Int. J. Adv. Res. Sci. Technol. Volume 4, Issue 1, 2015, pp.267-270.

IJARST

International Journal of Advanced Research in Science and Technology

journal homepage: www.ijarst.com

ISSN 2319 – 1783 (Print)

ISSN 2320 – 1126 (Online)

Optimized EDDR Design for Motion estimation applications

Pavan Srinivas* and D. P. Raju

Department of ECE, Koushik college of Engg., Visakhapatnam, Andhra Pradesh, India.

****Corresponding Author's Email: pavan.burle@gmail.com***

A R T I C L E I N F O

Article history:

Received 20 Nov. 2014

Accepted 11 Dec. 2014

Available online 20 Jan. 2015

Keywords:

Processing Element (PE), Error Detection and Data Recovery (EDDR),
Motion estimation (ME), Residue-and-quotient (RQ) code,
Built In Self Test (BIST).

A B S T R A C T

Given the critical role of motion estimation (ME) in a video coder, testing such a module is of priority concern. While focusing on the testing of ME in a video coding system, this work presents an error detection and data recovery (EDDR) design, based on the residue-and-quotient (RQ) code, to embed into ME for video coding testing applications. An error in processing elements (PEs), i.e. key components of a ME, can be detected and recovered effectively by using the proposed EDDR design. Experimental results indicate that the proposed Error Detection and Data Recovery design for motion estimation testing can detect errors and recover data with an acceptable area overhead and timing penalty. The functional verification and synthesis can be done by using any version of Xilinx ISE.

Introduction:

The new Joint Video Team (JVT) video coding standard has garnered increased attention recently. Generally, motion estimation computing array (MECA) performs up to 50% of computations in the entire video coding system, and is typically considered the computationally most important part of video coding systems. Thus, integrating the MECA into a system-on-chip (SOC) design has become increasingly important for video coding applications.

Although advances in VLSI technology allow integration of a large number of processing elements (PEs) in an MECA into an SOC, this increases the logic-per-pin ratio, there by significantly decreasing the efficiency of chip logic testing. For a commercial chip, a video coding system must introduce design for testability (DFT), especially in an MECA.

The objective of DFT is to increase the ease with which a device can be tested to guarantee high system reliability. Many DFT approaches have been developed. These approaches can be divided into three categories: ad hoc (problem oriented), structured, and built-in self-test (BIST). Among these techniques, BIST has an obvious advantage in that expensive test equipment is not needed and tests are low cost.

Generally, motion estimation computing array (MECA) performs up to 50% of computations in the entire video coding system (VCS).In VCS, Video data needs to be compressed before storage and transmission, complex algorithms are required to eliminate the redundancy, extracting the redundant information. Motion Estimation (ME) is the process of creating motion vectors to track the motion of objects within video footage. It is an essential part of many compression standards and is a crucial component of the H.264 video compression standard .In particular ME can consist of over 40% of the total computation.

Digital Video Compression:

Video compression is achieved on two separate fronts by eliminating spatial redundancies and temporal redundancies from video signals. Removing spatial redundancies involves the task of removing video information that is consistently repeated within certain areas of a single frame. For example a frame shot of a blue sky will have a consistent shade of blue across the entire frame. This information can be compressed through the use of various discrete cosine transformations that map a given image in terms of its light or color intensities. This paves the way for spatial compression by only capturing the distinct intensities, instead of the spread of intensities over the entire frame. Since compression through removing spatial redundancies does not involve the use of motion estimation, this topic is not examined further.

Among many other new features and enhancements, the most notable features of the H.264 standard for this work are its ability to achieve a finer granularity of

motion estimation and its ability to capture periodic motion.

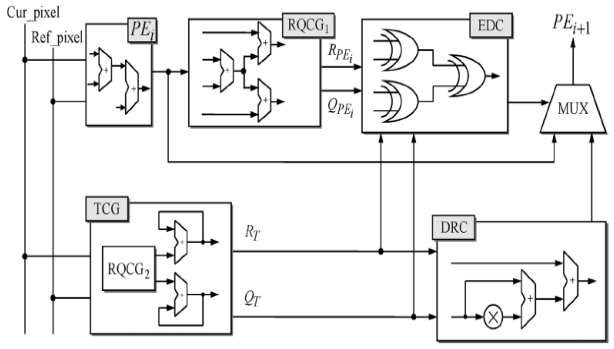

Fig. 1: Example of the self-detection/correction operations

The self-detection and self-correction operations (Fig.4.01) are simply described as follows. First, the input data of Current Pixel and Reference pixel for a specific PE_i in the MECA are sent to the test code generator (TCG) to generate the corresponding test codes. Second, the test codes from the TCG and output data from the specific PE_i are detected and verified in detector and selector (DAS) circuits to determine whether the specific PE_i has an error.

Processing Element (PE):

Processing element calculates the sum of absolute differences (SAD) between current pixels and reference pixels. Generally, a PE is made up of two adders (an 8-bit adder and a 12-bit adder) and accumulator.

The SAD is given by

$$SAD = \sum_{i=0}^{N-1}\sum_{j=0}^{N-1}|c(i,j) - r(i,j)| \qquad (1)$$

Where c (i, j) and r (i, j) are the current pixel and reference pixel. The 16 pixels absolute difference is given to adder unit to perform SAD.

Coder:

The following definitions, based on the bi-residues codes, are applied to verify the feasibility of the two coders in the TCG

Definition 1: $|N_1 + N_2|\varphi = ||N_1|_\varphi + |N_2|_\varphi|\varphi$ (2)

Definition 2: *Let* $N_j = n_1 + n_2 + \ldots\ldots\ldots + n_j$

Then

$$|N_J|\ \varphi = \left\| n_1 \right|_\varphi + \left| n_2 \right|_\varphi + \ldots\ldots + \left| n_j \right|_\varphi|\varphi \qquad (3)$$

In this paper we are using two Coder modules because bi residue Codes are used for calculation of SAD. And the Phi1 (φ1) and phi2 (φ2) values are selected by satisfying the conditions. i.e.

A = 2^a -1 and B = 2^b -1 such that GCD (a, b) = 1.

Detector:

Detector module will detect whether there is an error in output of PE i.e. SAD. Here the Error is calculated. The output of PE and theoretically calculated SAD will be subtracted which is given as e = SAD'-SAD.

Selector:

Selector takes the output of the processing element as an input. Another input to the selector is the output of the detector. If the Detector block detects any error in the PE output then the Selector block will give the PE output to Syndrome Decoder to detect in which bit position there is an error and also to the Corrector block to correct the single bit error.

Syndrome Decoder:

This module decodes the syndrome values which specify the error in SAD. In fiure first gate takes inputs from coder of PE and second gate takes input from TCG module. Syndromes can be expressed as

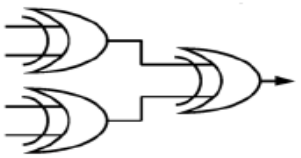

Fig. 2: Error detection circuit.

Design and Implementation:

PE Array architecture:

PE is a module that calculates the absolute difference between the pixel of the reference block and the pixel of the current block. Figure shows the architecture of PE Array 4x4. To enable the reference data shifting to top, bottom, right or left in PE Array 4x4, each PE is connected to the PE of top, bottom and right or left one. This structure generates SAD 4x1 by accumulating the absolute difference of each PE. Furthermore, SAD 4x4 is generated by accumulating generated SAD 4x1.

The traditional PE and the proposed one are shown in following Figures.

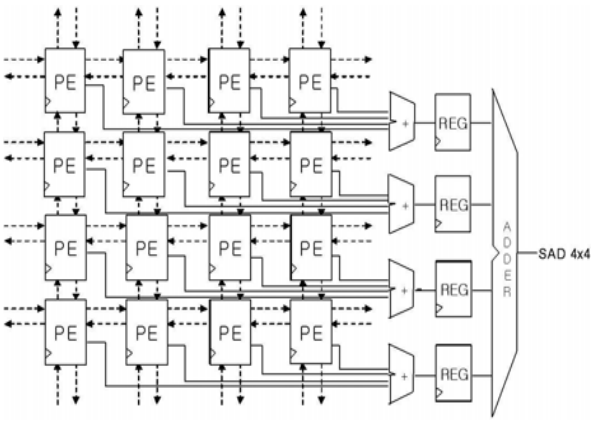

Fig. 3: PE Array 4x4 architecture.

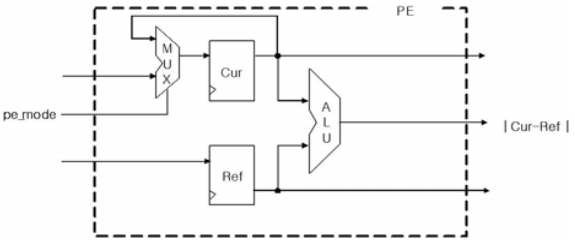

Fig. 4: Traditional PE architecture.

In figure Reference pixel data need input from four directions as well as output to four directions. Therefore, four input ports and four output ports are prepared in each PE. Each PE is connected with surrounding PEs: "from top" connects to "to bottom", "from bottom" connects to "to top", "from left" connects to "to right", and "from right" connects to "to left". An example when "from top" is selected by the multiplexer. The reference data from the output port "to bottom" of upper PEs is shifted to bottom PEs and the search position is shifted. In this way, each I/O port of PE enables the shift of the reference pixel data by selecting the input of reference pixel data using multiplexer.

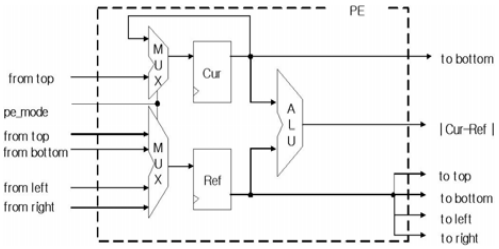

Fig. 5: Proposed PE architecture.

SAD Modules:

There are 16 SAD modules in the architecture, where each one is in charge of the SAD computation of one primitive 4x4 sub-block in parallel. In the SAD module, there are 16 absolute difference computing unit processing the 16 pair of pixels in parallel, and then the 16 absolute residues are fed into the adder unit to get one 4x4SAD. Processing element calculates the sum of absolute differences between current pixels and reference pixels. The 16 pixels absolute difference is given to adder unit to perform SAD.

Adder Unit:

The absolute differences of 16 pixels each is given to the adder unit which performs the SAD calculation. The architecture of adder tree is shown below, which consists of 3-2 compressor. The calculated SAD is 12 bit.

Compressor:

Compressor consists of full adders. As the pixel size is 8 bits hence the number of full adders required is 8. A more comprehensive fault model, i.e. the stuck-at (SA) model, must be adopted to cover actual failures in the interconnect data bus between PEs. The SA fault in a ME architecture can incur errors in computing SAD values. A distorted computational error and the magnitude of are assumed here to be equal to, where denotes the computed SAD value with SA faults.

Test Code Generator:

TCG is an important component of the proposed EDDR architecture. Notably, TCG design is based on the ability of the RQCG circuit to generate corresponding test codes in order to detect errors and recover data. The specific estimates the absolute difference between the Current_ pixel of the search area and the Reference_ pixel of the current macro block. Thus, by utilizing PEs, SAD shown in as follows, in a macro block with size of N × N can be evaluated:

$$\begin{aligned} SAD &= \sum_{i=0}^{N-1}\sum_{j=0}^{N-1}\left|X_{ij}-Y_{ij}\right| \\ &= \sum_{i=0}^{N-1}\sum_{j=0}^{N-1}\left|\left(q_{xij}.m+r_{xij}\right)-\left(q_{yij}.m+r_{yij}\right)\right| \end{aligned} \tag{4}$$

Where r_{xij},q_{xij} and r_{yij},q_{yij} denote the corresponding RQ code of X_{ij} and Y_{ij} modulo m. Importantly, X_{ij} and Y_{ij} represent the luminance pixel value of Cur_pixel and Ref_pixel, respectively.Based on the residue code, the definitions shown in (2) and (3) can be applied to facilitate generation of the RQ code and form TCG. Namely, the circuit design of TCG can be easily achieved by using remainder is computed as follows in hardware design.

$$\begin{aligned} R_T &= \left|\sum_{i=0}^{N-1}\sum_{j=0}^{N-1}\left(X_{ij}-Y_{ij}\right)\right| \\ &= \left|\left|\left(X_{00}-Y_{00}\right)\right|_m+\left|\left(X_{01}-Y_{01}\right)\right|_m+\ldots \right. \\ &\quad \left.+\left|\left(X_{(N-1)(N-1)}-Y_{(N-1)(N-1)}\right)\right|_m\right|_m \\ &= \left|\left|\left(q_{x00}.m-r_{x00}\right)\right|_m-\left|\left(q_{y00}.m+r_{y00}\right)\right|_m\right|_m \\ &\quad +\ldots\left|\left(q_{x(N-1)(N-1)}.m+r_{x(N-1)(n-1)}\right)\right| \\ &\quad \left.-\left(q_{y(N-1)(N-1)}.m+r_{y(N-1)(N-1)}\right)\right|_m\Big|_m \\ &= \left|\left|\left(r_{x00}-r_{y00}\right)\right|_m-\left|\left(r_{x01}-r_{y01}\right)\right|_m+\ldots \right. \\ &\quad \left.+\left|\left(r_{x(N-1)(n-1)}-r_{y(N-1)(n-1)}\right)\right|_m\right|_m \\ &= \left|\left|r_{00}\right|_m+\left|r_{01}\right|_m+\ldots+\left|r_{(N-1)(N-1)}\right|_m\right|_m \end{aligned} \tag{5}$$

The quotient is computed as follows:

$$\begin{aligned} Q_T &= \left\lfloor\frac{\sum_{i=0}^{N-1}\sum_{j=0}^{N-1}\left(X_{ij}-Y_{ij}\right)}{m}\right\rfloor \\ &= \left\lfloor\frac{\left(X_{00}-Y_{00}\right)+\left(X_{01}-Y_{01}\right)+\ldots..+\left(X_{(N-1)(N-1)}-Y_{(N-1)(N-1)}\right)}{m}\right\rfloor \\ &= \left\lfloor\frac{\left(q_{x00}.m-q_{y00}.m\right)}{m}+\frac{\left(r_{x00}-r_{y00}\right)}{m}+\frac{\left(q_{x01}.m-q_{y01}.m\right)}{m}+\frac{\left(r_{x01}-r_{y01}\right)}{m}+\ldots\right\rfloor \\ &= \left\lfloor\left(q_{x00}-q_{y00}\right)+\left(q_{x01}-q_{y01}\right)+\ldots+\frac{\left(r_{x00}-r_{y00}\right)+\left(r_{x01}-r_{y01}\right)+\ldots}{m}\right\rfloor \\ &= q_{00}+q_{01}+\ldots+q_{(N-1)(N-1)}+\left\lfloor\frac{r_{00}+r_{01}+\ldots+r_{(N-1)(N-1)}}{m}\right\rfloor \end{aligned} \tag{6}$$

EDDR Processes:

EDC, which is utilized to compare the outputs between TCG and in order to determine whether errors have occurred. If the values of and/or, then the errors in a specific can be detected. The EDC output is then used to generate a 0/1 signal to indicate that the tested is error-free/errancy. This work presents a mathematical statement to verify the operations of error detection. Based on the definition of the fault model, the SAD value is influenced if either SA1 and/or SA0 errors have occurred in a specific. In other words, the SAD value is transformed to if an error occurred. Notably, the error signal is expressed as

$$e = q_e.m + r_e \qquad (7)$$

to comply with the definition of RQ code. Under the faulty case, the RQ code from of the TCG is still equal to (5) and (6). The error in a specific can be detected if and only if (5) ≠(8) and/or (6)≠ (9):

$$\begin{aligned} R_{PE_i} &= \left|SAD'\right|_m \\ &= \left|\sum_{i=0}^{N-1}\sum_{j=0}^{N-1}\left(X_{ij} - Y_{ij}\right) + e\right| \qquad (8) \\ &= \left|\left|r_{00}\right|_m + \left|r_{01}\right|_m + \ldots + \left|r_{(N-1)(N-1)}\right|_m + \left|r_e\right|_m\right|_m \end{aligned}$$

Simulation Results:

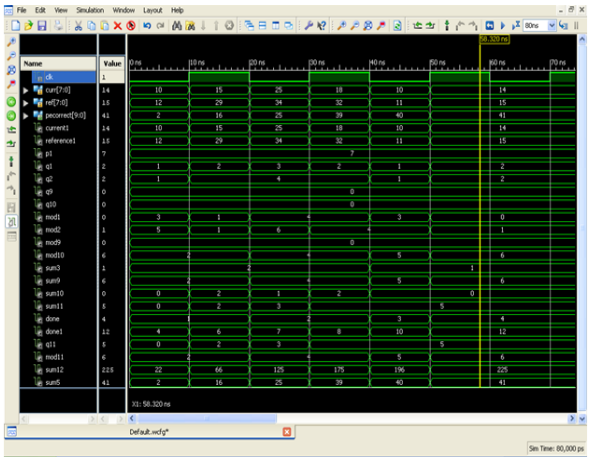

Fig. 6: Simulation Waveform at all the nodes

The Functional-simulation has been successfully carried out with the results matching with expected ones. The design functional verification and Synthesis is done by using Xilinx-ISE/XST and Cadence RTL Compiler of BISDC architecture for MECA.

Conclusion:

This project proposes BISDC architecture for self-detection and self-correction of errors of PEs in an MECA. Based on the error detection correction concepts of bi residue codes, this paper presents the corresponding definitions used in designing the BISD and BISC circuits to achieve self-detection and self-correction operations. Performance evaluation reveals that the proposed BISDC architecture effectively achieves self-detection and self-correction capabilities with minimal area.

About Authors:

Pavan Srinivas is a P.G student of Department of ECE, Koushik college of Engineering, Visakapatnam, Andhra Pradesh, India. And D. P. Raju is presently working as Assistant Professor in the Department of ECE, Koushik college of Engineering, Visakapatnam, Andhra Pradesh, India.

References:

1. Chun-lung Hsu, chang-Hsin Cheng, and Yu Liu, "Built- in self-detection/correction Architecture for Motion Estimation Computing Arrays", IEEE Transcations on Very Large Scale Integration (VLSI) systems, VOL.18, NO.2, February 2010, pp.319-324.
2. Thammavarapu R.N Rao, Member, IEEE, "Biresidue Error-Correcting Codes for Computer Arithmetic", IEEE Transactions on computers, VOL. C-19, NO. 5, May 1970, pp.398-402.
3. Meihua GU, Ningmei YU, Lei ZHU, Wenhua JIA, "High Throughput and Cost Efficient VLSI Architecture of Integer Motion Estimation for H.264/AVC", Journal of Computational Information Systems 7:4 (2011), pp.1310-1318.
4. Zhong-Li He, Chi-Ying Tsui, Member, IEEE, Kai-Keung Chan, and Ming L. Liou, Fellow, IEEE, "Low-Power VLSI Design for Motion Estimation Using Adaptive Pixel Truncation", IEEE Transactions on circuits and systems for video technology, VOL.10, NO.5, August 2000, pp.669- 677.
5. R. J. Higgs and J. F. Humphreys, "Two-error-location for quadratic residue codes," Proc. Inst. Electr.Eng. Commun, vol. 149, no. 3, Jun.2002, pp.129–131.