Testing architecture for motion estimation in video coding systems

R. Rukmani
PG Scholar, Department of VLSI design,
Sri Ramakrishna Engineering College,
Coimbatore, India.

Dr. M. Jagadeeswari
Professor and Head, Department of VLSI Design,
Sri Ramakrishna Engineering College,
Coimbatore, India.

Abstract— **Motion Estimation (ME) is the critical part of any video coding system. The video quality will be affected if an error has occurred in the ME. Motion Estimation is the process of determining the motion vectors that describe the transformation from one 2D image to other, usually from adjacent frames in a video sequence. In order to test Motion Estimation in a video coding system Error Detection and Correction Architecture (EDCA) is designed based on the Residue-and-Quotient (RQ) code. Multiple bit errors in the processing element (PE) i.e., key component of a ME, can be detected and the original data can be recovered effectively by using EDCA design. Experimental results indicate that the proposed design can detect multiple bit errors and effectively recover the data with reduced area and power. The area of the proposed architecture is reduced from 1152 to 114 and the power is reduced from 143.28mW to 59.39mW.**

Keyword- **Processing Element, Error detection, Error correction, Motion estimation, Residue-and-quotient (RQ) code.**

I. INTRODUCTION

Multimedia applications are becoming more flexible and reliable with the advancements in semiconductors, Digital Signal Processing and Communication technologies. Some of the Video Compression standards include MPEG-1, MPEG-2, and MPEG-4. The advanced Video Coding standard is MPEG-4. Video compression is essential in various applications to reduce the total amount of data required for transmitting or storing the video data. ME is of priority concern in removing the temporal redundancy between the successive frames in a video coding system and also it consumes more time. ME is considered as the intensive unit in terms of computation [1].

Regular arrangement of PEs with size 4x4 constitutes a ME. Advancements in VLSI technologies facilitate the integration of large number of PEs into a single chip. Large number of PEs arranged as an array helps in accelerating the computation speed. Testing of PEs is essential as an error in PE affects the video quality and signal-to-noise ratio. Numerous PEs in a ME can be tested concurrently using Concurrent Error Detection (CED) methods [8]. In this method, different operations are performed on the same operand. An error is detected by the conflicting results produced by the operations performed. Concurrent fault simulation is essentially an event-driven simulation with the fault-free circuit and faulty circuits simulated altogether [6].

Design for Testability (DFT) techniques are required in order to improve the quality and reduce the test cost of the digital circuit, while at the same time simplifying the test, debug and diagnose tasks [3].Logic built-in self-test (BIST) is a design for testability technique in which a portion of a circuit on a chip, board, or system is used to test the digital logic circuit itself [4]. BIST technique for testing logic circuits can be online or offline. Online BIST is performed when the functional circuitry is in normal operational mode. In concurrent online BIST, testing is conducted simultaneously during normal functional operation. The functional circuitry is usually implemented with coding techniques [5].

Any input pattern or sequence of input patterns that produces a different output response in a faulty circuit from that of the fault-free circuit is a test vector, or sequence of test vectors, which will detect the faults. The goal of test generation is to find an efficient set of test vectors that detects all faults considered for that circuit. Because a given set of test vectors is usually capable of detecting many faults in a circuit, fault simulation is typically used to evaluate the fault coverage obtained by that set of test vectors.

Because of the diversity of VLSI defects, it is difficult to generate tests for real defects. Fault models are necessary for generating and evaluating a set of test vectors. Generally, a good fault model should satisfy two criteria: (1) It should accurately reflect the behavior of defects, and (2) it should be computationally efficient in terms of fault simulation and test pattern generation.

The rest of this paper is organized as follows. Section II introduces the proposed EDCA, fault model definition, blocks in the architecture and test method. Next, Section III evaluates the performance in terms of area, delay to demonstrate the feasibility of the proposed EDCA for ME testing applications. Conclusions are finally drawn in Section IV.

II. PROPOSED EDCA DESIGN

The proposed EDCA scheme shown in Fig. 1 consists of two major blocks, i.e. error detection circuit (EDC) and data recovery circuit (DRC), to detect the errors and recover the corresponding data in a specific CUT [9]. The test code generator (TCG) in the architecture utilizes the concepts of RQ code to generate the corresponding test codes for error detection and data recovery.

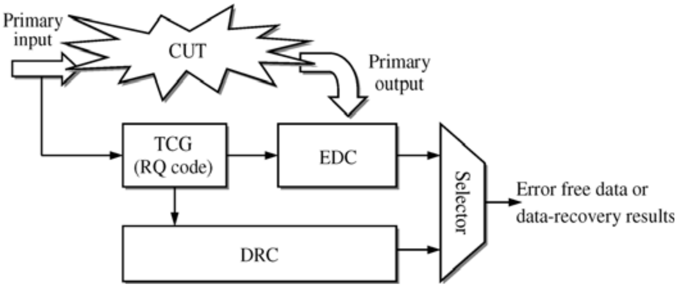

Fig.1.Conceptual View of the proposed EDCA design

The output from the circuit under test is compared with the test code values in the EDC. The output of EDC indicates the occurrence of error. DRC is in charge of recovering data from TCG. Additionally, a selector is enabled to select the error-free data or data-recovery results.

A ME consists of many PE's incorporated in a 1-D or 2-D array for video encoding applications. A PE generally consists of two ADDs (i.e. an 8-b ADD and a 12-b ADD) and an accumulator (ACC) [2]. Next, the 8-b ADD (a pixel has 8-b data) is used to estimate the addition of the current pixel (cur_pixel) and reference pixel (Ref_pixel). Additionally, a 12-b ADD and an ACC are required to accumulate the results from the 8-b ADD in order to determine the sum of absolute difference (SAD) value for video encoding application. Fig. 2 shows the proposed EDCA circuit design for a specific PE_i of a ME [9]. This architecture consists of blocks that generate the residue and quotient values that are used to detect the errors.

A. Fault Model

A more practical approach is to select specific test patterns based on circuit structural information and a set of fault models. This approach is called structural testing. Structural testing saves time and improves test efficiency.

A stuck-at fault affects the state of logic signals on lines in a logic circuit, including primary inputs (PIs), primary outputs (POs), internal gate inputs and outputs, fanout stems (sources), and fanout branches. A stuck-at fault transforms the correct value on the faulty signal line to appear to be stuck at a constant logic value, either logic 0 or logic 1, referred to as stuck-at-0 (SA0) or stuck-at-1 (SA1), respectively. The stuck-at fault model can also be applied to sequential circuits; however, high fault coverage test generation for sequential circuits is much more difficult than for combinational circuits. 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 [7]. A distorted computational error and the magnitude of *e* are assumed here to be equal to SAD′- SAD where SAD ′ denotes the computed SAD value with SA faults.

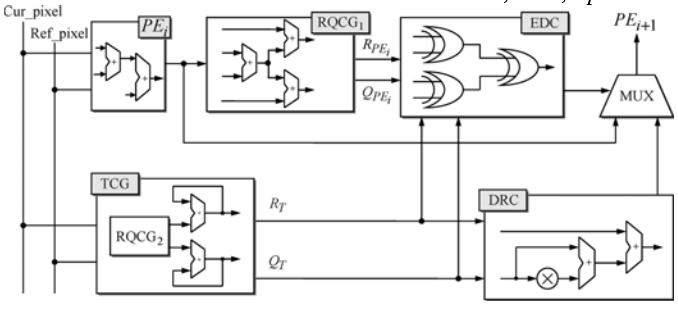

Fig.2. Proposed EDCA design

B. Residue-And-Quotient Code Generation

Earlier codes like Parity Codes, Berger Codes were used for detecting a single bit error. The residue codes evolved next which is generally a separable arithmetic code that estimates the residue of the given data and appends it to the data [10]. This code is capable of detecting a single bit error. Error detection logic using residue codes are simple and it can be easily implemented. For instance, assume that *N* denotes an integer, N_1 and N_2 represent data words, and *m* refers to the modulus. A separate residue code is one in which *N* is coded as a pair $(N, |N|_m)$. Notably, $|N|_m$ is the residue of *N* modulo *m*. However, only a bit error can be detected based on the residue code. Additionally, error correction is not possible by using the residue codes. Therefore, this work presents a quotient code to assist the residue code in detecting multiple bit errors and recovering errors. The mathematical model of RQ code is simply described as follows. Assume that binary data *X* is expressed as

$$X = \{b_{n-1}b_{n-2}......b_2b_1b_0\} = \sum_{j=0}^{n-1} b_j 2^j \qquad (1)$$

The RQ code for *X* modulo *m* is expressed as $R = |X|_m$ and $Q = \left\lfloor X/m \right\rfloor$, respectively [9]. Notably, $\lfloor i \rfloor$ denotes the largest integer not exceeding *i*.

According to the above RQ code expression, the corresponding circuit design of the RQCG can be realized. In order to simplify the complexity of circuit design, the implementation of the module is generally dependent on the addition operation. Additionally, based on the concept of residue code, the following definitions shown can be applied to generate the RQ code for circuit design.

Definition 1:

$$|N_1 + N_2|_m = \left||N_1|_m + |N_2|_m\right|_m \qquad (2)$$

Definition 2: Let, $N_j = n_1 + n_2 + ... + n_j$, then

$$|N_j|_m = \left||n_1|_m + |n_2|_m + ... + |n_j|_m\right|_m \qquad (3)$$

To accelerate the circuit design of RQCG, the binary data shown in (1) can generally be divided into two parts:

$$X = \sum_{j=0}^{n-1} b_j 2^j$$

$$= \left(\sum_{j=0}^{k-1} b_j 2^j \right) + \left(\sum_{j=k}^{n-1} b_j 2^{j-k} \right) 2^k$$

$$= Y_0 + Y_1 2^k \qquad (4)$$

Significantly, the value of *k* is equal to $\left\lfloor {}^{n}\!/_{2} \right\rfloor$ and the data formation of Y_0 and Y_1 are a decimal system. If the modulus $m = 2^k - 1$, then the residue code of *X* modulo *m* is given by,

$$R = \left| X \right|_m$$

$$= \left| Y_0 + Y_1 \right|_m = \left| Z_0 + Z_1 \right|_m = (Z_0 + Z_1)\alpha \qquad (5)$$

$$Q = \left\lfloor \frac{X}{m} \right\rfloor$$

$$= \left\lfloor \frac{Y_0 + Y_1}{m} \right\rfloor + Y_1 = \left\lfloor \frac{Z_0 + Z_1}{m} \right\rfloor + Z_1 + Y_1$$

$$= Z_1 + Y_1 + \beta \qquad (6)$$

where,

$$\alpha(\beta) = \begin{cases} 0(1), & \text{if } Z_0 + Z_1 = m \\ 1(0), & \text{if } Z_0 + Z_1 < m \end{cases}$$

Since the value of Y_0 and Y_1 is generally greater than that of modulus *m* , the equations in (5) and (6) must be simplified further to replace the complex module operation with a simple addition operation by using the parameters Z_0 ,Z_1,α and β [10].

Based on (5) and (6), the corresponding circuit design of the RQCG is easily realized by using the simple adders (ADDs). Namely, the RQ code can be generated with a low complexity and little hardware cost.

C. Test Code Generation

TCG is the main block of the proposed EDCA design. Test Code Generation (TCG) design is based on the ability of the RQCG circuit to generate corresponding test codes in order to detect errors and recovers data. The specific PE_i estimates the absolute difference between the Cur_pixel of the search area and the Ref_pixel of the current macroblock. The SAD value for the macroblock with size of *N*x*N* can be evaluated:

$$SAD = \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} \left| X_{ij} - Y_{ij} \right|$$

$$= \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} \left| (q_{xij}.m + r_{ij}) - (q_{yij}.m + r_{ij}) \right| \qquad (7)$$

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} represents the luminance pixel value of Cur_pixel and Ref_pixel, respectively [9].

Based on the definitions shown in (2) and (3) the generation of the RQ code (R_T and Q_T) from the TCG block is obtained. The circuit design of TCG is achieved by equations (8) and (9),

$$R_T = \left| \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} (X_{ij} - Y_{ij}) \right|_m$$

$$= \left\| X_{00} - Y_{00}) \right|_m + \left| (X_{01} - Y_{01}) \right|_m + \ldots + \left| (X_{(N-1)(N-1)} - Y_{(N-1)(N-1)}) \right|_m \Big|_m$$

$$= \left\| (q_{x00}.m + r_{x00}) - (q_{y00}.m + r_{y00}) \right|_m + \ldots + \Big| (q_{x(N-1)(N-1)}.m + r_{x(N-1)(N-1)})$$

$$- (q_{y(N-1)(N-1)}.m + r_{y(N-1)(N-1)}) \Big|_m \Big|_m$$

$$= \left\| (r_{x00} - r_{y00}) \right|_m + \left| (r_{x01} - r_{y01}) \right|_m + \ldots + \left| r_{x(N-1)(N-1)} - r_{y(N-1)(N-1)}) \right|_m \Big|_m$$

$$= \left\| r_{00} \right|_m + \left| r_{01} \right|_m + \ldots + \left| r_{(N-1)(N-1)} \right|_m \Big|_m \qquad (8)$$

$$Q_T = \left\lfloor \frac{\sum_{j=0}^{N-1} \sum_{j=0}^{N-1} (X_{ij} - Y_{ij})}{m} \right\rfloor$$

$$= \left\lfloor \frac{(X_{00} - Y_{00}) + (X_{01} - Y_{01}) + \ldots + (X_{(N-1)(N-1)} - Y_{(N-1)(N-1)})}{m} \right\rfloor$$

$$= \left\lfloor \frac{(q_{x00}.m - q_{y00}.m)}{m} + \frac{(r_{x00} - r_{y00})}{m} + \frac{(q_{x01}.m - q_{y01}.m)}{m} + \frac{(r_{x01} - r_{y01})}{m} + \ldots \right\rfloor$$

$$= \left\lfloor (q_{x00} - q_{y00}) + (q_{x01} - q_{y01}) + \ldots + \frac{(r_{x00} - r_{y00}) + (r_{x01} - r_{y01}) + \ldots}{m} \right\rfloor$$

$$= q_{00} + q_{01} + \ldots + q_{(N-1)(N-1)} + \left\lfloor \frac{(r_{00} + r_{01} + \ldots + r_{(N-1)(N-1)})}{m} \right\rfloor \qquad (9)$$

D. Error Detection Process

Error Detection Circuit (EDC) is used to perform the operation of error detection in the specific PE_i as shown in Fig.2. This block is used to compare the output from the TCG i.e., (R_T and Q_T) with output from RQCG1 i.e., (R_{PEi} and Q_{PEi}), in order to detect the occurrence of an error. If the value $R_{PEi} \neq R_T$ and $Q_{PEi} \neq Q_T$, then the error in the PE_i can be detected.

The EDC output is then generated as a 0/1 signal to indicate that the tested PE_i is error free/ with error.

E. Error Correction Process

The original data is recovered in the Data Recovery Circuit (DRC) during error correction process, by separating the RQ code from the TCG. The data recovery is possible by implementing the mathematical model as,

$$\begin{aligned} SAD &= m \times Q_T + R_T \\ &= (2^j - 1) \times Q_T + R_T \\ &= 2^j \times Q_T - Q_T + R_T \qquad (10) \end{aligned}$$

The proposed EDCA design executes the error detection and data recovery operations simultaneously. Additionally, error-free data from the tested PE_i or the data recovery that results from DRC is selected by a multiplexer (MUX) to pass to the next specific PE_{i+1} for subsequent testing.

	0	1	2	3
0	128	128	64	255
1	128	64	255	64
2	64	255	64	128
3	255	64	128	128

Cur_pixel

	0	1	2	3
0	1	1	2	3
1	1	2	3	4
2	2	3	4	5
3	3	4	5	5

Ref_pixel

Fig.3. Example of pixel values

F. Sample calculation

A sample of the 16 pixels for a 4x4 macroblock is taken. Fig. 3 presents an example of pixel values of the Cur_pixel and Ref_pixel [9]. Based on (7), the SAD value of the 4x4 macroblock is

$$\begin{aligned} SAD &= \sum_{i=0}^{3}\sum_{j=0}^{3}\left|X_{ij} - Y_{ij}\right| \\ &= \left|X_{00} - Y_{00}\right| + \left|X_{01} - Y_{01}\right| + ... + \left|X_{33} - Y_{33}\right| \\ &= (128-1) + (128-1) + ... + (128-5) \\ &= 2124 \qquad (11) \end{aligned}$$

The modulo is assumed here to be m=2^6-1=63. Thus, based on (8) and (9), the RQ code of the SAD value shown in (11) are R_{PEi} = R_T = $|2124|_{63}$ =45 and Q_{PEi} = Q_T = $\lfloor 2124/63 \rfloor$ =33. Since the value of R_{PEi} (Q_{PEi}) is equal to R_T(Q_T), EDC is enabled and a signal "0" is generated to describe a situation in which the specific is error-free. Conversely, if SA1 and SA0 errors occur in bits 1 and 12 of a specific PE_i, i.e., the pixel values of PE_i , $2124=100001001100_2$ is turned into $77=000001001101_2$, resulting in the transformation of the RQ code of R_{PEi} and Q_{PEi} into |77|=14 and $\lfloor 77/63 \rfloor$=1. Thus, an error signal "1" is generated from EDC and sent to the MUX in order to select the recovery results from DRC.

III. Results and Discussions

Residue-and-quotient code is used in the existing error detection architecture that has 16 Processing Elements (PEs) and 16 Test Code Generation (TCG) blocks for computing the test codes for each pixel value in the macroblock [10]. Though this architecture can detect multiple bit errors, the area of the architecture is high and the overall delay is high. To overcome this disadvantage, the proposed architecture is used which has a single PE and a TCG for computing the test codes for the pixel values.

Verification of the proposed design is performed by using VHDL and then synthesized by using Mentor Graphics with the target device as EP2S180F1508C from the family Stratix II. The performance of the proposed EDCA design is estimated in terms of its area and power. In this architecture, the pixel values in the 4x4 macroblock are taken out for the code generation based on the clocking signal. At each triggering edge of the clock the 8-bit pixel value from the ref_pixel and cur_pixel is taken and given to the TCG and PE blocks for code generation. The generated code from test code generation block and the RQCG output is compared in the EDC block, output of which indicates the error. As the code generation for each 8 bit pixel value is done based on the clock signal, the need for separate PE and TCG for each 16 elements of the 4x4 macroblock is avoided. By this the overall area and power of the proposed architecture is reduced.

TABLE I
Comparison of Area and Power

PARAMETERS	EXISTING ARCHITECTURE	PROPOSED ARCHITECTURE
AREA (No. of LUTs)	1152	114
POWER (mW)	143.28	59.39

Table I shows that the area and power are reduced by using a single PE architecture with a triggering clock instead of using 16 PEs and 16 TCGs that are employed for each pixel value of the 4x4 macroblock.

The overall test strategy is based on the RQ code generation which can detect multiple bit errors at a reduced power and very less area. This reduction in area can be utilized for VLSI circuit approach to design the proposed architecture in an area efficient way. By using a single clock triggering, the number of macro blocks can be extended with reduced area for testing the PEs of a ME.

IV. Conclusion

The motion estimation process is done by the video coding system to find the motion vector pointing to the best prediction macroblock in a reference frame or field. An error occurring in

ME can cause degradation in the video quality. In order to make the coding system efficient an Error Detection and Correction Architecture (EDCA) is designed for detecting the errors and recovering the data of the PEs in a ME. Based on the RQ code, a RQCG-based TCG design is developed to generate the corresponding test codes that can detect multiple bit errors and recover data. The proposed EDCA design is implemented using VHDL and synthesized by using Mentor Graphics with the target device as EP2S180F1508C from the family Stratix II. Experimental results indicate that the proposed EDCA design can effectively detect the errors and recover data in PEs of a ME with reduced area and power. The area of the proposed architecture is reduced from 1152 to 114 and the power is reduced from 143.28mW to 59.39mW.

ACKNOWLEDGEMENT

The authors would like to thank All India Council for Technical Education (AICTE), India, to financially support this work under the Grant 8023/BOR/RID/ RPS-48/2009-10. The authors would also like to thank the Management and Principal of Sri Ramakrishna Engineering College, Coimbatore for providing excellent computing facility and encouragement.

REFERENCES

[1] Y.W.Huang, B.Y.Hsieh,S.Y. Chien, S.Y. Ma, and L.G. Chen, “Analysis and complexity reduction of multiple reference frames motion estimation in H.264/AVC,” IEEE Trans. Circuits Syst.Video Technol., Vol.16,no.4,pp.507-522,Apr.2006.

[2] S.Surin and Y.H. Hu,”Frame-level pipeline motion estimation array processor,” IEEE Trans. Circuits Syst. Video Technol., vol. 11, no.2, pp.248-251, Feb.2001.

[3] M.Y.Dong, S.H.Yang, and S.K. Lu,” Design-for-testability techniques for motion estimation computing arrays,” in Proc. Int. Conf. Commun., Circuits Syst., May 2008, pp.1188-1191.

[4] J.F. Lin, J.C. Yeh, R.F. Hung, and C.W. Wu, “A built-in self repair design for RAMs with 2-D redundancy,” IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol.13, no. 6, pp.742-745, Jun. 2005.

[5] C.L. Hsu, C.H. Cheng, and Y. Liu,” Built-in self detection/correction architecture for motion estimation computing arrays,” IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 18, no.2, pp. 319-324, Feb.2010.

[6] S.Bayat-Sarmadi and M.A. Hasan,”On concurrent detection of errors in polynomial basis multiplication,” IEEE Trans. Very Large Scale Integr. (VLSI) Systs., vol. 15, no.4, pp. 413-426, Apr.2007.

[7] D. K. Park, H.M.Cho, S.B. Cho, and J.H. Lee, “A fast motion estimation algorithm for SAD optimization in sub-pixel,” Proc. Int. Symp. Integr. Circuits, Sep. 2007, pp. 528-531.

[8] C.W. Chiou, C.C. Chang, C.Y. Lee, T.W. Hou, and J.M. Lin,” Concurrent error detection and correction in Gaussian normal basis multiplier over GF (2^m), IEEE Trans. Comput., vol.58, no.6, pp.851-857, Jun.2009.

[9] Chang- Hsin Cheng, Yu Liu, and Chun-Lung Hsu, Member, IEEE “ Design of an Error Detection and Data recovery Architecture for Motion Estimation Testing Applications,” IEEE transactions on Very Large scale integration (VLSI) systems, vol.20, no.4, April.2012.

[10] L.Breveglieri, P. Maistri, and I. Koren, “ Anote on error detection in an RSA architecture by means of residue codes,” in Proc. IEEE Int. Symp. On-Line Testing, Jul. 2006, 176 177.