International Journal of Research
Available at https://edupediapublications.org/journals

e-ISSN: 2348-6848
p-ISSN: 2348-795X
Volume 05 Issue 04
February 2018

Input Based Dynamic Reconfigurable Approximation Architecture for MPEG Encoders

Nagam Lavanya sree & N. Anil

[1]M.Tech Scholar, DECS, Chaitanya Institute of Science and Technology, Kakinada,Andhra Pradesh,India, mail: lavanyasree269@gmail.com

[2]Assistant Professor,Dept of ECE, Chaitanya Institute of Science and Technology, Kakinada,Andhra Pradesh,India, mail: anilwincent@gmail.com

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. This paper addresses this issue by proposing a reconfigurable approximate architecture for MPEG encoders that optimizes delay by proposing 64 bit reconfigurable CLA. Toward this end, we design 64 bit 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. Experimental results show that our approach will occupy only 506 LUTs of SPARTAN 3E FPGA over total 9312 LUTs with a delay of 24.047ns.*

Keywords: Approximate circuits, approximate computing, low power design, quality configurable

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.

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-

e-ISSN: 2348-6848
p-ISSN: 2348-795X
Volume 05 Issue 04
February 2018

4 standards, videos can be squeezed to very small sizes. MPEG uses both inter frame and intra frame encoding for video compression. Intra frame 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. AnI-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 non encoded frame is divided into smaller.

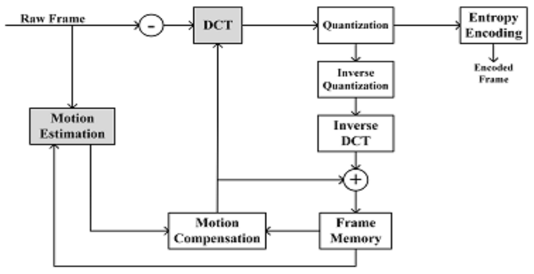


Fig 1.MPEG encoder block diagram.

macro blocks (MBs), typically 16×16 pixels. 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. The different steps involved in performing MPEG compression are shown in Fig. 1

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. 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.

Approximate adders:

Adders are utilized for calculating the addition (or sum) of two binary numbers. Two common types of adders are the ripple-carry adder (RCA) and the carry look ahead adder (CLA). In an n-bit RCA, n 1-bit full adders (FAs) are cascaded; the carry of each FA is propagated to the next FA, thus the delay of RCA grows in proportion to n (or O(n)). An n-bit CLA consists of n SPGs, which operate in parallel to produce the sum, generate (gi = ai bi) and propagate (pi = ai + bi) signals, and connected to a carry look ahead generator. For CLA, all carries are generated directly by the carry look ahead generator using only the generate and propagate signals, so the delay of CLA is logarithmic in n (or O(log(n))), thus significantly shorter than that of RCA. However, CLA requires larger circuit area and higher power dissipation. The carry look ahead generator becomes very complex for large n. The area complexity of CLA is O(nlog(n)) when the fan-in and fan-out of the constituent gates are fixed. Many approximation schemes have been proposed by reducing the critical path and hardware complexity of the accurate adder. An early methodology is based on a speculative operation. In an n-bit speculative adder, each sum bit is predicted by its previous k less significant bits(LSBs) (k < n). A speculative design makes an adder significantly faster than the conventional design. Segmented adders are proposed. An n-bit segmented adder is implemented by several smaller adders operating in parallel. Hence, the carry propagation chain is truncated into shorter segments. Segmentation is also utilized, but their carry inputs for each sub-adder are selected differently. This type of adder is referred to as a carry select adder. Another method for reducing the critical path delay and power dissipation of a conventional adder is by approximating the full adder; the approximate adder is usually applied to the LSBs of an accurate adder. In the sequel, the approximate adders are divided into four categories.

II.LITERATURE SURVEY

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, voltage over-scaling, and imprecise computation of metrics. 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 over scaling, intelligent logic manipulation, and circuit simplification using don’t care-based optimization techniques. The methods 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 re synthesizing the entire circuit. There also exist instances of approximations introduced in an MPEG encoder. Most of them exploit the inherent error resilience of the motion estimation (ME) algorithm, which results in minor quality degradation. For example, Moshnyagaet al. use a bit width compression technique to reduce power consumption of video frame memory. He and Liou and Heet al. use bit truncation to introduce approximations in the ME block of an MPEG encoder. An adaptive bit masking method is proposed, 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. 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.

III.PROPOSED SYSTEM

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, Guptaet al. proposed six different kinds of approximate circuits for adders. 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, 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 caseA, B,andCin are the 1-bit inputs to the full adder (FA), then the outputs are Sum=Band Cout=A. The resultant truth-table 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.

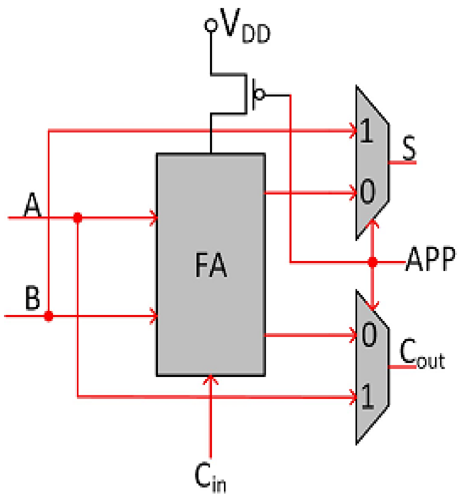


Fig 2:1-bit DMFA

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.

TABLE-I

POWER CONSUMPTION OF DIFFERENT DMFA MODES

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

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. 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.3.2. In addition, it also consists of the approximation controller for generating the appropriate select signals for the multiplexers.

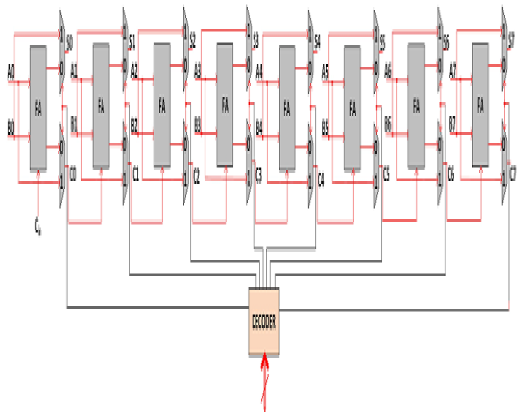


Fig.3. 8-bit reconfigurable RCA block

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.

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 look ahead 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.4. 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.

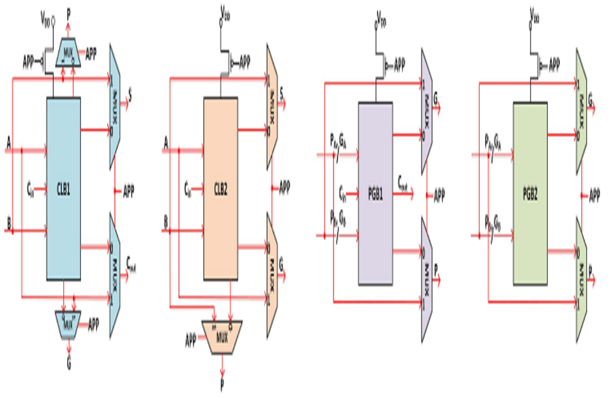


Fig.4. 1-bit dual-mode carry propagate generate blocks.

As shown in Fig.4, 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.

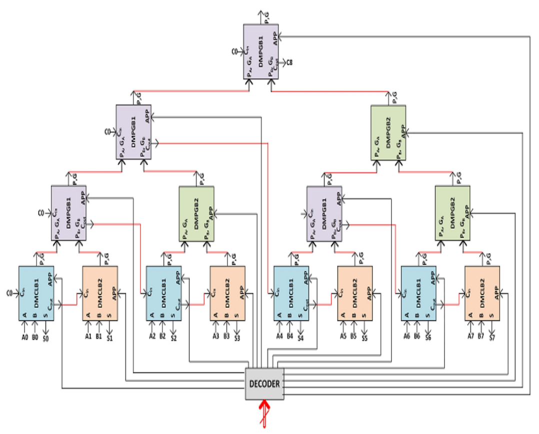


Fig.5. 8-bit reconfigurable CLA block.

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. Table II summarizes the outputs of each of the dual-mode blocks when operating in either accurate or approximate mode.

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)	$S = A \oplus B \oplus C_{in}$	$S = B$
	$C_{out} = AB + BC_{in} + AC_{in}$	$C_{out} = A$
DMCLB1 (CLA)	$P = A \oplus B$	$P = B$
	$G = AB$	$G = A$
	$S = P \oplus C_{in}$	$S = B$
	$C_{out} = G + PC_{in}$	$C_{out} = A$
DMCLB2 (CLA)	$P = A \oplus B$	$P = B$
	$G = AB$	$G = A$
	$S = P \oplus C_{in}$	$S = B$
DMPGB1 (CLA)	$P = P_A P_B$	$P = P_A$
	$G = G_B + G_A P_B$	$G = G_B$
	$C_{out} = G + PC_{in}$	$C_{out} = G + PC_{in}$
DMPGB2 (CLA)	$P = P_A P_B$	$P = P_A$
	$G = G_B + G_A P_B$	$G = G_B$

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

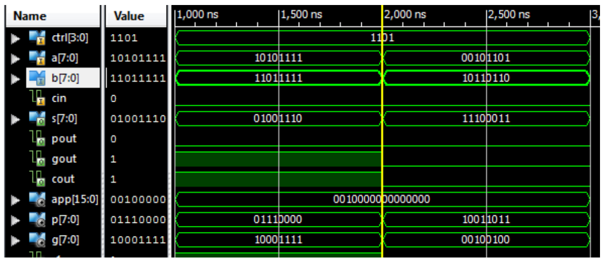
DMPGB1 and DMPGB2 blocks are approximated. This architecture can be easily extrapolated to other similar type CLAs, such as Kogge–Stone, Brent–Kung, Manchester-carry chain, and so on.

IV.Results

The written Verilog HDL Modules have successfully simulated and verified using Isim Simulator and synthesized using Xilinxise13.2.

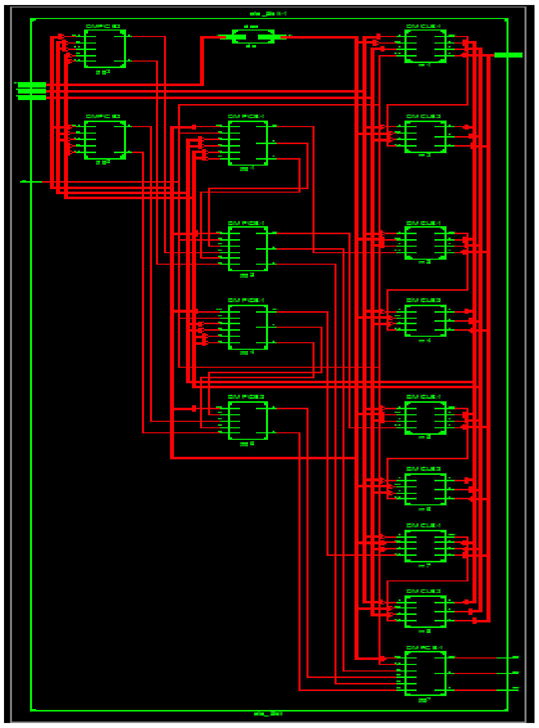
Proposed:

Simulation Results

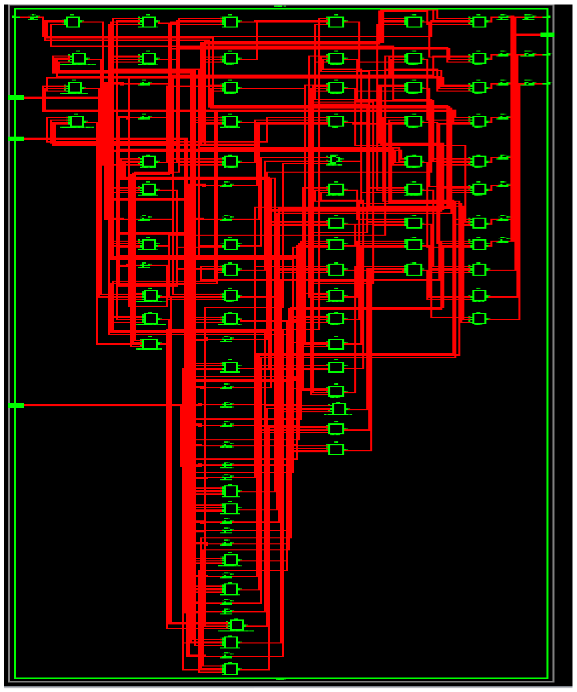


Synthesis results

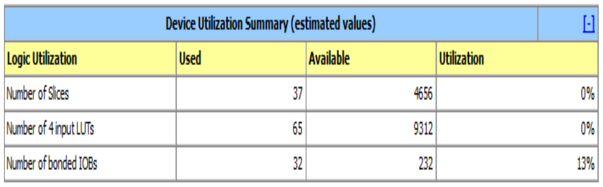
RTL schematic



Technology Schematic:



Design summary



Device Utilization Summary (estimated values)			
Logic Utilization	Used	Available	Utilization
Number of Slices	37	4656	0%
Number of 4 input LUTs	65	9312	0%
Number of bonded IOBs	32	232	13%

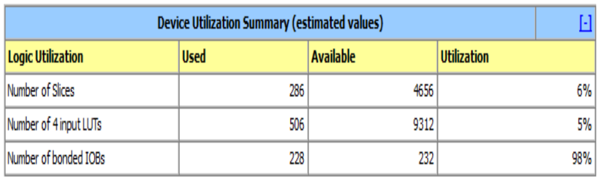
Timing Report

```
Timing Summary:
---------------
Speed Grade: -5

   Minimum period: No path found
   Minimum input arrival time before clock: No path found
   Maximum output required time after clock: No path found
   Maximum combinational path delay: 13.835ns
```

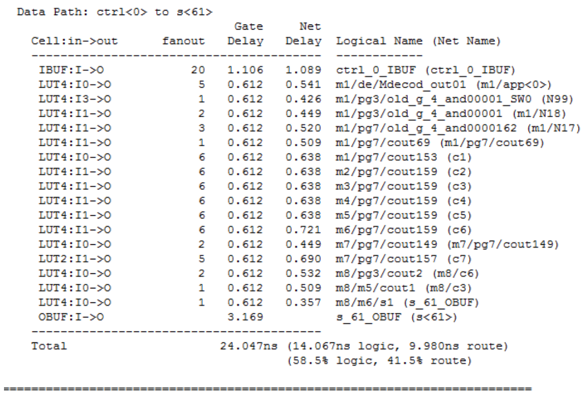
Extension:

Design summary



Device Utilization Summary (estimated values)			[-]
Logic Utilization	Used	Available	Utilization
Number of Slices	286	4656	6%
Number of 4 input LUTs	506	9312	5%
Number of bonded IOBs	228	232	98%

Timing Report



```
Data Path: ctrl<0> to s<61>
                                Gate     Net
  Cell:in->out      fanout   Delay   Delay  Logical Name (Net Name)
  ----------------------------------------  ------------
   IBUF:I->O            20   1.106   1.089  ctrl_0_IBUF (ctrl_0_IBUF)
   LUT4:I0->O            5   0.612   0.541  m1/de/Mdecod_out01 (m1/app<0>)
   LUT4:I3->O            1   0.612   0.426  m1/pg3/old_g_4_and00001_SW0 (N99)
   LUT4:I1->O            2   0.612   0.449  m1/pg3/old_g_4_and00001 (m1/N18)
   LUT4:I1->O            3   0.612   0.520  m1/pg7/old_g_4_and0000162 (m1/N17)
   LUT4:I1->O            1   0.612   0.509  m1/pg7/cout69 (m1/pg7/cout69)
   LUT4:I0->O            6   0.612   0.638  m1/pg7/cout153 (c1)
   LUT4:I1->O            6   0.612   0.638  m2/pg7/cout159 (c2)
   LUT4:I1->O            6   0.612   0.638  m3/pg7/cout159 (c3)
   LUT4:I1->O            6   0.612   0.638  m4/pg7/cout159 (c4)
   LUT4:I1->O            6   0.612   0.638  m5/pg7/cout159 (c5)
   LUT4:I1->O            6   0.612   0.721  m6/pg7/cout159 (c6)
   LUT4:I0->O            2   0.612   0.449  m7/pg7/cout149 (m7/pg7/cout149)
   LUT2:I1->O            5   0.612   0.690  m7/pg7/cout157 (c7)
   LUT4:I0->O            2   0.612   0.532  m8/pg3/cout2 (m8/c6)
   LUT4:I0->O            1   0.612   0.509  m8/m5/cout1 (m8/c3)
   LUT4:I0->O            1   0.612   0.357  m8/m6/s1 (s_61_OBUF)
   OBUF:I->O                 3.169          s_61_OBUF (s<61>)
  ----------------------------------------
  Total                     24.047ns (14.067ns logic, 9.980ns route)
                                     (58.5% logic, 41.5% route)

=========================================================================
```

CONCLUSION

This paper proposed a 64 bit reconfigurable approximate architecture for the CLA that optimize delay while maintaining output quality across different input videos. The proposed architecture is based on the concept of dynamically reconfiguring the level of approximation in the hardware based on the input characteristics. It requires the user to specify only the overall minimum quality for videos instead of having to decide the level of hardware approximation. Our Experimental results show that our approach will occupy only 506 LUTs of SPARTAN 3E FPGA over total 9312 LUTs with a delay of 24.047ns whereas for 8 bit CLA the delay will be 13.8ns.

REFERENCE

[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 digitalTV: 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,” inProc. 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,” inProc. 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,” inProc.
Int. Conf. Compil., Archit., Synth. Embedded Syst. (CASES), 2006, pp. 158–168.

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