEEE Int. Conf. Multimedia Expo, Jul. 2007, pp. 2046–2049.

[5] D. Mohapatra, G. Karakonstantis, and K. Roy, "Significance driven computation: A voltage-scalable, variation-aware, quality-tuning motion estimator," in Proc. 14th ACM/IEEE Int. Symp. Low Power Electron. Design (ISLPED), 2009, pp. 195–200.

[6] J. George, B. Marr, B. E. S. Akgul, and K. V.

Palem, "Probabilistic arithmetic and energy efficient embedded signal processing," in Proc. Int. Conf. Compil., Archit., Synth. Embedded Syst. (CASES), 2006, pp. 158–168.

[7] D. Shin and S. K. Gupta, "A re-design technique for datapath modules in error tolerant applications," in Proc. 17th Asian Test Symp. (ATS), 2008, pp. 431–437.

[8] S. Venkataramani, A. Sabne, V. Kozhikkottu, K. Roy, and A. Raghunathan, "SALSA: Systematic logic synthesis of approximate circuits," in Proc. 49th Annu. Design Autom. Conf. (DAC), Jun. 2012, pp. 796–801.

[9] V. Gupta, D. Mohapatra, S. P. Park, A. Raghunathan, and K. Roy, "IMPACT: IMPrecise adders for low-power approximate computing," in Proc. 17th IEEE/ACM Int. Symp. Low-Power Electron. Design (ISLPED), Aug. 2011, pp. 409–414.

[10] V. Gupta, D. Mohapatra, A. Raghunathan, and K. Roy, "Lowpower digital signal processing using approximate adders," IEEE Trans. Comput.-Aided Design Integr. Circuits Syst., vol. 32, no. 1, pp. 124–137, Jan. 2013.

[11] V. G. Moshnyaga, K. Inoue, and M. Fukagawa, "Reducing energy consumption of video memory by bit-width compression," in Proc. Int. Symp. Low Power Electron. Design (ISLPED), 2002, pp. 142–147.

[12] Z. He and M. L. Liou, "Reducing hardware complexity of motion estimation algorithms using truncated pixels," in Proc. IEEE Int. Symp. Circuits Syst. (ISCAS), vol. 4. Jun. 1997, pp. 2809–2812.

[13] Z.-L. He, C.-Y. Tsui, K.-K. Chan, and M. L. Liou, "Low-power VLSI design for motion estimation using adaptive pixel truncation," IEEE Trans. Circuits Syst. Video Technol., vol. 10, no. 5, pp. 669–678, Aug. 2000.

[14] A. Raha, H. Jayakumar, and V. Raghunathan, "A power efficient video encoder using reconfigurable approximate arithmetic units," in Proc. 27th Int. Conf. VLSI Design, 13th Int. Conf. Embedded Syst., Jan. 2014, pp. 324–329.

[15] P. M. Kuhn, Algorithms, Complexity Analysis and VLSI Architectures for MPEG-4 Motion Estimation, 1st ed. Norwell, MA, USA: Kluwer, 1999.

[16] K. Seshadrinathan, R. Soundararajan, A. C. Bovik, and L. K. Cormack, "Study of subjective and objective quality assessment of video," IEEE Trans. Image Process., vol. 19, no. 6, pp. 1427–1441, Jun. 2010.

[17] S. Winkler, "Video quality measurement standards—Current status and trends," in Proc. 7th Int. Conf. Inf., Commun., Signal Process. (ICICS), Dec. 2009, pp. 1–5.